A UNIVERSAL MULTI-DIMENSIONAL LOSSY COMPRESSION ALGORITHM

Murilo B. de Carvalho

Depto. de Eng. de Telecomunicações
Universidade Federal Fluminense
R. Passos da Pátria, 156
Niteroi - RJ
24210-240, BRAZIL
murilo@lps.ufrj.br

Eduardo A. B. da Silva

PEE/COPPE/DEL/EE
Universidade Federal do Rio de Janeiro
Cx. P. 68504,
Rio de Janeiro, RJ
21945-970, BRAZIL
eduardo@lps.ufrj.br

ABSTRACT

A universal algorithm to compress multi-dimensional data is presented. Besides being able to explore multi-dimensional correlations of the data, it incorporates two other fundamental innovations when compared to its predecessors, the Lossy Lempel-Ziv (LLZ) and Hierarchical String Matching algorithms: first, instead of building a dictionary of strings to match the data, it builds a dictionary of *basis functions* in which the data will be decomposed in the spirit of Mallat's Matching Pursuits; second, any basis function in the dictionary can be *dilated* or *contracted* when used to match the data. Simulation results show that it has good coding performance for a large class of image data. With Gaussian sources it has shown good performance, outperforming LLZ algorithms for low data rates, being close to the R(D). With real image data, when LLZ fails at all rates, it has performed even better, showing a great improvement over LLZ.

1. INTRODUCTION

The problem of compressing data from a source subject to a fidelity criterion has been actively studied in the information theory community for the last 50 years. Research on rate-distortion theory [1] provided some useful performance bounds, usually for very simple problems with simple statistical models. However, a general solution for an optimal compression algorithm has not been found yet. Some solutions are known for the problem of compressing with zero distortion. The Lempel-Ziv (LZ) [2] algorithm for instance, is asymptotically optimal, in the sense that for large blocks of input data the rate of the code approaches the theoretical minimum. A lossy version of this algorithm, the Lossy-Lempel-Ziv (LLZ) [3], is also optimal for zero distortion but is sub-optimal [4] at high distortion. These algorithms are universal in the sense that no prior knowledge of source statistics is needed. In other words, they adapt themselves to the characteristics of the data to code. Nevertheless, for high-distortion, and consequently low-rate, a universal algorithm for compression is unknown. At these rates, many successful compression algorithms make some kind of transformation on the data prior to a lossy compression scheme. In fact, it has recently been suggested that there may be a connection between optimal lossy compression techniques and harmonic analysis [5]. For example, for image compression, there are the EZW [7], SPIHT [8] and the old but popular JPEG [6] coder. However, even these algorithms that are built to deal with a restricted class of sources, will benefit from some kind of global optimization [9]. Therefore, for low rates, it seems to be a good strategy to build a coding scheme that performs something like an adaptive transform of the data to code. The remaining of this paper will describe a new class of such algorithms. In it, the data is decomposed in a basis that is built as the data is encoded, in a way inspired in the dictionary building in string-matching-based compression algorithms.

2. DESCRIPTION OF THE ALGORITHM

An attempt to build a compression algorithm that adapts itself to the data's most important features will benefit from a study of methods to expand functions over a set of waveforms. A fairly general procedure to decompose a signal in a basis is the Matching Pursuits algorithm [10]. It expands a signal using an dictionary of functions. The dictionary $\mathcal{D} = (g_\gamma)_{\gamma\in\Gamma}$ is composed of vectors in a Hilbert space $\mathbf{H}$ such that $||g_\gamma|| = 1$. It is complete if $\mathbf{V} = \mathbf{H}$, where $\mathbf{V}$ is the closed linear span of the dictionary vectors. The procedure to find a representation of a vector $f \in \mathbf{H}$ is to make successive

0-7803-5467-2/99/ $ 10.00 © 1999 IEEE

approximations of f with orthogonal projections on elements of $\mathcal{D}$. The decomposition begins by choosing the γ_0 that maximizes the inner product $< f, g_{\gamma_0} >$. Then a residue $R^0 f$ is computed as

$$R^0 f = f - < f, g_{\gamma_0} > g_{\gamma_0} \quad (1)$$

This residue $R^0 f$ is then expanded in the same way as f. The procedure continues until some energy threshold for the residue is reached. After M steps the vector f can be approximated as

$$\hat{f} = \sum_{i=0}^{M-1} < f, g_{\gamma_i} > g_{\gamma_i} \quad (2)$$

This procedure has some useful properties. For example, the vectors in the dictionary can be chosen in order to give better spatial resolution than DCT basis and at the same time better frequency resolution than wavelets. Also it has the energy compaction property since it selects the components of greater energy first. In fact the energy compaction will usually increase as the number of vectors in $\mathcal{D}$ increases. The procedure is quite general and any collection of arbitrary shaped vectors can be used as $\mathcal{D}$. The convergence is guaranteed as long as the dictionary set is complete. However, the rate of convergence is affected by the choice of the dictionary. Therefore, some choices will do better in a rate-distortion sense than others.

We have built a compression scheme based on the Matching Pursuits algorithm (UHMP - Universal Hierarchical Matching Pursuits) which builds its own dictionary as it codes the data, instead of using a dictionary $\mathcal{D}$ which is fixed a priory. UHMP starts with an initial dictionary $\mathcal{D}_0$ and an N-dimensional vector f to decompose . Then all the vectors in $\mathcal{D}_0$ are scaled to dimension N and the best approximation $\hat{f} = Q(< f, g_{\gamma_0} >) g_{\gamma_0}$ is selected. The quantization function Q(.) is included to deal with the finite precision of practical implementations. At this moment, the algorithm outputs the dictionary vector index and the quantized inner product value. If the residue's norm is smaller than a target distortion d^* the procedure outputs a one bit flag '1' , returns $\hat{f} = Q(< f, g_{\gamma_0} >) g_{\gamma_0}$ and stops. Otherwise, it outputs a one bit flag '0', the residue vector $R^0 f$ is parsed in two segments $R^0 f'$ and $R^0 f''$ and the parsing point is output. The parsing point is selected to optimize some objective function, for example, it can be chosen to minimize the overall distortion when using the dictionary to expand both segments. Next the procedure is recursively applied to each segment. When the coding of the two segments $R^0 f'$ and $R^0 f''$ is finished, $\hat{R}^0 f'$ and $\hat{R}^0 f''$ will be available and the algorithm can compute $\hat{R}^0 f$ as the concatenation of $\hat{R}^0 f'$ and $\hat{R}^0 f''$. Then it computes $\hat{f} = Q(< f, g_{\gamma_0} >) g_{\gamma_0} + \hat{R}^0 f$, includes $\hat{f}$ and $\hat{R}^0 f$ in the dictionary, returns $\hat{f}$ and stops. Decoding is as follows: The decoding algorithm starts with the same initial dictionary and vector dimension as the encoder. Then it receives a dictionary index and a quantized inner product value. It then computes $\hat{f} = Q(< f, g_{\gamma_0} >) g_{\gamma_0}$. Next it receives a flag bit. If it's a one, the decoder returns $\hat{f}$ and stops. Otherwise, the decoder reads the parsing point and the decoding procedure is recursively applied to recover $\hat{R}^0 f'$ and $\hat{R}^0 f''$. Then the decoder finds $\hat{R}^0 f$ as the concatenation of $\hat{R}^0 f'$ and $\hat{R}^0 f''$, evaluates $\hat{f} = Q(< f, g_{\gamma_0} >) g_{\gamma_0} + \hat{R}^0 f$, includes $\hat{f}$ and $\hat{R}^0 f$ in the dictionary, returns $\hat{f}$ and stops So long, we assumed that the dictionary vectors are normalized. If they're not, the best approximation becomes $\hat{f} = Q(\frac{< f, g_{\gamma_0} >}{||g_{\gamma_0}||^2}) g_{\gamma_0}$. UHMP is easily extendable to higher dimensions. A two-dimensional version of it can be described as follows:

Let:

- $S = (s_{ij})$, $i \in [1, M], j \in [1, N], s_{ij} \in \mathbb{R}$ be an $M \times N$ array of two-dimensional data;
- $\mathcal{D} = (G_\gamma)$, $\gamma \in [0, q-1]$ be an initial dictionary of matrixes of arbitrary dimensions $G_\gamma = (g_{\gamma i})$, $i, j \in I_\gamma$;
- $T_{mn}(G)$ a transformation that maps an $m' \times n'$ array G into an $m \times n$ array;
- $d(S_1, S_2)$ be the distortion when S_2 is used to replace S_1,
- $J(S, p_l, p_c, \mathcal{D})$, $p_l \in [0, M-1], p_c \in [0, N-1]$ be an objective function, and
- $Q(a)$, $a \in \mathbb{R}$ be a scalar quantizer

The procedure $\hat{S}$ =code$(S, \mathcal{D}, d^*)$ encodes S such that $d(S, \hat{S}) \leq d^*$ and is defined as

Procedure $\hat{S}$ =code$(S, \mathcal{D}, d^*)$:

step 1 find γ such that $d(S, \hat{S})$ is minimized, where
`if` $||T_{MN}(g_\gamma)||^2 > 0$
`then` $\hat{S} = Q(< S, \frac{T_{MN}(g_\gamma)}{||T_{MN}(g_\gamma)||^2} >) T_{MN}(g_\gamma)$
`else` $\hat{S} = 0$

step 2 output γ and $Q(< S, \frac{T_{MN}(g_\gamma)}{||T_{MN}(g_\gamma)||^2} >)$

step 3 `if` $d(S, \hat{S}) \leq d^*$ `then` output flag '1' and return $\hat{S}$.
`else` go to step 4.

step 4 Compute the residue $R = S - \hat{S}$

step 5 choose $p_l \in [0, M-1], p_c \in [0, N-1]$ to minimize $J(R, p_l, p_c, \mathcal{D})$ and then

split $R = \begin{pmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{pmatrix}$ where R_{11} is $p_l \times p_c$

step 6 output flag '0', p_l, p_c

step 7 Compute $\hat{R_{ij}} = \text{code}(R_{ij}, \mathcal{D}, d^*)$

step 8 make $\hat{R} = \begin{pmatrix} \hat{R_{11}} & \hat{R_{12}} \\ \hat{R_{21}} & \hat{R_{22}} \end{pmatrix}$, $\hat{S} = \hat{S} + \hat{R}$

step 9 $\mathcal{D} = \mathcal{D} \cup (\hat{R}) \cup (\hat{S})$

step 10 return $\hat{S}$

This algorithm builds iteratively the dictionary set. The elements of the dictionary are obtained from the data itself. For stationary sources, this elements are likely to be typical sequences [1], so the algorithm also performs entropy coding using string matching. It is important to note that, although just a two-dimensional version of the algorithm was described, it can be trivially extended to higher dimensions.

3. IMPLEMENTATION DETAILS

The algorithm described in section 2 was implemented in a computer program for simulation. The transformation $T_{mn}(g)$ used was a classical sampling rate change operation using a linear interpolator as the filter [11]. For example, in the one-dimensional case, to change from size m' to size m we first change to an intermediate size $m'm$ by linear interpolation. Then if $m' < m$ this $m'm$ vector is subsample to size m by retaining only m equally spaced samples. If $m' > m$, the $m'm$ vector is divided in m blocks of size m' and the mean of each block is computed. Each mean value will be a sample of the size m vector. In the two-dimensional case, we first change from size $m' \times n'$ to size $m \times n'$ by applying the one-dimensional procedure described to each column of the $m' \times n'$ array. Then we change to size $m \times n$ by operating on the rows. The objective function $J(R, p_l, p_c, \mathcal{D})$ used was the squared error sum $\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (r_{ij} - r'_{ij})^2$ where r_{ij} are the elements of the array R and r'_{ij} are the elements of the array $R' = \begin{pmatrix} R'_{11} & R'_{12} \\ R'_{21} & R'_{22} \end{pmatrix}$. R'_{ij} are the best approximations to R_{ij} using $\mathcal{D}$ and R_{11} is size $pl \times pc$. The target distortion can be set anywhere in $[0, \infty)$ provided that the quantizer Q has enough resolution, since any nonempty initial dictionary is complete. This is so because for any basis function there is a transformation $T_{mn}(G)$ which maps it into an impulse. The resolution requirement of Q is mild, because the quantization error is explicitly calculated during the expansion. To deal with the finite memory available, a pruning schedule was implemented: Each new element included in the dictionary is associated to a counter that is initially set to zero. When a dictionary element is selected as a "best match", then it's associate counter is set to zero while all other counters are incremented by one. when the dictionary reaches a given size, the element with the greatest counter value is pruned whenever its necessary to include a new element. The dictionary elements maximum size is also limited to make memory management easier. So if an element to be included in the dictionary has a column-row product greater then a given value, it is not included at all. To allow the use of exact $\log_2(|\mathcal{D}|)$ bits for each index , $\log_2(m)$ bits for p_l and $\log_2(n)$ bits for p_c, an arithmetic coder was used. It was assumed a uniform distribution in the $[0, |\mathcal{D}|)$ range for the indexes, a uniform distribution in $[0, m)$ for pl and a uniform distribution in $[0, n)$ for pc. The quantized inner products were coded by an adaptive arithmetic coder.

4. SIMULATION RESULTS

Figure 1 shows simulation results for the image LENA 256×256. The parameters used were: $|\mathcal{D}| \leq 256$ and initially $\mathcal{D} = (0, 32, 64, \ldots, 224)$, quantizer with reconstruction levels $(-1, -0.5, 0, 0.5, 1)$, elements of the dictionary with dimension restricted to $mn \leq 64$. The image was also initially divided in blocks of size 32×32 without overlapping. The distortion metric used is the peak signal-to-noise ratio $PSNR = 10 \log \left(\frac{255^2}{\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (s_{ij} - \hat{s_{ij}})^2} \right)$. The results for EZW, JPEG and LLZ are also shown. Although inferior to the image-optimized EZW and JPEG, the superiority of UHMP over the universal LLZ is clear.

Figure 2 shows a dictionary obtained for LENA at 1 bpp (with $|\mathcal{D}| \leq 64$). It is interesting that the algorithm learned basis functions that look like windowed sinusoids, which have quite a few similarities to the Karhunen-Loève Transform basis functions.

Figure 3 shows the results for a zero mean Gaussian source with variance $\sigma^2 = 957$. Results for EZW, JPEG, LLZ and the theoretical minimum $D(R) = \sigma^2 2^{-2R}$ [1] are also shown. Here, UHMP is the best at low rates, but is inferior to EZW at rates above 0.6 bpp and to LLZ above 0.7 bpp. UHMP performs better than JPEG at all rates. Although UHMP is the best at low rates, there is still plenty of room for

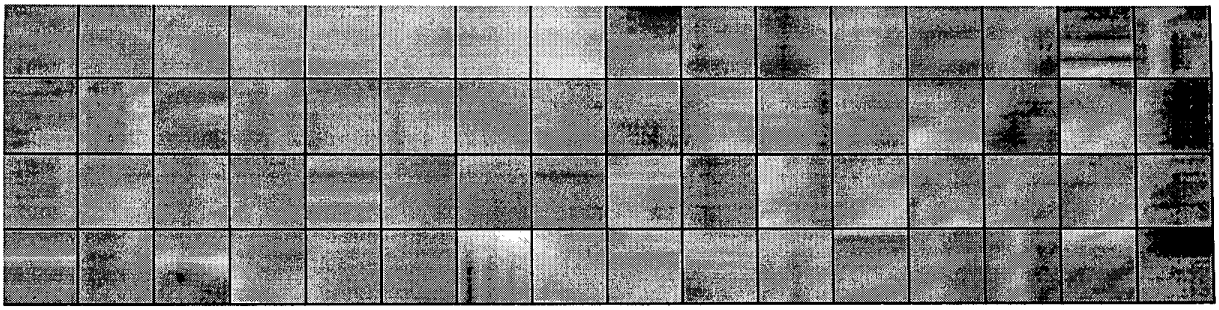

Figure 2: Dictionary for LENA 256 × 256.

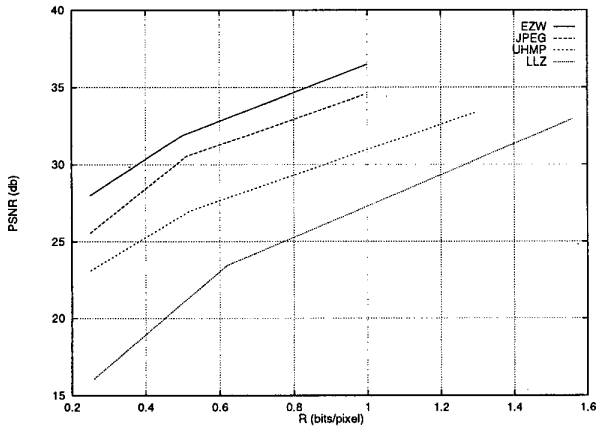

Figure 1: Performance for LENA 256 × 256.

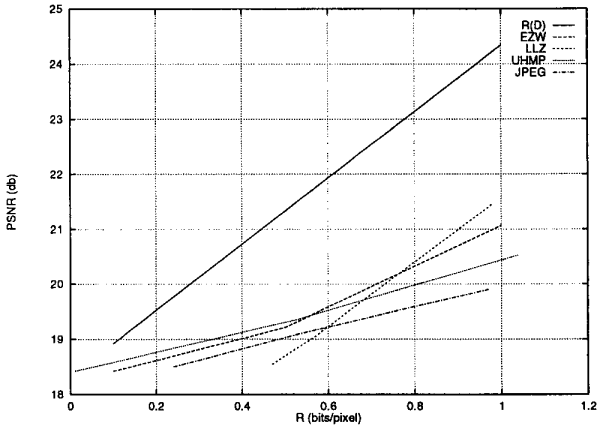

Figure 3: PSNR × rate for Gaussian 256 × 256.

improvement, as shown by the $D(R)$ curve.

5. CONCLUSION

A new universal multi-dimensional data compression algorithm was presented. Unlike its counterparts, the Lossy Lempel-Ziv algorithms [3] and Hierarchical String Matching algorithms [12], it relies on the decomposition of the data in a basis, which is built as the data is compressed. It is important to note that this is different from the Matching Pursuits [10] and Wavelet Packets [13] in which the whole basis is known previously and the algorithm just chooses the best expansion. Simulation results for Gaussian sources have shown that a two-dimensional version of it outperforms LLZ algorithms at low rates. With real image data, when LLZ fails at all rates, it has performed even better, showing a great improvement over LLZ. Also it can be generalized to higher dimensions in a straightforward manner, unlike LLZ. It has good performance with a broad class of data, but improvements are expected if parameters like the objective function $J(R, p, \mathcal{D})$, the transformation $T_{mn}(g)$ and the distortion metric $d(S, \hat{S})$ are tailored to match a narrower class of sources, such as images.

6. REFERENCES

[1] R. E. Blahut, "Principles and Practice of Information Theory" Addison-Wesley publishing Company, 1988.

[2] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," *IEEE Transactions on Information Theory*, vol. it-24, No. 5, pp. 530-536, September 1978.

[3] M. B. de Carvalho and W. A. Finamore, "Lossy Lempel-Ziv on subband coding of images," *IEEE International Symposium of Information Theory*, June 27 - July 1, 1994, Trondheim, Noruega.

[4] T. Łuczak and W. Szpankowski, " A suboptimal lossy date compression based on approximate pat-

tern matching," IEEE Transactions on Information Theory, Vol. 43, No. 5, September 1997.

[5] Donoho D. L., Vetterli M., DeVore R. A. and Daubechies I., " Data compression and harmonic analysis," IEEE Transactions on Information Theory, Vol. 44, No 6, October 1998.

[6] W. Pennebaker and J. Mitchel, "JPEG Still Image Data Compression Standard," Van Nostrand Reinhold, 1994.

[7] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coeficients," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 41, pp. 3445-3462, December 1993.

[8] A. Said and W. A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE transactions on circuits and systems for video technology*, vol. 6, pp. 243–250, June 1996.

[9] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Processing Magazine*, November 1998.

[10] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, No. 12, pp. 3397-3415, December 1993.

[11] P. P. Vaidyanathan, "Multirate Systems and Filter Banks," Prentice-Hall Inc., 1993.

[12] J. C. Kieffer, T. H. Park, Y. Xu, S. J. Yakowitz, "Progressive lossless image coding via self-referential partitions", *1998 IEEE International Conference on Image Processing*, October 1998, Chigago, Illinois.

[13] K. Ramchandran, M. Vetterli and C. Herley, "Wavelets, subband coding and best basis," *Proccedings of the IEEE*, pp.541–560, April 1996.