MULTIDIMENSIONAL SIGNAL COMPRESSION USING MULTI-SCALE RECURRENT PATTERNS WITH SMOOTH SIDE-MATCH CRITERION

Eddie B. L. Filho⋆†*, Murilo B. de Carvalho*‡*, Eduardo A. B. da Silva*†

⋆ Genius Institute of Technology
Av. Açaí, 875, BL. E, Manaus - AM, 69075-904, BRAZIL
† PEE/COPPE/DEL/POLI, Universidade Federal do Rio de Janeiro,
Cx. P. 68504, Rio de Janeiro - RJ, 21945-970, BRAZIL
‡ TET/CTC, Universidade Federal Fluminense,
R. Passos da Pátria, 156, Niteroi, RJ, 24210-240, BRAZIL
efilho@genius.org.br, murilo@telecom.uff.br, eduardo@lps.ufrj.br

ABSTRACT

The recently proposed method for image compression based on multi-scale recurrent patterns, the MMP (Multidimensional Multi-scale Parser) has been shown to perform well for a large class of images, specially for those containing text or graphics. However, its performance for coding smooth, gray scale images was still distant from the state-of-the art. In this paper we propose an extension for it, the SM-MMP (Side-match MMP). In it, as in MMP, a multidimensional signal is recursively segmented into variable-length blocks, and each segment is encoded using expansions and contractions of vectors in a dictionary. The dictionary is updated while the data is being encoded, using concatenations of expanded and contracted versions of previously encoded blocks. However, unlike MMP, in SM-MMP the dictionaries are built considering smoothness constraints around block boundaries, similarly to what happens in side-match vector quantization methods. This allows it to perform better than MMP when the images are smooth, without sacrificing its performance for images containing text or graphics. Indded, our simulation results show that the proposed method is effective, yielding improvements of the order of 1.5 dB over the original MMP for grayscale images, while preserving the high performance of the original MMP for graphics, text and mixed images.

Keywords: *Recurrent Pattern Matching, Multi-scale Decomposition, Multidimensional Signal Compression, Vector Quantization, Side Match*

1. INTRODUCTION

In a recent work [1] we have introduced the Multidimensional Multi-scale Parser algorithm (MMP), a lossy data compression method based on approximate pattern matching with scales. Simulation results have shown that MMP is very effective to compress a variety of image sources. When applied to images containing text or graphics, as well as compound images with text, graphics and grayscale images, the MMP algorithm outperformed other coders representing the state-of-the-art in image compression. However, for pure grayscale images, although its performance was better than the one of some transform based coders, such as the DCT-based JPEG, it did not perform as well as other coders based on the DWT, as the SPIHT [5] and JPEG2000 [4]. This can be explained by the fact that the success of transform-based schemes relies on the supposition that the image data is essentially of a lowpass nature, so that most of their important information will be clustered in the low frequency transform coefficients, leading to efficient encoding schemes for the quantized coefficients. However, this assumption is not true for a large class of image data, as, for example, text and graphics. All the tests were done with grayscale images, but it is easy to extend the algorithm to work with n-dimensional data. Unlike most methods used in image and video compression, the MMP encodes a source without using the transformation-quantization-entropy coding paradigm. In addition, it makes no strong a priori assumption about the signal it attempts to compress. Thus, although it performs well for a large class of image data, the MMP is not as successful to compress grayscale images as it is to compress mixed images when compared to DWT-based coders. Based on the above, one possible way to improve the performance of MMP with grayscale images is to assume some model for grayscale image sources and fine tune the algorithm accordingly. Ideally, this should be done in such a way that the good performance of MMP with text, graphics an mixed image sources is not impaired.

In this paper we propose an extension to MMP that we refer to as SM-MMP (Side Match-Multidimensional Multi-scale Parser). In it, as in the original MMP, a signal is encoded by first segmenting it into variable-length blocks. Each block is encoded using expansions and contractions of vectors from a dictionary. The dictionary is updated as the data is being encoded, with concatenations of expansions and contractions of previously encoded vectors. However, unlike the original MMP, the SM-MMP adopts an underlying model for the source. When attempting to encode a given block, SM-MMP first measures the behavior of the image in a causal neighborhood of the block. Then it builds a sub-dictionary, referred to as *state dictionary*, by selecting from the dictionary only those vectors which meet some smoothness criteria relative to the causal neighborhood. This way, the state dictionary can be of a size ranging from a minimum predefined size up to the size of the complete dictionary. This is equivalent to set the probability of occurrence of those vectors that do not meet the smoothness criteria to zero (they are out of the *state dictionary*), which is a reasonable assumption if the image is smooth. By reducing the dictionary to a state dictionary which contains only the most probable vectors, the SM-MMP has the potential to lower the mean rate while keeping the mean distortion unchanged. Note that this technique is similar to that used in the side-match vector quantization scheme [2].

The remaining of this paper is organized as follows. In section 2 the original version of the MMP algorithm is presented. In section 3, we propose the side match extension, the SM-MMP. Ex-

This work was accomplished through the partnership between UFAM (Universidade Federal do Amazonas) and UFRJ/COPPE, with the financial support provided by SUFRAMA (Superintendencia da Zona Franca de Manaus).

perimental results with image data are presented in section 4 and section 5 presents the conclusions.

2. THE MMP ALGORITHM

The MMP algorithm is a method to lossy compress data using *approximate matching with scales*. This is an extension of ordinary approximate pattern matching, where we allow vectors of different lengths to be matched. In order to do this, we use a scale transformation $T_N^M : \mathbb{R}^M \mapsto \mathbb{R}^N$ to adjust the sizes of the vectors prior to the matching attempt [1]. In the case of image sources, we can use a two-dimensional scale transformation to allow the matching of two matrices of different sizes.

In a two-dimensional MMP, an input matrix $\mathbf{X}$ is parsed in L non-overlapping blocks and each block is represented by a scale-transformed version $\mathbf{S}_i^s$ of a matrix $\mathbf{S}_i$ in a dictionary $\mathcal{D}$. This process is illustrated in figure 1(a).

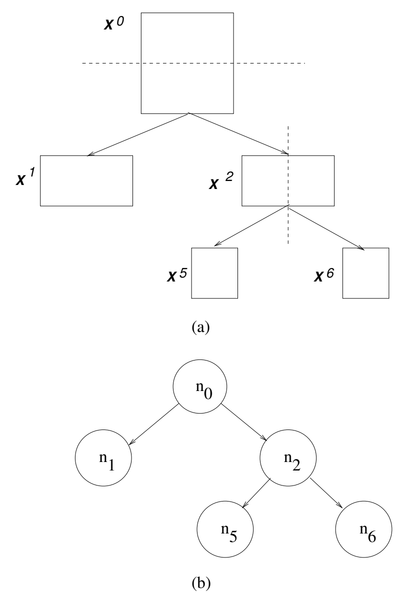

Fig. 1. Segmentation in MMP.

The segmentation of the input matrix $\mathbf{X}$ can be represented by a segmentation tree $\mathcal{S}$, as illustrated in figure 1(b). Each node n_j of $\mathcal{S}$ is associated to a block $\mathbf{X}^j$ of size $\left(2^{-\lfloor \frac{p+1}{2} \rfloor} N \times 2^{-\lfloor \frac{p}{2} \rfloor} N\right)$, where p is the depth of the node n_j in the segmentation tree and $\lfloor x \rfloor$ is the largest integer that is not greater than x. A node n_j of the segmentation tree has either two children, nodes n_{2j+1} and n_{2j+2}, or no child at all. A node that has no child is a leaf node. In MMP only the leaf nodes of the segmentation tree are associated to matrices in the dictionary that are used to approximate the input matrix.

The dictionary in MMP is updated as follows: Whenever the approximations $\hat{\mathbf{X}}^{2j+1}$ and $\hat{\mathbf{X}}^{2j+2}$ associated to the children nodes n_{2j+1} and n_{2j+2} are available, MMP forms an estimate $\hat{\mathbf{X}}^j$ of the block associated to the parent node n_j, as the concatenation of the approximations to the blocks associated to the two children nodes. In the example of figure 1, when $\hat{\mathbf{X}}^5$ and $\hat{\mathbf{X}}^6$ are available we can concatenate them to get a new approximation $\hat{\mathbf{X}}^2$. This new approximation can then be included in the dictionary. This procedure can be executed at the encoder side and repeated at the decoder side. We just need to output to the decoder the information regarding the segmentation tree and the dictionary indexes corresponding to the approximations at the leaf nodes. That is, the algorithm outputs an integer sequence i_m consisting of the dictionary indexes and a sequence of binary flags b_n that specify the segmentation tree $\mathcal{S}$. The sequence of flags represent $\mathcal{S}$ as a series of binary decisions, in a top-down fashion. For example, if we use the binary flag 0 to indicate splitting, the tree in figure 1(b) is represented by the sequence of flags 0, 1, 0, 1, 1.

The Segmentation tree $\mathcal{S}$ can be optimized in a rate-distortion sense. The basic optimization procedure, as described in [1], starts with a full binary segmentation tree and works from the leaves up to the root, pruning a pair of nodes n_{2j+1} and n_{2j+2} whenever the Lagrangian cost of the tree preserving them is greater than the cost associated to the segmentation tree with n_{2j+1} and n_{2j+2} pruned. The Lagrangian cost of a segmentation tree $\mathcal{S}$ is defined as $J(\mathcal{S}) = D(\mathcal{S}) + \lambda R(\mathcal{S})$, where $D(\mathcal{S})$ is the distortion obtained when using $\mathcal{S}$ and $R(\mathcal{S})$ is the rate.

3. SM-MMP

In the original MMP, each segmented block is independently coded, regardless of the neighbouring blocks. When the source image is not white noise, we can expect that the behavior of the neighborhood can be used to estimate the content of a given block. We then propose SM-MMP, an MMP variation that uses the information learned from previously coded blocks in a side-match vector quantization [2] fashion. This is explained in what follows.

For an $n \times m$ block $\mathbf{X}^j$, associated to the segmentation tree node n_j, we define the *upper neighbor* $\mathbf{U}^j$ as the *reconstructed* $n \times m$ block immediately above $\mathbf{X}^j$, and the *left neighbor* $\mathbf{L}^j$ as the *reconstructed* $n \times m$ block immediately to the left of $\mathbf{X}^j$. This is illustrated in figure 2. Both $\mathbf{U}^j$ and $\mathbf{L}^j$ are always available prior to the encoding of $\mathbf{X}^j$, because of the order that MMP traverses the segmentation tree.

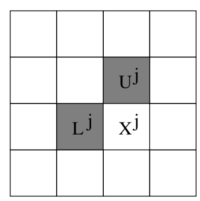

Fig. 2. Up and left neighbors.

As in the original MMP, we use the dictionary containing matrices of size $n \times m$ to encode the block $\mathbf{X}^j$. However, we use a *smoothness criterion* to modify the probability of occurrence associated to each element of the dictionary. In our model, those vectors in the dictionary that do not meet the smoothness criterion have their probabilities set to zero. This is equivalent to reduce the cardinality of the dictionary. We call the reduced dictionary the *state dictionary*. To build the state dictionary, we make decisions based on the *rugosity* metric $r_{ij} = R\left(\mathbf{U}^j, \mathbf{L}^j, \mathbf{S}^i\right)$, defined as:

$$
\begin{aligned}
&R\left(\mathbf{U}^j, \mathbf{L}^j, \mathbf{S}^i\right) = \\
&\sum_{k=0}^{m-1} \left| \left\lfloor \frac{U^j_{m-2,k} - U^j_{m-1,k} + S^i_{0,k} - S^i_{1,k}}{2} \right\rfloor + S^i_{0,k} - U^j_{m-1,k} \right| \\
&+ \sum_{p=0}^{n-1} \left| \left\lfloor \frac{L^j_{p,n-2} - L^j_{p,n-1} + S^i_{p,0} - S^i_{p,1}}{2} \right\rfloor + S^i_{p,0} - U^j_{p,n-1} \right|
\end{aligned}
\quad (1)
$$

The *rugosity* can be seen as a measure of how different are border pixels from neighboring blocks. As we assume that the image being processed is smooth, the first elements in the *state dictionary* are the ones that present the smallest *rugosities*.

To encode a given block $\mathbf{X}^j$, we choose a state dictionary $\mathcal{D}_s$ that is composed of the N least rugose elements $\mathbf{S}^i$ of the dictionary $\mathcal{D}$, according to equation (1), that is, the N elements with smaller values of r_{ij}. The size N of the state dictionary depends on the level of *activity* of the blocks $\mathbf{U}^j$ and $\mathbf{L}^j$. We evaluate the activity of a block $\mathbf{X}$ of size $n \times m$ by the function $A(\mathbf{X})$, defined as:

$$A\left(\mathbf{X}\right) = \max_{n,m} \left\{ \left(\sum_{k=0}^{m-1} |X_{n+1,k} - X_{n,k}| \right), \left(\sum_{p=0}^{n-1} |X_{p,m+1} - X_{p,m}| \right) \right\} \quad (2)$$

The *activity* is a measure of the uncertainty about the other pixels that do not belong to the border of the block. A high level of uncertainty about them means that we need a bigger *state dictionary* for a more precise encoding.

The size N of the state dictionary used to encode a block $\mathbf{X}^j$ is determined by the ratio $N_{max}\left(\left(A\left(\mathbf{U}^j\right) + A\left(\mathbf{L}^j\right)\right)/2\right) A_{max}^{-1}$, where A_{max} is the maximum value of $A(\mathbf{X})$ for the hole image and N_{max} is the maximum size allowed for the state dictionary.

To further improve the performance we modified the dictionary-updating procedure by forcing the algorithm to learn not only the concatenations of blocks, but also shifted versions of those. This allows the dictionary $\mathcal{D}$ to grow faster, reducing the mean distortion. Note that the mean rate is not significantly affected since the size N of the state dictionary $\mathcal{D}_s$ is independent of the size of $\mathcal{D}$ (see equation (2)).

4. EXPERIMENTAL RESULTS

We have implemented the original MMP and the SM-MMP. We used the programs to lossy compress gray-scale still image data. The input images were initially divided in 16×16 blocks that were sequentially processed by the algorithms. The initial dictionary contained only vectors at scale 1×1, and was set to $\mathcal{D}_0 = \{0, 2, \ldots, 254\}$. The vectors at all other scales were obtained by the use of a *separable* two-dimensional scale transformation that was implemented using classical sampling-rate change operations, as described in [1]. Since the block size is 16×16, 9 different scales have been used (see figure 1(a)).

Figures 3, 4, 5 and 6 show the R-D performance of the algorithms with the images Lena, F-16, PP1205 and PP1209, all of size 512×512. The results for the Side-Match vector quantizer described in [3] (just for Lena and F16) and for the SPIHT algorithm [5] are also shown for comparison. Since very similar results are obtained for JPEG2000 [4] and SPIHT algorithms, we have not included the results for the first (in fact, results for JPEG2000 tend to be slightly worse than the results for SPIHT, since, due to JPEG2000's much larger flexibility, its headers are much longer, what decreases its coding efficiency). The test images Lena and F-16 were downloaded from the location http://sipi.usc.edu. The images PP1205 and PP1209 were scanned, respectively, from pages 1205 and 1209 of the *IEEE Transactions on Image processing*, volume 9, number 7, July 2000. PP1205 contains only text and formulas, while PP1029 is a compound of gray-scale (two compressed versions of Lena), text, formulas and graphics. The figures show us that:

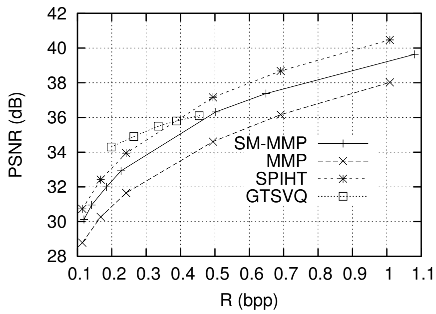

Fig. 3. R-D performance with Lena 512×512.

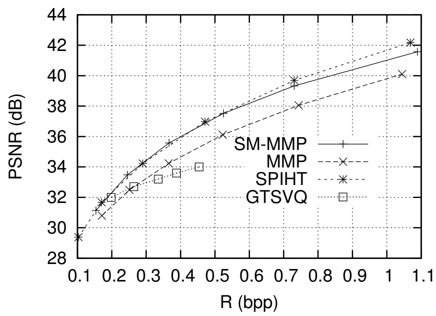

Fig. 4. R-D performance with F-16 512×512.

i) The SM-MMP algorithm outperformed MMP for almost all images of the test group and presented equivalent results for PP1205.

ii) Both SM-MMP and MMP algorithms outperformed SPIHT $\approx$ 2dB for PP1209 (compound grayscale, text and graphs) and by $\approx$ 4dB for PP1205 (text and equations)

iii) The SPIHT algorithm outperformed SM-MMP for the image Lena. The performance with F-16 below 0.5 bpp was equivalent for the two algorithms.

iv) SM-MMP outperformed the Side-Match VQ [3] for the F16 test image at all rates. For the image Lena, SM-MMP performed better above 0.4 bpp. Note although the rate range provided in reference [3] is very narrow, we can see a clear tendency for the improvement provided by SM-MMP over Side-Match VQ to increase with the rate (we did not have access for results of Side-Match VQ for either other images or other rates).

Figure 7 shows a detail of the image Lena compressed by SM-MMP and MMP. It is quite clear the reduction of the blocking effect by the use of the smoothness criterion in SM-MMP.

5. CONCLUSIONS

In this paper, we have proposed SM-MMP, a method for multi-dimensional signal compression based on MMP. We were able to achieve significant gains over the original MMP for smooth images by introducing an underlying statistical model to the image source. This was obtained without sacrificing its performance for images with texts and graphics. The adopted model is quite simple, based on the side-match vector quantisation approach. We expect even better results as we improve the image model, using for example Gibbs distribution models.

(a) (b)

Fig. 7. Image Lena (detail): (a) MMP at 0.30 bpp. PSNR = 32.71 dB; (b) SM-MMP at 0.30 bpp. PSNR = 34.13 dB.

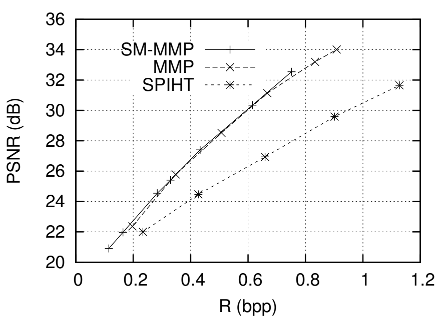

Fig. 5. R-D performance with PP1205 512×512.

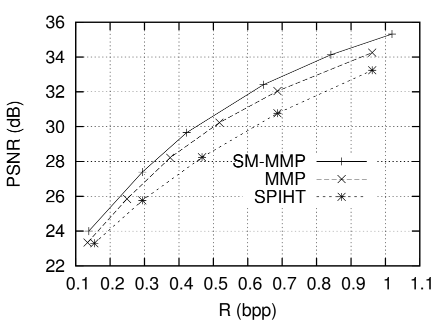

Fig. 6. R-D performance with PP1209 512×512.

6. REFERENCES

[1] M. B. de Carvalho, E. A. B. da Silva and W. A. Finamore, "Multidimensional Signal Compression using Multiscale Recurrent Patterns," Elsevier's Signal Processing, Vol. 82, No. 11, pp. 1559-1580, November 2002.

[2] S. B. Yang and L. Y. Tseng, "Smooth Side-Match Classified Vector Quantizer with Variable Block Size", IEEE Transactions on Image Processing, Vol. 10, No. 5, pp. 677–685, May 2001.

[3] S. B. Yang, "General-Tree-Structured Vector Quantizer for Image Progressive Coding Using the Smooth Side-Match method", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, No. 2, pp. 193–202, February 2003.

[4] D. S. Taubman, M. W. Marcellin, "JPEG2000: Image Compression Fundamentals, Standards, and Practice", Kluwer Academic Publishers, 2001.

[5] 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.