ADAPTIVE TREE-STRUCTURED LATTICE VECTOR QUANTIZATION FOR VIDEO CODING

Vincent Ricordel †*and Moncef Gabbouj* ‡
†MS-GESSY/ISITV, Universite de Toulon et du Var,
Av. Georges Pompidou BP56,
83162 La Valette du Var , FRANCE
‡DMI/SPL, Tampere University of Technology,
P.O. Box 553, 33101-Tampere, FINLAND
e-mail: ricordel@univ-tln.fr, gabbouj@cs.tut.fi

ABSTRACT

The purpose of this paper is to introduce a new adaptive vector quantization (AVQ) for the compression of digital image sequences. In our previous studies [13, 14] we proposed a lattice vector quantizer (LVQ) based on the hierarchical packing of embedded truncated lattices. Now we investigate the capability of this LVQ for AVQ, through a scheme taking account of its tree-structure. Precisely, in order to fit the spatiotemporal statistics of the image sequences, the codebook is designed using a training procedure and in two parts, with: a stump from several types of training sequences; some branches added according to the sequence to be coded. Experimental results are given with the adaptive LVQ taking place in a subband coder.

1 INTRODUCTION

Predictive coding and subband coding [11] are unavoidable decorrelation methods for the compression of digital image sequence. Because an hybrid coder aims to apply precisely two main rules : the non-transmission of predictable information and, the non-transmission of the no perceivable information by the human visual system (HVS). Fig 1 shows the generic coder for which our adaptive LVQ is devoted. Note that we doesn't study in the paper the quantization aspect of the information performed by the motion estimation [16] between each image pair of the input sequence, but the VQ [1] of the transformed prediction errors.

The inter-intra decorrelation steps shape the signal before its quantization. Because in each subimage the data are classified according to their frequential orientation and resolution. A separate VQ of each subimage is well-suited in order to exploit the dependencies between the transformed coefficients [3], and the bit allocation is performed such as, taking account of the HSV propriety, the low frequency subimages receive more bits (but the resulting gain is subjective). The monodimensional pdf of such subimage is commonly mapped by a Gaussian Generalized function whose narrowest highligts the correlation between the coefficients [3]. However the multidimensional signal obtained after the hybrid decorelation chain is always nonstationary [6, 1]. Only an adaptive VQ [9, 7] is capable of adapting to changing source statistics as the coding progresses. Such as, from a representative training sequence, a very fast updating of the VQ codebook is achieved.

[1]from now VQ means vector quantizer as well as vector quantization

2 SUMMARY OF THE PREVIOUS STUDY

2.1 Context : Lattice Vector Quantization

The LVQ [5, 3] has been successfully introduced in order to overcome the LBG-type algorithm drawbacks [8]. The encoding, based on rounding and scalling operations, is simple and independant of the codebook size. There is no need to search among all the reproduction vectors, and practically no norms are computed. Because of the predefined structure of the lattice, there is no need to transmit the codebook and no training procedure is required to design it. But a lattice can only quantize an uniform source and, because of its infinite size, it must be truncate to index the codewords. The choice of the metric L_2 [10, 2] (resp. L_1 [5, 10, 3]) permits to shape the codebook into an hyper-sphere (resp. an hyper-square) and to count the points. As a result the basic LVQ is only adapted for symetric and Gaussian (resp. Laplacian) source distributions which map such a codebook. This restrictive modelization of the source offers some accurated and sophisticated methods to achieve the LVQ design, but the well-perform condition collapses considering a complex source coding at low bit rate. With the tree-structure LVQ (TSLVQ), based on embedding lattice strategy, simple procedures are implemented to overcome these drawbacks.

2.2 Tree-Structure Lattice Vector Quantization

Considering the lattices for which Conway and Sloane determined fast quantizing and decoding algorithms [4], the TSLVQ is based on the hierarchical packing of embedded truncated lattices. In [13, 14] we investigate its complete design with : the lattice truncation, the multistages procedure of quantization, the unbalanced

tree-structured design and the determination of the best lattice respectively to this method. The detection and the processing of the probable outlying input vectors are defined too.
The TSLVQ offers some original solutions to usual LVQ drawbacks, with a space partition according to the source distribution, and a simple labelling of the lattice points. The resulting codebook automatically obtained is well suited for prediction error coding.

2.3 Binary Allocation Method

The bit allocation problem occurs when you have to share the bit ressource between the subbands [15]. We present briefly our solution with the TSLVQ
Precisely it consists in minimizing the distorsion D subject to the constrainst that the global rate R is under a threshold R_d. The transformation is supposed orthogonal and there are M subbands:

$$\min D = \min \sum_{j=0}^{M-1} d_{j,i} \text{ with } R = \sum_{j=0}^{M-1} r_{j,i} \leq R_d$$

For each subband j there are some potential quantizers $q_{j,i}$ (each of them corresponds to a configuration of the TSLVQ tree), $d_{j,i}$ is the distorsion for the rate $r_{j,i}$. For a combination of M quantizers (i.e. a separate TSLVQ for each subband) we get a bipoint (R,D) in the distorsion vs. rate space, and all the N^M combinations produce a cluster (fig 4 shows an example). The problem becomes the determination on the convex hull of this cluster of the bipoint whose rate is just lower than R_d. In order to reduce the amount of calculus the Lagragian multiplier λ is introduced to solve:

$$\min(D + \lambda.R) \Longleftrightarrow \sum_{j=0}^{M-1} \min_{q_{j,i}}(d_{j,i} + \lambda.r_{j,i})$$

The complexity decreases because the reduction of the distorsion is now achieved separately from each subband. The general form of the algorithm is:

1. The convex hull for each subimage is calculated. With the TSLVQ we obtain directly it when growing the tree.
2. The point on the global convex hull is determinated. Its rate is just below R_d.

For this last step the Shoham's method [15] can be applied. It is based on the calculation of singular values of λ (*i.e.* the slopes of the lines that pass through the consecutive points of the convex hull). So, from a first point of the hull and by successive calculations of singular values, we get the global convex hull. The fig 4a shows an experimental result with the TSLVQ. A drawback appears because there are some large gaps between some points of this global convex hull. So we want to modify the algorithm in order to get the "optimal quantizers" (see figure 4), namely the points just above the convex hull. We proceed in two steps:

1. The previous algorithm permits to get two points of the convex hull that surrounding R_d.
2. From these two points, the Shoham's method [15] is used again in order to get a local portion of convex hull.

The curve titled "intermediary quantizers" on the fig 4b, is achieved by calculating the portions of convex hulls between each bi-point of the global convex hull. As a result a lot of optimal quantizers are got.

3 ADAPTIVE VQ DESIGN

For the adaptive TSLVQ the codebook is designed using a training procedure and in two parts:

1. A stump from several types of training sequences (this stump construction is achieved outline).
2. Some branches added according to the sequence to be coded (these branches will constitute the only transmitted information to update the codebook).

Because of the binary allocation that implies a tree pruning, there are precisely four steps in the algorithm (see fig 2):

1. Construction of a large stump.
2. Binary allocation from this stump (the corresponding threshold R_d is very low).
3. Addition of large branches.
4. Final binary allocation with the desired bit rate.

In order to update the codebook, only the steps 3 and 4 are carried out. For the greedy tree growing method of the steps 1 and 3, see [13].

4 EXPERIMENTAL RESULTS

The coder (see fig 1) is MPEG based because the tools are a block matching for the motion estimation and a DCT for the transformation. The block size is 2×2 so four subbands are set up (see fig 3). The simulations are made using QCIF image sequences, and the computer is a SparcStation 20 (75 Mhz).
Images from four different sequences are used for the stump design, but we only use the images of one sequence to add branches (table 3 shows typical numerical results). A particular codebook is then applied to code the sequence that was used for the branches addition (see table 1). The CPU time for an image coding is about 1.8 s.
For comparison, the table 4 shows codebook design results with classical TSLVQ, and the table 2 the corresponding coding results.

5 CONCLUSION

With adaptive TSLVQ we get, with respect to our previous work, a more acurate binary allocation, and a better regularity for the reconstruction when decoding the sequences. The transmitted information to refresh the codebook is not large, and the CPU time to design or update the codebook is short.

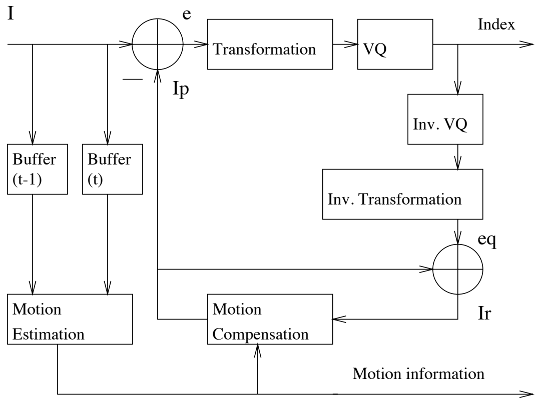

Figure 1: Hybrid coder. I: input image from the sequence; Ip: prediction image; Ir: reconstructed image; e: prediction error image; eq: quantized prediction error image.

sequence name	number of images	average entropy [bpp]	average PSNR [dB]
"Miss America"	150	0.273	38.03
"Salesman"	180	0.170	36.66
"Carphone"	180	0.257	32.65

Table 1: Coding of sequences with the adaptive TSLVQ.

sequence name	number of images	average entropy [bpp]	average PSNR [dB]
"Miss America"	150	0.134	36.87
"Salesman"	180	0.177	36.58
"Carphone"	180	0.381	34.12

Table 2: Coding of sequences with the classical TSLVQ.

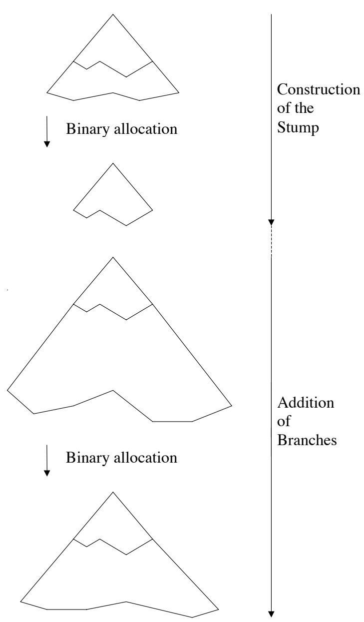

Figure 2: Adaptive TSLVQ, the four steps of the codebook design.

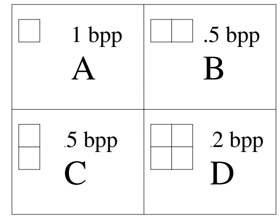

Figure 3: Codebook: Vector shapes and maximal entropies before bit allocation (note the subband labels).

References

[1] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies. Image coding using wavelet transform. *IEEE Transactions on Image Processing*, 2, avril 1992.

[2] M Barlaud. *Wavelets in image communication.* Elsevier, New York, 1994.

[3] M. Barlaud, P. Solé, T. Gaidon, M. Antonini, and P. Mathieu. Pyramidal lattice vector quantization for multiscale image coding. *IEEE Transactions on Image Processing*, 3(4):367–381, juillet 1994.

[4] J.H. Conway and Sloane N.J.A. Fast quantizing and decoding algorithms for lattice quantizers and codes. *IEEE Transactions on Information Theory*, 28(2):227–232, mars 1982.

[5] T.R. Fisher. A pyramid vector quantizer. *IEEE Transactions on Information Theory*, 32(4):568–583, juillet 1986.

[6] T.R. Fisher. Geometric source coding and vector quantization. *IEEE Transactions on Information Theory*, 35(1):137–145, janvier 1989.

[7] J.E. Fowler. *Adaptive Vector Quantization for the Coding of Nonstationary Sources.* PhD thesis, The Ohio State University, 1996.

[8] A. Gersho and R.M. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Boston, 1992.

[9] M. Goldberg, P.R. Boucher, and S. Schlien. Image compression using adaptative vector quantization. *IEEE Transactions on Communications*, 34(2):180–187, février 1986.

[10] D.G. Jeong and J.D. Gibson. Uniform and piecewise uniform lattice vector quantization for memoryless gaussian and laplacian sources. *IEEE Transactions on Information Theory*, 39(3), mai 1993.

[11] A.N. Netravali and B.G. Haskell. *Digital pictures : Representation and Compression.* Plenum Press, New York, 1988.

[12] K. Ramchandran and M. Vettrerli. Best wavelet packet bases in a rate-distorsion sense. *IEEE Transactions on Image Processing*, 2(2), avril 1993.

[13] V. Ricordel and C Labit. Vector quantization by packing of embedded truncated lattices. In *Proc. of International Conference on Image Processing ICIP*, volume 3, pages 292–295. Washington DC, USA, octobre 1995.

[14] V. Ricordel and C Labit. Tree-structured lattice vector quantization. In *Proc. of European Signal Processing Conference EUSIPCO*, volume 2, pages 731–734. Trieste, Italie, septembre 1996.

[15] Y. Shoham and A. Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Transactions on Acoust. Speech Signal Processing*, 36(9):1445–1453, septembre 1988.

[16] G. Tziritas and C. Labit. *Motion analysis for image sequence coding*, volume 4 of *Advances in image communication.* Elsevier Science Publishers, Londres, juillet 1994.

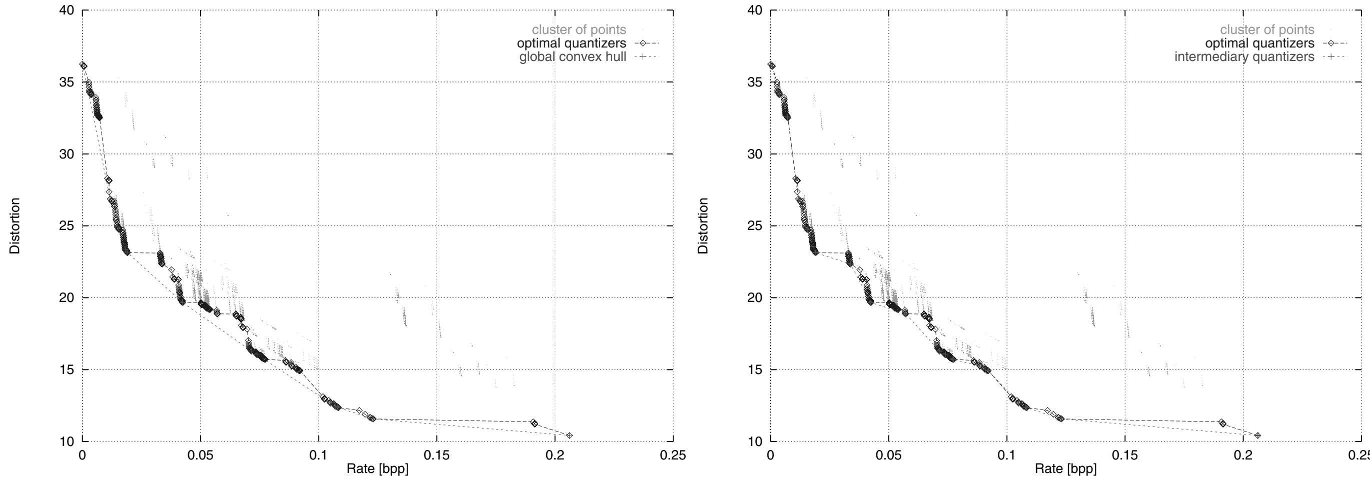

Figure 4: [a,b]: Cluster of bi-points (R, D) and hulls calculated for the binary allocation.

stump design from four sequences							
	before bit allocation				after bit allocation ($R_d = 0.1$bpp)		
subband labbel	training sequence size [images]	CPU time for construction [s]	number of codevectors	codebook entropy [bpp]	number of codevectors	codebook entropy [bpp]	codebook size [byte]
A	4	4.93	55	0.698	20	0.332	238
B	12	6.15	106	0.213	26	0.020	259
C	12	6.22	133	0.231	34	0.027	315
D	200	69.43	1982	0.083	92	0.003	810
addition of branches from the sequence "Salesman"							
	before bit allocation				after bit allocation ($R_d = 0.2$bpp)		
subband labbel	training sequence size [images]	CPU time for construction [s]	number of codevectors	codebook entropy [bpp]	number of codevectors	codebook entropy [bpp]	additional information [byte]
A	5	7.18	48	0.954	20	0.337	0
B	10	5.97	104	0.205	104	0.205	681
C	10	6.11	161	0.318	86	0.253	448
D	150	52.92	1348	0.078	92	0.002	0

Table 3: Codebook design with adaptive TSLVQ: numerical results.

	before bit allocation				after bit allocation ($R_d = 0.20$bpp)		
subband labbel	training sequence size [images]	CPU time for construction [s]	number of codevectors	codebook entropy [bpp]	number of codevectors	codebook entropy [bpp]	codebook size [byte]
A	5	7.15	27	0.964	21	0.420	244
B	10	5.48	137	0.258	37	0.033	336
C	10	5.50	148	0.281	148	0.281	1358
D	150	46.75	1248	0.087	529	0.066	4995

Table 4: Codebook design with classical TSLVQ and using the sequence "Salesman": numerical results.