16th European Signal Processing Conference (EUSIPCO 2008), Lausanne, Switzerland, August 25-29, 2008, copyright by EURASIP

DESIGN OF LOW-RESOLUTION MULTIPLE DESCRIPTION VECTOR QUANTIZERS BY MEANS OF THE SELF ORGANIZING MAPS

Giovanni Poggi, and Luisa Verdoliva

Dipartimento di Ingegneria Elettronica e delle Telecomunicazioni,
Università Federico II di Napoli, Italy, Via Claudio 21, 80125, Naples, Italy
email: {poggi, verdoliv}@unina.it

ABSTRACT

Multiple description coding is an appealing tool to guarantee graceful signal degradation in the presence of unreliable channels. While the principles of multiple description scalar quantization are well-understood and solid guidelines exist for the design of effective systems, the same does not apply to vector quantization, with burdensome and not always satisfactory design techniques.

In this work we show that good multiple description VQ codebooks can be designed in a simple and efficient way by resorting to the Self-Organizing Maps algorithm, where just a few parameters must be reasonably selected. Experimental results show that the proposed technique has a performance comparable to a well-known reference technique based on greedy optimization, but a much lower computational burden. In addition, the resulting codebook can be itself optimized, thus providing even better performance.

1. INTRODUCTION

Multimedia contents are becoming a larger and larger fraction of the traffic on wireless networks. In this context, characterized by bandwidth constraints and unreliable channels, the design of efficient communication systems must take into account the interplay between source and channel coding. In many scenarios, the conventional error control approaches, based on error detection and packet retransmission, are not acceptable because of exceedingly long delays, significant bit-rate overhead, or practical implementation issues [1]. In such cases, Multiple Description Coding (MDC) represents a valuable alternative since it allows, with limited overhead, to obtain a graceful degradation of the decoded signal even in the presence of one or more channel failures.

The simplest model of MDC, depicted in Fig.1, comprises only two alternative channels over which the information is split. If both channels work properly, the signal is reconstructed by the central decoder, with the full quality allowed by the source coding technique[1] at the given bit rate. However, even in the case of failure of one channel, the surviving side description will allow a reasonable reconstruction of the source, although at a reduced quality. This paradigm is therefore similar to scalable source coding, with the difference that, here, the descriptions are (in general) equally important, and each one can be used by itself to recover an approximation of the source. Needless to say, just like the constraint of scalability entails a performance loss with respect to unconstrained coding, the more stringent constraint imposed by MDC leads to a further loss, that is the price to pay in order to guarantee an acceptable performance on unreliable networks.

[1] Actually, the figure is drawn having already in mind the special case of MD Vector Quantization: the descriptions are just two scalar indexes i and j, that are used to recover codewords from the side codebooks $\mathscr{Y}^{(1)}$ and $\mathscr{Y}^{(2)}$ or, if both available, from the central codebook $\mathscr{Y}^{(0)}$.

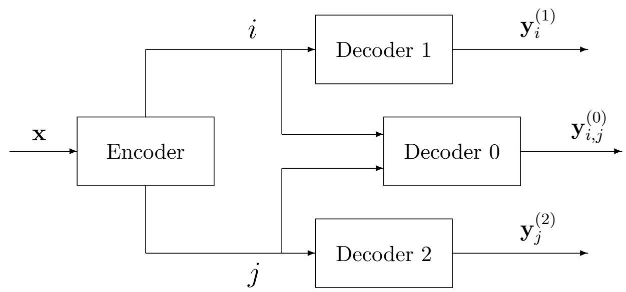

Figure 1: Block diagram of a 2-description Vector Quantizer.

Theoretical results on MDC are quite scant, major contributions are due to Ozarow [2] and El Gamal and Cover [3] which provided the rate-distortion region for the special but important case of memoryless Gaussian sources with squared error distortion. Later on, Vaishampayan proposed [4] a Lloyd-type algorithm for the design of MD scalar quantizers, sheding some light on the importance of a suitable index assignment, and suggesting some practical solutions to this problem. After this seminal work, the research on MDC has become very intense, with developments in several directions.

A topic of obvious interest is the extension of MD ideas from scalar to vector quantization, that is, the design of multiple-description vector quantization (MDVQ) systems, so as to take advantage of the higher structural freedom, and hence the potential gains, existing in a higher-dimensional space. Such an extension, however, is not trivial, just because of the index assignment problem, since codewords have a natural order in a 1-dimensional space but not in spaces of higher dimensionality. Some elegant solutions based on lattice vector quantization have been proposed [5], but these are not really general, since they work only with pretty regular sources, and very large codebooks.

For the low-resolution case, the key problem is the design of a VQ codebook having a suitable index assignment, and some techniques have been proposed lately for this task [6, 7, 8] following two quite different approaches: *i)* designing a VQ codebook which possesses *a priori* the desired index organization, or *ii)* designing a generic codebook and using some other processing afterwards to reassign the indexes. The first approach is certainly appealing, and the algorithm proposed in [6], for example, is relatively simple, and generates a tree-structured codebook that can be readily used for a multiple description VQ. However, it is also quite rigid as it only allows for non-redundant descriptions, implying a significant loss of quality for at least one of them. In addition, tree-structured VQ is suboptimal, and hence also the central decoder is suboptimal. As for the second approach, instead, the major drawback is the computational complexity of the index assignment problem which grows very fast with the codebook size. A brute-force optimization can be used only for very small codebooks, and one must therefore resort to suboptimal techniques, such as the MDBSA proposed in [7] which is a suitable modification of the well-known binary switching algorithm (BSA) [9]. MDBSA pursuits an optimal solution through local moves consisting in the reassignment of just a few indexes at a time. At each step, only the best move is accepted, until a local optimum is reached. MDBSA provides generally good results but, despite the simplified optimization path, its complexity is still prohibitive if the codebook size is relatively large, especially if more than two descriptions are considered.

In this work we propose a new technique for designing low-resolution MDVQ systems based on Kohonen's Self Organizing Maps (SOM) algorithm [10], also known as Learning VQ. The proposed algorithm is characterized by several appealing properties

- the index assignment is decided in advance, with no need for further rearrangements, hence our approach fits in the first group mentioned before;
- the quality of the central codebook is unaffected by the procedure, that is, it is comparable to that of unconstrained VQ;
- the computational complexity is quite limited, comparable to that of the well-known Generalized Lloyd algorithm;
- the algorithm is highly flexible: it works with any number of descriptions, be they redundant or non-redundant, possibly asymmetric, etc.

In next Section we recall the functioning and the basic properties of the SOM, with special attention to its codebook ordering ability. Then Section 3 describes the proposed solution, Section 4 presents some sample experimental results, and Section 5 draws conclusion and outlines future research.

2. SELF ORGANIZING MAPS FOR THE DESIGN OF ORDERED VQ CODEBOOKS

The SOM can be seen as an improvement of the well-known K-means clustering algorithm [11] also used for the codebook design. In the K-means, the arbitrary initial codebook $\mathscr{Y}_0 = \{\mathbf{y}_n \in \mathscr{R}^k, n = 1,\dots,N\}$, is gradually adjusted in response to the submission of a training set. Specifically, for each training vector $\mathbf{x}_s$ the nearest codevector $\mathbf{y}_{n_s}$ is singled out and displaced towards the input

$$\mathbf{y}_{n_s} = \mathbf{y}_{n_s} + \alpha(s)\,(\mathbf{x}_s - \mathbf{y}_{n_s}), \qquad (1)$$

thus reducing coding distortion, on the average. The function $\alpha(s)$ controls the speed of adaptation to the training set. It is rather large initially to quickly match the training set, and then decreases slowly to zero to guarantee convergence and fine tune the codebook.

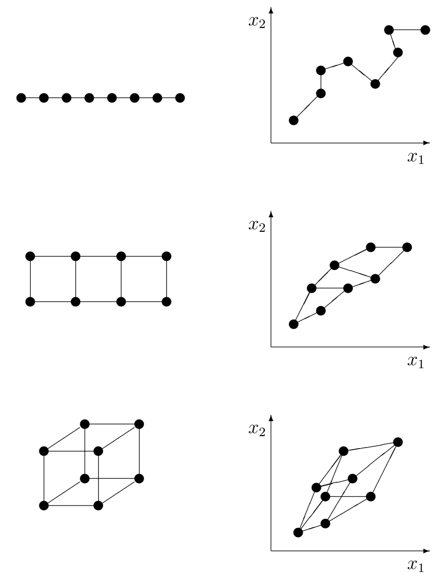

Figure 2: Three codebooks with different *a priori* index structure designed for the same source.

The major improvement of the SOM with respect to the K-means consists in the organization of the codebook according to a desired structure, typically a regular lattice in M dimensions. This is obtained by introducing some form of mutual interaction between codevectors. Accordingly, the adaptation formula becomes

$$\mathbf{y}_n = \mathbf{y}_n + \alpha(s)\,\beta(s,n,n_s)\,(\mathbf{x}_s - \mathbf{y}_n), \qquad n = 1,\dots,N, \quad (2)$$

where all codevectors, and not just the best matching, are updated with each new training vector. The function $\beta(\cdot,\cdot,\cdot)$ is responsible for the strength of interaction among codevectors: if codewords $\mathbf{y}_n$ and $\mathbf{y}_m$ are desired to be close in the codebook, $\beta(\cdot,n,m)$ must have a positive, and possibly large, value, while a negative value will ensure that such codewords will remain far apart. In any case, all interaction strengths vanish as s goes to infinity, when the desired codeword organization has been already obtained so as to allow for a fine tuning to the training set.

To gain some insight about the algorithm functioning, one can think of codevectors as linked by springs or rubber bands so that whenever one is displaced, during the learning phase, it trails all linked codevectors with it. By properly choosing the strength of each connection, namely, the intensity of interaction between couples of codevectors, one forces the codebook to assume the desired topology. An example is shown in Fig.2, where three codebooks with different desired structures are designed for the same source. All of the codebooks adapt quite well to the input distribution, a bivariate correlated Gaussian, but each of them preserves its own topology due to the interaction among codevectors.

In practice, the mutual interaction function depends in its turn on a measure $d(n,m)$ of the distance among indexes. By selecting a suitable index distance, one can obtain different codebook structures and then address different problems involving the use of ordered codebooks. For example, a linear distance $d(n,m) = |n-m|$ guarantees that codewords with close indexes are also close in the Euclidean space, a prop-

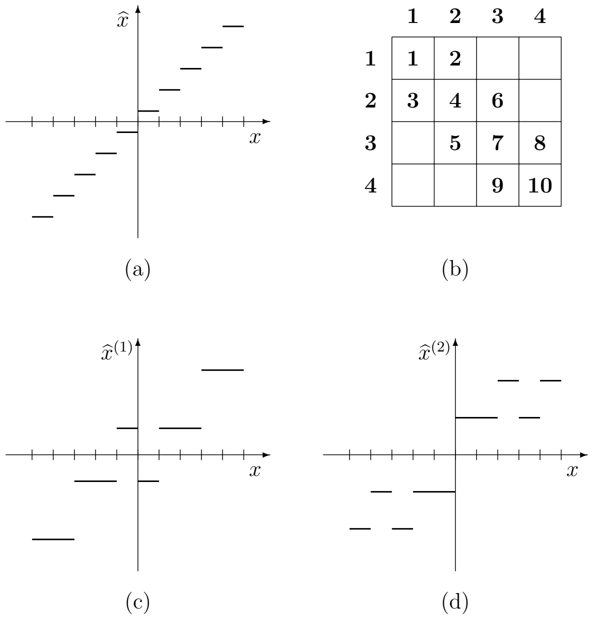

Figure 3: Example of MDSQ: (a) central quantizer, (b) index assignment matrix, (c)-(d) side quantizers.

erty that can be used to carry out an effective compression of the indexes output by a vector quantizer [12, 13]. The use of a hamming distance $d(n,m) = h(n,m)$, instead, produces a codebook with hypercubic structure, which allows one to obtain a zero-redundancy protection of VQ indexes from the effects of noise on transmission channels [12]. A more complex index distance allows one to address the design of Trellis-coded VQ systems [14].

In the following section we will see that, by using the SOM with suitable mutual interactions, we can easily design good codebooks for Multiple Description VQ. Before that, however, it is worth emphasizing that using the SOM in place of the well-known Generalized Lloyd Algorithm (GLA) [15] does not entail, in general, a performance loss. In fact, although SOM codebooks are not optimal for the given training set, the increase in MSE with respect to the optimal (at least locally) GLA codebooks is quite limited, especially if a reasonably large training set is used. In addition, for out-of-training data, SOM codebooks usually *outperform* GLA's [12] since each training vector is used to adjust several codewords at a time, thus increasing the effective training set size. As for the computational burden, although SOM performs more updates than GLA for each training vector, it needs a much smaller training set to reach near-optimality, and is therefore less complex [16].

3. PROPOSED SOLUTION

To gain insight about the proposed solution, let us consider for the time being the simple 2-description scalar quantizer depicted in Fig.3. Assuming a uniform pdf for the input, the 10-level central quantizer shown in part (a) is optimal. Part (b) shows a possible index assignment matrix (IAM), which leads to the side quantizers shown in parts (c) and (d). For example, the second cell of the first side quantizer is the union of cells 3, 4 and 6 of the original central quantizer, and the

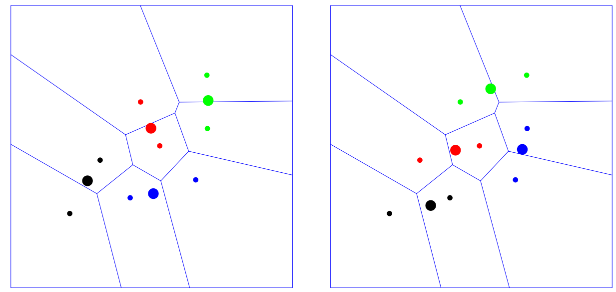

Figure 4: A 3-bit MDVQ codebook (small dots) with two 2-bit side codebooks (large dots).

corresponding optimal codeword is $\mathbf{y}_2^{(1)} = -7/30$. Selecting a suitable assignment matrix is the central problem in MDSQ: one should ensure that cells that appear in the same row or in the same column of the matrix are as close as possible and, since the original cells are aligned along the x axis, this is obtained by minimizing the index spread along rows and columns, a well-known rule first suggested by Vaishampayan [4].

It is worth underlining, however, that minimizing the index spread is only instrumental to achieving the ultimate goal, that is, ensuring that same-row and same-column cells are close. In a higher-dimensional space, where cells are no longer aligned, index spread is not so significant anymore, and we should instead pursue directly the real goal.

We can do this quite simply by resorting to the SOM. As a matter of fact, all that is required is to establish some attractive mutual interactions among codewords that lie in the same line. If the SOM is indeed able to enforce the constraints of interest, we can design a VQ codebook where same-line codewords are actually close and, at the same time, a high-quality reconstruction is guaranteed by the central decoder.

An interesting example is shown below with reference to a 2-d correlated ($\rho = 0.5$) Gaussian distribution, for which we want to design a 3-bit central decoder with two 2-bit side decoders. To this end, we use the following IAM

1	2		
	3	4	
		5	6
8			7

and, with reference to (2), define

$$\alpha(s) = \alpha_0[\Delta_\alpha]^s \qquad (3)$$

and

$$\beta(s,n,m) = \begin{cases} 1 & n = m, \\ \beta_0[\Delta_\beta]^s & n,m \in \text{ same row/column}, \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

with $\alpha_0, \Delta_\alpha, \beta_0, \Delta_\beta \in [0,1]$. As a result, the SOM produces the codebooks shown in Fig.4, where central codewords are marked by small dots, while the larger dots indicate the codewords of the first (left) and second (right) side description.

A few comments are in order: first of all, the average MSE of the SOM codebook is 0.1874 as opposed to 1.862

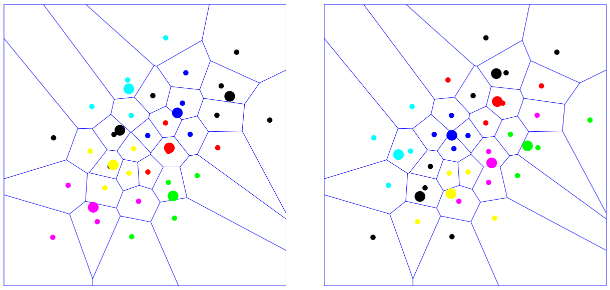

Figure 5: A 5-bit MDVQ codebook (small dots) with two 3-bit side codebooks (large dots).

for the corresponding GLA codebook (both computed on 100 trials and the same training set) showing that the mutual interaction constraints did not interfere with the correct development of the codebook. Moreover, codewords are obviously well-organized in order to provide good-quality side descriptions, as is made visually clear by the color coding. Finally, notice that the IAM does not respect the minimum-spread rule. This should be obvious, after a moment thought, since the first and last codewords need not be far apart in a k-d space, but this observation makes clear that there are indeed new degrees of freedom to be exploited, and problems to be solved.

This trivial example should not hide the complexity of the problem. Considering the same 2-d distribution as before, but going to 5-bit central and 3-bit side descriptions, a good organization of the codewords is again provided by the SOM, as shown by Fig.5, which could not be easily devised otherwise. In higher-dimensional spaces, of course, *guessing* a good solution is a lot more challenging.

At this point, we can summarize the design procedure. The first step consists, as usual, in the selection of the index assignment matrix. The examples discussed above make clear that the minimum-spread rule is no longer a reliable guideline, and that the assignment should depend on the dimensionality of the input space or, more precisely, on the effective dimensionality of the input pdf. However, it is not clear *how* to use such prior knowledge in the definition of the matrix and this is outside of the scope of the current work. Then, given the IAM, we define the mutual interactions according to (4) with suitable values selected for the parameters. This is a drawback of the proposed technique, since having to deal with parameters is always disturbing, but our experience shows that the optimum is pretty flat and reasonable parameters are quite easy to single out. At this point, the SOM algorithm produces automatically the desired MDVQ codebook.

Before providing a few sample numerical results, we want to underline the strengths of the proposed approach. First of all it is extremely flexible: it can work with any number of descriptions, redundant or non-redundant, symmetric or not. It is only required to select an acceptable IAM, with no particular constraints. In addition, after the design it is not necessary to resort to burdensome codebook reordering procedures, like the MDBSA. On the other hand, such procedures *can* be used to improve the codebook after SOM, which can therefore be regarded as an initialization procedure that provides a good starting point.

	training set		test set		
	cent.	side	cent.	side	time
GLA-BSA	.0288	.1509	.0307	.1518	181
SOM	.0288	.1693	.0306	.1710	55
SOM-BSA	.0288	.1482	.0306	.1500	163

Table 1: Central and side distortions (MSE) for proposed and reference techniques: 6-bit central and 4-bit side codebooks.

	training set		test set		
	cent.	side	cent.	side	time
GLA-BSA	.0150	.2552	.0167	.2621	1722
SOM	.0152	.3077	.0162	.3102	357
SOM-BSA	.0152	.2488	.0162	.2509	1111

Table 2: Central and side distortions (MSE) for proposed and reference techniques: 7-bit central and 4-bit side codebooks.

4. EXPERIMENTAL RESULTS

Here we present some experimental results carried out in order to compare the performance of the proposed solution with that of the reference MDBSA technique, and in fact repeat some experiments presented in [7]. As a matter of fact, we implemented the MDBSA as well, in order to compare also processing-time results.

First, we consider a 2-d Gaussian distribution with unit-variance i.i.d. components, and design a two-description MDVQ system with 6-bit central codebook and balanced 4-bit side codebooks. Results, averaged over 30 runs of the algorithms, are reported in Tab.1 in terms of mean-square error, computed both inside and outside the training set.

First of all, GLA and SOM provide central codebooks of about the same quality, with a small penalty for the test set performance which could be reduced by using a larger training set. As for the side descriptions, the SOM does a pretty good work but does not attain the performance of the GLA-MDBSA, with an MSE penalty of about 10% both in and out the training set. Work is underway to understand the reason for this impairment, maybe related to the initial IAM, which in our experiments is diagonal on a torus, that is, with first and last side indexes considered as neighbors, as in our preceding examples. However, using a truly diagonal matrix like those of [4] does not seem to provide improvements.

It is worth noting, however, that the computation time[2] of the SOM is much smaller than that of the GLA followed by the MDBSA, basically because of the complexity of the latter. Moreover, as suggested before, one can run the MDBSA on top of the SOM as well, and in this case, interestingly, the quality of the side descriptions improves beyond that of GLA-MDBSA, although just slightly, and still with a smaller computation time.

In a second test, we consider the same source as before, but the MDVQ system has 7-bit central description and 4-bit side descriptions. Results reported in Tab.2 are in line with

[2] CPU times (in seconds) are only for comparison, as they refer to a Matlab implementation on a Pentium PC.

	2×2 blocks			4×4 blocks		
	cent.	side	time	cent.	side	time
GLA-BSA	20.9	92.0	500	63.6	155.6	1158
SOM	21.4	102.6	124	64.4	191.6	118
SOM-BSA	21.4	79.3	517	64.4	151.3	492

Table 3: Central and side distortions (MSE) for image Lena with 6.21-bit central and 4-bit side codebooks.

Figure 6: Detail of Lena decoded with both description (left) or just one of them (right).

those of Tab.1 (and both with those of [7]), the main difference being the increase in the side-description MSE due to the reduced redundancy. It is just worth underlining, here, that all computation times have grown significantly, but especially those involving the MDBSA, and especially starting from a random (non ordered) codebook, to the point that it is definitely preferable using the SOM before the MDBSA both for performance and computation time.

Finally, as an example of a real-world application let us consider the encoding of gray-scale images. We use a barebone coding scheme, with just plain MDVQ of square blocks, without any prior decorrelating transform or prediction. In this case, since data exhibit a significant correlation, we use a "nested" index assignment that minimizes index spread, as suggested in [4], with 6.21-bit central description (74 codewords) and 4-bit side descriptions. Results are reported in Tab.3 for the well-known test image Lena and 2×2 and 4×4 blocks, and despite the obvious differences, show the same general behavior discussed above. A detail of the image encoded with 2×2 blocks, and decoded with both descriptions or just one of them is shown in Fig.6. There is an obvious degradation in the second case (just 16 codewords available) but the image is clearly recognizable anyway.

5. CONCLUSIONS

We have shown that the design of multiple description VQ codebooks can be carried out in a simple and efficient way by resorting to the SOM algorithm where suitable mutual interactions are established among codevectors. The most appealing feature of this approach, in our view, is its high flexibility: all that is needed to design an MDVQ system with more than two descriptions, for example, is the selection of the initial index assignment matrix. Sample experimental results prove the proposed approach to be effective and relatively fast, although performance (without further optimization) is not currently on par with established reference techniques.

A deeper analysis of the algorithm performance in various conditions is certainly needed, together with a better understanding of how the various parameters should be tuned. In particular, the choice of the IAM as a function of the source and system characteristics deserves further investigation because guidelines valid for the scalar case are not useful anymore in this setting.

REFERENCES

[1] V.K.Goyal, "Multiple description coding: compression meets the network," *IEEE Signal Proc. Magazine*, pp.74-93, Sep. 2001.

[2] L.Ozarow, "On a source coding problem with two channels and three receivers," *Bell Systems Technical Journal*, pp.1909-1921, Dec. 1980.

[3] A.A.El Gamal, T.M.Cover, "Achievable rates for multiple descriptions," *IEEE Trans. Info. Theory*, pp.851-857, Nov. 1982.

[4] V.A.Vaishampayan, "Design of multiple description scalar quantizer," *IEEE Trans. Info. Theory*, pp.821-834, May 1993.

[5] V.K.Goyal, J.A.Kelner, J.Kovacevic, "Multiple description vector quantization with a coarse lattice," *IEEE Trans. Info. Theory*, pp.781-788, Mar. 2002.

[6] M.Fleming, M. Effros, "Generalized multiple description vector quantization," *Data Compression Conference (DCC)*, pp.3-12, Mar. 1999.

[7] N.Gortz, P.Leelapornchai, "Optimization of the index assignments for multiple description vector quantizers," *IEEE Trans. Comm.*, pp.336-340, march 2003.

[8] J.Cardinal, "Entropy-constrained index assignments for multiple description quantizers," *IEEE Trans. Signal Proc.*, pp.265-270, Jan. 2004

[9] K.Zeger, A.Gersho: "Pseudo-Gray coding," *IEEE Trans. Comm.*, pp.2147-2158, Dec. 1990.

[10] T.Kohonen, *Self-Organizing Maps, 3rd Ed.*, Springer, Berlin, 2001.

[11] M.R.Anderberg: *Cluster analysis for applications*, Academic Press, New York, 1973.

[12] G.Poggi, "Applications of the Kohonen algorithm in vector quantization," *European Trans. Telecommunications and Related Technologies*, pp.191-202, March 1995.

[13] G.Poggi, "Generalized-cost-measure-based address-predictive vector quantization," *IEEE Trans. Image Proc.*, pp.49-55, Jan. 1996.

[14] C.D'Elia, G.Poggi: "Self-organizing codebooks for trellis-coded VQ," *IEEE Signal Proc. Letters*, pp.404-406, Dec. 2002.

[15] A.Gersho, R.M.Gray, *Vector quantization and signal compression*, Kluwer, Boston, 1992.

[16] E.de Bodt, M. Cottrell, P. Letremy, M. Verleysen, "On the use of self-organizing maps to accelerate vector quantization", *Neurocomputing*, pp.187-203, 2004.