A Method for Reducing the Computational Complexity of the Encoding Phase of Voice Waveform Vector Quantization

E. N. Arcoverde Neto, A. L. O. Cavalcanti Jr., W. T. A. Lopes, M. S. Alencar and F. Madeiro

***Abstract*—A method for codebook design with the purpose of reducing the computational complexity of the encoding phase of vector quantization (VQ) has been proposed in a recent work. The method consists of an efficient use of a symmetry observed in certain signals. This paper shows that an additional reduction of the computational complexity is obtained when the partial distance search algorithm is incorporated to the previously proposed method for the VQ encoding phase.**

***Index Terms*—Speech coding, vector quantization, computational complexity.**

I. Introduction

VECTOR quantization [1,2] may be defined as a mapping Q of an input vector $\boldsymbol{x}$ belonging to the K-dimensional Euclidean space, $\mathbb{R}^K$, to a vector belonging to a finite subset W of $\mathbb{R}^K$, that is,

$$Q : \mathbb{R}^K \rightarrow W. \quad (1)$$

The codebook $W = \{\boldsymbol{w}_i;\ i = 1,\ 2,\ \ldots,\ N\}$ is the set of reconstruction vectors (codevectors), K is the dimension of the vector quantizer and N is the codebook size, that is, the number of codevectors (or number of levels, in analogy with scalar quantization).

In a signal compression system based on vector quantization (VQ), a vector quantizer may be seen as a combination of two functions: a source encoder and a source decoder. Given a vector $\boldsymbol{x} \in \mathbb{R}^K$ from the source to be encoded, the encoder calculates the distortion $d(\boldsymbol{x}, \boldsymbol{w}_i)$ between the input vector (vector to be quantized) and each vector $\boldsymbol{w}_i,\ i = 1,\ 2,\ \ldots,\ N$ of the codebook W. The optimum rule for encoding is the nearest neighbor rule [3]: a binary representation of the index I, denoted by $\boldsymbol{b}_I$, is transmitted to the source decoder if the codevector $\boldsymbol{w}_I$ corresponds to the minimum distortion, that is, if $\boldsymbol{w}_I$ is the codevector that presents the greatest similarity to $\boldsymbol{x}$ among all the codevectors in the codebook. In other words, the encoder uses the encoding rule $\mathcal{C}(\boldsymbol{x}) = \boldsymbol{b}_I$ if $d(\boldsymbol{x}, \boldsymbol{w}_I) < d(\boldsymbol{x}, \boldsymbol{w}_i)$, $\forall i \neq I$. When the decoder (which has a copy of the codebook W) receives the binary representation $\boldsymbol{b}_I = (b_1(I), b_2(I), ..., b_m(I))$ of the index I, it simply searches for the I-th codevector and produces the vector $\boldsymbol{w}_I$ as the reproduction (quantized version) of $\boldsymbol{x}$. In other words, it uses the following decoding rule: $\mathcal{D}(\boldsymbol{b}_I) = \boldsymbol{w}_I$.

The code rate of a vector quantizer, which measures the number of bits by vector component, is given by $R = \frac{m}{K} = \frac{1}{K}\log_2 N$. In voice waveform coding (*e. g.* [4,5]), R is expressed in bits/sample. In image coding (*e.g.* [6,7]), R is expressed in bits per pixel (bpp).

In [8] a methodology for codebook design with the purpose of reducing the computational complexity of the encoding phase of VQ has been presented. The methodology consists of introducing a structured organization in the designed codebooks, with the objective of reducing the number of multiplications, additions, subtractions and comparisons performed in the minimum distortion encoding phase (nearest neighbor search). In [8] the authors have presented an efficient encoding method, which exploits the structured organization of the designed codebooks. In the present paper that methodology is combined with the partial distance search (PDS) algorithm [9] to obtain an additional reduction of the computational complexity.

The remaining of the paper is organized as follows. Section II presents a brief description of the computational complexity of VQ and describes the PDS algorithm. In order to maintain the paper self-contained, Section III describes the methodology of codebook design proposed in [8]. In Section IV, the method proposed in [8] for reducing the number of operations performed in the encoding phase of VQ is described. Section V presents the method proposed in the present paper for reducing the computational complexity of the encoding phase of VQ. Results and concluding remarks are presented in Sections VI and VII, respectively.

II. Computational Complexity of Vector Quantization

The computational complexity of the encoding phase is a relevant problem for vector quantization. To encode an input vector, the encoder must determine its distance to each one of the N codevectors and must compare the distances in order to find the codevector closest to the input vector, that is, the nearest neighbor.

In the conventional full search (FS) method, the encoding of an input vector requires N distance (distortion) computations and $N-1$ comparisons. When using the squared

E. N. Arcoverde Neto, A. L. O. Cavalcanti Jr. and F. Madeiro are with Department of Statistics and Computer Systems, Catholic University of Pernambuco, Recife, PE, Brazil, e-mail: {euneto,antonio,madeiro}@dei.unicap.br.

W. T. A. Lopes is with Department of Electrical Engineering, Faculty of Science and Technology – AREA1, Salvador, BA, Brazil, e-mail: waslon@area1.br.

M. S. Alencar is with Department of Electrical Engineering, Federal University of Campina Grande, Campina Grande, PB, Brazil, e-mail: malencar@dee.ufcg.edu.br.

Euclidean distortion, that is,

$$d(\boldsymbol{x}, \boldsymbol{w}_i) = \sum_{j=1}^{K} (x_j - w_{ij})^2, \tag{2}$$

where w_{ij} is the j-th component of the codevector $\boldsymbol{w}_i$ and x_j is the j-th component of the input vector $\boldsymbol{x}$, each distance computation requires K multiplications, K subtractions and $K-1$ additions. Thus, to encode each input vector, KN multiplications, KN subtractions, $(K-1)N$ additions and $N-1$ comparisons must be computed. The complexity of a vector quantizer may be alternatively expressed in terms of N multiplications, N subtractions, $(1-\frac{1}{K})N$ additions and $(N-1)/K$ comparisons per sample.

Hence, the computational complexity of a vector quantizer of dimension K and rate R requires a number of operations per sample of the order of $N = 2^{KR}$ for each input vector if a full search (exhaustive search) is performed in the codebook.

A. PDS Algorithm

The partial distance search (PDS) algorithm, proposed by Bei and Gray in [9], is a method for reducing the computational complexity of the nearest neighbor search (encoding phase). The PDS algorithm allows early termination of the distortion calculation between an input vector (vector to be encoded) and a codevector by introducing a condition for premature exit in the search process.

Assume that the smallest distortion is $d_{\min} = \min\{d(\boldsymbol{x}, \boldsymbol{w}_i) | \boldsymbol{w}_i$ has been inspected$\}$. If the uninspected codevector $\boldsymbol{w}_{i'}$ satisfies the condition $\sum_{j=1}^{q} (x_j - w_{i'j}) \geq d_{\min}$, which guarantees that $d(\boldsymbol{x}, \boldsymbol{w}_{i'}) \geq d_{\min}$, the codevector $\boldsymbol{w}_{i'}$ is rejected without computing the distance $d(\boldsymbol{x}, \boldsymbol{w}_{i'})$, where $1 \leq q < K$.

In other words, the encoder decides that a codevector is not the nearest neighbor if, for some $q < K$, the accumulated distance (that is, the partial distance) for the first q components (samples) of the input vector is larger than the smallest distance previously computed in the search. Then, the encoder stops the distance computation for that codevector and starts the distance computation for the next codevector. With this approach, the number of multiplications per sample is dramatically reduced. The PDS algorithm also leads to a reduction in the number of subtractions/additions per sample. Although PDS increases the number of comparisons, the global complexity of the nearest neighbor search is reduced.

III. The Codebook Design

Speech signals present an interesting symmetry: a type of correspondence between the speech vectors, in the sense that for a speech vector $\boldsymbol{x}$ there exists a corresponding vector, which, approximately, equals the symmetric $-\boldsymbol{x}$. In speech signals the symmetry is also observed as follows: approximately half the vectors have a positive mean[1] and approximately half the vectors have a negative mean.

[1] Throughout the text, the mean of a vector must be understood as the arithmetic mean of its components.

TABLE I

Codebook with eight codevectors $\boldsymbol{w}_i = [w_{i1}\ w_{i2}]^T$, with $1 \leq i \leq 8$. The binary word of the i-th codevector, $\boldsymbol{w}_i$, is denoted by $\boldsymbol{b}_i$, while w_{ij} denotes the j-th component of codevector $\boldsymbol{w}_i$.

i	w_{i1}	w_{i2}	$\boldsymbol{b}_i$
1	0.0973	0.0974	000
2	0.5884	0.5905	001
3	0.0151	0.0152	010
4	0.2329	0.2315	011
5	-0.2329	-0.2315	100
6	-0.0151	-0.0152	101
7	-0.5884	-0.5905	110
8	-0.0973	-0.0974	111

The methodology proposed in [8] for codebook design is an attempt to design a codebook which incorporates in its structure the symmetry of the speech signals. The set S of the K-dimensional training vectors is divided into two subsets, S_{POS} and S_{NEG}, where S_{POS} is formed by the training vectors that have a positive mean and S_{NEG} is formed by the training vectors that have a negative mean.

The subset S_{POS} is used for obtaining the first $N/2$ codevectors, by using a codebook design algorithm, such as the LBG (Linde-Buzo-Gray) algorithm [3], unsupervised learning algorithms [10–12] and fuzzy algorithms [13].

The first $N/2$ codevectors, that is, the codevectors $\boldsymbol{w}_i$, $1 \leq i \leq N/2$, have components whose mean value is positive. Those vectors will be denoted by $\boldsymbol{w}_{i,\text{POS}}$, with $1 \leq i \leq N/2$, where the subscript POS stands for the positive mean.

The remaining $N/2$ codevectors of the codebook are obtained as follows. For each codevector $\boldsymbol{w}_{i,\text{POS}}$, a corresponding codevector $\boldsymbol{w}_{N+1-i,\text{NEG}}$ is obtained, such that

$$\boldsymbol{w}_{N+1-i,\text{NEG}} = -\boldsymbol{w}_{i,\text{POS}}, 1 \leq i \leq N/2. \tag{3}$$

Thus, the last $N/2$ codevectors of the codebook have components whose mean is negative. Those vectors will be denoted by $\boldsymbol{w}_{i,\text{NEG}}$, with $N/2+1 \leq i \leq N$, where the subscript NEG stands for the negative mean.

Hence, according to Equation (3), the codebooks obtained with the previously described methodology present a remarkable symmetry: a codevector $\boldsymbol{w}_i$, $1 \leq i \leq N/2$ has a corresponding codevector $\boldsymbol{w}_{N+1-i}$ such that $\boldsymbol{w}_{N+1-i} = -\boldsymbol{w}_i$.

Table I presents a codebook of eight codevectors obtained with the methodology described in [8]. The first $N/2$ codevectors were obtained by using as training set the vectors belonging to S_{POS}. It is observed that the codevectors incorporate the symmetry of the speech signals: half the codevectors have a positive mean and half the vectors have a negative mean; each codevector has its corresponding symmetric codevector.

Due to the symmetry introduced in the codebook, only half the codevectors must be stored. In fact, by storing only the codevectors $\boldsymbol{w}_{i,\mathrm{POS}}$, the codevectors $\boldsymbol{w}_{i,\mathrm{NEG}}$ can be easily determined. This leads to half the conventional memory requirements to store a codebook. The symmetry of the codebook has also led to a method for reducing the computational complexity of the encoding phase (nearest neighbor search) of VQ. In that method, only half the codebook, corresponding to the codevectors $\boldsymbol{w}_{i,\mathrm{POS}}$, is effectively stored in the reference memory of the encoder. The decoder, by its turn, has all (N) codevectors of the codebook.

IV. Method For Reducing The Computational Complexity of VQ

The encoding method proposed in [8] for reducing the number of operations performed to encode an input vector (vector of the source to be encoded) $\boldsymbol{x}$ is described as follows: given $\boldsymbol{x}$, if $\mathrm{mean}(\boldsymbol{x}) \geq 0$ then[2] the encoder performs a search for the nearest neighbor of $\boldsymbol{x}$ only in the codevectors $\boldsymbol{w}_{i,\mathrm{POS}}$, that is, in the codevectors effectively stored in the reference memory of the encoder. Then, the encoder sends to the decoder a binary word beginning with 0 (indicating to the decoder that the codevector to be produced as the representation of $\boldsymbol{x}$ is a $\boldsymbol{w}_{i,\mathrm{POS}}$ vector), followed by $\log_2(N/2)$ bits needed to represent the index i of the vector $\boldsymbol{w}_{i,\mathrm{POS}}$ selected from the reference memory. On the other hand, if $\mathrm{mean}(\boldsymbol{x}) < 0$, the search for the nearest neighbor should be performed among the vectors $\boldsymbol{w}_{i,\mathrm{NEG}}$. Since those vectors are not stored in the reference memory of the encoder, the encoding consists of comparing the vector $-\boldsymbol{x}$ (symmetric vector of $\boldsymbol{x}$) with the codevectors $\boldsymbol{w}_{i,\mathrm{POS}}$. Once the closest one to $-\boldsymbol{x}$ is determined, the encoder sends to the decoder a binary word beginning with 1 (indicating to the decoder that the codevector to be produced as the representation of $\boldsymbol{x}$ is a $\boldsymbol{w}_{i,\mathrm{NEG}}$ vector), followed by a sequence of $\log_2(N/2)$ bits: each bit of this sequence is the complement of the corresponding bit of the sequence of $\log_2(N/2)$ bits needed to represent the index i of the vector $\boldsymbol{w}_{i,\mathrm{POS}}$ selected as the closest one to $-\boldsymbol{x}$ according to the minimum distortion criterion. It is important to note that, due to the symmetry introduced in the codebook, if $\boldsymbol{x}$ has $\boldsymbol{w}_{N+1-i,\mathrm{NEG}}$ as the nearest neighbor, then $-\boldsymbol{x}$ has $\boldsymbol{w}_{i,\mathrm{POS}} = -\boldsymbol{w}_{N+1-i,\mathrm{NEG}}$ as the nearest neighbor.

To illustrate the encoding method, consider the codebook of Table I (available to the decoder). The reference memory of the encoder, by its turn, concerns Table II, corresponding to the first $N/2$ codevectors ($\boldsymbol{w}_{i,\mathrm{POS}}$ vectors) in Table I. Suppose that the communication system receives $\boldsymbol{x} = [0.0121\ 0.0109]^T$ as the input vector. After evaluating the mean[3] of the components of the input vector, the encoder decides that the codevector to be selected as the representation (quantized version) of $\boldsymbol{x}$ is a $\boldsymbol{w}_{i,\mathrm{POS}}$ vector. It is determined that the first bit of the binary word to be transmitted to the decoder is 0. The nearest neighbor search is then performed in the codebook effectively stored (Table II) in the reference memory of the encoder. Following the minimum distortion criterion, vector $[0.0151\ 0.0152]^T$, with binary representation 10, is selected. Hence, the encoder transmits to the decoder the binary word 010, where the first bit informs that the quantized version of $\boldsymbol{x}$ is a $\boldsymbol{w}_{i,\mathrm{POS}}$ vector (that is, one of the first $N/2$ vectors of the codebook of N vectors in the decoder) and the last two bits are the binary representation of the index of the vector selected (from the table of the encoder) as the closest one (nearest neighbor) to $\boldsymbol{x}$. On the other side of the communication system based on VQ, when the decoder (which has the codebook of Table I) receives the binary word 010, it simply outputs the vector $[0.0151\ \ 0.0152]^T$.

Now suppose that the communication system receives $\boldsymbol{x} = [-0.5765\ \ -0.4902]^T$ as the input vector. After evaluating the mean of the input vector, the encoder decides that the quantized version of $\boldsymbol{x}$ is a $\boldsymbol{w}_{i,\mathrm{NEG}}$ vector. It is determined that the first bit of the binary word to be transmitted to the decoder is 1. The search for the nearest neighbor of $-\boldsymbol{x}$ is then performed in the $N/2$ codevectors available (Table II) to the encoder: the codevector $[0.5884\ \ 0.5905]^T$ is selected since it is the closest one to $[0.5765\ \ 0.4902]^T = -\boldsymbol{x}$. Thus, the encoder sends to the decoder the binary word 110, where the first bit informs that the codevector selected as the quantized version of the input vector (source vector) $\boldsymbol{x}$ is a $\boldsymbol{w}_{i,\mathrm{NEG}}$ vector and the last two bits correspond to the complement of the binary word 01 of the vector $[0.5884\ \ 0.5905]^T$ in the codebook effectively available to the encoder. In the other side of the communication system, when the decoder receives the binary word 110, it outputs the vector $[-0.5884\ \ -0.5905]^T$.

TABLE II

Two-dimensional codevectors effectively stored in the reference memory of the encoder.

i	w_{i1}	w_{i2}	Binary representation
1	0.0973	0.0974	00
2	0.5884	0.5905	01
3	0.0151	0.0152	10
4	0.2329	0.2315	11

The encoding method proposed in [8] will be referred to as 1/2COD, since the encoder uses only half the codebook. Table III presents a summary of the total number of operations performed by the conventional full search (FS) method (carried out in a codebook with N codevectors) and 1/2COD.

V. Algorithm 1/2COD+PDS

This work presents the method 1/2COD+PDS, which is an attempt to reduce the computational complexity of the encoding phase of VQ. The method consists of combining

[2] For a vector $\boldsymbol{x} = [x_1 x_2 \cdots x_K]^T$, where T is the transposition operation, $\mathrm{mean}(\boldsymbol{x}) = \frac{1}{K}\sum_{j=1}^{K} x_j$.

[3] If the sum of all components of a vector is positive then the arithmetic mean of the components is also positive. So, instead of determining the mean value of the vector components, the encoding method computes only the sum of the vector components. This leads to saving one division for each input vector.

TABLE III
NUMBER OF OPERATIONS PERFORMED BY FS AND 1/2COD TO ENCODE A VECTOR, AS A FUNCTION OF K AND N [8].

	Number of operations	
	FS	1/2COD
$\times$	KN	$KN/2$
-	KN	$KN/2$
$+$	$(K-1)N$	$(K-1)(1+N/2)$
Comp.	$N-1$	$N/2$

the encoding systematic proposed in [8] and described in Section 4 with the PDS algorithm. In other words, the method 1/2COD+PDS uses the PDS algorithm for determining the nearest neighbor of the input vector (if this vector has a positive mean) or of its symmetric vector (if the input vector has a negative mean) in the codebook available to the encoder (this codebook has only half the codevectors of the codebook available to the decoder, which has been designed as described in Section 3). Hence, the method outperforms 1/2COD in terms of reduction of the number of operations of the encoding phase of VQ.

VI. Results

This section presents results concerning voice waveform vector quantization. The acquisition (resolution of 8.0 bit/sample and sampling rate of 8 kHz) of the speech signals was performed by using a Sun workstation with audio processing tools.

All codebooks were designed using a training set of 150,080 samples, corresponding to 18.76 seconds. The codebooks were designed by using competitive learning for various values of dimension (K) and number of levels (N).

Table IV shows that the 1/2COD algorithm presents savings of 50% in terms of the number of multiplications when compared to FS. For different values of K and N corresponding to a code rate of 1.0 bit/sample, this table also shows that the PDS algorithm outperforms 1/2COD in terms of number of multiplications per sample. It is also observed in the table that the best performance is obtained by using 1/2COD+PDS. As an example, for $K = 8$ and $N = 256$ it is observed that 1/2COD+PDS achieves savings of 88.11% when compared to FS algorithm. In this case, PDS and 1/2COD saves 77.29% and 50.00%, respectively, in terms of the number of multiplications per sample. The table shows that the introduction of the PDS algorithm in the 1/2COD methodology leads to an additional savings in the number of multiplications. In fact, for $K = 6$ and $N = 64$, 1/2COD+PDS leads to savings of 81.34% while 1/2COD leads to 50.00% when compared to FS. The table also shows that the larger the codebook size (N) the greater is the superiority of 1/2COD+PDS over 1/2COD.

Table 5, which considers $K = 2$ and various values of N, reveals that the best performance in terms of the number of multiplications per sample is obtained by 1/2COD+PDS. The table also shows that the 1/2COD performance is better than the PDS performance in terms of the number of multiplications. Moreover, the superiority of 1/2COD+PDS over 1/2COD increases with the codebook size (N).

Table 6, which considers $K = 4$ and various values of N, shows that the best results in terms of savings in the number of multiplications per sample are obtained by 1/2COD+PDS. It is observed in Table 6 that PDS outperforms 1/2COD regarding savings in the number of multiplications per sample.

It is worth mentioning that, according to [9], the number of multiplications is generally used as the evaluation criterion of the computational complexity of the VQ encoding phase. This comes from the fact that the multiplication operation requires a larger computational effort when compared to the addition, subtraction or comparison.

VII. Concluding Remarks

This work presented a method for reducing the computational complexity of the encoding phase of vector quantization (VQ). The proposed method combines two techniques. The first one exploits a symmetry imposed in the voice waveform VQ codebook. By using this symmetry, the nearest neighbor search is performed by comparing the input vectors with only half the codevectors. The second technique corresponds to the partial distance search method. Simulations concerning voice waveform coding have shown that the combination of those techniques is a suitable method for reducing the computational complexity of the encoding phase of VQ.

As a future work, the authors will investigate the combination of the proposed method with techniques for codebook ordering with the purpose of obtaining an additional reduction in the computational complexity of the minimum distortion encoding of vector quantization.

References

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

[2] R. M. Gray. "Vector Quantization". *IEEE ASSP Magazine*, pp. 4–29, April 1984.

[3] Y. Linde, A. Buzo and R. M. Gray. "An Algorithm for Vector Quantizer Design". *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, January 1980.

[4] H. Abut, R. M. Gray and G. Rebolledo. "Vector Quantization of Speech and Speech-Like Waveforms". *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 30, no. 3, pp. 423–435, June 1982.

[5] A. Gersho and V. Cuperman. "Vector Quantization: A Pattern-Matching Technique for Speech Coding". *IEEE Communications Magazine*, pp. 15–20, December 1983.

[6] B. Ramamurthi and A. Gersho. "Classified Vector Quantization of Images". *IEEE Transactions on Communications*, vol. 34, no. 11, pp. 1105–1115, November 1986.

[7] N. M. Nasrabadi and R. A. King. "Image Coding Using Vector Quantization: A Review". *IEEE Transactions on Communications*, vol. 36, no. 8, pp. 957–971, August 1988.

[8] F. Madeiro, W. T. A. Lopes, B. G. Aguiar Neto and M. S. Alencar. "Construção de Dicionários Voltados para a Redução da Complexidade Computacional da Etapa de Codificação da Quantização Vetorial". In *Anais do VI Congresso Brasileiro de Redes Neurais (CBRN'03)*, pp. 439–444, São Paulo – SP, Brasil, Junho 2003.

TABLE IV

Number of multiplications per sample for FS, PDS, 1/2COD and 1/2COD+PDS algorithms. The savings with respect to FS is within parentheses. It was considered a 1.0 bit/sample vector quantization.

K	N	Number of multiplications per sample			
		FS	PDS	1/2COD	1/2COD+PDS
5	32	32	12.85 (59.84%)	16 (50.00%)	7.49 (76.59%)
6	64	64	19.37 (69.73%)	32 (50.00%)	11.94 (81.34%)
7	128	128	32.02 (74.98%)	64 (50.00%)	19.42 (84.83%)
8	256	256	58.14 (77.29%)	128 (50.00%)	30.43 (88.11%)

TABLE V

Number of multiplications per sample for FS, PDS, 1/2COD and 1/2COD+PDS algorithms. The savings with respect to FS is within parentheses. Codebooks with $K = 2$ were considered.

K	N	Number of multiplications per sample			
		FS	PDS	1/2COD	1/2COD+PDS
2	32	32	18.60 (41.88%)	16 (50.00%)	10.39 (67.53%)
2	64	64	36.88 (42.38%)	32 (50.00%)	20.34 (68.22%)
2	128	128	69.33 (45.84%)	64 (50.00%)	36.42 (71.55%)
2	256	256	137.30 (46.37%)	128 (50.00%)	71.34 (72.13%)

TABLE VI

Number of multiplications per sample for FS, PDS, 1/2COD and 1/2COD+PDS algorithms. The savings with respect to FS is within parentheses. Codebooks with $K = 4$ were considered.

K	N	Number of multiplications per sample			
		FS	PDS	1/2COD	1/2COD+PDS
4	32	32	11.35 (64.53%)	16 (50.00%)	8.51 (73.41%)
4	64	64	22.24 (65.25%)	32 (50.00%)	14.22 (77.78%)
4	128	128	42.11 (67.10%)	64 (50.00%)	24.88 (80.56%)
4	256	256	82.86 (67.63%)	128 (50.00%)	44.35 (82.68%)

[9] C.-D. Bei and R. M. Gray. "An Improvement of the Minimum Distortion Encoding Algorithm for Vector Quantization". *IEEE Transactions on Communications*, vol. 33, no. 10, pp. 1132–1133, October 1985.

[10] T. Kohonen. "The Self-Organizing Map". *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, September 1990.

[11] A. K. Krishnamurthy, S. C. Ahalt, D. E. Melton and P. Chen. "Neural Networks for Vector Quantization of Speech and Images". *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 8, pp. 1449–1457, October 1990.

[12] E. Yair, K. Zeger and A. Gersho. "Competitive Learning and Soft Competition for Vector Quantizer Design". *IEEE Transactions on Signal Processing*, vol. 40, no. 2, pp. 294–309, February 1992.

[13] N. B. Karayiannis and P.-I. Pai. "Fuzzy Vector Quantization Algorithms and Their Applications in Image Compression". *IEEE Transactions on Image Processing*, vol. 4, no. 9, pp. 1193–1201, September 1995.