Minimax Partial Distortion Competitive Learning for Optimal Codebook Design

Ce Zhu and Lai-Man Po, *Member, IEEE*

***Abstract*— The design of the optimal codebook for a given codebook size and input source is a challenging puzzle that remains to be solved. The key problem in optimal codebook design is how to construct a set of codevectors efficiently to minimize the average distortion. A *minimax* criterion of minimizing the maximum partial distortion is introduced in this paper. Based on the partial distortion theorem, it is shown that minimizing the maximum partial distortion and minimizing the average distortion will asymptotically have the same optimal solution corresponding to equal and minimal partial distortion. Motivated by the result, we incorporate the alternative *minimax* criterion into the on-line learning mechanism, and develop a new algorithm called *minimax partial distortion competitive learning* (MMPDCL) for optimal codebook design. A computation acceleration scheme for the MMPDCL algorithm is implemented using the partial distance search technique, thus significantly increasing its computational efficiency. Extensive experiments have demonstrated that compared with some well-known codebook design algorithms, the MMPDCL algorithm consistently produces the best codebooks with the smallest average distortions. As the codebook size increases, the performance gain becomes more significant using the MMPDCL algorithm. The robustness and computational efficiency of this new algorithm further highlight its advantages.**

***Index Terms*— Codebook design, competitive learning, vector quantization.**

I. Introduction

Vector quantization (VQ) has been widely applied to signal compression for reducing the bit rate required to represent signals for transmission or archiving [1], [2]. It has been shown that VQ is very effective at low to medium compression ratio while maintaining an acceptable fidelity [3], [4]. A vector quantizer can be defined as a mapping Q of k-dimensional Euclidean space R^k into a finite subset Y of R^k, i.e., Q: $R^k \rightarrow Y$, where $Y = \{\boldsymbol{y}_i \in R^k | i = 1, 2, \cdots, N\}$ is called a codebook, $\boldsymbol{y}_i$ is a codevector and N is the codebook size. From the definition, we can see that VQ is a lossy compression method, and accordingly some distortion measurements have been introduced with respect to different applications. The most commonly used distortion measure is the simple mean squared error (MSE) per sample, defined by $\mathrm{MSE} = (1/k)\ E[\|\boldsymbol{x} - Q(\boldsymbol{x})\|^2]$ where $\boldsymbol{x} \in R^k$ is an input random vector. For an unknown distribution input sequence $\{\boldsymbol{x}_i \in R^k | i = 1, 2, \cdots, M\}$, the MSE is always approximated by the time-averaged square error distortion given by $\mathrm{MSE} \approx (1/kM)\ \Sigma_{i=1}^{M}\ \|\boldsymbol{x}_i - Q(\boldsymbol{x}_i)\|^2$. In this paper we also use the MSE as the distortion measurement. For a given codebook size and input source, the optimal vector quantizer Q is the one which minimizes the MSE.

The design of a vector quantizer Q involves two aspects of the encoder and the decoder design. For a given decoder (i.e., the codebook is fixed), the best encoder should satisfy the nearest neighbor (NN) condition for partitioning the input space, while for a given encoder (i.e., the partition is fixed), the centroid condition is found to be both necessary and sufficient for optimizing the codebook in the sense of minimizing the MSE distortion. With the nearest neighbor encoding rule, the performance of the quantizer Q in terms of MSE distortion is dependent on its codebook. Hence, the design of optimal codebook is of prime importance in VQ. The generalized Lloyd algorithm (GLA) [5] was developed for codebook design which employs a batch processing iteration procedure based on the two necessary conditions of optimality. An open question regarding the GLA algorithm is the initialization of codevectors. The performance of the GLA algorithm strongly depends on the initial codebook. With improper initial locations of the codevectors, the GLA may lead to a local optimality corresponding to a low quality codebook. To obtain a high-quality codebook, some optimizing techniques have been developed, such as stochastic relaxation [6]. However, these techniques are typically very time consuming and substantially increase the complexity, especially when the designed codebook size is very large.

Recently, neural network based competitive learning (CL) algorithms have been developed for codebook design that are characterized by an on-line learning mechanism [7]–[16]. In contrast to the batch-processing GLA algorithm, the CL algorithms design the codebook on-line by updating the codevectors for each presentation of an input vector. The update rule for a typical CL algorithm is

$$\boldsymbol{y}_i(t+1) = \boldsymbol{y}_i(t) + S_i(\boldsymbol{x}(t))\alpha_i(t)[\boldsymbol{x}(t) - \boldsymbol{y}_i(t)] \qquad (1)$$

where $\boldsymbol{x}(t)$ is the current input vector, and $\alpha_i(t)$ is the learning rate for codevector $\boldsymbol{y}_i$ at time t. The selector function $S_i(\boldsymbol{x}(t))$ has a value of one, if $\boldsymbol{y}_i$ is the winner (e.g., the nearest neighbor codevector in the basic CL) and zero, otherwise. The main problem of the basic CL algorithm is that it may lead to a local minimum with high overall distortion,

Manuscript received October 22, 1996; revised December 5, 1997. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Antonio Ortega.

C. Zhu is with the Computer Science Department, Southwest China Normal University, Beibei, Chongqing 400715, China.

L.-M. Po is with the Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong.

Publisher Item Identifier S 1057-7149(98)06809-2.

even leading to the underutilization of some codevectors, in the case of improper initialization. Kohonen's self-organizing map (SOM) [17] employs the neighborhood effect as leaky learning for the topology preserving map, which alleviates the problem of some codevectors being underused. Desieno [7] proposed a heuristic CL algorithm based on the history-related "conscience" mechanism to overcome the underutilization problem. Moreover, the frequency-sensitive competitive learning (FSCL) algorithm [9], [10] is a straightforward and simple implementation of the "conscience" scheme, in which each competing codevector assumes an approximately uniform win rate to thoroughly eliminate the underuse problem. Some other improved CL algorithms utilize the soft competition scheme [11], [12] or the fuzzy mechanism [13], [14] to adjust all the codevectors with different adaptation weights for each presentation of an input vector. However, determining the weights requires additional time-consuming computations such as power operations.

Nevertheless, these competitive learning algorithms which aim at "fairly" utilizing each codevector cannot generally produce the optimal codebook corresponding to the minimum MSE. Naturally, the question is raised, how can the codevectors be utilized most efficiently for minimizing the MSE function? Concerning optimal quantization, a valuable contribution due to Gersho [18], which has come to be known as the *partial distortion theorem* [19], presents a theoretical guide to achieving the asymptotically optimal partition of an input space for codebook design. Inspired by this theorem, some preliminary prototypes of partial distortion sensitive CL algorithms for VQ were proposed by Zhu *et al.* [14]–[16]. Furthering the early work thoroughly, we propose in this paper an effective *minimax* criterion of minimizing the maximum partial distortion for optimal codebook design with a given codebook size, where the codevector indices are not entropy coded. The corresponding on-line competitive learning algorithm called *minimax partial distortion competitive learning* (MMPDCL) with its computation acceleration scheme is further developed. The effectiveness of the proposed algorithm is investigated in comparison with some other codebook design algorithms.

The remainder of this paper is organized as follows. In the following section, high-resolution optimal quantization and the asymptotic result, the partial distortion theorem, are briefly formulated as the fundamentals for designing the optimal codebook. The proposed minimax partial distortion criterion and the new MMPDCL algorithm with its computation acceleration scheme are presented in Section III. Extensive experiments on the Gauss–Markov process, speech and images are reported and discussed using different codebook design algorithms for comparison in Section IV. Concluding remarks are given in Section V.

II. High-Resolution Optimal Quantization

The average (mean squared error) distortion of a vector quantizer Q can be given by

$$D = \sum_{i=1}^{N} D_i \tag{2}$$

$$D_i = E[||\boldsymbol{x} - Q(\boldsymbol{x})||^2 | \boldsymbol{x} \in R_i] P(\boldsymbol{x} \in R_i) \tag{3}$$

$$Q(\boldsymbol{x}) = \boldsymbol{y}_i \quad \forall \boldsymbol{x} \in R_i \tag{4}$$

where N is the codebook size, R_i denotes the partition region corresponding to its representative codevector $\boldsymbol{y}_i$, and $P(\boldsymbol{x} \in R_i)$ is the probability that $\boldsymbol{x}$ is located in R_i. The term D_i is called the partial distortion in the region R_i for the codevector $\boldsymbol{y}_i$.

We consider the important case of high resolution regular quantization, where N is very large, the input probability density function (pdf) is reasonably smooth, and the MSE distortion is much less than input variance, so that the mathematical derivation of the performance results is tractable [18], [19]. According to [19], a signal-to-noise ratio (SNR) value of 10 dB is a borderline case between low and high resolution for scalar quantization. Therefore, most applications of quantization belong to the range of high resolution quantization. This approach to performance analysis, known as the asymptotic quantization approach [19], yields useful approximate results referred to as "asymptotic" results that become increasingly accurate as the resolution, R, or codebook size, $N = 2^R$, increases.

A. Optimal Output Point Density

In [18], the *output point density* of a vector quantizer at any point in the input space is defined as the number of codevectors per unit volume at that point, which is approximated as a continuous density function for sufficiently large N. We aim at achieving the *optimal output point density* in the form of optimal codebook corresponding to the minimum average distortion. It is shown in [18] that for large N, the output point density of the optimal codevectors is proportional to the $(k/(k+2))$th power of the input probability density under the MSE distortion measure. This result demonstrates that the optimal distribution of reproduction codevectors is not the same as the probability distribution of the input vectors for finite dimension k. Only as k approaches infinity does the optimal point density tend to become proportional to the input probability density, and in such case the optimal codebook will asymptotically result in uniform utilization of each codevector. Therefore, in the general case of finite dimension, the optimal codebook does not correspond to the equal utilization of each codevector except in some special cases, e.g., the uniform input distribution.

B. Partial Distortion Theorem

Another valuable asymptotic result is that *each partition region makes an equal contribution to the distortion for an optimal quantizer with sufficiently large* N [18], that is, *the partial distortions* D_i *in each quantization region* R_i *are the same as a constant independent of* i. This conclusion has recently become known as the partial distortion theorem [19, p. 187]. It has been used as a criterion for subcodebook size optimization in the classified VQ [20]. In fact, this asymptotic result of uniform partial distortion can be regarded as an additional rule for designing the optimal codebook of large size in addition to the well-known nearest neighbor and centroid conditions.

III. Minimax Partial Distortion Competitive Learning

A. Minimax Partial Distortion Criterion

From the partial distortion theorem, we know that an asymptotically necessary condition for optimality of minimizing MSE in VQ is to have equal partial distortion. This condition is obviously not sufficient. One has to find the solution with equal and minimal partial distortion to minimize MSE at high rates. Considering the *minimax* criterion of minimizing the maximum partial distortion

$$D' = \max_{i\in\{1,2,\cdots,N\}} (D_i) \tag{5}$$

where D_i is the partial distortion shown in (3), we have the following result which can be regarded as a corollary of the partial distortion theorem.

Corollary 1: For sufficiently large N, the optimal solution to minimizing the maximum partial distortion D' will have the equal partial distortion property and will be the same as the optimal solution to minimizing the average MSE distortion D.

Proof: It is obvious that of all solutions having the same average distortion, the one with minimum D' is always (i.e., not necessarily asymptotically) that with uniform partial distortion provided such a solution of uniform partial distortion exists. Furthermore, of all solutions having different average distortions, the one with minimum average distortion and equal partial distortion (if such solution exists) is that corresponding to minimum D'. According to the partial distortion theorem, for sufficiently large N, the solution with minimum MSE distortion has the property of equal partial distortion. Therefore, for sufficiently large N, the solution with minimum D' must be the one with equal partial distortion and minimum MSE distortion. The converse is also true. Hence, the two different minimizations asymptotically result in the same optimal solution. This completes the proof.

The corollary reveals the relationship between the two problems of minimizing the maximum partial distortion and minimizing the overall MSE distortion. It can be seen that the proposed *minimax partial distortion* rule is asymptotically necessary as well as asymptotically sufficient for minimizing the average distortion. Recall that a similar minimax criterion is also used in filter design [21]. In nonrecursive filter design, we apply the minimax criterion to minimize the largest ripples for a given frequency specification and filter order, which results in the Chebyshev solution of equiripple. In vector quantization, we employ the minimax scheme to minimize the maximum partial distortion for a given training set and codebook size which aims at obtaining the optimal codebook corresponding to uniform and minimal partial distortion. According to the corollary, the optimal codebook minimizing the maximum partial distortion is a good approximation to that minimizing the average MSE distortion, especially for large codebook size N.

Motivated by the result, we incorporate the *minimax partial distortion* criterion into the on-line competitive learning mechanism for constructing the optimal codebook with minimum average distortion.

B. On-Line Estimates of Partial Distortions

In competitive learning, we estimate on-line partial distortions using the presented samples under the usual ergodicity assumption. At time T, the partial distortion D_i can be estimated on-line as

$$\hat{D}_i(T) = \frac{1}{T}\sum_{t=1}^{T} S_i(\boldsymbol{x}(t))\|\boldsymbol{x}(t)-\boldsymbol{y}_i\|^2 \tag{6}$$

using the estimates

$$E[\|\boldsymbol{x}-Q(\boldsymbol{x})\|^2|\boldsymbol{x}\in R_i] \approx \sum_{t=1}^{T} S_i(\boldsymbol{x}_i(t))\,\|\boldsymbol{x}(t)-\boldsymbol{y}_i\|^2 \Big/ \sum_{t=1}^{T} S_i(\boldsymbol{x}_i(t))$$

and

$$P(\boldsymbol{x}\in R_i) \approx \frac{1}{T}\sum_{t=1}^{T} S_i(\boldsymbol{x}(t))$$

in which the selector function $S_i(\boldsymbol{x}(t))$ is defined as aforementioned. For computational convenience, we use an equivalent term $pd_i(t)$ to represent the on-line partial distortion estimate at time t

$$pd_i(T) = T\times\hat{D}_i(T) = \sum_{t=1}^{T} S_i(\boldsymbol{x}(t))\|\boldsymbol{x}(t)-\boldsymbol{y}_i\|^2.$$

Hence, $pd_i(t)$ can be simply adapted by

$$pd_i(t) = pd_i(t-1) + S_i(\boldsymbol{x}(t))\|\boldsymbol{x}(t)-\boldsymbol{y}_i\|^2. \tag{7}$$

We know that in the on-line learning process the codevectors change over time. To save computation as well as to maintain the on-line characteristic of no storage requirement for the training set, we always use the most recently updated codevectors for calculation.

C. On-Line Implementation of Minimax Partial Distortion Criterion

With the minimax partial distortion criterion in the on-line learning, we aim to realize the minimization of the maximum partial distortion for each presentation of an input vector. Specifically, we will determine the selector function $S_i(\boldsymbol{x}(t))$ for each presentation $\boldsymbol{x}(t)$ to achieve the minimax result.

If the codevector $\boldsymbol{y}_{i^*}$ is chosen as the winner for the current input $\boldsymbol{x}(t)$, i.e., $S_{i^*}(\boldsymbol{x}(t)) = 1$, its corresponding partial distortion estimate is updated as $pd_{i^*}(t) = pd_{i^*}(t-1) + \|\boldsymbol{x}(t)-\boldsymbol{y}_{i^*}\|^2$, while the partial distortions for the nonwinner codevectors remain unchanged, $pd_j(t) = pd_j(t-1), j\neq i^*$. So the updated $pd_{i^*}(t)$ may be maximal compared with the other partial distortions $pd_j(t), j\neq i^*$. To ensure the minimization of the maximum partial distortion for each input, we should choose the codevector $\boldsymbol{y}_i$ as the winner, which yields the minimum of updated partial distortion compared with choosing any other codevector as the winner. Accordingly, the selector function becomes

$$S_i(\boldsymbol{x}(t)) = \begin{cases} 1, & \text{if } pd_i(t-1) + \|\boldsymbol{x}(t) - \boldsymbol{y}_i\|^2 < pd_j(t-1) \\ & \quad + \|\boldsymbol{x}(t) - \boldsymbol{y}_j\|^2 \quad \text{for any } j \neq i \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

To obtain codebook optimality, the nearest neighbor partition rule and the corresponding centroid rule should be strictly satisfied by the final code. For this reason, various competitive learning algorithms gradually turn into the basic CL algorithm with time for the nearest neighbor partition. The soft competition algorithm gradually evolves into the basic CL learning as the temperature decreases with time [11], and the fuzzy CL algorithms change into the basic CL algorithm with decreasing fuzziness [13], [14]. The optimization techniques such as the stochastic relaxation with decreasing perturbation [6], [19] will also eventually evolve into the basic learning scheme based on the nearest neighbor partition and the centroid conditions. Therefore, we also adopt a similar mechanism in the learning process. In the early stage of codebook training process, we apply the rule of minimizing the maximum partial distortion for partition to produce good locations of the codevectors corresponding to the approximately uniform partial distortions. It could be argued that this early training process led to the production of a superior initial codebook. As the training progresses, the partition should become gradually dominated by the nearest neighbor rule for fine tuning to obtain the final optimal codevectors. Based on this expectation, we employ the selector function

$$S_i(\boldsymbol{x}(t)) = \begin{cases} 1, & \text{if } pd_i(t-1) \times e^{-t/T} + \|\boldsymbol{x}(t) - \boldsymbol{y}_i\|^2 < pd_j(t-1) \\ & \quad \times e^{-t/T} + \|\boldsymbol{x}(t) - \boldsymbol{y}_j\|^2 \\ & \quad \quad \text{for any } j \neq i \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

in which we introduce an attenuation factor $e^{-t/T}$ for $pd_i(t-1)$. T (>0) is a constant to be determined experimentally. In the early stage of training, the attenuation factor $e^{-t/T}$ is close to one for small t, thus the partition is performed as in (8) for minimizing the maximum partial distortion. As t increases, $e^{-t/T}$ approaches zero, and then the nearest neighbor partition rule is asymptotically satisfied.

D. Minimax Partial Distortion Competitive Learning Algorithm

From the partition rule in (9), the corresponding algorithm called *minimax partial distortion competitive learning* (MMPDCL) algorithm may be developed. The detailed steps of the proposed MMPDCL algorithm are as follows.

Step 1 (Initialization): Generate an initial codebook $Y = \{\boldsymbol{y}_t(0), i = 1, \cdots, N\}$ of size N randomly, and set each partial distortion $pd_i(t) = 0$ at time $t = 0$.

Step 2 (Distortion Measuring): Apply an input vector $\boldsymbol{x}(t)$ and calculate the distortion for each codevector

$$\begin{aligned} d_i(t) = & pd_i(t-1) \times e^{-m/T} + \|\boldsymbol{x}(t) - \boldsymbol{y}_i(t-1)\|^2 \\ & i = 1, 2, \cdots, N \end{aligned} \tag{10}$$

where m denotes the sweep index. (A sweep is one full cycle through the training set.)

Step 3 (Winner Selection): Select the winner index $i^* = \underset{i}{\operatorname{argmin}}\, d_i(t)$, and then $S_{i^*}(\boldsymbol{x}(t)) = 1\ S_j(\boldsymbol{x}(t)) = 0, \quad$ for $j \neq i^*$.

Step 4 (Adaptation): Adjust the codevectors according to

$$\boldsymbol{y}_i(t) = \boldsymbol{y}_i(t-1) + \alpha_i(t) S_i(\boldsymbol{x}(t))[\boldsymbol{x}(t) - \boldsymbol{y}_i(t-1)] \tag{11}$$

where $\alpha_i(t)$ is the learning rate of codevector $\boldsymbol{y}_i$ at time $t, i = 1, 2, \cdots, N$.

Update the value of partial distortion $pd_i (i = 1, 2, \cdots, N)$ as

$$pd_i(t) = pd_i(t-1) + S_i(\boldsymbol{x}(t)) \|\boldsymbol{x}(t) - \boldsymbol{y}_i(t)\|^2. \tag{12}$$

In fact, only the winner $\boldsymbol{y}_{i^*}$ and its corresponding partial distortion pd_{i^*} have been changed.

Step 5 (Iteration): Set $t{+}1 \rightarrow t$, and repeat Step 2) through Step 4) for all the training vectors, until a terminating condition is satisfied, e.g., the change of the quantization distortion between two sweeps is less than a small value ε, or the total number of sweeps reaches a predetermined value.

To meet the centroid condition for the final code, the learning rate is $\alpha_i(t) = 1/u_i(t)$, where $u_i(t)$ is the number of times that the codevector $\boldsymbol{y}_i$ has won until time t. (The detailed analysis for the learning rate can be found in [11].) We also employ the reinitialization technique [11] for the learning rate except for a slight modification. When the current sweep index equals a perfect square, the counters $u_i(t)$ are reset to a gradually increasing number (e.g., $2 + (int)(Sweep\ Number/10)$ as employed in our experiment) instead of being reinitialized to unity as adopted in [11]. This modified reinitialization scheme keeps the learning rate from being too small and so speeds up convergence. Also, it maintains the good codevector distribution gradually formed in the training process by avoiding large changes in the codevector locations due to decreasing the reinitialized learning rate.

From the above procedure, we can see that compared with the basic CL algorithm or GLA, for each presentation of a training vector, N more multiplications and additions in (10) are required while one more Euclidean distance calculation and an addition are attached in (12) by the MMPDCL algorithm. The calculation of the exponential $e^{-m/T}$ is performed once for each sweep, and within one sweep the attenuation factor is unchanged. These additional computations are inexpensive in comparison to the N Euclidean distance computations in (10) for each presentation of a training vector.

E. A Computation Acceleration Scheme for the MMPDCL Algorithm

The most computationally intensive part of the MMPDCL algorithm is the distortion calculation between an input vector and each codevector in Step 2) for finding the minimum distortion. The proposed distortion metric employed in (10) is just an additive combination of the weighted partial distortion $pd_i(t-1) \times e^{-m/T}$ and the Euclidean distance $||\boldsymbol{x}(t)-\boldsymbol{y}_i(t-1)||^2$. The Euclidean distance is also an additive combination of each component distance corresponding to each dimension of the vectors, i.e.,

$$||\boldsymbol{x}(t)-\boldsymbol{y}_i(t-1)||^2 = \sum_{j=1}^{k} (\boldsymbol{x}^j(t)-\boldsymbol{y}_i^j(t-1))^2$$

where k is the dimensionality of vectors. For such an additive combination in which each positive component contains at least one multiplication operation, the partial distance search (PDS) technique [22] is a simple and effective method to find the winner fast. The PDS method can remove many undesired codevectors with fewer multiplications without sacrificing performance.

The efficiency of the PDS method depends on several factors, such as the vector dimension, the order in which components are computed, and how early a codevector with a small enough distortion is found. In general, the higher the dimension of input vectors the more efficient the PDS technique. The earlier the nearest neighbor codevector (or a codevector with small distortion) is found, the greater the reduction in the number of multiplication operations the PDS method may achieve. Since there is a statistical correlation between adjacent input vectors, the codevector winner for the last input vector is considered to be a good initial choice of the winner for the current input vector because this increases the chance of finding a small distortion early in the winner searching process. Pertaining to the computing order of each component in the additive combination, it is obviously beneficial to start the PDS process from the largest component in order to identify the undesired codevectors most quickly. Then the computation time for finding the winning codevector can be significantly reduced. In the weighted partial distortion $pd_i(t-1) \times e^{-m/T}, pd_i(t-1)$ is progressively increased with the cumulation of many Euclidean distances shown in (12). When the attenuation factor $e^{-m/T}$ is not very small, e.g., in the early or intermediate stages of the training process, this weighted partial distortion is larger than the single Euclidean distance $||\boldsymbol{x}(t)-\boldsymbol{y}_i(t-1)||^2$, and of course much larger than each component of the Euclidean distance $(\boldsymbol{x}^j(t)-\boldsymbol{y}_i^j(t-1))^2, j=1,2,\cdots,k$. Therefore, to achieve high computational efficiency, the PDS process is started from the weighted partial distortion that is the largest component in the additive combination of the proposed distortion measurement. By incorporating these considerations, the PDS technique can be applied to the MMPDCL algorithm in a highly efficient manner.

IV. Experimental Results and Comparisons

Experiments on VQ codebook design have been carried out to investigate the proposed MMPDCL algorithm by comparing it with the GLA, splitting GLA, soft competition algorithm [11], and the FSCL algorithm, in terms of performance and computational complexity.

The following three of the commonly used sources, the first-order Gauss–Markov source, digitized speech waveforms and digital images, were applied in codebook design. The first-order Gauss–Markov or Gauss autoregressive source $\{x_n\}$ has the form $x_n = ax_{n-1} + w_n$, where the regression coefficient α has magnitude less than one, and $\{w_n\}$ is a zero mean, unit variance, i.i.d. Gaussian source. The values for α in our experiments were 0, 0.5, and 0.9. Codebooks were designed for this source using a training sequence of 25 600 samples with different dimensions and different sizes by the different algorithms. As a first source of practical importance, we used a training sequence of 320 000 16-b samples of ordinary speech from two male speakers sampled at 11.025 kHz for designing VQ codebooks. Eight 256-grey-level images of the size 256 $\times$ 256 (Lenna, lady, woman, house, camera, Walter, pattern, tree) were decomposed into 4 $\times$ 4 nonoverlapped blocks to form a set of 32 768 16-dimensional vectors for designing codebooks of different sizes using the different algorithms.

A. SNR (PSNR) Performance

The SNR results of the proposed MMPDCL algorithm and the reference algorithms are summarized in Tables I and II for the first-order Gauss–Markov sources and the speech source, respectively. SNR $= 10 \ \log_{10}(\sigma_x^2/\text{MSE})$ where σ_x^2 is the variance of the scalar source sample x. Table III presents the PSNR results on image coding, where PSNR $=$ $10 \ \log_{10}(255^2/\text{MSE})$, MSE being the average mean-squared quantization error per pixel for the eight images. It should be noted that for all the algorithms, we do not specifically deal with any possible empty cells because we mean to test the capability of the algorithms to circumvent the "empty cell" problem automatically. This also contributes to a more comprehensive and fairer comparison purely for the distinctive learning schemes employed by the different algorithms. Actually, there should generally be no empty cells for the algorithms if the initial codebooks are selected directly from the training set, as was made in our experiments. Especially for the FSCL and the proposed MMPDCL algorithms, empty cells will not emerge even with very poor initial codebooks (e.g., many duplications of codevectors in the initial codebook), because the two algorithms aim to achieve uniform codevector usage and uniform partial distortion, respectively.

For the GLA algorithm, we used several different initial codebooks randomly selected from the training vectors. (In our experiments, ten different initial codebooks were used for the Gauss–Markov source, and five different initial codebooks for the speech and image sources.) The highest SNR (or PSNR) achieved was chosen as the final result of the GLA in comparison with a single run of the other algorithms using only one initial codebook. For the soft competition algorithm, it requires time-consuming trial and error to find the appropriate parameters for different sources, e.g., obtaining the precise temperature schedules and specifically treating the cells containing a small number of input vectors, with a view to achieving the excellent results presented in [11].

TABLE I
SNR COMPARISON OF THE DIFFERENT ALGORITHMS FOR GAUSS–MARKOV SOURCES

			SNR (dB)				
α	Vector Dim.	Codebook Size	GLA	Splitting GLA	Soft Comp.	FSCL	MMPDCL
0	8	256	5.984	5.943	6.159	6.026	6.188
	4	256	10.902	10.911	10.984	10.825	11.053
	2	64	15.485	15.484	15.388	15.275	15.499
	2	128	18.405	18.416	18.377	18.156	18.519
	2	256	21.064	21.573	21.115	20.952	21.729
	2	512	23.648	24.856	23.777	23.611	25.164
0.5	8	256	7.009	7.033	7.180	7.047	7.223
	4	256	11.789	11.790	11.862	11.770	11.966
	2	64	16.068	16.092	15.952	15.818	16.092
	2	128	18.957	19.083	18.889	18.711	19.138
	2	256	21.680	22.257	21.744	21.458	22.408
	2	512	24.448	25.520	24.494	24.303	25.785
0.9	8	256	12.048	12.137	12.251	12.087	12.314
	4	256	16.267	16.300	16.305	16.240	16.472
	2	64	19.045	19.136	19.019	18.884	19.152
	2	128	21.884	22.130	21.812	21.526	22.232
	2	256	24.732	25.262	24.479	24.264	25.369
	2	512	27.447	28.568	27.409	27.176	28.763

TABLE II
SNR COMPARISON OF THE DIFFERENT ALGORITHMS FOR SPEECH

		SNR (dB)				
Vector Dim.	Codebook Size	GLA	Splitting GLA	Soft Comp.	FSCL	MMPDCL
8	256	13.066	13.851	13.384	12.876	13.914
	512	13.862	15.476	14.411	13.727	15.739
	1024	14.638	17.358	15.302	14.388	17.781
4	256	17.095	17.867	16.863	17.043	17.979
	512	18.210	20.021	18.158	18.254	20.157
	1024	18.944	22.156	19.286	18.763	22.504
2	64	19.102	19.103	17.954	18.769	19.103
	128	22.035	22.033	20.213	21.656	22.427
	256	24.308	25.003	21.950	23.981	25.729
	512	25.451	28.141	22.877	25.461	28.781
	1024	26.435	31.506	25.359	26.227	31.867

In our experiments, we adopted the temperature schedule[1] $T(m) = T_0\gamma^{-m}$ where $\gamma = 1.05$ and m was the sweep number, without specifically treating the empty cell problem just as mentioned in the last paragraph. Each initial codebook for the soft competition algorithm was randomly selected from each training set with no identical codevectors in each initial codebook. In the FSCL algorithm, we adopted $f(u_i) = u_i^{\exp(-m/T)}$ and $T = M/2$ as recommended in [9], where M was the specified total number of iterations (in sweep) for training (in our experiments, M is 400). In the proposed MMPDCL algorithm, the only undetermined parameter T in the attenuation factor $e^{-m/T}$ was chosen as $T = M/L$, where $L = 10$ for all the sources in our experiments. The FSCL and the MMPDCL algorithms used the same initial codebooks selected randomly from each training set without any restriction. From the results in Tables I–III, it can be clearly seen that the proposed MMPDCL algorithm achieves the best results over a wide range of sources, vector dimensions and codebook sizes. As the designed codebook size increases, the improvements become more significant.

[1]We would like to stress that better results for the soft competition algorithm may be obtained using other schedules (parameters).

Fig. 1 depicts the distributions of the partial distortions (in descending order) obtained from the different algorithms for the Gauss–Markov source with $\alpha = 0$, vector dimension 2, and codebook size 512. The MMPDCL algorithm renders the most even partial distortion distribution with the least maximum partial distortion. From the figure and the corresponding SNR results, it can also be seen that the more even the partial distortion distribution is, the higher SNR value the codebook achieves. The gains of the MMPDCL algorithm over the GLA, FSCL and soft competition algorithms are more than 1.5 dB, while the gain over the splitting GLA is more than 0.3 dB. Fig. 2 demonstrates the corresponding convergence curves of the algorithms for this case, where the best result from ten

TABLE III
PSNR COMPARISON OF THE DIFFERENT ALGORITHMS FOR IMAGE

	PSNR (dB)				
Codebook Size	GLA	Splitting GLA	Soft Comp.	FSCL	MMPDCL
64	27.149	27.134	27.135	27.039	27.190
128	28.235	28.244	28.284	28.034	28.305
256	29.206	29.266	29.345	29.008	29.446
512	30.058	30.355	30.376	29.795	30.542
1024	30.867	31.494	31.389	30.695	31.821
2048	31.768	32.939	32.477	31.622	33.483

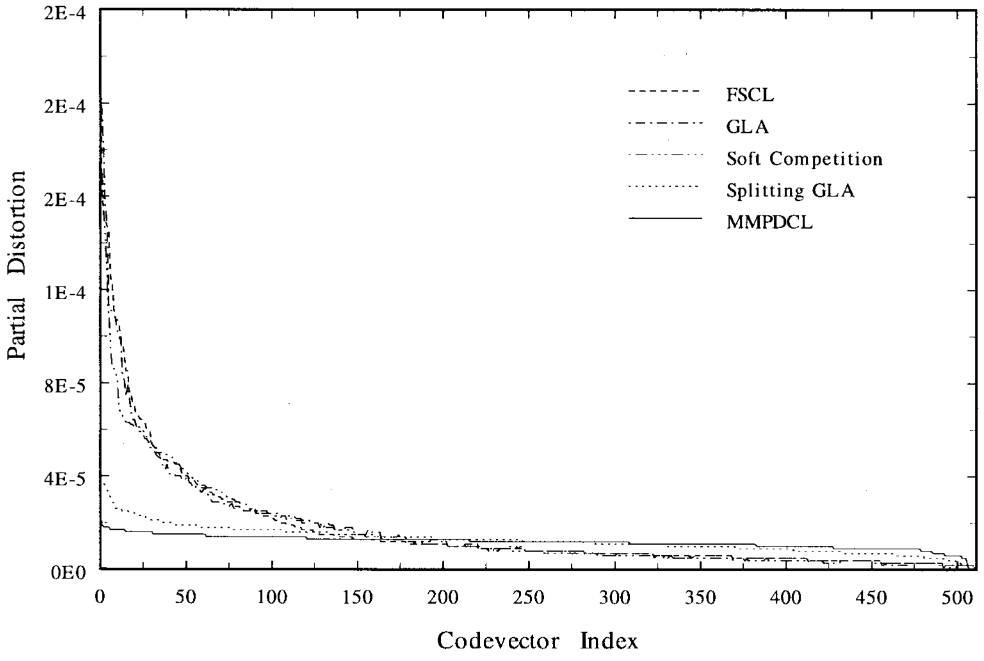

Fig. 1. Distributions of the partial distortions (sorted in descending order) obtained from the different algorithms, for the Gauss i.i.d. source (i.e., $\alpha = 0$ in the Gauss–Markov source) with vector dimension 2 and codebook size 512.

initial conditions is plotted for the GLA. The convergence curve for the splitting GLA is plotted after a good initial codebook has been obtained, so we can see the splitting GLA finds the final codebook very fast using the preprocessed initial codebook. The MMPDCL algorithm obtains a good result at the beginning and surpasses the splitting GLA algorithm at around 60 sweeps of iteration.

B. Robustness Performance

In the MMPDCL algorithm, there is only one undetermined parameter T, defined as $T = M/L$. To test the effect of the parameter T on performance, we equivalently consider the parameter L for a fixed M (= 400 in our experiment). Table IV presents the SNR values using the different values of L for the Gauss–Markov source with $\alpha = 0$, vector dimension 2, and codebook size 512. It can be seen that the performance of the MMPDCL algorithm is robust with respect to variation of the parameter L in the range of zero to 200. The corresponding mean SNR value is 25.052 with the variance 0.009 67. The performance is best for L around 10–20. Comparing Table IV with the relevant results in Table I, we can see the MMPDCL algorithm outperforms the other algorithms even for $L = 0$ or $L = 200$. For $L = 0$ corresponding to the attenuation factor $e^{-m/T} = 0$, the learning process is performed with (8). As L approaches infinity corresponding to $e^{-m/T}$ approaching zero, the learning algorithm tends to become the basic CL algorithm. In our experiment, the basic CL algorithm produces the SNR value of 23.890 for this source, which is more than 1 dB smaller than the SNR values obtained by the MMPDCL for L in the range of [0, 200].

Performance evaluation was also made on the sensitivity to initial codebooks for these algorithms. We used the Gauss–Markov source with $\alpha = 0.5$, vector dimension 2, and codebook size 512 for the test. Table V shows the MSE results for the algorithms using ten different initial codebooks randomly selected from the training set. As can be seen in Table V, the MMPDCL algorithm achieves stable results with similar SNR values for all initial codebooks. Therefore, we can expect that the MMPDCL algorithm is a robust method for codebook design.

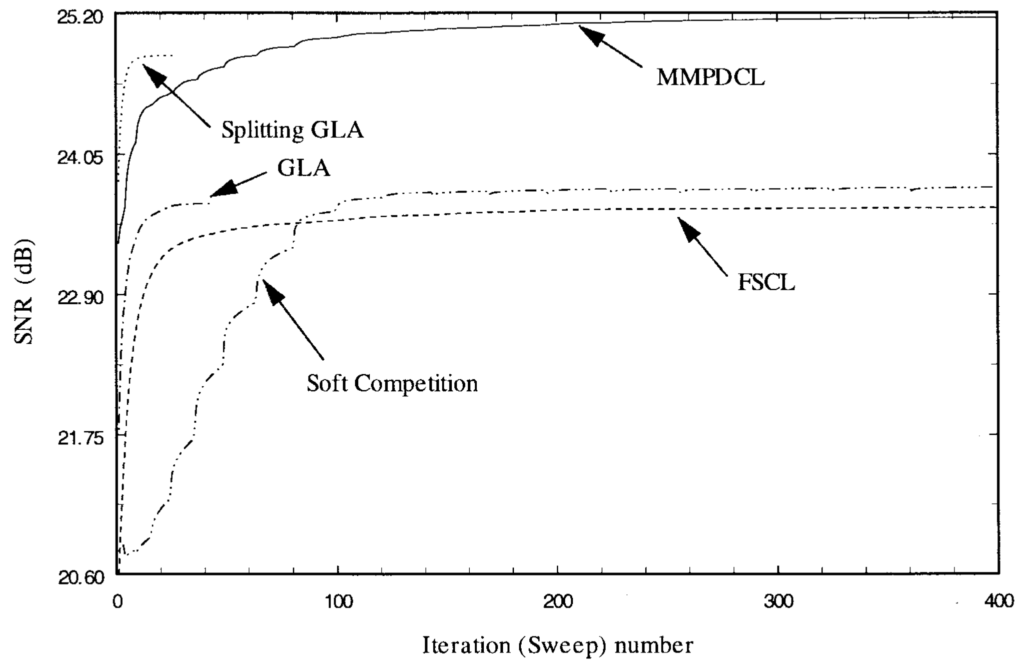

Fig. 2. Convergence of the GLA, splitting GLA, soft competition, FSCL, and MMPDCL algorithms for the Gauss i.i.d. source with vector dimension 2 and codebook size 512.

TABLE IV
SNR RESULTS USING DIFFERENT L FOR THE GAUSS–MARKOV SOURCE WITH $\alpha = 0$, CORRESPONDING TO AN SNR MEAN OF 25.052 AND VARIANCE OF 0.009 67. VECTOR DIMENSION IS 2, AND CODEBOOK SIZE IS 512

L	0	1	5	10	15	20	30	50	100	150	200
SNR	24.880	24.959	25.112	25.164	25.158	25.142	25.136	25.111	25.000	24.965	24.943

TABLE V
MSE RESULTS OBTAINED FROM THE GLA, SOFT COMPETITION, FSCL, AND MMPDCL ALGORITHMS USING TEN DIFFERENT INITIAL CODEBOOKS FOR THE GAUSS–MARKOV SOURCE WITH $\alpha = 0.5$. THE SUPERSCRIPTS "$*$" AND "#" DENOTE THE LARGEST AND SMALLEST MSE VALUES FOR EACH ALGORITHM, RESPECTIVELY. VECTOR DIMENSION IS 2, AND CODEBOOK SIZE IS 512

Initial Codebook	*MSE* $(\times 10^{-3})$			
Index	GLA	Soft Comp.	FSCL	MMPDCL
1	4.727 #	4.677	4.900	3.475
2	4.954	4.621 #	4.822	3.485 *
3	5.102 *	4.850	5.146 *	3.471
4	4.758	4.878	4.672 #	3.477
5	4.796	4.683	4.926	3.464
6	4.986	5.025 *	5.128	3.482
7	4.972	4.744	4.789	3.464
8	5.055	4.755	4.979	3.458 #
9	4.974	4.798	5.068	3.467
10	5.033	4.826	4.887	3.474
Average $(\times 10^{-3})$	4.9357	4.7857	4.9317	3.4717
Variance $(\times 10^{-9})$	15.165	12.386	20.815	0.066

C. Computational Efficiency

To make a fair comparison, we also applied the same partial distance search (PDS) scheme to the GLA and the splitting GLA for winner searching. For the FSCL algorithm [9] employing the distortion metric $f(u_i)\|\boldsymbol{y}_i - \boldsymbol{x}(t)\|^2$, the PDS method can be performed for $dmin/f(u)$, where $dmin$ denotes the current minimum distance. For the soft competition algorithm, all the Euclidean distances between a training vector and each codevector are required for the winning probability calculation [11], so the PDS method is not applicable in this case. Furthermore, the exponential operation for the probability calculation is time consuming. For the MMPDCL algorithm, we used the PDS computation acceleration scheme described in Section III-E. All the simulations were written in the C programming language and were run on a Sun Ultra 1 Model 140 workstation with the "gcc" compilation flags

TABLE VI
AVERAGE COMPUTATION TIME PER SWEEP OF ITERATION FOR THE DIFFERENT ALGORITHMS ACCELERATED BY THE PDS TECHNIQUE (FOR THE IMAGE CODEBOOK GENERATION)

Codebook	Average computation time per sweep of iteration (Seconds)				
Size	GLA	Splitting GLA	Soft Comp.	FSCL	MMPDCL
64	0.952	2.533	16.136	1.945	1.052
128	1.665	6.212	33.356	3.038	1.700
256	3.508	14.499	69.664	5.446	3.245
512	7.601	32.495	152.290	9.686	5.794
1024	15.767	74.038	285.212	17.955	11.529
2048	32.587	135.294	630.79	35.432	24.035

TABLE VII
AVERAGE PERCENTAGE SAVINGS IN TERMS OF THE 16-DIMENSIONAL EUCLIDEAN DISTANCE COMPUTATIONS IN THE WINNER-FINDING PROCESS (FOR THE IMAGE CODEBOOK GENERATION), BY THE ACCELERATED FSCL AND ACCELERATED MMPDCL ALGORITHMS, RESPECTIVELY

Average percentage savings of the Euclidean distance computations (%)						
Codebook Size	64	128	256	512	1024	2048
FSCL	71.55	73.14	74.41	75.48	76.28	76.97
MMPDCL	83.12	85.70	87.73	89.56	90.47	91.19

"-O5 -msupersparc." The average computation time per sweep of iteration by each algorithm is shown in Table VI for the image codebook generation. The average computation time per sweep means the ratio of the total time to the total number of sweeps used for each algorithm. The splitting GLA progresses level by level to obtain a good initial codebook of the designed size, which requires much time for the preprocessing. The average computation time per sweep for the splitting GLA is the ratio of the total time including the preprocessing to the number of sweeps required for the last level codebook of the designed size to converge.

From Table VI, we can see that the MMPDCL algorithm has the least computation time per sweep when the codebook size is larger than 128, up to 26 times faster than that of the soft competition algorithm. In our experiments, one run of the GLA generally took about 30–70 sweeps of iteration to converge, while the three competitive learning algorithms of the soft competition, FSCL and MMPDCL used 400 sweeps of iteration for training. Hence, one run of the GLA requires the least amount of time, and the time needed for the splitting GLA is the next shortest. Among the three competitive learning algorithms, the MMPDCL uses the least total time, only about 60% of that for the FSCL and 5% or less of that for the soft competition algorithm. Moreover, we found in the experiments that the MMPDCL algorithm achieved better results than all the other algorithms in fewer than 100 sweeps of iteration for large codebook sizes ($N \geq 512$), and obtained the near peak values around 100–200 sweeps. In this sense, the MMPDCL algorithm obtains better results than the splitting GLA with less time.

The most computationally intensive part in the training process for the algorithms is the Euclidean distance computations in the winner-finding process for each presentation of an input. With the PDS computation acceleration scheme, the MMPDCL algorithm can be accelerated substantially by a large reduction in the Euclidean distance computations in Step 2) of the algorithm. Table VII presents the average percentage savings of the 16-dimensional Euclidean distance computations by the accelerated FSCL and accelerated MMPDCL algorithms, respectively (for the image codebook generation). We can see that the accelerated MMPDCL algorithm can achieve a larger reduction in the number of Euclidean distance computations, which accounts for its faster running.

V. CONCLUDING REMARKS

A novel minimax criterion with respect to partial distortion has been introduced which is asymptotically necessary as well as asymptotically sufficient for minimizing the average distortion. By incorporating this alternative criterion into on-line competitive learning, the minimax partial distortion competitive learning (MMPDCL) algorithm with a computation acceleration scheme has been developed.

Experimental results have clearly shown that the proposed MMPDCL algorithm consistently produces the best codebooks with the minimum MSE. As the codebook size increases, the performance improvements of the MMPDCL algorithm over the other algorithms become more significant. Furthermore, the MMPDCL algorithm has demonstrated its robustness owing to its insensitivity to the value taken by the sole parameter L, and the choice of initial codebook. It has also been shown that the accelerated MMPDCL algorithm is the most computationally efficient among the three competitive learning algorithms. Therefore, it can be argued that the on-line MMPDCL algorithm is a most promising and practical method for the optimal codebook design, especially for the design of large size codebooks.

Recently, we found that two different algorithms were also developed for vector quantization based on Gersho's asymptotic result of equal subdistortion [18] (without being referred to as the name of partial distortion theorem that is firstly stated in [19]). One is called the *competitive and selective learning* (CSL) algorithm developed by Ueda and Nakano [23] for designing optimal vector quantizers, which employs the basic CL algorithm with a heuristic selection mechanism added that adjusts the number of representative codevectors in regions according to their subdistortions. The other is the optimal k-means algorithm with dynamic adjustment of learning rate developed by Chinrungrueng and Sequin [24], which attempts to minimize the within-region variation weighted MSE. However, in terms of optimization criterion/scheme and implementation, the proposed MMPDCL algorithm, which is based on the novel minimax partial distortion criterion, is very different to these two algorithms. A thorough comparison of our MMPDCL algorithm with the two algorithms will be investigated in the next step.

As a final remark, we would like to point out that there seem to be some interesting connections between filter design methods and codebook design techniques. Some of the algorithms in filter design may be introduced to codebook design. For example, the *Remez exchange algorithm* [21] may be adapted to codebook design by finding codevectors corresponding to (approximately) uniform and minimal partial distortion.

ACKNOWLEDGMENT

The authors gratefully acknowledge the constructive comments, valuable suggestions, and meticulous proofreading of the anonymous reviewers that led to great improvements in the technical and written presentation of this paper, and an anonymous reviewer who streamlined and reworded precisely the presentation of Corollary 1 and its proof (in Subsection III-A) for our final version. The authors also express sincere gratitude to Prof. K. Zeger for helpful discussions and suggestions, and to Dr. D. P. Webb and C. Thompson for carefully proofreading and correcting grammatical and editing aspects of this work. The first author thanks Dr. L. Li and J. Wang for discussions and help with the early work in 1992 to 1993.

REFERENCES

[1] R. M. Gray, "Vector quantization," *IEEE Acoust., Speech, Signal Processing Mag.*, vol. 1, pp. 4–29, Apr. 1984.

[2] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding," *Proc. IEEE*, vol. 73, pp. 1551–1588, Nov. 1985.

[3] N. M. Nasrabadi and R. A. King, "Image coding using vector quantization: A review," *IEEE Trans. Commun.*, vol. 36, pp. 957–971, 1988.

[4] N. Akrout, R. Prost, and R. Goutte, "Image compression by vector quantization: a review focused on codebook generation," *Image Vis. Comput.*, vol. 12, pp. 627–637, 1994.

[5] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COMM-28, pp. 84–95, Jan. 1980.

[6] K. Zeger, J. Vaisey, and A. Gersho, "Globally optimal vector quantizer design by stochastic relaxation," *IEEE Trans. Signal Processing*, vol. 40, pp. 310–322, 1992.

[7] D. Desieno, "Adding a conscience to competitive learning," in *Proc. IEEE Int. Conf. Neural Networks*, San Diego, CA, 1988, pp. 117–124.

[8] N. M. Nasrabadi and Y. Feng, "Vector quantization of images based upon the Kohonen self-organizing feature maps," in *Proc. IEEE Int. Conf. Neural Networks*, San Diego, CA, 1988, pp. 101–108.

[9] A. K. Krishnamurthy, S. C. Ahalt, D. E. Melton, and P. Chen, "Neural networks for vector quantization of speech and images," *IEEE J. Select. Areas Commun.*, vol. 8, pp. 1449–1457, Oct. 1990.

[10] S. C. Ahalt, A. K. Krishnamurthy, P. Chen, and D. E. Melton, "Competitive learning algorithms for vector quantization," *Neural Networks*, vol. 3, pp. 277–290, 1990.

[11] E. Yair, K. Zeger, and A. Gersho, "Competitive learning and soft competition for vector quantizer design," *IEEE Trans. Signal Processing*, vol. 40, pp. 294–309, 1992.

[12] T. M. Martinetz, S. G. Berkovich, and K. J. Schulten, " 'Neural-gas' network for vector quantization and its application to time-series prediction," *IEEE Trans. Neural Networks*, vol. 4, pp. 558–569, July 1993.

[13] F. L. Chung and T. Lee, "Fuzzy competitive learning algorithm with decreasing fuzziness," *Electron. Lett.*, vol. 29, pp. 1206–1208, 1993.

[14] C. Zhu, L. Li, T. Wang, and Z. He, "Partial-distortion-weighted fuzzy competitive learning algorithm for vector quantization," *Electron. Lett.*, vol. 30, pp. 505–506, Mar. 1994.

[15] C. Zhu, J. Wang, L. Li, and Z. He, "A new neural network-based algorithm for vector quantization," in *Proc. Int. Conf. Signal Processing Applications and Technology*, Santa Clara, CA, 1993, pp. 1072–1076.

[16] C. Zhu, L. Li, Z. He, and J. Wang, "A new competitive learning algorithm for vector quantization," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'94)*, Adelaide, Australia, Apr. 1994, pp. 557–560.

[17] T. Kohonen, *Self-Organization and Associate Memory*, 2nd ed. Berlin, Germany: Springer-Verlag, 1988.

[18] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inform. Theory*, vol. 25, pp. 373–380, July 1979.

[19] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression.* Boston, MA: Kluwer, 1992.

[20] B. Ramamurthi and A. Gersho, "Classified vector quantization of images," *IEEE Trans. Commun.*, vol. COMM-34, pp. 1105–1115, Nov. 1986.

[21] A. Antoniou, *Digital Filters: Analysis, Design, and Applications*, 2nd ed. New York: McGraw-Hil, 1993.

[22] C. D. Bei and R. M. Gray, "An improvement of the minimum distortion encoding algorithm for vector quantization," *IEEE Trans. Commun.*, vol. COMM-33, pp. 1132–1133, Oct. 1985.

[23] N. Ueda and R. Nakano, "A new competitive learning approach based on an equidistortion principle for designing optimal vector quantizers," *Neural Networks*, vol. 7, no. 8, pp. 1211–1227, 1994.

[24] C. Chinrungrueng and C. H. Sequin, "Optimal adaptive k-means algorithm with dynamic adjustment of learning rate," *IEEE Trans. Neural Networks*, vol. 6, pp. 157–169, Jan. 1995.

Ce Zhu was born in Sichuan Province, China, in 1969. He received the B.S. degree from Sichuan University, Chengdu, China, in 1989, and the M.S. and Ph.D. degrees from Southeast University, Nanjing, China, in 1992 and 1994, respectively, all in electronic engineering.

In July 1994, he joined Shantou University, Shantou, China. From April to September 1995, he was a Research Associate at the Chinese University of Hong Kong. From May 1996 to May 1997, he was with the Department of Electronic Engineering at the City University of Hong Kong, first as a Research Associate and then a Research Fellow. He was with the Department of Electrical and Electronic Engineering, University of Melbourne, Australia, as a Visiting Research Fellow from May 1997 to February 1998. He is now with Southwest China Normal University, Chongqing, China, and will be joining Nanyang Technological University, Singapore. His research interests include vector quantization, image processing/coding, and neural networks.

Lai-Man Po (S'88-M'91) received the B.Sc. and Ph.D. degrees from City University of Hong Kong, both in electronic engineering, in 1988 and 1991, respectively.

Since November 1991, he has been with the Department of Electronic Engineering, City University of Hong Kong, where he is currently Director of the Image Processing Laboratory and Assistant Professor of the Electronic Engineering Department. His research interests are in vector quantization and image and video compression.

Dr. Po was awarded First Prize (1988) in the Paper Contest for Students and Noncorporate Members organized by the Institute of Electronics and Radio Engineers of Hong Kong, and a Postgraduate Fellowship of the Sir Edward Youde Memorial Council (1988 to 1991).