Multiplication Free Vector Quantization Using L_1 Distortion Measure and Its Variants

V. John Mathews, *Senior Member, IEEE*

Abstract—**Vector quantization is a very powerful technique for data compression and consequently, it has attracted a lot of attention lately. One major drawback associated with this approach is its extreme computational complexity. This paper first considers vector quantization that uses the L_1 distortion measure for its implementation. The L_1 distortion measure is very attractive from an implementational point of view, since no multiplication is required for computing the distortion measure. Unfortunately, the traditional Linde–Buzo–Gray (LBG) method for designing the code book for the L_1 distortion measure involves several computations of medians of very large arrays and can become very complex. We propose a gradient-based approach for codebook design that does not require any multiplications or median computations. Convergence of this method is proved rigorously under very mild conditions. Simulation examples comparing the performance of this technique with the LBG algorithm show that the gradient-based method, in spite of its simplicity, produces codebooks with average distortions that are comparable to the LBG algorithm. The codebook design algorithm is then extended to a distortion measure that has piecewise-linear characteristics. Once again, by appropriate selection of the parameters of the distortion measure, the encoding as well as the codebook design can be implemented with zero multiplications. Finally, we apply our techniques in predictive vector quantization of images and demonstrate the viability of multiplication free predictive vector quantization of image data.**

I. Introduction

The advantages associated with representing, transmitting, and storing information in digital form are well known. Perhaps the most serious disadvantage associated with the conversion of information from analog to digital form is the substantial increase in storage and/or transmission bandwidth requirements. This disadvantage can be mitigated by operating on the data with an appropriate data compression algorithm prior to transmission or storage. Vector quantization is particularly effective, but computationally intensive, class of algorithms that has attracted a lot of interest in recent years. These algorithms take their name from the fact that they operate on entire k-dimensional vectors of waveform samples at once, rather than one sample at a time. They can be applied to a variety of waveforms such as speech [8], images [11], and modem signals [14], and they promise to be of great value in a variety of applications involving such waveforms.

The simplest form of a vector quantizer operates as follows. First, a codebook of k-dimensional vectors is created using a training set representative of the waveforms. Once the codebook is designed and the representative vectors are stored, the process of encoding the incoming waveform can begin. k consecutive samples of the waveform are grouped together to form a k-dimensional vector at the input of the vector quantizer. The input vector is successively compared with each of the stored vectors and a metric or distance is computed in each case. The representative vector closest to the input vector is identified, and the index of the closest vector is available at the output for transmission or storage.

Clearly, the full search vector quantizer described above can become computationally very intensive, depending on the distortion measure employed. For example, if the Euclidean distance measure, given by

$$d_2(X, Y) = \sum_{i=1}^{k} |x_i - y_i|^2. \tag{1}$$

(x_i and y_i are the ith elements of the k-dimensional vectors X and Y, respectively) is employed, and the transmission rate of the compressed data is b b/sample, vector quantization of the incoming waveform requires 2^{kb} multiplications/sample. One very easy way of reducing the complexity is to use the L_1 distortion measure where the distance between two vectors X and Y is computed as

$$d_1(X, Y) = \sum_{i=1}^{k} |x_i - y_i|. \tag{2}$$

It is evident from the above equation that vector quantization using the L_1 distortion measure requires zero multiplication for its implementation. However, the L_1 distortion measure may not be appropriate in all applications. For such cases, we will consider a piecewise-linear distortion measure given by

$$d_l(X, Y) = \sum_{i=1}^{k} f(x_i - y_i) \tag{3}$$

where

$$f(e) = \alpha_j |e| + \gamma_j \tag{4}$$

is a piecewise-linear function and α_j and γ_j are functions

Manuscript received May 18, 1989; revised January 9, 1991.

The author is with the Department of Electrical Engineering, University of Utah, Salt Lake City, UT 84112.

IEEE Log Number 9104329.

of $|e|$ and can take values from a finite set depending on which of the preselected intervals contain $|e|$. In practice, α_i and γ_j are chosen such that f is convex and continuous. Fig. 1 shows a typical f versus e curve. Note that if the slope values α_j's are selected to be integer powers of 2, the vector quantizers that make use of the distortion measure in (3) can be implemented without any multipliers. Furthermore, by appropriate choice of the thresholds τ_j's Fig. 1, we may be able to approximate several other distortion measures using the piecewise-linear distortion measure, and therefore, make the vector quantizer perform similar to those that use other distortion measures and at the same time implement it with highly reduced computational complexity. In a large number of applications, such approximations will result in quantized waveforms with subjective quality that is comparable to that of the waveforms quantized using the "exact" distortion measure even when the approximations are relatively crude.

This paper deals with multiplication free vector quantization using the L_1 and piecewise-linear distortion measures. In spite of the availability of several digital signal processing chips that can implement additions and multiplications in approximately the same time, reduction or elimination of the number of multiplications required to implement algorithms is important because: 1) there exists a large number of digital computers that require larger execution times to implement multiplications than additions, and 2) even in those processors that can execute multiplications and additions in comparable times, multipliers take up much larger chip areas than adders. It must be pointed out that there are several schemes such as tree-search vector quantizers [1] and lattice vectors quantizers [13] that are available in the literature that considerably reduce the computational complexity of vector quantizer systems. Our approach, if employed in conjunction with the above schemes, will reduce the complexity of such methods even further. However, the impact of our scheme is most dramatic on full search vector quantizers. The computational simplifications discussed in the paper are mostly useful in hardware realizations of vector quantizers and can help to simplify the design of several hardware and VLSI implementations of vector quantizers that have appeared in the literature [4], [12]. This paper first considers the design of a codebook when the L_1 distortion measure is used. The traditional Linde–Buzo–Gray (LBG) algorithm [7] when used with the L_1 distortion measure requires computation of the median values of several large sequences and can become computationally very complex. A gradient-based approach that does not require any multiplications or median computations for its implementation is presented in the next section. The convergence properties of the algorithm will be established under very mild conditions. This method is then extended to vector quantizers employing the piecewise-linear distortion measure. Finally, the technique is applied to predictive vector quantization of images. The implementation of the predictive vector quantizer is made multiplication free by selecting the coefficients of the predictor to be (possibly negative) integer powers of two.

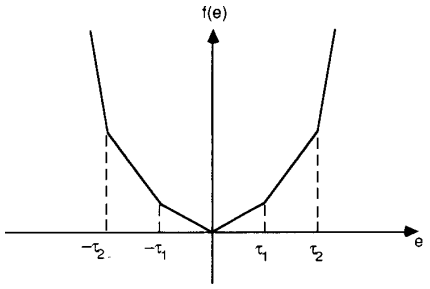


Fig. 1. An example of a function that describes the piecewise-linear distortion measure.

II. Codebook Design for the L_1 Distortion Measure

The traditional approach to codebook design for vector quantizers is a clustering method known as the LBG algorithm [7]. Given a representative training sequence and an initial codebook, the LBG method consists of encoding the training sequence and then replacing each code vector by the centroid of all the training vectors that were mapped into the code vector. This is repeated until convergence is achieved. For the L_1 distortion measure, the centroid is the vector whose elements are the medians of the corresponding elements of all the training vectors that were mapped into it. Calculation of the median of a large set of numbers can become very complicated and time consuming. This section presents a gradient-based algorithm that works very well in spite of the fact that it does not require any multiplications or median calculations.

A. Gradient Algorithm for Codebook Design

Let $\{Y_1, Y_2, \cdots, Y_p\}$ denote the training sequence and $C(0) = \{C_{0,1}, C_{0,2}, \cdots, C_{0,M}\}$ denote the initial codebook. At the mth iteration, encode Y_m using codebook $C(m-1)$. Let $\beta_m = Y_m - C_{m-1,L}$ denote the quantization error vector where $C_{m-1,L}$ is the code vector closest to Y_m in $C(m-1)$. The new codebook $C(m)$ is obtained by replacing $C_{m-1,L}$ in $C(m-1)$ with

$$C_{m,L} = C_{m-1,L} - \mu \frac{\partial}{\partial C_{m-1,L}} d_1(Y_m, C_{m-1,L}) \quad (5)$$

$$= C_{m-1,L} + \mu \operatorname{sign}\{\beta_m\} \quad (6)$$

where sign $\{\cdot\}$ is a vector consisting of the signum function of the corresponding elements of $\{\cdot\}$, and μ is a small convergence constant.

Note that the implementation does not require multiplications or median calculations.

B. Proof of Convergence

Note that the algorithm in (6) is very similar to the sign algorithm [5], [9] employed in adaptive filtering. In fact,

the convergence analysis here is an adaptation of the one in [5] to our particular situation. The only assumption that will be made is that as the number of training vectors becomes very large, the number of training vectors that are mapped into code vectors corresponding to each index ($i = 1, 2, \cdots, M$) will also become very large. This will eliminate the possibility that only part of the codebook is used during the algorithm and the optimization is effectively done for a smaller sized codebook than what is desired.

Let $CB(m)$ denote a vector of Mk elements formed by concatenating the elements of $C(m)$, i.e.:

$$CB(m) = (C_{m,1}^T, C_{m,2}^T, \cdots, C_{m,M}^T)^T \tag{7}$$

where $(\cdot)^T$ denotes the matrix transpose of $(\cdot)$. Similarly, let C_{opt} denote another vector of Mk elements formed by the elements of an "optimal" codebook that minimizes the total distortion in encoding the training sequence. Note that any codebook that minimizes the total distortion will suffice. Furthermore, the ordering of the M optimal code vectors in C_{opt} can be arbitrary.

As before, let $C_{m-1,L}$ denote the closest vector in $C(m-1)$ to Y_m. Define an Mk vector $G(m)$ as

$$G(m) = (0, 0, \cdots, 0, \beta_m^T, 0, 0, \cdots, 0)^T. \tag{8}$$

That is, $G(m)$ is a vector of zero elements, except for the positions corresponding to the codebook element in $C(m-1)$ that is closest to Y_m.

From (6) and the definitions above:

$$CB(m) = CB(m-1) + \mu \text{ sign } \{G(m)\} \tag{9}$$

where sign $\{0\}$ is taken to be zero. Let

$$V(m) = CB(m) - C_{\text{opt}}. \tag{10}$$

Subtracting C_{opt} from both sides and taking the squared norm of both sides of (9), we obtain

$$\begin{aligned} \|V(m)\|^2 &= \|V(m-1)\|^2 + \mu^2 k \\ &\quad + 2\mu V^T(m-1) \text{ sign } \{G(m)\} \\ &= \|V(m-1)\|^2 + \mu^2 k \\ &\quad + 2\mu\{C_{m-1,L} - C_{\text{opt},L}\} \text{ sign } \{\beta_m\}. \end{aligned} \tag{11}$$

The last term comes from the fact that all other elements of $G(m)$ are zero. Also, $C_{\text{opt},L}$ denotes the Lth k-tuple in $C_{\text{opt}}(m-1)$. Note that

$$\begin{aligned} &C_{m-1,L} - C_{\text{opt},L} \\ &\quad = (Y_m - C_{\text{opt},L}) - (Y_m - C_{m-1,L}) = \gamma_m - \beta_m \end{aligned} \tag{12}$$

where γ_m is the quantization error vector when Y_m is encoded using $C_{\text{opt},L}$. Recognizing that

$$\beta_m^T \text{ sign } \beta_m = d_1(Y_m, C_{m-1,L}) \tag{13}$$

is the distortion when Y_m is encoded using $C(m-1)$ and

$$\gamma_m^T \text{ sign } \{\beta_m\} \le \gamma_m^T \text{ sign } \{\gamma_m\} = d_1(Y_m, C_{\text{opt},L}) \tag{14}$$

and denoting the encoding distortion in (13) by $e(m)$ and the distortion in (14) by $\epsilon(m)$, we can obtain the following result:

$$\|V(m)\|^2 \le \|V(m-1)\|^2 + \mu^2 k - 2\mu e(m) + 2\mu\epsilon(m). \tag{15}$$

Iterating the inequality in (15) m times, we obtain the following inequality:

$$\begin{aligned} \|V(m)\|^2 &\le \|V(0)\|^2 + \mu^2 mk \\ &\quad - 2\mu \sum_{i=1}^{m} e(i) + 2\mu \sum_{i=1}^{m} \epsilon(i). \end{aligned} \tag{16}$$

Since $\|V(m)\|^2 \ge 0$, the above implies that

$$\frac{1}{m}\sum_{i=1}^{m} e(i) \le \frac{\|V(0)\|^2}{2\mu m} + \frac{\mu}{2}k + \frac{1}{m}\sum_{i=1}^{m}\epsilon(i). \tag{17}$$

Taking the limit as m goes to infinity:

$$\limsup_{m\to\infty} \frac{1}{m}\sum_{i=1}^{m} e(i) \le \zeta + \frac{\mu}{2}k \tag{18}$$

where ζ is the average distortion produced by mapping the training sequence into the optimal codebook. The mapping is such that whenever a training vector Y_m is closest to the ith code vector of the codebook $C(m-1)$, it is mapped into the ith k-tuple of the Mk vector C_{opt}. Even though this may not in general be the optimal mapping, the above result shows that the algorithm converges and that there is a one-to-one mapping from the codebook being designed and the optimal codebook such that the long-term average of the distortion during codebook design exceeds that due to the above mapping into the optimal codebook by $(\mu/2)k$ or less. Note that the ordering of the optimal code vectors in C_{opt} was arbitrary. In particular, if we choose the above one-to-one mapping to be the "best" possible one that minimizes the long-term average of the distortion produced by the mapping on the training sequence, the long-term average of the distortion produced by the codebook during the design process will still not exceed the performance of the optimal codebook and the one-to-one mapping by more than $\mu k/2$. Even though there is a possibility that this mapping may be very different from the optimal one, all the experiments that we have done have produced results that are comparable to that of the LBG algorithm.

C. Remarks

1) Note that M subsequences of the training sequences are generated during the codebook design process. Each subsequence trains one element of the codebook. By applying the above analysis to each vector separately, it is very straightforward to show that each code vector sequence converges to the centroid of the subsequence corresponding to that vector.

2) The bound in (18) was developed under the assumption that we have an infinitely long training sequence. In practice, we have only finitely many training vectors on

which we can train the codebook. In such situations, we will iterate the algorithm on the training sequence several times until the desired performance level has been achieved.

During each iteration, the algorithm will attempt to create a codebook that consists of the centroids of the subsequences discussed in 1). Thus the gradient algorithm tries to do the same thing that the LBG algorithm does (without computing the centroids). Therefore, we would expect the overall performance of both the algorithms to be similar.

To develop a stopping criterion for the algorithm, the following definition of average distortion over the whole training sequence during the rth pass was used. (One pass is defined as the process of updating the codebook elements using every training sequence vector once each. During the next pass, the algorithm will start with the codebook obtained from the previous pass and refine it further using all the elements of the training sequence):

$$\text{dist } (r) = \frac{1}{p} \sum_{i=1}^{p} e(i, r). \tag{19}$$

In (19) $e(i, r)$ denotes the distortion in encoding Y_i during the rth pass. At the end of the rth pass, we will stop iterating if

$$\frac{|\text{ dist } (r-1) - \text{dist } (r)|}{\text{dist } (r)} \leq \epsilon \tag{20}$$

where ϵ is a preselected threshold.

3) The foregoing analysis suggests that the algorithm will perform well, regardless of the initial codebook. This author has found that populating the initial codebook by starting with only one code vector and then increasing the codebook size using the (binary) splitting technique [7] does give some improvement in the performance over arbitrary selection of the initial codebook.

D. Experimental Results

The results in Table I present a comparison of the performance of the gradient descent method and the LBG algorithm for codebook design. Both codebook design techniques were implemented in Fortran on a Sun 3/50 workstation. Table I contains the results when the training sequence was a zero mean, pseudorandom Gaussian sequence obtained by passing a zero-mean and white Gaussian sequence with unit variance through a first-order autoregressive filter with transfer function:

$$H(z) = \frac{1}{1 - 0.9\, z^{-1}}. \tag{21}$$

The two algorithms were compared for a vector dimension of four and several codebook sizes. For the gradient method, the convergence parameter was chosen to be 0.004. Both the methods used $\epsilon = 0.001$ for the stopping criterion and populated the codebook beginning with one vector. The training sequence was 20 000 vectors long. The results show that the average distortions produced by the two methods are comparable. The table also contains information about the number of iterations (passes) on the training sequence before each algorithm converged. Note that the gradient method converges faster than the LBG algorithm in most instances. Since the gradient methods do not require median calculations, and it seems to converge faster than the LBG algorithm, the author believes that it is a better approach to codebook design when the L_1 distortion measure is employed.

TABLE I
COMPARISON OF THE PERFORMANCE OF THE GRADIENT AND LBG ALGORITHMS IN VECTOR QUANTIZER CODEBOOK DESIGN FOR L_1 DISTORTION MEASURE

	Gradient Method		LBG Algorithm	
Number of Code Vectors	Average Distortion	Number of Iterations	Average Distortion	Number of Iterations
1	7.34	2	7.34	1
2	4.78	3	4.78	4
4	3.36	3	3.36	6
8	2.69	3	2.69	7
16	2.20	4	2.22	8
32	1.85	6	1.86	13
64	1.58	9	1.59	13
128	1.34	10	1.34	14
256	1.12	15	1.12	13

III. Codebook Design for Piecewise-Linear Distortion Measures

Since in many data compression systems the ultimate objective is to produce quantized signals with a minimum amount of subjective distortion rather than produce quantized signals that minimize certain quantitative distortion measure, we must keep in mind that the L_1 distortion measures may not be the most suitable one for many applications. In order to overcome this limitation, we propose the use of piecewise-linear approximations for distortion measures that give rise to more complex encoding procedures. The functional form of the distortion measure is as given in (3) and (4). As stated earlier, if we choose the slopes α_i's of the piecewise-linear functions to be integer powers of two, the encoding can be done with only bit shifts, additions, and comparisons.

The codebook design algorithm of (6) can be easily extended to this case. We will use the same notations as in the previous section. Let $\beta_m = Y_m - C_L^{m-1}$ be the quantization error vector that is due to the code vector that minimizes the piecewise-linear distortion measure in (3). Let $\alpha_i(m)$ correspond to the slope of the piecewise-linear function in (4) that corresponds to the ith element of β_m. Define a diagonal matrix $\Lambda(m)$ as

$$\Lambda(m) = \text{diag } \{\alpha_1(m), \alpha_2(m), \cdots, \alpha_k(m)\}. \tag{22}$$

Then the update equation for the gradient descent codebook design algorithm becomes

$$C_L^m = C_L^{m-1} + \mu\Lambda(m) \text{ sign } \{\beta_m\}. \tag{23}$$

Again, note that the codebook design algorithm can be implemented using zero multiplications.

In the next section we will apply the above method to predictive vector quantization of digital images.

IV. Application to Predictive Vector Quantization of Images

Researchers have found that predictive vector quantization algorithms outperform direct vector quantization much as linear predictive coding algorithms work better than PCM in scalar quantization schemes [3], [10], [15]. There are two main reasons why predictive vector quantization algorithms are attractive: first, the prediction error sequence will in general have a smaller dynamic range than the original input waveform, and compression of the prediction error sequence can, in most cases, be done more effectively than the input waveform itself. The second reason addresses a problem that is typical of all vector quantization algorithms. Since the codebook is designed for a training sequence that is representative of the input waveforms to the quantizer, any variations in the structure of the input waveforms from that of the training sequence will degrade the performance of the vector quantizer. The prediction error sequences will have far less structure in them than the waveforms themselves, and therefore, a vector quantizer working on prediction error sequences will be more robust to variations in the structure of the input waveforms.

In this section we apply the results of the previous section to predictive vector quantization of digital images. The block diagram of the predictive vector quantizer is shown in Fig. 2. In the figure, $x(n, m)$ corresponds to the input image and B is an estimate of the mean value of the image. The rest of the notation is self-explanatory. The predictor coefficients are all chosen to be (possibly negative) integer powers of two, and therefore, the structure is truly multiplication free.

Note that the prediction filter predicts the current input vector using previous input vectors and the resulting error sequence is vector quantized. The codebook is designed from a set of training images. The predictor coefficients for the images are first estimated. Starting with an initial codebook, the mean-removed training image is quantized, and the codebook is updated each time after an input vector is quantized as described in Section III.

We now present a comparative study of the performances of the predictive vector quantizer when the distortion measures employed are the L_2 and piecewise-linear measures. The original image that is quantized is entitled "woman" and is shown in Fig. 3. It consists of 512 × 512 pixels with 8-b resolution. The predictor equations when the piecewise-linear distortion measure is used are given in Table II. The notational convention employed in Table II is shown in Fig. 4. These equations have been adopted with minor changes from those used in [15]. Even though these coefficients are not necessarily the optimal set of power-of-two predictor coefficients, they seem to work well in our application. The author's experience is that the vector quantizer can compensate for a certain amount of suboptimality in the predictor. Design of predictors with power-of-two coefficients can, in general, be done as in [6]. For the L_2 distortion measure, the predictor

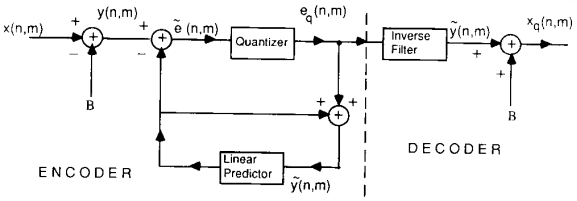


Fig. 2. Block diagram for predictive vector quantization of images.

Fig. 3. Original "woman" image used in the experiments.

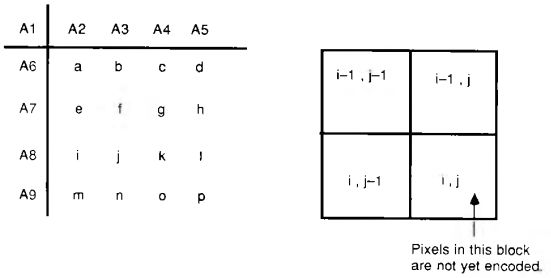


Fig. 4. Block and pixel notations for the predictive vector quantizer. $A1$-$A9$ are decoded pixels in blocks $(i-1, j-1)$, $(i-1, j)$ and $(i, j-1)$. a-p are pixels in block (i, j) that are predicted using $A1$-$A9$. (a) Pixel notation. (b) Block rotation.

TABLE II
Predictor Equations for the Predictive Vector Quantizer. ~ Denotes Predicted Quantities

$\tilde{f} = (A3 + A7)/2$	$\tilde{c} = (\tilde{f} + \tilde{h} + A3 + A5)/4$
$\tilde{b} = (\tilde{f} + A3)/2$	$\tilde{d} = (\tilde{h} + A5)/2$
$\tilde{e} = (\tilde{f} + A7)/2$	$\tilde{m} = (\tilde{n} + A9)/2$
$\tilde{a} = (\tilde{f} + A1 + A3 + A7)/4$	$\tilde{g} = (\tilde{f} + \tilde{h})/2$
$\tilde{p} = \tilde{f} + (A5 + A9)/4 - A1/2$	$\tilde{j} = (\tilde{f} + \tilde{m})/2$
$\tilde{n} = \tilde{f} + (\tilde{p} + A9)/4 - A3/2$	$\tilde{l} = (\tilde{h} + \tilde{p})/2$
$\tilde{i} = (\tilde{f} + \tilde{n} + A7 + A9)/4$	$\tilde{o} = (\tilde{n} + \tilde{p})/2$
$\tilde{h} = \tilde{f} + (\tilde{p} + A5)/4 - A7/2$	$\tilde{k} = (\tilde{f} + \tilde{h} + \tilde{n} + \tilde{p})/4$

coefficients were chosen to be the optimal least squares predictor coefficients for pixels a–p in Fig. 4 using the same set of pixels as in Table II. A gradient algorithm similar to that in [2] was used for designing the codebook for the L_2 distortion measure. Before being processed by the prediction filter, the mean value of the image was removed from the original image. The results presented were obtained using 4×4 vectors and 512 code vectors resulting in an effective data rate of 0.5625 b/pixel. Note that the purpose of this experiment is to compare the performances when the two distortion measures are used and we chose to use the "woman" image itself as the training sequence. The choice of various parameters for the experiments, including the piecewise-linear function that was employed is given in Table III. Fig. 5(a) and 5(b) displays the encoded image when the L_2 norm and the piecewise-linear measure, respectively, were used as the distortion measures. The average piecewise-linear distortion between Fig. 5(b) and the original image is 37.44. The average squared distortion between these two images is 34.60, which corresponds to a signal-to-quantization noise (SQR) ratio of 32.74 dB. The SQR is defined as

$$\text{SQR} = -10 \log \frac{\text{average squared distortion}}{(\text{peak-to-peak signal value})^2}. \quad (24)$$

The average squared distortion of the image created using the L_2 distortion measure is 35.60 (32.62 dB) and the corresponding piecewise-linear distortion is 38.54. We can see from the above numbers as well as the figures that the performance of the predictive vector quantizer when the piecewise-linear distortion measure is employed is very comparable to that when the L_2 distortion measure is used. (The slightly larger value of the squared distortion of Fig. 5(a) over that of Fig. 5(b) can be attributed in part to the slight suboptimality of the codebook design algorithms.) It would be appropriate at this time to compare the computational complexity of the two methods. The differences in complexity of the algorithms occur in the predictor part and the computation of the distortion function. The prediction problem for the piecewise-linear distortion measure requires 61 additions and 19 bit-shift operations per vector in our example. The vector quantizer using the L_2 distortion measure did not make use of power-of-two coefficients, and therefore, required 46 multiplications and 61 additions per vector to implement the predictor. Direct computation of the distortion measures requires at most 3 M (M is the number of codevectors and k is the number of samples per vector) comparisons, kM bit-shift operations, and $(2k - 1)$ M additions per vector for the piecewise-linear distortion measure, and kM squaring operations and $(2k - 1)$ M additions for the L_2 distortion measure. Assuming that it is better to replace each squaring operation by one bit-shift operation and at most 3 comparisons, the use of the piecewise-linear distortion measure is very attractive and useful in predictive vector quantization of images. This is particularly so since the use of the two distortion measures considered resulted in very comparable quantized images.

(a)

(b)

Fig. 5. Encoded "woman" image. (a) Result with the L_2 norm is employed (b) Results when the piecewise-linear norm is employed.

TABLE III
PARAMETERS OF THE PREDICTIVE VECTOR QUANTIZER IN THE EXPERIMENT

Range of $\lvert e \rvert$	Slope	Intercept
0.0 – 1.0	1.0	0.0
1.0 – 3.0	4.0	−3.0
3.0 – 5.0	8.0	−15.0
5.0 – 11.0	16.0	−55.0
11.0 – 21.0	32.0	−231.0
21.0 – ∞	64.0	−903.0

V. Concluding Remarks

The usefulness and effectiveness of vector quantizers in waveform compression is well known. However, their computational complexity has made their implementation a costly proposition. The purpose of this paper was to propose an approach to multiplication free vector quantization, and then demonstrate its usefulness in predictive vector quantization of digital images. Toward this end, this paper first presented an approach to a gradient-based codebook design algorithm for vector quantizers that required no multiplications or median computations. With very mild assumptions, it was shown that the algorithm converges, and the long-term average of the encoding distortion can be made arbitrarily close to that due to a cer-

tain mapping of the training sequence into the optimal codebook. Also, comparisons with the LBG algorithm showed that the gradient-descent algorithm produced codebooks that were comparable to the former. This approach was extended to the case when a piecewise-linear distortion measure is employed. This method was applied to predictive vector quantization of images and comparisons showed that the subjective quality of the encoded images obtained using the L_2 norm and the piecewise-linear distortion measure were similar. The ease of implementation and the quality of encoding associated with the methods suggest that they are very viable candidates in waveform coding applications involving vector quantization.

Acknowledgment

The author would like to acknowledge Mehrdji Khorchidian for help with programming some of the algorithms discussed in this paper.

References

[1] A. Buzo, A. H. Gray, R. M. Gray, and J. D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 562–574, Oct. 1980.

[2] P. C. Chang and R. M. Gray. "Gradient algorithms for designing predictive vector quantizers," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 679–690, Aug. 1986.

[3] V. Cuperman and A. Gersho, "Vector predictive coding of speech at 16 kbits/s," *IEEE Trans. Commun.*, vol. COM-33, pp. 685–696, July 1985.

[4] G. A. Davidson, P. R. Cappello, and A. Gersho, "Systolic architectures for vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 1651–1664, Oct. 1988.

[5] A. Gersho, "Adaptive filtering with binary reinforcement," *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 191–199, Mar. 1984.

[6] Y. C. Lin and S. R. Parker, "FIR filter design over a discrete power-of-two coefficient space," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, pp. 583–590, June 1983.

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

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

[9] V. J. Mathews and S. H. Cho, "Improved convergence analysis of stochastic gradient adaptive filters using the sign algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 450–454, Apr. 1987.

[10] V. J. Mathews, R. W. Waite, and T. D. Tran, "Image compression using vector quantization of linear (one-step) prediction errors," in *Proc. IEEE Int. Con. on Acoustics, Speech, and Signal Processing*, Dallas, Texas, April 6–7, 1987, pp. 733–736.

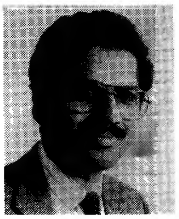
[11] N. M. Nasrabadi and R. A. King, "Image coding using vector quantization: A review," *IEEE Trans. Commun.* vol. 36, pp. 957–971, Aug. 1988.

[12] P. A. Ramamoorthy, B. Potu, and T. Tran, "Bit-serial VLSI implementation of vector quantizer for real-time image coding," *IEEE Trans. Circuits Syst.* vol. 36, pp. 1281–1290, Oct. 1989.

[13] K. Sayood, J. D. Gibson, and M. C. Rost, "An algorithm for uniform vector quantizer design," *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 805–814, Nov. 1984.

[14] T. D. Tran, V. J. Mathews, and C. K. Rushforth, "A baseband residual vector quantization algorithm for voiceband data signal compression," *IEEE Trans. Commun.*, vol. 37, pp. 949–955, Sept. 1989.

[15] B. Vasudev, "Predictive VQ schemes for gray scale image compression," in *Proc. GLOBECOM'87*, Tokyo, Japan, Nov. 1987, pp. 436–441.



V. John Mathews (S'82–M'85–SM'90) was born in Nedungadappally, Kerala, India, in 1958. He received the B.E. (hons.) degree in electronics and communication engineering from the University of Madras, India, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Iowa, Iowa City, in 1980, 1981, and 1984, respectively.

From 1980 to 1984 he held a Teaching Research Fellowship at the University of Iowa, where he also worked as a Visiting Assistant Professor with the Department of Electrical and Computer Engineering from 1984 to 1985. He is currently an Associate Professor with the Department of Electrical Engineering, University of Utah, Salt Lake City. His research interests include adaptive filtering, spectrum estimation, and data compression.

Dr. Mathews is an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING.