Fine–Coarse Vector Quantization

Nader Moayeri, *Member, IEEE*, David L. Neuhoff, *Senior Member, IEEE*, and Wayne E. Stark, *Member, IEEE*

***Abstract*—A fast method for searching an unstructured vector quantization (VQ) codebook is introduced and analyzed. Dubbed fine–coarse vector quantization (FCVQ), it operates in two stages: a "fine" structured VQ followed by a table lookup "coarse" unstructured VQ. Its rate, distortion, arithmetic complexity, and storage are investigated using analytical and experimental means. Optimality conditions and an optimizing algorithm are presented. The results of experiments with both uniform scalar quantization and tree-structured VQ (TSVQ) as the first stage are reported. Comparisons are made with other fast approaches to vector quantization, especially TSVQ. It is found that when rate, distortion, arithmetic complexity, and storage are all taken into account, FCVQ outperforms TSVQ in a number of cases. In comparison to full search quantization, FCVQ has much lower arithmetic complexity, at the expense of a slight increase in distortion and a substantial increase in storage. The increase in mean-squared error (over full search) decays as a negative power of the available storage.**

I. Introduction

VECTOR quantization (VQ) is well known to be a useful source coding scheme in the sense of achieving the rate-distortion function (cf. [1]–[4]). However, its implementation complexity can be excessive. For example, the full search quantization algorithm for an unstructured VQ codebook with dimension k and rate R b/sample stores the codebook's 2^{kR} quantization vectors and finds the best match for a source vector $\boldsymbol{x} = (x_1, \cdots, x_k)$ by calculating the distortion between $\boldsymbol{x}$ and each quantization vector. Thus, both storage and arithmetic complexity increase exponentially with k and R. On the other hand, the distortion achievable with rate R vector quantizers decreases with k, suggesting that k should be chosen as large as possible, or at least at the "knee" of the curve of distortion versus k. As a rule, this knee occurs later for sources with more memory. Hence, it is the interesting sources such as speech and images for which one would most benefit from using large k. With present technology, it appears to be the approximately 2^{kR} distortion calculations per source vector that limits the choice of k, rather than the storage of the 2^{kR} quantization vectors.

In this paper we introduce and analyze a new approach to quantizing with an unstructured codebook, called fine–coarse VQ (FCVQ), that in comparison to full search has substantially less arithmetic complexity at the cost of slightly more distortion and substantially more storage. In other words, it trades an increase in storage for a decrease in arithmetic complexity. The increase in mean-squared error (over full search) decays as $s^{-2/k}$, where s is the available storage.

There are also a number of well-known "structured" vector quantizers having quantization algorithms whose arithmetic complexity or storage requirements are less than full search. These include block transform, tree-structured, lattice, gain/shape, and multistage vector quantizers (cf. [1]). As a rule, these do not give as small distortion as full search of the best unstructured VQ codebook. More specifically, their distortion decreases and their complexity increases as their structure decreases. For example, the fidelity (inverse of distortion) and complexity of lattice quantizers increase both with their dimension and with the complexity of the shapes of their cells and support regions; and the fidelity and complexity of tree-structured VQ increases with dimension as well as with the order of the tree (2^m-ary instead of binary). Such structured vector quantizers are both competitors to and components of the fine–coarse approach, described next.

The basic fine–coarse approach is illustrated in Fig. 1. Given a k-dimensional "coarse" codebook $C = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_N\}$ for which one seeks a low distortion quantization algorithm, one first quantizes the given source vector $\boldsymbol{X} = (X_1, \cdots, X_k)$ using a structured "fine" vector quantizer having k-dimensional codebook $C_f = \{\boldsymbol{y}_{f,1}, \cdots, \boldsymbol{y}_{f,N_f}\}$ with $N_f >> N$ and also having low arithmetic complexity and very small distortion. Next, one quantizes the output $\boldsymbol{Y}_f$ using the minimum distortion quantization rule for C. Letting Q_f and Q_C denote the quantization rules just mentioned, the fine–coarse vector quantizer has overall quantization rule $Q^+(\boldsymbol{x}) = Q_C(Q_f(\boldsymbol{x}))$.

The benefit of this method is that the second stage of quantization can be performed by table lookup, and consequently, does not add arithmetic complexity to that of the "fast" first stage. Specifically, a table could be constructed to store $Q_C(\boldsymbol{y}_f)$ for each fine quantization vector $\boldsymbol{y}_f$ in C_f. However, as illustrated in Fig. 2, to actually do encoding and decoding with the fine–coarse approach, the output of the first stage of encoding is the index $U_f = e_f(\boldsymbol{X})$ of the fine quantization vector $\boldsymbol{Y}_f = Q_f(\boldsymbol{X})$ in the codebook C_f, and this index is used to address the table and retrieve the index $U = T(U_f)$ of the quantization vector $\boldsymbol{Y} = Q_C(Q_f(\boldsymbol{X}))$ in C. That is, the overall encoding rule produces $U = e^+(\boldsymbol{X}) \triangleq T(e_f(\boldsymbol{X}))$. Decoding is also performed by table lookup. In this case the table stores the coarse quantization vectors in C, is addressed by U,

Manuscript received September 28, 1988; revised July 26, 1990.

M. Moayeri is with the Department of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ 08855-0909.

D. L. Neuhoff and W. E. Stark are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2122.

IEEE Log Number 9144723.

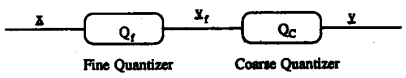


Fig. 1. Fine quantization followed by coarse.

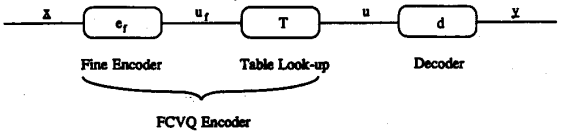


Fig. 2. FCVQ encoding and decoding.

and produces the quantization vector $Y^+ = d(U) \triangleq Q_C(Q_f(X))$ from C.

This fine–coarse approach will be more carefully introduced in Section III in a slightly more general framework. However, at this point it should be apparent that its distortion is larger than that of the minimum distortion quantization rule Q_C by an amount that will be small if the fine vector quantizer has small cells. Moreover, its arithmetic complexity is simply that of the algorithm for the fine encoding rule e_f (presumed to be small), and its storage requirements are those of e_f, the table for T and the decoding table for d.

The purpose of this paper is to introduce the fine–coarse approach and analyze its performance and complexity. We compare it to related approaches, give formulas and bounds for distortion, arithmetic complexity, and storage; provide optimality conditions and an optimization algorithm; and present the results of applying FCVQ to IID Gaussian, Gauss–Markov, and speech sources with both product quantizers and tree-structured vector quantizers as the fine quantizer. Tables at the end list a variety of specific FCVQ configurations that were found to have advantages over benchmark tree-structured VQ's.

Although the value of fine–coarse quantization rests on the cheapness of digital storage, one cannot ignore the cost of storage. There are two reasons. First, the performance of FCVQ improves as the first stage quantizer gets finer, and consequently, as the available storage increases (for T and in the case of some fine quantizer, for e_f). This implies that no matter how inexpensive a byte of storage becomes, the total investment in storage in a well-designed FCVQ system will be significant, for otherwise one could improve performance with an insignificant further investment in storage. Second, having removed arithmetic complexity as an obstacle, one will want to use codebooks with larger dimensions (because distortion decreases with dimension), which necessitates still more storage. For such reasons we will pay close attention to storage.

II. VQ Preliminaries

From an input–output point of view, a vector quantizer (VQ) is characterized by its dimension k, size N, codebook C, and quantizing partition S. The codebook $C = \{y_1, \cdots, y_N\}$ is a subset of $\mathbb{R}^k$ containing N quantization vectors, and the quantizing partition $S = \{S_1, \cdots, S_N\}$ is a partition of $\mathbb{R}^k$ into N quantization cells. A vector quantizer will be denoted (C, S) or C or S, depending on what needs emphasis.

Given a source vector $x \in \mathbb{R}^k$, the encoder determines the quantization cell S_i it lies in, and produces a $\lceil \log N \rceil$ bit binary representation of the index i, (all logarithms in this paper have base 2). In other words, the encoder uses the encoding rule $e(x) = i$ when $x \in S_i$. Upon receiving the binary representation of the index i, the decoder produces the quantization vector y_i as a reproduction of x. That is, it follows the decoding rule $d(i) = y_i$. Notice that the encoding rule is entirely determined by the quantizing partition, and the decoding rule is entirely determined by the codebook. The overall effect of the quantizer is captured by the quantization rule $Q(x) = y_i$ when $x \in S_i$.

The complexity of a vector quantizer, to which we pay close attention, is a property of the algorithms that implement the encoding and decoding rules. In order to make quantitative comparisons among various algorithms, we will count both the number of multiplications required per source sample, which we call the arithmetic complexity, and the number of bytes of memory required, which we call the storage. In particular, the arithmetic complexity and storage of a vector quantizer will be the sums of those of its encoding and decoding algorithms. No measures of complexity are entirely satisfactory, but it is felt that these lead to reasonable comparisons. One might wish, for example, to additionally count additions and binary comparisons. Fortunately, the algorithms we will analyze use them in proportion to multiplications. So a comparison based on multiplications is fairly robust.

The performance of a vector quantizer (C, S) is characterized by its rate and average distortion. The rate, generally denoted R, is defined to be $(\log N)/k$. Actually, $\lceil \log N \rceil / k$ would be more accurate, but we will generally be interested in VQ's where N is either very large or a power of two. The average distortion depends on a selected distortion measure and the statistics of the source to which it is applied. For simplicity, we adopt squared-error distortion, although it will be clear that the fine–coarse approach will work for most common waveform distortion measures. A source will be modeled as a stationary random process $\{X_i\}$. Letting $X = (X_1, \cdots, X_k)$ denote the random vector representing the first k source samples and letting $Y = Q(X)$, the average distortion of (C, S) is $D(C, S) = E\|X - Y\|^2/k$, where $\|\cdot\|$ denotes the Euclidean distance $(\Sigma_{i=1}^k (x_i - y_i)^2)^{1/2}$.

Given a source and a quantizing partition S, let C_S^* denote a codebook yielding least distortion, and let Q_S^* and $D^*(S)$ denote, respectively, the resulting quantization rule and distortion. Similarly, given a source and a codebook C, let S_C^* denote a quantizing partition that yields minimal distortion, and let Q_C^* and $D^*(C)$ denote the resulting quantization rule and distortion. Finally, given a source, let $(C_{k,N}^*, S_{k,N}^*)$ denote a VQ with least distortion among all VQ's with dimension k and size N, and let $D_{k,N}^*$ denote its distortion. The Appendix states several well-known

properties of minimum mean-squared-error quantizers that will be needed in later sections.

Given a codebook C, the minimum distortion quantization rule Q_C^* and corresponding encoding and decoding rules may be implemented by the full search algorithm, described here in more detail for squared-error distortion. The 2^{kR} quantization vectors are stored in a table. Given a source vector x, the squared distance $\|x - y_i\|^2$ to each entry in the table is calculated, with each calculation requiring k multiplications and $k - 1$ additions. Then with $2^{kR} - 1$ binary comparisons, the closest quantization vector to x is found. Its index is $e(x)$. The decoder has an identical table and uses the index to select the desired quantization vector. Assuming 4 bytes per vector component, the storage for encoding and decoding is $8k2^{kR}$ bytes. The arithmetic complexity, due entirely to the encoder, is 2^{kR} multiplications per source sample. The full search algorithm used with a minimum distortion codebook $C_{k,N}^*$ will be denoted VQ*.

III. Fine-Coarse Vector Quantization

In this section, the fine-coarse approach to vector quantization (FCVQ) is described in detail. There are three principal components: a first stage "fine" quantizing partition $S_f = \{S_{f,1}, \cdots, S_{f,N_f}\}$, a second stage "coarse" codebook $C = \{y_1, \cdots, y_N\}$, and a mapping T: $\{1, \cdots, N_f\} \rightarrow \{1, \cdots, N\}$. The codebook C has dimension k, rate $R = (\log N)/k$ and contains the overall quantization vectors of this approach. The fine quantizing partition S_f is that of a structured vector quantizer with dimension k, rate $R_f > R$, size $N_f >> N$, encoding rule e_f, distortion $D^*(S_f) << D^*(C)$ and an encoding algorithm with low arithmetic complexity. For example, C might be the codebook $C_{k,N}^*$ having smallest distortion among all codebooks with dimension k and size N; S_f might be the quantizing partition of a lattice or tree-structured vector quantizer; and T might be chosen so that $T(j) = i$ if y_i is the closest quantization vector in C to the centroid of $S_{f,j}$. As illustrated in Fig. 2, the fine-coarse vector quantizer, denoted (C, S_f, T), has encoding rule $e^+(x) = T(e_f(x))$. It follows that the quantizing partition S^+ has cells

$$S_i^+ = \{x: e^+(x) = i\} = \bigcup_{j: T(j) = i} S_{f,j}, \quad i = 1, \cdots, N$$

and, as usual, the decoding rule is $d^+(i) = y_i$ and the quantizing rule is $Q^+(x) = y_i$ when $x \in S_i^+$.

Clearly, the FCVQ quantizing partition S^+ is no better for C than the optimal partition S_C^*. Fig. 3 illustrates their differences. It also shows S_f. One may see that the basic idea in FCVQ is to approximate each cell in the optimal partition S_C^* by the union of fine cells from S_f. The benefit is that it is considerably easier to find which fine cell a source vector lies in, than to find which optimal cell it lies in. Clearly, the accuracy of these approximations increases as the fine quantization cells get smaller. Hence, the increase in distortion due to FCVQ encoding, namely

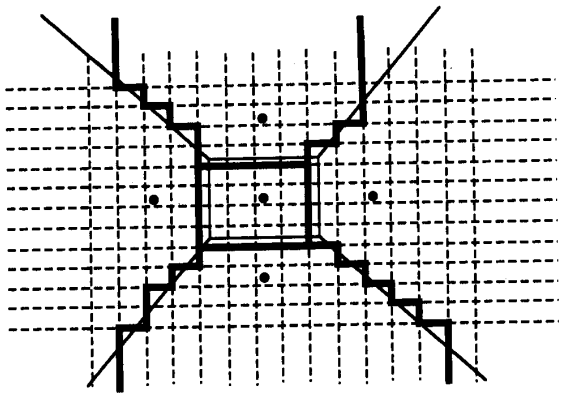


Fig. 3. Quantizing partitions of FCVQ: the dashed lines show the fine partition S_f; the dots show the coarse quantization vectors; the thin solid lines show the optimal coarse partition S_C^*; and the thick lines show the FCVQ partition S^+.

$D(C, S_f, T) - D^*(C)$, will be closely related to the distortion $D^*(S_f)$ of the fine VQ. This relationship will be explored analytically and experimentally in the next two sections.

The low complexity encoding algorithm for FCVQ consists of the encoding algorithm of the fine VQ, followed by a table lookup implementation of T. Specifically, the binary representation of the index $j = e_f(x)$ is used to address a table, whose jth entry is the kR-bit binary representation of $i = T(j)$. The decoding algorithm is simply a table lookup implementation of d^+, just as in the full search algorithm. Since the table lookup implementations of T and d^+ require no multiplications, the overall arithmetic complexity of FCVQ is simply that of the fine VQ encoding algorithm. As for storage, the FCVQ encoder requires whatever storage, denoted s_f, is needed by the encoding algorithm, plus $kRN_f/8$ bytes of storage for the table for T. Adding the storage for the decoding table gives a total of $s_f + (kRN_f/8) + 4k2^{kR}$ bytes for FCVQ.

There are obviously many possible choices for the fine quantizer. In this work we have experimented with two: tree-structured vector quantization (TSVQ) [5], [6] and the product of k uniform scalar quantizers. Specifically, we considered the most basic form of k-dimensional TSVQ in which there is a complete binary tree of depth L with a k-dimensional test vector at each node. Encoding proceeds by finding which of the two test vectors at depth 1 is closer to the given source vector, then finding which of the two children of this test vector is closer, and so on, until one finds a test vector at depth L. The sequence of L binary decisions is the encoder output, so the rate of TSVQ is $R = L/k$. As in full search, the decoder uses table lookup to select a quantization vector, namely, the test vector found at depth L. The storage and arithmetic complexity of TSVQ are summarized in Table I, where it is assumed that at each depth the closer child is found by comparing the inner product of the source vector and the difference between children to a threshold. In comparison

TABLE I
ARITHMETIC COMPLEXITY AND STORAGE

	Arithmetic Complexity	Storage (bytes)
VQ*	2^{kR}	$8k2^{kR}$
TSVQ	kR	$4(2k+1)2^{kR} - 4(k+1)$
FCVQ-U	1	$\frac{kR}{8}2^{kR_f} + 4k2^{kR}$
FCVQ-T	kR_f	$\left(4(k+1) + \frac{kR}{8}\right)2^{kR_f} + 4k2^{kR} - 4(k+1)$

to VQ*, also included in the table, TSVQ has much less arithmetic complexity at the expense of some increase in storage. Its distortion is generally near enough to that of VQ* (cf. [6]) that it represents a good low complexity VQ method. For this reason, we will use it as a benchmark for judging the effectiveness of FCVQ. Fine–coarse vector quantization with TSVQ as the fine quantizer will be denoted FCVQ-T. Its storage and arithmetic complexity are summarized in Table I.

When the fine quantizer is a product of k uniform scalar quantizers, the fine partition is, essentially, a lattice of cubic cells within a larger support cube. We say "essentially" because the cells farthest from the origin (outside the support cube) are unbounded. Encoding proceeds via a sequence of k uniform quantizations. Although there are many ways of implementing a uniform quantizer, to make reasonable comparisons we assume it requires one multiplication per sample. As it requires no appreciable storage, we just assume the storage required is zero. FCVQ based on uniform scalar quantization will be referred to as FCVQ-U. It's storage and arithmetic complexity are also given in Table I.

Our *a priori* expectations were that FCVQ-U, with its very low arithmetic complexity, would be better than FCVQ-T for small values of k and sources with little memory. On the other hand, we expected that FCVQ-T, with its somewhat higher complexity, would be better for larger values of k and sources with more memory, because in such cases the TSVQ first stage could make do with a significantly smaller number of (efficiently placed) fine quantization vectors than could a product quantizer.

Most previous low complexity approaches to vector quantization can be classified according to whether they perform a full search of an unstructured codebook or a low complexity search of a structured codebook. In the former category [7] and [8] present methods that are somewhat similar to FCVQ-U in that a first stage locates the source vector within a small cube or rectangle (though not generally as small a cube as in FCVQ-U), and a second stage identifies the closest codeword to the given source vector. Although these methods require very few or no multiplications, each requires on the order of 2^{kR} logical operations per sample, e.g., binary comparisons or AND's. On the other hand, [9]–[11] are somewhat similar to FCVQ-T in that a first stage uses a search tree of hyperplanes to locate the source vector within some small cell (not generally as small as the fine cells of FCVQ-T). A full search among a "bucket" of candidate quantization vectors is then performed, with one bucket associated with each leaf of the tree. (The buckets are not mutually exclusive.) The trees used in these methods are not as deep as and have fewer leaves than those used in FCVQ-T. Consequently, the tree searching step requires fewer arithmetic operations. However, the full search among candidates in a bucket requires many more operations than the additional tree searching steps of FCVQ-T. Thus, these methods have significantly greater arithmetic complexity than FCVQ-*T*. For example, complexity is reported to be on the order of $(R-1) + 2^k/k$ in [9]. In summary, the previous work on searching unstructured codebooks has substantially larger arithmetic complexity, requires substantially less storage, and achieves slightly lower distortion than FCVQ.

Many approaches to low complexity vector quantization have employed structured codebooks, for example, lattice, transform, gain/shape, tree-structured, multistage [12], and hierarchical [13] quantization. Generally, these encode in a "successive approximation" or "coarse-to-fine" fashion, whereby the quantizing partition becomes finer in two or more stages. This is readily apparent in product, tree-structured and multistage quantization. Although less apparent, it's also true for lattice quantization (the encoding algorithms generally determine on which side of various hyperplanes the source vector lies) and in block transform quantization (the scalar quantization of successive components of the transformed block contribute to finer and finer approximations). These successive approximation methods tend to have lower arithmetic complexity or storage than FCVQ and the other approaches employing unstructured codebooks. But they cannot achieve as low distortion, due to their suboptimum structured codebooks and suboptimum encoding rules. For example, successive approximation searches form quantization cells with fewer faces than the Voronoi regions. In the experimental results of Section V, we include TSVQ as an excellent representative of the structured successive approximation approach.

Another interesting structured approach to VQ is the hierarchical vector quantization of [14], which can be considered to be a form of FCVQ, in that it employs a fine scalar quantization followed by a table lookup lumping of product cells into coarser cells. Unlike the FCVQ discussed here, the lumping proceeds hierarchically in stages, with the result that the total table storage is significantly reduced. This is possible because the codebook is itself designed hierarchically in stages to match the lumping process. Although the highly structured codebook might not be close to optimum for its dimension and rate, this method permits larger dimensions (for a given storage and complexity constraint) and, consequently, may be quite effective. In contrast, in this paper we explore the advantages and disadvantages of searching minimum distortion codebooks.

IV. Average Distortion and Optimization

Although S_f is the only fine VQ feature of direct significance, for the purposes of analysis it is useful to introduce the minimum distortion fine codebook $C^*_{S_f} = \{y_{f,1}, \cdots, y_{f,N_f}\}$ for S_f, where $y_{f,j} = E[X \mid X \in S_{f,j}]$. Let d_f, $Q_f = Q^*_{S_f}$ and $D^*(S_f)$ denote, respectively, the corresponding decoding rule, quantization rule, and distortion. Let U_f, $Y_f U^+$, and Y^+ denote, respectively, $e_f(X)$, $Q_f(X)$, $e^+(X)$, and $Q^+(X)$. It follows that $Y_f = E[X \mid U_f]$. The situation is illustrated in Fig. 4.

To simplify matters, we make the reasonable assumption that

$$y_{f,j} \in S_{f,j}, \qquad j = 1, \cdots, N_f \tag{4.1}$$

which holds in the usual case that the $S_{f,j}$'s are convex. It is easy to show that this implies

$$Q^+(x) = Q^+(Q_f(x)) \tag{4.2}$$

or, equivalently, $Y^+ = Q^+(Y_f)$, so FCVQ may indeed be interpreted as fine quantization followed by coarse. This is the form described in the introduction (see Fig. 1), except that the final stage of quantization is not necessarily Q^*_C. (Later it will be shown that when T is optimally chosen, the final stage is equivalent to quantizing by Q^*_C.) The following shows that the distortion of FCVQ is the sum of the distortions introduced in each stage.

Proposition 4.1:

$$D(C, S_f, T) = D^*(S_f) + D_f(C, S^+) \tag{4.3}$$

where $D^*(S_f) = k^{-1}E\|X - Y_f\|^2$ is the distortion introduced by the first stage and $D_f(C, S^+) \triangleq k^{-1}E\|Y_f - Q^+(Y_f)\|^2$ is the distortion introduced by the second stage when quantizing Y_f.

Proof of Proposition 4.1: Observe that

$$\begin{aligned} kD(C, S_f, T) &= E\|X - Y^+\|^2 \\ &= E\|(X - Y_f) + (Y_f - Y^+)\|^2 \\ &= E\|X - Y_f\|^2 + E\|Y_f - Y^+\|^2 \\ &\quad + 2E((X - Y_f), (Y_f - Y^+)) \end{aligned} \tag{4.4}$$

where (v, w) denotes the inner product $\Sigma_{n=1}^k v_n w_n$. Conditioning the last term on the random variable U_f gives

$$\begin{aligned} &E[((X - Y_f), (Y_f - Y^+)) \mid U_f] \\ &\quad = ((E[X \mid U_f] - Y_f), (Y_f - Y^+)) = 0 \end{aligned}$$

where we have used the facts that Y_f and Y^+ are functions of U_f and that by our choice of the $y_{f,j}$'s, $E[X \mid U_f] = Y_f$. Substituting the above into (4.4) gives

$$D(C, S_f, T) = D^*(S_f) + \frac{1}{k}E\|Y_f - Y^+\|^2.$$

Finally, using (4.2) in the above establishes (4.3). ■

We now consider the selection of the three primary features of FCVQ: S_f, C, and T. There's not much to say about the choice of S_f. One may use any structured quantizer whatsoever; the only guideline is that its arithmetic complexity and average distortion should be small and its storage should be reasonable. On the other hand, as described below, once S_f is chosen, there are optimum choices for T and C, and these are just as easy to implement as any others. (Note that $C^*_{k,N}$, the best codebook for full search is not necessarily the best codebook for S_f.) In the following proposition, whose proof is given in the Appendix, we show that the best C and T are such that the resulting quantizer (C, S^+) is an optimum quantizer for Y_f, and we describe the best C for a given T, and vice versa. Later, we introduce an algorithm that attempts to optimize FCVQ by finding T, C that are simultaneously optimum for each other.

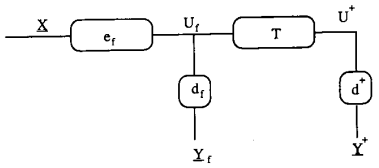


Fig. 4. FCVQ with the fine decoder shown.

Proposition 4.2: Suppose a source $\{X_i\}$, an integer N and a fine partition $S_f = \{S_{f,1}, \cdots, S_{f,N_f}\}$ are given. Let $C^*_{S_f}$ and Y_f be as previously defined. Each of the following presumes FCVQ with fine partition S_f and a coarse codebook of size N.

a) A coarse codebook C and mapping T minimize the distortion of FCVQ if and only if (C, S^+) is a minimum distortion quantizer of size N for Y_f, where S^+ is the resulting FCVQ quantizing partition.

b) Suppose (C', S') is a minimum distortion quantizer for Y_f of size N. Let $C = C'$ and

$$T(j) = i \qquad \text{if } y_{f,j} \in S'_i. \tag{4.5}$$

Then (C, S_f, T) is an FCVQ with minimum distortion.

c) Given a mapping T: $\{1, \cdots, N_f\} \to \{1, \cdots, N\}$, the coarse codebook C that minimizes distortion in FCVQ has quantization vectors

$$y_i = E[X \mid X \in S_i^+] = E\left[X \mid X \in \bigcup_{j:T(j)=i} S_{f,j}\right]. \tag{4.6}$$

d) Given a coarse codebook C of size N, the mapping T that minimizes distortion in FCVQ is

$$T(j) = \arg\min_i \|y_{f,j} - y_i\|. \tag{4.7}$$

e) Equations (4.6) and (4.7) are necessary conditions for the optimality of an FCVQ. □

It is interesting to note that if C satisfies (4.6), not only is each y_i the X centroid of the cell S_i^+, but as shown below, y_i is also the Y_f centroid:

$$\begin{aligned} E[Y_f \mid Y_f \in S_i^+] &= E[Y_f \mid X \in S_i^+] \\ &= E[E[X \mid U_f] \mid X \in S_i^+] = E[X \mid X \in S_i^+] \end{aligned}$$

where the last equality holds because each cell of S^+ is the union of some number of cells of S_f. It follows using

(A.5) that

$$E\|Y^+\|^2 = E\|X\|^2 - kD(C, S_f, T)$$
$$= E\|Y_f\|^2 - kD_f(C, S^+).$$

It is easy to see that if T satisfies (4.7), then in addition to (4.2), we also have

$$Q^+(x) = Q_C^*(Q_f(x)) \tag{4.8}$$

which is the form of FCVQ originally presented in the introduction (see Fig. 1). It follows that

$$D_f(C, S^+) = \frac{1}{k} E\|Y_f - Q_C^*(Y_f)\|^2 = D_f^*(C).$$

Substituting the above into (4.3) gives

$$D(C, S_f, T) = D^*(S_f) + D_f^*(C).$$

Since $D^*(S_f)$ is ordinarily quite small, one expects that $Y_f \cong X$ and, consequently, that $D_f^*(C) \cong D^*(C)$. Using this approximation in the above gives

$$D(C, S_f, T) \cong D^*(S_f) + D^*(C). \tag{4.9}$$

The following proposition, whose proof is given in the Appendix, provides a precise upper bound to $D(C, S_f, T)$.

Proposition 4.3: If T satisfies (4.7) (i.e., T is optimum for S_f and C), then

$$D(C, S_f, T) \le D^*(C) + 2D^*(S_f) + 2\sqrt{D^*(C)D^*(S_f)}. \tag{4.10}$$

□

Since the third term in the above dominates the second, this proposition shows that as the fine quantizer gets finer, $D(C, S_f, T) - D^*(C)$ decreases at least as rapidly as the square root of the fine quantizer distortion $D^*(S_f)$. On the other hand, the experimental results of the next section indicate that $D(C, S_f, T) - D^*(C)$ decreases linearly with $D^*(S_f)$ which is the faster convergence suggested by (4.9).

A. An Optimizing Algorithm

The necessary conditions of (4.6) and (4.7) suggest the following iterative algorithm for optimizing the codebook C and mapping T for a given fine partition S_f. Given S_f and some initial codebook C, alternately apply (4.7) to find the best mapping for the codebook and (4.6) to find the best codebook for the mapping, until no negligible improvements are found. Since each step of this algorithm cannot increase distortion, it produces a sequence of FCVQ's whose distortion converges, although not necessarily to the global minimum. If, additionally, the codebooks and mappings also converge (as one hopes), then (4.6) and (4.7) will hold for the limiting FCVQ. A good choice for starting the algorithm is $C_{k,N}^*$, the best codebook of dimension k and size N.

Because of the difficulty of working with k-dimensional probability distributions, a training sequence approach, as in [15], is advised. Specifically, given an initial codebook C and a suitably long sequence $x^1, x^2, x^3, \cdots, x^L$ of k-dimensional training vectors, one alternates the following steps until no further gains are apparent.

Optimizing T for a given C: Given a codebook $C = \{y_1, \cdots, y_N\}$, let

$$T(j) = \arg\min_i \|y_{f,j} - y_i\|^2, \quad j = 1, \cdots, N_f.$$

Optimizing C for a given T: Given a mapping T, let

$$y_i = \frac{\sum_{j:T(j)=i} y_{f,j}}{\sum_{j':T(j')=i} N_{j'}}, \quad i = 1, \cdots, N$$

where N_j is the number of training vectors that lie in $S_{f,j}$.

Although there is no general approach to optimizing S_f and T for some specified codebook C, in any specific FCVQ, it is wise to tailor the fine partition to C, insofar as possible. For example, with FCVQ-U one can adjust the support region of the scalar quantizer and with FCVQ-T one can tailor the tree to C, as described in [16], [17].

V. Experimental Results

Extensive experiments were run to test the performance of FCVQ. The principal goals were to determine how the distortion of FCVQ compares to that of full search quantization (for example, how rapidly does it decrease as the fine quantizer becomes finer) and to quantitatively evaluate FCVQ performance on the basis of rate, distortion, arithmetic complexity, and storage. A second set of experiments tested the algorithm for optimizing C and T for a given fine partition S_f.

A. First Experiments

Two types of FCVQ were tested: FCVQ-T and FCVQ-U, corresponding to fine partitions generated by TSVQ and uniform scalar quantization, respectively. In addition, a few experiments were performed with Lloyd–Max scalar quantizers generating the fine partition. But since the improvements over uniform quantization were so small, the results are not reported here. All tests were conducted on three sources: IID Gaussian, Gauss–Markov with correlation coefficient $\rho = 0.8$, and speech sampled at 6.5 kHz (provided by R. M. Gray), each normalized to zero mean and unit variance. The coarse codebooks C were either minimum distortion codebooks $C_{k,N}^*$ or TSVQ codebooks. The latter were included because TSVQ's are much simpler to design, and because it seemed likely that they would work well when the fine partition was generated by deeper levels of the same tree.

The "minimum distortion codebooks" were designed with the LBG algorithm [15], using binary splitting and training sequences of 2^{20} samples of each of the Gaussian sources and 640 000 samples of speech. The time consuming nature of this algorithm limited our experiments to codebooks with rate $R = 1$ and dimensions $k = 2, 3, 4, 6, 8$ and also with rate $R = 2$ and dimensions $k = 2, 3, 4$. Although there is no guarantee that the LBG algo-

rithm produced the true minimum distortion codebooks, the experimental results presented here have to do with the increase in distortion of FCVQ over full search, which we believe is largely independent of the codebook, and with the absolute comparison of FCVQ with TSVQ, making our comparisons overly pessimistic if in fact the LBG algorithm produced suboptimal codebooks.

For each dimension k of interest, a binary TSVQ tree of depth 14 was designed by the training sequence method of [6]. The training sequences mentioned above were not long enough to go deeper, and even the partitions at depth 14 did not appear to be particularly well designed. All TSVQ fine partitions and coarse codebooks in our experiments came from intermediate and terminal depths of these trees. The coarse codebooks were the centroids of the associated partitions. The fine partitions, taken at depths L from kR up to the smaller of 14 and 2^{kR}, had $N_f = 2^L$ cells and rates $R_f = L/k$. There was no reason to consider depths greater than 2^{kR} because encoding with these would have more arithmetic complexity than full search. With TSVQ fine partitions, the mapping T was chosen to be the best for the given S_f and C, as specified by (4.7).

In the experiments with FCVQ-U, the uniform scalar quantizers were chosen to have an odd number, $2M + 1$, of cells of width Δ, symmetrically positioned about the origin. For simplicity, we used the mapping $T(j) = i$ when the geometric centroid of the jth cell is closer to $\boldsymbol{y}_i$ than to any other quantization vector in C. For small Δ's, this approximates the minimum distortion mapping specified by (4.6). For experiments with FCVQ rate $R = 1$, the integer M ranged from 1 to 13, and for $R = 2$, it ranged from 4 to 200. The cell width Δ was chosen by a simple iterative algorithm to minimize the distortion of the resulting FCVQ. The lengths $\Gamma = (2M + 1)\Delta$ of the resulting support intervals are shown in Table II. As one would expect, Γ increased steadily from the first to the second value shown there as M increased. As also to be expected, the support intervals were not influenced by dimension but enlarged with the rate R. The speech source required substantially larger support cubes than did the two Gaussian sources, probably due to its first-order distribution having a heavier tail. Because it was costly to optimize Δ, for some of the very large fine partitions ($R = 1$, $k = 6$, $M \geq 4$ and $k = 8$, $M = 2$), we simply tried several values of Δ in order to find a reasonable value.

Let us begin the discussion of how well FCVQ performed by comparing its distortion, $D(C, S_f, T)$, to $D^*(C)$, the distortion of full search of the same codebook. For brevity, let $D^+ = D(C, S_f, T)$ and $D^* = D^*(C)$. An examination of our experimental results with the minimum distortion coarse codebooks showed that as the fine quantizers get finer, $D^+ - D^*$ decreases essentially in proportion to $D^*(S_f)$ with proportionality constants given in Table III. These constants depend upon the source, the type of fine quantizer, and the rate R, but do not depend upon the dimension k or the fineness of the fine quantizer (M for uniform scalar quantization and L for TSVQ). The values shown in Table III are accurate to about 10% over the range of k, M, and L. For the uniform scalar quantizers, $D^*(S_f)$ was approximated with the formula $\Delta^2/12$, which becomes increasingly accurate as M and R_f increase. Notice that $D^*(S_f)/(D^+ - D^*)$ decreases a little with rate and increases noticeably with source memory (from IID to Markov to speech).

TABLE II
SUPPORT INTERVAL LENGTHS Γ OF THE UNIFORM SCALAR FINE QUANTIZERS

	IID	Markov	Speech
$R = 1$	3 ↗ 5	3 ↗ 5	8 ↗ 13
$R = 2$	4 ↗ 6	4.5 ↗ 6	13 ↗ 19

TABLE III
$D^*(S_f)/(D^+ - D^*)$ FOR THE MINIMUM DISTORTION COARSE CODEBOOKS

Fine Q.		IID	Markov	Speech
Un. Sc.	$R = 1$	1	2	6
	$R = 2$	1	1.2	4
TSVQ	$R = 1$	1.5	2.5	3.5
	$R = 2$	1.2	1.4	2

When the TSVQ coarse codebooks were used with uniform scalar fine quantization, the ratios of $D^*(S_f)$ to $D^+ - D^*$ were essentially the same as shown in Table III. However, as shown in Table IV, when used with TSVQ fine quantization, the ratios were much larger. This better performance is due to the special match of tree and codebook, and becomes less significant as R_f increases. (The fine partition is a refinement of the partition that would be generated by direct TSVQ quantization.)

Let us next examine how rapidly D^+ approaches D^* as the fine quantizing partition gets finer. We begin by factoring $(D^+ - D^*)/D^*$ into two terms:

$$\frac{D^+ - D^*}{D^*} = \frac{D^+ - D^*}{D^*(S_f)} \frac{D^*(S_f)}{D^*}. \tag{5.1}$$

As we have seen, the first term on the right side remains essentially constant as S_f gets finer (except with TSVQ as both the fine and coarse quantizer). So the rate at which D^+ approaches D^* is determined by the second term.

For FCVQ-U

$$D^*(S_f) \cong \frac{\Delta^2}{12} = \frac{\Gamma^2}{12} 2^{-2R_f}. \tag{5.2}$$

With this and Zador's formula (A.6) as approximations, we find

$$\frac{D^*(S_f)}{D^*} \cong \frac{\Gamma^2}{12 c_k \|p_k\|_{k/k+2}} 2^{-2(R_f - R)} \tag{5.3}$$

where p_k, c_k, and $\|p_k\|$ are as defined in the Appendix. For FCVQ-T we found, experimentally, that

$$D^*(S_f) \cong b c_k \|p_k\|_{k/k+2} 2^{-2(R_f - R)} \tag{5.4}$$

TABLE IV
THE RANGE OF $D^*(S_f)/(D^+ - D^*)$ AS R_f GOES FROM SMALL TO LARGE VALUES, WITH TSVQ AS THE FINE AND COARSE QUANTIZERS

	IID	Markov	Speech
$R = 1$	70 \ 5	100 \ 5	70 \ 50
$R = 2$	22 \ 2	35 \ 4	25 \ 4

where b is in the range 1 to 1.3. This and Zador's formula (A.6) give

$$\frac{D^*(S_f)}{D^*} \cong b2^{-2(R_f - R)}. \tag{5.5}$$

Thus, in both cases we expect $(D^+ - D^*)/D^*$ to decrease exponentially with exponent $-2R_f$. It is not immediately apparent whether the factor in front of the exponential term is smaller (i.e., better) for FCVQ-U or FCVQ-T. However, it is clear that for the former this factor increases with both R_f and k, the first because Γ increases with R_f, the second because c_k and $\|p_k\|_{k/k+2}$ decrease with k, the latter most noticeably for sources with memory. This indicates that the performance of FCVQ-T improves relative to that of FCVQ-U as dimension increases, especially for sources with lots of memory. This is basically due to the fact that uniform quantization distributes its cells uniformly, whereas TSVQ puts them where they do the most good.

With the expectation that D^+ converges exponentially to D^*, as in (5.3) and (5.5), we computed the loss factor

$$\beta = \frac{D^+ - D^*}{D^*} 2^{2(R_f - R)} \tag{5.6}$$

for the FCVQ's based on minimum distortion codebooks. From (5.1)–(5.6) one expects β to be approximately $\Gamma^2/(12c_k\|p_k\|_{k/k+2})$ for FCVQ-U and b for FCVQ-T. As anticipated, β varied little with R_f, indicating that $(D^+ - D^*)/D^*$ does indeed decrease exponentially with exponent $-2R_f$. Table V gives representative values of β, and Figs. 5–7 plot β versus $R_f - R$ for the rate $R = 2$ schemes. The small upward slope of the plots for FCVQ-U is probably due to the increasing size of Γ, and the upward tailing of the FCVQ-T plots for large R_f is probably due to poorly designed last stages of the tree.

For FCVQ-U with TSVQ coarse codebooks, the values of β (not shown) were quite similar to those shown in Table V for the minimum distortion codebooks. And for FCVQ-T with TSVQ fine quantizers, the values of β (also not shown) were much smaller than those for the minimum distortion codebook, reflecting the special match of tree and codebook indicated in Table IV.

Having found that $D^+ - D^*$ decreases in proportion to $2^{-2R_f} = N_f^{-2/k}$ and using the fact illustrated in Table I, that N_f is the dominant term in the storage of FCVQ, we see that $D^+ - D^*$ is proportional to $(s^+)^{-2/k}$, where s^+ is the available storage. As for arithmetic complexity, with FCVQ-U the arithmetic complexity does not increase as the fine quantizer becomes finer. On the other hand, for

TABLE V
THE LOSS FACTOR β FOR FCVQ WITH MINIMUM DISTORTION CODEBOOKS

Fine Q.		IID	Markov	Speech
Un. Sc.	$R = 1$	1	2	5
	$R = 2$	2	3	8
TSVQ	$R = 1$	0.8	0.5	0.4
	$R = 2$	1.1	1	0.8

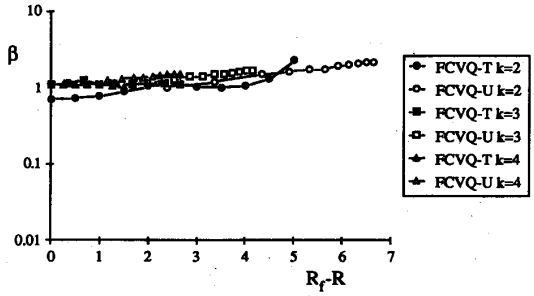


Fig. 5. β versus $R_f - R$ for an IID Gaussian source, $R = 2$.

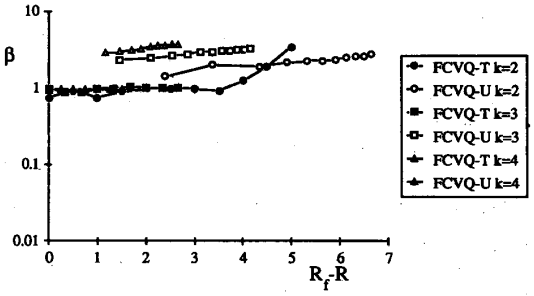


Fig. 6. β versus $R_f - R$ for a Gauss–Markov source, $R = 2$.

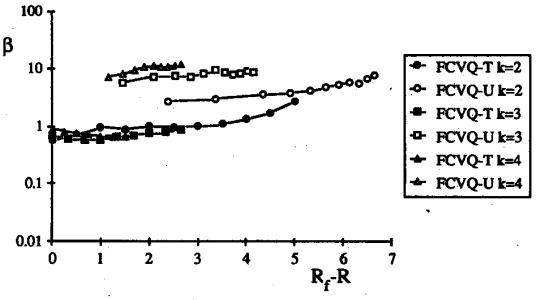


Fig. 7. β versus $R_f - R$ for speech, $R = 2$.

FCVQ-T the arithmetic complexity is kR_f, so $D^+ - D^*$ is proportional to $2^{-2K^+/k}$, where K^+ denotes the arithmetic complexity.

B. Distortion, Complexity, and Storage

Let us next examine the overall effectiveness of FCVQ on the basis of rate, distortion, arithmetic complexity, and storage. Figs. 8–13 plot storage versus distortion for the

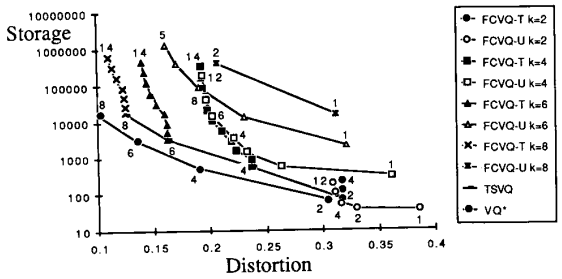


Fig. 8. Storage versus distortion for speech, $R = 1$.

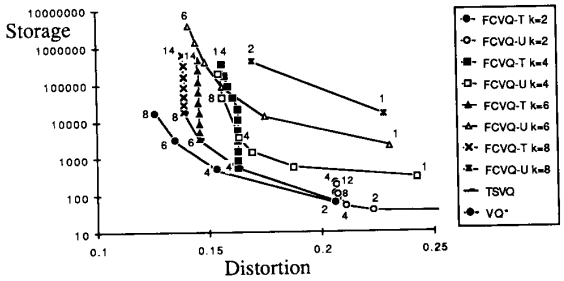


Fig. 9. Storage versus distortion for a Gauss-Markov source, $R = 1$.

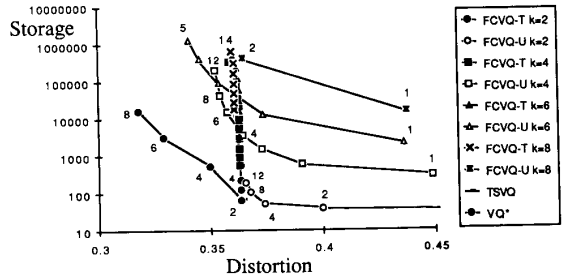


Fig. 10. Storage versus distortion for an IID Gaussian source, $R = 1$.

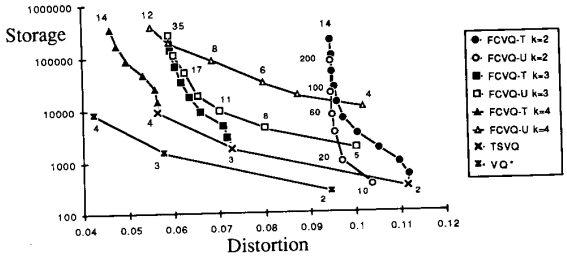


Fig. 11. Storage versus distortion for speech, $R = 2$.

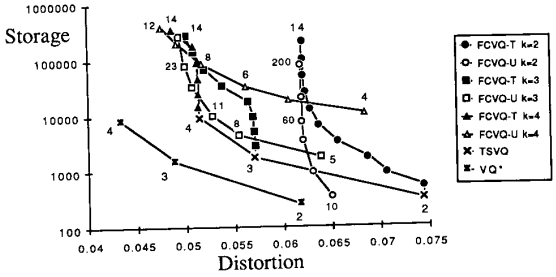


Fig. 12. Storage versus distortion for a Gauss-Markov source, $R = 2$.

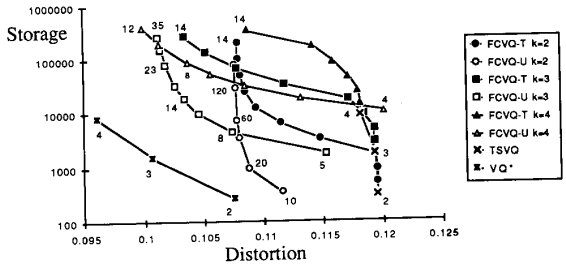


Fig. 13. Storage versus distortion for an IID Gaussian source, $R = 2$.

three sources and various FCVQ schemes of rates 1 and 2. For comparison, the storage and distortion of VQ* (full search of minimum distortion codebooks) and of TSVQ are also shown. Let us first look at Fig. 8 for speech and rate $R = 1$. For each of dimensions $k = 2, 4, 6, 8$ there is a solid line connecting the distortion and storage of FCVQ-U for various values of M (recall $N_f = (2M + 1)^k$). Some of the points have M labelled. There is a solid line connecting points representing the distortion and storage of FCVQ-T for all L from k to $L_{\max} = \min\{14, 2^{kR}\}$ (the point representing the largest L is labelled with $L_{\max}$). There is also a solid line connecting the VQ* points (each labelled with k) and one connecting the TSVQ points (also labelled with k). Notice that the FCVQ-T curves rise vertically out of the TSVQ point of the same dimension. When examining this figure, one should also be cognizant of the arithmetic complexities of the various schemes: 2^{kR} for VQ*, kR for TSVQ, L for FCVQ-T, and 1 for FCVQ-U.

A typical FCVQ-U curve (e.g., $k = 4, 6$) is convex and increases smoothly to infinity (from the right) as D^+ approaches D^*. The $k = 8$ curve is an exception simply because we were not able to choose R_f large enough to observe the asymptotic approach to D^*. In virtually all cases the minimum distortion codebooks performed at least as well as the TSVQ codebooks.

A typical FCVQ-T curve (e.g., $k = 6$) rises almost vertically from the point representing TSVQ with the given dimension, then curves to the left and approaches infinity as D^+ approaches D^* from the right. The almost vertical rise is due to the facts that for small values of L the TSVQ codebooks are best, that FCVQ-T with a TSVQ codebook and $L = kR$ is, essentially, just TSVQ, and that $D^*(C)$ for the TSVQ codebooks is, unfortunately for FCVQ-T, not much less than the distortion of TSVQ. Consequently, the distortion of FCVQ-T can decrease only a little from that of TSVQ as L increases. The sharp bend to the left is due to the fact that for sufficiently large L, the minimum

TABLE VI
CODES FOR IID GAUSS

k	Code	M	D	Complexity	Storage (bytes)
2	VQ*		0.364	4	64
3	VQ*		0.358	8	192
4	VQ*		0.350	16	512
2	TSVQ		0.364	2	68
3	TSVQ		0.363	3	208
4	TSVQ		0.363	4	556
6	TSVQ		0.363	6	3300
8	TSVQ		0.361	8	17372
3	FCVQ-U	5	0.367	1	595
3	FCVQ-U	6	0.365	1	920
4	FCVQ-U	5	0.360	1	7576
4	FCVQ-U	6	0.358	1	14537
4	FCVQ-U	7	0.356	1	25569
4	FCVQ-U	8	0.355	1	42017
6	FCVQ-U	3	0.354	1	117649
6	FCVQ-U	4	0.345	1	531441
6	FCVQ-U	5	0.341	1	1771561
2	VQ*		0.108	16	256
3	VQ*		0.101	64	1536
2	TSVQ		0.120	4	308
3	TSVQ		0.119	6	1776
4	TSVQ		0.118	8	9196
2	FCVQ-U	10	0.112	1	349
2	FCVQ-U	20	0.109	1	969
2	FCVQ-U	40	0.108	1	3409
3	FCVQ-U	8	0.107	1	4453
3	FCVQ-U	11	0.105	1	9893
3	FCVQ-U	14	0.103	1	19060
3	FCVQ-U	17	0.1025	1	32924
4	FCVQ-U	11	0.1005	1	283937
4	FCVQ-U	12	0.100	1	394721

TABLE VII
CODES FOR GAUSS-MARKOV

k	Code	M	D	Complexity	Storage (bytes)
2	FVQ		0.206	4	64
3	VQ*		0.170	8	192
4	VQ*		0.153	16	512
2	TSVQ		0.206	2	68
3	TSVQ		0.187	3	208
4	TSVQ		0.163	4	556
6	TSVQ		0.146	6	3300
8	TSVQ		0.139	8	17372
2	FCVQ-U	8	0.207	1	104
3	FCVQ-U	2	0.199	1	143
3	FCVQ-U	3	0.185	1	225
3	FCVQ-U	4	0.179	1	369
3	FCVQ-U	5	0.176	1	595
4	FCVQ-U	3	0.169	1	1457
4	FCVQ-U	4	0.163	1	3537
4	FCVQ-U	5	0.160	1	7577
6	FCVQ-U	3	0.157	1	89772
2	VQ*		0.062	16	256
3	VQ*		0.049	64	1536
2	TSVQ		0.074	4	308
3	TSVQ		0.057	6	1776
4	TSVQ		0.051	8	9196
2	FCVQ-U	10	0.065	1	349
2	FCVQ-U	20	0.063	1	969
3	FCVQ-U	8	0.056	1	4452
3	FCVQ-U	11	0.053	1	9893
3	FCVQ-U	14	0.051	1	19060

TABLE VIII
CODES FOR SPEECH

k	Code	M	D	Complexity	Storage (bytes)
2	VQ*		0.307	4	64
3	VQ*		0.243	8	192
4	VQ*		0.191	16	512
2	TSVQ		0.317	2	68
3	TSVQ		0.266	3	208
4	TSVQ		0.234	4	556
6	TSVQ		0.162	6	3300
8	TSVQ		0.126	8	17372
2	FCVQ-U	6	0.313	1	74
3	FCVQ-U	3	0.271	1	225
3	FCVQ-U	4	0.259	1	369
3	FCVQ-U	5	0.253	1	595
4	FCVQ-U	3	0.233	1	1457
4	FCVQ-U	4	0.221	1	3537
4	FCVQ-U	5	0.208	1	7577
6	FCVQ-U	2	0.231	1	13255
6	FCVQ-U*	2	0.190	1	13255
6	FCVQ-U	3	0.191	1	89772
6	FCVQ-U*	3	0.164	1	89772
2	VQ*		0.095	16	256
3	VQ*		0.058	64	1536
2	TSVQ		0.112	4	308
3	TSVQ		0.072	6	1776
4	TSVQ		0.056	8	9196
2	FCVQ-U	10	0.104	1	349
2	FCVQ-U	20	0.097	1	969
3	FCVQ-U	8	0.080	1	4452
3	FCVQ-U	11	0.070	1	9893
3	FCVQ-U	14	0.065	1	19060

distortion codebooks outperform the TSVQ codebooks. Then, as with FCVQ-U, storage increases smoothly to infinity as D^+ approaches D^* from the right.

Figs. 9–13 can be interpreted similarly. Notice that the Gauss–Markov results are quite similar to those of speech, whereas the IID Gaussian results are different in that the TSVQ points rise almost vertically as k increases, thus making it difficult to distinguish the various TSVQ and FCVQ-T points, especially for rate $R = 1$.

When rate, distortion, storage and arithmetic complexity are all taken into account, one sees, unfortunately, that the FCVQ-T schemes offer no advantages over TSVQ. Although for a given dimension k their distortion decreases below that of TSVQ, this is at the expense of increased storage and arithmetic complexity. One may see from the figures that there are TSVQ schemes with larger k that are better in all respects. On the other hand, the arithmetic complexity of FCVQ-U is so low that it stands up well to TSVQ. Although for the speech and Gauss–Markov sources, the points representing distortion/storage of FCVQ-U generally lie above and to the right of those representing TSVQ, the arithmetic complexity of the former is so much smaller that it is an attractive alternative. That is, for speech and Gauss–Markov sources, FCVQ-U offers the capability of trading an increase in distortion or storage for a decrease in arithmetic complexity. The IID Gaussian source is different in that TSVQ does not do very well and FCVQ-U offers a way to achieve simultaneously lower distortion, storage and arithmetic complexity than TSVQ. Tables VI–VIII list a number of specific FCVQ-U schemes that are attractive relative to TSVQ. For comparison, VQ* and TSVQ are also in-

cluded. A good way to examine these tables is to focus on a particular TSVQ scheme, look for FCVQ-U schemes with roughly the same distortion or storage, then compare arithmetic complexities.

C. Optimizing T and C for a Given S_f

For uniform scalar fine quantization we implemented the algorithm for optimizing T and C for a given fine partition S_f. Specifically, for all three sources the algorithm was run with rate $R = 1$, with dimensions $k = 3, 4, 6$, with several values of M, with Δ chosen as before and with $C^*_{k,2^k}$ as the initial codebook. In almost all cases, it was found that the decrease in distortion over the original FCVQ-U method ($C^*_{k,2^k}$, S_f, T) was small (less than 5%). The only exceptions were for speech with $k = 6$ and $M = 2, 3$, where the algorithm decreased distortion in the range of 15%. We also tried varying Δ from the originally chosen values, but found that this could further decrease distortion by at most 1%. The best results for $k = 6$ and $M = 2, 3$ are included in Table VIII as the points marked FCVQ-U*.

VI. Conclusions

Fine-coarse vector quantization has been introduced and its performance has been analyzed. It was shown that FCVQ-U, has much lower arithmetic complexity than TSVQ and, in many cases, gives comparable distortion with the same or somewhat higher storage. Replacing the uniform scalar quantizer with a lattice quantizer can be expected to further reduce the distortion, at the expense of a small increase in arithmetic complexity and with no effect on storage. On the other hand, FCVQ-T scheme showed no advantages over TSVQ. However, the tree design methods of [16] and [17] hold some promise of substantial improvements.

There are several ways to summarize the performance of FCVQ. One way is to view it (in particular FCVQ-U) as a low complexity way to achieve less distortion than structured quantizers such as TSVQ, which it does by using a better codebook and doing a better job of approximating the Voronoi regions. This is most advantageous for sources for which the latter do not have performance close to the rate-distortion function. (For example, the performance of TSVQ on an IID Gaussian source saturates as the dimension k increases, rather than approaches the rate-distortion function.) However, this reduction in distortion has a substantial cost in storage. Another viewpoint is that FCVQ is a way to trade increased storage for decreased arithmetic complexity at distortions comparable to what other structured quantizers achieve. Accordingly, one expects that FCVQ will become increasingly attractive as the cost of storage decreases. Finally, there are a number of cases where FCVQ outperforms TSVQ in the sense of achieving comparable or lower distortion with comparable storage and much less arithmetic complexity.

Appendix
Properties of Minimum Mean-Squared Error Quantizers

1) Given a source $\{X_i\}$ and quantizing partition S, there is a unique minimum distortion codebook C^*_S, whose quantization vectors are the centroids

$$y_i = E[\boldsymbol{X}|\boldsymbol{X} \in S_i], \qquad i = 1, \cdots, N. \tag{A.1}$$

Equivalently, the unique minimum distortion quantization rule consistent with S is $Q^*_S(\boldsymbol{x}) = E[\boldsymbol{X}|\boldsymbol{X} \in S_i]$, when $\boldsymbol{x} \in S_i$.

2) The best partition for a codebook C is comprised of the Voronoi cells. Because the second stage of FCVQ will, in effect, be quantizing discrete variables, we need the following careful statement of this principal. The Voronoi cells $\{V_1, \cdots, V_N\}$ corresponding to a codebook $C = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_N\}$, are the (nondisjoint) sets

$$V_i \triangleq \{\boldsymbol{x}: \|\boldsymbol{x} - \boldsymbol{y}_i\| \le \|\boldsymbol{x} - \boldsymbol{y}_j\|, \qquad j = 1, \cdots, N\}.$$

A partition S generates minimum distortion for a source X and codebook C if and only if

$$\Pr(\boldsymbol{X} \in S_i - V_i) = 0, \qquad i = 1, \cdots, N.$$

In other words, each S_i must be contained in the corresponding Voronoi cell V_i, except for a set of probability zero. Equivalently, a partition generates minimum distortion if and only if the resulting quantization rule Q satisfies

$$Q(\boldsymbol{x}) = \arg\min_{\boldsymbol{y} \in C} \|\boldsymbol{x} - \boldsymbol{y}\|^2 \qquad \text{with probability 1.} \tag{A.2}$$

3) If a source and quantizer satisfy (A.1), then [20]

$$\text{a)} \quad E[\boldsymbol{Y}] = E[\boldsymbol{X}] \tag{A.3}$$

$$\text{b)} \quad \boldsymbol{Y} = E[\boldsymbol{X}|e(\boldsymbol{X})] = E[\boldsymbol{X}|\boldsymbol{Y}] \tag{A.4}$$

$$\text{c)} \quad E\|\boldsymbol{X} - \boldsymbol{Y}\|^2 = E\|\boldsymbol{X}\|^2 - E\|\boldsymbol{Y}\|^2. \tag{A.5}$$

4) The following result, due to Zador [18], gives an approximate formula for $D^*_{k,N}$ for large values of N:

$$\lim_{N \to \infty} D^*_{k,N} N^{2/k} = c_k \|p_k\|_{k/(k+2)} \tag{A.6}$$

where c_k is a constant (not depending upon N or p_k) that decrease from $1/12$ to $1/2\pi e$ as k goes from 1 to ∞, where $p_k(\boldsymbol{x})$ is the probability density of $\boldsymbol{X} = (X_1, \cdots, X_k)$, and where $\|p_k\|_{k/(k+2)} = (\int (p_k(\boldsymbol{x}))^{k/(k+2)} d\boldsymbol{x})^{(k+2)/k}$. For stationary sources, $\|p_k\|_{k/(k+2)}$ decreases to $2^{2h_\infty(X)}$ as $k \to \infty$ [19] (although not necessarily monotonically), where $h_\infty(X) = \lim_{n \to \infty} 1/n h(X_1, \cdots, X_n)$ is the differential entropy rate of the source $\{X_i\}$. Among sources with similar first-order densities, those with more memory tend to have smaller values of $h_\infty(X)$ and, consequently, larger decreases in $\|p_k\|_{k/(k+2)}$ with increasing k. □

Proof of Proposition 4.2: We begin with b). Suppose (C', S') is a minimum distortion quantizer for $\boldsymbol{Y}_f$ of size N. Let (C, T) be an FCVQ with $C = C'$, with T defined by (4.5), and with overall quantizing partition S^+.

Observe that (4.5) implies $D_f(C', S') = D_f(C, S^+)$. Now let $(\hat{C}, S_f, \hat{T})$ be any other FCVQ and let $\hat{S}^+$ denote its overall quantizing partition. Then using Proposition 4.1,

$$\begin{aligned} D(\hat{C}, S_f, \hat{T}) &= D^*(S_f) + D_f(\hat{C}, \hat{S}^+) \\ &\geq D^*(S_f) + D_f(C', S') \\ &= D^*(S_f) + D_f(C, S^+) \\ &= D^*(C, S_f, T) \end{aligned}$$

and this shows that (C, S_f, T) is an optimum FCVQ.

a) Let (C, S_f, T) be an FCVQ such that (C, S^+) is a minimum distortion quantizer for Y_f. Then for any other FCVQ $(\hat{C}, S_f, \hat{T})$

$$\begin{aligned} D(\hat{C}, S_f, \hat{T}) &= D^*(S_f) + D_f(\hat{C}, \hat{S}^+) \\ &\geq D^*(S_f) + D_f(C, S^+) \\ &= D^*(C, S_f, T) \end{aligned}$$

which shows (C, S_f, T) is a minimum distortion FCVQ. Conversely, suppose (C, S_f, T) is a minimum distortion FCVQ. Let (C, S^+) be the resulting quantizer for Y_f. Let (C', S') be any other quantizer for Y_f and let $(\hat{C}, S_f, \hat{T})$ be an FCVQ for X with $\hat{C} = C'$, with $\hat{T}$ defined from S' by (4.5), and with overall quantizing partition $\hat{S}^+$. Then

$$\begin{aligned} D_f(C', S') &= D_f(C', \hat{S}^+) \\ &= D(C', S_f, \hat{T}) - D^*(S_f) \\ &\geq D(C, S_f, T) - D^*(S_f) \\ &= D_f(C, S^+). \end{aligned}$$

This shows that (C, S^+) is a minimum distortion quantizer for Y_f, and completes the proof of a).

c) The fine partition S_f and the mapping T determine the overall quantizing partition S^+. Accordingly, by (A.1) the minimum distortion quantization vectors are the centroids $y_i = E[X|X \in S_i^+]$.

d) First note that given S_f and C, the best mapping T is

$$T(j) = \arg\min_{i \in \{1, \cdots, N\}} E[\|X - y_i\|^2 | X \in S_{f,j}]. \quad \text{(A.7)}$$

However, one may straightforwardly show that for any i, j

$$\begin{aligned} &E[\|X - y_i\|^2 | X \in S_{f,j}] \\ &\quad = E[\|X - y_{f,j}\|^2 | X \in S_{f,j}] + \|y_i - y_{f,j}\|^2 \end{aligned}$$

and this implies (A.7) is equivalent to (4.7).

e) Follows immediately from (c) and (d). ■

Proof of Proposition 4.3: Let Y_f, Y^+, and Y_C denote, respectively, $Q_f(X)$, $Q^+(X) = Q_C(Q_f(X))$, and $Q_C^*(X)$. Together (4.8) and Proposition 4.1 imply

$$\begin{aligned} &kD(C, S_f, T) \\ &\quad = kD^*(S_f) + E\|Y_f - Q_C^*(Y_f)\|^2 \\ &\quad \leq kD^*(S_f) + E\|Y_f - Q_C^*(X)\|^2 \\ &\quad = kD^*(S_f) + E\|(X - Y_f) - (X - Y_C)\|^2 \\ &\quad \leq kD^*(S_f) + (\sqrt{E\|X - Y_f\|^2} + \sqrt{E\|X - Y_C\|^2})^2 \\ &\quad = kD^*(C) + 2kD^*(S_f) + 2\sqrt{D^*(C)D^*(S_f)} \end{aligned}$$

where the first inequality follows from the fact (see (A.2)) that $Q_C(Y_f)$ is closer to Y_f than any other quantization vector, e.g., $Q_C^*(X)$. The second inequality follows from Minkowski's inequality. Dividing the above by k establishes (4.10) and completes the proof of Proposition 4.3. ■

References

[1] R. M. Gray, "Vector quantization," *IEEE ASSP Mag.*, vol. 1, pp. 4-29, Apr. 1984.

[2] A. Gersho and V. Cuperman, "Vector quantization: A pattern matching technique for speech coding," *IEEE Commun. Mag.*, vol. 21, pp. 15-21, Dec. 1983.

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

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

[5] 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. 367-376, Aug. 1980.

[6] R. M. Gray and Y. Linde, "Vector quantizers and predictive quantizers for Gauss-Markov sources," *IEEE Trans. Commun.*, vol. COM-30, pp. 381-389, Feb. 1982.

[7] D.-Y. Cheng, A. Gersho, B. Ramamurthi, and Y. Shoham, "Fast search algorithms for vector quantization and pattern matching," in *Proc. ICAASP*, Mar. 1984, pp. 9.11.1-9.11.4.

[8] M. R. Soleymani and S. D. Morgera, "A high-speed search algorithm for vector quantization," in *Proc. ICAASP*, 1987, pp. 45.6.1-45.6.3.

[9] D.-Y. Cheng and A. Gersho, "A fast codebook search algorithm for nearest neighbor pattern matching," in *Proc. ICAASP*, 1986, pp. 6.14.1-6.14.4.

[10] A. Lowry, S. Q. A. M. A. Hossain, and W. Millar, "Binary search trees for vector quantization," in *Proc. ICAASP*, 1987, pp. 51.8.1-51.8.4.

[11] W. Equitz, "Fast algorithms for vector quantization picture coding," in *Proc. ICAASP*, 1987, pp. 18.1.1-18.1.4.

[12] B.-H. Juang and A. H. Gray, "Multiple stage vector quantization for speech coding," in *Proc. ICAASP*, Apr. 1982, pp. 597-600.

[13] A. Gersho and Y. Shoham, "Hierarchical vector quantization of speech with dynamic codebook allocation," in *Proc. ICAASP*, 1984, pp. 10.9.1-10.9.4.

[14] P. C. Chang, J. May, and R. M. Gray, "Hierarchical vector quantizers with table-lookup encoders," *Proc. ICC*, vol. 3, pp. 1452-1455, 1985.

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

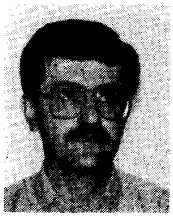
[16] N. Moayeri and D. L. Neuhoff, "Tree based fast quantization," in *Proc. 21st Conf. Inform. Sci. Syst.*, Mar. 1987, pp. 250-255.

[17] N. Moayeri and D. L. Neuhoff, "Decision trees for vector quantizer codebook searching," in *Proc. ICAASP*, Apr. 1988, pp. 255-258.

[18] P. L. Zador, "Asymptotic quantization error of continuous signals and the quantization dimension," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 139-149, Mar. 1982.

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

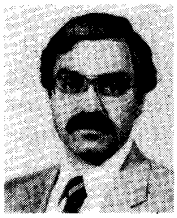
[20] N. C. Gallagher and J. A. Bucklew, "Properties of the minimum mean-squared error block quantizers," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 105-107, July 1982.



Nader Moayeri (S'79-M'80) was born in Hamadan, Iran, on August 31, 1956. He studied electrical engineering at the Sharif (formerly Arya-Mehr) University of Technology, Tehran, Iran, from 1974 to 1978. He received the M.S.E.E., M.S.C.I.C.E., and Ph.D. degrees in electrical engineering-systems from the University of Michigan, Ann Arbor, MI, in 1980, 1981, and 1986, respectively.

He joined the Department of Electrical and Computer Engineering at Rutgers, the State University of New Jersey, New Brunswick, NJ, in September 1986, where he is currently an Assistant Professor. His research interests are in data compression, joint source and channel coding, mobile radio communications, and information theory.

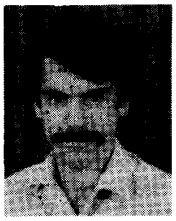
Dr. Moayeri has served the Princeton Section of the IEEE in various capacities, among which was founding the Information Theory Chapter. He is currently Chairman of the Information Theory Chapter.



David L. Neuhoff (S'72-M'74-SM'83) was born in Rockville Centre, NY, on August 18, 1948. He received the B.S.E. degree from Cornell University, Ithaca, NY, in 1970 and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1972 and 1974, respectively.

Since September 1974 he has been a Professor with the Electrical Engineering and Computer Science Department of the University of Michigan, Ann Arbor, MI. He was Associate Chairman from 1984 to 1989. From September 1989 through June 1990 he was on leave at AT & T Bell Laboratories, Murray Hill, NJ. His research and teaching interests are in communication and information theory, especially source coding, quantization, Shannon theory, and coding for magnetic recording.

Dr. Neuhoff is a member of Eta Kappa Nu, Tau Beta Pi, and Sigma Xi. He is Associate Editor for Source Coding for the IEEE TRANSACTIONS ON INFORMATION THEORY. He is a past Chairman of the Southeastern Michigan Chapter of Division I, and was Cochairman of the 1986 IEEE International Symposium on Information Theory in Ann Arbor.



Wayne Stark (S'77-M'78) was born in Illinois on February 26, 1956. He received the B.S. (with highest honors), M.S., and Ph.D. degrees in electrical engineering from the University of Illinois, Urbana, in 1978, 1979, and 1982, respectively.

Since September 1982 he has been at the Department of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor, where he is an Associate Professor. From 1984 to 1989 he was Editor for Communication Theory of the IEEE TRANSACTIONS ON COMMUNICATIONS in the area of spread-spectrum communications. His research interests are in the areas of information and coding and communication theory, especially for spread-spectrum communication systems.

Dr. Stark is a member of Eta Kappa Nu, Phi Kappa Phi, and Tau Beta Pi. He was named a 1985 Presidential Young Investigator by the National Science Foundation. He helped to plan and organize the 1986 International Symposium on Information Theory held in Ann Arbor.