Image Coding Using Entropy-Constrained Residual Vector Quantization*

Faouzi Kossentini and Mark J. T. Smith
Digital Signal Processing Laboratory
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, GA 30332

Christopher F. Barnes
Georgia Tech Research Institute
Georgia Institute of Technology
Atlanta, GA 30332

April 15, 1993

Abstract

In this paper, the residual vector quantization (RVQ) structure is exploited to produce a variable length codeword RVQ. Necessary conditions for the optimality of this RVQ are presented, and a new entropy-constrained RVQ (EC-RVQ) design algorithm is shown to be very effective in designing RVQ codebooks over a wide range of bit rates and vector sizes. The new EC-RVQ has several important advantages. It can outperform entropy-constrained VQ (EC-VQ) in terms of peak signal-to-noise ratio (PSNR), memory, and computation requirements. It can also be used to design high rate codebooks and codebooks with relatively large vector sizes. Experimental results indicate that when the new EC-RVQ is applied to image coding, very high quality is achieved at relatively low bit rates.

*This work was supported in part by the National Science Foundation under contract MIP-9116113 and by NASA.

1 Introduction

Vector quantization (VQ) has received much attention and is a powerful and effective technique for image compression [9]. A motivation for this approach is that the performance of vector quantizers can approach the distortion-rate bound $D(R)$ as the vector size becomes sufficiently large [3]. However, the rate at which the performance of VQ approaches the bound $D(R)$ as a function of increasing vector size is rather slow [3]. Moreover, both the computation and memory requirements associated with VQ increase exponentially as the vector size increases. Therefore, relatively small vectors, typically of size 4×4, are usually used in the design of unconstrained exhaustive search VQ codebooks for image coding.

Reducing the large complexity and memory requirements of VQ has been the focus of much research. Various imposed structural constraints have been considered, but such constraints generally lead to reduced performance for a given rate and dimension. However, the reduction in complexity obtained is often a good trade for the moderate loss in quality. Some examples of structured vector quantizers are lattice VQ [7], hierarchical VQ [27], and tree-searched VQ (TSVQ) [4, 10]. Residual vector quantization (RVQ) or multistage VQ is one such structured vector quantizer whose structure reduces both the memory and computation costs, and is able to operate over a large range of bit rates and vector sizes. The recent interest in RVQ is due largely to its good complexity/performance tradeoffs, and to the recent advances made in design methodology, which have resulted in noticeable improvements over previous design methods [14, 26].

The structural constraints of RVQ result in a performance degradation compared to an unconstrained VQ with the same bit rate and vector size. This degradation can be attributed to two factors. First, the RVQ decoder is constrained by a direct-sum codebook structure where all possible output vectors of the RVQ are formed by

the sum of stage code vectors—this set is called the direct-sum codebook. Second, the encoder typically employs an efficient sequential stage-wise search procedure for practical reasons. However, entanglements in the RVQ tree tend to reduce encoding accuracy when fast searching is performed. This difficulty is obviated by exhaustive searching or other forms of optimal sequential searching (see [1]) but the price paid in computational complexity is generally enormous.

Looking beyond this, however, the structure of RVQ has properties that make it attractive. The multi-stage structure can be exploited to produce variable number-of-stages RVQ (one form of variable rate RVQ), which was shown in [19, 20] to lead to improvements in performance over *fixed rate* RVQ. In addition, the direct-sum structural constraint usually leads to an RVQ output entropy which is much smaller than the logarithm of the number of direct-sum code vectors. Experimental evidence suggests that the decrease in output entropy compensates for the increase in average distortion which, in turn, leads to a very competitive coding system.

A simple approach to constructing another form of variable rate RVQ is to combine a fixed rate RVQ with a noiseless coder. However, a better approach is to directly incorporate entropy coding in the design process. The joint optimization of a VQ and an entropy coder was shown to lead to a significant improvement in performance for the conventional VQ case [5, 6]. This motivates the investigation of an RVQ design algorithm that minimizes the average distortion subject to a constraint on the output entropy of the RVQ. This paper introduces a new entropy-constrained RVQ (EC-RVQ) design algorithm that is very effective in designing variable rate RVQ codebooks. EC-RVQ is shown to be capable of outperforming conventional EC-VQ in terms of computational complexity, memory requirements and coding quality, and has the ability to operate over a much wider range of bit rates and vector sizes.

To set the mathematical notation and terminology used throughout this paper, the next section begins with a brief summary of fixed rate RVQ. To lay the founda-

tion for the discussion of variable rate RVQ and the development of the new EC-RVQ design algorithm, necessary conditions for the optimality of fixed rate RVQ and corresponding design algorithms are also discussed in the section as well. Next, methods of constructing three forms of variable rate RVQs are discussed and compared in Section 3. Necessary conditions for the optimality of variable rate RVQ are presented, and a discussion of the new EC-RVQ algorithm is considered in Section 4. Section 5 discusses the performance of EC-RVQ when used in image coding applications. The paper concludes with some general comments on improving EC-RVQ performance that reflect work presently under study.

2 Fixed Rate RVQ

Residual vector quantization (RVQ) or multistage VQ consists of a cascade of VQ stages, each operating on the "residual" of the previous stage. A block diagram of a P-stage RVQ is given in Figure 1 for illustration. A general RVQ consisting of P stages (with N_i vectors in the ith stage) is capable of uniquely representing $N = \prod_{i=1}^{P} N_i$ vectors with only $\sum_{i=1}^{P} N_i$ code vectors required for storage. Thus, the RVQ achieves tremendous savings over unconstrained VQ in terms of memory requirements, and may also achieve similar savings in computations.

To establish the notation and review the key points for optimal fixed rate RVQ, let $\boldsymbol{x}_1$ be a realization of the random k-dimensional vector $\boldsymbol{X}_1$ described by the probability density function (pdf) $f_{\boldsymbol{X}_1}(\boldsymbol{x}_1)$ on $\Re^k$ and assume this to be the input to the P-stage RVQ shown in Figure 1. For the pth stage VQ with $1 \leq p \leq P$, let us define the following symbols:

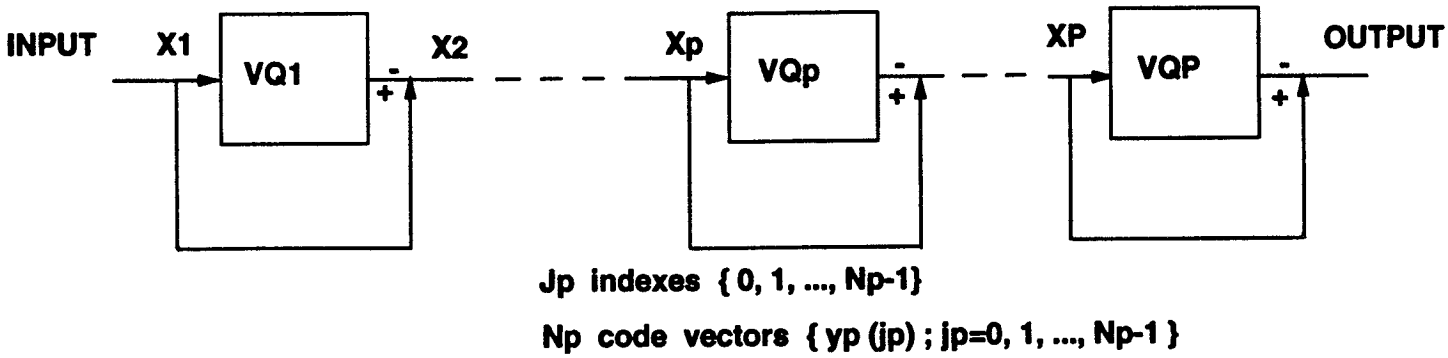

Figure 1: A P-stage residual vector quantizer

N_p	the pth stage codebook size (number of codebook vectors)
j_p	the pth stage index: $\{0 \leq j_p \leq N_p - 1\}$
J_p	the pth set of all possible values for j_p: i.e. $\{0, 1, 2, \ldots, N_p - 1\}$
$\boldsymbol{y}_p(j_p)$	the j_pth code vector
$S_p(j_p)$	the j_pth partition cell
$V_p(j_p)$	the j_pth conditional-stage residual cell
$\mathcal{C}_p$	the pth stage codebook $\{\boldsymbol{y}_p(j_p) : j_p \in J_p\}$
$\mathcal{P}_p$	the pth stage partition $\{S_p(j_p) : j_p \in J_p\}$
Q_p	the pth stage quantizer mapping

Associated with a P-stage RVQ is an equivalent single-stage direct-sum VQ. The direct-sum VQ and RVQ are identical in the sense that they produce the same representation of the source output and they have the same expected distortion. For the direct-sum VQ, let us define the following symbols:

N	direct-sum codebook size ($N = \prod_{i=1}^{P} N_i$)
$\boldsymbol{J}$	direct-sum P-tuple index set $J_1 \times J_2 \times \cdots \times J_P$
$\boldsymbol{j}$	a P-tuple index in $\boldsymbol{J}$
$\boldsymbol{y}(\boldsymbol{j})$	$\boldsymbol{j}$th direct-sum code vector
$\boldsymbol{V}(\boldsymbol{j})$	$\boldsymbol{j}$th direct-sum partition cell
$\boldsymbol{C}$	direct-sum codebook $\{\boldsymbol{y}(\boldsymbol{j}) : \boldsymbol{j} \in \boldsymbol{J}\}$
$\boldsymbol{P}$	direct-sum partition $\{\boldsymbol{V}(\boldsymbol{j}) : \boldsymbol{j} \in \boldsymbol{J}\}$
$\boldsymbol{Q}$	direct-sum mapping $\boldsymbol{Q}(\boldsymbol{x}_1) = \sum_{p=1}^{P} Q_p(\boldsymbol{x}_p)$

The direct-sum codebook contains all possible ordered sums of the stage code vectors, i.e., $C = C_1 \oplus C_2 \oplus \ldots \oplus C_P$. The direct-sum code vectors are given by $\boldsymbol{y}(\boldsymbol{j}) = \sum_{p=1}^{P} \boldsymbol{y}_p(j_p)$, where j_p is the pth member of the ordered P-tuple index $\boldsymbol{j}$. The direct-sum VQ quantizes the source vector $\boldsymbol{x}_1$ and outputs the representation $\hat{\boldsymbol{x}}_1 = \boldsymbol{Q}(\boldsymbol{x}_1)$ given by $\boldsymbol{Q}(\boldsymbol{x}_1) = \sum_{p=1}^{P} Q_p(\boldsymbol{x}_p)$, where we call $\boldsymbol{x}_p = \boldsymbol{x}_1 - \sum_{i=1}^{p-1} Q_i(\boldsymbol{x}_i)$ the pth stage *causal residual.* The term *causal* refers to the stages supporting the computation of the residual; i.e., the stage residuals are computed sequentially starting from the first stage to the pth stage.

To formalize this notion, let the distortion that results from representing the input $\boldsymbol{x}_1$ by the quantized output $\hat{\boldsymbol{x}}_1$ be expressed by $d(\boldsymbol{x}_1, \hat{\boldsymbol{x}}_1)$. The distortion measure $d(\boldsymbol{x}, \boldsymbol{y})$ is assumed to be a non-negative real-valued function that satisfies the following requirements:

1. For any fixed $\boldsymbol{x} \in \Re^k$, $d(\boldsymbol{x}, \boldsymbol{y})$ is a continuously differentiable function of $\boldsymbol{y} \in \Re^k$.

2. $d(\boldsymbol{x}, \boldsymbol{y})$ is translationally invariant.

3. For any fixed $\boldsymbol{x} \in \Re^k$, $d(\boldsymbol{x}, \boldsymbol{y})$ is a strictly convex function of $\boldsymbol{y}$, that is, $\forall \boldsymbol{y}_1, \boldsymbol{y}_2 \in \Re^k \text{and} \lambda \in (0,1), d(\boldsymbol{x}, \lambda \boldsymbol{y}_1 + (1-\lambda)\boldsymbol{y}_2) < \lambda d(\boldsymbol{x}, \boldsymbol{y}_1) + (1-\lambda) d(\boldsymbol{x}, \boldsymbol{y}_2)$.

A P-stage RVQ is said to be *optimal* if it gives at least a locally minimum value of the average distortion. There are two necessary conditions for the optimality of fixed rate RVQ [1, 2]. First, the encoder must map the input vectors according to the following nearest-neighbor rule:

$$\boldsymbol{x}_1 \in \boldsymbol{V}^*(\boldsymbol{j}) \quad \text{if and only if} \quad d(\boldsymbol{x}_1, \boldsymbol{y}(\boldsymbol{j})) \leq d(\boldsymbol{x}_1, \boldsymbol{y}(\boldsymbol{k})) \quad \text{for all} \quad \boldsymbol{k} \in \boldsymbol{J}. \tag{1}$$

Second, the stage code vectors $\boldsymbol{y}_p(j_p)$ at the pth stage must satisfy [2, 23]

$$\int d\left(\gamma_p, \boldsymbol{y}_p^*(j_p)\right) f_{\Gamma_p|j_p}(\gamma_p) d\gamma_p = \inf_{\boldsymbol{u} \in \Re^k} \int d(\gamma_p, \boldsymbol{u}) f_{\Gamma_p|j_p}(\gamma_p) d\gamma_p < \infty \tag{2}$$

where $\gamma_p = \boldsymbol{x}_1 - \sum_{\substack{i=1 \\ i \neq p}}^{P} \boldsymbol{y}_i(j_i)$ is a realization of the conditional-stage residual random vector $\boldsymbol{\Gamma}_p$, and the pdf $f_{\boldsymbol{\Gamma}_p|j_p}(\gamma_p)$ is related to the source pdf $f_{\boldsymbol{X}_1}(\cdot)$ according to

$$f_{\boldsymbol{\Gamma}_p|j_p}(\gamma_p) = \frac{\sum_{\boldsymbol{j} \in H_p(j_p)} \boldsymbol{I}\left(\boldsymbol{V}(\boldsymbol{j})\right) f_{\boldsymbol{X}_1}\left(\boldsymbol{g}(\beta_p(\boldsymbol{j})) + \gamma_p\right)}{\mathrm{pr}\left(\gamma_p \in V_p(j_p)\right)}, \tag{3}$$

where $\beta_p(\boldsymbol{j}) = (j_1, j_2, \ldots, j_{p-1}, j_{p+1}, \ldots, j_P)$, $\boldsymbol{g}(\beta_p(\boldsymbol{j})) = \sum_{\substack{i=1 \\ i \neq p}}^{P} \boldsymbol{y}_i(j_i)$, $H_p(j_p) \subset \boldsymbol{J}$ is the set of all indices $\boldsymbol{j} = (k_1, k_2, \ldots, k_{p-1}, j_p, k_{p+1}, \ldots, k_P)$ such that $j_p \in J_p$, and $\boldsymbol{I}[\boldsymbol{V}(\boldsymbol{j})]$ is an indicator function for the direct-sum partition cell $\boldsymbol{V}(\boldsymbol{j})$, that is, $\boldsymbol{I}[\boldsymbol{V}(\boldsymbol{j})] = 1$ if $\boldsymbol{x}_1 \in \boldsymbol{V}(\boldsymbol{j})$ and $\boldsymbol{I}[\boldsymbol{V}(\boldsymbol{j})] = 0$ otherwise. The $\boldsymbol{y}_p(j_p)$'s which satisfy equation (2) are generalized centroids of conditional-stage residual vectors (i.e., residual vectors formed from the encodings of all *prior* and *subsequent* RVQ stages). Hence, the second condition will be referred to as the conditional-stage residual centroid condition hereafter. A mathematical derivation of these two conditions is given in [2, 23].

2.1 The Fixed Rate RVQ Design Algorithm

The fixed rate RVQ design algorithm, introduced in [1], attempts to optimize all stage codebooks jointly to minimize the reconstruction error over all training data subject to a constraint on the number of direct-sum code vectors. Assuming that all stage codebooks are held fixed, optimization of the encoder implies that each training set vector is mapped to its closest direct-sum code vector using the nearest-neighbor rule (1). In general, this can be accomplished by exhaustively searching the direct sum codebook. However, this technique typically carries sufficient computational overhead to be unattractive. An alternative approach is to *sequentially* search the RVQ stage codebooks. This technique results in an increase in speed, but unfortunately leads to a significant degradation in performance since optimal code vector selection in the direct-sum codebook is no longer guaranteed. To address this issue, the M-search technique was explored and was shown to be very efficient when used to search the

RVQ tree [1, 18]. Small improvements can be obtained by simply using M-search when encoding the input using a sequentially-designed RVQ codebook. However, better results can be obtained by directly incorporating the M-search in the RVQ design as well as in the encoder [2, 18]. An additional gain can be achieved for the same complexity by allowing the value of M to be larger in some stages of the RVQ and smaller in others. This can be done by first defining a desired level for the average number of M-search computations. Using a large training set, the best value of M for each stage can be determined empirically such that the total number of M-search computations is within the pre-specified tolerance.

Given a fixed direct-sum partition, the fixed rate RVQ design method used in this work is simply an iterative Gauss-Seidel algorithm that jointly optimizes the stage codebooks by successively operating on each RVQ stage while holding fixed all other stage codebooks. At each stage optimization step, code vectors are found that simultaneously satisfy the conditional-stage residual centroid condition (2). Assuming that the squared error distortion measure is used, each "decoder-only" iteration will update the stage codebooks such that the average distortion will either be reduced or left unchanged [24]. Using theorems in [11], it can be shown [24] that if the encoder yields a Voronoi partition with respect to the direct-sum codebook, then the fixed rate RVQ design algorithm converges monotonically to a fixed point which satisfies the necessary conditions (1) and (2) for minimum squared error distortion.

This proven convergence behavior is based on an exhaustive search encoder, which is not realistic for a practical system in general. For practical applications, a sequential nearest neighbor or an M-search encoder is used. In these cases, the encoder optimization step may actually increase the average distortion and monotonic convergence cannot be guaranteed. However, experimental results have shown that the sequential-search RVQ design algorithm effectively reduces the average distortion with only occasional deviations from monotonicity. Furthermore, in all our experiments,

the M-search RVQ design algorithm converged monotonically to a local minimum, even when relatively small values of M (such as 2 or 3) were used.

2.2 Comments on RVQ Performance

An upper bound on the performance of fixed rate RVQ is the performance of exhaustive-search VQ [26]. For the same bit rate and vector size (i.e., same number of code vectors), the average distortion introduced by the RVQ can be shown to be generally larger than that introduced by an unconstrained VQ. For example, let's assume that a conventional VQ and an RVQ have the same fixed partition of $\Re^k$. It is shown in [11] that the average distortion can be minimized if and only if the code vectors are selected as the centroids of their respective partition cells. Since the code vectors in the conventional VQ codebook are structurally independent, this selection can be done separately for each partition cell. However, code vectors formed by direct-sums of stage code vectors are structurally dependent and hence it is unlikely *all* will be centroids of their respective direct-sum partition cells. As a matter of fact, these direct-sum code vectors are not guaranteed to even lie within their respective cells. Therefore, the average distortion of RVQ is higher than that of conventional VQ.

However, by using large vector sizes and multi-path searching, the RVQ performance is shown to exceed that of conventional VQ with only a fraction of the computation and memory requirements [18]. Moreover, the direct-sum codebook constraint usually leads to an output entropy H that is smaller than that of unconstrained VQ. This can be easily demonstrated using the fact that the joint entropy of a collection of sources (or random variables) is less than or equal to the sum of the entropies of the individual sources [8]. That is, given P random variables $X_1, \ldots, X_P$,

$$H(X_1, \ldots, X_P) \leq \sum_{p=1}^{P} H(X_p).$$

Given the set of P-tuple indices $\boldsymbol{J}$, one can uniquely index all the code vectors in an unconstrained VQ codebook (which has the same number of code vectors as the

direct-sum RVQ codebook) by the mapping $\gamma : J_1 \times \ldots \times J_P \mapsto \boldsymbol{J}$, where

$$\gamma(j_1, \ldots, j_P) = \sum_{p=1}^{P} j_p \prod_{k=0}^{p-1} |J_k|$$

where $|J_0| = 1$ and $|J_k|$ is the size of the set J_k. Since the $j_1, \ldots, j_P$ are independent (they are chosen arbitrarily), the output entropy of the unconstrained VQ is

$$H(\boldsymbol{J}) = \sum_{p=1}^{P} H(J_p),$$

where $H(J_p)$ is the entropy of J_p. One can also use the same indexing scheme to index the direct-sum codebook, except that now j_p denotes the index for the pth stage of the RVQ. As noted earlier, the RVQ stages are related by the direct-sum structure, and $j_1, \ldots, j_P$ are not independent. Thus, $H_{RVQ}(\boldsymbol{J}) = H(J_1, \ldots, J_P) \leq \sum_{p=1}^{P} H(J_p) = H_{VQ}(\boldsymbol{J})$. Notice that this result can also be obtained by using the fact that the entropy of the collection of the random variables $J_1, J_2, \ldots, J_P$ is equal to the sum of the conditional entropies, i.e.,

$$H(J_1, J_2, \ldots, J_P) = \sum_{p=1}^{P} H(J_p | J_{p-1}, \ldots, J_1).$$

The previous results suggest that the direct-sum codebook constraints can generally be expected to lead to both an increased average distortion and a decreased output entropy. This implies that for a given average bit rate, variable rate RVQ could conceivably have the potential to be competitive with variable rate VQ.

3 Variable Rate RVQ

For a given vector size k, variable rate VQ implementations are those that, if properly designed, can operate at bit rates close to the ones given by the kth order rate-distortion curve $R_k(D)$ of the input. There are several ways in which a variable rate RVQ can be constructed. As reported in [19], a variable rate implementation can be

achieved by exploiting the inherent multi-stage structure of RVQ. Since each stage contributes independently to the total bit rate, variable rate coding can be achieved easily by truncating the number of RVQ stages used for a given source vector. For each input vector, the encoding terminates once the distortion falls below a prescribed threshold. Clearly, the encoder and the decoder must both have knowledge of the number of stages (bit rate) used to encode a given vector. Sending such a rate to the decoder is usually done by sending side information, which can be very costly. However, when relatively large vector sizes (such as 8×8 or 16×16) are used, side information requires only a small fraction of the total bit rate [19]. This variable rate technique has two advantages: 1) Incorporating such a technique into the RVQ design algorithm leads to reduced encoding complexity because fewer distortion calculations are needed to encode vectors with low variances, and the centroid computation requires fewer additions; and 2) variable rate RVQ of this type tends to allow for a better match to the statistics of images. A large number of bits can be used to encode edge vectors while a small number can be used to encode low variance vectors [19].

Another approach to variable rate RVQ is to entropy code the RVQ output indices. In this case, a fixed rate RVQ is combined with a variable rate lossless coder (such as a Huffman coder). This can be done by considering the RVQ direct-sum code vectors to be symbols in an extended source alphabet and constructing a variable length lossless code for them. The complicated interdependencies among the stages of an RVQ often results in a direct-sum codebook where the code vectors have a very nonuniform probability distribution. Therefore, the output entropy of RVQ is usually much smaller than the logarithm of the number of direct-sum code vectors. Experimental results, reported in [20], show that the output entropy of the direct-sum codebook is much smaller than that of the unconstrained VQ codebook (for the same number of code vectors). Thus the RVQ/entropy coder combination can lead to a

substantially lower average bit rate while maintaining the same performance level of a fixed rate RVQ.

A superior approach to the variable rate RVQ implementation described above is one in which all code vectors and codewords are optimized with respect to each other. Therefore, the natural design problem for entropy-based RVQ is to find a direct-sum codebook whose vectors minimize the average reconstruction error over all training set data subject to a constraint on the output entropy of the RVQ. In the next section, necessary conditions for the optimality of variable rate RVQ are presented, an *entropy constrained* RVQ (EC-RVQ) design algorithm which satisfies these conditions is introduced, and the performance of this algorithm is demonstrated and discussed.

4 Entropy-Constrained RVQ

The high level structure of the EC-RVQ is illustrated in Figure 2. It consists of a P-stage RVQ where the stage codewords are input to a mapping operator. The mapping operator transforms the direct-sum index $\mathbf{j} = (j_1, j_2, \ldots, j_P)$ codeword into a variable length codeword $\mathbf{c}(\mathbf{j})$ that is then used as the representation of the compressed data. The mapping operator can be an entropy coder or a collection of stage entropy coders. The idea underlying the entropy mapping operation is that $\boldsymbol{j}$'s that occur very often are represented with short codewords and $\boldsymbol{j}$'s that occur infrequently are represented with longer codewords such that the average bit rate is reduced.

4.1 Necessary Conditions for Optimal Variable Rate RVQ

For the direct-sum VQ, let $\mathcal{J}$ be the set of variable length indices $\{c(\boldsymbol{j}), \boldsymbol{j} \in \boldsymbol{J}\}$. The direct-sum VQ, $\boldsymbol{Q} : \Re^k \mapsto \boldsymbol{C}$, quantizes the source vector $\boldsymbol{x}_1$ and outputs $\boldsymbol{Q}(\boldsymbol{x}_1)$, and may be realized by a composition of a variable length encoder mapping $\mathcal{E} : \Re^k \mapsto \mathcal{J}$

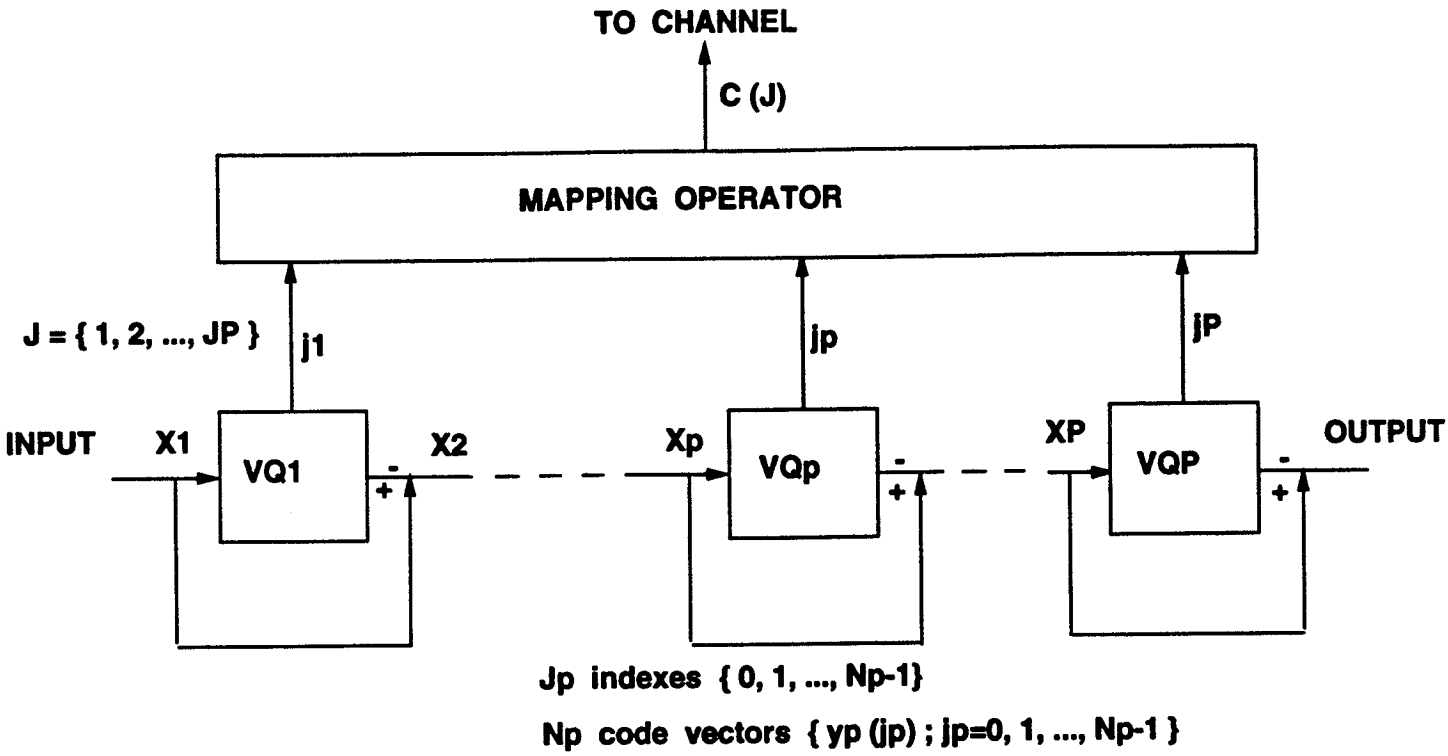

Figure 2: The EC-RVQ Structure

where

$$\mathcal{E}(\boldsymbol{x}_1) = c(j) \text{ if and only if } \boldsymbol{x}_1 \in V(j),$$

and a variable length decoder mapping $\mathcal{D} : \mathcal{J} \mapsto \boldsymbol{C}$ where

$$\mathcal{D}(c(j)) = \boldsymbol{y}(j).$$

The variable length encoder can be further decomposed into two mappings, $\mathcal{E} = \mathbf{L} \circ \mathbf{E}$, where $\mathbf{E} : \Re^k \mapsto \boldsymbol{J}$ and $\mathbf{L} : \boldsymbol{J} \mapsto \mathcal{J}$, and $\circ$ denotes composition. Similarly, one can decompose the variable length decoder into two mappings, $\mathcal{D} = \mathbf{D} \circ \mathbf{L}^{-1}$, where $\mathbf{L}^{-1} : \mathcal{J} \mapsto \boldsymbol{J}$, and $\mathbf{D} : \boldsymbol{J} \mapsto \boldsymbol{C}$. Note that the mapping $\mathbf{L}$ is an invertible mapping with inverse $\mathbf{L}^{-1}$.

Let $\boldsymbol{x}_1$ be a realization of the random k-dimensional vector $\boldsymbol{X}_1$ described by the probability density function (pdf) $f_{\boldsymbol{X}_1}(\boldsymbol{x}_1)$ on $\Re^k$. Also, let the distortion that results from representing $\boldsymbol{x}_1$ with $\hat{\boldsymbol{x}}_1$ be expressed by $d(\boldsymbol{x}_1, \hat{\boldsymbol{x}}_1)$. The distortion measure

$d(\boldsymbol{x}, \boldsymbol{y})$ is assumed to be a non-negative real valued function that satisfies the requirements (1)-(3) in Section 2. A variable rate P-stage RVQ (with an average rate $\leq R$) is said to be optimal for $f_{\boldsymbol{X}_1}(.)$ if it gives a locally or globally minimum value of the average distortion. The design problem can be stated as follows: Choose the codebook $\boldsymbol{C}$, partition $\boldsymbol{P}$ and mapping $\mathbf{L}$ that minimize the Lagrangian

$$J_\lambda(\mathbf{E}, \mathbf{L}, \mathbf{D}) = E\left\{d(\boldsymbol{x}_1, \hat{\boldsymbol{x}}_1) + \lambda\, |\mathbf{L}(\boldsymbol{j})|\right\} \tag{4}$$

where λ is the Lagrange multiplier and $|\mathbf{L}(\boldsymbol{j})|$ denotes the length of $\mathbf{L}(\boldsymbol{j})$.

There are three necessary conditions for the optimality of variable rate RVQ [23]. First, the encoder must map the input vectors according to the following nearest-neighbor encoding rule:

$$\boldsymbol{x}_1 \in \boldsymbol{V}^*(\boldsymbol{j}) \;\; \textit{iff} \;\; d(\boldsymbol{x}_1, \boldsymbol{y}(\boldsymbol{j})) + \lambda\, |\mathbf{L}(\boldsymbol{j})| \;\leq\; d(\boldsymbol{x}_1, \boldsymbol{y}(\boldsymbol{k})) + \lambda\, |\mathbf{L}(\boldsymbol{k})| \;\; \text{for all } \boldsymbol{k} \in \boldsymbol{J}. \tag{5}$$

Second, the mapping $\mathbf{L}$ must be one that minimizes the expected codeword length, $R = \sum_{\boldsymbol{j} \in \boldsymbol{J}} |\mathbf{L}(\boldsymbol{j})|\, \mathrm{pr}(\boldsymbol{j})$, where $\mathrm{pr}(\boldsymbol{j}) = \mathrm{pr}(\boldsymbol{x}_1 \in \boldsymbol{V}(\boldsymbol{j}))$. Setting the codeword length $|\mathbf{L}(\boldsymbol{j})|$ to

$$|\mathbf{L}^*(\boldsymbol{j})| = -\log_2 \mathrm{pr}(\boldsymbol{j}) = -\log_2 \mathrm{pr}(j_1, j_2, \ldots, j_P) \tag{6}$$

results in an average rate which is equal to the output entropy of the direct sum RVQ. Third, the stage code vectors $\boldsymbol{y}_p(j_p)$ at the pth stage must satisfy the conditional-stage residual centroid condition (2). A complete derivation of these conditions is involved and may be found in [23].

The probability $\mathrm{pr}(j_1, j_2, \ldots, j_P)$ of a path in the RVQ can also be written as the product of conditional probabilities, i.e.

$$\mathrm{pr}(j_1, j_2, \ldots, j_P) = \mathrm{pr}(j_P | j_{P-1}, \ldots, j_1)\, \mathrm{pr}(j_{P-1} | j_{P-2}, \cdots, j_1) \cdots\, \mathrm{pr}(j_2 | j_1)\, \mathrm{pr}(j_1)$$

Therefore,

$$\begin{aligned} |\mathbf{L}^*(\boldsymbol{j})| \;\; = \;\; & -\log_2 \mathrm{pr}(j_P | j_{P-1}, \ldots, j_1) - \log_2 \mathrm{pr}(j_{P-1} | j_{P-2}, \ldots, j_1) \\ & -\ldots - \log_2 \mathrm{pr}(j_2 | j_1) - \log_2 \mathrm{pr}(j_1) \end{aligned}$$

and

$$H^*(J_1, J_2, \ldots, J_P) = \sum_{p=1}^{P} H(J_p | J_{p-1}, \ldots, J_1).$$

4.2 The EC-RVQ Design Algorithm

The EC-RVQ design algorithm is an iterative descent algorithm similar to the one used for the design of EC-VQ codebooks. Each iteration consists of applying the transformation

$$(\mathbf{E}(t+1), \mathbf{L}(t+1), \mathbf{D}(t+1)) = T(\mathbf{E}(t), \mathbf{L}(t), \mathbf{D}(t))$$

where

$$\begin{aligned}
\mathbf{E}(t+1) &= \arg\min_{\mathbf{E}}(\mathbf{E}, \mathbf{L}(t), \mathbf{D}(t)) && \text{(optimum partitions)} \\
\mathbf{L}(t+1) &= \arg\min_{\mathbf{L}}(\mathbf{E}(t+1), \mathbf{L}, \mathbf{D}(t)) && \text{(optimum codeword lengths)} \\
\mathbf{D}(t+1) &= \arg\min_{\mathbf{D}}(\mathbf{E}(t+1), \mathbf{L}(t+1), \mathbf{D}) && \text{(optimum code vectors)}
\end{aligned}$$

Following the lines of argument of [5], one can show [24] that every limit point of the sequence $(\mathbf{E}(t), \mathbf{L}(t), \mathbf{D}(t))$, $t = 0, 1, \ldots$, generated by the transformation T minimizes the Lagrangian $J_\lambda(\mathbf{E}, \mathbf{L}, \mathbf{D})$ (as given by (4)). Therefore, the EC-RVQ design algorithm is guaranteed to converge to a local minimum.

To find the entire convex hull of the operational rate-distortion curve, the minimization of $J_\lambda(\mathbf{E}, \mathbf{L}, \mathbf{D})$ is repeated for various λ's. Starting with $\lambda = 0$ (which corresponds to the RVQ codebook designed by the fixed rate RVQ design algorithm), the EC-RVQ design algorithm uses a pre-determined sequence of λ's [5] to design variable rate EC-RVQ codebooks. A summary of the algorithm is given in Figure 3.

As in the design of fixed rate RVQ codebooks, multipath searching is used in the encoder optimization step of the EC-RVQ design algorithm to closely satisfy the encoding rule given by (5). The M-search algorithm is found to be very efficient in substantially reducing the encoding complexity of EC-RVQ for only a small loss in

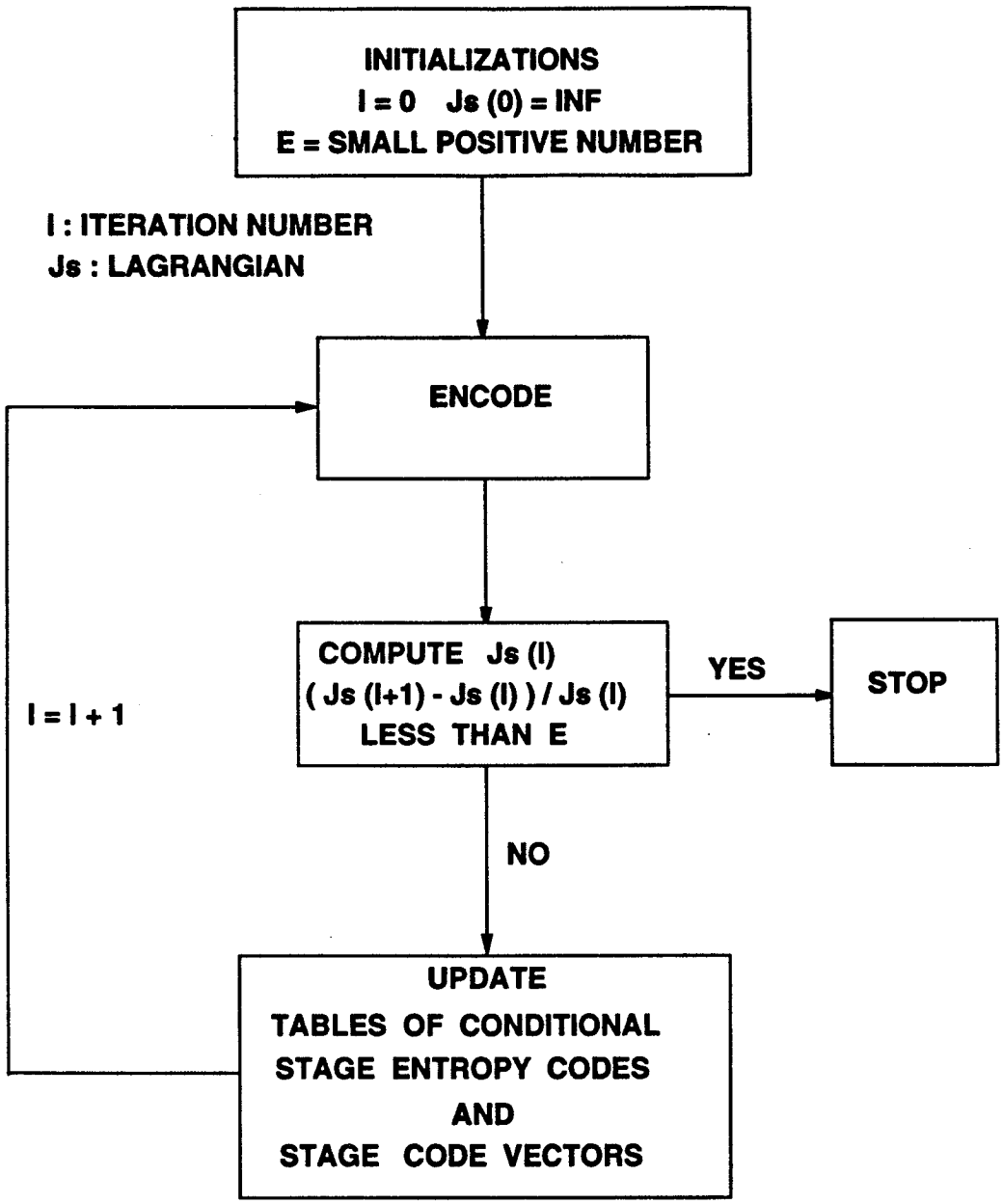

Figure 3: The EC-RVQ design algorithm

performance. Also, the Gauss-Seidel algorithm is used to find optimal stage code vectors (i.e., stage code vectors that simultaneously satisfy the conditional-stage residual centroid condition (2)).

Unique to EC-RVQ is optimization of the lengths of the codewords which represent direct-sum partition cells or code vectors. Allowing the use of non-integer codeword lengths, the *self-information* of a P-tuple index (or random variable) $\boldsymbol{j} = (j_1, j_2, \ldots, j_P)$, given by (6), is essentially the optimal length of the variable length codeword associated with that index $\boldsymbol{j}$. Equation (7) shows that such an optimal length is also the *sum* of P stage conditional self-information components. Because of the dependencies that usually exist between the stages of the RVQ, observations of past encoding decisions provides some partial information about the pth stage index j_p. While the estimation problem is difficult, one can still find a good estimate of the lengths of variable-length stage codewords by using a sufficiently large training set.

It is evident that the aggregate number of tables of conditional-stage entropy codes can become extremely large as the number of stages increases, and consequently the storage requirements for the entropy tables may very well offset the memory savings obtained by using RVQ, especially when the bit rate and/or the vector size is large. For example, consider the design of EC-RVQ codebooks where each direct-sum codebook contains 10 stages with 4 code vectors/stage. Surprisingly, more than 4 million ($4 + 4^2 + \ldots + 4^{10}$) scalar memory locations are needed to store the tables of conditional-stage entropy codes (for each direct-sum codebook). However, the number of tables can be made very small by limiting the number of previous stages upon which the conditioning is based. This can be accomplished by making a Markov-like assumption and using conditional probabilities which depend only on the last m ($m < p - 1$) stages. In other words, the direct-sum codeword length $|\mathbf{L}(\boldsymbol{j})|$ is approximated by

$$|\mathbf{L}(\boldsymbol{j})|_m = -\log_2 \mathrm{pr}(j_P|j_{P-1},\ldots,j_{P-m}) - \log_2 \mathrm{pr}(j_{P-1}|j_{P-2},\ldots,j_{P-m}) - \ldots - \log_2 \mathrm{pr}(j_2|j_1) - \log_2 \mathrm{pr}(j_1) \quad (7)$$

Obviously, since $H(J_p|J_{p-1},\ldots,J_1) \leq H(J_p|J_{p-1},\ldots,J_{p-m})$ for each $p = 1,2,\ldots,P$ and $m < p-1$, it is easy to show that $H_m(\boldsymbol{J}) = \sum_{p=1}^{P} H(J_p|J_{p-1},\ldots,J_{p-m}) \geq H(\boldsymbol{J})$. When m is small, the memory requirements to store these tables are relatively small, but (of course) the performance of the associated EC-RVQ is also not as good as that of the $(P-1)$th order ($m = P-1$) EC-RVQ.

It is of particular interest to find the performance gain as a function of m, which will help us assess how large a value of m is needed such that satisfactory performance is obtained. Such performance gain (as a function of m) can be estimated empirically. Figure 4 shows that the performance gain obtained when m is increased, ascends rapidly to the optimal and often saturates for very small values of m. Using small values for m has the advantage that the memory requirements can be substantially reduced.

5 Performance of EC-RVQ

In this section, experimental results are used to compare the performance and complexity of EC-RVQ with those of EC-VQ over a wide range of bit rates and vector sizes. The training set consists of six (512 × 512, 8-bit) monochrome images taken from the USC database. Shifts and rotations are used to generate additional training vectors, leading to more than 200,000 4 × 4 vectors and more than 500,000 8 × 8 vectors. The image Lena, shown in Figure 5, is used for testing, and was not included in the training set. In all experiments, the objective performance measure used is the peak signal-to-quantization noise ratio (PSNR) defined by

$$\mathrm{PSNR} = -10\log_{10} \frac{\sum_{i=1}^{N}\sum_{i=1}^{N}(x(i,j) - \hat{x}(i,j))^2}{(N)^2(255)^2}$$

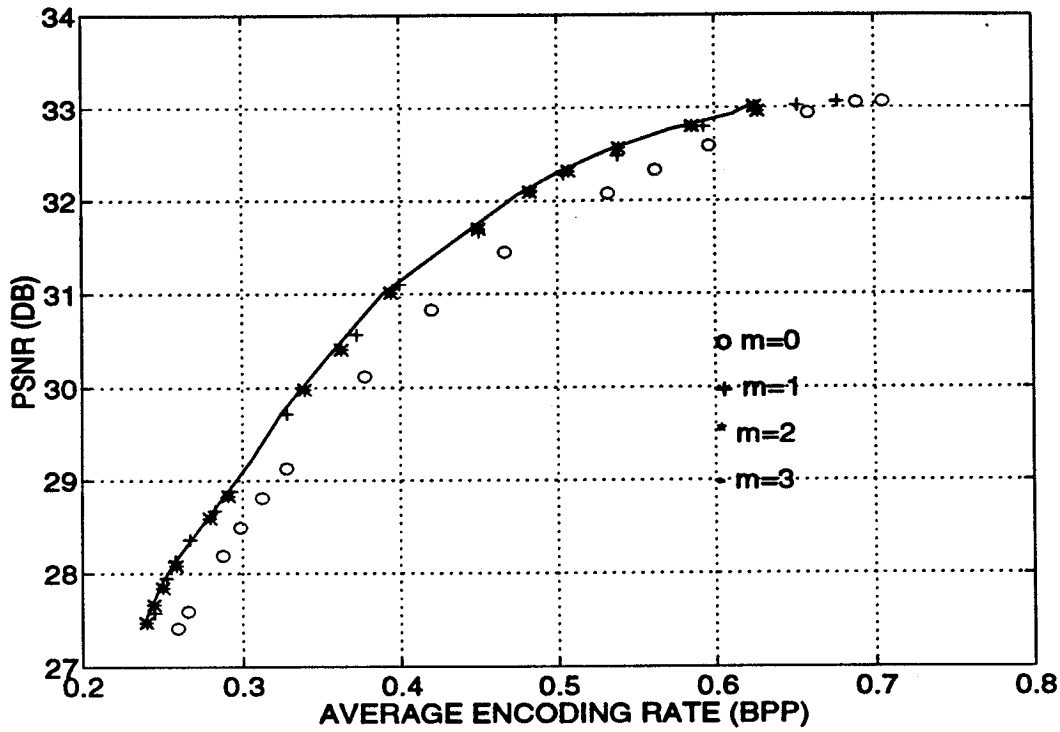

Figure 4: The rate-distortion performance of EC-RVQ (with 4 stages and 16 4×4 vectors/stage) for the test image Lena at increasing values of m.

where $N \times N$ is the size of the image (assumed to be squared) and $x(i,j)$ and $\hat{x}(i,j)$ represent the original and coded values (respectively) of the pixel at the ith row and the jth column of the image.

EC-RVQ systems based on 4×4 vectors were investigated first where the EC-RVQ design algorithm with $M = 4$ and $m = 1$ was used to design a sequence of variable rate RVQ codebooks. Each codebook contained 4 stage codebooks of size 16, leading to a *peak* encoding rate of 1.0 bit per pixel (bpp). Likewise, the conventional EC-VQ algorithm was used to design a sequence of codebooks of size $2^{12} = 4096$. Although a moderate peak bit rate (i.e., 0.75 bpp) was used in the design of EC-VQ codebooks, the design process required well over two months of CPU time on a Sun 4 Sparc station. Figure 6 compares the distortion *versus* rate performance on Lena, while Figure 7 compares the encoding complexity and the memory requirements for the EC-VQ and EC-RVQ. Table 1 shows PSNR comparisons of EC-RVQ and EC-VQ

Figure 5: The original image Lena at 8 bits/per pixel

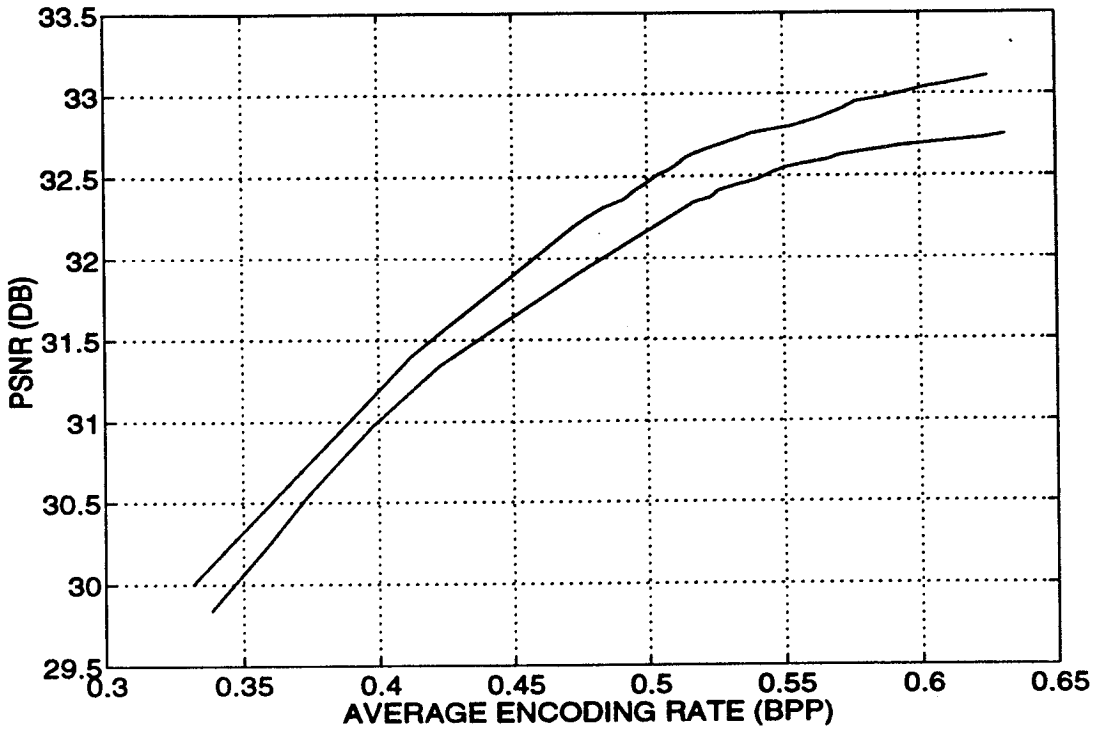

Figure 6: The rate-distortion performance of EC-RVQ (top) and EC-VQ (bottom) for the test image Lena. The vector size is 4×4.

for four images (taken from the USC database) all at an output bit rate of 0.40 bits/per pixel. EC-RVQ clearly outperforms EC-VQ in PSNR performance, encoding complexity and memory requirements. An important factor influencing the gain of EC-RVQ over EC-VQ is the very large number of direct-sum code vectors that EC-RVQ makes available, even while maintaining a low average encoding rate. The EC-VQ has a very limited codebook size (due to storage and search constraints), and the size constraint is not inactive as the theory requires.

In the second set of experiments, 4×4 vectors were used in the design of variable rate EC-RVQ codebooks with $M = 4$ and $m = 2$. The EC-RVQ codebooks contained 7 stage codebooks each with 16 code vectors, leading to a peak encoding rate of 1.75 bpp. Figure 8 show the PSNR performance for the EC-RVQ (at two different peak bit rates) for the test image Lena. As expected, EC-RVQ performance improves with increased peak bit rate, in spite of maintaining the same average output bit rate. It

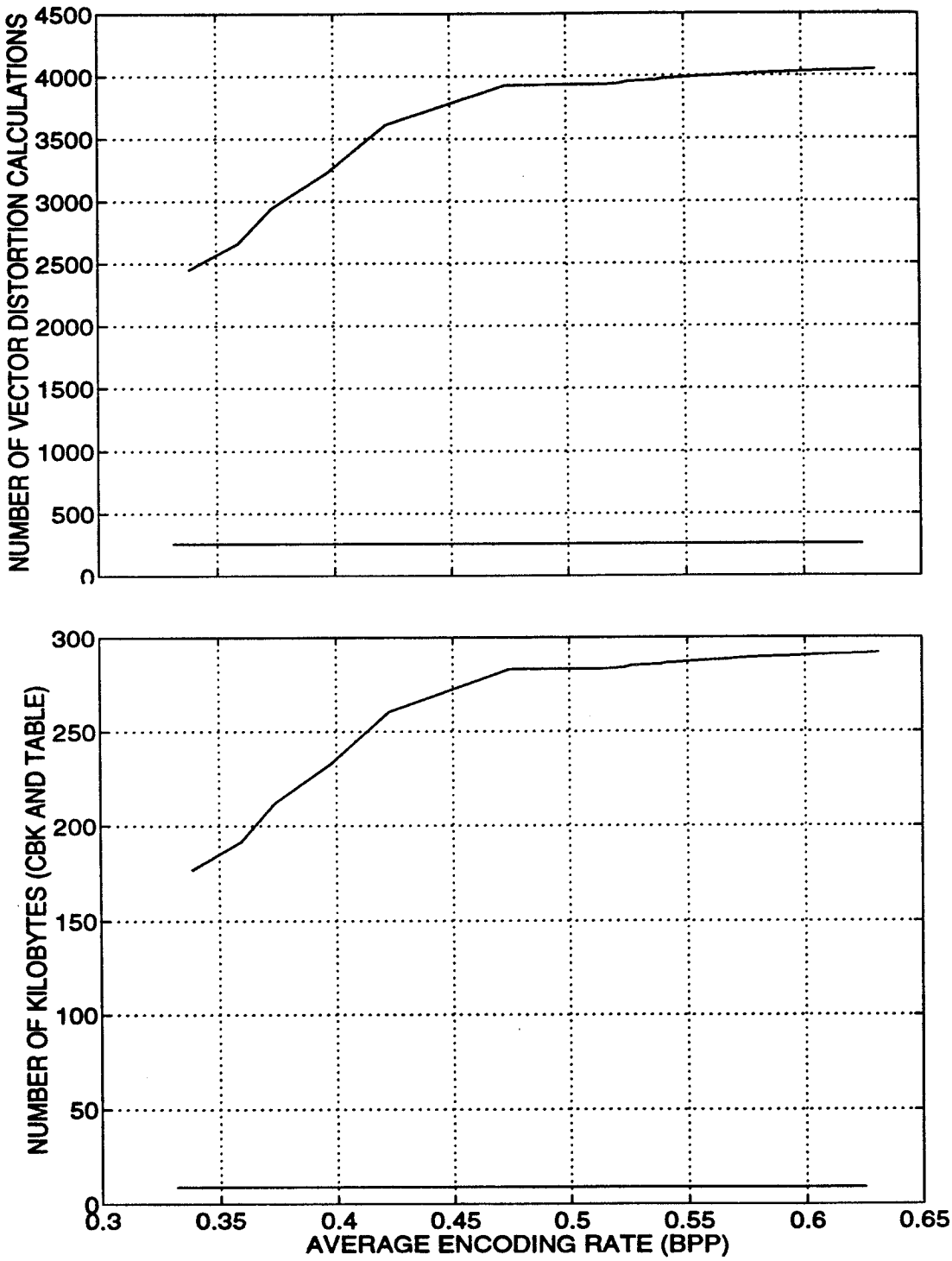

Figure 7: The encoding complexity (top figure) and memory requirements (bottom figure) of EC-VQ (top) and EC-RVQ (bottom) for the test image Lena.

	EC-VQ, peak=0.75 bpp	EC-RVQ, peak=1.00 bpp
Lena	30.97	31.27
Boat	29.63	30.21
Peppers	31.03	31.32
Tiffany	30.08	30.29

Table 1: PSNR of EC-RVQ and EC-VQ for four images taken from the USC database. The bit rate is 0.40 bpp. The vector size is 4×4.

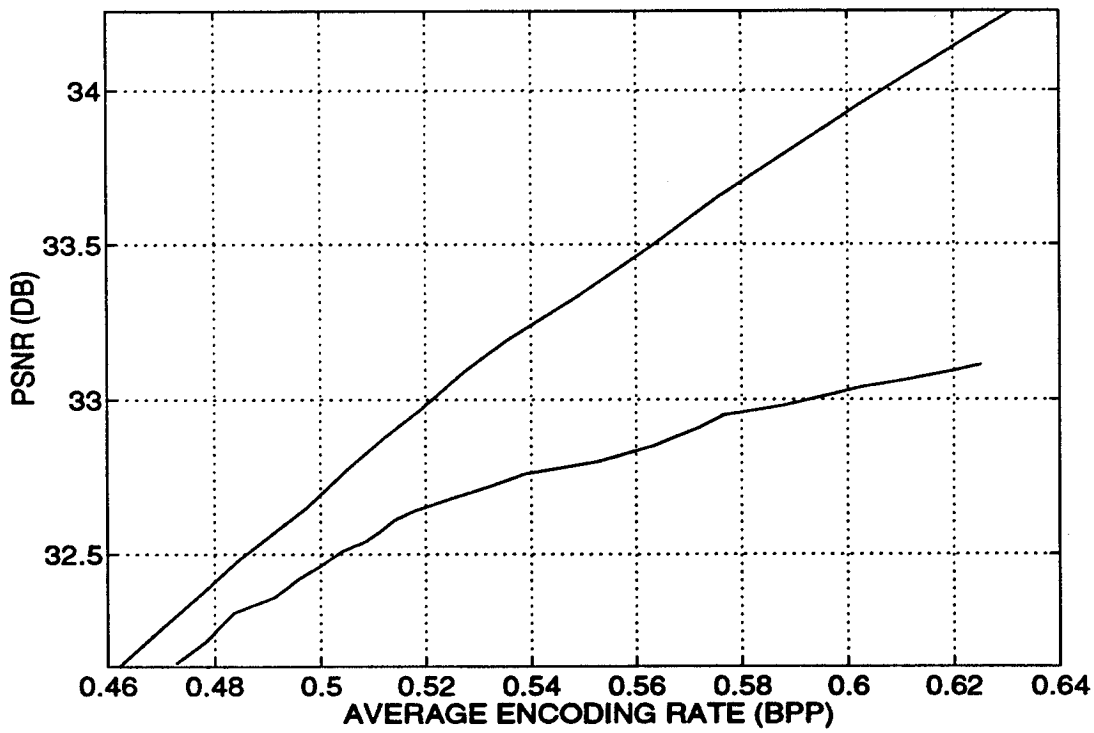

Figure 8: The rate-distortion performance of EC-RVQ for the test image Lena at two different peak bit rates (1.75 bpp for top, 1.00 bpp for bottom). The vector size is 4×4.

is noteworthy that EC-VQ based on high peak bit rates is not practical in general because of the large memory and complexity associated with the encoding and design procedures.

In the last set of experiments, 8×8 vector sizes were used in the design of EC-RVQ with $M = 4$ and $m = 2$ and with 7 stages codebooks of size 16. The maximum bit rate is then 0.4375 bpp. Figure 9 shows the coded image Lena at an average encoding rate of 0.1505 bpp. The PSNR is about 30 dB, and the subjective quality is rather good for a compression ratio of about 50 : 1. Practical EC-VQ systems are limited to relatively small vector sizes (typically 4×4) due to the exorbitant encoding and memory demands needed to implement such quantizers. While the EC-RVQ coding results (for 8×8 vectors) at such low bit rates cannot be compared with those of EC-VQ, they appear to be almost as good as those of more complex hybrid Subband/EC-RVQ/entropy coders reported in [21, 22].

6 Closing Remarks

The entropy-constrained RVQ introduced in this paper has many attractive features for data compression. In particular, the performance quality is among the best available to date. There also appear to be several areas where improvement can be made. One in particular is the entropy coding employed in the design algorithm. Equation (6) assumes the use of codewords that have non-integer lengths, and results in an average rate which is exactly equal to the output entropy of the EC-RVQ codebook. One can also employ (during the EC-RVQ design) an entropy coding algorithm of the entropy code that would follow the EC-RVQ. When employing a Huffman coding algorithm, both alternatives produced overall EC-VQ systems with nearly identical performance [5]. However, this may not be true in the case of EC-RVQ because the tables of conditional probabilities are usually very small (e.g. $4, 8, 16$), and the average lengths of the corresponding entropy codes may not be as close to the output

Figure 9: The image Lena coded at 0.1505 bpp. The vector size is 8×8. The PSNR is 30.05 dB.

entropy. Therefore, incorporating a Huffman encoder into the EC-RVQ design algorithm may lead to a significant increase in performance when that entropy coder is used to encode the RVQ stage indices. Another important issue that relates to the entropy coding problem is the fact that, since the conditional probability distributions of the latter stages are usually very skewed, other entropy coding techniques (such as arithmetic coding) may perform better than Huffman-based techniques. Experiments where both a Huffman encoder and an arithmetic encoder are separately incorporated into the EC-RVQ design algorithm are presently being investigated.

Another possible area for improvement is in the entropy coding structure. The present design algorithm is based on static entropy codes. However, with both the size and the number of the tables being relatively small, the possibility of adaptive entropy coding exists. This is another variation of the system presently being investigated.

Finally, we point out that EC-RVQ has some potential advantages in terms of channel insensitivity characteristics. Fixed rate RVQ tends to be less sensitive to channel errors than conventional VQ. A bit error could be disastrous for a conventional VQ, but is usually less serious when an RVQ is used. This nice property of RVQ seems to be lost when an EC-RVQ is used because a bit error in one of the stage codewords will very likely propagate through the subsequent stages, and will prevent the RVQ decoder from correctly decoding the variable length codewords of the remaining stages. However, the EC-RVQ variable length codewords become less sensitive to channel errors if we were to protect those variable length codewords of the first few stages.

List of Figures

List of Tables

References

[1] C. F. Barnes, *Residual Quantizers.* PhD thesis, Brigham Young University, Provo, Utah, Dec. 1989.

[2] C. F. Barnes and R. L. Frost, Necessary conditions for the optimality of residual vector quantizers. In *Proceedings of the IEEE International Symposium on Information Theory*, page 34, 1990.

[3] T. Berger, *Rate Distortion Theory*, Prentice Hall, Englewood Cliffs, NJ, 1971.

[4] A. Buzo, A. H. Gray, R. M. Gray, and J. D. Markel, "Speech coding based upon vector quantization", *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-28(5) pp.562-574, Oct. 1980.

[5] P. A. Chou, "Application of Entropy-Constrained Vector Quantization to Waveform Coding of Images," *Visual Communications and Image Processing IV*, SPIE-1199 pp. 970–978, 1989.

[6] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy-Constrained Vector Quantization", *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-37(1) pp. 31–42, January, 1989.

[7] J. H. Conway and N. J. A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes", *IEEE Transactions on Information Theory*, IT-28 pp. 227-232, Mar. 1982.

[8] R. G. Gallager, *Information Theory and Reliable Communication.* New York: Wiley, 1968.

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

[10] R. Gray, "Vector Quantization," ASSP Magazine, pp. 4-29, April 1984.

[11] R. M. Gray, J. C. Kieffer, and Y. Linde, "Locally optimal block quantizer design," *Information and Control*, 45(2) pp. 178-198, May 1980.

[12] D. A. Huffman, "A Method of The Construction of Minimum-Redundancy Codes," *Proceedings of the I.R.E.*, vol. 40, pp. 1098–1101, Sept. 1952.

[13] F. Jelinek and J. Anderson, "Instrumentable tree encoding for information sources," IEEE Transactions on Information Theory, IT-17 pp. 118-119, Jan 1971.

[14] B. H. Juang and A. H. Gray, Multiple stage vector quantization for speech coding. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 597–600, April 1982.

[15] F. Kossentini, W. Chung, and M. Smith, "Application of Entropy- Constrained Residual Vector Quantization to Coding Image Subbands," to appear in the *Proceedings of IEEE Int. Symposium on Circuits and Systems,* May 3-6, 1993.

[16] F. Kossentini, M. Smith, and C. Barnes,"Residual VQ with State Prediction: A New Method for Image Coding," *SPIE Symposium on Electronic Imaging*, February 24 - March 1, 1991.

[17] F. Kossentini, M. Smith, and C. Barnes, "A Perspective View of Finite State Binary Residual VQ," ISCAS91, volume 1, pages 300–303.

[18] F. Kossentini, M. Smith, and C. Barnes, "Large Block RVQ with Multipath Searching," ISCAS92, , volume 5, pages 2276–2279.

[19] F. Kossentini, M. Smith, and C. Barnes, "Image Coding with Variable Rate RVQ," ICASSP92, March 23-26, 1992.

[20] F. Kossentini, M. Smith, and C. Barnes, " Finite-State Residual Vector Quantization," Accepted for publication in *Journal of Visual Communication and Image Representation,* June 1993.

[21] F. Kossentini, W. Chung, and M. Smith, "Application of Entropy-Constrained RVQ to Coding Image Subbands," Proceeding of ISCAS93, Chicago, IL, May 3-6, 1993.

[22] F. Kossentini, W. Chung, and M. Smith, " Low Bit Rate Coding of Earth Science Images," Data Compression Conference, Snowbird, UT, March 29-April 1, 1993.

[23] F. Kossentini and M. Smith, and C. Barnes, " Necessary Conditions for Optimal Variable Rate RVQ," Submitted to *Transactions on Information Theory,* April 1993.

[24] F. Kossentini, M. Smith, and C. Barnes, " Locally Optimal RVQ Design Algorithms: Convergence," In preparation.

[25] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, COM-28:84–95, Jan. 1980.

[26] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding," *Proceedings of the IEEE*, 73(11):1551–1581, Nov. 1985.

[27] Y. Shoham, *Hierarchical Vector Quantization with Application to Speech Waveform Coding*, PhD thesis, University of California at Santa Barbara, Mar. 1985.