NDB MIP 9116113

IN-61-CR
027422
Ⓒ UNAIVED

Conditional Entropy-Constrained Residual VQ with Application to Image Coding

Faouzi Kossentini, Wilson C. Chung, *Student Member, IEEE*, and Mark J. T. Smith, *Fellow, IEEE*

NASA-CR-204396

***Abstract*— This paper introduces an extension of entropy-constrained residual vector quantization (VQ) where intervector dependencies are exploited. The method, which we call conditional entropy-constrained residual VQ, employs a high-order entropy conditioning strategy that captures local information in the neighboring vectors. When applied to coding images, the proposed method is shown to achieve better rate-distortion performance than that of entropy-constrained residual vector quantization with less computational complexity and lower memory requirements. Moreover, it can be designed to support progressive transmission in a natural way. It is also shown to outperform some of the best predictive and finite-state VQ techniques reported in the literature. This is due partly to the joint optimization between the residual vector quantizer and a high-order conditional entropy coder as well as the efficiency of the multistage residual VQ structure and the dynamic nature of the prediction.**

I. Introduction

ENTROPY coding is now being used frequently in conjunction with vector quantization (VQ) for image coding. Its use is motivated by the fact that the probability distribution of VQ coded images is generally skewed or nonuniform. While the average bit rate can most often be reduced by entropy coding the VQ codewords, improvement in rate-distortion performance is usually attainable by embedding the entropy coding in the design process such that both the VQ codebook and entropy coder are optimized jointly.

By generalizing the entropy-constrained scalar quantization design [1]–[3] to the vector case, Chou *et al.* introduced an iterative descent algorithm for the design of entropy-constrained vector quantizers (EC-VQ's) [4]. Later, Chou applied EC-VQ to image coding [5] and showed that the entropy-constrained optimization yields a significant performance gain. More recently, the entropy-constrained optimization was applied to residual VQ (RVQ), which is also known as multistage VQ [6], [7]. This form of VQ, which is called entropy-constrained residual VQ (EC-RVQ) [8]–[10], consists of a cascade of VQ stages where each stage operates on the input/output difference of the previous stage. The individual stage symbols are entropy coded based on a model using probabilities that are conditioned on previous stage output symbols. For a number of reasons stemming from the memory and computationally efficient structure of the RVQ, EC-RVQ can outperform EC-VQ and usually achieves image compression results competitive with those of JPEG and subband coding [8], [9].

Manuscript received July 5, 1994; revised July 26, 1995. This work was supported by the National Science Foundation under contract MIP-9116113 and the National Aeronautics and Space Administration.

The authors are with Digital Signal Processing Laboratory, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0250 USA.

Publisher Item Identifier S 1057-7149(96)01305-X.

Like EC-VQ, EC-RVQ is a memoryless vector quantizer. This is because the EC-RVQ design algorithm [8], [10] minimizes the distortion subject to a constraint on the first order (or zero order conditional) entropy of the vector quantizer output. However, better rate-distortion performance can generally be achieved by incorporating memory into the vector quantizer. One way to incorporate memory is to employ an entropy coder whose output is conditioned on previous quantizer outputs. This allows for the incorporation of information about neighboring vectors into the coding of the current vector, which is tantamount to exploiting the inherent memory of practical sources such as speech and images. In fact, by using a first-order conditional EC-VQ of linear predictive speech coefficients [11], Chou and Lookabaugh obtained a 40–60% reduction in bit rate as compared to conventional EC-VQ. However, even by limiting the number of conditioning symbols, or previously coded vectors, to *one*, $256^2 = 65\,536$ conditional probabilities had to be estimated and stored. Since the entropy coder cost, in terms of computation and memory, increases *exponentially* with the number of conditioning symbols and the size of the EC-VQ codebook, this technique can quickly become impractical.

This paper extends EC-RVQ to a vector quantizer that exploits memory by using *higher order* conditional entropy codes. Unlike EC-RVQ, which conditions on the output of the previous stages, the high-order conditional EC-RVQ (CEC-RVQ) introduced here takes advantage of the information available in previously coded vectors by conditioning over a spatial-stage region of support. While conditional EC-VQ is severely impaired by the exponential dependence of memory and complexity on the number of conditioning symbols and the VQ codebook size, CEC-RVQ is not as sensitive. However, some constraining exponential dependencies are still present. Hence, the central part of this work is the introduction of an effective strategy for achieving high rate-distortion performance subject to constraints on computation and memory. The high-order CEC-RVQ presented next is shown to achieve reductions in bit rate by typically 30–40% over EC-RVQ while maintaining the same reproduction quality. Perhaps even more significant is the fact that this improvement can be achieved without the enormous storage and complexity requirements that accompany high-order conditional EC-VQ, finite state VQ, and other predictive schemes of this type [7], [11]–[14]. Finally, as with other multistage and/or tree-structured VQ

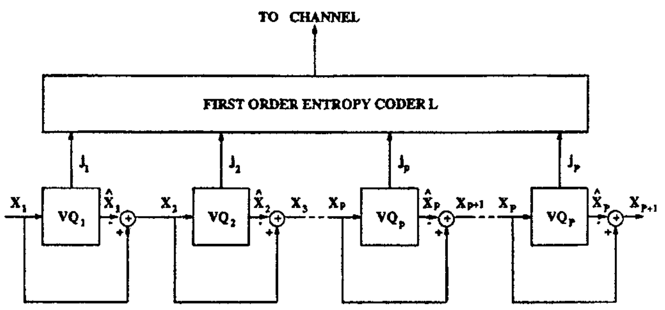
TO CHANNEL
FIRST ORDER ENTROPY CODER L
j_1 j_2 j_p j_P
X_1 VQ_1 $\hat{X}_1$ X_2 VQ_2 $\hat{X}_2$ X_3 X_p VQ_p $\hat{X}_p$ X_{p+1} X_P VQ_P $\hat{X}_P$ X_{P+1}

Fig. 1. High-level structure of a P-stage EC-RVQ encoder.

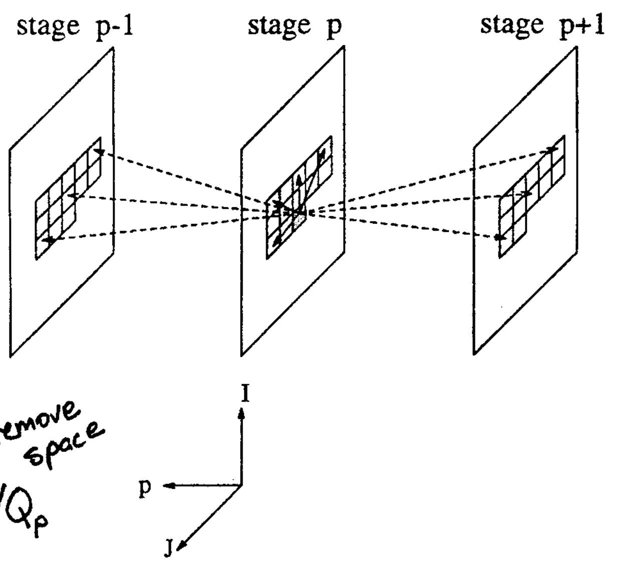

Fig. 2. Illustration of a conditioning structure for a 12th-order CEC-RVQ.

schemes, CEC-RVQ can be designed to support progressive transmission. Even though some compromises in overall R(D) performance are unavoidable, the losses are not severe, and the net results are very competitive.

remove space

VQ_p

II. High-Order Conditional EC-RVQ (CEC-RVQ)

To set the stage for discussion, consider the P-stage EC-RVQ encoder shown in Fig. 1. Each stage mapping, VQ_p is a composition of a stage encoder mapping E_p such that $E_p(X_p) = j_p$, where j_p is the output symbol or index and a stage decoder mapping D_p given by $D_p(j_p) = \hat{X}_p$. The outputs $j_1, j_2, \cdots, j_P$ of the P-stage encoders are entropy coded using the first-order entropy encoder L. More specifically, let X^n be the nth vector of k random variables taken from a discrete-time, continuous-valued source. The input $\boldsymbol{x}^n$, which is a realization of $\boldsymbol{X}^n$, is encoded using a fixed-length encoder mapping $E = (E_1, E_2, \cdots, E_P)$ to produce a vector of fixed-length codewords or symbols $\boldsymbol{j}^n = (j_1^n, j_2^n, \cdots, j_P^n)$. Assuming that a given stage VQ codebook $\mathcal{C}_p$ has N_p code vectors, the symbol j_p^n belongs to the set $\mathcal{J}_p = \{0, 1, \cdots, N_p - 1\}$. A variable-length entropy encoder L is then applied to map the symbols $j_1^n, j_2^n, \cdots, j_P^n$ into a variable-length codeword $c(j_1^n, j_2^n, \cdots, j_P^n)$, where $c(j_1^n, j_2^n, \cdots, j_P^n)$ is a concatenation of P stage-conditional variable-length codewords $c_1, c_2, \cdots, c_P$. The decoding process consists of applying the entropy decoder L^{-1} to recover the symbols $j_1^n, j_2^n, \cdots, j_P^n$, which are then used by the fixed-length decoder mapping $D = (D_1, D_2, \cdots, D_P)$ to identify the vectors in the appropriate stage codebooks. Reconstruction is achieved by simply *summing* the decoded stage code vectors. The encoding/decoding process can be represented compactly by $\hat{\boldsymbol{x}}_n = \boldsymbol{Q}(\boldsymbol{x}_n) = \boldsymbol{D}[\boldsymbol{E}(\boldsymbol{x}_n)]$, where $\hat{\boldsymbol{x}}_n$ is the reconstructed vector, and $\boldsymbol{Q}$, $\boldsymbol{E}$, and $\boldsymbol{D}$ are the direct-sum quantizer, direct-sum encoder, and direct-sum decoder, respectively.

The goal of the EC-RVQ design algorithm is to seek a set of stage codebooks that minimizes the average distortion subject to a constraint on the zero-order conditional entropy of the RVQ codewords. This is done iteratively (as in [4]) based on the necessary conditions derived in [10]. Specifically, the algorithm minimizes the Lagrangian

$$J_\lambda = E[d(X^n, \hat{X}^n)] + \lambda E[\ell(L(J^n))]$$

where $d(\boldsymbol{x}^n, \hat{\boldsymbol{x}}^n)$ is the distortion between $\boldsymbol{x}^n$ and $\hat{\boldsymbol{x}}^n$, and $\ell[L(\boldsymbol{j}^n)]$ is the length of the codeword $L(\boldsymbol{j}^n)$ associated with the vector of P-stage indices $\boldsymbol{j}^n = (j_1^n, j_1^n, \cdots, j_P^n)$,

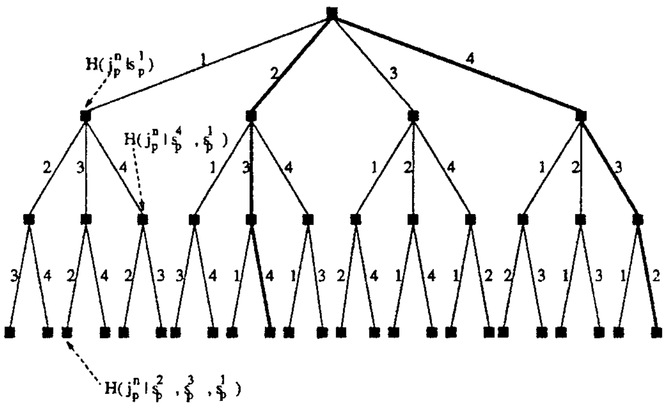

Fig. 3. Three-level, 4-ary tree with $R_p = 4$ and $m_p = 3$.

where $\boldsymbol{j}^n$ is a realization of J^n and a member of the set $\mathcal{J} = \mathcal{J}_1 \times \mathcal{J}_2 \times \cdots \times \mathcal{J}_P$. The Lagrangian parameter λ is used to weight the influence of the expected codeword length or, equivalently, the bit rate, against the expected distortion. For each parameter λ, the EC-RVQ algorithm finds points on the convex hull of the operational distortion-rate curve given by the function [4]

$$D_k(R) = \inf_{(\boldsymbol{E}, \boldsymbol{D}, L)} \{E[d(X^n, \boldsymbol{D}(\boldsymbol{E}(X^n)))] | E[\ell(L(J^n))] \le R\}.$$

The design algorithm for the mth-order *conditional EC-RVQ*, which we will call the CEC-RVQ algorithm, minimizes the Lagrangian

$$J_\lambda = E[d(X^n, \hat{X}^n)] + \lambda E[\ell(L(J^n | J^{n-1}, J^{n-2}, \cdots, J^{n-m}))]$$

where $\ell[L(\boldsymbol{j}^n | \boldsymbol{j}^{n-1}, \boldsymbol{j}^{n-2}, \cdots, \boldsymbol{j}^{n-m})]$ is the length of the codeword conditioned on m previous output symbols $\boldsymbol{j}^{n-1}, \boldsymbol{j}^{n-2}, \cdots, \boldsymbol{j}^{n-m}$. The encoder and decoder optimization steps of the mth-order CEC-RVQ algorithm are identical to those of the EC-RVQ algorithm. However, the entropy coder optimization is different because conditioning is performed on not only previously coded stage vectors but also on previously coded neighboring vectors. Assuming the use of an ideal entropy coder, the minimum expected length of the conditional entropy codes is essentially the mth-order conditional entropy of the RVQ codewords, i.e.

$$\min_{\boldsymbol{L}} E[\ell(L(\boldsymbol{J}^n|\boldsymbol{J}^{n-1}, \boldsymbol{J}^{n-2}, \cdots, \boldsymbol{J}^{n-m}))] = H(\boldsymbol{J}^n|\boldsymbol{J}^{n-1}, \boldsymbol{J}^{n-2}, \cdots, \boldsymbol{J}^{n-m}) \tag{1}$$

where H denotes the conditional entropy. Since $\boldsymbol{j} = (j_1, j_2, \cdots, j_P)$, we can write the conditional self-information, which corresponds to the entropy given by (1), as

$$\ell(L^*(\boldsymbol{j}^n|\boldsymbol{j}^{n-1}, \boldsymbol{j}^{n-2}, \cdots, \boldsymbol{j}^{n-m})) = \sum_{p=1}^{P} - \log_2 \mathrm{pr}\,(j_p^n | \underbrace{j_{p-1}^n, \cdots, j_1^n}_{p-1}, \overbrace{\underbrace{j_P^{n-1}, \cdots, j_1^{n-1}}_{P}, \underbrace{j_P^{n-2}, \cdots, j_1^{n-2}}_{P}, \cdots, \underbrace{j_P^{n-m}, \cdots, j_1^{n-m}}_{P}}^{m}) \tag{2}$$

where L^* satisfies (1). Since exact analytical descriptions for probability distributions of images do not exist, a large training set is employed in the entropy coder optimization step to estimate the conditional probabilities.

III. Complexity Issues

The complexity of the mth-order CEC-RVQ design algorithm can quickly become very large depending on the number of stage codebooks, the size of each stage codebook, and the order m. Fortunately, all of the complexity reduction techniques such as using M search in encoding, which have been developed for the EC-RVQ design algorithm [8], [10], can be used in this algorithm. However, the entropy coder optimization step of the proposed algorithm is much more complicated than that of the EC-RVQ algorithm. As can be seen from (2), the memory requirements, design, and implementation of the mth-order entropy coder can easily become unmanageable. The rest of this section discusses techniques for substantially reducing the computational complexity and memory of the high-order entropy coder while sacrificing minimal loss in performance.

Equation (2) reveals that the length of the optimal pth-stage variable length mapping is the conditional self-information given all previously coded $(mP+p-1)$ stage vectors. This is illustrated graphically (in the context of image coding) in Fig. 2, where the shaded block in the middle is the stage vector on which conditioning is being performed. A total of m (12 in this case) neighboring blocks is utilized for conditioning. These blocks define the spatial region supporting the conditioning. The solid arrows show these neighboring blocks at the pth stage. In addition to the spatial dimension, conditioning is based on corresponding blocks at different stage levels, which is shown in the figure using dashed arrows for the $(p-1)$th and $(p+1)$th stages. The 3-D spatial-stage region of support illustrated in Fig. 2 is uniquely represented by the two spatial displacements ΔI and ΔJ and the stage displacement Δp, all with positive values in the direction of the reference axis. Obviously, the number of all possible combinations of triples $(\Delta I, \Delta J, \Delta p)$ is equal to the number of previously coded $(mP+p-1)$-stage vectors.

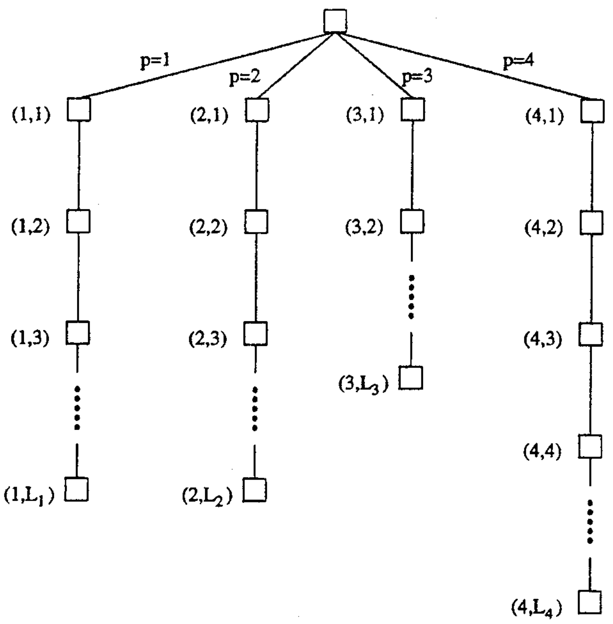

Fig. 4. 4-ary tree for allocation of orders of stage entropy coders. The pair (p, L_p) corresponds to $(\mathcal{N}_{p,L_p}, H_{p,L_p})$.

As is implied, the RVQ stage structure is assumed to have P stages with a fixed number N_p $(p = 1, 2, \cdots, P)$ of code vectors in each stage codebook. The set of all combinations of conditioning symbols representing blocks (or vectors) in the spatial-stage region of support is the set of conditioning states. Each state defines a unique set of codeword probabilities for the pth stage. The number of conditioning *states* S_p for the pth stage is given by

$$S_p = \left[\prod_{i=1}^{P} (N_i)^m\right] (N_1)(N_2)\cdots(N_{p-1}). \tag{3}$$

It is clear from the above equation that the number of conditioning states increases rapidly with the number of neighboring blocks, the number of stages P, and the stage codebook sizes $N_1, N_2, \cdots, N_P$. Even by limiting m, P, and $N_1, N_2, \cdots, N_P$ to small values, S_p can still be very large, requiring a large number of computations to estimate the conditional probabilities and an exorbitant amount of memory to store them. For example, consider a fourth-order CEC-RVQ (i.e., an EC-RVQ that is conditioned on four symbols representing four neighboring vectors), where the RVQ contains eight stage codebooks, each with four code vectors. Then, the number of conditioning stage symbols or previously coded stage vectors at the pth stage is $[(4)(8)+p-1]$ or $31+p$. Since (3) gives the number of states for each stage p, the total number of conditional probabilities that must be computed and stored for all P stages is $(S_1)(N_1) + (S_2)(N_2) + \cdots + (S_P)(N_P)$.

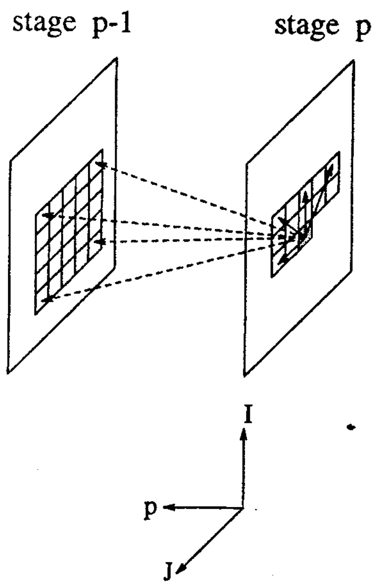

Fig. 5. Illustration of a conditioning structure for CEC-RVQ in a progressive transmission environment.

In our example, this is equivalent to $4^{33} + 4^{34} + \cdots + 4^{40} = (4^{33})(21845) \approx 1.61 \times 10^{24}$, which is unmanageable.

To reduce the complexity of the entropy coder, the number of conditioning stage symbols for a given stage p is limited to m_p, where $m_p \ll (mP + p - 1)$. Since this will increase the entropy in general, the appropriate solution is to select the particular m_p conditioning stage symbols that result in the lowest possible entropy for the pth stage. The optimal solution can be found by exhaustively searching the $[R_p!/m_p!(R_p - m_p)!]$ possibilities, where $R_p = mP + p - 1$. Of course, this can quickly become a formidable task, especially when m_p and R_p are large.

The problem of choosing a subset of conditioning symbols has been considered previously in other contexts, with varying degrees of success. There are several reasonable possibilities, some of which we highlight next. The selection of the best subset of conditioning symbols can be heuristically based. For example, symbols representing vectors closer in space to the current vector can be selected first. This leads to reasonable coding results in many practical predictive quantization systems. However, this approach falls short of achieving the best performance for a given number of conditioning symbols. This is because using neighboring vectors does not necessarily lead to the lowest achievable conditional entropy. A better and simple way to select, say, m_p conditioning stage symbols is to select $s_p^1 \in \mathcal{R}_\mathrm{p}$, where $\mathcal{R}_\mathrm{p}$ is the spatial-stage region that contains R_p conditioning stage symbols such that the first-order conditional entropy $H(j_p^n|s_p^1)$ is a minimum. Then, the rest of the sequence (i.e., $s_p^2, \cdots, s_p^{m_p}$) is determined by selecting s_p^i, where $H(j_p^n|s_p^i)$ is the ith smallest first-order conditional entropy. The problem with this technique, though, is that its effectiveness decreases quickly as the order m_p increases. This is because $H(j_p^n|s_p^{i-1}) < H(j_p^n|s_p^i) < H(j_p^n|s_p^{i+1})$ does not, in general, imply $H(j_p^n|s_p^{i-1}, s_p^i) < H(j_p^n|s_p^{i-1}, s_p^{i+1})$. In fact, since the selected conditioning stage symbols often belong to different stages, the statistical dependencies between them are usually complicated, making this simple technique inadequate even for moderate values of m_p.

Another approach, which is described in [15], consists of using a certain conditional entropy to decide the order of the conditioning symbols sequentially. Given that the first $(i-1)$ conditioning symbols $(s_p^1, s_p^2, \cdots, s_p^{i-1})$ have been decided, the ith conditioning symbol s_p^i is chosen to be that one such that the conditional entropy $H(j_p^n|s_p^i, s_p^1 = c_0, \cdots, s_p^{i-1} = c_0)$, where c_0 is the most probable previous symbol, is a minimum. This technique was shown to perform well when used in subband HDTV coding [15]. However, our experimental results show that this suboptimal method does not seem to consistently provide good complexity/performance tradeoffs.

In this paper, we introduce an effective and efficient algorithm that can achieve a performance arbitrarily close to that of the exhaustive search technique. This algorithm is based on the idea of tree search and is illustrated with the aid of Fig. 3. The tree shown in Fig. 3 is constructed such that the root node has R_p branches where the ith ($i = 1, 2, \cdots, R_p$) branch corresponds to the first-order conditional entropy $H(j_p^n|s_p^i)$. Then, the ith node at the first level has $R_p - 1$ branches where the jth ($j = 1, 2, \cdots, R_p - 1$) branch corresponds to the conditional entropy $H(j_p^n|s_p^j, s_p^i)$, and so on until the $(m_p - 1)$th level is reached, where each node has $(R_p - m_p + 1)$ branches corresponding to $H(j_p^n|s_p^1, s_p^2, \cdots, s_p^{m_p})$. Note that

$$s_p^1 \in \mathcal{R}_\mathrm{p},$$
$$s_p^2 \in \mathcal{R}_\mathrm{p} \quad \text{and} \quad s_p^2 \neq s_p^1,$$

and

$$s_p^{m_p} \in \mathcal{R}_\mathrm{p} \quad \text{and} \quad s_p^{m_p} \neq s_p^1, \cdots, s_p^{m_p} \neq s_p^{m_p - 1}.$$

One can easily see that this tree is symmetric in the sense that the *order* of the selected symbols is not important. For example, the path (2, 3, 4) shown in Fig. 3 is the same as the path (4, 3, 2). Exploiting this symmetry property can substantially increase the speed of the searching process. It is evident that the simplest approach is to sequentially search the tree. In such a case, the conditioning symbol leading to the lowest first-order conditional entropy must also be one of the two symbols resulting in the lowest second-order conditional entropy, and so on. Since this is generally not true, lower entropies can be obtained by using the (M, L) algorithm [16] or the dynamic M-search technique [17]. This is because such searching techniques usually save more than one tree path, which is a process that may produce a sequence of conditioning symbols, where none were chosen at previous levels of the tree. Dynamic M-search is used in this work because it achieves close-to-optimal performance with very little additional computational and memory complexity. Finally, note that depending on the size of the spatial-stage region of support or, equivalently, the number R_p of conditioning symbols, even stage-sequential searching of the tree shown in Fig. 3 can require an exorbitant amount of computations and tremendous memory requirements. Although the searching process is done during the design (i.e., off-line), this can still be an obstacle. However, by using the assumption that the

TABLE I
NUMBER AND LOCATION OF CONDITIONING SYMBOLS: SYMBOL NO. 1, SYMBOL NO. 2, SYMBOL NO. 3, AND SYMBOL NO. 4

Stage	No. of Symbols	$\mathcal{N}_p$		Symbol # 1	Symbol # 2	Symbol # 3	Symbol # 4
1	4	1024	ΔI	1	0	1	2
			ΔJ	0	1	-1	0
			Δp	0	0	0	-1
2	4	1024	ΔI	1	0	1	2
			ΔJ	0	1	-1	-1
			Δp	0	0	0	1
3	3	256	ΔI	1	0	0	N/A
			ΔJ	0	1	0	N/A
			Δp	0	0	1	N/A
4	4	1024	ΔI	0	0	1	1
			ΔJ	0	0	0	0
			Δp	1	2	0	1
5	2	64	ΔI	0	0	N/A	N/A
			ΔJ	1	0	N/A	N/A
			Δp	0	2	N/A	N/A
6	3	256	ΔI	0	0	0	N/A
			ΔJ	0	0	0	N/A
			Δp	1	4	3	N/A
7	3	256	ΔI	0	0	0	N/A
			ΔJ	0	0	0	N/A
			Δp	1	2	3	N/A
8	2	64	ΔI	0	1	N/A	N/A
			ΔJ	0	0	N/A	N/A
			Δp	1	0	N/A	N/A
9	2	64	ΔI	0	1	N/A	N/A
			ΔJ	0	0	N/A	N/A
			Δp	1	0	N/A	N/A
10	2	64	ΔI	1	0	N/A	N/A
			ΔJ	0	1	N/A	N/A
			Δp	1	0	N/A	N/A

memory of the source decays rapidly, we can choose a region of support containing only a moderate number R_p ($50 \leq R_p \leq 100$) of conditioning stage symbols. Experimental work in image coding supports the validity of such an assumption.

Since our objective is to minimize the average output rate of all the stages given a fixed level of entropy coder complexity and memory, the parameters $\{m_p, 1 \leq p \leq P\}$ must be carefully determined. For each stage p, complexity and memory of the entropy coder grow *exponentially* with m_p and the output alphabet sizes of the stage VQ's. This is because the number of conditioning *states* at stage p is equal to the product of the stage codebook sizes corresponding to the selected stage symbols s_p^i, i.e.

$$S_p = N_{\phi(s_p^1)} N_{\phi(s_p^2)} \cdots N_{\phi(s_p^{m_p})}. \tag{4}$$

The function ϕ maps the symbol s_p^i into its corresponding stage value. For example, if symbol s_p^i is the second-stage code vector in one of the neighboring blocks, then $\phi(s_p^i)$ is 2, and $N_{\phi(s_p^i)} = N_2$, which is the size of the second-stage codebook. The problem now translates into minimizing the output entropy

$$H = \sum_{p=1}^{P} H(j_p^n | s_p^1, s_p^2, \cdots, s_p^{m_p}) \tag{5}$$

subject to the constraint that

$$\sum_{p=1}^{P} \mathcal{N}_p \leq N^{\max}$$

where N_p is the size of the pth-stage codebook, $\mathcal{N}_p = S_p N_p$, and $N^{\max}$ is the maximum allowed number of conditional probabilities or variable length codes. $N^{\max}$ provides some control over the system cost because it is a measure of complexity and memory required by the entropy coder.

A solution to (5) can be found by constructing another tree, which is shown in Fig. 4. The root node of this tree has P branches: one per stage. The subtree rooted at each branch node p is a unary tree of length L_p. Each subtree has L_p nodes, where each node $(p, i)(1 \leq i \leq L_p)$ corresponds to a pair $(\mathcal{N}_{p,i}, H_{p,i})$, where $\mathcal{N}_{p,i} = S_{p,i} N_p = N_{\phi(s_p^1)} N_{\phi(s_p^2)} \cdots N_{\phi(s_p^i)} N_p$ is the number of conditional probabilities that correspond to the pth-stage conditional entropy $H_{p,i} = H(j_p^n | s_p^1, s_p^2, \cdots, s_p^i)$. Clearly, we must have, for $p = 1, 2, \cdots, P$.

$$H_{p,1} \geq H_{p,2} \geq \cdots \geq H_{p,L_p}.$$

Therefore, the node $(p, 1)$, or the node closest to the root node, corresponds to the pair $(\mathcal{N}_{p,1}, H_{p,1})$, and the node (p, L_p), or the node farthest from the root node, corresponds to the pair $(\mathcal{N}_{p,L_p}, H_{p,L_p})$. For each stage p, $H_{p,i}$ is obtained using dynamic M-search (as mentioned before) through locating, for each i, the best i conditioning symbols.

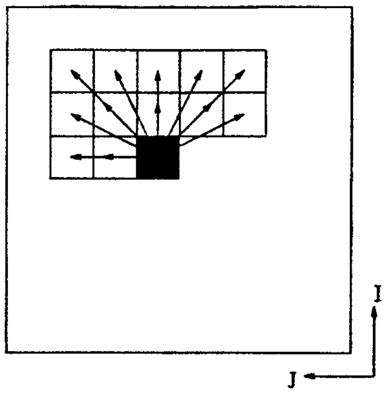

Fig. 6. Spatial region of support.

Let $\mathcal{S}$ be a pruned subtree of the constructed tree $\mathcal{T}$, where the stage associated with the pth branch now has a number of conditional probabilities α_p and an entropy h_p. The number of conditional probabilities associated with $\mathcal{S}$ is

$$\alpha(\mathcal{S}) = \sum_{p=1}^{P} \alpha_p$$

and the high-order conditional entropy is

$$h(\mathcal{S}) = \sum_{p=1}^{P} h_p$$

where $H_{p,1} \geq h_p \geq H_{p,L_p}$. The optimal pruning (or BFOS) algorithm described in [18] and [19] can be used to find the numbers $\alpha_1^*, \alpha_2^*, \cdots, \alpha_p^*$ that minimize $h(\mathcal{S})$ subject to $\alpha(\mathcal{S}) \leq N^{\max}$, where $N^{\max}$ is the maximum allowed number of conditional probabilities over all pruned subtrees $\mathcal{S} \preceq \mathcal{T}$.

Note that the depth L_p of the BFOS tree surely needs to be $\ll (mP + p - 1)$. Due to obvious exponential dependencies, increasing L_p will quickly increase both the search complexity and memory required to store the estimated probabilities. In fact, there is usually not enough image data to obtain reasonable estimates of very high-order conditional probabilities. Fortunately, very high-order stage statistical models are neither necessary nor useful. Experimental results indicate that most of the entropy reduction is usually obtained by using values of L_p between 1-8 and that very little gain is achieved by choosing larger values.

IV. Progressive Transmission

CEC-RVQ is potentially attractive for use in a progressive transmission environment. The successive approximation nature of the RVQ structure results in multistage approximations of the input image. Thus, a lowpass approximation can be obtained by transmitting information from only the first-stage indices. Then, the quality of the reproduction can be successively improved by transmitting information from subsequent stage VQ's. This suggests that the CEC-RVQ stage indices be coded and transmitted in a different order. Instead of sending all stage codewords that correspond to a block before moving to the next block, we send the codewords on a stage-order basis. In such a mode of operation, the conditioning structure shown in Fig. 2 is clearly not appropriate. Subsequent stage indices for previously coded vectors are no longer available to the decoder. However, noncausal spatial regions of support can be used instead. Fig. 5 illustrates a typical conditioning scheme in a progressive transmission environment. Notice that half-plane support is present for conditioning at the current stage level p, and full-plane support is available for the previous stage level $p-1$. Having full-plane support in stages 1 to $p-1$ allows CEC-RVQ to exploit more spatial dependencies, which are usually stronger that interstage dependencies. In fact, such a conditioning structure is shown experimentally to lead to a 1-5% reduction in average entropy for the same complexity.

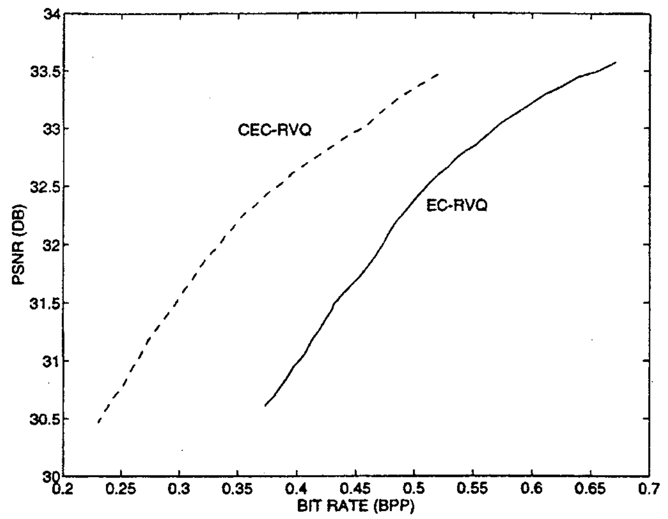

Fig. 7. PSNR (in decibels) for EC-RVQ and CEC-RVQ. The vector size is 4 × 4. The initial RVQ codebook contains 10 stage codebooks, each with 4 stage code vectors.

A negative consequence of this approach, however, is that noncausal encoding/decoding procedures cannot be accommodated. In particular, neither M-search nor joint RVQ decoder optimization should be used. This is because we do not know when transmission is going to be halted. Abandoning these noncausal procedures has the disadvantage of lowering the overall rate-distortion performance but, on the positive side, has the potential for better reconstruction quality at the intermediate stages. A description of a fully embedded 8×8 CEC-RVQ is presented in the next section.

V. System Specifications and Results

The CEC-RVQ was examined carefully in the context of image coding. Twelve 8 b/pixel monochrome images of size 512×512, including six luminance images extracted from color images taken from the USC database, were used to design CEC-RVQ codebooks. In all cases, test images were excluded from the training set.

In the first experiment, each CEC-RVQ codebook contains ten stage codebooks with four 4×4 code vectors in each codebook. The conditioning scheme we use is the one illustrated in Fig. 2. There are two types of dependencies exploited by the CEC-RVQ: intrastage, denoted by the solid arrows, and interstage, denoted by the dashed arrows. Both previous and subsequent stage symbols are used in the search for the best m_p conditioning symbols. The only exception is the current symbol where only previous stage symbols can be used. In other words, conditioning is restricted to previously coded vectors. Fig. 6 shows the spatial region of support for a single stage. As shown in Figs. 2 and 6, only $(12)(10) + p - 1 = 119 + p$ stage symbols are searched at stage p. This choice leads to a good compromise between rate-distortion performance and search complexity. The top two rows, left two columns, and right two columns of the input image are special cases because they are at the image boundaries and consequently are treated separately. For such input vectors, the entropy tables are constructed based on first-order interstage conditioning only. This does require storing an extra set of tables, but the additional memory involved is not significant.

To minimize the output entropy for a fixed maximum number of 4096 conditional probabilities, a balanced tree with depth 6 is constructed, where the best six conditioning stage symbols are used. The BFOS algorithm described in the previous section is used producing the results shown in Table I. As mentioned in Section III, the variable Δp denotes the stage displacement, with negative values indicating that the associated conditioning stages are subsequent ones, i.e., the conditioning stages are $|\Delta p|$ in advance of the current stage (see Fig. 2). The variables ΔI and ΔJ denote the 2-D displacements in the image that specify the conditioning stage symbols spatially. Negative values of ΔI and ΔJ indicate displacements to the bottom and the right, respectively. Notice that some of the symbols selected by the algorithm are not the closest ones to the current symbol. The variable $\mathcal{N}_p$ denotes the number of conditional probabilities or variable-length entropy codes (at the pth stage) that must be stored. Since all stage codebooks have four stage code vectors, $\mathcal{N}_p$ must be a power of 4. By adding the $\mathcal{N}_p$'s, we obtain 4096, which is the target number of conditional probabilities. For this constraint of 4096 conditional probabilities, the algorithm obtained the spatial-stage region shown in Table I as being the best one. These results may differ significantly for a different set of training images and a different initial region of support. However, the results presented in Table I, as well as results of other experiments using different sets of images, show consistently that a majority of the conditioning symbols represent adjacent coded blocks (or vectors), indicating that spatial dependencies are stronger than interstage dependencies.

Fig. 7 shows the PSNR performance of EC-RVQ and CEC-RVQ for the test image LENA based on 4×4 vectors. Each EC-RVQ codebook also contains ten stage codebooks with four code vectors per stage codebook. Both EC-RVQ and CEC-RVQ codebooks are optimized jointly and are searched using the M-search technique (with $M = 2$), requiring 76 vector Lagrangian calculations per input vector. Moreover, entropy conditioning was performed based on the four previous stages of the RVQ. The number of conditional probabilities that must be computed and stored is

$$4^1 + 4^2 + 4^3 + \underbrace{4^4 + \cdots + 4^4}_{7} = 6484.$$

TABLE II
PSNR (IN DECIBELS) OF CEC-RVQ, FS-VQ, VR-FSVQ, PR-VR-FSVQ, AND DFS-VQ FOR THE IMAGE LENA AT 0.38, 0.31, AND 0.25 bpp. THE VECTOR SIZE IS 4 × 4

Rate	CEC-RVQ1	CEC-RVQ2	FSVQ	VR-FSVQ	PR-VR-FSVQ	DFS-VQ
0.38	32.62	32.98	30.16	32.00	N/A	31.67
0.31	32.04	32.38	29.56	31.66	31.19	31.29
0.25	31.18	31.51	28.83	30.31	30.74	30.20

TABLE III
NUMBER OF SYMBOLS (NS) AND LOCATION (ΔI, ΔJ, Δp) OF CONDITIONING SYMBOLS: SYMBOL no. 1, SYMBOL no. 2, AND SYMBOL no. 3 FOR THE 8 × 8 CEC-RVQ

Stage	NS	$\mathcal{N}_p$	Symbol # 1	Symbol # 2	Symbol # 3
1	3	256	(1 0 0)	(0 1 0)	(2 0 −1)
2	2	64	(1 0 0)	(0 0 1)	N/A
3	2	64	(0 0 2)	(0 0 1)	N/A
4	2	64	(0 0 3)	(0 0 1)	N/A
5	2	64	(0 0 1)	(0 0 4)	N/A
6	1	16	(0 0 5)	N/A	N/A
7	2	64	(0 0 2)	(0 0 5)	N/A
8	2	64	(0 0 2)	(0 0 6)	N/A
9	1	16	(0 0 3)	N/A	N/A
10	2	64	(0 0 1)	(0 0 4)	N/A
11	2	64	(0 0 3)	(0 0 1)	N/A
12	1	16	(0 0 3)	N/A	N/A
13	2	64	(0 0 1)	(0 0 4)	N/A
14	2	64	(0 0 1)	(0 0 2)	N/A
15	1	16	(0 0 1)	N/A	N/A
16	1	16	(0 0 1)	N/A	N/A
17	0	4	N/A	N/A	N/A
18	1	16	(0 0 1)	N/A	N/A
19	1	16	(0 0 1)	N/A	N/A
20	0	4	N/A	N/A	N/A

Thus, the entropy coder complexity and memory of EC-RVQ is approximately 1.6 times that of the CEC-RVQ. By examining the figure, it is evident that the bit rate for CEC-RVQ is reduced by as much as 40% for the same PSNR as for EC-RVQ. In addition to the rate-distortion performance improvement, CEC-RVQ requires less entropy-coder complexity and memory.

Next, we compare CEC-RVQ with some of the finite-state and predictive VQ techniques described in the VQ literature. Table II shows the PSNR performance of two CEC-RVQ's, a mean-removed memoryless finite-state VQ (FS-VQ) [13], a mean-removed variable rate FSVQ (VR-FSVQ) using pruned tree-structured VQ [13], a predictive variable rate FSVQ (PR-VR-FSVQ) also using pruned tree-structured VQ [13], and a dynamic FSVQ (DFSVQ) [14]. CEC-RVQ1 is the one used in the previous experiment. CEC-RVQ2 is similar to CEC-RVQ1 but contains 32 stage codebooks, each with two code vectors of size 4×4. The stage codebooks are searched using the M-search algorithm with $M = 2$, and all stage codebooks are optimized jointly. The conditioning scheme used is the one described in the previous experiment and illustrated in Fig. 2. The BFOS algorithm is again applied to a balanced tree with depth 10, where the maximum number of conditional probabilities is set to 4096. As can be seen from Table II, both CEC-RVQ's outperform the other finite-state and predictive VQ techniques. In addition, CEC-RVQ generally requires less

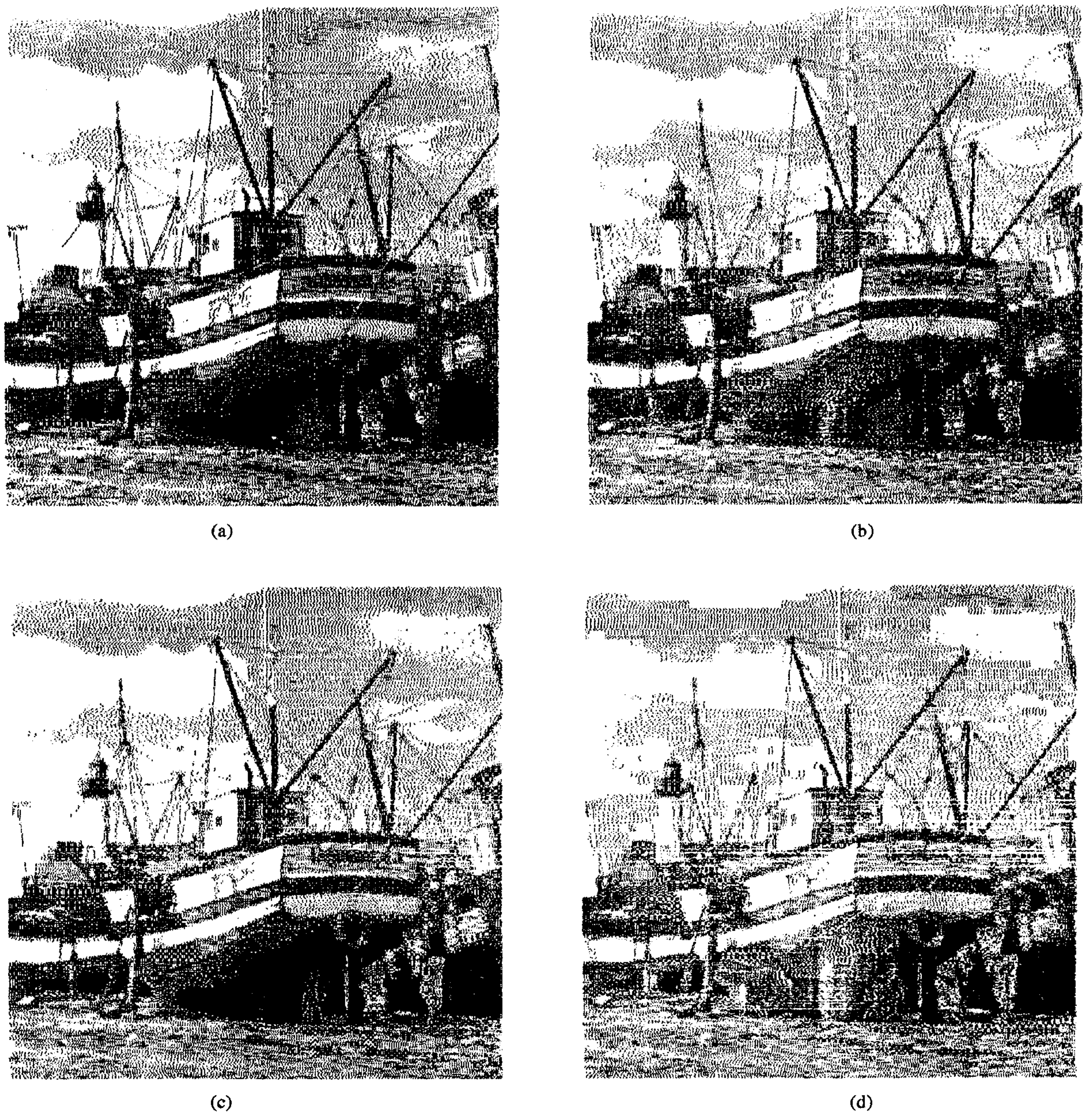

Fig. 8. Image BOAT coded at 0.18 b/pixel using (b) CEC-RVQ, (c) EC-RVQ, and (d) OPTIMIZED JPEG. The PSNR (in dB) is 29.74 for CEC-RVQ, 29.18 dB for EC-RVQ, and 27.46 dB for OPTIMIZED JPEG.

design complexity, encoding complexity, and memory.

We next consider using 8×8 vectors in the design of CEC-RVQ. Each codebook contains 20 stage codebooks. Each stage codebook contains four code vectors. M-search with $M = 2$ is used in encoding, and all stage codebooks are optimized jointly. Using the same conditioning scheme, the same region of spatial-stage support, and the BFOS algorithm applied to a balanced tree of depth 6 with a maximum of 1024 conditional probabilities, we obtain the results shown in Table III. By adding the $\mathcal{N}_p$'s, we obtain 1016, which is the actual number of conditional probabilities. Note that unlike the 4×4 case, interstage dependencies are here significantly stronger than spatial dependencies. In other words, more dependencies exist between stages representing the same image block than between stages representing adjacent image blocks. This is expected because when the image block size increases to 8×8, it is generally true that intrablock correlation increases, and interblock correlation decreases.

Since each symbol represents an 8×8 vector, a larger number of pixels is used in the conditioning, which leads to significant gains in rate-distortion performance. To illustrate this, Fig. 8 shows the BOAT image coded with CEC-RVQ, EC-RVQ, and optimized JPEG (using public JPEG software with -optimize). The image coded with CEC-RVQ is better than its EC-RVQ counterpart [9] subjectively and objectively, and both are better than the image coded with optimized JPEG. The PSNR at 0.18 b/pixel is 29.74 dB for CEC-RVQ, 29.18 dB for EC-RVQ, and 27.46 dB for optimized JPEG. The performance gain obtained by CEC-RVQ over EC-RVQ comes at the expense of some additional design complexity. However, the memory and encoding complexity of CEC-RVQ are actually smaller than those of EC-RVQ. More specifically,

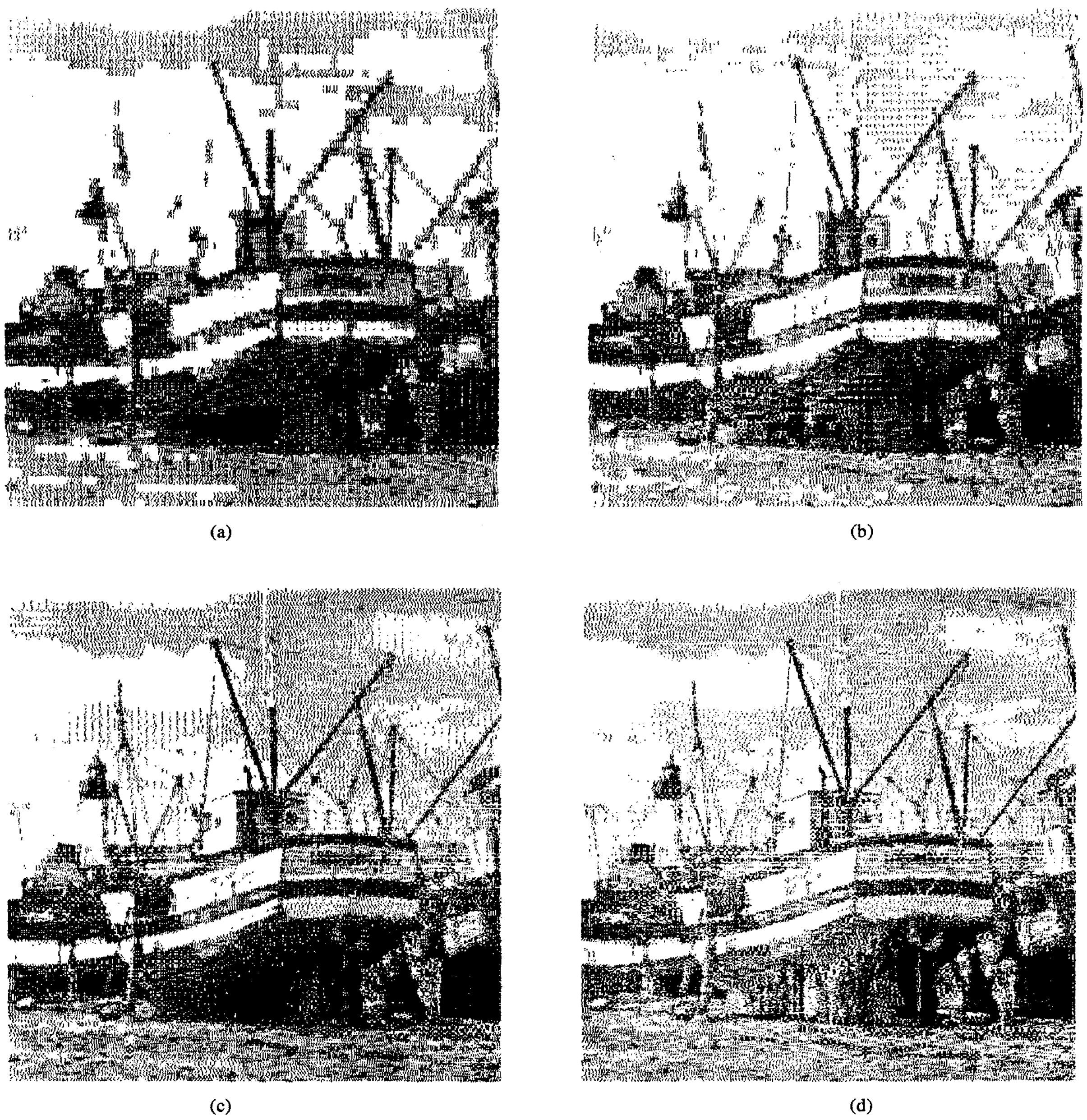

Fig. 9. Image BOAT coded using the progressive transmission CEC-RVQ at (a) 0.0161 bpp, (b) 0.045 bpp, (c) 0.117 bpp, and (d) 0.18 bpp. The PSNR's (in dB) are 21.88, 23.84, 26.86, and 29.08, respectively.

→ the EC-RVQ described in [9] requires the storage of 128 stage code vectors per codebook and 1808 conditional probabilities per set of entropy tables. The numbers for the CEC-RVQ designed here are 80 and 1024, respectively. Moreover, CEC-RVQ requires only 156 vector Lagrangian calculations per input vector, whereas 240 of them are needed for EC-RVQ. Comparing CEC-RVQ with JPEG, we first observe a large image reproduction quality difference in favor of CEC-RVQ. In terms of implementation costs, JPEG's decoding complexity is slightly higher than that of EC-RVQ and CEC-RVQ, but its encoding complexity and memory are substantially smaller.

Finally, the same 8×8 CEC-RVQ described above was modified and tested in a full-resolution progressive transmission environment. More specifically, the conditioning structure of Fig. 2 was replaced with the one in Fig. 5, sequential search was used in encoding, and each stage codebook was optimized sequentially, as was first described in [6]. Interestingly, the encoding/decoding complexity and memory of the fully embedded CEC-RVQ coder are smaller. In particular, only 80 vector Lagrangian calculations per input vector are now required to produce the indices for the final reconstructed image.

For a subjective comparison, Fig. 9 is provided. It shows the test image BOAT coded at (a) 0.0161 b/pixel, (b) 0.045 b/pixel, (c) 0.117 b/pixel, and (d) 0.18 b/pixel. Notice that only the first two stage indices were decoded in Fig. 9(a) (fast decoding), but the image can still be recognized. Moreover, although the PSNR obtained for the full-resolution reconstructed image is 0.66 dB smaller than that of nonprogressive CEC-RVQ, the visual quality is similar.