The Asymptotic Closed-Loop Approach to Predictive Vector Quantizer Design with Application in Video Coding

Hosam Khalil, *Student Member, IEEE*, Kenneth Rose, *Member, IEEE*, and Shankar L. Regunathan

***Abstract*—The basic vector quantization (VQ) technique employed in video coding belongs to the category of predictive vector quantization (PVQ), as it involves quantization of the (motion compensated) frame prediction error. It is well known that the design of PVQ suffers from fundamental difficulties, due to the prediction loop, which have an impact on the convergence and the stability of the design procedure. In this paper we propose an approach to PVQ design that enjoys the stability of open-loop design while it ensures ultimate optimization of the closed-loop system. The method is derived for general predictive quantization, and we demonstrate it on video compression at low bit rates, where it provides substantial improvement over standard open and closed loop design techniques. Further, the approach outperforms standard DCT-based video coding.**

***Index Terms*—Predictive vector quantization, video coding.**

I. Introduction

MOST video coding systems use predictive coding and are composed of two main functional modules: the frame prediction module, and the prediction error (residual) compression module (see Fig. 1). The objective of the first module is to exploit the temporal redundancy that exists between consecutive frames by predicting the contents of the current frame from the previous frame. Block-based motion compensation (whose parameters are transmitted as side-information) is used in this module to achieve better approximation of the current frame. The second module is the lossy part of the codec where the prediction error, or residual, is compressed to the appropriate bit rate.

The prediction residual is usually handled as a two-dimensional (2-D) signal and, more specifically, as if it were a still image. The predominant residual compression approach involves application of the discrete cosine transform (DCT), and this is the method of choice in the major standards including H.263 [1] and MPEG [2]. An important justification for the use of DCT in still image compression hinges on the assumption that the signal can be well modeled as a Gauss–Markov process with a high autocorrelation coefficient. It has been shown that the performance of the optimal (Karhunen–Loeve) transform on such a signal is closely approximated by that of DCT [3]. Wavelet and subband decomposition have also been proposed for coding the prediction error residual [4], [5]. The success of these techniques in compression of images is due to their decorrelation and energy compaction properties.

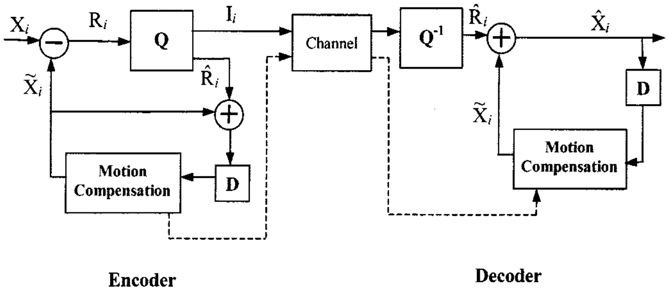

Fig. 1. Basic predictive video coding system.

However, the above arguments which build on statistical characteristics of still images, do not hold for the prediction residual of video signals whose statistical characteristics are considerably different. In fact, it may be argued that once an effective motion compensation is performed, the remaining residual exhibits too little correlation to warrant further application of a decorrelating transform. It is, therefore, plausible that an approach to direct compression of the residual, which takes into account the actual signal statistics, would provide substantial gains. Interesting alternatives to mainstream DCT-based coders are based on matching pursuits [6] and wavelets [7], [8].

We pursue a known alternative approach for direct compression of prediction residuals, which is based on vector quantization (VQ). There has been prior work on vector quantization of the residual [9]–[11]. In [9], a variable block size scheme is used where the motion compensated residual is divided into blocks of varying sizes to suit activity levels. In [10], the randomness of prediction residuals was considered, and a stochastic VQ scheme was proposed. Simulation results in these papers were not compared to standards. Other schemes were proposed in [12] but performance was reported to be inferior to the then-current standard.

There are several arguments in support of VQ for video compression. Shannon's theory implies that vector quantizers are asymptotically optimal, where asymptotic here is in terms of vector length. (Note, in particular, that typical block sizes in video coding correspond to long vectors.) Another important argument is that VQ is a very general framework and subsumes, for example, DCT compression as a special constrained case

Manuscript received June 16, 1999; revised July 24, 2000. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Rashid Ansari.

The authors are with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106 USA (e-mail: rose@ece.ucsb.edu).

Publisher Item Identifier S 1057-7149(01)00109-9.

1057–7149/01$10.00 © 2001 IEEE

[13]. Thus, it may be argued that DCT cannot outperform the best VQ. On the other hand, there exist various objections to the use of VQ in video coding. One major difficulty is that of complexity. The VQ complexity grows exponentially with the product of vector dimension and rate. Structurally-constrained VQ methods have reduced search and/or memory complexity, but their performance is inevitably compromised. However, it is important to note that for very low bit rate compression, which is our target area, even unconstrained VQ would be manageable. Another major objection is concerned with difficulties in the design of VQ for video coding applications. Predictive VQ (PVQ) design is problematic, and the design often fails to produce an optimal (and often even a good) VQ.

It is our premise here that suboptimal PVQ design is a major stumbling block on the way to a truly competitive VQ approach for video coding. We hence propose to attack this fundamental problem. We first review traditional design methods and explain the difficulties in the training procedure (Section II). We develop a novel approach to solve the PVQ design problem in Section III. An extension to the design of entropy constrained PVQ is outlined in Section IV. In Section V, we provide simulation results as experimental evidence that PVQ is indeed an attractive approach for video coding.

II. Conventional Predictive Vector Quantizer Design

A major issue in PVQ design involves the need to obtain a stable training set that accurately represents the true signal statistics. To clarify this difficulty, consider the design of a regular VQ system, where the quantizer directly encodes source samples. It is possible via the generalized Lloyd algorithm (GLA [14]) to iteratively adjust the quantizer parameters while decreasing the distortion, as computed over the training set, until convergence. In contrast with standard VQ, the PVQ system operates on the prediction error. But since the prediction is based on the reconstruction of past samples (previous reconstructed frame in the case of video), the prediction error depends on the quantizer itself. Clearly, the “effective training set” which is the sequence of prediction errors, is not fixed but changes every time the quantizer parameters are modified. In [15], two techniques were introduced for PVQ design and have been widely used since. In this section, we briefly sketch these approaches. The presentation is geared toward emphasizing the unresolved issues, and highlighting points of distinction with respect to the approach proposed in this paper.

A. Open-Loop Approach

This simple approach is depicted in Fig. 2. A training set of prediction error vectors is generated by using the original, *unquantized* source vectors for prediction. It is called “open-loop” (OL) because the reconstructed vectors are not fed back through the predictor. Specifically, given a set of original samples, X: $\{x_0, x_1, x_2, \cdots, x_N\}$, we generate the required training set via

$$t_n = x_n - P[x_{n-1}], \qquad n = 1, 2, \cdots, N \tag{1}$$

where P is the prediction operator.

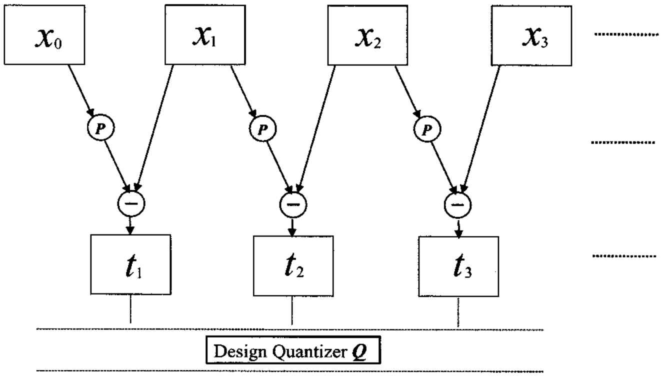

Fig. 2. Open-loop procedure: x_n denotes original sample n and t_n denotes prediction error n. P represents the predictor operator.

The main advantage of the OL approach is that the training set, T: $\{t_1, t_2, \cdots, t_N\}$ is fixed. Therefore, we can design the PVQ by applying a standard optimization technique such as GLA. Since the training set remains unchanged, the design algorithm is guaranteed to converge to at least a locally optimal solution.

However, the OL approach suffers from a serious shortcoming. The decoder does not have access to the original source vector for prediction. Therefore, during the actual operation of the compression system, prediction is performed using reconstructed source vectors. Thus, the training set of prediction errors is statistically different from the prediction errors to be quantized in practice. The statistical mismatch, which is further amplified by feedback through the prediction loop, leads to poor performance.

B. Closed-Loop Approach

To alleviate the statistical mismatch problem of the OL method, a closed-loop approach, denoted by CL, was presented in [15]. Fig. 3 shows the main steps. Here, a closed-loop (real) system is used to generate the prediction errors in an iterative fashion. Given a quantizer at iteration $i-1$, which we denote by $Q^{(i-1)}$, a training set of prediction errors $T^{(i)}$: $\{t_1^{(i)}, t_2^{(i)}, \cdots, t_N^{(i)}\}$, is generated for iteration i

$$t_n^{(i)} = x_n - P\left[\hat{x}_{n-1}^{(i)}\right] \tag{2}$$

where

$$\hat{x}_n^{(i)} = P\left[\hat{x}_{n-1}^{(i)}\right] + Q^{(i-1)}\left(x_n - P\left[\hat{x}_{n-1}^{(i)}\right]\right). \tag{3}$$

Equations (2) and (3) are *sequentially* calculated for each $n = 1, 2, \cdots, N$. For this set of prediction errors, a new quantizer, $Q^{(i)}$, is optimized. Next, a new sequence of prediction errors is generated for iteration $i+1$, and so on.

The initial quantizer $Q^{(0)}$ is usually chosen to be the outcome of the OL method. Since the training residuals were generated by the same closed-loop coder that will be used in the actual mode of operation, the input residual error statistics are expected to be similar to those used to train the quantizer. However, convergence of the algorithm is not guaranteed, as the training set

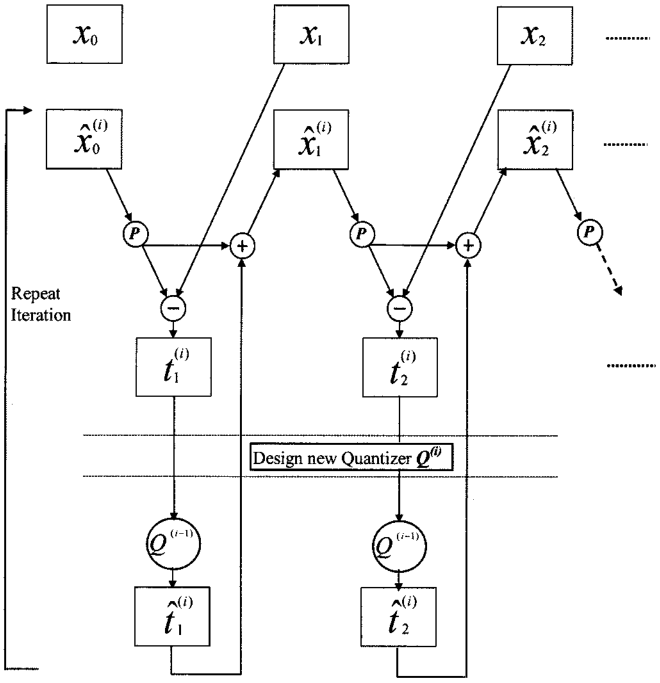

Fig. 3. Closed-loop procedure: x_n denotes original vector n, $\hat{x}_n^{(i)}$ denotes the nth reconstructed vector at iteration i, and $t_n^{(i)}$ denotes the nth prediction error at iteration i. $Q^{(i)}$ is the vector quantizer trained on prediction error sequence from iteration i, and $\hat{t}_n^{(i)}$ is $t_n^{(i)}$ quantized by $Q^{(i-1)}$. The design of $Q^{(i)}$ cannot proceed until all $t_n^{(i)}$, for $n = 1, 2, \cdots, N$, have been collected and the newly designed $Q^{(i)}$ will only be used in the next iteration $i + 1$.

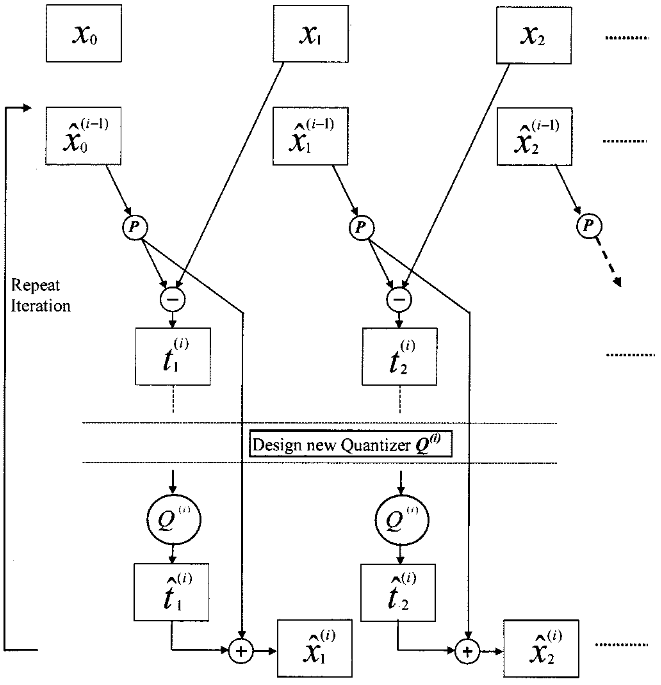

Fig. 4. Proposed ACL procedure: x_n denotes original vector n, $\hat{x}_n^{(i)}$ denotes the nth reconstructed sample at iteration i, and $t_n^{(i)}$ denotes the nth prediction error at iteration i. $Q^{(i)}$ is the vector quantizer trained on prediction error sequence from iteration i, and $\hat{t}_n^{(i)}$ is $t_n^{(i)}$ quantized by $Q^{(i)}$. Note that the newly designed $Q^{(i)}$ is used in the same iteration i to generate new reconstructed vectors in preparation for the next iteration $i + 1$. The main difference between this design and the CL design is that there is NO FEEDBACK; quantized prediction error is not fed back into the closed-loop system.

changes every iteration in an unpredictable fashion. The instability of the CL method is amplified at very low bit rates as will be demonstrated in the results section.

A notable alternative closed-loop method is the stochastic approach of Chang and Gray [16]. Another approach is the more recent constrained optimization of Rizvi and Nasrabadi [17]. However, it is generally known that the problem has not been satisfactorily solved as yet [11], [13], [18].

C. Summary of Shortcomings

The relative merits of the CL design versus the OL design are not clear. Although OL training has a fixed training set, and hence is ensured to converge, it is mismatched with the true mode of operation of the quantizer. On the other hand, the central design difficulty of the CL training technique is that quantization errors are fed back through the prediction loop, thus making the training of the quantizer a highly unstable procedure. In particular, the actual effective training set (the sequence of prediction errors) of CL is encoded by a quantizer optimized for the training set of the previous iteration. Due to the feedback loop, the effect of this mismatch builds up to large deviation in the statistics, and tends to confuse the design procedure. CL training "ignores" the above difficulty and iterates as if an improvement of the quantizer for the current set of prediction errors ensures better performance on the prediction errors of the next iteration.

III. Proposed Method: Asymptotic Closed-Loop Approach

The objective of the proposed design approach is to enjoy the best of both worlds, namely, to enjoy the design stability of the open-loop mode while ultimately optimizing the system for closed-loop operation. To achieve this, we propose the following procedure (illustrated in Fig. 4).

Let us first introduce some mathematical notation to facilitate the algorithm description. The main objective is to avoid accumulation of errors due to mismatched quantization through the prediction loop. We therefore base our prediction on the reconstructed samples of the *previous iteration*. The training set is, in effect, generated by

$$t_n^{(i)} = x_n - P\left[\hat{x}_{n-1}^{(i-1)}\right], \qquad n = 1, 2, \cdots, N. \qquad (4)$$

.

Having collected the set of training samples, we optimize a new quantizer $Q^{(i)}$ (via GLA). The new quantizer is then used to generate the new set of reconstruction samples based on

$$\hat{x}_n^{(i)} = P\left[\hat{x}_{n-1}^{(i-1)}\right] + Q^{(i)}\left(x_n - P\left[\hat{x}_{n-1}^{(i-1)}\right]\right), \quad n = 1, 2, \cdots, N. \qquad (5)$$

Compare (4) with equation (1) for standard open-loop, and (2) for the closed loop design. The CL design alternates between equations (2) and (3) with every n before moving to the next iteration. In our approach, execution of (4) is done for the *entire* sequence, without the effect of quantization error accumulation.

We then calculate (5) for *all* n, before moving on to the next iteration.

Note that the quantizer $Q^{(i)}$ is used to encode *exactly* the same prediction error vectors used for its design. Neglecting the possible local-optimality of the quantizer design algorithm, this is the best quantizer for these vectors. We are thus assured that the resulting reconstruction is improved, and this results in better prediction. Under the reasonable (and common) assumption that smaller prediction errors lead to smaller quantization error, and vice versa, we obtain monotonic improvement throughout the process.

Note that the entire design is in open-loop mode since we compute prediction errors for the entire sequence before quantization. As the distortion is generally decreasing, we expect the process to *converge*. At convergence, further iterations do not modify the quantizer

$$Q^{(i+1)} = Q^{(i)} \tag{6}$$

which immediately ensures that the reconstruction sequence is fixed

$$\hat{x}_n^{(i+1)} = \hat{x}_n^{(i)} \tag{7}$$

and that the next-frame prediction sequence is fixed

$$P\left[\hat{x}_{n-1}^{(i)}\right] = P\left[\hat{x}_{n-1}^{(i-1)}\right]. \tag{8}$$

This implies that the prediction would be unchanged if it were based on the reconstruction of the current iteration, instead of on the reconstruction from the previous iteration. In other words, the procedure is *asymptotically equivalent* to closed-loop design. But the algorithm is running at all times in open loop! We thus have developed a procedure which is "open-loop" in nature, yet converges to optimization of the closed-loop performance. We hence refer to this approach as the asymptotic closed-loop (ACL) approach.

The algorithm for ACL design of PVQ can be summarized as follows.

Step 1) Apply an initial PVQ to the training sequence of source samples to obtain a reconstructed sequence, with the corresponding sequences of next-sample prediction, and prediction error.

Step 2) Design an optimal VQ for the current sequence of prediction errors.

Step 3) Apply the current VQ to quantize the prediction errors used in its design.

Step 4) Add the sequence of quantized prediction errors to the next-sample-prediction sequence to obtain a new reconstructed sequence.

Step 5) Apply prediction to the reconstructed sequence to generate a next-sample-prediction sequence.

Step 6) Subtract the prediction sequence from the original sequence to generate the new sequence of prediction errors.

Step 7) Go to Step 2.

As our assumption (better quantization results in better prediction and vice versa) is not perfectly valid, in our experiments, the algorithm terminates in a small limit cycle instead of perfect convergence. However, this appears to have no practical significance (more on this issue in the results section).

IV. Variable Rate Predictive Vector Quantizer Design

For simplicity and clarity, the main ideas have been presented so far in the context of fixed-rate PVQ design. As the target application in the experimental part of the work is low bit rate video coding, we must account for variation in local signal statistics. It is well known that variable rate coders can adapt to changing statistics, and offer higher compression efficiency than fixed rate coders. Hence, we propose to design a variable rate PVQ system for video compression. The design algorithm described above can be easily adapted for this purpose by incorporating in Step 2 an appropriate technique for optimizing entropy-constrained quantizers. An entropy-constrained optimization produces variable length codewords, where code vectors of higher probability are assigned shorter codewords, so as to minimize the expected rate.

The standard optimization technique for entropy-constrained vector quantizers consists of a known modification of GLA which we will refer to as entropy-constrained GLA (EC-GLA) [19]. A Lagrangian formulation is employed, where the cost of encoding is a function of distortion and encoding rate: $L = D + \lambda R$. The Lagrangian multiplier, λ, controls the rate-distortion trade-off. The standard EC-GLA starts with a fixed rate ($\lambda = 0$) codebook and modifies it into a variable rate codebook by increasing λ in a series of steps. There are two drawbacks to standard EC-GLA. The computational complexity can be considerable if the codebook is large. Further, the final codebook heavily depends on the initialization, and the optimization may easily get stuck in a poor local minimum.

To reduce the complexity of EC-GLA, the pairwise nearest neighbor (PNN) algorithm [20] was extended to entropy constrained quantizer optimization by Finamore *et al.* [21] This design procedure uses the entire training set as the initial codebook, and recursively merges the pair of reproduction vectors that yields the least increase in distortion, until the desired codebook size is reached. The reduction in complexity is normally achieved at the cost of some degradation in performance of the codebook.

We instead propose a selective splitting approach to improve the optimization of entropy constrained quantizers. Our objective is two-fold: 1) *directly* optimize the codebook to operate at the desired rate/distortion tradeoff, in contrast to [19], and thereby reduce complexity and 2) improve the initialization. In a logical reverse of PNN, selected codevectors are recursively split. The splitting mechanism is closely related to the greedy splitting approach of Riskin and Gray [22], which was developed for the design of tree-structured VQ. However, we use splitting as a means to improve the initialization and reduce complexity, and not for imposing a structure on the solution. The selective splitting approach was used by us for designing standard variable rate quantizers and was found to outperform standard EC-GLA [23]. Here, it is applied to our primary objective of designing optimal variable rate PVQ for video compression. At the end of this section, we explain the additional advantages offered by selective splitting for PVQ design.

We now summarize the selective splitting approach for designing entropy-constrained quantizers.

Selective Splitting Approach:

Step 1) Initialize the codebook to contain only the centroid vector of the entire training set.

Step 2) For all entries c_i of the codebook, test for the cost effectiveness of a potential split by calculating

$$\Delta L_i = \Delta D_i - \lambda \Delta R_i \tag{9}$$

where

ΔL_i decrease in Lagrangian cost;
ΔD_i decrease in distortion;
ΔR_i increase in rate.

Step 3) Sort and list all entries of the codebook in decreasing order of ΔL_i.

Step 4) Starting at the top of the sorted list, split codewords one by one until a specified criterion is met. Insert the new codewords into the codebook.

Step 5) Given the codebook, run EC-GLA over the entire training set.

Step 6) If target codebook size is achieved, stop. Otherwise, go to Step 2.

Comments and Observations:

- In Step 2, we first calculate the distortion D_i of the training subset T_i associated with codevector i. Then we split the codevector into two new vectors, and apply GLA to T_i producing two training subsets T_i' and T_i''. For these two subsets, we calculate the corresponding distortions D_i' and D_i''. We then evaluate $\Delta D_i = D_i - (D_i' + D_i'')$. If the number of vectors in T_i is N_i, we assume (for simplicity) that we need an extra bit for each vector in T_i resulting in a rate increase of $\Delta R_i = N_i$ bits.
- A "healthy" split is ensured by testing for ΔL_i. Clearly, a negative ΔL_i indicates a counter-productive split, while large positive values of ΔL_i indicate advantageous splits.
- The complexity of testing for a codeword split is not excessive, as the training samples considered are only the subset associated with the codevector. The number of splits per iteration of Step 4 can be either predetermined, or vary depending on the current versus target codebook size. In general, the number of splits per iteration determines the tradeoff between quality and complexity.
- Step 5 allows rectification of "near-sighted" or overly greedy splits of individual codewords, as the whole training set is reconsidered. Usually, a couple of iterations are sufficient to ensure convergence.
- We reemphasize that selective splitting is used in Step 2 of the PVQ design algorithm for optimizing the entropy-constrained PVQ (ECPVQ) codebook. An important advantage of this algorithm for ECPVQ design is that it facilitates execution of Step 2 of the ACL method. When a new $Q^{(i)}$ is designed, we use the $Q^{(i-1)}$ as an initialization to speed up the algorithm. Some of its entries may become unused, and the selective splitting approach naturally solves this problem by dropping those unused vectors, and creating new ones by splitting existing codevectors.

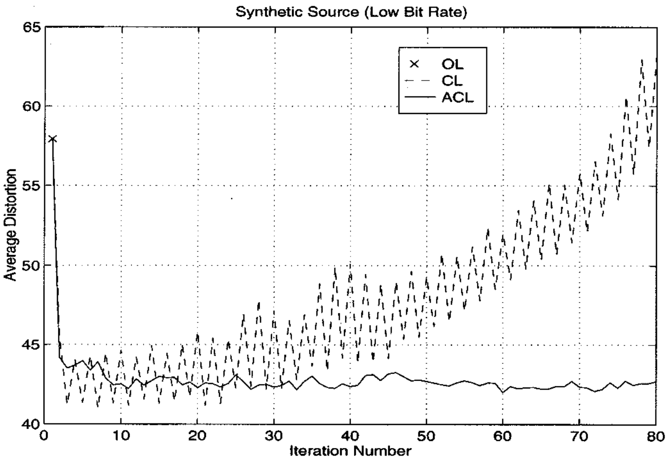

Fig. 5. Performance comparison of standard OL and CL designs and the proposed ACL approach to PVQ design at *low bit rate*. Average distortion over the synthetic first-order Gaussian–Markov training sequence is shown for the PVQ available at the end of each iteration. Both designs start from the same initial point using the outcome of OL design. Note how the CL design improves output initially but becomes unstable after a few iterations. The ACL design remains stable throughout all iterations.

V. Simulation Results

The proposed PVQ design is tested in the context of both synthetic sources and very low bit rate video coding.

A. Experiments on Synthetic Sources

A synthetic source process of first-order Gauss–Markov vectors was generated with intra-vector and inter-vector correlation coefficients of 0.9. A first-order predictor is used in the PVQ design. We present results for two target bit rates: 1) 0.83 bits/sample (referred to as low bit rate) and 2) 1.5 bits/sample (high bit rate).

Fig. 5 shows the average distortion over the training set and its evolution with the iterations. In the low bit rate case, the CL design becomes very unstable after a few iterations. The performance of the codebooks obtained via ACL design remains stable throughout the design process.

In Fig. 6, we present the corresponding results for the case of high bit rate. Here we note that both the CL and ACL designs are stable (though the CL design displays more pronounced oscillations). These results support our premise that the accumulation of errors is the main cause of difficulties in PVQ design. When the available bit rate is high, the accumulation of error is greatly reduced and thus, the traditional design approach is not severely affected.

It is interesting to note that the curve for the ACL method is much smoother than that of the CL method. Also, at low bit rates, the simple OL design may outperform the CL design if the latter training procedure is allowed to run long enough.

B. Experiments on Video Sequences

In the second set of experiments, we evaluate the proposed PVQ design on video sequences. VQ-based techniques in video coding applications are mainly employed at very low bit rates. It is thus expected that the CL design approach will meet with considerable training difficulties due to accumulation of errors.

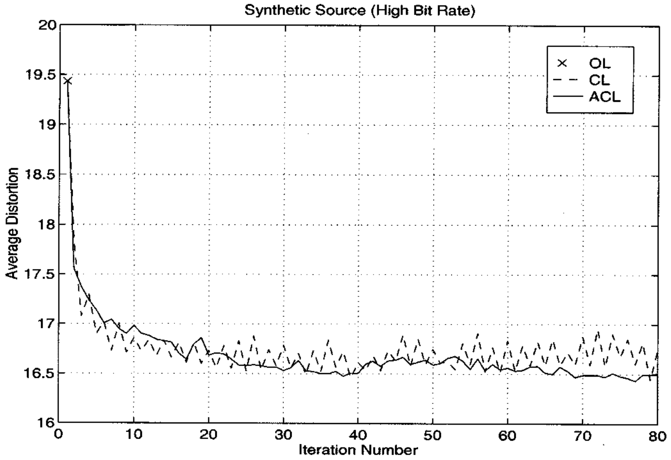

Fig. 6. Performance comparison of standard OL and CL design and the proposed ACL approach to PVQ design at *high bit rate*. Average Distortion over the synthetic first-order Gaussian–Markov training sequence is shown for the PVQ available at the end of each iteration. Both designs start from the same initial point using the outcome of OL design. Notice that in this experiment, high coding rate allows even the CL design to be stable.

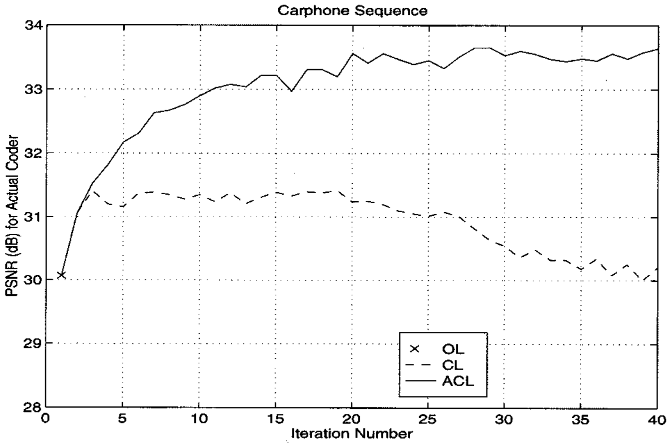

Fig. 7. Performance comparison of standard CL design and the proposed ACL approach to PVQ design. Average PSNR on the training sequence Carphone is shown for the PVQ available at the end of each iteration. Both designs start from the same initial point using the outcome of OL design.

In this section, we compare traditional approaches to the proposed ACL design.

We implemented a video codec where 8 × 8 residual blocks are used as vectors. The video sequences are in QCIF format and the frame rate is ten frames/sec. The general structure of the codec is as shown in Fig. 1. The system uses half-pixel motion compensation, and is basically a "bare-bones" H.263 scheme where the DCT/quantization module was replaced with the ECPVQ, and where each 8×8 block is considered as a separate macroblock. After the first frame, all frames are compressed in interframe mode. This simplification has the sole purpose of focusing the results on predictive coding and eliminating unrelated factors, but all features of H.263 can be readily added to the VQ system. The Lagrange multiplier λ is used to control the rate. In all our simulations, Huffman codes are employed to generate variable length codewords.

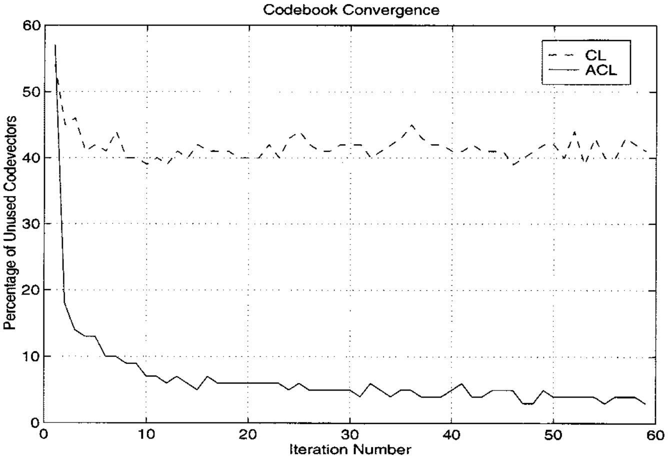

Fig. 8. Percentage of codevectors per codebook that become unused (and thus dropped) on updating the training set. Note that many more vectors in the CL design get dropped in every iteration indicating that its convergence capability is rather limited.

Two main experiments with video have been performed. In the first experiment, a total of 30 frames (luminance component) of the sequence *Carphone* were used as the training sequence. We design an ECPVQ using each of the OL, CL, and ACL techniques described in Section III. Fig. 7 compares the performance of ECPVQ designed by the proposed ACL design method with that of the standard CL design. The PSNR shown is that of the actual closed-loop performance of the coder using an ECPVQ obtained at each iteration and is equal to the average PSNR over the training video sequence. Note that both systems start their iterations by designing an OL-designed codebook, and thus have the same performance at the first iteration. Both systems improve performance in the first few iterations. However, the CL design algorithm leads to gradual accumulation of error in the system and causes the subsequent drop in overall PSNR. On the other hand, the proposed ACL approach shows persistent improvements, and eventually provides performance that is superior by several decibels. For reference, it should be mentioned that the corresponding "bare-bones" H.263 (with the standard DCT module) achieved PSNR of about 31 dB which is significantly below the performance of our ECPVQ. The bit-rate was fixed at about 12 kb/s for transmitting the prediction residual of this QCIF sequence. All other side information (including motion vectors) required rate identical for all coders. It is important to note the instability of the CL algorithm even within the training set. One may notice that the near-monotonic improvement of ACL with iterations is punctuated by occasional drops in PSNR. The reasons for this behavior are the suboptimalities in motion compensated prediction and minor fluctuations in rate.

The effective convergence of the ACL algorithm can be demonstrated as follows. Recall that the codebook of the previous iteration is used as initialization for the design of the codebook of the current iteration. When any of the codebook vectors become unused and are removed, the algorithm will fill up the empty codebook entries by selectively splitting additional vectors. The number of unused codevectors at each iteration is a good indication of how well the codebook is converging. Ideally, as the codebook approaches convergence, a minimal number of vectors will be dropped and updated, i.e., virtually all vectors will be retained. Fig. 8 shows the

TABLE I
PERFORMANCE COMPARISON OF H.263 AND PVQ FOR THE TEST IMAGE SEQUENCES "SALESMAN," "CLAIRE," AND "AKIYO." THE PVQ RESULTS SHOWN ARE FOR THE FOLLOWING: 1) OL METHOD, 2) CL METHOD AFTER ONE ITERATION (CL$^{(1)}$), 3) CL METHOD AFTER TWO ITERATIONS (CL$^{(2)}$), and 4) CL METHOD AFTER COMPLETION OF ITERATIONS (CL$^{(25)}$), AND v) ACL METHOD AFTER COMPLETION OF ITERATIONS. THE COMPARISON IS IN TERMS OF PSNR IN DECIBELS, RATE IN Kb/s, AND THE RATE-DISTORTION LAGRANGIAN PER PIXEL (LOWER VALUES ARE BETTER)

Sequence	Coder	Design	PSNR	Rate	$D + \lambda R$
Salesman	H.263		30.15	9.40	70.07
	PVQ	OL	30.23	8.99	68.73
		CL$^{(1)}$	29.21	9.28	69.29
		CL$^{(2)}$	29.36	9.97	83.20
		CL$^{(25)}$	28.24	17.67	111.47
		ACL	**30.41**	**9.37**	**66.56**
Claire	H.263		34.54	6.57	28.04
	PVQ	OL	34.74	6.04	26.59
		CL$^{(1)}$	34.73	6.28	26.85
		CL$^{(2)}$	34.14	6.38	30.13
		CL$^{(25)}$	32.03	9.46	48.21
		ACL	**35.02**	**6.53**	**25.60**
Akiyo	H.263		32.67	7.04	40.70
	PVQ	OL	32.98	6.81	38.08
		CL$^{(1)}$	32.90	6.89	38.81
		CL$^{(2)}$	31.70	7.27	49.70
		CL$^{(25)}$	30.20	12.89	72.30
		ACL	**33.15**	**7.02**	**37.00**

relative percentage of vectors that become unpopulated at each iteration. The decreasing percentage of unpopulated vectors indicates that the ACL algorithm is converging. Also included in the graph is a demonstration of the instability of the CL approach. It can be seen that almost 40% of the codebook entries become unpopulated in any CL iteration, indicating that further iterations are not producing representative training sets, nor will they lead to convergence of a good codebook.

So far, we have considered the performance on the training set so as to emphasize the power of the proposed ACL optimization technique over conventional ECPVQ design methods. We next present results demonstrating performance outside the training set. In this second experiment, in order for the ECPVQ to be statistically representative, we used a total of 13 video sequences in the training phase. Each video sequence is of length 20 frames. The test set is composed of the three independent (i.e., unused for training) video sequences, namely, *Salesman*, *Claire*, and *Akiyo*, each also of 20 frames. Table I compares the performance of H.263 with that of the various ECPVQ designs. Bit rates shown are those for coding the residual. The ECPVQ design, in this case, involved two codebooks: one codebook optimized for blocks whose motion vector was zero, and another

TABLE II
PERFORMANCE COMPARISON OF H.263 AND PVQ **AVERAGED** OVER ALL THREE *TEST* IMAGE SEQUENCES "SALESMAN," "CLAIRE," AND "AKIYO." THE PVQ RESULTS SHOWN ARE FOR THE FOLLOWING: 1) OL METHOD, 2) CL METHOD AFTER ONE ITERATION (CL$^{(1)}$), 3) CL METHOD AFTER TWO ITERATIONS (CL$^{(2)}$), 4) CL METHOD AFTER COMPLETION OF ITERATIONS (CL$^{(25)}$), AND 5) ACL METHOD AFTER COMPLETION OF ITERATIONS. RESULTS SHOWN HERE ARE SUMMARIZED FROM TABLE I, AND ARE AVERAGE RATE-DISTORTION LAGRANGIAN PER PIXEL OVER ALL THREE TEST SEQUENCES

Coder	Design	Average $D + \lambda R$
H.263		46.27
PVQ	OL	44.47
	CL$^{(1)}$	44.65
	CL$^{(2)}$	54.34
	CL$^{(25)}$	77.33
	ACL	**43.05**

TABLE III
PERFORMANCE COMPARISON OF H.263 AND PVQ **AVERAGED** OVER ALL 13 *TRAINING* IMAGE SEQUENCES. THE PVQ RESULTS SHOWN ARE FOR THE FOLLOWING: 1) OL METHOD, 2) CL METHOD AFTER ONE ITERATION (CL$^{(1)}$), 3) CL METHOD AFTER TWO ITERATIONS (CL$^{(2)}$), 4) CL METHOD AFTER COMPLETION OF ITERATIONS (CL$^{(25)}$), AND 5) ACL METHOD AFTER COMPLETION OF ITERATIONS. RESULTS ARE IN TERMS OF AVERAGE RATE-DISTORTION LAGRANGIAN PER PIXEL OVER ALL 13 TRAINING IMAGE SEQUENCES

Coder	Design	Average $D + \lambda R$
H.263		71.39
PVQ	OL	55.87
	CL$^{(1)}$	55.17
	CL$^{(2)}$	59.58
	CL$^{(25)}$	80.69
	ACL	**38.67**

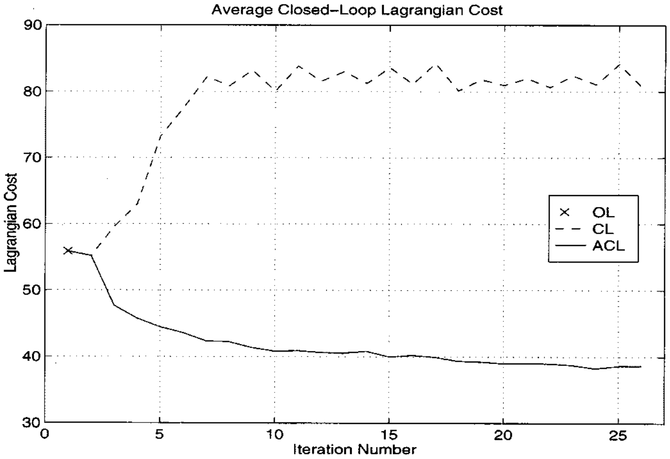

Fig. 9. Performance comparison of standard CL design and the proposed ACL approach to PVQ design. Average Lagrangian cost on the training sequences using the PVQ available at the end of each iteration is shown. Both techniques initially use the outcome of a simple OL design as initialization. Note the gradual decrease in average Lagrangian distortion of the ACL method indicating improvements in the quantizer design. On the other hand, the CL design procedure is unstable.

codebook optimized for blocks with nonzero motion. Note that the switching information need not be conveyed to the receiver as it is determined by the motion. (One can design more codebooks conditioned on the motion vector, but two codebooks seem to represent a reasonable compromise between compression performance, computational complexity, and storage requirements.) Codebook sizes were of about 12 000 and 2000, with average bit rates of 3.5 and 1.5 bits/vector, respectively. The CL and ACL ECPVQ designs were stopped after 25 iterations. The H.263 bit rate was controlled so as to match that of the ECPVQ system designed by the ACL approach.

For the ECPVQ results, we show the system performance in five settings:

1) OL method alone is used;
2) CL method after only one iteration;
3) CL method after two iterations;
4) CL method after the completion of the iterative design;
5) ACL method after completion of the iterative design.

It is worthwhile to note that, in this case, OL can outperform CL on the test set. In fact, CL's instability is such that further iterations are detrimental to its performance. On the other hand, ACL with the exact same initial conditions offers stable performance throughout the iterations, and finally achieves gains of 0.2–0.5 dB over H.263 over the test sequences. Table I also provides the average rate-distortion Lagrangian $D + \lambda * R$. Considerable improvements were obtained in all test sequences. Table II summarizes the results of Table I giving averages over the combined test sequences. We opted to show Lagrangians rather than PSNRs as they can be meaningfully averaged over several different input sequences. For completeness, Table III also provides the Lagrangian averages when each ECPVQ design is used to encode the training sequences. In this case, the ACL design of ECPVQ provides major improvements over H.263. The evolution of the CL and ACL performance with the number of iterations is shown in Fig. 9.

Figs. 10 and 11 give subjective comparisons of a sample frame from the two sequences *Salesman* and *Akiyo*, respectively. While blocking artifacts are clearly visible in the H.263 coded *Salesman* frame, this effect is significantly reduced in the ACL-designed ECPVQ coded frame. For the *Akiyo* frame, it can be seen that some of the fine details of the image are better reproduced by the ECPVQ coder, while H.263 causes blurring of such regions (see, e.g., the earring and the eyes). Note that the complexity of ECPVQ system for low bit rate coding can be reduced by exploiting the fact that a large fraction of residual blocks get quantized to zero. Other fast methods also exist in the literature (see, e.g., [24]).

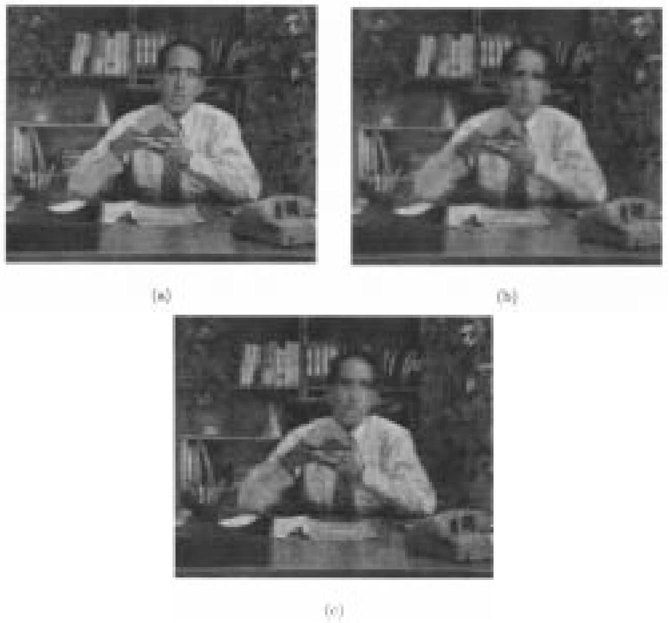

Fig. 10. Subjective comparison of a frame of the *Salesman* sequence: (a) original, (b) H.263, and (c) ECPVQ.

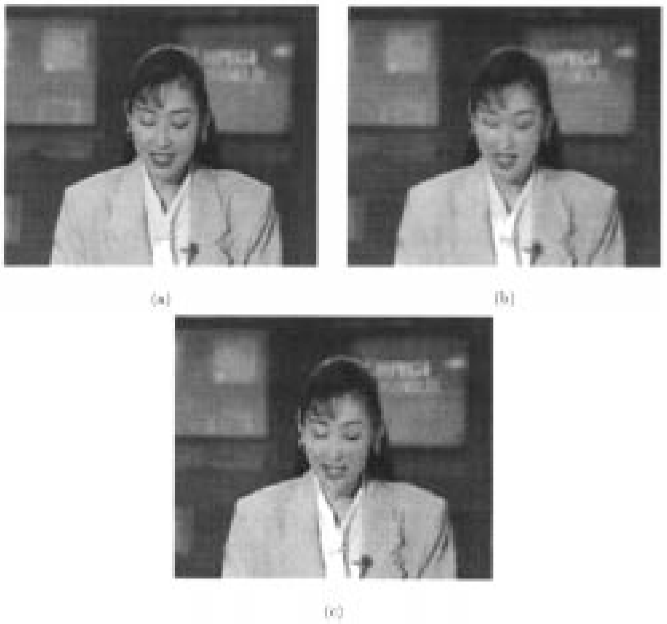

Fig. 11. Subjective comparison of a frame of the *Akiyo* sequence: (a) original, (b) H.263, and (c) ECPVQ.

VI. Conclusion

This paper describes a new approach to training predictive vector quantizers, which does not suffer from the statistical mismatch typical of OL training algorithms, nor from the instability of CL approaches. The proposed iterative ACL algorithm is open-loop in nature but asymptotically optimizes the closed-loop system. Simulation results were presented for a simple ECPVQ system for video coding, and showed the superiority of the proposed design algorithm over conventional approaches. Further results demonstrate that ACL-designed PVQ video compression system outperforms standard DCT-based video coding. It is expected that test set performance will be further improved by more extensive training with longer training sets. An important extension under investigation is that of adaptive PVQ which can exploit variation in local statistics. We are also examining the use of multistage codebooks in the design and use of PVQ to further reduce complexity.

REFERENCES

[1] *Video Coding for Low Bit Rate Communication*, ITU-T Recommend. H.263, 1995.

[2] "Information technology—Generic coding of moving pictures and associated audio information: Video,", MPEG2 ISO/IEC 13818-2, 1995.

[3] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, pp. 90–93, Jan. 1974.

[4] S. P. Voukelatos and J. J. Soraghan, "Very low bit-rate color video coding using adaptive subband vector quantization with dynamic bit allocation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 424–428, Apr. 1997.

[5] D. P. de Garrido, L. Lu, and W. A. Pearlman, "Conditional entropy-constrained vector quantization of displaced frame difference subband signals," in *IEEE Int. Conf. Image Processing '94*, vol. 1, Austin, TX, Nov. 1994, pp. 745–749.

[6] R. Neff and A. Zakhor, "Very low bit rate video coding based on matching pursuits," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 158–171, Feb. 1997.

[7] B.-J. Kim and W. A. Pearlman, "An embedded wavelet video coder using three-dimensional set partitioning in hierarchical trees (SPIHT)," in *Proc. Data Compression Conf.*, Snowbird, UT, Mar. 1997, pp. 251–260.

[8] A. Wang, Z. Xiong, P. A. Chou, and S. Mehrotra, "Three-dimensional wavelet coding of video with global motion compensation," in *Proc. Data Compression Conf.*, Snowbird, UT, Mar. 1999, pp. 404–413.

[9] L. Corte-Real and A. P. Alves, "A very low bit rate video coder based on vector quantization," *IEEE Trans. Image Processing*, vol. 5, pp. 263–273, Feb. 1996.

[10] L. Torres, M. Mora, and J. R. Casas, "Prediction error image coding using a modified stochastic vector quantization scheme," in *Proc. IEEE Int. Conf. Image Processing '96*, vol. 3, Lausanne, Switzerland, Sept. 1996, pp. 451–454.

[11] R. Baker and J. Salinas, "A motion compensated vector quantizer with filtered prediction," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. 2, 1988, pp. 1324–1327.

[12] T. Lookabaugh, E. A. Riskin, P. A. Chou, and R. M. Gray, "Variable rate vector quantizer for speech, image, and video compression," *IEEE Trans. Commun.*, vol. COM-28, pp. 84–95, Jan. 1989.

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

[14] 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.

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

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

[17] S. A. Rizvi and N. M. Nasrabadi, "Predictive vector quantizer using constrained optimization," *IEEE Signal Processing Lett.*, vol. 1, pp. 15–18, Jan. 1994.

[18] H. Kwon, M. Venkatramam, and N. M. Nasrabadi, "Very low bit-rate video coding using variable block-size entropy-constrained residual vector quantizers," *IEEE J. Select. Areas Commun.*, vol. 15, no. 9, pp. 1714–1725, Dec. 1997.

[19] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy constrained vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-37, pp. 31–42, Jan. 1989.

[20] W. H. Equitz, "A new vector quantization clustering algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 1568–1575, Oct. 1989.

[21] W. A. Finamore, D. P. de Garrido, and W. A. Pearlman, "A clustering algorithm for entropy-constrained vector quantizer design," in *Proc. SPIE 1360, Visual Commun. Image Processing '90*, Lausanne, Switzerland, Nov. 1990, pp. 837–846.

[22] E. A. Riskin and R. M. Gray, "A greedy tree growing algorithm for the design of variable rate vector quantizers," *IEEE Trans. Signal Processing*, vol. 39, pp. 2500–2507, Nov. 1991.

[23] H. Khalil and K. Rose, "Selective splitting approach to entropy-constrained single/multi-stage vector quantizer design," in *Proc. SPIE Conf. Electronic Imaging*, San Jose, CA, Jan. 2000.

[24] M. H. Johnson, R. Ladner, and E. A. Riskin, "Fast nearest neighbor search for ECVQ and other modified distortion measures," in *Proc. IEEE Int. Conf. Image Processing '96*, vol. 3, Lausanne, Switzerland, Sept. 1996, pp. 423–426.

Hosam Khalil (S'98) Hosam Khalil was born in Washington, DC, in 1972. He received the B.S. and M.S. degrees in electrical engineering with honors from Cairo University, Cairo, Egypt, in 1993 and 1996, respectively. He is currently pursuing the Ph.D. degree at the University of California, Santa Barbara.

From October 1993 to July 1996, he was an Assistant Teacher at Cairo University. He was also employed part-time during that period with IBM, Egypt. In the Summer of 1998, he was an Intern in the Multimedia Communications Research Laboratory, Bell Laboratories, Murray Hill, NJ. From March 1999 to March 2000, he was a Software Engineer at SignalCom, Inc., Goleta, CA. In April 2000, he joined Microsoft Corporation, Digital Media Division, where he is a Software Development Engineer. His research interests are in image, video, and speech compression, and speech recognition.

Mr. Khalil was a Recipient of the UC Regents Fellowship.

Kenneth Rose (S'85–M'91) received the B.Sc. (summa cum laude) and M.Sc. (magna cum laude) degrees in electrical engineering from Tel-Aviv University, Tel-Aviv, Israel, in 1983 and 1987, respectively, and the Ph.D. degree in electrical engineering from the California Institute of Technology (Caltech), Pasadena, in 1990.

From July 1983 to July 1988, he was with Tadiran Ltd., Israel, where he carried out research in the areas of image coding, image transmission through noisy channels, and general image processing. From September 1988 to December 1990, he was a Graduate Student at Caltech. In January 1991, he joined the Department of Electrical and Computer Engineering, University of California, Santa Barbara, where he is currently an Associate Professor. His research interests are in information theory, source and channel coding, image coding and processing, speech and general pattern recognition, and nonconvex optimization in general.

Dr. Rose is Editor for Source/Channel Coding for the IEEE TRANSACTIONS ON COMMUNICATIONS. He was co-recipient of the William R. Bennett Prize Paper Award of the IEEE Communications Society (1990).

Shankar L. Regunathan received the B.Tech. degree in electronics and communication from the Indian Institute of Technology, Madras, in 1994, and the M.S. degree in electrical engineering from University of California, Santa Barbara (UCSB), in 1995. He is currently a graduate student in electrical engineering at UCSB.