Distributed Predictive Coding For Spatio-Temporally Correlated Sources

Ankur Saxena and Kenneth Rose
Dept. of Electrical and Computer Engineering
University of California, Santa Barbara, CA 93106
Email:{ankur, rose}@ece.ucsb.edu

Abstract— **This paper investigates *distributed predictive* coding of correlated sources with memory, which are communicated to a central receiver. This is the setting typically encountered in sensor networks. While source memory may be exploited by distributed coding of large source blocks (vectors), the growth in complexity (and delay) is often unacceptable in practice, hence the interest in a low complexity predictive approach. We first consider the inherent "conflict" between distributed and predictive coding due to the impact of distributed quantization on the prediction loop. This is coupled with the effects of closed loop prediction, which destabilize standard Lloyd-like code design methods. An iterative algorithm is derived, which optimizes the overall system while imposing zero decoder drift due to distributed quantization. The approach circumvents convergence and stability issues of traditional predictive quantizer design by employing an "Asymptotic Closed Loop" framework which is adapted for distributed predictive system design. The scheme efficiently utilizes both the temporal and inter-source correlations and subsumes as extreme special cases both separate source predictive coding, and distributed coding of memoryless correlated sources.**

I. INTRODUCTION

Distributed source coding (DSC) has experienced significant revival in recent years, partly thanks to its relevance to sensor network applications. The basic setting involves multiple correlated sources (e.g., data collected by a number of spatially distributed sensors) which need to be transmitted to a central data collection unit. The main objective of DSC is to exploit inter-source (e.g., spatial) correlations despite the fact that each source must be encoded without access to the other sources (see Fig. 1). As will be briefly summarized below, there is a long history of research on the theoretical and asymptotic limits of such systems, as well as on practical design and optimization. The main motivation for the work presented herein is in the obvious observation that most correlated sources of interest are in fact sources with memory, i.e., they also exhibit temporal correlations. In particular, data collected from sensors will often show significant time correlations that are at least as important as the spatial (inter-source) correlations. We therefore consider the design of distributed *predictive* coding (DPC) systems. Given the historical focus on inter-source correlations, most existing work naturally addressed memoryless sources where one need not worry about temporal correlations.

We shall, however, see that in a DPC system, there are conflicting objectives of distributed coding versus efficient prediction. In other words, optimal distributed quantization (in terms of current reconstruction quality) may severely compromise the prediction loop at each source encoder. This work is concerned with the theoretical and practical problems that arise once one strives to efficiently exploit both temporal and inter-source correlations while optimizing the inherent tradeoffs between the two.

The field of DSC began in the seventies with the seminal work of Slepian and Wolf [11]. They showed, in the context of lossless coding, that side-information available only at the decoder can nevertheless be fully exploited as if it were available to the encoder, in the sense that there is no asymptotic performance loss. Later, Wyner and Ziv [13] extended the result to bound the performance of lossy coding with decoder side information. In the late nineties, constructive and practical code design techniques for distributed coding were proposed, notably by Pradhan and Ramchandran in their DISCUS approach [7]. The field eventually saw the emergence of various distributed coding techniques, most with an eye towards sensor networks.

DSC techniques can be categorized into two "camps", the one adopting channel coding ideas, some of which exploit long delays to achieve good performance, and the other builds directly on source coding methodologies. The source coding perspective will be most relevant to us here. Design of vector quantizers for distributed coding has received some attention in recent years. Algorithms for distributed vector quantizer design have been proposed in [1], [2], [8]. The high sensitivity of these algorithms to initialization and their susceptibility to poor local optima was addressed in [10], and an extension to robust DSC (accounting for possible channel failure) was proposed in [9]. One may naturally account for time correlations by blocking sources into large vectors for quantization, but such a strategy suffers from severe complexity problems and extremely exacerbates the sensitivity to initialization and poor local optima [10], [12]. Motivated by this observation, a notable approach to predictive coding of correlated sources was proposed by Tuncel in [12] where a uniform quantization grid was imposed on the product space (across sources) of prediction errors, on which the main support of the joint distribution was identified and a DSC code devised. The emphasis in that paper's results was on the design of optimal predictor filters in such distributed setting and on how they deviate from the case of non-distributed predictive coding. Our starting premise is different and we propose a DPC system which seeks

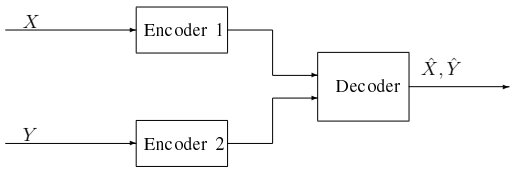

Fig. 1. Distributed coding of two correlated sources

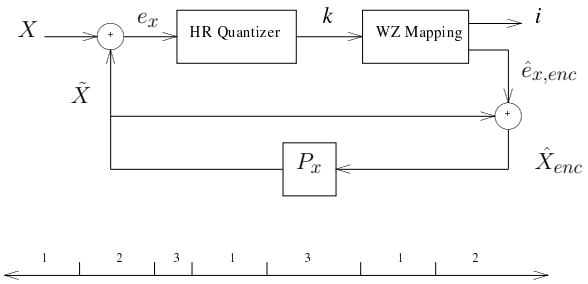

Fig. 2. Block diagram of a DPC encoder and a scalar example of WZ mapping from prototypes (Voronoi regions) to indices.

to jointly optimize predictive and distributed coding and is derived without recourse to any assumptions on the sources, nor restriction of the quantizer to a predefined grid. Our DPC coder design algorithm specializes to traditional predictive coding and memoryless distributed coding as extreme special cases.

A design difficulty whose origins are in standard predictive quantizer design [3] is exacerbated in the distributed setting. On the one hand, open-loop design is simple and stable but the quantizer is mismatched with the true prediction error statistics (as the system operates in closed loop.) On the other hand, closed-loop design is unstable and may not converge since the training set of prediction errors keeps changing as the quantizer (or predictor) is updated. To circumvent these difficulties we use the asymptotic closed loop (ACL) idea {[4], [5]} which we derive for DPC system design. Within the ACL framework, DPC system design operates in open-loop within iterations but asymptotically, the prediction error statistics converge to closed loop statistics, hence the prediction loop is effectively closed at convergence.

The rest of the paper is organized as follows. In Section II, we state the problem formally, introduce notation and specify the components of the DPC system. Section III provides an overview of ACL and presents the iterative algorithm for DPC design. Simulation results are summarized in Section IV.

II. DISTRIBUTED PREDICTIVE CODING

A. Preliminaries

Consider the distributed source coding scenario of Fig. 1. For brevity, but without loss of generality, we restrict the presentation to two sources. Here X and Y are two continuous valued, spatio-temporally correlated (possibly vector) sources, i.e, in addition to the spatial correlation between the sources, each source itself has some memory. The two source encoders compress and transmit source information at rates R_1 and R_2 bits per sample, respectively. The decoder may reconstruct either or both sources. The objective is to minimize the following expected distortion:

$$E\{\alpha d(X,\hat{X}) + (1-\alpha) d(Y,\hat{Y})\}, \qquad (1)$$

where $d(\cdot,\cdot)$ is an appropriately defined distortion measure, $\hat{X}$ and $\hat{Y}$ are the reconstruction values for X and Y, respectively and $\alpha \in [0,1]$ accounts for the relative importance of the sources at the decoder. We further assume that predictive coding is employed to exploit temporal redundancies (We will restrict the scope to linear prediction.).

The DPC encoder for X is shown in Fig. 2. Note that prediction errors e_x (for X) and e_y (for Y) are correlated. Therefore, instead of the usual predictive quantizer, we need to design a distributed coder to exploit inter-source correlations.

For the distributed coding modules we borrow the specific notation from [9]. Let us temporarily assume a fixed training set $\mathcal{T} \equiv \{e_x, e_y\}$. (Of course predictive coding will necessitate modifications to this assumption). High resolution quantizer Q_x assigns each training sample (prediction error) to one of the K regions, C_k^x. The regions C_k^x form a Voronoi partition of the e_x source space and each is represented by a prototype e_k^x. Region C_k^x is mapped to one of I indices, via the mapping $v(k) = i$, which we refer to as *"Wyner-Ziv" (WZ) mapping*, since the mapping exploits inter-source correlations within a lossy coding system. The index i is transmitted to the central unit (decoder). An example of WZ mapping with K = 7 and I = 3 is given in Fig. 2. The region associated with an index i is $R_i^x = \bigcup_{k;v(k)=i} C_k^x$. We similarly define the quantizer Q_y, regions C_l^y, R_j^y and prototypes e_l^y in the Y domain. Here, the L Voronoi regions are mapped to J indices via WZ mapping $w(l) = j$.

The joint decoder receives an index pair (i,j) to generate reconstruction values $\hat{e}_x$ and $\hat{e}_y$, and calculates $\hat{X}$ and $\hat{Y}$. We next explain the functioning of the distributed predictive coder.

B. Distributed Predictive Encoder

The DPC encoder for source X is depicted in Fig. 2. The input to the high resolution quantizer is $e_x = X - \tilde{X}$ where $\tilde{X}$ is the predicted value of X. The output of the high resolution quantizer is index k indicating the prototype region. The WZ mapping block takes in k and outputs index i for transmission over the communication channel, and an encoder prediction error reconstruction value $\hat{e}_{x,enc}$ for the prediction loop. The variable $\hat{e}_{x,enc}$ takes value in an encoder codebook of I entries via a look-up operation given index i. The reconstructed residual $\hat{e}_{x,enc}$ is added to $\tilde{X}$ to obtain $\hat{X}_{enc}$, the sample reconstruction value for the encoder prediction loop. A linear predictor P_x is used to predict the next sample of X from $\hat{X}_{enc}$. A similar encoder with the obvious corresponding notation handles the second source Y and transmits an index j. The decoder has access to the pair of indices (i,j).

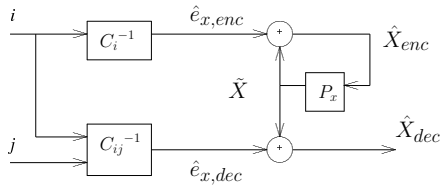

Fig. 3. The DPC decoder reconstructing source X

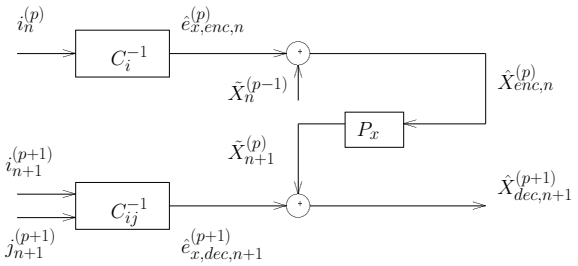

Fig. 4. DPC decoder in open loop during the design phase

C. Distributed Predictive Decoder

The DPC decoder modules in charge of reproducing X (see Fig. 3) receive index i to reconstruct $\hat{e}_{x,enc}$. The decoder mimics the encoder prediction loop to generate $\hat{X}_{enc}$ and produces $\tilde{X}$ using the predictor P_x. Given index pair (i, j), the decoder retrieves $\hat{e}_{x,dec}$ from the *decoder codebook*, and adds it to $\tilde{X}$ to obtain the decoder reconstruction $\hat{X}_{dec}$. Here C_i^{-1} and C_{ij}^{-1} represent "inverse quantizers", i.e., the corresponding table look-up operation applied to the respective codebooks.

We should re-emphasize that $\hat{X}_{enc}$ is a (coarse) reconstruction of X which only serves the prediction loop, and is generally different from $\hat{X}_{dec}$, the decoder reconstruction of X. Also note that the "encoder codebook" C_i^{-1} which is used in the prediction loop at both the encoder and the decoder is, in general, different from the "decoder codebook" C_{ij}^{-1} (used only at the decoder).

D. Components to Optimize

DPC design optimizes the predictors P_x, P_y, high rate quantizers, Wyner-Ziv mappings, encoder codebooks and decoder codebooks for all sources. For simplicity, we eliminate consideration of the less critical components: we design the predictors in open-loop, similar to standard practice in predictive quantization [3]. The high rate quantizers are then designed using Lloyd's algorithm [6] given the open-loop prediction error. These components are then kept fixed throughout the design.

We derive update rules for the WZ mappings, the Encoder and the Decoder code-books for both the sources and give an iterative algorithm to minimize the overall distortion. For conceptual simplicity, we analyze the DPC system assuming first order linear prediction.

Note that the quantized error sample $\hat{e}_{x,enc}$ at time n impacts $\tilde{X}$ and $\hat{X}_{dec}$ from time $n+1$ onwards due to the presence of prediction loop. On the other hand, $\hat{e}_{x,dec}$ at time n only impacts $\hat{X}_{dec}$ at time n. This is explicit in Fig. 3.

However, if the DPC decoder were to perform in "open-loop" as shown in Fig. 4, then a particular sample of $\hat{e}_{x,enc}$ will affect only the next sample of $\hat{X}_{dec}$ and not all the samples following it. This is our rationale of adopting the asymptotic closed loop (ACL) approach for DPC system design. The super-script p in Fig. 4 denotes a particular iteration of the ACL and will made more clearer when we describe ACL in the next section. In ACL, the coder design iteration is performed in open-loop by keeping the $\tilde{X}$ sequence (for all n) fixed throughout the iteration. A new $\tilde{X}$ sequence is then available for the next iteration of ACL. An important characteristic of the ACL technique is that the design is performed in open-loop but as the algorithm converges, the prediction loop is effectively closed and the operation mimics closed loop.

We next give a brief overview of ACL, cast the distributed prediction problem within the ACL framework and give the update rules (necessary conditions for optimality) for the WZ mappings, and encoder and decoder codebooks. For detailed treatment of ACL and its applications, see [4], [5].

III. DESIGN ALGORITHM

A. ACL Overview – Single Source

A predictive quantizer can be designed using an open-loop (OL) or a closed-loop (CL) approach [3]. In OL the training set of prediction errors for quantizer design is independent of the quantizer and the design algorithm is stable and converges to a local minimum. However, when the prediction loop is eventually closed for actual operation of the system, prediction error statistics differ from those observed during design. Hence, the system performance is suboptimal. In CL, the iteration consists of two steps: fix the current quantizer and run the system in closed loop to obtain a new training set of prediction errors; design a new quantizer for this training set. However, the training set keeps changing in an unpredictable fashion, and every updated quantizer is applied to error statistics it had not been designed for. Hence, there is no guarantee that the algorithm will converge. Moreover, the procedure may be unstable as errors build up through the prediction loop [3].

The ACL approach overcomes these shortcomings of traditional predictive coder design. The design is done in open-loop, and each quantizer is designed for the statistics of the exact signal it then quantizes to produce a new sequence of reconstruction for the next iteration, thereby circumventing stability problems. Asymptotically, when there is no improvement in the cost, the reconstruction sequence is essentially unchanged between iterations and the loop is effectively closed.

More specifically, for a given quantizer $Q^{(p-1)}$ at iteration $p-1$ of ACL, a new training set of prediction errors $T^{(p)} = \{e_n^{(p)}\}_{n=1}^N$ is generated as:

$$e_n^{(p)} = x_n - P_x \hat{x}_{n-1}^{(p-1)}, \tag{2}$$

where P_x is the predictor coefficient for first order linear prediction. Note that the subscript n here denotes time (we suppress the subscript x in e_x for notational simplicity).

Quantizer $Q^{(p)}$ is designed for the training set $T^{(p)}$ and new set of reconstruction values for x is obtained by applying $Q^{(p)}$ to $T^{(p)}$ itself:

$$\hat{x}_n^{(p)} = P_x \hat{x}_{n-1}^{(p-1)} + Q^{(p)}(e_n^{(p)}). \quad (3)$$

Since the quantizer is applied to the exact training set for which it was designed, it is the best quantizer for the job and hence the cost will decrease. This will result in better prediction. A new error training set $T^{(p+1)}$ is then obtained and the procedure is performed until convergence. Since the entire design is performed in open-loop, it is stable. At convergence, the quantizer updates are vanishingly small $Q^{(p+1)} \approx Q^{(p)}$. Therefore, the reconstructed sequence is unchanged with iterations, i.e., $\hat{x}_n^{(p+1)} \approx \hat{x}_n^{(p)}$ implying $P_x[\hat{x}_{n-1}^{(p+1)}] \approx P_x[\hat{x}_{n-1}^{(p)}]$ which means that although the loop is open, we are effectively predicting from the previous sample reconstruction in the same iteration. Hence, the loop is asymptotically closing. In summary, even though the algorithm is always running in open loop, the design is asymptotically equivalent to closed loop.

B. ACL for Distributed Prediction

The ACL distributed predictive decoder for source X is shown in Fig. 4. A similar decoder is derived for the second source Y. During the design iteration, the prediction loop is open as shown. The distortion cost to be minimized is:

$$E[\alpha\, d(X, \hat{X}_{dec}^{(p+1)}) + (1-\alpha)\, d(Y, \hat{Y}_{dec}^{(p+1)})]. \quad (4)$$

Note that during iteration p, we seek to minimize the cost at iteration $p+1$. Asymptotically, this makes no difference. The reason behind this subterfuge is to obtain effective update rules given that both $\hat{e}_{enc,n}^{(p)}$ and $\hat{e}_{dec,n+1}^{(p+1)}$ affect $\hat{X}_{dec,n+1}^{(p+1)}$. Note further that in ACL, the design is actually in open-loop and therefore $\hat{e}_{enc,n}^{(p)}$ affects $\hat{X}_{dec}^{(p+1)}$ at time $n+1$ only.

C. Update Rules

For simplicity, we assume mean-squared error distortion. While for completeness we provide the precise update rules below despite notational complexity, a high level and concise description of the algorithm is given in the next subsection.

The decoder codebook, encoder codebook and Wyner Ziv mappings are updated iteratively using the following steps:

1) **Decoder Codebook**: Entry (i,j), $i = 1:\mathcal{I}$ and $j = 1:\mathcal{J}$ is given by:

$$\hat{e}_{x,dec}(i,j) = \arg\min_{\phi} \sum_{n:(e_{x,n}^{(p+1)}, e_{y,n}^{(p+1)}) \in R_i \times R_j} d[e_{x,n}^{(p+1)}, \phi]. \quad (5)$$

2) **Encoder Codebook**: Entry i, $i = 1:\mathcal{I}$ is given by:

$$\hat{e}_{x,enc}(i) = \arg\min_{\psi} \sum_{n:e_{x,n}^{(p)} \in R_i} d_{n+1}^{(p+1)}, \quad (6)$$

employing compact notation for

$$d_{n+1}^{(p+1)} = \alpha\, d[e_{x,n+1}^{(p+1)}, \hat{e}_{x,dec}(i_{n+1}^{(p+1)}, j_{n+1}^{(p+1)})] + (1-\alpha)\, d[e_{y,n+1}^{(p+1)}, \hat{e}_{y,dec}(i_{n+1}^{(p+1)}, j_{n+1}^{(p+1)})], \quad (7)$$

where the resulting prediction error of source X depends on ψ: $e_{x,n+1}^{(p+1)} = x_{n+1} - P_x[\tilde{x}_n^{(p-1)} + \psi]$, and $(i_{n+1}^{(p+1)}, j_{n+1}^{(p+1)})$ is the index pair received at time $n+1$ in iteration $(p+1)$.

3) **WZ Mappings**: For $k = 1, \ldots, K$, assign region k to index $i = v(k)$ such that:

$$v(k) = \arg\min_{i \in \{1..I\}} \sum_{n: e_{x,n}^{(p)} \in C_k \text{ or } e_{x,n+1}^{(p+1)} \in C_k} d_{n+1}^{(p+1)}. \quad (8)$$

The update rules for WZ mappings, the encoder codebook and the decoder codebook for source Y are similarly obtained. Note that i and j point to codebook entries, subscript n indicates time, and superscript p indicates the ACL iteration. To reduce clutter, superscripts were omitted where obvious, e.g., R_i for R_i^x.

D. Design Algorithm Summary

The overall algorithm for distributed predictive coding system design is described as follows:

1) Design predictors P_x and P_y and high rate quantizers Q_x and Q_y, e.g., via classical predictive quantizer algorithm (see [3]).
2) Initialize (e.g., randomly) the WZ mappings, the *encoder* and the *decoder* codebooks for the sources. Set iteration counter $p = 1$.
3) Update the WZ mappings, the encoder and decoder codebooks.
4) Evaluate system performance. Check for convergence to **STOP**. Otherwise, calculate the new training set of errors for e_x (and e_y) using similar expressions applied on X (and Y) as in Eqn. 2. Increment p. Go to step 3.

IV. SIMULATION RESULTS

We use the following Gauss-Markov source model for the simulations:

$$X_n = \beta X_{n-1} + w_n \quad \text{and} \quad Y_n = \gamma Y_{n-1} + u_n. \quad (9)$$

where w_n, u_n are i.i.d., zero-mean, unit variance, jointly Gaussian scalar sources with correlation coefficient ρ. A training set of size 5000 scalars is generated. The predictors P_x (and P_y) are first-order linear predictors designed using X (and Y). In our simulations, we have taken $\beta = \gamma = 0.8$ and $\rho = 0.95$. The weighting coefficient of (1) is set to $\alpha = 0.5$ which implies that equal importance is given to both the sources at the decoder. The number of prototypes is 60 for each source.

Simulation results are depicted in Fig. 5. In the first experiment, we encoded both sources at the same rate and measured the weighted distortion at the decoder versus the number of transmitted bits for each source. We compare the three schemes: (a) non-distributed predictive coding, i.e., each source is compressed using standard predictive coding independently; (b) memoryless distributed coding, i.e., no prediction is performed; (c) distributed predictive coding (DPC). DPC clearly outperforms the other two compression schemes

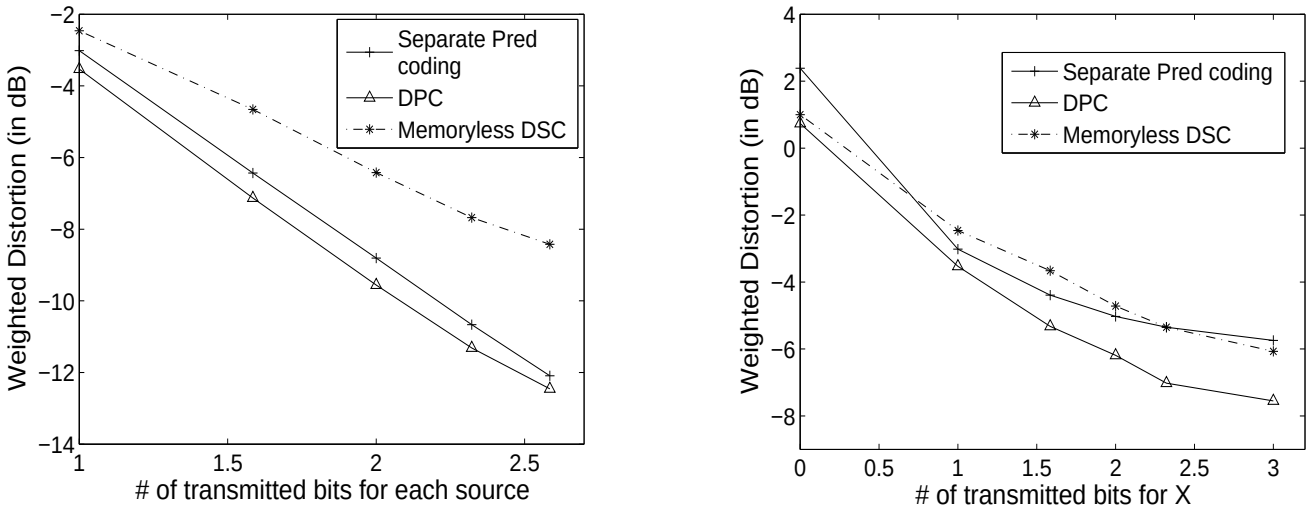

Fig. 5. Performance comparison: distributed predictive coding, non-distributed predictive coding, and memoryless distributed coding. In the left figure, both sources are encoded at the same rate. In the right figure, source Y is encoded at fixed rate of 1 bit per sample.

and gains of ~ 0.75 dB are achieved (e.g., at $R_1 = R_2 = 2$ bits/sample). In the second experiment, the transmission rate for Y was fixed at 1 bit/sample while the rate of X was varied. Here again, DPC is superior to either extreme special case (predictive but not distributed, or distributed but memoryless).

It should be mentioned that we have run the distributed prediction design algorithm multiple times since it is an iterative technique and susceptible to fall in a local minima trap depending on initialization. To circumvent this problem, less greedy techniques such as in [9] may be employed but fall outside the scope of this paper. Also, the predictors for the sources have been designed in open-loop and kept fixed throughout the system design without further optimization (as is often done in classical predictive quantizer design). This simplification may be revisited in future work. Further, to save in transmission rate, one can and probably should use entropy coding.

ACKNOWLEDGMENTS

This research was supported in part by the NSF under grant IIS-0329267, the University of California MICRO program, Applied Signal Technology Inc., Dolby Laboratories Inc., and Qualcomm Inc.

REFERENCES

[1] J. Cardinal and G. Van Assche, "Joint entropy-constrained multiterminal quantization," *Proc. IEEE ISIT*, p. 63, Jun. 2002.

[2] M. Fleming, Q. Zhao and M. Effros, "Network vector quantization," *IEEE Trans. on Information Theory*, vol. 50, no. 8, pp. 1584-1604, Aug. 2004.

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

[4] H. Khalil, K. Rose and S. L. Regunathan, "The asymptotic closed-loop approach to predictive vector quantizer design with application in video coding," *IEEE Trans. on Image Processing*, vol. 10, no. 1, pp. 15-23, Jan. 2001.

[5] H. Khalil and K. Rose, "Predictive vector quantizer design using deterministic annealing," *IEEE Trans. on Signal Processing*, vol. 51, no. 1, pp. 244-254, Jan. 2003.

[6] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. on Information Theory*, vol. 28, pp. 129-137, Mar. 1982.

[7] S. S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): design and construction," *Proc. IEEE Data Compression Conference*, pp. 158-167, Mar. 1999.

[8] D. Rebollo-Monedero, R. Zhang and B. Girod, "Design of optimal quantizers for distributed source coding," *Proc. IEEE Data Compression Conference*, pp. 13-22, Mar. 2003.

[9] A. Saxena, J. Nayak, K. Rose, "On efficient quantizer design for robust distributed source coding," *Proc. IEEE Data Compression Conference*, pp. 63-72, Mar. 2006.

[10] A. Saxena, J. Nayak, K. Rose, "A global approach to joint quantizer design for distributed coding of correlated sources," *Proc. IEEE ICASSP*, vol. 2, pp. 53-56, May 2006.

[11] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. on Information Theory*, vol. 19, no. 4, pp. 471-480, Jul. 1973.

[12] E. Tuncel, "Predictive coding of correlated sources," *IEEE Information Theory Workshop*, pp. 111-116, Oct. 2004.

[13] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. on Information Theory*, vol. 22, no. 1, pp. 1-10, Jan. 1976.