Weighted Universal Image Compression

Michelle Effros, *Member, IEEE,* Philip A. Chou, *Member, IEEE,* and Robert M. Gray, *Fellow, IEEE*

***Abstract*— We describe a general coding strategy leading to a family of universal image compression systems designed to give good performance in applications where the statistics of the source to be compressed are not available at design time or vary over time or space. The basic approach considered uses a two-stage structure in which the single source code of traditional image compression systems is replaced with a family of codes designed to cover a large class of possible sources. To illustrate this approach, we consider the optimal design and use of two-stage codes containing collections of vector quantizers (weighted universal vector quantization), bit allocations for JPEG-style coding (weighted universal bit allocation), and transform codes (weighted universal transform coding). Further, we demonstrate the benefits to be gained from the inclusion of perceptual distortion measures and optimal parsing. The strategy yields two-stage codes that significantly outperform their single-stage predecessors. On a sequence of medical images, weighted universal vector quantization outperforms entropy coded vector quantization by over 9 dB. On the same data sequence, weighted universal bit allocation outperforms a JPEG-style code by over 2.5 dB. On a collection of mixed text and image data, weighted universal transform coding outperforms a single, data-optimized transform code (which gives performance almost identical to that of JPEG) by over 6 dB.**

***Index Terms*—Adaptive coding, bit allocation, clustering, image compression, JPEG, perceptual distortion measures, transform coding, two-stage coding, universal coding, vector quantization.**

I. Introduction

TRADITIONAL image compression systems are designed using assumptions or *a priori* knowledge about the types of images to be compressed. For example, vector quantizers (VQ's) are designed using training sets of data believed to be representative of incoming data. Similarly, the JPEG image coding standard employs design choices made using assumptions about the types of data to be compressed. JPEG's quantization matrix, used in allocating the available rate among a collection of discrete cosine transform (DCT) coefficients, is typically designed not only to take advantage of the differing psychovisual importances of different frequency domain coefficients but also to match the statistics observed in a sequence of training images. Further, use of the DCT for image decomposition reflects an assumption that the majority of images to be compressed are "smooth," "natural" images, for which the DCT is best suited. When the assumptions upon which we build an image compression system fail, performance suffers. A vector quantizer (VQ) designed for one source and operated on another will not achieve the best possible performance for the source in operation, as shown by the example in Fig. 1. Similarly, when JPEG is used to compress images that fail to meet the "smooth," "natural" image assumption, poor performance results.

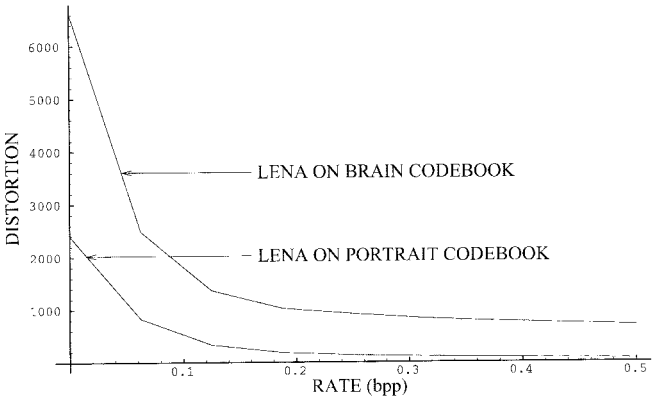


Fig. 1. Distortion-rate results for fixed-rate coding of a portrait (Lena) with a standard full search VQ trained on brain images compared to fixed-rate coding of the same image with a standard full search VQ trained on portraits. In both cases, the training and test sets do not overlap.

Unfortunately, in many image compression applications, the statistics of the images to be compressed are not known at design time and may in fact vary over space or time. If the encoder and decoder were "omniscient" and could independently identify the source in operation prior to coding, we could employ a coding strategy that switched codes during the compression process. That is, both encoder and decoder could independently switch among some collection of codes, always using the best code for the image in operation. Since real systems cannot be omniscient, an encoder that switches codes to match unknown or varying source statistics must use overhead bits to describe which code will be employed on each data block. The resulting compression system, illustrated in Fig. 2, is a two-stage or multicodebook code. Using this approach, the encoder describes data in two stages. In the first stage, the encoder describes a code from its collection. In the second stage, the encoder describes the data using the code described in the first-stage description. The decoder reverses the process, using the first-stage code description to choose

Manuscript received January 7, 1997; revised January 15, 1999. This work was presented in part at the 1994 IEEE Data Compression Conference, the 1995 IEEE International Conference on Acoustics, Speech, and Signal Processing, and the 1995 IEEE International Conference on Image Processing. This work was supported in part by the National Science Foundtion under Grant MIP-9501977, by a Bell Laboratories Ph.D. Scholarship, and by a grant from the Center for Telecommunications at Stanford University. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Roland Wilson.

M. Effros is with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125 USA (e-mail: effros@caltech.edu).

P. A. Chou is with the Microsoft Corporation, Redmond, WA 98052 USA.

R. M. Gray is with the Information Systems Laboratory, Department of Electrical Engineering, Stanford University, Stanford, CA 94305 USA.

Publisher Item Identifier S 1057-7149(99)07547-8.

Fig. 2. Two-stage code.

a code from its collection and then using the chosen code to decode the description in the second stage. The Rice machine, used for image compression on the spacecraft Voyager, is an example of a lossless two-stage code. The Rice machine independently encodes each block of 16 image pixels using the best of four memoryless entropy codes. The selected code is specified using a 2-b prefix for each coded block [1], [2]. Two-stage codes employing a collection of quantizers can likewise be designed for lossy source coding. For example, changing strategies from frame to frame in an MPEG coded sequence (and describing those changes to the decoder) is a form of two-stage coding.

A number of questions arise in the design of two-stage codes. How many bits should be used in the first-stage description? Which codes should be included in the collection? To answer these questions, we interpret the two-stage code as a "quantizer" that quantizes the space of possible codes. That is, given a collection of possible sources, a code type, and a target rate, associated with each source is an optimal code of the given type. Thus, associated with any class of possible sources there exists an analogous class of codes, where each code in the class is the optimal code corresponding to a single source in the collection. By designing any single code with the hopes of doing well across the images in our collection, we effectively quantize the space of possible codes to rate zero. That is, we choose one code that will do well on average across the images in the collection. Since only one code is used, no rate is required for describing the code in operation on a particular data block. In designing a collection of codes, we effectively quantize the space of possible codes to some rate greater than zero. Thus, optimal two-stage code design is in some way analogous to optimal quantizer design in the sense that the optimal two-stage system's collection of codes is the collection of codes that "quantizes" the space of possible codes in a rate-distortion optimal fashion.

In Section II, we present a descent algorithm for designing two-stage algorithms that allow code changes every n vectors. We call the resulting codes "weighted universal" codes. The weighted universal code design algorithm, which is functionally equivalent to the generalized Lloyd algorithm (GLA), or rather its entropy-constrained variation, arises as a direct consequence of the quantization interpretation. In Section III, we generalize the fixed dimension code changing strategy to achieve a variable dimension two-stage coding strategy that changes codes according to a variable schedule optimized to match changes in the source characteristics.

Two-stage coding strategies are applicable to a wide range of code types. A variety of sample applications are described, compared and discussed in the appendixes. In Appendix A, we describe the weighted universal VQ (WUVQ) algorithm, which employs a collection of VQ's and a fixed schedule for codebook switches. The optimal scheduling variation on this approach, called a variable dimension WUVQ (VDWUVQ) appears in Appendix B. A discussion of the weighted universal bit-allocation (WUBA) algorithm, which uses a collection of quantization matrices for JPEG-style coding, appears in Appendix C. In Appendix D, we demonstrate a weighted universal perceptual image code (WUPIC), which shows the performance achieved by replacing the traditional squared error distortion measure with a perceptually weighted distortion criterion. Appendix E treats a weighted universal transform code (WUTC), which uses an optimal collection of transforms in addition to an optimal collection of bit-allocations.

II. Code Design

Let $x^l = (x_1, \cdots, x_l) \in \mathcal{X}^l$ represent an l-dimensional data vector, typically comprised of the pixel values in a $\sqrt{l} \times \sqrt{l}$ image block. Let $\mathcal{C}^l$ be some family of length-l block codes, where associated with any code $C = \beta \circ \alpha \in \mathcal{C}^l$ is an encoder α: $\mathcal{X}^l \to \mathcal{S}$ that maps each input vector $x^l \in \mathcal{X}^l$ to a single binary codeword s from binary prefix code $\mathcal{S}$ and a decoder β: $\mathcal{S} \to \hat{\mathcal{X}}^l$, that maps the binary codeword $s \in \mathcal{S}$ to a reproduction $\hat{x}^l$ from the reproduction alphabet $\hat{\mathcal{X}}^l$. Let $d(x^l, C) = d(x^l, \beta(\alpha(x^l)))$ be the total distortion achieved by coding x^l with code C. Similarly, let $r(x^l, C) = |\alpha(x^l)|$ denote the associated rate.

Next consider a collection of such codes, $\{C_{\tilde{s}}\}$, indexed by a binary string $\tilde{s}$ from a prefix code $\tilde{\mathcal{S}}$. Each code $C_{\tilde{s}}$ in this collection is composed of an encoder $\alpha_{\tilde{s}} : \mathcal{X}^l \to \mathcal{S}_{\tilde{s}}$, which maps each l-dimensional vector x^l to a binary codeword $s_{\tilde{s}} \in \mathcal{S}_{\tilde{s}}$, and a decoder $\beta_{\tilde{s}} : \mathcal{S}_{\tilde{s}} \to \hat{\mathcal{X}}^l$, which maps each binary codeword $s_{\tilde{s}} \in \mathcal{S}_{\tilde{s}}$ to a reproduction vector $\hat{x}^l \in \hat{\mathcal{X}}^l$. Here $\mathcal{S}_{\tilde{s}}$ is the binary prefix code associated with code $C_{\tilde{s}}$.

We wish to code each block of n l-dimensional vectors with exactly one of the codes in this collection. For each such block, the encoder sends to the decoder the index $\tilde{s}$ of the code used for the block, then independently codes each of the n l-dimensional vectors with code $C_{\tilde{s}}$. Thus the value of n determines the (fixed) schedule according to which new codes can be used on data vectors x^l. We may set n equal to the number of l-dimensional vectors in a single image if exactly one code is desired for each image.Smaller values of n are useful for applications where allowing the coding strategy to change within a single image yields superior performance. However, for small values of n, the side information used to describe the code in use becomes more significant. Here we assume that n is a fixed constant; this assumption will be removed in Section III. By coding x^{ln} with code $C_{\tilde{s}}$, the total instantaneous distortion achieved is

$$d(x^{ln}, C_{\tilde{s}}) = \sum_{i=1}^{n} d(x_i^l, C_{\tilde{s}})$$

while the total instantaneous rate is

$$r(x^{ln}, C_{\tilde{s}}) = |\tilde{s}| + \sum_{i=1}^{n} r(x_i^l, C_{\tilde{s}}).$$

The total rate includes the rate associated with describing the selected code $C_{\tilde{s}}$ to the decoder.

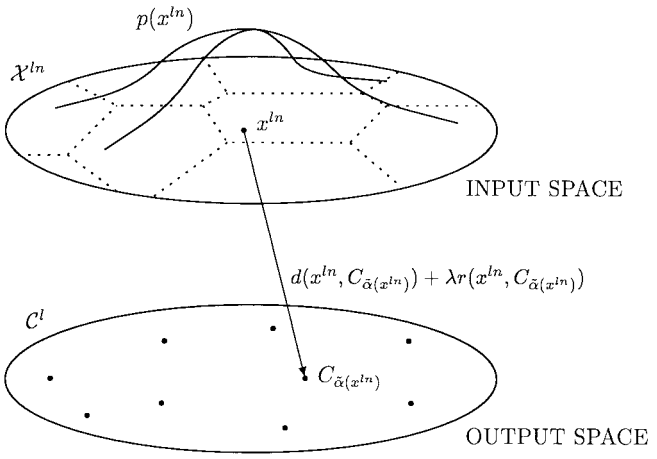


Fig. 3. First-stage quantizer.

The goal of code design is to find the prefix code $\tilde{\mathcal{S}}$ and the collection of codes $\{C_{\tilde{s}}\}_{\tilde{s}\in\tilde{\mathcal{S}}}$ that minimize the expected distortion subject to a constraint on the expected rate. This minimization can be performed using the quantization interpretation of two-stage weighted universal codes [3], in which the collection of codes $\{C_{\tilde{s}}\}$ is considered a quantization *codebook* with ''codewords'' $C_{\tilde{s}} \in \mathcal{C}^l$. With this interpretation, each data sequence x^{ln} is ''quantized'' to one of the codewords, say $C_{\tilde{s}}$, with resulting distortion and rate given by (1) and (2). To be precise, we define the ''first-stage'' quantizer $\tilde{C}$ (as distinct from the ''second-stage'' quantizers $C_{\tilde{s}}$, $\tilde{s} \in \tilde{\mathcal{S}}$), as composed of an encoder $\tilde{\alpha} : \mathcal{X}^{ln} \to \tilde{\mathcal{S}}$, which maps each data block x^{ln} to a binary description $\tilde{s}$ from the binary prefix code $\tilde{\mathcal{S}}$, and a decoder $\tilde{\beta} : \tilde{\mathcal{S}} \to \mathcal{C}^l$, which maps each binary description $\tilde{s} \in \tilde{\mathcal{S}}$ to its corresponding reproduction $C_{\tilde{s}}$.

Fig. 3 illustrates the first-stage quantizer. The first-stage quantizer's reproduction of a data block x^{ln} is the length-l block code $C_{\tilde{\alpha}(x^{ln})} = \tilde{\beta}(\tilde{\alpha}(x^{ln})) \in \mathcal{C}^l$ used to code the subblocks $x_1^l, \ldots, x_n^l$ of x^{ln}. In coding the block x^{ln} with code $\tilde{\beta}(\tilde{\alpha}(x^{ln}))$, the total distortion and rate are

$$d(x^{ln}, \tilde{\beta}(\tilde{\alpha}(x^{ln}))) = \sum_{i=1}^{n} d(x_i^l, \beta_{\tilde{\alpha}(x^{ln})}(\alpha_{\tilde{\alpha}(x^{ln})}(x_i^l))),$$

$$r(x^{ln}, \tilde{\beta}(\tilde{\alpha}(x^{ln}))) = |\tilde{\alpha}(x^{ln})| + \sum_{i=1}^{n} |\alpha_{\tilde{\alpha}(x^{ln})}(x_i^l)|.$$

Now, minimizing the expected distortion $E[d(X^{ln}, \tilde{\beta}(\tilde{\alpha}(X^{ln})))]$ subject to a constraint on the expected rate $E[r(X^{ln}, \tilde{\beta}(\tilde{\alpha}(X^{ln})))]$ is equivalent to minimizing the expected Lagrangian $E[d(X^{ln}, \tilde{\beta}(\tilde{\alpha}(X^{ln}))) + \lambda r(X^{ln}, \tilde{\beta}(\tilde{\alpha}(X^{ln})))]$ for some positive value of λ determined by the rate constraint.

The optimal design is achieved using an iterative descent technique formally equivalent to the GLA [4], or rather its entropy-constrained variation [5]. The algorithm is initialized with an arbitrary prefix code $\tilde{\mathcal{S}}$ and first-stage decoder $\{\tilde{\beta}(\tilde{s}): \tilde{s} \in \tilde{\mathcal{S}}\}$ such that $\tilde{\beta}(\tilde{s}) \in \mathcal{C}^l$ for all $\tilde{s} \in \tilde{\mathcal{S}}$. Each iteration in the algorithm is accomplished in the three steps enumerated below.

1) *Nearest Neighbor Encoding:* Optimize the first-stage encoder for the given first-stage decoder $\tilde{\beta}$ and prefix code $\tilde{\mathcal{S}}$. For a given collection of codes $\tilde{\beta}$, the optimal first-stage encoder $\tilde{\alpha}^\star$ is the first-stage encoder that satisfies $\tilde{\alpha}^\star(x^{ln}) = \arg\ \min_{\tilde{s}\in\tilde{\mathcal{S}}}[d(x^{ln}, \tilde{\beta}(\tilde{s})) + \lambda r(x^{ln}, \tilde{\beta}(\tilde{s}))]$ for every x^{ln}. We call the optimal first-stage encoder a *nearest neighbor* encoder. Nearest neighbor encoders can be implemented by encoding the incoming data vector with each code in the collection and choosing the code that yields the lowest Lagrangian performance.
2) *Decoding to the Centroid:* Optimize the first-stage decoder for the given first-stage prefix code $\tilde{\mathcal{S}}$ and the newly redesigned first-stage encoder $\tilde{\alpha}$. The optimal first-stage decoder $\tilde{\beta}^\star$ for a given first-stage encoder $\tilde{\alpha}$ satisfies $\tilde{\beta}^\star(\tilde{s}) = \arg\ \min_{C\in\mathcal{C}^l} E[d(X^{ln}, C) + \lambda r(X^{ln}, C)|\tilde{\alpha}(X^{ln}) = \tilde{s}]$ for every $\tilde{s} \in \tilde{\mathcal{S}}$. By analogy to the optimal decoder design in vector quantization, we call the process of designing the optimal first-stage decoder *decoding to the centroid*. For each $\tilde{s} \in \tilde{\mathcal{S}}$, decoding to the centroid involves choosing or designing a single code $C^\star \in \mathcal{C}^l$ that gives optimal performance for the set of data $\{x^{ln}: \tilde{\alpha}(x^{ln}) = \tilde{s}\}$. In practice, this is accomplished using an optimal design algorithm for the family $\mathcal{C}^l$ of codes under consideration. Descriptions of the design algorithms associated with a variety of code classes appear in the appendixes.
3) *Optimizing the Prefix Code:* Optimize the first-stage prefix code $\tilde{\mathcal{S}}$ for the newly redesigned first-stage encoder $\tilde{\alpha}$ and first-stage decoder $\tilde{\beta}$. For variable-rate coding, the optimal prefix code $\tilde{\mathcal{S}}^*$ is the entropy code matched to probabilities $P\{\tilde{\alpha}(X^{ln}) = \tilde{s}\}$, with ideal codelengths $|\tilde{s}^*| = -\log P\{\tilde{\alpha}(X^{ln}) = \tilde{s}\}$.

Notice that the prefix code may be a fixed- or variable-length code, and the family $\mathcal{C}^l$ may be any family of block codes. Thus the above algorithm could be used to design a collection of lossless codes like the one used in the Rice machine or a collection of fixed- or variable-rate quantizers, as is the focus in the examples in this paper.

For any family $\mathcal{C}^l$ of codes, the expected Lagrangian decreases or remains constant in each step of the above design algorithm. Since Lagrangian performance is bounded below by zero, it must decrease to a limit as the number of iterations grows. In fact, if the decoding to the centroid operation yields a global optimum or can itself be described as an iterative procedure in which every step guarantees a global optimum, then the above procedure guarantees a locally optimal collection of codes. This criterion is met in all of the cases considered in this paper.

The optimal design algorithm for a collection of codes employs a single codebook optimization algorithm within the framework provided by the GLA. Our algorithm is not the first to use the GLA with another optimization algorithm nested inside. For example, Buzo *et al.* [6] use the GLA to cluster linear predictors for speech, using the prediction error as the distortion measure and the Levinson algorithm to optimize the predictors. Rabiner *et al.* [7] use the GLA to cluster hidden Markov models for speech, using the log likelihood

as the distortion measure and the Baum–Welch algorithm to optimize the models; Safranek and Johnston [8] use the GLA to cluster entropy codes for subband image coding, using bit rate as the distortion measure and the Huffman algorithm to optimize the entropy codes; Chou [9] uses the GLA to cluster probability mass functions for classification tree design, using the Kullback–Leibler divergence as the distortion measure and conditional expectation to calculate the pmfs; and Chan and Gersho [10] use the GLA to cluster vector quantizers for residual VQ, using the overall distortion as the distortion measure and a nested GLA to optimize the individual quantizers. Our algorithm may be the first, however, to incorporate bit rates from both the first and second stages into the design. This is necessitated by the fact that both the first and second stages contribute to the rate in two-stage universal coding.

The number of bits contributed by the first stage in an optimal two-stage universal code is suggested by theory [3, pp. 1119–1120, Case 1 and Th. 6] to be some constant plus $(k/2)\log n$, where k is the number of parameters in the family of codes $\mathcal{C}^l$, i.e., the number of codewords in each $C \in \mathcal{C}^l$ times the dimension l of each codeword. In other words, the number of codebooks in an optimal two-stage code should grow as $n^{k/2}$. Experimental results showing that this formula is approximately followed in practice are shown in [3].

As compared to traditional one-stage codes, two-stage codes exhibit a growth in computational complexity roughly proportional to the number of codebooks in the two-stage code. This complexity increase results from the optimal encoding procedure, in which the data must effectively be quantized with all codebooks prior to coding in order to choose the optimal second-stage code. The complexity growth associated with going from a one-stage code to a two-stage code is analogous to the complexity growth associated with going from a rate zero VQ to a higher rate VQ. While the growth in complexity is not insignificant, it is far smaller than the growth in complexity associated with many "adaptive" codes that incorporate some level of system design in the encoder. (While the design of two-stage codes is computationally expensive, this design occurs off-line, and does not affect the run time experienced by the system user.) For many applications the significant performance improvements achieved by two-stage codes make the additional complexity worthwhile. As an example of the rate of complexity growth, for a collection of K l-dimensional, rate-R, fixed-rate VQ's and the squared-error distortion measure, encoding a sequence x^{ln} with a two-stage code requires approximately $K(2\ ln\ 2^{lR} + n)$ additions and $K\ ln\ 2^{lR}$ multiplications as compared to the $2\ ln\ 2^{lR}$ additions and $ln\ 2^{lR}$ multiplications required in a one-stage code of rate R. Of course, the complexity can be considerably reduced by using tree structures or other means.

III. Variable Dimension Two-Stage Codes

The above weighted universal codes use fixed first-stage vector dimension n. In this section, we consider two-stage codes in which the first-stage vector dimension is allowed to vary.

In variable dimension two-stage codes, each incoming data stream is partitioned into variable length first-stage vectors. The encoder then describes these first-stage vectors, one by one, to the decoder. To describe a first-stage vector, the encoder first describes the length of the first-stage vector, then describes the index of the code with which the component vectors of that first-stage vector will be encoded, and finally describes the component vectors using the chosen code. The optimal partition is the partition that minimizes the distortion subject to a constraint on the rate, which now includes the rate associated with describing the length of each first-stage vector. The use of optimally chosen first-stage vector dimensions allows variable dimension codes to better carve the data into its component subsources, coding each with an appropriately matched code.

Let $x_1^l, \cdots, x_N^l$ be an incoming data sequence of l-dimensional vectors. This data sequence will be broken into first-stage vectors of varying length, say $y_1, \cdots, y_k$, of lengths (in l-dimensional vectors) $|y_1|, \cdots, |y_k|$. The number of first-stage vectors and their lengths are subject only to the constraint that the sum of the lengths must equal the initial data length $(\Sigma_i |y_i| = N)$. Using x_1^{lN} to denote the sequence $x_1^l, \cdots x_N^l$, let $\{y_i\} \prec x_1^{lN}$ mean that $\{y_i\}$ partitions x_1^{lN} and satisfies the above constraint. Use $|\{y_i\}|$ to denote the number of elements in the partition. Since the first-stage vector length is allowed to vary, it must be described to the decoder along with the first- and second-stage code information. The encoder uses an entropy code γ to describe that length, where $|\gamma(m)|$ is the rate associated with describing a first-stage vector length of m. Notice that if $|\gamma(m)|$ is zero for length n and infinity otherwise, the variable dimension code will behave exactly like its fixed dimension counterpart. Thus, an optimal variable dimension two-stage code can only exceed an optimal fixed dimension two-stage code in performance. This performance gain is achieved at the expense of the additional complexity necessary for obtaining the optimal partition.

The optimal design algorithm for variable dimension two-stage codes is again a variation on the GLA, iteratively achieving descent on the Lagrangian performance. The system is initialized with an arbitrary first-stage encoder $\tilde{\alpha}$, first-stage decoder $\tilde{\beta}$, and pair of entropy codes γ and $\tilde{\mathcal{S}}$. Each iteration of the technique then proceeds as follows.

1) *Optimal Parsing:* Optimally parse the incoming data sequence for the given first-stage encoder $\tilde{\alpha}$, first-stage decoder $\tilde{\beta}$, and prefix codes γ and $\tilde{\mathcal{S}}$. The optimal parsing satisfies $\{y_i\} = \arg\ \min_{\{z_i\} \prec x_1^{ln}} [\Sigma_i\ d(z_i, \tilde{\beta}(\tilde{\alpha}(z_i))) + \lambda(|\gamma(|z_i|)| + r(z_i, \tilde{\beta}(\tilde{\alpha}(z_i))))]$.
2) *Nearest Neighbor Encoding:* Optimize the first-stage encoder for the given parsing $\{y_i\}$, first-stage decoder $\tilde{\beta}$, and prefix codes γ and $\tilde{\mathcal{S}}$. The optimal first-stage encoder $\tilde{\alpha}^*$ follows the nearest neighbor law, $\tilde{\alpha}^*(y_i) = \arg\ \min_{\tilde{s} \in \tilde{\mathcal{S}}} [d(y_i, \tilde{\beta}(\tilde{s})) + \lambda(|\gamma(|y_i|)| + r(y_i, \tilde{\beta}(\tilde{s})))]$, which maps each incoming block y_i to the index of the second stage code $\tilde{\beta}(\tilde{s})$ that encodes y_i to the lowest value of the Lagrangian performance measure.
3) *Decoding to the Centroid:* Optimize the first-stage decoder for the given parsing $\{y_i\}$, prefix codes γ and

$\tilde{\mathcal{S}}$, and first-stage encoder $\tilde{\alpha}$. The optimal first-stage decoder $\tilde{\beta}^*$ satisfies $\tilde{\beta}^*(\tilde{s}) = \arg\min_{C\in\mathcal{C}^l} E[d(Y_i, C) + \lambda(|\gamma(|Y_i|)| + r(Y_i, C))|\tilde{\alpha}(Y_i) = \tilde{s}]$, which assigns to each $\tilde{s} \in \tilde{\mathcal{S}}$ the blocklength-l code $C \in \mathcal{C}^l$ that minimizes the expected Lagrangian given that Y_i mapped to $\tilde{s}$ in the second step.

4) *Optimizing the Prefix Codes:* Optimize the prefix codes γ and $\tilde{\mathcal{S}}$ for the given parsing $\{y_i\}$, first-stage encoder $\tilde{\alpha}$, and first-stage decoder $\tilde{\beta}$. The optimal prefix codes γ^* and $\tilde{\mathcal{S}}^*$ are the entropy code matched to the probabilities $P\{\gamma(y_i) = m\}$ and $P\{\tilde{\alpha}(y_i) = \tilde{s}\}$, and thus the ideal codelengths are $|\gamma^*(m)| = -\log P\{|y_i| = m\}$ and $|\tilde{s}^*| = -\log P\{\tilde{\alpha}(y_i) = \tilde{s}\}$, respectively.

The optimal partition in step 1 is accomplished using the dynamic programming argument of variable dimension VQ [11], [12]. The argument proceeds as follows. Let $J_n = \min_{\{y_i\}\prec x_1^{ln}} \Sigma_i \ [d(y_i, \tilde{\beta}(\tilde{\alpha}(y_i))) + \lambda(|\gamma(|y_i|)| + r(y_i, \tilde{\beta}(\tilde{\alpha}(y_i))))]$ be the score of the best partition of the partial data sequence $x_1^l, \cdots, x_n^l$. If B is the maximum allowed first-stage vector length, then for $1 \le n \le N$

$$J_n = \min_{b\in\{1,\cdots,B\}} \{J_{n-b} + \min_{\tilde{s}\in\tilde{\mathcal{S}}} [d(x_{n-b+1}^{lb}, \tilde{\beta}(\tilde{s})) + \lambda(|\gamma(|x_{n-b+1}^{lb}|)| + r(x_{n-b+1}^{lb}, \tilde{\beta}(\tilde{s})))]\} \quad (1)$$

where $J_0 = 0$ and $J_n = \infty$ for $n < 0$. Let b_n be the minimizing vector length b in (1). Then b_N is the last first-stage vector length in the optimal partition of x_1^{lN}. The optimal partition of x_1^{lN} can be found by backtracking: $\{y_i\} = \{\cdots b_{N-|b_N|-|b_{N-|b_N|}|}, b_{N-|b_N|}, b_N\}$. While partitioning and encoding have been described as separate processes, they can be accomplished simultaneously (at the expense of greater storage requirements) by tracking not only the optimal performance and last first-stage vector length at each time n but also the encoding information for that first-stage vector.

As a final remark, if the minimization in (1) is taken over a restricted set $\mathcal{B}_n$ of vector lengths dependent on n, then a constrained partition results. For example, to obtain a binary tree segmentation of the data x_1^{lN}, then $\mathcal{B}_n$ can include $b = 2^i$ if and only if $n \bmod 2^i = 0, i = 0, 1, 2, \cdots$. In such cases, the prefix code γ should also depend on n.

IV. Examples

Two-stage codes containing collections of vector quantizers (WUVQ), bit allocations (WUBA), and transform codes (WUTC) are just a few examples of the types of codes that can be designed using the design algorithm described in Section II. The resulting codes yield significant performance improvements when compared to one-stage codes of the same type and optimized for the same training set. While the details of the algorithms and their experimental performance results are left to the appendixes, a few performance highlights are summarized below.

On a sequence of medical images, the performance of WUVQ exceeds that of ECVQ by 7–8 dB. (See Appendix A, Fig. 5.) On the same data sequence, WUBA outperforms a single-stage bit allocation scheme (functionally equivalent to JPEG) by over 2.5 dB. (See Appendix C, Fig. 9.) The performance gains associated with the same type of experiments using codes optimized for the perceptual distortion measure described in Appendix D are even more extreme, with the two-stage bit allocation scheme (WUPIC) achieving up to 5 dB improvement in perceptual distortion over the performance of a one-stage bit allocation scheme. (See Appendix D, Fig. 11.) On a collection of mixed text and image data, weighted universal transform coding outperforms a single-stage transform code by over 6 dB. (See Appendix E, Fig. 13.) The performance of the optimal single-stage transform code is almost identical to the performance of the JPEG image coding standard when used with a quantization matrix optimized for the given training set. Also included in the appendixes is an example of a variable dimension two-stage code (VDWUVQ). The VDWUVQ yields performance improvements of up to 4.8 dB over the performance of the fixed dimension WUVQ, thereby demonstrating the potential benefits of the variable dimension approach. (See Appendix B, Fig. 6.)

V. Discussion and Conclusions

The quantization interpretation of two-stage codes [3] yields an iterative descent technique for designing a wide variety of optimal two-stage data compression systems. We here demonstrate the application of that basic approach in a number of different systems. (See the appendixes.) The resulting two-stage codes replace the single code of a traditional one-stage compression system with a family of codes designed to do well across a wide variety of data types. The described technique can be used to design optimal collections of a wide variety of codes, such as noiseless codes, vector quantizers, JPEG-style codes, transform codes, and so on. Given a target rate and a code type, the design algorithm can be used to find both the number of codes needed to cover the space of possible sources and the optimal collection of codes to be used.

The computational expense associated with going from a one-stage code to a two-stage code is roughly proportional to the number of codes in the two-stage system. This expense results from the optimal encoding process, which effectively involves quantizing the data with every code in the collection in order to choose the code with the best performance. The computational cost of a two-stage code can be controlled in a number of ways. First, the choice of the second-stage code type and coding dimension should take into account the computational constraints of the system: a collection of simple codes may yield better performance than any single more complex code. Second, the complexity of an optimal two-stage code is easily capped at design time by choice of the maximal number of codes in the collection. Further, the complexity can be reduced considerably by the use of tree structures or other fast search techniques. Finally, two-stage codes are ideally suited for parallel implementation where available.

Universal source coding theory demonstrates the performance benefits of two-stage codes as the coding dimension grows without bound. In this paper, we focus on the performance of two-stage codes for practical applications where computation, and therefore coding dimension, are forced to

be low. On a number of image data sets, the performance of two-stage codes with coding dimensions ranging between 4 and 64 pixels yield as much as 10 dB performance improvement over their corresponding one-stage coding counterparts. (See Appendixes A–E.) Further, these performance gains are observed even in data sets where the space of images to be coded is quite homogeneous (e.g., the medical brain scans used in Appendixes A–D). The above performance gain can be attributed to the fact that while the images themselves bear a strong resemblance to each other, on the vector scale at which the images are coded there exists a great deal of statistical variation between data blocks. Thus, two-stage coding is an attractive option for a wide variety of practical data compression applications.

APPENDIX A
WEIGHTED UNIVERSAL VECTOR QUANTIZATION

The weighted universal code design algorithm may be applied to a wide variety of code types. A simple example is the vector quantizer (VQ). In using a VQ for lossy data compression, an incoming data stream is broken into contiguous vectors of some dimension l, and each data vector is mapped to the closest reproduction value in a fixed VQ codebook. In the WUVQ algorithm, we employ a family $\mathcal{C}^l$ of l-dimensional fixed- or variable-rate VQ's. The code is designed offline, using the iterative descent algorithm described in Section II. In this case, decoding to the centroid is accomplished using the GLA (for fixed-rate coding) or its entropy-constrained variation (for variable-rate coding). The resulting design algorithm may be described as a nested GLA. This nested process easily generalizes to design a large array of two-stage VQ's by replacing either or both of the uses of the GLA by other VQ design algorithms. Alternative VQ design algorithms that may be considered include tree-structured VQ, deterministic annealing, fast codebook search algorithms, and so on.

Once a WUVQ has been designed, copies of the code are given to both encoder and decoder. Thus both encoder and decoder have identical copies of the first-stage binary code $\tilde{\mathcal{S}}$ and a collection of VQs—the first-stage decoder $\{\tilde{\beta}(\tilde{s})\colon \tilde{s} \in \tilde{\mathcal{S}}\}$. To encode a given ln-vector, the encoder breaks that vector into n l-dimensional sub-vectors, and calculates for each $\tilde{s} \in \tilde{\mathcal{S}}$ the Lagrangian performance of $\tilde{\beta}(\tilde{s})$ on the collection of l-vectors. The index of the code yielding the best Lagrangian performance is described to the decoder in the first-stage description. The second-stage description contains the binary description of each l-vector using the code described in the first stage. The decoder reverses the process by using the code described in the first stage to decode the n vectors described in the second-stage description.

Figs. 4 and 5 show the performance of fixed- and variable-rate WUVQ on a sequence of medical images, comparing that performance to the performance of standard full search VQ and ECVQ respectively. The experiments use twenty 256×256 MR images as a training set and five 256×256 MR images as a test set. The two sets do not overlap. Performance results are reported as signal-to-quantization-noise ratio (SQNR), and the rate of variable-rate codes is reported as entropy. SQNR is calculated as $-10\log_{10}(D/D_0)$, where D is the current distortion, D_0 is the zero rate distortion of a standard full-search VQ, and distortion is measured by the squared error fidelity criterion. The codes considered use a vector dimension of $l = 4$ and the WUVQ makes an independent code choice for each two-by-two block of four-dimensional vectors, giving $n = 4$. Fixed-rate systems consist of up to 512 codebooks with up to 16 codewords per codebook. Codebook sizes for first- and second-stage codebooks in the case of variable-rate coding were initialized to 256 and 4, respectively.

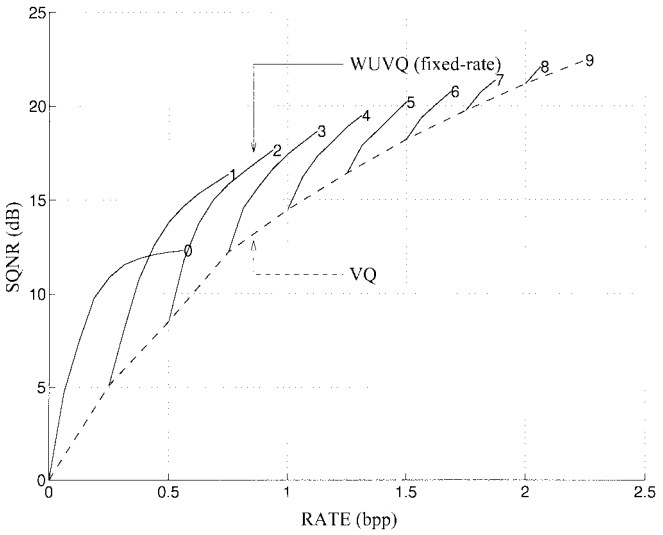


Fig. 4. SQNR versus rate results for fixed-rate coding of the medical test sequence.

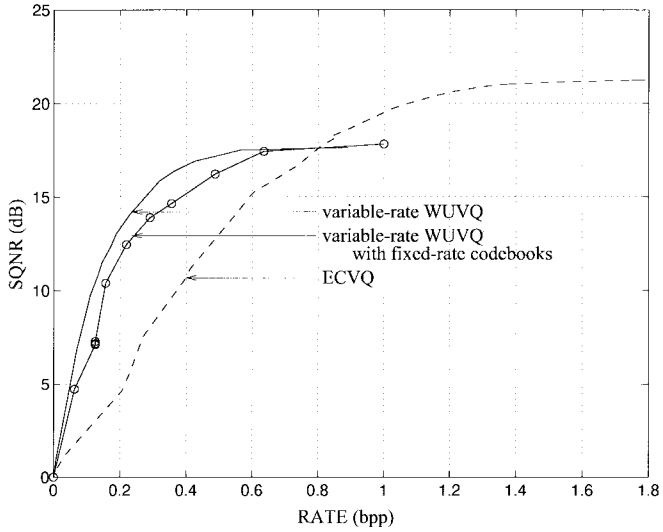


Fig. 5. SQNR versus rate results for variable-rate coding of the medical test sequence.

Fig. 4 shows the fixed-rate coding results. Each curve represents a constant value of the second-stage rate R, and is labeled with the appropriate R value. The lower dashed curve is a standard full search VQ, which is equivalent to a fixed-rate WUVQ with first-stage rate equal to zero. Fig. 5 shows the corresponding entropy-constrained results.

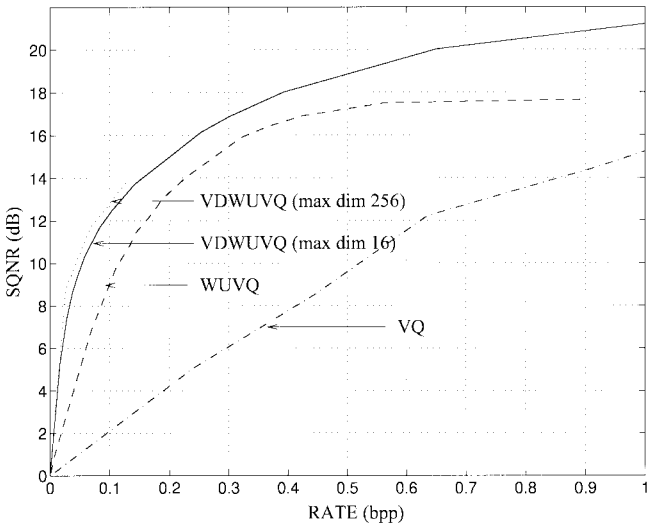


Fig. 6. Comparison of variable-rate SQNR results on a collection of medical brain scans.

In this case, we consider both fixed-rate first-stage codes with fixed-rate second-stage codes of varying rates and variable-rate first-stage codes with variable-rate second-stage codes and compare the resulting performance to that of an ECVQ. In both fixed-rate and variable-rate coding, significant gains are demonstrated. Indeed, at bit rates around 0.25–0.50 b/pixel, there is a 4–5 dB gain of fixed-rate WUVQ over standard VQ, and 7–8 dB gain of variable-rate WUVQ over ECVQ. Since ECVQ already represents a substantial improvement over entropy-coded standard VQ, variable-rate WUVQ achieves more than 9 dB gain over entropy-coded standard VQ.

Notice that the demonstrated performance improvements gained by going from single- to multi-codebook systems are achieved despite the homogeneity of the data set—a sequence of sagittal MR brain scans. By allowing the codebook to change *within* a given image, we are able to code each component of the image (in this case components might include bone, fat, gray matter, etc.) with a code matched to the statistics of that component.

Appendix B
Variable Dimension WUVQ

While variable dimension two-stage codes may be devised for any two-stage code, we here demonstrate the technique on the variable dimension WUVQ (VDWUVQ) [13] and compare the resulting performance with that of WUVQ and standard full-search VQ on the medical brain scan sequence.

All of the vector quantizers use vectors of dimension $l = 4$. The vectors are Peano scan ordered (e.g., [14]). The first-stage vectors in VDWUVQ are allowed to contain up to 16 l-dimensional vectors while the first-stage vectors in WUVQ contain exactly four l–dimensional vectors. All multicodebook systems consist of at most 256 codebooks, each with no more than four codewords.

Fig. 6 shows the performance of variable-rate VDWUVQ and WUVQ with varying values of λ and the performance of standard full-search VQ. All rates are reported in terms of entropy. VDWUVQ shows its greatest improvement over WUVQ at very low rates, below 0.1 b/pixel, with an SQNR up to 4.8 dB higher. Both VDWUVQ and WUVQ achieve their greatest gains over standard full-search VQ at slightly higher rates. VDWUVQ shows up to 11 dB improvement over standard VQ. Allowing first-stage vector lengths up to 256 vectors improves performance by about another dB at the expense of increased computation. Fig. 7 shows the resulting image coding performance. Fig. 8 demonstrates the sizes and shapes of the first-stage vectors used by the VDWUVQ algorithm in the previous figure.

Appendix C
Weighted Universal Bit-Allocation

While vector quantizers guarantee asymptotically optimal performance as the vector dimension goes to infinity, the computational complexity of a VQ grows exponentially in the vector dimension. Thus for complexity reasons, practical VQ's are typically implemented with very small vectors. Much of the advantage associated with high-dimensional vector quantization comes from the high-dimensional code's ability to exploit correlation between data samples. Transform codes, which decorrelate data prior to coding, achieve much of the advantage of high-dimensional vector codes at low complexity. Achieving this performance requires efficient allocation of the available rate among the transform coefficients. Thus, bit allocation, which is typically implemented with the use of quantization matrices, is a key step in a wide array of transform codes.

Unfortunately, the optimal bit allocation, like the optimal vector quantizer, is source dependent. For example, images containing large amounts of high-frequency information require that more rate be allocated to the high frequency coefficients than do images made up primarily of low-frequency information. An approach common to many transform codes (including the JPEG image coding standard) involves breaking an incoming data sequence into blocks, performing an independent transform on each of those blocks, and then coding the blocks using a single bit allocation (e.g., a single quantization matrix). The performance of this type of scheme can be improved by replacing the single bit allocation by a collection of bit allocations. While a single bit allocation can be designed to do well on average across a particular data sequence, no single bit allocation strategy can track local variation in data statistics. A code that incorporates a variety of bit allocation options and allows the bit allocation to change from image to image or even from data block to data block can better track local variation in source statistics.

The WUBA [15], [16] is a two-stage DCT-based transform code in which the family $\mathcal{C}^l$ of codes is a family of bit allocations. By designing a collection of quantization matrices that is optimal for a given class of possible images (rather than a single quantization matrix that is optimal on average over that class), WUBA allows us to achieve the advantage of a changing bit allocation strategy. Since the design procedure occurs offline before the coding process begins, the rate-distortion gains are achieved without the computational

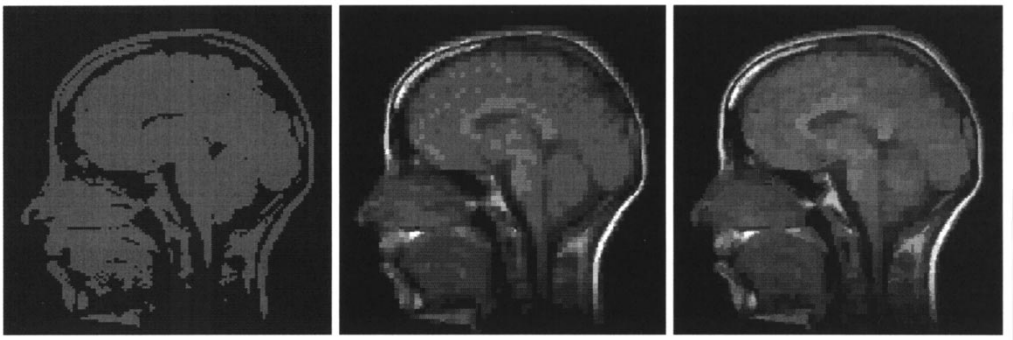


Fig. 7. From left to right: Standard VQ at 0.25 b/pixel and SQNR = 6.75 dB; WUVQ at 0.184 b/pixel and SQNR = 11.6 dB; and VDWUVQ at 0.166 b/pixel and SQNR = 13.2 dB.

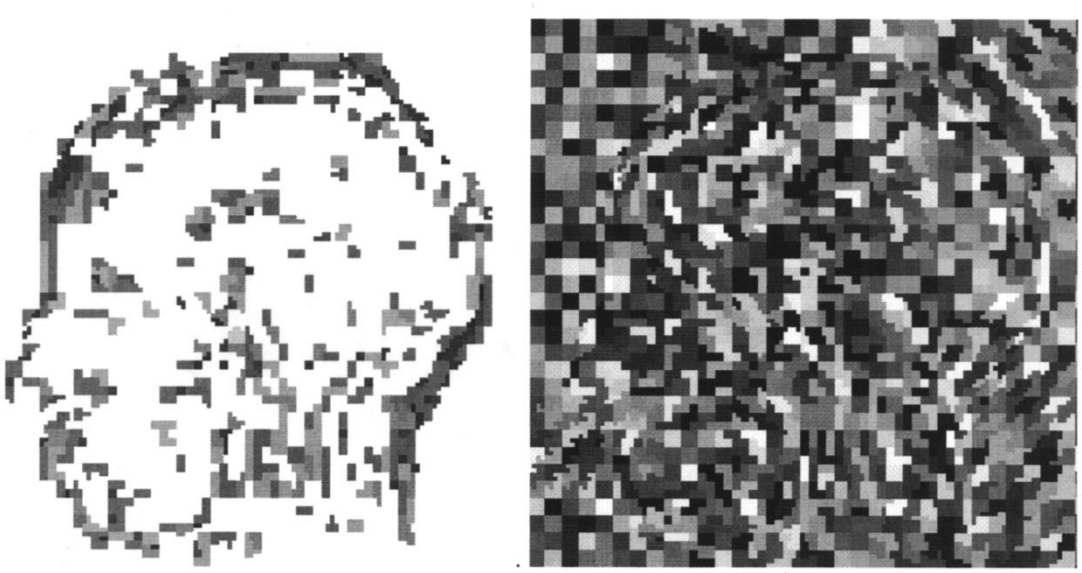


Fig. 8. First-stage vector sizes and shapes for the 0.166 b/pixel image. The left shows the size of the first-stage vector in which each pixel was coded, (black = size 1, white = size 16). The right image gives a random shade to each first-stage vector in order to demonstrate first-stage vector shapes.

expense of computing new bit allocations during the coding process.

Suppose that we are given some generic scheme for encoding x^l with bit allocation $C \in \mathcal{C}^l$. Then for any $C \in \mathcal{C}^l$ and $x^{ln} \in \mathcal{X}^{ln}, d(x^l, C)$ and $r(x^l, C)$, respectively, describe the distortion and rate associated with coding x^l using bit-allocation C. Thus, we can use the weighted universal code design strategy to design a collection of bit-allocations. In this case, decoding to the centroid may be accomplished by any number of optimal bit-allocation design algorithms. A simple DCT-based transform code and its corresponding optimal bit-allocation design algorithm are described below.

The following DCT-based transform code is similar to the JPEG algorithm but is simplified to permit an optimal centroid calculation. In this scheme, we code 8×8 data blocks $(l = 64)$. Each data block is treated independently. Each data block passes first through a two-dimensional (2-D) DCT transform. The encoder α: $\mathbb{R}^{64} \rightarrow \mathcal{S}$ maps the resulting 64-dimensional real data block into a binary string. This mapping occurs in two steps. The data block first passes through a quantization matrix $[Q_{u,v}]$, where the (u, v)th component $\mathcal{F}_{u,v}$ is divided by the corresponding quantization matrix component $Q_{u,v}$, truncating to obtain the integer $\mathcal{M}_{u,v} = \lfloor \mathcal{F}_{u,v}/Q_{u,v} + 1/2 \rfloor$. This truncation represents the lossy step in the quantization process. The encoder then losslessly describes the resulting quantized sequence to the decoder using a collection $\{\mathcal{S}_{u,v}\}$ of entropy codes. The decoder simply reverses the process, retrieving $[\mathcal{M}_{u,v}]$ from its binary representation, and then scaling back up to achieve a frequency domain reproduction $[\hat{\mathcal{F}}_{u,v}]$, where $\hat{\mathcal{F}}_{u,v} = \mathcal{M}_{u,v} Q_{u,v}$. We find the spatial domain reproduction $\hat{x}_{i,j}$ by taking an inverse DCT of $[\hat{\mathcal{F}}_{u,v}]$.

In this case, as in JPEG, bit-allocation is controlled by the quantization matrix $Q_{u,v}$ which determines how coarsely or finely a particular component will be quantized and, therefore, how much rate will be used in describing that component. The algorithm differs from the JPEG algorithm in several details. First, we remove the differential encoding of the DC component so that each source block may be treated independently. Second, we remove the run-length encoder from the lossless coding step, so that the quantizer may code

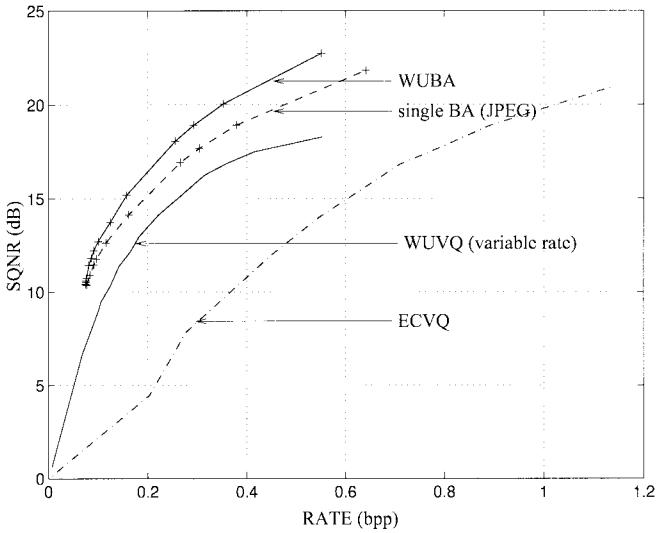


Fig. 9. Comparison of SQNR results on a collection of five MR brain scans.

each component within a given frequency block independently. We make these modifications to simplify the optimal bit allocation design.

Given the independence imposed by the above modifications to the JPEG algorithm and assuming an additive distortion measure in the frequency domain, changing the (u, v)th component in the quantization matrix affects only the rate and distortion associated with the (u, v)th component of the data blocks encoded with the given sequence. In designing a quantization matrix for a given sequence of training data, we may therefore consider each quantization matrix component independently. We choose each component $Q_{u,v}$ to minimize the Lagrangian performance on the (u, v)th transform coefficients from the training sequence. The optimal entropy code for each component is the entropy code matched to that component's statistics. We therefore achieve a simple design process for optimizing a bit-allocation system for a particular set of training data. The resulting quantization table represents a good starting value from which we can design the quantization and Huffman tables for the JPEG algorithm. A simple variation on the entropy-constrained variation of the GLA can then be used to iteratively update the quantization and Huffman tables for the given training set. Here, we stick with the earlier described procedure for simplicity. The result is a strategy for optimizing the multiple quantization matrices supported in JPEG and an efficient means of describing those quantization matrices. (Entropy coding the indices of chosen quantization matrices from a fixed collection of possibilities requires far fewer bits than repeated full descriptions of quantization matrices.)

In Fig. 9, we compare the performance of the WUBA algorithm to the performance of a single bit-allocation on the same training and test sets used in the previous section. WUBA contains 64 bit-allocations and uses $n = 1$ and $l = 64$. All rates are reported in terms of entropy. WUBA achieves up to 2.5 dB improvement over single bit-allocation systems. Further, WUBA achieves up to 5 dB improvement over WUVQ with first- and second-stage vector dimensions $n = 4$ and $l = 4$, respectively, and up to 12 dB improvement over ECVQ with $l = 4$. This improvement can be attributed to the higher effective coding dimension $(l = 64)$ of the WUBA scheme. (The WUVQ and ECVQ are implemented at far lower dimensions than the WUBA due to the prohibitive computational expense associated with high dimensional full search vector quantizers.) The performance curves for the WUBA and single bit-allocation systems can both be expected to shift slightly to the left if they use a lossless code more efficient than independent entropy coding, such as zerotree coding or run-length followed by Huffman coding as in JPEG. Fig. 10 compares the images resulting from quantizing the data at around 0.2 b/pixel.

APPENDIX D
WEIGHTED UNIVERSAL PERCEPTUAL IMAGE CODING

It is well known that improvements in the mean squared quantization error of an image coder do not necessarily translate into improvements in visual quality, since the amount of quantization error perceivable by the human visual system is frequency and signal dependent. For example, humans are least tolerant of quantization noise at mid-frequencies, especially when such quantization noise is imposed on smooth, mid-luminance regions of an image. Low amplitude quantization noise in these regions may be perceived more easily than high amplitude quantization noise in other regions.

Models of human visual perception have recently been used to improve the visual quality of coded images, e.g., [8], [17]. (See [18] for an extensive review.) Image coders based on such models are called *perceptual image coders*. Most perceptual image coders to date are transform or subband coders that choose a single quantization matrix in such a way that the resulting quantization noise is least perceptible. Although the quantization matrix may be image-dependent, this approach is limited, because it makes it impossible to hide more quantization noise in some regions of the image, and less in others.

A more recent approach to perceptual image coding involves applying a prequantization step to the images before coding them with an ordinary image coder [18]. The prequantization step sets to zero all frequency components in the image that fall below the quantization matrix thresholds. Because the quantization matrix can vary over the image according to the perceptual model and the image content, this scheme makes it possible to hide extra "quantization noise" in those regions of the image that will tolerate it. Unfortunately, the extra quantization noise can be hidden only in those regions whose frequency components fall below threshold.

The basic difficulty faced in both of these approaches is that maximally hiding quantization noise in a way that varies with the image in space and frequency seems to necessitate the description of an unacceptably large amount of side information to transmit the spatially varying quantization matrices. This difficulty is overcome by WUPIC, in which the side information is optimized by entropy-constrained clustering and coding of the quantization matrices.

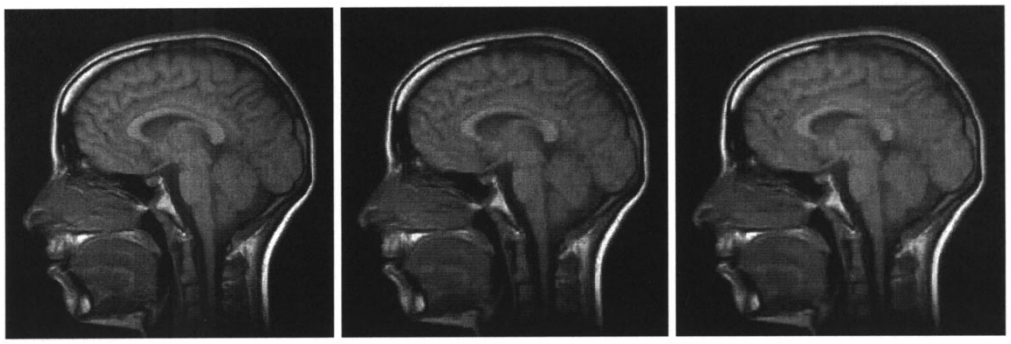


Fig. 10. Left: original image, coded image using a single bit-allocation at 0.45 b/pixel. Right: coded image using WUBA with 64 bit-allocations at 0.35 b/pixel.

To be precise, WUPIC is simply WUBA (described in Appendix C), for which the squared error distortion measure is replaced by an input-weighted squared error distortion measure, $d(\mathcal{F}_{u,v},\hat{\mathcal{F}}_{u,v}) = \Sigma_{u,v}\ w_{u,v}(\mathcal{F}_{u,v} - \hat{\mathcal{F}}_{u,v})^2$, where $\mathcal{F}_{u,v}$ is the (u,v)th transform or subband coefficient, $\hat{\mathcal{F}}_{u,v}$ is its reproduction, and $w_{u,v}$ is a weight that depends, through a perceptual model, on local image characteristics. Indeed, if a perceptual model chooses $Q_{u,v}$ as the quantization matrix desired for a particular block of the image, then $w_{u,v}$ is set to $Q_{u,v}^{-2}$ in that block. Because the decoder makes no explicit use of the distortion measure, the weighting, which may vary from block to block, need not be explicitly transmitted. However, the distortion measure will affect the design of the quantization matrices, and will affect the encoding. In this way, the desired perceptual quantization is performed. Quantization errors will be hidden in those regions of the image that can most tolerate them, from a perceptual point of view.

The perceptual model used in the experiments reported here is a typical three-part model, consisting of base threshold, luminance masking, and texture masking components. That is, $w_{u,v} = Q_{u,v}^{-2}$, where $Q_{u,v} = B_{u,v}L(\mathcal{F})T(\mathcal{F})$. In particular, we take the base threshold matrix $[B_{u,v}]$ to be the example JPEG quantization matrix described with the standard [19]:

$$B = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}.$$

We take the luminance masking function $L(\mathcal{F})$ to be a function of the DC component (DC = $\mathcal{F}_{0,0}/8$ in an 8×8 DCT) such that perceptual sensitivity is unmodified at mid-gray (DC $=$ 128), but ramps down linearly (in dB) to -6 dB at full white (DC $= 255$) and to -24 dB at full black (DC $= 0$), as follows:

$$L(\mathcal{F}) = \begin{cases} 10^{(6/20)(\mathrm{DC}-128)/128,} & \text{if DC} \geq 128 \\ 10^{(24/20)(128-\mathrm{DC})/128,} & \text{if DC} < 128 \end{cases}$$

and finally we take the the texture masking function $T(\mathcal{F})$ to be a function of the AC energy (AC $= \Sigma_{u\neq 0,v\neq 0}\ \mathcal{F}_{u,v}^2$) such that the perceptual sensitivity is unmodified with no texture (AC $= 0$), but ramps down to around -10.4 dB with increasing texture:

$$T(\mathcal{F}) = \begin{cases} (1+\sqrt{\mathrm{AC}}/16)^{-1/2,} & \text{if AC} \leq 160^2 \\ 11^{-1/2,} & \text{if AC} > 160^2. \end{cases}$$

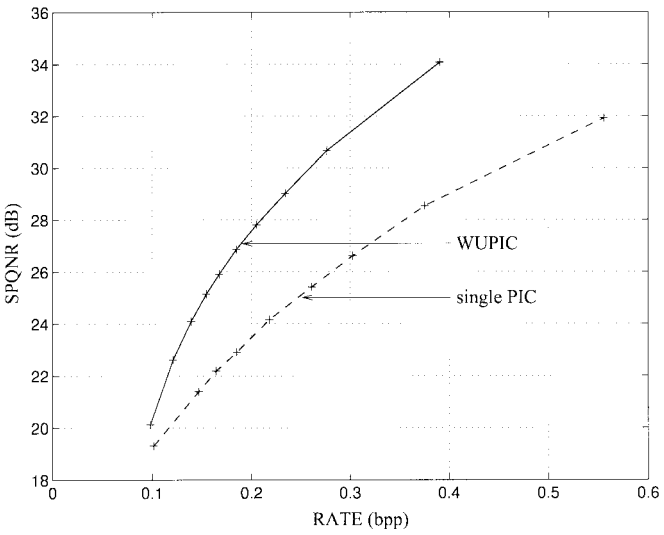


Fig. 11. Comparison of perceptual image coders with 1 and 64 quantization matrices.

The WUPIC experiments use the same training and test sets, and the same transform (DCT) and vector dimensions $(n = 1, l = 64)$, as the WUBA experiments. Fig. 11 compares the test set performance of WUPIC with 64 quantization matrices to the performance of a perceptual image coder with a single, perceptually optimal quantization matrix. In this case, distortion is reported as signal-to-perceptual quantization noise ratio (SPQNR), the ratio of the expected perceptual distortion of an optimal rate-zero quantizer to the expected perceptual distortion of the given quantizer (in dB). The results show a 5 dB improvement in perceptual distortion over a range of rates by going from 1 to 64 quantization matrices. This contrasts to the 2.5 dB improvement in mean square error (MSE) shown in the WUBA experiments (Fig. 9). Apparently, the perceptual distortion measure allows multiple codebook systems to gain an even greater advantage over single codebook systems, perhaps because the images appear more diverse under the perceptual metric than under the squared error, and hence have more to gain by using multiple codebooks. Fig. 12 shows images coded to about 0.4 b/pixel using 1 and 64 quantization

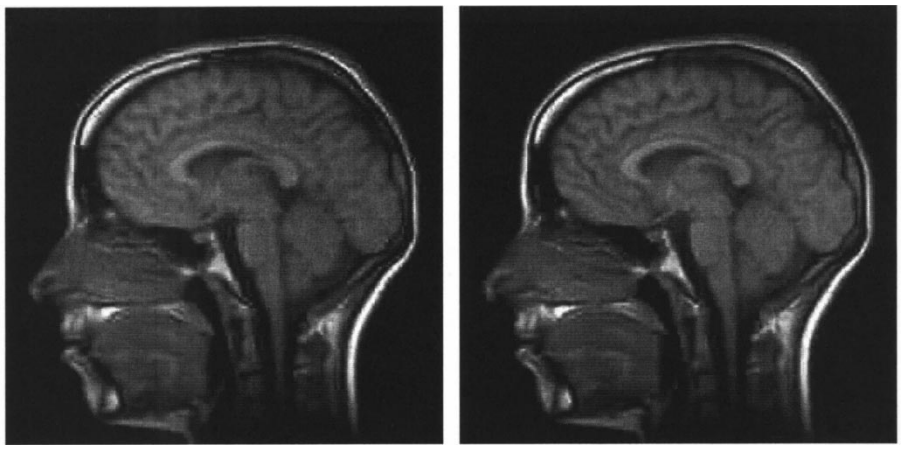


Fig. 12. The images are coded at 0.38–0.39 b/pixel using the perceptual distortion measure and (left) one and (right) up to 64 quantization matrix/entropy code pairs.

matrices, respectively, with the above perceptual distortion measure.

APPENDIX E
WEIGHTED UNIVERSAL TRANSFORM CODING

While WUBA allows algorithms like JPEG to achieve good performance at low complexity by designing a locally optimal collection of bit-allocations, the algorithm still suffers from the smooth, natural image assumption inherent to the JPEG algorithm. That is, while the DCT does a good job decorrelating the samples of images with primarily low-frequency information, its performance on images with large quantities of high-frequency information is less satisfactory. The Karhunen–Loève transform (KLT) is a data-dependent transform that achieves optimal decorrelation (all off-diagonal terms of the transformed data's covariance matrix are identically equal to zero) and optimal energy compaction [20]. (Other suboptimal data-dependent decompositions, such as wavelet packets, might also be considered.) To date, the KLT has not been popular in data compression algorithms due to its data-dependence. Relying solely on traditional techniques, the transform would have to either be chosen in advance (which means that the statistics of the data have to be known in advance) or computed during the encoding process and communicated to the decoder, which is expensive both in terms of computational complexity and in terms of rate. Given these difficulties with using the optimal transform, source code designers have, to date, relied primarily on nonoptimal, data-independent transform codes, such as the DCT used in JPEG. The strategy of course achieves less success with images that fail to match the assumptions upon which the choice of transform was based. Thus, for example, performance of the JPEG algorithm is less satisfactory on low correlation, high contrast images, such as images of text [21].

Here, we use the weighted universal code design algorithm to design multicodebook transform codes for which the performance on every source in some broad class of sources approaches the performance that would be achieved if the optimal transform code were used for every source encountered. The resulting algorithm is the WUTC. [22].

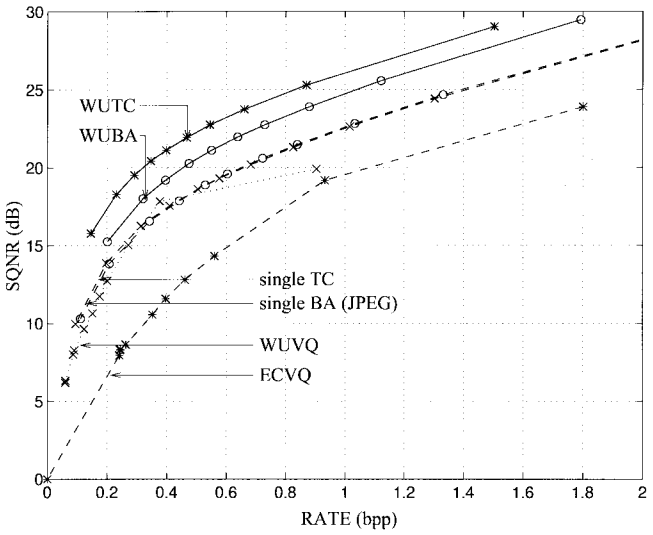


Fig. 13. Comparison of SQNR results on a collection of combined text and gray scale images.

In this case, $\mathcal{C}^l$ represents a family of transform codes, each of which contains its own transform and bit allocation. Decoding to the centroid is accomplished as follows. Given that the KLT maximizes the coding gain over all orthogonal transform codes (e.g., [23, App. C]), we here set the transform code's transform to the KLT matched to the statistics of the data to be coded. Using this choice, we accomplish the optimal decorrelation and energy compaction for the source in operation. The KLT is calculated as follows. For a given index $\tilde{s}$, let $V_{\tilde{s}}$ be the correlation matrix associated with these vectors, i.e., $V_{\tilde{s}} = (1/n)\ \Sigma_{i=1}^{n}\ E[(X_i^l)(X_i^l)'|\tilde{\alpha}(X^{ln}) = \tilde{s}]$. Then the transform $T_{\tilde{s}}^{\star}$ has, in the first row, the eigenvector corresponding to the largest eigenvalue of $V_{\tilde{s}}$, in the second row, the eigenvector corresponding to the second largest eigenvalue, and so on. Given a transform, the optimal bit-allocation may be accomplished by an optimal bit-allocation design algorithm such as the one described in the previous section.

In Fig. 13, we compare the performance of a WUTC to the performance of a single transform code. The WUTC contains 64 transform codes, and uses $n = 1, l = 64$ for all experiments.

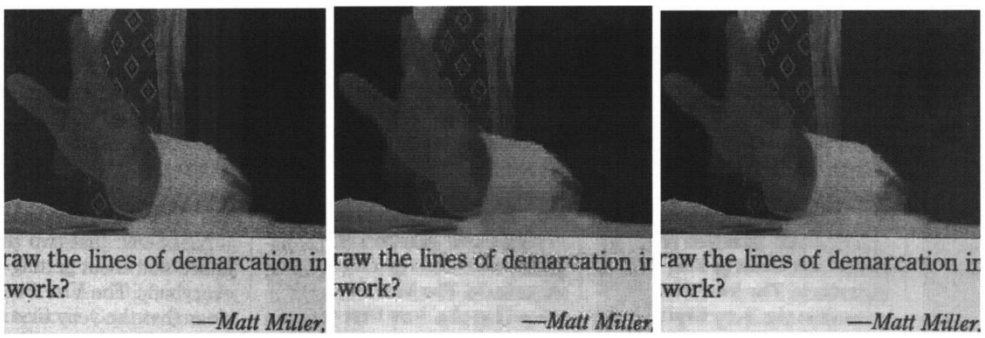


Fig. 14. Results of optimal single transform coding and optimal universal transform coding on a single mixed text and image file. From left to right: original image; image coded with optimal transform coding using a single transform and bit allocation and rate of 0.20 b/pixel; and image coded with universal transform coding using 64 transform codes and rate 0.23 b/pixel.

Each system is trained on a single 2048 pixel $\times$ 2048 pixel image scanned from a page of *IEEE Spectrum Magazine* and tested on another page from the same issue. Each page has roughly equal amounts of text and gray scale material. All rates are reported in terms of entropy. WUTC achieves up to 2 dB improvement over WUBA, up to 3 dB improvement over a single bit-allocation system (effectively equivalent to JPEG), up to 6 dB improvement over WUVQ, and up to 10 dB improvement over ECVQ. The performance curves for all transform coding systems should shift slightly to the left with application of a more efficient lossless code than the given independent entropy codes. Fig. 14 shows the performance of the WUTC and the performance of an optimal single transform code.

REFERENCES

[1] R. F. Rice and J. R. Plaunt, "The Rice machine: Television data compression," Tech. Rep. 900-408, Jet Propulsion Lab., Pasadena, CA, Sept. 1970.

[2] ——, "Adaptive variable-length coding for efficient compression of spacecraft television data," *IEEE Trans. Commun.*, vol. COMM-19, pp. 889–897, Dec. 1971.

[3] P. A. Chou, M. Effros, and R. M. Gray, "A vector quantization approach to universal noiseless coding and quantization," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1109–1138, July 1996.

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

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

[6] A. Buzo, A. H. Gray, Jr., R .M. Gray, and J .D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 562–574, Oct. 1980.

[7] L. R. Rabiner, J. G. Wilpon, and B.-H. Juang, "A segmental K-means training procedure for connected word recognition," *AT&T Tech. J.*, vol. 64, pp. 21–40, May 1986.

[8] R. J. Safranek and J. D. Johnston, "A perceptually tuned sub-band image coder with image dependent quantization and post-quantization data compression," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Glasgow, U.K., May 1989, pp. 1945–1948.

[9] P. A. Chou, "Optimal partitioning for classification and regression trees," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 340–354, Apr. 1991.

[10] W.-Y. Chan and A. Gersho, "Constrained-storage quantization of multiple vector sources by codebook sharing," *IEEE Trans. Commun.*, vol. 39, pp. 11–13, Jan. 1991.

[11] P. A. Chou and T. Lookabaugh, "Locally optimal variable-to-variable length source coding with respect to a fidelity criterion," in *Proc. IEEE Int. Symp. Information Theory*, Budapest, Hungary, June 1991, p. 238.

[12] ——, "Variable dimension vector quantization of linear predictive coefficients of speech," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Adelaide, Australia, Apr. 1994.

[13] M. Effros, P. A. Chou, and R. M. Gray, "Variable dimension weighted universal vector quantization and noiseless coding," in *Proc. IEEE Data Compression Conf.*, Snowbird, UT, Mar. 1994, pp. 2–11.

[14] R. J. Stevens, A. F. Lehar, and F. H. Preston, "Manipulation and presentation of multidimensional image data using the peano scan," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-5, pp. 520–533, Sept. 1983.

[15] M. Effros, *Universal and Adaptive Source Coding: Theory and Practice*, Ph.D. dissertation, Stanford Univ., Stanford, CA, Aug. 1994.

[16] M. Effros and P. A. Chou, "Weighted universal bit allocation," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Detroit, MI, May 1995, vol. 4, pp. 2343–2346.

[17] A. B. Watson, "Visually optimal DCT quantization matrices for individual images," in *Proc. IEEE Data Compression Conf.*. Snowbird, UT, Apr. 1993, pp. 178–187.

[18] N. S. Jayant, J. D. Johnston, and R. J. Safranek, "Signal compression based on models of human perception," *Proc. IEEE*, vol. 81, pp. 1385–1422, Oct. 1993.

[19] W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Compression Standard*. New York: Van Nostrand, 1993.

[20] A. Koschman, "On the filtering of nonstationary time series," in *Proc. Electronics Conf.*, 1954, p. 126.

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

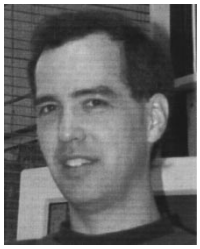
[22] M. Effros and P. A. Chou, "Weighted universal transform coding: Universal image compression with the Karhunen-Loève transform," in *Proc. IEEE Int. Conf. Image Processing*. Washington, DC, Oct. 1995.

[23] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

Michelle Effros (S'91–M'95) was born in New York, NY, in 1967. She received the B.S. degree with distinction in 1989, the M.S. degree in 1990, and the Ph.D. degree in 1994, all in electrical engineering, from Stanford University, Stanford, CA.

During the summers of 1988 and 1989 she was with Hughes Aircraft Company, researching modulation schemes, real time implementations of fast data rate error correction schemes, and future applications for fiber optics in space technology. Since 1994, she has been an Assistant Professor of electrical engineering at the California Institute of Technology, Pasadena. Her research interests include information theory, data compression, communications, pattern recognition, speech recognition, and image processing.

Dr. Effros received Stanford's Frederick Emmons Terman Engineering Scholastic Award (for excellence in engineering) in 1989, the Hughes Masters Full-Study Fellowship in 1989, the National Science Foundation Graduate Fellowship in 1990, the AT&T Ph.D. Scholarship in 1993, the NSF CAREER Award in 1995, the Charles Lee Powell Foundation Award in 1997, and the Richard Feynman-Hughes Fellowship in 1997. She is a member of Tau Beta Pi, Phi Beta Kappa, Sigma Xi, and IEEE Information Theory, Signal Processing, and Communications societies. She served as the Editor of the *IEEE Information Theory Society Newsletter* from 1995 to 1998, and has been a Member of the Board of Governors of the IEEE Information Theory Society since 1998.



Philip A. Chou (S'82–M'87) was born in Stamford, CT, on April 17, 1958. He received the B.S.E. degree from Princeton University, Princeton, NJ, in 1980, the M.S. degree from the University of California, Berkeley, in 1983, both in electrical engineering and computer science, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1988.

Since 1977, he has worked for IBM, AT&T Bell Laboratories, Princeton Plasma Physics Laboratory, Telesensory Systems, Speech Plus, Hughes, Xerox, VXtreme, and Microsoft, where he was involved variously in office automation, motion estimation in television, optical character recognition, LPC speech compression and synthesis, text-to-speech synthesis by rule, compression of digitized terrain, speech and document recognition, and video networking. His research interests are pattern recognition, data compression, and speech, image, and video processing. In 1994–1995, he was a Consulting Associate Professor at Stanford University. Currently, he is with Microsoft Corporation, Redmond, WA.

Dr. Chou serves on the editorial board of the IEEE TRANSACTIONS ON INFORMATION THEORY as an Associate Editor for Source Coding, and also serves on the IEEE Technical Committee for Image and Multidimensional Signal Processing. He is a member of Phi Beta Kappa, Tau Beta Pi, Sigma Xi, and the IEEE Computer, Information Theory, Signal Processing, and Communications societies, and was an active member of the MPEG committee. He is co-recipient (with T. Lookabaugh), of the 1993 Signal Processing Society Paper Award for a paper co-authored with R. M. Gray.

Robert M. Gray (S'68–M'69–SM'77–F'80) was born in San Diego, CA, on November 1, 1943. He received the B.S. and M.S. degrees from the Massachusetts Institute of Technology, Cambridge, in 1966, and the Ph.D. degree from the University of Southern California, Santa Barbara, in 1969, all in electrical engineering.

Since 1969, he has been with Stanford University, Stanford, CA, where he is currently Professor and Vice Chair of the Department of Electrical Engineering. His research interests are the theory and design of signal compression and classification systems. He is co-author (with L. D. Davisson) of *Random Processes* (Englewood Cliffs, NJ: Prentice-Hall, 1986), and *An Introduction to Statistical Signal Processing* (http://www-isl.stanford.edu\~gray/sp.html). He is co-author (with A. Gersho) of *Vector Quantization and Signal Compression* (Boston, MA: Kluwer, 1992), and (with J. W. Goodman) *Fourier Transforms* (Boston, MA: Kluwer, 1995). He is the author of *Probability, Random Processes, and Ergodic Properties* (Berlin, Germany: Springer-Verlag, 1988), *Source Coding Theory* (Boston, MA: Kluwer, 1990), and *Entropy and Information Theory* (Berlin, Germany: Springer-Verlag, 1990).

Dr. Gray was a member of the Board of Governors of the IEEE Information Theory Group (1974–1980, 1985–1988) as well as an Associate Editor (1977–1980) and Editor-in-Chief (1980–1983) of the IEEE TRANSACTIONS ON INFORMATION THEORY. He is currently a member of the Board of Governors of the IEEE Signal Processing Society. He was Co-Chair of the 1993 IEEE International Symposium on Information Theory and Program Co-Chair of the 1997 IEEE International Conference on Image Processing. He is an elected member of the Image and Multidimensional Signal Processing Technical Committee of the IEEE Signal Processing Society, an appointed member of the Multimedia Signal Processing Technical Committee, and a member of the Editorial Board of the *IEEE Signal Processing Magazine*. He was co-recipient (with L. D. Davisson) of the 1976 IEEE Information Theory Group Paper Award and co-recipient (with A. Buzo, A. H. Gray, and J. D. Markel) of the 1983 IEEE ASSP Senior Award. He was awarded an IEEE Centennial medal in 1984, the IEEE Signal Processing 1993 Society Award in 1994, and the IEEE Signal Processing Society Technical Achievement award and a Golden Jubilee Award for Technological Innovation from the IEEE Information Theory Society in 1998. He was elected Fellow of the Institute of Mathematical Statistics (1992) and has held fellowships from Japan Society for the Promotion of Science at the University of Osaka (1981), the Guggenheim Foundation at the University of Paris XI (1982), and NATO/Consiglio Nazionale delle Ricerche at the University of Naples (1990). During spring 1995 he was a Vinton Hayes Visiting Scholar at the Division of Applied Sciences, Harvard University. He is a member of Sigma Xi, Eta Kappa Nu, AAAS, AMS, and the Société des Ingénieurs et Scientifiques de France.