Image Coding using Transform Vector Quantization with Training Set Synthesis

Dorin Comaniciu[1] Richard Grisel[2]

[1] Real-Time Vision and Modeling Department
Siemens Corporate Research
755 College Road East, Princeton, NJ 08540
comanici@scr.siemens.com

[2]CPE-LISA
CNRS EP 0092
69616 Villeurbanne Cedex
France

Keywords: Image Coding – Training Set Synthesis – Transform Vector Quantization – Expectation-Maximization Algorithm.

Abstract

A new paradigm that combines data modeling and vector quantization in an effective coding technique is presented. We fit a statistical model to the input data and use the best fit parameters to synthesize training vector sets with statistics similar to the input. By knowing the best-fit parameters, the decoder can synthesize the same training sets, while identical codebooks are obtained at both encoder and decoder based on the same codebook generation procedure. As a result, complete codebook adaptation is achieved with a very small increase in the bit rate. The implementation of the new technique in the transform domain produced competitive results when compared to other methods relying on vector quantization and transform coding. In particular, the image Lena was coded at 0.28 bits/pixel with a peak signal to noise ratio of 32.51 dB.

Resumé

Un nouveau paradigme qui combine la modélisation des données avec le quantification vectorielle dans une technique efficace de codage est présenté. La méthode génère des "codebooks" relatifs individuellement à chaque image, donnant une forte adaptativité qui suit le comportement statistique de l'image. Les jeux de vecteurs sont calculés sur la base des statistiques des coefficients de la Transformée en Cosinus Discrète plutôt que remplies par construction classique. Ainsi, l'information transmise au décodeur consiste uniquement en paramètres de caractérisation statistique en lieu et place des références des vecteurs du "codebooks". Le décodeur synthétise le même jeu de vecteurs que ceux utilisés par le décodeur. De plus, en utilisant le même algorithme de classification, le décodeur génère un "codebook" identique à celui du codeur, et par voie de conséquence décode les indices reçus permettant la reconstruction des vecteurs au niveau du récepteur. De très bonnes qualités de reconstruction sont obtenues avec cette méthode pour des taux de compression relativement élevés. En particulier l'image test Lena est codée à 0.28 *bits/pixel et un rapport signal sur bruit crête de* 32.51 *dB.*

1 Introduction

Vector quantization (VQ) [4, 22, 23, 37] is an effective compression technique defined as an extension of scalar quantization to vectorial form. In its simplest form, VQ requires first breaking the image to be compressed into vectors which are quantized to the closest codeword of a codebook containing representative prototypes. Compression is achieved by only storing or transmitting the address of the closest codeword, instead of the whole vector of image pixels. For reconstruction, the decoder employs a codebook identical to the encoder. Transform vector

quantization (TVQ) assumes that the transform coefficients are vector quantized rather than the image pixels. The perceptual weighting in the transform domain, coefficient decorrelation, and potential use of a quality criterion matching that of the human visual system are benefits of TVQ. Block classification has been shown to improve compression performances of TVQ [28, 29, 30].

The lower distortion generated by VQ for a given bit rate is due to its boundary and granular gains over the scalar quantization [33]. Thus, the boundary of a vector quantizer is shaped to match the multivariate probability density function (PDF) of the input signal, while the boundary of a set of scalar quantizers is restricted to a hypercube shape. Also, for all dimensions greater than one, there exist sphere coverings of the Euclidean space with higher efficiency than those obtained by using integer or Z lattices, a property referred to as the granular gain.

The theoretical foundations of VQ have been developed within the framework of rate-distortion theory. It has been shown [21, 47, 51] that the performance of a vector quantizer with an arbitrarily large vector dimension can in principle approach the distortion-rate function (the minimum achievable distortion for a given rate) arbitrarily closely. The Bennett's integral which gives an approximate formula for the distortion has been recently extended from scalar to vector quantizers [36].

The key to an effective VQ system is a good codebook, usually derived from a set of training images. However, the nonstationary nature of image data often determines significant statistical difference between the image being coded and the training set the codebook was designed for [22]. Even in the case of large training sets of vectors, the input structure might not be reflected in the current codebook, which determines low reconstructed image quality. On the other side, an adaptive codebook typically requires large amounts of side information for the transmission of new codewords [20, 24, 31, 39].

This paper proposes *vector quantization with training set synthesis* (VQ-TSS) as a method to indirectly specify an adaptive codebook. We implemented VQ-TSS in the transform domain. The idea is to characterize first the transformed data by fitting a statistical model. The model parameters, called *training set parameters* (TSP), are then used to synthesize a training vector set which is an approximated replica of the original data. By knowing the TSP, the decoder can derive the same training set. Therefore, identical codebooks are obtained at both encoder and decoder sides through the same codebook generation procedure. Since the TSP size is small,

the technique results in low bit rate encoding with good reconstruction quality.

The organization of the paper is as follows. Section 2 presents the basic concepts of VQ and codebook generation, followed by a discussion of adaptive vector quantization. Section 3 defines the VQ-TSS paradigm. The implementation of VQ-TSS in the transform domain is discussed in Section 4. Experimental results and comparisons are given in Section 5. Section 6 contains the conclusions and further research directions.

2 Codebook Generation and Adaptive VQ

2.1 Definition of terms

A vector quantizer Q of dimension k and size N is defined as a mapping $Q : R^k \rightarrow C$ where R^k is the k-dimensional Euclidean space and the *codebook* C is a subset of R^k

$$C = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\} \subset R^k. \tag{1}$$

The output vectors $\{\mathbf{y}_i \in R^k\}_{i=1,\ldots,N}$ are referred to as *codewords*. Any N point vector quantizer defines a *partition* of R^k into N regions $\{S_i\}_{i=1,\ldots,N}$ with the ith region given by

$$S_i = \left\{\mathbf{x} \in R^k \mid Q(\mathbf{x}) = \mathbf{y}_i\right\} \equiv Q^{-1}(\mathbf{y}_i). \tag{2}$$

The *resolution* or *bit rate* of a vector quantizer is defined as the number of bits per vector component used to represent the input vector

$$R_{VQ} = \frac{\log_2 N}{k}. \tag{3}$$

For coding purposes, a VQ codec has two parts: an encoder and a decoder, each one equipped with the same codebook. The encoder assigns to each input vector $\mathbf{x} = (x_1, \ldots x_k)^\top$ an index i that points to the closest codeword $\mathbf{y}_i = (y_{i1}, \ldots y_{ik})^\top$ in the codebook, while the decoder employs the received index to extract the codeword from the codebook. The transmission rate is controlled by choosing the size of the codebook. The distortion between the input vector and its corresponding codeword is usually measured by the root mean squared error ($RMSE$) defined as

$$RMSE = d(\mathbf{x}, \mathbf{y}_i) = \left(\frac{1}{k}\|\mathbf{x} - \mathbf{y}_i\|^2\right)^{1/2} = \left(\frac{1}{k}\sum_{j=1}^{k}(x_j - y_{ij})^2\right)^{1/2}. \tag{4}$$

2.2 Codebook Generation

Once a distortion measure has been defined, the problem of vector quantization is to choose the partition and the codebook such that to minimize the overall distortion for the class of images to be processed. The most used method for populating the codebook is the generalized Lloyd algorithm (GLA) [32], an iterative procedure that starts with an initial codebook and monotonically decreases the distortion function towards a local minimum. Although conceptually simple and easy to implement, the GLA algorithm is not computationally efficient due to its exhaustive checking of every training vector against every codeword for the closest match.

Faster variants of GLA and other distinct codebook generation methods with increased efficiency have been developed [7, 11, 17]. In addition, a number of algorithms have been proposed that improve the codebook quality at the expense of increasing the computational complexity [27, 53]. Within this paper, we employ GLA to generate small size codebooks. However, for larger codebooks we take advantage of the particular structure of training vector sets, by using efficient partial search during the encoding step of GLA (see Section 4.5).

2.3 Adaptive Vector Quantization

Adaptive vector quantization (AVQ) assumes that the codebook is modified according to changes in the statistics of the input signal. The adaptation of the codebook is implemented either through updating the codewords in the current codebook, i.e., codebook transmission, or through switching between different codebooks which are known by both encoder and decoder. AVQ represents a solution of the quantizer mismatch problem [25]. The concordance between the PDF the quantizer is designed for and the input data PDF is particularly important for fixed-rate coding schemes (schemes not employing entropy coding). Experiments regarding PDF shape mismatch show strong decrease in performance when mismatches occur [34]. Fig 1

The block diagram of AVQ with codebook transmission [20] is shown in Figure 1. The input image is partitioned into a number of subimages, each one consisting of n vectors of dimension k. For each subimage, the locally generated codebook of size N is transmitted, followed by the codeword indices of the vectors in subimage. If b bits are used to represent a vector component (that is, b bits/pixel), the average bit rate R_T is given by the vector quantizer resolution R_{VQ} and the bit rate R_{CT} resulting from codebook transmission

$$R_T = R_{VQ} + R_{CT} = \frac{\log_2 N}{k} + \frac{bN}{n}. \tag{5}$$

The tradeoff between the two terms in (5) is analyzed in [52] using the high resolution quantization theory. Since R_{CT} is proportional with the codebook size N, the codebook transmission is effective only for small values of N.

A different solution is to use a large universal codebook that is designed off-line and populate the adaptive codebook with universal codewords [39, 31]. The universal codebook is also available at the decoder, therefore, the side information is represented only by the indices specifying the chosen codewords from the universal codebook. However, since the codewords of the adaptive codebook are restricted to universal codewords, an increase of the reconstruction distortion is expected in the case of significant changes in the input statistics.

3 Vector Quantization with Training Set Synthesis

The main contribution of this paper is a method to achieve the codebook adaptation to the input statistics with a very small overhead. The new compression paradigm, called *vector quantization with training set synthesis* (VQ-TSS), is shown in Figure 2. Fig 2

At the encoding side (Figure 2a) a statistical model is first fitted to the input data. The best-fit parameters, named training set parameters (TSP), are used to synthesize a training set (TS) with statistics similar to the input. The codebook C, populated according to the generalized Lloyd algorithm, is then employed to vector quantize the input data. Only the set of codeword indices I and the TSP are stored or transmitted.

Figure 2b presents the decoding side. The received TSP are used to synthesize the TS which is further employed to derive the codebook. An approximate reconstruction of the original data is finally obtained based on the indices I and the codebook C.

The VQ-TSS advantage is that very little side information, represented by the TSP, has to be transmitted. Thus, the complete codebook adaptation is accomplished with a small increase in the bit rate.

4 Transform VQ-TSS

The implementation of VQ-TSS should take into account that both the data modeling and codebook generation are performed on-line, hence, the associated procedures should be fast. We describe and analyze below a DCT [1, 2] domain implementation of the proposed method, called *transform VQ-TSS* (TVQ-TSS). The modeling of the transformed coefficients is less

complex, since they are (almost) decorrelated and have typically highly peaked histograms centered around zero. A different solution is to combine subband or wavelet decomposition with VQ-TSS, which is the subject of future work.

Unless otherwise specified, the numerical and graphical examples in this section correspond to the 512×512 gray level image *Lena*.

4.1 Encoder and Decoder Overview

Figure 3a presents the block diagram of the TVQ-TSS encoder which includes the following operations:

- The input image is partitioned into $B \times B$ blocks and 2-dimensional DCT is computed for each block.
- The DC coefficient is uniformly quantized and the resulting values are DPCM encoded and transmitted.
- A classified vector quantizer is used to classify the transform blocks into n_C equally populated classes while the bits are allocated according to Section 4.2. The bit allocation map (described in Section 4.3) and the class indices are transmitted to the decoder.
- The TSP are estimated through the expectation maximization (EM) algorithm [41] and the codebooks are derived as described in Section 4.4 and 4.5. The vector quantization of the transform data yields the set of codeword indices I which are transmitted together with the TSP.
- Finally, error analysis and reduction is performed according to Section 4.6 and the error information is transmitted.

Fig 3

Figure 3b presents the block diagram of the TVQ-TSS decoder. The processing starts with training set synthesis based on the received TSP, followed by codebook derivation and decoding of the codeword indices I. The error information is then used to reduce a selected set of errors. The inverse transformation of the corrected data produces an approximated replica of the original image.

4.2 Transform Block Classification and Bit Allocation Scheme

Let us consider the DCT of the $B \times B$ blocks representing the input image. Following the usual notation we denote the coefficient in the upper-left corner of a block as the DC coefficient, while the remaining coefficients are named AC coefficients.

To increase the adaptation, we use the procedure described in [8] which classifies the transformed blocks into n_C equally populated classes according to the energy of AC coefficients. Bits are then distributed between busy (high AC energy) and quiet (low AC energy) image areas. The overhead information in bits/pixel due to block classification is

$$R_{BC} = \left(\lceil \log_2(n_C) \rceil\right) / B^2 \tag{6}$$

where $\lceil \cdot \rceil$ is the ceiling of the argument. For example, in the case of $B = 8$ and $n_C = 4$, the overhead due to class information is

$$R_{BC} = \frac{2}{64} = 0.031 \text{ bits/pixel.} \tag{7}$$

A bit allocation scheme that gives real and positive bit rates is derived in [43] by supposing that each vector component is optimally encoded (in the distortion-rate sense). The overall distortion is minimized subject to the positive allocation restriction and an imposed bit rate R_{AC}. The scheme assigns to the AC coefficient in position (u, v), belonging to class c, and having the variance $\sigma_c^2(u, v)$, a number of bits equal to

$$R_c(u,v) = \left\{ \begin{array}{ll} \frac{1}{2}\log_2 \frac{\sigma_c^2(u,v)}{\theta^*}, & \text{if } 0 < \theta^* \leq \sigma_c^2(u,v) \\ 0 & \text{otherwise} \end{array} \right\} = \max\left(0, \frac{1}{2}\log_2 \frac{\sigma_c^2(u,v)}{\theta^*}\right) \tag{8}$$

where θ^* is the solution of

$$\frac{1}{2} \sum_{\theta^* \leq \sigma_c^2(u,v)} \log_2 \frac{\sigma_c^2(u,v)}{\theta^*} = R_{AC}. \tag{9}$$

We compute the bit allocation for each vector of DCT coefficients by summing the values derived in equation (8) and rounding the result.

Note that (8) and (9) give optimum allocation only in the ideal case when the rate-distortion bound is achieved. We however preferred the above formulation to more complex algorithms such as the one described in [45].

4.3 Vector Formation

Each energy class is treated separately after the allocation of bits. The vector formation in the case of $B = 8$ is shown in Figure 4. The DCT block is decomposed into the DC coefficient and 17

vectors taken in zigzag order and denoted by $\mathbf{v}_1 = (AC_1, AC_2)^\top$, $\mathbf{v}_2 = (AC_3, AC_4, AC_5)^\top, \ldots,$ $\mathbf{v}_{17} = (AC_{61}, AC_{62}, AC_{63})^\top$. As one can see, the maximum vector dimension is $k_{max} = 4$, due to efficiency constraints resulting from the GLA algorithm (we explain this limitation in Section 4.5). In addition, the first two vectors have lower dimension (2 and 3, respectively) to limit the size of the corresponding codebook. The reason is that most of the energy is typically concentrated on the first transform coefficients, which results in a large number of allocated bits for these coefficients. Recall that the size of a codebook depends exponentially on the number of bits allocated for a certain vector, which is equal to the sum of all allocations received by the vector components. Fig 4

Table 1 gives two examples of bit allocation based on equations (8) and (9). For both cases, the resulting bit rate is very close to the imposed bit rate R_{AC}. Note that the vectors with indices greater than 4 and 8, respectively, have zero bit allocation for each energy class. Table 1

4.4 Modeling the AC Coefficients

The analysis of AC coefficients is thoroughly covered in the literature. Most of the methods assume statistically independent coefficients and model them with Laplacian distribution [42], Gaussian or Gamma [38], Generalized Gaussian [6], or Mixture of Gaussian Distributions [12]. The continuous asymptotic equipartition property theorem [15] has been used in conjunction with the statistical analysis to design lattice vector quantizers with large vector dimensions [9, 19].

Since parameter estimation is performed on-line, the multivariate analysis is too computationally expensive. In addition, a multivariate model results in a large number of parameters whose transmission would increase the bit rate. The strongest argument, however, in favor of univariate models is that the AC coefficients are almost decorrelated. As a consequence, by neglecting the nonlinear dependencies among the coefficients (whatever dependencies remain after the correlation is removed), they can be approximated as statistically independent random variables.

We can improve this assertion by enhancing the vector formation procedure described in Section 4.3 with a simple rule that derives the vectors from coefficients belonging to spatially separated blocks [5]. Thus, if the first vector component is filled with a coefficient from the ith block, the second component is filled with a coefficient from the $(i+p)$th block, the third with a coefficient from $(i+2p)$th block, and so forth, where p is the separation distance. As a result,

the vector components are close to statistical independence.

We further assume that the underlying coefficient density is a *Mixture of Gaussian Distributions* (MGD). The MGD model has more degrees of freedom, therefore, it captures the input statistics better than models relying on one elementary distribution [18]. According to MGD, if $\mathbf{x} = (x_1, \ldots, x_k)^\top$ is a vector of AC coefficients resulting from the above vector formation, the PDF of its jth component is given by

$$f_j(x) = \sum_{m=1}^{M_j} P_{jm} g_{jm}(x), \quad j = 1, \ldots, k, \tag{10}$$

where M_j is the number of Gaussians employed in modeling and g_{jm} is the Gaussian distribution having a priori probability P_{jm}, mean μ_{jm}, and variance σ_{jm}^2, with

$$\sum_{m=1}^{M_j} P_{jm} = 1. \tag{11}$$

The maximum-likelihood estimates of $\left\{P_{jm}, \mu_{jm}, \sigma_{jm}^2\right\}$ with $m = 1, \ldots, M_j$, are part of the training set parameters and are obtained by differentiating the logarithm of likelihood function. The iterative procedure that solves the likelihood equations is the EM algorithm, described in Appendix A. Note that the MGD model and EM algorithm have recently been employed for clustering [54] and texture processing [40].

The derivation of the best value for M_j (in the maximum likelihood sense) requires multiple runs of the EM algorithm, which induces additional complexity. Therefore, we have off-line selected M_j as being equal to the number of Gaussians that maximize the compression performance. For most of the images we tested, $M_j = M = 4$ proved to be a good solution.

The estimated parameters corresponding to the first and second AC coefficients of the highest energy class are shown in Table 2b and Table 2d, respectively. The values used for parameter initialization were computed as in Appendix A and are presented in Table 2a and 2c, respectively. Table 2

Figure 5 shows the PDFs of the same coefficients derived with equation (10). In the same figure, we compare the MGD result with estimates obtained through nonparametric analysis with the optimal Epanechnikov kernel [13] (the optimal kernel is presented in Appendix B). The two curves are very close to each other. In addition, the PDF of the first coefficient is bimodal and asymmetric, which justifies the modeling based on a mixture of distributions. Fig 5

Following the MGD, the modeling of one coefficient yields $3M - 1 = 11$ parameters which have to be transmitted along with the minimum and maximum values of that coefficient. Thus,

only 13 parameters represent the overhead per coefficient paid to specify the adaptive codebook at the receiver. We note that for codebooks smaller than 13 codewords (that is, for vectors with $1, 2$, or 3 bits allocated) it is more efficient to use codebook transmission instead of modeling and synthesizing the training set. However, for larger codebooks, VQ-TTS is superior.

In conclusion, a given AC coefficient undergoes one of the following actions, according to the number of bits allocated to its vector:

- Modeling and quantization based on a codebook derived from the synthesized training set, when more than 3 bits are allocated.
- Quantization based on a codebook derived from the real data, when $1, 2$, or 3 bits are allocated.
- No action, when there are no bits allocated.

4.5 Training Set Synthesis and Codebook Generation

The joint PDF of a vector $\mathbf{x} = (x_1, \ldots, x_k)^\top$ whose components are assumed to be statistically independent is equal to the product of marginal densities. With the marginal densities modeled according to (10), the joint PDF of $\mathbf{x}$ is given by

$$f(\mathbf{x}) = \prod_{j=1}^{k} f_j(x) = \prod_{j=1}^{k} \sum_{m=1}^{M} P_{jm} g_{jm}(x). \tag{12}$$

The joint PDF corresponding to the vector $\mathbf{v}_1 = (AC_1, AC_2)^\top$ (see Figure 4) of the highest energy class is presented in Figure 6a. For comparison, Figure 6b shows the 2-dimensional Epanechnikov density estimate of the same data. Observe that the two surfaces have the same global features, each exhibiting two significant modes. Fig 6

To generate a training set whose underlying distribution is approximated by (12), we uniformly sample the space covered by $\mathbf{x}$ using a k-dimensional cubic lattice $\{\mathbf{x}_q\}_{q=1:L}$ with minimum point separation Δ. Then, we associate to each lattice point $\mathbf{x}_q$ the weight $f(\mathbf{x}_q)$. If $[x_{j,min}, x_{j,max}]$ is the range of values for the jth component of $\mathbf{x}$, then the number of samples for dimension j is $l_j = \lfloor (x_{j,max} - x_{j,min})/\Delta \rfloor$, where $\lfloor \cdot \rfloor$ is the down-rounded integer. The number of lattice points L is the product of the number of samples for each dimension

$$L = \prod_{j=1}^{k} \lfloor (x_{j,max} - x_{j,min})/\Delta \rfloor. \tag{13}$$

To reduce the error caused by sampling, the value of Δ should be small. However, equation (13) shows that the number of lattice points is inversely proportional to the kth power of Δ. Since the number of lattice points L and the vector dimensionality k directly determine the speed of the GLA algorithm for codebook generation [44], we limited their values to $L_{max} = 50,000$ and $k_{max} = 4$, which induced an overall compression/decompression time of only a few seconds. As an example, for the first three vectors (of dimension 2, 3, and 4) of the highest energy class derived from image *Lena*, the value of Δ resulting from (13) is 5, 14, and 27, respectively.

The lattice points and their weights constitute the training set used as input to the GLA algorithm. An efficient prediction-based implementation of GLA can be achieved by taking into account that the lattice points form an ordered and uniformly spaced set. Thus, there is a high probability that two lattice points with successive indices are allocated to the same codeword. The search for the closest codeword to the current lattice point can therefore be performed in a small neighborhood of the codeword associated with the previous lattice point. This technique saves a lot of unnecessary operations.

Fig 7
Table 3

Figure 7 presents two codebooks derived from synthesized training sets. They correspond to the vectors $\mathbf{v}_1 = (AC_1, AC_2)^\top$ and $\mathbf{v}_2 = (AC_3, AC_4, AC_5)^\top$ of the highest energy class of image *Lena*. The number of codewords in each codebook is according to the bit allocation scheme from Table 1a. Based on these two codebooks the vector quantization of the data corresponding to $\mathbf{v}_1$ and $\mathbf{v}_2$ yielded an error ($RMSE$) of 16.88 and 20.12, respectively. When actual data rather than synthesized vectors are used as the training set, the quantization errors become 16.45 and 18.67, respectively. The latter errors are very close to the former (0.22 dB and 0.65 dB), which shows that good quality codebooks can be populated based on synthesized training sets. Additional comparisons expressed in dB are presented in Table 3 for various images from our test set.

4.6 Error Analysis and Reduction

To further control the errors [35] generated by the quantization process, the largest E errors are considered and their positions inside the DCT blocks are coded and transmitted. The error reduction is achieved using 2 correcting values (one positive and one negative) for all errors. Each DCT block has a one-bit flag that shows whether inside the block corrections are operated or not.

Two additional bits are required for each correction. One indicates the sign of correction.

The other shows whether the next correction belongs to the same block as the current correction. In the case of image *Lena* of 512×512 pixels, the overhead required by the reduction of $E = 1024$ errors is

$$R_{ER} = \frac{1}{64} + \frac{1024 \times (6+2)}{512 \times 512} = 0.047 \text{ bits/pixel.} \tag{14}$$

The first term in (14) is due to the error flag, while the second term is due to the 6 bit-address specifying the position in the block plus the two correction bits.

4.7 Overall Bit Rate

When a uniform quantizer of 7 bits is used to process the DC coefficient of image *Lena*, the DC information results in a bit rate of

$$R_{DC} = 0.096 \text{ bits/pixel.} \tag{15}$$

The overall bit rate is computed by using (7), (14), and (15)

$$\begin{aligned} R_{OV} &= R_{AC} + R_{BC} + R_{DC} + R_{TSP} + R_{ER} \\ &= R_{AC} + 0.031 + 0.096 + 0.01 + 0.047 \approx R_{AC} + 0.18 \text{ bits/pixel} \end{aligned} \tag{16}$$

where we assumed that the bit rate R_{TSP} required by the TSP is about 0.01 bits/pixel. The allocated bit rate for the AC coefficients R_{AC} is used to control the overall bit rate.

5 Experimental Results and Comparisons

A compression/decompression module was implemented in C according to the encoder and decoder diagrams from Figure 3. We tested the new compression method on a Sun Ultra 60 Workstation. The 20 test images used are available via anonymous ftp to *whitechapel.media.mit.edu* under */pub/testimages*. They are all 512×512 pixel monochrome still images with 256 gray levels. As a compression quality measure we employed the peak signal to noise ratio ($PSNR$) which is expressed in dB as

$$PSNR = 20 \log_{10} \frac{255}{RMSE} \tag{17}$$

where $RMSE$ is the root mean squared error between the original and the reconstructed image.

A first set of results is presented in Table 4 containing the $PSNR$s of the images in the test set after compression/decompression at 0.28 bits/pixel. The coding time for one image was less then 5 seconds while the decoding took about 3 seconds. Table 4

The $PSNR$ values for the images *Goldhill* and *Lena* coded at different bit rates are given in Table 5. Figure 8 and 9 show the originals and encoded images at 0.25, 0.35, and 0.5 bits/pixel, respectively.

Table 5
Fig 8
Fig 9
Fig 10

The PSNR-based comparisons presented in Figure 10 show that TVQ-TSS performance is better that that of JPEG standard (sequential encoding mode). The improvement in PSNR is almost 1 dB for the *Lena* image. One can also observe that (for the same image) our method performs better than other three recent techniques which employ vector quantization of the DCT coefficients (classified VQ in the transform domain [29], VQ with variable block-size [30], and additive vector decoding of transform coded images [50]). Although the computational complexity is higher for TVQ-TSS, the processing time is only a few seconds on a standard workstation.

It is worth-noting that although TVQ-TSS is based on a fixed-rate allocation scheme, it performs very close to techniques based on variable rate VQ (see for example the entropy-coded lattice vector quantizer reported in [48]). However, while fixed-rate coding is often desirable, improved rate-distortion performance is possible with a variable rate scheme [22]. We therefore expect that a scheme combining the VQ-TSS principles with variable-rate coding would achieve even better results. Such a scheme would be based on entropy-constrained algorithms for codebook design [10].

6 Conclusions

This paper introduced VQ-TSS as a new method to achieve codebook adaptation to the input statistics with a small amount of side information. We presented a transform domain implementation of VQ-TSS, called TVQ-TSS, and showed that the performance of TVQ-TSS is competitive with other compression schemes based on VQ and transform coding. Further related research includes variable-rate coding based on VQ-TSS, and the implementation of VQ-TSS in the subband and wavelet domains [3, 14, 16, 46, 49].

Acknowledgment

The authors wish to thank Professor Michel Jourlin of CPE Lyon for his help and encouragement. Part of this work was supported by the Regional Council of Rhône-Alpes. We would also like thank the anonymous reviewers whose comments improved the quality of this manuscript.

Appendix A

The estimation of parameters for a Mixture of Gaussian Distributions has been first derived in [26]. Let us assume that n observations are taken from a mixture of M Gaussian subpopulations, where the value of M is known. The problem is to find the maximum-likelihood estimates of the a priori probabilities P_m, means μ_m, and variances σ_m^2, where $m = 1, \ldots, M$ and $\sum_{m=1}^{M} P_m = 1$. By taking the partial derivatives of the likelihood function and setting them equal to zero a set of nonlinear equations is obtained. The EM algorithm solves these equations using an iterative approach.

The expectation step involves the computation of the conditioned a posteriori probability that the ith observation of value x_i belongs to the mth Gaussian subpopulation, given x_i

$$P(m|x_i) = \frac{P_m g_m(x_i)}{\sum_{m=1}^{M} P_m g_m(x_i)}, \tag{A.1}$$

where g_m is the Gaussian of mean μ_m and variance σ_m^2.

The maximization step updates the values of P_m, μ_m, σ_m^2

$$P_m = \frac{1}{n} \sum_{i=1}^{n} P(m|x_i) \tag{A.2}$$

$$\mu_m = \frac{1}{nP_m} \sum_{i=1}^{n} P(m|x_i) x_i \tag{A.3}$$

$$\sigma_m^2 = \frac{1}{nP_m} \sum_{i=1}^{n} P(m|x_i)(x_i - \mu_m)^2. \tag{A.4}$$

For all experiments, we initialize the EM algorithm as following. The histogram of the n observations is divided into M parts corresponding to M equally populated sets. Then the mean μ_m and σ_m^2 is computed for each set. Since the sets are equally populated, the starting a priori probabilities have the same value, $P_m = \frac{1}{M}$.

When EM is used for modeling the AC coefficients, a faster histogram-based implementation is possible by taking into account that the coefficients have integer values in a the relatively small range. Let us denote by x_{min} and x_{max} the minimum and maximum values, and by $h : [x_{min}, x_{max}] \rightarrow \{0, 1 \ldots\}$ the histogram of the observations.

Then we can rewrite the expectation equation (A.1) as

$$P(m|j) = \frac{P_m g_m(j)}{\sum_{m=1}^{M} P_m g_m(j)}, \tag{A.5}$$

where $j \in [x_{min}, x_{max}]$.

The maximization equations (A.2), (A.3), and (A.4) become

$$P_m = \frac{1}{(x_{max} - x_{min} + 1)} \sum_{j=x_{min}}^{x_{max}} P(m|j)h(j) \tag{A.6}$$

$$\mu_m = \frac{1}{(x_{max} - x_{min} + 1)P_m} \sum_{j=x_{min}}^{x_{max}} P(m|j)h(j)j \tag{A.7}$$

$$\sigma_m^2 = \frac{1}{(x_{max} - x_{min} + 1)P_m} \sum_{j=x_{min}}^{x_{max}} P(m|j)h(j)(j - \mu_m)^2. \tag{A.8}$$

The second set of EM equations is faster when the range $(x_{max} - x_{min} + 1)$ of the observed values is smaller than n, the number of observations. This property is generally true for the AC coefficients.

Appendix B

Let $\{\mathbf{x}_i\}_{i=1,\ldots,n}$ be an arbitrary set of n vectors of dimension k. The kernel density estimate with kernel K and window width h computed in $\mathbf{x}$ is given by

$$f(\mathbf{x}) = \frac{1}{nh^k} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right). \tag{B.1}$$

The kernel that yields the minimum mean integrated square error is the Epanechnikov kernel

$$K_E(\mathbf{x}) = \begin{cases} \frac{1}{2}c_k^{-1}(k+2)(1 - \mathbf{x}^T\mathbf{x}) & \text{if } \mathbf{x}^T\mathbf{x} < 1 \\ 0 & \text{otherwise} \end{cases} \tag{B.2}$$

where c_k is the volume of the unit k-dimensional sphere.

For the 1-dimensional case, the Epanechnikov density estimate of the scalars $\{x_i\}_{i=1,\ldots,n}$ is

$$f(x) = \frac{3}{4nh} \sum_{i=1}^{n} \left(1 - \frac{(x - x_i)^2}{h^2}\right). \tag{B.3}$$

References

[1] N. Ahmed, T. Natarajan, and K.R. Rao, "Discrete cosine transform", *IEEE Trans. Comput.*, Vol. C-23, January 1974, pp. 90-93.

[2] K. Aizawa, H. Harashima, and H. Miyakawa, "Adaptive discrete cosine transform coding with vector quantization for color images", *Proc. IEEE Int. Conf. ASSP*, Tokyo, Japan, Vol. 2, April 1986, pp. 985-988.

[3] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform", *IEEE Trans. Image Processing*, Vol. 1, April 1992, pp. 205-220.

[4] R.L. Baker and R.M. Gray, "Image compression using non-adaptive spatial vector quantization", *Proc. 16th Asilomar Conf. Circuits, Syst. Comput.*, October 1982, pp. 55-61.

[5] F. Bellifemine and R. Picco, "Video signal coding with DCT and vector quantization", *IEEE Trans. Commun.*, Vol. 42, February/March/April 1994, pp. 200-207.

[6] K.A. Birney and T.R. Fisher, "On the modeling of DCT and subband image data for compression", *IEEE Trans. Image Processing*, Vol. 4, February 1995, pp. 186-193.

[7] C.K. Chan and C.K. Ma, "A fast method of designing better codebooks for image vector quantization", *IEEE Trans. on Commun.*, Vol. 42, February/March/April 1994, pp. 237-242.

[8] W.H. Chen and H. Smith, "Adaptive coding of monochrome and color images", *IEEE Trans. on Commun.*, Vol. COM-25, November 1977, pp. 1285-1292.

[9] F. Chen, Z. Gao, and J. Villasenor, ''Lattice vector quantization of generalized gaussian sources", *IEEE Trans. Info. Theory*, vol. IT-43, January 1997, pp. 92-103.

[10] P.A. Chou, T. Lookabaugh, and R.M. Gray, "Entropy-constrained vector quantization", *IEEE Trans. ASSP*, Vol. 37, January 1989, pp. 31-42.

[11] D. Comaniciu, "An efficient clustering algorithm for vector quantization", *Proc. 9th Scandinavian Conf. Image Analyis*, Uppsala, Sweden, June 1995, pp. 423-430.

[12] D. Comaniciu, R. Grisel, and F. Astrade, "Medical image compression using mixture distributions and optimal quantizers", *Proc. IASTED Int. Conf. on Signal and Image Processing*, Las Vegas, November 1995, pp. 89-92.

[13] D. Comaniciu and P. Meer, "Distribution free decomposition of multivariate data", *Pattern Analysis and Applications*, Vol. 2, No. 1, 1999, pp. 22-30.

[14] P.C. Cosman, R.M. Gray, and M. Vetterli, "Vector quantization of image subbands: A survey", *IEEE Trans. Image Processing*, Vol. 5, February 1996, pp. 202-225.

[15] T.M. Cover and J.A. Thomas, *Elements of Information Theory*, John Wiley & Sons, New York, 1991.

[16] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis", *IEEE Trans. Inform. Theory*, Vol. 36, May 1990, pp. 961-1005.

[17] W.H.Equitz, "A new vector quantization clustering algorithm", *IEEE Trans. ASSP*, Vol. 37, October 1989, pp. 1568-1575.

[18] T. Eude, R. Grisel, H. Cherifi, and R. Debrie, "On the distribution of the DCT coefficients", *Proc. IEEE Int. Conf. ASSP*, Adelaide, Australia, April 1994, pp. V.365-V.368.

[19] T.R. Fisher, "A pyramid vector quantizer", *IEEE Trans. Info. Theory*, vol. IT-32, July 1986, pp. 568-583.

[20] M. Goldberg, P.R. Boucher, and S. Shlien, "Image compression using adaptive vector quantization", IEEE Trans. Commun., Vol. COM-34, February 1986, pp. 180-187.

[21] A. Gersho, "Asymptotically optimal block quantization", *IEEE Trans. Inform. Theory*, Vol. IT-25, July 1979, pp. 373-380.

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

[23] A. Gersho and B. Ramamurthi, "Image coding using vector quantization", *Proc. IEEE Int. Conf. ASSP*, Vol. 1, May 1982, pp. 428-431.

[24] A. Gersho and M. Yano, "Adaptive vector quantization by progressive codevector replacement," *Proc. IEEE Int. Conf. ASSP*, 1985, pp. 4.6.1-4.6.4.

[25] N.S. Jayant and P. Noll, *Digital Coding of Waveforms*, Prentice-Hall Int., New Jersey, 1984.

[26] V. Hasselblad, "Estimation of parameters for a mixture of normal distributions", *Technometrics*, Vol. 8, August 1966, pp. 431-444.

[27] C.M. Huang and R.W. Harris, "A comparison of several vector quantization codebook generation approaches", *IEEE Trans. Image Processing*, Vol. 2, January 1993, pp. 108-112.

[28] D.S. Kim and S.H. Lee, "Image vector quantizer based on a classification in the DCT domain", *IEEE Trans. Commun.*, Vol. 38, April 1991, pp. 549-556.

[29] J.W. Kim and S.U. Lee, "A transform domain classifier vector quantizer for image coding", *IEEE Trans. Circuits Syst. Video Technol.*, No. 2, 1992, pp. 3-14.

[30] M.H. Lee and G. Crebbin, "Classified vector quantization with variable block-size DCT models", *IEE Proc. Vis. Image Signal Processing*, Vol. 141, February 1994, pp. 39-48.

[31] M. Lightstone and S.K. Mitra, "Image-adaptive vector quantization in an entropy-constrained framework", *IEEE Trans. Image Processing*, Vol. 6, June 1997, pp. 441-450.

[32] Y. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantizer design", *IEEE Trans. Commun.*, Vol. COM-28, January 1980, pp. 84-95.

[33] T.D. Lookabaugh and R.M. Gray, "High-resolution quantization theory and the vector quantizer advantage", *IEEE Trans. Inform. Theory*, Vol. 35, September 1989, pp. 1020-1033.

[34] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding", *Proc. IEEE*, Vol. 73, November 1985, pp. 1551-1588.

[35] M. Miyahara and K. Kotani, "Block distortion in orthogonal transform coding - analysis, minimization, and distortion measure", *IEEE Trans. Commun.*, Vol. COM-33, January 1985, pp. 90-96.

[36] S. Na and D.L. Neuhoff, "Bennett's integral for vector quantizers", *IEEE Trans. Inform. Theory*, Vol. 41, July 1995, pp. 886-900.

[37] N.M. Nasrabadi and R.A. King, "Image coding using vector quantization: A review", *IEEE Trans. Commun.*, Vol. 36, August 1988, pp. 957-971.

[38] A.N. Netravali and B.G. Haskell, *Digital Pictures, Representation and Compression*, Plenum Press, New York, 1989.

[39] S. Panchanathan and M. Goldberg, "Algorithms and architecture for image adaptive vector quantization", *Proc. SPIE Visual Commun. Image Processing*, Vol. 1001, November 1988, pp. 336-344.

[40] K. Popat and R.W. Picard, "Cluster-based probability model and its application to image and texture processing", *IEEE Trans. Image Processing*, Vol. 6, February 1997, pp. 268-284.

[41] R.A. Redner and H.F. Walker, "Mixture densities, maximum likelihood and the EM algorithm", *SIAM Review*, Vol. 26, April 1984, pp. 195-239.

[42] R.C. Reininger and J. Gibson, "Distribution of the two-dimensional DCT coefficients for images", *IEEE Trans. Commun.*, Vol. COM-31, June 1983, pp. 835-839.

[43] A. Segall, "Bit allocation and encoding for vector sources", *IEEE Trans. Inform. Theory*, Vol. IT-22, March 1976, pp. 162-169.

[44] J Shanbehzadeh and P.O. Ogunbona, "On the computational complexity of the LBG and PNN algorithms", *IEEE Trans. Image Processing*, Vol. 6, April 1997, pp. 614-616.

[45] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers", *IEEE Trans. ASSP*, Vol. ASSP-36, September 1988, pp. 1445-1453.

[46] M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding*, Prentice-Hall Int., New Jersey, 1995.

[47] Y. Yamada, S. Tazaki, and R.M. Gray, "Asymptotic performance of block quantizers with difference distortion measures", *IEEE Trans. Inform. Theory*, Vol. IT-26, January 1980, pp. 6-14.

[48] Z.M. Yusof and T.R. Fisher, "An entropy-coded lattice vector quantizer for transform and subband image coding", *IEEE Trans. Image Processing*, Vol. 5, February 1996, pp. 289-298.

[49] J.W. Woods and S.D. O'Neil, "Subband coding of images", *IEEE Trans. ASSP*, Vol. ASSP-34, October 1986, pp. 1278-1288.

[50] S.W. Wu and A. Gersho, "Additive vector decoding of transform coded images", *IEEE Trans. Image Processing*, Vol. 7, June 1998, pp. 794-803.

[51] P.L. Zador, "Asymptotic quantization error of continuous signals and the quantization dimension", *IEEE Trans. Inform. Theory*, Vol. IT-28, March 1982, pp. 139-149.

[52] K. Zeger, A. Bist, and T. Linder, "Universal source coding with codebook transmission", *IEEE Trans. Commun.*, Vol. 42, February/March/April 1994, pp. 336-346.

[53] K. Zeger, J. Vaisey, and A. Gersho, "Globally optimal vector quantization design by stochastic relaxation", *IEEE Trans. Signal Processing*, Vol. 40, February 1992, pp. 310-322.

[54] X. Zhuang, Y. Huang, K. Palaniappan, and Y. Zhao, "Gaussian mixture density modeling decomposition, and applications", *IEEE Trans. Image Processing*, Vol. 5, September 1996, pp. 1293-1302.

Table 1: Example of bit allocation for image *Lena*. The processing parameters are $B = 8$ and $n_C = 4$. (a) Imposed bit rate $R_{AC} = 0.1$ bits/pixel. Real bit rate (given by the sum of all allocated bits divided by $B^2 \times n_C$) = 0.105 bits/pixel. (b) $R_{AC} = 0.3$ bits/pixel. Real bit rate = 0.304 bits/pixel.

		Vector			
		$\mathbf{v}_1$	$\mathbf{v}_2$	$\mathbf{v}_3$	$\mathbf{v}_4$
Energy class	1	1	0	0	0
	2	1	0	0	0
	3	4	0	0	0
	4	7	6	5	3

(a)

		Vector							
		$\mathbf{v}_1$	$\mathbf{v}_2$	$\mathbf{v}_3$	$\mathbf{v}_4$	$\mathbf{v}_5$	$\mathbf{v}_6$	$\mathbf{v}_7$	$\mathbf{v}_8$
Energy class	1	1	0	0	0	0	0	0	0
	2	3	0	0	0	0	0	0	0
	3	6	5	4	2	2	0	0	0
	4	10	11	10	8	6	5	3	2

(b)

Table 2: The a priori probabilities, means, and square root of the variances (standard deviations) corresponding to the first two AC coefficients of the highest energy class of image *Lena*. (a) Initialization values for the parameters of AC_1. (b) Estimates after 100 EM iterations. (c) Initialization values for the parameters of AC_2. (d) Estimates after 100 EM iterations.

		P_{1m}	μ_{1m}	σ_{1m}
m	1	0.25	-192.121	68.912
	2	0.25	-64.019	33.041
	3	0.25	66.649	37.662
	4	0.25	214.756	91.401

(a)

		P_{1m}	μ_{1m}	σ_{1m}
m	1	0.304	-71.649	146.186
	2	0.218	-112.055	60.971
	3	0.133	123.358	31.939
	4	0.343	104.668	163.452

(b)

		P_{2m}	μ_{2m}	σ_{2m}
m	1	0.25	-118.574	67.331
	2	0.25	-24.409	15.203
	3	0.25	21.626	15.303
	4	0.25	130.079	68.912

(c)

		P_{2m}	μ_{2m}	σ_{2m}
m	1	0.196	14.206	164.537
	2	0.145	-22.670	33.696
	3	0.088	5.819	14.701
	4	0.569	3.116	90.617

(d)

Table 3: Differences between $RMSE$s generated by the codebooks derived from training sets and those derived from actual data.

Image name	Vector	$RMSE$ diff (dB)	Image name	Vector	$RMSE$ diff (dB)
Aero	$\mathbf{v}_1$	1.08	Face	$\mathbf{v}_1$	-0.78
	$\mathbf{v}_2$	1.11		$\mathbf{v}_2$	1.30
Baboon	$\mathbf{v}_1$	0.46	Girl	$\mathbf{v}_1$	0.71
	$\mathbf{v}_2$	1.03		$\mathbf{v}_2$	1.06
Couple	$\mathbf{v}_1$	0.46	Lena	$\mathbf{v}_1$	0.22
	$\mathbf{v}_2$	1.12		$\mathbf{v}_2$	0.65

Table 4: Coding performance for TVQ-TSS at a bit rate of 0.28 bits/pixel.

Image name	$PSNR$ (dB)	Image name	$PSNR$ (dB)
Al	32.87	Goldhill	30.08
Aero	29.65	Jet	30.85
Baboon	23.13	Lena	32.51
Bank	28.02	Loco	25.75
Barbara	26.59	London	32.25
Boat	30.11	Oleh	32.93
Couple	38.88	Pyramid	31.98
Einstein	34.14	Regan	32.05
Face	31.54	Wedding	30.57
Girl	33.94	Zelda	35.95

Table 5: TVQ-TSS numerical results corresponding to different bit rates.

Bit rate (bits/pixel)		0.25	0.30	0.35	0.40	0.45	0.50
$PSNR$ (dB)	Goldhill	29.49	30.27	30.90	31.47	31.86	32.02
	Lena	31.84	32.80	33.59	34.32	34.53	34.92

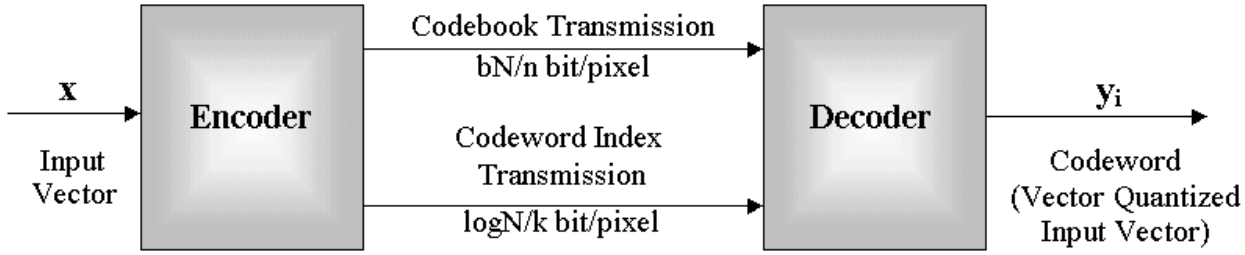


Figure 1: Adaptive VQ with codebook transmission.

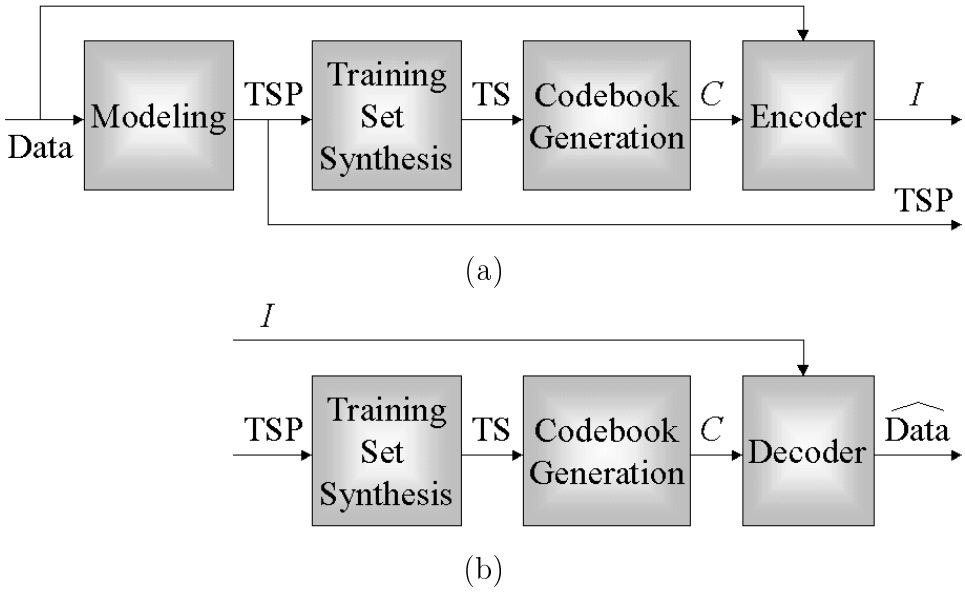


Figure 2: Vector quantization with training set synthesis. (a) Encoding side. (b) Decoding side.

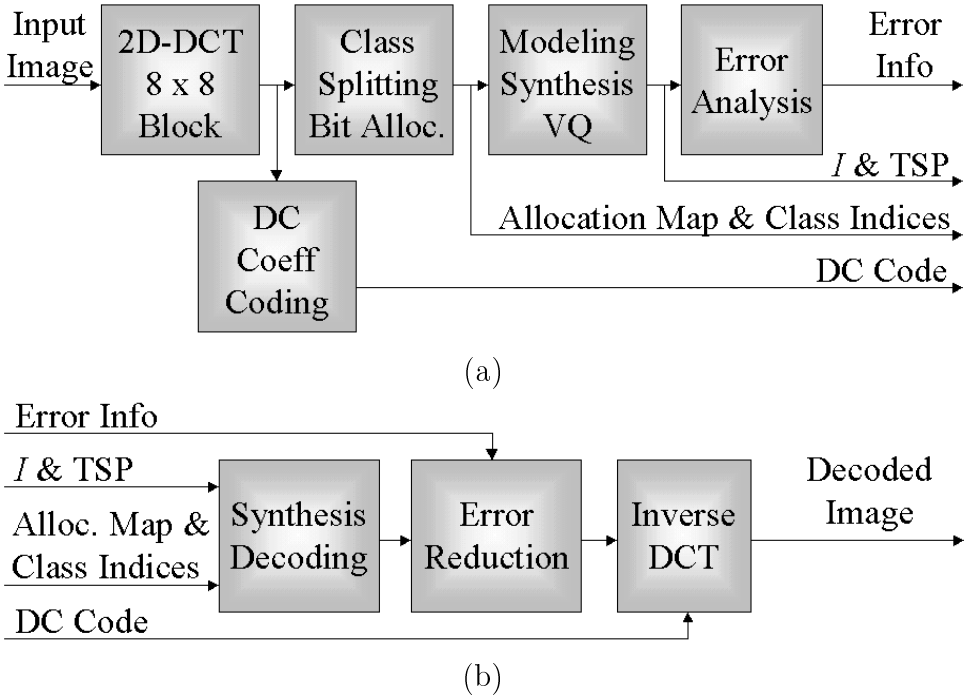
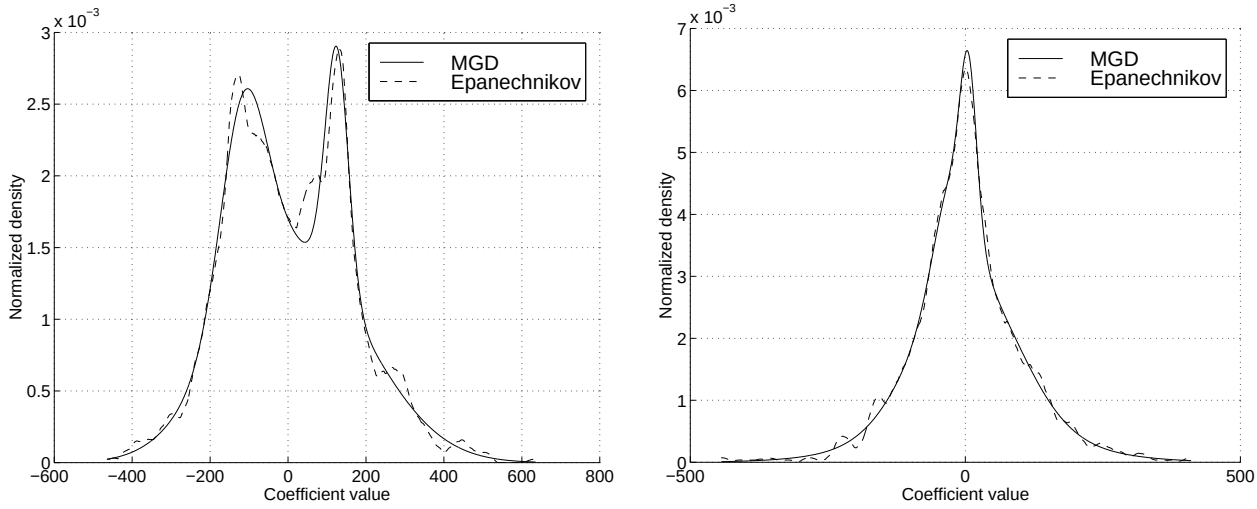


Figure 3: Block diagram of TVQ-TSS compression. (a) Encoder. (b) Decoder.

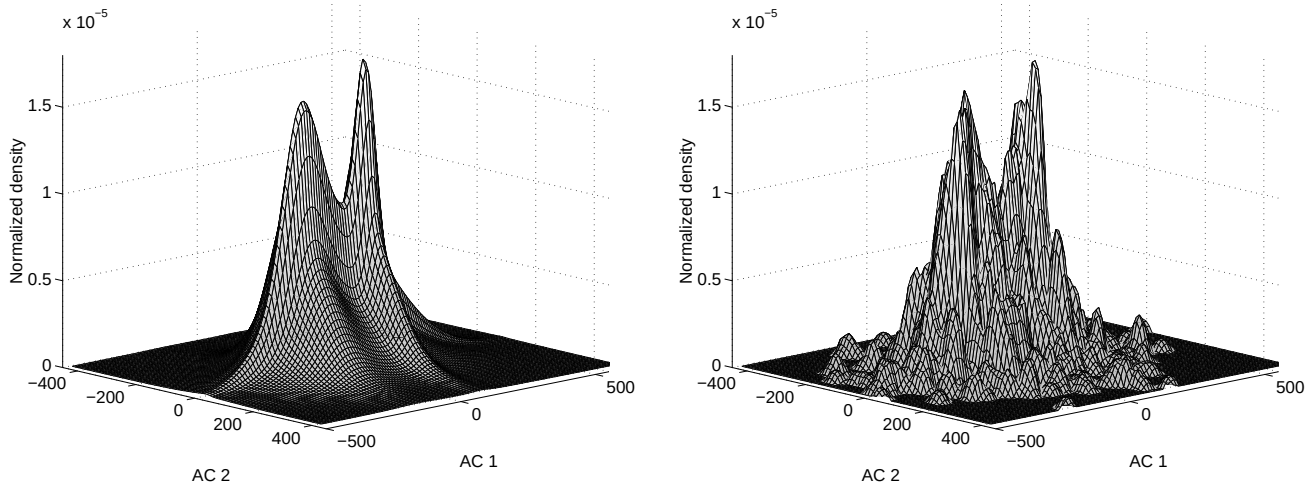
DC	1	2	3	5	5	8	8
1	2	3	4	5	8	8	12
2	3	4	5	7	9	11	12
3	4	6	7	9	11	12	14
4	6	7	9	11	12	14	15
6	7	9	11	13	14	15	16
6	10	10	13	14	15	16	17
10	10	13	13	15	16	17	17

Figure 4: Decomposition of the 8×8 DCT block into one scalar representing the DC coefficient and vectors $\{\mathbf{v}_i\}_{i=1,\ldots,17}$ representing the AC coefficients scanned in zigzag order.



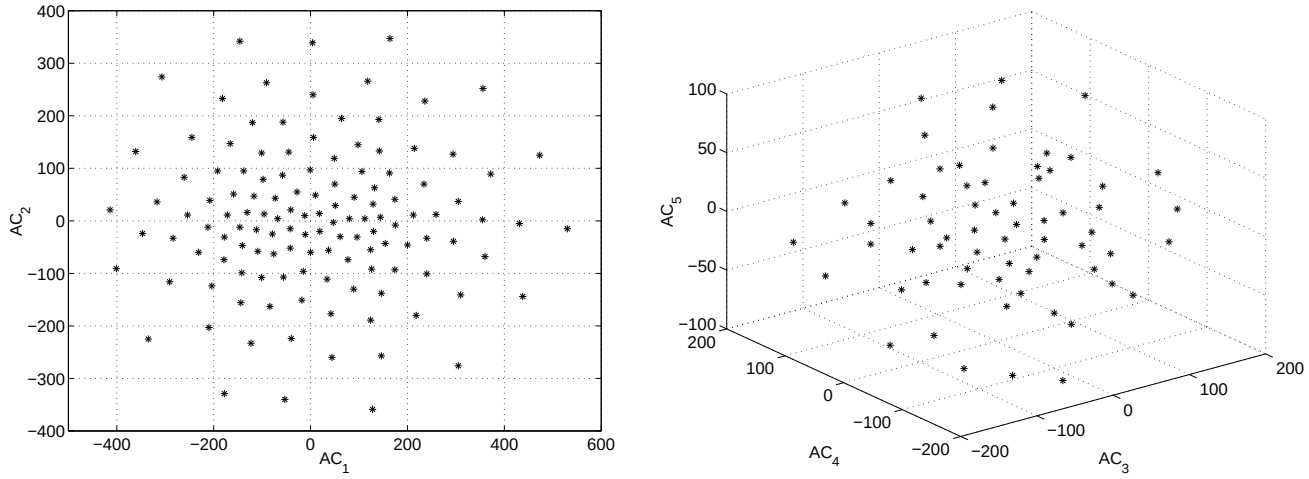
(a) (b)

Figure 5: The PDFs corresponding to estimated parameters given in Table 2 and the PDFs derived through nonparametric analysis with optimal kernel of window width $h = 30$. (a) Coefficient AC_1. (b) Coefficient AC_2.



(a) (b)

Figure 6: The joint PDF of the vector $\mathbf{v}_1 = (AC_1, AC_2)^\top$ of the highest energy class derived from image *Lena*. (a) MGD model. (b) 2-dimensional Epanechnikov estimate.



(a) (b)

Figure 7: Codebooks derived from synthesized training sets (highest energy class). (a) Coefficients $(AC_1, AC_2)^\top$, 128 codewords. (b) Coefficients $(AC_3, AC_4, AC_5)^\top$, 64 codewords.

(a) (b)

(c) (d)

Figure 8: TVQ-TSS results: image *Goldhill*. (a) Original. (b) 0.25 bits/pixel, 29.49 dB. (c) 0.35 bits/pixel, 30.90 dB. (d) 0.5 bits/pixel, 32.02 dB.

(a) (b)

(c) (d)

Figure 9: TVQ-TSS results: image *Lena*. (a) Original. (b) 0.25 bits/pixel, 31.84 dB. (c) 0.35 bits/pixel, 33.59 dB. (d) 0.5 bits/pixel, 34.92 dB.

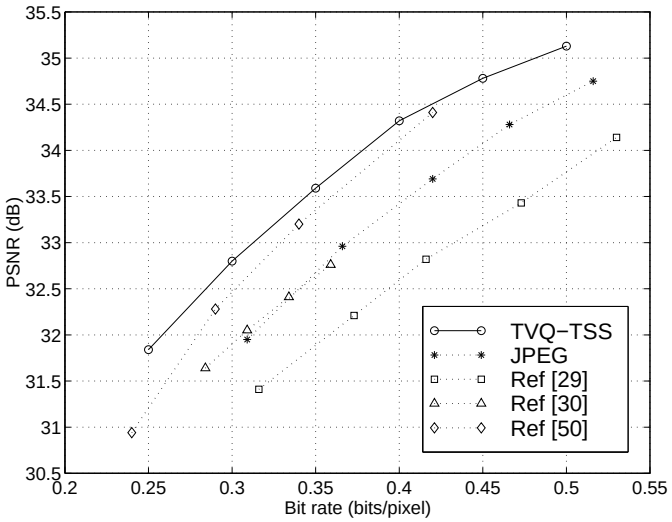


Figure 10: Performance of various coding systems for image *Lena*.