Technical Report # TR-86-37

MARYLAND

TECHNICAL
RESEARCH
REPORT

Optimal Block Cosine Transform Image Coding for Noisy Channels

by

V. Vaishampayan

and

N. Farvardin

SYSTEMS RESEARCH CENTER

UNIVERSITY OF MARYLAND

COLLEGE PARK, MARYLAND 20742

Optimal Block Cosine Transform Image Coding for Noisy Channels *

V. Vaishampayan and N. Farvardin
Electrical Engineering Department
and
Systems Research Center
University of Maryland
College Park, Maryland 20742

August 25, 1986

Abstract

The two-dimensional block transform coding scheme based on the discrete cosine transform has been studied extensively for image coding applications. While this scheme has proven to be efficient in the absence of channel errors, its performance degrades rapidly over noisy channels. In this paper we present a method for the joint source-channel coding optimization of a scheme based on the 2-D block cosine transform when the output of the encoder is to be transmitted via a memoryless binary symmetric channel. Our approach involves an iterative algorithm for the design of the quantizers (in the presence of channel errors) used for encoding the transform coefficients. This algorithm produces a set of locally optimum (in the mean squared-error sense) *quantizers* and the corresponding *binary code assignment* for the assumed transform coefficient statistics. To determine the *optimum bit assignment* among the transform coefficients, we have used an algorithm based on the steepest descent method, which under certain convexity conditions on the performance of the channel-optimized quantizers, yields the *optimal* bit allocation. Comprehensive simulation results for the performance of this locally optimum system over noisy channels have been obtained and appropriate comparisons against a reference system designed for no channel errors have been rendered. It is shown that substantial performance improvements can be obtained by using this scheme. Furthermore, theoretically predicted results for an assumed 2-D image model are provided.

*This work was supported in part by a grant from NASA, Langley Research Center under grant NAG-1-582, in part by Martin Marietta Laboratories and in part by the National Science Foundation under grant NSFD CDR-85-00108.

I Introduction

The two-dimensional (2-D) block cosine transform coding scheme has been well studied in image coding situations from a rate distortion-theoretic perspective. Apart from providing good performance it is image independent and can be efficiently implemented using fast algorithms [5].

It is known that the performance of the above system degrades rapidly in the presence of channel errors. In particular work done in [1] has shown that a careful selection of the parameters of the block cosine transform, as well as the channel coding and modulation methods, are essential for maintaining a good performance when data is to be transmitted over a noisy channel.

In this paper we optimize the mean squared-error of a block cosine transform coding scheme over a noisy channel, with a constraint on the average transmission rate, when the transform coefficients are encoded using zero-memory encoders. We show that the optimization problem can be reduced to:

1. optimal allocation of bits for encoding the transform coefficients in such a way that both the source and the channel characteristics are taken into consideration.

2. designing optimal zero-memory encoder/decoder pairs for the transmission of each of the transform coefficients over a noisy channel.

It has been shown [2] that even when a noisy channel is present, the optimal structure of a zero-memory encoder for a stationary one-dimensional random process, under the mean squared-error criterion consists of a quantizer followed by a codeword assignment to each of the quantization intervals. For a given bit allocation vector, we apply the results of a new algorithm [4] that designs an optimal quantizer, its associated codeword assignment and decoder (referred to as the encoder/decoder pair) so as to minimize the total average reconstruction error across a noisy channel. We then optimize the allocation of bits among the transform coefficients by formulating it as an integer programming problem and then use an incremental bit allocation procedure based on the method of steepest descent [3]. Under certain convexity conditions, it is possible to show that this algorithm yields the optimal bit allocation, in a number of steps equal to the total number of bits to be allocated per block.

We compare the performance of the optimal system, as applied to images, against a reference system, where each of the transform coefficients are encoded using Lloyd-Max quantizers followed by the natural binary code assignment. The performance gains of the optimal system over the reference system are extremely encouraging. Simulation studies have revealed improvements in the signal-to-noise ratio (SNR) of around 8 dB, at a transmission rate of 1 bit/pixel, and for a binary symmetric channel with a crossover probability of 0.05. The results are just as encouraging from a subjective viewpoint. The quality of the reconstructed image for the optimal system is seen to be far superior to the reference system, especially as

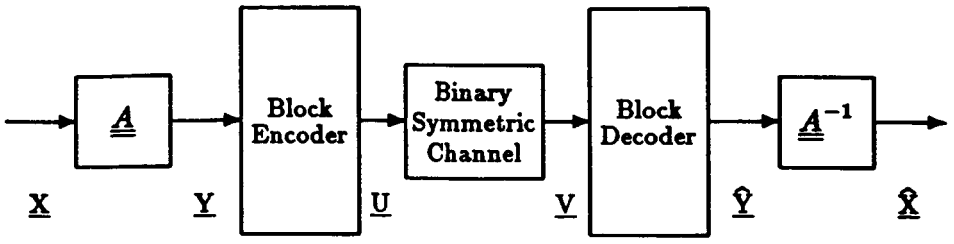


Figure 1: Block Transform Coding Scheme.

the channel gets noisier. The important point is that these performance improvements are not obtained at the expense of transmission bandwidth. Furthermore, we have obtained theoretically predicted performance results based on a 2-D separable Gauss-Markov model for the source.

The rest of this paper is organized as follows: In Section II we develop notation and describe the system under consideration. In Section III we present an analysis of the system and describe the algorithm used to design the encoder/decoder pairs optimally and the algorithm that does the optimal bit allocation. Section IV contains a description of the system as implemented, theoretically predicted performance results and simulation results on two test images. Finally a summary and conclusions is provided in Section V.

II Preliminaries

In what follows we assume that the source is an L-dimensional vector source represented by a zero mean, stationary, discrete-parameter stochastic process $\{\underline{X}_n\}$ whose moment matrix is denoted by $\underline{\underline{\Phi}}_{XX}$. In a typical block transform coding scheme the source output vector [1] $\underline{X} = (X_0, X_1, \ldots, X_{L-1})^T$ is transformed to a vector $\underline{Y} = (Y_0, Y_1, \ldots, Y_{L-1})^T$ by a nonsingular $(L \times L)$ transformation matrix $\underline{\underline{A}}$ according to

$$\underline{Y} = \underline{\underline{A}}\,\underline{X}. \tag{1}$$

Each of the L components of $\underline{Y}$ (also referred to as the *transform coefficients*) is separately quantized and encoded using fixed-length binary codewords. The resulting vector of binary codewords $\underline{U} = (U_0, U_1, \ldots, U_{L-1})^T$ is transmitted across the communication channel. We call the set of L encoders the "*block encoder.*" Here we assume that the communication channel is modeled as a memoryless binary symmetric channel (BSC) with crossover probability ϵ.

The received vector $\underline{V} = (V_0, V_1, \ldots, V_{L-1})^T$, each component of which is a binary sequence, is decoded component-wise by a decoder, and the resulting vector $\hat{\underline{Y}}$ is then transformed by $\underline{\underline{A}}^{-1}$ to yield a representation vector $\hat{\underline{X}}$, of the source vector $\underline{X}$, in the receiver, i.e.,

$$\hat{\underline{X}} = \underline{\underline{A}}^{-1}\hat{\underline{Y}}. \tag{2}$$

[1] The parameter index has been dropped since the source is stationary.

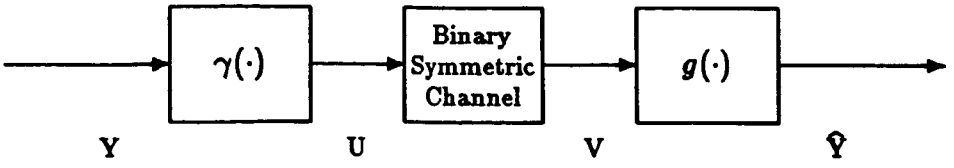


Figure 2: A Typical Zero-Memory Encoder/Decoder Pair.

The block diagram of this coding scheme is illustrated in Fig. 1. The i^{th} component, U_i, of $\underline{U}$ is an r_i-bit codeword and the vector $\underline{r} = (r_0, r_1, \ldots, r_{L-1})^T$ is called the *bit allocation* vector. For a given bit allocation vector $\underline{r}$, the average number of bits used to represent a source symbol is given by r_{av} described by

$$r_{av} \triangleq \frac{1}{L} \sum_{i=0}^{L-1} r_i. \tag{3}$$

The squared-error distortion measure is used exclusively in what follows. The average per-symbol distortion D is described by

$$D = \frac{1}{L} E\{tr[(\underline{X} - \widehat{\underline{X}})(\underline{X} - \widehat{\underline{X}})^T]\}. \tag{4}$$

Each of the L encoder/decoder pairs has a similar structure and so we shall describe one of them as illustrated in Fig. 2. As shown, Y represents a generic transform coefficient. In our scheme, the encoder $\gamma(\cdot)$ is essentially a quantizer followed by a *code assignment*, which maps every quantization level to a specific r-bit codeword. The encoder mapping is specified by a vector of quantization thresholds $\underline{T} = (T_0, T_1, \ldots, T_N)^T$, and a set of binary codewords of length r, described by $\mathcal{A}_u = \{u_1, u_2, \ldots, u_N\}$ where u_i is defined as the codeword to which all source outputs in the interval $(T_{i-1}, T_i]$ are mapped, i. e.,

$$\gamma(y) = u_i, \quad \text{if } y \in (T_{i-1}, T_i],\ i = 1, 2, \ldots, N, \tag{5}$$

and where N is the number of quantization levels. Let $\mathcal{A}_v = \{v_1, v_2, \ldots, v_M\}$ denote the set of all possible received r-bit sequences, and let $\underline{R} = (R_1, R_2, \ldots, R_M)^T$ be the set of reconstruction levels. The decoder mapping $g(\cdot)$ is defined in terms of the received sequences and the reconstruction levels by

$$g(v) = R_i, \quad \text{if } v = v_i,\ i = 1, 2, \ldots, M. \tag{6}$$

Note that the number of quantization levels satisfies $N \leq 2^r$, whereas the number of possible received sequences is given by $M = 2^r$.

III Problem Statement and Analysis

For the system described in the previous section, we wish to minimize the mean-squared error

$$D = \frac{1}{L}\ E\{tr\left[(\underline{X} - \underline{\widehat{X}})(\underline{X} - \underline{\widehat{X}})^T\right]\}, \tag{7}$$

subject to a constraint on the average number of bits per-symbol required to transmit the source, i. e.,

$$r_{av} = \frac{1}{L}\sum_{i=0}^{L-1} r_i \leq \tilde{r}, \tag{8}$$

and subject to a constraint on the minimum and the maximum number of bits that can be allocated to a single coefficient, i. e.,

$$0 \leq r_i \leq r_{max},\ \ i = 0, 1, \ldots, L-1, \tag{9}$$

where the r_i's and $L\tilde{r}$ are assumed to be integer-valued and r_{max} is a prescribed integer. If $\underline{\underline{A}}$ is chosen to be an orthonormal matrix, the distortion may be simplified to

$$D = \frac{1}{L}\ E\{tr[(\underline{Y} - \underline{\hat{Y}})(\underline{Y} - \underline{\hat{Y}})^T]\}, \tag{10}$$

which can then be expressed as a sum of the component distortions by

$$D = \frac{1}{L}\sum_{i=0}^{L-1} E(Y_i - \hat{Y}_i)^2. \tag{11}$$

The i^{th} component distortion, $E(Y_i - \hat{Y}_i)^2$, is a function of the number of bits r_i, allocated for transmitting the i^{th} component and will be denoted hereafter by $d_i(r_i)$. For a *given* bit allocation vector $\underline{r}$, in order to minimize D, it suffices to minimize each of the component distortions. Let $d_i^*(r_i)$ denote the minimum value of the i^{th} component distortion when r_i bits are used to to transmit the i^{th} transform coefficient. Then the minimum average distortion for a given bit allocation $\underline{r}$, denoted by $D^*(\underline{r})$ is given by

$$D^*(\underline{r}) = \frac{1}{L}\sum_{i=0}^{L-1} d_i^*(r_i). \tag{12}$$

Let us assume that the values of $d_i^*(r)$, $i = 0, 1, \ldots, L-1$, and $r = 0, 1, \ldots, r_{max}$, are known. In order to solve the optimization problem, it remains to determine the optimal bit allocation, say $\underline{r}^*$, that satisfies the constraints in (8) and(9) and such that $D^*(\underline{r}^*) \leq D^*(\underline{r})$ for all $\underline{r}$ which satisfy the same constraints.

In the rest of this section we will first briefly describe an algorithm that optimizes the performance of the zero-memory encoder/decoder pair across a noisy channel, thus providing us with the (locally) optimal distortion figures for the component distortions. We will then describe an optimal bit allocation algorithm, that, under some mild conditions, yields the optimal $\underline{r}^*$ in $L\tilde{r}$ steps. Together, these algorithms are then used to solve the constrained optimization problem as formulated at the beginning of this section.

A Optimal Zero-Memory Encoder/Decoder Design

In this subsection we describe an algorithm to minimize the component distortions $d_i(r_i)$ for a given r_i. Since the optimization procedure is the same for each of the L transform coefficients, we shall consider a generic term and denote it by $d(r)$.

We refer to an earlier paper [4] in which we described an iterative algorithm that converges to a locally optimal encoder/decoder design for the system shown in Fig. 2. This procedure results in the optimal value of the component distortion $d(r)$ for a given value of r.

The encoder essentially consists of an N-level quantizer followed by a code assignment map. As shown in [4], the average squared-error distortion is a function of the following parameters:

1. The set of threshold levels $\underline{T} = (T_0, T_1, \ldots, T_N)^T$;
2. The value of N, the number of quantization intervals;
3. The code assignment, i.e., the codewords to which the quantizer intervals should be mapped, and,
4. The set of reconstruction levels in the decoder.

The distortion[2] $d(r)$ can be expressed as

$$d(r) = \int_{-\infty}^{\infty} p_Y(y) E\{(y - \hat{Y})^2 | Y = y\} dy, \tag{13}$$

in which $p_Y(\cdot)$ denotes the probability density function (p.d.f.) of the source to be encoded (in our application, a generic transform coefficient), and $E\{\cdot\}$ is the expectation operator. The necessary conditions for optimality are developed by deriving:

a. The necessary and sufficient conditions for optimizing the decoder mapping g for a fixed encoder mapping γ.

b. The necessary and sufficient conditions for optimizing the encoder mapping γ for a fixed decoder mapping g.

The system that satisfies the above two conditions simultaneously is a locally optimal system. The optimal decoder mapping for a fixed encoder mapping follows directly from a well-known result in estimation theory and is given by

$$R_i = E\{Y | V = v_i\}, \ 1, 2, \ldots, M. \tag{14}$$

On the other hand, for a fixed decoder mapping, it is straightforward to show that the optimal encoder mapping is such that it maps a value of $Y = y$ to a codeword u_i, if and only if

$$E\{(y - \hat{Y})^2 | U = u_i\} \leq E\{(y - \hat{Y})^2 | U = u_j\}, \ \forall j \neq i. \tag{15}$$

[2]The distortion also depends on γ and g but, for the sake of brevity, this is not explicitly reflected in our notation.

For a fixed decoder $g(\cdot)$, upon defining $A_i(g)$ as the set of values of y that should be mapped to the codeword u_i in preference to any other codeword, and $A_{ij}(g)$ as the set of values of y that should be mapped to u_i in preference to u_j, we may express $A_i(g)$ as,

$$A_i(g) = \bigcap_{j=1, j\neq i}^{N} A_{ij}(g). \tag{16}$$

Analysis of (13) shows that $A_{ij}(g)$ may be expressed as

$$\begin{aligned} A_{ij}(g) &= \{y : 2y\left[E\{\hat{Y}|U=u_j\} - E\{\hat{Y}|U=u_i\}\right] \\ &\quad \leq E\{\hat{Y}^2|U=u_j\} - E\{\hat{Y}^2|U=u_i\}\}. \end{aligned} \tag{17}$$

We now define

$$\alpha_{ij} \triangleq E\{\hat{Y}^2|U=u_j\} - E\{\hat{Y}^2|U=u_i\}, \tag{18}$$

$$\beta_{ij} \triangleq E\{\hat{Y}|U=u_j\} - E\{\hat{Y}|U=u_i\}, \tag{19}$$

and

$$t_{ij} \triangleq \alpha_{ij}/2\beta_{ij}, \quad \beta_{ij} \neq 0, \tag{20}$$

from which it follows that

$$A_{ij}(g) = \begin{cases} (-\infty, t_{ij}], & \text{if } \beta_{ij} > 0, \\ [t_{ij}, \infty), & \text{if } \beta_{ij} < 0, \\ (-\infty, \infty), & \text{if } \beta_{ij} = 0, \alpha_{ij} \geq 0, \\ \phi, & \text{if } \beta_{ij} = 0, \alpha_{ij} < 0. \end{cases} \tag{21}$$

Therefore, $A_{ij}(g)$ is an interval, and hence, so is $A_i(g)$. We may now define the upper and lower endpoints of $A_i(g)$ by

$$t_i^u = \min_{j:\beta_{ij}>0} \{t_{ij}\}, \tag{22}$$

and

$$t_i^l = \max_{j:\beta_{ij}<0} \{t_{ij}\}, \tag{23}$$

respectively, and use (22) and (23) to characterize $A_i(g)$ by

$$A_i(g) = \begin{cases} \phi, & \text{if } \beta_{ij} = 0 \text{ and } \alpha_{ij} < 0 \text{ for some } j, \\ \mathcal{R}, & \text{if } \beta_{ij} = 0,\ \alpha_{ij} \geq 0,\ \forall j, \\ \left[t_i^l, t_i^u\right], & \text{otherwise,} \end{cases} \tag{24}$$

provided that $t_i^l \leq t_i^u$ for all $i = 1, 2, \ldots, N$. In order to resolve the ambiguity on the endpoints of the quantization intervals, we define the set $\hat{A}_i(g)$ to be identical to $A_i(g)$, except when $A_i(g) = [t_i^l, t_i^u]$, in which case $\hat{A}_i(g) \triangleq (t_i^l, t_i^u]$. Then, the optimum encoder mapping γ for a *fixed* decoder mapping g is given by

$$\gamma(y) = u_i, \quad y \in \hat{A}_i(g), \quad i = 1, 2, \ldots, M. \tag{25}$$

It is important to note that when the channel is noisy, we do encounter situations in which $t_i^l > t_i^u$, even though $\beta_{i,j} \neq 0$ for all $j \neq i$. The correct interpretation of the situation when $t_i^l > t_i^u$ for some i, is that no value of the source output is to be encoded by the i^{th} codeword and hence this codeword is not to be transmitted. Equation (24) thus provides us with a method of *identifying* a subset of codewords which should be used to encode the output of the zero-memory encoder/decoder pair. If the number of codewords in this subset is N, $N \leq M$ then (24), in effect, tells us that an N-level quantizer is optimum for minimizing the mean squared-error. Note that if $t_i^l > t_i^u$ for some i, then (24) cannot be used to obtain the optimal encoder. In [4], we describe a method for obtaining the optimal encoder even when $t_i^l > t_i^u$ for some i. By successive application of this method and (14), an iterative design algorithm is developed that converges to a locally optimal encoder/decoder pair. In order to be brief we shall not describe this algorithm here but shall refer the interested reader to [4].

It is interesting to mention, however, that for a Gaussian source and a BSC with a crossover probability of 0.01 and a rate of 8 bits/sample, our results indicate a (locally) optimal quantizer that possesses only 29 of the possible 256 levels.

We used this algorithm to generate the (locally) optimal, distortion vs. rate performance, $d^*(r)$, needed for the bit allocation algorithm described next. We mention at this point that the distortion vs. rate performance of the optimal encoder/decoder pair, was observed to be convex for rates up to 8 bits/sample, and for a Gaussian source density. This observation is important since convexity is required in the development of the optimal bit allocation algorithm.

B Optimal Bit Allocation Algorithm

In the absence of an analytical expression for the distortion vs. rate performance of the optimal encoder/decoder pairs described above, we resort to an integer programming algorithm to determine the optimal bit allocation vector $\underline{r}^*$. The algorithm described in the previous section yields the following distortion values:

$$d_i^*(r_i), \quad i = 0, 1, \ldots, L-1; \quad r_i = 0, 1, \ldots, \tilde{r}L. \tag{26}$$

We shall assume that the functions $d_i^*(\cdot)$ are convex and decreasing for all $i = 0, 1, \ldots, L-1$. Here, by convexity, we mean that $d_i(r-1) - d_i(r) \geq d_i(r) - d_i(r+1)$, for all integers $r \geq 1$. Let us consider the problem of minimizing

$$D^*(\underline{r}) = \frac{1}{L} \sum_{i=0}^{L-1} d_i^*(r_i), \tag{27}$$

subject to

$$\frac{1}{L} \sum_{i=0}^{L-1} r_i = \tilde{r}, \tag{28}$$

and

$$r_i \geq 0, \quad i = 0, 1, \ldots, L-1. \tag{29}$$

After describing a steepest descent algorithm that yields the optimal bit allocation $\underline{r}^*$ for the above problem, we will show how the same algorithm may be used to determine the optimal bit allocation, when an additional constraint is placed on the *maximum* number of bits that can be allocated to each component, i.e., when

$$r_i \leq r_{max}, \quad i = 0, 1, \ldots, L-1. \tag{30}$$

The algorithm proceeds as follows [3].

1. Set k=0; Set $r_i = 0, \quad i = 0, 1, \ldots, L-1$.

2. Set $k = k + 1$; compute the index i_k which satisfies

$$d_{i_k}^*(r_{i_k}) - d_{i_k}^*(r_{i_k} + 1) = \max_{0 \leq i \leq L-1} \{d_i^*(r_i) - d_i^*(r_i + 1)\}. \tag{31}$$

3. Set $r_{i_k} = r_{i_k} + 1$. If $k < \tilde{r}L$, go to step 2; else stop.

We state the following theorem without proof, details of which may be found in [3].

Theorem 1: If the distortion functions in (26) are convex and non-increasing, then the steepest descent algorithm as described above, yields the bit allocation vector $\underline{r}^*$ that minimizes (27) subject to the constraints in (28) and (29).

In order to use the steepest descent algorithm to solve the problem under the additional constraint on individual bit allocations, as stated in (30), we define a new set of distortion functions $\tilde{d}_i(r)$ by

$$\tilde{d}_i(r) \triangleq \begin{cases} d_i^*(r), & \text{if } 0 \leq r \leq r_{max}, \\ d_i^*(r_{max}), & \text{if } r > r_{max}, \end{cases} \quad i = 0, 1, \ldots, L-1. \tag{32}$$

The corresponding average distortion $\widetilde{D}(\underline{r})$ is given by,

$$\widetilde{D}(\underline{r}) = \frac{1}{L} \sum_{i=0}^{L-1} \tilde{d}_i(r_i). \tag{33}$$

The following theorem establishes that the minimization of $\widetilde{D}(\underline{r})$ subject to (28) and (29) is equivalent to the minimization of $D^*(\underline{r})$ subject to (28)–(30).

Theorem 2: If the functions described in (26) are convex and strictly decreasing, then the bit allocation vector $\underline{r}^*$, that minimizes $\widetilde{D}(\underline{r})$ subject to (28) and (29), minimizes $D^*(\underline{r})$ subject to (28), (29) and the additional constraint (30).

Proof: The set of feasible bit allocation vectors that satisfy constraints (28), (29) and (30) is contained in the set of feasible bit allocation vectors that satisfy constraints (28) and (29). The functions $\tilde{d}_i(r), \quad i = 0, 1, \ldots, L-1$, are convex and non-increasing since we assumed that the functions $d_i^*(r), \quad i = 0, 1, \ldots, L-1$, were convex and decreasing. Hence by applying the steepest descent algorithm, we may determine the bit allocation vector $\underline{r}^*$ that minimizes $\widetilde{D}(\underline{r})$ subject to (28), (29). Further, since $\tilde{r} \leq Lr_{max}$, the steepest descent algorithm will never allocate more

than r_{max} bits to any of the L components. To see this, assume, without any loss of generality, that at the k^{th} step of the bit allocation process, $\underline{r} = (r_{max}, r_1, \ldots, r_{L-1})^T$, where $r_i < r_{max}$, $i = 1, 2, \ldots, L-1$. At the $(k+1)^{st}$ step, an index i_{k+1} is selected for which (31) is true. Since $(\tilde{d}_0(r_{max}) - \tilde{d}_0(r_{max}+1)) = 0$, it follows that $i_{k+1} \neq 0$, which proves the theorem.

IV Numerical and Simulation Results

We have implemented an image coding system using the method outlined in the previous sections. The 2-D discrete cosine transform has been chosen as the source transformation and we have assumed that the image could be modeled by a stationary 2-D Gaussian random field. In order to compute the theoretically predicted performance, we have made the additional assumption, that the model has a 2-D separable, first-order, Gauss-Markov structure. These results serve as a useful benchmark of system performance. The 2-D Gauss-Markov random field is described according to

$$\begin{aligned} X(i,j) &= \rho_r X(i-1,j) + \rho_c X(i,j-1) - \rho_r \rho_c X(i-1,j-1) + W(i,j), \\ & \quad i,j = 0,1,\ldots,L-1, \end{aligned} \tag{34}$$

where ρ_r and ρ_c are the vertical (row) and horizontal (column) correlation coefficients, respectively, and $W(i,j)$ is a 2-D sequence of independent and identically distributed Gaussian random variables with zero-mean and variance σ_W^2. It is also assumed that values of $X(-1,j)$ are known for $j = -1, 0, 1, \ldots, L-1$ and the values of $X(i,-1)$ are known for $i = 0, 1, \ldots, L-1$. For a stationary process the source variance σ_X^2 and σ_W^2 must be related by

$$\sigma_W^2 = \sigma_X^2 (1-\rho_r^2)(1-\rho_c^2). \tag{35}$$

The image frame is blocked into blocks of size $L \times L$. These blocks are then operated on by the 2D-DCT defined by

$$\begin{aligned} Y(m,n) &= \frac{2}{L} C(m) C(n) \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} X(i,j). \\ & \quad \cos \frac{(2i+1)m\pi}{2L} \cos \frac{(2j+1)n\pi}{2L}, \\ & \quad m,n = 0,1,\ldots,L-1, \end{aligned} \tag{36}$$

where $C(0) = 1/\sqrt{2}$ and $C(m) = 1$ for $m = 1, 2, \ldots, L-1$. The inverse transform

(2D-IDCT) is defined by

$$X(i,j) = \frac{2}{L}\sum_{m=0}^{L-1}\sum_{n=0}^{L-1} C(m)C(n)Y(m,n). \cos\frac{(2i+1)m\pi}{2L}\cos\frac{(2j+1)n\pi}{2L},$$

$$i,j = 0,1,\ldots,L-1. \tag{37}$$

The 2D-DCT is a separable orthonormal transformation and the analysis of Section II extends to two dimensions along exactly the same lines. In particular, the source is now represented by a matrix $\underline{\underline{X}}$ and by using the orthonormality of the transformation, we can express the average squared-error distortion as

$$D^*(\underline{\underline{r}}) = \frac{1}{L^2}\sum_{m=0}^{L-1}\sum_{n=0}^{L-1} d^*_{mn}(r_{mn}), \tag{38}$$

where $\underline{\underline{r}}$ is now the bit allocation *matrix,* and $d^*_{mn}(r_{mn})$ is the optimal distortion incurred in transmitting the $(m,n)^{th}$ transform coefficient across the channel using r_{mn} bits. The optimal bit allocation algorithm now requires $L^2\tilde{r}$ steps to complete the allocation, where $L^2\tilde{r}$ is again the total number of bits that are to be allocated to the image block of size $L \times L$.

Since the source is assumed to be Gaussian, each of the transform coefficients are also Gaussian, which implies that the component distortions may all be expressed in terms of a set of (at most) r_{max} variance-normalized distortion functions as

$$d^*_{mn}(r_{mn}) = \sigma^2_{mn} d^*_{nor}(r_{mn}), \quad m,n = 0,1,\ldots,L-1. \tag{39}$$

Here, $d^*_{nor}(r)$ is the minimum distortion achievable by a zero-memory encoder/decoder pair designed for a *unit-variance Gaussian* source as a function of the transmission rate, r. Also σ^2_{mn} is the variance of the $(m,n)^{th}$ transform coefficient.

For the 2D-DCT and the 2-D Gauss-Markov model the variance σ^2_{mn} may be expressed [1] as the product of $\sigma^2_c(m)$ and $\sigma^2_r(n)$ which are defined by

$$\sigma^2_c(m) \triangleq \frac{2\sigma_X}{L}C^2(m)\sum_{i=0}^{L-1}\sum_{j=0}^{L-1}\rho_c^{|i-j|} \cos\frac{(2i+1)m\pi}{2L}\cos\frac{(2j+1)m\pi}{2L},$$

$$m = 0,1,\ldots,L-1, \tag{40}$$

and

$$\sigma_r^2(n) \triangleq \frac{2\sigma_X}{L} C^2(n) \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \rho_r^{|i-j|} \cos\frac{(2i+1)n\pi}{2L} \cos\frac{(2j+1)n\pi}{2L},$$

$$n = 0, 1, \ldots, L-1, \tag{41}$$

respectively.

In order to compute the theoretically predicted performance for the assumed 2-D Gauss-Markov image model, we have estimated the values of ρ_r, ρ_c and σ_X^2 from the "GIRL" and the "MOON" images, each of size 256×256. These values are summarized in Table 1. The variances of the transform coefficients (for an $L \times L$ transform) have then been computed using (40) and(41), following which, the optimal bit allocation vector and the corresponding performance results have been obtained using the steepest descent algorithm with r_{max} set to 8 bits. Results based on the 2-D Gauss-Markov model have also been computed for a reference system in which the transform coefficients are quantized by Lloyd-Max quantizers designed for a Gaussian source and encoded using the natural binary code. The bit allocation matrix used here is the one that is optimal for the case when no channel errors are present ($r_{max} = 8$). These results, referred to as the *numerical results* are presented in Tables 2 and 3 for the "MOON" image at rates of 1 bit/pixel and 0.5 bit/pixel respectively, and in Tables 6 and 7 for the "GIRL" image at rates of 1 bit/pixel and 0.5 bit/pixel respectively. Each of these tables contains results for block sizes of 32×32, 16×16 and 8×8, at values of $\epsilon = 0.0$, 0.005, 0.01 and 0.05.

Monte-Carlo simulation results for real-world images have also been obtained, without any modeling assumptions. Here each image of size 256×256 has been blocked off into blocks of size $L \times L$. Each block has then been transformed, and the variances of the transform coefficients have been estimated from the transformed image data. We assume the mean and variance data of the transform coefficients is available at the receiver, and that when zero bits are allocated to any transform coefficient, it is decoded to its mean value in the receiver. Simulations have also been run for the reference system which was described earlier. Examples of reconstructed image quality obtained through simulations are illustrated in Figures 3–8. These figures illustrate the performance of the optimal systems for values of $\epsilon = 0.005$, $\epsilon = 0.01$ and $\epsilon = 0.05$, for a block size of 32×32, at an average rate of 1 bit/pixel. To consider the effects of mismatch on the system performance, we have also included the performance of the optimal systems, designed for the values of ϵ mentioned above, when a noiseless channel is present. The performance results obtained through simulations are summarized in Tables 4 and 5 for the "MOON" image and in Tables 8 and 9 for the "GIRL" image. These results are presented for block sizes of 32×32, 16×16 and 8×8, at values of $\epsilon = 0.0$, 0.005, 0.01 and 0.05.

It is obvious from the numerical and simulation results that the optimal system results in improvements in signal-to-noise ratio over the reference system. These

improvements are particularly significant for very noisy channels. To be specific, the numerical results for the "GIRL" image indicate performance improvements of approximately 8 dB at a rate of 1 bit/pixel, for a block size of 8×8 and a channel crossover probability of 0.005. The performance improvements are seen to increase as the block size increases and as the crossover probability increases. By comparing corresponding values for the "GIRL" and the "MOON" image it can also be seen that the improvements are larger for the "GIRL" image which possesses the higher correlation coefficients of the two images considered. An interesting point is that the performance of the reference system does not always improve as the block size is increased or as the the bit rate is increased for a *fixed* channel crossover probability. It does however for the optimal system, a result that is intuitively satisfying.

To illustrate the effect of the noisy channel on the bit allocation matrix we have also presented in Figures 9–11 sample bit allocation matrices at 1 bit/pixel for the optimal system designed for values of $\epsilon = 0.0$, 0.01 and 0.05 . The size of these bit allocation matrices is 32×32 and they have been obtained for the 2-D Gauss-Markov image model having the same correlation coefficients as the "GIRL" image. The sample bit allocations for the optimal system indicate that as the channel crossover probability increases, there is a tendency to allocate more bits to the high energy coefficients (those in the top left corner of the matrix) and thus provide greater channel protection to these coefficients at the expense of more distortion for the lower energy coefficients.

The trends in the simulation results are seen to be similar to those in the numerical results, though the performance improvements are not quite as large. For example, an improvement of 2.3 dB is observed by simulating the system for the "GIRL" image, at a rate of 1 bit/pixel, a block size of 8×8 and for a crossover probability of 0.005. The corresponding figure predicted by the numerical results is approximately 8 dB. We believe that the difference in the numerical results and the simulation results, is most probably a result of inappropriate modeling of the source. The Gaussian assumption is not always a good one as shown in [6]. But what is probably more important is the fact that nonstationary image models are inherently superior to stationary image models [7]. It would be useful to note here that the system optimization method is general enough to handle situations where the transform coefficients are non-Gaussian, with (possibly) distinct p.d.f.'s.

The examples of reconstructed image quality at 1 bit/pixel, confirm the trends indicated by the numerical and simulation results. The image quality for the optimal system is definitely superior to that of the reference system, and the improvements in quality are more noticeable for the highly correlated "GIRL" image than for the "MOON" image.

V Summary and Conclusions

We have optimized the performance of the 2-D discrete cosine transform coding scheme over a noisy channel by (i) optimizing the bit allocation among the transform

coefficients and (ii) designing optimal encoder/decoder pairs for transmission of the transform coefficients over a noisy channel.

We have obtained theoretically predicted performance results based on an assumed 2-D Gauss-Markov model, as well as simulation results for a real-world image. In both cases it is shown that the optimal system offers noticeable performance improvements over the conventional system based on Lloyd-Max quantization of the transform coefficients. The performance improvements are more noticeable at higher bit rates and for noisier channels.

References

[1] J. W. Modestino, D. G. Daut and A. L. Vickers, "Combined Source-Channel Coding of Images Using the Block Cosine Transform," *IEEE Trans. Commun.*, vol. COM-29, pp. 1261–1274, September 1981.

[2] T. Fine, "Properties of an Optimum Digital System and Applications," *IEEE Trans. Inform. Theory,* vol. IT-10, pp. 287–296, October 1964.

[3] A. V. Trushkin, "Optimal Bit Allocation Algorithm for Quantizing a Random Vector," *Problems on Information Transmission,* pp. 156–161, Jan. 1982.

[4] N. Farvardin and V. Vaishampayan, "Optimal Quantizer Design for Noisy Channels: An Approach to Combined Source-Channel Coding," Submitted to *IEEE Trans. Inform. Theory* for publication.

[5] M. J. Narasimha and A. M. Peterson, "On the Computation of the Discrete Cosine Transform," *IEEE Trans. Commun.*, vol. COM-26, pp. 934–936, June 1978.

[6] R. C. Reininger and J. D. Gibson, "Distribution of the Two-Dimensional DCT Coefficients for Images," *IEEE Trans. Commun.*, vol. COM-31, pp. 835–839, June 1983.

[7] B. R. Hunt and T. M. Cannon, "Nonstationary Assumptions for Gaussian Models of Images," *IEEE Trans. Systems, Man, and Cybernetics,* pp. 876–882, December 1976.

IMAGE	MEAN	VARIANCE	ρ_R	ρ_C
MOON	127.23	823.78	0.9017	0.9090
GIRL	73.57	1816.56	0.9790	0.9746

Table 1: Image Statistics.

BLOCK SIZE		$\epsilon = 0.0$		$\epsilon = 0.005$		$\epsilon = 0.01$		$\epsilon = 0.05$	
		REF.	OPT.	REF.	OPT.	REF.	OPT.	REF.	OPT.
8×8	MSE	18.77	18.77	57.37	29.36	95.79	35.29	396.75	78.26
8×8	SNR	16.42	16.42	11.57	14.48	9.34	13.68	3.17	10.22
16×16	MSE	14.38	14.38	53.80	23.97	93.04	29.30	400.43	69.58
16×16	SNR	17.58	17.58	11.85	15.36	9.47	14.49	3.13	10.73
32×32	MSE	12.64	12.64	52.50	21.44	92.19	26.42	402.99	65.71
32×32	SNR	18.14	18.14	11.96	15.85	9.51	14.94	3.105	10.98

Table 2: Numerical Results; 1 bit/pixel; "MOON" Image.

BLOCK SIZE		$\epsilon = 0.0$		$\epsilon = 0.005$		$\epsilon = 0.01$		$\epsilon = 0.05$	
		REF.	OPT.	REF.	OPT.	REF.	OPT.	REF.	OPT.
8×8	MSE	52.69	52.69	84.88	67.99	116.94	76.60	368.47	125.65
8×8	SNR	11.94	11.94	9.87	10.83	8.48	10.32	3.49	8.17
16×16	MSE	40.28	40.28	75.80	52.91	111.16	59.91	388.27	105.88
16×16	SNR	13.11	13.11	10.36	11.92	8.70	11.38	3.27	8.91
32×32	MSE	34.84	34.84	70.74	46.51	106.48	53.03	386.58	97.77
32×32	SNR	13.74	13.74	8.59	12.48	8.88	11.91	3.29	9.26

Table 3: Numerical Results; 0.5 bits/pixel; "MOON" Image.

BLOCK SIZE		$\epsilon = 0.0$		$\epsilon = 0.005$		$\epsilon = 0.01$		$\epsilon = 0.05$	
		REF.	OPT.	REF.	OPT.	REF.	OPT.	REF.	OPT.
8×8	MSE	62.24	62.24	102.44	79.77	147.90	88.28	443.52	143.90
8×8	SNR	11.22	11.22	9.05	10.14	7.46	9.70	2.69	7.58
16×16	MSE	46.04	46.04	80.19	69.92	143.59	75.71	447.05	124.80
16×16	SNR	12.53	12.53	10.11	10.71	7.59	10.37	2.65	8.20
32×32	MSE	31.95	31.95	75.83	43.87	130.76	50.32	481.38	100.46
32×32	SNR	14.11	14.11	10.36	12.73	7.99	12.14	2.33	9.14

Table 4: Simulation Results; 1 bit/pixel; "MOON" Image.

BLOCK SIZE		$\epsilon = 0.0$		$\epsilon = 0.005$		$\epsilon = 0.01$		$\epsilon = 0.05$	
		REF.	OPT.	REF.	OPT.	REF.	OPT.	REF.	OPT.
8×8	MSE	99.34	99.34	140.42	115.90	173.56	124.16	437.47	175.63
8×8	SNR	9.19	9.19	7.68	8.52	6.76	8.22	2.75	6.71
16×16	MSE	75.95	75.95	118.12	101.20	157.12	106.15	442.16	154.56
16×16	SNR	10.35	10.35	8.44	9.11	7.20	8.90	2.70	7.27
32×32	MSE	56.40	56.40	96.12	68.68	136.14	75.47	490.22	124.68
32×32	SNR	11.65	11.65	9.33	10.79	7.82	10.38	2.25	8.20

Table 5: Simulation Results; 0.5 bits/pixel; "MOON" Image.

BLOCK SIZE		$\epsilon = 0.0$		$\epsilon = 0.005$		$\epsilon = 0.01$		$\epsilon = 0.05$	
		REF.	OPT.	REF.	OPT.	REF.	OPT.	REF.	OPT.
8×8	MSE	4.80	4.80	111.62	17.43	217.89	24.86	1047.69	99.98
8×8	SNR	25.78	25.78	12.12	20.18	9.21	18.64	2.39	12.59
16×16	MSE	2.776	2.776	109.27	14.24	215.22	21.02	1042.72	93.36
16×16	SNR	28.16	28.16	12.21	21.06	9.26	19.37	2.41	12.89
32×32	MSE	2.14	2.14	110.02	13.11	217.34	19.61	1055.37	91.01
32×32	SNR	29.29	29.29	12.18	21.42	9.22	19.67	2.36	13.00

Table 6: Numerical Results; 1 bit/pixel; "GIRL" Image.

BLOCK SIZE		$\epsilon = 0.0$		$\epsilon = 0.005$		$\epsilon = 0.01$		$\epsilon = 0.05$	
		REF.	OPT.	REF.	OPT.	REF.	OPT.	REF.	OPT.
8×8	MSE	21.61	21.61	111.69	37.54	201.35	46.91	903.31	126.81
8×8	SNR	19.25	19.25	12.11	16.85	9.55	15.88	3.03	11.56
16×16	MSE	10.13	10.13	114.84	23.60	219.02	31.50	1032.73	108.10
16×16	SNR	22.54	22.54	11.99	18.86	9.19	17.61	2.45	12.25
32×32	MSE	7.16	7.16	110.41	19.61	213.14	26.98	1015.93	101.37
32×32	SNR	24.04	24.04	12.16	19.67	9.31	18.28	2.52	12.53

Table 7: Numerical Results; 0.5 bits/pixel; "GIRL" Image.

BLOCK SIZE		$\epsilon = 0.0$		$\epsilon = 0.005$		$\epsilon = 0.01$		$\epsilon = 0.05$	
		REF.	OPT.	REF.	OPT.	REF.	OPT.	REF.	OPT.
8×8	MSE	47.00	47.00	119.29	70.19	211.67	84.24	842.22	190.63
8×8	SNR	15.87	15.87	11.83	14.13	9.34	13.34	3.34	9.79
16×16	MSE	35.79	35.79	112.92	60.17	247.04	77.46	895.70	184.35
16×16	SNR	17.05	17.05	12.06	14.80	8.66	13.70	3.07	9.94
32×32	MSE	24.74	24.74	105.02	49.86	273.60	67.51	981.89	148.11
32×32	SNR	18.66	18.66	12.38	15.62	8.22	14.30	2.67	10.89

Table 8: Simulation Results; 1 bit/pixel; "GIRL" Image.

BLOCK SIZE		$\epsilon = 0.0$		$\epsilon = 0.005$		$\epsilon = 0.01$		$\epsilon = 0.05$	
		REF.	OPT.	REF.	OPT.	REF.	OPT.	REF.	OPT.
8×8	MSE	80.12	80.12	152.11	109.67	252.78	124.40	827.09	231.58
8×8	SNR	13.55	13.55	10.77	12.19	8.56	11.64	3.42	8.95
16×16	MSE	65.25	65.25	127.21	90.99	246.77	109.70	838.02	217.56
16×16	SNR	14.45	14.45	11.55	13.00	8.67	12.19	3.36	9.22
32×32	MSE	48.65	48.65	133.36	75.70	313.66	94.81	984.55	176.07
32×32	SNR	15.72	15.72	11.34	13.80	7.63	12.82	2.66	10.14

Table 9: Simulation Results; 0.5 bits/pixel; "GIRL" Image.

Original "GIRL" Image.

Ref. designed for $\epsilon = 0.00$.

Opt. designed for $\epsilon = 0.005$,
Actual $\epsilon = 0.00$.

Figure 3: The Original "GIRL" Image and Simulation Results at 1 bit/pixel, Block Size = 32 × 32, for the Optimum System (Opt.), and for the Reference System (Ref.).

Opt. designed for $\epsilon = 0.01$,
Actual $\epsilon = 0.00$.

Opt. designed for $\epsilon = 0.05$,
Actual $\epsilon = 0.0$.

Opt.designed for $\epsilon = 0.005$,
Actual $\epsilon = 0.005$.

Ref.,
Actual $\epsilon = 0.005$.

Figure 4: Simulation Results for the "GIRL" Image at 1 bit/pixel, Block Size= 32×32.

Opt. designed for $\epsilon = 0.01$,
Actual $\epsilon = 0.01$.

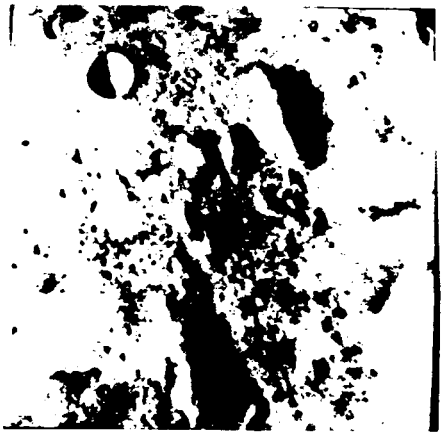
Ref.,
Actual $\epsilon = 0.01$.

Opt. designed for $\epsilon = 0.05$,
Actual $\epsilon = 0.05$.

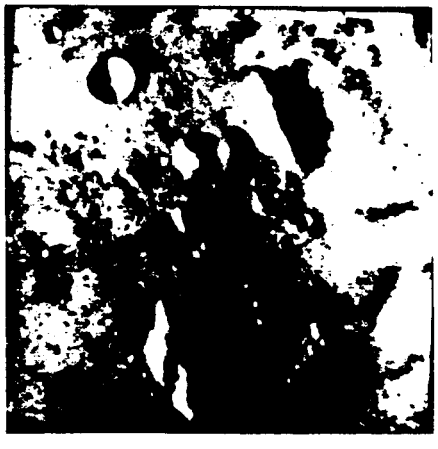
Ref.,
Actual $\epsilon = 0.05$.

Figure 5: Simulation Results for the "GIRL" Image at 1 bit/pixel, Block Size= 32×32.

Original "MOON"
Image.



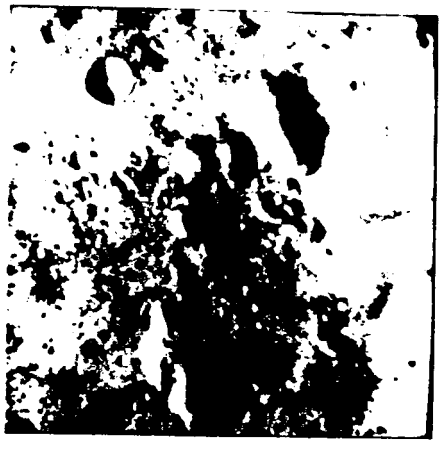
Ref. designed for $\epsilon = 0.00$.



Opt. designed for $\epsilon = 0.005$,
Actual $\epsilon = 0.00$.

Figure 6: The Original "MOON" Image and Simulation Results at 1 bit/pixel, Block Size = 32×32, for the Optimum System (Opt.), and for the Reference System (Ref.).

Opt. designed for $\epsilon = 0.01$,
Actual $\epsilon = 0.00$.

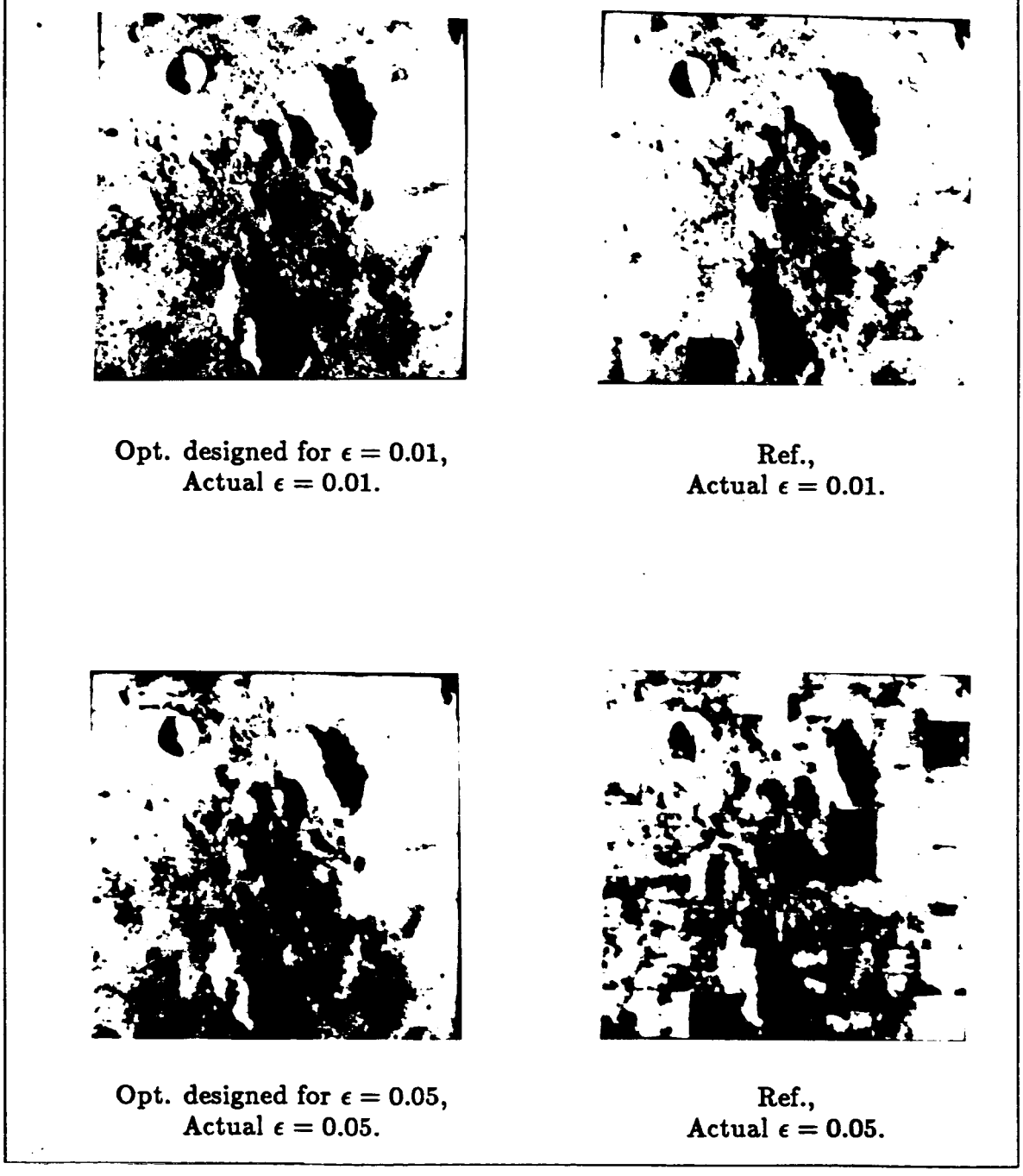


Opt. designed for $\epsilon = 0.05$,
Actual $\epsilon = 0.0$.

Opt.designed for $\epsilon = 0.005$,
Actual $\epsilon = 0.005$.

Ref.,
Actual $\epsilon = 0.005$.

Figure 7: Simulation Results for the "MOON" Image at 1 bit/pixel, Block Size= 32×32.



Opt. designed for $\epsilon = 0.01$,
Actual $\epsilon = 0.01$.

Ref.,
Actual $\epsilon = 0.01$.

Opt. designed for $\epsilon = 0.05$,
Actual $\epsilon = 0.05$.

Ref.,
Actual $\epsilon = 0.05$.

Figure 8: Simulation Results for the "MOON" Image at 1 bit/pixel, Block Size= 32×32.

8	8	8	7	7	6	6	6	6	6	6	5	5	5	5	5	5	5	5	5	5	5	5	5	4	4	4	4	4	4	4	4
8	7	6	6	5	5	5	5	4	4	4	4	4	4	4	4	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
8	6	6	5	5	4	4	4	4	3	3	3	3	3	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
7	6	5	4	4	4	3	3	3	3	3	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1
7	5	5	4	4	3	3	3	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	5	4	4	3	3	3	2	2	2	2	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
6	5	4	3	3	3	2	2	2	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	5	4	3	3	2	2	2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	4	4	3	2	2	2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	4	3	3	2	2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	4	3	3	2	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	4	3	2	2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4	3	2	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4	3	2	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4	3	2	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4	3	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 9: Sample Bit Allocation Matrix for the Optimal System, Block Size 32×32, $\epsilon = 0.0$, 1 bit/pixel.

8	8	8	8	8	8	8	8	8	8	8	8	8	7	7	7	7	7	7	6	6	6	6	6	6	6	6	6	6	6	6	6
8	8	8	8	8	7	7	6	6	5	5	5	4	3	3	3	3	3	3	3	3	3	3	3	3	2	2	2	2	2	2	2
8	8	8	7	6	5	5	3	3	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1
8	8	7	6	5	3	3	3	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	8	6	5	3	3	2	2	2	2	2	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	7	5	3	3	2	2	2	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	7	5	3	2	2	2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	6	3	3	2	2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	6	3	2	2	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	5	3	2	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	5	3	2	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	5	3	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	4	2	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	4	2	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 10: Sample Bit Allocation Matrix for the Optimal System, Block Size 32×32, $\epsilon = 0.01$, 1 bit/pixel.

8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	7	7	7	7	7	7	7	6	6
8	8	8	8	8	8	8	8	7	6	6	6	6	5	3	3	3	3	3	3	3	3	2	2	2	2	2	2	2	2	2	2
8	8	8	8	8	6	6	6	3	3	3	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	8	8	7	6	3	3	3	2	2	2	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
8	8	8	6	3	3	2	2	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	8	6	3	3	2	2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	8	6	3	2	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	8	6	3	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	7	3	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	6	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	6	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	6	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	6	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	6	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 11: Sample Bit Allocation Matrix for the Optimal System, Block Size 32×32, $\epsilon = 0.05$, 1 bit/pixel.