

- Proc. 6th IBM Medical Symp.*, pp. 545-551, Oct. 1964.
- [114] C. Vallbona, E. Pevny, and F. McMath, "Computer analysis of blood gases and of acid-base status," *Comput. Biomed. Res.*, vol. 4, pp. 623-633, Dec. 1971.
- [115] J. M. Vanderplas, "A method for determining probability for correct use of Bayes' theorem in medical diagnosis," *Comput. Biomed. Res.*, vol. 1, pp. 215-220, 1967.
- [116] A. J. Van Woerkom and K. Brodman, "Statistics for a diagnostic model," *Biometrics*, pp. 299-318, June 1961.
- [117] H. R. Warner, C. M. Olmsted and B. D. Rutherford, "HELP—a program for medical decision making," *Comput. Biomed. Res.*, vol. 5, pp. 65-74, 1972.
- [118] H. R. Warner, B. D. Rutherford, and B. A. Houtches, "A sequential Bayesian approach to history taking and diagnosis," *Comput. Biomed. Res.*, vol. 5, pp. 256-262, June 1972.
- [119] H. R. Warner, A. F. Toronto, and L. G. Veasy, "Experience with Bayes' theorem for computer diagnosis of congenital heart disease," *Ann. N.Y. Acad. Sci.*, vol. 115, pp. 558-567, 1964.
- [120] H. R. Warner, A. F. Toronto, L. G. Veasey and R. Stephenson, "A mathematical approach to medical diagnosis application to congenital heart disease," *J. Amer. Med. Assoc.*, vol. 177, pp. 177-183, 1961.
- [121] J. Wartak, "A practical approach to automated diagnoses," *IEEE Trans. Biomed. Eng.*, vol. BME-17, pp. 37-43, Jan. 1970.
- [122] J. Wartak, J. A. Milliken, and J. Karchmar, "Computer program for pattern recognition of electrocardiograms," *Comput. Biomed. Res.* vol. 3, pp. 344-374, Aug. 1970.
- [123] J. Wartak, J. A. Milliken, and J. Karchmar, "Computer program for diagnostic evaluation of electrocardiograms," *Comput. Biomed. Res.*, vol. 4, pp. 225-238, June 1971.
- [124] P. M. Wortman, "Medical diagnosis: An information processing approach," *Comput. Biomed. Res.*, vol. 5, pp. 315-328, Aug. 1972.
- [125] R. D. Yoder, R. E. Bonner, and H. C. Becker, "Pattern recognition of electroencephalograms," in *Proc. 5th IBM Medical Symp.*, pp. 421-436, 1963.
- [126] S. Y. A. Zaslavsky and K. A. Ivanov-Muromsky, "Solving systems prognosticating disease course," *Math. Biosci.*, vol. 3, pp. 243-263, 1970.
- [127] I. K. Zola, "The use of aides in a management program for chronic disease," M.I.T. Lincoln Lab., Lexington, Mass., Rep. ACP-13, Dec. 1970.

# Comparative Study of a Discrete Linear Basis for Image Data Compression

ROBERT M. HARALICK, MEMBER, IEEE, AND K. SHANMUGAM, MEMBER, IEEE

**Abstract**—Transform image data compression consists of dividing the image into a number of nonoverlapping subimage regions and quantizing and coding the transform of the data from each subimage. Karhunen-Loève, Hadamard, and Fourier transforms are most commonly used in transform image compression. This paper presents a new discrete linear transform for image compression which we use in conjunction with differential pulse-code modulation on spatially adjacent transformed subimage samples. For a set of thirty-three  $64 \times 64$  images of eleven different categories, we compare the performances of the discrete linear

transform compression technique with the Karhunen-Loève and Hadamard transform techniques. Our measure of performance is the mean-squared error between the original image and the reconstructed image. We multiply the mean-squared error with a factor indicating the degree to which the error is spatially correlated. We find that for low compression rates, the Karhunen-Loève outperforms both the Hadamard and the discrete linear basis method. However, for high compression rates, the performance of the discrete transform method is very close to that of the Karhunen-Loève transform. The discrete linear transform method performs much better than the Hadamard transform method for all compression rates.

Manuscript received February 21, 1973; revised July 9, 1973.  
R. M. Haralick is with the Center for Research, Inc., and the Department of Electrical Engineering, University of Kansas, Lawrence, Kans. 66044.

K. Shanmugam was with the Center for Research, Inc., and the Department of Electrical Engineering, University of Kansas, Lawrence, Kans. He is now with the Department of Electrical Engineering, Wichita State University, Wichita, Kans. 67218.

## I. INTRODUCTION

**T**HE OBJECTIVE of efficient compression (coding) of image data is to remove the redundancy from the image in transmission across a digital communication link.

Two important classes of compression schemes that make use of the spatial correlation of image elements are the differential pulse-code modulation (DPCM) schemes and transform compression techniques. DPCM techniques are based on the principle that it is more efficient to code the difference between the gray-tone values of adjacent image elements than the actual values because of the high correlation between the gray-tone values of adjacent image elements. In transform data compression the mathematical transform of the imagery data from nonoverlapping subimage regions is taken and the transform coefficients which contain a significant portion of the image energy per unit area are coded and transmitted. The reconstruction of the imagery is done using the decoded values of the transform coefficients and the appropriate basis vectors which were used in transforming the data.

The idea of linear transforms and block quantization was suggested by Huang and Woods [1], [2]. Various linear transform techniques include the Fourier, Hadamard, and Karhunen-Loève transforms. Fourier, Hadamard, and "slant" transforms have been studied in detail by Pratt *et al.* [3], [4]. Habibi and Wintz [5] have compared the performance of Hadamard, Fourier, and Karhunen-Loève transforms on a set of four  $256 \times 256$  pictures at 0.5, 1.0, and 2.0 b/picture element.

An adaptive version of the transform data compression technique was investigated by Tasto and Wintz [6] and a comparison of the performance of DPCM and transform techniques may be found in [7]. There is an excellent summary of transform coding techniques by Wintz [8].

The general consensus of the previous studies on image data compression techniques is that the Fourier and Hadamard transforms provide adequate domains for coding even though their performance is inferior to those obtained by the Karhunen-Loève transform. However, the fast versions of the Hadamard and Fourier transforms provide considerable simplification in implementation.

We present a new discrete linear basis (DLB) for transform data compression which offers some of the computational advantages of the Fourier and Hadamard transforms without sacrificing the performance compared to the Karhunen-Loève transform. The basis vectors are calculated *a priori*, as in the Hadamard and Fourier transforms, and a fast implementation of the DLB transform is also possible.

In much of the earlier work on transform data compression, the transformed samples of the imagery data from spatially adjacent subimage regions were quantized independently using the simple Max's nonuniform quantizer [9] or a sophisticated coding scheme such as the Huffman code. We found that while the transformed samples of data from one subimage region were uncorrelated, in general there was high correlation between transformed samples of data from two spatially adjacent subimages. Hence we used a differential coding scheme to remove some of this redundancy in transform samples from adjacent subimages. The use of a DPCM scheme on the transformed samples was first suggested by Haralick *et al.* [10]. Recently, Habibi

[11], [12], and Arisana and Kanaya [13] have also presented results on various possible ways of using DPCM systems to quantize transformed data.

We compared the performance of the DLB transform with that of the Hadamard and Karhunen-Loève transform techniques on a set of thirty-three  $64 \times 64$  images using a subimage size of  $4 \times 4$ . For the subimage size we had chosen, which was recommended by Wintz [8], the Fourier transform does not offer any significant improvement in performance over the Hadamard transform, and hence we did not consider the Fourier transform in our studies. In addition to using the mean-squared error and subjective picture quality as measures of performance, we also used a correlated error measure for comparing the performance of the DLB transform with the Karhunen-Loève and Hadamard transforms. Based on the comparative performance on the set of 33 images, we conclude the DLB transform offers a good tradeoff between the ease of implementation of the Hadamard transform and the superior performance of the Karhunen-Loève transform.

## II. TRANSFORM IMAGE DATA COMPRESSION

Since transform image data compression techniques have been studied in detail by a number of investigators, only a brief outline of the technique will be presented here. The first step in implementing a transform data compression technique is to divide the  $M \times M$  image into a set of nonoverlapping subimages (or windows) of size  $n \times n$ ,  $n < M$ . If we interpret an  $n \times n$  subimage as a point in an  $n^2$ -dimensional space where each of the  $n^2$  coordinates corresponds to one of the  $n^2$  pels in the subimage, then the image is a collection of  $k$ ,  $k = (M/n)^2$ , vectors of dimension  $n^2$ . We will let  $N$  denote the value of  $n^2$ , and  $X_1, X_2, \dots, X_k$  the collection of the data vectors from  $k$  nonoverlapping windows in the image.

The second step in transform image data compression is to project the  $N$ -dimensional data into an  $r$ -dimensional subspace using a set of  $r$  basis vectors  $V_1, V_2, \dots, V_r$ . The  $j$ th projection of the data vector  $X_i$  is given by

$$y_{ij} = V_j^T X_i \quad (1)$$

where  $V_j^T = (V_{j1}, V_{j2}, \dots, V_{jN})$  is the transpose of the  $j$ th basis vector.

The  $r$  projections of each of the data vectors  $X_i$  are coded independently and the code is transmitted over a binary channel to the receiver. The reconstruction of the data vector  $X_i$  at the receiver takes place using the decoded values  $\hat{y}_{ij}$  of  $y_{ij}$  according to

$$\hat{X}_i = \sum_{j=1}^r \hat{y}_{ij} V_j \quad (2)$$

where  $\hat{X}_i$  is the reconstructed value of  $X_i$ .

If we have a noise-free channel, then there are two sources of error in a transform image coding scheme, namely, the truncation error and the quantization error. The truncation error can be reduced by a suitable choice of the basis vectors and the number of projections used, and the quantization

error can be reduced by the use of an efficient coding scheme for the projections.

In the principal components method of transform data compression, the basis vectors are comprised of the  $r$  eigenvectors of the sample covariance matrix of  $X_1, X_2, \dots, X_k$  corresponding to the  $r$  largest eigenvalues of the covariance matrix. In the Hadamard transform method the projections onto the  $r$  lower sequency Walsh functions are used. Complete details of these procedures may be found in [8].

In much of the earlier work in transform image data compression, the projections of the data vectors have been coded independently. The number of bits assigned to the  $j$ th projection is either proportional to the variance of the projection or proportional to the logarithm of the variance of the projection. Bit assignments proportional to variance seem to yield better pictures [5]. Other variations in the quantization (coding) procedure may be found in [8]. Using an adaptive version of the Karhunen-Loève transform with different Huffman coding for each projection, Tasto and Wintz [6] have proposed a coding scheme whose mean-squared-error performance nearly attains the upper bound on the rate-distortion function [14], [15] for discrete ergodic sources with memory.

The selection of a particular set of basis vectors for transform image coding can be justified if the image information lost in using the set of basis vectors with the associated quantizing procedure can be quantified by a measure of distortion. At this point, mathematical tractability seems to dictate that the distortion measure be of the mean-squared-error type. For Gaussian sources it is possible to specify the minimum bit rate  $R$  for a fixed mean-squared-error  $D$  parametrically as [15]

$$R(D) = \sum_{\lambda_i > \epsilon} \frac{1}{2} \log_2 \frac{\lambda_i}{\epsilon}$$

$$D = \sum_{\lambda_i > \epsilon} \epsilon + \sum_{\lambda_i \leq \epsilon} \lambda_i \quad (3)$$

where  $\lambda_i$  are the eigenvalues of the covariance matrix of the source random process. This optimum performance can be approached by applying the Karhunen-Loève (principal components) transformation and quantizing the transform coefficients separately. In spite of some of the simplified procedures suggested [16] for finding and implementing the Karhunen-Loève transform, the fact that the procedure is image dependent makes it computationally complex, and this has restricted its use.

Fortunately, the Karhunen-Loève expansion for a discrete sample of a stationary process is approximately given by a Fourier expansion. Also, there is experimental evidence that replacing the finite Fourier matrix with the Walsh-Hadamard matrix does not result in a serious degradation [17], [18]. Hence, the Fourier and Hadamard transforms have been widely used. Another factor in favor of their use is their ease of implementation. For a subimage size  $n \times n$ , the Hadamard transform requires  $2n^2 \log_2 n^2$  operations as compared to  $2n^4$  operations required for the principal components transformation.

### III. DISCRETE LINEAR TRANSFORM

#### Basis Vectors

The basis vectors which define the new discrete linear transform are generated by requiring that 1) the  $N$  vectors  $V_1, V_2, \dots, V_N \in R^N$  be mutually orthogonal; 2) the components of the vectors be integer valued with no common factors; and 3) each vector be either odd or even

$$V = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{pmatrix} \text{ is } \begin{cases} \text{even,} & \text{if } a_i = a_{N+1-i}, & i = 1, 2, \dots, N \\ \text{odd,} & \text{if } a_i = -a_{N+1-i}, & i = 1, 2, \dots, N. \end{cases}$$

We suggest one way to generate such a basis. First, let us consider generating the even vectors. Let

$$\left[ \frac{N}{2} \right] = \begin{cases} \frac{N}{2}, & \text{if } N \text{ is even} \\ \frac{N+1}{2}, & \text{if } N \text{ is odd.} \end{cases}$$

There will be a total of  $[N/2]$  even vectors in the basis. We begin by taking all of the components of the first even vector to be unity. Suppose we have generated even vectors  $V_1, V_2, \dots, V_{k-1}$ ; then we generate  $V_k$  as follows. Let

$$V_k = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{[N/2]} \\ a_{[N/2]} \\ \vdots \\ a_2 \\ a_1 \end{pmatrix}, \quad \text{if } N \text{ is even}$$

$$V_k = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{[N/2]} \\ \vdots \\ a_2 \\ a_1 \end{pmatrix}, \quad \text{if } N \text{ is odd.}$$

We require that the components of  $V_k$  satisfy the following  $[N/2] - k$  constraints:

$$ra_1 - sa_k = ra_2 - sa_{k+1} = \dots = ra_{[N/2]-k+1} - sa_{[N/2]} \quad (3)$$

In addition to (3), we require that  $V_k$  be orthogonal to  $V_1, V_2, \dots, V_{k-1}$ . Hence we have a total of  $[N/2] - 1$  constraints on the  $[N/2]$  components of  $V_k$ . We can use the one remaining degree of freedom to make the components have integer values without any integer common factor. We know we can do this since we require the  $[N/2]$  components to satisfy  $[N/2] - 1$  linear equations having integer coefficients. Therefore, besides the irrational solutions, there are an infinite number of rational solutions. We can take any one of these solutions and multiply out the denominators to obtain the integer-valued vector.

Now we consider the odd vectors. Let

$$\binom{N}{2} = \begin{cases} \frac{N}{2}, & \text{if } N \text{ is even} \\ \frac{N-1}{2}, & \text{if } N \text{ is odd.} \end{cases}$$

There are  $(N/2)$  components to be determined for any odd vector. Suppose we have generated odd vectors  $W_1, \dots, W_{k-1}$ . We generate  $W_k$  as follows. Let

$$W_k = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{(N/2)} \\ 0 \\ -b_{(N/2)} \\ \vdots \\ -b_2 \\ -b_1 \end{pmatrix}, \quad \text{if } N \text{ is odd}$$

$$W_k = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{(N/2)} \\ -b_{(N/2)} \\ \vdots \\ -b_2 \\ -b_1 \end{pmatrix}, \quad \text{if } N \text{ is even.}$$

Require

$$rb_1 - sb_{k+1} = rb_2 - sb_{k+2} = \dots = rb_{(N/2)-k+1}$$

and that  $W_k$  be orthogonal to  $W_1, W_2, \dots, W_{k-1}$ . These requirements represent  $(N/2) - 1$  constraints. Since there are  $(N/2)$  components to be determined, we have one degree of freedom left, and we can use this degree of freedom to make the components have integer values without any integer common factor. The  $N$  basis vectors generated using the previous procedure are then rearranged such that the first basis vector has no sign changes (sequency 0), the second vector has one sign change (sequency 1) and so on. That is, the basis vectors are arranged in the ascending order of their sequencies. Although we have not proven it, we have conjectured that the procedure generates basis vectors having sequencies 0 through  $N - 1$ , one basis vector per sequency. With  $r = 1, s = -2$ , and  $N = 4$ , the four basis vectors are

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 3 \\ 1 \\ -1 \\ -3 \end{pmatrix} \quad \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 1 \\ -3 \\ 3 \\ -1 \end{pmatrix}$$

The five basis vectors for  $N = 5$  (with  $r = 1, s = -1$  for even vectors, and  $r = s = 1$  for odd vectors) are

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 2 \\ 1 \\ 0 \\ -1 \\ -2 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 0 \\ -2 \\ 0 \\ 1 \end{pmatrix} \quad \begin{pmatrix} -1 \\ 2 \\ 0 \\ -2 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 2 \\ -3 \\ 2 \\ -3 \\ 2 \end{pmatrix}$$

With  $r = 1, s = 1$ , and  $N = 8$ , the eight basis vectors are

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 4 \\ 3 \\ 2 \\ 1 \\ -1 \\ -2 \\ -3 \\ -4 \end{pmatrix} \quad \begin{pmatrix} 3 \\ 1 \\ -1 \\ -3 \\ -3 \\ -1 \\ 1 \\ 3 \end{pmatrix} \quad \begin{pmatrix} 0 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \end{pmatrix} \quad \begin{pmatrix} -3 \\ 2 \\ 4 \\ -2 \\ 2 \\ -4 \\ -2 \\ 3 \end{pmatrix} \quad \begin{pmatrix} 1 \\ -3 \\ 3 \\ -1 \\ -1 \\ 3 \\ -3 \\ 1 \end{pmatrix} \quad \begin{pmatrix} 8 \\ -9 \\ 4 \\ -13 \\ 13 \\ -4 \\ 9 \\ 8 \end{pmatrix}$$

For certain values of  $r$  and  $s$ , some of the basis vectors of the DLB transform are the same as some of the basis vectors of the slant transform (which is defined for values of  $N$  equal to integer powers of 2). In general, however, the basis vectors of the DLB transform and slant transforms are different.

#### Implementation of the DLB Transform

The projection operations and the reconstruction for the DLB transform can be implemented directly using (1) and (2). The number of operations required for computing all the  $N$  projections for a data vector in  $R^N$  is  $2N^2$ , which is the same as the number of operations required for implementing the Karhunen-Loève transform. However, this is considerably more than the  $2N \log_2 N$  operations required for implementing the Hadamard transform. It is possible to reduce the number of operations required to obtain the DLB transform by using a fast algorithm, described below.

The fast algorithm is based on the principle that the basis vectors for  $R^N$  can be represented by taking the proper dot products of the basis vectors of two lower dimensional subspaces  $R^{N_1}$  and  $R^{N_2}$ , where  $N_1$  and  $N_2$  are such that  $N = N_1 N_2$ . Let  $U = (u_1, u_2, \dots, u_{N_1})^T$  and  $V = (v_1, v_2, \dots, v_{N_2})^T$  be two vectors of dimension  $N_1$  and  $N_2$ , respectively. Then we can define a product vector of dimension  $N_1 N_2$  by

$$UV = (u_1 v_1, u_1 v_2, \dots, u_1 v_{N_2}, u_2 v_1, u_2 v_2, \dots, u_2 v_{N_2}, \dots, u_{N_1} v_1, \dots, u_{N_1} v_{N_2})^T \quad (4)$$

Suppose we have a set of vectors  $U_1, U_2, \dots, U_{N_1}$  which form an orthogonal basis for  $R^{N_1}$ , and a set of vectors  $V_1, V_2, \dots, V_{N_2}$  which form an orthogonal basis for  $R^{N_2}$ ; then we can define a set of product vectors

$$\{U_i V_j \mid i = 1, 2, \dots, N_1; j = 1, 2, \dots, N_2\}$$

as defined in (4). We will show that  $\{U_i V_j\}$  is an orthogonal basis for  $R^{N_1 N_2}$ , and by repeated construction we will develop a fast algorithm for the DLB transform which will

require a considerably smaller number of operations than the  $2N^2$  operations required by the direct implementation.

*Lemma:* Let  $\{X_1, X_2, \dots, X_{N_1}\}$  be an orthogonal basis for  $R^{N_1}$  and  $\{Y_1, Y_2, \dots, Y_{N_2}\}$  be an orthogonal basis for  $R^{N_2}$ . Then  $\{X_i Y_j \mid i = 1, 2, \dots, N_1; j = 1, 2, \dots, N_2\}$  is an orthogonal basis for  $R^{N_1 N_2}$ .

Proof of this lemma is given in the Appendix.

We will now develop a fast algorithm to represent an  $N$ -dimensional vector  $V$  in terms of the product vector of the basis vectors in lower dimensional spaces. Let  $N = N_1 N_2 \dots N_k$  and let  $\{X_i^k \mid i = 1, \dots, N_k\}$  be an orthogonal basis vector for  $N_k$ . Then, by successive application of the preceding lemma,

$$B = \{X_{i_1}^1 X_{i_2}^2 \dots X_{i_k}^k \mid i_1 = 1, \dots, N_1; i_2 = 1, \dots, N_2; \dots; i_k = 1, \dots, N_k\} \quad (5)$$

is an orthogonal basis for  $R^N$ . The usual way to represent  $V$ ,  $V = (v_1, v_2, \dots, v_N)^T$  in terms of basis  $B$  is to take the dot product of  $V$  (projection of  $V$ ) with each of the vectors in  $B$ . This requires  $(N_1 N_2 \dots N_k)^2$  operations of addition and multiplication. We can reduce the number of add and multiply operations to  $(N_1 + N_2 + N_k) \times N_1 N_2 \dots N_k$  by the following algorithm.

Consider the calculation of  $w_{i_1, i_2, \dots, i_k}$ , the dot product of the vector  $V$  with the basis vector  $X_{i_1}^1 X_{i_2}^2 \dots X_{i_k}^k$  for all  $i_1, i_2, \dots, i_k$ . For simplicity in calculation, we use a  $k$ -dimensional indexing notation on the  $N_1 N_2 \dots N_k$  components of the vector  $V$ . The components of  $V$  will be denoted by  $v$  with a  $k$ -dimensional index. An index of  $(j_1, j_2, \dots, j_k)$  on  $v$  means the component

$$(j_1 - 1)N_2 N_3 \dots N_k + \dots + (j_{k-2} - 1)N_{k-1} \dots N_k + (j_{k-1} - 1)N_k + j_k.$$

We begin with

$$\begin{aligned} w_{i_1, i_2, \dots, i_k} &= \sum_{j_1=1}^{N_1} \sum_{j_2=1}^{N_2} \dots \sum_{j_k=1}^{N_k} X_{j_1}^{i_1} X_{j_2}^{i_2} \dots X_{j_k}^{i_k} v_{j_1, j_2, \dots, j_k} \\ &= \sum_{j_1=1}^{N_1} X_{j_1}^{i_1} \sum_{j_2=1}^{N_2} X_{j_2}^{i_2} \dots \sum_{j_k=1}^{N_k} X_{j_k}^{i_k} v_{j_1, j_2, \dots, j_k}. \end{aligned} \quad (6)$$

Now, to determine

$$\sum_{j_k=1}^{N_k} X_{j_k}^{i_k} v_{j_1, j_2, \dots, j_k}$$

for each  $i_k, j_1, j_2, \dots, j_{k-1}$  requires  $N_k$  add and multiply operations. There are a total of  $(N_k N_1 N_2 \dots N_{k-1})N_k$  add and multiply operations. Let  $v_{j_1, j_2, \dots, j_{k-1}}^{i_k}$  denote the last summation on the right-hand side of (6); then

$$w_{i_1, i_2, \dots, i_k} = \sum_{j_1=1}^{N_1} X_{j_1}^{i_1} \sum_{j_2=1}^{N_2} X_{j_2}^{i_2} \dots \sum_{j_{k-1}=1}^{N_{k-1}} v_{j_1, j_2, \dots, j_{k-1}}^{i_k}. \quad (7)$$

Now, to determine the value of the last summation on the right-hand side of (7) for each  $i_{k-1}, i_k, j_1, j_2, \dots, j_{k-2}$  requires  $N_{k-1}$  operations and there are a total of  $(N_{k-1} N_k N_1 \dots N_{k-2})N_{k-1}$  add and multiply operations. Continuing

this reasoning, there are  $k$  summations. The  $l$ th summation requires  $(N_l N_{l+1} \dots N_k N_1 N_2 \dots N_{l-1})N_l$  operations. Hence the total number of add and multiply operations required is

$$\begin{aligned} T &= \sum_{l=1}^k (N_l N_{l+1} \dots N_k N_1 N_2 \dots N_{l-1})N_l \\ &= (N_1 N_2 \dots N_k)(N_1 + N_2 + \dots + N_k). \end{aligned} \quad (8)$$

Let us now compare the number of operations required for different transforms for  $N = 16$  (subimage size  $4 \times 4$ ). The number of operations required for obtaining the Karhunen-Loève transform is 512. The number of operations required for the Hadamard transform will be obtained from (8), with  $k = 4$ ,  $N_1 = N_2 = N_3 = N_4 = 2$ , as 128, noting that there are no multiplication operations in obtaining the Hadamard transform. If we implement the DLB transform using the four vectors given in the previous section, the total number of operations required can be obtained from (8), with  $k = 2$ ,  $N_1 = 4$ ,  $N_2 = 4$ , as 256. These 256 operations for the DLB can be carried out in integer arithmetic, whereas the 512 operations for the Karhunen-Loève transform must be done in floating point arithmetic since the eigenvectors have components with irrational values. Also, the Karhunen-Loève transform requires eigenvector computations. Hence it can be seen that the DLB transform is computationally much simpler than the Karhunen-Loève transform. However, it requires more operation than the Hadamard transform.

The DLB transform has considerable advantages over the Karhunen-Loève transform and the Hadamard transform for subimage size  $6 \times 6$  for which fast Hadamard transform does not exist. If we implement the DLB using  $k = 2$ ,  $N_1 = 6$ ,  $N_2 = 6$ , the number of integer operations required is 864. The Karhunen-Loève transform requires 2592 floating point operations.

#### IV. QUANTIZATION PROCEDURE

After the  $N$  picture elements in each  $n \times n$  subimage region have been transformed into  $r \leq N$  projections, each projection must be quantized and coded. Many quantization procedures have been suggested in the past. These procedures are designed to minimize the mean-square error in quantization or to improve the subjective picture quality by coding only those projections which correspond to the spatial frequencies to which the eye is most sensitive. All of these procedures, however, quantize the projections of data from a subimage region independent of the projections of data from another spatially adjacent subimage region. We suggest a differential quantizing scheme to remove the redundancy in the projections of data from spatially adjacent subimage regions.

Consider a sequence of projections  $Y_{1k}, Y_{2k}, \dots, Y_{Kk}$ , which are projections of data vectors  $X_1, X_2, \dots, X_K$  onto the  $k$ th basis vectors. If the data vectors  $X_1, X_2, \dots, X_K$  are from spatially adjacent subimage regions, then the sequence of projections  $Y_{1k}, Y_{2k}, \dots, Y_{Kk}$  will be correlated. Hence a differential coding of this correlated sequence of projections will be more efficient than direct coding of the projections. Depending on the size of the subimage region and the

spatial correlation function of the image, the differential coding scheme can be designed with an appropriate predictor.

In our comparative studies, we used a differential coder with a one-stage predictor for coding these coefficients. The quantizer levels were determined using an equal-probability algorithm on the estimated probability distribution of the first-order difference sequence.

The number of bits used for coding the various coefficient sequences were determined as follows. Let  $\sigma_1^2, \sigma_2^2, \dots, \sigma_r^2$  be the variances of the  $r$  coefficients and let  $n_1, n_2, \dots, n_r$  be the number of bits used to code each coefficient. Given the total number of bits NB which we want to use, the number of bits used to code each coefficient is determined from  $2^{n_i} = \lceil \alpha \sigma_i^2 \rceil$ , where  $\alpha$  is related to NB and  $\sigma_i^2, i = 1, 2, \dots, r$ , by

$$2^{\text{NB}} = \prod_{i=1}^r \alpha \sigma_i^2. \quad (10)$$

Since the  $n_i$  are integers and we do not impose the constraint  $\sum n_i = \text{NB}$ , the actual number of bits used for coding may be slightly larger or lower than NB.

Various possible ways of using DPCM systems to quantize the transformed data are also discussed by Habibi [11], [12] and Arisana and Kanaya [13]. In particular, a variation in the way we assign bits would be to assign bits to the DPCM system in proportion to the variance of the differential signal in the DPCM system.

## V. COMPARISON OF PERFORMANCE

### Description of the Image Set Used

For comparing the performance of the DLB transform data compression technique with the Karhunen-Loève and Hadamard transforms we used a set of 33 images of size  $64 \times 64$  resolution cells. These images were obtained by digitizing  $\frac{1}{8} \times \frac{1}{8}$ -in sections of standard 1:20 000,  $9 \times 9$ -in aerial photography supplied by the United States Army Engineer Topographic Laboratories. Each image was digitized into a  $64 \times 64$  resolution-cell matrix and the levels of digitization ranged from 1 to 64. There were 11 scene categories in the image set comprised of scrub, wood, river, orchard, marsh, swamp, old residential, new residential, urban, road, and railroad yard scenes.

### Indexes of Performance

Three performance indexes are used in comparing the performance of the different image coding techniques. These measures of performance are 1) the root-mean-square (rms) error between the original and the reconstructed images, 2) visual comparison of reconstructed images, and 3) correlated rms error.

The gray values of the reconstructed images are first rounded off to integers, the error images are obtained, and the rms error is computed from the error picture. The correlated rms error is the product of the rms error times a spatial correlation measure obtained from the error image. The correlation measure, which is a measure of the spatial, neighborly dependence of the error within the image, is computed as follows.

Let  $L_x = \{1, 2, \dots, N_x\}$  and  $L_y = \{1, 2, \dots, N_y\}$  be the  $x$  and  $y$  spatial domains of the error picture, and let  $G = \{1, 2, \dots, N_g\}$  be the set of gray tones in the error image. We first compute four matrices of relative frequencies  $P(i, j, d, \theta)$ , with which two neighboring resolution cells separated by a distance of  $d$  occur in the error image, one with gray-tone value  $i$  and the other with gray-tone value  $j$ . The angular relationship between the neighboring resolutions cells is denoted by  $\theta$ . Formally, for angles quantized to 45 intervals, the unnormalized frequencies are defined by

$$P(i, j, d, 0^\circ) = \#\{((k, l), (m, n)) \in (L_y \times L_x) \times (L_y \times L_x) \mid k - m = 0, |l - n| = d, I(k, l) = i, I(m, n) = j\}$$

$$P(i, j, d, 45^\circ) = \#\{((k, l), (m, n)) \in (L_y \times L_x) \times (L_y \times L_x) \mid (k - m = d, l - n = -d) \text{ or } (k - m = -d, l - n = d), I(k, l) = i, I(m, n) = j\}$$

$$P(i, j, d, 90^\circ) = \#\{((k, l), (m, n)) \in (L_y \times L_x) \times (L_y \times L_x) \mid |k - m| = d, l - n = 0, I(k, l) = i, I(m, n) = j\}$$

$$P(i, j, d, 135^\circ) = \#\{((k, l), (m, n)) \in (L_y \times L_x) \times (L_y \times L_x) \mid (k - m = d, l - n = d) \text{ or } (k - m = -d, l - n = -d), I(k, l) = i, I(m, n) = j\} \quad (11)$$

where  $I: L_y \times L_x \rightarrow G$ , and  $\#$  denotes the number of elements in the set. Note that these matrices are symmetric:  $P(i, j; d, \theta) = P(j, i; d, \theta)$ . The distance metric  $d$  implicit in the preceding equations can be explicitly defined by  $d((k, l), (m, n)) = \max\{|k - m|, |l - n|\}$ .

Let  $P(i, j)$  denote one of the normalized gray-tone transition matrices. Then the correlation measure is defined by the following equations:

$$C = -\sum_i \sum_j P(i, j) \log \left\{ \frac{P(i, j)}{P(i)P(j)} \right\}$$

$$h = -\sum_k P(k) \log P(k)$$

$$P(k) = \sum_i P(i, k) = \sum_i P(k, i)$$

and

$$\rho = \frac{C}{h}. \quad (12)$$

The correlation measure  $\rho$  defined in (12) is also a measure of independence between the variables of the distribution  $P(i, j)$ . For details of the properties of the correlation measure, the reader is referred to [19].

After the correlation measure has been computed for each of the four angular gray-tone transition matrices, the average value of  $\rho$  is calculated. The average correlation measure contains information about the nature of the spatial distribution of the error picture and the rms error gives an idea of the magnitude of the error between the original and reconstructed images. Thus the product of  $\rho$  and rms error is a measure of the magnitude and the spatial distribution of the errors.

TABLE I  
PERFORMANCE OF KARHUNEN-LOÈVE TRANSFORM METHOD

SCENE NO	BITPEL	RMS ERROR	CORREL ERROR	BITPEL	RMS ERROR	CORREL ERROR	BITPEL	RMS ERROR	CORREL ERROR
SCRUB-1	0.740	0.50870	0.02495	1.210	0.47930	0.01946	1.770	0.46950	0.02105
SCRUB-2	0.750	0.50800	0.02454	1.190	0.47850	0.02205	1.760	0.47620	0.02174
SCRUB-5	0.780	0.57920	0.09183	1.210	0.51820	0.04179	1.760	0.50460	0.03636
ORCH-6	0.780	0.57320	0.05722	1.200	0.51750	0.02789	1.750	0.49830	0.03483
HWOOD-7	0.750	0.56830	0.10220	1.250	0.54060	0.08673	1.820	0.52850	0.07381
HWOOD-11	0.770	0.47980	0.02967	1.230	0.45610	0.01834	1.790	0.44820	0.01898
ORCH-12	0.830	0.79180	0.16722	1.150	0.65960	0.08185	1.720	0.57790	0.05277
HWOOD-13	0.760	0.50510	0.02230	1.220	0.47570	0.02173	1.780	0.46800	0.02371
ORCH-15	0.790	0.63220	0.10630	1.180	0.55600	0.04845	1.750	0.53400	0.03164
SUBUR-23	0.810	1.53480	0.74526	1.250	1.18060	0.49131	1.770	1.07060	0.42900
SUBUR-24	0.780	1.35870	0.49905	1.240	1.10530	0.37236	1.760	1.02270	0.32457
RIVER-27	0.740	0.48890	0.02050	1.250	0.47960	0.02304	1.860	0.47730	0.02453
RIVER-29	0.760	1.41270	0.57497	1.190	1.04550	0.34248	1.750	0.93780	0.28417
URBAN-30	0.840	1.23950	0.52604	1.140	1.02010	0.36560	1.700	0.86390	0.26551
URBAN-31	0.760	1.46900	0.65230	1.220	1.06650	0.35998	1.770	0.84560	0.19289
SUBUR-32	0.780	1.44380	0.59282	1.190	1.09760	0.36844	1.760	0.95540	0.25700
RIVER-33	0.750	0.49060	0.02152	1.250	0.47470	0.01548	1.810	0.46850	0.01762
RIVER-34	0.760	1.07110	0.27001	1.260	0.97440	0.19417	1.840	0.93670	0.15649
ROAD-39	0.850	1.17630	0.49559	1.090	0.77870	0.25230	1.610	0.62400	0.15932
SWAMP-42	0.740	1.32990	0.58864	1.250	1.10810	0.23455	1.840	1.05750	0.21009
ROAD-44	0.760	1.12720	0.41340	1.240	0.89960	0.25655	1.780	0.84030	0.21404
RAILY-48	0.760	1.15140	0.35395	1.190	0.86740	0.19620	1.720	0.80040	0.14635
SWAMP-50	0.750	1.06200	0.28341	1.250	0.90770	0.20131	1.820	0.87740	0.18916
RAILY-51	0.770	1.34760	0.48332	1.220	1.11680	0.31642	1.770	1.03310	0.26699
MARSH-52	0.790	0.75370	0.17105	1.230	0.67390	0.12507	1.780	0.63810	0.10083
ROAD-53	0.770	0.81510	0.23901	1.170	0.66270	0.12269	1.720	0.60430	0.07804
SWAMP-54	0.730	0.94800	0.20312	1.240	0.81950	0.12506	1.830	0.79790	0.11468
LAKE-57	0.780	0.47620	0.03908	1.220	0.45980	0.03388	1.780	0.45180	0.01497
LAKE-58	0.720	0.47830	0.02633	1.210	0.45390	0.02105	1.790	0.44880	0.01779
LAKE-59	0.730	0.51300	0.05188	1.240	0.50680	0.04801	1.830	0.49850	0.04892
MARSH-60	0.770	0.66790	0.10930	1.220	0.60580	0.06980	1.770	0.58110	0.05199
RAILY-61	0.760	1.24380	0.39736	1.250	0.94530	0.24392	1.760	0.82320	0.18491
MARSH-65	0.760	0.61060	0.07601	1.250	0.56290	0.03457	1.810	0.55070	0.03379

TABLE II  
PERFORMANCE OF HADAMARD TRANSFORM METHOD

SCENE NO.	BITPEL	RMS ERROR	CORREL ERROR	BITPEL	RMS ERROR	CORREL ERROR	BITPEL	RMS ERROR	CORREL ERROR
SCRUB-1	0.810	0.56160	0.09990	1.100	0.54840	0.09540	1.680	0.54400	0.09194
SCRUB-2	0.800	0.53470	0.05300	1.140	0.52220	0.04654	1.720	0.52030	0.04256
SCRUB-5	0.780	0.61080	0.10656	1.170	0.56810	0.08022	1.740	0.55730	0.06644
ORCH-6	0.810	0.64230	0.09918	1.190	0.59990	0.07640	1.730	0.59500	0.07346
HWOOD-7	0.810	0.63140	0.21178	1.190	0.61340	0.20786	1.690	0.60700	0.20722
HWOOD-11	0.760	0.50220	0.04837	1.190	0.48990	0.04006	1.750	0.48790	0.04362
ORCH-12	0.840	0.93030	0.23052	1.140	0.82340	0.18257	1.710	0.79960	0.16767
HWOOD-13	0.760	0.55950	0.08431	1.190	0.53760	0.06639	1.760	0.53630	0.06787
ORCH-15	0.820	0.79950	0.20435	1.160	0.74640	0.19300	1.700	0.72850	0.19882
SUBUR-23	0.860	1.87330	0.94809	1.210	1.68170	0.90045	1.710	1.60340	0.90312
SUBUR-24	0.860	1.71560	0.67609	1.210	1.59080	0.65590	1.710	1.54380	0.64462
RIVER-27	0.760	0.51060	0.04801	1.250	0.50340	0.04082	1.840	0.50340	0.04624
URBAN-29	0.880	1.71490	0.80110	1.170	1.56230	0.74604	1.660	1.48640	0.77019
URBAN-30	0.840	1.45230	0.64399	1.190	1.31920	0.57282	1.680	1.22250	0.52173
URBAN-31	0.860	1.64200	0.79316	1.210	1.45440	0.70864	1.670	1.36730	0.67631
SUBUR-32	0.860	1.86310	0.84345	1.190	1.65670	0.72365	1.710	1.52490	0.66450
RIVER-33	0.760	0.50170	0.05647	1.230	0.49340	0.04931	1.790	0.49140	0.04732
RIVER-34	0.780	1.19130	0.32112	1.240	1.09560	0.25959	1.830	1.06700	0.24623
ROAD-39	0.880	1.32360	0.67885	1.110	1.14390	0.57263	1.510	1.10090	0.56021
SWAMP-42	0.760	1.46650	0.43625	1.250	1.30310	0.32803	1.830	1.27220	0.30754
ROAD-44	0.760	1.26560	0.47135	1.190	1.08470	0.36784	1.760	1.03350	0.34138
RAILY-48	0.790	1.34400	0.42699	1.160	1.16500	0.35990	1.700	1.10980	0.33493
SWAMP-50	0.760	1.17820	0.34610	1.240	1.07760	0.28819	1.810	1.05790	0.27906
RAILY-51	0.770	1.55370	0.54539	1.210	1.33680	0.40292	1.780	1.28600	0.35764
MARSH-52	0.760	0.80930	0.19296	1.220	0.74430	0.16697	1.770	0.72200	0.15700
ROAD-53	0.840	0.99150	0.33573	1.120	0.91070	0.31219	1.630	0.88730	0.31467
SWAMP-54	0.780	1.03370	0.24356	1.240	0.94770	0.19531	1.820	0.93270	0.18578
LAKE-57	0.790	0.50320	0.07916	1.200	0.49580	0.07000	1.760	0.48890	0.05923
LAKE-58	0.740	0.48130	0.04908	1.210	0.46900	0.03867	1.770	0.46900	0.04202
LAKE-59	0.800	0.56600	0.13455	1.220	0.55680	0.13162	1.720	0.55640	0.13819
MARSH-60	0.770	0.70800	0.14709	1.170	0.65160	0.11035	1.750	0.63260	0.09200
RAILY-61	0.770	1.55390	0.53819	1.240	1.37640	0.42690	1.800	1.29430	0.35727
MARSH-65	0.750	0.64570	0.10101	1.230	0.60680	0.06726	1.790	0.60310	0.06839

TABLE III  
PERFORMANCE OF DISCRETE LINEAR TRANSFORM METHOD

SCENE NO	BITPEL	RMS ERROR	CORREL ERROR	BITPEL	RMS ERROR	CORREL ERROR	BITPEL	RMS ERROR	CORREL ERROR
SCRUB- 1	0.800	0.52340	0.83311	1.120	0.49850	0.82376	1.680	0.47060	0.82735
SCRUB- 2	0.790	0.52150	0.82326	1.160	0.49560	0.82105	1.730	0.48290	0.82503
SCRUB- 3	0.770	0.59050	0.89564	1.170	0.51730	0.84160	1.750	0.49580	0.82587
ORCH - 6	0.800	0.52750	0.84896	1.190	0.52060	0.82904	1.740	0.50510	0.83258
HWOOD- 7	0.810	0.56750	0.10345	1.200	0.52080	0.89136	1.680	0.51090	0.89669
HWOOD- 11	0.750	0.49210	0.82800	1.190	0.45710	0.82208	1.760	0.44390	0.81730
ORCH - 12	0.830	0.79100	0.16408	1.140	0.65440	0.86618	1.720	0.60190	0.86243
HWOOD- 13	0.780	0.51990	0.82422	1.200	0.47160	0.81777	1.770	0.46170	0.81765
ORCH - 15	0.830	0.66090	0.11492	1.170	0.58750	0.10048	1.720	0.54130	0.10665
SUBUR- 23	0.850	1.58810	0.70653	1.210	1.28540	0.51372	1.710	1.16520	0.50161
SURUR- 24	0.840	1.38990	0.45463	1.210	1.19250	0.39609	1.720	1.11510	0.39652
RIVER- 27	0.710	0.49830	0.82611	1.240	0.48130	0.81938	1.840	0.48010	0.81974
URBAN- 29	0.870	1.49090	0.61718	1.170	1.25070	0.46698	1.680	1.14160	0.46163
URBAN- 30	0.850	1.26220	0.51005	1.170	1.05730	0.39863	1.700	0.92810	0.34054
URBAN- 31	0.850	1.39540	0.60610	1.210	1.15500	0.48689	1.680	1.00220	0.42840
SUBUR- 32	0.850	1.52510	0.63678	1.190	1.23990	0.49636	1.710	1.00890	0.41163
RIVER- 33	0.760	0.49360	0.83070	1.240	0.47240	0.82222	1.800	0.46930	0.82464
RIVER- 34	0.760	1.11750	0.26316	1.250	0.98960	0.18296	1.830	0.96200	0.18839
ROAD - 39	0.880	1.13110	0.53521	1.120	0.87640	0.38920	1.520	0.78920	0.34658
SWAMP- 42	0.780	1.35070	0.35265	1.250	1.16780	0.23012	1.830	1.13060	0.20919
ROAD - 44	0.770	1.17370	0.42703	1.230	0.93210	0.28365	1.780	0.84720	0.24864
RAILY- 48	0.780	1.20130	0.36702	1.170	0.94420	0.24479	1.710	0.85170	0.21470
SWAMP- 50	0.750	1.11380	0.30618	1.250	0.95870	0.23928	1.830	0.93820	0.24435
RAILY- 51	0.770	1.42540	0.46496	1.230	1.17950	0.33490	1.790	1.12170	0.30789
MARSH- 52	0.790	0.76210	0.15921	1.240	0.68910	0.12225	1.780	0.64610	0.09816
ROAD - 53	0.860	0.87530	0.25654	1.130	0.76210	0.20749	1.630	0.71570	0.20696
SWAMP- 54	0.780	0.98240	0.20259	1.250	0.87950	0.13907	1.830	0.85870	0.12790
LAKE - 57	0.790	0.48310	0.84693	1.200	0.46610	0.83779	1.770	0.45340	0.82174
LAKE - 58	0.740	0.47850	0.82367	1.190	0.45710	0.81775	1.770	0.45200	0.81626
LAKE - 59	0.810	0.51730	0.86364	1.230	0.50000	0.86120	1.720	0.49780	0.86003
MARSH- 60	0.760	0.67980	0.11069	1.180	0.60110	0.87548	1.760	0.57850	0.86496
RAILY- 61	0.760	1.36090	0.47117	1.250	1.09050	0.31726	1.810	0.97060	0.27963
MARSH- 65	0.780	0.60620	0.87442	1.250	0.57000	0.85484	1.800	0.56030	0.84834

The third measure of performance used was a direct comparison of the original and reconstructed integer images. For this purpose, the images were equal probability quantized into 13 levels and digitally printed out; the error pictures were also printed out for comparison.

*Results of Transform Data Compression Experiments*

The set of 33 images was processed using the Karhunen-Loève, Hadamard, and DLB transform compression techniques. The window size used was 4 x 4. For all three methods, we found that the projections of the data onto 4 of the 16 basis vectors accounted for most of the image "energy"; the variances of the remaining 12 transform coefficients were very small compared to the first 4, and hence the bit-assignment algorithm ignored these coefficients. The 4 basis vectors for the Karhunen-Loève transform were the 4 eigenvectors of the covariance matrix of the image (which was computed for each image) corresponding to the 4 largest eigenvalues. For the Hadamard and DLB transforms, the basis vectors with sequency (0,0), (1,0), (0,1), and (1,1) were used.

The original images had gray-tone values in the range 1-64 (6 bits) and the performances of the compression techniques were evaluated at bit rates of 0.8, 1.2, and 1.75 b/picture element. The corresponding bit compression rates were 7.5, 5, and 3.4, respectively. The rms errors, correlated rms errors, and bit rates for the Karhunen-Loève, Hadamard, and DLB transform coding techniques are shown in Table I-Table III for the 33 images we processed. Plots of correlated rms error versus bit rate are shown plotted for four scenes in Fig. 1(a)-(d). The average values of the rms error versus bit rate for the 33 images are shown in Fig. 2,

and the average values of the correlated rms error versus bit rate for the 33 images are shown in Fig. 3.

Comparison of the performance of the three compression methods in terms of rms error reveals the following.

- 1) As one might expect, the Karhunen-Loève transform technique yields the minimum rms error.
- 2) The rms errors for images compressed by the DLB transform are comparable to the rms errors for the Karhunen-Loève transform and are considerably lower than the rms error for the Hadamard transform coding.

In terms of correlated rms error, the average performance of the DLB transform is quite close to the performance of the Karhunen-Loève transform at high compression ratios. On some images the DLB transform performed better than the Karhunen-Loève transform at high compression ratios. Both the DLB and Karhunen-Loève transforms performed much better than the Hadamard transform at all compression ratios.

For four scenes shown in Fig. 4, the reconstructed pictures and error pictures are shown in Figs. 5-8. The pictures in the top rows of Figs. 5-8 were produced by the Karhunen-Loève transform, the middle row by the DLB transform, and the bottom row by the Hadamard transform. The error pictures show the absolute value of the error between the original and reconstructed pictures.

The error pictures and reconstructed pictures for the orchard scene shown in Fig. 5 reveal that there are no significant differences in the pictures produced by the three methods. The Hadamard transform has a slightly larger rms error and correlated rms error.

The error pictures and reconstructed pictures for the suburban scene shown in Fig. 6 reveal significant differences in the pictures produced by the three methods. The error

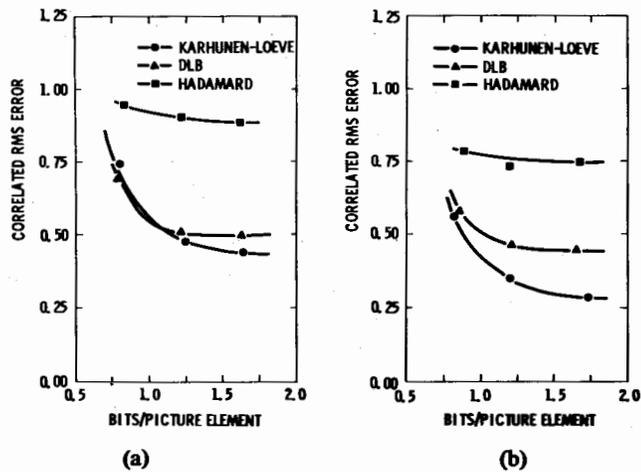


Fig. 1. Bit rate versus correlated error for four scenes. (a) Scene 23, suburban. (b) Scene 29, urban. (c) Scene 12, orchard. (d) Scene 61, railyard.

pictures for the Karhunen-Loève and DLB transforms are nearly identical at low bit rates. The error pictures for the Hadamard transform show the presence of larger errors; also, these error pictures have a higher spatial correlation, as evidenced by the well-defined boundaries. The high spatial correlation is also reflected in higher values for the correlated rms error, showing the usefulness of the correlated rms error as a measure of the magnitude and spatial distribution of the errors. In all of the error pictures, the boundaries show up clearly at high bit rates due to the fact that the average error decreases at higher bit rates and hence the lighter background due to smaller errors brings out the boundaries better. The error pictures for the urban and railyard scenes show the same trends seen in the suburban scene. Namely, the performance of the DLB transform is comparable to that of the Karhunen-Loève transform and is much better than that of the Hadamard transform.

Apart from the performance indexes we have used, another way of comparing the performance of unitary transforms is to judge by how uncorrelated the transform coefficients are. This correlation between transform coefficients is reflected in the variance of the transform

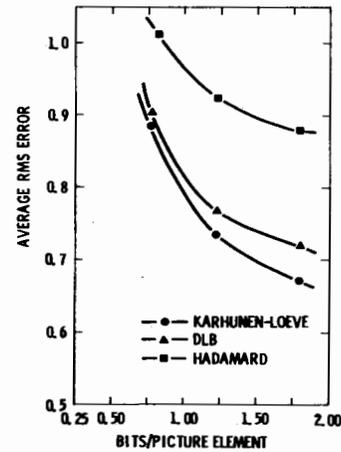


Fig. 2. Bit rate versus average rms error for 33 scenes.

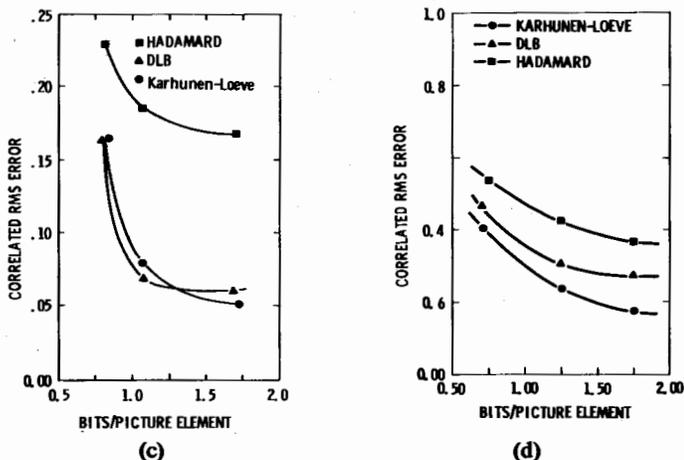


Fig. 3. Bit rate versus average correlated rms error for 33 scenes.

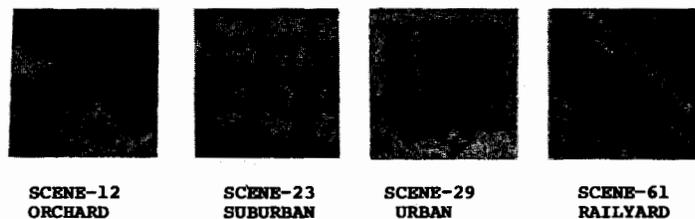


Fig. 4. Original pictures.

coefficients. We found that for the 33 scenes we processed, the variances of the transform coefficients produced by the DLB transform were very close to the variances of the transform coefficients produced by the Karhunen-Loève transform.

## VI. CONCLUDING REMARKS

We have presented a new discrete linear transform method of data compression which we use in conjunction with a DPCM procedure for removing some of the redundancy in image data. We have compared the performance of the new method with the performance of Karhunen-Loève and Hadamard transform methods on a

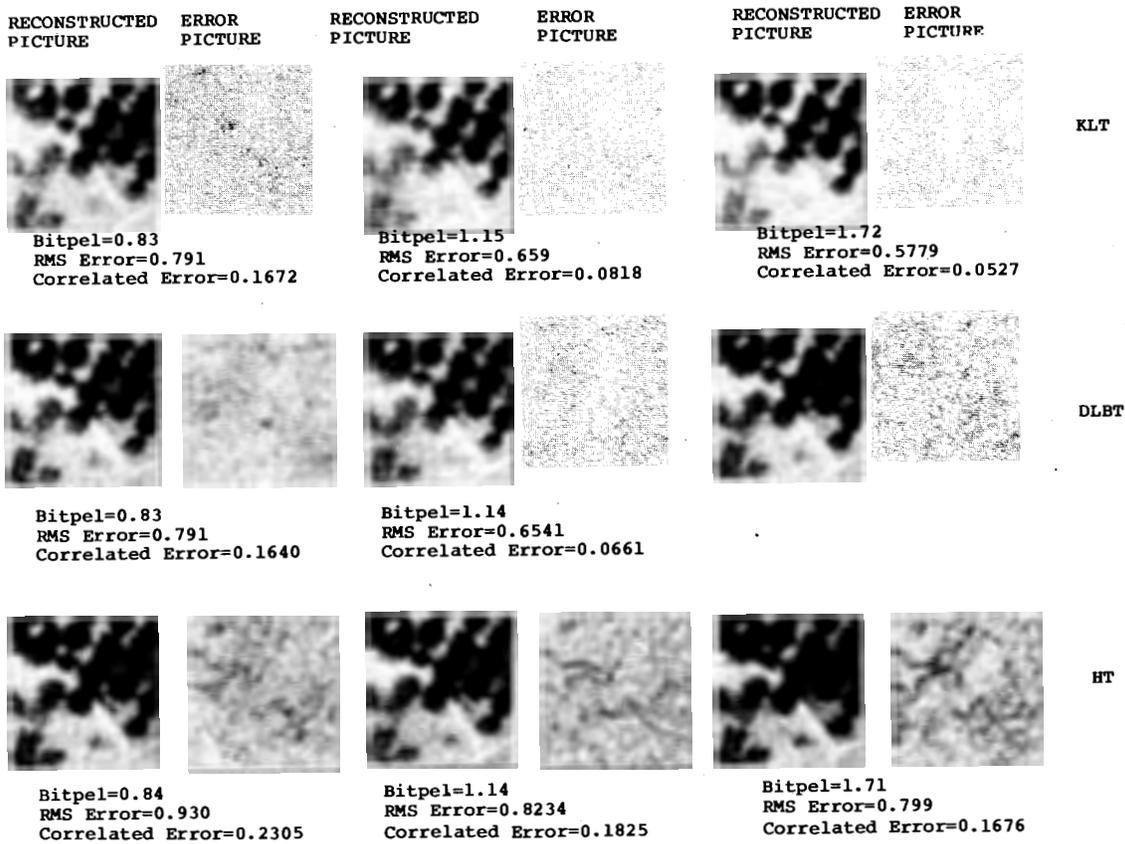


Fig. 5. Reconstructed and error pictures for scene 12, orchard.

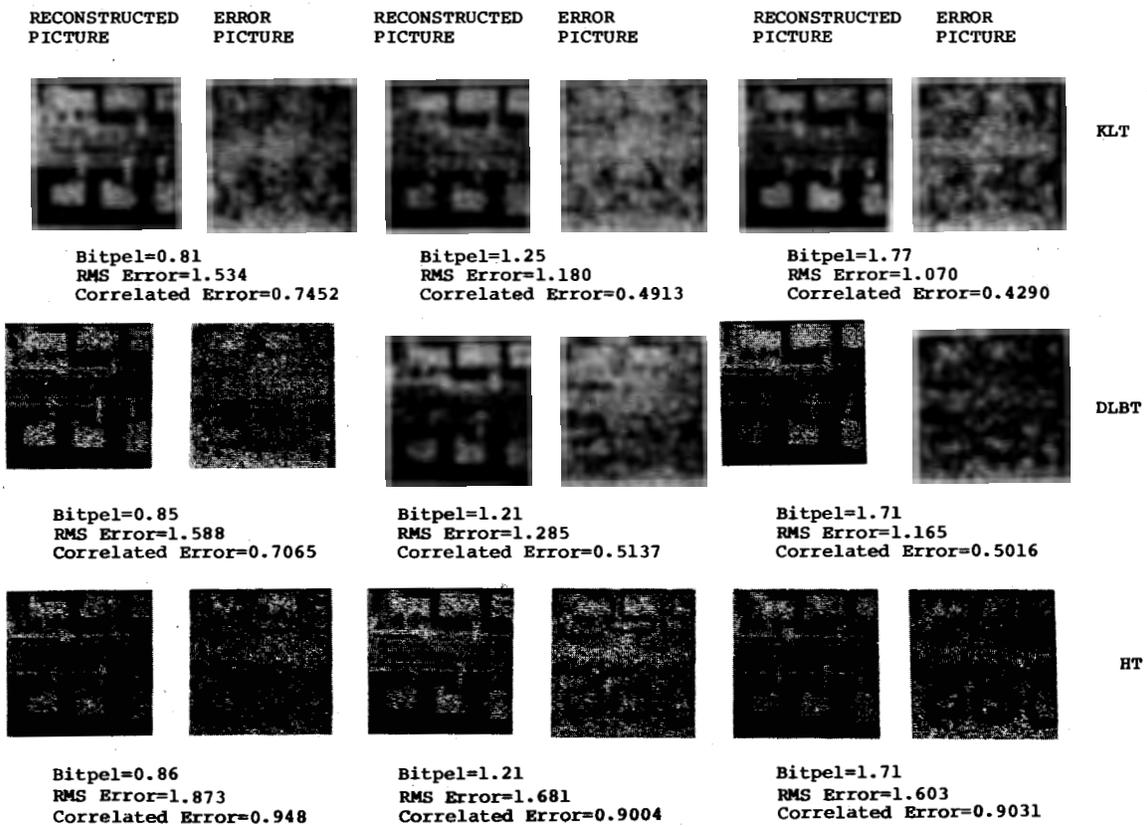


Fig. 6. Reconstructed and error pictures for scene 23, suburban.

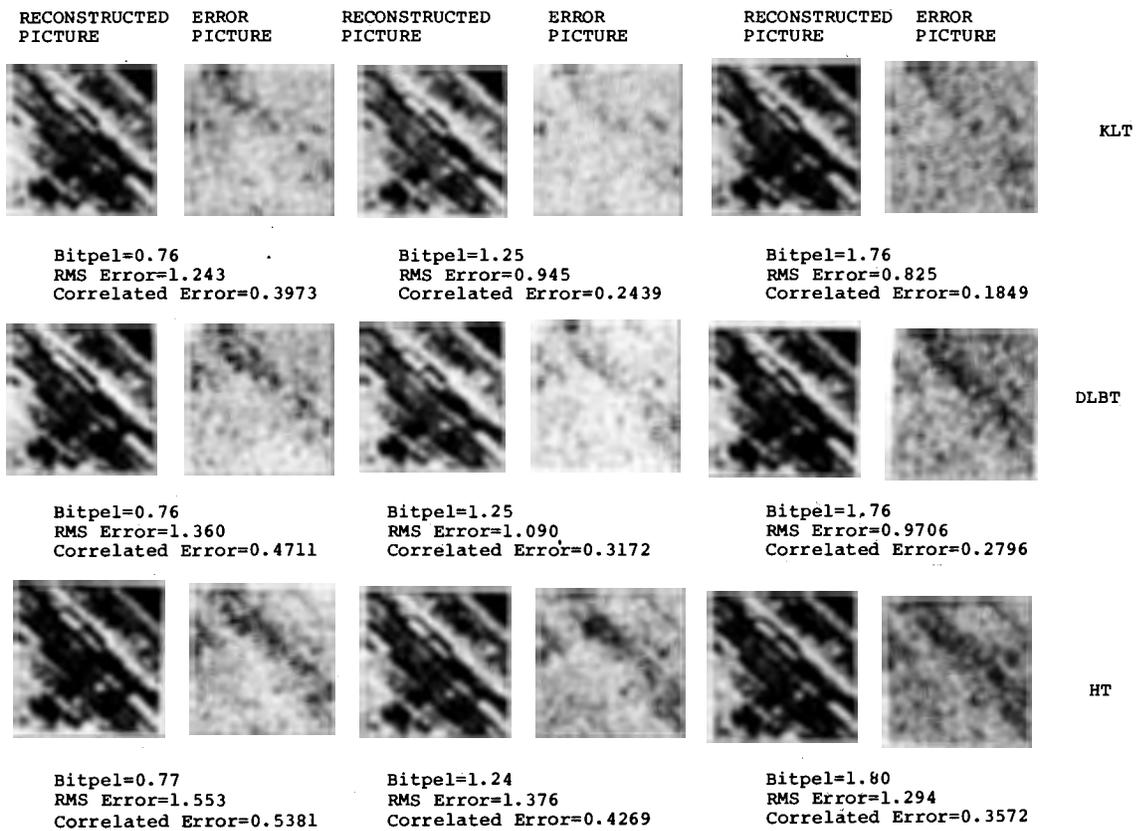


Fig. 7. Reconstructed and error pictures for scene 61, railyard.

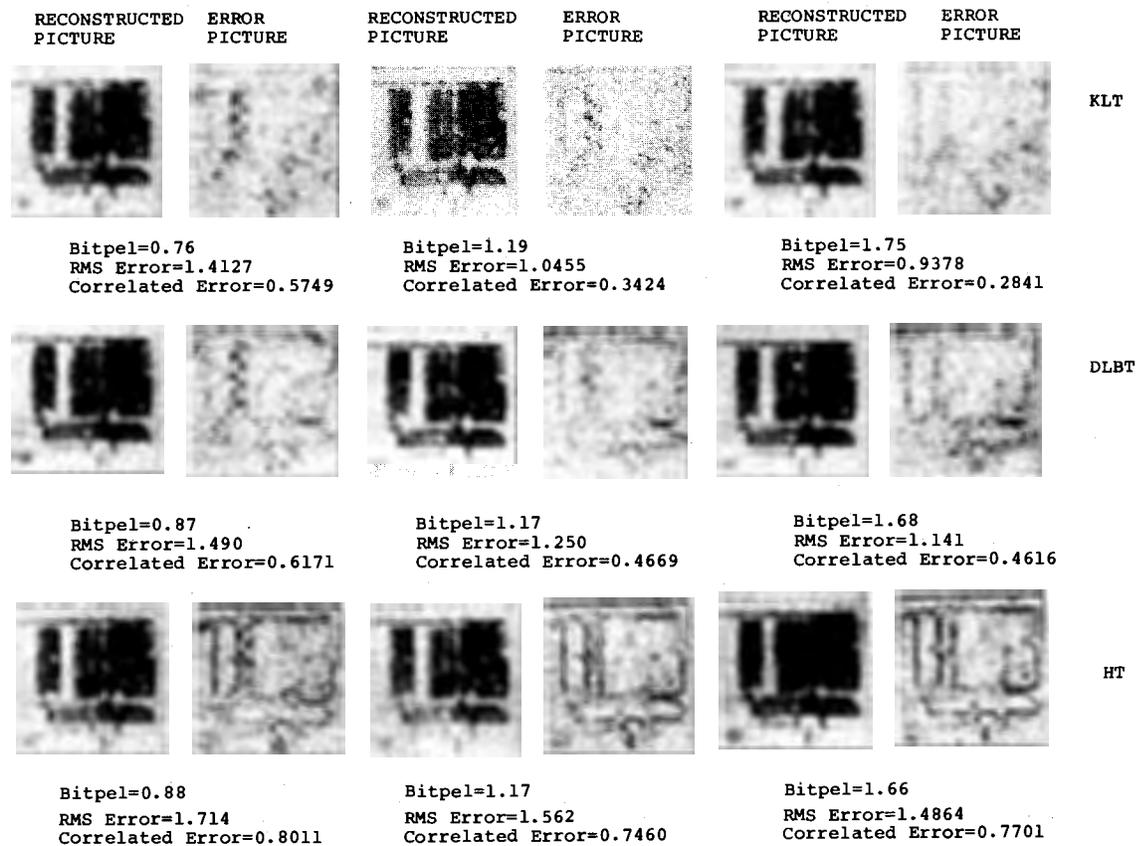


Fig. 8. Reconstructed and error pictures for scene 29, urban.

set of 33 images. Based on the comparative performance on 33 images, we conclude that the new discrete linear transform offers a good tradeoff between the ease of implementation of the Hadamard transform and superior performance to the Karhunen-Loève transform.

Other fast transforms such as the slant transform [4] or discrete cosine transform [20] also seem to perform well compared to the Hadamard transform, while retaining some of the computational advantages of the Hadamard transform. Further theoretical and experimental work is needed to compare the performances of DLB, Hadamard, and discrete cosine transforms.

APPENDIX

*Lemma:* Let  $\{X_1, X_2, \dots, X_{N_1}\}$  be an orthogonal basis for  $R^{N_1}$  and  $\{Y_1, Y_2, \dots, Y_{N_2}\}$  be an orthogonal basis for  $R^{N_2}$ . Then  $\{X_i Y_j \mid i = 1, 2, \dots, N_1; j = 1, 2, \dots, N_2\}$  is an orthogonal basis for  $R^{N_1 N_2}$ .

*Proof:* We will use the notation  $X_{ik}$  to denote the  $k$ th component of the vector  $X_i$  and we will first prove orthogonality of the basis and then prove the linear independence. Consider

$$\begin{aligned} (X_i Y_j)^T (X_n Y_m) &= \sum_{k=1}^{N_1} \sum_{l=1}^{N_2} X_{ik} Y_{jl} X_{nk} Y_{ml} \\ &= \left( \sum_{k=1}^{N_1} X_{ik} X_{nk} \right) \left( \sum_{l=1}^{N_2} Y_{jl} Y_{ml} \right). \end{aligned}$$

Since both  $\{X_i\}$  and  $\{Y_j\}$  are orthogonal basis,  $X_i$  is orthogonal to  $X_n$  and  $Y_j$  is orthogonal to  $Y_m$  so that  $(X_i Y_j)^T (X_n Y_m) = 0$  and hence  $X_i Y_j$  is orthogonal to  $X_n Y_m$ . The proof for linear independence is as follows. Suppose that  $\{c_{ij} \mid i = 1, 2, \dots, N_1; j = 1, 2, \dots, N_2\}$  is a set of  $N_1 N_2$  constants not all zero and that

$$\sum_{i=1}^{N_1} \sum_{j=1}^{N_2} c_{ij} X_i Y_j = 0$$

thereby implying that  $\{X_i Y_j\}$  is linearly dependent. We will show that this implies that  $\{X_i\}$  is not linearly independent, a contradiction. We begin with

$$\begin{aligned} 0 &= \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} c_{ij} X_i Y_j \\ &= \sum_{i=1}^{N_1} X_i \left( \sum_{j=1}^{N_2} c_{ij} Y_j \right). \end{aligned}$$

However, since the  $\{Y_j\}$  are linearly independent and since the  $\{c_{ij}\}$  are not all zero, there must be some constant  $\alpha$  such that  $\sum_{j=1}^{N_2} c_{aj} Y_j \neq 0$ .

Let

$$\sum_j c_{aj} Y_j = (b_{i1}, b_{i2}, \dots, b_{iN_2})^T, \quad i = 1, 2, \dots, N_1.$$

Then

$$0 = \sum_{i=1}^{N_1} \left( \sum_{j=1}^{N_2} c_{ij} X_i Y_j \right)$$

implies

$$\sum_{i=1}^{N_1} X_{ik} b_{im} = 0, \quad k = 1, \dots, N_1; \quad m = 1, \dots, N_2$$

where  $X_{ik}$  is the  $k$ th component of  $X_i$ . However,

$$\sum_{j=1}^{N_2} c_{aj} Y_j \neq 0$$

implies that for some  $\alpha$  not all components of  $(b_{\alpha 1}, b_{\alpha 2}, \dots, b_{\alpha N_2})^T$  are zero. Hence there exist some  $b_{\alpha\beta} \neq 0$ . Finally, since  $b_{\alpha\beta} \neq 0$ , and

$$\sum_{i=1}^{N_1} X_{i\beta} b_{i\beta} = 0$$

implies that the  $\{X_i\}$  are linearly independent, we have obtained the contradiction and the proof.

ACKNOWLEDGMENT

The authors wish to thank J. Young and D. Goel of the Center for Research, Inc., University of Kansas, for their help in developing computer programs for implementing the data compression schemes discussed in this paper.

REFERENCES

- [1] T. S. Huang and J. W. Woods, "Picture bandwidth compression by block quantization," presented at the 1969 Int. Symp. Information Theory, Ellenville, N.Y., Jan. 28-31.
- [2] —, "Picture bandwidth compression by linear transformation and block quantization," presented at the 9th Symp. Picture Bandwidth Compression, M.I.T., Cambridge, Mass.
- [3] W. K. Pratt, J. Kane, and H. C. Andrews, "Hadamard transform image coding," *Proc. IEEE*, vol. 57, pp. 58-68, Jan. 1969.
- [4] W. K. Pratt, L. R. Welch, and W. Chen, "Slant transforms for image coding," in *Proc. 1972 Applications of Walsh Functions Symp.*
- [5] A. Habibi and P. A. Wintz, "Image coding by linear transformations and block quantization," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 50-62, Feb. 1971.
- [6] M. Tasto and P. A. Wintz, "Image coding by adaptive block quantization," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 957-971, Dec. 1971.
- [7] A. Habibi, "Comparison of  $n$ th-order DPCM encoder with linear transformation and block quantization techniques," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 948-956, Dec. 1971.
- [8] P. A. Wintz, "Transform picture coding," *Proc. IEEE*, vol. 60, pp. 809-820, July 1972.
- [9] J. Max, "Quantization for minimum distortion," *IRE Trans. Inform. Theory*, vol. IT-6, pp. 7-12, Mar. 1960.
- [10] R. M. Haralick, K. Shanmugam, J. Young, and D. Goel, "A comparative study of transform data compression techniques for digital imagery," in *Proc. Nat. Electronics Conf.*, vol. 27, pp. 89-94, 1972.
- [11] A. Habibi, "Unitary transformations and DPCM coding of pictorial data," in *Proc. Image Coding Symp.*, 1973.
- [12] —, "Unitary transformations and DPCM coding of pictorial data," in *Proc. Applications of Walsh Functions Symp.*, 1973.
- [13] M. Arisana and F. Kanaya, "Image coding by differential Hadamard transform," in *Proc. Applications of Walsh Functions Symp.*, 1973.
- [14] M. Tasto and P. A. Wintz, "A bound on the rate-distortion function and application to images," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 150-159, Jan. 1972.
- [15] L. D. Davison, "Rate distortion theory and applications," *Proc. IEEE*, vol. 60, pp. 800-809, July 1972.
- [16] K. Shanmugam and R. M. Haralick, "A computationally simple procedure for imagery data compression by the Karhunen-Loève method," *IEEE Trans. Syst., Man, Cybern.* (Corresp.), vol. SMC-3, pp. 202-204, Mar. 1973.
- [17] W. K. Pratt and H. C. Andrews, "Transform image coding," Univ. Southern California, Los Angeles, USCEE Rep. 387, Mar. 1970.
- [18] M. Tasto and P. A. Wintz, "Picture bandwidth compression by adaptive block quantization," School Elec. Eng. Tech. Dep., Purdue Univ., Lafayette, Ind., Tech. Rep. EE-70-14, June 1970.
- [19] C. E. Bell, "Mutual information and maximal correlation as measures of dependence," *Ann. Math. Statist.*, vol. 43, pp. 587-595, 1962.
- [20] N. Ahmed, T. Natarajan, and K. R. Rao, "On image processing and a discrete cosine transform," IEEE Computer Society Repository, Paper R73-96.