

4 Transform Coding

As introduced in the previous chapter, differential coding achieves high coding efficiency by utilizing the correlation between pixels existing in image frames. Transform coding (TC), the focus of this chapter, is another efficient coding scheme based on utilization of interpixel correlation. As we will see in Chapter 7, TC has become a fundamental technique recommended by the international still image coding standard, JPEG. Moreover, TC has been found to be efficient in coding prediction error in motion-compensated predictive coding. As a result, TC was also adopted by the international video coding standards such as H.261, H.263, and MPEG 1, 2, and 4. This will be discussed in Section IV.

4.1 INTRODUCTION

Recall the block diagram of source encoders shown in Figure 2.3. There are three components in a source encoder: transformation, quantization, and codeword assignment. It is the transformation component that decides which format of input source is quantized and encoded. In DPCM, for instance, the difference between an original signal and a predicted version of the original signal is quantized and encoded. As long as the prediction error is small enough, i.e., the prediction resembles the original signal well (by using correlation between pixels), differential coding is efficient.

In transform coding, the main idea is that if the transformed version of a signal is less correlated compared with the original signal, then quantizing and encoding the transformed signal may lead to data compression. At the receiver, the encoded data are decoded and transformed back to reconstruct the signal. Therefore, in transform coding, the transformation component illustrated in Figure 2.3 is a transform. Quantization and codeword assignment are carried out with respect to the transformed signal, or, in other words, carried out in the transform domain.

We begin with the Hotelling transform, using it as an example of how a transform may decorrelate a signal in the transform domain.

4.1.1 HOTELLING TRANSFORM

Consider an N -dimensional vector \vec{z}_s . The ensemble of such vectors, $\{\vec{z}_s\}$ $s \in I$, where I represents the set of all vector indexes, can be modeled by a random vector \vec{z} with each of its component z_i , $i = 1, 2, \dots, N$ as a random variable. That is,

$$\vec{z} = (z_1, z_2, \dots, z_N)^T \quad (4.1)$$

where T stands for the operator of matrix transposition. The mean vector of the population, $m_{\vec{z}}$, is defined as

$$m_{\vec{z}} = E[\vec{z}] = (m_1, m_2, \dots, m_N)^T \quad (4.2)$$

where E stands for the expectation operator. Note that $m_{\vec{z}}$ is an N -dimensional vector with the i th component, m_i , being the expectation value of the i th random variable component in \vec{z} .

$$m_i = E[z_i] \quad i = 1, 2, \dots, N \quad (4.3)$$

The covariance matrix of the population, denoted by $C_{\bar{z}}$, is equal to

$$C_{\bar{z}} = E[(\bar{z} - m_{\bar{z}})(\bar{z} - m_{\bar{z}})^T]. \quad (4.4)$$

Note that the product inside the E operator is referred to as the *outer product* of the vector $(\bar{z} - m_{\bar{z}})$. Denote an entry at the i th row and j th column in the covariance matrix by $c_{i,j}$. From Equation 4.4, it can be seen that $c_{i,j}$ is the covariance between the i th and j th components of the random vector \bar{z} . That is,

$$c_{i,j} = E[(z_i - m_i)(z_j - m_j)] = \text{Cov}(z_i, z_j). \quad (4.5)$$

On the main diagonal of the covariance matrix $C_{\bar{z}}$, the element $c_{i,i}$ is the variance of the i th component of \bar{z} , z_i . Obviously, the covariance matrix $C_{\bar{z}}$ is a real and symmetric matrix. It is real because of the definition of random variables. It is symmetric because $\text{Cov}(z_i, z_j) = \text{Cov}(z_j, z_i)$. According to the theory of linear algebra, it is always possible to find a set of N orthonormal eigenvectors of the matrix $C_{\bar{z}}$, with which we can convert the real symmetric matrix $C_{\bar{z}}$ into a fully ranked diagonal matrix. This statement can be found in texts of linear algebra, e.g., in (Strang, 1998).

Denote the set of N orthonormal eigenvectors and their corresponding eigenvalues of the covariance matrix $C_{\bar{z}}$ by \bar{e}_i and λ_i , $i = 1, 2, \dots, N$, respectively. Note that eigenvectors are column vectors. Form a matrix Φ such that its rows comprise the N transposed eigenvectors. That is,

$$\Phi = (\bar{e}_1, \bar{e}_2, \dots, \bar{e}_N)^T. \quad (4.6)$$

Now, consider the following transformation:

$$\bar{y} = \Phi(\bar{z} - m_{\bar{z}}) \quad (4.7)$$

It is easy to verify that the transformed random vector \bar{y} has the following two characteristics:

1. The mean vector, $m_{\bar{y}}$, is a zero vector. That is,

$$m_{\bar{y}} = 0. \quad (4.8)$$

2. The covariance matrix of the transformed random vector $C_{\bar{y}}$ is

$$C_{\bar{y}} = \Phi C_{\bar{z}} \Phi^T = \begin{bmatrix} \lambda_1 & & & & 0 \\ & \lambda_2 & & & \\ & & \ddots & & \\ & & & \ddots & \\ 0 & & & & \lambda_n \end{bmatrix}. \quad (4.9)$$

This transform is called the Hotelling transform (Hotelling, 1933), or eigenvector transform (Tasto, 1971; Wintz, 1972).

The inverse Hotelling transform is defined as

$$\bar{z} = \Phi^{-1} \bar{y} + m_{\bar{z}}, \quad (4.10)$$

where Φ^{-1} is the inverse matrix of Φ . It is easy to see from its formation discussed above that the matrix Φ is orthogonal. Therefore, we have $\Phi^T = \Phi^{-1}$. Hence, the inverse Hotelling transform can be expressed as

$$\bar{z} = \Phi^T \bar{y} + m_{\bar{z}} \tag{4.11}$$

Note that in implementing the Hotelling transform, the mean vector $m_{\bar{z}}$ and the covariance matrix $C_{\bar{z}}$ can be calculated approximately by using a given set of K sample vectors (Gonzalez and Woods, 1992).

$$m_{\bar{z}} = \frac{1}{K} \sum_{s=1}^K \bar{z}_s \tag{4.12}$$

$$C_{\bar{z}} = \frac{1}{K} \sum_{s=1}^K \bar{z}_s \bar{z}_s^T - m_{\bar{z}} m_{\bar{z}}^T \tag{4.13}$$

The analogous transform for continuous data was devised by Karhunen and Loeve (Karhunen, 1947; Loeve, 1948). Alternatively, the Hotelling transform can be viewed as the discrete version of the Karhunen-Loeve transform (KLT). We observe that the covariance matrix $C_{\bar{y}}$ is a diagonal matrix. The elements in the diagonal are the eigenvalues of the covariance matrix $C_{\bar{z}}$. That is, the two covariance matrices have the same eigenvalues and eigenvectors because the two matrices are similar. The fact that zero values are everywhere except along the main diagonal in $C_{\bar{y}}$ indicates that the components of the transformed vector \bar{y} are uncorrelated. That is, the correlation previously existing between the different components of the random vector \bar{z} has been removed in the transformed domain. Therefore, if the input is split into *blocks* and the Hotelling transform is applied blockwise, the coding may be more efficient since the data in the transformed block are uncorrelated. At the receiver, we may produce a replica of the input with an inverse transform. This basic idea behind transform coding will be further illustrated next. Note that transform coding is also referred to as block quantization (Huang, 1963).

4.1.2 STATISTICAL INTERPRETATION

Let's continue our discussion of the 1-D Hotelling transform. Recall that the covariance matrix of the transformed vector \bar{y} , $C_{\bar{y}}$, is a diagonal matrix. The elements in the main diagonal are eigenvalues of the covariance matrix $C_{\bar{z}}$. According to the definition of a covariance matrix, these elements are the variances of the components of vector \bar{y} , denoted by $\sigma_{y,1}^2, \sigma_{y,2}^2, \dots, \sigma_{y,N}^2$. Let us arrange the eigenvalues (variances) in a nonincreasing order. That is, $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_N$. Choose an integer L , and $L < N$. Using the corresponding L eigenvectors, $\vec{e}_1, \vec{e}_2, \dots, \vec{e}_L$, we form a matrix $\bar{\Phi}$ with these L eigenvectors (transposed) as its L rows. Obviously, the matrix $\bar{\Phi}$ is of $L \times N$. Hence, using the matrix $\bar{\Phi}$ in Equation 4.7 we will have the transformed vector \bar{y} of $L \times 1$. That is,

$$\bar{y} = \bar{\Phi}(\bar{z} - m_{\bar{z}}) \tag{4.14}$$

The inverse transform changes accordingly:

$$\bar{z}' = \bar{\Phi}^T \bar{y} + m_{\bar{z}} \tag{4.15}$$

Note that the reconstructed vector \vec{z} , denoted by \vec{z}' , is still an $N \times 1$ column vector. It can be shown (Wintz, 1972) that the mean square reconstruction error between the original vector \vec{z} and the reconstructed vector \vec{z}' is given by

$$MSE_r = \sum_{i=L+1}^N \sigma_{y,i}^2. \quad (4.16)$$

This equation indicates that the mean square reconstruction error equals the sum of variances of the discarded components. Note that although we discuss the reconstruction error here, we have not considered the quantization error and transmission error involved. Equation 4.15 implies that if, in the transformed vector \vec{y} , the first L components have their variances occupy a large percentage of the total variances, the mean square reconstruction error will not be large even though only the first L components are kept, i.e., the $(N - L)$ remaining components in the \vec{y} are discarded. Quantizing and encoding only L components of vector \vec{y} in the transform domain lead to higher coding efficiency. This is the basic idea behind transform coding.

4.1.3 GEOMETRICAL INTERPRETATION

Transforming a set of statistically dependent data into another set of uncorrelated data, then discarding the insignificant transform coefficients (having small variances) illustrated above using the Hotelling transform, can be viewed as a statistical interpretation of transform coding. Here, we give a geometrical interpretation of transform coding. For this purpose, we use 2-D vectors instead of N -D vectors.

Consider a binary image of a car in Figure 4.1(a). Each pixel in the shaded object region corresponds to a 2-D vector with its two components being coordinates z_1 and z_2 , respectively. Hence, the set of all pixels associated with the object forms a population of vectors. We can determine its mean vector and covariance matrix using Equations 4.12 and 4.13, respectively. We can then apply the Hotelling transform by using Equation 4.7. Figure 4.1(b) depicts the same object after the application of the Hotelling transform in the y_1 - y_2 coordinate system. We notice that the origin of the new coordinate system is now located at the centroid of the binary object. Furthermore, the new coordinate system is aligned with the two eigenvectors of the covariance matrix $C_{\vec{z}}$.

As mentioned, the elements along the main diagonal of $C_{\vec{y}}$ (two eigenvalues of the $C_{\vec{y}}$ and $C_{\vec{z}}$) are the two variances of the two components of the \vec{y} population. Since the covariance matrix

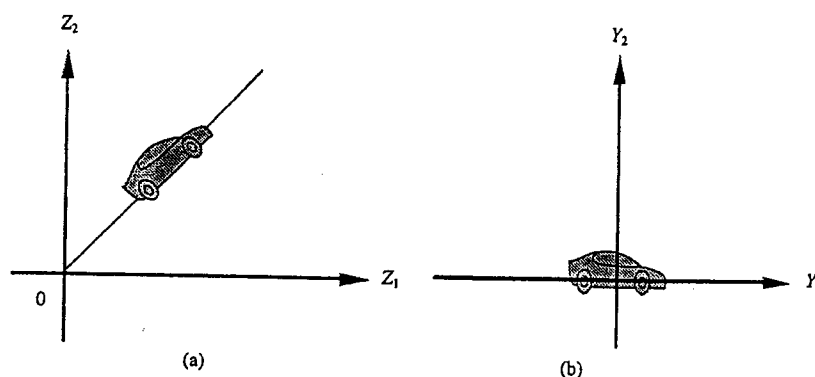


FIGURE 4.1 (a) A binary object in the z_1 - z_2 coordinate system. (b) After the Hotelling transform, the object is aligned with its principal axes.

tor. It can be vector \vec{z} and

(4.16)

of variances here, we have implies that ge percentage ough only the re discarded. ead to higher

ed data, then l above using ing. Here, we ectors instead

object region respectively. tors. We can pectively. We e same object otice that the Furthermore, atrix $C_{\vec{z}}$. of the $C_{\vec{y}}$ and riance matrix

x_i

orm, the object

$C_{\vec{y}}$ is a diagonal matrix, the two components are uncorrelated after the transform. Since one variance (along the y_1 direction) is larger than the other (along the y_2 direction), it is possible for us to achieve higher coding efficiency by ignoring the component associated with the smaller variance without too much sacrifice of the reconstructed image quality.

It is noted that the alignment of the object with the eigenvectors of the covariance matrix is of importance in pattern recognition (Gonzalez and Woods, 1992).

4.1.4 BASIS VECTOR INTERPRETATION

Basis vector expansion is another interpretation of transform coding. For simplicity, in this subsection we assume a zero mean vector. Under this assumption, the Hotelling transform and its inverse transform become

$$\vec{y} = \Phi \vec{z} \tag{4.17}$$

$$\vec{z} = \Phi^T \vec{y} \tag{4.18}$$

Recall that the row vectors in the matrix Φ are the transposed eigenvectors of the covariance matrix $C_{\vec{z}}$. Therefore, Equation 4.18 can be written as

$$\vec{z} = \sum_{i=1}^N y_i \vec{e}_i. \tag{4.19}$$

In the above equation, we can view vector \vec{z} as a linear combination of *basis vectors* \vec{e}_i , $i = 1, 2, \dots, N$. The components of the transformed vector \vec{y} , y_i , $i = 1, 2, \dots, N$ serve as coefficients in the linear combination, or weights in the weighted sum of basis vectors. The coefficient y_i , $i = 1, 2, \dots, N$ can be produced according to Equation 4.17:

$$y_i = \vec{e}_i^T \vec{z}. \tag{4.20}$$

That is, y_i is the *inner product* between vectors \vec{e}_i and \vec{z} . Therefore, the coefficient y_i can be interpreted as the amount of correlation between the basis vector \vec{e}_i and the original signal \vec{z} .

In the Hotelling transform the coefficients y_i , $i = 1, 2, \dots, N$ are uncorrelated. The variance of y_i can be arranged in a nonincreasing order. For $i > L$, the variance of the coefficient becomes insignificant. We can then discard these coefficients without introducing significant error in the linear combination of basis vectors and achieve higher coding efficiency.

In the above three interpretations of transform coding, we see that the linear unitary transform can provide the following two functions:

1. Decorrelate input data; i.e., transform coefficients are less correlated than the original data, and
2. Have some transform coefficients more significant than others (with large variance, eigenvalue, or weight in basis vector expansion) such that transform coefficients can be treated differently: some can be discarded, some can be coarsely quantized, and some can be finely quantized.

Note that the definition of *unitary* transform is given shortly in Section 4.2.1.3.

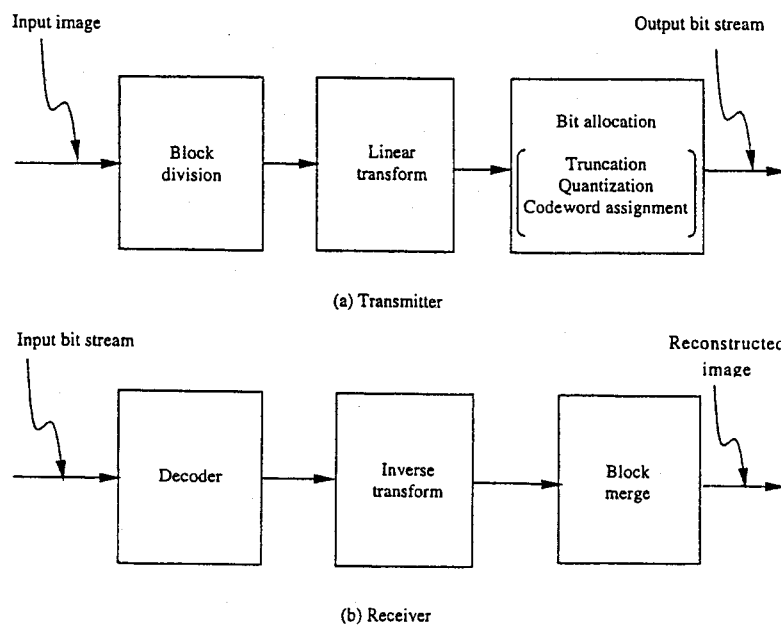


FIGURE 4.2 Block diagram of transform coding.

4.1.5 PROCEDURES OF TRANSFORM CODING

Prior to leaving this section, we summarize the procedures of transform coding. There are three steps in transform coding as shown in Figure 4.2. First, the input data (frame) are divided into blocks (subimages). Each block is then linearly transformed. The transformed version is then truncated, quantized, and encoded. These last three functions, which are discussed in Section 4.4, can be grouped and termed as bit allocation. The output of the encoder is a bitstream.

In the receiver, the bitstream is decoded and then inversely transformed to form reconstructed blocks. All the reconstructed blocks collectively produce a replica of the input image.

4.2 LINEAR TRANSFORMS

In this section, we first discuss a general formulation of a linear unitary 2-D image transform. Then, a basis image interpretation of TC is given.

4.2.1 2-D IMAGE TRANSFORMATION KERNEL

There are two different ways to handle image transformation. In the first way, we convert a 2-D array representing a digital image into a 1-D array via row-by-row stacking, for example. That is, from the second row on, the beginning of each row in the 2-D array is cascaded to the end of its previous row. Then we transform this 1-D array using a 1-D transform. After the transformation, we can convert the 1-D array back to a 2-D array. In the second way, a 2-D transform is directly applied to the 2-D array corresponding to an input image, resulting in a transformed 2-D array. These two ways are essentially the same. It can be straightforwardly shown that the difference between the two is simply a matter of notation (Wintz, 1972). In this section, we use the second way to handle image transformation. That is, we work on 2-D image transformation.

Assume a digital image is represented by a 2-D array $g(x, y)$, where (x, y) is the coordinates of a pixel in the 2-D array, while g is the gray level value (also often called intensity or brightness) of the pixel. Denote the 2-D transform of $g(x, y)$ by $T(u, v)$, where (u, v) is the coordinates in the transformed domain. Assume that both $g(x, y)$ and $T(u, v)$ are a square 2-D array of $N \times N$; i.e., $0 \leq x, y, u, v \leq N - 1$.

The 2-D forward and inverse transforms are defined as

$$T(u, v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} g(x, y) f(x, y, u, v) \quad (4.21)$$

and

$$g(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) i(x, y, u, v) \quad (4.22)$$

where $f(x, y, u, v)$ and $i(x, y, u, v)$ are referred to as the forward and inverse *transformation kernels*, respectively.

A few characteristics of transforms are discussed below.

4.2.1.1 Separability

A transformation kernel is called separable (hence, the transform is said to be separable) if the following conditions are satisfied.

$$f(x, y, u, v) = f_1(x, u) f_2(y, v), \quad (4.23)$$

and

$$i(x, y, u, v) = i_1(x, u) i_2(y, v). \quad (4.24)$$

Note that a 2-D separable transform can be decomposed into two 1-D transforms. That is, a 2-D transform can be implemented by a 1-D transform rowwise followed by another 1-D transform columnwise. That is,

$$T_1(x, v) = \sum_{y=0}^{N-1} g(x, y) f_2(y, v), \quad (4.25)$$

where $0 \leq x, v \leq N - 1$, and

$$T(u, v) = \sum_{x=0}^{N-1} T_1(x, v) f_1(x, u), \quad (4.26)$$

where $0 \leq u, v \leq N - 1$. Of course, the 2-D transform can also be implemented in a reverse order with two 1-D transforms, i.e., columnwise first, followed by rowwise. The counterparts of Equations 4.25 and 4.26 for the inverse transform can be derived similarly.

4.2.1.2 Symmetry

The transformation kernel is symmetric (hence, the transform is symmetric) if the kernel is separable and the following condition is satisfied:

$$f_1(y, v) = f_2(y, v). \quad (4.27)$$

That is, f_1 is functionally equivalent to f_2 .

4.2.1.3 Matrix Form

If a transformation kernel is symmetric (hence, separable) then the 2-D image transform discussed above can be expressed compactly in the following matrix form. Denote an *image matrix* by G and $G = \{g_{i,j}\} = \{g(i-1, j-1)\}$. That is, a typical element (at the i th row and j th column) in the matrix G is the pixel gray level value in the 2-D array $g(x, y)$ at the same geometrical position. Note that the subtraction of one in the notation $g(i-1, j-1)$ comes from Equations 4.21 and 4.22. Namely, the indexes of a square 2-D image array are conventionally defined from 0 to $N-1$, while the indexes of a square matrix are from 1 to N . Denote the *forward transform matrix* by F and $F = \{f_{i,j}\} = \{f_1(i-1, j-1)\}$. We then have the following matrix form of a 2-D transform:

$$T = F^T G F \quad (4.28)$$

where T on the left-hand side of the equation denotes the matrix corresponding to the transformed 2-D array in the same fashion as that used in defining the G matrix. The inverse transform can be expressed as

$$G = I^T T I \quad (4.29)$$

where the matrix I is the *inverse transform matrix* and $I = \{i_{j,k}\} = \{i_1(j-1, k-1)\}$. The forward and inverse transform matrices have the following relation:

$$I = F^{-1} \quad (4.30)$$

Note that all of the matrices defined above, G , T , F , and I are of $N \times N$.

It is known that the discrete Fourier transform involves complex quantities. In this case, the counterparts of Equations 4.28, 4.29, and 4.30 become Equations 4.31, 4.32, and 4.33, respectively:

$$T = F *^T G F \quad (4.31)$$

$$G = I *^T T I \quad (4.32)$$

$$I = F^{-1} = F *^T \quad (4.33)$$

where $*$ indicates complex conjugation. Note that the transform matrices F and I contain complex quantities and satisfy Equation 4.33. They are called unitary matrices and the transform is referred to as a unitary transform.

4.2.1.4 Orthogonality

A transform is said to be orthogonal if the transform matrix is orthogonal. That is,

$$F^T = F^{-1} \tag{4.34}$$

Note that an orthogonal matrix (orthogonal transform) is a special case of a unitary matrix (unitary transform), where only real quantities are involved. We will see that all the 2-D image transforms, presented in Section 4.3, are separable, symmetric, and unitary.

4.2.2 BASIS IMAGE INTERPRETATION

Here we study the concept of *basis images* or *basis matrices*. Recall that we discussed basis vectors when we considered the 1-D transform. That is, the components of the transformed vector (also referred to as the transform coefficients) can be interpreted as the coefficients in the basis vector expansion of the input vector. Each coefficient is essentially the amount of correlation between the input vector and the corresponding basis vector. The concept of basis vectors can be extended to basis images in the context of 2-D image transforms.

Recall that the 2-D inverse transform introduced at the beginning of this section is defined as

$$g(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) i(x, y, u, v) \tag{4.35}$$

where $0 \leq x, y \leq N - 1$. This equation can be viewed as a *component* form of the inverse transform. As defined above in Section 4.2.1.3, the whole image $\{g(x, y)\}$ is denoted by the image matrix G of $N \times N$. We now denote the "image" formed by the inverse transformation kernel $\{i(x, y, u, v), 0 \leq x, y \leq N - 1\}$ as a 2-D array $I_{u,v}$ of $N \times N$ for a specific pair of (u, v) with $0 \leq u, v \leq N - 1$. Recall that a digital image can be represented by a 2-D array of gray level values. In turn the 2-D array can be arranged into a matrix. Namely, we treat the following three: a digital image, a 2-D array (with proper resolution), and a matrix (with proper indexing), interchangeably. We then have

$$I_{u,v} = \begin{bmatrix} i(0,0,u,v) & i(0,1,u,v) & \dots & \dots & i(0,N-1,u,v) \\ i(1,0,u,v) & i(1,1,u,v) & \dots & \dots & i(1,N-1,u,v) \\ \vdots & \vdots & \dots & \dots & \vdots \\ \vdots & \vdots & \dots & \dots & \vdots \\ i(N-1,0,u,v) & i(N-1,1,u,v) & \dots & \dots & i(N-1,N-1,u,v) \end{bmatrix} \tag{4.36}$$

The 2-D array $I_{u,v}$ is referred to as a basis image. There are N^2 basis images in total since $0 \leq u, v \leq N - 1$. The inverse transform expressed in Equation 4.35 can then be written in a *collective* form as

$$G = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) I_{u,v} \tag{4.37}$$

We can interpret this equation as a series expansion of the original image G into a set of N^2 basis images $I_{u,v}$. The transform coefficients $T(u, v)$, $0 \leq u, v \leq N - 1$, become the coefficients of the expansion. Alternatively, the image G is said to be a weighted sum of basis images. Note that,

similar to the 1-D case, the coefficient or the weight $T(u,v)$ is a correlation measure between the image G and the basis image $I_{u,v}$ (Wintz, 1972).

Note that basis images have nothing to do with the input image. Instead, it is completely defined by the transform itself. That is, basis images are the attribute of 2-D image transforms. Different transforms have different sets of basis images.

The motivation behind transform coding is that with a proper transform, hence, a proper set of basis images, the transform coefficients are more independent than the gray scales of the original input image. In the ideal case, the transform coefficients are statistically independent. We can then optimally encode the coefficients independently, which can make coding more efficient and simple. As pointed out in (Wintz, 1972), however, this is generally impossible because of the following two reasons. First, it requires the joint probability density function of the N^2 pixels, which have not been deduced from basic physical laws and cannot be measured. Second, even if the joint probability density functions were known, the problem of devising a reversible transform that can generate independent coefficients is unsolved. The optimum linear transform we can have results in uncorrelated coefficients. When Gaussian distribution is involved, we can have independent transform coefficients. In addition to the uncorrelatedness of coefficients, the variance of the coefficients varies widely. Insignificant coefficients can be ignored without introducing significant distortion in the reconstructed image. Significant coefficients can be allocated more bits in encoding. The coding efficiency is thus enhanced.

As shown in Figure 4.3, TC can be viewed as expanding the input image into a set of basis images, then quantizing and encoding the coefficients associated with the basis images separately. At the receiver the coefficients are reconstructed to produce a replica of the input image. This strategy is similar to that of subband coding, which is discussed in Chapter 8. From this point of view, transform coding can be considered a special case of subband coding, though transform coding was devised much earlier than subband coding.

It is worth mentioning an alternative way to define basis images. That is, a basis image with indexes (u, v) , $I_{u,v}$, of a transform can be constructed as the *outer product* of the u th basis vector, \vec{b}_u , and the v th basis vector, \vec{b}_v , of the transform. The basis vector, \vec{b}_u , is the u th column vector of the inverse transform matrix I (Jayant and Noll, 1984). That is,

$$I_{u,v} = \vec{b}_u \vec{b}_v^T. \quad (4.38)$$

4.2.3 SUBIMAGE SIZE SELECTION

The selection of subimage (block) size, N , is important. Normally, the larger the size the more decorrelation the transform coding can achieve. It has been shown, however, that the correlation between image pixels becomes insignificant when the distance between pixels becomes large, e.g., it exceeds 20 pixels (Habibi, 1971a). On the other hand, a large size causes some problems. In adaptive transform coding, a large block cannot adapt to local statistics well. As will be seen later in this chapter, a transmission error in transform coding affects the whole associated subimage. Hence a large size implies a possibly severe effect of transmission error on reconstructed images. As will be shown in video coding (Section III and Section IV), transform coding is used together with motion-compensated coding. Consider that large block size is not used in motion estimation; subimage sizes of 4, 8, and 16 are used most often. In particular, $N = 8$ is adopted by the international still image coding standard, JPEG, as well as video coding standards H.261, H.263, MPEG 1, and MPEG 2.

4.3 TRANSFORMS OF PARTICULAR INTEREST

Several commonly used image transforms are discussed in this section. They include the discrete Fourier transform, the discrete Walsh transform, the discrete Hadamard transform, and the discrete Cosine and Sine transforms. All of these transforms are symmetric (hence, separable as well),

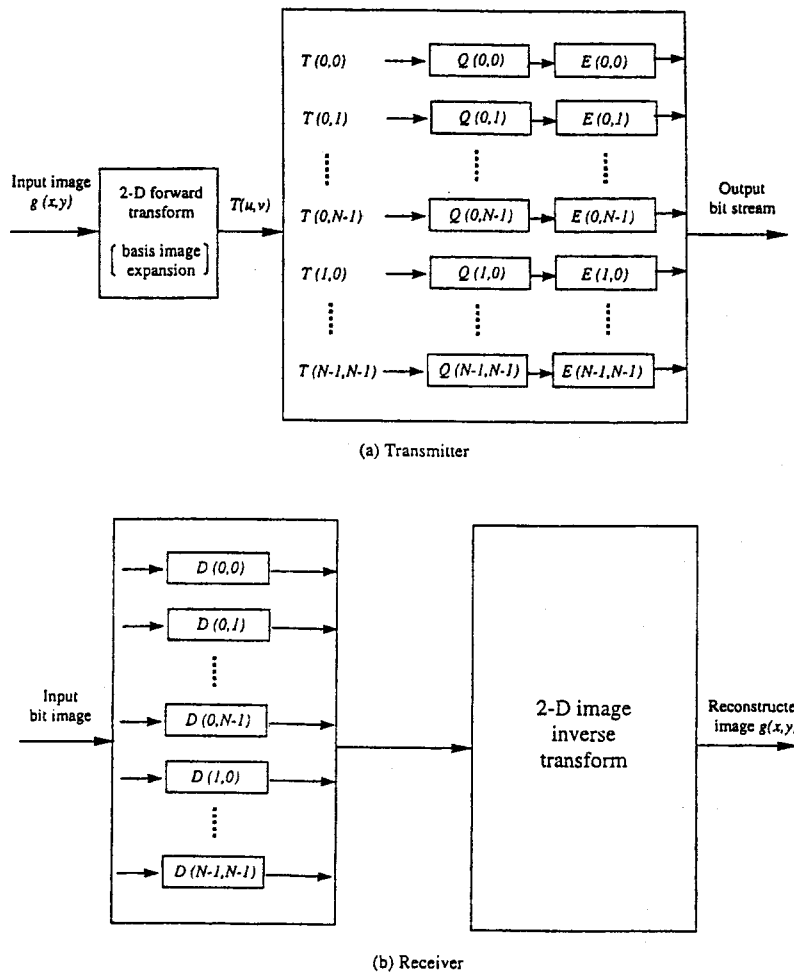


FIGURE 4.3 Basis image interpretation of TC (Q: quantizer, E: encoder, D: decoder).

unitary, and reversible. For each transform, we define its transformation kernel and discuss its basis images.

4.3.1 DISCRETE FOURIER TRANSFORM (DFT)

The DFT is of great importance in the field of digital signal processing. Owing to the fast Fourier transform (FFT) based on the algorithm developed in (Cooley, 1965), the DFT is widely utilized for various tasks of digital signal processing. It has been discussed in many signal and image processing texts. Here we only define it by using the transformation kernel just introduced above. The forward and inverse transformation kernels of the DFT are

$$f(x, y, u, v) = \frac{1}{N} \exp\{-j2\pi(xu + yv)/N\} \tag{4.39}$$

and

$$i(x, y, u, v) = \frac{1}{N} \exp\{j2\pi(xu + yv)/N\} \tag{4.40}$$

Clearly, since complex quantities are involved in the DFT transformation kernels, the DFT is generally complex. Hence, we use the unitary matrix to handle the DFT (refer to Section 4.2.1.3). The basis vector of the DFT \vec{b}_u is an $N \times 1$ column vector and is defined as

$$\vec{b}_u = \frac{1}{\sqrt{N}} \left[1, \exp\left(j2\pi \frac{u}{N}\right), \exp\left(j2\pi \frac{2u}{N}\right), \dots, \exp\left(j2\pi \frac{(N-1)u}{N}\right) \right]^T \quad (4.41)$$

As mentioned, the basis image with index (u, v) , $I_{u,v}$, is equal to $\vec{b}_u \vec{b}_v^T$. A few basis images are listed below for $N = 4$.

$$I_{0,0} = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \quad (4.42)$$

$$I_{0,1} = \frac{1}{4} \begin{pmatrix} 1 & j & -1 & -j \\ 1 & j & -1 & -j \\ 1 & j & -1 & -j \\ 1 & j & -1 & -j \end{pmatrix} \quad (4.43)$$

$$I_{1,2} = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & -j \\ j & -j & j & -j \\ -1 & 1 & -1 & 1 \\ -j & -j & -j & j \end{pmatrix} \quad (4.44)$$

$$I_{3,3} = \frac{1}{4} \begin{pmatrix} 1 & -j & -1 & j \\ -j & -1 & j & 1 \\ -1 & j & 1 & -j \\ j & 1 & -j & -1 \end{pmatrix} \quad (4.45)$$

4.3.2 DISCRETE WALSH TRANSFORM (DWT)

The transformation kernels of the DWT (Walsh, 1923) are defined as

$$f(x, y, u, v) = \frac{1}{N} \prod_{i=0}^{n-1} [(-1)^{p_i(x)p_{n-1-i}(u)} (-1)^{p_i(y)p_{n-1-i}(v)}] \quad (4.46)$$

and

$$i(x, y, u, v) = f(x, y, u, v). \quad (4.47)$$

where $n = \log_2 N$, $p_i(\arg)$ represents the i th bit in the natural binary representation of the arg, the 0 th bit corresponds to the least significant bit, and the $(n-1)$ th bit corresponds to the most significant

the DFT is
in 4.2.1.3).

(4.41)

images are

(4.42)

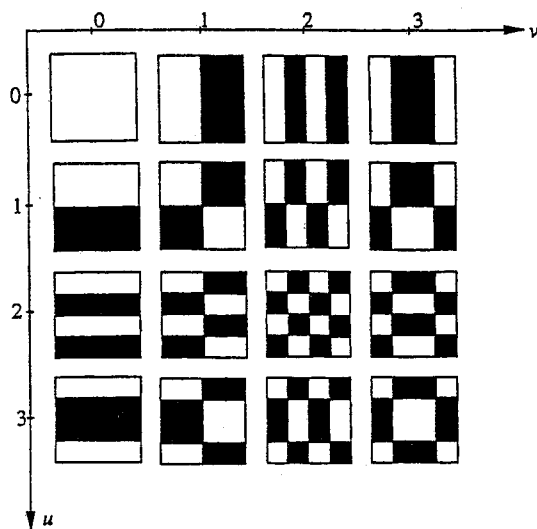


FIGURE 4.4 When $N = 4$: a set of the 16 basis images of DWT.

(4.43)

bit. For instance, consider $N = 16$, then $n = 4$. The natural binary code of number 8 is 1000. Hence, $p_0(8) = p_1(8) = p_2(8) = 0$, and $p_3(8) = 1$. We see that if the factor $1/N$ is put aside then the forward transformation kernel is always an integer: either +1 or -1. In addition, the inverse transformation kernel is the same as the forward transformation kernel. Therefore, we conclude that the implementation of the DWT is simple.

(4.44)

When $N = 4$, the 16 basis images of the DWT are shown in Figure 4.4. Each corresponds to a specific pair of (u, v) and is of resolution 4×4 in the x - y coordinate system. They are binary images, where the bright represents +1, while the dark -1. The transform matrix of the DWT is shown below for $N = 4$.

(4.45)

$$F = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \quad (4.48)$$

4.3.3 DISCRETE HADAMARD TRANSFORM (DHT)

The DHT (Hadamard, 1893) is closely related to the DWT. This can be seen from the following definition of the transformation kernels.

(4.46)

$$f(x, y, u, v) = \frac{1}{N} \prod_{i=0}^n [(-1)^{p_i(x)p_i(u)} (-1)^{p_i(y)p_i(v)}] \quad (4.49)$$

and

(4.47)

$$i(x, y, u, v) = f(x, y, u, v) \quad (4.50)$$

where the definitions of n , i , and $p_i(\text{arg})$ are the same as in the DWT. For this reason, the term Walsh-Hadamard transform (DWHT) is frequently used to represent either of the two transforms.

arg, the
significant

When N is a power of 2, the transform matrices of the DWT and DHT have the same row (or column) vectors except that the order of row (or column) vectors in the matrices are different. This is the only difference between the DWT and DHT under the circumstance $N = 2^n$. Because of this difference, while the DWT can be implemented by using the FFT algorithm with a straightforward modification, the DHT needs more work to use the FFT algorithm. On the other hand, the DHT possesses the following recursive feature, while the DWT does not:

$$F_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (4.51)$$

and

$$F_{2N} = \begin{bmatrix} F_N & F_N \\ F_N & -F_N \end{bmatrix} \quad (4.52)$$

where the subscripts indicate the size of the transform matrices. It is obvious that the transform matrix of the DHT can be easily derived by using the recursion.

Note that the number of sign changes between consecutive entries in a row (or a column) of a transform matrix (from positive to negative and from negative to positive) is known as *sequency*. It is observed that the sequency does not monotonically increase as the order number of rows (or columns) increases in the DHT. Since sequency bears some similarity to frequency in the Fourier transform, sequency is desired as an increasing function of the order number of rows (or columns). This is realized by the *ordered* Hadamard transform (Gonzalez, 1992).

The transformation kernel of the ordered Hadamard transform is defined as

$$f(x, y, u, v) = \frac{1}{N} \prod_{i=0}^{N-1} [(-1)^{p_i(x)d_i(u)} (-1)^{p_i(y)d_i(v)}] \quad (4.53)$$

where the definitions of i , $p_i(\arg)$ are the same as defined above for the DWT and DHT. The $d_i(\arg)$ is defined as below.

$$\begin{aligned} d_0(\arg) &= b_{n-1}(\arg) \\ d_1(\arg) &= b_{n-1}(\arg) + b_{n-2}(\arg) \\ d_{n-1}(\arg) &= b_1(\arg) + b_0(\arg) \end{aligned} \quad (4.54)$$

The 16 basis images of the ordered Hadamard transform are shown in Figure 4.5 for $N = 4$. It is observed that the variation of the binary basis images becomes more frequent monotonically when u and v increase. Also we see that the basis image expansion is similar to the frequency expansion of the Fourier transform in the sense that an image is decomposed into components with different variations. In transform coding, these components with different coefficients are treated differently.

4.3.4 DISCRETE COSINE TRANSFORM (DCT)

The DCT is the most commonly used transform for image and video coding.

same row (or different. This cause of this straightforward and, the DHT

(4.51)

(4.52)

e transform

column) of s *sequency*, of rows (or the Fourier r columns).

(4.53)

The $d_i(\arg)$

(4.54)

for $N = 4$. It notonically frequency onents with are treated

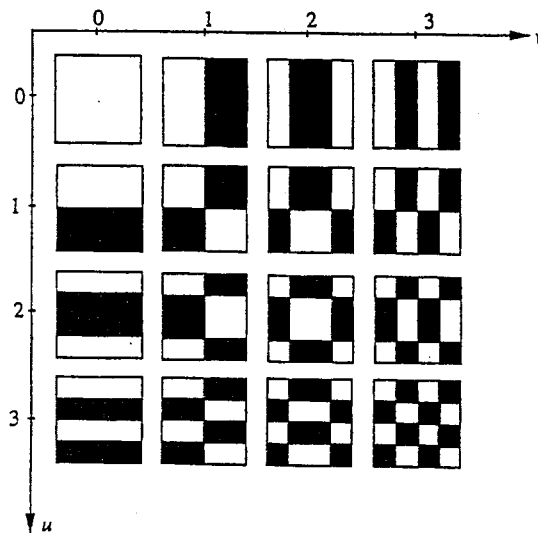


FIGURE 4.5 When $N = 4$: a set of the 16 basis images of the ordered DHT.

4.3.4.1 Background

The DCT, which plays an extremely important role in image and video coding, was established by Ahmed et al. (1974). There, it was shown that the *basis member* $\cos[(2x + 1)u\pi/2N]$ is the u th Chebyshev polynomial $T_u(\xi)$ evaluated at the x th zero of $T_N(\xi)$. Recall that the Chebyshev polynomials are defined as

$$T_0(\xi) = 1/\sqrt{2} \tag{4.55}$$

$$T_k(\xi) = \cos[k \cos^{-1}(\xi)] \tag{4.56}$$

where $T_k(\xi)$ is the k th order Chebyshev polynomial and it has k zeros, starting from the 1st zero to the k th zero. Furthermore, it was demonstrated that the basis vectors of 1-D DCT provide a good approximation to the eigenvectors of the class of Toeplitz matrices defined as

$$\begin{bmatrix} 1 & \rho & \rho^2 & \dots & \rho^{N-1} \\ \rho & 1 & \rho & \dots & \rho^{N-2} \\ \rho^2 & \rho & 1 & \dots & \rho^{N-3} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ \rho^{N-1} & \rho^{N-2} & \rho^{N-3} & \dots & 1 \end{bmatrix}, \tag{4.57}$$

where $0 < \rho < 1$.

4.3.4.2 Transformation Kernel

The transformation kernel of the 2-D DCT can be extended straightforwardly from that of 1-D DCT as follows:

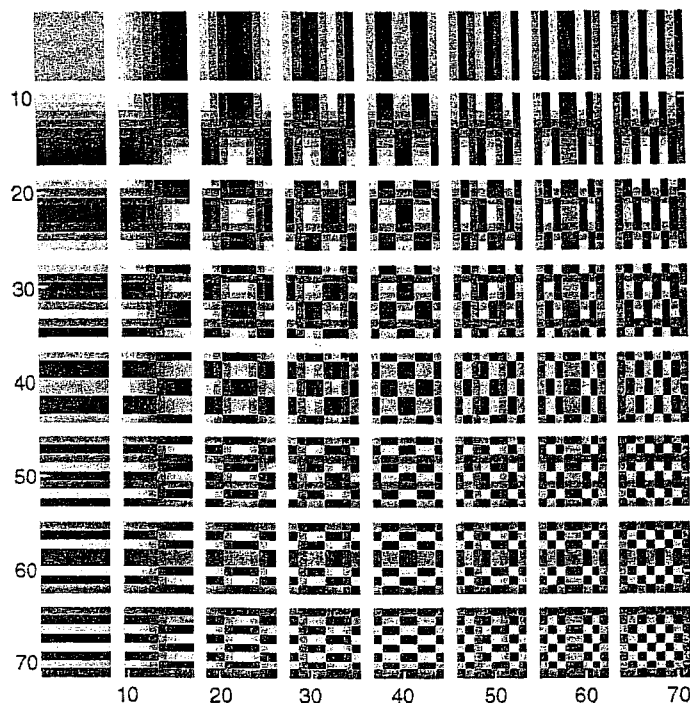


FIGURE 4.6 When $N = 8$: a set of the 64 basis images of the DCT.

$$f(x, y, u, v) = C(u)C(v) \cos\left(\frac{(2x+1)u\pi}{2N}\right) \cos\left(\frac{(2y+1)v\pi}{2N}\right) \quad (4.58)$$

where

$$C(u) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for } u = 0 \\ \sqrt{\frac{2}{N}} & \text{for } u = 1, 2, \dots, N-1 \end{cases} \quad (4.59)$$

$$i(x, y, u, v) = f(x, y, u, v). \quad (4.60)$$

Note that the $C(v)$ is defined the same way as in Equation 4.59. The 64 basis images of the DCT are shown in Figure 4.6 for $N = 8$.

4.3.4.3 Relationship with DFT

The DCT is closely related to the DFT. This can be examined from an alternative method of defining the DCT. It is known that applying the DFT to an N -point sequence $g_N(n)$, $n = 0, 1, \dots, N-1$, is equivalent to the following:

1. Repeating $g_N(n)$ every N points, form a periodic sequence, $\tilde{g}_N(n)$, with a fundamental period N . That is,

$$\tilde{g}_N(n) = \sum_{i=-\infty}^{\infty} g_N(n - iN). \quad (4.61)$$

2. Determine the Fourier series expansion of the periodic sequence $\tilde{g}_N(n)$. That is, determine all the coefficients in the Fourier series which are known to be periodic with the same fundamental period N .
3. Truncate the sequence of the Fourier series coefficients so as to have the same support as that of the given sequence $g_N(n)$. That is, only keep the N coefficients with indexes $0, 1, \dots, N-1$ and set all the others to equal zero. These N Fourier series coefficients form the DFT of the given N -point sequence $g_N(n)$.

An N -point sequence $g_N(n)$ and the periodic sequence $\tilde{g}_N(n)$, generated from $g_N(n)$, are shown in Figure 4.7(a) and (b), respectively. In summary, the DFT can be viewed as a correspondence between two periodic sequences. One is the periodic sequence $\tilde{g}_N(n)$, which is formed by periodically repeating $g_N(n)$. The other is the periodic sequence of Fourier series coefficients of $\tilde{g}_N(n)$.

The DCT of an N -point sequence is obtained via the following three steps:

1. Flip over the given sequence with respect to the end point of the sequence to form a $2N$ -point sequence, $g_{2N}(n)$, as shown in Figure 4.7(c). Then form a periodic sequence $\tilde{g}_{2N}(n)$, shown in Figure 4.7(d), according to

$$\tilde{g}_{2N}(n) = \sum_{i=-\infty}^{\infty} g_{2N}(n - 2iN) \quad (4.62)$$

(4.58)

2. Find the Fourier series coefficients of the periodic sequences $\tilde{g}_{2N}(n)$.
3. Truncate the resultant periodic sequence of the Fourier series coefficients to have the support of the given finite sequence $g_N(n)$. That is, only keep the N coefficients with indexes $0, 1, \dots, N-1$ and set all the others to equal zero. These N Fourier series coefficients form the DCT of the given N -point sequence $g_N(n)$.

(4.59)

(4.60)

A comparison between Figure 4.7(b) and (d) reveals that the periodic sequence $\tilde{g}_N(n)$ is not smooth. There usually exist discontinuities at the beginning and end of each period. These end-head discontinuities cause a high-frequency distribution in the corresponding DFT. On the contrary, the periodic sequence $\tilde{g}_{2N}(n)$ does not have this type of discontinuity due to flipping over the given finite sequence. As a result, there is no high-frequency component corresponding to the end-head discontinuities. Hence, the DCT possesses better energy compaction in the low frequencies than the DFT. By better energy compaction, we mean more energy is compacted in a fraction of transform coefficients. For instance, it is known that the most energy of an image is contained in a small region of low frequency in the DFT domain. Vivid examples can be found in (Gonzalez and Woods, 1992). In terms of energy compaction, when compared with the Karhunen-Loeve transform (the Hotelling transform is its discrete version), which is known as the optimal, the DCT is the best among the DFT, DWT, DHT, and discrete Haar transform.

Besides this advantage, the DCT can be implemented using the FFT. This can be seen from the above discussion. There, it has been shown that the DCT of an N -point sequence, $g_N(n)$, can be obtained from the DFT of the $2N$ -point sequence $g_{2N}(n)$. Furthermore, the even symmetry

of the DCT

of defining
..., $N-1$, is

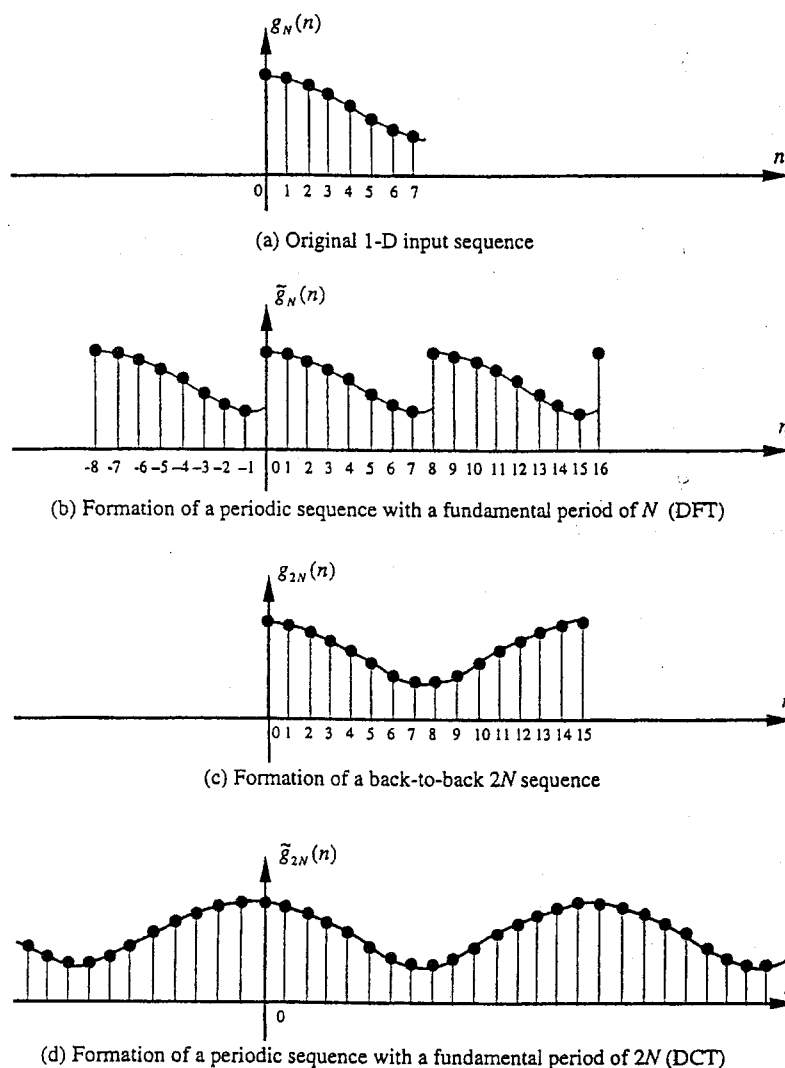


FIGURE 4.7 An example to illustrate the differences and similarities between DFT and DCT.

in $\tilde{g}_{2N}(n)$ makes the computation required for the DCT of an N -point equal to that required for the DFT of the N -point sequence. Because of these two merits, the DCT is the most popular image transform used in image and video coding nowadays.

4.3.5 PERFORMANCE COMPARISON

In this subsection, we compare the performance of a few commonly used transforms in terms of energy compaction, mean square reconstruction error, and computational complexity.

4.3.5.1 Energy Compaction

Since all the transforms we discussed are symmetric (hence separable) and unitary, the matrix form of the 2-D image transform can be expressed as $T = F^T G F$ as discussed in Section 4.2.1.3. In the 1-D case, the transform matrix F is the counterpart of the matrix Φ discussed in the Hotelling

transform. Using the F , one can transform a 1-D column vector \vec{z} into another 1-D column vector \vec{y} . The components of the vector \vec{y} are transform coefficients. The variances of these transform coefficients, and therefore the signal energy associated with the transform coefficients, can be arranged in a nondecreasing order. It can be shown that the total energy before and after the transform remains the same. Therefore, the more energy compacted in a fraction of total coefficients, the better energy compaction the transform has. One measure of energy compaction is the *transform coding gain* G_{TC} , which is defined as the ratio between the arithmetic mean and the geometric mean of the variances of all the components in the transformed vector (Jayant, 1984).

$$G_{TC} = \frac{\frac{1}{N} \sum_{i=0}^{N-1} \sigma_i^2}{\left(\prod_{i=0}^{N-1} \sigma_i^2 \right)^{\frac{1}{N}}} \quad (4.63)$$

A larger G_{TC} indicates higher energy compaction. The transform coding gains for a first-order autoregressive source with $\rho = 0.95$ achieved by using the DCT, DFT, and KLT was reported in (Zelinski and Noll, 1975; Jayant and Noll, 1984). The transform coding gain afforded by the DCT compares very closely to that of the optimum KLT.

4.3.5.2 Mean Square Reconstruction Error

The performance of the transforms can be compared in terms of the mean square reconstruction error as well. This was mentioned in Section 4.1.2 when we provided a statistical interpretation for transform coding. That is, after arranging all the N transformed coefficients according to their variances in a nonincreasing order, if $L < N$ and we discard the last $(N - L)$ coefficients to reconstruct the original input signal \vec{z} (similar to what we did with the Hotelling transform), then the mean square reconstruction error is

$$MSE_r = E[\|\vec{z}' - \vec{z}\|^2] = \sum_{i=L+1}^N \sigma_i^2, \quad (4.64)$$

where \vec{z}' denotes the reconstructed vector. Note that in the above-defined mean square reconstruction error, the quantization error and transmission error have not been included. Hence, it is sometimes referred to as the mean square approximation error. Therefore it is desired to choose a transform so that the transformed coefficients are "more independent" and more energy is concentrated in the first L coefficients. Then it is possible to discard the remaining coefficients to save coding bits without causing significant distortion in input signal reconstruction.

In terms of the mean square reconstruction error, the performance of the DCT, KLT, DFT, DWT, and discrete Haar transform for the 1-D case was reported in Ahmed et al. (1974). The variances of the 16 transform coefficients are shown in Figure 4.8 when $N = 16$, $\rho = 0.95$. Note that N stands for the dimension of the 1-D vector, while the parameter ρ is shown in the Toeplitz matrix (refer to Equation 4.57). We can see that the DCT compares most closely to the KLT, which is known to be optimum.

Note that the unequal variance distribution among transform coefficients has also found application in the field of pattern recognition. Similar results to those in Ahmed et al. (1974) for the DFT, DWT, and Haar transform were reported in (Andrews, 1971).

A similar analysis can be carried out for the 2-D case (Wintz, 1972). Recall that an image $g(x, y)$ can be expressed as a weighted sum of basis images $I_{u,v}$. That is,

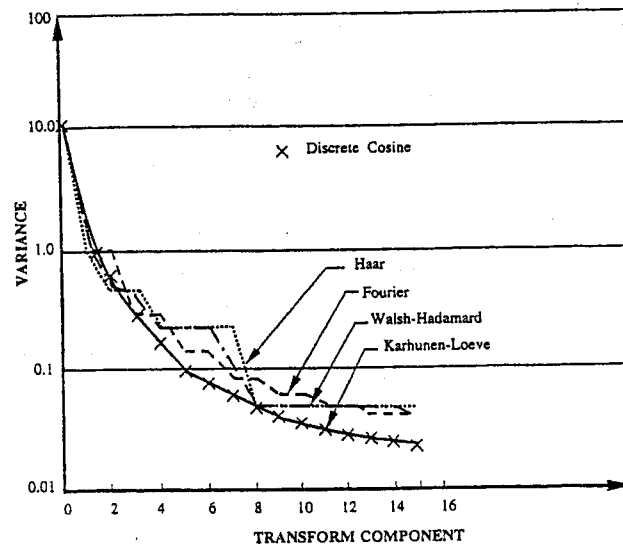


FIGURE 4.8 Transform coefficient variances when $N = 16$, $\rho = 0.95$. (From Ahmed, N. et al., *IEEE Trans. Comput.*, 90, 1974. With permission.)

$$G = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) I_{u,v} \quad (4.65)$$

where the weights are transform coefficients. We arrange the coefficients according to their variances in a nonincreasing order. For some choices of the transform (hence basis images), the coefficients become insignificant after the first L terms, and the image can be approximated well by truncating the coefficients after L . That is,

$$G = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) I_{u,v} \approx \sum_{u=0}^{L} \sum_{v=0}^{L} T(u, v) I_{u,v} \quad (4.66)$$

The mean square reconstruction error is given by

$$MSE_r = \sum_L \sum_{u,v} \sigma_{u,v}^2 \quad (4.67)$$

A comparison among the KLT, DHT, and DFT in terms of the mean square reconstruction error for 2-D array of 16×16 (i.e., 256 transform coefficients) was reported in (Figure 5, Wintz, 1972). Note that the discrete KLT is image dependent. In the comparison, the KLT is calculated with respect to an image named "Cameraman." It shows that while the KLT achieves the best performance, the other transforms perform closely.

In essence, the criteria of mean square reconstruction error and energy compaction are closely related. It has been shown that the discrete Karhunen transform (KLT), also known as the Hotelling transform, is the optimum in terms of energy compaction and mean square reconstruction error. The DWT, DHT, DFT, and DCT are close to the optimum (Wintz, 1972; Ahmed et al., 1974); however, the DCT is the best among these several *suboptimum* transforms.

Note that the performance comparison among various transforms in terms of bit rate vs. distortion in the reconstructed image was reported in (Pearl et al., 1972; Ahmed et al., 1974). The same conclusion was drawn. That is, the KLT is optimum, while the DFT, DWT, DCT, and Haar transforms are close in performance. Among the suboptimum transforms, the DCT is the best.

4.3.5.3 Computational Complexity

Note that while the DWT, DHT, DFT, and DCT are input image independent, the discrete KLT (the Hotelling transform) is input dependent. More specifically, the row vectors of the Hotelling transform matrix are transposed eigenvectors of the covariance matrix of the input random vector. So far there is no fast transform algorithm available. This computational complexity prohibits the Hotelling transform from practical usage. It can be shown that the DWT, DFT, and DCT can be implemented using the FFT algorithm.

4.3.5.4 Summary

As pointed out above, the DCT is the best among the suboptimum transforms in terms of energy compaction. Moreover, the DCT can be implemented using the FFT. Even though a $2N$ -point sequence is involved, the even symmetry makes the computation involved in the N -point DCT equivalent to that of the N -point FFT. For these two reasons, the DCT finds the widest application in image and video coding.

4.4 BIT ALLOCATION

As shown in Figure 4.2, in transform coding, an input image is first divided into blocks (subimages). Then a 2-D linear transform is applied to each block. The transformed blocks go through truncation, quantization, and codeword assignment. The last three functions: truncation, quantization, and codeword assignment, are combined and called bit allocation.

From the previous section, it is known that the applied transform decorrelates subimages. Moreover, it redistributes image energy in the transform domain in such a way that most of the energy is compacted into a small fraction of coefficients. Therefore, it is possible to discard the majority of transform coefficients without introducing significant distortion.

As a result, we see that in transform coding there are mainly three types of errors involved. One is due to truncation. That is, the majority of coefficients are truncated to zero. Others come from quantization. (Note that truncation can also be considered a special type of quantization). Transmission errors are the third type of error. Recall that the mean square reconstruction error discussed in Section 4.3.5.2 is in fact only related to truncation error. For this reason, it was referred to more precisely as a mean square approximation error. In general, the reconstruction error, i.e., the error between the original image signal and the reconstructed image at the receiver, includes three types of errors: truncation error, quantization error, and transmission error.

There are two different ways to truncate transform coefficients. One is called *zonal coding*, while the other is *threshold coding*. They are discussed below.

4.4.1 ZONAL CODING

In zonal coding, also known as *zonal sampling*, a zone in the transformed block is predefined according to a statistical average obtained from many blocks. All transform coefficients in the zone are retained, while all coefficients outside the zone are set to zero. As mentioned in Section 4.3.5.1, the total energy of the image remains the same after applying the transforms discussed there. Since it is known that the DC and low-frequency AC coefficients of the DCT occupy most of the energy, the zone is located in the top-left portion of the transformed block when the transform coordinate

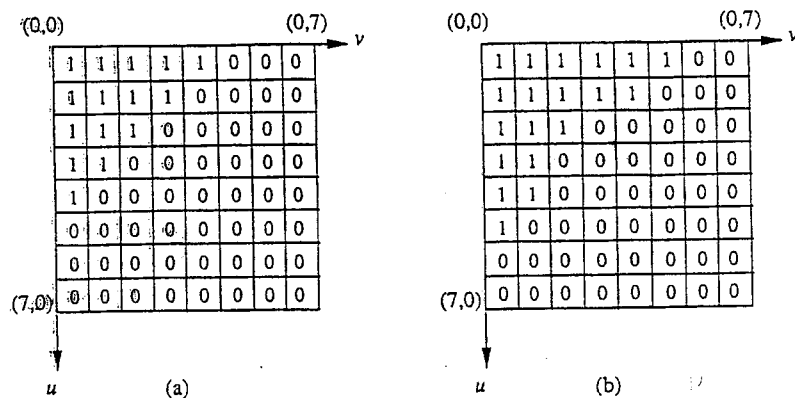


FIGURE 4.9 Two illustrations of zonal coding.

system is set conventionally. (Note that by DC we mean $u = v = 0$. By AC we mean u and v do not equal zero simultaneously.) That is, the origin is at the top-left corner of the transformed block. Two typical zones are shown in Figure 4.9. The simplest uniform quantization with natural binary coding can be used to quantize and encode the retained transform coefficients. With this simple technique, there is no overhead side information that needs to be sent to the receiver, since the structure of the zone, the scheme of the quantization, and encoding are known at both the transmitter and receiver.

The coding efficiency, however, may not be very high. This is because the zone is predefined based on average statistics. Therefore some coefficients outside the zone might be large in magnitude, while some coefficients inside the zone may be small in quantity. Uniform quantization and natural binary encoding are simple, but they are known not to be efficient enough.

For further improvement of coding efficiency, an adaptive scheme has to be used. There, a two-pass procedure is applied. In the first pass, the variances of transform coefficients are measured or estimated. Based on the statistics, the quantization and encoding schemes are determined. In the second pass, quantization and encoding are carried out (Habibi, 1971a; Chen and Smith, 1977).

4.4.2 THRESHOLD CODING

In threshold coding, also known as threshold sampling, there is not a predefined zone. Instead, each transform coefficient is compared with a threshold. If it is smaller than the threshold, then it is set to zero. If it is larger than the threshold, it will be retained for quantization and encoding. Compared with zonal coding, this scheme is adaptive in truncation in the sense that the coefficients with more energy are retained no matter where they are located. The addresses of these retained coefficients, however, have to be sent to the receiver as side information. Furthermore, the threshold is determined after an evaluation of all coefficients. Hence, it was usually a two-pass adaptive technique.

Chen and Pratt (1984) devised an efficient adaptive scheme to handle threshold coding. It is a one-pass adaptive scheme, in contrast to the two-pass adaptive schemes. Hence it is fast in implementation. With several effective techniques that will be addressed here, it achieved excellent results in transform coding. Specifically, it demonstrated a satisfactory quality of reconstructed frames at a bit rate of 0.4 bits per pixel for coding of color images, which corresponds to real-time color television transmission over a 1.5-Mb/sec channel. This scheme has been adopted by the international still coding standard JPEG. A block diagram of the threshold coding proposed by Chen and Pratt is shown in Figure 4.10. More details and modification made by JPEG will be described in Chapter 7.

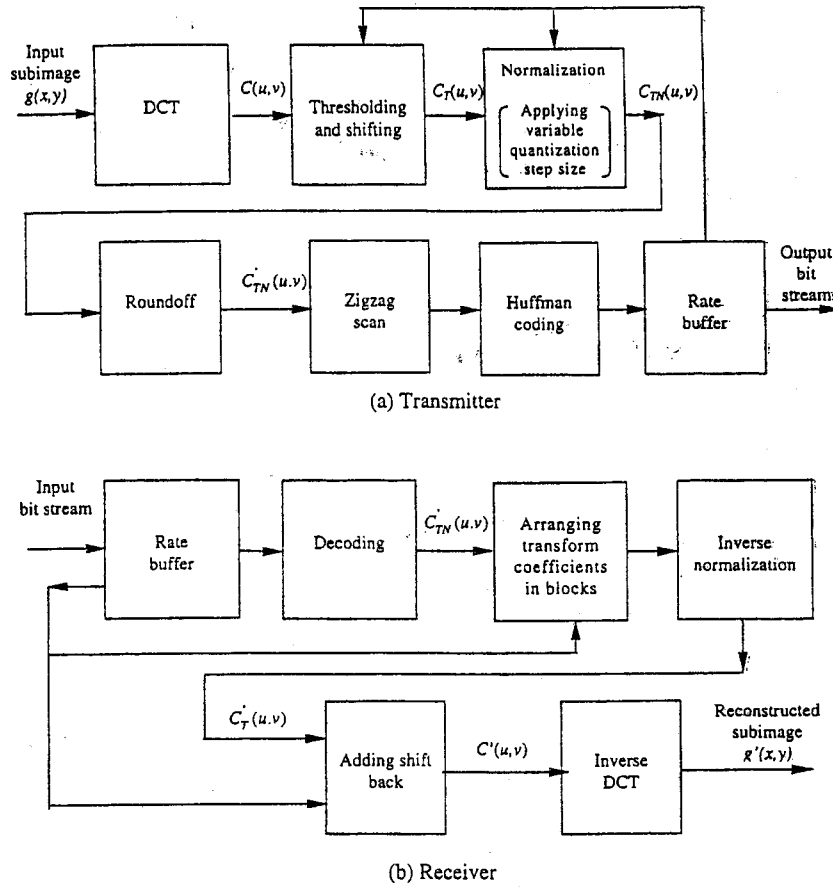


FIGURE 4.10 Block diagram of the algorithm proposed by Chen and Pratt (1984).

4.4.2.1 Thresholding and Shifting

The DCT is used in the scheme because of its superiority, described in Section 4.3. Here we use $C(u,v)$ to denote the DCT coefficients. The DC coefficient, $C(0,0)$, is processed differently. As mentioned in Chapter 3, the DC coefficients are encoded with a differential coding technique. For more details, refer to Chapter 7. For all the AC coefficients, the following thresholding and shifting are carried out:

$$C_T(u,v) = \begin{cases} C(u,v) - T & \text{if } C(u,v) > T \\ 0 & \text{if } C(u,v) \leq T \end{cases} \quad (4.68)$$

where T on the right-hand side is the threshold. Note that the above equation also implies a shifting of transform coefficients by T when $C(u,v) > T$. The input-output characteristic of the thresholding and shifting is shown in Figure 4.11.

Figure 4.12 demonstrates that more than 60% of the DCT coefficients normally fall below a threshold value as low as 5. This indicates that with a properly selected threshold value it is possible to set most of the DCT coefficients equal to zero. The threshold value is adjusted by the feedback from the rate buffer, or by the desired bit rate.

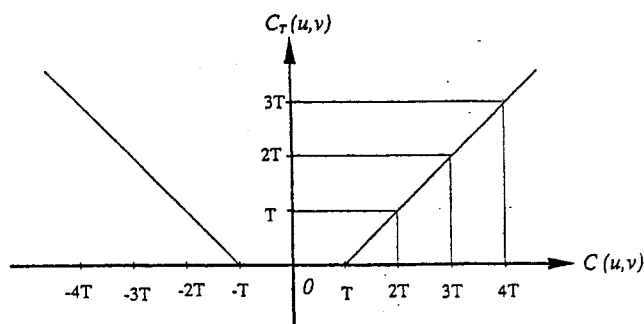


FIGURE 4.11 Input-output characteristic of thresholding and shifting.

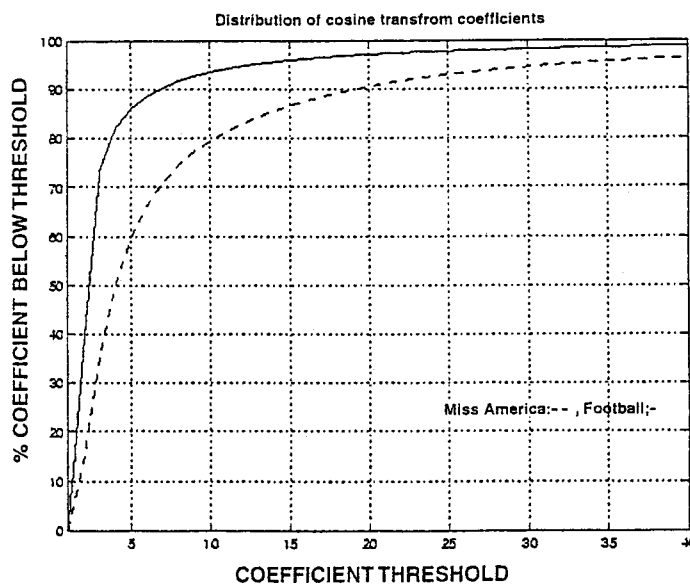


FIGURE 4.12 Amplitude distribution of the DCT coefficients.

4.4.2.2 Normalization and Roundoff

The threshold subtracted transform coefficients $C_T(u,v)$ are normalized before roundoff. The normalization is implemented as follows:

$$C_{TN}(u,v) = \frac{C_T(u,v)}{\Gamma_{u,v}} \quad (4.69)$$

where the normalization factor $\Gamma_{u,v}$ is controlled by the rate buffer. The roundoff process converts floating point to integer as follows.

$$R[C_{TN}(u,v)] = C_{TN}(u,v) = \begin{cases} \lfloor C_{TN}(u,v) + 0.5 \rfloor & \text{if } C_{TN}(u,v) \geq 0 \\ \lceil C_{TN}(u,v) - 0.5 \rceil & \text{if } C_{TN}(u,v) < 0 \end{cases} \quad (4.70)$$

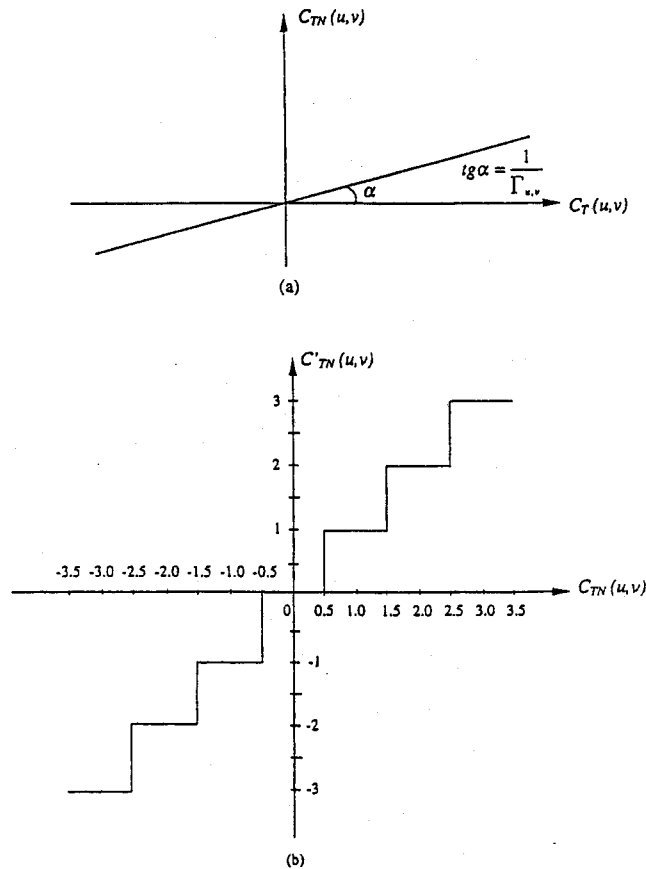


FIGURE 4.13 Input-output characteristic of (a) normalization, (b) roundoff.

where the operator $\lfloor x \rfloor$ means the largest integer smaller than or equal to x , the operator $\lceil x \rceil$ means the smallest integer larger than or equal to x . The input-output characteristics of the normalization and roundoff are shown in Figure 4.13(a) and (b), respectively.

From these input-output characteristics, we can see that the roundoff is a uniform midtread quantizer with a unit quantization step. The combination of normalization and roundoff is equivalent to a uniform midtread quantizer with the quantization step size equal to the normalization factor $\Gamma_{u,v}$. Normalization is a scaling process, which makes the resultant uniform midtread quantizer adapt to the dynamic range of the associated transform coefficient. It is therefore possible for one quantizer design to be applied to various coefficients with different ranges. Obviously, by adjusting the parameter $\Gamma_{u,v}$ (quantization step size) a variable bit rate and mean square quantization error can be achieved. Hence, the selection of the normalization factors for different transform coefficients can take the statistical feature of the images and the characteristics of the human visual system (HVS) into consideration. In general, most image energy is contained in the DC and low-frequency AC transform coefficients. The HVS is more sensitive to a relatively uniform region than to a relatively detailed region, as discussed in Chapter 1. Chapter 1 also mentions that, with regard to the color image, the HVS is more sensitive to the luminance component than to the chrominance components.

These have been taken into consideration in JPEG. A matrix consisting of all the normalization factors is called a quantization table in JPEG. A luminance quantization table and a chrominance quantization table used in JPEG are shown in Figure 4.14. We observe that in general in both tables

roundoff. The nor-

(4.69)

process converts

(4.70)

4.4.2.4 Huffman Coding

Statistical studies of the magnitude of nonzero DCT coefficients and the run-length of zero DCT coefficients in zigzag scanning were conducted in (Chen and Pratt, 1984). The domination of the coefficients with small amplitudes and the short run-lengths was found and is shown in Figures 4.16 and 4.17. This justifies the application of the Huffman coding to the magnitude of nonzero transform coefficients and run-lengths of zeros.

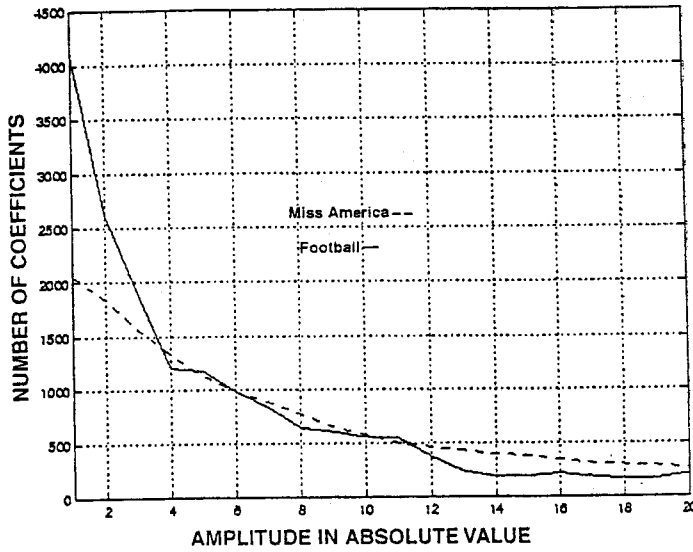


FIGURE 4.16 Histogram of DCT coefficients in absolute amplitude.

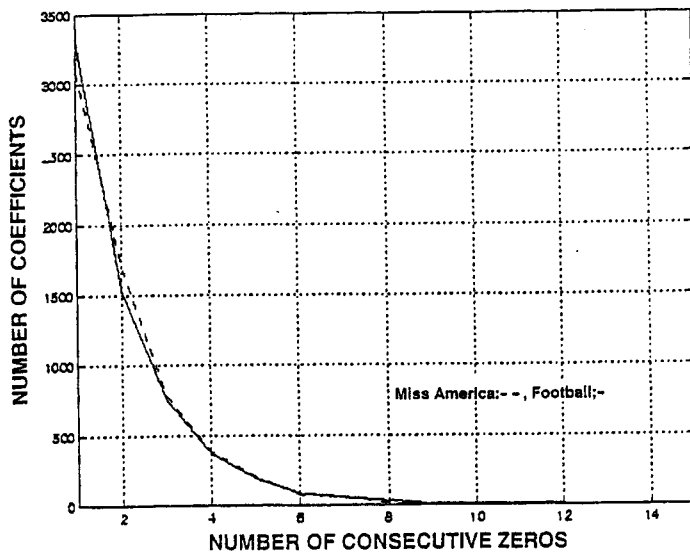


FIGURE 4.17 Histogram of zero run length.

4.4.2.5 Special Codewords

Two special codewords were used by Chen and Pratt (1984). One is called *end of block* (EOB). Another is called *run-length prefix*. Once the last nonzero DCT coefficients in the zigzag is coded, EOB is appended, indicating the termination of coding the block. This further saves bits used in coding. A run-length prefix is used to discriminate the run-length codewords from the amplitude codewords.

4.4.2.6 Rate Buffer Feedback and Equalization

As shown in Figure 4.10, a rate buffer accepts a variable-rate data input from the encoding process and provides a fixed-rate data output to the channel. The status of the rate buffer is monitored and fed back to control the threshold and the normalization factor. In this fashion a one-pass adaptation is achieved.

4.5 SOME ISSUES

4.5.1 EFFECT OF TRANSMISSION ERRORS

In transform coding, each pixel in the reconstructed image relies on all transform coefficients in the subimage where the pixel is located. Hence, a bit reversal transmission error will spread. That is, an error in a transform coefficient will lead to errors in all the pixels within the subimage. As discussed in Section 4.2.3, this is one of the reasons the selected subimage size cannot be very large. Depending on which coefficient is in error, the effect caused by a bit reversal error on the reconstructed image varies. For instance, an error in the DC or a low-frequency AC coefficient may be objectionable, while an error in the high-frequency coefficient may be less noticeable.

4.5.2 RECONSTRUCTION ERROR SOURCES

As discussed, three sources: truncation (discarding transform coefficients with small variances), quantization, and transmission contribute to the reconstruction error. It is noted that in TC the transform is applied block by block. Quantization and encoding of transform coefficients are also conducted blockwise. At the receiver, reconstructed blocks are put together to form the whole reconstructed image. In the process, block artifacts are produced. Sometimes, even though it may not severely affect an objective assessment of the reconstructed image quality, block artifacts can be annoying to the HVS, especially when the coding rate is low.

To alleviate the blocking effect, several techniques have been proposed. One is to overlap blocks in the source image. Another is to postfilter the reconstructed image along block boundaries. The selection of advanced transforms is an additional possible method (Lim, 1990).

In the block-overlapping method, when the blocks are finally organized to form the reconstructed image, each pixel in the overlapped regions takes an average value of all its reconstructed gray level values from multiple blocks. In this method, extra bits are used for those pixels involved in the overlapped regions. For this reason, the overlapped region is usually only one pixel wide.

Due to the sharp transition along block boundaries, block artifacts are of high frequency in nature. Hence, low-pass filtering is normally used in the postfiltering method. To avoid the blurring effect caused by low-pass filtering on the nonboundary image area, low-pass postfiltering is only applied to block boundaries. Unlike the block-overlapping method, the postfiltering method does not need extra bits. Moreover, it has been shown that the postfiltering method can achieve better results in combating block artifacts (Reeve and Lim, 1984; Ramamurthi and Gersho, 1986). For these two reasons, the postfiltering method has been adopted by the international coding standards.