Table 4.1 Horizontal eigenvalues for the first picture in the "Mobile and Calendar" sequence.

Eigenvalue number	Eigenvalue	Percentage of energy
1	25,664	82.4
2	2,104	6.8
3	1,224	3.9
4	877	2.8
5	615	2.0
6	383	1.2
7	203	0.7
8	56	0.2

■

4.4. THE DISCRETE COSINE TRANSFORM

Although the Karhunen–Loeve transform is the optimum transform in terms of energy compaction, it suffers from the significant difficulty that the transform needs to be defined for each picture (or even each 8×8 block within a picture). This requires a significant amount of computation both to calculate the covariance matrix and then the eigenvectors that are used as the basis vectors of the transform. In addition, the transform basis functions (eigenvectors) required for each picture (or each 8×8 block within a picture) need to be transmitted to the decoder so that the picture can be correctly decoded. This represents a significant overhead.

For this reason, a fixed transform known as the discrete cosine transform (DCT) is commonly used in picture and video coding applications. Although suboptimal when compared to the Karhunen–Loeve transform, it has the advantage that it can be used without the need for calculating covariance matrices and eigenvectors. In addition, there is no need to transmit information on the basis vectors used to the receiver.

The basis vectors of an N-point DCT in one-dimension are defined as

$$C(u) = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos\left[\frac{(2x+1)u\pi}{2N}\right] \qquad u = 0,1,2,...,N-1$$

Similarly, the inverse DCT is defined as

$$f(x) = \sum_{u=0}^{N-1} \alpha(u) C(u) \cos\left[\frac{(2x+1)u\pi}{2N}\right] \qquad x = 0,1,2,...,N-1$$

In both of these equations

$$\alpha(u) = \begin{cases} \sqrt{\dfrac{1}{N}} & \text{for } u = 0 \\ \sqrt{\dfrac{2}{N}} & \text{for } u = 1,2,...N-1 \end{cases}$$

$$\begin{bmatrix}
+0.354 & +0.354 & +0.354 & +0.354 & +0.354 & +0.354 & +0.354 & +0.354 \\
+0.490 & +0.416 & +0.278 & +0.098 & -0.098 & -0.278 & -0.416 & -0.490 \\
+0.462 & +0.191 & -0.191 & -0.462 & -0.462 & -0.191 & +0.191 & +0.462 \\
+0.416 & -0.098 & -0.490 & -0.278 & +0.278 & +0.490 & +0.098 & -0.416 \\
+0.354 & -0.354 & -0.354 & +0.354 & +0.354 & -0.354 & -0.354 & +0.354 \\
+0.278 & -0.490 & +0.098 & +0.416 & -0.416 & -0.098 & +0.490 & -0.278 \\
+0.191 & -0.462 & +0.462 & -0.191 & -0.191 & +0.462 & -0.462 & +0.191 \\
+0.098 & -0.278 & +0.416 & -0.490 & +0.490 & -0.416 & +0.278 & -0.098
\end{bmatrix}$$

Figure 4.8 Basis functions for the DCT.

The basis vectors for an eight-point DCT are given in the matrix shown in Figure 4.8. They are also shown in Figure 4.9.

These are just sampled versions of cosine waveforms of increasing frequency ranging from 0 periods per vector (i.e., constant) in the case of the first vector to 3.5 periods per vector in the case of the last vector with each vector containing 0.5 additional periods to the one before it.

Comparing the eigenvectors from the Karhunen–Loeve transform shown in Figure 4.7 with those for the DCT shown in Figure 4.9, we note a remarkable

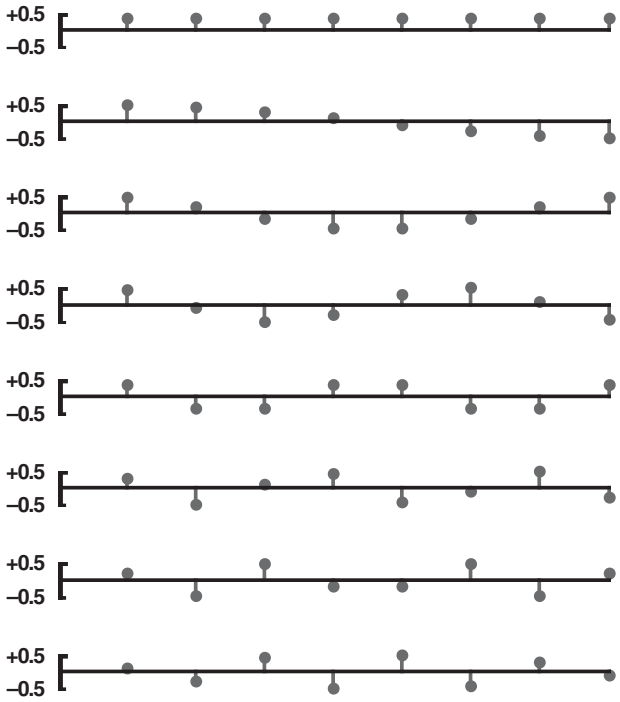


Figure 4.9 Basis vectors for the eight-point discrete cosine transform.

similarity in general shape. The most significant difference is that several of the corresponding eigenvectors in each figure are approximately the negative of each other. However, the effect of changing the sign of an eigenvector is just that the sign of the associated transform coefficients will be changed—energy compaction is identical. The DCT then seems to be a reasonable approximation to the Karhunen–Loeve transform at least for this picture.

For pictures, a two-dimensional DCT is required. The forward and reverse transform are calculated according to the equations

$$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\left[\frac{(2y+1)v\pi}{2N}\right] \quad u,v = 0,1,2,...,N-1$$

and

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\left[\frac{(2y+1)v\pi}{2N}\right] \quad x,y = 0,1,2,...,N-1$$

where again

$$\alpha(u) = \begin{cases} \sqrt{\dfrac{1}{N}} & \text{for } u = 0 \\ \sqrt{\dfrac{2}{N}} & \text{for } u = 1, 2, ..., N-1 \end{cases}$$

The basis vectors $t(u,v)$ in this case are two-dimensional arrays defined by

$$t(u,v) = \alpha(u)\alpha(v)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\left[\frac{(2y+1)v\pi}{2N}\right] \quad \text{for } x,y = 0,1,...,N-1$$

Pictures of the 64 two-dimensional basis vectors from an 8×8 transform are shown in Figure 4.10 that includes an offset so that negative values appear dark and positive values appear light with mid-gray representing values close to zero. Vertical frequency increases from top to bottom whereas horizontal frequency increases from left to right. Any 8×8 pixel block can be represented by a weighted sum of these two-dimensional basis vectors.

Fortunately, the DCT is a separable transform. This means that the two-dimensional transform can be obtained by first applying a one-dimensional transform across the rows of the data. The result after the horizontal transform has a one-dimensional transform applied vertically to yield the final two-dimensional transform result. This greatly simplifies the transform procedure as well as significantly increasing the speed of the transform.

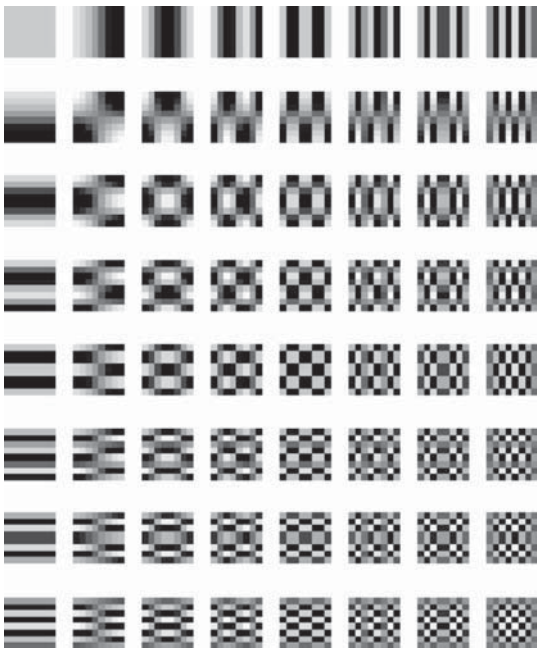


Figure 4.10 Basis vectors for two-dimensional discrete cosine transform.

EXAMPLE 4.3—MATLAB

For the 8 × 8 pixel block shown in Figure 4.11, calculate the two-dimensional DCT. Hence, show that the picture can be represented by a weighted sum of the two-dimensional basis vectors shown in Figure 4.10.

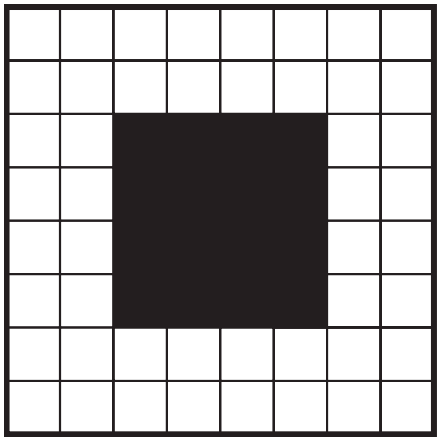


Figure 4.11 Block to be analyzed in Example 4.3.

This picture is represented by the 8 × 8 pixel array of data shown in Figure 4.12 where white is represented by 255 and black by 0.

255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255
255	255	0	0	0	0	255	255
255	255	0	0	0	0	255	255
255	255	0	0	0	0	255	255
255	255	0	0	0	0	255	255
255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255

Figure 4.12 Numerical picture data for Example 4.3.

As explained earlier, the two-dimensional DCT can be calculated by first taking the one-dimensional DCT horizontally and then vertically. After applying the horizontal DCT and rounding to the nearest integer we obtain the result shown in Figure 4.13.

+721	0	0	0	0	0	0	0
+721	0	0	0	0	0	0	0
361	0	333	0	0	0	–138	0
361	0	333	0	0	0	–138	0
361	0	333	0	0	0	–138	0
361	0	333	0	0	0	–138	0
+721	0	0	0	0	0	0	0
+721	0	0	0	0	0	0	0

Figure 4.13 Data of Figure 4.12 after one-dimensional horizontal DCT.

The final two-dimensional DCT is obtained by repeating the one-dimensional discrete cosine transform down the columns of this result. The final result is shown in Figure 4.14.

+1530	0	471	0	0	0	–195	0
0	0	0	0	0	0	0	0
+471	0	–435	0	0	0	+180	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
–195	0	+180	0	0	0	–75	0
0	0	0	0	0	0	0	0

Figure 4.14 Data of Figure 4.12 after two-dimensional DCT.

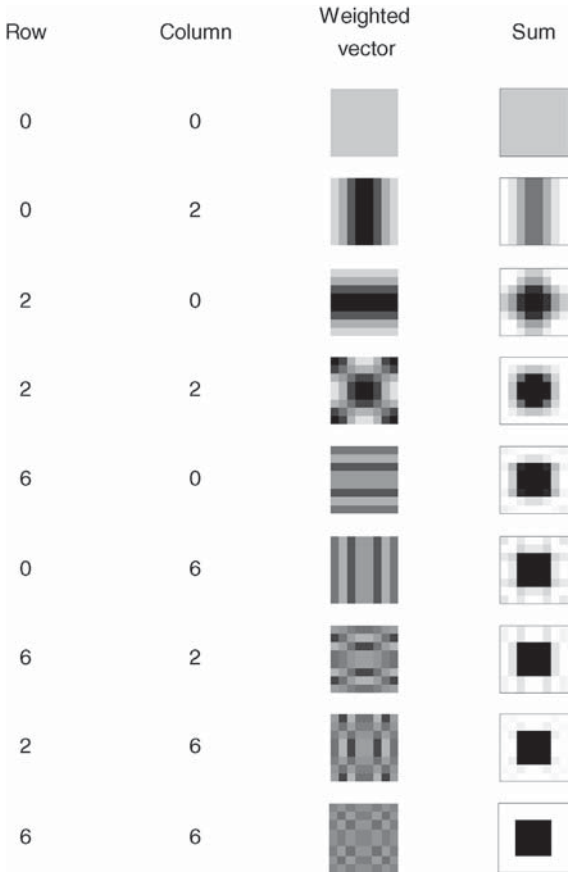


Figure 4.15 Weighted two-dimensional basis vectors for the block of Figure 4.11 together with the result from summing the weighted basis vectors to reconstruct the original block.

The weighted two-dimensional basis vectors for each nonzero term in the two-dimensional DCT are shown in Figure 4.15. Also shown is the result when the current weighted basis vector is added to the sum of all the weighted basis vectors appearing above it in Figure 4.15. When all the weighted basis vectors have been summed, we end up with the original block. The last few basis vectors make only a small change to the final block despite the fact that the block chosen has a number of sharp discontinuities that usually imply significant energy at high frequencies. A smoother block would show even less distortion when high frequency basis vectors were omitted. ■

4.4.1. Choice of Transform Block Size

Example 4.3 uses a block size of 8×8 pixels. The appropriate block size is a compromise between the amount of compression achieved (which tends to increase with block size), the correlation within the picture (which tends to decrease with block size), the ability to adapt to local picture statistics (which is better as block size

decreases), and computational complexity (which increases with block size). The block size is invariably chosen to be a power of two (i.e., 4×4, 8×8, and 16×16 pixels) as this simplifies computational complexity.

EXAMPLE 4.4—MATLAB

For the first picture of the "Mobile and Calendar" sequence, divide the picture into square blocks of size $N = 2$, 4, 8, 16, 32, and 64 pixels and calculate the DCT of each block. Now retain only the top $N/2 \times N/2$ pixels and calculate the RMS error for each reconstructed picture. Hence, comment on the most appropriate choice of transform block size.

The MATLAB function dct2(A) will calculate the two-dimensional DCT of a block of data. Appropriate MATLAB code to perform this task is given below. The DCT block size is set by the variable tf_size.

```
tf_size = 8;
for irow = 1:tf_size:row
  for icol = 1:tf_size:col
    dct_block = dct2(A(irow:irow1(tf_size-1),icol:icol+(tf_size-1)));
    limit = (tf_size/2)+1;
    dct_block(limit:tf_size,:) = zeros((tf_size/2),tf_size);
    dct_block(:,limit:tf_size) = zeros(tf_size,(tf_size/2));
    rec(irow:irow+(tf_size-1),icol:icol+(tf_size-1)) = round(idct2(dct_block));
    end
end
    rms = sqrt(mean(mean((A - rec). * (A-rec))));
```

The result when applied to the picture is shown in Figure 4.16.

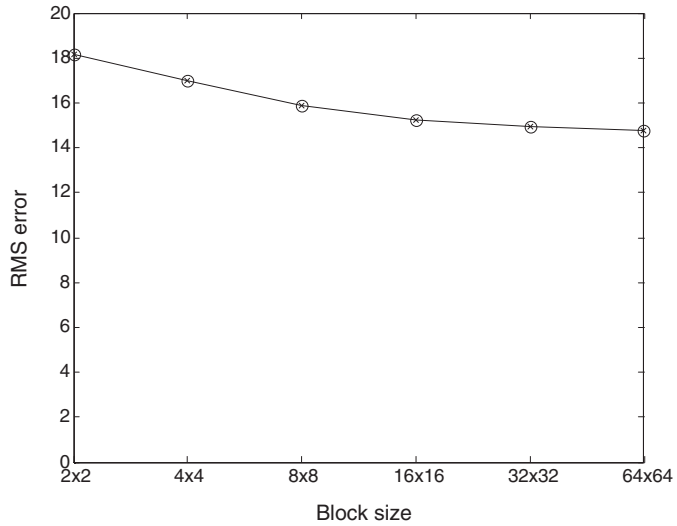


Figure 4.16 Result of deleting all but the top left quarter coefficients for the first picture of the "Mobile and Calendar" sequence.

It is clear that most of the savings are achieved by the time a block size between 8×8 and 16×16 pixels is reached. Hardware complexity considerations lead to the choice of an 8×8 pixel block size. ■

4.4.2. Quantization of DCT Transform Coefficients

We have now succeeded in transforming integer pixel values into real transform coefficients. Transmitting these coefficients without any further processing would probably lead to an increase in the number of bits of information required to represent the picture. However, the transform has packed most of the energy of the picture into a small number of coefficients. Quantizing these coefficients and then transmitting only the significant ones can result in a significant saving. The question remains how best to do this. As the energy is compacted primarily into the first few (low frequency) coefficients, one approach would be to simply not transmit a number of the other (high frequency) coefficients. This is considered in Example 4.5.

EXAMPLE 4.5—MATLAB

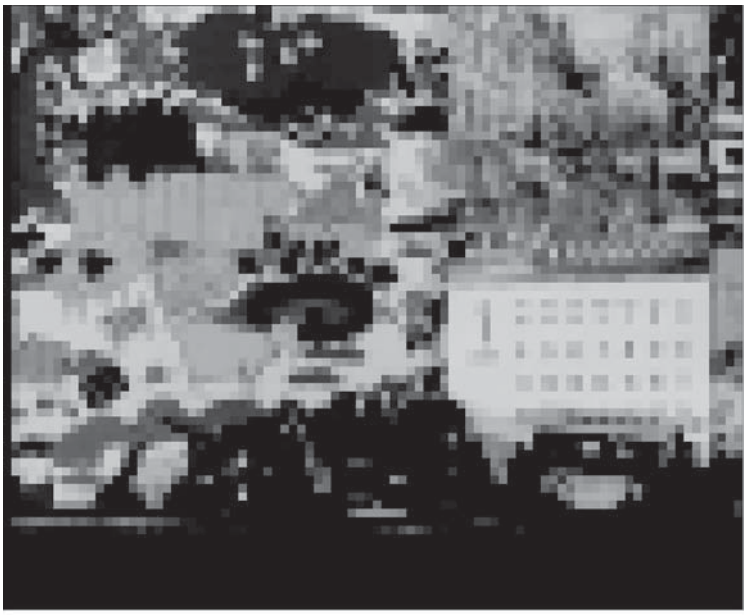
For the first luminance picture in the sequence "Mobile and Calendar," calculate the resulting picture when only the top left 4×4, 2×2, and 1×1 DCT coefficients are retained.

The results are shown in Figure 4.17. Even retaining the top 4×4 low-frequency coefficients (Fig. 4.17a) leads to significant blurring in the reconstructed picture. Reducing this

(a)

Figure 4.17 Effect of deleting high-frequency DCT coefficients: (a) top 4×4 coefficients retained; (b) top 2×2 coefficients retained; (c) top 1×1 coefficient retained.

(b)



(c)

Figure 4.17 (*Continued*)

to the top 2×2 low-frequency coefficients (Fig. 4.17b) greatly increases the blurring. In addition, the edges of the individual 8×8 pixel blocks start to become obvious. When only the single DC coefficient is retained (Fig. 4.17c) then the picture becomes a series of blocks. This is hardly surprising as retaining only the DC coefficient means that each pixel in the 8×8 pixel block is replaced by the average value of the block. ■

The previous example has demonstrated that performing the DCT and then simply deleting the higher frequency coefficients is not a satisfactory approach if high-quality reconstructed pictures are required. Although low-frequency information is almost always important, simply removing high-frequency information leads to blurring at sharp edges where high-frequency information is significant.

After quantization, it is desirable that the maximum number of coefficients are zero as this reduces the amount of information that needs to be transmitted. For this reason, a quantizer with a larger than normal "dead zone'' (i.e., a quantization region where the coefficient will be set to zero) as shown in Figure 4.18 is commonly employed.

By comparison, a completely linear quantizer would have decision levels at … $-2.5Q$, $-1.5Q$, $-0.5Q$, $+0.5Q$, $+1.5Q$, $+2.5Q$… and reconstruction levels at …$-2Q$, $-Q$, 0, $+Q$, $+2Q$… The larger dead zone ensures that all coefficients in the range $-Q$ to Q are set to zero. The value of the quantizer (Q) is chosen by the user to ensure an adequate representation of the picture. More is said on this topic later.

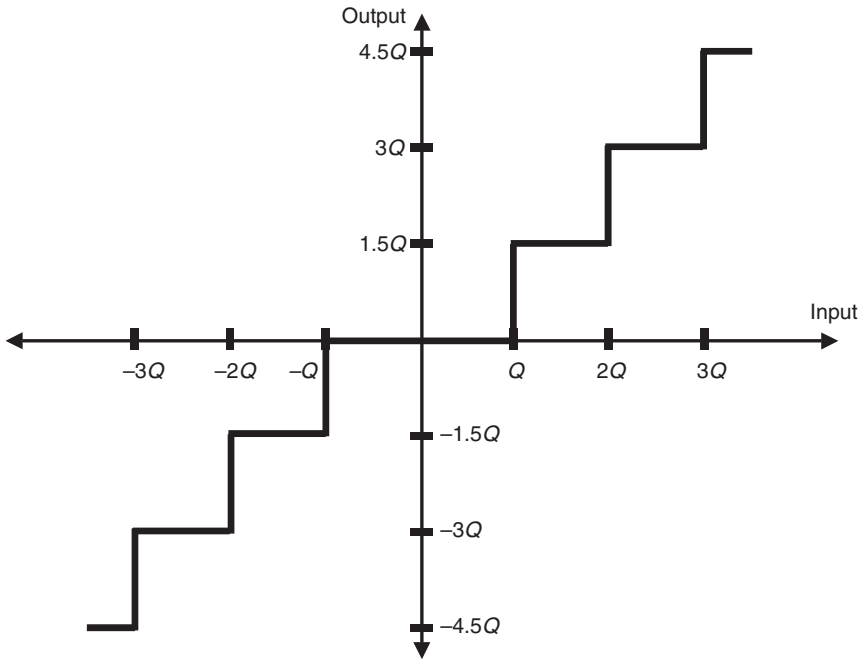


Figure 4.18 Quantizer with central dead zone.

4.4.3. Quantization of DCT Coefficients Based on the Human Visual System

Despite the results shown in Example 4.5, it is well known that the sensitivity of the human visual system does indeed decrease as the spatial frequency (usually measured in cycles per degree of arc) increases. Figure 4.19 shows an indicative plot of the relative spatial frequency response of the eye as a function of spatial frequency measured in cycles per degree of sight.

It is clear that the response peaks at a spatial frequency around 5–10 cycles/degree and falls off sharply at higher frequencies. However, even at these higher frequencies a significant signal will still be observable.

Almost invariably, the transform coefficients are quantized by a linear or near linear quantizer. However, the step size of the quantizer can be varied according to the spatial frequency represented with the step size increasing as the spatial frequency increases. This ensures that high-frequency coefficients will be quantized to zero unless they are sufficiently large that they are likely to be observable to the human visual system.

One way that this can be done is to use a quantization relationship such as

$$\hat{C}_{i,j} = \text{round}\left(\frac{8 \times C_{i,j}}{Q \times W_{i,j}}\right)$$

where $\hat{C}_{i,j}$ is the value of the quantized transform coefficient, $C_{i,j}$ is the value of the original transform coefficient, Q is the quantizer step size for a particular

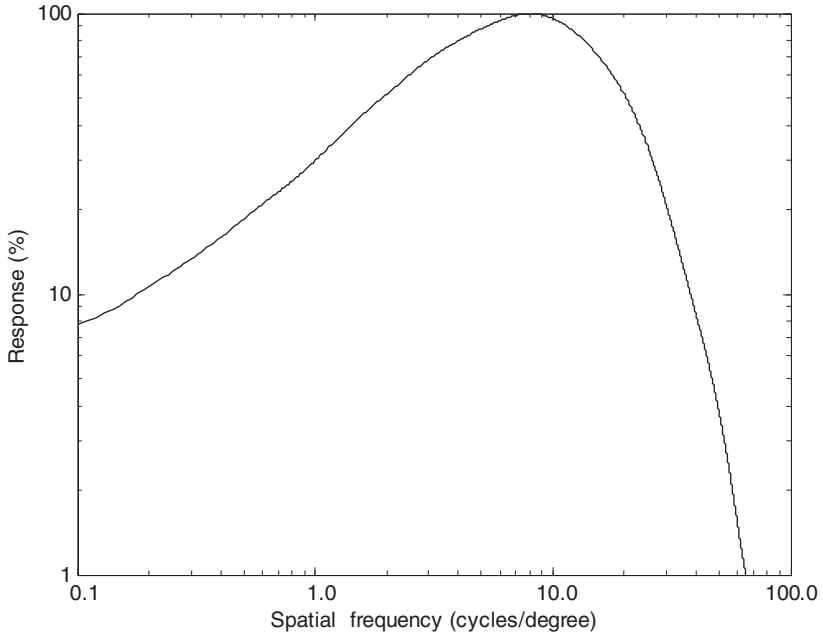


Figure 4.19 Relative spatial frequency response of human visual system.

8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

Figure 4.20 Typical values of weighting matrix $W_{i,j}$.

block of data, and $W_{i,j}$ is the weighting value for this particular transform coefficient.

The weighting value $W_{i,j}$ increases as the horizontal and vertical frequencies increase. An example of a matrix of weighting values is shown in Figure 4.20.

Thus if the quantizer step size is 16 and the value of $c(4,4)$[8] for a particular intrablock is 75 then the value of the quantized DCT coefficient, noting that the appropriate quantizer matrix value is 32, would be calculated as shown.

$$c_q(4,4) = \text{round}\left(\frac{8\times 75}{16\times 32}\right) = \text{round}(1.17) = 1$$

If the coefficient $c(1,1)$ had the same value of the original DCT coefficient and the quantizer step size was unchanged then the quantized DCT coefficient value would be calculated as shown.

$$c_q(1,1) = \text{round}\left(\frac{8\times 75}{16\times 16}\right) = \text{round}(2.34) = 2$$

EXAMPLE 4.6

An 8×8 block of data from a picture is shown in Figure 4.21.

(a) Calculate the two-dimensional DCT of the data.
(b) Quantize the data using a quantizer step size of 8.
(c) Quantize the data using a quantizer step size of 8 using the weighting matrix given in Figure 4.20.

91	42	67	72	83	189	245	241
75	74	171	245	240	227	216	221
50	45	75	65	119	228	245	234
72	93	198	246	239	225	214	222
33	58	75	72	155	242	242	229
75	106	215	248	237	223	216	223

Figure 4.21 Picture data for Example 4.6

[8]The DC DCT coefficient would of course be $c(0,0)$.

(a) The result after a two-dimensional DCT performed in MATLAB after rounding to the nearest integer is shown in Figure 4.22.

1294	–495	–104	0	–22	34	48	7
–66	–13	84	13	–1	30	5	–8
–15	7	26	1	21	24	–1	–4
–35	–7	53	12	–17	–4	–5	1
–10	8	19	–9	6	15	2	–2
–60	–24	71	25	–26	–7	–1	1
–7	19	35	–25	–8	24	11	0
–189	–117	173	99	–76	–31	0	–2

Figure 4.22 Picture data of Figure 4.21 after two-dimensional DCT.

(b) Quantizing using a quantizer step size of 8 and rounding to the nearest integer yields the quantized DCT coefficients shown in Figure 4.23.

161	–61	–13	0	–2	4	6	0
–8	–1	10	1	0	3	0	–1
–1	0	3	0	2	3	0	0
–4	0	6	1	–2	0	0	0
–1	1	2	–1	0	1	0	0
–7	–3	8	3	–3	0	0	0
0	2	4	–3	–1	3	1	0
–23	–14	21	12	–9	–3	0	0

Figure 4.23 Quantized DCT coefficients—no weighting matrix.

(c) Quantizing when the weighting factors used in Figure 4.20 are employed yields the results shown in Figure 4.24.

161	–30	–5	0	0	1	1	0
–4	0	3	0	0	1	0	0
0	0	1	0	0	0	0	0
–1	0	2	0	0	0	0	0
0	0	0	0	0	0	0	0
–2	0	2	0	0	0	0	0
0	0	1	0	0	0	0	0
–7	–4	4	2	–1	0	0	0

Figure 4.24 Quantized DCT coefficients—weighting matrix in Figure 4.20 are employed.

Note that the number of DCT coefficients quantized to zero when the weighting matrix is used (45) is considerably greater than when the quantization matrix is not employed (16). ■

4.4.4. Coding of Nonzero DCT Coefficients

We have now produced an 8×8 array of quantized DCT coefficients many of which are zero. We need to be able to entropy code these coefficients and then transmit them to the receiver. The first step of this process is to scan the two-dimensional array of coefficients into a one-dimensional array. This is achieved by scanning the coefficients in the zig-zag scan order shown in Figure 4.25.

This order ensures that the DC coefficient is scanned first followed by the low-frequency AC coefficients. Higher frequency coefficients are scanned toward the end of the scan. Because there is usually less energy at high-frequencies and also because high-frequency coefficients are often quantized more coarsely than low-frequency coefficients to match the characteristics of the human visual system, it is likely that the last nonzero coefficient will be met well before the end of the scan. As we shall see, the scan process can be terminated after the last nonzero coefficient.

After the zig-zag scanning, each nonzero coefficient is grouped with a the run of zero coefficients that proceeds it to form a (run,level) pair. Consider the quantized DCT coefficients shown in Figure 4.26.

After zig-zag scanning in accordance with Figure 4.25, the one-dimensional array is shown in Figure 4.27.

The resulting run-coefficient pairs are as shown in Figure 4.28 with all of the remaining coefficients being zero.

Each run–coefficient pair is not equally likely and so a saving in the number of bits required to transmit the information occurs if the run–coefficient pairs are encoded using a Huffman code. A special Huffman code word is used to indicate that the last nonzero coefficient in a block has been transmitted and is called the end of

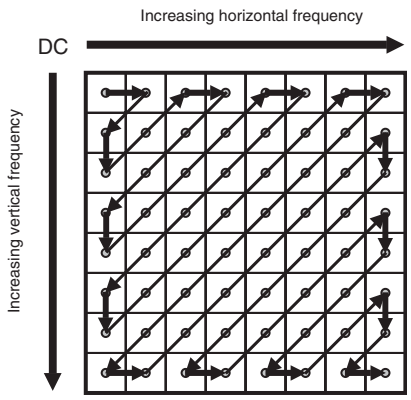


Figure 4.25 Zig-zag scan order of block of quantized transform coefficients.

$$\begin{bmatrix} +38 & 0 & +4 & -2 & 0 & 0 & 0 & 0 \\ -5 & +1 & 0 & +1 & 0 & 0 & 0 & 0 \\ +7 & +2 & -2 & +1 & 0 & 0 & 0 & 0 \\ +2 & -1 & +1 & -1 & 0 & 0 & 0 & 0 \\ -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 4.26 Block of quantized DCT coefficients.

+38, 0, –5, +7, +1, +4, –2, 0, +2, +2, –2, –1, –2, +1, 0, 0, 0, +1, +1, 0, 0, 0, 0, 0, –1, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0

Figure 4.27 Coefficients for Figure 4.26 after zig-zag scanning.

(0, +38) (1, –5) (0, +7) (0, +1) (0, +4) (0, –2) (1, +2) (0, +2) (0, –2) (0, –1) (0, –2)

(0, +1) (3, +1) (0, +1) (5, –1)

Figure 4.28 Coefficients of Figure 4.27 after coding into (run,coefficient) pairs.

Code word$_{(0, +38)}$, Code word$_{(1, -5)}$, Code word$_{(0, +7)}$, Code word$_{(0, +1)}$, Code word$_{(0, +4)}$,

Code word$_{(0, -2)}$, Code word$_{(1, +2)}$, Code word$_{(0, +2)}$, Code word$_{(0, -2)}$, Code word$_{(0, -1)}$,

Code word$_{(0, -2)}$, Code word$_{(0, +1)}$, Code word$_{(3, +1)}$, Code word$_{(0, +1)}$, Code word$_{(5, -1)}$,

Code word$_{EOB}$.

Figure 4.29 Huffman code words used to represent the quantized DCT coefficients of Figure 4.26.

block (EOB) code word. Because the EOB code word is sent with every transmitted block, it occurs quite commonly and so is able to be represented by a short Huffman code word. For the (run,level) pairs given above, the transmitted code words would be as given in Figure 4.29.

The Huffman coding tables for the set of possible (run,level) pairs have been developed by standards bodies and are based on the statistics of typical sequences of video material.

4.5. MOTION-COMPENSATED DCT ENCODERS AND DECODERS

While we have so far only considered the application of the DCT to original pictures, it can also be used to code the prediction difference after motion-compensated

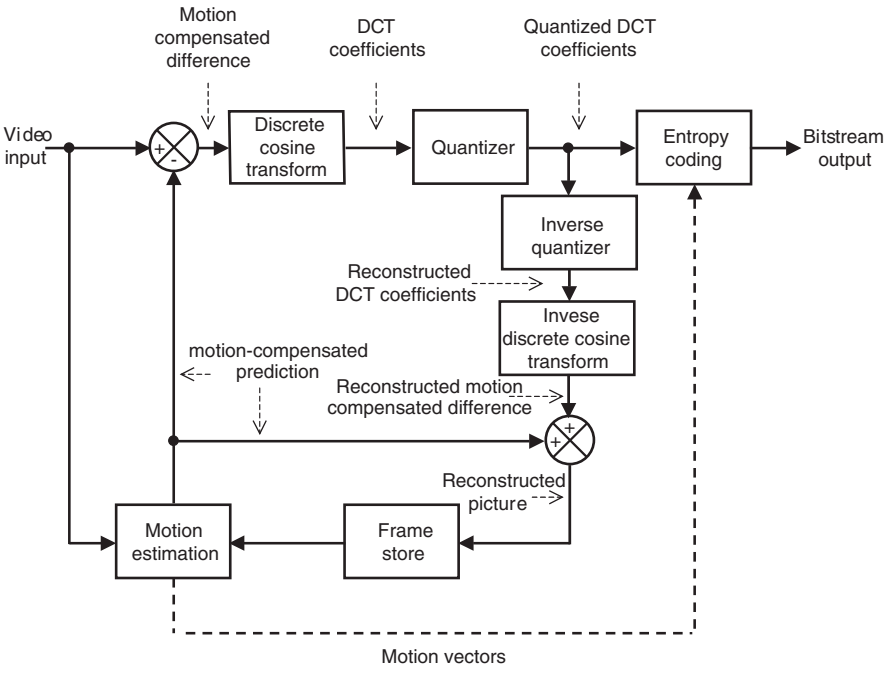


Figure 4.30 Motion-compensated discrete cosine transform encoder.

prediction. Figure 4.30 shows the block diagram of a motion-compensated DCT encoder. The video input has the motion-compensated prediction subtracted from it. The motion-compensated prediction difference is then processed with a two-dimensional DCT prior to quantization. Finally, the quantized DCT coefficients together with the relevant motion vectors are entropy coded and transmitted. The feedback loop of the encoder is equivalent to a decoder and consists of an inverse quantizer[9] followed by an inverse two-dimensional DCT. This produces a reconstruction of the motion-compensated prediction difference. The motion-compensated prediction is then added to the reconstruction of the motion-compensation prediction difference to form the reconstructed picture, which is stored in a frame store for use in the prediction of a subsequent picture.

The corresponding decoder is shown in Figure 4.31. Apart from the initial entropy decoding stage to produce the transform coefficients and motion vectors, this is identical to the feedback loop of the encoder.

Motion-compensated DCT encoders and decoders are the key coding tools of the MPEG-2 video compression standard that is used for digital television broadcasting. Refinements and improvements introduced during the standardization process significantly enhance the performance of the basic architecture. We will consider this topic in considerable detail in Chapter 6.

[9]However, remember that quantization is a lossy process and so cannot be perfectly reversed.