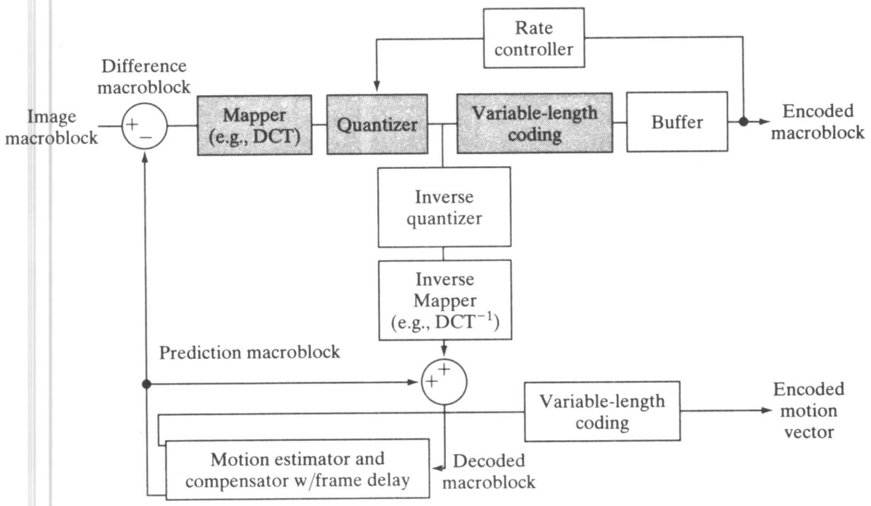


**FIGURE 8.39** A typical motion compensated video encoder.

or the difference between a conventional macroblock and a prediction of it based on previous and/or subsequent video frames (for P- and B-frames). The encoder includes an *inverse quantizer* and *inverse mapper* (e.g., inverse DCT) so that its predictions match those of the complementary decoder. Also, it is designed to produce compressed bit streams that match the capacity of the intended video channel. To accomplish this, the quantization parameters are adjusted by a *rate controller* as a function of the occupancy of an output *buffer*. As the buffer becomes fuller, the quantization is made coarser, so that fewer bits stream into the buffer.

Quantization as defined earlier in the chapter is irreversible. The "inverse quantizer" in Fig. 8.39 does not prevent information loss.

**EXAMPLE 8.22:** Video compression example.

■ We conclude our discussion of motion compensated predictive coding with an example illustrating the kind of compression that is possible with modern video compression methods. Figure 8.40 shows fifteen frames of a 1 minute HD (1280 × 720) full-color NASA video, parts of which have been used throughout this section. Although the images shown are monochrome, the video is a sequence of 1,829 full-color frames. Note that there are a variety of scenes, a great deal of motion, and multiple fade effects. For example, the video opens with a 150 frame fade-in from black, which includes frames 21 and 44 in Fig. 8.40, and concludes with a fade sequence containing frames 1595, 1609, and 1652 in Fig. 8.40, followed by a final fade to black. There are also several abrupt scene changes, like the change involving frames 1303 and 1304 in Fig. 8.40.

See the book Web site for the NASA video segment used in this section.

An H.264 compressed version of the NASA video stored as a Quicktime file (see Table 8.4) requires 44.56 MB of storage—plus another 1.39 MB for the associated audio. The video quality is excellent. About 5 GB of data would be needed to store the video frames as uncompressed full-color images. It should be noted that the video contains sequences involving both rotation and scale change (e.g., the sequence including frames 959, 1023, and 1088 in Fig. 840). The discussion in this section, however, has been limited to translation alone. ■

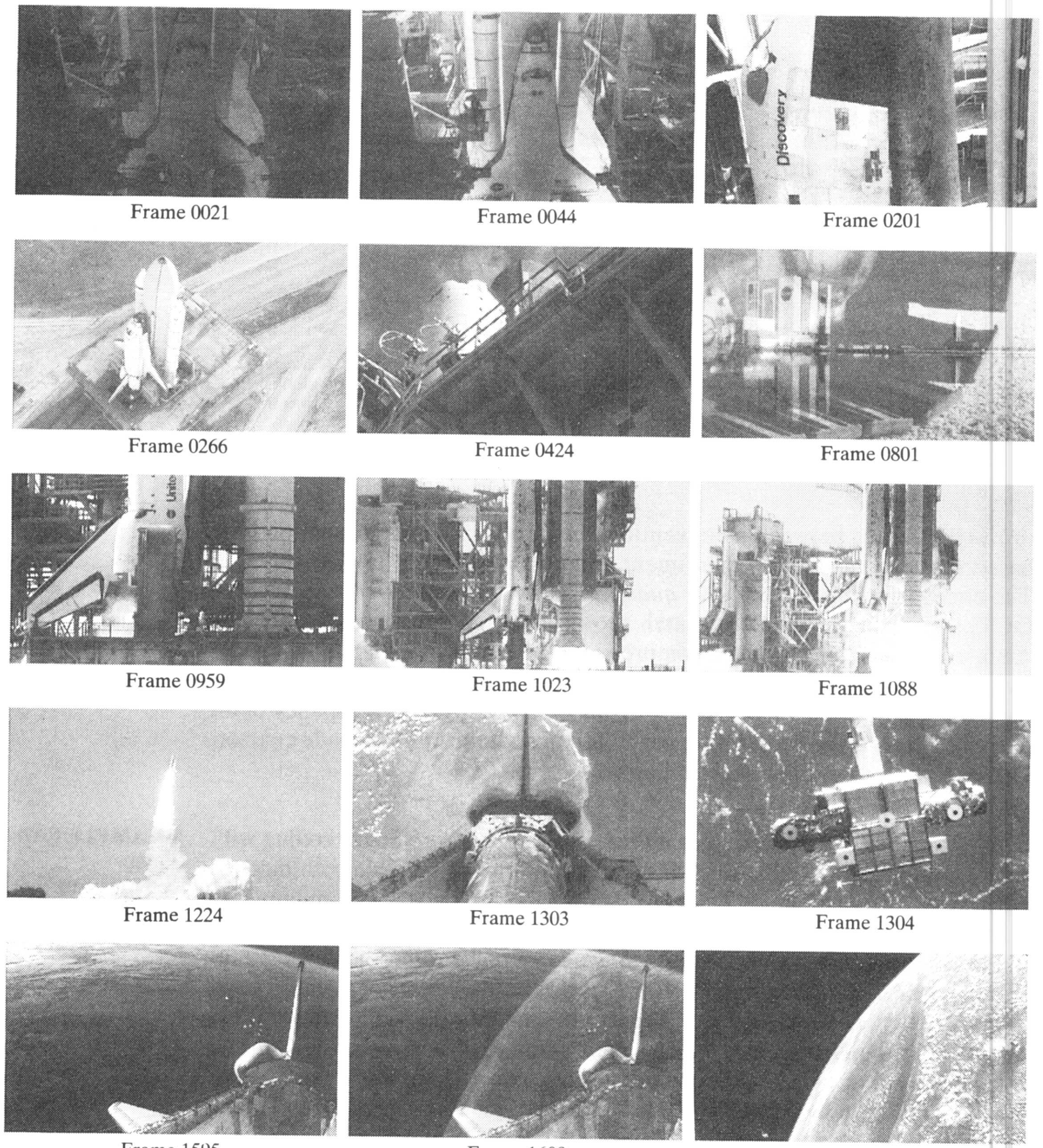
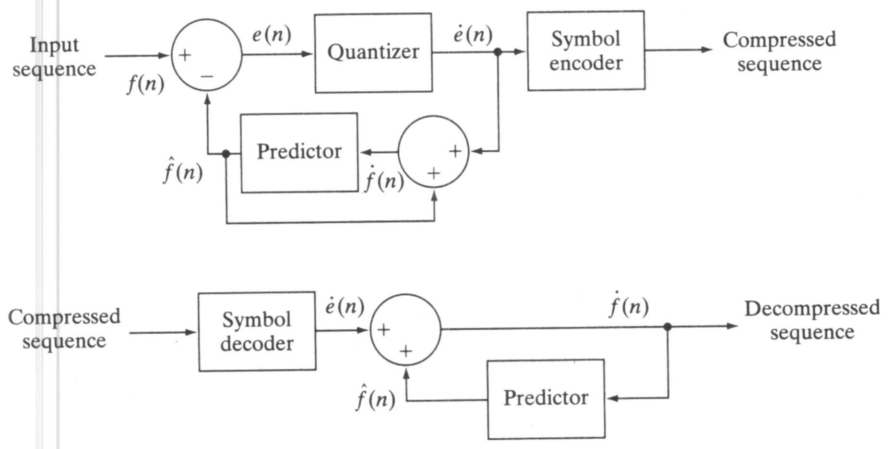


**FIGURE 8.40** Fifteen frames from an 1829-frame, 1-minute NASA video. The original video is in HD full color. (Courtesy of NASA.)

### Lossy predictive coding

In this section, we add a quantizer to the lossless predictive coding model introduced earlier and examine the trade-off between reconstruction accuracy and compression performance within the context of spatial predictors. As Fig. 8.41 shows, the quantizer, which replaces the nearest integer function of the error-free



a
b

**FIGURE 8.41** A lossless predictive coding model: (a) encoder; (b) decoder.

encoder, is inserted between the symbol encoder and the point at which the prediction error is formed. It maps the prediction error into a limited range of outputs, denoted $\dot{e}(n)$, which establish the amount of compression and distortion that occurs.

In order to accommodate the insertion of the quantization step, the error-free encoder of Fig. 8.33(a) must be altered so that the predictions generated by the encoder and decoder are equivalent. As Fig. 8.41(a) shows, this is accomplished by placing the lossy encoder's predictor within a feedback loop, where its input, denoted $\dot{f}(n)$, is generated as a function of past predictions and the corresponding quantized errors. That is,

$$\dot{f}(n) = \dot{e}(n) + \hat{f}(n) \tag{8.2-38}$$

where $\hat{f}(n)$ is as defined earlier. This closed loop configuration prevents error buildup at the decoder's output. Note in Fig. 8.41(b) that the output of the decoder is given also by Eq. (8.2-38).

**EXAMPLE 8.23:** Delta modulation.

■ *Delta modulation* (DM) is a simple but well-known form of lossy predictive coding in which the predictor and quantizer are defined as

$$\hat{f}(n) = \alpha \dot{f}(n-1) \tag{8.2-39}$$

and

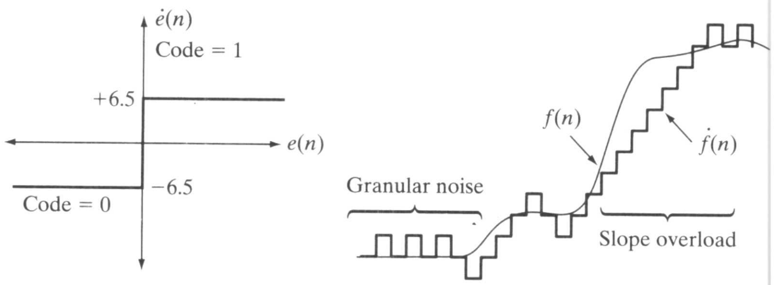
$$\dot{e}(n) = \begin{cases} +\zeta & \text{for } e(n) > 0 \\ -\zeta & \text{otherwise} \end{cases} \tag{8.2-40}$$

where $\alpha$ is a prediction coefficient (normally less than 1) and $\zeta$ is a positive constant. The output of the quantizer, $\dot{e}(n)$, can be represented by a single bit [Fig. 8.42(a)], so the symbol encoder of Fig. 8.41(a) can utilize a 1-bit fixed-length code. The resulting DM code rate is 1 bit/pixel.

Figure 8.42(c) illustrates the mechanics of the delta modulation process, where the calculations needed to compress and reconstruct input sequence

a b
c

**FIGURE 8.42** An example of delta modulation.



| Input | | Encoder | | | | Decoder | | Error |
|---|---|---|---|---|---|---|---|---|
| $n$ | $f(n)$ | $\hat{f}(n)$ | $e(n)$ | $\dot{e}(n)$ | $\dot{f}(n)$ | $\hat{f}(n)$ | $\dot{f}(n)$ | $f(n) - \dot{f}(n)$ |
| 0 | 14 | — | — | — | 14.0 | — | 14.0 | 0.0 |
| 1 | 15 | 14.0 | 1.0 | 6.5 | 20.5 | 14.0 | 20.5 | −5.5 |
| 2 | 14 | 20.5 | −6.5 | −6.5 | 14.0 | 20.5 | 14.0 | 0.0 |
| 3 | 15 | 14.0 | 1.0 | 6.5 | 20.5 | 14.0 | 20.5 | −5.5 |
| . | . | . | . | . | . | . | . | . |
| . | . | . | . | . | . | . | . | . |
| 14 | 29 | 20.5 | 8.5 | 6.5 | 27.0 | 20.5 | 27.0 | 2.0 |
| 15 | 37 | 27.0 | 10.0 | 6.5 | 33.5 | 27.0 | 33.5 | 3.5 |
| 16 | 47 | 33.5 | 13.5 | 6.5 | 40.0 | 33.5 | 40.0 | 7.0 |
| 17 | 62 | 40.0 | 22.0 | 6.5 | 46.5 | 40.0 | 46.5 | 15.5 |
| 18 | 75 | 46.5 | 28.5 | 6.5 | 53.0 | 46.5 | 53.0 | 22.0 |
| 19 | 77 | 53.0 | 24.0 | 6.5 | 59.6 | 53.0 | 59.6 | 17.5 |
| . | . | . | . | . | . | . | . | . |
| . | . | . | . | . | . | . | . | . |

{14, 15, 14, 15, 13, 15, 15, 14, 20, 26, 27, 28, 27, 27, 29, 37, 47, 62, 75, 77, 78, 79, 80, 81, 81, 82, 82} with $\alpha = 1$ and $\zeta = 6.5$ are tabulated. The process begins with the error-free transfer of the first input sample to the decoder. With the initial condition $\dot{f}(0) = f(0) = 14$ established at both the encoder and decoder, the remaining outputs can be computed by repeatedly evaluating Eqs. (8.2-39), (8.2-30), (8.2-40), and (8.2-38). Thus, when $n = 1$, for example, $\hat{f}(1) = (1)(14) = 14$, $e(1) = 15 - 14 = 1$, $\dot{e}(1) = +6.5$ (because $e(1) > 0$), $\dot{f}(1) = 6.4 + 14 = 20.5$, and the resulting reconstruction error is $(15 - 20.5)$, or $-5.5$.

Figure 8.42(b) shows graphically the tabulated data in Fig. 8.42(c). Both the input and completely decoded output $[f(n)$ and $\dot{f}(n)]$ are shown. Note that in the rapidly changing area from $n = 14$ to 19, where $\zeta$ was too small to represent the input's largest changes, a distortion known as *slope overload* occurs. Moreover, when $\zeta$ was too large to represent the input's smallest changes, as in the relatively smooth region from $n = 0$ to $n = 7$, *granular noise* appears. In images, these two phenomena lead to blurred object edges and grainy or noisy surfaces (that is, distorted smooth areas). ■

The distortions noted in the preceding example are common to all forms of lossy predictive coding. The severity of these distortions depends on a complex set of interactions between the quantization and prediction methods employed. Despite these interactions, the predictor normally is designed with the assumption of no quantization error, and the quantizer is designed to minimize its own error. That is, the predictor and quantizer are designed independently of each other.

### Optimal predictors

In many predictive coding applications, the predictor is chosen to minimize the encoder's mean-square prediction error[†]

$$E\{e^2(n)\} = E\{[f(n) - \hat{f}(n)]^2\} \tag{8.2-41}$$

subject to the constraint that

$$\dot{f}(n) = \dot{e}(n) + \hat{f}(n) \approx e(n) + \hat{f}(n) = f(n) \tag{8.2-42}$$

and

$$\hat{f}(n) = \sum_{i=1}^{m} \alpha_i f(n-i) \tag{8.2-43}$$

That is, the optimization criterion is minimal mean-square prediction error, the quantization error is assumed to be negligible [$\dot{e}(n) \approx e(n)$] and the prediction is constrained to a linear combination of $m$ previous samples.[‡] These restrictions are not essential, but they simplify the analysis considerably and, at the same time, decrease the computational complexity of the predictor. The resulting predictive coding approach is referred to as *differential pulse code modulation* (DPCM).

Under these conditions, the optimal predictor design problem is reduced to the relatively straightforward exercise of selecting the $m$ prediction coefficients that minimize the expression

$$E\{e^2(n)\} = E\left\{\left[f(n) - \sum_{i=1}^{m} \alpha_i f(n-i)\right]^2\right\} \tag{8.2-44}$$

Differentiating Eq. (8.2-44) with respect to each coefficient, equating the derivatives to zero, and solving the resulting set of simultaneous equations under the assumption that $f(n)$ has mean zero and variance $\sigma^2$ yields

$$\boldsymbol{\alpha} = \mathbf{R}^{-1}\mathbf{r} \tag{8.2-45}$$

where $\mathbf{R}^{-1}$ is the inverse of the $m \times m$ autocorrelation matrix

$$\mathbf{R} = \begin{bmatrix} E\{f(n-1)f(n-1)\} & E\{f(n-1)f(n-2)\} & \cdots & E\{f(n-1)f(n-m)\} \\ E\{f(n-2)f(n-1)\} & \vdots & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ E\{f(n-m)f(n-1)\} & E\{f(n-m)f(n-2)\} & \cdots & E\{f(n-m)f(n-m)\} \end{bmatrix} \tag{8.2-46}$$

[†]The notation $E\{\cdot\}$ denotes the statistical expectation operator.

[‡]In general, the optimal predictor for a non-Gaussian sequence is a nonlinear function of the samples used to form the estimate.

and $\mathbf{r}$ and $\boldsymbol{\alpha}$ are the $m$-element vectors

$$\mathbf{r} = \begin{bmatrix} E\{f(n)f(n-1)\} \\ E\{f(n)f(n-2)\} \\ \vdots \\ E\{f(n)f(n-m)\} \end{bmatrix} \quad \text{and } \boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix} \tag{8.2-47}$$

Thus for any input sequence, the coefficients that minimize Eq. (8.2-44) can be determined via a series of elementary matrix operations. Moreover, the coefficients depend only on the autocorrelations of the samples in the original sequence. The variance of the prediction error that results from the use of these optimal coefficients is

$$\sigma_e^2 = \sigma^2 - \boldsymbol{\alpha}^T\mathbf{r} = \sigma^2 - \sum_{i=1}^{m} E\{f(n)f(n-i)\}\alpha_i \tag{8.2-48}$$

Although the mechanics of evaluating Eq. (8.2-45) are quite simple, computation of the autocorrelations needed to form $\mathbf{R}$ and $\mathbf{r}$ is so difficult in practice that *local* predictions (those in which the prediction coefficients are computed for each input sequence) are almost never used. In most cases, a set of *global* coefficients is computed by assuming a simple input model and substituting the corresponding autocorrelations into Eqs. (8.2-46) and (8.2-47). For instance, when a 2-D Markov image source (see Section 8.1.4) with separable autocorrelation function

$$E\{f(x,y)f(x-i,y-j)\} = \sigma^2\rho_v^i\rho_h^j \tag{8.2-49}$$

and generalized fourth-order linear predictor

$$\hat{f}(x,y) = \alpha_1 f(x,y-1) + \alpha_2 f(x-1,y-1) + \alpha_3 f(x-1,y) + \alpha_4 f(x-1,y+1) \tag{8.2-50}$$

are assumed, the resulting optimal coefficients (Jain [1989]) are

$$\alpha_1 = \rho_h \quad \alpha_2 = -\rho_v\rho_h \quad \alpha_3 = \rho_v \quad \alpha_4 = 0 \tag{8.2-51}$$

where $\rho_h$ and $\rho_v$ are the horizontal and vertical correlation coefficients, respectively, of the image under consideration.

Finally, the sum of the prediction coefficients in Eq. (8.2-43) normally is required to be less than or equal to one. That is,

$$\sum_{i=1}^{m} \alpha_i \leq 1 \tag{8.2-52}$$

This restriction is made to ensure that the output of the predictor falls within the allowed range of the input and to reduce the impact of transmission noise [which generally is seen as horizontal streaks in reconstructed images when the input to Fig. 8.41(a) is an image]. Reducing the DPCM decoder's susceptibility to input noise is important, because a single error (under the right circumstances) can propagate to all future outputs. That is, the decoder's output may become unstable. By further restricting Eq. (8.2-52) to be strictly less than 1 confines the impact of an input error to a small number of outputs.

**EXAMPLE 8.24:** Comparison of prediction techniques.

■ Consider the prediction error that results from DPCM coding the monochrome image of Fig. 8.9(a) under the assumption of zero quantization error and with each of four predictors:

$$\hat{f}(x, y) = 0.97f(x, y - 1) \tag{8.2-53}$$

$$\hat{f}(x, y) = 0.5f(x, y - 1) + 0.5f(x - 1, y) \tag{8.2-54}$$

$$\hat{f}(x, y) = 0.75f(x, y - 1) + 0.75f(x - 1, y) - 0.5f(x - 1, y - 1) \tag{8.2-55}$$

$$\hat{f}(x, y) = \begin{cases} 0.97f(x, y - 1) & \text{if } \Delta h \leq \Delta v \\ 0.97f(x - 1, y) & \text{otherwise} \end{cases} \tag{8.2-56}$$

where $\Delta h = |f(x - 1, y) - f(x - 1, y - 1)|$ and $\Delta v = |f(x, y - 1) - f(x - 1, y - 1)|$ denote the horizontal and vertical gradients at point $(x, y)$. Equations (8.2-53) through (8.2-56) define a relatively robust set of $\alpha_i$ that provide satisfactory performance over a wide range of images. The adaptive predictor of Eq. (8.2-56) is designed to improve edge rendition by computing a local measure of the directional properties of an image ($\Delta h$ and $\Delta v$) and selecting a predictor specifically tailored to the measured behavior.

Figures 8.43(a) through (d) show the prediction error images that result from using the predictors of Eqs. (8.2-53) through (8.2-56). Note that the

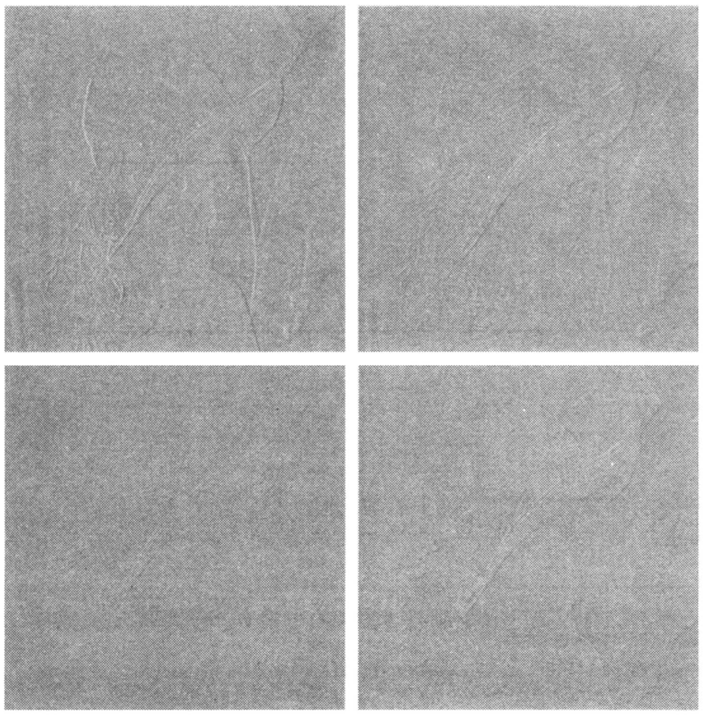

a b
c d

**FIGURE 8.43** A comparison of four linear prediction techniques.

visually perceptible error decreases as the order of the predictor increases.[†] The standard deviations of the prediction errors follow a similar pattern. They are 11.1, 9.8, 9.1, and 9.7 intensity levels, respectively. ■

## Optimal quantization

The staircase quantization function $t = q(s)$ in Fig. 8.44 is an odd function of $s$ [that is, $q(-s) = -q(s)$] that can be described completely by the $L/2$ values of $s_i$ and $t_i$ shown in the first quadrant of the graph. These break points define function discontinuities and are called the *decision* and *reconstruction levels* of the quantizer. As a matter of convention, $s$ is considered to be mapped to $t_i$ if it lies in the half-open interval $(s_i, s_{i+1}]$.

The quantizer design problem is to select the best $s_i$ and $t_i$ for a particular optimization criterion and input probability density function $p(s)$. If the optimization criterion, which could be either a statistical or psychovisual measure,[‡] is the minimization of the mean-square quantization error (that is, $E\{(s_i - t_i)^2\}$) and $p(s)$ is an even function, the conditions for minimal error (Max [1960]) are

$$\int_{s_{i-1}}^{s_i} (s - t_i)p(s)\,ds \quad i = 1, 2, \ldots, \frac{L}{2} \tag{8.2-57}$$

$$s_i = \begin{cases} 0 & i = 0 \\ \dfrac{t_i + t_{i+1}}{2} & i = 1, 2, \ldots, \dfrac{L}{2} - 1 \\ \infty & i = \dfrac{L}{2} \end{cases} \tag{8.2-58}$$

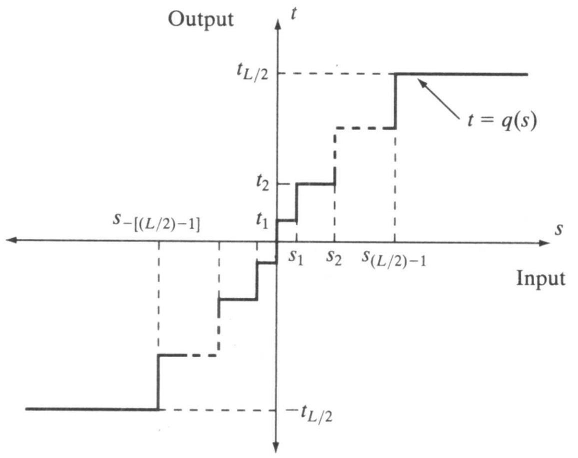


**FIGURE 8.44** A typical quantization function.

[†]Predictors that use more than three or four previous pixels provide little compression gain for the added predictor complexity (Habibi [1971]).

[‡]See Netravali [1977] and Limb and Rubinstein [1978] for more on psychovisual measures.

and

$$s_{-i} = -s_i \quad t_{-i} = -t_i \tag{8.2-59}$$

Equation (8.2-57) indicates that the reconstruction levels are the centroids of the areas under $p(s)$ over the specified decision intervals, whereas Eq. (8.2-58) indicates that the decision levels are halfway between the reconstruction levels. Equation (8.2-59) is a consequence of the fact that $q$ is an odd function. For any $L$, the $s_i$ and $t_i$ that satisfy Eqs. (8.2-57) through (8.2-59) are optimal in the mean-square error sense; the corresponding quantizer is called an $L$-level *Lloyd-Max* quantizer.

Table 8.12 lists the 2-, 4-, and 8-level Lloyd-Max decision and reconstruction levels for a unit variance Laplacian probability density function [see Eq. (8.2-35)]. Because obtaining an explicit or closed-form solution to Eqs. (8.2-57) through (8.2-59) for most nontrivial $p(s)$ is difficult, these values were generated numerically (Paez and Glisson [1972]). The three quantizers shown provide fixed output rates of 1, 2, and 3 bits/pixel, respectively. As Table 8.12 was constructed for a unit variance distribution, the reconstruction and decision levels for the case of $\sigma \neq 1$ are obtained by multiplying the tabulated values by the standard deviation of the probability density function under consideration. The final row of the table lists the step size, $\theta$, that simultaneously satisfies Eqs. (8.2-57) through (8.5-59) *and* the additional constraint that

$$t_i - t_{i-1} = s_i - s_{i-1} = \theta \tag{8.2-60}$$

If a symbol encoder that utilizes a variable-length code is used in the general lossy predictive encoder of Fig. 8.41(a), an *optimum uniform quantizer* of step size $\theta$ will provide a lower code rate (for a Laplacian PDF) than a fixed-length coded Lloyd-Max quantizer with the same output fidelity (O'Neil [1971]).

Although the Lloyd-Max and optimum uniform quantizers are not adaptive, much can be gained from adjusting the quantization levels based on the local behavior of an image. In theory, slowly changing regions can be finely quantized, while the rapidly changing areas are quantized more coarsely. This approach simultaneously reduces both granular noise and slope overload, while requiring only a minimal increase in code rate. The trade-off is increased quantizer complexity.

| Levels | 2 | | 4 | | 8 | |
|---|---|---|---|---|---|---|
| $i$ | $s_i$ | $t_i$ | $s_i$ | $t_i$ | $s_i$ | $t_i$ |
| 1 | $\infty$ | 0.707 | 1.102 | 0.395 | 0.504 | 0.222 |
| 2 | | | $\infty$ | 1.810 | 1.181 | 0.785 |
| 3 | | | | | 2.285 | 1.576 |
| 4 | | | | | $\infty$ | 2.994 |
| $\theta$ | 1.414 | | 1.087 | | 0.731 | |

**TABLE 8.12** Lloyd-Max quantizers for a Laplacian probability density function of unit variance.

### 8.2.10 Wavelet Coding

With reference to Tables 8.3 and 8.4, wavelet coding is used in the

- JPEG-2000

compression standard.

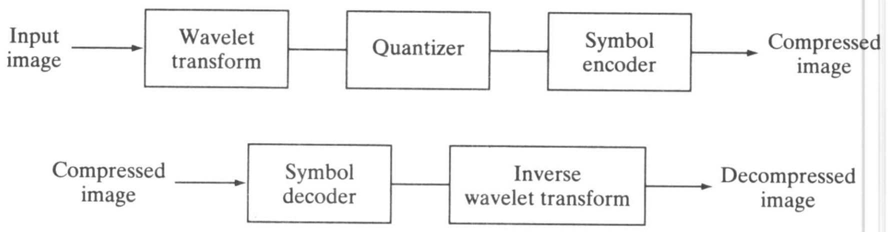
As with the transform coding techniques of Section 8.2.8, wavelet coding is based on the idea that the coefficients of a transform that decorrelates the pixels of an image can be coded more efficiently than the original pixels themselves. If the basis functions of the transform—in this case wavelets—pack most of the important visual information into a small number of coefficients, the remaining coefficients can be quantized coarsely or truncated to zero with little image distortion.

Figure 8.45 shows a typical wavelet coding system. To encode a $2^J \times 2^J$ image, an analyzing wavelet, $\psi$, and minimum decomposition level, $J - P$, are selected and used to compute the discrete wavelet transform of the image. If the wavelet has a complementary scaling function $\varphi$, the fast wavelet transform (see Sections 7.4 and 7.5) can be used. In either case, the computed transform converts a large portion of the original image to horizontal, vertical, and diagonal decomposition coefficients with zero mean and Laplacian-like probabilities. Recall the image of Fig. 7.1 and the dramatically simpler statistics of its wavelet transform in Fig. 7.10(a). Because many of the computed coefficients carry little visual information, they can be quantized and coded to minimize intercoefficient and coding redundancy. Moreover, the quantization can be adapted to exploit any positional correlation across the $P$ decomposition levels. One or more lossless coding methods, like run-length, Huffman, arithmetic, and bit-plane coding, can be incorporated into the final symbol coding step. Decoding is accomplished by inverting the encoding operations—with the exception of quantization, which cannot be reversed exactly.

The principal difference between the wavelet-based system of Fig. 8.45 and the transform coding system of Fig. 8.21 is the omission of the subimage processing stages of the transform coder. Because wavelet transforms are both computationally efficient and inherently local (i.e., their basis functions are limited in duration), subdivision of the original image is unnecessary. As you will see later in this section, the removal of the subdivision step eliminates the blocking artifact that characterizes DCT-based approximations at high compression ratios.

#### Wavelet selection

The wavelets chosen as the basis of the forward and inverse transforms in Fig. 8.45 affect all aspects of wavelet coding system design and performance. They impact directly the computational complexity of the transforms and, less directly, the system's ability to compress and reconstruct



a
b

**FIGURE 8.45** A wavelet coding system: (a) encoder; (b) decoder.

images of acceptable error. When the transforming wavelet has a companion scaling function, the transformation can be implemented as a sequence of digital filtering operations, with the number of filter taps equal to the number of nonzero wavelet and scaling vector coefficients. The ability of the wavelet to pack information into a small number of transform coefficients determines its compression and reconstruction performance.

In digital filtering, each filter tap multiplies a filter coefficient by a delayed version of the signal being filtered.

The most widely used expansion functions for wavelet-based compression are the Daubechies wavelets and biorthogonal wavelets. The latter allow useful analysis properties, like the number of zero moments (see Section 7.5), to be incorporated into the decomposition filters, while important synthesis properties, like smoothness of reconstruction, are built into the reconstruction filters.

**EXAMPLE 8.25:** Wavelet bases in wavelet coding.

■ Figure 8.46 contains four discrete wavelet transforms of Fig. 8.9(a). Haar wavelets, the simplest and only discontinuous wavelets considered in this example, were used as the expansion or basis functions in Fig. 8.46(a). Daubechies wavelets, among the most popular imaging wavelets, were used in Fig. 8.46(b),

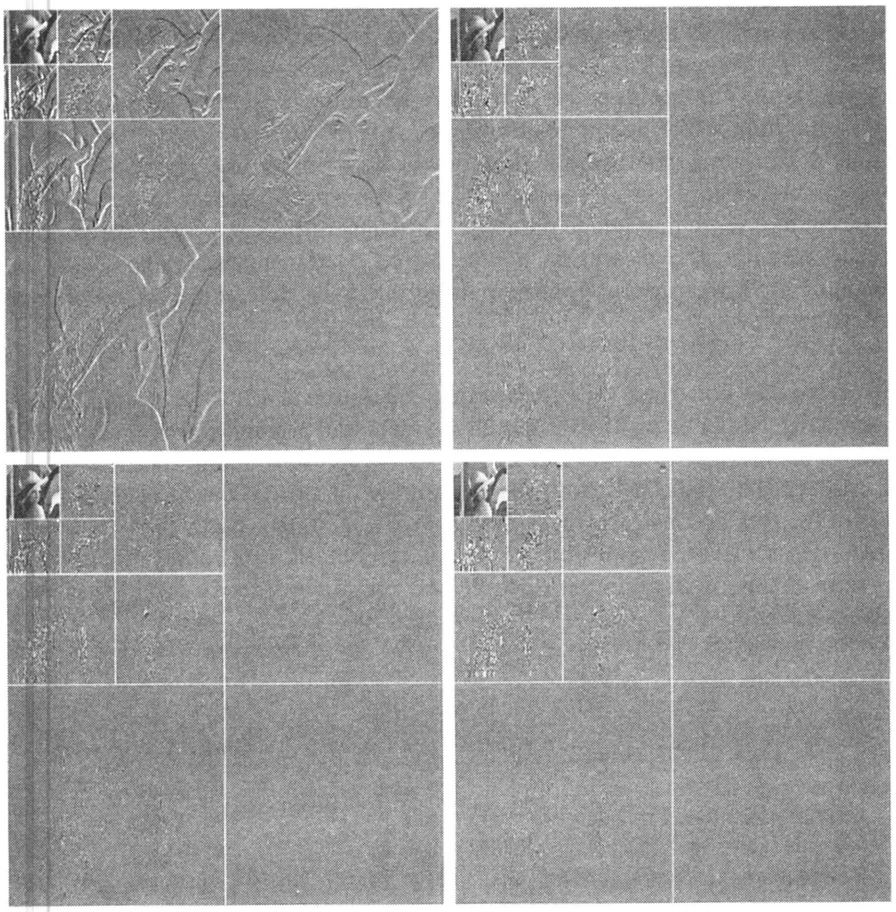

a b
c d

**FIGURE 8.46** Three-scale wavelet transforms of Fig. 8.9(a) with respect to (a) Haar wavelets, (b) Daubechies wavelets, (c) symlets, and (d) Cohen-Daubechies Feauveau biorthogonal wavelets.

and symlets, which are an extension of the Daubechies wavelets with increased symmetry, were used in Fig. 8.46(c). The Cohen-Daubechies-Feauveau wavelets that were employed in Fig. 8.46(d) are included to illustrate the capabilities of biorthogonal wavelets. As in previous results of this type, all detail coefficients were scaled to make the underlying structure more visible—with intensity 128 corresponding to coefficient value 0.

DWT detail coefficients are discussed in Section 7.3.2.

As you can see in Table 8.13, the number of operations involved in the computation of the transforms in Fig. 8.46 increases from 4 to 28 multiplications and additions per coefficient (for each decomposition level) as you move from Fig. 8.46(a) to (d). All four transforms were computed using a fast wavelet transform (i.e., filter bank) formulation. Note that as the computational complexity (i.e., the number of filter taps) increases, the information packing performance does as well. When Haar wavelets are employed and the detail coefficients below 1.5 are truncated to zero, 33.8% of the total transform is zeroed. With the more complex biorthogonal wavelets, the number of zeroed coefficients rises to 42.1%, increasing the potential compression by almost 10%. ■

## Decomposition level selection

Another factor affecting wavelet coding computational complexity and reconstruction error is the number of transform decomposition levels. Because a $P$-scale fast wavelet transform involves $P$-filter bank iterations, the number of operations in the computation of the forward and inverse transforms increases with the number of decomposition levels. Moreover, quantizing the increasingly lower-scale coefficients that result with more decomposition levels affects increasingly larger areas of the reconstructed image. In many applications, like searching image databases or transmitting images for progressive reconstruction, the resolution of the stored or transmitted images and the scale of the lowest useful approximations normally determine the number of transform levels.

**EXAMPLE 8.26:** Decomposition levels in wavelet coding.

■ Table 8.14 illustrates the effect of decomposition level selection on the coding of Fig. 8.9(a) using biorthogonal wavelets and a fixed global threshold of 25. As in the previous wavelet coding example, only detail coefficients are truncated. The table lists both the percentage of zeroed coefficients and the resulting rms reconstruction errors from Eq. (8.1-10). Note that the initial decompositions are responsible for the majority of the data compression. There is little change in the number of truncated coefficients above three decomposition levels. ■

**TABLE 8.13** Wavelet transform filter taps and zeroed coefficients when truncating the transforms in Fig. 8.46 below 1.5.

| Wavelet | Filter Taps (Scaling + Wavelet) | Zeroed Coefficients |
|---|---|---|
| Haar (see Ex. 7.10) | 2 + 2 | 33.8% |
| Daubechies (see Fig. 7.8) | 8 + 8 | 40.9% |
| Symlet (see Fig. 7.26) | 8 + 8 | 41.2% |
| Biorthogonal (see Fig. 7.39) | 17 + 11 | 42.1% |

| Decomposition Level (Scales or Filter Bank Iterations) | Approximation Coefficient Image | Truncated Coefficients (%) | Reconstruction Error (rms) |
|---|---|---|---|
| 1 | 256 × 256 | 74.7% | 3.27 |
| 2 | 128 × 128 | 91.7% | 4.23 |
| 3 | 64 × 64 | 95.1% | 4.54 |
| 4 | 32 × 32 | 95.6% | 4.61 |
| 5 | 16 × 16 | 95.5% | 4.63 |

**TABLE 8.14** Decomposition level impact on wavelet coding the 512 × 512 image of Fig. 8.9(a).

### Quantizer design

The most important factor affecting wavelet coding compression and reconstruction error is coefficient quantization. Although the most widely used quantizers are uniform, the effectiveness of the quantization can be improved significantly by (1) introducing a larger quantization interval around zero, called a *dead zone*, or (2) adapting the size of the quantization interval from scale to scale. In either case, the selected quantization intervals must be transmitted to the decoder with the encoded image bit stream. The intervals themselves may be determined heuristically or computed automatically based on the image being compressed. For example, a global coefficient threshold could be computed as the median of the absolute values of the first-level detail coefficients or as a function of the number of zeroes that are truncated and the amount of energy that is retained in the reconstructed image.

One measure of the energy of a digital signal is the sum of the squared samples.

**EXAMPLE 8.27:** Dead zone interval selection in wavelet coding.

■ Figure 8.47 illustrates the impact of dead zone interval size on the percentage of truncated detail coefficients for a three-scale biorthogonal wavelet-based encoding of Fig. 8.9(a). As the size of the dead zone increases, the number of truncated coefficients does as well. Above the knee of the curve (i.e., beyond 5), there is little gain. This is due to the fact that the histogram of the detail coefficients is highly peaked around zero (see, for example, Fig. 7.10).

The rms reconstruction errors corresponding to the dead zone thresholds in Fig. 8.47 increase from 0 to 1.94 intensity levels at a threshold of 5 and to 3.83 intensity levels for a threshold of 18, where the number of zeroes reaches 93.85%. If every detail coefficient were eliminated, that percentage would increase to about 97.92% (about 4%), but the reconstruction error would grow to 12.3 intensity levels. ■

### JPEG-2000

JPEG-2000 extends the popular JPEG standard to provide increased flexibility in both the compression of continuous-tone still images and access to the compressed data. For example, portions of a JPEG-2000 compressed image can be extracted for retransmission, storage, display, and/or editing. The standard is based on the wavelet coding techniques just described. Coefficient quantization is adapted to individual scales and subbands and the quantized

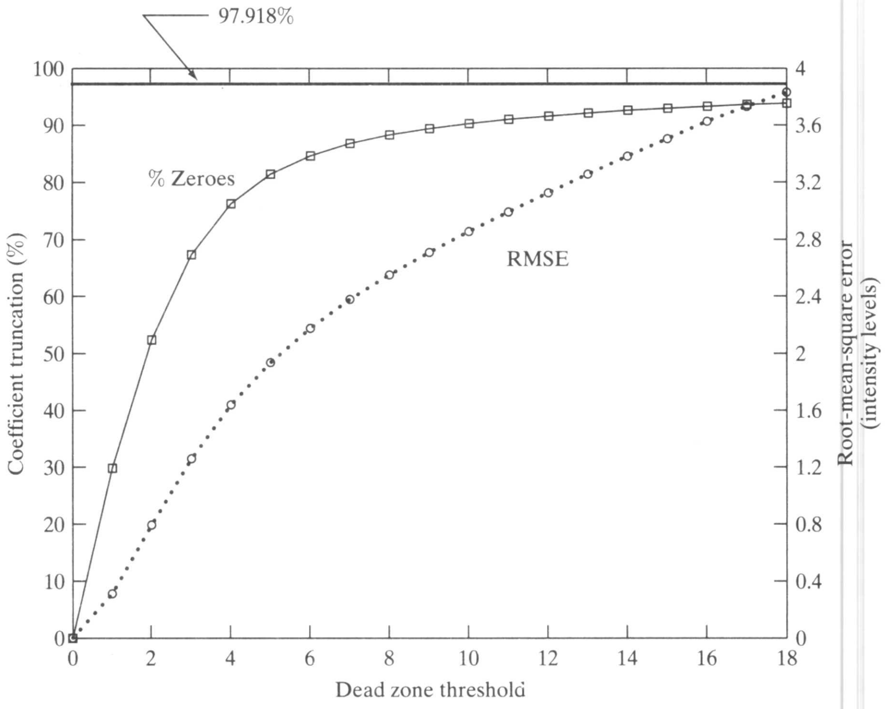


**FIGURE 8.47** The impact of dead zone interval selection on wavelet coding.

coefficients are arithmetically coded on a bit-plane basis (see Sections 8.2.3 and 8.2.7). Using the notation of the standard, an image is encoded as follows (ISO/IEC [2000]).

*Ssiz* is used in the standard to denote intensity resolution.

The first step of the encoding process is to DC level shift the samples of the *Ssiz*-bit unsigned image to be coded by subtracting $2^{Ssiz-1}$. If the image has more than one *component*—like the red, green, and blue planes of a color image—each component is shifted individually. If there are exactly three components, they may be optionally decorrelated using a reversible or nonreversible linear combination of the components. The *irreversible component transform* of the standard, for example, is

The irreversible component transform is the component transform used for lossy compression. The component transform itself is not irreversible. A different component transform is used for reversible compression.

$$
\begin{aligned}
Y_0(x, y) &= 0.299 I_0(x, y) + 0.587 I_1(x, y) + 0.114 I_2(x, y) \\
Y_1(x, y) &= -0.16875 I_0(x, y) - 0.33126 I_1(x, y) + 0.5 I_2(x, y) \\
Y_2(x, y) &= 0.5 I_0(x, y) - 0.41869 I_1(x, y) - 0.08131 I_2(x, y)
\end{aligned}
\tag{8.2-61}
$$

where $I_0$, $I_1$, and $I_2$ are the level-shifted input components and $Y_0$, $Y_1$, and $Y_2$ are the corresponding decorrelated components. If the input components are the red, green, and blue planes of a color image, Eq. (8.2-61) approximates the $R'G'B'$ to $Y'C_bC_r$ color video transform (Poynton [1996]).[†] The goal of the transformation is to improve compression efficiency; transformed components $Y_1$ and $Y_2$ are difference images whose histograms are highly peaked around zero.

[†] $R'G'B'$ is a gamma corrected, nonlinear version of a linear CIE (International Commission on Illumination) *RGB* colorimetry value. $Y'$ is luminance and $C_b$ and $C_r$ are color differences (i.e., scaled $B' - Y'$ and $R' - Y'$ values).

After the image has been level shifted and optionally decorrelated, its components can be divided into *tiles*. Tiles are rectangular arrays of pixels that are processed independently. Because an image can have more than one component (e.g., it could be made up of three color components), the tiling process creates *tile components*. Each tile component can be reconstructed independently, providing a simple mechanism for accessing and/or manipulating a limited region of a coded image. For example, an image having a 16:9 aspect ratio could be subdivided into tiles so that one of its tiles is a subimage with a 4:3 aspect ratio. That tile could then be reconstructed without accessing the other tiles in the compressed image. If the image is not subdivided into tiles, it is a single tile.

The 1-D discrete wavelet transform of the rows and columns of each tile component is then computed. For error-free compression, the transform is based on a biorthogonal, 5-3 coefficient scaling and wavelet vector (Le Gall and Tabatabai [1988]). A rounding procedure is defined for non-integer-valued transform coefficients. In lossy applications, a 9-7 coefficient scaling-wavelet vector (Antonini, Barlaud, Mathieu, and Daubechies [1992]) is employed. In either case, the transform is computed using the fast wavelet transform of Section 7.4 or via a complementary *lifting-based* approach (Mallat [1999]). For example, in lossy applications, the coefficients used to construct the 9-7 FWT analysis filter bank are given in Table 8.15. The complementary lifting-based implementation involves six sequential "lifting" and "scaling" operations:

Lifting-based implementations are another way to compute wavelet transforms. The coefficients used in the approach are directly related to the FWT filter bank coefficients.

$$
\begin{aligned}
Y(2n+1) &= X(2n+1)+\alpha\big[X(2n)+X(2n+2)\big], && i_0 - 3 \le 2n + 1 < i_1 + 3\\
Y(2n) &= X(2n)+\beta\big[Y(2n-1)+Y(2n+1)\big], && i_0 - 2 \le 2n < i_1 + 2\\
Y(2n+1) &= Y(2n+1)+\gamma\big[Y(2n)+Y(2n+2)\big], && i_0 - 1 \le 2n + 1 < i_1 + 1\\
Y(2n) &= Y(2n)+\delta\big[Y(2n-1)+Y(2n+1)\big], && i_0 \le 2n < i_1\\
Y(2n+1) &= -K\cdot Y(2n+1), && i_0 \le 2n + 1 < i_1\\
Y(2n) &= Y(2n)/K, && i_0 \le 2n < i_1 \qquad (8.2\text{-}62)
\end{aligned}
$$

Here, $X$ is the tile component being transformed, $Y$ is the resulting transform, and $i_0$ and $i_1$ define the position of the tile component within a component. That is, they are the indices of the first sample of the tile-component row or column being transformed and the one immediately following the last sample. Variable $n$ assumes values based on $i_0$, $i_1$, and which of the six operations is

| Filter Tap | Highpass Wavelet Coefficient | Lowpass Scaling Coefficient |
|---|---|---|
| 0 | −1.115087052456994 | 0.6029490182363579 |
| ±1 | 0.5912717631142470 | 0.2668641184428723 |
| ±2 | 0.05754352622849957 | −0.07822326652898785 |
| ±3 | −0.09127176311424948 | −0.01686411844287495 |
| ±4 | 0 | 0.02674875741080976 |

**TABLE 8.15** Impulse responses of the low- and highpass analysis filters for an irreversible 9-7 wavelet transform.

being performed. If $n < i_0$ or $n \geq i_1$, $X(n)$ is obtained by symmetrically extending $X$. For example, $X(i_0 - 1) = X(i_0 + 1)$, $X(i_0 - 2) = X(i_0 + 2)$, $X(i_1) = X(i_1 - 2)$, and $X(i_1 + 1) = X(i_1 - 3)$. At the conclusion of the lifting and scaling operations, the even-indexed values of $Y$ are equivalent to the FWT lowpass filtered output; the odd-indexed values of $Y$ correspond to the highpass FWT filtered result. Lifting parameters $\alpha, \beta, \gamma$, and $\delta$ are $-1.586134342$, $-0.052980118$, $0.882911075$, and $0.433506852$, respectively. Scaling factor $K$ is $1.230174105$.

These lifting-based coefficients are specified in the standard.

Recall from Chapter 7 that the DWT decomposes an image into a set of band-limited components called subbands.

The transformation just described produces four subbands—a low-resolution approximation of the tile component and the component's horizontal, vertical, and diagonal frequency characteristics. Repeating the transformation $N_L$ times, with subsequent iterations restricted to the previous decomposition's approximation coefficients, produces an $N_L$-scale wavelet transform. Adjacent scales are related spatially by powers of 2 and the lowest scale contains the only explicitly defined approximation of the original tile component. As can be surmised from Fig. 8.48, where the notation of the JPEG-2000 standard is summarized for the case of $N_L = 2$, a general $N_L$-scale transform contains $3N_L + 1$ subbands whose coefficients are denoted $a_b$, for $b = N_LLL, N_LHL, \ldots, 1HL, 1LH, 1HH$. The standard does not specify the number of scales to be computed.

When each of the tile components has been processed, the total number of transform coefficients is equal to the number of samples in the original image—but the important visual information is concentrated in a few coefficients. To reduce the number of bits needed to represent the transform, coefficient $a_b(u, v)$ of subband $b$ is quantized to value $q_b(u, v)$ using

$$q_b(u, v) = \text{sign}\left[a_b(u, v)\right] \cdot \text{floor}\left[\frac{|a_b(u, v)|}{\Delta_b}\right] \tag{8.2-63}$$

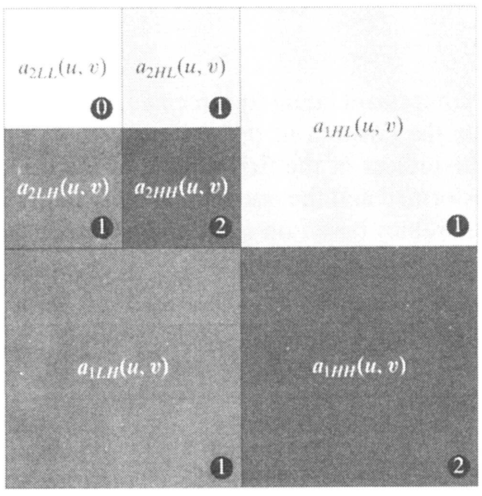


**FIGURE 8.48** JPEG 2000 two-scale wavelet transform tile-component coefficient notation and analysis gain.

where the *quantiztion step size* $\Delta_b$ is

$$\Delta_b = 2^{R_b - \varepsilon_b}\left(1 + \frac{\mu_b}{2^{11}}\right) \tag{8.2-64}$$

$R_b$ is the *nominal dynamic range* of subband $b$, and $\varepsilon_b$ and $\mu_b$ are the number of bits allotted to the *exponent* and *mantissa* of the subband's coefficients. The nominal dynamic range of subband $b$ is the sum of the number of bits used to represent the original image and the *analysis gain* bits for subband $b$. Subband analysis gain bits follow the simple pattern shown in Fig. 8.48. For example, there are two analysis gain bits for subband $b = 1HH$.

Do not confuse the standard's definition of nominal dynamic range with the closely related definition in Chapter 2.

For error-free compression, $\mu_b = 0$, $R_b = \varepsilon_b$, and $\Delta_b = 1$. For irreversible compression, no particular quantization step size is specified in the standard. Instead, the number of exponent and mantissa bits must be provided to the decoder on a subband basis, called *expounded quantization*, or for the $N_L LL$ subband only, called *derived quantization*. In the latter case, the remaining subbands are quantized using extrapolated $N_L LL$ subband parameters. Letting $\varepsilon_0$ and $\mu_0$ be the number of bits allocated to the $N_L LL$ subband, the extrapolated parameters for subband $b$ are

$$\begin{aligned} \mu_b &= \mu_0 \\ \varepsilon_b &= \varepsilon_0 + n_b - N_L \end{aligned} \tag{8.2-65}$$

where $n_b$ denotes the number of subband decomposition levels from the original image tile component to subband $b$.

In the final steps of the encoding process, the coefficients of each transformed tile-component's subbands are arranged into rectangular blocks called *code blocks*, which are coded individually, one bit plane at a time. Starting from the most significant bit plane with a nonzero element, each bit plane is processed in three passes. Each bit (in a bit plane) is coded in only one of the three passes, which are called *significance propagation*, *magnitude refinement*, and *cleanup*. The outputs are then arithmetically coded and grouped with similar passes from other code blocks to form *layers*. A layer is an arbitrary number of groupings of coding passes from each code block. The resulting layers finally are partitioned into *packets*, providing an additional method of extracting a spatial region of interest from the total code stream. Packets are the fundamental unit of the encoded code stream.

JPEG-2000 decoders simply invert the operations described previously. After reconstructing the subbands of the tile-components from the arithmetically coded JPEG-2000 packets, a user-selected number of the subbands is decoded. Although the encoder may have encoded $M_b$ bit planes for a particular subband, the user—due to the embedded nature of the code stream—may choose to decode only $N_b$ bit planes. This amounts to quantizing the coefficients of the code block using a step size of $2^{M_b - N_b} \cdot \Delta_b$. Any nondecoded bits are set to zero and the resulting coefficients, denoted