Contents

List of algorithms

Chapter 14

Image data compression

Image processing is often very difficult because of the large amounts of data used to represent an image. Technology permits ever-increasing image resolution (spatially and in gray-levels), and increasing numbers of spectral bands, and there is a consequent need to limit the resulting data volume. Consider an example from the remote sensing domain, where image data compression is a very serious problem.

A Landsat D satellite broadcasts 85×10^6 bits of data every second and a typical image from one pass consists of 6100×6100 pixels in seven spectral bands—260 megabytes of image data. As another example, the Japanese Advanced Earth Observing Satellite (ADEOS), which has the ability to observe the Earth's surface with a spatial resolution of 8 meters for the polychromatic band and 16 meters for the multi-spectral bands has a transmitted data rate of 120 Mbps. Thus the amount of storage media needed for archiving of such remotely sensed data is enormous. The situation is similar in medical imaging with 3D and 4D data sets being common. For example, a single head-to-toe 64-detector CT scan of a human body can be acquired in under 10 second. Such a CT machine can deliver volumetric images at the resolution of $0.5\,\mathrm{mm}^3$. A full-body scan at this resolution corresponds to almost 2 GB of data ($512 \times 512 \times 3600 \times 2$ bytes). Similar data file sizes are obtained from micro-CT scanners used for small animal research. One possible approach to decreasing the necessary amount of storage is to work with compressed image data.

We have seen that segmentation techniques have the side effect of image compression; by removing all areas and features that are not of interest, and leaving only boundaries or region descriptors, the reduction in data quantity is considerable. However, from this sort of representation no image reconstruction to the original uncompressed image (or only a very limited reconstruction) is possible. Conversely, image compression algorithms aim to remove redundancy in data in a way which makes image reconstruction possible; this is sometimes called *information preserving compression*. Compression is the main goal of the algorithm—we aim to represent an image using fewer bits per pixel, without losing the ability to reconstruct the image. It is necessary to find statistical properties of the image to design an appropriate compression transformation of the image; the more correlated the image data are, the more data items can be removed. In this chapter, we will discuss this group of methods which do not change image entropy or image information content. More detailed surveys of image compression techniques may be found in [Rosenfeld and Kak 82; Clarke 85; Netravali 88; Rabbani 91; Witten et al. 94; Furht et al. 95; Clarke 95; Shi and Sun 99].

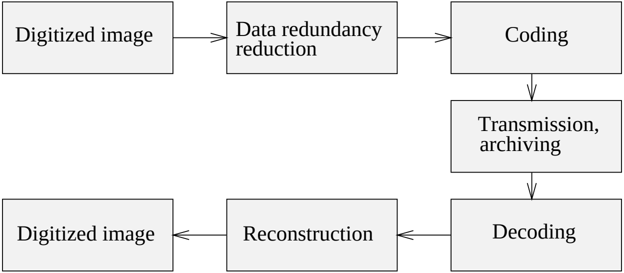


Figure 14.1: Data compression and image reconstruction.

A general algorithm for data compression and image reconstruction is shown in a block diagram in Figure 14.1. The first step removes information redundancy caused by high correlation of image data—transform compressions, predictive compressions, and hybrid approaches are used. The second step is coding of transformed data using a code of fixed or variable-length. An advantage of variable-length codes is the possibility of coding more frequent data using shorter code words and therefore increasing compression efficiency, while an advantage of fixed length coding is a standard codeword length that offers easy handling and fast processing. Compressed data are decoded after transmission or archiving and reconstructed. Note that no non-redundant image data may be lost in the data compression process—otherwise error-free reconstruction is impossible.

Data compression methods can be divided into two principal groups: **information preserving** compressions permit error-free data reconstruction (lossless compression), while compression methods **with loss of information** do not preserve the information completely (lossy compression). In image processing, a faithful reconstruction is often not necessary in practice and then the requirements are weaker, but the image data compression must not cause significant changes in an image. Data compression success in the reconstructed image is usually measured by the mean square error (MSE), signal-to-noise ratio etc., although these global error measures do not always reflect subjective image quality.

Image data compression design consists of two parts. Image data properties must be determined first; gray-level histograms, image entropy, various correlation functions, etc., often serve this purpose. The second part yields an appropriate compression technique design with respect to measured image properties.

Data compression methods with loss of information are typical in image processing and therefore this group of methods is described in some detail. Although lossy compression techniques can give substantial image compression with very good quality reconstruction, there are considerations that may prohibit their use. For example, diagnosis in medical imaging is often based on visual image inspection, so no loss of information can be tolerated and information preserving techniques must be applied. Information preserving compression methods are mentioned briefly at the end of the chapter.

14.1 Image data properties

Information content of an image is an important property, of which **entropy** is a measure (Section **??**). If an image has G gray-levels and the probability of gray-level k is $P(k)$ (see Section **??**), then entropy H_e, not considering correlation of gray-levels, is defined as

$$H_e = -\sum_{k=0}^{G-1} P(k) \log_2 \big(P(k)\big) . \tag{14.1}$$

Information **redundancy** r is defined as

$$r = b - H_e , \tag{14.2}$$

where b is the smallest number of bits with which the image quantization levels can be represented. This definition of image information redundancy can be evaluated only if a good estimate of entropy is available, which is usually not so because the necessary statistical properties of the image are not known. Image data entropy however can be estimated from a gray-level histogram [Moik 80; Pratt 91]. Let $h(k)$ be the frequency of gray-level k in an image f, $0 \leq k \leq 2^b - 1$, and let the image size be $M \times N$. The probability of occurrence of gray-level k can be estimated as

$$\tilde{P}(k) = \frac{h(k)}{MN} \tag{14.3}$$

and the entropy can be estimated as

$$\tilde{H}_e = - \sum_{k=0}^{2^b-1} \tilde{P}(k) \log_2 \left(\tilde{P}(k) \right) . \tag{14.4}$$

The information redundancy estimate is $\tilde{r} = b - \tilde{H}_e$. The definition of the **compression ratio** K is then

$$K = \frac{b}{\tilde{H}_e} . \tag{14.5}$$

Note that a gray-level histogram gives an inaccurate estimate of entropy because of gray-level correlation. A more accurate estimate can be obtained from a histogram of the first gray-level differences.

Theoretical limits of possible image compression can be found using these formulae. For example, the entropy of satellite remote sensing data may be $\tilde{H}_e \in [4, 5]$, where image data are quantized into 256 gray-levels, or 8 bits per pixel. We can easily compute the information redundancy as $\tilde{r} \in [3,4]$ bits. This implies that these data can be represented by an average data volume of 4–5 bits per pixel with no loss of information, and the compression ratio would be $K \in [1.6, 2]$.

14.2 Discrete image transforms in image data compression

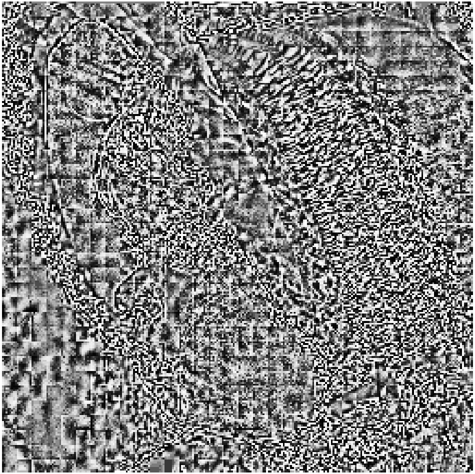
Image data representation by coefficients of discrete image transforms (see Section **??**) is the basic idea of this approach. The transform coefficients are ordered according to their importance, i.e., according to their contribution to the image information contents, and the least important (low-contribution) coefficients are omitted. Coefficient importance can be judged, for instance, in correspondence to spatial or gray-level visualization abilities of the display; image correlation can then be avoided and data compression may result.

To remove correlated image data, the **Karhunen-Loève** transform is the most important. This transform builds a set of non-correlated variables with decreasing variance. The variance of a variable is a measure of its information content; therefore, a compression strategy is based on considering only transform variables with high variance, thus representing an image by only the first k coefficients of the transform. More details about the Karhunen-Loève transform can be found in Section **??**.

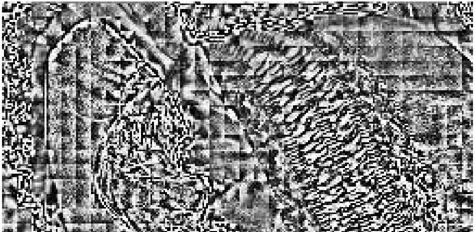
The Karhunen-Loève transform is computationally expensive, with a two-dimensional transform of an $M \times N$ image having computational complexity $\mathcal{O}(\mathcal{M}^{\in}\mathcal{N}^{\in})$. It is the only transform that guarantees non-correlated compressed data, and the resulting data compression is optimal in the statistical sense. This makes the transform basis vectors image dependent, which also makes this transform difficult to apply for routine image compression. Therefore, the Karhunen-Loève transform is used mainly as a benchmark to evaluate other transforms. For example, one reason for the popularity of the discrete cosine transform DCT-II is that its performance approaches the Karhunen-Loève transform better than others.

Other discrete image transforms (see Section **??**) are computationally less demanding—fast algorithms of these transforms have computational complexity $\mathcal{O}\big(\mathcal{M}\mathcal{N}\log_{\in}(\mathcal{M}\mathcal{N})\big)$. Cosine, Fourier, Hadamard, Walsh, or binary transforms are all suitable for image data compression. If an image is compressed using discrete transforms, it is usually divided into subimages of 8×8 or 16×16 pixels to speed up calculations, and then each subimage is transformed and processed separately. The same is true for image reconstruction, with each subimage being reconstructed and placed into the appropriate image position. This image segmentation into a grid of subimages does not consider any possible data redundancy caused by subimage correlation even if this correlation is the most serious source of redundancy. **Recursive block** coding [Farelle 90] is an important novel approach to reducing inter-block redundancy and tiling effects (blockiness). The most popular image transform used for image compression seems to be the discrete cosine transform with many modifications, and variations of wavelet transforms (Section **??**).

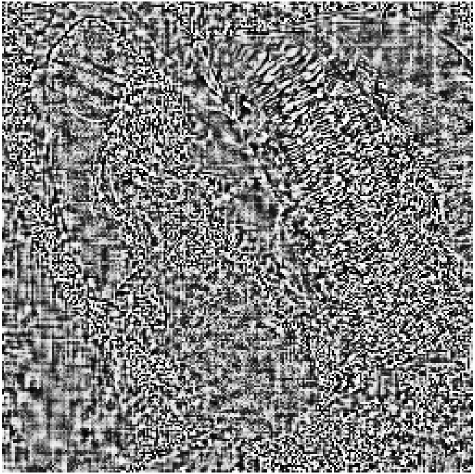
(a)



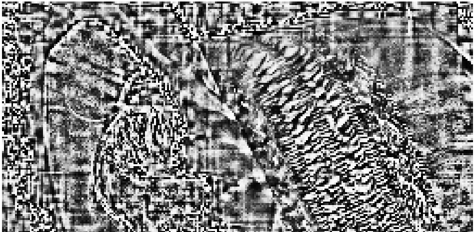
(b)



(a)



(b)



Discrete cosine transform image compression possibilities are shown in Figure 14.2. The DCT-II applied here provides good compression with low computational demands, the compression ratios being $K = 6.2$ and $K = 10.5$. The lower compression ratio was achieved after setting 90.0% of the transform coefficients to zero; the higher compression ratio resulted after setting 94.9% of the transform coefficients to zero. Note that square blocks resulting from DCT compression and reconstruction decrease the image quality for larger compression ratios. Consequently, **wavelet image compression** is of interest, since it can be efficiently applied to the entire image and thus the square image compression artifacts are not present. Wavelet compression consists of the same steps as DCT compression, but the DCT is replaced by a wavelet transform followed by generally identical quantization and coding. Figure 14.3 shows the reconstructed image after wavelet compression with two different compression ratios, $K = 6.2$ and $K = 10.5$. The lower compression ratio (Figure 14.3a,b) was achieved after setting 89.4% of the transform coefficients to zero, the higher compression ratio (Figure 14.3a,b) resulted after setting 94.4% of the transform coefficients to zero. Note that no blocking artifacts exist.

While DCT compression is the basis for the widely used JPEG compression standard (Section 14.9.1), wavelet compression has become the basis for a new image compression standard called JPEG–2000 (Section 14.9.2).

14.3 Predictive compression methods

Predictive compressions use image information redundancy (correlation of data) to construct an estimate $\tilde{f}(i,j)$ of the gray-level value of an image element (i,j) from values of gray-levels in the neighborhood of (i,j). In image parts where data are not correlated, the estimate $\tilde{f}$ will not match the original value. The differences between estimates and reality, which may be expected to be relatively small in absolute terms, are coded and transmitted (stored) together with prediction model parameters—the whole set now represents compressed image data. The gray value at location (i,j) is reconstructed from a computed estimate $\tilde{f}(i,j)$ and the stored difference $d(i,j)$

$$d(i,j) = \tilde{f}(i,j) - f(i,j)\,. \tag{14.6}$$

This method is called differential pulse code modulation (DPCM)—its block diagram is presented in Figure 14.4. Experiments show that a linear predictor of the third order is sufficient for estimation in a wide variety of images. If the image is processed line by line, the estimate $\tilde{f}$ can be computed as

$$\tilde{f}(i,j) = a_1\, f(i,j-1) + a_2\, f(i-1,j-1) + a_3\, f(i-1,j)\,, \tag{14.7}$$

where a_1, a_2, a_3 are image prediction model parameters. These parameters are set to minimize the mean quadratic estimation error e

$$e = \mathcal{E}\Big\{\big(\tilde{f}(i,j) - f(i,j)\big)^2\Big\}\,, \tag{14.8}$$

and the solution, assuming f is a stationary random process with a zero mean, using a predictor of the third order, is

$$\begin{aligned} a_1\,R(0,0) + a_2\,R(0,1) + a_3\,R(1,1) &= R(1,0)\,, \\ a_1\,R(0,1) + a_2\,R(0,0) + a_3\,R(1,0) &= R(1,1)\,, \\ a_1\,R(1,1) + a_2\,R(1,0) + a_3\,R(0,0) &= R(0,1)\,, \end{aligned} \tag{14.9}$$

where $R(m,n)$ is the autocorrelation function of the random process f (see Chapter **??**). The image data autocorrelation function is usually of exponential form and the variance of differences $d(i,j)$ is usually smaller than the variance of the original values $f(i,j)$, since the differences $d(i,j)$ are not correlated. The (probable) relatively small magnitude of the differences $d(i,j)$ makes data compression possible.

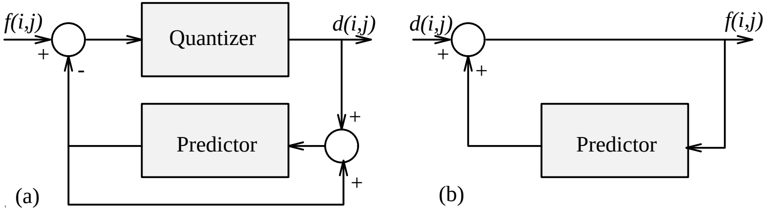


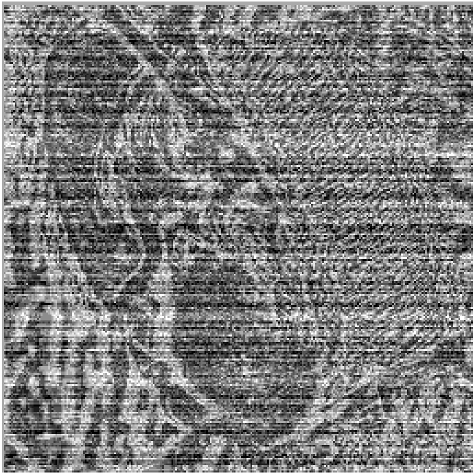
Figure 14.4: Differential pulse code modulation: (a) compression; (b) reconstruction.

Predictive compression algorithms are described in detail in [Rosenfeld and Kak 82; Netravali 88]. A predictive method of second order with variable code length coding of the differences $d(i, j)$ was used to obtain the compressed images shown in Figure 14.5; data compression ratios $K = 3.8$ and $K = 6.2$ were achieved. Note that horizontal lines and false contours resulting from the predictive compression and reconstruction decrease the image quality for larger compression ratios.

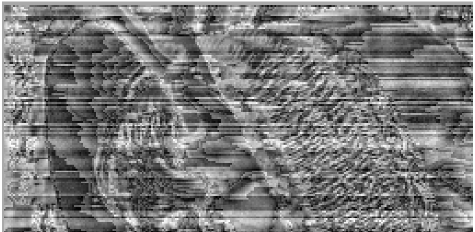
Many modifications of predictive compression methods can be found in the literature, some of them combining predictive compression with other coding schemes [Daut and Zhao 90; Zailu and Taxiao 90].

14.4 Vector quantization

Dividing an image into small blocks and representing these blocks as vectors is another option [Gray 84; Chang et al. 88; Netravali 88; Gersho and Gray 92]. The basic idea for this approach comes from information theory (Shannon's rate distortion theory), which states that better compression performance can always be achieved by coding vectors instead of scalars. Input data vectors are coded using unique codewords from a codeword dictionary, and instead of vectors, the vector codes are stored or transmitted. The codeword choice is based on the best similarity between the image block represented by a coded vector and the image blocks represented by codewords from the dictionary. The code dictionary (code book) is transmitted together with the coded data. The advantage of vector quantization



(a) (b)



is a simple receiver structure consisting of a look-up table, but a disadvantage is a complex coder. The coder complexity is not caused directly by the vector quantization principle; the method can be implemented in a reasonably simple way, but the coding will be very slow. To increase the processing speed, special data structures (K-D trees) and other special treatments are needed which increase the coder complexity. Further, the necessary statistical properties of images are usually not available. Therefore, the compression parameters must be based on an image training set and the appropriate code book may vary from image to image. As a result, images with statistical properties dissimilar from images in the training set may not be well represented by the code vectors in the look-up table. Furthermore, edge degradation may be more severe than with other techniques. To decrease the coder complexity, the coding process may be divided into several levels, two being typical. The coding process is hierarchical, using two or more code books according to the number of coding levels. However, the combination of a complex coder facilitating high compression ratios and a simple decoder may be advantageous in **asymmetric** applications when the image is compressed once and decompressed many times. Within such a scenario, the higher compression ratio gained by the more complex coder and/or more time-consuming compression algorithm does not matter as long as the decompression process is simple and fast. Multimedia encyclopedias and paperless publishing serve as good examples. On the other hand, in **symmetric** applications such as video conferencing, similar complexity of coding and decoding operations is required.

A modification that allows blocks of variable size is described in [Boxerman and Lee 90], where a segmentation algorithm is responsible for detecting appropriate image blocks. The block vector quantization approach may also be applied to compression of image sequences. Identifying and processing only blocks of the image that change noticeably between consecutive frames using vector quantization and DPCM is also possible. Hybrid DPCM combined with vector quantization of colored prediction errors is presented in [De Lameillieure and Bruyland 90].

14.5 Hierarchical and progressive compression methods

Multi-resolution pyramids have been mentioned many times throughout this book, and they may also be used for efficient hierarchical image compression. **Run length** codes were introduced in Section **??**, Figure **??**; run length coding identifies long runs of the same value pixels, and stores them as this value together with a word count. If the image is characterized by such long runs, this will significantly reduce storage requirements. A similar approach may be applied to image pyramids. A substantial reduction in bit volume can be obtained by merely representing a source as a pyramid [Rao and Pearlman 91], and even more significant reduction can be achieved for images with large areas of the same gray-level if a quadtree coding scheme is applied (see Section **??**). An example is given in Figure 14.6, where the principle of quadtree image compression is presented. Large image areas of the same gray-level can be represented in higher-level quadtree nodes without the necessity of including lower-level nodes in the image representation [White 87]. Clearly, the compression ratio achieved is image dependent and, for instance, a fine checkerboard image will not be represented efficiently using quadtrees. Modifications of the basic method exist, some of them successfully applied to motion image compression [Strobach 90] or incorporating hybrid schemes [Park and Lee 91].

Nevertheless, there may be an even more important aspect connected with this compression approach—the feasibility of progressive image transmission and the idea of smart compression.

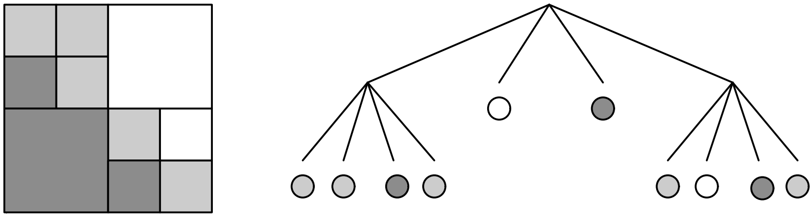


Figure 14.6: Principle of quadtree image compression: original image and corresponding quadtree.

Progressive image transmission is based on the fact that transmitting all image data may not be necessary under some circumstances. Imagine a situation in which an operator is searching an image database looking for a particular image. If the transmission is based on a raster scanning order, all the data must be transmitted to view the whole image, but often it is not necessary to have the highest possible image quality to find the image for which the operator is looking. Images do not have to be displayed with the highest available resolution, and lower resolution may be sufficient to reject an image and to begin displaying another one. This approach is also commonly used to decrease the waiting time needed for the image to start appearing after transmission and is used by World Wide Web image transmissions. In progressive transmission, the images are represented in a pyramid structure, the higher pyramid levels (lower resolution) being transmitted first. The number of pixels representing a lower-resolution image is substantially smaller and thus the user can decide from lower-resolution images whether further image refinement is needed. A standard M-pyramid (mean or matrix pyramid) consists of about one third more nodes than the number of image pixels. Several pyramid encoding schemes have been designed to decrease the necessary number of nodes in pyramid representation: reduced sum pyramids, difference pyramids, and reduced difference pyramids [Wang and Goldberg 89]. The reduced difference pyramid has the number of nodes exactly equal to the number of image pixels and can be used for a lossless progressive image transmission with some degree of compression. Using an appropriate interpolation method in the image reconstruction stage, reasonable image quality can be achieved at a bit rate of less than 0.1 bit/pixel and excellent quality at a bit rate of about 1.2 bits/pixel. Progressive image transmission stages can be seen in Figure **??**, where a sequence of four image resolutions is presented.

Considering a hypothetical progressive image transmission, a 1/8-resolution image is transmitted first (Figure **??**d). Next, the image is transmitted and displayed in 1/4 resolution (Figure **??**c), followed by 1/2 resolution (Figure **??**b) and then full resolution (Figure **??**a).

The concept of **smart compression** is based on the sensing properties of human visual sensors. The spatial resolution of the human eye decreases significantly with increasing distance from the optical axis. Therefore, the human eye can only see in high resolution in a very small area close to the point where the eye is focused. Similarly, as with image displays, where it does not make sense to display or even transmit an image in higher resolution than that of the display device, it is not necessary to display an image in full resolution in image areas where the user's eyes are not focused. This is the principle of smart image compression. The main difficulty remains in determining the areas of interest in the image on which the user will focus. When considering a smart progressive image transmission, the image should be transmitted first in higher resolution in areas of interest—this improves the subjective rating of transmission speed as sensed by a human user. The areas of interest may be obtained in a feedback control manner by tracking the user's eyes (assuming the communication channel is fast enough). The image point on which the user is focused may be used to increase the resolution in that particular image area so that the most important data are transmitted first. This smart image transmission and compression may be extremely useful if applied to dynamic image generators in driving or flight simulators, or to high-definition television.

14.6 Comparison of compression methods

The main goal of image compression is to minimize image data volume with no significant loss of information, and all basic image compression groups have advantages and disadvantages. Transform-based methods better preserve subjective image quality, and are less sensitive to statistical image property changes both inside a single image and between images. Prediction methods, on the other hand, can achieve higher compression ratios in a much less expensive way, tend to be much faster than transform-based or vector quantization compression schemes, and can easily be realized in hardware. If compressed images are transmitted, an important property is insensitivity to transmission channel noise. Transform-based techniques are significantly less sensitive to channel noise—if a transform coefficient is corrupted during transmission, the resulting image distortion is spread homogeneously through the image or image part and is not too disturbing. Erroneous transmission of a difference value in prediction compressions causes not only an error in a particular pixel, it influences values in the neighborhood because the predictor involved has a considerable visual effect in a reconstructed image. Vector quantization methods require a complex coder, their parameters are very sensitive to image data, and they blur image edges. The advantage is in a simple decoding scheme consisting of a look-up table only. Pyramid-based techniques have a natural compression ability and show a potential for further improvement of compression ratios. They are suitable for dynamic image compression and for progressive and smart transmission approaches.

Hybrid compression methods combine good properties of the various groups. A hybrid compression of three-dimensional image data (two spatial dimensions plus one spectral dimension) is a good example. A two-dimensional discrete transform (cosine, Hadamard, . . .) is applied to each mono-spectral image followed by a predictive compression in the third dimension of spectral components. Hybrid methods combine the different dimensionalities of transform compressions and predictive compressions. As a general rule, at least a one-dimensional transform compression precedes predictive compression steps. In addition to combinations of transform and predictive approaches, predictive approaches are often combined with vector quantization. A discrete cosine transform combined with vector quantization in a pyramid structure is described in [Park and Lee 91].

For more detailed comparisons of some image compression techniques refer to [Chang et al. 88; Jaisimha et al. 89; DiMento and Berkovich 90; Hung and Meng 94].

14.7 Other techniques

Various other image data compression methods exist. If an image is quantized into a small number of gray-levels and if it has a small number of regions of the same gray-level, an effective compression method may be based on **coding region borders** [Wilkins and Wintz 71]. Image representation by its **low and high frequencies** is another method—image reconstruction is a superposition of inverse transforms of low- and high-frequency components. The low-frequency image can be represented by a significantly smaller volume of data than the original image. The high-frequency image has significant image edges only and can be represented efficiently [Graham 67]. The **region growing process** compression method stores an algorithm for region growing from region seed points, each region being represented by its seed point. If an image can be represented only by region seed points, significant data compression is achieved.

Block truncation coding divides an image into small square blocks of pixels and each pixel value in a block is truncated to one bit by thresholding and moment preserving selection of binary levels [Delp and Mitchell 79; Rosenfeld and Kak 82; Kruger 92]. One bit value per pixel has to be transmitted, together with information describing how to recreate the moment preserving binary levels during reconstruction. This method is fast and easy to implement. **Visual pattern image** coding is capable of high-quality compression with very good compression ratios (30:1) and is exceptionally fast [Silsbee et al. 91].

Fractal image compression is another approach offering extremely high compression ratios and high-quality image reconstruction. Additionally, because fractals are infinitely magnifiable, fractal compression is resolution independent and so a single compressed image can be used efficiently for display in any image resolution including resolution higher than the original [Furht et al. 95]. Breaking an image into pieces (fractals) and identifying self-similar ones is the main principle of the approach [Barnsley and Hurd 93; Fisher 94]. First, the image is partitioned into non-overlapping *domain* regions of any size and shape that completely cover it. Then, larger *range* regions are defined that can overlap and need not cover the entire image. These range regions are geometrically transformed using affine transforms (Section ??) to match the domain regions. Then the set of affine coefficients together with information about the selection of domain regions represents the fractal image encoding. The fractally compressed images are stored and transmitted as recursive algorithms—sets of equations with instructions on how to reproduce the image. Clearly, fractal compression is compute demanding. However, decompression is simple and fast; domain regions are iteratively replaced with appropriately geometrically transformed range regions using the affine coefficients. Thus, fractal compression represents another example of an extremely promising asymmetric compression-decompression scheme.

14.8 Coding

In addition to techniques designed explicitly to cope with 2D (or higher-dimensional) data, there is a wide range of well-known algorithms designed with serial data (e.g., simple text files) in mind. These algorithms see wide use in the compression of ordinary computer files to reduce disk consumption. Very well known is **Huffman encoding**, which can provide optimal compression and error-free decompression [Rosenfeld and Kak 82]. The main idea of Huffman coding is to represent data by codes of variable length, with more frequent data being represented by shorter codes. Many modifications of the original algorithm [Huffman 52] exist, with adaptive Huffman coding algorithms requiring only one pass over the data [Knuth 85; Vitter 87].

The **Lempel-Ziv** (or Lempel-Ziv-Welch, LZW) algorithm of **dictionary-based** coding [Ziv and Lempel 78; Nelson 89] has found wide favor as a standard compression algorithm. In this approach, data are represented by pointers referring to a dictionary of symbols.

These, and a number of similar techniques, are in widespread use for de-facto standard image representations which are popular for Internet and World Wide Web image exchange. Of these, the **GIF** format (Graphics Interchange Format) is frequently used. GIF is a creation of Compuserve, Inc., and is designed for the encoding of RGB images (and the appropriate palette) with pixel depths between 1 and 8 bits. Blocks of data are encoded using the LZW algorithm. GIF has two main versions—87a and 89a [Compuserve 89], the latter supporting the storing of text and graphics in the same file. Additionally, **TIFF** (Tagged Image File Format) is widely encountered (and is the cause of much popular confusion). TIFF was first defined by the Aldus Corporation in 1986, and has gone through a number of versions to incorporate RGB color, compressed color (LZW), other color formats, and ultimately (in Version 6 [Aldus 92]), JPEG compression (see Section 14.9)—these versions all have backward compatibility. There are some recorded problems with the JPEG implementation, and TIFF has a reputation for being complex, although this is undeserved and it is a powerful programmer's tool. It is a particularly popular format among desktop publishers, and for scanners.

14.9 JPEG and MPEG image compression

There is an increasing effort to achieve standardization in image compression. The Joint Photographic Experts Group (JPEG) has developed an international standard for general purpose, color, still image compression. As a logical extension of JPEG still image compression, the Motion Picture Experts Group (MPEG) standard was developed for full-motion video image sequences with applications to digital video distribution and high-definition television (HDTV) in mind.

14.9.1 JPEG—still image compression

The JPEG compression system is widely used in many application areas. Four compression modes are furnished:

- Sequential DCT-based compression.
- Progressive DCT-based compression.
- Sequential lossless predictive compression.
- Hierarchical lossy or lossless compression.

While the lossy compression modes were designed to achieve compression ratios around 15 with very good or excellent image quality, the quality deteriorates for higher compression ratios. A compression ratio between 2 and 3 is typically achieved in the lossless mode.

Sequential JPEG compression

Following Figure 14.1, sequential JPEG compression consists of a forward DCT transform, a quantizer, and an entropy encoder, while decompression starts with entropy decoding followed by dequantizing and inverse DCT.

In the compression stage, the unsigned image values from the interval $[0, 2^b - 1]$ are first shifted to cover the interval $[-2^{b-1}, 2^{b-1} - 1]$. The image is then divided into 8×8 blocks and each block is independently transformed into the frequency domain using the DCT-II transform [Section **??**, equation (**??**)]. Many of the 64 DCT coefficients have zero or near-zero values in typical 8×8 blocks, which forms the basis for compression. The 64 coefficients are quantized using a quantization table $Q(u, v)$ of integers from 1 to 255 that is specified by the application to reduce the storage/transmission requirements of coefficients that contribute little or nothing to the image content. The following formula is used for quantization:

$$F_Q(u, v) = \text{round}\left(\frac{F(u, v)}{Q(u, v)}\right) . \tag{14.10}$$

After quantization, the dc coefficient $F(0, 0)$ is followed by the 63 ac coefficients that are ordered in a 2D matrix in a zigzag fashion according to their increasing frequency. The dc coefficients are then encoded using predictive coding (Section 14.3), the rationale being that average gray-levels of adjacent 8×8 blocks (dc coefficients) tend to be similar.

The last step of the sequential JPEG compression algorithm is entropy encoding. Two approaches are specified by the JPEG standard. The baseline system uses simple Huffman coding, while the extended system uses arithmetic coding and is suitable for a wider range of applications.

Sequential JPEG decompression uses all the steps described above in the reverse order. After entropy decoding (Huffman or arithmetic), the symbols are converted into DCT coefficients and dequantized:

$$F'_Q(u,v) = F_Q(u,v)\,Q(u,v)\,, \tag{14.11}$$

where again, the $Q(u,v)$ are quantization coefficients from the quantization table that is transmitted together with the image data. Finally, the inverse DCT transform is performed according to equation (**??**) and the image gray values are shifted back to the interval $[0, 2^b - 1]$.

The JPEG compression algorithm can be extended to color or multi-spectral images with up to 256 spectral bands.

Progressive JPEG compression

The JPEG standard also facilitates progressive image transmission (Section 14.5). In the progressive compression mode, a sequence of scans is produced, each scan containing a coded subset of DCT coefficients. Thus, a buffer is needed at the output of the quantizer to store all DCT coefficients of the entire image. These coefficients are selectively encoded.

Three algorithms are defined as part of the JPEG progressive compression standard: **progressive spectral selection**, **progressive successive approximation**, and the **combined progressive algorithm**. In the progressive spectral selection approach, the dc coefficients are transmitted first, followed by groups of low-frequency and higher-frequency coefficients. In the progressive successive approximation, all DCT coefficients are sent first with lower precision, and their precision is increased as additional scans are transmitted. The combined progressive algorithm uses both of the above principles together.

Sequential lossless JPEG compression

The lossless mode of the JPEG compression uses a simple predictive compression algorithm and Huffman coding to encode the prediction differences (Section 14.3).

Hierarchical JPEG compression

Using the hierarchical JPEG mode, decoded images can be displayed either progressively or at different resolutions. A pyramid of images is created and each lower-resolution image is used as a prediction for the next-higher-resolution pyramid level (Section 14.5). The three main JPEG modes can be used to encode the lower-resolution images—sequential DCT, progressive DCT, or lossless.

In addition to still image JPEG compression, motion JPEG (MJPEG) compression exists that can be applied to real-time full motion applications. However, MPEG compression represents a more common standard and is described below.

14.9.2 JPEG–2000 compression

- DCT compression — basis for JPEG
- wavelet compression — basis for JPEG–2000

- JPEG–2000 — new international standard for still image compression
  - overcomes some limitations of original JPEG standard
  - not its extension
  - new, powerful, flexible environment for image compression
  - flexibility allows compression of different types of still images (bi-level, gray-level, color, multi-band) with different characteristics (natural images, scientific, medical, military imagery, text, rendered graphics) within a unified system

  - removes need for different compression mechanisms for lossless and lossy compression
  - represents lossless compression as cohesive extension of lossy compression

- $\Rightarrow$ important paradigm shift — allows compression of image data in a lossless manner and—at a later time—a selective data removal to represent images in a lossy fashion while increasing the compression ratio
- **quality scalability** — lossless and lossy behavior from the same compressed image data source
- **resolution scalability**— extraction of lower resolution images from the same data source
- **spatial scalability** — selective reconstruction of individually defined regions from compressed image data source

- JPEG 2000 standard creates unified image compression environment
- but – only specifies
  - decoder operations
  - bitstream syntax
  - file format
- this allows for future improvements and innovations of coding

- Encoding — two primary paths and several options
- RCT – reversible component transform is used with the 5×3 wavelet filter for lossless compression
- Decreased bit rates and increased compression ratios achieved by truncation during the quantization step (decrease in image quality)

- Purely lossy coding
- YCbCr transforms RGB signal to intensity component Y and two color components (blue/red)

$$\begin{aligned} Y &= +0.299\,R + 0.587\,G + 0.114\,B\,, \\ C_b &= -0.168736\,R - 0.331264\,G + 0.5\,B\,, \\ C_r &= +0.5\,R - 0.418688\,G - 0.081312\,B\,. \end{aligned} \tag{14.12}$$

- folloed by 9×7 wavelet transform
- then arbitrary quantization by division in addition to truncation

- such main paths have several options for identification of the region of interest, coding options to trade complexity and performance, and choices about the amount of scalability in the bitstream

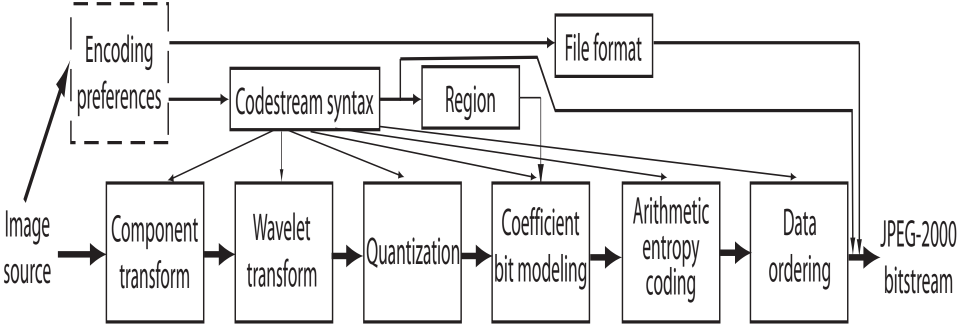


Figure 14.7: Main data path of JPEG–2000 data compression.

- image is divided into rectangular, non-overlapping tiles on a regular grid
- border tiles may be sized as needed
- arbitrary tile sizes allowed, up to using a single tile representing the entire image

- **component transform** block — input: original image data ... decorrelates image components of multi-band image—typically the R,G,B channels of the color image
- decorrelation yields improved compression performance
- allows for visually relevant quantization

- when lossy (irreversible) path is used the floating-point YCbCr transform is employed in the same way as it is used in the original color JPEG compression.

- **wavelet transform** is the heart of the JPEG–2000 compression
- can be performed in two ways
- both ways provide lower resolution images and spatial decorrelation of the images
  - 9×7 biorthogonal Daubechies filter – highest compression
  - Le Gall 5×3 filter is of lower complexity – lossless compression
- advanced parts of JPEG–2000
  - simultaneous use of multiple wavelets
  - including user-defined wavelet transforms for which coefficients are specified in bitstream
- blocky character of JPEG image - most typical artifact
- wavelet compression can be applied to the entire image
  converted into a series of wavelets
  $\Rightarrow$ blockiness may be completely removed
- even if block-based wavelet transformation is employed, the blockiness is substantially decreased

- **quantization** step offers trade-off between compression ratio and image quality
- similar to JPEG, wavelet coefficients can be divided by a different value for each image subband
- some coded data can be discarded to increase compression ratio
- **codestream syntax** prescribes marker segments, which determine the location of the coded data with respect to a given spatial image location, resolution, and quality

Application 1

- web – JPEG–2000 allows initial and quick display of low resolution image (map)
- later, any part of image (map) can be requested via the region of interest selection, server only provides necessary additional data for that spatial region at the required resolution
- if user requests a printout of that region of interest, a higher resolution version that is matched to the printer resolution may be fetched
- based on gray-level or color printer capabilities, only grayscale or color information would be transferred.

- $\Rightarrow$ selective transmission of only necessary data by the specific application is an inherent and intriguing feature of the JPEG–2000 standard

Application 2

- storing high resolution digital photographs ... running out of space
- currently, one photograph must be deleted before we can store another image
- if stored using JPEG–2000, possible slightly to decrease quality of all stored images
  ... make space for that one more important photograph to be taken and stored, or archived

- JPEG–2000 is much better compression tool than JPEG when high image quality is demanded, even when using lossy compression

- for lossy compression, JPEG–2000 can typically compress images 20–200% more than JPEG

- JPEG–2000 can handle up to 256 image channels while original JPEG was, due to its common implementation, limited to only 3-channel color data

- JPEG–2000 compression ratios of about 2.5 are typical for lossless compression

- Replacing *Motion JPEG* (for editing production-quality video, but no international standard),
  JPEG–2000 includes standardized **Motion JPEG-2000** format

- JPEG–2000 shall be the compression standard of choice
- but – original JPEG standard is not likely to disappear quickly

14.9.3 MPEG—full-motion video compression

Video and associated audio data can be compressed using MPEG compression algorithms. Using inter-frame compression, compression ratios of 200 can be achieved in full-motion, motion-intensive video applications maintaining reasonable quality. MPEG compression facilitates the following features of the compressed video; random access, fast forward/reverse searches, reverse playback, audio-visual synchronization, robustness to error, editability, format flexibility, and cost trade-off [LeGall 91; Steinmetz 94]. Three standards are frequently cited:

- MPEG-1 for compression of low-resolution (320×240) full-motion video at rates of 1–1.5 Mb/s
- MPEG-2 for higher-resolution standards such as TV and HDTV at rates of 2–80 Mb/s
- MPEG-4 for small-frame full motion compression with slow refresh needs, rates of 9–40 kb/s for video telephony and interactive multimedia such as video conferencing

MPEG can be equally well used for both symmetric and asymmetric applications. Here, MPEG video compression will be described; a description of the audio compression that is also part of the MPEG standard can be found elsewhere [Pennebaker and Mitchell 93; Steinmetz 94].

The video data consist of a sequence of image frames. In the MPEG compression scheme, three frame types are defined: **intraframes I**; **predicted frames P**; and

forward, backward, or bi-directionally predicted or interpolated frames B. Each frame type is coded using a different algorithm; Figure 14.8 shows how the frame types may be positioned in the sequence.

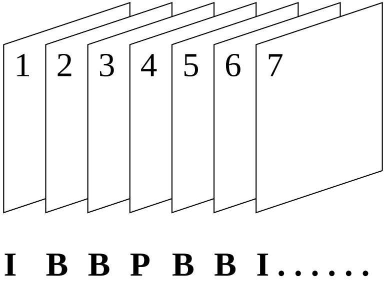


Figure 14.8: MPEG image frames.

I-frames are self-contained and coded using a DCT-based compression method similar to JPEG. Thus, I-frames serve as random access frames in MPEG frame streams. Consequently, I-frames are compressed with the lowest compression ratios. P-frames are coded using forward predictive coding with reference to the previous I- or P-frame, and the compression ratio for P-frames is substantially higher than that for I-frames. B-frames are coded using forward, backward, or bi-directional motion-compensated prediction or interpolation using two reference frames, closest past and future I- or P-frames, and offer the highest compression ratios.

Note that in the hypothetical MPEG stream shown in Figure 14.8, the frames must be transmitted in the following sequence (subscripts denote frame numbers): I_1–P_4–B_2–B_3–I_7–B_5–B_6– etc.; the frames B_2 and B_3 must be transmitted after frame P_4 to enable frame interpolation used for B-frame decompression. Clearly, the highest compression ratios can be achieved by incorporation of a large number of B-frames; if only I-frames are used, MJPEG compression results. The following sequence seems to be effective for a large number of applications [Steinmetz 94]

$$(\mathrm{I\ B\ B\ P\ B\ B\ P\ B\ B\ })(\mathrm{I\ B\ B\ P\ B\ B\ P\ B\ B\ })\ldots \tag{14.13}$$

While coding the I-frames is straightforward, coding of P- and B-frames incorporates motion estimation (see also Chapter **??**). For every 16×16 block of P- or B-frames, one motion vector is determined for P- and forward or backward predicted B-frames, two motion vectors are calculated for interpolated B-frames. The motion estimation technique is not specified in the MPEG standard, but block matching techniques are widely used, generally following the matching approaches presented in Section **??**, equations (**??**)–(**??**) [Furht et al. 95]. After the motion vectors are estimated, differences between the predicted and actual blocks are determined and represent the error terms which are encoded using DCT. As usually, entropy encoding is employed as the final step.

MPEG-1 decoders are widely used in video players for multimedia applications and on the World Wide Web.

14.10 Summary

- **Image data compression**
  - The main goal of image compression is to minimize image data volume with no significant loss of information.
  - Image compression algorithms aim to remove redundancy present in data (correlation of data) in a way which makes image reconstruction possible; this is called **information preserving compression**.
  - A typical image **compression/decompression** sequence consists of data redundancy reduction, coding, transmission, decoding, and reconstruction.
  - Data compression methods can be divided into two principal groups:
    * **Information preserving** compressions permit error-free data reconstruction (**lossless compression**).
    * Compression methods **with loss of information** do not preserve the information completely (**lossy compression**).
- **Image data properties**
  - Information content of an image is an important property of which **entropy** is a measure.
  - Knowing image entropy, information **redundancy** can be determined.
- **Discrete image transforms in image data compression**

- Image data are represented by **coefficients** of discrete image transforms. The transform coefficients are **ordered** according to their importance, i.e., according to their contribution to the image information contents, and the least important (low-contribution) coefficients are omitted.
- To remove correlated (redundant) image data, the **Karhunen-Loève** transform is the most effective.
- **Cosine, Fourier, Hadamard, Walsh**, or **binary** transforms are all suitable for image data compression.
- Performance of **discrete cosine transform DCT-II** approaches that of the Karhunen-Loève transform better than others. The DCT is usually applied to small image blocks (typically 8×8 pixels), yielding quality-decreasing blocking artifacts for larger compression ratios.
- Consequently, **wavelet image compression** is of interest because it does not generate square image compression artifacts.

- **Predictive compression methods**
  - Predictive compressions use image information redundancy to construct an **estimate** of the gray-level value of an image element from values of gray-levels in its neighborhood.
  - The **differences** between estimates and reality, which are expected to be relatively small in absolute terms, are coded and transmitted together with prediction model parameters.

- **Vector quantization**
  - Vector quantization compression is based on dividing an image into small blocks and representing these blocks as **vectors**.
  - Input data vectors are coded using unique **codewords** from a **codeword dictionary**; instead of vectors, the vector codes are stored or transmitted.
  - The code dictionary (code book) is transmitted together with the coded data.

- **Hierarchical and progressive compression methods**
  - Substantial reduction in bit volume can be obtained by merely representing a source as a pyramid. Even more significant reduction can be achieved for images with large areas of the same gray-level in a quadtree coding scheme.
  - Hierarchical compression facilitates **progressive** and **smart** image transmission.
  - Progressive image transmission is based on the fact that transmitting all image data may not be necessary under some circumstances.
  - Smart compression is based on the sensing properties of human visual sensors—it is not necessary to display an image in full resolution in image areas where the user's eyes are not focused.

- **Comparison of compression methods**

- Transform-based methods better preserve subjective image quality, and are less sensitive to statistical image property changes both inside a single image and between images.
- Prediction methods can achieve larger compression ratios in a much less expensive way, and tend to be much faster than transform-based or vector quantization compression schemes.
- Vector quantization methods require a complex coder, their parameters are very sensitive to image data, and they blur image edges. The advantage is in a simple decoding scheme consisting of a look-up table only.

- **Other techniques**
  - Various other image data compression methods exist.
  - **Fractal image compression** offers extremely high compression ratios and high-quality image reconstruction. Breaking an image into pieces (fractals) and identifying self-similar ones is the main principle of the approach. Fractals are infinitely magnifiable, thus fractal compression is **resolution independent** and a single compressed image can be efficiently used for display in any image resolution.

- **Coding**
  - **Huffman encoding** can provide optimal compression and error-free decompression. The main idea of Huffman coding is to represent data by

codes of variable length, with more frequent data being represented by shorter codes.

- **JPEG and MPEG image compression**
  - JPEG and JPEG-2000 represent international **standards** in image compression.
  - JPEG image compression was developed for general-purpose, color, still image compression. This standard is widely used in many application areas. Four JPEG compression modes exist:
    * Sequential DCT-based compression
    * Progressive DCT-based compression
    * Sequential lossless predictive compression
    * Hierarchical lossy or lossless compression
  - JPEG-2000 is designed to overcome some limitations of the JPEG standard. Despite the naming similarity, it is not an extension of the earlier JPEG standard; rather, it is a new image compression approach.
  - JPEG-2000 is wavelet-transform based and offers a rich and flexible set of new functionalities in respect of quality, resolution, and spatial scalability.
  - JPEG-2000 typically outperforms JPEG compression in applications requiring either high quality image reconstruction or low bitrate compression.
  - The **MPEG standard** was developed for full-motion video image sequences.
  - Three standards are frequently cited:

* MPEG-1 for compression of low-resolution full-motion video
* MPEG-2 for higher-resolution standards
* MPEG-4 for small-frame full-motion compression with slow refresh needs

14.11 References

[Aldus 92] Aldus. *TIFF Developer's Toolkit, Revision 6.0.* Aldus Corporation, Seattle, WA, 1992.

[Barnsley and Hurd 93] M Barnsley and L Hurd. *Fractal Image Compression.* A K Peters Ltd., Wellesley, MA, 1993.

[Boliek et al. 00] M. Boliek, S. Houchin, and G. Wu. JPEG 2000 next generation image compression system: Features and syntax. In *Proceedings ICIP-2000, Vol. 2*, pages 45–48. IEEE, 2000.

[Boxerman and Lee 90] J L Boxerman and H J Lee. Variable block-sized vector quantization of grayscale images with unconstrained tiling. In *Visual Communications and Image Processing '90,* Lausanne, Switzerland, pages 847–858, SPIE, Bellingham, WA, 1990.

[Chang et al. 88] C Y Chang, R Kwok, and J C Curlander. Spatial compression of Seasat SAR images. *IEEE Transactions on Geoscience and Remote Sensing*, 26(5):673–685, 1988.

[Clarke 85] R J Clarke. *Transform Coding of Images.* Academic Press, London, 1985.

[Clarke 95] R J Clarke. *Digital Compression of Still Images and Video.* Academic Press, London, New York, 1995.

[Colyer and Clark 03] Greg Colyer and Richard Clark. *Guide to the practical implementation of JPEG 2000 – PD 6777.* British Standards Institute, London, 2003.

[Compuserve 89] Compuserve. *Graphics Interchange Format: Version 89a.* CompuServe Incorporated, Columbus, OH, 1989.

[Daut and Zhao 90] D G Daut and D Zhao. Improved DPCM algorithm for image data compression. In *Image Processing Algorithms and Techniques,* Santa Clara, CA, pages 199–210, SPIE, Bellingham, WA, 1990.

[De Lameillieure and Bruyland 90] J De Lameillieure and I Bruyland. Single stage 280 Mbps coding of HDTV using HDPCM with a vector quantizer based on masking functions. *Signal Processing: Image Communication*, 2(3):279–289, 1990.

[Delp and Mitchell 79] E J Delp and O R Mitchell. Image truncation using block truncation coding. *IEEE Transactions on Communications*, 27:1335–1342, 1979.

[DiMento and Berkovich 90] L J DiMento and S Y Berkovich. The compression effects of the binary tree overlapping method on digital imagery. *IEEE Transactions on Communications*, 38(8):1260–1265, 1990.

[Farelle 90] P M Farelle. *Recursive Block Coding for Image Data Compression.* Springer Verlag, New York, 1990.

[Fisher 94] Y Fisher. *Fractal Compression: Theory and Applications to Digital Images.* Springer Verlag, Berlin, New York, 1994.

[Furht et al. 95] B Furht, S W Smoliar, and H Zhang. *Video and Image Processing in Multimedia Systems.* Kluwer, Boston–Dordrecht–London, 1995.

[Gersho and Gray 92] A Gersho and R M Gray. *Vector Quantization and Signal Compression*. Kluwer, Norwell, MA, 1992.

[Gormish et al. 00] M. J. Gormish, D. Lee, and M. W. Marcellin. JPEG 2000: Overview, architecture, and applications. In *Proceedings ICIP-2000, Vol. 2*, pages 29–32, 2000.

[Graham 67] D N Graham. Image transmission by two–dimensional contour coding. *Proceedings IEEE*, 55:336–346, 1967.

[Gray 84] R M Gray. Vector quantization. *IEEE ASSP Magazine*, 1(2):4–29, 1984.

[Huffman 52] D A Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of IRE*, 40(9):1098–1101, 1952.

[Hung and Meng 94] A C Hung and T H Y Meng. A comparison of fast inverse discrete cosine transform algorithms. *Journal of Multimedia Systems*, 2:204–217, 1994.

[Jaisimha et al. 89] M Y Jaisimha, H Potlapalli, H Barad, and A B Martinez. Data compression techniques for maps. In *Energy and Information Technologies in the Southeast,* Columbia, SC, pages 878–883, IEEE, Piscataway, NJ, 1989.

[Knuth 85] D E Knuth. Dynamic Huffman coding. *Journal of Algorithms*, 6:163–180, 1985.

[Kruger 92] A Kruger. Block truncation compression. *Dr Dobb's J Software Tools*, 17(4):48–55, 1992.

[LeGall 91] D LeGall. MPEG: A video compression standard for multimedia applications. *Communications of the ACM*, 34:45–68, 1991.

[Marcellin et al. 00] Michael W. Marcellin, Michael J. Gormish, Ali Bilgin, and Martin P. Boliek. An overview of JPEG-2000. In *Proc. of IEEE Data Compression Conference*, pages 523–541. IEEE, 2000.

Moik 80] J G Moik. *Digital Processing of Remotely Sensed Images*. NASA SP–431, Washington, DC, 1980.

Nelson 89] M R Nelson. LZW data compression. *Dr Dobb's J Software Tools*, 14, 1989.

Netravali 88] A N Netravali. *Digital Pictures: Representation and Compression*. Plenum Press, New York, 1988.

Park and Lee 91] S H Park and S U Lee. Pyramid image coder using classified transform vector quantization. *Signal Processing*, 22(1):25–42, 1991.

Pennebaker and Mitchell 93] W B Pennebaker and J L Mitchell. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, 1993.

Pratt 91] W K Pratt. *Digital Image Processing*. Wiley, New York, 2nd edition, 1991.

Rabbani 91] M Rabbani. *Digital Image Compression*. SPIE Optical Engineering Press, Bellingham, WA, 1991.

Rao and Pearlman 91] R P Rao and W A Pearlman. On entropy of pyramid structures. *IEEE Transactions on Information Theory*, 37(2):407–413, 1991.

Rosenfeld and Kak 82] A Rosenfeld and A C Kak. *Digital Picture Processing*. Academic Press, New York, 2nd edition, 1982.

hi and Sun 99] Yun Q. Shi and Huifang Sun. *Image and Video Compression for Multimedia Engineering: Fundamentals, Algorithms, and Standards*. CRC Press, Boca Raton, FL, 1999.

ilsbee et al. 91] P Silsbee, A C Bovik, and D Chen. Visual pattern image sequencing coding. In *Visual Communications and Image Processing '90*, Lausanne, Switzerland, pages 532–543, SPIE, Bellingham, WA, 1991.

Steinmetz 94] R Steinmetz. Data compression in multimedia computing—standards and systems, parts i and ii. *Journal of Multimedia Systems*, 1:166–172 and 187–204, 1994.

Strobach 90] P Strobach. Tree-structured scene adaptive coder. *IEEE Transactions on Communications*, 38(4):477–486, 1990.

Taubman and Marcellin 01] David S. Taubman and Michael W. Marcellin. *JPEG 2000: Image Compression Fundamentals, Standards and Practice*. Kluwer, Boston, MA, 2001.

Taubman et al. 00] D. S. Taubman, E. Ordentlich, M. Weinberger, G. Seroussi, I. Ueno, and F. Ono. Embedded block coding in JPEG 2000. In *Proceedings ICIP-2000, Vol. 2*, pages 33–36. IEEE, 2000.

Vitter 87] J S Vitter. Design and analysis of dynamic Huffman codes. *Journal of the ACM*, 34(4):825–845, 1987.

Wang and Goldberg 89] L Wang and M Goldberg. Reduced-difference pyramid: A data structure for progressive image transmission. *Optical Engineering*, 28(7):708–716, 1989.

White 87] R G White. Compressing image data with quadtrees. *Dr Dobb's J Software Tools*, 12(3):16–45, 1987.

Wilkins and Wintz 71] L C Wilkins and P A Wintz. Bibliography on data compression, picture properties and picture coding. *IEEE Transactions on Information Theory*, 17:180–199, 1971.

Witten et al. 94] I H Witten, A Moffat, and T C Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Van Nostrand Reinhold, New York, 1994.

Zailu and Taxiao 90] H Zailu and W Taxiao. MDPCM picture coding. In *1990 IEEE International Symposium on Circuits and Systems*, New Orleans, LA, pages 3253–3255, IEEE, Piscataway, NJ, 1990.

eng et al. 00] W. Zeng, S. Daly, and S. Lei. Visual optimization tools in JPEG 2000. In *Proceedings ICIP-2000, Vol. 2*, pages 37–40. IEEE, 2000.

iv and Lempel 78] J Ziv and A Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, 1978.