We are witnessing phenomenal increases in the use of images in many different applications. This is mainly due to: 1) technological advances impacting several image operations; 2) the availability of sophisticated software tools for the manipulation and management, and 3) the World Wide Web (WWW) providing easy access to a wide range of users. Typical applications using huge amounts of images are medical imaging, remote sensing, entertainment, digital libraries, distance learning and training and multimedia.

Digital images require huge amounts of space for storage and large bandwidths for transmission. For example, a single 640×480 pixel color image using 24 bits/pixel requires close to one megabyte of space. Despite the technological advances in storage and transmission, the demands placed on the storage capacities and on the bandwidth of communication exceed the availability. *Image compression* has proved to be a viable technique as one solution response.

Digital images generally contain significant amounts of *spatial* and *spectral* redundancy. Spatial redundancy is due to the correlation between neighboring pixel values, and spectral redundancy is due to the correlation between different color planes. *Image compression* (*coding*) techniques reduce the number of bits required to represent an image by taking advantage of these redundancies. An inverse process called *decompression* (*decoding*) is applied to the compressed data to get the reconstructed image. The objective of compression is to reduce the number of bits as much as possible, while keeping the resolution and the visual quality of the reconstructed image as close to the original image as possible.

This article gives an overview of the major image compression techniques. The decoding steps for most of the coding schemes are quite intuitive and are usually the reverse of the encoding steps. The reader is referred to the "Read more about it" for the details. In this article, the terms *compression* and *coding* are used synonymously.

Basics of image representation

An *image* is essentially a 2-D signal processed by the Human Visual System. The signals representing images are usually in analog form. However, for processing, storage and transmission by computer applications, they are converted from analog to digital form. For display and presentation, however, they usually need to be in analog form.

In this article, the term "image" refers to "digital image." A *digital image* is basically a 2-dimensional array of *pixels* (picture elements). An image whose pixels have one of only two intensity levels (black and white) is called a *bi-tonal* (or *bi-level*) image. Printed text on paper is a common example of this class of images.

In a *continuous-tone* image, the pixels have a range of values. For example, in a typical gray-scale image, the pixels could have values in the range [0 - 255], representing different gray levels.

In a typical color image used for display, each pixel has three color components (*R, G, B*) corresponding to the three primary colors, red, green and blue. Each pixel of a typical color image to be transmitted has three components (*Y, I, Q*), where *Y* is the luminance (brightness) component and *I* and *Q* are chrominance (color) components. Each component of (*R, G, B*) or (*Y, I, Q*) requires 8 bits/pixel. Thus, color images (usually) require 24 bits/pixel. The number of pixels in each dimension in an image defines the image's *resolution*—more pixels mean more details are seen in the image.

The taxonomy

The image compression techniques are broadly classified as either *lossless* or *lossy*, depending, respectively, on whether or not an exact replica of the original image could be reconstructed using the compressed image. Lossless compression is also referred to as *entropy coding*. In addition to using the spatial and spectral redundancies, lossy techniques also take advantage of the way people see to discard data that are perceptually insignificant.

Lossy schemes provide much higher compression ratios than lossless schemes. Lossless compression is used only for a few applications with stringent requirements such as medical imaging. Lossy schemes are widely used since the quality of the reconstructed images is adequate for most applications. A taxonomy of image compression techniques is given in Fig. 1.

Practical compression systems and standards use *hybrid coding*. This is a combination of several basic lossy coding techniques. They include: a) transform coding and predictive coding, b) subband coding and transform coding and c) predictive coding and vector quantization. In addition, the output of the lossy coding scheme is further compressed using a lossless coding scheme such as Huffman or Arithmetic coding.

Lossless image compression

In lossless compression techniques, the original image can be perfectly recovered from the compressed (encoded) image. These are also called *noiseless* [since they do not add noise to the signal

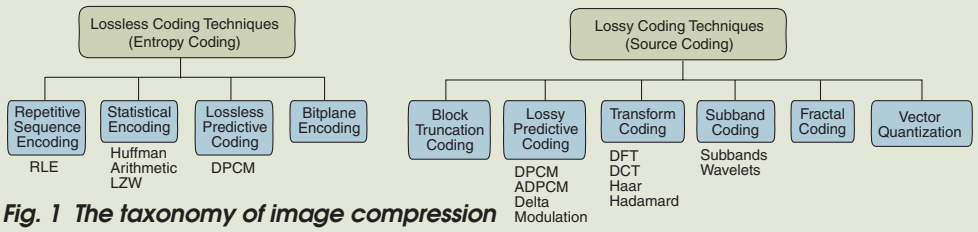

Fig. 1 The taxonomy of image compression

(image)], or *entropy coding* (since they use statistical/decomposition techniques to eliminate/minimize redundancy).

Run length encoding. This technique replaces sequences of identical symbols (pixels), called *runs* by shorter symbols. This technique is usually used as a post-processing step after applying a lossy technique to the image and obtaining a set of data values that are suitably re-ordered to get long runs of similar values.

Huffman coding. This is a general technique for coding symbols based on their statistical occurrence frequencies (probabilities). The pixels in the image are treated as symbols. The symbols that occur more frequently are assigned a smaller number of bits, while the symbols that occur less frequently are assigned a relatively larger number of bits. Huffman code is a *prefix code*. This means that the (binary) code of any symbol is not the prefix of the code of any other symbol. Most image coding standards use lossy techniques in the earlier stages of compression and use Huffman coding as the final step.

Arithmetic coding. Like Huffman coding, this is a statistical technique. However, instead of coding each symbol separately, the whole data sequence is coded with a single code. Thus, the correlation between neighboring pixels is exploited. Arithmetic coding is based on the following principle. Given that a) the symbol alphabet is finite; b) all possible symbol sequences of a given length are finite; c) all possible sequences are countably infinite; d) the number of real numbers in the interval [0, 1] is uncountably infinite, we can assign a unique subinterval for any given input (sequence of symbols). This is the code (tag) for the input.

The *cumulative density function* (CDF) of the symbol probabilities is used to partition the interval (usually [0, 1]) into subintervals and map the sequence of symbols to a unique subinterval. This scheme is well suited to small set of symbols with highly skewed probabilities of occurrence. Arithmetic coding is used as the final step in several image coding applications and standards.

Lempel-Ziv coding. This is based on storing frequently occurring sequences of symbols (pixels) in a dictionary (table). Such frequently occurring sequences in the original data (image) are represented by just their indices into the dictionary. This has been used in *TIFF* (Tagged Image File Format) and *GIF* (Graphical Interchange Format) file formats. This scheme has also been used for compressing half-tone images. (Half-tone images are binary images that provide the visual effect of continuous-tone gray images by using variations of the density of black dots in the images).

Predictive coding. This is based on the assumption that the pixels in images conform to the *autoregressive model*, where each pixel is a linear combination of its immediate neighbors. The lossless differential pulse code modulation (DPCM) technique is the most common type of lossless predictive coding. In the lossless DPCM scheme, each pixel value (except at the boundaries) of the original image is first predicted based on its neighbors to get a *predicted image*. Then the difference between the actual and the predicted pixel values is computed to get the *differential* or *residual image*. The residual image will have a much less dynamic range of pixel values. This image is then efficiently encoded using Huffman coding.

Bit-plane encoding. In this scheme, the binary representations of the values of the pixels in the image are considered. The corresponding bits in each of the positions in the binary representation form a binary image of the same dimensions as the original image. This is called a *bit plane*. Each of the bit planes can then be efficiently coded using a lossless technique.

The underlying principle is that (in most images) the neighboring pixels are correlated. That means the values of the neighboring pixels differ by small amounts. They can be captured by the representation of pixel values in *gray code* so that the values of neighboring bits in the bit planes are similar. This makes the individual bit planes amenable for good compression.

Lossy image compression

All known lossy image compression techniques take advantage of how we see things. The human visual system is more sensitive to the lower frequencies than to the higher frequencies in the visual spectrum. Thus, we derive the (spatial) frequencies of an image and suitably allocate more bits for those frequency components that have more visual impact. We then allocate less bits, or even discard, the insignificant components. The resulting image is represented with fewer bits and reconstructed with a better closeness to the original.

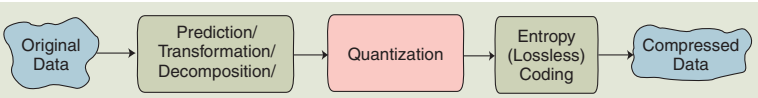

Fig. 2 Outline of lossy image compression

To achieve this goal, one of the following operations is generally performed: 1) a *predicted image* is formed. Its pixels are predicted based on the values of neighboring pixel of the original image, and then a *differential* (residual) image is derived (it is the difference between the original and the predicted image.); 2) a *transformed image* is derived by applying a transform to the original image. This essentially transforms the pixel values to the frequency domain; 3) the original image is decomposed into different components (in the frequency domain).

In 1), the dynamic range of the signal values is reduced; in 2) and 3) a representation that is more efficiently coded is derived. In each case, there exits an inverse operation, which yields the original image (lossless), when it is applied to the new representation.

However, to achieve compression the "redundant" information the human eye considers perceptually insignificant is discarded. This is done using *quantization*. The new representation has desirable properties. The quantized data has much less variance than the original. Entropy coding is then applied to achieve further compression.

The outline of lossy compression techniques is shown in Fig. 2. Please note that the prediction-transformation-decomposition process is completely reversible. The *quantization* process (see box) results in loss of information. The

entropy coding after the quantization step, however, is lossless. The decoding is a similar but reverse process: a) entropy decoding is applied to the compressed data to get the quantized data, b) dequantization is applied to it, and then c) the inverse transformation to get the *reconstructed image*. (This is an approximation of the original image.)

Major performance considerations of a lossy compression scheme are: a) the compression ratio (CR), b) the signal-to-noise ratio (SNR) of the reconstructed image with respect to the original, and c) the speed of encoding and decoding. The compression ratio is given by:

$$\mathrm{CR} = \frac{\text{size of uncompressed data}}{\text{size of compressed data}}$$

The PSNR is given by:

$\mathrm{PSNR} = 20 \log_{10}(\text{peak data value/RMSE})$

where RMSE is the root mean square error, given by:

$$\mathrm{RMSE} = \sqrt{\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\left[I_{i,j} - \hat{I}_{i,j}\right]^2}$$

where $N \times M$ is the image size, $I_{i,j}$ and $I_{i,j}$ are values of pixels at (i,j) in the original and the reconstructed (compressed-decompressed) images, respectively.

Predictive coding. In most images, there is a high correlation among neighboring pixels. This fact is used in predictive coding. Differential Pulse Code Modulation (DPCM) is a popular predictive coding technique. The lossy DPCM is very similar to the lossless version. The major difference is that in lossy DPCM, the pixels are predicted based on the "reconstructed values" of certain neighboring pixels. The difference between the predicted value and the actual value of the pixels is the differential (residual) image. It is much less correlated than the original image. The differential image is then quantized and encoded.

The schematic for lossy DPCM coder is shown in Fig. 3, along with a third-order predictor. (In a third-order predictor, three previous values are used to predict each pixel.) Note that the decoder has access only to the reconstructed values of (previous) pixels while forming predictions of pixels. Since the quantization of the differential image introduces error, the reconstructed values generally differ from the original values. To ensure identical predictions at both the encoder and decoder, the encoder also uses the "reconstructed pixel values" in its prediction. This is done by using the quantizer within the prediction loop. (In essence, the decoder is built into the encoder).

The design of a DPCM coder involves optimizing the predictor and the quantizer. The inclusion of the quantizer in the prediction loop results in a complex dependency between the prediction error and the quantization error. However, the predictor and quantizer are usually optimized separately, since a joint optimization is usually complex. [Under mean-squared error (MSE) optimization criterion, independent optimizations of the predictor and quantizer are good approximations to the jointly optimized solution.]

Block truncation coding. In this scheme, the image is divided into non-overlapping blocks of pixels. For each block, *threshold* and *reconstruction* values are determined. The threshold is usually the mean of the pixel values in the block. Then a bitmap of the block is derived by replacing all pixels whose values are greater than or equal (less than) to the threshold by a 1 (0). Then for each segment (group of 1s and 0s) in the bitmap, the reconstruction value is determined. This is the average of the values of the corresponding pixels in the original block. The broad outline of block truncation coding of images is shown in Fig. 4.

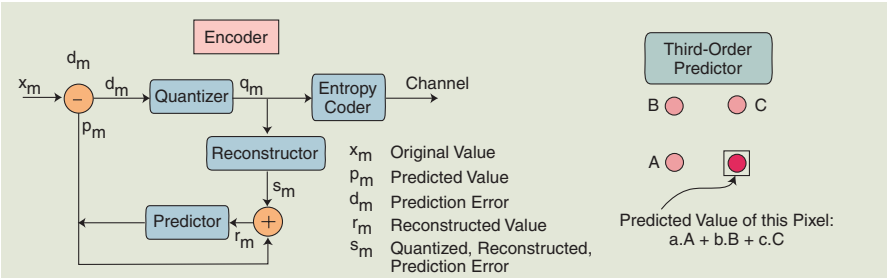

Fig. 3 Lossy DPCM coding scheme

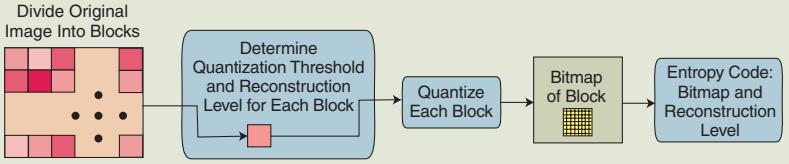

Fig. 4 Outline of block truncation coding of images

Transform coding. In this coding scheme, transforms such as DFT (Discrete Fourier Transform) and DCT (Discrete Cosine Transform) are used to change the pixels in the original image into frequency domain coefficients (called transform coefficients).

These coefficients have several desirable properties. One is the energy compaction property that results in most of the *energy* of the original data being concentrated in only a few of the *significant* transform coefficients. This is the basis of achieving the compression. Only those few significant coefficients are selected and the remaining are discarded. The selected coefficients are considered for further quantization and entropy encoding. DCT coding has been the most common approach to transform coding. It is also adopted in the JPEG image compression standard. The broad outline of transform coding of images is shown in Fig. 5.

Subband coding. In this scheme, the image is analyzed to produce the components containing frequencies in well-defined bands, the *subbands*. Subsequently, quantization and coding is applied to each of the bands. The advantage of this scheme is that the quantization and coding well suited for each of the subbands can be designed separately. The broad outline of transform coding of images is shown in Fig. 6.

Vector quantization. The basic idea in this technique is to develop a *dictionary* of fixed-size *vectors*, called *code vectors*. A vector is usually a block of pixel values. A given image is then partitioned into non-overlapping blocks (vectors) called *image vectors*. Then for each image vector, the closest matching vector in the dictionary is determined and its index in the dictionary is used as the encoding of the original image vector. Thus, each image is represented by a sequence of indices that can be further entropy coded. The outline of the scheme is shown in Fig. 7.

Fractal coding. The essential idea here is to decompose the image into *seg-*

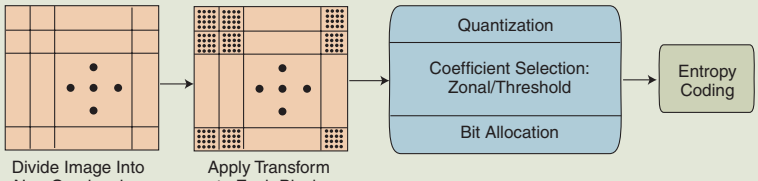

Fig. 5 Outline of transform coding of images

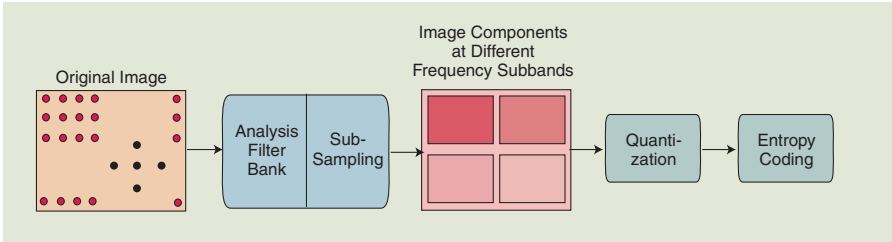

Fig. 6 Outline of subband coding of images

ments by using standard image processing techniques such as color separation, edge detection, and spectrum and texture analysis. Then each segment is looked up in a library of fractals. The library actually contains *codes* called iterated function system (IFS) codes, which are compact sets of numbers. Using a systematic procedure, a set of codes for a given image are determined, such that when the IFS codes are applied to a suitable set of image blocks yield an image that is a very close approximation of the original. This scheme is highly effective for compressing images that have good regularity and self-similarity. The broad outline of fractal coding of images is shown in Fig. 8.

Image compression standards

Image compression standards have been developed to facilitate the interoperability of compression and decompression of schemes across several hardware platforms, operating systems and applications. Most standards are hybrid systems making use of a few of the basic techniques already mentioned. The major image compression standards are Group 3, Group 4, and JBIG (Joint Bi-level Image Group) for bi-tonal images, and JPEG (Joint Photographic Experts Group) for continuous-tone images. The most common application that uses compression of bi-tonal images is digital facsimile (FAX).

Group 3 Fax

The image is scanned left-to-right and top-to-bottom and the runs of each color—black and white—are determined. A run refers to a sequence of consecutive pixels of the same value. The first run on each line is assumed to be white. Each line is considered to be made up of 1728 pixels. Thus each line is reduced to alternating runs of white and black pixels. The runs are then encoded. Each end of a line is marked with an EOL. Page breaks are denoted with two successive EOLs.

Two types of encodings are used for run lengths—terminating codes and make-up codes. Terminating codes are used for runs with lengths less than 64. For runs with length greater than 64, a make-up code is followed by a terminating code. The make-up codes represent run lengths of multiples of 64 (64,128,192,...). Tables specifying the terminating codes and make-up codes for the white and black runs are provided by the standard.

Group 4 Fax

The Group 4 (G4) fax standard is a superset of the Group 3 standard and is backwards compatible with it. The G4 standard is said to use a 2-dimensional coding scheme. This is because it uses spatial redundancy in the vertical direction also by making use of the previous line as a reference while coding the current line.

Most runs on a line usually lie nearly directly below a run of the same color in the previous line. The differences in the run boundaries between successive lines are coded. The cases where a line may have fewer or more lines than the reference lines are suitably handled. The Group 4 standard generally provides more efficient compression than Group 3.

JBIG

The Joint Bi-Level Image Group (JBIG) standard was developed by the

Scalar quantization

Quantization is a process (function) that maps a very large (possibly infinite) set of values to a much smaller (finite) set of values. In scalar quantization, the values that are mapped are scalars (numbers). In the context of image coding and decoding, the range of pixel values say N, is divided into L non-overlapping intervals, also known as *quantization levels*.

Each interval i is defined by its *decision boundaries* (d_i, d_{i+1}). During encoding, the quantizer maps a given pixel value x to a quantization level l: $l = Q(x)$, such that $d_l \le x < d_{l+1}$. Each quantization level i has its associated *reconstruction level* r_i.

During decoding, the (de)quantizer maps a given level l to a reconstruction pixel value $r_l = \hat{x}$, $\hat{x} = Q^{-1}(l)$. This introduces noise or error in the image (signal) called *quantization error*. This is the root mean square value of the $x - \hat{x}$.

The essential difference among various types of quantizers is in terms of how the forward and inverse mappings are defined. These definitions are dictated according to the number of quantization levels, the decision boundaries and the reconstruction values. The basic design objective of a quantizer is to minimize the quantization error, while being computationally simple. The quantizer has a large impact on the compression ratio and image quality of a lossy scheme.

There are two broad types of scalar quantizers—uniform and non-uniform. In a uniform quantizer of k levels, the range of values is divided into k equally spaced intervals. The reconstruction values are the mid-points of the intervals. This is simple to implement but it does not attempt to minimize the quantization error. A quantizer that takes into account the probability distributions of the pixels in images performs better. Such a quantizer is a non-uniform quantizer, where the intervals are non-uniform. The most common non-uniform quantizer is the Lloyd-Max quantizer. For it, the decision boundaries and the reconstruction levels are determined using the probability model of the image pixels such that the quantization error is minimized.—*SRS*

International Standards Organization (ISO) for the lossless compression of bi-level images. Typically, these are printed pages of text whose corresponding images contain either black or white pixels.

JBIG uses a combination of bit-plane encoding and arithmetic coding. The adaptivity of the arithmetic coder to the statistics of the image results in the improved performance of JBIG. JBIG also incorporates a progressive transmission mode. This can be used for the compression of gray-scale and color images. Each bit plane of the image is treated as a bi-level image. This provides lossless compression and enables progressive buildup.

JPEG

The Joint Photographic Experts Group (JPEG) is a standard developed for compressing continuous-tone still images. JPEG has been widely accepted for still image compression throughout the industry. JPEG can be used on both gray-scale and color images. JPEG consists of four modes: *lossless*, *sequential*, *progressive* and *hierarchical*. The first one is a lossless mode and the other three are lossy modes. The sequential mode, also called *baseline* JPEG, is the most commonly used scheme.

The lossless JPEG mode uses linear predictive schemes. It provides seven different predictors. Pixel values (except those at the boundaries) are predicted based on neighboring pixels. The *residual*, which is the difference between the original and the predicted image, is encoded using entropy (lossless) coding such as Huffman or arithmetic coding.

In the baseline JPEG scheme, the image is divided into non-overlapping blocks of 8 x 8 pixels. DCT is applied to each block to obtain the transform coefficients. The coefficients are then quantized using a table specified by the standard, which contains the quantizer step sizes. The quantized coefficients are then ordered using a *zigzag* ordering. The ordered quantized values are then encoded using Huffman coding tables, specified by the standard.

Progressive JPEG compression is similar to the sequential (baseline) JPEG scheme in the formation of DCT coefficients and quantization. The key difference is that each coefficient (image component) is coded in multiple scans instead of a single scan. Each successive scan refines the image until the quality determined the quantization tables are reached.

Hierarchical JPEG compression offers a progressive representation of a decoded image similar to progressive JPEG, but also provides encoded images at multiple resolutions. Hierarchical JPEG creates a set of compressed images beginning with small images, and continuing with images with increased resolutions. This process is also called *pyramidal coding*. Hierarchical JPEG mode requires significantly more storage space, but the encoded image is immediately available at different resolutions.

Summary

The generation and use of digital images is expected to continue at an ever faster pace in the coming years. The huge size requirements of images coupled with the explosive increases are straining the storage capacities and transmission bandwidths. Compression is a viable way to overcome these bottlenecks.

All the techniques described here are considered "first-generation" techniques. The second generation of compression techniques—already underway—use a *mode-based* approach. The images are analyzed using image processing and pattern recognition techniques to derive high-level objects. The images are then described with well-defined image models. They are expressible using much less information than the original data. The challenge is in devising good models that achieve good compression without loss of fidelity.

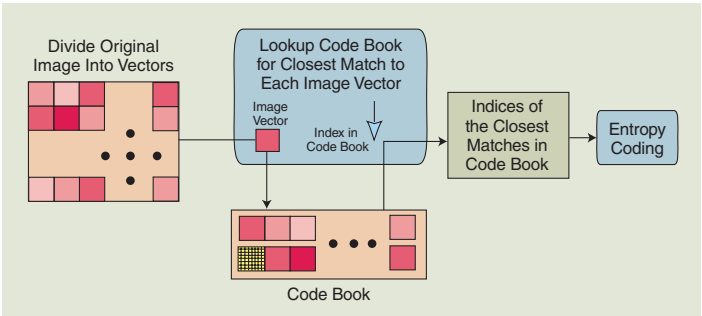

Fig. 7 Outline of vector quantization of images

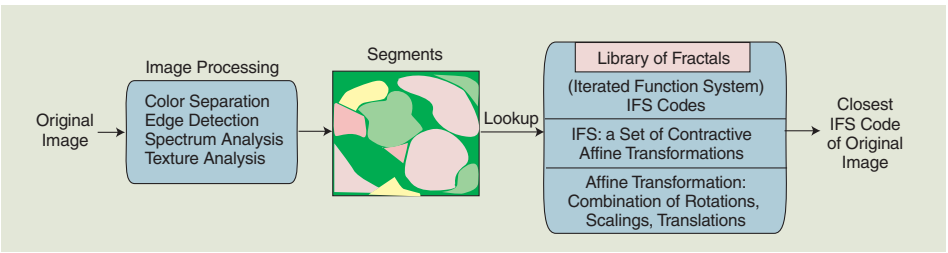

Fig. 8 Outline of fractal coding of images

Read more about it

• M. Y. Barnsley and L. P. Hurd, *Fractal Image Compression*, A. K. Peters, 1993.

• V. Bhaskaran and K. Konstantinides, *Video and Image Compression Standards*, Kluwer Academic, 1995.

• R. J. Clarke, *TransformCoding of Images*, Academic Press, London, 1985.

• A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic, 1992.

• H-M. Hang and J. W. Woods (Eds.), *Handbook of Visual Communications*, Academic Press, 1995.

• A. K. Jain, "Image Data Compression: A Review," *Proc. IEEE*, 69(3), 1981, pp. 349-389.

• W. Kou, *Digital Image Compression: Algorithms and Standards*, Kluwer Academic, 1995.

• A.N. Netravali and B.G. Haskell, *Digital Pictures: Representation, Compression, and Standards* (2nd edition), Plenum Press, 1995.

• M. Rabbani and P.W. Jones, *Digital Image Compression Techniques*, SPIE, Vol. TT7, 1991.

• K. Sayood, *Introduction to Data Compression* (2nd edition), Morgan-Kaufmann, 2000.

About the author

R. Subramanya obtained his Ph.D. in Computer Science from George Washington University, where he received the Richard Merwin memorial award from the EECS department in 1996. He received the Grant-in-Aid of Research award from Sigma-Xi for his research in audio data indexing in 1997. He is currently an Assistant Professor of Computer Science at the University of Missouri-Rolla.