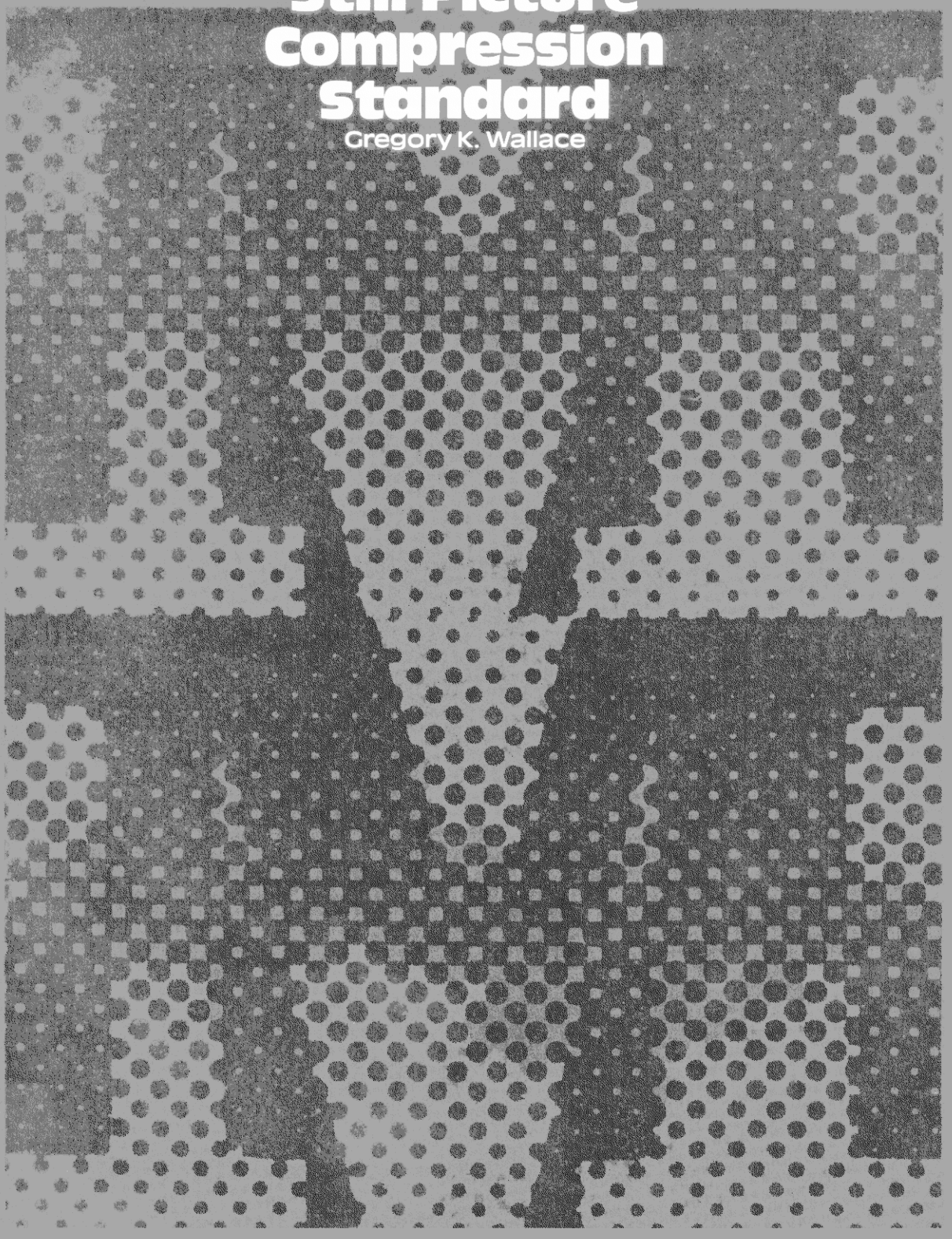
The JPEG Still Picture Compression Standard

Gregory K. Wallace



Advances over the past decade in many aspects of digital technology—especially devices for image acquisition, data storage, and bitmapped printing and display—have brought about many applications of digital imaging. However, these applications tend to be specialized due to their relatively high cost. With the possible exception of facsimile, digital images are not commonplace in general-purpose computing systems the way text and geometric graphics are. The majority of modern business and consumer usage of photographs and other types of images takes place through more traditional analog means.

The key obstacle for many applications is the vast amount of data required to represent a digital image directly. A digitized version of a single, color picture at TV resolution contains on the order of one million bytes; 35mm resolution requires ten times that amount. Use of digital images often is not viable due to high storage or transmission costs, even when image capture and display devices are quite affordable.

Modern image compression technology offers a possible solution. State-of-the-art techniques can compress typical images from 1/10 to 1/50 their uncompressed size without visibly affecting image quality. But compression technology alone is not sufficient. For digital image applications involving storage or transmission to become widespread in today's marketplace, a *standard* image compression method is needed to enable interoperability of equipment from different manufacturers. The CCITT recommendation for today's ubiquitous Group 3 fax machines [16] is a dramatic example of how a standard compression method can enable an important image application. The Group 3 method, however, deals with bilevel images only and does not address photographic image compression.

For the past few years, a standardization effort known by the acronym JPEG, for Joint Photographic Experts Group, has been working toward establishing the first international digital image compression standard for continuous-tone (multilevel) still images, both grayscale and color. The "joint" in JPEG refers to a collaboration between CCITT and ISO. JPEG convenes officially as the ISO committee designated JTC1/SC2/WG10, but operates in close informal collaboration with CCITT SGVIII.

Photovideotex, desktop publishing, graphic arts, color facsimile, newspaper wirephoto transmission, medical imaging, and many other continuous-tone image applications require a compression standard in order to develop significantly beyond their present state. JPEG has undertaken the ambitious task of developing a *general-purpose* compression standard to meet the needs of almost all continuous-tone still-image applications.

If this goal proves attainable, not only will individual applications flourish, but exchange of images across application boundaries will be facilitated. This latter feature will become increasingly important as more image applications are implemented on general-purpose computing systems, which are themselves becoming increasingly interoperable and internetworked. For applications which require specialized VLSI to meet their compression and decompression speed requirements, a common method will provide economies of scale not possible within a single application.

This article gives an overview of JPEG's proposed image-compression standard. Readers without prior knowledge of JPEG or compression based on the Discrete Cosine Transform (DCT) are encouraged to study first the detailed description of the Baseline sequential codec, which is the basis for all of the DCT-based decoders. While this article provides many details, many more are necessarily omitted. The reader should refer to the ISO draft standard [2] before attempting implementation.

Interestingly, some of the earliest industry attention to the JPEG proposal has been focused on the Baseline sequential codec as a *motion* (intraframe) image compression method. (See the associated sidebar, "NEXTstep: Putting JPEG to Multiple Uses.") The fact that it has not been in JPEG's charter as an ISO working group to address this application may indicate that distinction between still- and motion-image coding can sometimes be artificial.

Background: Requirements and Selection Process

JPEG's goal has been to develop a method for continuous-tone image compression which meets the following requirements:

a) be at or near the state of the art with regard to compression rate and accompanying image fidelity, over a wide range of image quality ratings, and especially in the range where visual fidelity to the original is characterized as "very good" to "excellent"; also, the encoder should be parameterizable, so that the application (or user) can set the desired compression/quality tradeoff;

b) be applicable to practically any kind of continuous-tone digital source image (i.e., for most practical purposes not be restricted to images of certain dimensions, color spaces, pixel aspect ratios, etc.), and not be limited to classes of imagery with restrictions on scene content, such as complexity, range of colors, or statistical properties;

c) have tractable computational complexity, to make feasible software implementations with viable performance on a range of CPUs, as well as hardware implementations with viable cost for applications requiring high performance;

d) have the following modes of operation:

- Sequential encoding: each image component is encoded in a single left-to-right, top-to-bottom scan;
- Progressive encoding: the image is encoded in multiple scans for applications in which transmission time is long, and the viewer prefers to watch the image build up in multiple coarse-to-clear passes;
- Lossless encoding: the image is encoded to guarantee exact recovery of every source image sample value (even though the result is low compression compared to the lossy modes);
- Hierarchical encoding: the image is encoded at multiple resolutions, so that lower-resolution versions may be accessed without first having to decompress the image at its full resolution.

In June 1987, JPEG conducted a selection process based on a blind assessment of subjective picture quality, and narrowed 12 proposed methods to three. Three informal working groups formed to refine them, and in January 1988, a second, more rigorous selection process [18] revealed the "ADCT" proposal [10], based on the 8 × 8 DCT, had produced the best picture quality.

At the time of its selection, the DCT-based method was only partially defined for some of the modes of operation. From 1988 through 1990, JPEG undertook the sizable task of defining, documenting, simulating, testing, validating, and simply agreeing on the plethora of details necessary for genuine interoperability and universality. Further history of the JPEG effort is contained in [5, 6, 8, 17].

Architecture of the Proposed Standard

The proposed standard contains the four "modes of operation" identified previously. For each mode, one or more distinct codecs are specified. Codecs within a mode differ according to the precision of source image samples they can handle or the entropy coding method they use. Although the word codec (encoder/decoder) is used frequently in this article, there is no requirement that implementations must include both an encoder and a decoder. Many applications will have systems or devices which require only one or the other.

The four modes of operation and their various codecs have resulted from JPEG's goal of being generic and from the diversity of image formats across applications. The multiple pieces can give the impression of undesirable complexity, but they should actually be regarded as a comprehensive "toolkit" which can span a wide range of continuous-tone image applications. It is unlikely that many implementations will utilize every tool—indeed, most of the early implementations now on the market (even before final ISO approval) have implemented only the Baseline sequential codec.

The Baseline sequential codec is inherently a rich and sophisticated compression method which will be sufficient for many applications. Getting just this minimum JPEG capability implemented properly and interoperably will provide the industry with an important initial capability for exchange of images across vendors and applications.

Processing Steps for DCT-Based Coding

Figures 1 and 2 show the key processing steps which are the heart of the DCT-based modes of operation. These figures illustrate the special case of single-component (grayscale) image compression. The reader can grasp the essentials of DCT-based compression by thinking of it as essentially compression of a stream of 8 × 8 blocks of grayscale image samples. Color image compression can then be approximately regarded as compression of multiple grayscale images, which are either compressed entirely one at a time, or are compressed by alternately interleaving 8 × 8 sample blocks from each in turn.

For DCT sequential-mode codecs, which include the Baseline sequential codec, the simplified diagrams indicate how single-component compression works in a fairly complete way. Each 8 × 8 block is input, makes its way through each processing step, and yields output in compressed form into the data stream. For DCT progressive-mode codecs, an image buffer exists prior to the entropy coding step, so that an image can be stored and then parcelled out in multiple scans with successively improving quality. For the hierarchical mode of operation, the steps shown are used as building blocks within a larger framework.

8 × 8 FDCT and IDCT

At the input to the encoder, source image samples are grouped into 8 × 8 blocks, shifted from unsigned integers with range $[0, 2^P - 1]$ to signed integers with range $[-2^{P-1}, 2^{P-1} - 1]$, and input to the Forward DCT (FDCT). At the output from the decoder, the Inverse DCT (IDCT) outputs 8 × 8 sample blocks to form the reconstructed image. The following equations are the idealized mathematical definitions of the 8 × 8 FDCT and 8 × 8 IDCT:

$$F(u, v) = \frac{1}{4}C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x, y) * \cos\frac{(2x + 1)u\pi}{16}\cos\frac{(2y + 1)v\pi}{16}\right] \quad (1)$$

$$f(x, y) = \frac{1}{4}\left[\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)F(u, v) * \cos\frac{(2x + 1)u\pi}{16}\cos\frac{(2y + 1)v\pi}{16}\right]$$

where: $C(u), C(v) = 1/\sqrt{2}$ for $u, v = 0$; $C(u), C(v) = 1$ otherwise. (2)

The DCT is related to the Discrete Fourier Transform (DFT). Some simple intuition for DCT-based

compression can be obtained by viewing the FDCT as a harmonic analyzer and the IDCT as a harmonic synthesizer. Each 8×8 block of source image samples is effectively a 64-point discrete signal which is a function of the two spatial dimensions x and y. The FDCT takes such a signal as its input and decomposes it into 64 orthogonal basis signals. Each contains one of the 64 unique two-dimensional (2D) "spatial frequencies" which comprise the input signal's "spectrum." The output of the FDCT is the set of 64 basis-signal amplitudes or "DCT coefficients" whose values are uniquely determined by the particular 64-point input signal.

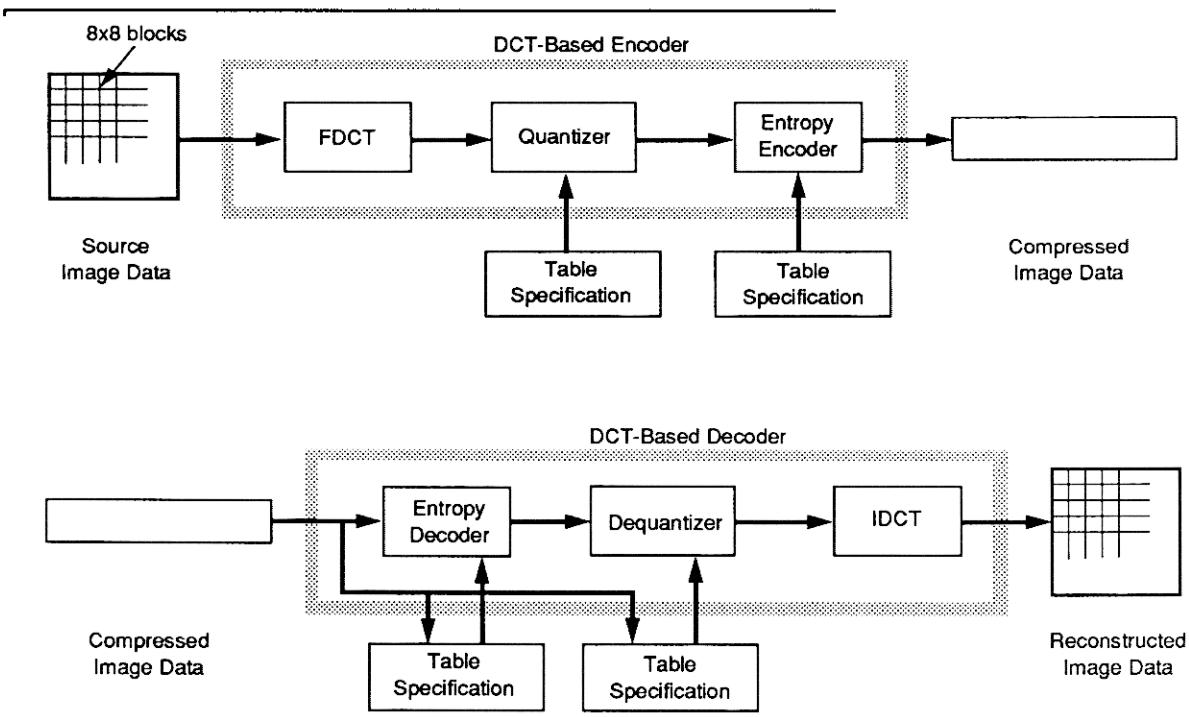
The DCT coefficient values can thus be regarded as the relative amounts of the 2D spatial frequencies contained in the 64-point input signal. The coefficient with zero frequency in both dimensions is called the "DC coefficient" and the remaining 63 coefficients are called the "AC coefficients." Because sample values typically vary slowly from point to point across an image, the FDCT processing step lays the foundation for achieving data compression by concentrating most of the signal in the lower spatial frequencies. For a typical 8×8 sample block from a typical source image, most of the spatial frequencies have zero or near-zero amplitude and need not be encoded.

At the decoder the IDCT reverses this processing step. It takes the 64 DCT coefficients (which at that point have been quantized) and reconstructs a 64-point output image signal by summing the basis signals. Mathematically, the DCT is a one-to-one mapping of 64-point vectors between the image and the frequency domains. If the FDCT and IDCT could be computed with perfect accuracy and if the DCT coefficients were not quantized as in the following description, the original 64-point signal could be exactly recovered. In principle, the DCT introduces no loss to the source image samples; it merely transforms them to a domain in which they can be more efficiently encoded.

Some properties of practical FDCT and IDCT implementations raise the issue of what precisely should be required by the JPEG standard. A fundamental property is that the FDCT and IDCT equations contain transcendental functions. Consequently, no physical implementation can compute them with perfect accuracy. Because of the DCT's application importance and its relationship to the DFT, many different algorithms by which the FDCT and IDCT may be approximately computed have been devised [15]. Indeed, research in fast DCT algorithms is ongoing,

FIGURE 1.
DCT-Based Encoder Processing Steps

FIGURE 2.
DCT-Based Decoder Processing Steps



and no single algorithm is optimal for all implementations. What is optimal in software for a general-purpose CPU is unlikely to be optimal in firmware for a programmable DSP and is certain to be suboptimal for dedicated VLSI.

Even in light of the finite precision of the DCT inputs and outputs, independently designed implementations of the very same FDCT or IDCT algorithm which differ even minutely in the precision by which they represent cosine terms or intermediate results, or in the way they sum and round fractional values, will eventually produce slightly different outputs from identical inputs.

To preserve freedom for innovation and customization within implementations, JPEG has chosen to specify neither a unique FDCT algorithm nor a unique IDCT algorithm in its proposed standard. This makes compliance somewhat more difficult to confirm, because two compliant encoders (or decoders) generally will not produce identical outputs given identical inputs. The JPEG standard will address this issue by specifying an accuracy test as part of its compliance tests for all DCT-based encoders and decoders; this is to ensure against crudely inaccurate cosine basis functions which would degrade image quality.

For each DCT-based mode of operation, the JPEG proposal specifies separate codecs for images with 8-bit and 12-bit (per component) source image samples. The 12-bit codecs, needed to accommodate certain types of medical and other images, require greater computational resources to achieve the required FDCT or IDCT accuracy. Images with other sample precisions can usually be accommodated by either an 8-bit or 12-bit codec, but this must be done outside the JPEG standard. It is the responsibility of applications to decide how to fit or pad a 6-bit sample into the 8-bit encoder's input interface, how to unpack it at the decoder's output, and how to encode any necessary image characteristics, display characteristics and viewing distance. For applications in which these variables can be reasonably well defined, psychovisual experiments can be performed to determine the best thresholds. The experiment described in [11] has led to a set of Quantization Tables for CCIR-601 [4] images and displays. These have been used experimentally by JPEG members and will appear in the ISO standard as a matter of information, but not as a requirement.

Quantization

After output from the FDCT, each of the 64 DCT coefficients is uniformly quantized in conjunction with a 64-element Quantization Table, which must be specified by the application (or user) as an input to the encoder. Each element can be any integer value from 1 to 255, which specifies the step size of the quantizer for its corresponding DCT coefficient. The purpose of quantization is to achieve further compression by representing DCT coefficients with no greater precision than is necessary to achieve the desired image quality. Stated another way, the goal of this processing step is to discard information which is not visually significant. Quantization is a many-to-one mapping, and therefore is fundamentally lossy. It is the principal source of lossiness in DCT-based encoders.

Quantization is defined as division of each DCT coefficient by its corresponding quantizer step size, followed by rounding to the nearest integer:

$$F^Q(u, v) = Integer\ Round\left(\frac{F(u, v)}{Q(u, v)}\right). \quad (3)$$

This output value is normalized by the quantizer step size. Dequantization is the inverse function, which in this case means simply that the normalization is removed by multiplying by the step size, which returns the result to a representation appropriate for input to the IDCT:

$$F^{Q'}(u, v) = F^Q(u, v) * Q(u, v) \quad (4)$$

When the aim is to compress the image as much as possible without visible artifacts, each step size ideally should be chosen as the perceptual threshold or "just noticeable difference" for the visual contribution of its corresponding cosine basis function. These thresholds are also functions of the source

DC Coding and Zig-Zag Sequence

After quantization, the DC coefficient is treated separately from the 63 AC coefficients. The DC coefficient is a measure of the average value of the 64 image samples. Because there is usually strong correlation between the DC coefficients of adjacent 8 × 8 blocks, the quantized DC coefficient is encoded as the difference from the DC term of the previous block in the encoding order (defined in the following), as shown in Figure 3. This special treatment is worthwhile, as DC coefficients frequently contain a significant fraction of the total image energy.

Finally, all of the quantized coefficients are ordered into the "zig-zag" sequence, also shown in Figure 3. This ordering helps to facilitate entropy coding by placing low-frequency coefficients (which are more likely to be nonzero) before high-frequency coefficients.

Entropy Coding

The final DCT-based encoder processing step is entropy coding. This step achieves additional compression losslessly by encoding the quantized DCT coefficients more compactly based on their statistical characteristics. The JPEG proposal specifies two entropy coding methods—Huffman coding [7] and arithmetic coding [14]. The Baseline sequential codec uses Huffman coding, but codecs with both methods are specified for all modes of operation.

It is useful to consider entropy

coding as a 2-step process. The first step converts the zig-zag sequence of quantized coefficients into an intermediate sequence of symbols. The second step converts the symbols to a data stream in which the symbols no longer have externally identifiable boundaries. The form and definition of the intermediate symbols is dependent on both the DCT-based mode of operation and the entropy coding method.

Huffman coding requires that one or more sets of Huffman code tables be specified by the application. The same tables used to compress an image are needed to decompress it. Huffman tables may be predefined and used within an application as defaults, or computed specifically for a given image in an initial statistics-gathering pass prior to compression. Such choices are the business of the applications which use JPEG; the JPEG proposal specifies no required Huffman tables. Huffman coding for the Baseline sequential encoder is described in detail later in this article.

By contrast, the particular arithmetic coding method specified in the JPEG proposal [2] requires no tables to be externally input, because it is able to adapt to the image statistics as it encodes the image. (If desired, statistical conditioning tables can be used as inputs for slightly better efficiency, but this is not required.) Arithmetic coding has produced 5–10% better compression than Huffman for many of the images which JPEG members have tested. However, some feel it is more complex than Huffman coding for certain implementations, for example, the highest-speed hardware implementations. (Throughout JPEG's history, "complexity" has proved to be most elusive as a practical metric for comparing compression methods.)

If the only difference between two JPEG codecs is the entropy coding method, transcoding between the two is possible by simply entropy decoding with one method and entropy recoding with the other.

Compression and Picture Quality

For color images with moderately complex scenes, all DCT-based modes of operation typically produce the following levels of picture quality for the indicated ranges of compression. These levels are only a guideline—quality and compression can vary significantly according to source image characteristics and scene content. (The units "bits/pixel" here mean the total number of bits in the compressed image—including the chrominance components—divided by the number of samples in the luminance component.)

- 0.25–0.5 bits/pixel: moderate to good quality, sufficient for some applications;
- 0.5–0.75 bits/pixel: good to very good quality, sufficient for many applications;
- 0.75–1.5 bits/pixel: excellent quality, sufficient for most applications;
- 1.5–2.0 bits/pixel: usually indistinguishable from the original, sufficient for the most demanding applications.

Later in this article, Figure 11 shows an example of the picture quality obtained for a CCIR-601 image at various stages and bit rates of a progressive encoding. Because FDCT and Quantization are common to progressive and sequential DCT-based modes, the quality and compression shown in Figure 11 is also indicative of the trade-offs that can be expected for sequential coding.

Processing Steps for Predictive Lossless Coding

After its selection of a DCT-based method in 1988, JPEG discovered that a DCT-based lossless mode was difficult to define as a practical standard against which encoders and decoders could be independently implemented, without placing severe constraints on both encoder and decoder implementations.

JPEG, to meet its requirement for a lossless mode of operation, has chosen a simple predictive method which is wholly independent of the DCT processing described previously. Although not the result of rigorous competitive evaluation as was the DCT-based method, the predictive method produces results which, in light of its simplicity, are surprisingly close to the state of the art for lossless continuous-tone compression.

Figure 4 shows the main processing steps for a single-component image. A predictor combines the

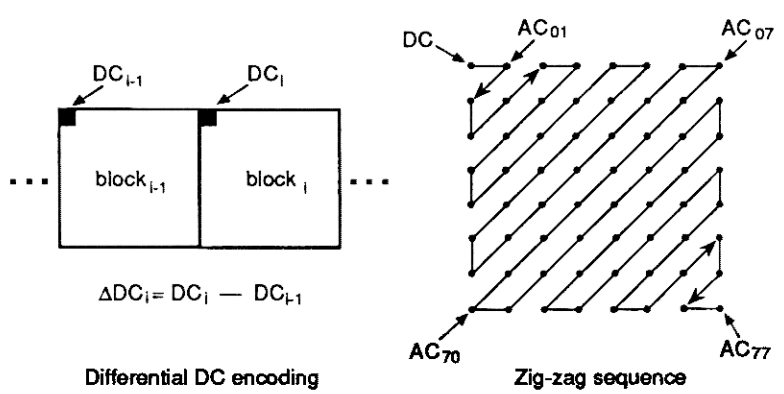


FIGURE 3.
Preparation of Quantized Coefficients for Entropy Coding

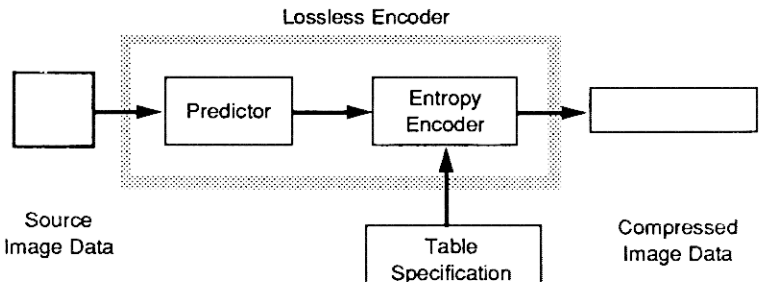


FIGURE 4.
Lossless Mode Encoder Processing Steps

TABLE 1.

Predictors for Lossless Coding

SELECTION-VALUE	PREDICTION
0	no prediction
1	A
2	B
3	C
4	A + B − C
5	A + ((B − C)/2)
6	B + ((A − C)/2)
7	(A + B)/2

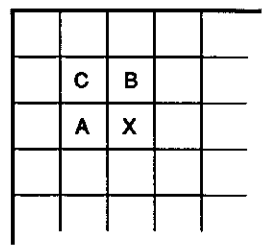


FIGURE 5.
3-Sample Prediction Neighborhood

values of up to three neighboring samples (A, B, and C) to form a prediction of the sample indicated by X in Figure 5. This prediction is then subtracted from the actual value of sample X, and the difference is encoded losslessly by either of the entropy coding methods—Huffman or arithmetic. Any one of the eight predictors listed in Table 1 (under "selection-value") can be used.

Selections 1, 2 and 3 are one-dimensional predictors and selections 4, 5, 6, and 7 are two-dimensional predictors. Selection-value 0 can only be used for differential coding in the hierarchical mode of operation. The entropy coding is nearly identical to that used for the DC coefficient as described later (for Huffman coding).

For the lossless mode of operation, two different codecs are specified—one for each entropy coding method. The encoders can use any source image precision from 2 to 16 bits/sample, and can use any of the predictors except selection-value 0. The decoders must handle any of the sample precisions and any of the predictors.

Lossless codecs typically produce around 2:1 compression for color images with moderately complex scenes.

Multiple-Component Images

The previous sections discussed the key processing steps of the DCT-based and predictive lossless codecs for the case of single-component source images. These steps accomplish the image data compression. But a good deal of the JPEG proposal is also concerned with the handling and control of color (or other) images with multiple components. JPEG's aim for a generic compression standard requires its proposal to accommodate a variety of source image formats.

Source Image Formats

The source image model used in the JPEG proposal is an abstraction from a variety of image types and applications, and consists only of what is necessary to compress and reconstruct digital image data. The reader should recognize that the JPEG compressed data format does not encode enough information to serve as a complete image representation. For example, JPEG does not specify or encode any information on pixel aspect ratio, color space, or image acquisition characteristics.

Figure 6 illustrates the JPEG source image model. A source image contains from 1 to 255 image components, sometimes called color or spectral bands or channels. Each component consists of a rectangular array of samples. A sample is defined to be an unsigned integer with precision P bits, with any value in the range $[0, 2^P - 1]$. All samples of all components within the same source image must have the same precision P. P can be 8 or 12 for DCT-based codecs, and 2 to 16 for predictive codecs.

The ith component has sample dimensions x_i by y_i. To accommodate formats in which some image components are sampled at different rates than others, components can have different dimensions. The dimensions must have a mutual integral relationship defined by H_i and V_i, the relative horizontal and vertical sampling factors, which must be specified for each component. Overall image dimensions X and Y are defined as the maximum x_i and y_i for all components in the image, and can be any number up to 2^{16}. H and V are allowed only the integer values 1 through 4. The encoded parameters are X, Y, and the H_is and V_is for each component. The decoder reconstructs the dimensions x_i and y_i for each component, according to the following relationship shown in Equation 5:

$$x_i = \left\lceil X \times \frac{H_i}{H_{max}} \right\rceil \text{ and } y_i = \left\lceil Y \times \frac{V_i}{V_{max}} \right\rceil \quad (5)$$

where $\lceil\ \rceil$ is the ceiling function.

Encoding Order and Interleaving

A practical image compression standard must address how systems will need to handle the data during the process of decompression. Many applications need to pipeline the process of displaying or printing multiple-component images in parallel with the process of decompression. For many systems, this is only feasible if the components are interleaved together within the compressed data stream.

To make the same interleaving machinery applicable to both DCT-based and predictive codecs, the JPEG proposal has defined the concept of "data unit." A data unit is a sample in predictive codecs and an 8×8 block of samples in DCT-based codecs.

The order in which compressed data units are placed in the compressed data stream is a generalization of raster-scan order. Generally, data units are ordered from left-to-right and top-to-bottom according to the orientation shown in Figure 6. (It is the responsibility of applications to define which edges of a source image are top, bottom, left, and right.) If an image component is noninterleaved (i.e., compressed without being interleaved with other components), compressed data units are ordered in a pure raster scan as shown in Figure 7.

When two or more components are interleaved, each component C_i is partitioned into rectangular regions of H_i by V_i data units, as shown in the generalized example of Figure 8. Regions are ordered within a component from left-to-right and top-to-bottom, and within a region, data units are ordered from left-to-right and top-to-bottom. The JPEG proposal defines the term Minimum Coded Unit (MCU) to be the smallest group of interleaved data units. For the example shown, MCU_1 consists of data units taken first from the top-left-most region of C_1, followed by data units from the same region of C_2, and likewise for C_3 and C_4. MCU_2 continues the pattern as shown.

Thus, interleaved data is an ordered sequence of MCUs, and the number of data units contained in an MCU is determined by the number of components interleaved and their relative sampling factors. The maximum number of components which can be interleaved is 4 and the maximum number of data units in an MCU is 10. The latter restriction is expressed as shown in Equation 6, where the summation is over the interleaved components:

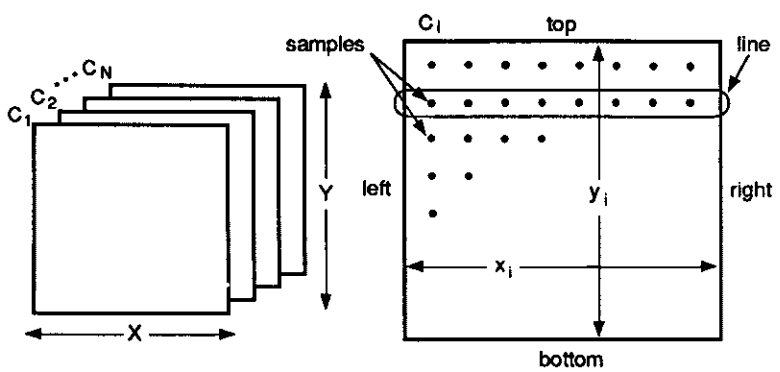
$$\sum_{\substack{all\ i\ in \\ interleave}} H_i \times V_i \leq 10 \qquad (6)$$

Because of this restriction, not every combination of 4 components which can be represented in non-interleaved order within a JPEG-compressed image is allowed to be interleaved. Also, note that the JPEG proposal allows some components to be interleaved and some to be noninterleaved within the same compressed image.

Multiple Tables

In addition to the interleaving control discussed previously, JPEG codecs must control application of the proper table data to the proper components. The same quantization table and the same entropy coding table (or set of tables) must be used to encode all samples within a component.

JPEG decoders can store up to 4 different quantization tables and up to 4 different (sets of) entropy coding tables simultaneously. (The Baseline sequential decoder is the exception; it can only store up to 2 sets of entropy coding tables.) This is necessary for switching between different tables during decompression of a scan containing multiple (interleaved) components, in order to apply the proper table to the proper component. (Tables cannot be loaded during decompression of a scan.) Figure 9 illustrates the table-switching control that must be managed in conjunc-



(a) Source image with multiple components

(b) Characteristics of an image component

FIGURE 6.
JPEG Source Image Model

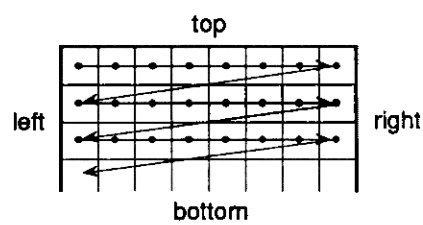


FIGURE 7.
Noninterleaved Data Ordering

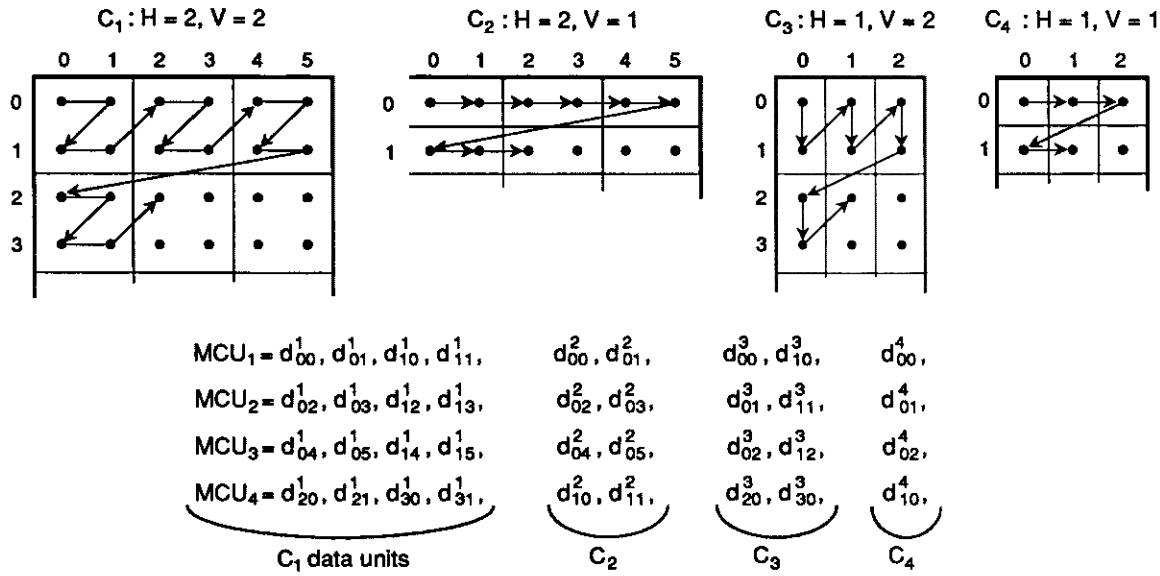


FIGURE 8.
Generalized Interleaved Data Ordering Example

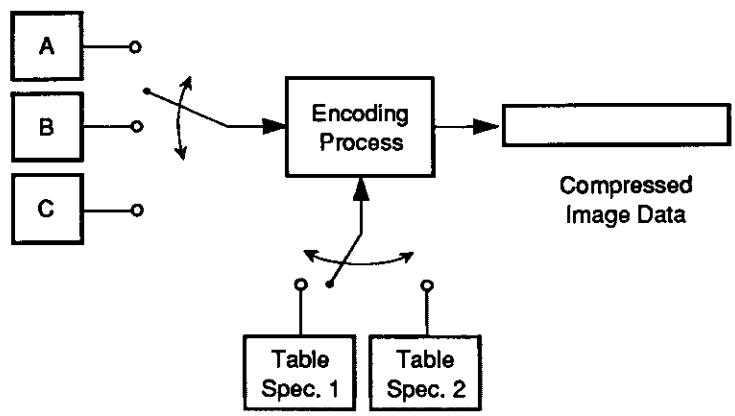


FIGURE 9.
Component-Interleave and Table-Switching Control

tion with multiple-component interleaving for the encoder side. (This simplified view does not distinguish between quantization and entropy coding tables.)

Baseline and Other DCT Sequential Codecs

The DCT sequential mode of operation consists of the FDCT and Quantization steps from the section entitled "Processing Steps for DCT-Based Coding" and the multiple-component control from the previous section on multiple component images. In addition to the Baseline sequential codec, other DCT sequential codecs are defined to accommodate the two different sample precisions (8 and 12 bits) and the two different types of entropy coding methods (Huffman and arithmetic).

Baseline sequential coding is for images with 8-bit samples and uses Huffman coding only. It also differs from the other sequential DCT codecs in that its decoder can store only two sets of Huffman tables (one AC table and one DC table per set). This restriction means that, for images with three or four interleaved components, at least one set of Huffman tables must be shared by two components. This restriction poses no limitation at all for noninterleaved components; a new set of tables can be loaded into the decoder before decompression of a noninterleaved component begins.

For many applications which do need to interleave three color components, this restriction is hardly a limitation at all. Color spaces (YUV, CIELUV, CIELAB, and others) which represent the chromatic ("color") information in two components and the achromatic ("grayscale") information in a third are more efficient for compression than spaces like RGB. One Huffman table set can be used for the achromatic component and one for the chrominance components. DCT coefficient statistics are similar for the chrominance components of most images, and one set of Huffman tables can encode both almost as optimally as two.

The committee also felt that early availability of single-chip im-

plementations at commodity prices would encourage early acceptance of the JPEG proposal in a variety of applications. In 1988 when Baseline sequential was defined, the committee's VLSI experts felt that current technology made the feasibility of crowding four sets of loadable Huffman tables—in addition to four sets of Quantization tables—onto a single commodity-priced codec chip a risky proposition.

The FDCT, Quantization, DC differencing, and zig-zag ordering processing steps for the Baseline sequential codec proceed just as described in the section "Processing Steps for DCT-Based Coding." Prior to entropy coding, there usually are few nonzero and many zero-valued AC coefficients. The task of entropy coding is to encode these few coefficients efficiently. The description of Baseline sequential entropy coding is given in two steps: conversion of the quantized DCT coefficients into an intermediate sequence of symbols and assignment of variable-length codes to the symbols.

Intermediate Entropy Coding Representations

In the intermediate symbol sequence, each nonzero AC coefficient is represented in combination with the "runlength" (consecutive number) of zero-valued AC coefficients which precede it in the zig-zag sequence. Each such runlength/nonzero-coefficient combination is (usually) represented by a pair of symbols:

symbol-1
(RUNLENGTH, SIZE)

symbol-2
(AMPLITUDE)

Symbol-1 represents two pieces of information, RUNLENGTH and SIZE. Symbol-2 represents the single piece of information designated AMPLITUDE, which is simply the amplitude of the nonzero AC coefficient. RUNLENGTH is the number of consecutive zero-valued AC coefficients in the zig-zag sequence preceding the nonzero AC coefficient being represented. SIZE is the number of bits used to encode AMPLITUDE—that is, to encode symbol-2, by the signed-integer encoding used with JPEG's particular method of Huffman coding.

RUNLENGTH represents zero-runs of length 0 to 15. Actual zero-runs in the zig-zag sequence can be greater than 15, so the symbol-1 value (15, 0) is interpreted as the extension symbol with runlength = 16. There can be up to three consecutive (15, 0) extensions before the terminating symbol-1 whose RUNLENGTH value completes the actual runlength. The terminating symbol-1 is always followed by a single symbol-2, except for the case in which the last run of zeros includes the last (63d) AC coefficient. In this frequent case, the special symbol-1 value (0, 0) means EOB (end of block), and can be viewed as an "escape" symbol which terminates the 8×8 sample block.

Thus, for each 8×8 block of samples, the zig-zag sequence of 63 quantized AC coefficients is represented as a sequence of symbol-1, symbol-2 symbol-pairs, though each "pair" can have repetitions of symbol-1 in the case of a long runlength or only one symbol-1 in the case of an EOB.

The possible range of quantized AC coefficients determines the range of values which both the AMPLITUDE and the SIZE information must represent. A numerical analysis of the 8×8 FDCT equation shows that, if the 64-point (8×8 block) input signal contains N-bit integers, then the nonfractional part of the output numbers (DCT coefficients) can grow by at most 3 bits. This is also the largest possible size of a quantized DCT coefficient when its quantizer step size has integer value 1.

Baseline sequential has 8-bit integer source samples in the range $[2^7, 2^7 - 1]$, so quantized AC coefficient amplitudes are covered by integers in the range $[-2^{10}, 2^{10} - 1]$. The signed-integer encoding uses symbol-2 AMPLITUDE codes of 1 to 10 bits in length (so SIZE also represents values from 1 to 10), and RUNLENGTH represents values from 0 to 15 as discussed previously. For AC coefficients, the structure of the symbol-1 and symbol-2 intermediate representations is illustrated in Tables 2 and 3, respectively.

The intermediate representation for an 8×8 sample block's differential DC coefficient is structured similarly. Symbol-1, however, represents only SIZE information; symbol-2 represents AMPLITUDE information as before:

symbol-1 (SIZE) symbol-2 (AMPLITUDE)

Because the DC coefficient is differentially encoded, it is covered by twice as many integer values, $[-2^{11}, 2^{11} - 1]$ as the AC coefficients, so

TABLE 2.

Baseline Huffman Coding Symbol-1 Structure

		SIZE					
		0	1	2	. . .	9	10
RUNLENGTH	0	EOB					
	.	X					
	.	X			RUN-SIZE VALUES		
	.	X					
	15	ZRL					

one additional level must be added to the bottom of Table 3 for DC coefficients. Symbol-1 for DC coefficients thus represents a value from 1 to 11.

Variable-Length Entropy Coding

Once the quantized coefficient data for an 8 × 8 block is represented in the intermediate symbol sequence described above, variable-length codes are assigned. For each 8 × 8 block, the DC coefficient's symbol-1 and symbol-2 representation is coded and output first.

For both DC and AC coefficients, each symbol-1 is encoded with a variable-length code (VLC) from the Huffman table set assigned to the 8 × 8 block's image component. Each symbol-2 is encoded with a "variable-length integer" (VLI) code whose length in bits is given in Table 3. VLCs and VLIs both are codes with variable lengths, but VLIs are not Huffman codes. An important distinction is that the length of a VLC (Huffman code) is not known until it is decoded, but the length of a VLI is stored in its preceding VLC.

Huffman codes (VLCs) must be specified externally as an input to JPEG encoders. (Note that the form in which Huffman tables are represented in the data stream is an indirect specification with which the decoder must construct the tables themselves prior to decompression.) The JPEG proposal includes an example set of Huffman tables in its informational annex, but because they are application-specific, it specifies none for required use. The VLI codes, in contrast, are "hardwired" into the proposal. This is appropriate, because the VLI codes are far more numerous, can be computed rather than stored, and have not been shown to be appreciably more efficient when implemented as Huffman codes.

Other DCT Sequential Codecs

The structure of the 12-bit DCT sequential codec with Huffman coding is a straightforward extension of the entropy coding method described previously. Quantized DCT coefficients can be 4 bits larger, so the SIZE and AMPLITUDE information extend accordingly. DCT sequential with arithmetic coding is described in detail in [2].

DCT Progressive Mode

The DCT progressive mode of operation consists of the same FDCT and Quantization steps from the section "Processing Steps for DCT-Based Coding" that are used by DCT sequential mode. The key difference is that each image component is encoded in multiple scans rather than in a single scan. The first scan(s) encode a rough but recognizable version of the image which can be transmitted quickly in comparison to the total transmission time, and are refined by succeeding scans until reaching the level of picture quality that was established by the quantization tables.

TA **2.**

Baseline Entropy Coding Symbol-2 Structure

SIZE	AMPLITUDE
	−1,1
2	−3,−2,2,3
3	−7..−4,4..7
4	−15..−8,8..15
5	−31..−16,16..31
6	−63..−32,32..63
7	−127..−64,64..127
8	−255..−128,128..255
9	−511..−256,256..511
10	−1023..−512,512..1023

To achieve this requires the addition of an image-sized buffer memory at the output of the quantizer, before the input to the entropy encoder. The buffer memory must be of sufficient size to store the image as quantized DCT coefficients, each of which (if stored straightforwardly) is 3 bits larger than the source image samples. After each block of DCT coefficients is quantized, it is stored in the coefficient buffer memory. The buffered coefficients are then partially encoded in each of multiple scans.

There are two complementary methods by which a block of quantized DCT coefficients may be partially encoded. First, only a specified "band" of coefficients from the zig-zag sequence need be encoded within a given scan. This procedure is called "spectral selection," because each band typically contains coefficients which occupy a lower or higher part of the spatial-frequency spectrum for that 8 × 8 block. Secondly, the coefficients within the current band need not be encoded to their full (quantized) accuracy in a given scan. Upon a coefficient's first encoding, the N most significant bits can be encoded first, where N is specifiable. In subsequent scans, the less significant bits can then be encoded. This procedure is called successive approximation. Both procedures can be used separately, or mixed in flexible combinations.

Some intuition for spectral selection and successive approximation can be obtained from Figure 10. The quantized DCT coefficient information can be viewed as a rectangle for which the axes are the DCT coefficients and their amplitudes. Spectral selection slices the information in one dimension and successive approximation in the other.

For comparative purposes, Figure 11 shows an example of both progressive encoding methods.

Hierarchical Mode Of Operation

The hierarchical mode provides a "pyramidal" encoding of an image at multiple resolutions, each differing in resolution from its adjacent encoding by a factor of two in either the horizontal or vertical dimension or both. The encoding procedure can be summarized as follows:

a) Filter and down-sample the original image by the desired number of multiples of 2 in each dimension.

b) Encode this reduced-size image using one of the sequential DCT, progressive DCT, or lossless encoders described previously.

c) Decode this reduced-size image and then interpolate and up-sample it by 2 horizontally and/or vertically, using the identical interpolation filter which the receiver must use.

d) Use this up-sampled image as a prediction of the original at this resolution, and encode the difference image using one of the sequential DCT, progressive DCT, or lossless encoders described previously.

e) Repeat steps c) and d) until the full resolution of the image has been encoded.

The encoding in steps b) and d) may be done using only DCT-based processes, only lossless processes, or DCT-based processes with a final lossless process for each component.

Hierarchical encoding is useful in applications in which a very high resolution image must be accessed by a lower-resolution device, which does not have the buffer capacity to reconstruct the image at its full resolution and then scale it down for the lower-resolution display. An example is an image scanned and compressed at high resolution for a very high-quality printer, where the image must also be displayed on a low-resolution PC video screen.

Other Aspects of the JPEG Proposal

Some key aspects of the proposed standard can only be mentioned briefly. Foremost among these are points concerning the coded representation for compressed image data specified in addition to the encoding and decoding procedures.

Most importantly, an *interchange format* syntax is specified which ensures that a JPEG-compressed image can be exchanged successfully between different application environments. The format is structured in a consistent way for all modes of operation. The interchange format always includes all quantization and entropy-coding

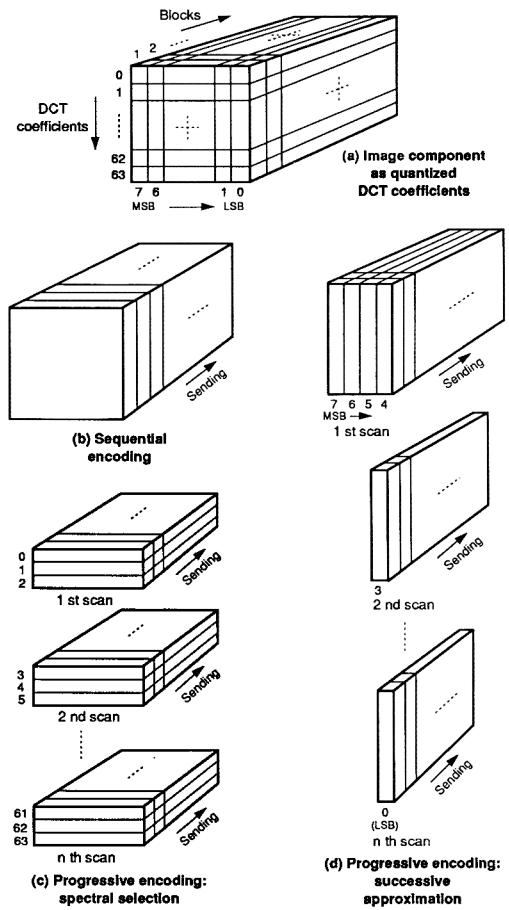


FIGURE 10.
Spectral Selection and Successive Approximation Methods of Progressive Encoding


a
b1
b2
b3
b4
c1
c2
c3
c4

tables which were used to compress the image.

Applications (and application-specific standards) are the "users" of the JPEG standard. The JPEG standard imposes no requirement that, within an application's environment, all or even any tables must be encoded with the compressed image data during storage or transmission. This leaves applications the freedom to specify default or referenced tables if they are considered appropriate. It also leaves them the responsibility to ensure that JPEG-compliant decoders used within their environment get loaded with the proper tables at the proper times, and that the proper tables are included in the interchange format when a compressed image is "exported" outside the application.

Some of the important applications that are already in the process of adopting JPEG compression or have stated their interest in doing so are Adobe's PostScript language for printing systems [1], the Raster Content portion of the ISO Office Document Architecture and Interchange Format [12], the future CCITT color facsimile standard, and the European ETSI videotex standard [9].

Standardization Schedule

JPEG's ISO standard will be divided into two parts. Part 1 [2] will specify the four modes of operation, the different codecs specified for those modes, and the interchange format. It will also contain a substantial informational section on implementation guidelines. Part 2 [3] will specify the compliance tests which will determine whether an implementation of an encoder or decoder specified in Part 1 conforms to the standard.

There are two key balloting phases in the ISO standardization process: a Committee Draft (CD) is balloted to determine promotion to Draft International Standard (DIS), and a DIS is balloted to determine promotion to International Standard (IS). Each ballot requires four to six months. JPEG's Part 1 began CD ballot in February 1991, and Part 2 is expected to begin CD ballot by June 1991.

Though there is no guarantee that the first ballot of each phase will result in promotion to the next, JPEG's CD Part 1 contains no technical changes (other than some minor corrections) from JPEG's latest Technical Specification [13]. Successive revisions of the Technical Specification were widely distributed and subjected to informal review in many forums throughout 1990, and yet the technical content has been stable for nearly a year.

Conclusions

The emerging JPEG continuous-tone image compression standard is not a panacea that will solve the myriad issues which must be addressed before digital images will be fully integrated within all the applications that will ultimately benefit from them. For example, if two applications cannot exchange uncompressed images because they use incompatible color spaces, aspect ratios, dimensions, etc., then a common compression method will not help.

However, a great many applications are "stuck" because of storage or transmission costs, because of argument over which (nonstandard) compression method to use, or because VLSI codecs are too expensive due to low volumes. For these applications, the thorough technical evaluation, testing, selection, validation, and documentation work which JPEG committee members have performed is expected to soon yield an approved international standard that will withstand the tests of quality and time. As diverse imaging applications become increasingly implemented on open, networked computing systems, the ultimate measure of the committee's success will be when JPEG-compressed digital images come to be regarded and even taken for granted as "just another data type," as text and graphics are today.

For more information

Regarding the proposed standard itself, instructions on how to obtain the ISO Committee Draft Part 1, the JPEG Technical Specification, which preceded it, and other key documents as they become available can be obtained by writing the author at the following address:

Digital Equipment Corporation
146 Main Street, MLO5-2/G1
Maynard, MA 01754-2571

Floppy disks containing uncompressed, compressed, and reconstructed data for the purpose of informally validating whether an encoder or decoder implementation conforms to the proposed standard are available. Thanks to the following JPEG committee member and his company who have agreed to provide these for a nominal fee on behalf of the committee until arrangements can be made for ISO

FIGURE 11.
Progressive Build-up, showing Spectral Selection vs. Successive Approximation
a) Original Image (CCIR-601 Format: YUV, 720 × 576 Y samples)
b) Spectral Selection
b1. DC coefficients only: 0.19 bits/pixel
b2. Addition of 1 AC coefficient: 0.32 bits/pixel
b3. Addition of 2d. AC coefficient: 0.43 bits/pixel
b4. Addition of 3d.-9th AC coefficients: 0.96 bits/pixel
c) Successive Approximation
c1. 7 MSBs of DC coefficient: 0.15 bits/pixel
c2. Addition of 5 MSBs of AC coefficients: 0.3 bits/pixel
c3. Addition of 6th MSB of AC coefficients: 0.49 bits/pixel
c4. Addition of 7th MSB of AC coefficients: 0.8 bits/pixel

to provide them:

Eric Hamilton
C-Cube Microsystems
399-A W. Trimble Road
San Jose, CA 95131

Acknowledgments

The following longtime JPEG core members have spent untold hours (usually in addition to their "real jobs") to make this collaborative international effort succeed. Each has made specific substantive contributions to the JPEG proposal: Aharon Gill (Zoran, Israel), Eric Hamilton (C-Cube, USA), Alain Leger (CCETT, France), Adriaan Ligtenberg (Storm, USA), Herbert Lohscheller (ANT, Germany), Joan Mitchell (IBM, USA), Michael Nier (Kodak, USA), Takao Omachi (NEC, Japan), William Pennebaker (IBM, USA), Henning Poulsen (KTAS, Denmark), and Jorgen Vaaben (AutoGraph, Denmark). The leadership efforts of Hiroshi Yasuda (NTT, Japan), the Convenor of JTC1/SC2/WG8 from which JPEG was spawned, Istvan Sebestyen (Siemens, Germany), the Special Rapporteur from CCITT SGVIII, and Graham Hudson (British Telecom, U.K.), former JPEG chair and founder of the effort which became JPEG. The author regrets that space does not permit recognition of the many other individuals who contributed to JPEG's work.

William Pennebaker and colleagues at IBM Research provided the processed JPEG test images in Figure 11.

The author's role within JPEG has been supported in a great number of ways by Digital Equipment Corporation. ■

References

1. Adobe Systems Inc. *PostScript Language Reference Manual.* Second Ed. Addison Wesley, Menlo Park, Calif. 1990.
2. Digital Compression and Coding of Continuous-Tone Still Images, Part 1, Requirements and Guidelines. ISO/IEC JTC1 Committee Draft 10918-1, Feb. 1991.
3. Digital Compression and Coding of Continuous-Tone Still Images, Part 2, Compliance Testing. ISO/IEC JTC1 Committee Draft 10918-2. To be published Summer 1991.
4. Encoding parameters of digital television for studios. CCIR Recommendations, Recommendation 601, 1982.
5. Hudson, G.P. The development of photographic videotex in the UK. In *Proceedings of the IEEE Global Telecommunications Conference, IEEE Communications Society,* 1984, pp. 319–322.
6. Hudson, G.P., Yasuda, H., and Sebestyen, I. The international standardization of a still picture compression technique. In *Proceedings of the IEEE Global Telecommunications Conference, IEEE Communications Society* (Nov. 1988), pp. 1016–1021.
7. Huffman, D.A. A method for the construction of minimum redundancy codes. In *Proceedings IRE,* vol. 40, 1962, pp. 1098–1101.
8. Leger, A. Implementations of fast discrete cosine transform for full color videotex services and terminals. In *Proceedings of the IEEE Global Telecommunications Conference, IEEE Communications Society* (1984), pp. 333–337.
9. Leger, A., Omachi, T., and Wallace, G. The JPEG still picture compression algorithm and its applications. *Optical Eng.* To be published.
10. Leger, A., Mitchell, M., and Yamazaki, Y. Still picture compression algorithms evaluated for international standardization. In *Proceedings of the IEEE Global Telecommunications Conference, IEEE Communications Society* (Nov. 1988), pp. 1028–1032.
11. Lohscheller, H. A subjectively adapted image communication system. *IEEE Trans. Commun. COM-32* (Dec. 1984), 1316–1322.
12. Office Document Architecture (ODA) and Interchange Format, Part 7: Raster Graphics Content Architectures. ISO/IEC JTC1 International Standard 8613-7.
13. Pennebaker, W.B., JPEG Tech. Specification, Revision 8. Informal working paper JPEG-8-R8, Aug. 1990.
14. Pennebaker, W.B., Mitchell, J.L., et. al. Arithmetic coding articles. *"IBM J. Res. Dev. 32,* 6 Special Issue (Nov. 1988), 717–774.
15. Rao, K.R., and Yip, P. *Discrete Cosine Transform—Algorithms, Advantages, Applications.* Academic Press, Inc, London, 1990.
16. Standardization of Group 3 facsimile apparatus for document transmission. CCITT Recommendations, Fascicle VII.2, Recommendation T.4, 1980.
17. Wallace, G.K. Overview of the JPEG (ISO/CCITT) still image compression standard. Image Processing Algorithms and Techniques. In *Proceedings of the SPIE,* vol. 1244 (Feb. 1990), pp. 220–233.
18. Wallace, G., Vivian, R., and Poulsen, H. Subjective testing results for still picture compression algorithms for international standardization. In *Proceedings of the IEEE Global Telecommunications Conference, IEEE Communications Society* (Nov. 1988), 1022–1027.

CR Categories and Subject Descriptors: I.4.2 [**Image Processing**]: Compression (coding)—*Approximate methods, exact coding*

General Terms: Design, Standardization

Additional Key Words and Phrases: JPEG

About the Author:

GREGORY K. WALLACE is Manager of Workstations Multimedia Advanced Product Development and is Convenor of the ISO/IEC JTC1/SC2/WG10 (JPEG) committee at Digital Equipment Corporation. His research interests include compression, signal processing, and integration of real-time video and audio into workstations. **Author's Present Address:** Digital Equipment Corporation, 146 Main Street, ML05-2/G1, Maynard MA 01754-2571. email: wallace@gauss.enet.dec.com