CHAPTER 17

IMAGE COMPRESSION AND CODING

WHAT WILL WE LEARN?

- What is the meaning of compressing an image?
- Which types of redundancy can be exploited when compressing images and how is this usually done?
- What are the main (lossy and lossless) image compression techniques in use today?
- How can we evaluate the (subjective) quality of an image that has undergone some type of processing (e.g., lossy compression) in an objective way?
- What are the most popular contemporary image compression standards?

17.1 INTRODUCTION

Images can be represented in digital format in many different ways, the simplest of which consists in simply storing all the pixel values in a file following a certain convention (e.g., row first, starting from the top left corner). The result of using such raw image representations is the need for large amounts of storage space (and proportionally long transmission times in the case of file uploads/downloads). The need to save storage space and shorten transmission time and the human visual system

Practical Image and Video Processing Using MATLAB®. By Oge Marques.

tolerance to a modest amount of loss have been the driving factors behind image compression techniques.

Image compression techniques are behind the most popular image and video applications and devices in use today, such as DVD players, digital cameras, and web-based video streaming. Contemporary image compression algorithms are capable of encoding images with high compression efficiency and minimal noticeable degradation of visual quality.

Compression methods can be *lossy*—when a tolerable degree of deterioration in the visual quality of the resulting image is acceptable—or *lossless*—when the image is encoded in its full quality and the original image can be fully recovered at the decoding stage. The overall results of the compression process in terms of both storage savings—usually expressed with figures of merit such as compression ratio (CR) or bits per pixel (bpp)—and resulting quality loss (for the case of lossy techniques) may vary depending on the technique, format, options (such as the "quality" setting for JPEG), and image contents. As a general guideline, lossy compression should be used for general-purpose photographic images, whereas lossless compression should be preferred when dealing with line art, technical drawings, or images in which no loss of detail may be tolerable (most notably, space images and medical images).

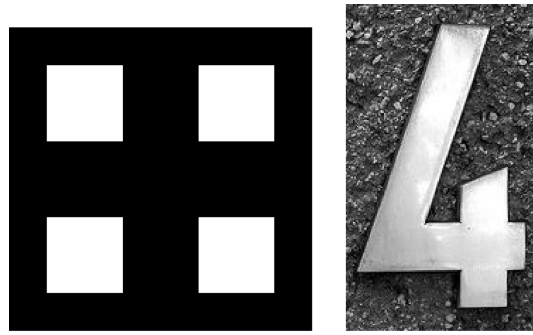
In the remainder of this chapter, we provide an overview of image compression and coding. This is a vast and ever-growing field. Keeping up with the main goals of this book, we emphasize key concepts, provide a brief overview of the most representative contemporary image compression techniques and standards, and show how to evaluate image quality when lossy image compression is used. Readers interested in a more detailed analysis of specific techniques and standards will find many useful pointers to further their studies in the "Learn More About It" section at the end of the chapter.

17.2 BASIC CONCEPTS

The general problem of image compression is to reduce the amount of *data* (i.e., bits) required to represent a digital image. The underlying basis of the reduction process is the removal of redundant data. Mathematically, visual data compression typically involves transforming (encoding) a 2D pixel array into a statistically uncorrelated data set. This transformation is applied prior to storage or transmission. At some later time, the compressed image is decompressed to reconstruct the original image information (if lossless techniques are used) or an approximation of it (when lossy techniques are employed).

17.2.1 Redundancy

Image compression is a particular case of *data compression*: the process of reducing the amount of *data* required to represent a given quantity of *information*. If the same information can be represented using different amounts of data, it is reasonable to believe that the representation that requires more data contains what is technically called *data redundancy*.



(a) (b)

FIGURE 17.1 Two ways to represent the same information using different amounts of data. See text for details.

■ EXAMPLE 17.1

To understand the difference between *data* and *information*, these are five different ways of representing the same information (the quantity *four*) using different representations (in increasing order of size in bits):

- The binary equivalent of the number 4 (`100`): 3 bits
- The Roman numeral `IV` encoded using the ASCII characters for `I` and `V`: 16 bits
- The word "quatro," meaning "four" in Portuguese, encoded as a string of ASCII characters: 48 bits
- An uncompressed 64×64 binary image with four white squares against a black background (Figure 17.1a): 32,768 bits
- A compressed 143×231 grayscale image (Figure 17.1b): 21,7632 bits

The mathematical definition of data redundancy is given in equation (17.1). It is assumed that b_1 and b_2 represent the number of information carrying units (or simply *bits*) in two data sets that represent the same information. In this case, the relative redundancy (R) of the first data set (represented by b_1) can be defined as

$$R = 1 - \frac{1}{\text{CR}} \tag{17.1}$$

where the parameter CR, commonly known as *compression ratio*, is given by

$$\text{CR} = \frac{b_1}{b_2} \tag{17.2}$$

For the case $b_2 = b_1$, CR $= 1$, and $R = 0$, it can be concluded that the first data set contains no redundant data. When $b_2 \ll b_1$, CR $\to \infty$, and $R \to 1$, indicating high redundancy, and, consequently, high compression. Finally, when $b_2 \gg b_1$, CR $\to 0$

and $R \to -\infty$, so it can be concluded that the second data set contains much more data than the first, suggesting the undesirable case of data expansion. In general, CR and R lie within the $[0, \infty]$ and $[-\infty, 1]$, respectively. A compression ratio such as 100 (or 100:1) basically means that the first data set has 100 information units (e.g., bits) for each unit in the second (compressed) data set. The corresponding redundancy (0.99 in this case) indicates that 99% of the data in the first data set are redundant.

Image compression and coding techniques exploit three types of redundancies that are commonly present in 2D image arrays: coding redundancy, interpixel (spatial) redundancy, and psychovisual redundancy. The way each of them is exploited is briefly described next.

Coding Redundancy The 8 bits per pixel (per color) conventionally used to represent pixels in an image contain an amount of redundancy. From the perspective of coding theory, each 8-bit value is a fixed-length *codeword* used to represent a piece of information (in this case, the intensity or color values of each pixel). Entropy-based coding techniques (e.g., Huffman coding, Golomb coding, and arithmetic coding) replace those fixed-length codewords with a variable-length equivalent in such a way that the total number of bits needed to represent the same information (b_2) is less than the original number of bits (b_1).[1] This type of coding is always reversible and usually implemented using lookup tables (LUTs).

Interpixel Redundancy This type of redundancy—sometimes called *spatial* redundancy, *interframe* redundancy, or *geometric* redundancy—exploits the fact that an image very often contains strongly correlated pixels—in other words, large regions whose pixel values are the same or almost the same. This redundancy can be exploited in several ways, one of which is by predicting a pixel value based on the values of its neighboring pixels. To do so, the original 2D array of pixels is usually mapped into a different format, for example, an array of differences between adjacent pixels. If the original image pixels can be reconstructed from the transformed data set, the mapping is said to be *reversible*. Examples of compression techniques that exploit the interpixel redundancy include constant area coding (CAC), Lempel–Ziv–Welch (LZW), (1D or 2D) run-length encoding (RLE) techniques, and many predictive coding algorithms such as differential pulse code modulation (DPCM).

Psychovisual Redundancy Many experiments on the psychophysical aspects of human vision have proven that the human eye does not respond with equal sensitivity to all incoming visual information; some pieces of information are more important than others. The knowledge of which particular types of information are more or less relevant to the human visual system has led to image compression techniques that aim at eliminating or reducing any amount of data that is psychovisually redundant. The end result of applying these techniques is a compressed image file, whose size

[1] You might remember from Chapter 2 that the total number of bits occupied by an image in uncompressed form, b_1, can be easily calculated by multiplying the width, height, number of bits per color channel (usually 8), and number of color channels (1 for monochrome, 3 for color images) in the input image.

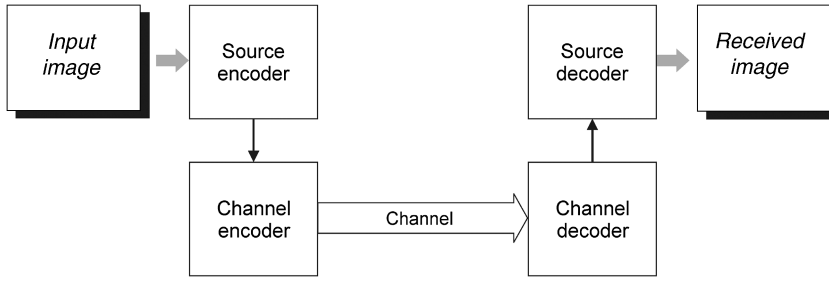


FIGURE 17.2 A general image encoding and decoding model.

and quality are smaller than those of the original information, but whose resulting quality is still acceptable for the application at hand. The loss of quality that ensues as a by-product of such techniques is frequently called *quantization*, so as to indicate that a wider range of input values is normally mapped into a narrower range of output values through an irreversible process.

Video compression techniques also exploit a fourth type of redundancy, interframe (or *temporal*) redundancy, caused by the fact that consecutive frames in time are usually very similar.

17.2.2 Image Encoding and Decoding Model

Figure 17.2 shows a general image encoding and decoding model. It consists of a source encoder, a channel encoder, the storage or transmission media (also referred to as *channel*), a channel decoder, and a source decoder. The source encoder reduces or eliminates any redundancies in the input image, which usually leads to bit savings. Source encoding techniques are the primary focus of this chapter. The channel encoder increases noise immunity of source encoder's output, usually adding extra bits to achieve its goals. If the channel is noise-free, the channel encoder and decoder may be omitted. At the receiver's side, the channel and the source decoder perform the opposite functions and ultimately recover (an approximation of) the original image.

Figure 17.3 shows the source encoder in further detail. Its main components are as follows:

- *Mapper*: It transforms the input data into a (usually nonvisual) format designed to reduce interpixel redundancies in the input image. This operation is generally

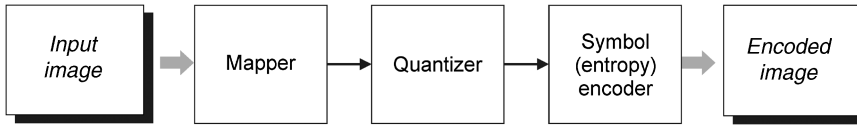


FIGURE 17.3 Source encoder.

reversible and may or may not directly reduce the amount of data required to represent the image.

- *Quantizer*: It reduces the accuracy of the mapper's output in accordance with some pre-established fidelity criterion. This block exploits the psychovisual redundancies of the input image. This operation is not reversible and must be omitted if lossless compression is desired.
- *Symbol (Entropy) Encoder*: It creates a fixed- or variable-length code to represent the quantizer's output and maps the output in accordance with the code. In most cases, a variable-length code is used. This block exploits the coding redundancies of the data produced at the output of the quantizer. This operation is reversible.

17.3 LOSSLESS AND LOSSY COMPRESSION TECHNIQUES

In this section, we present a broad overview of the techniques that lie at the core of the most common lossless and lossy compression algorithms and standards in use today.

17.3.1 Lossless Compression Techniques

Error-Free Compression Using Variable Length Coding (VLC) Error-free compression techniques usually rely on entropy-based encoding algorithms. The concept of entropy provides an upper bound on how much compression can be achieved, given the probability distribution of the source. In other words, it establishes a theoretical limit on the amount of lossless compression that can be achieved using entropy encoding techniques alone. Most entropy-based encoding techniques rely on assigning variable-length codewords to each symbol, whereas the most likely symbols are assigned shorter codewords. In the case of image coding, the symbols may be raw pixel values or the numerical values obtained at the output of the mapper stage (e.g., differences between consecutive pixels, run lengths, etc.). The most popular entropy-based encoding technique is the Huffman code, which is used in the JPEG standard (among others).

Run-Length Encoding RLE is one of the simplest data compression techniques. It consists of replacing a sequence (run) of identical symbols by a pair containing the symbol and the run length. It is used as the primary compression technique in the 1D CCITT Group 3 fax standard and in conjunction with other techniques in the JPEG image compression standard.

Differential Coding Differential coding techniques exploit the interpixel redundancy in digital images. The basic idea consists of applying a simple difference operator to neighboring pixels to calculate a difference image, whose values are likely to follow within a much narrower range than the original gray-level range.

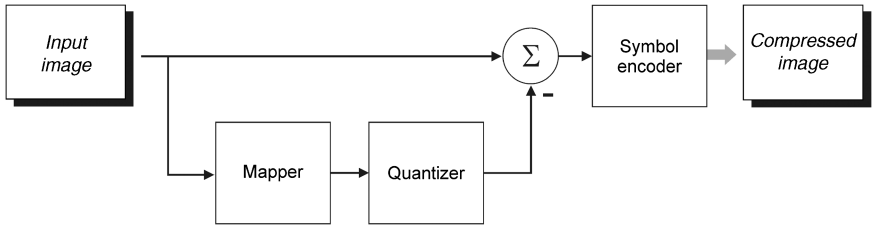


FIGURE 17.4 Lossless predictive encoder.

As a consequence of this narrower distribution, and consequently reduced entropy, Huffman coding or other VLC schemes will produce shorter codewords for the difference image.

Predictive Coding Predictive coding techniques also exploit interpixel redundancy. The basic idea, in this case, is to encode only the new information in each pixel, which is usually defined as the difference between the actual and the predicted value of that pixel. Figure 17.4 shows the main blocks of a lossless predictive encoder. The key component is the *predictor*, whose function is to generate an estimated (predicted) value for each pixel from the input image based on previous pixel values. The predictor's output is rounded to the nearest integer and compared with the actual pixel value: the difference between the two (called *prediction error*) is then encoded by a VLC encoder. Since prediction errors are likely to be smaller than the original pixel values, the VLC encoder will likely generate shorter codewords.

There are several local, global, and adaptive prediction algorithms in the literature. In most cases, the predicted pixel value is a linear combination of previous pixels.

Dictionary-Based Coding Dictionary-based coding techniques are based on the idea of incrementally building a dictionary (a special type of *table*) while receiving the data. Unlike VLC techniques, dictionary-based techniques use fixed-length codewords to represent variable-length strings of symbols that commonly occur together. Consequently, there is no need to calculate, store, or transmit the probability distribution of the source, which makes these algorithms extremely convenient and popular. The best-known variant of dictionary-based coding algorithms is the LZW (Lempel–Ziv–Welch) encoding scheme, used in popular multimedia file formats such as GIF, TIFF, and PDF.

17.3.2 Lossy Compression Techniques

Lossy compression techniques deliberately introduce a certain amount of distortion to the encoded image, exploring the psychovisual redundancies of the original image. These techniques must find an appropriate balance between the amount of error (loss) and the resulting bit savings.

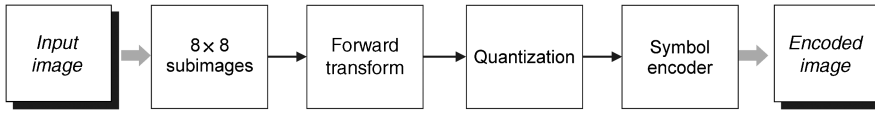


FIGURE 17.5 Transform coding diagram.

Quantization The quantization stage is at the core of any lossy image encoding algorithm. Quantization, in this context, is the process of partitioning of the input data range into a smaller set of values. There are two main types of quantizers: *scalar* quantizers and *vector* quantizers. A scalar quantizer partitions the domain of input values into a smaller number of intervals. If the output intervals are equally spaced, the process is called *uniform scalar quantization* or otherwise, for reasons usually related to minimization of total distortion, it is called *nonuniform scalar quantization*. Vector quantization (VQ) techniques extend the basic principles of scalar quantization to multiple dimensions. VQ-based coding schemes are computationally efficient because the decoding can be done using lookup tables.

Transform Coding The techniques discussed so far work directly on the pixel values and are usually called *spatial-domain techniques*. Transform coding techniques use a reversible, linear mathematical transform to map the pixel values onto a set of coefficients that are then quantized and encoded. The key factor behind the success of transform-based coding schemes lies in the ability of the selected mathematical transform to encode a significant amount of the visual information originally present in the image using the few first coefficients. Consequently, most of the resulting coefficients have small magnitudes and can be quantized (or discarded altogether) without causing significant distortion in the decoded image. The discrete cosine transform (DCT) has become the most widely used transform coding technique. DCT-based coding is an essential component of the JPEG and JPEG-LS compression standards.

Transform coding algorithms (Figure 17.5) usually work on a block basis. They start by partitioning the original image into subimages (blocks) of small size (usually 8×8).[2] For each block, the transform coefficients are calculated, effectively converting the original 8×8 array of pixel values into an array of coefficients within which the coefficients closer to the top left corner usually contain most of the information needed to quantize and encode (and eventually perform the reverse process at the decoder's side) the image with little perceptual distortion. The resulting coefficients are then quantized and the output of the quantizer is used by a (combination of) symbol encoding technique(s) to produce the output bitstream representing the encoded image. At the decoder's side, the reverse process takes place, with the obvious difference that the "dequantization" stage will generate only an approximated version of the original coefficient values; in other words, whatever loss was introduced by the quantizer in the encoder stage is not reversible at the decoder.

[2] An undesired side effect of this division is the introduction of noticeable artifacts (known as *blockiness*) in a compressed image encoded using low quality (see Tutorial 17.1).

Wavelet Coding Wavelet coding techniques are also based on the idea of using a transform that decorrelates the pixels of the input image, converting them into a set of coefficients that can be coded more efficiently than the original pixel values themselves. Contrary to DCT-based coding, the subdivision of the input image into smaller subimages is not necessary in wavelet coding. Wavelet coding is a core technique in the JPEG 2000 compression standard.

17.4 IMAGE COMPRESSION STANDARDS

Work on international standards for image compression started in the late 1970s with a United Nations organization called the *Consultative Committee of the International Telephone and Telegraph* (CCITT) (currently *International Telecommunications Union* (ITU-T)) and its need to standardize binary image compression algorithms for Group 3 facsimile communications. Since then, many other committees and standards have been formed to produce *de jure* standards (such as JPEG or JPEG 2000), while several commercially successful initiatives have effectively become *de facto* standards (such as the Adobe Systems' PDF format for document representation). Image compression standards bring many benefits, such as easier exchange of image files between different devices and applications, reuse of existing hardware and software for a wider array of products, and availability of benchmarks and reference data sets for new and alternative developments.

17.4.1 Binary Image Compression Standards

Work on binary image compression standards was initially motivated by CCITT Group 3 and 4 facsimile standards. The Group 3 standard uses Huffman coding and a nonadaptive, 1D RLE technique in which the last $K - 1$ lines of each group of K lines (for $K = 2$ or 4) are optionally coded in a 2D manner, using the Modified Relative Element Address Designate (MREAD) algorithm. The Group 4 standard uses only the MREAD coding algorithm. Both classes of algorithms are nonadaptive and were optimized for a set of eight test images, containing a mix of representative documents, which sometimes resulted in data expansion when applied to different types of documents (e.g., halftone images). The Joint Bilevel Image Group (JBIG)—a joint committee of the ITU-T and the International Standards Organization (ISO)—has addressed these limitations and proposed two new standards (JBIG—also known as JBIG1—and JBIG2) that can be used to compress binary and grayscale images of up to 6 gray-coded bits/pixel.

17.4.2 Continuous Tone Still Image Compression Standards

For photograph quality images (both grayscale and color), different standards have been proposed, mostly based on lossy compression techniques. The most popular standard in this category is the JPEG standard, a lossy, DCT-based coding algorithm.

Despite its great popularity and adoption, ranging from digital cameras to the World Wide Web, certain limitations of the original JPEG algorithm have motivated the recent development of two alternative standards, JPEG 2000 and JPEG-LS (lossless). JPEG, JPEG 2000, and JPEG-LS are described next.

17.4.3 JPEG

The JPEG format was originally published as a standard (ISO IS 10918-1) by the Joint Photographic Experts Group in 1994. It has become the most widely used format for storing digital photographs ever since. The JPEG specification defines how an image is transformed into a stream of bytes, but not how those bytes are encapsulated in any particular storage medium. Another standard, created by the independent JPEG group, called JFIF (JPEG File Interchange Format) specifies how to produce a file suitable for computer storage and transmission from a JPEG stream.

Even though the original JPEG specification defined four compression modes (sequential, hierarchical, progressive, and lossless), most JPEG files used today employ the sequential mode. The baseline JPEG encoder (Figure 17.6, top) consists of the following main steps:

1. The original RGB color image is converted to an alternative color model (YCbCr) and the color information is subsampled.
2. The image is divided into 8×8 blocks.
3. The 2D DCT is applied to each block image; the resulting 64 values are referred to as *DCT coefficients*.
4. DCT coefficients are quantized according to a quantization table; this is the step where acceptable loss is introduced.

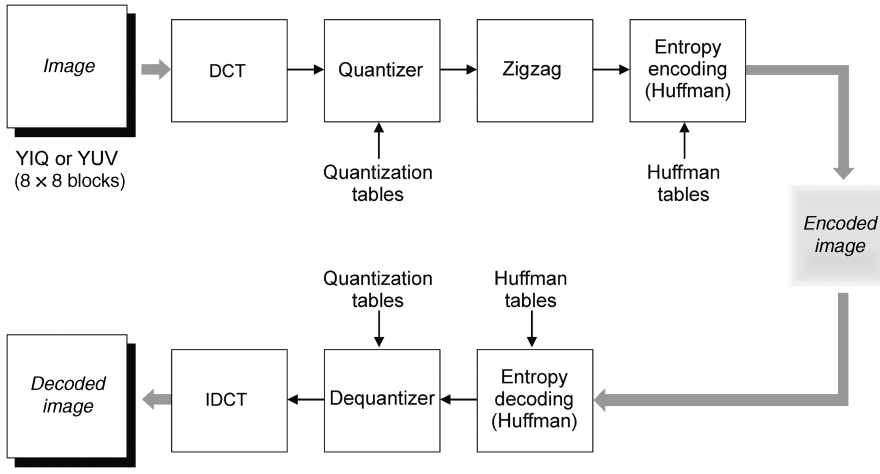


FIGURE 17.6 JPEG encoder and decoder.

5. Quantized DCT coefficients are scanned in a zigzag fashion (from top left to bottom right). The resulting sequence is run-length encoded, in preparation for the entropy encoding step.
6. The run-length encoded sequences are converted to variable-length binary codewords using Huffman encoding.

At the decoder side (Figure 17.6, bottom), the process is reversed; it should be noted that the loss introduced at the quantizer stage in the encoder cannot be canceled at the "dequantizer" stage in the decoder; that is, the values recovered at the output of the "dequantizer" block are approximations of the values produced at the output of the DCT block in the encoder.

17.4.4 JPEG 2000

The JPEG 2000 format [TM01,AT04] is a wavelet-based image compression standard (ISO/IEC 15444-1:2000), created by the Joint Photographic Experts Group committee with the intention of superseding their original DCT-based JPEG standard. The usual file extension is .jp2. It addresses several well-known limitations of the original JPEG algorithm and prepares the way for next-generation imagery applications. Some of its advertised advantages are as follows:

1. Low bit-rate compression
2. Superior lossless and lossy compression in a single bitstream
3. Ability to handle very large images without need for tiling
4. Single decompression architecture
5. Error resilience for transmission in noisy environments, such as wireless communication networks
6. Region of interest (ROI) coding
7. Metadata mechanisms for incorporating additional nonimage data as part of the file

JPEG 2000 is not yet widely supported in web browsers, and hence it is not generally used on the World Wide Web.

17.4.5 JPEG-LS

JPEG-LS is an image compression standard (ISO/IEC-14495-1/ITU-T Rec. T.87) designed to provide effective lossless and near-lossless compression of continuous-tone, grayscale, and color still images. *Near lossless*, in this context, means a guaranteed maximum error between the original image data and the reconstructed image data. One of its main potential driving forces for this encoding mode is the compression of medical images without any quality loss. The core algorithm behind JPEG-LS is called low complexity lossless compression for images (LOCO-I), proposed by

Hewlett-Packard. LOCO-I builds upon a concept known as *context modeling*. The main idea behind context modeling is to take advantage of the structure in the input source, modeled in terms of conditional probabilities of what pixel values follow from each other in the image.

17.5 IMAGE QUALITY MEASURES

The human visual system is the final link in the perception of images and the ultimate judge of image quality. Therefore, in the design and specification of image processing systems, it is necessary to establish methods for measuring the quality of the images displayed to the viewer. Such need can be justified by looking at two extreme cases: (1) there is no reason for designing a system whose image quality exceeds the ability of the human eye to perceive it; (2) on the other hand, since different processing and encoding techniques are likely to cause noticeable changes to the resulting images, it becomes necessary to evaluate the amount and impact of the resulting loss in visual quality.

Measuring image quality is a difficult and often imprecise task, in part because of the vast number of variables that may impact the final result (ranging from display technology to ambient lighting to the subject's mood, among many others). Assessment of visual quality is inherently a subjective process that has traditionally been evaluated with objective quality measures and criteria because such objective measures provide accurate, repeatable results and usually rely on fewer, controllable factors. Despite their mathematical convenience, however, most quantitative objective quality measures do not correlate well with the subjective experience of a human viewer watching the image. Consequently, the design of objective quality measurement systems that match the subjective human judgments is an open research topic.

17.5.1 Subjective Quality Measurement

The subjective quality of an image or video sequence can be established by asking human observers to report, choose, or rank their visual experiences within a controlled experiment. These measures can be *absolute* or *relative*.

Absolute techniques are those in which an image is classified on an isolated basis and without regard to another image. An example is the classification of an image as *excellent*, *fine*, *passable*, *marginal*, *inferior*, or *unusable* according to the criteria of the Television Allocations Study Organization [FB60].

Relative measurements ask subjects to compare an image against another and decide which is best. A popular protocol for these tests is the *double stimulus continuous quality scale* (DSCQS) method defined in ITU-R Recommendation BT.500-10 [ITU00]. In this method, the subject is presented with a pair of images or short video sequences A and B, one after another, and is asked to provide a score recorded on a continuous line with five gradings: *excellent*, *good*, *fair*, *poor*, and *bad*. Experiments usually consist of several tests in which either A or B is (randomly) assigned to the "original" image or video, whereas the other is the video after some processing,

whose impact on perceptual quality is being evaluated. The DSCQS test is often used as a measure of subjective visual quality, but it suffers from several shortcomings, such as the dependency on prior knowledge among (*experts* and *nonexperts*) subjects.

17.5.2 Objective Quality Measurement

Image and video developers often rely on objective measures of visual quality for two main reasons:

- They provide an alternative to subjective measurements and their limitations.
- They are easy to compute.

The most common measures of image quality are the root mean square (RMS) error and the peak signal to noise ratio (PSNR). Let $f(x, y)$ be the original (or reference) image and $f'(x, y)$ the modified (or reconstructed) image. For every value of x and y, the error $e(x, y)$ between $f(x, y)$ and $f'(x, y)$ can be defined as

$$e(x, y) = f'(x, y) - f(x, y) \tag{17.3}$$

and the total error between the two images (whose sizes are $M \times N$) is

$$E = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} |f'(x, y) - f(x, y)| \tag{17.4}$$

The RMS error, e_{rms}, between $f(x, y)$ and $f'(x, y)$ can be calculated by

$$e_{\text{rms}} = \sqrt{\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[f'(x, y) - f(x, y)\right]^2} \tag{17.5}$$

If we consider the resulting image, $f'(x, y)$, as the "signal" and the error as "noise," we can define the RMS signal to noise ratio (SNR_{rms}) between modified and original images as

$$\text{SNR}_{\text{rms}} = \sqrt{\frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f'(x, y)^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[f'(x, y) - f(x, y)\right]^2}} \tag{17.6}$$

If we express the SNR as a function of the peak value of the original image and the RMS error (equation (17.5)), we obtain another metric, the PSNR, defined as

$$\text{PSNR} = 10 \log_{10} \frac{(L - 1)^2}{(e_{\text{rms}})^2} \tag{17.7}$$

where L is the number of gray levels (for 8 bits/pixel monochrome images, $L = 256$).

TABLE 17.1 Objective Quality Measures for Three JPEG Images with Different Quality Factors

Quality Factor	File Size (B)	RMS Error	PSNR (dB)
90	10,8792	2.1647	41.4230
20	20,781	6.3553	32.0681
5	5992	7.6188	30.4931

■ EXAMPLE 17.2

In this example, we use the `imwrite` function in MATLAB to generate three JPEG files (using different quality factors) from the same input image. The result of image degradation caused by lossy compression is measured by the RMS error (equation (17.5)) and the PSNR (equation (17.7)). Clearly, the use of lower quality factors leads to a degradation in subjective quality: the image in Figure 17.7c is significantly worse than the image in parts (a) or (b).

Table 17.1 summarizes the quality factors, file size (in B), RMS error, and PSNR for the images in Figure 17.7a–c.

Objective quality measures suffer from a number of limitations:

- They usually require an "original" image (whose quality is assumed to be perfect). This image may not always be available and the assumption that it is "perfect" may be too strong in many cases.
- They do not correlate well with human subjective evaluation of quality. For example, in Figure 17.8 most subjects would say that the rightmost image (in which an important portion of the image got severely blurred) has significantly worse subjective quality than the one on the center (where the entire image was subject to a mild blurring), even though their RMS errors (measured against the reference image on the left) are comparable.

Finding objective measures of image and video quality that match human quality evaluation is an ongoing field of research and the subject of recent standardization efforts by the ITU-T Video Quality Experts Group (VQEG), among other groups.

17.6 TUTORIAL 17.1: IMAGE COMPRESSION

Goal

The goal of this tutorial is to show how we can perform image compression using MATLAB.

Objectives

- Experiment with image compression concepts, such as compression ratio.
- Calculate objective quality measures on compressed images.

(a)

(b)

(c)

FIGURE 17.7 Measuring objective image quality after compression: (a) original; (b) compressed version of (a) (using quality factor = 90), $e_{\text{rms}} = 2.1647$, PSNR = 41.4230 dB; (c) compressed version of (a) (using quality factor = 5), $e_{\text{rms}} = 7.6188$, PSNR = 30.4931 dB.

(a)

(b)

(c)

FIGURE 17.8 The problem of poor correlation between objective and subjective measures of image quality: (a) original; (b) blurred version of (a) (using a 5×5 average filter), $e_{\text{rms}} = 0.0689$, PSNR $= 71.3623$ dB; (c) partially blurred version of (a) (after applying a severe blurring filter only to a small part of the image), $e_{\text{rms}} = 0.0629$, PSNR $= 72.1583$ dB.

What You Will Need

- `coat_of_arms.jpg`
- `espresso_color.jpg`
- `statue.png`

Procedure

In this tutorial, you will have a chance to adjust the quality factor in a JPEG encoder and make quantitative observations on the trade-off between the resulting compression ratio and the associated quality loss.

1. Use the sequence below to calculate the compression ratio in the JPEG test file `coat_of_arms.jpg`.

```
filename = 'coat_of_arms.jpg';
I = imread(filename);
fileInfo = dir(filename)
imageInfo = whos('I')
fileSize = fileInfo.bytes
imageSize = imageInfo.bytes
CR = imageSize / fileSize
```

2. Repeat the process for the JPEG test file `espresso_color.jpg`.

Question 1 Are the compression ratio values calculated for the two test images comparable? Explain.

3. Load file `statue.png`, convert it to JPEG using MATLAB's `imwrite` function, and choose five different values for quality, 50, 25, 15, 5, and 0, following the example below. Store these results in files named `statue50.jpg`, ..., `statue00.jpg`.

```
org = imread('statue.png');
imwrite(org, 'statue50.jpg', 'quality', 50);
```

4. Display the original image as well as the five compressed images (in decreasing degree of quality).
5. Calculate the compression ratio for each file following the example below:

```
K = imfinfo('statue50.jpg');
image_bytes = K.Width*K.Height*K.BitDepth/8;
compressed_bytes = K.FileSize;
CR = image_bytes / compressed_bytes
```

TABLE 17.2 Compression Ratio for Five JPEG Images with Different Quality Factors

File Name	File Size (B)	CR
`statue.png`	243825	N/A
`statue50.jpg`		
`statue25.jpg`		
`statue15.jpg`		
`statue05.jpg`		
`statue00.jpg`		

TABLE 17.3 Objective Quality Measures for Five JPEG Images with Different Quality Factors

File Name	RMS Error	PSNR (dB)	Subjective Quality
`statue.png`	N/A	N/A	Very good
`statue50.jpg`			
`statue25.jpg`			
`statue15.jpg`			
`statue05.jpg`			
`statue00.jpg`			

Question 2 How do the results for CR obtained with this method compare with the ones obtained with the method suggested in step 1?

6. Log the results to Table 17.2.
7. Write code to calculate and display the RMS error and the PSNR (dB) between the original image and each of the lower quality resulting images.
8. Show the results in Table 17.3 and compare them with your subjective evaluation of image quality.

WHAT HAVE WE LEARNED?

- Image compression is a process by which the total size of a digital image representation (in bytes) is reduced without significant loss in the visual quality of the resulting image. The compressed image will occupy less storage space on a computer and take less time to be transmitted from one computer (e.g., Web server) to another (e.g., Web client browser) over a computer network.
- Three types of redundancy are usually exploited when compressing images: coding, interpixel (spatial), and psychovisual. Video compression techniques also exploit interframe (temporal) redundancy to achieve higher compression ratios (fewer bits per pixel).

- Image compression techniques often employ general data compression techniques capable of exploiting coding redundancy (e.g., Huffman and arithmetic coding) and spatial redundancy (e.g., predictive coding).
- The main difference between image compression methods and general data compression techniques is that most image compression techniques allow a certain amount of loss to be introduced during the compression process. This loss, which usually can be controlled by some parameters in the encoding algorithm, is the consequence of exploiting psychovisual redundancies within the image. Image compression techniques are classified as *lossy* or *lossless* depending on whether such quality loss is present.
- The field of image compression has experienced intense activity over the past 30 years. As a result, many (lossy and lossless) image compression techniques have been proposed, a fraction of which have been adopted as part of widely used image standards and file formats.
- Subjective evaluation of image compression algorithms can be done in absolute or relative ways. Objective evaluation consists of straightforward methods and calculations (e.g., PSNR and RMS errors). The design of objective image and video quality measures with strong correlation to subjective perception of quality is an ongoing research topic.
- The most popular image compression standards in use today are JPEG, JPEG 2000, and JPEG-LS. Other image file formats that use some form of compression are GIF, TIFF, and PNG.

LEARN MORE ABOUT IT

- There are entire books dedicated to classic data compression, such as [NG95], [Say05], and [Sal06].
- Many books on image processing and multimedia have one or more chapters devoted to image coding and compression, for example, Chapter 8 of [GW08] and Chapter 10 of [Umb05].
- The JPEG standard is described in detail in [PM92].
- The JPEG 2000 standard is presented in detail in [TM01,AT04].

ON THE WEB

- The JPEG standard
  http://www.w3.org/Graphics/JPEG/itu-t81.pdf
- The JPEG committee home page
  http://www.jpeg.org/
- ITU-T Video Quality Experts Group
  http://www.its.bldrdoc.gov/vqeg/