12

Image Compression: The JPEG Standard

Presenting the JPEG standard at the level of detail contained in this chapter will require about four hours. To fit within this amount of time, you will have to skip Section 12.4; this section proves the orthogonality of the matrix C and can be seen as the advanced part of this chapter. It is necessary, however, to discuss the relationship between the matrices f and α and to present the 64 basis elements A_{ij}. The central idea underlying the JPEG standard is a change of basis in a 64-dimensional space; this chapter provides the perfect occasion to review this portion of linear algebra.

12.1 Introduction: Lossless and Lossy Compression

Data compression is at the very heart of computer science, and the Internet has made its use an everyday occurrence for most. Many of us may not even know we are using compression, or at least have little knowledge of how the underlying algorithms work. Even so, many compression algorithms have names that are familiar to general computer users (*WinZip*, *gzip*, and, in the UNIX world, *compress*), to music lovers and Internet users (*GIF*, *JPG*, *PNG*, *MP3*, *AAC*, etc). If not for the common use of compression algorithms, the Internet would be completely paralyzed by the volume of uncompressed data being transferred.

The goal of this chapter is to study a commonly used algorithm for the compression of black-and-white or color still images ("still" as opposed to "moving" images). This method of compression is commonly known as JPEG, the acronym of *Joint Photographic Experts Group*, the consortium of companies and researchers that developed and popularized it. The group started its work in June 1987, and the first draft of the standard was published in 1991. Internet users will no doubt associate this compression method with the "jpg" suffix that is a part of the names of many images and photographs transmitted over the Internet. The JPEG algorithm is the most commonly used compression method in digital cameras.

C. Rousseau and Y. Saint-Aubin, *Mathematics and Technology*,
DOI: 10.1007/978-0-387-69216-6_12, © Springer Science+Business Media, LLC 2008

Before diving into the details of this algorithm and the underlying mathematics it is good to have a basic knowledge of data compression in general. There are two broad families of data compression algorithms: those that actually degrade the original information to some extent (called *lossy* algorithms) and those that allow for the reconstruction of the original with perfect accuracy (called *lossless* algorithms). Two simple observations can be made.

The first is that it is impossible to compress *without loss* all files of a given size using the same algorithm. Suppose that such a technique exists for files of exactly N bits in length. Each of these bits can take on 2 different values (0 or 1) and thus there are 2^N distinct N-bit files. If the algorithm compresses each of these files, then each one of them will be represented by some new file containing at most $N-1$ bits. There are 2^{N-1} distinct files of $N-1$ bits, 2^{N-2} distinct files of $N-2$ bits, ..., 2^1 distinct files of 1 bit and a single one with 0 bit. Thus, the number of distinct files containing at most $N-1$ bits is

$$1+2^1+2^2+\cdots+2^{N-2}+2^{N-1}=\sum_{n=0}^{N-1}2^i=\frac{2^N-1}{2-1}=2^N-1.$$

Thus the algorithms we are using must compress at least two of the original N-bit files to some identical file containing fewer than N bits. These two compressed files will then be indistinguishable, and it is impossible to determine which original file they should decompress to. Again: *it is impossible to losslessly compress all files of a given size!*

The second observation is a consequence of the first: when developing a compression algorithm, the person charged with this task must decide whether the information must be preserved perfectly or whether a slight loss (or transformation) is tolerable. Two examples can help make this choice clear while also demonstrating different approaches once this decision has been made.

Webster's Ninth New Collegiate Dictionary has 1592 pages, most being typeset in two columns, each column having around 100 lines, each line having about 70 characters, spaces, or punctuation marks. This amounts to a total of about 22 million characters. These characters can be represented by an alphabet of 256 characters, each being coded by 8 bits, or 1 byte (see Section 12.2). About 22 MB are therefore needed to hold *Webster's*. If one recalls that compact disks store approximately 750 MB, a single CD can carry 34 copies of the whole of *Webster's* (without the figures and drawings, however). No author of a dictionary, an encyclopedia, or a textbook (or *any* book for that matter!) would tolerate the changing of a single character. Thus, in compressing such material it is extremely important to use a lossless compression algorithm allowing for a perfect reconstruction of the original document.

A simple approach to such an algorithm assigns variable length codes to each letter of the alphabet.[1] The most common characters in English are the "␣" (space) character

[1]This approach is common to text compression. Different algorithms may assign codes to "words" rather than "letters," and more complicated algorithms may change the assigned codes based on context.

and the letter “e” followed by the letters “t”, “a”, “o”, “i”, “n”, “h”, “s”, “r” (see Table 12.1). The most uncommonly used letters are “x”, “z”, “j”, and “q”. The actual frequencies depend on the author and the text. They may vary significantly if the text is short. It is natural to try to assign short codes to more frequently occurring characters (such as “␣” and “e”) and longer codes to less frequently occurring ones (such as “j” and “q”). In this manner, characters are represented by a variable number of bits rather than always requiring a single byte. Does this approach violate our first observation? No, since in order for each assigned code to be uniquely decodable the codes for rarely occurring letters will be *longer* than 8 bits. Thus, files containing an unusually high percentage of such characters will actually be longer than the original uncompressed file. The idea of assigning variable length codes to individual symbols as a function of their frequency of use is the main idea underlying Huffman codes.

letter	frequency	letter	frequency	letter	frequency
e	0.125	d	0.047	p	0.018
t	0.088	l	0.041	b	0.016
a	0.080	u	0.027	v	0.010
o	0.077	m	0.026	k	0.0090
i	0.069	w	0.025	j	0.0014
n	0.068	c	0.023	x	0.0014
h	0.066	g	0.022	q	0.0010
s	0.060	f	0.021	z	0.0002
r	0.059	y	0.021		

Table 12.1. Frequencies of letters in Dickens’s *Oliver Twist.* (Spaces and punctuation marks have been ignored. Capital letters have been mapped to the corresponding lowercase letters. *Oliver Twist* contains a little over 680,000 letters.)

Our second example lies a little closer to the subject of this chapter. All computer screens have a finite resolution. Usually, this is measured by counting the number of pixels that it can display. Each pixel may be illuminated to take on any color and intensity.[2] Early screens could display $640 \times 480 = 307{,}200$ pixels.[3] (Resolution is

[2]This is not exactly true. Computer screens are able to reproduce only a portion of the visible color gamut, broken down into a finite set of discrete colors that are roughly uniformly close to each other. As such, they can generally reproduce a large number of colors but not the entire visible spectrum.

[3]It is now common to have displays capable of displaying many millions of pixels, with the largest surpassing four million.

normally reported as “number of pixels per horizontal line $\times$ number of lines.”) Suppose that the Louvre decided to digitize its entire collection of painted works. The museum would ideally like to do this with sufficient quality so as to please art experts. However, at the same time they would like to have lower-quality versions for transmission over the Internet and display on typical computer screens. In this case, it doesn’t make any sense for the image to be of a higher resolution than a typical computer monitor. Thus, the image satisfying art experts and that for display on a typical computer monitor are going to be of very different resolutions and sizes. The latter will contain significantly less detail but will be entirely satisfactory for displaying on a monitor. In fact, transmitting the higher-quality image would be a complete waste of time given the limited resolution of the display! The decision about the number of pixels to send is then a fairly obvious one. But suppose that Louvre technical people want to further reduce the size of the transmitted files. They argue that mathematicians often approximate functions around a given point by a straight line, and if one looks at the graph of the function and the approximating line they usually agree fairly well, at least locally. If we imagine the pale tones of a picture as the peaks and ridges of a function graph and the dark ones as its valleys, could we use the mathematical idea of approximation to this “function”?

This last question is more physiological than mathematical: can one fool the user by sending a picture that has been “mathematically approximated”? If the answer is yes, it will mean that a certain loss of quality is acceptable depending on the use of the data. Other criteria (such as human physiology) therefore play an equally important role in deciding how to compress. For example, in digitizing music it is useful to know that the (average) human ear is unable to perceive sounds above 20,000 Hz. In fact, the standard used for recording compact discs ignores frequencies over 22,000 Hz and is capable of accurately reproducing only those frequencies below this threshold, a loss that would bother only dogs, bats, or other animals with a keener sense of hearing than our own. For images are there limits to the variations in colors and intensities of light that may be perceived by the human eye? Are our eyes and mind content with receiving less than an exact reproduction of an image? Should photographic images and cartoons be compressed in the same manner? The JPEG compression standard, through its successes and its limits, answers these questions.

12.2 Zooming in on a JPEG Compressed Digital Image

A photograph can be digitized in a variety of ways. In the JPEG method the photograph is first divided into very small elements, called *pixels*, each one associated with a uniform color or gray tone. The photograph of a cat in Figure 12.1 has been subdivided into 640×640 pixels. Each of these $640 \times 640 = 409{,}600$ pixels has been associated with a uniform tone of gray between black and white. This particular photograph has been digitized using a scale of 256 gray tones where 0 represents black and 255 represents white. Since $256 = 2^8$, each of these values may be stored using 8 bits (a single byte).

Without compression we would require 409,600 bytes to store the photo of the cat, which equates to roughly 410 KB. (Here we are using the metric convention: a KB represents 1000 bytes, a MB represents 10^6 bytes, etc.) To encode a color image, each pixel is associated with three color values (red, green, and blue) each encoded using an 8-bit value between 0 and 255. An image of this size would require over 1.2 MB to store uncompressed. However, as frequent users of the Internet will know, large color JPEG-compressed images (files with a “jpg” suffix) rarely exceed 100 KB. The JPEG method is thus able to efficiently store the information in the image. The JPEG algorithm's utility is not strictly confined to the Internet. It is the principal standard used in digital photography. Nearly all digital cameras will compress images to JPEG format by default; the compression occurs at the instant the photo is taken, and therefore a part of the information is lost forever. As we will see in this chapter, this loss is usually acceptable, but sometimes it is not. Depending on the specific use of the camera, it is up to the photographer to decide. (Exercise: As of 2006, many digital cameras offer resolutions exceeding 10 million pixels (megapixels). What is the space that would be required by such a color image in an uncompressed form?)

Rather than processing the entire photograph at once, the JPEG standard divides the image into little tiles of 8×8 pixels. Figure 12.1 shows two closeups of the image of the cat. In the bottom left, a 32×32 pixel region has been shown. The bottom right shows a further closeup of an 8×8 region of this closeup. The closeups focus on a small region depicting the intersection of two of the cat's whiskers close to the edge of the table. This particular block of the image is unique in that it contains fine details and high contrast. This is not typical of most 8×8 tiles! In most of the image we see that the changes in color and texture are quite gradual. The surface under the table, the table itself, and even the cat's fur consist largely of smooth gradients when looked at as 8×8 blocks. This is the case with most photographs; just think of any landscape photo containing open regions of land, water or sky. The JPEG standard was built on this uniformity; it tries to represent a nearly uniform 8×8 block using as little information as possible. When such a block contains significant detail (such as is the case in our closeup), the use of more space is accepted.

12.3 The Case of 2×2 Blocks

It is simpler to characterize 2×2 blocks than 8×8 blocks, so we will start with that.

We have seen that gray tones are typically represented using a scale with 256 increments. We could equally imagine a scale with infinitely fine increments that covers all of $[-1, 1]$ or any interval $[-L, L]$ of $\mathbb{R}$. In this case, we may associate negative values with dark grays tending to black and positive values to lighter grays tending to white. The origin would then correspond to a gray between levels 127 and 128 on the scale with 256 levels. Even though this change of scale and origin may be perfectly natural in some ways, it is not necessary for our discussion. We will, however, ignore the fact that our gray tones are integers between 0 and 255 and instead treat them as real numbers in

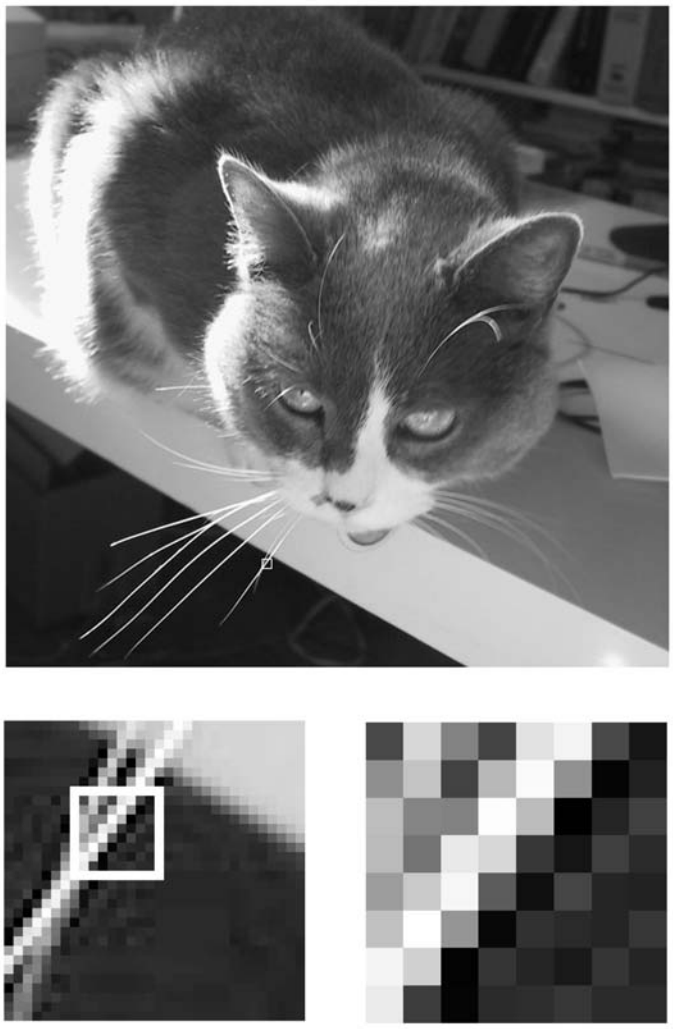

Fig. 12.1. Two successive closeups are made of the original photo (top), which contains 640×640 pixels. The first closeup (bottom left) contains 32×32 pixels. The second closeup (bottom right) contains 8×8 pixels. The white frames on the first and second images denotes the boundaries of the 8×8 closeup in the last image.

this same range. The tone of each pixel will therefore be represented by a real number, and a 2×2 block will require four such values, or equivalently, a point in $\mathbb{R}^4$. (When we are dealing with an $N \times N$ block, we can consider it as a vector in $\mathbb{R}^{N^2}$.)

Given that we perceive the blocks in two dimensions, it is more natural to number the individual pixels using two indices i and j from the set $\{0, 1\}$ (or the set $\{0, 1, \ldots, N-1\}$ when we are dealing with $N \times N$ blocks). The first index will indicate the row, while the second will indicate the column, as is typical in linear algebra. For example, the values of the function f giving the gray tones on the 2×2 square of Figure 12.2 are

$$f = \begin{pmatrix} f_{00} & f_{01} \\ f_{10} & f_{11} \end{pmatrix} = \begin{pmatrix} 191 & 207 \\ 191 & 175 \end{pmatrix}.$$

Many of the functions that we will study naturally take their values in the range $[-1, 1]$. When representing them as gray tones we will use the obvious affine transformation to map them to the range $[0, 255]$. This transformation can be

$$\mathrm{aff}_1(x) = 255(x+1)/2 \tag{12.1}$$

or

$$\mathrm{aff}_2(x) = [255(x+1)/2], \tag{12.2}$$

where $[x]$ denotes the integer part of x. (This last transformation will be used when the values need to be constrained to integers in the range $[0, 255]$. See Exercise 1.) We will use f to denote a function defined in the range $[0, 255]$ and g to denote functions defined in the range $[-1, 1]$. The following box summarizes this notation and specifies the translation we will use. Using this method, the function g associated with the above function f is

$$g = \begin{pmatrix} g_{00} & g_{01} \\ g_{10} & g_{11} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{5}{8} \\ \frac{1}{2} & \frac{3}{8} \end{pmatrix}:$$

$$\boxed{\begin{array}{ccc} f_{ij} \in [0, 255] \subset \mathbb{Z} & \longleftrightarrow & g_{ij} \in [-1, 1] \subset \mathbb{R} \\ f_{ij} = \mathrm{aff}_2(g_{ij}), \quad \text{where} & & \mathrm{aff}_2(x) = \left[\frac{255}{2}(x+1)\right]. \end{array}}$$

We will graphically represent a 2×2 block in two different manners. The first will be simply to draw it using the associated gray tones that would appear in a photograph. The second is to interpret the values g_{ij} as a two-dimensional function of the variables i and j, $i, j \in \{0, 1\}$. Figure 12.2 represents the function $g = (g_{00}, g_{01}, g_{10}, g_{11}) = (\frac{1}{2}, \frac{5}{8}, \frac{1}{2}, \frac{3}{8})$ in these two manners. The coefficients giving the gray values for both the top left g_{00} and bottom left g_{10} pixels are identical. Those of the right column are g_{01} (the paler of the two) and g_{11}. In other words, if we use the matrix notation

$$g = \begin{pmatrix} g_{00} & g_{01} \\ g_{10} & g_{11} \end{pmatrix},$$

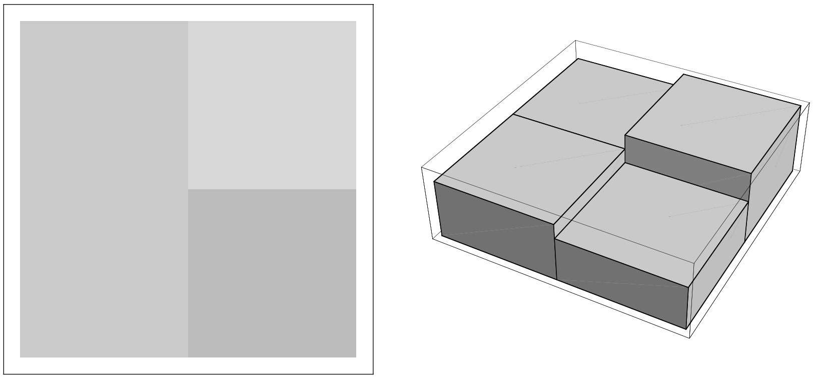

Fig. 12.2. Two graphical representations of the function $g = (g_{00}, g_{01}, g_{10}, g_{11}) = (\frac{1}{2}, \frac{5}{8}, \frac{1}{2}, \frac{3}{8})$.

then the elements of the matrix g are in the same positions as the pixels of Figure 12.2. The second image interprets these same values but displays them as a histogram in two variables i and j, with darker colors being associated to lesser heights. This particular 2×2 block was chosen because all of the pixels are closely related gray tones, as is typical of most 2×2 blocks in a photograph. (In fact, the higher the resolution of the photo, the gentler the gradients become.)

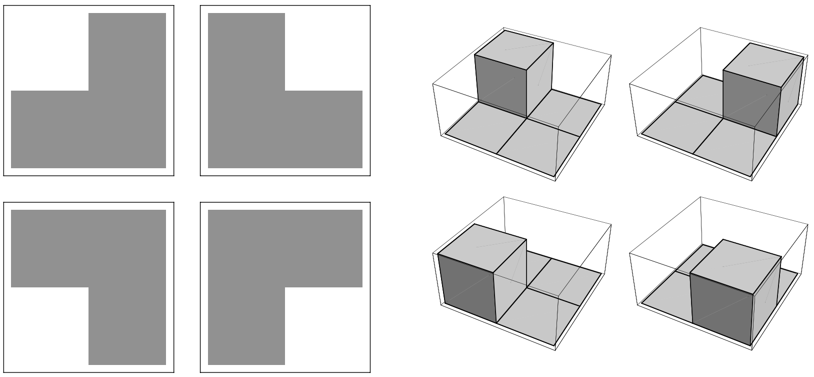

Fig. 12.3. The four elements of the usual basis $\mathcal{B}$ of $\mathbb{R}^4$ represented graphically.

The coordinates $(g_{00}, g_{01}, g_{10}, g_{11})$ (or equivalently $(f_{00}, f_{01}, f_{10}, f_{11})$) represent the small 2×2 block without any loss. (In other words, no compression has yet been done.) These coordinates are expressed in the usual basis $\mathcal{B}$ of $\mathbb{R}^4$, where each element of the basis contains a single nonzero entry with value 1. This basis is depicted graphically in

Figure 12.3. If we were to apply a change of basis

$$[g]_{\mathcal{B}} = \begin{pmatrix} g_{00} \\ g_{01} \\ g_{10} \\ g_{11} \end{pmatrix} \mapsto [g]_{\mathcal{B}'} = \begin{pmatrix} \beta_{00} \\ \beta_{01} \\ \beta_{10} \\ \beta_{11} \end{pmatrix} = [P]_{\mathcal{B}'\mathcal{B}}[g]_{\mathcal{B}},$$

the new coordinates β_{ij} would also accurately represent the contents of the block. The coordinates g_{ij} are not appropriate to our end goal. In fact, we would like to easily recognize blocks where all of the pixels are nearly the same color or gray tone. To do this, it is useful to construct a basis in which completely uniform blocks are represented by a single nonzero coefficient. Similarly, we would like a cursory inspection of the coordinates to reveal when the block is far from being uniform.

The JPEG standard proposes using another basis $\mathcal{B}' = \{A_{00}, A_{01}, A_{10}, A_{11}\}$. Each element A_{ij} of this basis can be expressed using the standard basis shown in Figure 12.3. In the standard basis $\mathcal{B}$ their coefficients are

$$[A_{00}]_{\mathcal{B}} = \begin{pmatrix} \frac12 \\ \frac12 \\ \frac12 \\ \frac12 \end{pmatrix}, \quad [A_{01}]_{\mathcal{B}} = \begin{pmatrix} \frac12 \\ -\frac12 \\ \frac12 \\ -\frac12 \end{pmatrix}, \quad [A_{10}]_{\mathcal{B}} = \begin{pmatrix} \frac12 \\ \frac12 \\ -\frac12 \\ -\frac12 \end{pmatrix}, \quad [A_{11}]_{\mathcal{B}} = \begin{pmatrix} \frac12 \\ -\frac12 \\ -\frac12 \\ \frac12 \end{pmatrix}. \tag{12.3}$$

The elements of this new basis are represented graphically in Figure 12.4. The first element A_{00} represents a uniform block. If the 2×2 block is completely uniform, only the coefficient of A_{00} will be nonzero. The two elements A_{01} and A_{10} represent left/right and top/bottom contrasts, respectively. The last element A_{11} represents a mixture of these two, where each pixel is in contrast with its neighbor along both directions, much like a checkerboard.

Knowing the A_{ij} in the standard basis, it is easy to obtain the change of basis matrix $[P]_{\mathcal{B}\mathcal{B}'}$ from $\mathcal{B}'$ to $\mathcal{B}$. In fact, its columns are given by the coordinates of the elements of $\mathcal{B}'$ expressed in the basis $\mathcal{B}$. It is therefore given by

$$[P]_{\mathcal{B}\mathcal{B}'} = [P]^{-1}_{\mathcal{B}'\mathcal{B}} = \begin{pmatrix} \frac12 & \frac12 & \frac12 & \frac12 \\ \frac12 & -\frac12 & \frac12 & -\frac12 \\ \frac12 & \frac12 & -\frac12 & -\frac12 \\ \frac12 & -\frac12 & -\frac12 & \frac12 \end{pmatrix}. \tag{12.4}$$

To calculate $[g]_{\mathcal{B}'}$ we will need to use $[P]_{\mathcal{B}'\mathcal{B}}$, that is, the inverse of $[P]_{\mathcal{B}\mathcal{B}'}$. Here the matrix $[P]_{\mathcal{B}\mathcal{B}'}$ is orthogonal. (Exercise: A matrix A is orthogonal if $A^tA = AA^t = I$. Verify that $P_{\mathcal{B}\mathcal{B}'}$ is orthogonal.) The computation is therefore easy:

$$[P]_{\mathcal{B}'\mathcal{B}} = [P]^{-1}_{\mathcal{B}\mathcal{B}'} = [P]^{t}_{\mathcal{B}\mathcal{B}'} = [P]_{\mathcal{B}\mathcal{B}'}.$$

The last equality comes from the fact that the matrix $[P]_{\mathcal{B}\mathcal{B}'}$ is symmetric. The coefficients of g in this basis are simply

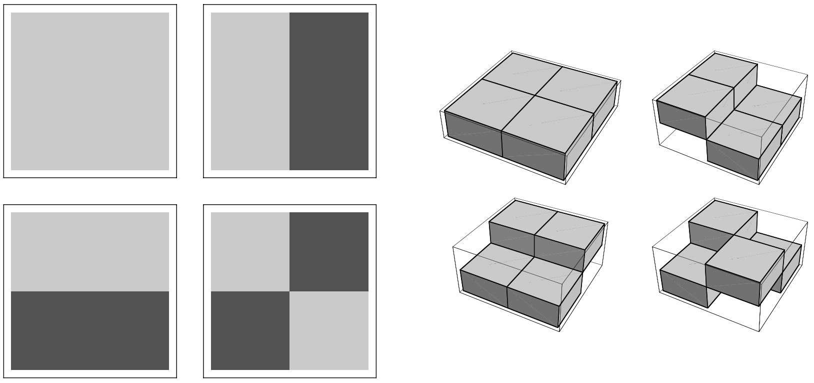

Fig. 12.4. The four elements of the proposed basis $\mathcal{B}'$. (Element A_{00} is at the upper left and element A_{01} is at the upper right.)

$$[g]_{\mathcal{B}'} = \begin{pmatrix} \beta_{00} \\ \beta_{01} \\ \beta_{10} \\ \beta_{11} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} \frac{1}{2} \\ \frac{5}{8} \\ \frac{1}{2} \\ \frac{3}{8} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \frac{1}{8} \\ -\frac{1}{8} \end{pmatrix}.$$

In this basis the largest coefficient is $\beta_{00} = 1$. This is the weight of the element A_{00} that gives an equal importance to each of the four pixels; in other words, this element of the new basis assigns them all the same gray tone. The two remaining nonzero coefficients, both much smaller in magnitude ($\beta_{10} = -\beta_{11} = \frac{1}{8}$), contain information regarding the small amount of contrast between the left and the right columns, and between the two pixels in the right column. The careful choice of the basis highlights spatial contrast information rather than giving individual pixel information. This is the heart of the JPEG standard. To make this technique lossy, one needs only to decide what coefficients correspond to visible contrasts for each of the elements of the basis. The rest of the coefficients may simply be thrown away.

12.4 The Case of $N \times N$ Blocks

The JPEG standard divides the image into 8×8 blocks. The definition of the basis that puts the focus on contrast information rather than individual pixels can equally be defined for arbitrary $N \times N$ blocks. The basis $\mathcal{B}'$ that we introduced in the previous section ($N = 2$) and that used in the JPEG standard ($N = 8$) are particular cases.

The *discrete cosine transform*[4] replaces the function $\{f_{ij},\ i,j = 0,1,2,\dots,N-1\}$ defined over an $N \times N$ square grid by a set of coefficients $\alpha_{kl},\ k,l = 0,1,\dots,N-1$.

[4]The discrete cosine transform is a particular instance of a more general mathematical technique called *Fourier analysis*. Introduced at the beginning of the nineteenth century by

The coefficients α_{kl} are given by

$$\alpha_{kl} = \sum_{i,j=0}^{N-1} c_{ki} c_{lj} f_{ij}, \qquad 0 \le k, l \le N-1, \tag{12.5}$$

where the c_{ij} are defined as

$$c_{ij} = \frac{\delta_i}{\sqrt{N}} \cos \frac{i(2j+1)\pi}{2N}, \qquad i, j = 0, 1, \dots, N-1, \tag{12.6}$$

with

$$\delta_i = \begin{cases} 1, & \text{if } i = 0, \\ \sqrt{2}, & \text{otherwise.} \end{cases} \tag{12.7}$$

(Exercise: For the case $N = 2$, show that the coefficients c_{ij} are given by

$$C = \begin{pmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}.$$

Is it possible for the transformation (12.5) to be equivalent to the change of basis embodied by the matrix $[P]_{\mathcal{B}\mathcal{B}'}$ of (12.4)? Explain.)

The transformation in (12.5) from the $\{f_{ij}\}$ to the $\{\alpha_{kl}\}$ is clearly linear. By writing

$$\alpha = \begin{pmatrix} \alpha_{00} & \alpha_{01} & \cdots & \alpha_{0,N-1} \\ \alpha_{10} & \alpha_{11} & \cdots & \alpha_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{N-1,0} & \alpha_{N-1,1} & \cdots & \alpha_{N-1,N-1} \end{pmatrix}, \quad f = \begin{pmatrix} f_{00} & f_{01} & \cdots & f_{0,N-1} \\ f_{10} & f_{11} & \cdots & f_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ f_{N-1,0} & f_{N-1,1} & \cdots & f_{N-1,N-1} \end{pmatrix},$$

and

$$C = \begin{pmatrix} \sqrt{\frac{1}{N}} & \sqrt{\frac{1}{N}} & \cdots & \sqrt{\frac{1}{N}} \\ \sqrt{\frac{2}{N}} \cos \frac{\pi}{2N} & \sqrt{\frac{2}{N}} \cos \frac{3\pi}{2N} & \cdots & \sqrt{\frac{2}{N}} \cos \frac{(2N-1)\pi}{2N} \\ \sqrt{\frac{2}{N}} \cos \frac{2\pi}{2N} & \sqrt{\frac{2}{N}} \cos \frac{6\pi}{2N} & \cdots & \sqrt{\frac{2}{N}} \cos \frac{2(2N-1)\pi}{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \sqrt{\frac{2}{N}} \cos \frac{(N-1)\pi}{2N} & \sqrt{\frac{2}{N}} \cos \frac{3(N-1)\pi}{2N} & \cdots & \sqrt{\frac{2}{N}} \cos \frac{(2N-1)(N-1)\pi}{2N} \end{pmatrix},$$

we see that the transformation of (12.5) takes on the matrix form

$$\alpha = C f C^t, \tag{12.8}$$

Jean Baptiste Joseph Fourier for studying the propagation of heat, this technique has since invaded the world of engineering. It also plays an important role in Chapter 10.

where C^t denotes the transpose of the matrix C. In fact,

$$\alpha_{kl} = [\alpha]_{kl} = [CfC^t]_{kl} = \sum_{i,j=0}^{N-1} [C]_{ki}[f]_{ij}[C^t]_{jl} = \sum_{i,j=0}^{N-1} c_{ki} f_{ij} c_{lj},$$

which is the same as (12.5).

This transformation is an isomorphism if the matrix C is invertible. (That this is the case will be shown later.) If it is so, we are able to write

$$f = C^{-1}\alpha(C^t)^{-1}$$

and recover the values $f_{ij}, i, j = 0, 1, \ldots, N-1$, from the $\alpha_{kl}, k, l = 0, 1, \ldots, N-1$. The transformation $f \mapsto \alpha$ given by (12.8) is also a linear transformation. Indeed, suppose that f and g are related to α and β through (12.8) (namely $\alpha = CfC^t$ and $\beta = CgC^t$). Then

$$C(f+g)C^t = CfC^t + CgC^t = \alpha + \beta$$

follows from the distributivity of matrix multiplication. And if $c \in \mathbb{R}$ then

$$C(cf)C^t = c(CfC^t) = c\alpha.$$

The two previous identities are the defining properties of linear transformations. Since this linear transformation is an isomorphism, it is a *change of basis*! Note that the passage from f to α is not expressed through a matrix $[P]_{\mathcal{B}'\mathcal{B}}$ as in the previous section. But linear algebra assures us that the transformation $f \mapsto \alpha$ could be written with such a matrix. (If the two indices of f run through $\{0, 1, \ldots, N-1\}$, then there are N^2 coordinates f_{ij}, and the matrix $[P]_{\mathcal{B}'\mathcal{B}}$ doing the change of basis is of size $N^2 \times N^2$. The form (12.8) has the advantage of using only $N \times N$ matrices.)

The proof of the invertibility of C rests on the observation that C is orthogonal:

$$C^t = C^{-1}. \tag{12.9}$$

This observation simplifies the calculations because the above expression for f becomes

$$f = C^t\alpha C. \tag{12.10}$$

We will give a proof of this property at the end of the section.

For the moment we will accept this fact and give an example of the transformation $f \mapsto \alpha$. To do this we will use the gray tones defined over the 8×8 block of Figure 12.1. The $f_{ij}, 0 \le i, j \le 7$, are given in Table 12.2. The positions of pixels in the picture correspond to positions of entries in the table, and the entries are the gray intensities with 0 = black and 255 = white. The large numbers (> 150) correspond to the two white whiskers. The principal characteristic of this 8×8 block is the presence of diagonal stripes with high contrast. We will see how this contrast influences the coefficients α of this function.

The α_{kl} of the function f from Table 12.2 are given in Table 12.3. They are presented in the same order as previously, with α_{00} in the upper left and α_{07} in the upper right. None of the entries are exactly zero-valued, but we see that the largest coefficients (in terms of absolute value) are α_{00}, α_{01}, α_{12}, α_{23}, ... To interpret these numbers we need to have a better "visual" understanding of the elements of the basis $\mathcal{B}'$.

Consider once again the change of basis expressions

$$\alpha = CfC^t \qquad \text{and} \qquad f = C^t\alpha C.$$

In terms of the coefficients themselves, the relationship giving f from α is

$$f_{ij} = \sum_{k,l=0}^{N-1} \alpha_{kl}(c_{ki}c_{lj}).$$

Let A_{kl} be the $N \times N$ matrix whose elements are $[A_{kl}]_{ij} = c_{ki}c_{lj}$. We see that f is a linear combination of the matrices A_{kl} with weights α_{kl}. The set of N^2 matrices $\{A_{kl}, 0 \le k, l \le N-1\}$ forms a basis in terms of which the function f is described. The 64 basis matrices A_{kl} of this example ($N = 8$) are shown in Figure 12.5. Matrix

40	193	89	37	209	236	41	14
102	165	36	150	247	104	7	19
157	92	88	251	156	3	20	35
153	75	220	193	29	13	34	22
116	173	240	54	11	38	20	19
162	255	109	9	26	22	20	29
237	182	5	28	20	15	28	20
222	33	8	23	24	29	23	23

Table 12.2. The 64 values of the function f.

681.63	351.77	−8.671	54.194	27.63	−55.11	−23.87	−15.74
144.58	−94.65	−264.52	5.864	7.660	−89.93	−24.28	−12.13
−31.78	−109.77	9.861	216.16	29.88	−108.14	−36.07	−24.40
23.34	12.04	53.83	21.91	−203.72	−167.39	0.197	0.389
−18.13	−40.35	−19.88	−35.83	−96.63	47.27	119.58	36.12
11.26	9.743	24.22	−0.618	0.0879	47.44	−0.0967	−23.99
0.0393	−12.14	0.182	−11.78	−0.0625	0.540	0.139	0.197
0.572	−0.361	0.138	−0.547	−0.520	−0.268	−0.565	0.305

Table 12.3. The 64 coefficients α_{kl} of the function f.

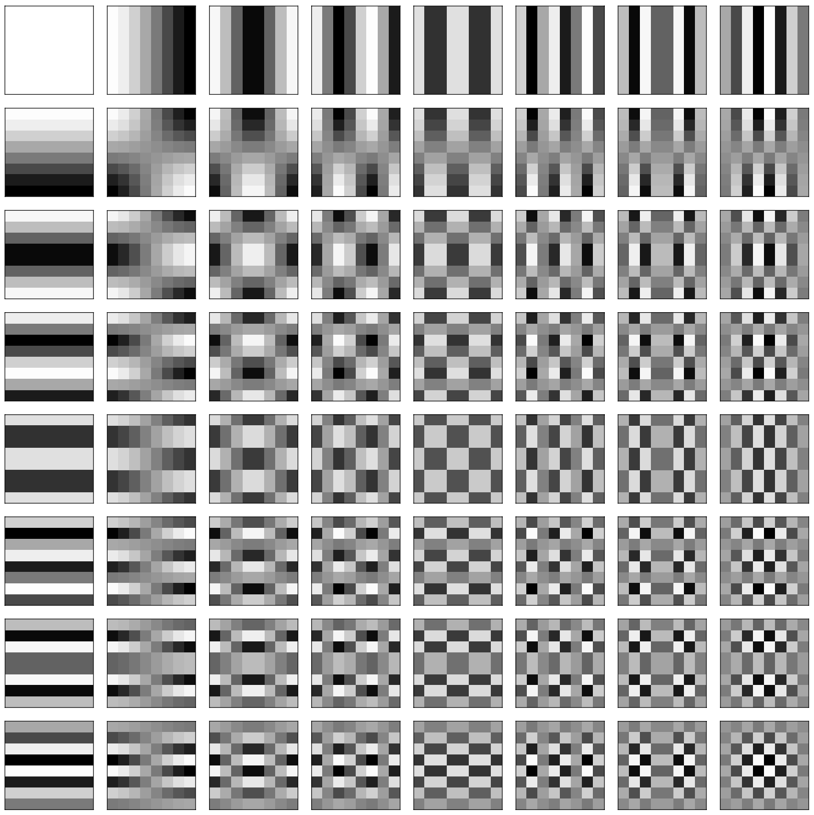

Fig. 12.5. The 64 elements A_{kl} of the basis $\mathcal{B}'$. Element A_{00} is at the upper left and element A_{07} is at the upper right.

A_{00} is in the upper left corner of the image, while A_{07} is found in the upper right. To graphically represent each basis matrix we needed to have their coefficients mapped to gray tones in the range 0 to 255. This was done by first replacing the $[A_{kl}]_{ij}$ by

$$[\tilde{A}_{kl}]_{ij} = \frac{N}{\delta_k \delta_l}[A_{kl}]_{ij},$$

where δ_k and δ_l are given by (12.7). This transformation ensures that $[\tilde{A}_{kl}]_{ij} \in [-1, 1]$. Next, the transformation aff_2 of (12.2) was applied to each scaled coefficient to obtain

$$[B_{kl}]_{ij} = \mathrm{aff}_2([\tilde{A}_{kl}]_{ij}) = \left[\frac{255}{2}([\tilde{A}_{kl}]_{ij} + 1)\right].$$

The $[B_{kl}]_{ij}$ can be directly interpreted as gray tones, since $0 \le [B_{kl}]_{ij} \le 255$. These are the values represented in Figure 12.5.

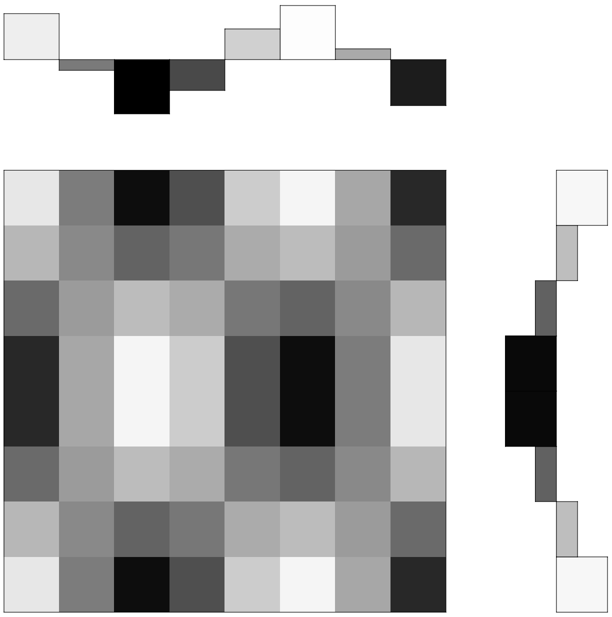

Fig. 12.6. Constructing the graphic representation of A_{23}.

It is possible to understand the graphic representations of the A_{kl} directly from their definitions. Here we consider the details of the construction of the element A_{23}, given by

$$[A_{23}]_{ij} = \frac{2}{N} \cos \frac{2(2i+1)\pi}{2N} \cos \frac{3(2j+1)\pi}{2N}.$$

The upper portion of Figure 12.6 shows the function

$$\cos \frac{3(2j+1)\pi}{16},$$

and at right, vertically, the function

$$\cos \frac{2(2i+1)\pi}{16}$$

has been shown. Since j varies from 0 to $N-1=7$, the argument of the cosine of the first function passes from $3\pi/16$ to $3 \cdot 15\pi/16 = 45\pi/16 = 2\pi + 13\pi/16$ and the figure therefore shows roughly one and one-half cycles of the cosine. Each rectangle of the histogram has been assigned the gray tone corresponding to

$$\frac{255}{2}\left(\cos\left(\frac{3(2j+1)\pi}{16}\right)+1\right).$$

The same process has been repeated for the second function, $\cos 2(2i+1)\pi/16$, and the results of this shown vertically at the right of the figure. The function A_{23} is obtained by multiplying these two functions. This multiplication is between two cosine functions, thus between values in the range $[-1,1]$. The result of this multiplication can be interpreted visually from the image. Multiplying two very light rectangles (corresponding to values near $+1$) or two very dark rectangles (corresponding to values near -1) results in light values. The 8×8 "product" of the two histograms is the matrix of basis element A_{23}.

We return to the 8×8 block depicting the two cat whiskers. What coefficients α_{kl} will be the most important? A coefficient α_{kl} will have larger magnitude if the extrema of the basis matrix correspond roughly to those of f. For example, the basis A_{77} (bottom right corner of Figure 12.5) alternates rapidly between black and white in both directions. It has many extrema, while f depicts only a diagonal pattern. As can be predicted, the associated coefficient is quite small at $\alpha_{77} = 0.305$. On the other hand, the coefficient α_{01} will be quite large. The basis matrix A_{01} (second from the left in the top row of Figure 12.5) contains a bright left half and a dark right half. Even though the two white whiskers of f extend into the right half of the 8×8 block, the left half is significantly paler than the right one. The actual coefficient is $\alpha_{01} = 351.77$.

How should we interpret a negative coefficient α_{kl}? The coefficient $\alpha_{12} = -264.52$ is negative, and a closer inspection yields an answer. The basis matrix A_{12} is roughly divided into six contrasting bright and dark regions, three at the top and three at the bottom. Observe that two of the dark regions are roughly aligned with the brightest region of f, the whiskers. Multiplying this basis matrix by -1 would make these dark regions light, indicating that $-A_{12}$ describes the contrast between the whiskers and the background relatively well, thus the importance of this (negative) coefficient. We can easily repeat this "visual calculation" for each of the basis matrices, but it quickly becomes tedious. In fact, it is faster to program a computer to perform the calculations of (12.5). Regardless, this discussion has demonstrated the following intuitive rule: *the coefficient α_{kl} associated with a function f will have a significant magnitude if the extrema of A_{kl} are similar to those of f. A negative coefficient indicates that the bright spots of f matched dark spots of the basis element and vice versa.* As such, the nearly constant basis matrices A_{00}, A_{01}, and A_{10} are likely to have large factors α_{kl} for nearly constant functions f. At the other extreme, the basis matrices A_{67}, A_{76}, and A_{77} will be important for representing rapidly varying functions.

Proof of the orthogonality of C (12.11): To show this somewhat surprising fact, we rewrite the identify $C^tC = I$ in terms of its coefficients:

$$[C^tC]_{jk} = \sum_{i=0}^{N-1}[C^t]_{ji}[C]_{ik} = \sum_{i=0}^{N-1}[C]_{ij}[C]_{ik} = \delta_{jk} = \begin{cases} 1, & \text{if } j = k,\\ 0, & \text{otherwise,}\end{cases}$$

or equivalently

$$[C^tC]_{jk} = \sum_{i=0}^{N-1} \frac{\delta_i^2}{N} \cos\frac{i(2j+1)\pi}{2N} \cos\frac{i(2k+1)\pi}{2N} = \delta_{jk}. \tag{12.11}$$

Proving (12.11) is equivalent to proving (12.9), the orthogonality of C, which implies the invertibility of (12.5). The proof that follows is not that difficult, but it contains several cases and subcases that must be carefully considered.

We expand the product of cosines from (12.11) using the trigonometric identity

$$\cos\alpha\cos\beta = \frac{1}{2}\cos(\alpha+\beta) + \frac{1}{2}\cos(\alpha-\beta).$$

Let $S_{jk} = [C^tC]_{jk}$. Then we have that

$$\begin{aligned} S_{jk} &= \sum_{i=0}^{N-1} \frac{\delta_i^2}{N} \cos\frac{i(2j+1)\pi}{2N} \cos\frac{i(2k+1)\pi}{2N} \\ &= \sum_{i=0}^{N-1} \frac{\delta_i^2}{2N} \left(\cos\frac{i(2j+2k+2)\pi}{2N} + \cos\frac{i(2j-2k)\pi}{2N}\right) \\ &= \sum_{i=0}^{N-1} \frac{\delta_i^2}{2N} \left(\cos\frac{2\pi i(j+k+1)}{2N} + \cos\frac{2\pi i(j-k)}{2N}\right). \end{aligned}$$

Since $\delta_i^2 = 1$ if $i = 0$ and $\delta_i^2 = 2$ otherwise, we can add the $i = 0$ term and subtract it to obtain

$$S_{jk} = \frac{1}{N}\sum_{i=0}^{N-1} \left(\cos\frac{2\pi i(j+k+1)}{2N} + \cos\frac{2\pi i(j-k)}{2N}\right) - \frac{1}{N}.$$

We split the proof into the following three cases: $j = k$, $j - k$ is even but nonzero, $j - k$ is odd. Observe that exactly one of $(j - k)$ and $(j + k + 1)$ is even, while the other is odd. We consider each of these cases by separating the sum and the term $-\frac{1}{N}$ as follows:

$\boxed{j = k}$ We write $S_{jk} = S_1 + S_2$ with

$$S_1 = -\frac{1}{N} + \frac{1}{N}\sum_{i=0}^{N-1} \cos\frac{2\pi il}{2N}, \qquad S_2 = \frac{1}{N}\sum_{i=0}^{N-1} \cos\frac{2\pi il}{2N},$$

where $l = j + k + 1$ is odd, where $l = j - k = 0$.

$\boxed{j - k \text{ even and } j \neq k}$ Write $S_{jk} = S_1 + S_2$ with

$$S_1 = -\frac{1}{N} + \frac{1}{N}\sum_{i=0}^{N-1} \cos\frac{2\pi il}{2N}, \qquad S_2 = \frac{1}{N}\sum_{i=0}^{N-1} \cos\frac{2\pi il}{2N},$$

where $l = j + k + 1$ is odd, where $l = j - k$ is even and nonzero.

$\boxed{j-k \text{ odd}}$ Write $S_{jk} = S_1 + S_2$ with

$$S_1 = \frac{1}{N}\sum_{i=0}^{N-1}\cos\frac{2\pi il}{2N}, \qquad S_2 = -\frac{1}{N} + \frac{1}{N}\sum_{i=0}^{N-1}\cos\frac{2\pi il}{2N},$$

where $l = j+k+1$ is even, nonzero, and $< 2N$, where $l = j-k$ is odd.

There are three distinct sums to be studied:

$$\frac{1}{N}\sum_{i=0}^{N-1}\cos\frac{2\pi il}{2N}, \qquad \text{where } l = 0, \tag{12.12}$$

$$\frac{1}{N}\sum_{i=0}^{N-1}\cos\frac{2\pi il}{2N}, \qquad \text{where } l \text{ even, nonzero, and } < 2N, \tag{12.13}$$

$$-\frac{1}{N} + \frac{1}{N}\sum_{i=0}^{N-1}\cos\frac{2\pi il}{2N}, \qquad \text{where } l \text{ odd.} \tag{12.14}$$

The first case is simple, since if $l = 0$ it follows that

$$\frac{1}{N}\sum_{i=0}^{N-1}\cos\frac{2\pi il}{2N} = \frac{1}{N}\sum_{i=0}^{N-1} 1 = \frac{N}{N} = 1.$$

Since we wish to show that S_{jk} is zero unless $j = k$ (otherwise, $S_{jj} = 1$), the proof is finished if we can show that (12.13) and (12.14) are both zero. For (12.13) recall that

$$\sum_{i=0}^{2N-1} e^{2\pi il\sqrt{-1}/2N} = \frac{e^{2\pi l\cdot 2N\sqrt{-1}/2N} - 1}{e^{2\pi l\sqrt{-1}/2N} - 1} = 0 \tag{12.15}$$

if $e^{2\pi l\sqrt{-1}/2N} \neq 1$. If $l < 2N$ this inequality is always satisfied. By taking the real part of (12.15) we find that

$$\sum_{i=0}^{2N-1}\cos\frac{2\pi il}{2N} = 0.$$

The sum contains twice as many terms as (12.13). However, we can rewrite it as

$$\begin{aligned}
0 &= \sum_{i=0}^{2N-1} \cos \frac{2\pi il}{2N} \\
&= \sum_{i=0}^{N-1} \cos \frac{2\pi il}{2N} + \sum_{i=N}^{2N-1} \cos \frac{2\pi il}{2N} \\
&= \sum_{i=0}^{N-1} \cos \frac{2\pi il}{2N} + \sum_{j=0}^{N-1} \cos \frac{2\pi(j+N)l}{2N}, \qquad \text{for } i = j+N, \\
&= \sum_{i=0}^{N-1} \cos \frac{2\pi il}{2N} + \sum_{j=0}^{N-1} \cos \left(\frac{2\pi jl}{2N} + \frac{2\pi Nl}{2N} \right).
\end{aligned}$$

If l is even, the phase $\frac{2\pi Nl}{2N} = \pi l$ is an even multiple of π and can therefore be dropped, since the cosine is periodic with period 2π. Thus

$$0 = \sum_{i=0}^{N-1} \cos \frac{2\pi il}{2N} + \sum_{j=0}^{N-1} \cos \frac{2\pi jl}{2N} = 2 \sum_{i=0}^{N-1} \cos \frac{2\pi il}{2N},$$

and hence the sum of (12.13) is zero-valued.

Observe that the first term $i = 0$ of the sum from (12.14) is

$$\frac{1}{N} \cos \frac{2\pi \cdot 0 \cdot l}{2N} = \frac{1}{N},$$

which cancels the term $-\frac{1}{N}$. As such, the sum from (12.14) simplifies to

$$\sum_{i=1}^{N-1} \cos \frac{2\pi il}{2N}.$$

We must now divide case (12.14) into two subcases, N even and N odd. We divide the sum $\sum_{i=1}^{N-1} \cos \frac{2\pi il}{2N}$ as follows:

$\boxed{N \text{ odd}}$

$$\sum_{i=1}^{\frac{N-1}{2}} \cos \frac{2\pi il}{2N} \qquad \text{and} \qquad \sum_{i=\frac{N-1}{2}+1}^{N-1} \cos \frac{2\pi il}{2N}$$

and

$\boxed{N \text{ even}}$

$$\text{the term } i = \frac{N}{2}, \qquad \sum_{i=1}^{\frac{N}{2}-1} \cos \frac{2\pi il}{2N}, \qquad \text{and} \qquad \sum_{i=\frac{N}{2}+1}^{N-1} \cos \frac{2\pi il}{2N}.$$

We start with this last subcase. If N is even, then for $i = N/2$ we have

$$\cos \frac{2\pi}{2N} \cdot \frac{N}{2} \cdot l = \cos \frac{\pi}{2} l = 0,$$

since l is odd. Rewrite the second sum by letting $j = N - i$; since $\frac{N}{2} + 1 \leq i \leq N - 1$, the domain of j is $1 \leq j \leq \frac{N}{2} - 1$:

$$\sum_{i=\frac{N}{2}+1}^{N-1} \cos \frac{2\pi i l}{2N} = \sum_{j=1}^{\frac{N}{2}-1} \cos \frac{2\pi(N-j)l}{2N} = \sum_{j=1}^{\frac{N}{2}-1} \cos \left(\pi l - \frac{2\pi j l}{2N} \right).$$

And since l is odd, the phase πl is always an odd multiple of π, and

$$\sum_{i=\frac{N}{2}+1}^{N-1} \cos \frac{2\pi i l}{2N} = \sum_{j=1}^{\frac{N}{2}-1} - \cos \left(- \frac{2\pi j l}{2N} \right).$$

Since the cosine function is even, we have finally that

$$\sum_{i=\frac{N}{2}+1}^{N-1} \cos \frac{2\pi i l}{2N} = - \sum_{j=1}^{\frac{N}{2}-1} \cos \frac{2\pi j l}{2N},$$

and the two sums of the subcase cancel each other. The subcase of (12.14) where N is odd is left as an exercise to the reader. □

12.5 The JPEG Standard

As discussed in the introduction, a good compression method will be tailored to the specific use and type of the object being compressed. The JPEG standard is intended for use in compressing images, more specifically photorealistic ones. As such, the compression technique is based on the fact that most photographs consist primarily of gentle gradients and transitions, while rapid variations are relatively rare. With what we have just learned about the discrete cosine transform and the coefficients α_{kl}, it seems natural to let the low-frequency components (with small l and k) play a large role, while letting high-frequency components (with l and k near N) play a small role. The following rule serves as a guide: all loss of information that is imperceptible to the human visual system (eyes and brain) is acceptable.

The compression algorithm can be broken down into the following major steps:

- translation of the image function,
- application of the discrete cosine transform to each 8×8 block,
- quantization of the transformed coefficients,

- zigzag ordering and encoding of the quantized coefficients.

We will describe each of these steps as applied to the image of a cat from Figure 12.1. This photo was taken by a digital camera that natively compressed the image in JPEG format. A 640×640 crop of the image was taken and subsequently converted to grayscale, with each pixel taking an integer value between 0 and 255. Recall that each pixel requires one byte of raw storage and therefore that the image requires 409,600 B = 409.6 KB = 0.4096 MB to store uncompressed.

Translation of the image function. The first step is the *translation* of the values of f by the quantity 2^{b-1}, where b is the number of bits (or *bit depth*) used to represent each pixel. In our case we are using $b = 8$, and we therefore subtract $2^{b-1} = 2^7 = 128$ from each pixel. This first step produces a function $\tilde{f}$ whose values are in the interval $[-2^{b-1}, 2^{b-1} - 1]$, which is (nearly) symmetric with respect to the origin, like the range of the cosine functions that form the basis matrices A_{kl}. We will follow the details of the algorithm on the 8×8 block shown in Table 12.2. The values of the translated function $\tilde{f}_{ij} = f_{ij} - 128$ are shown in Table 12.4, while the original values of the function f may be found in Table 12.2.

-88	65	-39	-91	81	108	-87	-114
-26	37	-92	22	119	-24	-121	-109
29	-36	-40	123	28	-125	-108	-93
25	-53	92	65	-99	-115	-94	-106
-12	45	112	-74	-117	-90	-108	-109
34	127	-19	-119	-102	-106	-108	-99
109	54	-123	-100	-108	-113	-100	-108
94	-95	-120	-105	-104	-99	-105	-105

Table 12.4. The 64 values of the function $\tilde{f}_{ij} = f_{ij} - 128$.

Discrete cosine transformation of each 8×8 block. The second step consists in partitioning the image into nonoverlapping blocks of 8×8 pixels. (If the image width is not a multiple of 8, then columns are added to the right until it is. The pixels in these additional columns are assigned the same gray tone as the rightmost pixel in each row of the original image. A similar treatment is applied to the bottom of the picture if the height is not a multiple of 8.) After *partitioning* the image into 8×8 blocks the *discrete cosine transform* is applied to each block. The result of this second step as applied to $\tilde{f}$ is given in Table 12.5. If we compare these coefficients to the α_{kl} of f shown in Table 12.3, we see that only the coefficient α_{00} has changed. This is no coincidence and is a direct result of the fact that $\tilde{f}$ is obtained from f by a translation. Exercise 11 (b) investigates why this happens.

−342.38	351.77	−8.671	54.194	27.63	−55.11	−23.87	−15.74
144.58	−94.65	−264.52	5.864	7.660	−89.93	−24.28	−12.13
−31.78	−109.77	9.861	216.16	29.88	−108.14	−36.07	−24.40
23.34	12.04	53.83	21.91	−203.72	−167.39	0.197	0.389
−18.13	−40.35	−19.88	−35.83	−96.63	47.27	119.58	36.12
11.26	9.743	24.22	−0.618	0.0879	47.44	−0.0967	−23.99
0.0393	−12.14	0.182	−11.78	−0.0625	0.540	0.139	0.197
0.572	−0.361	0.138	−0.547	−0.520	−0.268	−0.565	0.305

Table 12.5. The 64 coefficients α_{kl} of the function $\tilde{f}$.

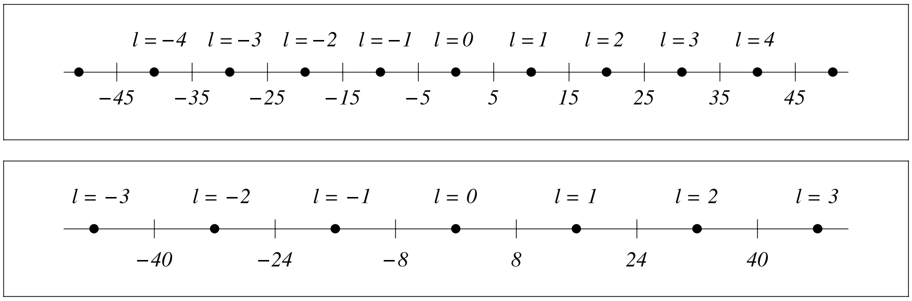

Fig. 12.7. The discrete scales used to measure α_{00} (top) and both α_{01} and α_{10} (bottom).

Quantization. The third step is called *quantization*: it consists in transforming the real-valued coefficients α_{kl} into integers ℓ_{kl}. The integer ℓ_{kl} is obtained from α_{kl} and q_{kl} by the formula

$$\ell_{kl} = \left[\frac{\alpha_{kl}}{q_{kl}} + \frac{1}{2}\right], \tag{12.16}$$

where $[x]$ is the integer part of x.

We explain the origins of this formula. Since the set of real numbers that can be represented on a computer is finite, the mathematical concept of the real line is not natural on computers. These numbers must be discretized, but must it be to the full precision that the computer is capable of representing? Could we not discretize them at a coarser scale? The JPEG standard gives a large amount of flexibility at this step: each coefficient α_{kl} is discretized with an individually chosen quantization step. The size of the step is encoded in the *quantization table*, which is fixed across all 8×8 blocks in a single image. The quantization table that we will use is shown in Table 12.6. For this table the step size for α_{00} will be 10, while already for α_{01} and α_{10} it will be 16. Figure 12.7 shows the effects of these step sizes for these three coefficients. Observe that all α_{00} from 5 up to but not including 15 will be mapped to the value $\ell_{00} = 1$; in

10	16	22	28	34	40	46	52
16	22	28	34	40	46	52	58
22	28	34	40	46	52	58	64
28	34	40	46	52	58	64	70
34	40	46	52	58	64	70	76
40	46	52	58	64	70	76	82
46	52	58	64	70	76	82	88
52	58	64	70	76	82	88	94

Table 12.6. The quantization table q_{kl} used in this example.

fact, from

$$\ell_{00}(5) = \left[\frac{5}{10} + \frac{1}{2}\right] = [1] = 1$$

and

$$\ell_{00}(15-\epsilon) = \left[\frac{15-\epsilon}{10} + \frac{1}{2}\right] = \left[2 - \frac{\epsilon}{10}\right] = 1$$

for an arbitrarily small positive number ϵ. Figure 12.7 shows the window of values that are mapped to the same quantized coefficient, each delimited by a small vertical bar. Any values of α_{kl} between two numbers below the axis will share the same ℓ at the moment of reconstruction, the ℓ noted above the central dot. These dots indicate the middle of each region, and the value $\ell_{kl} \times q_{kl}$ will be assigned to the coefficient when they are uncompressed. The fraction $\frac{1}{2}$ in (12.16) ensures that $\ell_{kl} \times q_{kl}$ falls in the middle of each window. The second axis of Figure 12.7 depicts the situation for α_{01} and α_{10}, whose quantification factor is larger, namely $q_{01} = q_{10} = 16$. More values of α_{01} (and α_{10}) will be identified to the same ℓ_{01} (and ℓ_{10}) due to this wider window. As can be seen, the larger the value of q_{kl}, the rougher the approximation of the reconstructed α_{kl} and the more information that is lost. The largest step size in our quantification table is $q_{77} = 94$. All coefficients α_{77} whose values lie in the range $[-47, 47)$ will map to the value $\ell_{77} = 0$. The precise value of the original coefficient in this interval will be irrevocably lost during the compression process.

Having chosen the quantization table shown in Table 12.6, we can quantifiy the transform coefficients of the original block f; they are shown in Table 12.7.

Most digital cameras offer a way to save images at various quality levels (basic, normal, and fine, for example). Most software packages for manipulating digital images offer similar functionality. Once a given quality level has been chosen, the image is compressed using a quantization table that has been predetermined by the makers of the hardware or software. The same quantization table is used for *all* 8×8 blocks

-34	22	0	2	1	-1	-1	0
9	-4	-9	0	0	-2	0	0
-1	-4	0	5	1	-2	-1	0
1	0	1	0	-4	-3	0	0
-1	-1	0	-1	-2	1	2	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Table 12.7. The quantization ℓ_{kl} of the transformed coefficients α_{kl}.

in the image. It is transmitted once from the header of the JPEG file, followed by the transformed, quantized, and compressed block coefficients. Even though the JPEG standard suggests a family of quantization tables, any one may be used. As such, the quantization table offers a large amount of flexibility to the end user.

Zigzag ordering and encoding. The last step of the compression algorithm is the *encoding* of the table of quantized coefficients ℓ_{kl}. We will not delve too far into the details of this step. We will say only that the coefficient ℓ_{00} is encoded slightly differently from the rest and that the encoding uses the ideas discussed in the introduction: the values of ℓ_{kl} occurring more frequently are assigned shorter code words and vice versa. What are the most likely values? The JPEG standard prefers coefficients with a small absolute value: the smaller $|\ell_{kl}|$, the smaller the code word for ℓ_{kl}. Is it surprising that many coefficients ℓ_{kl} are nearly zero-valued? No, it is not if we recall that the α_{kl} (and hence the ℓ_{kl}) typically measure changes that are relatively small in scope with respect to the actual size of the image.

Thanks to the quantization step, many ℓ_{kl} with large k and l are zero-valued. The encoding makes use of this fact by ordering the coefficients such that long strings of zero-valued coefficients are more likely. The precise ordering defined by the JPEG standard is shown in Figure 12.8: $\ell_{01}, \ell_{10}, \ell_{20}, \ell_{11}, \ell_{02}, \ell_{03}, \dots$. Given that most of the nonzero coefficients tend to be clustered in the upper left corner, it often happens that the coefficients ordered in this manner are terminated by a long run of zero values. Rather than encoding each of these zero values, the encoder sends a single special code word indicating the "end of block." When the decoder encounters this symbol it knows that the rest of the 64 symbols are to filled in with zeros. Looking at Table 12.7, note that $\ell_{46} = 2$ is the last nonzero coefficient in the proposed zigzag ordering. The eleven remaining coefficients $(\ell_{37}, \ell_{47}, \ell_{56}, \ell_{65}, \ell_{74}, \ell_{75}, \ell_{66}, \ell_{57}, \ell_{67}, \ell_{76}, \ell_{77})$ are all zero-valued and will not be explicitly transmitted. As we will see in the example of the image of the cat, this provides an enormous gain to the compression ratio.

Reconstruction. A computer can quickly reconstruct a photo from the information in a JPEG file. The quantification table is first read from the file header. Then the

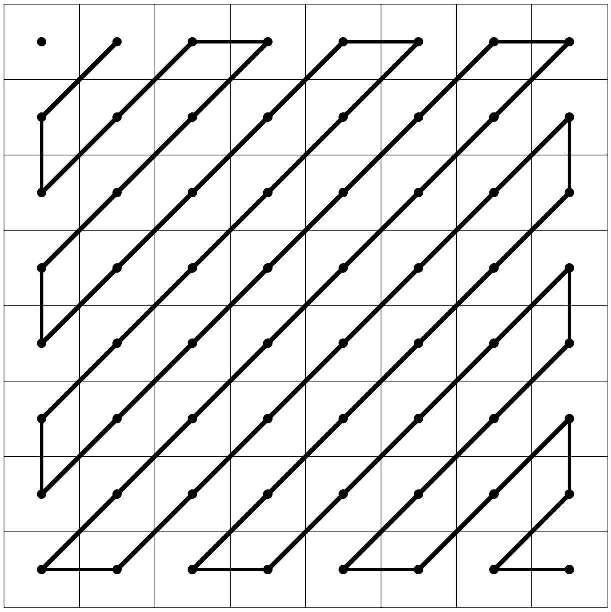

Fig. 12.8. The order in which the coefficients ℓ_{kl} are transmitted: $\ell_{01}, \ell_{1,0}, \ldots, \ell_{77}$.

following steps are performed for each 8×8 block: the information for a block is read until the "end of block" signal is encountered. If fewer than 64 coefficents were read, the missing ones are set to zero. The computer then multiplies each ℓ_{kl} by the corresponding q_{kl}. The coefficient $\beta_{kl} = \ell_{kl} \times q_{kl}$ is therefore chosen in the middle of the quantification window where the original α_{kl} lay. The inverse of the discrete cosine transformation (12.10) is then applied to the β's to get the new gray tones $\bar{f}$:

$$\bar{f} = C^t \beta C.$$

After correcting for the translation of the original image, the gray tones for this 8×8 block are ready to be shown on screen.

Figure 12.9 shows the visual results of JPEG compression, applied to the entire image as described in this section. Recall that the original photo contains 640×640 pixels and therefore $80 \times 80 = 6400$ blocks of 8×8 pixels. The four steps (translation, transformation, quantization, and encoding) are thus performed 6400 times. The left column of Figure 12.9 contains the original image plus two successive closeups.[5] The right column contains the same image after being JPEG compressed and decompressed using the quantization table of Table 12.6.

[5]Recall that the original photo was obtained from a digital camera that itself stores the image in JPEG form.

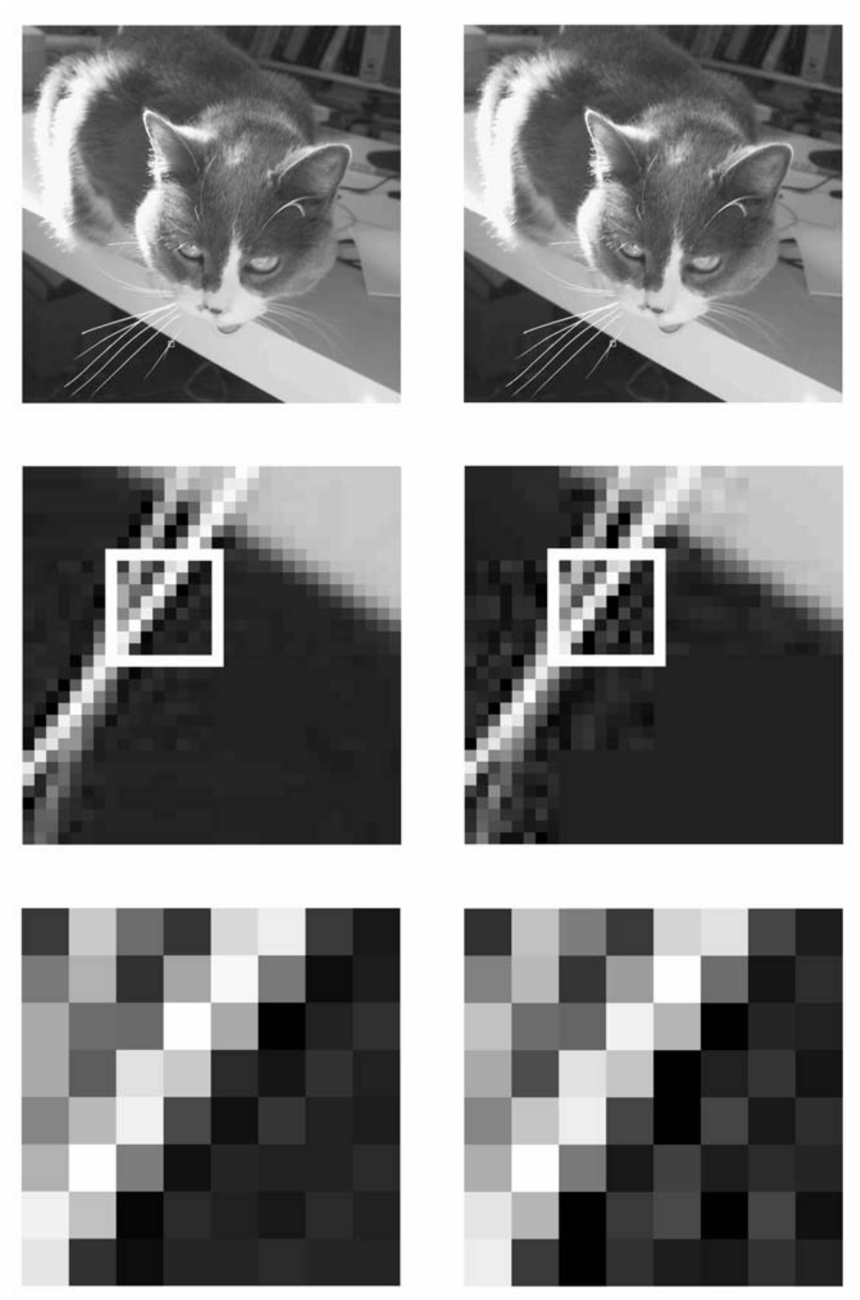

Fig. 12.9. The three images at the left are the same as those of Figure 12.1. Those at the right have been obtained from this image after being heavily JPEG compressed. The middle blocks are 32×32 pixels, while those at the bottom are 8×8 pixels.

The 8×8 block containing the crossing of two whiskers has been chosen because it is a block with high contrast. These are the types of blocks that are the least well compressed by the JPEG standard. By comparing the closeups we can see the effect of the aggressive compression. Close to the border between the highly contrasting regions the effect is most noticeable. Since this block contains high-contrast quickly varying data, we would have had to store the coefficients α_{kl} with more precision in order to reproduce them clearly. The aggressive zeroing of many of these coefficients in the quantization step has introduced a certain "noise" close to the whiskers. Note that a certain amount of noise was already present in this region in the original photograph, a clear sign that the camera was using JPEG compression. Another clear sign that JPEG compression has been used is the often visible boundaries of 8×8 blocks, specifically blocks containing high contrast next to smooth blocks, as is the case in the region of the whiskers. Notice the 8×8 block second from the bottom and third from the left of the 32×32 blocks in Figure 12.9. This block is completely "under the table" and has been compressed to a uniform gray. As such, it is not surprising that after quantization it contains only two nonzero coefficients (ℓ_{00} and ℓ_{10}). The encoding of this block omits 62 coefficients, and the compression is very good!

Is this block the rule or the exception? There are $640 \times 640 = 409{,}600$ pixels in the entire image. After transforming and quantizing these coefficients, the image is encoded by a series of 409,600 coefficients ℓ_{kl}. By ordering them in zigzag order and omitting the trailing runs of zeros, we are able to avoid storing over 352,000 zeros, roughly $\frac{7}{8}$ of the coefficients! It is not surprising that the compression achieved by the JPEG standard is so good.[6]

The ultimate test is the comparison of the two images with the naked eye. It is up to the user to judge whether the compression (in this case, the zeroing of roughly $\frac{7}{8}$ of the Fourier coefficients α_{kl}) has damaged the photograph. It is important to note that this comparison should be performed under the same conditions in which the compressed photograph will be used. Recall the example of the digitized works from the Louvre. If the image is going to be looked at using a low-resolution screen, then the compression can be relatively aggressive. However, if the image is to be closely studied by art historians, is to be printed at high resolution, or is to be viewed through software that allows zooming in, then a higher resolution and a less-aggressive compression should be used.

The JPEG standard offers an enormous amount of flexibility through its quantization tables. In certain cases we can imagine that using even higher values in this table will lead to better compression and acceptable quality. However, the weaknesses of the JPEG standard are made apparent in areas of high contrast and detail, especially when the quantization table contains overly large values. This is why the JPEG standard performs so poorly at compressing line art and cartoons, which consist largely of black lines on a white background. These lines become marred (with a characteristic JPEG

[6]Through careful choice of the quantization table this photo can be compressed to less than 30 KB in size (compared to 410 KB uncompressed) without the degradation being intolerable.

"speckle") after aggressive compression. It would be equally inappropriate to take a picture of a page of text and compress it using the JPEG standard; the letters are in high contrast with the page and would become blurred. The JPEG standard was created with the goal of compressing photographs and photorealistic images and it excels at this task.

What about color images? It is well known that colors can be described using three dimensions. For example, the color of a pixel on a computer screen is normally described as a ratio of the three (additive) primary colors: red, green, and blue. The JPEG standard uses a different set of coordinates (or *color space*). It is based on recommendations made by the *Commission internationale de l'éclairage* (International Commission on Illumination), which in the 1930s developed the first standards in this domain. The three dimensions of this color space are separated, leading to three independent images. These images, each corresponding to one coordinate, are then individually treated in the same manner as discussed in this chapter for gray tones. (For those who want to learn more, the book [2] contains a self-contained description of the standard with enough information to fully implement the standard, a discussion of the science underlying the various mathematical tools used in it, and the necessary knowledge on the human visual system. References [3, 4] are good entry points in the field of data compression.)

12.6 Exercises

1. **(a)** Verify that if $x \in [-1,1] \subset \mathbb{R}$, then $\mathrm{aff}_1(x) = 255(x+1)/2$ is an element of $[0,255]$.
(b) Is aff_1 the ideal transformation? For which x will $\mathrm{aff}_1(x) = 255$? Can you propose a function aff' such that all integers in $\{0,1,2,\dots,255\}$ will be images of equal-length subintervals of $[-1,1]$?
(c) Give the inverse of aff_1. The function aff' cannot have an inverse. Why? Despite this, can you propose a rule that would allow you to construct a function g starting from a function f as in Section 12.3?

2. **(a)** Verify that the four vectors A_{00}, A_{01}, A_{10}, and A_{11} of (12.3) (expressed in the usual basis $\mathcal{B}$) are orthonormal, that is, they have length 1 and are pairwise orthogonal.
(b) Let v be the vector whose coefficients in the basis $\mathcal{B}$ are

$$[v]_{\mathcal{B}} = \begin{pmatrix} -\frac{3}{8} \\ \frac{5}{8} \\ \frac{1}{2} \\ -\frac{1}{2} \end{pmatrix}.$$

Give the coefficients of this vector in the basis $\mathcal{B}' = \{A_{00}, A_{01}, A_{10}, A_{11}\}$. What is the largest coefficient of $[v]_{\mathcal{B}'}$ in terms of absolute value? Could you have guessed which one it was going to be without explicitly calculating them? How?

3. **(a)** Show that the $N \times N$ matrix C used in the discrete cosine transform for $N = 4$ is given by

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \gamma & \delta & -\delta & -\gamma \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \delta & -\gamma & \gamma & -\delta \end{pmatrix}.$$

Express the two unknowns γ and δ in terms of the cosine function.
(b) Using the trigonometric identity $\cos 2\theta = 2\cos^2\theta - 1$, explicitly give the numbers γ and δ. (Here "explicitly" means as an algebraic expression with integer numbers and radicals *but* without the cosine function.) Using these expressions, show that the second line of C represents a vector with unit norm as is required by the orthogonality of C.

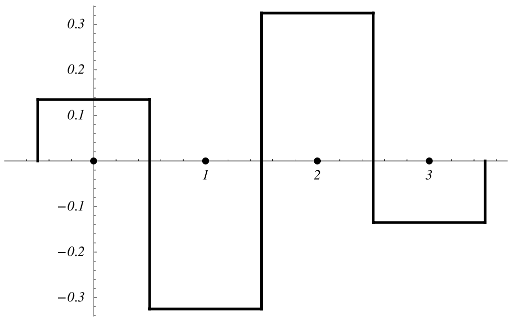

Fig. 12.10. The discrete function g of Exercise 4 (b).

4. **(a)** The discrete cosine transformation allows the expression of discrete functions $g : \{0, \ldots, N-1\} \to \mathbb{R}$ (given by $g(i) = g_i$) as linear combinations of the N discrete basis vectors C_k, where $C_k(i) = (C_k)_i = c_{ki}, k = 0, 1, 2, \ldots, N-1$. This transformation expresses g in the form $g = \sum_{k=0}^{N-1} \beta_k C_k$, which yields

$$g_i = \sum_{k=0}^{N-1} \beta_k (C_k)_i.$$

For $N = 4$, represent the function $(C_2)_i$ by a histogram. (This exercise reuses results from Exercise 3, but the reader is not required to have completed that exercise.)
(b) Knowing that the numeric values of γ and δ of the previous exercise are roughly 0.65 and 0.27 respectively, what will be the coefficient β_k with the largest magnitude for the function g represented in Figure 12.10?

5. Complete the calculation of (12.14) for the subcase in which N is odd.

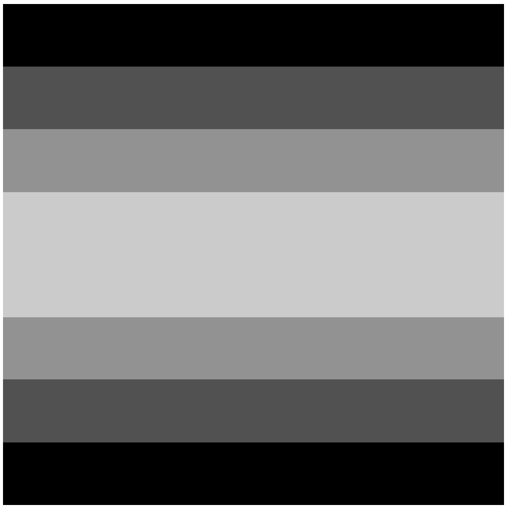

Fig. 12.11. The function f of Exercise 6.

6. A function

$$f : \{0,1,2,3,4,5,6,7\} \times \{0,1,2,3,4,5,6,7\} \to \{0,1,2,\ldots,255\}$$

is represented graphically by the gray tones of Figure 12.11. The values f_{ij} are constant along a given row; in other words, $f_{ij} = f_{ik}$ for all $j, k \in \{0,1,2,\ldots,7\}$.
(a) If $f_{0j} = 0, f_{1j} = 64, f_{2j} = 128, f_{3j} = 192, f_{4j} = 192, f_{5j} = 128, f_{6j} = 64, f_{7j} = 0$ for all j, calculate α_{00} as defined by the JPEG standard, but without doing the translation of f as described in the first step of Section 12.5.
(b) If the discrete cosine transform is carried out as suggested by the JPEG standard, several of the coefficients α_{kl} will be zero-valued. Determine which elements of α_{kl} will be zero-valued and explain why.

7. Let C be the matrix representing the discrete cosine transform. Its elements $[C]_{ij} = c_{ij}, 0 \le i,j \le N-1$, are given by (12.6). Let N be even. Show that each of the elements of rows i of C where i is odd is one of the following N values:

$$\pm\sqrt{\frac{2}{N}}\cos\frac{k\pi}{2N}, \qquad \text{with } k \in \{1,3,5,\ldots,N-1\}.$$

8. Figure 12.12 displays an 8×8 block of gray tones. Which coefficient α_{ij} will have the largest magnitude (ignoring α_{00})? What will its sign be?

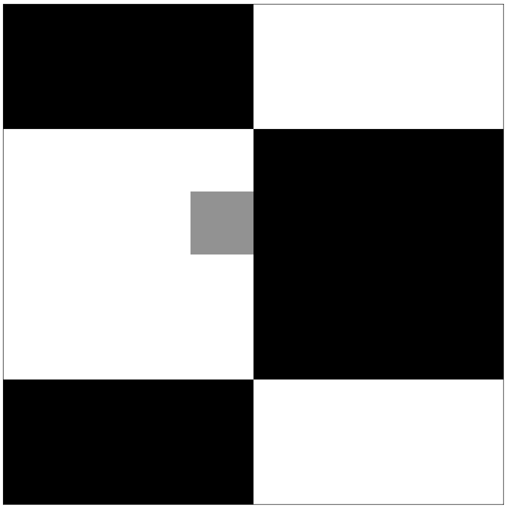

Fig. 12.12. An 8×8 block of gray tones for Exercise 8.

9. With the rising popularity of digital photography, programs allowing for the manipulation and retouching of photographs have become increasingly popular. Among other things, they allow images to be reframed (or *cropped*) by removing rows or columns from the outer edges. If an image is JPEG compressed, explain why it is better to remove groups of rows or columns that are multiples of 8.

10. **(a)** Two copies of the same photograph are independently compressed using distinct quantization tables q_{ij} and q'_{ij}. If $q_{ij} > q'_{ij}$ for all i and j, what will be, in general, the larger file, the second or the first? Which quantization table will lead to a larger loss of quality in the photograph?

(b) If the quantization table from Table 12.6 is used and if $\alpha_{34} = 87.2$, what will be the value of ℓ_{34}? What if $\alpha_{34} = -87.2$?

(c) What is the smallest value of q_{34} that will lead to a zero-valued ℓ_{34} for the values of α_{34} in the preceding question?

(d) Does $\ell_{kj}(-\alpha_{kj}) = -\ell_{kj}(\alpha_{kj})$? Explain.

Note: Another slightly different problem is raised by technology. Suppose a photo is already in the JPEG format and is available through the Internet. If the file remains large, it could be useful to recompress the file using a more aggressive quantification table for users having slower Internet connections. The choice of the new quantification table would then depend on the speed of the connection and perhaps on the use of the photo. It turns out that the choice of this second table is delicate, since the degradation of the picture does not increase monotonically with the size of its coefficients. See, for example, [1].

11. **(a)** Calculate the difference between the α_{00} of the function f given in Table 12.2 and that of the function $\tilde{f}$ obtained through translation.
(b) Show that a translation of f by any constant (for example 128) changes only the coefficient α_{00}.
(c) Using the definition of the discrete cosine transform, predict the difference between the two coefficients α_{00} calculated in (a).
(d) Show that α_{00} is N times the average gray tone of the block.

12. Let g be a step function representing a checkerboard: the upper left corner $(0,0)$ has value $+1$, and the rest of the squares are filled in such a way that they have the opposite sign to their horizontal and vertical neighbors.
(a) Show that the step function g_{ij} can be described by the formula

$$g_{ij} = \sin(i+\tfrac{1}{2})\pi \cdot \sin(j+\tfrac{1}{2})\pi.$$

(b) Calculate the eight numbers

$$\lambda_i = \sum_{j=0}^{7} c_{ij} \sin(j+\tfrac{1}{2})\pi, \qquad \text{for } i = 0, \dots, 7,$$

where c_{ij} is given by (12.6). (If this exercise is taking too long to perform by hand, consider using a computer!)
(c) Calculate the coefficients β_{kl} of the checkboard function g given by $\beta_{kl} = \sum_{i,j=0}^{N-1} c_{ki}c_{lj}g_{ij}$ (calculating the values λ_i is helpful). Could you have guessed exactly which coefficients would be zero-valued? Is the position of the largest nonzero coefficient β_{kl} surprising?

References

[1] H.H. Bauschke, C.H. Hamilton, M.S. Macklem, J.S. McMichael, and N.R. Swart. Recompression of JPEG images by requantization. *IEEE Transactions on Image Processing*, 12:843–849, 2003.

[2] W.B. Pennebaker and J.L. Mitchell. *JPEG Still Image Data Compression Standard*. Springer, New York, 1996.

[3] D. Salomon. *Data Compression: The Complete Reference*. Springer, New York, 2nd edition, 2000.

[4] K. Sayood. *Introduction to Data Compression*. Morgan Kaufmann, San Francisco, 1996.