Chapter 2
Linear Filters and Frequency Analysis

This chapter reviews some classical image analysis tools: linear filtering, linear bases, frequency analysis, and space-frequency analysis. Some of the basic ideas are illustrated in Fig. 2.1. These basic methods need to be understood before the results of statistical image models can be fully appreciated. The idea of processing of different frequencies is central in the reviewed tools. Therefore, a great deal of the following material is devoted to explaining what a *frequency-based representation* of images is, and why it is relevant in image analysis.

2.1 Linear Filtering

2.1.1 Definition

Linear filtering is a fundamental image-processing method in which a *filter* is applied to an input image to produce an output image.

Figure 2.2 illustrates the way in which the filter and the input image interact to form an output image: the filter is centered at each image location (x, y), and the pixel value of the output image $O(x, y)$ is given by the linear correlation of the filter and the filter-size subarea of the image at coordinate (x, y). (Note that the word "correlation" is used here in a slightly different way than in the statistical context.) Letting $W(x, y)$ denote a filter with size $(2K + 1) \times (2K + 1)$, $I(x, y)$ the input image, and $O(x, y)$ the output image, linear filtering is given by

$$O(x, y) = \sum_{x_*=-K}^{K} \sum_{y_*=-K}^{K} W(x_*, y_*) I(x + x_*, y + y_*). \tag{2.1}$$

An example of linear filtering is shown in Fig. 2.3.

What (2.1) means is that we "slide" the filter over the whole image and compute a weighted sum of the image pixel values, separately at each pixel location.

Visual inspection of a filter alone is usually not sufficient to interpret a filtering operation. This is also the case in the example in Fig. 2.3: what does this filtering operation actually accomplish? For a complete interpretation of a filtering operation a different type of mathematical language is needed. This language utilizes a frequency-based representation of images, as explained in Sect. 2.2 below.

A. Hyvärinen, J. Hurri, P.O. Hoyer, *Natural Image Statistics,*
Computational Imaging and Vision 39,

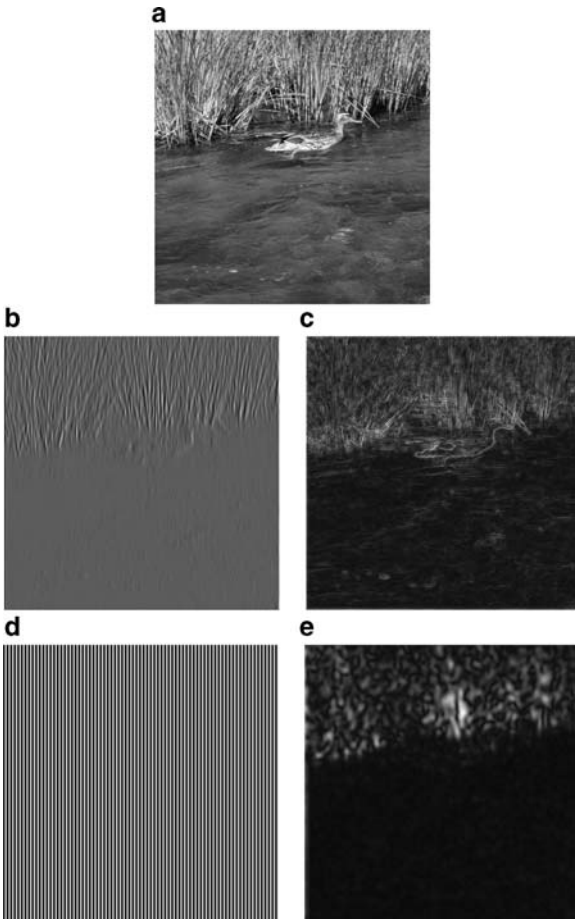

Fig. 2.1 The two classical image analysis tools reviewed in this chapter are linear filtering (**b**–**c**) and space-frequency analysis (**d**–**e**). **a** An input image. **b** An example output of linear filtering of **a**; in this case, the filter has retained medium-scaled vertical structures in the image. A more complete description of what a linear filtering operation does is provided by the frequency representation (Sect. 2.2). **c** An example of how the outputs of several linear filters can be combined in image analysis. In this case, the outputs of four filters have been processed non-linearly and added together to form an edge image: in the image, *lighter areas* correspond to image locations with a luminance edge. This kind of a result could be used, for example, as a starting point to locate objects of a certain shape. **d**–**e** An example of space-frequency analysis, where the main idea is to analyze the magnitude of a frequency **d** at different locations. The end result **e** reflects the magnitude of this frequency at different points in the input image **a**. From the point of view of image analysis, this result suggests that the *upper part* of the image is of different texture than the *lower part*

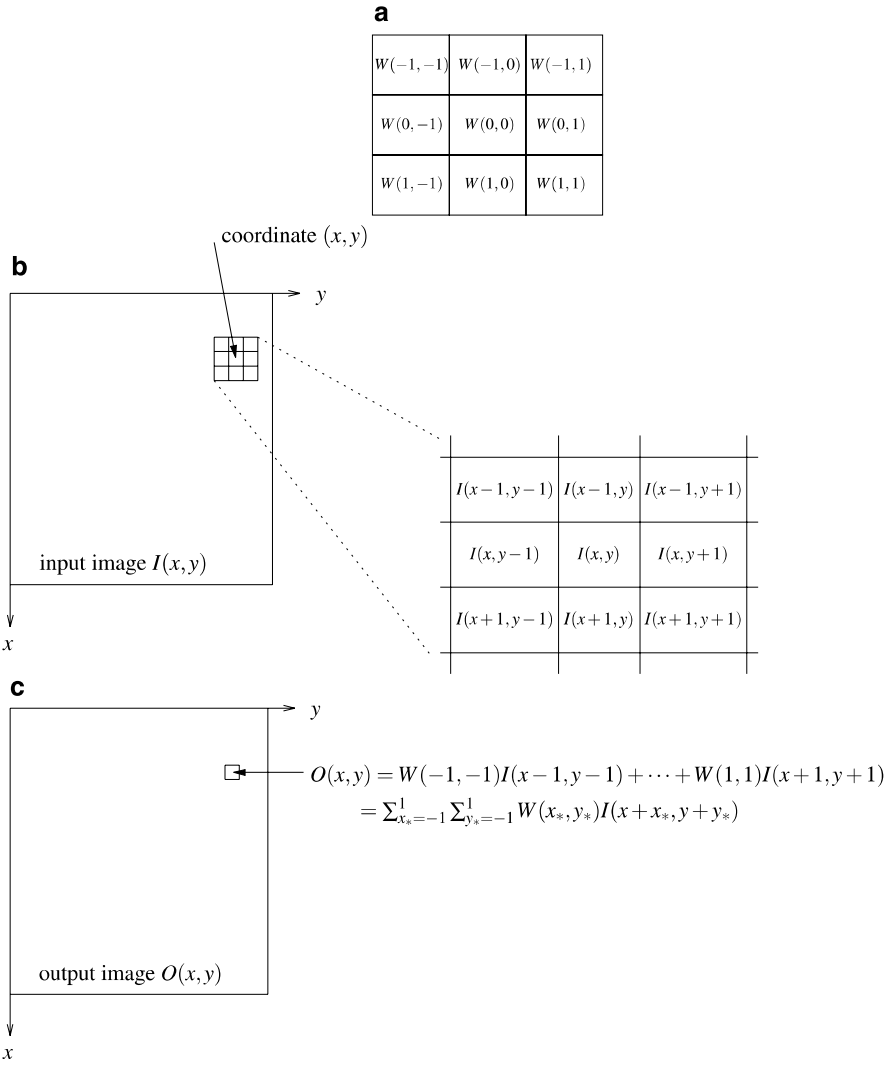

Fig. 2.2 Linear filtering is an operation that involves a filter (denoted here by $W(x, y)$) an input image (here $I(x, y)$) and yields an output image (here $O(x, y)$). The pixel value of the output image at location (x, y), that is, $O(x, y)$, is given by the linear correlation of the filter $W(x, y)$ and a filter-size subarea of the input image $I(x, y)$ centered at coordinate (x, y). **a** A 3×3 linear filter (template) $W(x, y)$. **b** An image $I(x, y)$ and a 3×3 subarea of the image centered at location (x, y). **c** The output pixel value $O(x, y)$ is obtained by taking the pixel-wise multiplication of the filter **a** and image subarea **b**, and summing this product over both x- and y-dimensions. Mathematically, $O(x, y) = \sum_{x_*} \sum_{y_*} W(x_*, y_*) I(x + x_*, y + y_*)$

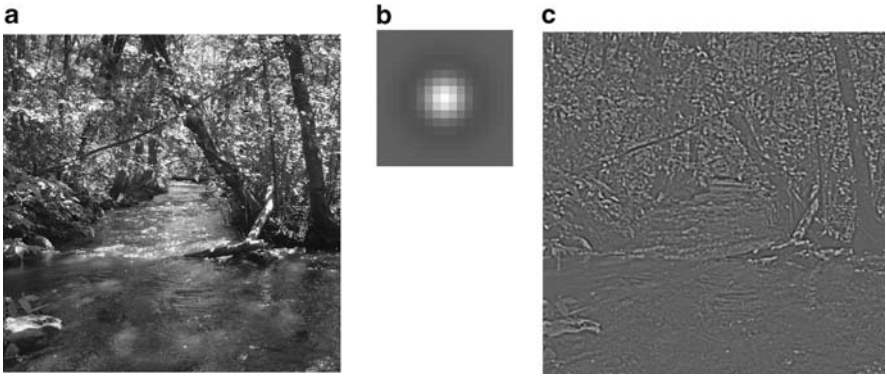

Fig. 2.3 An example of linear filtering. **a** An input image. **b** A filter. **c** An output image

2.1.2 Impulse Response and Convolution

The impulse response $H(x, y)$ is the response of a filter to an impulse

$$\delta(x, y) = \begin{cases} 1, & \text{if } x = 0 \text{ and } y = 0, \\ 0, & \text{otherwise,} \end{cases} \tag{2.2}$$

that is, to an image in which a single pixel is "on" (equal to 1) and the others are "off" (equal to 0). The impulse response characterizes the system just as well as the original filter coefficients $W(x, y)$. In fact, in frequency-based analysis of linear filtering, rather than filtering with a filter, it is customary to work with impulse responses and an operation called *convolution*. This is because the frequency-modifying properties of the linear filter can be read out directly from the frequency-based representation of the impulse response, as will be seen below. (In general, this holds for any linear shift-invariant system, which are defined in Sect. 20.1.)

Based on the definition of filtering in (2.1), it is not difficult to see that

$$H(x, y) = W(-x, -y) \tag{2.3}$$

Thus, the impulse response $H(x, y)$ is a "mirror image" of the filter weights $W(x, y)$: the relationship is simply that of a 180° rotation around the center of the filter. This is due to the change of signs of x_* and y_*; the impulse response is equal to one only if $x + x_* = 0$, which implies that only at points $x_* = -x$ we have one and elsewhere the impulse response is zero. For a filter that is symmetric with respect to this rotation, the impulse response is identical to the filter.

The convolution of two images I_1 and I_2 is defined as

$$I_1(x, y) * I_2(x, y) = \sum_{x_*=-\infty}^{\infty} \sum_{y_*=-\infty}^{\infty} I_1(x - x_*, y - y_*) I_2(x_*, y_*) \tag{2.4}$$

The only real difference to the definition of a filtering operation in (2.1) is that we have minus signs instead of plus signs. Note that convolution is symmetric in the sense that we can change the order of I_1 and I_2, since by making the change in summation index $x'_* = x - x_*$ and $y'_* = y - y_*$ we get the same formula with the roles of I_1 and I_2 interchanged (this is left as an exercise).

Therefore, we can express the filtering operation using the impulse response (which is considered just another image here) and the convolution simply as

$$O(x, y) = I(x, y) * H(x, y) \tag{2.5}$$

which is a slight modification of the original definition in (2.1). Introduction of this formula may seem like splitting hairs, but the point is that convolution is a well-known mathematical operation with interesting properties, and the impulse response is an important concept as well, so this formula shows how filtering can be interpreted using these concepts.

2.2 Frequency-Based Representation

2.2.1 Motivation

Frequency-based representation is a very useful tool in the analysis of image-processing systems. In particular, a frequency-based representation can be used to interpret what happens during linear filtering: it describes linear filtering as modification of strengths (amplitudes) and spatial locations (phases) of frequencies (sinusoidal components) that form the input image. As an example and sneak preview, Figs. 2.8a–d on page 36 show how the filtering operation of Fig. 2.3 can be interpreted as attenuation of low and high frequencies, which can be seen in the output image as disappearance of large- and fine-scale structures or, alternatively, preservation of medium-scale structures. This interpretation can be read out from Fig. 2.8d, which shows the frequency amplification map for this filter: this map, which is called the *amplitude response* of the filter, shows that both low frequencies (in the middle of the figure) and high frequencies (far from the middle) are attenuated; in the map, higher grey-scale value indicates larger amplitude response.

In what follows, we will first describe the frequency-based representation, and then demonstrate its special role in the analysis and design of linear filters.

2.2.2 Representation in One and Two Dimensions

Figure 2.4 illustrates the main idea of the frequency-based representation in the case of one-dimensional data. In the usual (spatial) representation (Fig. 2.4a), a signal is represented by a set of numbers at each point $x = 0, \ldots, N - 1$; in this example, $N = 7$. Therefore, to reconstruct the signal in Fig. 2.4a, we need 7 numbers. In the

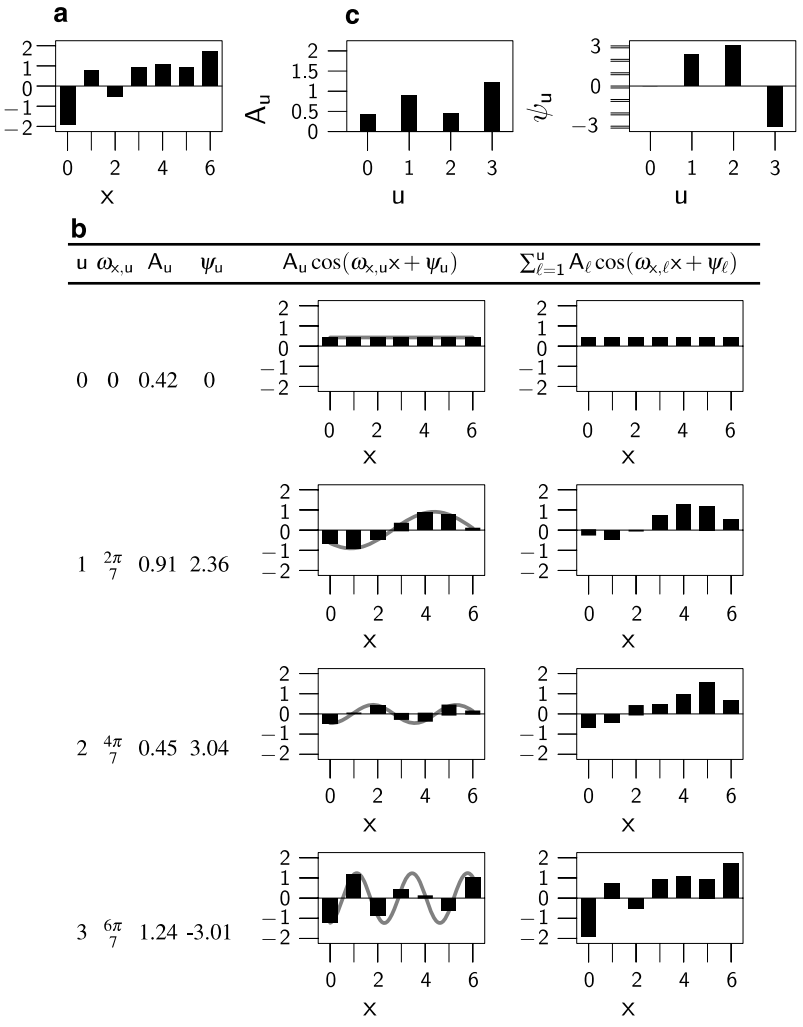

Fig. 2.4 In frequency-based representation, a signal is represented as amplitudes and phases of sinusoidal components. **a** A signal. **b** A table showing how the signal in **a** can be constructed from sinusoidal components. In the table, the number of sinusoidal components runs from 1 to 4 (frequency index u runs from 0 to 3), and the *rightmost column* shows the cumulative sum of the sinusoidal components with frequencies $\omega_{x,u}$ having amplitudes A_u and phases ϕ_u. In the *fifth column*, the *grey continuous lines* show the continuous frequency components from which the discrete versions have been sampled. Here, the frequency components are added in increasing frequency, that is, $\omega_{x,u_1} > \omega_{x,u_2}$ if $u_1 > u_2$. **c** A frequency-based representation for the signal in **a**: the signal is represented by the set of frequency amplitudes A_u, which is also called the amplitude spectrum (on the *left*), and the set of frequency phases ψ_u (on the *right*) of the corresponding frequencies $\omega_{x,u}$, $u = 0, \ldots, 3$. Note that the phase of the constant component $u = 0$ corresponding to frequency $\omega_{x,0} = 0$ is irrelevant; thus 7 numbers are needed in the frequency-based representation of the signal, just as in the usual representation **a**

frequency-based representation of this signal, we also use 7 numbers to describe the contents of the signal, but in this case the numbers have a totally different meaning: they are the *amplitudes* and *phases* of sinusoidal components, that is, parameters A and ψ of signals of the form $A\cos(\omega x + \psi)$, where ω is the frequency parameter; see Fig. 2.4c.

The theory of the *discrete Fourier transform* (treated in detail in Chap. 20) states that *any* signal of length 7 can be represented by the four amplitudes and the three phases of the four frequencies; the phase of the constant signal corresponding to $\omega = 0$ is irrelevant because the constant signal does not change when it is shifted spatially. For a family of signals of given length N, the set of frequencies ω_u, $u = 0, \ldots, U-1$, employed in the representation is fixed; in our example, these frequencies are listed in the second column of the table in Fig. 2.4b. Overall, the frequency-based representation is given by the sum

$$I(x) = \sum_{u=0}^{U-1} A_u \cos(\omega_u x + \psi_u), \tag{2.6}$$

where ω_u are the frequencies and A_u their amplitudes and ψ_u their phases.

In the case of images—that is, two-dimensional data—the sinusoidal components are of the form

$$A\cos(\omega_x x + \omega_y y + \psi), \tag{2.7}$$

where ω_x is the frequency in the x-direction and ω_y in the y-direction. In order to grasp the properties of such a component, let us define vector $\omega = (\omega_x, \omega_y)$, and denote the dot-product by $\langle . \rangle$. Then the component (2.7) can be written as

$$\begin{aligned} A\cos(\omega_x x + \omega_y y + \psi) &= A\cos\big(\langle (x, y), \omega\rangle + \psi\big) \\ &= A\cos\left(\underbrace{\|\omega\|}_{\text{“frequency”}} \underbrace{\left\langle (x, y), \frac{\omega}{\|\omega\|}\right\rangle}_{\text{projection}} + \psi\right), \end{aligned} \tag{2.8}$$

which shows that computation of the argument of the cosine function can be interpreted as a projection of coordinate vector (x, y) onto the direction of the vector ω, followed by a scaling with frequency $\|\omega\|$. Figure 2.5 illustrates this dependency of the frequency and the direction of the sinusoidal component on ω_x and ω_y.

Figure 2.5 also illustrates why it is necessary to consider both positive and negative values of either ω_x or ω_y: otherwise, it is not possible to represent all directions in the (x, y)-plane. However, there is a certain redundancy in this representation. For example, the frequency pairs $\omega = (\omega_x, \omega_y)$ and $-\omega = (-\omega_x, -\omega_y)$ represent sinusoidal components that have the same direction and frequency, because ω and $-\omega$ have the same direction and length. So, it can be seen that any half of the (ω_x, ω_y)-plane suffices to represent all directions. In practice, it has become customary to use

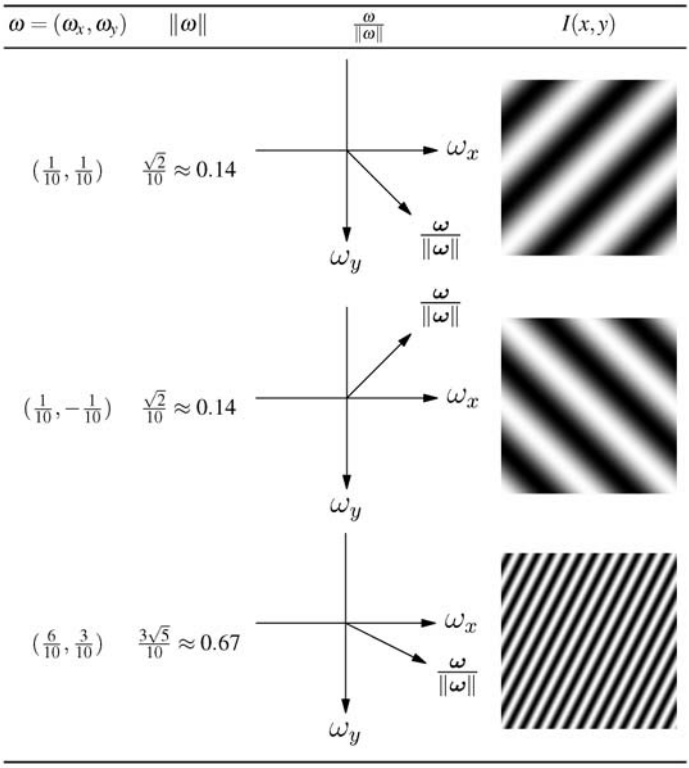

Fig. 2.5 In an equation $I(x, y) = \cos(\omega_x x + \omega_y y)$ of a two-dimensional sinusoidal image, the frequencies ω_x and ω_y determine the direction and the frequency of the component. More specifically, if $\omega = (\omega_x, \omega_y)$, then the length of ω determines the frequency of the component, and the direction of ω determines the direction of the component. This is illustrated here for three different (ω_x, ω_y) pairs; the sinusoidal components are of size 128×128 pixels. Notice that in the plots in the *third column*, ω_y runs from *top* to *bottom* because of the convention that in images the y-axis runs in this direction. See (2.8) on page 31 for details

a redundant frequency representation which employs the whole (ω_x, ω_y)-plane, that is, negative and positive parts of *both* the ω_x- and ω_y-axis.[1]

Figure 2.6 shows an example of the resulting frequency representation. Again, the theory of discrete Fourier transform (see Chap. 20) states that *any* image of size 3×3 pixels can be represented by five amplitudes and four phases, a total of

[1] In fact, because cosine is an even function—that is, $\cos(-\alpha) = \cos(\alpha)$—the frequency components corresponding to these two frequency pairs are identical when $A\cos[\langle(x, y), \omega\rangle + \psi] = A\cos[-(\langle(x, y), \omega\rangle + \psi)] = A\cos[\langle(x, y), (-\omega)\rangle + (-\psi)]$, that is, when their amplitudes are the same and their phases are negatives or each other. Therefore, when employing the whole (ω_x, ω_y)-plane, the amplitude of a frequency component are customarily split evenly among the frequency pairs ω and $-\omega$ with phases ψ and $-\psi$.

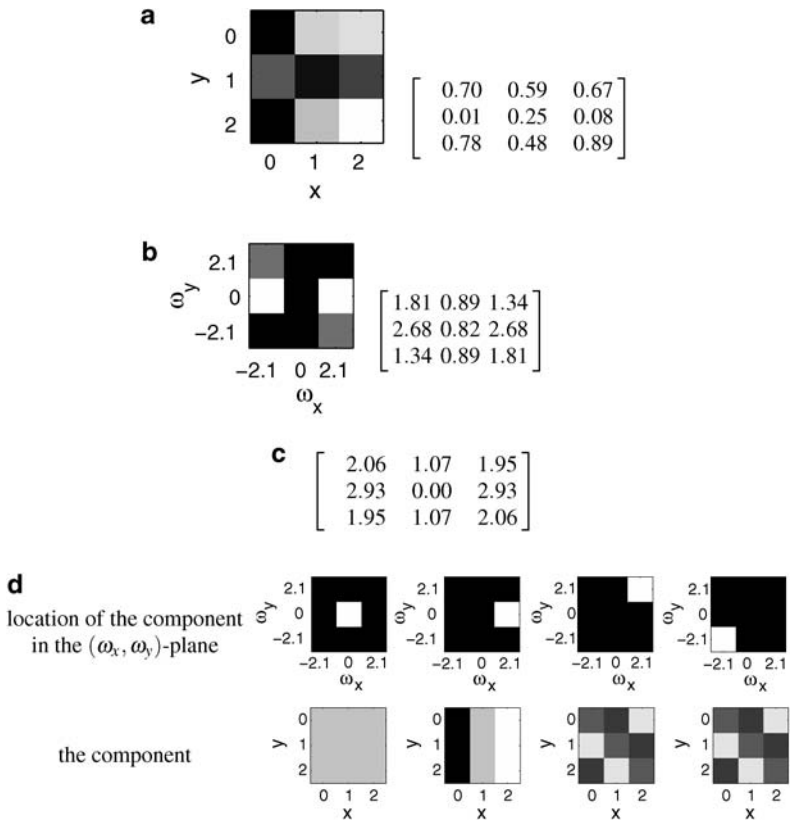

Fig. 2.6 An example of a two-dimensional frequency representation. **a** The grey-scale (*left*) and numerical (*right*) representation of an image of size 3×3 pixels. **b** Amplitude information of the frequency representation of the image in **a**: the grey-scale (*left*) and numerical (*right*) representation of the amplitudes of the different frequencies. Notice the symmetries/redundancies: the amplitude of frequency ω is the same as the amplitude of frequency $-\omega$. **c** Phase information of the frequency representation of the image in **a**; the axis of this representation are the same as in **b**. Notice the symmetries/redundancies: the phase of ω is the negative of the phase of $-\omega$. **d** Four examples of the actual sinusoidal components that make up the image in **a** in the frequency representation. In each column, the *first row* shows the location of the component in the (ω_x, ω_y)-plane, while the *second row* shows the actual component. The leftmost component is the constant component corresponding to $\omega = (0, 0)$. The second component is a horizontal frequency component. Because of the symmetry in the frequency representation, the third and the fourth components are identical. Notice that the second component (the horizontal frequency component) is stronger than the other components, which can also be seen in the amplitude representation in **b**

9 numbers as in the usual spatial representation; the phase of the constant signal corresponding to $\omega = (0, 0)$ is irrelevant as before.

Note on Terminology The square of the amplitude A^2 is also called the *Fourier energy* or *power*, and when it is computed for many different frequencies, we get what is called the *power spectrum*. The power spectrum is a classic way of charac-

terizing which frequencies are present and with which "strengths" in a given signal or image. If the original amplitudes are used, we talk about the amplitude spectrum (Fig. 2.4c). It should be noted that the terminology of frequency-based analysis is not very well standardized, so other sources may use different terminology.

2.2.3 Frequency-Based Representation and Linear Filtering

Sinusoidals also play a special role in the analysis and design of linear filters. What makes sinusoidals special is the fact that when a sinusoidal is input into a linear filter, the response is a sinusoidal *with the same frequency*, but possibly different amplitude and phase. Figure 2.7 illustrates this phenomenon. Furthermore, both the

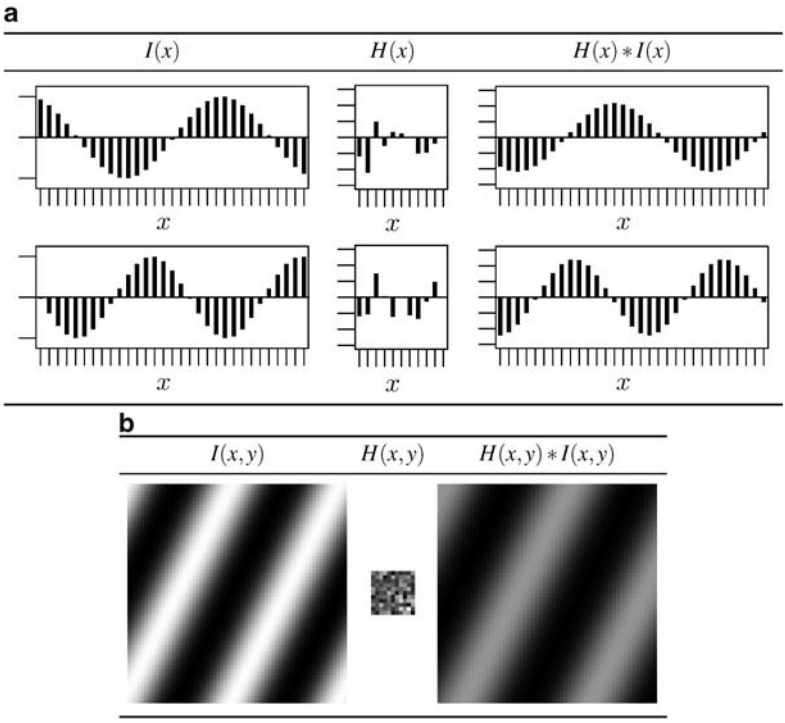

Fig. 2.7 Sinusoidal components play a special role in the analysis and design of linear systems, because if a sinusoidal is input into a linear filter, the output is a sinusoidal with the same frequency but possibly different amplitude and phase. **a** Two examples of this phenomenon are shown here in the one-dimensional case, one on each row. The *left column* shows the input signal, which is here assumed to be a sinusoidal of infinite duration. The *middle column* shows a randomly selected impulse response, and the *column on the right* the response of this linear system to the sinusoidal. Notice the different scale in the output signals, which is related to the amplitude change taking place in the filtering. **b** An illustration of the two-dimensional case, with a 64×64-pixel input *on the left*, a random 11×11-pixel impulse response *in the middle*, and the output *on the right*

amplification factor of the amplitude and the shift in the phase depend only on the frequency of the input, and not its amplitude or phase (see Chap. 20 for a detailed discussion).

For a linear filter with impulse response $H(x, y)$, let $|\tilde{H}(\omega_x, \omega_y)|$ denote the amplitude magnification factor of the system for horizontal frequency ω_x and vertical frequency ω_y, and $\angle\tilde{H}(\omega_x, \omega_y)$ denote the phase shift of the filter. Then if the input signal has frequency-based representation

$$I(x, y) = \sum_{\omega_x} \sum_{\omega_y} A_{\omega_x,\omega_y} \cos(\omega_x x + \omega_y y + \psi_{\omega_x,\omega_y}), \tag{2.9}$$

where the sum over ω_x and ω_y is here and below taken over both positive and negative frequencies, the response of the linear filter has the following frequency-based representation

$$\begin{aligned} O(x, y) &= H(x, y) * I(x, y) \\ &= \sum_{\omega_x} \sum_{\omega_y} \underbrace{|\tilde{H}(\omega_x, \omega_y)| A_{\omega_x,\omega_y}}_{\text{amplitude}} \cos\Big(\omega_x x + \omega_y y + \underbrace{\psi_{\omega_x,\omega_y} + \angle\tilde{H}(\omega_x, \omega_y)}_{\text{phase}}\Big). \end{aligned} \tag{2.10}$$

The amplitude magnification factor $|\tilde{H}(\omega_x, \omega_y)|$ is called the *amplitude response* of the linear system, while the phase shift $\angle\tilde{H}(\omega_x, \omega_y)$ is called the *phase response*. The way these quantities are determined for a linear filter is described shortly below; for now, let us just assume that they are available.

Figure 2.8 shows an example of the insight offered by the frequency representation of linear filtering (2.10). The example shows how a linear filter can be analyzed or designed by its amplitude response (the phase response is zero for all frequencies in this example). Notice that while relating the forms of the filters themselves (Figs. 2.8b and f) to the end result of the filtering is very difficult, describing what the filter does is straightforward once the frequency-based representation (Figs. 2.8d and e) is available.

How can the amplitude and phase responses of a linear system be determined? Consider a situation where we feed into the system a signal which contains a mixture of all frequencies with unit amplitudes and zero phases:

$$I(x, y) = \sum_{\omega_x} \sum_{\omega_y} \cos(\omega_x x + \omega_y y). \tag{2.11}$$

Then applying (2.10), the frequency-based representation of the output is

$$\begin{aligned} O(x, y) &= H(x, y) * I(x, y) \\ &= \sum_{\omega_x} \sum_{\omega_y} \underbrace{|\tilde{H}(\omega_x, \omega_y)|}_{\text{amplitude}} \cos\Big(\omega_x x + \omega_y y + \underbrace{\angle\tilde{H}(\omega_x, \omega_y)}_{\text{phase}}\Big). \end{aligned} \tag{2.12}$$

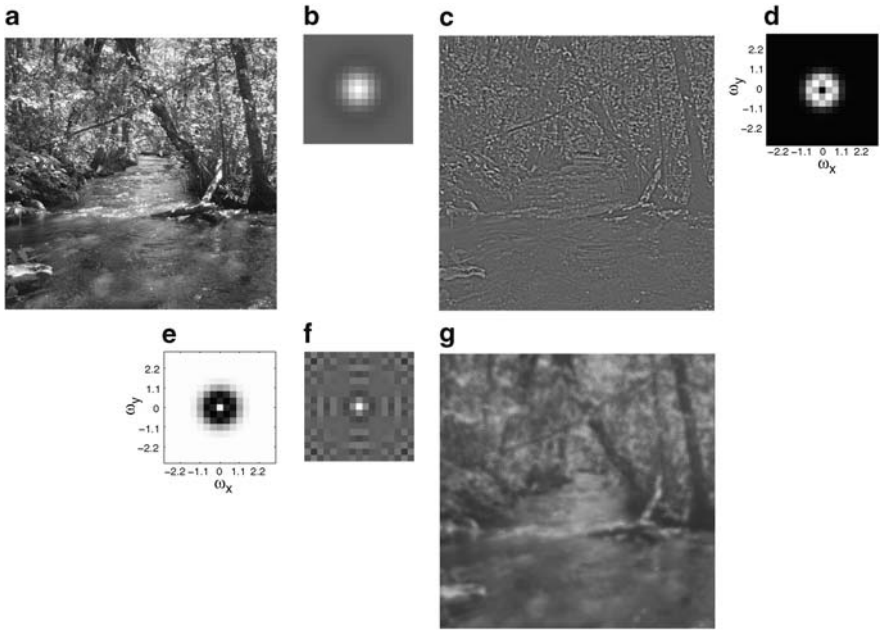

Fig. 2.8 An example of the usefulness of frequency-based representation in the analysis and design of linear filters. **a** An input image. **b** A filter of size 17×17 pixels. **c** The output of linear filtering of image **a** with filter **b**. **d** The amplitude response of the filter in **b**; in this representation dark pixels indicate amplitude response values close to zero and bright pixels values close to one. The amplitude response shows that the filter *attenuates low and high frequencies*, that is, frequencies which are either close to the origin $(\omega_u, \omega_v) = (0, 0)$ or far away from it. This can be verified in **c**, where medium-scaled structures have been preserved in the image, while details and large-scale grey-scale variations are no longer visible. The phase response of the filter is zero for all frequencies. **e** Assume that we want to design a filter that has a reverse effect than the filter shown in **b**: our new filter attenuates medium frequencies. The filter can be designed by specifying its amplitude and phase response. The amplitude response is shown here, the phase response is zero for all frequencies. **f** The filter corresponding to the frequency-based representation **e**. **g** The result obtained when the filter **f** is applied to the image in **a**. The results is as expected: the filter preserves details and large-scale grey-scale variations, while medium-scale variations are no longer visible. Notice that just by examining the filters themselves (**b** and **f**) it is difficult to say what the filters do, while this becomes straightforward once the frequency-based representations (**d** and **e**) are available

In other words, the amplitude and phase responses of the linear system can be read from the frequency-based representation of the output $O(x, y)$. What remains to be determined is what kind of a signal the signal $I(x, y)$ in (2.11) is. The theory of the Fourier transform states that the image obtained when all frequencies with identical amplitudes and zero phases are added together is an impulse. Figure 2.9 illustrates this when image size is 3×3 pixels. To summarize, when we feed an impulse into a linear filter,

- from the point of view of frequency-based description, we are giving the system an input with equal amplitudes of all frequencies at phase zero

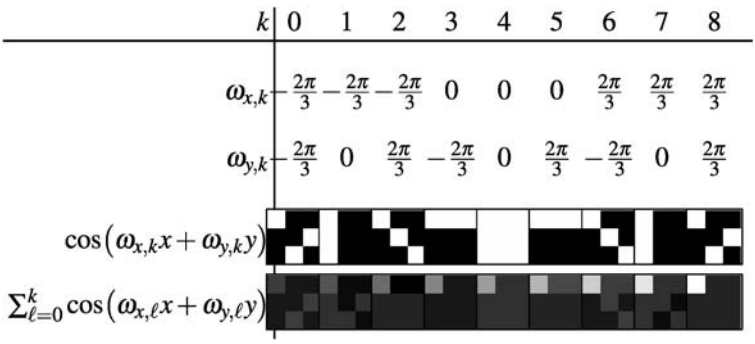

Fig. 2.9 An image containing all frequencies with unit amplitudes and zero phases is an impulse. Here, the different frequency components are being added from left to right; the right-most image is an impulse response

- the linear system modifies the amplitudes and phases of these frequencies according to the amplitude and phase response of the system
- the amplitude and phase response properties can be easily read out from the impulse response, since the amplitudes of the input were equal and phases were all zero.

In other words, the *amplitude and phase responses of a linear filter are obtained from the frequency-based representation of the impulse response: the amplitude responses are the amplitudes of the frequencies, and the phase responses are the phases of the frequencies.* An example of this principle is shown in Fig. 2.8: the amplitude response images (d) and (e) are in fact the amplitudes of the frequency-based representations of the impulse responses (b) and (f).

2.2.4 Computation and Mathematical Details

Above, we have outlined the nature of the frequency-based representation in the one- and two-dimensional case, and the usefulness of this representation in the design and analysis of linear systems. The material presented so far should therefore provide the reader with the knowledge needed to understand what the frequency-based representation is, and why it is used.

However, a number of questions have been left unanswered in the text above, including the following:

- What exactly are the values of the frequencies ω used in the frequency-based representation?
- How is the frequency-based representation computed?
- What guarantees that a frequency-based representation exists?

The set of mathematical tools used to define and analyze frequency-based representations are part of mathematics called *Fourier analysis*. In particular, *Fourier*

transforms are used to convert data and impulse responses to and from frequency-based representation. There are different types of Fourier transforms for different purposes: continuous/discrete and finite/infinite data. When working with digital images, the most important Fourier transform is the *discrete Fourier transform (DFT)*, which is particularly suited for representation of discrete and finite data in computers. The basics of DFT are described in Chap. 20. The computational implementation of DFT is usually through a particular algorithm called Fast Fourier Transform, or FFT.

The DFT has fairly abstract mathematical properties, because complex numbers are employed in the transform. The results of the DFT are, however, quite easily understood in terms of the frequency-based representation: for example, Fig. 2.8d was computed by taking the DFT of the filter in Fig. 2.8b, and then showing the magnitudes of the complex numbers of the result of the DFT.

A working knowledge of the frequency-based representation is not needed in reading this book: it is sufficient to understand what the frequency-based representation is and why it is used. If you are interested in working with frequency-based representations, then studying the DFT is critical, because the DFT has some counterintuitive properties that must be known when working with results of transforms; for example, the DFT assumes that the data (signal or image) is periodic, which causes periodicity effects when filtering is done in the frequency-based representation.

2.3 Representation Using Linear Basis

Now we consider a general and widely-used framework for image representation: a linear basis. We will also see how frequency-based representation can be seen as a special case in this framework.

2.3.1 Basic Idea

A traditional way to represent grey-scale images is the pixel-based representation, where the value of each pixel denotes the grey-scale value at that particular location in the image. For many different purposes, more convenient representations of images can be devised.

Consider the three 3×2 images f_1, f_2 and I_3 shown in Fig. 2.10. The traditional pixel-based representations of the images are

$$f_1 = \begin{bmatrix} 4\,0\,0 \\ 4\,0\,0 \end{bmatrix},$$

$$f_2 = \begin{bmatrix} 0\,0\,4 \\ 0\,0\,4 \end{bmatrix},$$

Fig. 2.10 Three different 3×2 images consisting of vertical lines: **a** f_1; **b** f_2; **c** I_3. Here, *black* denotes a grey-scale value of zero, *light grey* a grey-scale value of 4, and the *darker grey* a grey-scale value of 2

$$I_3 = \begin{bmatrix} 2\,0\,2 \\ 2\,0\,2 \end{bmatrix}.$$

These images consist of vertical lines with different grey-scale values. A compact way to describe a set of images containing only vertical lines is to define the following *basis images*

$$B_1 = \begin{bmatrix} 1\,0\,0 \\ 1\,0\,0 \end{bmatrix},$$

$$B_2 = \begin{bmatrix} 0\,1\,0 \\ 0\,1\,0 \end{bmatrix},$$

$$B_3 = \begin{bmatrix} 0\,0\,1 \\ 0\,0\,1 \end{bmatrix},$$

and then represent each image as a *weighted sum* of these basis images. For example, $f_1 = 4B_1$, $f_2 = 4B_3$ and $I_3 = 2B_1 + 2B_3$. Such a representation could convey more information about the inherent structure in these images. Also, if we had a very large set of such images, consisting of only vertical lines, and were interested in compressing our data, we could store the basis images, and then for each image just save the three coefficients of the basis images.

This simple example utilized a special property of the images f_1, f_2, and I_3, that is, the fact that in this case each image contains only vertical lines. Note that not every possible 3×2 image can be represented as a weighted sum of the basis images B_1, B_2, and B_3 (one example is an image with a single non-zero pixel at any image location). This kind of a basis is called an *under-complete* basis. Usually the word *basis* is used to refer to a *complete basis*, a basis which—when used in the form of a weighted sum—can be used to represent any image. For the set of 3×2 images, one example of a complete basis is the set of 6 images with a single one at exactly one image location (x, y). This basis is typically associated with the traditional pixel-based image representation: each pixel value denotes the coefficient of the corresponding basis image.

A particularly important case is an *orthogonal* basis. Then the coefficients in the basis simply equal the dot-products with the basis vectors. For more on bases and orthogonality, see Sect. 19.6.

The use of different bases in the representation of images has several important areas of application in image processing. Two of these were mentioned already: the description and analysis of the structure of images, and image compression. Other important applications are in the domain of image processing systems: different image representations are central in the analysis of these systems (including the analysis of the visual system), design of such systems, and in their efficient implementation.

2.3.2 Frequency-Based Representation as a Basis

Now, we consider how frequency-based representation can be rephrased as finding the coefficients in a basis. Consider the situation where we want to analyze, in a given signal, the amplitude A and phase ψ of a sinusoidal component

$$A\cos(\omega x + \psi). \tag{2.13}$$

The key point here is that instead of determining A and ψ directly, we can determine the coefficients C and S of a cosine and sine signal with (centered) phase zero:

$$C\cos(\omega x) + S\sin(\omega x). \tag{2.14}$$

To show this, we will demonstrate that there is a one-to-one correspondence between signals of the form (2.13) and of the form (2.14). First, a given sinusoidal of the form (2.13) can be represented in form (2.14) as follows:

$$\begin{aligned} A\cos(\omega x + \psi) &= A\big(\cos(\omega x)\cos\psi - \sin(\omega x)\sin\psi\big) \\ &= \underbrace{A\cos\psi}_{=C}\cos(\omega x) + \underbrace{A(-\sin\psi)}_{=S}\sin(\omega x) \\ &= C\cos(\omega x) + S\sin(\omega x). \end{aligned} \tag{2.15}$$

Conversely, if we are given a signal in form (2.14), the derivation (2.15) can be reversed (this is left as an exercise), so that we get

$$A = \sqrt{C^2 + S^2}, \tag{2.16}$$

$$\psi = -\operatorname{atan}\frac{S}{C}. \tag{2.17}$$

Thus, to analyze the amplitude and phase of frequency ω in a given signal, it suffices to find coefficients C and S in equation $C\cos(\omega x) + S\sin(\omega x)$; after the coefficients have been computed, (2.16) and (2.17) can be used to compute the amplitude and phase. In particular, the square of the amplitude ("Fourier power" or "Fourier energy") is obtained as the sum of squares of the coefficients.

The formula in (2.16) is also very interesting from a neural modelling viewpoint because it shows how to compute the amplitude using quite simple operations, since

computation of the coefficients in a linear basis is a linear operation (at least if the basis is complete). Computation of Fourier energy in a given frequency thus requires two linear operations, followed by squaring, and summing of the squares. As we will see in Chap. 3, something similar to linear Fourier operators seems to be computed in the early parts of visual processing, which makes computation of Fourier energy rather straightforward.

How are the coefficients C and S then computed? The key is orthogonality. The signals $\cos(\omega x)$ and $\sin(\omega x)$ are orthogonal, at least approximately. So, the coefficients C and S are simply obtained as the dot-products of the signal with the basis vectors given by the cos and sin functions.

In fact, Discrete Fourier Transform can be viewed as defining a basis with such cos and sin functions with many different frequencies, and those frequencies are carefully chosen so that the sinusoidals are exactly orthogonal. Then the coefficients for *all* the sin and cos functions, in the different frequencies, can be computed as the dot-products

$$\sum_x I(x)\cos(\omega x) \quad \text{and} \quad \sum_x I(x)\sin(\omega x). \tag{2.18}$$

The idea generalizes to two dimensions (images) in the same way as frequency-based analysis was shown to generalize above. More details on the DFT can be found in Chap. 20.

2.4 Space-Frequency Analysis

2.4.1 Introduction

The frequency representation utilizes global sinusoidal gratings, that is, components which span the whole image (see, e.g. Fig. 2.5). This is particularly useful for the analysis and design of linear filters. However, because of the global nature of sinusoidals, the frequency representation tells us nothing about the spatial relationship of different frequencies in an image. This is illustrated in Figs. 2.11a and b, which show an image and the amplitudes of its frequency representation. The upper part of the image contains grass, which tends to have a more vertical orientation and sharper structure than the lower part of the image. The amplitudes of the frequency representation (Fig. 2.11b) show that many horizontal or near-horizontal frequencies—that is, frequencies in the vicinity of axis $\omega_y = 0$—have a relatively large amplitude, even at fairly high frequencies. (Notice that structures with vertical lines correspond to *horizontal* frequencies.) From the amplitude spectrum there is, however, no way to tell the spatial location of these frequencies.

The spatial locations of different frequencies can contain important information about the image. In this example, if we are able to locate those areas which tend to have more horizontal frequencies, we can use that information, for example, to

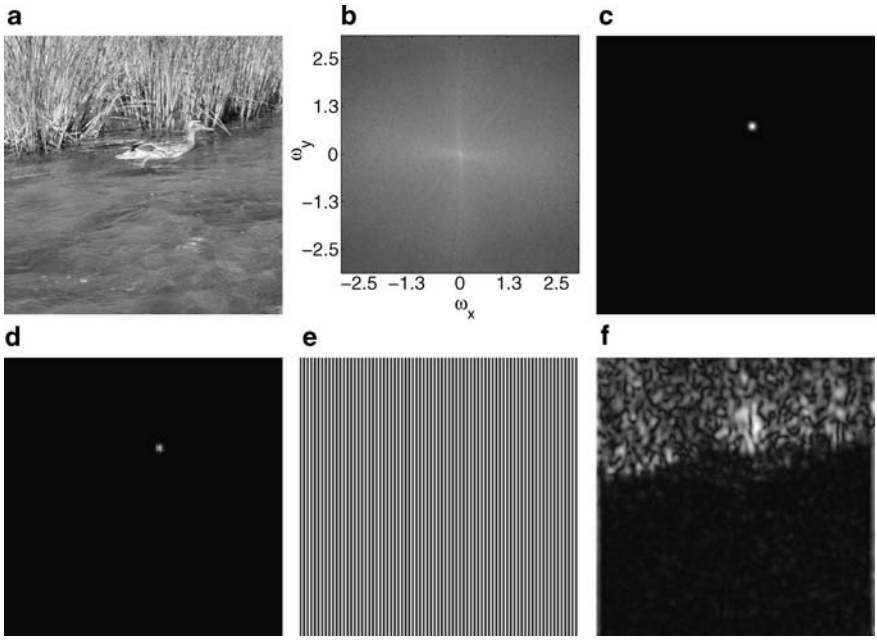

Fig. 2.11 The main idea in space-frequency analysis is to consider the amplitudes/phases of different frequencies *at different locations in an image*. **a** An image. Notice how different areas of the image differ in their frequency contents. **b** The standard frequency representation: the amplitudes of the different frequencies that make up the frequency representation of the image in **a**. Note that while this representation suggests that fairly high horizontal frequencies are present in the image, it does not convey information about the spatial location of different frequencies. For purposes of visibility, the amplitudes of different frequencies have been transformed with logarithmic mapping before display. **c** A spatially localized non-negative windowing function. **d** A localized area of the image can be obtained by multiplying the image **a** with the windowing function **c**. **e** In this example, the amplitude (strength) of this horizontal frequency is analyzed at each point in the image. **f** Applying the weighting scheme **c**–**d** at each point in the image **a**, and then analyzing the amplitude of the frequency **e** at each of these points results in this spatial map of the amplitude of the frequency. As can be seen, the frequency tends to have larger amplitudes in the upper part of the image, as can be expected

facilitate the identification of the grass area in the image. How can the spatial locations of these frequencies be found? A straightforward way to do this is to analyze the frequency contents of *limited spatial areas*. Figures 2.11c–f illustrate this idea. The original image (Fig. 2.11a) is multiplied with a non-negative *windowing function* (Fig. 2.11c) to examine the frequency contents of a spatially localized image area (Fig. 2.11d). For example, for the horizontal frequency shown in Fig. 2.11e, a spatial map of the amplitude of this frequency is shown in Fig. 2.11f; the map has been obtained by applying the weighting scheme (Figs. 2.11c and d) at *every point* of the image, and analyzing the amplitude of the frequency in the localized image area. Now, we see that in Fig. 2.11f that the different areas (grass vs. water) are clearly separated by this space-frequency analysis.

In the case of space-frequency analysis, the computational operations underlying the analysis are of great interest to us. This is because some important results obtained with statistical modelling of natural images can be interpreted as performing space-frequency analysis, so the way the analysis is computed needs to be understood to appreciate these results. Because of this connection, we need to delve a little deeper into the mathematics.

Before going into the details, we first state the main result: space-frequency analysis can be done by the method illustrated in Fig. 2.12: by filtering the image with two different localized sinusoidal filters, and computing the amplitudes and phases (in this example only the amplitudes) from the outputs of these two filters. The following section explains the mathematics behind this method.

2.4.2 Space-Frequency Analysis and Gabor Filters

Consider a situation where we want to analyze the local frequency contents of an image: we want to compute the amplitude and phase for each location (x_0, y_0). Alternatively, we could compute a set of coefficients $C(x_0, y_0)$ and $S(x_0, y_0)$ as in Sect. 2.3.2, which now are functions of the location. The analysis is made local by applying a weighting function, say $W(x, y)$, centered at (x_0, y_0) before the analysis.

We can simply modify the analysis in Sect. 2.3.2 by centering the signal f around the point (x_0, y_0) and weighting it by w before the analysis. Thus, we get a formula for the coefficient C at a given point:

$$C(x_0, y_0) \approx \sum_x \sum_y I(x, y) W(x - x_0, y - y_0) \cos\big(\omega_x (x - x_0) + \omega_y (y - y_0)\big) \tag{2.19}$$

and similarly for $S(x_0, y_0)$. Note that the order in which we multiply the three images (image f, weighting image w, sinusoidal cos) inside the sum is irrelevant. Therefore, it does not make any difference whether we apply the weighting to the image $I(x, y)$ or to the sinusoidal $\cos(\omega_x (x - x_0) + \omega_y (y - y_0))$. If we define a new weighting function

$$W_2(x, y) = W(x, y) \cos(\omega_x x + \omega_y y), \tag{2.20}$$

(2.19) becomes

$$C(x_0, y_0) \approx \sum_x \sum_y I(x, y) W_2(x - x_0, y - y_0). \tag{2.21}$$

Equations (2.20) and (2.21) show that computation of coefficients $C(x_0, y_0)$ can be approximated by filtering the image with a filter that is the product of a sinusoidal $\cos(\omega_x x + \omega_y y)$ and the weighting window. Similar analysis applies to coefficients $S(x_0, y_0)$, except that in that case the sinusoidal is $\sin(\omega_x x + \omega_y y)$. Because the magnitude of the weighting function $W(x, y)$ typically falls off quite fast from the

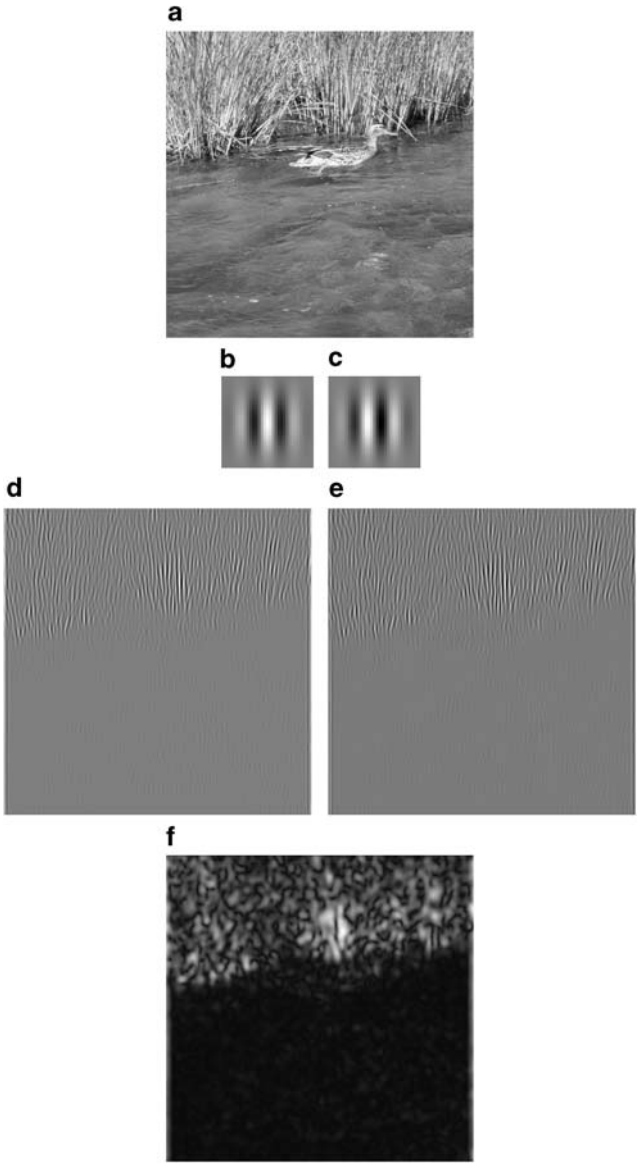

Fig. 2.12 An example showing how the spatial map of amplitudes of Fig. 2.11f on page 42 was computed. **a** The analyzed image. **b** The spatial filter (called a Gabor filter) obtained by multiplying $\cos(\omega_x x + \omega_y y)$ with the weighting window $W(x, y)$; the filter has been truncated to a size of 33×33 pixels. **c** The spatial filter obtained by multiplying $\sin(\omega_x x + \omega_y y)$ with the weighting window $W(x, y)$. **d** Coefficients $C(x_0, y_0)$, obtained by filtering image **a** with filter **b**. **e** Coefficients $S(x_0, y_0)$, obtained by filtering image **a** with filter **c**. **f** The spatial amplitude map $A(x_0, y_0) = \sqrt{C(x_0, y_0)^2 + S(x_0, y_0)^2}$

origin $(x, y) = (0, 0)$, computational savings can be obtained by truncating the filter to zero for (x, y) far away from the origin.

To summarize the result in an example, Fig. 2.12 shows how the amplitude map of Fig. 2.11f on page 42 was computed: $C(x_0, y_0)$ and $S(x_0, y_0)$ were computed by filtering the image with weighted versions of $\cos(\omega_x x + \omega_y y)$ and $\sin(\omega_x x + \omega_y y)$, respectively, and the amplitude map $A(x_0, y_0)$ was obtained by $A(x_0, y_0) = \sqrt{C(x_0, y_0)^2 + S(x_0, y_0)^2}$.

The two filters used in the computation of $C(x_0, y_0)$ and $S(x_0, y_0)$ are often called a *quadrature-phase pair*. This is because $\sin(x + \frac{\pi}{2}) = \cos(x)$, so the two filters are $W(x, y)\cos(\omega_x x + \omega_y y)$ and $W(x, y)\cos(\omega_x x + \omega_y y + \frac{\pi}{2})$, that is, they are otherwise identical expect for a a phase difference of one quarter of a whole cycle: $\frac{2\pi}{4} = \frac{\pi}{2}$.

When the weighting function is a Gaussian window, which in the one-dimensional case is of the form

$$W_\sigma(x) = \frac{1}{d} e^{-\frac{x^2}{\sigma^2}}, \tag{2.22}$$

the resulting filter is called a *Gabor filter*; parameter σ determines the width of the window, and the scaling constant d is typically chosen so that $\sum_x W_\sigma(x) = 1$. Overall, a one-dimensional Gabor function

$$W_{\sigma,\omega,\psi}(x) = W_\sigma(x)\cos(\omega x + \psi) \tag{2.23}$$

has three parameters, width σ, frequency ω and phase ψ. One-dimensional Gabor functions are illustrated in Fig. 2.13.

In the two-dimensional case, a Gabor filter has a few additional parameters that control the two-dimensional shape and orientation of the filter. When the sinusoidal in the filter has vertical orientation the filter is given by the following equation

$$W_{\sigma_x,\sigma_y,\omega,\psi}(x, y) = \frac{1}{d} e^{-(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2})} \cos(\omega x + \psi); \tag{2.24}$$

here, σ_x and σ_y control the width of the weighting window in the x- and y-directions, respectively. A Gabor-filter with orientation α can be obtained by rotating the original (x, y) coordinate system by $-\alpha$ to yield a new coordinate system (x_*, y_*) (this rotation is equivalent to the rotation of the filter itself by α). The equations that relate the two coordinate systems are

$$x = x_* \cos\alpha + y_* \sin\alpha, \tag{2.25}$$

$$y = -x_* \sin\alpha + y_* \cos\alpha. \tag{2.26}$$

Substituting (2.25) and (2.26) into (2.24) gives the final form (not shown here).

Examples of two-dimensional Gabor functions were already given in Figs. 2.12b–c; two more will be given in the next section, Fig. 2.15.

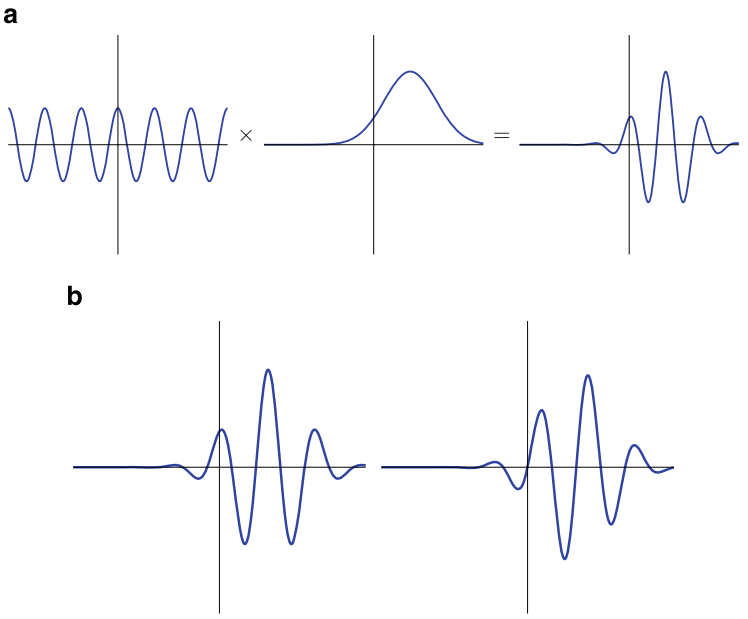

Fig. 2.13 Illustration of one-dimensional Gabor functions. **a** Construction of the function by multiplication of the envelope with a sinusoidal function. **b** Two Gabor functions in quadrature phase.

2.4.3 Spatial Localization vs. Spectral Accuracy

Above, we have outlined a scheme where the frequency contents of an image at a certain point is analyzed by first multiplying the image with a localized weighting window, and then analyzing the frequency contents of the weighted image. How accurate is this procedure, that is, how well can it capture the localized frequency structure?

The answer is that there is a trade-off between spatial localization and spectral (frequency) accuracy because the use of a weighting window changes the spectral contents. Figure 2.14 illustrates this phenomenon by showing how the results of space-frequency analysis of a pure sinusoidal $I(x) = \cos(\frac{1}{2}x)$ depend on the degree of spatial localization. The mathematical theory behind this phenomenon goes under the name *time-bandwidth product theorem* in signal processing, or *uncertainty principle* in physics. These theories state that there is a lower bound on the product of the spread of the energy in the spatial domain and the frequency domain. See the References at the end of this chapter for more details.

With images, the extra dimensions introduce another factor: uncertainty about orientation. This parameter behaves just like frequency and location in the sense that if we want to have a filter which is very localized in orientation, we have to give up localization in the other parameters. This is illustrated in Fig. 2.15, in which we see that a basic Gabor which is very localized in space (a) responds to a wider range of different orientations than the Gabor in (b). The Gabor in (b) has has been

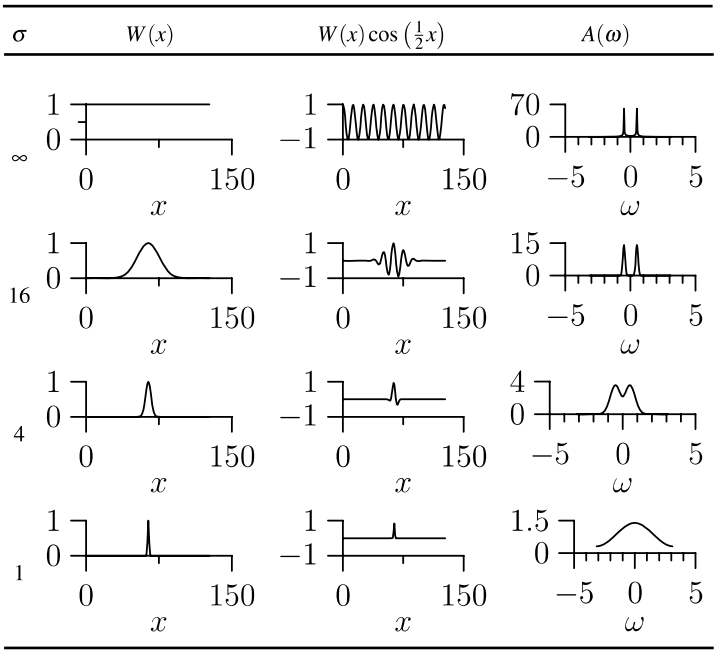

Fig. 2.14 In space-frequency analysis, there is a trade-off between spatial localization and spectral accuracy. This example shows how the use of a weighting window $W(x)$ (*second column*) changes the spectral (frequency) contents of a pure sinusoidal $I(x) = \cos(\frac{1}{2}x)$, $x = 0, \ldots, 127$. The *rightmost column* shows the amplitude spectrum of the localized signal $W(x)I(x)$, which in turn is plotted in the *third column*. With no spatial localization (window width $\sigma = \infty$; *top row*), the amplitude spectrum $A(\omega)$ shows a clear peak at the location $\omega = \frac{1}{2}$. As window width σ decreases, spatial localization increases, but accuracy in the spectral domain decreases: the two peaks in $A(\omega)$ spread out and eventually fuse so that $A(\omega)$ is a unimodal function when $\sigma = 1$

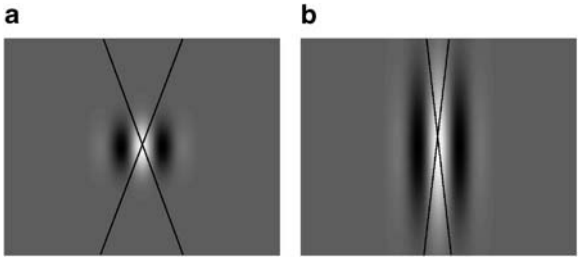

Fig. 2.15 Uncertainty in two dimensions. Compared with a "basic" Gabor function in **a**, the Gabor in **b** is very localized in orientation, that is, it responds to only a small range of different orientations. The *thin black lines* show the orientations of *thin lines* which still fall on the white (positive) area of the function: a line which is more oblique will partly fall on the black (negative) areas, and thus the response of the filter will the reduced. We can see that in **b**, the range of orientations producing very large responses (falling on the white area only) is much smaller than in **a**. This illustrates that in order to make the basic Gabor function in **a** more localized in orientation, it is necessary to make it longer, and thus to reduce its spatial localization

designed to respond only to a small range of orientations, which was only possible by making it more extended in space.

2.5 References

Most of the material covered in this chapter can be found in most image-processing textbooks. A classic choice is (Gonzales and Woods 2002), which does not, however, consider space-frequency analysis. A large number of textbooks explain time-frequency analysis, which is the one-dimensional counterpart of space-frequency analysis, for example (Cohen 1995). Related material can also be found in textbooks on wavelets, which are a closely related method (see Section 17.3.2 for a very short introduction), for example (Vetterli and Kovačević 1995; Strang and Nguyen 1996).

2.6 Exercises

Mathematical Exercises

1. Show that convolution is a symmetric operation.
2. Show (2.3).
3. Prove (2.17). Hint: Find two different values for x so that you get the two equations

$$A\cos\psi = C, \tag{2.27}$$

$$-A\sin\psi = S. \tag{2.28}$$

   Now, solve for A and ψ as follows. First, take the squares of both sides of both (2.27) and (2.28) and sum the two resulting equations. Recall the sum of squares of a sine and a cosine function. Second, divide both sides of (2.27) and (2.28) with each other.

Computer Assignments

The computer assignments in this book are designed to be made with Matlab™. Most of them will work on Matlab clones such as Octave. We will assume that you know the basics of Matlab.

1. The command `meshgrid` is very useful for image processing. It creates two-dimensional coordinates, just like a command such as `[5:.01:5]` creates a one-dimensional coordinate. Give the command `[X,Y]=meshgrid([-5:0.1:5])`; and plot the matrices `X` and `Y` using `imagesc`.

2. Create Fourier gratings by sin(X), sin(Y), sin(X+Y). Plot the gratings.
3. Create a Gabor function using these X and Y, simply by plugging in those matrices in the formula in (2.24). Try out different values for the parameters until you get a function which looks like the one in Fig. 2.12b.
4. Change the roles of X and Y to get a Gabor filter in a different orientation.
5. Try out a Gabor function of a different orientation by plugging in X+Y instead of X and X-Y instead of Y.
6. Linear filtering is easily done with the function conv2 (the "2" means two-dimensional convolution, i.e. images). Take any image, import it to Matlab, and convolve it with the three Gabor functions obtained above.

http://www.springer.com/978-1-84882-490-4

Natural Image Statistics
A Probabilistic Approach to Early Computational Vision.
Hyvärinen, A.; Hurri, J.; Hoyer, P.O.
2009, XIX, 448 p., Hardcover
ISBN: 978-1-84882-490-4