This leads to the following iteration scheme for computing a discrete scale space:

$$g(x, \xi + \Delta\xi) = \Delta\xi g(x + \Delta x, \xi) + (1 - 2\Delta\xi) g(x, \xi) + \Delta\xi g(x - \Delta x, \xi) \tag{8.138}$$

or written with discrete coordinates

$$g_{n,\xi+1} = \Delta\xi g_{n+1,\xi} + (1 - 2\Delta\xi) g_{n,\xi} + \Delta\xi g_{n-1,\xi} \tag{8.139}$$

Lindeberg [16] shows that this iteration results in a discrete scale space that meets the minimum-maximum principle and the semi-group property if and only if

$$\Delta\xi \le \frac{1}{4} \tag{8.140}$$

The limit case of $\Delta\xi = 1/4$ leads to the especially simple iteration

$$g_{n,\xi+1} = 1/4 g_{n+1,\xi} + 1/2 g_{n,\xi} + 1/4 g_{n-1,\xi} \tag{8.141}$$

Each step of the scale-space computation is given by a smoothing of the signal with the binomial mask $\boldsymbol{B}^2 = [1/4\ 1/2\ 1/4]$ (Section 9.5.4). We can also formulate the general scale-space generating operator in Eq. (8.139) using the convolution operator $\mathcal{B}$. Written in the operator notation introduced in Section 9.1.3, the operator for one iteration step to generate the discrete scale space is

$$(1 - \epsilon)\mathcal{I} + \epsilon\mathcal{B}^2 \quad \text{with} \quad \epsilon \le 1 \tag{8.142}$$

where $\mathcal{I}$ denotes the identy operator.

This expression is significant, as it can be extended directly to higher dimensions by replacing $\mathcal{B}^2$ with a correspondingly higher-dimensional smoothing operator. The convolution mask $\boldsymbol{B}^2$ is the simplest mask in the class of smoothing binomial filters. These filters will be discussed in Section 9.5.4. A detailed discussion of discrete linear scale spaces is given by Lindeberg [16, Chapters 3 and 4].

8.10 Multigrid representations

8.10.1 Basics

The scale space discussed in Section 8.9 has one significant disadvantage. The use of the additional scale parameter adds a new dimension to the images and thus leads to an explosion of the data storage requirements and, in turn, the computational overhead for generating the scale space and for analyzing it. Thus, it is not surprising that before the evolution of the scale space more efficient multiscale storage schemes, especially pyramids, found widespread application in image processing. With data structures of this type, the resolution of the images decreases to such an extent as the scale increases. In this way an optimum balance between spatial and wave-number resolution is achieved in the sense of the *uncertainty relation* (Section 8.6.3). Data structures of this type are known as *multiresolution representations* [15].

The basic idea is quite simple. While the representation of fine scales requires the full resolution, coarser scales can be represented at lower resolution. This leads to a scale space with smaller and smaller images as the scale parameter increases. In the following two sections we will discuss the *Gaussian pyramid* (Section 8.10.2) and the *Laplacian pyramid* (Section 8.10.3) as efficient discrete implementations of discrete scale spaces. In addition, while the Gaussian pyramid constitutes a standard scale space, the Laplacian pyramid is a discrete version of a differential scale space (Section 8.9.3). The Gaussian and Laplacian pyramids are examples of multigrid data structures, which were introduced into digital image processing in the early 1980s and since then have led to a tremendous increase in speed of image-processing algorithms. A new research area, *multiresolutional image processing*, was established [15].

8.10.2 Gaussian pyramid

When *subsampling* an image, for example, by taking every second pixel in every second line it is important to consider the *sampling theorem* (Section 8.4.2). Before subsampling, the image must be smoothed to an extent that no aliasing occurs in the subsampled image. Consequently, for subsampling by a factor two, we must ensure that all structures, which are sampled less than four times per wavelength, are suppressed by an appropriate smoothing filter. This means that size reduction must go hand-in-hand with appropriate smoothing.

Generally, the requirement for the smoothing filter can be formulated as

$$\hat{B}(\tilde{\boldsymbol{k}}) = 0 \quad \forall \tilde{k}_d \geq \frac{1}{r_d} \tag{8.143}$$

where r_d is the subsampling rate in the direction of the dth coordinate.

The combined smoothing and size reduction can be expressed in a single operator by using the following notation to compute the $q+1$th level of the Gaussian pyramid from the qth level:

$$\boldsymbol{G}^{(q+1)} = \mathcal{B}_{\downarrow 2}\boldsymbol{G}^{(q)} \tag{8.144}$$

The number behind the $\downarrow$ in the index denotes the subsampling rate. Level 0 of the pyramid is the original image: $\boldsymbol{G}^{(0)} = \boldsymbol{G}$.

If we repeat the smoothing and subsampling operations iteratively, we obtain a series of images, which is called the *Gaussian pyramid*. From level to level, the resolution decreases by a factor of two; the size of the images decreases correspondingly. Consequently, we can think of the series of images as being arranged in the form of a pyramid.

The pyramid does not require much storage space. Generally, if we consider the formation of a pyramid from a D-dimensional image with a subsampling factor of two and N pixels in each coordinate direction, the total number of pixels is given by

$$N^D\left(1+\frac{1}{2^D}+\frac{1}{2^{2D}}+\ldots\right)<N^D\frac{2^D}{2^D-1} \qquad (8.145)$$

For a 2-D image, the whole pyramid needs just 1/3 more space than the original image, for a 3-D image only 1/7 more. Likewise, the computation of the pyramid is equally effective. The *same* smoothing filter is applied to each level of the pyramid. Thus the computation of the *whole* pyramid needs only 4/3 and 8/7 times more operations than for the first level of a 2-D and 3-D image, respectively.

The pyramid brings large scales into the range of local neighborhood operations with small kernels. Moreover, these operations are performed efficiently. Once the pyramid has been computed, we can perform neighborhood operations on large scales in the upper levels of the pyramid—because of the smaller image sizes—much more efficiently than for finer scales.

The Gaussian pyramid constitutes a series of low-pass filtered images in which the cutoff wave numbers decrease by a factor of two (an octave) from level to level. Thus the Gaussian pyramid resembles a logarithmic scale space. Only a few levels of the pyramid are necessary to span a wide range of wave numbers. If we stop the pyramid at an 8×8 image, we can usefully compute only a seven-level pyramid from a 512×512 image.

8.10.3 Laplacian pyramid

From the Gaussian pyramid, another pyramid type can be derived, that is, the *Laplacian pyramid*. This type of pyramid is the discrete counterpart to the *differential scale space* discussed in Section 8.9.3 and leads to a sequence of bandpass-filtered images. In contrast to the Fourier transform, the Laplacian pyramid leads only to a coarse wave-number decomposition without a directional decomposition. All wave numbers, independently of their direction, within the range of about an octave (factor of two) are contained in one level of the pyramid.

Because of the coarse wave number resolution, we can preserve a good spatial resolution. Each level of the pyramid contains only matching scales, which are sampled a few times (two to six) per wavelength.

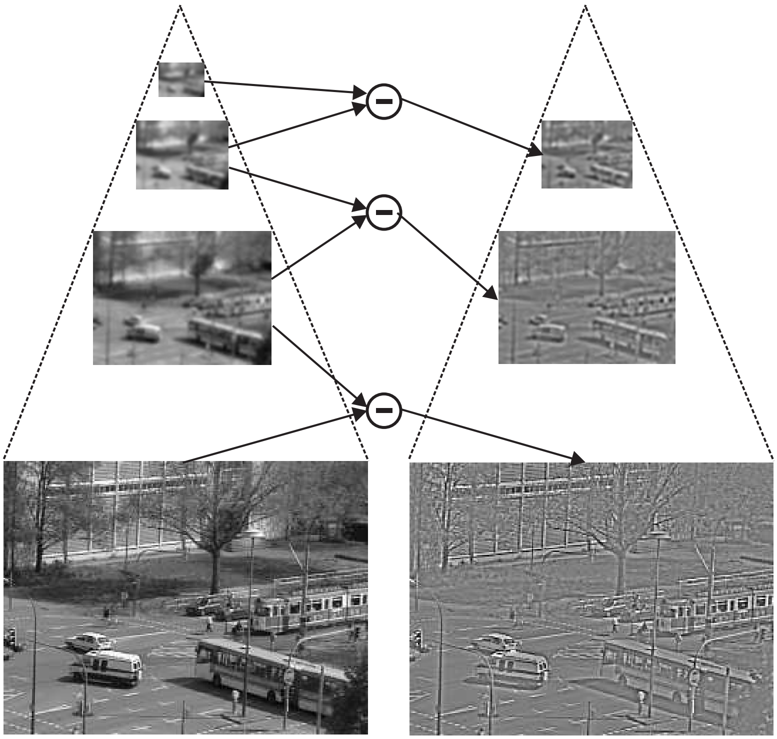

Figure 8.17: *Construction of the Laplacian pyramid (right column) from the Gaussian pyramid (left column) by subtracting two consecutive planes of the Gaussian pyramid.*

In this way, the Laplacian pyramid is an efficient data structure well adapted to the limits of the product of wave number and spatial resolution set by the *uncertainty relation* (Section 8.6.3).

The differentiation in scale direction in the continuous scale space is approximated by subtracting two levels of the Gaussian pyramid in the discrete scale space. In order to do so, first the image at the coarser level must be expanded. This operation is performed by an *expansion operator* $\mathcal{E}_{\uparrow 2}$. As with the reducing smoothing operator, the degree of expansion is denoted by the figure after the ↑ in the index.

The expansion is significantly more difficult than the size reduction because the missing information must be interpolated. For a size increase of two in all directions, first, every second pixel in each row must be interpolated and then every second row. Interpolation is discussed in detail in Section 9.6. With the introduced notation, the generation of the pth level of the Laplacian pyramid can be written as:

$$\mathcal{L}^{(p)} = \boldsymbol{G}^{(p)} - \mathcal{E}_{\uparrow 2}\boldsymbol{G}^{(p+1)} \tag{8.146}$$

The Laplacian pyramid is an effective scheme for a *bandpass decomposition* of an image. The center wave number is halved from level to level. The last image of the Laplacian pyramid is a low-pass-filtered image containing only the coarsest structures.

The Laplacian pyramid has the significant advantage that the original image can be reconstructed quickly from the sequence of images in the Laplacian pyramid by recursively expanding the images and summing them up. In a Laplacian pyramid with $p+1$ levels, the level p (counting starts with zero!) is the coarsest level of the Gaussian pyramid. Then the level $p-1$ of the Gaussian pyramid can be reconstructed by

$$\boldsymbol{G}^{(p-1)} = \mathcal{L}^{(p-1)} + \mathcal{E}_{\uparrow 2}\boldsymbol{G}^{p} \tag{8.147}$$

Note that this is just the inversion of the construction scheme for the Laplacian pyramid. This means that even if the interpolation algorithms required to expand the image contain errors, they affect only the Laplacian pyramid and not the reconstruction of the Gaussian pyramid from the Laplacian pyramid, because the same algorithm is used. The recursion in Eq. (8.147) is repeated with lower levels until level 0, that is, the original image, is reached again. As illustrated in Fig. 8.17, finer and finer details become visible during the reconstruction process.

8.11 References

[1] Wandell, B. A., (1995). *Foundations of Vision.* Sunderland, MA: Sinauer Associates.

[2] Papoulis, A., (1991). *Probability, Random Variables, and Stochastic Processes.* New York: McGraw-Hill.

[3] Kittel, C., (1971). *Introduction to Solid State Physics.* New York: Wiley.

[4] Hoffman, K. and Kunze, R., (1971). *Linear Algebra,* 2nd edition. Englewood Cliffs, NJ: Prentice-Hall.

[5] Jaroslavskij, J. P., (1985). *Einführung in die digitale Bildverarbeitung.* Berlin: VEB Deutscher Verlag der Wissenschaften.

[6] Besslich, P. W. and Lu, T., (1990). *Diskrete Orthogonaltransformation. Algorithmen und Flußgraphen für die Signalverarbeitung.* Berlin: Springer.

[7] Poularikas, A. D. (ed.), (1995). *The Transforms and Applications Handbook.* Boca Raton, FL: CRC Press.

[8] Bracewell, R., (1986). *The Fourier Transform and Its Applications,* 2nd edition, revised. New York: McGraw-Hill.

[9] Madisetti, V. K. and Williams, D. B. (eds.), (1997). *The Digital Signal Processing Handbook.* Boca Raton, FL: CRC Press.