2 Preprocessing

Preprocessing methods are often employed in multimedia signal analysis, compression, and identification. Nonlinear filtering methods are preferably applied in preprocessing of image signals, as they can preserve characteristic signal properties like edges, lines, blobs and corners much better than linear filtering. Linear filter kernels can also be used in combination with adaptation mechanisms, which take into account specific local properties of the signal that shall be preserved, or employ mechanisms which are specifically tuned to the expected disturbances which shall be removed. Amplitude transformations establish another group of signal modifications, which are mainly based on manipulations of sample statistics. Different interpolation methods are introduced which have to be employed when signal samples are needed at quasi-continuous positions not available from the given sampling grid. Multi-resolution methods introduced in the last part of this chapter play an important role in various feature extraction methods, to stabilize them against noise disturbances, and make them invariant against scale changes.

In filtering operations, the amplitude value of the sample at position (n_1,n_2) is set in relation with values from a neighborhood $\mathcal{N}(n_1,n_2)$, and modified accordingly. The neighborhood system can also be interpreted by determining the shape of the *filter mask*. The output is computed by applying a combining function to any position of the input. It shall be assumed here that the output has the same number and arrangement of samples as the input. If the combining function $f[\cdot]$ is nonlinear, the system is a nonlinear filter, as shown in Fig. 2.1 for the case of 2D processing over a finite neighborhood[1]. If the system would be linear and shift invariant (LSI), the combining function would be its impulse response (in case of finite neighborhood, representing an FIR filter), which could further be mapped into a Fourier transfer function for an interpretation by a *frequency response*. The latter

[1] Nonlinear filters are subsequently explained mostly for the example of 2D signals (images). Equivalent one- or multi-dimensional nonlinear filtering approaches can be defined by using equivalent neighborhoods over only one or more than two coordinate axes.

J.-R. Ohm, *Multimedia Content Analysis*, Signals and Communication Technology,
DOI 10.1007/978-3-662-52828-0_2

is generally not possible for the case of a nonlinear system[2]. When the combining function does not change depending on the coordinate position where it is applied, the nonlinear system is shift invariant.

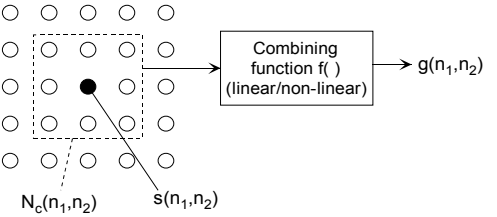

Fig. 2.1. Principle of linear or nonlinear 2D filtering using a finite symmetric neighborhood system

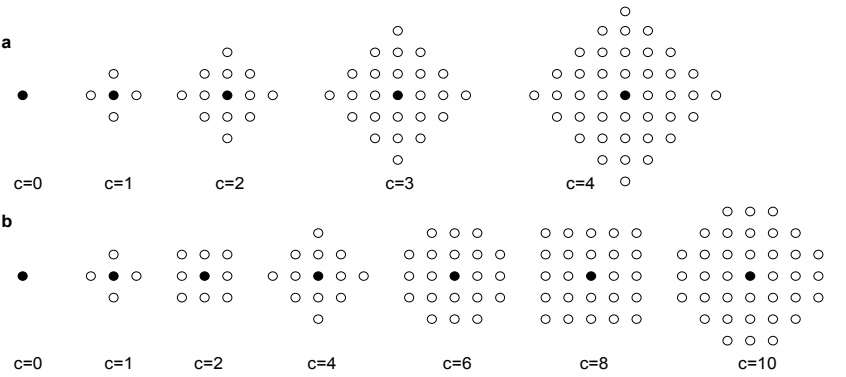

Fig. 2.2. Homogeneous 2D neighborhood systems $\mathcal{N}_c^{(P)}(n_1,n_2)$ with $P=1$ (**a**) and $P=2$ (**b**) for various values of c.

Symmetric neighborhood systems are often used in context of linear and nonlinear filtering of images, avoiding shifting and degenerative effects of local structures in the output. This property is fulfilled by a *homogeneous neighborhood system*, where samples at positions **m** establish the neighborhood of a sample at position **n** according to a maximum distance norm of order P[3]:

$$\mathcal{N}_c^{(P)}(\mathbf{n}) = \left\{ \mathbf{m} = \begin{bmatrix} m_1 & \dots & m_\kappa \end{bmatrix}^{\mathrm{T}} \;:\; 0 < \sum_{i=1}^{\kappa} \left| m_i - n_i \right|^P \le c \right\}, \left(P \vee c \right) \ge 0 \,. \qquad (2.1)$$

The parameter c influences the size, while P influences the shape of the neighbor-

[2] The system transfer function of polynomial filters (Sec. 2.1.3) could be mapped into a higher-order spectral transfer function by applying multi-dimensional Fourier transform.

[3] For images, the number of dimensions is $\kappa=2$. In case of symmetric neighborhood systems, the current sample at position **n** is also a member of the corresponding neighborhood systems of any of its neighbors, $\mathbf{m} \in \mathcal{N}(\mathbf{n}) \Leftrightarrow \mathbf{n} \in \mathcal{N}(\mathbf{m})$.

hood system. Fig. 2.2 shows examples for distance norms $P=1$, the 'diamond shaped' neighborhood, and $P=2$, the circular neighborhood. The trivial case $c=0$ means that no neighborhood is defined beyond the current sample $\mathbf{n}$, while $P=0$ would extend the neighborhood to infinity for any value $c \geq \kappa$. Various values of c are shown, the position $\mathbf{n}=[n_1\ n_2]^T$ is marked by '•'.

2.1 Nonlinear filters

Image signals show some properties which can hardly be modeled by band-limited LSI systems, in particular when structures with amplitude discontinuities (e.g. edges) are present. Consequently, methods of nonlinear filtering are widely used in image processing applications. Different types of filters which are particularly suitable for image signal simplification, outlier removal and enhancement, while usually retaining or possibly sharpening important structures such as edges, are explained in the following subsections. Rank-order filters and morphological filters implement combination functions which are based on value comparisons and logical operations within the neighborhood. Herein, a certain overlap of subtypes exists between the two categories, as illustrated in Fig. 2.3 left. When sample amplitudes from the neighborhood establish the weighted elements within a polynomial of certain degree, the approach is denoted as *polynomial filter*. In case of linear filters, the highest degree is one, in case of Volterra filters, the highest degree is two. An important subclass of linear filters are the time or shift invariant systems (LTI/LSI). Diffusion filters are based on an iterative application of LTI/LSI smoothing filters. Linear filter kernels are also frequently used in some adaptive context, where the mode of operation is varied, or the kernel is steered based on local signal properties. By this, shift invariance, as well as the validity of the superposition principle (flexibility to perform additive superposition of signals either at input or after output) is lost. More detail about those filter types is contained in Sec. 2.5.

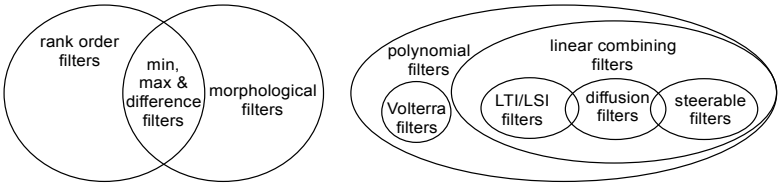

Fig. 2.3. Categories of linear and non-linear filters

2.1.1 Median filters and rank-order filters

The median is a value from a set, for which at least half the number of values is less or equal, and at least half the number of values is larger or equal. In median filtering of image signals, the amplitude values from the neighborhood of position

n are forming the set, which should consist of an odd number (e.g. 3x3 or 5x5 samples) for unique definition[4]. By ordering the set as a list of increasing amplitudes, the median is found in the mid of the list, being released as output at position **n**. Median filtering has an effect of eliminating isolated outlier values. It can also be applied for nonlinear prediction and interpolation, as an alternative instead of the linear filters typically used for such purposes.

Example: Median filter over neighborhood $\mathcal{N}_2^{(2)}(n_1,n_2)$. As an example, an image matrix **S** is given in (2.2), with constant-value extension assumed where the filter mask would access samples from outside of **S**. With 3x3 filter mask centered at the second sample in the second row, the set $\mathcal{M}$ = [10,10,20,20,10,20,10,10,10] is the filter input. Amplitude re-ordering gives $\mathcal{M}'$ = [10,10,10,10,**10**,10,20,20,20], resulting in the value MED[$\mathcal{M}$] = 10. The sample remains unchanged, as its value is identical to the median. For the third sample in the third row, the set is $\mathcal{M}$ = [10,20,20,10,10,20,10,20,20], re-ordered $\mathcal{M}'$ =[10,10,10,10,**20**,20,20,20,20], and MED[$\mathcal{M}$]=20. The median output is not identical, but it is one original value stemming from the neighborhood. Application of the same operation to any position gives the output matrix **G**, from which it is apparent that the median filter eliminates single, isolated amplitude values and straightens edge boundaries between areas of constant amplitude:

$$\mathbf{S} = \begin{bmatrix} 10 & 10 & 20 & 20 \\ 20 & \underline{10} & 20 & 20 \\ 10 & 10 & \underline{10} & 20 \\ 10 & 10 & 20 & 20 \end{bmatrix}; \quad \mathbf{G} = \mathrm{MED}[\mathbf{S}] = \begin{bmatrix} 10 & 10 & 20 & 20 \\ 10 & 10 & 20 & 20 \\ 10 & 10 & 20 & 20 \\ 10 & 10 & 20 & 20 \end{bmatrix}. \tag{2.2}$$

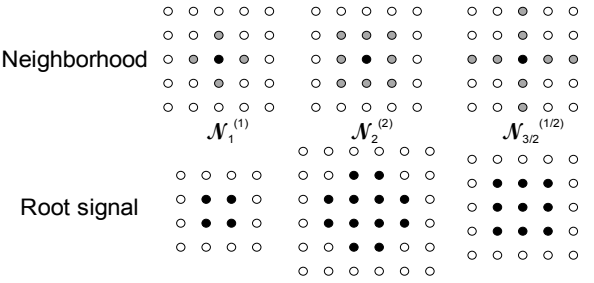

Fig. 2.4. 2D median filter geometries and related root signals

The *root signal* of a median filter with a certain filter mask geometry is the smallest neighborhood constellation of samples with identical amplitudes that will remain unchanged in case of an iterated application of the filter. For any position

[4] Methods which allow usage of even number of input values are weighted median filtering (see below), or averaging the two values at the center of the ordered set, or systematically selecting one of them by definition.

of the root signal, the majority of neighbors shall then also be a member of the root signal. In case of symmetric neighborhoods, the root signal will be symmetric as well. The shape of the root signal relates to the resolution preservation capability of a median filter; any detail structures that are 'smaller' than the root signal could possibly be eliminated by the filter. Examples of 2D median filter geometries and their root signals are shown in Fig. 2.4.

Median filters have an effect of 'equalization', they tend to reduce the number of distinct amplitudes within a local environment. Even though, positions of amplitude discontinuities are usually preserved. Fig. 2.5a shows the effects of a median filter and of an LSI mean-value filter applied to an idealized edge (amplitude discontinuity, shown as section over one dimension). The discontinuity is smoothed by the LSI system, but preserved by the median filter. Edge ringing is also eliminated by median filters, since the overshoots are considered as outliers (Fig. 2.5b). However, thin lines, which are typically relevant structures of images, are often eliminated as well.

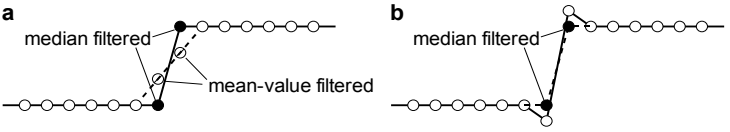

Fig. 2.5. Effects of filters with neighborhood $\mathcal{N}_2^{(2)}$ (width 3) **a** Median filter and mean-value filter at a flat amplitude discontinuity **b** Median filter at an amplitude discontinuity with overshoots (e.g. edge ringing often caused by linear lowpass filters)

Variants of median filters are:

- Weighted median filters: For each position under the filter mask, an integer-number weighting factor $w(m_1,m_2)$ is defined. In the sets $\mathcal{M}$ and $\mathcal{M}'$, the value of the respective sample is included $w(m_1,m_2)$ times. Typically, the sample positioned at the center of the mask is given the highest weight. The root signal will then cover a smaller area and also retain smaller structures[5]. By application of weighted median filters, thin lines in images can be preserved, while single isolated samples of different amplitude would still be discarded. This is illustrated in the example (2.3), where output values of '10' would be produced at all positions by a non-weighted median filter of neighborhood size 3 x 3. If the center sample is weighted by $w(0,0)=5$, the single isolated value '20' in the second column of **S** is erased, but the column of values '20' is preserved.

$$\mathbf{S}=\begin{bmatrix}10&10&10&20&10\\10&10&10&20&10\\10&20&10&20&10\\10&10&10&20&10\\10&10&10&20&10\end{bmatrix};\ \mathbf{G}=\mathrm{MED}_{\mathbf{W}}[\mathbf{S}]=\begin{bmatrix}10&10&10&20&10\\10&10&10&20&10\\10&10&10&20&10\\10&10&10&20&10\\10&10&10&20&10\end{bmatrix};\ \mathbf{W}=\begin{bmatrix}1&1&1\\1&5&1\\1&1&1\end{bmatrix}. \quad (2.3)$$

[5] For definition of root signals for weighted median filters, see Problem 2.2.

- Hybrid linear/median filters: Output signals of different linear filters establish the set for median computation.

These generalized types of median filters are also belonging to the category of *rank order filters*, as they operate by selecting values from ordered lists. Other types of rank order filters are

- Minimum-value filters, producing as output the *minimum amplitude* from the neighborhood,

$$g_{\min}(\mathbf{n}) = \min_{\mathbf{m}\in\mathcal{N}(\mathbf{n})}\left[s(\mathbf{m})\right]; \tag{2.4}$$

- Maximum-value filters, producing as output the *maximum amplitude* from the neighborhood,

$$g_{\max}(\mathbf{n}) = \max_{\mathbf{m}\in\mathcal{N}(\mathbf{n})}\left[s(\mathbf{m})\right]; \tag{2.5}$$

- Difference filters, producing as output the *maximum difference between any two values* from the neighborhood, which is always positive,

$$g_{\text{diff}}(\mathbf{n}) = g_{\max}(\mathbf{n}) - g_{\min}(\mathbf{n}). \tag{2.6}$$

Example. The image matrix **S** is transformed into the following output images if minimum, maximum and difference filters of mask size 3 x 3 are applied:

$$\mathbf{S} = \begin{bmatrix} 10 & 10 & 10 & 20 & 20 \\ 10 & 10 & 10 & 20 & 20 \\ 10 & 10 & 10 & 20 & 20 \\ 10 & 10 & 10 & 20 & 20 \\ 10 & 10 & 10 & 20 & 20 \end{bmatrix} \Rightarrow \mathbf{G}_{\min} = \begin{bmatrix} 10 & 10 & 10 & 10 & 20 \\ 10 & 10 & 10 & 10 & 20 \\ 10 & 10 & 10 & 10 & 20 \\ 10 & 10 & 10 & 10 & 20 \\ 10 & 10 & 10 & 10 & 20 \end{bmatrix};$$

$$\mathbf{G}_{\max} = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}; \quad \mathbf{G}_{\text{diff}} = \begin{bmatrix} 0 & 0 & 10 & 10 & 0 \\ 0 & 0 & 10 & 10 & 0 \\ 0 & 0 & 10 & 10 & 0 \\ 0 & 0 & 10 & 10 & 0 \\ 0 & 0 & 10 & 10 & 0 \end{bmatrix}. \tag{2.7}$$

The effect of a minimum-value filter *erodes* isolated peaks or amplitude plateaus from a signal, while a maximum-value filter discards minima or fills up troughs of the amplitude shape. The difference filter allows to analyze a kind of *nonlinear gradient* within an image signal. These latter types of rank order filters also have an interpretation as *morphological filters* (see the subsequent section). According to their effect, the minimum-value filter is then denoted as *erosion filter*, while the maximum-value filter is the *dilation filter*.

Median filtering can be extended to vector processing, e.g. when color sample values of a picture are used as input. In this context, the following options can be applied, where it depends on the goal of the filtering which maybe the best choice:

- Separate processing of the scalar elements in the vector, which has the disadvantage that the output vector may be a combination of scalar values not contained in the input set;
- Ordering based on magnitude or angle of the vector;
- Ordering based on some other logical or arithmetic combination of the elements of the vector, e.g. non-Euclidean norm;
- Ordering based on only a subset of the scalar elements of the vector.

2.1.2 Morphological filters

The term morphology is deduced from an ancient Greek word for 'shape' or 'figure'. Morphological filtering is originally applied to manipulate geometric shapes expressed as logical (binary) signals, where

$$b(\mathbf{n}) = \begin{cases} 1: & \text{part of the region shape,} \\ 0: & \text{not part of the region shape.} \end{cases} \tag{2.8}$$

By generalization described later, morphological filters can also be applied to signals and functions of multiple-level amplitude values, such as gray-level or color images. They can further be used for nonlinear contrast enhancement, elimination or emphasis of local details, or detection of characteristic points such as edges and corners in image signals.

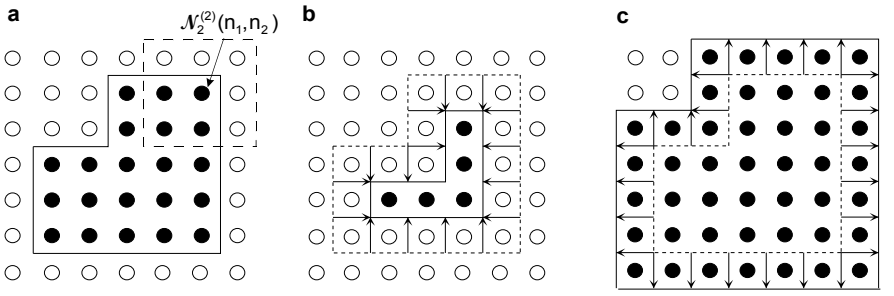

Fig. 2.6. Basic morphological operations: **a** Binary object $\boldsymbol{O}$ with structure element $\boldsymbol{E}$, centered around position $\mathbf{n}$ **b** Output after erosion **c** Output after dilation [black bullets indicate $b(\mathbf{n})=1$, white bullets $b(\mathbf{n})=0$]

Operation with binary signals. The two basic operations in morphological filtering are *erosion* and *dilation*. Fig. 2.6a shows the example of a binary shape of an object $\boldsymbol{O}(\mathbf{n})$ constituted by the logical '1' values in $b(\mathbf{n})$, with a *structure element* $\boldsymbol{E}(\mathbf{n})$ of size 3×3 samples centered around position $\mathbf{n}$, which can be expressed as a homogeneous neighborhood $\mathcal{N}_2^{(2)}(\mathbf{n})$. The structure element is testing all posi-

tions of $\boldsymbol{O}(\mathbf{n})$. In case of erosion, a logical '1' output is produced, when *all values* of $\boldsymbol{O}$ are set as $b(\mathbf{n})=1$ under $\boldsymbol{E}$ at a given position (Fig. 2.6b)[6],

$$\boldsymbol{O}(\mathbf{n}) \ominus \boldsymbol{E}(\mathbf{n}) = b_{\text{er}}(\mathbf{n}) = \begin{cases} 1 \text{ if } \sum\limits_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} b(\mathbf{m}) = |\mathcal{N}|, \\ 0 \text{ if } \sum\limits_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} b(\mathbf{m}) < |\mathcal{N}|. \end{cases} \tag{2.9}$$

The counterpart operation is *dilation*, where a logical '1' output is produced, when *at least one* non-zero value of $\boldsymbol{O}$ is found under $\boldsymbol{E}$ at the respective position (Fig. 2.6c)[7],

$$\boldsymbol{O}(\mathbf{n}) \oplus \boldsymbol{E}(\mathbf{n}) = b_{\text{di}}(\mathbf{n}) = \begin{cases} 1 \text{ if } \sum\limits_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} b(\mathbf{m}) > 0, \\ 0 \text{ if } \sum\limits_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} b(\mathbf{m}) = 0. \end{cases} \tag{2.10}$$

In the example shown by Fig. 2.6, the operations of erosion and dilation are *reversible*, such that from the shape of Fig. 2.6b the original object (Fig. 2.6a) is reconstructed by dilation; from the shape in Fig. 2.6c it is reconstructed by erosion. Reversibility is not generally guaranteed, in particular single holes in the object shape, noses or troughs in the object boundaries would usually be lost. From the basic operations erosion and dilation, additional morphological features can be defined. The *inner contour* of an object is given by the logical *exclusive-or* combination (or absolute difference) of the original and eroded signals,

$$\boldsymbol{O}(\mathbf{n}) - \big(\boldsymbol{O}(\mathbf{n}) \ominus \boldsymbol{E}(\mathbf{n})\big), \tag{2.11}$$

while the *outer contour* results by logical *exclusive-or* combination of original and dilated signals,

$$\big(\boldsymbol{O}(\mathbf{n}) \oplus \boldsymbol{E}(\mathbf{n})\big) - \boldsymbol{O}(\mathbf{n}) \,. \tag{2.12}$$

By appropriate choice of the structure element's shape or additional criteria (e.g. minimum or maximum number of samples that must belong to the object when the set under the structure element is analyzed), further features like corner samples of an object shape can be extracted. The operation of *opening*,

$$\boldsymbol{O}(\mathbf{n}) \circ \boldsymbol{E}(\mathbf{n}) = \big(\boldsymbol{O}(\mathbf{n}) \ominus \boldsymbol{E}(\mathbf{n})\big) \oplus \boldsymbol{E}(\mathbf{n}) \tag{2.13}$$

is defined as erosion followed by dilation, which straightens convex shapes and eliminates thin noses. The counterpart is *closing*

---

[6] $|\mathcal{N}|$: Size of neighborhood (by sample count). Erosion is alternatively defined via *Minkovsky subtraction* $\boldsymbol{O} \ominus \boldsymbol{E} = \big\{\mathbf{n} \big| [\boldsymbol{E} + \mathbf{n}] \subseteq \boldsymbol{O}\big\}$. Herein, the expression '$\boldsymbol{E}+\mathbf{n}$' characterizes a shift of $\boldsymbol{E}$ to position $\mathbf{n}$.

[7] Alternatively defined via *Minkovsky addition* $\boldsymbol{O} \oplus \boldsymbol{E} = \big\{\mathbf{n} \big| [\boldsymbol{E} + \mathbf{n}] \cap \boldsymbol{O} \neq 0\big\}$.

$$\boldsymbol{O}(\mathbf{n}) \bullet \boldsymbol{E}(\mathbf{n}) = \big(\boldsymbol{O}(\mathbf{n}) \oplus \boldsymbol{E}(\mathbf{n})\big) \ominus \boldsymbol{E}(\mathbf{n}) \,, \tag{2.14}$$

defined as dilation followed by erosion, having an effect of straightening of concave shapes and elimination of holes, channels and troughs. Examples of opening and closing are illustrated in Fig. 2.7, where again a structure element of size 3x3 samples was used.

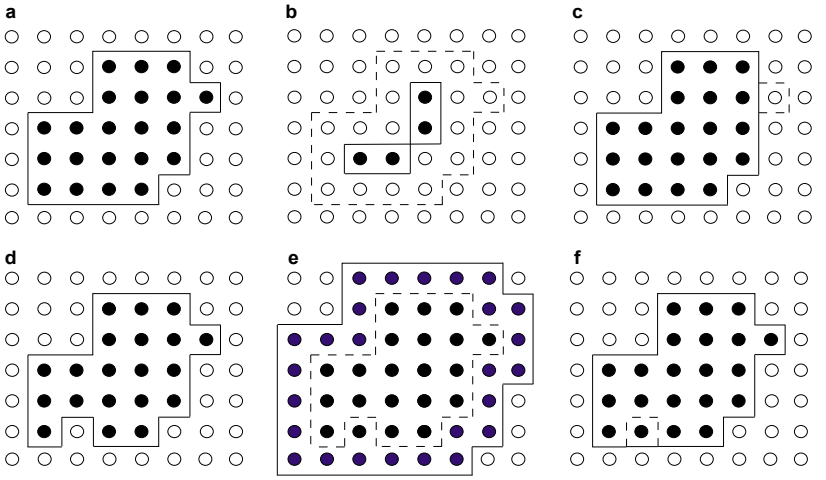

Fig. 2.7. **a** Original shape **b** erosion **c** opening (dilation of result in b)
d Original shape **e** dilation **f** closing (erosion of result in e)

Finally, it may be useful to identify those samples that were removed by opening, or added by closing. This can be done by the operations 'open by reconstruction' $\boldsymbol{O}(\mathbf{n}) - \big(\boldsymbol{O}(\mathbf{n}) \circ \boldsymbol{E}(\mathbf{n})\big)$ and 'close by reconstruction' $\big(\boldsymbol{O}(\mathbf{n}) \bullet \boldsymbol{E}(\mathbf{n})\big) - \boldsymbol{O}(\mathbf{n})$.

In principle, mainly the size of the structure element influences the strength of the effect that morphological filters impose. Alternatively, filters defined by small structure elements can also be applied iteratively to achieve a stronger effect. In some cases, it is also desirable to adapt the effect by the size of the object in general.

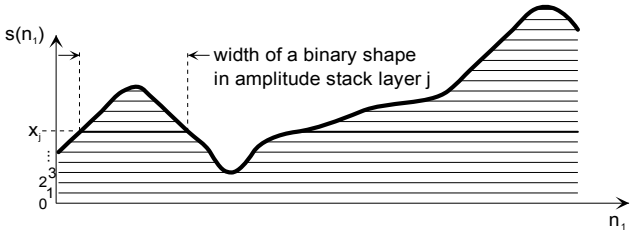

Fig. 2.8. Signal of multiple amplitude levels, composed from 'amplitude layers'

Operation with non-binary signals. A non-binary signal $s(\mathbf{n})$ has $J>2$ amplitude levels x_j, typically with uniform step size Δ and in case of positive amplitude $x_j=j\Delta$, $0<j<J$. For 2D signals, this can be interpreted such that the amplitudes represent a height surface. The volume under this surface is stacked by a number of $J-1$ 'layers' of height Δ, each of which possesses a binary 2D shape $\boldsymbol{O}_j(\mathbf{n})$[8]. At a sample position $\mathbf{n}$ which has a discrete amplitude value x_j, the stack has j layers. An additional condition is

$$\boldsymbol{O}_{j+1}(\mathbf{n}) \subseteq \boldsymbol{O}_j(\mathbf{n}) \,, \tag{2.15}$$

i.e. a binary shape referring to layer $j+1$ can never be logical '1' at a position when a lower layer at the same position is zero. A one-dimensional section of this stack, e.g. the amplitude profile along one image row, is shown in Fig. 2.8. The non-binary signal can be reconstructed as

$$s(\mathbf{n}) = \sum_{j=1}^{J-1} b_j(\mathbf{n}) \;\text{ with }\; \mathrm{b}_j(\mathbf{n}) = \begin{cases} 1 \;\text{ if }\; s(\mathbf{n}) \geq x_j \,, \\ 0 \;\text{ if }\; s(\mathbf{n}) < x_j \,. \end{cases} \tag{2.16}$$

The operations of erosion and dilation could now be interpreted such that they are executed separately on the binary images $b_j(\mathbf{n})$, once for each layer. The eroded or dilated shapes retain their original stacking order, however due to (2.15) a '1' shape in an eroded (or dilated) higher layer of the stack will still be smaller or equal in size compared to any of eroded (or dilated) lower layers, no cavities can ever appear in the height surface. Therefore erosion usually removes mass from the peaks of the surface and can eliminate high amplitudes, but never produces amplitudes that are lower than those originally existing in the neighborhood; dilation fills up mass to valleys in the surface and can eliminate low amplitudes, but never produces higher values than were existing before. Therefore, the results of dilation and erosion are exactly equivalent to the respective effects of maximum-value filters and minimum-value filters defined in (2.4) and (2.5), directly applied to the non-binary $s(\mathbf{n})$, which allows implementing the procedure much more efficiently than separate binary layer processing (Fig. 2.9a).

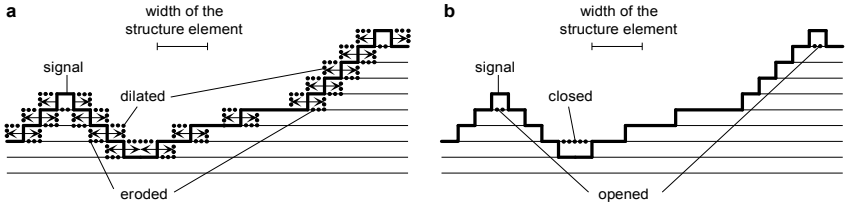

Fig. 2.9. Application of morphological operations to a non-binary signal
a Erosion and dilation **b** opening and closing

[8] Note that the layer referring amplitude $s(\mathbf{n})=0$ is not necessary in this representation, it could be interpreted to be $b_0(\mathbf{n})=0$ over all $\mathbf{n}$.

By tendency, the dilated image will have an increased mean value of amplitude, while for the eroded image the mean value is lowered. Opening and closing are defined as before by subsequent processing of erosion and dilation or vice versa, but now using min and max value filters as well. Both of these functions have the effect of a nonlinear equalization of signal amplitudes (Fig. 2.9b), where the opening eliminates high peaks, and closing fills up troughs of the amplitude volume. In principle, this modifies the signal such that plateaus of equal amplitudes are generated, similar to median filtering, but selectively for cases of low and high outliers.

Analogous with the inner and outer contour definitions in (2.11) and (2.12), *morphological gradients* can be defined for multi-level signals. A typical goal of gradient analysis is finding of contour points in 2D images. In the binary case, contours were defined by exclusive-or combinations of a shape and dilated (for outer contour) or eroded versions (for inner contour) thereof. Accordingly, gradients of multiple-level amplitude signals are defined by differencing the values of the eroded or dilated signals with the original signal. These are described as erosion or dilation gradients[9]. The *morphological gradient* is the difference between the dilated and eroded signals; this will be identical to the result of the min-max difference filter (2.6). Operations such as open-by-reconstruction and close-by reconstruction can be likewise applied by computing differences between the open/close result and the original; this identifies positions of high and low outliers (small peaks and valleys) in the amplitude surface of $s(\mathbf{n})$, respectively.

2.1.3 Polynomial filters

Plain (non-adaptive, shift invariant) polynomial filters are by conception a superset of LSI systems (the latter characterized by their impulse response), but include also nonlinear combinations of samples, e.g. by combining multiplications from several samples of the input signal. Practically, only polynomial filters up to order two, the *Volterra filters*, are relevant in multimedia signal processing. The transfer equation of a one-or multidimensional Volterra filter is defined as

$$g(\mathbf{n}) = F_1\left[s(\mathbf{n})\right] + F_2\left[s(\mathbf{n})\right], \tag{2.17}$$

where the linear (FIR) term is a conventional convolution (LSI operation)

$$F_1\left[s(\mathbf{n})\right] = \sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} a(\mathbf{m})s(\mathbf{n}-\mathbf{m}), \tag{2.18}$$

and the nonlinear (quadratic) term is

$$F_2\left[s(\mathbf{n})\right] = \sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})}\sum_{\mathbf{p}\in\mathcal{N}(\mathbf{n})} b(\mathbf{m},\mathbf{p})s(\mathbf{n}-\mathbf{m})s(\mathbf{n}-\mathbf{p}). \tag{2.19}$$

---

[9] For strictly positive values, the erosion gradient is defined by $s(\mathbf{n}) - g_{\min}(\mathbf{n})$, the dilation gradient by $g_{\max}(\mathbf{n}) - s(\mathbf{n})$.

Recursive structures can be defined similarly, where however stability of the non-linear component is not as straightforward to test as for the case of LSI systems. Since the computation of the output signal in the higher-order terms is similar to the computation of higher-order moments (Sec. 3.1), the latter can be used for optimizing coefficients $b(\mathbf{m},\mathbf{p})$ in (2.19), similarly as autocorrelation or autocovariance function are commonly used for optimizing linear filters (e.g. in the Wiener-Hopf equation (A.96)).

2.2 Amplitude-value transformations

Amplitude-value transformations define mappings of input amplitudes to output amplitudes. This can be interpreted as manipulation of probability distributions, e.g. a modification of the histogram for discrete-amplitude or of the PDF for continuous-amplitude signals. Amplitude mapping can either be performed globally or locally within small segments of a signal.

Contrast enhancement is a typical goal of amplitude value transformations applied to images. In audio signals, it is often desirable to *compress* amplitudes into a pre-defined range, such that dynamic fluctuations are limited. For example, it is convenient for the user of a radio, if the loudness stays relatively constant in audio broadcasting applications, such that it is not necessary to change the volume settings depending on local signal behavior. Another application of amplitude mapping is *companding* of signals, which is often done in transmission systems where noise is expected to interfere with the signal; when the low amplitudes are increased prior to transmission, the noise will be suppressed by an *expansion* (the reverse principle of compression), applied after receiving, which reduces the noise amplitude and reconstructs the signal into its original amplitude range. For example, this principle is applied in PCM encoding of speech signals where it provides suppression of quantization noise in low amplitude levels.

For images, amplitude mapping can be applied either to the luminance or to the color component amplitudes. In color mapping, it is not straightforward to define objective contrast enhancement functions, as the subjective color impression could be falsified in an undesirable manner. An extreme example is the usage of *color lookup tables*, which are used to systematically highlight specific image content without being coherent with the original natural color any more. Color lookup tables are also used when the number of different colors that can be displayed is limited; in such a case, the lookup table can be determined by a vector quantizer codebook design (cf. [MSCT, SEC. 4.5]). *Quantization* and *re-quantization* can indeed be considered as specific optimized cases of non-linear amplitude mapping.

2.2.1 Amplitude mapping characteristics

The amplitude value of a sample $s(\mathbf{n})$ shall be mapped into the output value $g(\mathbf{n})$. This relationship can be described by a mapping characteristic $\theta(\cdot)$, which could be linear or nonlinear:

$$g(\mathbf{n}) = \theta\left[s(\mathbf{n})\right]. \tag{2.20}$$

If the mapping characteristic is steady, unique and monotonous, the mapping is invertible, i.e.

$$s(\mathbf{n}) = \theta^{-1}\left[g(\mathbf{n})\right]. \tag{2.21}$$

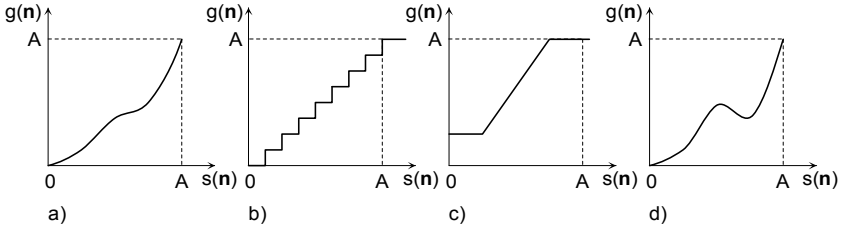

Fig. 2.10. Examples of mapping characteristics (explanations see text below)

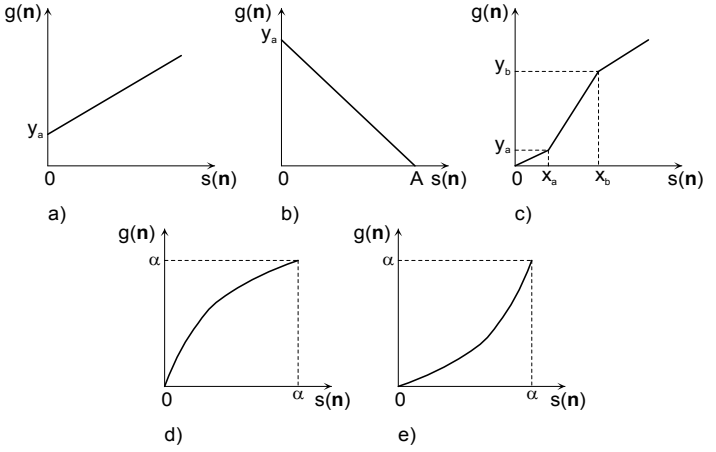

Fig. 2.11. Examples of invertible mapping characteristics (2.22)-(2.25)

Examples of mapping functions are shown in Fig. 2.10, where it is additionally assumed that the amplitude range of both $s(\mathbf{n})$ and $g(\mathbf{n})$ shall be in the interval $[0,A]$. The function in Fig. 2.10a would be invertible in case of continuous amplitudes. Non-invertible functions are shown in Fig. 2.10b (quantizer characteristic, which is unsteady), Fig. 2.10c (clipping characteristic, which is not unique in

certain ranges) and Fig. 2.10d (non-monotonous function). Some important invertible mapping characteristics are:

– *Linear characteristic* (Fig. 2.11a) :

$$g(\mathbf{n}) = \alpha s(\mathbf{n}) + y_a \; ; \quad s(\mathbf{n}) = \frac{1}{\alpha} g(\mathbf{n}) - \frac{y_a}{\alpha} \; , \tag{2.22}$$

which includes the case ($\alpha = -1$, $g_a = -A$) of negative amplitude mapping (Fig. 2.11b);

– *Piecewise-linear characteristic* (Fig. 2.11c):

$$g(\mathbf{n}) = \begin{cases} \alpha\, s(\mathbf{n}) & \text{for} \quad s(\mathbf{n}) \le x_a \\ \beta\big[s(\mathbf{n}) - x_a\big] + y_a & \text{for} \quad x_a \le s(\mathbf{n}) \le x_b \\ \ldots & \\ \sigma\big[s(\mathbf{n}) - x_r\big] + y_r & \text{for} \quad x_r \le s(\mathbf{n}) \end{cases} \tag{2.23}$$

with $y_a = \alpha x_a$, $y_b = \beta[x_b - x_a] + y_a$ etc.; this can be used with arbitrary number of pieces, and is invertible if all slopes ($\alpha, \beta, \ldots$) are non-zero with equal sign;

– *Root and quadratic characteristics*, which are examples of invertible compression/expansion[10] function pairs as illustrated in Figs. 2.11d/e

$$g(\mathbf{n}) = \sqrt{\alpha \left| s(\mathbf{n}) \right|}\, \mathrm{sgn}\big(s(\mathbf{n})\big) \quad ; \quad s(\mathbf{n}) = \frac{g^2(\mathbf{n})}{\alpha} \mathrm{sgn}\big(g(\mathbf{n})\big), \; \alpha > 0 \; ; \tag{2.24}$$

– *Logarithmic and exponential characteristics,* establishing another compression/expansion pair by

$$\begin{aligned} g(\mathbf{n}) &= \log_\alpha \big(1 + \left| s(\mathbf{n}) \right|\big) \mathrm{sgn}\big(s(\mathbf{n})\big) \\ s(\mathbf{n}) &= \big(\alpha^{|g(\mathbf{n})|} - 1\big) \mathrm{sgn}\big(g(\mathbf{n})\big) \end{aligned} \quad , \; \alpha > 1 \, . \tag{2.25}$$

2.2.2 Probability distribution modification and equalization

Mapping functions can be determined systematically, provided that criteria for optimization are given. As an example, the goal of a mapping might be

– to obtain a desired probability distribution at the output, e.g. to maximize the contrast of a signal or achieve a uniform distribution of probabilities in a discrete representation;
– to minimize the resulting error in the mapping from a continuous-value to a discrete-value signal (quantization)[11].

---

[10] The invertible combination of compression and expansion is also denoted as *companding*. It is e.g. used for noise suppression in transmission.

[11] See optimization of non-uniform quantizer characteristics, [MSCT, SEC. 4.1].

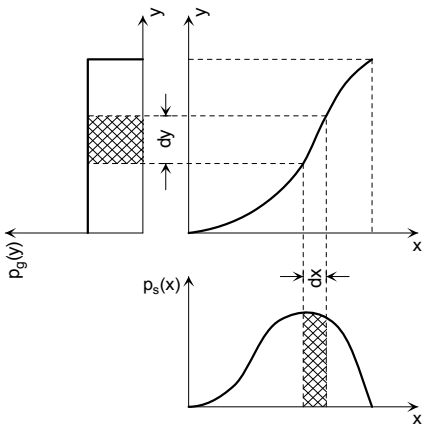

Fig. 2.12. Relationships of PDFs for input and output signals in amplitude mapping

For steady, monotonous functions, the areas under the PDFs within a differential amplitude range of the input and the corresponding range of the output have to be identical (see Fig. 2.12):

$$p_s(x)\,\mathrm{d}x = p_g(y)\,\mathrm{d}y \quad \Rightarrow \quad \frac{\mathrm{d}\theta(x)}{\mathrm{d}x} = \frac{p_s(x)}{p_g(y)} \quad \text{or} \quad \frac{\mathrm{d}\theta^{-1}(y)}{\mathrm{d}y} = \frac{p_g(y)}{p_s(x)}. \tag{2.26}$$

Further, the probability of samples within an amplitude interval $[x_a,x_b]$ of the input has to be identical to the probability within the corresponding output interval $[\theta(x_a),\theta(x_b)]$:

$$\int_{x_a}^{x_b} p_s(x)\,\mathrm{d}x = \int_{\theta(x_a)}^{\theta(x_b)} p_g(y)\,\mathrm{d}y\,. \tag{2.27}$$

The cumulative distribution functions of input and output signals must hence be related by

$$\Pr\left[x \le x_a\right] = \int_{-\infty}^{x_a} p_s(x)\,\mathrm{d}x = \int_{-\infty}^{\theta(x_a)} p_g(y)\,\mathrm{d}y = \Pr\left[\theta(x) \le \theta(x_a)\right]. \tag{2.28}$$

Of particular interest is the case where the mapping characteristic shall result in a uniform distribution of the output in the amplitude range $0 \le x \le A_{max}$, which is a solution to maximize the contrast of an image signal. Here, $p_g(y) = 1/A_{max}$ such that $\Pr[y \le y_a] = y_a/A_{max}$.. Assuming that the input is restricted to amplitudes $0 \le x \le A_{max}$, using (2.28) gives the mapping characteristic

$$\frac{\theta(x)}{A_{\max}} = \int_0^x p_s(\xi)\,\mathrm{d}\,\xi \quad \Rightarrow \quad \theta(x) = A_{\max} \int_0^x p_s(\xi)\,\mathrm{d}\,\xi\,. \tag{2.29}$$

In principle, any PDF can be targeted for the output signal, but the solution is more complicated for the case of a non-uniform target, as the linear dependency of $\theta(x)$ on the left side of (2.29) would be replaced by an integral condition. These methods can similarly be adapted for the case of direct mapping into discrete probability distributions $\Pr(x_j)$, which could be interpreted as a design of a non-uniform quantizer characteristic. Such a quantizer would not be optimized for *minimum distortion*, but rather for *maximization of contrast*, or *maximization of sample entropy* in the output[12].

Once the discrete-to-discrete mapping is determined, its processing can efficiently be implemented using a lookup table (LUT). Furthermore, the general approach is not restricted to the case of single (monochrome) components. Nonlinear mapping functions can likewise be defined for input/output relationships of color triplets, or even LUT mapping from a monochrome input to a multi-component (color) output can be applied[13].

2.3 Interpolation

In signal analysis and signal output presentation, it is often necessary to generate values of signals *between available sampling positions*. The ultimate goal of this interpolation is the reconstruction of a continuous signal from discrete samples, which is perfectly possible in case of band-limited signals, following the condition that sampling should be performed with a rate that is at least double the bandwidth of the signal; otherwise, it can be interpreted as an estimation problem. The known positions will subsequently be denoted as control positions, which can in principle be defined for arbitrary sampling locations, including cases with non-equidistant positions. With irregular sampling (Fig. 2.13b), it is necessary to describe the actual positions $t(n)$, whereas in the regular (or time invariant) case (Fig. 2.13a) $t(n)=nT$, where T is the sampling distance. In the irregular case, reconstruction by high-quality lowpass filters[14] is not feasible, even if the sampled signal would have been band limited. Further, irregular sampling does not have a

---

[12] Note that, when such a mapping is defined with an input that already has discrete amplitudes, the output may not be approximated as precise uniformly distributed, unless the number of quantization levels J_i of the input is much higher than the number J_o for the output, and none of the input amplitudes has a probability larger than $1/J_o$.

[13] Determining optimum mapping functions for a given signal in such cases is often using an approach of *vector quantization* (cf. [MSCT, SEC. 4.5]), which is highly similar to k-means clustering (Sec. 5.6.4).

[14] For a broader background on the interpretation of interpolation using lowpass filters, refer to [MSCT, SEC. 2.8.1] or (A.27)ff.

straightforward spectral interpretation, the occurrence of alias is random rather than being deterministic.

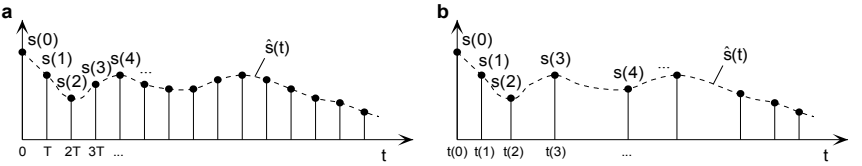

Fig. 2.13. Interpolation of a signal in cases of equidistant (*a*) and non-equidistant (*b*) positions of support.

The elements to be combined for the computation of the interpolation result need not necessarily be original samples from the signal. In a generalized approach, these can be considered as *control coefficients* $c(n)$. If the interpolation is defined piecewise as in (2.33) and (2.34), the continuous basis functions used for the interpolation are shift variant and valid only at one local position. If $P+1$ control coefficients are involved in the interpolation at any position t, the interpolation system has order P. This generic formulation of interpolation can be expressed for the one-dimensional case as

$$\hat{s}(t) = \sum_n c(n)\phi_n(t) \; . \tag{2.30}$$

$\phi_n(t)$ is a member from a system of continuous basis functions, which is weighted by coefficient $c(n)$ for its contribution to the interpolation at position t. In simplest case, the available samples are directly used as $c(n)$; in some of the interpolation methods described hereafter, control coefficients or additional parameters have to be computed first. Note that in general, the summation in (2.30) could have finite or infinite order (in terms of number of the control coefficients used to determine the interpolated result at position t), and finite or infinite support (in terms of the length of the basis functions).

It shall further be noted that in many applications of interpolation used in the context of this book, it is not necessary to generate continuous functions; moreover, interpolation is often performed for the purpose of generating more or alternative discrete positions, which means that (2.30) only needs to be computed by using sampled values from the functions $\phi_n(t)$.

2.3.1 Zero and first order interpolation basis functions

Let values of the signal $s(t)$ be known at control positions $t(n)$. Two very simple interpolation functions which can directly use available samples as control coefficients for the case of regular sampling $t(n)=nT$ are the *zero-order hold functions*

$$\phi_n^{(0)}(t) = \mathrm{rect}\left(\frac{t}{T} - n - \frac{1}{2}\right) \quad \text{or} \quad \phi_n^{(0)}(t) = \mathrm{rect}\left(\frac{t}{T} - n\right), \tag{2.31}$$

where the left-hand version can also be denoted as nearest-neighbor interpolator, and the *first-order linear interpolation function*

$$\phi_n^{(1)}(t) = \Lambda\left(\frac{t}{T} - n\right) \text{ with } \Lambda(t) = \begin{cases} 1 - |t|, |t| < 1, \\ 0, |t| \geq 1. \end{cases} \tag{2.32}$$

These basis functions are shown in Fig. 2.14. Examples of interpolation using the nearest neighbor and linear interpolation are shown in Fig. 2.15a/b. The 'order' of the interpolator refers to the number of basis functions that are superimposed to generate an interpolated sample; for the zero-order interpolation, only one basis function is involved; for the first-order interpolation, two; etc.

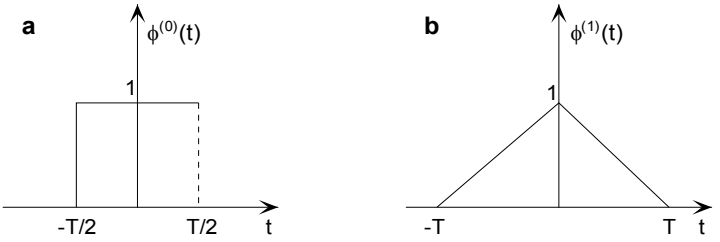

Fig. 2.14. Interpolation functions of order $P = 0$ (case of nearest neighbor interpolation) and $P = 1$ for the case of a regular distance of sampling positions

Zero-order hold and linear interpolation both violate the conditions for perfect reconstruction even in the regular sampling case due to the fact that the Fourier transfer functions of the basis functions are not flat in the range up to half sampling rate $1/(2T)$, and that they are not zero for frequencies beyond that point; this introduces alias (appearing as edges and corners, i.e. discontinuous signal behavior) in the interpolated signal. Nevertheless, both concepts can straightforwardly be extended to the case of irregular sampling, where for the zero-order case (Fig. 2.15c)[15]

$$\begin{aligned} &\phi_n^{(0)}(t) = \mathrm{rect}\left(\frac{t - t(n)}{t(n+1) - t(n)} - \frac{1}{2}\right) \\ &\text{or} \quad \phi_n^{(0)}(t) = \varepsilon\left(t - t(n) - \frac{t(n-1) - t(n)}{2}\right) - \varepsilon\left(t - t(n) - \frac{t(n+1) - t(n)}{2}\right), \end{aligned} \tag{2.33}$$

and for the first-order case (Fig. 2.15d)

[15] The upper case is holding the value $c(n)$ until the subsequent value becomes effective; the lower case spreads the value half-way towards both the previous and the next sampling positions, which is nearest neighbor interpolation.

$$\phi_n^{(1)}(t) = \begin{cases} \big[t - t(n-1)\big] \big/ \big[t(n) - t(n-1)\big] & \text{for } t(n-1) < t \le t(n), \\ \big[t - t(n+1)\big] \big/ \big[t(n) - t(n+1)\big] & \text{for } t(n) < t \le t(n+1), \\ 0 \text{ else.} \end{cases} \tag{2.34}$$

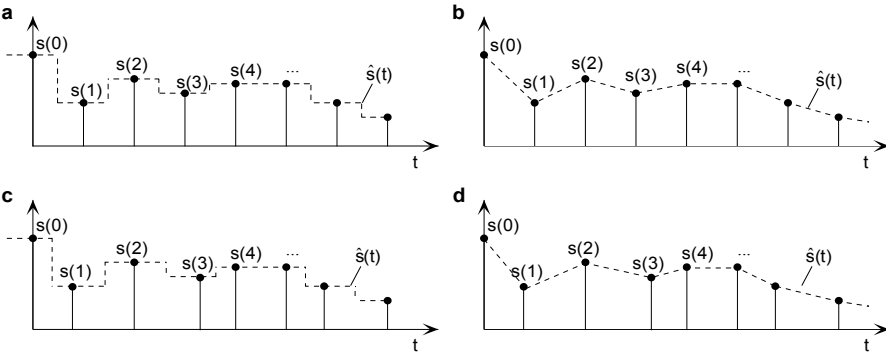

Fig. 2.15. Interpolation using systems **a/c** of order zero (nearest-neighbor hold element) **b/d** of order one (linear interpolator) [**a/b** equidistant samples, **c/d** non-equidistant samples]

2.3.2 LTI systems as interpolators

In case when linear time invariant systems are used for interpolation, the function $\phi_n(t)=h(t-nT)$ is defined via an impulse response of a lowpass filter. Due to time invariance of the impulse response, this approach is only applicable in the context of regular grids, following the principle of the sampling theorem, where the filter shall in ideal case completely suppress the periodic alias spectra which occur in sampling, and shall have a flat response within band limit $1/(2T)$, i.e. half of the sampling rate. Such an *ideal interpolator* is the *sinc function* (A.30) which has an infinite impulse response and perfect lowpass property in the frequency domain, and also allows direct usage of samples as weights $c(n)$, where

$$\phi_n^{(\infty)}(t) = \mathrm{si}\big(\pi t / T - n\big) \quad \text{with} \quad \mathrm{si}\big(x\big) = \frac{\sin x}{x} \text{ and } \mathrm{si}(0) = 1\,. \tag{2.35}$$

This interpolator cannot be practically realized, requiring an infinite number of samples to be fed into a non-causal filter; it has furthermore the disadvantage that the sinc function exhibits significant negative side lobes, which causes ringing in the interpolated signals when discontinuities (such as edges in images) are present.

Finite-length modifications of the sinc function can still provide close-to-optimum interpolation of finite order, and often even reduce the ringing at discontinuities. One common approach is *windowing*, i.e. multiplying the infinite basis by a finite function $w(t)$, such that $\phi_n^{(P_W)}(t) = \phi_n^{(\infty)}(t) w(t-n)$, which relates to a convolution of a perfect lowpass (rectangle) frequency transfer by the spectrum

$W(f)$ of the window in the Fourier domain. Windowing functions are often designed such that ripples in the Fourier transfer function of the interpolation filter are avoided, but a reasonably sharp cut-off transition is still preserved. An example is the *Hamming window* of duration D,

$$w(t) = \left[0.54 + 0.46 \cos\left(\frac{2\pi t}{D} \right) \right] \mathrm{rect}\left(\frac{t}{D} \right). \tag{2.36}$$

Depending on the choice of D in relation with the sampling distance T, the finite order P_W of the interpolator can be adjusted. Another example is the Lanczos filter proposed in [DUCHON 1979]. Its continuous impulse response is defined as

$$\phi_n^{(P_W)}(t) = \mathrm{si}\left(\frac{\pi t}{T} - n \right) \mathrm{si}\left(\frac{\pi t}{2DT} - n \right) \mathrm{rect}\left(\frac{t}{D} - n \right). \tag{2.37}$$

The multiplication of the two sinc functions in t provides a rectangle with linear roll-off (trapeze with linear ramps of width $1/D$) as frequency transfer. This is further convolved by a sinc (due to the rectangular cutoff in t) in the frequency domain, which however is less critical than in the case of ideal interpolation, since the transition from pass band to stop band is more smooth with the trapeze.

Optimized LSI interpolation functions, with adaptation to properties of the signal to be interpolated, can also be obtained from Wiener filter designs (see Sec. 3.3.2). These can further be optimized to avoid influence of additive noise which may be present in the samples used for interpolation.

2.3.3 Spline, Lagrangian and polynomial interpolation

When the original samples from the signal are directly used as weighting coefficients, i.e. $c(n) = s[t(n)]$, the following condition must hold for any pair (m,n)[16]:

$$\hat{s}\left[t(n)\right] = \sum_m s\left[t(m)\right] \cdot \phi_m(t) \overset{!}{=} s\left[t(n)\right] \quad \Rightarrow \phi_m(t(n)) = \delta(n-m), \tag{2.38}$$

which means that at a given sampling position, there shall be no influence of any sample from other sampling positions, and the sample's own weight shall be unity to make the interpolated value coincide with the original signal value at the known sampling positions. This condition is fulfilled for any interpolation function introduced in Sec. 2.3.1 and 2.3.2. It is also fulfilled by *Lagrangian interpolation* with basis functions

[16] The expression $\delta(k)$ used in the subsequent equation is the *Kronecker Delta* with $\delta(k)=1$ for $k=0$, $\delta(k)=0$ for $k\neq 0$.

$$\phi_n^{(P)}(t) = \prod_{\substack{m \neq n \\ |m-n| \leq P}} \frac{t - t(m)}{t(n) - t(m)}. \tag{2.39}$$

The Lagrangian basis can again be defined using a finite support, when limiting the value range of m in the product to values in the closer neighborhood of n, such that $\phi_n(t) = 0$ for values t which are farther away from $t(n)$. In case of P=1, limiting the support to the two values which are closest to t, $\varphi_n(t)$ becomes identical to linear interpolation (Fig. 2.15b/d).

For the case of equidistant sampling $t(n)=nT$, Lagrangian interpolation is shift invariant with $\phi_n(t) = \phi_0(t - nT)$. In the case of an infinite series of samples,

$$\phi_0(t) = \prod_{m \neq 0} \frac{t - mT}{-mT} = \prod_{m=1}^{\infty} \left(1 - \frac{t}{mT}\right)\left(1 + \frac{t}{mT}\right) = \prod_{m=1}^{\infty} \left(1 - \left(\frac{t}{mT}\right)^2\right). \tag{2.40}$$

Then, with the following product expansion of a sine function, the Lagrangian basis function for this specific case translates into the sinc function,

$$\sin(x) = x \cdot \prod_{m=1}^{\infty} \left(1 - \left(\frac{x}{m\pi}\right)^2\right) \Rightarrow \phi_0(t) = \mathrm{si}\left(\frac{\pi t}{T}\right). \tag{2.41}$$

Like the sinc function, the Lagrangian interpolation basis can become negative for $P>1$, which may cause ringing (overshoots or oscillations) at discontinuities. To prevent such effects, smooth and non-negative interpolation functions would be more favorable, which however can lead to a violation of (2.38), such that available samples $s[t(n)]$ cannot directly be used as weights $c(n)$ in (2.53).

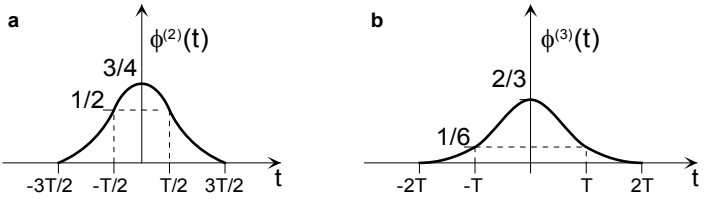

Fig. 2.16. Centered *B*-spline functions of orders $P = 2$ (**a**) and $P = 3$ (**b**) for the case of a regular grid of control positions

Basis splines (denoted as B-spline) are an example for smooth and non-negative interpolation functions. They can best be interpreted through a recursive development starting from a zero-order hold function[17],

[17] The following general equation refers to a function which starts at $t(n)$ and ends at $t(n+P+1)$. To make them center aligned, as shown in Figs. 2.14/2.16 and subsequent equations, it is necessary to apply a shift, e.g., by $t(n+P+1)-t(n)$ for the equidistant sampling case.

$$\phi_n^{(P)}(t) = \frac{t - t(n)}{t(n+P) - t(n)} \phi_n^{(P-1)}(t) + \frac{t(n+P+1) - t}{t(n+P+1) - t(n+1)} \phi_{n+1}^{(P-1)}(t). \tag{2.42}$$

In the case of regular sampling $t(n) = nT$, the B-spline of P^{th} order would be constructed by P-fold convolution of rectangles of width T (i.e. $P+1$ rectangles are convolved). This gives the zero-order hold element (2.33) for $P=0$ and the linear interpolator (2.34) for $P=1$. For $P=2$ and $P=3$, the quadratic and cubic B-spline functions are defined as (Fig. 2.16)

$$\phi_n^{(2)}(t) = \begin{cases} \frac{3}{4} - \left(\frac{t - t(n)}{T} \right)^2 & \text{for } |t - t(n)| \le \frac{T}{2}, \\ \frac{\left(1.5 - |(t - t(n))/T|\right)^2}{2} & \text{for } \frac{T}{2} < |t - t(n)| \le \frac{3}{2}T, \\ 0 \text{ for } |t - t(n)| > \frac{3}{2}T, \end{cases} \tag{2.43}$$

$$\phi_n^{(3)}(t) = \begin{cases} \frac{4 + 3|(t - t(n))/T|^3 - 6|(t - t(n))/T|^2}{6} & \text{for } |t - t(n)| \le T, \\ \frac{\left(2 - |(t - t(n))/T|\right)^3}{6} & \text{for } T < |t - t(n)| \le 2T, \\ 0 \text{ for } |t - t(n)| > 2T. \end{cases} \tag{2.44}$$

For $P \to \infty$, the iterative convolution of rectangles converges into a Gaussian function. For the case of irregular sampling, the functions are stretched according to the distances between the samples in the respective section.

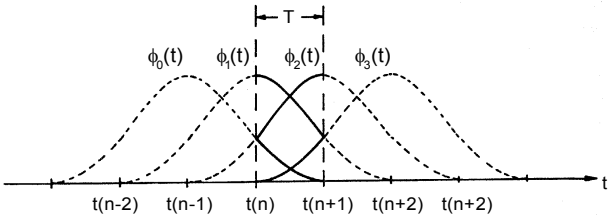

Fig. 2.17. Cubic spline interpolation and basis functions which are combined for interpolation within an interval $t(m) \le t < t(m+1)$

Fig. 2.17 illustrates the computation of a cubic spline interpolation ($P = 3$) within an interval $t(n) \le t < t(n+1)$. A total of 4 control coefficients $c(m)$, $n-1 \le m \le n+2$ is used to weight the respective interpolation functions, which have their centers at $t(n-1)$, $t(n)$, $t(n+1)$ and $t(n+2)$. The result of interpolation is determined from

the general formulation (2.30), using the basis function (2.44). The argument of the interpolation function is expressed here as $t' = t - t(n)$. Further, normalization of sampling distance $T = 1$ is assumed. This gives the interpolated result within the respective range $t(n) \le t < t(n+1)$, mapped into $0 \le t' < 1$,

$$\begin{aligned}\hat{s}(t) &= c(n-1)\frac{\big(2-(t'+1)\big)^3}{6} + c(n)\frac{4+3t'^3-6t'^2}{6} \\ &\quad + c(n+1)\frac{4-3\big(t'-1\big)^3-6\big(t'-1\big)^2}{6} + c(n+2)\frac{\big(2+(t'-2)\big)^3}{6} \\ &= c(n-1)\frac{-t'^3+3t'^2-3t'+1}{6} + c(n)\frac{3t'^3-6t'^2+4}{6} \\ &\quad + c(n+1)\frac{-3t'^3+3t'^2+3t'+1}{6} + c(n+2)\frac{t'^3}{6}.\end{aligned} \tag{2.45}$$

(2.45) can be re-written by the following matrix expression:

$$\hat{s}(t) = \frac{1}{6}\begin{bmatrix} t'^3 & t'^2 & t' & 1 \end{bmatrix}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}\begin{bmatrix} c(n-1) \\ c(n) \\ c(n+1) \\ c(n+2) \end{bmatrix}. \tag{2.46}$$

The remaining problem is optimization of the control coefficients $c(n)$. At the known sampling positions, $t' = 0$, such that

$$\hat{s}\big[t(n)\big] = \frac{1}{6}\big[c(n-1) + 4c(n) + c(n+1)\big]. \tag{2.47}$$

The condition

$$\hat{s}\big[t(n)\big] \overset{!}{=} s\big[t(n)\big] \tag{2.48}$$

can now be used to determine the $c(n)$ values. This is an underdetermined problem if only one condition (2.47) is used. As however any coefficient $c(n)$ takes influence on the interpolation within four different intervals and three existing sampling positions, a unique solution becomes possible. Optimizing all control coefficients jointly over a finite signal segment with M sampling positions, the result of interpolation at these positions can be written by the following vector-matrix expression, assuming a circular (periodic) extension of the sequence of control coefficients here:

$$\begin{bmatrix} \hat{s}\big[t(1)\big] \\ \hat{s}\big[t(2)\big] \\ \hat{s}\big[t(3)\big] \\ \vdots \end{bmatrix} = \frac{1}{6}\begin{bmatrix} 4 & 1 & 0 & \cdots & 0 & 1 \\ 1 & 4 & 1 & 0 & & 0 \\ 0 & 1 & 4 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \end{bmatrix}\begin{bmatrix} c(1) \\ c(2) \\ c(3) \\ \vdots \end{bmatrix} \tag{2.49}$$

$$\underbrace{\begin{bmatrix} \\ \hat{s}[t(M)] \end{bmatrix}}_{\hat{\mathbf{s}}} \quad \underbrace{\begin{bmatrix} 0 & & 0 & 1 & 4 & 1 \\ 1 & 0 & \cdots & 0 & 1 & 4 \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} \\ c(M) \end{bmatrix}}_{\mathbf{c}}.$$

Using (2.48), the control coefficients can now be computed by multiplication of the inverted matrix **H** from (2.49) by the vector of original signal values:

$$\begin{bmatrix} c(1) \\ c(2) \\ c(3) \\ \vdots \\ \\ c(M) \end{bmatrix} = 6 \begin{bmatrix} 4 & 1 & 0 & \cdots & 0 & 1 \\ 1 & 4 & 1 & 0 & & 0 \\ 0 & 1 & 4 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ 0 & & 0 & 1 & 4 & 1 \\ 1 & 0 & \cdots & 0 & 1 & 4 \end{bmatrix}^{-1} \begin{bmatrix} s[t(1)] \\ s[t(2)] \\ s[t(3)] \\ \vdots \\ \\ s[t(M)] \end{bmatrix}. \tag{2.50}$$

Alternatively, it is also possible to reduce the number of control coefficients compared to the alternative signal samples. In that case, the matrix **H** would no longer be square, and instead of the inversion the *pseudo inverse*, which is the best approximation of the given information from the series of samples in terms of a least-squares fitting optimization can be used (see section 3.4).

Another solution guaranteeing reproduction of available samples is *polynomial interpolation*. The assumption is that the signal can be approximated from a P^{th} order polynomial, i.e.

$$\hat{s}(t) = \alpha_P t^P + \alpha_{P-1} t^{P-1} + \ldots + \alpha_1 t + \alpha_0 \,. \tag{2.51}$$

To determine the coefficients α_p, at least P+1 samples ned to be known to solve the following equation system[18], i.e.

$$\hat{s}(t) = s(t) \;\text{ for }\; t = t(n)$$

$$\Rightarrow \begin{bmatrix} s[t(n-P/2)] \\ \vdots \\ s[t(n)] \\ \vdots \\ s[t(n+P/2)] \end{bmatrix} \overset{!}{=} \begin{bmatrix} t_{n-P/2}{}^P & t_{n-P/2}{}^{P-1} & \cdots & t_{n-P/2} & 1 \\ \vdots & & & & \vdots \\ t_n{}^P & t_n{}^{P-1} & \cdots & t_n & 1 \\ \vdots & & & & \vdots \\ t_{n+P/2}{}^P & t_{n+P/2}{}^{P-1} & \cdots & t_{n+P/2} & 1 \end{bmatrix} \begin{bmatrix} \alpha_P \\ \alpha_{P-1} \\ \vdots \\ \alpha_1 \\ \alpha_0 \end{bmatrix}. \tag{2.52}$$

Polynomial interpolation again does not necessarily guarantee smooth behaviour of the interpolation function between the samples, particularly if the order P has a high value, and if significant amplitude variations occur in the set of observed

[18] In the given example, an odd number of samples is assumed around some given center position $t(n)$

samples. Polynomial interpolation can also be applied segment-wise over pieces of a longer extended signal; if the available samples used in adjacent segments are partially shared, this diminishes the possibility that discontinuities (edges) would occur at segment boundaries.

2.3.4 Interpolation on 2D grids

For the case of 2D interpolation, (2.30) can be extended as follows:

$$\hat{s}(t_1,t_2)=\sum_{n_1}\sum_{n_2}c(n_1,n_2)\phi_{n_1,n_2}(t_1,t_2)\,. \tag{2.53}$$

In case of regular grids, or generally when no interdependency between the sampling positions in t_1 and t_2 exists, it is possible to use separable basis functions,

$$\phi_{n_1,n_2}(t_1,t_2)=\phi_{n_1}(t_1)\,\phi_{n_2}(t_2)\,. \tag{2.54}$$

As an example, the 2D separable version of (2.32) is *bilinear interpolation*. The principle is illustrated in Fig. 2.18. The value to be estimated at position (t_1,t_2) is computed from samples of four neighboring positions, which are weighted depending on the horizontal and vertical fractional distances d_1 and d_2 (normalized by the sampling distances):

$$\begin{aligned}\hat{s}(t_1,t_2)=&\,s(n_1,n_2)(1-d_1)(1-d_2)+s(n_1+1,n_2)d_1(1-d_2)\\&+s(n_1,n_2+1)(1-d_1)d_2+s(n_1+1,n_2+1)d_1d_2\end{aligned}$$

$$\text{with}\quad d_i=\frac{t_i}{T_i}-n_i,\ n_i=\left\lfloor\frac{t_i}{T_i}\right\rfloor. \tag{2.55}$$

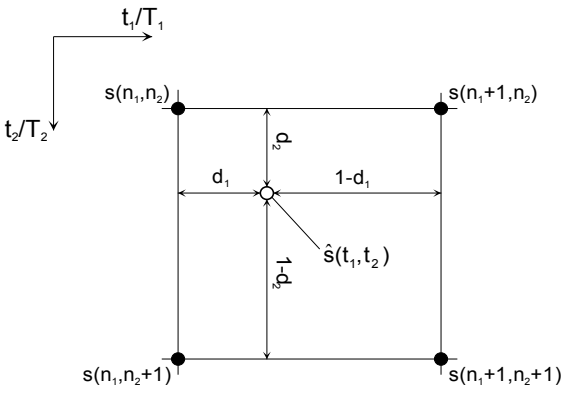

Fig. 2.18. Bilinear interpolation

In case of irregular grids, it has to be identified which are the closest sample positions to be used for each interpolated position. An efficient approach for doing

this with regard to Euclidean distance are *Delaunay triangulation* and the associated *Voronoi net*, as illustrated in Fig. 2.19. The *Delaunay net* is established by the interconnection lines between control positions, and whenever two interconnection lines would intersect, the longer of these is discarded[19]. This way, an ensemble of triangular patches is constructed, bounded by the Delaunay lines which are as short as possible. Dividing them by half and drawing perpendicular lines (denoted as Voronoi lines) provides the Voronoi net, with corner points that define the centers of circumscribing circles of the triangles. The *Voronoi regions* which are bounded by the Voronoi lines establish areas which are closest to the corresponding control position. In case of zero-order hold (nearest neighbor) interpolation, all samples falling into the same Voronoi region would inherit the amplitude of its centroid sample[20]. The topology of the Delaunay net can further be used to identify whether additional control positions are close enough to be considered in the interpolation at a given position.

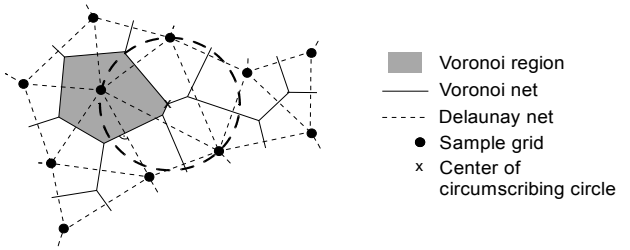

Fig. 2.19. Sample grid, Delaunay triangulation and Voronoi diagram

One possible approach for 2D interpolation from a finite set of nearest-neighbored irregularly positioned samples is *reciprocal distance weighting*, where the influence of a sample is reduced when it is farther away from the position **t** to be interpolated. Let the Euclidean distance between $\mathbf{t}=[t_1,t_2]^{\mathrm{T}}$, and one of the P nearest neighbor positions p, $p = 1, \ldots, P$, ($P = 4$ in the example shown in Fig. 2.20) be defined as

$$\mathbf{d}_p = \begin{bmatrix} t_1 - t_1(p) & t_2 - t_2(p) \end{bmatrix}^{\mathrm{T}} \quad \Rightarrow \left\| \mathbf{d}_p \right\| = \sqrt{\mathbf{d}_p{}^{\mathrm{T}} \mathbf{d}_p} \,. \tag{2.56}$$

The interpolated value is then computed as

$$\hat{s}(t_1,t_2) = \frac{\sum_{p=1}^{P} \frac{s\left[t_1(p),t_2(p)\right]}{\left\| \mathbf{d}_p \right\|}}{\sum_{p=1}^{P} \frac{1}{\left\| \mathbf{d}_p \right\|}} \,. \tag{2.57}$$

---

[19] For efficient methods of computing Delaunay nets, the reader is referred to [CHENG, DEY, SHEWCHUK 2012].

[20] In separable sampling, Voronoi regions would be rectangles of size $T_1 T_2$.

At sample positions, the interpolation is not necessary, and only one closest sample's amplitude will become dominant for any $\|\mathbf{d}_p\| \to 0$. A simple check is also possible whether a position to be interpolated is included within the area of the polygon spanned by the current set of control positions. It is necessary that the difference vector $\mathbf{d}_p$ relating to a control position is right-sided of a vector $\mathbf{v}_p$, which clock-wise connects the current control position with the next, such that

$$\begin{aligned} \mathbf{v}_p &= \left[t_1\left(\mathrm{mod}(p,P)+1\right) - t_1(p) \quad t_2\left(\mathrm{mod}(p,P)+1\right) - t_2(p) \right]^{\mathrm{T}} \\ &\Rightarrow \det\left|\mathbf{v}_p \quad \mathbf{d}_p\right| \overset{!}{\geq} 0, \quad p = 1,...,P. \end{aligned} \tag{2.58}$$

(2.58) can also be used to determine the set of suitable control positions for a position $\mathbf{t}$, but it should also be observed that all $\|\mathbf{d}_p\|$ values are small enough (as would implicitly be the case for Delaunay triangulation)

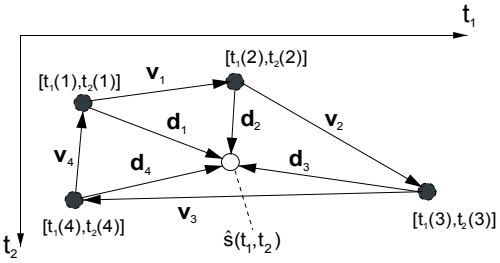

Fig. 2.20. Interpolation of a position $\hat{s}(t_1,t_2)$ from the vertices of a quadrangular polygon

Alternative approaches for interpolation on irregular 2D grids are based on 2D extensions of polynomial fitting (2.51), such that a 2D surface function is determined by the amplitudes of the available control positions. After performing a Delaunay triangulation, the interpolation within each triangle can be performed as

$$\hat{s}(t_1,t_2) = \alpha_0 + \alpha_1 t_1 + \alpha_2 t_2 \,. \tag{2.59}$$

To determine the coefficients α_i, the following equation system relating to the control position amplitudes (which are the triangle's vertices or corners) has to be solved for $\mathbf{a}=\mathbf{A}^{-1}\mathbf{s}$[21]:

$$\underbrace{\begin{bmatrix} s\left[t_1(1),t_2(1)\right] \\ s\left[t_1(2),t_2(2)\right] \\ s\left[t_1(3),t_2(3)\right] \end{bmatrix}}_{\mathbf{s}} = \underbrace{\begin{bmatrix} 1 & t_1(1) & t_2(1) \\ 1 & t_1(2) & t_2(2) \\ 1 & t_1(3) & t_2(3) \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{bmatrix}}_{\mathbf{a}} . \tag{2.60}$$

[21] The solution is simplified when one of the control positions (e.g. $p=1$) is defined as (0,0), and the other coordinates of control positions are adjusted accordingly as $t_i'(p)=t_i(p)-t_i(1)$ in (2.62), and the interpolated positions expressed as $t_i'=t_i-t_i(1)$ in (2.61).

Within each triangle, the interpolated amplitude surface is describing a planar equation according to (2.59). When four control positions are used, the method can be extended into *bilinear mapping*[22]

$$\hat{s}(t_1,t_2) = \alpha_0 + \alpha_1 t_1 + \alpha_2 t_2 + \alpha_3 t_1 t_2 \ . \tag{2.61}$$

Here, the equation system for determining the coefficients α is

$$\underbrace{\begin{bmatrix} s[t_1(1),t_2(1)] \\ s[t_1(2),t_2(2)] \\ s[t_1(3),t_2(3)] \\ s[t_1(4),t_2(4)] \end{bmatrix}}_{\mathbf{s}} = \underbrace{\begin{bmatrix} 1 & t_1(1) & t_2(1) & t_1(1)t_2(1) \\ 1 & t_1(2) & t_2(2) & t_1(2)t_2(2) \\ 1 & t_1(3) & t_2(3) & t_1(3)t_2(3) \\ 1 & t_1(4) & t_2(4) & t_1(4)t_2(4) \end{bmatrix}}_{\mathbf{T}} \underbrace{\begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix}}_{\mathbf{a}} . \tag{2.62}$$

This method is further extensible to use higher order surface-fitting polynomials. However, increasing the number of control positions may not be the best choice, since they typically would be at farther distance from the position to be interpolated. An alternative is *bi-cubic interpolation*, which additionally requires knowledge about the derivatives in horizontal, vertical and both directions at the four control positions (for discrete approximation of derivatives, see Sec. 4.3.1). The interpolation is performed as[23]

$$\hat{s}(t_1',t_2') = \sum_{j_1=0}^{3} \sum_{j_2=0}^{3} \alpha_{j_1 j_2} (t_1')^{j_1} (t_2')^{j_2} \ . \tag{2.63}$$

The coefficients $\alpha_{j_1 j_2}$ can be obtained by computing the partial derivatives of (2.63)

$$\frac{\partial}{\partial t_1} \hat{s}(t_1',t_2'), \frac{\partial}{\partial t_2} \hat{s}(t_1',t_2'), \frac{\partial^2}{\partial t_1 \partial t_2} \hat{s}(t_1',t_2') \,, \tag{2.64}$$

and then use the known values at the four control positions, as well as the corresponding derivatives at those positions, to solve an equation system with 16 unknowns to determine the coefficients[24]. This approach of bi-cubic interpolation can also be applied on regular sampling grids, then typically giving improved interpolation results over (2.55).

It should be kept in mind that all interpolation methods introduced here are based on the assumption of smoothness of the signal to be interpolated and will

---

[22] In case of separable sampling, this is equivalent to (2.55).

[23] Using normalized coordinates t' according to the footnote on p. 34.

[24] Note that for computation of the derivatives, it may not be sufficient to use only the samples at the four control positions; better accuracy and independence against noise should be achieved when additional samples from the neighborhood are taken into consideration. The usage of gradients, which are likewise applied in neighbored polygonal patches, guarantees for smooth transition beyond patch boundaries.

fail in case of discontinuities (e.g., edges in images). For reconstruction of discontinuity positions with sub-sample accuracy, or on irregular sampling grids, additional assumptions are necessary such as straightness or smooth curvedness of an edge contour over a 2D field. This can be achieved by matching the given set of samples against a model basis function (cf. Sec. 4.3.4).

Higher-resolution images can also be generated from a series of pictures of low resolution showing the same content but with slightly different sampling positions. For an overview on *super-resolution methods*, see [PARK, PARK AND KANG 2003]. The aggregation of information from the samples requires a registration (practically done by correspondence analysis, cf. Sec. 3.9), combination into a non-uniform grid of samples, and transformation into a uniform grid of higher resolution. Simple methods use direct interpolation from the non-uniform grid, or estimation based methods such as constrained least squares, maximum a posteriori or projection onto convex sets, where the latter typically provide better quality. The quality of correspondence estimation is crucial for the reliability of any of the methods, and care should be taken that super-resolution information is only generated in cases where the identical area is safely available in multiple shots of the same scene in a set of different pictures (or a sequence of video pictures taken by a moving camera).

2.4 Multi-resolution representation

In signal analysis, multi-resolution representations (also denoted as scale-space representation) are widely used, providing both reliability and efficiency. The basic assumption is that the appearance of certain features is invariant against observation in different resolutions (scales). It should be noted here that typically such representations are *overcomplete* in the sense that they consist of more samples than the original signal. This implies a certain amount of redundancy, which however is not harmful for the goal of reliable analysis, but should be avoided as far as possible, since it might cause unreasonable processing complexity. On the other hand, when downsampling is applied in the scales representing lower frequencies, the analysis could be made even more efficient, as it can possibly be avoided to consider corresponding positions in higher scales with more samples, if the result is already conclusive from the lower scale.

The general principle of generating such a representation is shown for the case of 2D signals in Fig. 2.21a. The scaled signal $s_{u-1}(n)$ is generated from $s_u(n)$ by low pass filtering and downsampling by a factor $|\mathbf{U}_u|$ (representing the determinant of a sampling matrix in the multi-dimensional case, see (A.57)). The simplest case uses separable downsampling by factors of two (per dimension), which gives a dyadic 'pyramid' representation (indicated by solid lines). For signal analysis, finer steps between the scales may be desirable; basically, if the downsampling factor shall be non-integer (dotted lines), the filter needs to include phase shifts by

using an appropriate set of sub-sample interpolation filters[25]. However, the more scales are added, the more overcomplete the entire representation becomes.

Alternatively, to avoid non-dyadic re-sampling, it is also possible to use several lowpass filters with different cutoff within each dyadic scale, even though this adds even more over-completeness (since the sampling rate is higher than double cutoff frequency for the additional intermediate scales). A corresponding scheme is shown in Fig. 2.21b.

Generally, filters with Gaussian shaped impulse response (or a sampled equivalent) are advantageous, since the iterated convolution of Gaussians again results in a wider Gaussian. Therefore, these types of scale-space representations are often denoted as *Gaussian pyramids*.

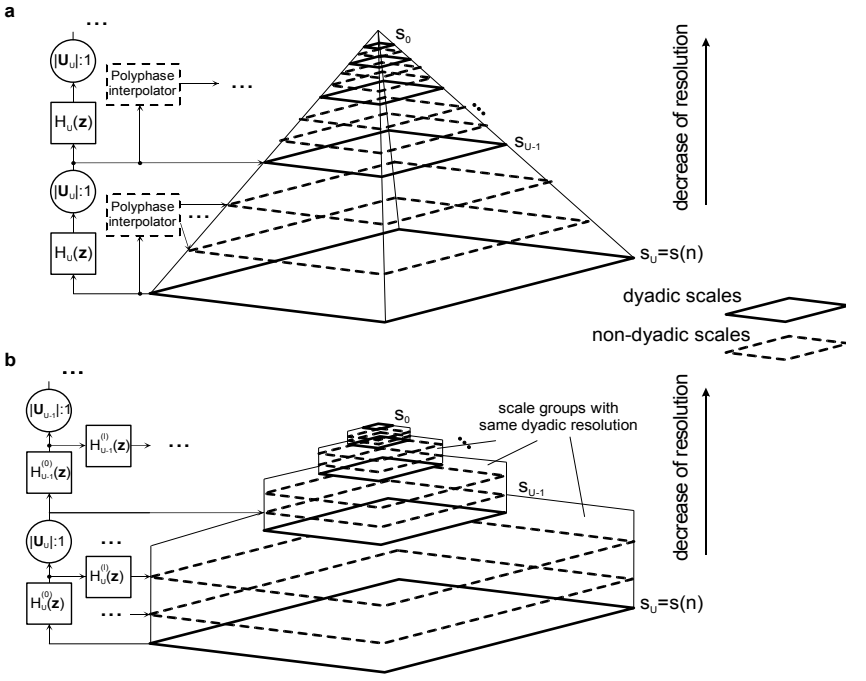

Fig. 2.21. Generation and illustration of a scale space representation **a** dyadic and non-dyadic scales with most compact (Nyquist rate) representation **b** Subsampling only done once per dyadic scale, with intermediate scales generated by additional lowpass filters

Multi-resolution pyramids establish a stack of representations of the signal with different bandwidth (lowpass frequency cutoff), where it can be expected that the same signal properties (or features) appear at collocated positions, but may give complementary clues; particularly, the low-resolution versions would not suffer from high-frequency noise, whereas the high resolution versions carry additional

[25] This can efficiently be implemented using polyphase filters, cf. [MSCT, SEC. 2.8.3]

information e.g. about sharpness and position of edges. Furthermore, complexity can be saved when processing is done in a downsampled resolution, and only refined at positions where relevant clues were found. An example for such an approach is hierarchical motion estimation (Sec. 4.6.3).

On the other hand, important information can also be contained in the additional detail carried in the higher-resolution representations. A common approach extracting this directly from the pyramid representation is by computing the difference between the signals at two adjacent scales. This usually requires upsampling (sample interpolation) for the lower resolution signal, which can be omitted within a group of scales with same downsampling resolution, as in the stacks of Fig. 2.21b. When the different scales are computed by using filters with Gaussian shape of impulse response (or spectral response), this is denoted as *Difference of Gaussian* (DoG) representation. The discrete filter coefficients are often derived by sampling from a continuous circular-symmetric 2D Gaussian function,

$$h_{\mathrm{G}}(t_1, t_2) = \frac{1}{\sqrt{2\pi\tau^2}} \mathrm{e}^{-\frac{t_1^2 + t_2^2}{2\tau^2}} . \tag{2.65}$$

The parameter τ influences the width of the Gaussian shape and thereby the strength of lowpass filtering. The second derivative of this function is the *Laplacian of Gaussian* (LoG)

$$\nabla^2_{\mathrm{G}}(t_1, t_2) = \frac{1}{\pi^2\tau^4}\left[2 - \frac{t_1^2}{\tau^2} - \frac{t_2^2}{\tau^2}\right] \mathrm{e}^{-\frac{t_1^2 + t_2^2}{2\tau^2}} . \tag{2.66}$$

This radial-symmetric function is also denoted as *Mexican hat filter*, of which a 1D section is shown in Fig. 2.22. For a discrete approximation, the center of the sampled impulse response would be at $t=0$ (maximum of the continuous function). The parameter τ could also be set to 1, and the strength of lowpass filtering then simply varied by changing the sampling distance, which is equivalent to scaling.

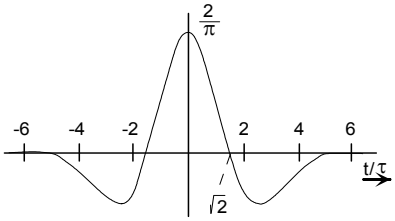

Fig. 2.22. 'Mexican hat' filter impulse response

In terms of the spectral behavior, DoG and LoG are similar. When designed in a way such that the bandwidths of the underlying Gaussian lowpass functions increase on a logarithmic scale, the corresponding amplitude transfer functions

represent a bank of bandpass filters with logarithmically increasing center frequencies and bandwidths, as shown in Fig. 2.23.

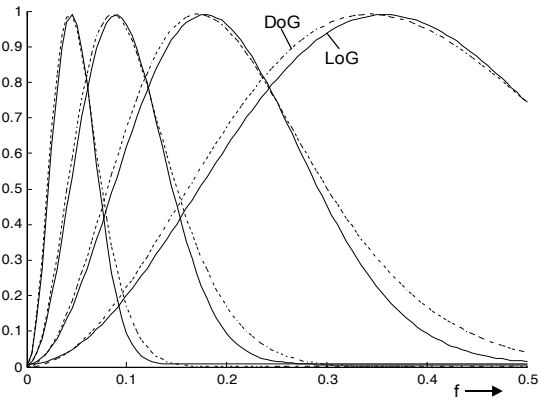

Fig. 2.23. Fourier amplitude transfer functions of DoG (---) and LoG (—), dyadic scale of Gaussian lowpass filter bandwidths

Alternatively, modulated functions can be used to perform bandpass analysis. One advantage compared to the DoG or LoG approaches is decreased overlap of the bandpass functions towards lower frequency (which happens particularly when the roll-off towards higher frequency is relatively flat, as in the Gaussian case). An approach that is based on modulated Gaussian functions with frequency maxima positioned on a logarithmic scale is *Gabor wavelet analysis*, where the underlying filter impulse responses are products of Gaussian hull functions and complex periodic exponentials. A general form can be written as

$$h_{f_C,b}(t) = \frac{1}{\sqrt{2\pi b^2}} \mathrm{e}^{-\frac{\tau^2}{2b^2}} \mathrm{e}^{j2\pi f_C t} \circ\!\!-\!\!\bullet\, H_{f_C,\sigma_t}(f) = \mathrm{e}^{-2\pi^2 b^2 (f-f_C)^2} \,, \tag{2.67}$$

where f_C relates to the center frequency, and b is a parameter to control the width of the Gaussian hull (or reciprocally the bandwidth of the filter). Gabor functions establish overcomplete, non-orthogonal basis sets; it can be shown that any band-limited infinite function $s(t)$ can be reconstructed exactly from an infinite series of discrete Gabor coefficients, when available at regularly-spaced sampling positions in both signal and frequency domains [GABOR 1946]. For cases of finite band-limited signals, the number of coefficients should then become finite as well. When Gabor functions shall be used with discrete positions on a *logarithmic* frequency axis, in order to implement a discrete wavelet transform (DWT) [MSCT, SEC. 2.8.4], the condition of constant log-scale spacing between two adjacent center frequencies, e.g. $\Delta = \log[f_C(k+1)/f_C(k)] = \text{const.}$ over all k has to be observed. For the example of dyadic spacing, which is often used in the context of a discrete wavelet transform, $\Delta = \log(2)$. In this case, the center frequencies will thus be positioned at

$$f_C(k) = f_0 \cdot 2^{(k-K)\Delta} , \tag{2.68}$$

where f_0 represents the center frequency of the highest-frequency bandpass filter with index $k=K$, and K is the total number of bandpass channels. Further, the bandwidth of bandpass channel k has to be proportional with the distances from its neighbored bands, and therefore also scales logarithmically with increased frequency, such that

$$b(k) = \frac{\beta}{f_C(k)} . \tag{2.69}$$

β is a constant which can be derived by additional criteria, such as maximum constancy of the sum over all bandpass functions, as necessary to retain all relevant information. In the case of dyadic band spacing, a factor of $\sqrt{2 \ln 2}$ has been reported to be suitable for that purpose [HALEY, MANJUNATH 1999]. However, unlike bi-orthogonal basis functions that are frequently used in the context of the DWT, these types of Gabor wavelets do not directly allow perfect reconstruction of the signal when operating with a finite number of coefficients (sampled in time and frequency).

Due to the complex basis, a separable 2D implementation of Gabor wavelets fully retains directional analysis properties. Examples of 2D bandpass impulse responses analyzing different scales (center frequencies) and orientations are shown in Fig. 2.24.

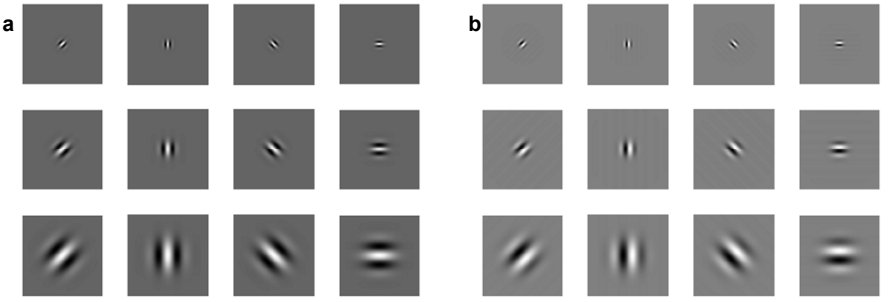

Fig. 2.24. Impulse responses of directionally oriented complex 2D Gabor bandpass functions with 4 orientations and 3 different scales **a** real part **b** imaginary part

The separable approach has however the disadvantage that the 2D combinations of Gaussian functions representing different scales (or center frequencies) with different bandwidth along the horizontal and vertical axes of the frequency domain result in circular or elliptic shapes of the corresponding 2D functions. The case of positioning center frequencies at discrete positions separately on the f_1 and f_2 axes is shown in Fig. 2.25a. This leads to a relatively irregular layout of combining scales and orientations, including the fact that for increasing frequencies the number of angular orientations increases as well, which may not be necessary

for analysis, and also may cause inconsistent analysis results when analysis shall be performed w.r.t. a certain angular orientation across various scale resolutions.

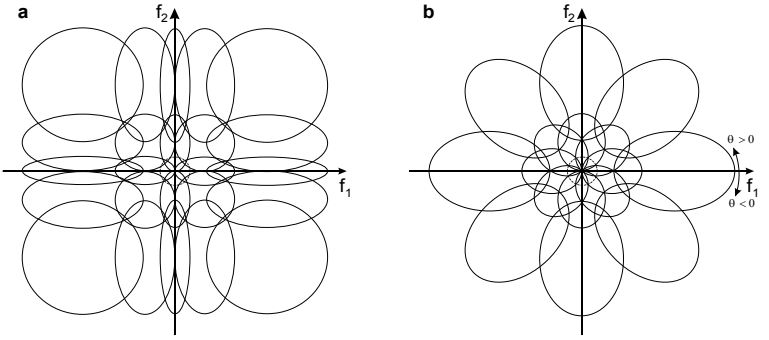

Fig. 2.25. Frequency layout of separable (**a**) and non-separable (**b**) Gabor wavelet transform

To allow for more consistent analysis, a separable definition along the dimensions of angle and scale (instead of horizontal and vertical) can be employed. This results in a polar-form representation of the 2D wavelet functions [LEE 1996], where the center frequencies of the bands are located in an octave-band schema at different radial (scale) orientations, and in addition a number of uniformly-spaced angular (directional) orientations is introduced at each scale, as shown in Fig. 2.25b. In the subsequent definition, the frequency f relates to the radial orientation (or the center frequency f_C which is the distance from the origin of the 2D frequency plane), and the angular parameter θ_C relating to the angular orientation (values of θ in the range of $-1/2 \ldots 1/2$ specifying angles between $-\pi/2$ and $\pi/2$ relative to the f_1 axis). The parameters b_ρ and b_θ reciprocally control the bandwidths in radial and angular orientations, respectively, of the Fourier transfer function

$$H_{\text{Polar}}(f,\theta) = H_{f_C,b_\rho}(f)\,\mathrm{e}^{-\frac{\pi^2 b_\theta^2 (\theta-\theta_C)^2}{2}} \quad \text{with } f = \sqrt{f_1^2 + f_2^2},\ \theta = \arctan\left(\frac{f_2}{f_1}\right), \quad (2.70)$$

with the radial 1D function over f as defined in (2.67). The corresponding complex impulse responses are again modulated 2D Gaussian hulls similar as in Fig. 2.24 but typically not of circular shape, where the oscillation propagates with frequency f_C along the direction $\pi\theta_C$. K discrete scale-related frequencies f_C are usually defined on a logarithmic scale as explained above. For the orientation, it is sufficient to define L discrete angles which are uniformly spaced by π/L, ranging from $-\pi/2$ to $\pi/2-\pi/L$ or alternatively ranging from 0 to $\pi-\pi/L$. This only covers two quadrants of the (f_1,f_2) plane, however in case of analyzing real-valued signals, the opposite quadrants bear complex conjugate spectra, and therefore do not add more information to the analysis.

An alternative approach is the 'steerable pyramid' proposed in [SIMONCELLI ET AL. 1992], and a complex version providing similar analysis as complex Gabor

wavelets described in [PORTILLA, SIMONCELLI 2000]. The steerable pyramid decomposes a 2D picture into a lowpass band with circular symmetric Fourier transfer function, and a number of angular highpass bands (see Fig. 2.26). The decomposition can be made invertible by the property that the different filters' frequency transfer functions are designed with complementary overlapping cosine-shaped roll-off, in the radial direction between lowpass and highpass, and in the angular direction between the different highpass bands. This way, $\Sigma|H_k(\mathbf{f})|^2 = \text{const.}$ over all filter transfer functions, and perfect reconstruction applies when complex conjugate filters and superposition of their outputs are applied for synthesis. The representation is overcomplete as well, as the highpass bands are not downsampled for further analysis. To establish a pyramid representation, the lowpass band is downsampled and an equivalent decomposition is again applied to the frequency ranges between −1/4 and 1/4, etc. Again, the angular orientations only need to cover a range of 180^O, since the opposite spectral bands for the case of real-valued signals are complex conjugate.

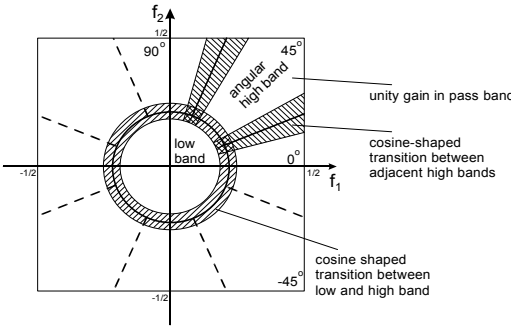

Fig. 2.26. Example frequency layout for one level of the 'steerable pyramid'

2.5 Locally adaptive filters

Some classes of nonlinear filters discussed in Sec. 2.1 show behaviour that is preferable over linear filter operations in terms of preserving discontinuities, removal of outlier samples and denoising. However, similar effects can also be achieved by employing linear filter kernels, which have to be adapted locally depending on properties of the signal itself. Methods of this category are discussed in the subsequent subsections.

2.5.1 Steerable smoothing filters

One of the most relevant purposes of applying nonlinear processing is for preserving relevant structures such as edges and corners while removing local variations

e.g. caused by noise or other irregular natural structures. If the disturbance is high frequent, a simple way of achieving this is by (weighted) averaging of samples that are likely belonging to an area of same amplitude. For example, if an edge structure is present in a picture, directional lowpass filtering *in parallel* with that edge would be useful, without having the effect of smoothing the edge itself. The necessary control information can be obtained by directional edge analysis (Sec. 4.3.1). For example, when the edge orientation is categorized into four classes, the following set of binomial filters[26] for horizontal, vertical and the two diagonal directions could be applied,

$$\mathbf{H}_{\mathrm{h}} = \frac{1}{4}\begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix};\ \mathbf{H}_{\mathrm{d}^+} = \frac{1}{4}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix};\ \mathbf{H}_{\mathrm{v}} = \frac{1}{4}\begin{bmatrix} 0 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \end{bmatrix};\ \mathbf{H}_{\mathrm{d}^-} = \frac{1}{4}\begin{bmatrix} 0 & 0 & 1 \\ 0 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix}. \quad (2.71)$$

Whereas the single filter kernels of (2.71) belong to the class of LSI systems where a local adaptation selects one of them, the class of filters discussed in the remaining part of this section is based on defining a local smoothing operator where samples from a neighborhood $\mathcal{N}$ are superimposed with a certain linear weight w in the output

$$g(\mathbf{n}) = \frac{\sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} s(\mathbf{m})w(\mathbf{m},\mathbf{n})}{\sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} w(\mathbf{m},\mathbf{n})}, \quad (2.72)$$

where the weight is controlled by local image properties such as strength and orientation of gradient, or potentially additional clues which are associated with the local position, e.g. depth or motion. For the case of invariant weights $w(\mathbf{m},\mathbf{n})$, this could also be interpreted as a convolution operation. In case of adaptation (as in the schemes subsequently described) the LSI property is lost.

Non-local means (NLM) filtering. In this type of filter [BUADES 2005], the weight is derived such that it becomes high when the sample is close to the mean μ over the neighborhood around position $\mathbf{n}$, and low otherwise. A simple approach is using a Gaussian weighting

$$w(\mathbf{m},\mathbf{n}) = \mathrm{e}^{-\frac{[s(\mathbf{m})-\mu(\mathbf{n})]^2}{b^2}} \quad \text{with} \quad \mu(\mathbf{n}) = \frac{1}{|\mathcal{N}|}\sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} s(\mathbf{m}), \quad (2.73)$$

where b determines the width (strength) of the weighting function, and $|\mathcal{N}|$ is the

[26] A binomial filter has an impulse response constructed as binomial series, for which a vivid interpretation is given by *Pascal's triangle*. Another interpretation can be given by iterated convolution of a 2-tap averaging filter $\mathbf{h} = [0.5\ 0.5]^{\mathrm{T}}$. The binomial series can also be interpreted as a discrete approximation of a Gaussian function, since the latter can be obtained by iterated convolution of a large number of rect functions.

number of samples in the neighborhood. With $d=s(\mathbf{m})-\mu(\mathbf{n})$, examples of alternative weighting functions are exponential, 'Perona-Malik', and 'Tukey's biweight' functions,

$$w_{\mathrm{Exp}}(d)=\mathrm{e}^{-\frac{|d|}{b}};\quad w_{\mathrm{PeMa}}(d)=\frac{1}{1+\left(\frac{d}{b}\right)^2};\quad w_{\mathrm{TuBi}}(d)=\begin{cases}\left(1-\left(\frac{d}{b}\right)^2\right)^2 & \text{if } |d|<|b|,\\ 0, & \text{else.}\end{cases} \tag{2.74}$$

Note that the definitions of functions given in (2.73)-(2.75) are not normalized, but normalization is performed in the context of their usage, e.g. in the denominator of (2.72). Another variant is the linear ramp[27] function defined between two threshold values b_1 and b_2[28],

$$w_{\mathrm{Ramp}}[d]=\begin{cases}1 & \text{if } |d|\le b_1,\\ 1-\dfrac{|d|-b_1}{b_2-b_1} & \text{if } b_1<|d|\le b_2,\\ 0 & \text{if } |d|>b_2.\end{cases} \tag{2.75}$$

Bilateral filter. In NLM, a weighted mean could alternatively be applied, where samples that are closer to $\mathbf{n}$ take higher influence in the mean computation. A similar effect is achieved by *bilateral filtering* [TOMASI, MANDUCHI 1998], where however the difference expression influencing the weight w_1 is not related to the mean, but rather to the amplitude deviation between a sample from the neighborhood and the current sample, i.e. $d(\mathbf{m})=s(\mathbf{m})-s(\mathbf{n})$. Further, the geometric (Euclidean) distance $\|\mathbf{m}-\mathbf{n}\|$ between sample locations influences a second weighting function w_2 to generate the output

$$g(\mathbf{n})=\frac{\sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} s(\mathbf{m})w_1(d(\mathbf{m}))w_2(\|\mathbf{m}-\mathbf{n}\|)}{\sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} w_1(d(\mathbf{m}))w_2(\|\mathbf{m}-\mathbf{n}\|)}. \tag{2.76}$$

Whereas the Gaussian (2.73) is most commonly used as weight, any of the aforementioned weighting functions would be applicable as well (also different types for w_1 and w_2, or with individual b values).

2.5.2 Iterative smoothing (diffusion filters)

In case of median filtering, iterative operation would delete all structures which are smaller than the corresponding root signals, and in larger areas would by ten-

---

[27] The symmetric ramp function has the shape of a trapeze.

[28] Alternative nonlinear transition definitions could be based on $\cos^2$, arctan and sigmoid (5.156) functions. Higher-dimensional functions can be separable or circular symmetric.

dency let all samples converge towards homogeneity, while still preserving discontinuities. A similar approach is possible in the context of edge-preserving smoothing, where the output $g(\mathbf{n})$ is iteratively determined as an estimate from the input $s(\mathbf{n})$. Even though estimation algorithms will be further discussed in Ch. 1, some basic concepts are introduced here to demonstrate methods taking into account the local properties of the signal in this context.

Total variation denoising. For the output $g(\mathbf{n})$, the local variation between a sample and its neighbors at each position $\mathbf{n}$, and subsequently the total variation (TV) over all samples can be defined as

$$V\{g(\mathbf{n})\} = \sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} |g(\mathbf{n}) - g(\mathbf{m})| . \tag{2.77}$$

However, $g(\mathbf{n})$ should also be similar to the input $s(\mathbf{n})$, which can for example be tested by the Euclidean distance

$$\mathrm{d}\{g(\mathbf{n}), s(\mathbf{n})\} = |g(\mathbf{n}) - s(\mathbf{n})|^2 . \tag{2.78}$$

Both criteria can jointly be optimized by minimizing the following criterion over al samples,

$$J = \sum_{\mathbf{n}} \left[d\{g(\mathbf{n}), s(\mathbf{n})\} + \lambda V\{g(\mathbf{n})\} \right], \tag{2.79}$$

which can be achieved by differentiating (2.79) over $g(\mathbf{n})$. By increased λ, the homogeneity in $g(\mathbf{n})$ gets higher weight than the similarity to the original $s(\mathbf{n})$; by tendency, TV denoising will retain the local mean value as well as discontinuities.

Anisotropic diffusion. A diffusion process solves a stochastic differential equation on a given input signal by continuously (i.e. taking the previous output as recursive input to the next step) aligning a sample with its neighbors under the assumption that they stem from the same stochastic process. If the underlying distribution is Gaussian, this is equivalent to iteratively convolving the signal by a Gaussian impulse response, such that it becomes more and more smooth. The final result could also be obtained by convolving with a very wide Gaussian filter kernel, which would smooth out any structure; this case is called *isotropic diffusion*[29].

In anisotropic diffusion [PERONA, MALIK 1990], basically the same procedure is applied, but the smoothing is not performed when a neighbor has a large deviation in amplitude, such that discontinuities are preserved. Basically the same types of weighting functions as previously suggested can be applied here, but according to their role of preventing the diffusion these are denoted as *stopping functions*. An

[29] This is to some extent equivalent to a scale-space approach of the Gaussian pyramid (Sec. 2.4), when subsampling would be omitted.

equivalent discrete implementation (typically using a 4-neighbor system $\mathcal{N}_1^{(1)}$) could compute the result of the r^{th} iteration as

$$\hat{s}^{(r)}(\mathbf{n}) = \hat{s}^{(r-1)}(\mathbf{n}) + \frac{\lambda}{|\mathcal{N}|} \sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} w(\mathbf{m},\mathbf{n}) \nabla(\mathbf{m},\mathbf{n}) \hat{s}^{(r-1)}(\mathbf{n}) \quad \text{with } \nabla(\mathbf{m},\mathbf{n}) = \hat{s}^{(r-1)}(\mathbf{m}) - \hat{s}^{(r-1)}(\mathbf{n}), \tag{2.80}$$

where $w(\mathbf{m},\mathbf{n})$ uses functions as in (2.73)-(2.75). The sample's amplitude is driven towards those of its neighbors unless the stopping function $w(\mathbf{m},\mathbf{n})$ prevents this (i.e. in cases where $\nabla(\mathbf{m},\mathbf{n})$ is large). The diffusion process starts with $\hat{s}^{(0)}(\mathbf{n}) = s(\mathbf{n})$ and ends after R iterations by $g(\mathbf{n}) = \hat{s}^{(R)}(\mathbf{n})$.

Another equivalent method applies iterative filtering (e.g. using Gaussian kernels) where the stopping function in iteration r is directly implemented as part of the impulse response, with criteria based on the result from iteration $r-1$. In the following example takes into account the absolute values of the differences in the second derivative between the current sample and a neighbor from $\mathcal{N}(\mathbf{n})$[30]

$$\Delta(\mathbf{m},\mathbf{n}) = \left| \nabla^2 s^{(r-1)}(\mathbf{n}) - \nabla^2 s^{(r-1)}(\mathbf{n}+\mathbf{m}) \right|, \tag{2.81}$$

as well as deviations by absolute intensity $|\nabla(\mathbf{m},\mathbf{n})|$. Optionally, an external control function $\theta(\mathbf{n})$ with values between 0 and 1 can be defined. Setting $\theta(\mathbf{n})=1$ at the given position enforces an isotropic diffusion process (which means smoothing across edges will be performed even though $|\nabla|$ and Δ are large)[31]. The weight in (2.80) can then be determined at position $\mathbf{n}$ by the following method:

$$w(\mathbf{m},\mathbf{n}) = \left(w_1\left[\nabla(\mathbf{m},\mathbf{n})\right] \cdot w_2\left[\Delta(\mathbf{m},\mathbf{n})\right] \right) \cdot [1-\theta(\mathbf{n})] + \theta(\mathbf{n}) \,. \tag{2.82}$$

For the underlying functions $w_1(\cdot)$ and $w_2(\cdot)$, (2.73)-(2.76) can again be used. From the corresponding isotropic Gaussian kernel $h_G(\mathbf{m})$ (e.g. a sampled version of (2.65) in case of 2D signals), the anisotropic kernel at position $\mathbf{n}$ is then computed as

$$\tilde{h}(\mathbf{m},\mathbf{n}) = \frac{w(\mathbf{m},\mathbf{n}) h_G(\mathbf{m})}{\sum_{\mathbf{k}\in\mathcal{N}(\mathrm{n})} w(\mathbf{k},\mathbf{n}) h_G(\mathbf{k})}, \tag{2.83}$$

and the signal in iteration r is generated as

---

30 The second derivative $\nabla^2(\mathbf{n})$ can be computed using the LoG (2.66), or by the filter kernels (4.60)-(4.62). Large differences in the second derivative indicate a change in gradient, which is a typical indicator for the presence of edge positions, cf. Sec. 4.3.2.

31 This can be based on additional rules or criteria, e.g. desirable minimum size of a smooth regions, or other features gained independently like homogeneity of a motion vector field, when the method shall be used for smoothing the appearance of a moving object.

$$\hat{s}^{(r)}(\mathbf{n}) = \sum_{\mathbf{m}\in\mathcal{N}(\mathbf{n})} \hat{s}^{(r-1)}(\mathbf{m})\tilde{h}(\mathbf{m},\mathbf{n}) \,. \tag{2.84}$$

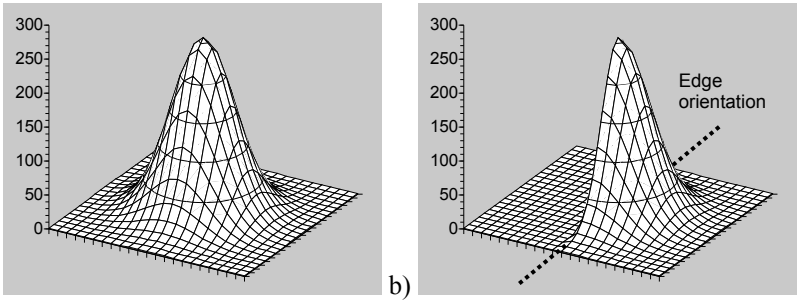

Fig. 2.27. Gaussian diffusion filter kernels: **a** isotropic **b** anisotropic-weighted (example with hard limitation of filter weighting beyond a step edge)
[source: IZQUIERDO/OHM]

The anisotropic diffusion method may use small Gaussian filter kernels by each iteration step, since the iterative application of Gaussian impulse response is equivalent to stronger Gaussian lowpass filtering. Fig. 2.27 shows 2D impulse response shapes of an isotropic and a weighted Gaussian kernel for the example of an image edge that is present in the neighborhood.

2.6 Problems

Problem 2.1
The following image signal is given.

$$\begin{bmatrix} 5 & 5 & 15 & 15 \\ 10 & \boxed{30} & 25 & 20 \\ 5 & 20 & \boxed{25} & 20 \\ 5 & 5 & 10 & 20 \end{bmatrix}$$

a) At the highlighted positions, perform median filtering using the following non-weighted and weighted configurations:
   i) 4-neighborhood $\mathcal{N}_1^{(1)}(n_1,n_2)$
   ii) 8-neighborhood $\mathcal{N}_2^{(2)}(n_1,n_2)$
   iii) Neighborhood $\mathcal{N}_1^{(1)}(n_1,n_2)$, center sample weighted three-fold
   iv) Neighborhood $\mathcal{N}_2^{(2)}(n_1,n_2)$, center sample weighted three-fold
b) Sketch the root signals for the median filters of iii) and iv).

Problem 2.2
Sketch the root signals of the median filter geometries shown in Fig. 2.28. Black dots indicate positions of values belonging to the filter masks.

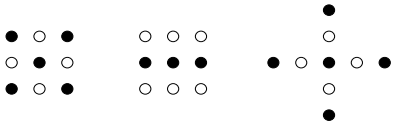

Fig. 2.28. Median filter masks for which root signals shall be found

Problem 2.3
Within the area marked in the following image matrix, perform nonlinear filter operations using 3x3 filter masks. Whenever necessary, use constant-value extensions of the images:
a) Median b) Maximum value (dilation) c) Minimum value (erosion)
d) Maximum-difference e) opening f) closing

$$\mathbf{S} = \begin{bmatrix} 10 & 10 & 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 10 & 20 & 20 & 20 & 20 \\ 10 & 10 & 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 10 & 10 & 10 & 20 & 20 \\ 10 & 10 & 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

Problem 2.4
For the 1D signal $s(n)$ shown in Fig. 2.29, sketch the results by the following nonlinear filter operations. Use a structure element of length 3 and assume constant-value extension at the boundary:
i) Median filter ii) Maximum-difference filter iii) Erosion filter
iv) Dilation filter v) Opening filter vi) Closing filter

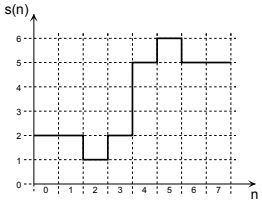

Fig. 2.29. 1D amplitude shape

Problem 2.5
The amplitude values of an image signal (2,3,4,5) shall be known at the positions shown in Fig. 2.30.

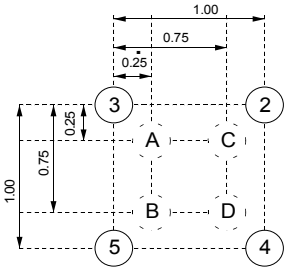

Fig. 2.30. 2D interpolation problem: Integer numbers indicate known amplitudes and their position, letters are positions to be interpolated

a) Determine the intermediate values **A**,**B**,**C**,**D** by median filter interpolation. Herein, the median values shall be computed from each three sampling positions which are nearest neighbors of the interpolated position.
b) At positions **A** and **B**, determine the deviations of median-interpolated values from the values of bilinear interpolation (2.55).

Problem 2.6

Basis functions of quadratic spline interpolation are piecewise defined as in (2.43).

a) Construct the matrix form of the equation to determine the value $\hat{s}(t)$ from the coefficients c_0, c_1, c_2 in dependency of t'.
b) Determine conditions to compute the coefficients $c(m)$ from the known samples of a signal.

http://www.springer.com/978-3-662-52826-6

Multimedia Content Analysis
Ohm, J.
2016, X, 417 p. 171 illus., Hardcover
ISBN: 978-3-662-52826-6