

Data Analysis

25. Data Analysis

From the beginning of science, visual observation has played a major role. At that time, the only way to document the results of an experiment was by verbal description and manual drawings. The next major step was the invention of *photography* more than one and a half centuries ago, which enabled experimental results to be documented objectively. In experimental fluid mechanics, flow visualization techniques gave direct insight into complex flows, but it was very difficult and time consuming to extract quantitative measurements from photographs and films.

Nowadays, we are in the middle of a second revolution sparked by the rapid progress in both photonics and computer technology. Sensitive solid-state cameras are available that acquire digital image data, and standard personal computers and workstations have become powerful enough to process these data. These technologies are now available to any scientist or engineer. As a consequence, image processing has expanded and continues to expand rapidly from a few specialized applications into a standard scientific tool.

This chapter gives a brief presentation of some of the most important general image processing

25.1 Image Processing	1437
25.1.1 Sampling and Quantization	1437
25.1.2 Radiometric Corrections	1440
25.1.3 Geometric Corrections	1442
25.1.4 Averaging and Noise Suppression	1445
25.1.5 Edge and Line Extraction	1451
25.1.6 Direction and Orientation	1453
25.1.7 Local Wavenumber and Local Phase	1458
25.1.8 Multiscale Processing	1461
25.2 Motion Analysis	1464
25.2.1 General Considerations on Motion Analysis	1464
25.2.2 Correlation-Based Velocity Analysis	1469
25.2.3 Least-Squares Matching	1473
25.2.4 Tracking Techniques	1474
25.2.5 Optical-Flow-Based Velocity Analysis	1481
References	1488

techniques that are required to process image data in experimental fluid mechanics. The second section (Sect. 25.2) deals with motion analysis. The most important methods are introduced and classified according to the fundamental principles, assumptions and approximations upon which they are based.

25.1 Image Processing

25.1.1 Sampling and Quantization

Computers process digital numbers. Therefore, the final steps of digital image formation are *digitization* and *quantization*.

Sampling Theorem

Digitization means sampling the gray values at a discrete set of points, which can be represented by a matrix. Sampling may already occur in the sensor that converts

the collected photons into an electrical signal. A CCD camera already has a matrix of discrete sensors. Each sensor is a sampling point on a two-dimensional (2-D) grid.

Sampling not only leads to a reduction in resolution, but also to a loss of information as structures of about the scale of the sampling distance and finer will be lost. It also introduces considerable distortions if fine structures are sampled. Figure 25.1 shows a simple example. Digitization is simulated by taking only every fourth pixel in

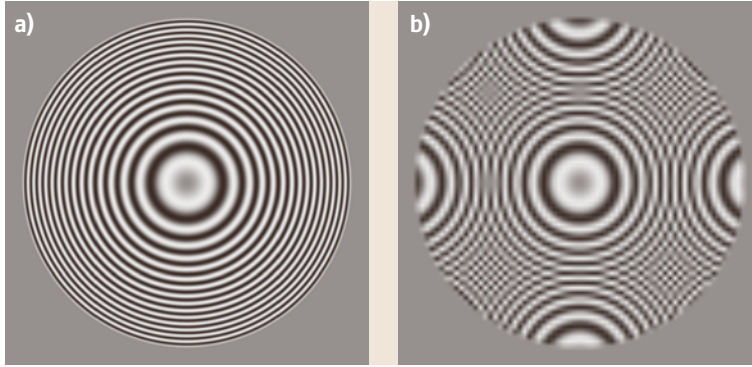


Fig. 25.1a,b Explanation of the moiré effect with a ring test pattern: **(a)** original pattern with structures that are sampled four times per wavelength at the edge of the ring; **(b)** Pattern first downsampled using only every fourth column and row and then up-sampled using linear interpolation to the original size

every fourth row. This kind of image distortion is called the *moiré* effect. The same phenomenon, called *aliasing*, is known for one-dimensional signals, especially time series.

The sampling theorem states under which conditions these distortions can be avoided and, even more, whether a complete reconstruction of the sampled continuous image is possible; if the two-dimensional (2-D) spectrum $\hat{g}(k_1, k_2)$ of a continuous image function $g(x_1, x_2)$ is band-limited, i. e.,

$$\hat{g}(\mathbf{k}) = 0 \quad \forall |\mathbf{k}| \geq k_{\max 1, \max 2}, \quad (25.1)$$

then it can be reconstructed exactly from samples with a distance

$$\Delta x_w = \frac{1}{2k_{\max 1, \max 2}}. \quad (25.2)$$

In other words, at least two samples per wavelength are required. The maximum wavenumber that can be sampled without errors is called the *Nyquist* or *limiting wavenumber* k_{\max} . Often dimensionless wavenumbers $\tilde{k} = k/k_{\max}$ that are scaled by the *Nyquist wavenumber* and are confined to an interval $[-1, 1]$ are used.

Standard Sampling

An array of photosensitive sensor elements does not perform a point sampling. Rather the average over the light-sensitive area of each sensor element is taken. This corresponds to a convolution of the image signal by the spatial distribution of the light sensitivity. In the best case, the whole area of the sensor element is equally sensitive. This is known as *standard sampling*. It is a kind of *regular sampling*, because each point in the continuous space is equally weighted.

The averaging over the light-sensitive area causes spatial blurring of the signal and thus some band limitation. However, this is not sufficient to avoid

moiré effects. Convolution by a box function $\Pi(x/\Delta x)$ of the width Δx in the spatial domain is equivalent to a multiplication by

$$\text{sinc}(\tilde{k}/2) = \frac{\sin \pi \tilde{k}/2}{\pi \tilde{k}/2}$$

in the Fourier domain. At the Nyquist wavenumber $\tilde{k} = 1$, the Fourier transform of the box function is still $2/\pi$. The first zero crossing occurs only at twice the Nyquist wavenumber. The band limitation is worse with a real imaging sensor, when only a fraction of the sensor element area is light sensitive.

Because of the insufficient band limitation by the imaging sensor, other means have to be taken in order to avoid moiré effects. The best situation is when the imaged object is bandlimited itself. Additional band limitation can also be introduced by the optical system.

Reconstruction from Samples

Reconstruction is performed by a suitable *interpolation of the sampled points*. Generally, the interpolated points at continuous positions $g_r(\mathbf{x})$ are calculated from the sampled values $g(\mathbf{r}_{m,n})$ on a regular grid

$$\mathbf{r}_{m,n} = [m\Delta x_1, n\Delta x_2]^T \quad \text{with } m, n \in \mathbb{Z}. \quad (25.3)$$

weighted with suitable factors depending on the distance from the interpolated point:

$$g_r(\mathbf{x}) = \sum_{m,n} h(\mathbf{x} - \mathbf{r}_{m,n}) g_s(\mathbf{r}_{m,n}). \quad (25.4)$$

From the sampling theorem it can be inferred that an *exact* reconstruction of the continuous image is possible when the original continuous image meets the sampling theorem and the transfer function of the *interpolation kernel* $h(\mathbf{x})$ is a box function:

$$\hat{g}_r(\mathbf{k}) = \Pi(\tilde{k}_1/2, \tilde{k}_2/2) \hat{g}(\mathbf{k}). \quad (25.5)$$

Then the ideal interpolation function is the inverse Fourier transform of the box function, a sinc function:

$$h(x) = \text{sinc}(x_1/\Delta x_1) \text{sinc}(x_2/\Delta x_2). \quad (25.6)$$

Oversampling

Unfortunately, the ideal interpolation function only decreases like $1/x$ towards zero. Therefore, correct interpolation requires many sampling points and is therefore inefficient.

More-efficient solutions to the interpolation problem can be obtained if the sampling theorem is *overfilled*, i. e., $\hat{g}(k)$ is already zero before the Nyquist wavenumber is reached. Then $\hat{h}(k)$ can have any value in the region where \hat{g} vanishes without introducing errors. This freedom can be used to construct an interpolation function that decreases more quickly in the spatial domain, i. e., has a minimum-length interpolation mask. We can also start from a given interpolation formula. Then the deviation of its Fourier transform from a box function tells us to what extent structures will be distorted as a function of the wavenumber.

The principle of *oversampling* is not only of importance for the construction of effective interpolation functions. It is also essential for the design of any type of precise filter with small filter masks (Sects. 25.1.4, 25.1.5). Generally, the rate of oversampling, which increases the number of data points, and the requirements of the filter design must be balanced. Practical experience shows that a sample rate between three and six samples per wavelength, i. e., 1.5- to 3-fold oversampling, is a good compromise.

Quantization and Resolution

Computer can only handle digital numbers. Therefore continuous number are mapped onto a limited number Q of discrete gray values (*quantization*):

$$[0, \infty[\xrightarrow{Q} \{g_0, g_1, \dots, g_{Q-1}\} = G.$$

Quantization always introduces errors, as the true value g is replaced by one of the quantization levels g_q . If the quantization levels are equally spaced with a distance Δg and if all gray values are equally probable, the variance introduced by the quantization is given by

$$\sigma_q^2 = \frac{1}{\Delta g} \int_{g_q - \Delta g/2}^{g_q + \Delta g/2} (g - g_q)^2 dg = \frac{1}{12} (\Delta g)^2. \quad (25.7)$$

The standard deviation σ_q is about 0.3 times the distance between the quantization levels Δg .

With respect to the quantization, the question arises of the accuracy to which we can measure a gray value. At first glance, the answer to this question seems to be trivial and given by (25.7): the maximum error is half a quantization level and the mean error is about 0.3 quantization levels.

But what if we measure the value repeatedly? This could happen if we take many images of the same object or if we have an object of a constant gray value and want to measure the mean gray value of the object by averaging over many pixels.

For repeated measurements, the error of the mean value decreases with the number N of measurements according to

$$\sigma_{\text{mean}} \approx \frac{1}{\sqrt{N}} \sigma, \quad (25.8)$$

where σ is the standard deviation of the individual measurements and N is the number of measurements taken. If 100 measurements are taken, the error of the mean should be just about a tenth of the error of the individual measurement.

Taking quantization into account, however, averaging requires a more-detailed analysis. If no noise is present, the same quantized value is always measured. Then the result cannot be more accurate than the individual measurements.

However, if the measurements are noisy, we would obtain different values for each measurement. The probability for the different values reflects the mean and variance of the noisy signal, and because we can the distribution, we can estimate both the mean and the variance.

As an example, a standard deviation of the noise equal to the quantization level is discussed. Then, the standard deviation of an individual measurement is about three times larger than the standard deviation due to the quantization. However, already with 100 measurements, the standard deviation of the mean value is only 0.1, or 3 times lower than that of the quantization.

As in images we can easily obtain many measurements by spatial averaging, there is the potential to measure mean values with standard deviations that are much smaller than the standard deviation of quantization in (25.7).

The accuracy is also limited, however, by other, systematic errors. The most significant source is the unevenness of the quantization levels. In a real quantizer, such as an analog-to-digital converter, the quantization levels are not equally distant but show systematic deviations that may be up to half a quantization interval. Thus,

careful investigation of the analog-to-digital converter is required to estimate what really limits the accuracy of the gray value measurements in images.

25.1.2 Radiometric Corrections

The first image processing steps include two classes of operations: point and geometric operations. Essentially, these two types of operations modify the *what* and *where* of a pixel.

Point operations modify the gray values at individual pixels depending only on the gray value and possibly on the position of the pixels. Generally, such a kind of operation is expressed by

$$G'_{mn} = P_{mn}(G_{mn}) . \quad (25.9)$$

The indices of the function P denote the possible dependence of the point operation on the position of the pixel.

Homogeneous point operators are the same for all pixel and can be implemented via look-up tables. They are a very useful tool for inspecting images. As the look-up table operations work in real time, images can be manipulated interactively. If only the output look-up table is changed, the original image content remains unchanged. Here, some typical tasks are demonstrated.

Evaluating and Optimizing Illumination

With the naked eye, it is very hard to estimate the homogeneity of an illuminated area (Fig. 25.2a). We need to mark gray scales such that absolute gray levels become perceivable for the human eye. If the radiance distribution is continuous, it is sufficient to use equidensities. This technique uses a staircase-type homogeneous point operation by mapping a certain range of gray scales onto one. This point operation is achieved by zeroing the p least-significant bits with a *logical AND operation*:

$$q' = P(q) = q \wedge \overline{(2^p - 1)} , \quad (25.10)$$

where \wedge denotes the logical (bitwise) *and* and the overline denotes *negation*. This point operation limits the resolution to $Q - p$ bits and, thus, 2^{Q-p} quantization levels. Now, the jump between the remaining quantization levels is large enough to be perceived by the eye and to see contour lines of equal absolute gray scale in the image (Fig. 25.2). Another way to mark absolute gray values is the so-called *pseudocolor image*. With this technique, a gray level q is mapped onto a *red-green-blue (RGB) triple* for display. As color is much better

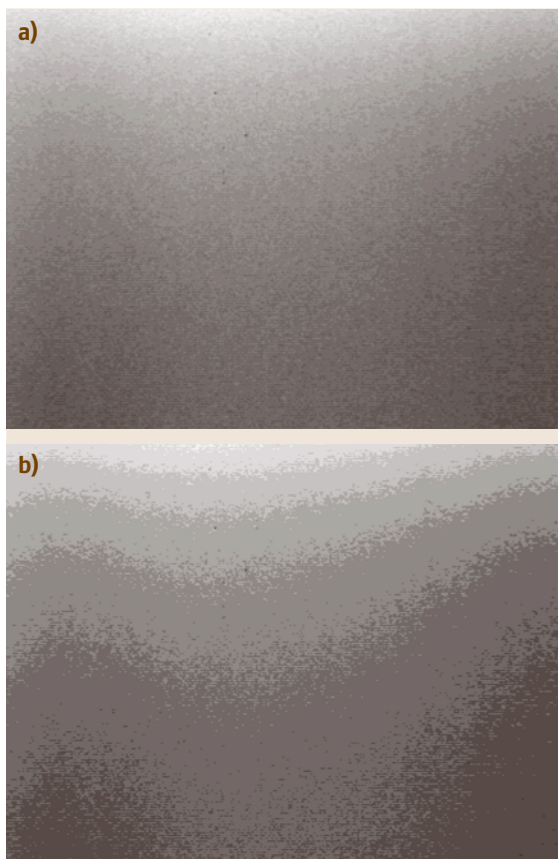


Fig. 25.2 (a) and (b) (contrast-enhanced, gray scale 184–200): edges artificially produced by a staircase look-up table (LUT) with a step height of 1.0 and 2.0 make contours of constant irradiance easily visible

recognized by the eye, this helps reveal absolute gray levels.

Detection of Underflow and Overflow

Under- and overflows of the gray values of a digitized image often go unnoticed and cause serious bias in further processing, for instance, for mean gray values of objects or the center of gravity of an object. In most cases, such areas cannot be detected directly. They may only become apparent in textured areas when the texture is bleached out. Over- and underflow are detected easily in histograms by strong peaks at the minimum and/or maximum gray values (Fig. 25.3). With pseudocolor mapping, the few lowest and highest gray values could be displayed, for example, in blue and red, respectively. Then, gray values dangerously close to the limits

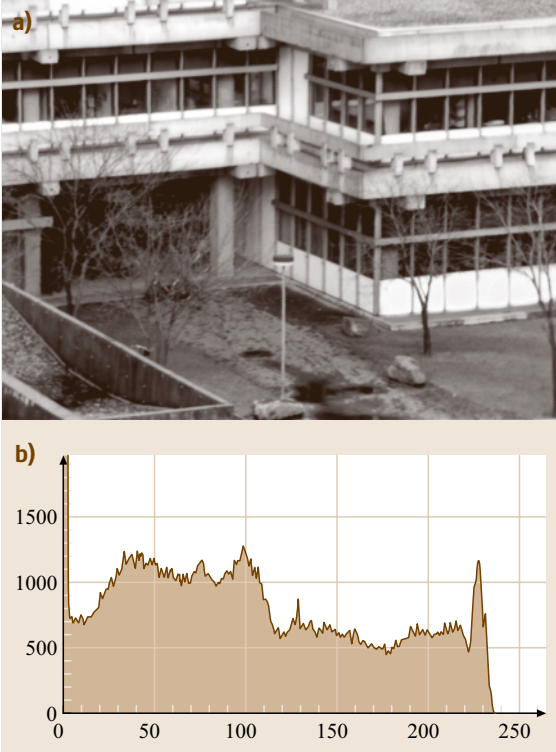


Fig. 25.3a,b Detection of underflow and overflow in digitized images by histograms: **(a)** image with underflow and **(b)** its histogram

immediately pop out of the image and can be avoided by correcting the illumination, lens aperture or gain and offset of the camera.

Noise Variance Equalization

The variance of the noise of a linear image sensor (Sect. 24.4.9) is not constant but depends on the image intensity g according to

$$\sigma_g^2(g) = \sigma_0^2 + Kg. \quad (25.11)$$

Many statistical image analysis procedures, however, are based on gray-value-independent, normally distributed, additive noise. Because brighter regions in the image have a larger variance, their influence is generally overestimated, while darker regions still contains valid information with less statistical uncertainty, which is not adequately used.

The nonlinear gray value transform

$$h(g) = \gamma g_{\max} \frac{\sqrt{\sigma_0^2 + Kg - \sigma_0}}{\sqrt{\sigma_0^2 + Kg_{\max} - \sigma_0}}, \quad (25.12)$$

maps the gray values into the interval $[0, \gamma g_{\max}]$ and the standard deviation

$$\sigma_h = \frac{\gamma K g_{\max} / 2}{\sqrt{\sigma_0^2 + Kg_{\max} - \sigma_0}} \quad (25.13)$$

is independent of the gray value.

The nonlinear transform becomes particularly simple for an ideal imaging sensor with no dark noise ($\sigma_0 = 0$). Then a square-root transform must be applied to obtain an intensity-independent noise variance:

$$h(g) = \gamma \sqrt{g g_{\max}} \quad \text{and} \quad \sigma_h = \frac{\gamma}{2} \sqrt{K g_{\max}}. \quad (25.14)$$

Correction for Inhomogeneous Illumination

Every real-world application has to contend with *uneven illumination* of the observed scene. Even if a lot of effort is spent optimizing the illumination setup, it is still very hard to obtain perfectly even object irradiance. A nasty problem is caused by small dust particles in the optical path, especially on the glass window close to the charge-coupled device (CCD) sensor. Because of the distance of the window from the imager, these particles – if they are not too large – are blurred to such an extent that they are not directly visible. However, they still absorb some light and thus cause a drop in the illumination level in a small area. These effects are not easily visible in a scene with high contrast and many details, but become very apparent in the case of a uniform background (Fig. 25.2a,b). Some imaging sensors, especially complementary metal oxide semiconductor (CMOS) sensors, also show considerable uneven sensitivity of the individual photoreceptors, which adds to the nonuniformity of the image. These distortions can severely limit the quality of the images. These effects make it more difficult to separate an object from the background, and introduce systematic errors for subsequent image processing steps.

It is possible to correct for these effects if we know the nature of the distortion and can take suitable reference images. A simple *two-point radiometric calibration* of an imaging sensor can be applied for every sensor with a linear response. The following reference images are taken: firstly, a dark image B without any illumination and secondly a reference image R with an object of constant radiance, e.g., by looking with the camera into an *integrating sphere*. Then, a normalized image corrected for both an inhomogeneous

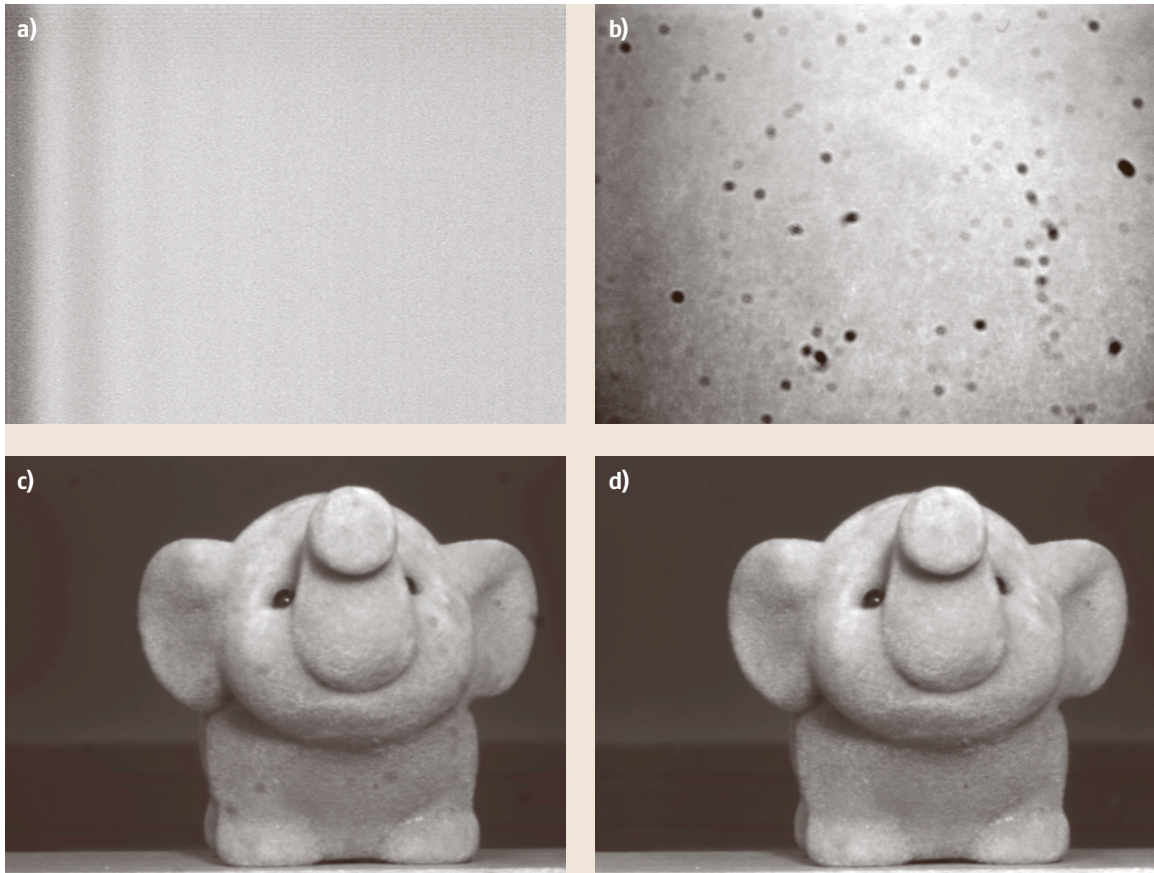


Fig. 25.4a–d Contrast-enhanced (a) dark image and (b) reference image for a two-point radiometric calibration of a CCD camera with analog video output. Two-point radiometric calibration with the dark and reference image: (c) original image and (d) calibrated image; in the calibrated image the dark spots caused by dust are no longer visible

dark image and inhomogeneous sensitivity is given by

$$G' = c \frac{G - B}{R - B}. \quad (25.15)$$

Figures 25.4a,b show a contrast-enhanced dark image and reference image of a CCD camera. Typical signal distortions can be observed. The signal oscillation at the left edge of the dark image results from an electronic interference, while the dark blobs in the reference image are caused by dust on the glass window in front of the sensor. The improvement due to the radiometric calibration according to (25.15) can clearly be seen in Figs. 25.4c,d.

Often, the quantity to be measured by an imaging sensor is related in a nonlinear way to the measured

gray value. In such cases a more-complex *nonlinear radiometric calibration* is required.

25.1.3 Geometric Corrections

Geometric transforms of images are required to correct for geometric distortions introduced during the image formation process or if scaling, rotating or any other kind of geometric transform of images is required. These *geometric operations* modify only the position of a pixel. A pixel located at the position \mathbf{x} is relocated to a new position \mathbf{x}' . These operations include two major steps. In most applications, the mapping function is not given explicitly but must be derived from corresponding points. When an image is warped by a geometric transform, the pixels in the original and warped images almost never fall onto each other. Thus, it is required to in-

interpolate gray values at these pixel from neighboring pixels.

Forward and Inverse Mapping

Geometric transforms define the relationship between the points in two images. This relation can be expressed in two ways. Either the coordinates of the output image \mathbf{x}' can be specified as a function of the input coordinates \mathbf{x} or vice versa:

$$\mathbf{x}' = M(\mathbf{x}) \quad \text{or} \quad \mathbf{x} = M^{-1}(\mathbf{x}'), \quad (25.16)$$

where M specifies the mapping function and M^{-1} is its inverse. The two expressions in (25.16) give rise to two principal kinds of spatial transformation: *forward* and *inverse mapping*.

With *forward mapping*, a pixel of the input image is mapped onto the output image (Fig. 25.5a). Generally, a pixel of the input image lies between the pixels of the output image. With forward mapping, it is not appropriate just to assign the value of the input pixel to the nearest pixel in the output image (point-to-point or nearest-neighbor mapping), as it may happen that the transformed image contains holes as a value is never assigned to a pixel in the output image or that a value is assigned more than once to a point in the output image. An appropriate technique distributes the value of the input pixel to several output pixels. The easiest procedure is to regard pixels as squares and to take the fraction of the area of the input pixel that covers the output pixel

as the weighting factor. Each output pixel accumulates the corresponding fractions of the input pixels which – if the mapping is continuous – add up to cover the whole output pixel.

With *inverse mapping*, the coordinates of a point in the output image are mapped back onto the input image (Fig. 25.5b). It is obvious that this scheme avoids holes and overlaps in the output image as all pixels are scanned sequentially. Now, the interpolation problem occurs in the input image. The coordinates of the output image in general do not hit a pixel in the input image but lie in between the pixels. Thus, its correct value must be interpolated from the surrounding pixels. Generally, inverse mapping is a more-flexible technique, as it is easier to implement various types of interpolation techniques.

Ideal Interpolation

The basis of interpolation is the sampling theorem. The problem is related to the fact that the reconstruction of the continuous image from the sampled image in practice is quite involved and can be performed only approximately because the ideal continuous interpolation mask h , the sinc function, requires too many operations.

Any approximate solution should still reproduce the grid points and not depend on any other grid point (the *interpolation condition*):

$$h(\mathbf{x}_{m,n}) = \begin{cases} 1 & m = 0, n = 0 \\ 0 & \text{otherwise} \end{cases}. \quad (25.17)$$

Any interpolation mask must, therefore, as for the ideal interpolation mask, have zero crossings at all grid points except the zero point, where it is 1.

The ideal interpolation function in (25.6) is separable. Therefore, interpolation can be as easily formulated for higher-dimensional images. We can expect that all solutions to the interpolation problem will also be separable. Consequently, we need only discuss the one-dimensional (1-D) interpolation problem. Once it is solved, we also have a solution for the multidimensional interpolation problem.

An important special case is the interpolation to intermediate grid points halfway between the existing grid points. This scheme doubles the resolution and image size in all directions in which it is applied. Then, the continuous interpolation kernel reduces to a discrete convolution mask. As the interpolation kernel (25.5) is separable, we can first interpolate the intermediate points in a row in the horizontal direction before we apply vertical interpolation to the intermediate rows. In linear interpolation three dimensions, a third 1-D interpolation

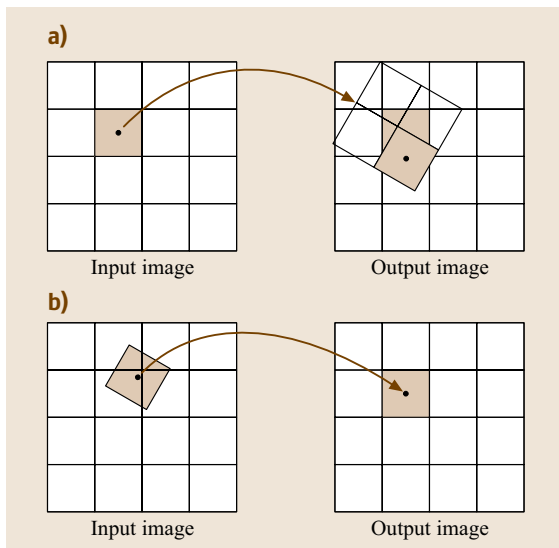


Fig. 25.5a,b Illustration of (a) forward mapping and (b) inverse mapping for spatial transformation of images

is added in the z or t direction. The interpolation kernels are the same in all directions. We need the continuous kernel $h(x)$ only at half-integer values for $x/\Delta x$.

Linear Interpolation

Linear interpolation is the classic approach to interpolation. The interpolated points lie on pieces of straight lines connecting neighboring grid points. In order to simplify the expression, we use in the following normalized spatial coordinates $\tilde{x} = x/\Delta x$. We locate the two grid points at $-1/2$ and $1/2$. This yields the *interpolation equation*

$$g(\tilde{x}) = \frac{g_{1/2} + g_{-1/2}}{2} + (g_{1/2} - g_{-1/2})\tilde{x} \quad (25.18)$$

for $|\tilde{x}| \leq 1/2$. The continuous interpolation mask for linear interpolation is

$$h_1(\tilde{x}) = \begin{cases} 1 - |\tilde{x}| & |\tilde{x}| \leq 1 \\ 0 & \text{otherwise} \end{cases} \quad (25.19)$$

Linear interpolation introduces serious distortions: while low wavenumbers (and especially the mean value $\tilde{k} = 0$) are interpolated correctly, high wavenumbers are slightly reduced in amplitude, resulting in some degree of smoothing. Furthermore spurious high wavenumbers are introduced, because the first derivative is discontinuous at the grid points.

Spline-Based Interpolation

Besides of its limited accuracy, linear and higher-order polynomial interpolation has another significant disadvantage: the interpolated curve is already discontinuous in its first derivative at the grid points. This is due to the fact that, for each interval between grid points, another polynomial is taken. Thus, only the interpolated function is continuous at the grid points but not the derivatives.

Splines avoid this disadvantage by additional constraints for the continuity of derivatives at the grid points. From the wide classes of splines, *B-splines* prove to be most useful for interpolation. As B-splines are separable, it is sufficient to discuss the properties of 1-D B-splines. From the background of image processing, the easiest access to B-splines is their convolution property. The kernel of a P -order B-spline curve is generated by convolving the box function $P + 1$ times with itself:

$$\beta_P(\tilde{x}) = \underbrace{\Pi(\tilde{x}) * \dots * \Pi(\tilde{x})}_{(P+1) \text{ times}} \quad (25.20)$$

with the Fourier transform (the *transfer function*)

$$\hat{\beta}_P(\hat{k}) = \left(\frac{\sin \pi \tilde{k}/2}{(\pi \tilde{k}/2)} \right)^{P+1} \quad (25.21)$$

The B-spline function itself is not a suitable interpolation function. The transfer function decreases too early, indicating that B-spline interpolation performs too much averaging, and the B-spline kernel does not meet the interpolation condition (25.17) for $P > 1$. B-splines can only be used for interpolation if the discrete grid points are first transformed in such a way that a following convolution with the B-spline kernel restores the original values at the grid points. This transformation is known as the *B-spline transformation* and is constructed from the condition:

$$g_P(x) = \sum_n c_n \beta_P(x - x_n) \quad \text{with} \quad g_P(x_n) = g(x_n). \quad (25.22)$$

If centered around a grid point, the B-spline interpolation kernel is unequal to zero for only three grid points. The coefficients $\beta_3(-1) = \beta_{-1}$, $\beta_3(0) = \beta_0$, and $\beta_3(1) = \beta_1$ are $1/6$, $2/3$, and $1/6$, respectively. The convolution of this kernel with the unknown B-spline transform values c_n should result in the original values g_n at the grid points.

The B-spline coefficients can be computed very efficiently by a recursive filter that is applied first in the forward and then in the backward direction with the following recursion [25.1]:

$$\begin{aligned} g'_n &= g_n - (2 - \sqrt{3})(g'_{n-1} - g_n), \\ c'_n &= g'_n - (2 - \sqrt{3})(c_{n+1} - g'_n). \end{aligned} \quad (25.23)$$

The whole operation takes only two multiplications and four additions.

The B-spline interpolation is applied after the B-spline transformation. In the continuous case this yields the *effective transfer function*

$$\hat{\beta}_I(\tilde{k}) = \frac{\sin^4(\pi \tilde{k}/2)/(\pi \tilde{k}/2)^4}{(2/3 + 1/3 \cos \pi \tilde{k})} \quad (25.24)$$

Essentially, the B-spline transformation performs an amplification of high wavenumbers (at the Nyquist wavenumber $\tilde{k} = 1$ by a factor 3). This compensates for the smoothing of the B-spline interpolation to a large extent.

For an image enlargement by a factor of two, the intermediate points are given by a convolution with the mask

$$[1 \ 23 \ 23 \ 1]/48. \quad (25.25)$$

25.1.4 Averaging and Noise Suppression

In a region with totally independent pixel gray values, nothing can be recognized. *Spatial coherency* either in one or two dimensions is required in order to recognize lines and regions, respectively (Fig. 25.6). Averaging within regions of constant gray values (an object of interest) appears to be a central tool in image processing. In addition, the deviation of the gray values of the pixel in the neighborhood gives a quantitative measure as to how well a region of constant gray values is encountered in the neighborhood. This approach is very much the same as that used for any type of measurements. A single measurement is meaningless. Only repeated measurements give us a reliable estimate both of the measured quantity and its uncertainty. In image processing, averaging needs not necessarily be performed by taking several images, although this is a very useful procedure. Because spatial information is obtained with images, spatial averaging offers an alternative.

Many objects do not show a distinct constant gray value, but we can still recognize them if the pattern they show differs from the background. After suitable preprocessing such an image can be converted into a *feature image*. Then, the feature image can be handled in the same way as a gray scale image for simple objects.

An important general question for smoothing is how it can be computed efficiently. This question is of special

importance if we want to analyze coarse features in the images that require averaging over larger distances and thus large smoothing masks (or if we apply smoothing to higher-dimensional images such as volumetric images or image sequences).

While averaging works well within regions, it is questionable at the edges of an object. When the filter mask contains pixels from both the object and the background, averaged values have no useful meaning. These values cannot be interpreted as an object-related feature since this depends on the fraction of the object pixels contained in the mask of the smoothing operator. Therefore, smoothing techniques that stop or at least diminish averaging at discontinuities are discussed. Such an approach is not trivial as it requires the detection of the *edges* before the operation can be applied. Obviously, such advanced smoothing techniques need to analyze the local neighborhoods in more detail and adapt the smoothing process in one or the other way to the structure of the local neighborhood.

Box Filters

The simplest type of averaging filters is the *box filter* or *running mean*. It averages $R \times R$ pixel around a central pixel and writes this average to the central pixel. This procedure is repeated for all pixels of an image and is mathematically a convolution of the image with an $R \times R$ mask of equal coefficients with the value $1/R^2$.

The *spatial variance* of a 1-D box filter with R coefficients is given by

$$\sigma_x^2 = \frac{1}{12}(R^2 - 1). \quad (25.26)$$

The *standard deviation* σ of smoothing increases approximately linearly with the size of the mask. The box filter is separable. Higher-dimensional box filters result from a cascaded application of the 1-D box filter in all directions.

In the Fourier domain, convolution reduces to a multiplication of the Fourier transformed image $\hat{g}(\tilde{k})$ by the Fourier transform of the smoothing mask, which is known as the transfer function. The transfer function of the box filter

$$R\hat{f}(\tilde{k}) = \frac{\sin(R\pi\tilde{k}/2)}{R\sin(\pi\tilde{k}/2)} \quad (25.27)$$

is one at the wavenumber zero (preservation of the mean value) and decreases for small wavenumbers proportionally to the wavenumber squared:

$$R\hat{f} \approx 1 - \frac{R^2 - 1}{24}(\pi\tilde{k})^2. \quad (25.28)$$

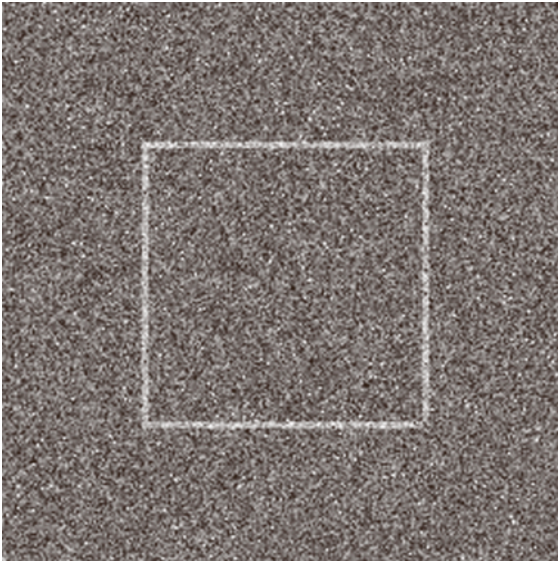


Fig. 25.6 Spatial coherency is required to recognize objects. No object can be recognized in a region that contains only randomly distributed gray values

The attenuation of large wavenumbers is not steeper than $\propto \tilde{k}^{-1}$. This is only a very weak attenuation of high wavenumbers, which renders box filters useless for many applications. With the exception of the 2R box filter (the box filter $[1\ 1]/2$), the transfer function does not decrease monotonically; rather it shows significant oscillations. This leads to the following disadvantages. First, certain wavenumbers are eliminated:

$${}^2R\hat{r}(\tilde{k}) = 0 \quad \forall \tilde{k} = \frac{n}{R/2}, \quad 1 \leq n \leq R. \quad (25.29)$$

Note that the transfer function for the highest wavenumber $\tilde{k} = 1$ vanishes only for even-sized box filters. Second, the transfer function becomes negative in certain wavenumber ranges. This means a 180° phase shift and a contrast inversion.

Because the box filter simply computes the average of $R \times R$ pixels (*running mean*), the variance of the averaged pixel reduces to

$$\frac{\sigma'}{\sigma} = \frac{1}{R}, \quad (25.30)$$

provided that the input pixels are statistically independent (*white noise*).

A box filter is isotropic only for small wavenumbers. Generally, it has the other significant disadvantage that it is strongly nonisotropic. Structures along the axes are attenuated much less than in the direction of the diagonals.

The only big advantage of the box filter is that it can be computed very efficiently as a recursive filter according to the following equation:

$$g'_n = g'_{n-1} + \frac{1}{2R+1}(g_{n+R} - g_{n-R-1}). \quad (25.31)$$

This recursion can be understood by comparing the computations for the convolution at neighboring pixels. When the box mask is moved one position to the right, it contains the same weighting factors for all pixels except for the last and the first pixel. Thus, we can simply take the result of the previous convolution, (g'_{n-1}), subtract the first pixel that just moved out of the mask (g_{n-R-1}) and add the gray value at the pixel that just came into the mask (g_{n+R}). In this way, the computation of a box filter does not depend on its size; the number of computations is of $\mathcal{O}(R^0)$. Only one addition, one subtraction, and one multiplication are required per pixel.

Thus the box filter is a fast filter with bad properties. Cascading box filters avoid or at least diminish many of the disadvantages of box filters. Since individual box filters can be computed independently of their

size, cascading them still remains independent of the size. The computational effort can be balanced against the remaining anisotropy and other distortions of the filter.

Binomial Filter

Binomial filters are built by cascading the simplest and most elementary smoothing mask

$$B = \frac{1}{2}[1\ 1], \quad (25.32)$$

taking the mean value of the two neighboring pixel. Cascading this mask R times results in the filter mask

$$\frac{1}{2^R} \underbrace{[1\ 1] * [1\ 1] * \dots * [1\ 1]}_{R \text{ times}}. \quad (25.33)$$

Only odd-sized masks should be applied if the resulting smoothed image should lie on the same grid.

Some examples of the resulting filter masks are:

$$\begin{aligned} B^2 &= [1\ 2\ 1]/4 \\ B^4 &= [1\ 4\ 6\ 4\ 1]/16 \\ B^8 &= [1\ 8\ 28\ 56\ 70\ 56\ 28\ 8\ 1]/256. \end{aligned} \quad (25.34)$$

The coefficients of the mask are the values of the *binomial distribution*. The iterative composition of the mask by consecutive convolution with the $1/2[1\ 1]$ mask is equivalent to the computation scheme of *Pascal's triangle* (Table 25.1).

The standard deviation σ of the binomial mask B^R is generally given by

$$\sigma^2 = \frac{R}{4}. \quad (25.35)$$

Table 25.1 Computation of binomial coefficients using Pascal's triangle. R denotes the order of the binomial, f the scaling factor 2^{-R} , and σ^2 the variance, i. e., the effective width of the mask.

R	f		σ^2
0	1	1	0
1	1/2	1 1	1/4
2	1/4	1 2 1	1/2
3	1/8	1 3 3 1	3/4
4	1/16	1 4 6 4 1	1
5	1/32	1 5 10 10 5 1	5/4
6	1/64	1 6 15 20 15 6 1	3/2
7	1/128	1 7 21 35 35 21 7 1	7/4
8	1/256	1 8 28 56 70 56 28 8 1	2

The standard deviation increases only with the square root of the mask size. For high R the mask size $(R + 1)$ coefficients) is therefore much larger than the standard deviation $\sqrt{R}/2$.

Two- and higher-dimensional binomial filters can be composed by cascading filters along the corresponding axes: The smallest odd-sized mask ($R = 2$) of this kind is a 3×3 binomial filter in 2-D:

$$\frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} * \frac{1}{4} \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

The transfer function of \mathcal{B}^R is then given as the R -th power:

$$\hat{b}^R(\tilde{k}) = \cos^R(\pi\tilde{k}/2), \quad (25.36)$$

which can be approximated for small wavenumbers by

$$\hat{b}^R(\tilde{k}) \approx 1 - \frac{R}{8}(\pi\tilde{k})^2. \quad (25.37)$$

The transfer function decreases monotonically and approaches zero at the largest wavenumber. The smallest mask \mathcal{B}^2 has a halfwidth of $\tilde{k}/2$. For larger masks, both the transfer function and the filter masks quickly approach the Gaussian distribution with an equivalent variance. Larger masks result in smaller half-width wavenumbers according to the *uncertainty relation*.

The noise suppression factors of a 2-D binomial mask for uncorrelated pixels is given by

$$\frac{\sigma'}{\sigma} = \frac{(2R)!}{4^R(R!)^2} \approx \left(\frac{1}{R\pi}\right)^{1/2} \left(1 - \frac{1}{8R}\right). \quad (25.38)$$

A direct computation of a $(R + 1) \times (R + 1)$ filter mask requires $(R + 1)^2$ multiplications and $(R + 1)^2 - 1$ additions. If we decompose the binomial mask into elementary smoothing masks $1/2 [1 \ 1]$ and apply this mask in horizontal and vertical directions R times each, we only need $2R$ additions. All multiplications can be handled much more efficiently as shift operations. Despite the efficient implementation of binomial smoothing filters \mathcal{B}^R by cascaded convolution with \mathcal{B} , the number of computations increases drastically because the smoothing distance σ is only proportional to the square root of R according to (25.35). Doubling σ quadruples the number of computations.

Multistep Averaging

The problem of slow large-scale averaging originates from the small distance between the pixels averaged in the elementary $\mathcal{B} = 1/2 [1 \ 1]$ mask. This problem can

be overcome if the same elementary averaging process is used with more-distant pixels:

$$\mathcal{B}_{2x} = \frac{1}{4} [1 \ 0 \ 2 \ 0 \ 1], \quad \mathcal{B}_{2y} = \frac{1}{4} \begin{bmatrix} 1 \\ 0 \\ 2 \\ 0 \\ 1 \end{bmatrix}. \quad (25.39)$$

The subscripts in these masks denote the stepping width and coordinate direction. The standard deviation of these filters is proportional to the distance between the pixels. The most efficient implementations are multistep masks along the axes. Because of separability, this approach can be applied to image data of arbitrary dimensions.

The problem with these filters is that they perform subsampling. Consequently, they are no longer smoothing filters for larger wavenumbers. Used individually, these filters are not useful.

Cascaded multistep binomial filtering with recursive step doubling

$$\underbrace{\mathcal{B}_{2^{S-1}x}^R \cdots \mathcal{B}_{8x}^R \mathcal{B}_{4x}^R \mathcal{B}_{2x}^R \mathcal{B}_x^R}_{S \text{ times}} \quad (25.40)$$

leads to a significant performance increase for large-scale smoothing. For normal separable binomial filtering, the number of computations is proportional to $\sigma^2 [\mathcal{O}(\sigma^2)]$. For multistep binomial filtering it depends only logarithmically on σ [$\mathcal{O}(\log \sigma^2)$].

The standard deviation of smoothing is

$$\sigma^2 = \underbrace{R/4 + R + 4R + \cdots + 4^{S-1}R}_{S \text{ times}} = \frac{R}{12}(4^S - 1) \quad (25.41)$$

and the transfer function is

$$\prod_{s=0}^{S-1} \cos^R(2^{s-1}\pi\tilde{k}). \quad (25.42)$$

Thus, for S steps only RS additions are required, while the standard deviation grows exponentially with $\approx \sqrt{R/12} \cdot 2^S$.

With the parameter R , the degree of isotropy and the degree of residual inhomogeneities in the transfer function can be adjusted. For the most efficient implementation with $R = 2$ ($\mathcal{B}^2 = [1 \ 2 \ 1]/4$ in each direction), the residual side peaks at high wavenumbers with maximal amplitudes up to 0.08 still cause significant disturbances. With the next larger odd-sized masks

($R = 4$, $B^4 = [1\ 4\ 6\ 4\ 1]/16$ in each direction) these residual side peaks at high wavenumbers are suppressed well below 0.005.

Nonlinear Averaging

Linear smoothing filters cannot distinguish between a useful feature and noise. This property can be best demonstrated in the Fourier space (Fig. 25.7). White noise is added to the image. Because of the linearity of the Fourier transform, the Fourier transform of the white noise adds directly to the Fourier transform of the image.

Any linear filter operator works in such a way that the Fourier transform of the image is multiplied by the Fourier transform of the filter. The result is that at each wavenumber the noise level and the image features are attenuated by the same factor. Thus, nothing has improved at all. The signal-to-noise ratio is just the same. The noise level is reduced but so is the signal.

It is obvious that more-complex approaches than linear filtering are required. Common to all these approaches is that in one or the other way the filters are dependent on the context. Therefore a kind of control strategy is an important part of adaptive filtering that tells us which filter or in which way a filter has to be applied at a certain point in the image. In the following sections, some classes of nonlinear filter techniques are discussed.

Problem-Specific Nonlinear Filters. This approach is the oldest one and works if a certain specific type of distortion is present in an image. A well-known example is the *median filter*, which can excellently remove single-distorted pixels with minimum changes to the im-

age features. This approach, of course, only works for the type of distortion it is designed for. A median filter is excellent for removing a single pixel that has a completely incorrect gray value because of a transmission or data error; it is less well suited, however, to the reduction of white noise.

Weighted Averaging. So far, each pixel was treated equally assuming that the information it carries is of equal significance. While this seems to be a reasonable first approximation, it is certain that it cannot be generally true. Already during image acquisition, the sensor area may contain bad sensor elements that lead to erroneous gray values at certain positions in the image. Likewise, transmission errors may occur so that individual pixels may carry wrong information. In one way or another we may attach a certainty measurement to each data point.

Once a certainty measurement has been attached to a pixel, it is obvious that standard convolution operators are no longer a good choice. Instead, the weight we attach to the pixel has to be considered when performing any kind of operation with it. Each pixel enters the convolution sum with a weighting factor associated with it. This kind of approach is called *normalized convolution*. Thus, normalized convolution requires two images. One is the image to be processed, the other is an image with the weighting factors:

$$G' = \frac{H * (W \cdot G)}{H * W}, \quad (25.43)$$

where H is any convolution mask, G is the image to be processed, and W is the image with the weighting factors. A normalized convolution with the mask H essentially transforms the set of the image G and the weighting image W into a new image G' and a new weighting image $W' = H * W$, which can undergo further processing. *Standard* convolution can be regarded as a special case of normalized convolution where all pixels are assigned the same weighting factor and a weighting image is not required, since the factor remains constant.

Actually, this type of approach seems very natural to a scientist or engineer, as they are used to qualifying any data by a measurement error, which is then used in any further evaluation of the data. Normalized convolution applies this common principle to image processing.

The power of this approach is related to the fact that there are varied possibilities for the definition of the certainty of the measurement; it does not only have to be related to a direct measurement error of a single pixel. If we are, for example, interested in computing an estimate

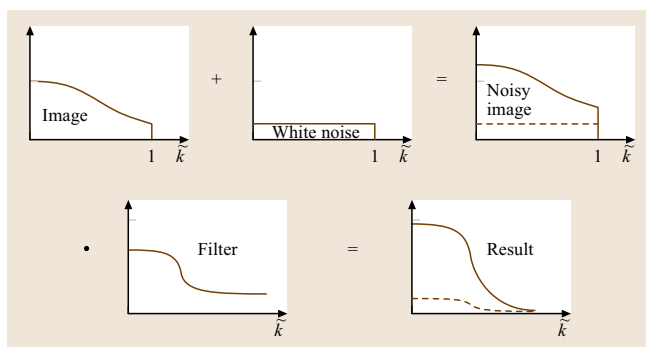


Fig. 25.7 A linear smoothing filter does not distinguish between useful features and noise in an image. It reduces the feature and the noise amplitudes equally at each wave number so that the signal-to-noise ratio remains the same

of the mean gray value in an object, we could devise a kind of certainty measurement that analyzes neighborhoods and attaches low weighting factors where we suspect an edge so that these pixel do not contribute much to the mean gray value or feature of the object. In a similar way, we could, for instance, also check how likely the gray value of a certain pixel is if we suspect some distortion by transmission errors or defective pixels. If the certainty measurement of a certain pixel is below a critical threshold, it is effectively replaced by a weighted value from the surrounding pixels.

Adaptive Filtering. *Adaptive filters* in the narrower sense use a different strategy than normalized convolution. Now, the filter operation itself is made dependent on the neighborhood. Adaptive filtering can best be explained by a classical application, the suppression of noise without significant blurring of image features. The basic idea of adaptive filtering is that in certain neighborhoods a smoothing operation can be applied without blurring structures. If, for instance, the neighborhood is flat, it can be assumed that this is an area within an object of constant features and thus an isotropic smoothing operation can be to this pixel to reduce the noise level. If an edge is present in the neighborhood, some smoothing is still possible, namely along the edge. In this way, some noise is removed but the edge is not blurred. With this approach, we need a kind of large filter set of directional smoothing operations. Because of the many filters involved, it appears that adaptive filtering might be a very computational-intensive approach; this is indeed the case if either the coefficients of the filter to be applied have to be computed for every pixel or if a large set of filters has to be used. With the discovery of *steerable filters* [25.2], however, adaptive filtering techniques have become attractive and computationally much more efficient.

With this approach a small set of base filters is used to compute a set of filtered images. Then, these images are interpolated using parameters that depend on the adjustable parameters. In operator notation this reads

$$\mathcal{H}(\alpha) = \sum_{p=1}^P f_p(\alpha) \mathcal{H}_p \quad (25.44)$$

where \mathcal{H}_p is the p -th filter and $f_p(\alpha)$ is a scalar function of the steering parameter α . Two problems must be solved to use steerable filters. First, and most basically, it is not clear that such a filter base H_p exists at all. Second, the relation between the

steering parameter(s) α and the interpolation coefficients f_p must be found. If the first problem is solved, we mostly get the solution to the second for free.

A simple example of a steerable filter is directional smoothing. A directional smoothing filter is to be constructed with the following transfer function

$$\hat{h}_\theta(k, \phi) = f(k) \cos^2(\phi - \theta). \quad (25.45)$$

In this equation cylinder coordinates (k, ϕ) are used in the Fourier domain. The filter in (25.45) is a *polar separable filter* with an arbitrary radial function $f(k)$. This radial component provides an isotropic bandpass filtering.

The steerable angular term is given by $\cos^2(\phi - \theta)$. Structures oriented into the direction θ remain in the image, while those perpendicular to θ are completely filtered out. The angular width of the directional filter is $\pm 45^\circ$.

Using elementary trigonometry it can be shown that this filter can be computed from only two base filters in the following way:

$$\begin{aligned} \hat{h}_\theta(k, \phi) = \frac{1}{2} + \frac{1}{2} [\cos(2\theta) \hat{H}_0(k, \phi) \\ + \sin(2\theta) \hat{H}_{\pi/4}(k, \phi)] \end{aligned} \quad (25.46)$$

with the filter base

$$\begin{aligned} \hat{h}_0(k, \phi) &= f(k) \cos^2 \phi, \\ \hat{h}_{\pi/4}(k, \phi) &= f(k) \sin^2(\phi). \end{aligned} \quad (25.47)$$

The two base filters are directed towards 0° and 45° . The directional filter \hat{h}_θ can be steered into any direction between -90° and 90° .

Using separable filters, a polar separable directional filter can be approximated only in a limited wavenumber range. Thus $f(k)$ must be a bandpass filter. The following filter set turns out to be a good approximation. It uses only binomial filters along the axes (\mathcal{B}_x and \mathcal{B}_y) and diagonals (\mathcal{B}_{x-y} and \mathcal{B}_{x+y}) with equal variance:

$$\begin{aligned} \mathcal{H}_0 &= \frac{1 - \mathcal{B}_x^{2R} \mathcal{B}_y^{2R} - (\mathcal{B}_x^{2R} - \mathcal{B}_y^{2R})}{2}, \\ \mathcal{H}_{\pi/4} &= \frac{1 - \mathcal{B}_x^{2R} \mathcal{B}_y^{2R} - (\mathcal{B}_{x+y}^R - \mathcal{B}_{x-y}^R)}{2}. \end{aligned} \quad (25.48)$$

For small wavenumbers the transfer function of the filter steered in the direction θ agrees with the required form (25.45). Thus it is not surprising that this filter

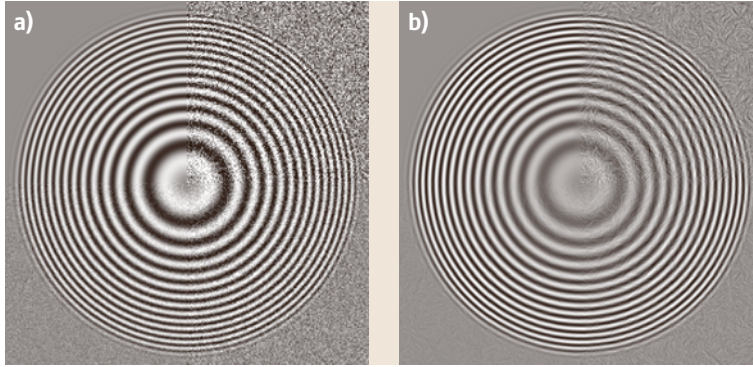


Fig. 25.8 (a) Ring test image with an amplitude of 100 superposed by zero mean normal distributed noise with a standard deviation of 10, 20, and 40 in three quadrants. (b) Image (a) after two iterations of steerable directional smoothing in the direction of constant gray values

can be used well to remove noise in images with directed gray value structures when this filter is steered to smooth in the direction of constant gray values (Fig. 25.8).

Nonlinear Diffusion Filters. In recent years, a whole new class of image processing operators has been investigated, known as *diffusion filters* [25.3]. Diffusion is a transport process that tends to level out concentration differences and thus work like a smoothing filter. Diffusion processes govern the transport of heat, matter, and momentum. To apply a diffusion process to an image, the gray value g is regarded as the concentration of a chemical species. The elementary law of diffusion states that the flux induced by a concentration difference is against the direction of the concentration gradient and proportional to it:

$$\mathbf{j} = -D\nabla g, \quad (25.49)$$

where the constant D is known as the *diffusion coefficient*. Using the continuity equation

$$\frac{\partial g}{\partial t} + \nabla \cdot \mathbf{j} = 0, \quad (25.50)$$

the nonstationary diffusion equation is

$$\frac{\partial g}{\partial t} = \nabla \cdot (D\nabla g). \quad (25.51)$$

For the case of a homogeneous diffusion process (D does not depend on the position), the equation reduces to

$$\frac{\partial g}{\partial t} = D\Delta g. \quad (25.52)$$

The general solution to this equation is equivalent to a convolution with a smoothing mask. A spatial Fourier transform, which results in

$$\frac{\partial \hat{g}(\mathbf{k})}{\partial t} = -D|\mathbf{k}|^2 \hat{g}(\mathbf{k}), \quad (25.53)$$

reduces the equation to a linear first-order differential equation with the general solution

$$\hat{g}(\mathbf{k}, t) = \exp(-D|\mathbf{k}|^2 t) \hat{g}(\mathbf{k}, 0), \quad (25.54)$$

where $\hat{g}(\mathbf{k}, 0)$ is the Fourier-transformed image at time zero. Multiplication of $\hat{g}(\mathbf{k}, 0)$ in the Fourier space with the Gaussian function $\exp(-|\mathbf{k}|^2/(2\sigma_k^2))$ with $\sigma_k^2 = 1/(2Dt)$ as given by (25.54) is equivalent to a convolution with the same function but of reciprocal width. Thus,

$$g(\mathbf{x}, t) = \frac{1}{2\pi\sigma^2(t)} \exp\left(-\frac{|\mathbf{x}|^2}{4Dt}\right) g(\mathbf{x}, 0). \quad (25.55)$$

Equation (25.55) establishes the equivalence between a diffusion process and convolution with a Gaussian kernel. In the discrete case, the Gaussian kernel can be replaced by binomial filters.

Given the equivalence between convolution and a diffusion process, it is possible to adapt smoothing to the local image structure by making the diffusion constant dependent on the position (*inhomogeneous diffusion*) and/or the direction (*anisotropic diffusion*).

To avoid smoothing of edges, it appears logical to attenuate the diffusion coefficient there. Thus, the diffusion coefficient is made dependent on the strength of the edges as given by the magnitude of the gradient

$$D(g) = D(|\nabla g|). \quad (25.56)$$

Perona and Malik [25.4] used the following dependency of the diffusion coefficient on the magnitude of the gradient:

$$D = D_0 \frac{\lambda^2}{|\nabla g|^2 + \lambda^2}, \quad (25.57)$$

where λ is an adjustable parameter. For small gradients $|\nabla g| \ll \lambda$, D approaches D_0 ; for high gradients $|\nabla g| \gg \lambda$, D tends to zero.

As simple and straightforward as this idea appears, it is not without problems. Depending on the functional form of D on ∇g , the diffusion process may become unstable, resulting even in steepening of the edges. A safe way to avoid this problem is to use a regularized gradient obtained from a smoothed version of the image [25.3].

Inhomogeneous diffusion has one significant disadvantage: it stops diffusion completely and in all directions at edges, leaving them noisy. Edges are, however, only blurred by diffusion perpendicular to them while diffusion parallel to edges is even advantageous since it stabilizes the edge.

An approach that makes diffusion independent of the direction of edges is known as anisotropic diffusion. With this approach, the flux is no longer parallel to the gradient. Therefore, the diffusion can no longer be described by a scalar diffusion coefficient as in (25.49). Now, a *diffusion tensor* is required:

$$j = -D\nabla g = - \begin{pmatrix} D_{11} & D_{12} \\ D_{12} & D_{22} \end{pmatrix} \begin{pmatrix} \partial g / \partial x \\ \partial g / \partial y \end{pmatrix}. \quad (25.58)$$

The properties of the diffusion tensor can best be seen if the symmetric tensor is brought into its principal axis system by a rotation of the coordinate system. Then, (25.58) reduces to

$$j = - \begin{pmatrix} D_{x'} & 0 \\ 0 & D_{y'} \end{pmatrix} \begin{pmatrix} \partial g / \partial x' \\ \partial g / \partial y' \end{pmatrix} = - \begin{pmatrix} D_{x'} \partial g / \partial x' \\ D_{y'} \partial g / \partial y' \end{pmatrix}. \quad (25.59)$$

Now, the diffusion in the two directions of the axes is decoupled. The two coefficients on the diagonal $D_{x'}$ and $D_{y'}$ are the *eigenvalues* of the diffusion tensor. In analogy to isotropic diffusion, the general solution of the anisotropic diffusion can be written

$$\hat{g}(\mathbf{x}, t) = \frac{1}{2\pi\sigma_{x'}(t)\sigma_{y'}(t)} \exp\left(-\frac{x'^2}{4D_{x'}t}\right) \times \exp\left(-\frac{y'^2}{4D_{y'}t}\right) g(\mathbf{x}, 0) \quad (25.60)$$

in the spatial domain, provided that the diffusion tensor does not depend on the position.

This means that anisotropic diffusion is equivalent to cascaded convolution with two 1-D Gaussian convolution kernels that are steered into the directions of the principal axes of the diffusion tensor.

If one of the two eigenvalues of the diffusion tensor is significantly larger than the other, diffusion occurs

only in the direction of the corresponding eigenvector. Thus the gray values are smoothed only in this direction. Implementation details for nonlinear diffusion filters are given in [25.3, 5].

25.1.5 Edge and Line Extraction

Averaging filters suppress structures with high wavenumbers. Edge detection requires a filter operation that emphasizes the spatial changes in signal values and suppresses areas with constant values. Derivative operators are suitable for such an operation in the one-dimensional case. The first derivative shows an extreme at the edge (maximal positive or negative steepness), while the second derivative crosses zero (vanishing curvature) where the edge has its steepest ascent or descent. Both criteria can be used to detect edges.

In higher dimensions the description of signal change is more complex. In 2-D images edges, corners, lines, and local extremes can be distinguished as relevant features for image processing. At an *edge*, we have a large change of the signal value perpendicular to the direction of the edge, but in the direction of the edge the change is low. However, if the curvature perpendicular to the gradient is high, the edge becomes a *corner*. A *line* is characterized by low first- and second-order derivatives along the line and a maximal curvature perpendicular to the direction of the line. *Local extremes* are characterized by zero first-order derivatives, but large curvatures in all directions.

In three dimensions, i.e., *volumetric images*, there can be *surfaces* with a strong first-order change in the direction perpendicular to the surface and low slopes and curvatures in the two directions within the surface. At an edge, there are low signal changes only in the direction of the edge, while at a corner the signal changes in all directions. All the local features described in multi-dimensional signals can be well represented with first- and second-order derivatives.

First-Order Derivation, Gradient

A p -th-order *partial derivative* operator corresponds to multiplication by $(2\pi i k)^p$ in wavenumber space. The first-order partial derivatives into all directions of a W -dimensional signal form the W -dimensional *gradient vector*:

$$\nabla = \left(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \dots, \frac{\partial}{\partial x_W} \right)^T \circ \bullet 2\pi i \mathbf{k}. \quad (25.61)$$

The magnitude of the gradient vector,

$$|\nabla| = \|\nabla\|_2 = (\nabla^T \nabla)^{1/2} = \left[\sum_{w=1}^W \left(\frac{\partial}{\partial x_w} \right)^2 \right]^{1/2}, \quad (25.62)$$

is invariant to rotation of the coordinate system and thus a good measure for edge strength.

First-order discrete differences are the simplest approximation to compute the gradient vector. For the first partial derivative in the x direction, the symmetric difference is the most useful, with the convolution mask

$$D_{2x} = 1/2[1 \ 0 \ -1] \quad (25.63)$$

and the transfer function

$$\hat{d}_{2x} = i \sin(\pi \tilde{k}_x). \quad (25.64)$$

Second-Order Derivation, Curvature

Second-order derivatives can be used to detect edges as zero crossings and lines and other second-order features by measurement of the *curvature*. All possible combinations of second-order partial differential operators of a W -dimensional signal form a symmetric $W \times W$ matrix, known as the *Hessian matrix*:

$$\mathbf{H} = \begin{pmatrix} \frac{\partial^2}{\partial x_1^2} & \frac{\partial^2}{\partial x_1 x_2} & \cdots & \frac{\partial^2}{\partial x_1 x_W} \\ \frac{\partial^2}{\partial x_1 x_2} & \frac{\partial^2}{\partial x_2^2} & \cdots & \frac{\partial^2}{\partial x_2 x_W} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2}{\partial x_1 x_W} & \frac{\partial^2}{\partial x_2 x_W} & \cdots & \frac{\partial^2}{\partial x_W^2} \end{pmatrix}. \quad (25.65)$$

It is always possible to find a coordinate transform \mathbf{R} into the *principal coordinate system* so that the Hessian matrix becomes diagonal. In two dimensions this is

$$\mathbf{H}' = \begin{pmatrix} \frac{\partial^2}{\partial x'^2} & 0 \\ 0 & \frac{\partial^2}{\partial y'^2} \end{pmatrix}. \quad (25.66)$$

The gradient has only one nonzero component in the principal coordinate system. This is not the case for curvatures. Generally, *all* curvatures are nonzero in the principal coordinate system.

There are two curvature parameters that are invariant to a rotation of the coordinate system. The first is the trace of this matrix, i.e., the sum of the diagonal, called the *Laplacian operator* or the *mean curvature* and denoted by Δ :

$$\Delta = \text{tr } \mathbf{H} = \sum_{w=1}^W \frac{\partial^2}{\partial x_w^2} \quad \circ \bullet \quad -4\pi k^2. \quad (25.67)$$

The second invariant is the Gaussian curvature, which is equal to the determinant of the Hessian matrix:

$$\det \mathbf{H} = \frac{\partial^2}{\partial x'^2} \frac{\partial^2}{\partial y'^2} = \frac{\partial^2}{\partial x^2} \frac{\partial^2}{\partial y^2} - \left(\frac{\partial^2}{\partial x \partial y} \right)^2. \quad (25.68)$$

The simplest discrete approximations of second-order derivative filters are the following second-order differences:

$$\begin{aligned} D_x^2 &= \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} \\ L &= \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \end{aligned} \quad (25.69)$$

Regularized Edge Detection

The edge detectors discussed so far are still poor performers, especially in noisy images. Because of their small mask sizes, they are most sensitive to high wavenumbers. At high wavenumbers there is often more noise than signal in images. Thus an optimum edge detector is tuned to the scale (wavenumber range) with the maximum signal-to-noise ratio. Consequently, we must design filters that perform a derivation in one direction but also smooth the signal in all directions.

Smoothing is particularly effective in higher-dimensional signals because it does not blur the edge in all directions perpendicular to the direction of the gradient. Derivative filters that incorporate smoothing are also known as *regularized edge detectors* because they result in robust solutions for the ill-posed problem of estimating derivatives from discrete signals.

2 × 2 Cross-Smoothing Operator. The smallest cross-smoothing derivative operator has the following 2 × 2 masks

$$\frac{1}{2} \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} \quad \text{and} \quad \frac{1}{2} \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}. \quad (25.70)$$

There is nothing that can be optimized with this small filter mask. Because of the imperfect approximation, the direction of the gradient computed with this operator has errors of up to 5° at large wavenumbers ($\tilde{k} = 0.5$) and the computed magnitude of the gradient depends on the direction of the edge [25.5].

Sobel Edge Detector. The *Sobel operator* is the smallest difference filter with an odd number of coefficients that

averages the image in the direction perpendicular to the differentiation:

$$\frac{1}{8} \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \quad \frac{1}{8} \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}. \quad (25.71)$$

The errors in the magnitude and direction of the gradient are similar to the 2×2 cross-smoothing difference operator.

Optimized Regularized Edge Detectors. An optimized regularized derivative operator with about a 10 times lower error in the estimate of the direction of edges is [25.6]:

$$\frac{1}{32} \begin{bmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix}, \quad \frac{1}{32} \begin{bmatrix} 3 & 10 & 3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{bmatrix}. \quad (25.72)$$

Similar optimizations are possible for larger-sized regularized derivative filters [25.6].

25.1.6 Direction and Orientation

A local neighborhood could also contain more-complex patterns than edges and constant regions. If it contains oriented patterns, it is denoted as a simple neighborhood or *linear symmetry* [25.7]. In 2-D images this could be edges between objects of constant intensity, oriented patterns or, in a space–time image, objects moving with a constant velocity (Fig. 25.9). Although the three examples refer to entirely different image data, they have in common that the local structure is characterized by an orientation,

Simple Neighborhood in the Spatial Domain

A local neighborhood with ideal local orientation is characterized by the fact that the gray value only changes in one direction. In all other directions it is constant. If the coordinate system is oriented along the principal directions, the gray values become a 1-D function of only one coordinate. Generally, we will denote the direction of local orientation with a unit vector \bar{n} perpendicular to the lines of constant gray values. Then, a simple neighborhood is mathematically represented by

$$g(\mathbf{x}) = g(\mathbf{x}^T \bar{n}), \quad (25.73)$$

where we denote the scalar product simply by $\mathbf{x}^T \bar{n}$. Equation (25.73) is also valid for image data with more

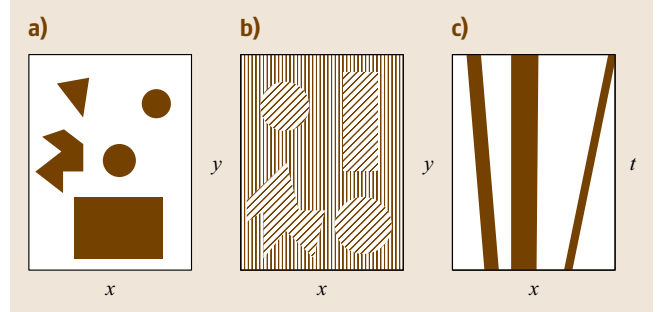


Fig. 25.9a–c Three different interpretations of local structures in 2-D images: **(a)** edge between uniform object and background; **(b)** orientation of pattern; **(c)** orientation in a 2-D space–time image indicating the velocity of 1-D objects

than two dimensions. The projection of the vector \mathbf{x} onto the unit vector \bar{n} makes the gray values depend only on a scalar quantity, the coordinate in the direction of \bar{n} (Fig. 25.10). The gradient lies in the direction of \bar{n} .

Representation in the Fourier Domain

A simple neighborhood also has a special form in Fourier space. If the whole image is described by (25.73), i. e., \bar{n} does not depend on the position, then the Fourier transform must be confined to a line. The direction of the line is given by \bar{n} :

$$g(\mathbf{x}^T \bar{n}) \quad \longleftrightarrow \quad \hat{g}(k) \delta[\mathbf{k} - \bar{n}(k^T \bar{n})], \quad (25.74)$$

where k denotes the coordinate in the Fourier domain in the direction of \bar{n} . The argument in the δ function is only zero when \mathbf{k} is parallel to \bar{n} . If (25.74) is restricted to a local neighborhood around \mathbf{x}_0 , this corresponds to a multiplication of $g(\mathbf{x}^T \bar{n})$ by a window function $w(\mathbf{x} - \mathbf{x}_0)$ in the spatial domain. The size and shape of the neighborhood is determined by the window function. A window function that gradually decreases to zero

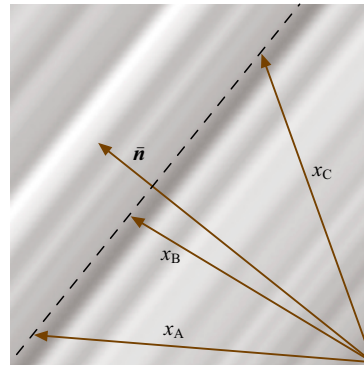


Fig. 25.10

Illustration of a linear symmetric or simple neighborhood. The gray values depend only on a coordinate given by a unit vector \bar{n}

diminishes the influence of pixels as a function of their distance from the outer pixel. Thus,

$$w(\mathbf{x}-\mathbf{x}_0)\cdot g(\mathbf{x}^T\overline{\mathbf{n}}) \quad \circ\bullet$$
$$\hat{w}(\mathbf{k})\ast \hat{g}(k)\delta[\mathbf{k}-\overline{\mathbf{n}}(k^T\overline{\mathbf{n}})]\,, \tag{25.75}$$

where $\hat{w}(\mathbf{k})$ is the Fourier transform of the window function.

The limitation to a local neighborhood thus blurs the line in Fourier space to a *sausage-like* shape. Because of the reciprocity of scales between the two domains, its thickness is inversely proportional to the size of the window. Thus the accuracy of the orientation estimate is directly related to the ratio of the window size to the wavelength of the smallest structures in the window.

The Structure Tensor

A suitable representation should be able to determine a unique orientation (given by a unit vector $\overline{\mathbf{n}}$) and to distinguish constant neighborhoods from neighborhoods without local orientation.

Such a representation can be introduced by the following optimization strategy to determine the orientation of a simple neighborhood. The optimum orientation is defined as the orientation that shows the least deviations from the directions of the gradient. A suitable measure for the deviation must treat gradients pointing in opposite directions equally. The squared scalar

product between the gradient vector and the unit vector representing the local orientation $\overline{\mathbf{n}}$ meets this criterion:

$$(\nabla g^T\overline{\mathbf{n}})^2 = |\nabla g|^2 \cos^2\left[\angle(\nabla g, \overline{\mathbf{n}})\right] \, . \tag{25.76}$$

This quantity is proportional to the cosine squared of the angle between the gradient vector and the orientation vector and is thus maximal when ∇g and $\overline{\mathbf{n}}$ are parallel or antiparallel, and zero if they are perpendicular to each other. Therefore, the following integral is maximized in a two-dimensional local neighborhood:

$$\int w(\mathbf{x}-\mathbf{x}')[\nabla g(\mathbf{x}')^T\overline{\mathbf{n}}]^2 \, \mathrm{d}^2x' \, , \tag{25.77}$$

where the window function w determines the size and shape of the neighborhood around a point \mathbf{x} at which the orientation is averaged. The maximization problem must be solved for each point \mathbf{x} . Equation (25.77) can be rewritten in the following way:

$$\overline{\mathbf{n}}^T \mathbf{J} \overline{\mathbf{n}} \rightarrow \max \tag{25.78}$$

with

$$\mathbf{J} = \int w(\mathbf{x}-\mathbf{x}')[\nabla g(\mathbf{x}')\nabla g(\mathbf{x}')^T] \, \mathrm{d}^2x' \, ,$$

where $\nabla g\nabla g^T$ denotes an outer (Cartesian) product. The components of this symmetric 2×2 tensor, named the

Table 25.2 Eigenvalue classification of the structure tensor in 2-D images

Condition	Rank(J)	Description
$\lambda_1 = \lambda_2 = 0$	0	Both eigenvalues are zero. The mean squared magnitude of the gradient ($\lambda_1 + \lambda_2$) is zero. The local neighborhood has constant values.
$\lambda_1 > 0, \lambda_2 = 0$	1	One eigenvalue is zero. The values do not change in the direction of the corresponding eigenvector. The local neighborhood is a simple neighborhood with ideal orientation (straight edge or 1-D texture).
$\lambda_1 > 0, \lambda_2 > 0$	2	Both eigenvalues are unequal to zero. The gray values change in all directions. In the special case of $\lambda_1 = \lambda_2$, we speak of an isotropic gray value structure as it changes equally in all directions.

Table 25.3 Eigenvalue classification of the structure tensor in 3-D (volumetric) images

Condition	Rank(J)	Description
$\lambda_1 = \lambda_2 = \lambda_3 = 0$	0	The gray values do not change in any direction; constant neighborhood.
$\lambda_1 > 0, \lambda_2 = \lambda_3 = 0$	1	The gray values change only in one direction. This direction is given by the eigenvector to the nonzero eigenvalue. The neighborhood includes a boundary between two objects (surface) or a layered texture. In a space–time image: constant motion of a spatially oriented pattern (<i>planar wave</i>).
$\lambda_1 > 0, \lambda_2 > 0, \lambda_3 = 0$	2	The gray values change in two directions and are constant in a third (edge or extruded texture). In a space–time image: constant motion of a spatially distributed pattern. The eigenvector to the zero eigenvalue gives the direction of the constant gray values.
$\lambda_1 > 0, \lambda_2 > 0, \lambda_3 > 0$	3	The gray values change in all three directions.

structure tensor, are

$$J_{pq}(\mathbf{x}) = \int_{-\infty}^{\infty} w(\mathbf{x} - \mathbf{x}') \left(\frac{\partial g(\mathbf{x}')}{\partial x'_p} \frac{\partial g(\mathbf{x}')}{\partial x'_q} \right) d^2 x' . \quad (25.79)$$

These equations indicate that a tensor is an adequate first-order representation of a local neighborhood. More-complex structures such as structures with multiple orientations cannot be distinguished.

By a rotation of the coordinate system, this can be brought into a diagonal form. Then, (25.78) reduces to

$$J' = (\bar{n}'_1, \bar{n}'_2) \begin{pmatrix} J'_{11} & 0 \\ 0 & J'_{22} \end{pmatrix} \begin{pmatrix} \bar{n}'_1 \\ \bar{n}'_2 \end{pmatrix} . \quad (25.80)$$

A unit vector $\bar{n}' = (\cos \theta \sin \theta)$ in the direction θ gives the values

$$J' = J'_{11} \cos^2 \theta + J'_{22} \sin^2 \theta .$$

Without loss of generality, we assume that $J'_{11} \geq J'_{22}$. Then, it is obvious that the unit vector $\bar{n}' = (1 \ 0)^T$ maximizes (25.80). The maximum value is J'_{11} . In conclusion, this approach not only yields a tensor representation for the local neighborhood but also shows the way to determine the orientation. Essentially, (25.78) constitutes an *eigenvalue problem*. The eigenvalues λ_w and eigenvectors \mathbf{e}_w of a 2×2 matrix are defined by

$$\mathbf{J} \mathbf{e}_w = \lambda_w \mathbf{e}_w . \quad (25.81)$$

An eigenvector \mathbf{e}_w of \mathbf{J} is thus a vector that is not turned in direction by multiplication by the matrix \mathbf{J} but is only multiplied by a scalar factor, the eigenvalue λ_w . This implies that the structure tensor becomes diagonal in a coordinate system that is spanned by the eigenvectors (25.80). For a symmetric matrix the eigenvalues are all real and nonnegative, and the eigenvectors form an orthogonal basis. According to the *maximization problem* formulated here, the eigenvector to the maximum eigenvalue gives the orientation of the local neighborhood.

Classification of Local Neighborhoods

The power of the tensor representation becomes apparent if we classify the eigenvalues of the structure tensor. The classifying criterion is the number of eigenvalues that are zero. If an eigenvalue is zero, this means that the gray values in the direction of the corresponding eigenvector do not change. The number of zero eigenvalues is also closely related to the rank of a matrix. The *rank* of a matrix is defined as the dimension of the subspace for which $\mathbf{J}\mathbf{k} \neq \mathbf{0}$. The space for which is $\mathbf{J}\mathbf{k} = \mathbf{0}$ is denoted

as the *null space*. The dimension of the null space is the dimension of the matrix minus the rank of the matrix and is equal to the number of zero eigenvalues. We will perform an analysis of the eigenvalues for two and three dimensions. In two and three dimensions, we can distinguish the cases summarized in Tables 25.2 and 25.3, respectively.

In practice, it will not be checked whether the eigenvalues are zero but below a critical threshold that is determined by the noise level in the image.

Orientation Vector

With the simple convolution and point operations discussed in the previous section, we computed the components of the structure tensor. In two dimensions, we can readily solve the eigenvalue problem. The orientation angle can be determined by rotating the inertia tensor into the principal axes coordinate system. The orientation angle is given by

$$\tan 2\theta = \frac{2J_{12}}{J_{22} - J_{11}} . \quad (25.82)$$

Because $\tan 2\theta$ is gained from a quotient, we can regard the dividend as the y and the divisor as the x component of a vector and can form the *orientation vector* \mathbf{o} , as introduced by Granlund [25.8]:

$$\mathbf{o} = \begin{pmatrix} J_{22} - J_{11} \\ 2J_{12} \end{pmatrix} . \quad (25.83)$$

The argument of this vector gives the orientation angle and the magnitude a certainty measure for the local *orientation*. The term orientation is used in all cases where an angle range of only 180° is required. It is not possible to distinguish between patterns that are rotated by 180° . Orientation is still, of course, a *cyclic* quantity.

The orientation vector can be well represented as a color image. It appears natural to map the certainty measure onto the luminance and the orientation angle as the hue of the color. Our attention is then drawn to the bright parts in the images where we can distinguish the colors well. The darker a color is, the more difficult it becomes to distinguish the different colors visually. In this way, our visual impression coincides with the orientation information in the image.

Structure Operator and Coherency

The orientation vector reduces local structure to local orientation. From three independent components of the symmetric tensor only two are used. When we fail to

observe an orientated structure in a neighborhood, we do not know whether no gray value variations or distributed orientations are encountered. This information is included in the not yet used component of the tensor, $J_{11} + J_{22}$, which gives the mean square magnitude of the gradient. Consequently, a well-equipped structure operator also needs to include the third component. A suitable linear combination is

$$s = \begin{pmatrix} J_{11} + J_{22} \\ J_{22} - J_{11} \\ 2J_{12} \end{pmatrix}. \quad (25.84)$$

This structure operator contains the two components of the orientation vector and, as an additional component, the mean square magnitude of the gradient, which is a rotation-invariant parameter. Comparing the latter with the magnitude of the orientation vector, a constant gray value area and an isotropic gray value structure without a preferred orientation can be distinguished. In the first case, both squared quantities are zero; in the second only the magnitude of the orientation vector. In the case of a perfectly oriented pattern, both quantities are equal. Thus their ratio seems to be a good *coherency measure*

c_c for local orientation:

$$c_c = \frac{\sqrt{(J_{22} - J_{11})^2 + 4J_{12}^2}}{J_{11} + J_{22}} = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}. \quad (25.85)$$

The coherency ranges from 0 to 1. For ideal local orientation ($\lambda_2 = 0, \lambda_1 > 0$) it is 1, for an isotropic gray value structure ($\lambda_1 = \lambda_2 > 0$) it is 0.

A color representation of the structure tensor requires only two slight modifications compared to the color representation for the orientation vector. First, instead of the length of the orientation vector, the squared magnitude of the gradient is mapped onto the intensity. Second, the coherency measure (25.85) is used as the saturation. In the color representation for the orientation vector, the saturation is always one. The angle of the orientation vector is still represented as the hue.

In practice, a slight modification of this color representation is useful. The squared magnitude of the gradient shows variations too large to be displayed in the narrow dynamic range of a display screen with only 256 luminance levels. Therefore, a suitable normalization is required. The basic idea of this normalization is to compare the squared magnitude of the gradient

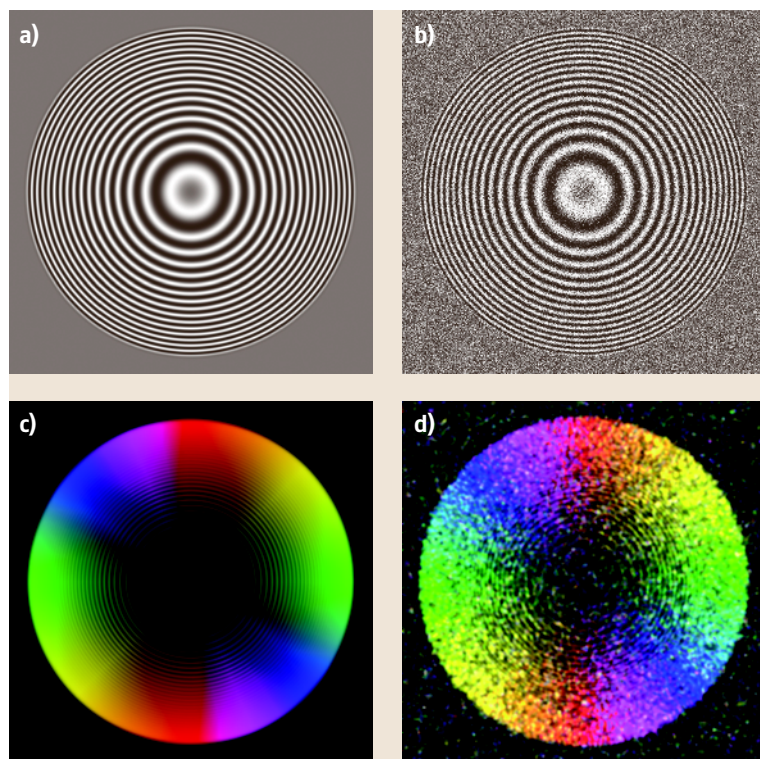


Fig. 25.11a–d Demonstration of the computation of the structure tensor with the ring test pattern: (a) ring without noise, (b) ring with a signal-to-noise ratio of 2, (c) color presentation of the structure tensor computed from (a), (d) the same from (b)

with the noise level. Once the gradient is well above the noise level it is regarded as a significant piece of information. This train of thoughts suggests the following normalization for the intensity I :

$$I = \frac{J_{11} + J_{22}}{(J_{11} + J_{22}) + \gamma \sigma_n^2}, \quad (25.86)$$

where σ_n is an estimate of the standard deviation of the noise level. This normalization provides a rapid transition of the luminance from 1, when the magnitude of the gradient is larger than σ_n , to 0 when the gradient is smaller than σ_n . The factor γ is used to optimize the display.

Implementation

The structure tensor can be computed straightforwardly as a combination of *linear convolution* and *nonlinear point operations*. The partial derivatives in (25.79) are approximated by discrete derivative operators. The integration weighted with the window function is replaced by a convolution with a smoothing filter that has the shape of the window function. If we denote the discrete partial derivative operator with respect to the coordinate p by the operator \mathcal{D}_p and the (isotropic) smoothing operator by \mathcal{B} , the local structure of a gray value image can be computed with the structure tensor

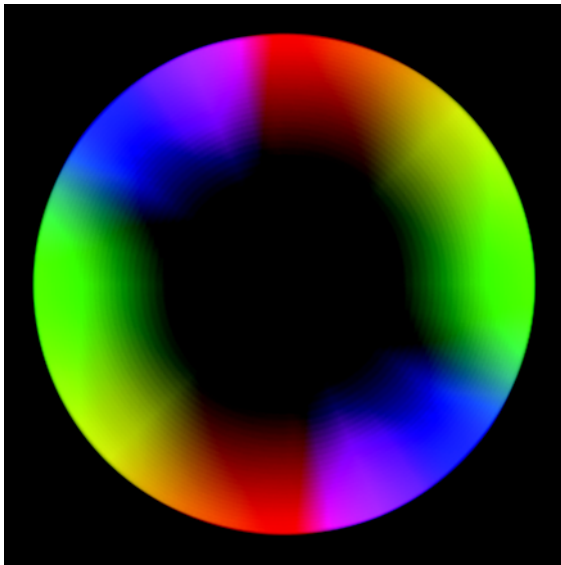


Fig. 25.12 Color presentation of the energy tensor (compare with Fig. 25.11c) [25.9]

operator

$$\mathcal{J}_{pq} = \mathcal{B}(\mathcal{D}_p \cdot \mathcal{D}_q). \quad (25.87)$$

The equation is written in an operator notation. Pixel-wise multiplication is denoted by the dot to distinguish it from successive application of convolution operators.

These operators are valid in images of any dimension $W \geq 2$. In a W -dimensional image, the structure tensor has $W(W+1)/2$ independent components, hence three in 2-D, six in 3-D, and ten in 4-D images. These components are best stored in a multichannel image with $W(W+1)/2$ components.

The smoothing operations consume the largest number of operations. Therefore, a fast implementation must, in the first place, apply a fast smoothing algorithm. A fast algorithm can be established based on the general observation that higher-order features always show a lower resolution than the features from which they are computed. This means that the structure tensor can be stored on a coarser grid and thus in a smaller image. A convenient and appropriate subsampling rate is to reduce the scale by a factor of two by storing only every second pixel in every second row.

The accuracy of the orientation angle depends strongly on the implementation of the derivative filters. The straightforward implementation of the algorithm using the standard derivative filter mask $(1/2)[1 \ 0 \ -1]$ or the *Sobel operator* results in surprisingly high errors, with a maximum error in the orientation angle of more than 7° at a wavenumber of $\tilde{k} = 0.7$. The error depends on both the wavenumber and the orientation of the local structure. For orientation angles in the direction of axes and diagonals, the error vanishes.

The error in the orientation angle can be suppressed significantly if better derivative filters are used. The little extra effort invested in optimizing the derivative filters thus pays off in an accurate orientation estimate (Fig. 25.11b). A residual angle error of less than 0.5° is sufficient for almost all applications. The various derivative filters discussed for edge and line extraction give the freedom to balance computational effort with accuracy.

Even with a low signal-to-noise ratio, the orientation estimate is still correct if a suitable derivative operator is used. With increasing noise level, the coherency decreases and the statistical error of the orientation angle estimate increases (Fig. 25.11d).

Energy Tensor

Recently, a phase-invariant extension of the structure tensor was proposed, named the *energy tensor*, \mathbf{E} , de-

defined as

$$\mathbf{E} = \nabla g, \nabla g^T - g, \mathbf{H}g$$

$$= \begin{pmatrix} g_x^2 - gg_{xx} & g_x g_y - gg_{xy} \\ g_x g_y - gg_{xy} & g_y^2 - gg_{yy} \end{pmatrix}, \quad (25.88)$$

where \mathbf{H} is the Hessian matrix. The energy tensor can be computed accurately using the filter optimization techniques described in Sect. 25.1.5 if the second-order derivative filters are computed by consecutive application of first-order filters.

The energy tensor requires no averaging. Firstly, this constitutes a significant saving in terms of number of computing operations. Secondly, the energy tensor gives better results. This can be demonstrated by computing the structure tensor and energy tensor of the ring test pattern in Fig. 25.11. While the averaging of the structure tensor is not sufficient at large wavelengths (small wavenumbers) close to the center of the ring pattern, this effect does not show up in the energy tensor (Fig. 25.12).

25.1.7 Local Wavenumber and Local Phase

Local structure is not only characterized by orientation but also by a local scale that can be represented by a local wavenumber, i.e., the number of periods of a structure per unit length. The determination of the amplitude, phase, wavenumber, and orientation is, of course, a central image processing task for any type of technique delivering fringe patterns.

Phase

The key to determining the local wavenumber is the *phase* of the signal. Consider the one-dimensional periodic signal

$$g(x) = g_0 \cos(kx). \quad (25.89)$$

The argument of the cosine function is known as the phase of the periodic signal

$$\phi(x) = kx. \quad (25.90)$$

Thus the phase is a *linear function* of the position and the wavenumber. The wavenumber of the periodic signal is given by the first-order spatial derivative of the phase signal

$$\frac{\partial \phi(x)}{\partial x} = k. \quad (25.91)$$

Hilbert Filter and Analytic Signal

The key to determining the phase is an operator that delays the signal by a phase of 90° . This operator would convert the $g(x) = g_0 \cos(kx)$ signal into a $g'(x) = -g_0 \sin(kx)$ signal. Using both signals, the phase of $g(x)$ can be computed by

$$\phi(g(x)) = \arctan\left(\frac{-g'(x)}{g(x)}\right). \quad (25.92)$$

As only the ratio of $g'(x)$ and $g(x)$ goes into (25.92), the phase is indeed independent of amplitude. Together with the signs of the two functions $g'(x)$ and $g(x)$, the phase can be computed over the full range of 360° .

Thus the phase of a signal is determined by a linear operator that shifts the phase of a signal by 90° . Such an operator is known as the *Hilbert filter* \mathbf{H} or *Hilbert operator* \mathcal{H} and has the transfer function

$$\hat{h}(k) = \begin{cases} i & k > 0 \\ 0 & k = 0 \\ -i & k < 0 \end{cases}. \quad (25.93)$$

The magnitude of the transfer function is 1, as the amplitude remains unchanged. As the Hilbert filter has a purely imaginary transfer function, it must have odd symmetry to generate a real-valued signal. Therefore positive wavenumbers are shifted by 90° ($\pi/2$) and negative wavenumbers by -90° ($-\pi/2$). A special situation is given for zero wavenumber, for which the transfer function is 0. A signal with zero wavenumber is a constant and can be regarded as a cosine function with infinite wavenumber sampled at the phase zero. Consequently, the Hilbert-filtered signal is the corresponding sine function at phase zero, i.e., zero.

Because of the discontinuity of the transfer function of the Hilbert filter at the origin, its point spread function is of infinite extent

$$h(x) = -\frac{1}{\pi x}. \quad (25.94)$$

The convolution with (25.94) can be written

$$g_h(x) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{g(x')}{x' - x} dx'. \quad (25.95)$$

This integral transform is known as the *Hilbert transform* [25.10].

Because the convolution mask of the Hilbert filter is infinite, it is impossible to design an exact discrete Hilbert filter for arbitrary signals. This is only possible if we restrict the class of signals to which it is applied. Thus

the following approach is taken to design an effective implementation of a Hilbert filter.

First, the filter should shift the phase by precisely $\pi/2$. This requirement comes from the fact that we cannot afford an error in the phase because it includes the position information. A wavenumber-dependent phase shift would cause wavenumber-dependent errors. This requirement is met by any convolution kernel of odd symmetry.

Second, the requirements for a magnitude of 1 can be relaxed if the Hilbert filter is applied to a band-passed signal. Then, the Hilbert filter must only show a magnitude of one in the pass-band range of the bandpass filter used. This approach avoids discontinuities in the transfer function at the wavenumber 0 and thus results in finite-sized convolution kernels. Optimized Hilbert filters are discussed by Jähne [25.11].

A real-valued signal and its Hilbert transform can be combined into a complex-valued signal by

$$g_a = g - i g_h. \quad (25.96)$$

This complex-valued signal is denoted as the *analytic function* or analytic signal. According to (25.96) the analytic filter has the point spread function

$$a(x) = 1 + \frac{i}{\pi x} \quad (25.97)$$

and the transfer function

$$\hat{a}(k) = \begin{cases} 2 & k > 0 \\ 1 & k = 0 \\ 0 & k < 0 \end{cases}. \quad (25.98)$$

Thus all negative wavenumbers are suppressed. Although the transfer function of the analytic filter is real, it results in a complex signal because it is asymmetric. For a real signal no information is lost by suppressing the negative wavenumbers. They can be reconstructed as the Fourier transform of a real signal is Hermitian. The analytic signal can be regarded as just another representation of a real signal with two important properties. The magnitude of the analytic signal gives the *local amplitude*

$$|a|^2 = g^2 + g_h^2. \quad (25.99)$$

and the argument the *local phase*

$$\phi = \arg(a) = \arctan\left(\frac{-g_h}{g}\right). \quad (25.100)$$

The original signal and its Hilbert transform can be obtained from the analytic signal using (25.96) by

$$\begin{aligned} g(x) &= [g_a(x) + g_a^*(x)]/2 \\ g_h(x) &= i[g_a(x) - g_a^*(x)]/2. \end{aligned} \quad (25.101)$$

To determine the local wavenumber, the first spatial derivative of the phase signal is computed (25.91). This derivative has to be applied in the same direction as the Hilbert or quadrature filter has been applied. However, direct computation of the partial derivatives is not advisable, because of the inherent discontinuities in the phase signal. A phase computed with the inverse tangent restricts the phase to the main interval $[-\pi, \pi[$ and thus inevitably leads to a *phase wrapping* from π to $-\pi$ with the corresponding discontinuities.

If only the local wavenumber is of interest, this problem can be avoided by computing the phase derivative directly from the derivatives of g and g_h [25.12]. The result is

$$k = \frac{\partial}{\partial x} \arctan(-g_h/g) = \frac{g_h \partial g / \partial x - g \partial g_h / \partial x}{g^2 + g_h^2}. \quad (25.102)$$

This procedure to compute the phase gradient also eliminates the need to use trigonometric functions and is, therefore, significantly faster.

Phase in Higher-Dimensional Signals and the Monogenic Signal

The concept of the analytic signal makes it possible to extend the ideas of local phase into multiple dimensions. The transfer function of the analytic operator uses only the positive wavenumbers, i.e., only half of the Fourier space. If we extend this partitioning to multiple dimensions, we have more than one choice to partition the Fourier space into two half-spaces. Instead of the wavenumber, we can take the scalar product between the wavenumber vector \mathbf{k} and any unit vector $\bar{\mathbf{n}}$ and suppress the half-space for which the scalar product $\mathbf{k}\bar{\mathbf{n}}$ is negative:

$$\hat{a}(\mathbf{k}) = \begin{cases} 2 & \mathbf{k}\bar{\mathbf{n}} > 0 \\ 1 & \mathbf{k}\bar{\mathbf{n}} = 0 \\ 0 & \mathbf{k}\bar{\mathbf{n}} < 0 \end{cases}. \quad (25.103)$$

The unit vector $\bar{\mathbf{n}}$ gives the direction in which the Hilbert filter is to be applied. The definition (25.103) of the transfer function of the analytic signal implies that the Hilbert operator can only be applied to directionally filtered signals. This results from the following considerations. For one-dimensional signals we have seen that a discrete Hilbert filter does not work well for small wavenumbers. In multiple dimensions this means that a Hilbert filter does not work well if $\mathbf{k}\bar{\mathbf{n}} \ll 1$. Thus no wavenumbers near an orthogonal to the direction of the Hilbert filter may exist, in order to avoid errors.

This fact makes the application of Hilbert filters and thus the determination of the local phase in higher-dimensional signals significantly more complex. It is not sufficient to use isotropic band-pass-filtered images. In addition, the band-pass-filtered images must be further decomposed into directional components. At least as many directional components as the dimensionality of the space are required.

The extension of the Hilbert transform from a 1-D signal to higher-dimensional signals is not satisfactory because it can only be applied to directionally filtered signals. For wavenumbers close to the separation plane, the Hilbert transform does not work. What is really required is an isotropic extension of the Hilbert transform.

A vector-valued extension of the analytic signal meets both requirements. It is known as the *monogenic signal* and was introduced to image processing by Felsberg and Sommer [25.13]. The monogenic signal is constructed from the original signal and its *Riesz transform*. The transfer function of the Riesz transform is given by

$$\hat{h}(\mathbf{k}) = i \frac{\mathbf{k}}{|\mathbf{k}|}. \quad (25.104)$$

The magnitude of the vector \mathbf{h} is 1 for all values of \mathbf{k} . The Riesz transform is thus isotropic. It also has odd symmetry because

$$\hat{h}(-\mathbf{k}) = -\hat{h}(\mathbf{k}). \quad (25.105)$$

The Riesz transform can be applied to a signal of any dimension. For a 1-D signal it reduces to the Hilbert transform.

For a 2-D signal the transfer function of the Riesz transform can be written using polar coordinates as

$$\hat{h}(\mathbf{k}) = i(\cos \theta, \sin \theta)^T. \quad (25.106)$$

The transfer function is similar to the transfer function for the gradient operator (25.61). The convolution mask or point spread function (PSF) of the Riesz transform is given by

$$h(\mathbf{x}) = -\frac{\mathbf{x}}{2\pi|\mathbf{x}|^3}. \quad (25.107)$$

The original signal and the signal convolved by the Riesz transform can be combined for a 2-D signal to the 3-D monogenic signal as

$$\mathbf{g}_m(\mathbf{x}) = (p, q_1, q_2)^T \quad (25.108)$$

with $p = g$, $q_1 = h_1 * g$, and $q_2 = h_2 * g$. The local amplitude of the monogenic signal is given as the norm of

the vector of the monogenic signal as in the case of the analytic signal (25.99):

$$|\mathbf{g}_m|^2 = p^2 + q_1^2 + q_2^2. \quad (25.109)$$

The monogenic signal does not only give an estimate for the *local phase* ϕ as the *analytic signal* does. The monogenic signal also gives an estimate of the *local orientation* θ by the following relations:

$$\begin{aligned} p &= a \cos \phi, \\ q_1 &= a \sin \phi \cos \theta, \\ q_2 &= a \sin \phi \sin \theta. \end{aligned} \quad (25.110)$$

Therefore the monogenic signal combines the estimation of local orientation and local phase. This is of high significance for image processing because the two most important features of a local neighborhood, the local orientation and the local wavenumber can be estimated in a unified way.

It is significantly more complex to compute the local wavenumber from the *monogenic signal*, because there are three signals in two dimensions. From (25.110) we obtain two different equations for the phase:

$$\phi_1 = \operatorname{arccot} \left(\frac{p \cos \theta}{q_1} \right), \quad \phi_2 = \operatorname{arccot} \left(\frac{p \sin \theta}{q_2} \right). \quad (25.111)$$

It is necessary to combine these equations because each of them gives no result for certain directions. The solution is use the *directional derivative*. When differentiating the phase in the direction of the wavenumber vector, the magnitude of the wavenumber vector is obtained:

$$k = \frac{\partial \phi}{\partial \mathbf{k}} = \cos \theta \frac{\partial \phi_1}{\partial x} + \sin \theta \frac{\partial \phi_2}{\partial y}. \quad (25.112)$$

The terms $\cos \theta$ and $\sin \theta$ can be obtained from (25.110):

$$\cos^2 \theta = \frac{q_1^2}{q_1^2 + q_2^2} \quad \text{and} \quad \sin^2 \theta = \frac{q_2^2}{q_1^2 + q_2^2}. \quad (25.113)$$

Then the magnitude of the wavenumber vector results in

$$k = \frac{p(q_{1x} + q_{2y}) - q_1 p_x - q_2 p_y}{p^2 + q_1^2 + q_2^2}. \quad (25.114)$$

The components of the wavenumber vector $\mathbf{k} = (k \cos \theta, k \sin \theta)$ can be computed by combining (25.114) and (25.112).

Quadrature Filters

Quadrature filters is an alternative approach to getting a pair of signals that differ only by a phase shift of

90° ($\pi/2$). It is easiest to introduce the complex form of the quadrature filters. Essentially, the transfer function of a *quadrature filter* is also zero for $k\bar{n} < 0$, like the transfer function of the analytic filter. However, the magnitude of the transfer function is not one but can be any arbitrary real-valued function $h(\mathbf{k})$:

$$\hat{q}(\mathbf{k}) = \begin{cases} 2h(\mathbf{k}) & k\bar{n} > 0 \\ 0 & \text{otherwise} \end{cases}. \quad (25.115)$$

The quadrature filter thus also transforms a real-valued signal into an analytical signal. In contrast to the analytical operator, a wavenumber weighting is applied. From the complex form of the quadrature filter, we can derive the real quadrature filter pair by observing that they are the part of (25.115) with even and odd symmetry. Thus

$$\begin{aligned} \hat{g}_+(\mathbf{k}) &= [\hat{q}(\mathbf{k}) + \hat{q}(-\mathbf{k})]/2, \\ \hat{g}_-(\mathbf{k}) &= [\hat{q}(\mathbf{k}) - \hat{q}(-\mathbf{k})]/2. \end{aligned} \quad (25.116)$$

The even and odd part of the quadrature filter pair show a phase shift of 90° and can thus also be used to compute the local phase.

Quadrature filters can also be designed on the basis of the monogenic signal. These quadrature filters have one component more than the dimension of the signal. The transfer function is

$$[\hat{h}(\mathbf{k}), i\mathbf{k}\hat{h}(\mathbf{k})/|\mathbf{k}|]^T. \quad (25.117)$$

The best-known quadrature filter pair is the *Gabor filter*. A Gabor filter is a bandpass filter that selects a certain wavelength range around the center wavelength k_0 using the Gauss function. The complex transfer function of the Gabor filter is

$$\hat{g}(\mathbf{k}) = \begin{cases} \exp(|\mathbf{k} - \mathbf{k}_0|^2 \sigma_x^2 / 2) & k k_0 > 0 \\ 0 & \text{otherwise} \end{cases}. \quad (25.118)$$

If $|\mathbf{k}_0| \sigma_x > 3$, (25.118) can be approximated by

$$\hat{g}(\mathbf{k}) \approx \exp\left(-|\mathbf{k} - \mathbf{k}_0|^2 \frac{\sigma_x^2}{2}\right). \quad (25.119)$$

Using the relations in (25.116), the transfer function for the even and odd component are given by

$$\begin{aligned} \hat{g}_\pm(\mathbf{k}) &= \left[\exp\left(-|\mathbf{k} - \mathbf{k}_0|^2 \frac{\sigma_x^2}{2}\right) \right. \\ &\quad \left. \pm \exp\left(-|\mathbf{k} + \mathbf{k}_0|^2 \frac{\sigma_x^2}{2}\right) \right]. \end{aligned} \quad (25.120)$$

The point spread function of these filters is

$$\begin{aligned} g_+(\mathbf{x}) &= \cos(\mathbf{k}_0 \mathbf{x}) \exp\left(-\frac{|\mathbf{x}|^2}{2\sigma_x^2}\right), \\ g_-(\mathbf{x}) &= i \sin(\mathbf{k}_0 \mathbf{x}) \exp\left(-\frac{|\mathbf{x}|^2}{2\sigma_x^2}\right), \end{aligned} \quad (25.121)$$

or combined into a complex filter mask:

$$g(\mathbf{x}) = \exp(i\mathbf{k}_0 \mathbf{x}) \exp\left(-\frac{|\mathbf{x}|^2}{2\sigma_x^2}\right). \quad (25.122)$$

25.1.8 Multiscale Processing

The powerful concept of neighborhood operations is only the starting point for image analysis. This class of operators can only extract local features at scales of at most a few pixels distance. It is obvious that images contain information also at larger scales. To extract object features at these larger scales, we need correspondingly larger filter masks. The use of large masks, however, results in a significant increase in computational costs. If we use a mask of size R^W in a W -dimensional image the number of operations is proportional to R^W . Thus a doubling of the scale leads to a four- and eightfold increase in the number of operations in 2- and 3-D images, respectively.

The explosion in computational cost is only the superficial expression of a problem with deeper roots. The more important question is at which scale can a certain feature in an image be detected in an optimal way? This scale depends, of course, on the characteristic scales contained in the object to be detected. Optimal processing of an image thus requires the representation of an image at different scales.

If an $N \times N$ image is represented on a grid in the spatial domain, we do not have any information at all about the wavenumbers contained at that point in the image. We know the position with an accuracy of the grid constant Δx , but the local wavenumber at this position may be anywhere in the range of the possible wavenumbers from 0 to $N\Delta k = 1/\Delta x$.

In the wavenumber representation, each pixel represents one wavenumber with a wavenumber resolution of $\Delta k = 1/(N\Delta x)$. However, any positional information is lost, as one point in the wavenumber space represents a periodic structure that is spread over the whole image. Thus the positional uncertainty is $N\Delta x$.

This discussion shows that the representation of an image in either the spatial or wavenumber domain constitutes one of two opposite extremes. Either the spatial

or the wavenumber resolution is maximal, but the resolution in the other domain is completely lost. A multiscale image representation requires a type of joint resolution that allows for a separation into different wavenumber ranges (scales) but still preserves as much spatial resolution as possible.

This can be done in the most efficient way in a *multi-grid representation*. The basic idea is simple. While the representation of fine scales requires the full resolution, coarse scales can be represented at lower resolution. This leads to a scale space with smaller and smaller images as the scale parameter increases.

Gaussian Pyramid

If we want to reduce the size of an image, we cannot just *subsample* the image by taking, for example, every second pixel in every second line. This would disregard the *sampling theorem*. For example, a structure that is sampled three times per wavelength in the original image would only be sampled one and a half times in the subsampled image and thus appear as an aliased pattern. Consequently, we must ensure that all structures that are sampled fewer than four times per wavelength are suppressed by an appropriate smoothing filter to ensure a properly subsampled image. This means that size reduction must go hand in hand with appropriate smoothing.

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

$$\hat{B}(\tilde{\mathbf{k}}) = 0 \quad \forall \tilde{k}_p \geq \frac{1}{r_p}, \quad (25.123)$$

where r_p is the subsampling rate in the direction of the p -th coordinate.

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

$$\mathbf{G}^{(0)} = \mathbf{G}, \quad \mathbf{G}^{(q+1)} = \mathcal{B}_{\downarrow 2} \mathbf{G}^{(q)}. \quad (25.124)$$

The number next to the \downarrow in the index denotes the subsampling rate. The 0-th level of the pyramid is the original image.

Repeated smoothing and subsampling operations result in a series of images called the *Gaussian pyramid*. From level to level, the resolution decreases by a factor of two; the size of the images decreases correspondingly. Consequently, we can think of the series of images as being arranged in the form of a pyramid, as illustrated in Fig. 25.13.

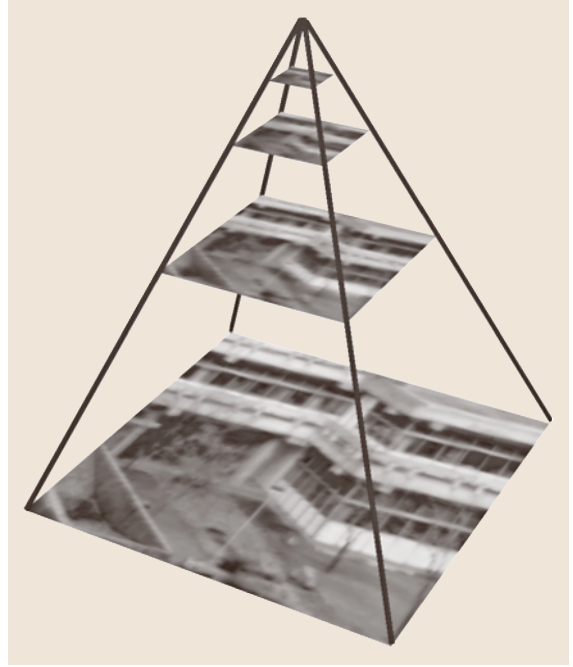


Fig. 25.13 Gaussian pyramid

The pyramid does not require much storage space. Generally, if we consider the formation of a pyramid from a W -dimensional image with a subsampling factor of 2 and M pixels in each coordinate direction, the total number of pixel is given by

$$M^W \left(1 + \frac{1}{2^W} + \frac{1}{2^{2W}} + \dots \right) < M^W \frac{2^W}{2^W - 1}. \quad (25.125)$$

For a two-dimensional image, the whole pyramid needs only one third more space than the original image, and for a three-dimensional image only one seventh more. The computation of the pyramid is equally effective. The *same* smoothing filter is applied to each level of the pyramid. Thus the computation of the *whole* pyramid only needs 4/3 and 8/7 times more operations than for the first level of a two-dimensional and three-dimensional image, respectively.

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

The Gaussian pyramid constitutes a series of low-pass-filtered images in which the cut-off wavenumbers decrease by a factor of two (an octave) from level to level. Thus only the coarser details remain in the smaller images (Fig. 25.13). Only a few levels of the pyramid are necessary to span all possible wavenumbers. For an $N \times N$ image we can compute at most a pyramid with $\text{ld}N + 1$ levels. The smallest image consists of a single pixel.

Laplacian Pyramid

From the Gaussian pyramid, another pyramid type can be derived, the *Laplacian pyramid*, which leads to a sequence of band-pass-filtered images. In contrast to the Fourier transform, the Laplacian pyramid only leads to a coarse wavenumber decomposition without a direc-

tional decomposition. All wavenumbers, independent of their direction, within the range of about an octave (a factor of two) are contained in one level of the pyramid.

Because of the coarse wavenumber resolution, we can preserve good spatial resolution. Each level of the pyramid only contains matching scales, which are sampled a few times (two to six) per wavelength. In this way, the Laplacian pyramid is an efficient data structure that is well adapted to the limits of the product of wavenumber and spatial resolution set by the *uncertainty relation*.

In order to achieve this, we subtract two levels of the Gaussian pyramid. This requires an upsampling of the image at the coarser level. This operation is performed by an *expansion operator* \uparrow_2 . The degree of

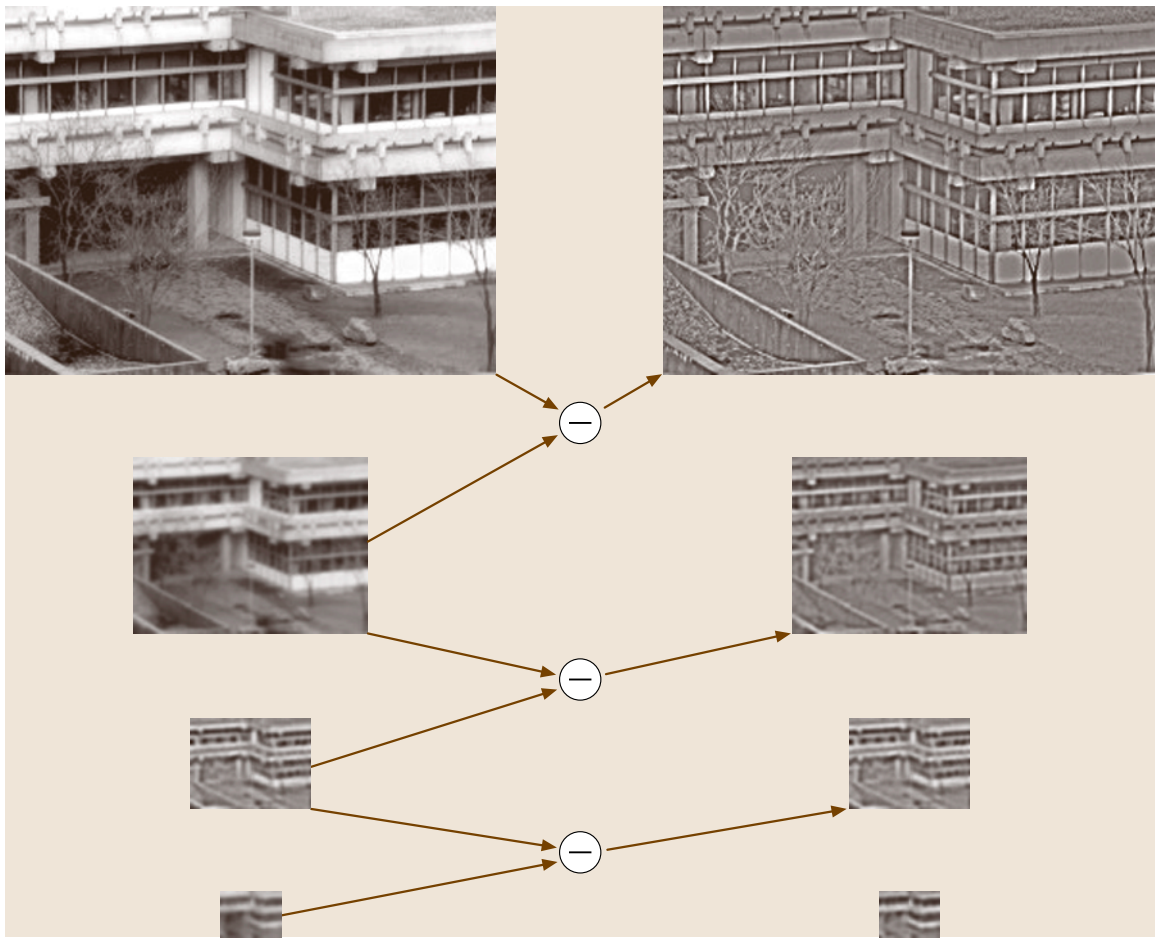


Fig. 25.14 Construction of the Laplacian pyramid (*right column*) from the Gaussian pyramid (*left column*) by subtracting two consecutive planes of the Gaussian pyramid

expansion or upsampling is denoted by the figure after the \uparrow in the index, in a similar notation as for the *reduction operator* (25.124).

The expansion is significantly more difficult than the size reduction as the missing information must be interpolated. For a size increase of two in all directions, first every second pixel in each row must be interpolated and then every second row. Interpolation is discussed in detail in Sect. 25.1.3. Using this notation, the generation of the p -th level of the Laplacian pyramid can be written

$$L^{(p)} = G^{(p)} - \uparrow_2, G^{(p+1)}, \quad L^{(p)} = G^{(p)}. \quad (25.126)$$

The Laplacian pyramid is an effective scheme for a *bandpass decomposition* of an image. The center wavenumber is halved from level to level. The last image of the Laplacian pyramid $L^{(p)}$ is a low-pass-filtered image $G^{(p)}$ containing only the coarsest structures.

The Laplacian pyramid has the significant advantage that the original image can be reconstructed quickly from the sequence of images in the Laplacian pyramid by recursively expanding the images and summing them. The recursion is the inverse of the recursion in (25.126). In a Laplacian pyramid with $p + 1$ levels, the level p

(where the counting starts from zero) is the coarsest level of the Gaussian pyramid. Then the level $p - 1$ of the Gaussian pyramid can be reconstructed by

$$G^{(p)} = L^{(p)}, \quad G^{(p-1)} = L^{(p-1)} + \uparrow_2, G^{(p)}. \quad (25.127)$$

This is just an inversion of the construction scheme for the Laplacian pyramid. This means that, even if the interpolation algorithms required to expand the image contain errors, they affect only the Laplacian pyramid and not the reconstruction of the Gaussian pyramid from the Laplacian pyramid, as the same algorithm is used. The recursion in (25.127) is repeated with lower levels until level 0, i. e., the original image, is reached again. As illustrated in Fig. 25.14, finer and finer details become visible during the reconstruction process. Because of the progressive reconstruction of details, the Laplacian pyramid has also been used as a compact scheme for image compression. Nowadays, more-efficient schemes are available on the basis of wavelet transforms, but they operate on principles that are very similar to those of the Laplacian pyramid [25.14, 15].

25.2 Motion Analysis

25.2.1 General Considerations on Motion Analysis

Image-based whole-field velocimetry methods are used to measure the flow field of a fluid, based on the analysis of an image sequence visualizing the flow under consideration. A large number of different methods have been developed and successfully applied during the past two decades, with both camera/computer-hardware and image processing algorithms being constantly improved upon in terms of accuracy, spatial and temporal resolution, and dynamic range. In general, all these methods estimate the flow velocity by determining the displacements of some kind of image features in a number of successive frames (at least two). In a computer vision context, this displacement field is called the *optical flow* $f(x, t)$.

Computing optical flow based on motion analysis in general is one of the major issues of computer vision, with applications not restricted to fluid flow but including any kind of dynamic processes and scenes. In addition, photogrammetrists use matching methods that are closely related to motion analysis, e.g., to establish

correspondences between two stereo images to compute a disparity map or to locate target patterns within an image. Accordingly, there is a huge amount of methods, algorithms, and publications spread out through the computer vision, photogrammetry and fluid mechanics literature. The vast terminology for the different methods might confuse the unfamiliar reader, particularly because notions like optical flow, image matching, image correlation, and tracking are not always used consistently by the different communities (or even within the same community).

The methods differ in the following aspects:

- What kind of image features are used?
 - A1 Single particles, i. e., discrete features,
 - A2 particle patterns, i. e., patterns of discrete features,
 - A3 continuous features.
- What kind of input data is used for the velocity estimation?
 - B1 Spatial information, i. e., positions of features in the image plane,

Table 25.4 Examples of different approaches to image-based velocity analysis

Method	Reference	Features	Data	Calculation
Standard PIV	Willert and Gharib [25.16], Westerweel [25.17]	A2	B1,B3	C1
Correlation-based tracking, correlation imaging velocimetry	Fincham and Spedding [25.18]	A2	B1,B3	C1
Image correlation velocimetry, adaptive least-squares matching	Tokumaru and Dimotakis [25.19], Gruen [25.20]	A2,A3	B1,B3	C2
Multi-grid PIV with deformable windows	Scarano and Riethmuller [25.21]	A2	B1,B3	C1
Hybrid PIV/PTV	Cowen and Monismith [25.22], Bastiaans et al. [25.23]	A1,A2	B1,B3	C1,C3
Two-frame tracking	Baek and Lee [25.24], Ohmi and Li [25.25]	A1	B1,B3	C3
Four-frame tracking	Hassan and Canaan [25.26], Malik et al. [25.27]	A1	B1,B2,B3	C3
Kalman filtering	Takehara et al. [25.28]	A1	B1,B2,B3	C3
Optical flow techniques	Jähne et al.[25.29]	A2,A3	B1,B2,B3,B4	C2

- B2 temporal information, i.e., more than two frames are used,
- B3 intensity, i.e., gray values of features,
- B4 intensity gradients, i.e., local gray value differences.

- What is the computational approach to solve the motion correspondence problem?

- C1 Cross-correlation,
- C2 least-squares optimization (linear or nonlinear),
- C3 discrete tracking techniques (kinematic models, combinatorial optimization, Kalman filters).

Algorithms using almost any combination of image features, input data, and calculation method can be found in the literature. Some examples are compiled in Table 25.4. A common classification is to divide the methods into two major groups: *region-based methods* and *feature-based methods*.

Region-based methods estimate the motion of gray value patterns within small image patches, so-called *interrogation areas* or *interrogation windows*. The most common region-based method used in fluid-mechanic applications is *particle image velocimetry (PIV)* (Sect. 25.2.2), whereas in computer vision and photogrammetry, *optical flow techniques* (Sect. 25.2.5) and *least-squares matching* (Sect. 25.2.3) are frequently used. Since in all these methods, the image is divided into a regularly spaced array of interrogation windows, the result is a displacement field on a regular grid (which may be the pixel grid itself, as in optical flow methods of computer vision).

In contrast, feature-based methods try to identify single objects in the image, segment them from the background, and follow their motion throughout an image sequence. Thus, feature-based methods yield randomly spaced velocity information, depending on the distribution of objects in the image. The most important feature-based method for flow visualization is *particle-tracking velocimetry (PTV)* (Sect. 25.2.4), where individual tracer particle images are the objects to be tracked.

In general, both classes of methods have advantages and disadvantages, which are outlined in the following sections. Note that there are also hybrid methods that try to combine the advantages of region- and feature-based approaches, and consequently have better performance in many cases.

To summarize, four important groups of methods can be distinguished: correlation-based analysis (Sect. 25.2.2), least-squares matching (Sect. 25.2.3), tracking techniques (Sect. 25.2.4), and optical flow methods (Sect. 25.2.5). In the remainder of this section, some general aspects of motion analysis, which apply equally to all the different approaches, are outlined.

Dynamic Range, Sampling Theorem, and Subpixel Accuracy

Dynamic Range. An important quantity characterizing the potential of a motion estimator is its *dynamic range DR*, which is defined as the ratio of the maximum to the minimum displacement that can be measured:

$$DR = \frac{\xi_{\max}}{\xi_{\min}}. \quad (25.128)$$

Obviously, a dynamic range as high as possible is desirable, in particular with regard to the measurement of turbulent flows, which may contain strong velocity fluctuations.

The fundamental limits on the dynamic range of a digital imaging method are related to the discrete nature of the image data. The measurement of large displacements is limited by the *temporal sampling theorem* [25.29], while the measurement of small displacements is limited by the maximum *subpixel accuracy* that can be achieved, which in turn depends on the sampling and quantization of the image intensity [25.29]. Approaches to overcome the limitations of large displacements and increase the dynamic range are outlined below. If such an approach is used together with a subpixel-accurate determination of small displacements (see below), a dynamic range of 100–1000 can be achieved using standard equipment.

Sampling Theorem and Motion Correspondence. To estimate an object's velocity given two successive image frames, the *motion correspondence problem* has to be solved, i.e., a unique correspondence between two images of the same object in the two successive frames has to be established. This can only be achieved, if the *temporal sampling theorem* is valid. In simple words, the (temporal) sampling theorem (or *Nyquist criterion*) states, that the motion between two images, i.e., the optical flow \mathbf{f} , should be less than half the smallest local spatial scale $\lambda_{g,\min}$ of the image intensity $g(\mathbf{x}, t)$:

$$|\mathbf{f}| \Delta t = |\mathbf{f}| \cdot (1 \text{ frame}) < \frac{1}{2} \lambda_{g,\min}, \quad (25.129)$$

where Δt is the time interval between two successive images, in units of frames (thus, $\Delta t = 1$), and \mathbf{f} is the optical flow in units of pixel/frame. The sampling theorem puts a fundamental limit on the relation between the size and intensity structure of an object and its motion, i.e., on the relation between the spatial and temporal intensity gradients. Given just two images, the motion of an object can only be recovered unambiguously if (25.129) is valid. In this case, the motion correspondence problem can be solved. Otherwise, temporal aliasing occurs [25.29], and the problem of motion estimation becomes ill-posed, i.e., there is no unique solution.

As a consequence of the sampling theorem, there is a maximum allowed displacement that can be recovered by any region-based method. For example, consider a differential optical flow technique (Sect. 25.2.5). In this case, there has to be a unique relation between the spatial and temporal gray value gradients. To estimate the

motion of a single particle with a symmetrical Gaussian intensity distribution (as typical for PIV and PTV particle images), the maximum allowed displacement corresponds roughly to the standard deviation of the Gaussian radius of the particle. As a second example, for a quadratic PIV interrogation window of length L , the maximum displacement corresponds roughly to $L/4$, assuming that the intensity distribution or particle density within the window is homogeneous and sufficiently large. The latter result is known as the one-quarter rule in the PIV literature [25.30]. Obviously, the smallest spatial scale within a PIV interrogation window depends on the particle distribution and density within that window. Generally, an optimal density of about 10 particles per interrogation window is recommended in the PIV literature. Assuming a homogeneous distribution, the particles form a periodic intensity pattern of wavelength $\lambda = L/2$. Hence, the one-quarter rule follows from the sampling theorem. Note that these limits are not strict but should be considered as more or less accurate estimates, since the actual spatial frequency content of the image depends on the particle distribution. The latter is a stochastic quantity, with varying values for different interrogation windows within one image.

The situation is a bit different for feature-based tracking methods. As a simple example, if only one object is visible in the image, its motion can be tracked with the only restriction that it stays within the field of view, since in any case two successive images of the object can be related to each other unambiguously. In this case, the wavelength of the spatial image structure corresponds to the size of the image (or twice this size). However, such a case is of limited practical importance, since there will always be more than one object to be tracked. With increasing object or particle density, the spatial image scales become smaller and the motion correspondence becomes more difficult, which again is a manifestation of the sampling theorem. Still, tracking algorithms are able to track motions violating the sampling theorem. However, the latter is only possible, if further information is used (apart from two successive images). For example, a common assumption is that object trajectories are smooth, i.e., the direction and speed of an object does not change abruptly between two frames. In this case, it is possible to use information from previous frames as input to a motion model and predict the position of the object in the next frame by extrapolation. If the model provides a good description of the actual motion, much larger displacements can be handled compared to approaches using only two frames without any modeling.

Subpixel Accuracy. The subpixel accuracy of a velocity estimator determines the minimum displacement that can be measured. Since a digital image provides a sampled version of the original intensity distribution of the physical image, with gray values defined on an integer grid (pixel positions), the position of an object, e.g., a particle image or the correlation peak resulting from the cross-correlation of two [PIV](#) images, can only be determined with an accuracy of ± 0.5 pixel. To achieve higher accuracy, some kind of subpixel interpolation has to be carried out. One way to do this is to use a model of the intensity distribution of the object and determine the best fit of this model to the image data in a least squares sense. The most common model in [PIV](#) and [PTV](#) is a Gaussian distribution, since it provides a very good approximation to both the image of a single tracer particle and the displacement peak in a [PIV](#) correlation. The subpixel-accurate coordinates are introduced as parameters of the model and determined using a least-squares algorithm.

Another common approach to achieve subpixel accuracy in [PIV](#) and optical flow techniques is to warp the original images according to an estimated flow model. The warping is carried out iteratively, and a refined technique of the velocity field is computed in each iteration. Since the warped image will generally be defined on noninteger pixel positions, warping requires a precise method to interpolate gray values; see, e.g., Sect. 25.1.3.

In applying subpixel interpolation, one should keep the following considerations in mind. To compute a subpixel-accurate position within an image, the information contained in the image intensity, i. e., in the gray values, is translated into geometric information, i. e., position in the image. This translation is based on certain assumptions concerning radiometric aspects of the imaging process. One such assumption is the Gaussian intensity distribution mentioned above. Further important assumptions, which are often taken for granted, are the linearity and homogeneity of the sensor and a homogeneous illumination. If any of these assumptions is violated, subpixel accuracy will deteriorate or even become meaningless. Thus, it is very important to take into account the radiometric properties of the cameras and illumination, if very high accuracy is required. For example, if the cameras suffer from strong fixed pattern noise, a radiometric correction should be applied to the images. Even if the image data is perfect and all the assumptions are valid, the result of the subpixel interpolation may still be biased. For example, one source of bias in [PIV](#) evaluation is the so-called peak-locking effect.

As a general limit, for typical 8 bit images with 256 gray levels, one can expect a (theoretical) maximum subpixel accuracy of the order of magnitude of 0.01 pixel, given optimal image data, a good object model, and an unbiased estimator. Note that it may be very difficult to actually achieve such ideal circumstances in real [PIV](#) or [PTV](#) applications, where measurement errors are typically in the range of 0.05–0.2 pixel.

Hierarchical Multigrid Approaches

As explained in the previous section, the maximum displacement that can be determined by region-based approaches such as [PIV](#) and differential optical flow methods is limited by the smallest spatial scales of the underlying image structure. However, images also contain information at larger scales than the neighborhood size of the interrogation windows. The basic idea of iterative, hierarchical methods is to start the estimation of the optical flow at the largest image scales, which enable the determination of large displacements in a first iteration. This first estimation may be applied to warp the second image back along the estimated displacement field and refine the estimation at smaller spatial scales. An efficient implementation of such a coarse-to-fine strategy is a *Gaussian image pyramid* (Sect. 25.1.8), which is basically a multigrid representation of an image at different spatial scales. The efficiency of Gaussian pyramids is due to the reduction of the linear image size by a factor of two at each level of the pyramid. This reduction makes the large-scale information in the image available to small filter masks. However, at the same time the image becomes increasingly blurred. Hence, we have to take care in applying Gaussian pyramids to [PIV](#) images, since the small particle images may soon be completely smoothed out. Large-scale information can only be obtained if there is a certain fraction of larger particles in the image or the particle density varies locally. Hierarchical [PIV](#) approaches are often realized by starting with large interrogation windows and iteratively decreasing the size of the interrogation windows instead of decreasing the image size as in a Gaussian pyramid.

Modeling of Displacement Fields

Given two successive images $g_0 = g(\mathbf{x}, t_0)$ and $g_1 = g(\mathbf{x}, t_1)$ of a flow field, the displacement field $\xi(\mathbf{x}, t)$ can be thought of as the transformation, or mapping, of the spatial image intensity field from the first image to the second. The optical flow is the time derivative of this mapping: $\mathbf{f}(\mathbf{x}, t) = \partial_t \xi(\mathbf{x}, t)$. Within a local neighborhood N centered at \mathbf{x}_0 , the displacement field may be

approximated by a Taylor expansion

$$\xi(\mathbf{x}, t) = \xi(\mathbf{x}_0, t) + (\mathbf{x} - \mathbf{x}_0) \nabla \xi(\mathbf{x}_0, t) + \frac{1}{2!} [(\mathbf{x} - \mathbf{x}_0) \nabla]^2 \xi(\mathbf{x}_0, t) + \dots$$

Taking into account only the first-order terms, the equivalent formulation for the optical flow reads

$$\mathbf{f}(\mathbf{x}, t) = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix} + \begin{pmatrix} a_5 \\ a_6 \end{pmatrix} \quad (25.130)$$

$$= \mathbf{A}(\mathbf{x} - \mathbf{x}_0) + \mathbf{t}. \quad (25.131)$$

In this first-order approximation, the displacement field consists of a constant shift \mathbf{t} and a linear (affine) deformation of the local neighborhood, described by the matrix \mathbf{A} . Note that in such a formulation, the spatial derivatives of the flow field are introduced as parameters:

$$\begin{aligned} a_1 &= \frac{\partial f_x}{\partial x}, & a_2 &= \frac{\partial f_x}{\partial y}, \\ a_3 &= \frac{\partial f_y}{\partial x}, & a_4 &= \frac{\partial f_y}{\partial y}. \end{aligned} \quad (25.132)$$

This offers the possibility to estimate spatial velocity gradients without performing explicit differentiation of the velocity field. Thus, important hydromechanic quantities like the in-plane vorticity $\omega_z = \partial_x f_y - \partial_y f_x$ and the rate-of-strain tensor \mathbf{S} can be directly estimated, since

$$\mathbf{A} = \begin{pmatrix} 0 & -\frac{\omega_z}{2} \\ \frac{\omega_z}{2} & 0 \end{pmatrix} + \mathbf{S}, \quad (25.133)$$

with

$$\mathbf{S} = \begin{pmatrix} \partial_x f_x & \frac{1}{2}(\partial_y f_x + \partial_x f_y) \\ \frac{1}{2}(\partial_y f_x + \partial_x f_y) & \partial_y f_y \end{pmatrix}. \quad (25.134)$$

Similar to this spatial modeling of the displacement fields, the temporal evolution of the motion of a single particle along its Lagrangian trajectory around the point \mathbf{x}_0 may be approximated using a Taylor expansion in time:

$$\xi(\mathbf{x}_0, t) = \xi(\mathbf{x}_0, t_0) + \mathbf{v}(t - t_0) + \frac{1}{2} \mathbf{a}(t - t_0)^2 + \dots \quad (25.135)$$

This kind of modeling is frequently applied in particle-tracking algorithms, see Sect. 25.2.4.

A more-detailed discussion of the modeling of flow fields is given by Jähne et al. [25.29].

Confidence Measures, Validation and Postprocessing

As any measurement technique, the result of a velocity estimation should not only supply the velocity field, but also a *measure of confidence*. To enable a reliable interpretation of the velocity field, gross errors have to be detected and removed. The optical flow methods discussed in Sect. 25.2.5 yield confidence measures as an integral part of the result. In PIV, typically the ratio of the tallest to the second tallest correlation peak is used to detect unreliable measurements. Based on such confidence measures, questionable measurements are detected and removed from the velocity field, which is typically done in a postprocessing step after the velocity field has been computed. However, in iterative methods, where the results strongly depend on the quality of the velocity estimates in previous iterations, the validation should be done after each iteration.

After the erroneous vectors have been removed, the resulting gaps in the velocity field may be filled by applying an interpolation technique, e.g., *adaptive Gaussian windowing* (AGW) [25.31]. Such interpolation techniques can also be used to interpolate the randomly distributed velocity vectors resulting from a PTV technique to a regular grid. Basically, the interpolation corresponds to a convolution of the velocity field using a special convolution kernel, e.g., a Gaussian in the AGW. To account for the varying uncertainty of the computed velocity vectors, a *normalized convolution* may be computed, where pixels with suspicious information (as indicated by their confidence measure) are given a low weighting factor in the convolution sum. For further information on interpolation and convolution techniques, refer to [25.29].

3-D Motion Estimation

Most of the methods discussed in this chapter refer to the case of 2-D motion estimation within a plane, i. e., the image plane. However, all of these methods can easily be extended to the case of 3-D motion estimation within a volume in space. From an algorithmic point of view, there is no principal difference between, e.g., computing a cross-correlation in 2-D and in 3-D. Optical flow algorithms and tracking methods can also be applied to the 3-D case simply by adding a further dimension. The challenge in 3-D motion estimation is rather a technological one: the acquisition of 3-D image data. Most approaches to 3-D velocity measurement are based on *stereoscopic* or *multi-view imaging* using two or more views of the same flow scene to recover the 3-D velocity field. The most prominent method applied to flow meas-

urement is stereoscopic PIV [25.32]. Some 3-D PTV approaches are discussed in Sect. 25.2.4.

The basic new ingredient of 3-D methods as compared to 2-D methods is a *geometric camera calibration*. This calibration is necessary because perspective effects have to be taken into account in the evaluation of stereo images. The task of the stereo evaluation is to establish stereoscopic correspondences between two different views of the same scene. Thus, in addition to the motion correspondence problem (temporal correspondence), the *stereo correspondence* problem (spatial correspondence) has to be solved: given two views of the same scene, e.g., a flow field visualized by tracer particles, a unique correspondence between the particle images in the two views has to be found. The camera calibration provides the geometric relationship between the two views, the so-called *epipolar geometry*. If this relationship is known, the stereo correspondence problem can be solved much easier and faster. Further, the calibration also provides the necessary geometric information to compute the 3-D position of an object by triangulation of two or more views.

Stereo algorithms can be implemented very efficiently and transparently in terms of *projective geometry*. The (projective) geometry of multiple views and its implications for motion analysis are extensively discussed in the computer vision and photogrammetry literature [25.33]. Camera calibration is a classic topic of photogrammetry [25.34].

Another powerful but experimentally very elaborate approach is holographic imaging [25.35]. More information on this and other 3-D flow visualization methods can be found in [25.36].

25.2.2 Correlation-Based Velocity Analysis

In this section, approaches to velocity analysis based on the computation of cross-correlation coefficients are discussed. These approaches belong to the region-based methods. The focus is on particle image velocimetry (PIV), which is the method that is most often applied in fluid mechanics applications.

Standard Digital PIV Analysis

Basic Principle. Particle image velocimetry (PIV) is a technique to determine the two-component displacement vectors of tracer particle patterns in a 2-D plane (light sheet) within a flow. The result is a snapshot of the Eulerian flow field. The displacements are found by dividing two subsequent frames of a PIV sequence,

$g_1 = g(\mathbf{x}, t_1)$ and $g_2 = g(\mathbf{x}, t_2)$ into interrogation windows, typically of a size of 16×16 or 32×32 pixels, and computing the cross correlation coefficient $r(\mathbf{x}, \mathbf{s})$ of two corresponding windows

$$r(\mathbf{x}, \mathbf{s}) = \frac{\langle g_1(\mathbf{x}') g_2(\mathbf{x}' - \mathbf{s}) \rangle}{\langle g_1^2(\mathbf{x}') \rangle \cdot \langle g_2^2(\mathbf{x}') \rangle}, \quad (25.136)$$

using the abbreviation

$$\langle a(\mathbf{x}') \rangle = \int_{-\infty}^{\infty} w(\mathbf{x} - \mathbf{x}') a(\mathbf{x}') d^2 \mathbf{x}',$$

where the weight function $w(\mathbf{x} - \mathbf{x}')$ represents the size of the interrogation window and it is assumed that the local mean values over the interrogation windows have been subtracted from g_1 and g_2 . The correlation coefficient is computed for a given 2-D range of displacements \mathbf{s} of the interrogation window, resulting in a so-called *correlation plane*.

Computation of Velocity. Because the direct evaluation of the cross correlation coefficient (25.136) is computationally very expensive, it is usually computed using fast Fourier transform (FFT) methods, because in Fourier space the double summation is replaced by a simple pointwise multiplication. Once the correlation plane has been determined, the correct displacement is given by the maximum correlation peak. Thus, the optical flow is approximated as

$$\mathbf{f}(\mathbf{x}, t) \approx \frac{1}{\Delta t} \arg \max r(\mathbf{x}, \mathbf{s}), \quad (25.137)$$

where Δt is the time difference between the two successive images. Subpixel accuracy is achieved by centroiding or fitting a Gaussian to the correlation peak. Usually, in both methods only three neighboring correlation values in each direction are used (three-point estimators). Depending on the image quality and the evaluation method, the accuracy of the displacement estimation is of the order of 0.01–0.1 pixels and the dynamic range of the method is of the order of 100–1000.

Velocity Postprocessing. To compute an instantaneous velocity field, interrogation windows are distributed on a regular grid and evaluated by cross-correlation. The result is the instantaneous Eulerian velocity field. Since PIV is a statistical evaluation method, this vector field will contain a certain amount of spurious vectors (outliers), which result from interrogation windows containing an insufficient number of particle images.

These outliers have to be removed prior to any further evaluation of the velocity field. Some methods for outlier removal and interpolation of the resulting gaps are mentioned in Sect. 25.2.1. After this postprocessing, higher-order quantities such as vorticity, divergence, or rate of strain may be computed.

Limitations. The basic approach to PIV as discussed in this section suffers from a number of shortcomings that limit the accuracy, dynamic range, and spatial resolution of PIV. The main origin of these shortcomings is the fixed, finite size or shape of the interrogation window used in the correlation analysis, which effectively acts as a spatial low-pass filter on the estimated velocity field. Another source of error stems from the spatial discretization of digital particle images. In detail, the following limitations exist:

- *In-plane loss of pairs.* Particles may enter or leave the finite interrogation window between subsequent frames, in particular those that are moving faster than the mean velocity within the window. Thus, fast particles will not contribute to the correlation peak, since they do not have a matching partner within the interrogation window. This results in a bias of the estimated velocity towards lower values.
- *Velocity gradients.* Spatial velocity gradients within the interrogation window also contribute to the in-plane loss of pairs and thereby reduce the signal-to-noise ratio in the correlation plane since not all particles within an interrogation window correlate equally well due to their nonuniform motion. As a rule of thumb, the degradation of the PIV result becomes significant if the displacement of tracer particles due to local flow gradients gets larger than the image diameter of the particles.
- *Out-of-plane loss of pairs.* Particles may enter or leave the light sheet along the optical axis during the time of two successive exposures. Such particles are only visible in one of the images and do not have a matching partner. Again, this results in a reduction of the signal-to-noise ratio. The out-of-plane loss of pairs is a principal physical limitation of PIV that can only be overcome by adjusting the experimental parameters, e.g., the thickness of the light sheet or the frame rate of the cameras.
- *Computational aspects.* To reduce the computational load, the correlation is often computed in the Fourier domain using FFT methods. However, the necessary assumption of the periodicity of the image data within the interrogation window introduces inaccuracies

compared to a direct spatial cross-correlation, which is, in principal, more accurate [25.37].

- *Peak-locking or pixel-locking.* The discrete nature of the PIV images introduces a bias towards integer displacements in the subpixel evaluation of the displacement peak. Peak-locking is the result of a biased subpixel estimation, if the input data (i.e., correlation values) is distributed asymmetrically around the maximum peak. The degree of peak-locking depends on the size of the particle images.

Apart from peak-locking, all these limitations are a consequence of the finite extent of the interrogation windows. The size of the window is given by a trade-off between dynamic range and accuracy on one hand and spatial resolution on the other hand. Large interrogation windows can resolve large motions and provide good accuracy due to a high signal-to-noise ratio, given that the window contains only weak velocity gradients. Large windows are also more robust to outliers. On the other hand, smaller windows provide a better spatial resolution and are less affected by velocity gradients, e.g., shear flows or strong vortices. However, to enable a reliable evaluation of the cross-correlation, the windows must contain a sufficient number of particle images and thus must have a certain minimum size, which depends on the particle density.

The limitations imposed by fixed interrogation windows can also be explained by looking at the spatial Taylor expansion of the velocity field (25.130). The standard PIV approach can only compute a straight shift of the interrogation windows between two frames. The velocity field within the interrogation window is assumed to be constant. This corresponds to a zeroth-order expansion of the velocity field. Linear effects like rotation, shear and dilation, or higher-order deformations are not accounted for. Due to the spatial averaging over the interrogation window, flow scales smaller than the window size cannot be recovered.

To summarize, the accuracy, spatial resolution, and dynamic range of the standard PIV method are coupled by the size of the interrogation window. The performance of PIV depends on three main factors: particle size and density, the size of the interrogation window, and the presence of velocity gradients. Particle size and density can be controlled during the setup of the experiment and are not discussed further. Recommendations for optimal settings are given in the PIV literature [25.30]. In the following sections, some advanced PIV methods are discussed. The goal of these

methods is to overcome the limitations of the standard **PIV** approach to increase the accuracy, resolution, and dynamic range. Towards this end, the latter three performance measures have to be decoupled. Most of the advanced methods rely on the following three major ideas: iterative methods instead of a single-pass evaluation to refine the solution, hierarchical multigrid approaches to resolve both large and small motions, and higher-order approximations of the velocity field to account for velocity gradients and higher-order deformations.

Advanced Digital PIV Analysis

Multiple-Pass Interrogation with Window Shifting. To reduce the in-plane loss of pairs, a discrete integer window offset determined in a first interrogation pass is introduced before doing a second interrogation using the shifted window. The increased number of matched particle pairs results in an increased signal-to-noise ratio of the correlation peak. Iterations of the window shifting may be carried out until the displacement determined in the final iteration is below one pixel. Due to the discrete window shifting, the result still suffers from peak-locking, which can be reduced by applying continuous window shifting.

Correlation-Based Tracking. In the standard **PIV** approach, the interrogation windows in the first and second frame are of the same size and at the same location within the image. This is the major reason for the low-velocity bias error due to the in-plane loss of pairs. A simple modification to eliminate this error is to use a larger interrogation window in the second frame, centered around the smaller window in the first frame [25.18]. In this case, the correlation coefficient has to be computed directly in the spatial domain for all displacements of the small window within the large window. Fincham et al.[25.18] have termed this approach *correlation image velocimetry* to distinguish it from the standard correlation-based interrogation using equally sized windows. In a number of more-recent references, the approach using differently sized windows is referred to as *correlation-based tracking*, since the particle pattern defined by the small interrogation window is tracked within a search area defined by the large interrogation window.

Multiple Passes with Decreasing Window Size. The optimal interrogation window size for **PIV** depends on the local flow conditions and the local seeding particle density, which means that it is rarely constant from

one region of the flow to another. Thus, instead of using fixed window sizes, the size of the window should be dynamically adapted to the local flow conditions. A simple way to implement this idea is to refine the correlation interrogation in an iterative way by starting with large windows and decreasing the window size during the course of the iterations [25.38]. In such a multigrid approach, the maximum in-plane displacement is decoupled from the interrogation window size, which increases the dynamic range without decreasing the spatial resolution. The displacements computed with larger interrogation windows can be used as predictions for further interrogations with smaller windows to shift the windows according to the prediction before the next interrogation is calculated. Thus, a high signal-to-noise ratio can be maintained also with small interrogation windows. The size of the interrogation windows may be decreased down to a correlation of single particles.

Since in such iterative methods, the quality of the final result depends on the results of previous iterations, in particular the first iteration, validation methods (Sect. 25.2.1) should be applied after each iteration. Since the first iteration will generally be a standard **PIV** correlation and as such suffer from all the basic limitations mentioned above, more-sophisticated methods have been developed for the first iteration [25.39].

Deformable Windows and Higher-Order Approximations of the Displacement Field. In the standard **PIV** evaluation, interrogation windows with fixed size and shape are used, and the velocity field is assumed to be constant within the windows, which is a zeroth-order approximation of the velocity field (Sect. 25.2.1). To account for variations of the velocity within the windows due to velocity gradients and higher-order effects, deformations of the interrogation windows of the particle images have to be considered, corresponding to a higher-order approximation of the velocity field. Towards this end, [25.40, 41] introduced the *particle image distortion* technique: they use fixed interrogation windows, but apply an iterative deformation of the images to compensate for in-plane loss of pairs. In each iteration, the image area within the interrogation window is deformed according to the displacement field calculated in the previous iteration. To compute the deformed images, some kind of image interpolation has to be applied, e.g., bilinear interpolation or spline interpolation (Sect. 25.1.3). Care has to be taken in the interpolation step not to spoil the accuracy gain due to the window deformation with an inaccurate interpolation scheme. A further advantage

of the window deformation using image interpolation is the possibility to introduce continuous window offsets, which reduces the peak-locking effect.

Second-Order Correlation. As an effective method to suppress false correlation peaks and amplify the correct one, [25.42] introduced the second-order correlation method, which is simply a multiplication of the correlation plane of an interrogation area by the correlation plane of one or more neighboring interrogation areas (overlapping by, e.g., 50%). Thus, it is a correlation of the correlation. Since any peak that does not appear in both planes is eliminated, correlation anomalies are suppressed, resulting in more-reliable and accurate velocity estimates. Unlike statistical PIV postprocessing methods to remove spurious vectors, which rely on the accuracy and similarity of neighboring vectors, errors are directly eliminated in the correlation data. The second-order correlation may be applied together with any of the PIV methods discussed in this section to validate the results already during the computation step.

Super-Resolution PIV, Hybrid PIV/PTV Methods. Clearly, the maximum amount of information contained in a PIV image is the motion of the individual particles. The number of particles within an image, i. e., the particle density, defines the maximum spatial resolution of the velocity field that can be achieved. The approach of particle-tracking velocimetry (Sect. 25.2.4) is to actually exploit the maximum resolution by identifying the individual particles and measuring their motion. However, such an approach is not feasible in the evaluation of PIV images, since the particle density is much higher than in PTV images, which gives rise to ambiguities in the temporal correspondence analysis of the particle motion that cannot be resolved without further information. The idea of super-resolution PIV respectively hybrid PIV/PTV is to combine PIV and PTV. The goal is to increase the spatial resolution and accuracy of PIV and overcome the averaging and gradient-biasing effects of the standard PIV interrogation by tracking the individual particles within the interrogation windows [25.22]. The initial result of a coarse PIV interrogation is used in a predictor step to direct the particle-matching algorithm in the right direction and thereby reduce the size of the search area. With a smaller search area located near the correct match partner, the probability of ambiguities is reduced. If a unique match is established, the particle velocity is calculated from the two positions in the successive images by a finite-difference scheme.

3-D/3-C PIV

Several methods have been proposed to extend the PIV technique towards measurements of full three-component (3-C) vectors respective measurements within a three-dimensional (3-D) volume in space. Stereoscopic PIV is the method that is applied most frequently.

Stereoscopic PIV. Stereoscopic PIV enables the measurement of 3-C vectors within a plane in space. Hence, it is a 2-D/3-C method. For a review, see [25.32]. The basic idea is to use two cameras observing the light sheet, and to compute the third velocity component (i. e., the out-of-plane motion) from the disparity map between the two particle images. Further, stereoscopic PIV also offers the possibility to eliminate perspective errors, which may contaminate the in-plane measurements if perspective effects are strong, i. e., when the lateral dimensions of the object plane are comparable to its depth.

Stereoscopic PIV systems can be arranged in two configurations. *Translational systems* have parallel optical axes, whereas in *rotational systems* the two optical axes are arranged enclosing a convergence angle α . Both arrangements have advantages and disadvantages [25.32].

3-C vectors are obtained by mapping the displacements from each image plane to the object plane and combining them to obtain the third component. There are three different approaches [25.32]:

1. **Geometric reconstruction.** A priori knowledge of the complete recording geometry is necessary. This information is used to perform an explicit ray tracing of the projection rays. This method is tedious and not very accurate, since the necessary geometric parameters (e.g., stereo baseline, depth of the measurement plane) often cannot be measured with sufficient accuracy.
2. **2-D calibration.** A calibration is performed using one image of a calibration target, which has to coincide exactly with the plane of the light sheet during flow measurements. A general polynomial transformation (typically up to second or third order to account for lens distortions) between the object plane and the image planes of the two cameras is estimated, based on the known correspondences between object and image points of the calibration target. The final step of determining the 3-C velocity uses reconstruction equations that still require some knowledge of the geometry such as the separation between the

lenses, the object distance, or the angular orientation of the cameras to the object plane.

3. **3-D calibration.** A full 3-D geometric camera calibration is performed, using several images of translated calibration planes. To compute 3-C vectors, explicit knowledge of the system geometry is not required. General higher-order polynomial transformations are also frequently applied in the 3-D calibration. Instabilities related to overparameterization might be introduced if the measurements are noisy, since typically ≈ 40 free parameters are calibrated for each camera. The application of photogrammetric pinhole-camera models and self-calibration methods in stereoscopic PIV is a rather recent development [25.43].

Defocusing PIV. [25.44] introduced defocusing PIV as a method to obtain 3-D/3-C velocity fields. A volume illumination is applied, and the defocus principle is used to identify three-dimensional particle locations. *Pereirra et al.* [25.45] use a similar technique to obtain full 3-D information. A volumetric cross-correlation is computed to estimate the velocity field.

Multiplane Stereoscopic PIV. The idea of multiplane stereoscopic PIV is to use several light sheets in different depths to obtain flow information from a number of different planes within a 3-D volume. The planes may be illuminated either simultaneously or sequentially. In the former case, several stereo camera setups are used to acquire the images. For details, see [25.46]. A recent variant of multiplane stereoscopic PIV is the XPIV method [25.47]. It combines stereoscopic PIV, multisheet illumination and defocusing PIV. The latter is applied to separate the different depth planes which are all projected simultaneously into the same camera.

Photogrammetric PIV. *Pereirra et al.* [25.48] describe a photogrammetric PIV system. The principle is similar to that of a 3-D particle-tracking velocimetry setup (Sect. 25.2.4). Three cameras are used to acquire images of the flow. The 3-D particle positions are reconstructed by triangulation, based on a geometric camera calibration that is performed prior to the flow measurements. The 3-D/3-C flow field is obtained by computing the volumetric cross-correlation of the particle positions in subsequent frames. The only difference between 3-D particle-tracking and photogrammetric PIV is that the former tracks single particles in 3-D, whereas the latter computes the cross-correlation of 3-D interrogation areas, i. e., volumetric, spatial particle patterns.

Holographic PIV. In contrast to all the methods discussed so far, holographic PIV [25.35] requires a volumetric illumination with *coherent* light to record holograms of the flow. The 3-D/3-C flow field is recovered by interrogating the holograms with coherent light beams. In principle, holographic PIV is superior to all the other methods, but the experimental setup and the data evaluation is very complex. For these reasons, holographic PIV does currently not provide the ability to collect large data bases for statistical analyses. Hence, the application of holographic PIV is limited to relatively simple flow configurations.

Further Reading

Different PIV methods are reviewed by *Adrian* [25.49]. Information on autocorrelation PIV including film-based acquisition and optical evaluation methods can be found in *Kean and Adrian* [25.50, 51]. The theory of cross-correlation PIV is developed in [25.52]. The fundamentals of digital PIV are discussed by *Westerweel* [25.17], *Willert and Gharib* [25.16] and in the books by *Westerweel* [25.53] and *Raffel et al.* [25.30]. The latter gives a large number of references to further information on PIV.

25.2.3 Least-Squares Matching

Basic Principle

Least-squares matching is an alternative approach to maximizing the cross-correlation between two image patches to estimate the interframe motion. Like correlation techniques, it also belongs to the region-based methods of motion estimation. Given two successive images $g_1 = g(\mathbf{x}, t_1)$ and $g_2 = g(\mathbf{x}, t_2)$, an interrogation window is selected in the first frame, and a larger search area centered around this interrogation window is selected in the second frame. The displacement of the interrogation window is calculated by *minimizing* a distance measure that quantifies the dissimilarity between two image regions. This distance measure is given by the *sum-of-squared differences* (SSD, the *least squares*) of the gray values within the interrogation window between the first and second frame:

$$d(\mathbf{x}, s) = \int_{-\infty}^{\infty} w(\mathbf{x} - \mathbf{x}') [g_1(\mathbf{x}') - g_2(\mathbf{x}' - s)]^2 d^2 \mathbf{x}', \quad (25.138)$$

where the weight function $w(\mathbf{x} - \mathbf{x}')$ represents the size of the interrogation window. The optical flow is approx-

imated as

$$f(\mathbf{x}, t) \approx \frac{1}{\Delta t} \arg \min d(\mathbf{x}, s), \quad (25.139)$$

where Δt is the time difference between the two successive images. Subpixel accuracy may be achieved using the same methods as in correlation-based approaches, e.g., by fitting a Gaussian function to the (inverse) displacement peak.

Least-squares matching is also referred to as *image correlation velocimetry* [25.19], *adaptive least-squares correlation* [25.20], and the *minimum quadratic differences* (MQD) method [25.54].

Relation to Other Region-Based Approaches

Least-squares matching is closely related to the other region-based approaches of motion estimation, namely optical flow techniques, and correlation-based analysis.

The similarity between least-squares matching and differential optical flow techniques (Sect. 25.2.5) is revealed by approximating $g_2(\mathbf{x}' - s)$ in (25.138) by a Taylor expansion about $s = 0$ and skipping all terms above first order. The resulting expression is the gradient-based formulation of the optical flow. For the case of subpixel motions, the equivalence of first-order differential optical flow estimation and least-squares matching using bilinear interpolation is shown by [25.55].

Gui and Merzkirch [25.54, 56] discuss the relation between least-squares matching and correlation techniques. Expanding the squared term in (25.139) shows that this expression contains the (negative) cross-correlation coefficient as used in a PIV evaluation. But in addition, there is a term accounting for nonuniformities in the particle image distribution and nonuniform illumination. Gui and Merzkirch [25.56] show that this term is responsible for the superiority of the least-squares matching as compared to conventional correlation-based methods.

Advanced Least-Squares Matching

The real strength of least-squares matching is revealed if it is combined with similar advanced evaluation methods as outlined in Sect. 25.2.2. In particular, iterative approaches using a coarse-to-fine strategy together with a higher-order approximation of the displacement field (Sect. 25.2.1) are widely used and show good performance [25.19]. Radiometric effects, i.e., intensity changes, may also be included in the model. The implementation of such methods within a general least-squares parameter estimation framework provides the

ability to estimate higher-order quantities such as vorticity and rate of strain directly. Towards this end, these quantities are introduced as parameters to be estimated within the least-squares optimization. Thus, no explicit finite differences of the velocity field, which are very sensitive to numerical errors and noise, have to be calculated. In addition, no explicit differentiation of the image data is required. In most cases, only a few parameters are extracted from the optimization (e.g., six parameters for a general 2-D affine transformation), but many more data points are used for the computation (e.g., 256 pixels in a 16×16 interrogation window). Due to this strong overdetermination, the least-squares matching is quite immune to image noise. Furthermore, the precision and the reliability of the estimated parameters can be easily assessed by a covariance analysis of the least-squares result. Overparameterization can be avoided by selecting a model of the displacement fields based on the significance of the computed parameters. Various models can be used for different interrogation windows, making the method adaptive to the local image structure.

25.2.4 Tracking Techniques

In this section, various tracking techniques are reviewed. Tracking techniques are feature-based motion estimation algorithms: individual tracer particle images are the features that are tracked throughout an image sequence. The most important tracking method used to measure fluid flow is *particle-tracking velocimetry* (PTV). This method is described in detail in this section.

Besides PTV, some region-based methods are also referred to as tracking techniques: *correlation-based tracking*, *least-squares tracking* or *Kanade–Lucas tracker* (KLT) tracking. The latter methods consider the particle or gray value patterns within the interrogation windows as features that are tracked. Correlation-based tracking ([25.18] and Sect. 25.2.2) is a special PIV evaluation mode, where the interrogation window in the second frame (the search window) is larger than that in the first frame, as opposed to the standard correlation-based interrogation mode of PIV where both windows are of equal size. Thus, in this method, the correlation coefficient is used as a tracking criterion. Least-squares tracking was discussed in Sect. 25.2.3. The KLT tracker [25.57] is a differential optical flow method based on the early work of Lucas and Kanade [25.58] (Sect. 25.2.5). This method performs tracking in the sense that individual image regions are automatically selected and tracked if the image structure within the regions is sufficient to compute the optical flow.

Particle-Tracking Velocimetry

Basic Principle. The basic idea of particle-tracking velocimetry (PTV) is to identify single particle images within an image, segment them from the background, and track them along their trajectories throughout an image sequence. Thus, PTV is a feature-based approaches to motion estimation. A PTV algorithm has to solve the following three tasks: *particle segmentation*, determination of *particle position*, and *particle matching*, i. e., solving the motion correspondence problem. Since in most applications, the particles cannot be distinguished from each other reliably (e.g., by their shape or intensity), the latter task is the most difficult, due to ambiguities occurring especially for high particle densities. Thus, the particle density in PTV applications is generally lower than in PIV applications.

As a simple example, consider the following optimal conditions for PTV: high image contrast (bright particles on a dark, noise-free background), low particle density, and small interframe motions. The latter two conditions imply that the mean distance between particles is much larger than their interframe motion. In this case, a very simple PTV approach may be used: segment the particles by a global intensity threshold, determine their position by centroiding, and match each particle in the first frame to that particle in the second frame that is closest to its position in the first frame (i.e., its nearest neighbor). However, since the optimal conditions assumed above will rarely be given in real applications, more-sophisticated algorithms are needed, in particular for particle segmentation (see below) and the tracking (Sect. 25.2.4).

Differences to Region-Based Approaches. The major differences between tracking techniques and region-based motion estimation are the following:

- **Temporal scope:** To increase the reliability of the tracking, most PTV techniques use more than two successive frames to establish the temporal correspondences (*multi-frame tracking*), e.g., three-frame tracking [25.59], four-frame tracking [25.26], five-frame tracking [25.60]. Note that there are also two-frame tracking techniques [25.24] as well as techniques that try to find the optimal set of trajectories by taking into account their complete (visible) length within a global optimization [25.61]. Most correlation-based approaches and least squares techniques try to establish a matching between only two frames, while in optical flow techniques, more than two frames may also be used, e.g., for a more-

accurate computation of temporal gradients or to stabilize the results by temporal smoothing [25.62].

- **Particle density and spatial resolution:** Since PTV aims at identifying individual particles and finding corresponding match partners in the next frame, the particle density is generally lower than in PIV, which results in a lower spatial resolution of the underlying flow field. On the other hand, the resolution given by the particle seeding is fully exploited without any averaging effects as in PIV (Sect. 25.2.2), and therefore with a higher accuracy. For example, the motion of two particles located within one PIV interrogation window can be resolved individually. Thus, the local spatial resolution is higher than in PIV.
- **Spatio-temporal distribution of velocity vectors:** Feature-based approaches compute the *Lagrangian representation* of a flow field, i. e., the result of such algorithms is a set of trajectories of the tracked objects. Thus, the velocity information is given at random locations depending on the tracer distribution and density. In contrast to region-based methods, no dense, instantaneous flow field defined on a regular grid is computed (which is the *Eulerian representation* of a flow field). On the other hand, tracking methods allow the motion of individual particles to be followed in time, enabling e.g., a Lagrangian study of diffusion, which is not possible using PIV results.
- **Large motions:** Since feature-based methods do not exploit the relation of spatial and temporal intensity gradients for velocity estimation, they are less sensitive to violations of the sampling theorem. Thus, feature-based methods are more suitable for the handling of larger motions, given that the motion correspondence problem can be solved using one of the advanced methods outlined below.

Particle Segmentation. In the particle segmentation step, a decision has to be made for each pixel of the image, whether it belongs to a particle or to the image background. Thus, for an image $g(\mathbf{x}, t)$, the segmented image $g_s(\mathbf{x}, t)$ is given by the following operation:

$$g_s(\mathbf{x}, t) = \begin{cases} 1 & : g(\mathbf{x}, t) \in \text{object(particle)} \\ 0 & : g(\mathbf{x}, t) \in \text{image background} \end{cases} \quad (25.140)$$

The result of the segmentation is a binary image in which the particle images are marked with the value 1 and the background is marked with the value 0.

Particle segmentation is rather difficult because most particle images do not have a bimodal gray value histogram. Therefore, a simple segmentation by a global intensity threshold is not feasible. Particle images do not have a uniform (mean) gray value for the following reasons:

- *Image noise*: Image noise introduces false positives and false negatives, especially in the segmentation of low-contrast particles.
- *Motion blur*: While small and slow particles are imaged as bright circular spots, the shape of the image of a faster particle is elongated in the direction of its motion due to the integration time of the camera. Faster particles cover a larger area in the image, i. e., the irradiance is distributed over a larger number of pixels. Thus, faster particles have a lower intensity and may appear as faint objects with gray values close to the image background.
- *Inhomogeneous illumination*: Several factors may contribute to an inhomogeneous illumination of the images, e.g., the general intensity distribution of the light source, glow of dirty water, or particles reflecting light to their neighbors. In particular, for 3-D applications using a volume illumination, inhomogeneities have to be taken into account.
- *Out-of-focus imaging*: In applications with volume illumination, particles may be out-of-focus, which reduces the image contrast.

As a result of all these factors, the (mean) gray values of particle images vary locally and may cover the complete range from the background noise level to the saturation of the sensor. Many authors have pointed out that the particle segmentation is one of the most crucial steps in a PTV algorithm, since it is the dominating factor controlling both the reliability and the accuracy of the tracking [25.63].

Another difficulty is introduced by overlapping particles. Especially in 3-D applications using volume illumination, particle images may partly overlap, or particles may be completely occluded by others. Particle occlusion is a principal physical limitation of 3-D techniques using a volume illumination. A trade-off has to be made between the seeding density (i. e., spatial resolution of the flow field) and the depth of view. Different techniques to resolve overlapping particle images are discussed by [25.64, 65].

Particle Position. Once the particle images have been segmented from the background, various methods are available for the subpixel-accurate determination of

the position of individual particles: centroiding, least squares fits, e.g., in the form of a Gaussian three-point estimator as applied in PIV, fit of extended models taking into account the motion of the particles or template matching by cross-correlation.

The performance of the different methods depends on the properties of the images, e.g., particle size and density, particle contrast, and image noise. In general, errors in the determination of the particle positions are introduced by the discretization, noise in the signal (photon shot noise), and noise related to the image acquisition. The discretization errors for typical particle images are of the order of magnitude of 0.01–0.1 pixel [25.29]. Wernet and Pline [25.60] show that the Cramer–Rao lower bound for the error in determining the position of a Gaussian-shaped particle is 0.015 pixel. In practical applications with additional noise sources related to the image acquisition, the mean centroid estimation error is generally larger (0.1–0.2 pixel). If highest accuracy is desired, the bias towards integer positions becomes important (peak-locking as discussed in Sect. 25.2.2). In this case, iterative weighted least-squares methods have to be applied to compute an unbiased estimate.

Computation of Velocity. If a unique particle match is found, a first-order approximation of the particle's velocity may be computed by subtracting the positions of the particle in the two successive frames and dividing by the frame period of the camera. Assuming that the latter is given free of error, the absolute error in estimating the particle displacement is $\sqrt{2}$ times the particle position error. Therefore, errors of individual velocity vectors may be higher than in PIV. However, Wernet and Pline [25.60] have shown that better accuracy than with PIV can be achieved by averaging the PTV results over an area of the size of a typical PIV interrogation window. Higher-order finite differences or spline fits to particle trajectories may also be used to achieve a better quality of the velocity results. Further, the errors in the particle positions of two nearby particles in successive frames are typically highly correlated. The systematic error cancels out in the difference of the two positions. Hence, the error in the velocity estimation will be lower than the conservative estimate given above.

Velocity Postprocessing. Techniques to remove outliers in the velocity field and interpolate the resulting gaps as well as techniques to interpolate randomly spaced data to a regular grid have been mentioned in Sect. 25.2.1. Similar techniques may be applied to PTV data, where

outliers have to be defined relative to a particle trajectory. *Malic* and *Dracos* [25.66] present dedicated interpolation schemes for three-dimensional velocity fields from scattered data using Taylor expansions. Besides the interpolation to regular grids, the final **PTV** results (i. e., the flow trajectories) enable a Lagrangian analysis of the flow field.

Limitations. An important parameter describing the difficulty of particle tracking is the particle spacing displacement ratio r_p [25.27]:

$$r_p = \frac{\Lambda_0}{\Lambda_t}, \quad (25.141)$$

where Λ_0 is the average distance between the particles and Λ_t is the average particle displacement between two successive frames. Tracking is easy, if $r_p \gg 1$. In this case, even simple nearest-neighbor approaches such as the one described above may yield good results. The tracking difficulty increases for $r_p \approx 1$, and tracking becomes virtually impossible for $r_p \ll 1$. In the latter case, the probability of ambiguities in the particle matching is very high. The motion correspondence problem cannot be solved if no additional information about the particles or their motion (e.g., size, shape, color, intensity, direction of motion) is available. The fundamental reason is the violation of the sampling theorem (Sect. 25.2.1) in the case of high particle density and large motion.

Thus, for a given particle density, the tracking difficulty is related to the maximum possible displacement between two frames, which depends on the frame rate, image magnification, and the flow field under investigation. For a reliable tracking, Λ_t should be small, i. e., the frame rate of the camera should be sufficiently high. On the other hand, a larger Λ_t yields a lower relative error of the velocity vector, since the absolute error in determining particle positions is independent of Λ_t . The basic idea to enable reliable particle tracking for values of $r_p \approx 1$ and smaller is to take into account additional information about the flow and use this information to guide the particle matching. Towards this end, most **PTV** approaches introduce a motion model (Sect. 25.2.1). Some advanced **PTV** techniques based on motion models are discussed in Sect. 25.2.4.

Another difficulty in **PTV** stems from the fact that the number of particles in two successive frames may not be equal. Even if the number of particles is equal, the assumption that the same set of physical particles

is visible in both images is not valid. Instead, a **PTV** algorithm has to handle the following events:

- Entry respective trajectory initialization: Particles may enter the field of view. These particles do not have a correspondence partner in the previous frame.
- Exit respective trajectory termination: Particles may leave the field of view. These particles do not have a correspondence partner in the following frame.
- False positives and false negatives: Image noise may result in spurious particle images at locations where actually no particle is visible. Segmentation failures may result in a loss of particles, e.g., particles are not segmented due to their low intensity.
- Overlapping particles and occlusion: Especially for high particle densities, an overlap and occlusion handling has to be introduced.

Since all these events and their consequences have to be considered for each particle in each image of a sequence, **PTV** algorithms tend to be quite complex. In the following sections, some advanced **PTV** techniques are discussed.

Advanced Techniques to Solve the Motion Correspondence Problem

As discussed in the previous section, the most difficult step of a **PTV** algorithm is to find the unique, correct solution to the *motion correspondence* problem: all particles in an image have to be associated to their correct matching partners in the next frame. This association is difficult for higher particle densities, since the probability of *ambiguities* becomes large, i. e., there will be several possible matching partners within a search region in the next frame. The basic idea of advanced **PTV** algorithms is to use additional *a priori* knowledge and assumptions about the flow field. Based on such information, a model of the flow can be introduced. Further, kinematic constraints can be used to reduce the number of corresponding particles and thus the number of ambiguities in the matching. The reduction of the number of ambiguities becomes possible for two reasons. First, using a model of the flow field enables the *prediction* of the future position of particles by extrapolation. Thus, the search region for the correspondence search may be located at this predicted position. Second, if the flow model is a good approximation of the actual flow field, there will be a high probability that the predicted position is already at the correct location of the match partner. Thus, the size of the search region may be reduced. If there is only one particle within the predicted search region,

a unique match is established. If still several matching partners remain, one of them may be chosen as the correct match according to certain criteria, which are again based on additional information or assumptions on the flow.

The main assumption that is made in many tracking algorithms is the *smoothness of the flow field*, which is based on the physical principle of inertia. Due to inertia, the motion of an object will not change abruptly between two frames, given that the frame rate is sufficiently high. Inertia is related to the *temporal* smoothness of particle trajectories. In addition, *spatial smoothness* (or *spatial coherence*) of the velocity field may also be assumed. In particular, for incompressible, viscous flows, the velocity vectors within a spatial neighborhood will vary smoothly. Thus, velocity vectors next to each other will be similar in speed and direction.

In addition to the basic smoothness assumptions, any a priori knowledge about the flow field may also be incorporated into the tracking. For example, if the flow is known to have a mean bulk velocity, this velocity can be used as an offset in the tracking algorithm. Search radii defining the area where matching particles are supposed to be found can be defined based on *hydromechanic knowledge*, e.g., maximum expected velocities or turbulence scales such as the Kolmogorov scales or Taylor microscales [25.27]. Such considerations are particularly important for the initialization of trajectories, since for particles entering the field of view, no velocity vector is available that can be used to predict their next position.

Another approach to resolve ambiguities in the correspondence analysis is to take into account several possible matches and defer the decision of the correct match on later frames. Such approaches are realized using techniques of *statistical data association* or *combinatorial optimization*. These techniques can also deal with particle occlusion and therefore resolve crossing trajectories. This is possible because a larger temporal scope is taken into account when solving the correspondence problem, e.g., a temporal neighborhood of three previous and three future frames. The correspondence is solved by finding an optimal set of trajectories within this temporal neighborhood, where optimality is expressed, e.g., in terms of trajectory smoothness.

Finally, some remarks concerning the optimal choice of thresholds and other tracking parameters, e.g., size of search regions, shall be made. In many cases, the optimal parameters depend on the flow conditions, i. e., particle density and flow velocity. However, the latter are not

constant throughout the whole image, but there may be significant variations of these quantities within a single image. Obviously, optimal performance of a tracking algorithm cannot be achieved using a fixed set of parameters. Instead, the parameters should adapt to the local flow conditions. For example, it does not make sense to use a search region based on a global maximum velocity constraint within a region of the image where the velocity is very small and the particle density is very high. In this case, a search region based on a maximum velocity constraint will be much too large. It may also be advantageous to adapt the shape of the search region to the flow conditions. For example, if there is a prominent main flow direction, the search region should be elongated along this direction.

In the following subsections, some implementations of the ideas developed in this section are presented.

Two-Frame Tracking. The simplest two-frame tracking technique performs a nearest neighbor search based on a minimum-velocity constraint. *Hering et al.* [25.67] use the spatial overlap of particle images to identify the nearest neighbor. The overlap is caused by overlapping integration times of the even and odd fields in an interlaced camera frame. If non-interlaced cameras are used, the overlap may be created artificially by a morphological dilation. The approach is limited to low particle densities.

The performance of two-frame tracking can be increased towards higher particle densities by including a spatial coherence constraint and requiring velocity vectors within a spatial neighborhood to be similar [25.25].

Multiframe Tracking. Multiframe techniques are based on the assumption of trajectory smoothness. They use a model of the particle motion to predict the particle positions in the next frame. This model is given by the Taylor expansion of the particle trajectory equation (25.135). The difference in these techniques is the degree of approximation in the Taylor expansion. *Three-frame techniques* use the actual frame and one previous frame to compute a first-order approximation of the particle velocity. The resulting velocity vector is used to predict the particle position in the next frame, assuming that the velocity stays constant. If several match candidates are found within a neighborhood around the predicted particle position, the particle with the smallest distance to the predicted position is chosen. This choice corresponds to a *minimum acceleration constraint* on the particle motion. In a similar manner, higher-order terms in the Taylor expansion may be taken into ac-

count to improve the accuracy of the predicted particle position [25.27].

From the point of view of implementation, imposing the temporal smoothness constraint is actually a matter of defining search areas around predicted particle positions. In most applications, circular areas are used. The radii of these areas are chosen according to the kinematic constraints imposed by the flow model or according to prior hydromechanic knowledge about the flow, for example, to initialize a trajectory, the search radius in the second frame may be defined by the maximum expected velocity. The centers of the search areas in the third and fourth frame are predicted by extrapolating the model, while their radii may be chosen according to the expected fluctuations in the velocity. The latter can be estimated from the Kolmogorov scales of the flow and the imaging parameters [25.27].

Combinatorial Optimization. The motion correspondence problem between two sets of features in two successive frames may be formulated as a *combinatorial optimization problem*. Based on such a formulation, results and algorithms developed in graph theory and operations research may be applied to PTV.

Given are two sets of particle coordinates P_1 and P_2 (particle images in the first frame at $t = t_1$ and particle images in the second frame at $t = t_2 = t_1 + \Delta t$):

$$P_1 = \{p_{1,i}, i = 1..N_1\} , \quad (25.142)$$

$$P_2 = \{p_{2,j}, j = 1..N_2\} , \quad (25.143)$$

where $p_{i,j} = (x_{i,j}, y_{i,j})$ are the coordinates of particle j in frame i . The particle matching between two successive frames may be described by an association matrix $\alpha = (\alpha_{ij})$, with $\alpha_{ij} = 1$ if particle i in the first frame is matched with particle j in the second frame and $\alpha_{ij} = 0$ elsewhere. The task is to find an *optimal assignment* between the elements of the first and those of the second set.

Optimality is expressed in terms of an objective function d that is linear in the associations between the two sets:

$$d = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} \alpha_{ij} c_{ij} , \quad (25.144)$$

where c_{ij} is the cost for the association α_{ij} . The costs c_{ij} are chosen according to the kinematic constraints discussed in the previous section, e.g., favoring smooth trajectories. The optimal assignment is determined by

minimizing the objective function (25.144). This formulation of the two-frame tracking is equivalent to a so-called *bipartite graph matching* or *assignment problem*, which is a basic problem of combinatorial optimization occurring in many applications. Efficient algorithms to solve it have been developed [25.68]. For an example of a PTV application based on bipartite graph matching (see [25.69]). *Stellmacher* and *Obermayer* [25.70] present an interesting approach to simultaneously estimate particle correspondences and a local affine transformation by applying a combined discrete and continuous optimization method. This method has been originally proposed by *Gold* et al. [25.71] for solving point-matching problems in statistical pattern recognition.

If the temporal scope of the tracking is extended to three or more frames, the problem becomes a *multidimensional assignment problem*. Such problems are known to be *NP*-complete, i. e. there is no efficient algorithm to compute their solution [25.68]. However, approximate solutions can be found using greedy search techniques and other heuristics [25.61]. The complexity of the correspondence analysis is significantly increased due to the extended temporal scope. Therefore, combinatorial techniques are computationally expensive. On the other hand, these techniques are also able to resolve crossing trajectories and find an optimal set of trajectories in the presence of particle occlusions and dropouts due to segmentation failures.

Statistical Techniques. Statistical approaches to solve the motion correspondence problem (*statistical data association*) have been originally developed in the context of radar target tracking and surveillance, where particle tracking is referred to as multiple target tracking. A large number of different approaches and algorithms has been published. For an introduction see [25.72, 73]. A review of statistical data association techniques in the context of computer vision is given by *Cox* [25.74].

Statistical data association techniques are specifically developed to resolve ambiguities in motion correspondences, including events like track initiation and termination, particle occlusion, and false positives/negatives. Like the other advanced techniques discussed so far, most statistical techniques are also based on the two paradigms of prediction of an optimal search region and postponing assignment decisions by examining subsequent frames. The difference of statistical approaches is that they model the motion of a particle over time as a *stochastic process*. The optimal particle state according to the measurements and the underly-

ing model is then computed within the framework of Bayesian inference [25.75]. Within this framework, the *probability distribution of the particle state* is propagated over time and updated by the measurements in each frame. A (stochastic) state vector is introduced, which describes the actual state of a particle, e.g., its position, velocity and acceleration. This state vector follows a probability distribution, the so-called *prior distribution*. When measurements of the particle state become available (e.g., its position resulting from the particle segmentation), this measurement information is converted into a *likelihood function* defined on the particle state space. Likelihood functions are presumed to contain all the relevant information in the observed data. They provide a probabilistically correct method of combining all types of sensor information and incorporating it into a tracker's estimate of the particle state. Note that this may include the detailed physics of the camera response and its noise characteristics. Bayes rule is applied to combine all this information and compute the *posterior distribution* on the target state by combining the prior distribution with the likelihood function. Finally, optimal particle associations are computed from these probability distributions. All statistical data association techniques may be formulated within this *Bayesian framework*.

The simplest statistical tracking technique is the *Kalman filter* [25.73, 76]. It is based on the assumptions of a linear model of particle motion (like (25.135)) and Gaussian distributions of particle state and measurement error. The Kalman filter computes the optimal position of a particle in the next frame, by predicting its position according to a model and combining this prediction with a measurement. Note that the prediction also includes the precision of the particle location. Optimality is achieved by taking the errors of the prediction and the measurement into account. For example, the shape of the search region around the predicted position may be chosen according to the covariance of the predicted position, which typically results in elliptical search regions instead of circular ones. Examples of Kalman filters applied in *PTV* are described in [25.28].

Simple Kalman filters have several drawbacks. They provide a statistically optimal estimate of particle position and thus can guide the tracking and implicitly reduce ambiguities. However, they cannot handle ambiguities explicitly. An extension of a Kalman filter with increased capabilities in the handling of ambiguities is the *multiple hypothesis tracker* (*MHT*) [25.73, 77]. In the case of ambiguities, the latter takes into account the *k*-best hypotheses (computed by a Kalman filter). Probabilities for

all the hypotheses are computed and the most probable hypothesis is chosen. Thus, the *MHT* is a combined approach based on statistical reasoning and combinatorial optimization.

In complex tracking problems (such as turbulent motion), the assumptions of the Kalman filter, i. e., a linear model and Gaussian probability distributions, may be too simplistic. It is possible to relax these assumptions and work with general probability distributions of arbitrary shape as well as arbitrary nonlinear models. This general framework of *Bayesian multiple-target tracking* is discussed in the book by Stone et al. [25.75].

Hybrid Methods Using Velocity Estimation Techniques. Hybrid methods combining a *PIV* velocity estimation with the tracking of single particles [25.22, 63, 78] have already been discussed in Sect. 25.2.2. Bastiaans et al. [25.23] present a hybrid method combining a *PIV* prediction step with a tracking step based on combinatorial optimization. An important condition for hybrid *PIV/PTV* methods is the spatial coherence of the flow field, since only then the *PIV* prediction step will yield a reasonable estimate of the local velocity.

3-D/3-C Tracking

PTV can be extended to *3-D* measurements using the same basic techniques as in *3-D PIV*, namely stereoscopic or multicamera image acquisition. In addition to solving the motion correspondence problem (temporal correspondence), a *3-D PTV* algorithm has to solve the *stereo correspondence problem* (spatial correspondence), i. e., to find corresponding particles in two or more images taken from different viewpoints. It is necessary to establish the stereoscopic correspondences of particles uniquely in order to compute their *3-D* position by triangulation of the optical rays.

Basically, there are two different approaches to *3-D PTV*. They differ in the order in which the spatial and temporal correspondences are solved. One approach first performs particle tracking in the image planes, resulting in a set of *2-D* particle trajectories. Thus, the motion correspondence is solved first. Then, the stereo correspondences of the particle trajectories are analyzed to find the *3-D* coordinates [25.29]. The other approach first solves the stereoscopic correspondences of single particles to compute their *3-D* coordinates. Afterwards, particle tracking is performed in *3-D* space [25.27, 64]. For this approach, three or more views of the flow field are needed in order to resolve stereoscopic ambiguities in the determination of the particle positions.

A recent approach is to combine temporal and spatial information and solve the correspondence analysis simultaneously [25.79].

The goal of **3-D PTV** is to measure the three components (3-C) of the velocity vectors within a **3-D** volume in space. Towards this end, a volume illumination has to be used instead of a light sheet that is typically used in 2-D applications. The volume illumination introduces a number of difficulties concerning the imaging of particles and the processing of such images:

- **Particle occlusion:** Particles located along the same optical ray occlude each other, which limits the particle density or the depth of view. A trade-off between these two parameters has to be chosen according to the goals of the measurement.
- **Projection of the 3-D scene:** A particle image obtained using volume illumination is the 2-D projection of a **3-D** scene with a certain depth range. Depending on the particle density, this will increase the probability of spatial overlap of the particle images or trajectories crossing each other in the image plane. In addition, the assumption of spatial coherence of the projected flow field is no longer valid, since particles that are actually far away from each other in the **3-D** volume may be located next to each other in the image.
- **Out-of-focus imaging:** If the illuminated volume is larger than the depth of field of the lens, some particles will be out of focus, resulting in a larger size and lower contrast of these particles. Such effects have to be considered in the particle segmentation.

Further Readings

Further information about tracking methods can be found in [25.81, 82] and [25.83] in the context of computer vision and in [25.36] in the context of flow measurement. In-depth textbooks on Kalman filtering in the context of radar applications are available [25.72, 73, 75]. Many ideas and techniques described there can also be successfully applied in visual tracking applications like **PTV**. Cook et al. [25.84] and Nemhauser and Wolsey [25.68] discuss algorithms to solve combinatorial optimization problems, similar to those arising in tracking applications with increased temporal scope.

25.2.5 Optical-Flow-Based Velocity Analysis

Optical flow can be considered as the distribution of apparent velocities of movement of brightness patterns in

an image [25.85]. More precisely the optical flow is an approximation to the *two-dimensional* motion field of an image sequence. We obtain this motion field by projecting the three-dimensional velocities of object points in three-dimensional space onto the two-dimensional image plane [25.86].

In estimating the optical flow and in determining motion fields from the estimated optical flow, there arise a number of peculiarities and difficulties, which have to be taken into account:

- There may be regions in the images where no motion can be determined (the *black wall problem*) or where only the normal velocity component can be determined (the *aperture problem*). Figure 25.15 illustrates these situations. In order to obtain a dense flow field an interpolation or a regularization technique has to be applied. This is an intrinsic problem in optical flow computation, and it is addressed throughout the chapter.
- There may be regions in the image, where the motion is nontranslational. Especially for applications in fluid dynamics the nonrigid behavior of fluids has to be taken into account. For these situations appropriate models have been developed.
- For large displacements the temporal sampling theorem may be violated. Coarse-to-fine techniques

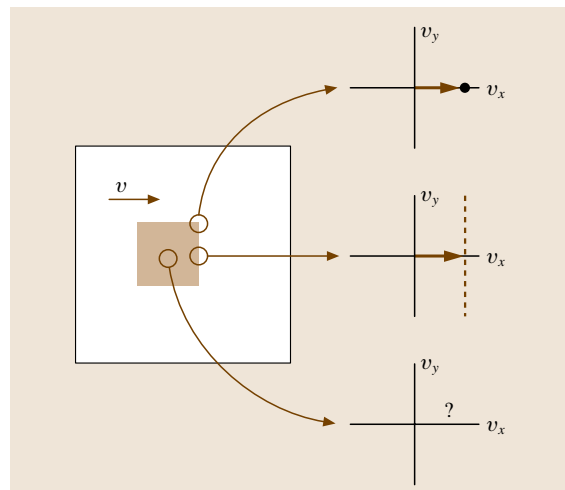


Fig. 25.15 The black wall and aperture problems. At the corners of the moving square, a unique velocity can be estimated. On the edges of the squares only the normal component of the velocity is determinable. In the center the motion is completely unconstrained (from [25.80])

applied on bandpass-filtered image sequences help to cope with this problem.

- In the presence of motion discontinuities, reflections or corrupted pixels, parameterized flow field models fail. Using a robust approach, described in the paragraph on *robust estimation* below, we can determine a correct motion field even in these situations.
- Brightness changes have to be taken into account. Illumination changes lead to a misinterpretation of the optical flow field. This can be nicely demonstrated by considering a rigid sphere with homogeneous surface reflectance, spinning around an axis through the center of the sphere (Fig. 25.16). If the surface is not textured and the illumination stays constant, the optical flow field would be zero over the entire sphere. If a directional light source moves around the same sphere the illumination changes would be falsely attributed to motion of the sphere surface. In many situations we can apply physical models of brightness variations to deal with brightness changes.

Optical flow methods can be classified as belonging to one of these groups:

- Differential techniques, which compute image velocity from spatiotemporal intensity derivatives.
- Frequency-based techniques, which use energy/phase information in the output of velocity tuned filters.
- Tensor-based techniques, which deduce the motion field from the local image brightness distribution represented as a structure tensor.

Though the attempts to estimate optical-flow look very different at first glance, all of these approaches

are closely related. *Simoncelli* [25.80] showed that differential techniques are equivalent to frequency-based technique, provided that the derivatives and filters are chosen appropriately. The structure tensor can be constructed based on operations in the spatiotemporal domain [25.62], but it can also be obtained by linear combinations of outputs of filters in frequency-domain [25.87]. *Jähne et al.*[25.29] and *Barron et al.*[25.86] provide a well-founded overview on the field of optical-flow estimation and give a quantitative comparison of the results.

Though the method of optical-flow is quite common in computer vision, the extent of application in experimental fluid dynamics is relatively small, so far. Therefore in the following we will provide a detailed review of the optical flow methods classified above, and we will show, how to cope with each of the difficulties listed above. We conclude this chapter with a literature review.

Optical-Flow Methods

Differential Techniques. The general task is to determine the optical flow field $\mathbf{f} = (f_1, f_2)^T = (dx/dt, dy/dt)^T$ from the gray values of an image sequence. In the simplest case it is assumed, that the gray value $g(\mathbf{x}, t)$ along a path $\mathbf{x}(t)$ remains constant for all time:

$$g(\mathbf{x}(t), t) = \text{const.}$$

By taking the temporal derivative on both sides and applying the chain rule one obtains:

$$\frac{dg}{dt} = \frac{\partial g}{\partial x} \frac{dx}{dt} + \frac{\partial g}{\partial y} \frac{dy}{dt} + \frac{\partial g}{\partial t} = 0.$$

Writing down in vector notation using $\nabla g = (\partial g/\partial x, \partial g/\partial y)^T$ this yields the brightness constancy constraint equation (BCCE):

$$(\nabla g)^T \mathbf{f} + g_t = 0. \quad (25.145)$$

Because the partial derivatives $g_x = \partial g/\partial x$, $g_y = \partial g/\partial y$ and $g_t = \partial g/\partial t$ are accessible by the application of a derivative filter, one obtains one constraint for the two-component flow field. Dealing with two unknowns in one equation we have an ill-posed problem. Graphically spoken the solution of (25.145) determines a *line*, containing all vectors that are possible candidates for the true optical flow vector (Fig. 25.15). Without further assumptions only the flow perpendicular to the constraint line can be estimated. This problem is commonly referred as the *aperture problem* of motion estimation.

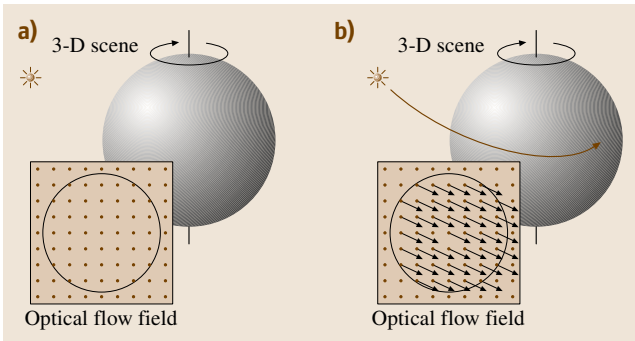


Fig. 25.16a,b Illumination changes and optical-flow. (a) Spinning sphere with fixed illumination leads to zero optical flow. (b) Moving illumination source causes apparent optical-flow field without motion of the sphere (after [25.29])

Solving (25.145) for points \mathbf{x}_U in a sufficiently large neighborhood U around \mathbf{x} we may get other constraint lines, so that we can determine the true optical flow vector by the intersecting point of the constraint lines. But the neighborhood must not be chosen too large, because it is not assured, that the motion is constant in a larger area. How large to choose the neighborhood is referred as the *generalized aperture problem*. One way to weaken the assumption of constant motion is finding a local parameterization of the flow field, so that one demands local coherency instead of local constancy. This leads us to the second constraint of differential optical flow estimation, the *spatial coherence constraint*.

The concept of optical flow originates from hydrodynamics. Grey values *flow* over the image plane, like volume elements flow in fluids. In hydrodynamics the principle of conservation of mass is formulated by the continuity equation, which reads in its differential form

$$\frac{\partial \rho}{\partial t} + \nabla(\rho \mathbf{u}) = \frac{\partial \rho}{\partial t} + \mathbf{u} \nabla \rho + \rho \nabla \mathbf{u} = 0. \quad (25.146)$$

The three-dimensional velocity \mathbf{u} of a fluid element with density ρ in three-dimensional space is apparently analogous to the two-dimensional optical flow \mathbf{f} of a gray value g in two-dimensional space. The BCCE (25.145) corresponds to the continuity equation (25.146), if one drops $\rho \nabla \mathbf{u}$. Why do we have to drop the last term? Consider an object moving away from the camera. In this case the total brightness change dg/dx of a gray value on a path $x(t)$ is zero, because the irradiance in the image plane remains the same for an object moving perpendicular to the image plane. Because both g_t and ∇g are zero, we can apply (25.145). But we cannot apply (25.146), because the additional term $\rho \nabla \mathbf{u}$ or $g \nabla \mathbf{f}$ would *not* be zero, because the motion is not divergence free.

However, under certain conditions the use of a two-dimensional continuity equation instead of the BCCE can be motivated. This is the case, if one deals with 2-D transmittance images of a 3-D fluid flow, so that the imaged 2-D flow is the density weighted average of the physical 3-D flow [25.88, 89]. The constraint on the data now becomes

$$g_x f_1 + g_y f_2 + g f_1 + g f_2 + g_t = 0. \quad (25.147)$$

Local Weighted Least Squares. Assuming the optical-flow to be constant within a small neighborhood, [25.58] proposed a local method to estimate the optical-flow. Goal is to minimize the squared left-hand side of the BCCE (25.145) in a local neighborhood U around \mathbf{x} , which is given by the weighting (or window) function

$w(\mathbf{x} - \mathbf{x}')$:

$$\hat{\mathbf{f}} = \arg \min_{\mathbf{f}} \int_{-\infty}^{\infty} w(\mathbf{x} - \mathbf{x}') [(\nabla g)^T \mathbf{f} + g_t]^2 d\mathbf{x}'.$$

The weighting function in the simplest case is given by a box-filter (all points in the neighborhood are weighted equally), but better results can be achieved using a binomial filter. Standard least-squares minimization (setting the partial derivatives of the functional with respect to f_1 and f_2 to zero) yields the equation system

$$\underbrace{\begin{pmatrix} \langle g_x g_x \rangle & \langle g_x g_y \rangle \\ \langle g_x g_y \rangle & \langle g_y g_y \rangle \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} f_1 \\ f_2 \end{pmatrix}}_{\mathbf{f}} = - \underbrace{\begin{pmatrix} \langle g_x g_t \rangle \\ \langle g_y g_t \rangle \end{pmatrix}}_{\mathbf{b}} \quad (25.148)$$

with the abbreviation

$$\langle a \rangle = \int_{-\infty}^{\infty} w(\mathbf{x} - \mathbf{x}') a d\mathbf{x}'.$$

The solution of (25.148) is given by

$$\mathbf{f} = \mathbf{A}^{-1} \mathbf{b},$$

provided that the inverse of \mathbf{A} exists, i. e.. the determinant of \mathbf{A} is unequal to zero:

$$\det \mathbf{A} = \langle g_x g_x \rangle \langle g_y g_y \rangle - \langle g_x g_y \rangle^2 \neq 0.$$

This is not the case, if all spatial derivatives in the local neighborhood are zero (the *black wall problem*), or if all gradients in the local neighborhood point into the same direction (the *aperture problem*).

Global Constraints. Instead of assuming local spatial constancy (and therefore coherence) by introducing a window function we can demand global spatial coherence. One can determine the optical-flow by minimizing the BCCE (25.145) over the entire image Ω . To make the problem well-posed an additional term, the regularizing spatial coherence constraint $\|\mathbf{e}_S^2\|$, is introduced:

$$\hat{\mathbf{f}} = \arg \min_{\mathbf{f}} \int_{\Omega} [(\nabla g)^T \mathbf{f} + g_t]^2 d\mathbf{x}' + \lambda^2 \|\mathbf{e}_S^2\|. \quad (25.149)$$

The parameter λ controls the influence of the spatial coherence term. Horn et al.[25.85] propose global smoothness for the spatial coherence constraint:

$$\|\mathbf{e}_S^2\| = \int_{\Omega} \|\nabla f_1(\mathbf{x}')\|^2 + \|\nabla f_2(\mathbf{x}')\|^2 d\mathbf{x}'.$$

There are other suggestions for $\|e_S^2\|$, which may be better suited to special kinds of problems (e.g., fluid flow analysis). To solve the minimization problem a variational approach can be adapted. Thus, the integral equation (25.149) can be solved by a system of Euler–Lagrange equations:

$$\begin{aligned} L_{f_1} - \frac{\partial}{\partial x} L_{f_{1x}} - \frac{\partial}{\partial y} L_{f_{1y}} &= 0, \\ L_{f_2} - \frac{\partial}{\partial x} L_{f_{2x}} - \frac{\partial}{\partial y} L_{f_{2y}} &= 0. \end{aligned}$$

The integrand of (25.149) can be identified with the Lagrange function:

$$\begin{aligned} L = & (g_x f_1 + g_y f_2 + g_t)^2 \\ & + \lambda^2 (f_{1x}^2 + f_{1y}^2 + f_{2x}^2 + f_{2y}^2), \end{aligned}$$

L plugged into the Euler–Lagrange equations yields the diffusion–reaction system

$$((\nabla g)^T f + g_t) \nabla g - \lambda^2 \nabla^2 f = 0.$$

For the case, that there is a high spatial gray value variation (that means ∇g is large), the first summand dominates in the equation, and the optical flow is calculated using the BCCE. But if there is a *black wall problem*, the optical flow is calculated from the last summand, which states the Laplacian equation $\nabla^2 f = 0$.

The discretization can be performed using finite differences or finite elements. Once guaranteed, that the problem is well-posed, there exist a number of minimization schemes like Gauss–Jordan elimination or Gauss–Seidel iteration, which can be applied.

Frequency-Based Techniques. The concept of identifying sequences of 2-D images as 3-D spatiotemporal structures allows one to analyze motion in the corresponding spatiotemporal frequency domain (the Fourier domain). Let $g(\mathbf{x}, t)$ be an image sequence of any pattern moving with constant velocity, causing the optical flow \mathbf{f} at any point in the image plane, the resulting spatiotemporal structure can be described by

$$g(\mathbf{x}, t) = g(\mathbf{x} - \mathbf{f}t). \quad (25.150)$$

The spatiotemporal Fourier transform $\hat{g}(\mathbf{k}, \omega)$ of equation (25.150) is given by

$$\hat{g}(\mathbf{k}, \omega) = \hat{g}(\mathbf{k}) \delta(\mathbf{k}^T \omega), \quad (25.151)$$

where $\hat{g}(\mathbf{k})$ is the spatial Fourier transform of the pattern, and $\delta(\cdot)$ denotes Dirac’s delta distribution. This equation states, that the three-dimensional Fourier spectrum of a pattern moving with constant velocity condenses to

a plane in Fourier space. The plane equation in Fourier domain is given by the argument of the delta distribution in (25.151) and can be considered as an alternative formulation of the BCCE (25.145):

$$\omega(\mathbf{k}, \mathbf{f}) = \mathbf{k}^T \mathbf{f}. \quad (25.152)$$

Taking the derivatives of $\omega(\mathbf{k}, \mathbf{f})$ with respect to k_x and k_y yields both components of the optical flow:

$$\mathbf{f} = \nabla_{\mathbf{k}} \omega(\mathbf{k}, \mathbf{f}).$$

Quadrature-filter techniques try to estimate the orientation of this plane by using velocity tuned filters in the Fourier-domain. A quadrature-filter pair is a real frequency selective filter together with its imaginary Hilbert transform (Sect. 25.1.7). Its transfer function can be written in complex notation $\hat{q}(\mathbf{k})$.

The most common quadrature-filter pair is the *Gabor filter*, which selects a certain spatiotemporal frequency region with a Gaussian window centered at (\mathbf{k}_0, ω_0) . Its complex transfer function is

$$\hat{G}(\mathbf{k}, \omega) = \exp \left(-\frac{1}{2} \sqrt{(\mathbf{k} - \mathbf{k}_0)^2 + (\omega - \omega_0)^2} \sigma^2 \right).$$

From this the spatiotemporal filter mask can be computed using the shift theorem:

$$\begin{aligned} G(\mathbf{x}, t) = & \frac{1}{(2\pi)^{3/2} \sigma^3} \exp[i(\mathbf{k}_0 \mathbf{x} + \omega_0 t)] \\ & \exp \left[-\left(\frac{x^2 + y^2 + t^2}{2\sigma^2} \right) \right]. \end{aligned}$$

By applying this filter for different parameter sets (\mathbf{k}, ω) on the original spatiotemporal image we get estimates of the spectral density (or *energy*) of the corresponding periodic image structure belonging to these parameter sets. Ideally, for a single translational motion, the responses of these filters are concentrated about a plane in \mathbf{k} – ω -space, so that we are able to get the optical flow by a least-squares fit to the data. Another way to calculate the optical flow is by constructing a structure tensor composed of the filter outputs [25.90].

Tensor-Based Techniques. Optical-flow estimation can be formulated as orientation analysis in a three-dimensional spatiotemporal image. The concept of orientation analysis of a pattern in 2-D (Fig. 25.17a) can be generalized to 3-D (Fig. 25.17b). Any constantly moving gray value structure causes inclined patterns. Goal of tensor-based optical-flow estimation is to find the orientation of these patterns, provided that there exist any oriented patterns.

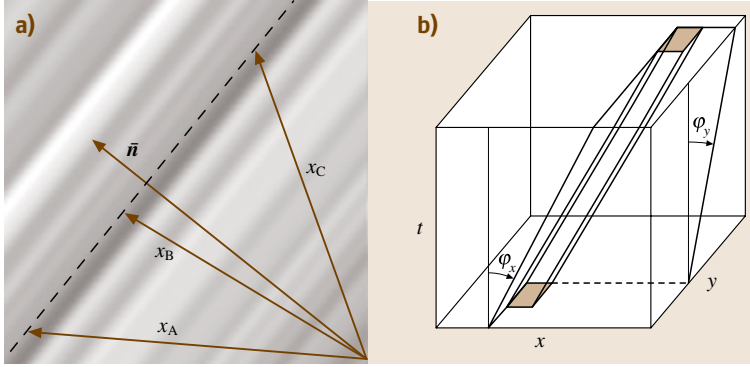


Fig. 25.17 (a) A simple neighborhood in 2-D. The grey values depend on one coordinate in the direction of the unit vector \hat{n} only. (b) Space–time diagram with two spatial components and a temporal component (after [25.62])

Let $\mathbf{r} = (r_1, r_2, r_3)^T$ be the vector pointing into the direction of constant brightness within the three-dimensional $\mathbf{x}-t$ domain. Once estimated \mathbf{r} one obtains for the optical flow:

$$\mathbf{f} = (f_1, f_2)^T = \frac{1}{r_3}(r_1, r_2)^T \quad (25.153)$$

Here \mathbf{r} points orthogonally to the spatiotemporal gradient vector $\nabla_{xt}g = (g_x, g_y, g_t)^T$. Therefore the scalar product between \mathbf{r} and $\nabla_{xt}g$ has to vanish:

$$(g_x, g_y, g_t) \cdot (r_1, r_2, r_3)^T = r_3[(\nabla g)^T \mathbf{f} + g_t] = 0.$$

We arrive at the well-known **BCCE** equation (25.145). Instead of the approach of [25.58], where the **BCCE** is minimized in a spatial neighborhood here one minimizes $\nabla_{xt}g \cdot \mathbf{r}$ in a *spatiotemporal* neighborhood U , which is characterized by the window function $w(\mathbf{x} - \mathbf{x}', t - t')$.

$$\hat{\mathbf{r}} = \arg \min_{\mathbf{r}} \langle [\nabla_{xt}g \cdot \mathbf{r}]^2 \rangle$$

using the abbreviation

$$\langle a \rangle = \int_{-\infty}^{\infty} w(\mathbf{x} - \mathbf{x}', t - t') a \, d\mathbf{x}' dt'.$$

Under the assumption of constant \mathbf{r} (that is, constant \mathbf{f}) within U , we the minimization problem can be reformulated as

$$\hat{\mathbf{r}} = \arg \min_{\mathbf{r}} [\mathbf{r}^T \langle \nabla_{xt}g \cdot \nabla_{xt}g^T \rangle \mathbf{r}] \quad (25.154)$$

$$= \arg \min_{\mathbf{r}} \mathbf{r}^T \mathbf{J} \mathbf{r}, \quad (25.155)$$

where \mathbf{J} with its components $J_{pq} = \langle g_p g_q \rangle$ is the three-dimensional symmetric *structure tensor*, and g_p , $p \in \{x, y, t\}$, denotes the partial derivative along the coordinate p .

The structure tensor can be transformed into diagonal shape by means of rotation. Thus the principal

axes of the structure tensor can be found by solving the eigenvalue problem

$$\mathbf{J} \mathbf{r} = \lambda \mathbf{r}.$$

The eigenvector to the corresponding *minimal* eigenvalue denotes the direction of constant brightness in the $\mathbf{x}-t$ domain, from which the optical flow can be calculated according to (25.153). From the rank of the structure tensor the type of motion can be deduced: Constant brightness (the *black wall problem*, $\text{rank}(\mathbf{J}) = 0$), spatial orientation and constant motion (the *aperture problem*, $\text{rank}(\mathbf{J}) = 1$), distributed spatial structures and constant motion ($\text{rank}(\mathbf{J}) = 2$), and distributed spatial structures but no coherent motion ($\text{rank}(\mathbf{J}) = 3$). The structure tensor technique is not only able to give an estimate for the optical flow, but is also able to present a confidence measure which assesses the quality of the estimate. A detailed analysis of the structure tensor technique and its practical application to optical flow computation can be found in [25.29]. The structure tensor technique can be formulated as a solution of the total least-squares (TLS) problem in a more-general way [25.91].

Implementation of the Structure Tensor. The expression for the structure tensor in (25.154) can be written explicitly as:

$$\mathbf{J} = \begin{pmatrix} \langle g_x g_x \rangle & \langle g_x g_y \rangle & \langle g_x g_t \rangle \\ \langle g_x g_y \rangle & \langle g_y g_y \rangle & \langle g_y g_t \rangle \\ \langle g_x g_t \rangle & \langle g_y g_t \rangle & \langle g_t g_t \rangle \end{pmatrix},$$

using the abbreviation

$$\langle a \rangle = \int_{-\infty}^{\infty} w(\mathbf{x} - \mathbf{x}', t - t') a \, d\mathbf{x}' dt'.$$

The implementation of the structure tensor can be carried out very efficiently by standard image processing operations. Identifying the convolution with a smoothing operation (for example with the isotropic binomial operator \mathcal{B}), and the derivatives in the p -th respective q -th direction with edge detectors (for example the optimized *Sobel filters* \mathcal{D}_p and \mathcal{D}_q [25.92]), we can construct a structure tensor operator

$$\mathcal{J}_{pq} = \mathcal{B}(\mathcal{D}_p \cdot \mathcal{D}_q),$$

where the dot signals a pixel-wise multiplication.

Because smoothing in general comes along with a loss of information, the result after applying the structure tensor operator can be stored in a more compact sequence than the original data. In practise this can be handled by downsampling the resulting sequence by a factor of two, for instance. In order to find a procedure to perform the principal axis transformations efficiently, we bear in mind that we are dealing with very small and symmetric matrices. These types of data can be covered by using the numerical method of Jacobi transformations to find the eigenvalues and eigenvectors [25.93].

Improvements of Optical Flow Determination

In this section improvements of determining optical flow will be presented. These improvements will help to solve the difficulties mentioned in the introduction. Each improvement can be applied to more than one (but not necessarily to all) methods of determining optical flow.

Parameterization of 2-D Optical-Flow Fields. As mentioned in the introduction to this chapter the standard local optical-flow methods assume that the optical flow $\mathbf{f}(\mathbf{x}, t)$ is constant within the local neighborhood U around \mathbf{x} . However, the optical flow may be expanded to a first order Taylor series in the vicinity of (\mathbf{x}_0, t_0) , in the way as in Sect. 25.2.1 (25.130). Then one is capable to estimate implicitly spatial gradients of the flow fields, for instance.

The BCCE supplemented by this parameterization yields the *extended brightness change constraint equation* (EBCCE):

$$(\nabla g)^T(t + \mathbf{A}\mathbf{x} + \mathbf{a}t) + g_t = 0. \quad (25.156)$$

Coarse-to-Fine Techniques. The temporal sampling theorem, already noted in Sect. 25.2.1, states a theoretical upper limit for the magnitude of displacements that are able to be analyzed. Apparently the maximum determinable displacements are limited by the magnitude of

the highest spatial wavenumbers, which are contained in the image.

Coarse-to-fine techniques or hierarchical multigrid approaches (Sect. 25.2.1) help to estimate large motions. By smoothing the image the high frequency content can be eliminated from the image sequence, so that one is able to estimate a coarse motion field. Then the motion can be *undone* by transforming the image back by means of the estimated coarse motion field. Now the higher-frequency content can be used to estimate a finer motion field. Added to the previously coarse motion field the fine motion field provides a more accurate approximation to the real motion field.

This procedure can be improved in several ways:

- If an estimation on a coarse level is incorrect, the fine-level estimate has no chance of correcting the errors. To fix this, we must have knowledge of the error in the coarse-level estimates. This suggests working in a probabilistic framework. Indeed, a *state evolution equation* and a *measurement equation* can be proposed similar to Kalman filtering [25.80].
- For more-accurate computation additional filters can be introduced, which slice the bandwidth in smaller pieces than these given by a dyadic pyramid structure. This results in a combined multiresolution and multiscale approach [25.94].
- The motion can be distributed very irregularly over the image plane. In these situations a selective multiresolution approach is suggested [25.95].

Robust Estimation. There are situations in which even parameterized flow field models fail to determine the optical flow correctly; these include the presence of multiple motions, such as motion discontinuities at boundaries (occluded multiple motions), or different motions being overlaid (transparent multiple motions) but also the presence of reflexes or corrupted pixels.

Least squares estimation tries to minimize a quadratic objective function $\rho(\mathbf{x})$:

$$\hat{\mathbf{f}} = \arg \min_{\mathbf{f}} \int_{-\infty}^{\infty} w(\mathbf{x} - \mathbf{x}') \rho[(\nabla g)^T \mathbf{f} + g_t] d\mathbf{x}'$$

with $\rho(x) = x^2$. The influence function $\psi(x)$ of the objective function is defined as the derivative of ρ with respect to x :

$$\psi(x) = \frac{\partial \rho(x)}{\partial x}.$$

In the least-squares case the influence of data points increases linearly and without bound, so that outliers

that do not fit to the model such as corrupted pixels have a great influence and distort the estimation of the correct optical flow dramatically. This is due to the fact that, by using a quadratic objective function, we inherently assumed that the residual errors are Gaussian and independently distributed within the neighborhood U .

To achieve a more-robust parameter estimation we have to replace the quadratic objective function by a suitable other function, which is referred to as an *M-estimator* in statistics. The influence function of an M-estimator has to be re-descending, i. e., it has to approach zero for large residuals after an initial increase for small values. [25.96] proposed a commonly used M-estimator (Fig. 25.18), which reads together with its influence function

$$\rho(x, \sigma) = \frac{x^2}{\sigma + x^2}, \quad \psi(x, \sigma) = \frac{2x\sigma}{(\sigma + x^2)^2}, \quad (25.157)$$

where σ is a scale parameter.

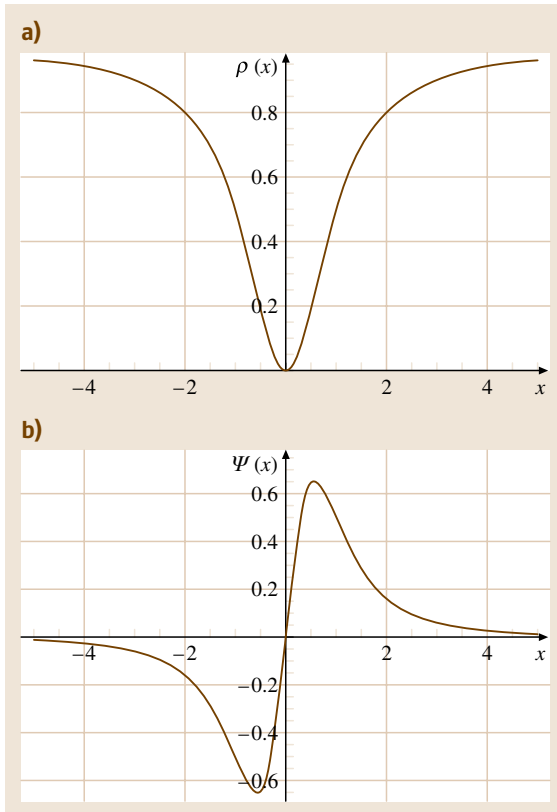


Fig. 25.18a,b An example for an M-estimator: (a) Geman and McClure norm (25.157). (b) Its derivative (after [25.29])

Given a robust formulation, there are numerous optimization techniques that can be employed to recover the motion estimates, and the most appropriate technique will depend on the particular formulation and choice of the ρ function. For detailed information about robust estimation the reader is referred to [25.97].

Dealing with Brightness Changes. In many situations the constraint of brightness constancy (25.145) is violated. In some cases we are able to find a physical model for the time-dependent brightness variation. So one can estimate both the correct optical flow field \mathbf{f} and the parameters \mathbf{a} of the underlying physical process. The approach of [25.98] constitutes an extension of the brightness change constraint equation to parameterized models of brightness variation, provided that these models are linear in \mathbf{a} or can be linearized by a Taylor series expansion.

In Sect. 25.2.5 we have stated that for the case that the brightness $g(\mathbf{x}, t)$ is constant along a path $\mathbf{x}(t)$ for all times, we are able to derive a constraint on the optical-flow. Now we allow, that the brightness along the path may change according to a time-dependent parameterized function $h(g_0, t, \mathbf{a})$:

$$g(\mathbf{x}(t), t) = h(g_0, t, \mathbf{a}), \quad (25.158)$$

where $g_0 = g(\mathbf{x}(t_0), t_0)$ denotes the image at time t_0 , and \mathbf{a} is the parameter vector for the brightness change model. The total derivative on both sides of (25.158) yields the *generalized brightness change constraint equation* (GBCCE),

$$(\nabla g)^T \mathbf{f} + g_t = \frac{d}{dt} h(g_0, t, \mathbf{a}),$$

which reduces to the well-known BCCE if h is constant.

In the following some models of brightness variation are presented:

Linear source terms. When sources are present, the brightness depends linearly on time: $h = qt$, where q denotes the source strength. The GBCCE becomes

$$(\nabla g)^T \mathbf{f} + g_t = q.$$

Exponential decay. In relaxation processes the time-dependent brightness can be modeled by an exponential decay: $h = g_0 \exp(-\kappa t)$, where κ denotes the relaxation constant. By differentiating h with respect to t the exponential function reproduces itself, so that we can write for the GBCCE:

$$(\nabla g)^T \mathbf{f} + g_t = -\kappa g.$$

Diffusion process. Fick's second law, which states that for isotropic diffusion the rate of change of the gray value is proportional to its Laplacian, tells us what the **GBCCE** looks like in this case, where the diffusion constant D is the proportional constant:

$$(\nabla g)^T f + g_t = D \nabla^2 g,$$

One can combine these physical models of brightness variation with various differential or tensor-based techniques: One has to replace the **BCCE** by the **GBCCE**, and the minimization has to be carried out over the optical flow f and over the parameters a simultaneously.

Literature Review

Each of the optical-flow methods (differential, frequency-based, or tensor-based approaches) presented in Sect. 25.2.5 was adapted to evaluation in the field of fluid mechanics. The term *optical flow* suggests its application to image sequences dealing with continuous tracer, such as heat or concentration. Indeed, most of the literature addresses continuous tracer. On the other hand, in practice fluid flow analysis is to a great extent particle based, which does not prevent optical-flow-based methods from being applied as well. Examples can be found in the literature.

Differential Techniques. *Ruhnau et al.* [25.94] applied the method of *Horn et al.* [25.85] (25.149) to images recorded with the conventional **PIV** technique. They used a coarse-to-fine strategy. The same authors replaced the global smoothness constraint by a regularizing coherence constraint, relying on the Stokes equation [25.99] and on the vorticity transport equation [25.100].

Instead of applying the **BCCE**, other authors used the 2-D continuity equation (25.147) as model. This was justified by the special kind of recording technique (such as transmittance imagery [25.88]) or data (such as satellite imagery [25.89]). The latter used a second-order div-curl regularization scheme instead of just assuming global smoothness. Moreover, they applied their scheme to **PIV** sequences [25.101].

Cohen et al. [25.95] applied a global method using a nonquadratic regularization technique to atmospheric and oceanographic image sequences. They have shown, that using an appropriate tessellation of the image according to an estimate of the motion field can improve optical flow accuracy and yields more reliable flows. This method defines a nonuniform multiresolution approach for coarse-to-fine grid generation.

Frequency-Based Techniques. *Larsen* [25.90] applied the local energy distribution to satellite images using the frequency-based techniques presented in Sect. 25.2.5. They used the optical-flow estimates together with confidence measures as an input for a regularization method based on the Markovian random-field approach.

Tensor-Based Techniques. *Garbe et al.* [25.102] estimated both velocity vector field and heat flux simultaneously in image sequences, recorded using infrared thermography. They expanded the structure tensor technique by a model including brightness changes.

Jehle and Jähne [25.103] estimated the wall shear stress in a medical engineering application directly, without previous computation of the velocity vector fields. The wall shear rate emerges as some components of the velocity gradient tensor, which is a **3-D** generalization of the matrix **A**.

References

- 25.1 M. Unser, A. Aldroubi, M. Eden: Fast B-spline transforms for continuous image representation and interpolation, *IEEE Trans. PAMI* **13**, 277–285 (1991)
- 25.2 W.T. Freeman, E.H. Adelson: The design and use of steerable filters, *IEEE Trans. PAMI* **13**, 891–906 (1991)
- 25.3 J. Weickert: *Anisotropic Diffusion in Image Processing* (Teubner, Stuttgart 1998)
- 25.4 P. Perona, J. Malik: Scale-space and edge detection using anisotropic diffusion, *IEEE Trans. PAMI* **12**, 629–639 (1990)
- 25.5 B. Jähne: *Digital Image Processing*, 6th edn. (Springer, Heidelberg 2005)
- 25.6 B. Jähne, H. Scharr, S. Körgel: Principles of filter design. In: *Computer Vision and Applications*, Signal Processing and Pattern Recognition, Vol. 2, ed. by B. Jähne, H. Haußecker, P. Geißler (Academic, San Diego 1999) pp. 125–151
- 25.7 J. Bigün, G.H. Granlund: Optimal orientation detection of linear symmetry, *ICCV'87* (IEEE, Washington 1987) 433–438
- 25.8 G.H. Granlund: In search of a general picture processing operator, *Comput. Graph. Imag. Process.* **8**, 155–173 (1978)
- 25.9 M. Felsberg, G.H. Granlund: POI detection using channel clustering and the 2D energy tensor, *Pattern Recognition: 26th DAGM Symposium*, Tübingen, Germany, LNCS, Vol. 3175 (Springer, Berlin 2004) 103–110

- 25.10 V.K. Madiseti, D.B. Williams: *The Digital Signal Processing Handbook* (CRC, Boca Raton 1998)
- 25.11 B. Jähne: *Handbook of Digital Image Processing for Scientific and Technical Applications*, 2nd edn. (CRC, Boca Raton 2004)
- 25.12 D.J. Fleet: *Measurement of Image Velocity* (Dissertation University of Toronto, Canada 1990)
- 25.13 M. Felsberg, G. Sommer: A new extension of linear signal processing for estimating local properties and detecting features. In: *Mustererkennung 2000*, 22. DAGM Symposium, Kiel, Informatik aktuell, ed. by G. Sommer, N. Krüger, C. Perwass (Springer, Berlin 2000) pp.195–202
- 25.14 C.K. Chui (Ed.): *Wavelets: A Tutorial in Theory and Applications* (Academic, Boston 1992)
- 25.15 T. Acharya, P.-S. Tsai: *JPEG2000 Standard for Image Compression* (Wiley, New York 2005)
- 25.16 C.E. Willert, M. Gharib: Digital particle image velocimetry, *Exp. Fluids* **10**, 181–193 (1991)
- 25.17 J. Westerweel: Fundamentals of digital particle image velocimetry, *Meas. Sci. Technol.* **8**, 1379–1392 (1997)
- 25.18 A.M. Fincham, G.R. Spedding: Low cost, high resolution DPIV for measurement of turbulent fluid flow, *Exp. Fluids* **23**, 449–462 (1997)
- 25.19 P.T. Tokumaru, P.E. Dimotakis: Image correlation velocimetry, *Exp. Fluids* **19**, 1–15 (1995)
- 25.20 A.W. Gruen: Adaptive least squares correlation: a powerful image matching technique, *S. Afr. J. Photogramm. Remote Sensing Cartogr.* **14**(3), 175–187 (1985)
- 25.21 F. Scarano, M.L. Riethmüller: Advances in iterative multigrid PIV image processing, *Exp. Fluids* **29**, S51–S60 (2000)
- 25.22 E. Cowen, S. Monismith: A hybrid digital particle tracking velocimetry technique, *Exp. Fluids* **22**, 199–211 (1997)
- 25.23 R.J.M. Bastiaans, G.A.J. van der Plas, R.N. Kieft: The performance of a new PTV algorithm applied in super-resolution PIV, *Exp. Fluids* **32**, 346–356 (2002)
- 25.24 S.J. Baek, S.J. Lee: A new two-frame particle tracking algorithm using match probability, *Exp. Fluids* **22**, 23–32 (1996)
- 25.25 K. Ohmi, H.-Y. Li: Particle tracking velocimetry with new algorithms, *Meas. Sci. Technol.* **11**(6), 603–616 (2000)
- 25.26 Y.A. Hassan, R.E. Canaan: Full-field bubbly flow velocity measurements using a multiframe particle tracking technique, *Exp. Fluids* **12**, 49–60 (1991)
- 25.27 N.A. Malik, T. Dracos, D. Papantoniou: Particle tracking velocimetry in three-dimensional flows, *Exp. Fluids* **15**, 279–294 (1993), Part II: Particle tracking
- 25.28 K. Takehara, R.J. Adrian, G.T. Etoh, K.T. Christensen: A Kalman tracker for super-resolution PIV, *Exp. Fluids* **29**, S34–S41 (2000)
- 25.29 B. Jähne, H. Haussecker, P. Geissler: *Handbook of Computer Vision and Applications* (Academic, San Diego 1999)
- 25.30 M. Raffel, C. Willert, J. Kompenhans: *Particle Image Velocimetry: A Practice Guide* (Springer, Heidelberg 1998)
- 25.31 G.R. Spedding, E.J.M. Rignot: Performance analysis and application of grid interpolation techniques for fluid flows, *Exp. Fluids* **15**, 417–430 (1993)
- 25.32 A.K. Prasad: Stereoscopic particle images velocimetry, *Exp. Fluids* **29**, 103–116 (2000)
- 25.33 R. Hartley, A. Zisserman: *Multiple View Geometry in Computer Vision* (Cambridge University Press, Cambridge 2000)
- 25.34 C.S. Slama: *Manual of Photogrammetry*, 4th edn. (American Society of Photogrammetry, Falls Church 1980)
- 25.35 K.D. Hinsch: Holographic particle image velocimetry, *Meas. Sci. Technol.* **13**, R61–R72 (2002)
- 25.36 T. Dracos: *Three-Dimensional Velocity and Vorticity Measuring and Image Analysis Techniques* (Kluwer Academic, Dordrecht 1996)
- 25.37 S.P. McKenna, W.R. McGillis: Performance of digital image velocimetry processing techniques, *Exp. Fluids* **32**, 2 (2002)
- 25.38 D.P. Hart: Super-Resolution PIV by Recursive Local-Correlation, *J. Visual.* **3**(2), 187–194 (2000)
- 25.39 H.J. Lin, M. Perlin: Improved methods for thin, surface boundary layer investigations, *Exp. Fluids* **25**, 431–444 (1998)
- 25.40 H.T. Huang, H.F. Fielder, J.J. Wang: Limitation and improvement of PIV, *Exp. Fluids* **15**, 168–174 (1993), Part I: Limitation of conventional techniques due to deformation of particle image patterns
- 25.41 H.T. Huang, H.F. Fielder, J.J. Wang: Limitation and improvement of PIV, *Exp. Fluids* **15**, 263–273 (1993), Part II: Particle image distortion, a novel technique
- 25.42 D.P. Hart: PIV error correction, *Exp. Fluids* **29**, 13–22 (2000)
- 25.43 B. Wienecke: Stereo-PIV using self-calibration on particle images, 5th International Symposium on Particle Image Velocimetry (Busan, Korea 2003)
- 25.44 C.E. Willert, M. Gharib: Three-dimensional particle imaging with a single camera, *Exp. Fluids* **12**, 353–358 (1992)
- 25.45 F. Pereira, M. Gharib, D. Dabiri, M. Modarress: Defocusing PIV: a three component 3-D DPIV measurement technique, *Exp. Fluids* **29**, S78–S84 (2000), Application to bubbly flows
- 25.46 C. Kähler, J. Kompenhans: Fundamentals of multiple plane stereo particle image velocimetry, *Exp. Fluids* **29**, 70–77 (2000)
- 25.47 A. Liberzon, R. Gurka, G. Hetsroni: XPIV-Multiplane stereoscopic particle image velocimetry, *Exp. Fluids* **36**, 355–362 (2004)
- 25.48 A. Schimpf, S. Kallweit, J.B. Richon: Photogrammetric particle image velocimetry, 5th Int. Symp. on Particle Image Velocimetry (2003)

- 25.49 R.J. Adrian: Particle-imaging techniques for experimental fluid mechanics, *Annu. Rev. Fluid Mech.* **23**, 261–304 (1991)
- 25.50 R.D. Keane, R.J. Adrian: Optimization of particle image velocimeters, *Meas. Sci. Technol.* **1**, 1202–1215 (1990), Part I: Double pulsed systems
- 25.51 R.D. Keane, R.J. Adrian: Optimization of particle image velocimeters, *Meas. Sci. Technol.* **2**, 963–974 (1991), Part II: Multiple pulsed systems
- 25.52 R.D. Keane, R.J. Adrian: Theory of crosscorrelation analysis of PIV images, *Appl. Sci. Res.* **49**, 191–215 (1992)
- 25.53 J. Westerweel: *Digital Particle Image Velocimetry – Theory and Application* (Delft Univ. Press, Delft 1993)
- 25.54 L.C. Gui, W. Merzkirch: A method for tracking ensembles of particle images, *Exp. Fluids* **21**, 465–468 (1996)
- 25.55 C.Q. Davis, Z.Z. Karu, D.M. Freeman: Equivalence of subpixel motion estimators based on optical flow and block matching, *Int. Symposium on Computer Vision* (Coral Gables, Florida 1995)
- 25.56 L.C. Gui, W. Merzkirch: A comparative study of the MQD method and several correlation-based PIV evaluation algorithms, *Exp. Fluids* **28**, 36–44 (2000)
- 25.57 J. Shi, C. Tomasi: Good features to track, *Computer Vision Pattern Recognition* (1994)
- 25.58 B.D. Lucas, T. Kanade: An iterative image registration technique with an application to stereo vision, *Imaging Understanding Workshop* (1981) 121–130
- 25.59 D. Papantoniou, T. Dracos: Analyzing 3-D turbulent motions in open channel flow by use of stereoscopy and particle tracking, *Adv. Turb.* **2**, 278–285 (1989)
- 25.60 M.P. Wernet, A. Pline: Particle displacement tracking technique and Cramer–Rao lower bound error in centroid estimates from CCD imagery, *Exp. Fluids* **15**, 295–307 (1993)
- 25.61 C.J. Veenman, M.J.T. Reinders, E. Backer: Establishing motion correspondence using extended temporal scope, *Artif. Intell.* **145**(1–2), 227–243 (2003)
- 25.62 B. Jähne: *Digital Image Processing*, 5th edn. (Springer, Heidelberg 2002)
- 25.63 Y.G. Geuzennec, N. Kiritzis: Statistical investigation of errors in particle image velocimetry, *Exp. Fluids* **10**, 138–146 (1990)
- 25.64 H.G. Maas, A. Gruen, D. Papantoniou: Particle tracking velocimetry in three-dimensional flows, *Exp. Fluids* **15**, 133–146 (1993), Part I: Photogrammetric determination of particle coordinates
- 25.65 Y.G. Geuzennec, R.S. Brodkey, N. Trigui, J.C. Kent: Algorithms for fully automated three-dimensional particle tracking velocimetry, *Exp. Fluids* **17**, 209–219 (1994)
- 25.66 N.A. Malik, T. Dracos: Interpolation schemes for three-dimensional velocity fields from scattered data using Taylor expansions, *J. Comput. Phys.* **119**, 231–243 (1995)
- 25.67 F. Hering, D. Wierzimok, C. Leue, B. Jähne: Particle tracking velocimetry beneath water waves, *Exp. Fluids* **23**(6), 472–482 (1997), Part I: Visualization and tracking algorithms
- 25.68 G. Nemhauser, L. Wolsey: *Integer and Combinatorial Optimization* (Wiley, New York 1999)
- 25.69 S.B. Dalziel: Decay of rotating turbulence: some particle tracking experiments, In: *Flow Visualization and Image Analysis*, ed. by F.T.M. Nieuwstadt (Kluwer Academic, Dordrecht 1993)
- 25.70 M. Stellmacher, K. Obermayer: A new particle tracking algorithm based on deterministic annealing and alternative distance measures, *Exp. Fluids* **28**, 506–518 (2000)
- 25.71 S. Gold, A. Rangarajan, C.-P. Lu, S. Pappu: New algorithms for 2D and 3D point matching: pose estimation and correspondence, *Pattern Recog.* **31**, 1019–1031 (1998)
- 25.72 E. Brookner: *Tracking and Kalman Filtering made easy* (Wiley, New York 1998)
- 25.73 S. Blackman, R. Popoli: *Design and Analysis of Modern Tracking Systems* (Artech House, Boston 1999)
- 25.74 I.J. Cox: A review of statistical data association techniques for motion correspondence, *Int. J. Comput. Vis.* **10**(1), 53–66 (1993)
- 25.75 L.D. Stone, C.A. Barlow, T.L. Corwin: *Bayesian Multiple Target Tracking* (Artech House, Boston 1999)
- 25.76 G. Welch, G. Bishop: *An Introduction to the Kalman Filter*, Tech. Rep. TR 95-041 (Univ. North Carolina, Chapel Hill 2001)
- 25.77 I.J. Cox, S.L. Hingorani: An efficient implementation of Reid’s multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking, *IEEE Trans. Pattern Anal.* **18**(2), 138–150 (1996)
- 25.78 R.D. Keane, R.J. Adrian, Y. Zhang: Superresolution particle imaging velocimetry, *Meas. Sci. Technol.* **6**, 754–768 (1995)
- 25.79 J. Willneff, B. Lüthi: Particle tracking velocimetry measurements for lagrangian analysis of turbulent flows, 6th Conference on Optical 3-D Measurement Techniques, Vol. 2 (Zurich 2003) 191–198
- 25.80 E.P. Simoncelli: *Distributed representation and analysis of visual motion*, Ph.D. Thesis (MIT, Cambridge 1993)
- 25.81 O. Faugeras: *Three Dimensional Computer Vision: A Geometric Viewpoint* (MIT Press, Cambridge 1993)
- 25.82 B.F. Murray, D.W. Buxton: *Experiments in the Machine Interpretation of Visual Motion* (MIT Press, Cambridge 1990)
- 25.83 Z. Zhang, O. Faugeras: *3D Dynamic Scene Analysis*, 27 Springer Inform. Sci. (Springer, Heidelberg 1992)
- 25.84 W.J. Cook, W.H. Cunningham, W.R. Pulleyblank, A. Schrijver: *Combinatorial Optimization* (Wiley, New York 1998)

- 25.85 B.K.P. Horn, B.G. Schunk: Determining optical flow, *Artif. Intell.* **17**, 185–204 (1981)
- 25.86 J.L. Barron, D.J. Fleet, S.S. Beauchemin: Performance of optical flow techniques, *Int. J. Comput. Vis.* **12**(1), 43–77 (1994)
- 25.87 G.H. Granlund, H. Knutsson: *Signal Processing for Computer Vision* (Kluwer, Dordrecht 1995)
- 25.88 R. Wildes, M. Amabile, A.–M. Lanzileto, T.–S. Leu: Recovering estimates of fluid flows from image sequence data, *Comput. Vis. Image Underst.* **80**, 246–266 (2000)
- 25.89 T. Corpetti, E. Memin, P. Perez: Dense estimation of fluid flows, *IEEE Trans. Pattern Anal. Machine Intell.* **24**(3), 365–380 (2002)
- 25.90 R. Larsen: Estimation of dense image flow fields in fluids, *IEEE T. Geosci. Remote Sens.* **36**(1), 256–264 (1998)
- 25.91 S. van Huffel, J. Vandewalle: *The Total Least Squares Problem: Computational Aspects and Analysis* (SIAM, Philadelphia 1991)
- 25.92 H. Schar: *Optimal Operators in Digital Image Processing*, Ph.D. Thesis (University of Heidelberg, Heidelberg 2000)
- 25.93 W.H. Press, S.A. Teukolsky, W. Vetterling, B. Flannery: *Numerical Recipes in C: The Art of Scientific Computing* (Cambridge Univ. Press, New York 1992)
- 25.94 P. Ruhnau, T. Kohlberger, C. Schnörr, H. Nobach: Variational optical flow estimation for particle image velocimetry, *Exp. Fluids* **38**, 21–32 (2005)
- 25.95 I. Cohen, I. Herlin: Non uniform multiresolution method for optical flow and phase portrait models: environmental applications, *Int. Comput. Vis.* **33**(1), 24–49 (1999)
- 25.96 S. Geman, D.E. McClure: Bayesian image analysis: An application to single photon emission tomography, *Am. Statist. Assoc. Statist. Comput. Sect.* (1984) 12–18
- 25.97 M.J. Black, P. Anandan: The robust estimation of multiple motions: parametric and piecewise-smooth flow fields, *Comput. Vis. Image Understand.* **63**, 75–104 (1996)
- 25.98 H.W. Haussecker, D.J. Fleet: Computing optical flow with physical models of brightness variation, *IEEE Trans. Pattern Anal.* **23**(6), 661–673 (2001)
- 25.99 P. Ruhnau, C. Schnörr: Optical Stokes flow: an image based control approach, *Exp. Fluids* **42**, 61–78 (2007)
- 25.100 P. Ruhnau, A. Stahl, C. Schnörr: On-line variational estimation of dynamical fluid flows with physics-based spatio-temporal regularization, 26th DAGM (2006), *Pattern Recognition*
- 25.101 T. Corpetti, D. Heitz, G. Arroyo, E. Memin, A. Santa-Cruz: Fluid experimental flow estimation based on an optical-flow scheme, *Exp. Fluids* **40**(1), 80–97 (2005)
- 25.102 C. Garbe, H. Spies, B. Jähne: Estimation of surface flow and net heat flux from infrared image sequences, *J. Math. Imag. Vis.* **19**, 159–174 (2003)
- 25.103 M. Jehle, B. Jähne: Direct estimation of the wall shear rate using parametric motion models in 3D, *Lect. Notes Comput. Sci.* **174**, 434–443 (2006)