UNIT II IMAGE PRE-PROCESSING

Local pre-processing - Image smoothing - Edge detectors - Zero-crossings of the second derivative - Scale in image processing - Canny edge detection - Parametric edge models - Edges in multispectral images - Local pre-processing in the frequency domain - Line detection by local preprocessing operators - Image restoration.

2.1. Local pre-processing:

Local pre-processing methods are divided into two groups according to the goal of the processing.

1) **Smoothing** aims to suppress noise or other small fluctuations in the image; it is equivalent to the suppression of high frequencies in the Fourier transform domain. Unfortunately, smoothing also blurs all sharp edges that bear important information about the image.

2) **Gradient operators** are based on local derivatives of the image function. Derivatives are bigger at locations of the image where the image function undergoes rapid changes, and the aim of gradient operators is to indicate such locations in the image.

   Gradient operators have a similar effect to suppressing low frequencies in the Fourier transform domain.

Noise is often high frequency in nature; if a gradient operator is applied to an image, the noise level increases simultaneously. Clearly, smoothing and gradient operators have conflicting aims. Some pre-processing algorithms solve this problem and permit smoothing and edge enhancement simultaneously.

Another classification of local pre-processing methods is according to the transformation properties; **linear** and **non-linear** transformations can be distinguished. Linear operations calculate the resulting value in the output image pixel f (i, j) as a linear combination of brightnesses in a local neighborhood O of the pixel g(i, j) in the input image. The contribution of the pixels in the neighborhood O is weighted by coefficients h:

$$f(i,j) = \sum_{(m,n)\in\mathcal{O}} \sum h(i-m, j-n)\, g(m,n)\,. \qquad (5.23)$$

Equation (5.23) is equivalent to discrete convolution with the kernel h, which is called a **convolution mask**. Rectangular neighborhoods are often used with an odd number of pixels in rows and columns, enabling specification of the central pixel of the neighbor- hood.

The choice of the local transformation, size, and shape of the neighborhood depends strongly on the size of objects in the processed image. If objects are rather large, an image can be enhanced by smoothing of small degradations.

2.2 Image smoothing:

Image smoothing uses redundancy in image data to suppress noise, usually by some form of averaging of brightness values in some neighborhood . Smoothing poses the problem of blurring sharp edges, and so we shall consider smoothing methods which are **edge preserving** here, the average is computed only from points in the neighborhood which have similar properties to the point being processed.

Local image smoothing can effectively eliminate impulse noise or degradations appearing as thin stripes, but does not work if degradations are large blobs or thick stripes. Such problems may be addressed by image restoration techniques.

Averaging, statistical principles of noise suppression

Assume that the noise value ν at each pixel is an independent random variable with zero mean and standard deviation σ. We might capture the same static scene under the same conditions n times. From each captured image a particular pixel value gi, i = 1, . . . , n is selected. An estimate of the correct value can be obtained as an average of these values, with corresponding noise values ν1, . . . , νn

$$\frac{g_1 + \ldots + g_n}{n} + \frac{\nu_1 + \ldots + \nu_n}{n} .$$

The second term here describes the noise, which is again a random value with zero mean and standard deviation $\sigma/\sqrt{n}$. Thus, if n images of the same scene are available, smoothing can be accomplished without blurring the image by

$$f(i,j) = \frac{1}{n} \sum_{k=1}^{n} g_k(i,j) .$$

This reasoning is a well-known statistical result: a random sample is taken from a population and the corresponding sample mean value is calculated. If random samples are repeatedly selected and their sample mean values calculated, we would obtain a distribution of sample mean values. This distribution of sample means has some useful properties:

- The mean value of the distribution of sample mean values is equal to the population mean.
- The distribution of sample mean values has variance $\sigma/\sqrt{n}$, which is clearly smaller than that of than the original population.
- If the original distribution is normal (Gaussian) then the distribution of sample mean values is also normal. Better, the distribution of sample means

converges to normal whatever the original distribution. This is the **central limit theorem**.

- From the practical point of view, it is important that not too many random se- lections have to be made. The central limit theorem tell us the distribution of sample mean values without the need to create them. In statistics, usually about

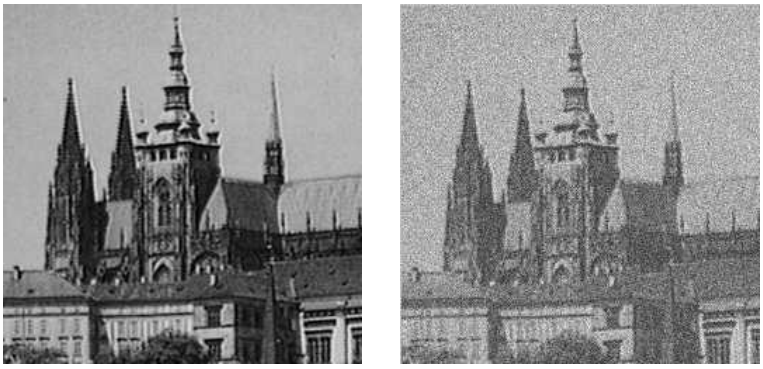
30 samples are considered the lowest limit of the necessary number of observations. Usually, only one noise corrupted is available, and averaging is then performed in a local neighborhood. Results are acceptable if the noise is smaller in size than the smallest objects of interest in the image, but blurring of edges is a serious disadvantage. Averaging is a special case of discrete convolution [equation (5.23)]. × For a 3x 3 neighborhood, the convolution mask h is

$$h = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} .$$

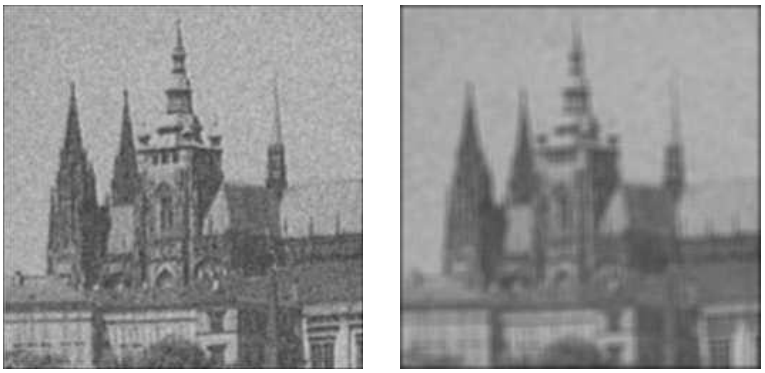
The significance of the pixel in the center of the convolution mask h or its 4neighbors is sometimes increased, as it better approximates the properties of noise with a Gaussian probability distribution.

$$h = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} , \qquad h = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

There are two commonly used smoothing filters whose coefficients gradually decrease to have near-zero values at the window edges. This is the best way to minimize spurious oscillations in the frequency spectrum. These are the Gaussian and the Butterworth filters. Larger convolution masks for averaging by Gaussian filter are created according to the Gaussian distribution formula (equation 5.47) and the mask coefficients are normalized to have a unit sum.



(a) (b)



(c) (d)

Figure 5.9: Noise with Gaussian distribution and averaging filters. (a) Original image. (b) Superimposed noise (random Gaussian noise characterized by zero mean and standard deviation equal to one-half of the gray-level standard deviation of the original image). (c) 3 × 3 averaging. (d) 7 × 7 averaging.

An example will illustrate the effect of this noise suppression (low resolution images, 256 256, were chosen deliberately to show the discrete nature of the process). Figure 5.9a shows an original image of Prague castle; Figure 5.9b shows the same image with superimposed additive noise with Gaussian distribution; Figure 5.9c shows the result of averaging with a 3x 3 convolution mask (equation 5.27)— noise is significantly reduced and the image is slightly blurred. Averaging with a larger mask (7 x 7) is demonstrated in Figure 5.9d, where the blurring is much more serious.

Such filters can be very computationally costly, but this is considerably reduced in the important special case of **separable filters**. Separability in 2D means that the convolution kernel can be factorized as a product of two one-dimensional vectors, and theory provides a clue as to which convolution masks are separable.

As an example, consider a binomic filter. Its elements are binomic numbers which are created as a sum of the corresponding two numbers in Pascal's triangle. Consider such a filter of size 5x 5—it can be decomposed into a product of two

1D vectors, h1, h2.

$$\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} = \begin{bmatrix} \\ h_1 \\ \\ \end{bmatrix} \begin{bmatrix} \quad h_2 \quad \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Suppose a convolution kernel is of size 2N + 1. Equation (5.23) allows the convolution to be Rewritten taking account of the special propertiesof separability

$$g(x,y) = \sum_{m=-N}^{N} \sum_{n=-N}^{N} h(m,n)\, f(x+m, y+n) = \sum_{m=-N}^{N} h_1(m) \sum_{n=-N}^{N} h_2(n)\, f(x+m, y+n).$$

The direct calculation of the convolution according to equation (5.23) would need, in our case of 5 5 convolution kernel, 25 multiplications and 24 additions for each pixel. If the separable filter is used then only 10 multiplications and 8 additions suffice.

Averaging with limited data validity

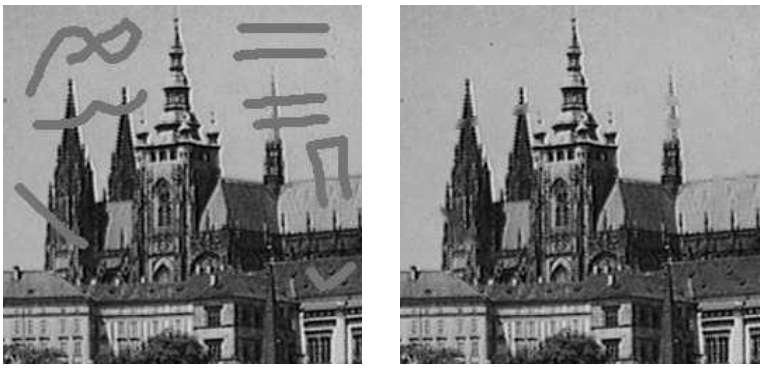
Methods that average with limited data validity try to avoid blurring by averaging only those pixels which satisfy some criterion, the aim being to prevent involving pixels that are part of a separate feature.

A very simple criterion is to define a brightness interval of invalid data [min, max] (typically corresponding to noise of known image faults), and apply image averaging only to pixels in that interval. For a point (m, n), the convolution mask is calculated in the neighborhood O by the non-linear formula

$$h(i,j) = \begin{cases} 1 & \text{for } g(m+i, n+j) \notin [\text{min,max}] \\ 0 & \text{otherwise} \end{cases},$$

where (i, j) specify the mask element. Therefore, only values of pixels with invalid gray-levels are replaced with an average of their neighborhoods, and only valid data contribute to the averages.

A second method performs averaging only if the computed brightness change of a pixel is in some pre-defined interval; this permits repair to large-area errors resulting from slowly changing brightness of the background without affecting the rest of the image. A third method uses edge strength (i.e., gradient magnitude) as a criterion. The magnitude of some gradient operator is first computed for the entire image, and only pixels with a small gradient are used in averaging. This method effectively rejects averaging at edges and therefore suppresses blurring, but setting of the threshold is laborious.



(a) (b)

Figure 5.10: Averaging with limited data validity. (a) Original corrupted image. (b) Result of corruption removal.

Averaging according to inverse gradient

Within a convolution mask of odd size, the inverse gradient δ of a point (i, j) with respect to the central pixel (m, n) is defined as

$$\delta(i,j) = \frac{1}{|g(m,n) - g(i,j)|} \,. \tag{5.29}$$

If g(m, n) = g(i, j), then we define δ(i, j) = 2, so δ is in the interval (0, 2], and is smaller at the edge than in the interior of a homogeneous region. Weight coefficients in the convolution mask h are normalized by the inverse gradient, and the whole term is multiplied by 0.5 to keep brightness values in the original range: the mask coefficient corresponding to the central pixel is defined as h(i, j) = 0.5. The constant 0.5 has the effect of assigning half the weight to the central pixel (m, n), and the other half to its neighborhood

$$h(i,j) = 0.5 \frac{\delta(i,j)}{\sum_{(m,n)\in\mathcal{O}} \delta(i,j)} . \qquad (5.30)$$

This method assumes sharp edges. When the convolution mask is close to an edge, pixels from the region have larger coefficients than pixels near the edge, and it is not blurred. Isolated noise points within homogeneous regions have small values of the inverse gradient; points from the neighborhood take part in averaging and the noise is removed.

Averaging using a rotating mask

The smoothing discussed thus far was linear, which has the disadvantage that edges in the image are inevitably blurred. Alternative non-linear methods exist which reduce this. The neighborhood of the current pixel is inspected and divided into two subsets by a homogeneity criterion of the user's choice. One set consists of all pixels neighboring the current pixel or any pixel already included in this set, which satisfy the homogeneity criterion. The second set is the complement. This selection operation is non-linear and causes the whole filter to be non-linear. Having selected the homogeneous subset containing the current pixel, the most probable value is sought in it by a linear or non- linear technique.

Averaging using a rotating mask is such a non-linear method that avoids edge blur- ring, and the resulting image is in fact sharpened. The brightness average is calculated only within this region; a brightness dispersion σ^2 is used as the region homogeneity measure. Let n be the number of pixels in a region R and g be the input image. Dispersion σ^2 is calculated as

$$\sigma^2 = \frac{1}{n} \sum_{(i,j)\in R} \left(g(i,j) - \frac{1}{n} \sum_{(i,j)\in R} g(i,j) \right)^2 . \qquad (5.31)$$

Having computed region homogeneity, we consider its shape and size. The eight possible 3x3 masks that cover a 5 x 5 neighborhood of a current pixel (marked by the small cross) are shown in Figure 5.11. The ninth mask is the 3 x 3 neighborhood of the current pixel itself. Other mask shapes—larger or smaller—can also be used.

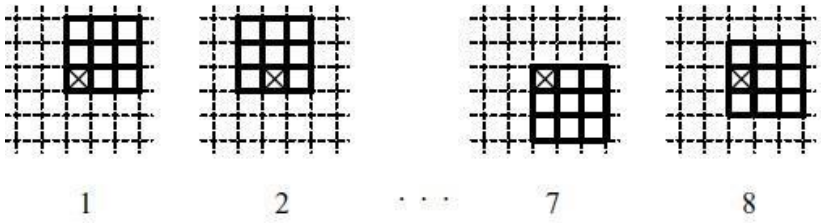


Figure 5.11: Eight possible rotated 3×3 masks.

Algorithm 5.2: Smoothing using a rotating mask

1. Consider each image pixel (i, j).
2. Calculate dispersion for all possible mask rotations about pixel (i, j) according to equation (5.31).
3. Choose the mask with minimum dispersion.
4. Assign to the pixel f (i, j) in the output image f the average brightness in the chosen mask.

Algorithm 5.2 can be used iteratively and the process converges quite quickly to a stable state. The size and shape of masks influence the convergence—the smaller the mask, the smaller are the changes and more iterations are needed. A larger mask suppresses noise faster and the sharpening effect is stronger. On the other hand, information about details smaller than the mask may be lost. The number of iterations is also influenced by the shape of regions in the image and noise properties.

Median filtering

In probability theory, the **median** divides the higher half of a probability distribution from the lower half. For a random variable x, the median M is the value for which the probability of the outcome $x < M$ is 0.5. The median of a finite list of real numbers is simply found by ordering the list and selecting the middle member. Lists are often constructed to be odd in length to secure uniqueness.

Median filtering is a non-linear smoothing method that reduces the blurring of edges, in which the idea is to replace the current point in the image by the median of the brightnesses in its neighborhood. The median in the neighborhood is not affected by individual noise spikes and so median smoothing eliminates impulse noise quite well. Further, as median filtering does not blur edges much, it can be applied iteratively. Clearly, performing a sort on pixels within a (possibly large) rectangular window at every pixel position may become very expensive. A more efficient approach [Huang et al., 1979; Pitas and Venetsanopoulos, 1990] is to notice that as the window moves across a row by one column, the only change to its

– contents is to lose the leftmost column and replace it with a new right column—for a median window of m rows and n columns, mn 2m pixels are unchanged and do not need re-sorting. The algorithm is as follows:

Algorithm 5.3: Efficient median filtering

1. Set
$$t = \frac{mn}{2} .$$
(We would always avoid unnecessary floating point operations: if m and n are both odd, round t.)

2. Position the window at the beginning of a new row, and sort its contents. Construct a histogram H of the window pixels, determine the median m, and record n_m, the number of pixels with intensity less than or equal to m.

3. For each pixel p in the leftmost column of intensity p_g, perform
$$H[p_g] = H[p_g] - 1 .$$
Further, if $p_g \leq m$, set
$$n_m = n_m - 1 .$$

4. Move the window one column right. For each pixel p in the rightmost column of intensity p_g, perform
$$H[p_g] = H[p_g] + 1 .$$
If $p_g \leq m$, set
$$n_m = n_m + 1 .$$

5. If $n_m = t$ then go to (8).

6. If $n_m > t$ then go to (7).
Repeat
$$m = m + 1 ,$$
$$n_m = n_m + H[m] ,$$
until $n_m \geq t$. Go to (8).

7. (We have $n_m > t$, if here). Repeat
$$n_m = n_m - H[m] , \; m = m - 1 ,$$
until $n_m \leq t$.

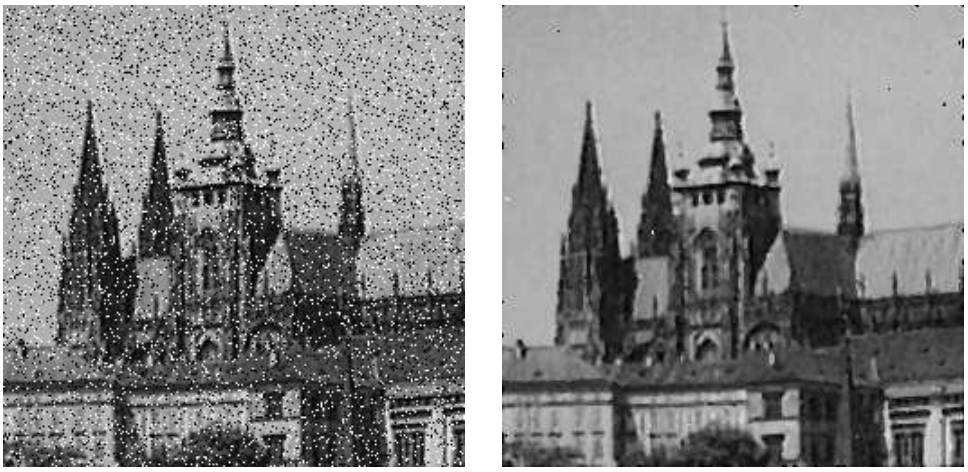
8. If the right-hand column of the window is not at the right-hand edge of the image, go to (3).

9. If the bottom row of the window is not at the bottom of the image, go to (2).

Median filtering is illustrated in Figure 5.12. The main disadvantage of median filtering in a rectangular neighborhood is its damaging of thin lines and sharp corners—this can be avoided if another shape of neighborhood is used. For instance, if horizontal/vertical lines need preserving, a neighborhood such as that

in Figure 5.13 can be used.

Median smoothing is a special instance of more general **rank filtering** techniques, the idea of which is to order pixels in some neighborhood into a sequence. The results of pre-processing are some statistics over this sequence, of which the median is one possibility. Another variant is the maximum or the minimum values of the sequence. This defines generalizations of dilation and erosion operators in images with more brightness values.



(a) (b)

Figure 5.12: Median filtering. (a) Image corrupted with impulse noise (14% of image area covered with bright and dark dots). (b) Result of 3x 3 median filtering.

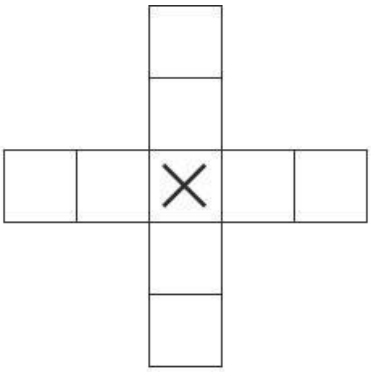


Figure 5.13: Horizontal/vertical line preserving neighbor- hood for median filtering

Non-linear mean filter

The non-linear mean filter is another generalization of averaging techniques; it is defined by

$$f(m,n) = u^{-1}\left(\frac{\sum_{(i,j)\in\mathcal{O}} a(i,j)\, u\big(g(i,j)\big)}{\sum_{(i,j)\in\mathcal{O}} a(i,j)}\right), \qquad (5.32)$$

where f (m, n) is the result of the filtering, g(i, j) is the pixel in the input image,O and is a local neighborhood of the current pixel (m, n). The function u of one variable has an inverse function u^{-1}; the a(i, j) are weight coefficients.

If the weights a(i, j) are constant, the filter is called **homomorphic**. Some

homomorphic filters used in image processing are:

- Arithmetic mean, u(g) = g .
- Harmonic mean, u(g) = 1/g .
- Geometric mean, u(g) = log g .

2.3. Edge detectors:

Edge detectors are a collection of very important local image pre-processing methods used to locate changes in the intensity function; edges are pixels where brightness changes abruptly.

- Edges are those places in an image that correspond to **object boundaries.**
- Edges are pixels where image brightness changes abruptly.

Neurological and psychophysical research suggests that locations in the image in which the function value changes abruptly are important for image perception. Edges are to a certain degree invariant to changes of illumination and viewpoint. If only edge elements with strong magnitude (edges) are considered, such information often suffices for image understanding. The positive effect of such a process is that it leads to significant reduction of image data. Nevertheless such data reduction does not undermine understanding the content of the image (interpretation) in many cases. Edge detection provides appropriate generalization of the image data; for instance, line drawings perform such a generalization.

We shall consider which physical phenomena in the image formation process lead to abrupt changes in image values—see Figure 5.15. Calculus describes changes of continuous functions using derivatives; an image function depends on two variables—co-ordinates in the image plane—and so operators describing edges are expressed using partial derivatives. A change of the image function can be described by a gradient that points in the direction of the largest growth of the image function.

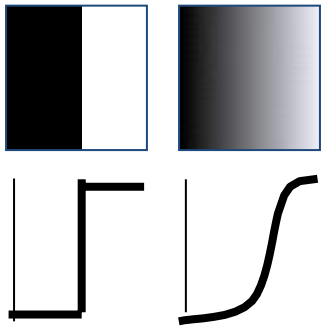


Figure 5.15: Origin of edges, i.e., physical phenomena in image formation process which lead to edges in images. At right, a Canny edge detection.

An edge is a property attached to an individual pixel and is calculated from the image function behavior in a neighborhood of that pixel. It is a **vector variable** with two components, **magnitude** and **direction**. The edge magnitude is the magnitude of the gradient, and the edge direction φ is rotated with respect to −

the gradient direction ψ by 90◦. The gradient direction gives the direction of maximum growth of the function, e.g., from black f (i, j) = 0 to white f (i, j) = 255. This is illustrated in Figure 5.16, in which closed lines are lines of equal brightness.

The orientation 0◦ points east.

Edges are often used in image analysis for finding region boundaries. Provided that the region has homogeneous brightness, its boundary is at the pixels where the image function varies and so in the ideal case without noise consists of pixels with high edge

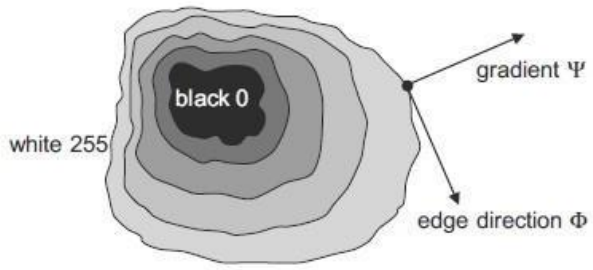


Figure 5.16: Gradient direction and edge direction.

magnitude. It can be seen that the boundary and its parts (edges) are perpendicular to the direction of the gradient.

Figure 5.17 shows examples of several standard edge profiles. Edge detectors are usually tuned for some type of edge profile.

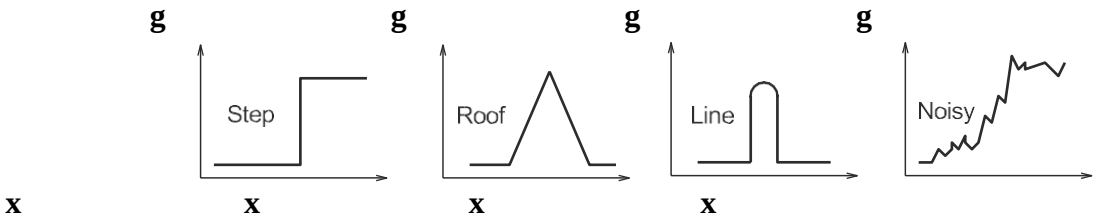


Figure 5.17: Typical edge profiles.

The gradient magnitude | grad g(x, y)| and gradient direction ψ are continuous image functions calculated as

$$|\operatorname{grad} g(x,y)| = \sqrt{\left(\frac{\partial g}{\partial x}\right)^2 + \left(\frac{\partial g}{\partial y}\right)^2},$$

$$\psi = \arg\left(\frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}\right),$$

where arg(x, y) is the angle (in radians) from the x axis to (x, y). Sometimes we are interested only in edge magnitudes without regard to their orientations— a linear differential operator called the **Laplacian** may then be used. The Laplacian has the same properties in all directions and is therefore invariant to rotation. It is defined as

$$\nabla^2 g(x,y) = \frac{\partial^2 g(x,y)}{\partial x^2} + \frac{\partial^2 g(x,y)}{\partial y^2}.$$

Image **sharpening** [Rosenfeld and Kak, 1982] has the objective of making edges steeper— the sharpened image is intended to be observed by a human. The sharpened output image f is obtained from the input image g as

$$f(i, j) = g(i, j) - C\, S(i, j), \qquad (5.36)$$

where C is a positive coefficient which gives the strength of sharpening and S(i, j) is a measure of the image function sheerness, calculated using a gradient operator. The Laplacian is very often used for this purpose. Figure 5.18 gives an example of image sharpening using a Laplacian.

Image sharpening can be interpreted in the frequency domain as well. We know that the result of the Fourier transform is a combination of harmonic functions.

The derivative of the harmonic function sin(nx) is n cos(nx); thus the higher the frequency, the higher the magnitude of its derivative.

A similar image sharpening technique to that of equation (5.36), called **unsharp masking**, is often used in printing industry applications. A signal proportional to an unsharp (e.g., heavily blurred by a smoothing operator) image is subtracted from the original image. A digital image is discrete in nature and so equations (5.33) and (5.34), containing derivatives, must be approximated by **differences**. The first differences of the image g in the vertical direction (for fixed i) and in the horizontal direction (for fixed j) are given by

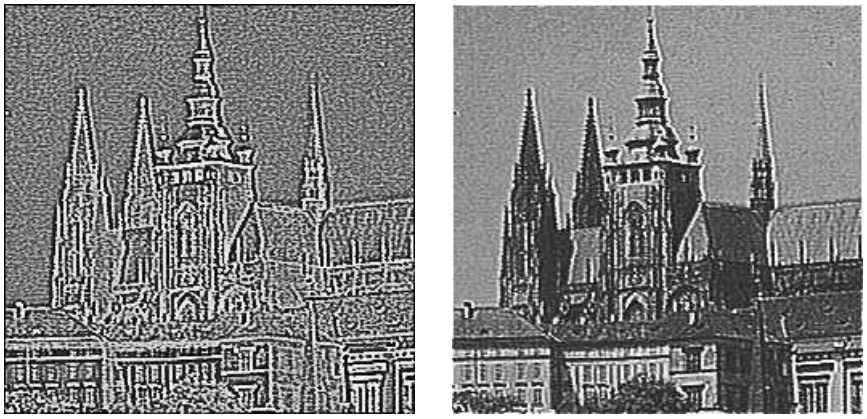
$$\Delta_i\ g(i, j) = g(i, j) - g(i - n, j) ,$$
$$\Delta_j\ g(i, j) = g(i, j) - g(i, j - n) ,$$

(5.37)

where n is a small integer, usually 1. The value n should be chosen small enough to provide a good approximation to the derivative, but large enough to neglect unimportant changes in the image function. Symmetric expressions for the differences,

$$\Delta_i\ g(i, j) = g(i + n, j) - g(i - n, j) ,$$
$$\Delta_j\ g(i, j) = g(i, j + n) - g(i, j - n) ,$$

are not usually used because they neglect the impact of the pixel (i, j) itself.



(a) (b)

Figure 5.18: Laplace gradient operator. (a) Laplace edge image using the 8-connectivity mask.
(b) Sharpening using the Laplace operator equation 5.36, C = 0.7 . Compare the sharpening effect with the original image in Figure 5.9a.

Gradient operators as a measure of edge sheerness can be divided into three categories:

1. Operators approximating derivatives of the image function using differences. Some are rotationally invariant (e.g., Laplacian) and thus are computed from one convolution mask only. Others, which approximate first derivatives, use several masks. The orientation is estimated on the basis of the best matching of several simple patterns.
2. Operators based on zero-crossings of the image function second derivative (e.g., Marr-Hildreth or Canny edge detectors).
3. Operators which attempt to match an image function to a parametric model of edges.

Edge detection is an extremely important step facilitating higher-level imageanalysis and remains an area of active research. Examples of the variety of

approaches found in current literature are fuzzy logic, neural networks, or wavelets. It may be difficult to select the most appropriate edge detection strategy.

Individual gradient operators that examine small local neighborhoods are in fact convolutions (cf. equation 5.23), and can be expressed by convolution masks. Operators which are able to detect edge direction are represented by a collection of masks, each corresponding to a certain direction.

Roberts operator

The Roberts operator is one of the oldest [Roberts, 1965], and is very easy to compute as it uses only a 2 × 2 neighborhood of the current pixel. Its masks are

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

so the magnitude of the edge is computed as g(i, j) − g(i + 1, j + 1) + g(i, j + 1) − g(i + 1, j) . (5.40)

The primary disadvantage of the Roberts operator is its high sensitivity to noise, because very few pixels are used to approximate the gradient.

Laplace operator

The Laplace operator ∇^2 is a very popular operator approximating the second derivative which gives the edge magnitude only. The Laplacian, equation (5.35), is

× approximated in digital images by a convolution sum. A 3x3 mask h is often used; for 4-neighborhoods and 8-neighborhoods it is defined as

$$h = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \qquad h = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

A Laplacian operator with stressed significance of the central pixel or its neighborhood is sometimes used. In this approximation it loses invariance to rotation

$$h = \begin{bmatrix} 2 & -1 & 2 \\ -1 & -4 & -1 \\ 2 & -1 & 2 \end{bmatrix}, \qquad h = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix}$$

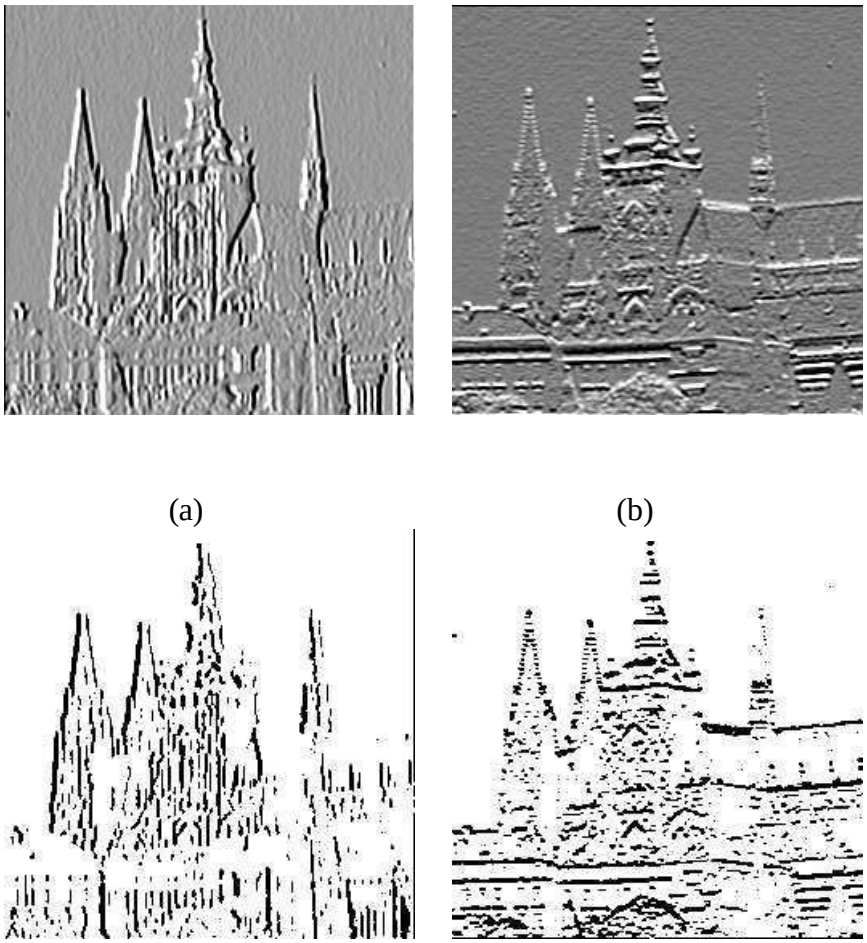
The Laplacian operator has a disadvantage—it responds doubly to some edges in the image.

Prewitt operator

The Prewitt operator, similarly to the Sobel, Kirsch, and some other operators, approximates the first derivative. The gradient is estimated in eight (for a 3x 3 convolution mask) possible directions, and the convolution result of greatest magnitude indicates the gradient direction. Larger masks are possible. We present only the first three 3 x 3 masks for each operator; the others can be created by simple rotation.

$$h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}, \quad h_2 = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}, \quad h_3 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

The direction of the gradient is given by the mask giving maximal response. This is also the case for all the following operators approximating the first derivative.



(a) (b)

(c) (d)

Figure 5.19: First-derivative edge detection using Prewitt operators. (a) North direction (the brighter the pixel value, the stronger the edge). (b) East direction. (c) Strong edges from (a).

(d) Strong edges from (b).

Sobel operator

The Sobel operator is often used as a simple detector of horizontality and verticality of edges, in which case only masks h1 and h3 are used. If the h1 response is y and the h3 response x, we might then derive edge strength (magnitude) as

$$\sqrt{x^2 + y^2} \qquad \text{or} \qquad |x| + |y|$$

And direction as arctan(y/x).

Kirsch operator

$$h_1 = \begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{bmatrix}, \quad h_2 = \begin{bmatrix} 3 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & -5 & 3 \end{bmatrix}, \quad h_3 = \begin{bmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{bmatrix}$$

To illustrate the application of gradient operators on real images, consider again the image given in Figure 5.9a. The Laplace edge image calculated is shown in Figure 5.18a; the value of the operator has been histogram equalized to enhance its visibility.

The properties of an operator approximating the first derivative are demonstrated using the Prewitt operator—results of others are similar. The original image is again given in Figure 5.9a; Prewitt approximations to the directional gradients are in Figures 5.19a,b, in which north and east directions are shown. Significant edges (those with above-threshold magnitude) in the two directions are given in Figures 5.19c,d.

2.4. Zero-crossings of the second derivative:

In the 1970s, Marr's theory concluded from neurophysiological experiments that

object boundaries are the most important cues that link an intensity image with its interpretation. Edge detection techniques existing at that time (e.g., the Kirsch, Sobel, and Pratt operators) were based on convolution in very small neighborhoods and worked well only for specific images. The main disadvantage of these edge detectors is their dependence on the size of the object andsensitivity to noise.

An edge detection technique based on the **zero-crossings** of the second derivative (**Marr-Hildreth** edge detector) explores the fact that a step edge corresponds to an abrupt change in the image function. The first derivative of the image function should have an extremum at the position corresponding to the edge in the image, and so the second derivative should be zero at the same position; however, it is much easier and more precise to find a zero-crossing position than an extremum. In Figure 5.20 this principle is illustrated in 1D for the sake of simplicity. Figure 5.20a shows step edge profiles of the original image function with two different slopes, Figure 5.20b depicts the first derivative of the image function, and Figure 5.20c illustrates the second derivative; notice that this crosses the zero level at the same position as the edge.

Considering a step-like edge in 2D, the 1D profile of Figure 5.20a corresponds to a cross section through the 2D step. The steepness of the profile will change if the

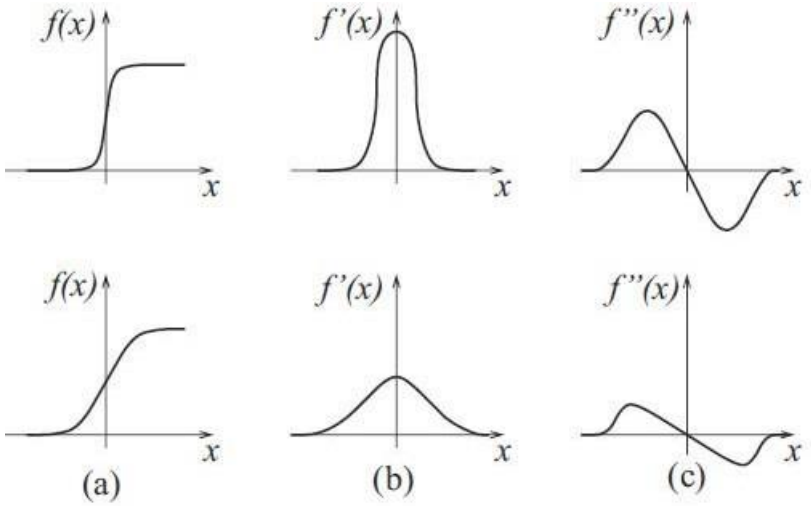


Figure 5.20: 1D edge profile of the zero-crossing.

orientation of the cutting plane changes—the maximum steepness is observed when the plane is perpendicular to the edge direction.

The crucial question is how to compute the second derivative robustly. One possibility is to smooth an image first (to reduce noise) and then compute second derivatives. When choosing a smoothing filter, there are two criteria that should be fulfilled. First, the filter should be smooth and roughly band limited in the frequency domain to reduce the possible number of frequencies at which function changes can take place. Second, the constraint of spatial localization requires the response of a filter to be from nearby points in the image. These two criteria are conflicting, but they can be optimized simultaneously using a Gaussian distribution. In practice, one has to be more precise about what is meant by the localization performance of an operator, and the Gaussian may turn out to be suboptimal. We shall consider this in the next section.

The 2D Gaussian smoothing operator G(x, y) (also called a Gaussian filter, or simply a Gaussian) is given by

$$G(x,y) = e^{-(x^2+y^2)/2\sigma^2} ,$$

where x, y are the image co-ordinates and σ is a standard deviation of the associated probability distribution. Sometimes this is presented with a normalizing factor

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \qquad \text{or} \qquad G(x,y) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-(x^2+y^2)/2\sigma^2}$$

The standard deviation σ is the only parameter of the Gaussian filter—it isproportional to the size of the neighborhood on which the filter operates. Pixels more

distant from the center of the operator have smaller influence, and pixels farther than 3σ from the center have negligible influence.

Our goal is to obtain a second derivative of a smoothed 2D function f (x, y). We have already seen that the Laplace operator gives the second derivative, and is non- directional (isotropic). Consider then the Laplacian of an image f (x, y) smoothed by a Gaussian (expressed using a convolution). The operation is often abbreviated as **LoG**, from **Laplacian of Gaussian**

$$\nabla^2 \left[G(x, y, \sigma) * f(x, y) \right] .$$

The order of performing differentiation and convolution can be interchanged because of the linearity of the operators involved

$$\left[\nabla^2 G(x, y, \sigma) \right] * f(x, y) .$$

The derivative of the Gaussian filter $\nabla^2 G$, can be pre-computed analytically, since

it is independent of the image under consideration, and so the complexity of the composite operation is reduced. From equation (5.47), we see

$$\frac{\partial G}{\partial x} = -\left(\frac{x}{\sigma^2} \right) e^{-(x^2+y^2)/2\sigma^2}$$

and similarly for y. Hence

$$\frac{\partial^2 G}{\partial x^2} = \frac{1}{\sigma^2} \left(\frac{x^2}{\sigma^2} - 1 \right) e^{-(x^2+y^2)/2\sigma^2} , \quad \frac{\partial^2 G}{\partial y^2} = \frac{1}{\sigma^2} \left(\frac{y^2}{\sigma^2} - 1 \right) e^{-(x^2+y^2)/2\sigma^2}$$

After introducing a normalizing multiplicative coefficient c, we get aconvolution mask of a LoG operator:

$$h(x, y) = c \left(\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} \right) e^{-(x^2+y^2)/2\sigma^2} ,$$

where c normalizes the sum of mask elements to zero. Because of its shape, the inverted LoG operator is commonly called a **Mexican hat**. An example of a 5 × 5 discrete approximation (wherein a 17 × 17 mask is also given) is

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}.$$

Of course, these masks represent truncated and discrete representations of infinite continuous functions, and care should be taken in avoiding errors in moving to this representation.

Finding second derivatives in this way is very robust. Gaussian smoothing effectively suppresses the influence of the pixels that are more than a distance 3σ from the current pixel; then the Laplace operator is an efficient and stable measure of changes in the image.

After image convolution $\nabla^2 G,$ with the locations in the convolved image where the zero level is crossed correspond to the positions of edges. The advantage of this approach compared to classical edge operators of small size is that a larger area surrounding the current pixel is taken into account; the influence of more distant points decreases according to the σ of the Gaussian. In the ideal case of an isolated step edge, the σ variation does not affect the location of the zero-crossing. Convolution masks become large for larger σ; for

example, σ = 4 needs a mask about 40 pixels wide. Fortunately, there is a separable decomposition of the $\nabla^2 G,$ operator that can speed up computation considerably.

The practical implication of Gaussian smoothing is that edges are found reliably. If only globally significant edges are required, the standard deviation σ of the Gaussian smoothing filter may be increased, having the effect of suppressing less significant evidence.

The $\nabla^2 G,$ operator can be very effectively approximated by convolution with a mask that is the difference of two Gaussian averaging masks with substantially different σ—this method is called the **difference of Gaussians**, abbreviated as **DoG**.

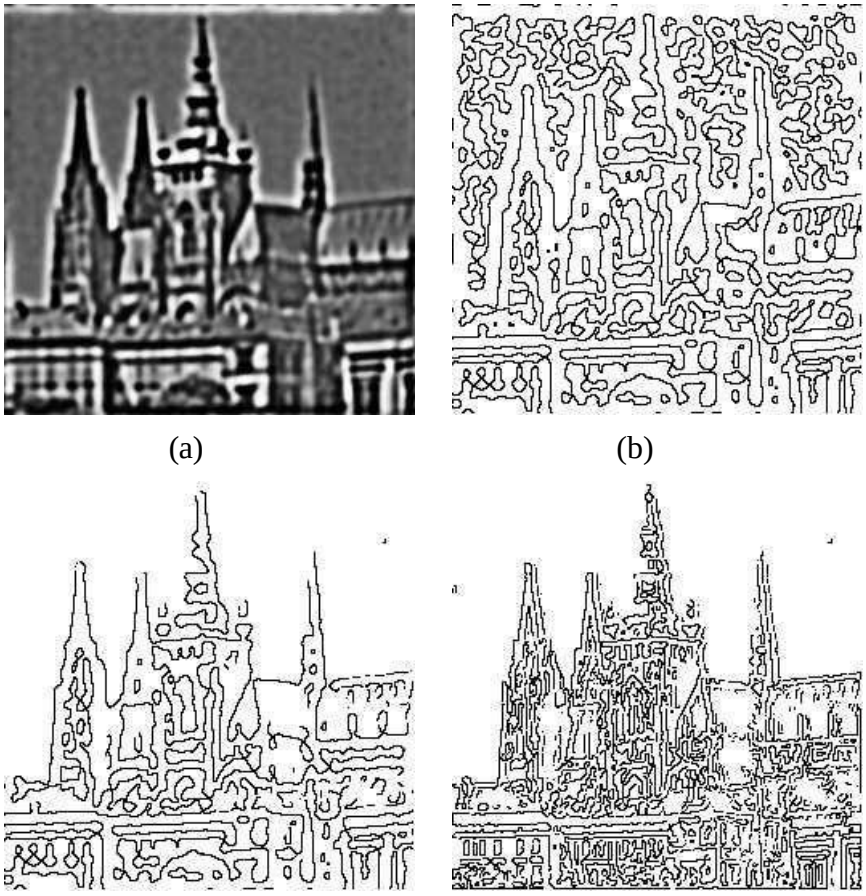
When implementing a zero-crossing edge detector, trying to detect zeros in the LoG or DoG image will inevitably fail, while naive approaches of thresholding the

×

LoG/DoG image and defining the zero-crossings in some interval of values close to zero give piece- wise disconnected edges at best. To create a well-functioning second-derivative edge detector, it is necessary to implement a true zero-crossing detector. A simple detector may identify a zero-crossing in a moving 2 × 2 window, assigning an edge label to any one corner pixel, say the upper left, if LoG/DoG image values of both polarities occur in the 2 x 2 window;

no edge label would be given if values within the window are either all positive or all negative. Another post-processing step to avoid detection of zero- crossings corresponding to non-significant edges in regions of almost constant gray-level would admit only those zero-crossings for which there is sufficient edge evidence from a first-derivative edge detector. Figure 5.21 provides several examples of edge detection using zero crossings of the second derivative.

Many other approaches improving zero-crossing performance can be found in the literature; some of them are used in pre-processing or post-processing steps. The traditional second-derivative zero-crossing technique has disadvantages as



(a) (b)

(c) (d)

Figure 5.21: Zero-crossings of the second derivative, see Figure 5.9a for the original image.

(a) DoG image ($\sigma 1 = 0.10$, $\sigma 2 = 0.09$), dark pixels correspond to negative values, bright pixels to positive. (b) Zero-crossings of the DoG image. (c) DoG zerocrossing edges after removing edges lacking first-derivative support. (d) LoG zero-crossing edges ($\sigma = 0.20$) after removing edges lacking first-derivative support—note different scale of edges due to different Gaussian smoothing parameters.

well. First, it smooths the shape too much; for example, sharp corners are lost. Second, it tends to create closed loops of edges (nicknamed the 'plate of spaghetti' effect).

Neurophysiological experiments provide evidence that the human eye retina in the form of the **ganglion cells** performs operations very similar to the $\nabla^2 G$, operations. Each such cell responds to light stimuli in a local neighborhood called the **receptive field**, which has a center-surround organization of two complementary types, off-center and on-center. When a light stimulus occurs, activity of on-center cells increases and that of off-center cells is inhibited. The retinal operation on the image can be described analytically as the convolution of the image with the $\nabla^2 G$ operator.

2.5. Scale in image processing:

- Many image processing techniques work locally, theoretically at the level of individual pixels—edge detection methods are an example. The essential problem in such computation is **scale**.
- Edges correspond to the gradient of the image function, which is computed as a difference between pixels in some neighborhood.
- There is seldom a sound reason for choosing a particular size of neighborhood, since the 'right' size depends on the size of the objects under investigation.
- To know what the objects are assumes that it is clear how to interpret an image, and this is not in general known at the pre-processing stage.
- The solution to the problem formulated above is a special case of a general paradigm called the **system approach**. This methodology is common in cybernetics or general system theory to study complex phenomena.
- The phenomenon under investigation is expressed at different resolutions of the de- scription, and a formal model is created at each resolution. Then the qualitative behavior of the model is studied under changing resolution of the description. Such a methodology enables the deduction of meta-knowledge about the phenomenon that is not seen at the individual description levels.
- Different description levels are easily interpreted as different scales in the domain of digital images. The idea of scale is fundamental to Marr's edge detection technique, where different scales are provided by different sizes of Gaussian filter masks. The aim was not only to eliminate fine scale noise but also to separate events at different scales arising from distinct physical processes.
- Assume that a signal has been smoothed with several masks of variable sizes. Every setting of the scale parameters implies a different description, but it is not known which one is correct; for many tasks, no one scale is categorically correct. If the ambiguity introduced by the scale is inescapable, the goal of scale-independent description is to reduce this ambiguity as much as possible.
- Here we shall consider just three examples of the application of multiple scale description to image analysis.

1. The first approach aims to process planar noisy curves at a range of scales— the segment of curve that represents the underlying structure of

the scene needs to be found. The problem is illustrated by an example of two noisy curves.

- One of these may be interpreted as a closed curve, while the other could be described as two intersecting straight lines.
- Local tangent direction and curvature of the curve are significant only with some idea of scale after **the curve is smoothed by a Gaussian filter with varying standard deviations.**

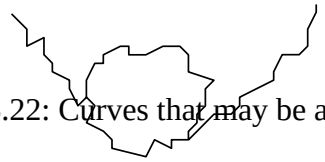


Figure 5.22: Curves that may be analyzed at multiple scales.

2. A second approach, called **scale-space filtering**, tries to describe signals qualitatively with respect to scale. The problem was formulated for 1D signals f (x), but it can easily be generalized for 2D functions as images. The original 1D signal f (x) is smoothed by convolution with a 1D Gaussian

$$G(x,\sigma) = e^{-x^2/2\sigma^2} . \qquad (5.51)$$

If the standard deviation σ is slowly changed, the function

$$F(x,\sigma) = f(x) * G(x,\sigma) \qquad (5.52)$$

represents a surface on the (x, σ) plane that is called the **scale-space image**.

Inflection points of the curve F (x, σ0) for a distinct value σ0

$$\frac{\partial^2 F(x,\sigma_0)}{\partial x^2} = 0 \qquad \text{and} \qquad \frac{\partial^3 F(x,\sigma_0)}{\partial x^3} \neq 0 \qquad (5.53)$$

describe the curve f (x) qualitatively. The positions of inflexion points can be drawn as a set of curves in (x, σ) co-ordinates. Coarse to fine analysis of the curves corresponding to inflexion points, i.e., in the direction of decreasing value of the σ, localizes large-scale events.

The qualitative information contained in the scale-space image can be transformed into a simple **interval tree** that expresses the structure of the signal f (x) over all observed scales. The interval tree is built from the root that corresponds to the largest scale (σ_{max}), and then the scale-space image is searched in the direction of decreasing σ. The interval tree branches at those points where new curves corresponding to inflexion points appear

3. The third example of the application of scale is that used by the popular **Canny edge detector**. Since the Canny detector is a significant and widely used contribution to edge detection techniques, its principles will be explained in detail.

2.6. Canny edge detection:

Canny proposed an approach to edge detection that is optimal for step edges corrupted by white noise. The optimality of the detector is related to three criteria.

1) The **detection** criterion expresses the fact that important edges should not be missed and that there should be no spurious responses.
2) The **localization** criterion says that the distance between the actual and located position of the edge should be minimal.
3) The **one response** criterion minimizes multiple responses to a single edge. This is partly covered by the first criterion, since when there are two responses to a single edge, one of them should be considered as false. This third criterion solves the problem of an edge corrupted by noise and works against non-smooth edge operators.

Canny's derivation is based on several ideas.

1. The edge detector was expressed for a 1D signal and the first two optimality criteria. A closed-form solution was found using the calculus of variations.
2. If the third criterion (multiple responses) is added, the best solution may be found by numerical optimization. The resulting filter can be approximated effectively with error less than 20% by the first derivative of a Gaussian smoothing filter with standard deviation σ [Canny, 1986]; the reason for doing this is the existence of an effective implementation. There is a strong similarity here to the LoG based Marr-Hildreth edge detector.
3. The detector is then generalized to two dimensions. A step edge is given by its position, orientation, and possibly magnitude (strength). It can be shown that convolving an image with a symmetric 2D Gaussian and then differentiating in the direction of the gradient (perpendicular to the edge direction) forms a simple and effective directional operator (recall that the Marr-Hildreth zerocrossing operator does not give information about edge direction, as it uses a Laplacian filter).

   Suppose G is a 2D Gaussian [equation (5.47)] and assume we wish to convolve the image with an operator $G_{\mathbf{n}}$ which is a first derivative of G in some direction **n**

$$G_n = \frac{\partial G}{\partial \mathbf{n}} = \mathbf{n}\,\nabla G\,. \qquad (5.54)$$

   We would like **n** to be perpendicular to the edge: this direction is not known in advance, but a robust estimate of it based on the smoothed gradient direction is available. If f is the image, the normal to the edge **n** is estimated as

$$\mathbf{n} = \frac{\nabla (G * f)}{\left|\nabla (G * f)\right|}\,. \qquad (5.55)$$

The edge location is then at the local maximum of the image f convolved with the operator G_n in the direction **n**

$$\frac{\partial}{\partial \mathbf{n}} G_n * f = 0 \,. \qquad (5.56)$$

Substituting in equation (5.56) for G_n from equation (5.54), we get

$$\frac{\partial^2}{\partial \mathbf{n}^2} G * f = 0 \,. \qquad (5.57)$$

This equation (5.57) illustrates how to find local maxima in the direction perpendicular to the edge; this operation is often referred to as **non-maximal suppression** (see also Algorithm 6.4).

As the convolution and derivative are associative operations in equation (5.57), we can first convolve an image f with a symmetric Gaussian G and then compute the directional second-derivative using an estimate of the direction **n** computed according to equation (5.55). The strength of the edge (magnitude of the gradient of the image intensity function f) is measured as

$$\left|G_n * f\right| = \left|\nabla (G * f)\right| \,. \qquad (5.58)$$

4. Spurious responses to a single edge caused by noise usually create a 'streaking' problem that is very common in edge detection in general. The output of an edge detector is usually thresholded to decide which edges are significant, and streaking may break up edge contours as the operator fluctuates above and below the threshold. Streaking can be eliminated by **thresholding with**

Algorithm 5.4: Canny edge detector

1. Convolve an image f with a Gaussian of scale σ.
2. Estimate local edge normal directions $\mathbf{n}$ using equation (5.55) for each pixel in the image.

$$\mathbf{n} = \frac{\nabla(G * f)}{|\nabla(G * f)|} . \qquad (5.55)$$

3. Find the location of the edges using equation (5.57) (non-maximal suppression).

$$\frac{\partial^2}{\partial \mathbf{n}^2} G * f = 0 . \qquad (5.57)$$

4. Compute the magnitude of the edge using equation (5.58).

$$|G_n * f| = |\nabla(G * f)| . \qquad (5.58)$$

5. Threshold edges in the image with hysteresis to eliminate spurious responses.
6. Repeat steps (1) through (5) for ascending values of the standard deviation σ.
7. Aggregate the final information about edges at multiple scale using the 'feature synthesis' approach.

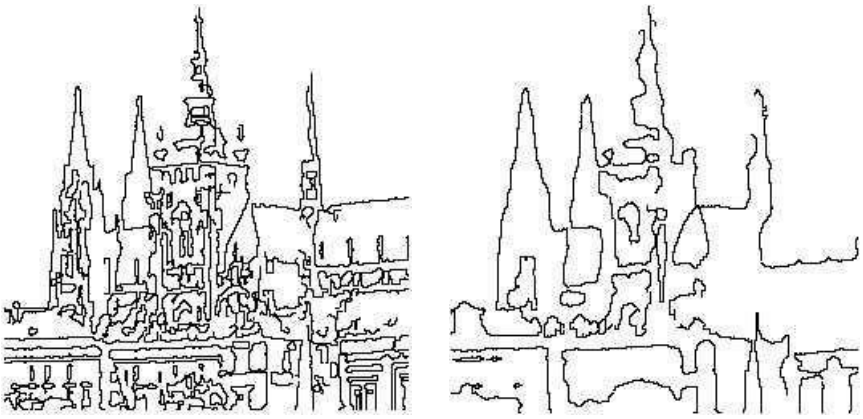
hysteresis, employing a hard (high) threshold and a soft (lower) threshold— see Algorithm 6.5. The low and high thresholds are set according to an estimated signal-to-noise ratio.

5. The correct scale for the operator depends on the objects contained in the image. The solution to this unknown is to use multiple scales and aggregate information from them. Different scales for the Canny detector are represented by different standard deviations σ of the Gaussians. There may be several scales of operators that give significant responses to edges (i.e., signal-to-noise ratio above the thresh- old); in this case the operator with the smallest scale is chosen, as it gives the best localization of the edge.

   Canny proposed a **feature synthesis** approach. All significant edges from the operator with the smallest scale are marked first, and the edges of a hypothetical operator with larger σ are synthesized from them (i.e., a prediction is made of how the large σ should perform on the evidence gleaned from the smaller σ. Then the synthesized edge response is compared with the actual edge response for larger σ. Additional edges are marked only if they have a significantly stronger response than that predicted from synthetic output.

   This procedure may be repeated for a sequence of scales, a cumulative edge map being built by adding those edges that were not identified at smaller scales.

Figure 5.23a shows the edges of Figure 5.9a detected by a Canny operator with σ = 1.0. Figure 5.23b shows the edge detector response for σ = 2.8 (feature synthesis has not been applied here).



(a) (b)

Figure 5.23: Canny edge detection at two different scales. © Cengage Learning 2015.

Canny's detector represents a complicated but major contribution to edge detection. Its full implementation is unusual, it being common to find implementations that omit feature synthesis—that is, just steps 1–5 of Algorithm 5.4.

2.7. Parametric edge models:

Parametric models are based on the idea that the discrete image intensity function can be considered a sampled and noisy approximation of an underlying continuous or piecewise continuous image intensity function.

While this function is not known, it can be estimated from the available discrete image intensity function and image properties can be determined from this continuous estimate, possibly with subpixel precision.

It is usually impossible to represent image intensities using a single continuous function since a single function leads to high-order intensity functions in x and y. Instead, piecewise continuous function estimates called **facets** are used to represent (a neighborhood of) each image pixel. Such an image representation is called a **facet model.**

The intensity function in a neighborhood can be estimated using models of different complexity.

The simplest one is the flat facet model that uses piecewise constants and each pixel neighborhood is represented by a flat function of constant intensity. The sloped model uses piecewise linear functions forming a sloped plane fitted to local image intensities.

Quadratic and bi-cubic facet models employ more complex functions.

Once the facet model parameters are available for each image pixel, edges can be detected as extrema of the first directional derivative and/or zero-crossings of the second directional derivative of the local continuous facet model functions.

An example will illustrate: consider a bi-cubic facet model $g(i, j) = c1 + c2\ x + c3\ y + c4\ x^2 + c5\ x\ y + c6\ y^2 + c7\ x^3 + c8\ x^2\ y + c9\ x\ y^2 + c10\ y^3$, (5.59) whose parameters are estimated from a pixel neighborhood (the co-ordinates of the central pixel are (0,0)). This may be performed by, e.g., a least-squares method with SVD; alternatively, coefficients c_i can be computed directly using a set of ten 5x5 kernels. Once parameters are available at each pixel, edges may be located as extrema of the first directional derivative, or zero crossings of the second derivative, of the local facet model functions.

Benefits:

1) Edge detectors based on parametric models describe edges more precisely than convolution-based edge detectors.
2) They carry the potential for subpixel edge localization.

Limitations:

- Their computational requirements are much higher.
- Promising extensions combine facet models with Canny's edge detection criteria and relaxation labeling.

2.8. Edges in multi-spectral images:

One pixel in a multi-spectral image is described by an n-dimensional vector, and brightness values in n spectral bands are the vector components. There are several possibilities for the detection of edges in multi-spectral images.

Trivially, we might detect edges separately in individual image spectral components using the ordinary local gradient operators. Individual images of edges can be combined to get the resulting image, with the value corresponding to edge magnitude and direction being a selection or combination of the individual edge spectral components.

Alternatively, we may create a multi-spectral edge detector which uses brightness information from all n spectral bands; this approach is also applicable to multi-dimensional images forming three- or higher-dimensional data volumes. The neighborhood used has size 2x n pixels, where the 2x 2 neighborhood is similar to that of the Roberts gradient, equation (5.39). The coefficients weighting the influence of the component pixels are similar to the correlation coefficients. Let f(i, j) denote the arithmetic mean of the brightnesses corresponding to the pixels with the same co-ordinates (i, j) in all n spectral component images, and f_r be the brightness of the r^{th} spectral component. The edge detector result in pixel (i, j) is given as the minimum of the following expression:

$$\frac{\sum_{r=1}^{n}\left[d(i,j)\right]\left[d(i+1,j+1)\right]}{\sqrt{\sum_{r=1}^{n}\left[d(i,j)\right]^2\sum_{r=1}^{n}\left[d(i+1,j+1)\right]^2}}\;\frac{\sum_{r=1}^{n}\left[d(i+1,j)\right]\left[d(i,j+1)\right]}{\sqrt{\sum_{r=1}^{n}\left[d(i+1,j)\right]^2\sum_{r=1}^{n}\left[d(i,j+1)\right]^2}}\,,$$

where $d(k,l) = f_r(k,l) - \overline{f}(k,l)$. (5.60)

2.9. Local pre-processing in the frequencydomain:

The Fourier transform makes convolution of two images in the frequency domain very easy, and it is natural to consider applying many of the filters in the frequency domain. Such operations are usually called **spatial frequency filtering**.

Assume that f is an input image and F is its Fourier transform. A convolution filter h can be represented by its Fourier transform H; h may be called the unit pulse response of the filter and H the frequency transfer function, and either of the representations h or H can be used to describe the filter. The Fourier transform of the filter output after an image f has been convolved with the filter h can be computed in the frequency domain

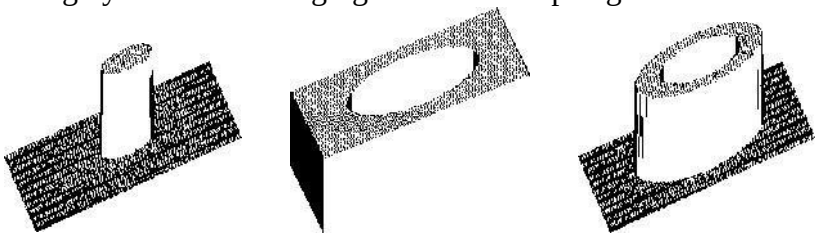
$$G = F.* \,, \tag{5.61}$$

where . represents an element$*$ -by-element multiplication of matrices F and H (not matrix multiplication). The filtered image g can be obtained by applying the inverse

Fourier transform to **G**—equation (3.28).

Some basic examples of spatial filtering are linear **low-pass**, **high-pass**, and **band- pass** frequency filters.

1) A low-pass filter is defined by a frequency transfer function H(u, v) with small values at points located far from the co-ordinate origin in the frequency domain (that is, small transfer values for high spatial frequencies) and large values at points close to the origin (large transfer values for low spatial frequencies)—see Figure 5.24a. It preserves low spatial frequencies and suppresses high spatial frequencies, and has behavior similar to smoothing by standard averaging—it blurs sharp edges.



(a) **(b)** **(c)**

Figure 5.24: Frequency filters displayed in 3D. (a) Low-pass filter. (b) High-pass filter. (c) Band- pass filter.

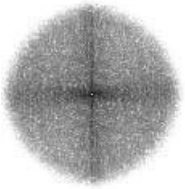
2) A high-pass filter is defined by small transfer function values located around the frequency co-ordinate system origin, and larger values outside this area— larger transfer coefficients for higher frequencies (Figure 5.24b).

- Band-pass filters, which select frequencies in a certain range for enhancement, are constructed in a similar way, and also filters with directional response, etc. (Fig- ure 5.24c).

The most common image enhancement problems include noise suppression, edge enhancement, and removal of noise which is structured in the frequency spectrum. Noise represents a high-frequency image component, and it may be suppressed applying a low-pass filter as shown in Figure 5.25, which demonstrates the principles of frequency filtering on Fourier image spectra; the original image spectrum is multiplied by the filter spectrum and a low-frequency image spectrum results. Unfortunately, all high-frequency phenomena are suppressed, including high frequencies that are not related to noise (sharp edges, lines, etc.). Low-pass filtering results in a blurred image.

(a)

(b)



(c)

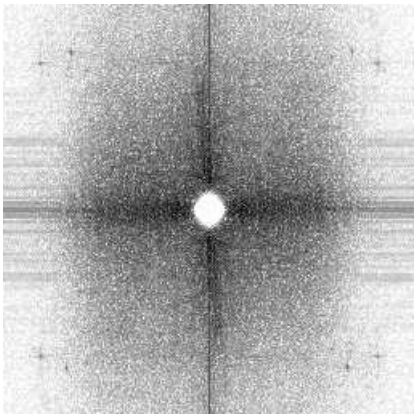
(d)

Figure 5.25: Low-pass frequency-domain filtering—for the original image and its spectrum see Figure 3.7. (a) Spectrum of a low-pass filtered image, all higher frequencies filtered out. (b) Image resulting from the inverse Fourier transform applied to spectrum (a). (c) Spectrum of a low-pass filtered image, only very high frequencies filtered out. (d) Inverse Fourier transform applied to spectrum (c).

Again, edges represent a high-frequency image phenomenon. Therefore, to enhance them, low-frequency components of the image spectrum must be suppressed—to achieve this, a high-frequency filter must be applied.

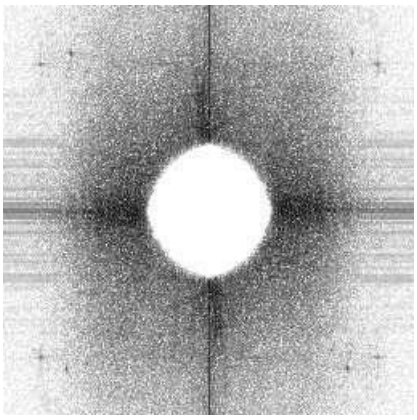
To remove noise which is structured in the frequency domain, the filter design must include a priori knowledge about the noise properties. This knowledge may be acquired either from the image data or from the corrupted image Fourier spectrum, where the structured noise usually causes notable peaks.

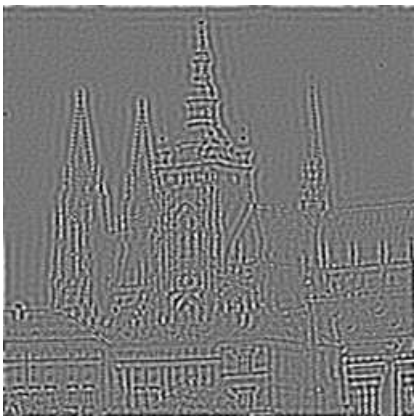
Some examples of frequency domain image filtering are shown in Figures 5.25–5.28. The original image was shown in Figure 3.8 and its frequency spectrum in Figure 3.7. Figure 5.26 shows results after application of a high-pass filter followed by an inverse Fourier transform. It can be seen that edges represent high-frequency phenomena in the image. Results of band-pass filtering can be seen in Figure 5.27. Figure 5.28 gives an even more powerful example of frequency filtering—removal of periodic noise. The vertical noise lines in the original image are transformed into frequency spectrum peaks after the transform. To remove these frequencies, a filter was designed which suppresses the periodic noise in the image, which is visible as white circular areas.



(a)

(b)





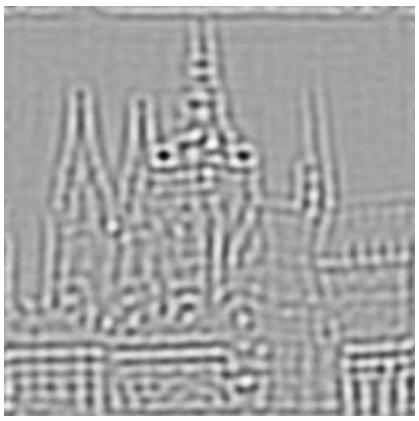
(c) (d)

Figure 5.26: High-pass frequency domain filtering. (a) Spectrum of a high-pass filtered image, only very low frequencies filtered out. (b) Image resulting from the inverse Fourier transform applied to spectrum (a). (c) Spectrum of a high-pass filtered image, all lower frequencies filtered out. (d) Inverse Fourier transform applied to spectrum (c).

There are several filters which prove useful for filtering in the frequency domain: two important representatives of them are the Gaussian and Butterworth filters. Choose an isotropic filter for simplicity,

$D(u,v) = D(r) = \sqrt{u^2 + v^2}$, and let D_0 be a parameter of the filter called the cut-off frequency. For the Gaussian, D_0 coincides with the dispersion σ. The Fourier spectrum of a low-pass Gaussian filter G_{low} is

$$G_{\text{low}}(u,v) = \exp\left(-\frac{1}{2}\left(\frac{D(u,v)}{D_0}\right)^2\right) . \qquad (5.62)$$



(a) (b)

Figure 5.27: Band-pass frequency domain filtering. (a) Spectrum of a band-passfiltered image, low and high frequencies filtered out. (b) Image resulting from the inverse Fourier transform applied to spectrum (a).

The Butterworth filter is specified to have maximally flat frequency response over a spectrum band, and is also called a 'maximally flat magnitude filter'. The frequency response of the 2D low-pass Butterworth filter B_{low} of degree n is

$$B_{\text{low}} = \frac{1}{1 + \left(\frac{D(u,v)}{D_0} \right)^n} . \qquad (5.63)$$

The usual Butterworth filter degree is n = 2, which will be used here. Figure 5.29 illustrates the shape of the Gaussian and Butterworth filters for $D_0 = 3$ in 1D plots.

The high-pass filter is created easily from the low-pass filter. If the Fourier frequency spectrum of a low-pass filter is H_{low}, the high-pass filter can be created by just flipping it vertically, $H^{-}_{high} = 1\text{-}H_{low}$.

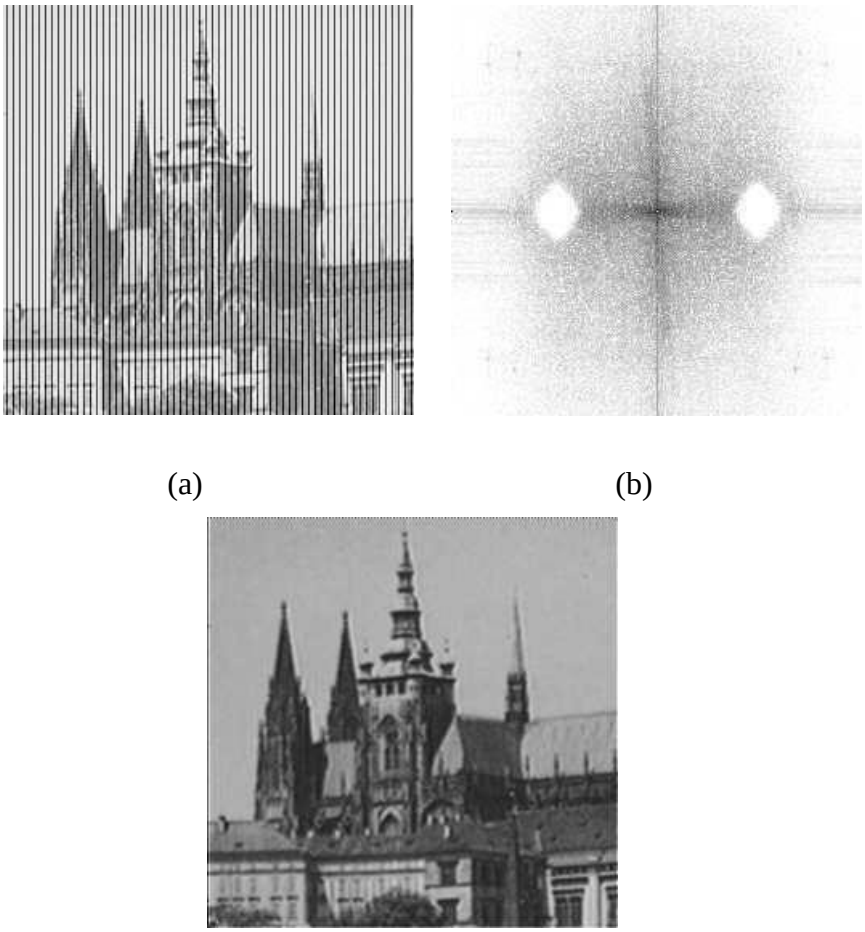
Another useful pre-processing technique operating in the frequency domain is an instance of **homomorphic filtering**. Homomorphic filtering is used to remove multiplicative noise. The aim of the particular homomorphic filtering is to simultaneously increase contrast and normalize image intensity across the image.

The assumption is that the image function f (x, y) can be factorized as a product of two independent multiplicative components in each pixel: illumination i(x, y) and the reflectance r(x, y) at the point in the observed scene, f (x, y) = i(x, y) r(x, y). These two components can be separated in some images because the illumination component tends to vary slowly and the reflectance component varies more quickly. The idea of the separation is to apply a logarithmic transform to the input image

$$z(x, y) = \log f (x, y) = \log i(x, y) + \log r(x, y) . \qquad (5.64)$$

If the image z(x, y) is converted to Fourier space (denoted by capital letters) then its additive components remain additive due to the linearity of the Fourier transform

$$Z(u, v) = I(u, v) + R(u, v) . \qquad (5.65)$$



(a) (b)

(c)

Figure 5.28: Periodic noise removal. (a) Noisy image. (b) Image spectrum used for image reconstruction—note that the areas of frequencies corresponding with periodic vertical lines are filtered out. (c) Filtered image. © Cengage Learning 2015.

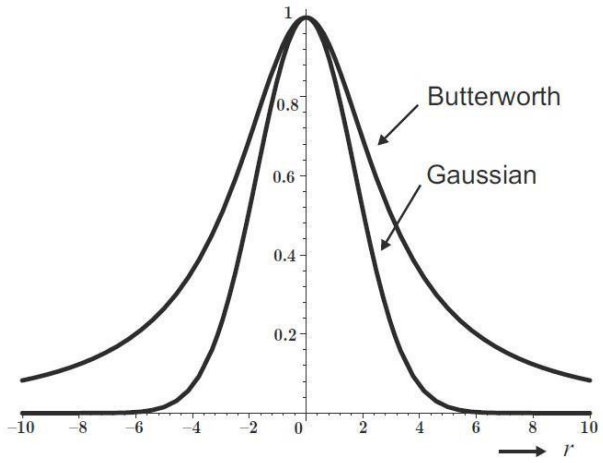


Figure 5.29: Gaussian and Butterworth low-pass filters. © *Cengage Learning 2015.*

Assume that the Fourier spectrum Z(u, v) is filtered by the filter H(u, v) and the spectrum S(u, v) is the result

$$S = H .* Z = H .* I + H .* R . \quad (5.66)$$

Usually a high-pass filter is used for this purpose; assuming a high-pass Butterworth filter, it has to be damped in order not to suppress low frequencies entirely as they bear needed information too. The Butterworth filter modified by damping coefficient 0.5 is shown in Figure 5.30

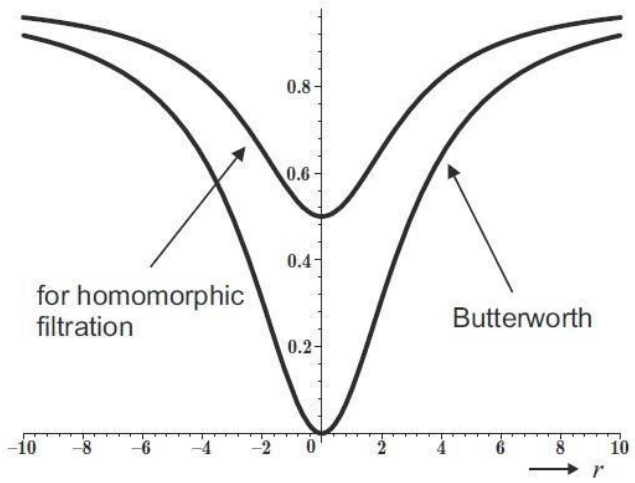


Figure 5.30: High-pass filter used in homomorphic filtering. It is a Butterworth filter damped by a 0.5 coefficients to retain some low frequencies. © *Cengage Learning 2015.*

Having the filtered spectrum S(u, v), we can return to spatial coordinates using the inverse Fourier transform, s(x, y) = −1S(u, v). Recall that the logarithm was first applied to the input image f (x, y) in equation (5.64). Now the image has to be transformed by the logarithm inverse function; this inverse function is the exponential. The result—the image g(x, y) filtered by the homomorphic filter—is given by g(x, y) = exp s(x, y) .

An illustration of the effect of homomorphic filtering is in Figure 5.31, an image of a person in a dark tunnel with strong illumination at the entrance. Detail of the tunnel surface on the top and right side are not visible because the surface is too dark. The result of homomorphic filtering is in Figure 5.31b. More details can be seen in this image.

(a) (b)

Figure 5.31: Illustration of homomorphic filtering. (a) Original image. (b) Homomorphic filtering.

2.10. Line detection by local pre-processingoperators:

Several other local operations exist which do not belong to the taxonomy given in Section 5.3, as they are used for different purposes such as line finding, line thinning, and line filling operators. Another group of operators finds '**interest points**' or '**locations of interest**' in the image.

It is interesting to seek features richer than edges which can be reliably detected in the image and which can outperform simple edge detectors in some classes of applications. Line detectors and corner detectors are some such. Line detectors are used to detect linear objects such as dimension lines in engineering drawings or railways or roads in satellite images. Corner detectors and other interest point-like detectors are used mainly to register two or more images one to the other (e.g, in stereo vision, motion analysis, panorama stitching, object recognition from images) or to index the image or dominant objects in it to an image database.

Line finding operators aim to find very thin curves in the image; it is assumed that curves do not bend sharply. Such curves and straight lines are called **lines** for the purpose of describing this technique. If a cross section perpendicular in direction to the tangent of a line is examined, we get a roof profile (see Figure 5.17) when examining edges. We assume that the width of the lines is approximately one or two pixels.

The presence of a line may be detected by local convolution of the image with con- volution kernels which serve as line patterns. The simplest collection of four such patterns of size 3 x 3 is able to detect lines rotated modulo the angle 45o. Three of four such convolution kernels are

$$h_1 = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}, \quad h_2 = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}, \quad h_3 = \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}, \quad \ldots \quad (5.67)$$

A similar principle can be applied to bigger masks. The case of 5x5 masks is common. Such line detectors sometimes produce more lines than needed, and other non-linear constraints may be added to reduce this number. More sophisticated approaches deter- mine lines in images as ridges and ravines using the facet model. Line detection is frequently used in remote sensing and in document processing;

Local information about edges is the basis of a class of image segmentation techniques. Edges which are likely to belong to object boundaries are usually found by simple thresholding of the edge magnitude—such edge thresholding does not provide ideal contiguous boundaries that are one pixel wide. Sophisticated segmentation techniques that are dealt with in the next chapter serve this purpose. Here, much simpler edge thinning and filling methods are described. These techniques are based on knowledge of small local neighborhoods and are very similar to other local pre-processing techniques.

Thresholded edges are usually wider than one pixel, and **line thinning** techniques may give a better result. One line thinning method uses knowledge about edge orientation and in this case edges are thinned before thresholding. Edge magnitudes and directions provided by some gradient operator are used as input, and

the edge magnitudes of two neighboring pixels perpendicular to the edge direction are examined for each pixel in the image. If at least one of these pixels has edge magnitude higher than the edge magnitude of the examined pixel, then the edge magnitude of the examined pixel is assigned a zero value.

There are many other line thinning methods. In most cases the best results are achieved using mathematical morphology methods.

2.11. Image restoration:

Pre-processing methods that aim to suppress degradation using knowledge about its nature are called **image restoration**. Most image restoration methods are based on convolution applied globally to the whole image. There is a wide literature on restoration and only the basic principles and some simple degradations are considered here.

Image degradation can have many causes: defects of optical lenses, nonlinearity of the electro-optical sensor, graininess of the film material, relative motion between an object and camera, wrong focus, atmospheric turbulence in remote sensing or astronomy, scanning of photographs, etc. The objective of image restoration is to reconstruct the original image from its degraded version.

Image restoration techniques can be classified as **deterministic** or **stochastic**. Deterministic methods are applicable to images with little noise and a known degradation function. The original image is obtained by applying the function inverse to the degraded one. Stochastic techniques try to find the best restoration according to a particular statistical criterion, e.g., a least-squares method. There are three typical degradations with a simple function: relative constant speed movement of the object with respect to the camera, wrong lens focus, and atmospheric turbulence.

In most practical cases, there is insufficient knowledge about the degradation, and it must be estimated and modeled. This may be done on an a priori or a posteriori basis:

2.11.1. A priori knowledge about degradation is either known in advance or can be obtained before restoration. For example, if it is known that the image was degraded

by relative motion of an object with respect to the sensor, then the modeling determines only the speed and direction of the motion. Alternatively, we may seek to to estimate parameters of a device such as a TV camera or digitizer, whose degradation remains unchanged over a period of time and can be modeled by studying a known sample image and its degraded version.

2.11.2. A posteriori knowledge is that obtained by analyzing the degraded image. A typical example is to find some interest points in the image (e.g., corners, straight lines) and guess how they looked before degradation. Another possibility is to use spectral characteristics of the regions in the image that are relatively homogeneous.

A degraded image g can arise from the original image f by a process which can be expressed as

$$g(i,j) = s\left(\int\int_{(a,b)\in\mathcal{O}} f(a,b)\,h(a,b,i,j)\,\mathrm{d}a\,\mathrm{d}b\right) + \nu(i,j)\,, \qquad (5.74)$$

where s is some non-linear function and ν describes the noise. This is often simplified by neglecting the non-linearity and assuming that the function h is invariant with respect to position in the image, giving

$$g(i, j) = (f * h)(i, j) + \nu(i, j)\,. \qquad (5.75)$$

If the noise is not significant in this equation, then restoration equates to inverse convolution (also called deconvolution). If noise is not negligible, then the inverse convolution is solved as an overdetermined system of linear equations. Methods based on minimization of least square error such as Wiener filtering (off-line) or Kalman filtering (recursive, on-line; see Section 16.6.1) are examples [Bates and McDonnell, 1986].

2.11.1. Degradations that are easy to restore

In the Fourier domain, we can express equation (5.75) as

$$G = H\,F\,. \qquad (5.76)$$

Therefore, overlooking image noise ν, knowledge of the degradation function fully facilitates image restoration by inverse convolution (Section 5.4.2).

Relative motion of camera and object

Relative motion of a camera with a mechanical shutter and the photographed object during the shutter open time T causes smoothing of the object in the image. Suppose V is the constant speed in the direction of the x axis; the Fourier transform H(u, v) of the degradation caused in time T.

$$H(u,v) = \frac{\sin(\pi\,V\,T\,u)}{\pi\,V\,u}\,.$$

Wrong lens focus

Image smoothing caused by imperfect focus of a thin lens can be described by the function

$$H(u,v) = \frac{J_1(a\,r)}{a\,r}\,,$$

where J_1 is the Bessel function of the first order, $r^2 = u^2 + v^2$, and a is the displacement— the model is not space invariant.

Atmospheric turbulence

Atmospheric turbulence is degradation that needs to be restored in remote sensing and astronomy. It is caused by temperature non-homogeneity in the

atmosphere that deviates passing light rays. One mathematical model [Hufnagel and Stanley, 1964]
is

$$H(u, v) = e-c(u2 +v2)5/6 , \quad (5.79)$$

where c is a constant that depends on the type of turbulence which is usually found experimentally. The exponent 5/6 is sometimes replaced by 1.

2.11.2. Inverse filtering

Inverse filtering assumes that degradation was caused by a linear function h(i, j) (cf. equation 5.75) and considers the additive noise ν as another source of degradation. It is further assumed that ν is independent of the signal. After applying the Fourier transform to equation (5.75), we get

$$G(u, v) = F (u, v) H(u, v) + N (u, v) . \quad (5.80)$$

The degradation can be eliminated using the restoration filter with a transfer function that is inverse to the degradation h. We derive the original image F (its Fourier transform to be exact) from its degraded version G (equation 5.80), as

$$F (u, v) = G(u, v) H^{-1}(u, v) − N (u, v) H^{-1}(u, v) . \quad (5.81)$$

This shows that inverse filtering works well for images that are not corrupted by noise [not considering possible computational problems if H(u, v) gets close to zero at some location of the u, v space—fortunately, such locations can be neglected without perceivable effect on the restoration result]. However, if noise is present, two problems arise. First, the noise influence may become significant for frequencies where H(u, v) has small magnitude. This situation usually corresponds to high frequencies u, v. In reality, H(u, v) usually decreases in magnitude much more rapidly than N (u, v) and thus the noise effect may dominate the entire restoration result. Limiting the restoration to a small neighborhood of the u, v origin in which H(u, v) is sufficiently large overcomes this problem, and the results are usually quite acceptable. Secondly, we usually do not have enough information about the noise to determine N (u, v) sufficiently.

2.11.3. Wiener filtering

Wiener (least mean square) filtering [Wiener, 1942; Gonzalez and Woods, 1992; Castle- man, 1996] attempts to take account of noise properties by incorporating a priori know- ledge in the image restoration formula. Restoration by the **Wiener filter** gives an estimate f^ of the original uncorrupted image f with minimal mean square error

$$e^2 = \mathcal{E}\left\{\left(f(i,j) - \hat{f}(i,j)\right)^2\right\} , \quad (5.82)$$

where denotes the mean operator.S If no constraints are applied to the solution of equation (5.82), then an optimal estimate f^ is the conditional mean value of the ideal image f under the condition g. This approach is complicated from the computational point of view. Moreover, the conditional probability density between the optimal image f and the corrupted image g is not usually known. The optimal estimate is in general a non-linear function of the image g.

Minimization of equation (5.82) is easy if the estimate f is^ a linear combination of the values in image g; the estimate f^ is then close (but not necessarily equal) to the theoretical optimum. The estimate is equal to the theoretical optimum only if the stochastic processes describing images f , g, and the noise ν are homogeneous, and their probability density is Gaussian. These conditions are not usually fulfilled for typical images.

Denote the Fourier transform of the Wiener filter by HW . Then, the estimate F^ of the Fourier transform F of the original image f can be obtained as

$$\hat{F}(u,v) = H_W(u,v)\, G(u,v)\ . \tag{5.83}$$

HW is not derived here, but may be found elsewhere [Gonzalez and Woods, 1992] as

$$H_W(u,v) = \frac{H^*(u,v)}{\left|H(u,v)\right|^2 + \left[S_{\nu\nu}(u,v)/S_{ff}(u,v)\right]}\ , \tag{5.84}$$

where H is the transform function of the degradation, denotes complex conjugate, Sνν is the spectral density of the noise, and Sff is the spectral density of the undegraded image.

If Wiener filtering is used, the nature of degradation H and statistical parameters of the noise need to be known. Wiener filtering theory solves the problem of optimal a posteriori linear mean square estimates—all statistics (for example, power spectrum)

should be available in advance. Note the term Sff (u, v) in equation (5.84), which rep- resents the spectrum of the undegraded image, which may be difficult to obtain with no foreknowledge of the undegraded image.

Restoration is illustrated in Figure 5.36 where an image that was degraded by 5 pixels motion in the direction of the x axis: Figure 5.36b shows the result of restoration by Wiener filtering.

(a) (b)

Figure 5.36: Restoration of motion blur using Wiener filtering. Courtesy of P.Kohout, Criminalistic Institute, Prague.

Despite its unquestionable power, Wiener filtering suffers several substantiallimitations. First, the criterion of optimality is based on minimum mean square error and weights all errors equally, a mathematically fully acceptable criterion that unfortunately does not perform well if an image is restored for human viewing. The reason is that humans perceive the restoration errors more seriouslyin constant-graylevel areas and in bright regions, while they are much less sensitive to errors located in dark regions and in high-gradient areas. Second, spatially variant degradations cannot be restored using the standard Wiener filtering approach, and these degradations are common. Third, most images are highly non-stationary, containing large homogeneous areas separated by high- contrast edges. Wiener filtering cannot handle non-stationary signals and noise. To deal with real-life image degradations, more sophisticated approaches may be needed. Examples include **power spectrum equalization** and **geometric meanfiltering**. These and other specialized restoration techniques can be found in higher-level texts devoted to this topic; is well suited for such a purpose.