Chapter 3. Intensity Transformations and Spatial Filtering

Bin Liu
BIN's BIN

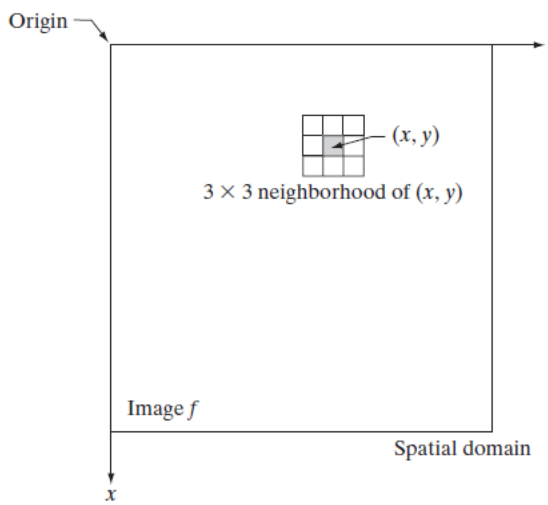
1 Background

1.1 The Basics of Intensity Transformations and Spatial Filtering

The spatial domain processes in this chapter can be denoted by

$$g(x, y) = T[f(x, y)] \tag{1}$$

where $f(x, y)$ is the input image, $g(x, y)$ is the output image, and T is an operator on f defined over a neighborhood of point (x, y). The operator can apply to a single image or to a set of images.



The process in the above figure consists of moving the origin of the neighborhood from pixel to pixel and applying the operator T to the pixels in the neighborhood to yield the output at that location.

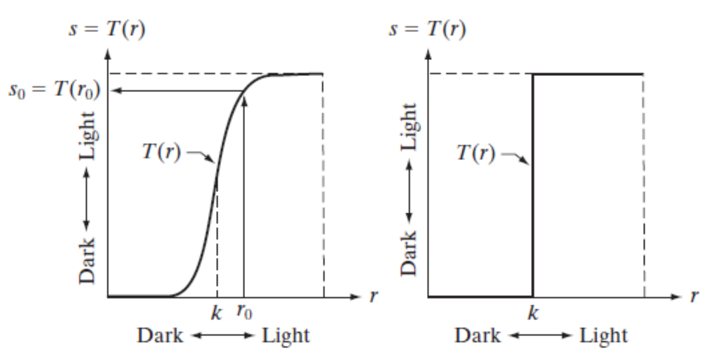
This procedure is called *spatial filtering*, in which the neighborhood, along with a predefined operation, is called a *spatial filter* (also called *spatial mask, kernel, template, or window*).

The smallest possible neighborhood is of size 1×1. Then Equation 1 becomes an *intensity transformation function* of the form

$$s = T(r) \tag{2}$$

where s and r are variables denoting the intensity of g and f at any point (x, y), respectively.

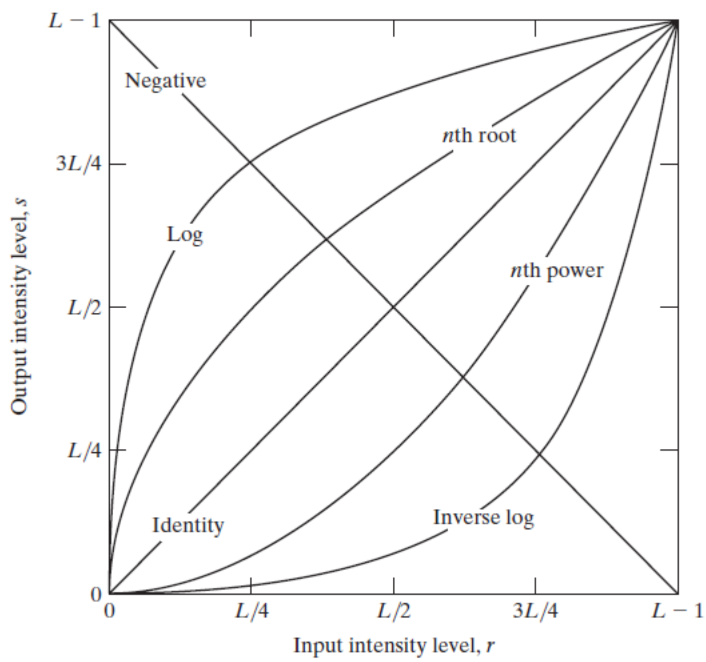
The *contrast streching*, in the left panel, and *thresholding*, in the right panel, are shown below.



2 Some Basic Intensity Transformation Functions

The values of pixels, before and after processing, will be denoted by r and s, respectively, which are related by $s = T(r)$, where T is a transformation.

The figure below shows some basic transformations.



2.1 Image Negatives

The negative of an image with intensity levels in the range $[0, L-1]$ is given by

$$s = L - 1 - r \tag{3}$$

Reversing the intensity levels of an image produces the equivalent of a photographic negative. This is suited for enhancing white or gray detail embedded in dark regions image, especially when the black areas are dominant in size.

2.2 Log Transformations

The general form of the log transformation is

$$s = c\log(1 + r) \tag{4}$$

where c is a constant, and $r \geq 0$. The figure above shows that this transformation maps a narrow range of low intensity values in the input into a wider range of output levels. This type of transformation is used to expand the values of dark pixels in an image while compressing the higher-level values. The inverse log transformation does the opposite.

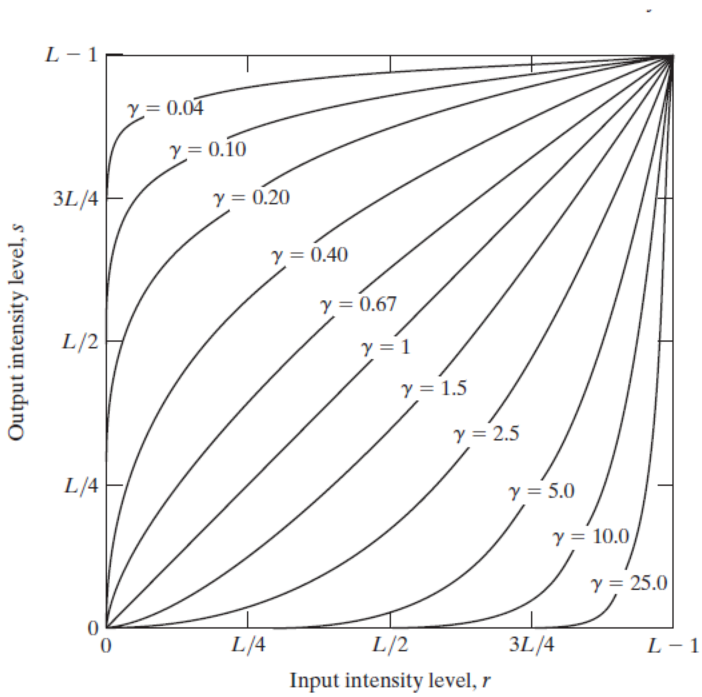
The log function compresses the dynamic range of images with large variations in pixel values.

2.3 Power-Law (Gamma) Transformations

Power-law transformation is given by

$$s = cr^{\gamma} \tag{5}$$

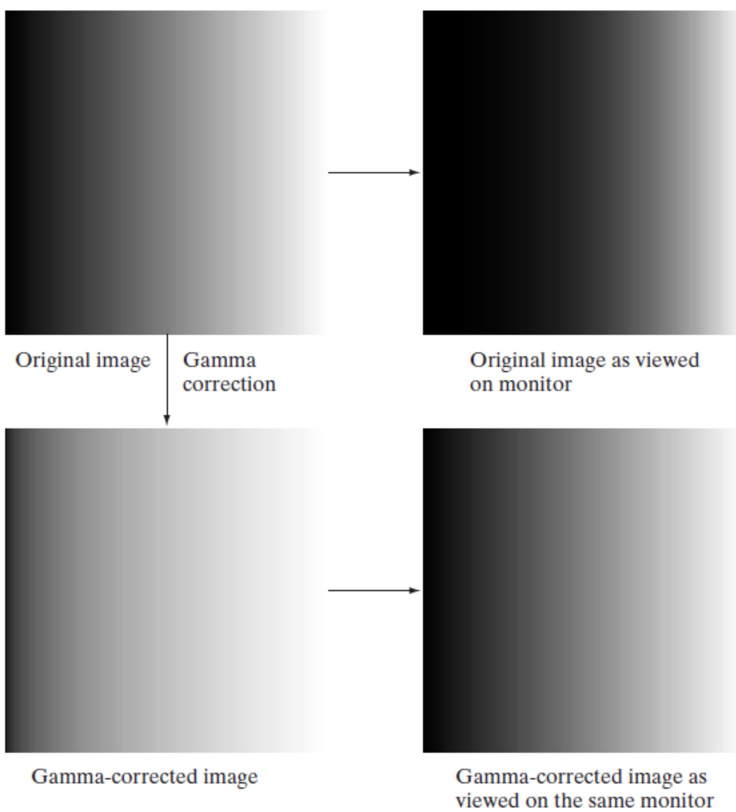
where c and γ are positive constants. Sometimes it can be written as $s = c(r + \epsilon)^{\gamma}$ to account for an offset (a measurable output when the input is zero). The figure below shows various values of γ.



Similar to the log transformation, power-law curves with fractional values of γ map a narrow range of dark input values into a wider range of output values, with the opposite being true for higher values of input levels.

Unlike the log function, however, the curves generated with values of $\gamma > 1$ have exactly the opposite effect as those generated with values of $\gamma < 1$. It reduces to the identity transformation when $c = \gamma = 1$.

The process used to correct the power-law response phenomena (image capture, printing, and display respond) is called *gamma correction*.

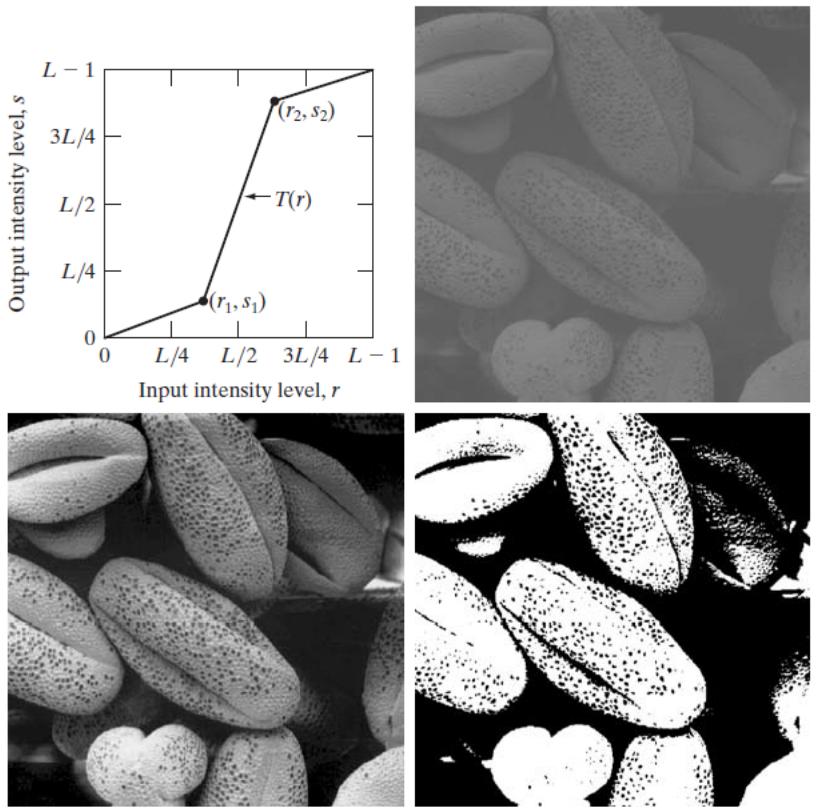


Gamma correction is important if displaying an image accurately on a computer screen is of concern. Images that are not corrected properly can look either bleached out or too dark.Trying to reproduce colors accurately also requires some knowledge of gamma correction because varying the value of gamma changes not only the intensity, but also the ratios of red to green to blue in a color image.

2.4 Piecewise-Linear Transformation Functions

Contrast stretching

Low-contrast images can result from poor illumination, lack of dynamic range in the imaging sensor, or even the wrong setting of a lens aperture during image acquisition. *Contrast stretching* is a process that expands the range of intensity levels in an image so that it spans the full intensity range of the recording medium or display device.

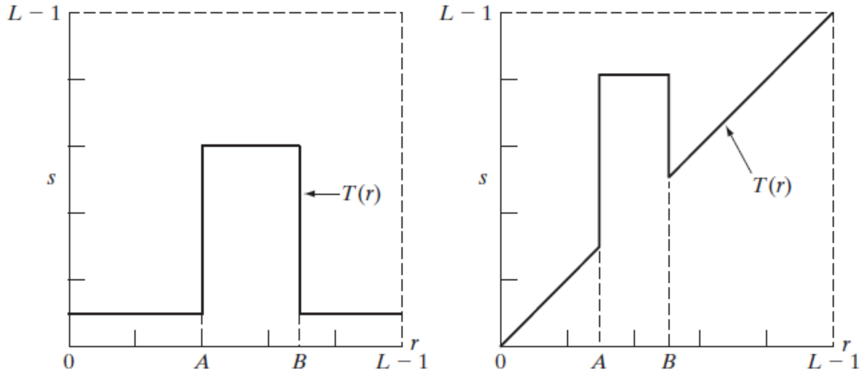


The figure on the top left above shows a typical transformation used for contrast stretching. (r_1, s_1) and (r_2, s_2) control the shape of the transformation function. If $r_1 = s_1$ and $r_2 = s_2$, the transformation is a linear function that produces no changes in intensity levels. If $r_1 = r_2$, $s_1 = 0$ and $s_2 = L - 1$, then the transformation becomes a *thresholding function* that creates a binary image, as shown on top right above.

The top right figure above shows an 8-bit image with low contrast. The bottom left shows the result of contrast stretching with $(r_1, s_1) = (r_{\min}, 0)$ and $(r_2, s_2) = (r_{\max}, L - 1)$. The bottom right shows the result of using the thresholding function defined with $(r_1, s_1) = (m, 0)$ and $(r_2, s_2) = (m, L - 1)$, where m is the mean intensity level in the image.

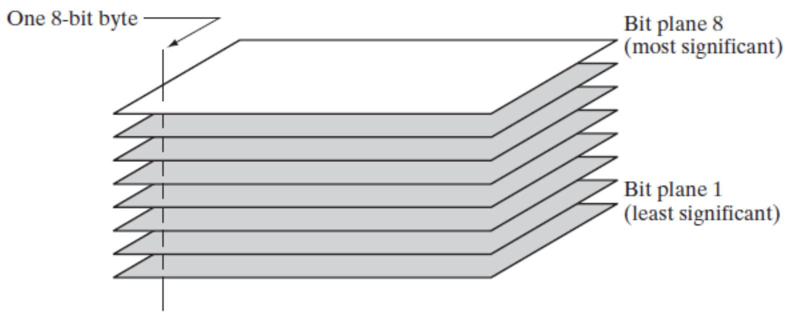
Intensity-level slicing

The *intensity-level slicing* process is used to highlight a specific range of intensities in an image. One approach is to display all the values in the range of interest into one value and all other intensities into another. This transformation shown in the left panel below produces a binary image. Another approach based on the right panel below brightens (or darkens) the desired range of intensities but leaves all other intensity levels in the image unchanged.

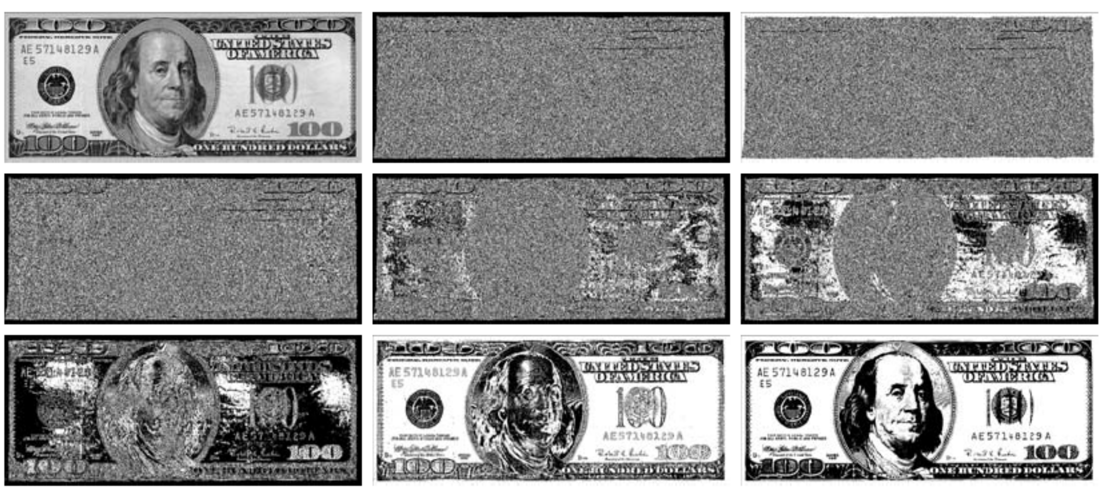


Bit-plane slicing

Pixels are digital numbers composed of bits. The intensity of each pixel in a 256-level gray-scale image is composed of 8 bits (i.e., one byte). Instead of highlighting intensity-level ranges, the specific bits can be highlighted in the image.



The figure above shows an 8-bit image composing of eight 1-bit planes, with plane 1 containing the lowest-order bit of all pixels in the image and plane 8 all the highest-order bits.

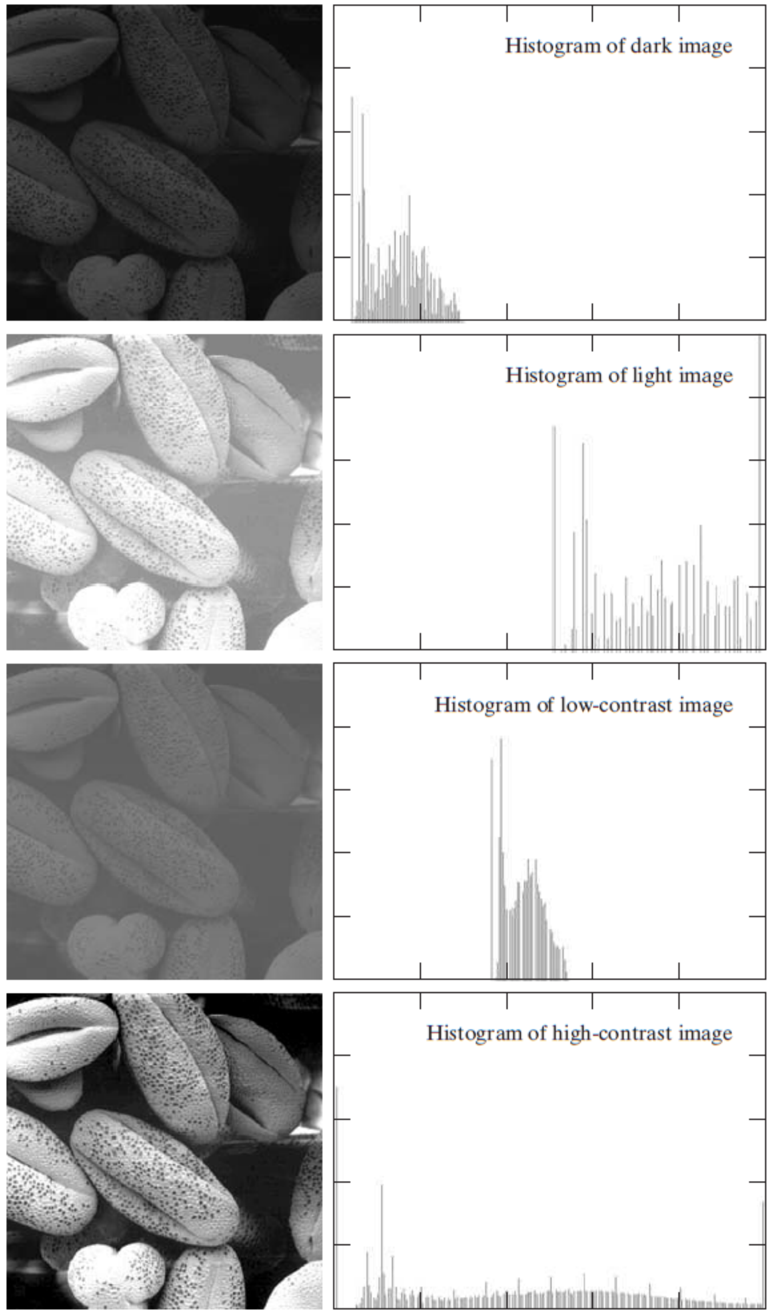


The figure above shows an 8-bit gray-scale image with its eight 1-bit planes from the lowest-order bit to the highest-order bit.

3 Histogram Processing

The *histogram* of a digital image with intensity levels in the range $[0, L-1]$ is a discrete function $h(r_k) = n_k$, where r_k is the kth intensity value and n_k is the number of pixels in the image with intensity r_k. The normalized histogram is also an estimate of the probability of occurrence of intensity level r_k in an image and is given by $p(r_k) = n_k/MN$, for $k = 0, 1, 2, ..., L-1$.

The figure below shows a pollen image in four intensity characteristics: dark, light, low contrast, and high contrast, with the corresponding histograms.



An image whose pixels tend to occupy the entire range of possible intensity levels and tend to be distributed uniformly will have an appearance of high contrast and will exhibit a large variety of gray tones.

3.1 Histogram Equalization

Consider continuous intensity values and let the variable r denote the intensities of an image to be processed, where r is in the range $[0, L-1]$, with $r = 0$ representing black and $r = L-1$ representing white. The intensity transformation is given by

$$s = T(r) \quad 0 \le r \le L-1 \tag{6}$$

assuming that

(a) $T(r)$ is a monotonically increasing function in the interval $0 \le r \le L-1$; and

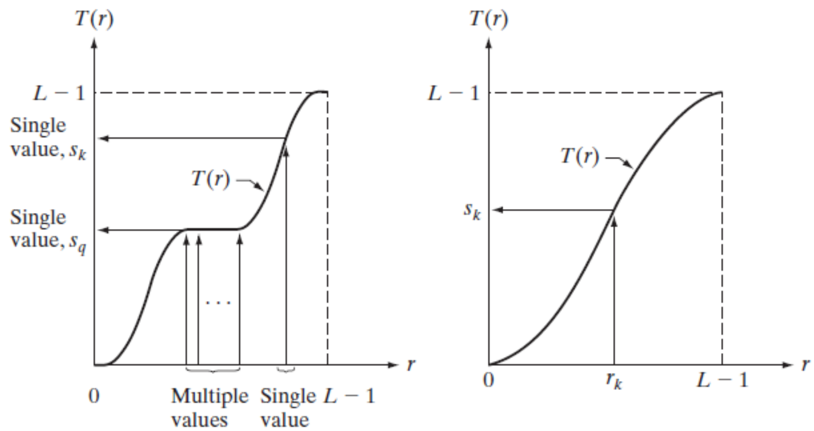
(b) $0 \le T(r) \le L-1$ for $0 \le r \le L-1$

The inverse is

$$r = T^{-1}(s) \quad 0 \le s \le L-1 \tag{7}$$

in which case the condition (a) is changed to (a') $T(r)$ is a strictly monotonically increasing function in the interval $0 \leq r \leq L-1$

The figure below shows a function that satisfies conditions (a) and (b). A monotonic transformation function performs a one-to-one or many-to-one mapping from r to s, but not strictly monotonic from s to r.



The intensity levels in an image may be viewed as random variables in the interval $[0, L-1]$. Let $p_r(r)$ and $p_s(s)$ denote the probability density functions (PDFs) of r and s, respectively. If $p_r(r)$ and $T(r)$ are known, and $T(r)$ is continuous and differentiable over the range of values of interest, then the PDF of the transformed (mapped) variable s can be obtained using

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right| \tag{8}$$

A transformation function used here is given by

$$s = T(r) = (L-1)\int_0^r p_r(w)dw \tag{9}$$

where w is a dummy variable of integration. The right side is the cumulative distribution function (CDF) of random variable r.

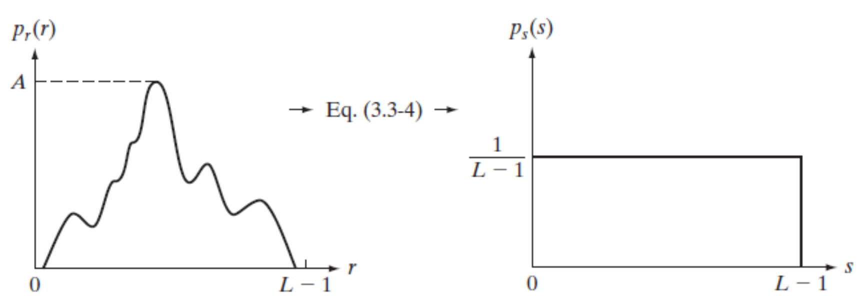
To find $p_s(s)$, apply Leibniz's rule, stating that the derivative of a definite integral with respect to its upper limit is the integrand evaluted at the limit:

$$\begin{aligned} \frac{ds}{dr} &= \frac{dT(r)}{dr} \\ &= (L-1)\frac{d}{dr}\left[\int_0^r p_r(w)dw\right] \\ &= (L-1)p_r(r) \end{aligned} \tag{10}$$

Then substitute,

$$\begin{aligned} p_s(s) &= p_r(r)\left|\frac{dr}{ds}\right| \\ &= p_r(r)\left|\frac{1}{(L-1)p_r(r)}\right| \\ &= \frac{1}{L-1} \quad 0 \leq s \leq L-1 \end{aligned} \tag{11}$$

which is a *uniform* probability density function. This shows that the resulting $p_s(s)$ *always* is uniform, *independently* of the form of $p_r(r)$, shown in the figure below.



For discrete values, probabilities (histogram values) and summations are used instead of PDFs and integrals. The probability of occurrence of intensity level r_k in a digital image is approximated by

$$p_r(r_k) = \frac{n_k}{MN}, \quad k = 0, 1, 2, ..., L-1 \tag{12}$$

where MN is the total number of pixels in the image, n_k is the number of pixels that have intensity r_k, and L is the number of possible intensity levels in the image.

The discrete form of the transformation is

$$\begin{aligned} s_k =& T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) \\ =& \frac{(L-1)}{MN}\sum_{j=0}^{k} n_j \quad k = 0, 1, 2, ..., L-1 \end{aligned} \tag{13}$$

The transformation (mapping) $T(r_k)$ in this equation is called a *histogram equalization* or *histogram linearization* transformation.

Given an image, the process of histogram equalization consists simply of implementing Equation 13, which is based on information that can be extracted directly from the given image, without the need for further paramter specifications.

The *inverse transformation* from s back to r is denoted by

$$r_k = T^{-1}(s_k) \quad k = 0, 1, 2, ..., L-1 \tag{14}$$

3.2 Histogram Matching (Specification)

Histogram equalization automatically determines a transformation function that seeks to produce an output image that has a uniform histogram, but uniform histogram may not be the best approach. The *histogram matching* or *histogram specification* is used to generate a processed image that has a specified histogram.

Consider the continuous intensities r and z, denoted as the input and output images, respectively, and let $p_r(r)$ and $p_z(z)$ denote their corresponding continuous probability density functions. $p_r(r)$ can be estimated from the given input image, while $p_z(z)$ is the *specified* probability density function for the output image.

Let s be a random variable with the property

$$s = T(r) = (L-1)\int_0^r p_r(w)dw \tag{15}$$

which is used for histogram equalization for continuous variables.

A random variable z is defined with the property

$$G(z) = (L-1)\int_0^z p_z(t)dt = s \tag{16}$$

where t, same as w, is a dummy variable of integration. Thus, z must satisfy the condition

$$z = G^{-1}[T(r)] = G^{-1}(s) \tag{17}$$

$T(r)$ can be obtained once $p_r(r)$ has been estimated from the input image. $G(z)$ can be obtained from the given $p_z(z)$:

1. Obtain $p_r(z)$ from the input image and use Equation 15 to obtain the values of s.

2. Use the specified PDF in Equation 16 to obtain the transformation function $G(z)$.

3. Obtain the inverse transformation $z = G^{-1}(s)$.

4. Obtain the output image by first equalizing the input image using Equation 15; the pixel values in this image are the s values. Perform the inverse mapping $z = G^{-1}(s)$ to obtain the corresponding pixel in the output image for each pixel with values s in the equalized image. The PDF of the output image is the specified PDF when all pixels have been processed.

The discrete formation is given by

$$\begin{aligned} s_k = T(r_k) =& (L-1)\sum_{j=0}^{k} p_r(r_j) \\ =& \frac{(L-1)}{MN}\sum_{j=0}^{k} n_j \quad k = 0, 1, 2, ..., L-1 \end{aligned} \tag{18}$$

Similarly, given a specific value of s_k, the discrete formation of Equation 16 is given by

$$G(z_q) = (L-1)\sum_{i=0}^{q} p_z(z_j) \tag{19}$$

for a value of q, so that

$$G(z_q) = s_k \tag{20}$$

where $p_z(z_i)$ is the ith value of the specified histogram. The desired value z_q is obtain by the inverse transformation:

$$z_q = G^{-1}(s_k) \tag{21}$$

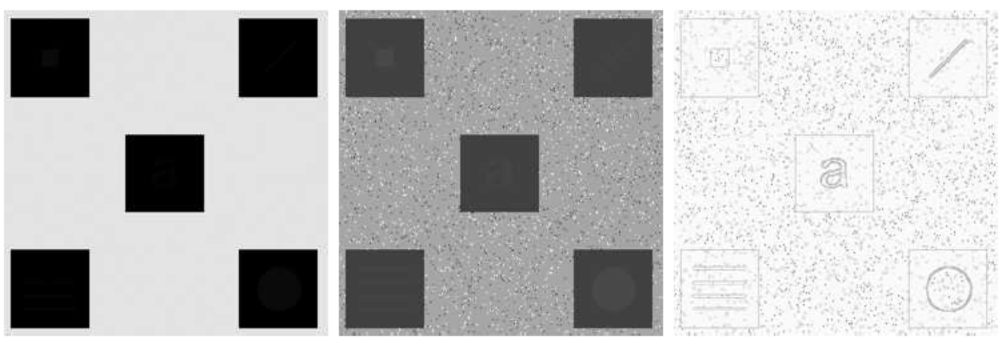
Histogram-specification procedure:

1. Compute the histogram $p_r(r)$ of the given image, and use it to find the histogram equalization transformation in Equation 18. Round the resulting values, s_k, to the integer range $[0, L-1]$.
2. Compute all values of the transformation function G using Equation 19 for $q = 0, 1, 2, ..., L-1$, where $p_z(z_i)$ are the values of the specified histogram. Round the values of G to integers in the range $[0, L-1]$. Store the values of G in a table.
3. For every value of $s_k, k = 0, 1, 2, ..., L-1$, use the store values of G from step 2 to find the corresponding value of z_q so that $G(z_q)$ is closest to s_k and store these mappings from s to z. When more than one value of z_q satisfies the given s_k, choose the smallest value by convention.
4. Form the histogram-specified image by first histogram-equalizing the input image and then mapping every equalized pixel value, s_k, of this image to the corresponding value z_q in the histogram-specified image using the mapping found in step 3.

3.3 Local Histogram Processing

The histogram processing techniques previously described are easily adapted to local enhancement.

The procedure is to define a neighborhood and move its center from pixel to pixel. At each location, the histogram of the points in the neighborhood is computed and either a histogram equalization or histogram specification transformation function is obtained, which used to map the intensity of the pixel centered in the neighborhood.



The left figure above is the original image, the middle one is the result of global histogram equalization, and the right is the result of local histogram equalization using a heighborhood of size 3×3.

3.4 Using Histogram Statistics for Image Enhancement

Let r denote a discrete random variable representing intensity values in the range $[0, L-1]$, and let $p(r_i)$ denote the normalized histogram component corresponding to value r_i. Then, $p(r_i)$ may be viewed as an estimate of the probability that intensity r_i in the image from which the histogram was obtained.

The nth moment of r about its mean is denoted as

$$\mu_n(r) = \sum_{i=0}^{L-1} (r_i - m)^n p(r_i) \tag{22}$$

where m is the mean (average intensity) value of r (i.e., the average intensity of the pixels in the image):

$$m = \sum_{i=0}^{L-1} r_i p(r_i) \tag{23}$$

The second moment is defined as

$$\mu_2(r) = \sum_{i=0}^{L-1} (r_i - m)^2 p(r_i) \tag{24}$$

which is the intensity variance, denoted by σ^2. Whereas the mean is a measure of average intensity, the variance is a measure of contrast in an image.

The estimates of the mean and variance are called *sample mean* and *sample variance*, which are given by

$$m = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \tag{25}$$

and

$$\sigma^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x,y) - m]^2 \tag{26}$$

The *global* mean and variance are computed over an entire image and are useful for gross adjustments in overall intensity and contrast. The *local* mean and variance are used as the basis for making changes that depend on image characteristics in a neighborhood about each pixel in an image.

Let (x,y) denote the coordinates of any pixel in a given image, and let S_{xy} denote a neighborhood (subimage) of specified size, centered on (x,y). The mean value of the pixels in this neighborhood is given by

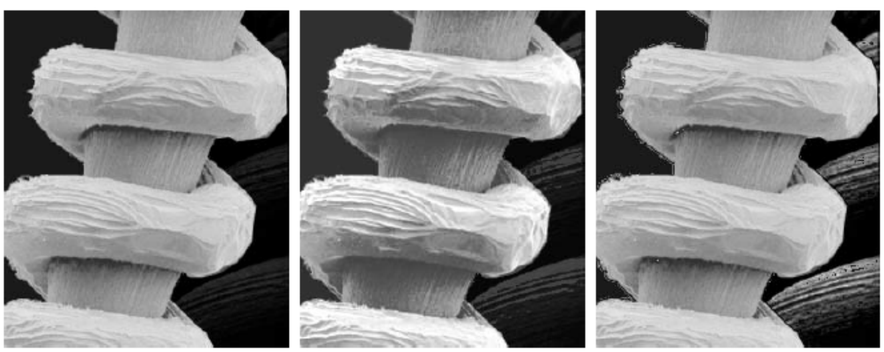
$$m_{S_{xy}} = \sum_{i=0}^{L-1} r_i p_{S_{xy}}(r_i) \tag{27}$$

where $p_{S_{xy}}$ is the histogram of the pixels in region S_{xy}. This histogram has L components, corresponding to the L possible intensity values in the input image, but many of the components are 0 depending on the size of S_{xy}.

The variance of the pixels in the neighborhood similarly is given by

$$\sigma^2_{S_{xy}} = \sum_{i=0}^{L-1} (r_i - m_{S_{xy}})^2 p_{S_{xy}}(r_i) \tag{28}$$

The local mean is a measure of average intensity in neighborhood S_{xy}, and the local variance is a measure of intensity contrast in that neighborhood.



The figure above is an SEM (scanning electron microscope) image of a tungsten filament wrapped around a support. The left one is the original image, the middle one is the result of global histogram equalization, and the right one is the image enhanced using local histogram statistics. The goal is to enhance dark areas in the original image (left) while leaving the light area as unchanged as possible because it does not require enhancement.

A measure of whether an area is relatively light or dark at a point (x,y) is to compare the average local intensity $m_{S_{xy}}$ to the average image intensity, m_G, called the *global mean*. Consider the pixel at a point (x,y) as a candidate for processing if $m_{S_{xy}} \le k_0 m_G$, where k_0 is a positive constant with value less than 1.0.

To focus on the area having low contrast, consider the pixel at a point (x,y) as a candidate for enhancement if $\sigma_{S_{xy}} \le k_2 \sigma_G$, where σ_G is the *global standard deviation* and k_2 is a positive constant, which is greater than 1.0 if interested in enhancing light areas and less than 1.0 for dark areas.

Finally, the lowest values of contrast need to be restricted to avoid enhancing constant areas. Set a lower limit on the local standard deviation by requiring that $k_1 \sigma_G \le \sigma_{S_{xy}}$, with $k_1 < k_2$. A pixel at (x,y) that meets all the conditions for local enhancement is processed by multiplying it by a specified constant, E, to increase (or decrease) the value of its intensity level relative to the rest of the image.

Procedure Summary

Let $f(x,y)$ represent the value of an image at any image coordinates (x,y), and let $g(x,y)$ represent the corresponding enhanced value at those coordinates. Then

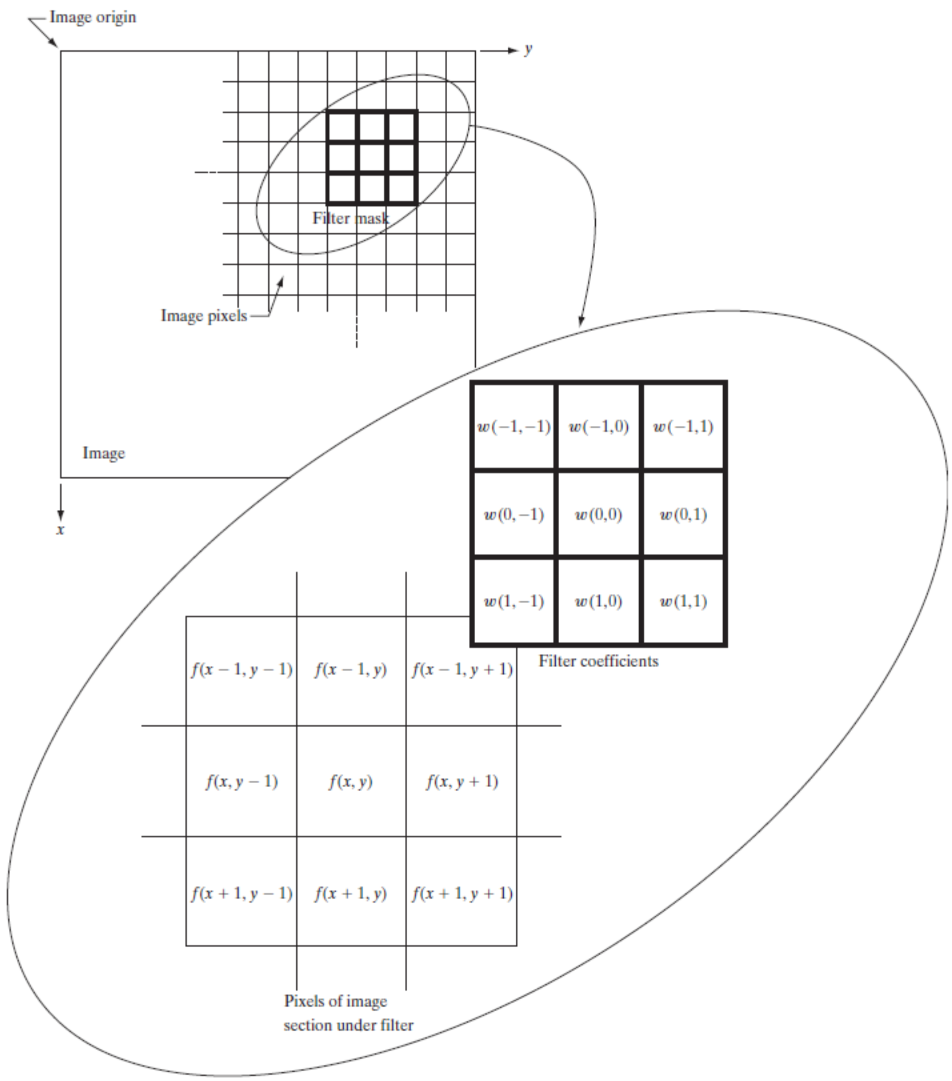
$$g(x,y) = \begin{cases} E \cdot f(x,y) & \text{if } m_{S_{xy}} \le k_0 m_G \text{ AND } k_1\sigma_G \le \sigma_{S_{xy}} \le k_2\sigma_G \\ f(x,y) & \text{otherwise} \end{cases} \tag{29}$$

The values of parameters requires a bit of experimentation with a given image or class of images.

4 Fundamentals of Spatial Filtering

4.1 The Mechanics of Spatial Filtering

A processed (filtered) image is generated as the center of the filter visits each pixel in the input image. If the operation performed on the image pixels is linear, then the filter is called a *linear spatial filter*. Otherwise, the filter is *nonlinear*.



The figure above illustrates the mechanics of linear spatial filtering using a 3×3 neighborhood. At any point (x,y) in the image, the response, $g(x,y)$, of the filter is the sum of products of the filter coefficients and the image pixels encompassed by the filter:

$$g(x,y) = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + ... + w(0,0)f(x,y) + ... + w(1,1)f(x+1,y+1)$$

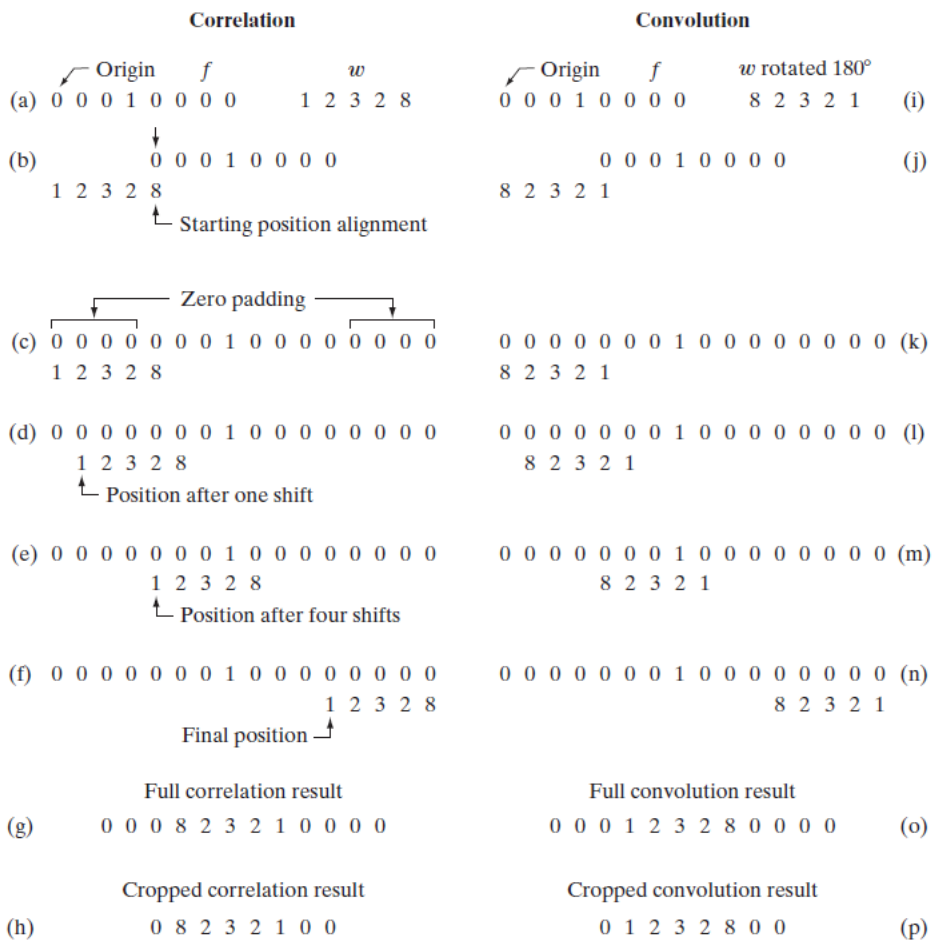
For a mask of size $m \times n$, $m = 2a+1$ and $n = 2b+1$, where a and b are positive integers. In general, linear spatial filtering of an image of size $M \times N$ with a filter of size $m \times n$ is given by

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t) \tag{30}$$

where x and y are varied so that each pixel in w visits every pixel in f.

4.2 Spatial Correlation and Convolution

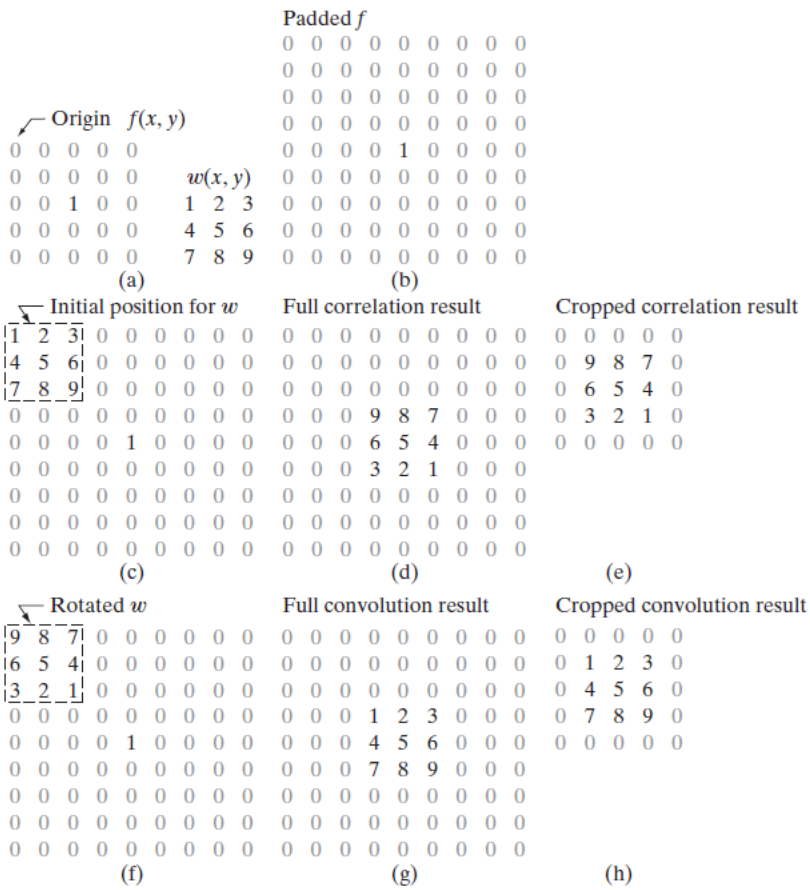
Correlation is the process of moving a filter mask over the image and computing the sum of products at each location. *Convolution* is the same except that the filter is first rotated by 180 degrees.



The figure above shows the 1-D case of correlation and convolution.

- Correlation is a function of *displacement* of the filter
- Correlating a filter w with a function that contains all 0s and a single 1 yields a result that is a *copy* of w, but *rotated* by 180 degrees.

A function that contains a single 1 with the rest being 0s is called a *discrete unit impulse*. Correlation of a function with a discrete unit impulse yields a rotated version of the function at the location of the impulse.



The figure above shows the general 2-D case.

The correlation of a filter $w(x,y)$ of size $m \times n$ with an image $f(x,y)$ is denoted as $w(x,y) \,☆\, f(x,y)$ and given by

$$w(x,y) \,☆\, f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t) \tag{31}$$

which is evaluated for all values of the displacement variables x and y so that all elements of w visit every pixel in f, where f is assumed to be padded appropriately.

The convolution of $w(x,y)$ and $f(x,y)$, denoted by $w(x,y) \bigstar f(x,y)$, is given by

$$w(x,y) \bigstar f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t) \tag{32}$$

where the minus sign on the right flip f (i.e., rotate it by 180 degrees).

4.3 Vector Representation of Linear Filtering

The characteristic response R of a mask is defined as

$$\begin{aligned} R =& w_1 z_1 + w_2 z_2 + \ldots + w_{mn} z_{mn} \\ =& \sum_{k=1}^{mn} w_k z_k \\ =& \boldsymbol{w}^T \boldsymbol{z} \end{aligned} \tag{33}$$

where ws are the coefficients of an $m \times n$ filter and the zs are the corresponding image intensities encompassed by the filter. For example, the figure below shows a general 3×3 mask with coefficients labeled. In this case,

$$\begin{aligned} R =& w_1 z_1 + w_2 z_2 + \ldots + w_9 z_9 \\ =& \sum_{k=1}^{9} w_k z_k \\ =& \boldsymbol{w}^T \boldsymbol{z} \end{aligned} \tag{34}$$

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

5 Smoothing Spatial Filters

5.1 Smoothing Linear Filters

The output (response) of a smoothing, linear spatial filter is the average of the pixels contained in the neighborhood of the filter mask, sometimes called *averaging filters*, also referred to a *lowpass filters*.

The smoothing filters reduce "sharp" transitions in intensities. However, edges also are characterized by sharp intensity transitions, so averaging filters have the undesirable side effect that they blur edges.

The left figure below shows a standard average of the pixels under the mask, given by

$$R = \frac{1}{9}\sum_{i=1}^{9} z_i \tag{35}$$

which is the average of the intensity levels of the pixels in the 3×3 neighborhood defined by the mask.

$$\frac{1}{9} \times \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \qquad \frac{1}{16} \times \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

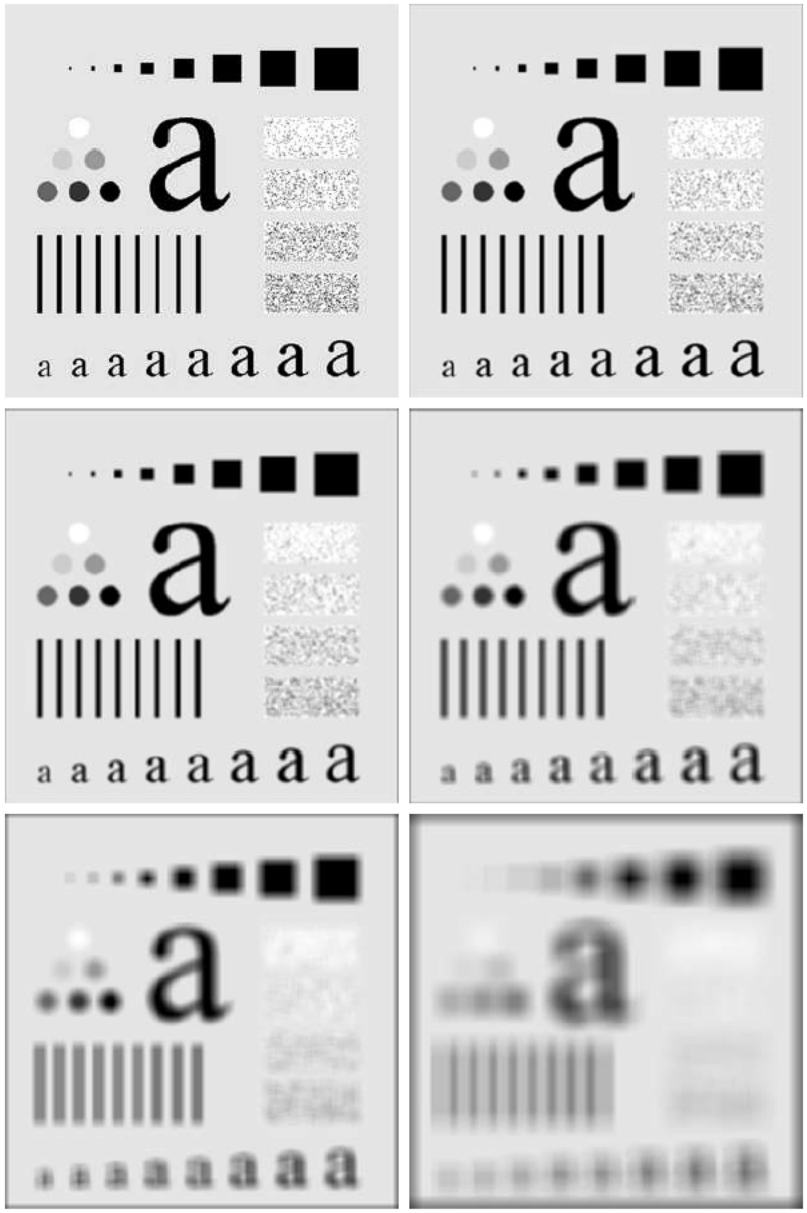
The right figure above is a *weighted average*. This is used to reduce blurring in the smoothing process by assigning higher weights in the center of the mask.

The general implementation for filtering an $M \times N$ image with a weighted averaging filter of size $m \times n$ (m and n odd) is given by

$$g(x,y) = \frac{\sum_{s=-1}^{a}\sum_{t=-b}^{b} w(s,t) f(x+s, y+t)}{\sum_{s=-1}^{a}\sum_{t=-b}^{b} w(s,t)} \tag{36}$$

for $x = 0,1,2,...,M-1$ and $y = 0,1,2,...,N-1$.

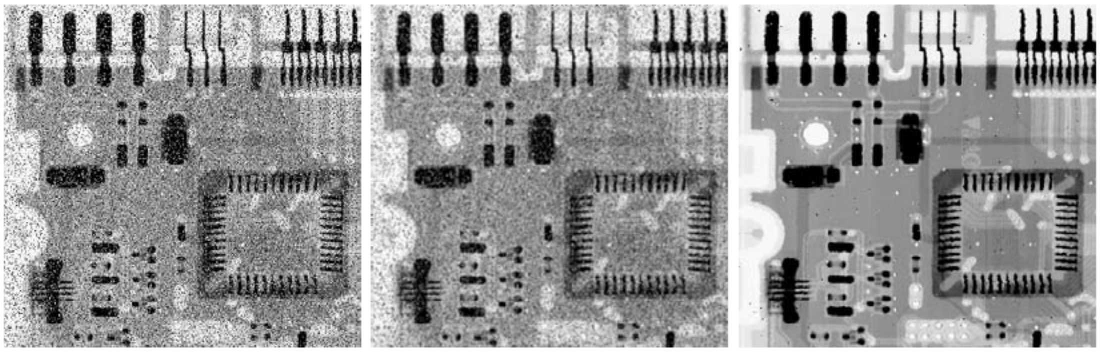
The figure below shows the effects of smoothing as a function of filter size.



5.2 Order-Statistic (Nonlinear) Filters

Order-statistic filters are nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the center pixel with the value determined by the ranking result.

The *mediam filters* are effective in the presence of *impulse noise*, also called *salt-and-pepper noise* because of its appearance as white and black dots superimposed on an image. The left figure below shows a X-ray image of circuit board corrupted by salt-and-pepper noise, the middle one is the result of noise reduction with a 3×3 averaging mask, and the right oen is the noise reduction with a 3×3 median filter.



The *max filter* is useful for finding the brightest points in an image. The *min filter* is used for the opposite purpose.

6 Sharpening Spatial Filters

The principal objective of sharpening is to highlight transitions in intensity. Because averaging is analogous to integration, sharpening can be accomplished by spatial differentiation.

6.1 Foundation

The derivatives of a digital function are defined in terms of differences.

The requirements for a *first derivative*

- must be zero in areas of constant intensity;
- must be nonzero along ramps.

The requirements for a *second derivative*

- must be zero in constant areas;
- must be nonzero at the onset and end of an intensity step or ramp;
- must be zero along ramps of constant slope.

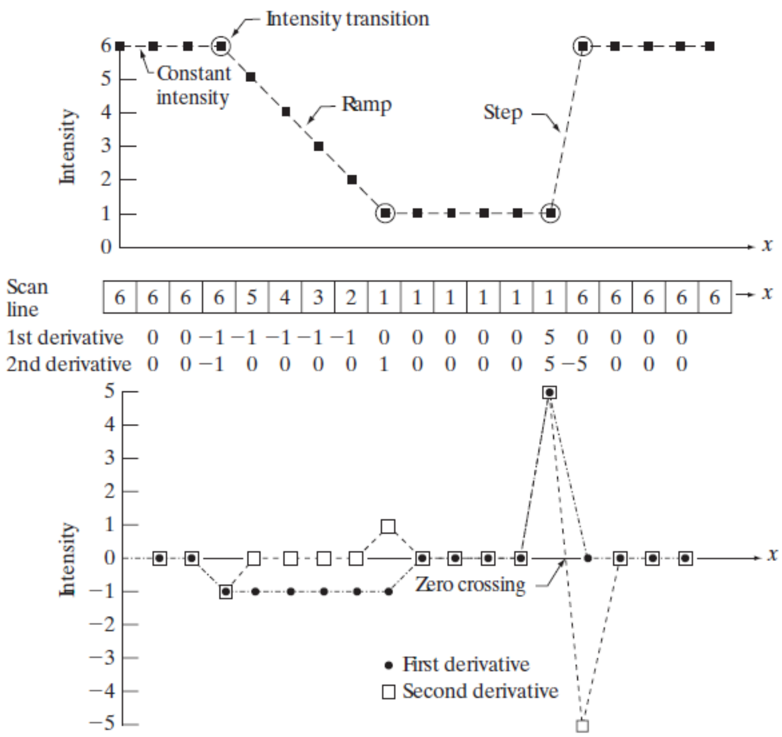
The definition of the first-order derivative of a one-dimensional function $f(x)$ is given by

$$\frac{\partial f}{\partial x} = f(x+1) - f(x) \tag{37}$$

The second-order derivative of $f(x)$ is defined as

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x) \tag{38}$$

The example of first and second derivatives on 1-D function is shown below.



The *zero crossing* property is useful for locating edges. Edges in digital images are ramp-like transitions in intensity, in which case the first derivative of the image would result in thick edges. The second derivative would produce a double edge on pixel thick, separated by zeros.

6.2 Using the Second Derivative for Image Sharpening - The Laplacian

The *isotropic* filters has a response independent of the direction of the discontinuities in the image to which the filter is applied. In other words, isotropic filters are *rotation invariant*, in the sense that rotating the image and then applying the filter gives the same result as applying the filter to the image first and then rotating the result.

The simplest isotropic derivative operator is the Laplacian, which is defined as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{39}$$

Because derivatives of any order are linear operations, the Laplacian is a linear operator. The discrete form in x-direction is given by

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y) \tag{40}$$

and in the y-direction,

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) + 2f(x, y) \tag{41}$$

Therefore, the discrete Laplacian,

$$\nabla^2 f(x, y) = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y) \tag{42}$$

The figure below also shows various Laplacian masks.

0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

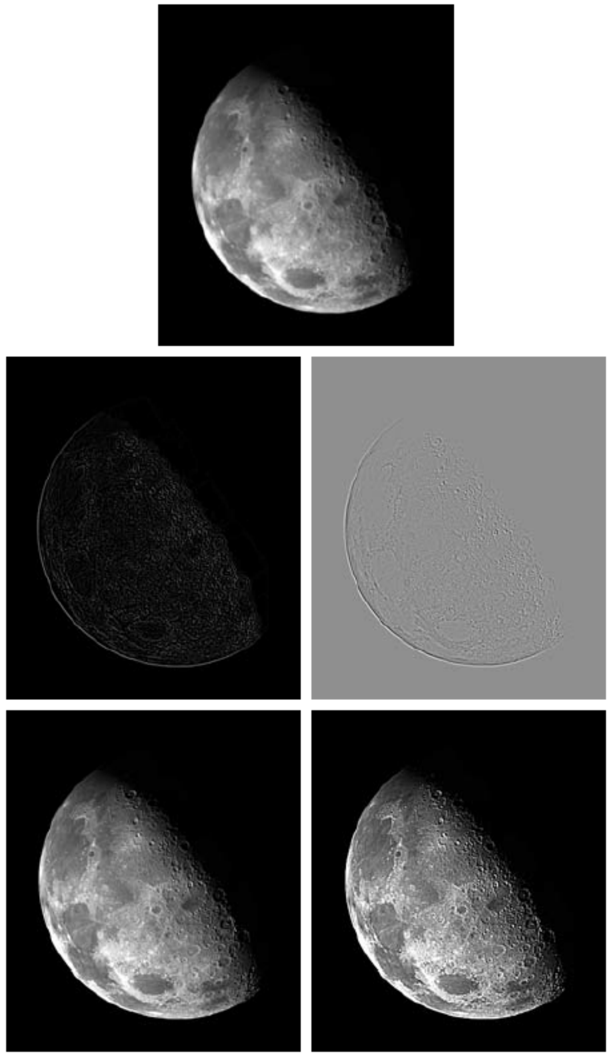
Because the Laplacian is a derivative operator, its use highlights intensity discontinuities in an image and deemphasizes regions with slowly varying intensity levels.

The basic way to use the Laplacian for image sharpening is

$$g(x,y) = f(x,y) + c\left[\nabla^2 f(x,y)\right] \tag{43}$$

where $f(x,y)$ and $g(x,y)$ are the input and sharpened images, respectively. The constant is $c = -1$ if the Laplacian filters shown as the top two filters in the figure above are used, and $c = 1$ if the bottom two filters are used.

A typical way to scale a Laplacian image is to add to it its minimum value to bring the new minimum to zero and then scale the result to the full $[0, L-1]$ intensity range.



The first row in the figure above is the original image. The second row shows the image without and with scaling, respectively. The grayish appearance is typical of Laplacian images that have been scaled properly. The bottom left panel shows the result obtained with $c = -1$. The bottom right panel is the result of repeating the preceding procedure.

6.3 Unsharp Masking and Highboost Filtering

The *unsharp masking* is to sharpen images by substrcting an unsharped (smoothed) version of an image from the original image:

1. Blur the original image.
2. Subtract the blurred image from the original (the resulting difference is called the *mask*)
3. Add the mask to the original

Letting $\overline{f}(x,y)$ denote the blurred image. The unsharp masking starts with obtaining the mask:

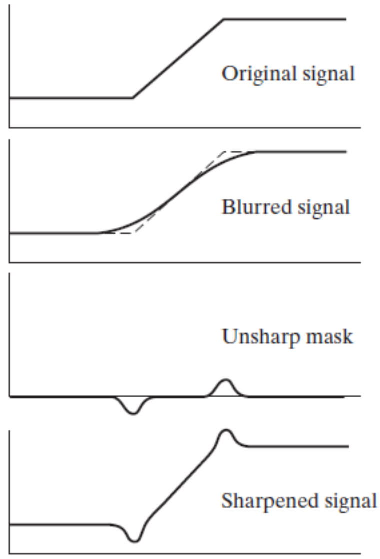
$$g_{\text{mask}}(x,y) = f(x,y) - \overline{f}(x,y) \tag{44}$$

Then a weighted portion of the mask is added back to the original image:

$$g(x,y) = f(x,y) + k * g_{\text{mask}}(x,y) \tag{45}$$

where a weight $k (k \geq 0)$ is for generality. When $k = 1$, the unsharp masking is obtained as defined. When $k > 1$, the process is referred to as *highboost filtering.* $k < 1$ de-emphasizes the contribution of the unsharp mask.

The figure explains how unsharp masking works.



6.4 Using First-Order Derivatives for (Nonlinear) Image Sharpening - The Gradient

For a function $f(x, y)$, the gradient of f at coordinates (x, y) is defined as the two-diemsnional column *vector*

$$\nabla f = \text{grad}(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{46}$$

which points in the direction of the greatest rate of change of f at location (x, y).

The *magnitude (length)* of vector ∇f, denoted as $M(x, y)$, where

$$M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2} \tag{47}$$

is the value at (x, y) of the rate of change in the direction of the gradient vector. $M(x, y)$ is the *gradient image*. In some implementations, it is more suitable computationally to approximate the squares and square root operations by absolute values:

$$M(x, y) \approx |g_x| + |g_y| \tag{48}$$

The following figure is used to denote the intensities of image points in a 3×3 region.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

From this notation above, the simplest approximations to a first-order derivative that satisfy the conditions stated in the previous section are $g_x = (z_8 - z_5)$ and $g_y = (z_6 - z_5)$. The cross differences can also be used:

$$g_x = (z_9 - z_5) \quad \text{and} \quad g_y = (z_8 - z_6) \tag{49}$$

then the gradient image is computed as

$$M(x, y) = \left[(z_9 - z_5)^2 + (z_8 - z_6)^2\right]^{1/2} \tag{50}$$

For approximation,

$$M(x, y) \approx |z_9 - z_5| + |z_8 - z_6| \tag{51}$$

The masks figure below are referred to as the *Roberts cross-gradient operators*.

−1	0
0	1

0	−1
1	0

Approximations to g_x and g_y using a 3×3 neighborhood centered on z_5 are

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z + 8 + z_9) - (z_1 + 2z_2 + z_3) \tag{52}$$

and

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \tag{53}$$

which are shown in the figure below, where the left one is in x-direction and the right one in y-direction.

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

These masks are called the *Sobel operators*. The magnitude of the gradient,

$$M(x, y) \approx |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)| \tag{54}$$

7 Using Fuzzy Techniques for Intensity Transformations and Spatial Filtering

7.1 Introduction

A *set* is a collection of objects (elements) and *set theory* is the set of tools that deals with operations on and among sets.

7.2 Principles of Fuzzy Set Theory

Definitions

Let Z be a set of elements (objects), with a generic element of Z denoted by z; that is, $Z = \{z\}$, which is called the *universe of discourse*. A *fuzzy set* A in Z is characterized by a *membership function*, $\mu_A(z)$, that associates with each element of Z a read number in the interval $[0, 1]$. The value of $\mu_A(z)$ at z represents the *grade of membership* of z in A. The nearer the value of $\mu_A(z)$ is to unity, the higher the membership grade of z in A. With fuzzy sets, all zs for which $\mu_A(z) = 1$ are *full* members of the set, all zs for which $\mu_A(z) = 0$ are *not* members of the set, and all zs for which $\mu_A(z)$ is between 0 and 1 have *partial* membership in the set. Therefore, a fuzzy set is an *ordered pair* consisting of values of z and a corresponding membership function that assigns a grade of membership to each z. That is,

$$A = \{z, \mu_A(z) | z \in Z\} \tag{55}$$

Empty set: A fuzzy set is *empty* if and only if its membership function is identically zero in Z.

Equality: Two fuzzy sets A and B are *equal*, written $A = B$, if and only if $\mu_A(z) = \mu_B(z)$ for all $z \in Z$.

Complement: The *complement* (NOT) of a fuzzy set A, denoted by $\overline{A}$, or NOT(A), is defined as the set whose membership function is

$$\mu_{\overline{A}}(z) = 1 - \mu_A(z) \tag{56}$$

for all $z \in Z$.

Subset: A fuzzy set A is a *subset* of a fuzzy set B if and only if

$$\mu_A(z) \le \mu_B(z) \tag{57}$$

for all $z \in Z$.

Union: The *union* (OR) of two fuzzy sets A and B, denoted $A \cup B$, or A OR B, is a fuzzy set U with membership function

$$\mu_U(z) = \max[\mu_A(z), \mu_B(z)] \tag{58}$$

for all $z \in Z$.

Intersection: The *intersection* (AND) of two fuzzy sets A and B, denoted $A \cap B$, or A AND B, is a fuzzy set I with membership function

$$\mu_I(z) = \min[\mu_A(z), \mu_B(z)] \tag{59}$$

for all $z \in Z$.

Some common membership functions

Triangular:

$$\mu(z) = \begin{cases} 1 - (a - z)/b & a - b \le z < a \\ 1 - (z - a)/c & a \le z \le a + c \\ 0 & \text{otherwise} \end{cases} \tag{60}$$

Trapezoidal:

$$\mu(z) = \begin{cases} 1 - (a - z)/c & a - c \le z < a \\ 1 & a \le z < b \\ 1 - (z - b)/d & b \le z \le b + d \\ 0 & \text{otherwise} \end{cases} \tag{61}$$

Sigma:

$$\mu(z) = \begin{cases} 1 - (a - z)/b & a - b \le z \le a \\ 1 & z > a \\ 0 & \text{otherwise} \end{cases} \tag{62}$$

S-shape:

$$S(z; a, b, c) = \begin{cases} 0 & z < a \\ 2\left(\frac{z-a}{c-a}\right)^2 & a \le z \le b \\ 1 - 2\left(\frac{z-c}{c-a}\right)^2 & b < z \le c \\ 1 & z > c \end{cases} \tag{63}$$

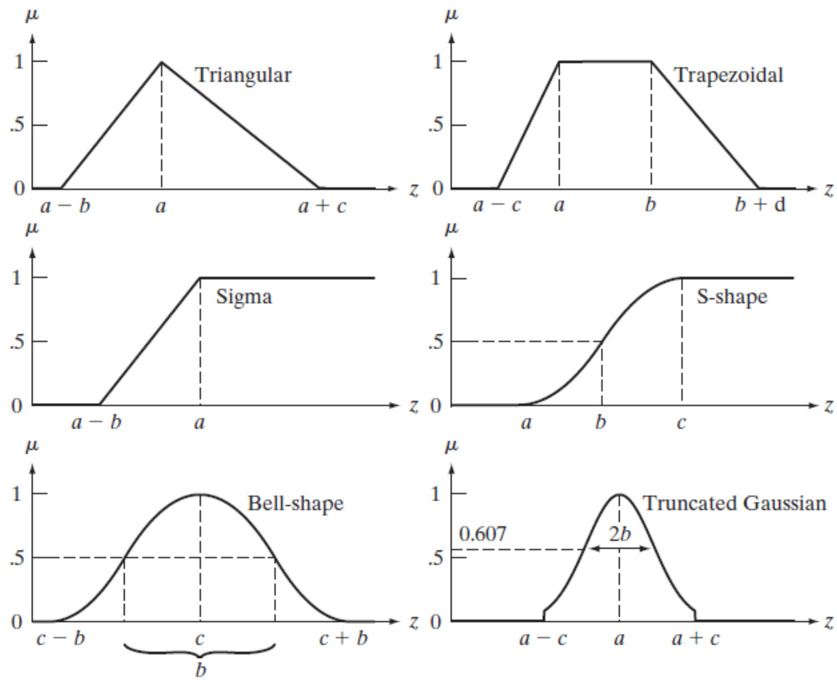
Bell-shape:

$$\mu(z) = \begin{cases} S(z; c - b, c - b/2, c) & z \le c \\ 1 - S(z; c, c + b/2, c + b) & z > c \end{cases} \tag{64}$$

Truncated Gaussian:

$$\mu(z) = \begin{cases} e^{-\frac{(z-a)^2}{2b^2}} & a - c \le z \le a + c \\ 0 & \text{otherwise} \end{cases} \tag{65}$$

The figure below shows the above membership functions as examples.



7.3 may revisit someday