COMPUTER VISION

Course 3

Anca Ignat
Faculty of Computer Science
Alexandru Ioan Cuza University of Iași

Probabilistic Methods

z_i = the values of all possible intensities in an $M \times N$ digital image, $i=0, 1, \ldots, L-1$

$p(z_k)$ = the probability that the intensity level z_k occurs in the given image

$$p(z_k) = \frac{n_k}{M \cdot N}$$

n_k = the number of times that intensity z_k occurs in the image (MN is the total number of pixels in the image)

$$\sum_{k=0}^{L-1} p(z_k) = 1$$

The mean (average) intensity of an image is given by:

$$m = \sum_{k=0}^{L-1} z_k \, p(z_k)$$

The variance of the intensities is:

$$\sigma^2 = \sum_{k=0}^{L-1} (z_k - m)^2 \, p(z_k)$$

The variance is a measure of the spread of the values of ***z*** about the mean, so it is a measure of image contrast. Usually, for measuring image contrast the standard deviation (σ) is used.

The ***n***-th moment of a random variable ***z*** about the mean is defined as:

$$\mu_n(z) = \sum_{k=0}^{L-1} (z_k - m)^n \, p(z_k)$$

$$(\mu_0(z) = 1 \,, \;\; \mu_1(z) = 0 \;\; , \;\; \mu_2(z) = \sigma^2)$$

$\mu_3(z) > 0 \rightarrow$ the intensities are biased to values higher than the mean;

$\mu_3(z) < 0 \rightarrow$ the intensities are biased to values lower than the mean;

$\mu_3(z) = 0 \rightarrow$ the intensities are distributed approximately equally on both side of the mean

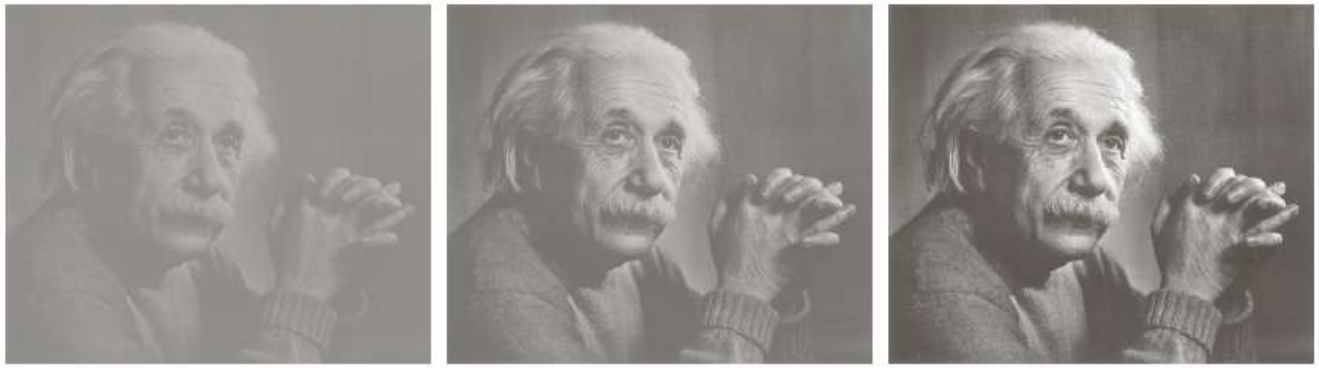


Fig.1 (a) Low contrast (b) medium contrast (c) high contrast

Figure 1(a) – standard deviation $\sigma = 14.3$ (variance = ***204.5***)

Figure 1(b) – standard deviation $\sigma = 31.6$ (variance = ***998.6***)

Figure 1(c) – standard deviation $\sigma = 49.2$ (variance = ***2420.6***)

Intensity Transformations and Spatial Filtering

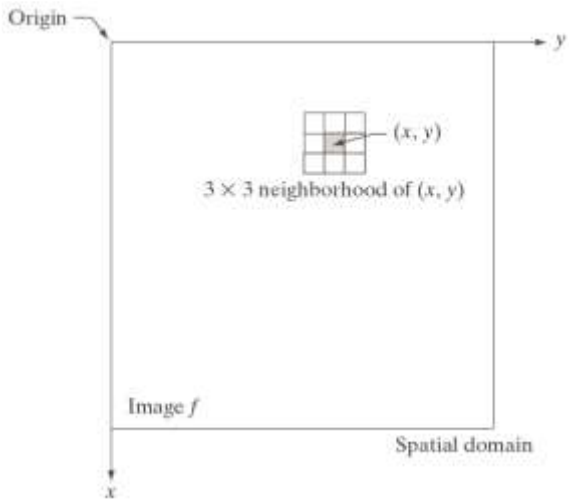
$$g(x,y) = T\left[f(x,y)\right]$$

f(x, y) – input image ,

g(x, y) – output image ,

T – an operator on ***f*** defined over a neighborhood of ***(x, y)***.

- the neighborhood of the point ***(x, y)***, S_{xy} usually is rectangular, centered on ***(x, y)***, and much smaller in size than the image



- *spatial filtering*, the operator ***T*** (the neighborhood and the operation applied on it) is called *spatial filter (spatial mask, kernel, template* or *window)*

$S_{xy} = \{(x, y)\}$ → ***T*** becomes an *intensity (gray-level* or *mapping) transformation function*

$$s = T(r)$$

s and ***r*** are denoting, respectively, the intensity of ***g*** and ***f*** at ***(x, y)***.

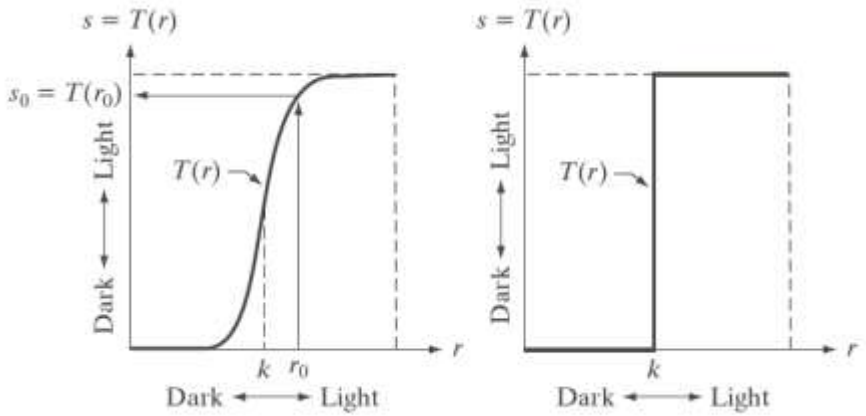


Fig. 2
Intensity transformation functions.
left - contrast stretching
right - thresholding function

Figure 2 left - ***T*** produces an output image of higher contrast than the original, by darkening the intensity levels below ***k*** and brightening the levels above ***k*** – this technique is called ***contrast stretching***.

Figure 2 right - ***T*** produces a binary output image. A mapping of this form is called *thresholding* function.

Some Basic Intensity Transformation Functions

Image Negatives

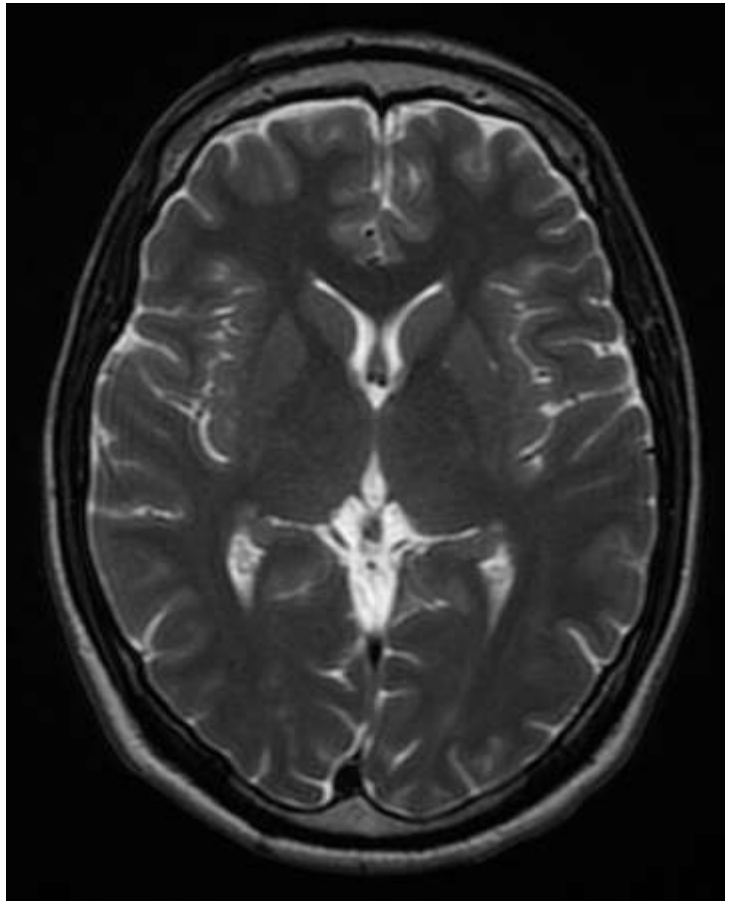
The negative of an image with intensity levels in ***[0, L-1]*** is obtain using the function

$$s = T(r) = L - 1 - r$$

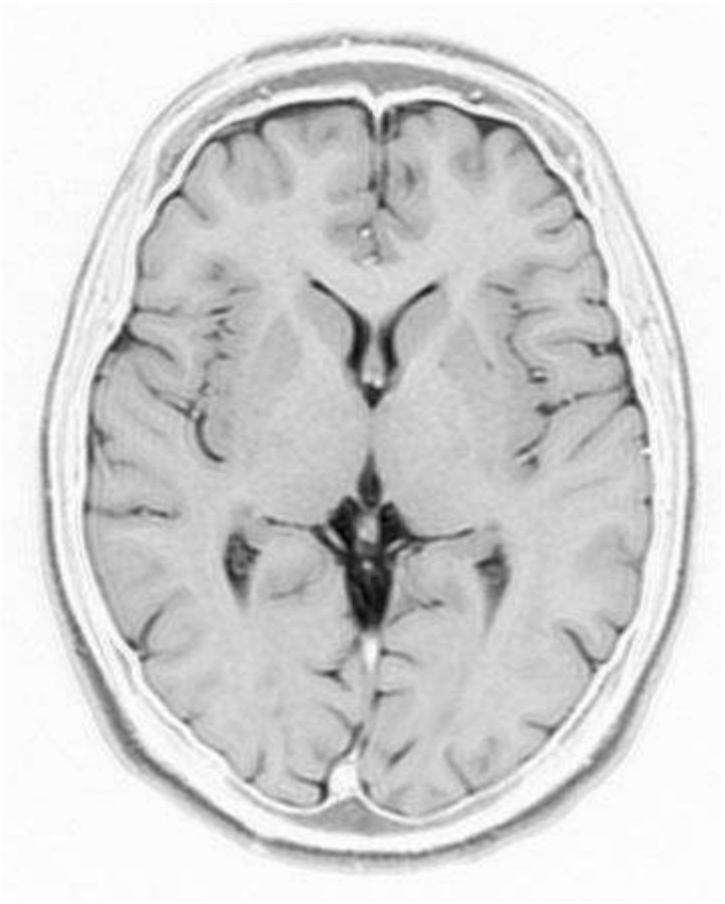
- equivalent of a photographic negative
- technique suited for enhancing white or gray detail embedded in dark regions of an image

Original

Negative image



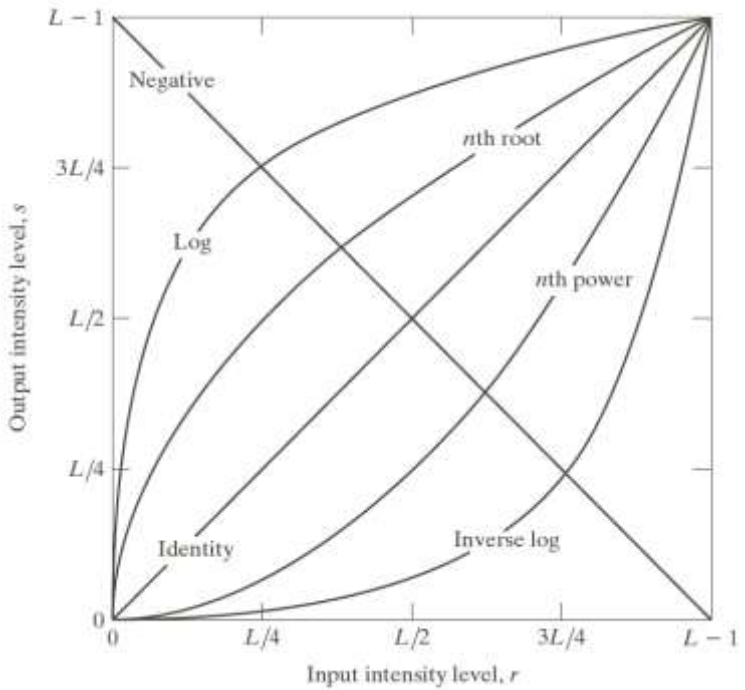
Original



Negative

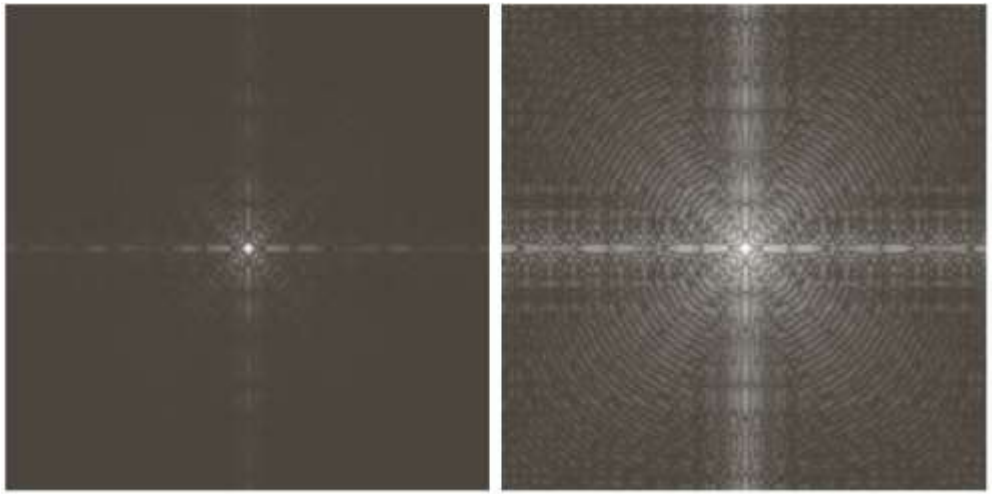
Log Transformations

$$s = T(r) = c\ \log(1+r)\ ,\ \ c \text{ - constant },\ r \geq 0$$



Some basic intensity transformation functions

This transformation maps a narrow range of low intensity values in the input into a wider range. An operator of this type is used to expand the values of dark pixels in an image while compressing the higher-level values. The opposite is true for the inverse ***log*** transformation. The ***log*** functions compress the dynamic range of images with large variations in pixel values.



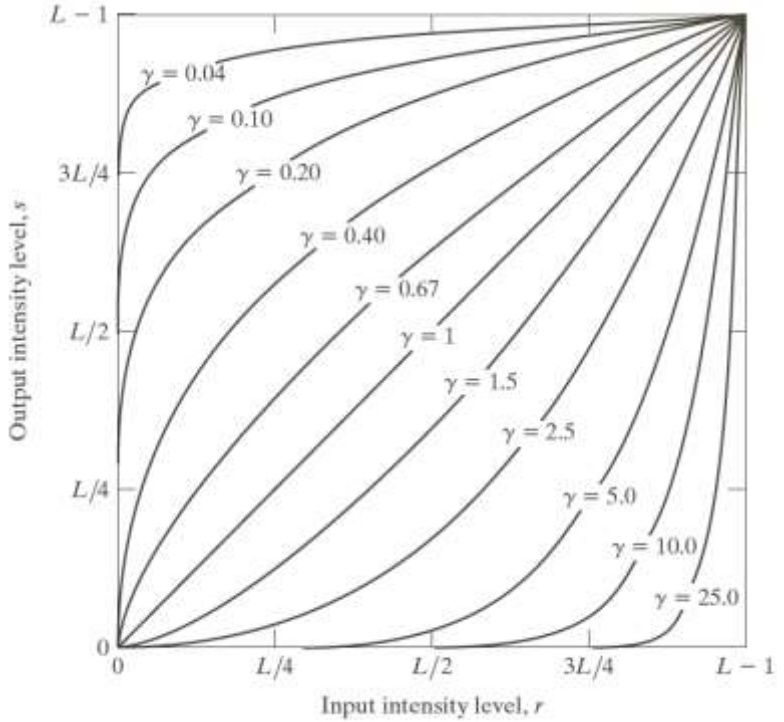
a b
(a) – Fourier spectrum
(b) – log transformation applied to (a), *c=1*
Fig. 4

Figure 4(a) – intensity values in the range ***0*** to ***1.5 x 10^6***

Figure 4(b) = log transformation of Figure 4(a) with ***c=1*** – range ***0*** to ***6.2***

Power-Law (Gamma) Transformations

$$s = T(r) = c\, r^{\gamma} \quad , \quad c, \gamma \text{ - positive constants } \left(\quad s = c(r+\varepsilon)^{\gamma} \quad \right)$$



Plots of gamma transformation for different values of γ (*c=1)*

Power-law curves with $\gamma < 1$ 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 values. The curves with $\gamma > 1$ have the opposite effect of those generated with values of $\gamma < 1$.

$c = \gamma = 1$ - identity transformation.

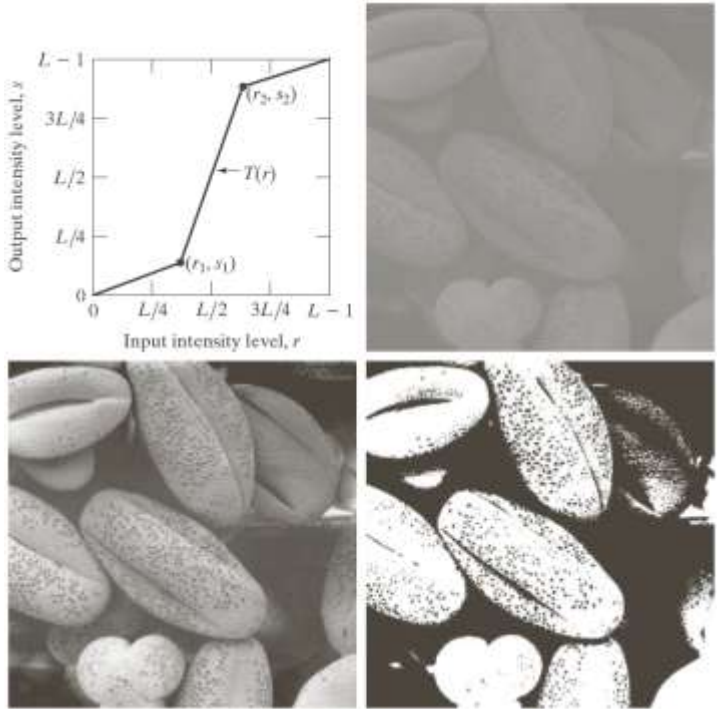
A variety of devices used for image capture, printing, and display respond according to a power law. The process used to correct these power-law response phenomena is called *gamma correction*.

a b
c d
(a) – aerial image
(b) – (d) – results of applying gamma transformation with *c=1* and
γ=3.0, 4.0 and *5.0* respectively

Piecewise-Linear Transformations Functions

Contrast stretching

A process that expands the range of intensity levels in an image so it spans the full intensity range of the recording tool or display device



a b
c d

Fig.5

(http://www.tripletwist.com/makeuphairblog/bridal/nikond3/)

$$T(r)=\begin{cases}\dfrac{s_1}{r_1}r & r\in[0,r_1]\\ \dfrac{s_2(r-r_1)}{(r_2-r_1)}+\dfrac{s_1(r_2-r)}{(r_2-r_1)} & r\in[r_1,r_2]\\ \dfrac{s_2(L-1-r)}{(L-1-r_2)} & r\in[r_2,L-1]\end{cases}$$

$r_1=s_1$, $r_2=s_2$ → identity transformation (no change)

$r_1=r_2$, $s_1=0$, $s_2=L-1$ → *thresholding function*

Figure 5(b) shows an 8-bit image with low contrast.

Figure 5(c) - contrast stretching, obtained by setting the parameters $(r_1, s_1) = (r_{\min}, 0)$, $(r_2, s_2) = (r_{\max}, L-1)$ where r_{min} and r_{max} denote the minimum and maximum gray levels in the image, respectively. Thus, the transformation function stretched the levels linearly from their original range to the full range ***[0, L-1]***.

Figure 5(d) - the thresholding function was used with $(r_1, s_1) = (m, 0)$, $(r_2, s_2) = (m, L-1)$ where m is the mean gray level in the image.

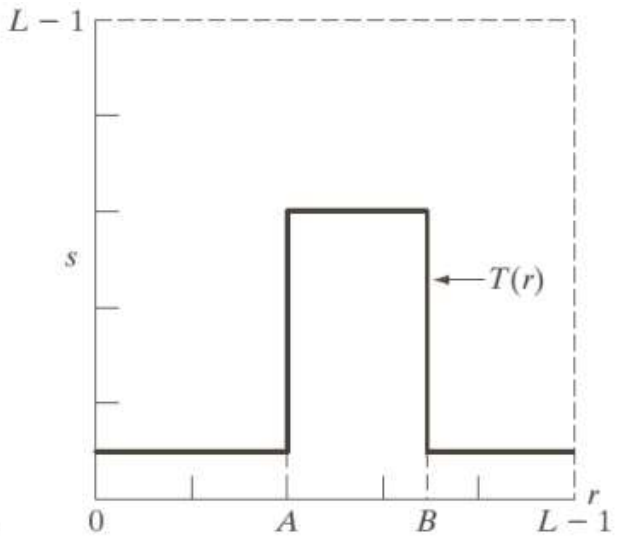
The original image on which these results are based is a scanning electron microscope image of pollen, magnified approximately 700 times.

Intensity-level slicing

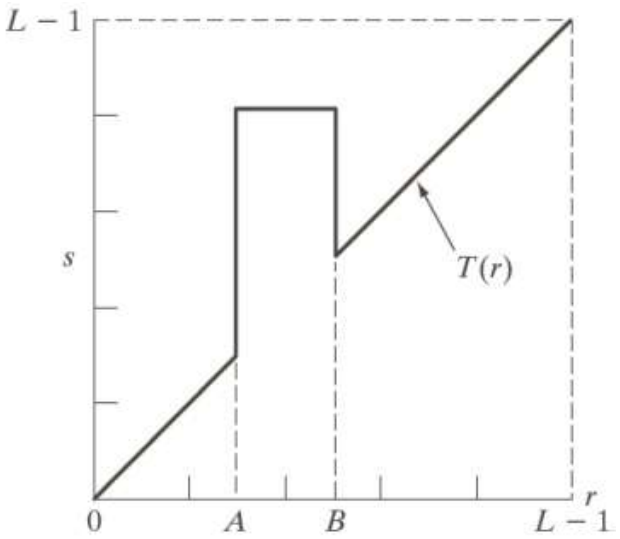
- highlighting a specific range of intensities in an image

There are two approaches for *intensity-level slicing*:

1. display in one value (white, for example) all the values in the range of interest and in another (say, black) all other intensities (Figure 3.11 (a))
2. brighten (or darken) the desired range of intensities but leaves unchanged all other intensities in the image (Figure 3.11 (b)).



Highlights intensity range [A, B] and reduces all other intensities to a lower level



Highlights range [A, B] and preserves all other intensities

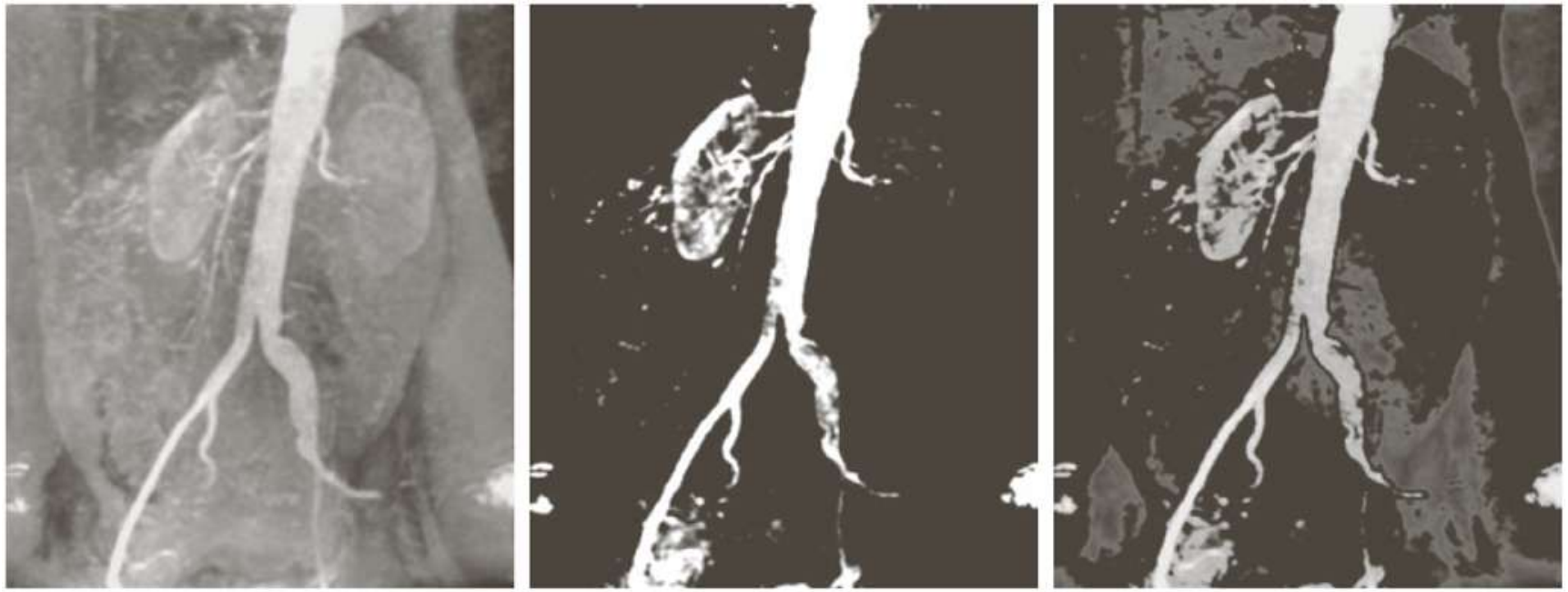


Fig. 6 - Aortic angiogram and intensity sliced versions

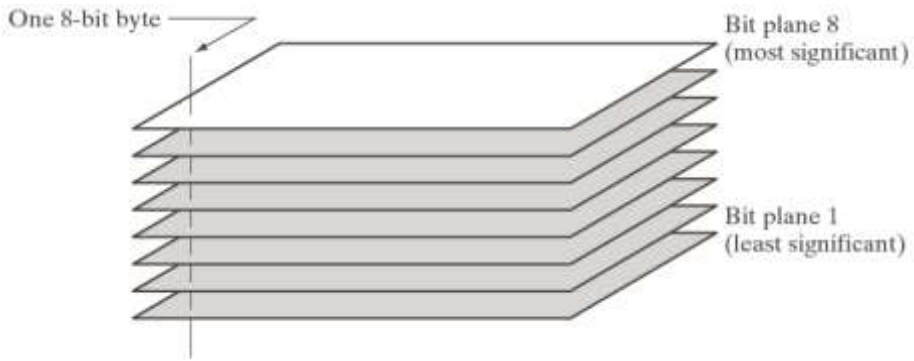
Figure 6 (left) – aortic angiogram near the kidney. The purpose of intensity slicing is to highlight the major blood vessels that appear brighter as a result of injecting a contrast medium. Figure 6(middle) shows the result of applying the first technique for a band near the top of the scale of intensities. This type of enhancement produces a binary image which is useful for studying the shape of the flow of the contrast substance (to detect blockages…)

In Figure 3.12(right) the second technique was used: a band of intensities in the mid-gray image around the mean intensity was set to black, the other intensities remain unchanged.

Bit-plane slicing

For an 8-bit image, ***f(x, y)*** is a number in ***[0,255]***, with 8-bit representation in base 2

This technique highlights the contribution made to the whole image appearances by each of the bits. An 8-bit image may be considered as being composed of eight 1-bit planes (plane 1 – the lowest order bit, plane 8 – the highest order bit)



a b c
d e f
g h i

(a) An 8-bit gray-scale image of size 500 × 1192 pixels. (b) through (i) Bit planes 1 through 8, with bit plane 1 corresponding to the least significant bit. Each bit plane is a binary image.

The binary image for the 8-th bit plane of an 8-bit image can be obtained by processing the input image with a threshold intensity transformation function that maps all the intensities between 0 and 127 to 0 and maps all levels between 128 and 255 to 1.

The bit-slicing technique is useful for analyzing the relative importance of each bit in the image – helps in determining the proper number of bits to use when quantizing the image. The technique is also useful for image compression.

a b c

Images reconstructed using (a) bit planes 8 and 7; (b) bit planes 8, 7, and 6; and (c) bit planes 8, 7, 6, and 5.

Histogram processing

The *histogram* of a digital image is with intensity levels in ***[0, L-1]***:

$$h(r_k) = n_k \ , \ k = 0,1,...,L-1$$

r_k the k-th intensity level

n_k the number of pixels in the image with intensity r_k

Normalized histogram for an ***M x N*** digital image:

$$p(r_k) = \frac{n_k}{MN} \ , \ k = 0,1,...,L-1$$

$p(r_k)$ = an estimate of the probability of occurrence of intensity level r_k in the image

$$\sum_{k=0}^{L-1} p(r_k) = 1$$

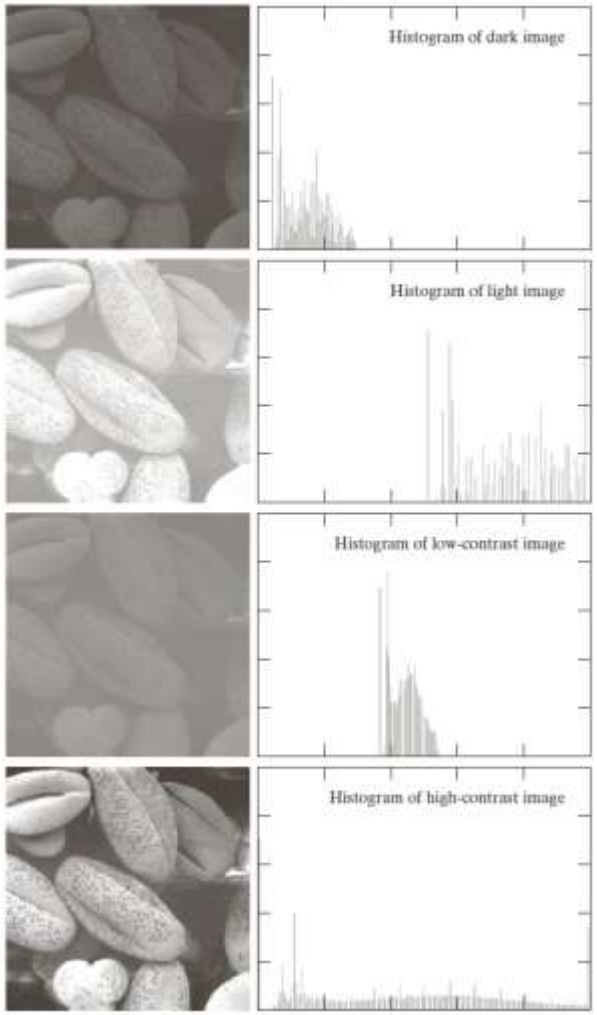


Fig. 8 – dark and light images, low-contrast, and high-contrast images and their histograms

Histogram Equalization (example)

1. determine a transformation function that seeks to produce an output image that has a uniform histogram

$$s = T(r) \quad , \quad 0 \le r \le L-1$$

   1. ***T(r)*** – monotonically increasing
   2. $0 \le T(r) \le L-1$ for $0 \le r \le L-1$

T(r) monotonically increasing – guarantees that intensity values in output image will not be less than the corresponding input values

Relation (b) requires that both input and output images have the same range of intensities

Histogram equalization or *histogram linearization* transformation

$$s_k = T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) = \frac{(L-1)}{M \cdot N}\sum_{j=0}^{k} n_j$$

The output image is obtained by mapping each pixel in the input image with intensity r_k into a corresponding pixel with intensity s_k in the output image.

Consider the following example: 3-bit image (***L=8***), ***64x64*** image (***M=N=64, MN=4096***)

r_k	n_k	$p_r(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02

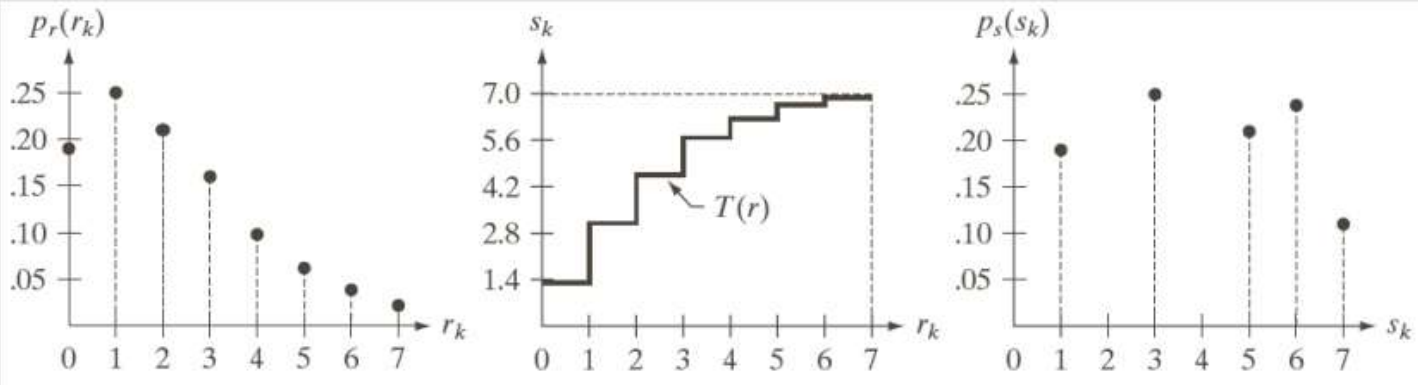
Intensity distribution and histogram values for a 3-bit 64×64 digital image

$$s_0 = T(r_0) = 7\sum_{j=0}^{0} p_r(r_j) = 7\,p_r(r_0) = 1.33$$

$$s_1 = T(r_1) = 7\sum_{j=0}^{1} p_r(r_j) = 7\,p_r(r_0) + 7\,p_r(r_1) = 3.08$$

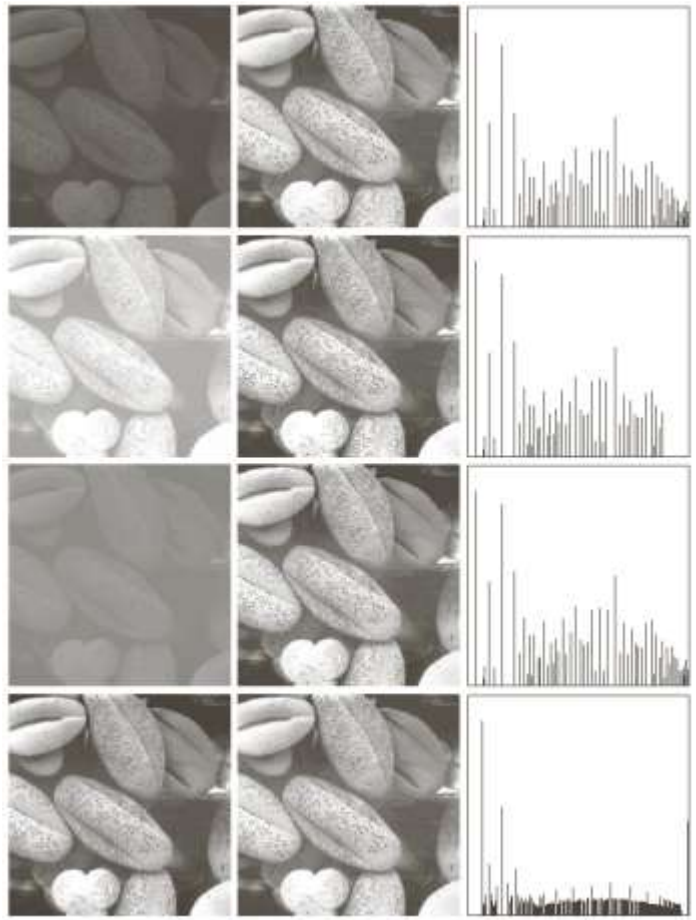
$$s_2 = 4.55 \quad , \quad s_3 = 5.67 \quad , \quad s_4 = 6.23 \quad , \quad s_5 = 6.65 \quad , \quad s_6 = 6.86 \quad , \quad s_7 = 7.00$$

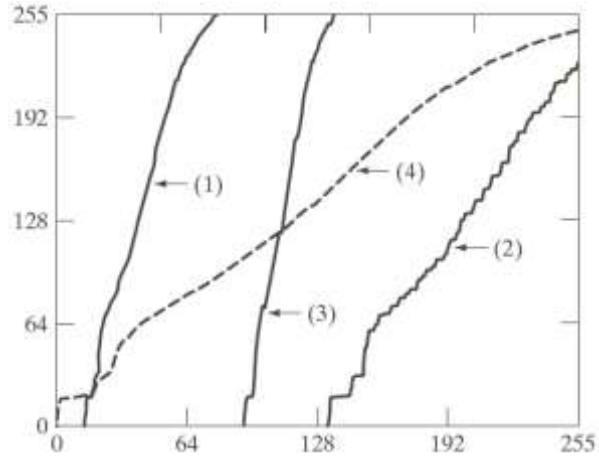
$$s_0 = 1.33 \rightarrow 1 \qquad s_4 = 6.23 \rightarrow 6$$
$$s_1 = 3.08 \rightarrow 3 \qquad s_5 = 6.65 \rightarrow 7$$
$$s_2 = 4.55 \rightarrow 5 \qquad s_6 = 6.86 \rightarrow 7$$
$$s_3 = 5.67 \rightarrow 6 \qquad s_7 = 7.00 \rightarrow 7$$



a b c

Illustration of histogram equalization of a 3-bit (8 intensity levels) image. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

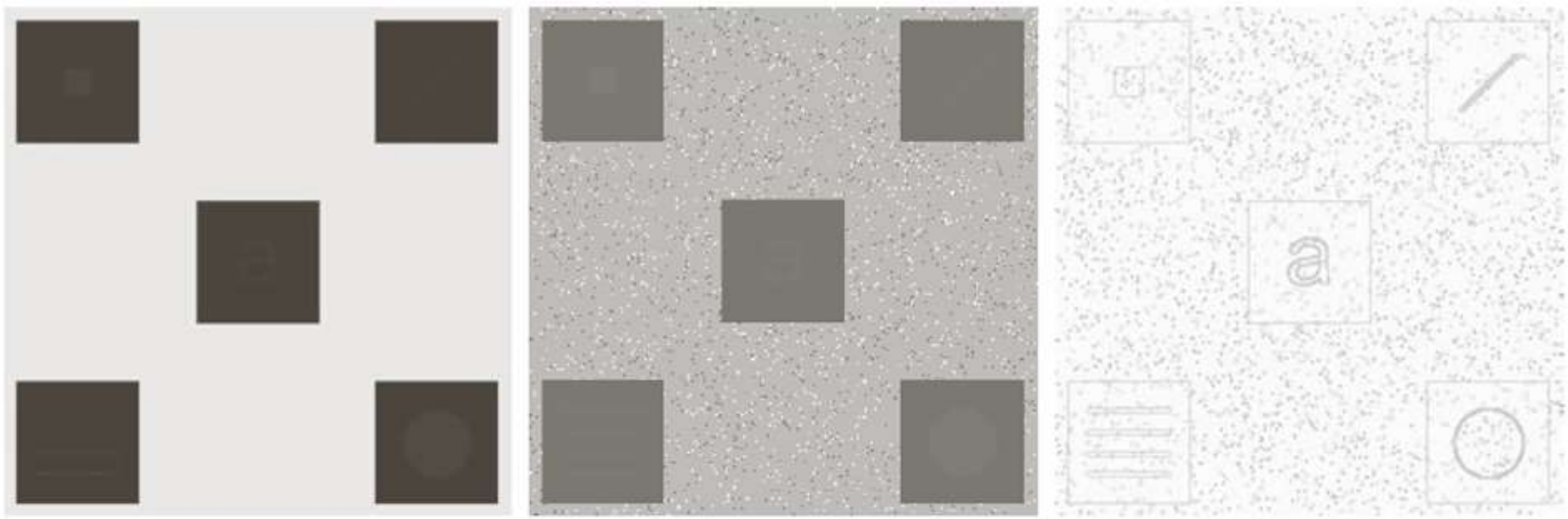




Transformation functions for histogram equalization. Transformations (1) through (4) were obtained from the histograms of the images

Local Histogram Processing

The histogram processing techniques previously described are easily adaptable to local enhancement. The procedure is to define a square or rectangular neighborhood and move the center of this area 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. This function is finally used to map the gray level of the pixel centered in the neighborhood. The center of the neighborhood region is then moved to an adjacent pixel location and the procedure is repeated. Updating the histogram obtained in the previous location with the new data introduced at each motion step is possible.



a b c

(a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization applied to (a), using a neighborhood of size 3 × 3.

Using Histogram Statistics for Image Enhancement

Let ***r*** denote a discrete random variable representing discrete gray-levels in ***[0, L-1]***, and let ***p(r_i)*** denote the normalized histogram component corresponding to the ***i***-th value of ***r***. The ***n***-th moment of ***r*** about its mean is defined as:

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

m is the mean (average intensity) value of ***r***:

$$m = \sum_{i=0}^{L-1} r_i \, p(r_i) \quad \text{- measure of average intensity}$$

$$\mu_2(r) = \sigma^2 = \sum_{i=0}^{L-1} (r_i - m)^2 \, p(r_i) \;, \quad \sigma - \text{ measure of contrast}$$

Sample mean and *sample variance*:

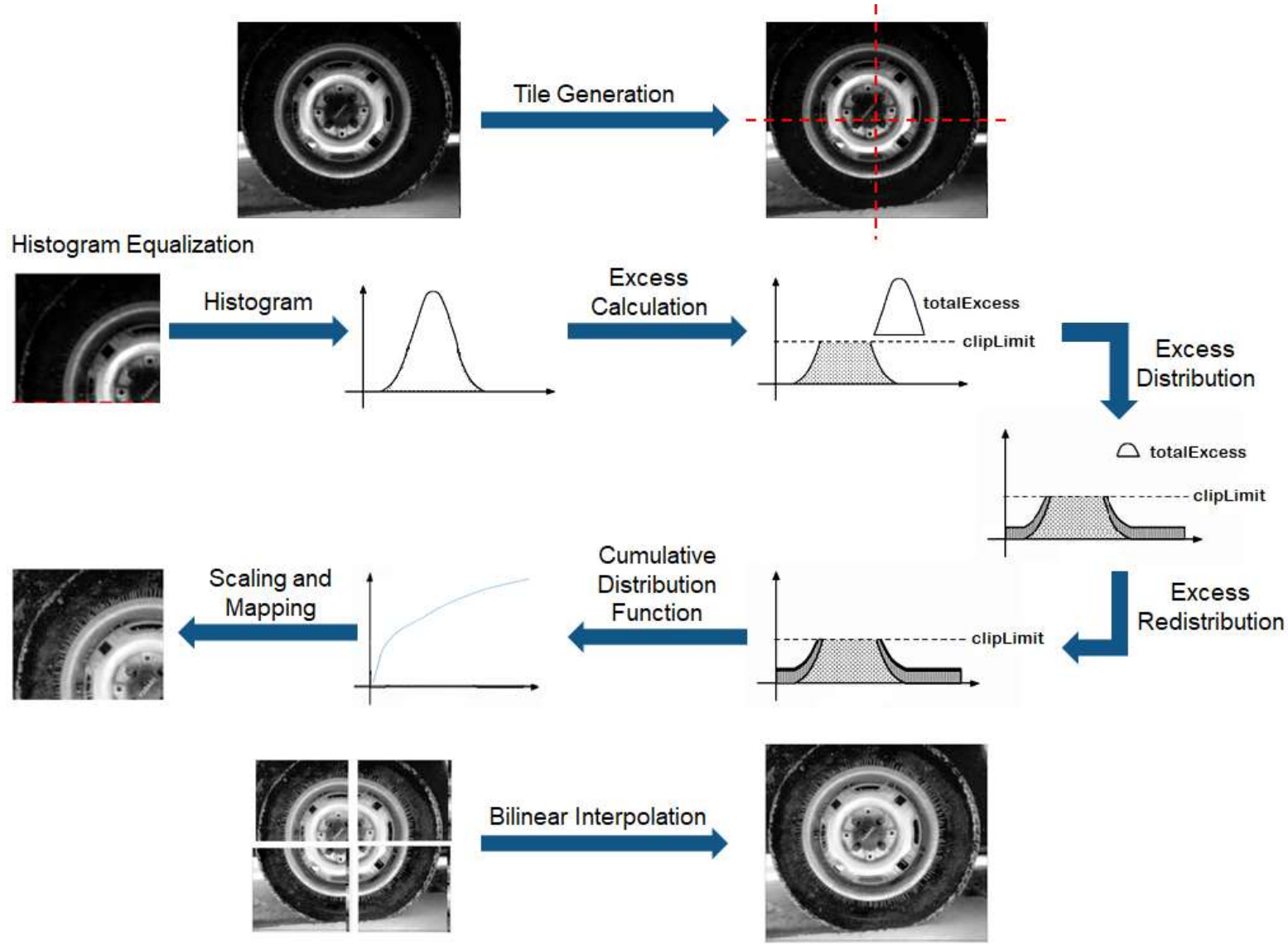
$$m = \frac{1}{M \cdot N} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \quad , \quad \sigma^2 = \frac{1}{M \cdot N} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[f(x,y) - m \right]^2$$

CLAHE – Contrast Limited Adaptive Histogram Equalization

https://ww2.mathworks.cn/help/visionhdl/ug/contrast-adaptive-histogram-equalization.html

1. **Tiles/blocks generation** – division of image in non-overlapping *m*×*n* regions
2. **Computing the histogram for each block**: A pixel intensity histogram is calculated for each block, representing the distribution of brightness levels within that specific region.

3. **Histogram clippin**g: If any histogram bin exceeds a predefined maximum threshold, it is clipped to prevent excessive contrast amplification. The **clipped excess** is then **redistributed** evenly across all histogram bins, reducing noise in uniform areas.
4. **Local histogram equalization**: After adjusting the histogram, histogram equalization is applied to each block. This is done using the cumulative distribution function (CDF), which redistributes pixel intensity levels more uniformly.
5. **Bilinear interpolation between blocks**: To prevent abrupt transitions between blocks when reconstructing the image, bilinear interpolation is applied. This smooths out contrast differences between regions, resulting in a more natural and continuous output.
6. **Merging the processed tiles/blocks**: Finally, after equalization and interpolation, the processed blocks are merged to form the final image with enhanced contrast.


Tile Generation
Histogram Equalization
Histogram
Excess Calculation
totalExcess
clipLimit
Excess Distribution
totalExcess
clipLimit
Excess Redistribution
clipLimit
Cumulative Distribution Function
Scaling and Mapping
Bilinear Interpolation

Spatial Filtering

The name *filter* is borrowed from frequency domain processing, where '*filtering*' means accepting (passing) or rejecting certain frequency components. Filters that pass low frequency are called *lowpass* filters. A lowpass filter has the effect of blurring (smoothing) an image. The *filter*s are also called *masks*, *kernels*, *templates* or *windows*.

The Mechanics of Spatial Filtering

A spatial filter consists of:

1) a *neighborhood* (usually a small rectangle)
2) a *predefined operation* performed on the pixels in the neighborhood

Filtering creates a new pixel with the same coordinates as the pixel in the center of the neighborhood, and whose intensity value is modified by the filtering operation.

If the operation performed on the image pixels is linear, the filter is called *linear spatial filter*, otherwise the filter is *nonlinear*.

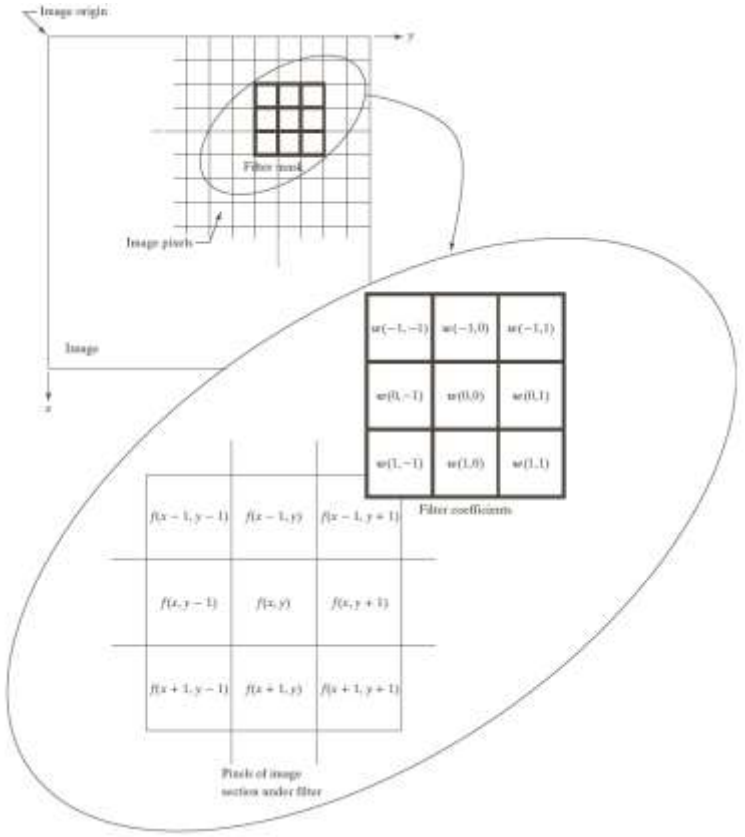


Fig. 10 – Linear spatial filtering with a ***3 × 3*** filter mask

In Figure 10 is pictured a ***3 × 3*** linear filter:

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

For a mask of size ***m× n***, we assume ***m=2a+1*** and ***n=2b+1***, where ***a*** and ***b*** are positive integers. The general expression of a linear spatial filter of an image of size ***M × N*** with a filter of size ***m× n*** is:

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

Smoothing Linear Filters

A smoothing linear filter computes the average of the pixels contained in the neighborhood of the filter mask. These filters are called sometimes *averaging filters* or *lowpass filters*.

The process of replacing the value of every pixel in an image by the average of the intensity levels in the neighborhood defined by the filter mask produces an image with reduced “sharp” transitions in intensities. Usually, random noise is characterized by such sharp transitions in intensity levels → smoothing linear filters are applied for noise reduction. The problem is that edges are also characterized by sharp intensity transitions, so averaging filters have the undesirable effect that they blur edges.

A major use of averaging filters is the reduction of “irrelevant” detail in an image (pixel regions that are small with respect to the size of the filter mask).

There is the possibility of using *weighted average*: the pixels are multiplied by different coefficients, thus giving more importance (weight) to some pixels at the expense of other.

$\frac{1}{9}\times$

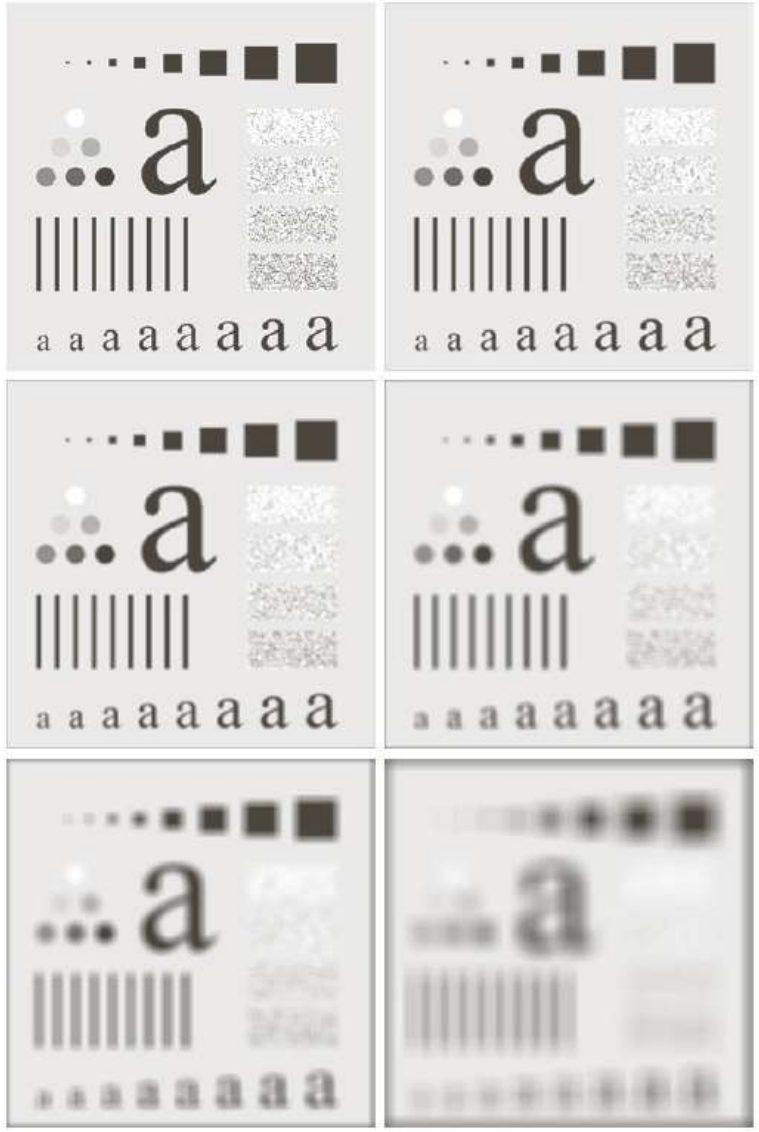
1	1	1
1	1	1
1	1	1

$\frac{1}{16}\times$

1	2	1
2	4	2
1	2	1

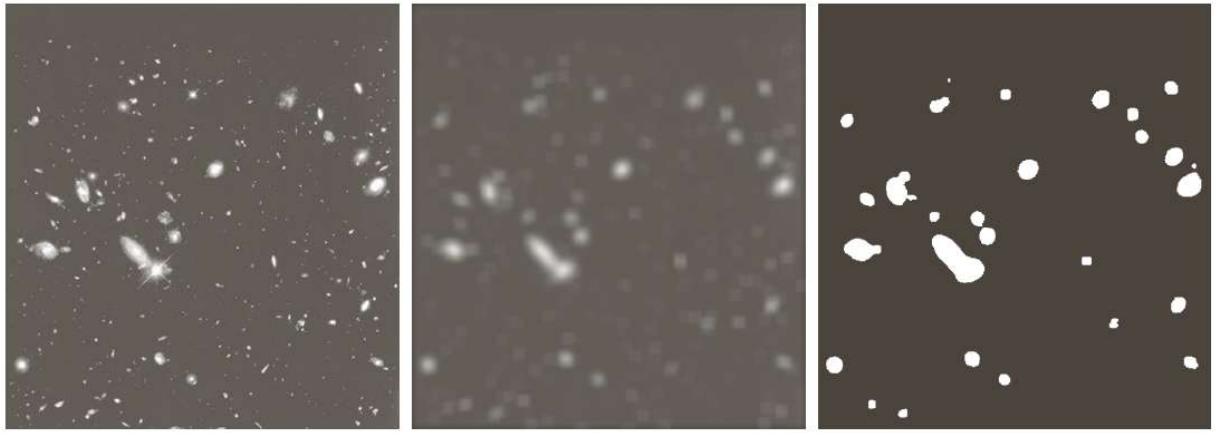
A general weighted averaging filter of size ***m*** $\times$ ***n*** (***m*** and ***n*** are odd) for an ***M*** $\times$ ***N*** image is given by the expression:

$$g(x,y)=\frac{\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)f(x+s,y+t)}{\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)} \qquad x=0,1,...,M-1\ ,\ y=0,1,...,N-1$$



a b
c d
e f
(a) – original image 500×500
(b) – (f) – results of smoothing with square averaging filters of size *m*=3,5,9,15, and 35, respectively
The black squares at the top are of size 3, 5, 9, 15, 25, 35, 45, 55. The letters at the bottom range in size from 10 cu 24 points. The vertical bars are 5 pixels wide and 100 pixels high, separated bu 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart. The noisy rectangles are 50×120 pixels.

An important application of spatial averaging is to blur an image for the purpose of getting a gross representation of objects of interest, such that the intensity of smaller objects blends with the background and larger object become “blob like” and easy to detect. The size of the mask establishes the relative size of the objects that will “disappear” in the background.



Left – image from the Hubble Space Telescope, 528×485; Middle – Image filtered with a 15×15 averaging mask; Right – result of thresholding the middle image

1

2-bit image $L = 2^2 = 4$

$\{0, 1, 2, 3\}$

$$I \begin{bmatrix} 0 & 0 & 1 \\ 1 & 2 & 3 \\ 3 & 1 & 1 \end{bmatrix} \quad 9 = 3 \times 3$$

$h(0) = 2$, $h(1) = 4$

$h(2) = 1$, $h(3) = 2$

for an 8-bit image $\Rightarrow$
h has 256 values.

Normalized histogram

$p(0) = \frac{2}{9}$, $p(1) = \frac{4}{9}$, $p(2) = \frac{1}{9}$

$p(3) = \frac{2}{9}$

2

10	10^1	10^2	20^1	30
0	10^2	0	20^2	30
0	20^1	20^2	0^1	0 0

$S_{xy} = 3 \times 3$ $\quad W = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \cdot \frac{1}{16}$

$0 \rightarrow \{ 10*1 + 10*2 + 20*1 + 10*2 + 0*4 + 20*2 + 20*1 + 20*2 + 0*1 \} / 16$

$$\frac{160}{16} = 10$$

10 10 10 10
10 10 10 10
200 200 200 200

$10 \rightarrow \{ 1*(10+10+200+200) + 2*(10+10+10+200) + 10*4 \} / 16$

3

$$\left[420 + 460 + 40\right] / 16 = \frac{920}{16} \simeq 58$$

$10 \xrightarrow{57}$ $10 \xrightarrow{57}$ $10 \xrightarrow{57}$

$200 \quad 200 \quad 200 - 153$

$200 \quad 200 \quad 200$

$200 \rightarrow \left[1 * (10 + 10 + 200 + 200) + 2 * (10 + 200 + 200 + 200) + 4 * 200\right] / 16$

$$\frac{420 + 1220 + 800}{16} = 153$$