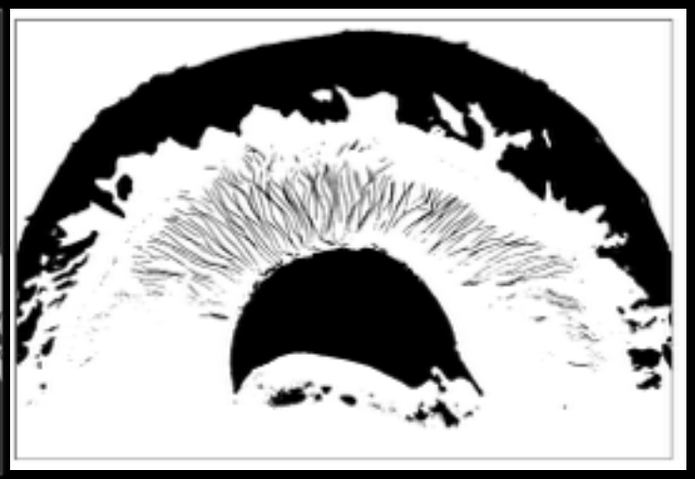
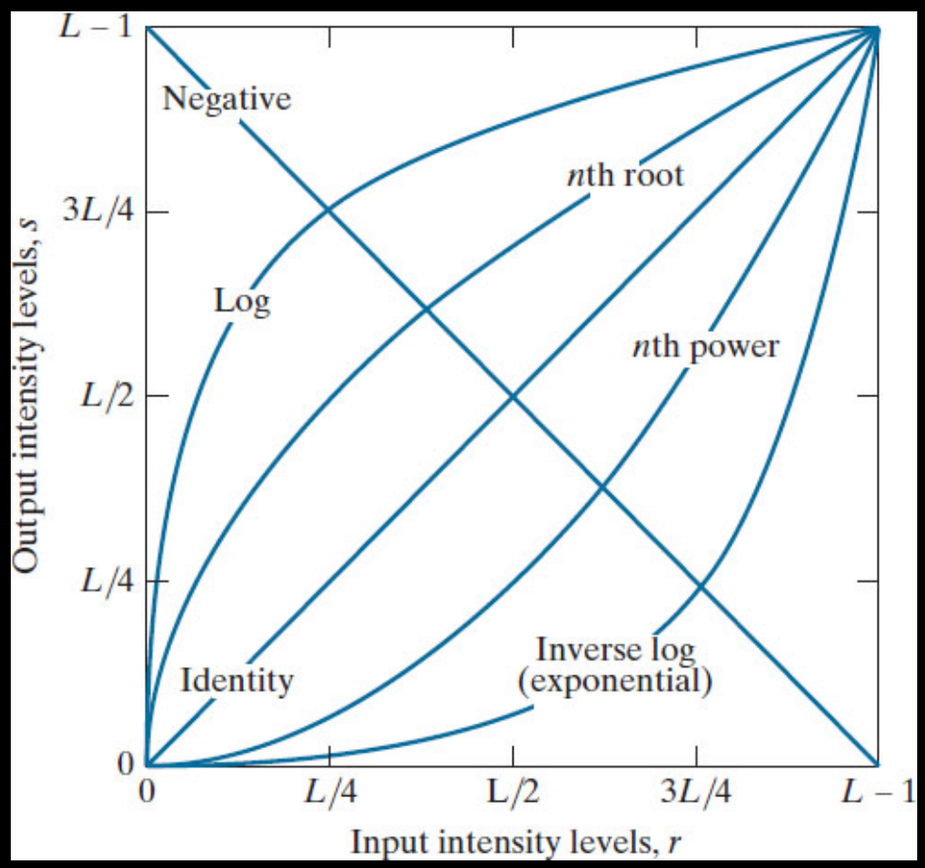
Chapter 3
Gonzales and Woods

Contrast and display scales




L – 1
3L/4
L/2
L/4
0
Output intensity levels, s
Negative
nth root
Log
nth power
Identity
Inverse log
(exponential)
0
L/4
L/2
3L/4
L – 1
Input intensity levels, r

Histograms

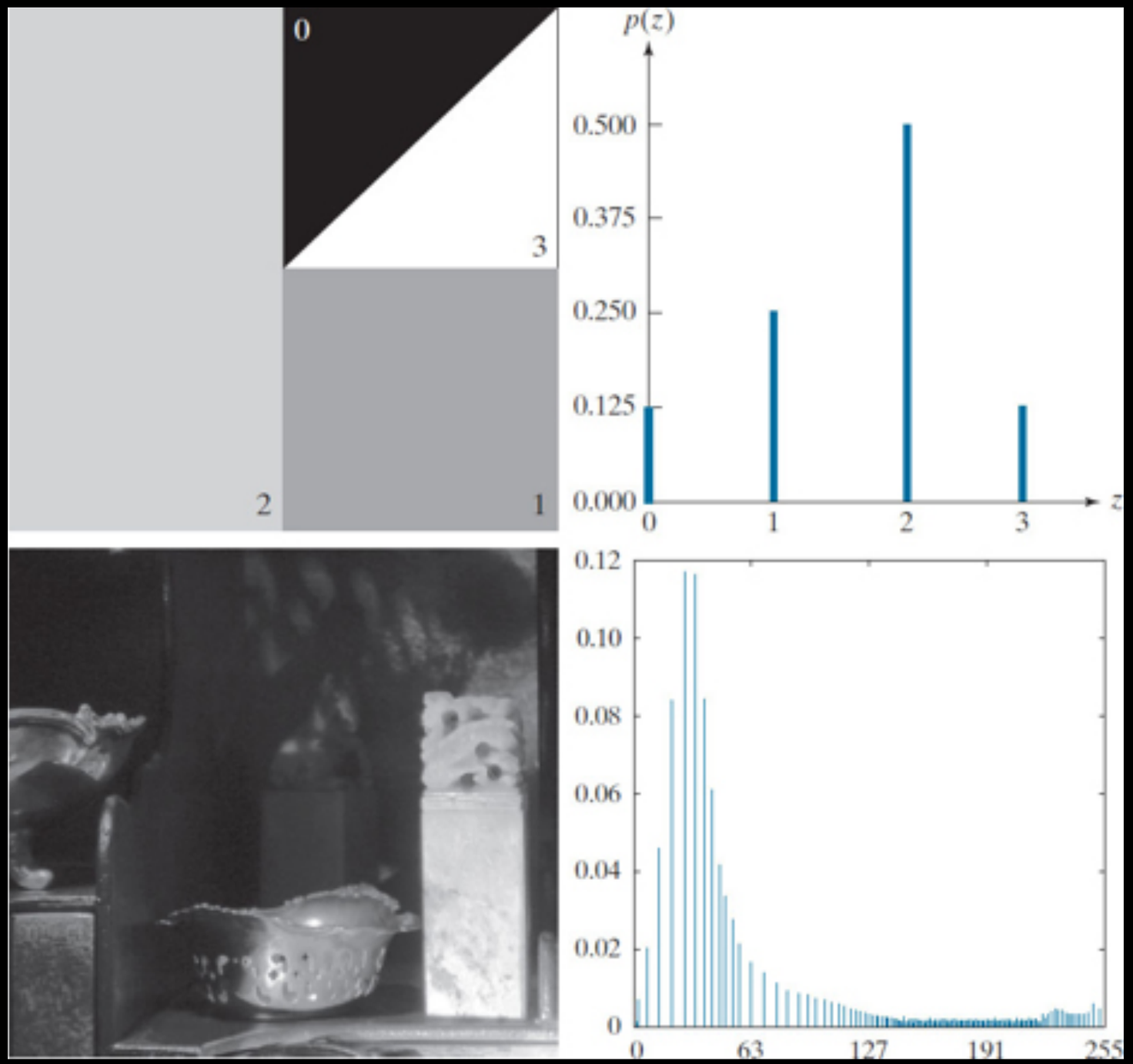
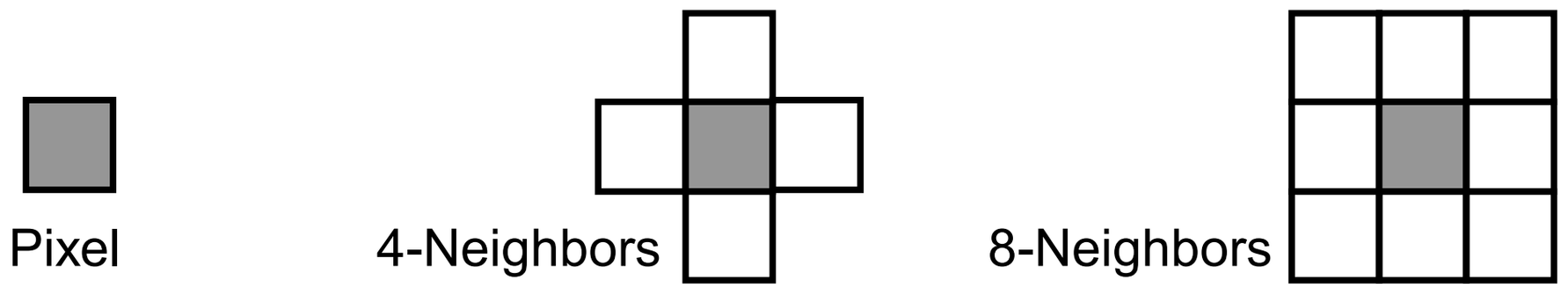

0
3
2
1
p(z)
0.500
0.375
0.250
0.125
0.000
0
1
2
3
z
0.12
0.10
0.08
0.06
0.04
0.02
0
0
63
127
191
255

Image Enhancement: Operating on the Pixels

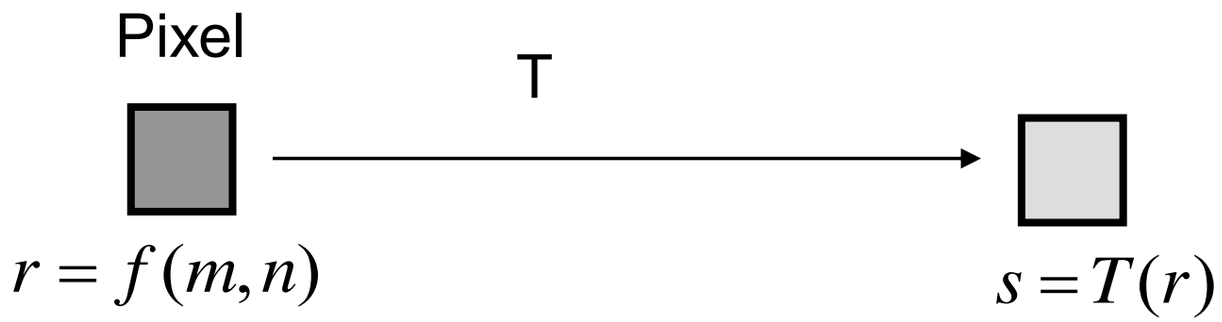
- Make image "better" for a specific application
  - The idea of "better" is somewhat subjective

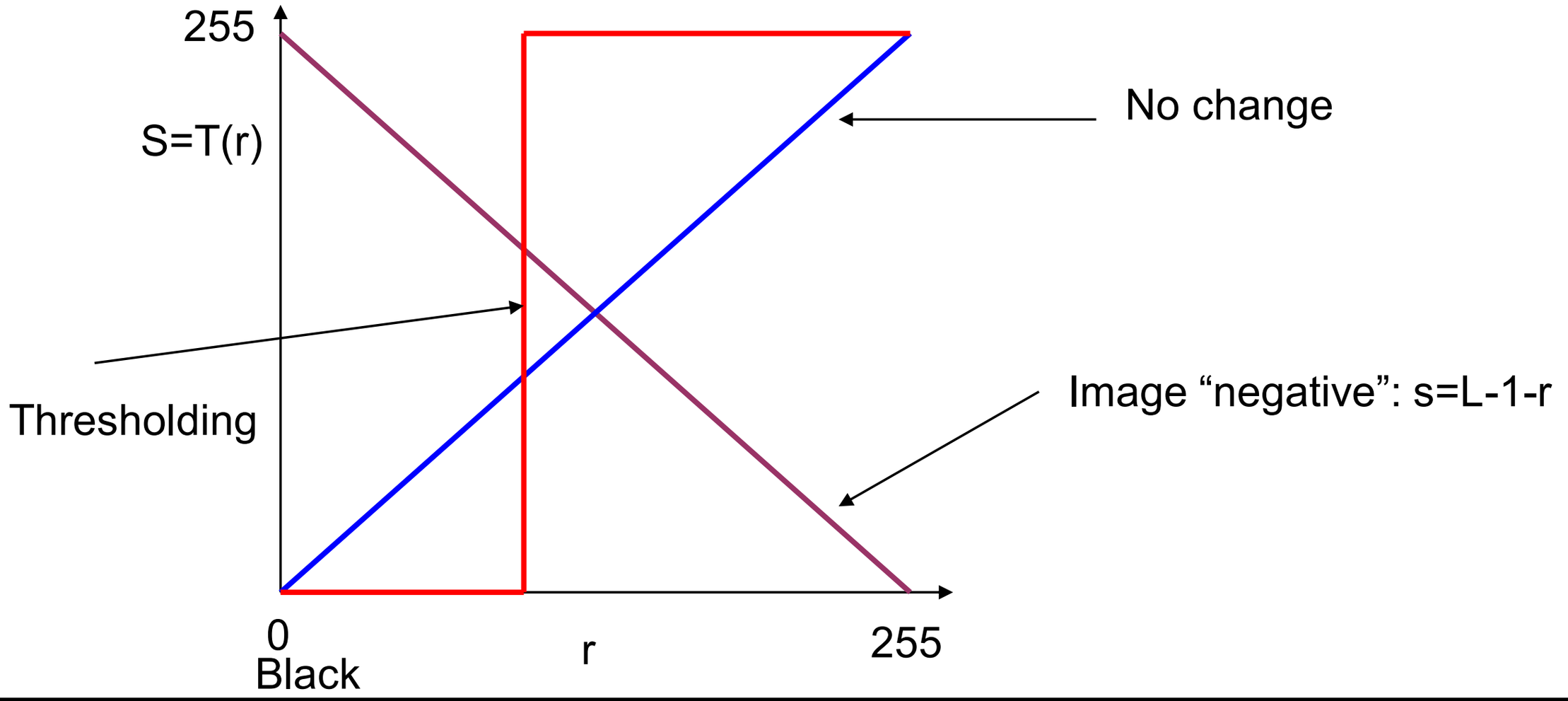
- We distinguish two domains:
  - Spatial or Pixel domain: $f(x, y)$ or $f(m, n)$
  - Frequency Domain: $F(w_x, w_y)$ or $F(u, v)$

- For this section: Pixel Domain
  - Operations on single pixel at a time
  - Operations on groups of pixels (neighborhoods)

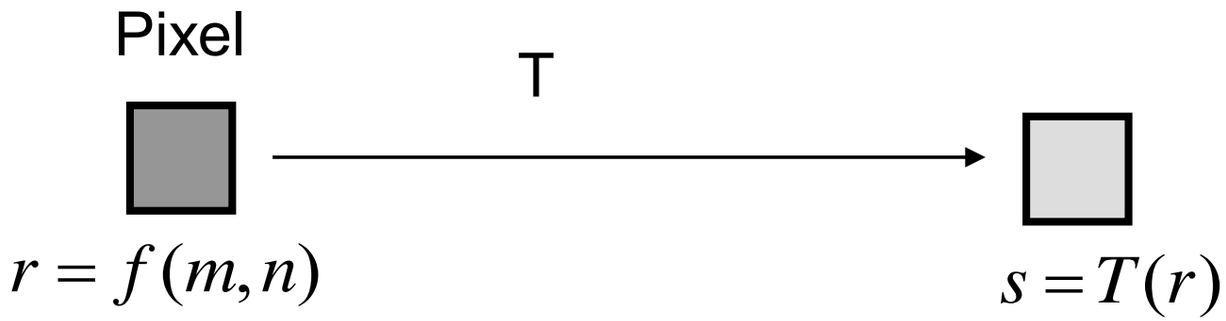


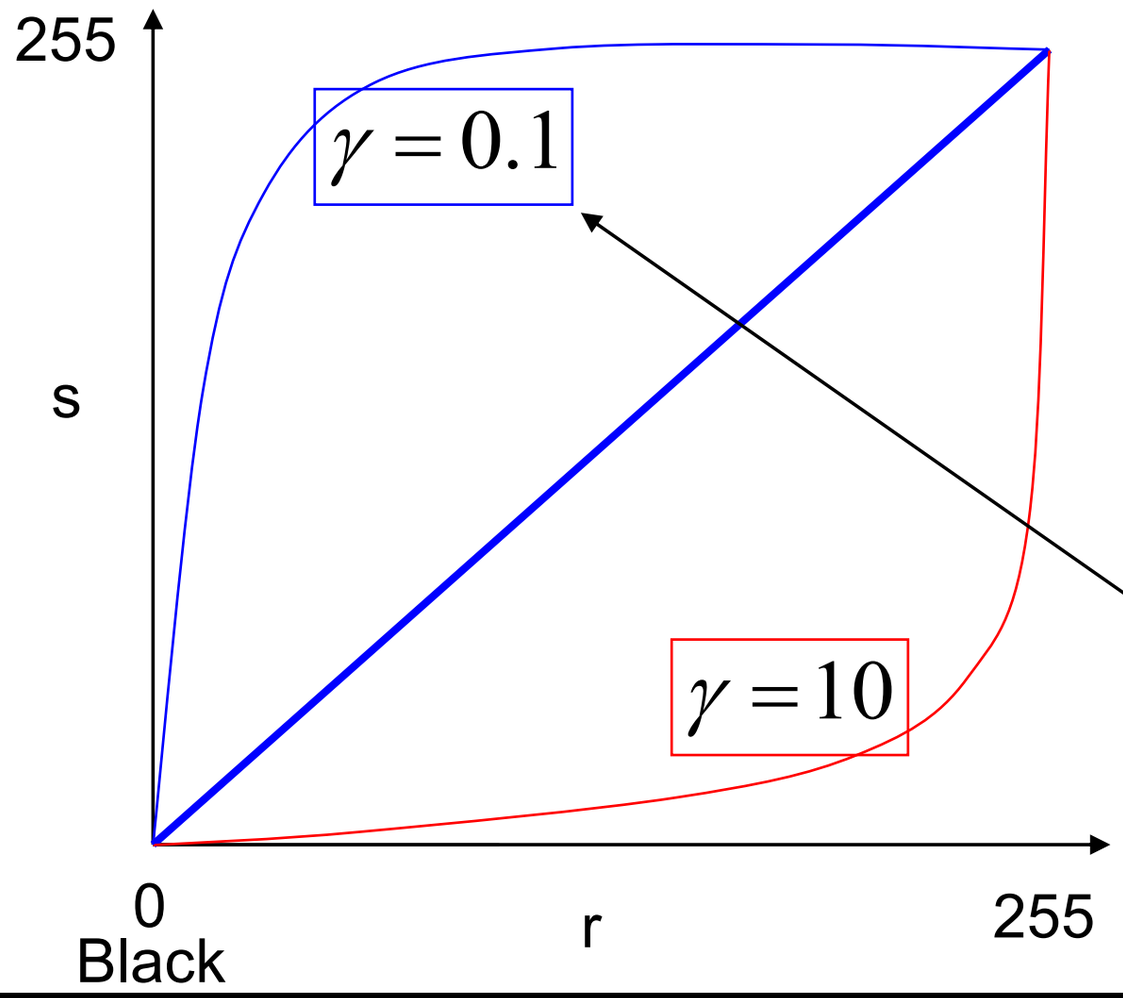
Simplest form of processing: Point Processing





Simplest form of processing: Point Processing





Common Examples:

- Dynamic Range Compression

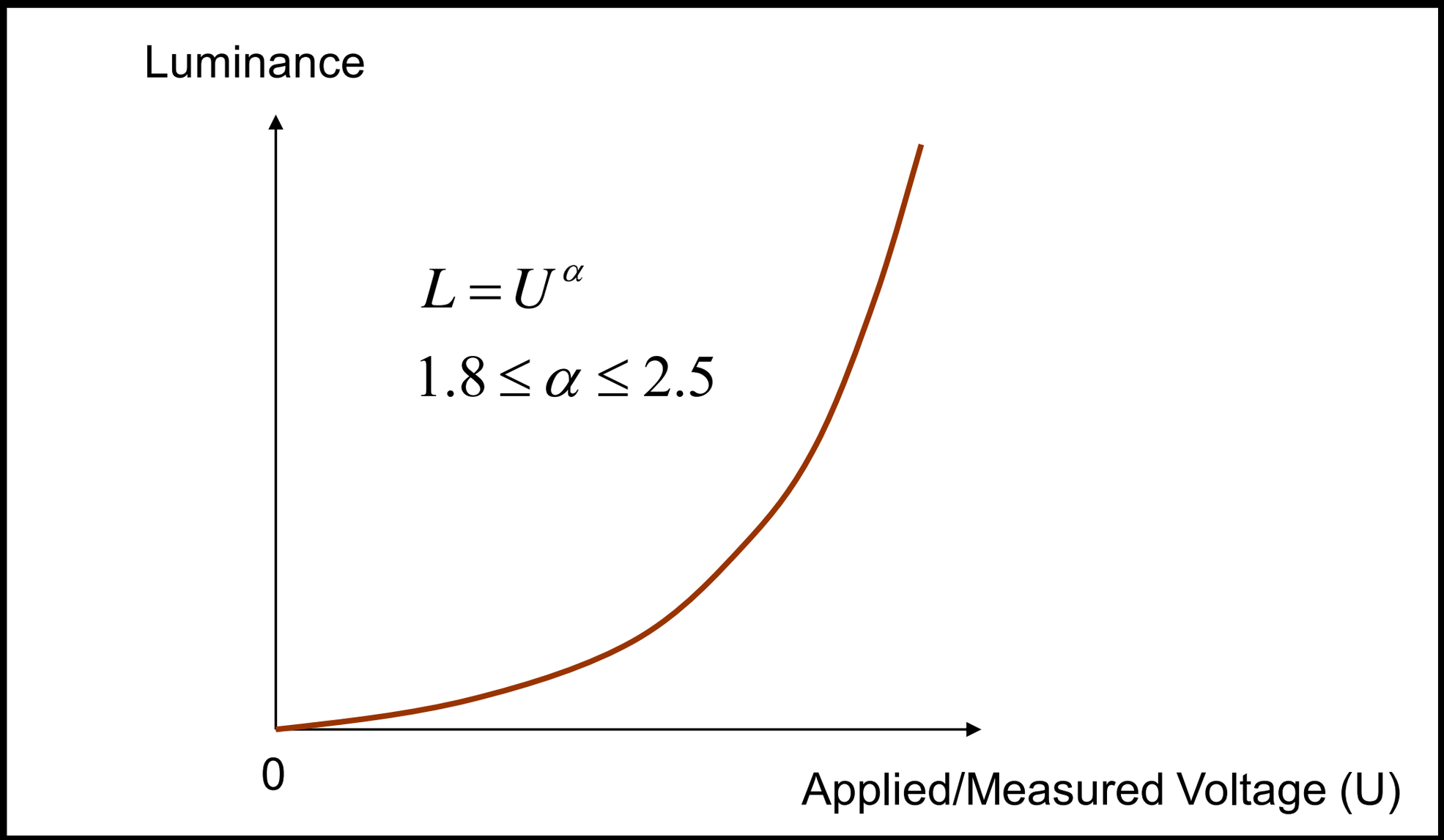
$$T(r) = c\log(1+r)$$

- Gamma Correction

$$T(r) = cr^{\gamma}$$

Narrow range of “dark” gets mapped to broad range of “gray”

Gamma Correction



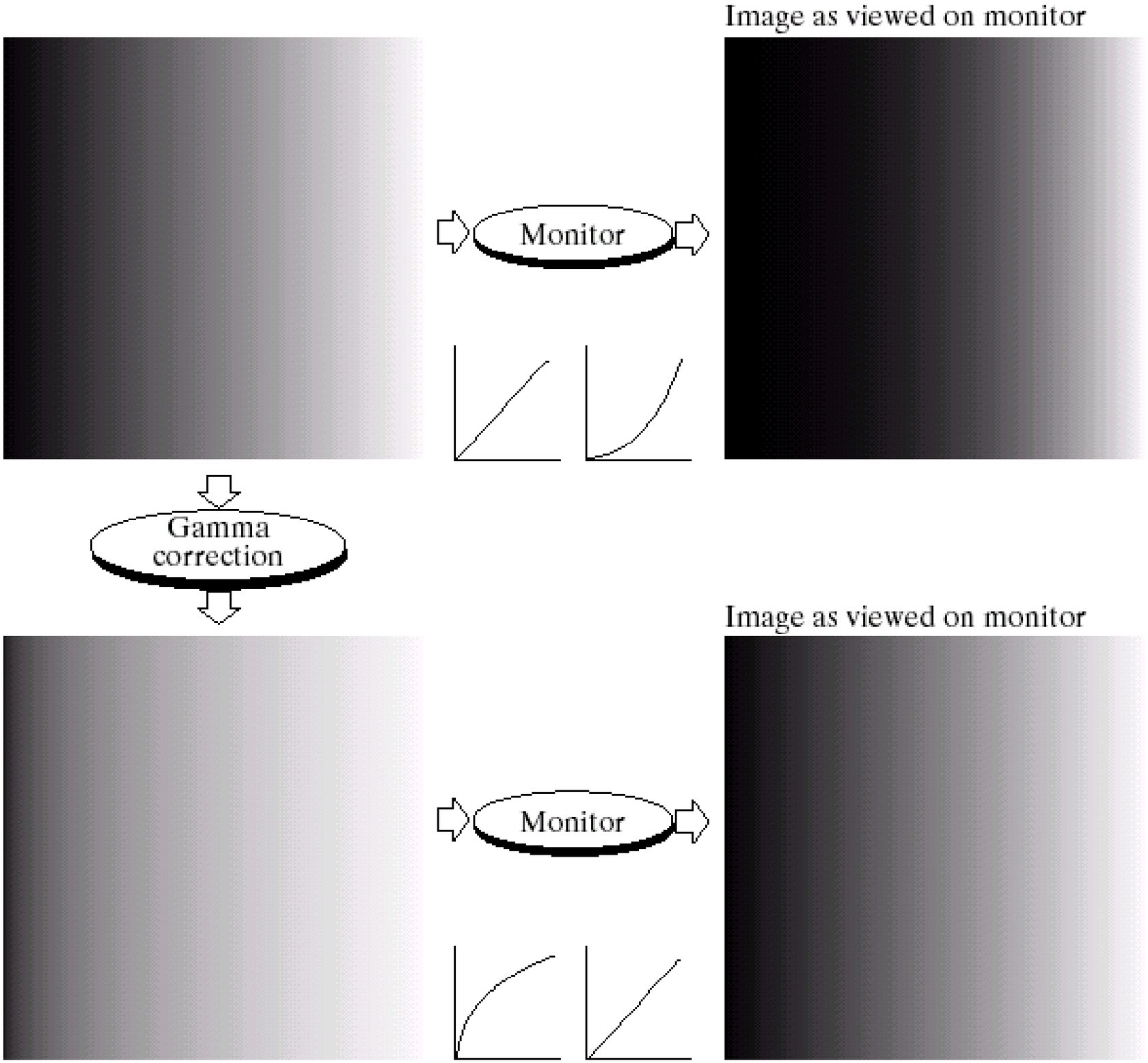
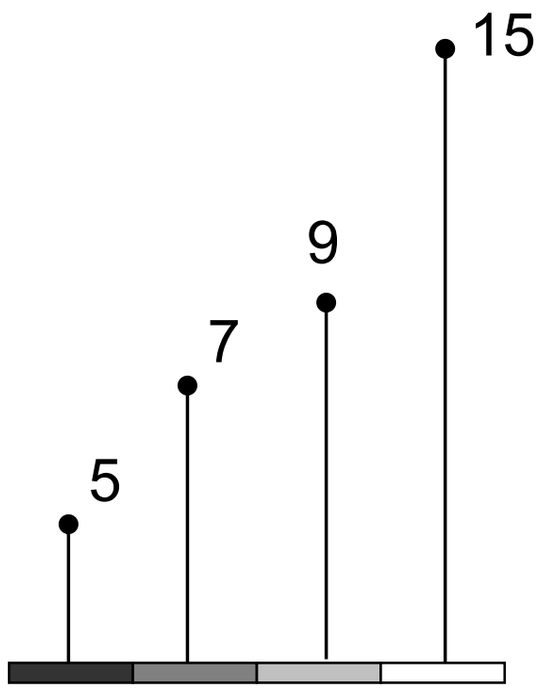


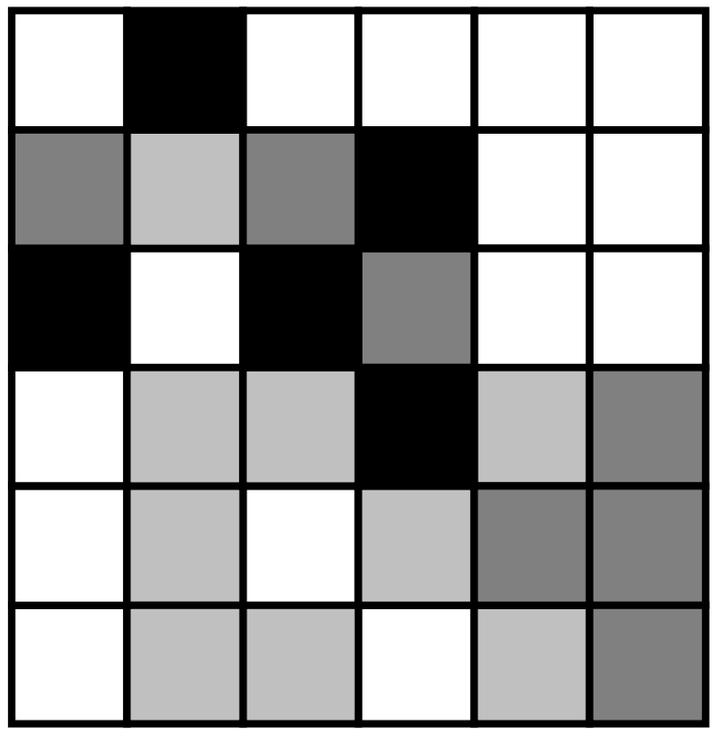
Image as viewed on monitor
Monitor
Gamma correction
Image as viewed on monitor
Monitor

Histogram Processing:

- Distribution of gray-levels can be judged by measuring a Histogram

Histogram:





Histogram Processing:

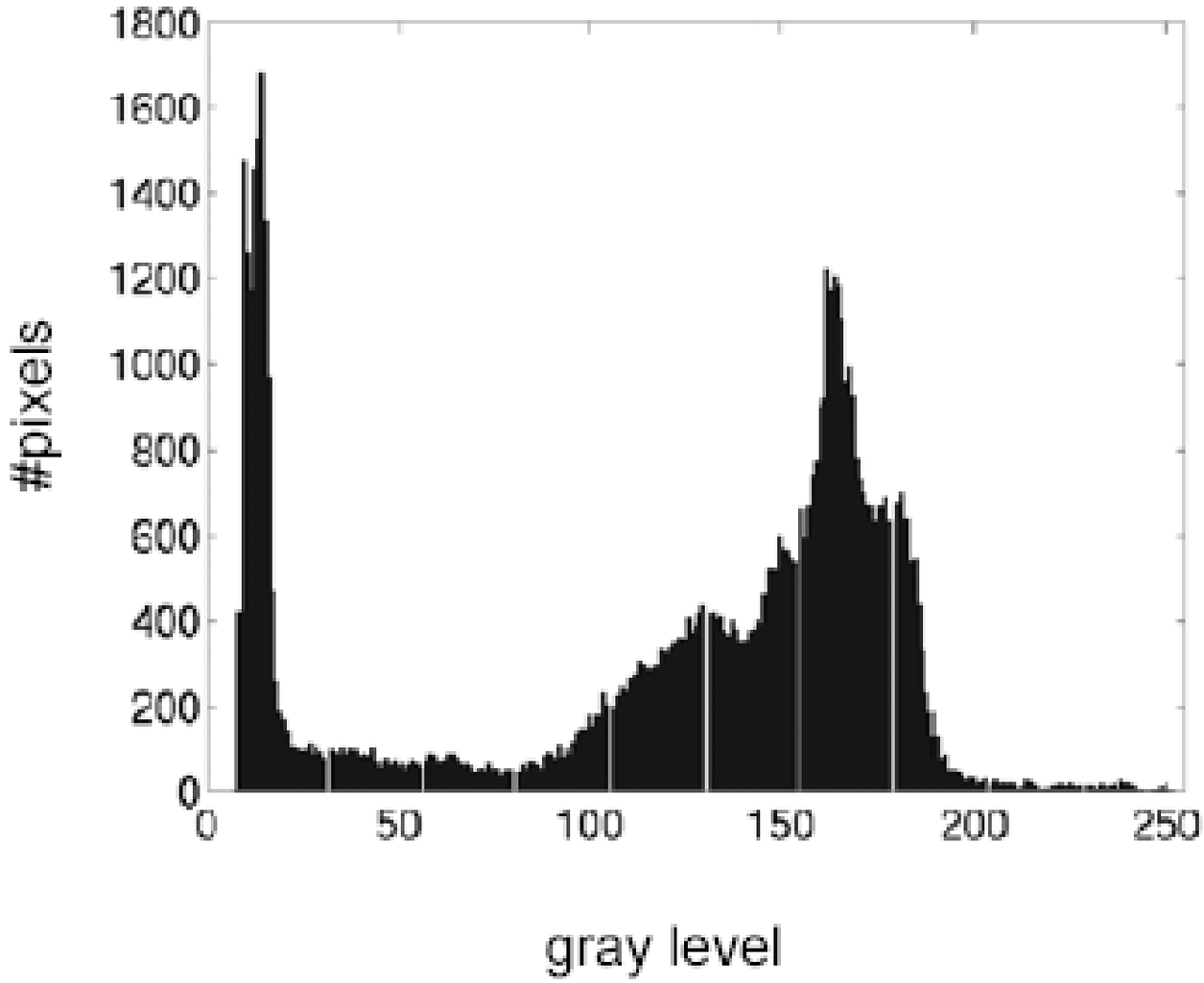
For B-bit image,

- Initialize 2^B counters with 0
- Loop over all pixels x,y
- When encountering gray level $f(x,y)=i$, increment the counter number i

- With proper normalization, the histogram can be interpreted as an estimate of the probability density function (pdf) of the underlying random variable (the graylevel)

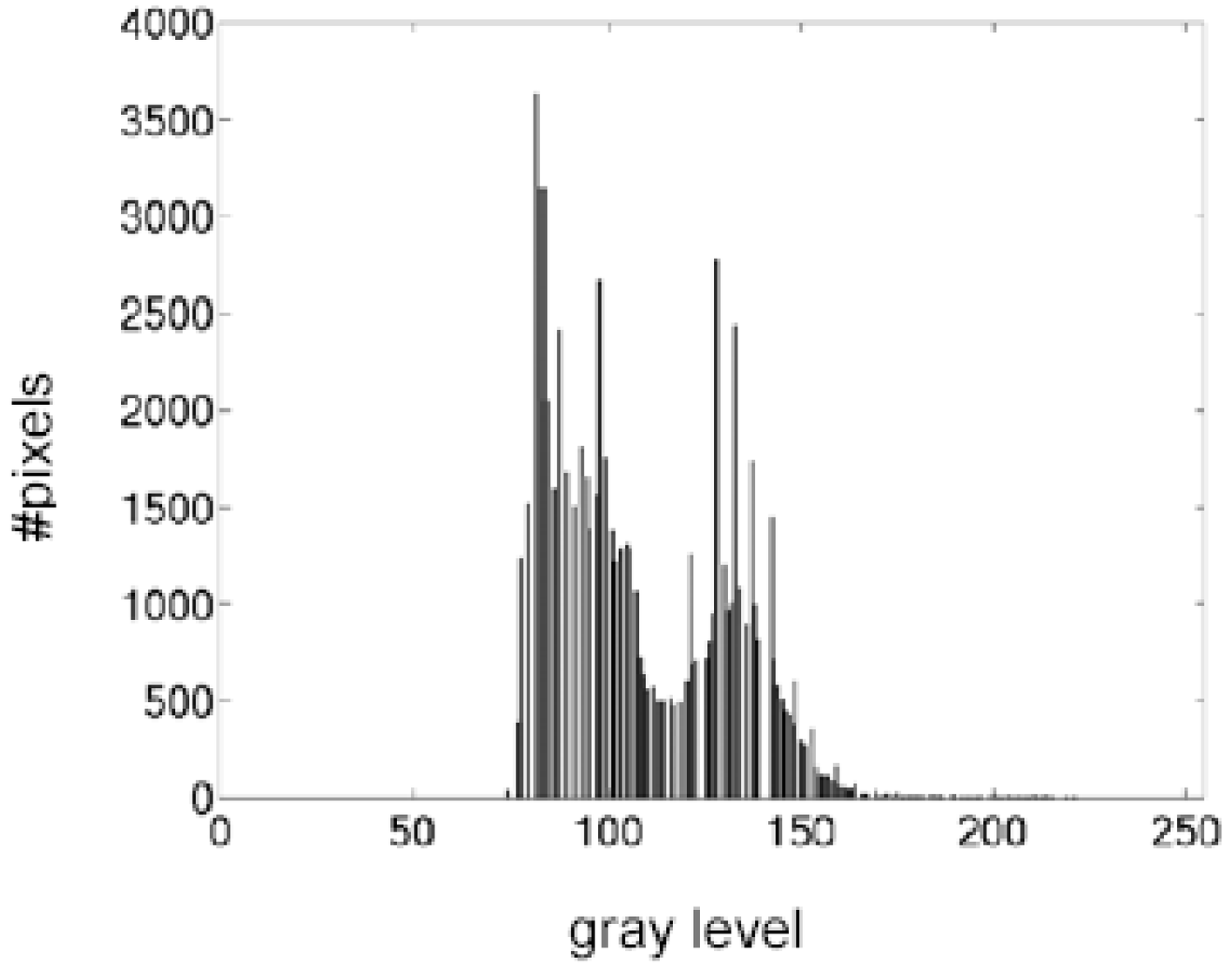
- You can also use fewer, larger bins to trade off amplitude

Example:



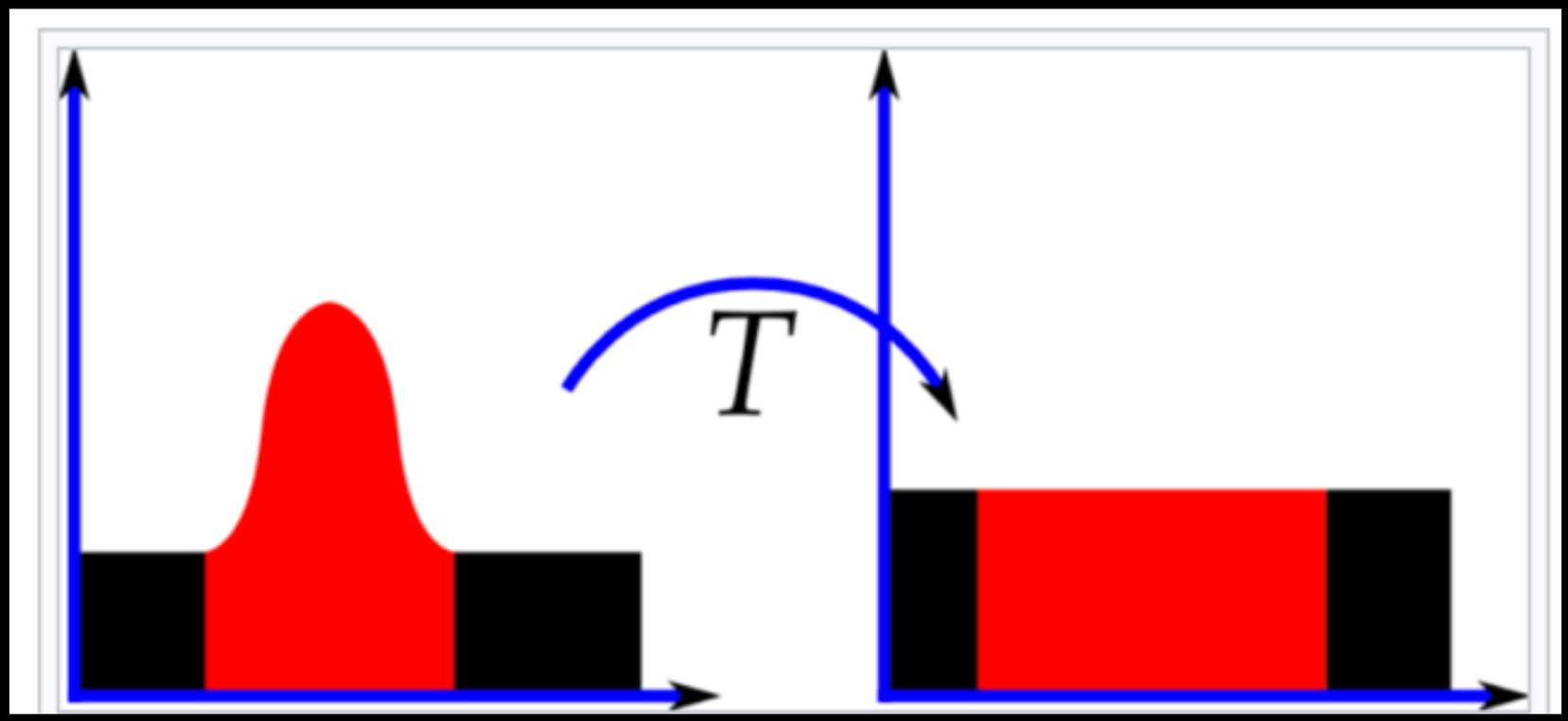
Cameraman image

Example:

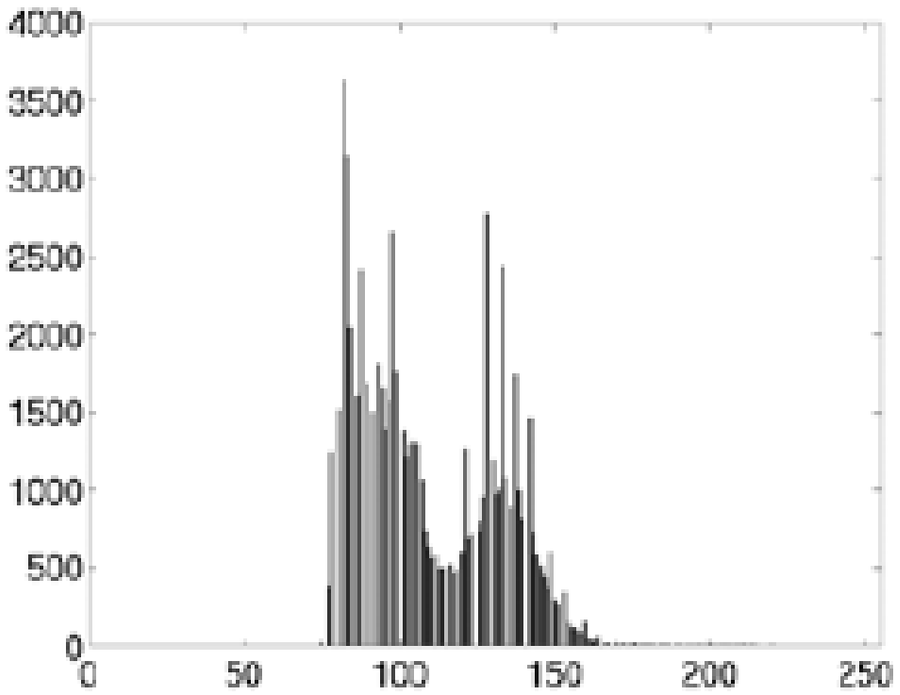


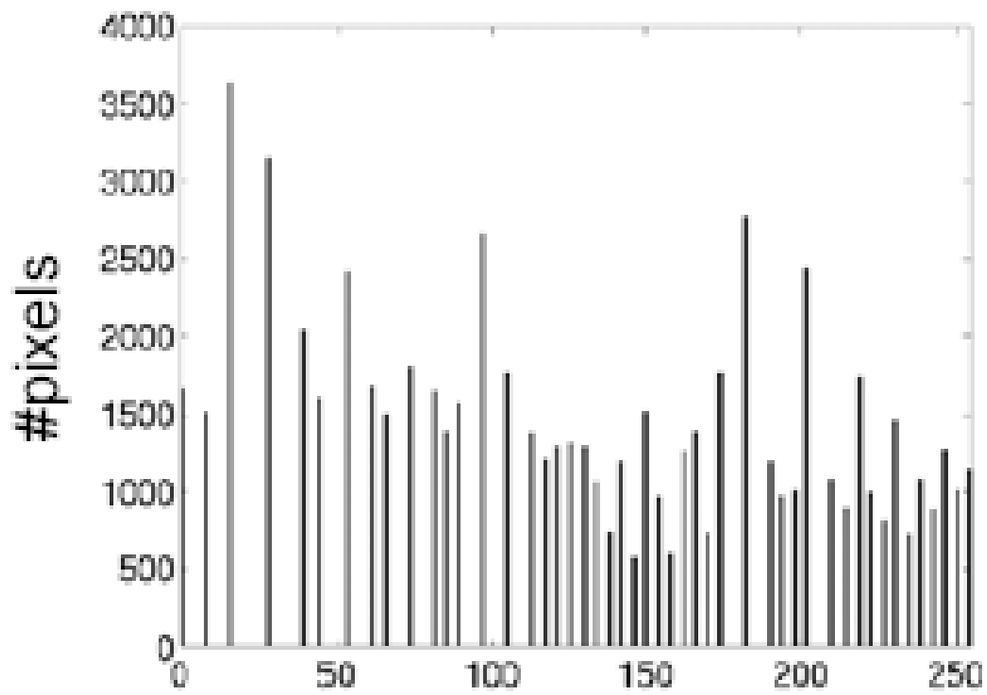
Pout image

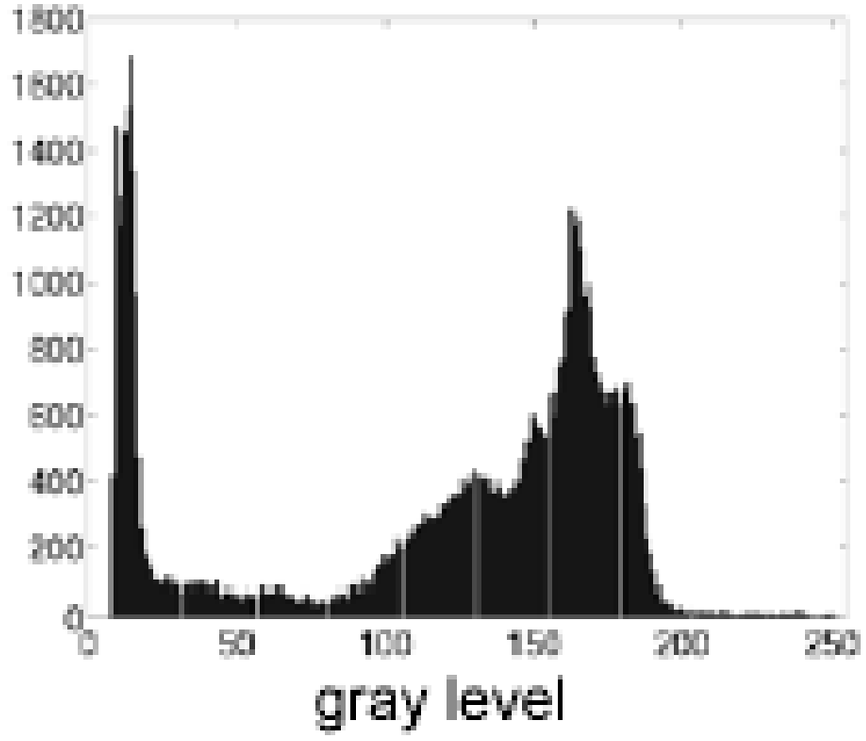
Histogram Equalization

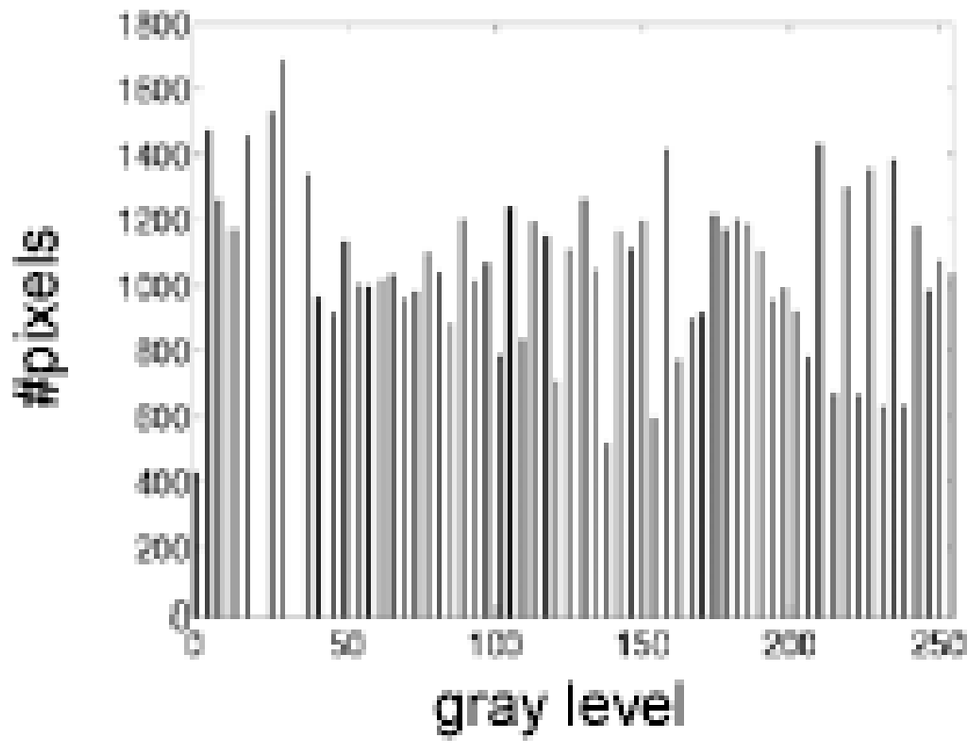


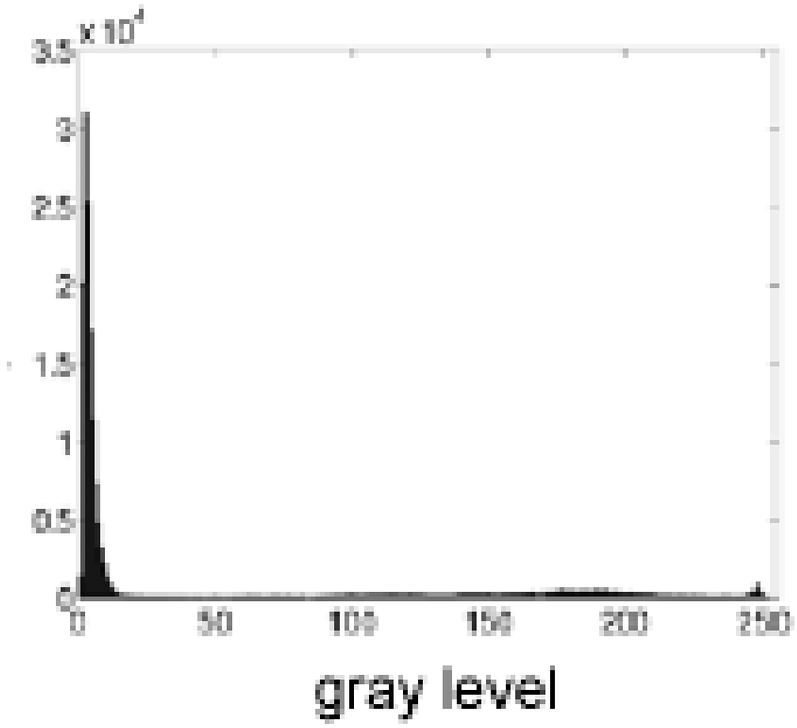
- Make it flat and spread it out
- This is a nonlinear operation

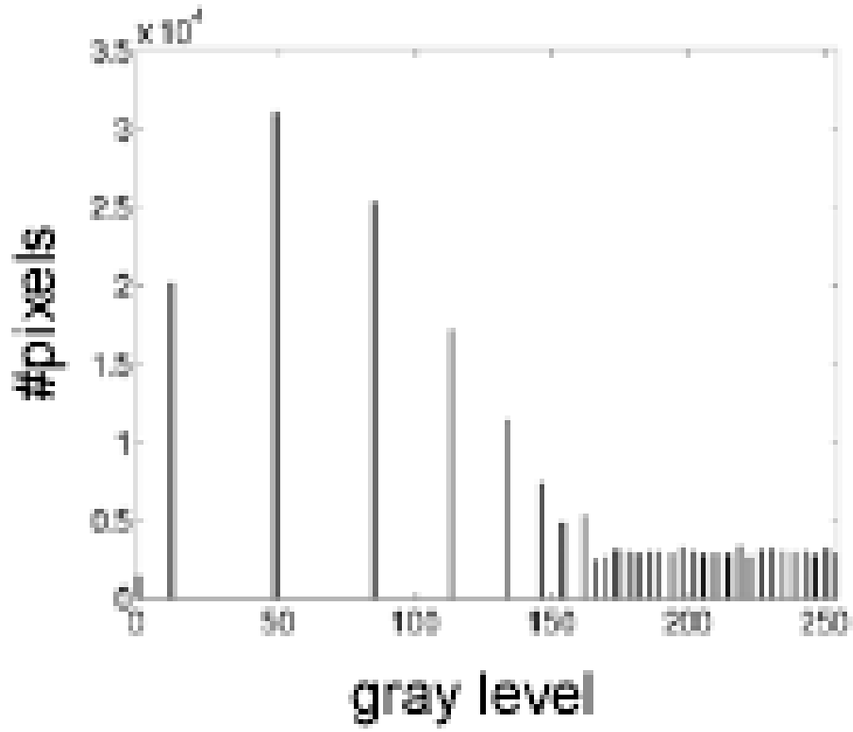

4000
3500
3000
2500
2000
1500
1000
500
0
0
50
100
150
200
250
gray level


#pixels
4000
3500
3000
2500
2000
1500
1000
500
0
0
50
100
150
200
250
gray level


1800
1600
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250
gray level


#pixels
1800
1600
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250
gray level


x 10^4
3.5
3
2.5
2
1.5
1
0.5
0
0
50
100
150
200
250
gray level


x 10^4
3.5
3
2.5
2
1.5
1
0.5
0
#pixels
0
50
100
150
200
250
gray level

Generalizations

- Adaptive Histogram Equalization
- Histogram Matching

  – Instead of flattening the histogram, make the histogram of A look like the histogram of another image B.

  – Procedure given two images:
    - Find transformation (T and U) that flatten images A, B
    - Required transformation is $U^{-1}(T(A))$

Color Constancy

- Metamerism, the perceiving of colors within two separate scenes

Retinex

- RETINEX: 're-tin-ex, 'ret-nex; noun; (pl) retinexes; from Medieval Latin retina and Latin cortic. Edwin Land coined word for his model of human color vision, combining the retina of the eye and the cerebral cortex of the brain.

- More specifically defined in image processing as a process that automatically provides visual realism to images.

Dynamic range compression

Retinex Image Enhancement—Technical

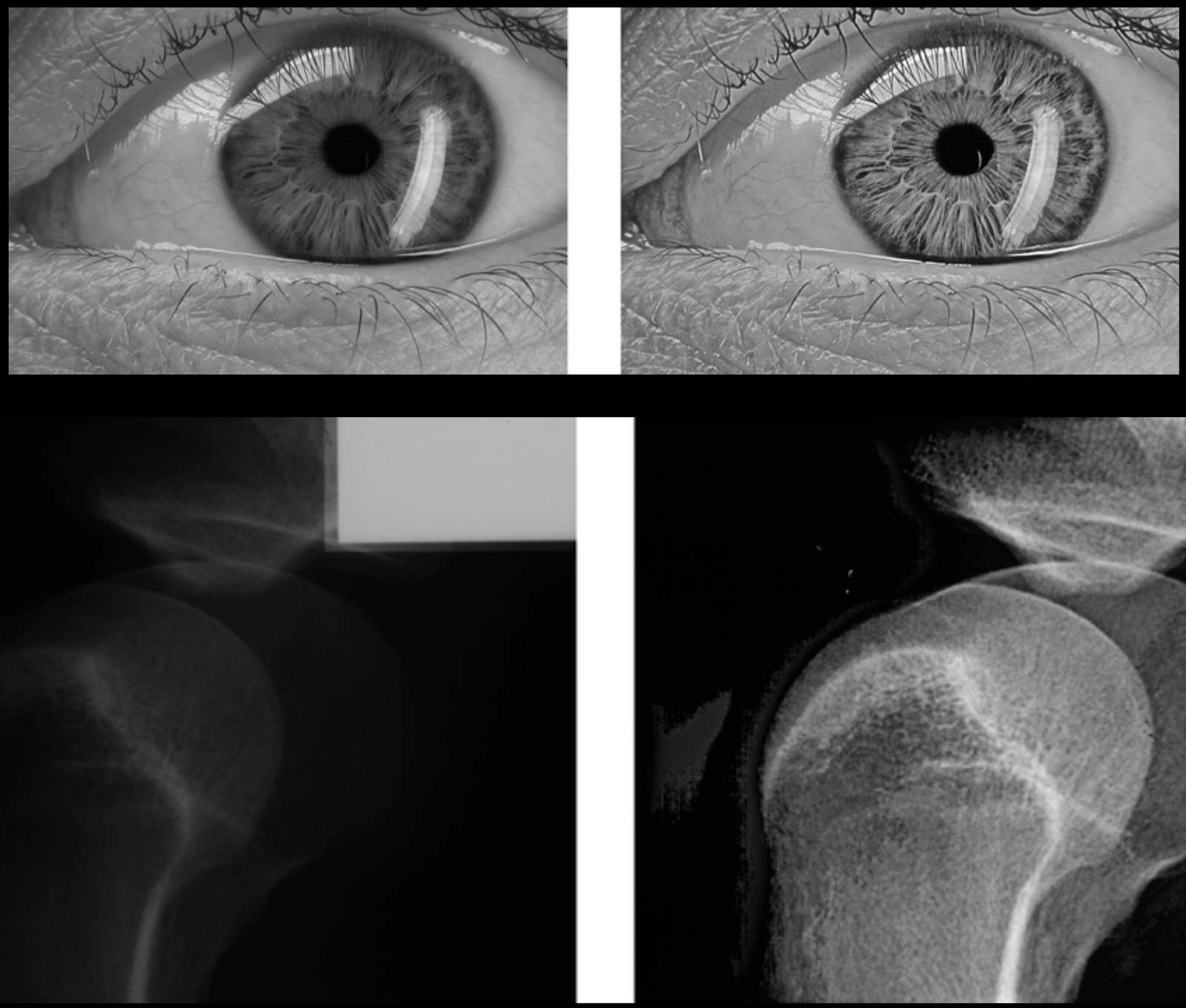
- The Retinex takes an input digital image I and produces an output image R on a pixel by pixel basis in the following manner:

$$\begin{aligned} R(x,y) &= \log\big(I(x,y)\big) - \log\big(I(x,y) * M(x,y)\big) \\ &= \log\left(\frac{I(x,y)}{I(x,y) * M(x,y)}\right) \end{aligned}$$

where $M(x,y) = \exp\big((x^2+y^2)/\sigma^2\big)$, σ is a constant which controls the extent of M, and $*$ represents spatial convolution.

https://dragon.larc.nasa.gov/retinex/

Improve contrast in grayscale display



And in color

Image Enhancement: Operating on the Pixels

- Make image "better" for a specific application
  - The idea of "better" is somewhat subjective

- We distinguish two domains:
  - Spatial or Pixel domain: $f(x, y)$ or $f(m, n)$
  - Frequency Domain: $F(w_x, w_y)$ or $F(u, v)$

- For this section: Pixel Domain
  - Operations on single pixel at a time
  - Operations on groups of pixels (neighborhoods)

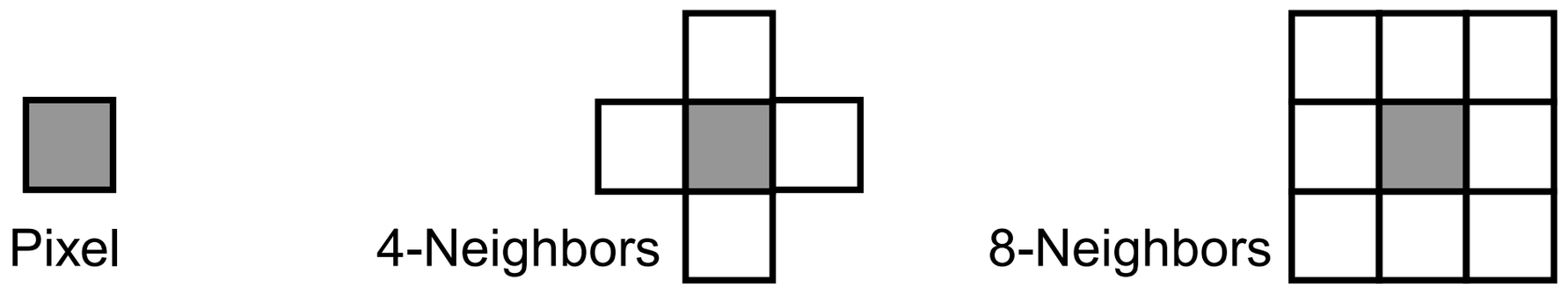
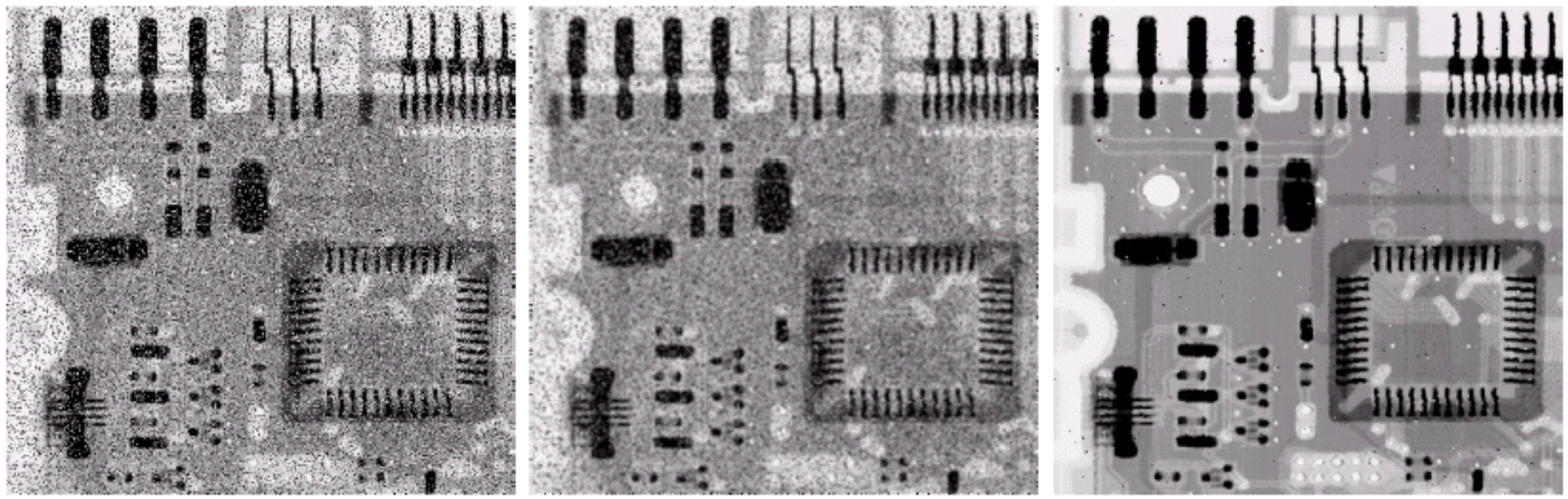


Image Enhancement:Spatial Filtering Operation

- Idea: Use a “mask” to alter pixel values according to local operation
- Aim: (De)-Emphasize some spatial frequencies in the image.



a b c

FIGURE 3.37 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Overview of Spatial Filtering

- Local linear oprations on an image

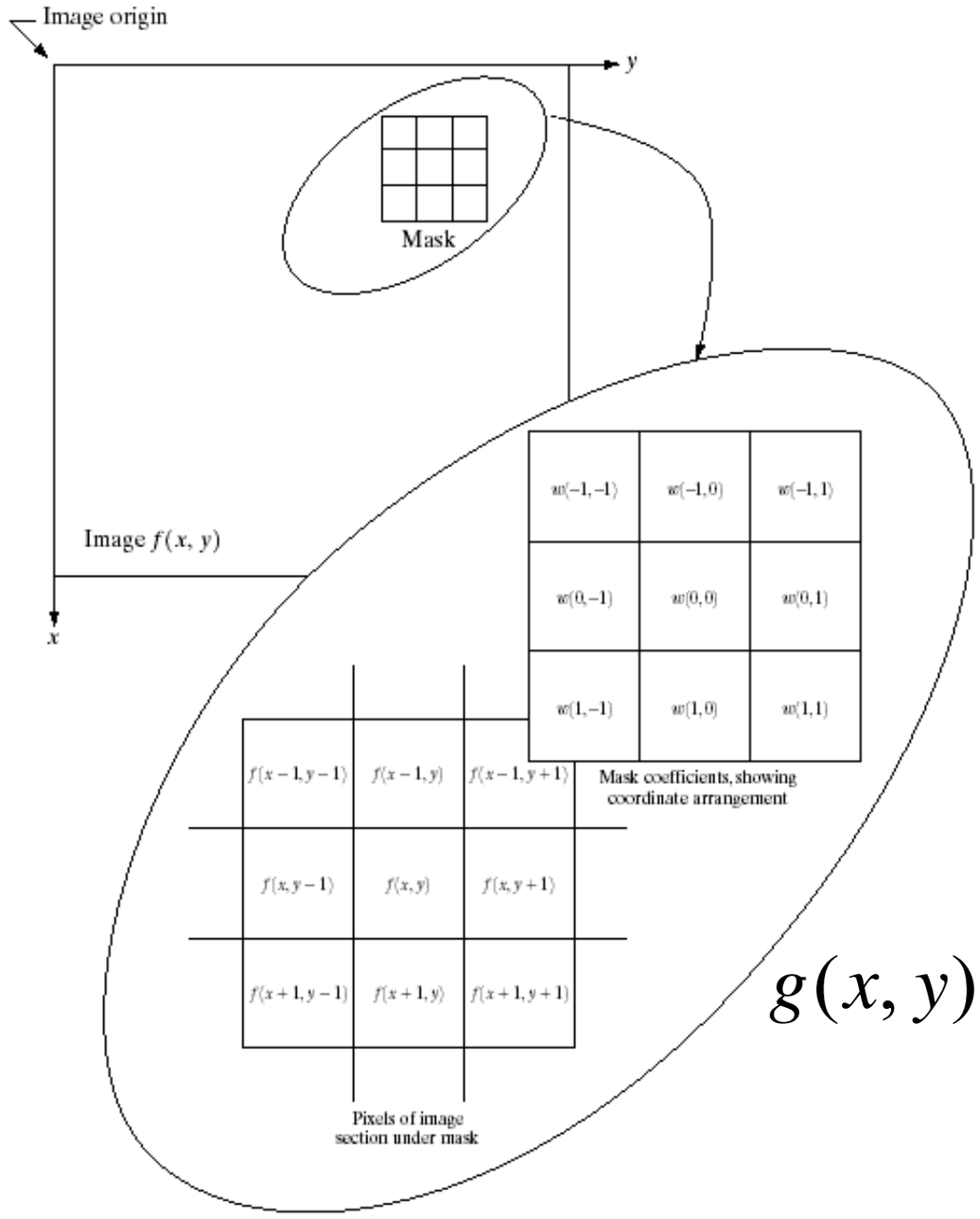


FIGURE 3.32 The mechanics of spatial filtering. The magnified drawing shows a 3×3 mask and the image section directly under it; the image section is shown displaced out from under the mask for ease of readability.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Input: f(x,y), Output: g(x,y)

$$g(x,y) = w_1 f(x-1, y-1) + w_2 f(x-1, y) + \cdots$$
$$w_8 f(x+1, y-1) + w_9 f(x+1, y+1)$$

Moving average

- We replace each pixel with a *weighted* average of its neighborhood
- The weights are called the *filter kernel*
- What are the weights for the average of a 3x3 neighborhood?

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

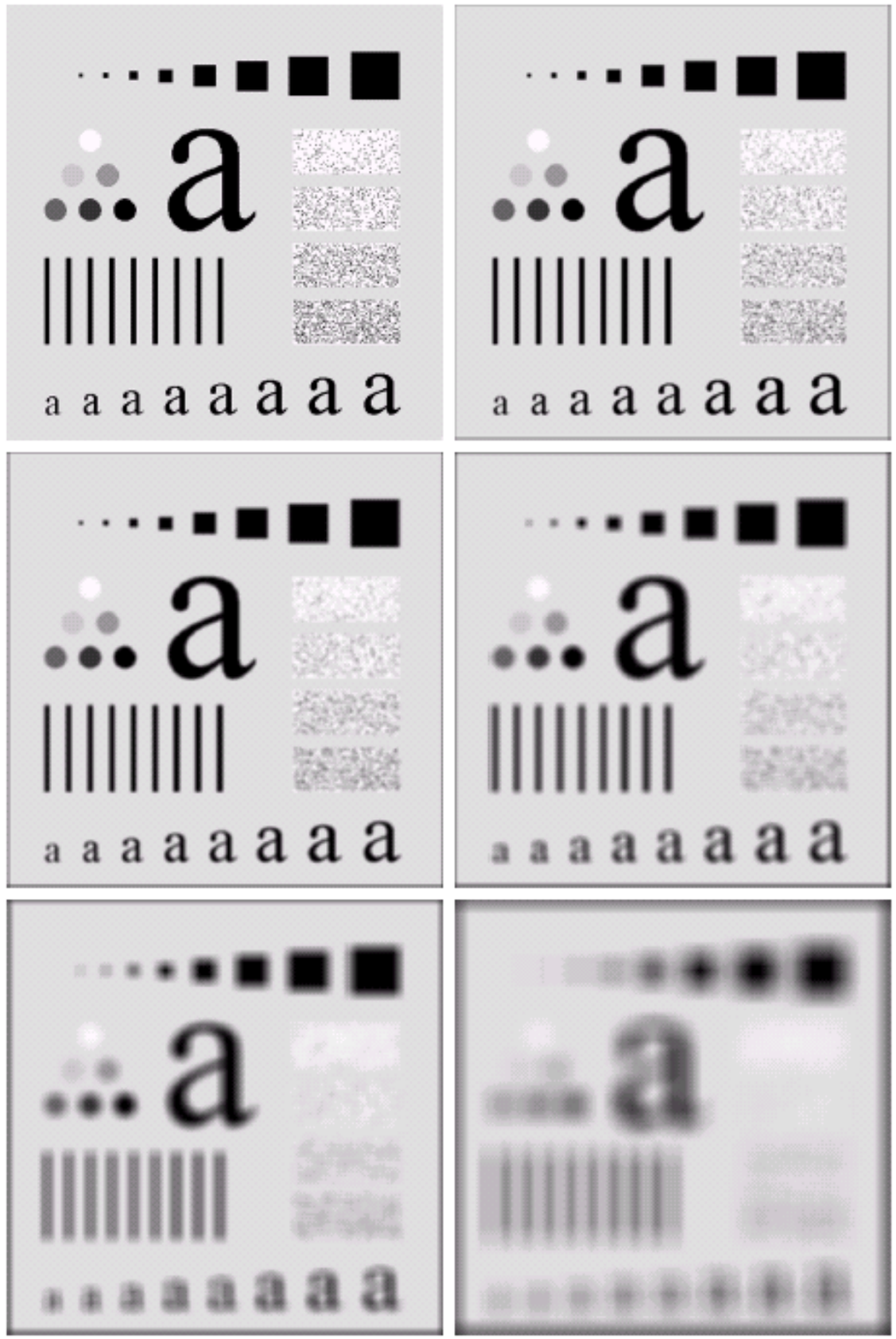
"box filter"

Spatial Filtering: Blurring

- Example

Averaging Mask:

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



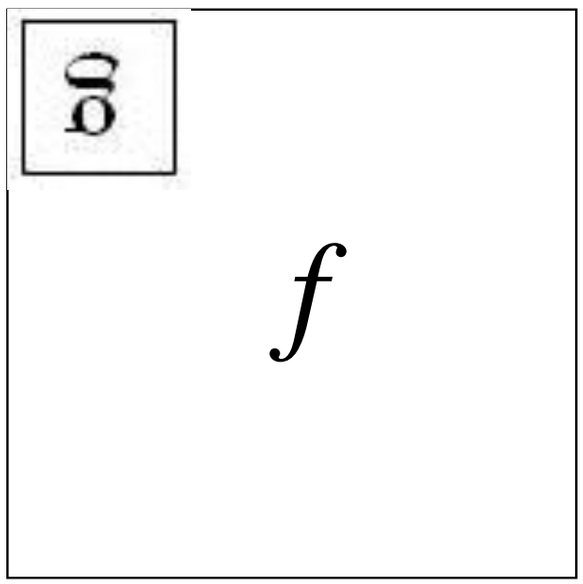
a b
c d
e f

FIGURE 3.35 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $n = 3, 5, 9, 15$, and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their gray levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

Convolution

- Let *f* be the image and *g* be the kernel. The output of convolving *f* with *g* is denoted *f* * *g*.

$$(f * g)[m,n] = \sum_{k,l} f[m-k, n-l]\, g[k,l]$$



- Convention: kernel is “flipped”
- MATLAB: conv2 vs. filter2 (also imfilter)

Convolution

Key properties

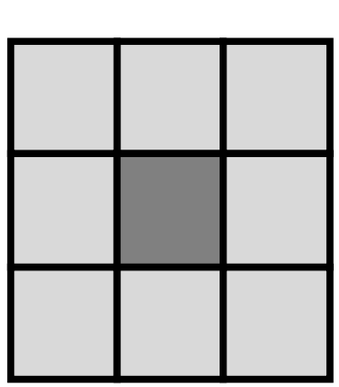
- **Linearity:** filter(f_1 + f_2) = filter(f_1) + filter(f_2)

- **Shift invariance:** same behavior regardless of pixel location: filter(shift(f)) = shift(filter(f))

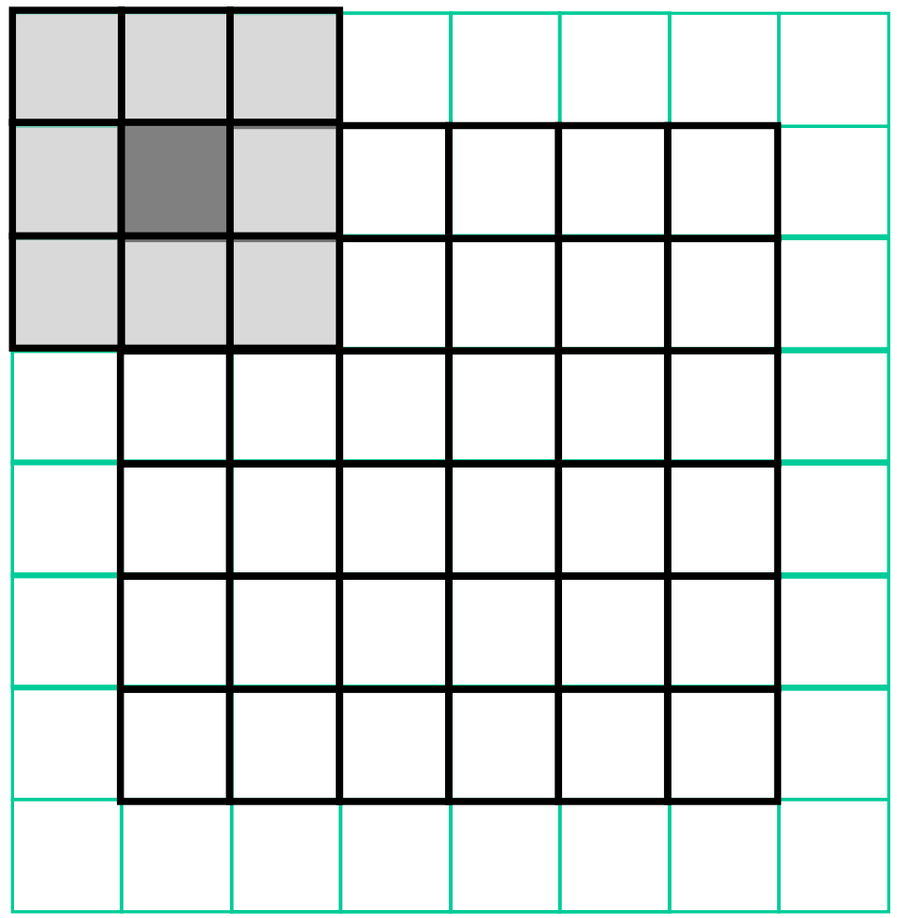
- Theoretical result: any linear shift-invariant operator can be represented as a convolution

What happens at the edge?

- An important point: Edge Effects
  - To compute all pixel values in the output image, we need to fill in a “border”



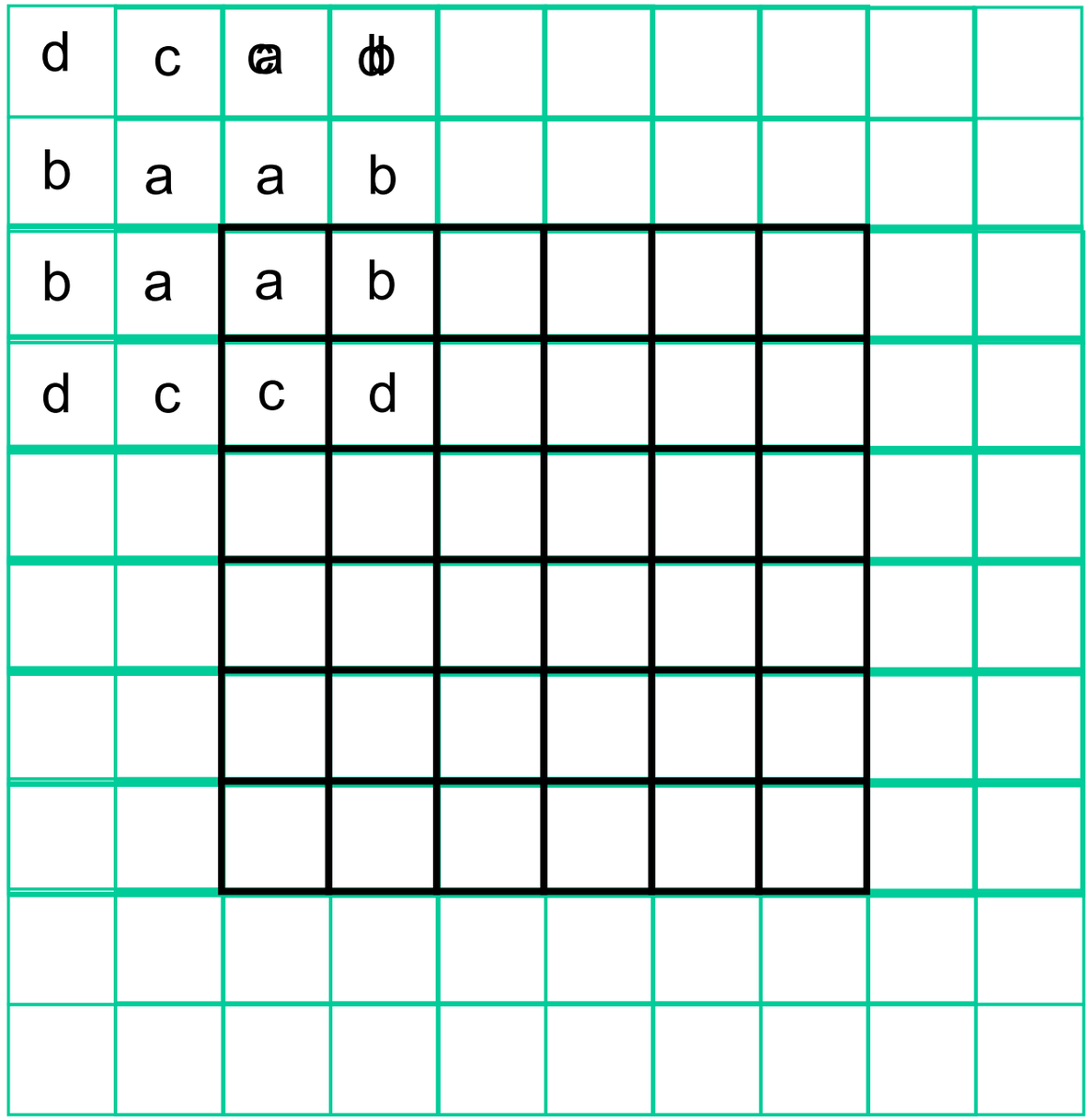
Mask dimension = 2M+1



Border dimension = M

Image Enhancement:Spatial Filtering Operation

- An important point: Edge Effects (Ex.: 5x5 Mask)
  - How to fill in a “border”
    - Zeros (Ringing)
    - Replication (Better)
    - Reflection (“Best”)



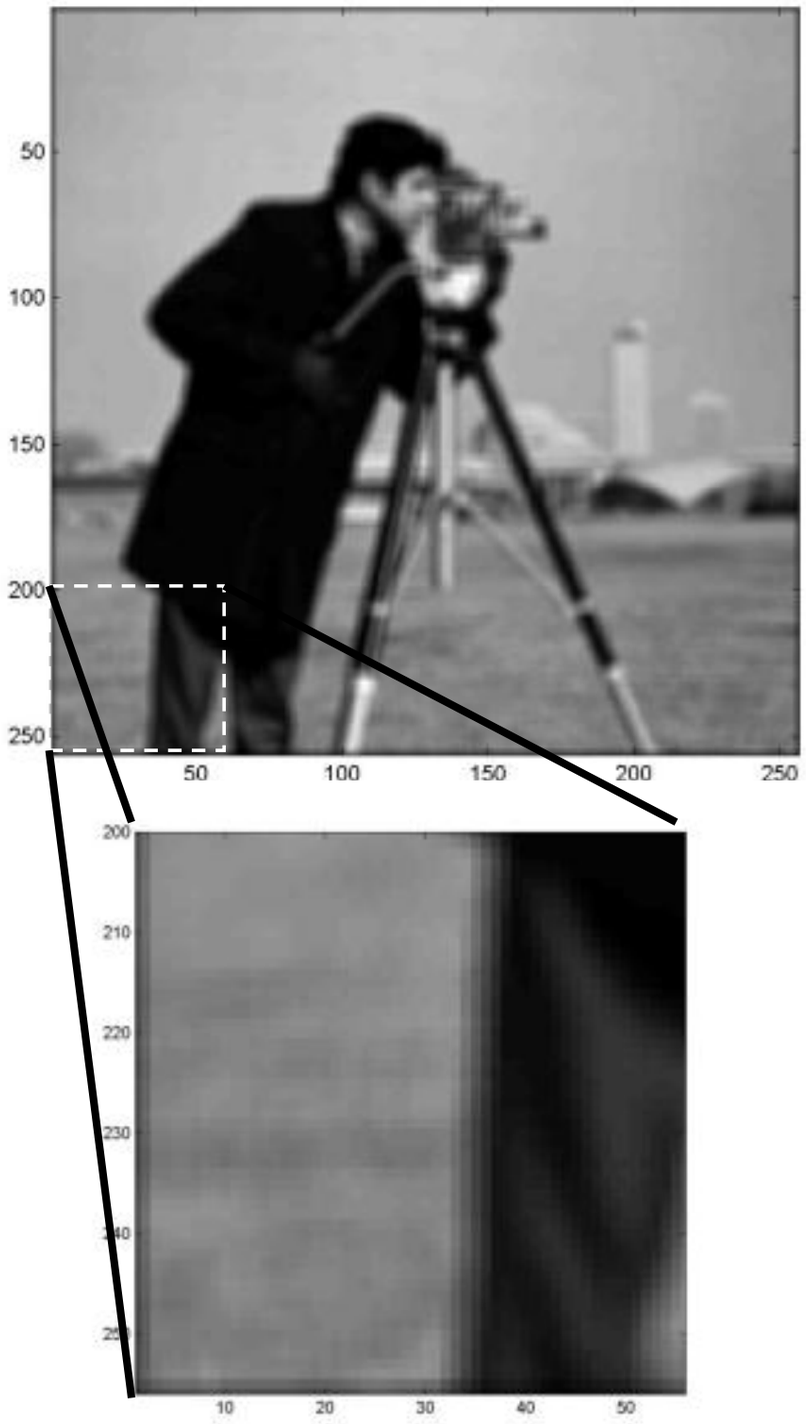
- Procedure:
  - Replicate row-wise
  - Replicate column-wise
  - Apply filtering
  - Remove borders

Implementation

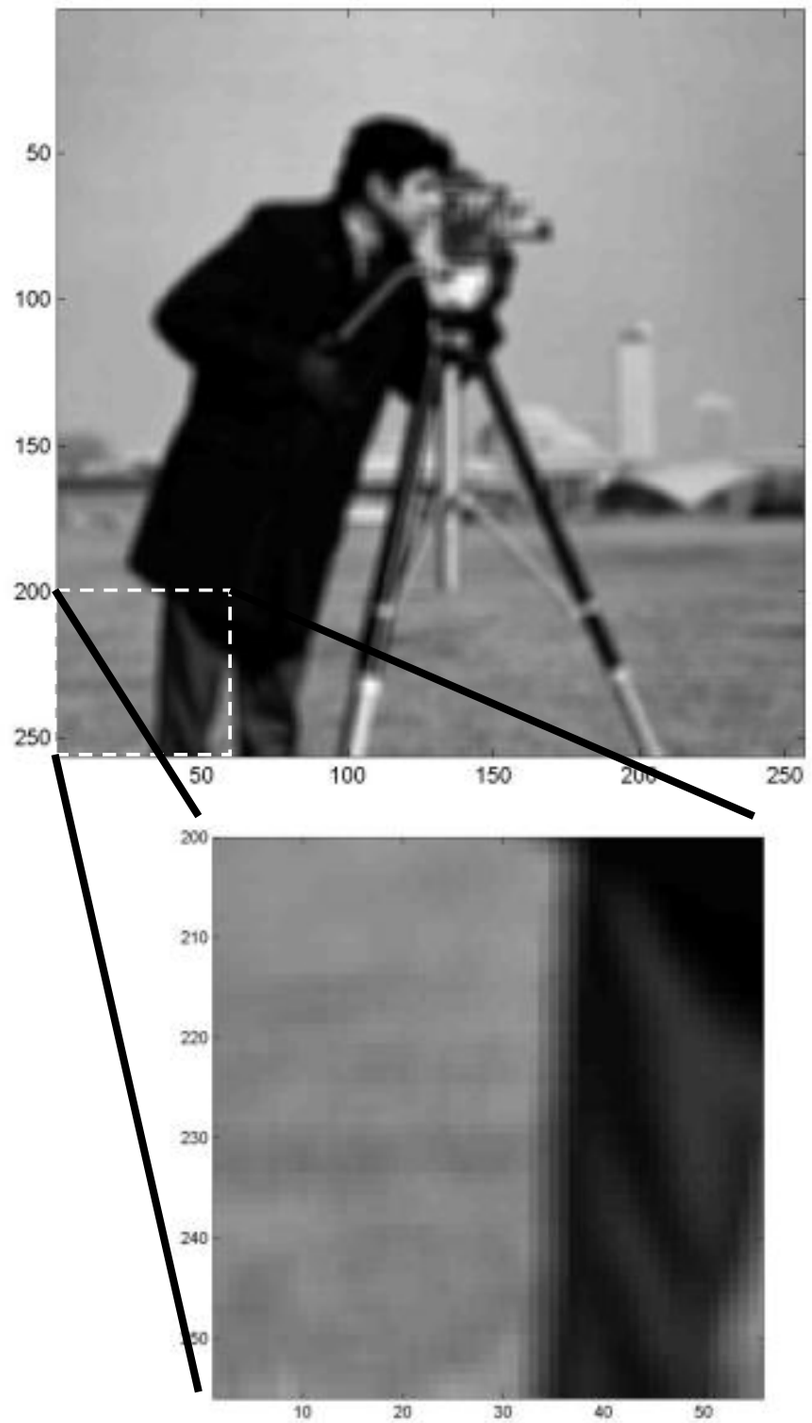
- What about near the edge?
  - the filter window falls off the edge of the image
  - need to extrapolate
  - methods (MATLAB):
    - clip filter (black): imfilter(f, g, 0)
    - wrap around: imfilter(f, g, ‘circular’)
    - copy edge: imfilter(f, g, ‘replicate’)
    - reflect across edge: imfilter(f, g, ‘symmetric’)

Image Enhancement:Spatial Filtering Operation

5x5 Blurring with 0-padding



5x5 Blurring with reflected padding



Examples:

Original

0	0	0
0	1	0
0	0	0

?

Original

0	0	0
0	1	0
0	0	0

Filtered
(no change)

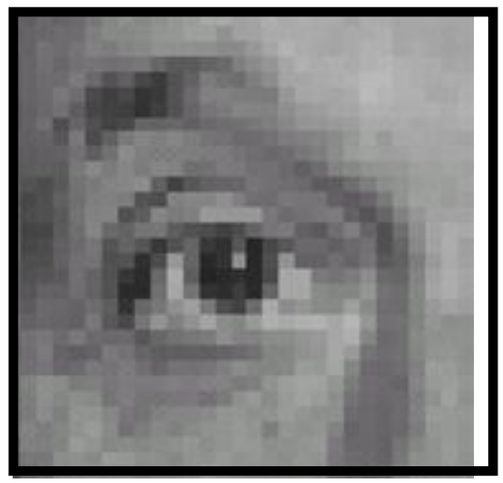
Original

0	0	0
0	0	1
0	0	0

?

Original

0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

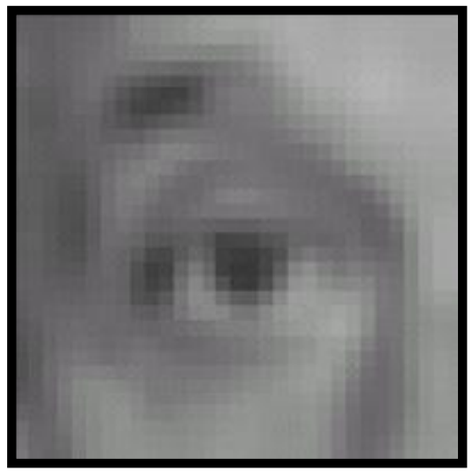
Original

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

?

Original

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



Blur (with a box filter)

Original

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad ?$$

(Note that filter sums to 1)

Original

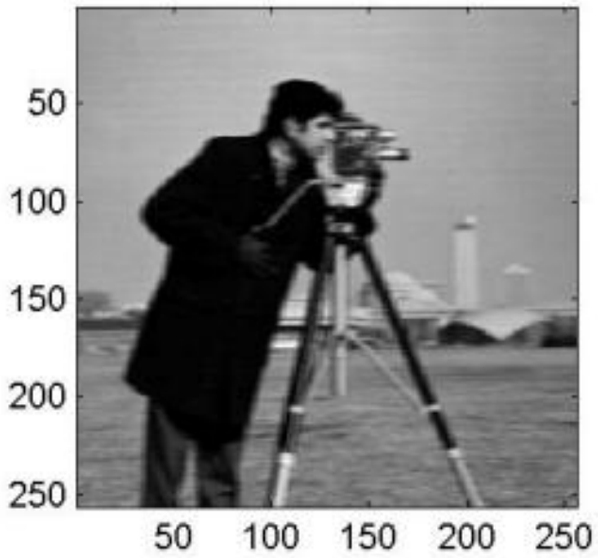
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

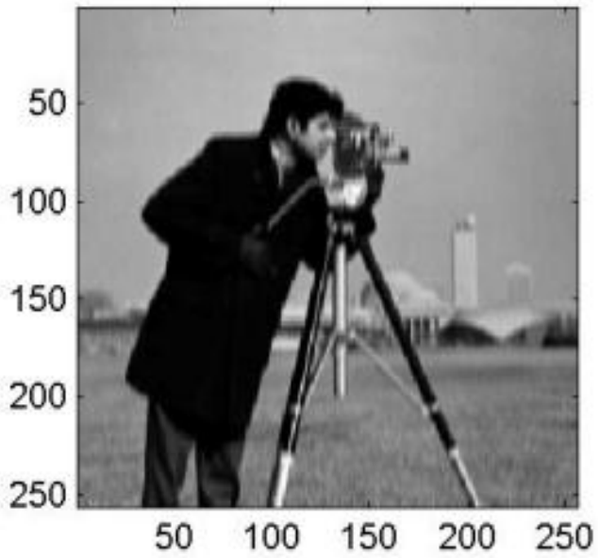
Sharpening filter

- Accentuates differences with local average

Examples of some other smoothing or “low-pass” filters:

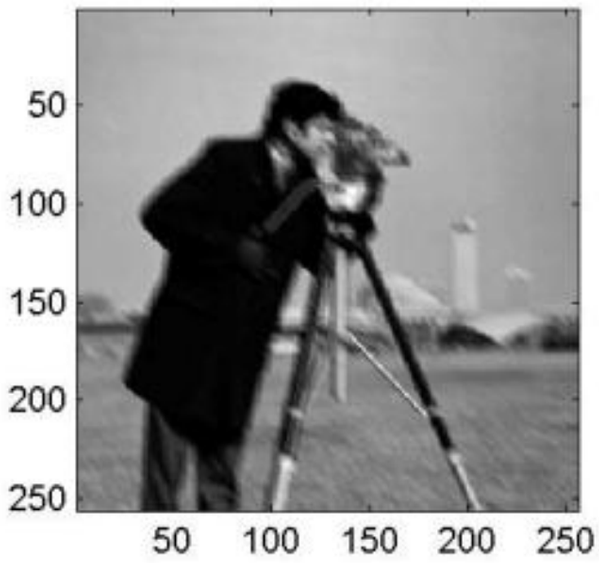
$$\frac{1}{5}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

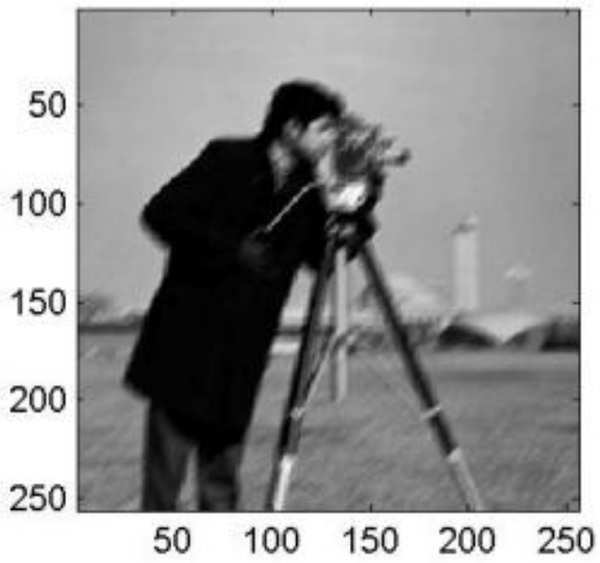




$$\frac{1}{5}\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\frac{1}{5}\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

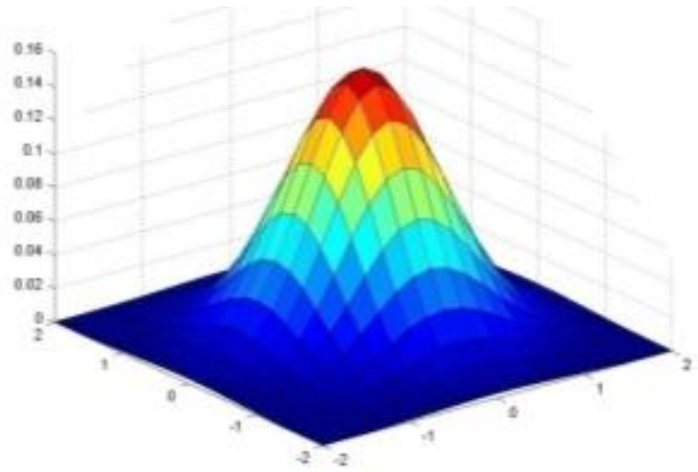


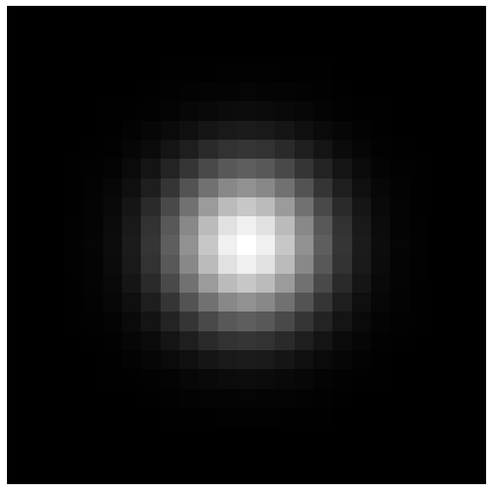


$$\frac{1}{5}\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Gaussian Kernel

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$





0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

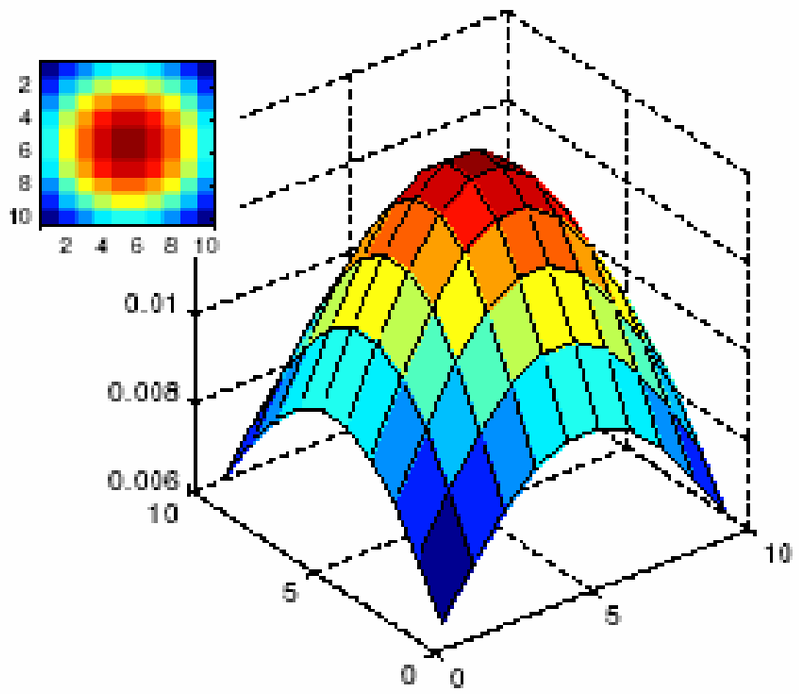
5 x 5, $\sigma = 1$

fspecial('gauss',5,1)

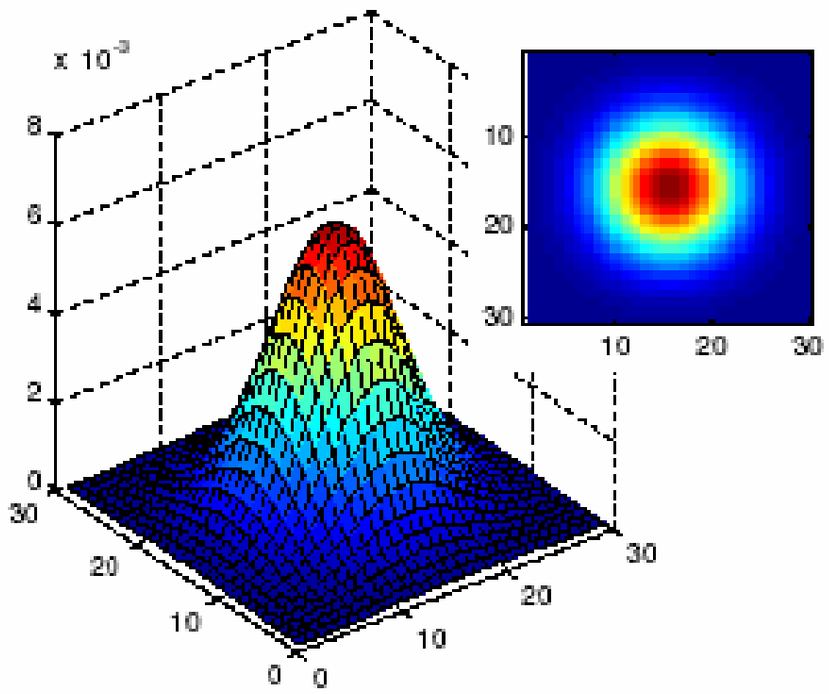
- Constant factor at front makes volume sum to 1 (can be ignored, as we should re-normalize weights to sum to 1 in any case)

Choosing kernel width

- Gaussian filters have infinite support, but discrete filters use finite kernels

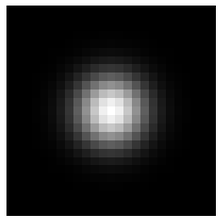


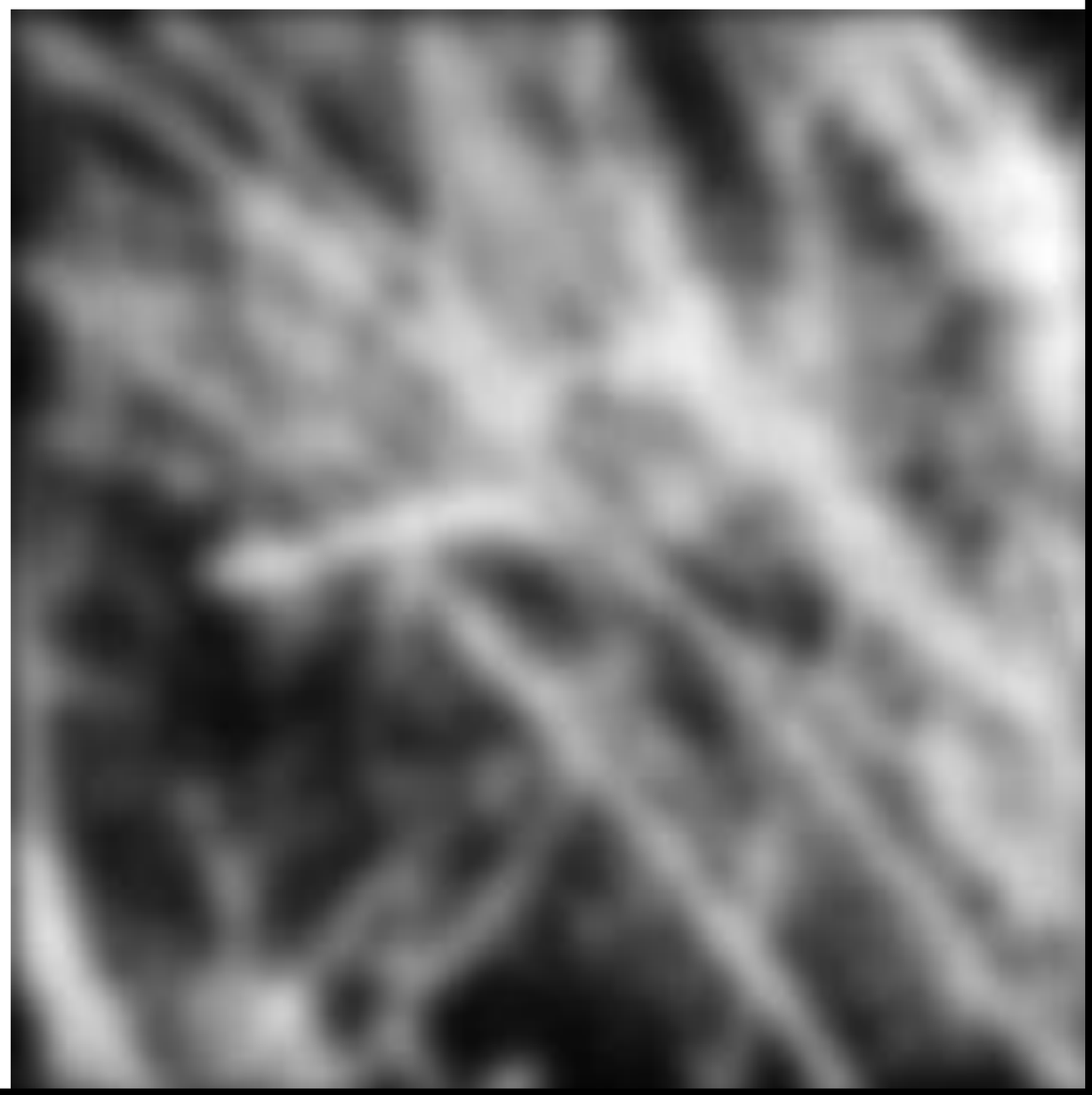
σ = 5 with 10x10 kernel



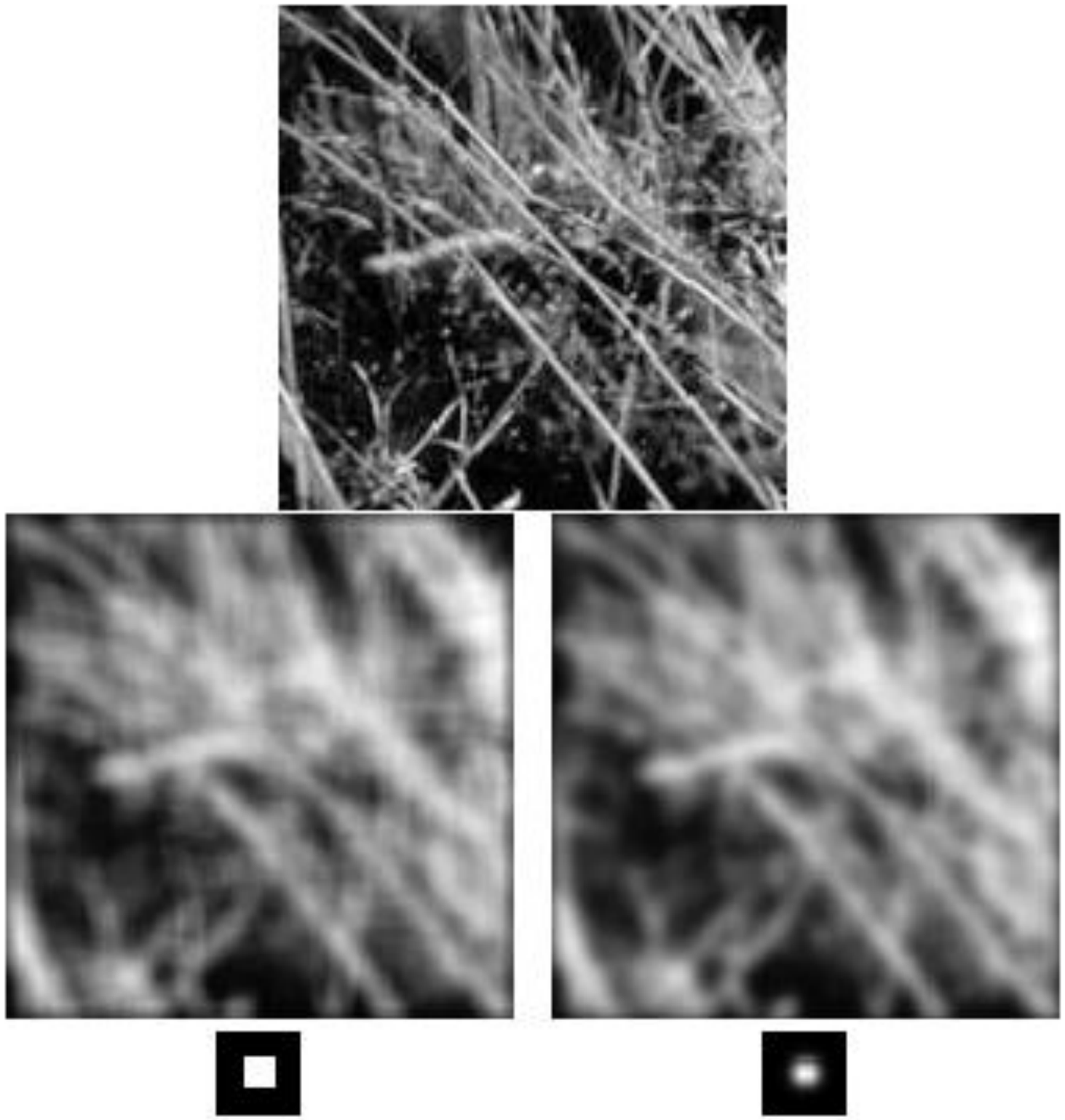
σ = 5 with 30x30 kernel

Example: Smoothing with a Gaussian





Mean vs. Gaussian filtering



Gaussian filters

- Remove “high-frequency” components from the image (low-pass filter)
- Convolution with self is another Gaussian
  - So can smooth with small-width kernel, repeat, and get same result as larger-width kernel would have
  - Convolving two times with Gaussian kernel of width σ is same as convolving once with kernel of width $\sigma\sqrt{2}$
- *Separable* kernel
  - Factors into product of two 1D Gaussians

Use this to sharpen!

–

=

Let’s add it back:

+ α

=

More on Linear Operations: Sharpening Filters

- Sharpening filters use masks that typically have + and – numbers in them.

- They are useful for highlighting or enhancing details and high-frequency information (e.g. edges)

- They can (and often are) based on derivative-type operations in the image (whereas smoothing operations were based on “integral” type operations)

Derivatives

Differentiation and convolution

• Recall, for 2D function, f(x,y):

$$\frac{\partial f}{\partial x} = \lim_{\varepsilon \to 0} \left(\frac{f(x+\varepsilon, y)}{\varepsilon} - \frac{f(x,y)}{\varepsilon} \right)$$

• We could approximate this as

$$\frac{\partial f}{\partial x} \approx \frac{f(x_{n+1}, y) - f(x_n, y)}{\Delta x}$$

• This is linear and shift invariant, so must be the result of a convolution.

• which is obviously a convolution with kernel

-1	1

Derivative-type Filters

$$\begin{bmatrix} -1 & 1 \end{bmatrix} \longrightarrow \frac{\partial f}{\partial \mathrm{x}} \approx f(x+1, y) - f(x, y)$$

$$\begin{bmatrix} -1 \\ 1 \end{bmatrix} \longrightarrow \frac{\partial f}{\partial \mathrm{y}} \approx f(x, y+1) - f(x, y)$$

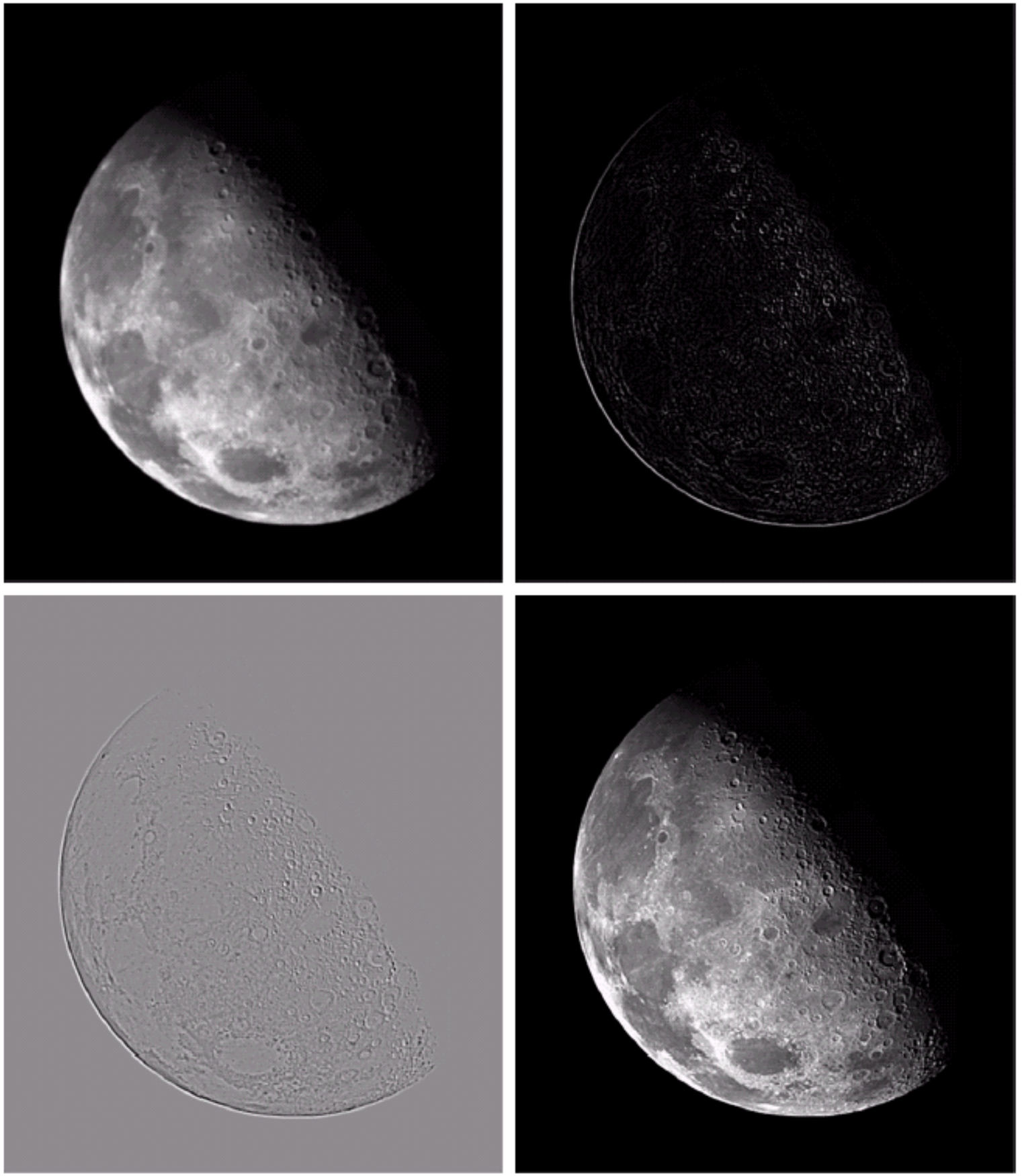
$$\begin{bmatrix} 1 & -2 & 1 \end{bmatrix} \longrightarrow \frac{\partial^2 f}{\partial \mathrm{x}^2} \approx f(x+1, y) - 2f(x, y) + f(x-1, y)$$

$$\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} \longrightarrow \frac{\partial^2 f}{\partial \mathrm{y}^2} \approx f(x, y+1) - 2f(x, y) + f(x, y-1)$$

Laplacian: $\nabla^2 f = \dfrac{\partial^2 f}{\partial \mathrm{x}^2} + \dfrac{\partial^2 f}{\partial \mathrm{y}^2} \Rightarrow \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

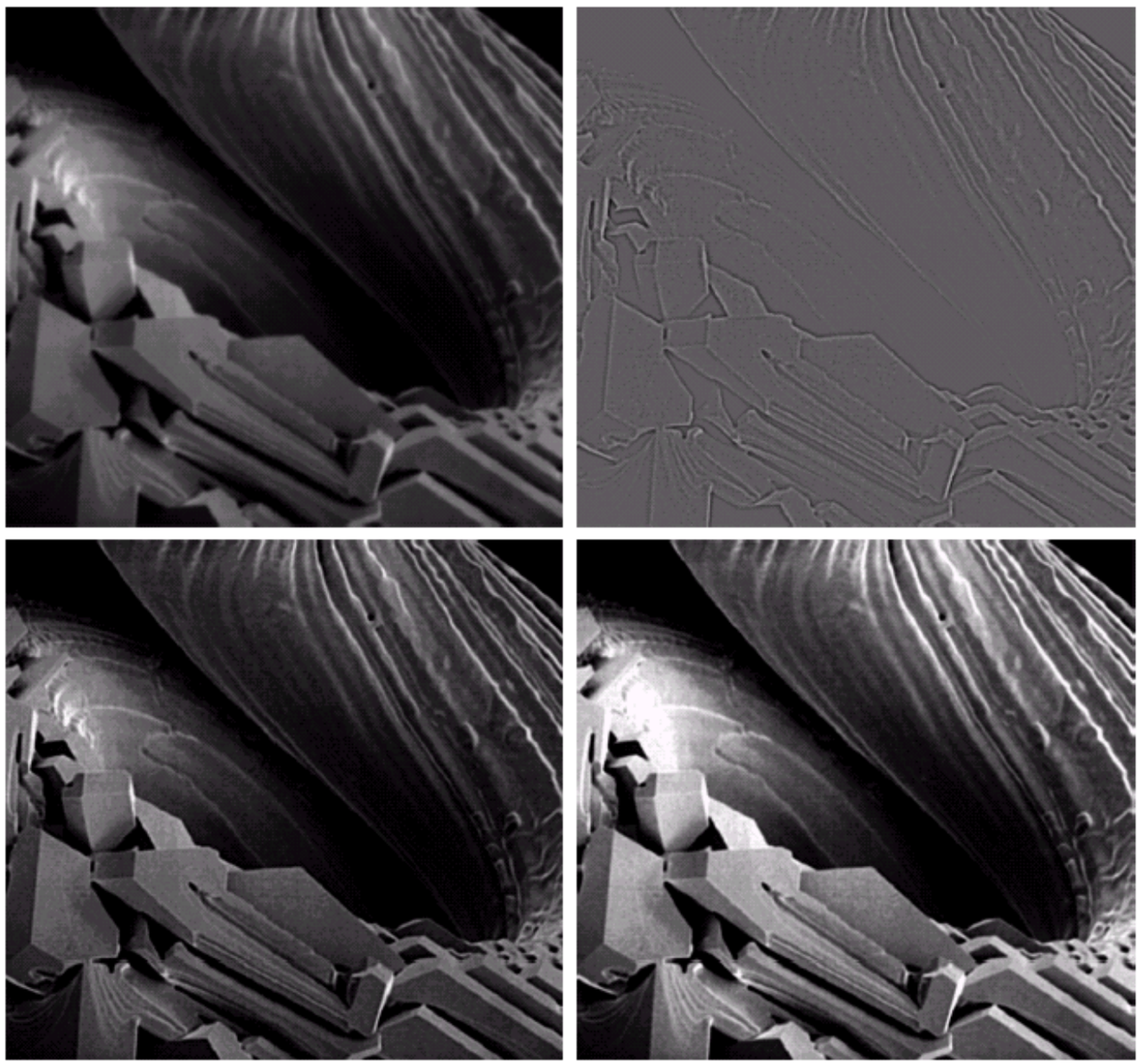
a b
c d

FIGURE 3.40
(a) Image of the North Pole of the moon.
(b) Laplacian-filtered image.
(c) Laplacian image scaled for display purposes.
(d) Image enhanced by using Eq. (3.7-5).
(Original image courtesy of NASA.)



Sharpening Using the Laplacian Filter

$$g(x,y) = A\,f(x,y) - \nabla^2 f(x,y) \longrightarrow \begin{bmatrix} -1 & -1 & -1 \\ -1 & A+8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$



a b
c d

FIGURE 3.43
(a) Same as Fig. 3.41(c), but darker. (a) Laplacian of (a) computed with the mask in Fig. 3.42(b) using $A = 0$. (c) Laplacian enhanced image using the mask in Fig. 3.42(b) with $A = 1$. (d) Same as (c), but using $A = 1.7$.

Boosting High Frequencies

Laplacian of Gaussian

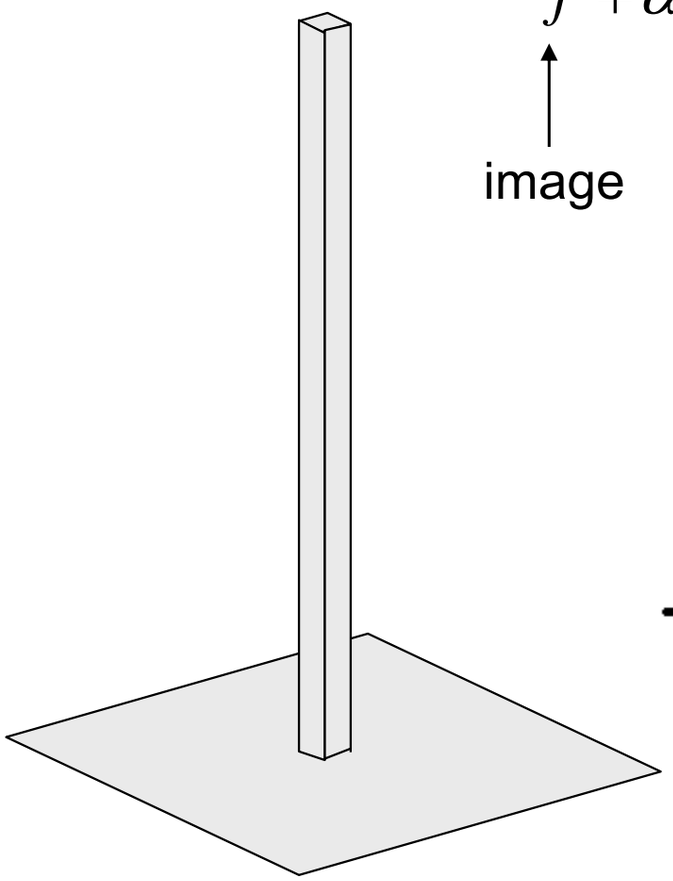
Gaussian Unsharp Mask Filter

$$f + \alpha(f - f * g) = (1+\alpha)f - \alpha\, f * g = f * ((1+\alpha)e - g)$$

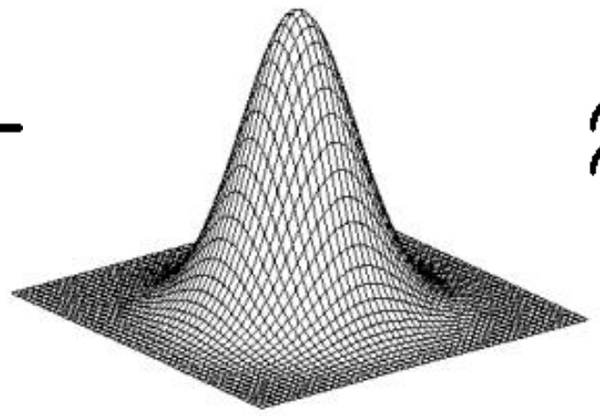
image

blurred image

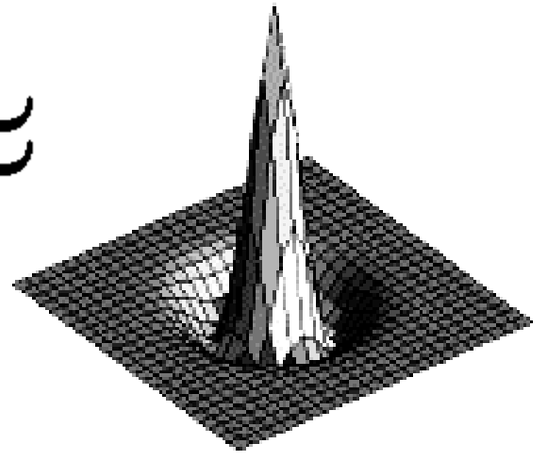
unit impulse (identity)



unit impulse



Gaussian

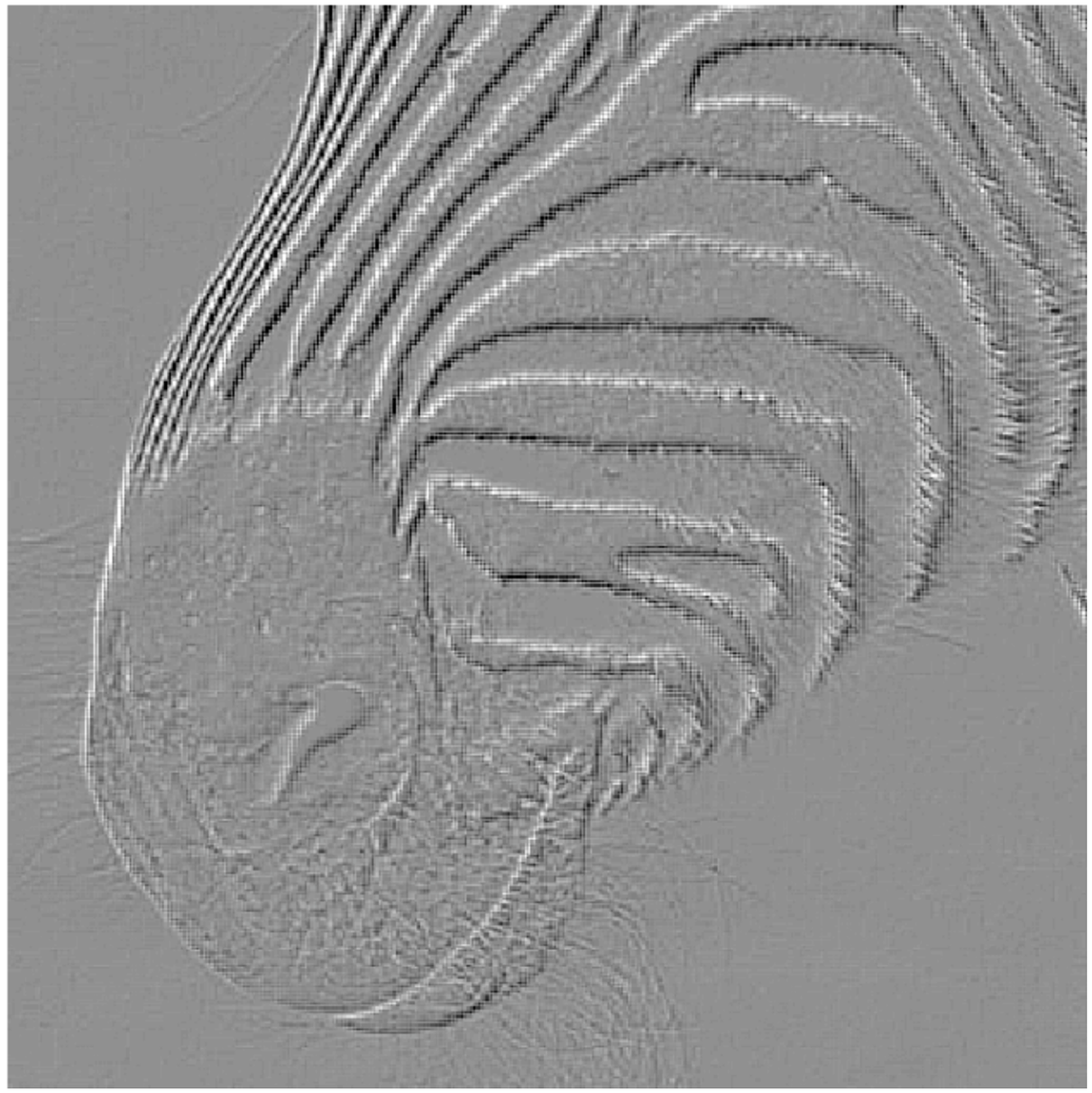


Laplacian of Gaussian

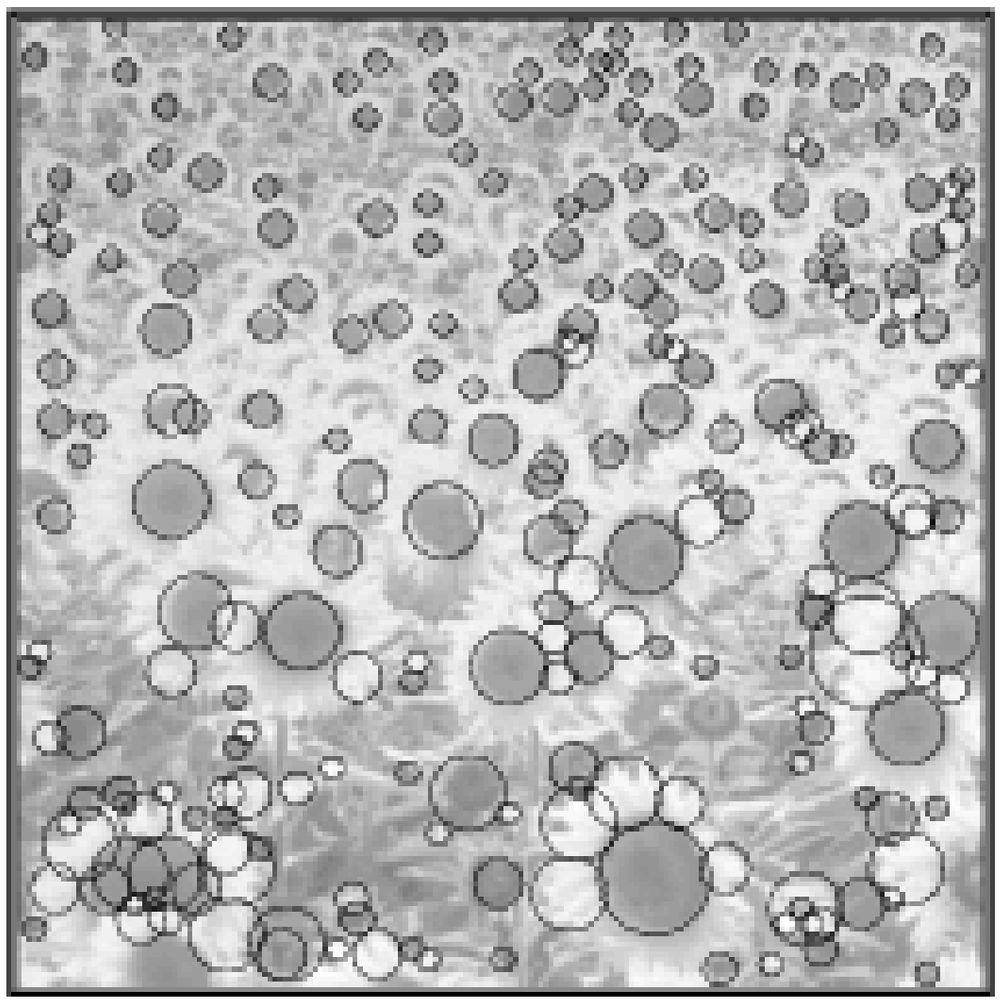
Edge detection

Original

LoG-filtered



Blob detection



Projects suggestions:

- Retinex
- Histogram equalization
- Wiener filtering
- Object recognition
- Tracking
- Motion correction
-