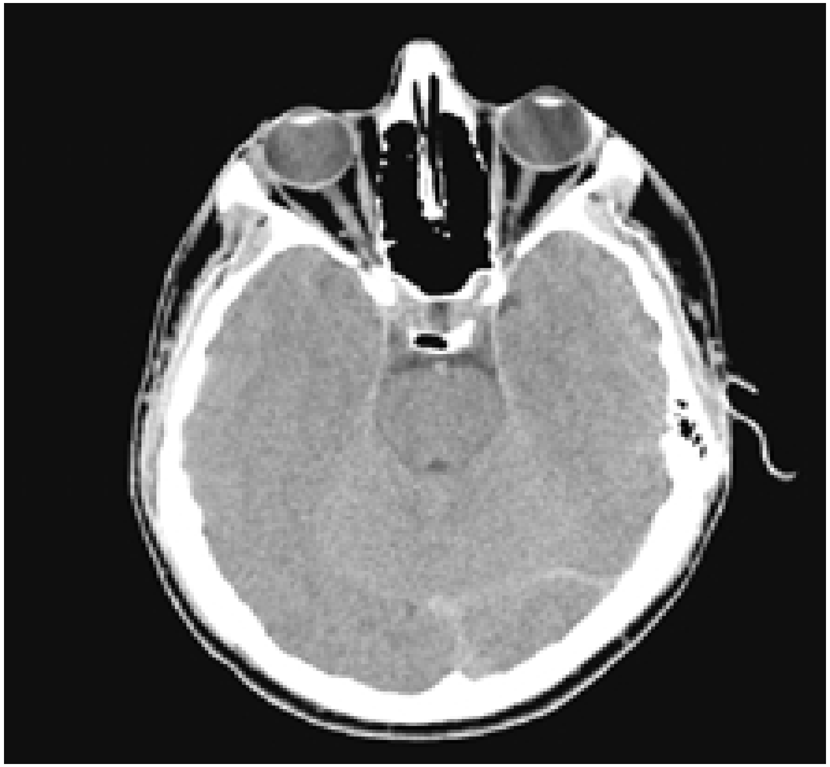
IMAGE FILTERING AND ENHANCEMENT

- Multidimensional signals
- Spatial convolution
- Smoothing spatial filters
- Smoothing spatial filters – Gaussian filter
- Smoothing spatial filter versus Gaussian filter
- Using the second-order derivative for image sharpening - the Laplacian
- How to avoid negative values of pixels?
- (Color images)
- (Intensity transformations)
- (Median smoothing spatial filters
- (Smoothing spatial filters (examples)
- (Using the second-order derivative for image sharpening – joint mask
- (Unsharp masking and high-boost filtering

Multidimensional signals

- Multidimensional signals (images) *f(x, y)* depend on several variables such as spatial coordinates *(x, y)*



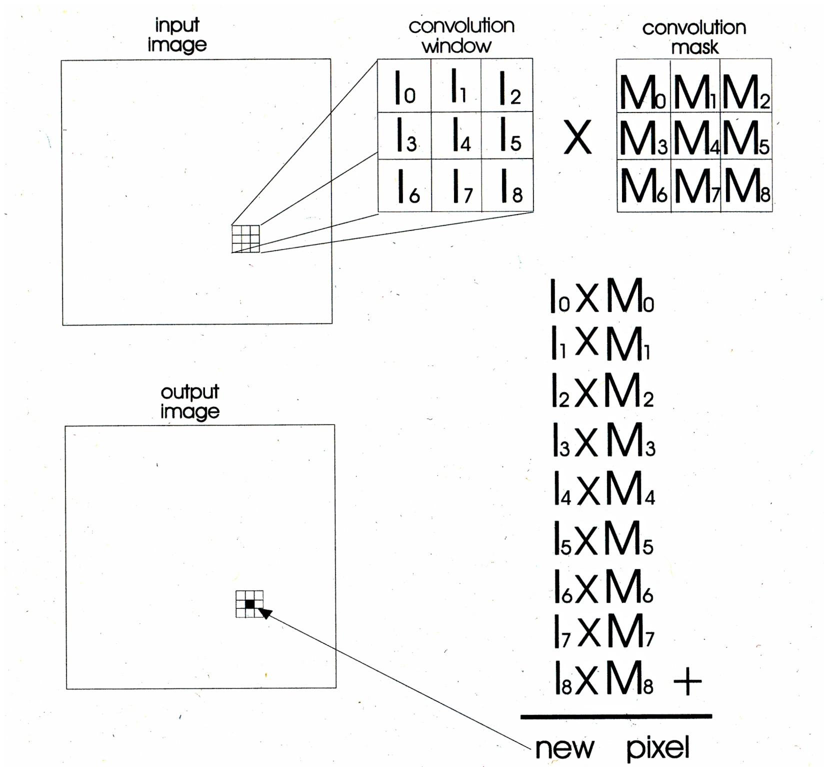
$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix}$$

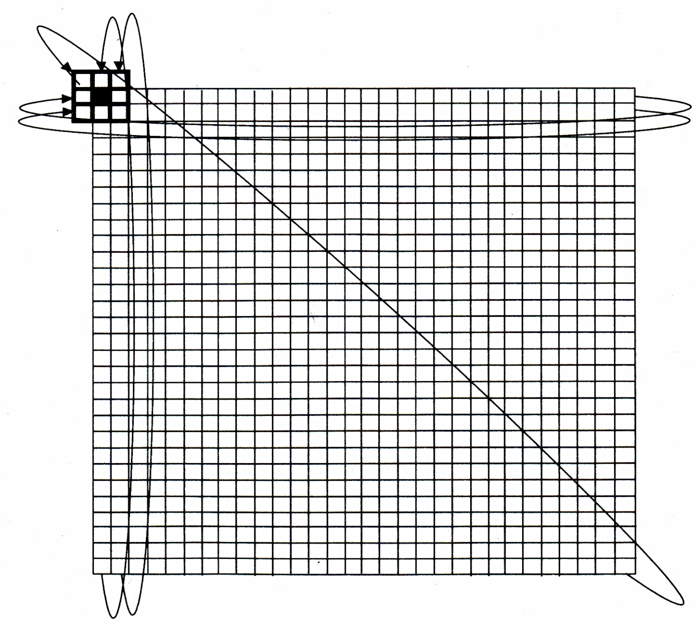
(Gonzales, Woods)

Spatial convolution

- Convolution (*convolution kernel*, *impulse response*, *spatial mask*, *template*)
  g(x,y) = w(s,t) X f(x,y)

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





Smoothing spatial filters

- Smoothing (blurring)
  - Rearranging intensities in image with the aim to smooth sharp peaks
  - Filtering using linear low-pass filters, positive coefficients of the mask
  - Smoothing using moving average (a box filter), smoothing using weighted moving average

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

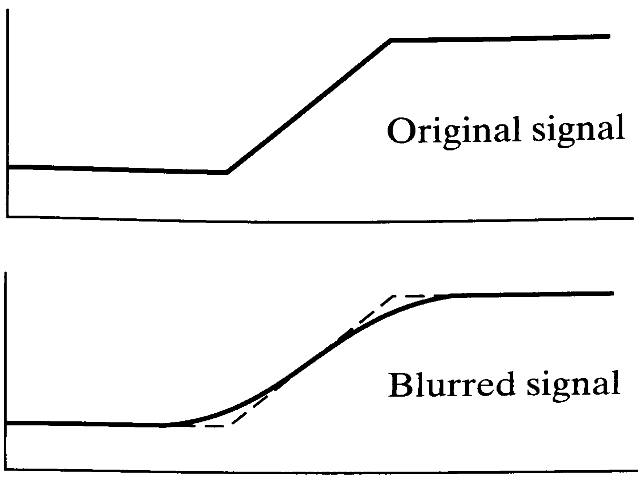
$$g(x, y) = \frac{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t)}{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t)}$$

- **Color images**
  → the same operation (smoothing, sharpening, ...) is performed **in each channel**

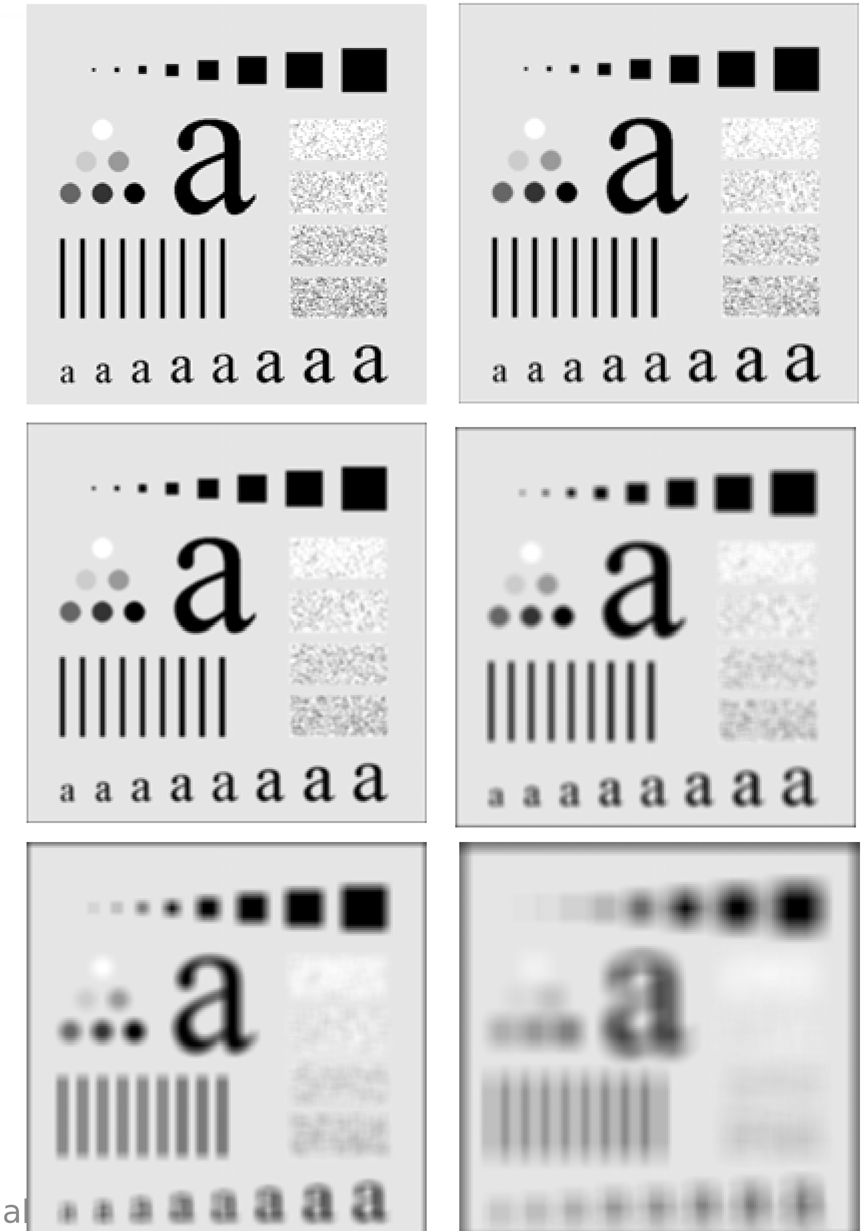
(Gonzales, Woods)

Smoothing spatial filters

- Results of smoothing with square averaging filter (sizes of masks,
  m = 3, 5, 9, 15, 35)



(Gonzales, Woods)



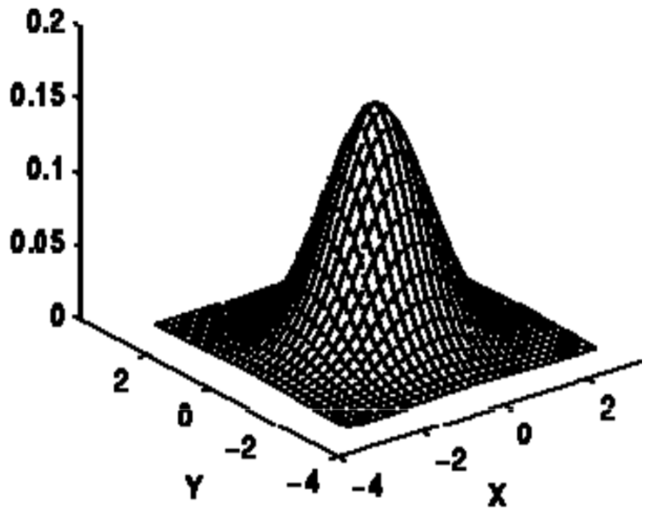
Smoothing spatial filters – Gaussian filter

- Gaussian filter

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

$$\downarrow$$

$$\begin{bmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{bmatrix}$$



$$G[x,y] = \frac{e^{\frac{-(x^2+y^2)}{2\sigma^2}}}{2\pi\sigma^2}$$

$\frac{1}{273}$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad R = \frac{1}{9}\sum_{i=1}^{9} z_i$$

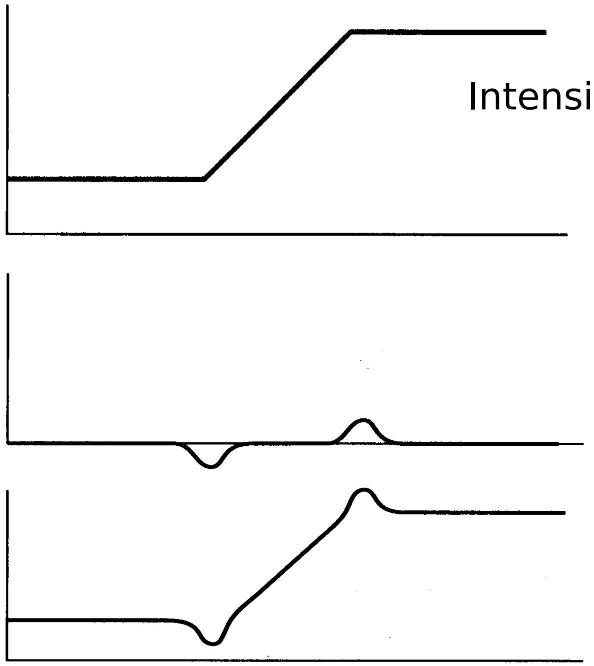
Smoothing spatial filter versus Gaussian filter

Using the second-order derivative for image sharpening – the Laplacian

- Sharpening
  - Rearranging intensities in image with the aim to rise differences in intensities of the neighboring pixels to emphasize tiny details
  - Filtering using high-pass filters, second order derivative, central coefficients positive and neighboring coefficients negative (or vice versa), sum of the coefficients equals zero

(x axis)

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



Using the second-order derivative for image sharpening – the Laplacian

- The Laplacian operator
  (2D second-order derivative)

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

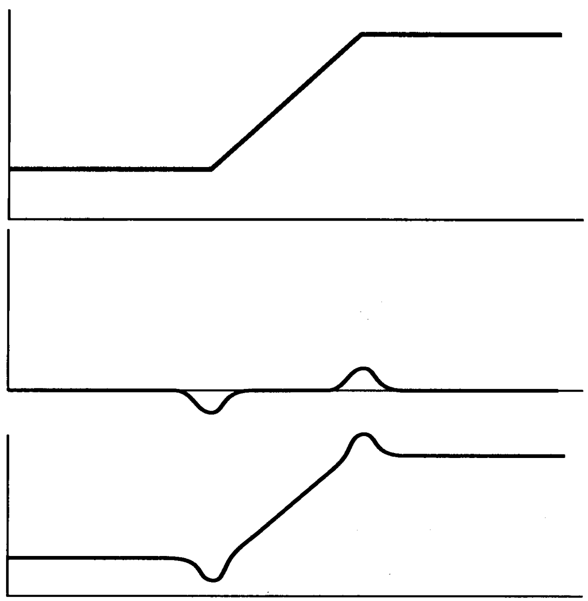
$$\frac{\partial^2 f}{\partial^2 x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

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

$$\nabla^2 f = \left[f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1)\right] - 4f(x, y)$$

Using the second-order derivative for image sharpening – the Laplacian

- The Laplacian operator



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

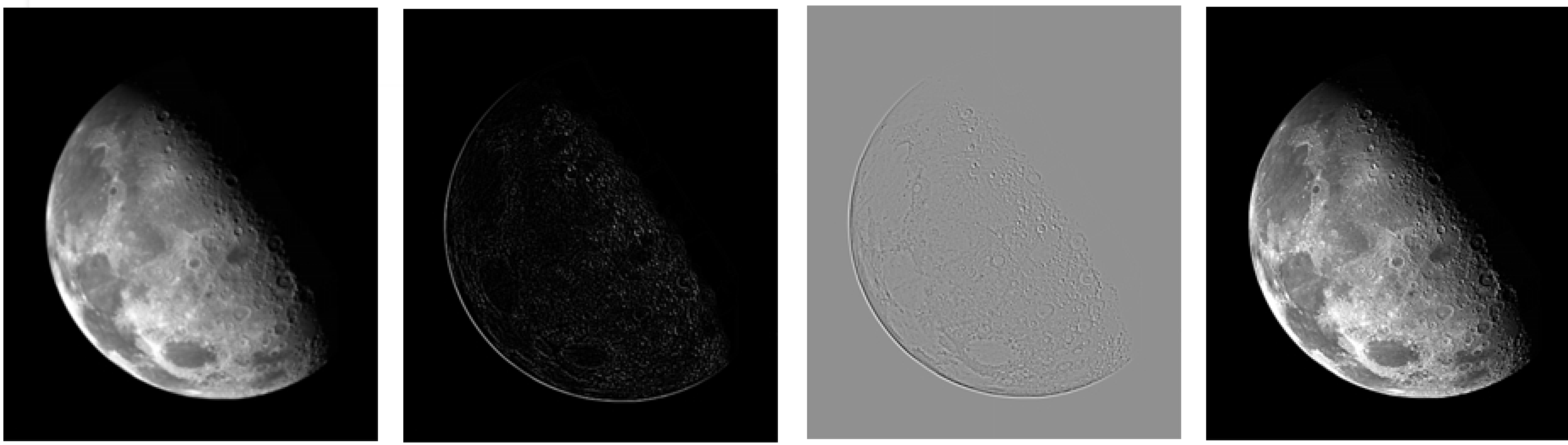
−1	−1	−1
−1	8	−1
−1	−1	−1

$$g(x, y) = \begin{cases} f(x, y) - \nabla^2 f(x, y) & \text{if the center coefficient of the Laplacian mask is negative} \\ f(x, y) + \nabla^2 f(x, y) & \text{if the center coefficient of the Laplacian mask is positive.} \end{cases}$$

(Gonzales, Woods)

Using the second-order derivative for image sharpening – the Laplacian

- Image sharpening using the Laplacian, original image, Laplacian without scaling, Laplacian with scaling (rise, scale and truncate), sharpened image



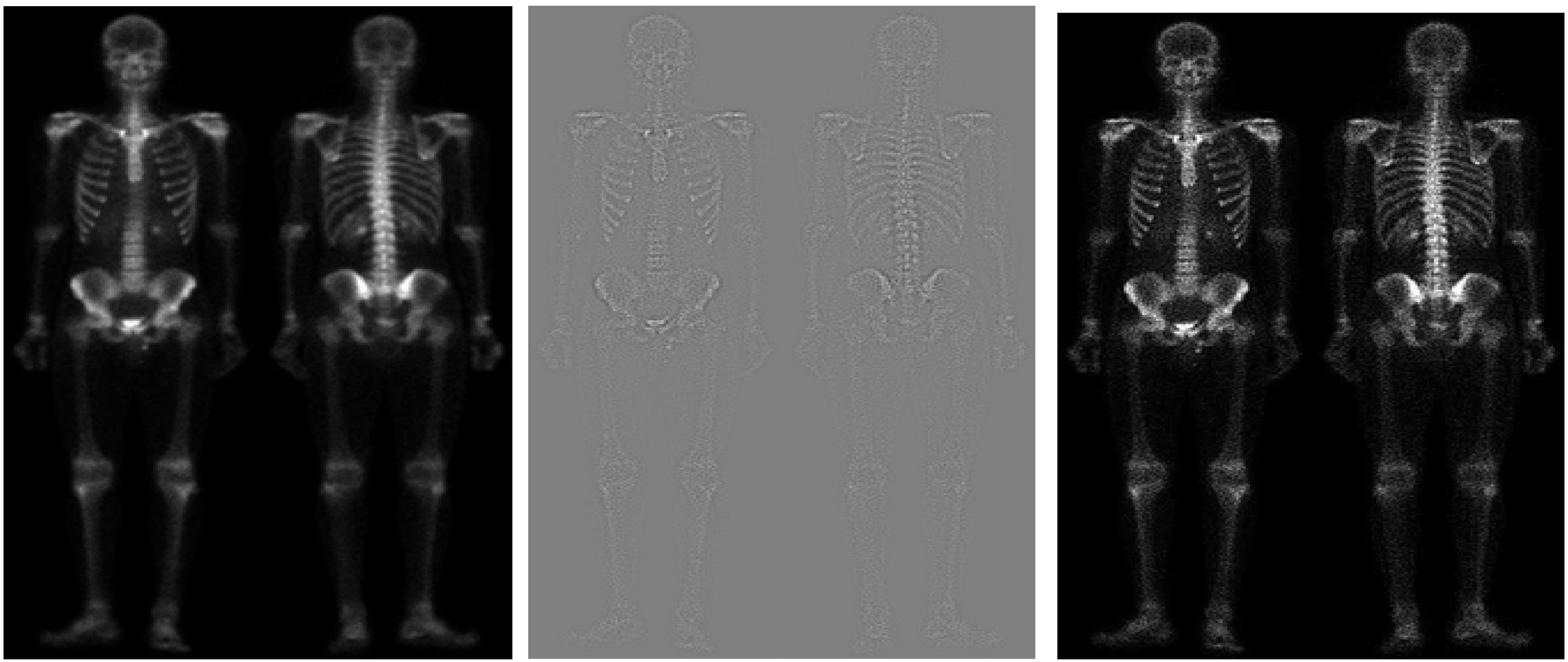
1	1	1
1	−8	1
1	1	1

$$g(x, y) = \begin{cases} f(x, y) - \nabla^2 f(x, y) & \text{if the center coefficient of the Laplacian mask is negative} \\ f(x, y) + \nabla^2 f(x, y) & \text{if the center coefficient of the Laplacian mask is positive.} \end{cases}$$

(Gonzales, Woods)

Using the second-order derivative for image sharpening – the Laplacian

- Image of whole body bone scan, Laplacian of the image, sharpened image



[Gonzales, Woods]

How to avoid negative values of pixels?

- How to display images of which values of pixels are negative or above the value of $2^n - 1$ (n – number of bits, $n = 8$)?

Rise and truncate

1. Add a constant of $2^n / 2$ to the value of each pixel of an image:

$$Value = Value + 2^n / 2$$

2. Truncate the values of pixels of the image:

$$\text{if } (Value < 0) \quad \text{then } Value = 0,$$
$$\text{if } (Value > 2^n - 1) \quad \text{then } Value = 2^n - 1$$

Move and scale

1. Move the values of pixels of an image, i.e, create an image, $f\mathrm{m}$, whose minimum value is 0:

$$f\mathrm{m} = f - min(f)$$

2. Scale the values of pixels of the image $f\mathrm{m}$ to fit between 0 and $2^n - 1$:

$$f\mathrm{s} = K \,.\, [\, f\mathrm{m} / max(f\mathrm{m}) \,], \qquad K = 2^n - 1$$

Rise, scale and truncate

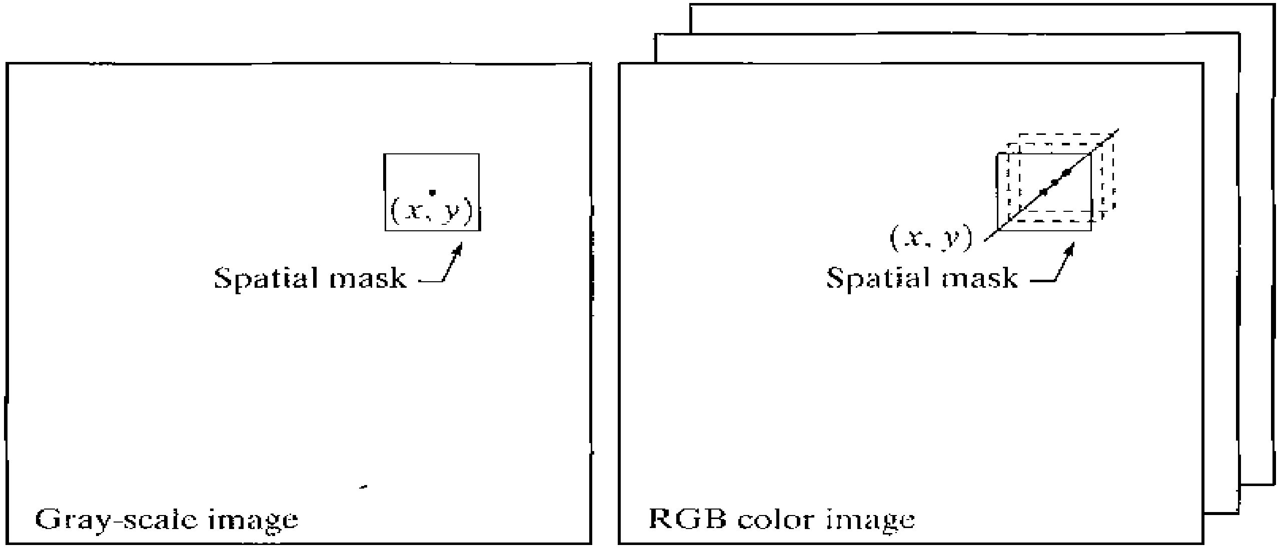
$$f\mathrm{s} = (f + K) / 2, \qquad K = 2^n - 1$$

(Color images)

- Color images

  Three channels; red, green, blue; 3 X 2-D: {*r(x, y), g(x, y), b(x,y)*}

$$\boldsymbol{c}(x,y) = \begin{bmatrix} c_R(x,y) \\ c_G(x,y) \\ c_B(x,y) \end{bmatrix} = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix}$$

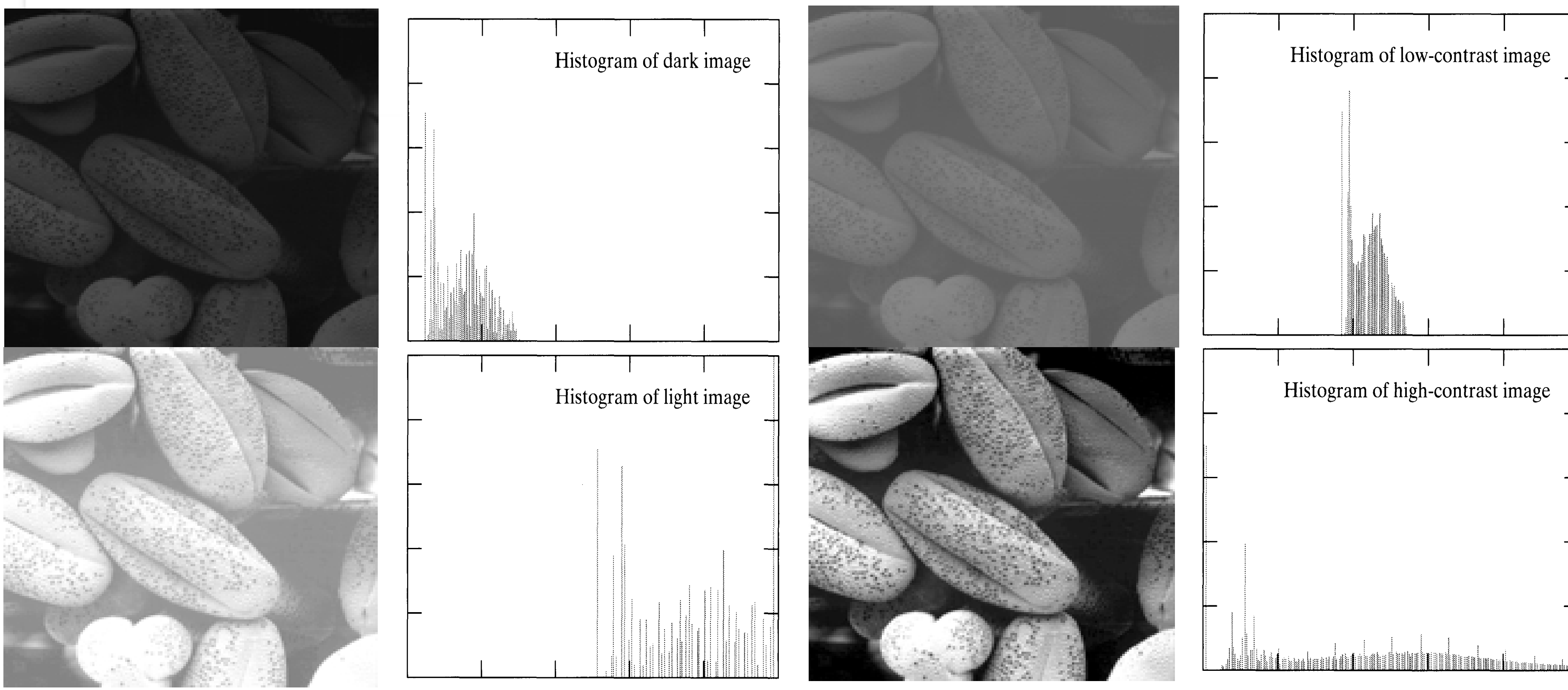


(Gonzales, Woods)

(Intensity transformations)

- A histogram

A plot of $p(r_k)$ versus r_k

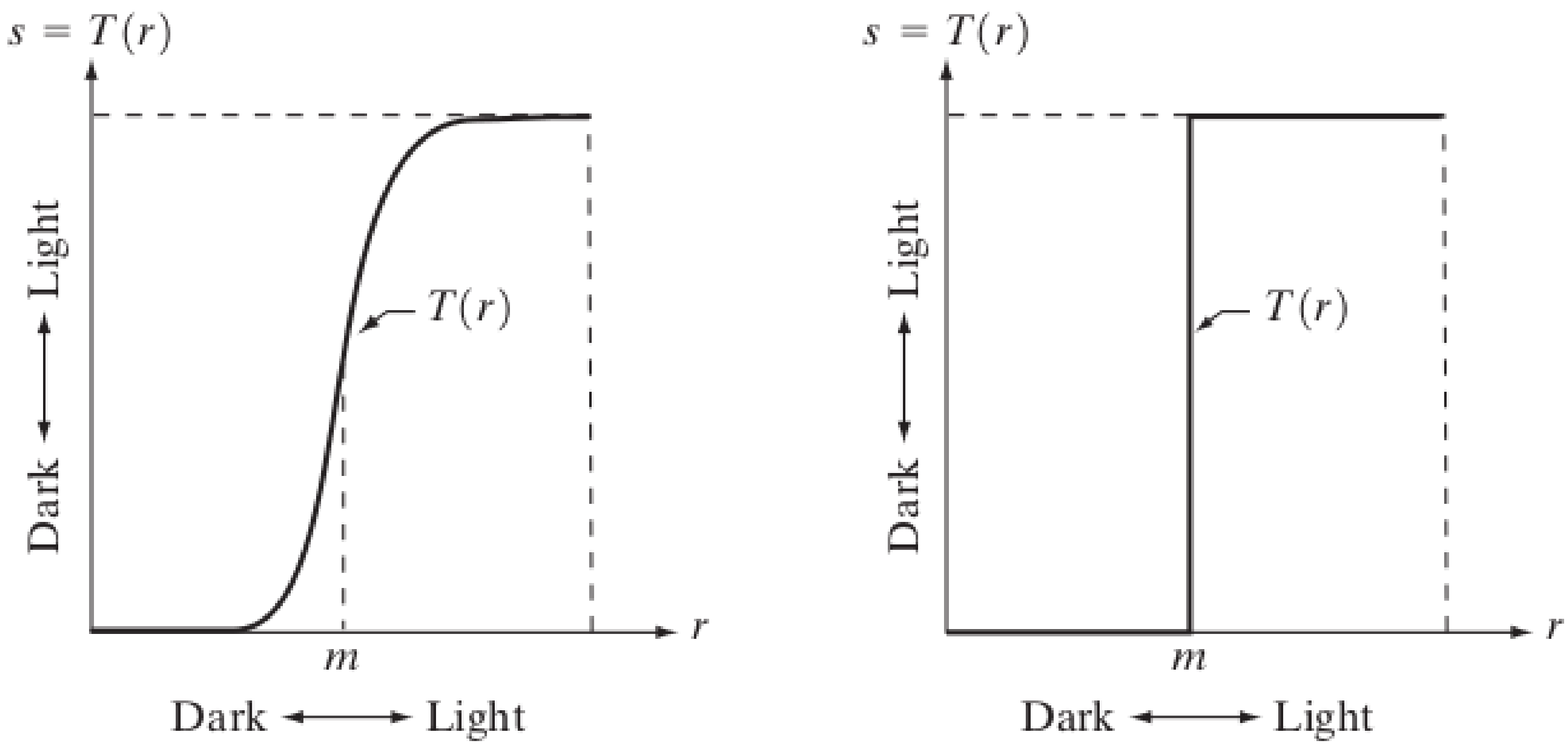


(Gonzales, Woods)

(Intensity transformations)

- Intensity transformation functions

  (a) Contrast stretching function (b) Thresholding function

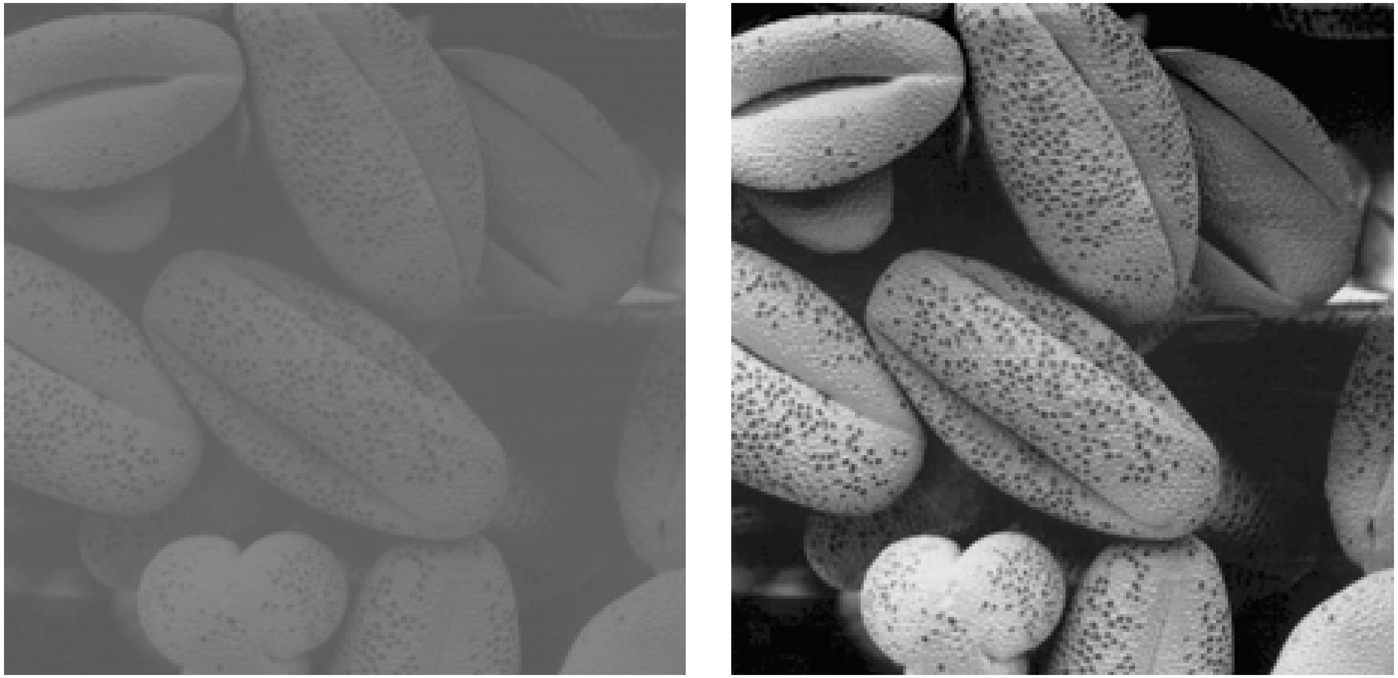


(Gonzales, Woods)

(Intensity transformations)

- Intensity transformation functions

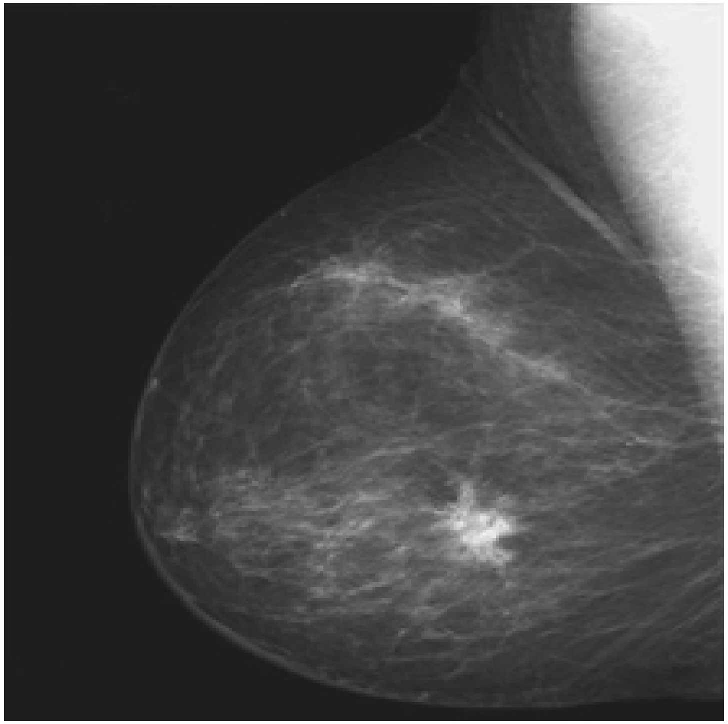
(a) Contrast stretching function

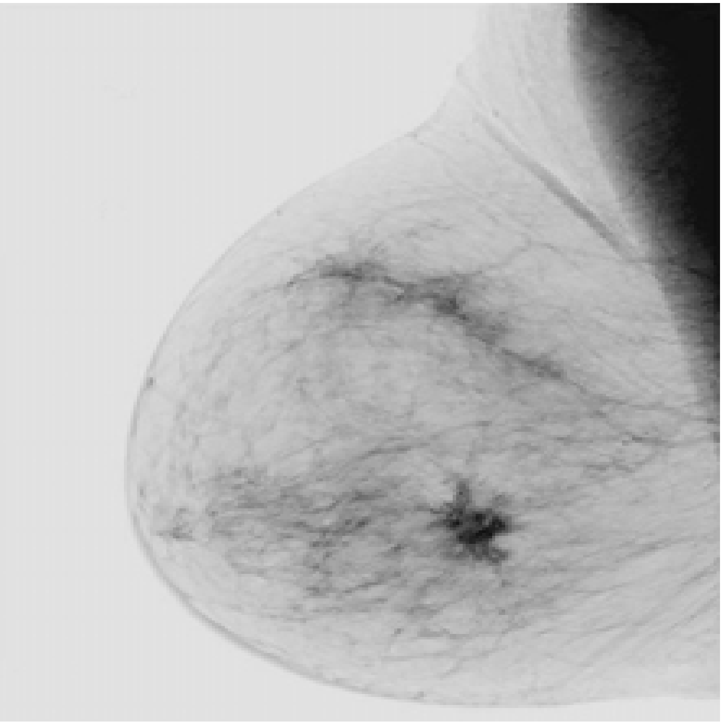


(Gonzales, Woods)

(Intensity transformations)

- (a) Original mammogram
  (b) Negative image obtained using the negative transformation

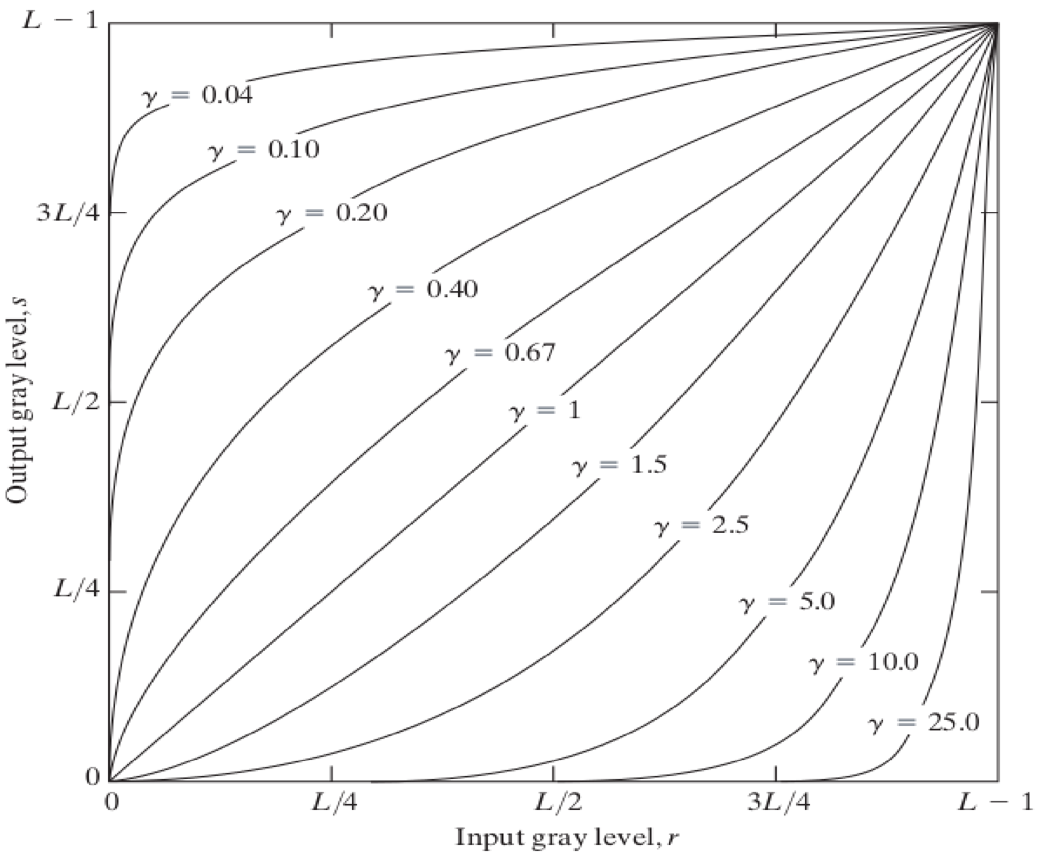




(Gonzales, Woods)

(Intensity transformations)

- Power-law (gamma) transformations $s = cr^{\gamma}$ (*c* = *1* in all cases)



(Gonzales, Woods)

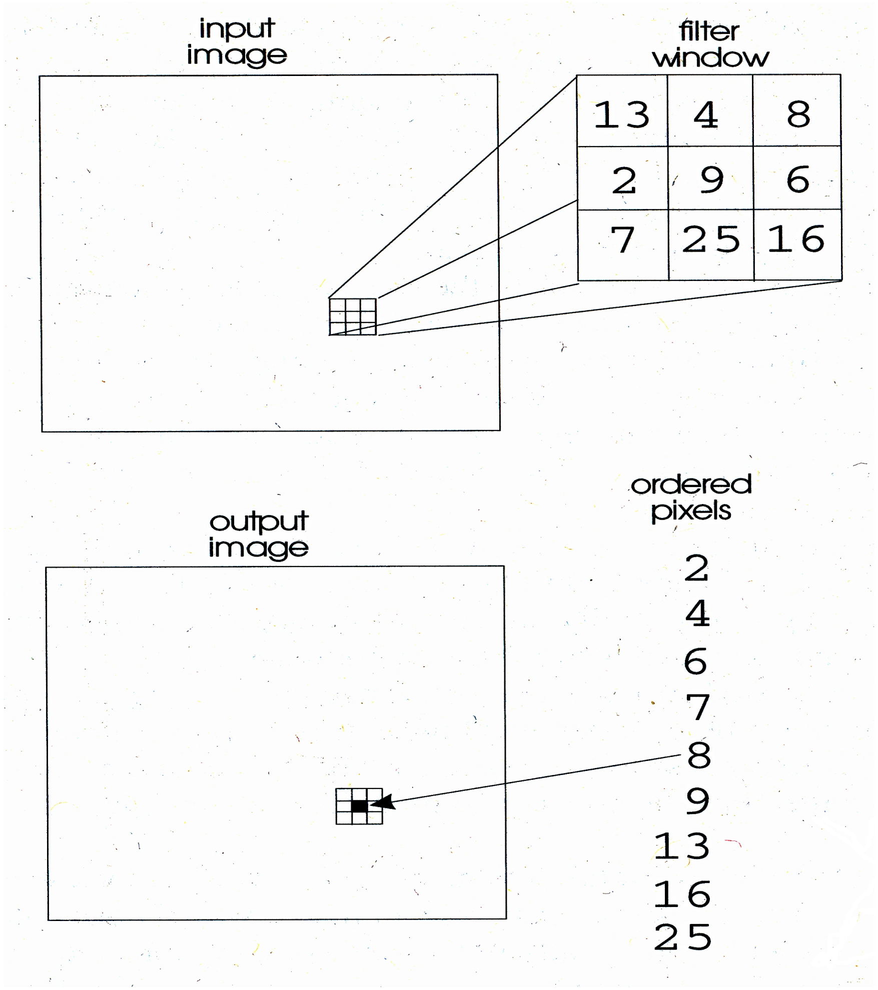
(Intensity transformations)

- (a) Magnetic resonance image of a fractured human spine
  (b - d) After gamma transformation ($\gamma = 0.6, 0.4,$ and 0.3)

(Gonzales, Woods)

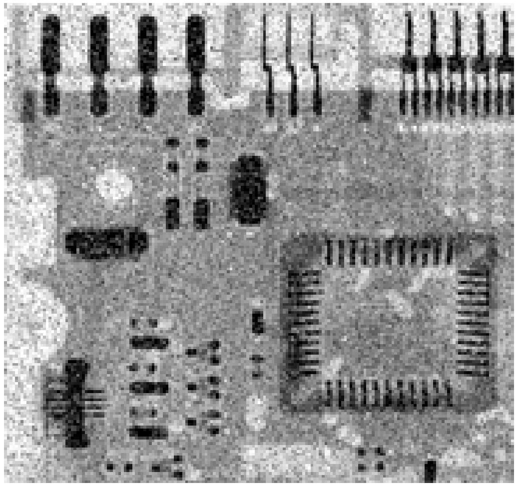
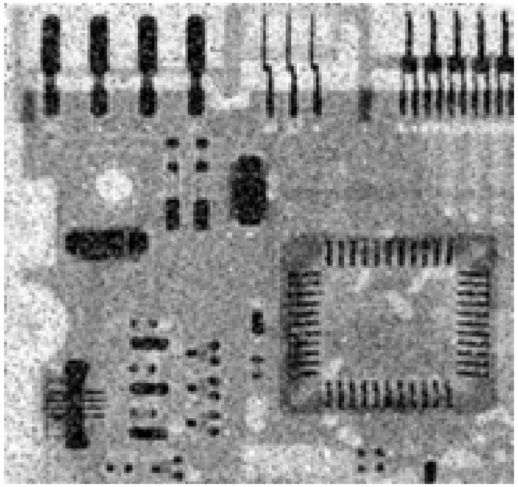
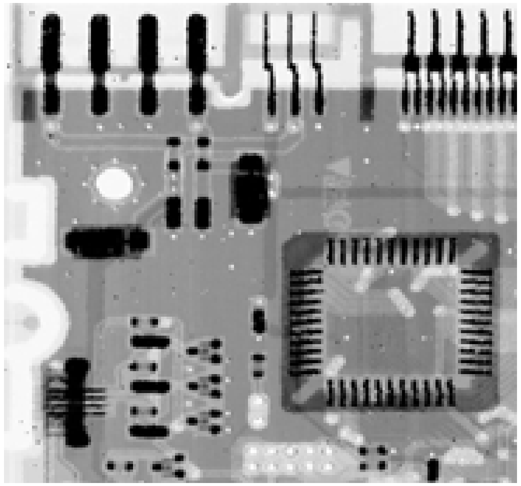
(Median smoothing spatial filters)

- Median filter



(Smoothing spatial filters (examples))

- Results of smoothing with square averaging filter (size of mask, m = 3) and with 3 X 3 median filter

(Gonzales, Woods)

University of Ljubljana
Faculty of Computer and Information Science

(Using the second derivative for image sharpening – joint mask)

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is negative} \\ f(x,y) + \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is positive.} \end{cases}$$

$$\nabla^2 f = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y)$$

$$\begin{aligned} g(x,y) &= f(x,y) - [f(x+1,y) + f(x-1,y) \\ &\quad + f(x,y+1) + f(x,y-1)] + 4f(x,y) \\ &= 5f(x,y) - [f(x+1,y) + f(x-1,y) \\ &\quad + f(x,y+1) + f(x,y-1)] \end{aligned}$$

Laplace

0	1	0
1	–4	1
0	1	0

Joint mask

0	–1	0
–1	5	–1
0	–1	0

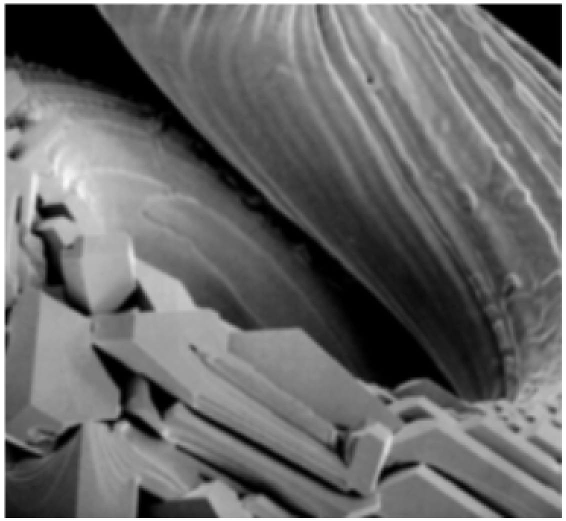
Joint mask with diagonals

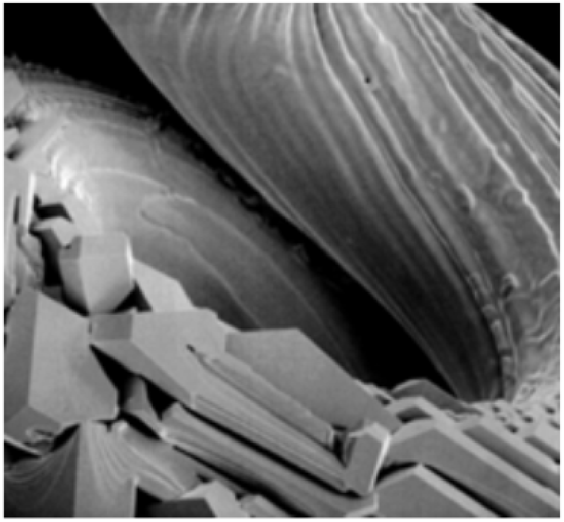
–1	–1	–1
–1	9	–1
–1	–1	–1

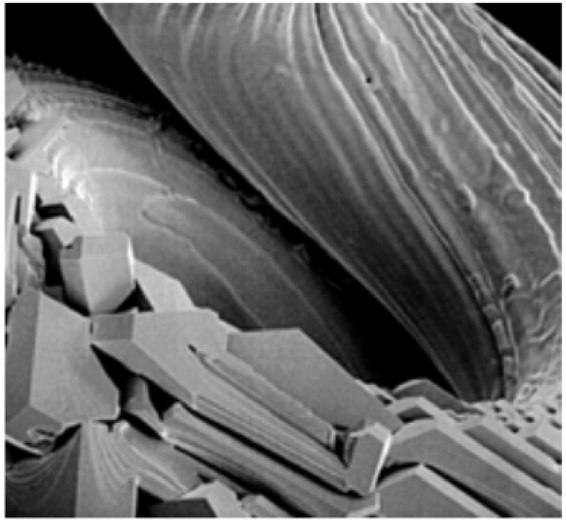
(Using the second-order derivative for image sharpening – joint mask)

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$







[Gonzales, Woods]

(Unsharp masking and high-boost filtering)

1. Blur the original image *f(x,y)*
2. Subtract the blurred image *fb(x,y)* from the original (the *mask*)

   Unsharp masking:

   fs(x,y) = f(x,y) - fb(x,y)
3. Add the mask to the original:

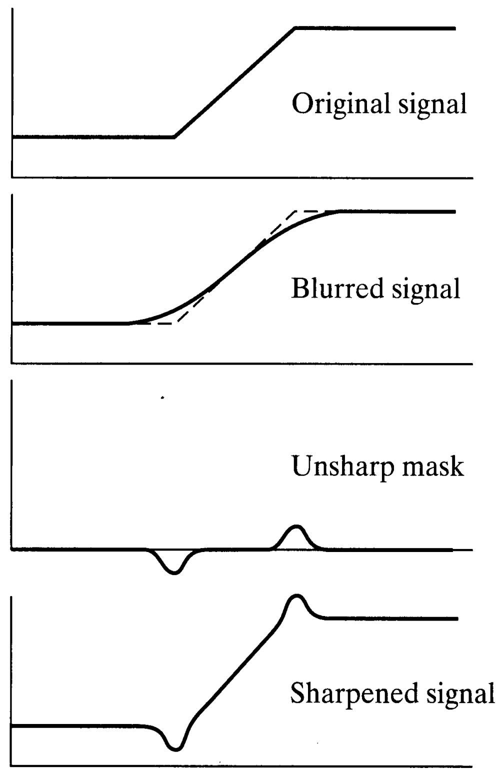
   g(x,y) = f(x,y) + fs(x,y)

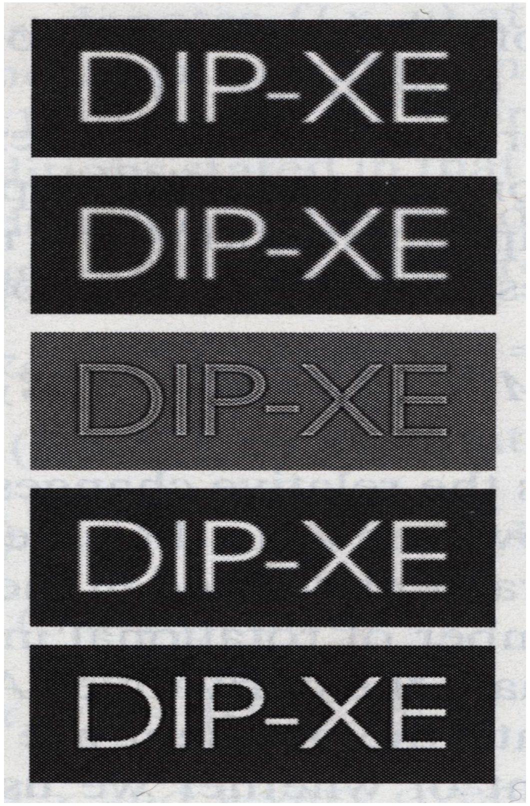
High-boost filtering (A > 1):

fhb(x,y) = A . fs(x,y)

3. Add the mask to the original:

   g(x,y) = f(x,y) + fhb(x,y)





Original image, result of blurring with a Gaussian filter, unsharp mask, result of using unsharp masking, result of using high-boost filtering

(Gonzales, Woods)