COMPUTATIONAL VISION: Digital Image, Linear Operators, Linear Filters

Master in Artificial Intelligence

Department of Mathematics and Computer Science

2019-2020

UNIVERSITAT DE BARCELONA

Outline

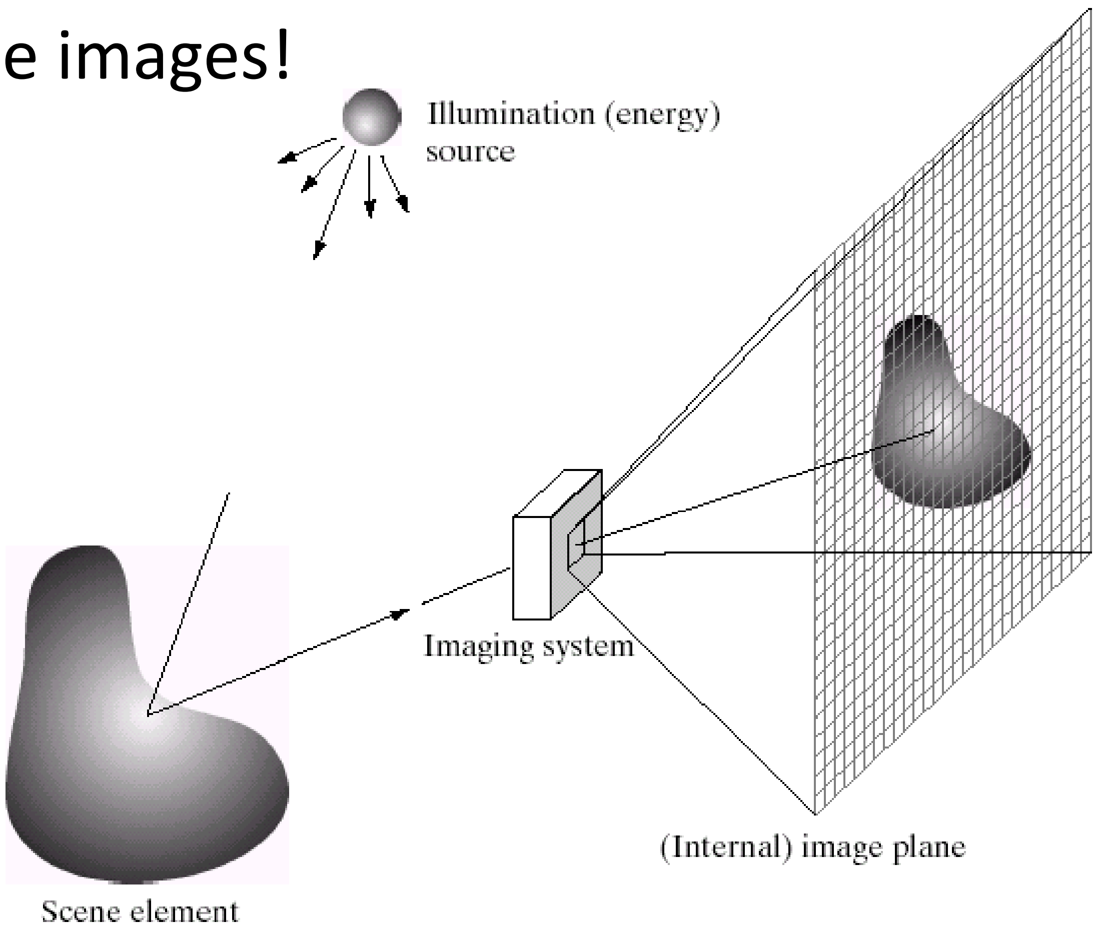
- **Digital images**

- Spatial and photometric resolution
  - Histogram
  - Image contrast enhancement

- Linear filters
  - Examples: smoothing filters
  - Convolution
  - Gaussian filter

Image Formation

Light is an energy source that carries coded information about the world, which can be read from a distance through the images!



Slide credit: Derek Hoiem

Digital camera

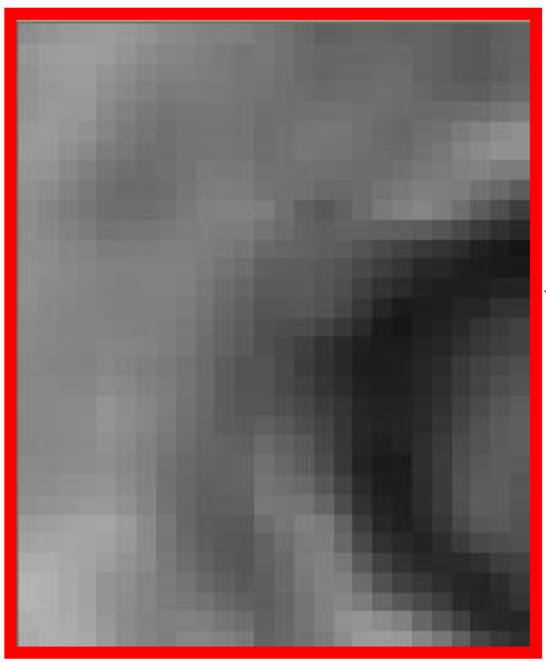
The image I(x,y) measures how much light is captured at pixel (x,y): INTENSITY or BRIGHTNESS.

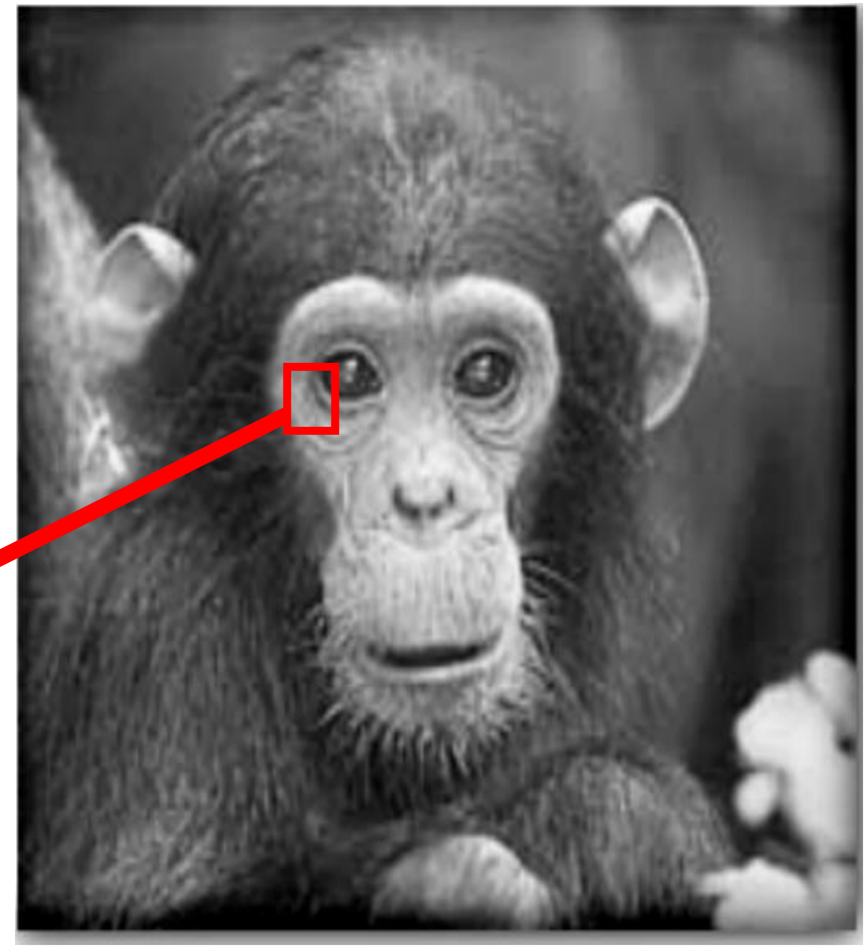
- Proportional to the number of photons captured at the sensor element (CCD, CMOS, ..) in a time interval.

http://electronics.howstuffworks.com/cameras-photography/digital/digital-camera.htm

Digital images

Think of images as matrices taken from a sensor array.



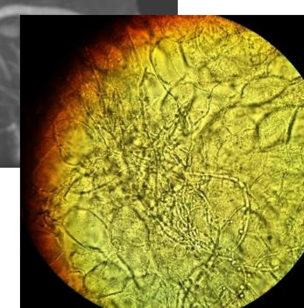


K. Grauman

Digital images

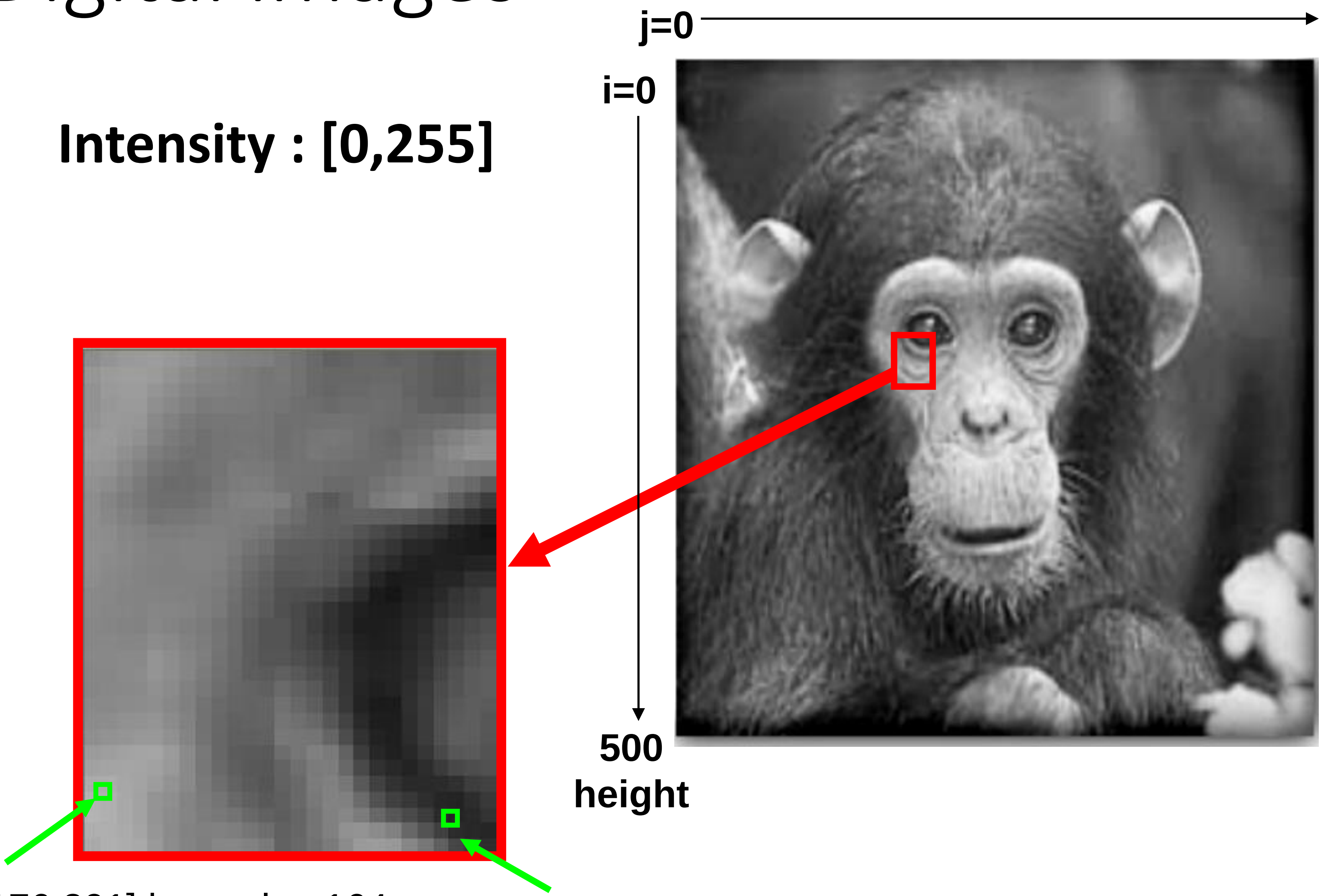
Different kinds of images:

- Natural images
- Other modalities:
  - X-rays, MRI…
  - Light Microscopy, Electron Microscopy…
  - …



Digital images

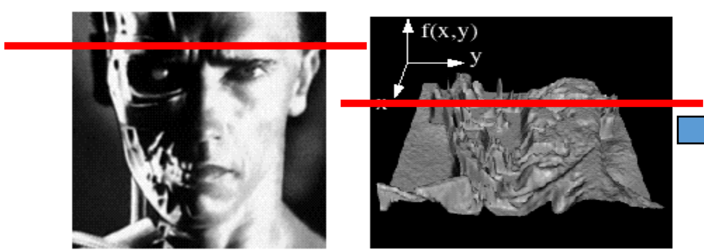
Intensity : [0,255]



K. Grauman

Digital images

- **Sample** the 2D space on a regular grid
- **Quantize** each sample (round to nearest integer)
- Image is represented as:
  - a matrix of integer values or
  - a surface map.

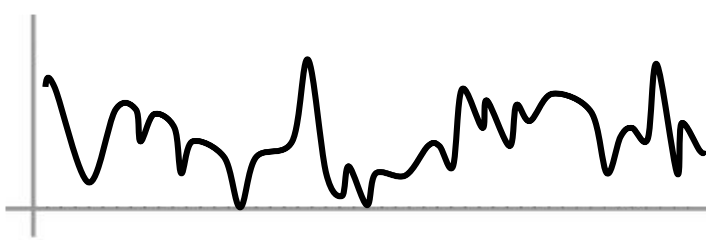


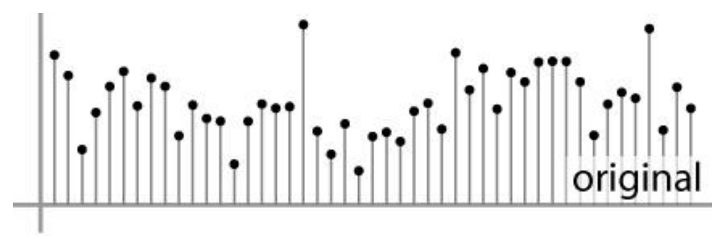
y

x

62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

2D

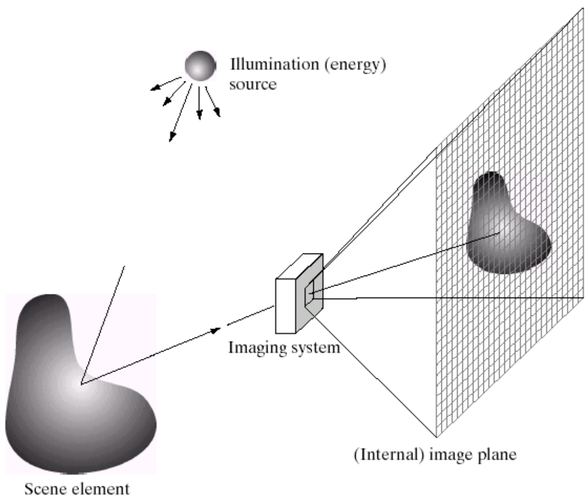


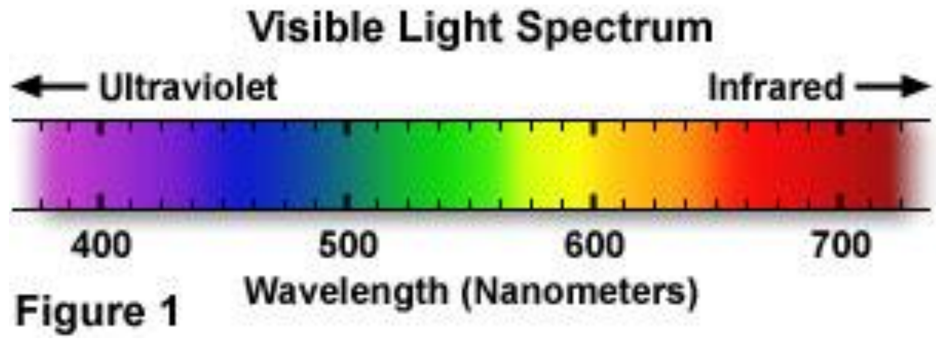


1D

Adapted from S. Seitz

How do we obtain color images?



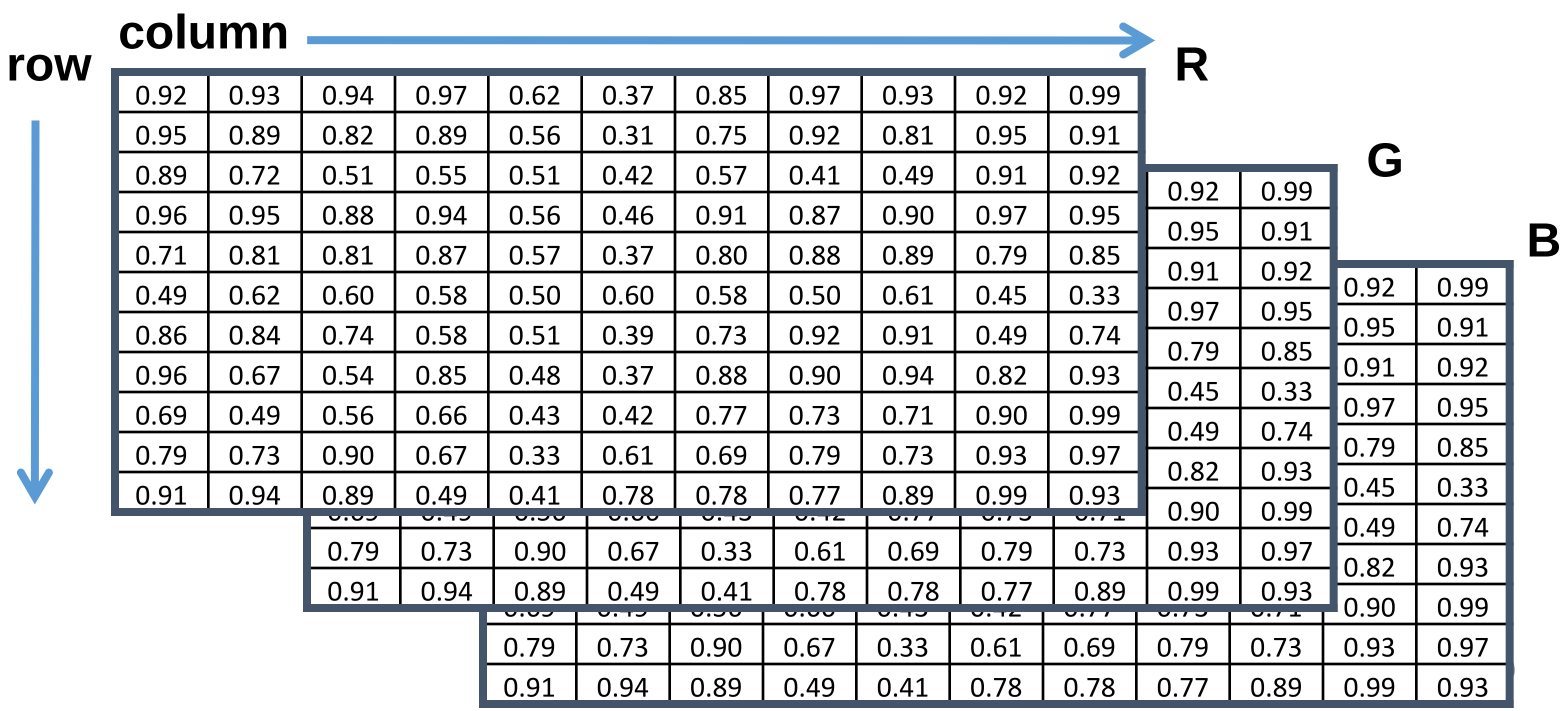


A typical human eye will respond to wavelengths from about **380 to 750 nm.**

In order to get a full color image, most sensors use filtering to look at the light in its three primary colors (Red, Green, Blue). Once the camera records all three colors, it combines them to create the full spectrum.

Color Images in Matlab

- Image represented as a matrix
- Suppose we have a NxM RGB image called “im”, then:
  - `im[0,0,0]` = top-left pixel value in Red channel
  - `im[y, x, b]` = y pixels down, x pixels to right in the b^{th} channel
  - `im[N, M, 3]` = bottom-right pixel in Blue channel



0.92	0.93	0.94	0.97	0.62	0.37	0.85	0.97	0.93	0.92	0.99
0.95	0.89	0.82	0.89	0.56	0.31	0.75	0.92	0.81	0.95	0.91
0.89	0.72	0.51	0.55	0.51	0.42	0.57	0.41	0.49	0.91	0.92
0.96	0.95	0.88	0.94	0.56	0.46	0.91	0.87	0.90	0.97	0.95
0.71	0.81	0.81	0.87	0.57	0.37	0.80	0.88	0.89	0.79	0.85
0.49	0.62	0.60	0.58	0.50	0.60	0.58	0.50	0.61	0.45	0.33
0.86	0.84	0.74	0.58	0.51	0.39	0.73	0.92	0.91	0.49	0.74
0.96	0.67	0.54	0.85	0.48	0.37	0.88	0.90	0.94	0.82	0.93
0.69	0.49	0.56	0.66	0.43	0.42	0.77	0.73	0.71	0.90	0.99
0.79	0.73	0.90	0.67	0.33	0.61	0.69	0.79	0.73	0.93	0.97
0.91	0.94	0.89	0.49	0.41	0.78	0.78	0.77	0.89	0.99	0.93

Color images, RGB color space

R

G

B

Why are the chairs tops appearing in black in two of the channels?
What are the values of a white pixel in the color image?

Outline

- Digital images
- **Spatial and photometric resolution**
  - Histogram
  - Image contrast enhancement
- Linear filters
  - Examples: smoothing filters
  - Convolution
  - Gaussian filter

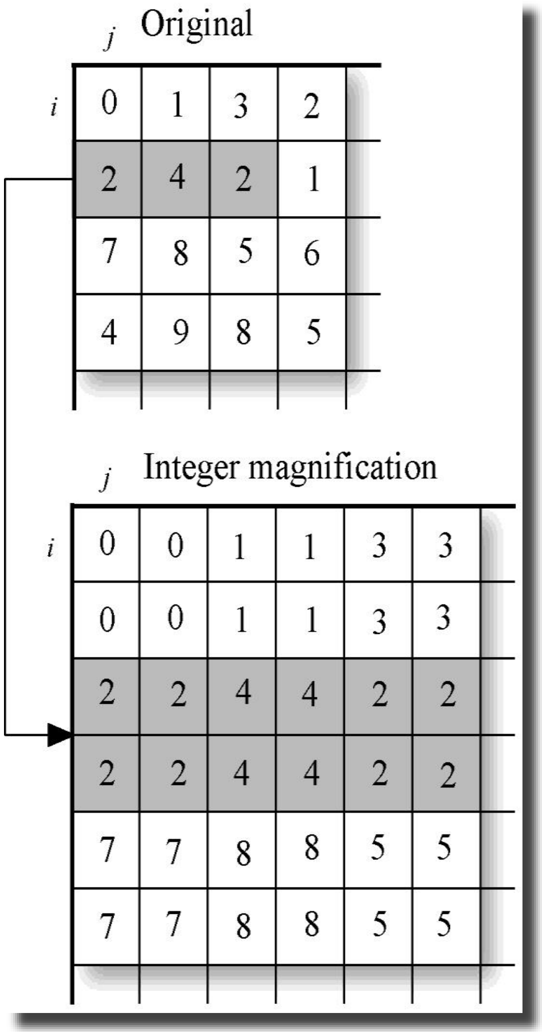
Resolution

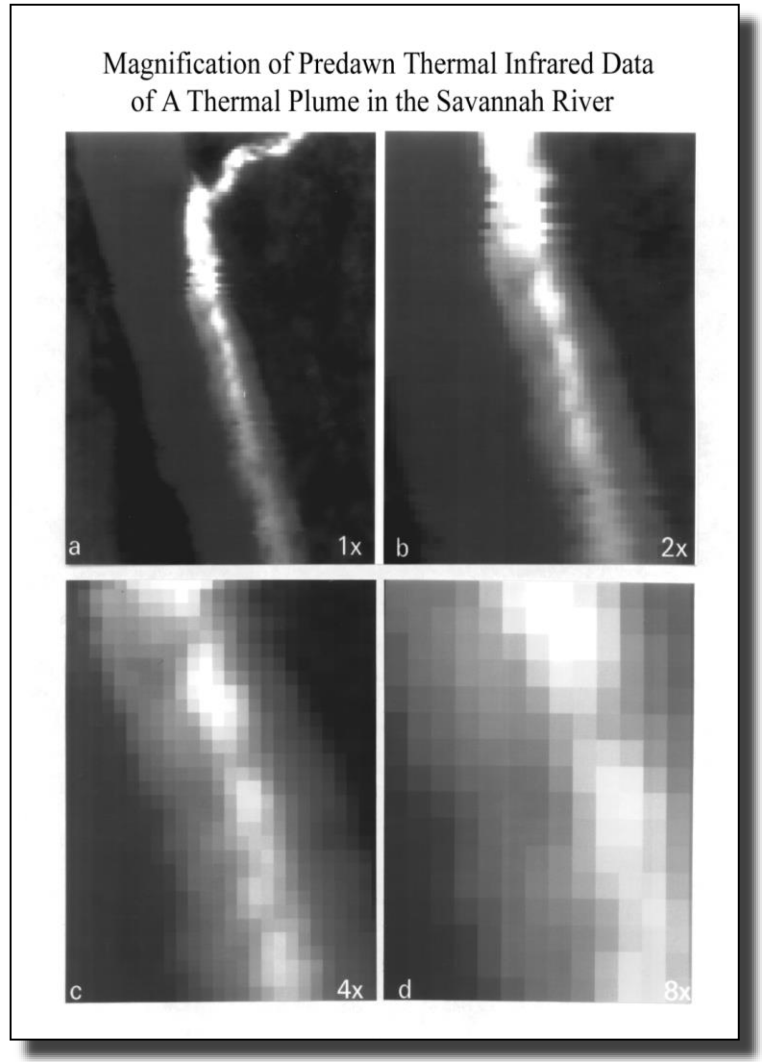
- **Sensor resolution**: size of real world scene element that images to a single pixel
- **Image resolution**: number of pixels

[fig from Mori et al]

Influences what analysis is feasible, affects best representation choice.

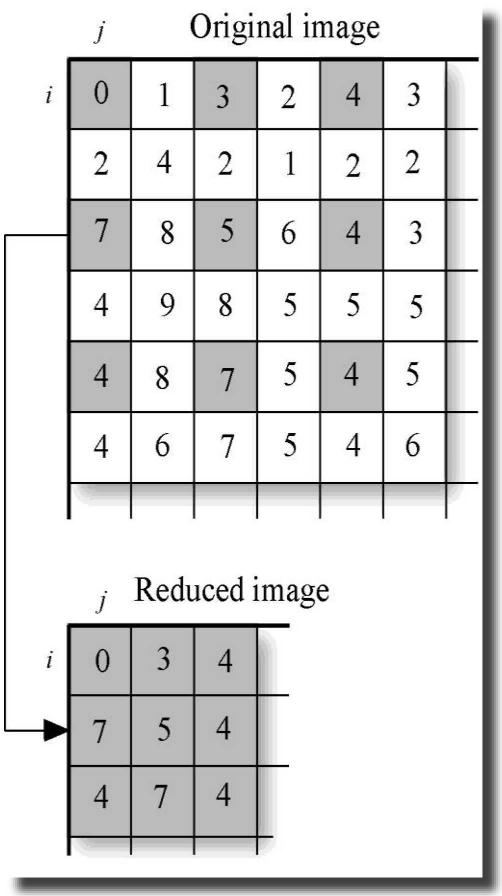
Image Magnification

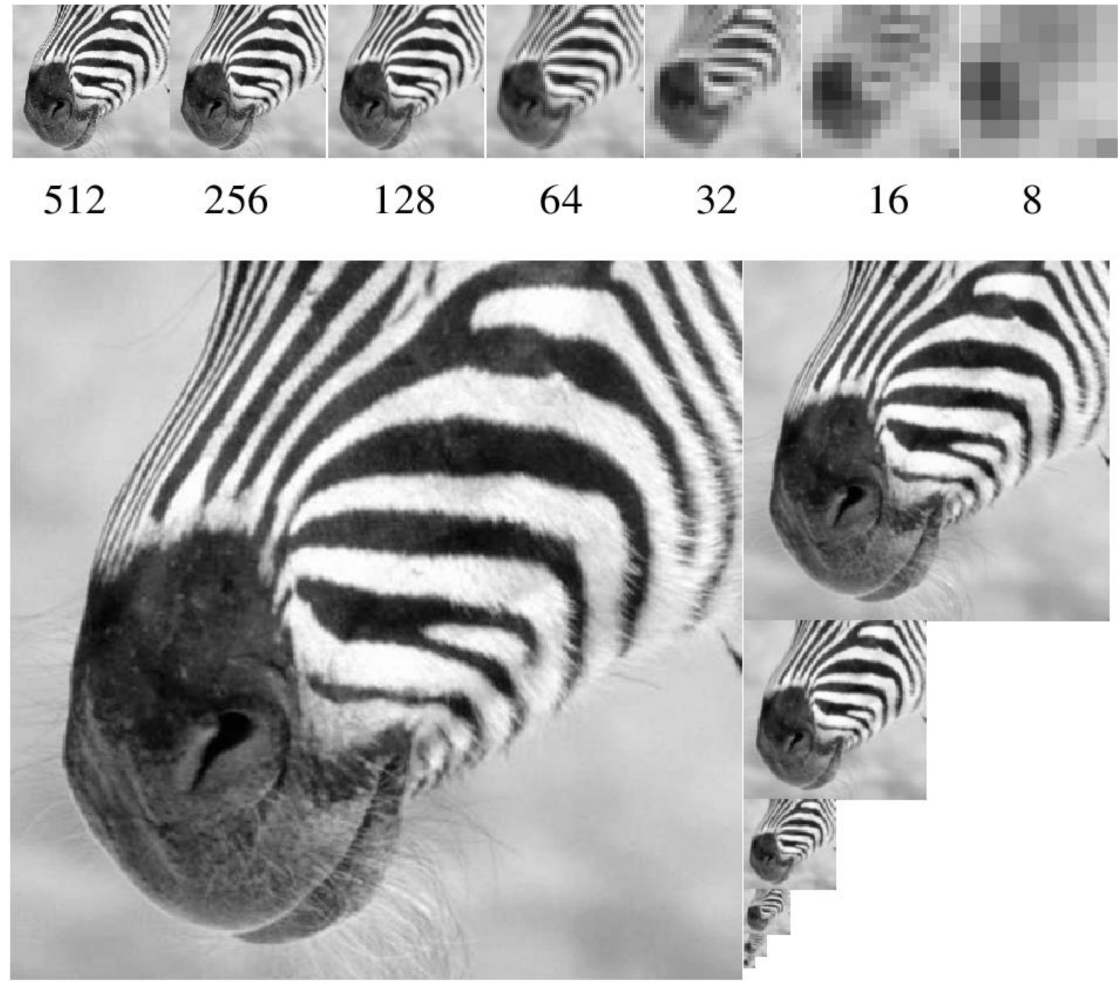




rescale()

Image Reduction

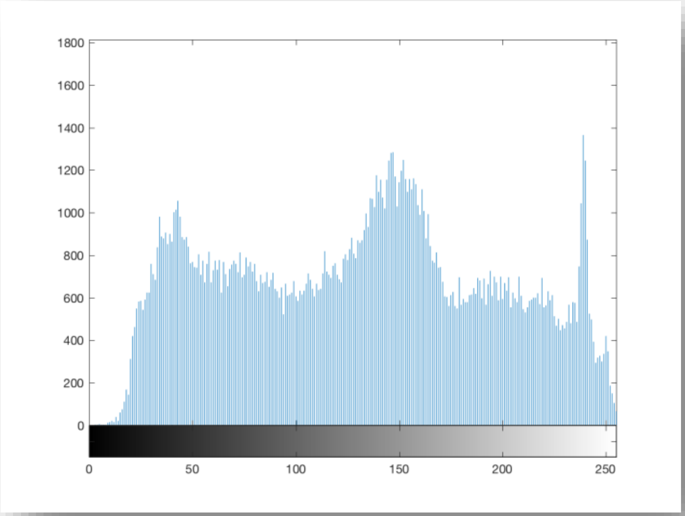





```
rescale(im,0.5)
```

Photometric resolution of the image

The number of different grey levels (different pixel values in each color channel).



A **histogram** of an image represents the frequencies of the image gray levels.

- Does it depend on the spatial distribution?
- Can it be considered as a measure of image quality?

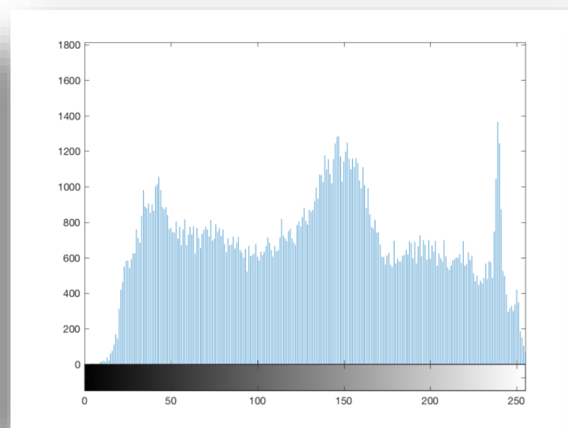
Histogram

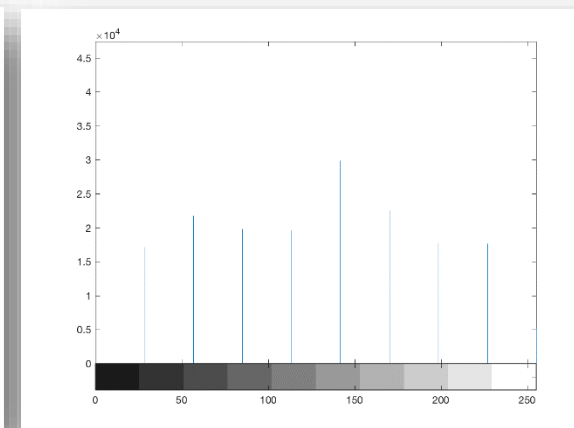
```
mm=np.zeros((256, 256,3) dtype=np.uint8);
```

→ Creates an image of what color?

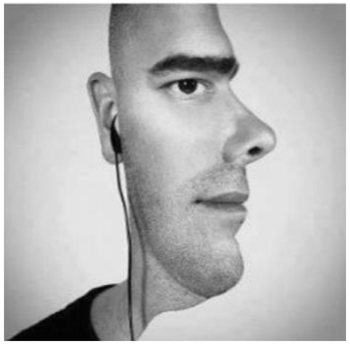
- Given an image of type uint8, how many grey levels we can have at most?

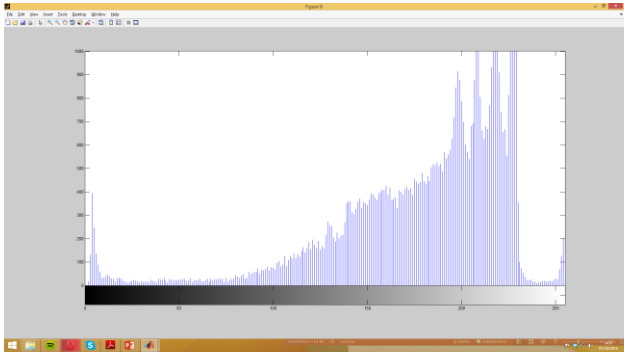
- We can change the number of bins of the histogram:

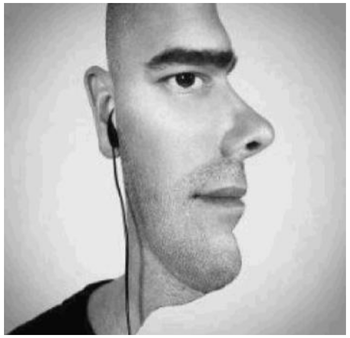


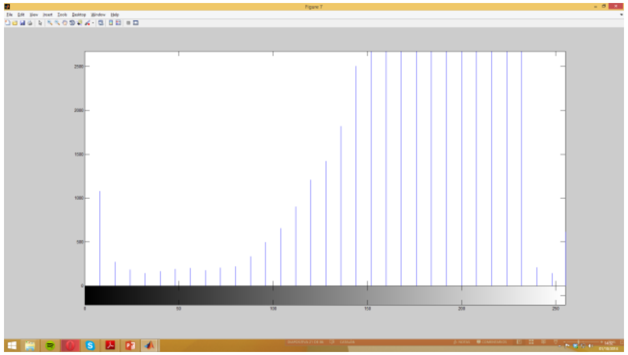


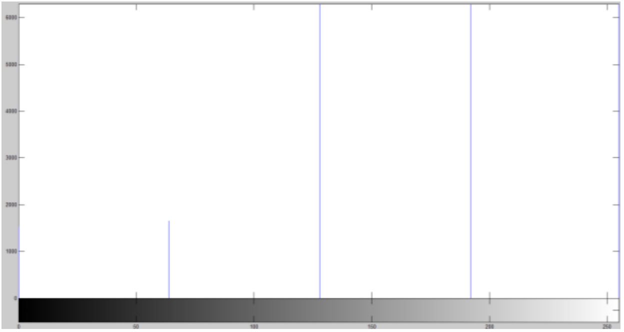
Histogram: How should the histograms look?







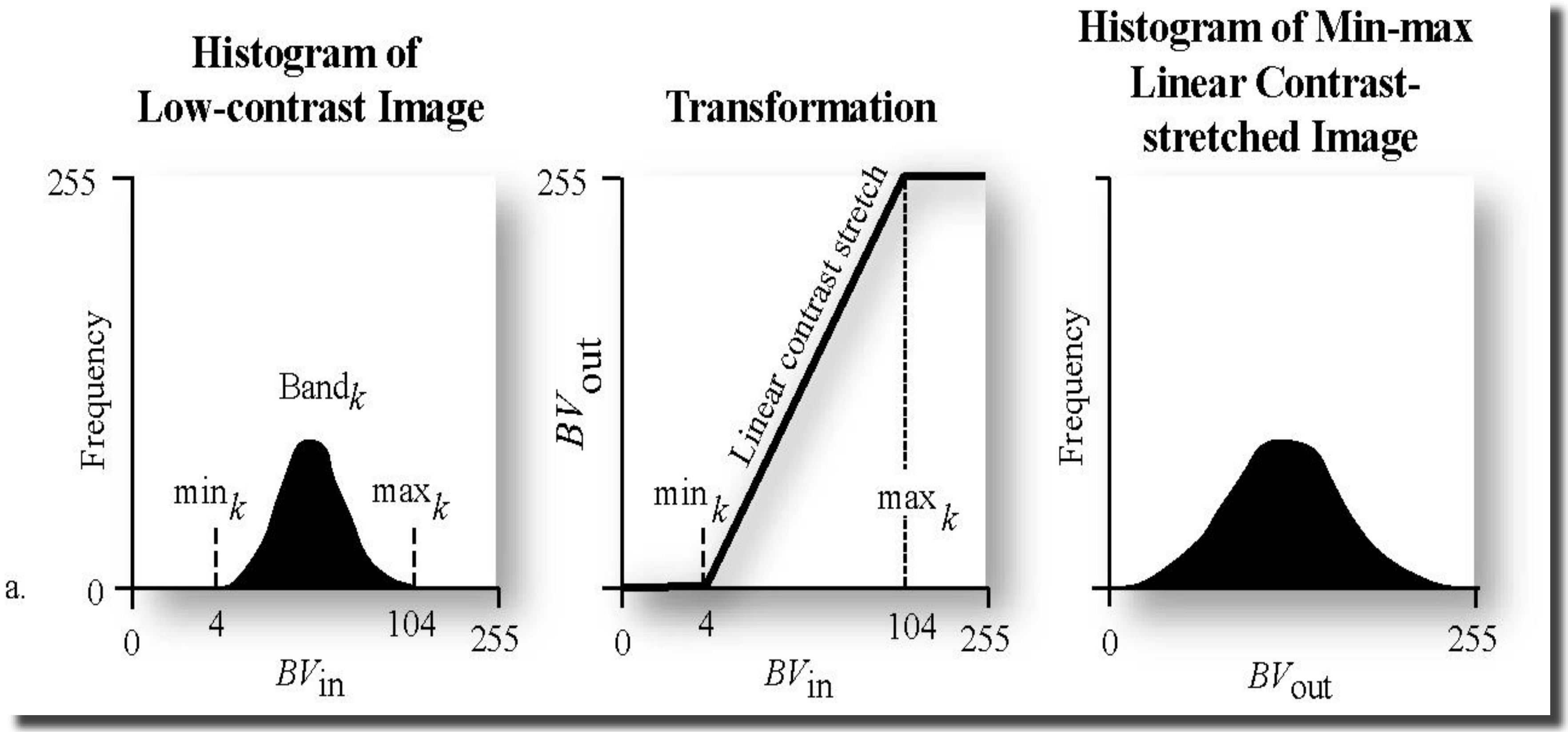




Histogram manipulation for image contrast enhancement

- Minimum-maximum contrast stretch
- Percentage linear contrast stretch

Histogram manipulation

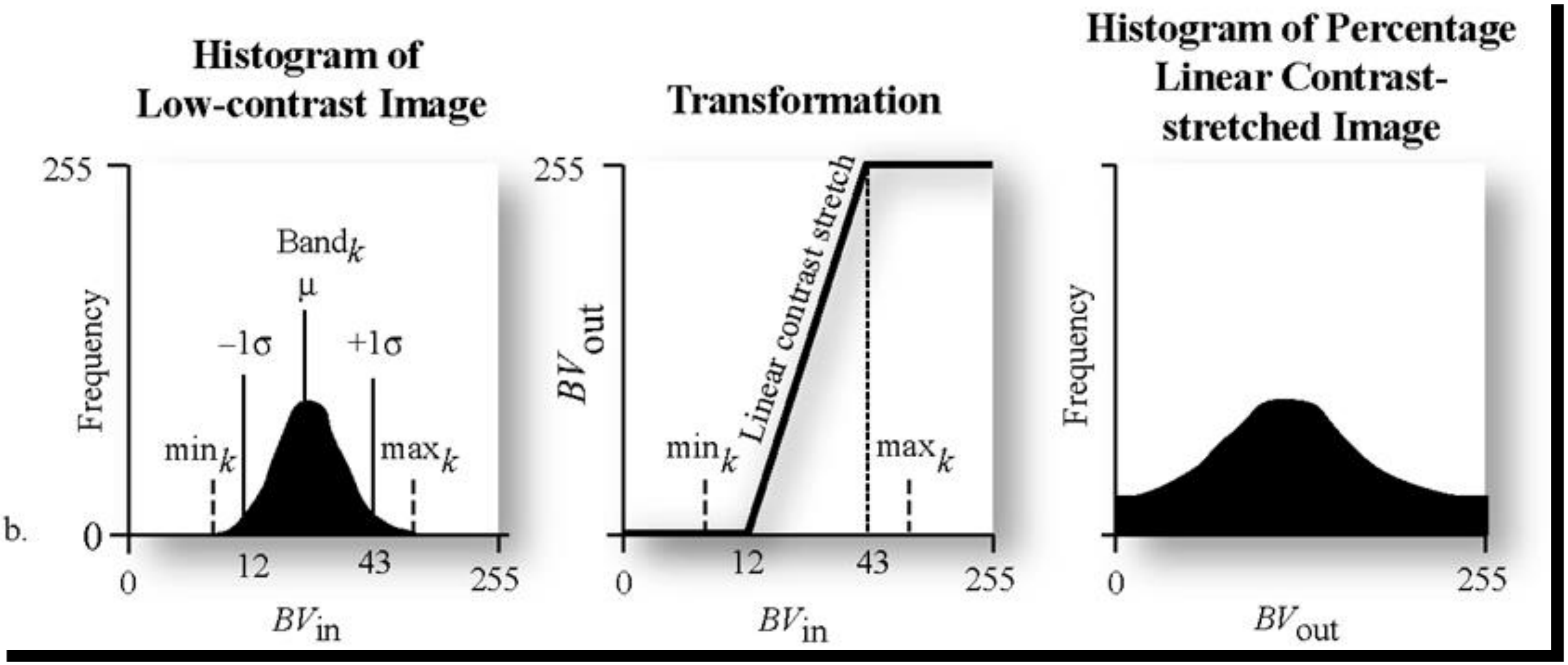


$$BV_{out} = \left(\frac{BV_{in} - \min_k}{\max_k - \min_k} \right) quant_k$$

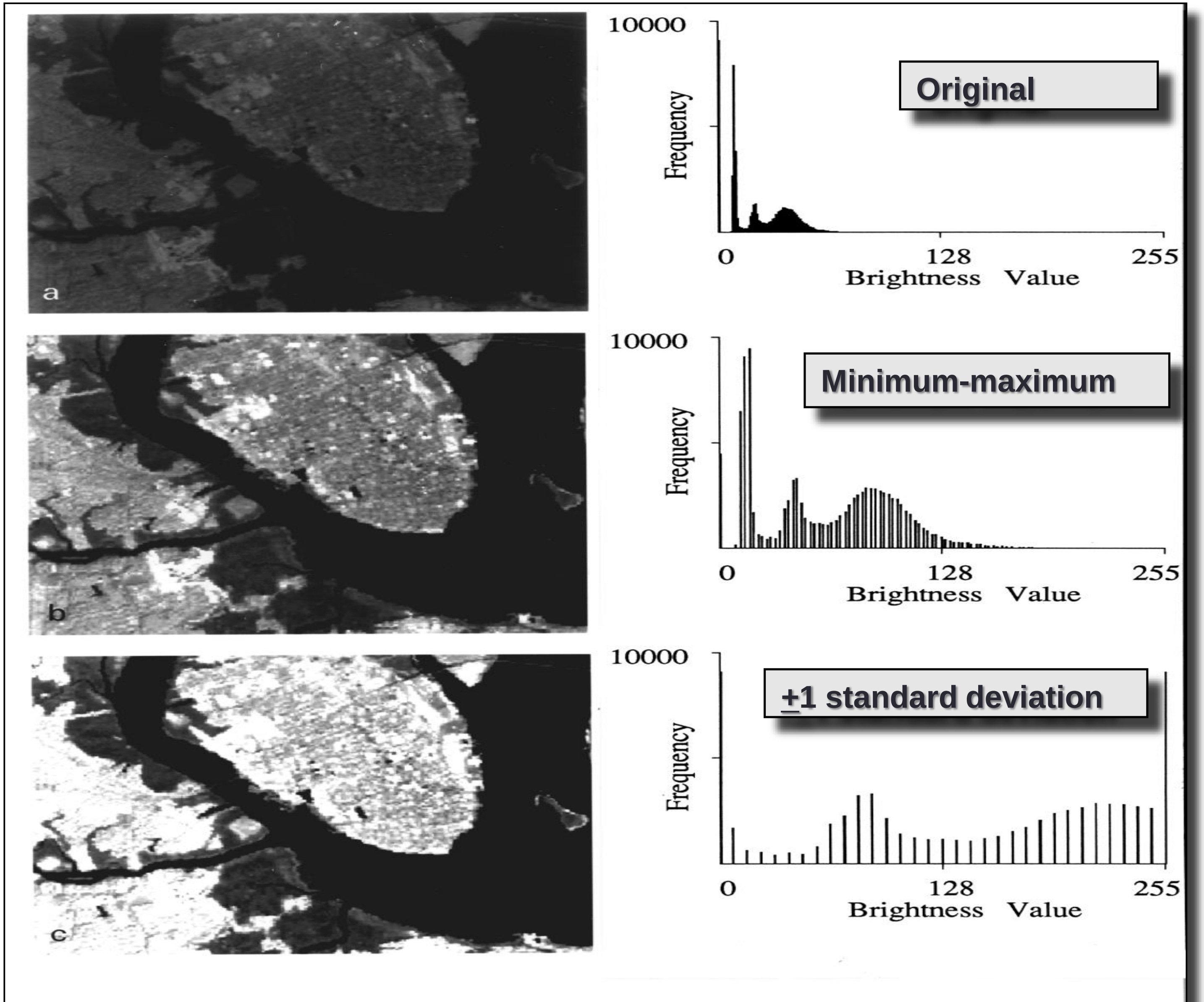
- BV_{in} is the original input brightness value
- $quant_k$ is the range of the brightness values that can be displayed (e.g. 255),
- min_k is the minimum value in the image,
- max_k is the maximum value in the image, and
- BV_{out} is the output brightness value.

Histogram manipulation

Percentage linear and standard deviation contrast stretch



Histogram manipulation for image contrast enhancement



Outline

- Digital images
- Spatial and photometric resolution
  - Histogram
  - Image contrast enhancement
- **Linear filters**
  - Examples: smoothing filters
  - Convolution
  - Gaussian filter

Image filtering

- **Linear filtering**: Computing a function of the local neighborhood at each pixel in the image
  - Function specified by a “filter” or mask saying **how to combine** values from neighbors.

- We can see different uses of filtering:
  - Soft blur, smoothing
  - Enhance an image (remove noise)
  - Extract information: Sharpen or accentuate details (texture, edges)
  - Detect patterns (feature detection and matching)

Adapted from Derek Hoiem

Common types of noise

- **Salt and pepper noise**: random occurrences of black and white pixels
- **Impulse noise:** random occurrences of white pixels
- **Gaussian noise**: variations in intensity drawn from a Gaussian normal distribution

Original

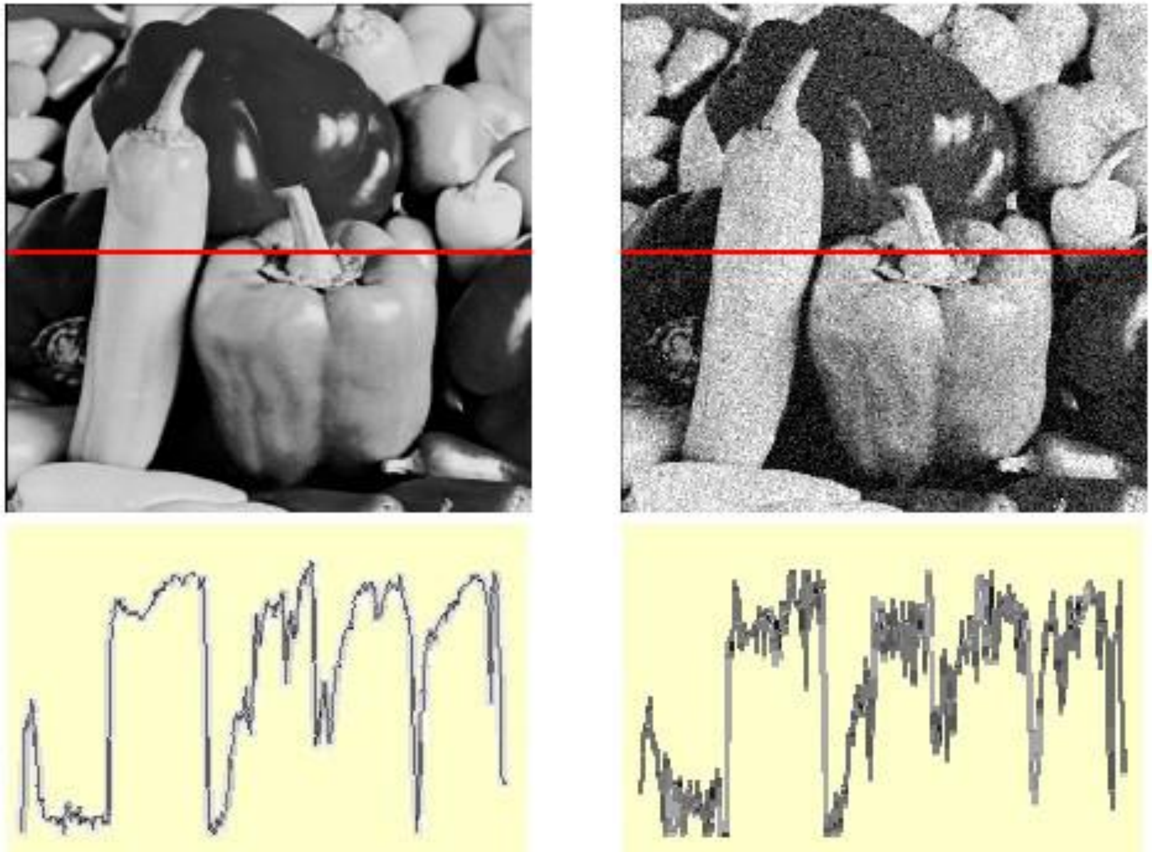
Salt and pepper noise

Impulse noise

Gaussian noise

Source: S. Seitz

Gaussian noise



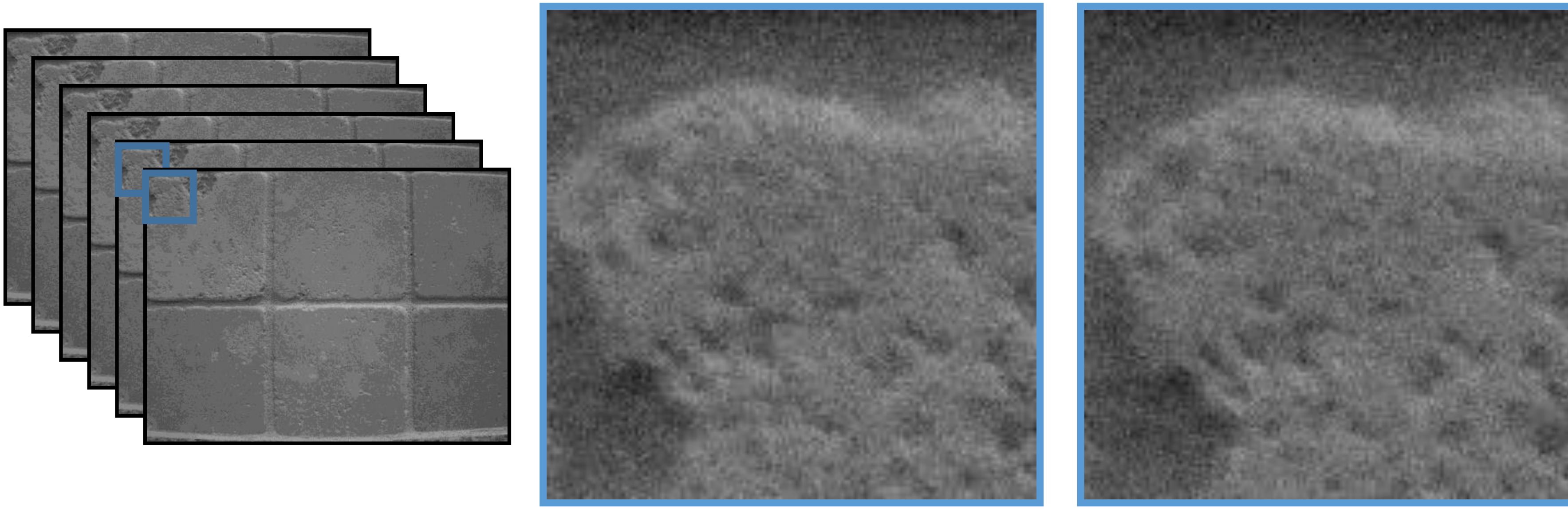
$$f(x,y) = \overbrace{\bar{f}(x,y)}^{\text{Ideal Image}} + \overbrace{\eta(x,y)}^{\text{Noise process}}$$

Gaussian i.i.d. ("white") noise:
$\eta(x,y) \sim \mathcal{N}(\mu, \sigma)$

What is the impact of sigma?

Fig: M. Hebert

Noise reduction



- Even multiple images of the same static scene will not be identical.
- How could we reduce the noise, i.e., give an estimate of the true intensities?

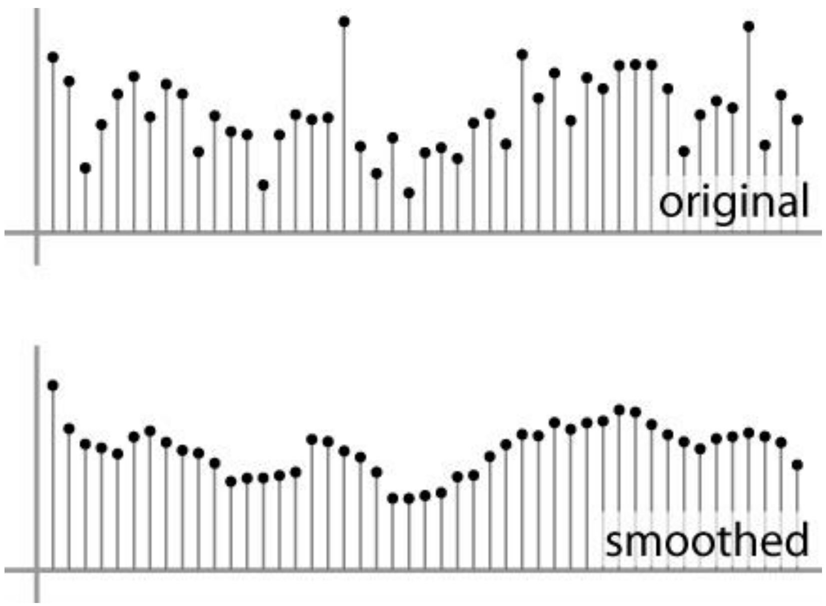
First attempt for a solution

- Let’s replace each pixel with an average of all the values in its neighborhood

- Assumptions:
  - Expect pixels to be like their neighbors
  - Expect noise processes to be independent from pixel to pixel

First attempt for a solution

- Let's replace each pixel with an average of all the values in its neighborhood
- Moving average in 1D:

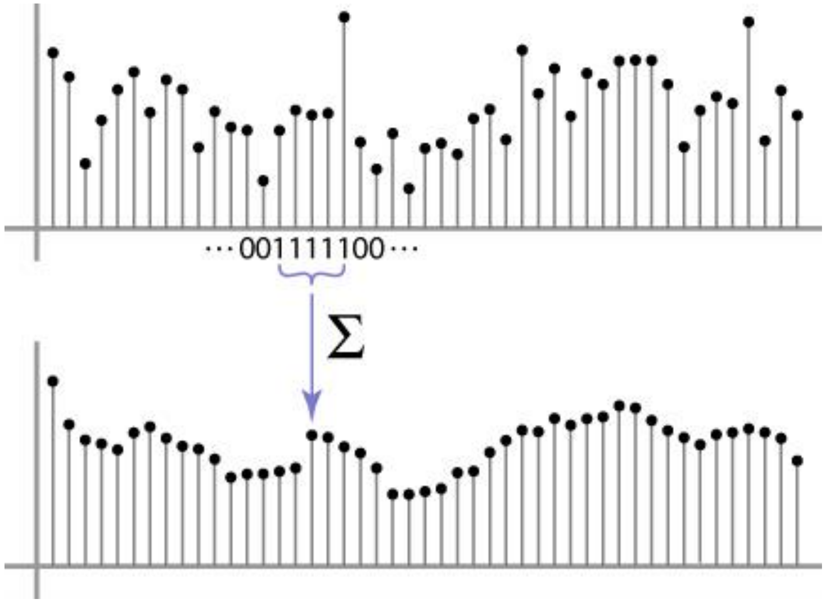


Source: S. Marschner

Moving Average

- We can add weights to our moving average
- *Weights* [1, 1, 1, 1, 1] / 5

Note that we have to divide by 5 to normalize the mask!

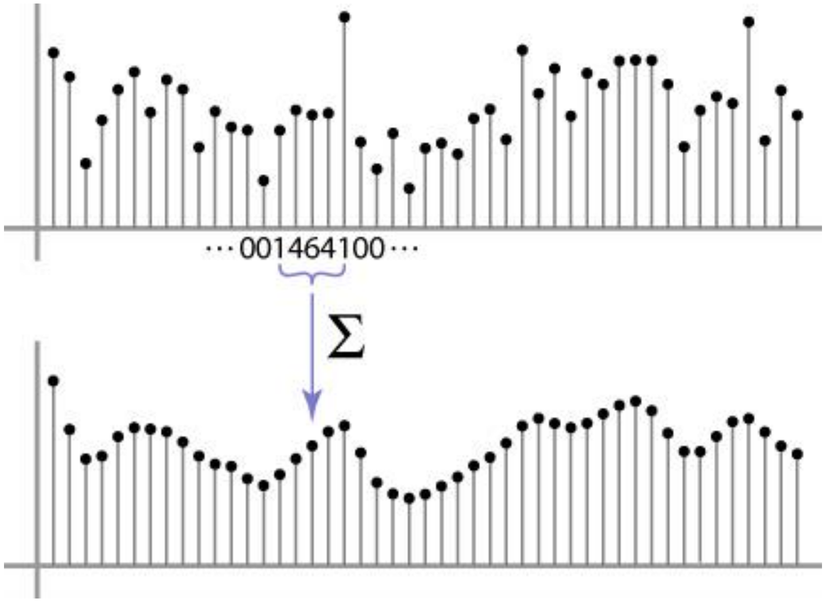


Source: S. Marschner

Weighted Moving Average

- Non-uniform weights [1, 4, 6, 4, 1] / 16

- What is the difference with the previous one?



Source: S. Marschner

Moving Average in 2D

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Image with noise

$G[x, y]$

	0								

Source: S. Seitz

Moving Average In 2D

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10							

Source: S. Seitz

Moving Average In 2D

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20						

Source: S. Seitz

Moving Average In 2D

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20	30					

Source: S. Seitz

Moving Average In 2D

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20	30	30				

Source: S. Seitz

Moving Average In 2D

$$F[x, y]$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

noise

Image with noise

$$G[x, y]$$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Filtered Image

Source: S. Seitz

Correlation filtering

Moving average in 2D with an **averaging** window of size *2k+1 x 2k+1*:

$$G[i,j] = \underbrace{\frac{1}{(2k+1)^2}}_{\textit{Attribute uniform weight to each pixel}} \underbrace{\sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]}_{\textit{Loop over all pixels in neighborhood around image pixel F[i,j]}}$$

Weighted moving average in 2D:

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} \underbrace{H[u,v]}_{\textit{Non-uniform weights}} F[i+u, j+v]$$

Correlation filtering

Filtering an image: replace each pixel with a linear combination of its neighbors.

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v] F[i+u, j+v]$$

We denote the **operation** as: $G = H \otimes F$

The filter "**kernel**" or "**mask**" H[u,v] is the prescription for the weights in the linear combination.

Averaging filter

- What values do belong in the kernel *H* for the moving average example?

$F[x,y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$\otimes$

$H[u,v]$

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

"box filter"

$G[x,y]$

	0	10	20	30	30				

$$G = H \otimes F$$

Practice with linear filters

Original

0	0	0
0	1	0
0	0	0

?

Source: D. Lowe

Practice with linear filters

0	0	0
0	1	0
0	0	0

Original

Filtered
(no change)

Source: D. Lowe

Practice with linear filters

Original

0	0	0
0	0	1
0	0	0

?

Source: D. Lowe

Practice with linear filters

Original

0	0	0
0	0	1
0	0	0

Shifted left by 1 pixel with correlation

Source: D. Lowe

Practice with linear filters

Original

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

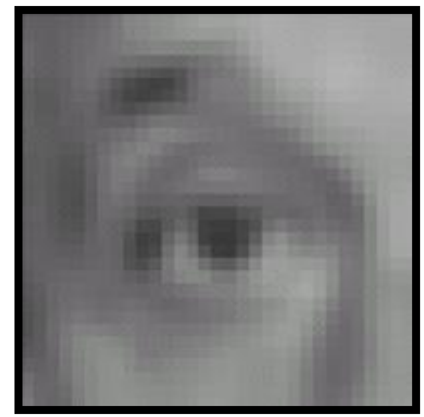
?

Source: D. Lowe

Practice with linear filters

Original

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

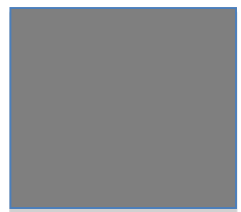


Blur (with a box filter)

Source: D. Lowe

Linear filter

A



h[i,j]

=

f[x,y]

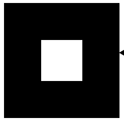
I[x,y]

Mask= 1/25

1,1,1,1,1
1,1,1,1,1
1,1,1,1,1
1,1,1,1,1
1,1,1,1,1

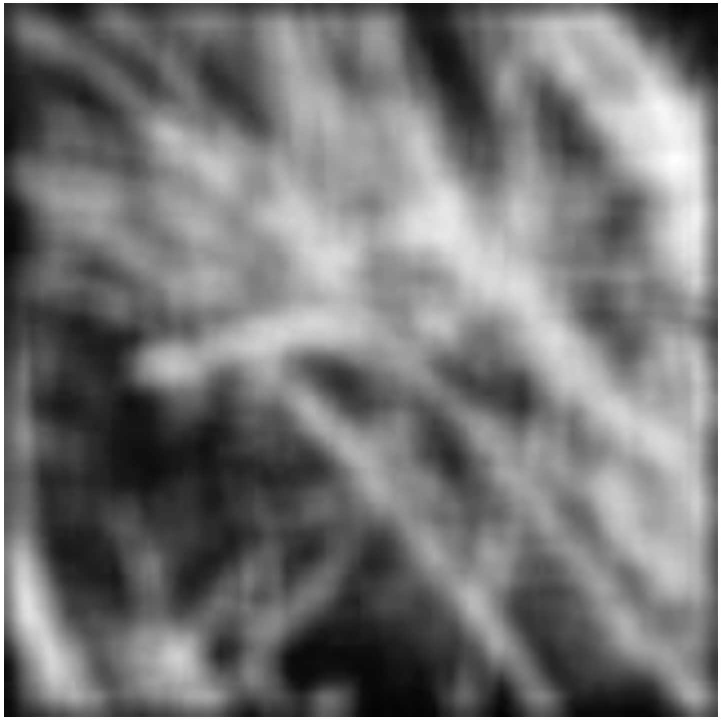
Smoothing by averaging

Weighted filter:



Box filter:
white = high value
black = low value (not zero!)

original



filtered

What if the filter size was 5 x 5 instead of 3 x 3?

Gaussian filter

- What if we want nearest neighboring pixels to have the most influence on the output?

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$F[x, y]$

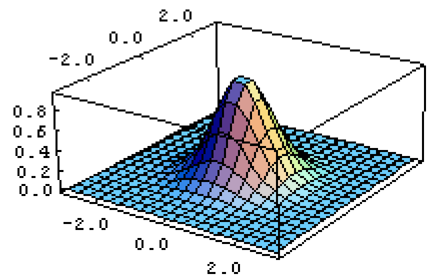
$\frac{1}{16}$

1	2	1
2	4	2
1	2	1

$H[u, v]$

This kernel is an approximation of a 2D Gaussian function:

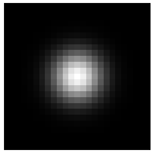
$$h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$

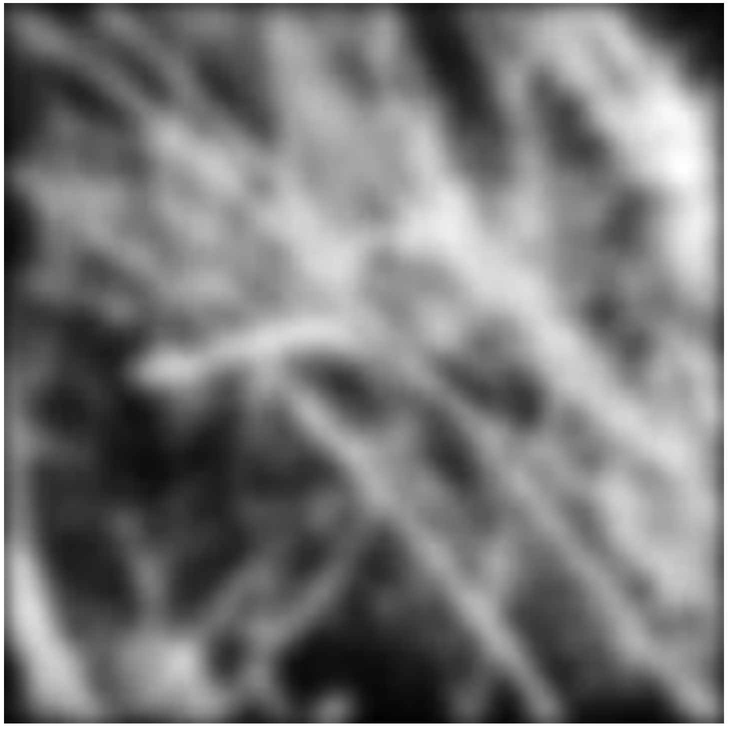


- Removes high-frequency components from the image ("low-pass filter").

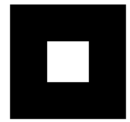
Source: S. Seitz

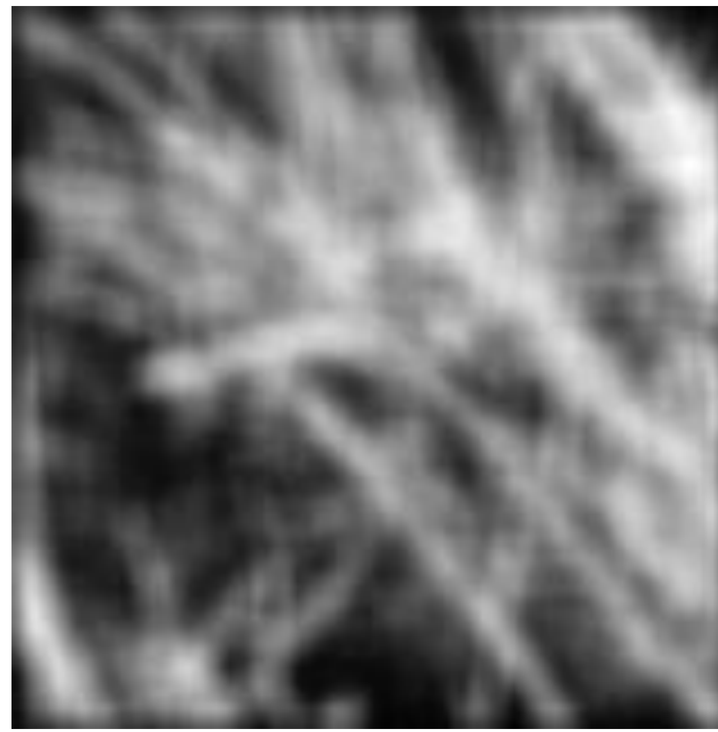
Smoothing with a Gaussian



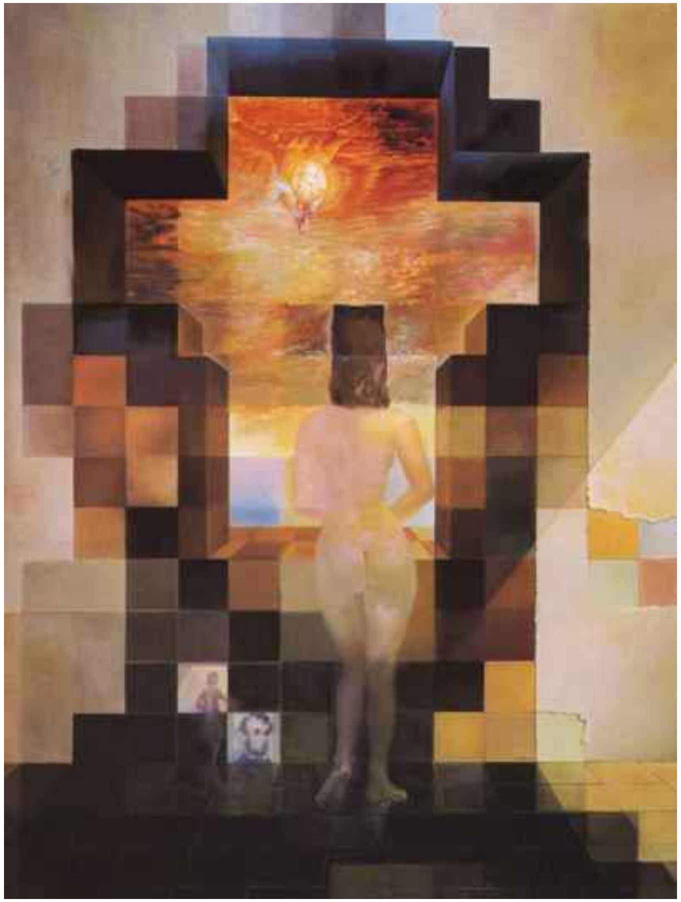


Smoothing by averaging (previous)

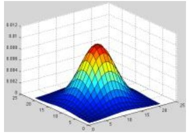




Gaussian filter: Local vs global analysis.



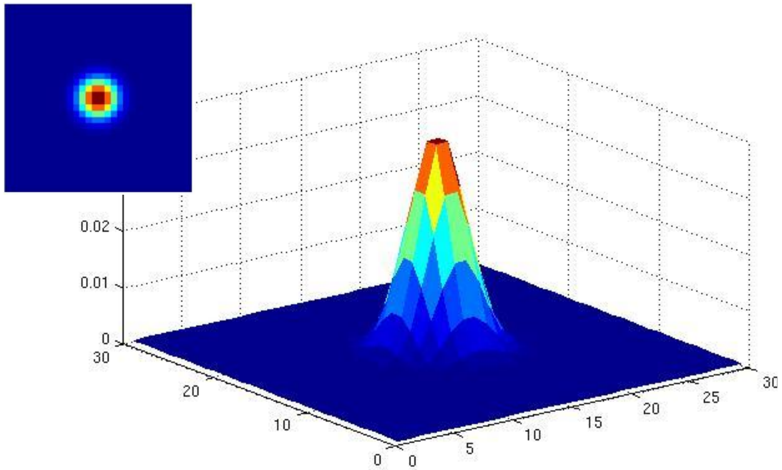
Dali



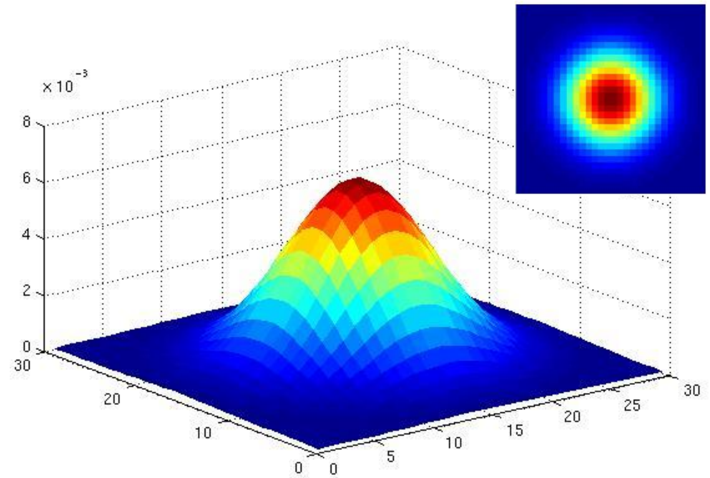
Allows to analyze the global structure of the image, but lose the details.

Gaussian filters

- What parameters do matter here?
  - σ of the mask
  - Two different examples:



σ = 2 with 30 x 30 kernel

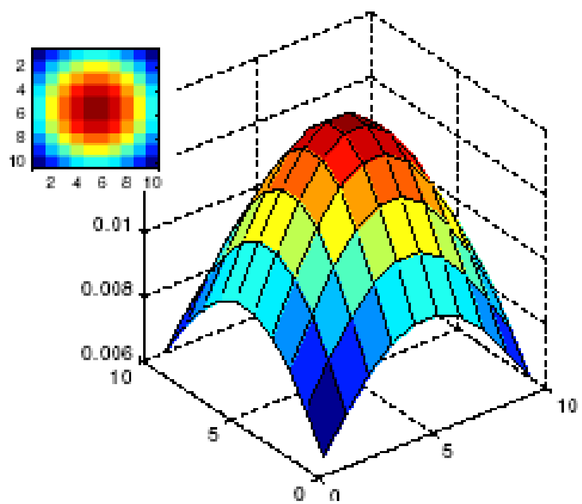


σ = 5 with 30 x 30 kernel

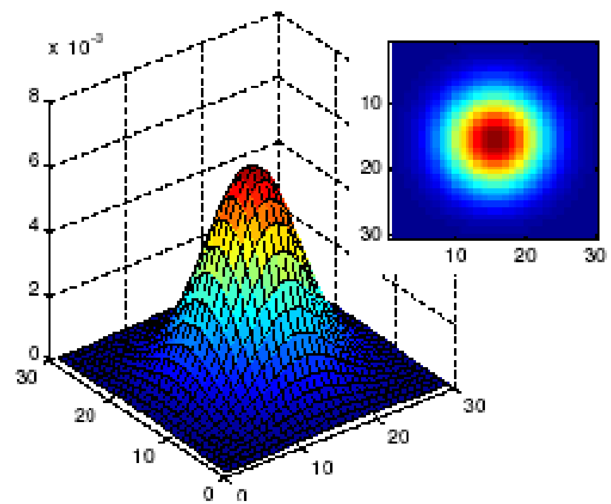
Parameter σ (referred to as scale/width/spread) is the standard deviation of the Gaussian kernel, and controls the amount of smoothing.

Gaussian filters

- What parameters do matter here?
- **Size** of kernel or mask
  - Note that Gaussian function has infinite support, but discrete filters use finite kernels
- Two different examples:



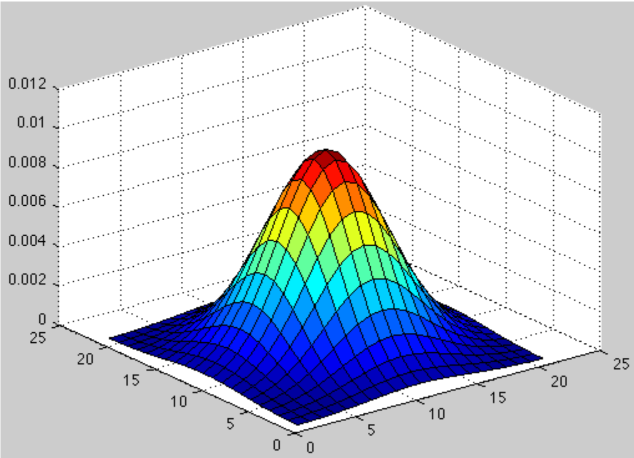
σ = 5 with 10 x 10 kernel



σ = 5 with 30 x 30 kernel

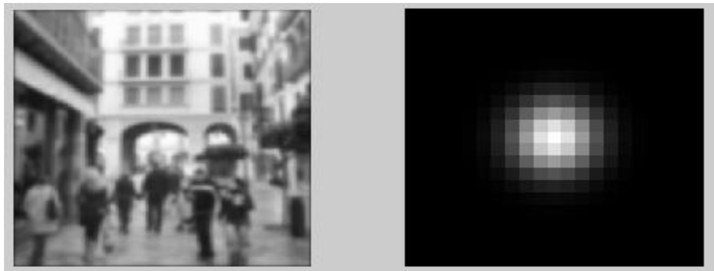
The Gaussian filter

Effect of parameter σ:

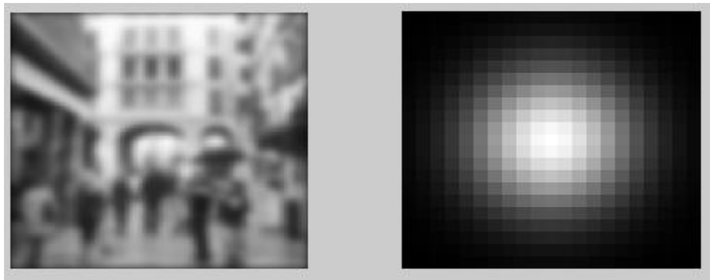


$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

σ=1



σ=2

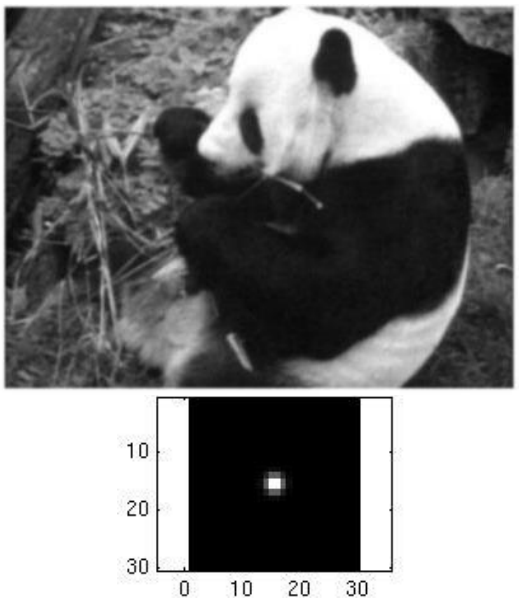


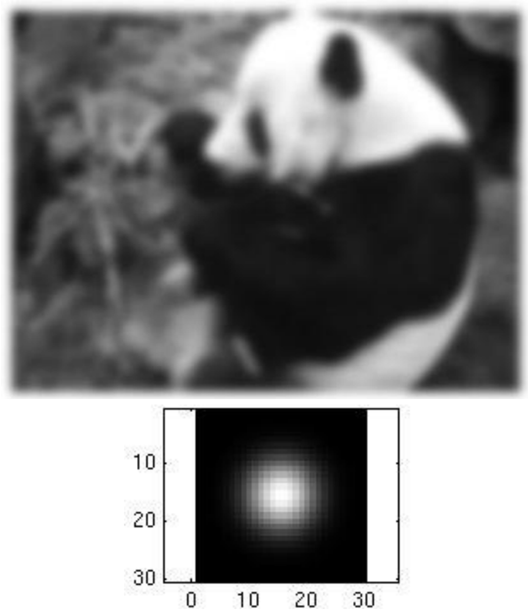
σ=4

```
outim = gaussian(im, sigma=1)
```

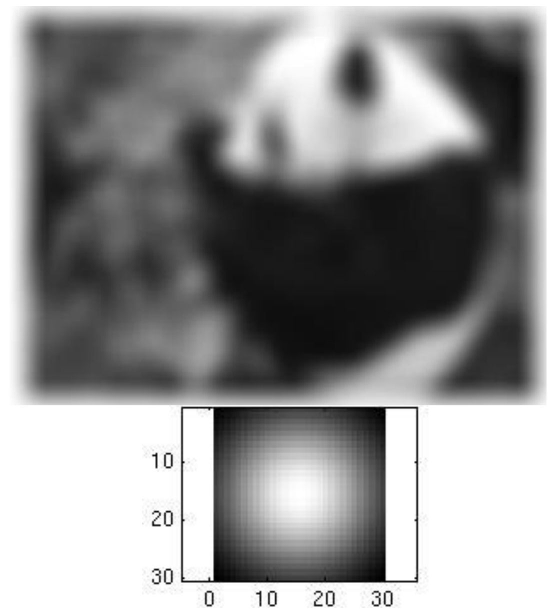
Smoothing with a Gaussian filter

Effect of parameter σ:





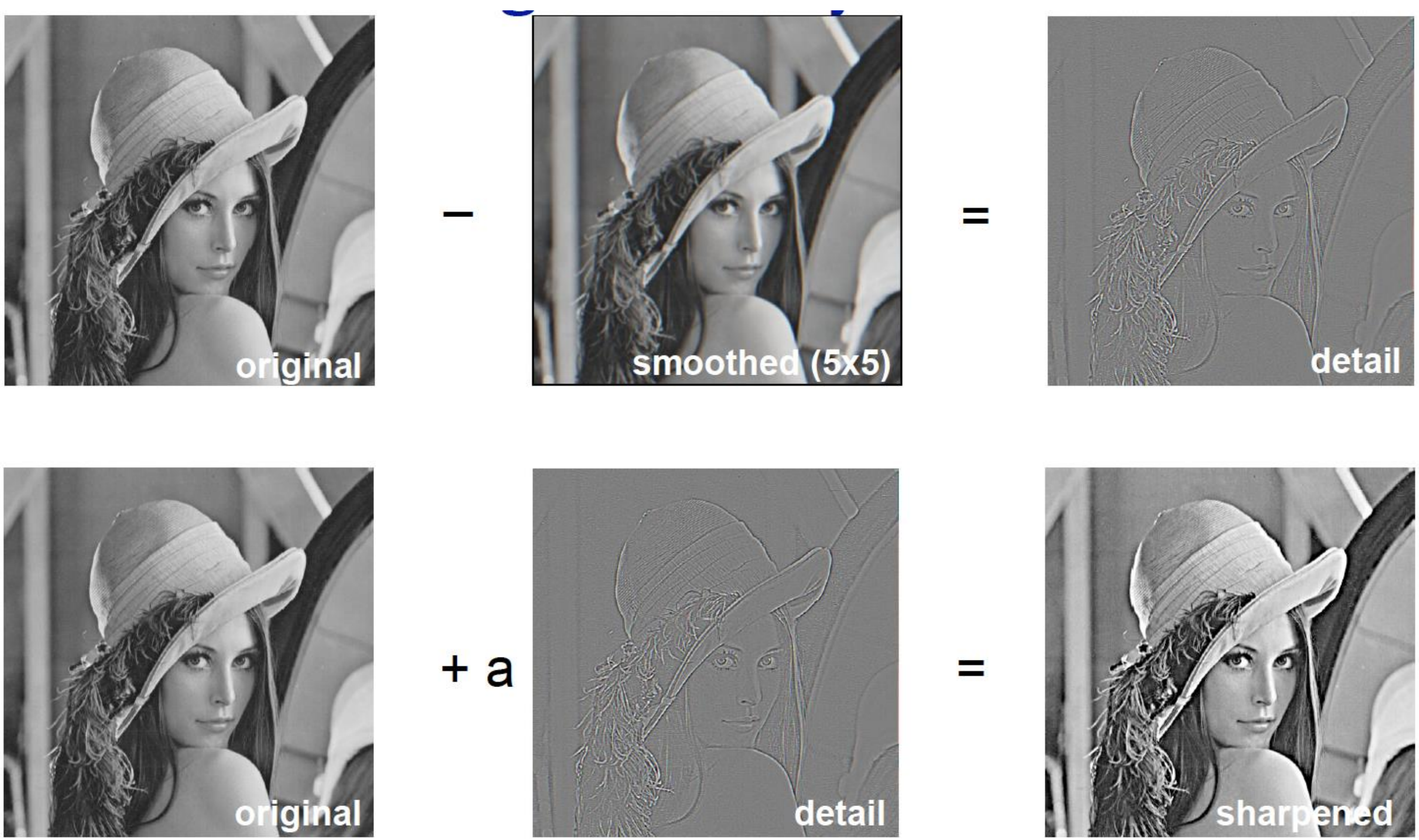
…



Properties of smoothing filters

- Values positive
- Sum to 1 → constant regions same as input
- Amount of smoothing proportional to mask parameters
- Remove “high-frequency” components; “low-pass” filter

Sharpening



Summary

- Images: resolution, “noise”
- Linear filters and convolution useful for
  - Enhancing images (smoothing, removing noise)
    - Box filter
    - Gaussian filter
    - Impact of scale / width of smoothing filter
  - Detecting edges (next lecture)

References

- Source for slides are from the courses of:
  - J. Malik, UBerkeley
  - Prof. K. Grauman, UT-Austin
  - D. Hoiem
  - S. Marschner
  - D. Lowe
  - S. Seitz
  - Some Figures from [Mori et al]

COMPUTATIONAL VISION: Digital Image, Linear Operators, Linear Filters

Master in Artificial Intelligence

Department of Mathematics and Computer Science

2019-2020

UNIVERSITAT DE BARCELONA