UniGe | MaLGa

image processing

Computational Vision

(these slides are based on DSIP material)

main concepts (mostly from DSIP)

gray level / color images

global descriptors (histograms, signatures,...)

linear filters (convolution in the space domain)

local features (edges, corners, ...)

PIXEL CONTENTS

Pixels contents depend on the image type

- Pictorial digital images (photos): Intensity (“black & white”), color
- Range images: Depth information
- Medical images: Radiations absorbance level
- Termal images: Heat
- ....

Dynamic Range

- Total number of distinctive values occurring in the image

this is related to the quantization process...

- it is limited by the number of bit per pixel we may want to use
- it is also limited by the physical dynamic range of the sensor

- to represent black-white images 1 bit is sufficient
- gray level images: usually associate a byte to a pixel 2^8=256 gray levels
- color images: usually 1 byte per channel (“millions of color”)

Color images

- we will consider color RGB images where each pixel is described by a triplet (R,G,B)
- a standard 24-bit image (also called *full color)* associates 1 byte per pixel to each color field (overall 3 bytes per pixel)
- a color image is usually acquired by using 3 filters sensitive to red green and blue. we obtain 3 monochrome images which are then combined in a single image

Color image processing

- In many of the methods we will cover there are two main choices:
  - processing the 3 channels independently and *then* merge the results
  - devise *ad hoc* algorithms for color images
- Some problems are specific for color images (color constancy, ...)

Example problem: image segmentation

detecting groups of close (*connected*) and *similar* pixels

Similarity is evaluated with respect to a specific attribute (brightness, color, texture, motion,...)

Example problem: image segmentation

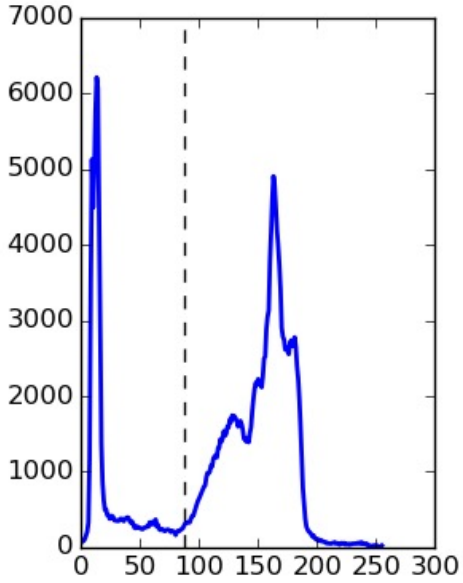
similar brightness: brightness levels that are “close” in the selected coding

common coding for gray level (brightness) images:

8 bits [0-255]

binarization

brightness histogram



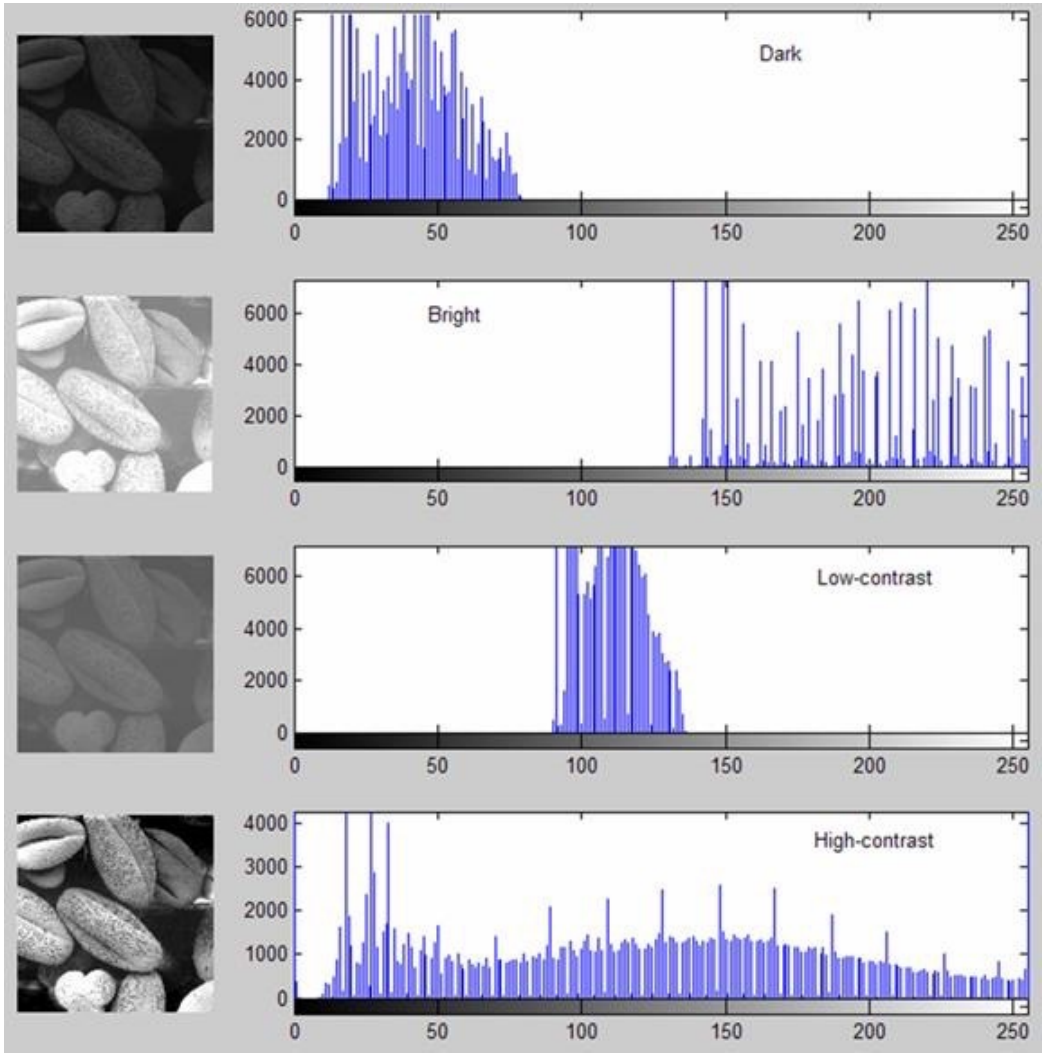
Example problem: image segmentation

similar brightness:

brightness levels that are “close” in the selected coding

careful! in general it’s not so easy to identify an appropriate threshold!

grey-level histograms



Example problem: image segmentation

in the case of color,

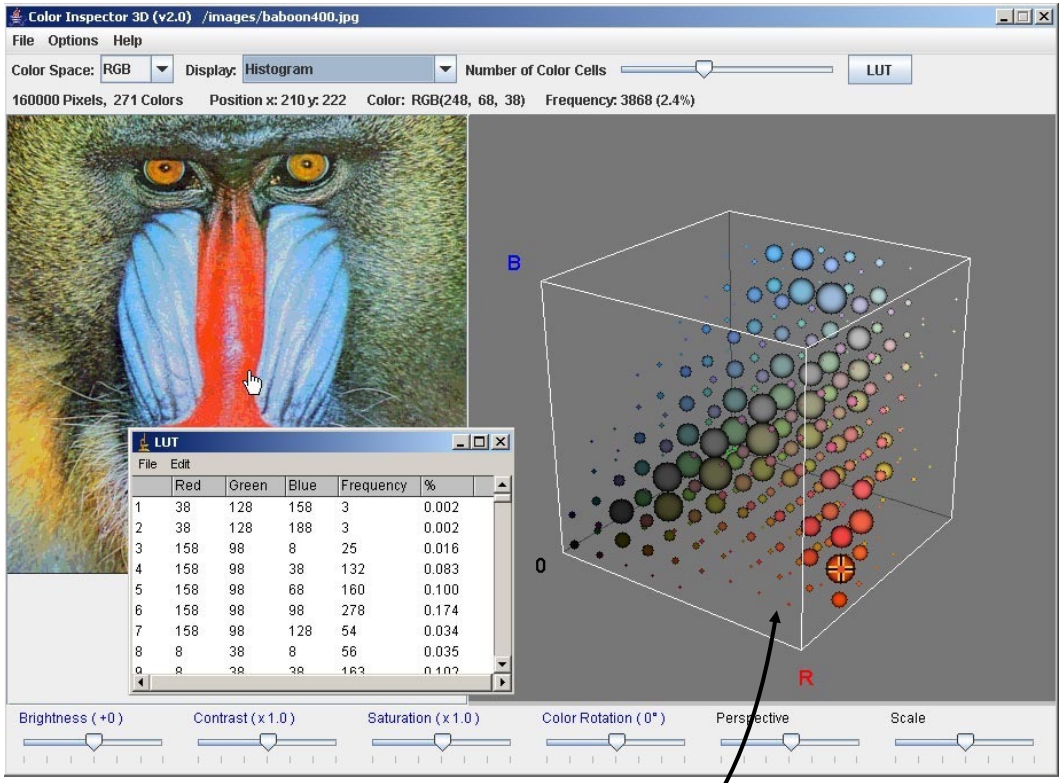
“close” does not mean

similar

a careful choice of the

color space is

recommended

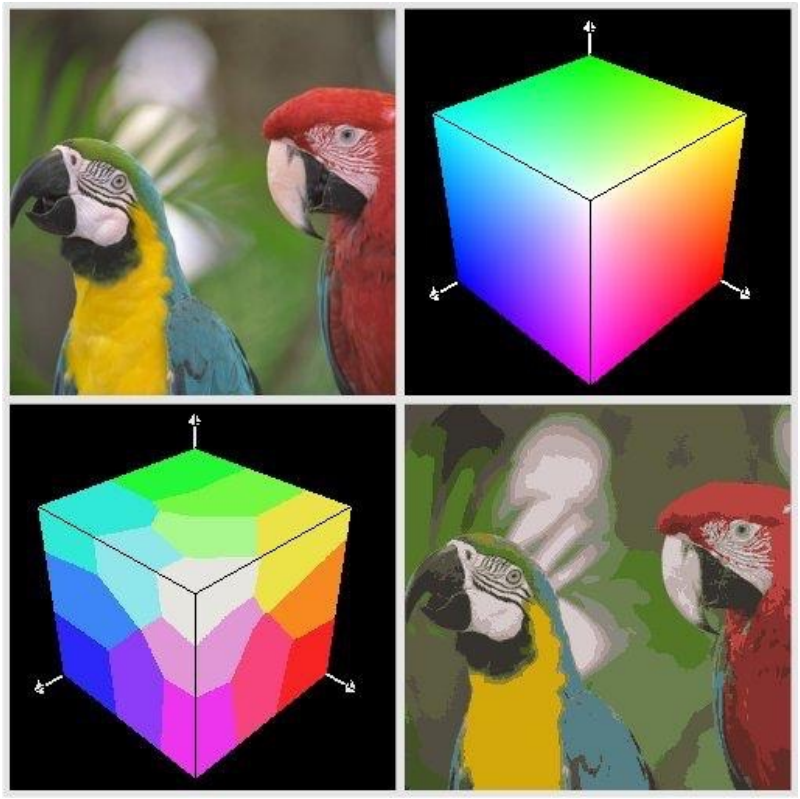


a visualization of a color histogram

Image quantization

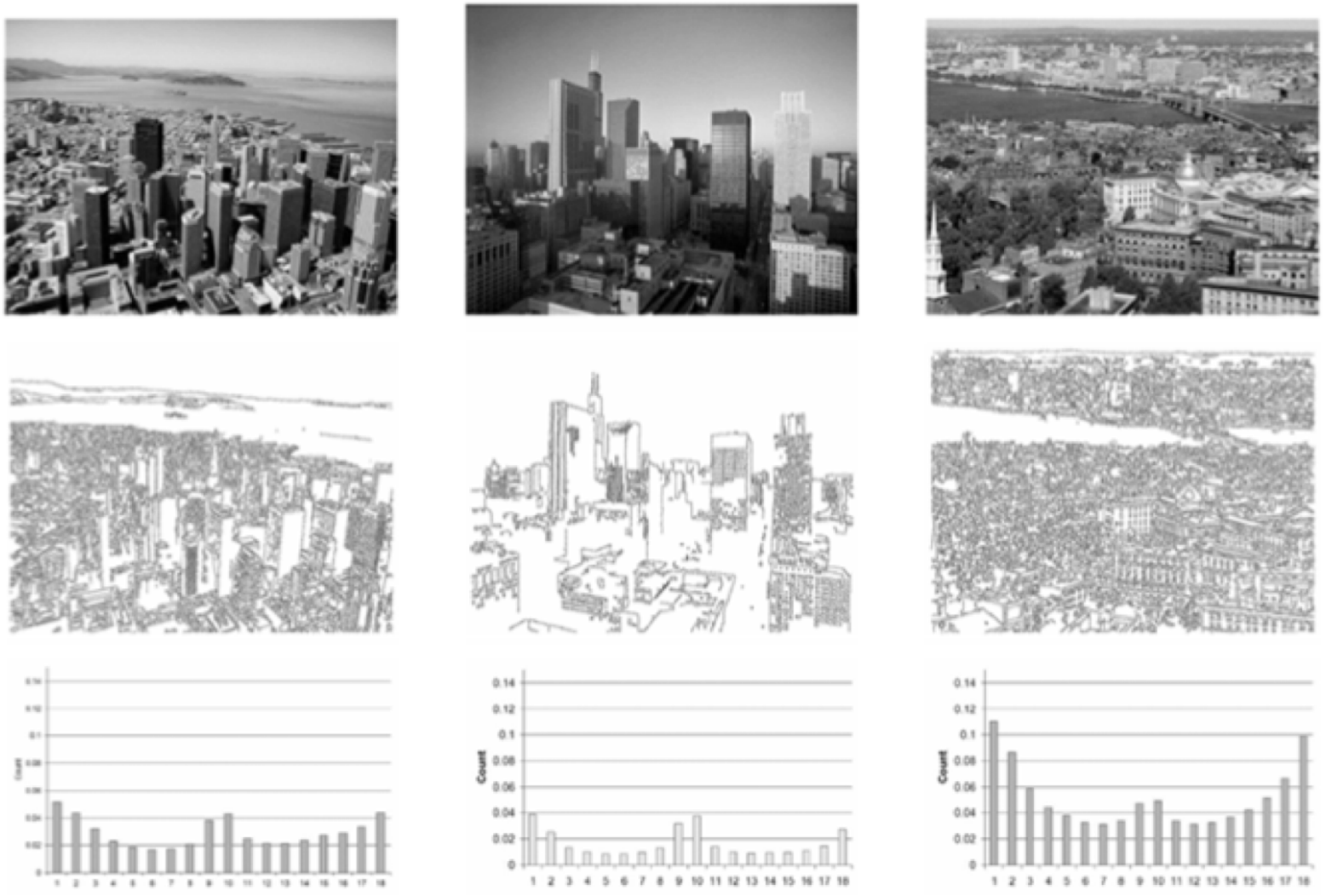
Reducing the image dynamic range

it can be a useful pre-processing to image segmentation



Describing the whole image

histograms may be adopted on raw info (brightness, color, ...) or on higher level information (gradient, texture, ...)



Describing the whole image

histograms can be used as a way of obtaining an overall description of the image content

+ easy to compute, robust to geometrical variations and to scale changes (if normalized)

- incorporate redundant information, clutter, and loose spatial information

Image filters

Read Szeliski
Section 3.2

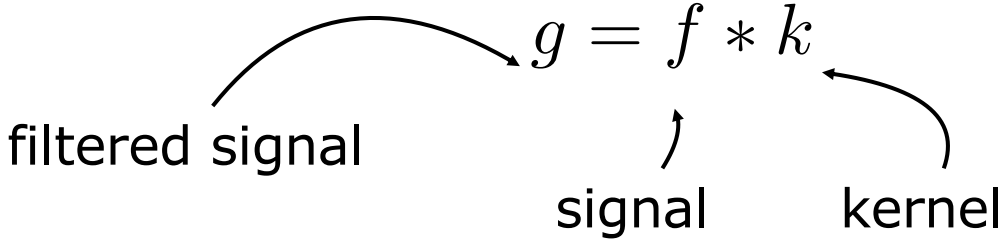
a basic tool in image processing

used for a variety of tasks, included noise reduction and signal enhancement

we will consider linear filters in the space domain (will be useful in different parts of the course)

Linear filtering in space - convolution

In linear cases, filtering corresponds to applying to a signal f a convolution operator



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

45	60	98	127	132	133	137	133
46	65	98	123	126	128	131	133
47	65	96	115	119	123	135	137
47	63	91	107	113	122	138	134
50	59	80	97	110	123	133	134
49	53	68	83	97	113	128	133
50	50	58	70	84	102	116	126
50	50	52	58	69	86	101	120

$f(x,y)$

*

0.1	0.1	0.1
0.1	0.2	0.1
0.1	0.1	0.1

$h(x,y)$

=

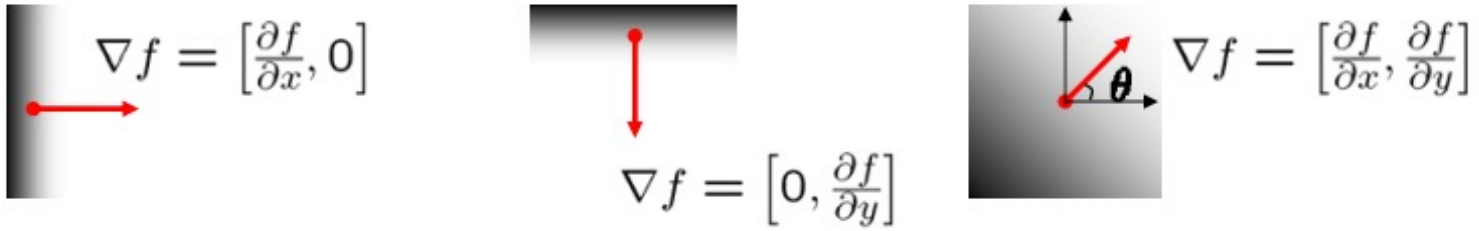
69	95	116	125	129	132
68	92	110	120	126	132
66	86	104	114	124	132
62	78	94	108	120	129
57	69	83	98	112	124
53	60	71	85	100	114

$g(x,y)$

Image gradient

The gradient points in the direction of most rapid change in intensity

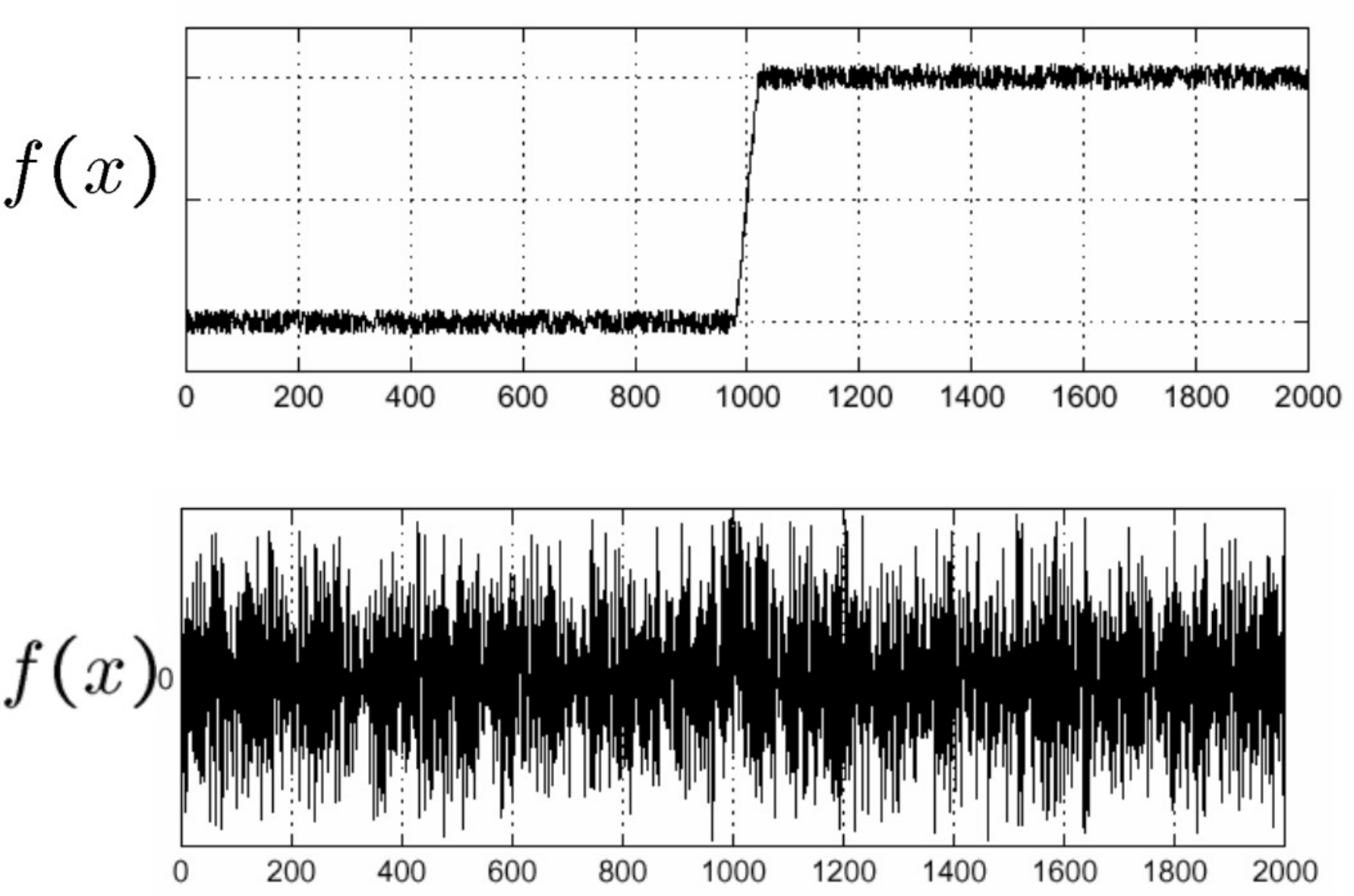
$$\nabla f = grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$



Gradient magnitude: $M(x, y) = \sqrt{g_x^2 + g_y^2}$

Gradient orientation: $\theta(x, y) = \arctan \frac{g_y}{g_x}$

The effect of noise



based on S. Seitz slides

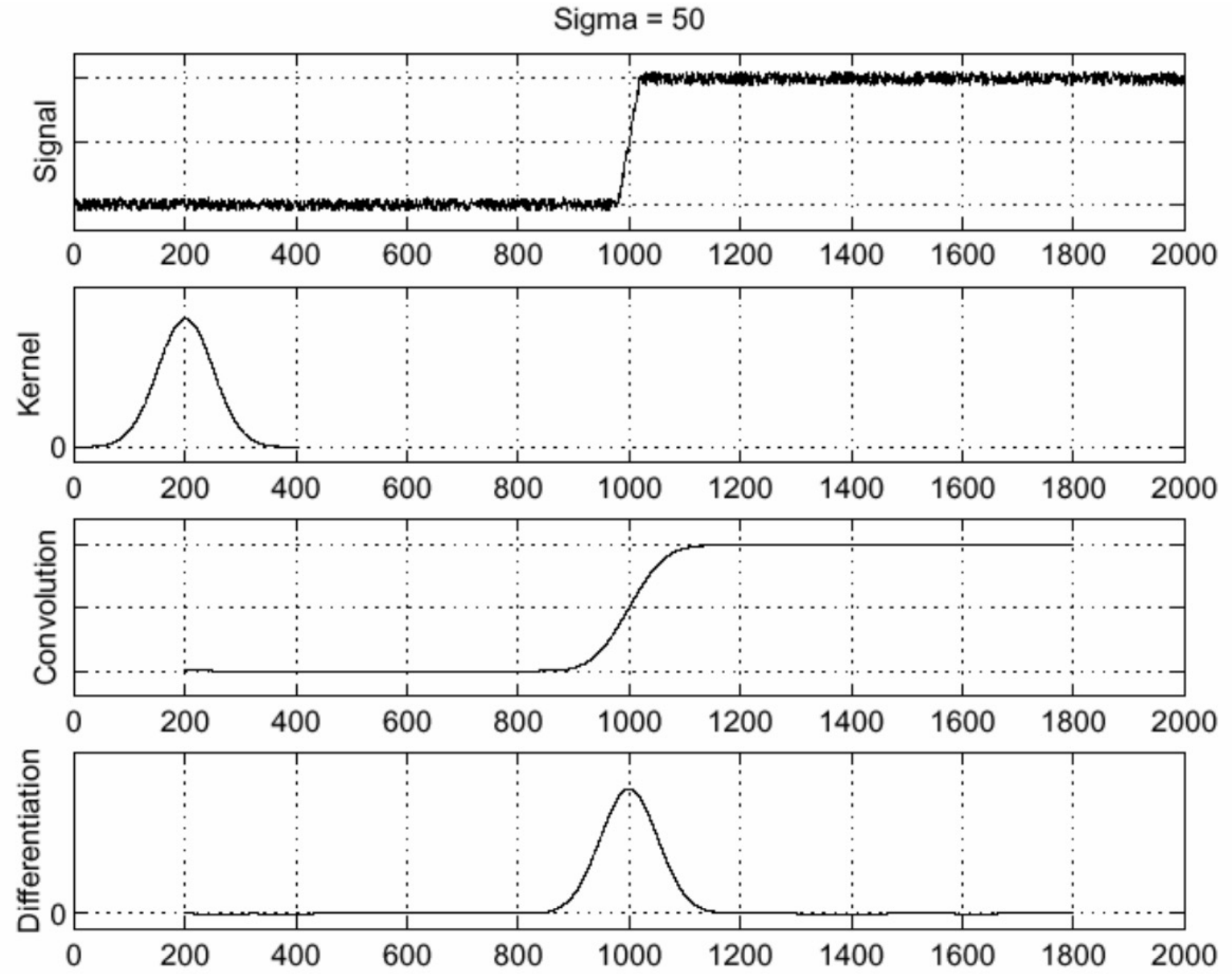
Solution: smooth first

f

h

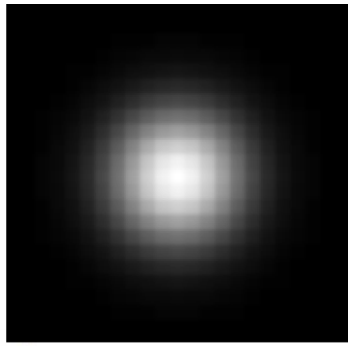
$h \star f$

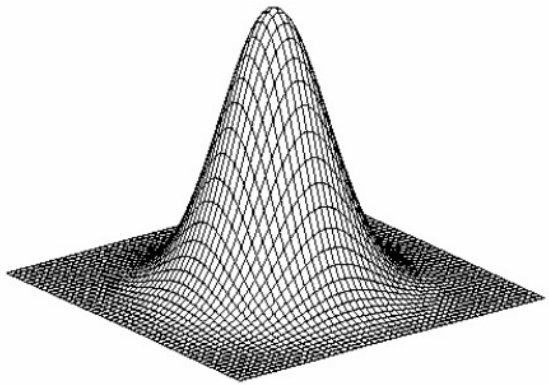
$\frac{\partial}{\partial x}(h \star f)$



based on S. Seitz slides

Smoothing filters (e.g. Gaussian)





Gaussian

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

Solution: convolution properties

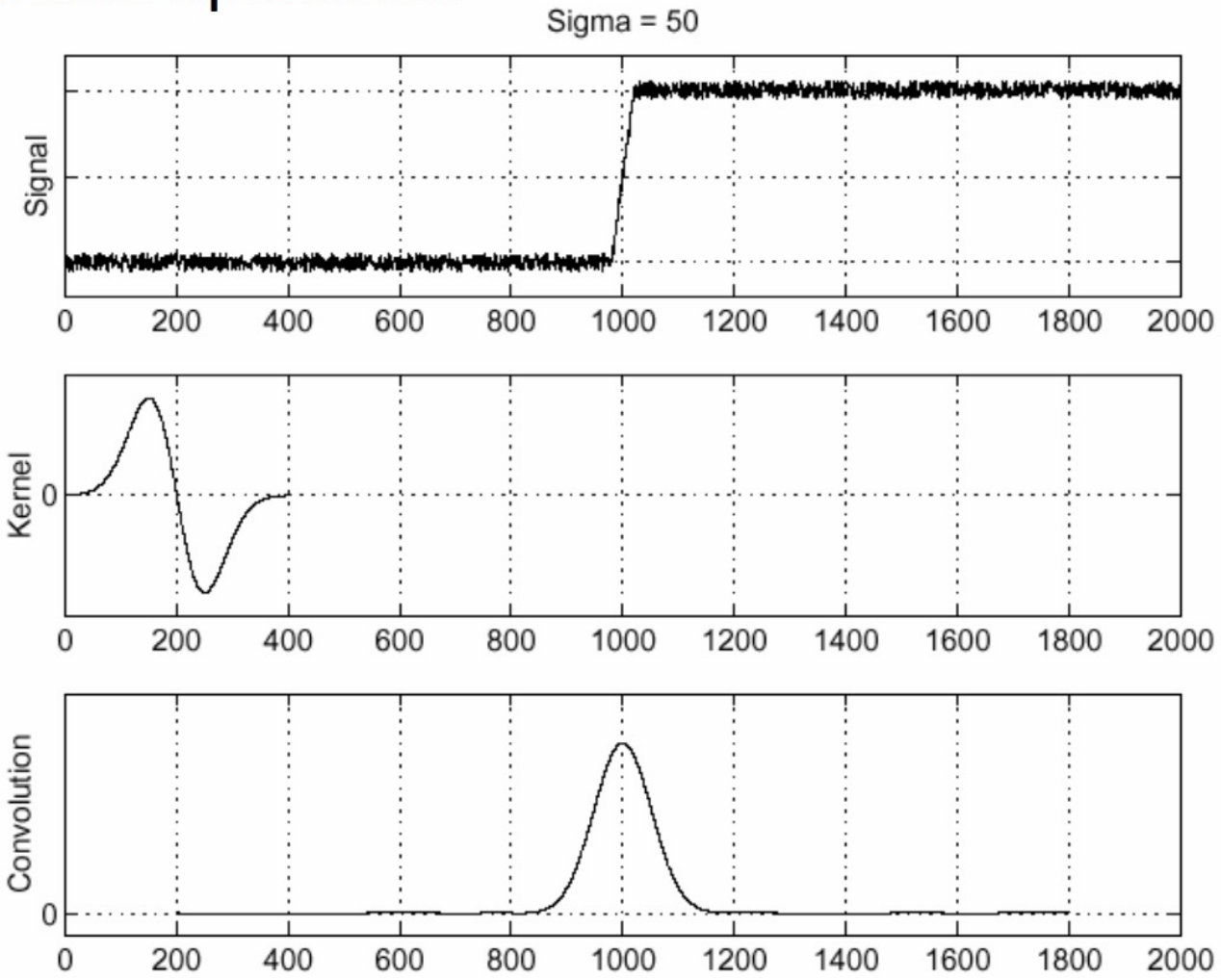
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

This saves us one operation:

f

$\frac{\partial}{\partial x}h$

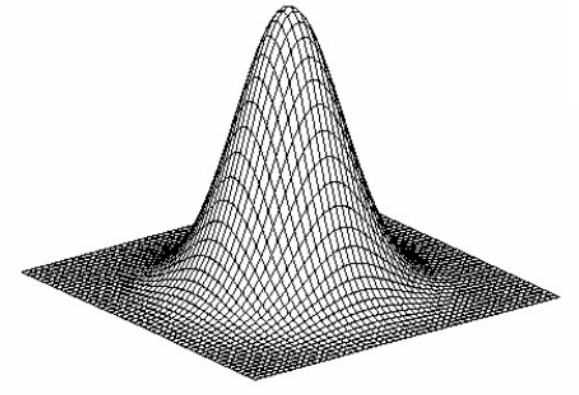
$(\frac{\partial}{\partial x}h) \star f$



based on S. Seitz slides

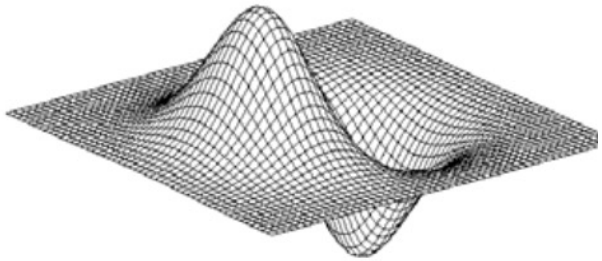
Solution: convolution properties

all in one filter...



Gaussian

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

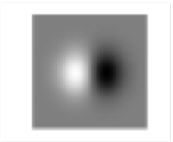


derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

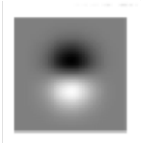
Enhancement filters

The first x derivative of a Gaussian is

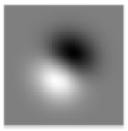


$$G_1^{0^o}(x,y) = \frac{\partial}{\partial x} e^{-(x^2+y^2)} = -2xe^{-(x^2+y^2)}$$

The function rotated of 90 degrees is



$$G_1^{90^o}(x,y) = \frac{\partial}{\partial y} e^{-(x^2+y^2)} = -2ye^{-(x^2+y^2)}$$



$$G_1^{\theta}(x,y) = \cos(\theta)G_1^{0^o} + \sin(\theta)G_1^{90^o}$$

basis filters

Local features: edges

Edges **are pixels at which the image values undergo a *sharp variation***

Edge detection: **given a image locate edges most likely to be generated by scene elements and not by noise**

- - Noise smoothing (low pass)
  - Edge enhancement (high pass)
  - Edge localization (thresholding)

Local features: edges

Noise smoothing
(gaussian filter)

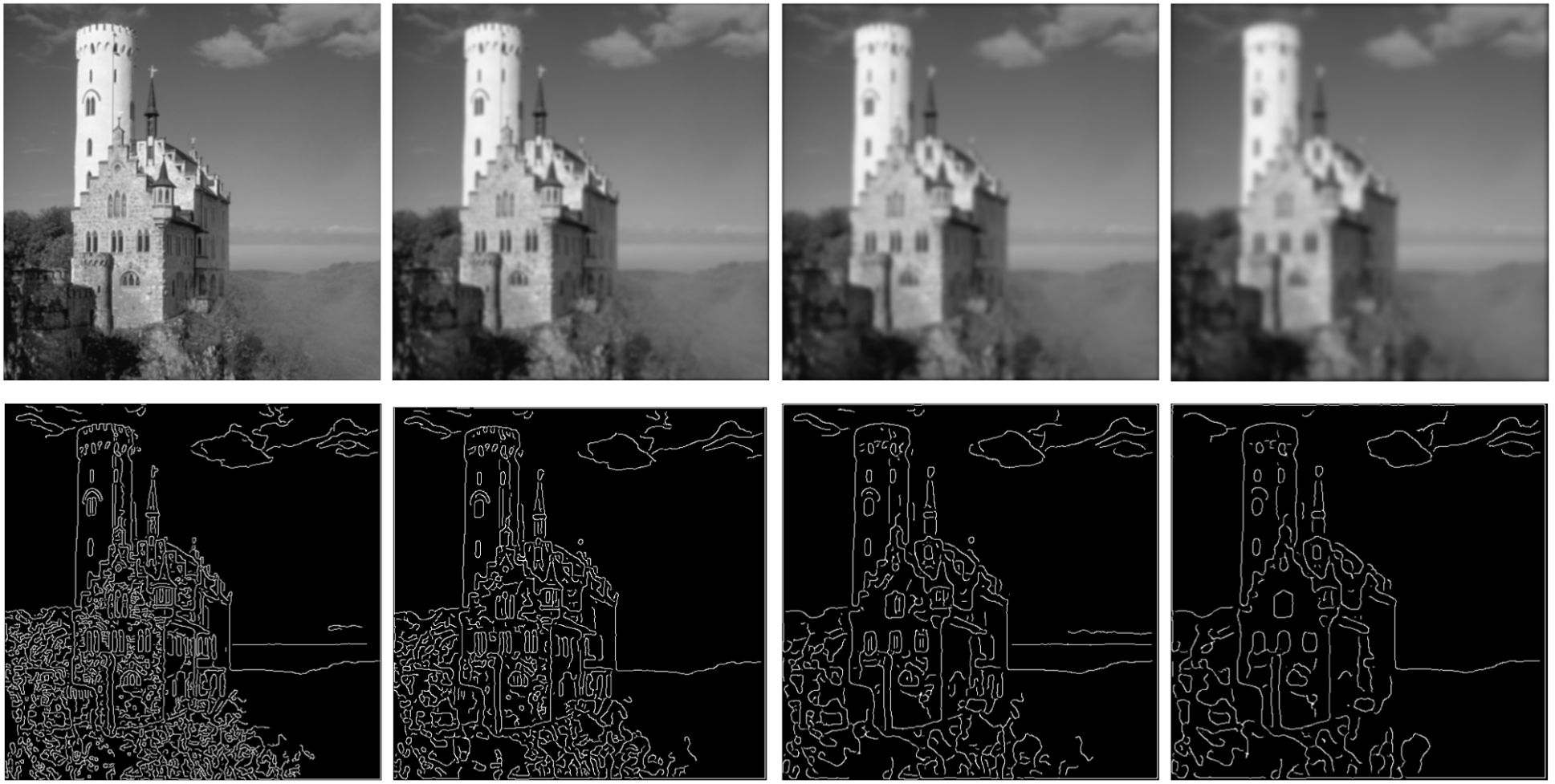
Edge enhancement
(gradient magnitude)

Edge localization
(non-maxima suppression
+ thresholding)

Reference algorithm: Canny edge detector

TRADE-OFF Localization - Detection

on the selection of appropriate parameters



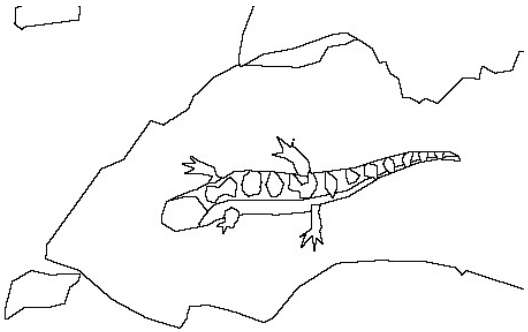
Edges or objects boundaries?

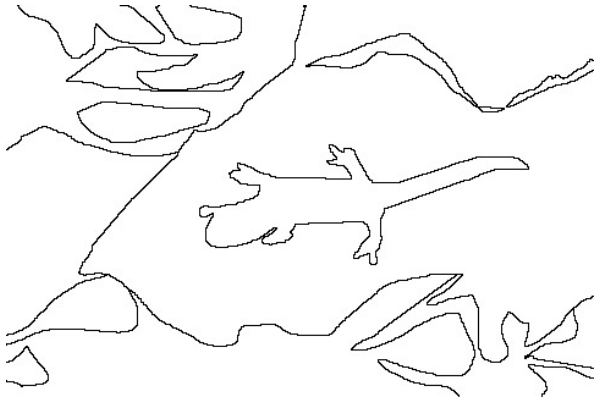
As a side note: if we are interested in image understanding why are we looking for edges?

edge detection —> boundary detection

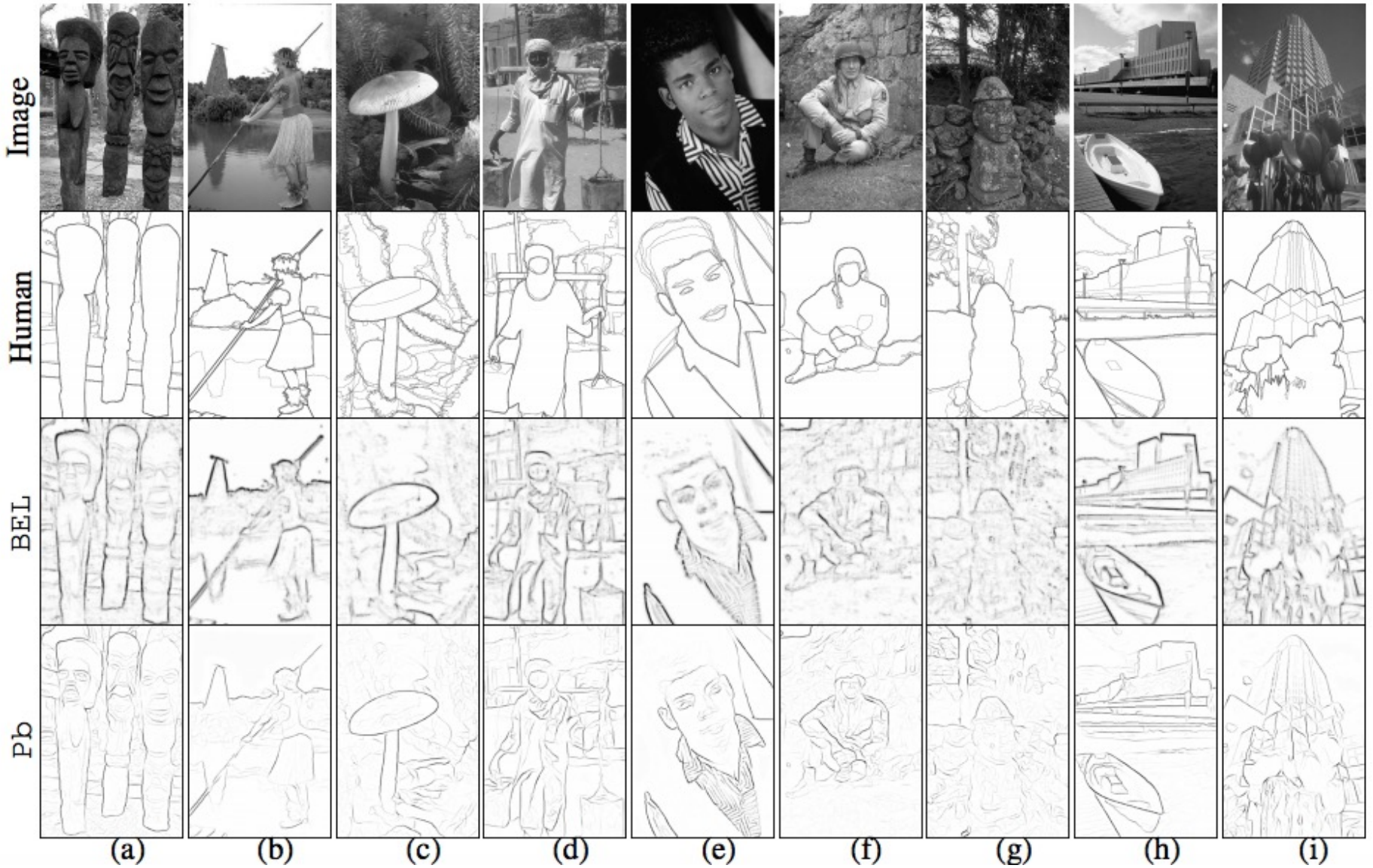
Supervised approach: ask humans to label boundaries and learn them from examples

(see the Berkeley Segmentation Dataset BSDS500)





Edges or objects boundaries?



UniGe | MaLGa

Dollar et al, 2006: Supervised learning of edges and objects boundaries

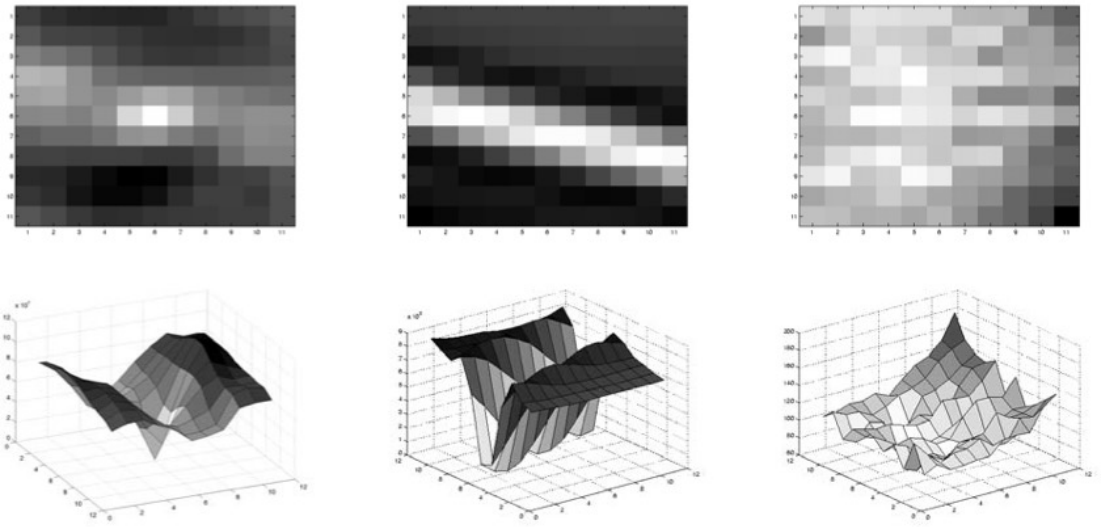
Corners: GOOD FEATURES TO MATCH

- We observe how patches with gradients in at least two (significantly) different orientations are the easier to localize
- This can be formalized by analysing a simple matching criterium (Summed Square Difference)
- In particular we use it to check how stable the patch is wrt small variations in position **u** (SSD **autocorrelation function**):

$$E_{AC}(\mathbf{u}) = \sum_i [I(\mathbf{x}_i + \mathbf{u}) - I(\mathbf{x}_i)]^2$$

AUTOCORRELATION

(a)



AUTOCORRELATION

– Using a Taylor series expansion

$$I(\mathbf{x}_i + \mathbf{u}) = I(\mathbf{x}_i) + \nabla I(\mathbf{x}_i) \cdot \mathbf{u} + \mathcal{O}(\mathbf{x}_i^2)$$

– we obtain an auto-correlation function as follows

$$\begin{aligned} E_{AC}(\mathbf{u}) &= \sum_i [I(\mathbf{x}_i + \mathbf{u}) - I(\mathbf{x}_i)]^2 \\ &\sim \sum_i [I(\mathbf{x}_i) + \nabla I(\mathbf{x}_i) \cdot \mathbf{u} - I(\mathbf{x}_i)]^2 \\ &= \sum_i [\nabla I(\mathbf{x}_i) \cdot \mathbf{u}]^2 \\ &= \mathbf{u}^\top A \mathbf{u} \end{aligned}$$

$$A = \begin{bmatrix} \sum I_x^2 & \sum_p I_x I_y \\ \sum I_x I_y & \sum_p I_y^2 \end{bmatrix}$$

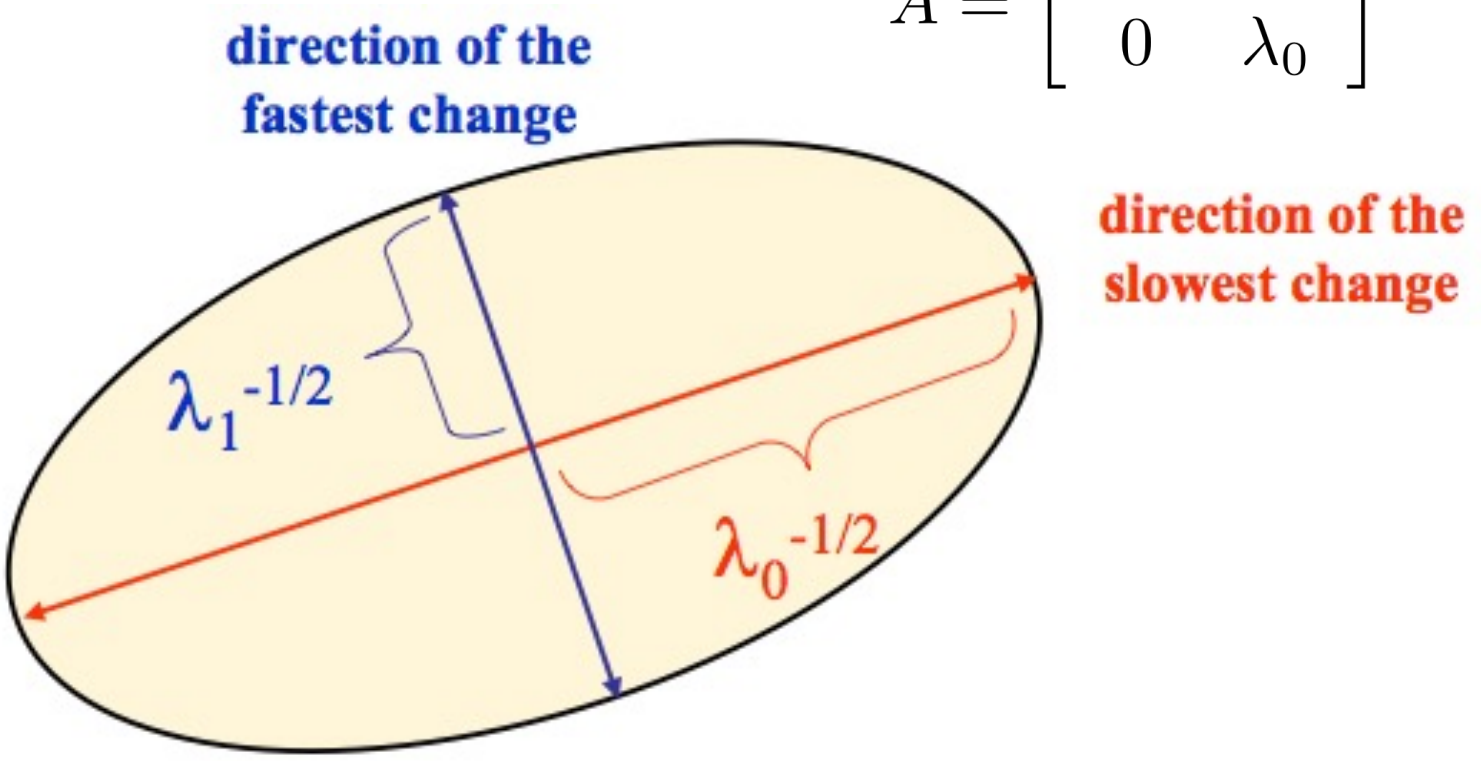
AUTOCORRELATION

– Autocorrelation matrix and its eigenvalues

$$A = \begin{bmatrix} \sum I_x^2 & \sum_p I_x I_y \\ \sum I_x I_y & \sum_p I_y^2 \end{bmatrix}$$

$$\mathrm{eig}(A) = [\lambda_1, \lambda_0] \quad \lambda_1 > \lambda_0$$

$$A = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_0 \end{bmatrix}$$



Local features: corners

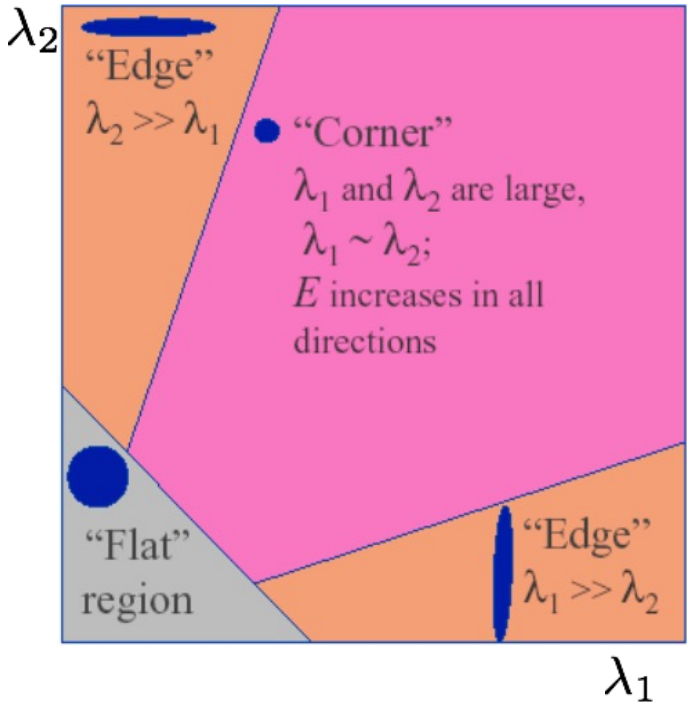
Corners correspond to points where the image gradient varies in at least two directions.

they can be detected by computing and analyzing the autocorrelation matrix A of a patch around each point

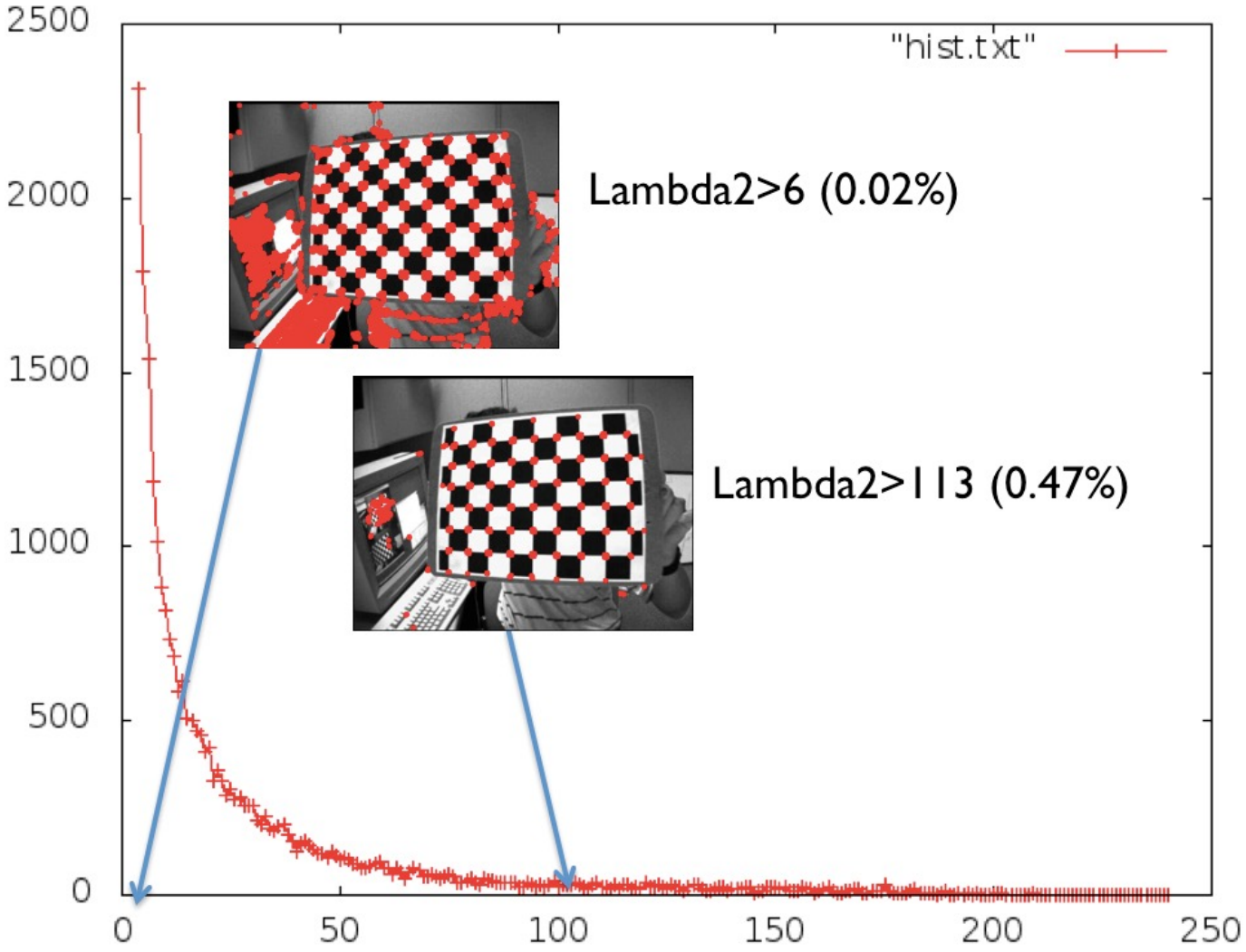
$$\nabla I = [I_x, I_y]^\top$$

$$A = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

$$eig(A) = [\lambda_2, \lambda_1] \lambda_2 > \lambda_1$$



Local features: corners



Histogram of the Smallest eigenvalue (on all points of an image)

on the selection of appropriate parameters

UniGe | MaLGa

What next? ...

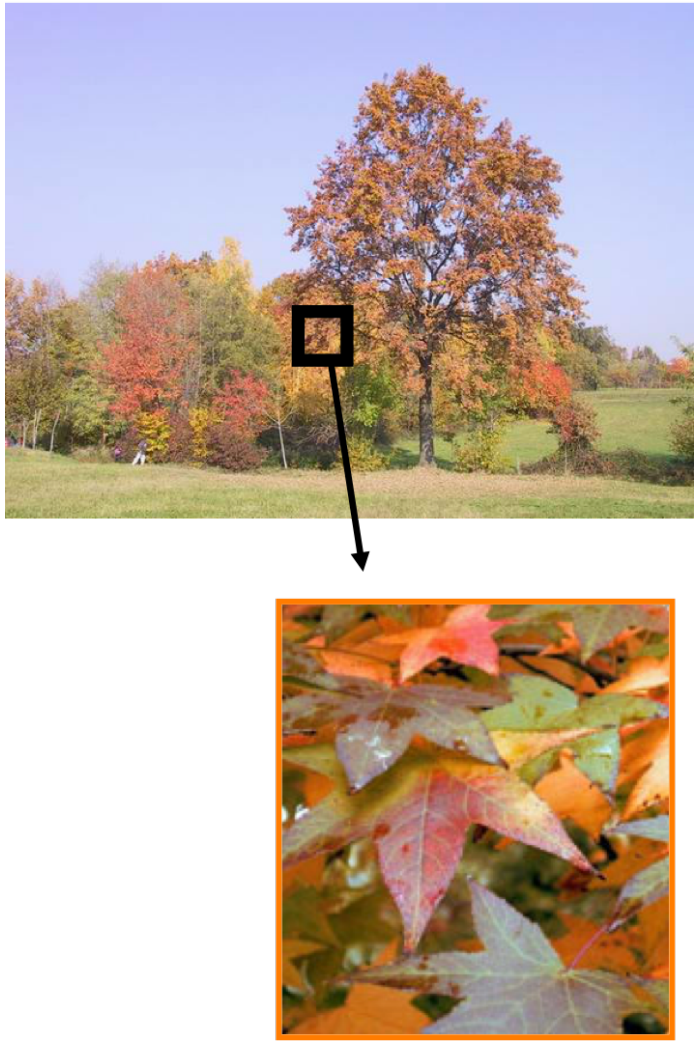
... how will we proceed in the next classes?
Some anticipations

Local features and scale

Scale changes affect image content dramatically

Objects may look very different, with new details emerging as we get close

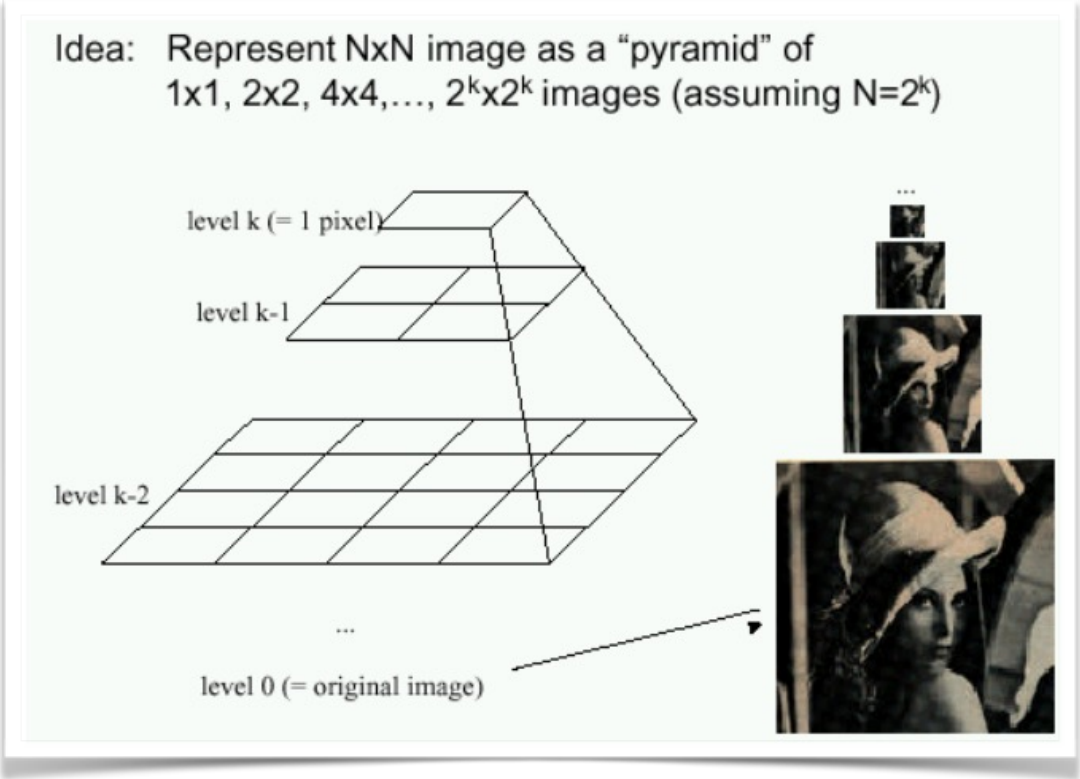
This general observation has an impact on image understanding since the very first processing stages (e.g. local features detection)



How to incorporate scale in our representations

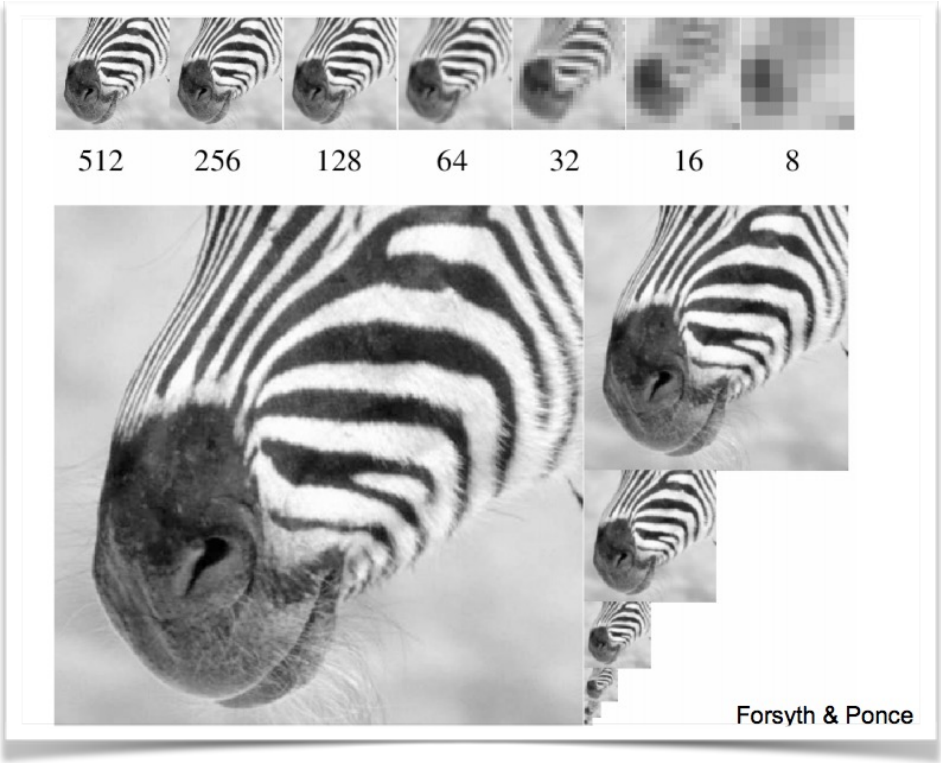
Pyramids

Gaussian Pyramid [Burt & Adelson, 1983]

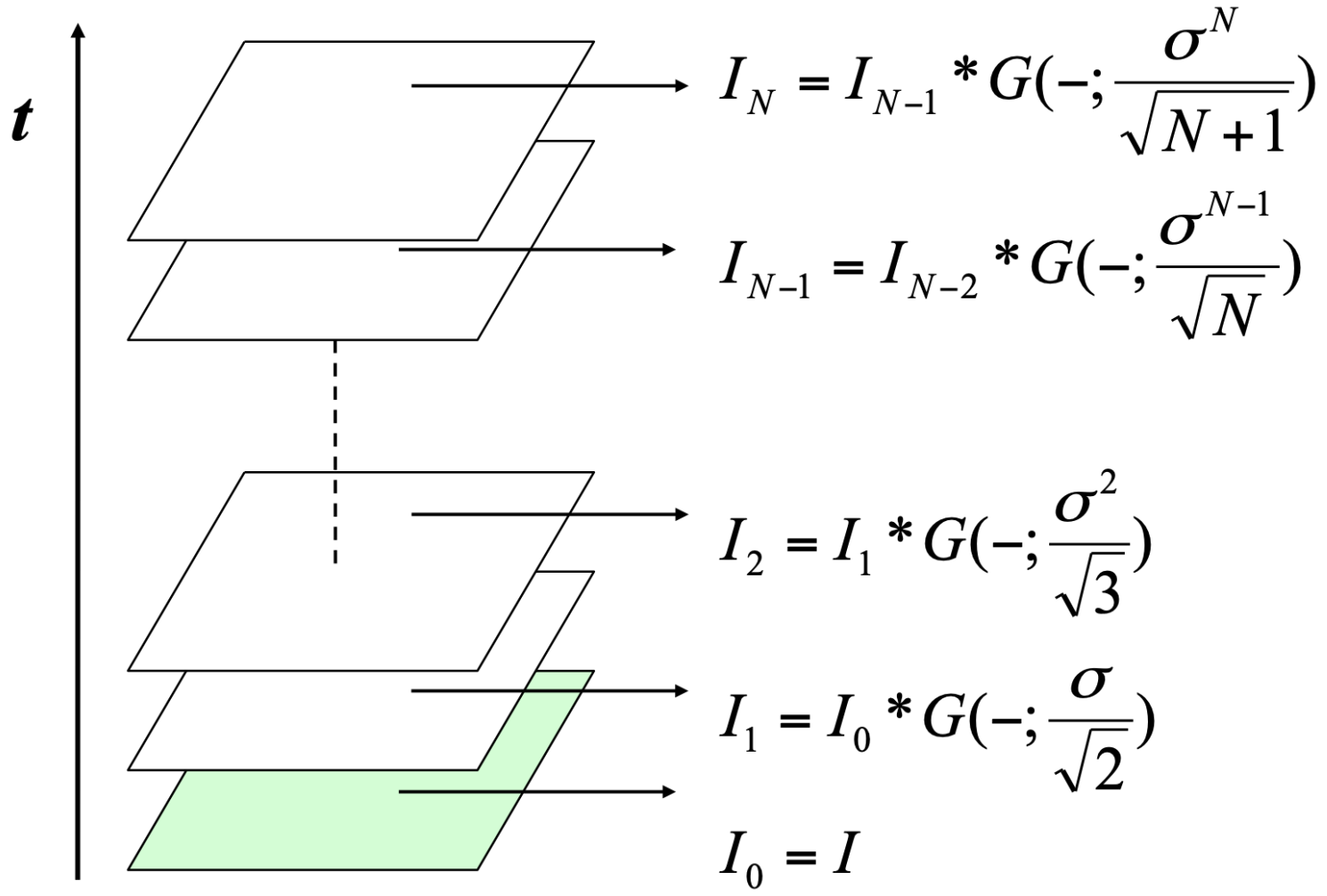


Useful for
– Coarse to fine computations

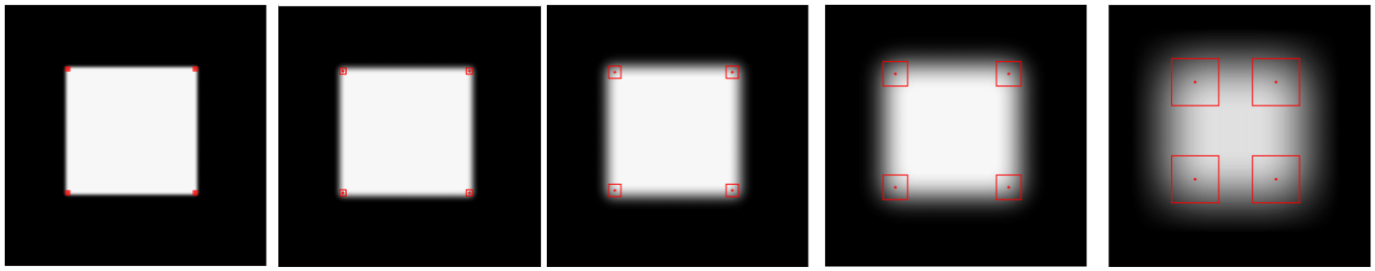
– Feature matching across scales
– Searching over scale
- .....

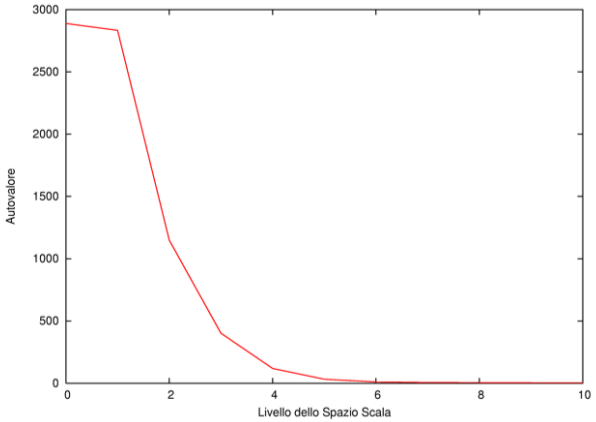


scale-space implementation



corners and scale-space

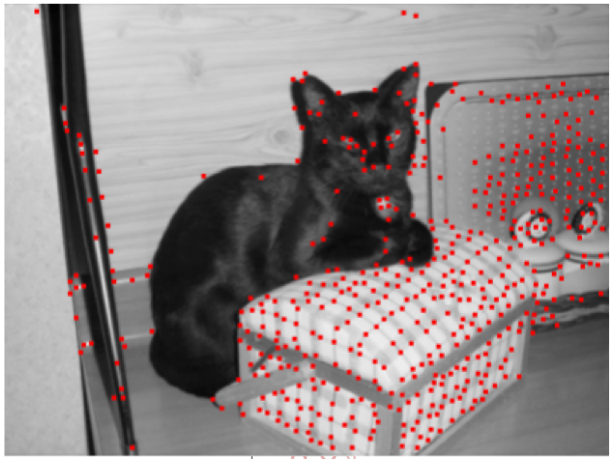




- compute corners at each image layer on an appropriate neighbourhood

- for each corner choose the most appropriate scale and suppress the others (for this, you'd need a *corner SCALE signature (Lindberg 1994)*)

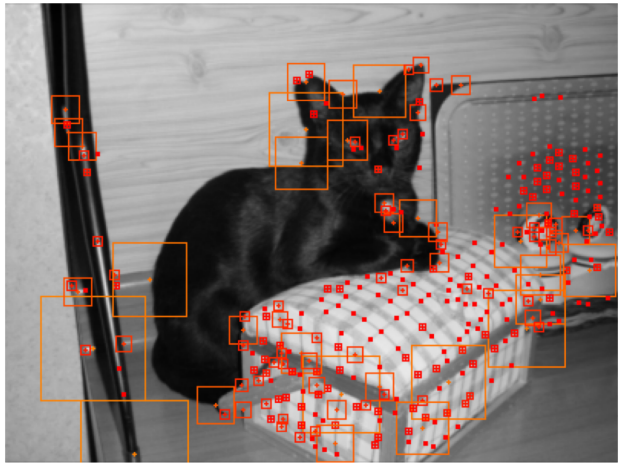
level1



....

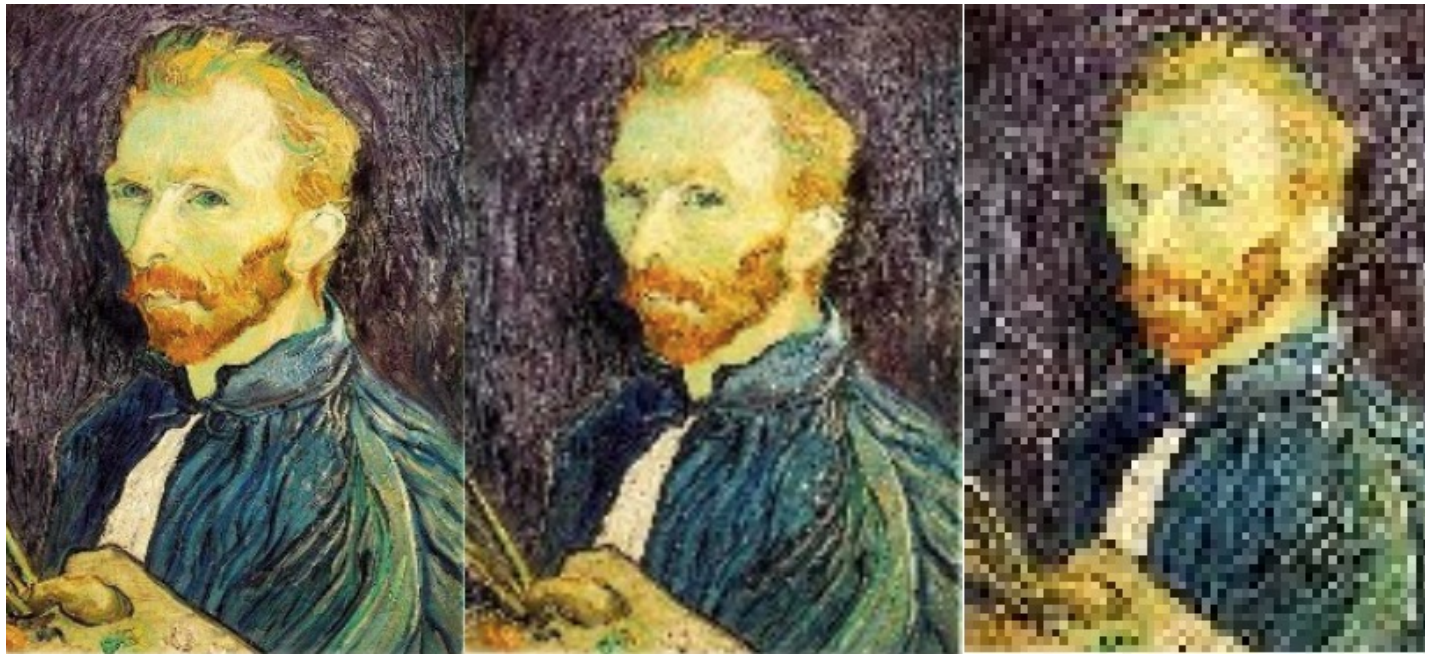
level5

output

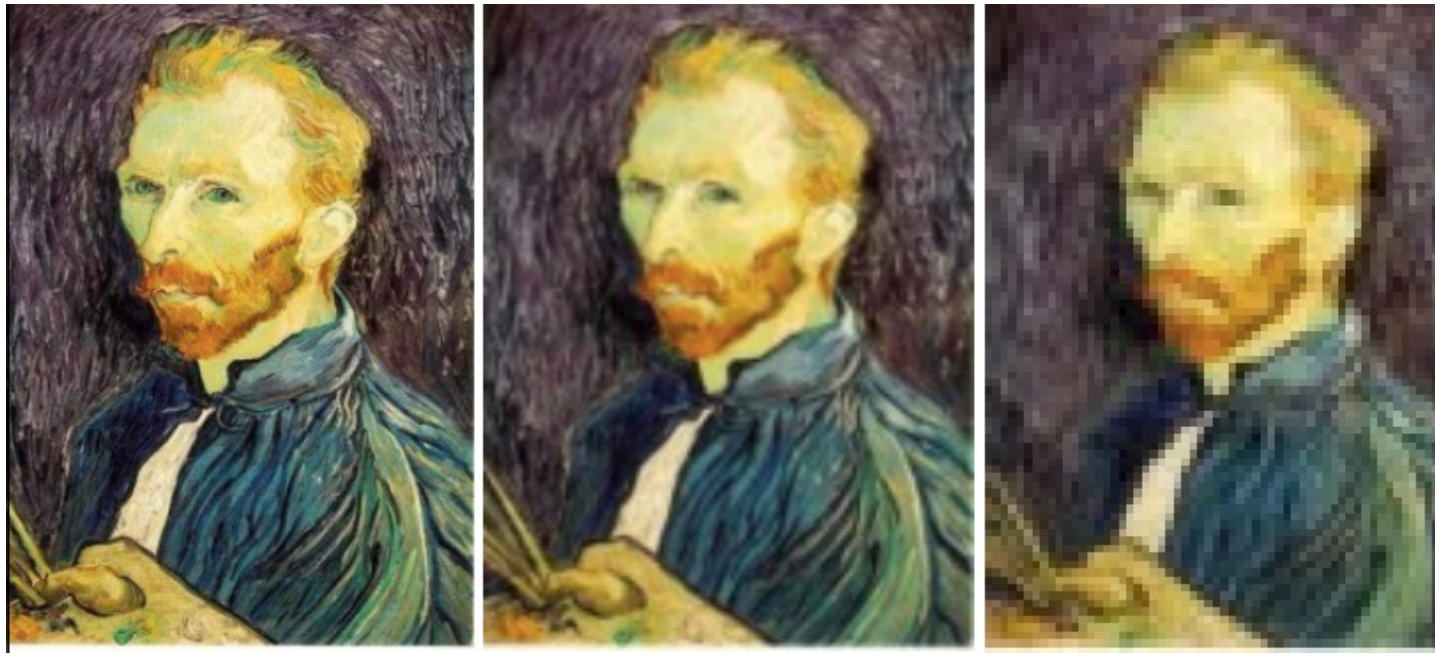


Aliasing

When downsampling we may have troubles...



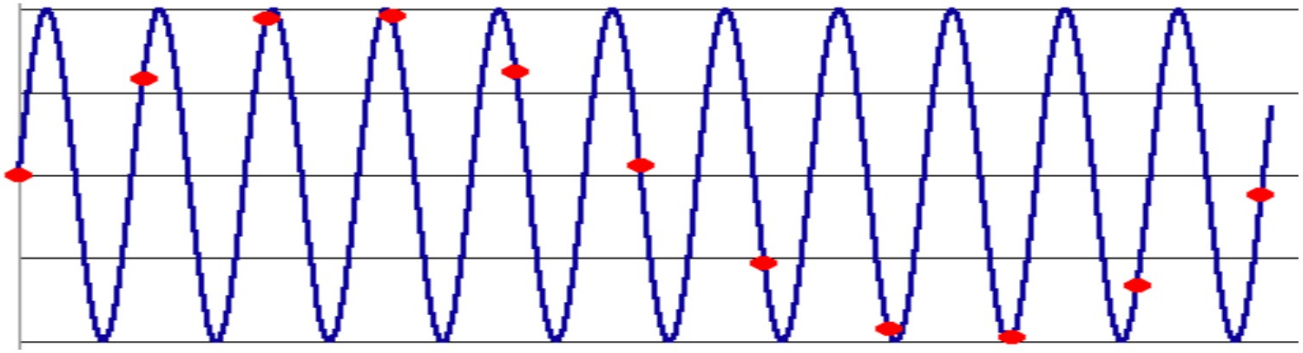
A good practice: first smooth, then sub-sample



based on K. Grauman slides

More details on aliasing

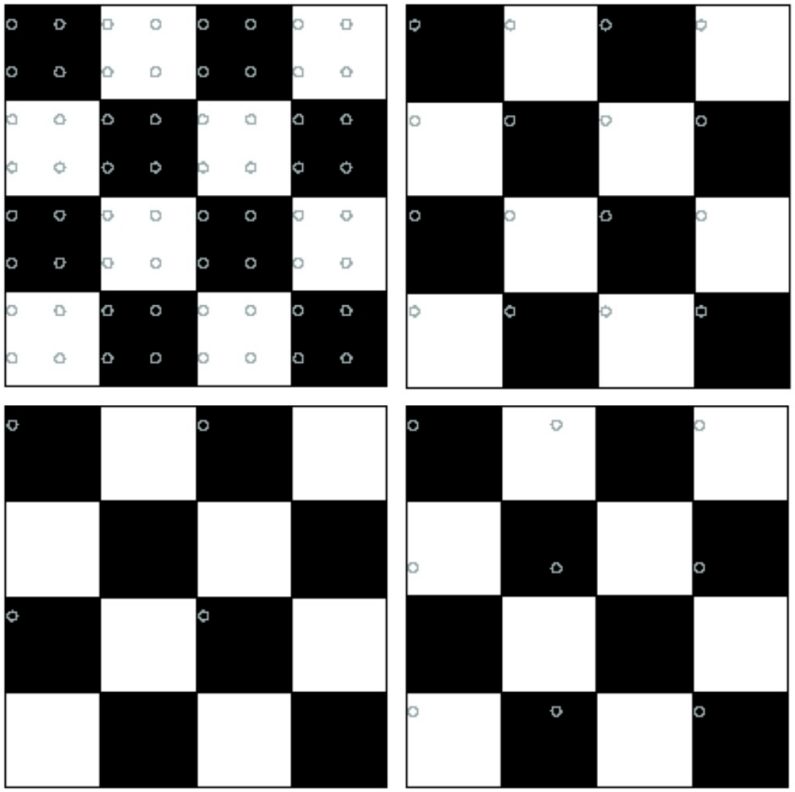
- Occurs when your sampling rate is not high enough to capture the amount of detail in your image



- To do sampling right, need to understand the structure of your signal/image
- The minimum sampling rate is called the **Nyquist rate**

[Source: R. Urtasun]

More on aliasing



Good sampling

Bad sampling

[Source: N. Snavely]