



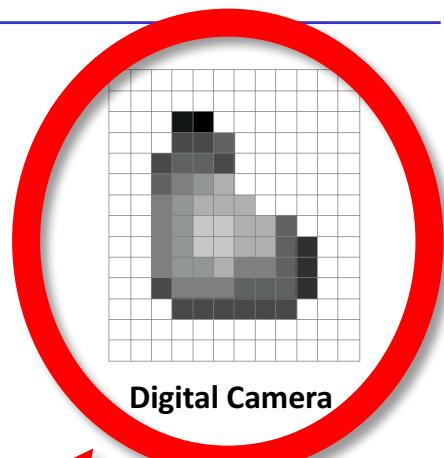
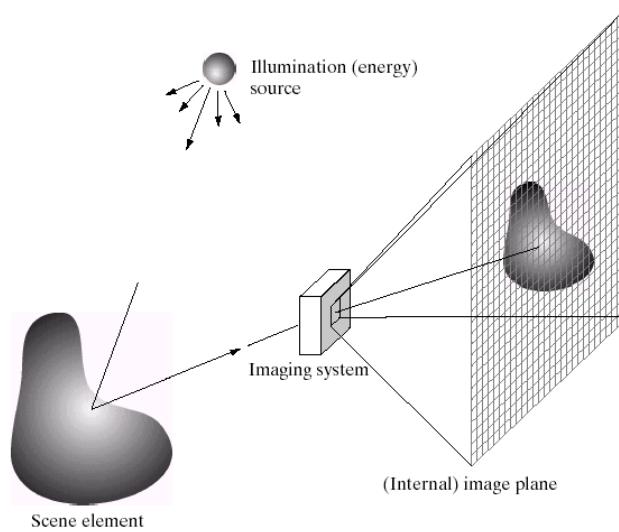
## Image processing

jorge s. marques, josé santos-victor, 2017



images

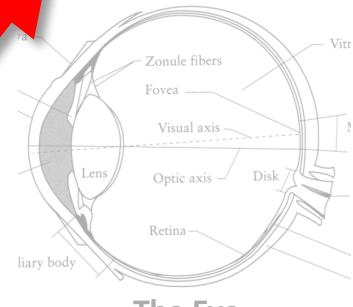
# what is an image?



We'll focus on these in this class

(More on this process later)

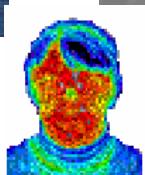
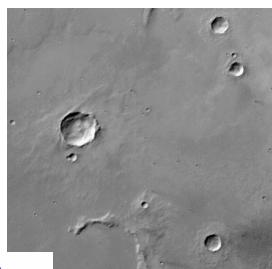
jorge s. marques, josé santos-victor, 2017



The Eye

Source: A. Efros<sup>3</sup>

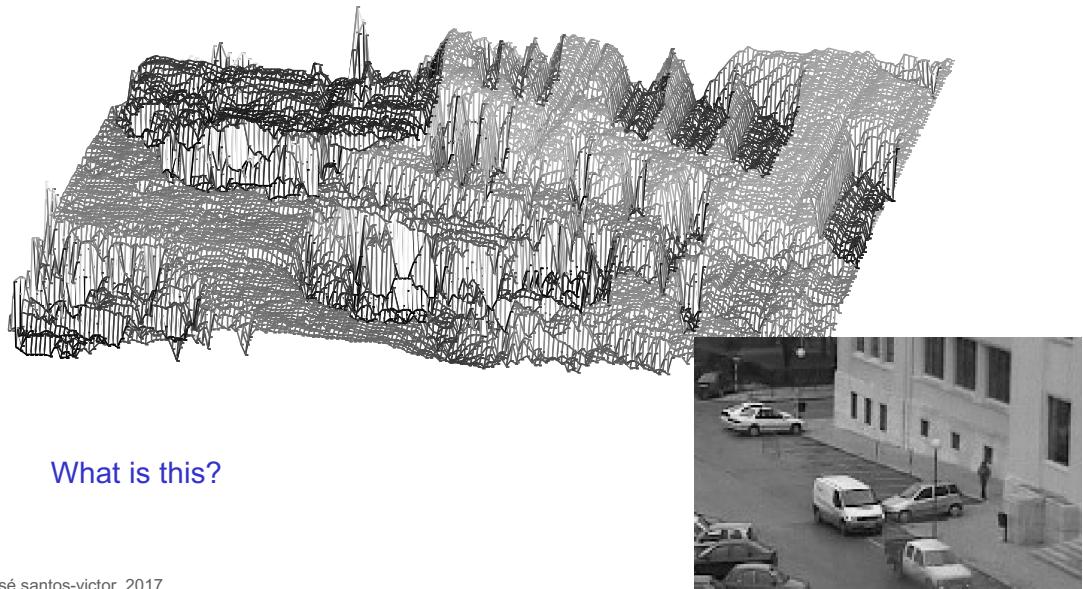
## images: what are they?



jorge s. marques, josé santos-victor, 2017

## what is shown in this image?

---



What is this?

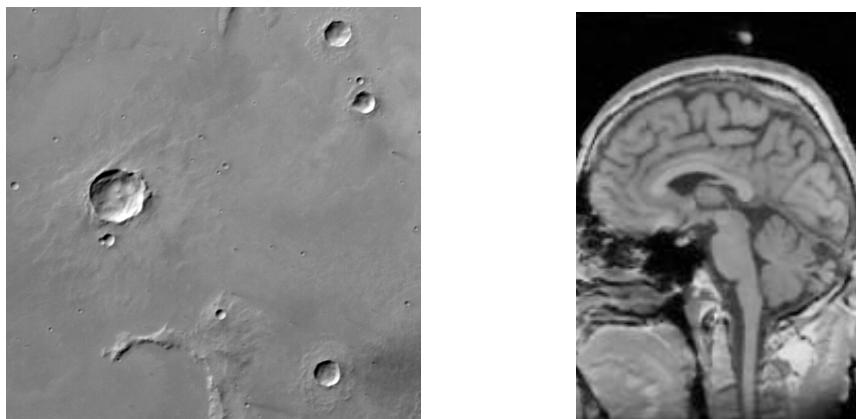
jorge s. marques, josé santos-victor, 2017

5

## what is an image

---

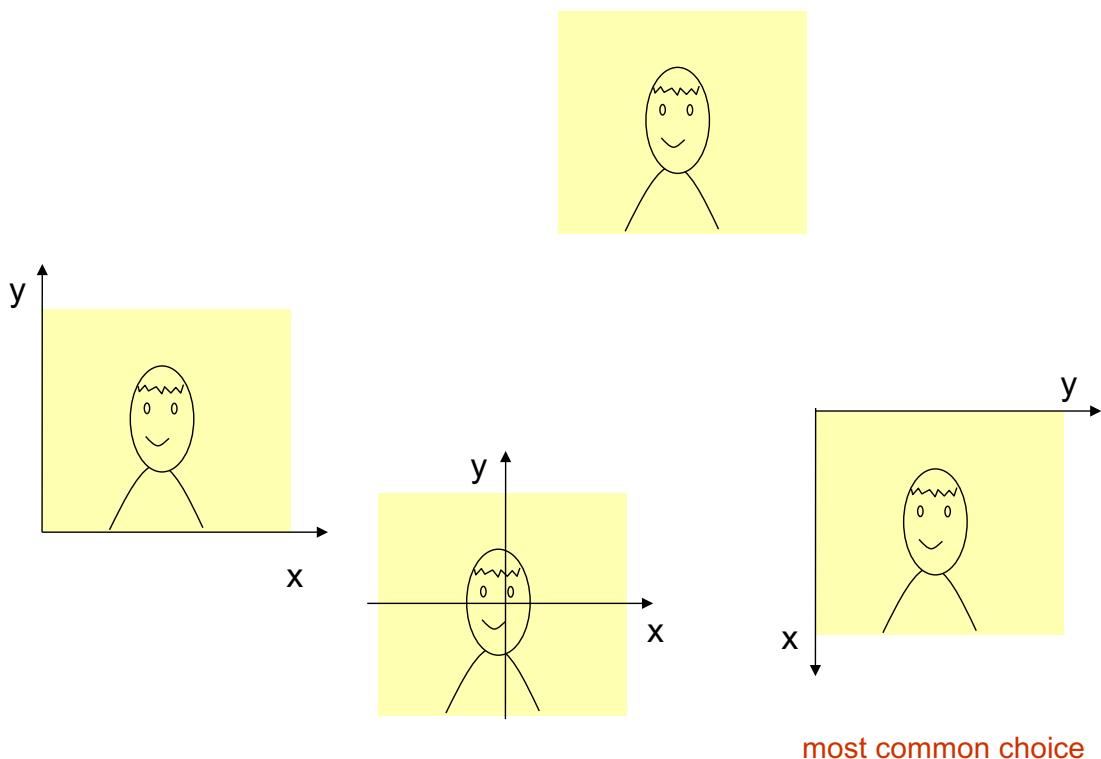
images describe the **evolution of physical variables** (intensity, color, reflectance, conductivity) in a plane or in a 3D volume.



jorge s. marques, josé santos-victor, 2017

6

## coordinate systems

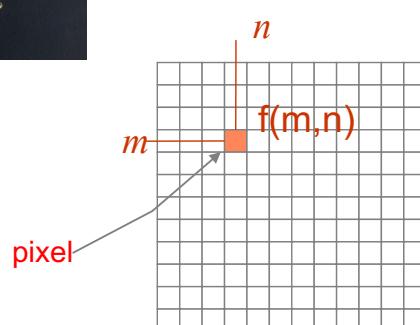
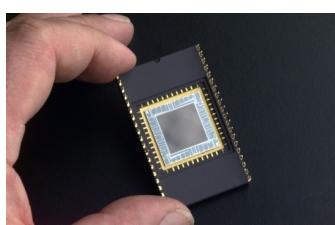


jorge s. marques, josé santos-victor, 2017

7

## discrete images

Most electronic sensors produce discrete images defined on a grid of points



42	45	49	57	65	54	66	63	42	64	95
73	72	68	67	69	75	59	62	55	35	37
82	83	83	84	85	81	79	90	77	76	73
96	95	95	92	90	93	88	79	95	88	93
93	91	93	95	95	96	97	104	94	96	104
110	107	102	104	110	107	103	102	110	97	106
109	112	113	109	105	107	101	108	116	115	111
109	107	109	112	113	101	119	128	143	121	122
114	114	115	115	113	119	146	154	115	110	116
125	125	120	118	121	160	140	112	118	116	122
114	110	115	137	161	132	119	119	107	124	120

**pixel – picture element**

each pixel is characterized by its position and value

jorge s. marques, josé santos-victor, 2017

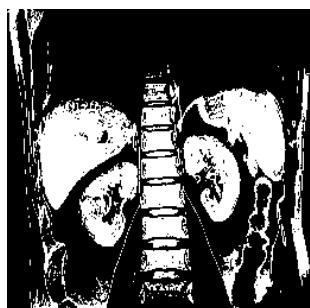
8

# image types

gray scale images



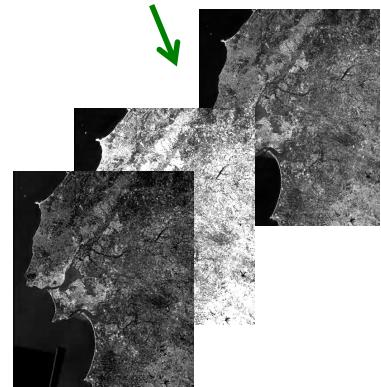
binary images



multi-spectral images

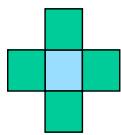


color images



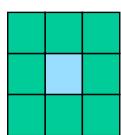
# topological concepts

what is the neighborhood of a pixel  $(i,j)$  ?



neighborhood 4

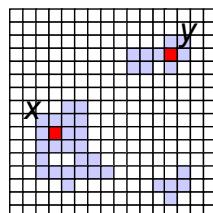
$$N_4(i,j) = \{(i-1,j), (i+1,j), (i,j-1), (i,j+1)\}$$



neighborhood 8

$$N_8(i,j) = \{(i-1,j-1), (i-1,j), (i-1,j+1), (i,j-1), (i,j+1), (i+1,j-1), (i+1,j), (i+1,j+1)\}$$

## connectivity



are x, y connected?

NO

binary image

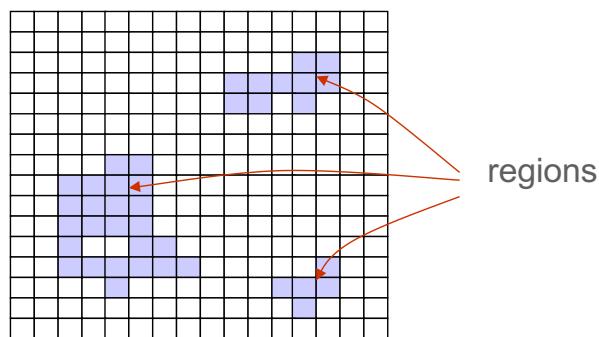
**connectivity:** two pixels  $x, y$  of a binary image  $u(m,n)$  are connected iff (if and only if) there is a sequence of points  $z^1, z^2, \dots, z^n$ :

$$z^1 = x, z^n = y$$

$z^i, z^{i+1}$  are neighbors  $\forall i \in \{1, 2, \dots, n-1\}$

$$u(z^i) = 1 \quad \forall i = 1, \dots, n$$

## regions



**Region**  $R$  is a set of fully connected pixels i.e.,

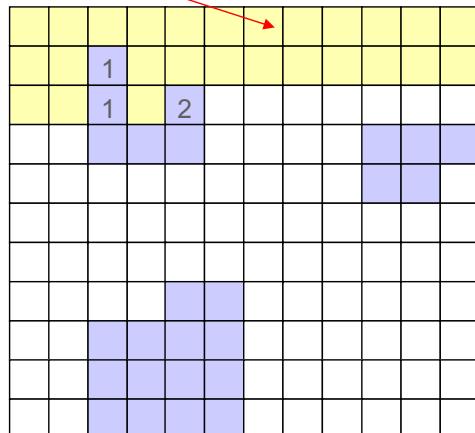
$$\forall x, y \in R, x, y \text{ are connected}$$

A maximal set of connected pixels is called a **connected component**.

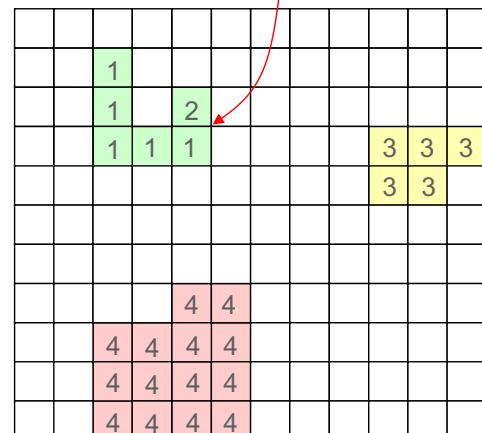
## extraction of connected components

The algorithm assigns different labels to pixels belonging to different connected components

Assign the label of the previous neighbors if they are equal

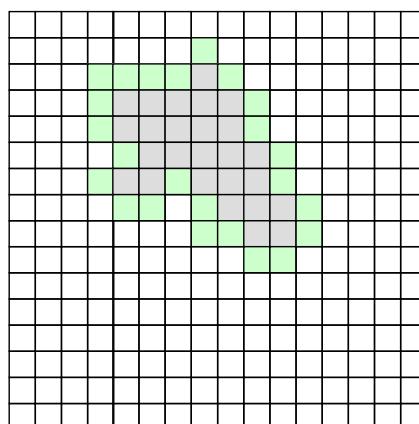


If the neighbors have different labels create an equivalence table  
1↔2



## boundary

The boundary of a region  $R$  is the set of  $R$  pixels whose neighborhood is not contained in  $R$ .



# image filtering

Modify the pixels in an image based on some function of a local neighborhood of each pixel

10	5	3
4	5	1
1	1	7

Local image data

Some function

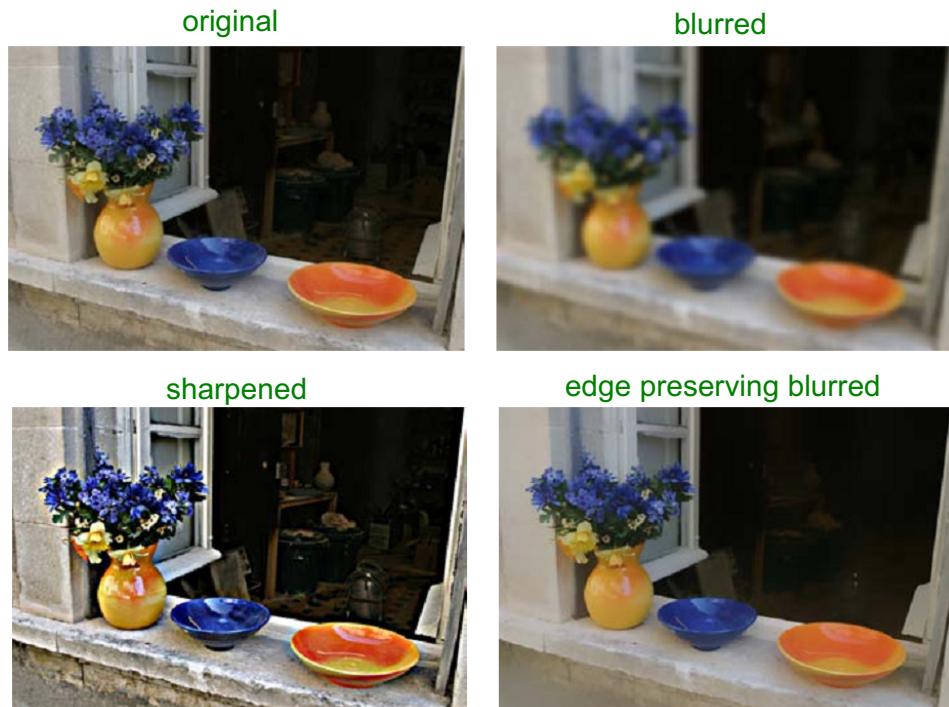
	7	

Modified image data



## linear filtering

## examples



jorge s. marques, josé santos-victor, 2017

17

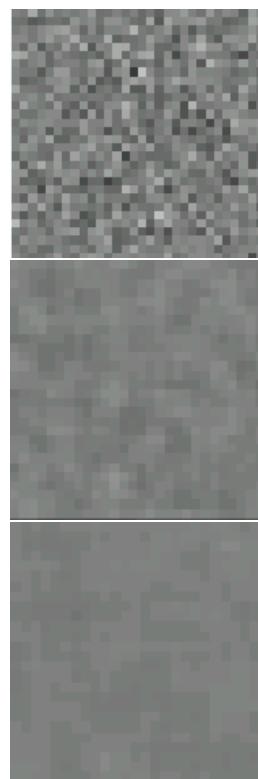
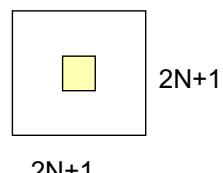
## noise reduction

how can we remove the noise in an (homogeneous) image?

**idea:** replace each pixel by an average

$$g(i,j) = \frac{1}{(2N+1)^2} \sum_{k=i-N}^{i+N} \sum_{l=j-N}^{j+N} f(k,l)$$

**mean filter**



jorge s. marques, josé santos-victor, 2017

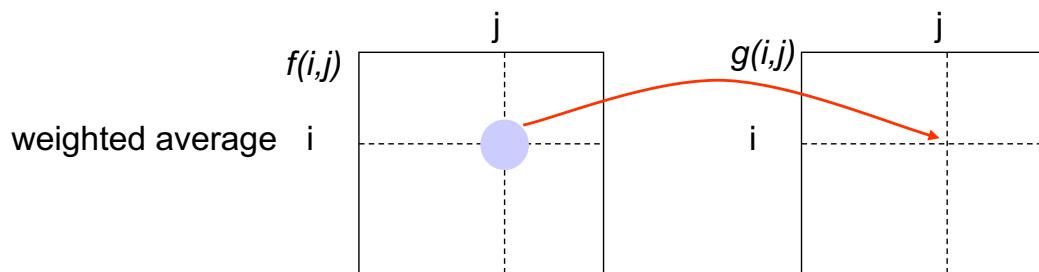
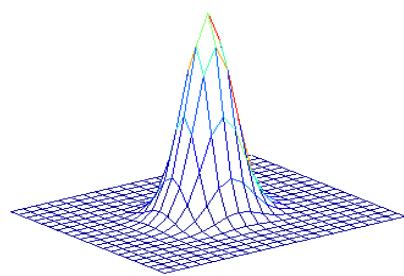
18

## gaussian filter

filtered image

$$g(i,j) = \sum_{k,l} G(k-i, l-j) f(k, l)$$

$$G(k, l) = C e^{-\frac{k^2 + l^2}{2\sigma^2}}$$



jorge s. marques, josé santos-victor, 2017

19

## Gaussian filtering

orig.



$\sigma=0.2$



$\sigma=1$



$\sigma=2$

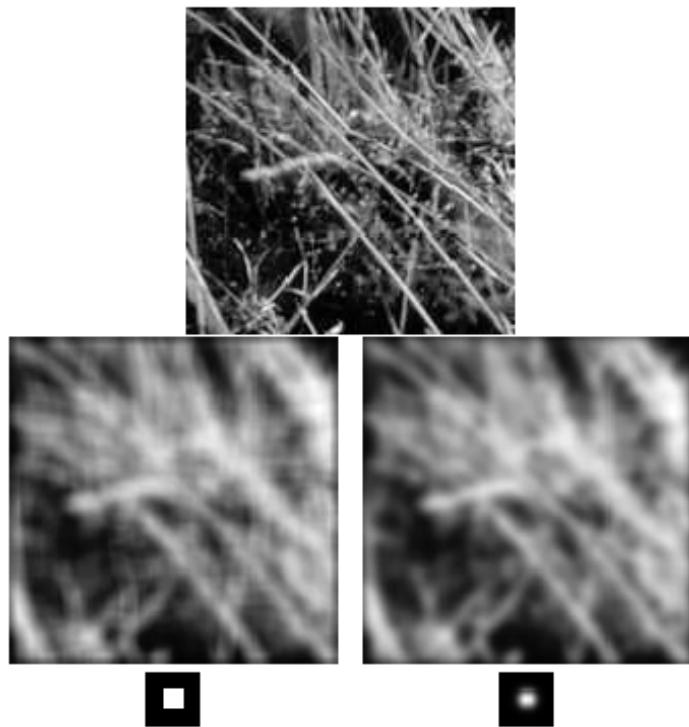


jorge s. marques, josé santos-victor, 2017

the gaussian filter destroys the transitions!!

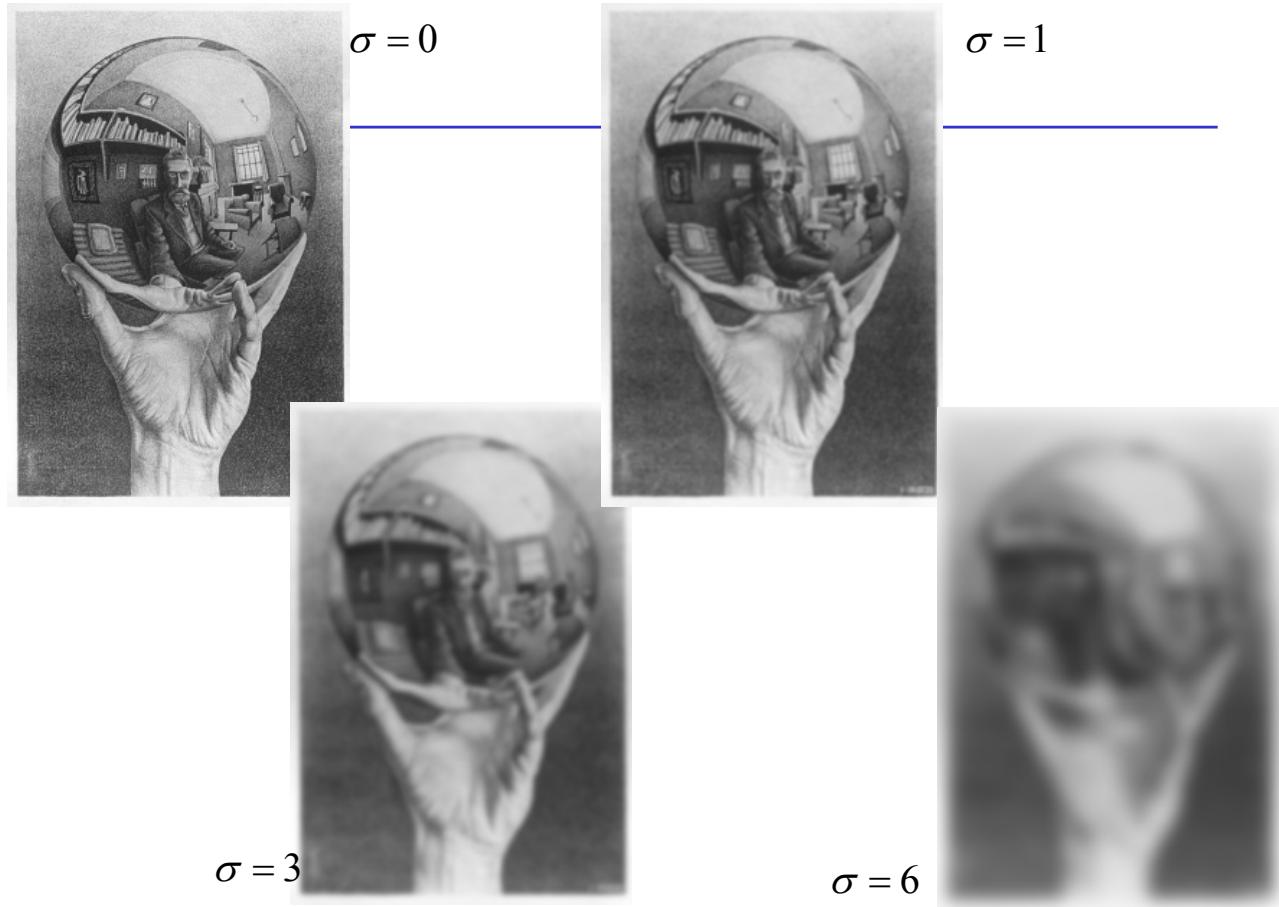
20

## Mean vs. Gaussian filtering



jorge s. marques, josé santos-victor, 2017

21



jorge s. marques, josé santos-victor, 2017

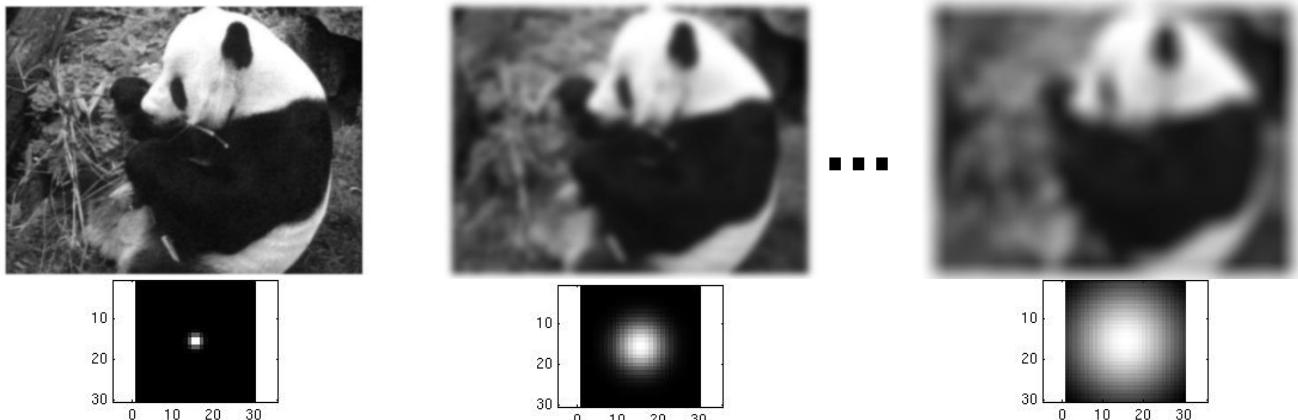
22

## smoothing with a Gaussian

Recall:

parameter  $\sigma$  is the “scale” / “width” / “spread” of the Gaussian kernel, and controls the amount of smoothing.

Kristen Grauman, UT-Austin



## linear filtering

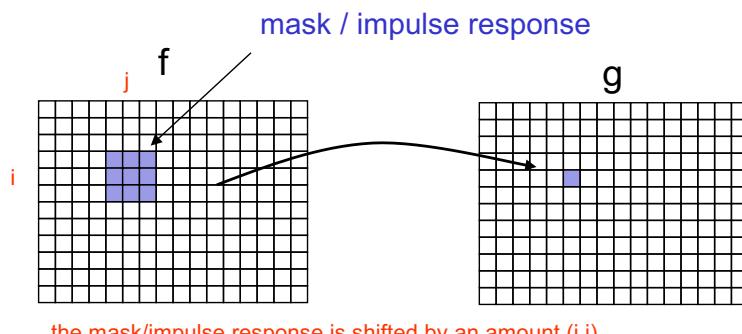
$$g(i,j) = \sum_{k,l} M(k-i, l-j) f(k, l) = M(i, j) \otimes f(i, j)$$

2D correlation

$$g(i,j) = \sum_{k,l} h(i-k, j-l) f(k, l) = h(i, j) * f(i, j)$$

2D convolution sum

$M(i, j)$  mask  
 $h(i, j) = M(-i, -j)$  impulse response



## separable filters

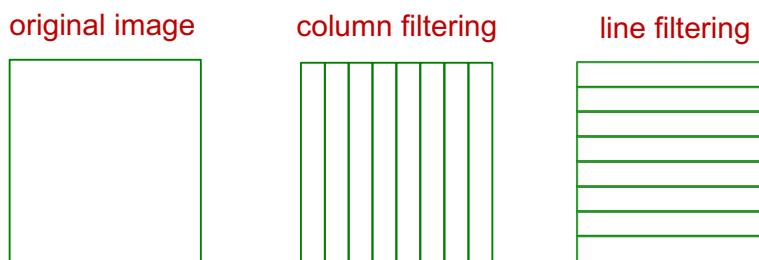
Separable filters have the following property:

$$M(i, j) = M_1(i) M_2(j), \quad h(i, j) = h_1(i) h_2(j)$$

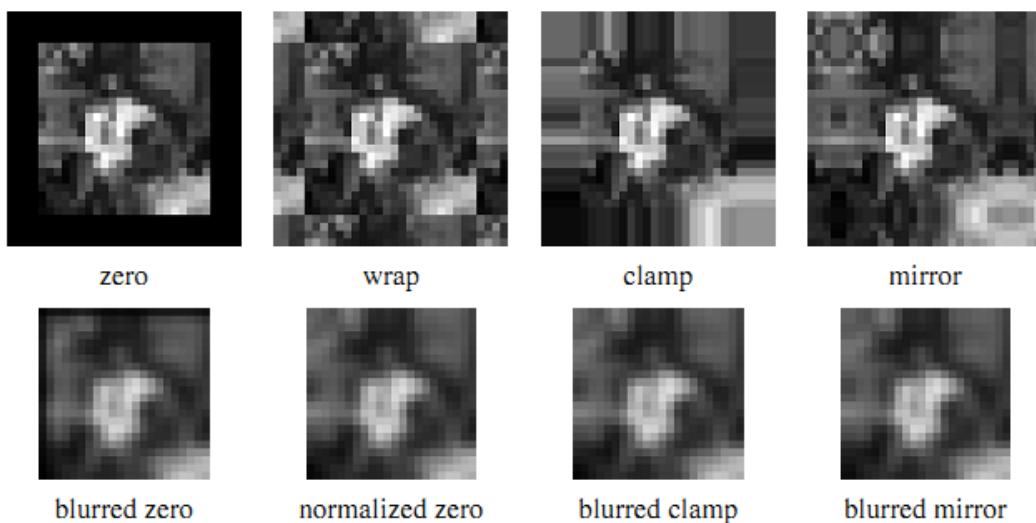
Separable filters can be implemented in a very efficient way.

### Algorithm:

1. filter all the image columns using filter  $h_1$  (1D)
2. filter all the lines of the previous image using filter  $h_2$  (1D)



## border effects



## matrix notation

---

Linear filtering can be written in matrix notation.

$$g = Hf \quad g, f \in IR^p, \quad H \in IR^{p \times p}$$

**g**, **f** are column vectors with all the pixel intensities;  
**H** is a (huge) square matrix;  
**p** is the number of pixels in the image.

This notation is **very elegant** but image filtering cannot be implemented in this way.

## examples

---

$$M(i,j) = \begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}$$

weighted average of  
neighboring pixels

$$M(i,j) = \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$$

vertical difference,  
horizontal mean

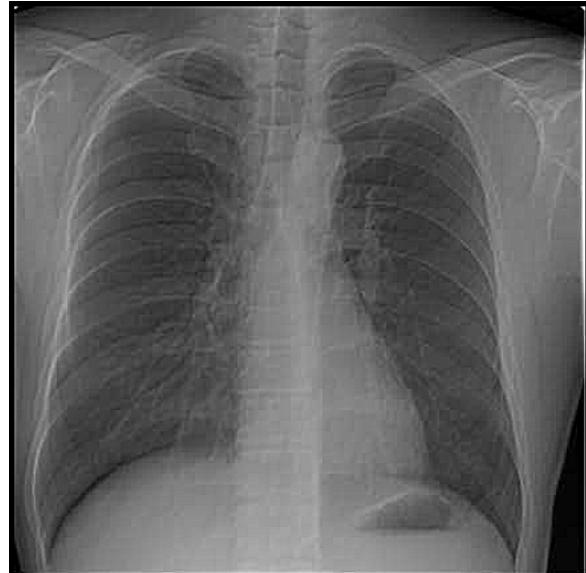
## example

$$J(m,n) = h(m,n) * I(m,n)$$

$$h(m,n) = \delta(m,n) - G(m,n)$$



original



filtered

## 1<sup>st</sup> order derivatives: gradient vector

gradient vector:

$$\nabla g(x) = \begin{bmatrix} g_1(x) \\ g_2(x) \end{bmatrix} = \begin{bmatrix} \frac{\partial f(x)}{\partial x} \\ \frac{\partial f(x)}{\partial y} \end{bmatrix}$$

$f(x)$  continuous image

$x = (x_1, x_2)$  – continuous variables

length and direction

$$\|g(x)\| = \sqrt{g_1^2(x) + g_2^2(x)}$$

$$\angle g(x) = \arctan\left(\frac{g_2(x)}{g_1(x)}\right)$$

# gradient discretization

1) Derivatives can be approximated by finite differences

$$g_1(m,n) \approx \frac{I(m+1,n) - I(m-1,n)}{2}$$

$$g_2(m,n) \approx \frac{I(m,n+1) - I(m,n-1)}{2}$$

-1
0
1

-1	0	1
----	---	---

Linear filtering!

2) a better choice: Sobel masks

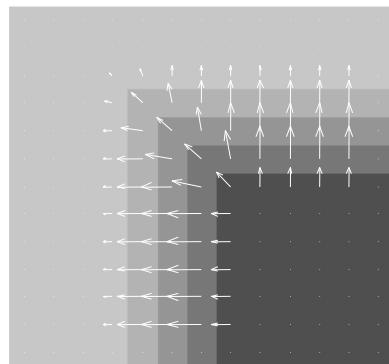
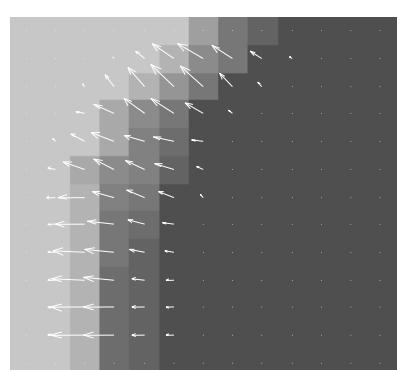
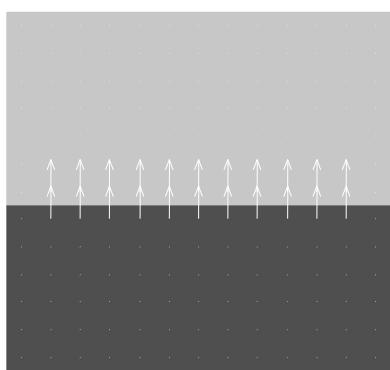
-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

High pass filtering in one direction, and low pass filtering in the orthogonal direction

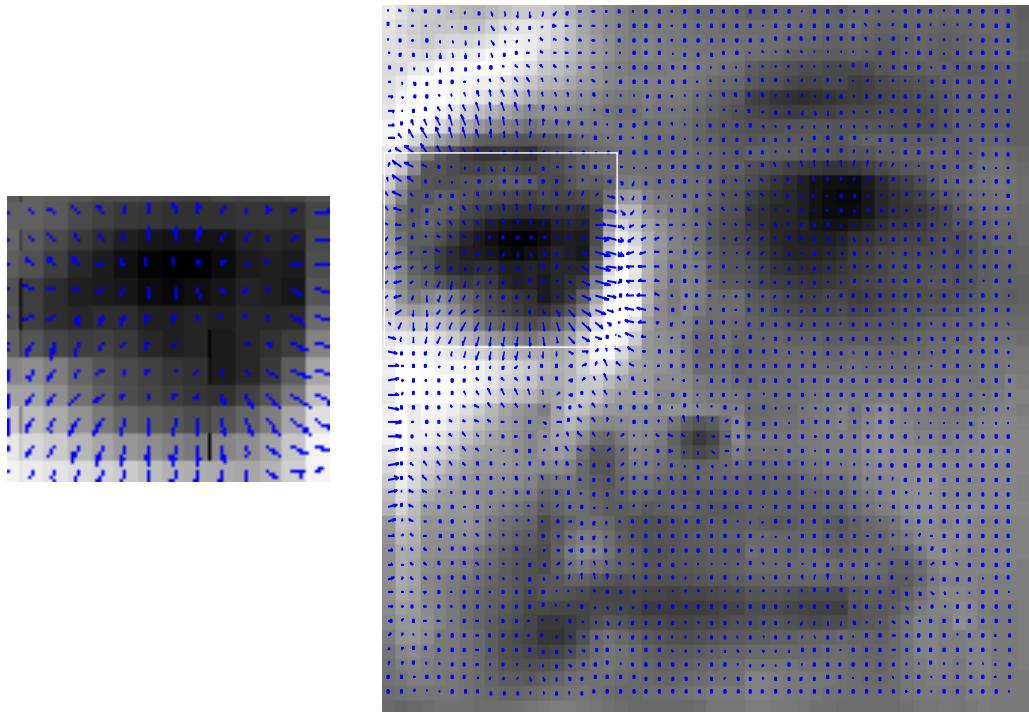
jorge s. marques, josé santos-victor, 2017

## example



jorge s. marques, josé santos-victor, 2017

## example

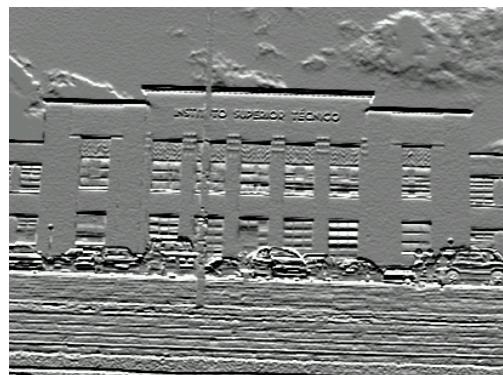


jorge s. marques, josé santos-victor, 2017

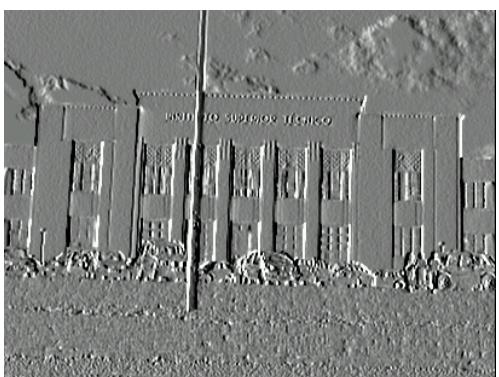
## example: gradient



$g_1$



$g_2$



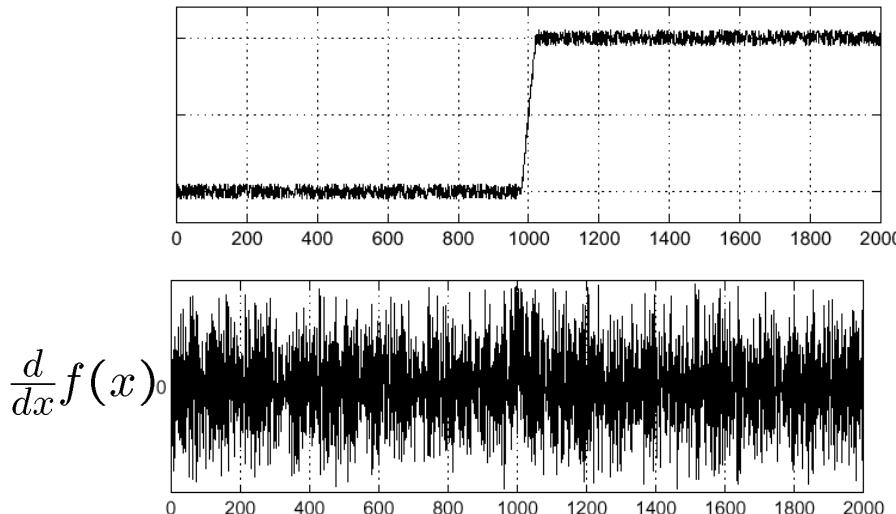
$|g|$



jorge s. marques, josé santos-victor, 2017

## effects of noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal

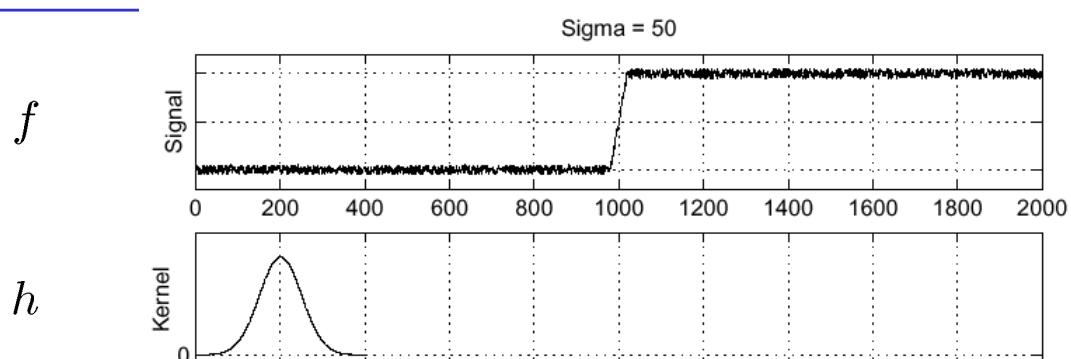


Where is the edge?

jorge s. marques, josé santos-victor, 2017

Source: S. Seitz 35

## solution: smooth first



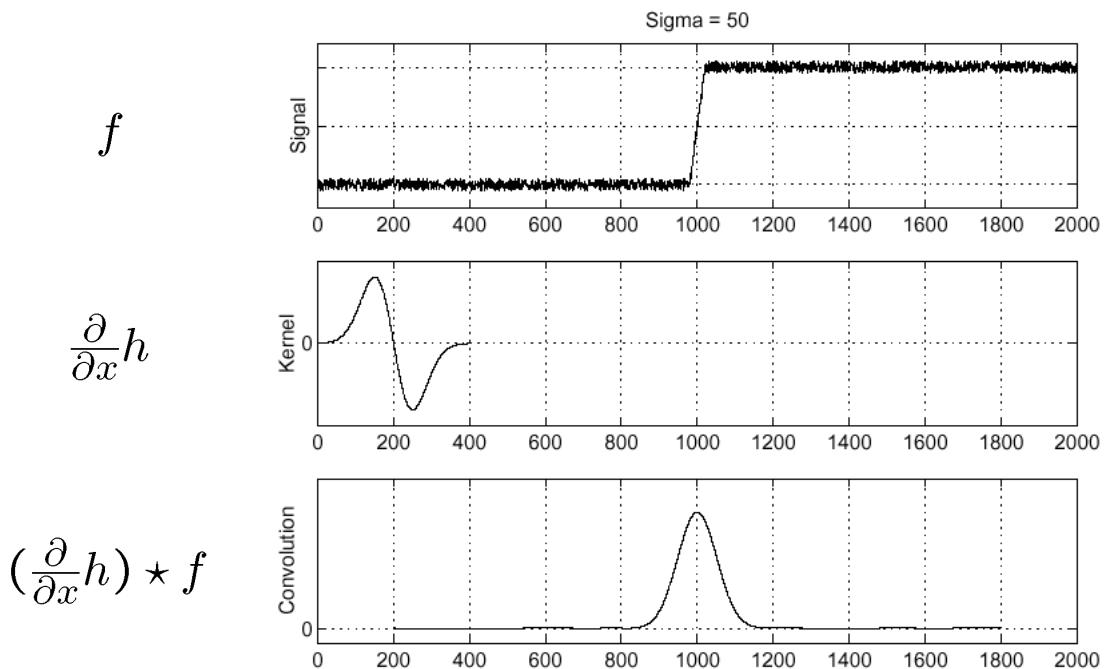
Where is the edge? Look for peaks in  $\frac{\partial}{\partial x}(h * f)$

jorge s. marques, josé santos-victor, 2017

Source: S. Seitz 36

## Derivative theorem of convolution

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

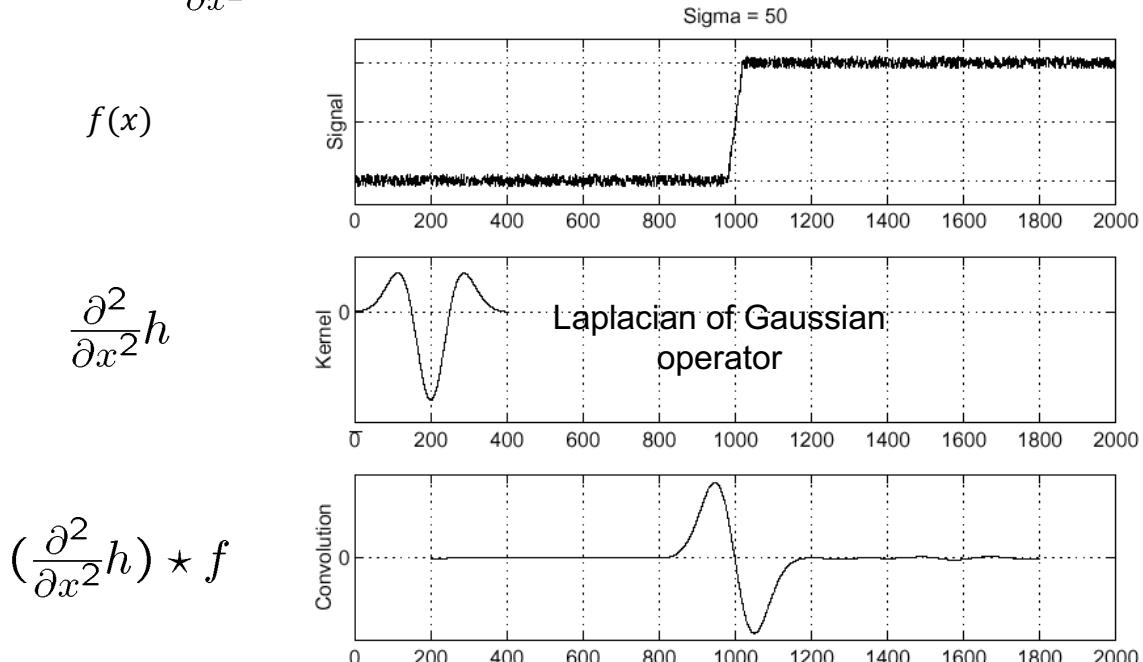


**Slide credit Steve Seitz**  
jorge s. marques, josé santos-victor, 2017

37

## Laplacian of Gaussian

Consider  $\frac{\partial^2}{\partial x^2}(h \star f)$



Where is the edge? Zero-crossings of bottom graph

jorge s. marques, josé santos-victor, 2017

38

## 2nd order derivatives: Laplacian filter

Laplacian of a 2D image

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

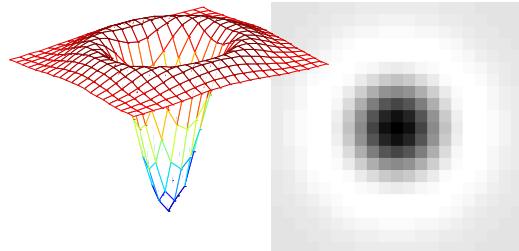
zero crossings are associated to transitions

Since the derivatives increase noise, the image is first filtered using a Gaussian filter

$$\nabla^2[G(x, y) * f(x, y)] = [\nabla^2 G(x, y)] * f(x, y)$$

where

$$\nabla^2 G(x, y) = \left( \frac{x^2 + y^2}{\sigma^4} - \frac{2}{\sigma^2} \right) G(x, y)$$



jorge s. marques, josé santos-victor, 2017

## Directional filters: Gabor filters

Some cells of the visual cortex are activated by directional stimuli. Gabor filters approximate the behavior of these cells.

Their impulse response is the multiplication of a sinusoid by a Gaussian envelope in a rotated 2D space.

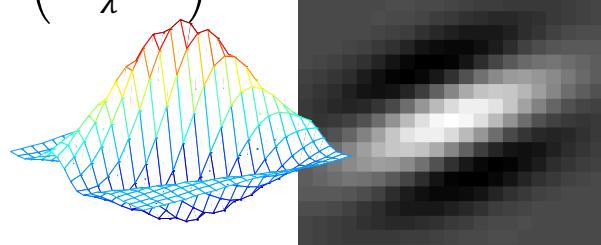
$$h_1(x, y) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \cos\left(2\pi \frac{x'}{\lambda} + \psi\right)$$

$$h_2(x, y) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \sin\left(2\pi \frac{x'}{\lambda} + \psi\right)$$

rotation

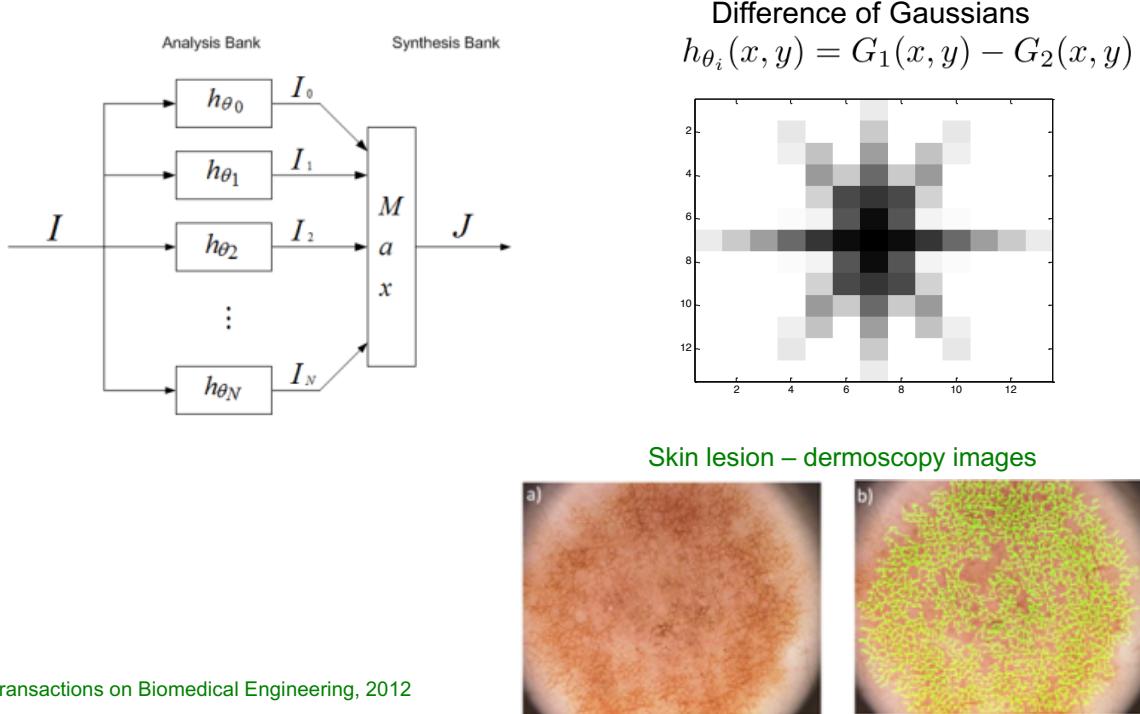
$$x' = x \cos \theta + y \sin \theta$$

$$y' = -x \sin \theta + y \cos \theta$$

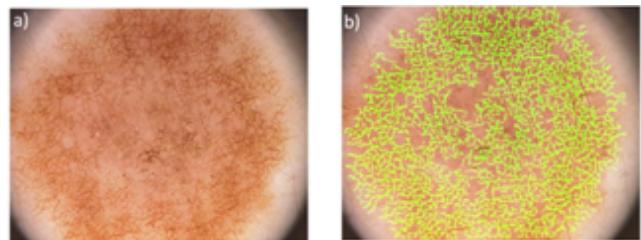


jorge s. marques, josé santos-victor, 2017

# Directional analysis: difference of Gaussians



Skin lesion – dermoscopy images



Barata, Transactions on Biomedical Engineering, 2012

jorge s. marques, josé santos-victor, 2017

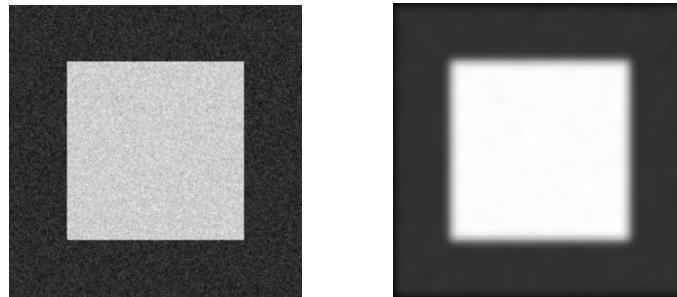
41



nonlinear filtering

jorge s. marques, josé santos-victor, 2017

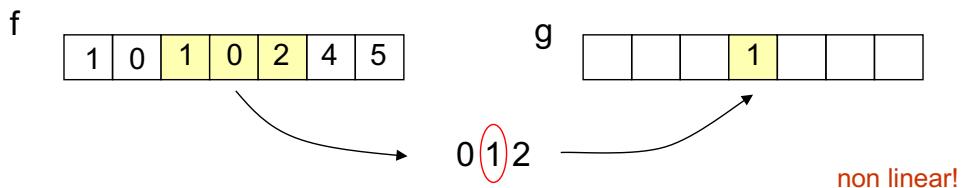
linear filters reduce noise but they destroy image contours.



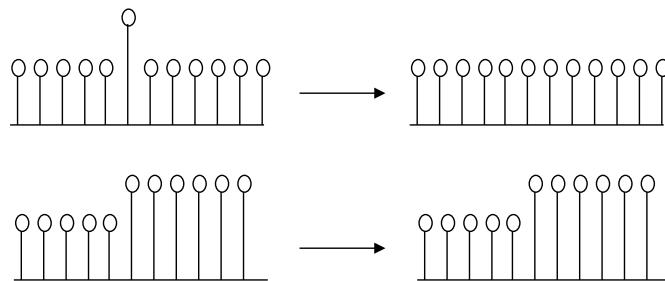
Is it possible to solve this conflict i.e., removing the noise,  
preserving the contours?

median filter ...

## 1D median filter



example

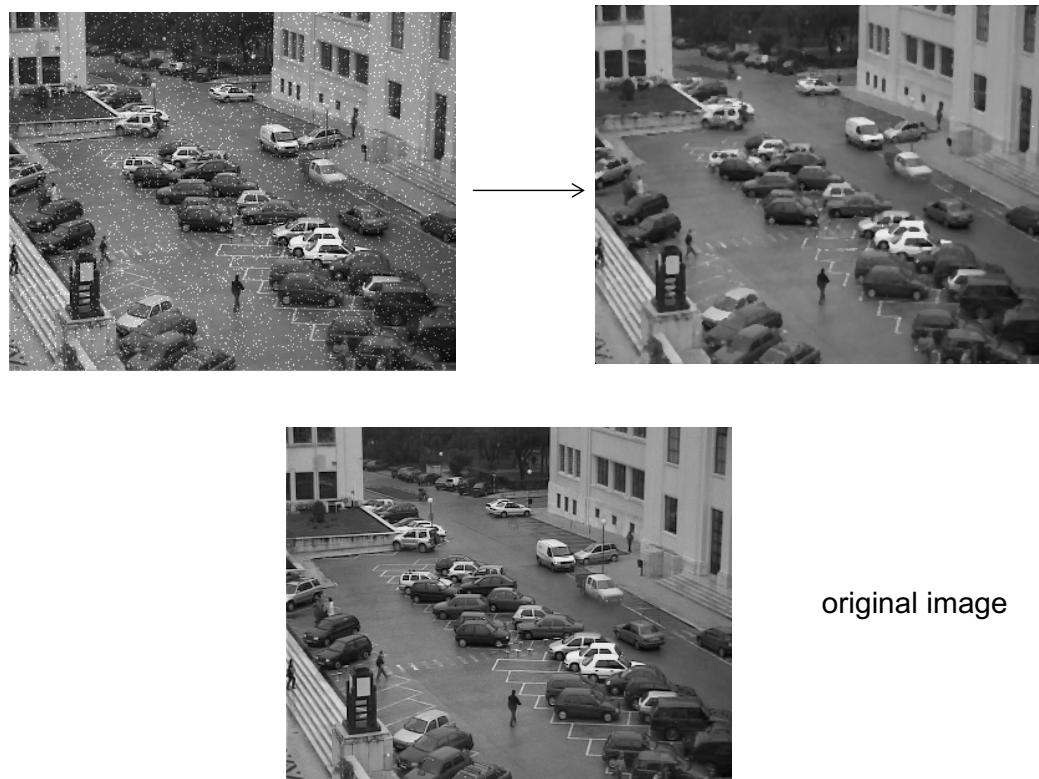


## 2d median filter



$$g(m, n) = \text{median}\{f(m - M, n - N), \dots, f(m + M, n + N)\}$$

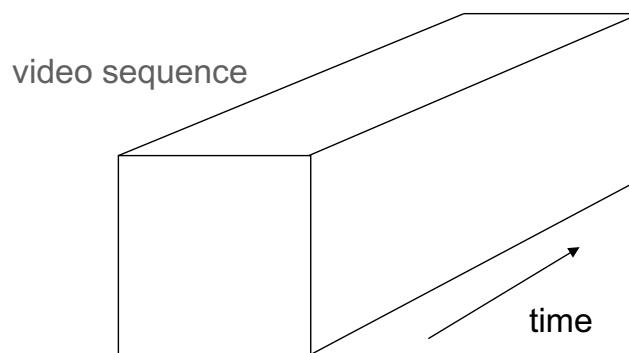
## example



jorge s. marques, josé santos-victor, 2017

47

## sequence filtering



median filter can be applied in **space**, in **time** or in **space-time**.

jorge s. marques, josé santos-victor, 2017

48

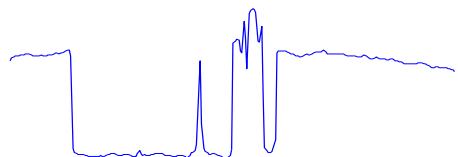
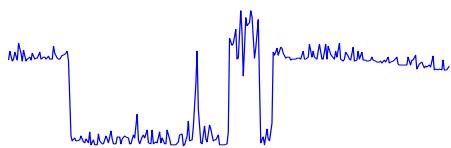
## example – time filtering



output: background image

## non-linear diffusion

non linear diffusion



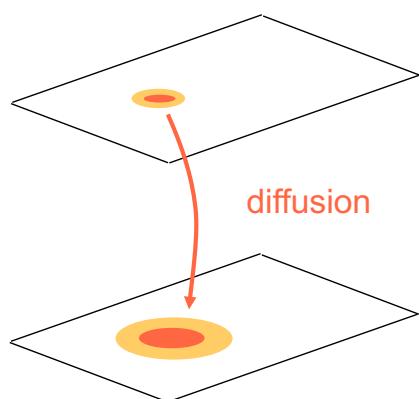
Perona & Malik

## heat equation

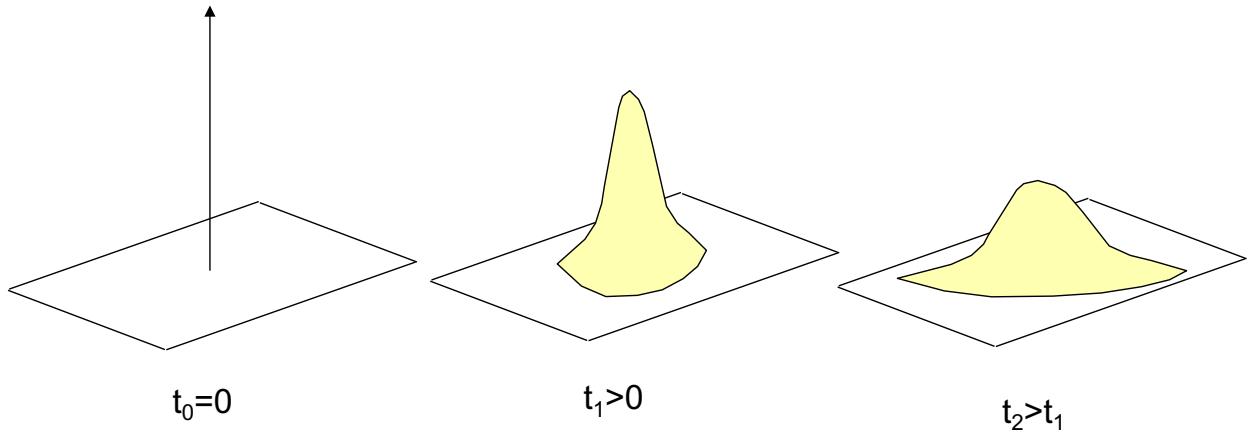
$$\frac{\partial u}{\partial t} = c \nabla^2 u$$

$u(x, t)$  temperature

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



## linear diffusion



Solution of heat equation:

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

$$\sigma = \sqrt{2t}$$

(check it!)

## linear diffusion

definition:  $\frac{\partial g}{\partial t} = \nabla^2 g$

$g(x, 0) = f(x)$ , initial condition

The equation has a unique solution

$$g(x, t) = \begin{cases} f(x) & t = 0 \\ (G_{\sqrt{2t}} * f)(x) & t > 0 \end{cases}$$

Gaussian filtering: beautiful!!

$$(G_\sigma * I)(x) = \int_{\mathbb{R}^2} G_\sigma(x - y) f(y) dy$$
$$G_\sigma(x) = \frac{1}{2\pi\sigma^2} e^{-\frac{\|x\|^2}{2\sigma^2}}$$

Linear diffusion leads to Gaussian filtering with parameter:  $\sigma = \sqrt{2t}$

## gaussian filtering



jorge s. marques, josé santos-victor, 2017

55

## Space-scale

Many images contain structures with different sizes. Therefore it is useful to represent it as a family of images with increasingly less details.

**Space scale** of an image  $f(x)$  is a family of images indexed by a parameter  $s$ , denoted as:

$$\{T_s f, s \geq 0\}$$

such that

$$\begin{aligned} T_0 f &= f \\ T_{s+t} f &= T_s(T_t f) \end{aligned}$$

semi-group property

## Gaussian space-scale

The Gaussian space-scale is obtained by filtering the image using a Gaussian kernel and changing the parameter  $s$  from 0 to infinity.

$$T_s f = G_s * f$$

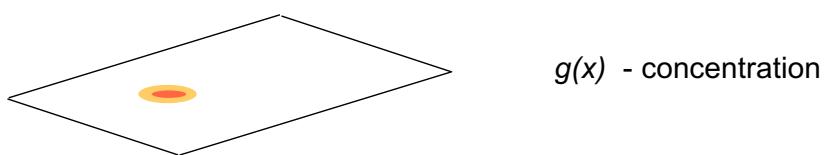
The semi-group property holds. Prove it !

Limitations:

It destroys the contour information: **how can we overcome this difficulty?**

Can be generated with a linear diffusion process

## Generalization



mass flows from regions of high concentration to regions with smaller concentration taking the gradient opposite direction.

Fick's law

$$j = -c \nabla g$$

Mass conservation law

$$\frac{\partial g}{\partial t} = -\operatorname{div} j$$

$j(x,t)$  – diffusion flux  
 $c(x,t)$  – diffusion coefficient  
 $g(x,t)$  – concentration

homogeneous material: constant diffusivity.

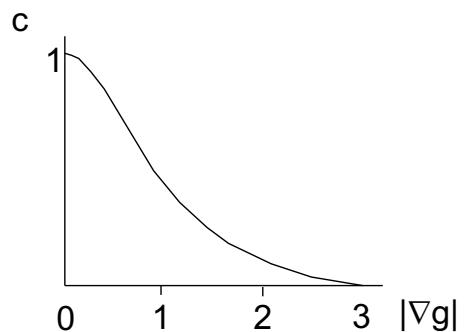
Perona & Malik: space dependent diffusivity

Idea: filter homogeneous regions and not transitions

## conductivity

conductivity (Perona & Malik)

$$c(x,t) = \frac{1}{1 + \alpha \|\nabla g(x,t)\|^2}$$



Conductivity depends on the concentration at a given position and time instant and changes along the time.

In **homogeneous regions** ( $|\nabla g|=0$ ) and conductivity is constant  $c=1$ .  
At the **contour points**, conductivity is almost zero.

## non linear diffusion

Diffusion equation (Perona & Malik)

$$\frac{\partial g}{\partial t} = \operatorname{div}(c(x,t)\nabla g)$$

$$c(x,t) = \frac{1}{1 + \alpha \|\nabla g(x,t)\|^2}$$

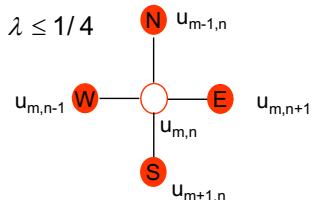
remark:  $\operatorname{div} v = \frac{\partial A}{\partial x} + \frac{\partial B}{\partial y}$        $v = (A, B)$

## discretization

Perona & Malik algorithm is based on a discretization of the diffusion equation in which the partial derivatives are replaced by first order differences.

where

$$g_{m,n}^{t+1} = g_{m,n}^t + \lambda [c_N \nabla_N g + c_S \nabla_S g + c_E \nabla_E g + c_W \nabla_W g]_{m,n}^t \quad 0 \leq \lambda \leq 1/4$$



$$\nabla_N u_{m,n} = u_{m-1,n} - u_{m,n}$$

$$\nabla_S u_{m,n} = u_{m+1,n} - u_{m,n}$$

$$\nabla_E g_{m,n} = g_{m,n+1} - g_{m,n}$$

$$\nabla_S g_{m,n} = g_{m,n-1} - g_{m,n}$$

$$c_{N m,n} = g(\|\nabla g\|_{m+1/2,n}) f_{m,n}$$

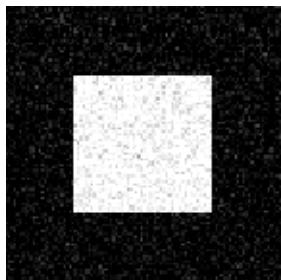
$$c_{S m,n} = g(\|\nabla g\|_{m-1/2,n})$$

$$c_{E m,n} = g(\|\nabla g\|_{m,n-1/2}) f_{m,n}$$

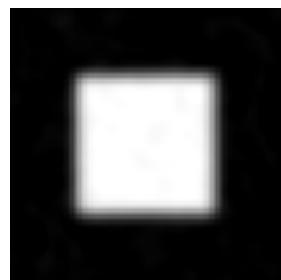
$$c_{W m,n} = g(\|\nabla g\|_{m,n+1/2})$$

## example

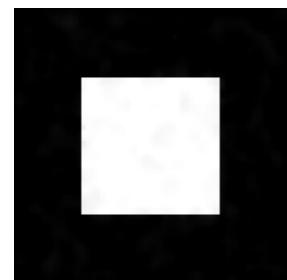
original



Gaussian filter



Perona-Malik

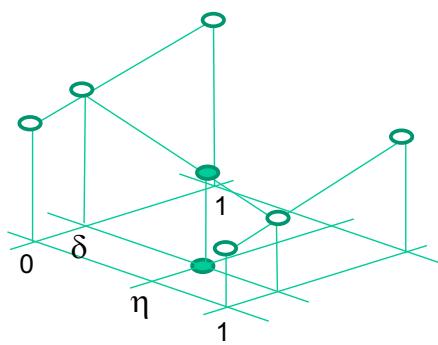




## interpolation and pyramids

jorge s. marques, josé santos-victor, 2017

## bilinear interpolation



Data:  $f(x, y) \quad x, y \in \{0,1\}$

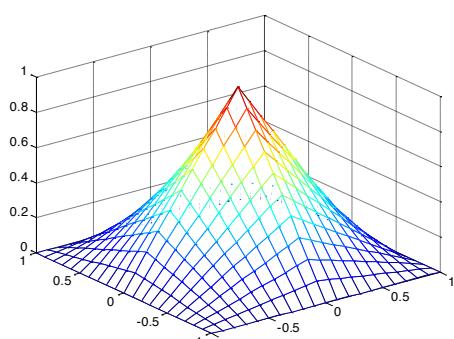
$$f(0, \delta) = (1 - \delta)f(0,0) + \delta f(0,1)$$

$$f(1, \delta) = (1 - \delta)f(1,0) + \delta f(1,1)$$

$$f(\eta, \delta) = (1 - \eta)f(0, \delta) + \eta f(1, \delta)$$

Replacing

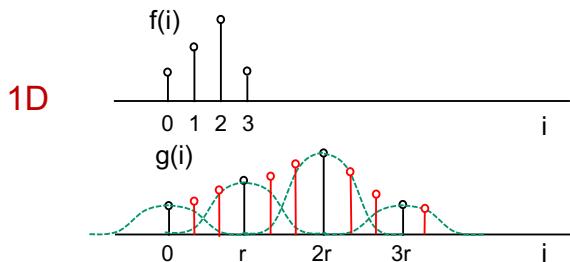
$$f(\eta, \delta) = (1 - \eta)(1 - \delta)f(0,0) + (1 - \eta)\delta f(0,1) \\ + \eta(1 - \delta)f(1,0) + \eta\delta f(1,1)$$



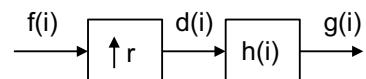
jorge s. marques, josé santos-victor, 2017

# interpolation at equally spaced points

Sometimes we wish to increase the resolution of images. This can be done by interpolation.



$$g(i) = \sum_k f(k)h(i - kr)$$



$$d(i) = \begin{cases} f\left(\frac{i}{r}\right) & i = kr \\ 0 & \text{otherwise} \end{cases}$$

$$g(i) = d(i) * h(i) = \sum_k d(k)h(i - kr)$$

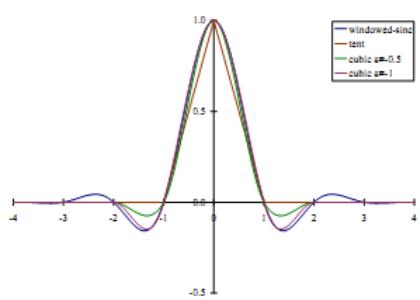
$$= \sum_k f(k)h(i - kr) \quad p = kr$$

2D

$$g(i, j) = \sum_{k,l} f(k, l)h(i - kr, j - lr)$$

## Interpolation results

filters



bilinear



bicubic (a=-1)



bicubic (a=-0.5)

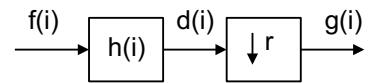
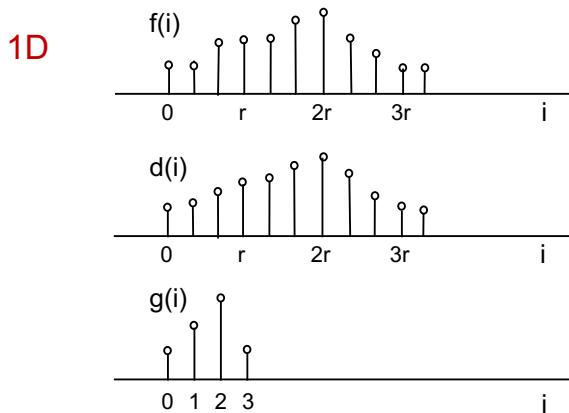


windowed sinc



## decimation

Decimation reduces the image resolution by a factor  $k$ . It involves two steps: 1) lowpass filtering (anti-aliasing) and 2) subsampling.



$$d(i) = \sum_k f(k)h(i - k)$$

$$g(i) = d(ri) = \sum_k f(k)h(ri - k)$$

**2D**

$$g(i, j) = \sum_{k,l} f(k, l)h(ri - k, rj - l)$$

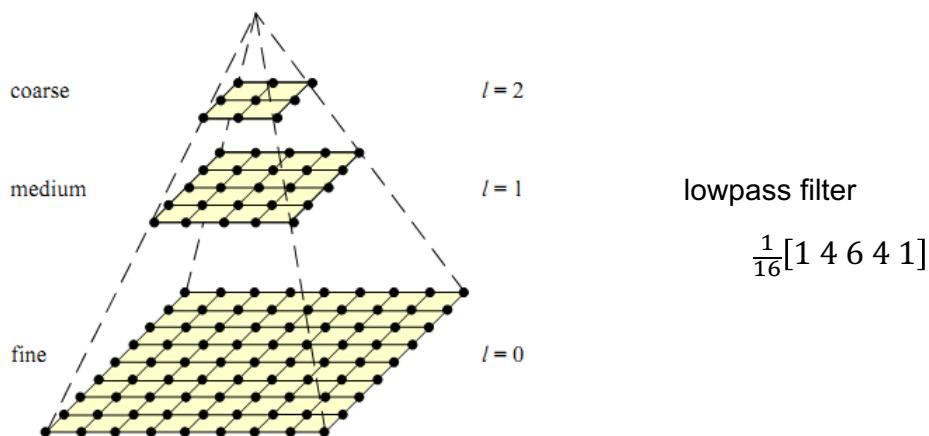
$$g(i, j) = \sum_{k,l} f(k, l)h(i - k/r, j - l/r)$$

jorge s. marques, josé santos-victor, 2017  **$h(k, l)$**  is the decimation kernel

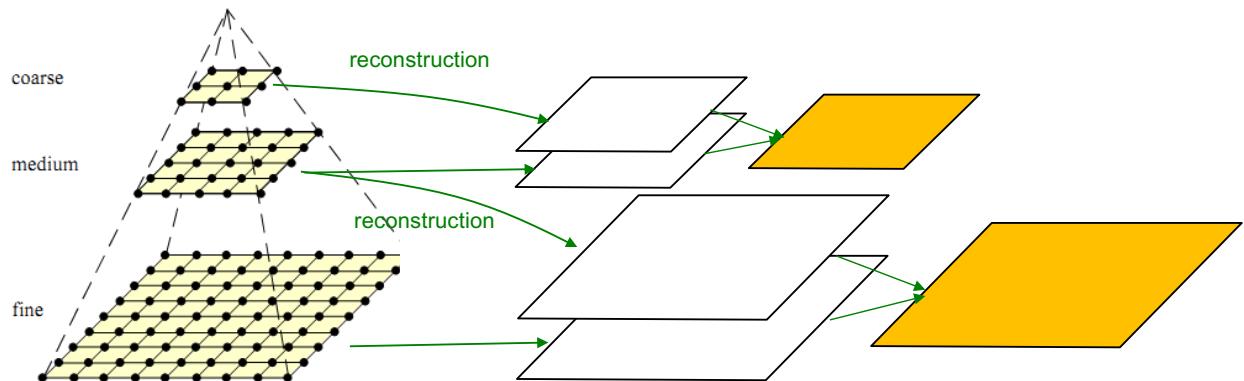
if  $h(k, l)$  is the interpolation kernel

67

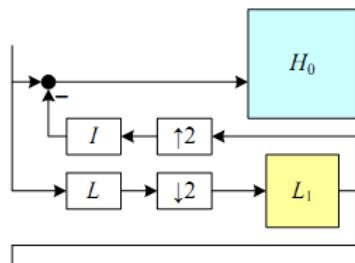
## Gaussian pyramid (Burt & Adelson)



## Laplacian pyramid (Burt & Adelson)



Block diagram



- perfect reconstruction
- inefficient representation

jorge s. marques, josé santos-victor, 2017

69

## Credits

Some slides are based on:

Richard Szeliski, Computer Vision: Algorithms and Application, Springer, 2011