

Processat de la imatge



Menú

- Introducció
- Point operations
 - Transformacions dels nivells de gris
 - Transformacions geomètriques
 - Operacions aritmètiques
- Processat local
 - Suavitzat de la imatge
 - Extracció del gradient
 - Operadors de 2^a derivada
- Detecció de contorns
 - L'operador de Marr-Hildreth
 - L'operador de Canny
- Múltiples escales



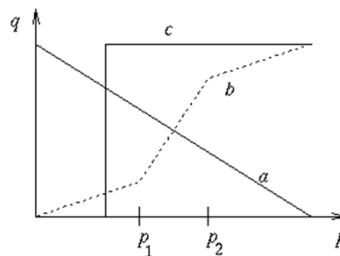
Introducció

- El processat consisteix en una sèrie d'operacions al nivell més baix d'abstracció.
- La entrada i sortida del pre-processat són imatges
- L'objectiu és la millora de la imatge
 - Suprimir les distorsions
 - Realçar les característiques més importants per a processats posteriors



Transformacions del nivell de gris

- Podem aplicar diferents transformacions $q = T(p)$

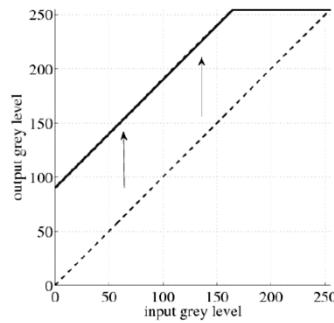


- a) Transformació inversa
- b) Realçat del contrast entre p_1 i p_2
- c) Binaritzat

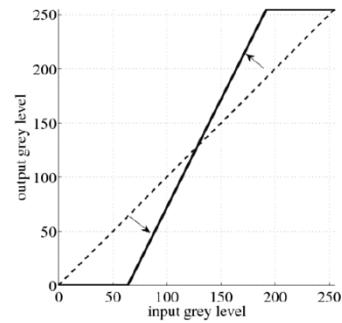
- Es solen implementar usant LUTs (lookup tables)



Common Mapping Functions



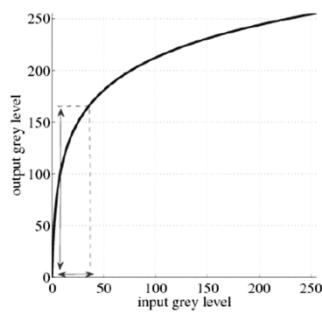
lightening



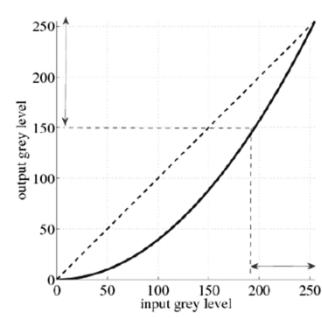
increase contrast



Common Mapping Functions



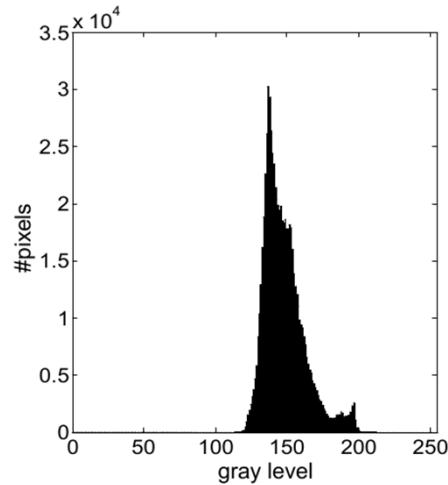
enhance contrast in dark regions



enhance contrast in light regions



Gray level histograms



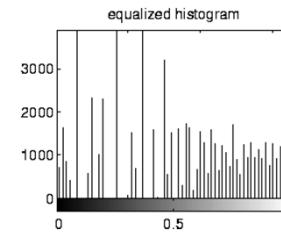
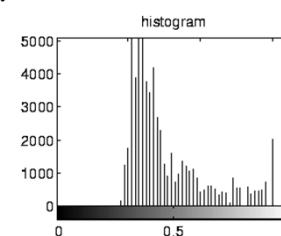
Bay image

- Es poden interpretar com a funcions de densitat de probabilitat



Transformacions del nivell de gris

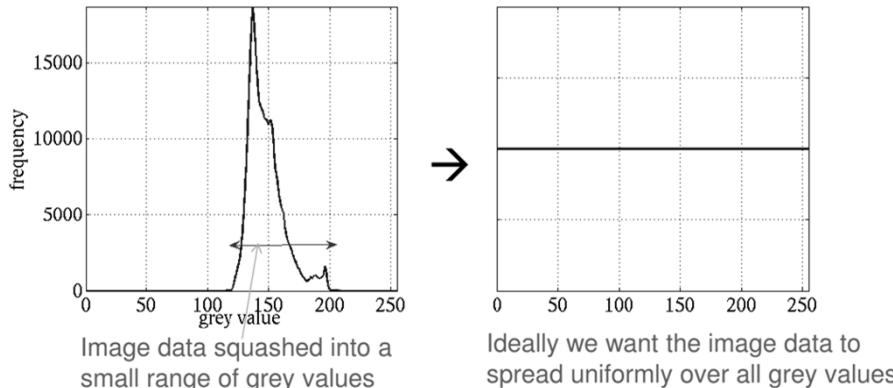
- Transformació típica: equalització de l'histograma
- L'objectiu és crear una imatge amb els nivells de gris distribuïts de forma uniforme.



Transformacions del nivell de gris

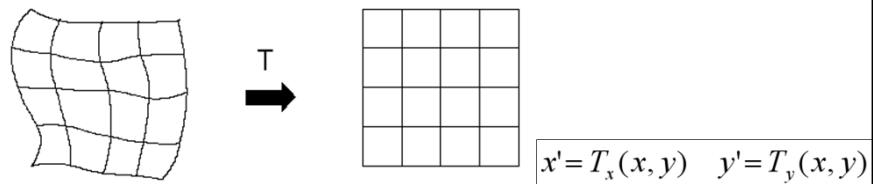
Histogram Equalization

↳ stretch out the histogram to produce a more uniform distribution



Transformacions geomètriques

- Permeten eliminar les distorsions geomètriques produïdes en la captura de la imatge.
- Es mapeja el píxel (x,y) en una nova posició (x',y')



- Rotació: $x' = x \cos \alpha + y \sin \alpha \quad y' = -x \sin \alpha + y \cos \alpha$
- Canvi d'escala: $x' = ax \quad y' = bx$

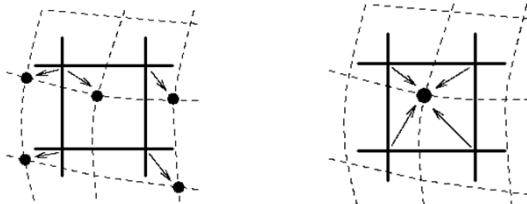


Transformacions geomètriques

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_0 & a_1 \\ b_0 & b_1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} + \begin{bmatrix} a_2 \\ b_2 \\ 0 \end{bmatrix} \Leftrightarrow \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

affine transformation in homogeneous coordinates

- Les noves coordenades no encaixaran en la matriu imatge discreta.
- Cal posar valors en les posicions discretes de la matriu imatge.
- El valor de cada píxel en la matriu es determina per interpolació dels valors dels píxels veïns.
- Una altra possibilitat és assignar-li el valor que ha caigut en la posició més propera.



Operacions aritmètiques

- Operem dues o més imatges per a obtenir una imatge resultat.
- Els píxels veïns no es tenen en compte
- Suma, resta, producte, màxim, mínim, operacions lògiques...
- Possiblement la més popular és la **resta d'imatges**
- Que passa si els valors del resultat es surten del rang?

- Valor absolut
- Rectificat positiu
- Rectificat negatiu
- Escalat lineal



Resta d'imatges



Imatge original

Imatge patró

Valor absolut



Rectificat positiu

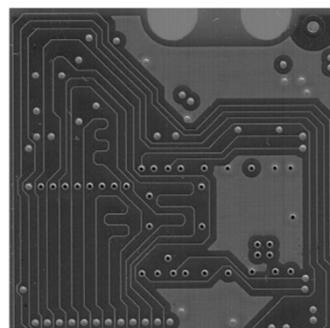
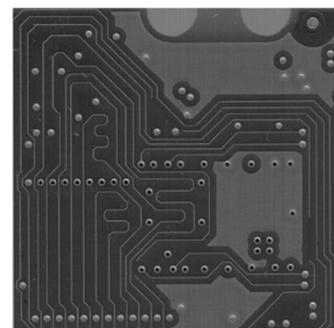
Rectificat negatiu

Re-escalat



Resta d'imatges

Where is the defect?

Image $g[x,y]$ (no defect)Image $f[x,y]$ (w/ defect)

Resta d'imatges

Absolute difference between two images



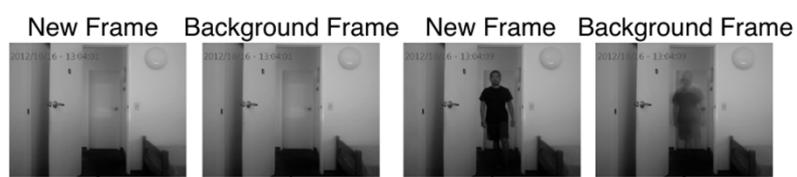
$|f-g|$ w/o alignment



$|f-g|$ w/ alignment



Video background subtraction



$\text{Abs}(\cdot) > \theta ?$

$\text{Abs}(\cdot) > \theta ?$

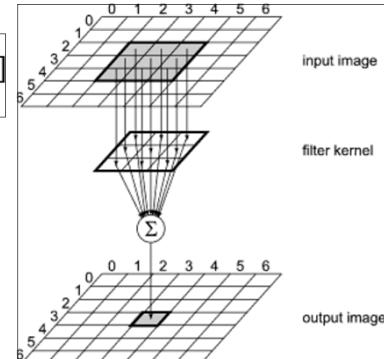
Update:
 $\text{Background}[t] := \alpha \text{ Background}[t-1] + (1 - \alpha) \text{ New}[t]$



Producte de convolució

- Es calcula un nou valor del píxel a partir dels valors dels píxels veïns
- La contribució dels píxels veïns es pondera per uns coeficients h.

$$g[i, j] = f[i, j] \otimes h[i, j] = \sum_{k=1}^n \sum_{l=1}^m h[k, l] f[i-k, j-l]$$



- Aquesta operació rep el nom de producte de convolució
- h és la finestra o màscara de convolució
- 2 grups de pre-processat: suavitzat i realçat

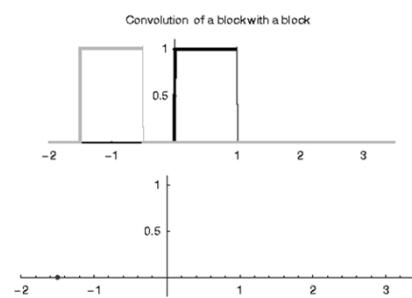


Suavitzat de la imatge

-Filtre Mitjana:

$$g[i, j] = \frac{1}{m \cdot n} \sum_{k=1}^n \sum_{l=1}^m f[k, l]$$

- S'implementa com una convolució



Suavitzat: filtre averaging

- ↳ So far we modified values of single pixels
- ↳ What if we want to take neighborhood into account?
- ↳ A common application of linear filtering is **image smoothing** using an **averaging filter** or averaging **mask** or averaging **kernel**
- ↳ Each point in the smoothed image $g(x, y)$ is obtained from the average pixel value in a neighbourhood of (x, y) in the input image
 - The averaging filter is also known as the **box filter**.
Each pixel under the mask is multiplied by 1/9, summed, and the result is placed in the output image
 - The mask is successively moved across the image.
That is, we **convolve** the image with the mask.

$$\frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$



Suavitzat de la imatge

- L'objectiu és eliminar soroll de la imatge
- És equivalent a suprimir les components d'alta freqüència
- **Filtre Mitjana:**

- El píxel es substitueix per un promitjat local

$$g[i, j] = \frac{1}{m \cdot n} \sum_{k=1}^n \sum_{l=1}^m f[k, l]$$

- S'implementa com una convolució
- És un filtre lineal i separable
- Perdem detalls. Components d'alta freqüència.

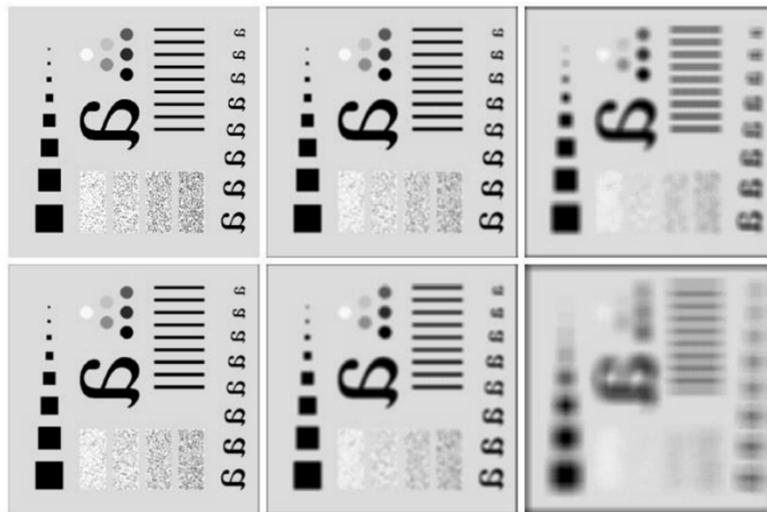
Suavitzem contorns.

- Ponderació dels pesos de la finestra

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



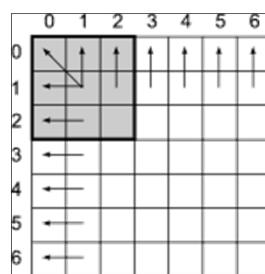
Example of 2-D Convolution



Convolution with box filters of size 1,5,15,3,9,35 (reading order)

Processat a les vores

- Què passa amb les vores ? No tenim valors vàlids



- Image cropping: No es fa el càcul on no es pot
- Inventem el valor dels píxels
- Operem només amb els píxels conegeuts
- Extenem el valor dels píxels propers
- Considerem la imatge cíclica

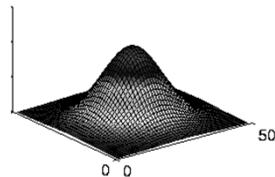
Suavitzat de la imatge

- Filtre gaussià:

- El més popular
- Pesos de la finestra seguint una gaussiana

$$h(x) = e^{-\frac{x^2}{2\sigma^2}}$$

$$h[i, j] = e^{-\frac{(i^2 + j^2)}{2\sigma^2}}$$

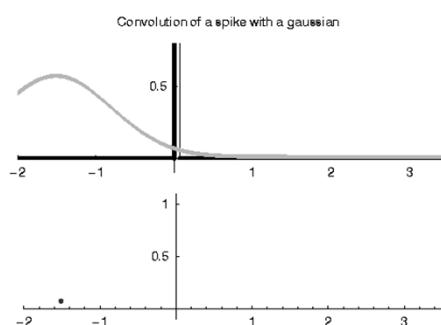
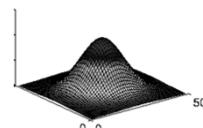


- La influència dels píxels veïns disminueix amb la distància al centre
- Filtre lineal i separable
- Simètrica rotacionalment => suavitzat isotòpic
- Amplada de la finestra (grau de suavitzat) proporcional a σ



Suavitzat de la imatge

- Filtre gaussià:



Filtratge no linial

- Filtre mediana:

- Promitjat: tendència al blurring. Pèrdua de discontinuïtats brusques
- Mediana: redueix el blurring
- Filtre no lineal
- Ordenem els píxels d'un entorn per nivell de gris
- Seleccionem el valor del mig i l'assignem al píxel central
- Molt bo per a eliminar soroll impulsional
- possibles variants:
 - Percentil
 - K-nearest neighbours
 - α -trimmed mean
 - Filtre sigma



Suavitzat de la imatge

original



added noise

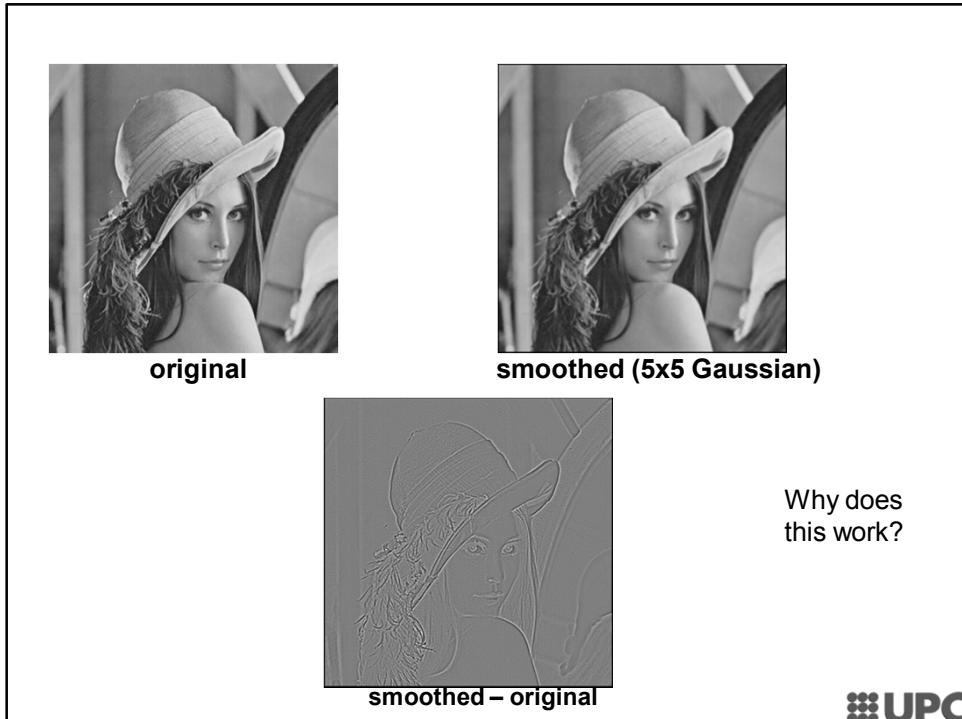


average



median





Detecció de contorns

- **Contorn** : Canvi local i significatiu en la intensitat de la imatge
 - És complicat obtenir les vores a partir dels nivells de gris. Cal un postprocés d'unió i reconstrucció a partir dels contorns

- Edge detection*: localitzar els píxels del contorn.
- Edge enhancement*: realçar els píxels del contorn.
- Edge tracing*: recollir els píxels del contorn en una llista.

-Problemes:

- Degut al soroll en les imatges reals, és difícil trobar un detector de contorns fiable
- El canvi d'intensitat s'estén a més d'un píxel

Significance of Edges in Images

- ↳ There is strong evidence that some form of data compression occurs at an early stage in the human visual system
- ↳ Research suggests that one form of this compression involves finding edges and other information-high features in images
- ↳ From edges we can construct a line drawing of the scene. This results in a huge data compression, but the information content remains very high
- ↳ When we look at a sketch of a scene we do not feel that we are missing a great deal of information



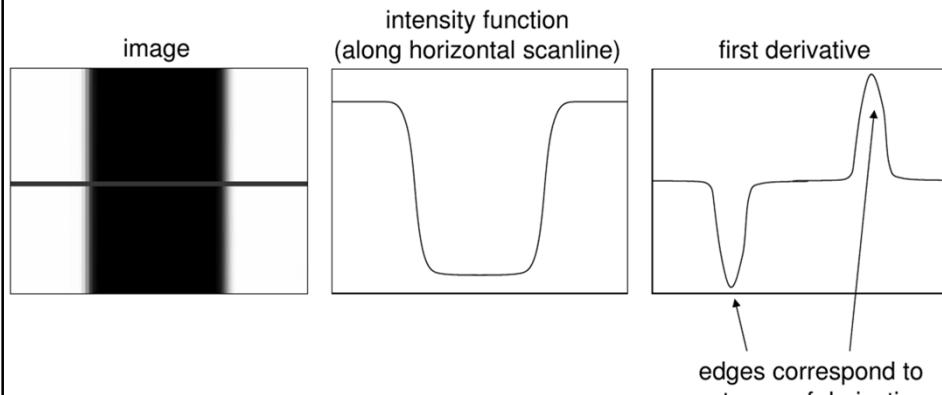
Edge Detection

- ↳ **Goal:** Identify sudden changes (discontinuities) in an image
 - Intuitively, most semantic and shape information from the image can be encoded in the edges
 - More compact than pixels
- ↳ **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

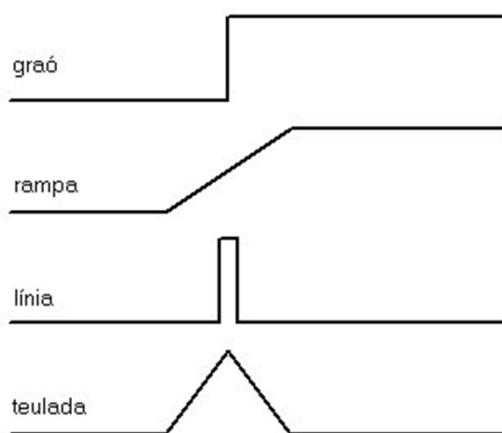


Characterizing Edges

- An edge is a place of rapid change in the image intensity function



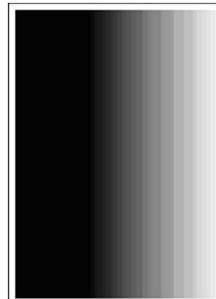
Models de contorn



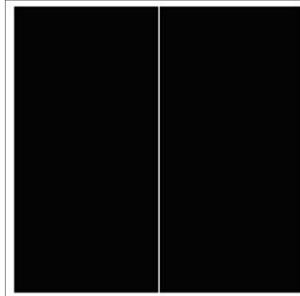
Models de contorn



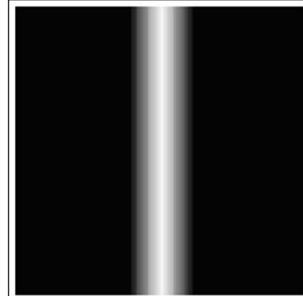
Graó



Rampa



línia

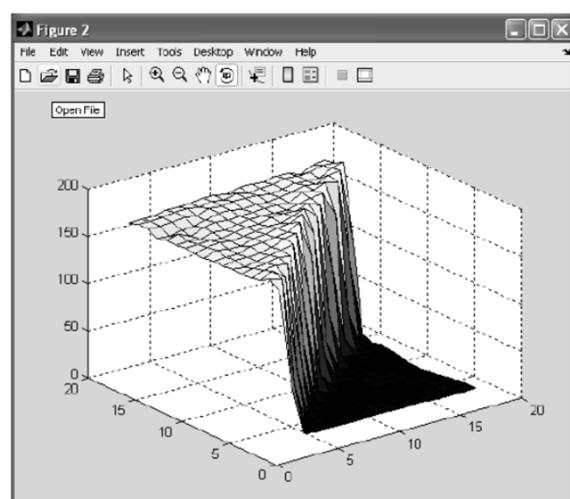


Teulada



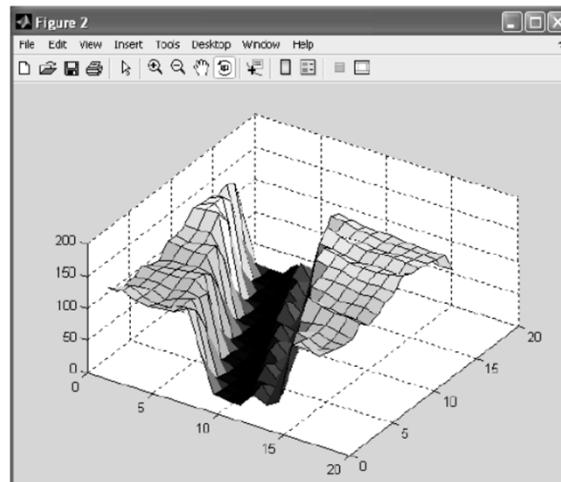
Models de contorn

How does an edge look like?



Models de contorn

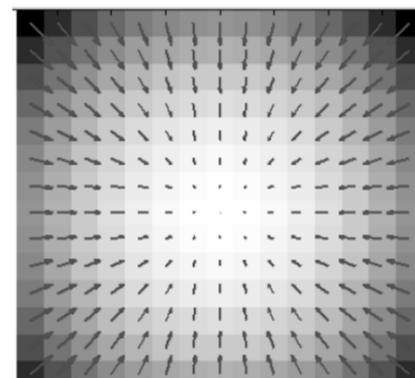
What would a line look like?



Gradient

2-D Gradients

- ↳ Plotted as vectors
- ↳ Gradient at each pixel points uphill
- ↳ The gradient indicates the direction of steepest ascent
- ↳ The gradient is zero at the peak



Operadors tipus gradient

- El gradient és una mesura de canvi d'una funció. S'usa per a detectar contorns
- És l'equivalent 2D de la primera derivada:

$$G[f(x, y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- El mòdul del vector gradient ens dona la magnitud del contorn
- la direcció ens dona l'orientació del contorn

$$\begin{aligned} |G[f(x, y)]| &= \sqrt{G_x^2 + G_y^2} \\ \alpha[G[f(x, y)]] &= \arctan\left(\frac{G_y}{G_x}\right) \end{aligned}$$



Operadors tipus gradient

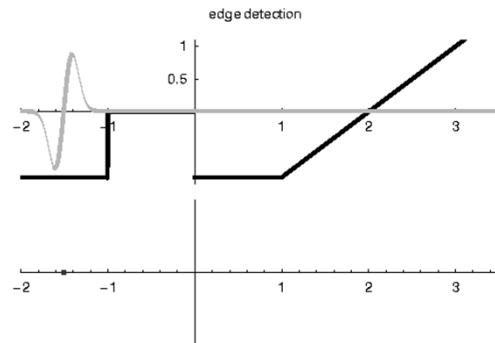
- L'aproximació més simple del gradient és:

$$\begin{aligned} G_x &\cong f[i, j+1] - f[i, j] \\ G_y &\cong f[i, j] - f[i+1, j] \end{aligned}$$

- Aquesta aproximació es pot implementar amb dues senzilles màscares de convolució:

$$M_x = [-1 \quad 1] \quad M_y = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$





Operadors tipus gradient

- L'operador de Roberts:

$$G[f[i, j]] = |f[i, j] - f[i+1, j+1]| + |f[i+1, j] - f[i, j+1]|$$

$$M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

- L'operador Prewitt:

$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$



Operadors tipus gradient

- L'operador Sobel:

$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

- Sobel pondera. Dona més pes als píxels centrals.
- És un filtre separable

$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$



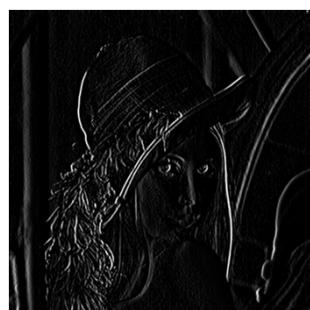
Operador Sobel



Original



Hztal



Vertical



Modul



Operador Sobel

Original



Mòdul



direcció



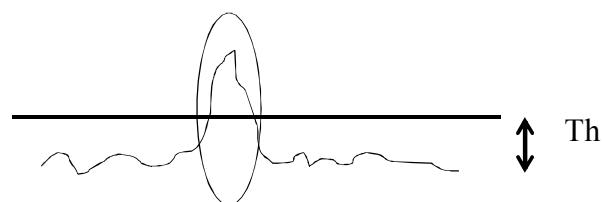
direcció



How do we “detect” edges?

E_s has large values at edges:

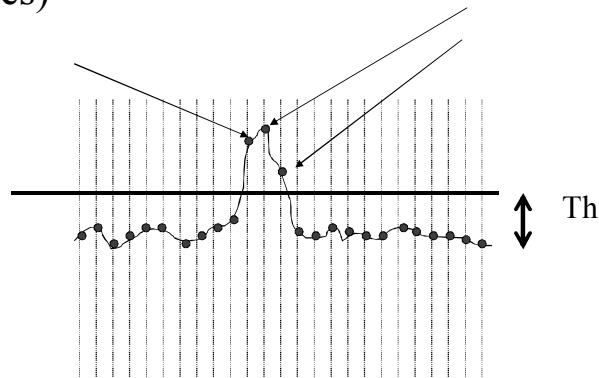
Find local maxima



I com fixem el llindar Th ????????



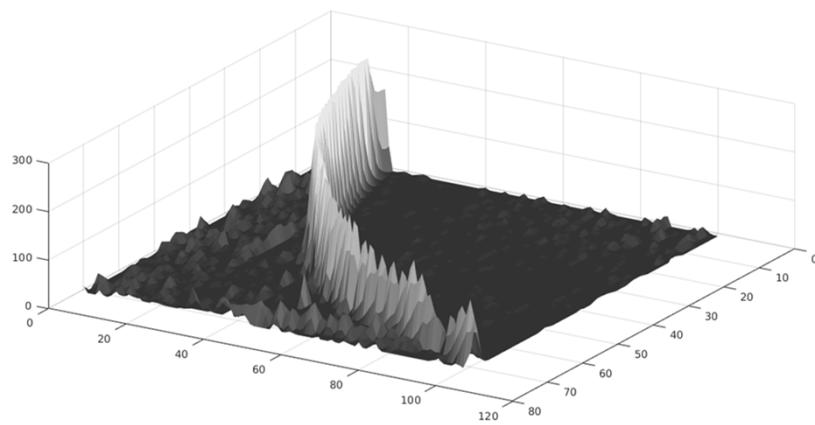
... but it also may have wide ridges around the local maxima (large values *around* the edges)



I com ens quedem amb el màxim local ????????

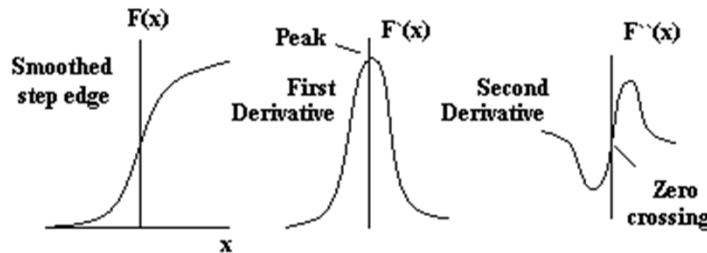


... but it also may have wide ridges around the local maxima (large values *around* the edges)



Operadors de 2^a derivada

- Enlloc de buscar màxims en la 1^a derivada es busquen passos per zero en la 2^a



Operadors de 2^a derivada

- L'operador Laplaciana:

$$\nabla^2 f = \frac{d^2 f}{dx^2} + \frac{d^2 f}{dy^2}$$

- Plantegem les segones derivades usant diferències:

$$\begin{aligned} \frac{d^2 f}{dx^2} &= \frac{dG_x}{dx} = \frac{d(f[i, j+1] - f[i, j])}{dx} = \frac{df[i, j+1]}{dx} - \frac{df[i, j]}{dx} = \\ &= (f[i, j+2] - f[i, j+1]) - (f[i, j+1] - f[i, j]) = f[i, j+2] - 2f[i, j+1] + f[i, j] \end{aligned}$$

- Centrem. Fem el canvi $j=j-1$, $i=i-1$:

$$\begin{aligned} \frac{d^2 f}{dx^2} &= f[i, j+1] - 2f[i, j] + f[i, j-1] \\ \frac{d^2 f}{dy^2} &= f[i+1, j] - 2f[i, j] + f[i-1, j] \end{aligned}$$

- Combinem i obtenim l'operador:

$$\nabla^2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

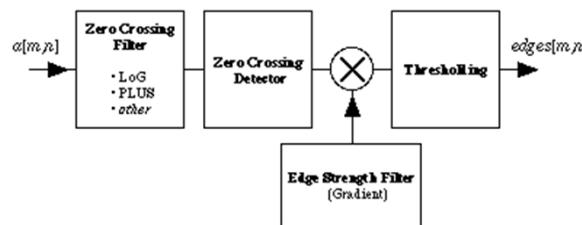


Operadors de 2^a derivada

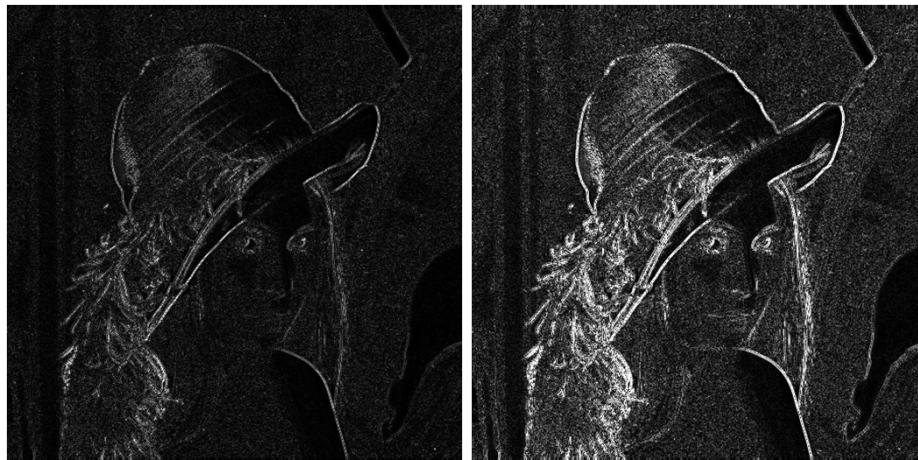
- Quan la sortida de l'operador presenta un pas per zero indica la presència de contorn
- Podem definir també l'operador per a veïnatge-8:

$$\nabla^2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

- També es sol usar una combinació de les tècniques de gradient i 2^a derivada



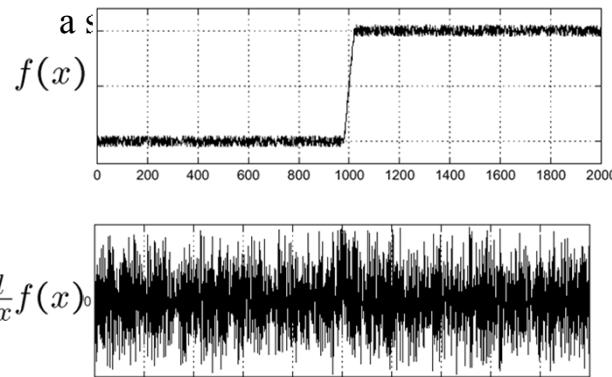
Operador laplaciana



Effects of noise

Consider a single row or column of the image

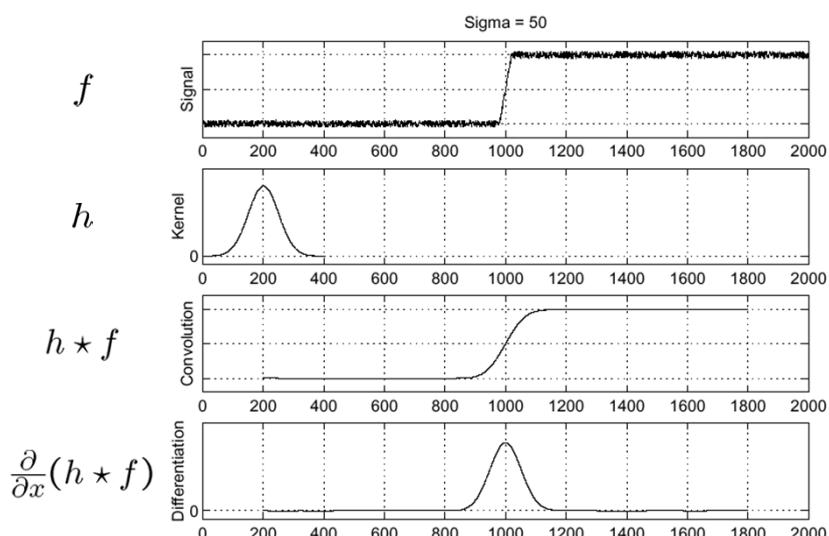
- Plotting intensity as a function of position gives



Where is the edge?



Solution: smooth first



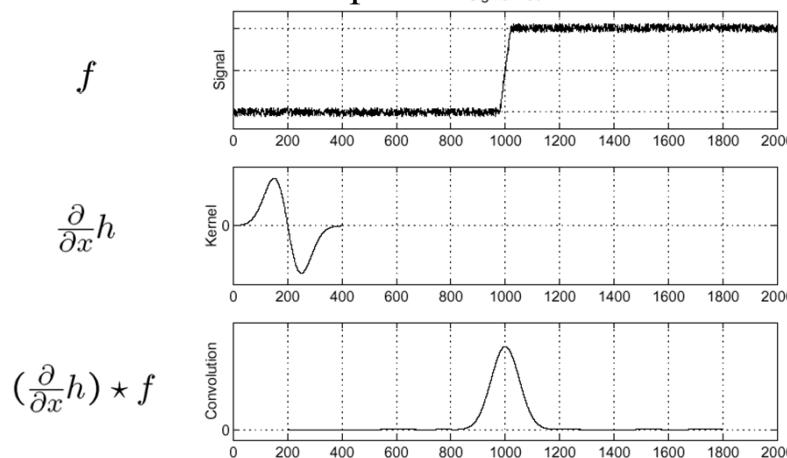
Where is the edge? Look for peaks in $\frac{\partial}{\partial x}(h \star f)$



Associative property of convolution

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

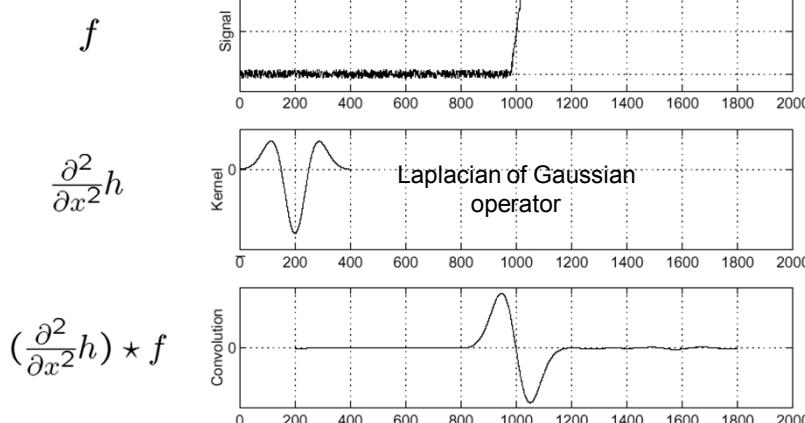
This saves us one operation:



Laplacian of Gaussian

Consider

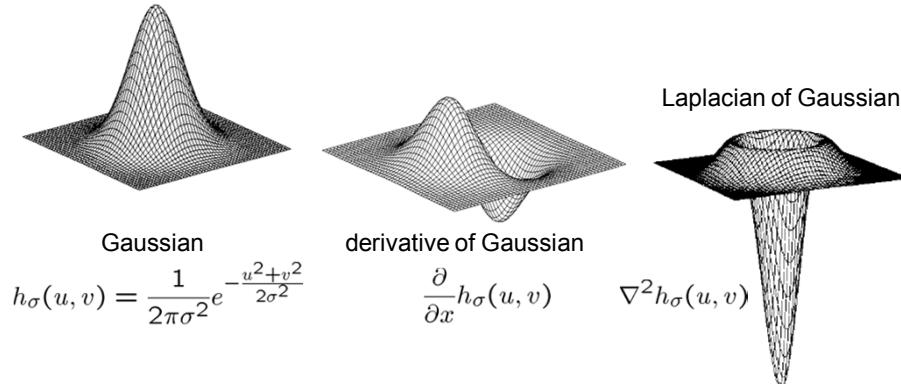
$$\frac{\partial^2}{\partial x^2}(h * f)$$



Where is the edge? Zero-crossings of bottom graph



2D edge detection filters



∇^2 is the **Laplacian** operator:

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



Detecció de contorns

- Etapes en la detecció de contorns:

- 1. Filtrat. El càlcul del gradient es basa en mesures locals, per tant, és sensible al soroll i a la digitalització. El filtrat és útil per a millorar el comportament d'un detector de contorns respecte al soroll.
- 2. Realçat. El realçat emfatitza els píxels on hi ha una variació significativa de la intensitat local.
- 3. Detecció. Només es desitgen els píxels que pertanyen al contorn. Molts dels píxels de la imatge presenten un gradient no nul i no són píxels del contorn. Es sol binaritzar per a eliminar aquests píxels.
- 4. Localització. Es pot estimar també a nivell de subpíxel. També es pot estimar l'orientació.



Operadors de 2^a derivada

- L'operador de Marr-Hildreth:

-Realitza un filtrat gaussià abans de passar el laplaciatà:

- 1. Filtrat gaussià.
- 2. Realçat laplaciatà.
- 3. Detecció de pas per zero de la 2^a derivada corresponent a un pic en la 1^a derivada.
- 4. Estimació de la localització a nivell de subpíxel

$$h(x,y) = \nabla^2 [g(x,y) \otimes f(x,y)] = [\nabla^2 g(x,y)] \otimes f(x,y)$$

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



L'operador de Canny

- Treballa amb la primera derivada de la gaussiana

- 1. Filtrat gaussià.
- 2. Realçat per gradient

$$P[i,j] = (S[i,j+1] - S[i,j]) + (S[i+1,j+1] - S[i+1,j]) / 2$$

$$Q[i,j] = (S[i,j] - S[i+1,j]) + (S[i,j+1] - S[i+1,j+1]) / 2$$

$$M[i,j] = \sqrt{P[i,j]^2 + Q[i,j]^2} \quad \alpha[i,j] = \arctan \frac{Q[i,j]}{P[i,j]}$$

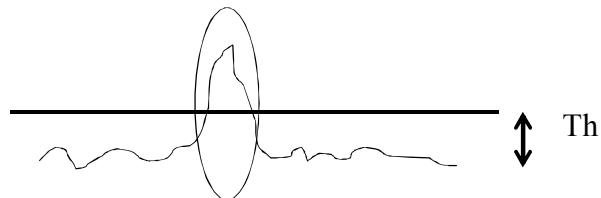
- 3. Supressió dels no-màxims. Obtenim contorn fi.
- 4. Binaritzat amb histèresi



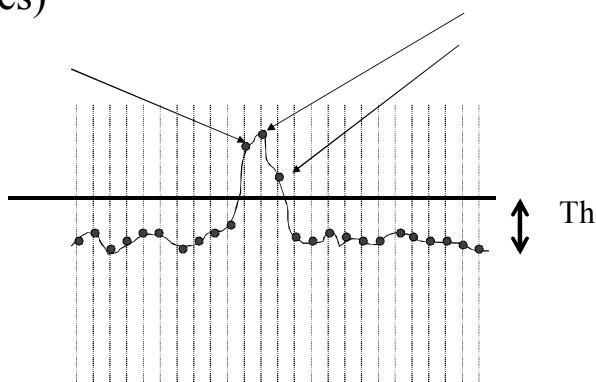
Canny. How do we “detect” edges?

E_s has large values at edges:

Find local maxima

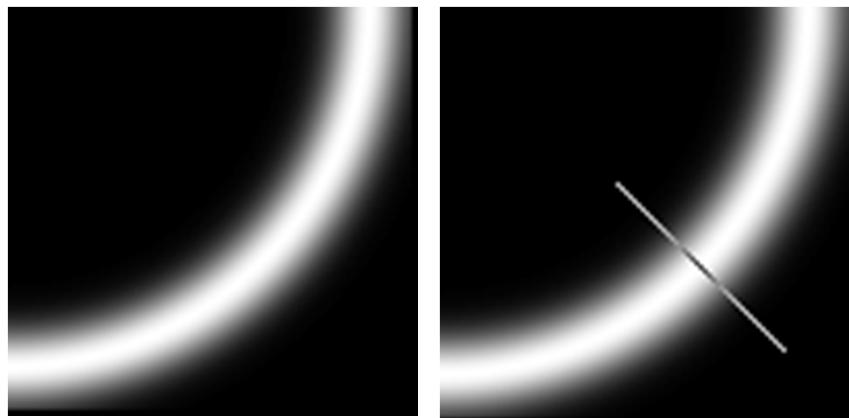


... but it also may have wide ridges around the local maxima (large values *around* the edges)



L'operador de Canny

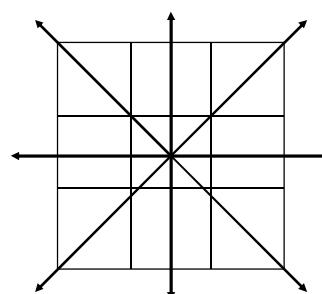
- Supressió de no-màxims:



L'operador de Canny

- Supressió de no-màxims:

- Quantificar 8 possibles direccions



- Comparar el píxel central amb els dos veïns en la direcció del gradient
- Si el valor del píxel central és menor que algun dels veïns, marcar-lo per a esborrar
- Esborrar tots els píxels marcats



L'operador de Canny

- Supressió de no-màxims:



Imatge Original

Magnitud del
gradient

No-màxims
suprimits



L'operador de Canny

- Binaritzat amb histèresi:

- Marcar tots els píxels de contorn amb magnitud superior a T_H com a correctes
- Marcar tots els píxels de contorn amb magnitud inferior a T_L com a incorrectes
- Repassar cada píxel de contorn entre T_L i T_H
- Si aquests pixels tenen un veí marcat com a correcte, els marquem com a correctes
- Repetim els dos últims passos fins que el procés s'estabilitzi



L'operador de Canny

Imatge
Original



Connectats!

forts + febles
connectats

Contorns
forts



Contorns
febles



courtesy of G. Loy



L'operador de Canny

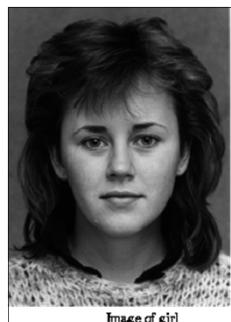


Image of girl



$\sigma=1, m=9, l=0.3, h=0.9$



$\sigma=0.5, m=7, l=0.3, h=0.9$

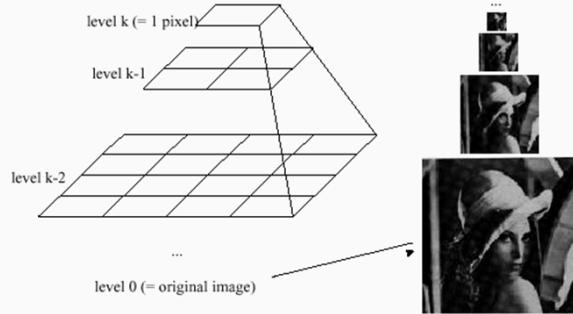


$\sigma=1, m=9, l=0.1, h=0.3$



Some times we want many resolutions

Idea: Represent NxN image as a "pyramid" of
 $1 \times 1, 2 \times 2, 4 \times 4, \dots, 2^k \times 2^k$ images (assuming $N=2^k$)



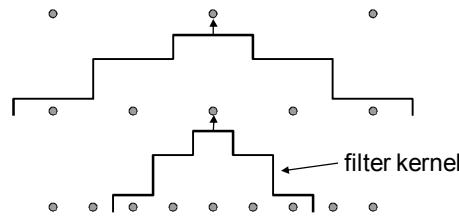
Known as a **Gaussian Pyramid** [Burt and Adelson, 1983]

- In computer graphics, a *mip map* [Williams, 1983]
- A precursor to *wavelet transform*

Gaussian Pyramids have all sorts of applications in computer vision



Gaussian pyramid construction



Repeat

- Filter
- Subsample

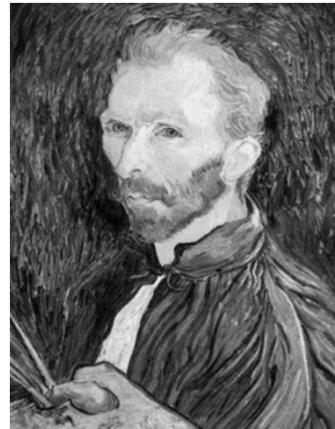
Until minimum resolution reached

- can specify desired number of levels (e.g., 3-level pyramid)

The whole pyramid is only $4/3$ the size of the original image!



Subsampling with Gaussian pre-filtering



Gaussian 1/2



G 1/4

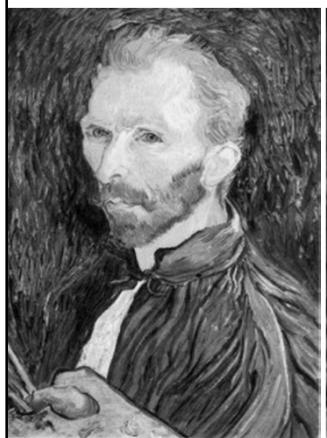


G 1/8

Filter the image, *then* subsample



Subsampling with Gaussian pre-filtering



Gaussian 1/2



G 1/4



G 1/8

Filter the image, *then* subsample



Subsampling *without* pre-filtering



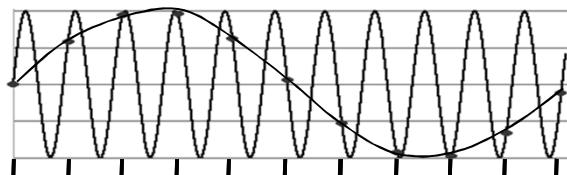
1/2

1/4 (2x zoom)

1/8 (4x zoom)



Sampling and the Nyquist rate



Aliasing can arise when you sample a continuous signal or image

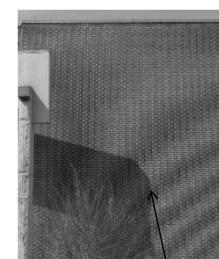
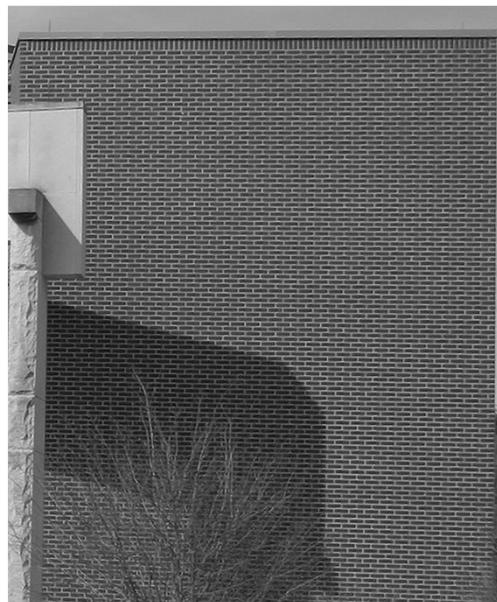
- occurs when your sampling rate is not high enough to capture the amount of detail in your image
- Can give you the wrong signal/image—an *alias*
- formally, the image contains structure at different scales
 - called “frequencies” in the Fourier domain
- the sampling rate must be high enough to capture the highest frequency in the image

To avoid aliasing:

- sampling rate $\geq 2 * \text{max frequency in the image}$
 - said another way: \geq two samples per cycle
- This minimum sampling rate is called the **Nyquist rate**



Aliasing



Aliased signal

<https://www.youtube.com/watch?v=yr3ngmRuGUc>