Robert Collins
CMPEN454

LoG and DoG Filters (second derivatives)

Today's Topics

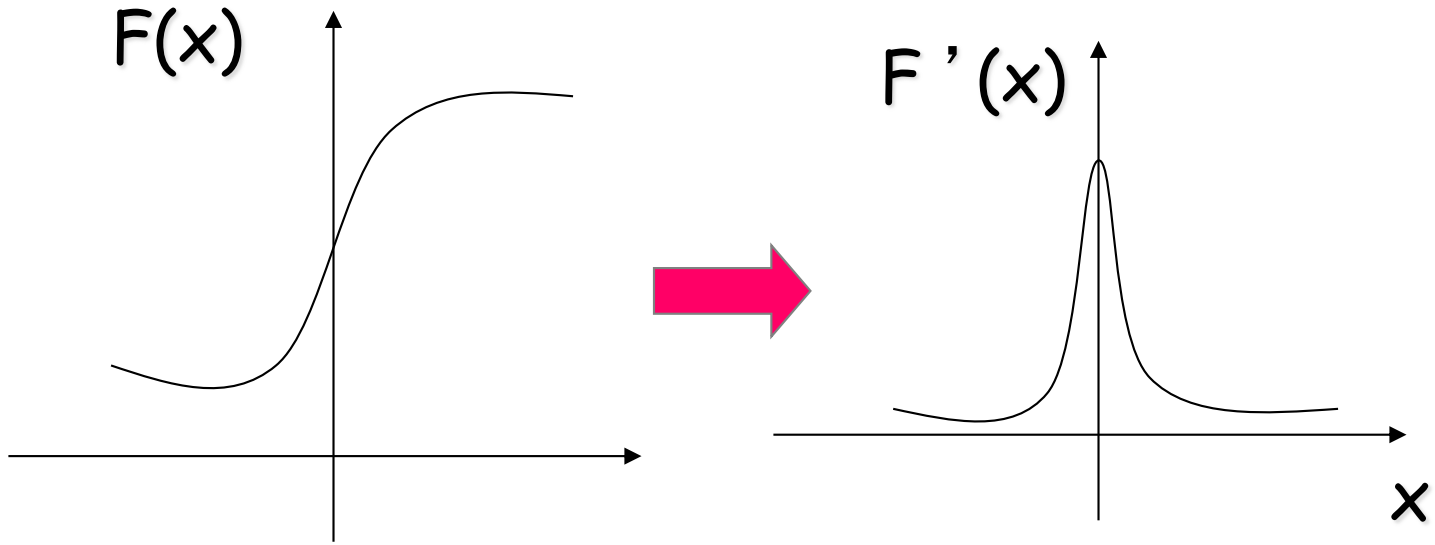
Second derivative filters

Laplacian of Gaussian (LoG) Filter

Approximation using Difference of Gaussian (DoG)

Recall: First Derivative Filters

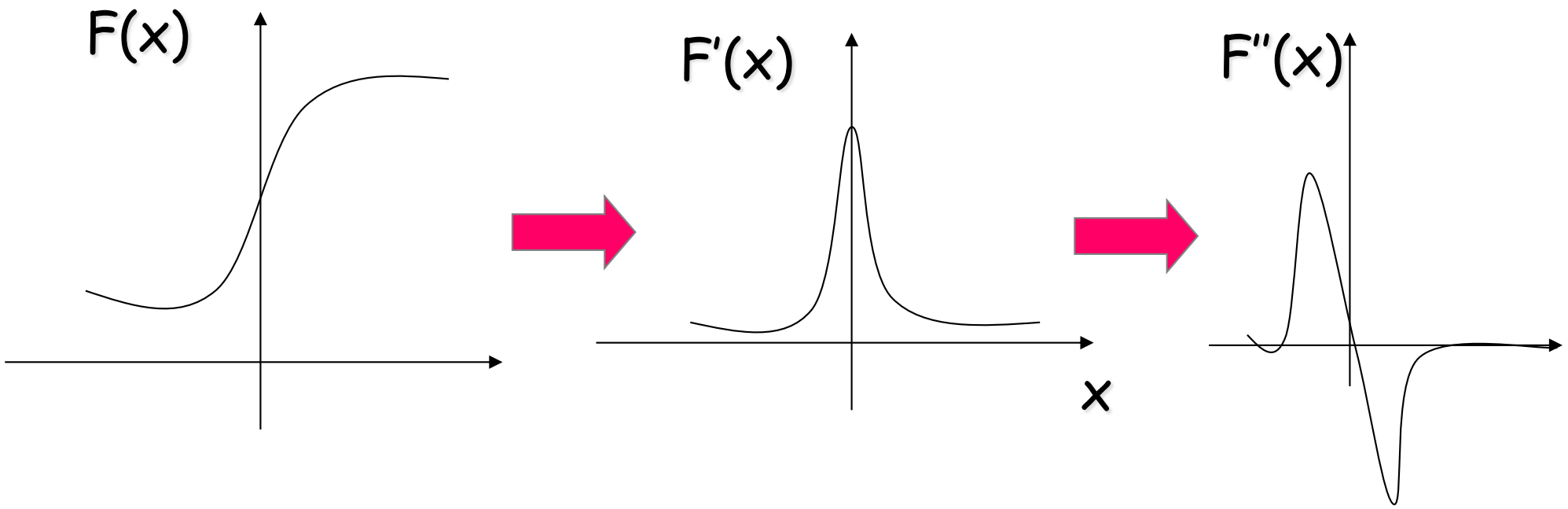
- Sharp changes in gray level of the input image correspond to "peaks or valleys" of the first-derivative of the input signal.



(1D example)

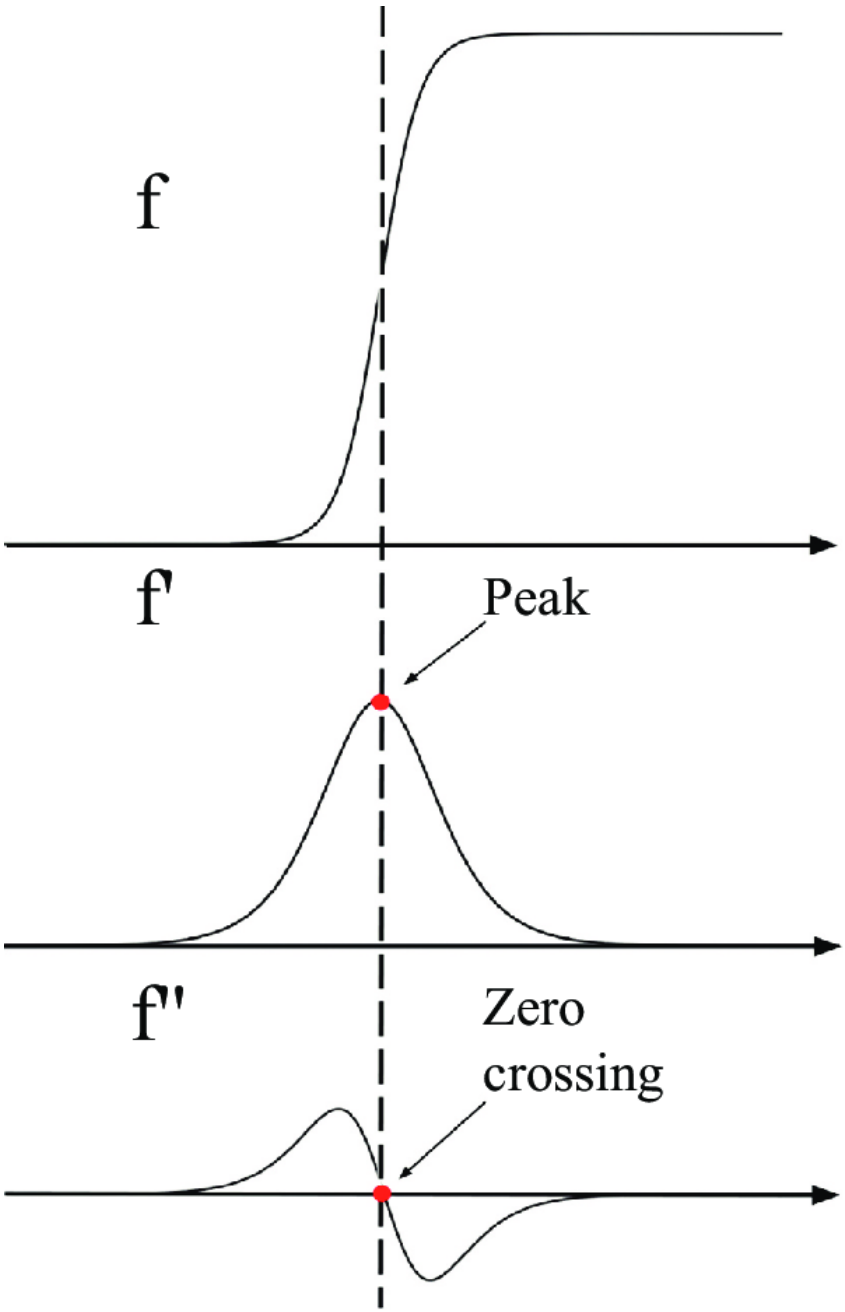
Second-Derivative Filters

- Peaks or valleys of the first-derivative of the input signal, correspond to "zero-crossings" of the second-derivative of the input signal.



Robert Collins
CMPEN454

Spatial Relationships



Intensity function

First deriv

First deriv of first deriv, aka Second deriv

Robert Collins
CMPEN454

Numerical Derivatives

See also T&V, Appendix A.2

Taylor Series expansion

$$f(x+h) = f(x) + hf'(x) + \frac{1}{2}h^2 f''(x) + \frac{1}{3!}h^3 f'''(x) + O(h^4)$$

add

$$+ \left[f(x-h) = f(x) - hf'(x) + \frac{1}{2}h^2 f''(x) - \frac{1}{3!}h^3 f'''(x) + O(h^4) \right]$$

$$f(x+h) + f(x-h) = 2f(x) + h^2 f''(x) + O(h^4)$$

$$\frac{f(x-h) - 2f(x) + f(x+h)}{h^2} = f''(x) + O(h^2)$$

1	-2	1

Central difference approx to second derivative

Robert Collins
CMPEN454

Example: Second Derivatives

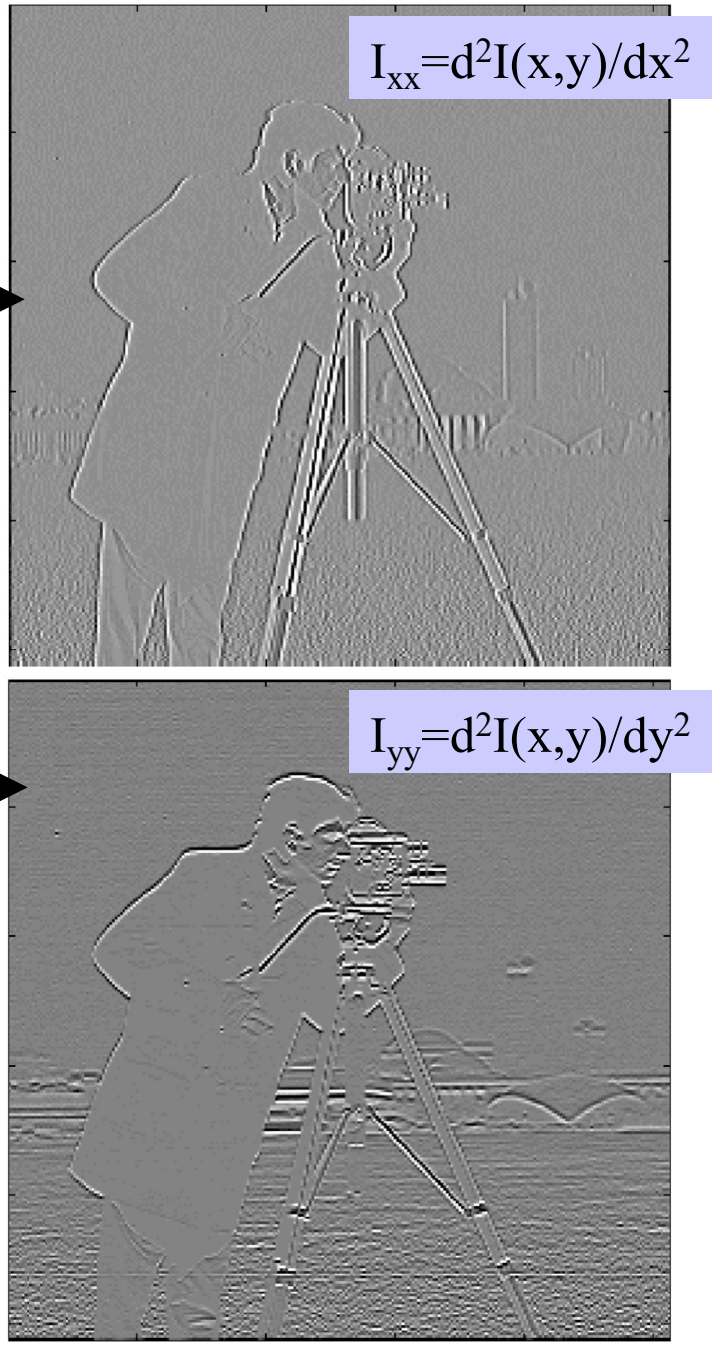


$$\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$$

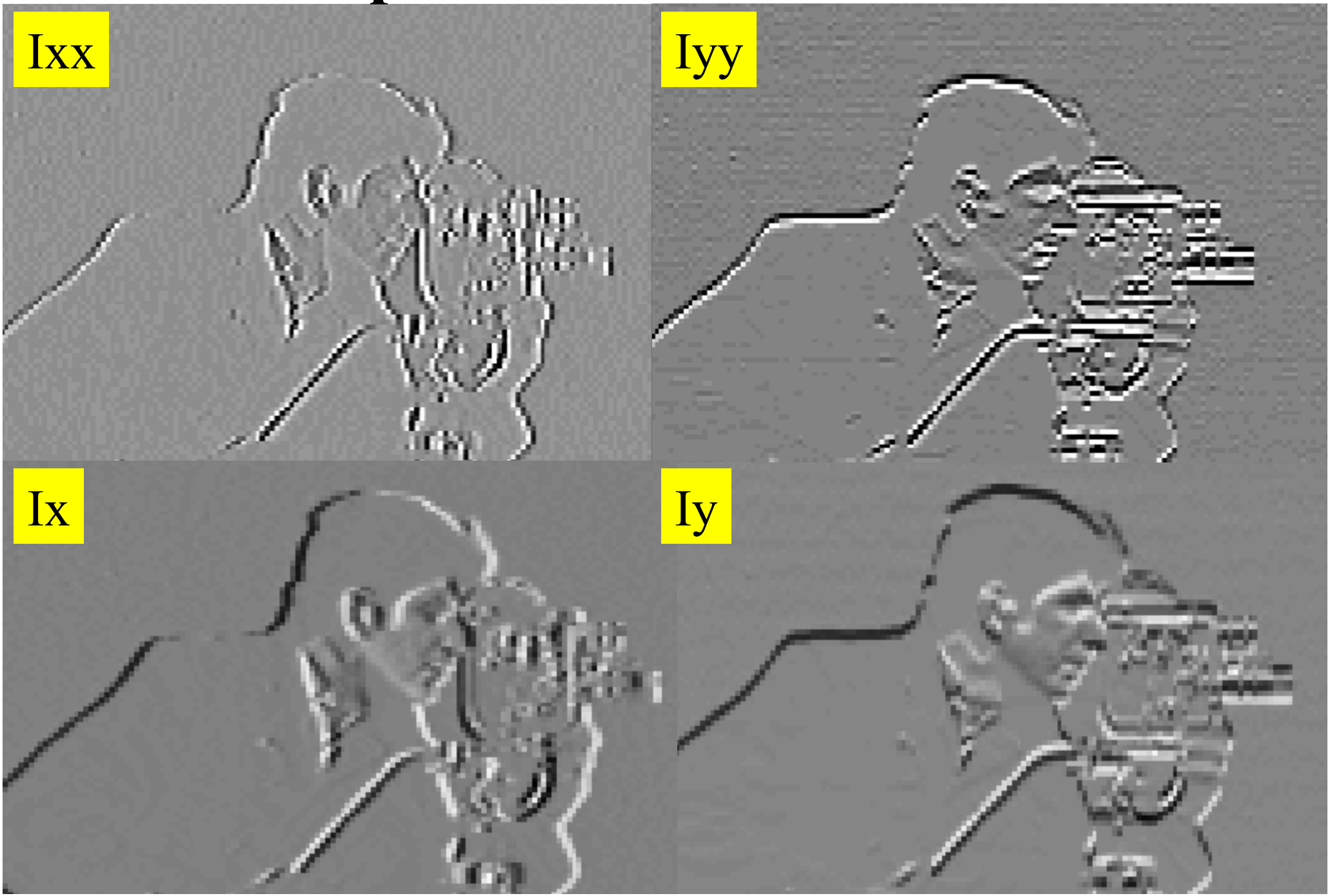
2nd Partial deriv wrt x

$$\begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}$$

2nd Partial deriv wrt y



Compare: 1st vs 2nd Derivatives



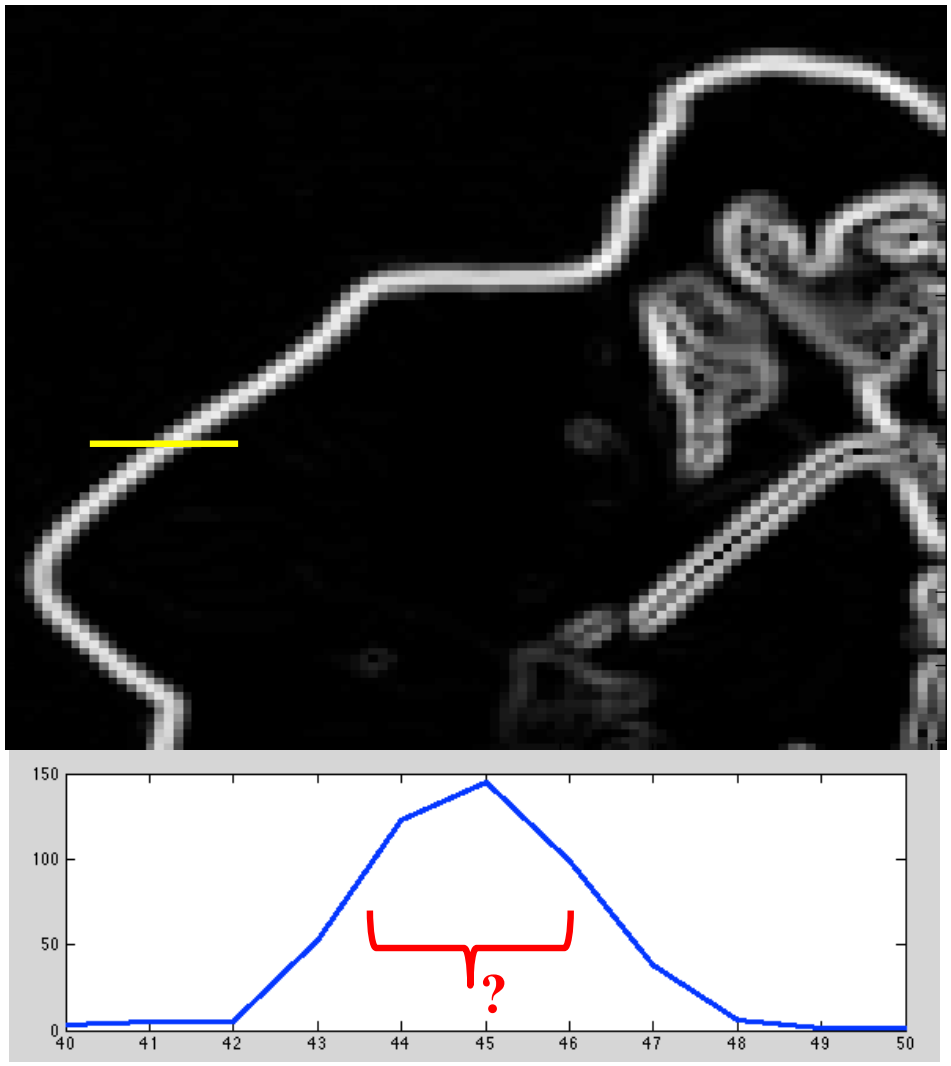
Finding Edges in 1D

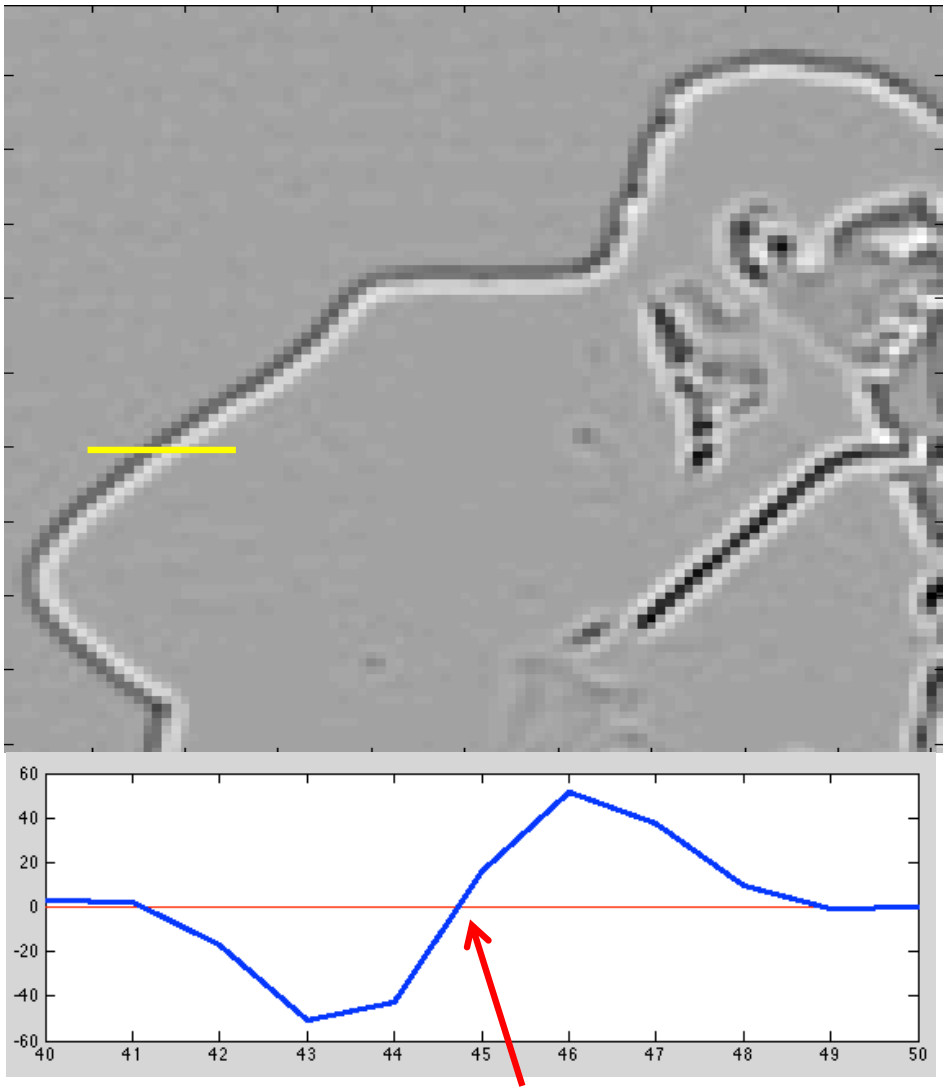
An alternative to finding edges as high magnitude values in first deriv is to find zero-crossings in second deriv.

In 1D, convolve with [1 -2 1] and look for pixels where response is (nearly) zero.

Why is this a Good Approach?

Gradient magnitude versus Laplacian





benefit: you get clear, subpixel localization of the edge, as opposed to a "smear" of high magnitude gradient values across edge using gradient magnitude

Finding Edges in 1D

An alternative to finding edges as high magnitude values in first deriv is to find zero-crossings in second deriv.

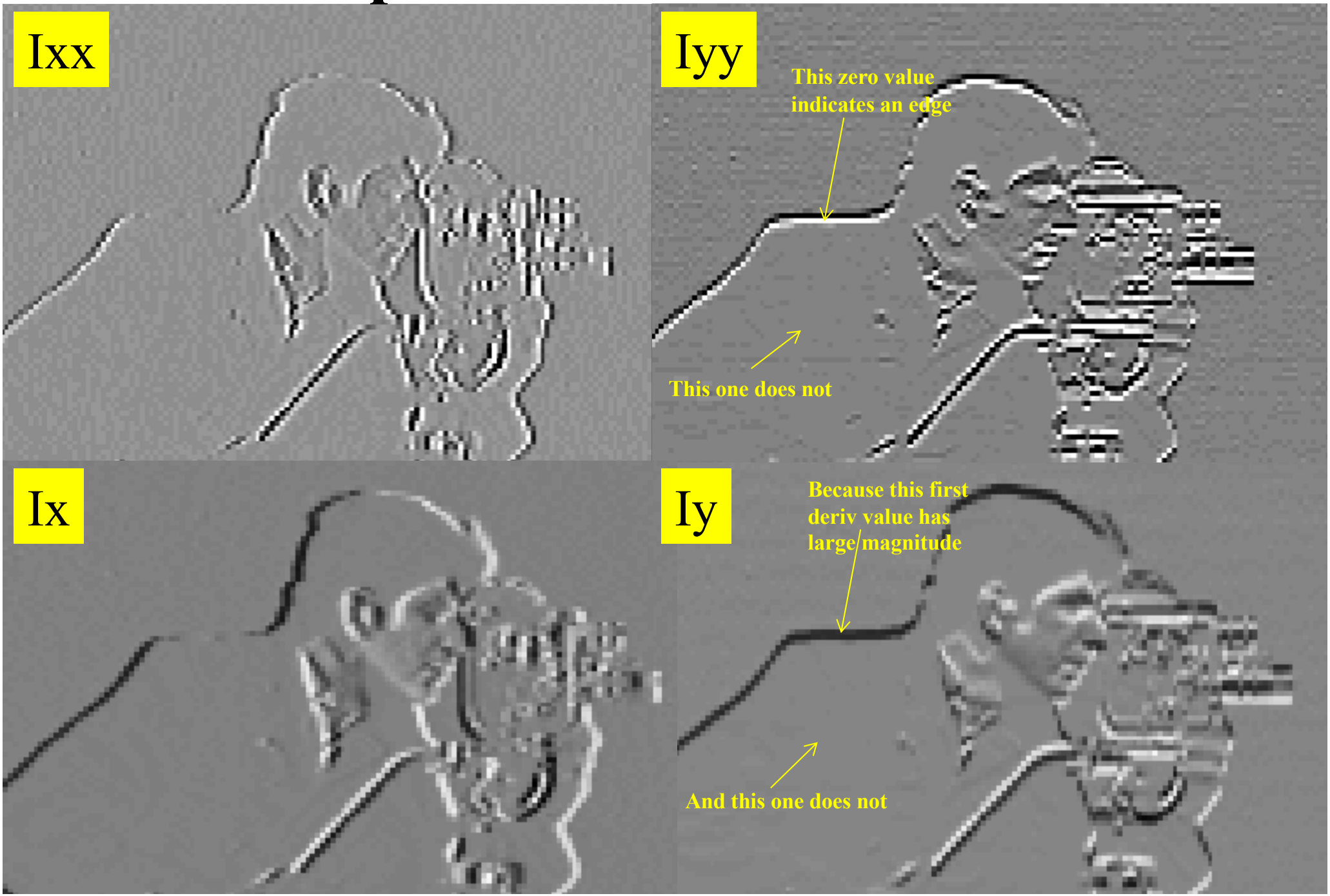
In 1D, convolve with [1 -2 1] and look for pixels where response is (nearly) zero.

Problem: when first deriv is zero, so is second. I.e. the filter [1 -2 1] also produces zero when convolved over regions of constant intensity.

So, in 1D, convolve with [1 -2 1] and look for pixels where response is nearly zero AND magnitude of first derivative is "large enough".

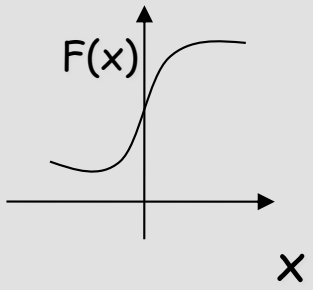
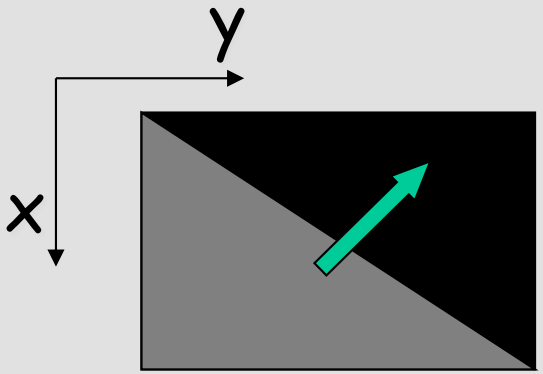
Robert Collins
CMPEN454

Compare: 1st vs 2nd Derivatives



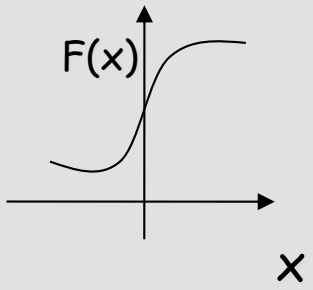
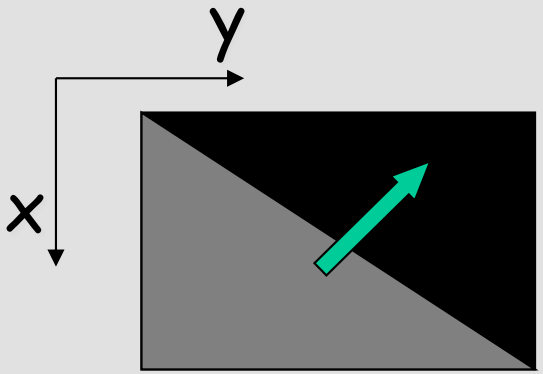
Robert Collins
CMPEN454

Edge Detection Summary

	1D	2D
step edge	$I(x)$ 	$I(x,y)$ 
1st deriv	$\left\| \frac{dI(x)}{dx} \right\| > Th$	$\|\nabla I(x,y)\| = (I_x^2(x,y) + I_y^2(x,y))^{1/2} > Th$ $\tan\theta = I_x(x,y) / I_y(x,y)$
2nd deriv	$\frac{d^2I(x)}{dx^2} = 0$	

Robert Collins
CMPEN454

Edge Detection Summary

	1D	2D
step edge	$I(x)$ 	$I(x,y)$ 
1st deriv	$\left\lvert \frac{dI(x)}{dx} \right\rvert > Th$	$\lvert \nabla I(x,y) \rvert = (I_x^2(x,y) + I_y^2(x,y))^{1/2} > Th$ $\tan\theta = I_x(x,y) / I_y(x,y)$
2nd deriv	$\frac{d^2I(x)}{dx^2} = 0$	$\nabla^2 I(x,y) = I_{xx}(x,y) + I_{yy}(x,y) = 0$ Laplacian

Finite Difference Laplacian

$$I_{xx} + I_{yy} = \left(\begin{bmatrix} 1 & -2 & 1 \end{bmatrix} * I \right) + \left(\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} * I \right)$$

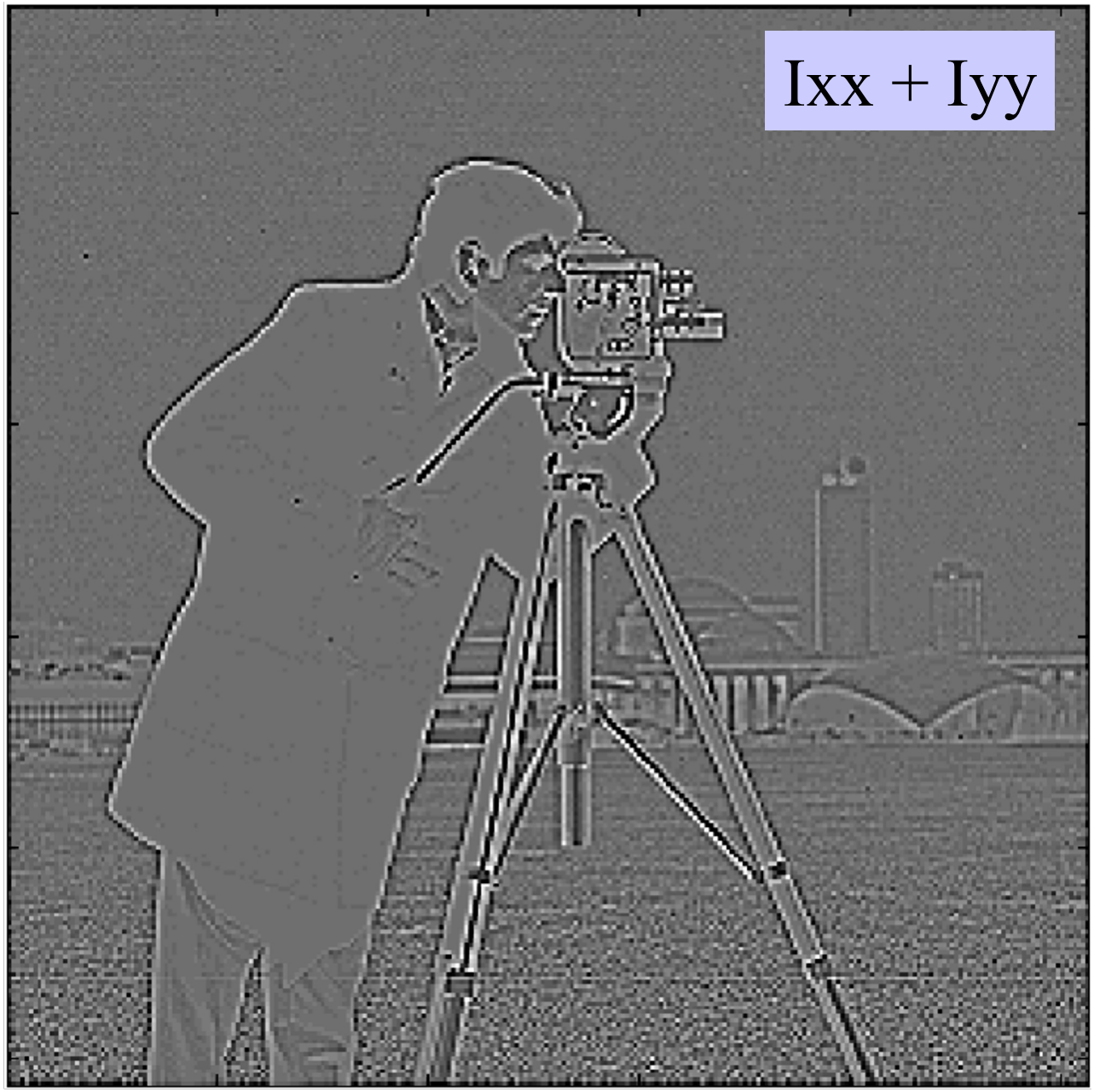
$$= \left(\begin{bmatrix} 1 & -2 & 1 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} \right) * I$$

$$= \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}}_{\text{Laplacian filter } \nabla^2 I(x,y)} * I$$

Example: Laplacian



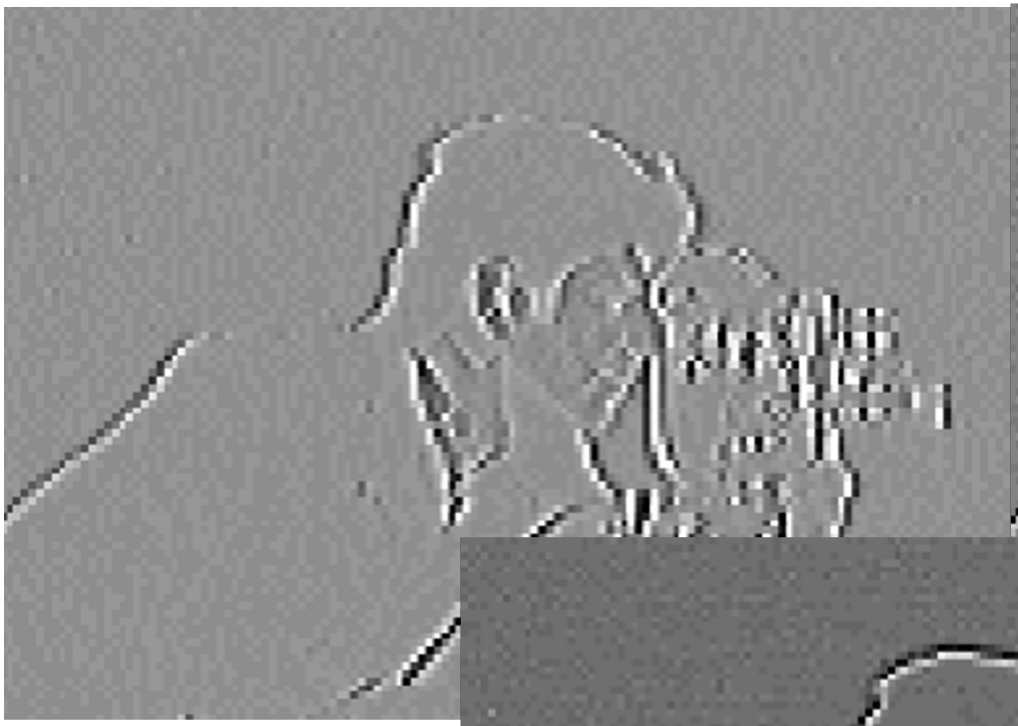
$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} * I$$



Example: Laplacian

Ixx

Iyy





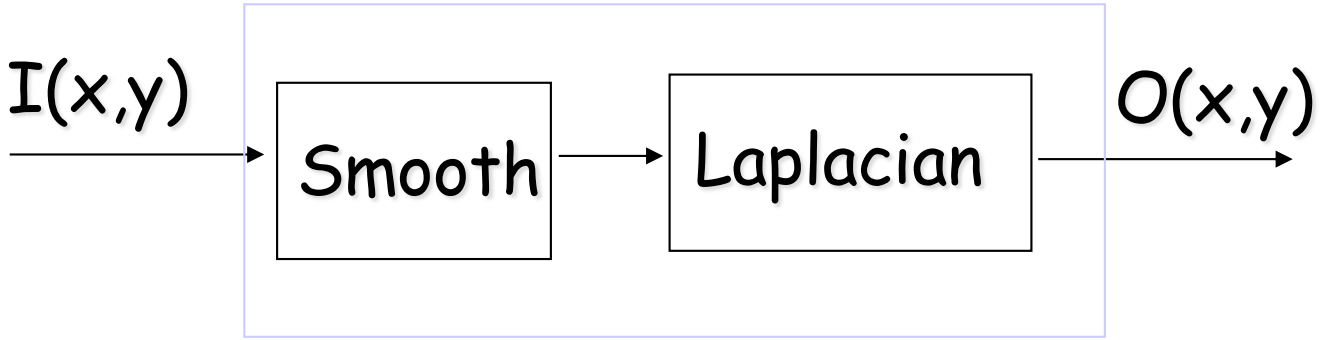


Ixx+Iyy

$\nabla^2 I(x,y)$

Notes about the Laplacian:

- $\nabla^2 I(x,y)$ is a SCALAR
  - ↑ Can be found using a SINGLE convolution filter
  - ↓ Orientation information is lost
- $\nabla^2 I(x,y)$ is the sum of SECOND-order derivatives
  - But taking derivatives increases noise
  - Very noise sensitive!
- It is always combined with a smoothing operation:



LoG Filter

- First smooth (Gaussian filter),
- Then, find zero-crossings (Laplacian filter):
  - $O(x,y) = \nabla^2(I(x,y) * G(x,y))$

Just another linear filter.

$$\nabla^2(f(x,y) \otimes G(x,y)) = \nabla^2 G(x,y) \otimes f(x,y)$$

Laplacian of Gaussian-filtered image

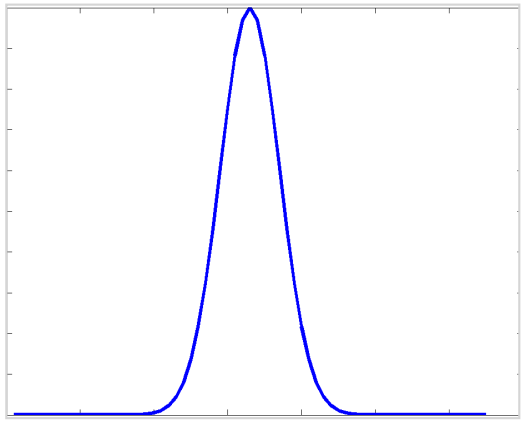
Laplacian of Gaussian (LoG) -filtered image

Do you see the distinction?

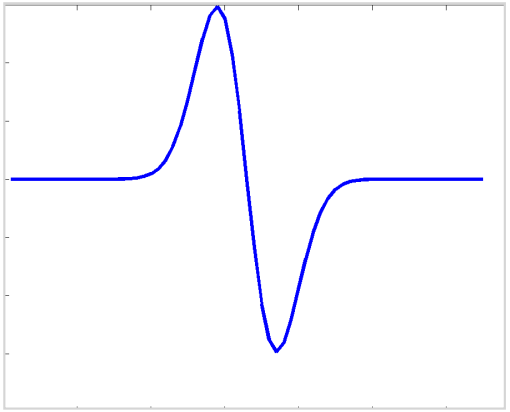
Robert Collins
CMPEN454

1D Gaussian and Derivatives

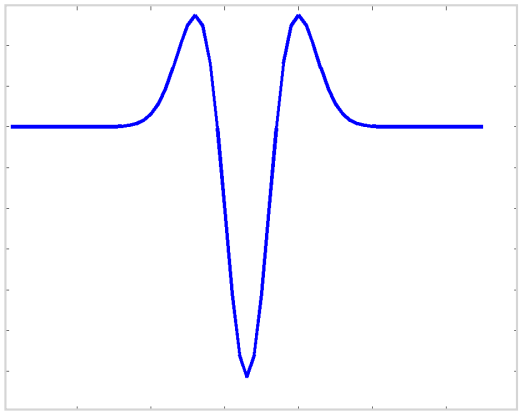
$$G(x;\sigma) = e^{-\frac{x^2}{2\sigma^2}}$$



$$G_X(x;\sigma) = -\frac{1}{2\sigma^2} 2xe^{-\frac{x^2}{2\sigma^2}} = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$



$$G_{XX}(x;\sigma) = \left(\frac{x^2}{\sigma^4} - \frac{1}{\sigma^2}\right) e^{-\frac{x^2}{2\sigma^2}}$$

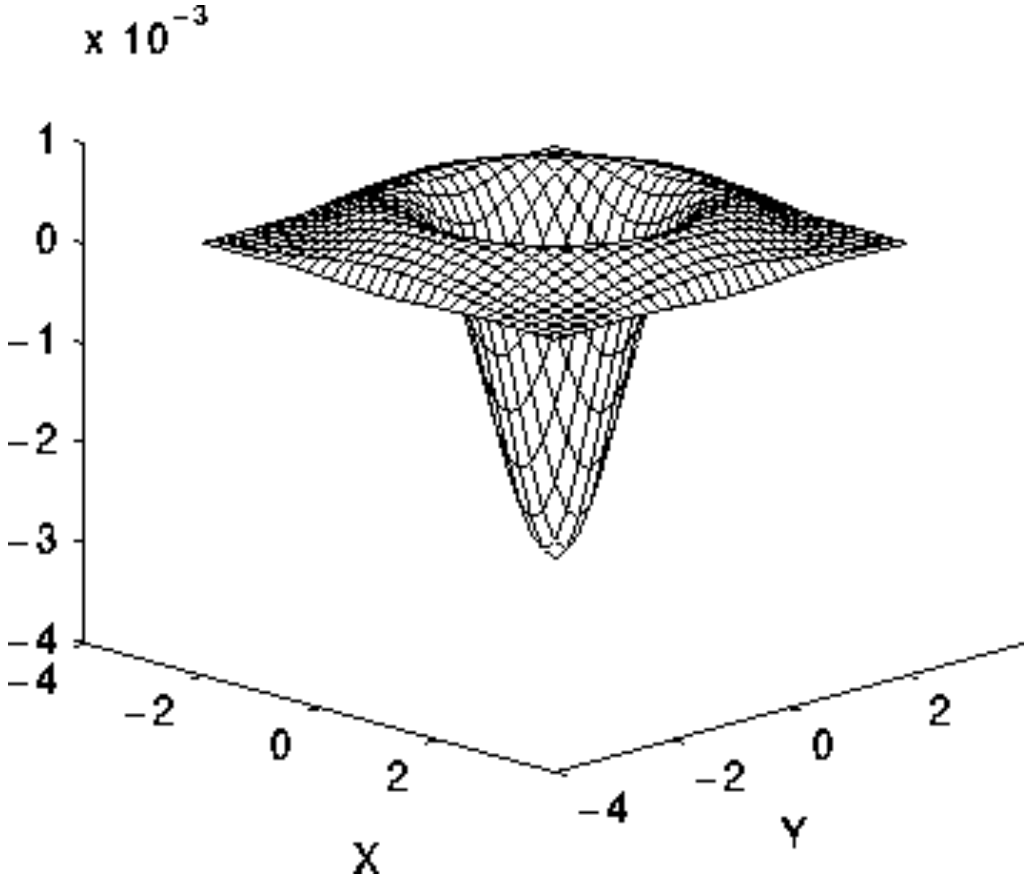


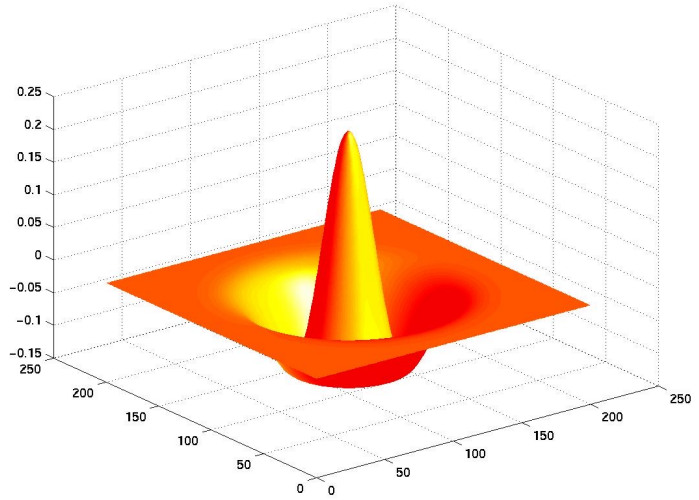
Robert Collins
CMPEN454

2D Laplacian of Gaussian

$$\text{LoG}(x,y;\sigma) = G_{xx}(x,y;\sigma) + G_{yy}(x,y;\sigma)$$

$$= -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$





Non-PC historical note:

"Mexican Hat" operator

Zero-Crossings as an Edge Detector

Raw zero-crossings (no contrast thresholding)



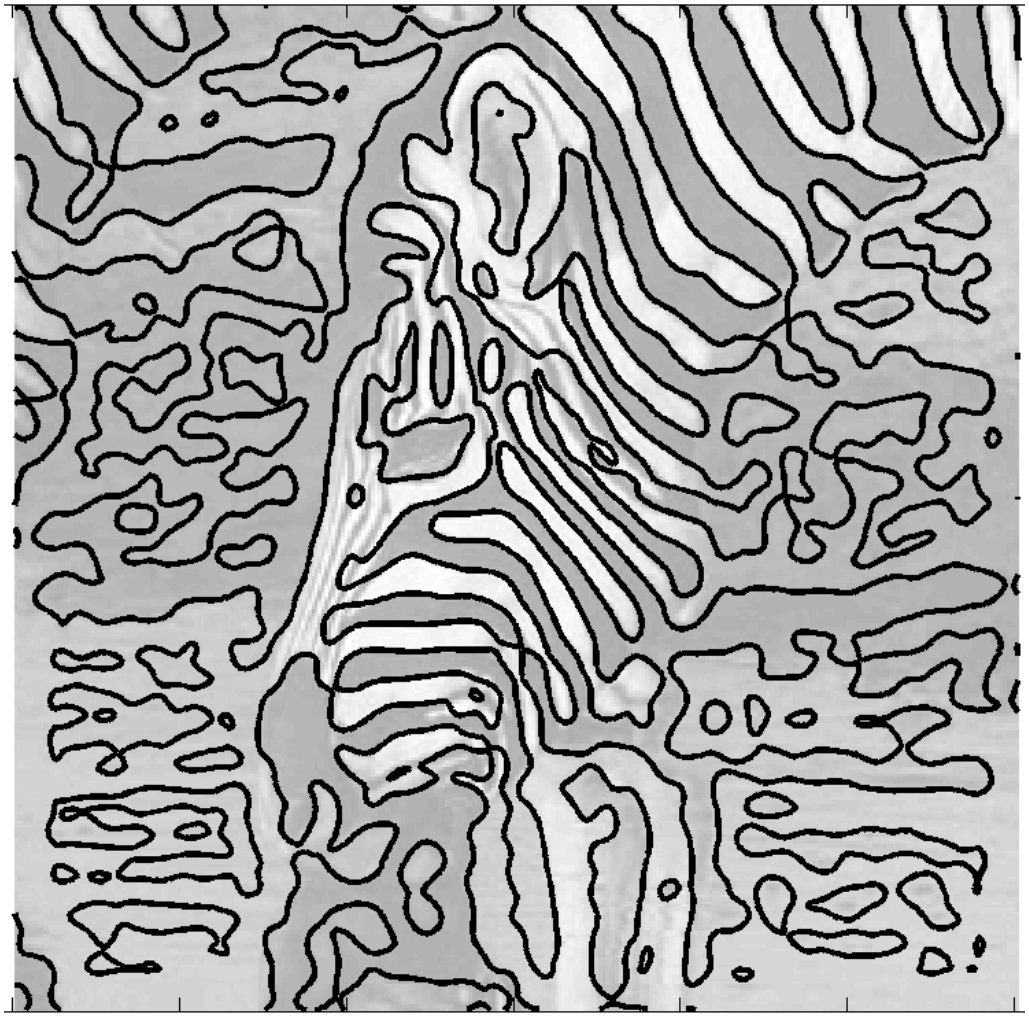


LoG sigma = 2, zero-crossing

Zero-Crossings as an Edge Detector

Raw zero-crossings (no contrast thresholding)



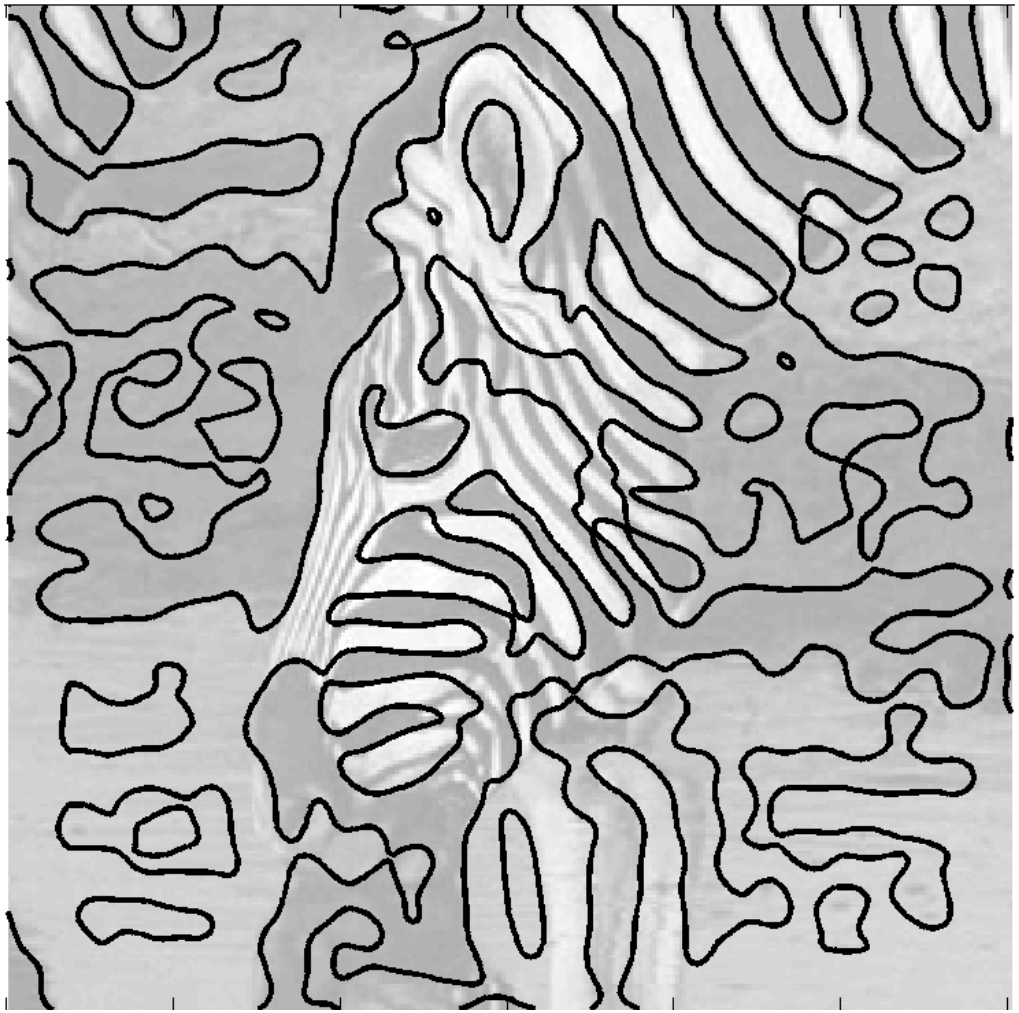


LoG sigma = 4, zero-crossing

Zero-Crossings as an Edge Detector

Raw zero-crossings (no contrast thresholding)





LoG sigma = 8, zero-crossing

Robert Collins
CMPEN454

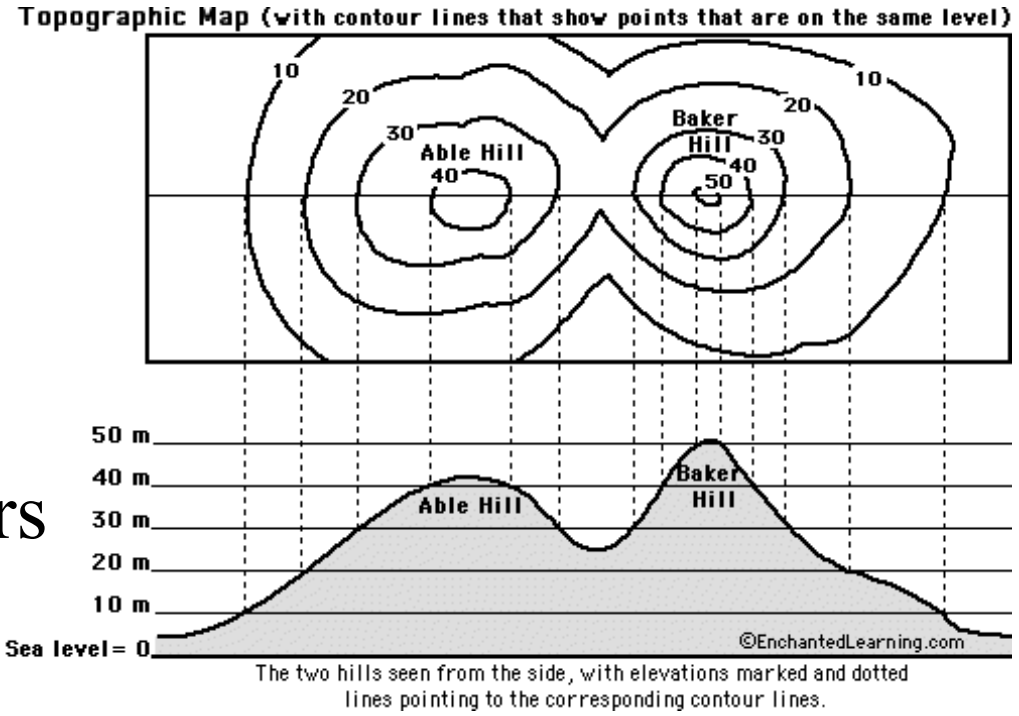
Note: Closed Contours

You may have noticed that zero-crossings form closed contours. It is easy to see why…

Think of equal-elevation contours on a topo map.

Each is a closed contour.

Zero-crossings are contours at elevation = 0 .



remember that in our case, the height map is of a LoG filtered image - a surface with both positive and negative "elevations"

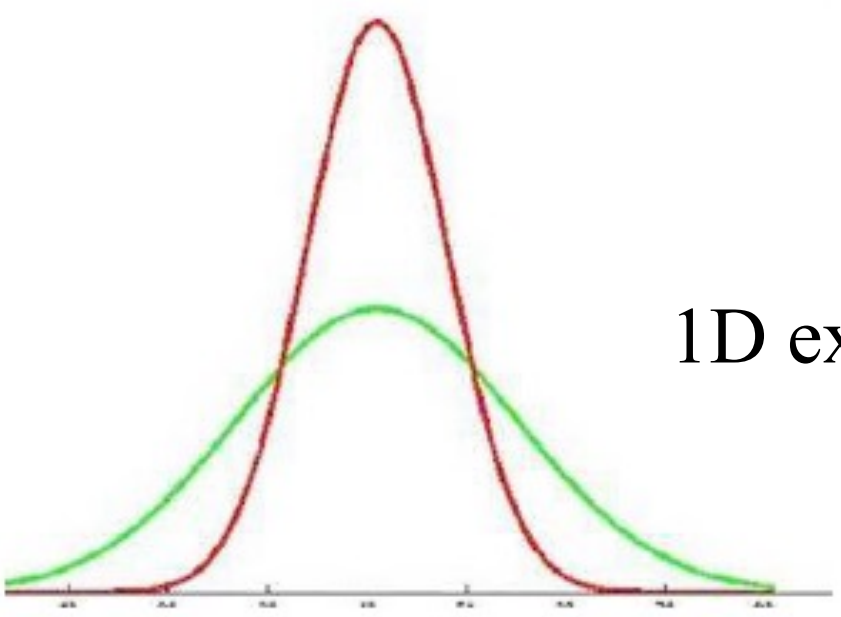
Efficient Implementation
Approximating LoG with DoG

LoG can be approximated by a Difference of two Gaussians (DoG) having different sigma values

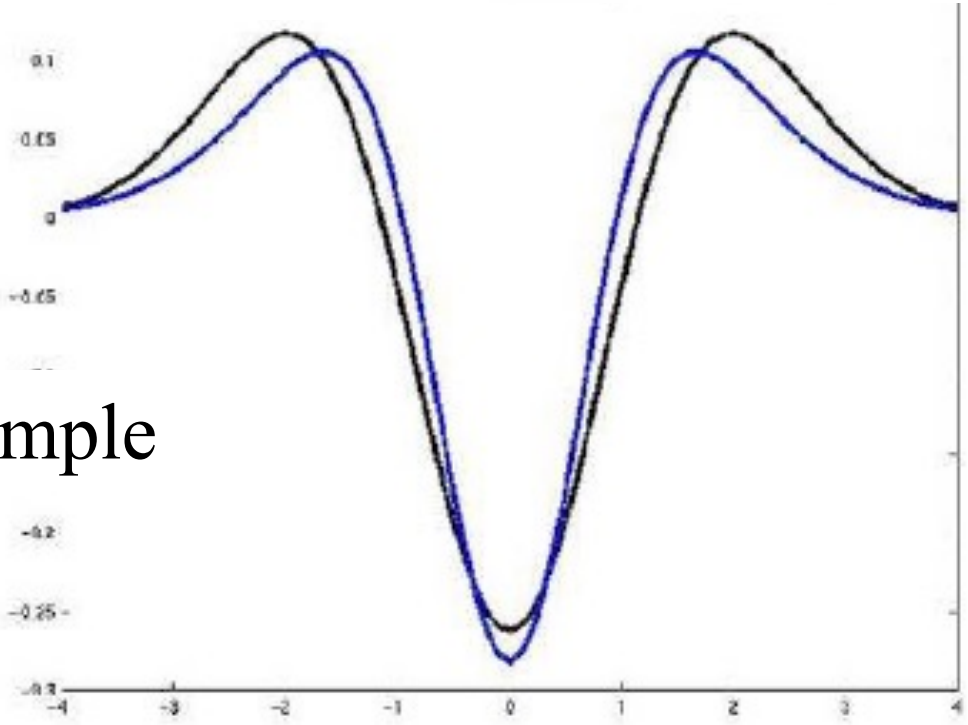
$$\nabla^2 G_\sigma \approx G_{\sigma_1} - G_{\sigma_2}$$

Best approximation when:

$\sigma_1/\sigma_2 \approx 1.6$



1D example



Efficient Implementation

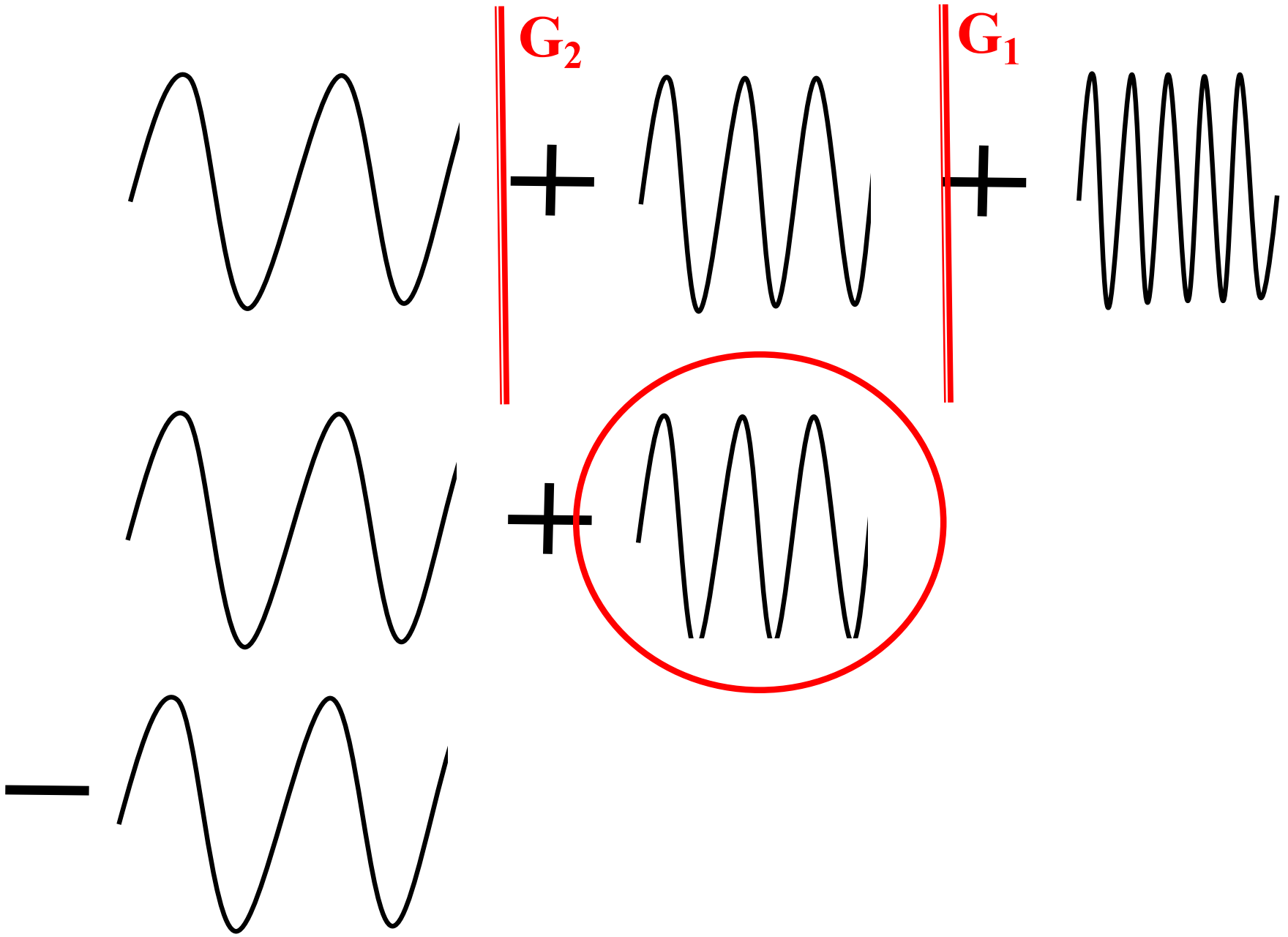
LoG can be approximate by a Difference of two Gaussians (DoG) at different scales.

Separability of and cascadability of Gaussians applies to the DoG, so we can achieve efficient implementation of the LoG operator.

DoG approx also explains bandpass filtering of LoG (think about it. Hint: Gaussian is a low-pass filter)

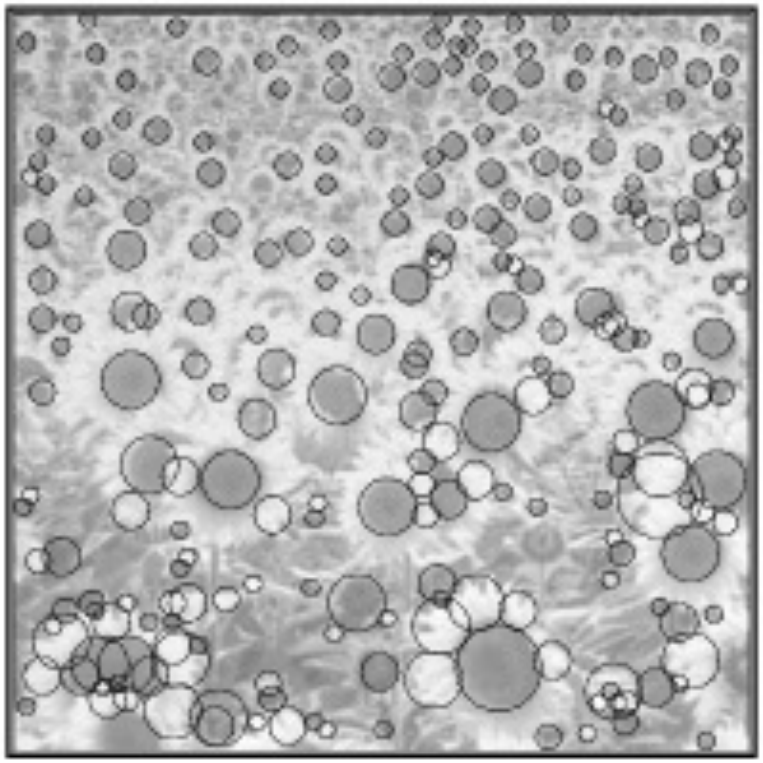
Robert Collins
CMPEN454

DoG Bandpass



Other uses of LoG: Blob Detection



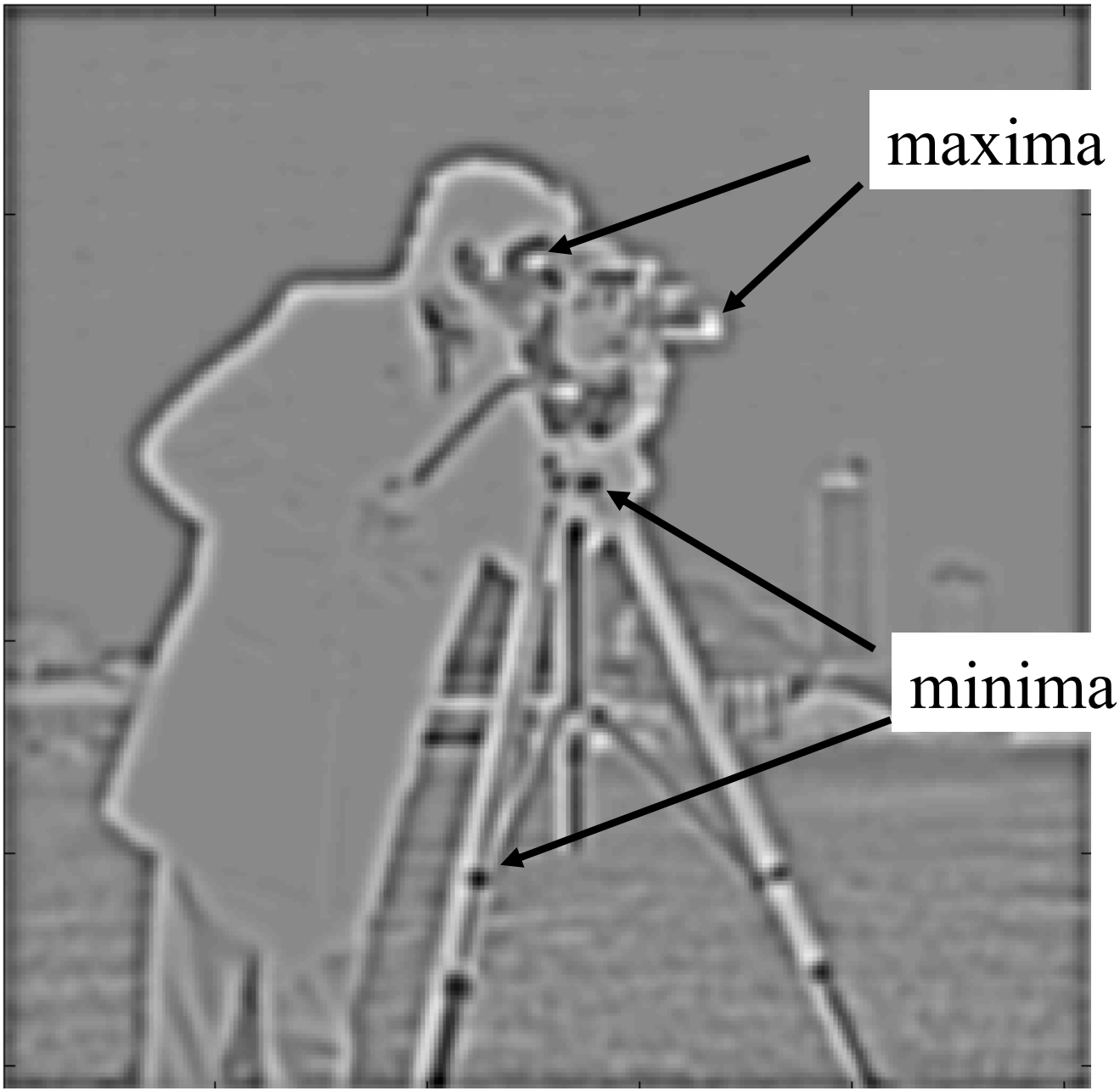


Lindeberg: “Feature detection with automatic scale selection”. International Journal of Computer Vision, vol 30, number 2, pp. 77--116, 1998.

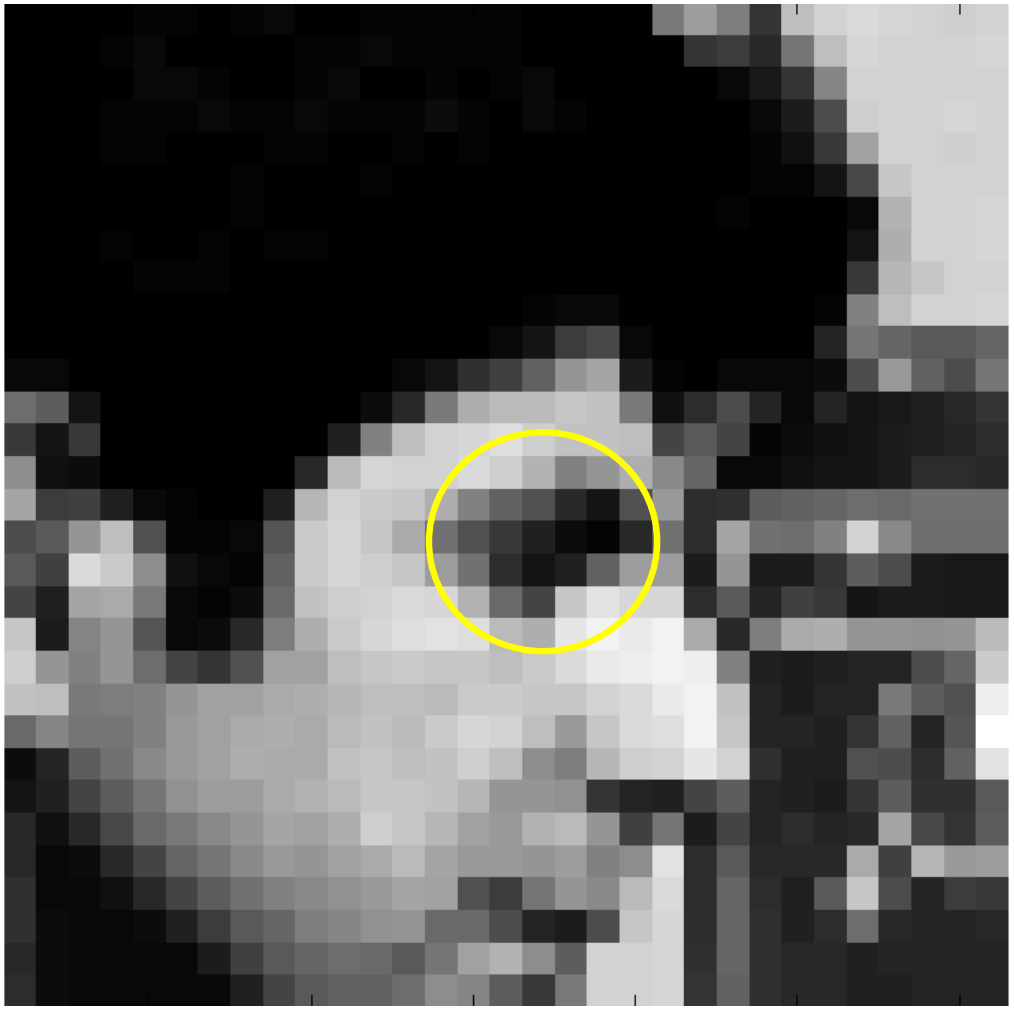


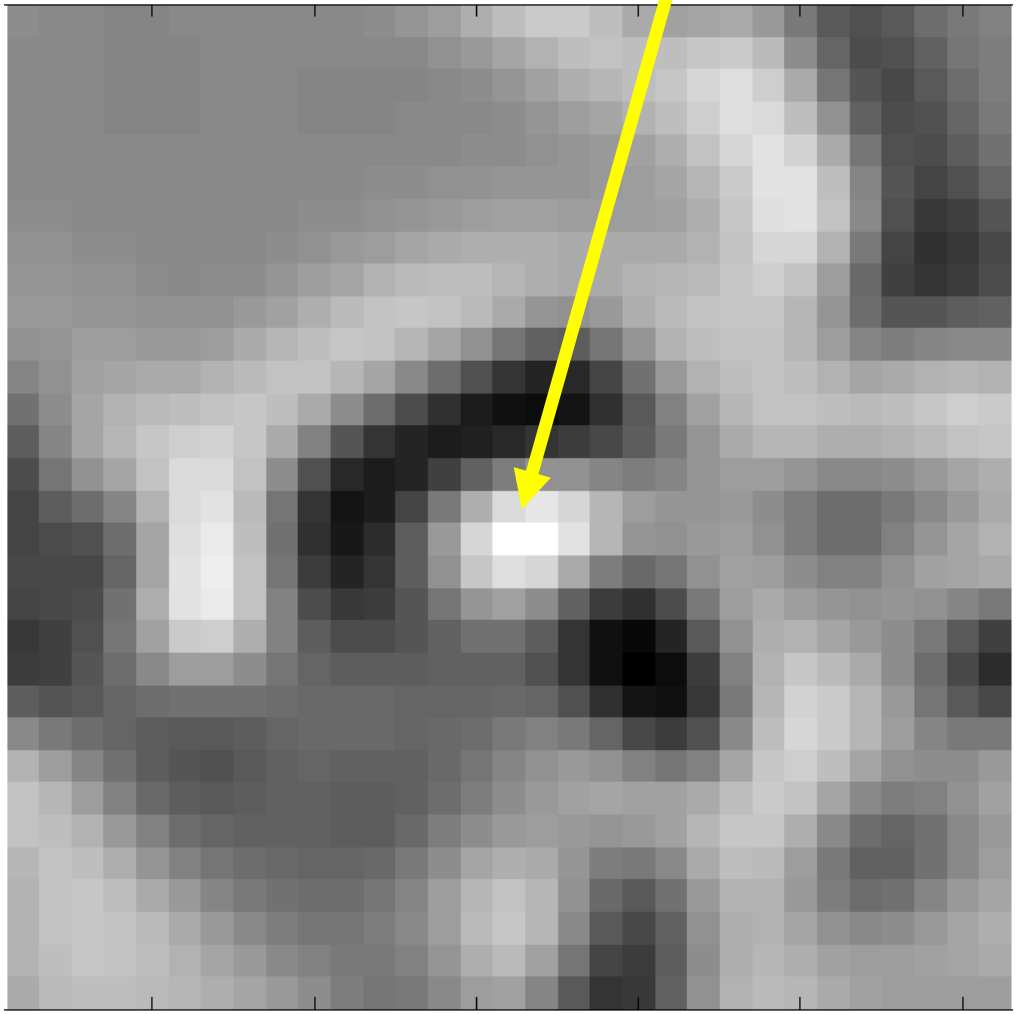
Example: LoG Extrema

LoG
sigma = 2



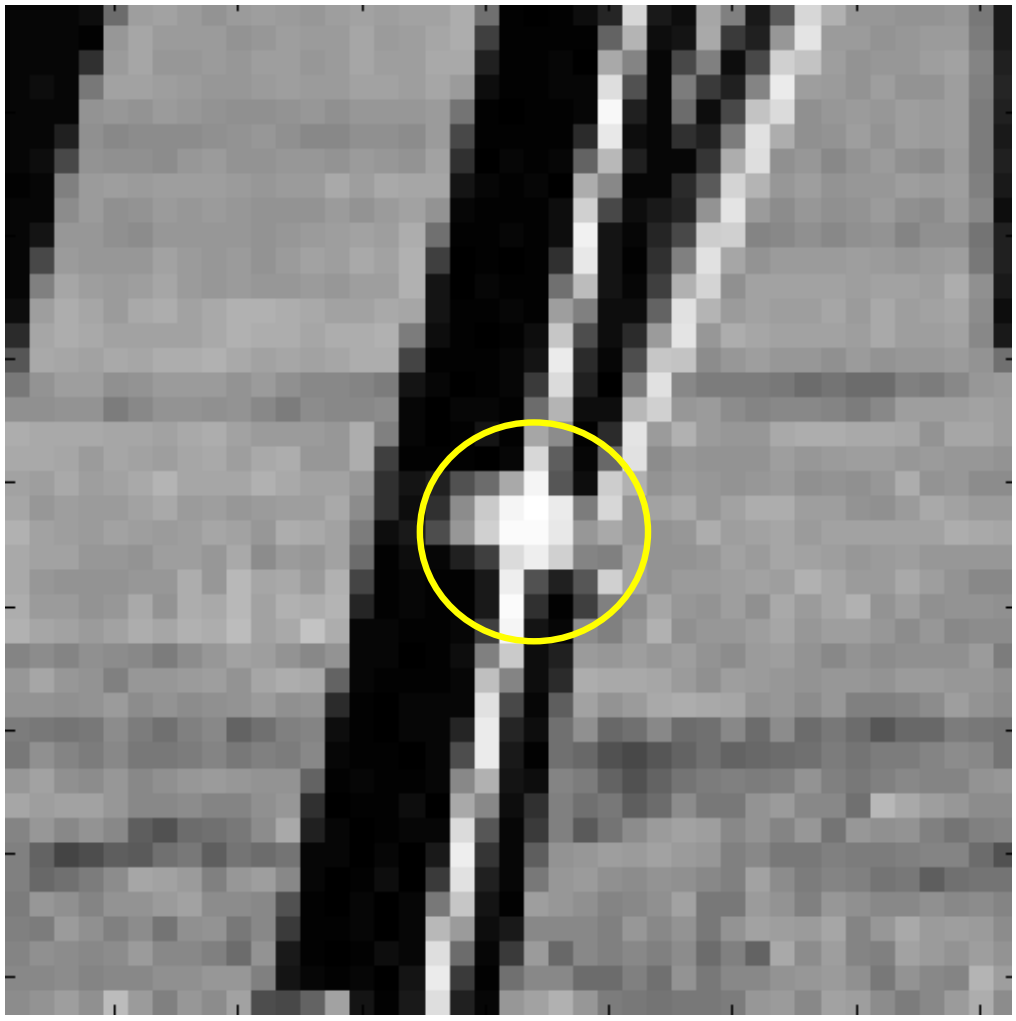
LoG Extrema, Detail

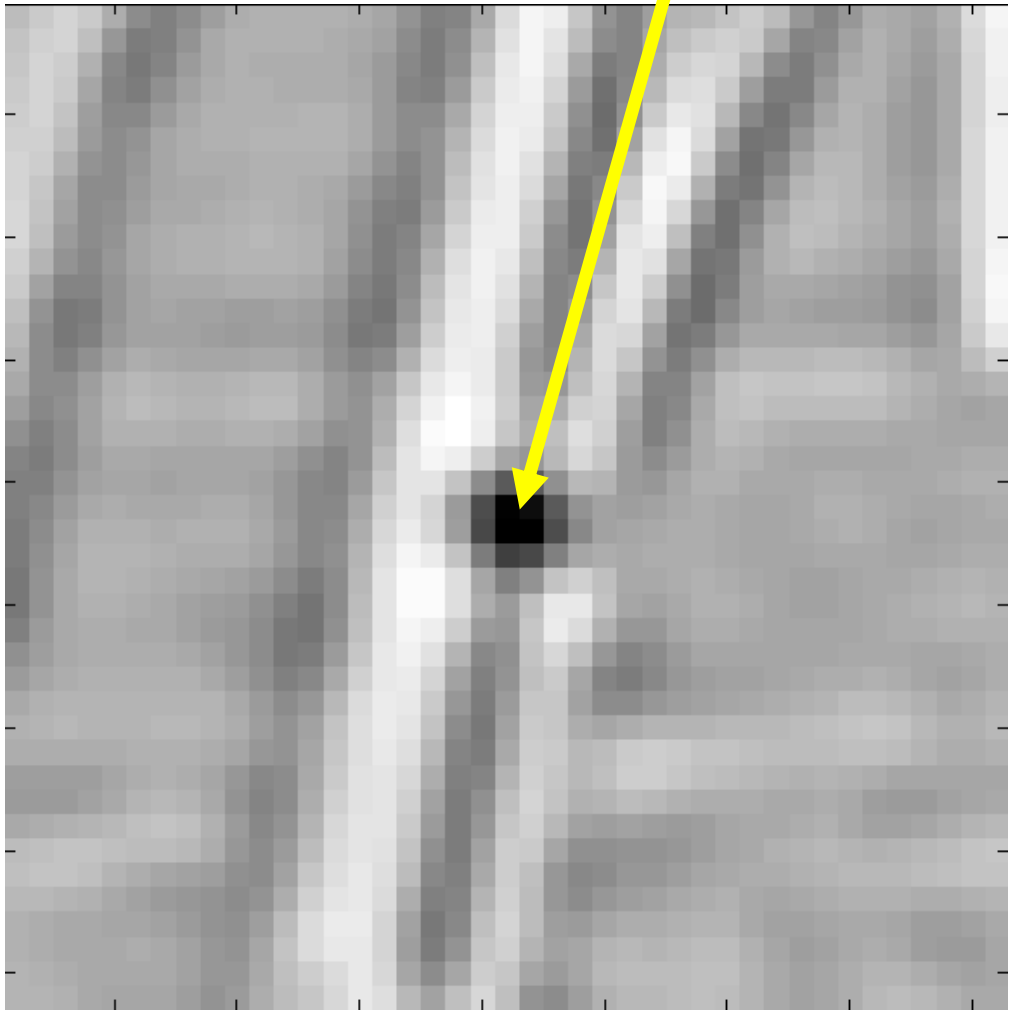




LoG sigma = 2

LoG Extrema, Detail





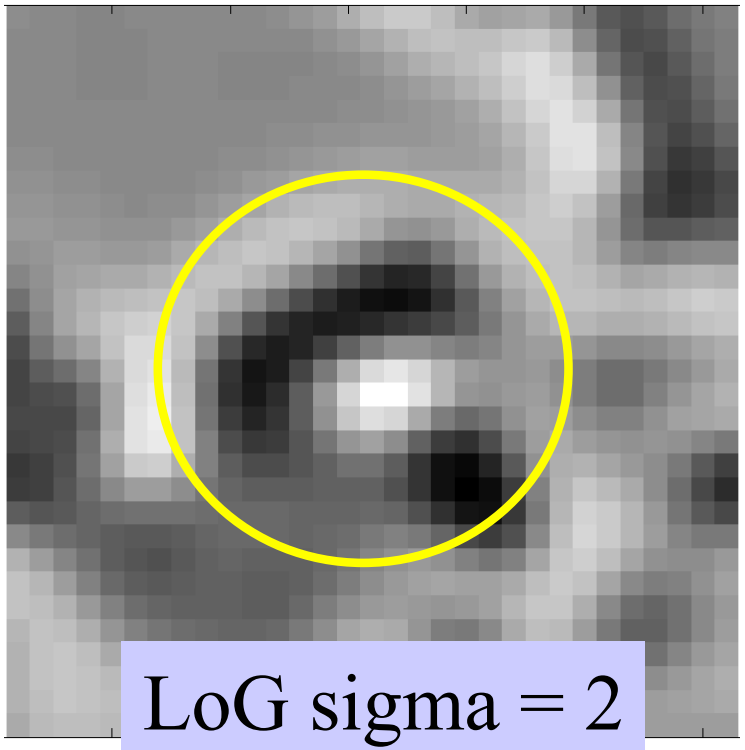
LoG sigma = 2

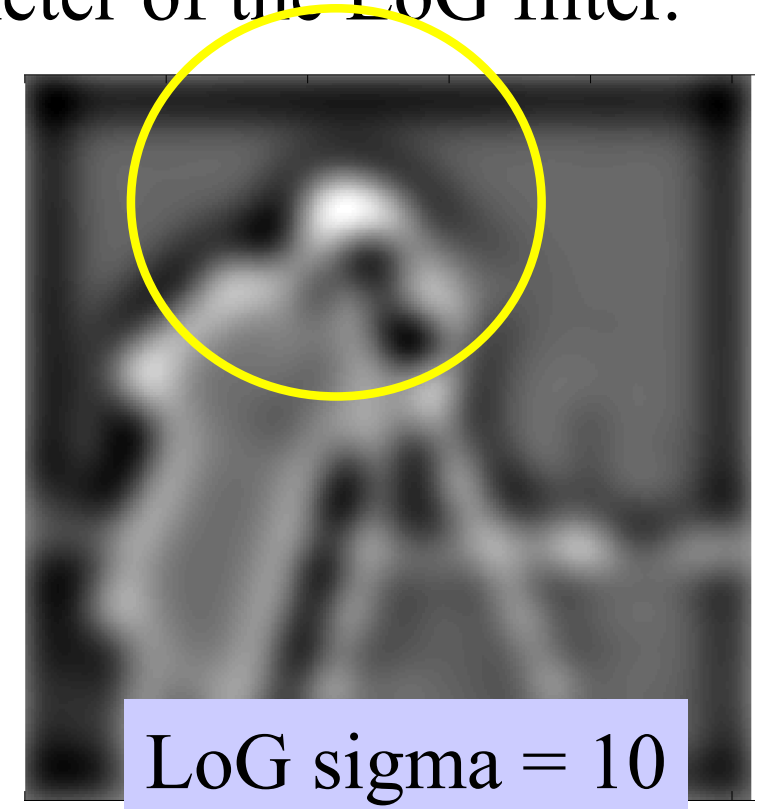
LoG Blob Finding

LoG filter extrema locates "blobs"

- maxima = dark blobs on light background
- minima = light blobs on dark background

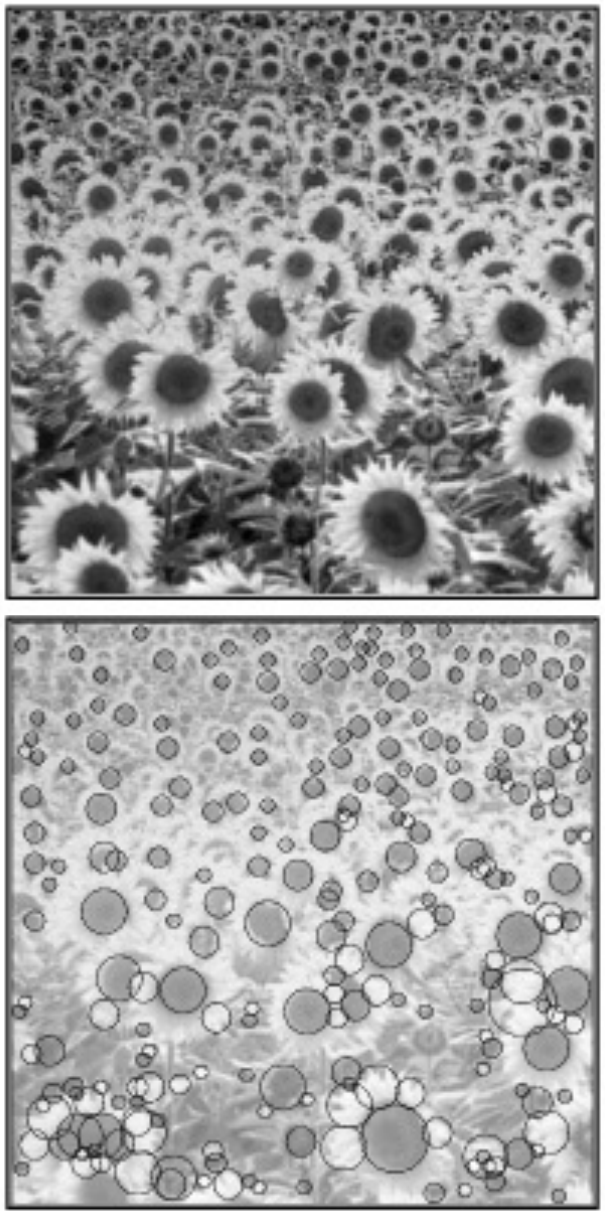
Scale of blob (size ; radius in pixels) is determined by the sigma (std.dev.) parameter of the LoG filter.

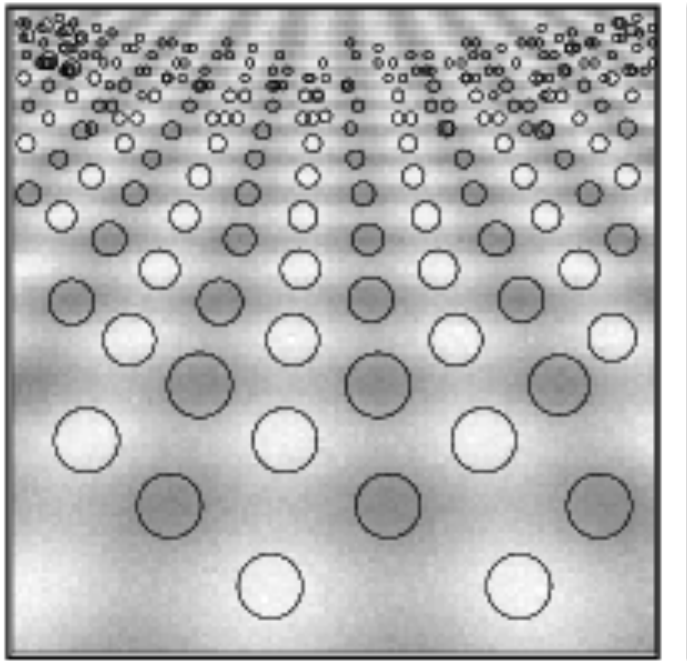




Scale Space

we will talk about this in a future lecture

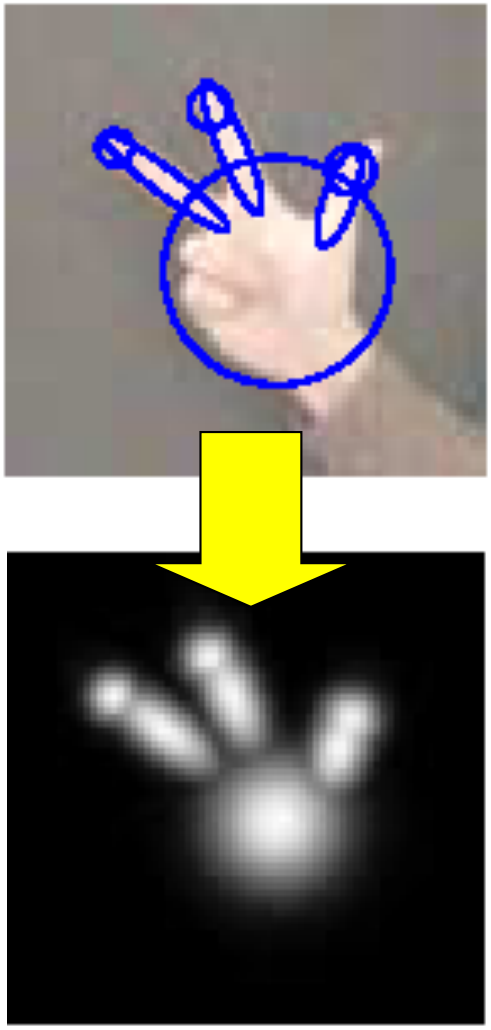




A mathematical theory and computational tool for determining the scale at which to describe scene structure at different locations in an image.

Applications: Multiscale Blob Detection

- Bretzner, Laptev, Lindeberg "Hand-gesture recognition using multi-scale colour features, hierarchical features and particle filtering", *Proc. Face and Gesture 2002*, Washington DC, pages 423--428. IEEE Computer Society Press. (PDF 159 kb)





Gesture recognition for the ultimate couch potato