Computer Vision
Edge Detection

WS 2019/2020

Prof. Dr. Simone Frintrop

Computer Vision Group, Department of Informatics
University of Hamburg, Germany

Content

- High-pass filtering and edges
- Derivatives, partial derivatives, gradient
- Edge filters: Derivatives of Gaussians, Sobel, Prewitt, Laplacian of Gaussian, Difference of Gaussian
- Filters as templates

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Edge Detection

- Why are we interested in edges?
- They often define the boundaries of objects and are important for object recognition

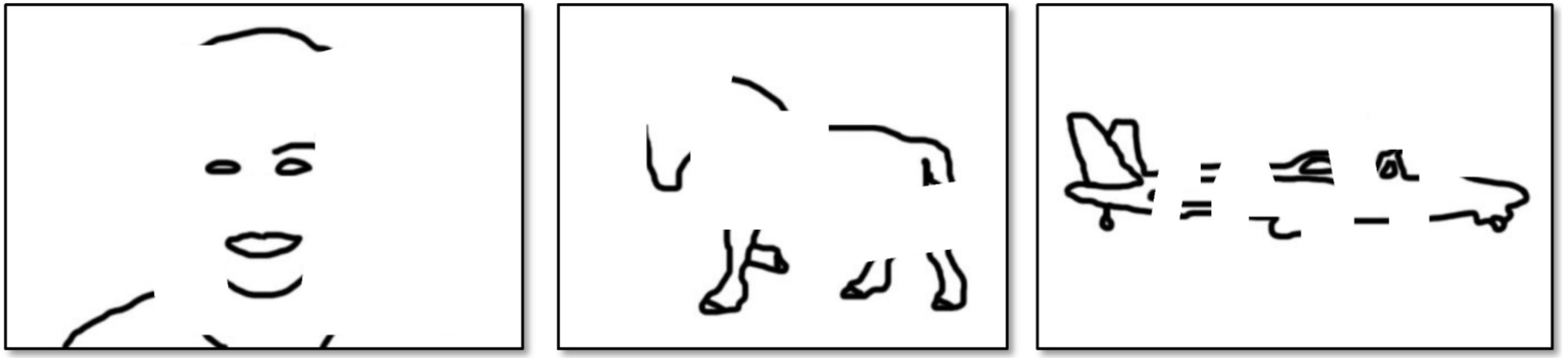
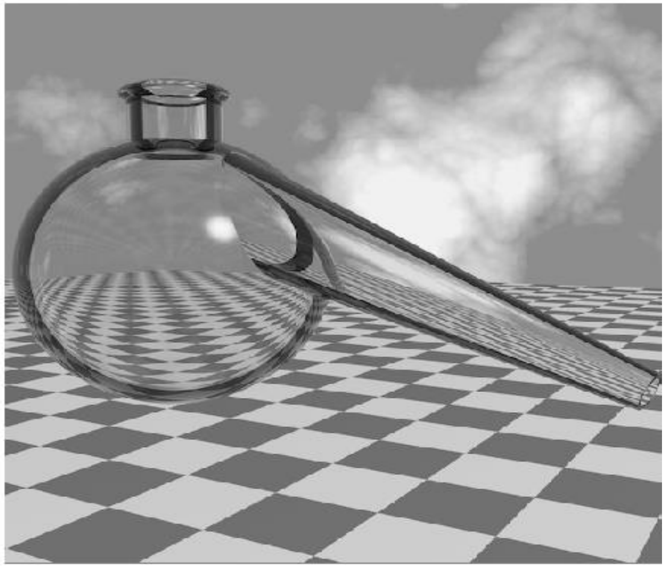


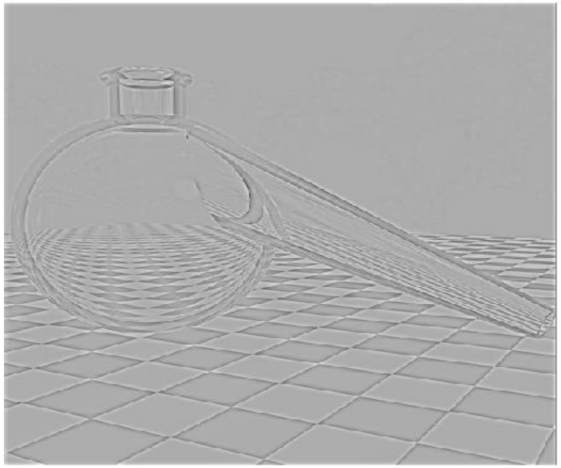
Figure from J. Shotton et al., PAMI 2007

High-pass Filtering

- High-pass filtering lets the high frequencies pass
- → Edge detection corresponds to high-pass filtering

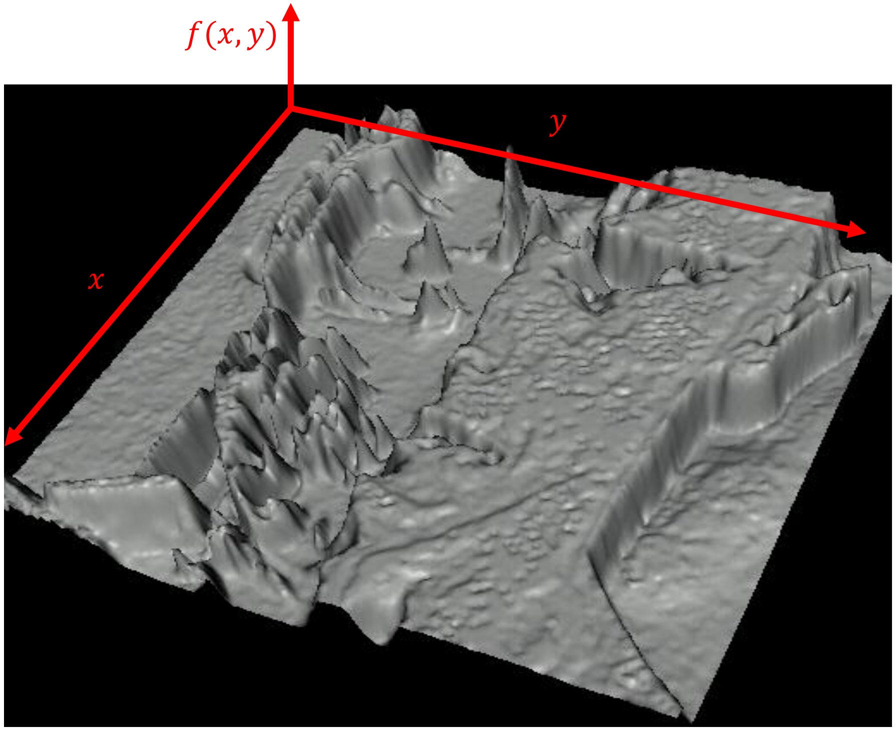


Original image



High-pass filtered

Recall: Images as Functions

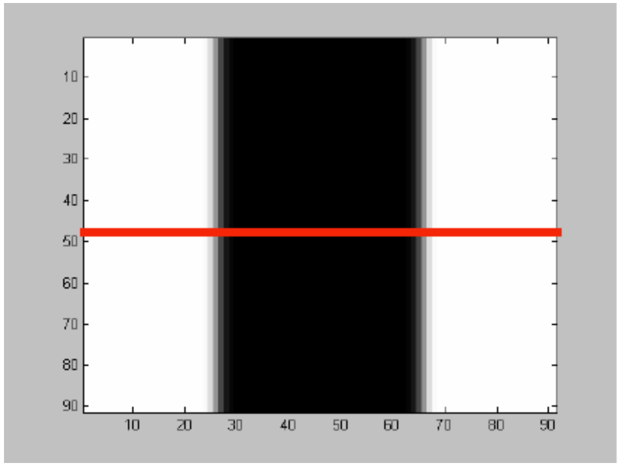


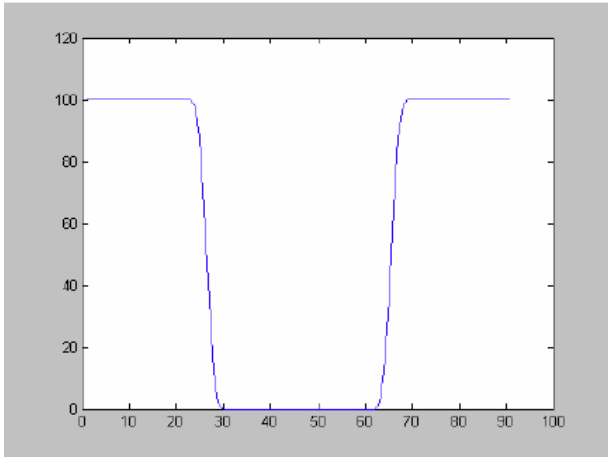
Edges look like steep cliffs

[Image: Steve Seitz]

Edge filters

- Remember: the image is a signal or function $f(x,y)$
- Edges are regions with a high slope
- What does the intensity profile of a slice through the below image at the red line look like?





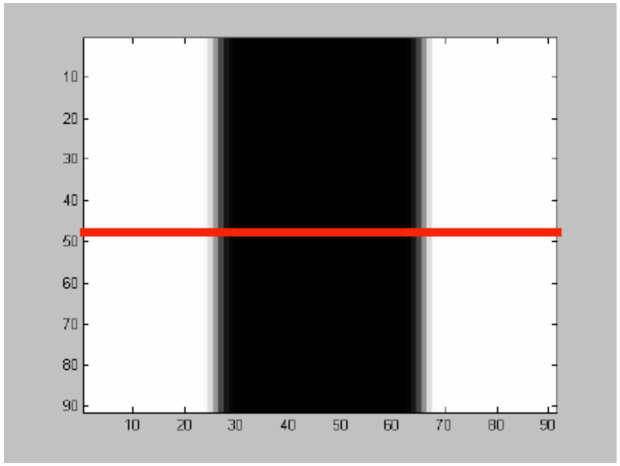
Intensity profile at red line

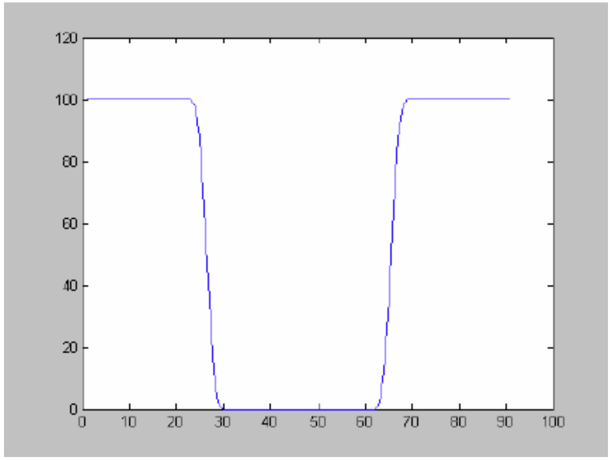
- High slopes in signals are found by derivatives
- Two approaches
  - Find maxima/minima in 1st derivative
  - Find zero-crossings in 2nd derivative

[Image: Bastian Leibe]

Question

What do the 1st and 2nd derivative of this function look like?

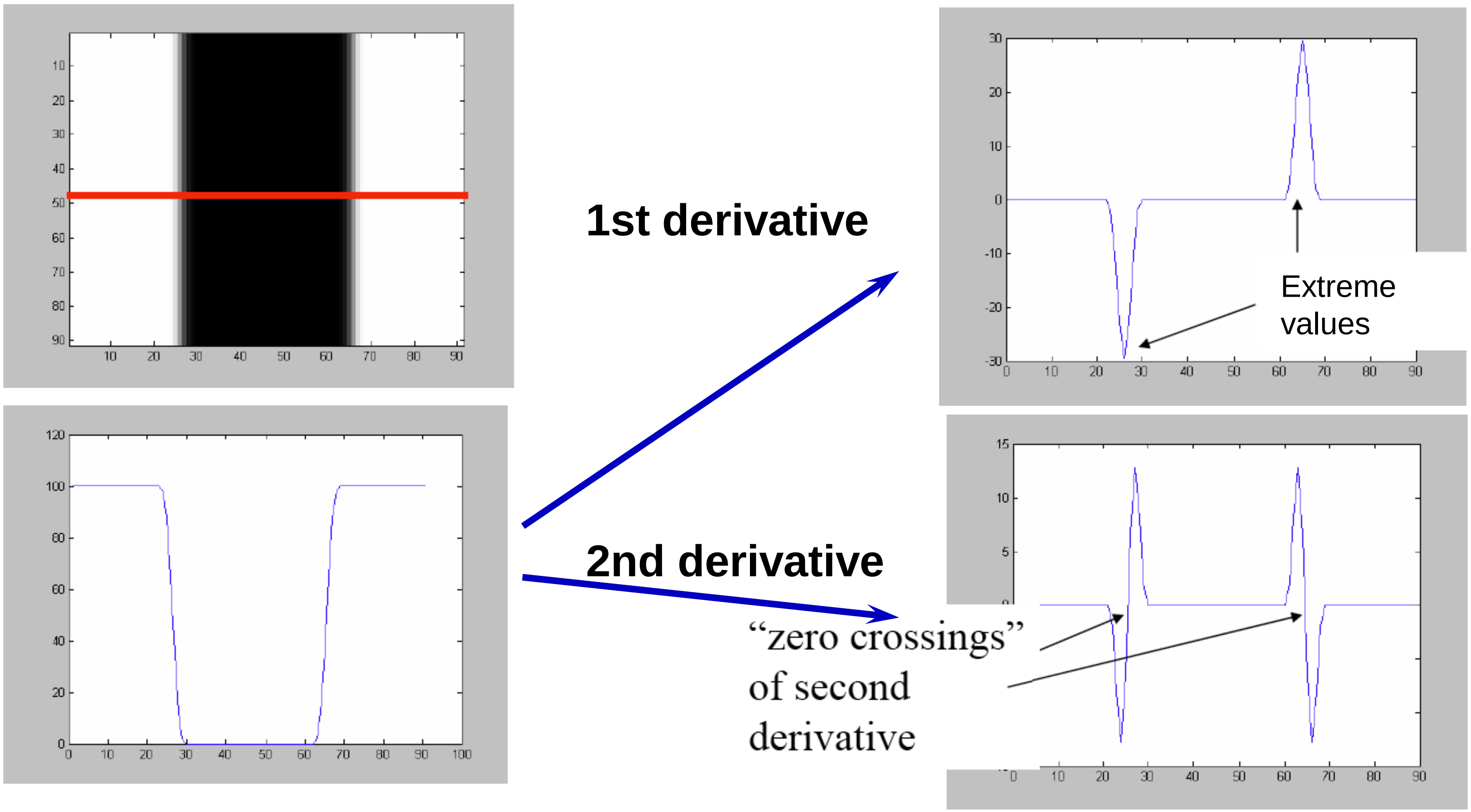




Intensity profile at red line

[Image: Bastian Leibe]

Derivatives and Edges…



Derivatives

- Derivatives are defined in terms of differences
- Approximation of 1st order derivative with finite differences:
  - Forward difference: $f'(x) = f(x+1) - f(x)$
  - Backward difference: $f'(x) = f(x) - f(x-1)$
  - Central difference: $f'(x) = f(x+1) - f(x-1)$
    or: $f'(x) = f(x+0.5) - f(x-0.5)$

current pixel

	-1	1
-1	1	
-1	0	1

Image row	**0**	**0**	**0**	**0**	**0**	**255**	**255**	**255**	**255**	**255**
1st derivative, forward difference	**0**	**0**	**0**	**0**	**255**	**0**	**0**	**0**	**0**	**0**
1st derivative, backward difference:	**0**	**0**	**0**	**0**	**0**	**255**	**0**	**0**	**0**	**0**
1st derivative, central difference:	**0**	**0**	**0**	**0**	**255**	**255**	**0**	**0**	**0**	**0**

Derivatives

- 1st order derivative (central difference):

$$f'(x) = f(x+0.5) - f(x-0.5)$$

- 2nd order derivative:

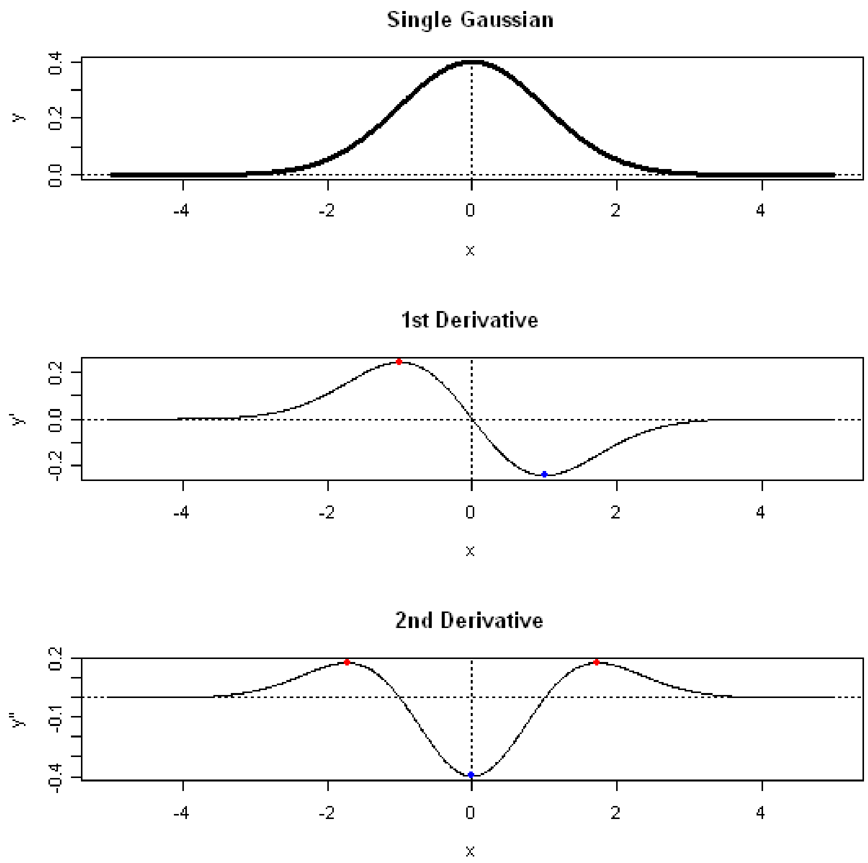
$$\begin{aligned} f''(x) &= (f(x+0.5) \quad - f(x-0.5))' \\ &= f'(x+0.5) \quad - f'(x-0.5) \\ &= \big(f(x+1) - f(x)\big) - (f(x) \; - f(x-1)) \\ &= f(x+1) \; - f(x) \; - f(x) \; + f(x-1)) \\ &= f(x+1) \quad - 2f(x) \quad + f(x-1) \end{aligned}$$

- What does the kernel look like?
- Kernel:

1	-2	1

Derivatives

- Derivatives of a Gaussian (1D)



Derivatives

Notation:

First order derivative:

$$f'(x) = \frac{d}{dx} f(x) = \frac{df}{dx} = \frac{\partial}{\partial x} f(x) = D_x f(x) = D_x$$

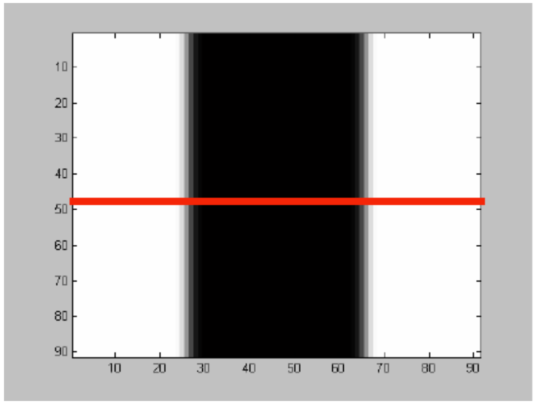
Second order derivative:

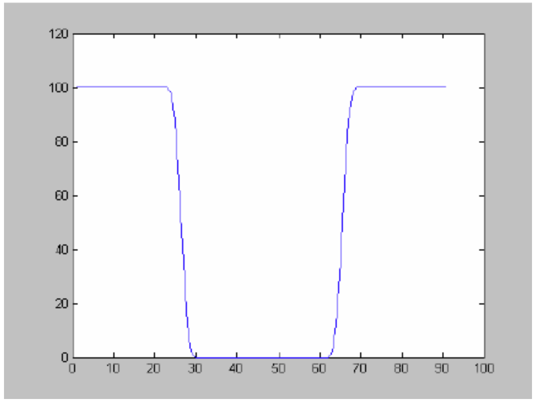
$$f''(x) = \frac{d^2}{dx^2} f(x) = \frac{d^2 f}{dx^2} = \frac{\partial^2}{\partial x^2} f(x) = D_x^2 f(x) = D_{xx}$$

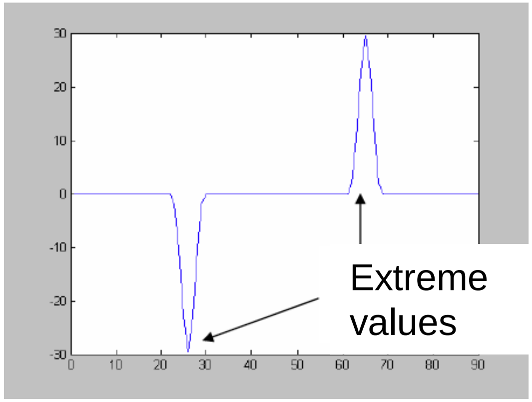
Edge Detection

First ideas for edge detection with 1st derivative for a 1D signal without noise:

1. Take the derivative of each point (apply central difference filter)
2. Search for local extrema of the derivative value







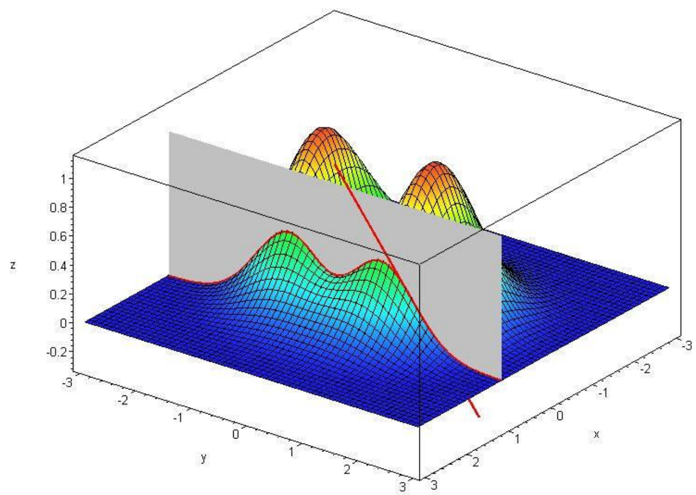
[Image: Bastian Leibe]

From 1D to 2D

Derivatives in images

Partial derivatives of function $f(x, y)$:

- In x-direction: $$\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y)$$
- In y-direction: $$\frac{\partial f}{\partial y} = f(x, y+1) - f(x, y)$$



y

x

What do the corresponding filter kernels look like?

Derivatives in images

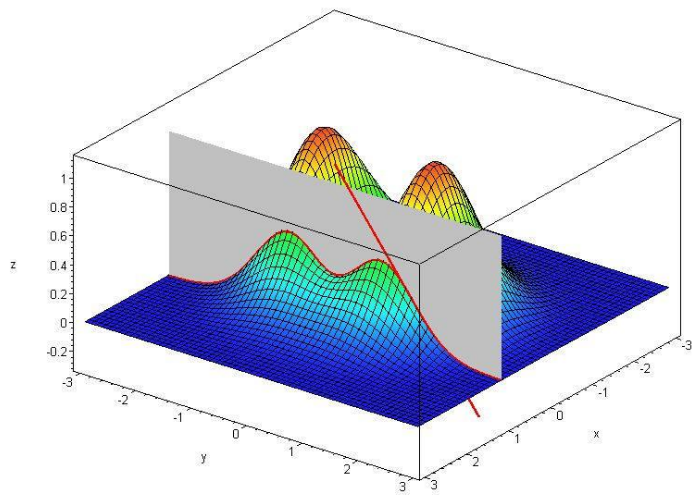
Partial derivatives of function $f(x, y)$:

- In x-direction: $$\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y)$$

-1
1

- In y-direction: $$\frac{\partial f}{\partial y} = f(x, y+1) - f(x, y)$$

-1	1



y

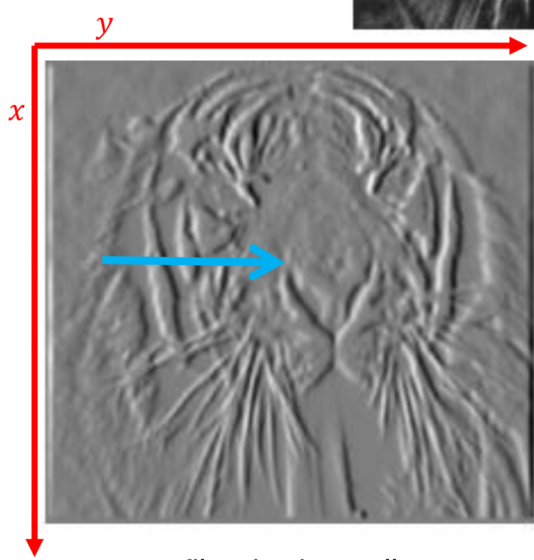
x

What do the corresponding filter kernels look like?

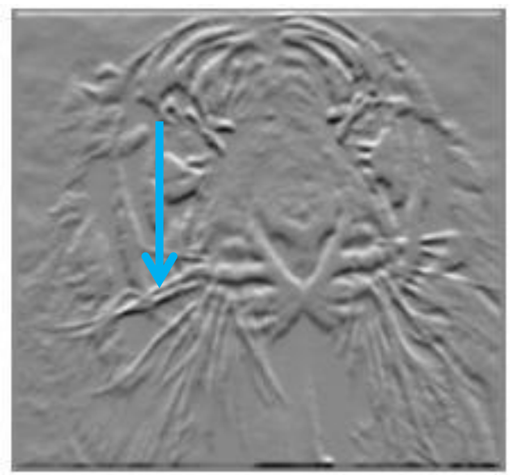
Partial Derivatives of an Image

$$\frac{\partial f(x,y)}{\partial y}$$

-1	1



Move filter horizontally,
Get vertical edges



Move filter vertically,
Get horizontal edges

$$\frac{\partial f(x,y)}{\partial x}$$

-1
1

Derivatives in images

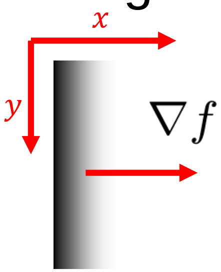
Partial derivatives of function $f(x, y)$:

- In x-direction: $$\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y)$$
- In y-direction: $$\frac{\partial f}{\partial y} = f(x, y+1) - f(x, y)$$
- Together, the partial derivatives form the gradient:
$$\nabla f = grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$
The gradient points into the direction of strongest increase of f

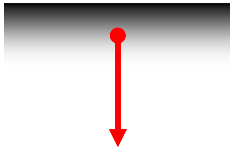
Image Gradient

Gradients in images: $$\nabla f = grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]^T$$

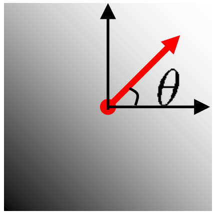
- The gradient points in the direction of most rapid intensity change



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]^T$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]^T$$



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]^T$$

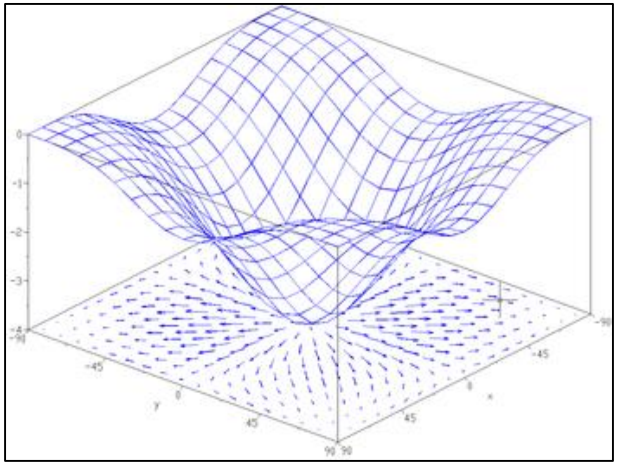


Image Gradient

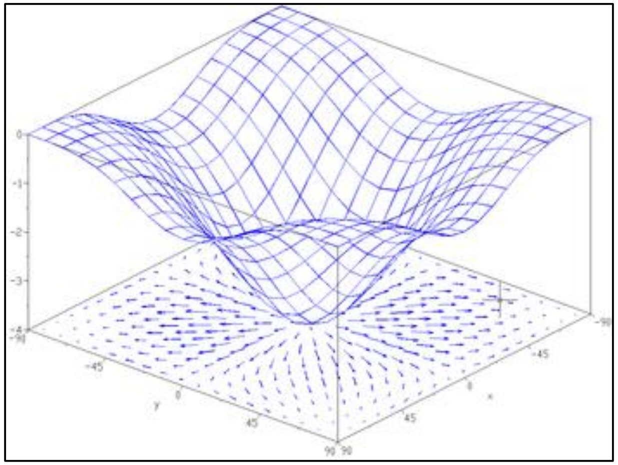
Gradients in images: $\nabla f = grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]^T$

- The gradient direction (orientation) is given by:

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

- The edge strength is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$



Edge Detection

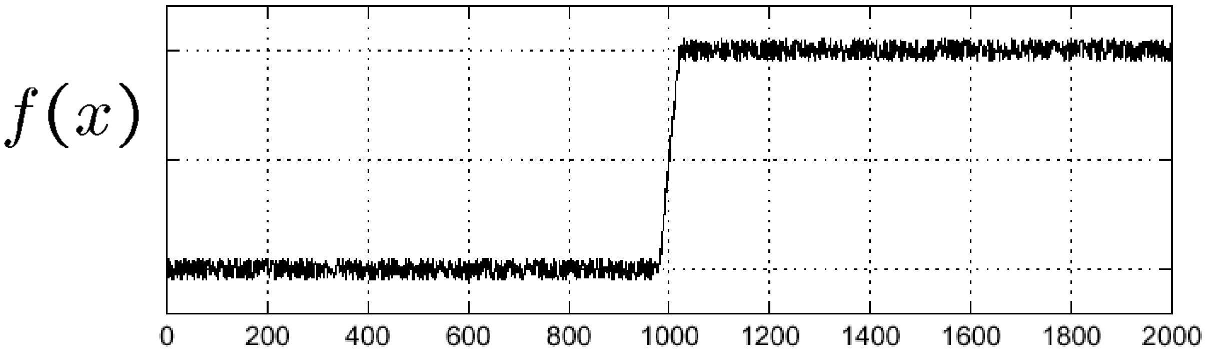
First ideas for edge detection with 1st derivative for a 2D signal without noise:

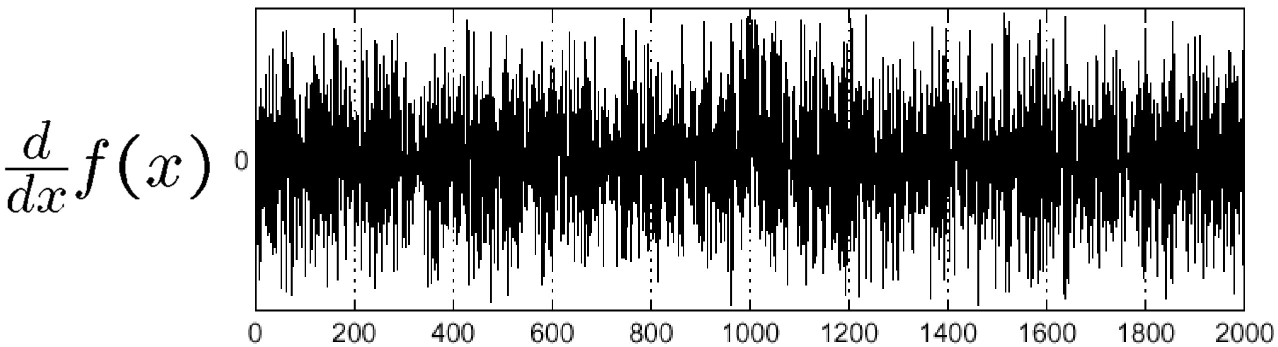
1. Take the partial derivatives in x and y direction
2. Compute the gradient magnitude $\|\nabla f\| = \sqrt{(\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2}$
3. Threshold on the gradient magnitude

Effect of Noise

Noise affects the derivatives strongly:

Intensity function of a row of a noisy image:





Where is the edge?

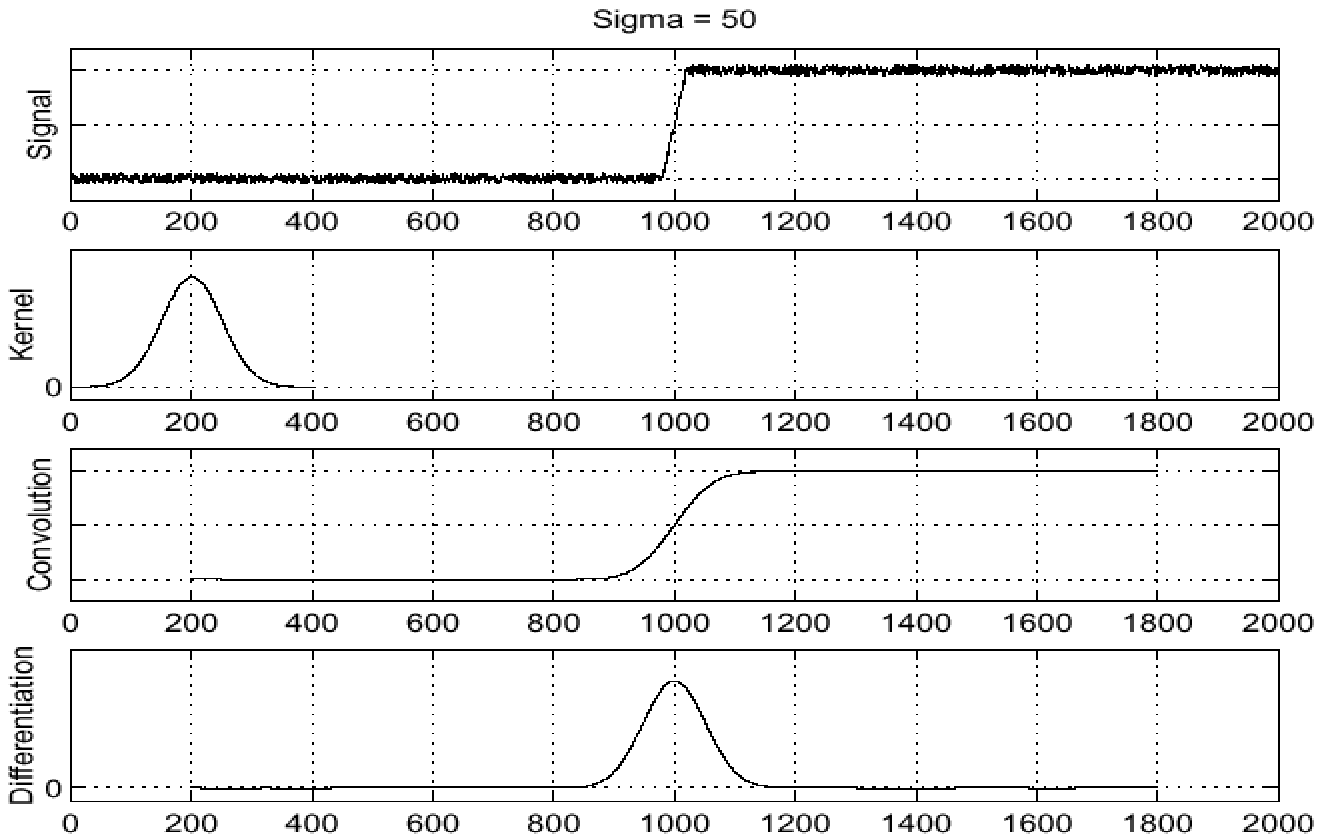
Solution: Smooth First

f

h

$h \star f$

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



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

Derivative Theorem of Convolution

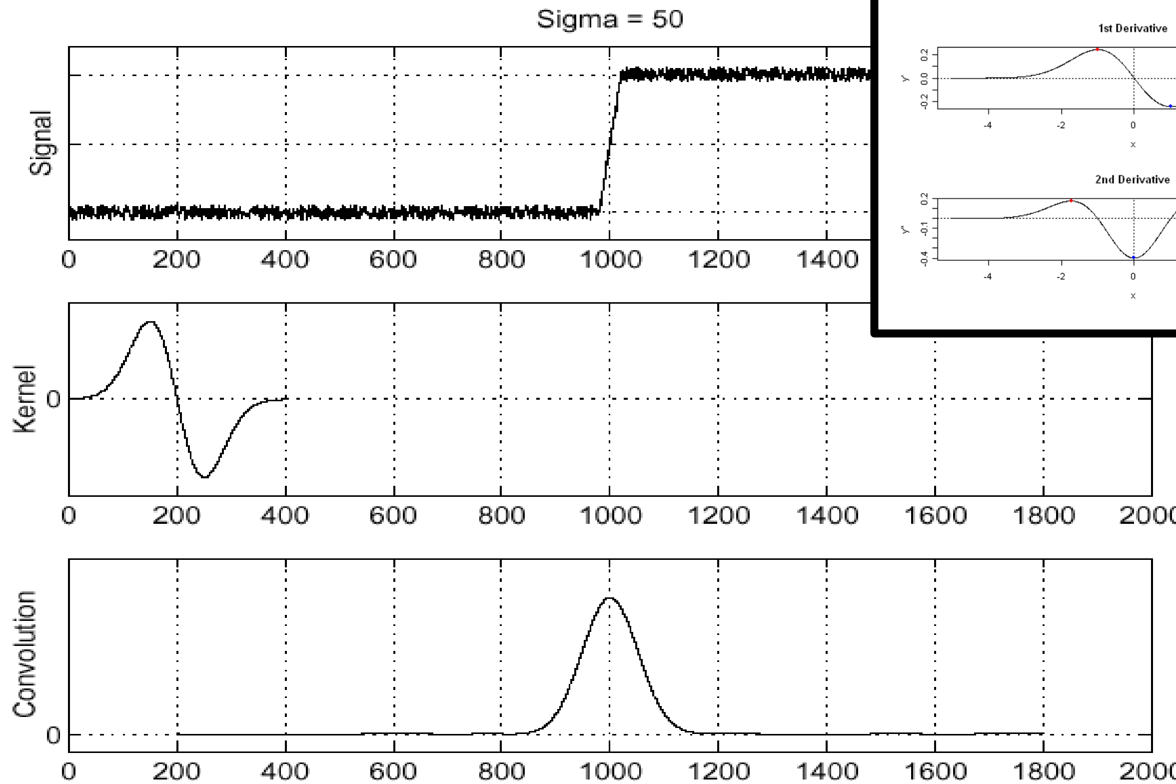
Remember: $\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$

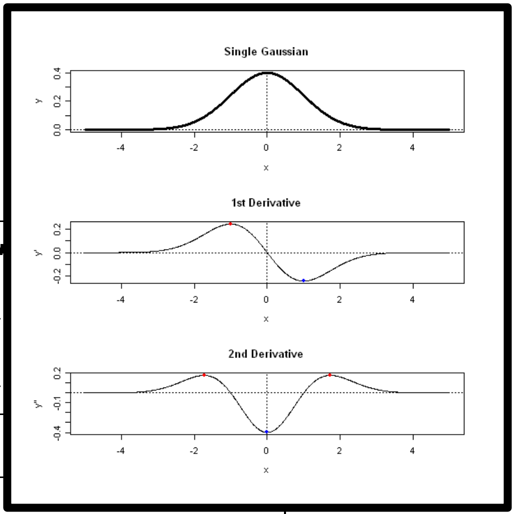
(Differentiation property of convolution)

f

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

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





Derivative of Gaussian Filters

The Gaussian derivative in 2D:

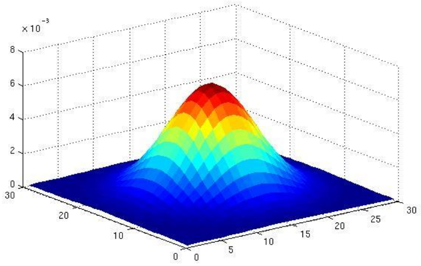
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

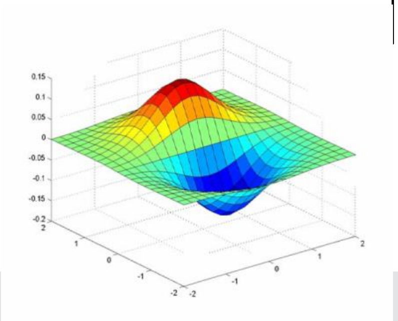
f: image

h: Gaussian,

$\frac{\partial}{\partial x}$ derivative in x direction

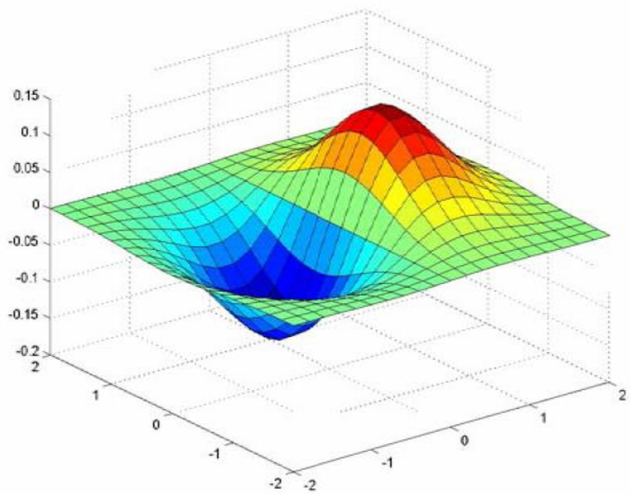
$$\begin{bmatrix} -1 \\ 1 \end{bmatrix} *$$

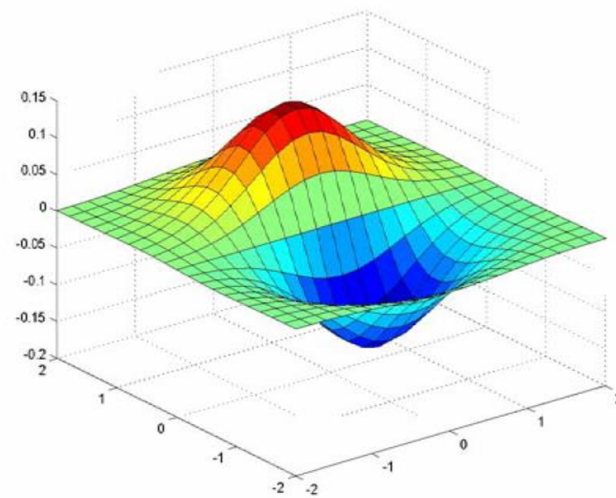




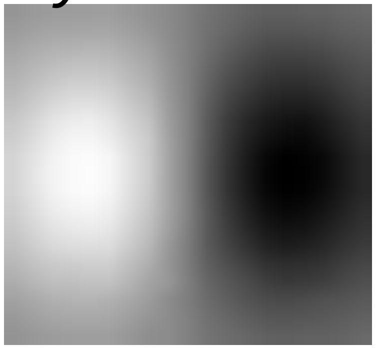
Gaussian derivative in x-direction

Derivative of Gaussian Filters

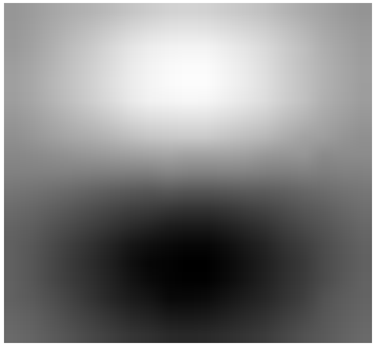




y-direction



x-direction



[Image: Svetlana Lazebnik]

Derivative of Gaussian Filters

- Approximation of 1st derivative of Gaussian with the Sobel filter:

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

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

- or even simpler with the Prewitt operator:

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

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

Edge Detection

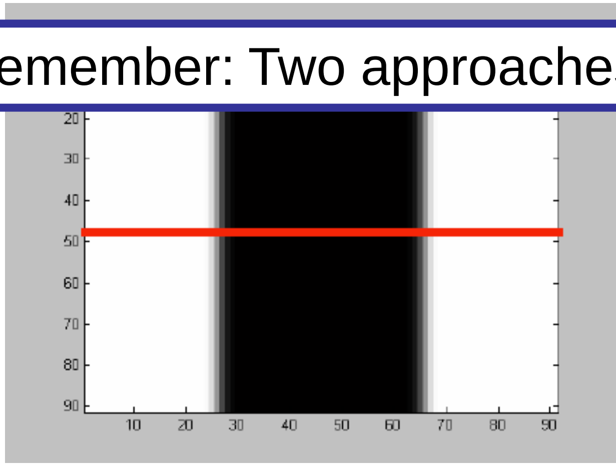
So, to summarize the 1st derivative approach for edge detection for a 2D signal with noise:

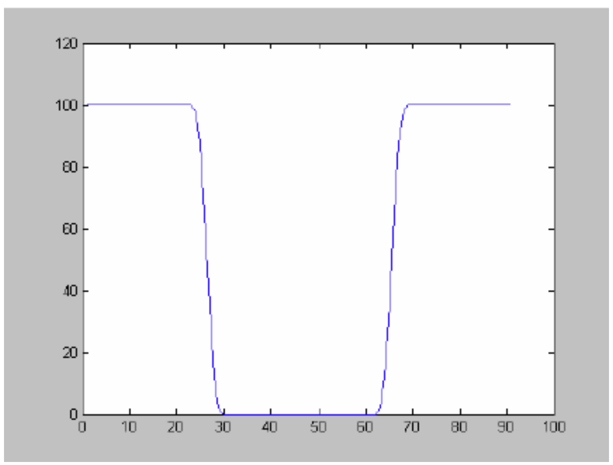
1. Smooth the image
2. Take the partial derivatives in x and y direction
3. Compute the gradient magnitude
4. Threshold on the gradient magnitude

1\. and 2. can be combined into directly smoothing with derivatives of smoothing kernels, e.g., with the Sobel filters

Derivatives and Edges…

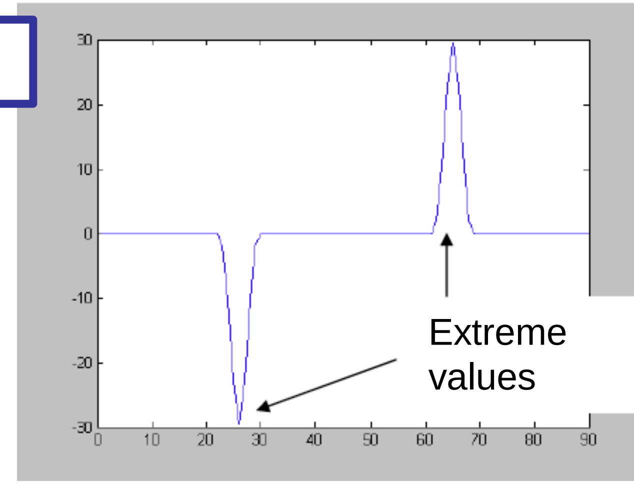
Remember: Two approaches for edge detection

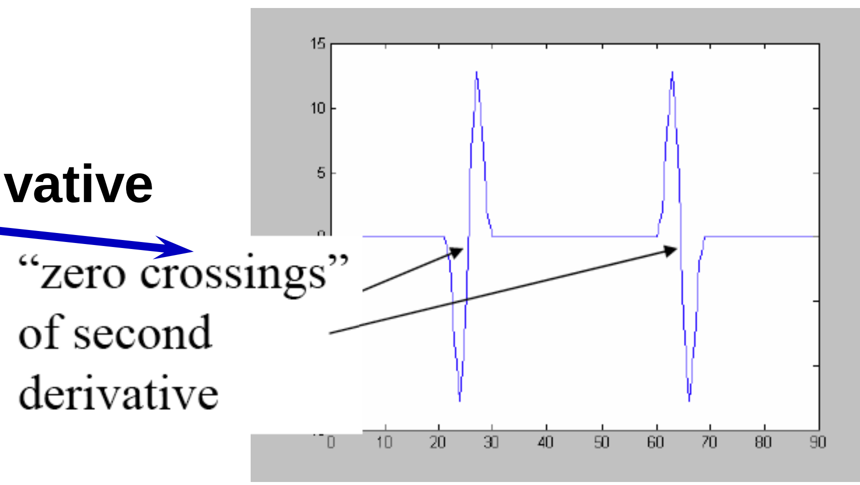




1st derivative

2nd derivative





2nd order derivatives

Remember the 1D case, 2nd order derivative:

$$\frac{\partial^2}{\partial x^2} f(x) = f(x+1) - 2f(x) + f(x-1)$$

1	-2	1

Extended to a 2nd order partial derivative (2D case):

1
-2
1

$$\frac{\partial^2}{\partial x^2} f(x,y) = f(x+1,y) - 2f(x,y) + f(x-1,y)$$

$$\frac{\partial^2}{\partial y^2} f(x,y) = f(x,y+1) - 2f(x,y) + f(x,y-1)$$

1	-2	1

The **Laplacian** is a 2nd order differential operator

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

$$= f(x+1,y) - 2f(x,y) + f(x-1,y) + f(x,y+1) - 2f(x,y) + f(x,y-1)$$

$$= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

2^{nd} order derivatives

The **Laplacian**

$$\nabla^2 f(x,y) = f(x,y+1) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

What does the corresponding filter mask look like?

2^{nd} order derivatives

The **Laplacian**

$$\nabla^2 f(x,y) = f(x,y+1) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

What does the corresponding filter mask look like?

0	1	0
1	-4	1
0	1	0

Note that this results from

1	-2	1

+

1
-2
1

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

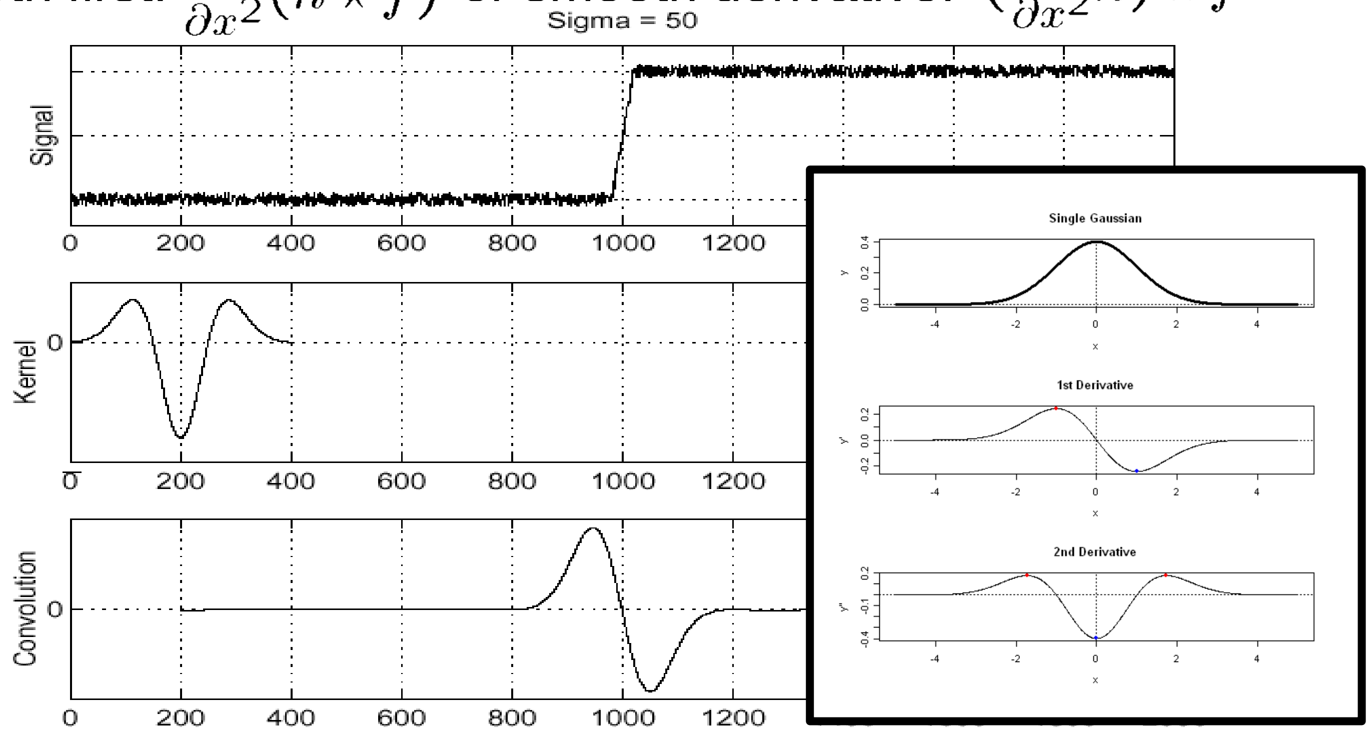
Laplacian of Gaussian (LoG)

- Edge detection with 2nd order derivatives.
- Again problem with noisy signals.
- Solution: smooth first: $\frac{\partial^2}{\partial x^2}(h \star f)$ or smooth derivative: $(\frac{\partial^2}{\partial x^2}h) \star f$

$$f$$

$$\frac{\partial^2}{\partial x^2}h$$

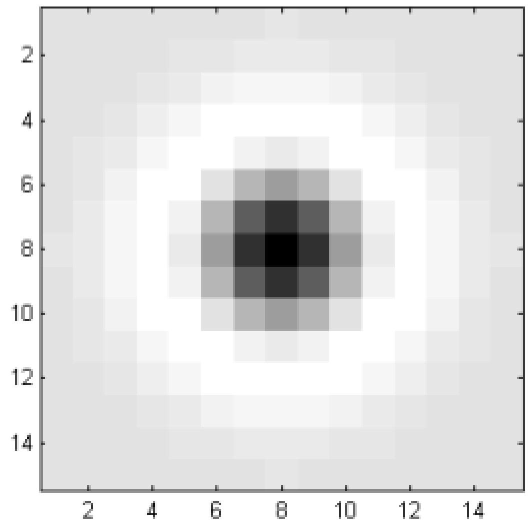
$$(\frac{\partial^2}{\partial x^2}h) \star f$$

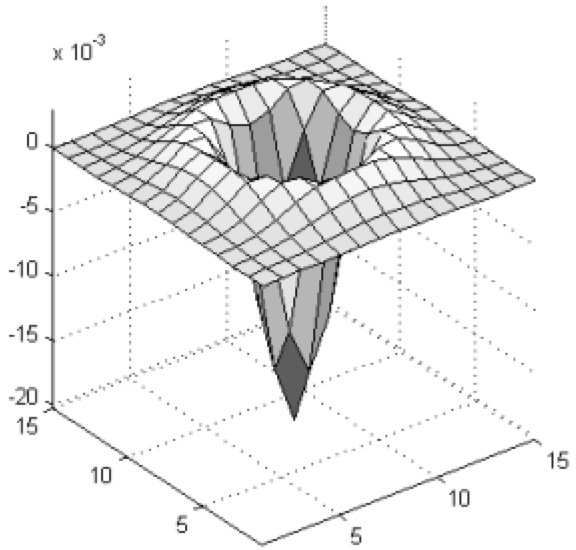


Edge is at zero-crossings of bottom graph

Laplacian of Gaussian (LoG)

- 2D Laplacian of Gaussian filter:

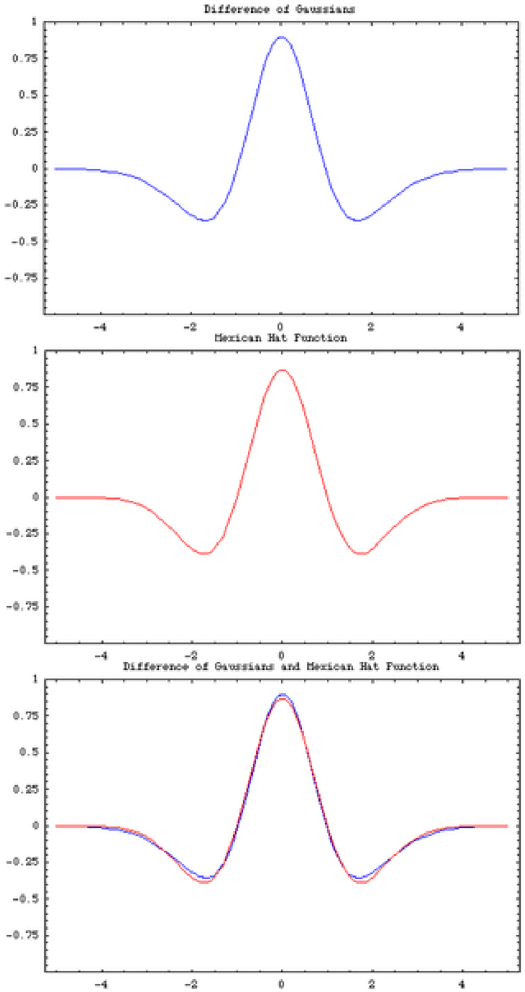




[Image: http://suinotes.wordpress.com/2010/05/27/generic-multivariate-gaussian-kernel-in-any-derivative-order/]

DoG versus LoG

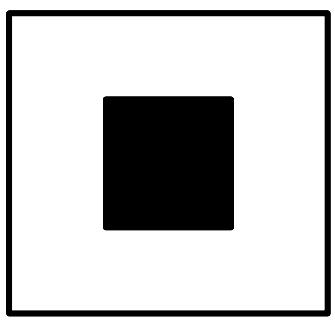
- The Laplacian (Mexican hat filter) can be approximated by the Difference of Gaussian filter (DoG)
- DoG is much cheaper, since it is separable
- This operator corresponds well to cells in the human visual system, e.g. retinal ganglion cells (See Lecture “Computer Vision 2)



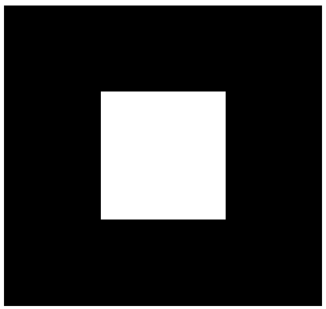
Laplacian of Gaussian (LoG)

A (very) simple approximation to the LoG:

- The DoB filter (Difference of Boxes) (or just center-surround filter):
- Computes difference between two mean filters of different sizes



surround - center



center - surround

Edge Detection

The 2nd derivative approach for edge detection for a 2D signal with noise:

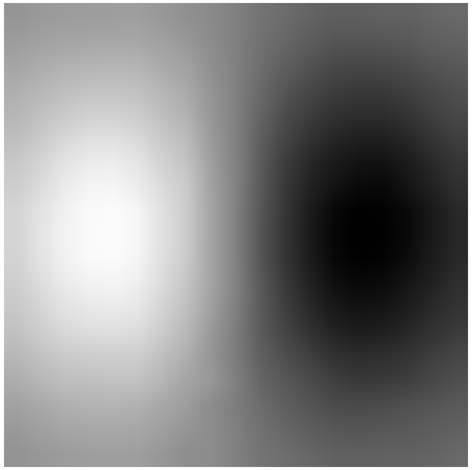
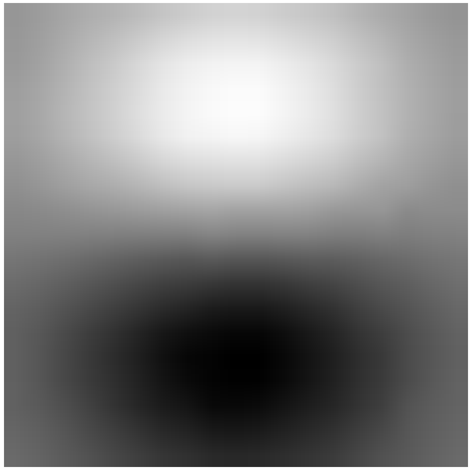
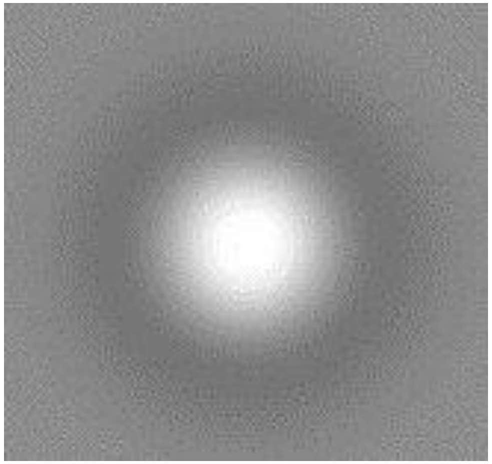
1. Smooth the image
2. Apply the Laplacian operator
3. Search for zero crossings
   (zero crossing at x: at least two opposing neighbors have different signs)

1. and 2. can be combined into directly smoothing with a 2nd derivative of a smoothing kernel, e.g., a Laplacian of Gaussian

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

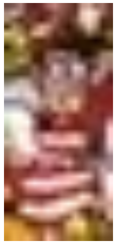
Filters as Templates

Filters can be used as templates:
They look like the effects they are intended to find

- positive values: white
- negative values: black
- zero values: gray

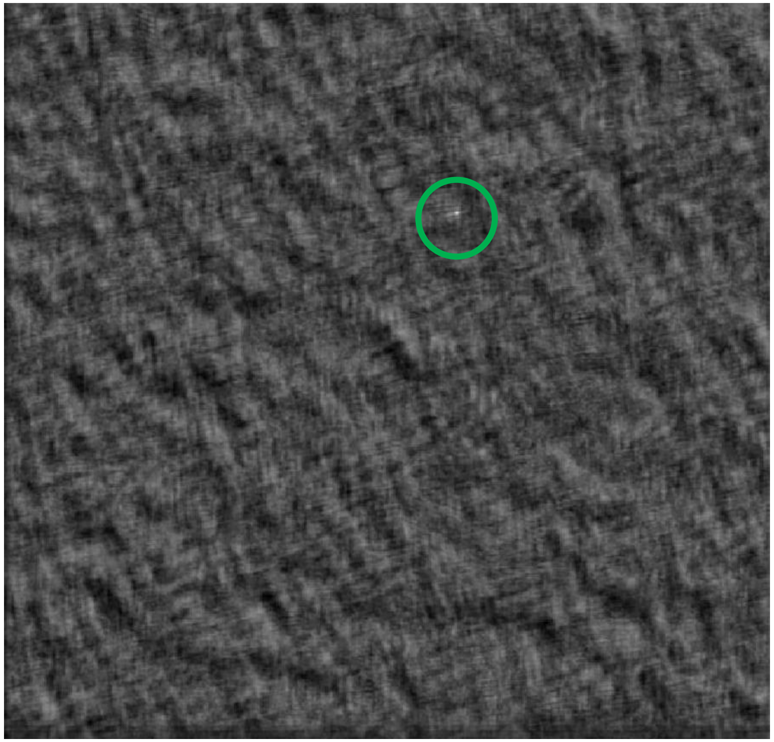
Where's Waldo?



Scene

Where's Waldo?

Detected template



Correlation map

In practical applications this usually does not work well, since the template never fits perfectly to the test data

Summary

- Edge detection with
  - 1^{st} order derivatives: smooth image
    compute partial derivatives D_x and D_y
    compute gradient magnitude
    threshold on gradient magnitude
  - 2^{nd} order derivatives: smooth image
    apply Laplacian operator
    search for zero crossings
- Derivatives of Gaussian filters
- Laplacian of Gaussian filters
- Filters as templates

Primary Literature

- Szeliski: parts from chapter 3 and 4
- Gonzalez/Woods, 4th edition: parts from chapter 3

Secondary Literature

- Shotton, Blake, Cipolla: “Multi-scale categorial object recognition using contour fragments”, PAMI 2007