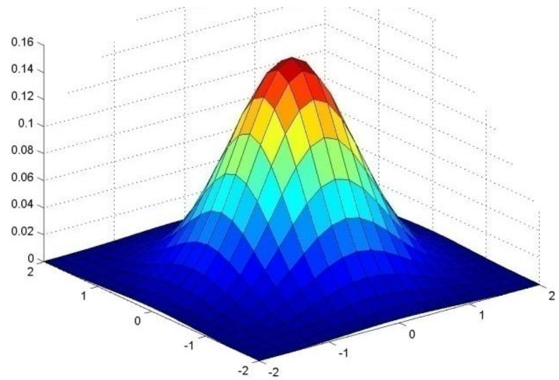
CS4670: Computer Vision

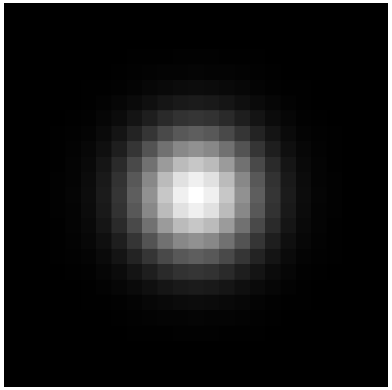
Noah Snavely

Lecture 2: Convolution and edge detection

From Sandlot Science

Gaussian Kernel

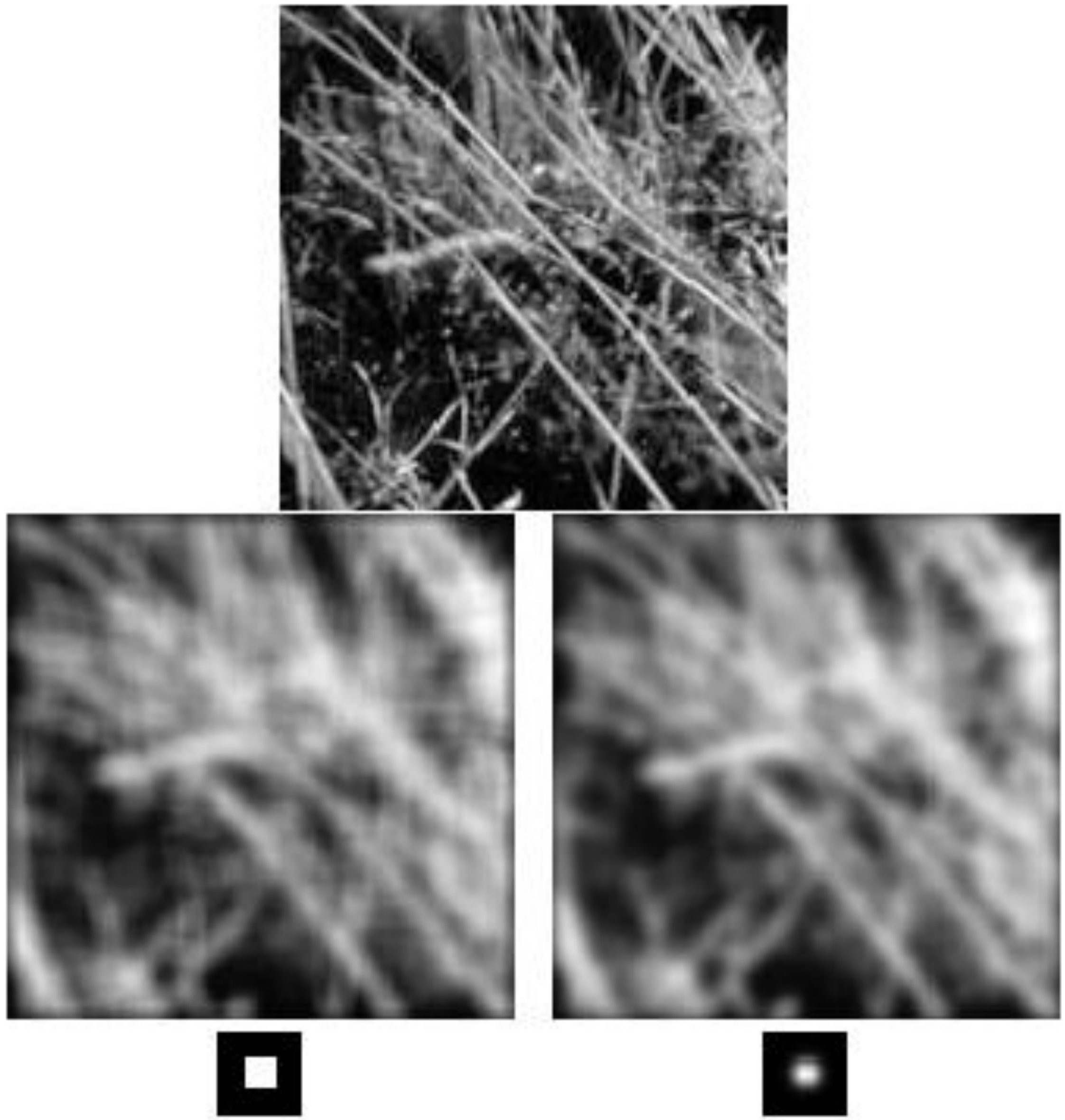




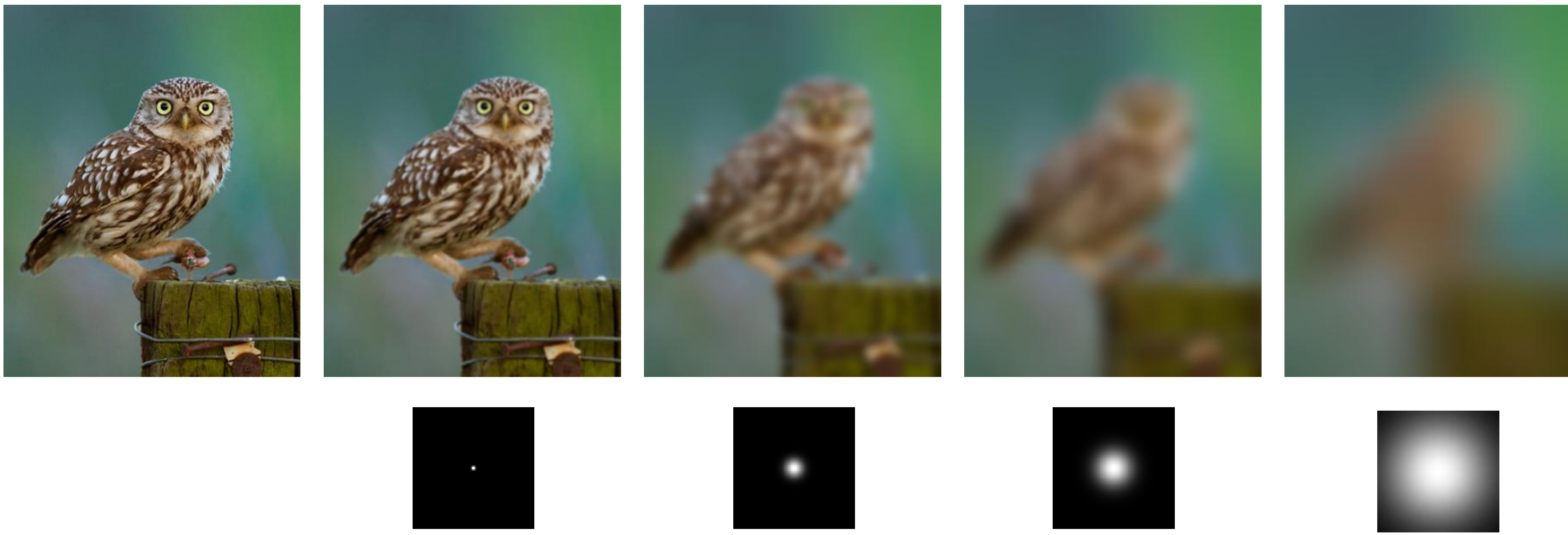
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Source: C. Rasmussen

Mean vs. Gaussian filtering



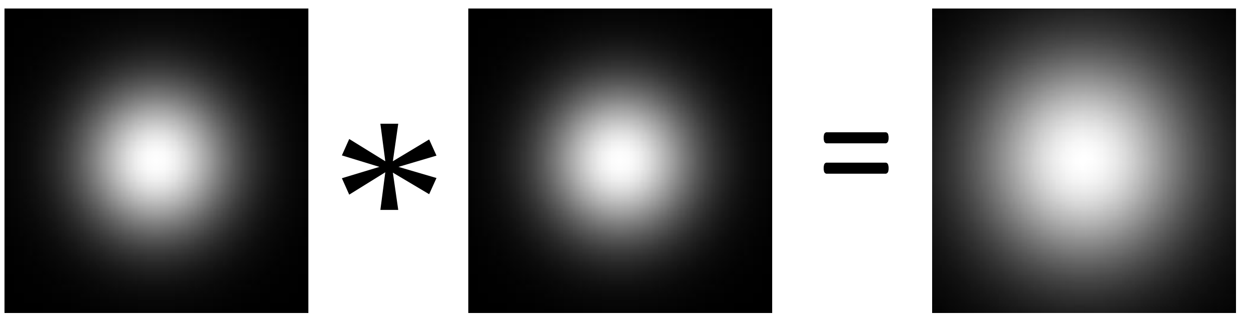
Gaussian filters



σ = 1 pixel σ = 5 pixels σ = 10 pixels σ = 30 pixels

Gaussian filter

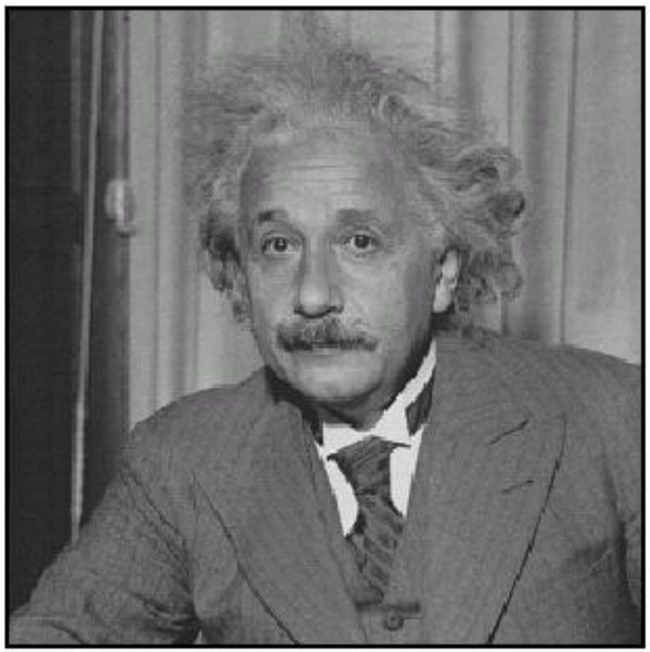
- Removes "high-frequency" components from the image (low-pass filter)
- Convolution with self is another Gaussian



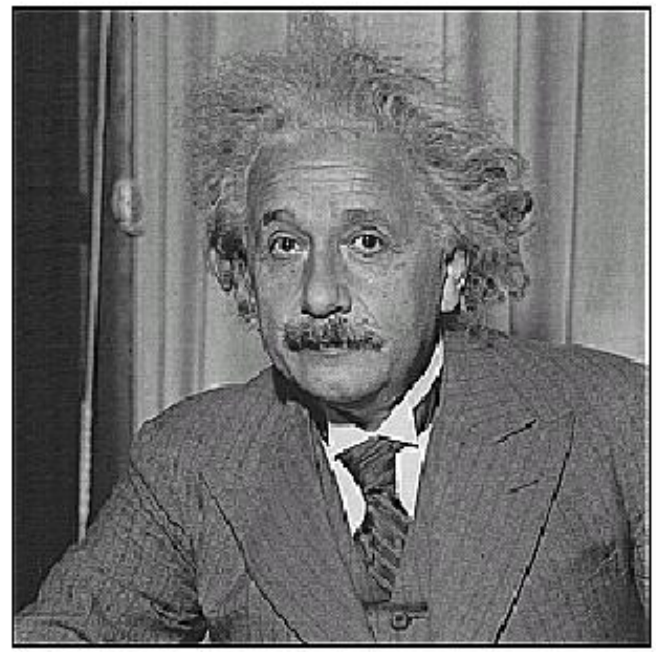
  - Convolving two times with Gaussian kernel of width σ = convolving once with kernel of width $\sigma\sqrt{2}$

Source: K. Grauman

Sharpening



before



after

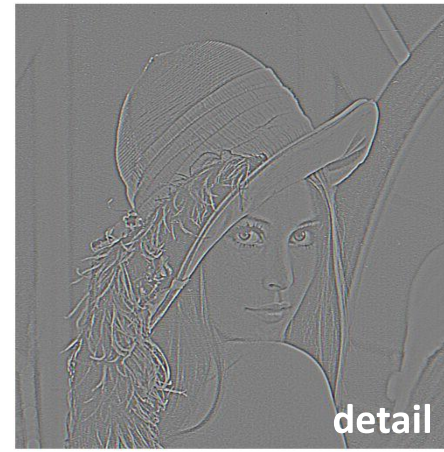
Source: D. Lowe

Sharpening revisited

- What does blurring take away?

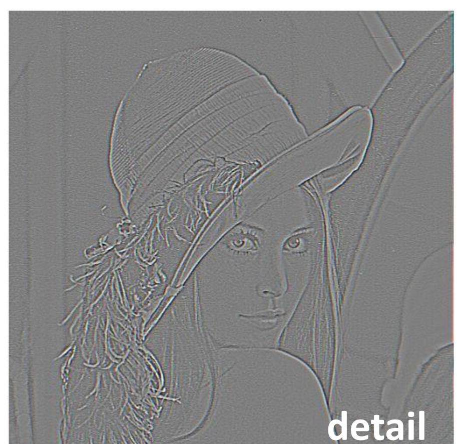
–

=

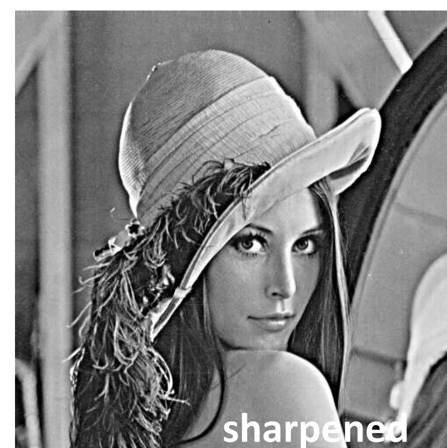


Let’s add it back:

+ α



=



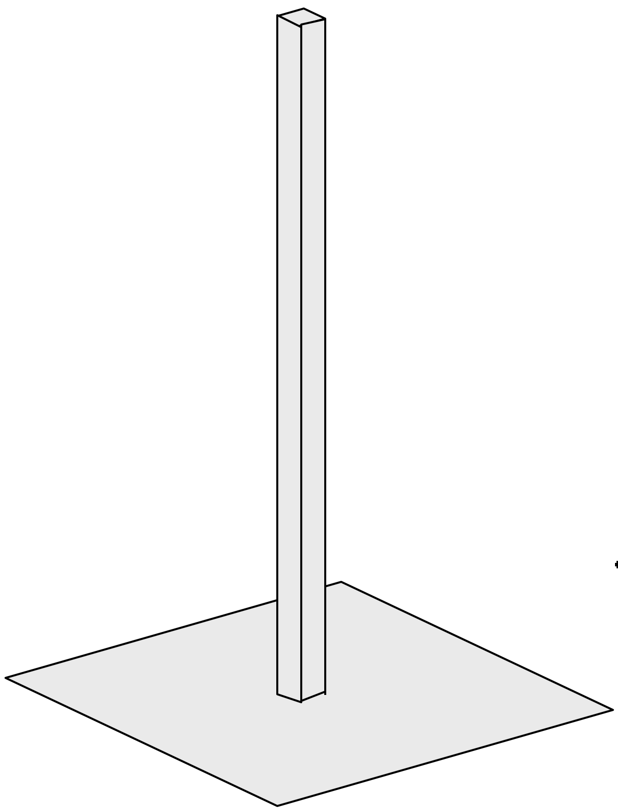
Source: S. Lazebnik

Sharpen filter

$$F + \alpha \left(F - \underbrace{F * H}_{\text{blurred image}}\right)$$

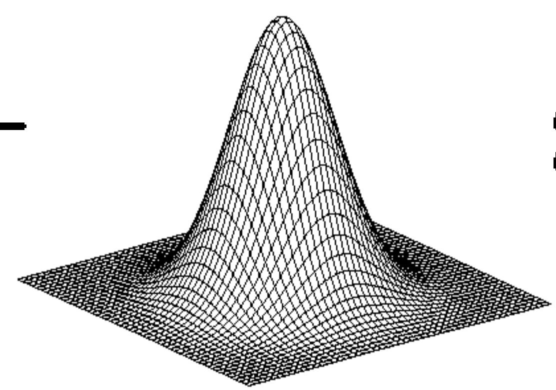
↑ image

↑ unit impulse (identity)



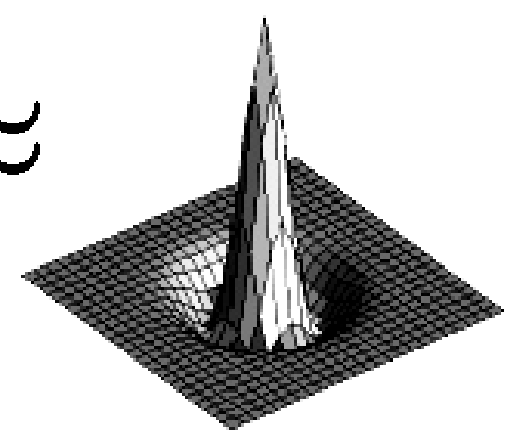
scaled impulse

$-$



Gaussian

$\approx$

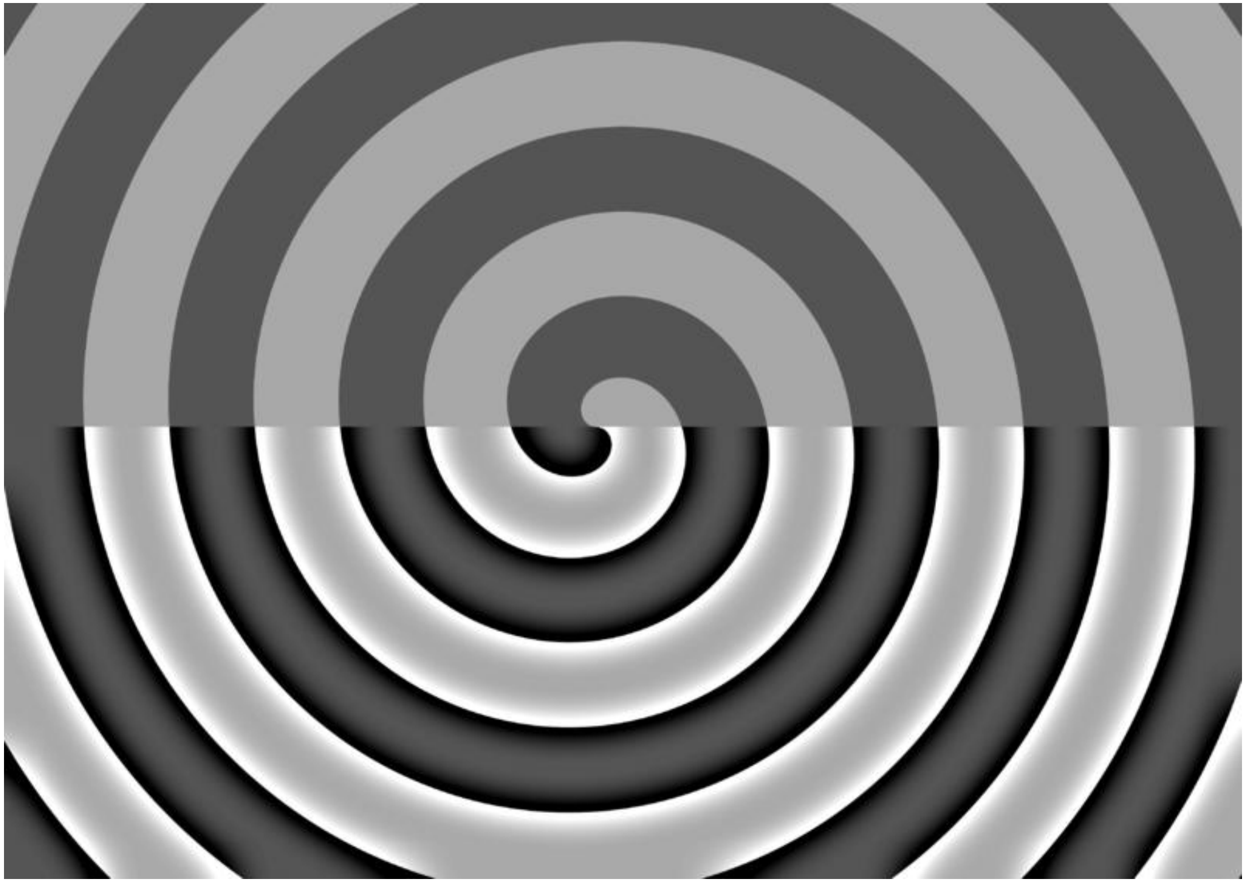


Laplacian of Gaussian

Sharpen filter

unfiltered

filtered



Convolution in the real world

Camera shake

Source: Fergus, *et al.* "Removing Camera Shake from a Single Photograph", SIGGRAPH 2006

Bokeh: Blur in out-of-focus regions of an image.

Source: http://lullaby.homepage.dk/diy-camera/bokeh.html

Questions?

Image noise

Original image

$F[x, y]$

White Gaussian noise

$F[x, y] + \mathcal{N}(0, \sigma)$

Salt and pepper noise

(each pixel has some chance of being switched to zero or one)

http://theory.uchicago.edu/~ejm/pix/20d/tests/noise/index.html

Gaussian noise

$F[x, y] + \mathcal{N}(0, 5\%)$

σ = 1 pixel

σ = 2 pixels

σ = 5 pixels

Smoothing with larger standard deviations suppresses noise, but also blurs the image

Salt & pepper noise – Gaussian blur

p = 10%

σ = 1 pixel

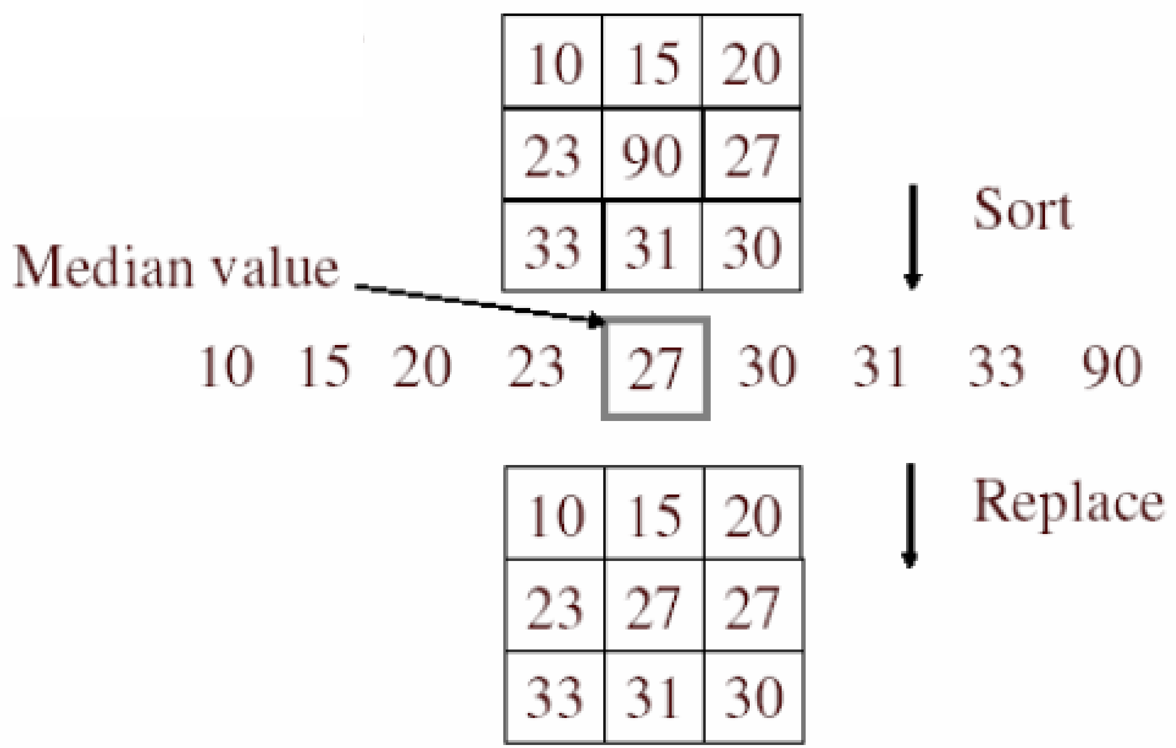
σ = 2 pixels

σ = 5 pixels

- What’s wrong with the results?

Alternative idea: Median filtering

- A **median filter** operates over a window by selecting the median intensity in the window

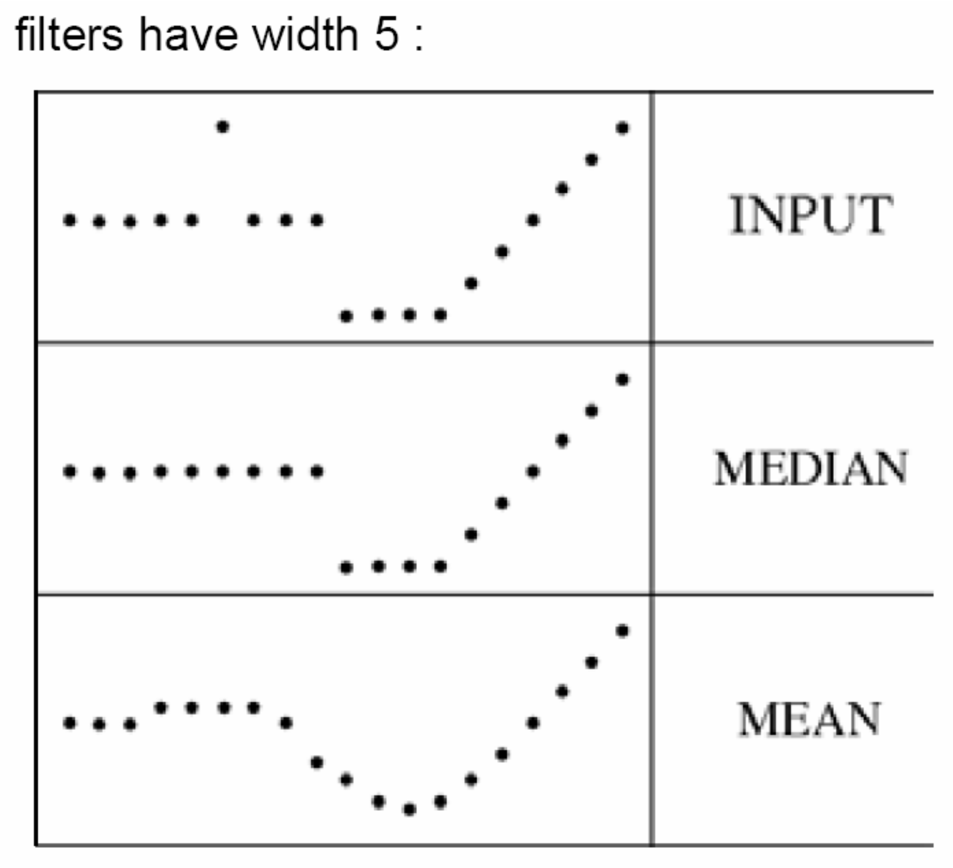


- Is median filtering linear?

Source: K. Grauman

Median filter

- What advantage does median filtering have over Gaussian filtering?



Source: K. Grauman

Salt & pepper noise – median filtering

p = 10%

σ = 1 pixel

σ = 2 pixels

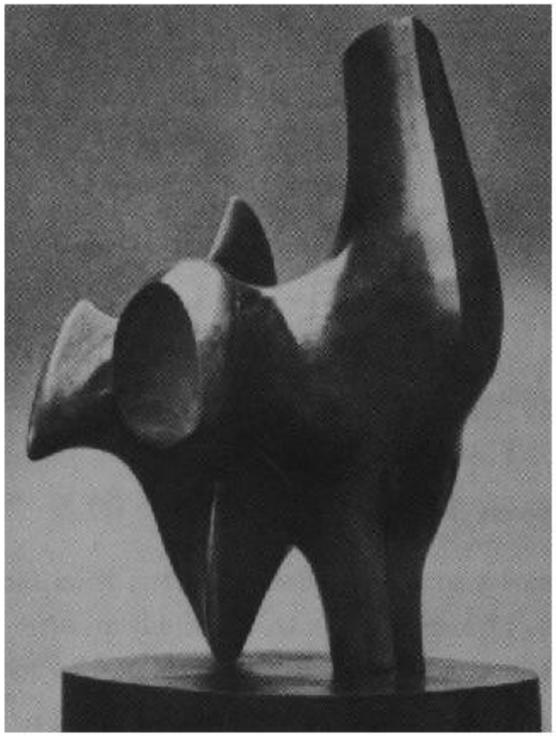
σ = 5 pixels

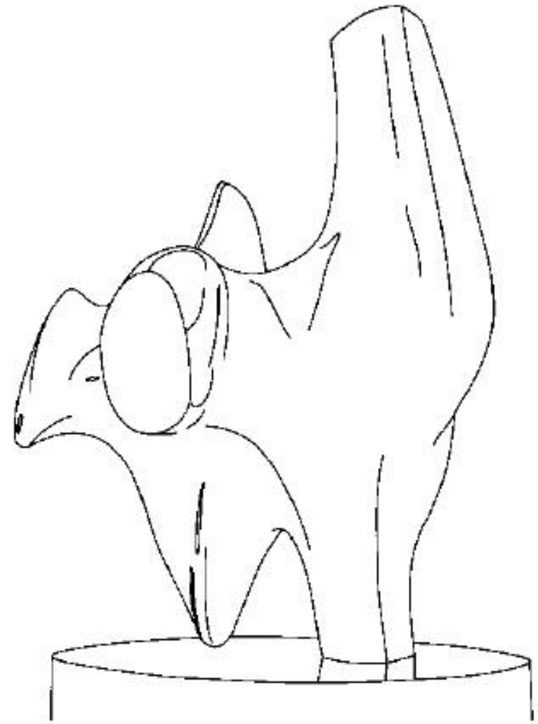
3x3 window

5x5 window

7x7 window

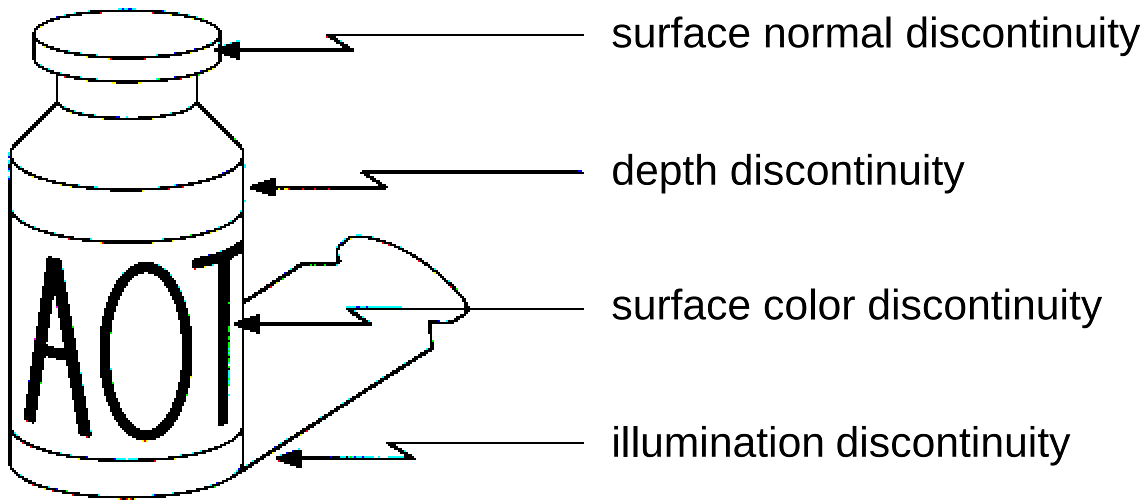
Edge detection





- Convert a 2D image into a set of curves
  - Extracts salient features of the scene
  - More compact than pixels

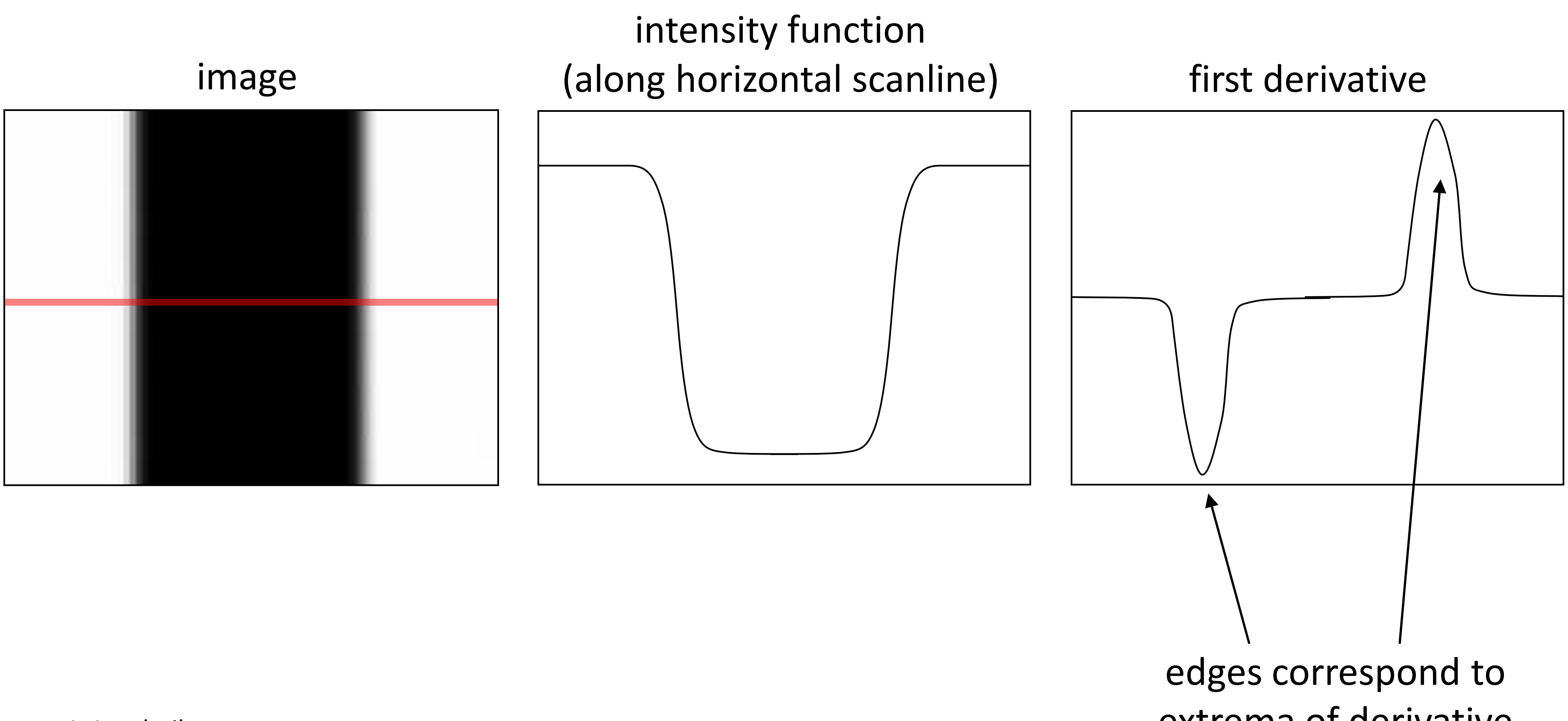
Origin of Edges



- Edges are caused by a variety of factors

Characterizing edges

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



Source: L. Lazebnik

Image derivatives

- How can we differentiate a *digital* image F[x,y]?
  - Option 1: reconstruct a continuous image, *f,* then compute the derivative
  - Option 2: take discrete derivative (finite difference)

$$\frac{\partial f}{\partial x}[x, y] \approx F[x+1, y] - F[x, y]$$

How would you implement this as a linear filter?

$\frac{\partial f}{\partial x}$:

H_x

$\frac{\partial f}{\partial y}$:

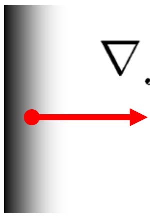
H_y

Source: S. Seitz

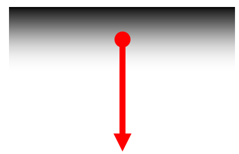
Image gradient

- The *gradient* of an image: $$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

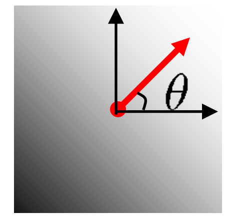
The gradient points in the direction of most rapid increase in intensity



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



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



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

The *edge strength* is given by the gradient magnitude:

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

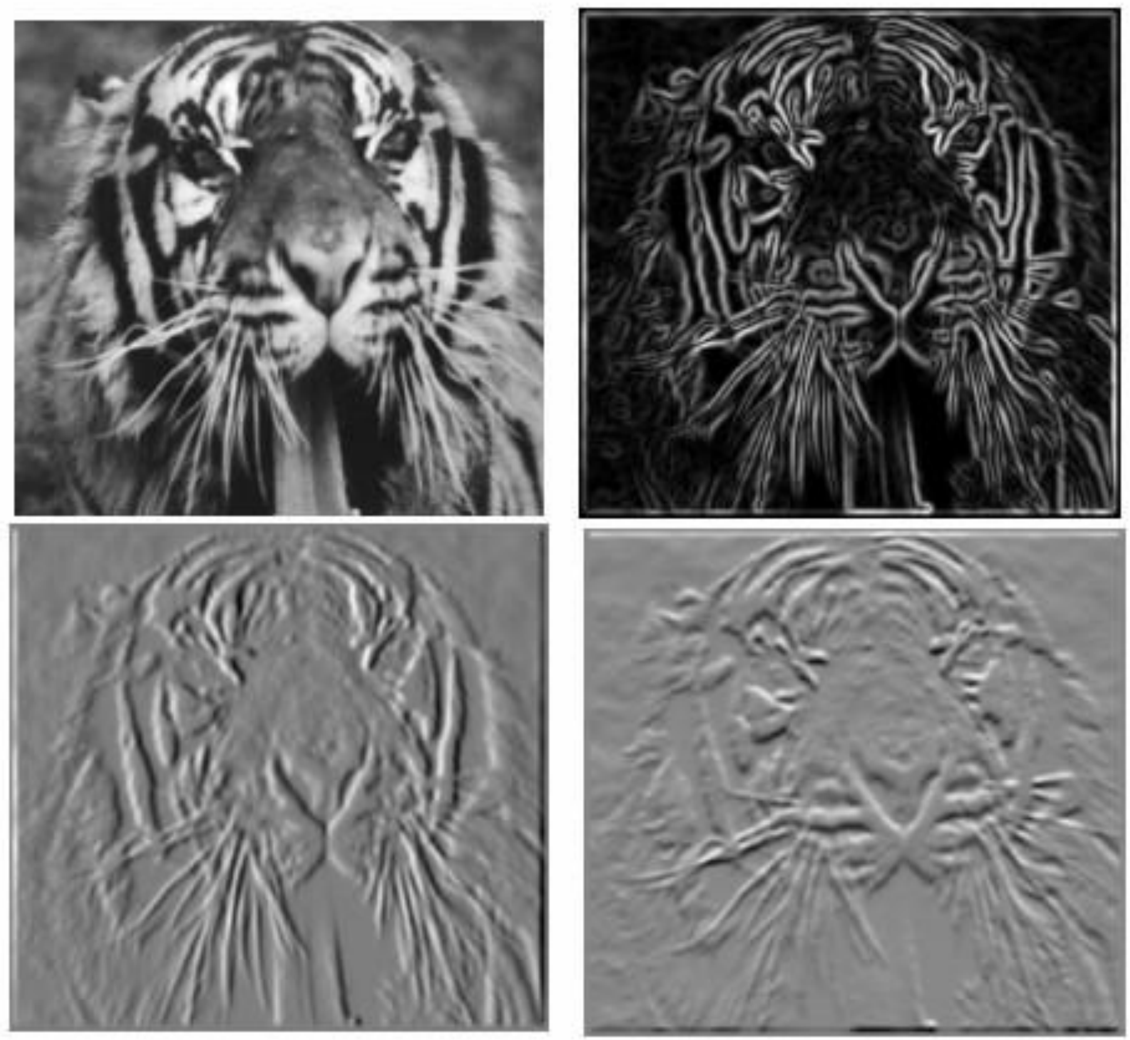
The gradient direction is given by:

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

- how does this relate to the direction of the edge?

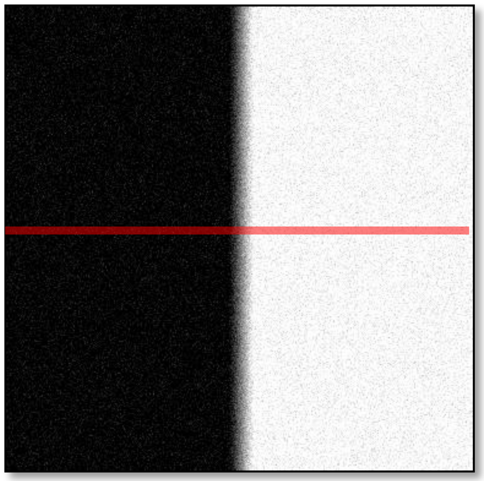
Source: Steve Seitz

Image gradient



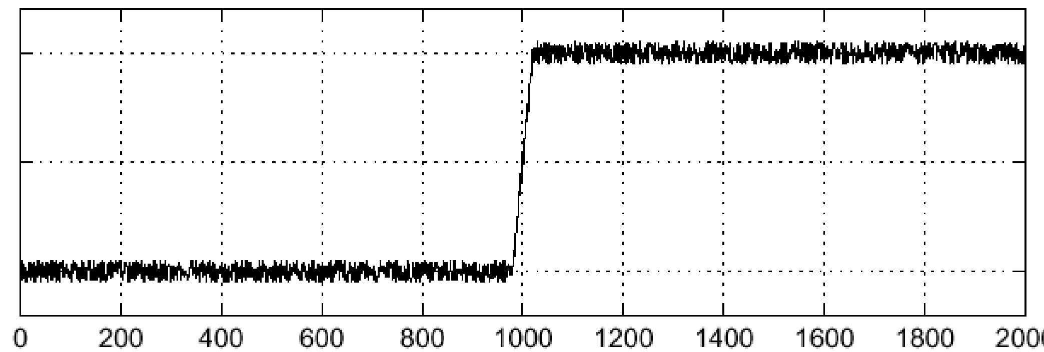
Source: L. Lazebnik

Effects of noise

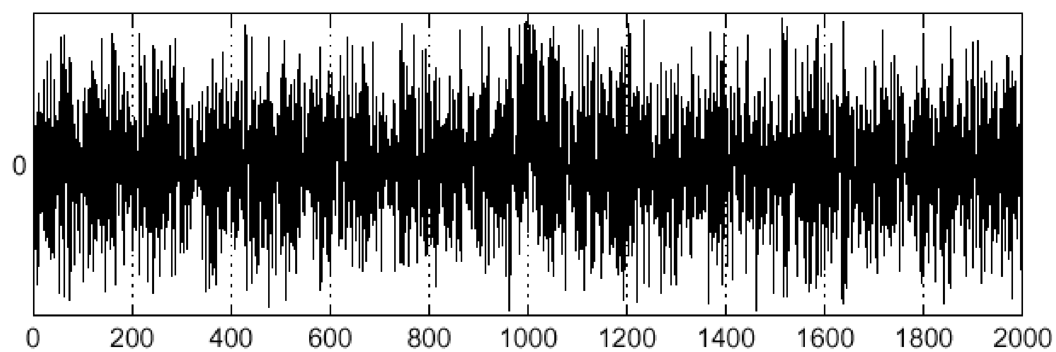


Noisy input image

$f(x)$



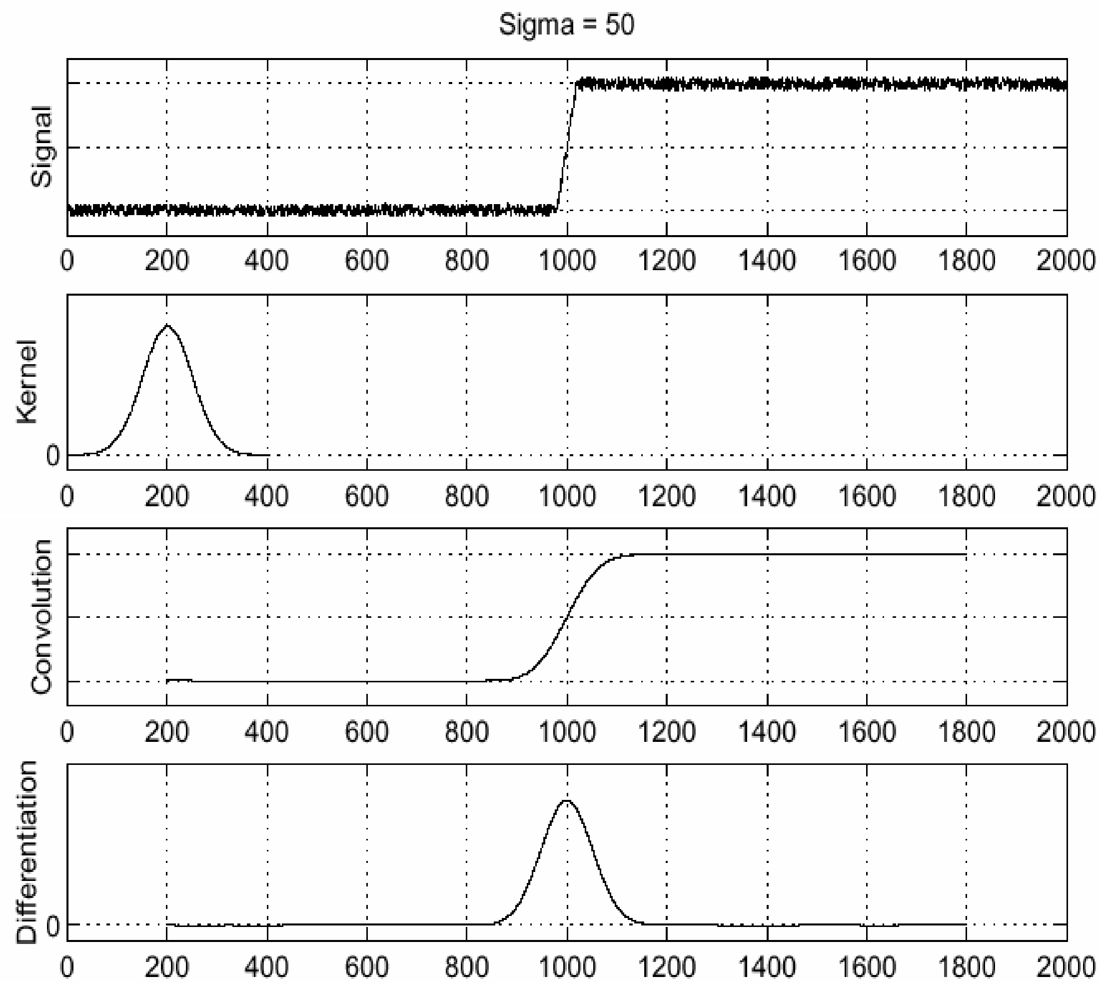
$\frac{d}{dx}f(x)$



Where is the edge?

Source: S. Seitz

Solution: smooth first



f

h

$f * h$

$\frac{d}{dx}(f * h)$

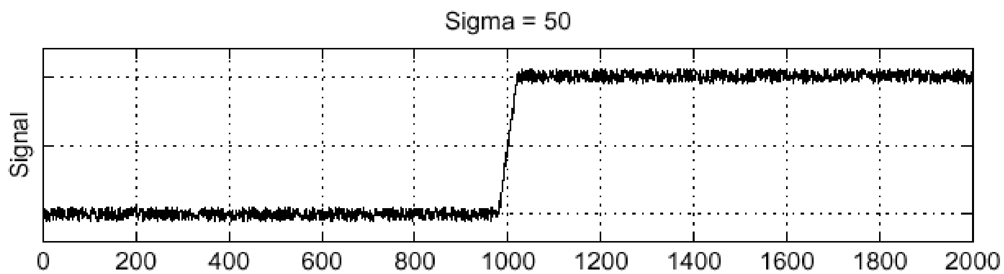
To find edges, look for peaks in $\frac{d}{dx}(f * h)$

Source: S. Seitz

Associative property of convolution

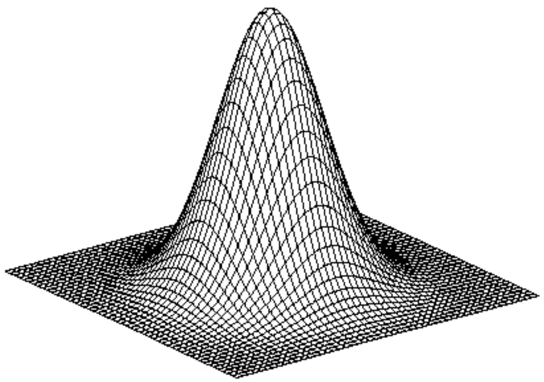
- Differentiation is convolution, and convolution is associative: $\frac{d}{dx}(f * h) = f * \frac{d}{dx}h$
- This saves us one operation:

f



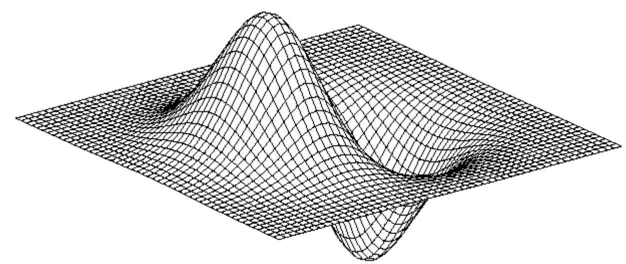
Source: S. Seitz

2D edge detection filters



Gaussian

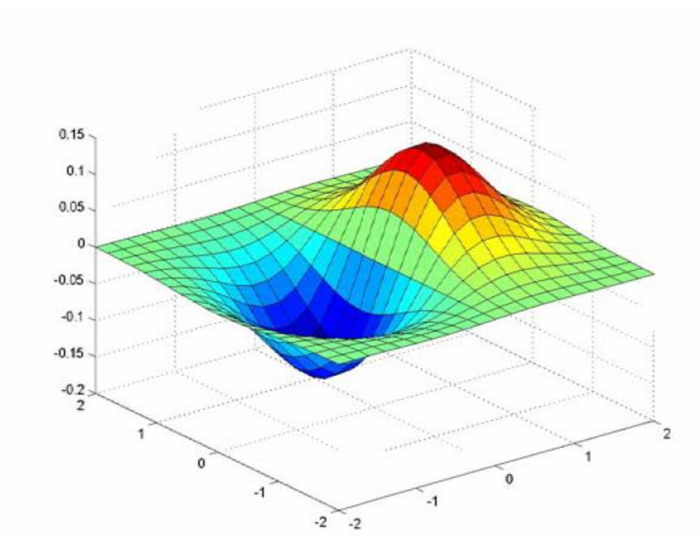
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



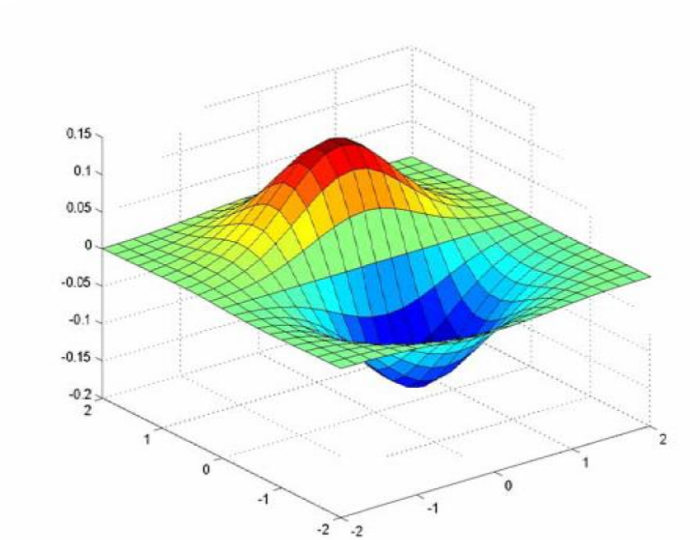
derivative of Gaussian (*x*)

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

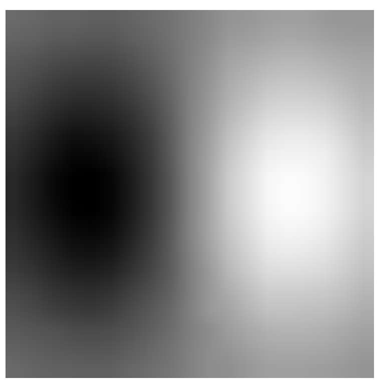
Derivative of Gaussian filter

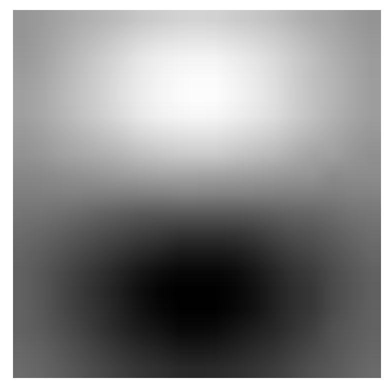


x-direction



y-direction





The Sobel operator

- Common approximation of derivative of Gaussian

$$\frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$$s_x \qquad\qquad s_y$$

- The standard defn. of the Sobel operator omits the 1/8 term
  - doesn't make a difference for edge detection
  - the 1/8 term **is** needed to get the right gradient value