Foundations of Computer Vision

Prof. Feng Liu

Spring 2024

http://www.cs.pdx.edu/~fliu/courses/cs410/

04/09/2024

Last Time

- Filter

Today

- More on Filter
- Feature Detection

Filter Re-cap

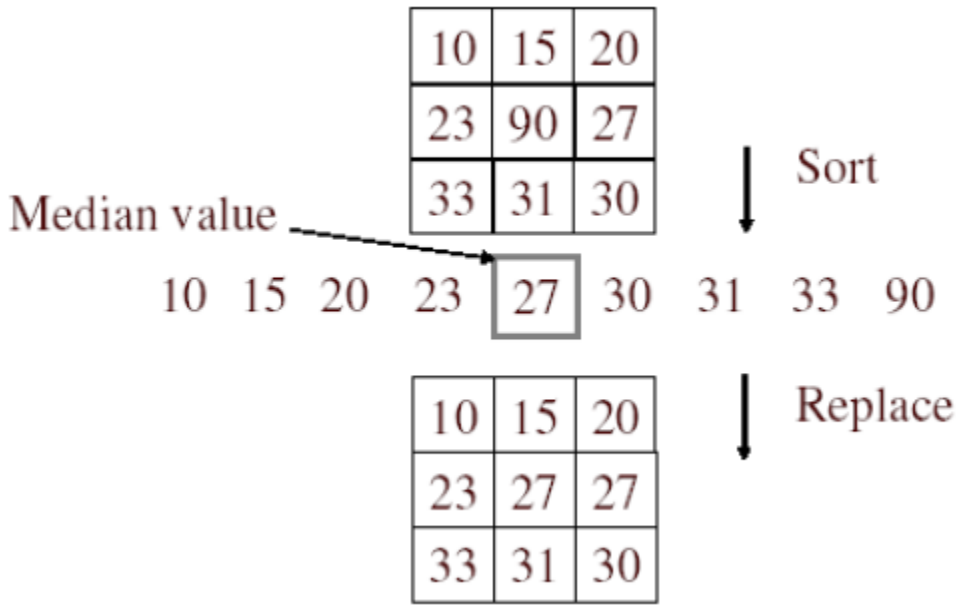
noisy image

naïve denoising
Gaussian blur

better denoising
edge-preserving filter

Slide credit: Sylvain Paris and Frédo Durand

Median Filter

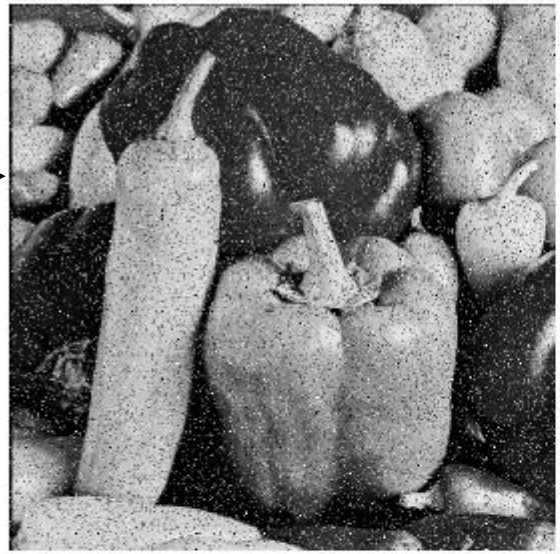


- Replace pixel by the median value of its neighbors
- No new pixel values introduced
- Removes spikes: good for impulse, salt & pepper noise
- **Nonlinear filter**

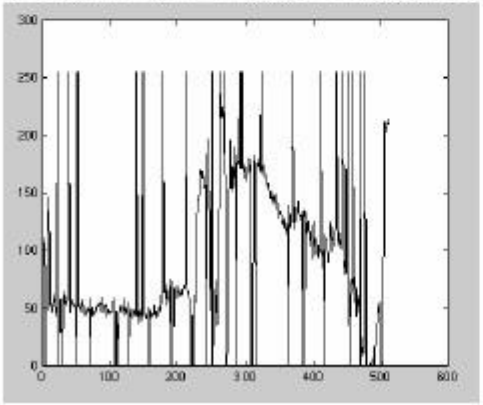
Slide credit: C. Dyer

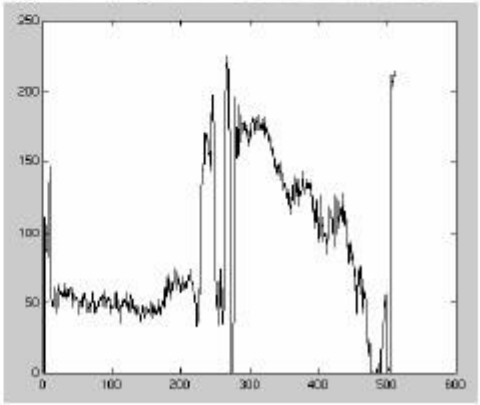
Median Filter

Salt and pepper noise



Median filtered





Plots of a row of the image

Matlab: output im = medfilt2(im, [h w]);

Slide credit: M. Hebert, C. Dyer

Bilateral filter

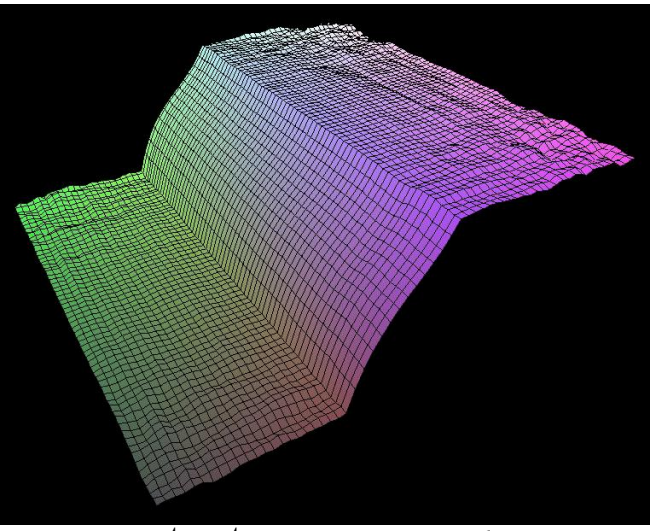
- Tomasi and Manduci 1998
  http://www.cse.ucsc.edu/~manduchi/Papers/ICCV98.pdf
- Related to
  - SUSAN filter
    [Smith and Brady 95]
    http://citeseer.ist.psu.edu/smith95susan.html
  - Digital-TV [Chan, Osher and Chen 2001]
    http://citeseer.ist.psu.edu/chan01digital.html
  - sigma filter
    http://www.geogr.ku.dk/CHIPS/Manual/f187.htm

Slide credit: F. Durand

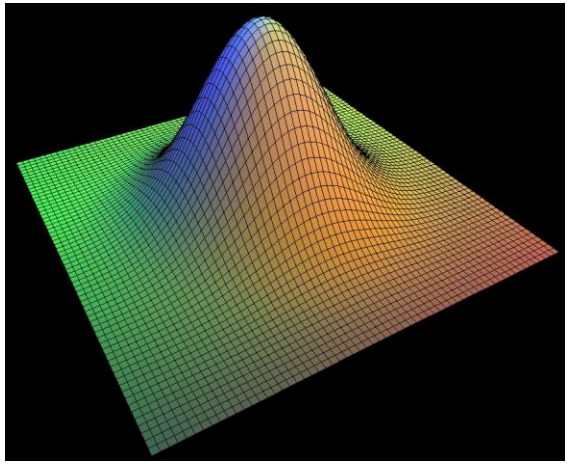
Start with Gaussian filtering

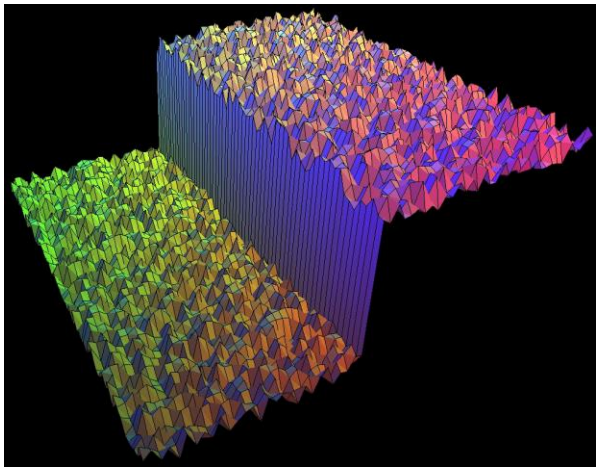
- Here, input is a step function + noise

$$J = f \otimes I$$



output





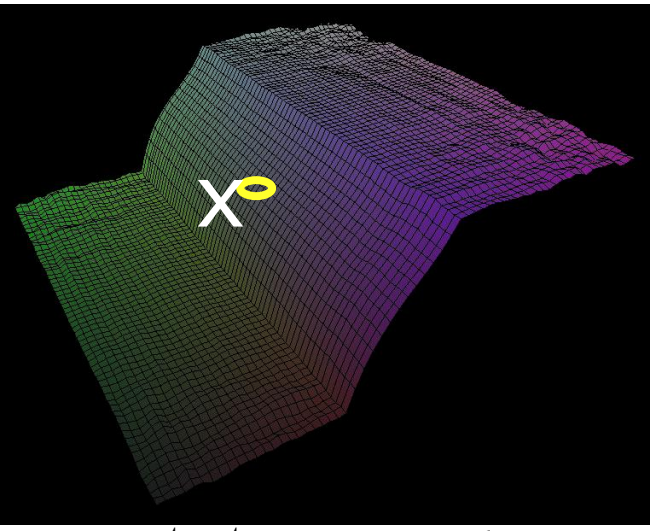
input

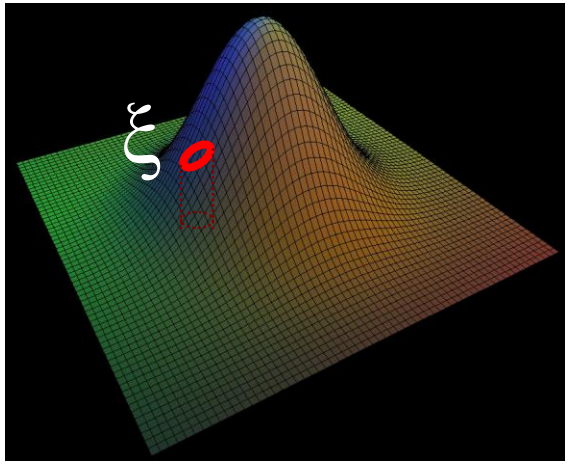
Slide credit: F. Durand

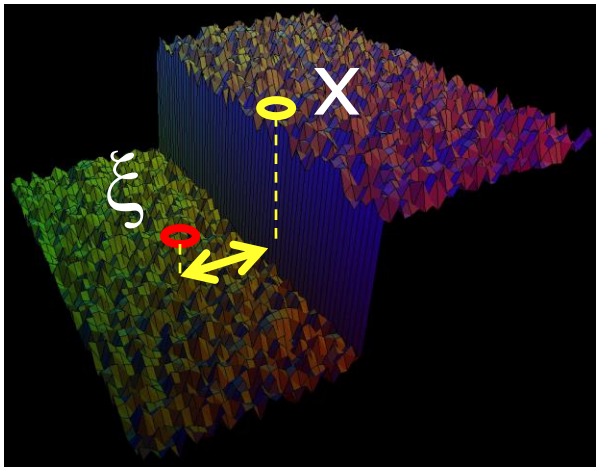
Gaussian filter as weighted average

- Weight of ξ depends on distance to x

$$J(x) = \sum_{\xi} f(x,\xi) \qquad I(\xi)$$







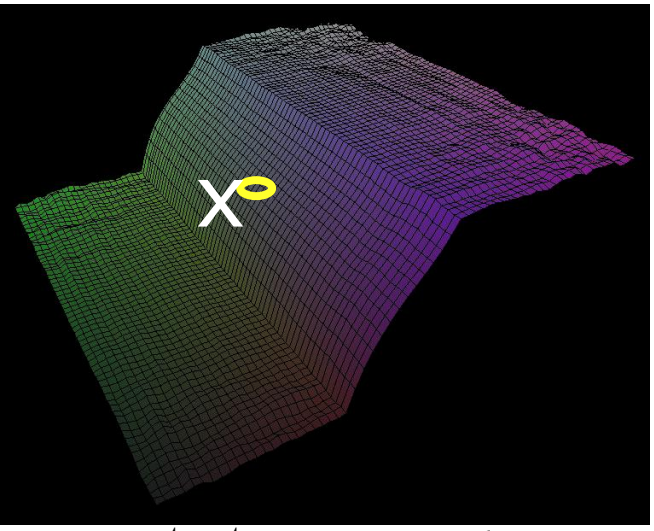
output ⟵ input

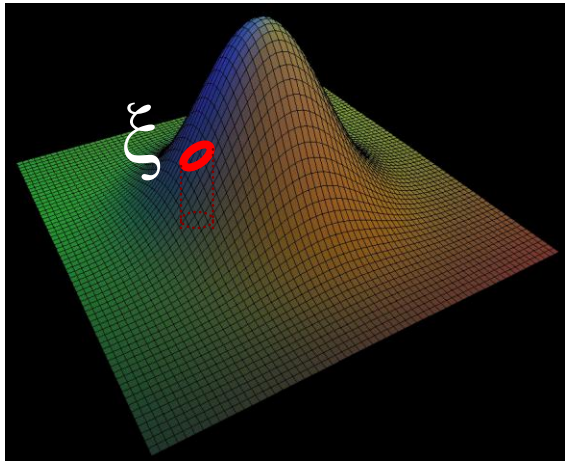
Slide credit: F. Durand

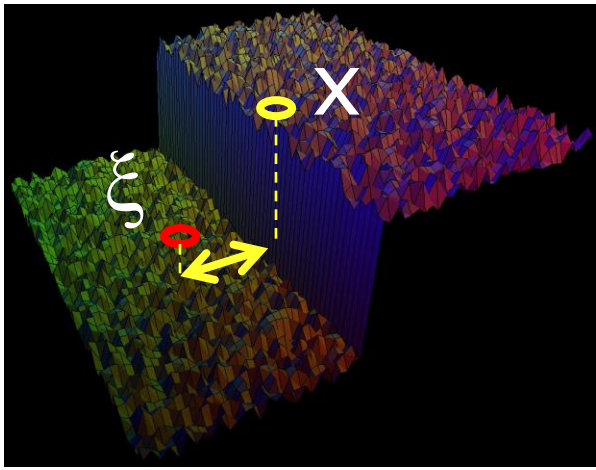
The problem of edges

- Here, $I(\xi)$ "pollutes" our estimate J(x)
- It is too different

$$J(x) = \sum_{\xi} f(x,\xi) \qquad I(\xi)$$







output ⟵ input

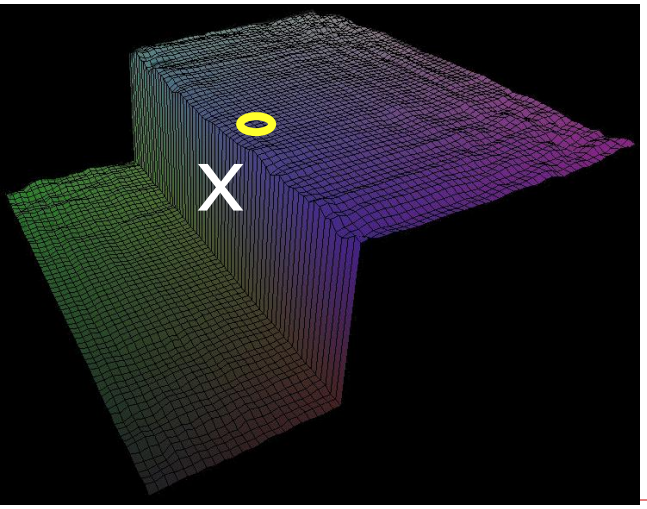
Slide credit: F. Durand

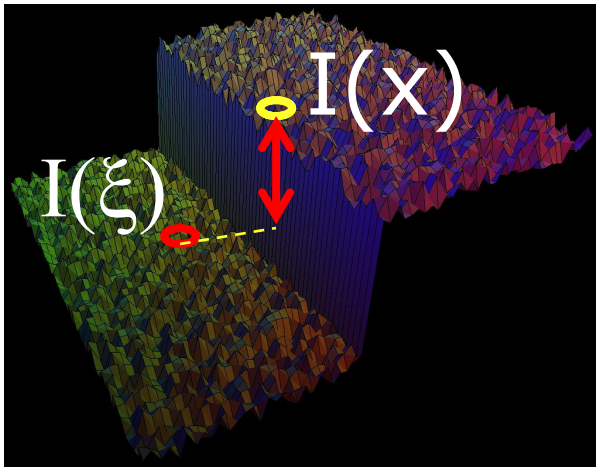
Principle of Bilateral filtering

[Tomasi and Manduchi 1998]

- Penalty g on the intensity difference

$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi) \quad g(I(\xi) - I(x)) \quad I(\xi)$$





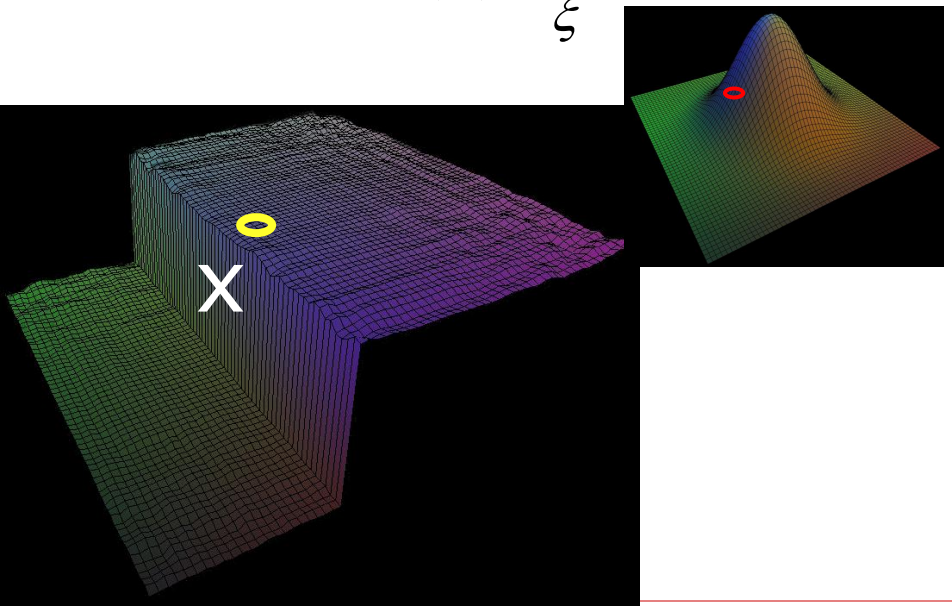
output ⟵ input

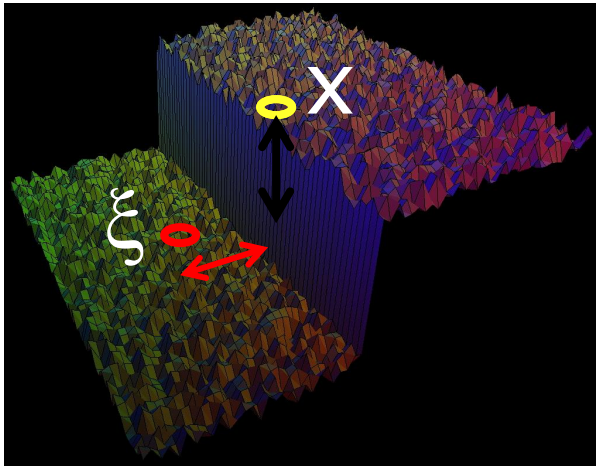
Slide credit: F. Durand

Bilateral filtering [Tomasi and Manduchi 1998]

- Spatial Gaussian f

$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi) g(I(\xi) - I(x)) \quad I(\xi)$$





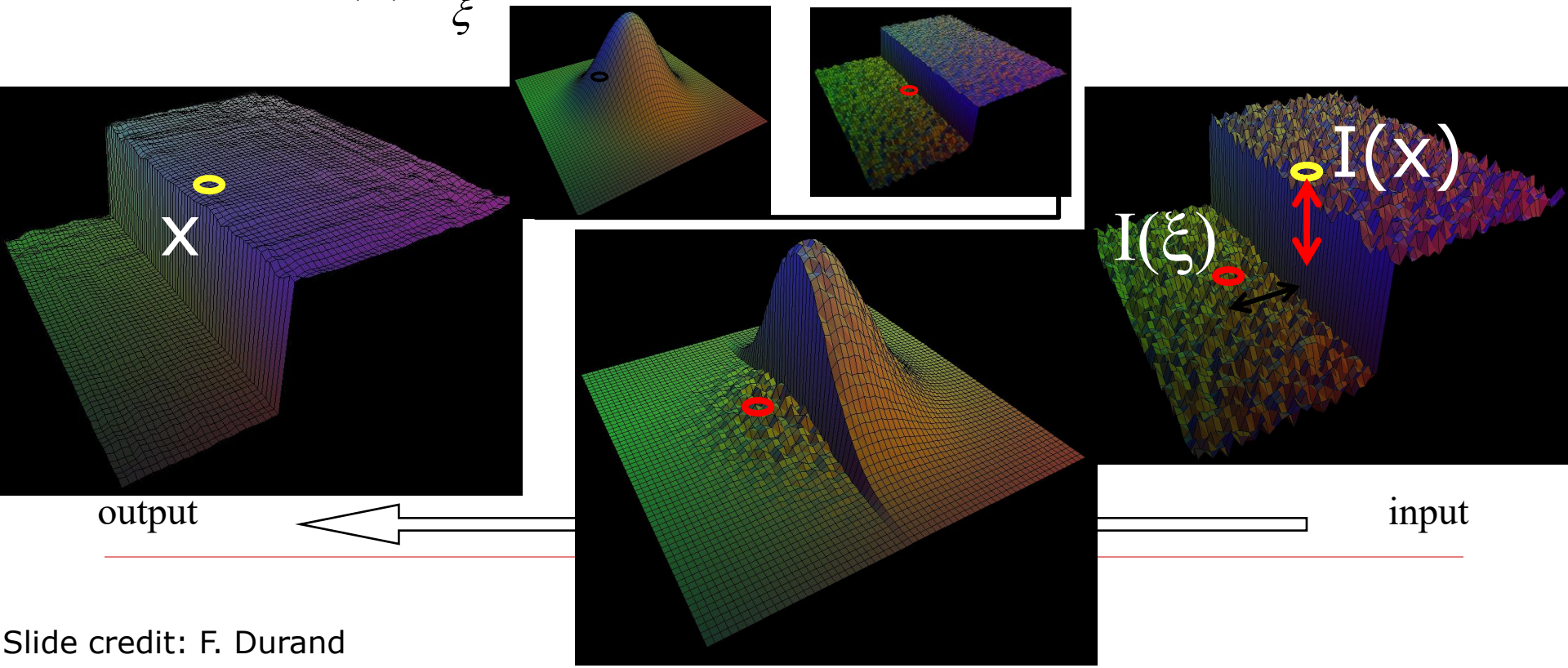
output ⟵ input

Slide credit: F. Durand

Bilateral filtering [Tomasi and Manduchi 1998]

- Spatial Gaussian f
- Gaussian g on the intensity difference

$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi)\, g(I(\xi) - I(x))\, I(\xi)$$

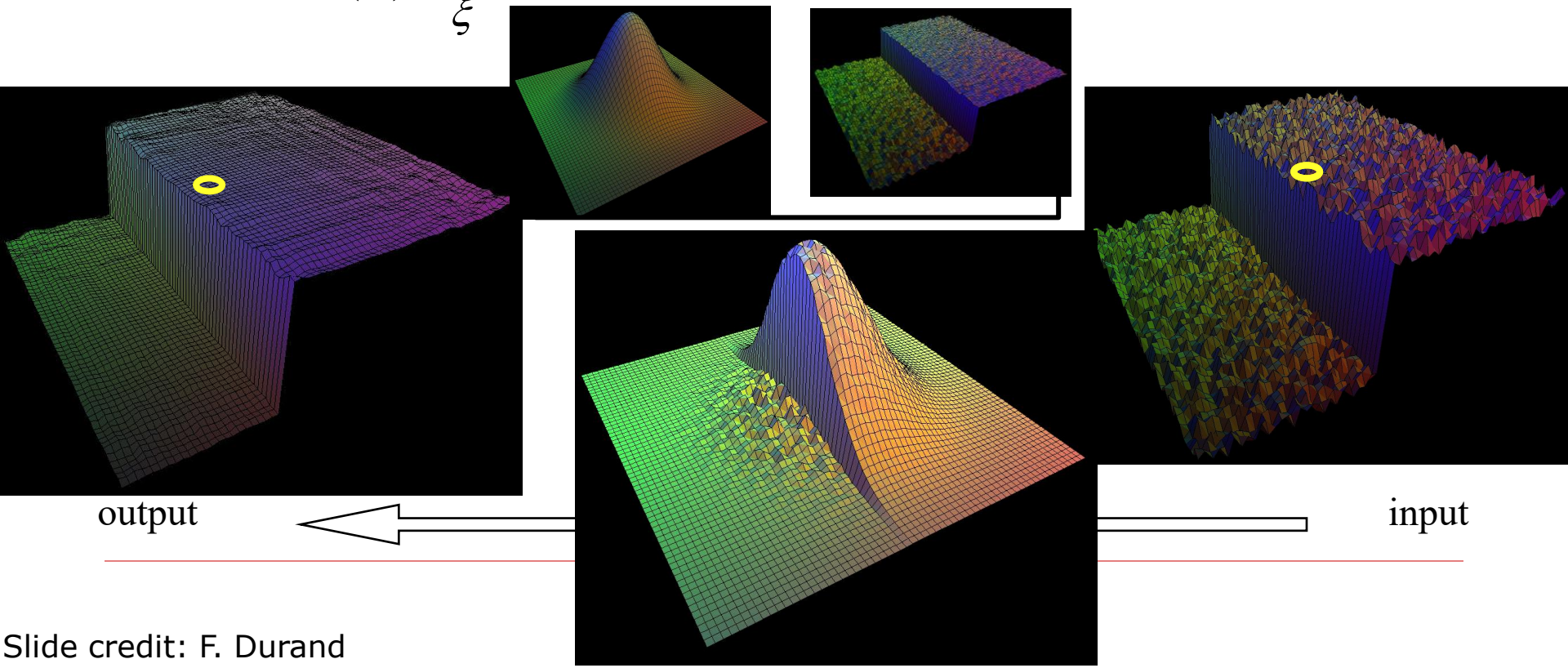


Slide credit: F. Durand

Normalization factor [Tomasi and Manduchi 1998]

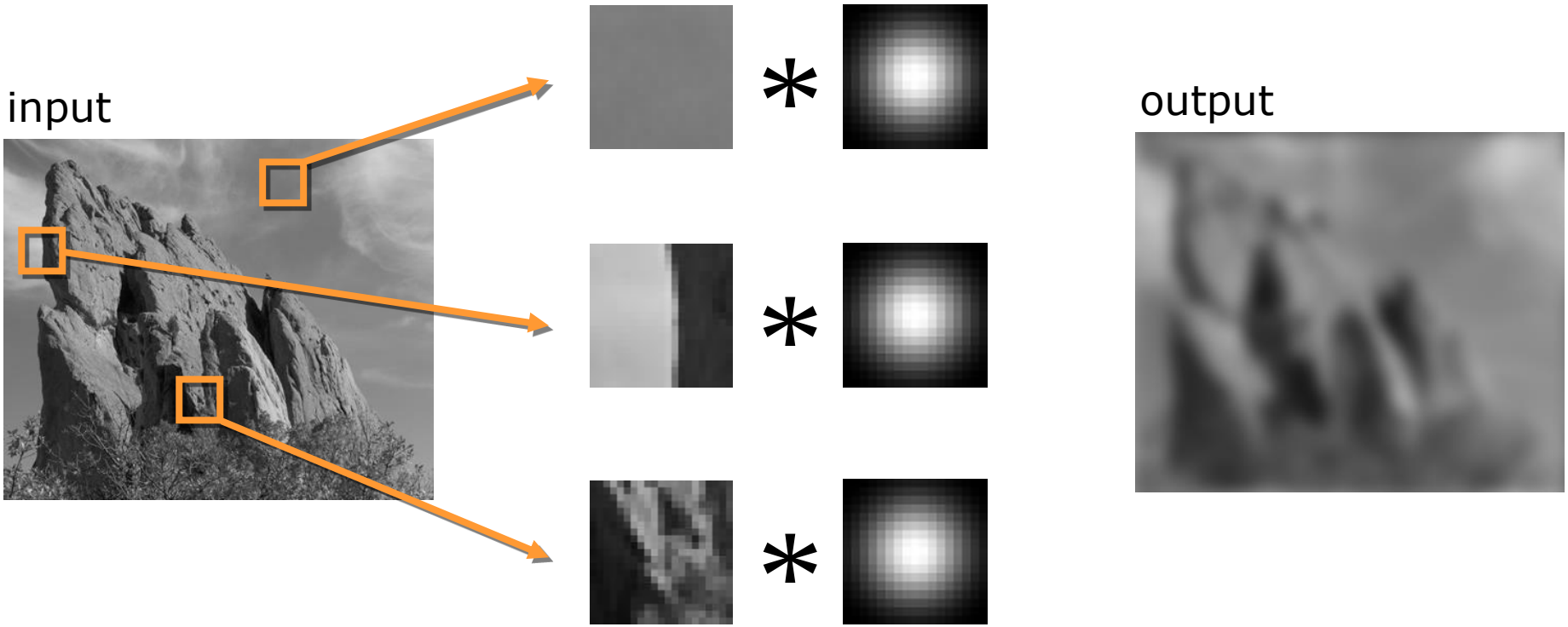
- k(x)= $\sum_{\xi} f(x,\xi) \quad g(I(\xi)-I(x))$

$$J(x)=\frac{1}{k(x)}\sum_{\xi} f(x,\xi) \quad g(I(\xi)-I(x)) \quad I(\xi)$$



Slide credit: F. Durand

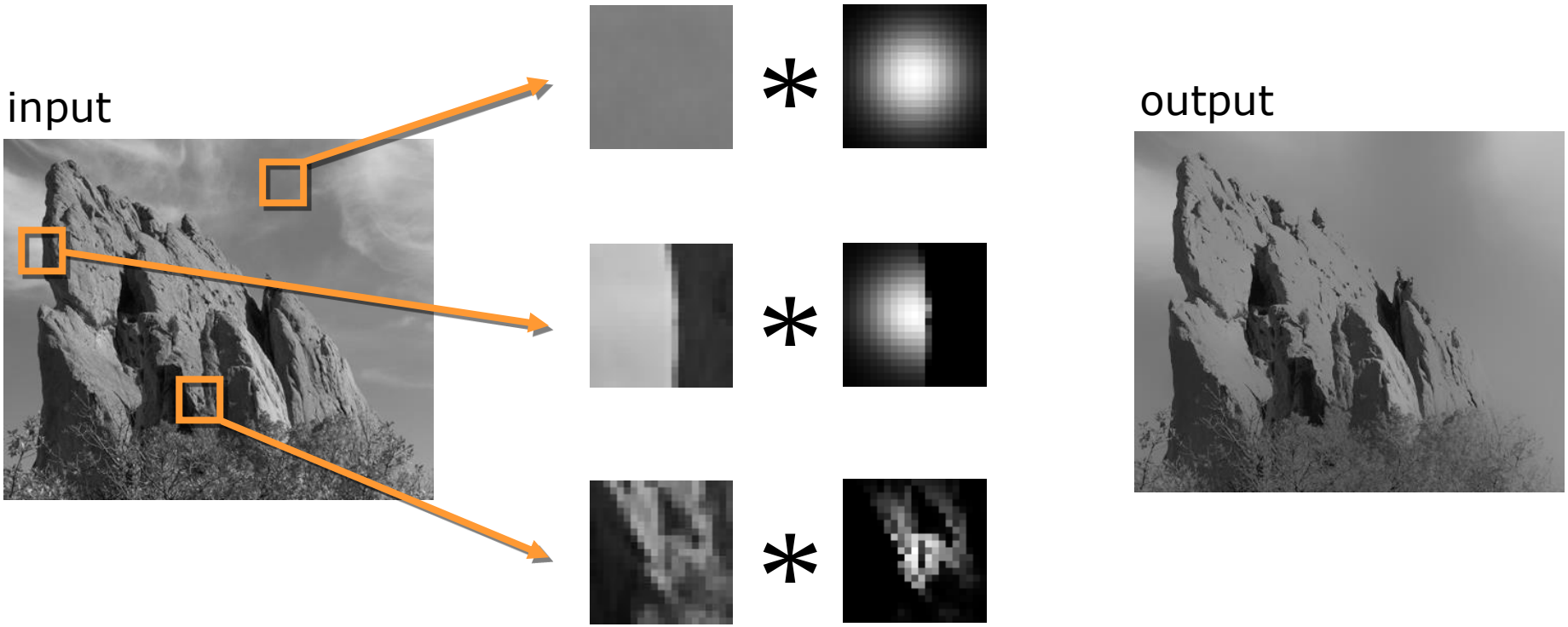
Blur comes from averaging across edges



Same Gaussian kernel everywhere.

Slide credit: P. Sylvain

Bilateral filter: no averaging across edges



The kernel shape depends on the image content.

Slide credit: P. Sylvain

input

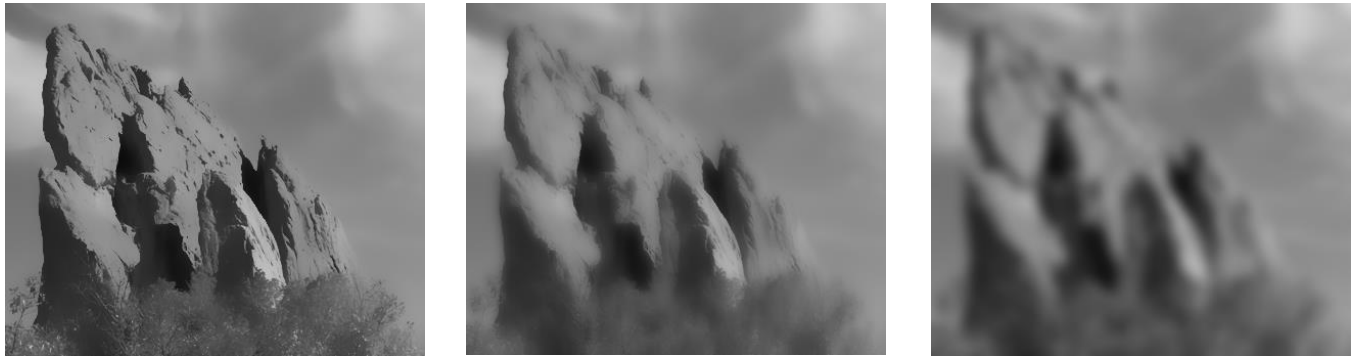
Parameter for intensity difference Gaussian *g*

$\sigma_r = 0.1$ | $\sigma_r = 0.25$ | $\sigma_r = \infty$ (Gaussian blur)

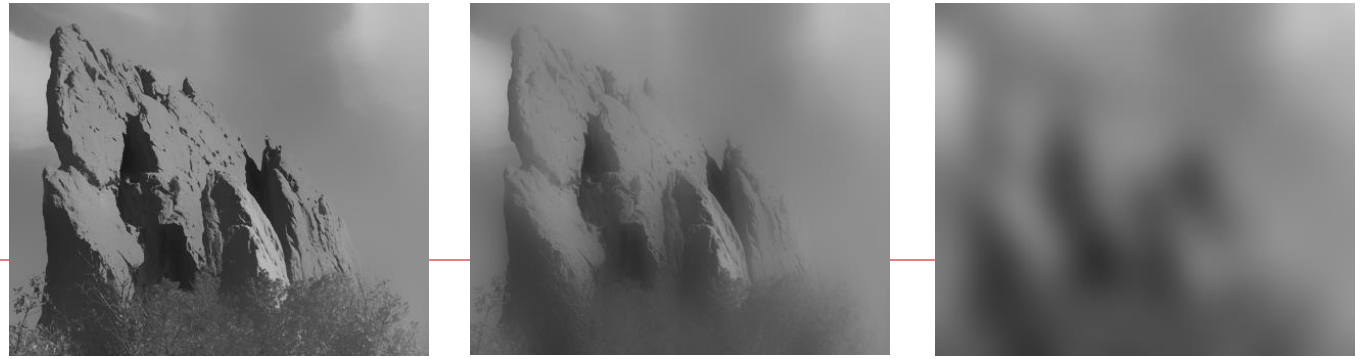
Parameter for spatial distance Gaussian *f*

$\sigma_s = 2$

$\sigma_s = 6$



$\sigma_s = 18$



Slide credit: P. Sylvain

input

Parameter for intensity difference Gaussian *g*

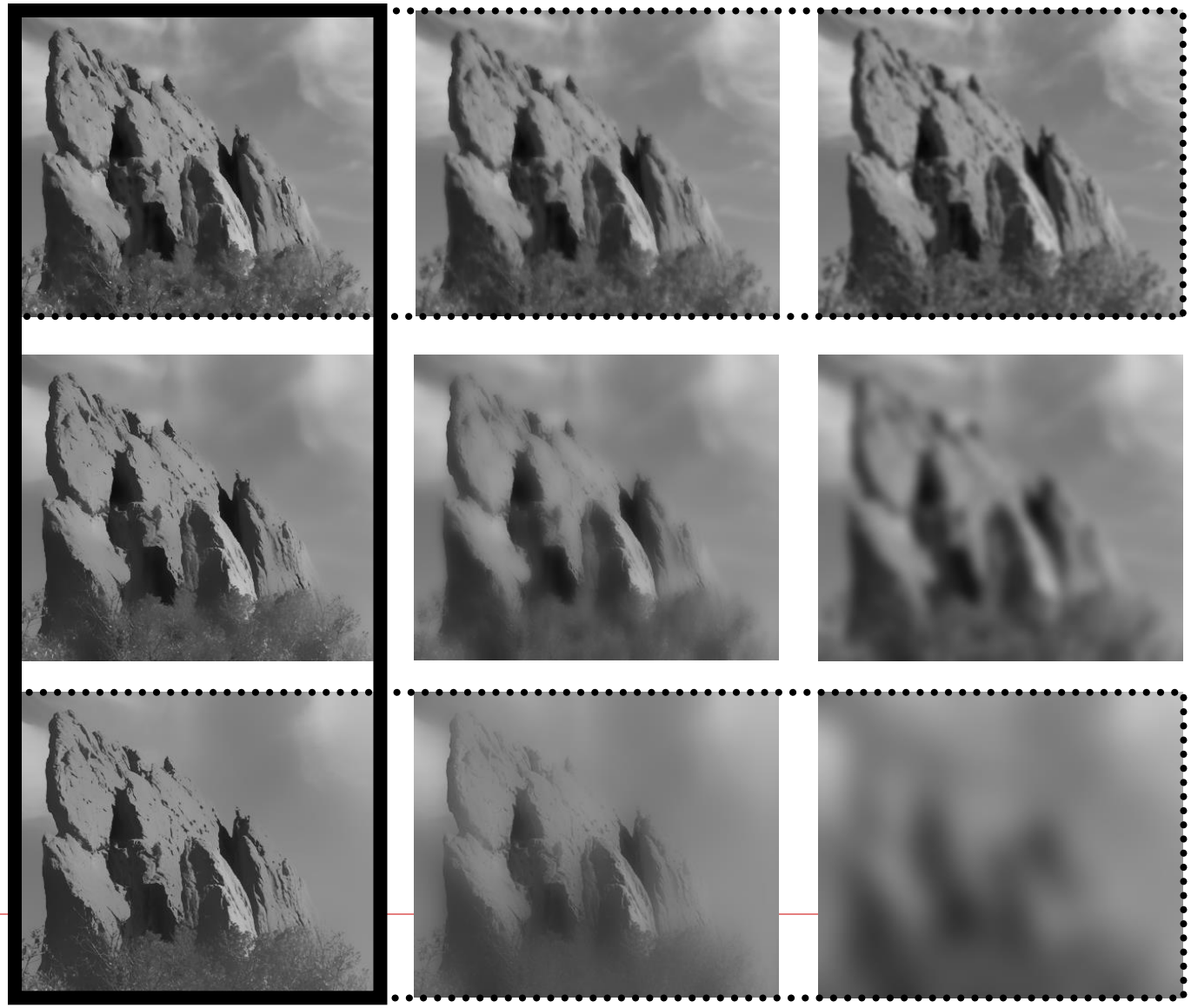
$\sigma_r = 0.1$ $\sigma_r = 0.25$ $\sigma_r = \infty$ (Gaussian blur)

Parameter for spatial distance Gaussian *f*

$\sigma_s = 2$

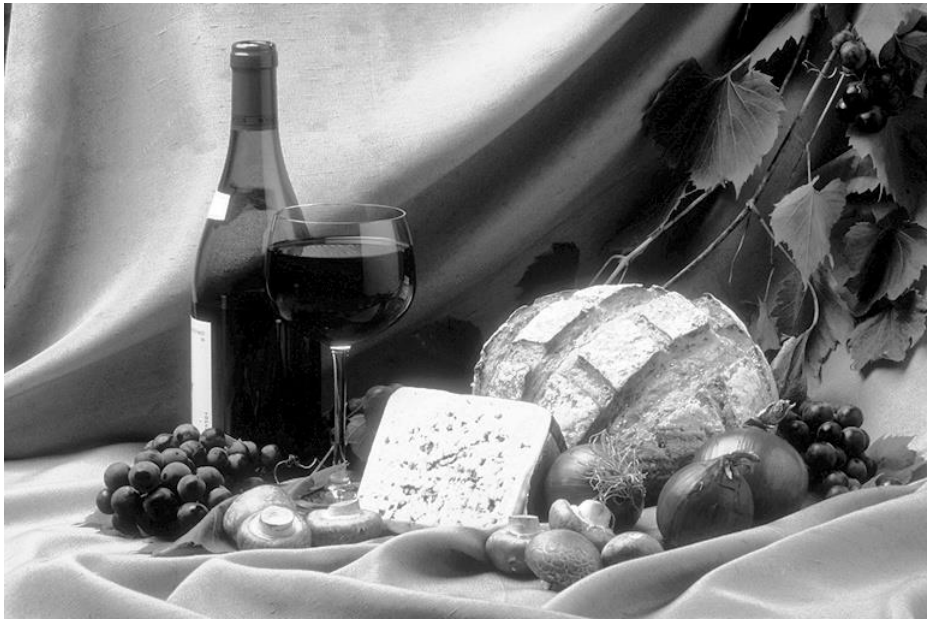
$\sigma_s = 6$

$\sigma_s = 18$



Slide credit: P. Sylvain

Result



Input

Output

Reprint from Tomasi and Manduchi 1998

Why do we say it is non-linear?

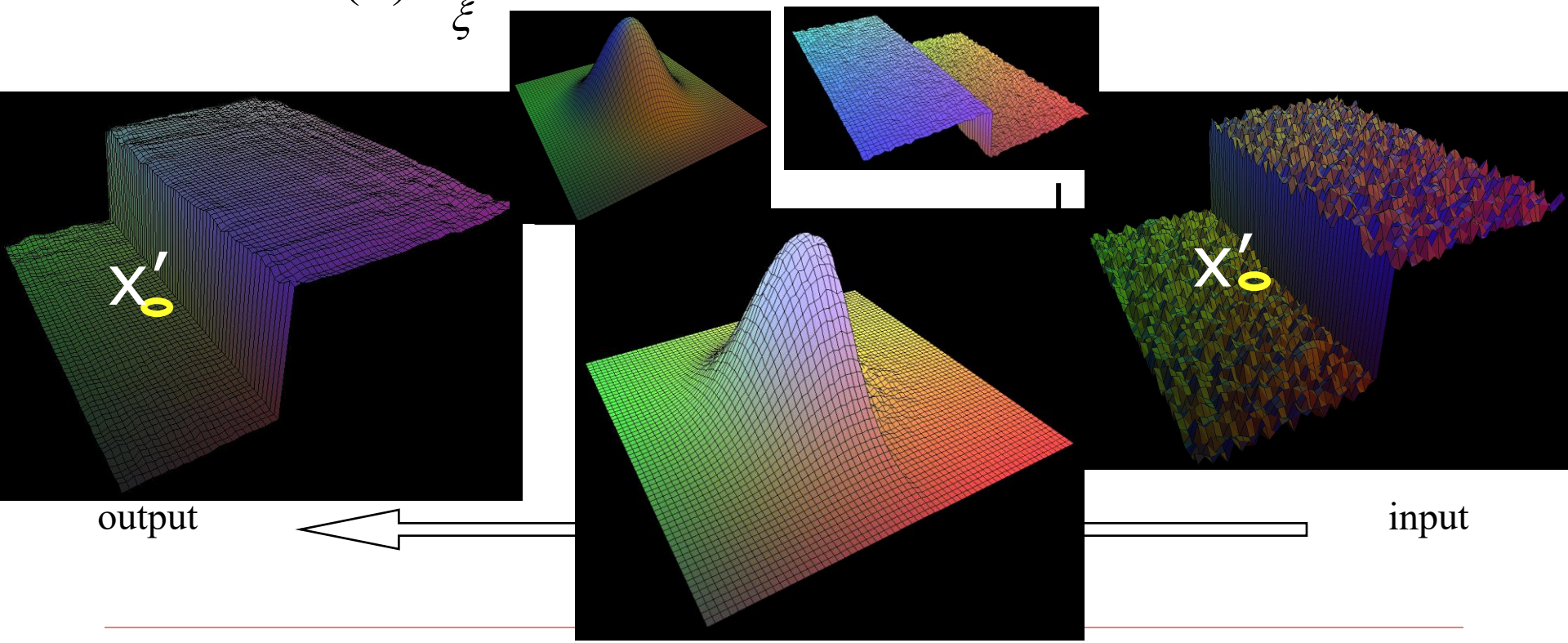
- It does not respect bila(f+g)=bila(f)+bila(g)

Slide credit: F. Durand

Bilateral filtering is non-linear

- The weights are different for each output pixel

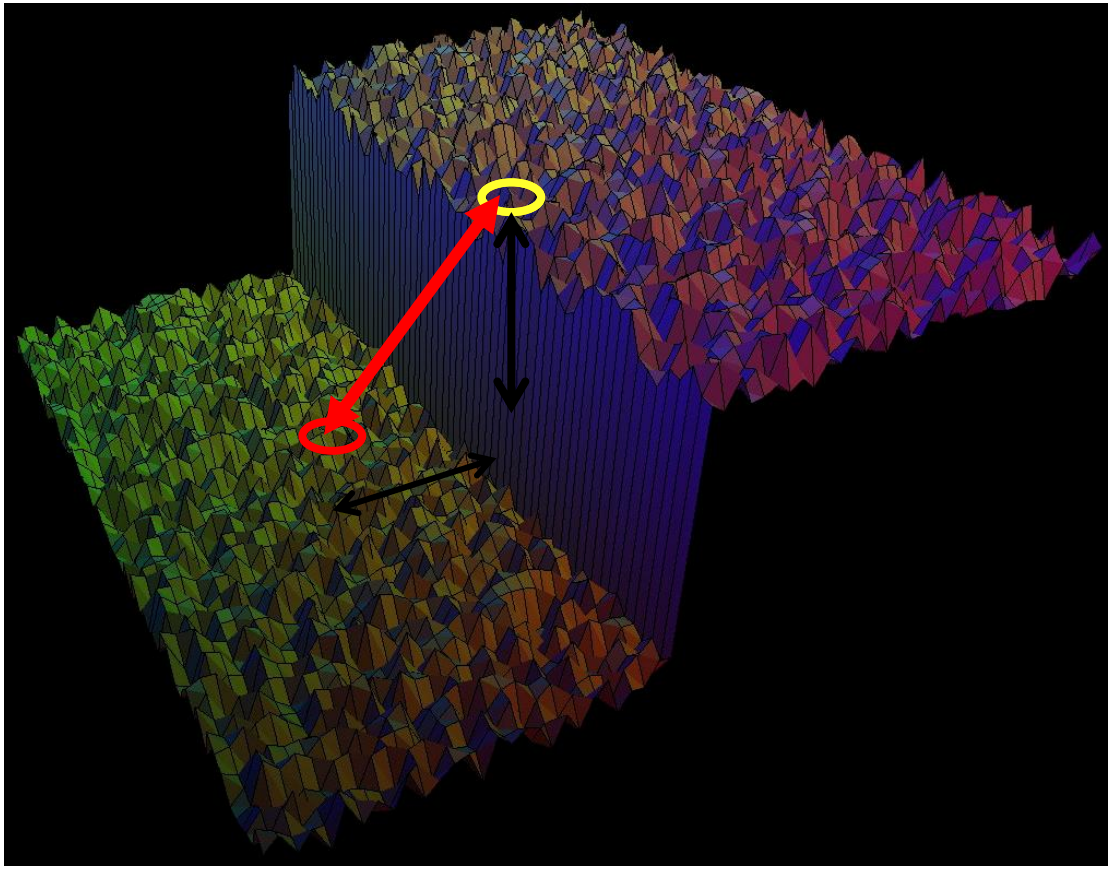
$$J(x) = \frac{1}{k(x)} \sum_{\xi} f(x,\xi) \quad g(I(\xi) - I(x)) \quad I(\xi)$$



Slide credit: F. Durand

Other view

- The bilateral filter uses the 3D distance



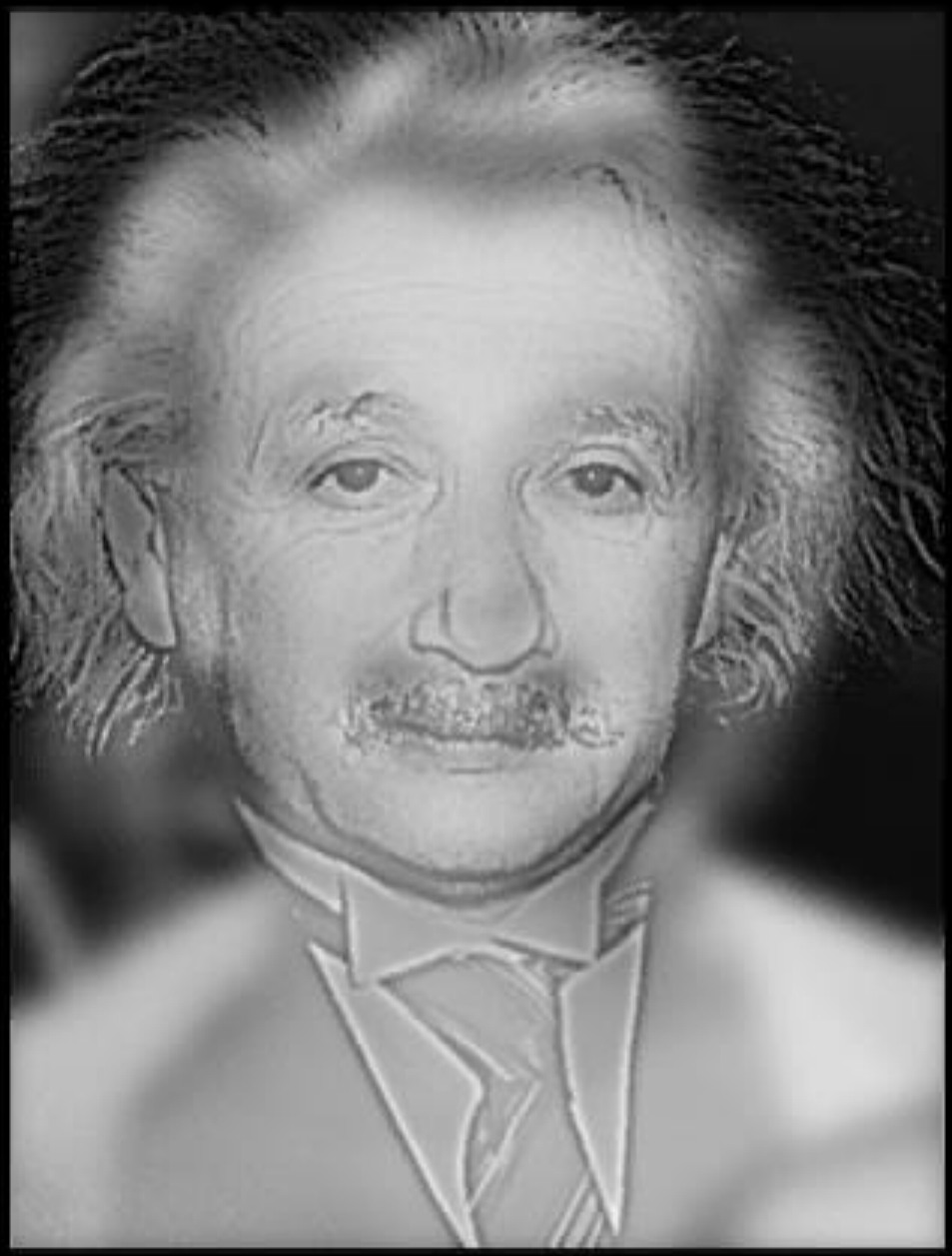
Slide credit: F. Durand

Speed

- Direct bilateral filtering is slow (minutes)

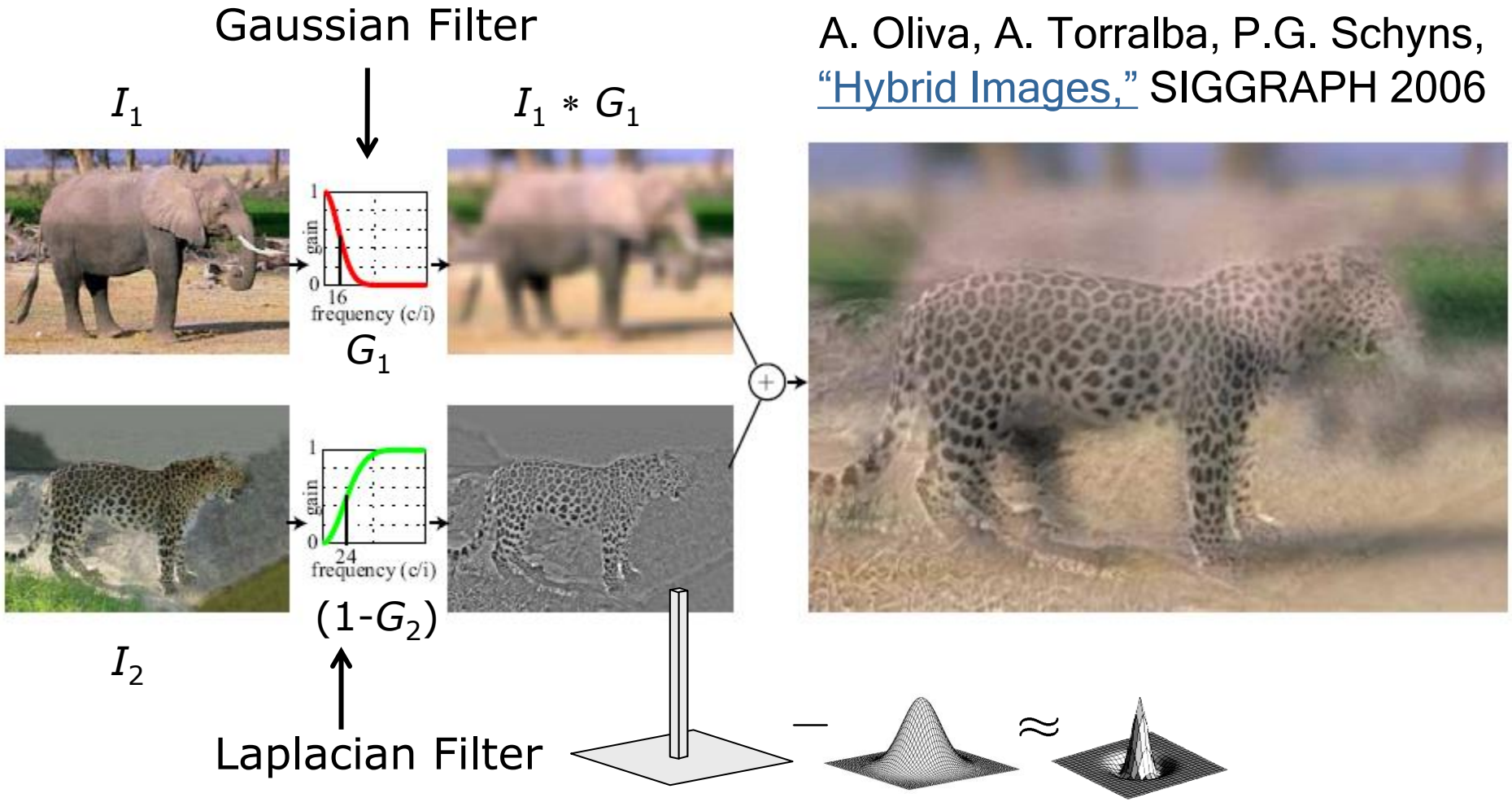
- Accelerations exist:
  - Subsampling in space & range
    - Durand & Dorsey 2002
    - Paris & Durand 2006
  - Limit to box kernel & intelligent maintenance of histogram
    - Weiss 2006

Slide credit: F. Durand



Application: Hybrid Images

A. Oliva, A. Torralba, P.G. Schyns, "Hybrid Images," SIGGRAPH 2006

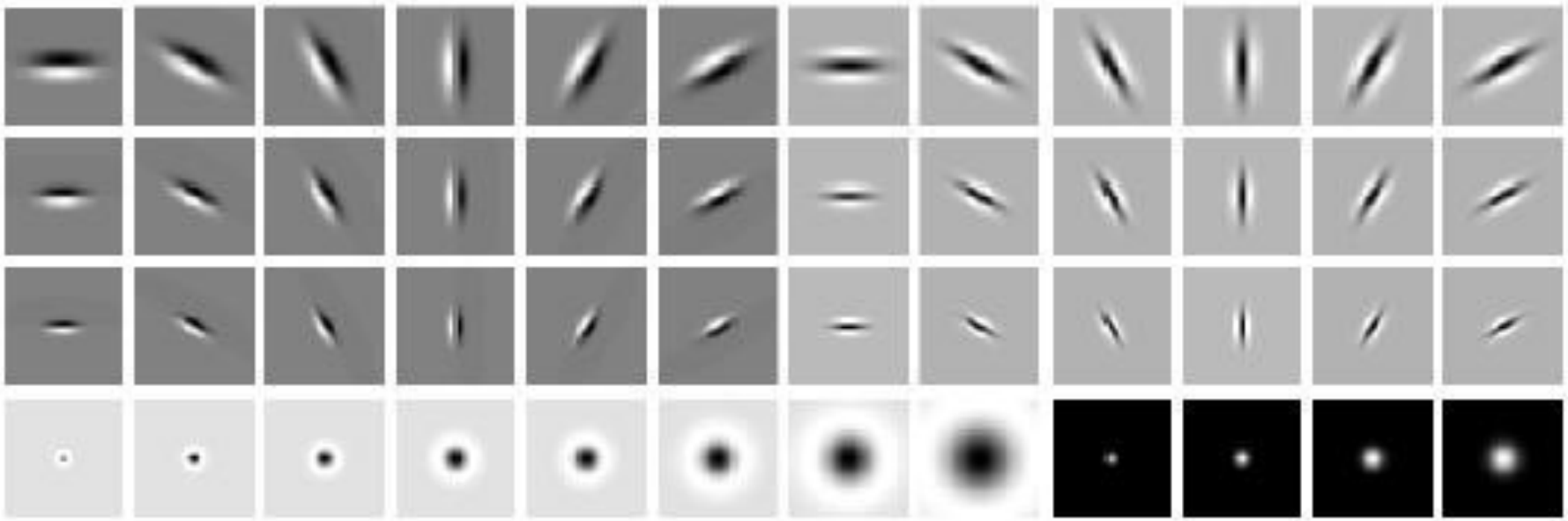


Project Instructions: http://www.cs.illinois.edu/class/fa10/cs498dwh/projects/hybrid/ComputationalPhotography_ProjectHybrid.html

Slide credit: C. Dyer

Application: Filter Banks for Feature Detection

LM Filter Bank

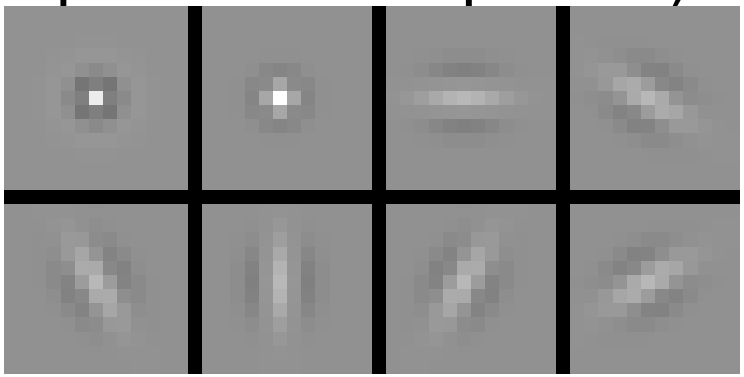


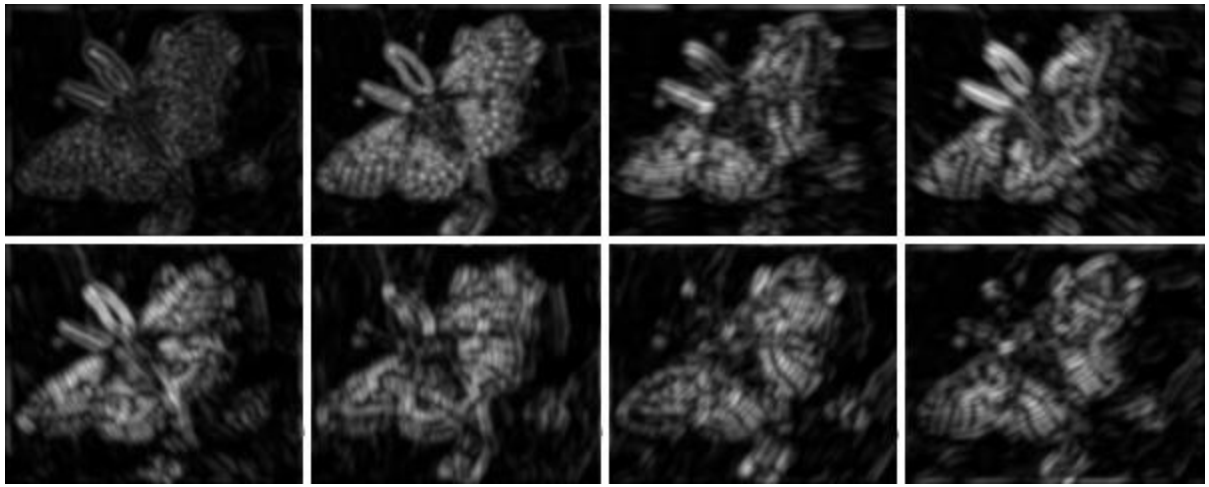
Code for filter banks: *www.robots.ox.ac.uk/~vgg/research/texclass/filters.html*

Slide credit: C. Dyer

Filter Banks

- Process image with each filter and keep responses (or squared/abs responses)





Slide credit: C. Dyer

Feature detection

- Edge
- Corner
- Blob

Edge detection

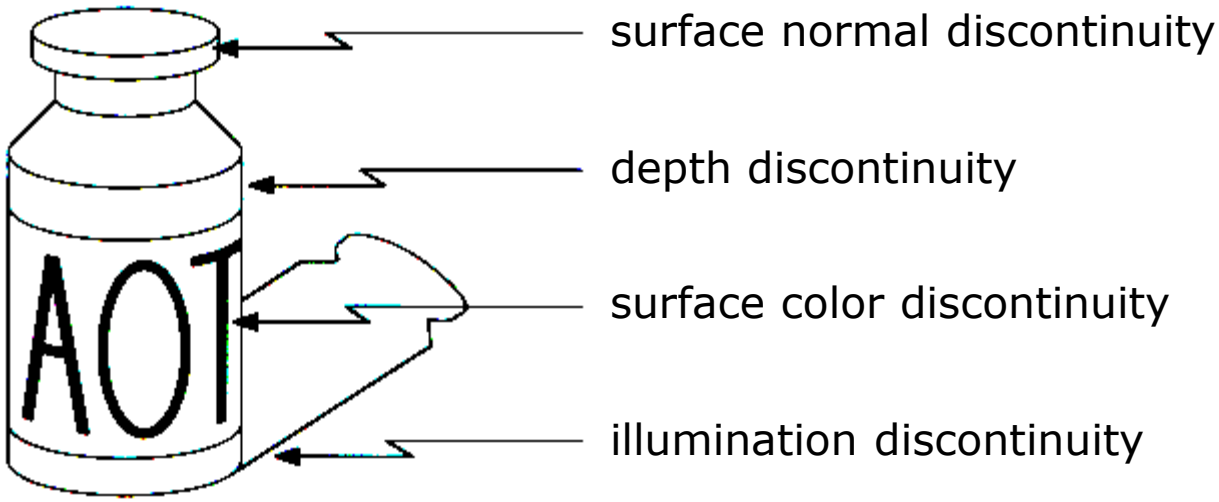
- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels

- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Source: D. Lowe

Origin of edges

Edges are caused by a variety of factors:

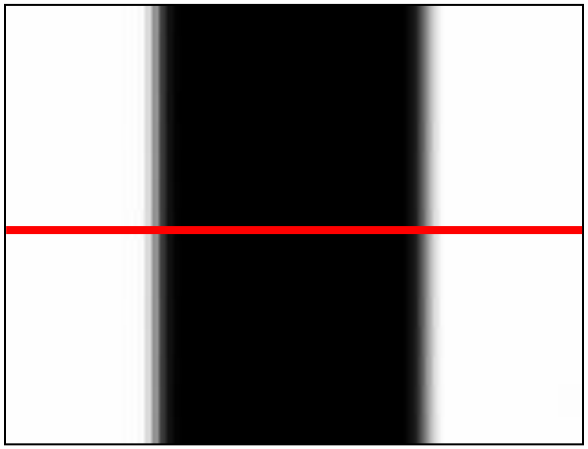


Source: Steve Seitz

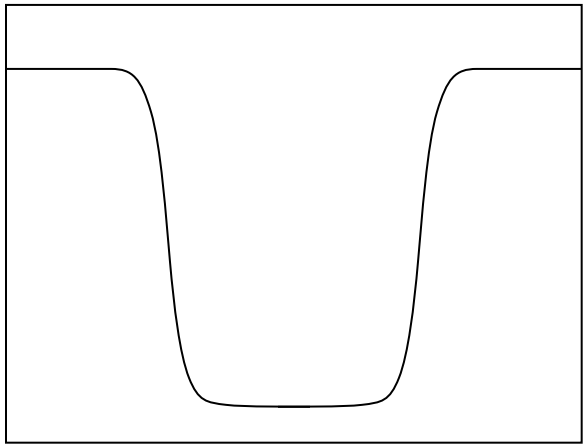
Characterizing edges

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

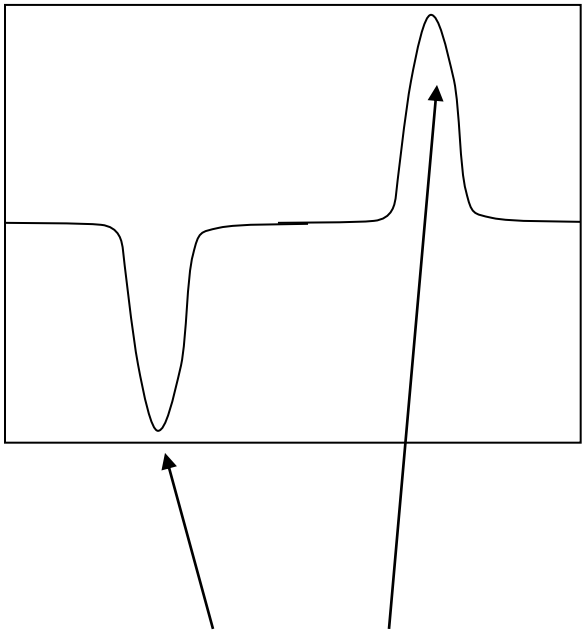
image



intensity function
(along horizontal scan line)



first derivative



edges correspond to
extrema of derivative

Source: S. Lazebnik

Derivatives with convolution

For 2D function f(x,y), the partial derivative is:

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\varepsilon \to 0} \frac{f(x+\varepsilon, y) - f(x,y)}{\varepsilon}$$

For discrete data, we can approximate using finite differences:

$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1, y) - f(x,y)}{1}$$

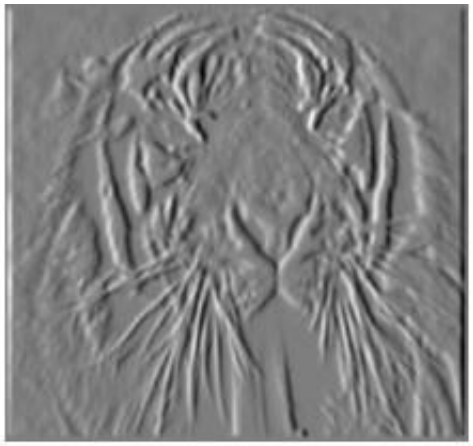
To implement above as convolution, what would be the associated filter?

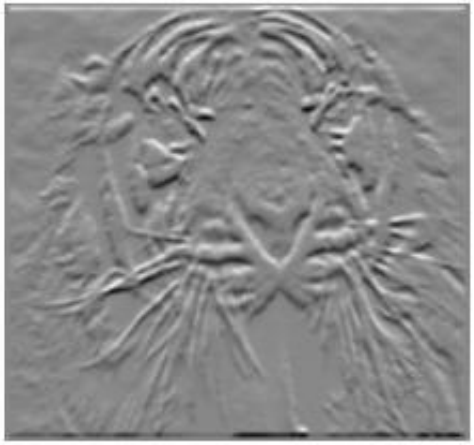
Source: K. Grauman

Partial derivatives of an image

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

-1	1





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

-1
1

or

1
-1

Which shows changes with respect to x?

Source: S. Lazebnik

Finite difference filters

Other approximations of derivative filters exist:

Prewitt: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

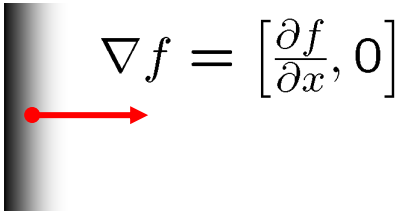
Sobel: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

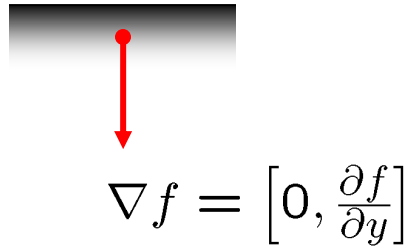
Roberts: $M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

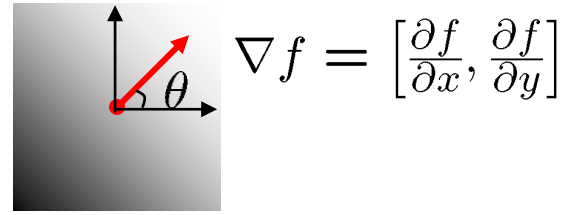
Source: K. Grauman

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

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

The gradient direction is given by $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$

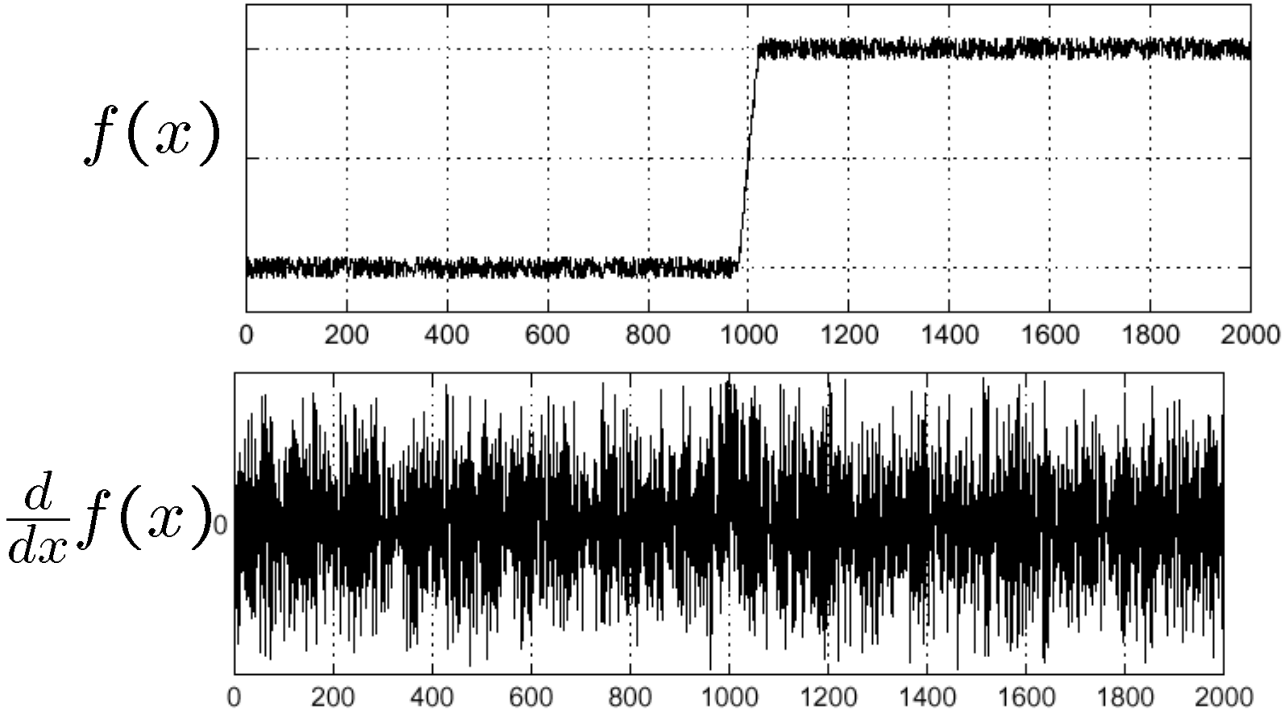
The *edge strength* is given by the gradient magnitude

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

Source: Steve Seitz

Effects of noise

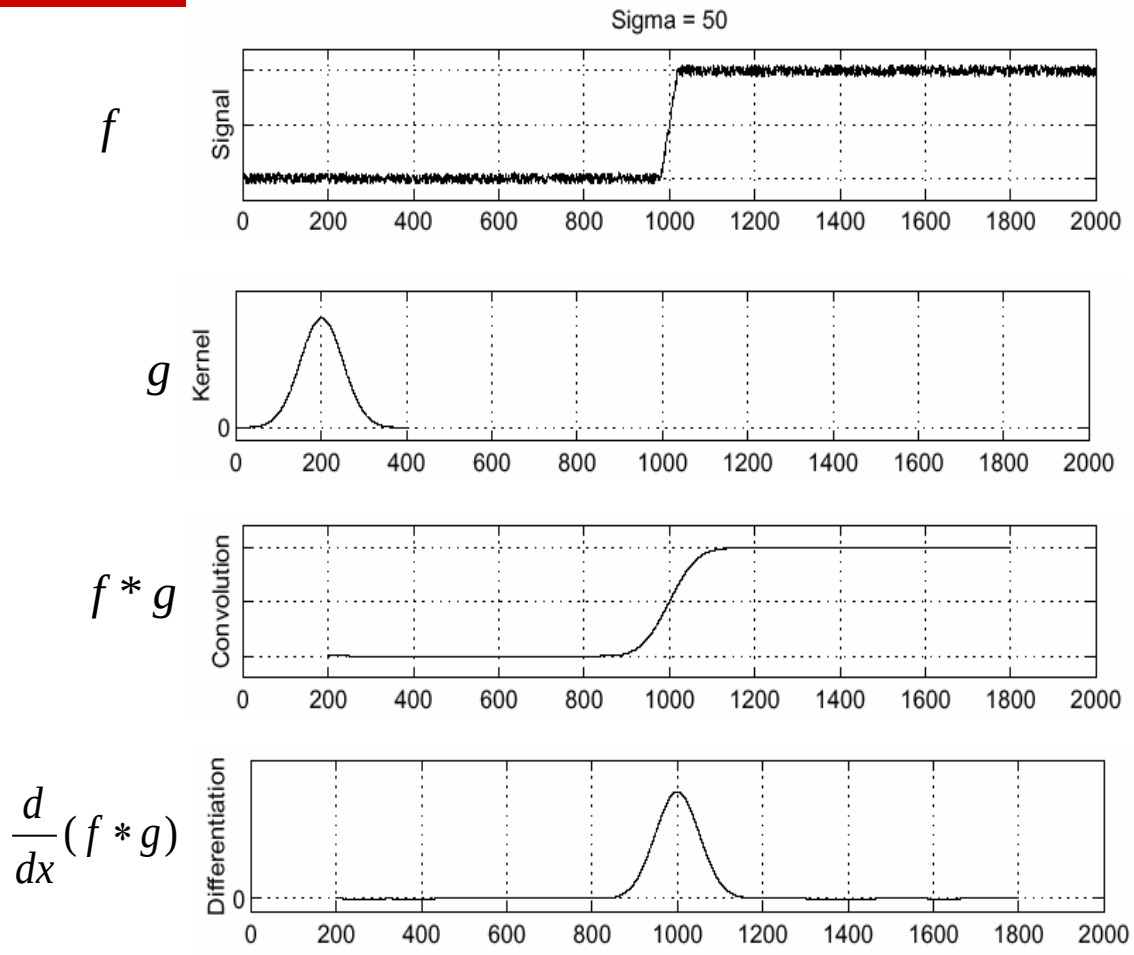
- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal



Where is the edge?

Source: S. Seitz

Solution: smooth first



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

Source: S. Seitz

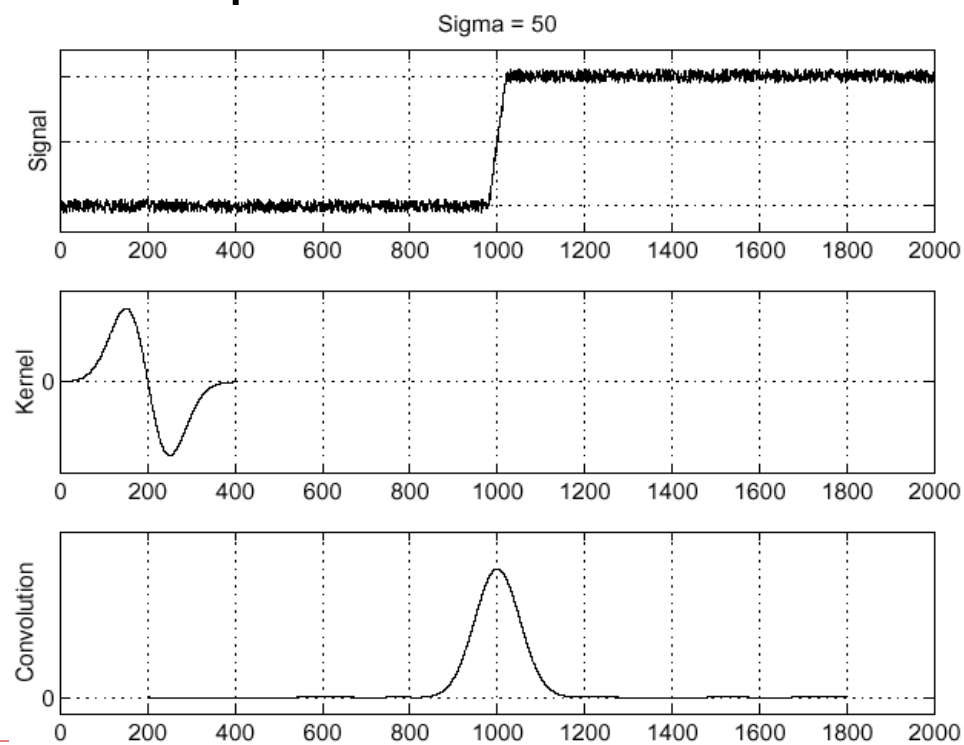
Derivative theorem of convolution

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

f

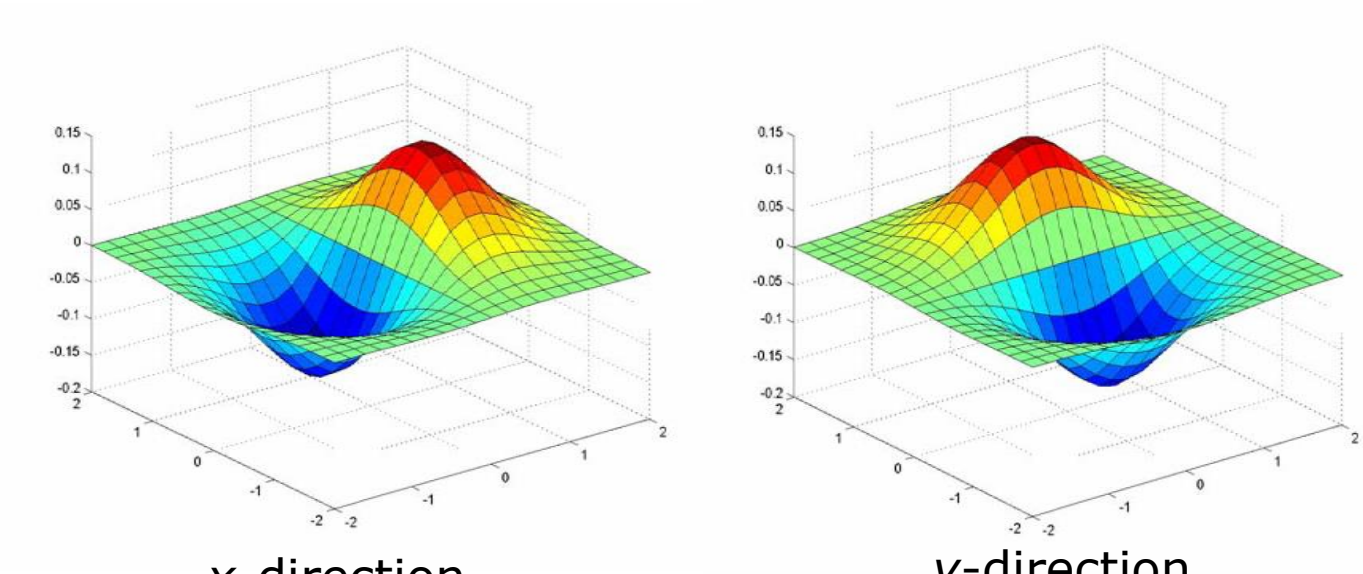
$\frac{d}{dx} g$

$f * \frac{d}{dx} g$



Source: S. Seitz

Derivative of Gaussian filter



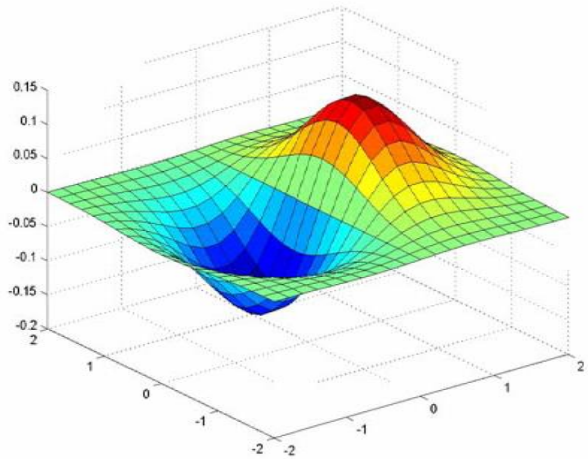
x-direction

y-direction

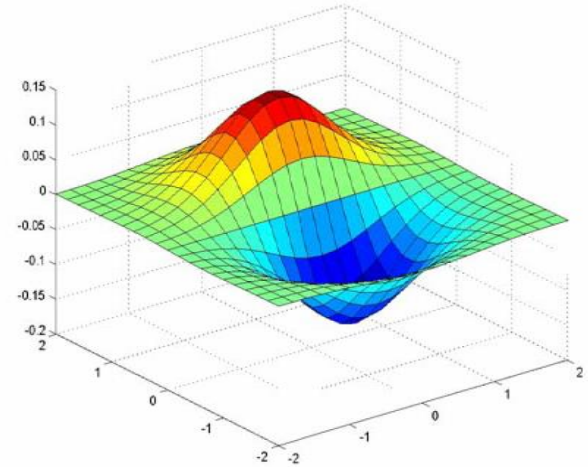
Are these filters separable?

Source: S. Lazebnik

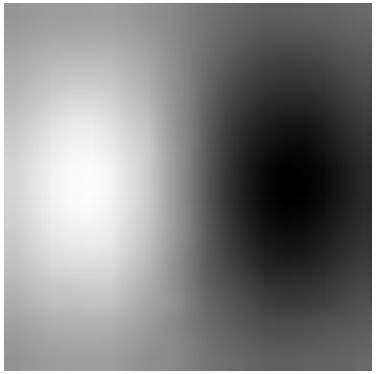
Derivative of Gaussian filter

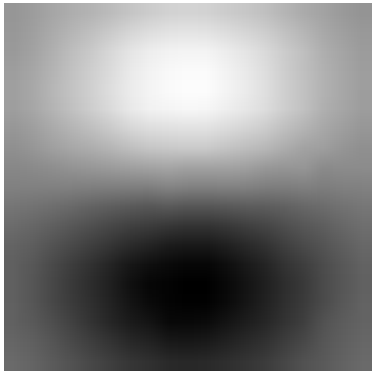


x-direction



y-direction

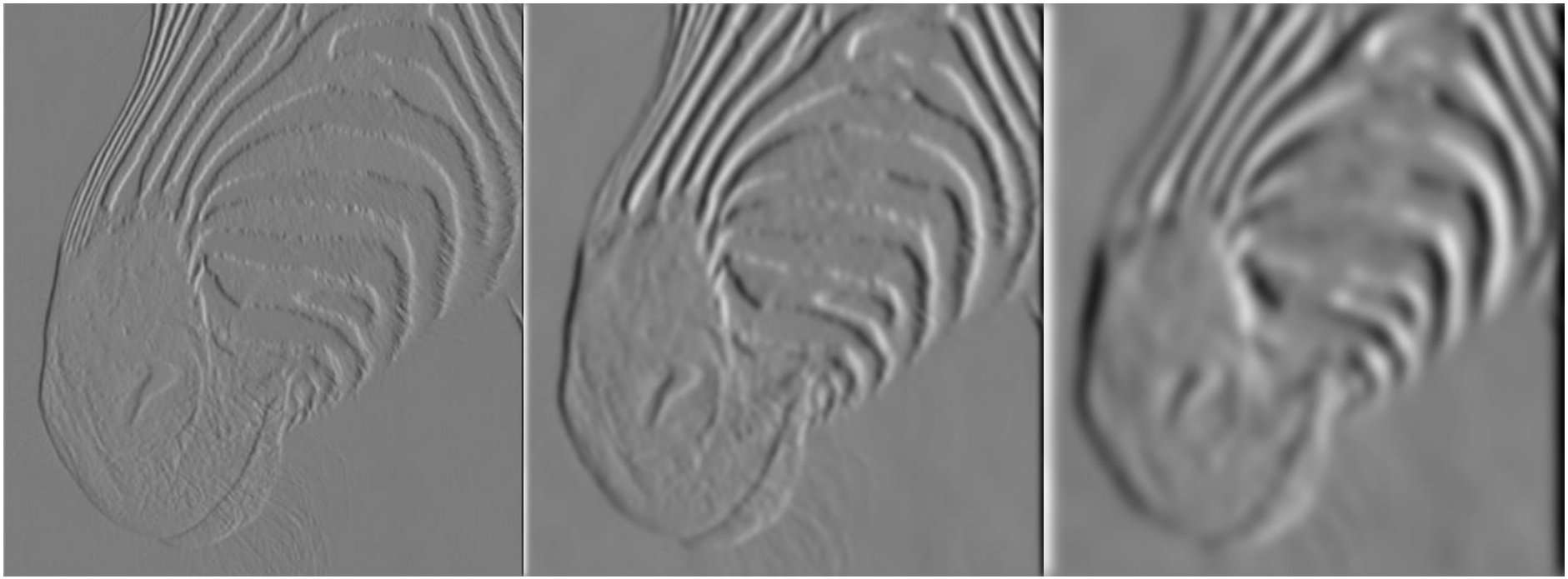




Which one finds horizontal/vertical edges?

Source: S. Lazebnik

Scale of Gaussian derivative filter

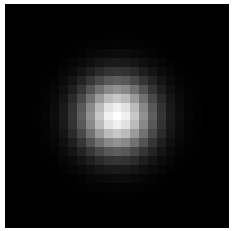


1 pixel 3 pixels 7 pixels

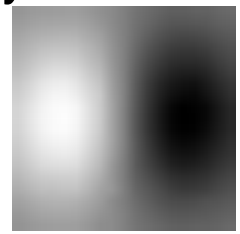
Smoothed derivative removes noise, but blurs edge. Also finds edges at different “scales”

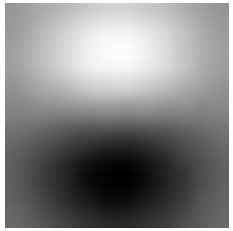
Source: D. Forsyth

Review: smoothing vs. derivative filters



- Smoothing filters
  - Gaussian: remove “high-frequency” components; “low-pass” filter
  - Can the values of a smoothing filter be negative?
  - What should the values sum to?
    - **One:** constant regions are not affected by the filter





- Derivative filters
  - Derivatives of Gaussian
  - Can the values of a derivative filter be negative?
  - What should the values sum to?
    - **Zero:** no response in constant regions
  - High absolute value at points of high contrast

Source: K. Grauman and S. Lazebnik

The Canny edge detector

original image

Slide credit: Steve Seitz

The Canny edge detector

norm of the gradient

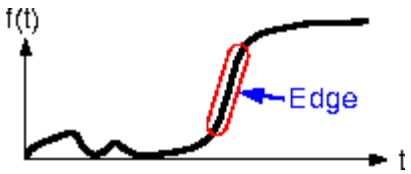
Slide credit: Steve Seitz

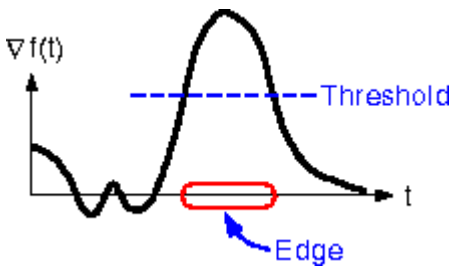
The Canny edge detector

thresholding

Slide credit: Steve Seitz

The Canny edge detector



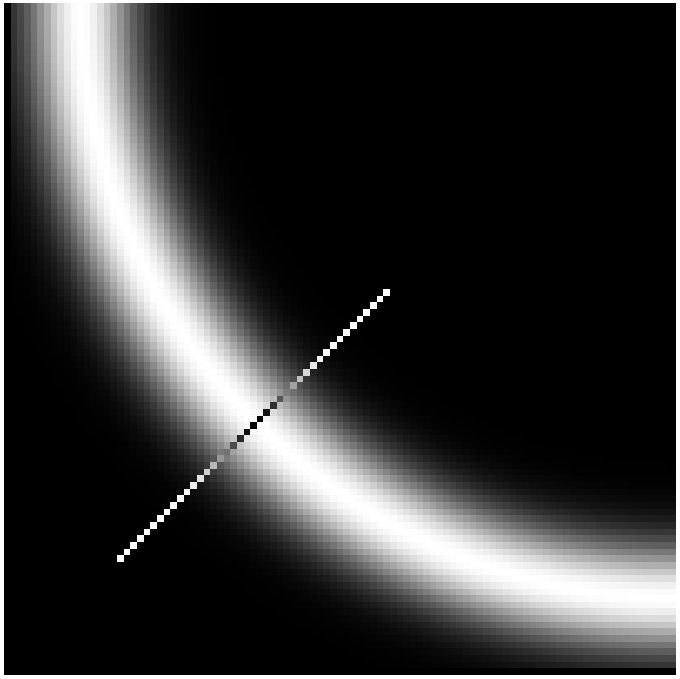


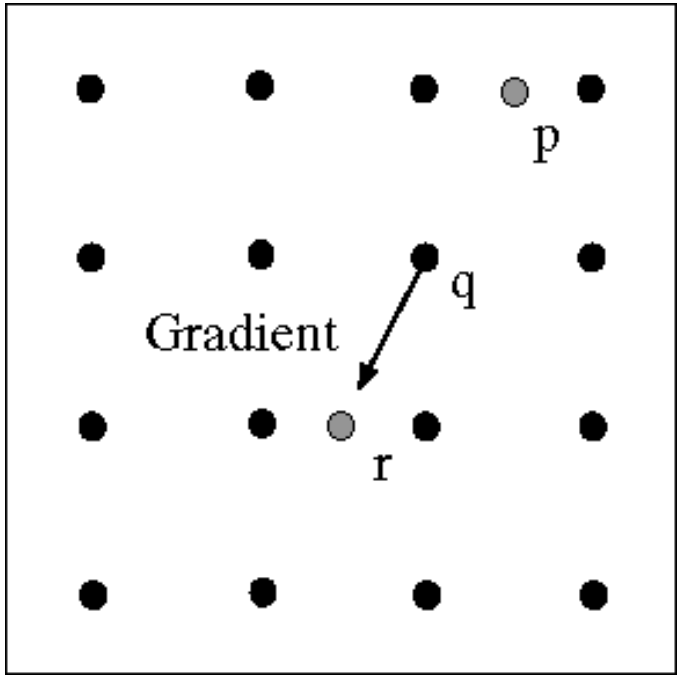
How to turn these thick regions of the gradient into curves?

thresholding

Slide source: Steve Seitz, S. Lazebnik

Non-maximum suppression





Check if pixel is local maximum along gradient direction, select single max across width of the edge

- requires checking interpolated pixels p and r

Slide source: Steve Seitz, S. Lazebnik

The Canny edge detector

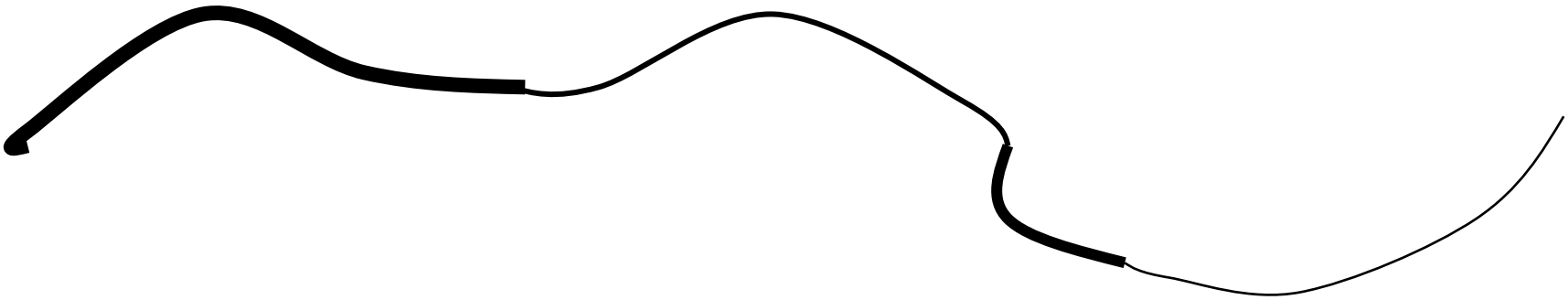
Thinning
(non-maximum suppression)

Problem: pixels along this edge didn't survive the thresholding

Slide source: Steve Seitz, S. Lazebnik

Hysteresis thresholding

Use a high threshold to start edge curves, and a low threshold to continue them.



Source: Steve Seitz

Hysteresis thresholding

original image

high threshold
(strong edges)

low threshold
(weak edges)

hysteresis threshold

Source: L. Fei-Fei

Recap: Canny edge detector

1. Filter image with derivative of Gaussian
2. Find magnitude and orientation of gradient
3. **Non-maximum suppression**:
   - Thin wide “ridges” down to single pixel width
4. **Linking and thresholding** (**hysteresis**):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them

J. Canny, ***A Computational Approach To Edge Detection***, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

Source: D. Lowe, L. Fei-Fei

Next Time

- More feature detection
  - Corner and blob