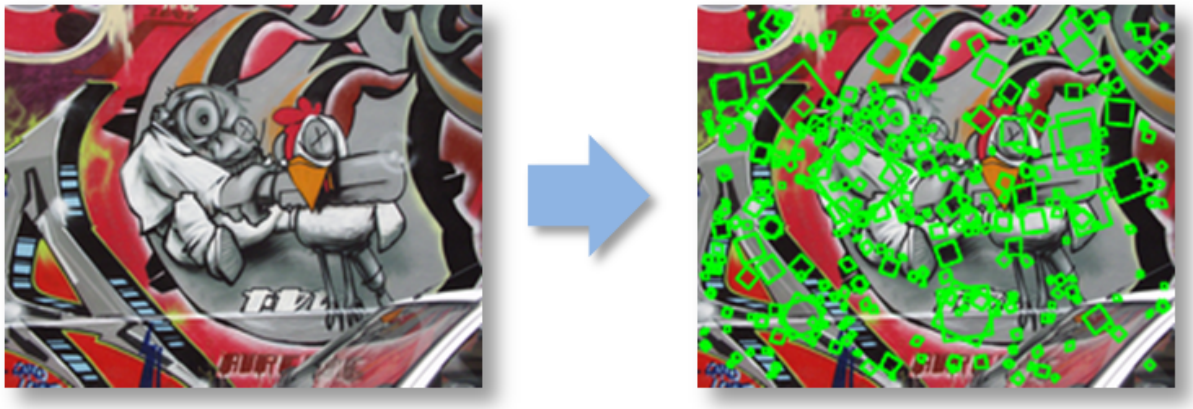
CS4670/5670: Computer Vision

Kavita Bala

Lecture 8: Edges and Feature Detection

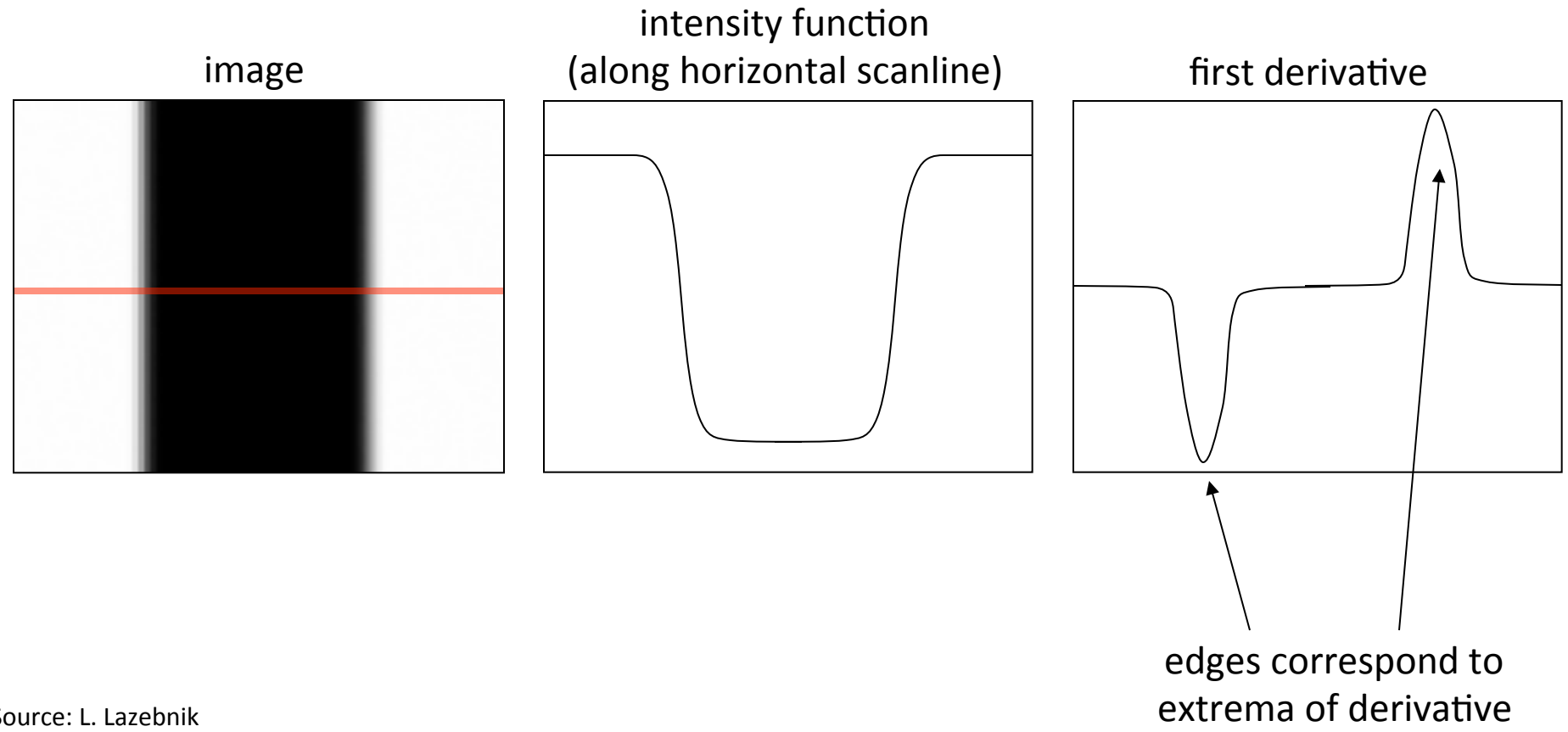


Announcements

- PA 1 (A) is PA 1
- Winter break
- A quiz some time after break

Characterizing edges

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



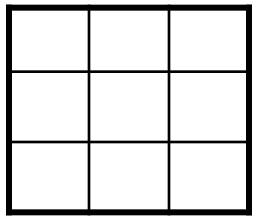
Source: L. Lazebnik

Image derivatives

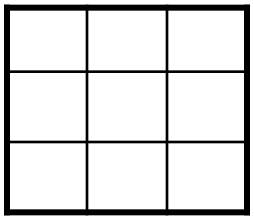
- How can we differentiate a *digital* image F[x,y]?
  - Take discrete derivative (finite difference)

$$\frac{\partial f}{\partial x}[x,y] \approx F[x+1,y] - F[x,y]$$
$$\frac{\partial f}{\partial x}[x,y] \approx \frac{F[x+1,y] - F[x-1,y]}{2}$$

How would you implement this as a linear filter?

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

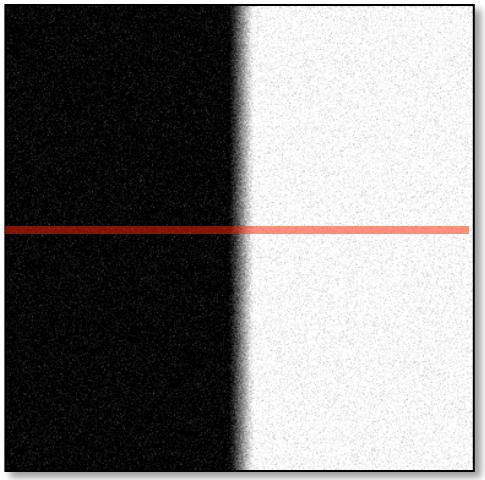
H_x

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

H_y

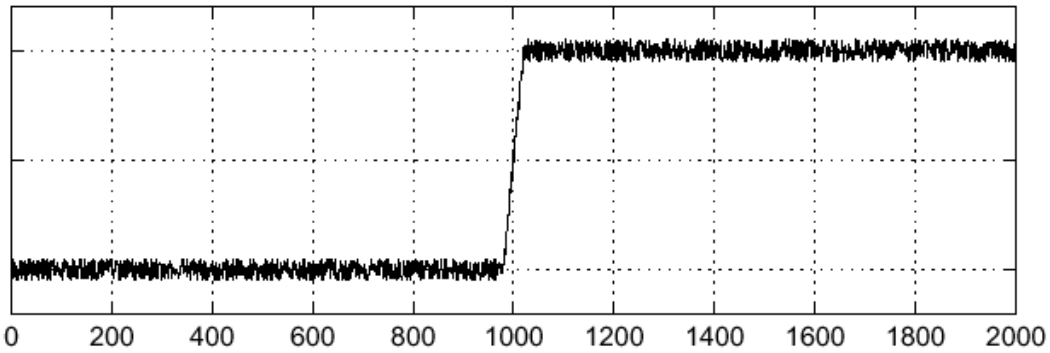
Source: S. Seitz

Effects of noise

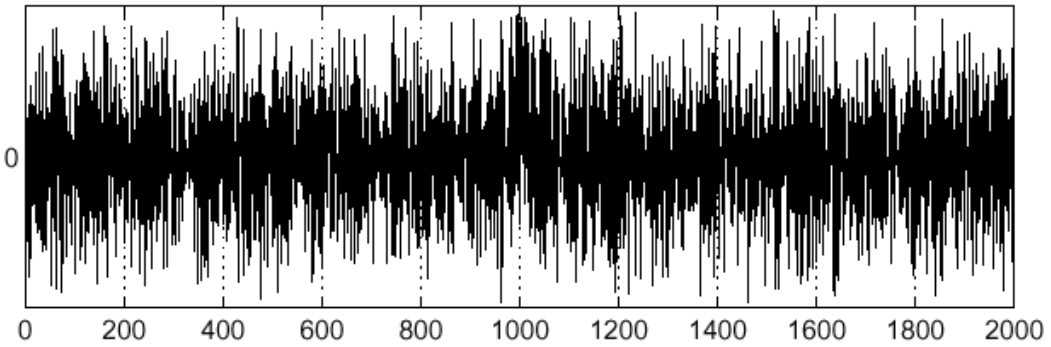


Noisy input image

$f(x)$



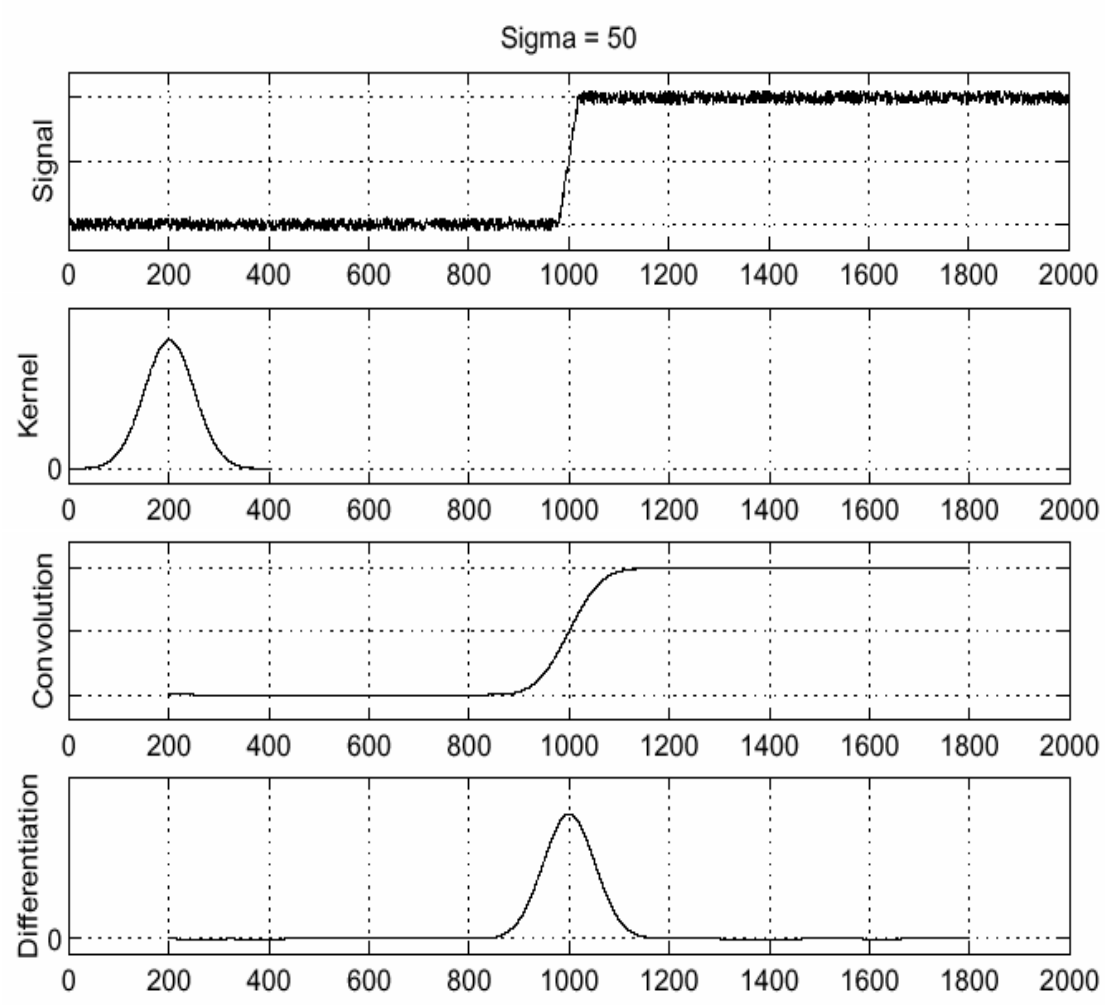
$\frac{d}{dx}f(x)$



Where is the edge?

Source: S. Seitz

Solution: smooth first



f

h

$f * h$

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

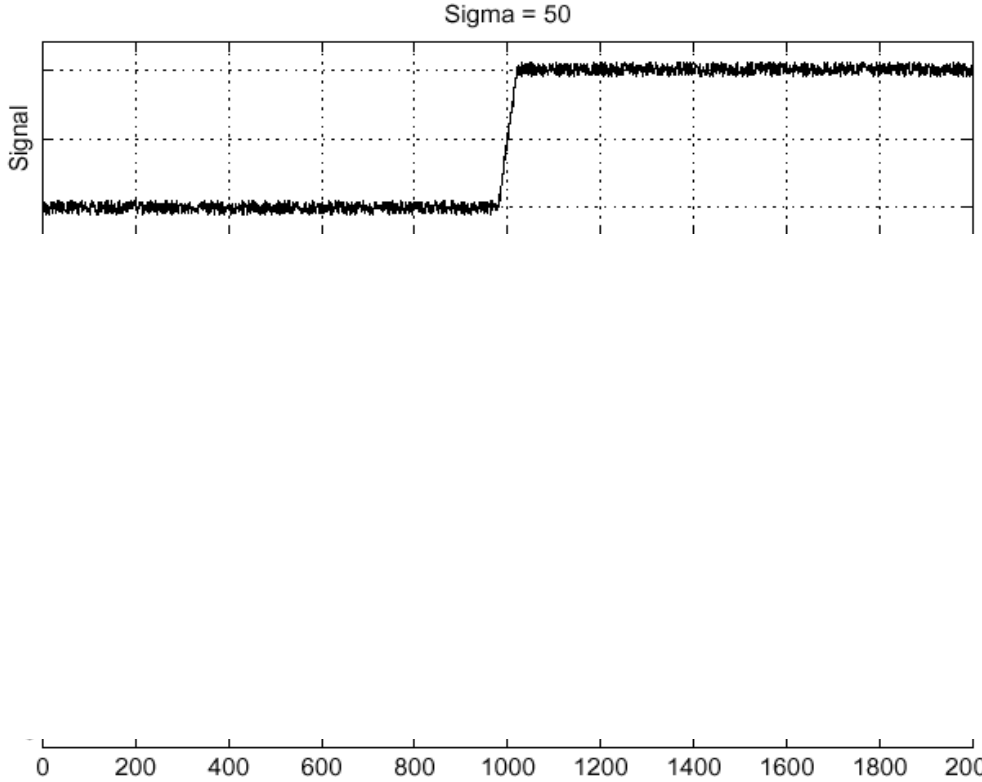
To find edges, look for peaks in $\frac{d}{dx}(f * h)$

Source: S. Seitz

Associative property of convolution

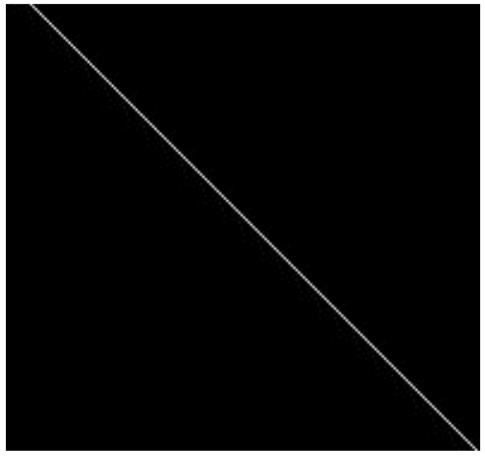
- Differentiation is a convolution
- Convolution is associative: $\frac{d}{dx}(f * h) = f * \frac{d}{dx}h$
- This saves us one operation:

f



Source: S. Seitz

Image with Edge



Edge Location

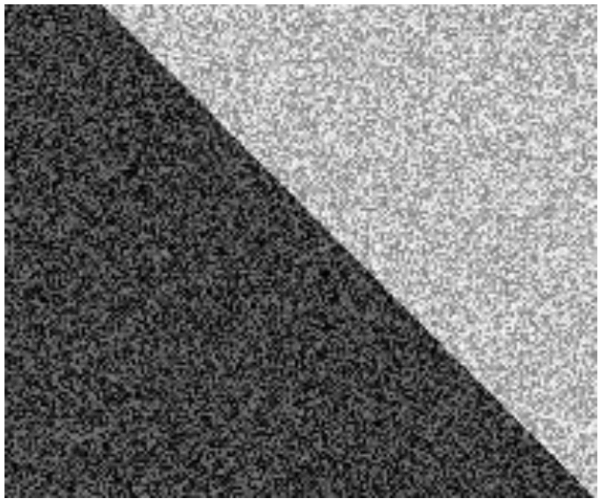
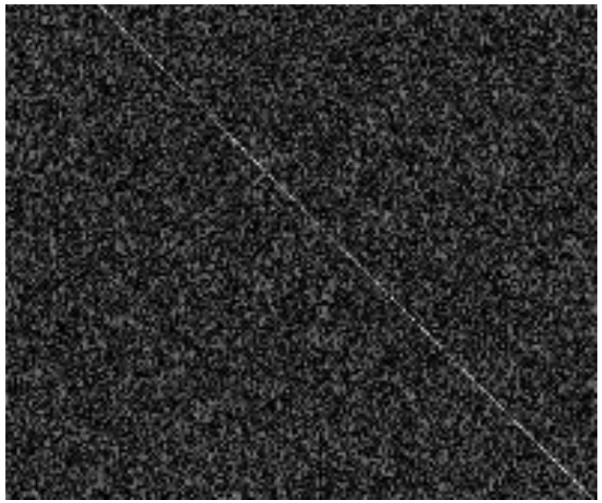
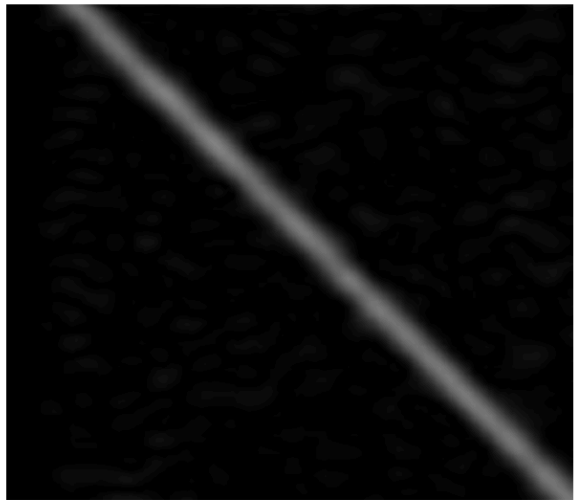


Image + Noise



Derivatives detect
edge *and* noise

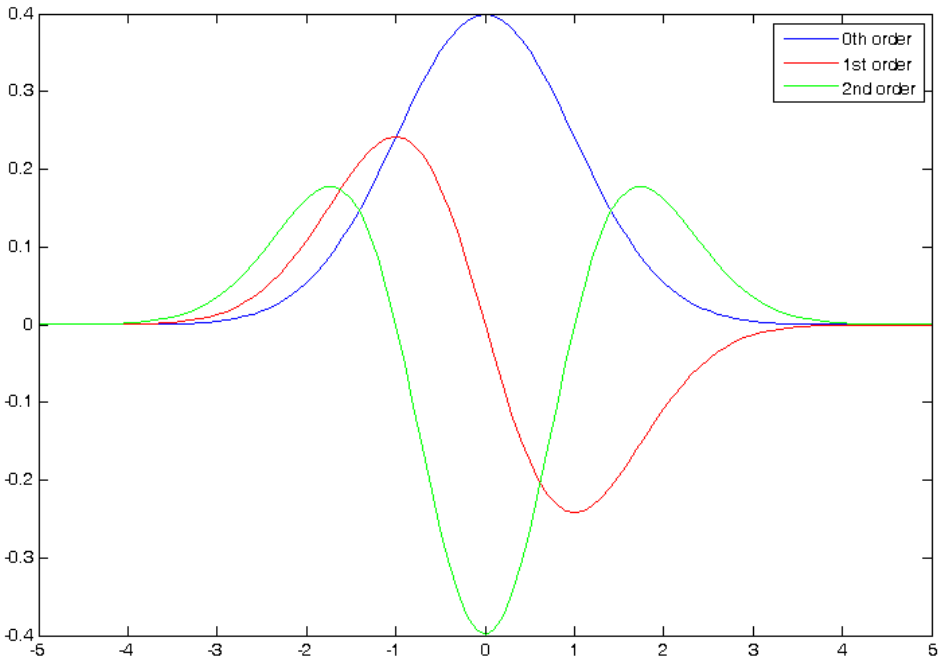


Smoothed derivative removes
noise, but blurs edge

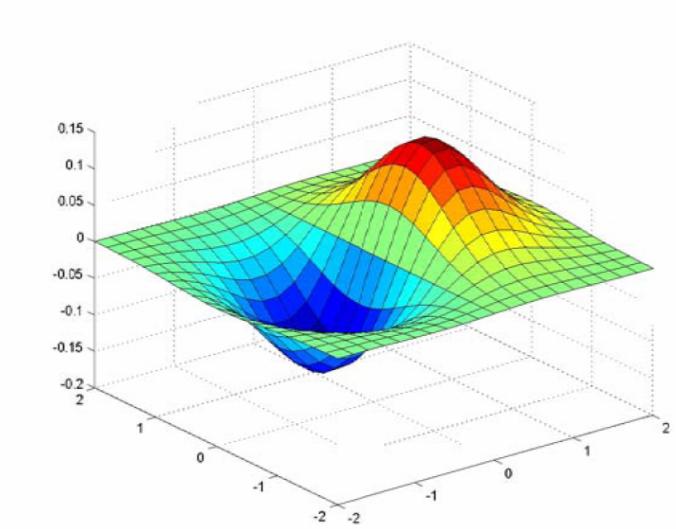
The 1D Gaussian and its derivatives

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

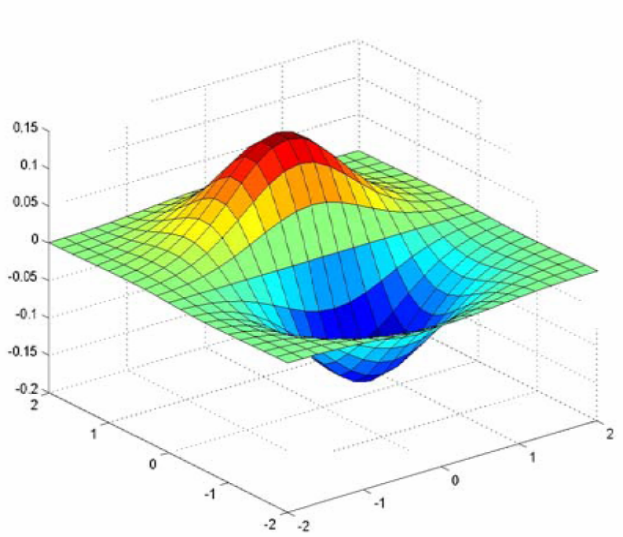
$$G'_\sigma(x) = \frac{d}{dx} G_\sigma(x) = -\frac{1}{\sigma}\left(\frac{x}{\sigma}\right) G_\sigma(x)$$



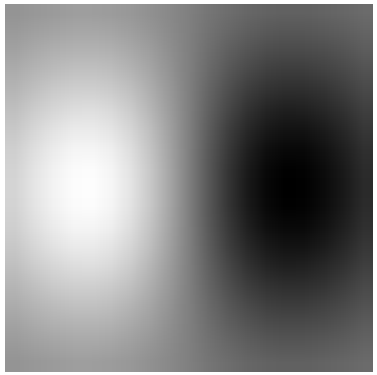
Derivative of Gaussian filter

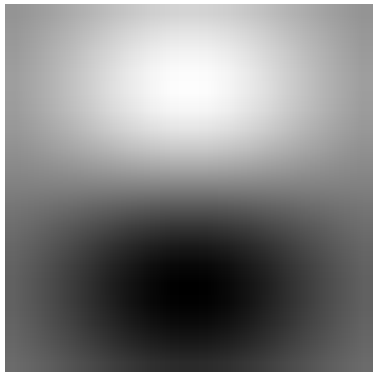


x-direction

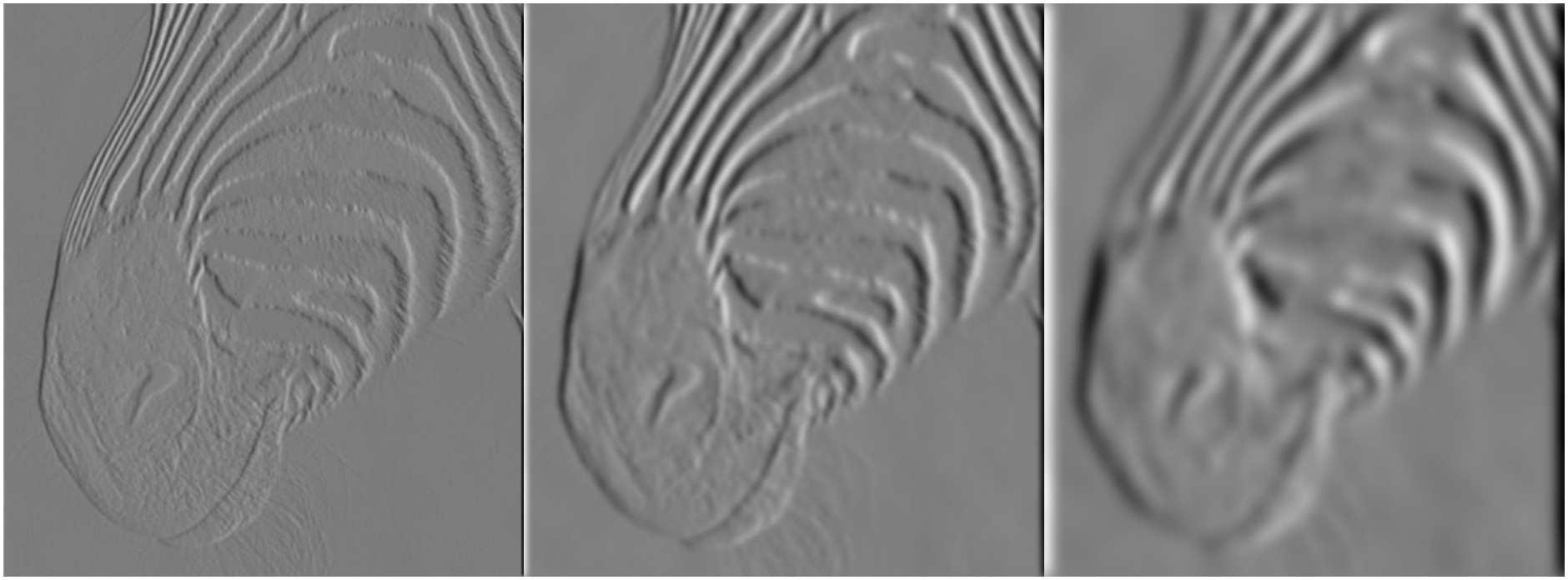


y-direction





Tradeoff between smoothing and localization



1 pixel 3 pixels 7 pixels

- Smoothed derivative removes noise, but blurs edge. Also finds edges at different "scales".

Source: D. Forsyth

The Sobel operator

- Common approximation of derivative of Gaussian
  - A mask (not a convolution kernel)

$\frac{1}{8}$

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

s_x

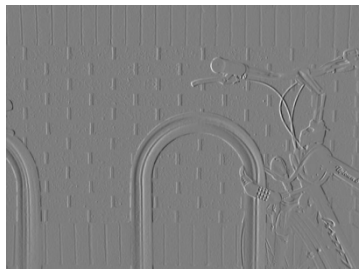
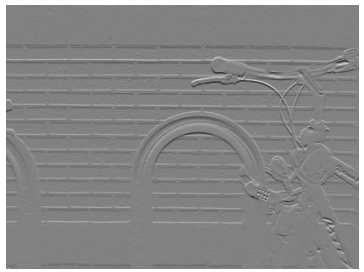
$\frac{1}{8}$

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

s_y

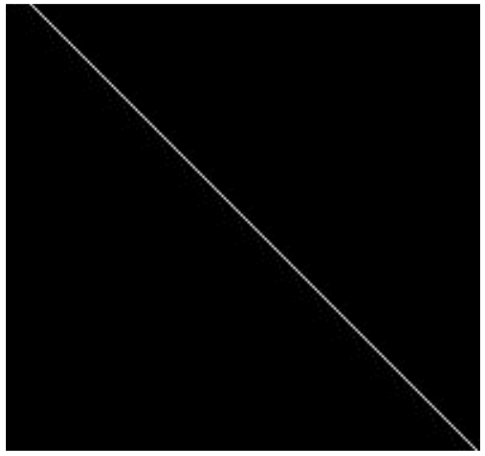
- The standard defn. of the Sobel operator omits the 1/8 term
  - doesn't make a difference for edge detection
  - the 1/8 term **is** needed to get the right gradient magnitude

Sobel operator: example

Source: Wikipedia

Image with Edge



Edge Location

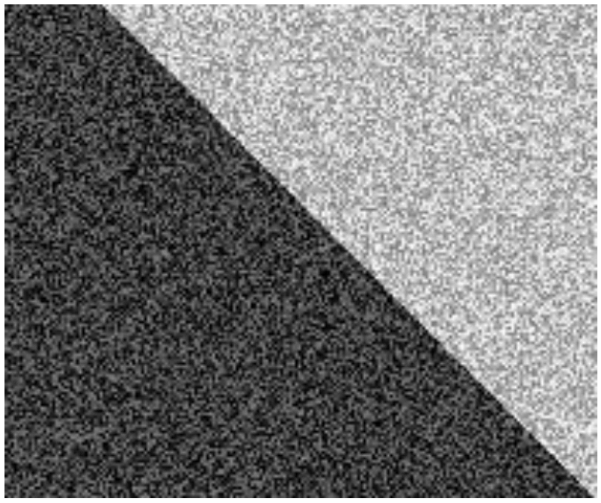
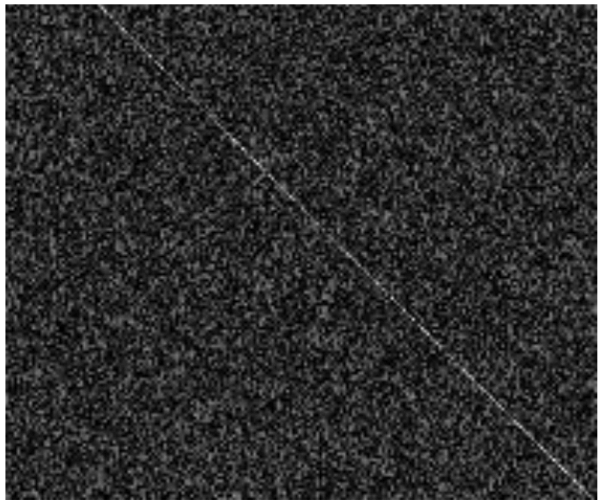
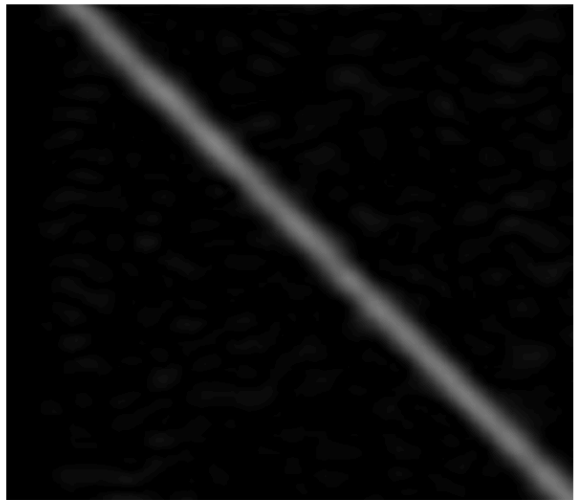


Image + Noise

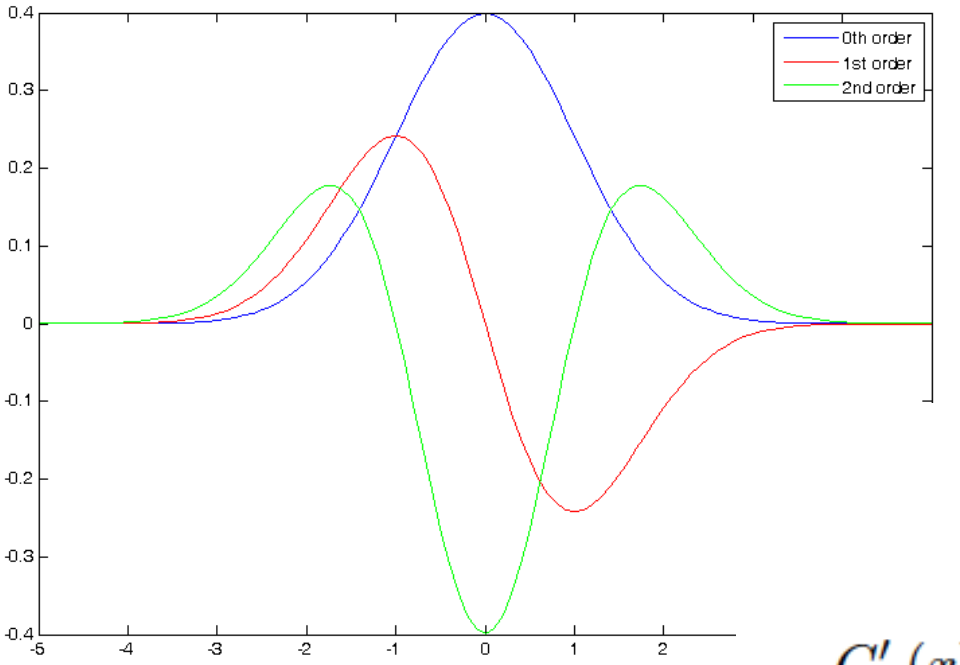


Derivatives detect
edge *and* noise



Smoothed derivative removes
noise, but blurs edge

The 1D Gaussian and its derivatives



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

$$G'_\sigma(x) = \frac{d}{dx} G_\sigma(x) = -\frac{1}{\sigma}\left(\frac{x}{\sigma}\right) G_\sigma(x)$$

$$G''_\sigma(x) = \frac{d^2}{dx^2} G_\sigma(x) = \frac{1}{\sigma^2}\left(\frac{x^2}{\sigma^2} - 1\right) G_\sigma(x)$$

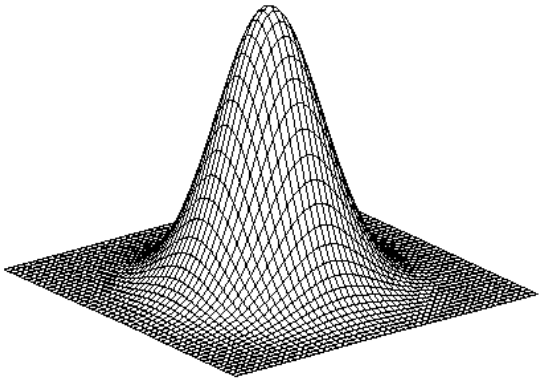
$G'_\sigma(x)$'s maxima/minima occur at $G''_\sigma(x)$'s zeros. And, we can see that $G'_\sigma(x)$ is an odd symmetric function and $G''_\sigma(x)$ is an even symmetric function.

2D edge detection filters

∇^2 is the **Laplacian** operator:

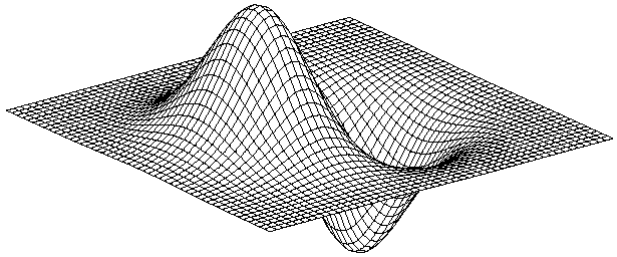
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

2D edge detection filters



Gaussian

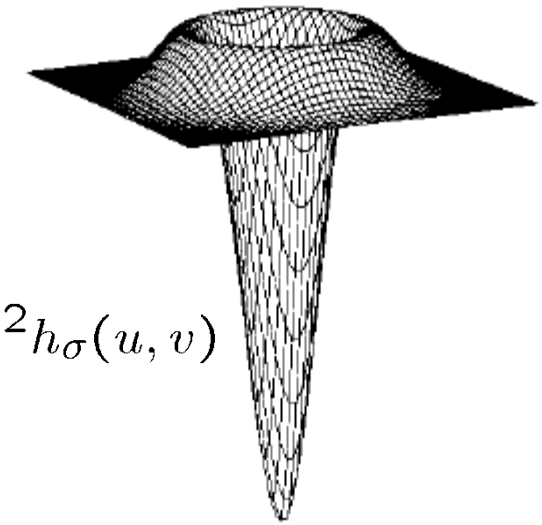
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

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

Laplacian of Gaussian



$$\nabla^2 h_\sigma(u,v)$$

∇^2 is the **Laplacian** operator:

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

Sharpening revisited

- What does blurring take away?

–

=

Let's add it back:

+ α

=

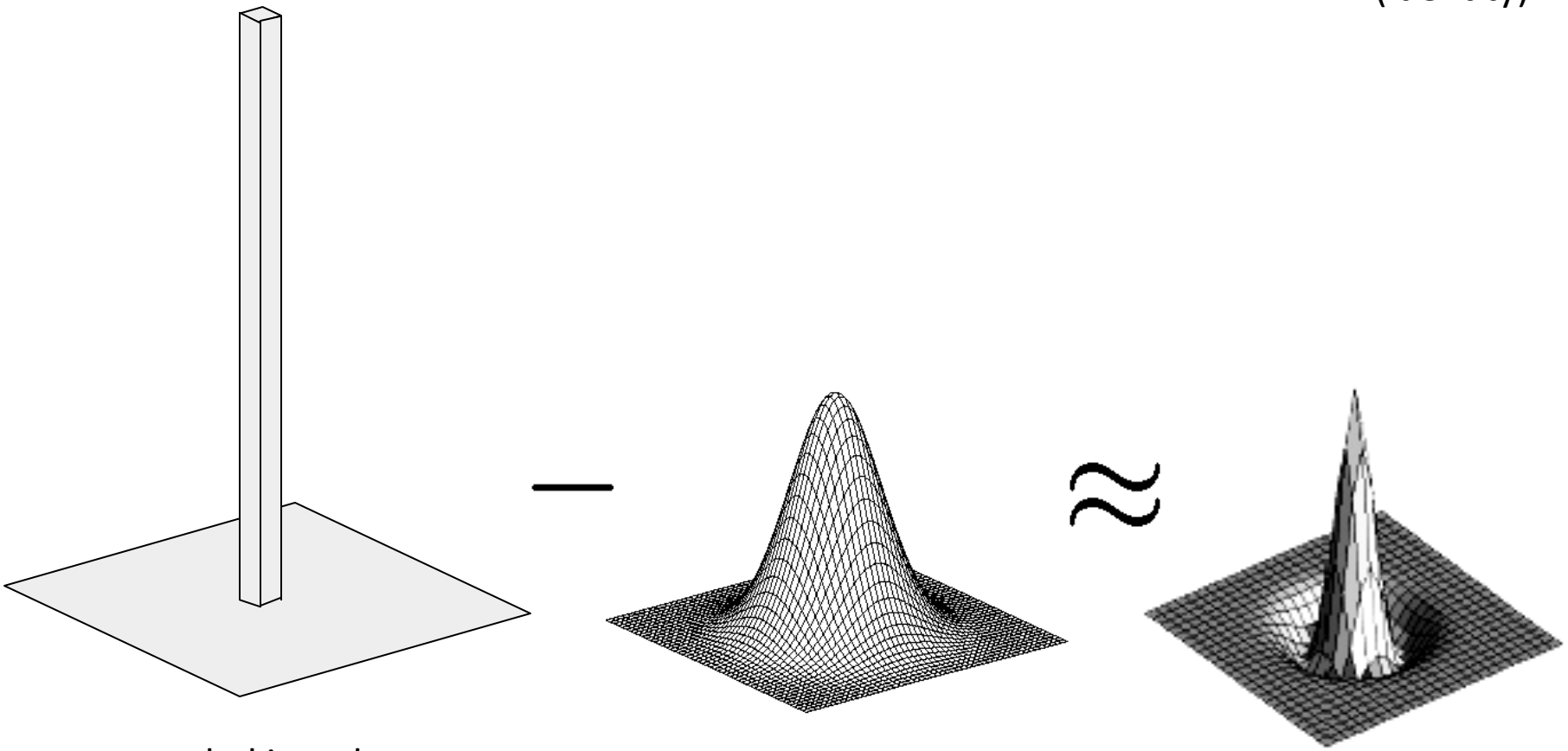
Source: S. Lazebnik

Sharpen filter

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

image

unit impulse (identity)



scaled impulse

Gaussian

Laplacian of Gaussian

Sharpen filter

Super-resolution

Designing an edge detector

- Criteria for a good edge detector:
  - **Good detection:** the optimal detector should find all real edges, ignoring noise or other artifacts
  - **Good localization**
    - the edges detected must be as close as possible to the true edges
    - the detector must return one point only for each true edge point
- Cues of edge detection
  - Differences in color, intensity, or texture across the boundary
  - Continuity and closure
  - High-level knowledge

Source: L. Fei-Fei

Canny edge detector

- This is probably the most widely used edge detector in computer vision
- Theoretical model: step-edges corrupted by additive Gaussian noise

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

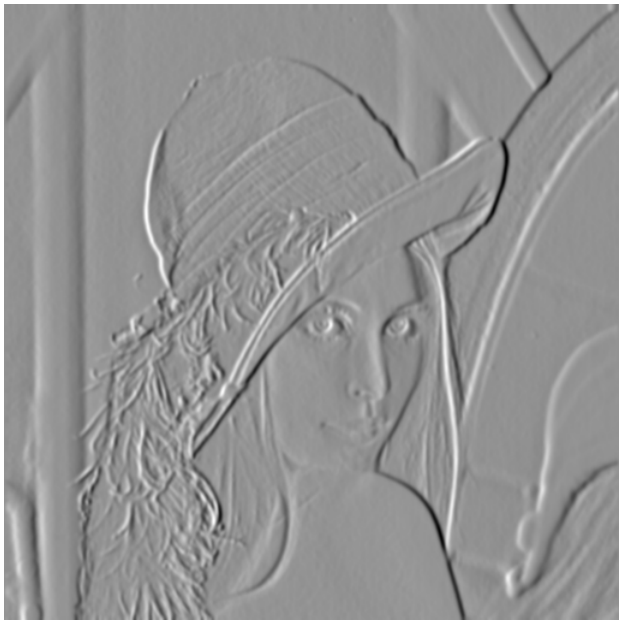
22,000 citations!

Source: L. Fei-Fei

Example

original image (Lena)

Compute Gradients (DoG)



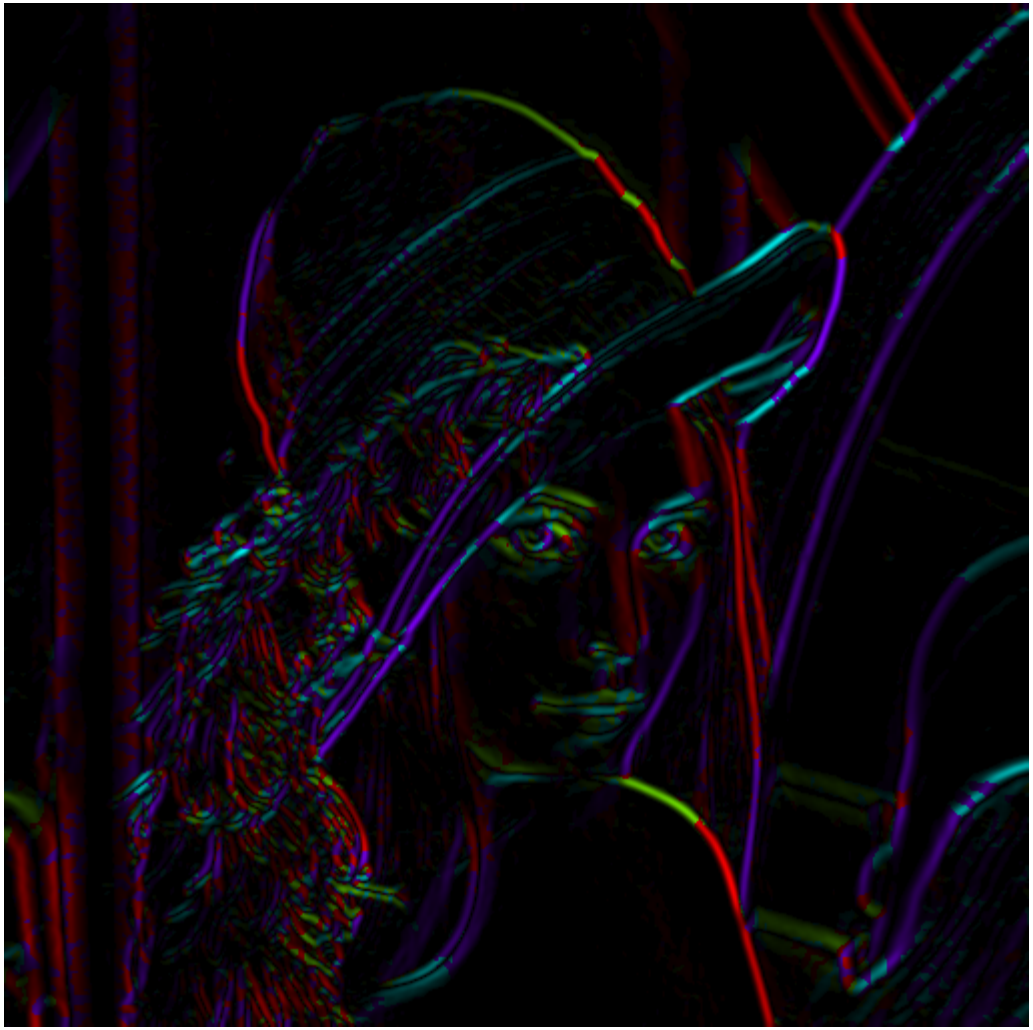
X-Derivative of Gaussian

Y-Derivative of Gaussian

Gradient Magnitude

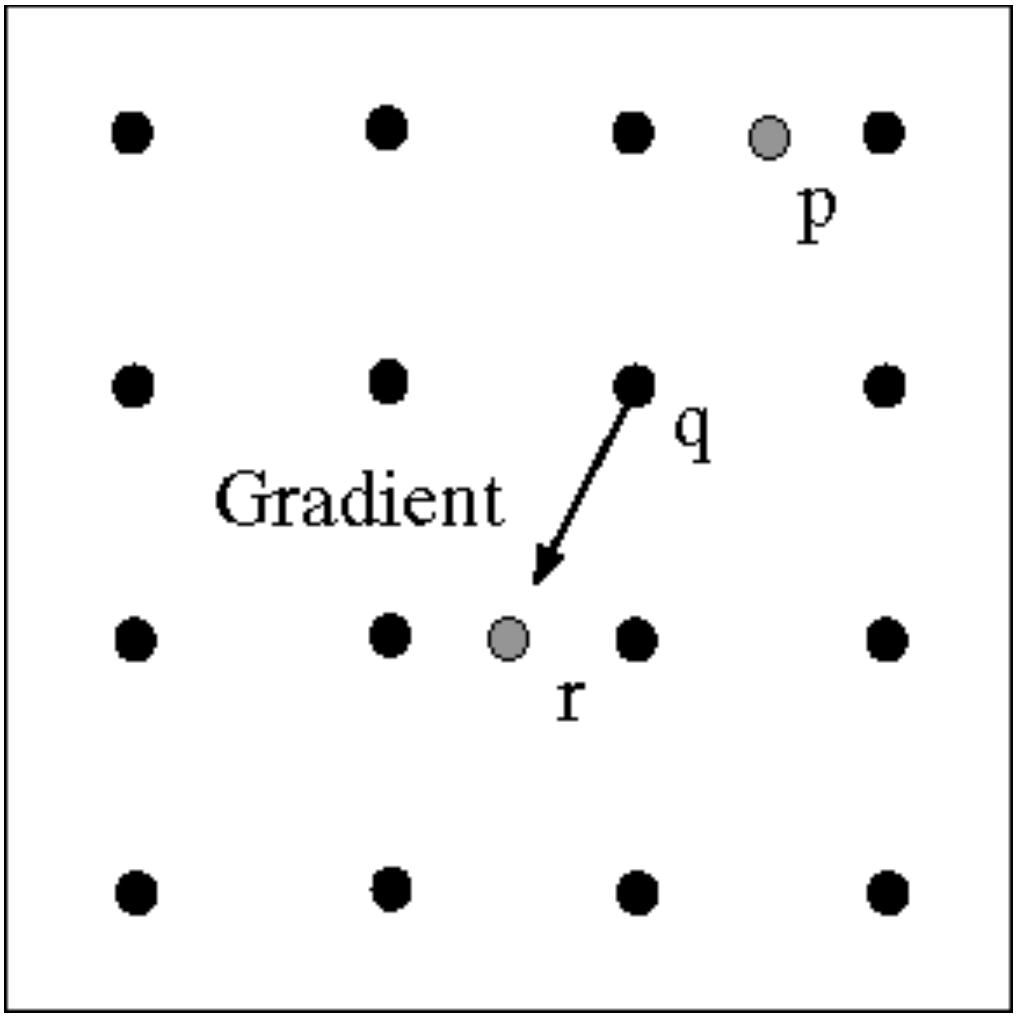
Get Orientation at Each Pixel

- Threshold at minimum level
- Get orientation

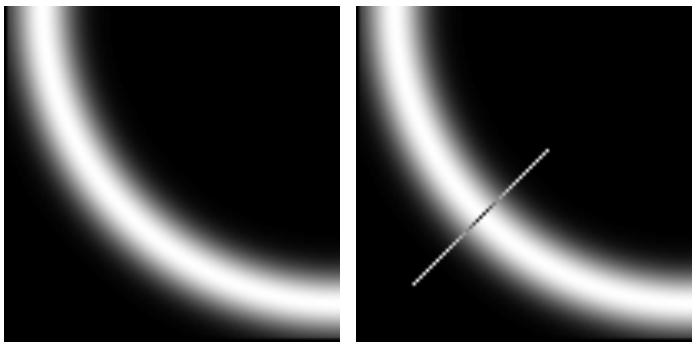


theta = atan2(gy, gx)

Non-maximum suppression for each orientation



At q, we have a maximum if the value is larger than those at both p and at r. Interpolate to get these values.



Source: D. Forsyth

Before Non-max Suppression

After Non-max Suppression

Hysteresis thresholding

- Threshold at low/high levels to get weak/strong edge pixels
- Do connected components, starting from strong edge pixels

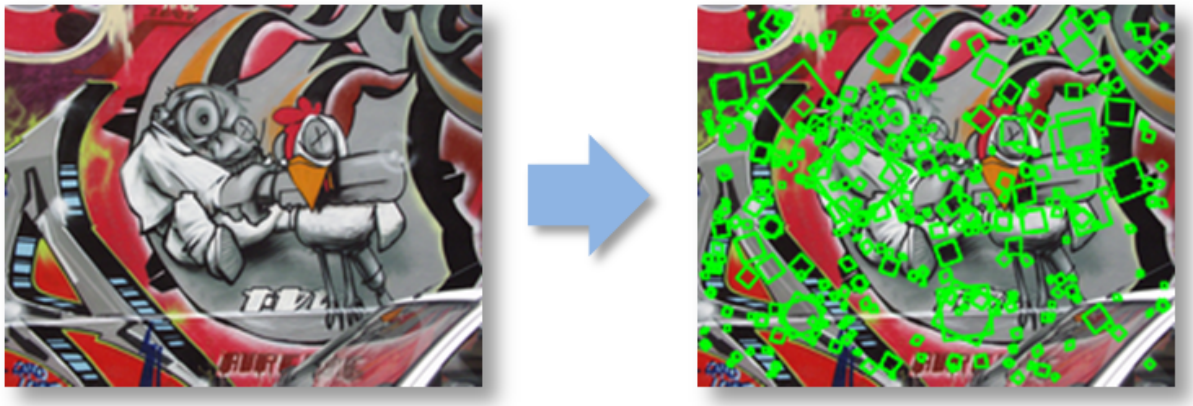
Final Canny Edges

Canny edge detector

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

Source: D. Lowe, L. Fei-Fei

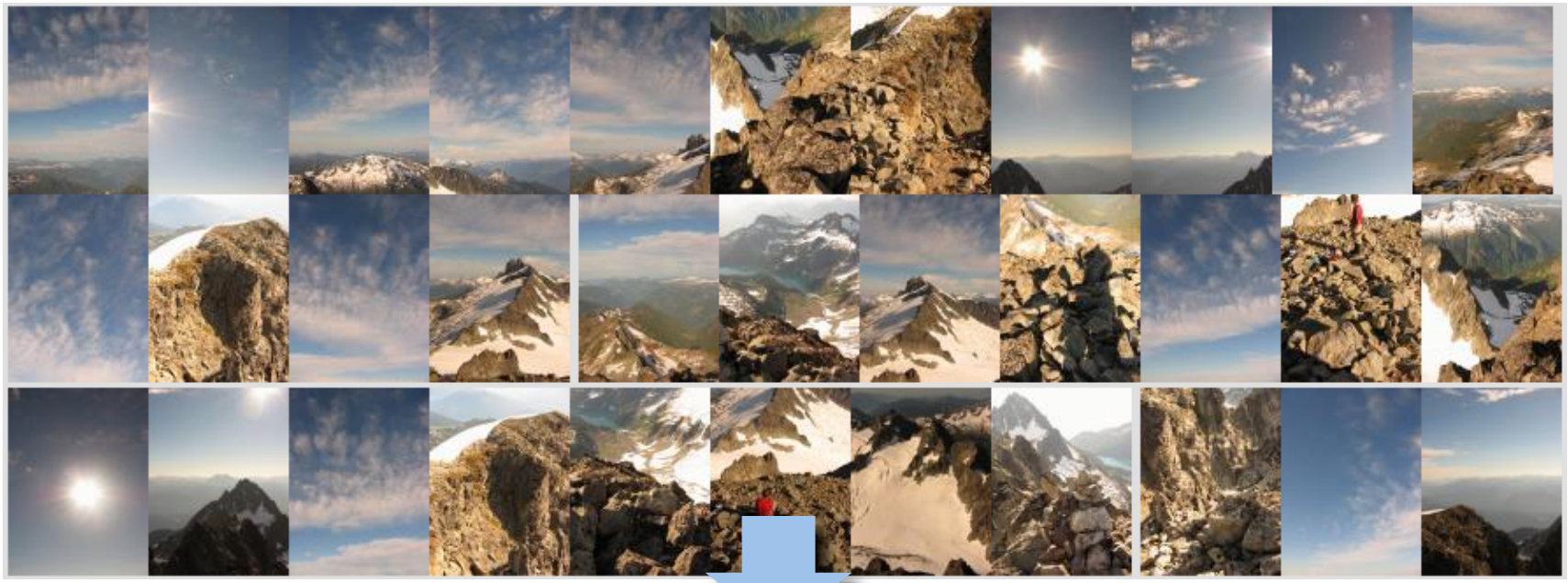
Feature detection and matching

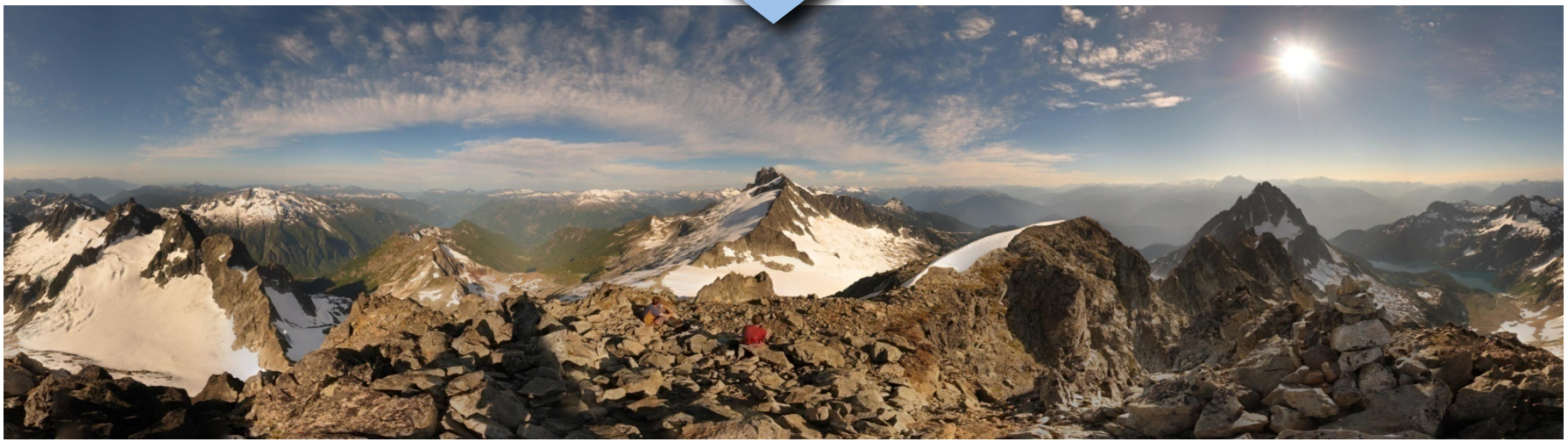


Reading

- Szeliski: 4.1

Motivation: Automatic panoramas





Credit: Matt Brown

Microsoft puts some pizzazz into panoramic photos

The company's ICE software now can stitch video frames into panoramic images and fill in inevitable gaps. It shows the field of computational photography is still in its early days.

by Stephen Shankland @stshank / February 5, 2015 9:30 AM PST

Microsoft ICE stitches still photos or video frames into a single panoramic image. Version 2.0 can fill in gaps so you don't have to crop as much.

Screenshot by Stephen Shankland/CNET

http://gigapixelartzoom.com