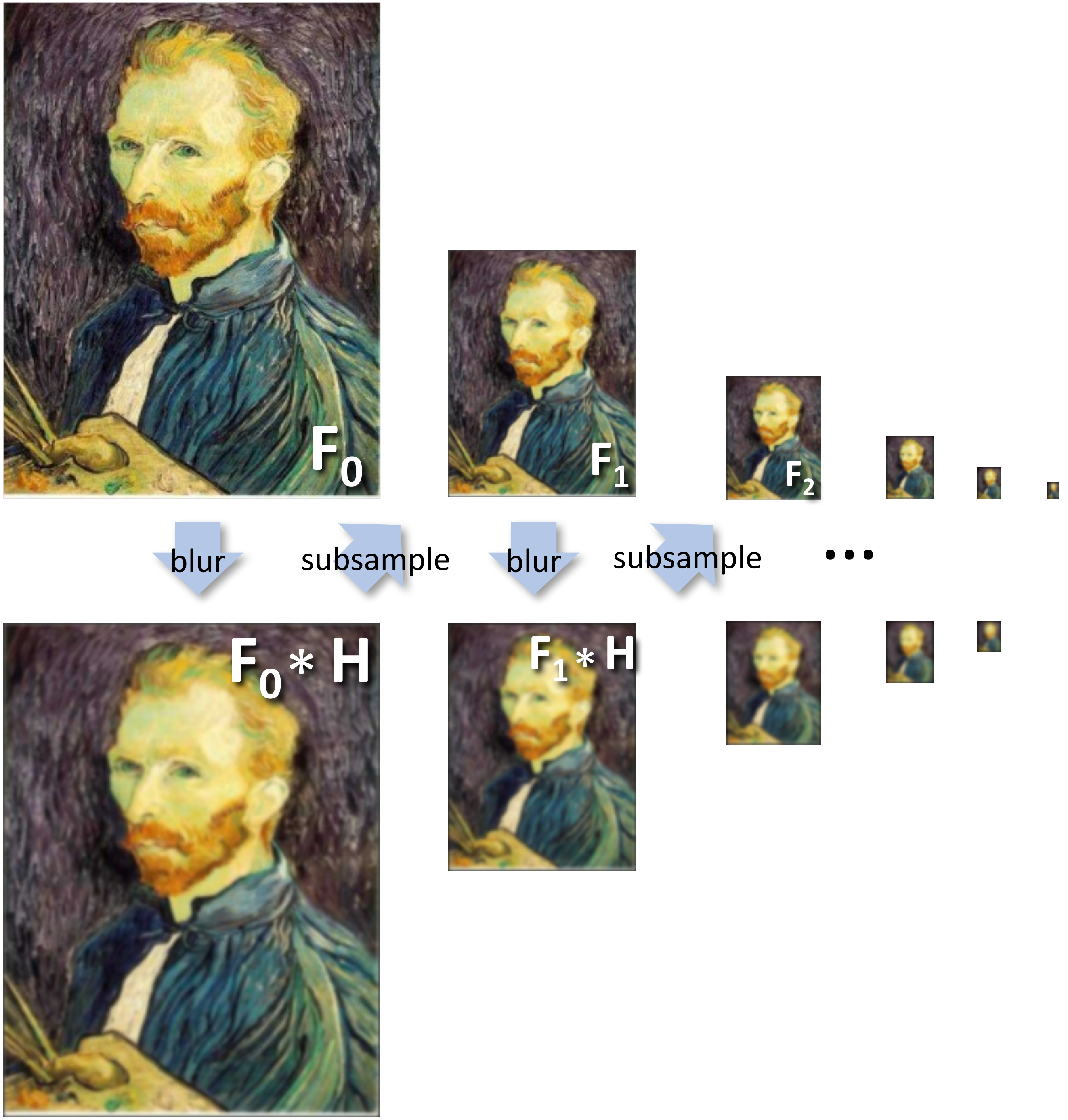
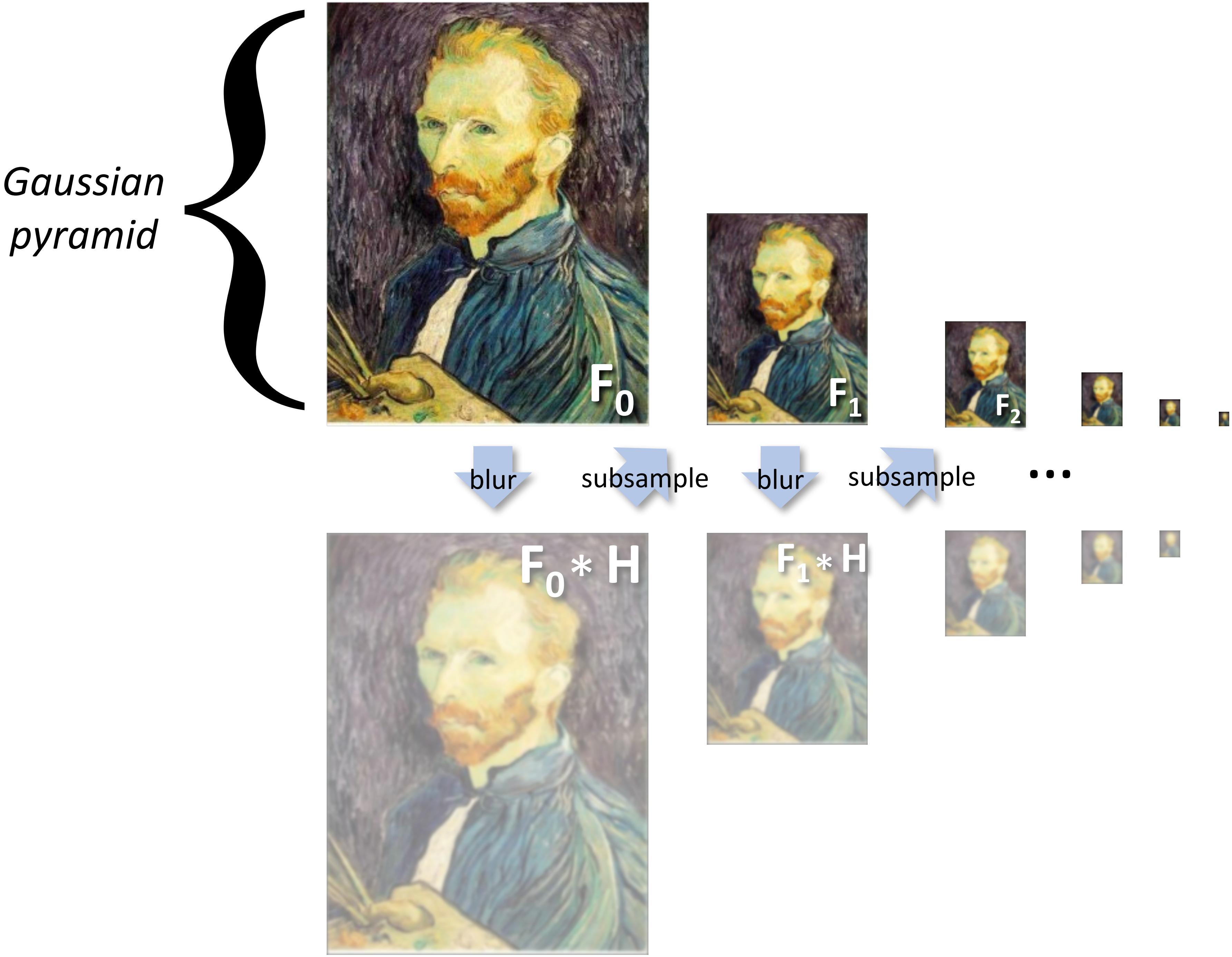
Pyramids

Gaussian pre-filtering

- Solution: filter the image, *then* subsample

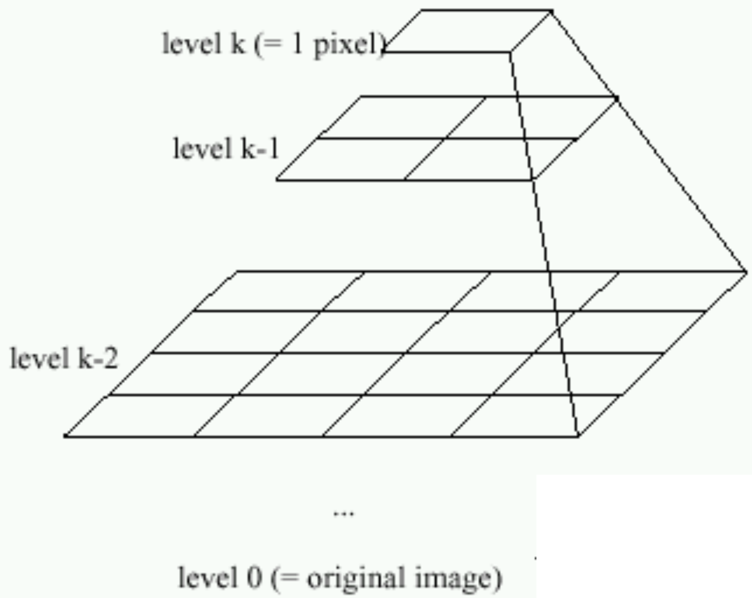




Gaussian pyramid
F0
F1
F2
blur
subsample
blur
subsample
...
F0 * H
F1 * H

Gaussian pyramids
[Burt and Adelson, 1983]

Idea: Represent NxN image as a "pyramid" of $1\text{x}1, 2\text{x}2, 4\text{x}4, \ldots, 2^k\text{x}2^k$ images (assuming $N=2^k$)

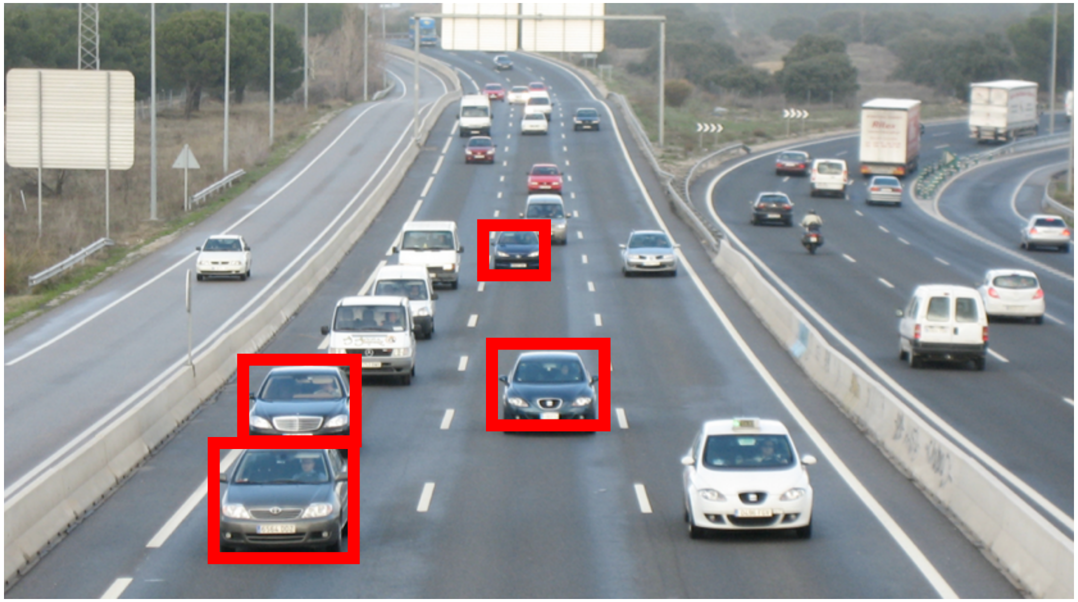


- In computer graphics, a *mip map* [Williams, 1983]

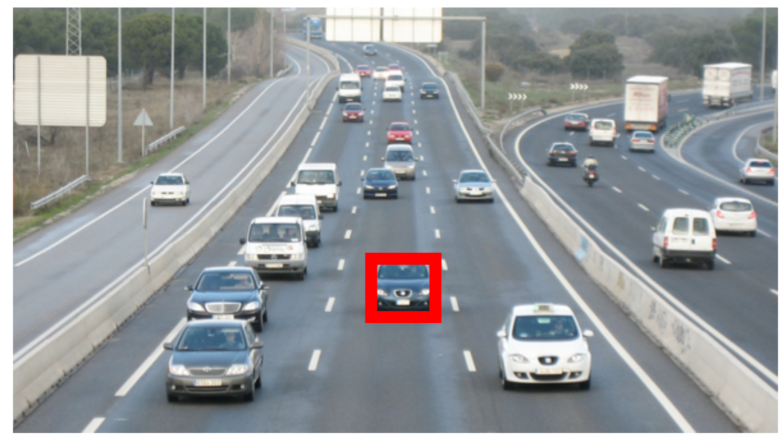
Gaussian Pyramids have all sorts of applications in computer vision

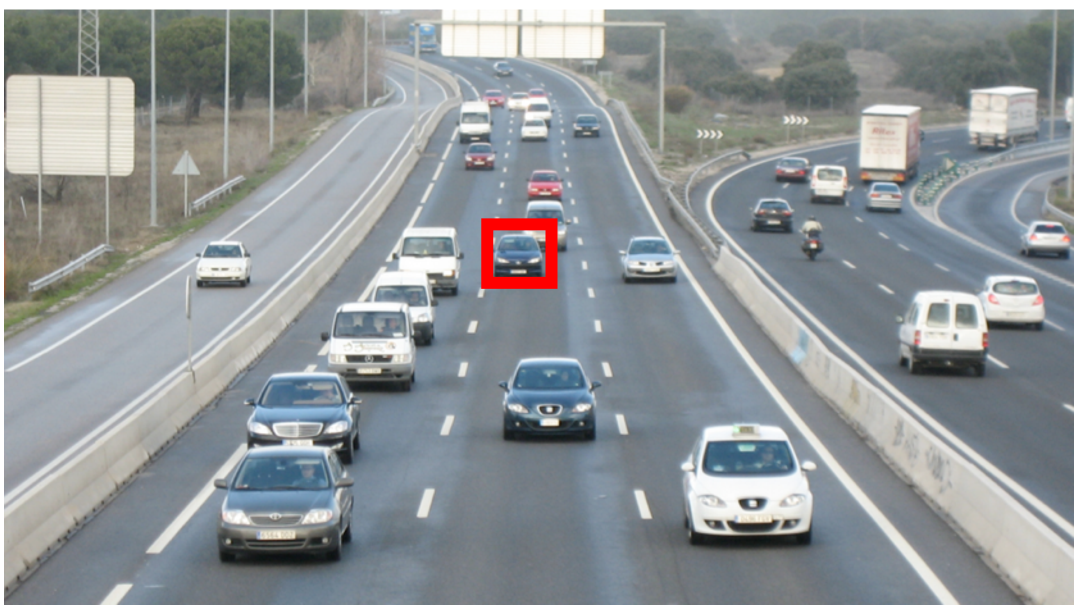
Source: S. Seitz

Gaussian pyramids - Searching over scales

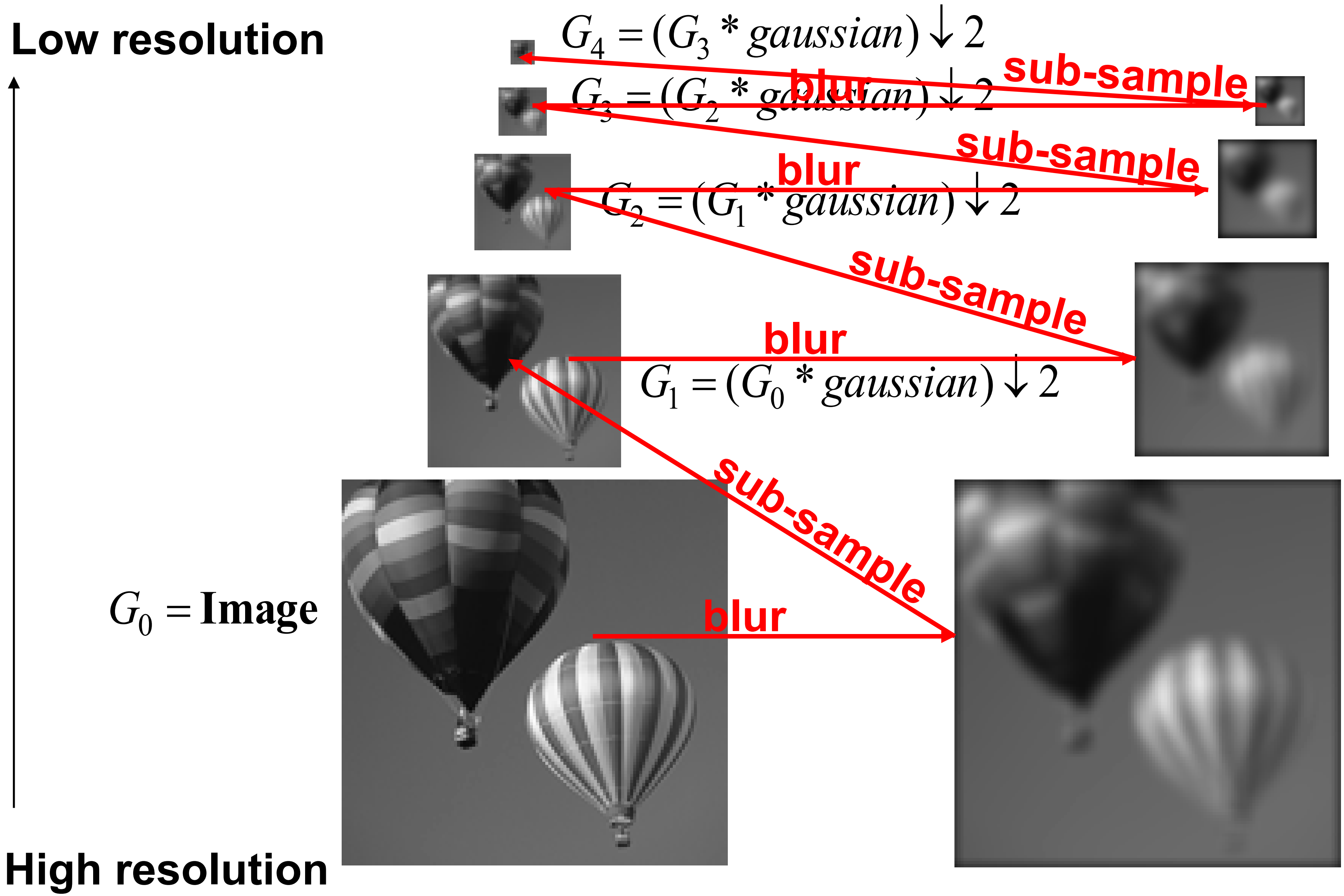


Gaussian pyramids - Searching over scales

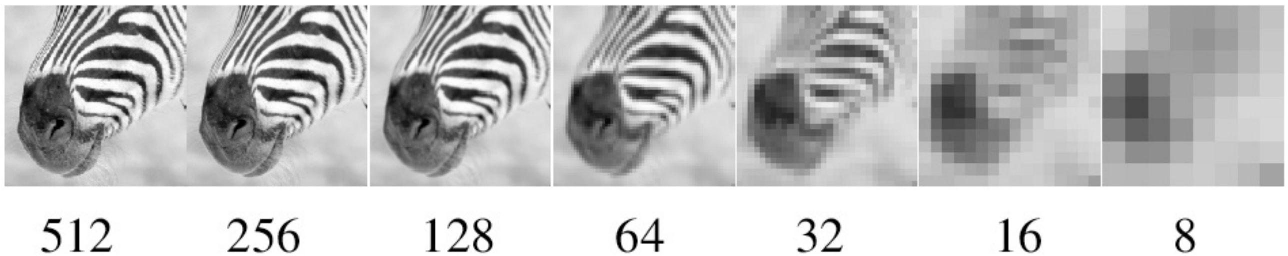




The Gaussian Pyramid



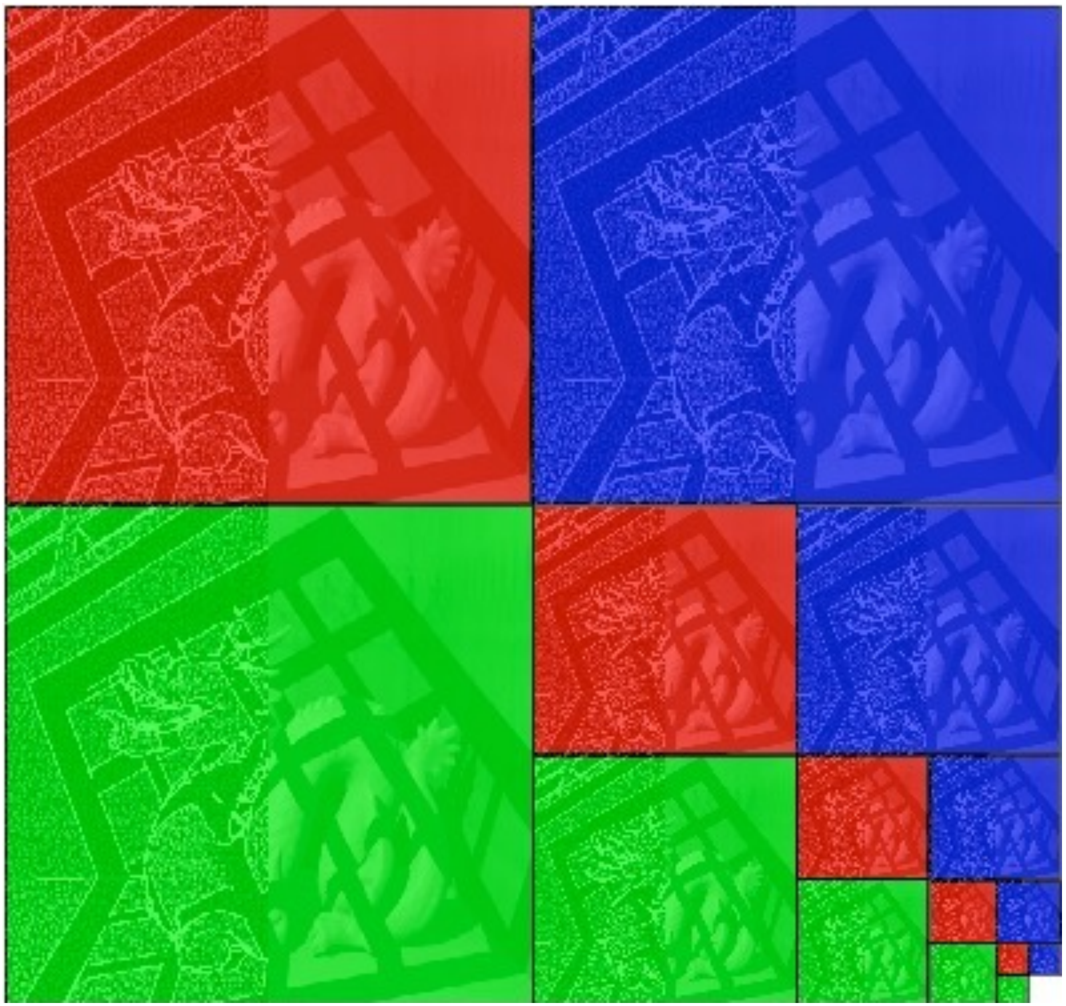
Gaussian pyramid and stack

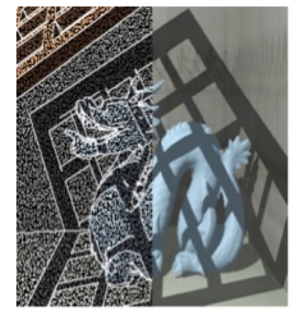


Source: Forsyth

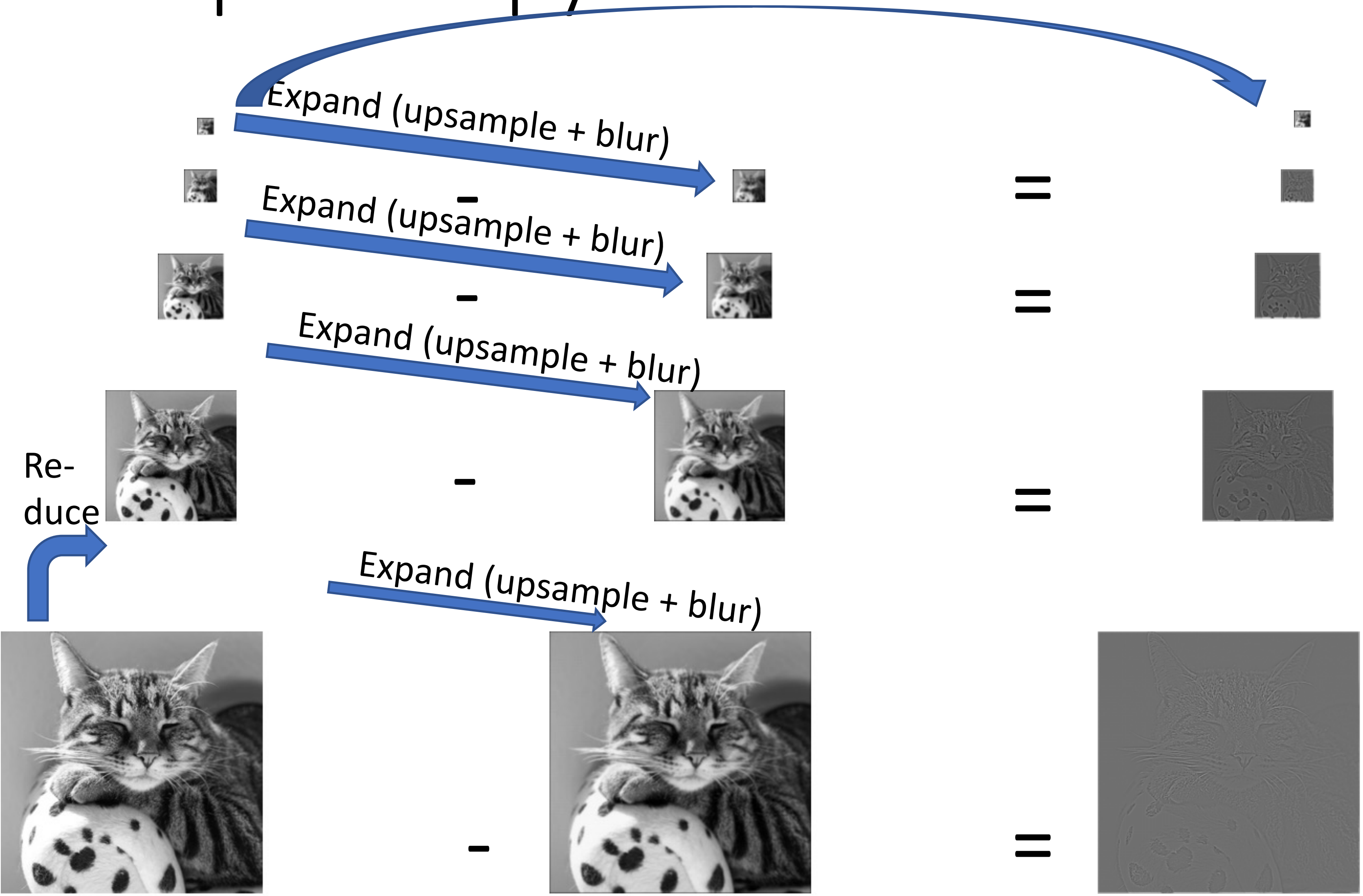
Memory Usage

- What is the size of the pyramid?





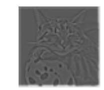
Laplacian pyramid

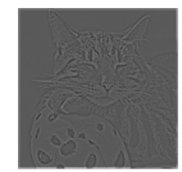


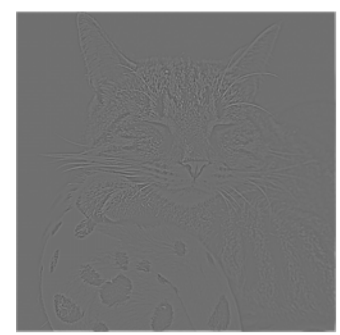
Laplacian pyramid

$L_4 = G_4 =$ 

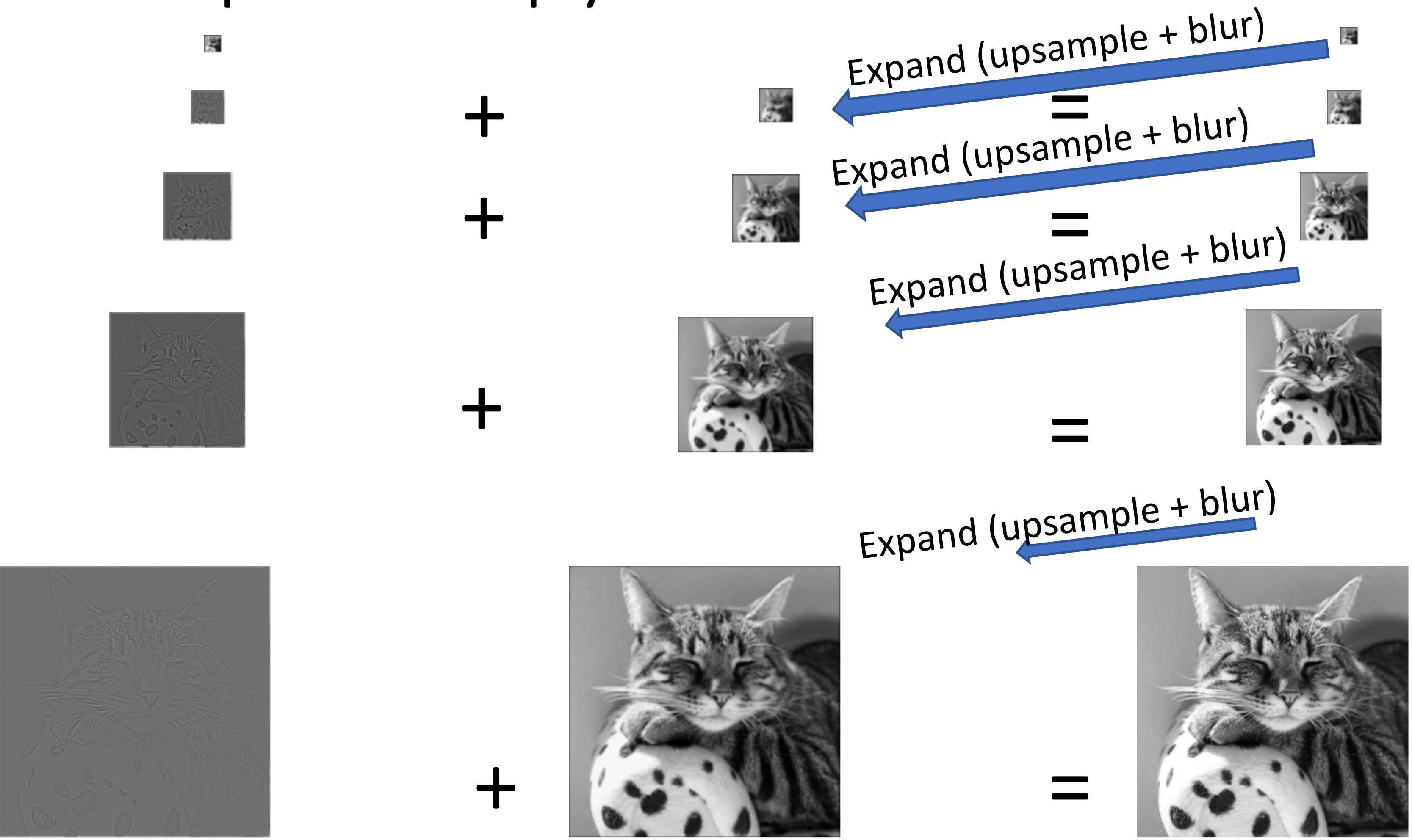
$L_3 = G_3 - \text{expand}(G_4) =$ 

$L_2 = G_2 - \text{expand}(G_3) =$ 

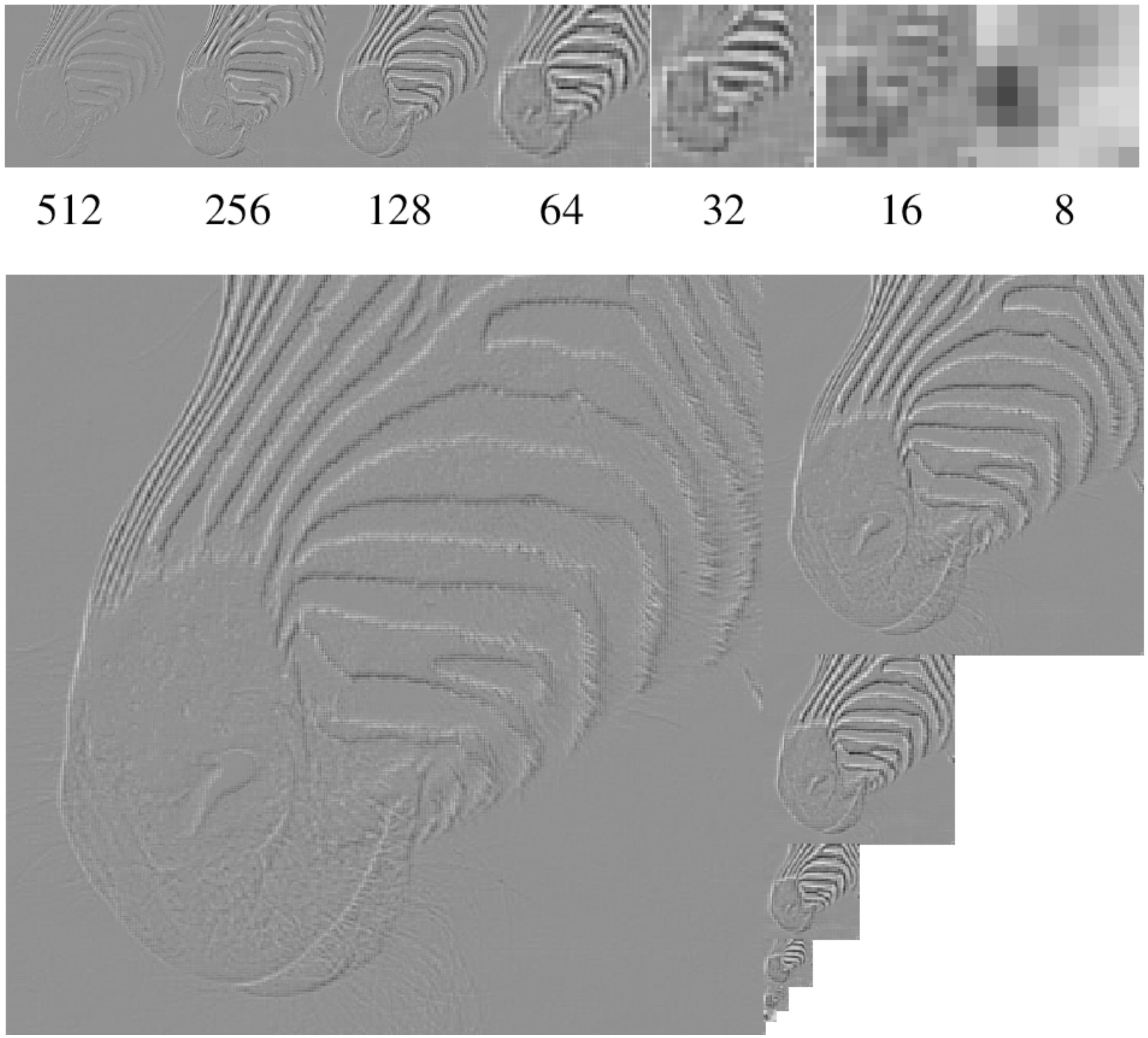
$L_1 = G_1 - \text{expand}(G_2) =$ 

$L_0 = G_0 - \text{expand}(G_1) =$ 

Reconstructing the image from a Laplacian pyramid



Laplacian pyramid



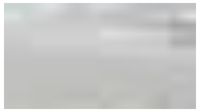
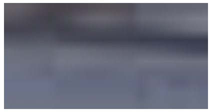
Source: Forsyth

Edge detection

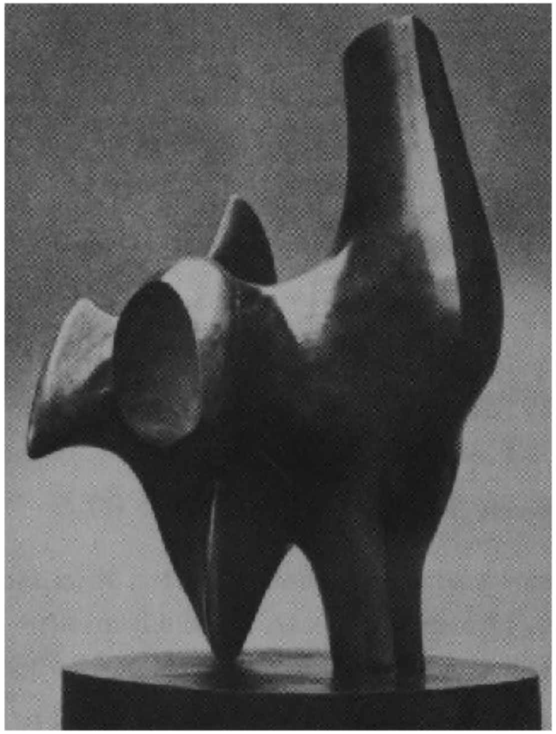
Why edges?

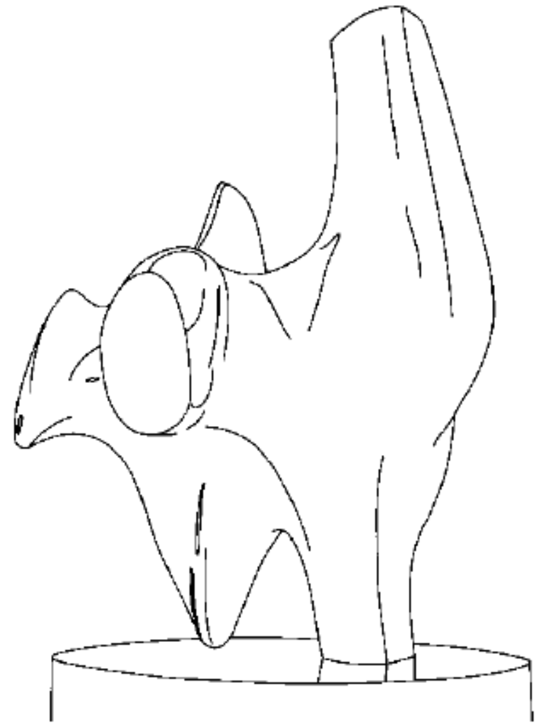
- Resilience to lighting and color
  - useful for recognition, matching patches across images




Why edges?



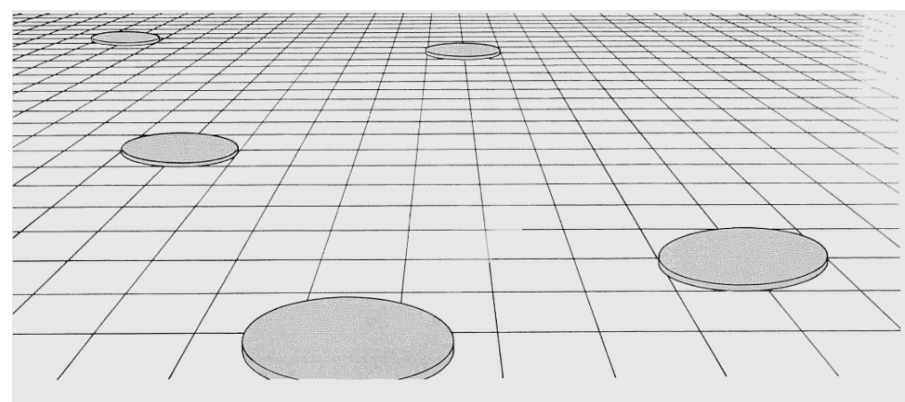


- Humans are sensitive to edges
- Convert a 2D image into a set of curves
  - Extracts salient features of the scene, more compact

Why edges?

- Cue to shape and geometry
  - useful for recognition, understanding 3D structure

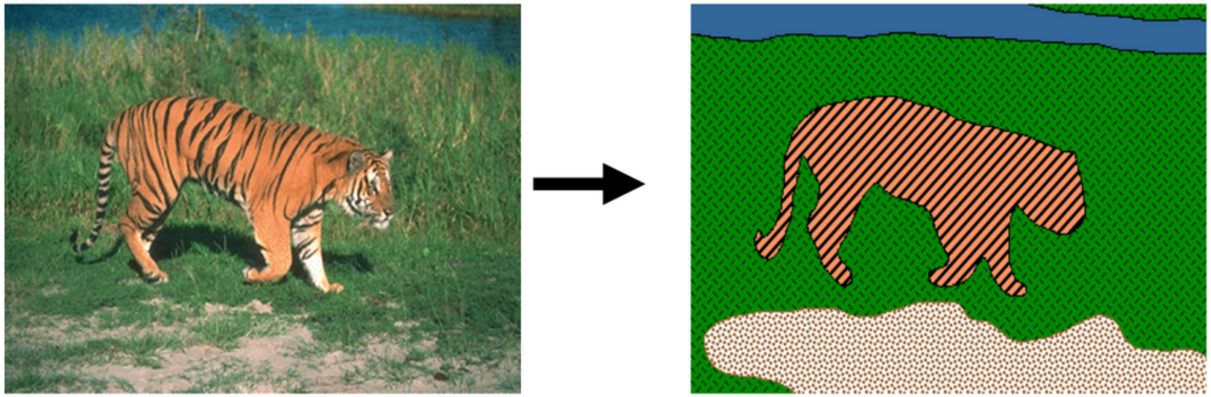
Credit: Attneave



Credit: Jitendra Malik

Why edges?

- Grouping pixels into objects (“perceptual organization”)

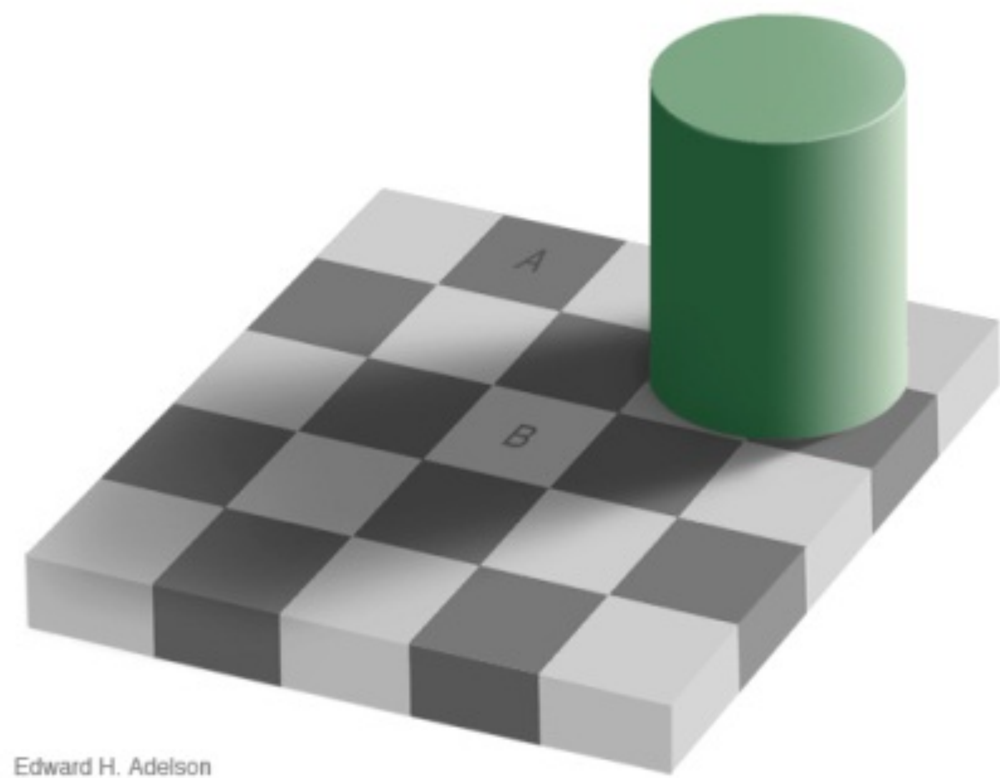


This lecture

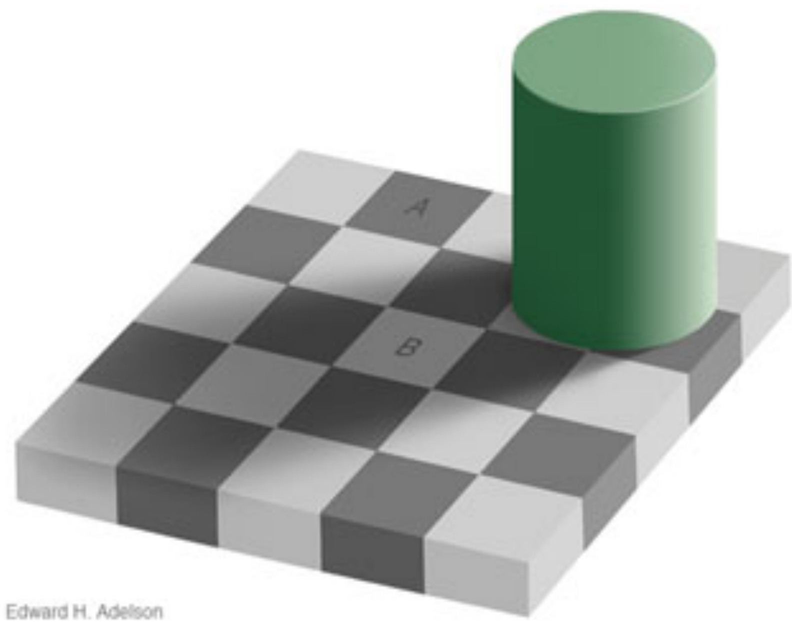
- Edge detection in general
- Edge detection for grouping

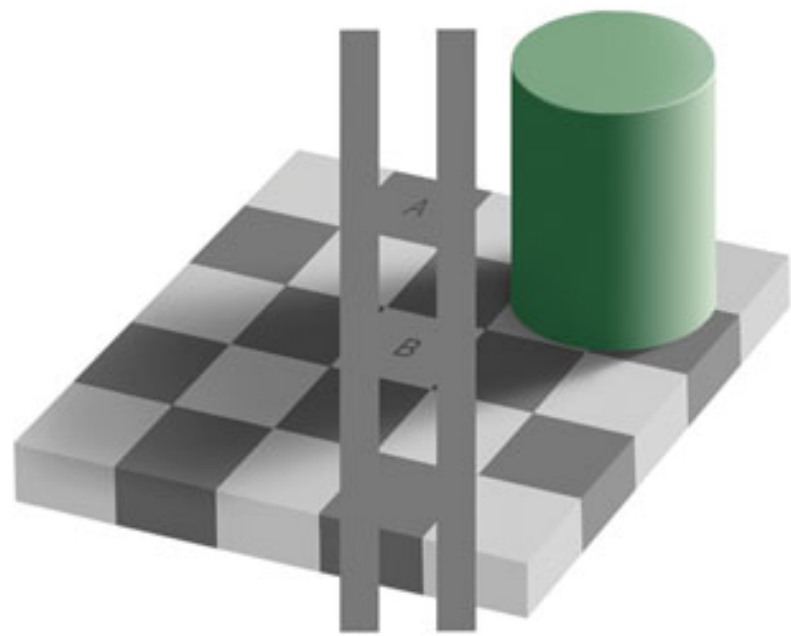
Edges

- Edges are curves in the image, across which the brightness changes “a lot”
- Corners/Junctions



Aside

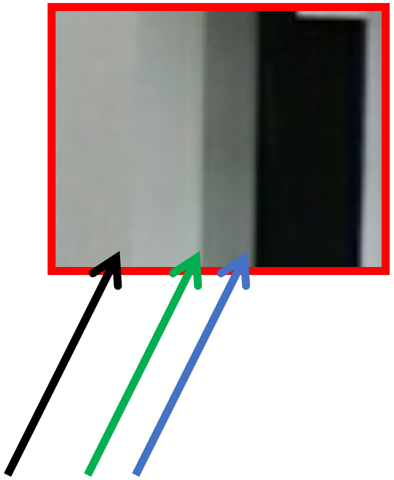




Closeup of edges

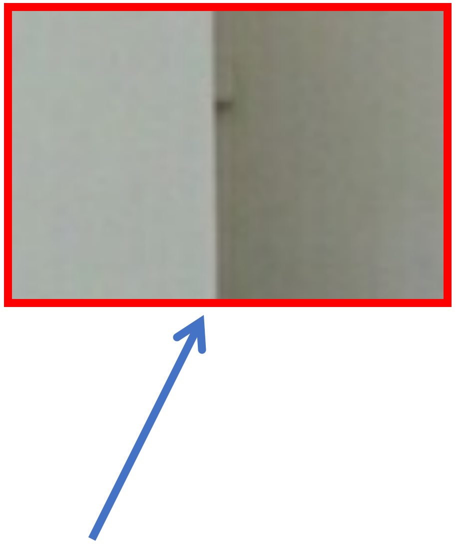
Source: D. Hoiem

Closeup of edges



Source: D. Hoiem

Closeup of edges



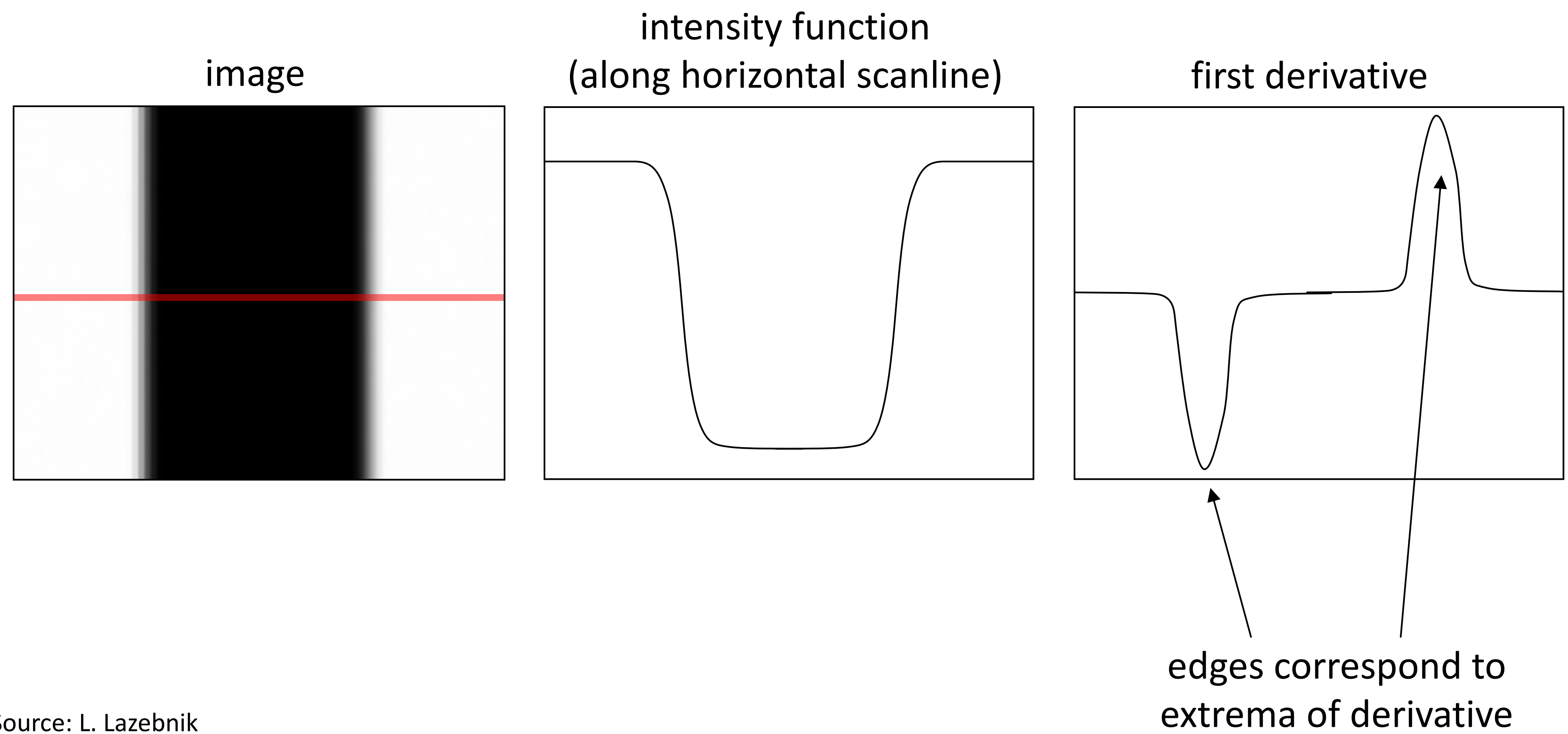
Source: D. Hoiem

Closeup of edges

Source: D. Hoiem

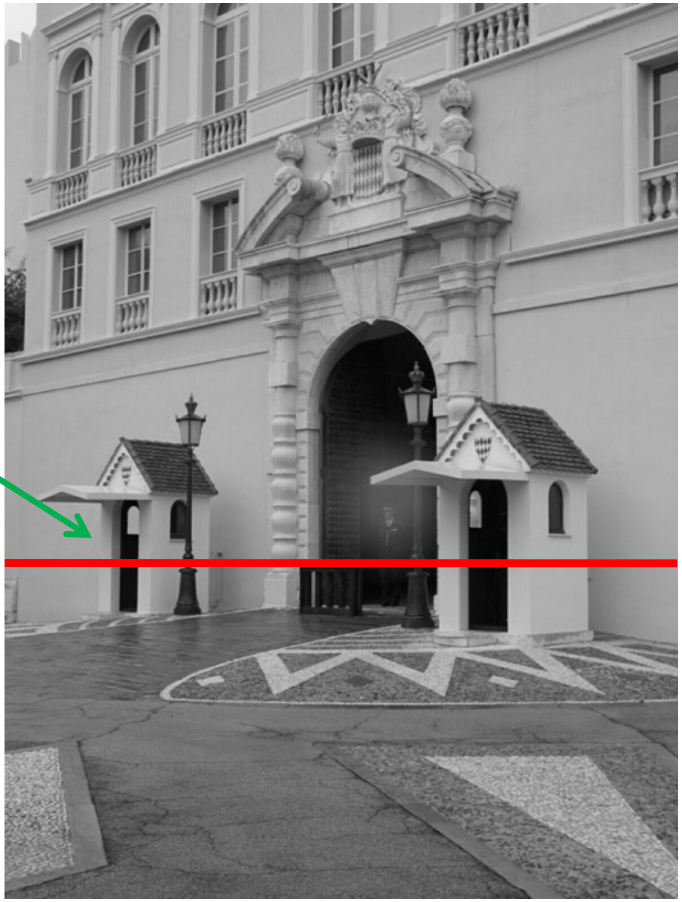
Characterizing edges

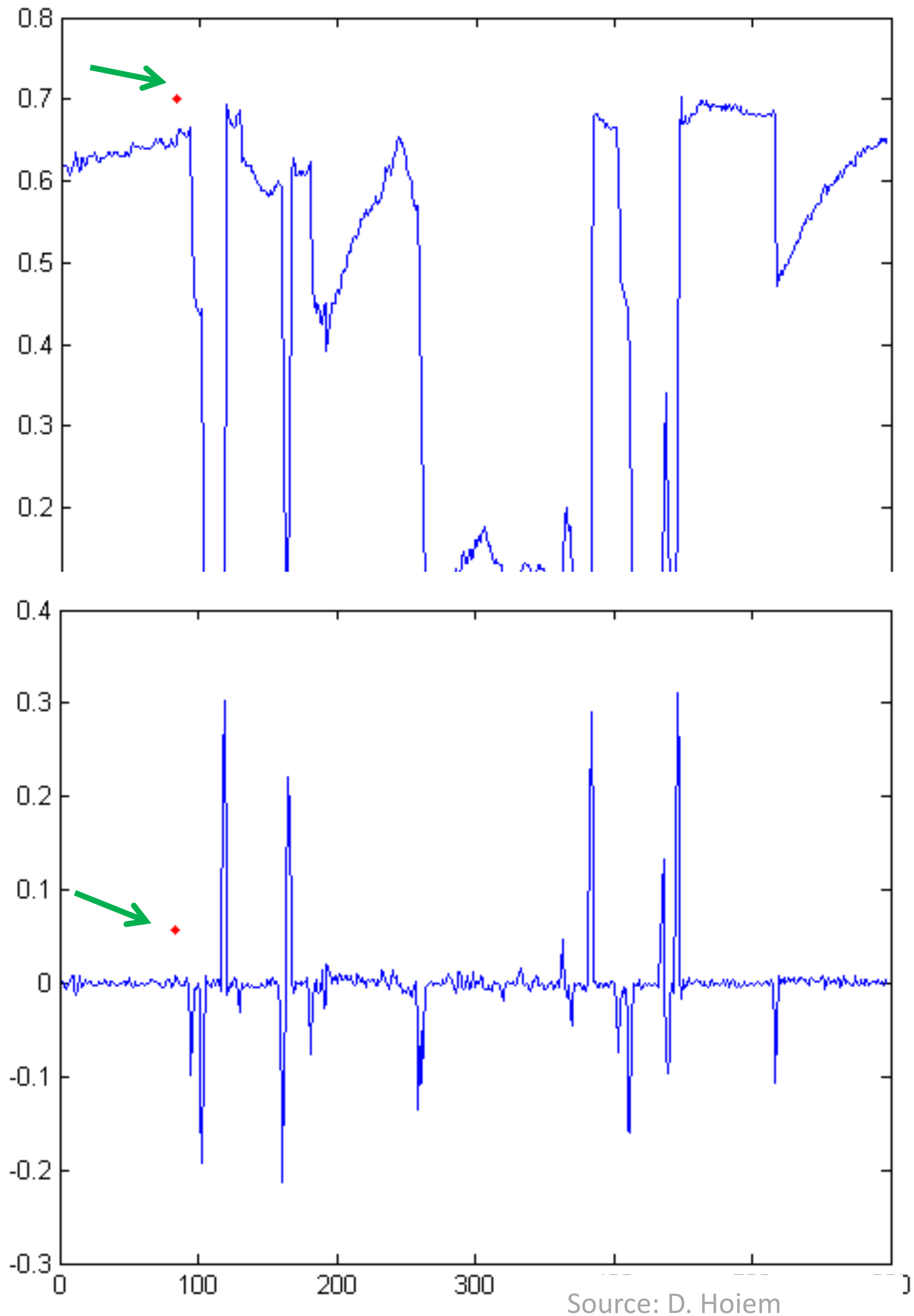
- An edge is a place of *rapid change* in the image intensity function



Source: L. Lazebnik

Intensity profile





Source: D. Hoiem

Derivatives and convolution

- Differentiation is *linear*

$$\frac{\partial(af(x) + bg(x))}{\partial x} = a\frac{\partial f(x)}{\partial x} + b\frac{\partial g(x)}{\partial x}$$

- Differentiation is *shift-invariant*
  - Derivative of shifted signal is shifted derivative

- Hence, differentiation can be represented as convolution!

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

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

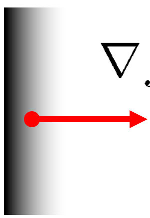
H_y

Source: S. Seitz

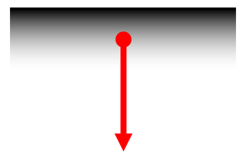
Image gradient

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

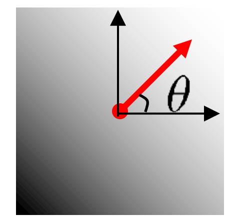
  The gradient points in the direction of most rapid increase in intensity



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



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



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

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

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

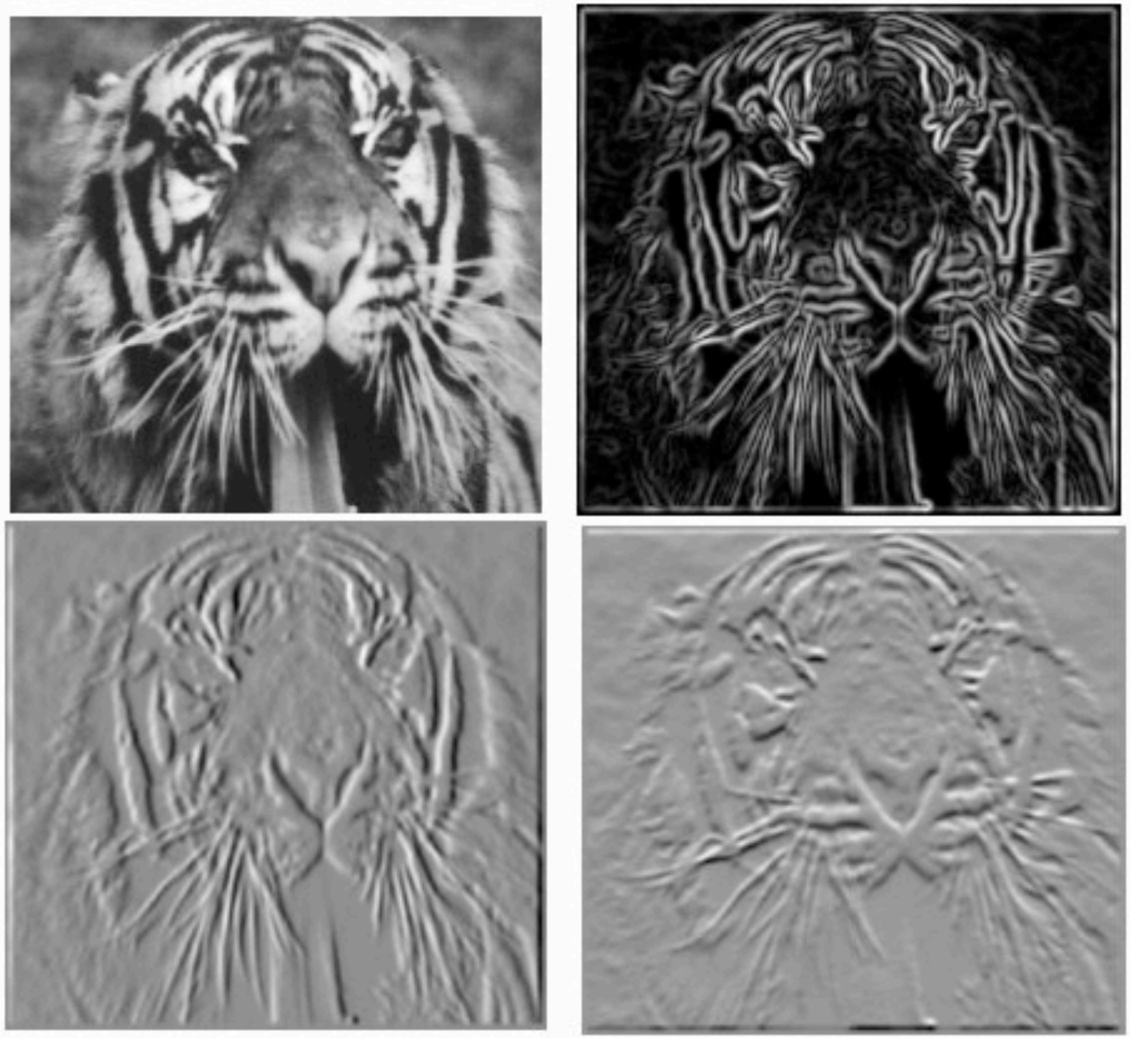
The gradient direction is given by:

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

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

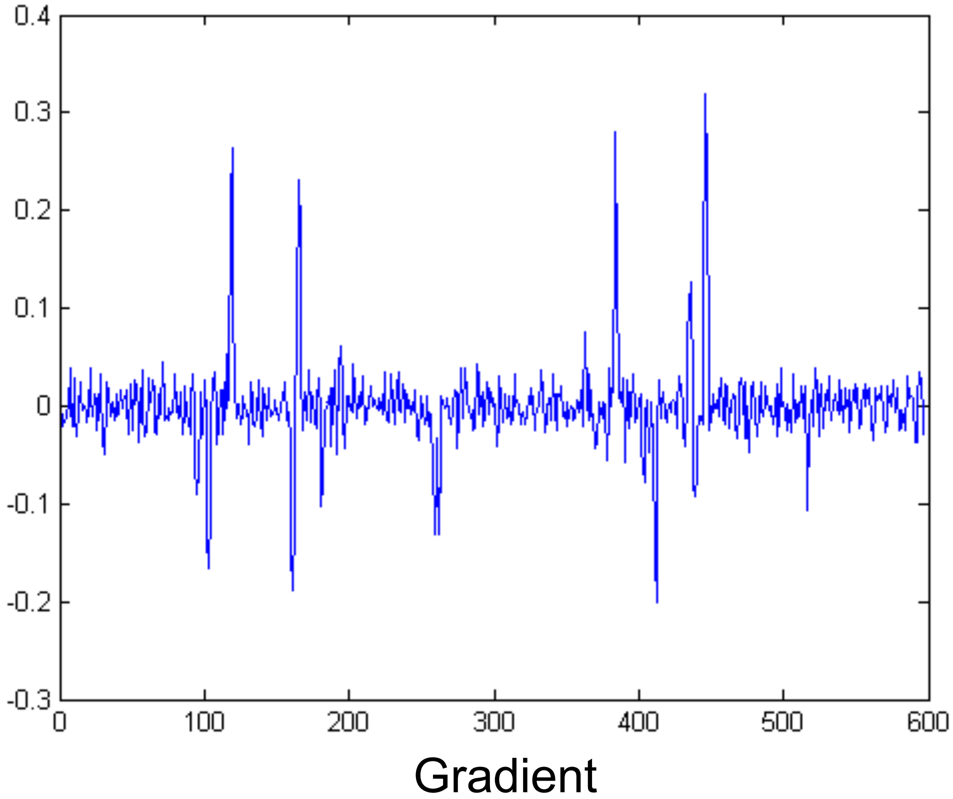
Source: Steve Seitz

Image gradient



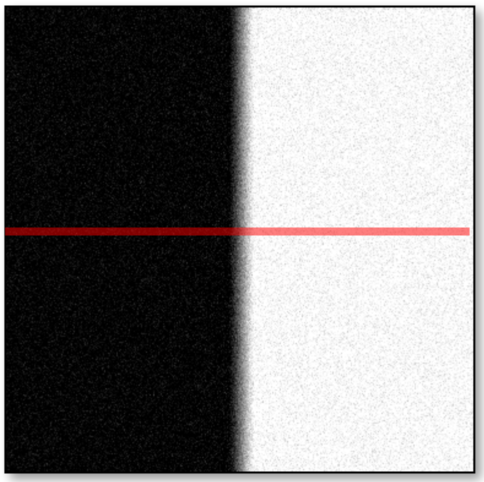
Source: L. Lazebnik

With a little Gaussian noise

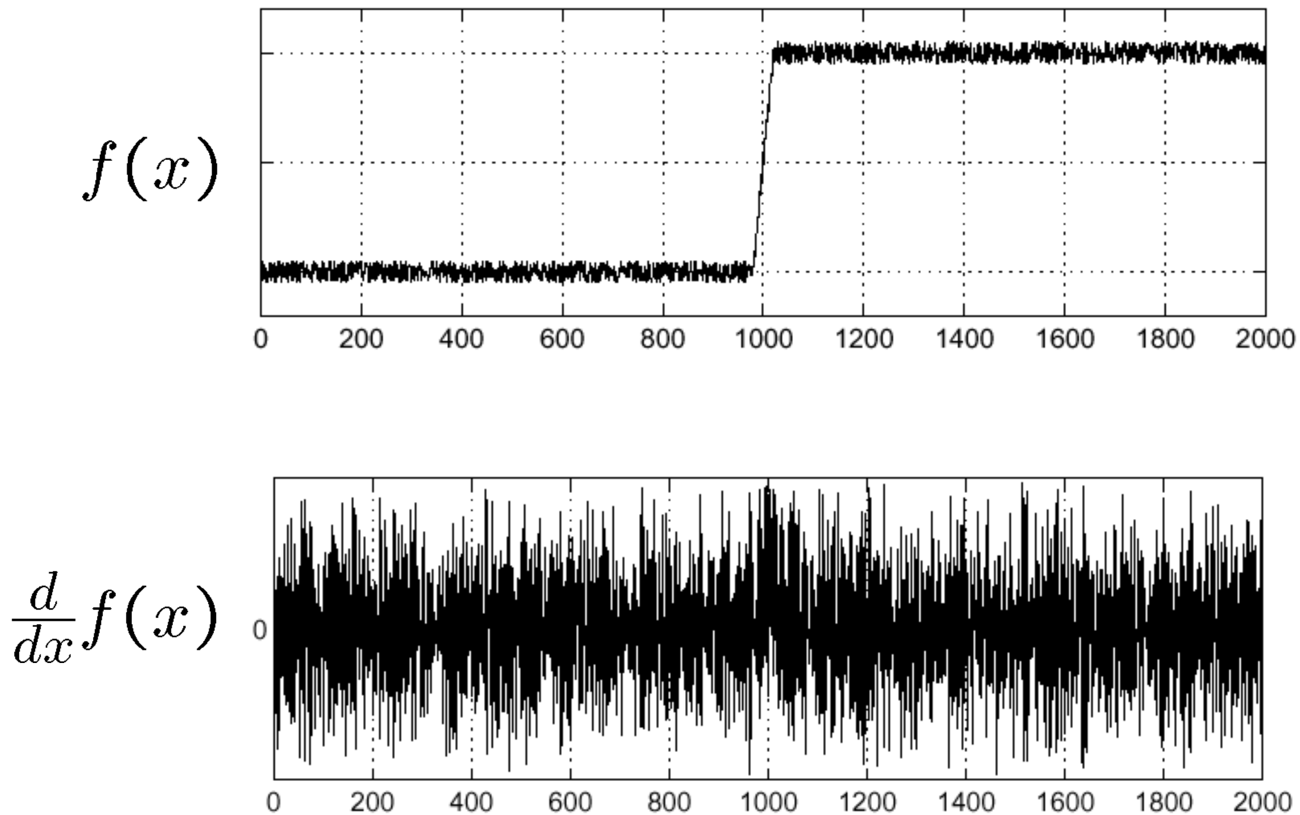


Source: D. Hoiem

Effects of noise



Noisy input image



Where is the edge?

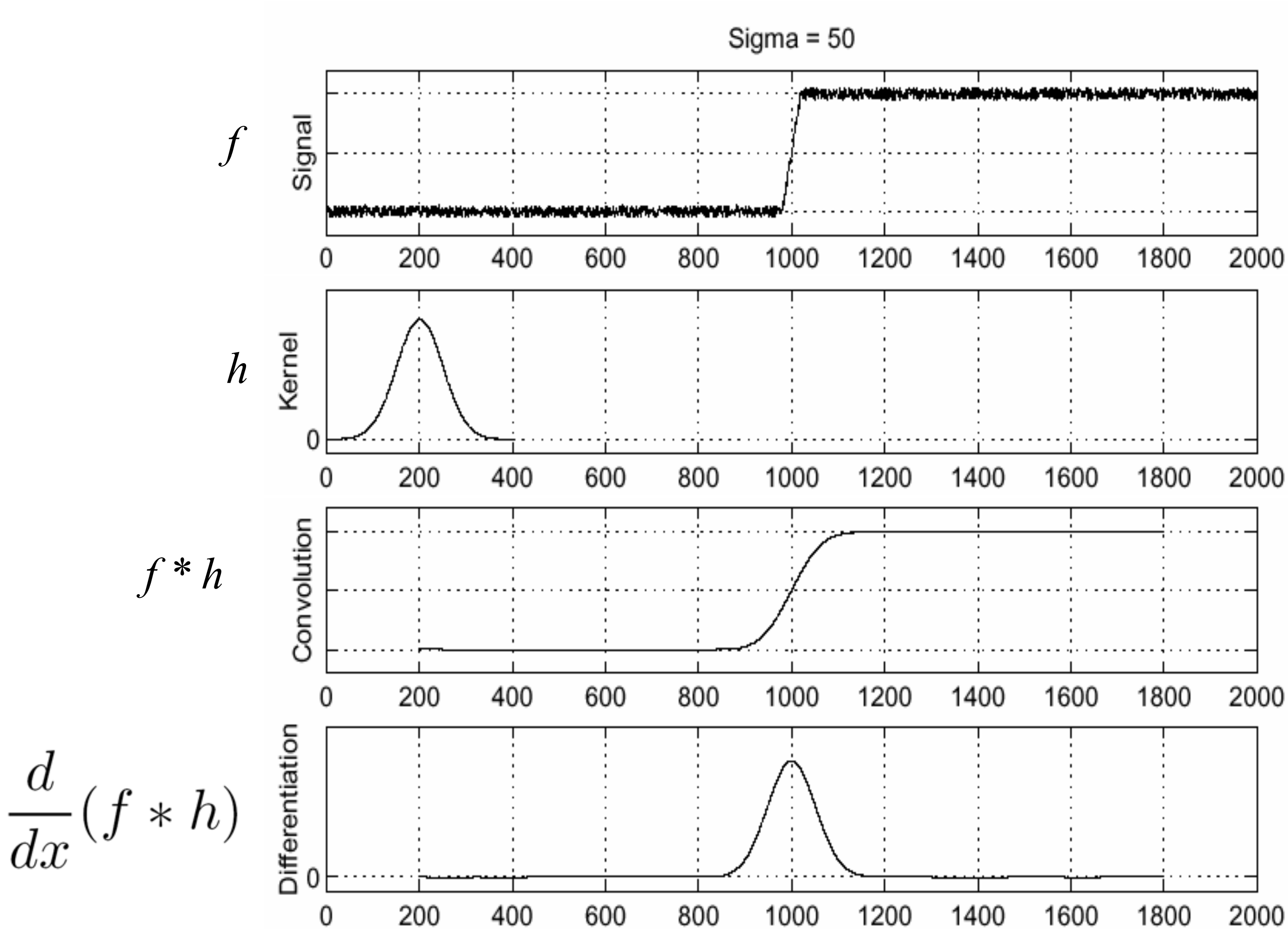
Source: S. Seitz

Effects of noise

- Noise is high frequency
- Differentiation accentuates noise

$$\frac{d \sin \omega x}{dx} = \omega \cos \omega x$$

Solution: smooth first



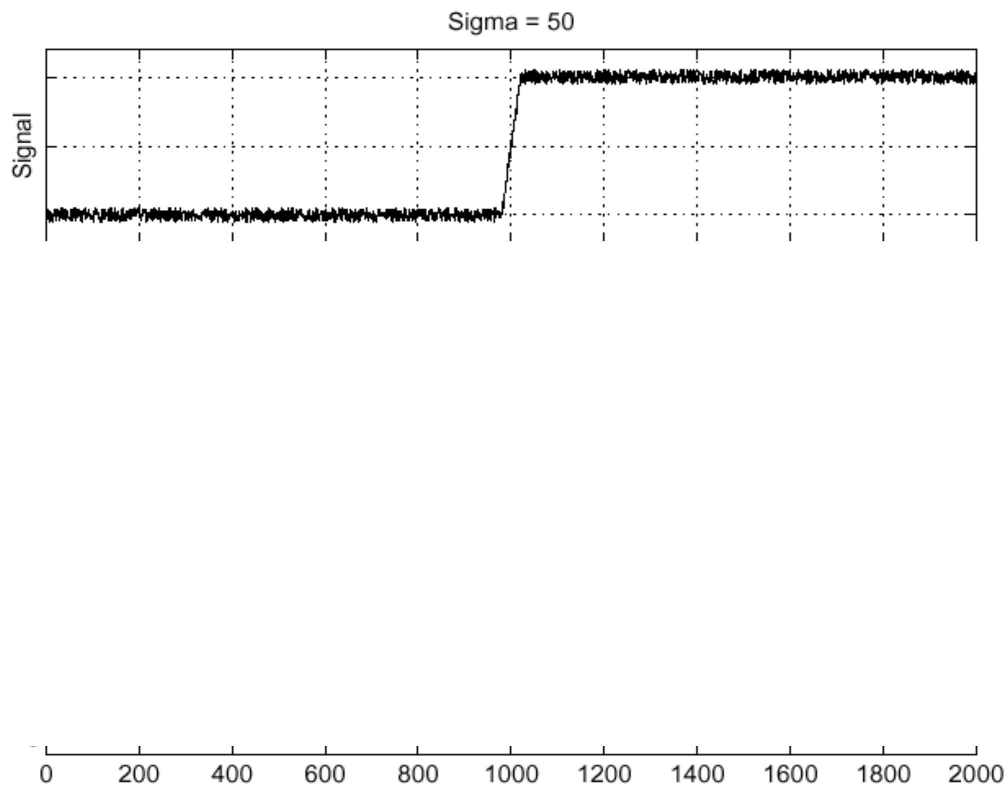
To find edges, look for peaks in $\frac{d}{dx}(f * h)$

Source: S. Seitz

Associative property of convolution

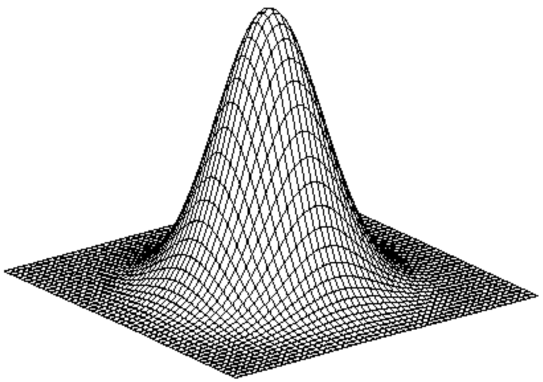
- Differentiation is a convolution
- Convolution is associative: $\frac{d}{dx}(f * h) = f * \frac{d}{dx} h$
- This saves us one operation:

f



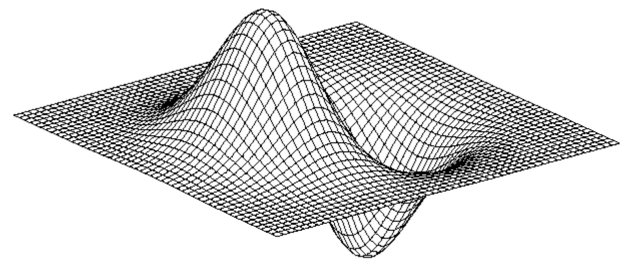
Source: S. Seitz

2D edge detection filters



Gaussian

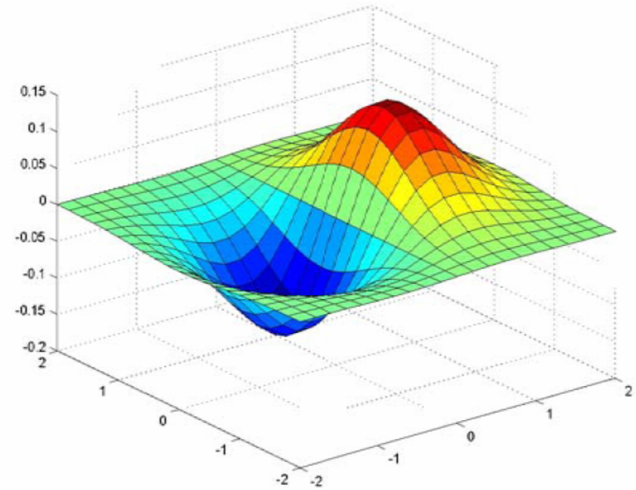
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



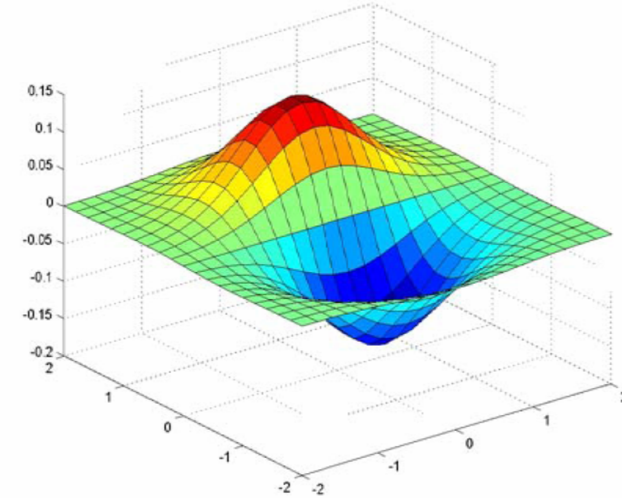
derivative of Gaussian (x)

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

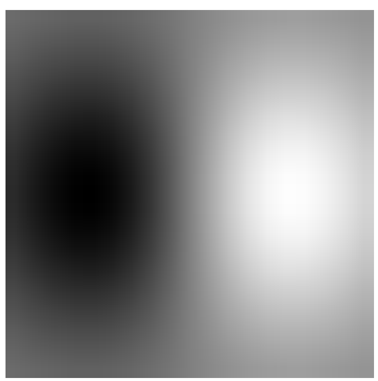
Derivative of Gaussian filter

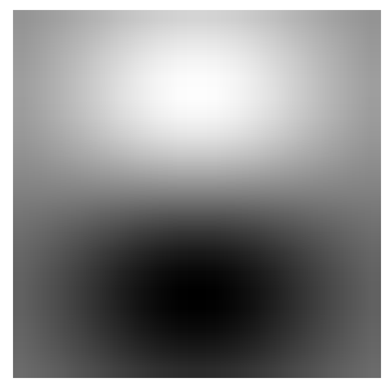


x-direction



y-direction





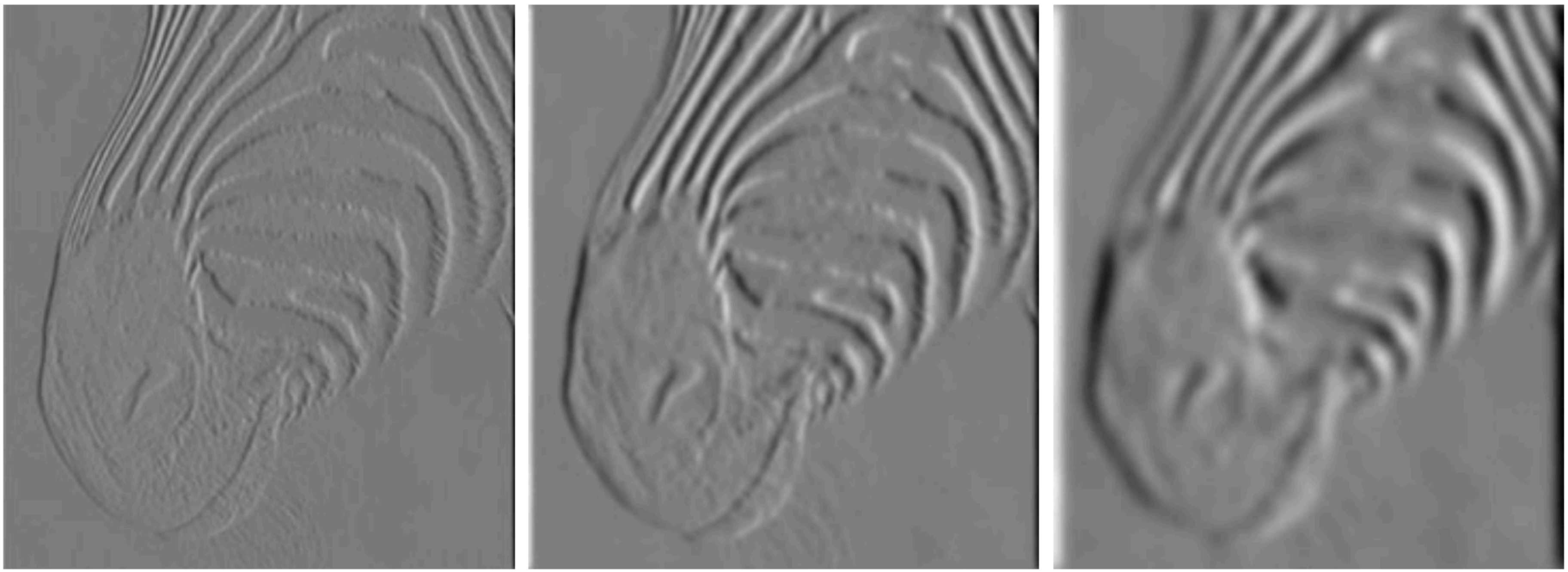


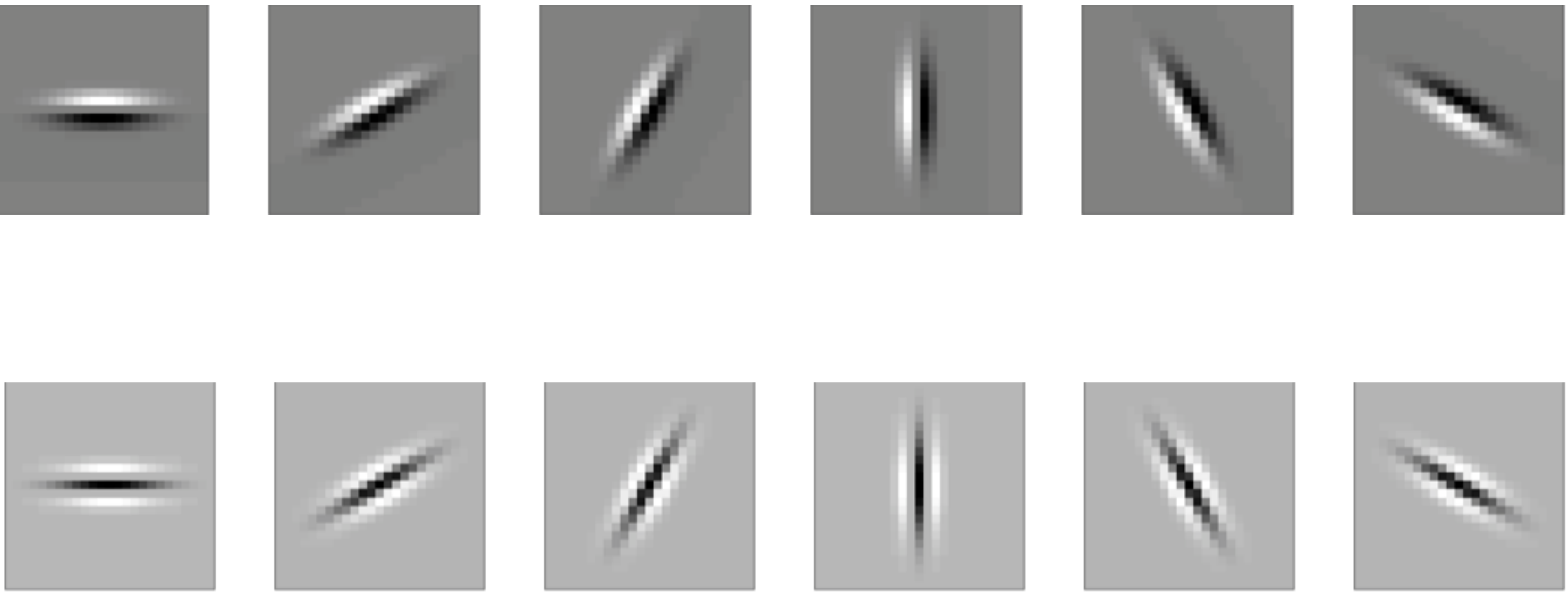
FIGURE 5.3: The scale (i.e., σ) of the Gaussian used in a derivative of Gaussian filter has significant effects on the results. The three images show estimates of the derivative in the x direction of an image of the head of a zebra obtained using a derivative of Gaussian filter with σ one pixel, three pixels, and seven pixels (**left** to **right**). Note how images at a finer scale show some hair, the animal's whiskers disappear at a medium scale, and the fine stripes at the top of the muzzle disappear at the coarser scale.

Two Dimensional Gaussian

$$\text{Anisotropic: } G_{\sigma_x,\sigma_y}(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)}$$

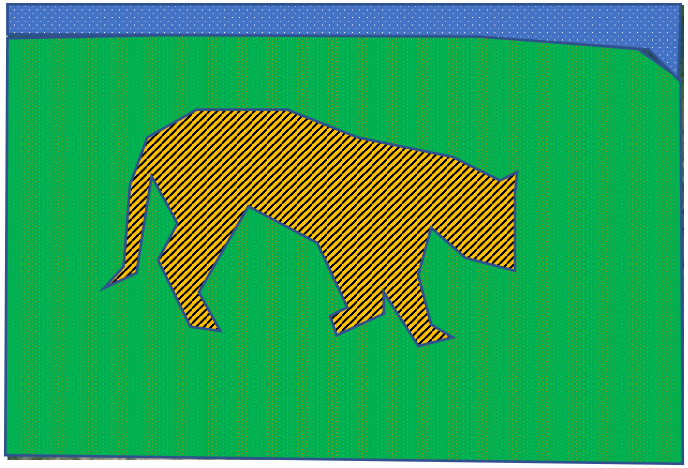
$$\text{Isotropic: } G_{\sigma}(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{r^2}{2\sigma^2}}$$

Oriented Gaussian First and Second Derivatives



Grouping

What is grouping?

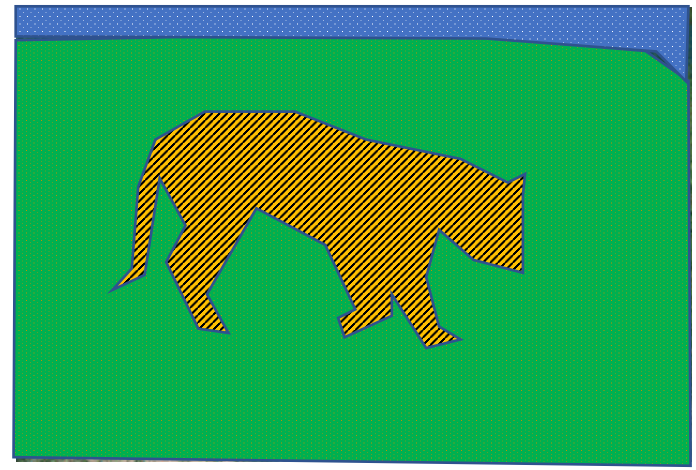


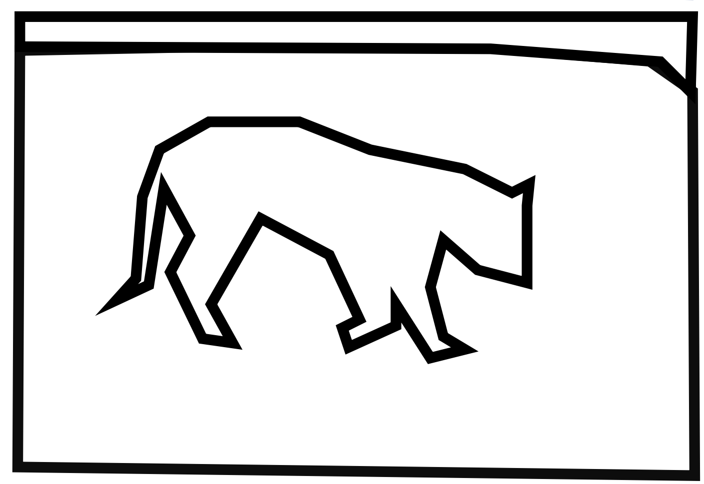
Why grouping?

- Pixels property of sensor, not world
- Reasoning at object level (might) make things easy:
  - objects at consistent depth
  - objects can be recognized
  - objects move as one

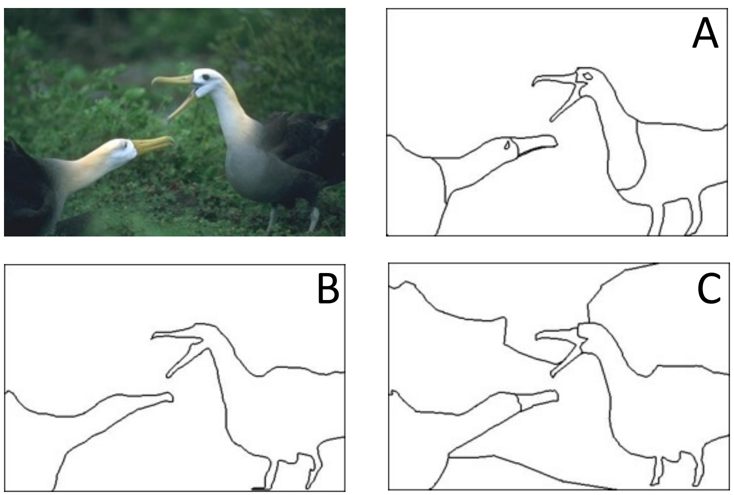
"I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have "327"? No. I have sky, house, and trees."
Max Wertheimer

Regions ⟷ Boundaries



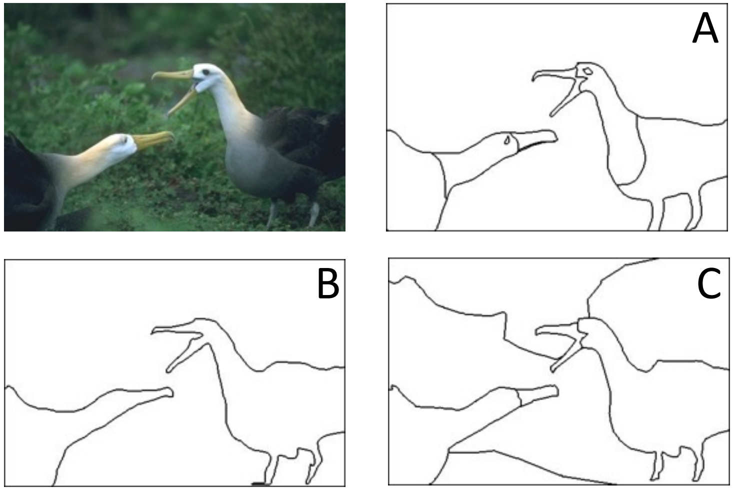


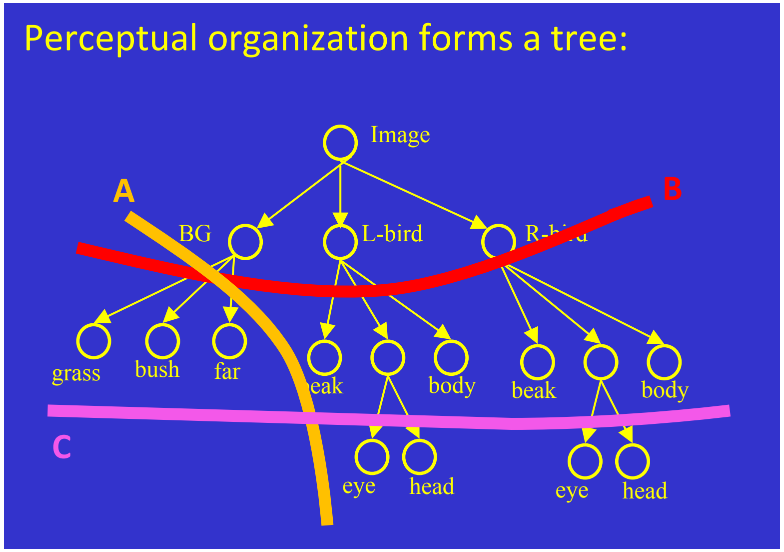
Is grouping well-defined?



- Depends on purpose
  - Object parts
  - Background segmentation

Is grouping well-defined?

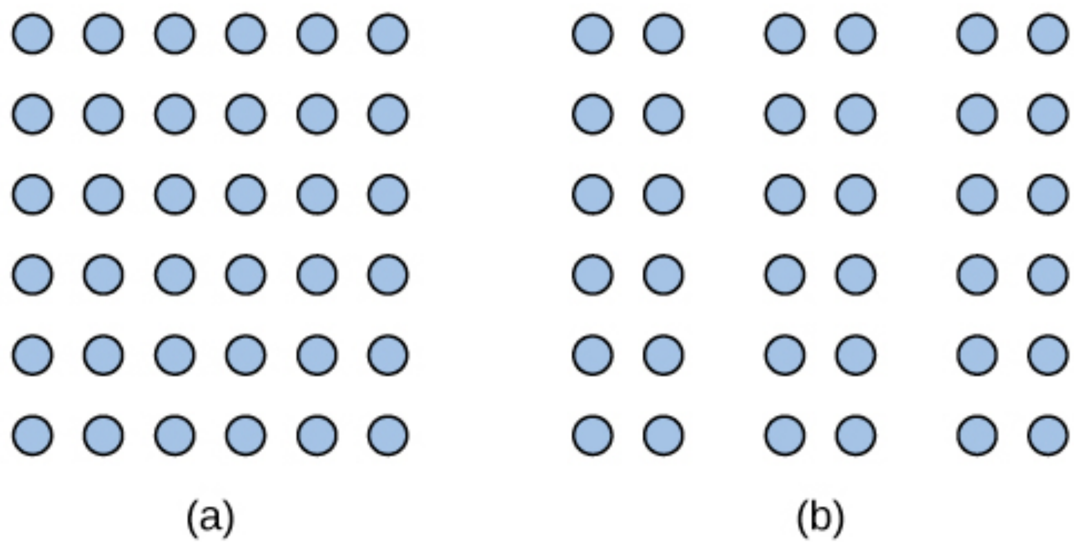




D. Martin, C. Fowlkes, D. Tal, J. Malik. "A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics", ICCV, 2001

How do we group things?

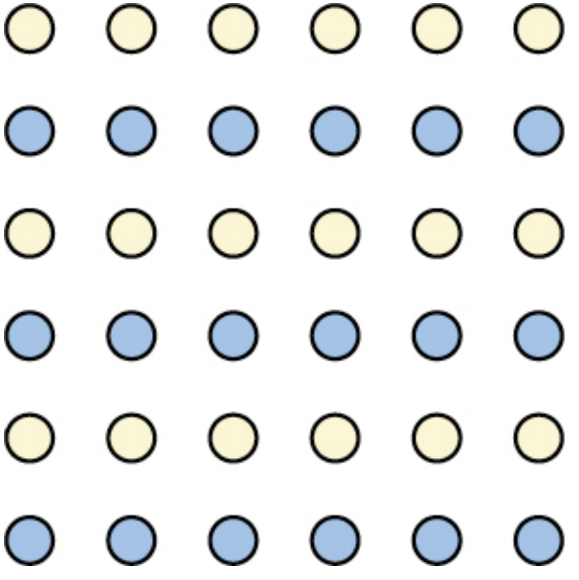
- *Gestalt* principles
- Principle of *proximity*



https://courses.lumenlearning.com/wsu-sandbox/chapter/gestalt-principles-of-perception/

How do we group things?

- Gestalt principles
- Principle of *similarity*



https://courses.lumenlearning.com/wsu-sandbox/chapter/gestalt-principles-of-perception/

How do we group things?

- Gestalt principles
- Principle of *continuity* and *closure*

https://courses.lumenlearning.com/wsu-sandbox/chapter/gestalt-principles-of-perception/

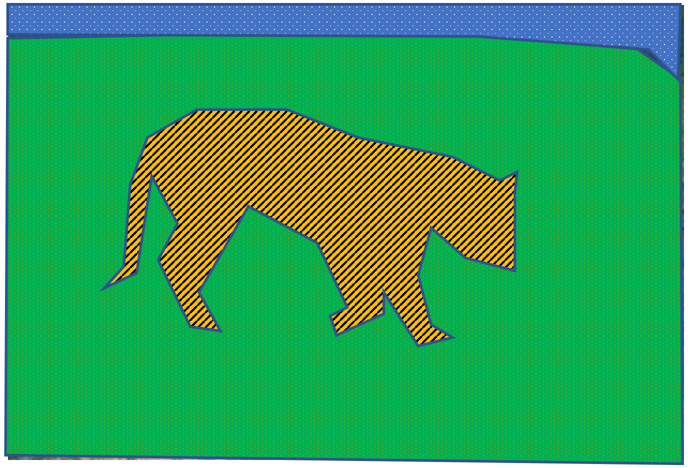
How do we group things?

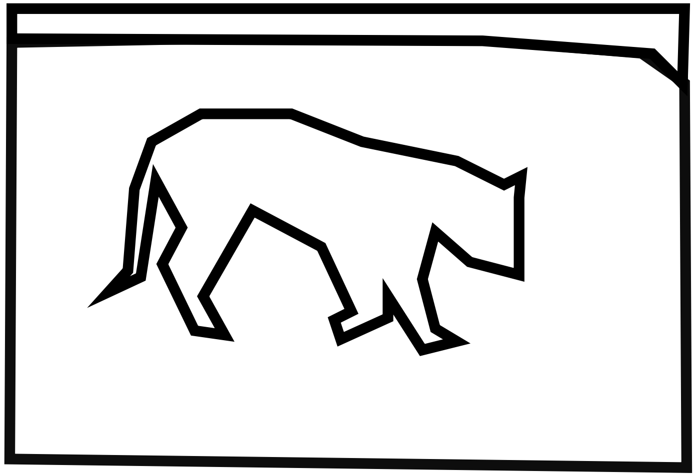
- Gestalt principles
- Principle of *common fate*

Gestalt principles in the context of images

- Principle of proximity: nearby pixels are part of the same object
- Principle of similarity: similar pixels are part of the same object
    - Look for differences in color, intensity, or texture across the boundary
- Principle of closure and continuity: contours are likely to continue
- High-level knowledge?

Regions ⟷ Boundaries





Designing a good boundary detector

- Differences in color, intensity, or texture across the boundary
- Continuity and closure
- High-level knowledge

Criteria for a good boundary detector

- Criteria for a good boundary detector:
  - **Good detection:** Fire only on real edges, not anywhere else
  - **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

Source: L. Fei-Fei

Canny edge detector

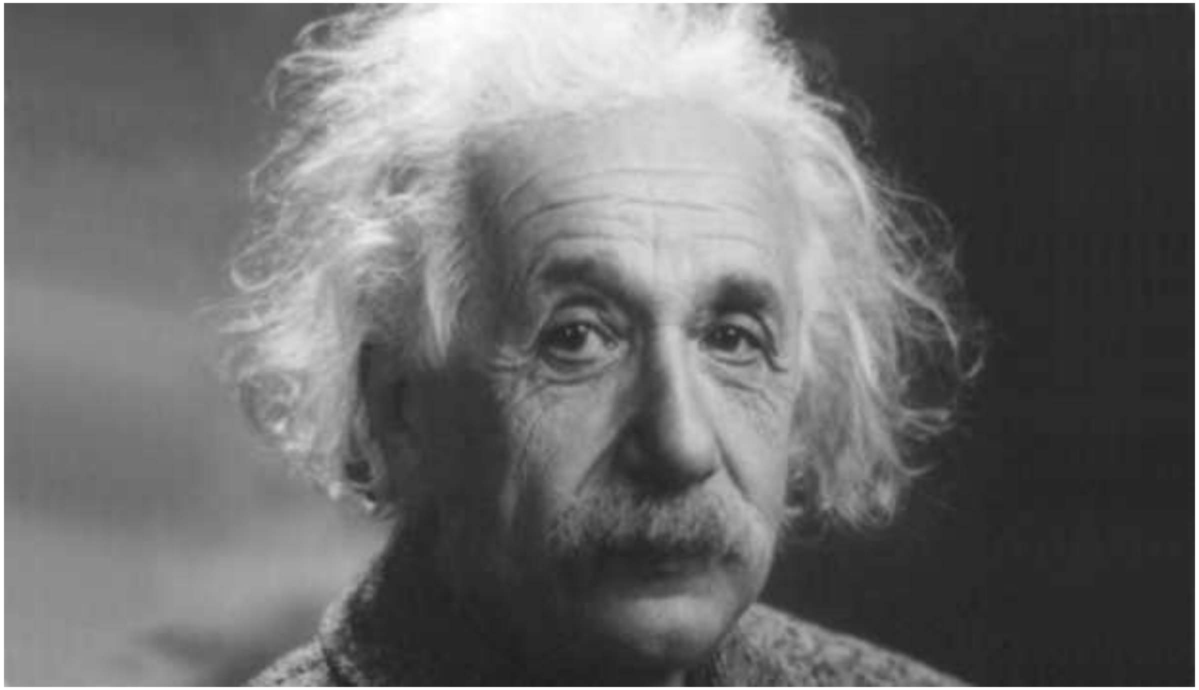
- The classic edge detector
- Baseline for all later work on grouping
- Theoretical model: step-edges corrupted by additive Gaussian noise

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

22,000 citations!

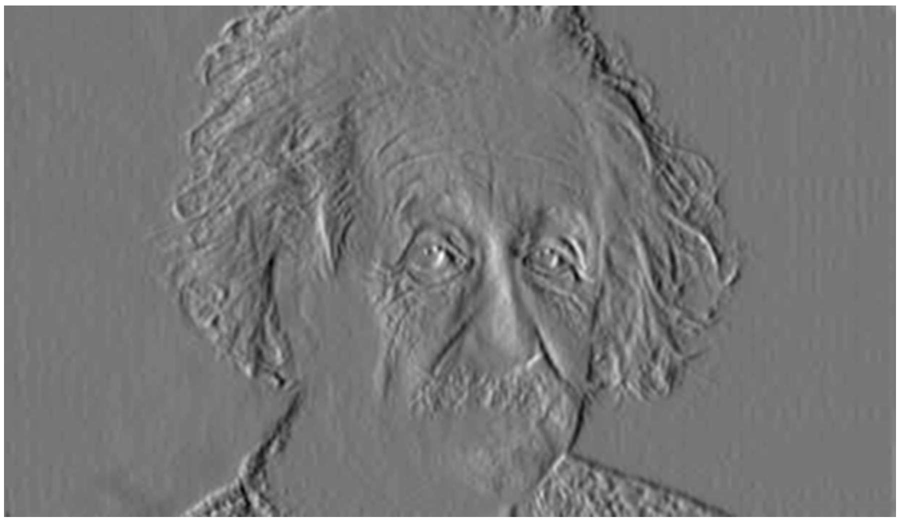
Source: L. Fei-Fei

Example

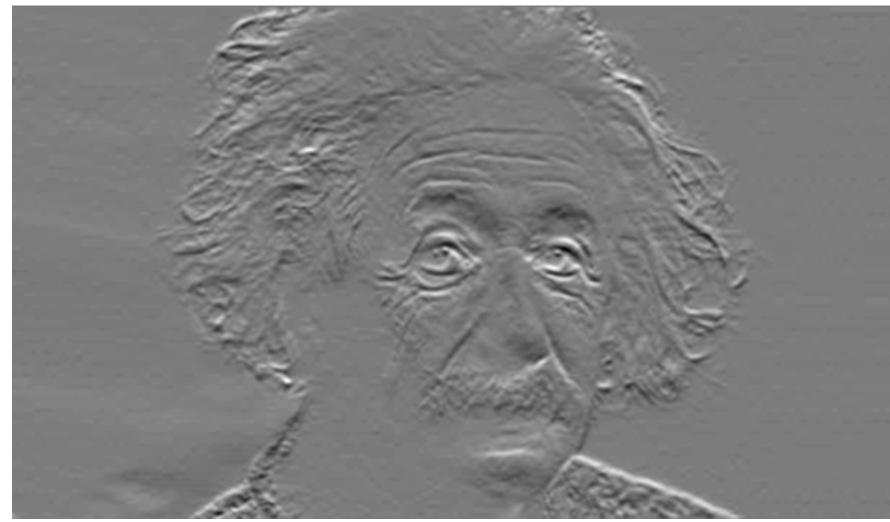


original image

Compute Gradients (DoG)



X-Derivative of Gaussian



Y-Derivative of Gaussian

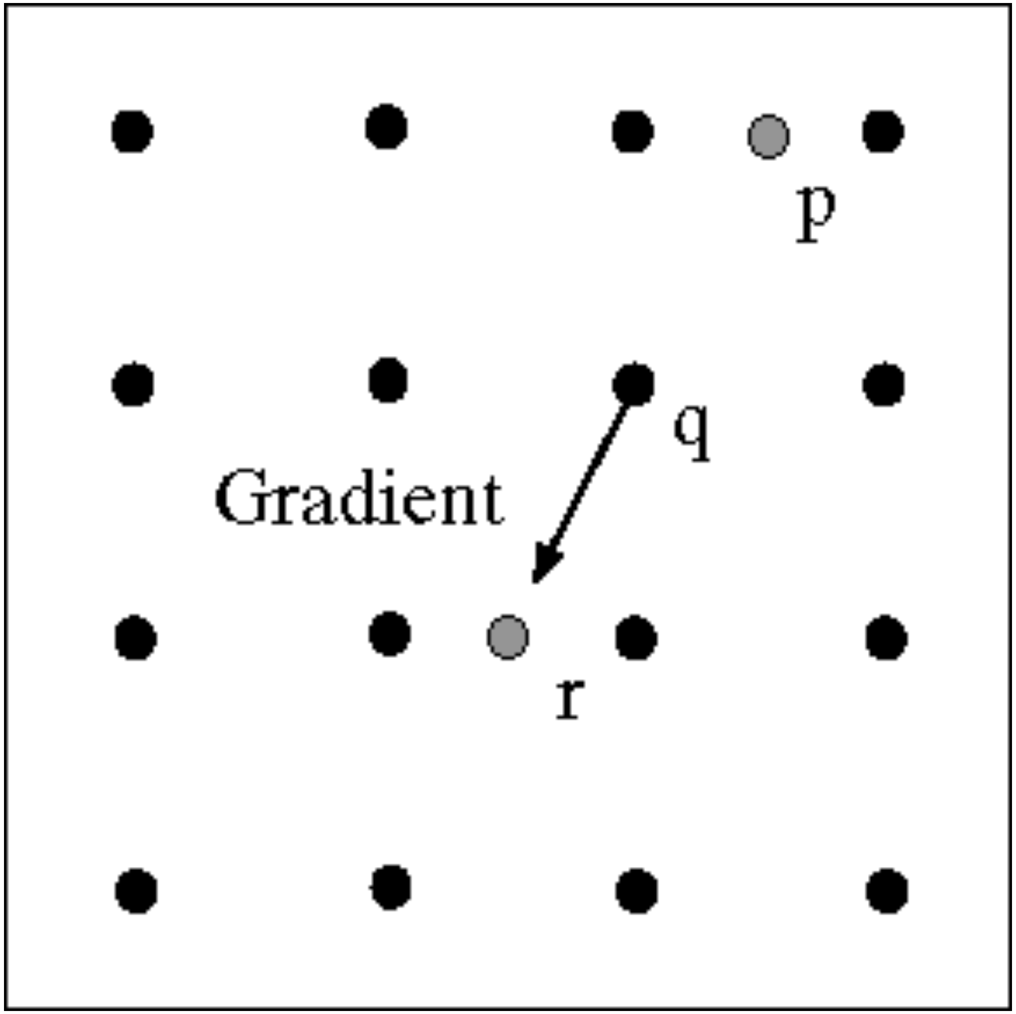
Gradient magnitude and orientation

- Orientation is undefined at pixels with 0 gradient

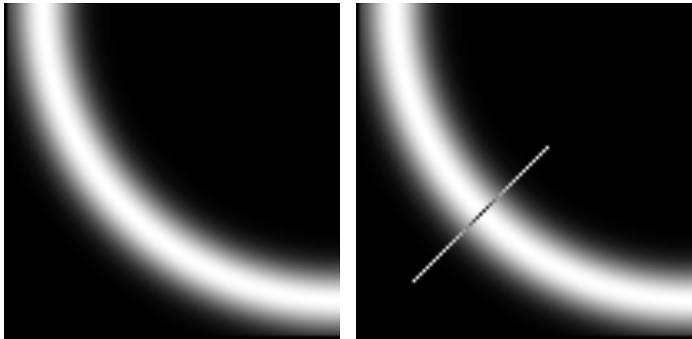
Magnitude

Orientation
theta = numpy.arctan2(gy, gx)

Non-maximum suppression for each orientation



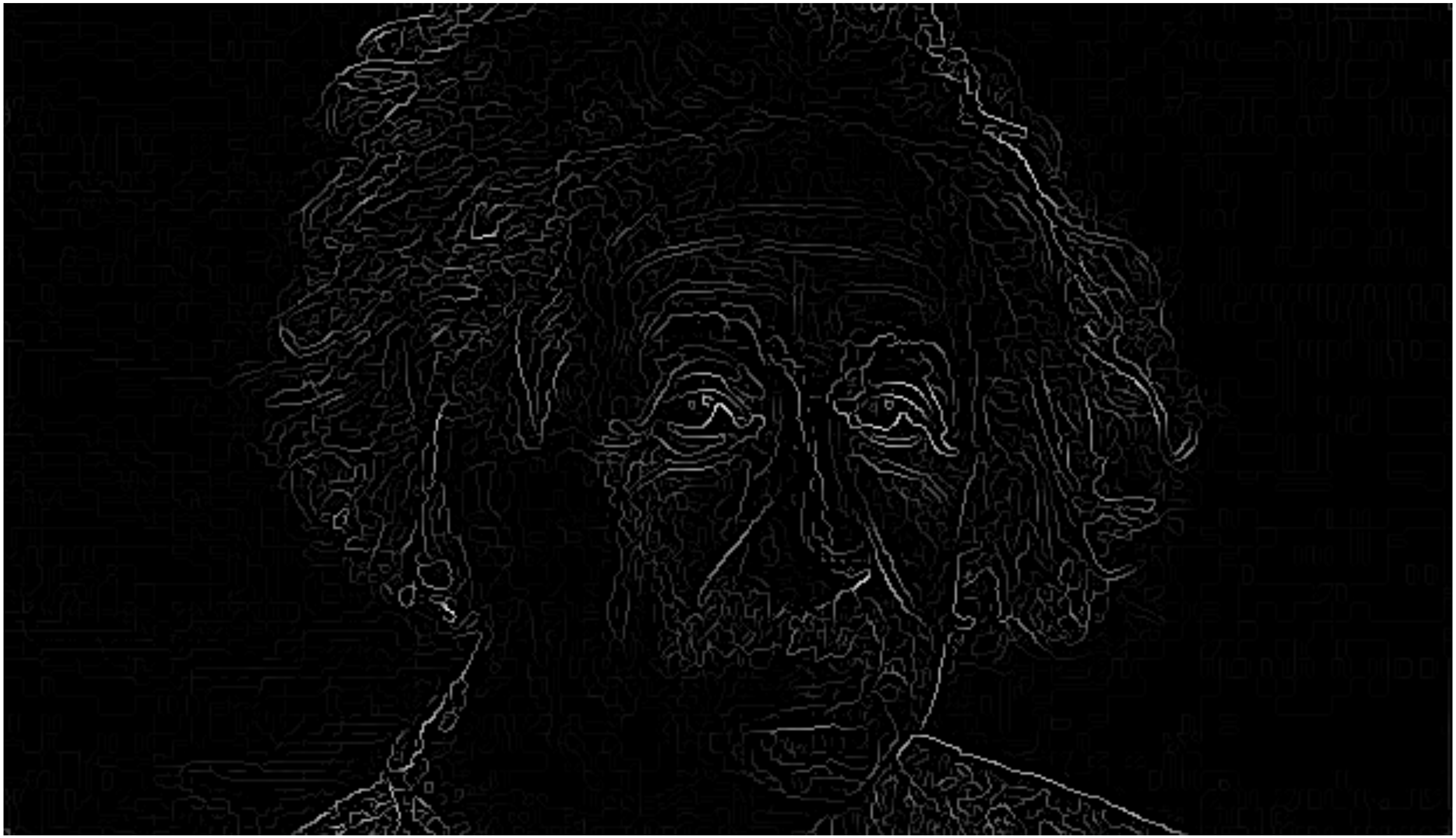
At q, we have a maximum if the value is larger than those at both p and at r. Interpolate to get these values.



Source: D. Forsyth

Before Non-max Suppression

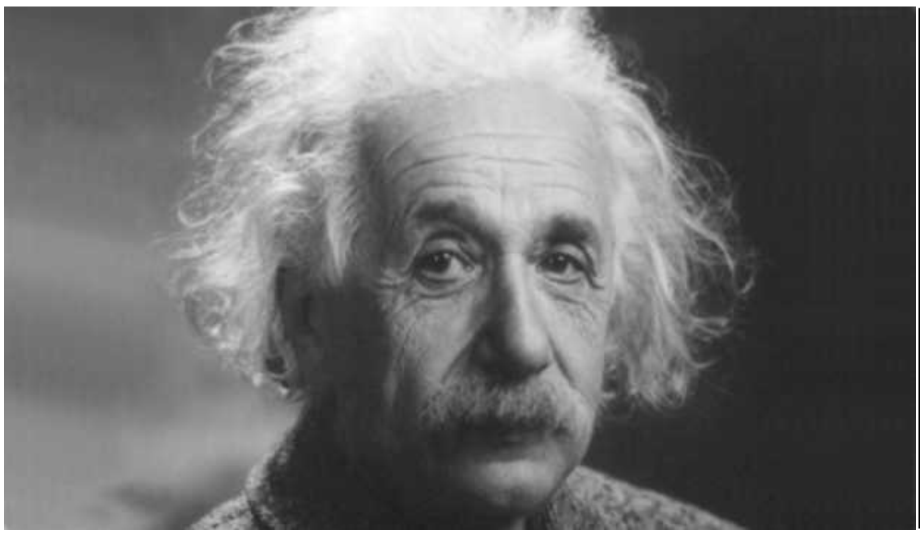
After Non-max Suppression



Hysteresis thresholding

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

Final Canny Edges

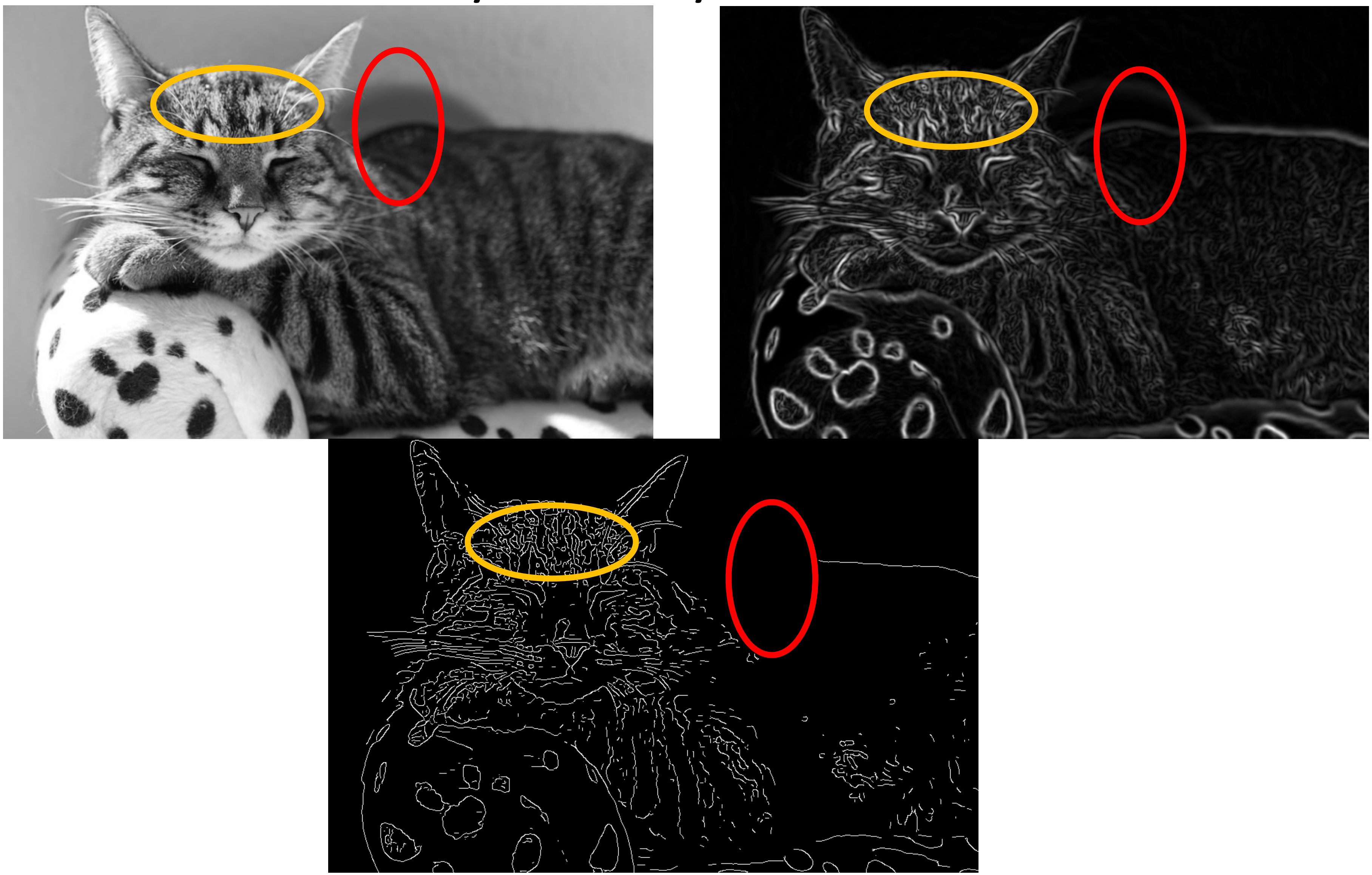


Canny edge detector

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

Does Canny always work?



The challenges of edge detection

- Texture
- Low-contrast boundaries