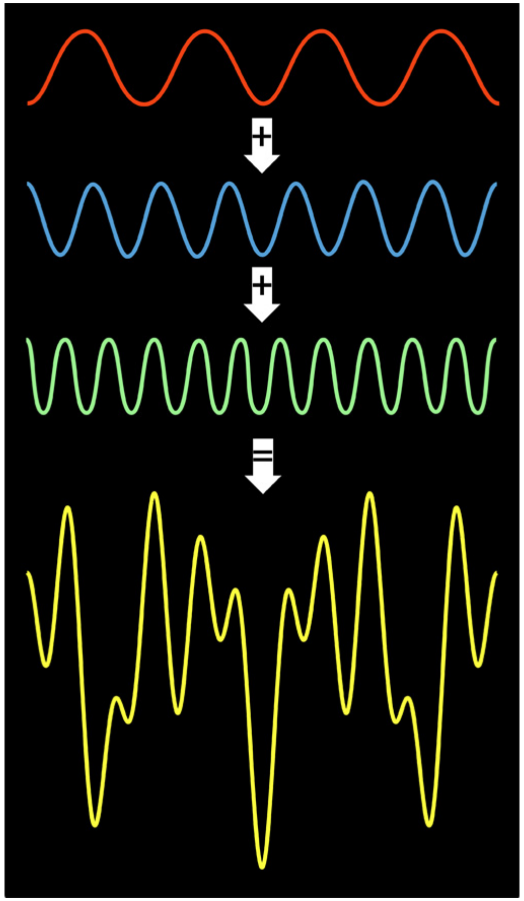
BIL 717
Image Processing
Mar. 5, 2013

Erkut Erdem
Dept. of Computer Engineering
Hacettepe University

Edge Detection

Signals and Images

- A signal is composed of low and high frequency components



low frequency components: smooth / piecewise smooth

Neighboring pixels have similar brightness values

You're within a region

high frequency components: oscillatory

Neighboring pixels have different brightness values

You're either at the edges or noise points

Edge detection

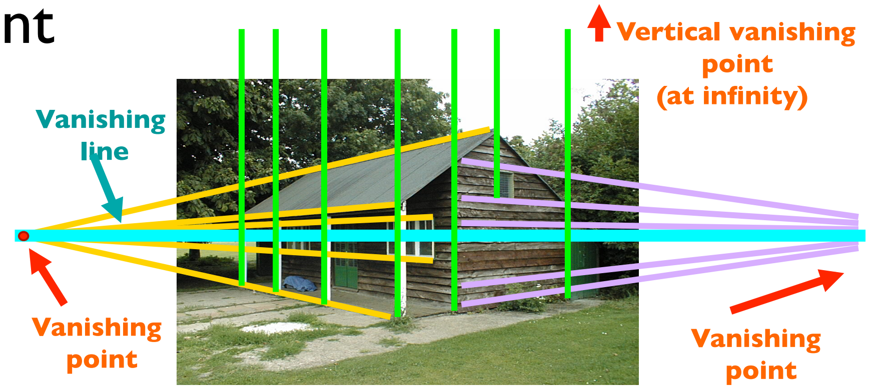
- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels
- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Slide credit: D. Lowe

Why do we care about edges?

- Extract information, recognize objects

- Recover geometry and viewpoint

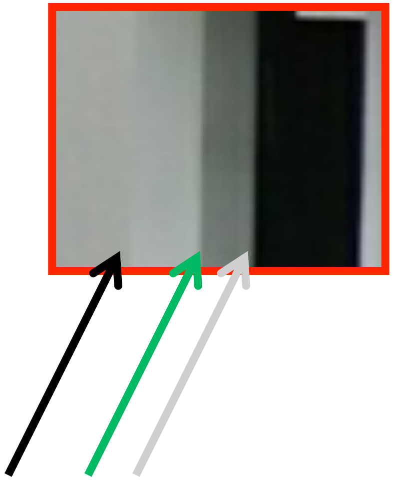


Source: J. Hays

Closeup of edges

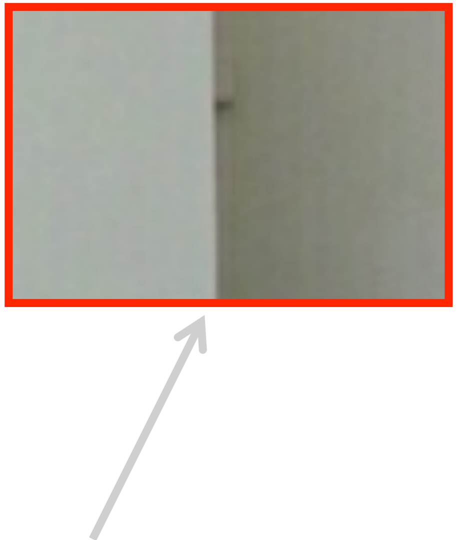
Slide credit: D. Hoiem

Closeup of edges



Slide credit: D. Hoiem

Closeup of edges

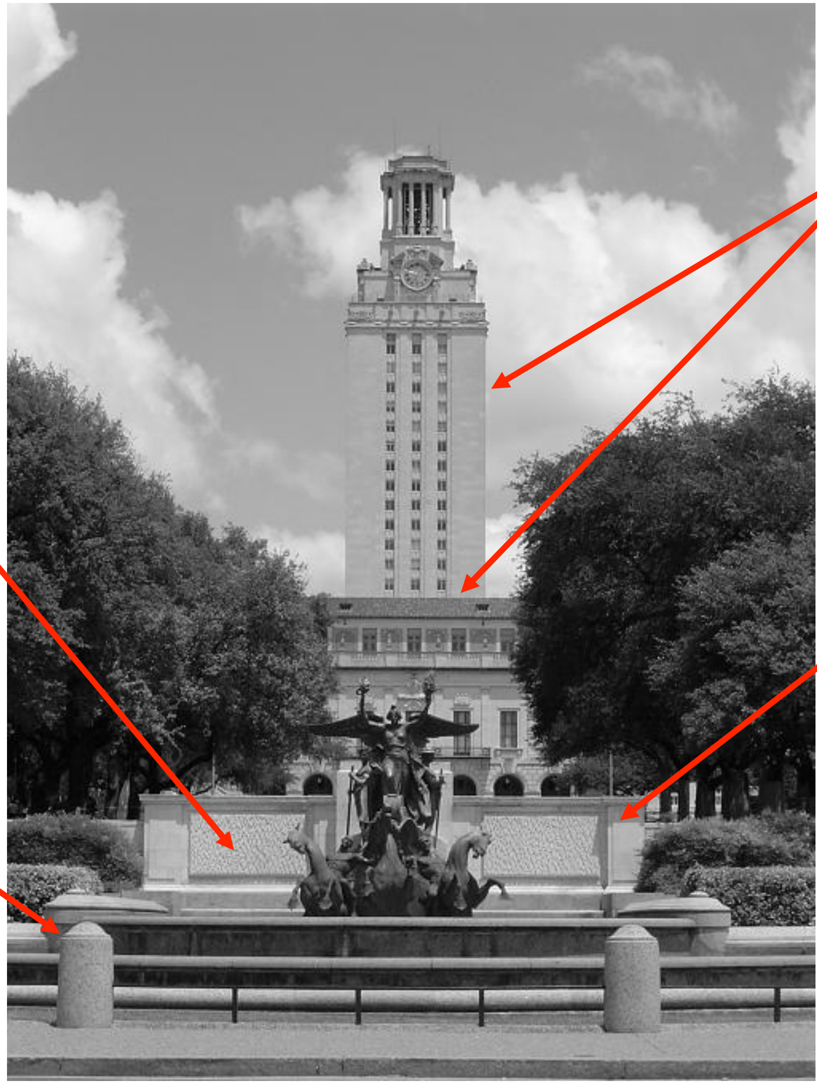


Slide credit: D. Hoiem

Closeup of edges

Slide credit: D. Hoiem

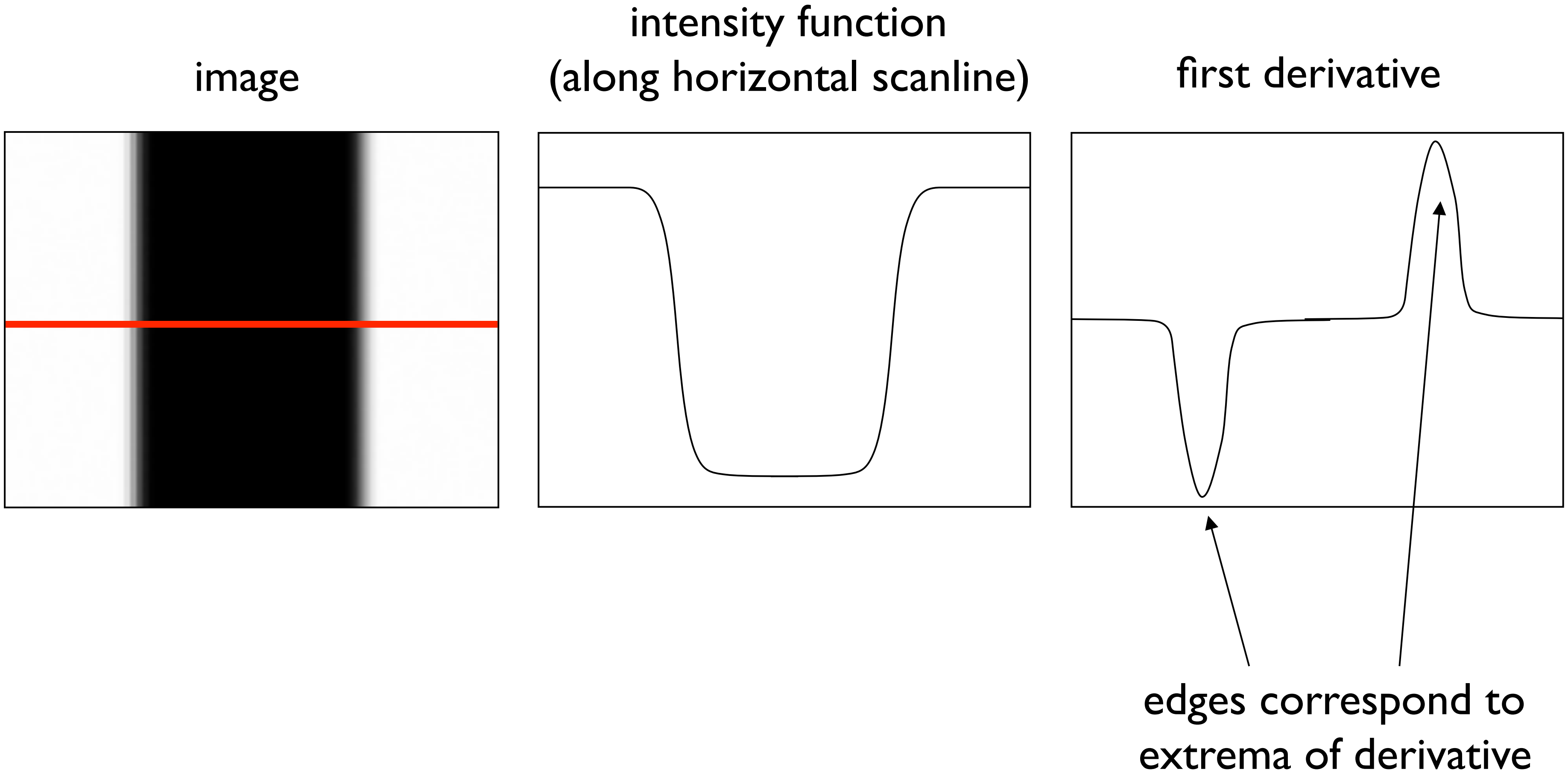
What causes an edge?



Slide credit: K. Grauman

Characterizing edges

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



Slide credit: K. Grauman

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:

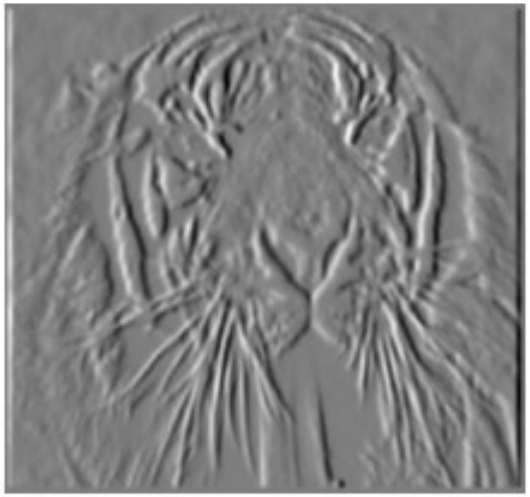
$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1, y) - f(x,y)}{1}$$

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

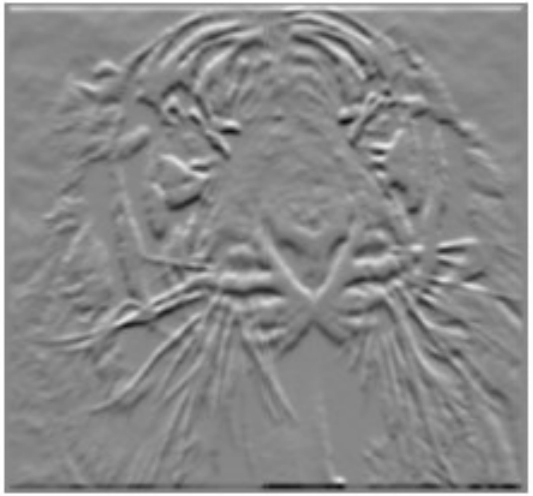
Slide credit: K. Grauman

Partial derivatives of an image

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



-1	1



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

-1
1

Which shows changes with respect to x?

Slide credit: K. Grauman

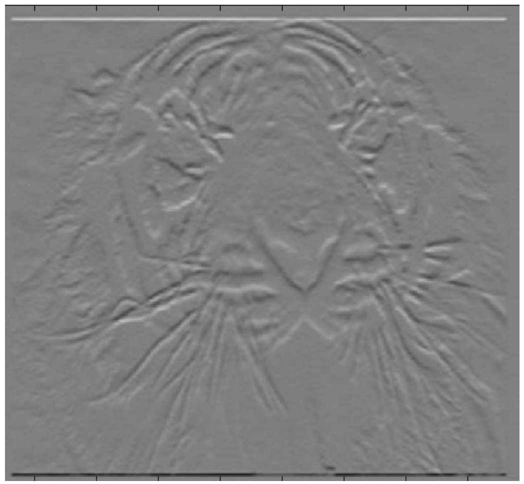
Assorted finite difference filters

Prewitt: $M_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$

Sobel: $M_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$; $\boxed{M_y = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}}$

Roberts: $M_x = \begin{array}{|c|c|}\hline 0 & 1 \\ \hline -1 & 0 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array}$

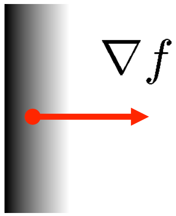
```
>> My = fspecial(‘sobel’);
>> outim = imfilter(double(im), My);
>> imagesc(outim);
>> colormap gray;
```



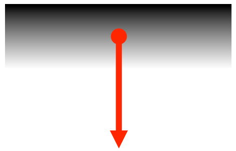
Slide credit: K. Grauman

Image gradient

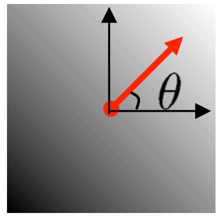
- The gradient of an image: $$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

- 

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



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



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

The 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

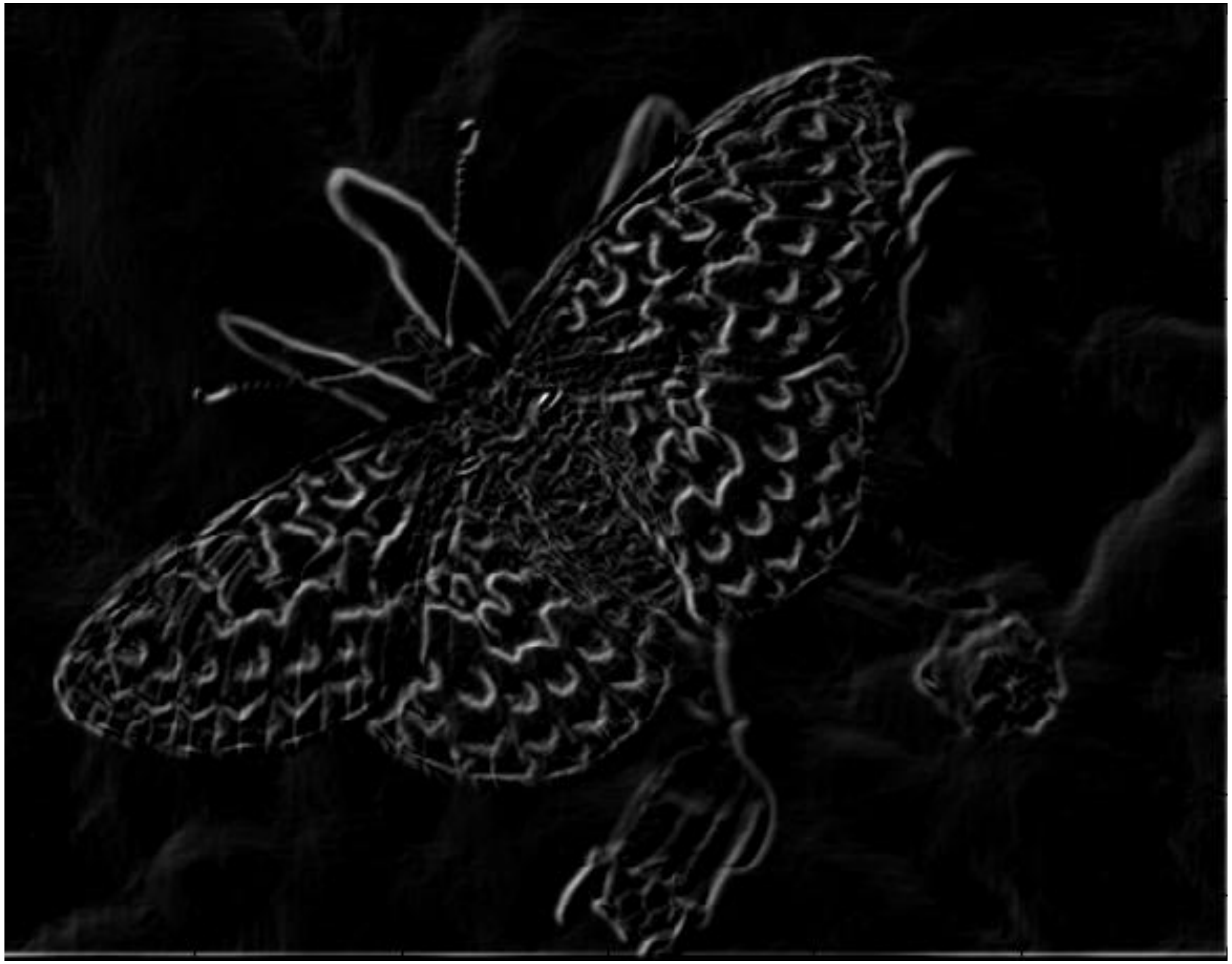
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Slide credit: S. Seitz

Original Image

Slide credit: K. Grauman

Gradient magnitude image



Slide credit: K. Grauman

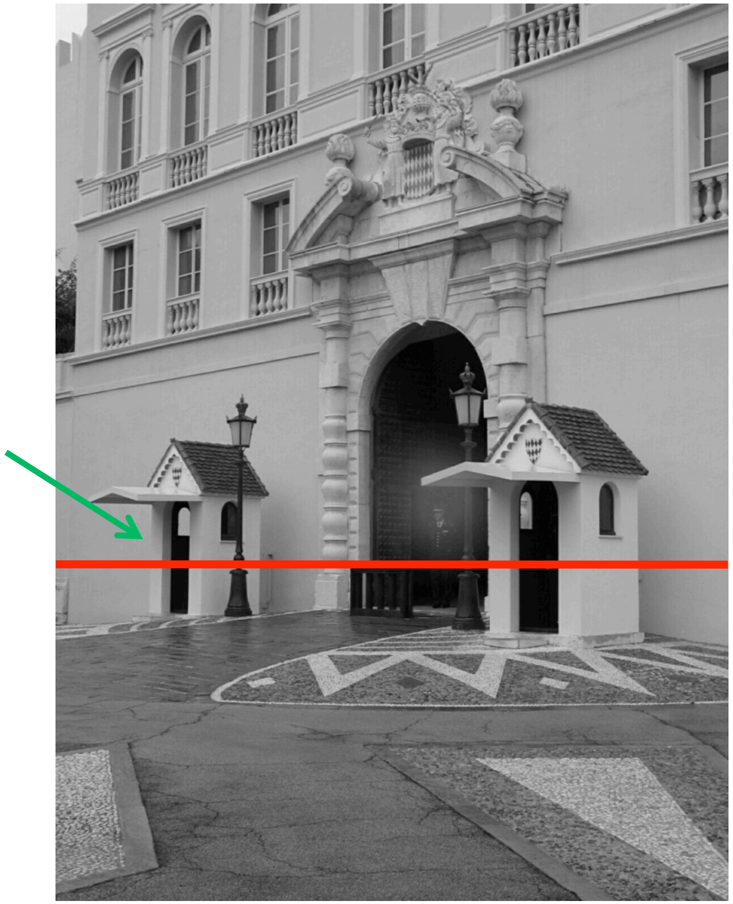
Thresholding gradient with a lower threshold

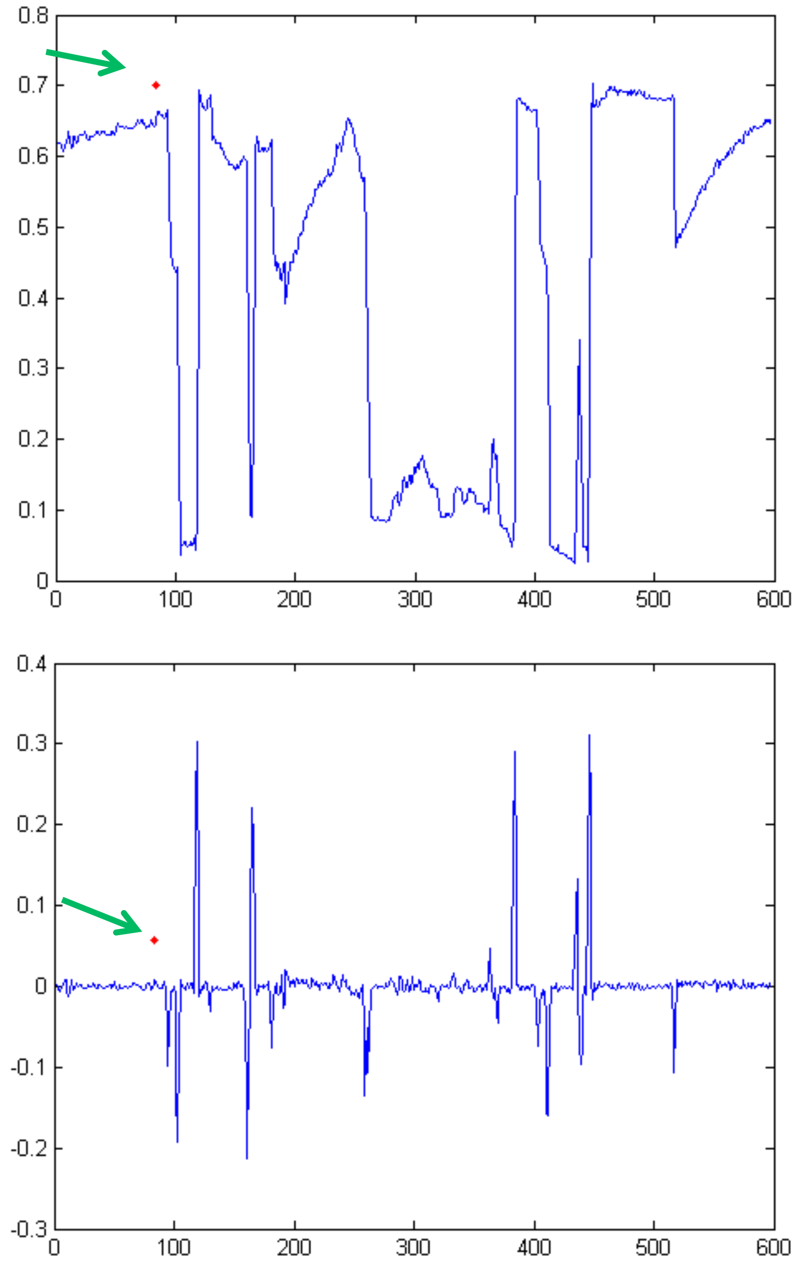
Slide credit: K. Grauman

Thresholding gradient with a higher threshold

Slide credit: K. Grauman

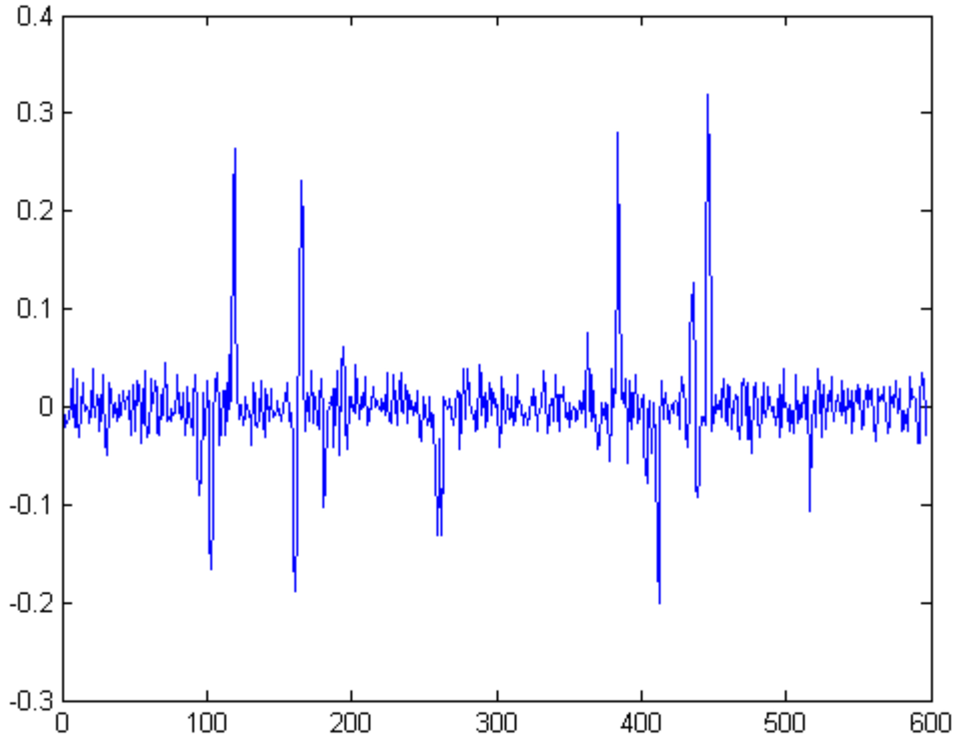
Intensity profile





Slide credit: D. Hoiem

With a little Gaussian noise



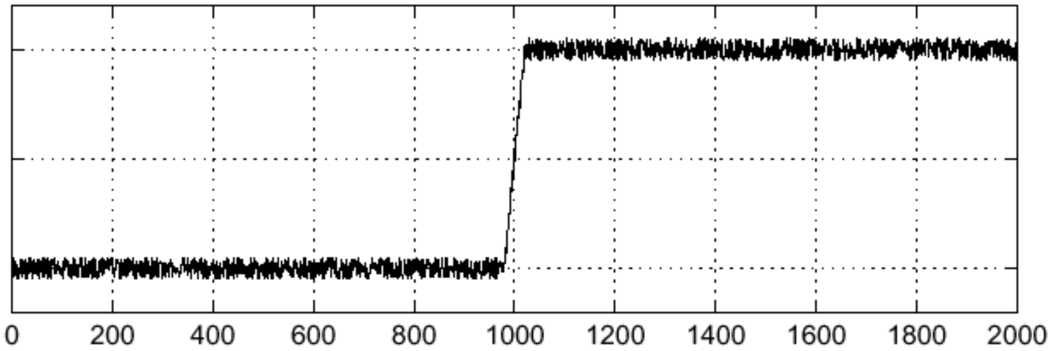
Gradient

Slide credit: D. Hoiem

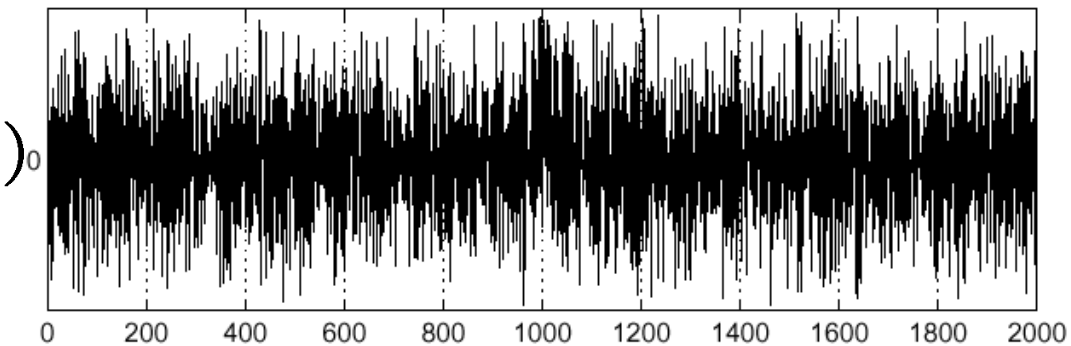
Effects of noise

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

$f(x)$



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



Where is the edge?

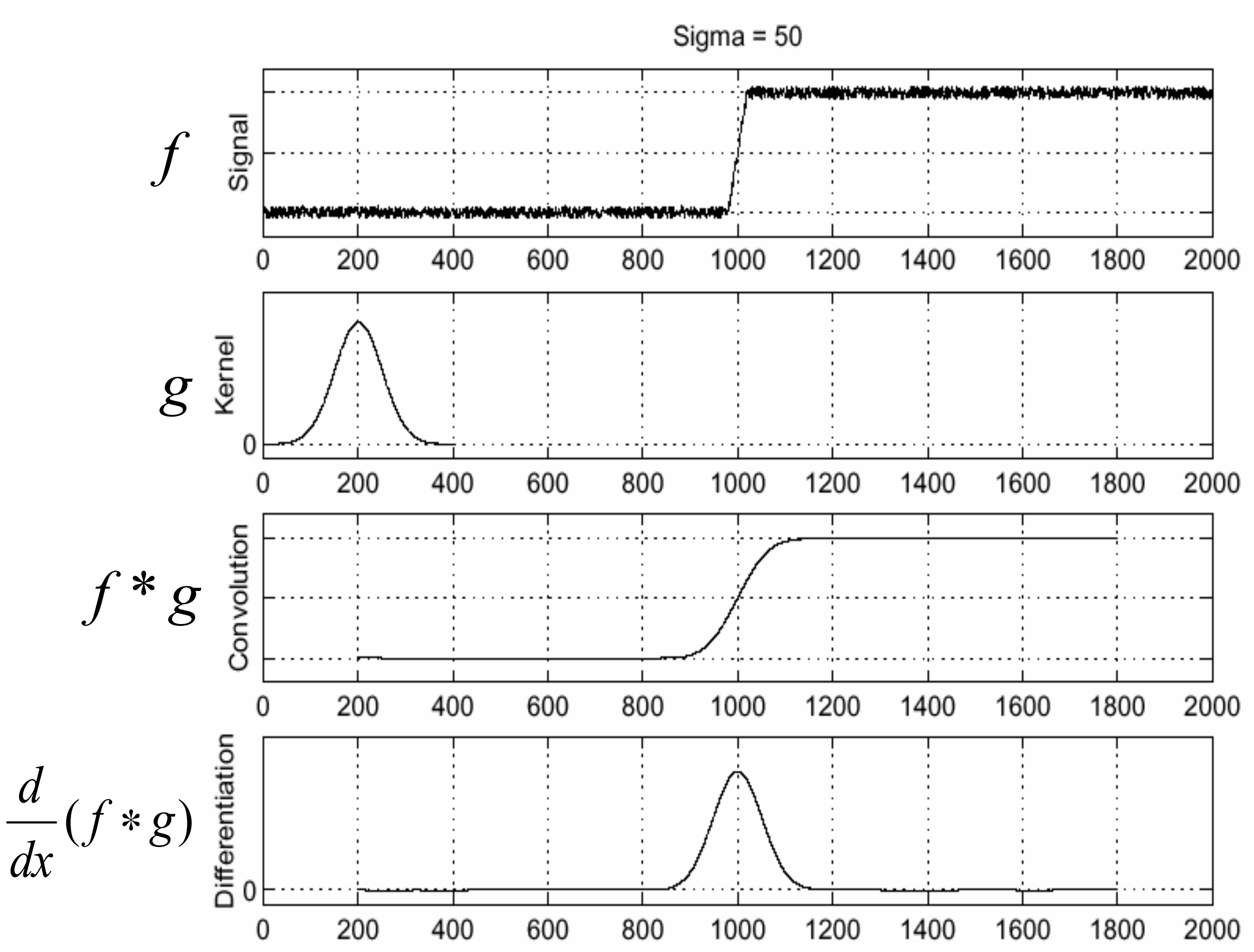
Slide credit: S. Seitz

Effects of noise

- Difference filters respond strongly to noise
  - Image noise results in pixels that look very different from their neighbors
  - Generally, the larger the noise the stronger the response
- What can we do about it?

Slide credit: D. Forsyth

Solution: smooth first

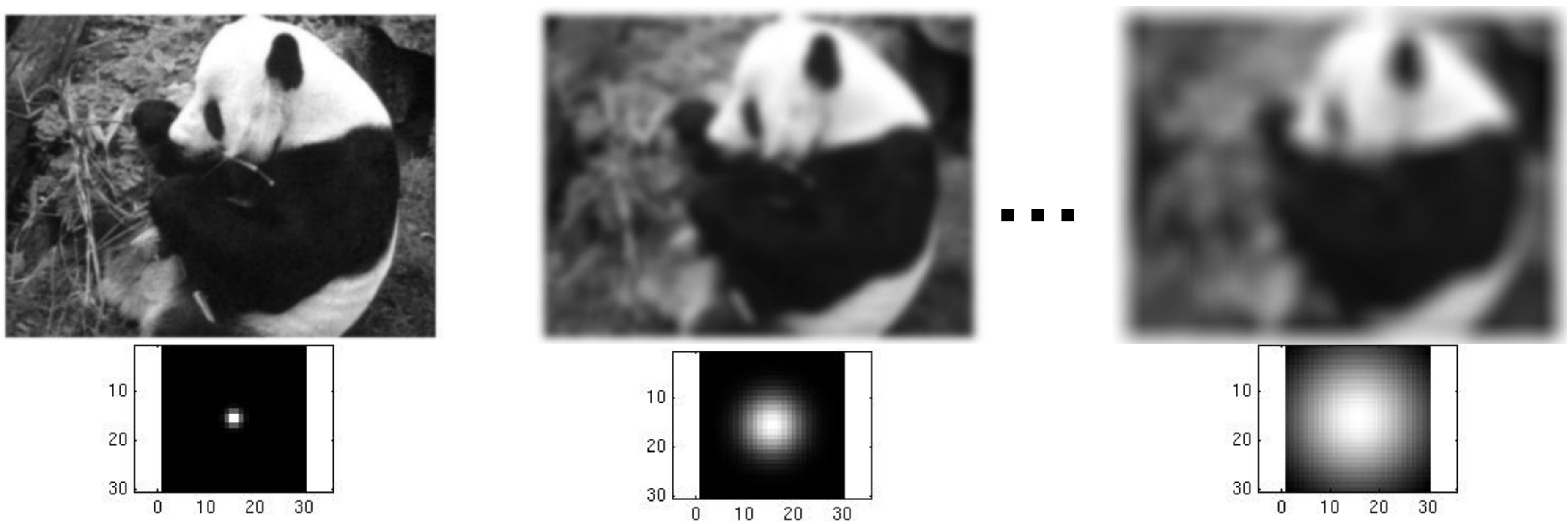


- To find edges, look for peaks in $\frac{d}{dx}(f * g)$

Slide credit: S. Seitz

Smoothing with a Gaussian

Recall: parameter σ is the "scale" / "width" / "spread" of the Gaussian kernel, and controls the amount of smoothing.



Slide credit: K. Grauman

Effect of σ on derivatives

σ = 1 pixel

σ = 3 pixels

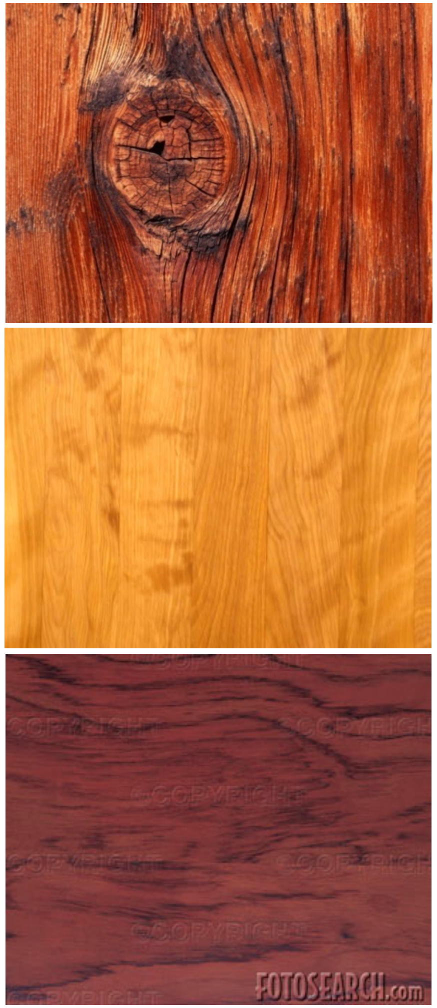
The apparent structures differ depending on Gaussian's scale parameter.

Larger values: larger scale edges detected
Smaller values: finer features detected

Slide credit: K. Grauman

So, what scale to choose?

It depends what we're looking for.



Slide credit: K. Grauman

Smoothing and Edge Detection

- While eliminating noise via smoothing, we also lose some of the (important) image details.
  - Fine details
  - Image edges
  - etc.
- What can we do to preserve such details?
  - Use edge information during denoising!
  - This requires a definition for image edges.

Chicken-and-egg dilemma!

- Edge preserving image smoothing (Next week's topic!)

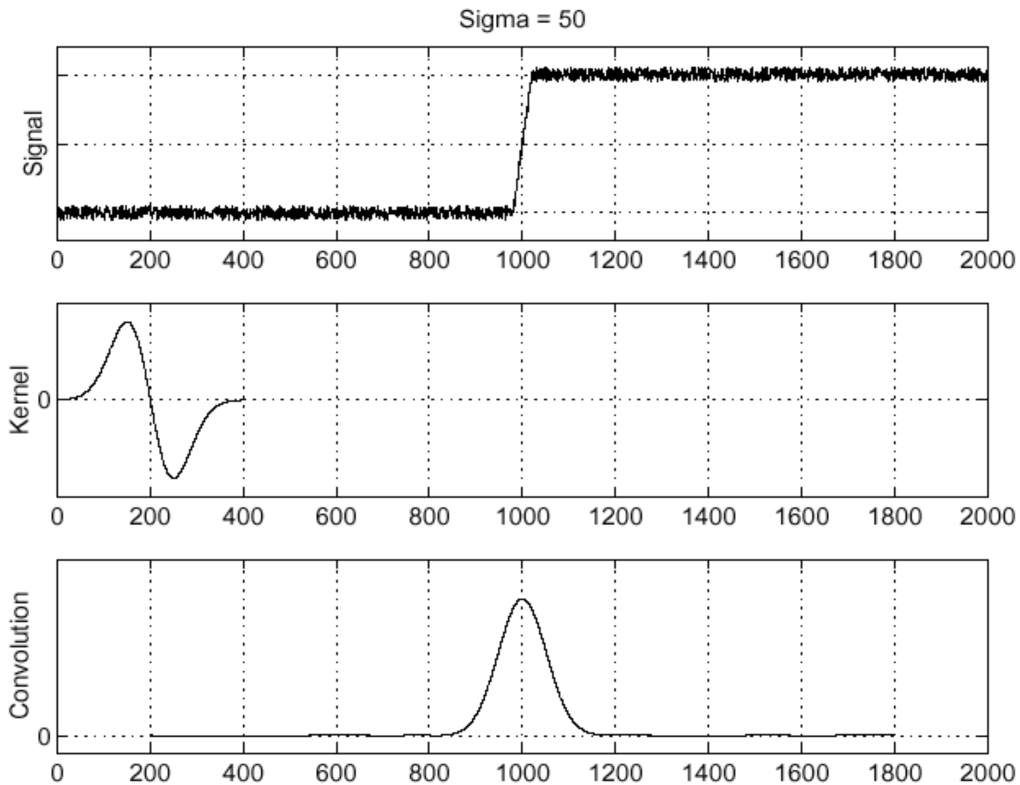
Derivative theorem of convolution

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

f

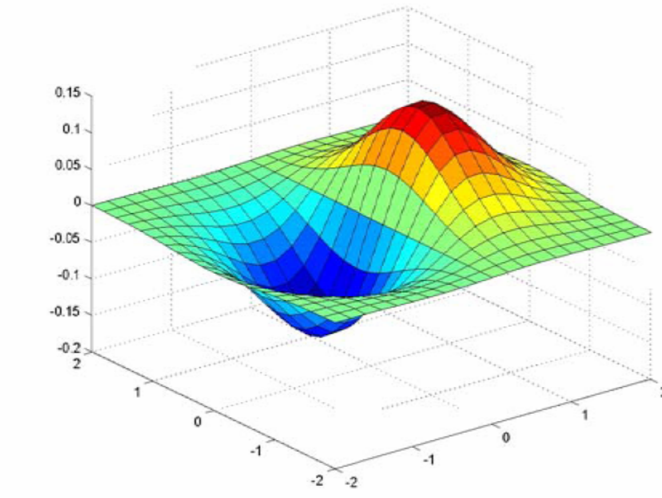
$\frac{d}{dx} g$

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

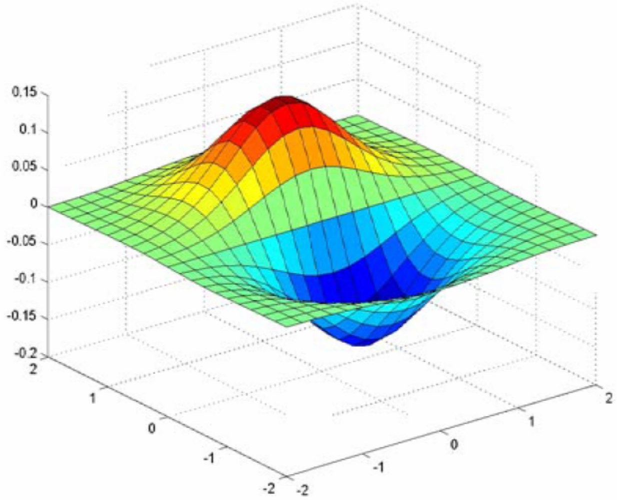


Slide credit: S. Seitz

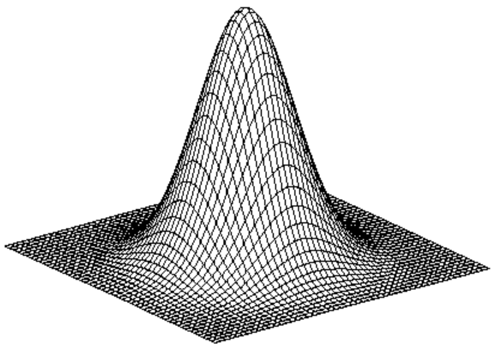
Derivative of Gaussian filter



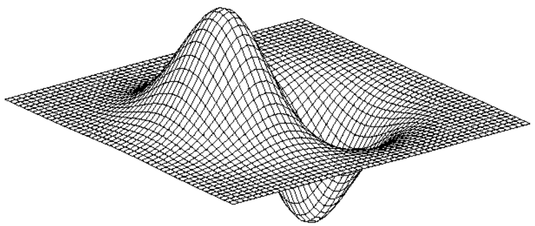
x-direction



y-direction

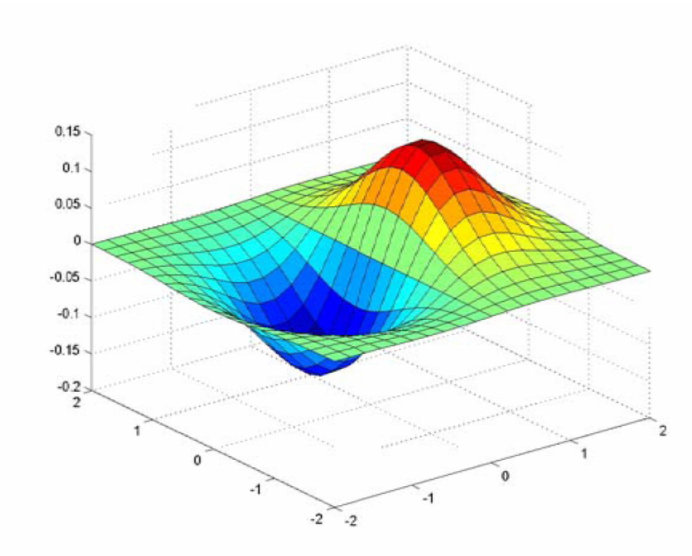


* [1 -1] =

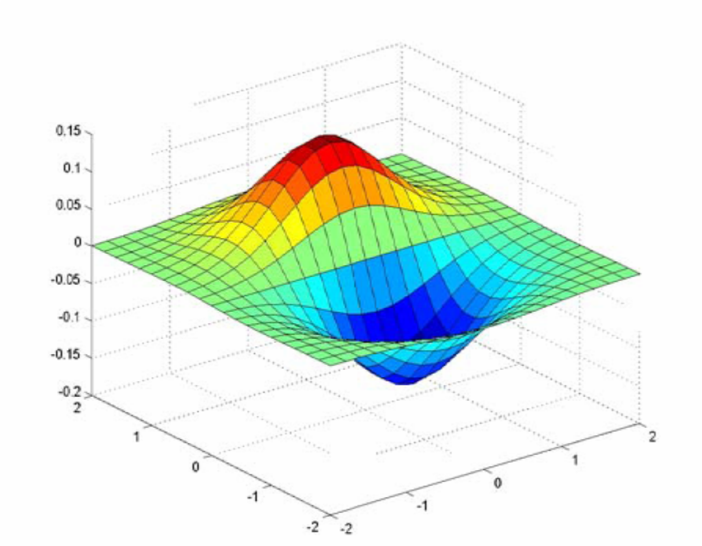


Slide credit: S. Lazebnik

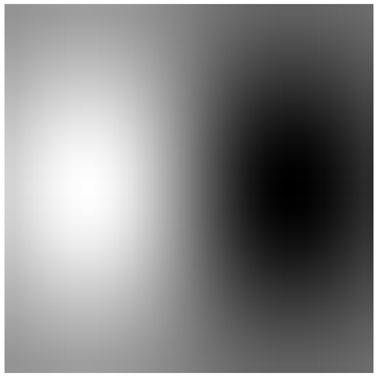
Derivative of Gaussian filter

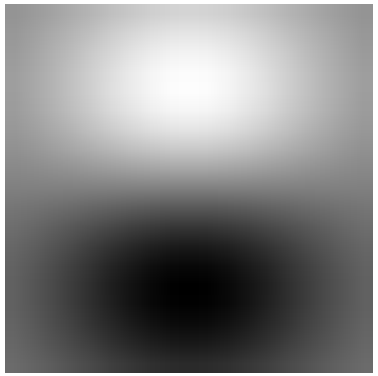


x-direction



y-direction



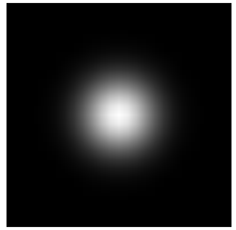


- Which one finds horizontal/vertical edges?

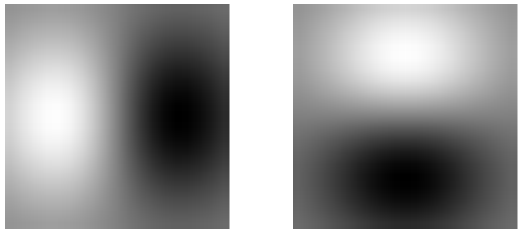
Slide credit: S. Lazebnik

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



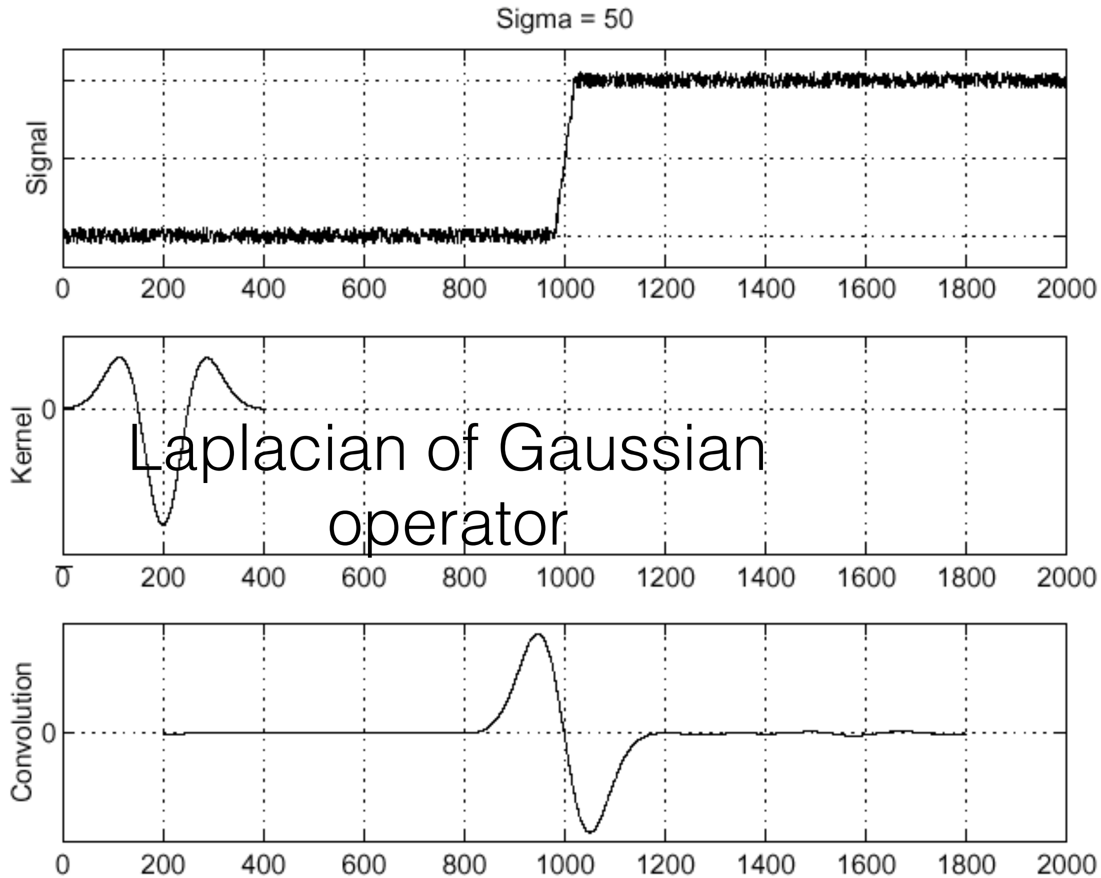
Slide credit: S. Lazebnik

Laplacian of Gaussian

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

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

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

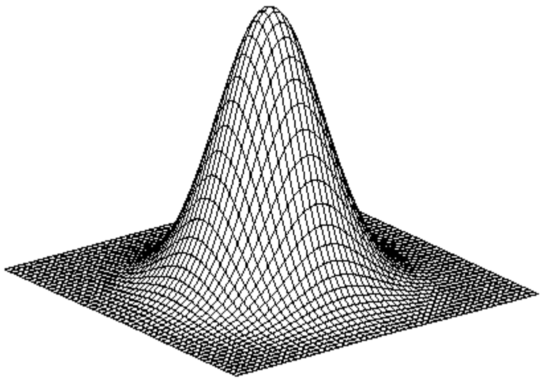


Laplacian of Gaussian operator

Where is the edge? Zero-crossings of bottom graph

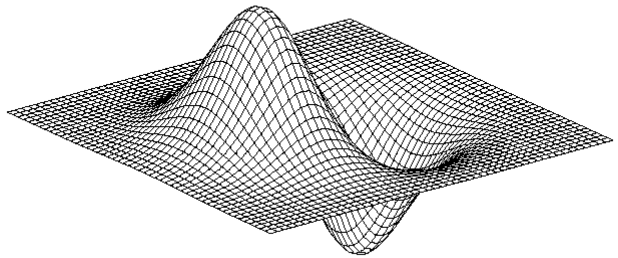
Slide credit: K. Grauman

2D edge detection filters



Gaussian

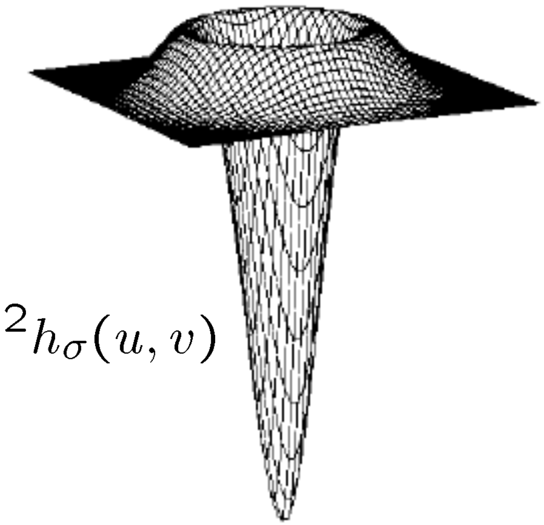
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

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

Laplacian of Gaussian



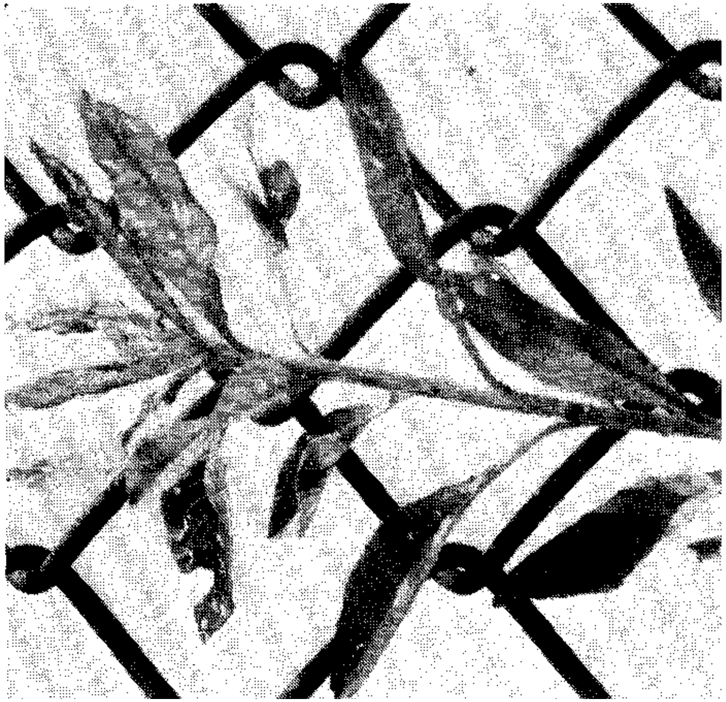
$$\nabla^2 h_\sigma(u,v)$$

- The Laplacian operator:

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

Slide credit: K. Grauman

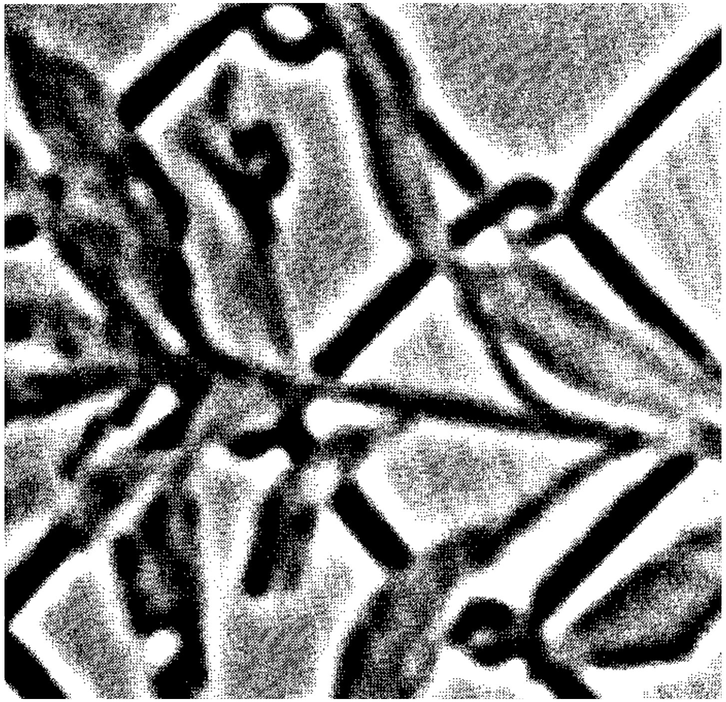
Laplacian of Gaussian



original image

Source: D. Marr and E. Hildreth (1980)

Laplacian of Gaussian



convolution with

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

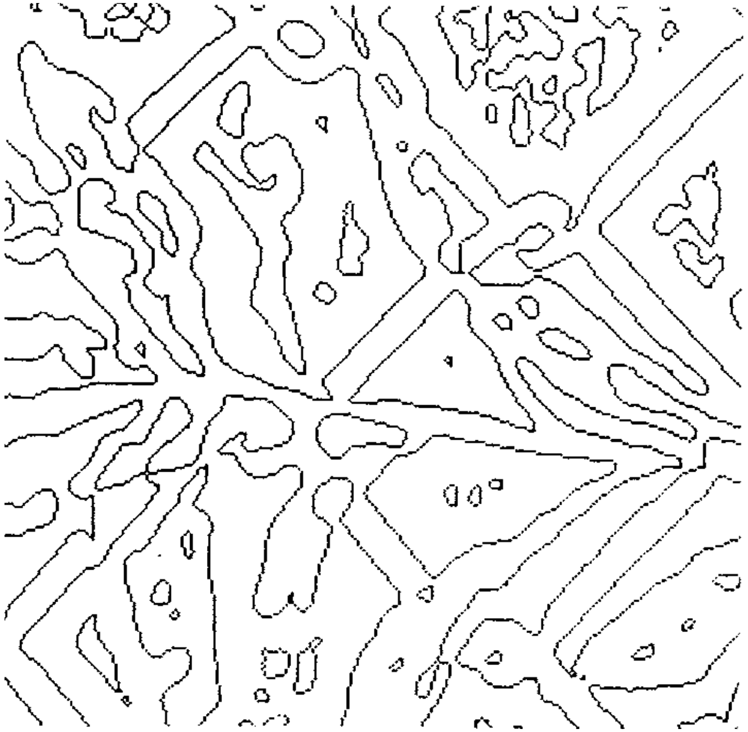
Source: D. Marr and E. Hildreth (1980)

Laplacian of Gaussian

convolution with
$$\nabla^2 h_\sigma(u,v)$$
(pos. values – white, neg. values – black)

Source: D. Marr and E. Hildreth (1980)

Laplacian of Gaussian



zero-crossings

Source: D. Marr and E. Hildreth (1980)

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

Slide credit: L. Fei-Fei

The Canny edge detector

original image (Lena)

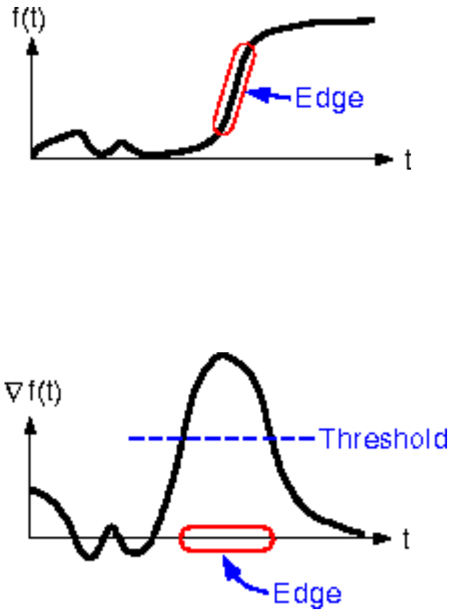
Slide credit: K. Grauman

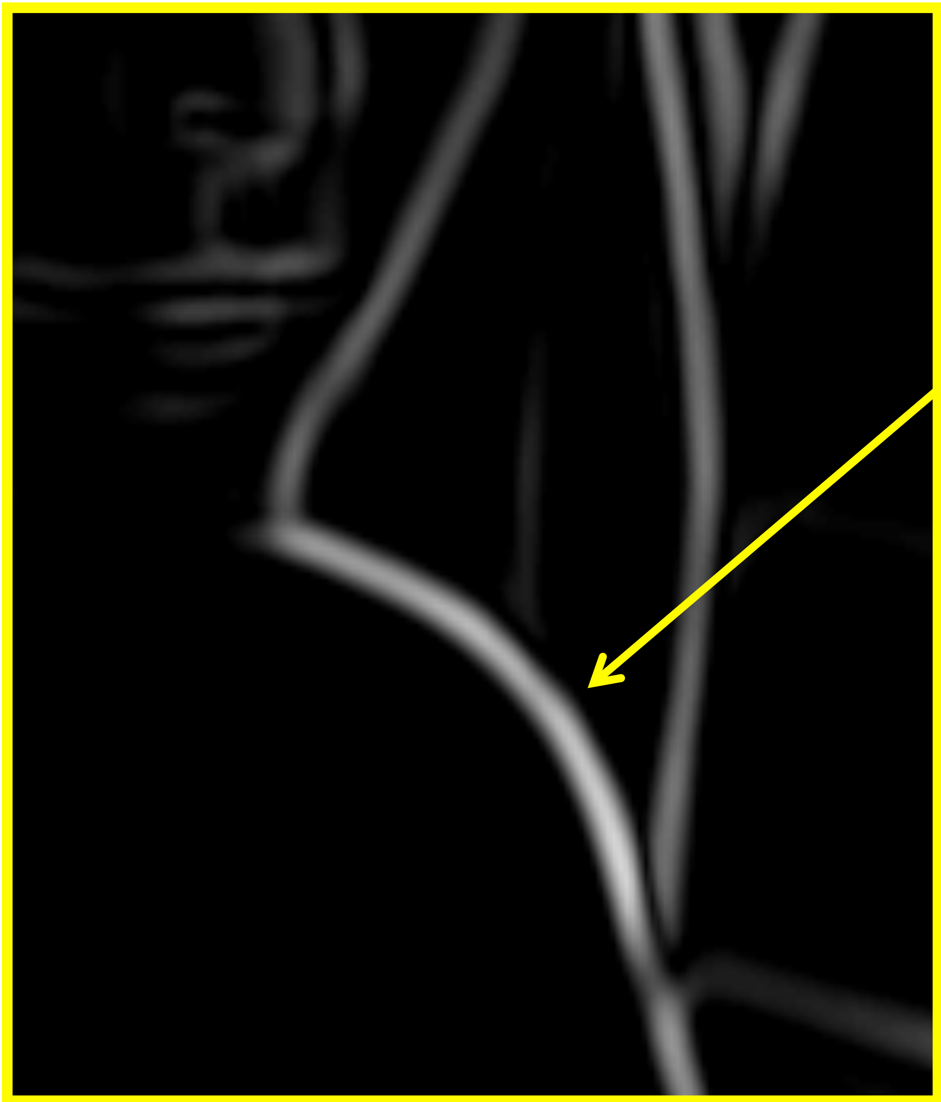
The Canny edge detector

thresholding

Slide credit: K. Grauman

The Canny edge detector

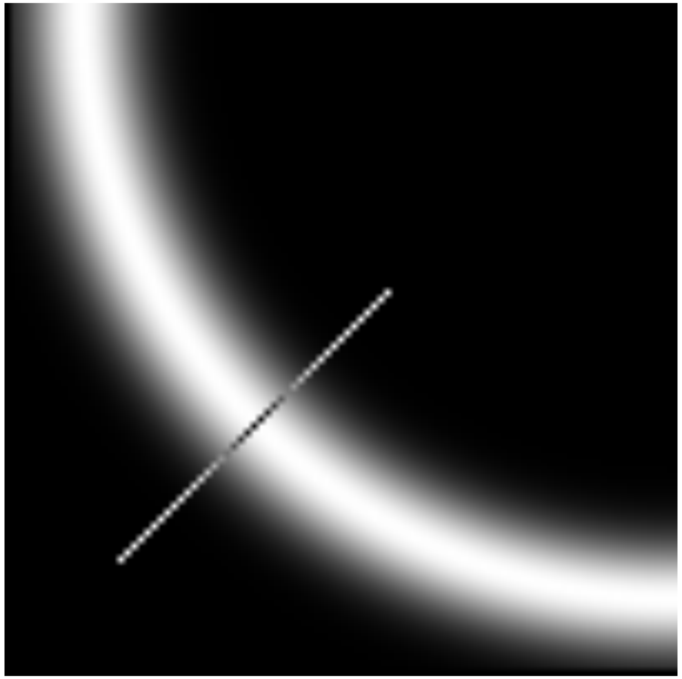


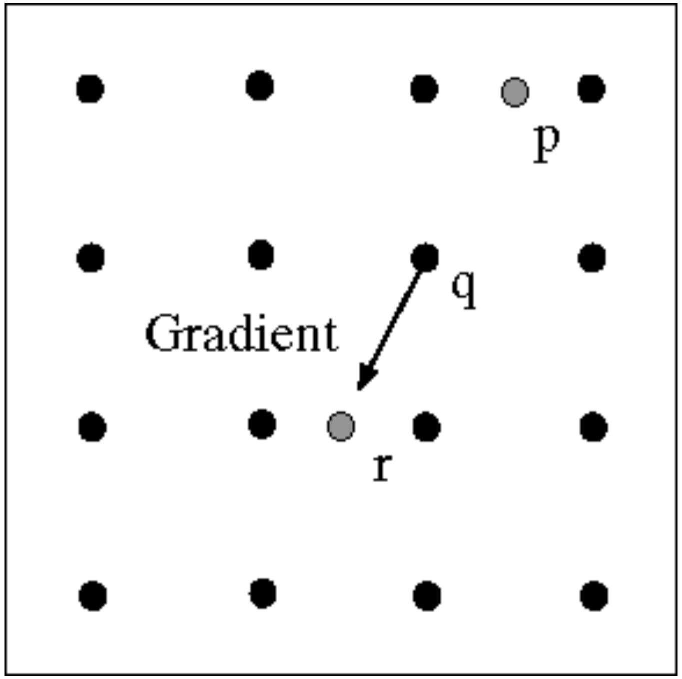


How to turn these thick regions of the gradient into curves?

Slide credit: K. Grauman

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 credit: K. Grauman

The Canny Edge Detector

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

thinning

(non-maximum suppression)

Slide credit: K. Grauman

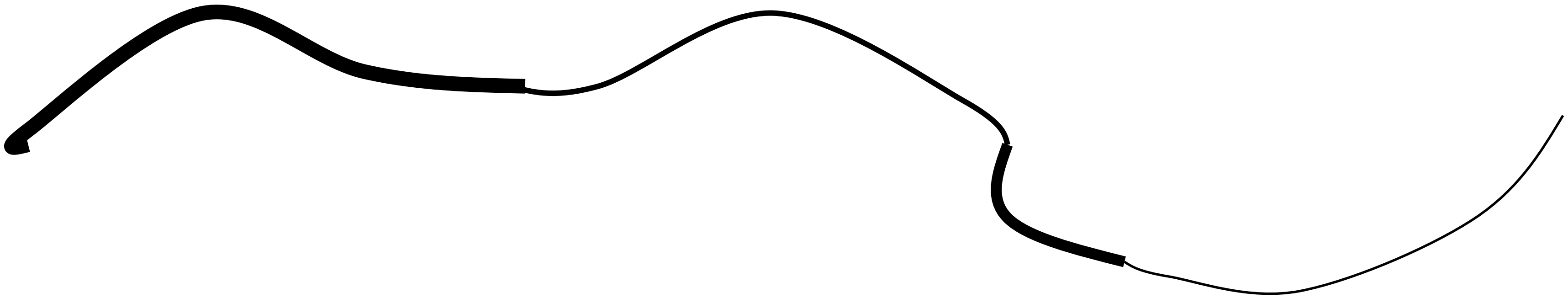
Hysteresis thresholding

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

Slide credit: J. Hays

Hysteresis thresholding

- Check that maximum value of gradient value is sufficiently large
  - drop-outs? use **hysteresis**
    - use a high threshold to start edge curves and a low threshold to continue them.



Slide credit: S. Seitz

Hysteresis thresholding

original image

high threshold
(strong edges)

low threshold
(weak edges)

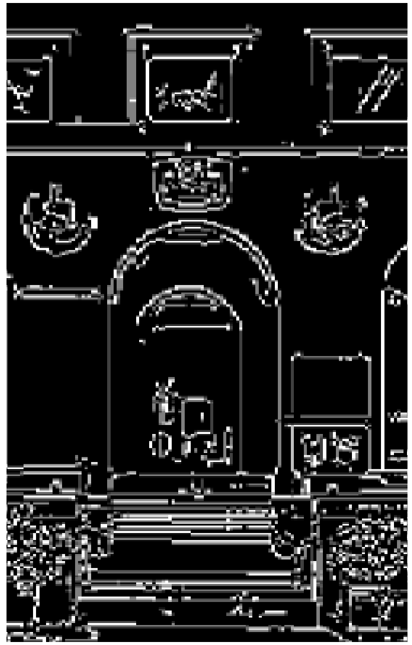
hysteresis threshold

Slide credit: L. Fei-Fei

Hysteresis thresholding

high threshold
(strong edges)

low threshold
(weak edges)



hysteresis threshold

Slide credit: 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

- MATLAB: `edge(image, 'canny');`

Slide credit: D. Lowe, L. Fei-Fei

Effect of σ (Gaussian kernel spread/size)

original

Canny with $\sigma = 1$

Canny with $\sigma = 2$

The choice of σ depends on desired behavior

- large σ detects large scale edges
- small σ detects fine features

Slide credit: S. Seitz

Low-level edges vs. perceived contours

Background

Texture

Shadows

Slide credit: K. Grauman

Edge detection is just the beginning…

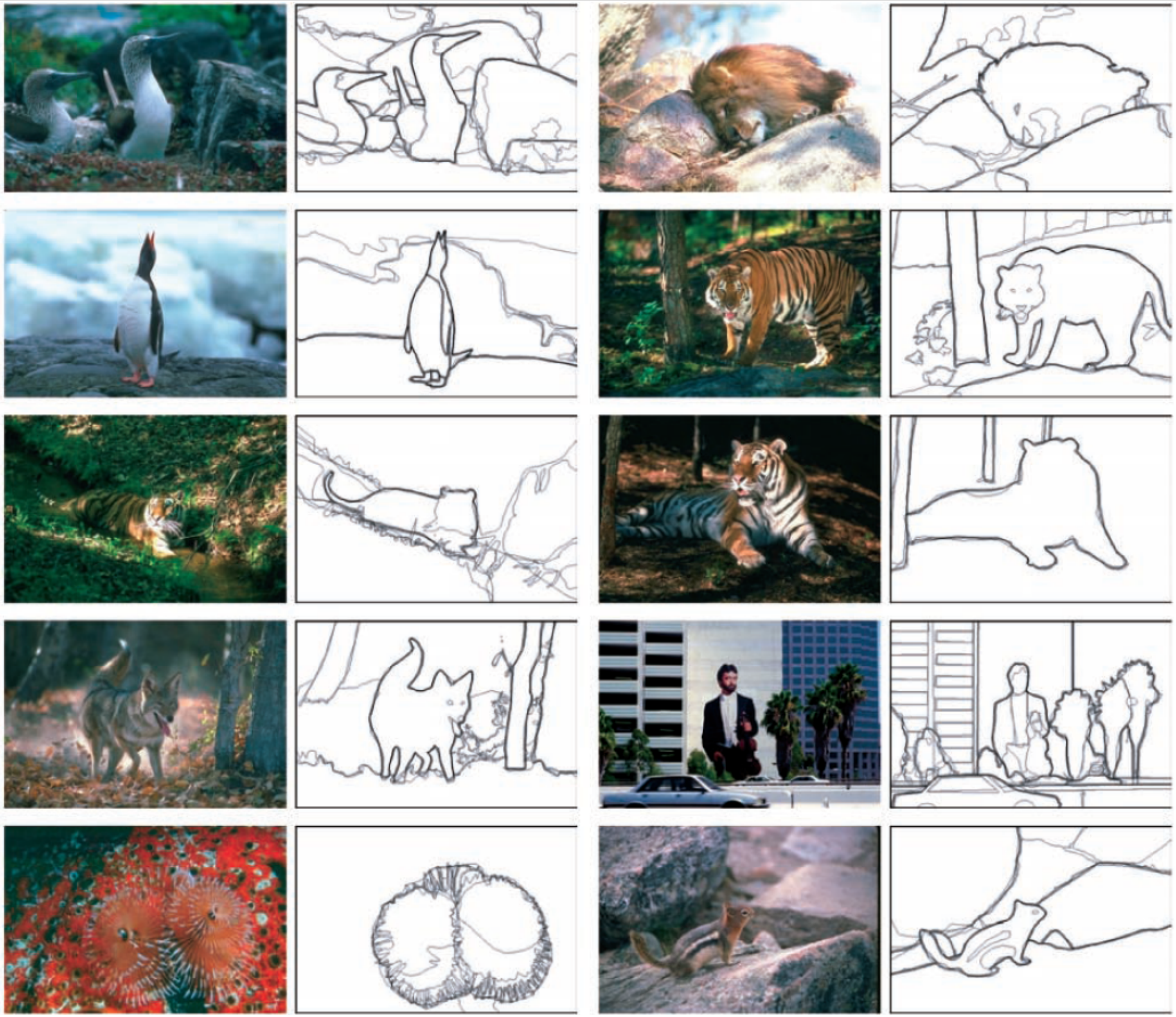
image | human segmentation | gradient magnitude

- Berkeley segmentation database:
  http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Source: S. Lazebnik

Learn from humans which combination of features is most indicative of a "good" contour?

[D. Martin et al. PAMI 2004]



Human-marked segment boundaries

Slide credit: K. Grauman