CMPSCI 370: Intro to Computer Vision

Image processing: #3

linear filtering continued …

University of Massachusetts, Amherst

February 18, 2016

Instructor: Subhransu Maji

Administrivia

- Homework 2 due Tue., Feb. 23 before class
  - Linearity of light
  - Color constancy
  - Hybrid images
- Today's lecture
  - Review of last lecture
  - Edge detection

Joys of computer vision research

http://xkcd.com/1425/

Motivation: Image de-noising

- How can we reduce noise in a photograph?

Moving average

- Let's replace each pixel with a ***weighted*** average of its neighborhood
- The weights are called the ***filter kernel***
- What are the weights for the average of a 3x3 neighborhood?

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

"box filter"

Source: D. Lowe 5

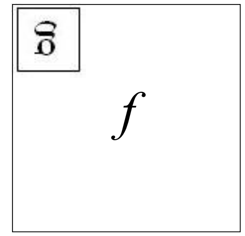
5

Convolution

- Let *f* be the image and *g* be the kernel. The output of convolving *f* with *g* is denoted *f * g*.

$$(f * g)[m,n] = \sum_{k,l} f[m-k, n-l]\, g[k,l]$$

Convention: kernel is "flipped" for convolution

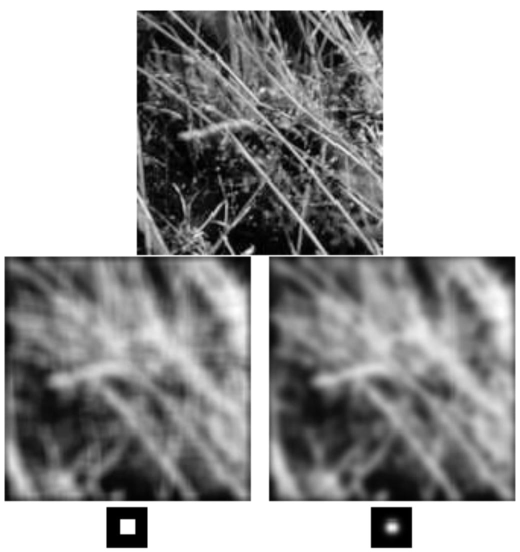


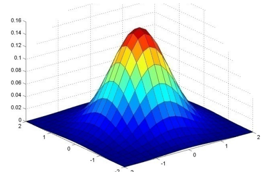
- MATLAB functions: conv2, filter2, imfilter

Source: F. Durand 6

6

Gaussian vs. box filtering





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

7

7

Reducing salt-and-pepper noise

3x3 5x5 7x7

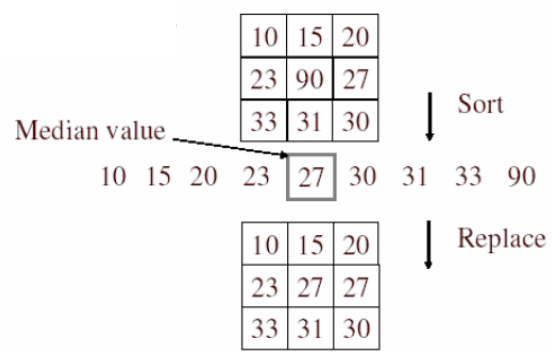
Gaussian smoothing fails to get rid of salt-and-pepper noise

8

8

Alternative idea: Median filtering

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



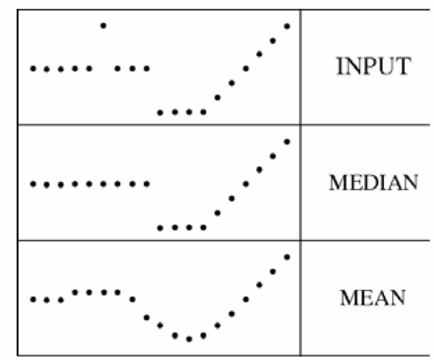
The median filtering is not linear

Source: K. Grauman 9

Median filter

- What advantage does median filtering have over Gaussian filtering?
- Answer: Robustness to outliers

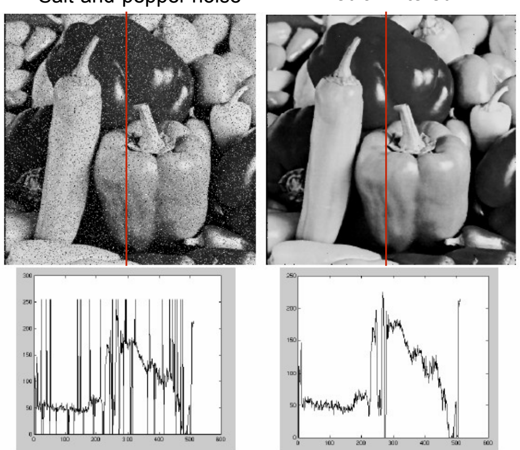
filters have width 5 :



Source: K. Grauman 10

Median filter

Salt-and-pepper noise | Median filtered



MATLAB: medfilt2(image, [h w])

Source: M. Hebert 11

Gaussian vs. median filtering

3x3 | 5x5 | 7x7

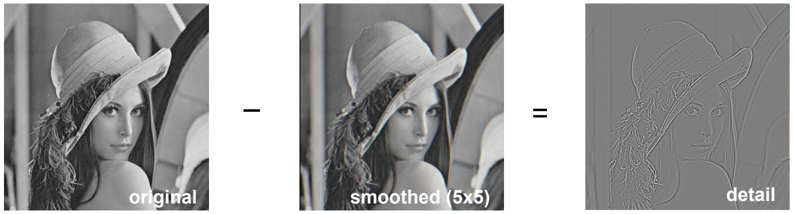
Gaussian

Median

12

Sharpening revisited

What does blurring take away?



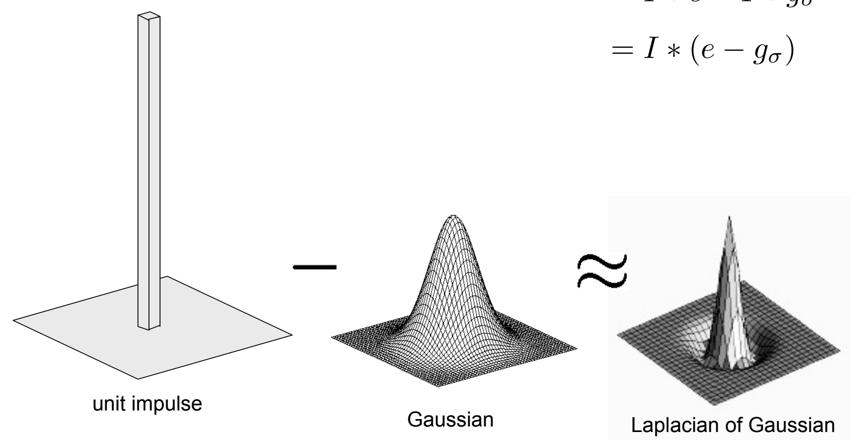
Let's add it back:

13

Sharpening filter

$$I = blurry(I) + sharp(I)$$

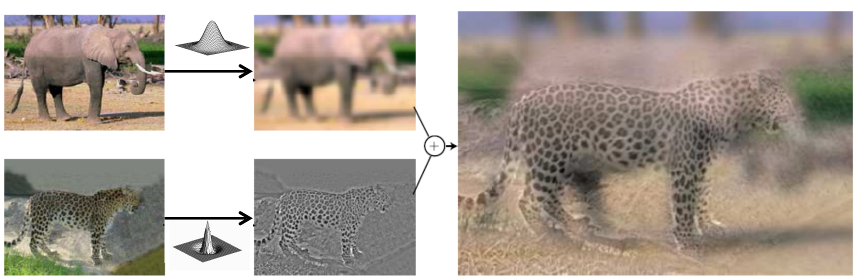
$$\begin{aligned} sharp(I) &= I - blurry(I) \\ &= I * e - I * g_\sigma \\ &= I * (e - g_\sigma) \end{aligned}$$



14

Application: Hybrid Images

Gaussian Filter



Laplacian Filter

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

15

Changing expression

SIGGRAPH2006

Sad ←→ Surprised

motorcycle and bicycle

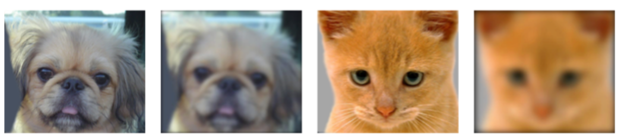
dolphin and car

Homework 2, part 3

$$I_{hybrid} = blurry(I_1, \sigma_1) + sharp(I_2, \sigma_2) = I_1 * g(\sigma_1) + I_2 - I_2 * g(\sigma_2)$$

↑

$$I = blurry(I) + sharp(I)$$



dog image — $\sigma = 4$ — cat image — $\sigma = 10$

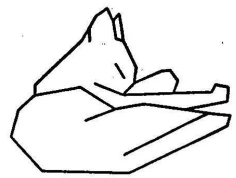
Next: edge detection

Winter in Kraków photographed by Marcin Ryczek

But first, any questions?

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)

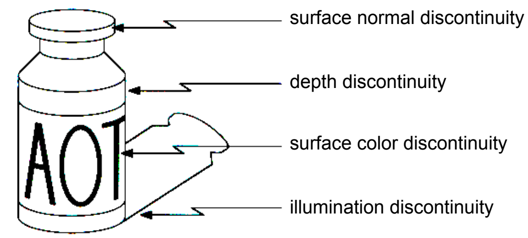


Attneave's Cat (1954)

Source: D. Lowe

Origin of edges

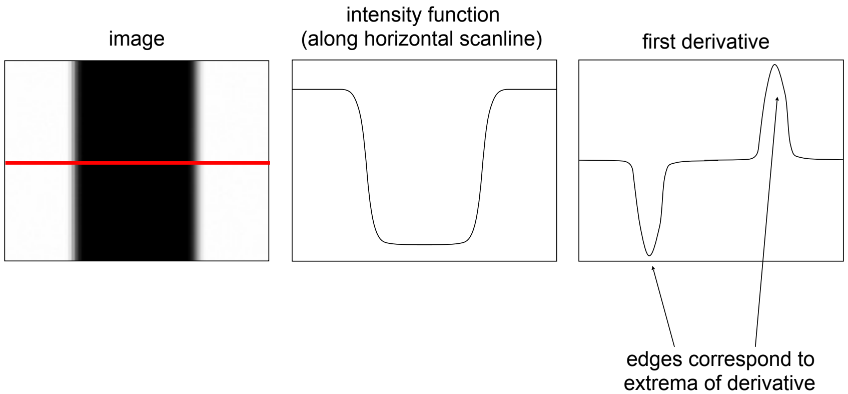
Edges are caused by a variety of factors:



Source: Steve Seitz

Edge detection

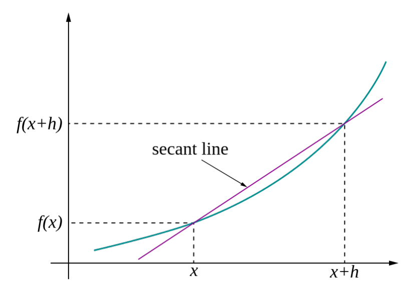
- An edge is a place of rapid change in the image intensity function

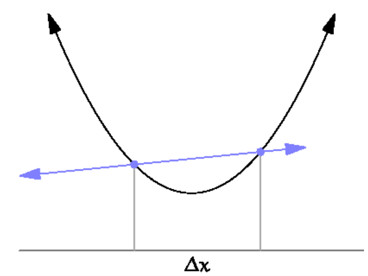


One dimensional derivatives

$y = f(x)$ Gradient $m = \frac{\Delta y}{\Delta x}$

$$m = \frac{f(x+h) - f(x)}{(x+h) - x} = \frac{f(x+h) - f(x)}{h}$$



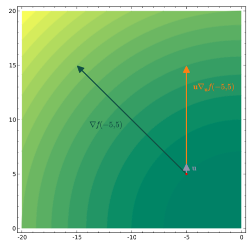


https://en.wikipedia.org/wiki/Derivative

Two dimensional derivatives

For 2D function f(**x**), one can compute a derivative for each direction **v**

$$\nabla_{\mathbf{v}} f(\mathbf{x}) = \lim_{h \to 0} \frac{f(\mathbf{x} + h\mathbf{v}) - f(\mathbf{x})}{h}.$$



Directional derivatives of the function along the axes are called partial derivatives. For example the partial derivative with respect to x is:

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

Source: K. Grauman

Partial derivatives with convolutions

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 the 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 one shows changes with respect to x?

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

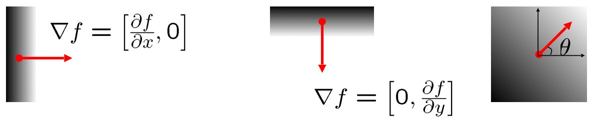
Roberts: $M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

Source: K. Grauman

Image gradient

The gradient of an image:

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



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

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

Edge detection example

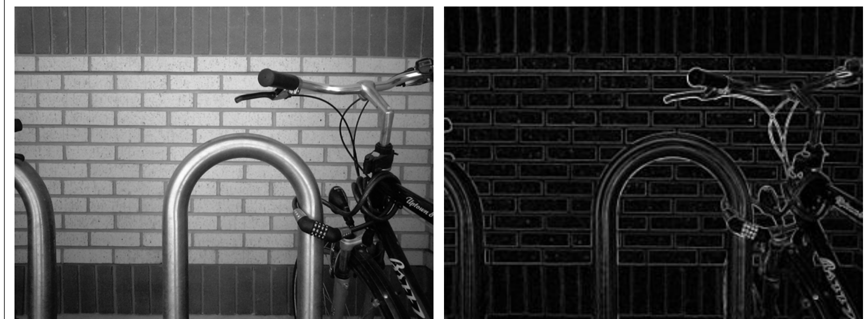


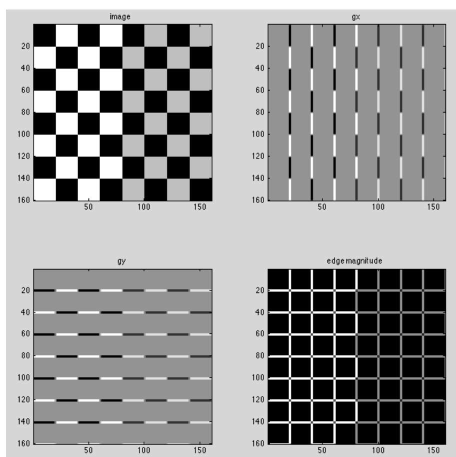
image　　　edge magnitude

$$\mathbf{G_x} = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G_y} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} * \mathbf{A}$$

https://en.wikipedia.org/wiki/Prewitt_operator

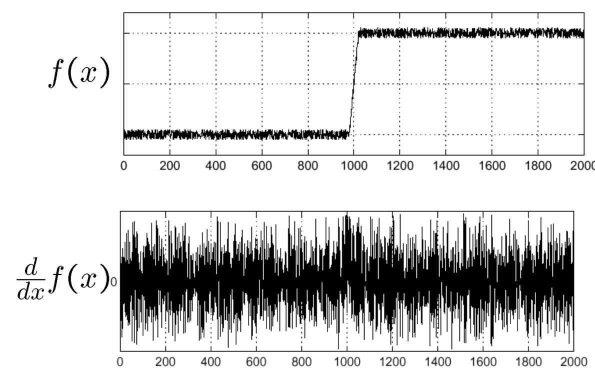
Edge detection in Matlab

```
im = checkerboard(20);

% Prewitt filters
fx = [-1 0 1;
      -1 0 1;
      -1 0 1];
fy = fx';

% Convolutions
gx = conv2(im, fx, 'same');
gy = conv2(im, fy, 'same');

% Display image
figure(1); clf;
subplot(2,2,1);
imagesc(im);
title('image');

subplot(2,2,2);
imagesc(gx); colormap gray
title('gx');

subplot(2,2,3);
imagesc(gy); colormap gray
title('gy');

subplot(2,2,4);

imagesc(gx.^2 + gy.^2); colormap gray;
title('edge magnitude');
```



Effects of noise

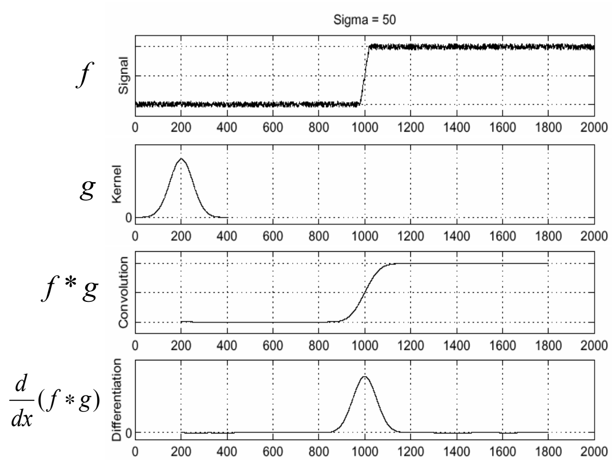
Consider a single row or column of the image



Where is the edge?

Source: S. Seitz

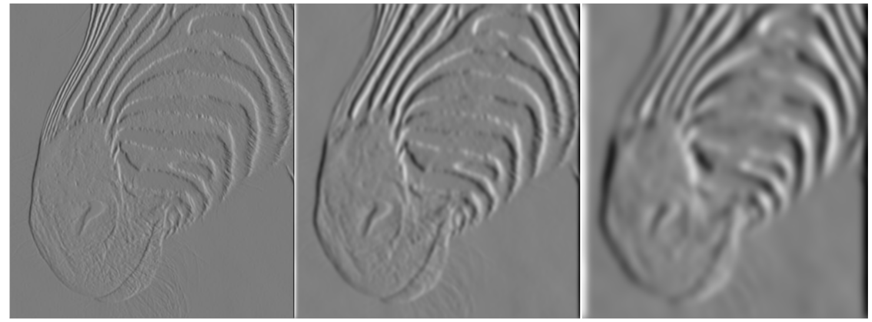
Solution: smooth first



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

Source: S. Seitz 33

Scale of smoothing

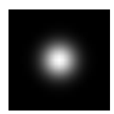


1 pixel　　3 pixels　　7 pixels

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

Source: D. Forsyth 34

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

- Prewitt filter
- 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

35