A. Basic Image Processing

Image Filters, Template matching using correlation

Instructor: Shaifali Parashar

Many Slides from Fei-Fei Li’s lectures at Standford

Images are functions

Different types of images

- Radiance images, where a pixel value corresponds to the radiance from some point in the scene in the direction of the camera.
- X-Ray, MRI, Ultrasound, Light Microscopy, Electron Microscopy

$f(x,y) = [0, 255]^M \quad R^2 \rightarrow R^M$

$x = [a,b]$, $y = [c,d]$

Can be transformed with other functions

$f: [a,b] \times [c,d] \rightarrow [0,255]$

Domain support

range

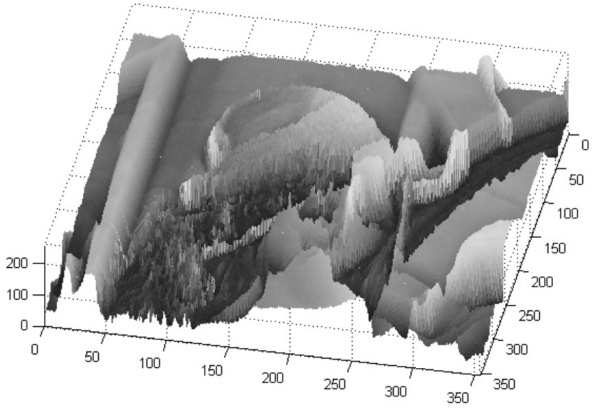


Image processing goals

- Image Compression
  - JPEG, JPEG2000, MPEG..

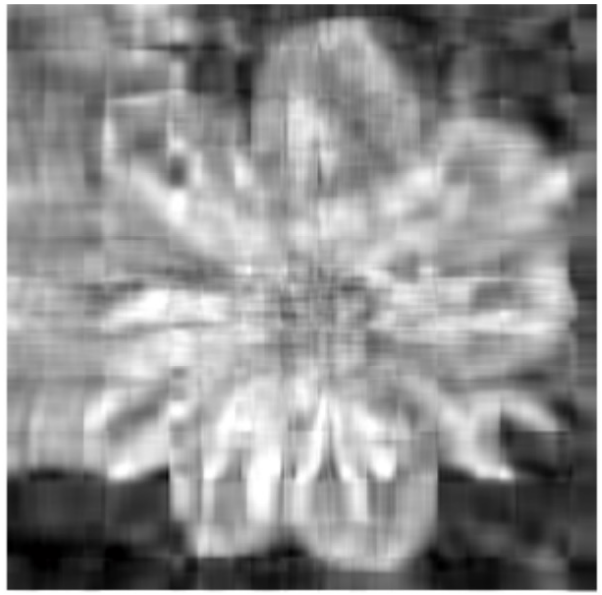


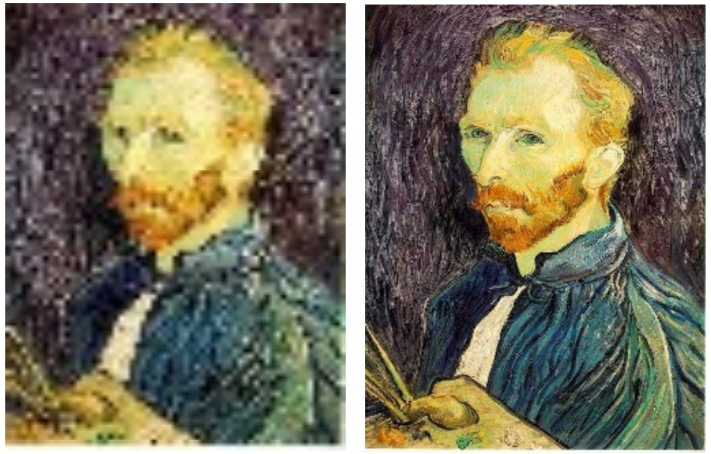
Image processing goals

- Image Restoration
  - denoising
  - deblurring (super-resolution)
  - In-painting

De-noising

Salt and pepper noise

Super-resolution

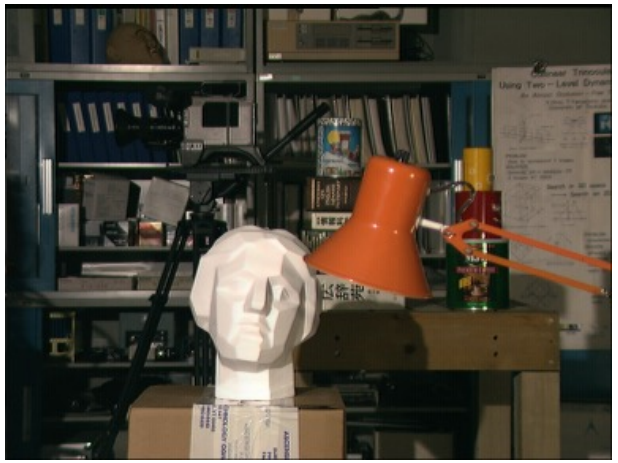


In-painting

Bertamio et al

Image processing goals

- Computing Field Properties
  - orientation
  - optical flow
  - …



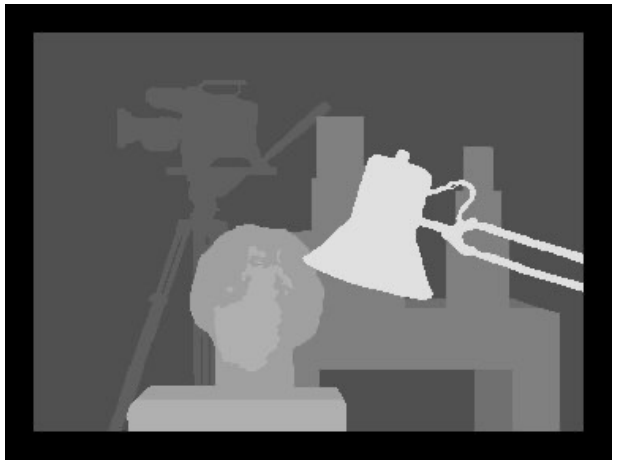
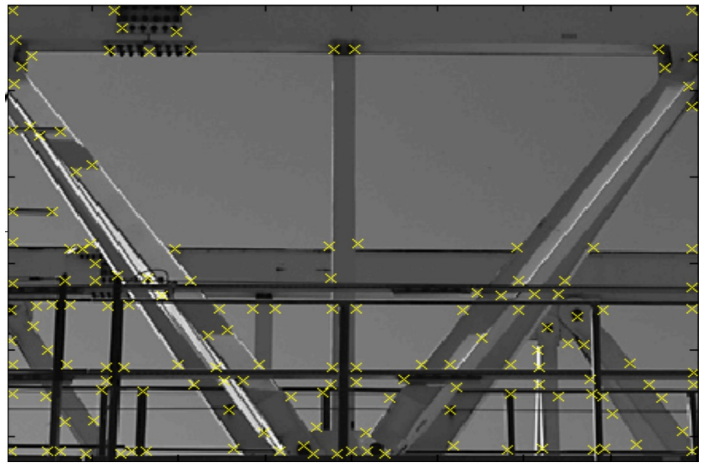


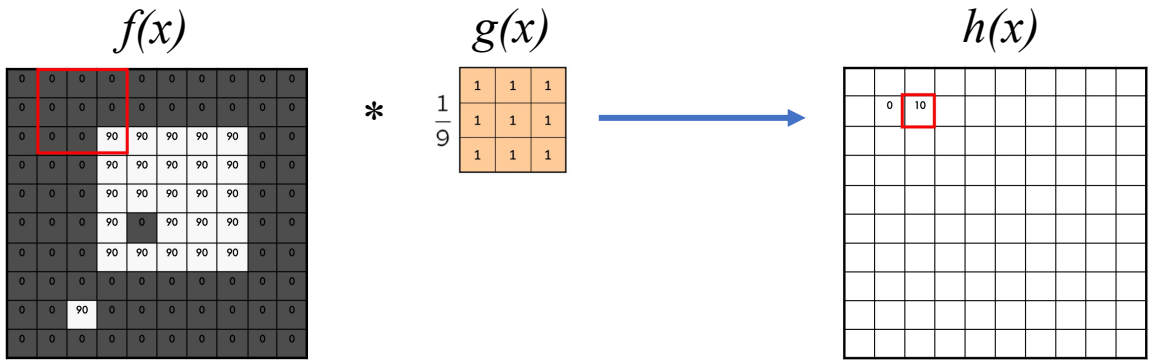
Image processing goals

- Locating Structural Features
  - corners
  - Edges
  - …



Moving average filter

- Replaces each pixel with an average of its neighborhood
- Achieves smoothing, reduces sharpness



In case you want to preserve edges, use gaussian filter: consider g(x) to be a gaussian

Some noises are better removed using frequency domain

Thresholding filter

- Image segmentation by simple thresholding

$$h[n,m] = \begin{cases} 255, & f[n,m] > 100 \\ 0, & \text{otherwise.} \end{cases}$$

C&C: Convolution and Correlation

2-D case

Correlation

Convolution

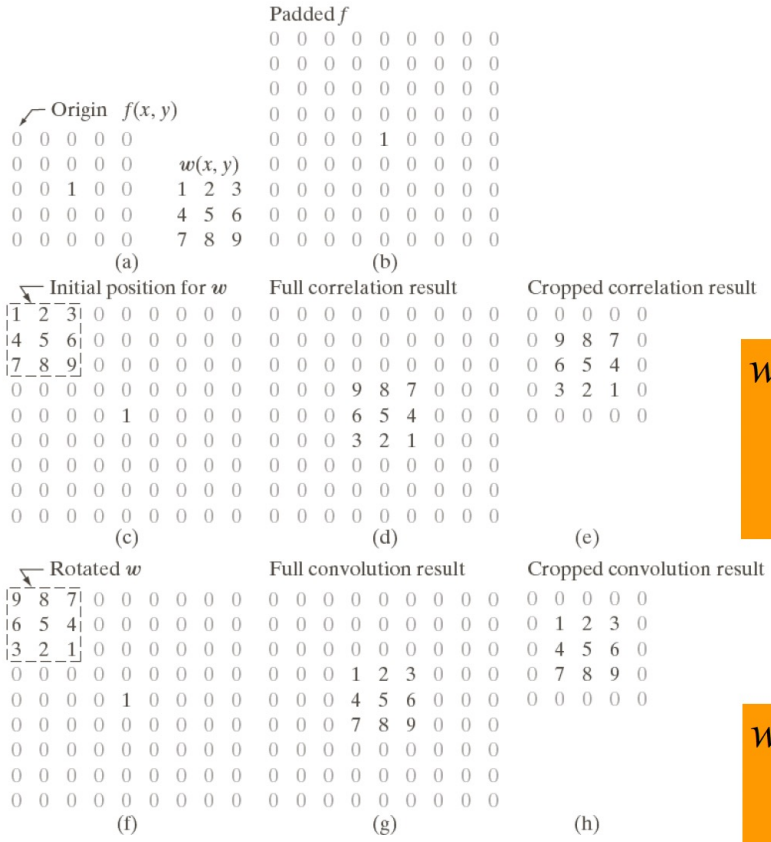


FIGURE 3.30 Correlation (middle row) and convolution (last row) of a 2-D filter with a 2-D discrete, unit impulse. The 0s are shown in gray to simplify visual analysis.

$$w(x,y) \otimes f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$$

$$w(x,y) * f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t)$$

C&C: Convolution & Correlation

Convolution: generic name of kernel/filter-based processing to design sharpening, blurring filters etc

Discrete convolution:

1. Take a filter. Flip by origin such that g(x,y)=g(-x,-y)
2. Shift the folded results by x_0, y_0 to form $g(x_0 - x, y_0 - y)$
3. Multiply by signal $f(x_0, y_0)$ and sum all quantities

Correlation = convolution without a flip

Cross-correlation: between 2 signals (f and g)

Auto-correlation: between same signal (f & f)

Convolution in 2D

Convolution: generic name of kernel/filter-based processing to design sharpening, blurring filters etc

Represented by *

$$\left(\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} * \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}\right)[2,2] = (i \cdot 1) + (h \cdot 2) + (g \cdot 3) + (f \cdot 4) + (e \cdot 5) + (d \cdot 6) + (c \cdot 7) + (b \cdot 8) + (a \cdot 9).$$

2D Convolution as a sharpening filter

Original

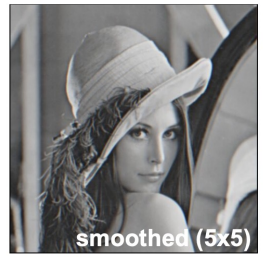
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = ?$$

(Note that filter sums to 1)

"details of the image"

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

–

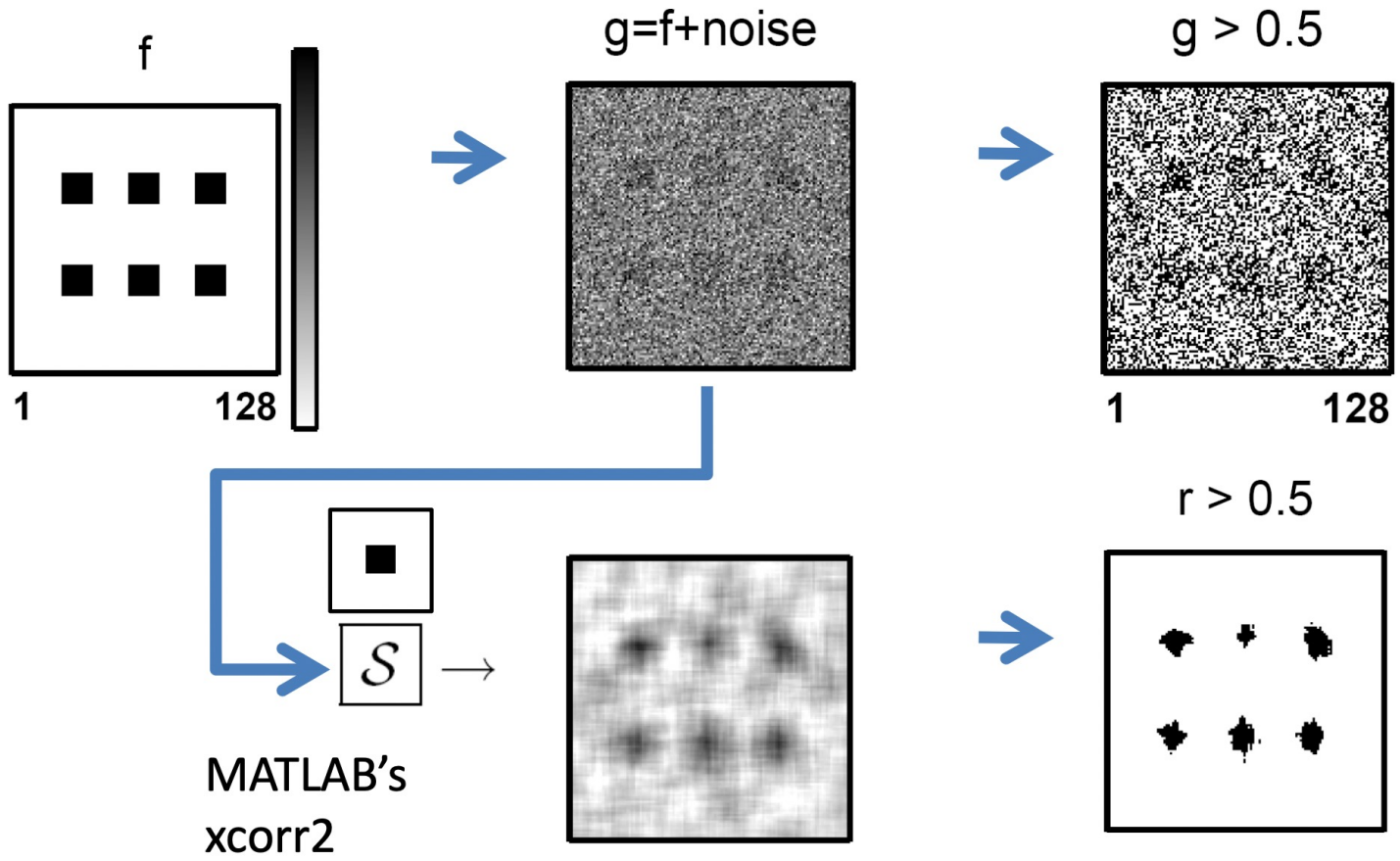


=

+ a

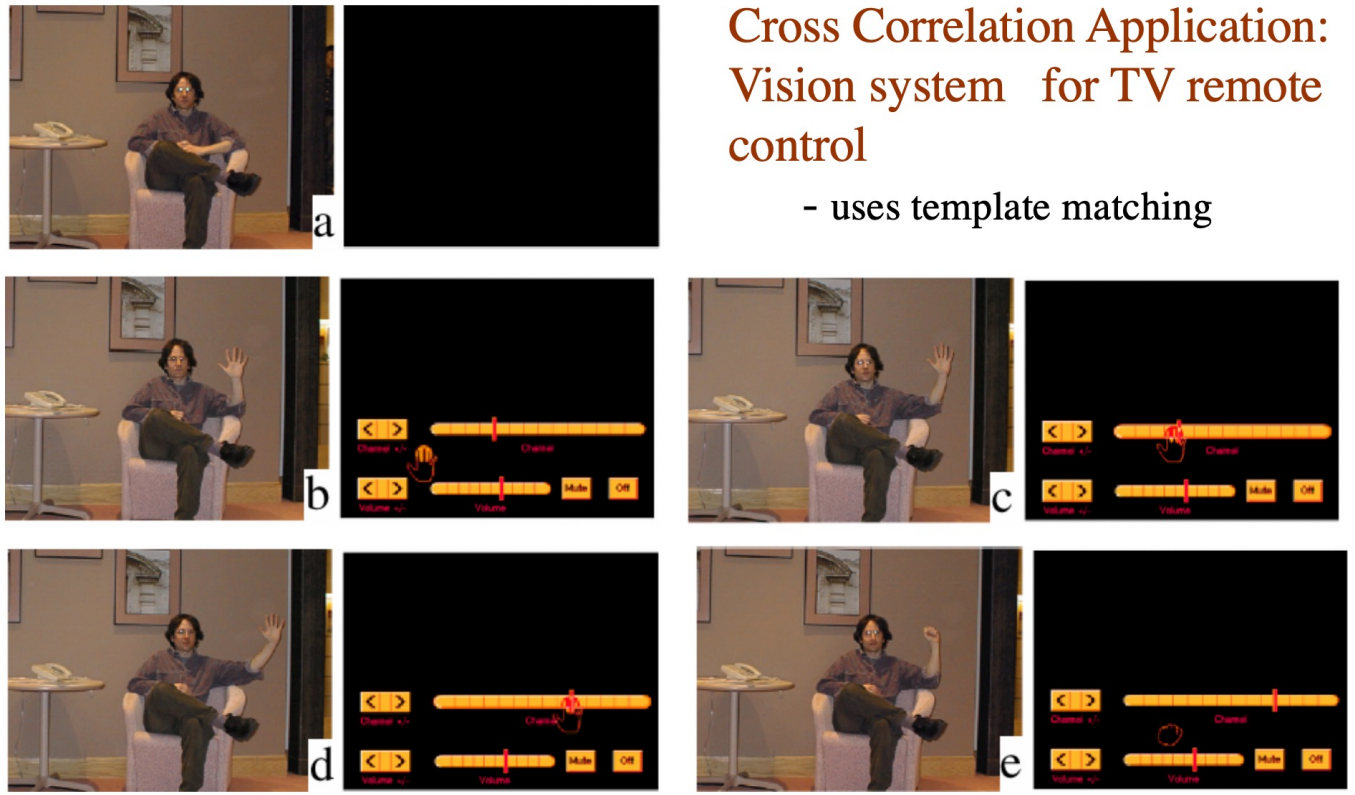
=

Correlation and Template Matching



The correlation result reaches a maximum at the time when the two signals match best.

Cross-correlation



Cross Correlation Application: Vision system for TV remote control

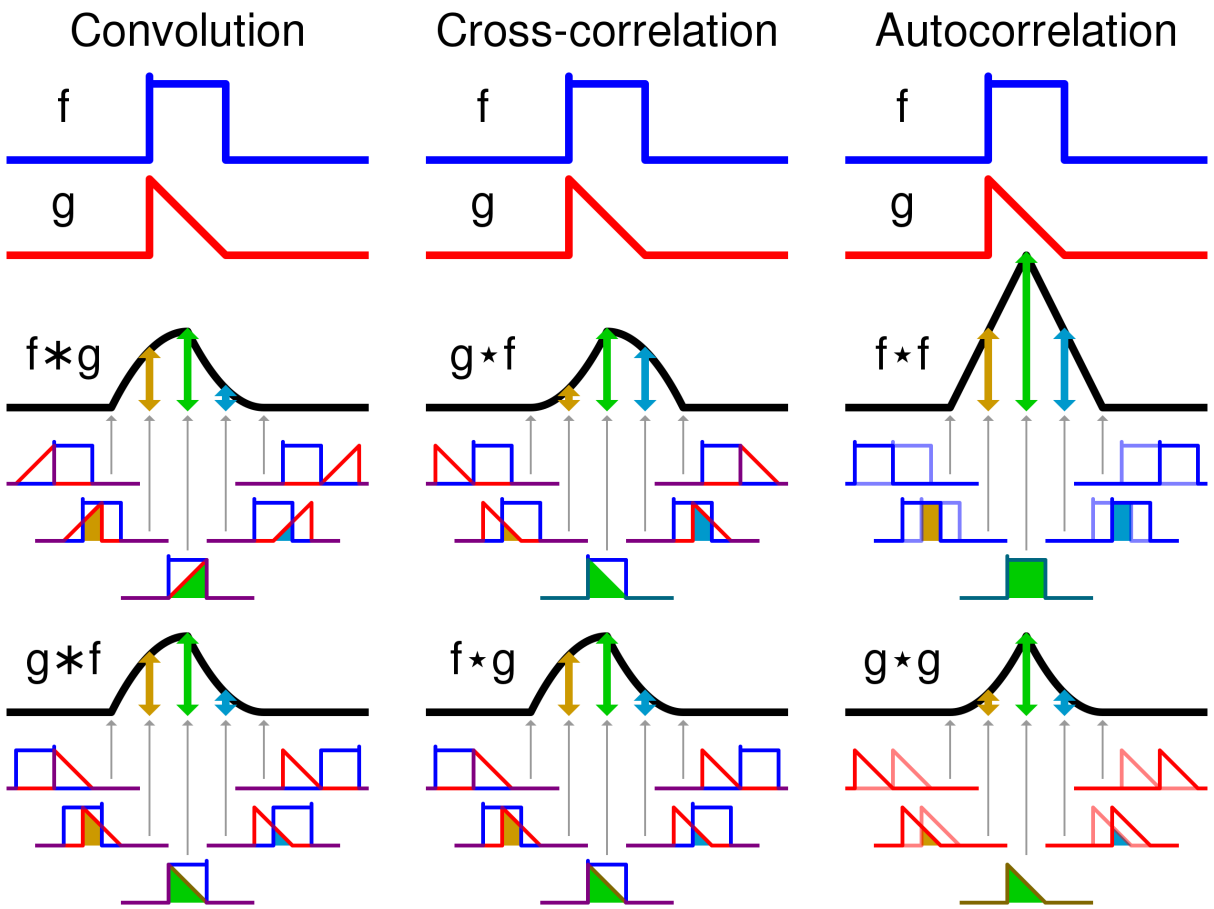
- uses template matching

Figure from "Computer Vision for Interactive Computer Graphics," W.Freeman et al, IEEE Computer Graphics and Applications, 1998 copyright 1998, IEEE

Convolution and Cross-correlation

- A convolution is a filtering operation that expresses the amount of overlap of one function as it is shifted over another function.
- Correlation compares the similarity of two sets of data. It computes a measure of similarity of two input signals as they are shifted by one another.

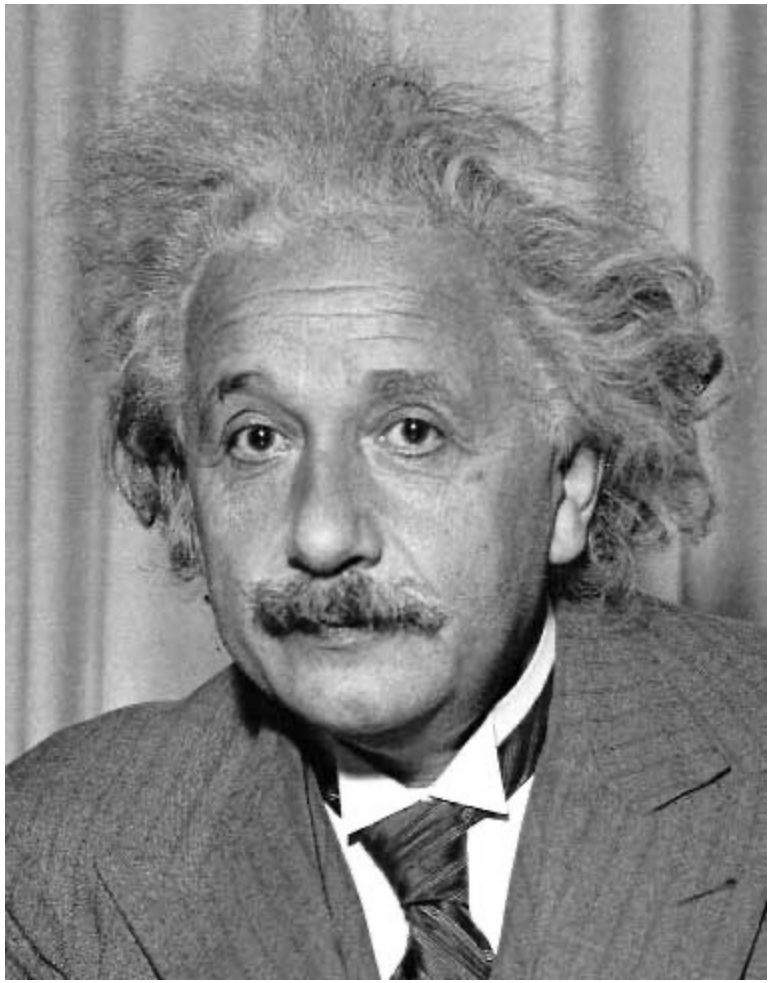
Cross-correlation and Auto-correlation



Template matching

- Goal: find  in image

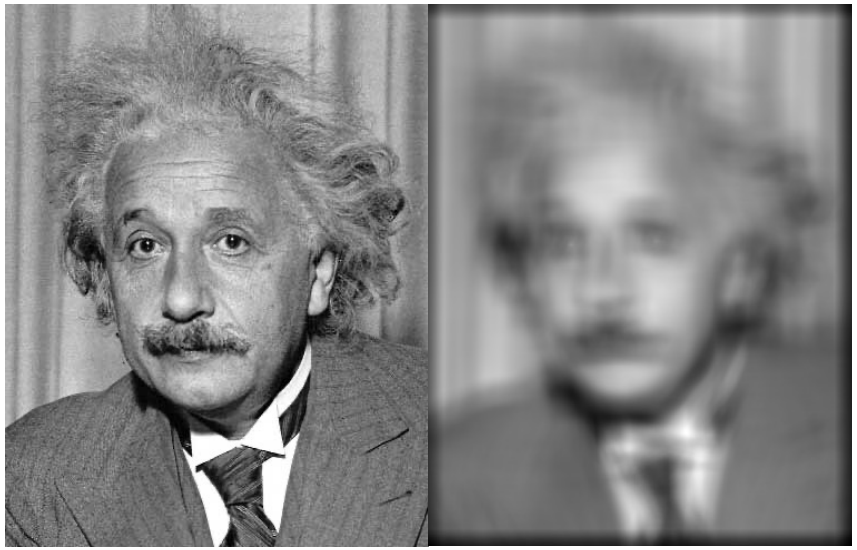
- Main challenge: What is a good similarity or distance measure between two patches?
  - Correlation
  - Zero-mean correlation
  - Sum Square Difference
  - Normalized Cross Correlation



Template matching

Correlation

$$h[m,n] = \sum_{k,l} g[k,l]\, f[m+k, n+l]$$

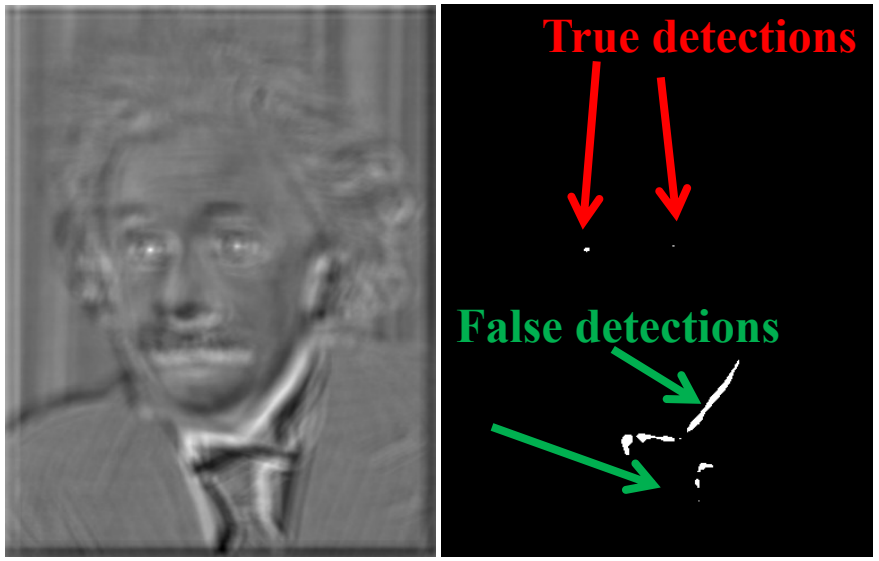


Input Filtered Image

f = image
g = filter

Correlation with zero mean

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

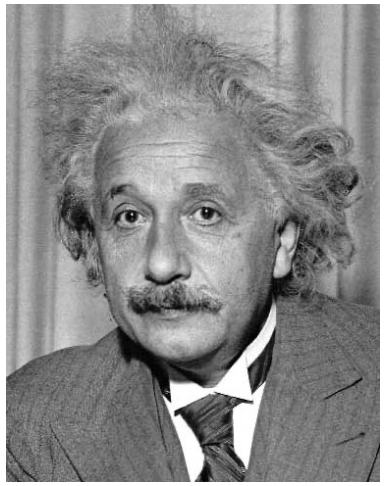


Filtered Image (scaled) Thresholded Image

Template matching

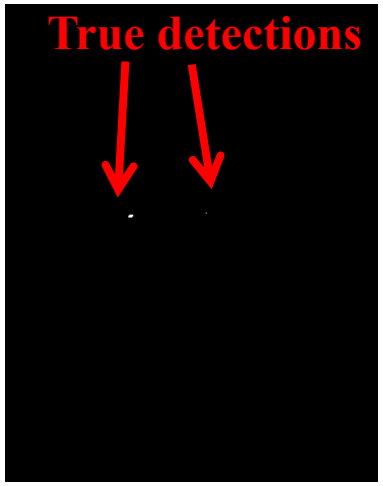
Sum of squared differences (SSD)

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k,n+l])^2$$



Input

1- sqrt(SSD)

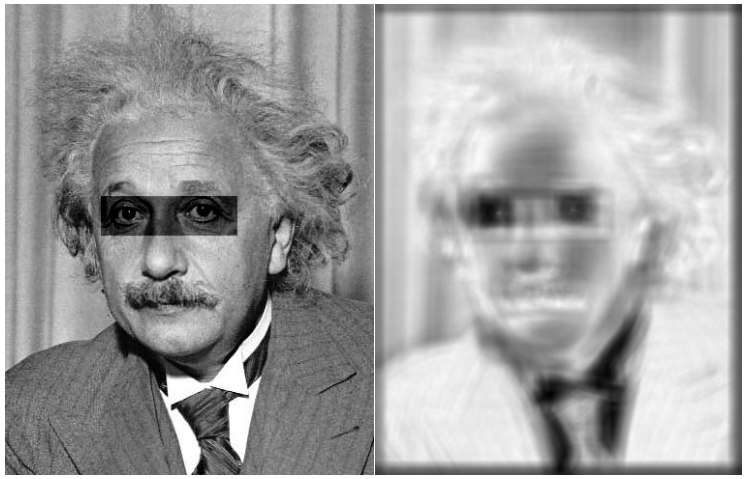


Thresholded Image

f = image
g = filter

Problem with SSD:

Sensitive to Image intensity



Template matching

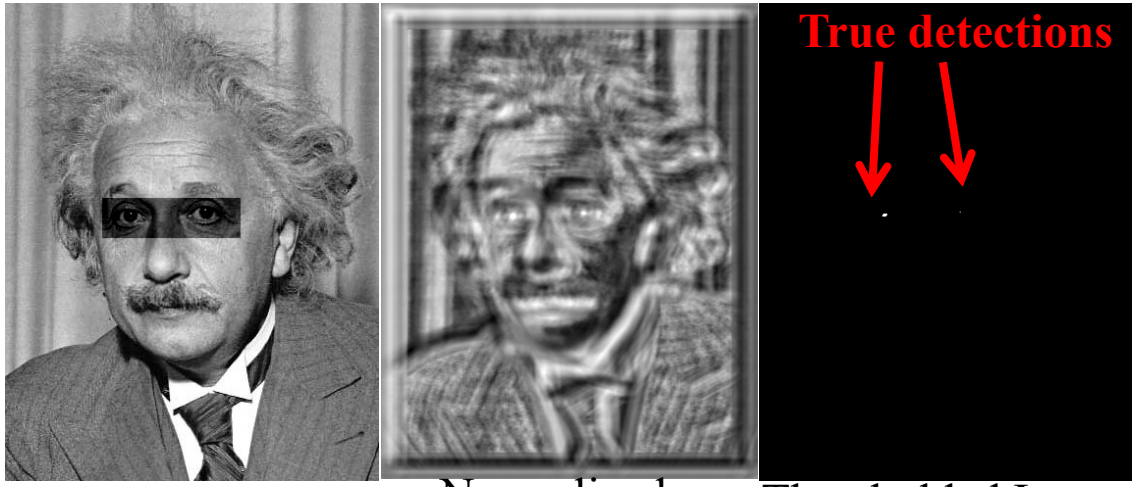
Normalized cross-correlation

$$h[m,n] = \frac{\sum_{k,l}(g[k,l]-\bar{g})(f[m+k,n+l]-\bar{f}_{m,n})}{\left(\sum_{k,l}(g[k,l]-\bar{g})^2 \sum_{k,l}(f[m+k,n+l]-\bar{f}_{m,n})^2\right)^{0.5}}$$

mean template

mean image patch

Input | Normalized X-Correlation | Thresholded Image

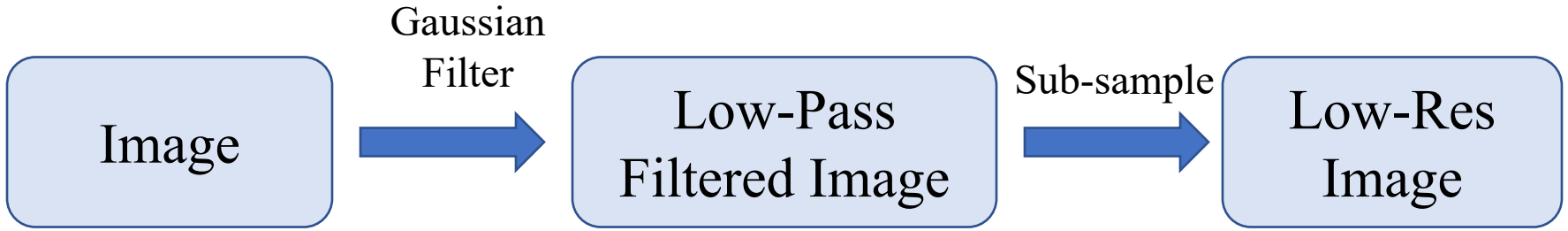


Input | Normalized X-Correlation | Thresholded Image

Pick the correlation according to the problem; the computational complexity increases with accuracy

Template matching: handling size variations

What if we want to find larger or smaller eyes? Solution: Image Pyramids

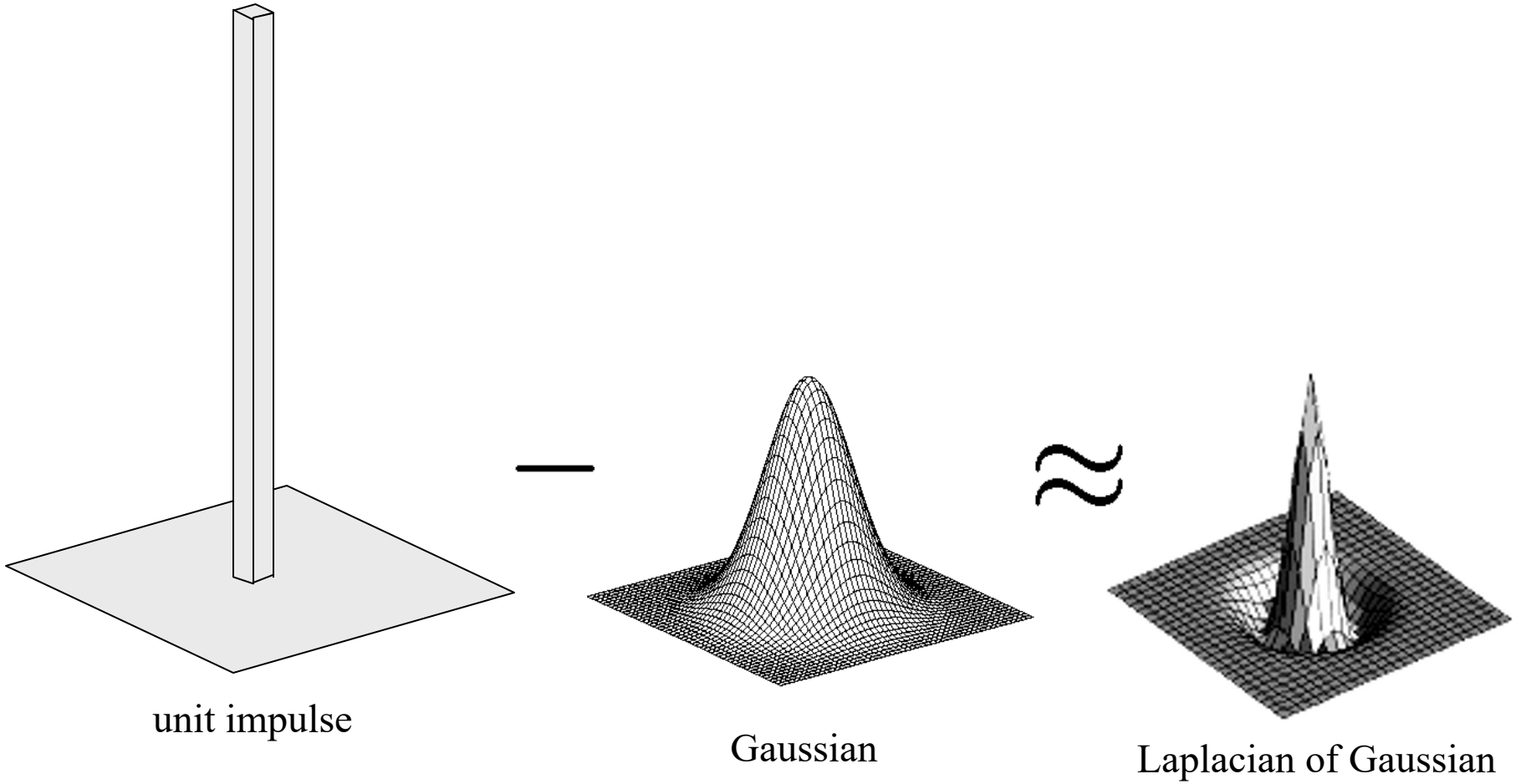


Template Matching with Image Pyramids

Input: Image, Template

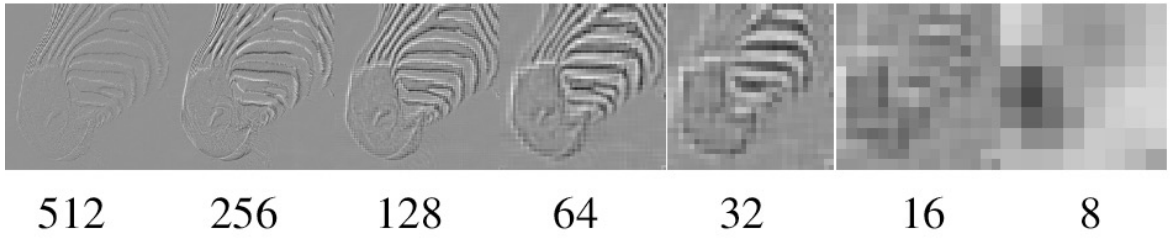
1. Match template at current scale
2. Downsample image (In practice, scale step of 1.1 to 1.2)
3. Repeat 1-2 until image is very small
4. Take responses above some threshold, perhaps with non-maxima suppression

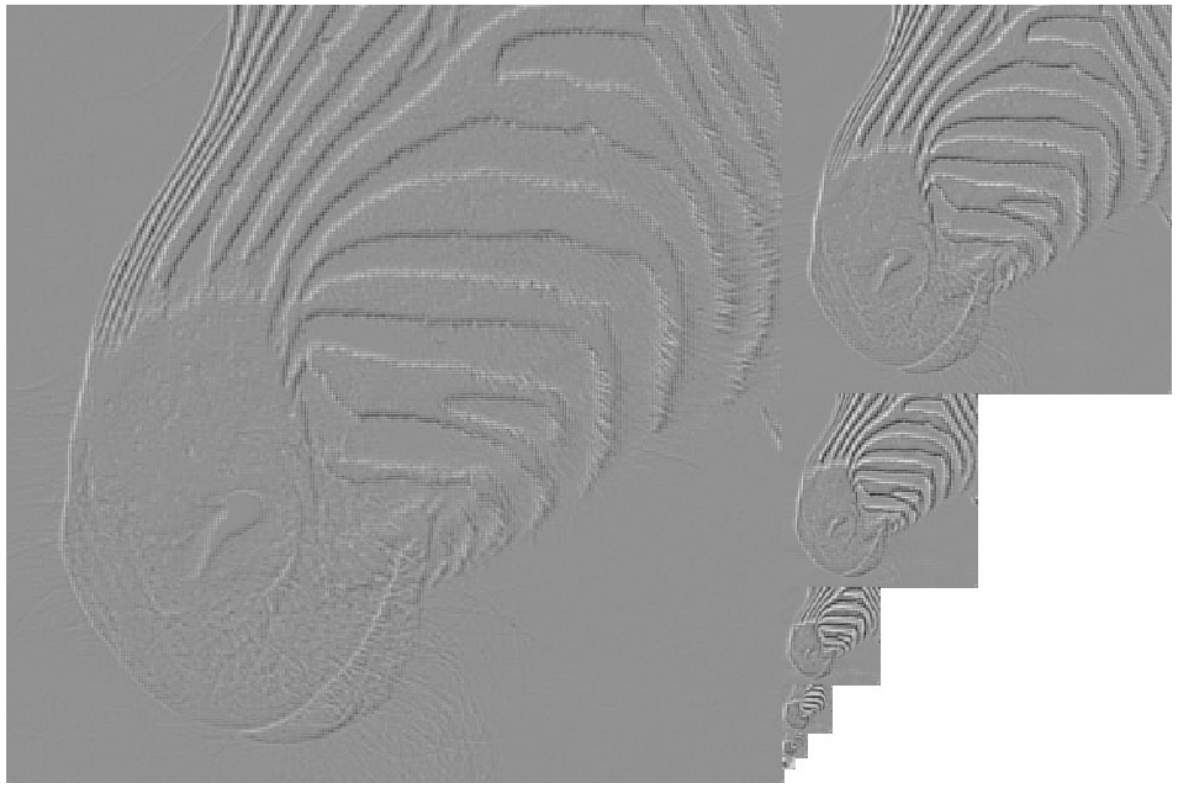
Laplacian filter



Source: Lazebnik

Laplacian pyramid





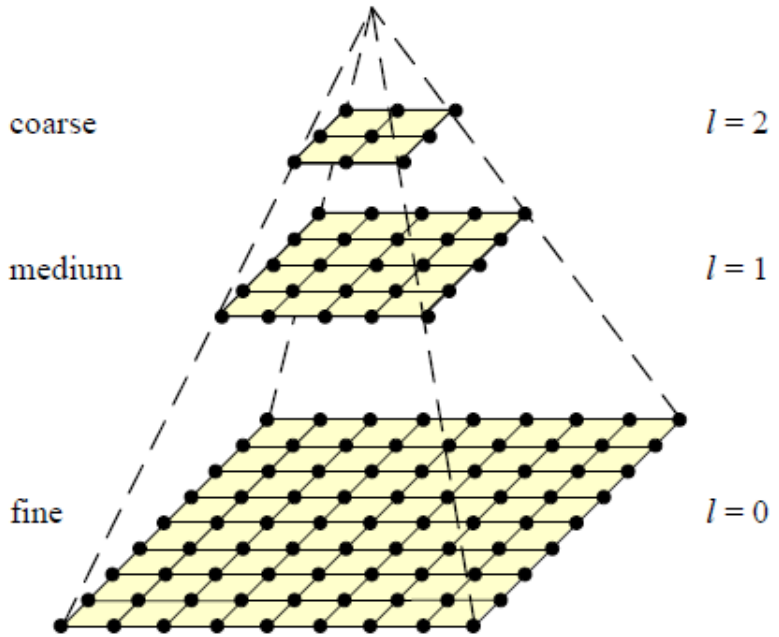
Source: Forsyth

Coarse-to-fine Image Registration

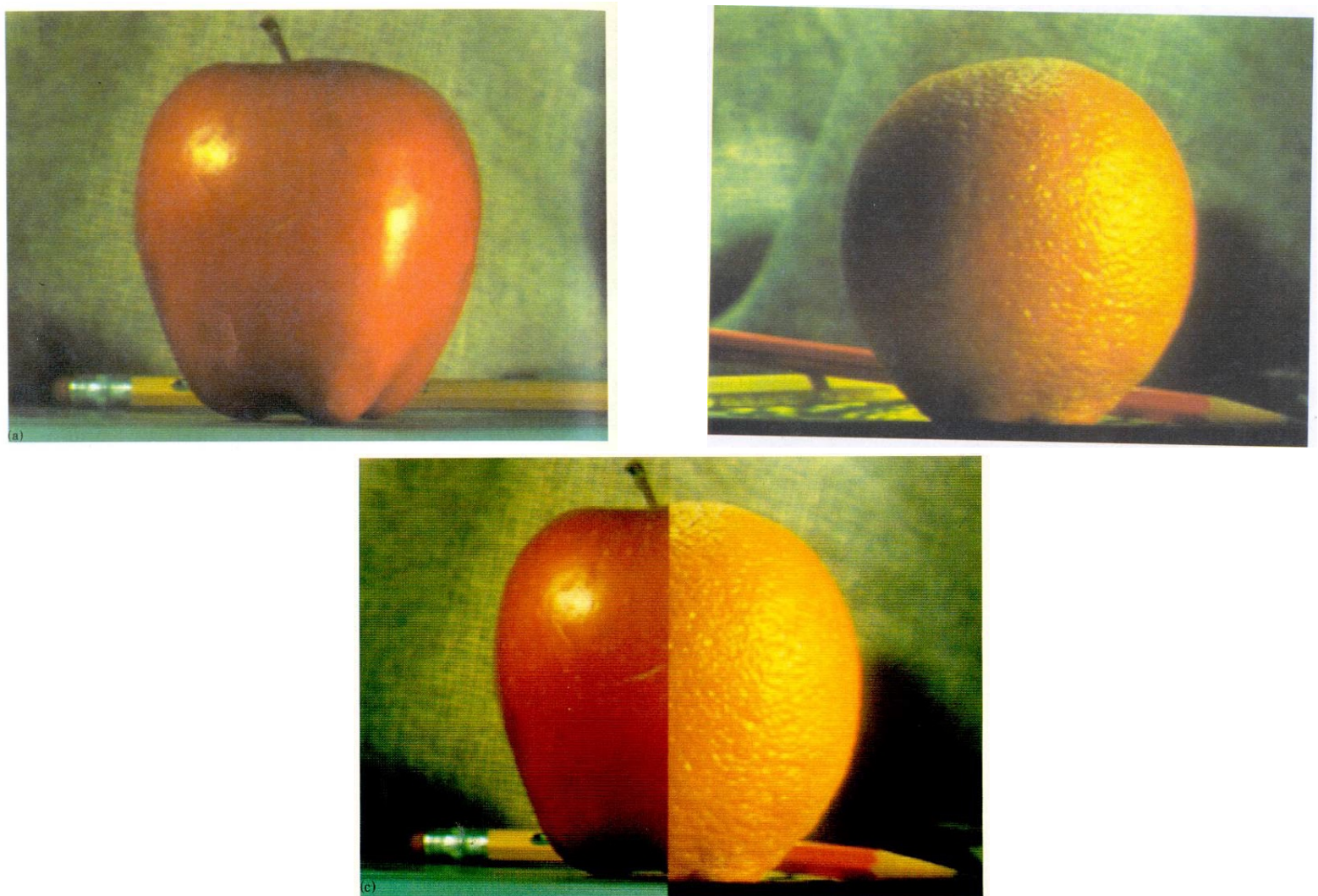
1. Compute Gaussian pyramid
2. Align with coarse pyramid
3. Successively align with finer pyramids
   - Search smaller range

Why is this faster?

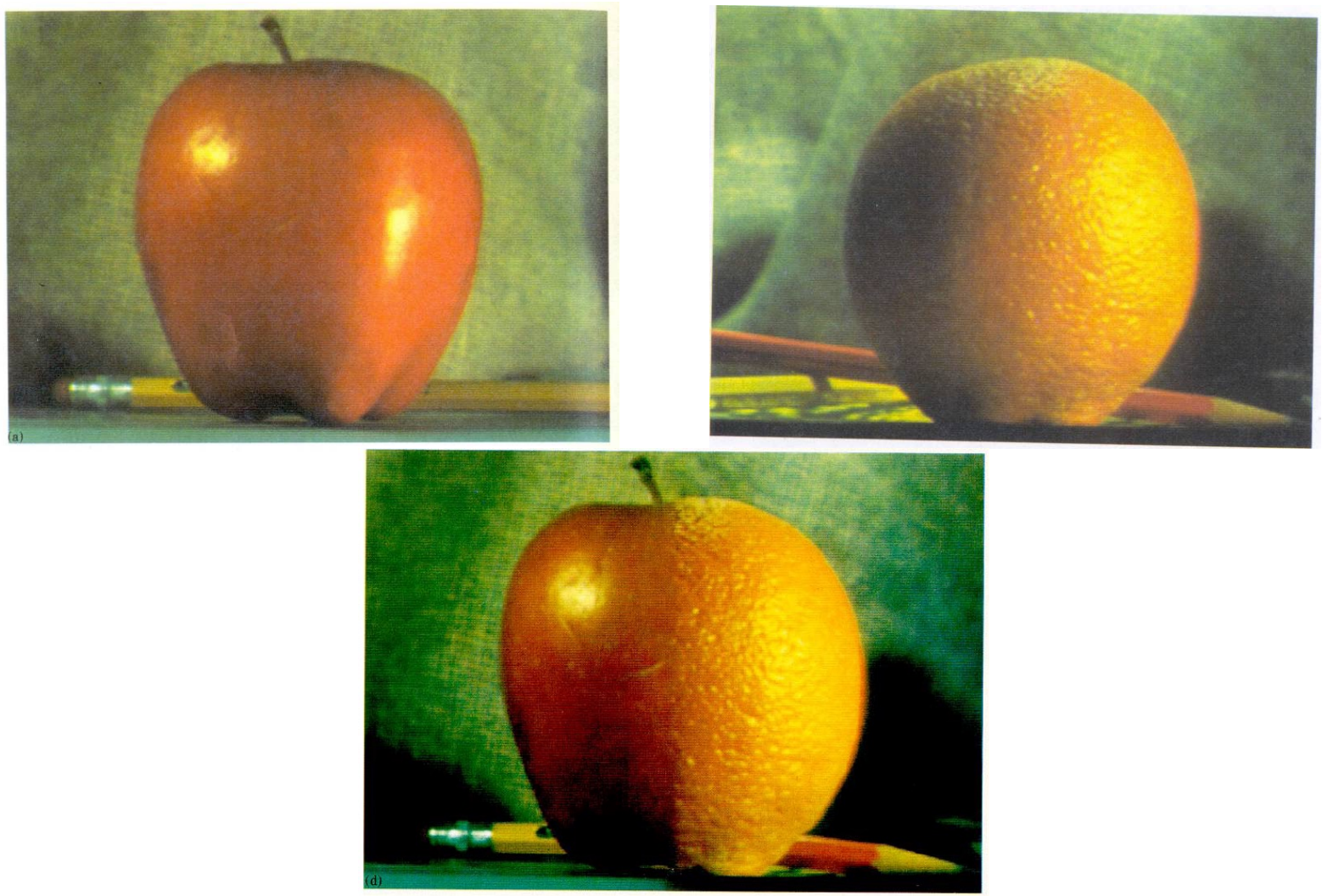
Are we guaranteed to get the same result?



Applications: Pyramid Blending

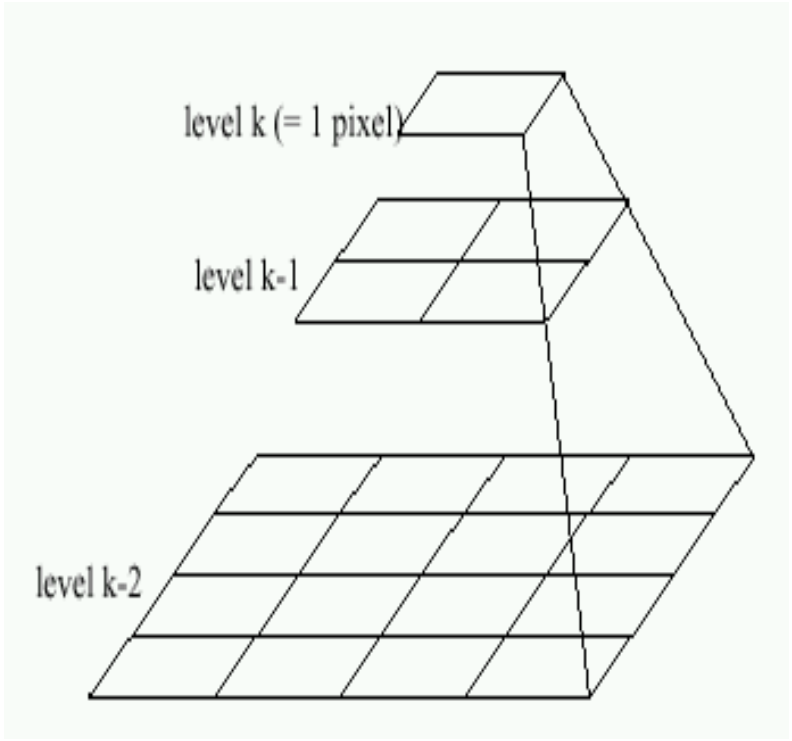


Applications: Pyramid Blending

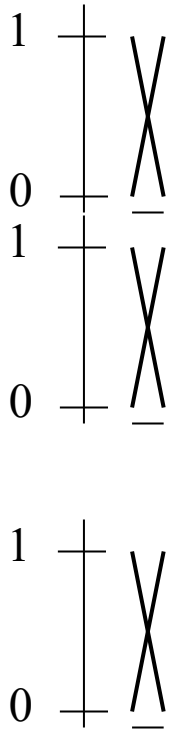


Pyramid Blending

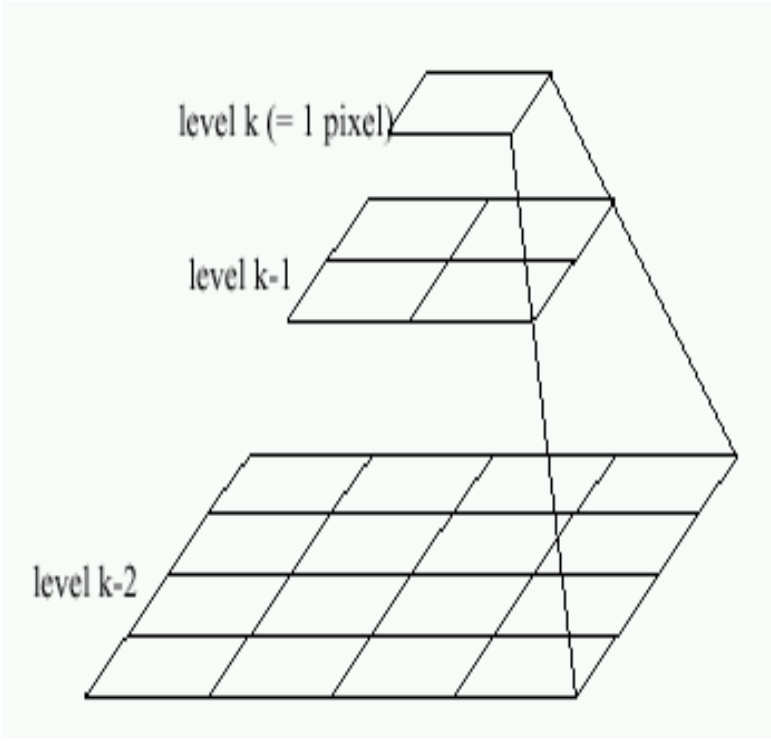
- At low frequencies, blend slowly
- At high frequencies, blend quickly



Left pyramid



blend



Right pyramid

Image representation

- Pixels:
  - great for spatial resolution, poor access to frequency
- Fourier transform:
  - great for frequency, not for spatial info
- Pyramids/filter banks:
  - balance between spatial and frequency information