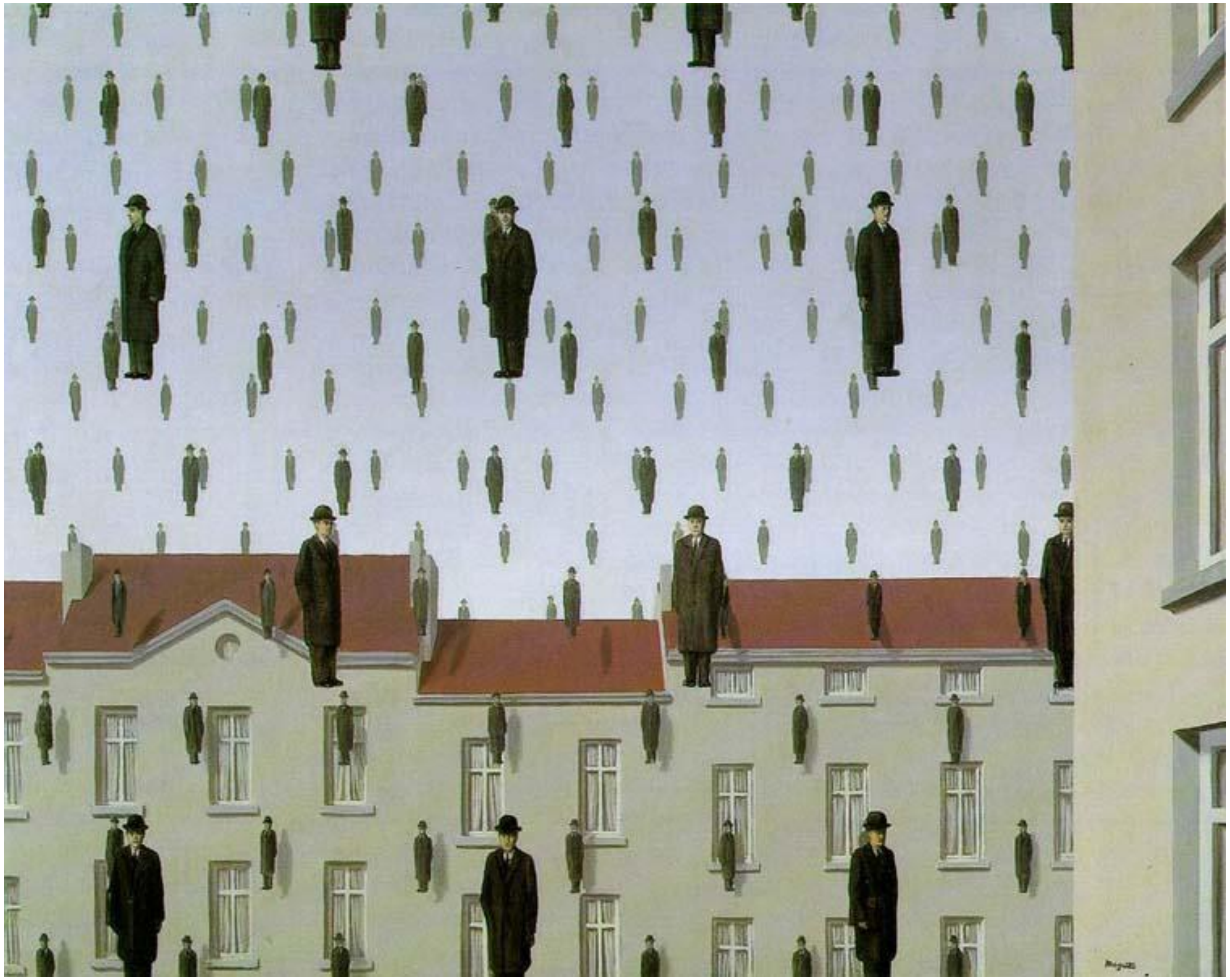
09/19/11

Templates, Image Pyramids, and Filter Banks



Computer Vision

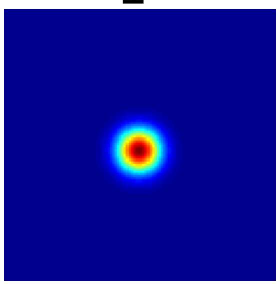
James Hays, Brown

Slides: Hoiem and others

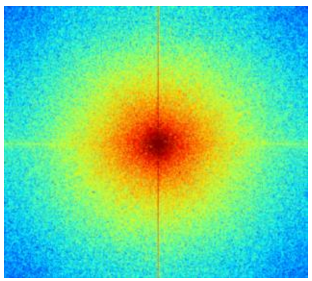
Review

1. Match the spatial domain image to the Fourier magnitude image

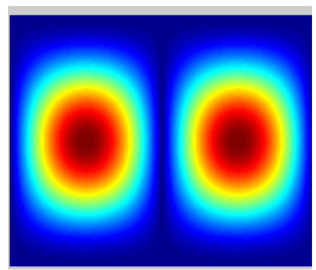
1



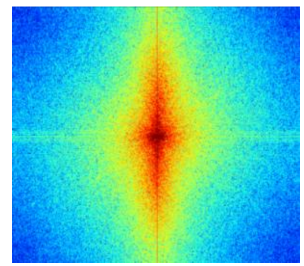
2



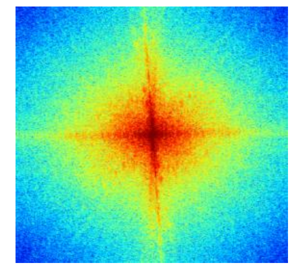
3



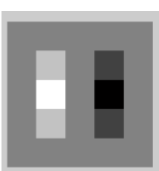
4



5

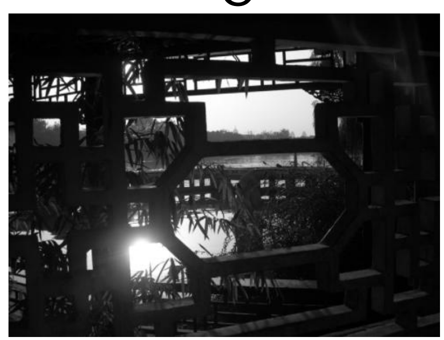


A

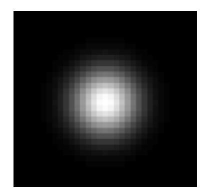


B

C



D



E

Slide: Hoiem

Reminder

- Project 1 due in one week

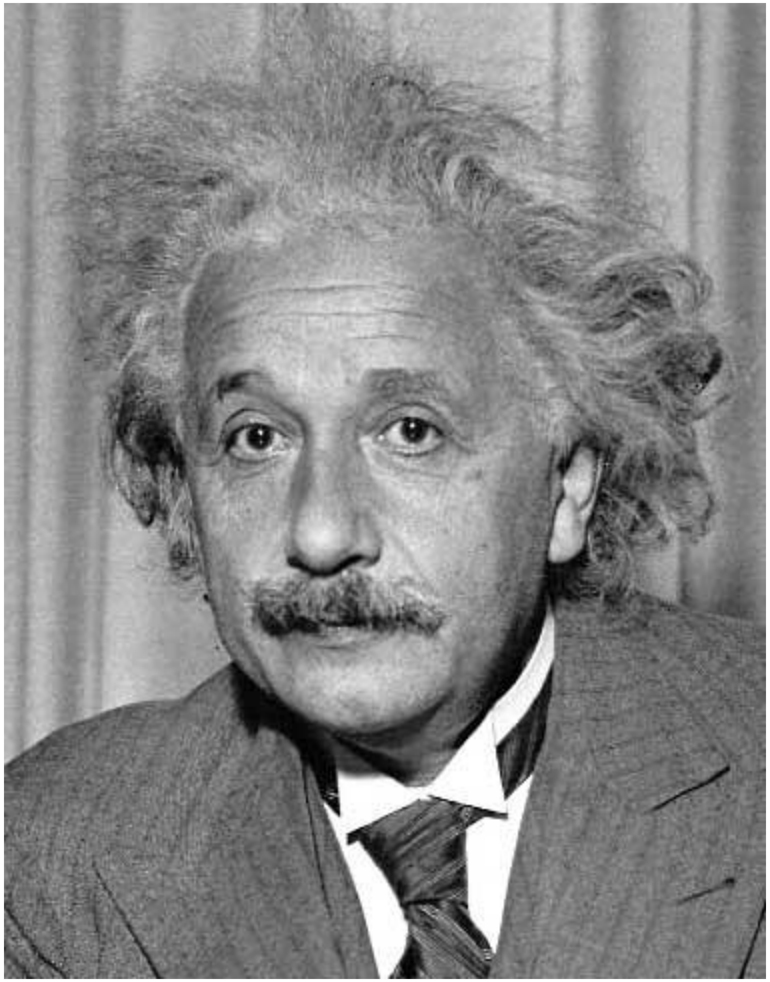
Today’s class

- Template matching
- Image Pyramids
- Filter banks and texture
- Denoising, Compression

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



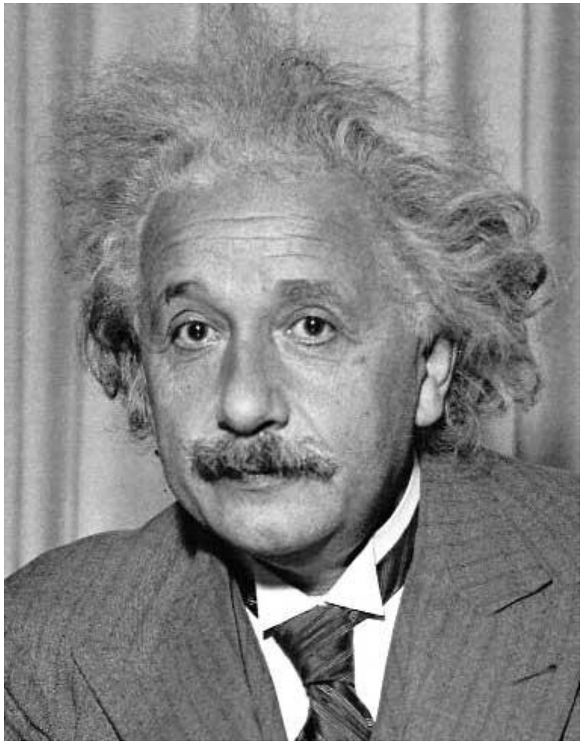
Slide: Hoiem

Matching with filters

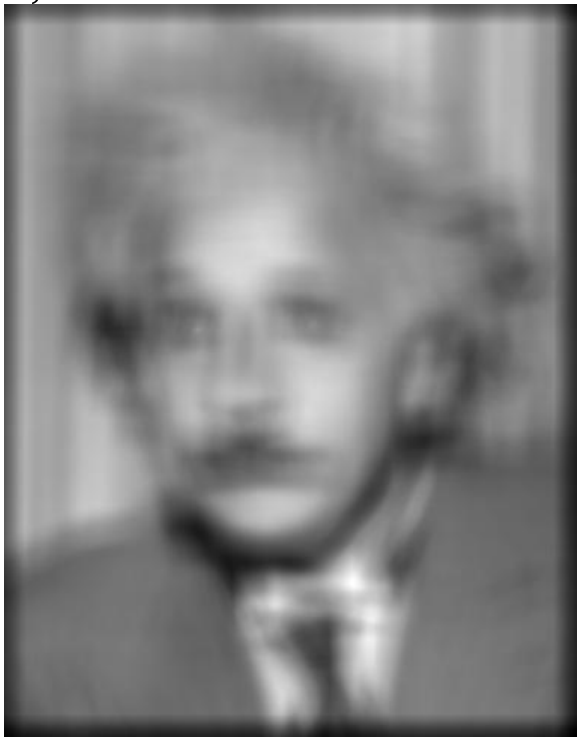
- Goal: find  in image
- Method 0: filter the image with eye patch

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

f = image
g = filter



Input



Filtered Image

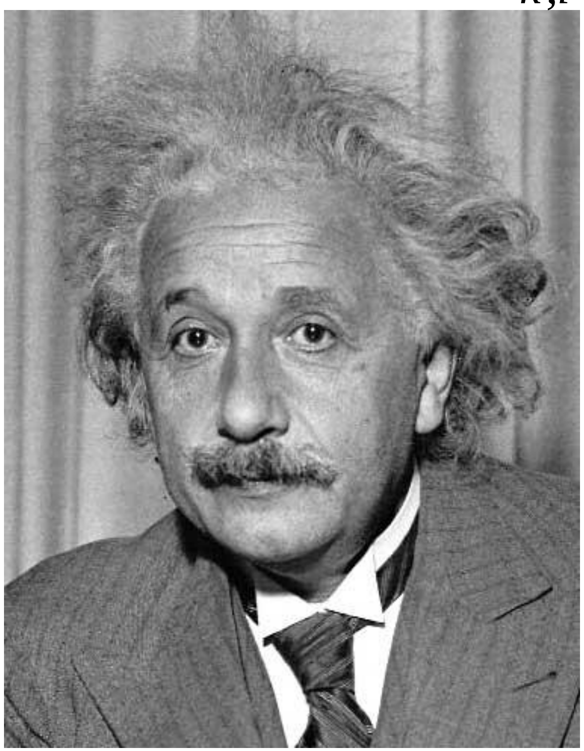
What went wrong?

Matching with filters

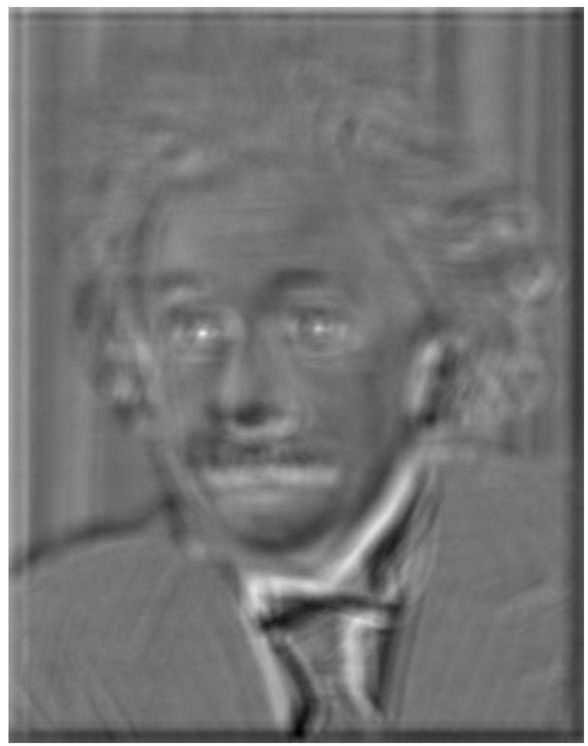
- Goal: find  in image
- Method 1: filter the image with zero-mean eye

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

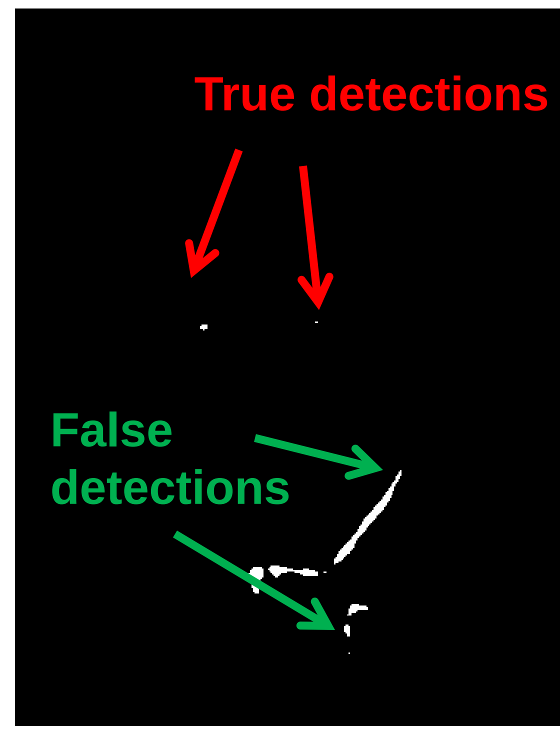
mean of f



Input



Filtered Image (scaled)



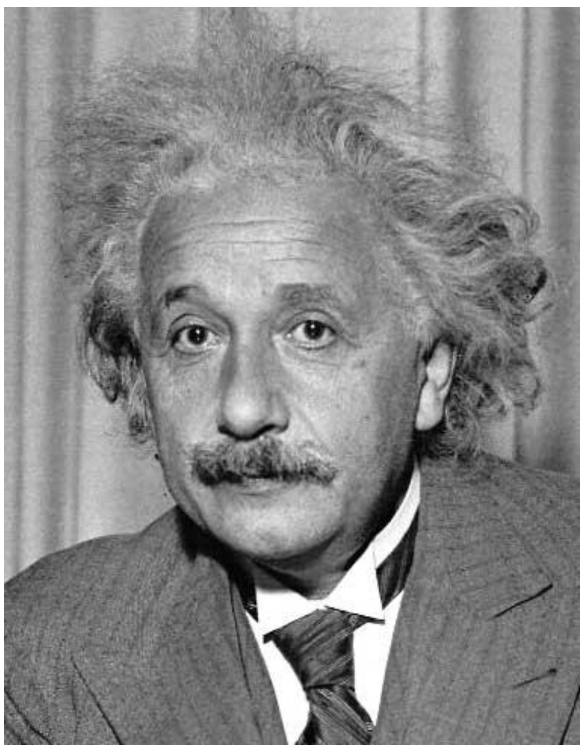
Thresholded Image

Slide: Hoiem

Matching with filters

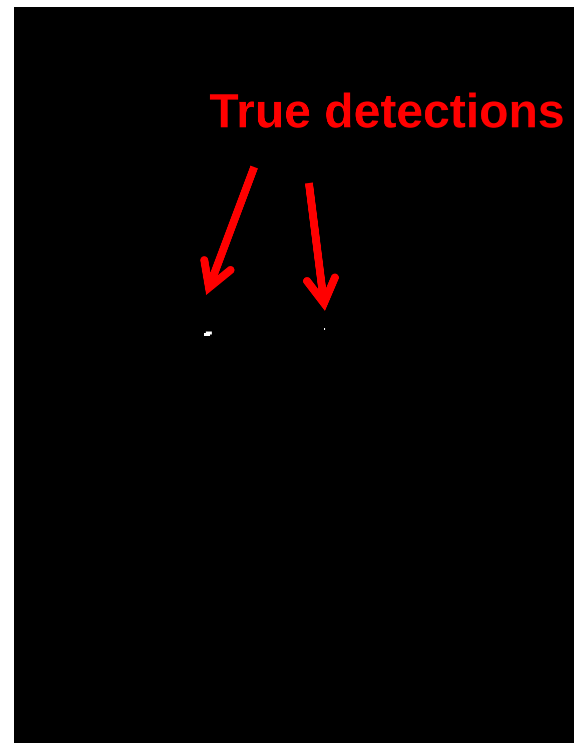
- Goal: find  in image
- Method 2: SSD

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



Input

1- sqrt(SSD)



Thresholded Image

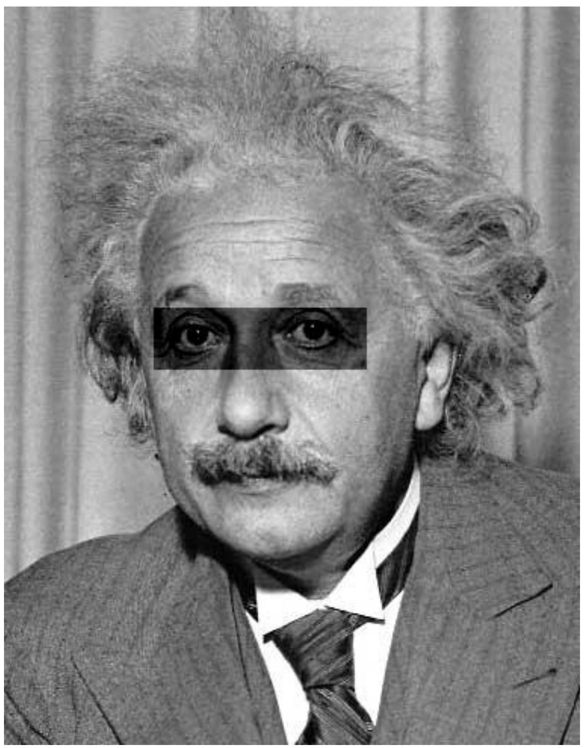
Slide: Hoiem

Matching with filters

- Goal: find  in image
- Method 2: SSD

What's the potential downside of SSD?

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



Input

1- sqrt(SSD)

Slide: Hoiem

Matching with filters

- Goal: find  in image
- Method 3: Normalized cross-correlation

mean template

mean image patch

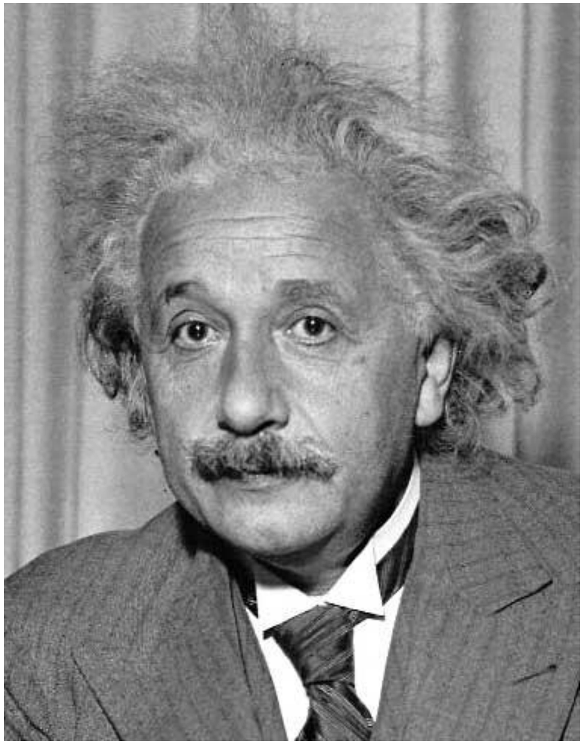
$$h[m,n] = \frac{\sum_{k,l}(g[k,l]-\bar{g})(f[m-k,n-l]-\bar{f}_{m,n})}{\left(\sum_{k,l}(g[k,l]-\bar{g})^2 \sum_{k,l}(f[m-k,n-l]-\bar{f}_{m,n})^2\right)^{0.5}}$$

Matlab: `normxcorr2(template, im)`

Slide: Hoiem

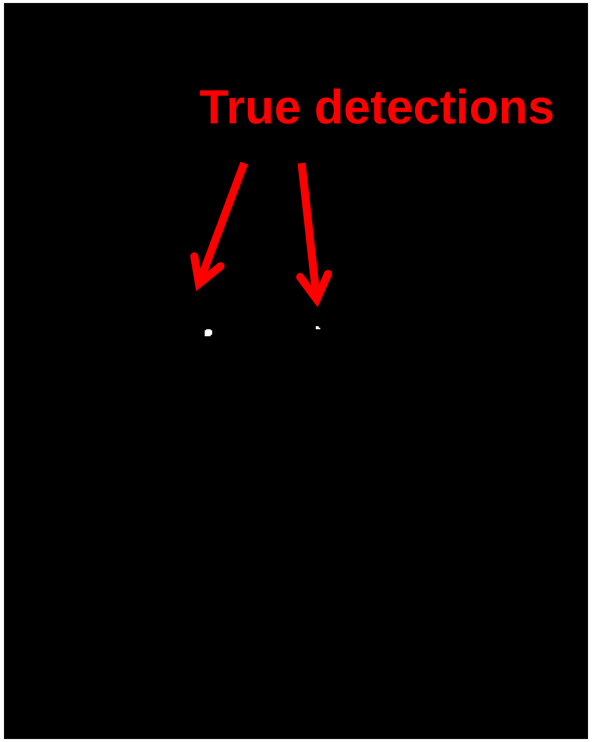
Matching with filters

- Goal: find  in image
- Method 3: Normalized cross-correlation



Input

Normalized X-Correlation

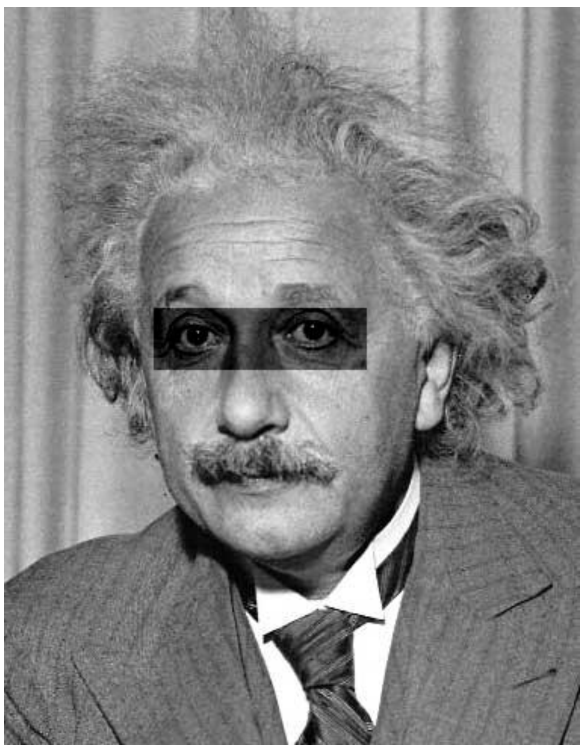


Thresholded Image

Slide: Hoiem

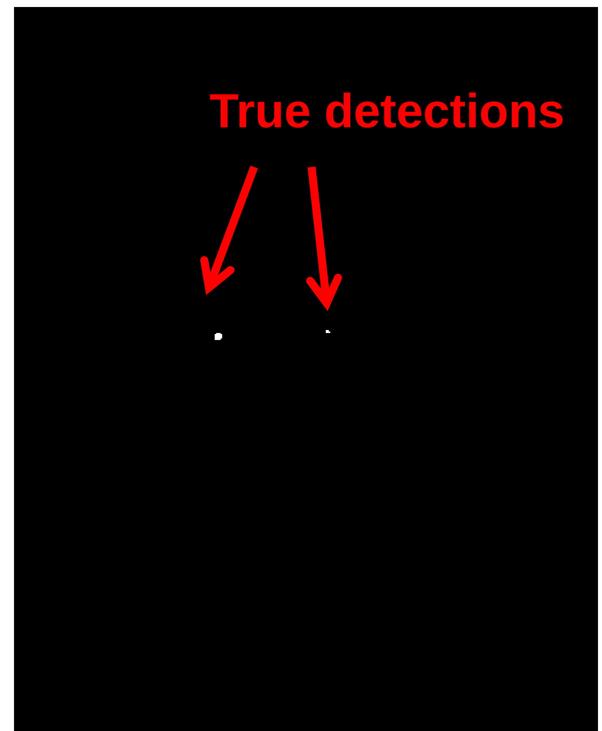
Matching with filters

- Goal: find  in image
- Method 3: Normalized cross-correlation



Input

Normalized X-Correlation



Thresholded Image

Q: What is the best method to use?

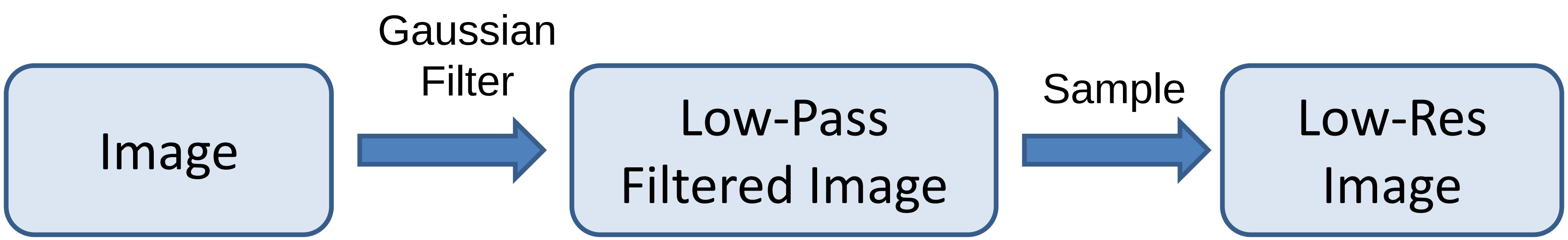
A: Depends

- SSD: faster, sensitive to overall intensity
- Normalized cross-correlation: slower, invariant to local average intensity and contrast

Q: What if we want to find larger or smaller eyes?

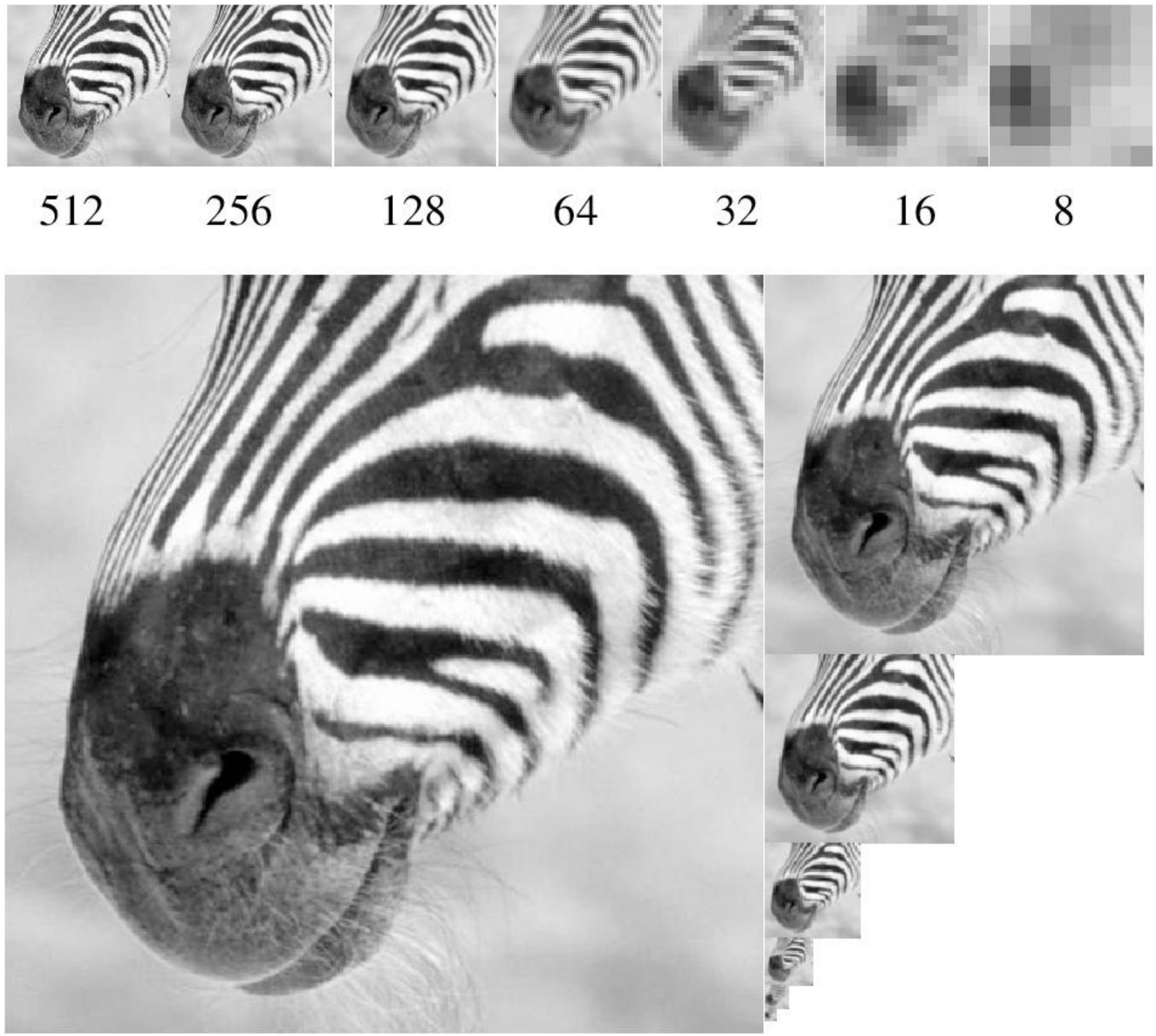
A: Image Pyramid

Review of Sampling



Slide: Hoiem

Gaussian pyramid



Source: Forsyth

Template Matching with Image Pyramids

Input: Image, Template

1. Match template at current scale

2. Downsample image

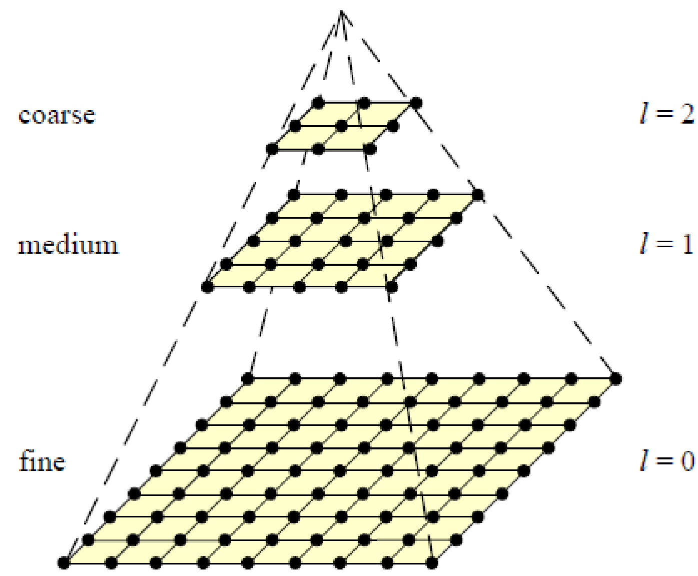
3. Repeat 1-2 until image is very small

4. Take responses above some threshold, perhaps with non-maxima suppression

Slide: Hoiem

Coarse-to-fine Image Registration

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

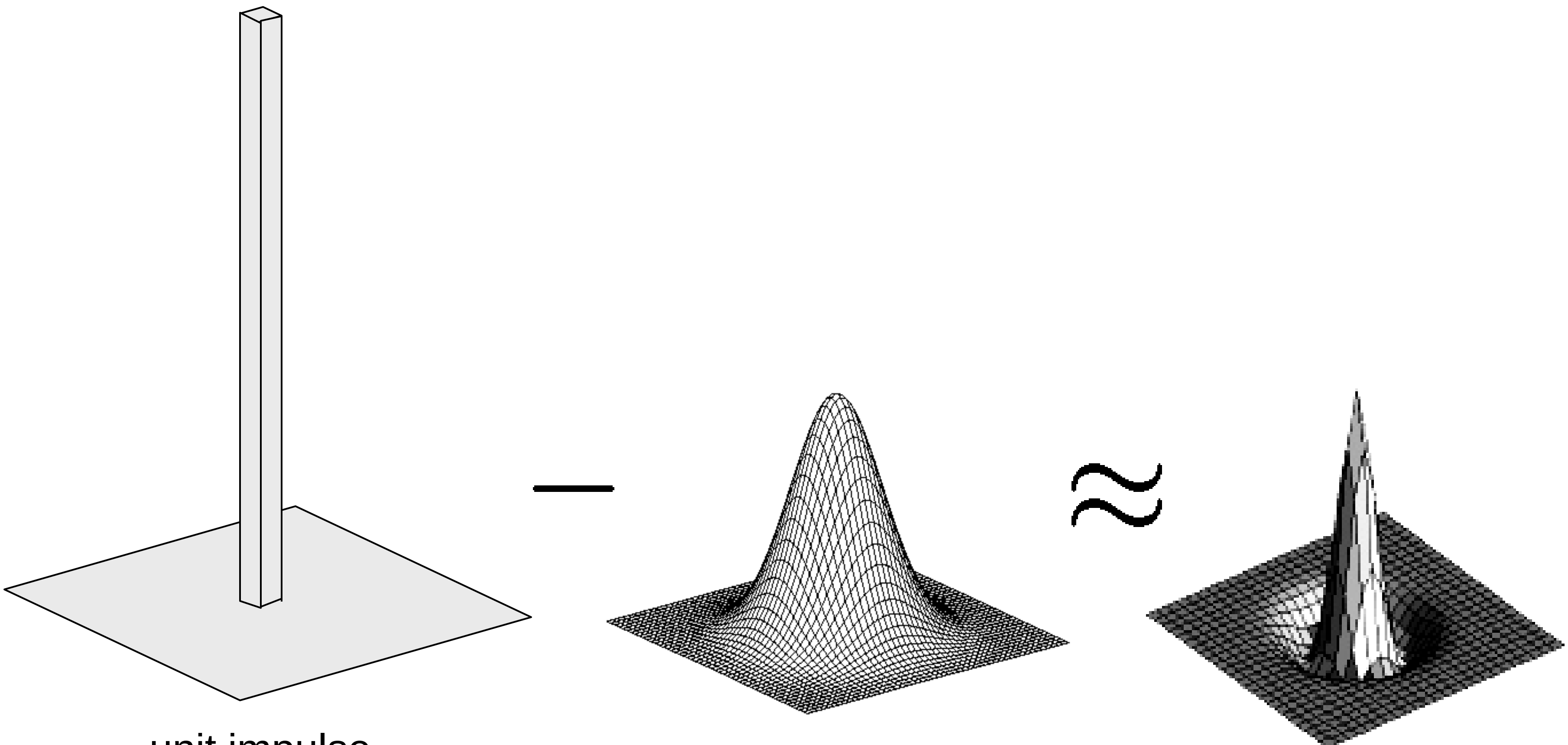


Why is this faster?

Are we guaranteed to get the same result?

Slide: Hoiem

Laplacian filter



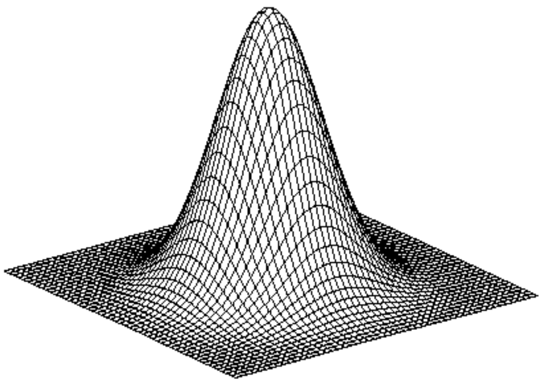
unit impulse

Gaussian

Laplacian of Gaussian

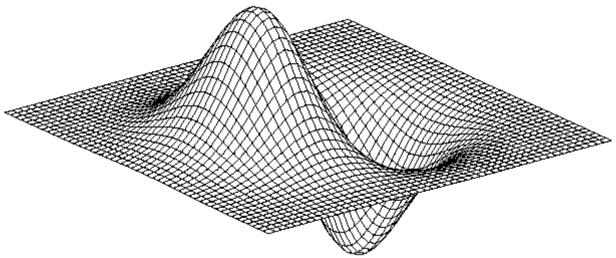
Source: Lazebnik

2D edge detection filters



Gaussian

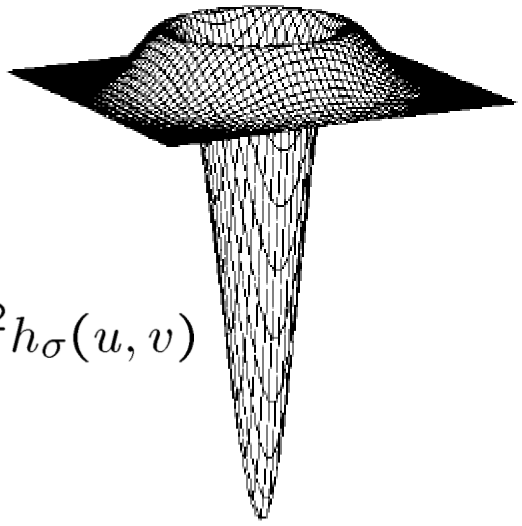
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

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

Laplacian of Gaussian

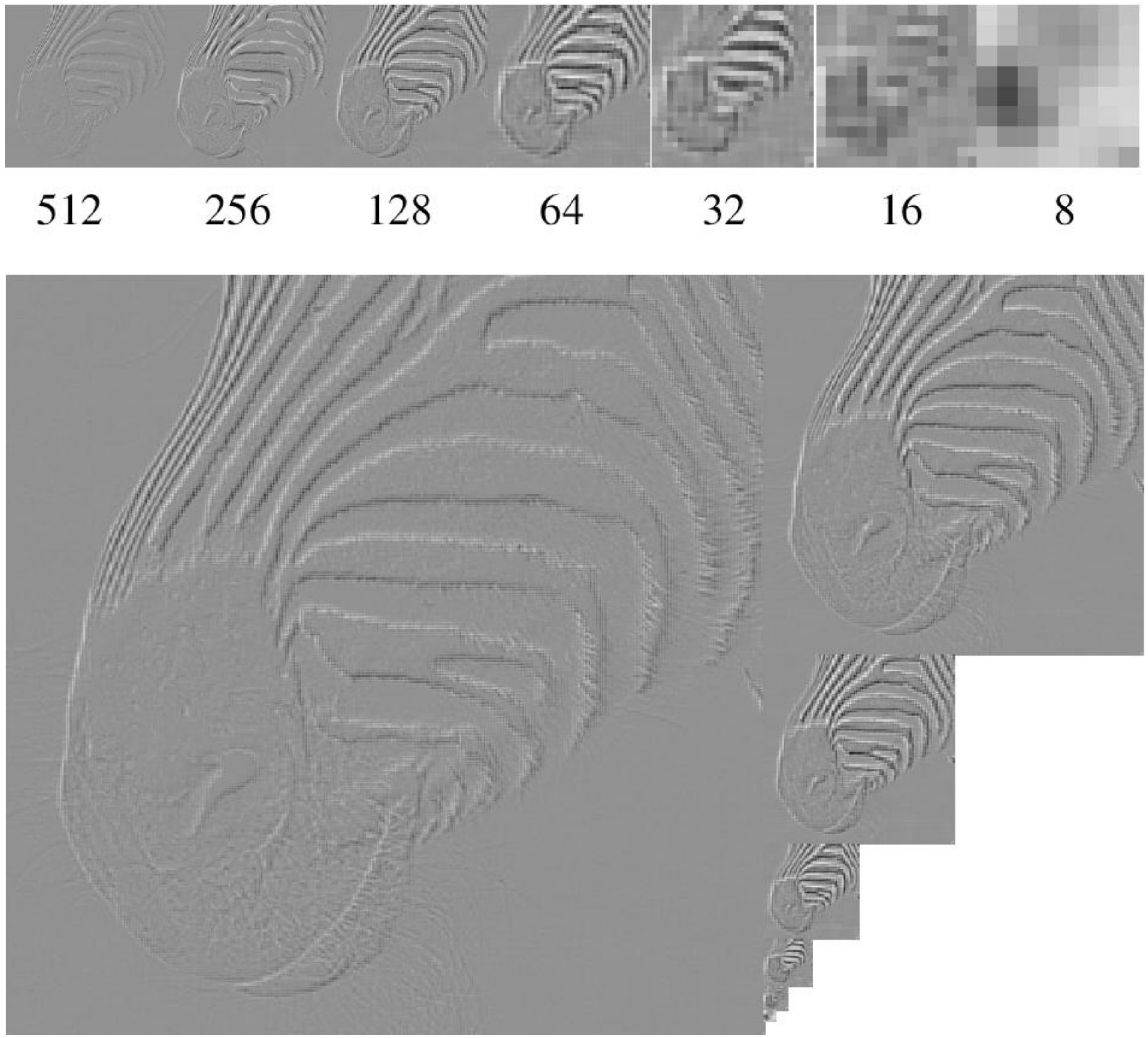


$$\nabla^2 h_\sigma(u,v)$$

∇^2 is the **Laplacian** operator:

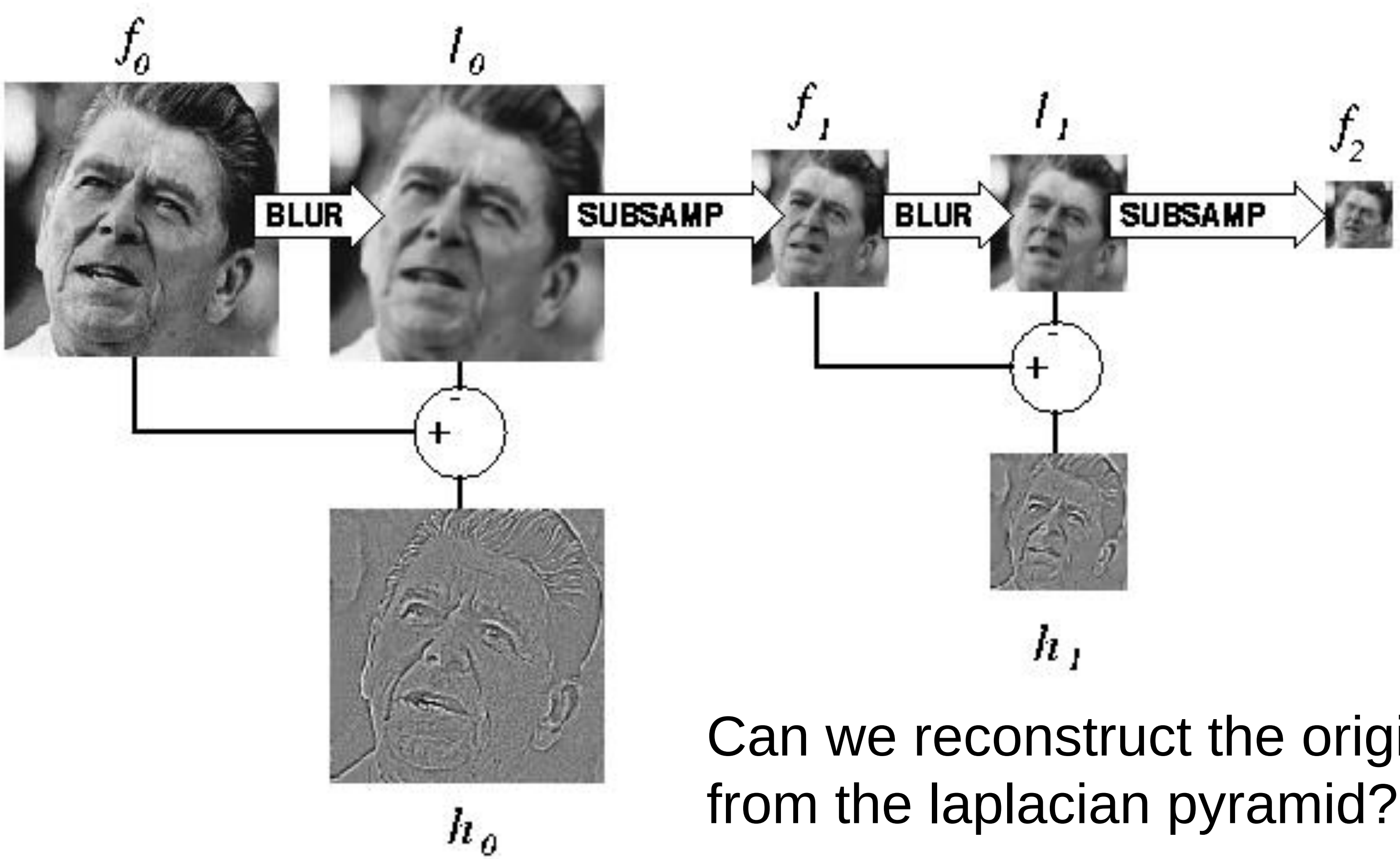
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Laplacian pyramid



Source: Forsyth

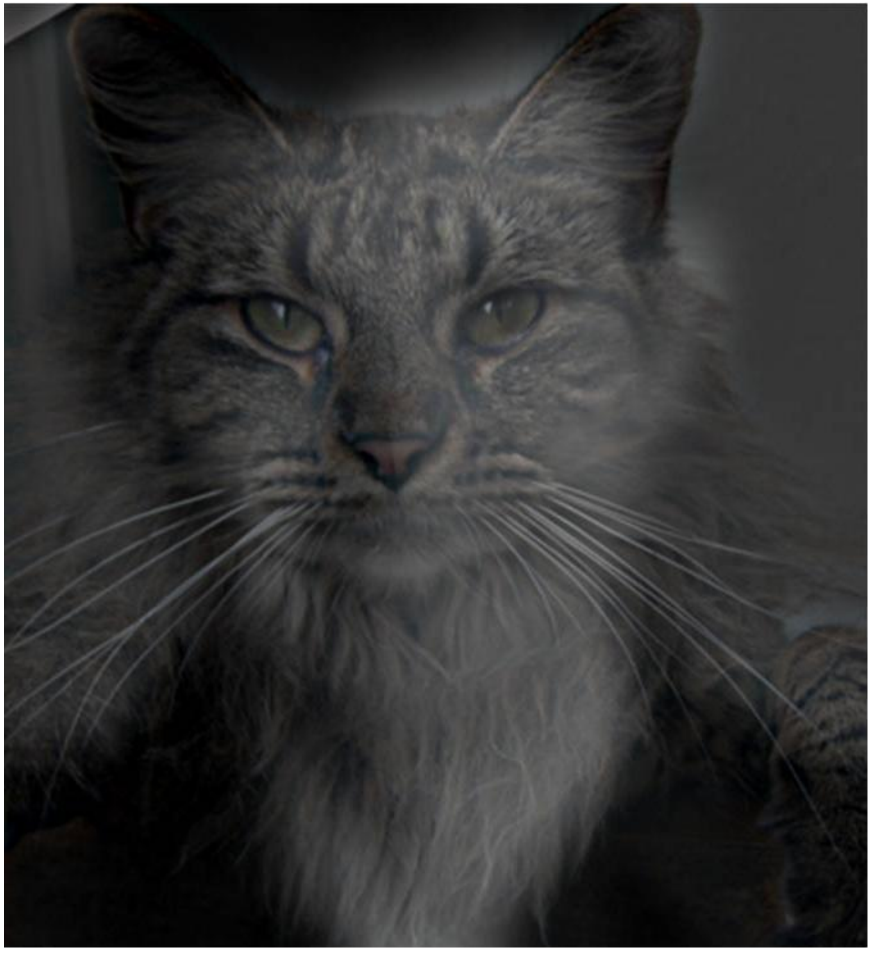
Computing Gaussian/Laplacian Pyramid



Can we reconstruct the original from the laplacian pyramid?

http://sepwww.stanford.edu/~morgan/texturematch/paper_html/node3.html

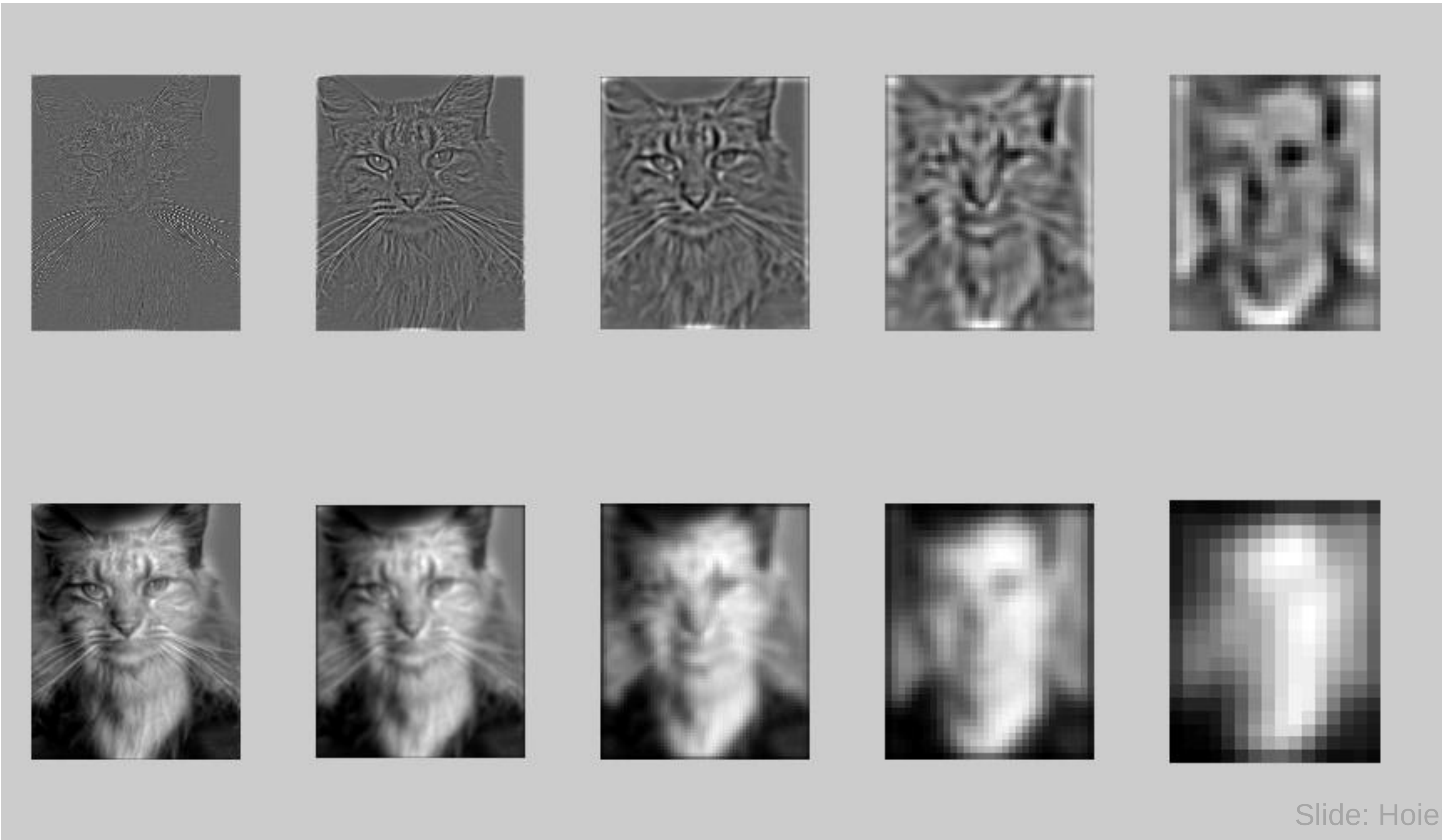
Hybrid Image



Slide: Hoiem

Hybrid Image in Laplacian Pyramid

High frequency → Low frequency



Slide: Hoiem

Image representation

- Pixels: great for spatial resolution, poor access to frequency

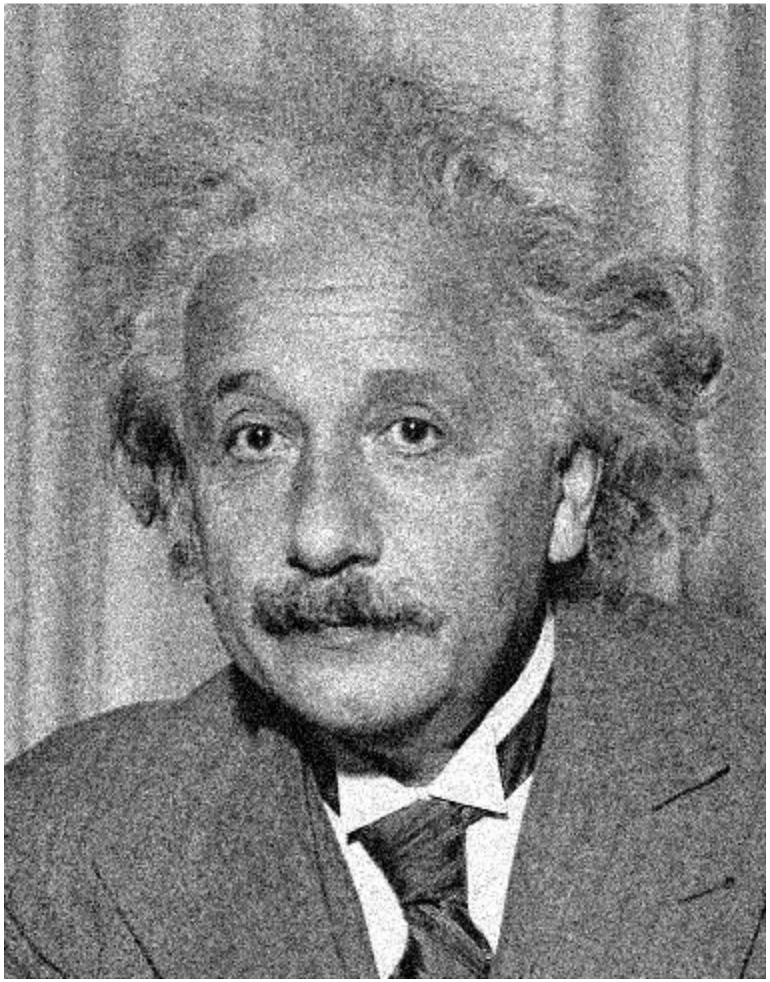
- Fourier transform: great for frequency, not for spatial info

- Pyramids/filter banks: balance between spatial and frequency information

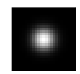
Major uses of image pyramids

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration
  - Course-to-fine

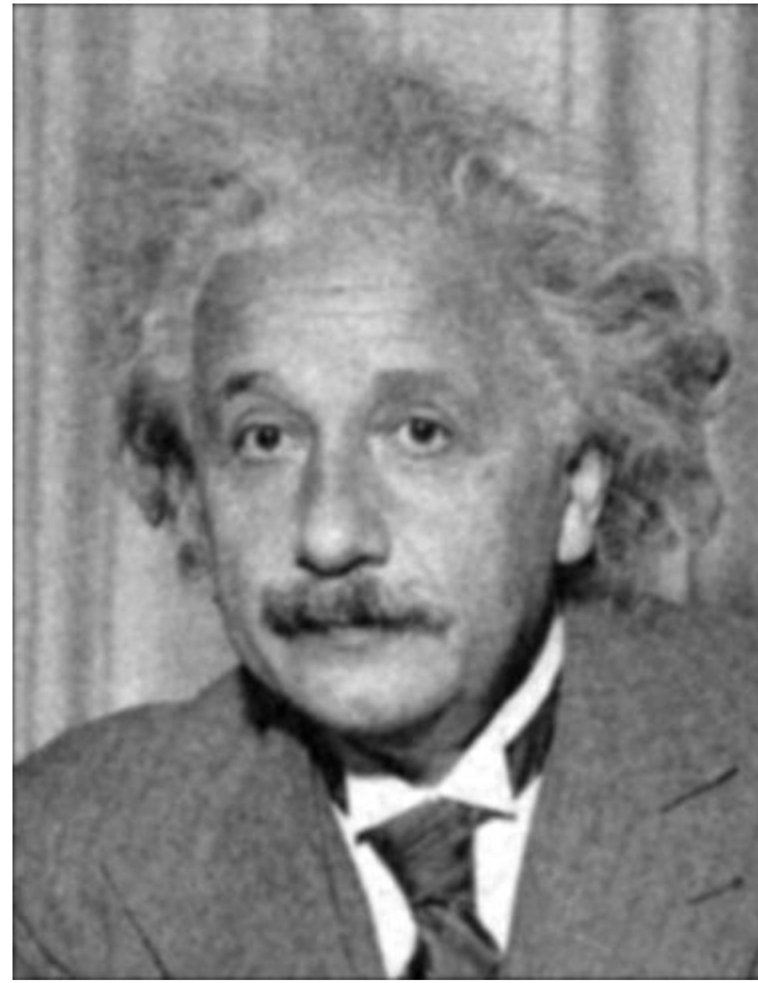
Denoising



Additive Gaussian Noise

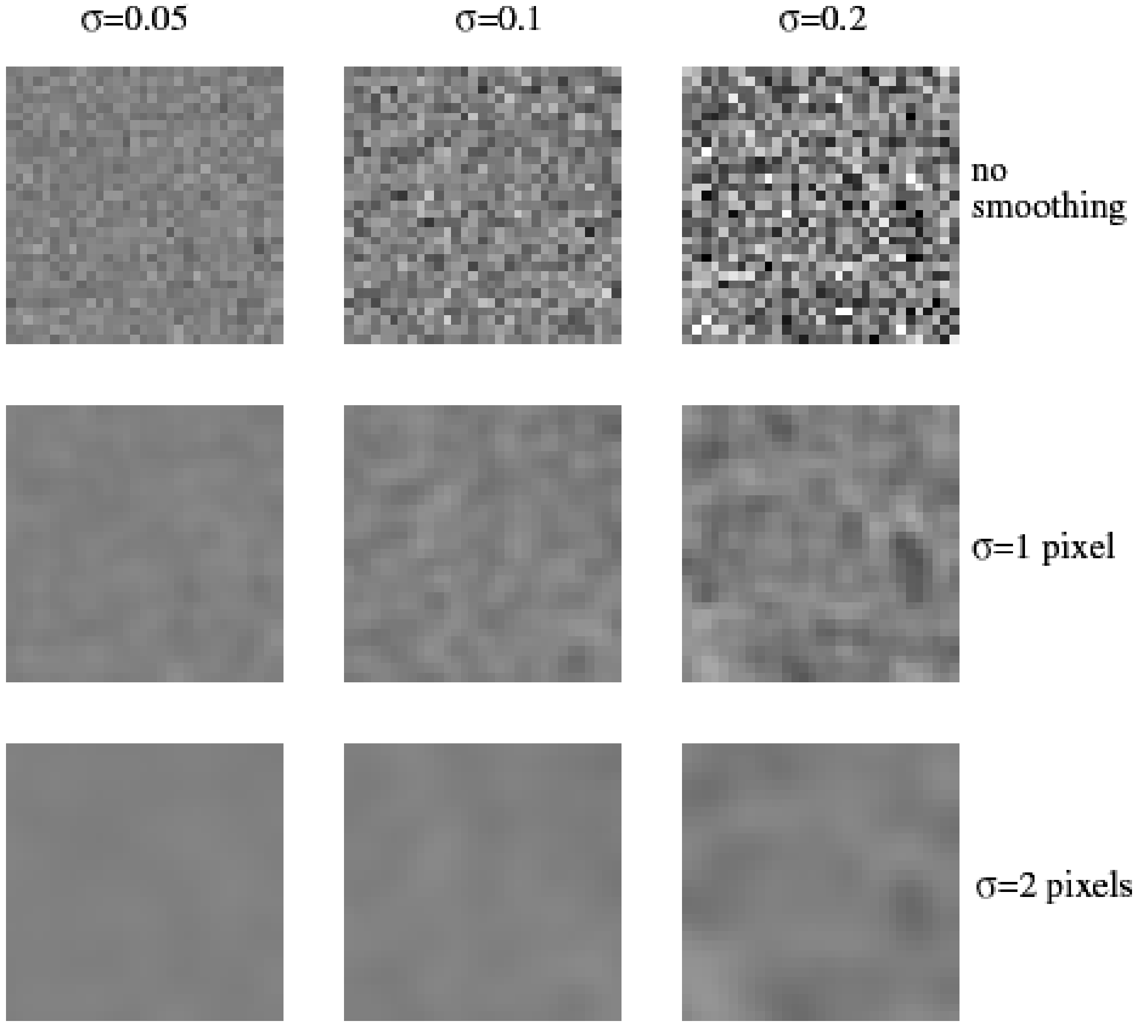


Gaussian Filter



Slide: Hoiem

Reducing Gaussian noise



Smoothing with larger standard deviations suppresses noise, but also blurs the image

Source: S. Lazebnik

Reducing salt-and-pepper noise by Gaussian smoothing

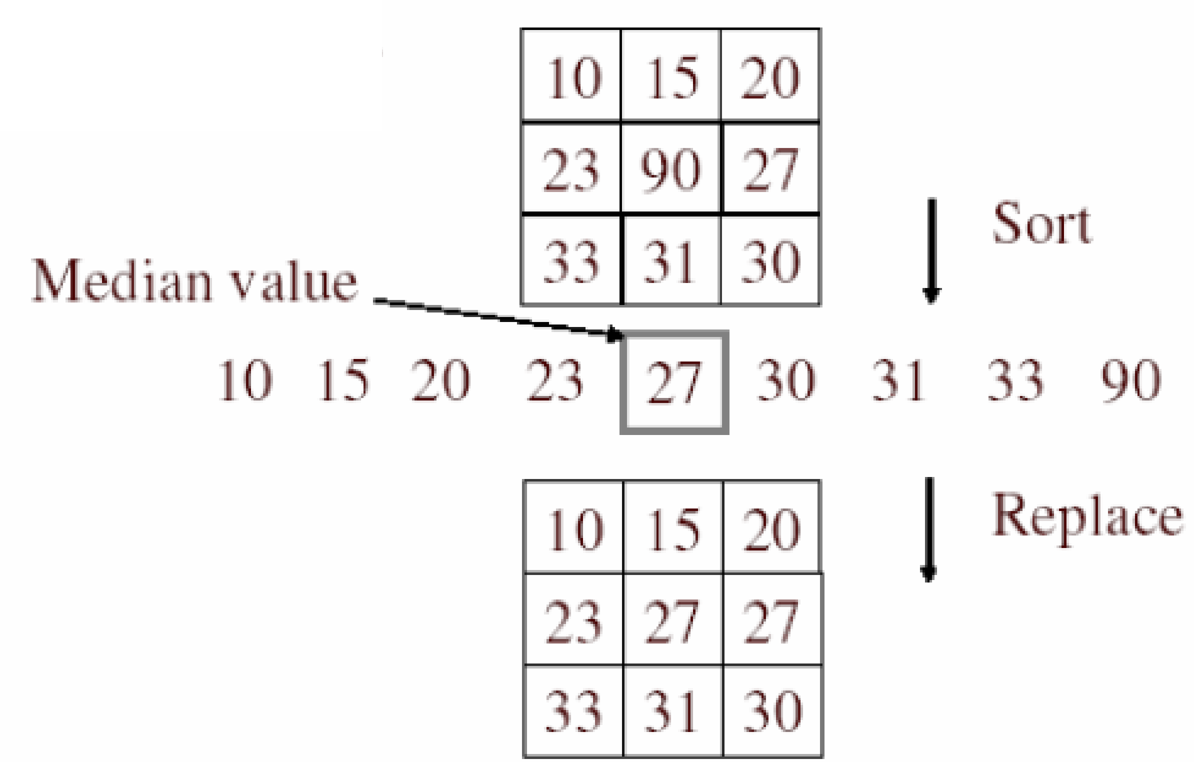
3x3

5x5

7x7

Alternative idea: Median filtering

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

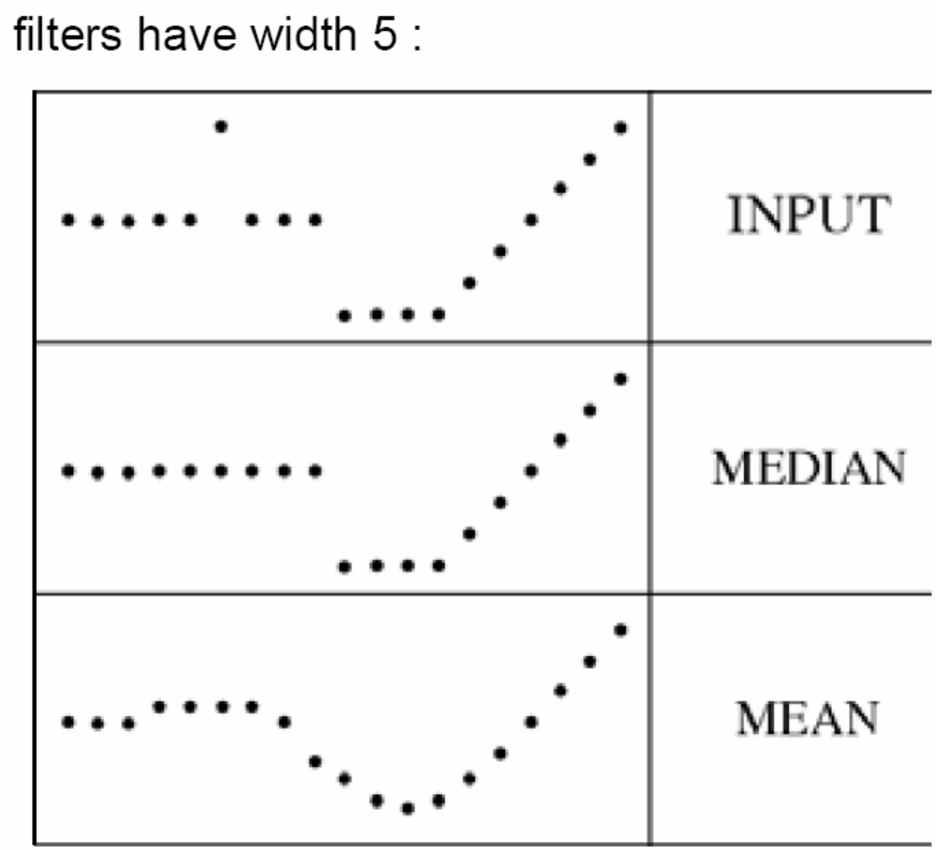


- Is median filtering linear?

Source: K. Grauman

Median filter

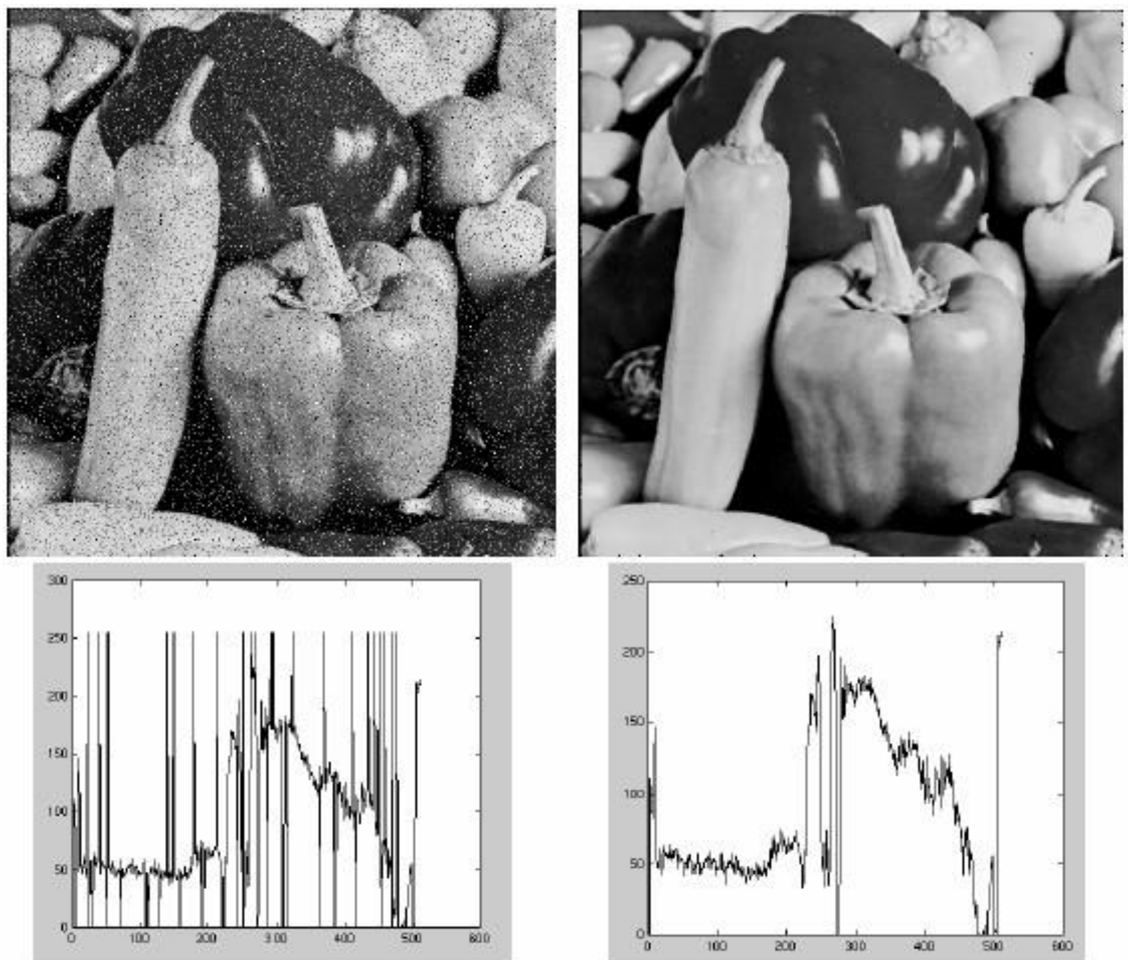
- What advantage does median filtering have over Gaussian filtering?
  - Robustness to outliers



Source: K. Grauman

Median filter

Salt-and-pepper noise Median filtered



- MATLAB: medfilt2(image, [h w])

Source: M. Hebert

Median vs. Gaussian filtering

3x3 5x5 7x7

Gaussian

Median

Other non-linear filters

- Weighted median (pixels further from center count less)
- Clipped mean (average, ignoring few brightest and darkest pixels)
- Bilateral filtering (weight by spatial distance *and* intensity difference)

Bilateral filtering

Image: http://vision.ai.uiuc.edu/?p=1455

Review of last three days

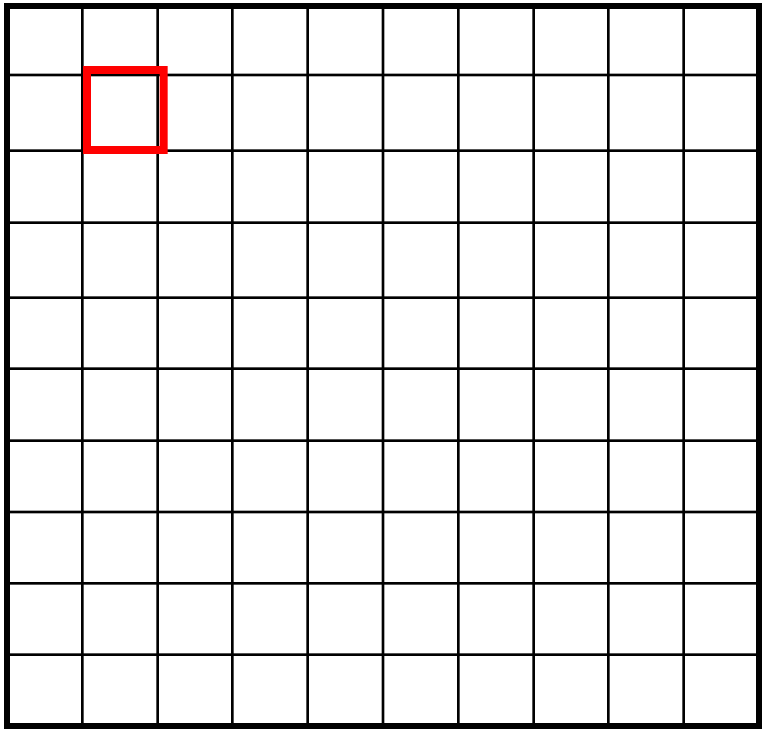
Review: Image filtering

$$g[\cdot\,,\cdot\,]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$



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

Credit: S. Seitz

Image filtering

$$g[\cdot\,,\cdot\,]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10							

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

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]\ \frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

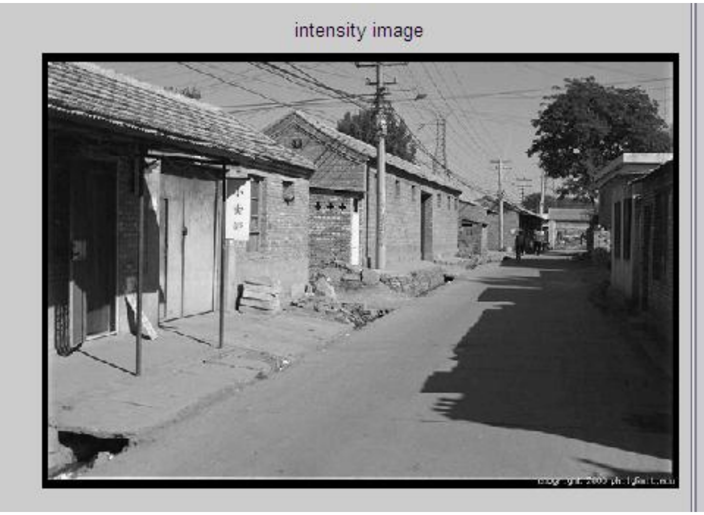
	0	10	20						

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

Credit: S. Seitz

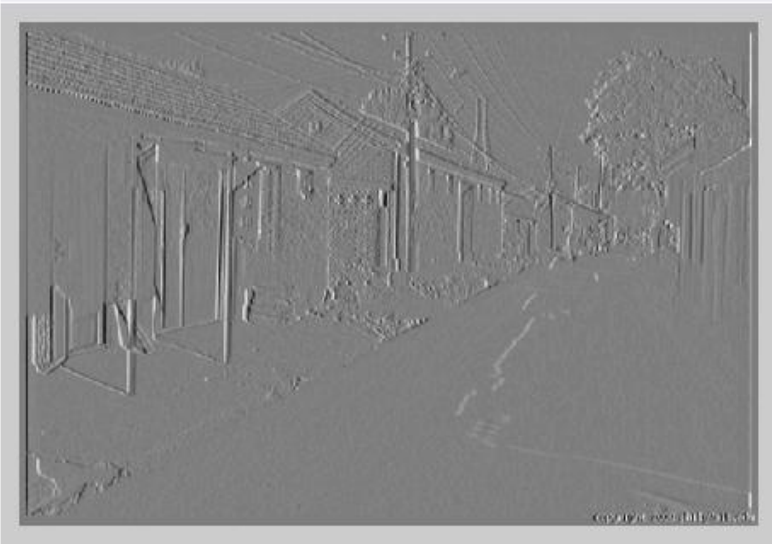
Filtering in spatial domain

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

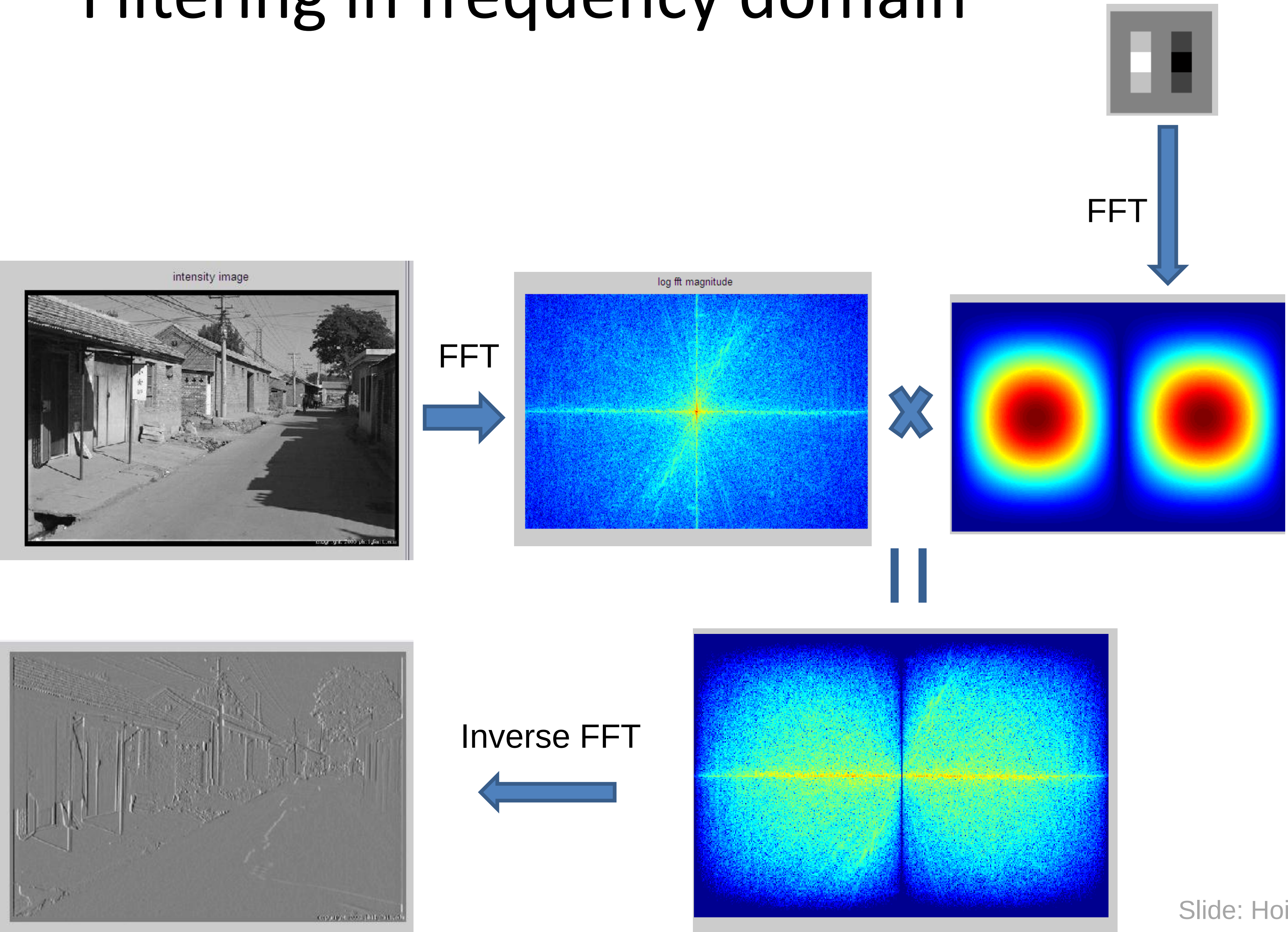


*

=



Filtering in frequency domain

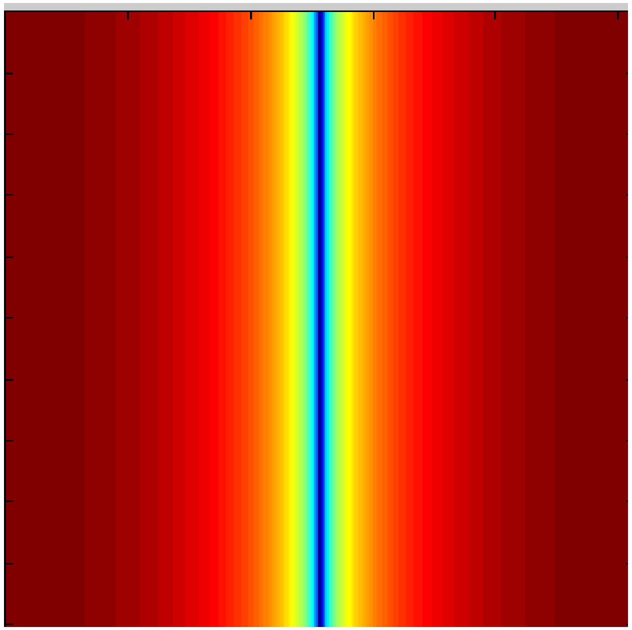


Slide: Hoiem

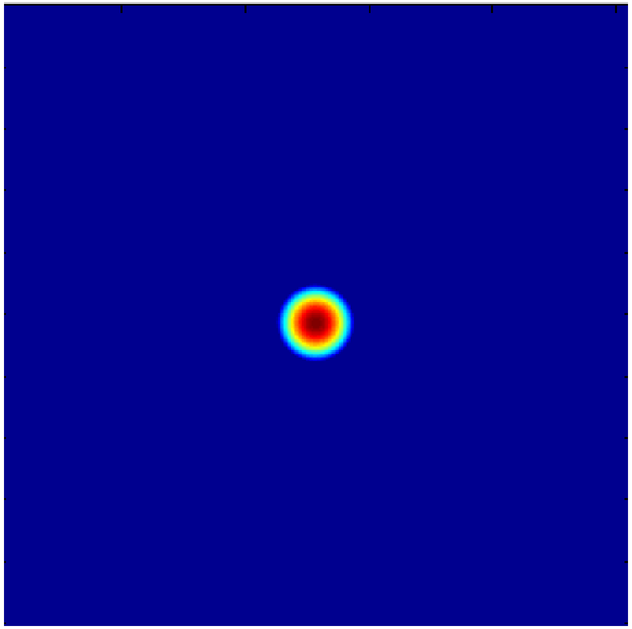
Review of Last 3 Days

- Linear filters for basic processing
  - Edge filter (high-pass)
  - Gaussian filter (low-pass)

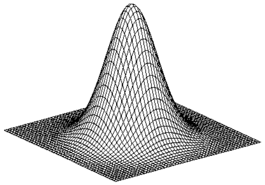
[-1 1]



FFT of Gradient Filter



FFT of Gaussian

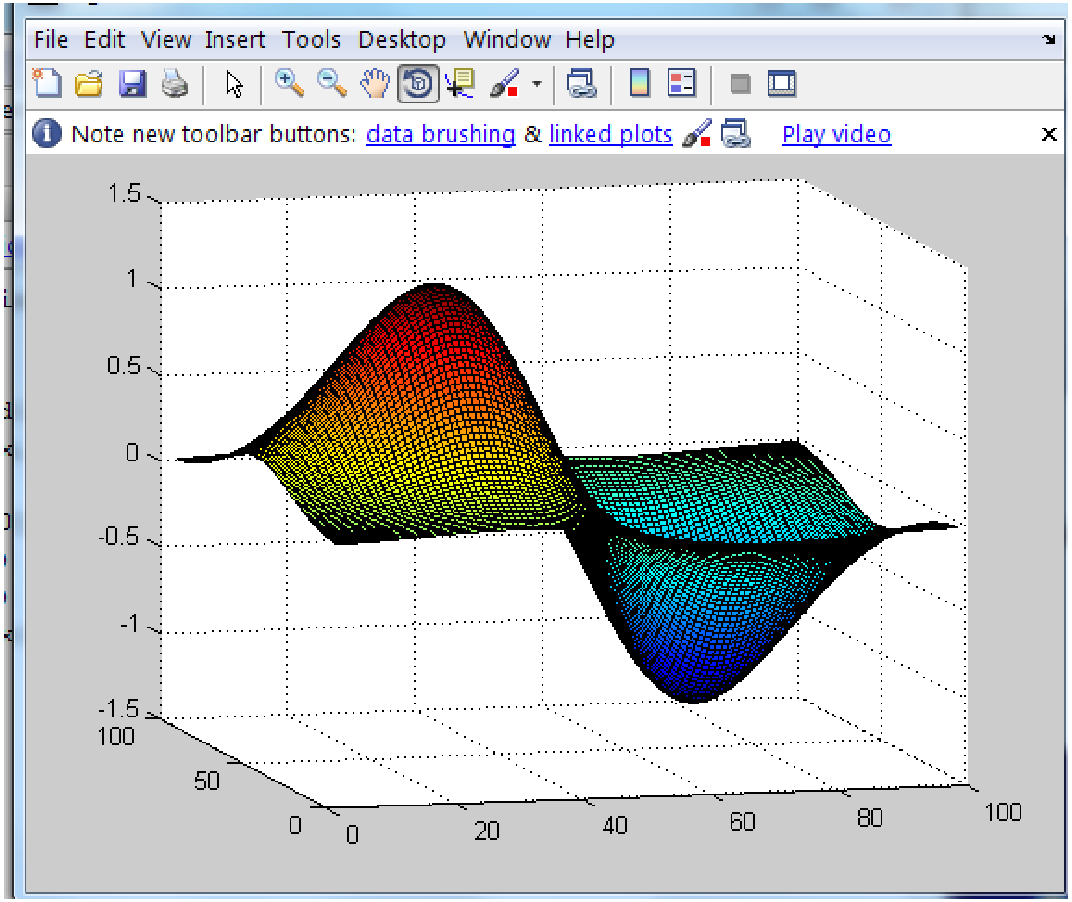


Gaussian

Slide: Hoiem

Review of Last 3 Days

- Derivative of Gaussian



Slide: Hoiem

Review of Last 3 Days

- Applications of filters
  - Template matching (SSD or Normxcorr2)
    - SSD can be done with linear filters, is sensitive to overall intensity
  - Gaussian pyramid
    - Coarse-to-fine search, multi-scale detection
  - Laplacian pyramid
    - More compact image representation
    - Can be used for compositing in graphics
  - Downsampling
    - Need to sufficiently low-pass before downsampling

Next Lectures

- Machine Learning Crash Course