Introduction to Computer Vision

Instructors: Jean Ponce and Matthew Trager
jean.ponce@inria.fr, matthew.trager@cims.nyu.edu

TAs: Jiachen (Jason) Zhu and Sahar Siddiqui
jiachen.zhu@nyu.edu, ss12414@nyu.edu

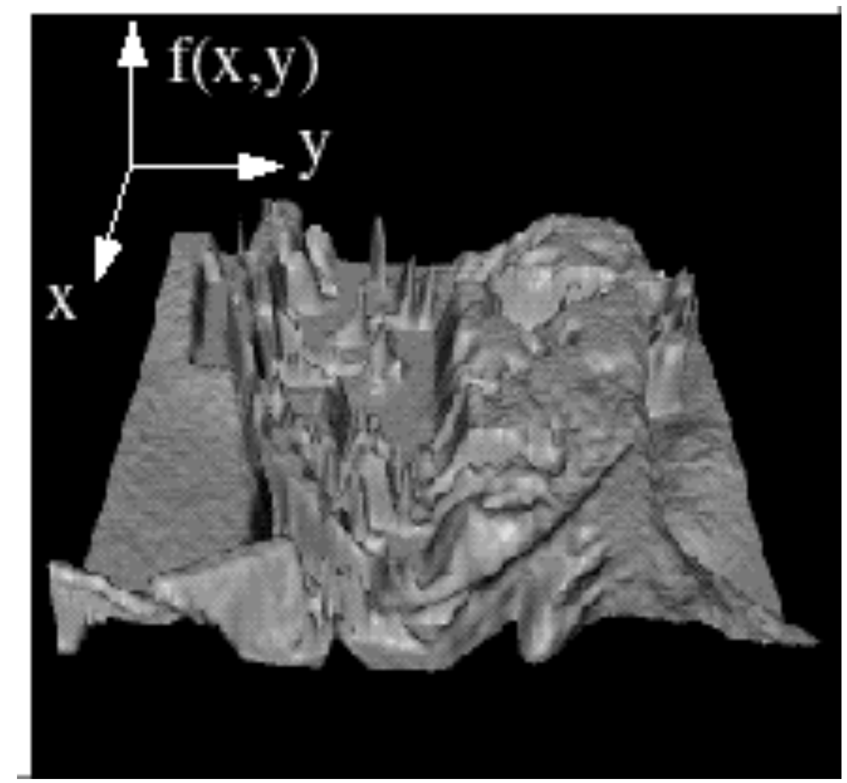
Outline

Image filtering

- Image filters in spatial domain
  - Filter is a mathematical operation on values of each patch
  - Smoothing, sharpening, measuring texture

- Image filters in the frequency domain
  - Filtering is a way to modify the frequencies of images
  - Denoising, sampling, image compression

What is a (digital) image?

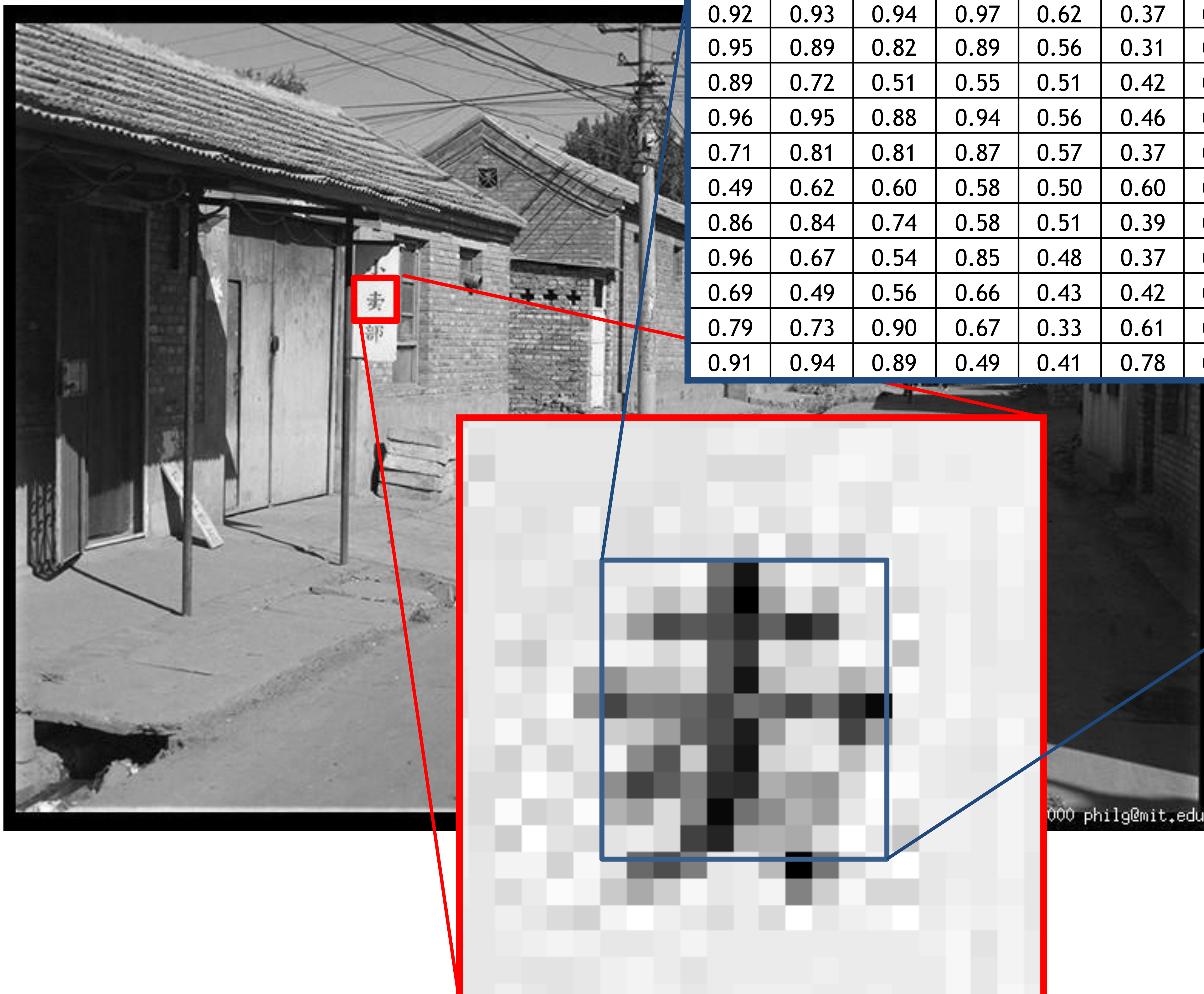


An image is function $f : \Omega \rightarrow V$ defined on a rectangular array of pixels:

$$\Omega = \{(x, y) \mid 1 \leq x \leq N_{cols},\ 1 \leq y \leq N_{rows}\} \subset \mathbb{Z}^2 .$$

For scalar images, the range is usually a discrete set, $V = \{0,\ldots,2^a - 1\}$.
Thus, f can also be viewed as a grid of integers.

The raster image (pixel matrix)



0.92	0.93	0.94	0.97	0.62	0.37	0.85	0.97	0.93	0.92	0.99
0.95	0.89	0.82	0.89	0.56	0.31	0.75	0.92	0.81	0.95	0.91
0.89	0.72	0.51	0.55	0.51	0.42	0.57	0.41	0.49	0.91	0.92
0.96	0.95	0.88	0.94	0.56	0.46	0.91	0.87	0.90	0.97	0.95
0.71	0.81	0.81	0.87	0.57	0.37	0.80	0.88	0.89	0.79	0.85
0.49	0.62	0.60	0.58	0.50	0.60	0.58	0.50	0.61	0.45	0.33
0.86	0.84	0.74	0.58	0.51	0.39	0.73	0.92	0.91	0.49	0.74
0.96	0.67	0.54	0.85	0.48	0.37	0.88	0.90	0.94	0.82	0.93
0.69	0.49	0.56	0.66	0.43	0.42	0.77	0.73	0.71	0.90	0.99
0.79	0.73	0.90	0.67	0.33	0.61	0.69	0.79	0.73	0.93	0.97
0.91	0.94	0.89	0.49	0.41	0.78	0.78	0.77	0.89	0.99	0.93

Image filtering

For each pixel, compute function of a neighborhood and output a new value:

$$h[i,j] = g(f[i+k,j+l]_{k,l})\,.$$

If g is linear, we talk about *linear filtering*:

$$h[i,j] = \sum_{k,h} g(k,l)\cdot f(i+k,j+l)\,.$$

- Same function applied at each position
- Output and input image are typically the same size

Applications

- Enhance images
  - Denoise, resize, increase contrast, etc.
- Extract information from images
  - Texture, edges, distinctive points, etc.
- Detect patterns
  - Template matching
- Convolutional Neural Networks

Example: box filter

- 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”

Slide credit: David Lowe (UBC)

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								

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

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							

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

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	20						

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

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	30					

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

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	20	30	30				

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

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	20	30	30				
				?					

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

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	20	30	30				
						?			
				50					

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

Credit: S. Seitz

Image filtering

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

$f[.,.]$

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	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

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

Credit: S. Seitz

Box filter

What does it do?

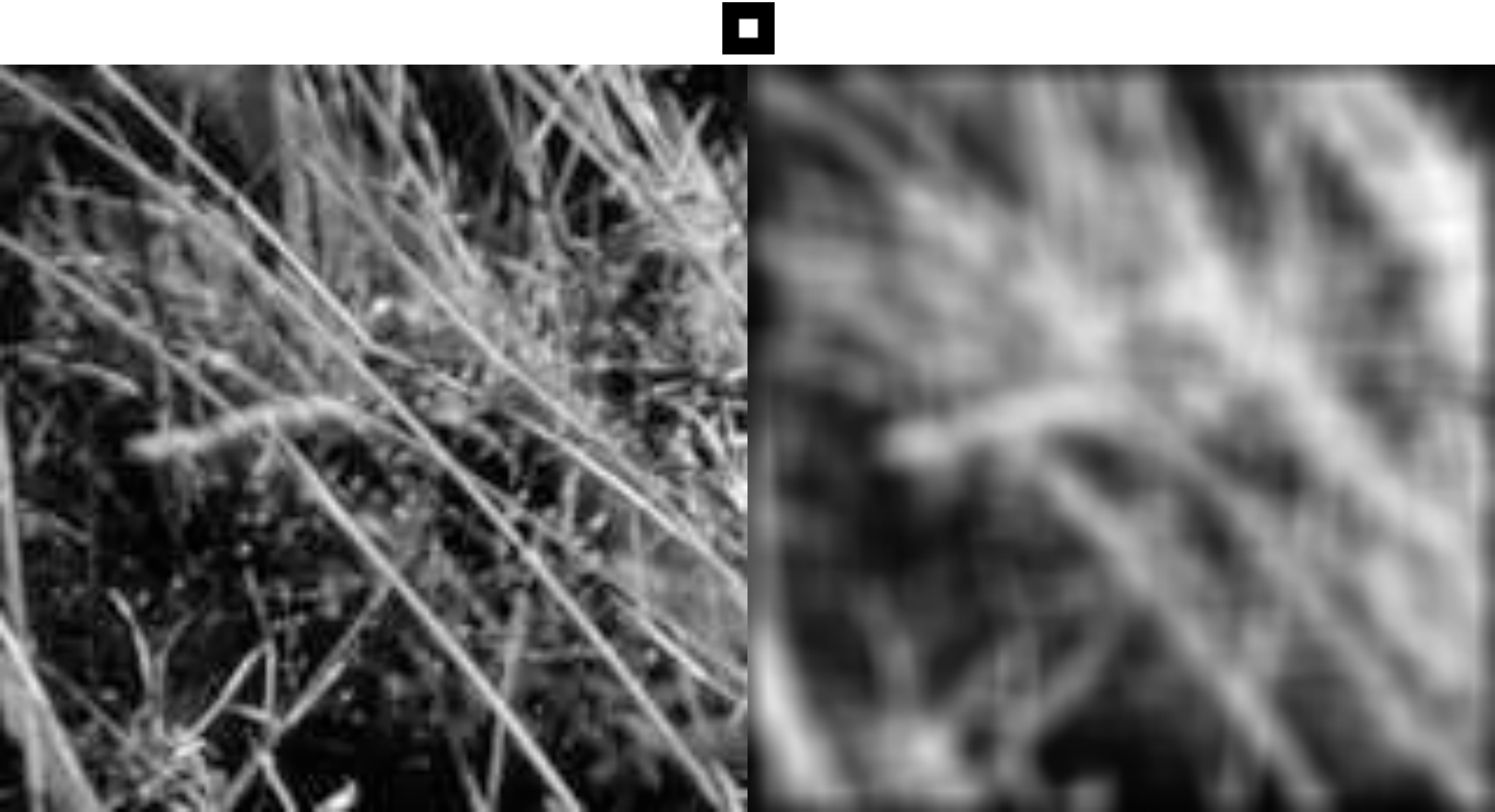
- Replaces each pixel with an average of its neighborhood
- Achieve smoothing effect (remove sharp features)

$$g[\cdot\,,\cdot\,]$$

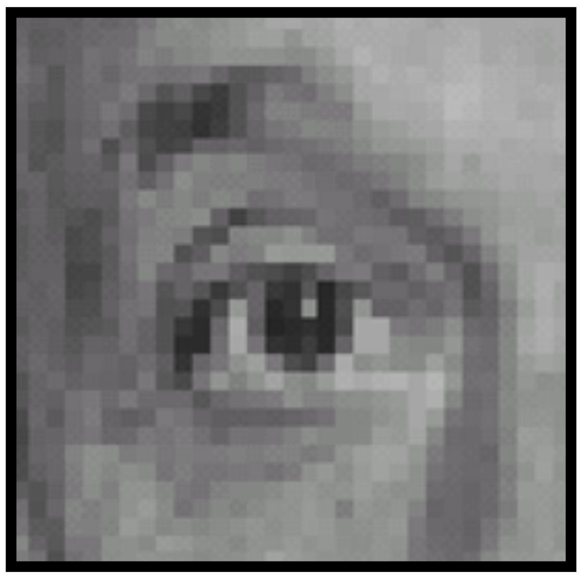
$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$$

Slide credit: David Lowe (UBC)

Smoothing with box filter



Practice with linear filters



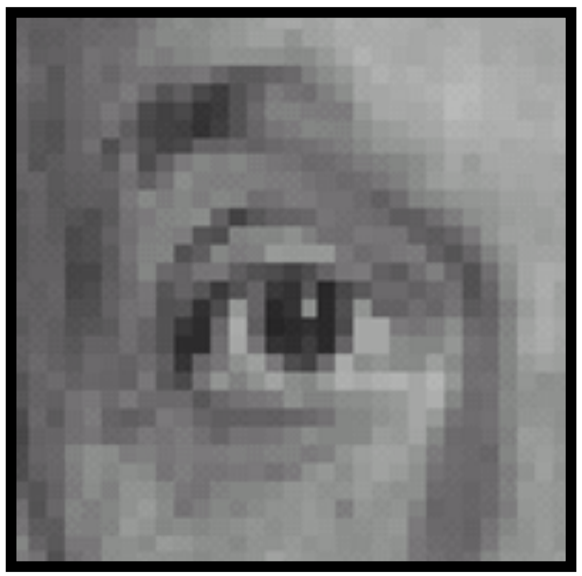
Original

0	0	0
0	1	0
0	0	0

?

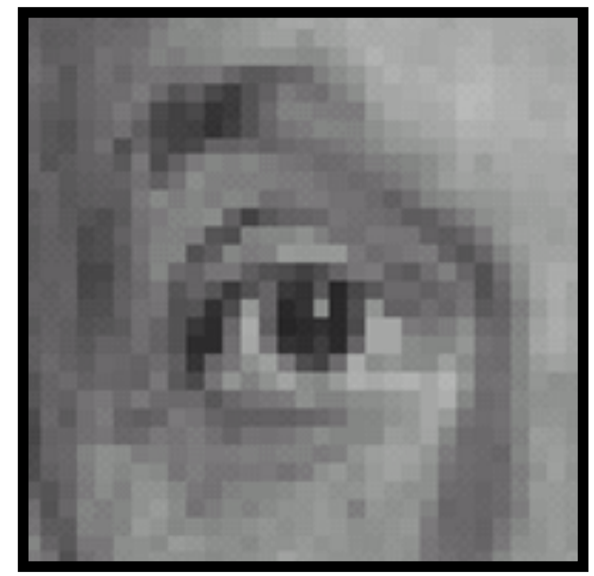
Source: D. Lowe

Practice with linear filters



Original

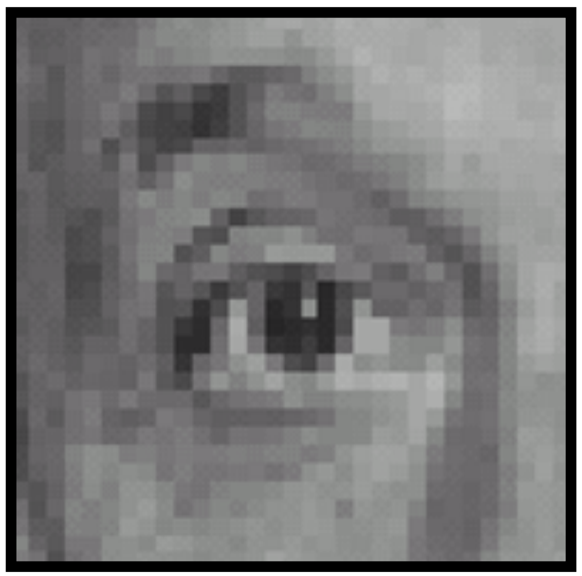
0	0	0
0	1	0
0	0	0



Filtered
(no change)

Source: D. Lowe

Practice with linear filters



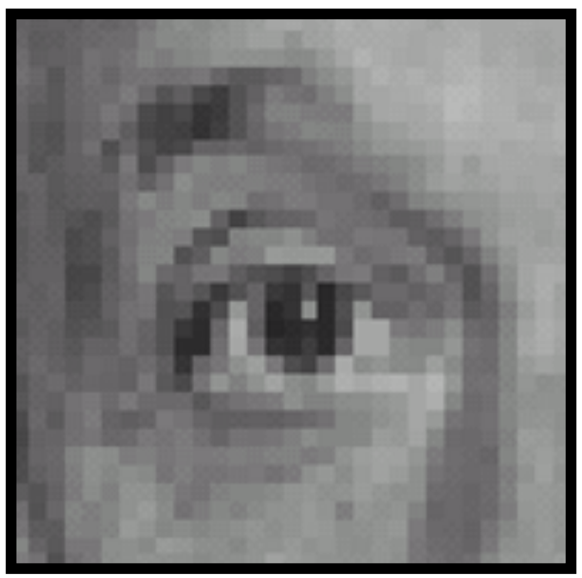
Original

0	0	0
0	0	1
0	0	0

?

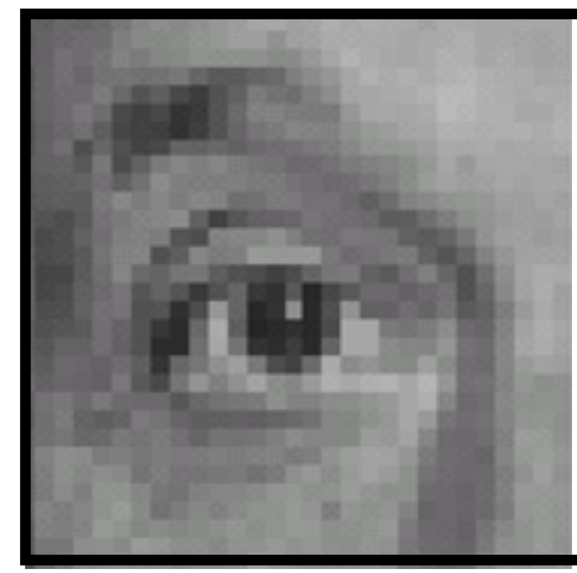
Source: D. Lowe

Practice with linear filters



Original

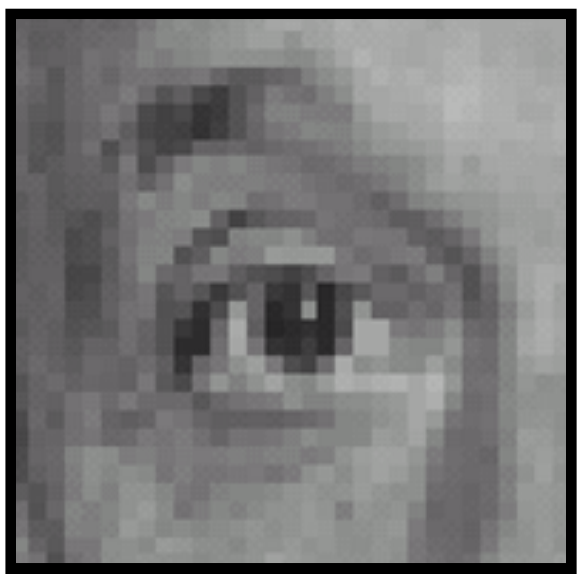
0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

Source: D. Lowe

Practice with linear filters



Original

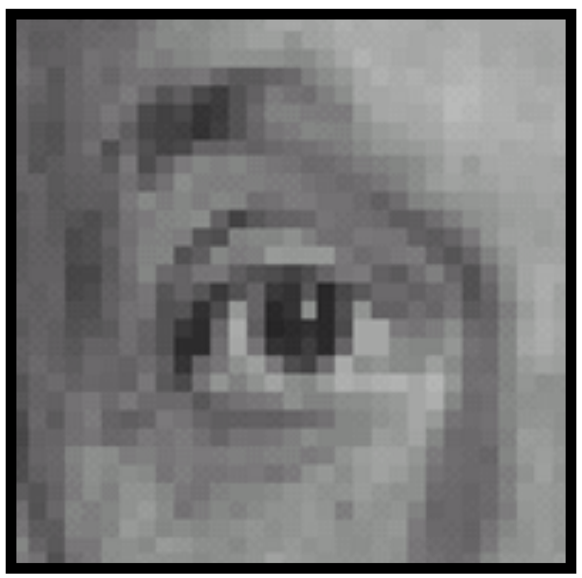
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

?

(Note that filter sums to 1)

Source: D. Lowe

Practice with linear filters



Original

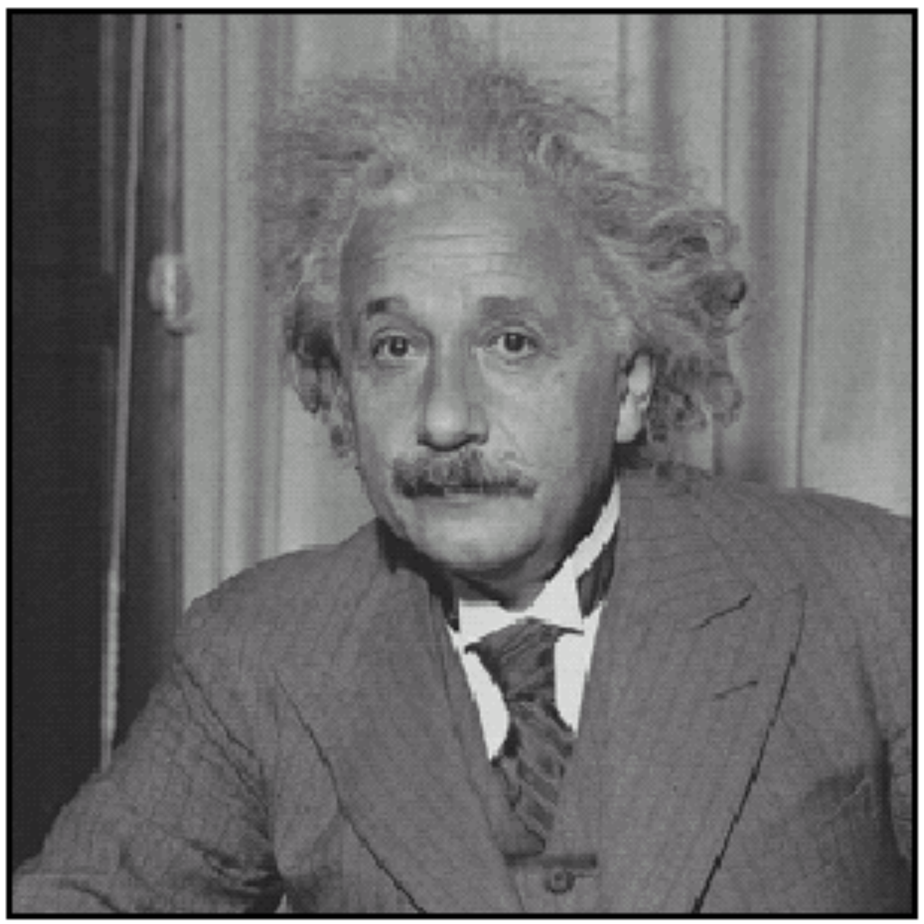
$$\begin{bmatrix}0&0&0\\0&2&0\\0&0&0\end{bmatrix} - \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

Sharpening filter

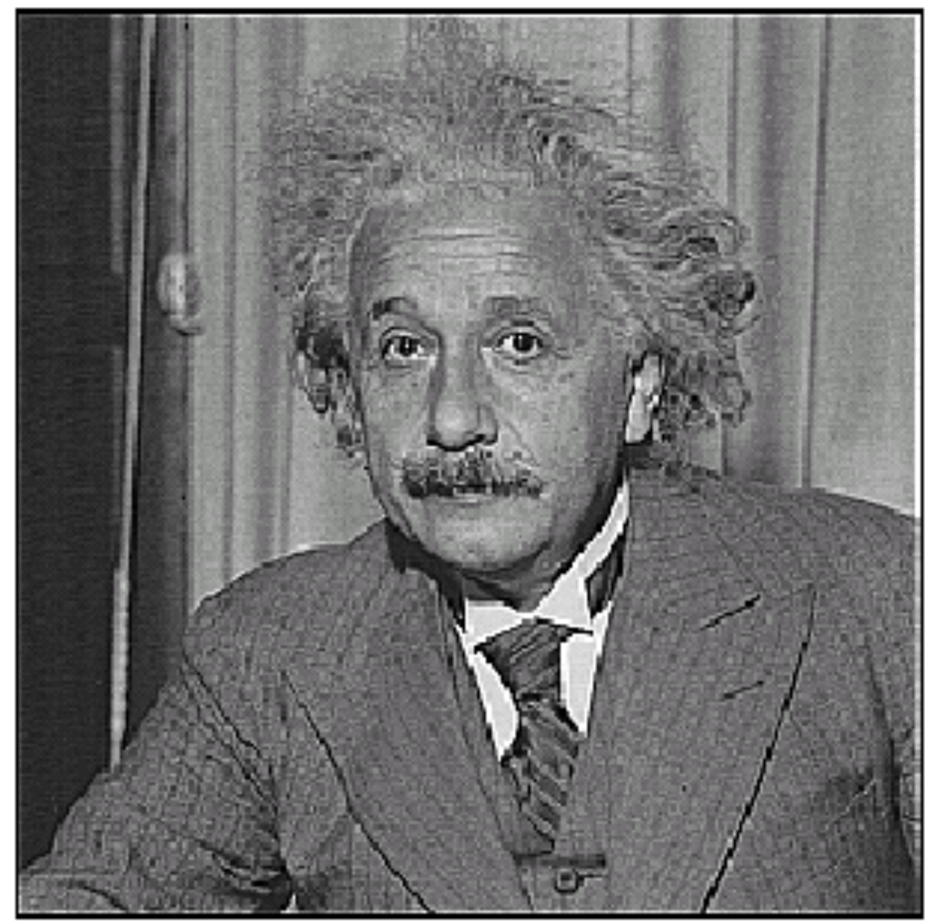
- Accentuates differences with local average

Source: D. Lowe

Sharpening



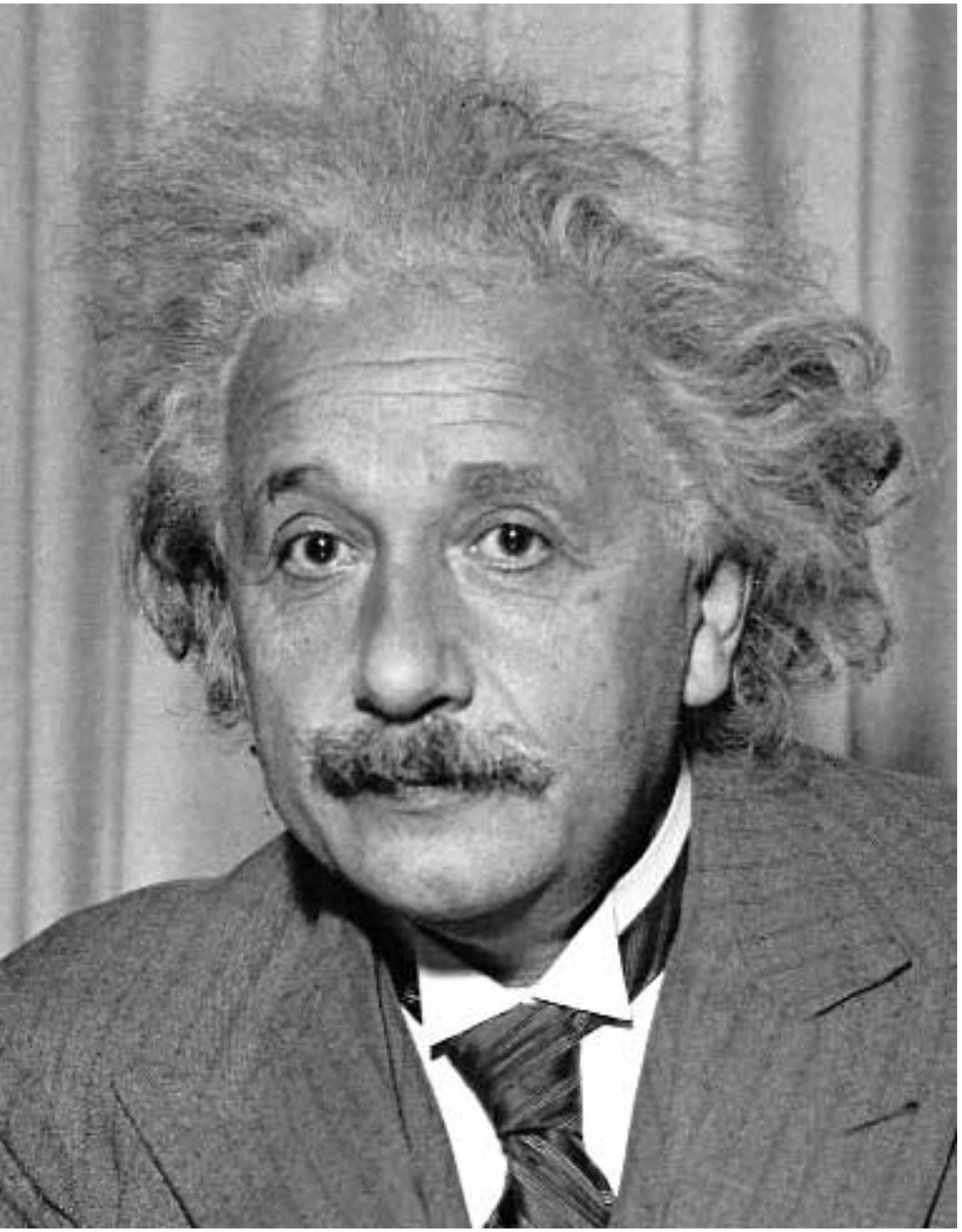
before



after

Source: D. Lowe

Other filters

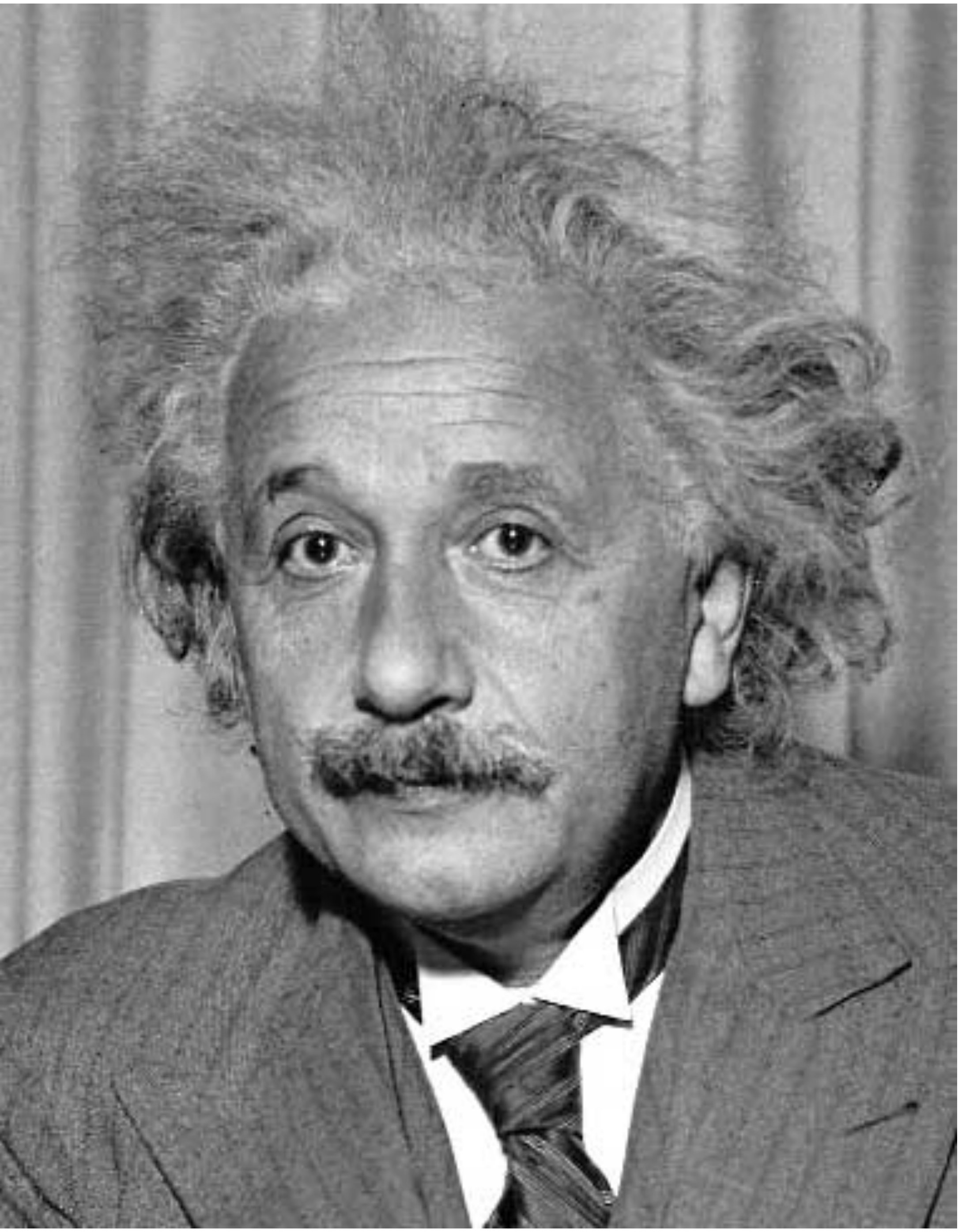


1	0	-1
2	0	-2
1	0	-1

Sobel

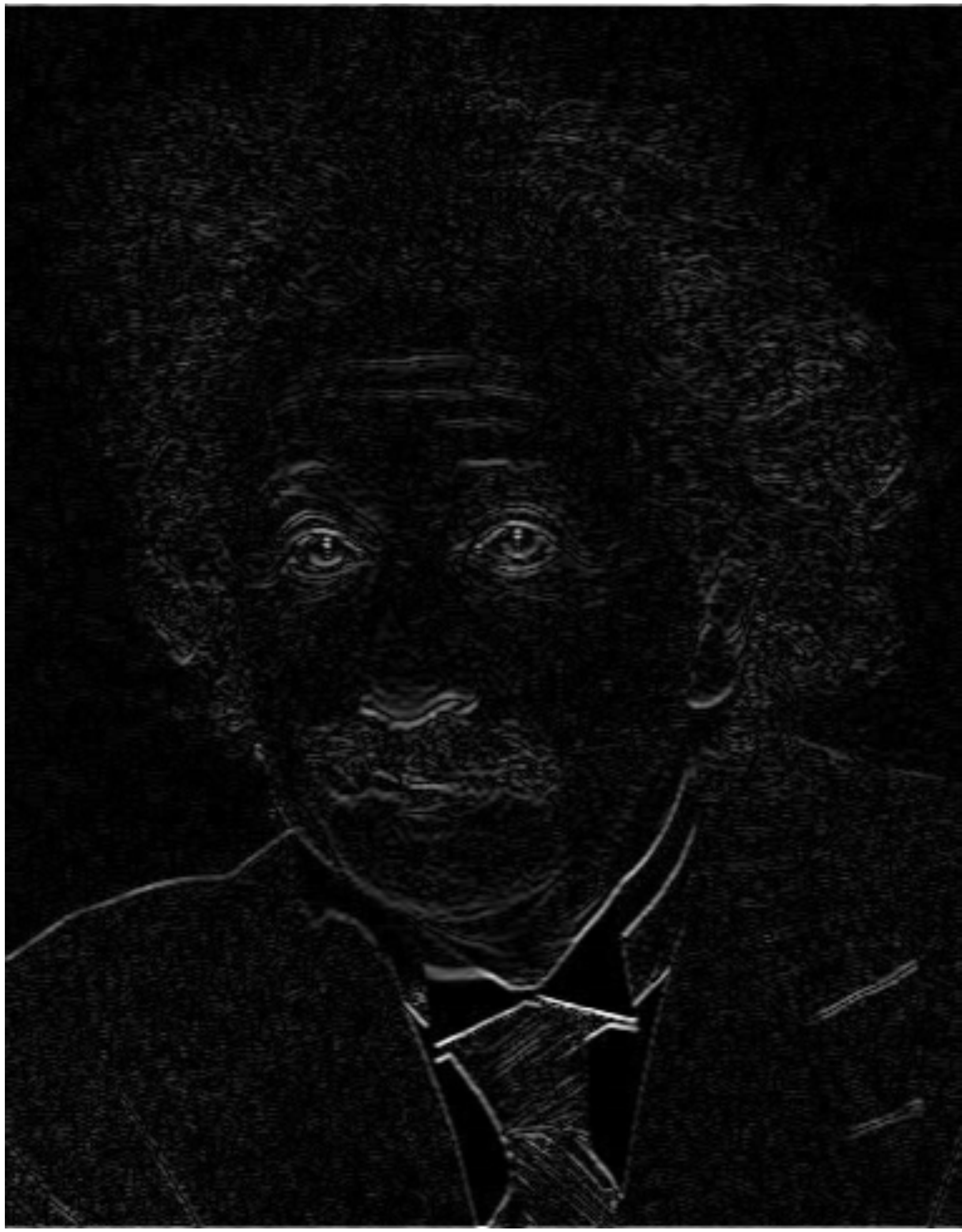
Vertical Edge
(absolute value)

Other filters



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

Sobel



Horizontal Edge
(absolute value)

Basic gradient filters

Horizontal Gradient

0	0	0
-1	0	1
0	0	0

or

-1	0	1

Vertical Gradient

0	-1	0
0	0	0
0	1	0

or

-1
0
1

Filter vs. convolution

Filtering (correlation):

$$h[i,j] = f \otimes g = \sum_{k,h} g[k,l] \cdot f[i+k, j+l] \, .$$

Convolution:

$$h[i,j] = f \star g = \sum_{k,h} g[k,l] \cdot f[i-k, j-l] \, .$$

Clearly equivalent if $g[i,j] = g[-i,-j]$, however in general there are differences.

Some properties

- Linearity:

$$g \star (f_1 + f_2) = g \star f_1 + g \star f_2, \quad g \otimes (f_1 + f_2) = g \otimes f_1 + g \otimes f_2$$

- Stationarity: if $T_{[u,v]}(f)[i,j] = f[i-u, j-v]$

$$g \star T_{[u,v]}(f) = T_{[u,v]}(g \star f), \quad g \otimes T_{[u,v]}(f) = T_{[u,v]}(g \otimes f)$$

Theorem: any linear shift-invariant operator can be represented as a convolution

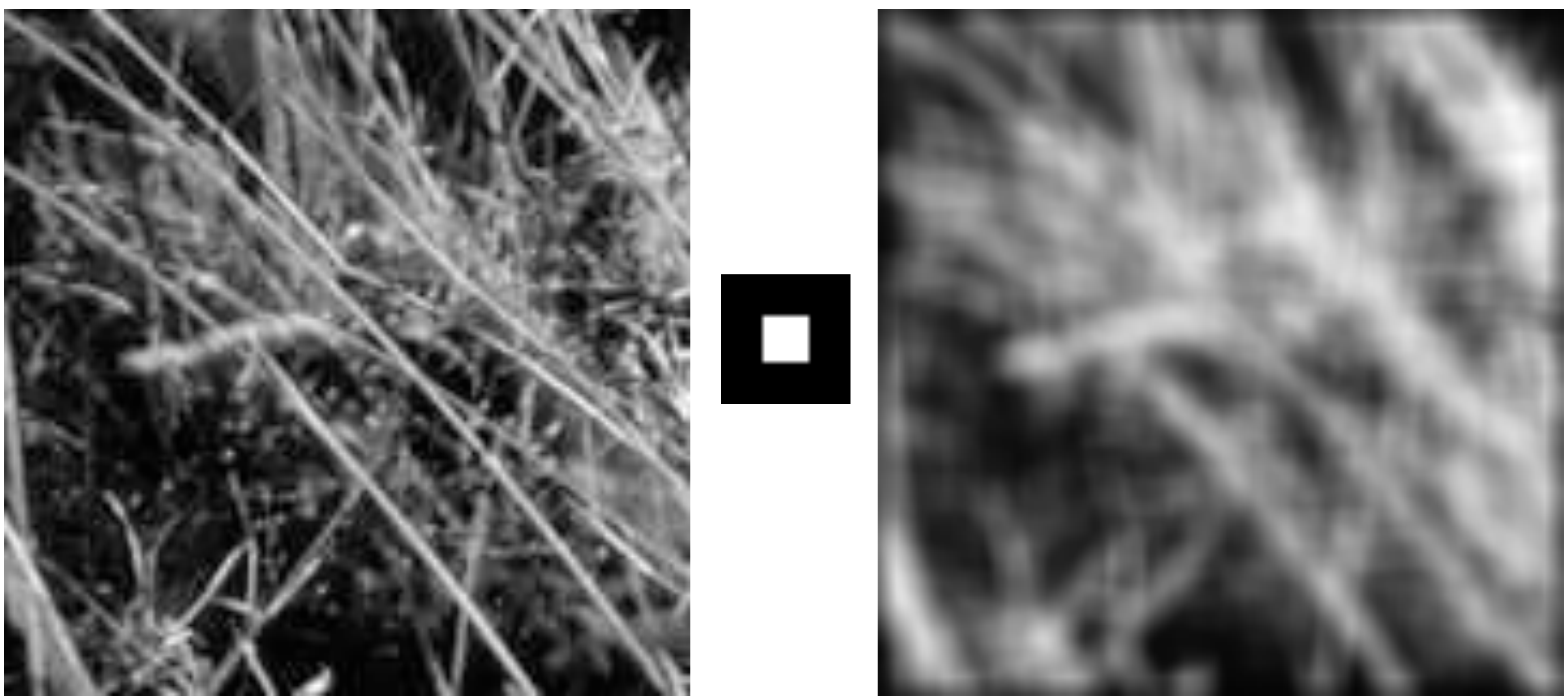
- Associativity and commutativity:

$$g \star (h \star f) = (g \star h) + g \star f, \quad g \otimes (h \otimes f) \neq (g \otimes h) \otimes f$$

$$g \star f = f \star g, \quad g \otimes f \neq f \otimes g$$

Smoothing with box filter revisited

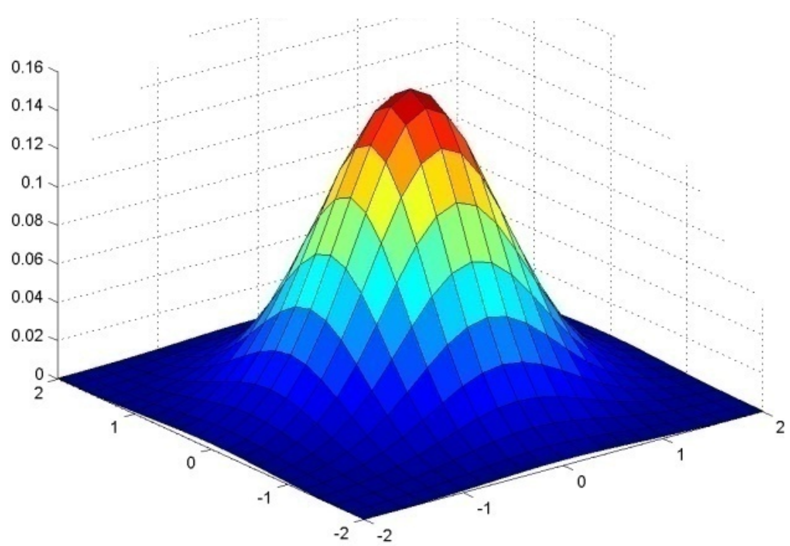
- What's wrong with this picture?
- What's the solution?

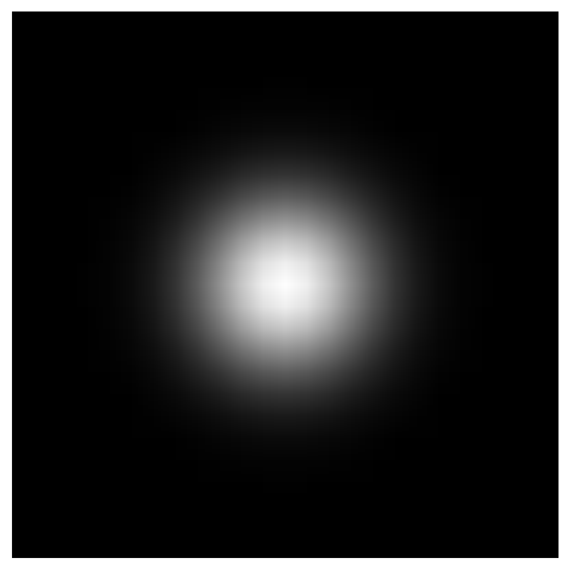


Source: D. Forsyth

Gaussian Kernel

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





0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

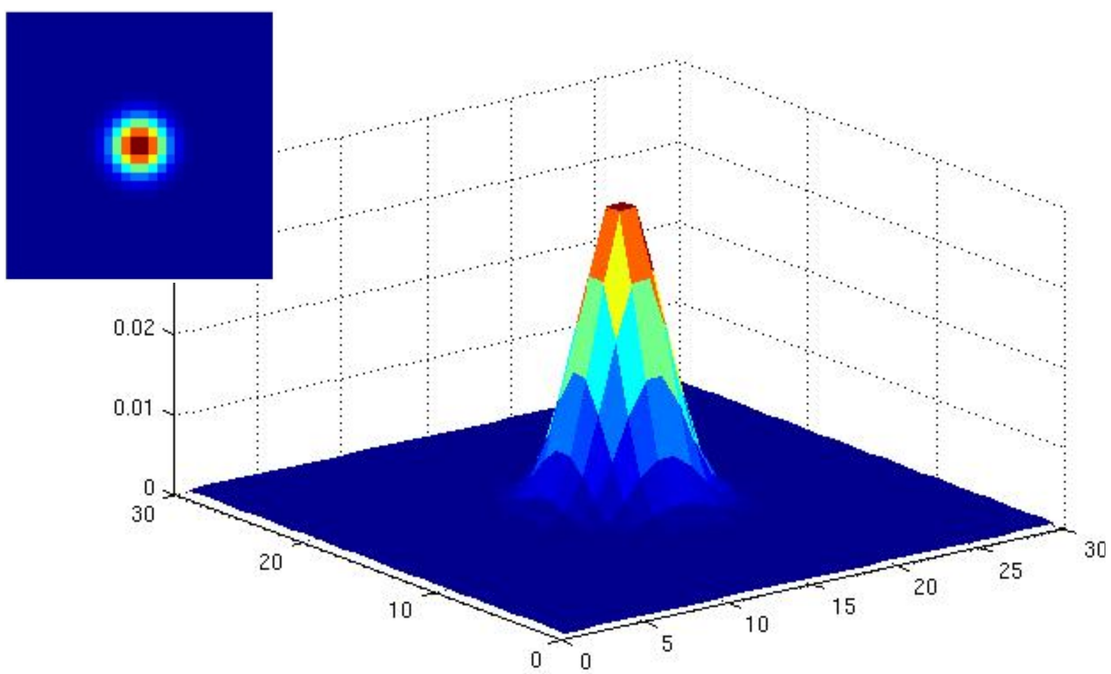
5 x 5, $\sigma = 1$

- Constant factor at front makes volume sum to 1 (can be ignored when computing the filter values, as we should renormalize weights to sum to 1 in any case)

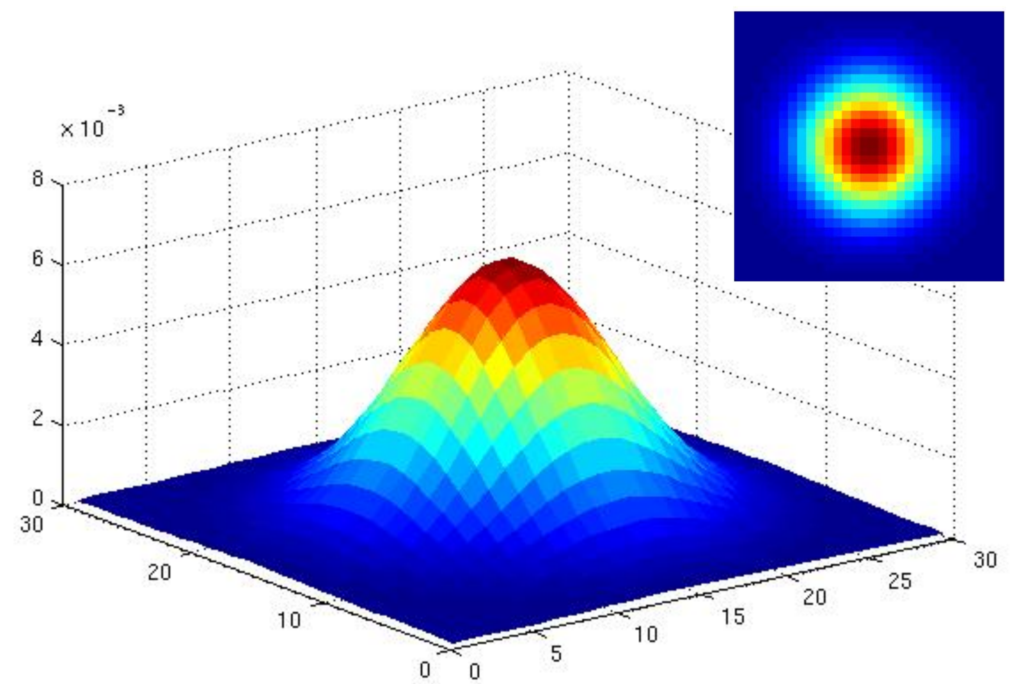
Source: C. Rasmussen

Gaussian Kernel

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



σ = 2 with 30 x 30
kernel



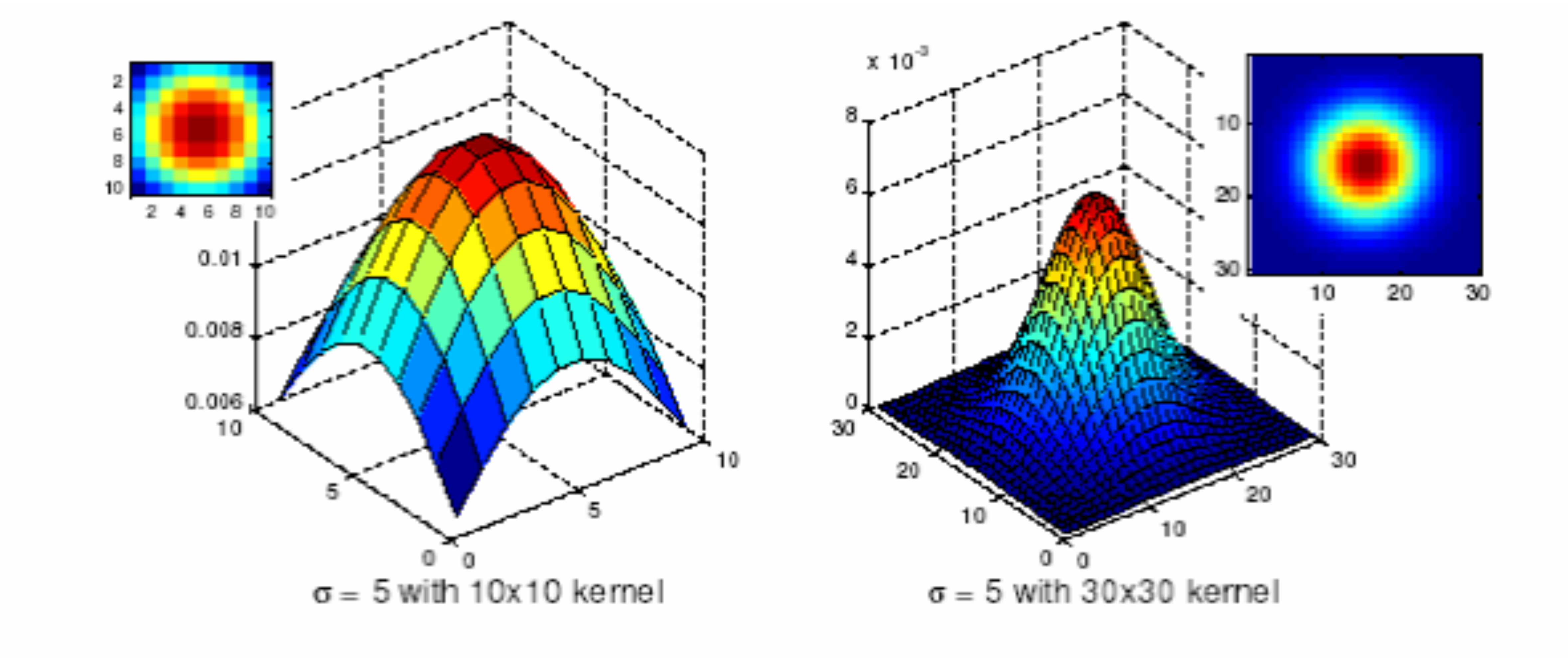
σ = 5 with 30 x 30
kernel

- Standard deviation σ: determines extent of smoothing

Source: K. Grauman

Choosing kernel width

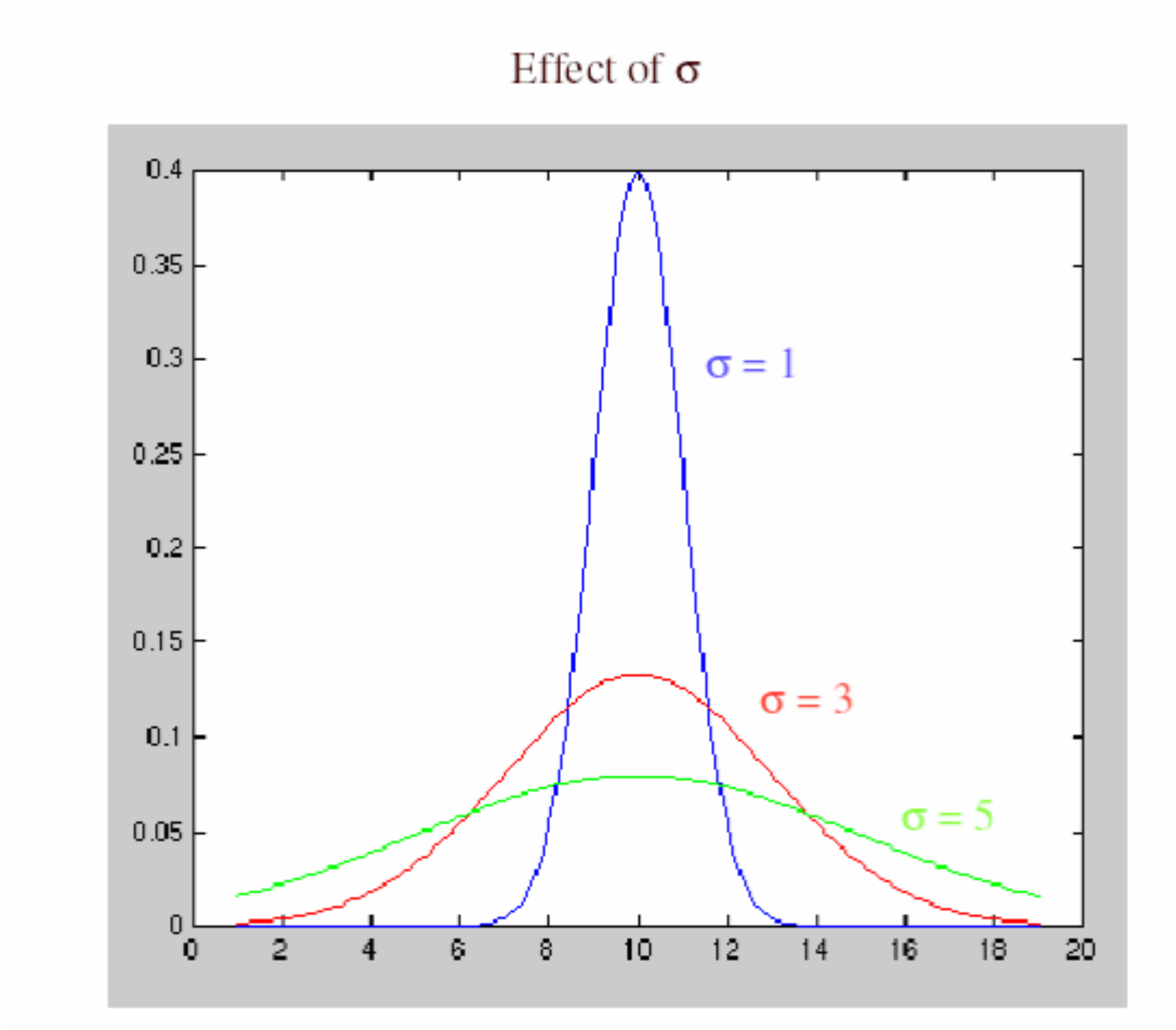
- The Gaussian function has infinite support, but discrete filters use finite kernels



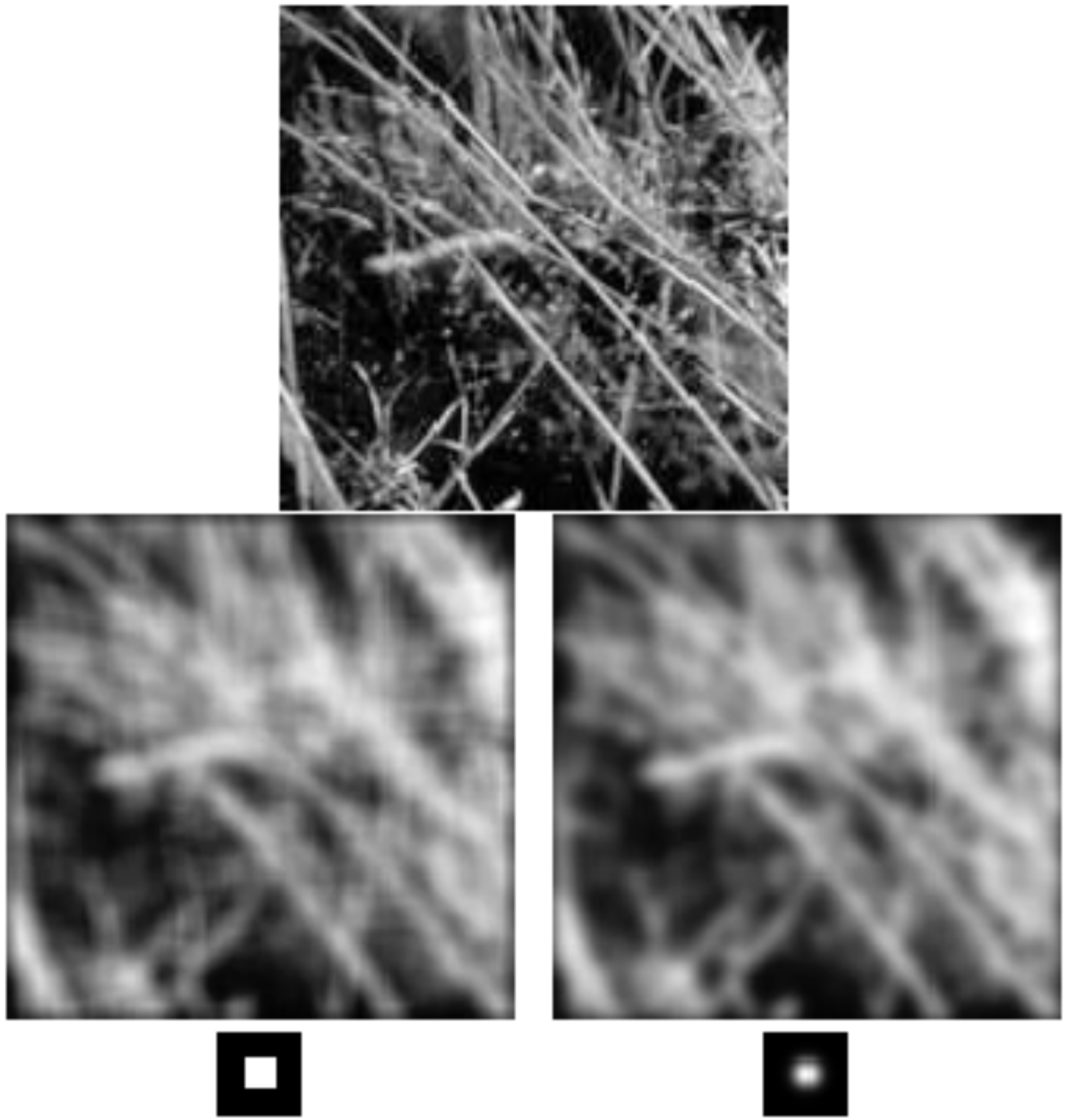
Source: K. Grauman

Choosing kernel width

Rule of thumb: set filter half-width to about 3σ



Gaussian vs. box filtering



Source: S. Lazebnik

Gaussian filters

- Convolution with self is another Gaussian
  - So can smooth with small-σ kernel, repeat, and get same result as larger-σ kernel would have
  - Convolving two times with Gaussian kernel with std. dev. σ is same as convolving once with kernel with std. dev. $\sigma\sqrt{2}$
- Separable kernel
  - Factors into product of two 1D Gaussians
  - Discrete example:

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

Source: K. Grauman

Separability of the Gaussian filter

$$
\begin{aligned}
G_\sigma(x,y) &= \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{2\sigma^2}} \\
&= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)
\end{aligned}
$$

The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

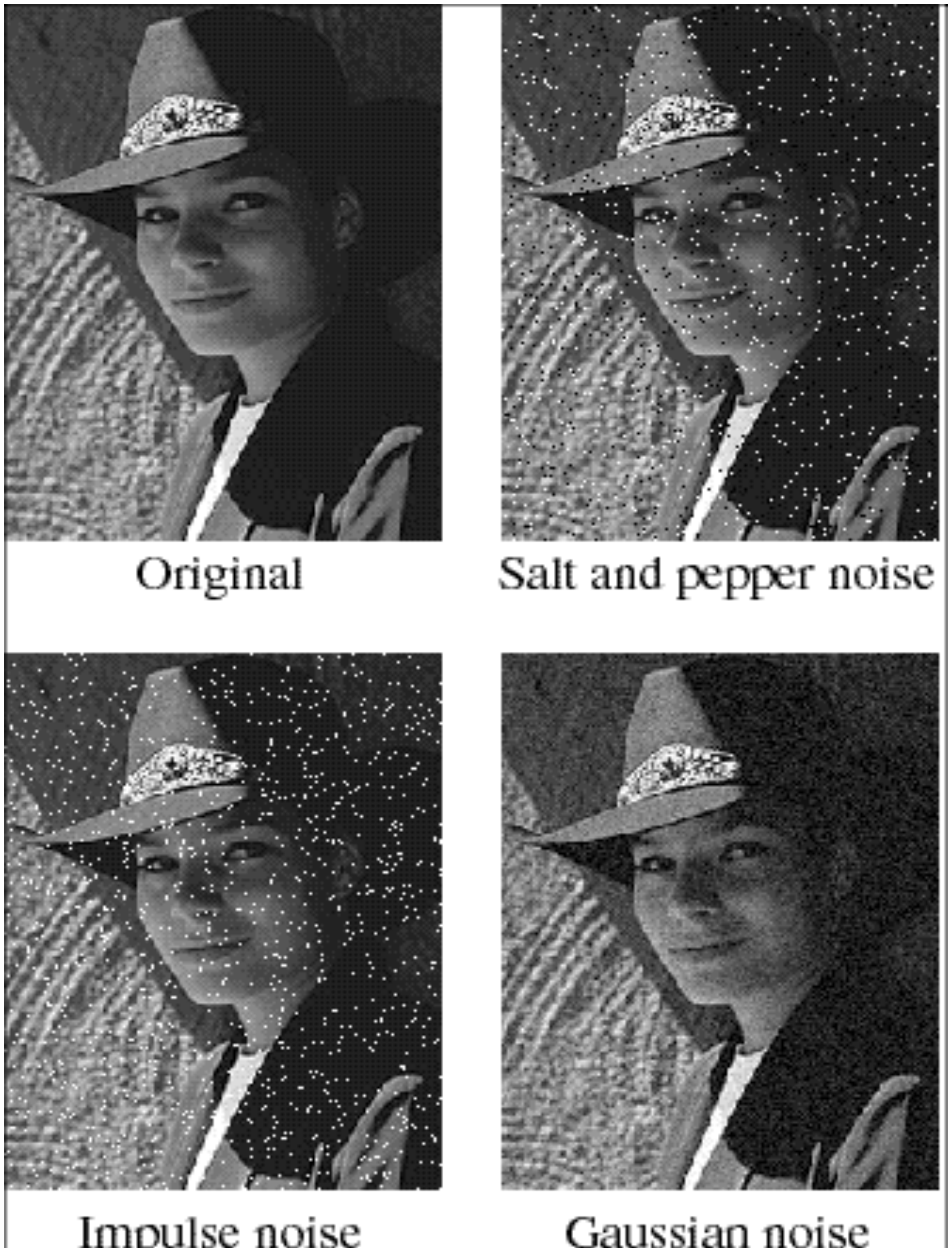
In this case, the two functions are the (identical) 1D Gaussian

Source: D. Lowe

Why is separability useful?

- Separability means that a 2D convolution can be reduced to two 1D convolutions (one along rows and one along columns)
- What is the complexity of filtering an $n \times n$ image with an $m \times m$ kernel?
  - $O(n^2 m^2)$
- What if the kernel is separable?
  - $O(n^2 m)$

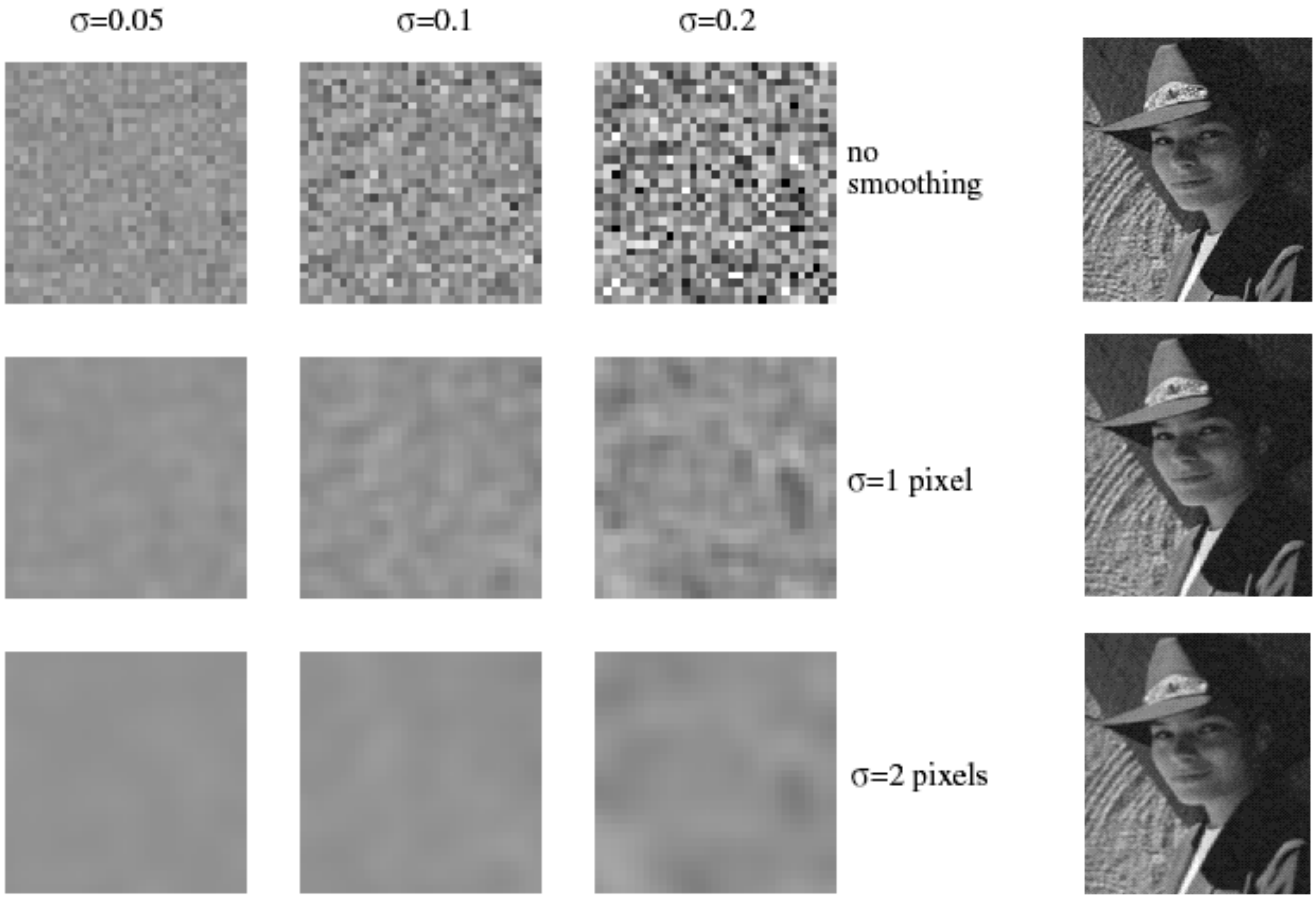
Noise



- **Salt and pepper noise:** contains random occurrences of black and white pixels
- **Impulse noise:** contains random occurrences of white pixels
- **Gaussian noise:** variations in intensity drawn from a Gaussian normal distribution

Source: S. Seitz

Reducing Gaussian noise



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

Source: S. Lazebnik

Reducing salt-and-pepper noise

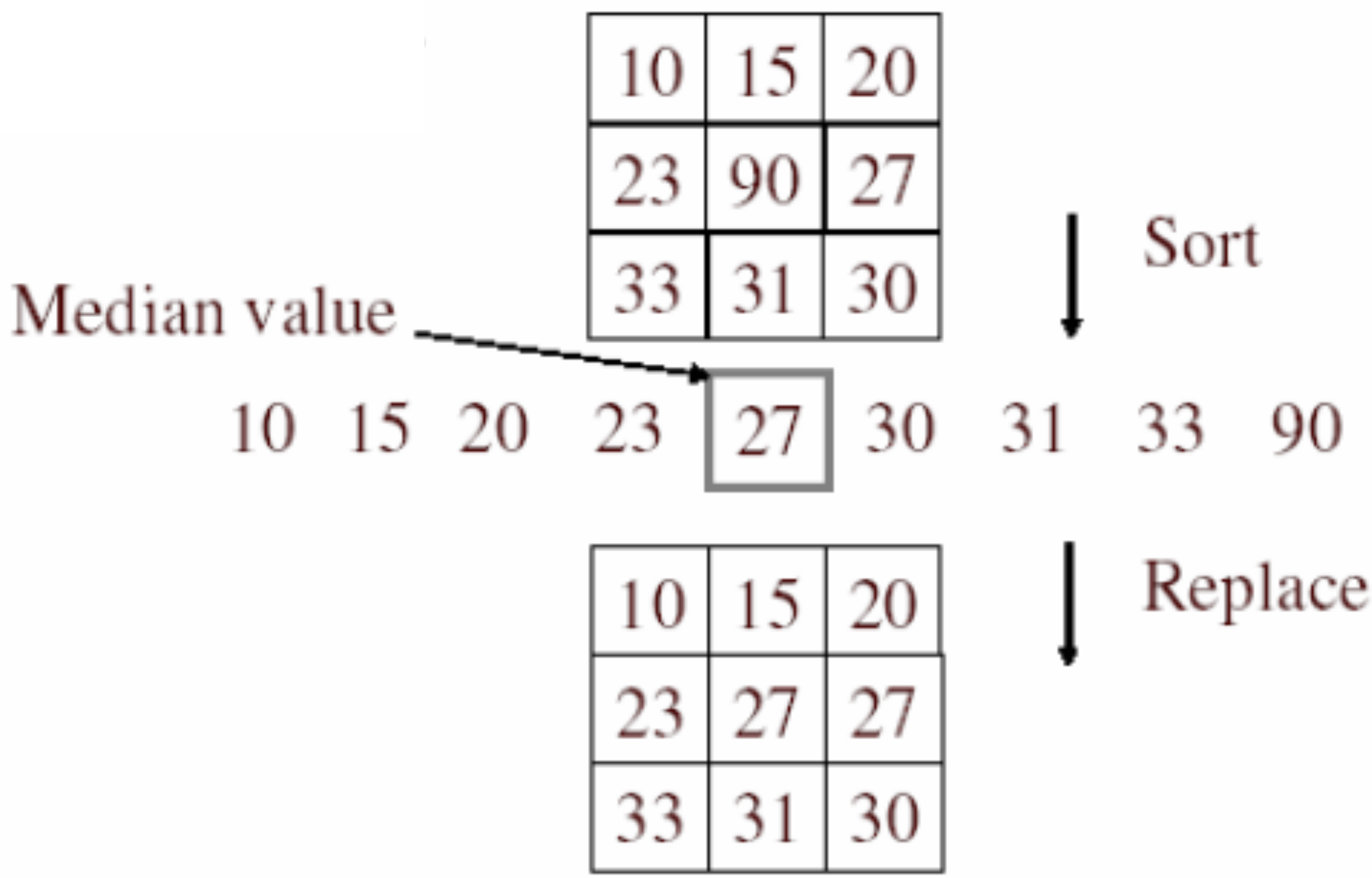
What's wrong with the results?

3x3 5x5 7x7

Source: S. Lazebnik

Alternative idea: Median filtering

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



Source: K. Grauman

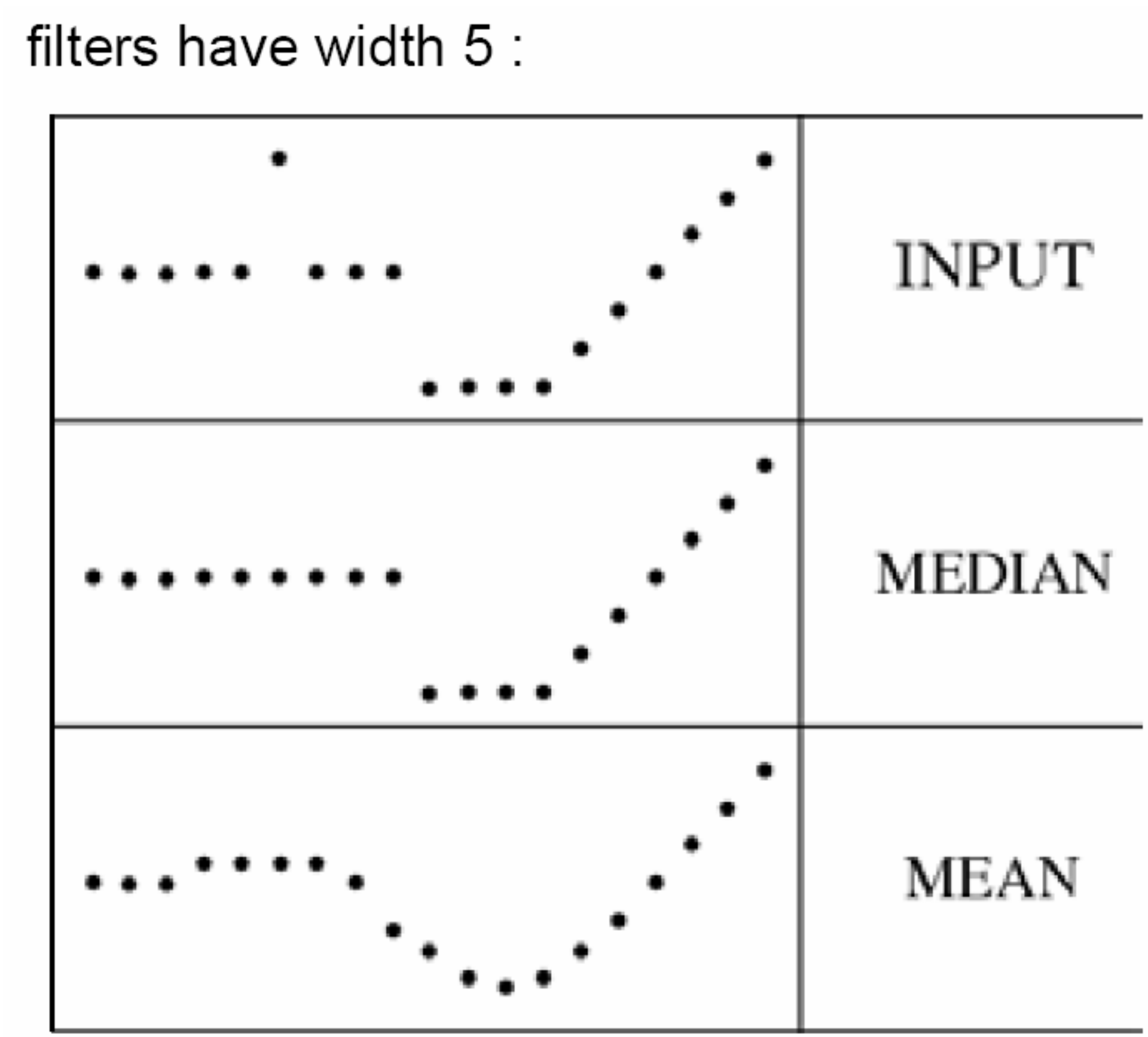
Median filter

- Is median filtering linear?
- Let's try filtering

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 2 & 2 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Median filter

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



Source: K. Grauman

Gaussian vs. median filtering

3x3 5x5 7x7

Gaussian

Median

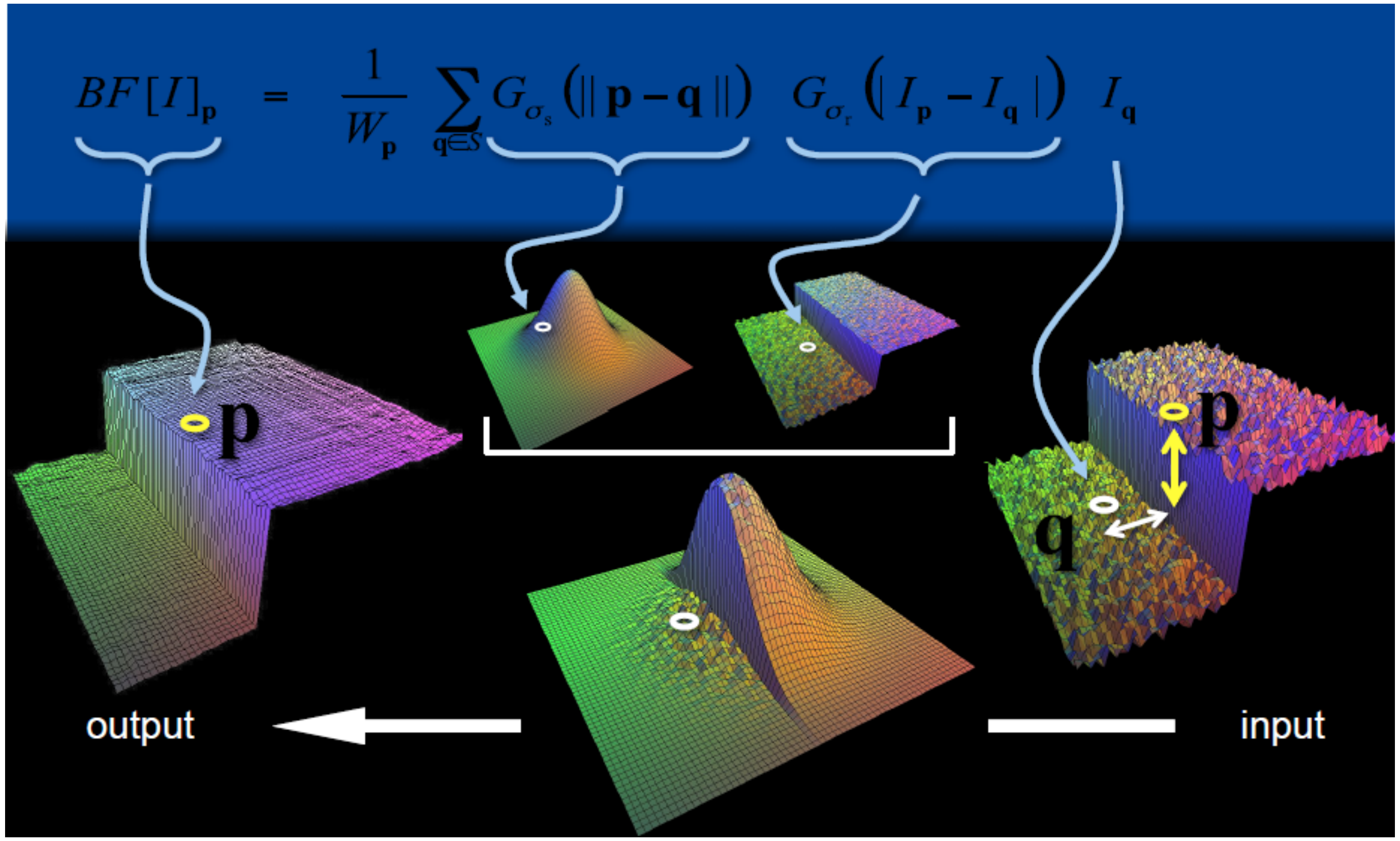
Source: S. Lazebnik

Other non-linear filters

- Weighted median (pixels further from center count less)
- Clipped mean (average, ignoring few brightest and darkest pixels)
- Max or min filter
- Bilateral filtering: to avoid blurring edges, only average with similar intensity values.

$$
\begin{aligned}
I^{\mathrm{b}}_{\mathbf{p}} &= \frac{1}{W^{\mathrm{b}}_{\mathbf{p}}} \sum_{\mathbf{q} \in \mathcal{S}} G_{\sigma_{\mathrm{s}}}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_{\mathrm{r}}}(|I_{\mathbf{p}} - I_{\mathbf{q}}|)\, I_{\mathbf{q}} \\
\text{with} \quad W^{\mathrm{b}}_{\mathbf{p}} &= \sum_{\mathbf{q} \in \mathcal{S}} G_{\sigma_{\mathrm{s}}}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_{\mathrm{r}}}(|I_{\mathbf{p}} - I_{\mathbf{q}}|)
\end{aligned}
$$

Bilateral filters



Border effects

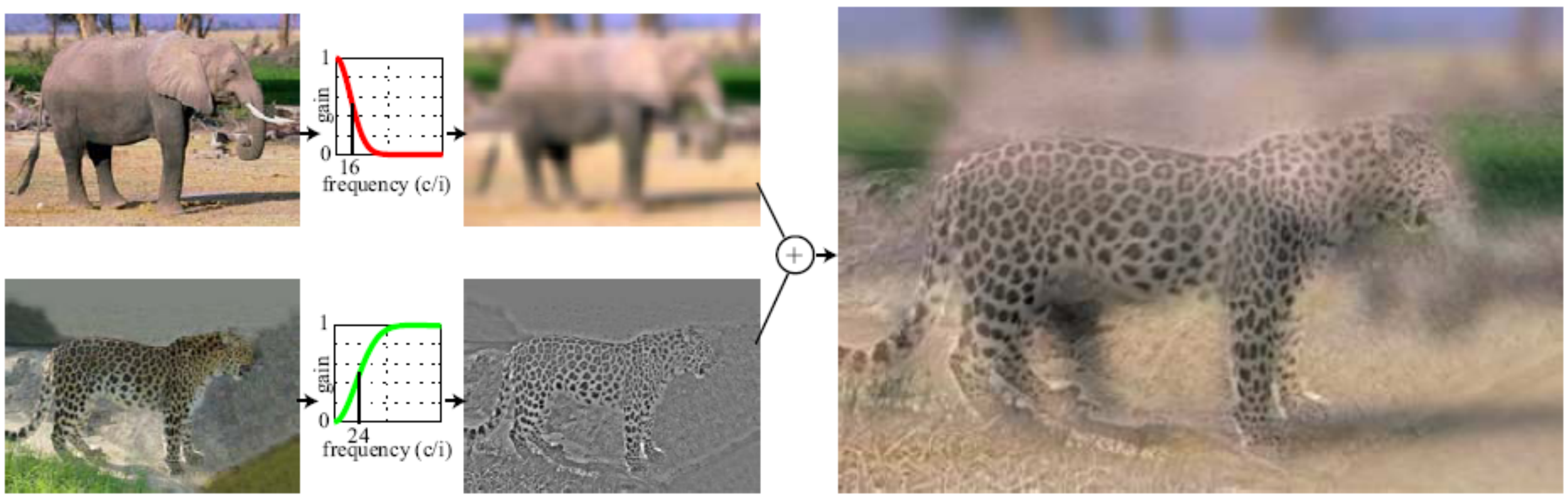
- What about near the edge? need to extrapolate!

  – Methods:
    - clip filter (black)
    - wrap around
    - copy edge
    - reflect across edge

Source: S. Marschner

Images in frequency domain

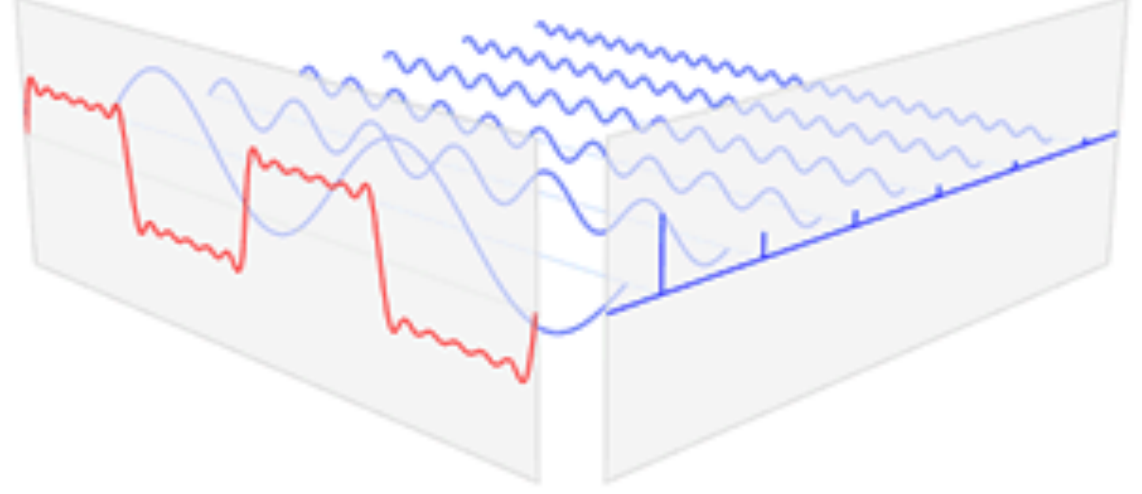
- Wide range of applications: image analysis, image filtering, image reconstruction, and image compression.



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

Fourier analysis

- Joseph Fourier: “Any function is a weighted combination of sines and cosines.”



https://youtu.be/-qgreAUpPwM?t=302

The Fourier transform

- Continuous transform:

$$f : \mathbb{R} \to \mathbb{R}, \qquad \hat{f}(\xi) = \int_{-\infty}^{+\infty} f(x)\ e^{-ix\xi}\, dx\,.$$

- Discrete transform:

$$f : [0, N-1] \to \mathbb{R}, \qquad \hat{f}(k) = \frac{1}{N} \sum_{m=0}^{N-1} f(m)\ e^{-\frac{2\pi i}{N}km}\,.$$

- Intuition: $\hat{f}$ collects coefficients in the representation of f in Fourier basis

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \hat{f}(\xi) e^{i\xi x} d\xi, \qquad f(m) = \sum_{k=0}^{N-1} \hat{f}(k) e^{\frac{2\pi i}{N}km}\,.$$

Some useful properties

- f discrete $\Leftrightarrow \hat{f}$ periodic
- $\hat{f}$ real $\Leftrightarrow f(x) = f(-x)$
- Convolution Theorem: $(\widehat{f \star g}) = \hat{f} \cdot \hat{g}$
- Differentiation: $(\widehat{f'})(\xi) = i\xi\hat{f}(\xi)$
- Energy conservation: $\int |f(x)|^2 dx = \int |\hat{f}(\xi)|^2 d\xi$

Discrete 2D transform

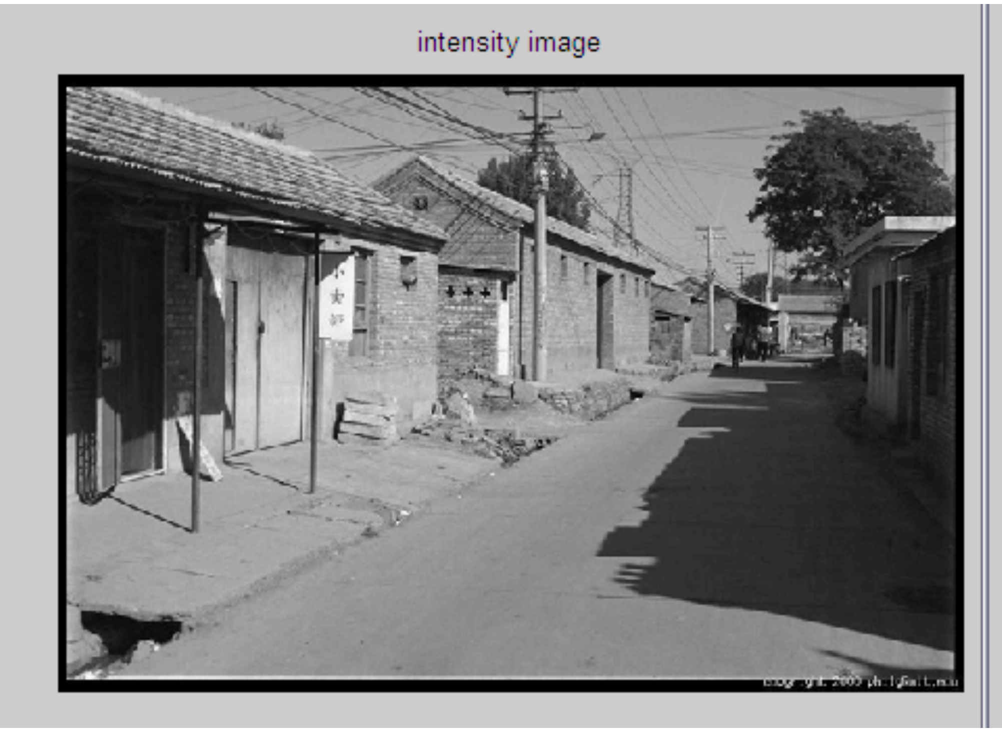
- Transforms an $M \times N$ pixel grid $f(x, y)$ into an $M \times N$ grid of complex numbers $H(k_x, k_y)$:

$$H(k_x, k_y) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-i2\pi \frac{2\pi}{MN}(k_x x + k_y y)}$$

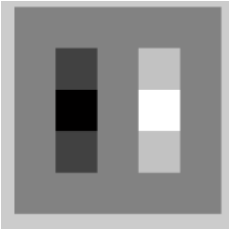
- Usually, the log-magnitude at every pixel is usually plotted.
- Pixels in the center of the transform correspond to low frequencies/long wavelengths.

Filtering in spatial domain

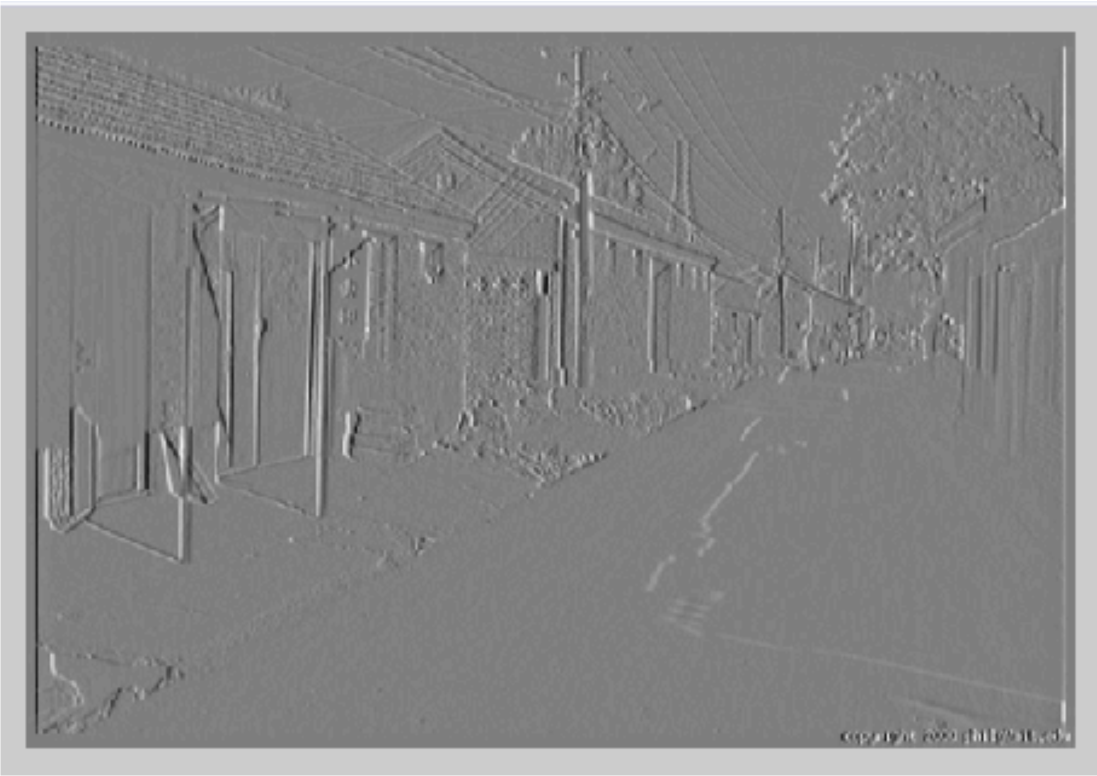
1	0	-1
2	0	-2
1	0	-1



*

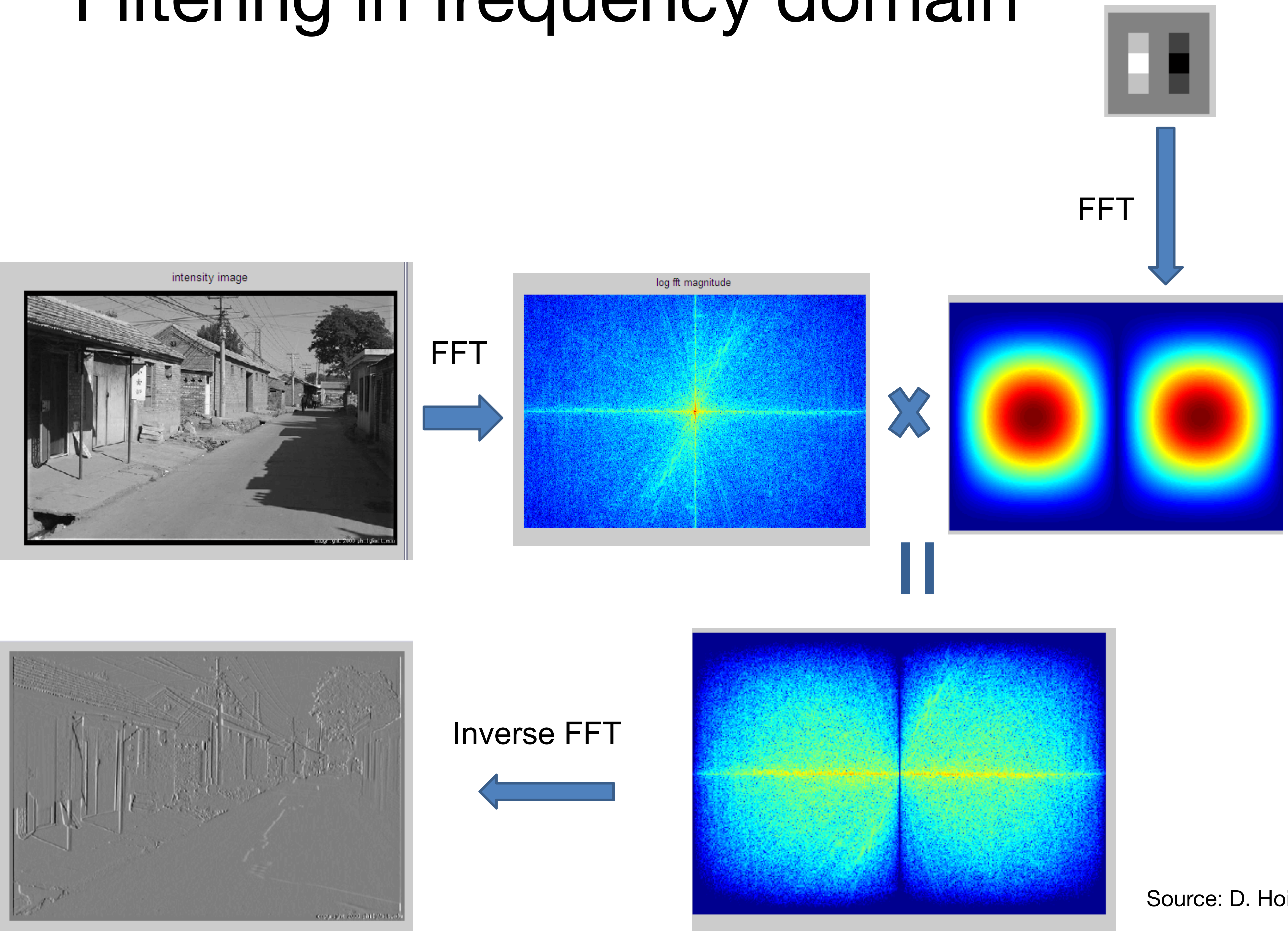


=



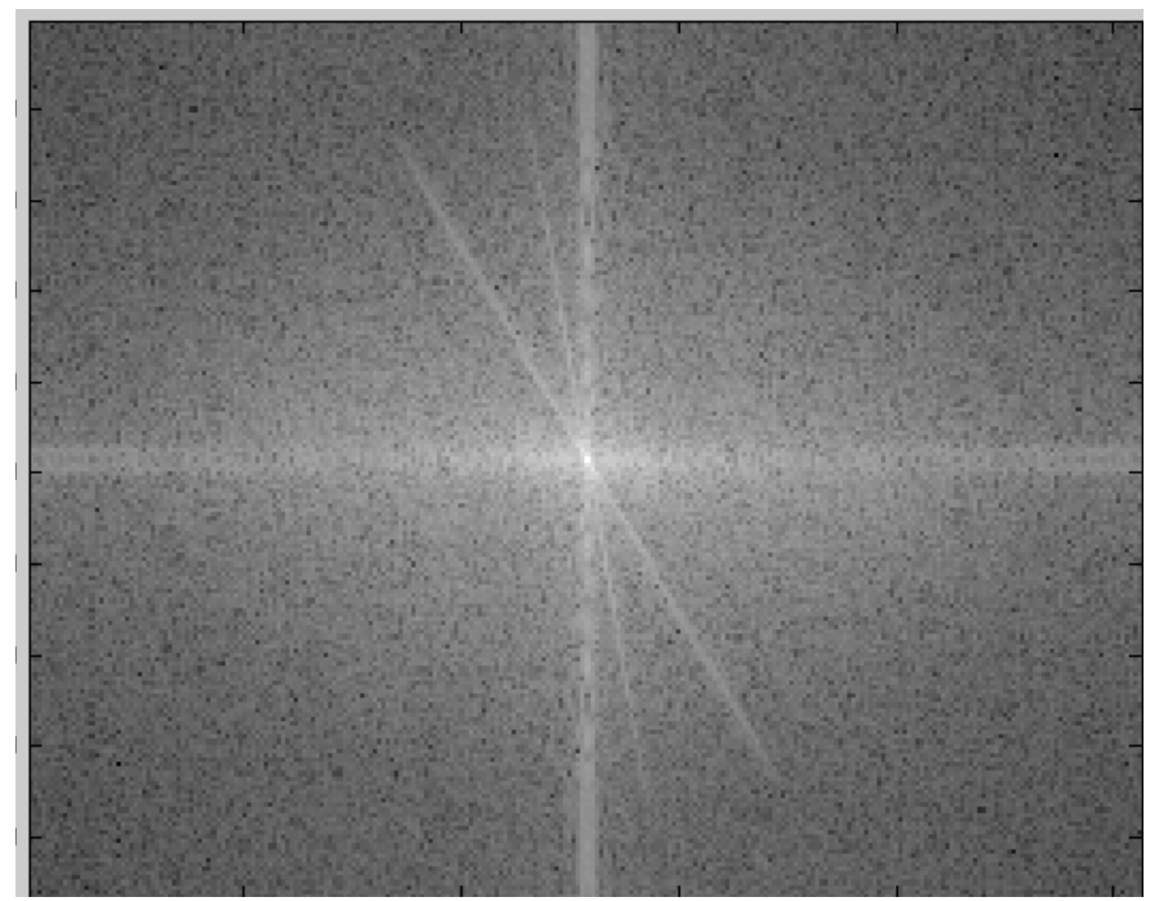
Source: D. Hoiem

Filtering in frequency domain



Source: D. Hoiem

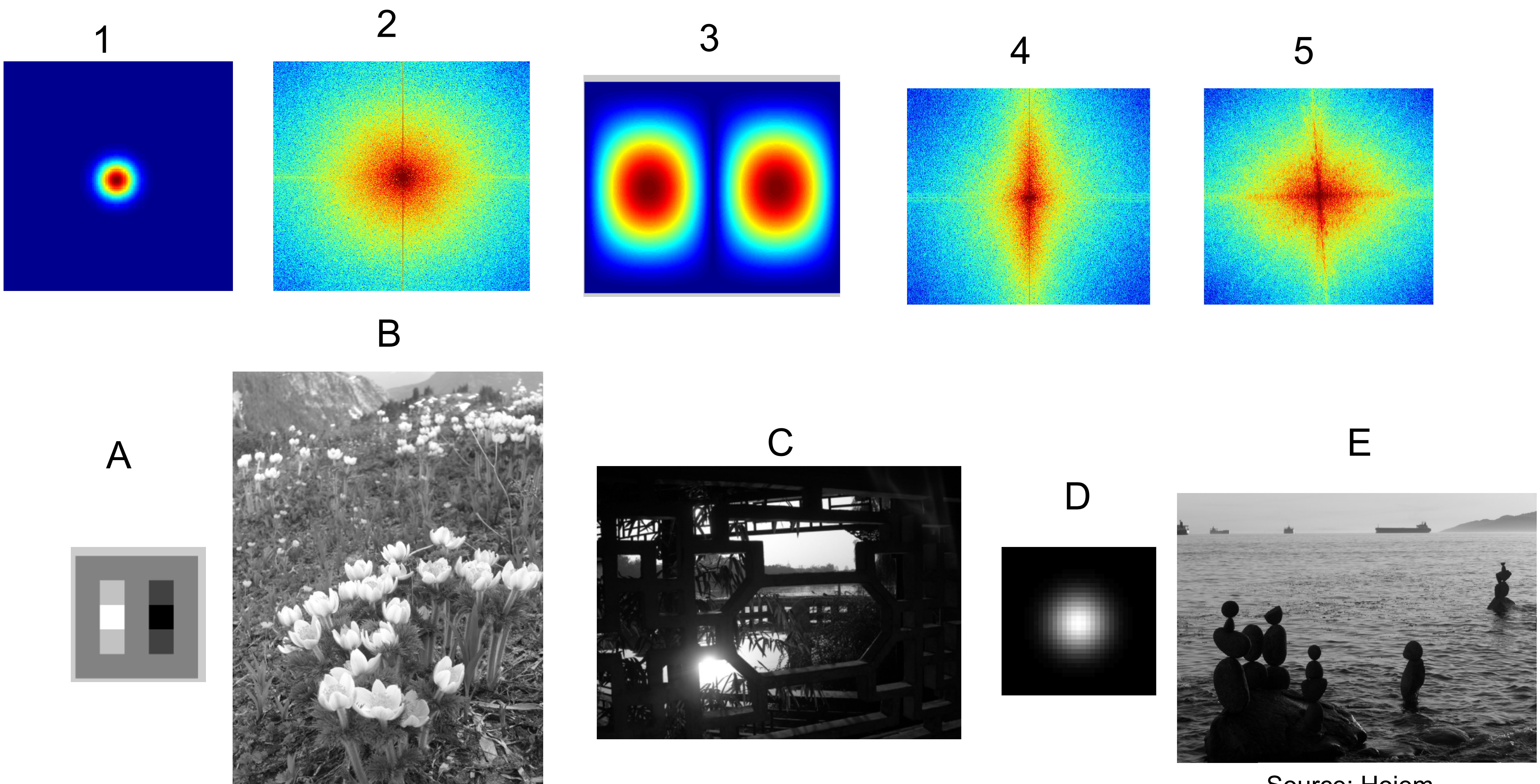
Fourier transform of a scene



Source: J. Hays

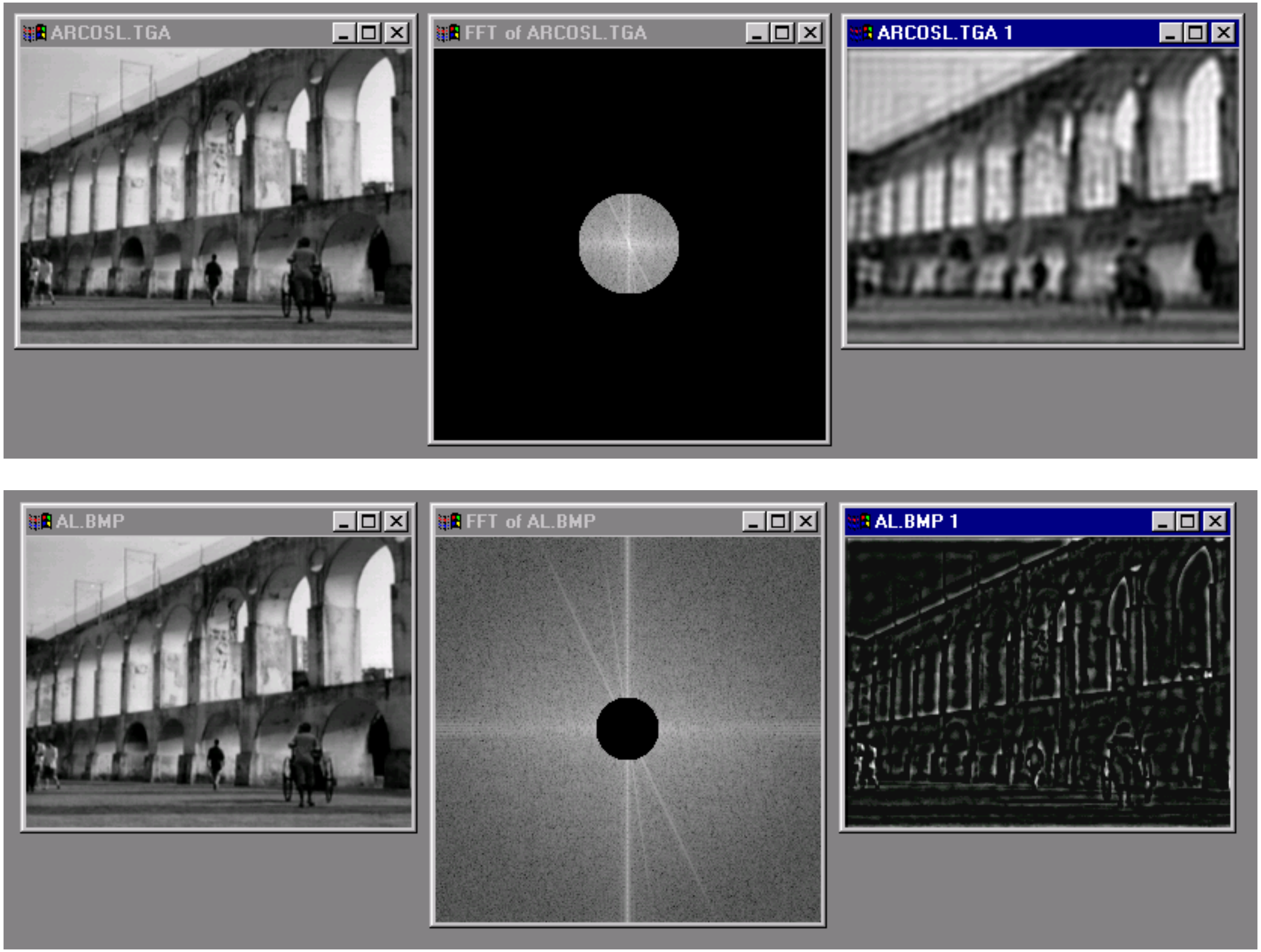
Question

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



Source: Hoiem

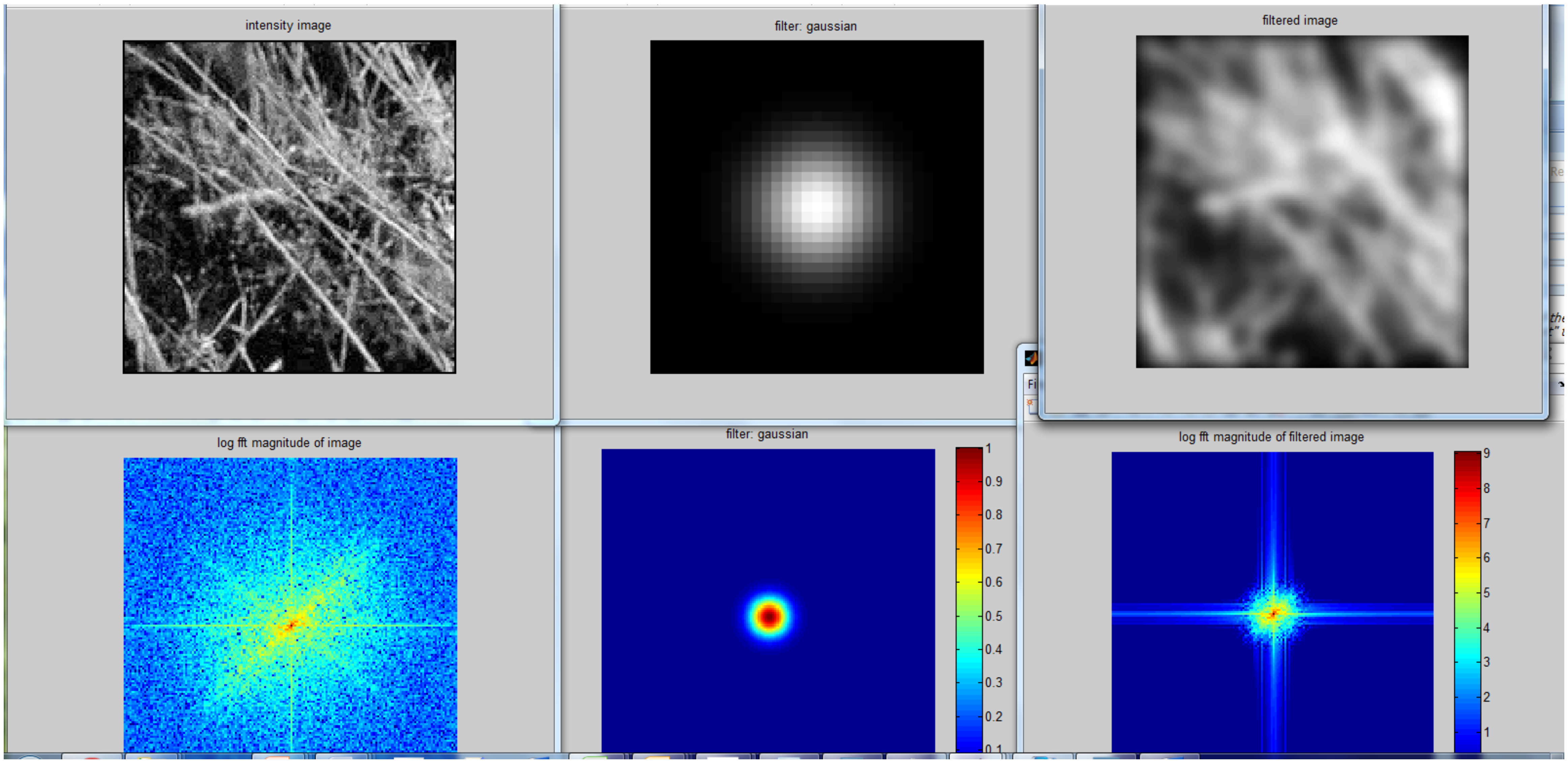
Low and high pass filtering



Source: J. Hays

Filters in frequency domain

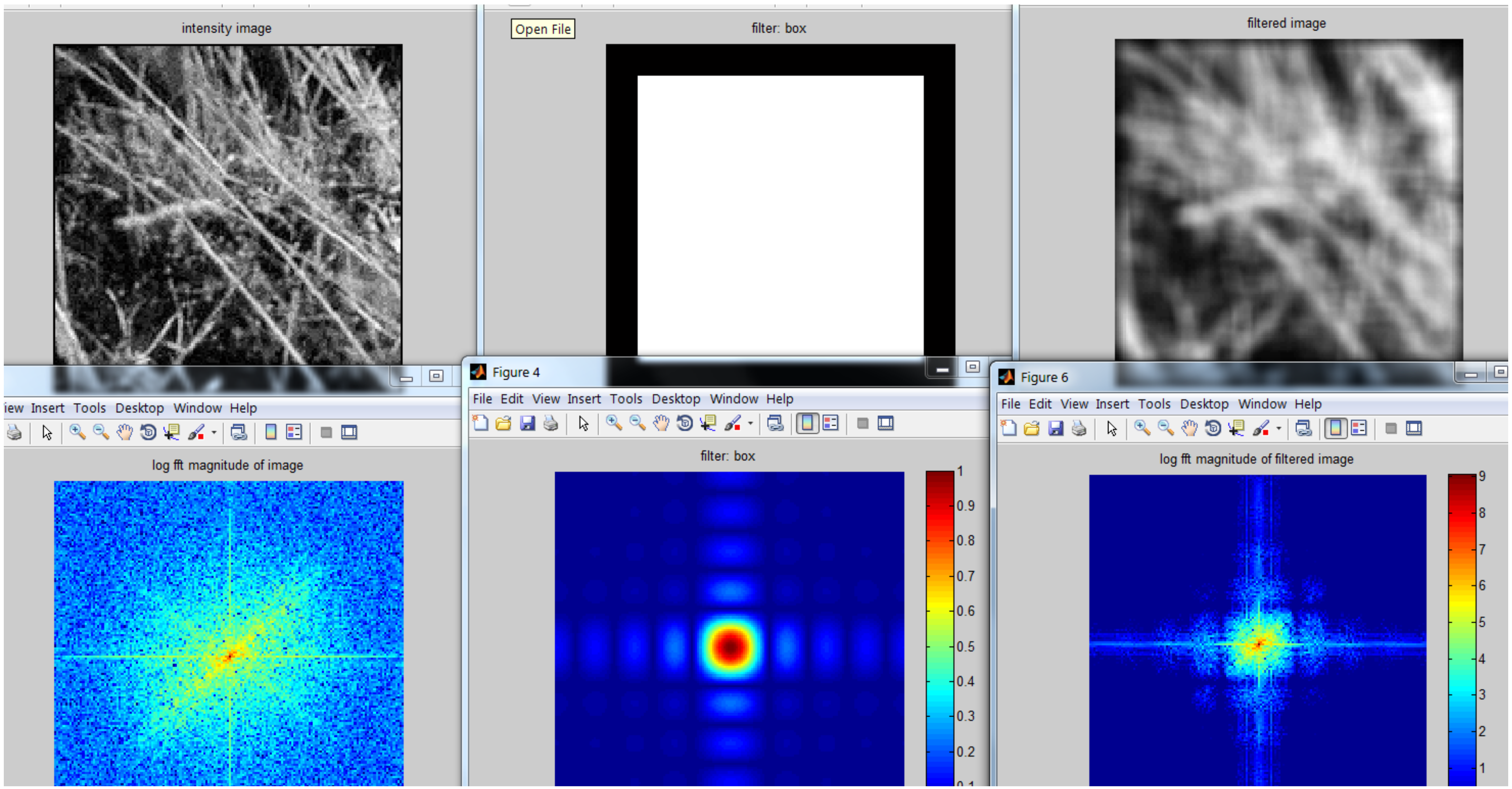
Gaussian



Source: J. Hays

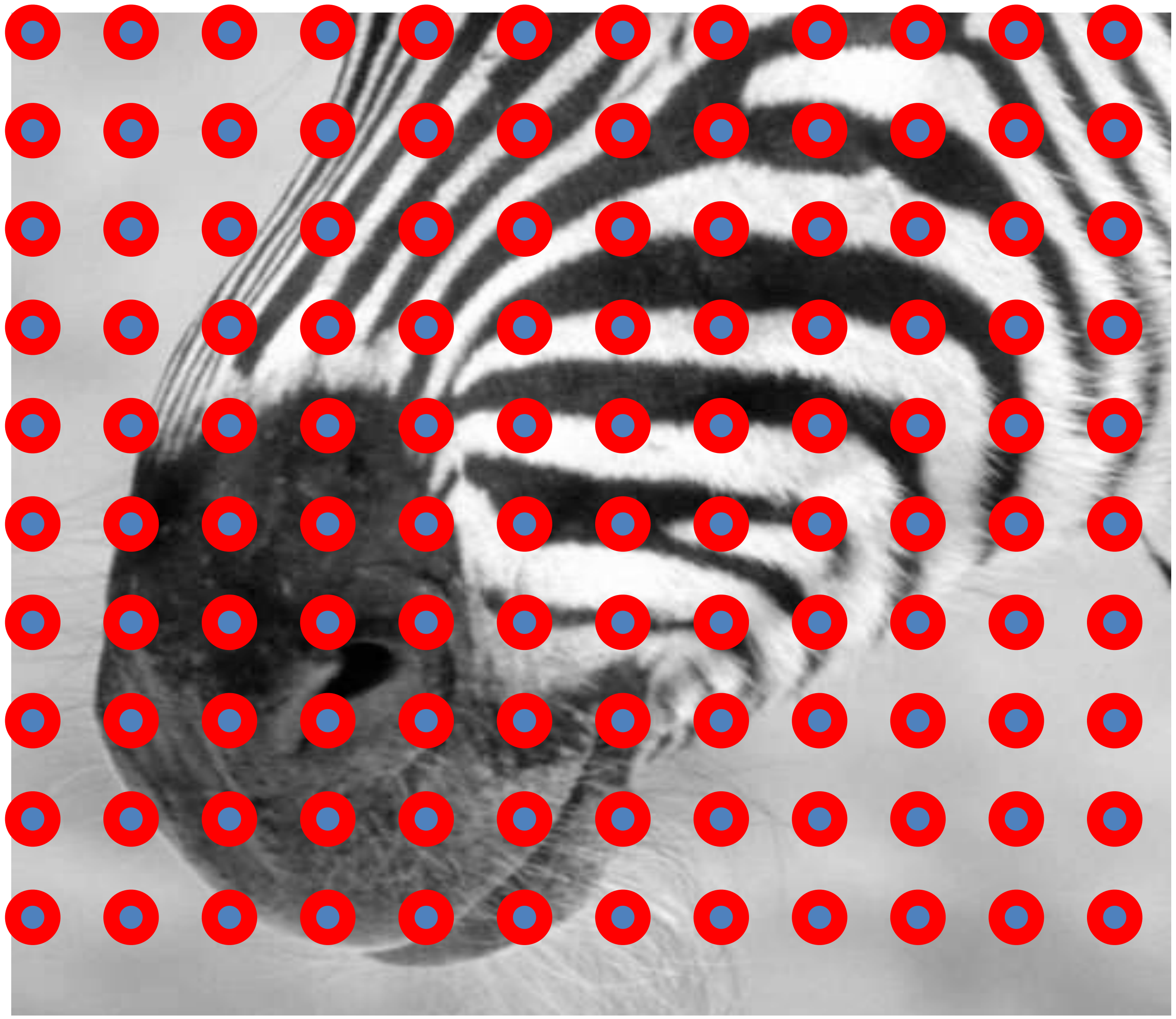
Filters in frequency domain

Box



Source: J. Hays

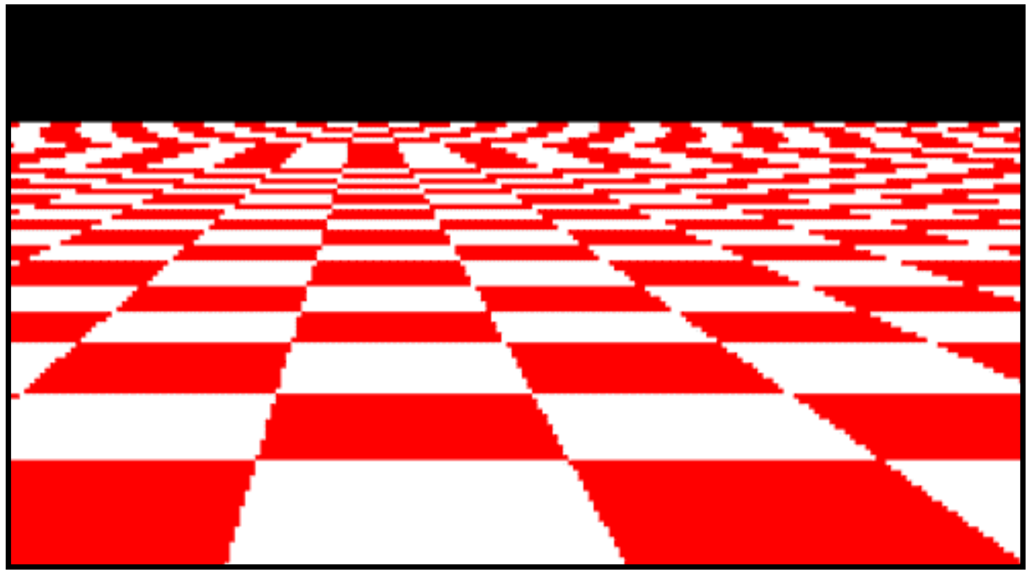
Subsampling by a factor of 2



Throw away every other row and column to create a 1/2 size image

Source: D. Hoiem

Aliasing

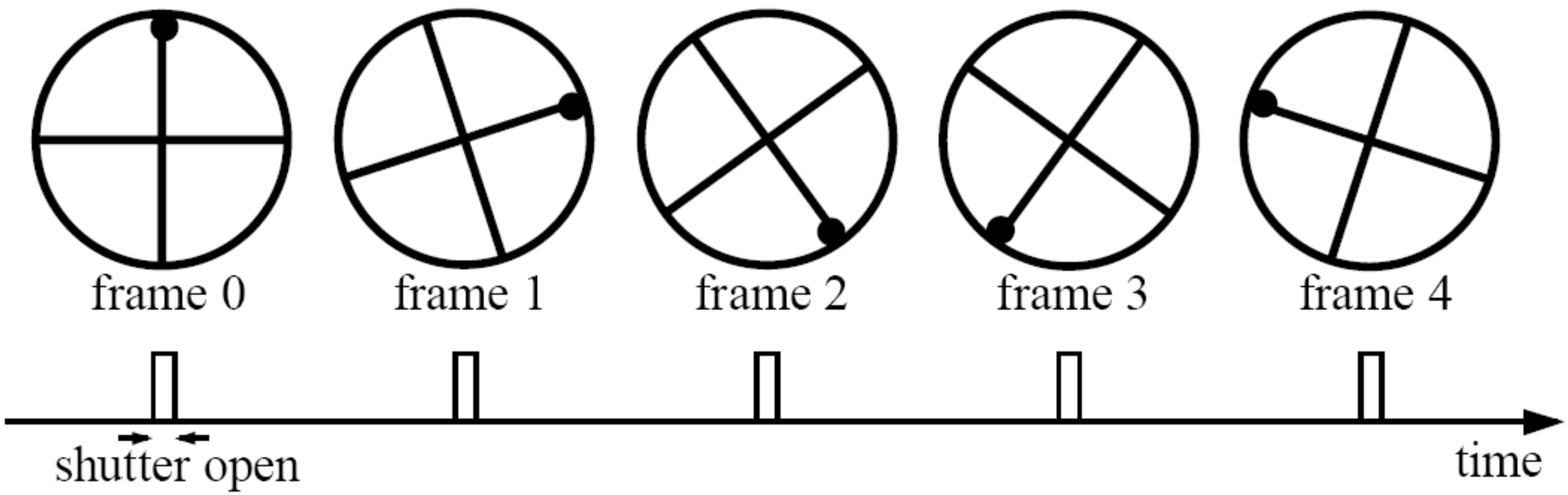


- Sub-sampling may be dangerous!
- Characteristic errors may appear:
  - Wagon wheels rolling the wrong way in movies
  - Checkerboards disintegrate in ray tracing
  - Striped shirts look funny on color television

Aliasing in video

Imagine a spoked wheel moving to the right (rotating clockwise). Mark wheel with dot so we can see what's happening.

If camera shutter is only open for a fraction of a frame time (frame time = 1/30 sec. for video, 1/24 sec. for film):

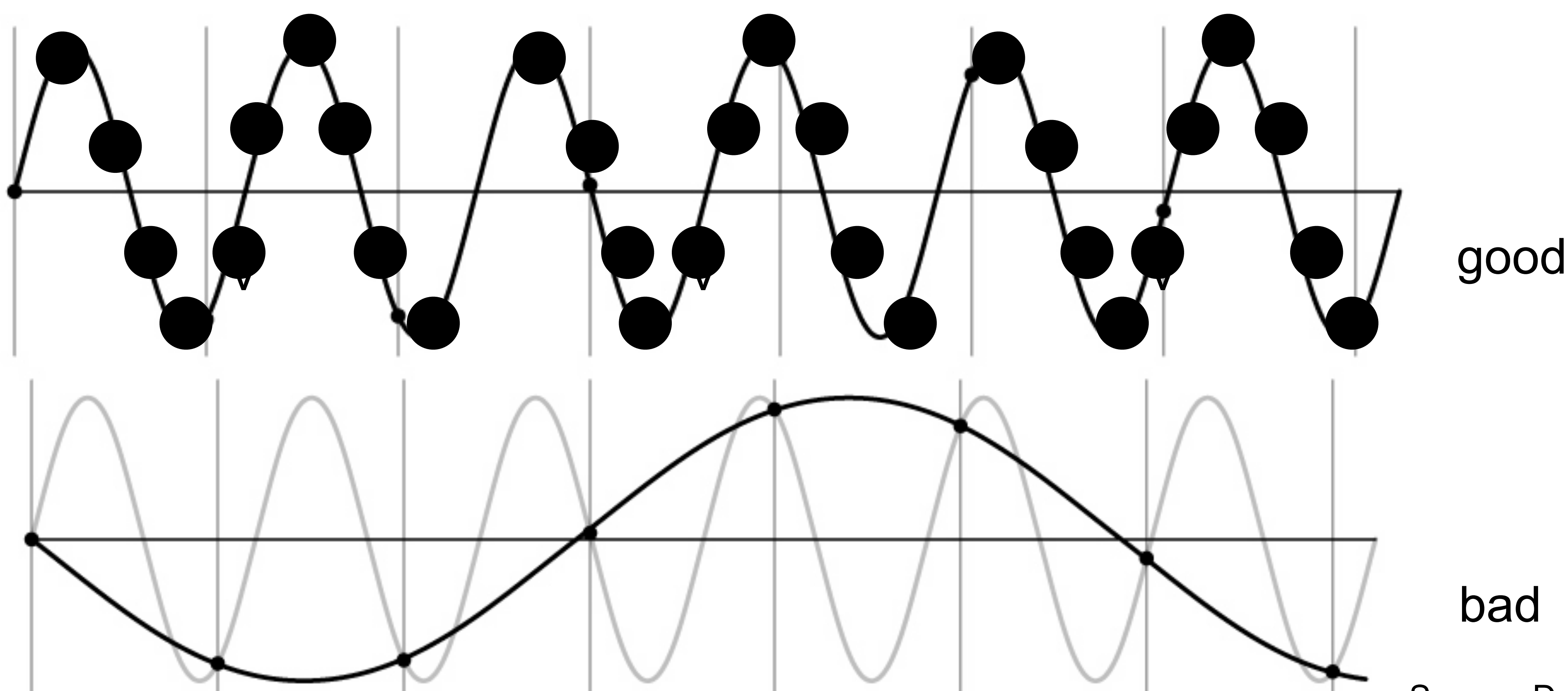


Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

Slide by Steve Seitz

How to sample

- When sampling a signal at discrete intervals, the sampling frequency must be $\geq 2 \times f_{max}$
- f_{max} = max frequency of the input signal
- This will allows to reconstruct the original perfectly from the sampled version



Source: D. Hoiem

Anti-aliasing

Solutions:

- Sample more often

- Get rid of all frequencies that are greater than half the new sampling frequency
  - Will lose information
  - But it's better than aliasing
  - Apply a smoothing filter

Source: D. Hoiem

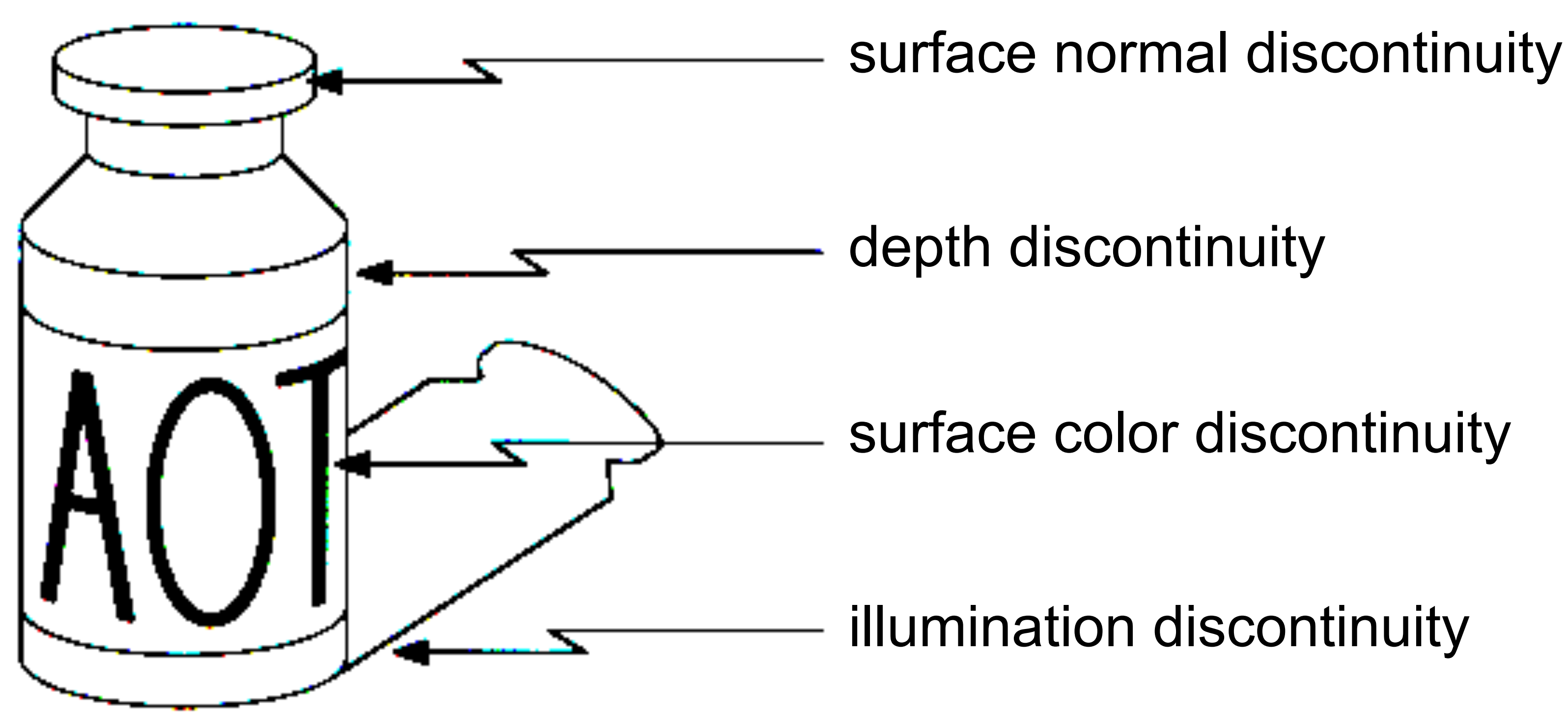
Summary

- Filters are useful tools for manipulating images (denoising, sharpening, etc.)
- Spacial domain: linear filters (box, Gaussian), median filter, bilateral filter.
- Frequency domain: high and low pass filtering, aliasing.

Edge detection

- **Goal:** Identify sudden changes (discontinuities) in an image
- Intuitively, edges carry most of the semantic and shape information from the image

Edges are caused by a variety of factors

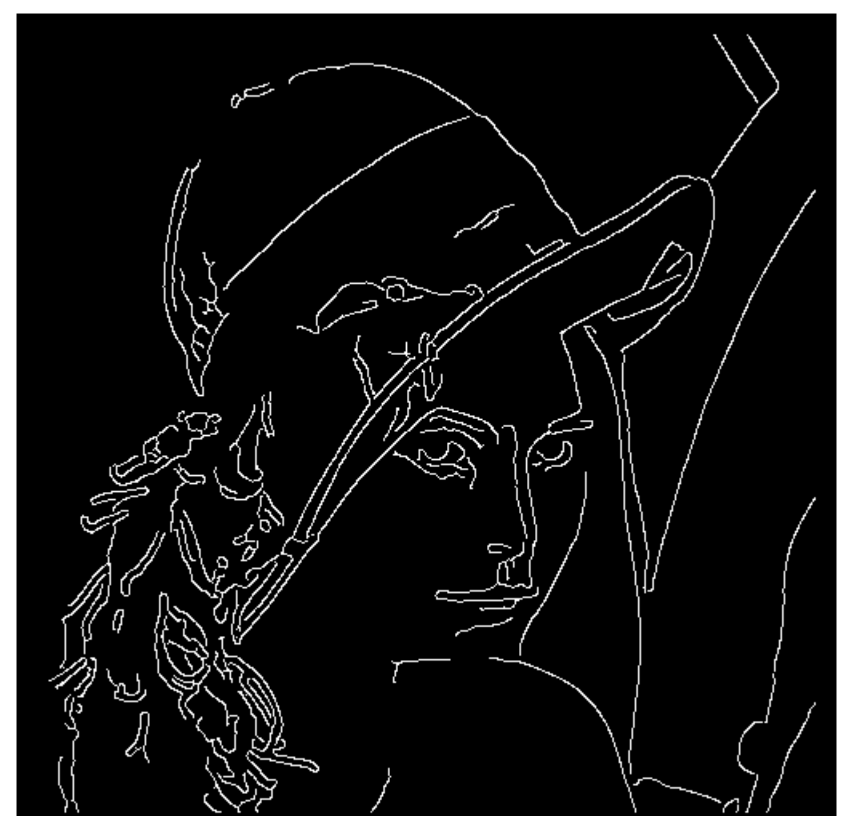


Sources: D. Lowe and S. Seitz

Edge detection

- **Ideal:** artist's line drawing

- **Reality:**

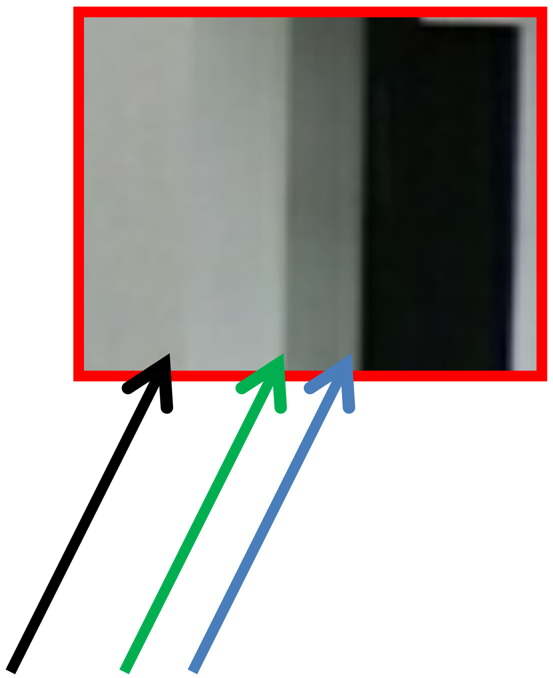


Source: S.Lazebnik

Closeup of edges

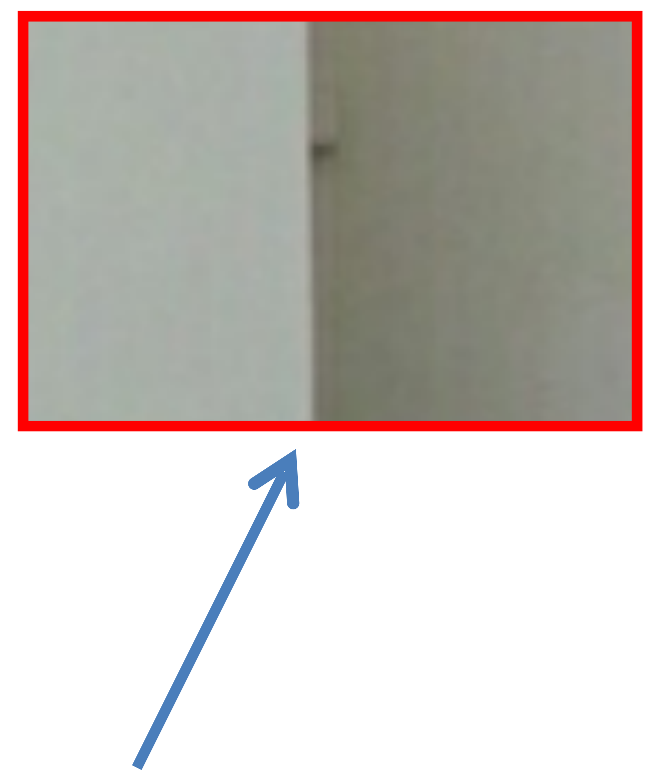
Source: D. Hoiem

Closeup of edges



Source: D. Hoiem

Closeup of edges



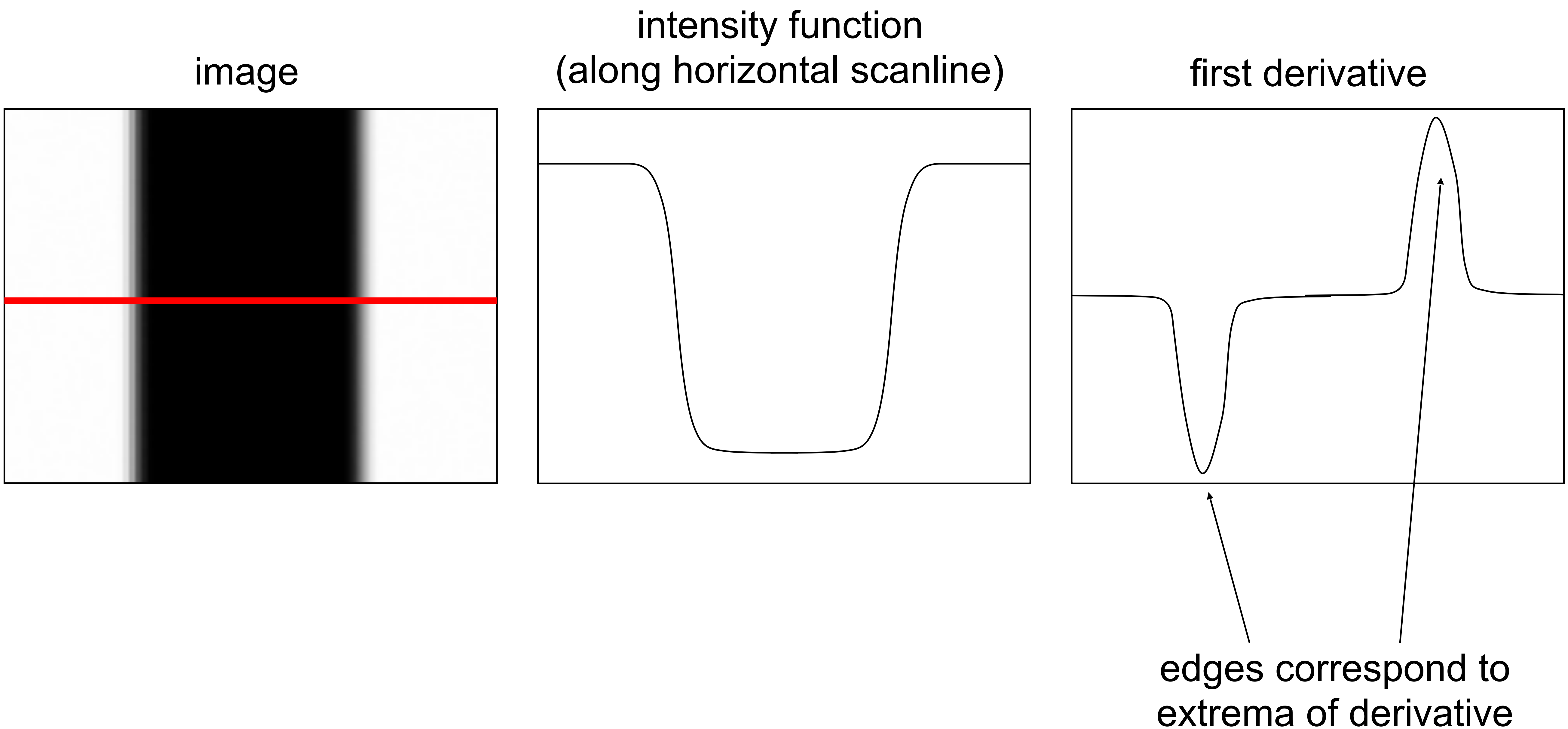
Source: D. Hoiem

Closeup of edges

Source: D. Hoiem

Edge detection

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

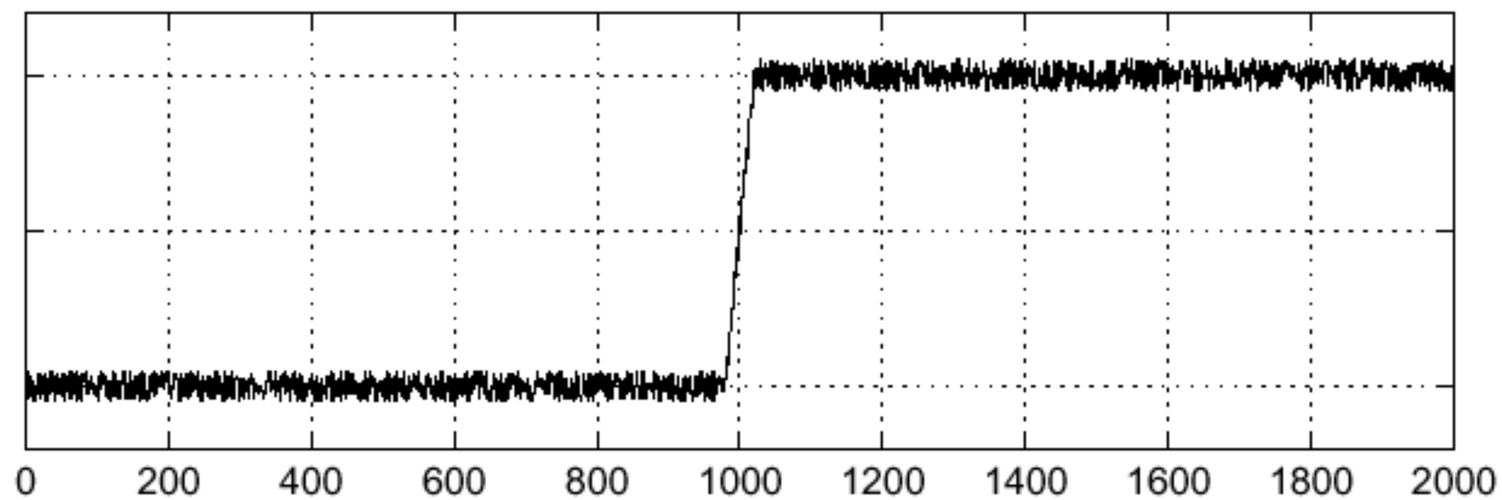


Source: S.Lazebnik

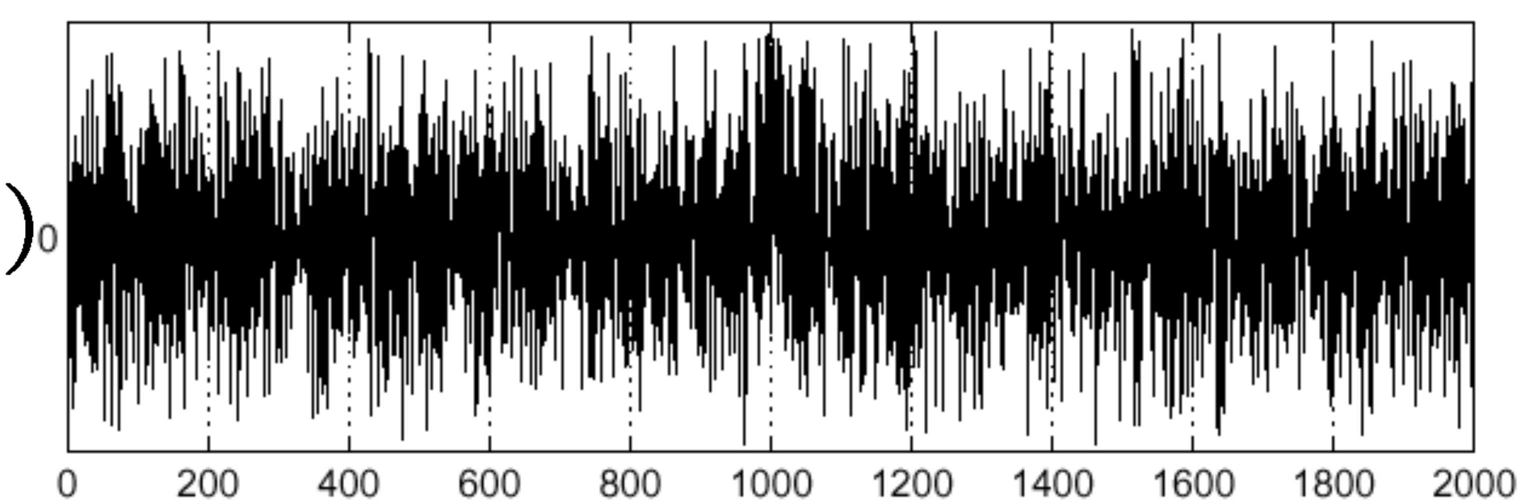
Effects of noise

Consider a single row or column of the image

$f(x)$



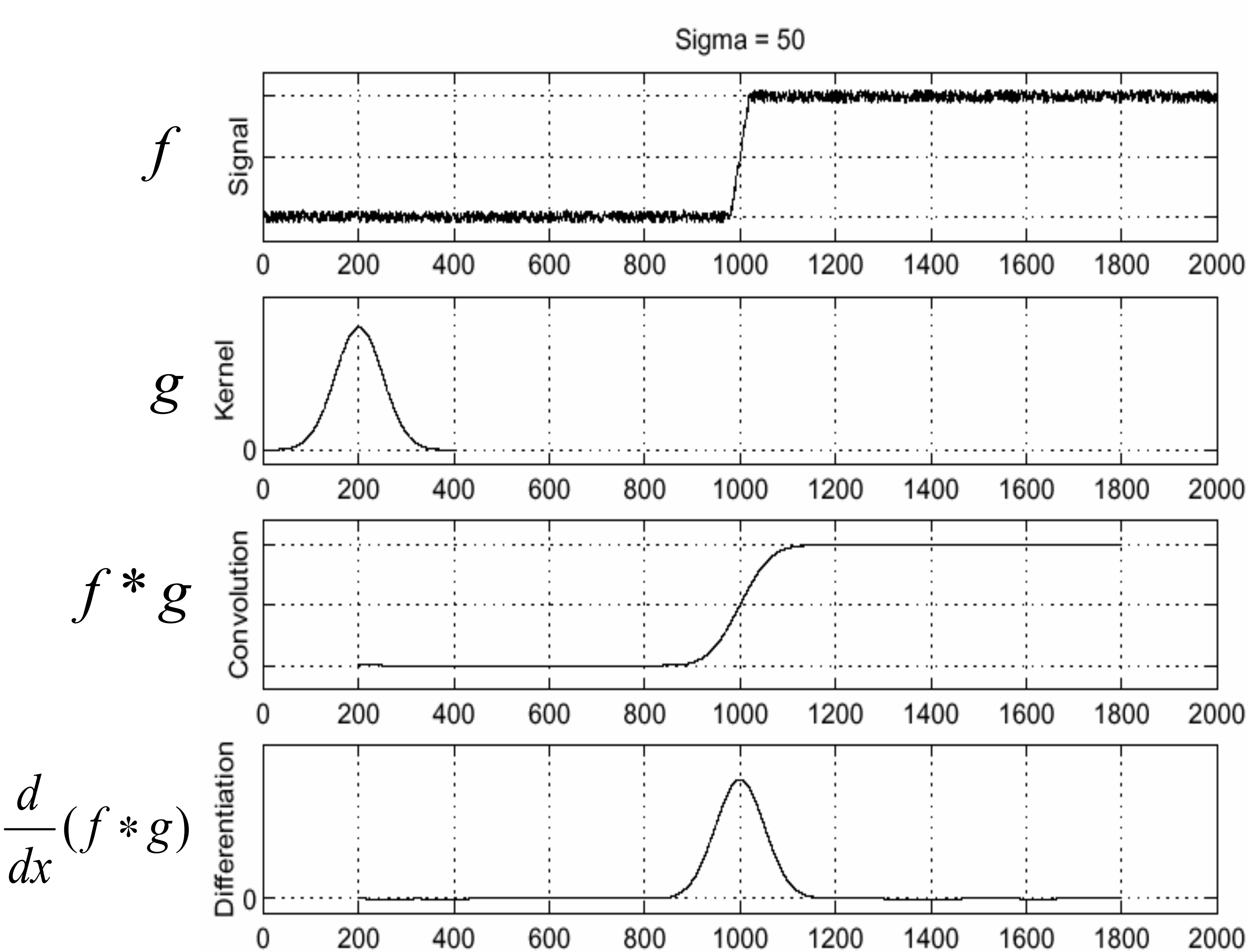
$\frac{d}{dx}f(x)$



Where is the edge?

Source: S. Seitz

Solution: smooth first

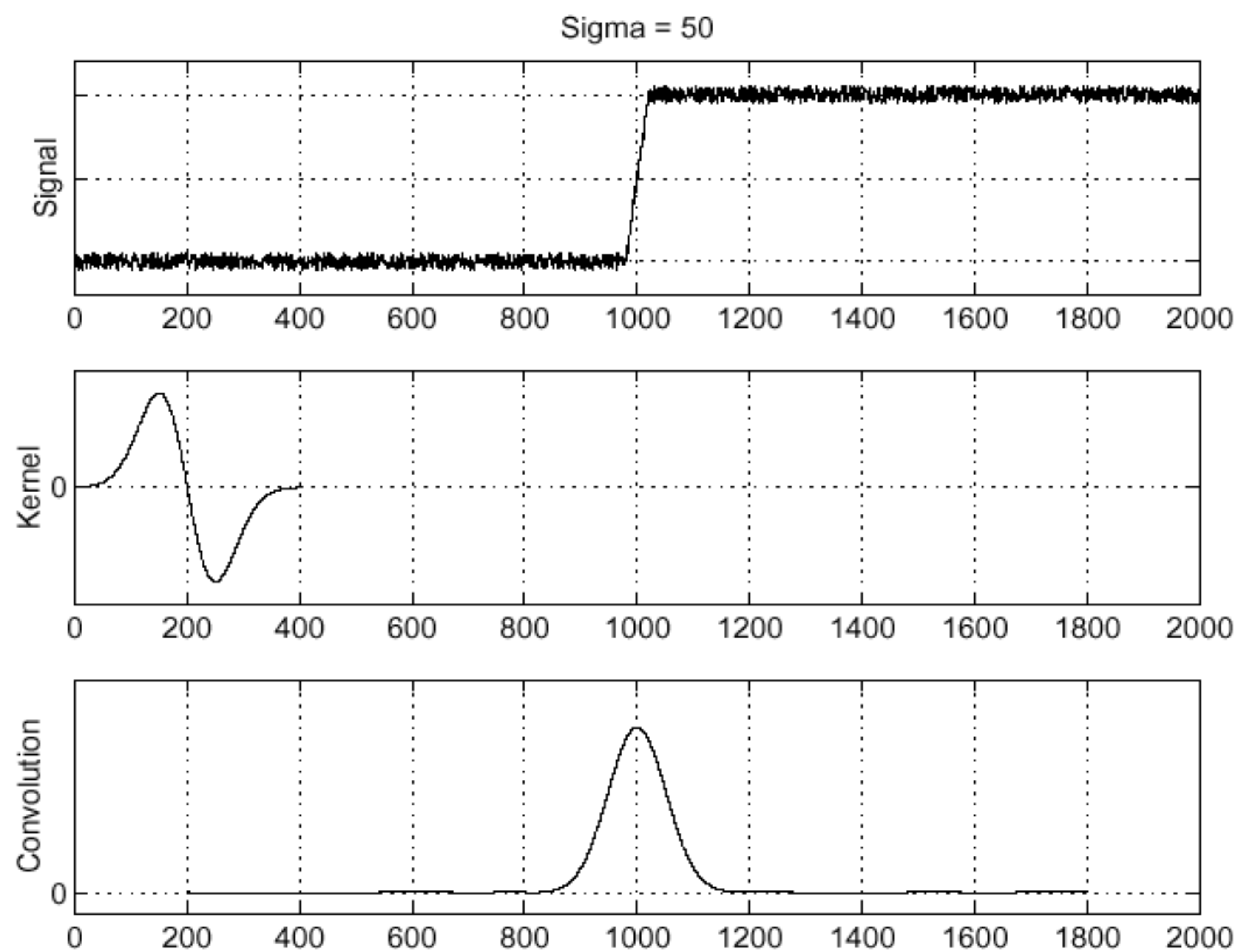


- To find edges, look for peaks in $\frac{d}{dx}(f * g)$

Source: S. Seitz

Derivative theorem of convolution

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



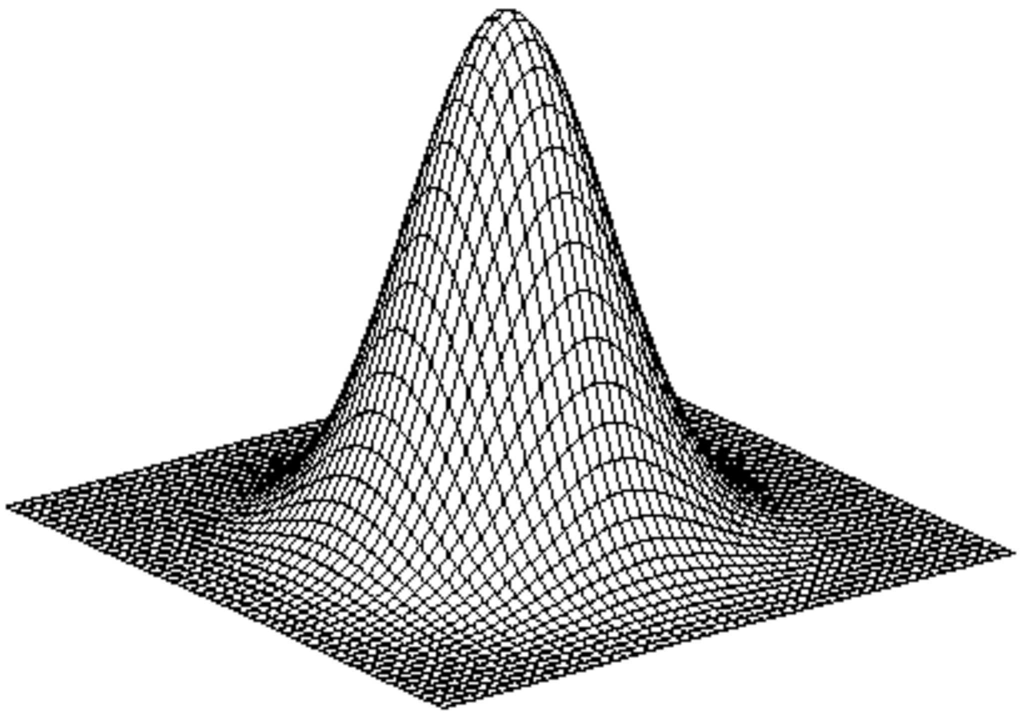
f

$\frac{d}{dx}g$

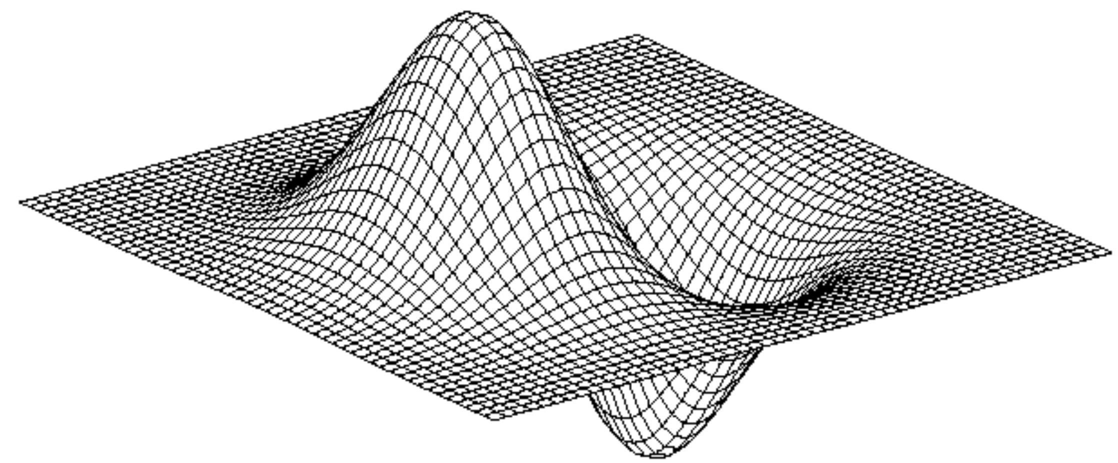
$f * \frac{d}{dx}g$

Source: S. Seitz

Derivative of Gaussian filter



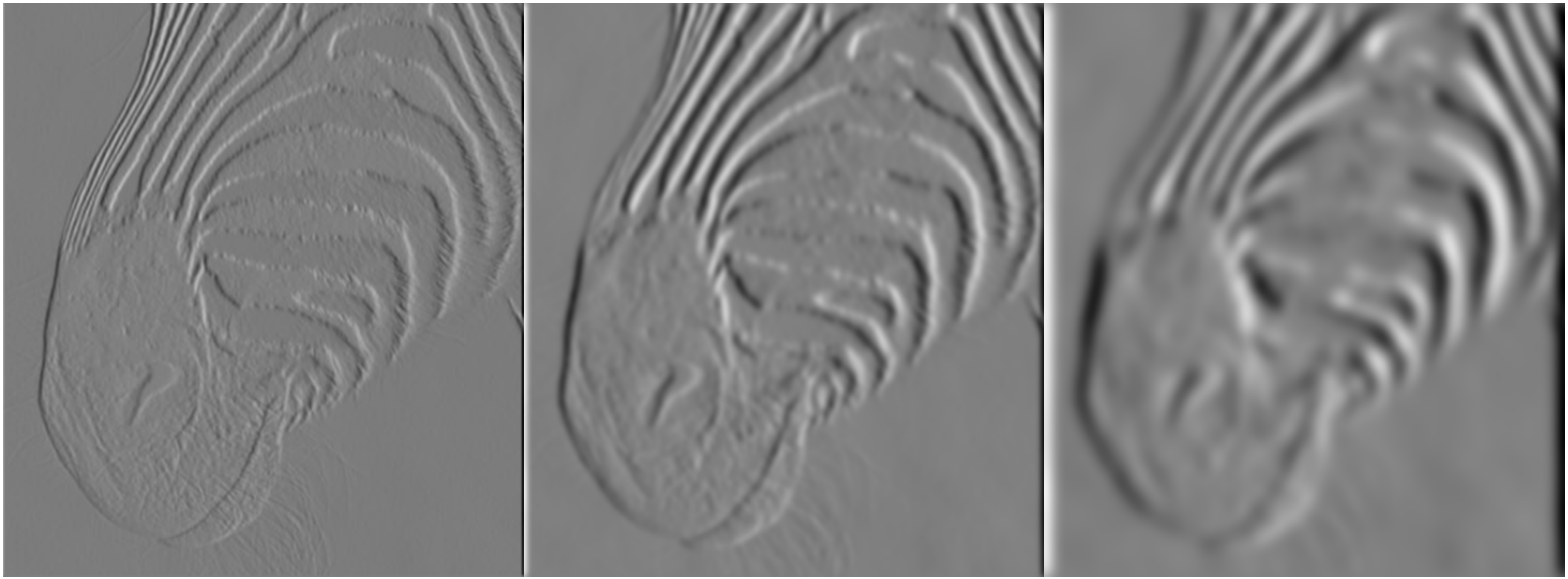
* [1 0 -1] =



- Is this filter separable?

Source: D. Hoiem

Tradeoff between smoothing and localization



1 pixel | 3 pixels | 7 pixels

- Smoothed derivative removes noise, but blurs edge. Also finds edges at different “scales”.

Source: D. Forsyth

Designing an edge detector

- Criteria for a good edge detector:
  - **Good detection:** find all real edges, ignoring noise or other artifacts
  - **Good localization**
    - detect edges as close as possible to the true edges
    - 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

Source: L. Fei-Fei

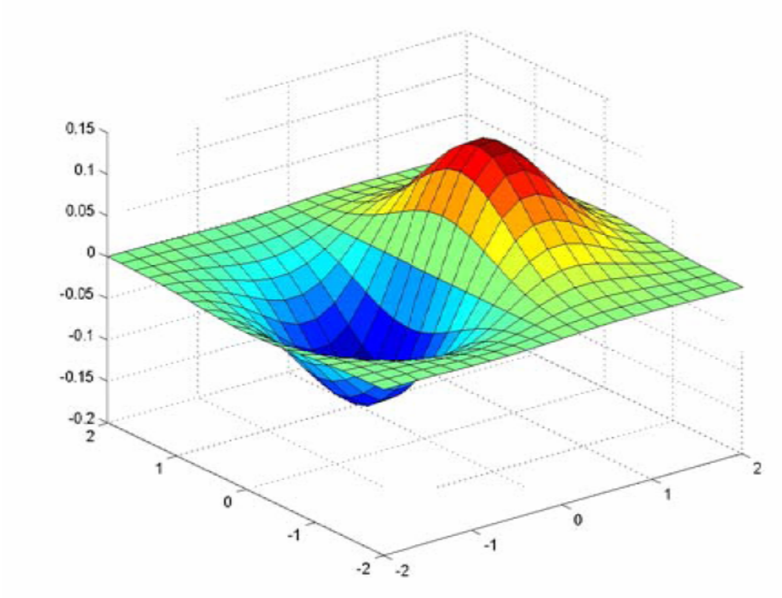
Canny edge detector

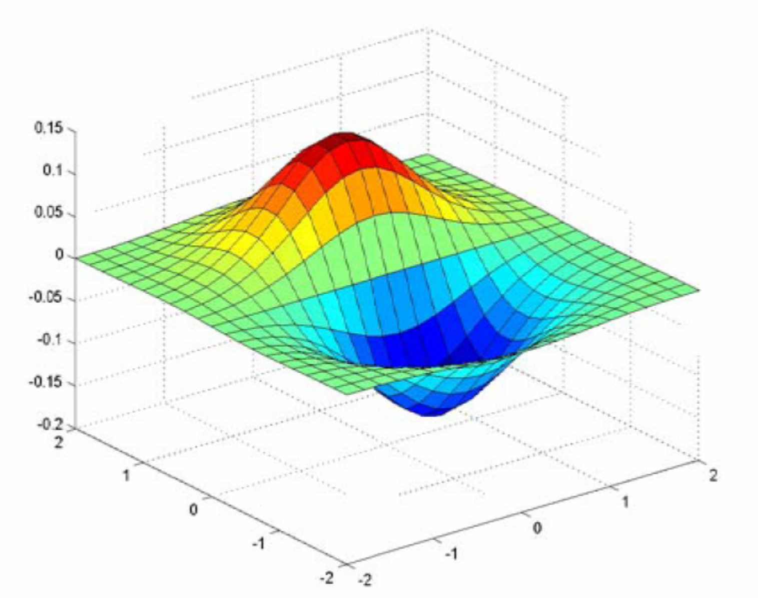
- This is probably the most widely used edge detector in computer vision
- Theoretical model: step-edges corrupted by additive Gaussian noise
- Canny has shown that the first derivative of the Gaussian closely approximates the operator that optimizes the product of *signal-to-noise ratio* and localization

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

Source: L. Fei-Fei

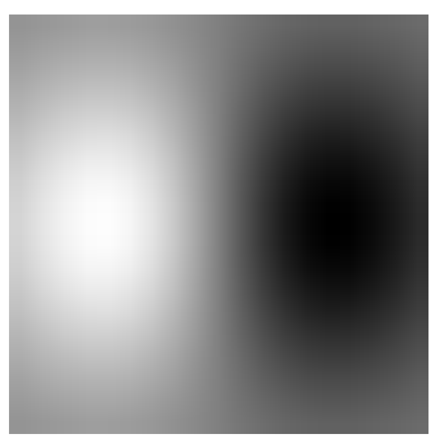
Derivative of Gaussian filters

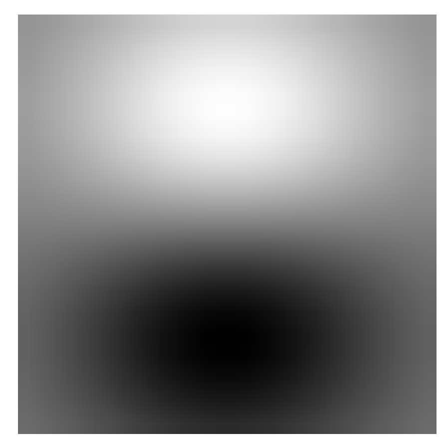




x-direction

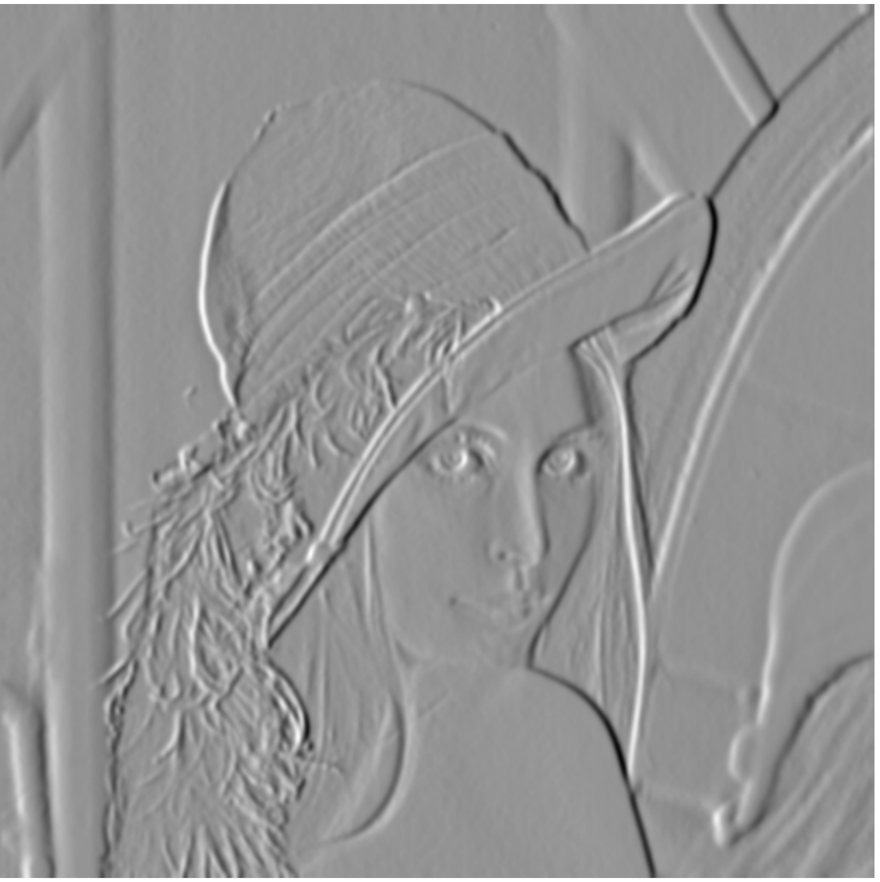
y-direction





Source: S.Lazebnik

Compute Gradients (DoG)



X-Derivative of Gaussian

Y-Derivative of Gaussian

Gradient Magnitude

Source: D. Hoiem

Building an edge detector

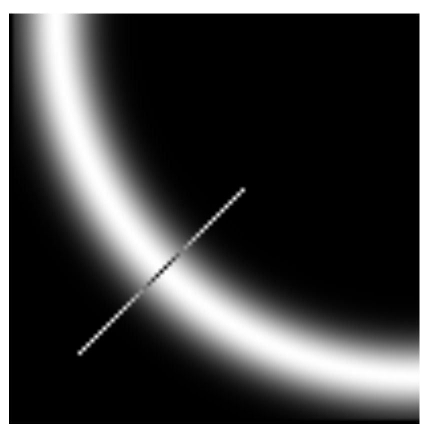
How to turn these thick regions of the gradient into curves?

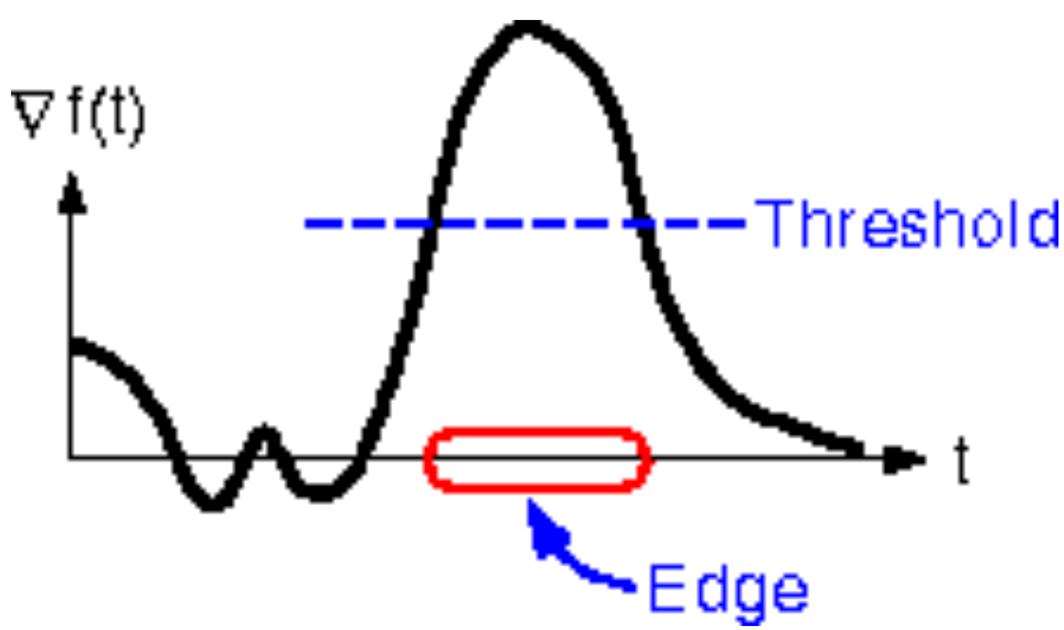
Thresholded norm of the gradient

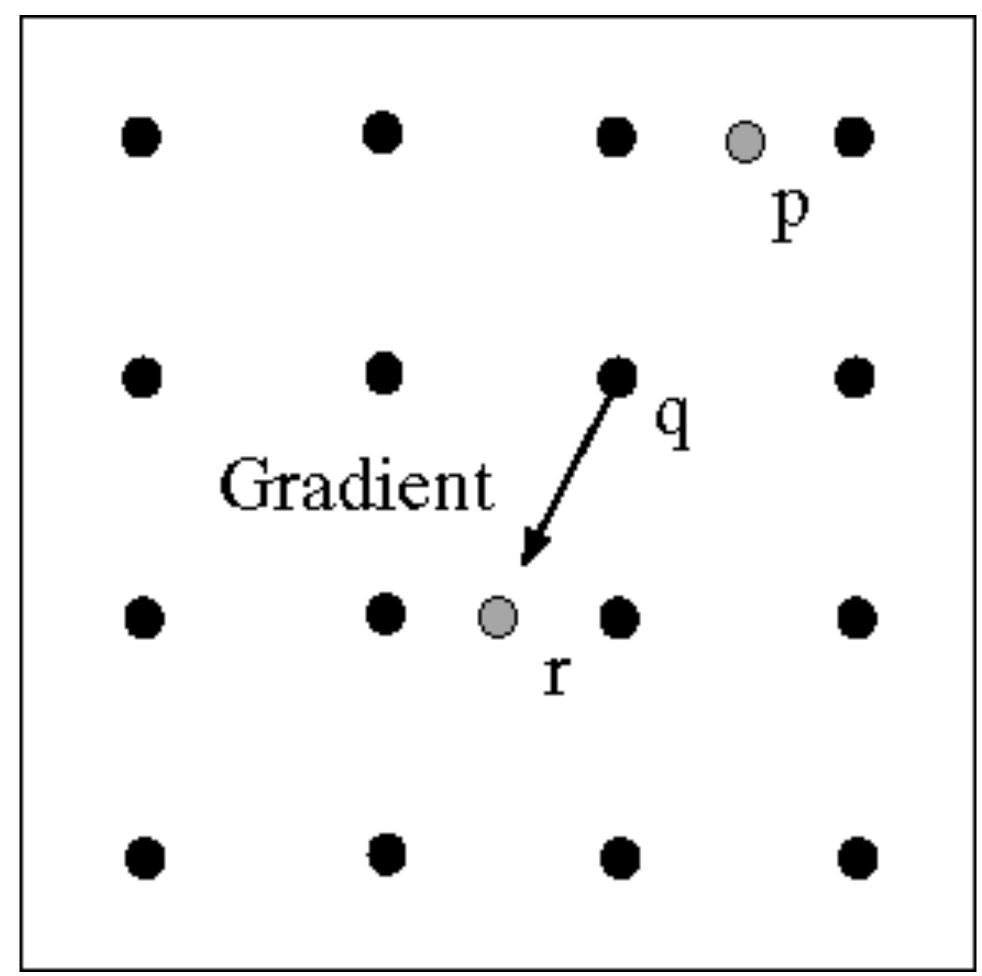
Source: S.Lazebnik

Non-maximum suppression

- For each location q above threshold, check that the gradient magnitude is higher than at neighbors p and r along the direction of the gradient
  - May need to interpolate to get the magnitudes at p and r







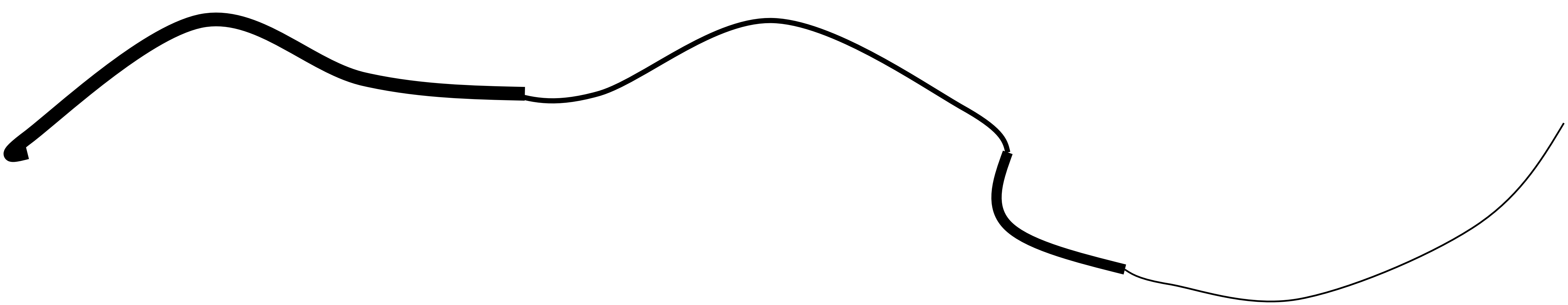
Source: S.Lazebnik

Before Non-max Suppression

After non-max suppression

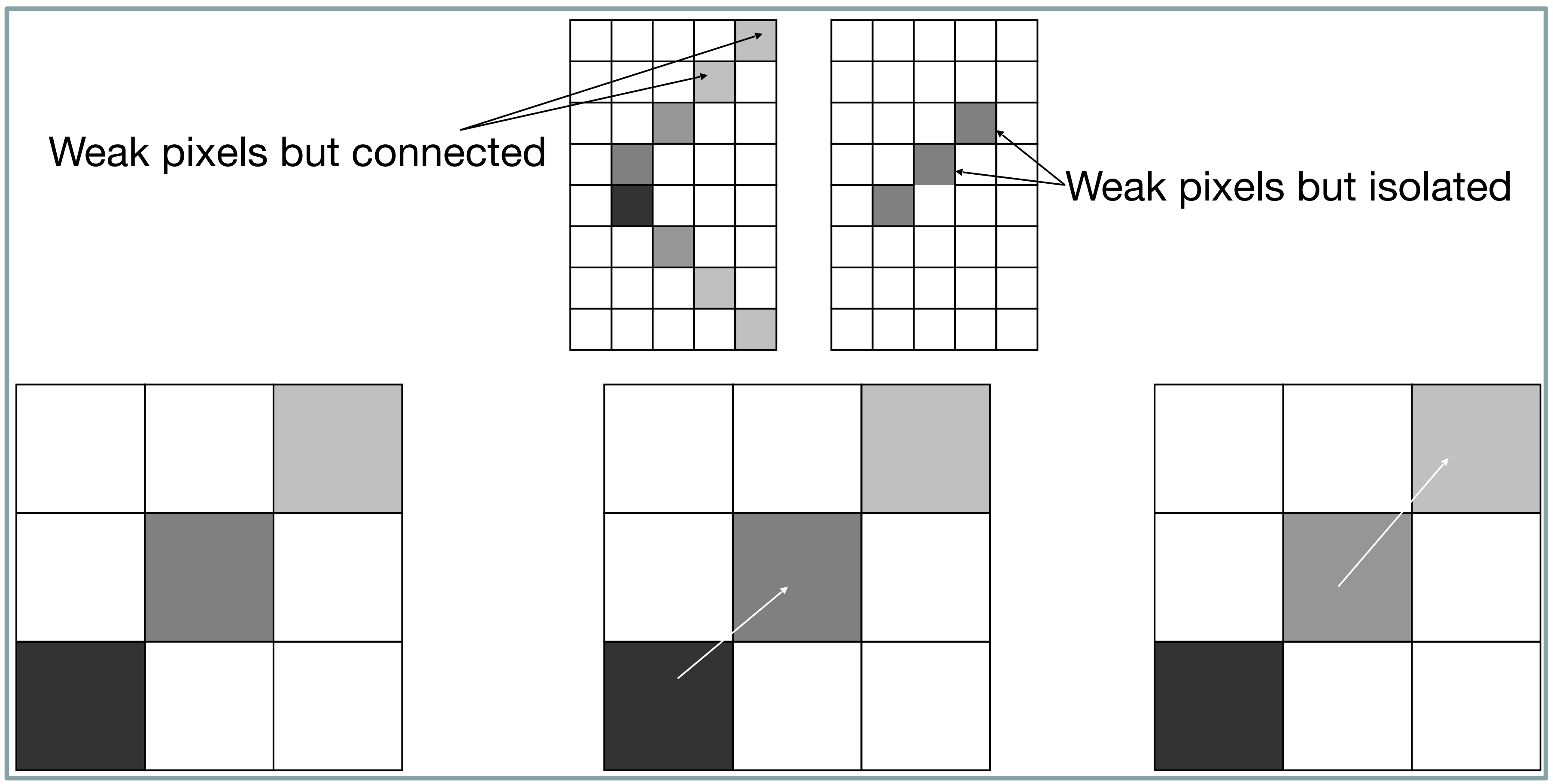
Hysteresis thresholding

- 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.



Source: S. Seitz

Canny edge detector



Very strong edge response.
Let's start here

Weaker response but it is
connected to a confirmed
edge point. Let's keep it.

Continue…

Final Canny Edges

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

Source: S. Seitz

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