Introduction to Computer Vision

Instructors: Jean Ponce and Elena Sizikova
jean.ponce@inria.fr, es5223@nyu.edu

Slides adapted from Mathew Trager, and other computer vision courses

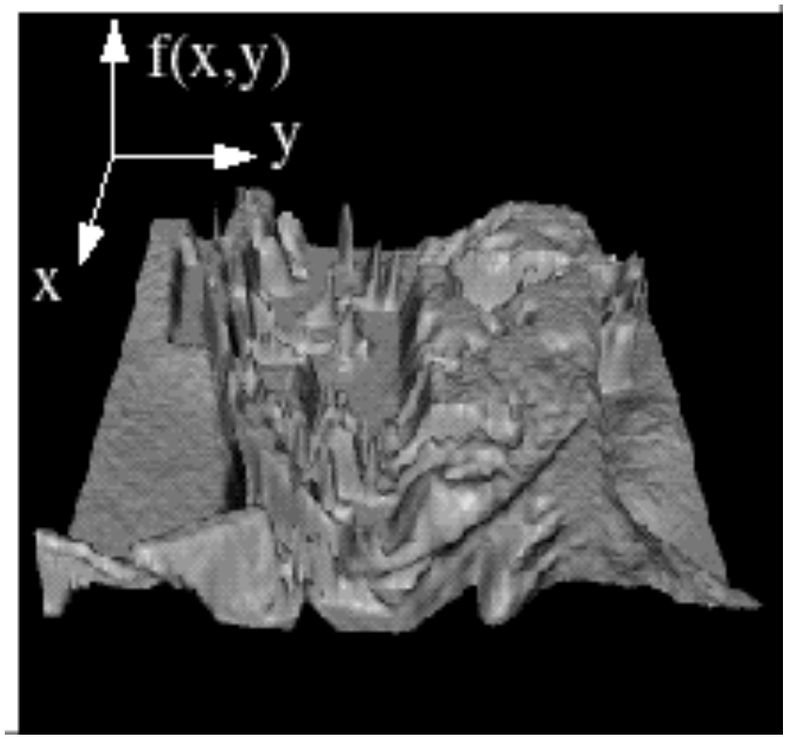
Outline

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

Linear Filters

- Given an image $In(x, y)$ generate a new image $Out(x, y)$:
  - For each pixel (x, y), $Out(x, y)$ is a specific linear combination of pixels in $In(x, y)$

- Properties:
  - Linear in input values (intensities)
  - Shift-invariant

Slide credit: Szymon Rusinkiewicz (Princeton)

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

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

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

$f[\cdot,\cdot]$

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

$h[\cdot,\cdot]$

	0								

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

Credit: S. Seitz

Image filtering

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

$f[\cdot,\cdot]$

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

$h[\cdot,\cdot]$

	0	10							

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

Credit: S. Seitz

Image filtering

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

$f[\cdot,\cdot]$

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

$h[\cdot,\cdot]$

	0	10	20						

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

Credit: S. Seitz

Image filtering

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

$f[\cdot,\cdot]$

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

$h[\cdot,\cdot]$

	0	10	20	30					

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

Credit: S. Seitz

Image filtering

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

$f[\cdot,\cdot]$

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

$h[\cdot,\cdot]$

	0	10	20	30	30				

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

Credit: S. Seitz

Image filtering

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

$f[\cdot,\cdot]$

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

$h[\cdot,\cdot]$

	0	10	20	30	30				
				?					

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

Credit: S. Seitz

Image filtering

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

$f[\cdot,\cdot]$

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

$h[\cdot,\cdot]$

	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

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

1	1	1
1	1	1
1	1	1

$f[\cdot,\cdot]$

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

$h[\cdot,\cdot]$

	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

- Replaces each pixel with an average of its neighborhood

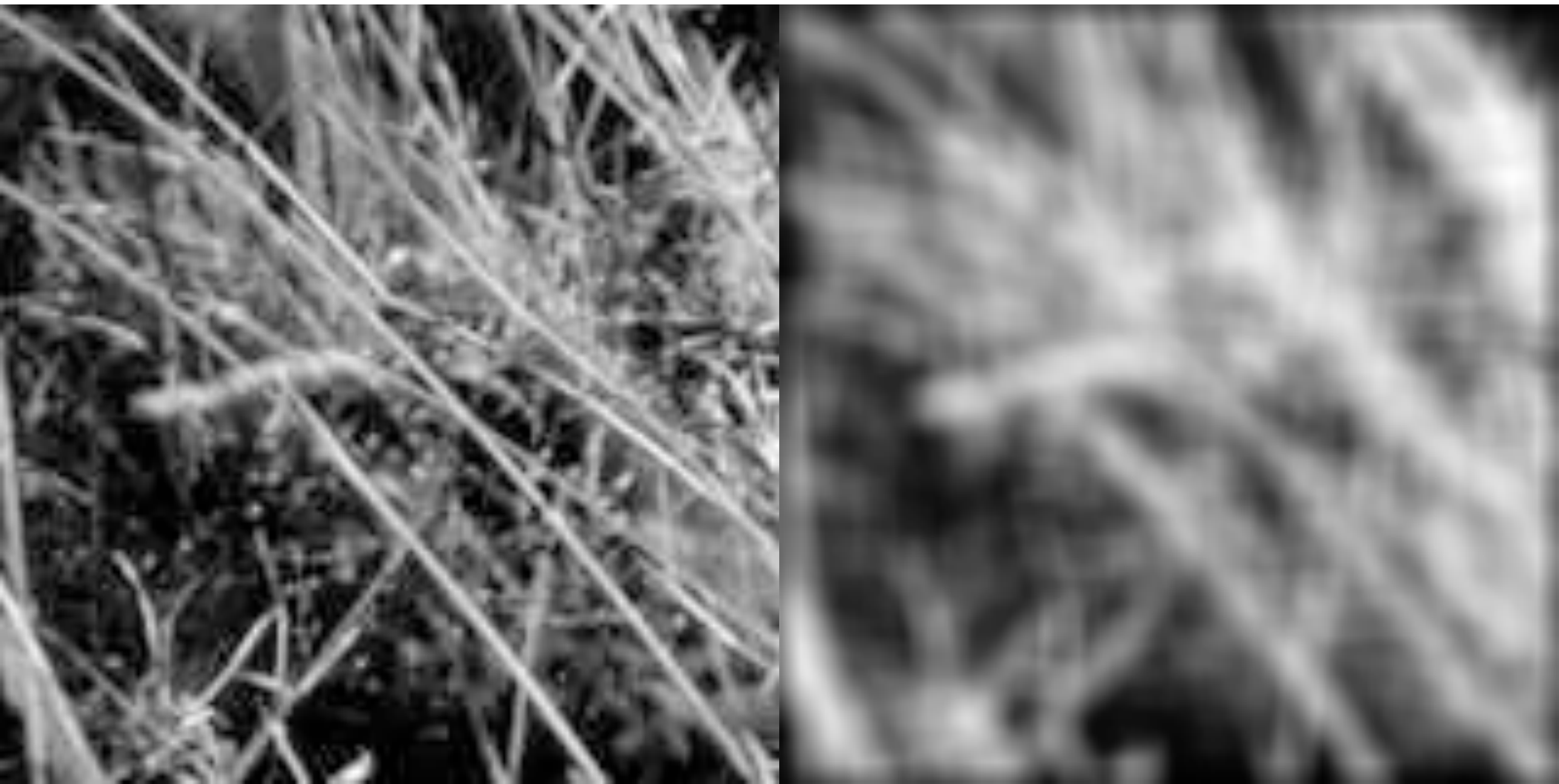
- Achieve smoothing effect (remove sharp features)

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

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

Slide credit: David Lowe (UBC)

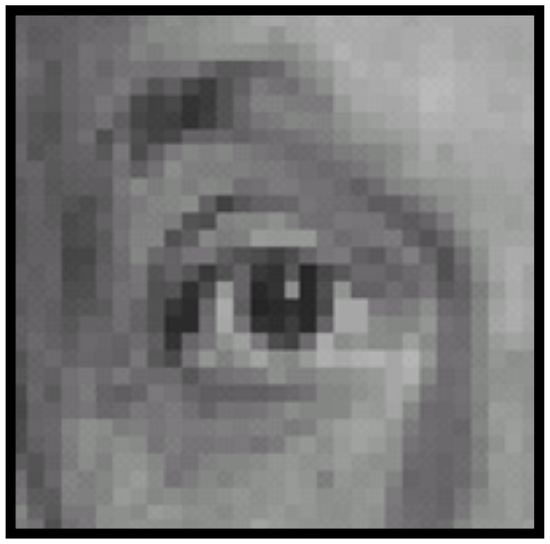
Smoothing with box filter



Applications

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

Practice with linear filters



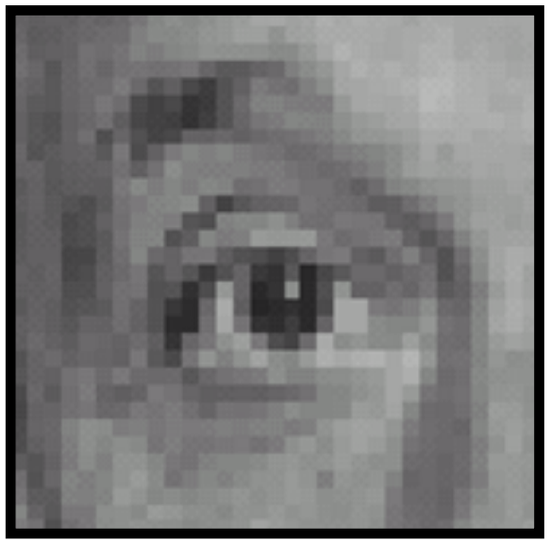
Original

0	0	0
0	1	0
0	0	0

?

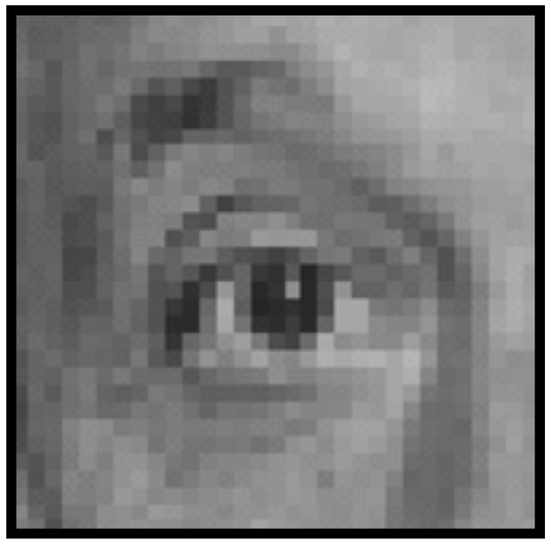
Source: D. Lowe

Practice with linear filters



Original

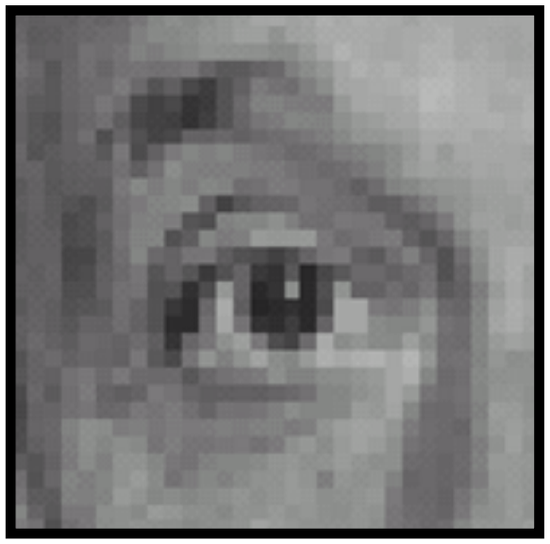
0	0	0
0	1	0
0	0	0



Filtered
(no change)

Source: D. Lowe

Practice with linear filters



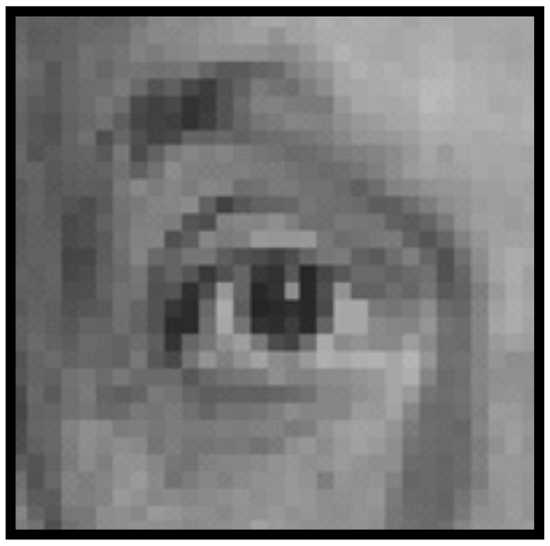
Original

0	0	0
0	0	1
0	0	0

?

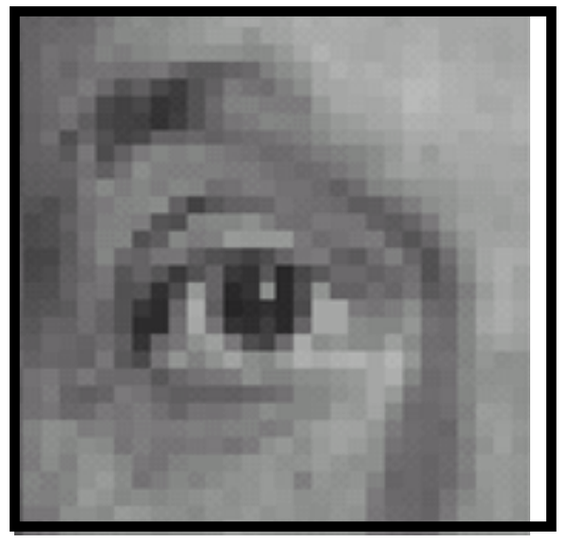
Source: D. Lowe

Practice with linear filters



Original

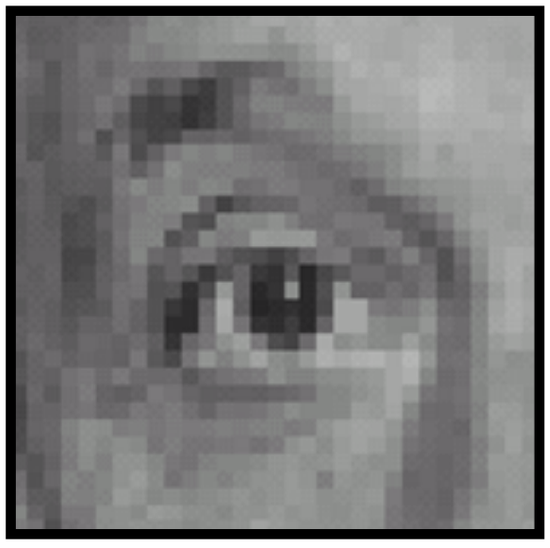
0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

Source: D. Lowe

Practice with linear filters



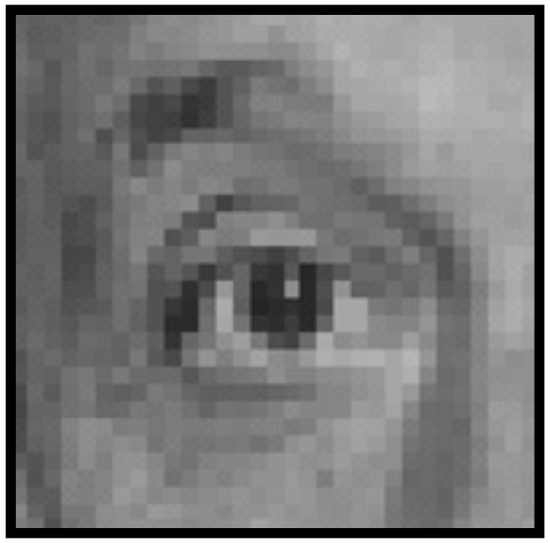
Original

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

(Note that filter sums to 1)

Source: D. Lowe

Practice with linear filters



Original

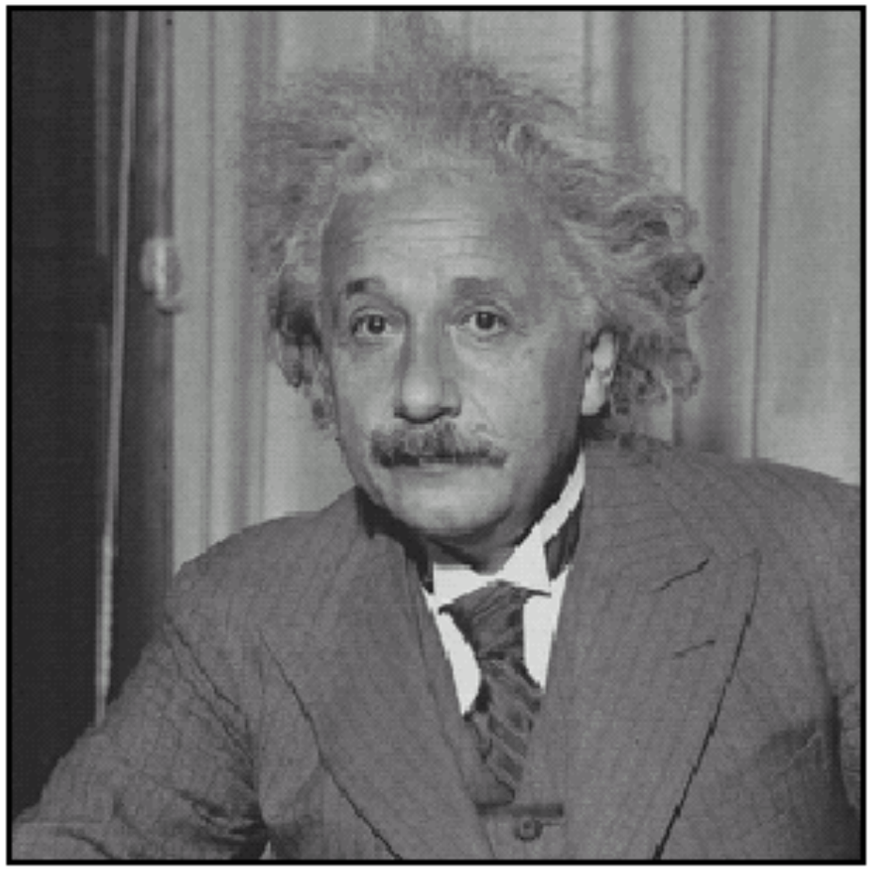
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Sharpening filter

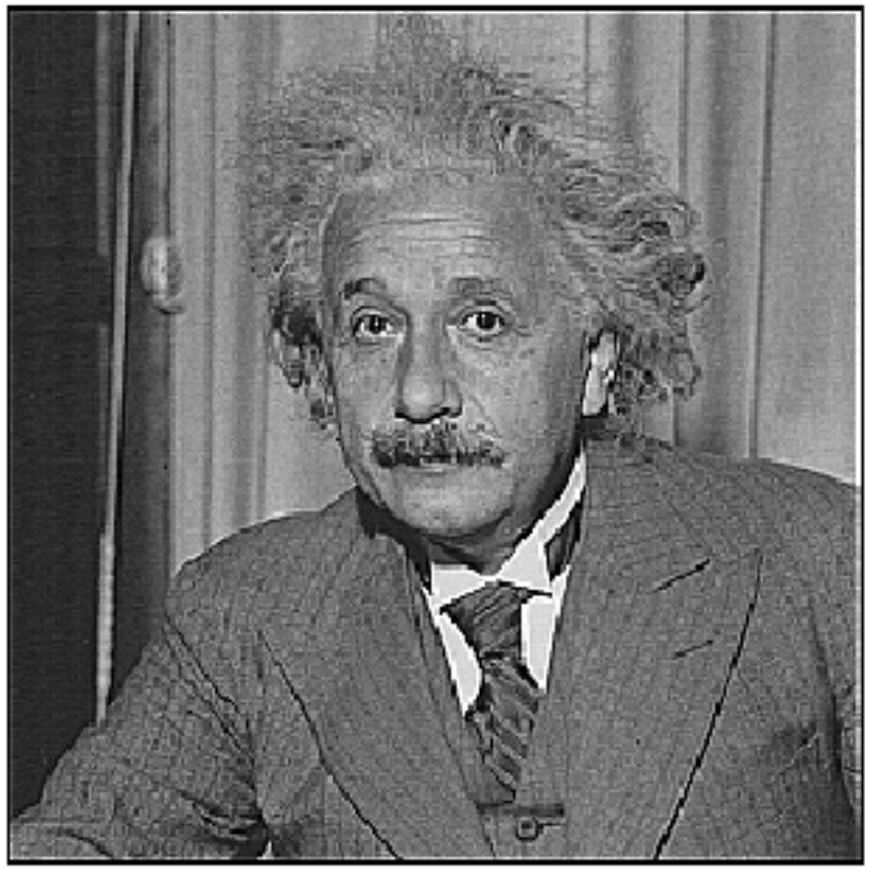
- Accentuates differences with local average

Source: D. Lowe

Sharpening



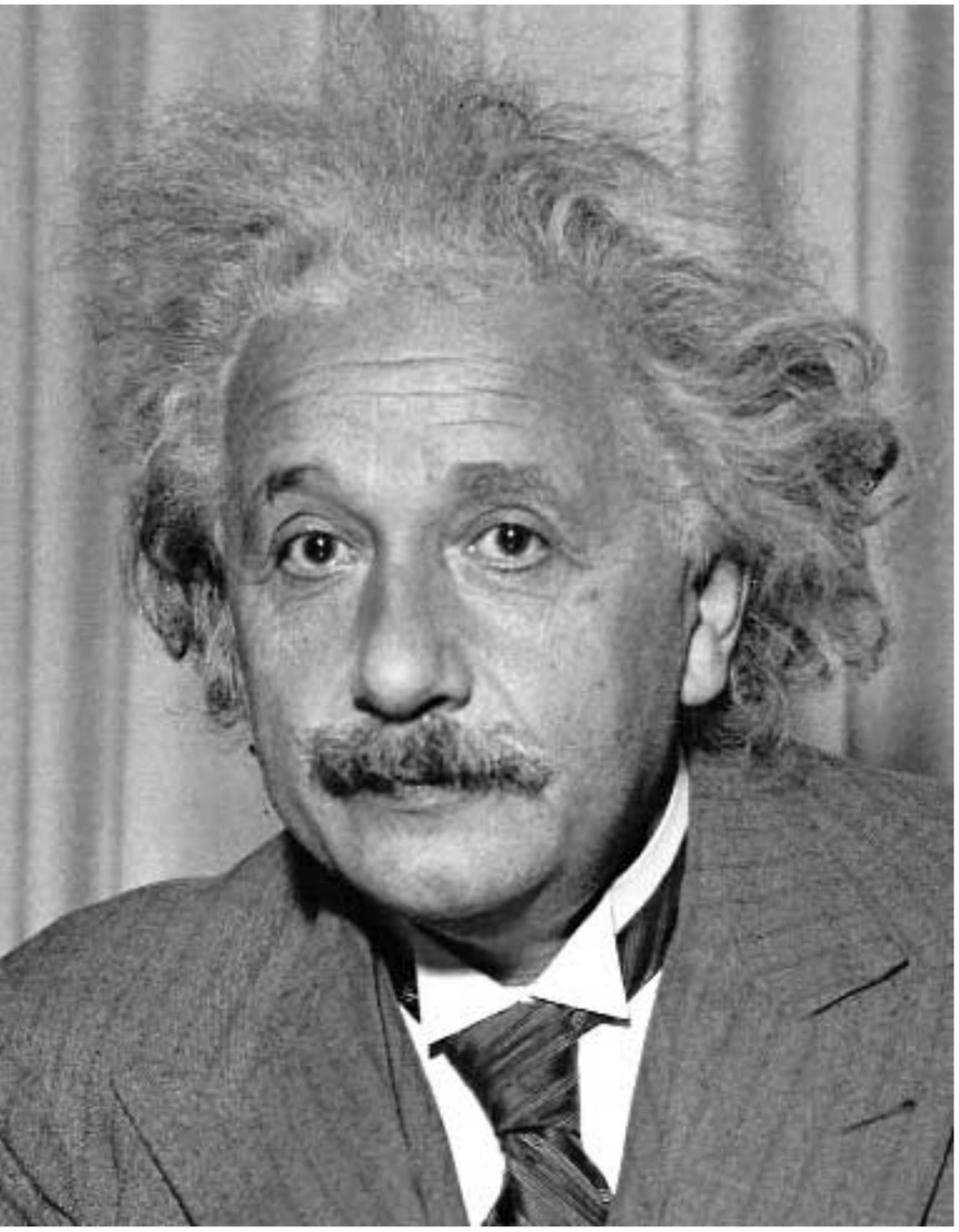
before



after

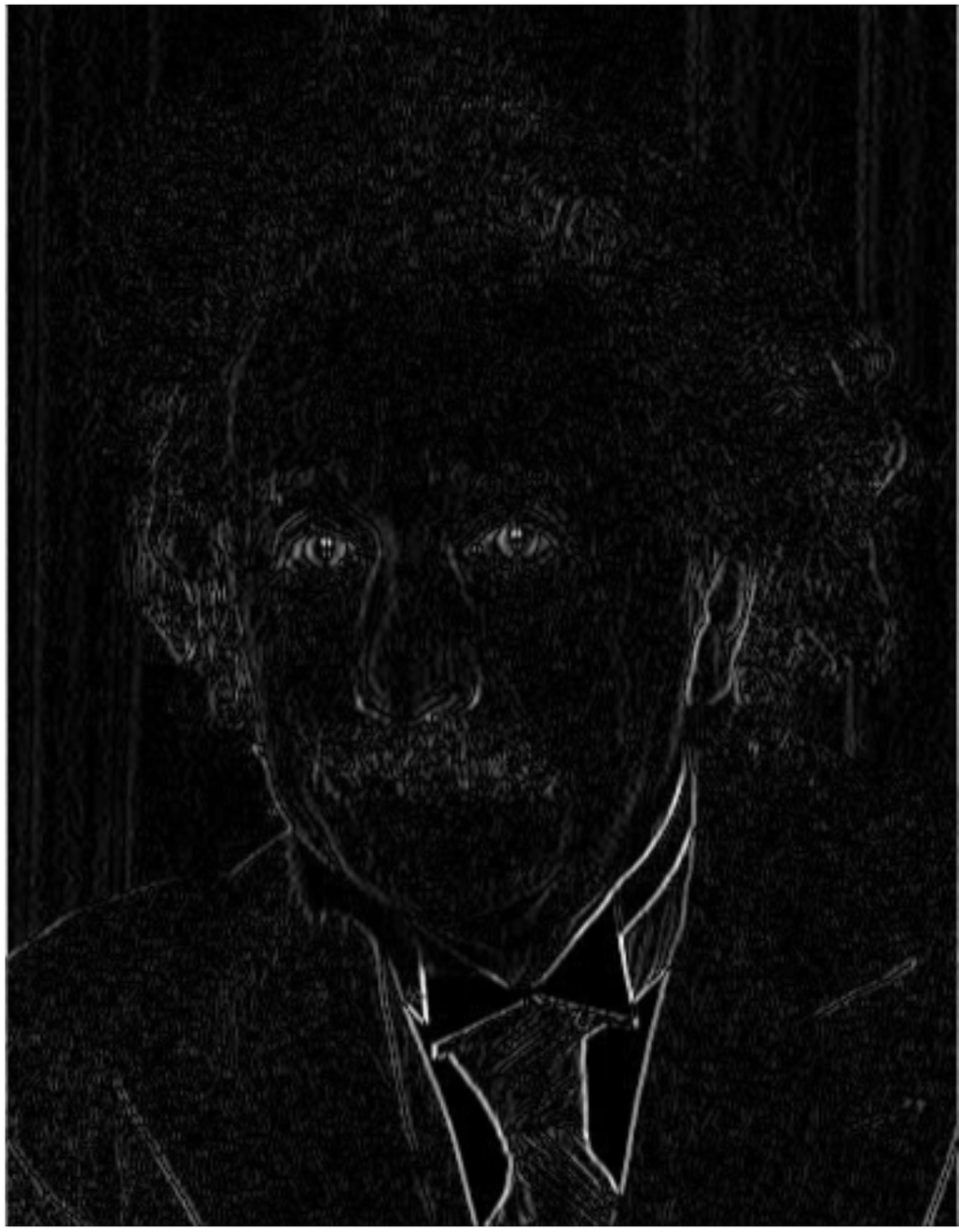
Source: D. Lowe

Other filters



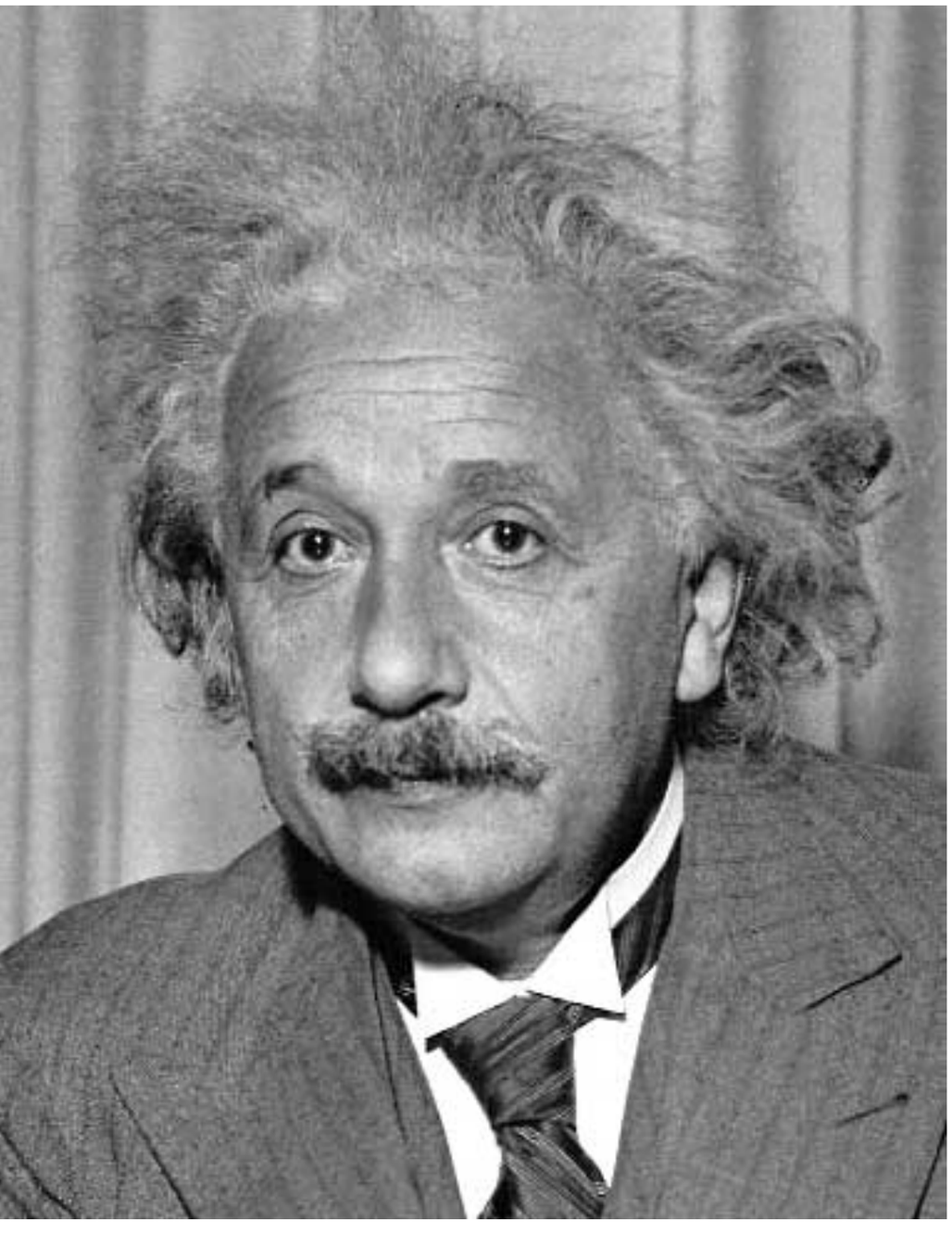
1	0	-1
2	0	-2
1	0	-1

Sobel



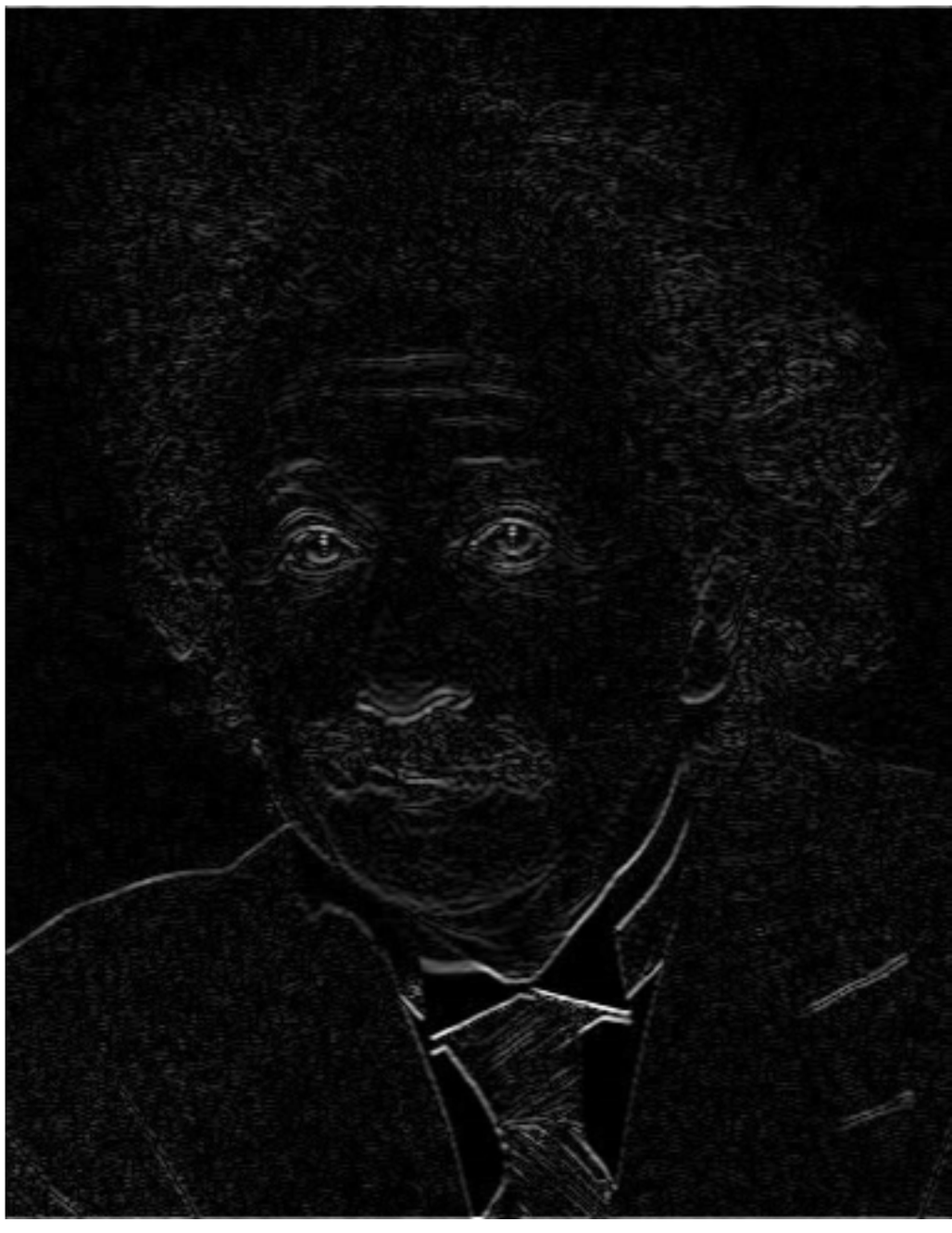
Vertical Edge
(absolute value)

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

Correlation vs convolution

Correlation (linear filtering):

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

Convolution:

Equivalent to flip the filter in both dimensions (bottom to top, right to left) and apply cross-correlation.

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

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

Correlation vs convolution

- Convolution is both:
  - Commutative $f \star g = g \star f$
  - Associative $(f \star g) \star h = f \star (g \star h)$

- Both correlation and convolution are:
  - linear (scalars a, b and operators h, f, g):

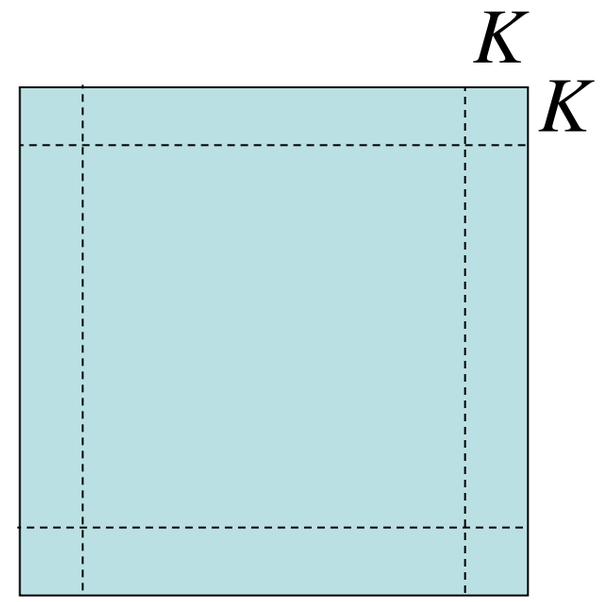
$$h \circ (af + bg) = ah \circ f + bh \circ g$$

  - Shift-invariant

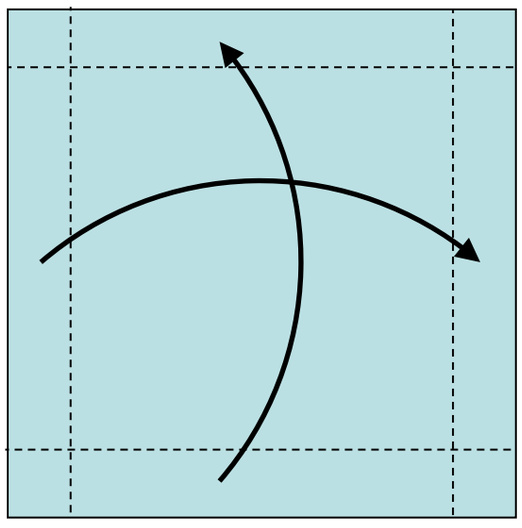
if $g(i,j) = f(i+k, j+l) \leftrightarrow (h \circ g)(i,j) = (h \circ f)(i+k, j+l)$

Practicalities

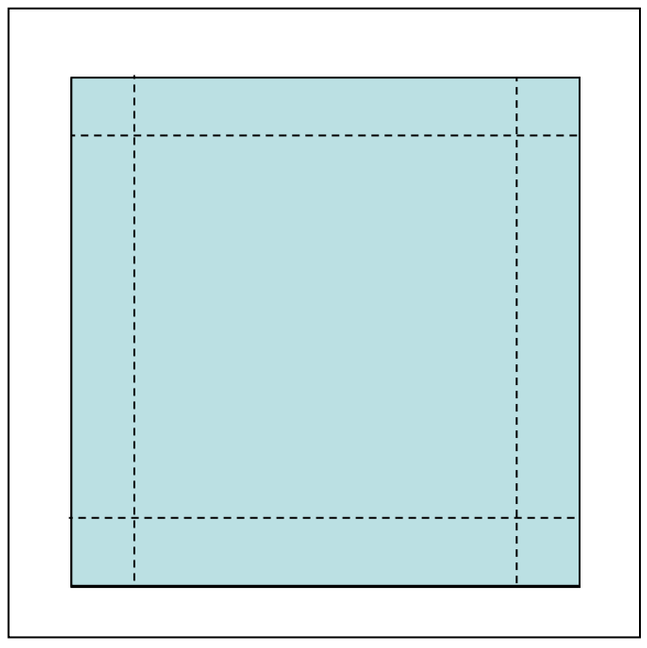
- MATLAB: conv (1D) or conv2 (2D)
- Border issues:
  - When applying convolution with a KxK kernel, the result is undefined for pixels closer than K pixels from the border on the image



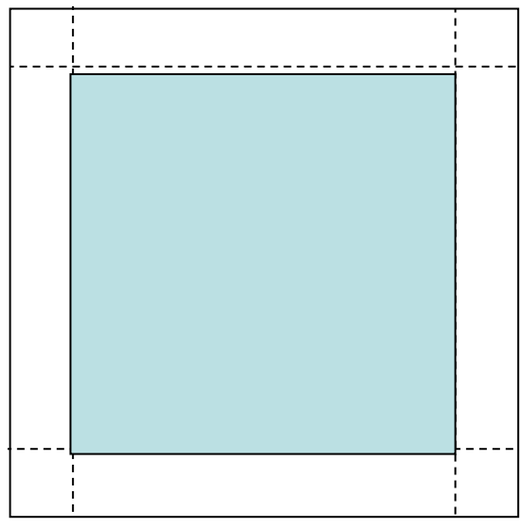
- Solutions:



Wrap Around



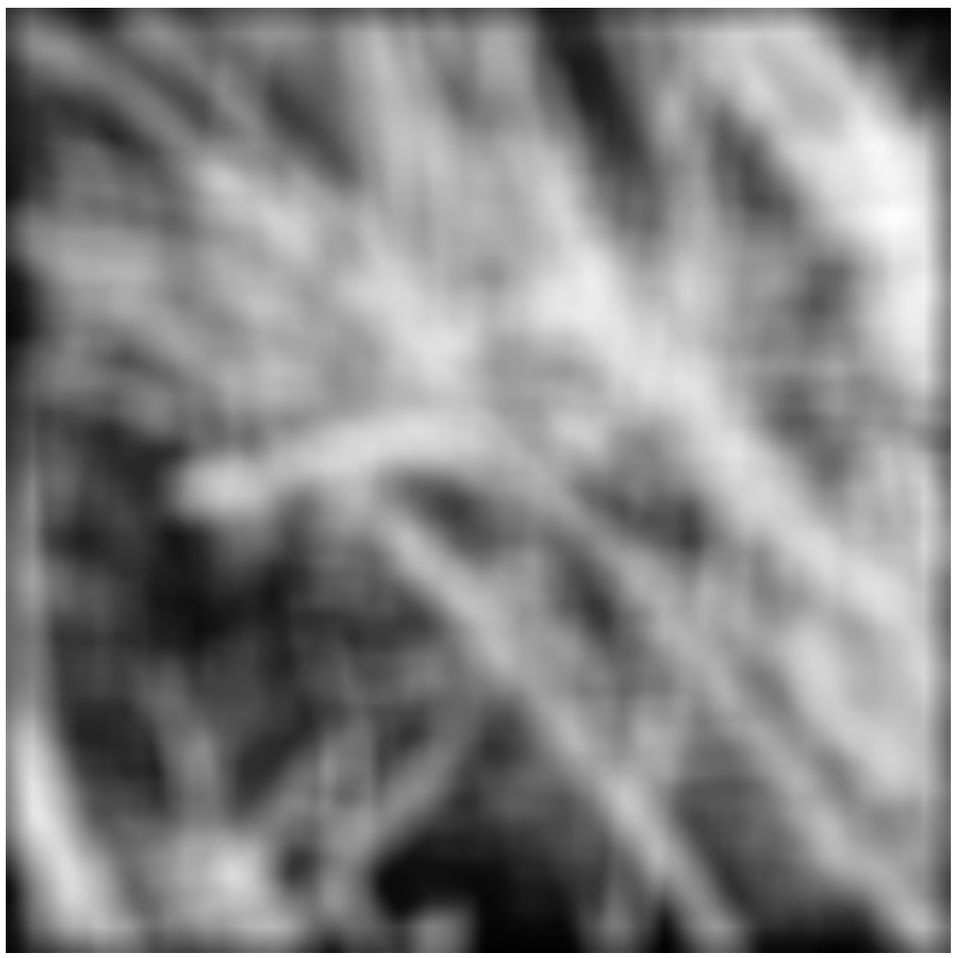
Expand/Pad



Crop

Smoothing with box filter revisited

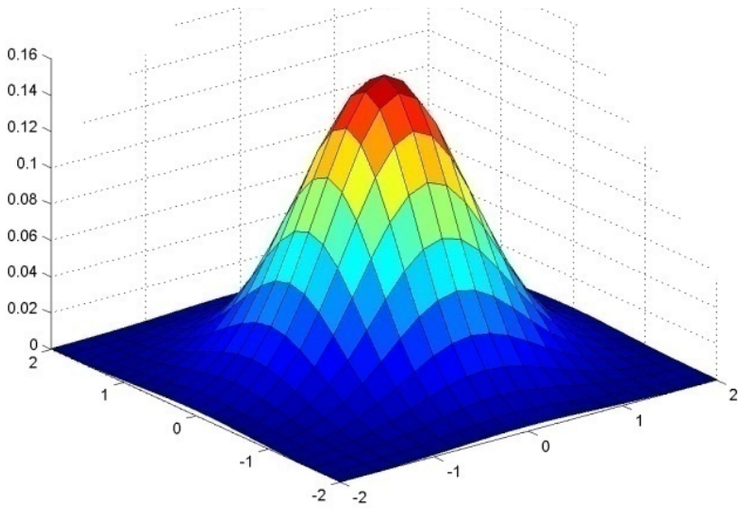
- What's wrong with this picture?
- What's the solution?

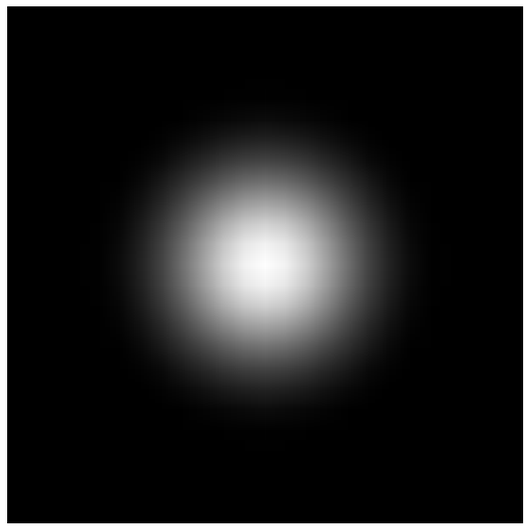


Source: D. Forsyth

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

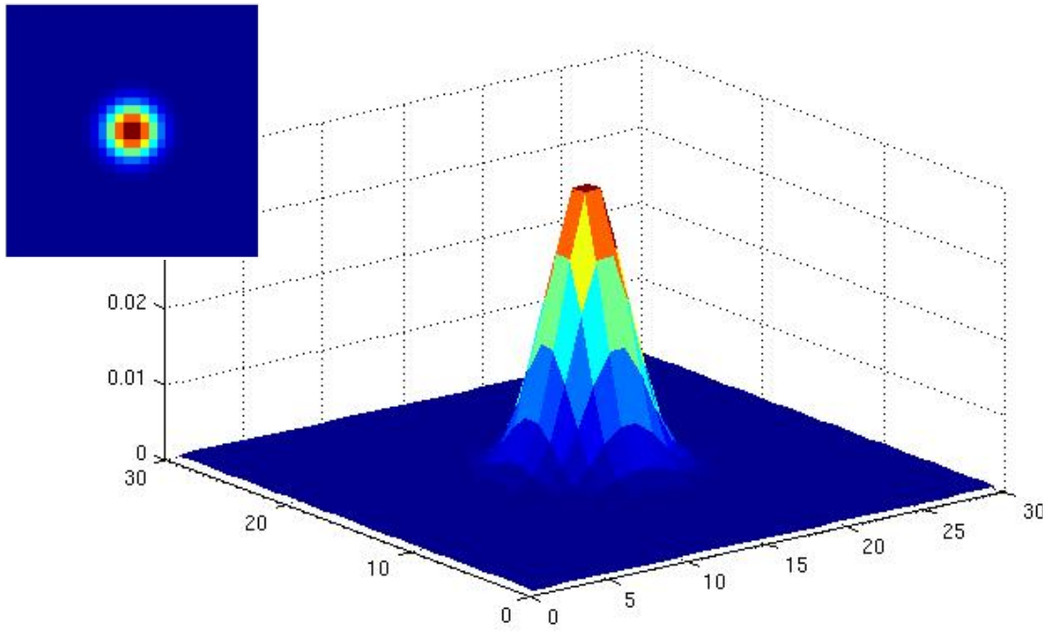
5 x 5, $\sigma = 1$

- 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)

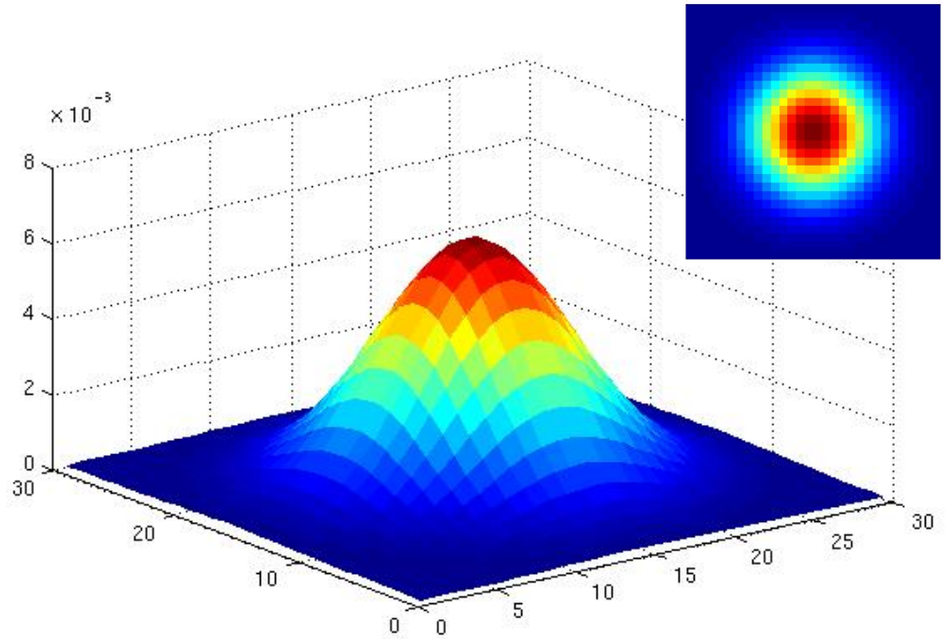
Source: C. Rasmussen

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



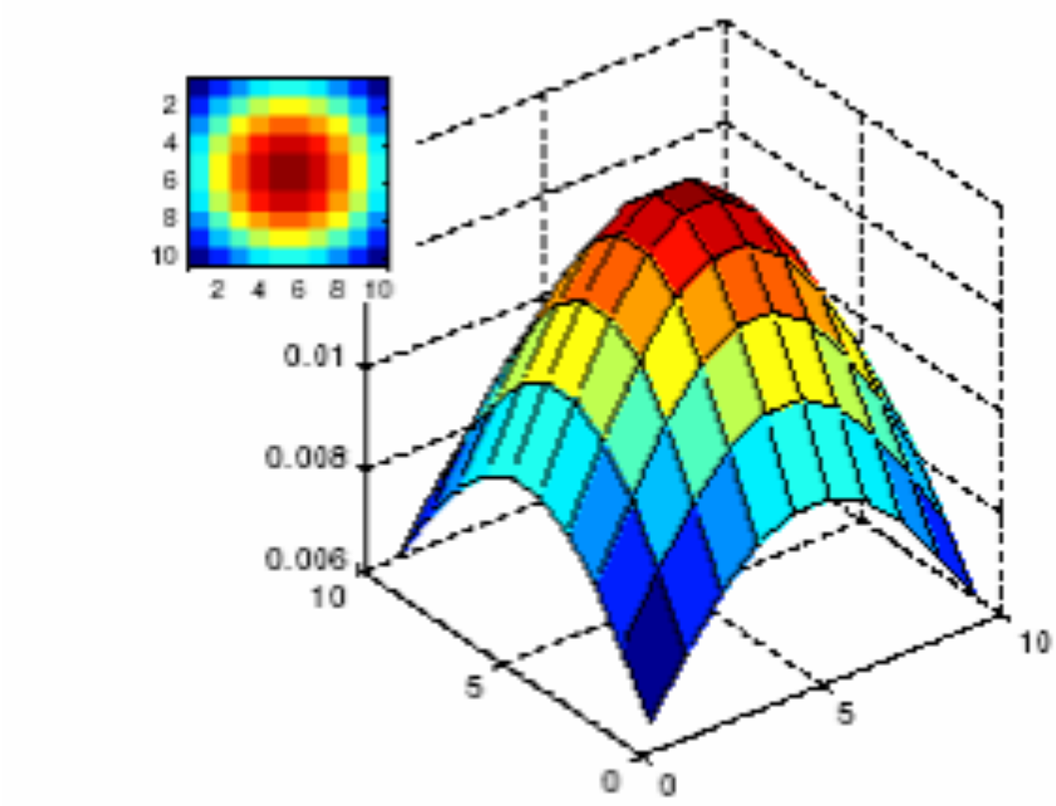
σ = 5 with 30 x 30 kernel

- Standard deviation σ: determines extent of smoothing

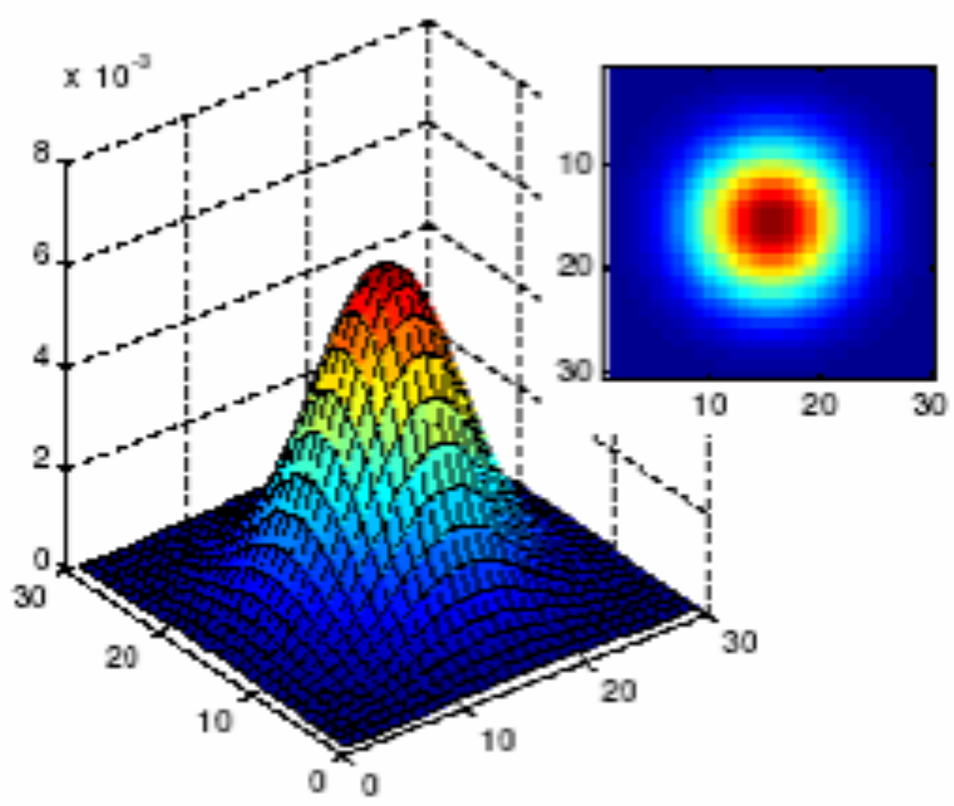
Source: K. Grauman

Choosing kernel width

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



σ = 5 with 10x10 kernel

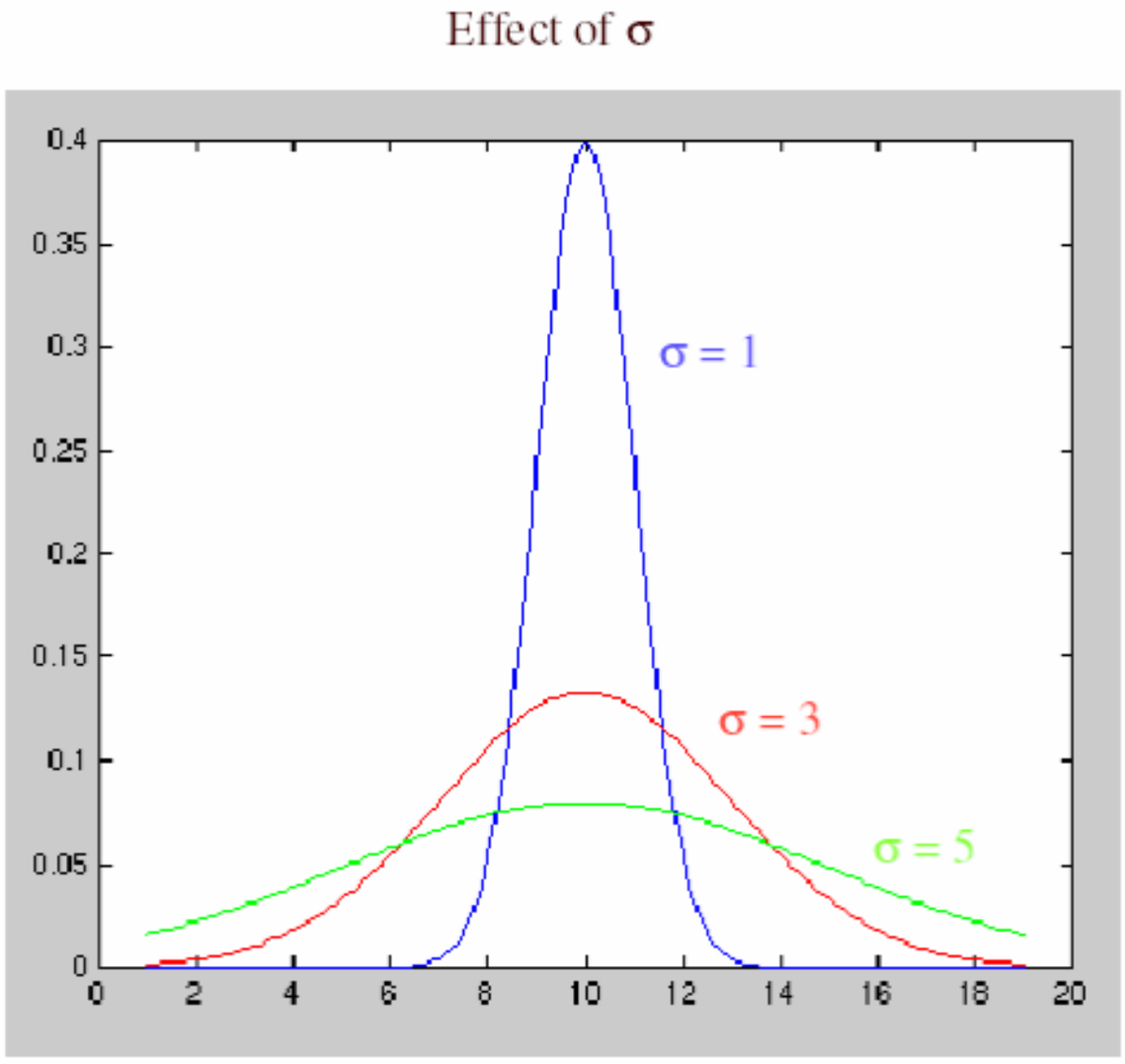


σ = 5 with 30x30 kernel

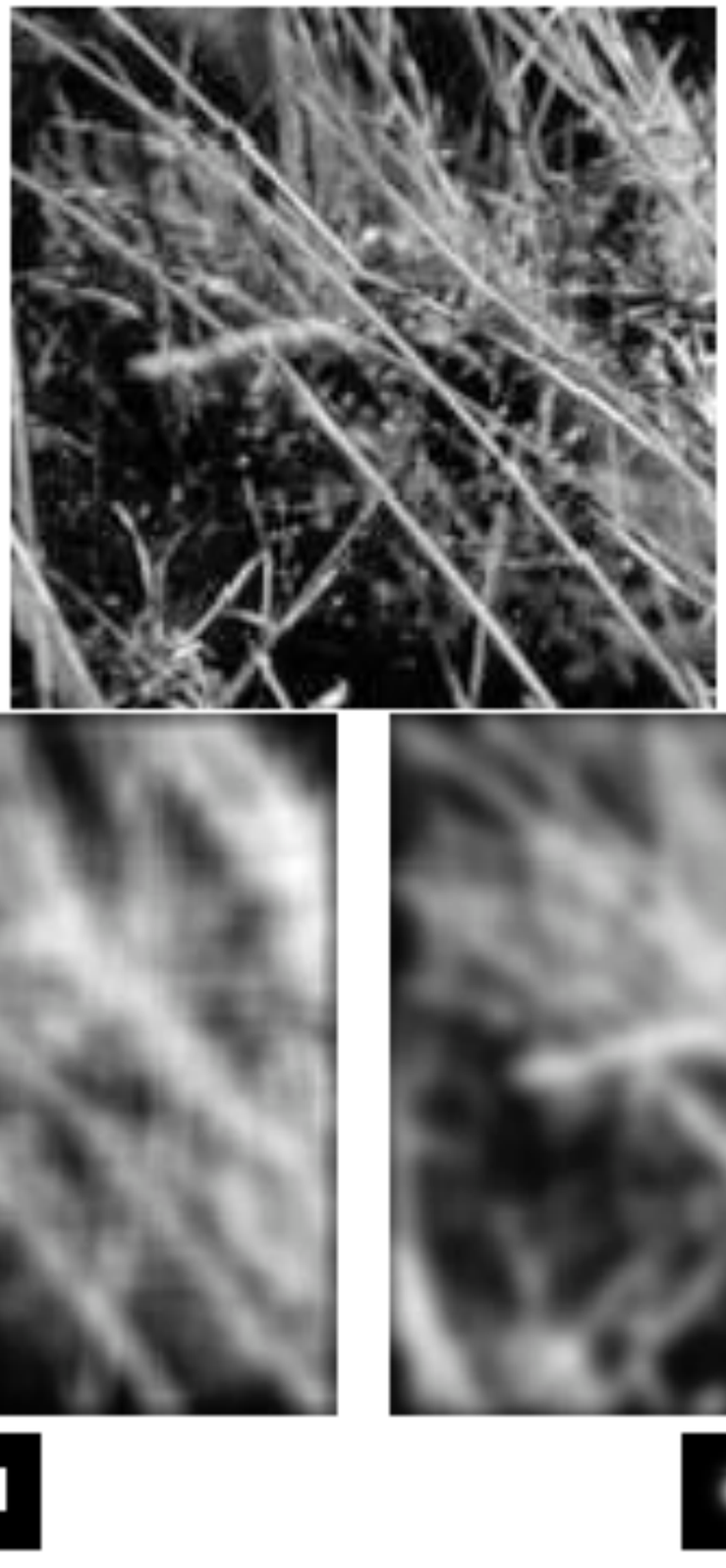
Source: K. Grauman

Choosing kernel width

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



Gaussian vs. box filtering



Source: S. Lazebnik

Properties of Gaussian Filters

- Smooth

- Decay to zero rapidly

- Simple analytic formula

- **Central limit theorem**: limit of applying (most) filters multiple times is some Gaussian

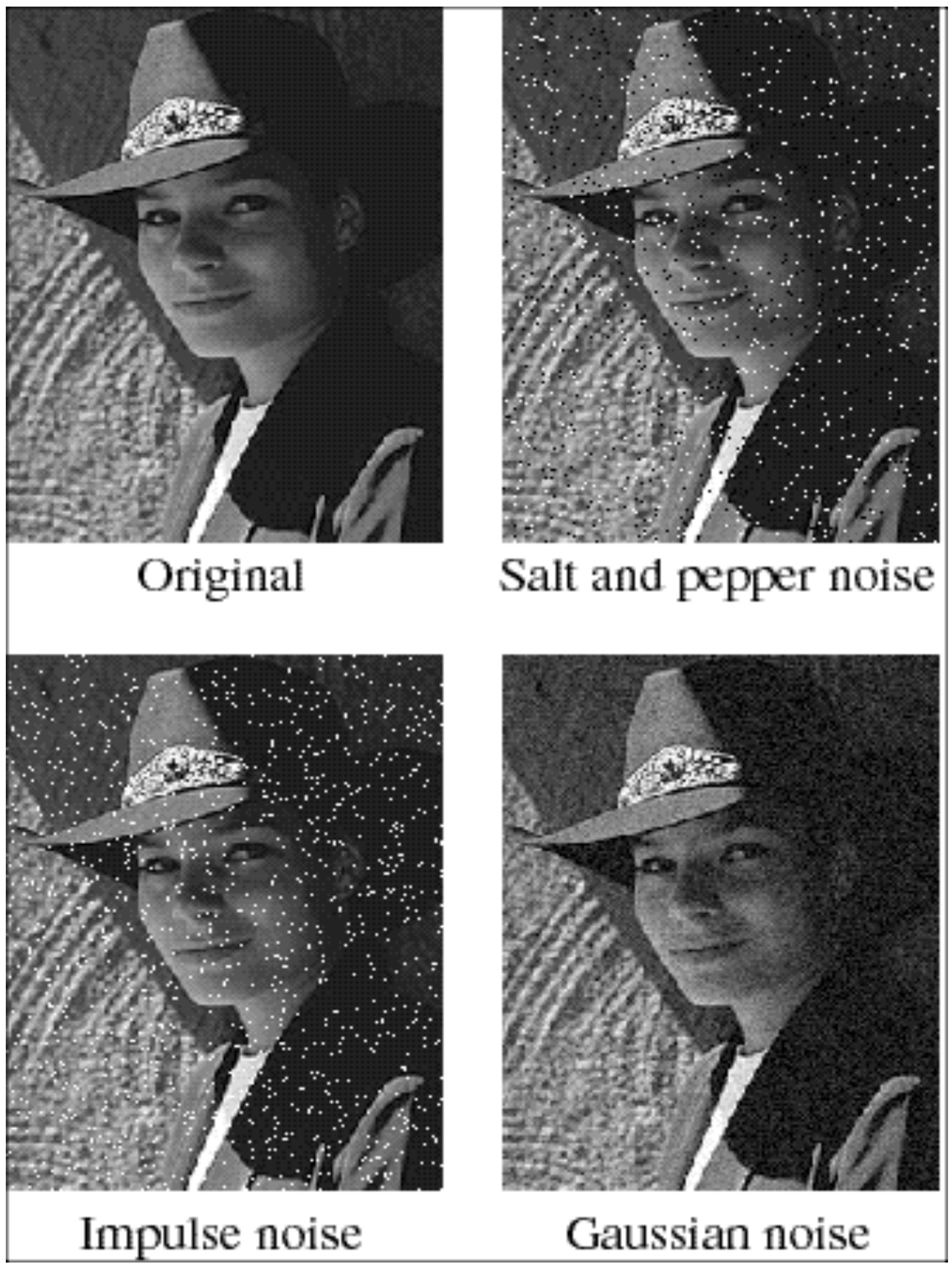
- Separable: $G_\sigma(x, y) = G_1(x)G_1(y)$

Slide credit: Szymon Rusinkiewicz (Princeton)

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?
  - $O(n^2 m^2)$
  - ok for small filter kernels, expensive for large ones
- What if the kernel is separable?
  - $O(n^2 m)$

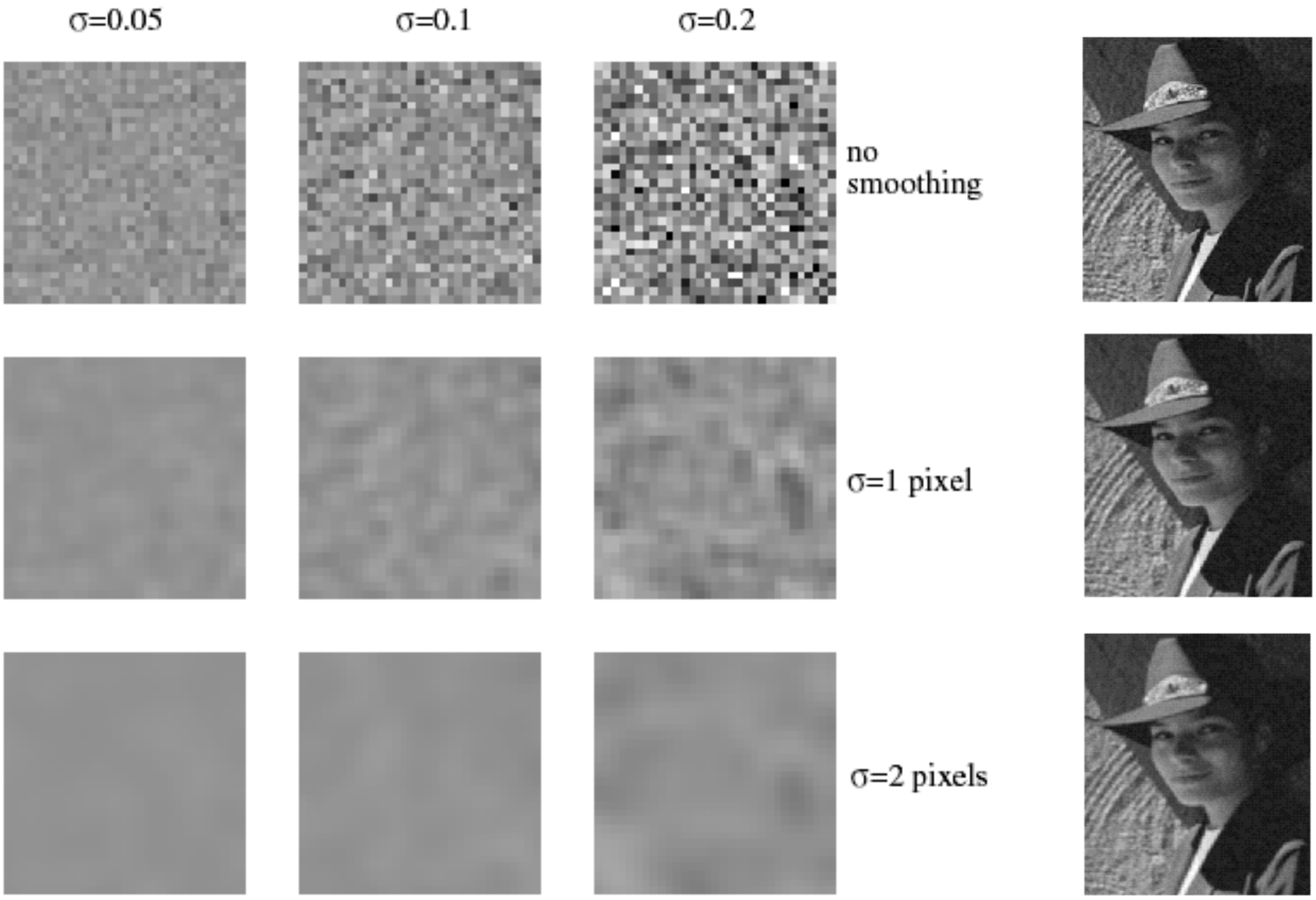
Noise



- **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

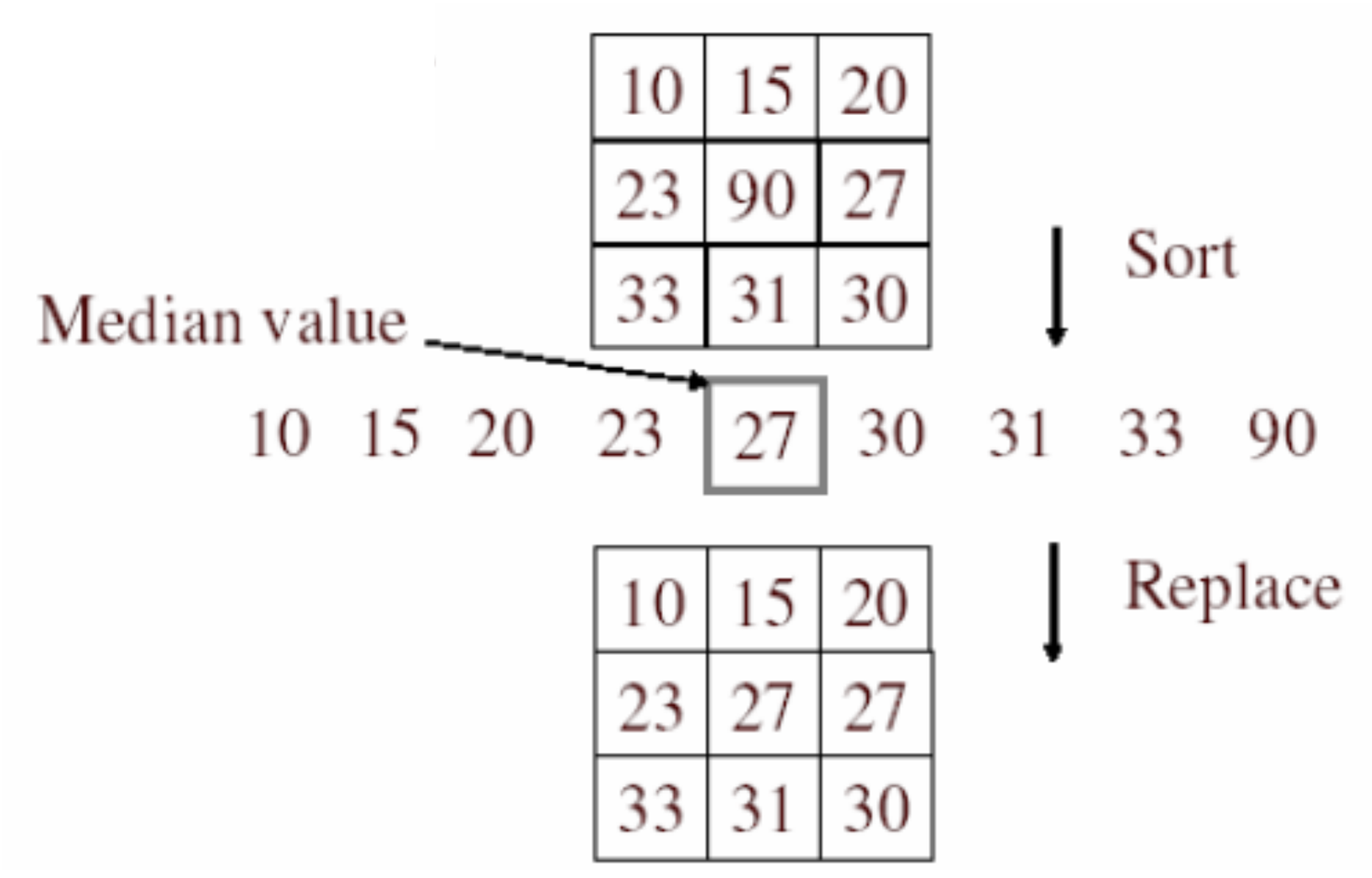
5x5

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

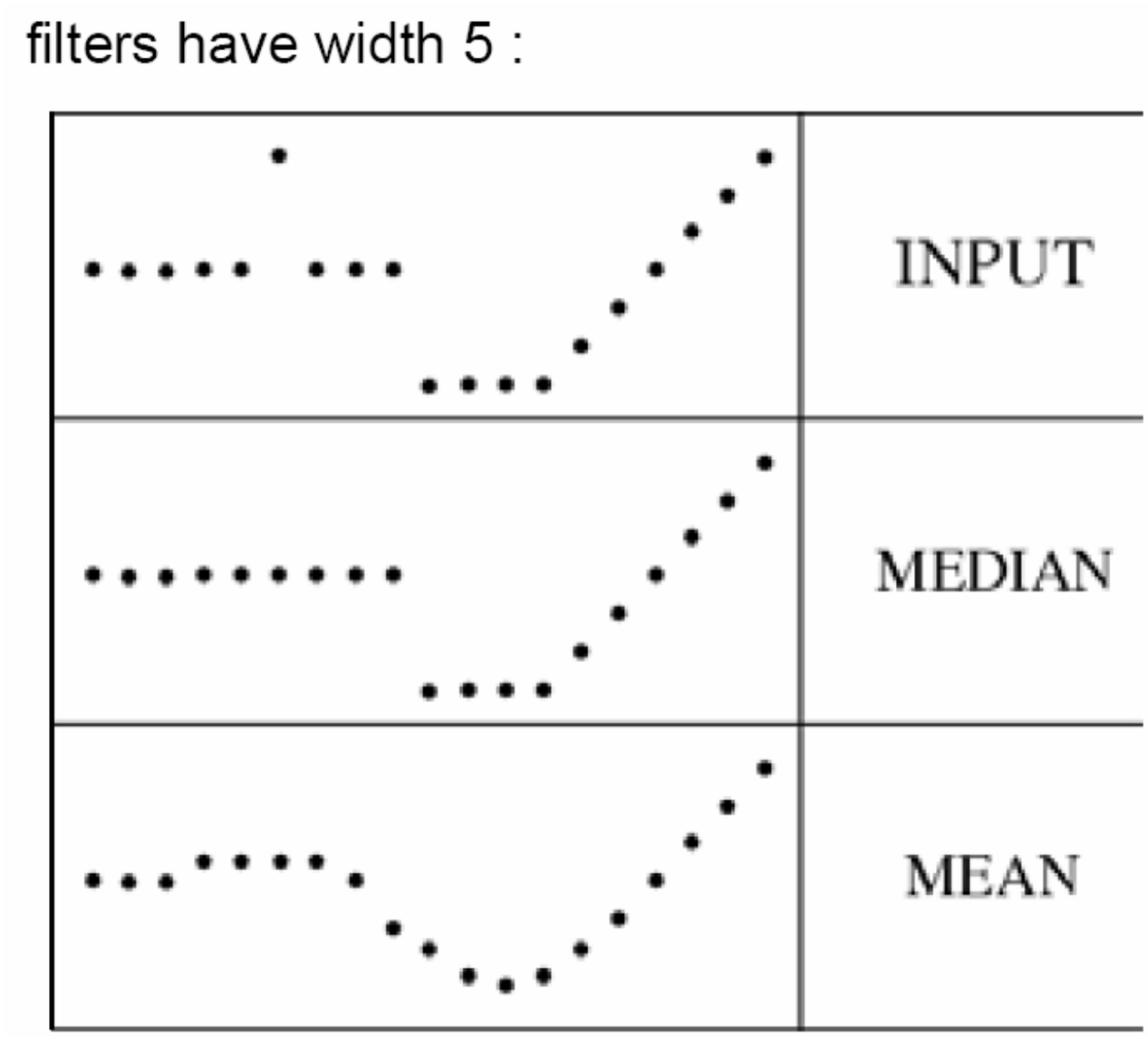
- Is median filtering linear?
- Let's try filtering

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

Hint: $F_{med}(A + B) \neq F_{med}(A) + F_{med}(B)$

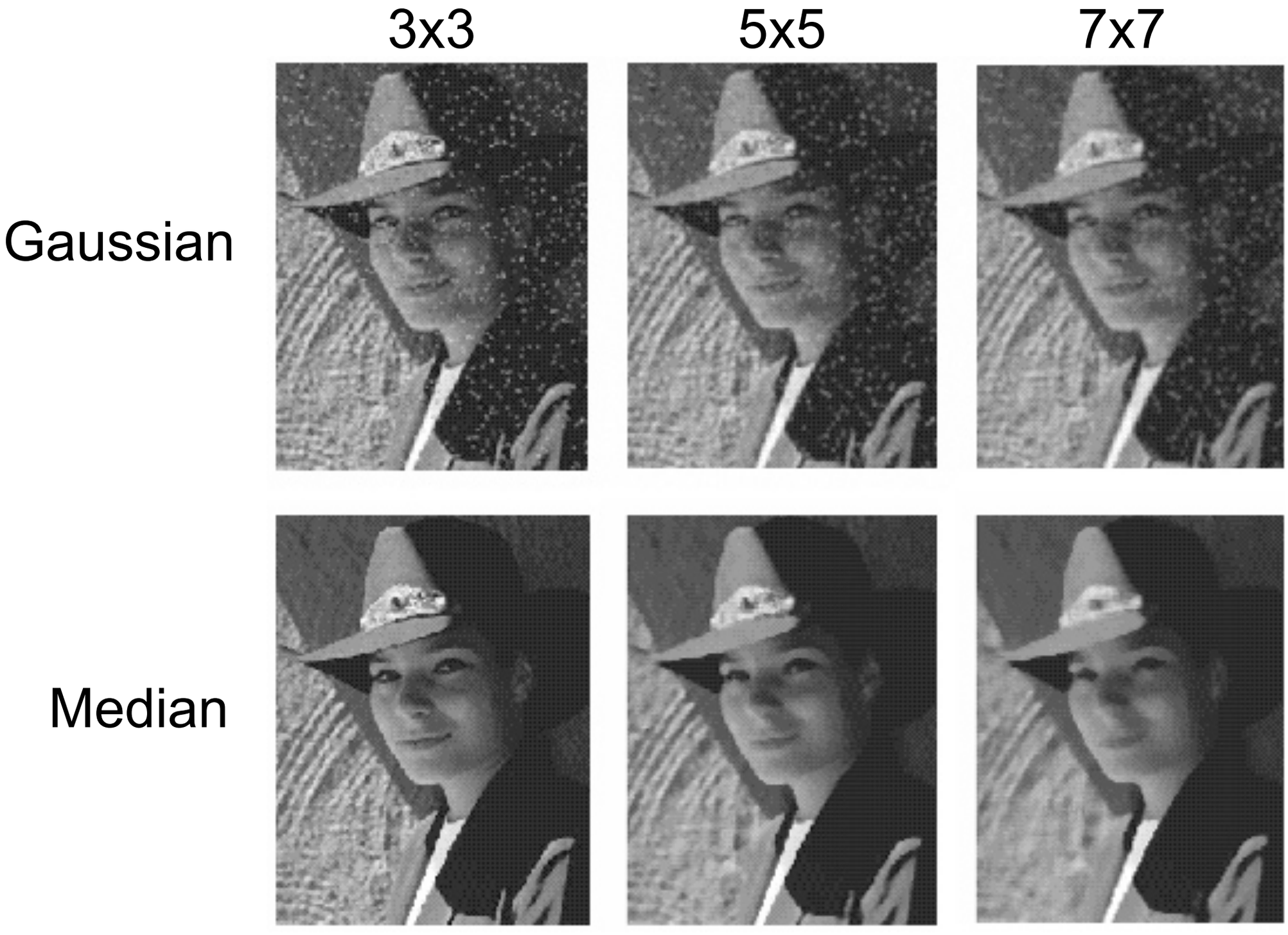
Median filter

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



Source: K. Grauman

Gaussian vs. median filtering



Source: S. Lazebnik

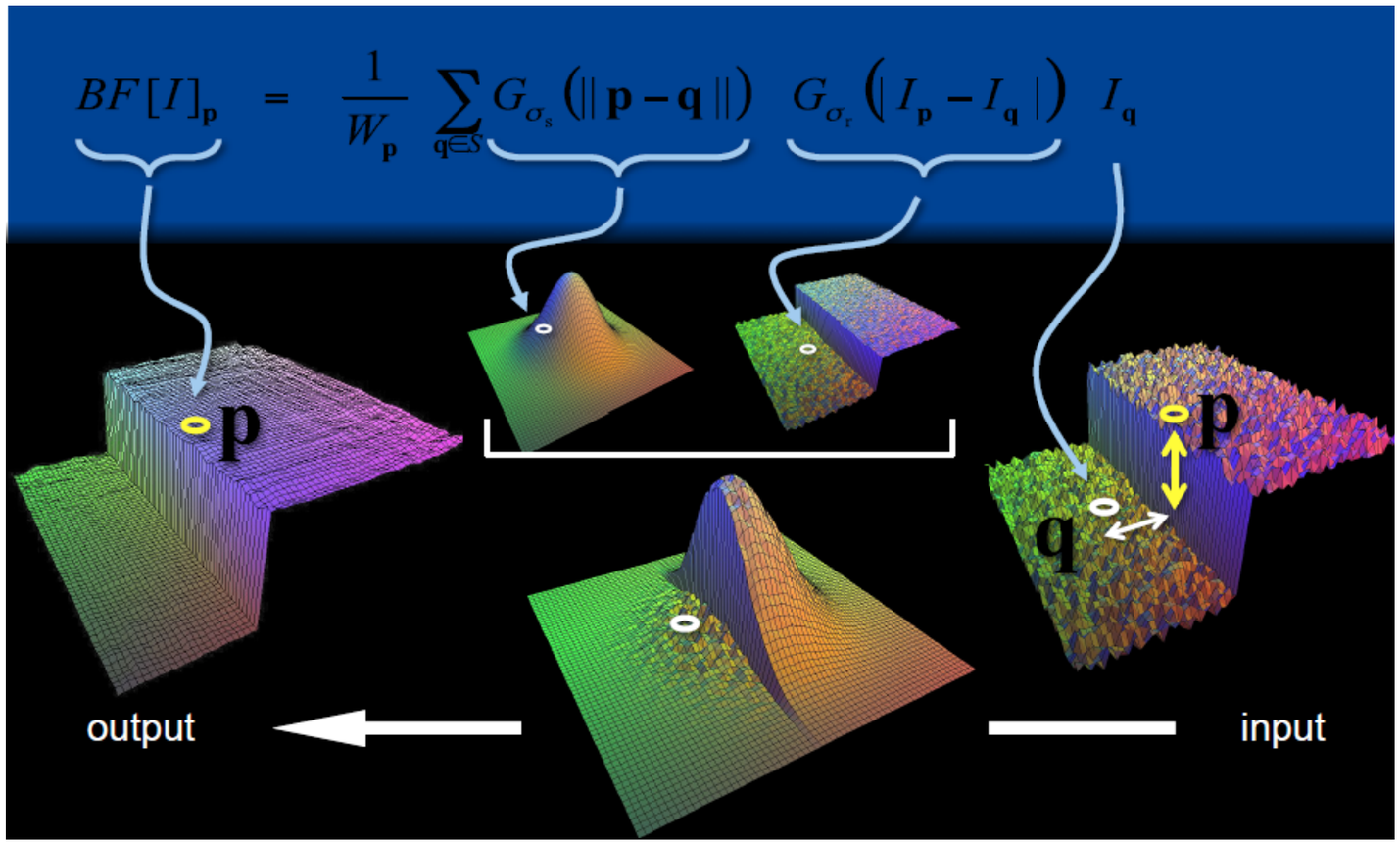
Other non-linear filters

- 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.

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

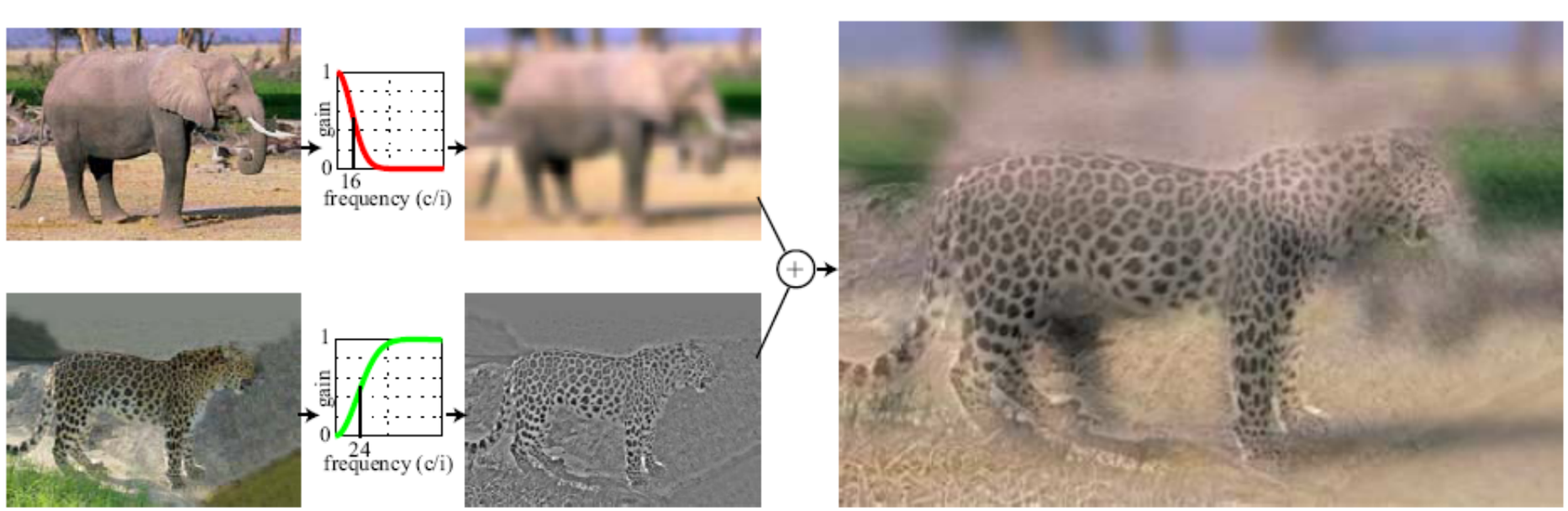
$$\text{with} \quad W^{\mathrm{b}}_{\mathbf{p}} = \sum_{\mathbf{q}\in\mathcal{S}} G_{\sigma_{\mathrm{s}}}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_{\mathrm{r}}}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)$$

Bilateral filters



Images in frequency domain

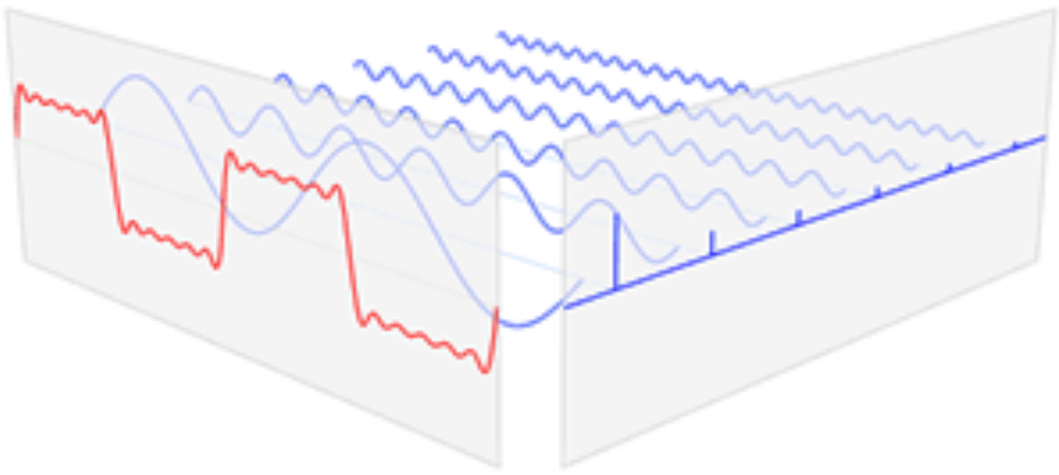
- Wide range of applications: image analysis, image filtering, image reconstruction, and image compression.



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.”



Fourier transform

- Define the Fourier transform of a function f as:

$$F(\xi) = \mathbb{F}(f(x)) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{i\xi x} dx$$

- F is a function of frequency: describes how much of each frequency is contained in f

- Fourier transform is invertible

Slide credit: Szymon Rusinkiewicz (Princeton)

Fourier transform and convolution

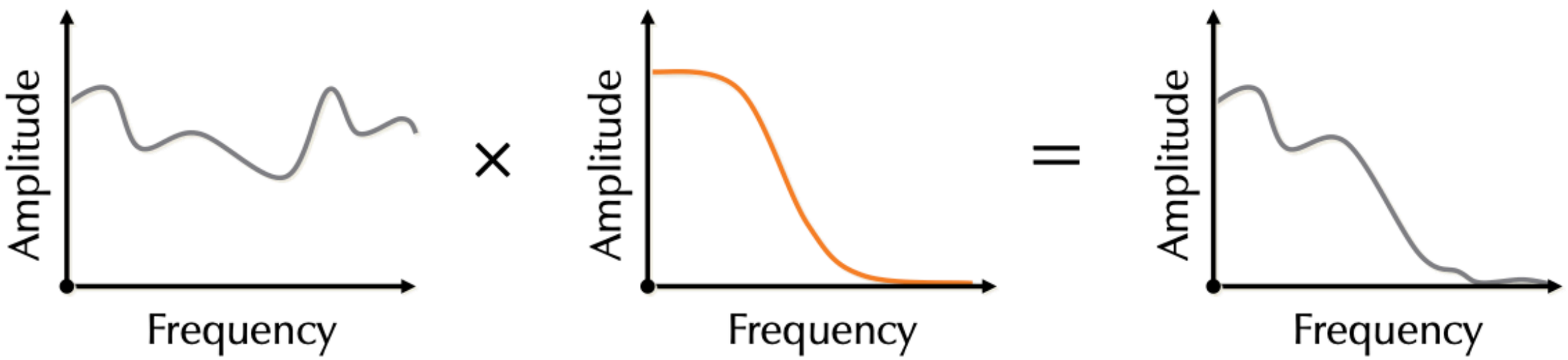
- Fourier transform turns convolution into multiplication:

$$F(f(x) \star g(x)) = F(f(x)) \star F(g(x))$$

Slide credit: Szymon Rusinkiewicz (Princeton)

Fourier transform and convolution

- Application #1: use frequency space to understand effects of filters
  - Example: Fourier transform of a Gaussian is a Gaussian
  - Thus: attenuates high frequencies



Slide credit: Szymon Rusinkiewicz (Princeton)

Fourier transform and convolution

- Application #2: Efficient computation
  - Fast Fourier transform (FFT) takes time

$$O(n \log n)$$

  - This, convolution can be performed in time

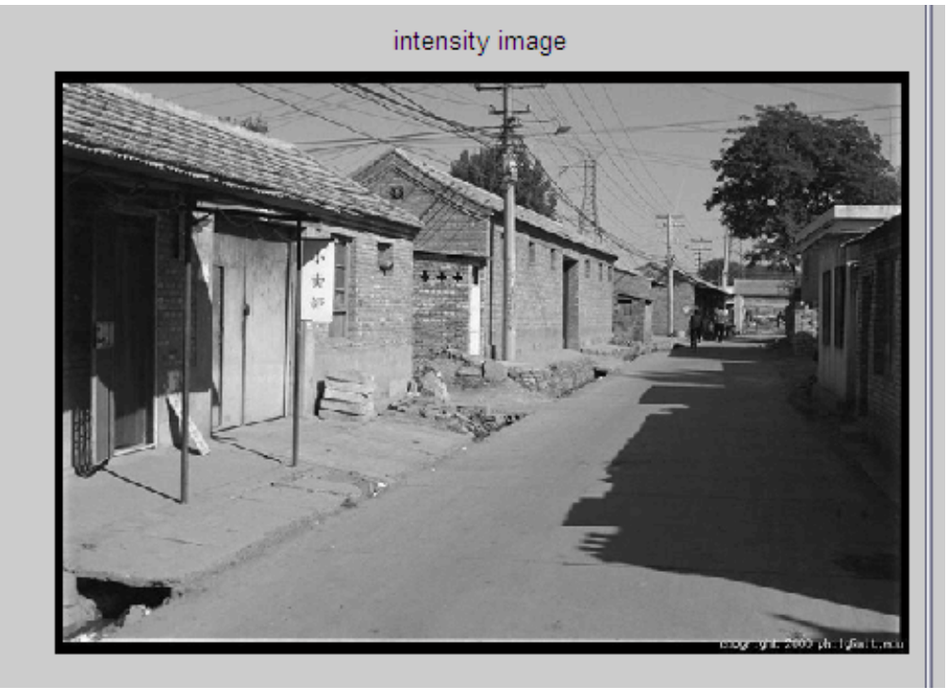
$$O(n \log n + m \log m)$$

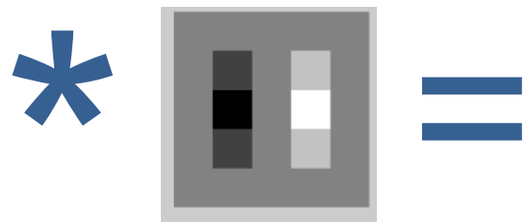
  - Greatest efficiency gains for large filters ($m \approx n$)

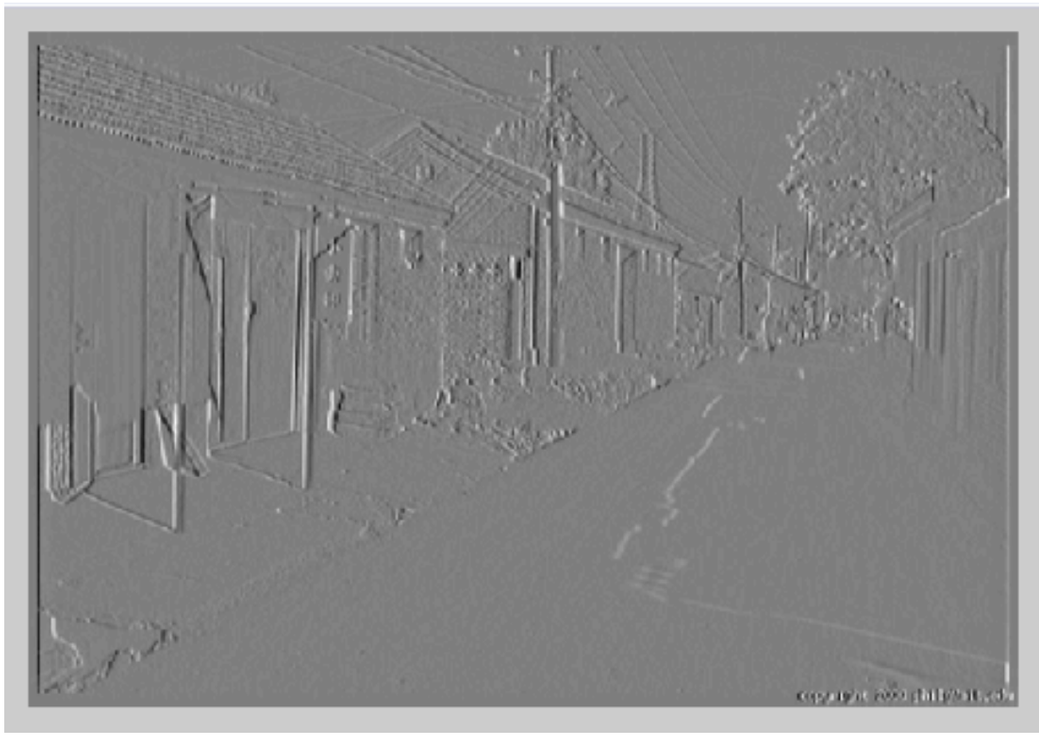
Slide credit: Szymon Rusinkiewicz (Princeton)

Filtering in spatial domain

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

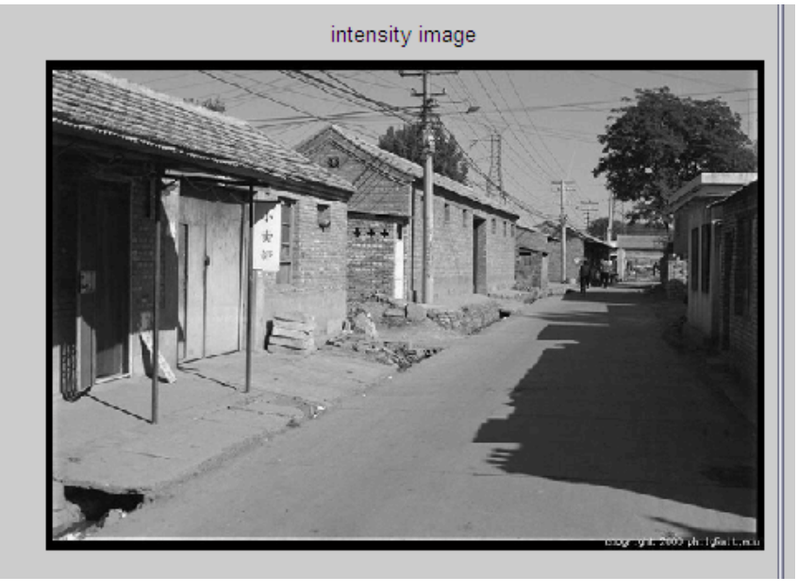




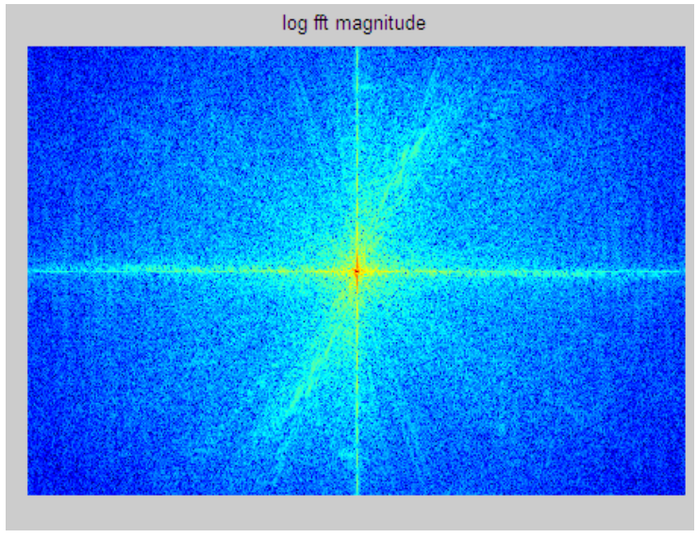


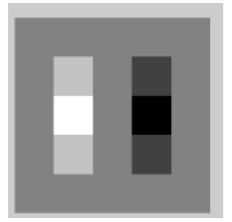
Source: D. Hoiem

Filtering in frequency domain

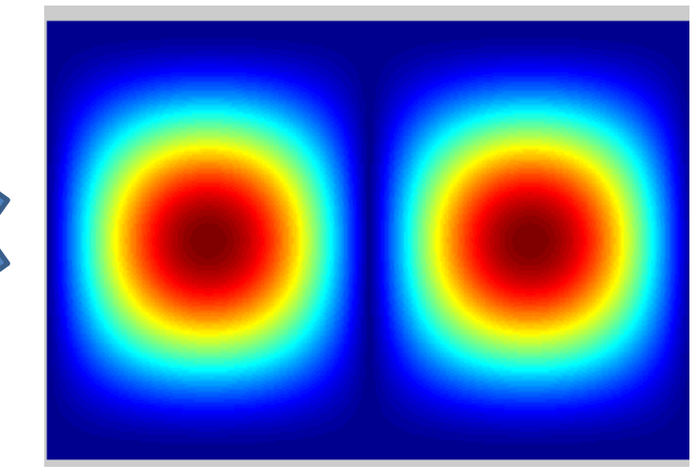


FFT

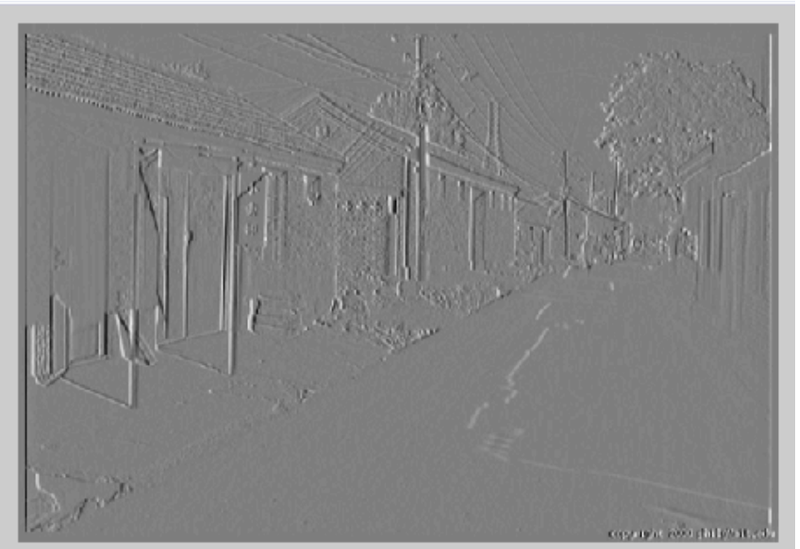




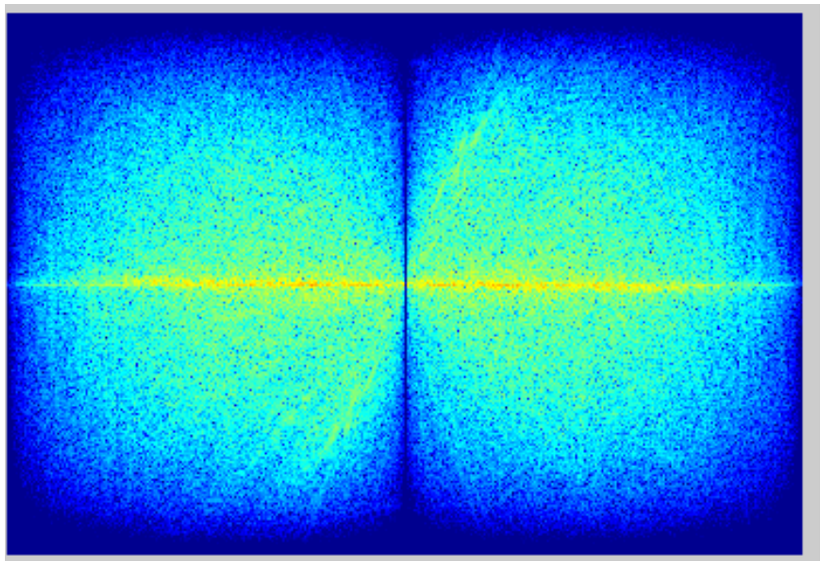
FFT



=

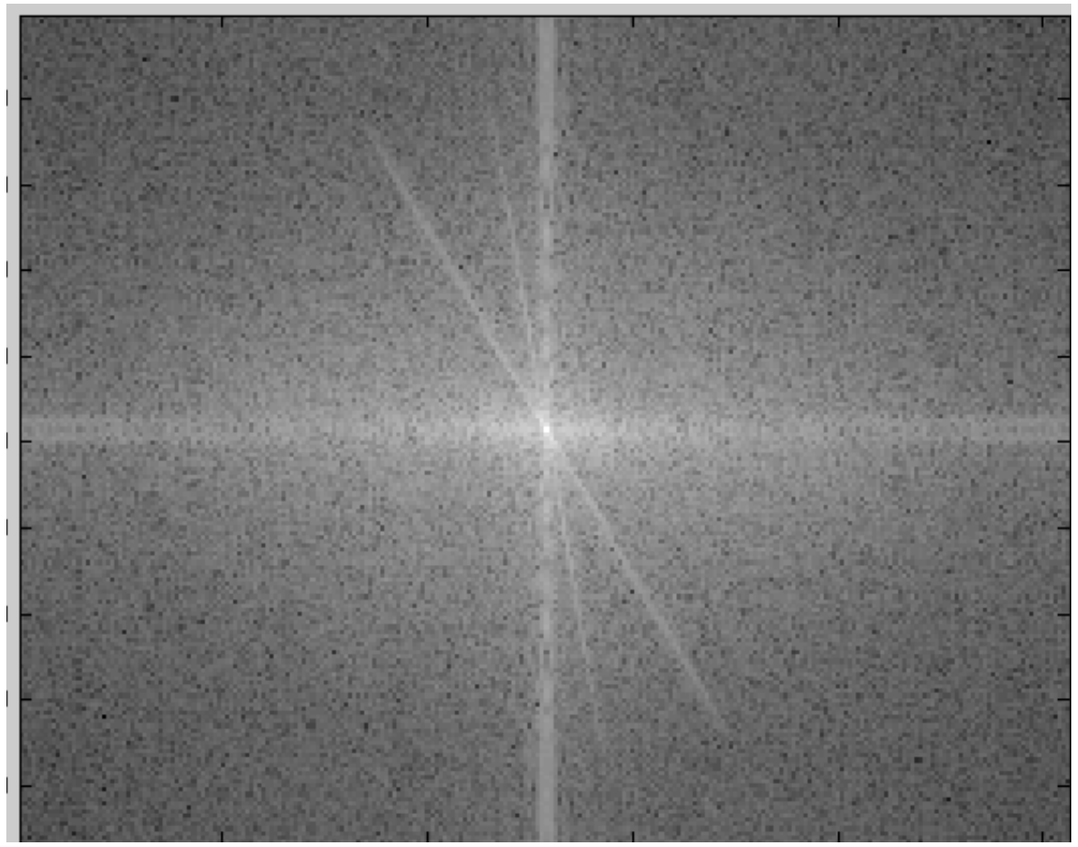


Inverse FFT



Source: D. Hoiem

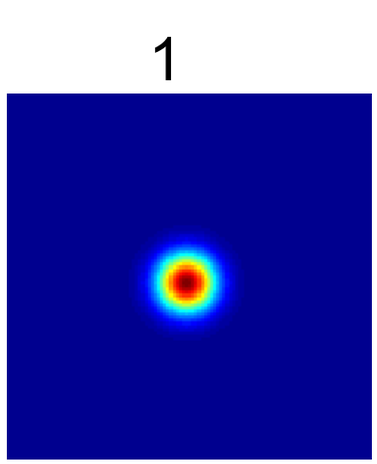
Fourier transform of a scene

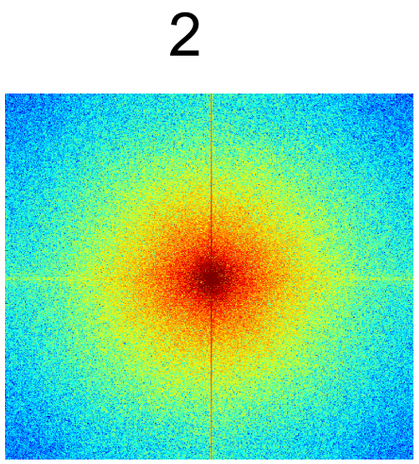


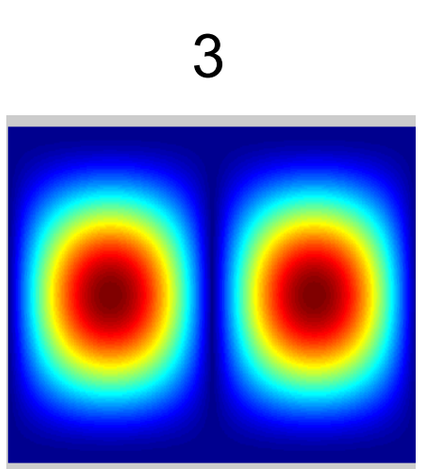
Source: J. Hays

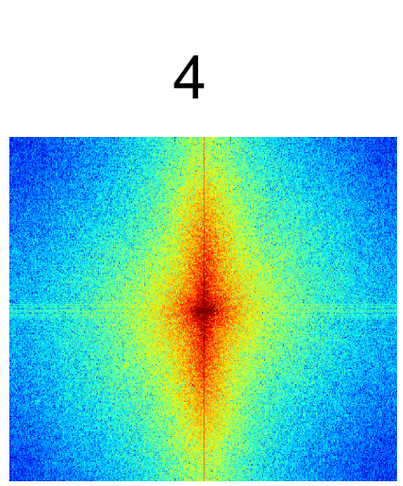
Question

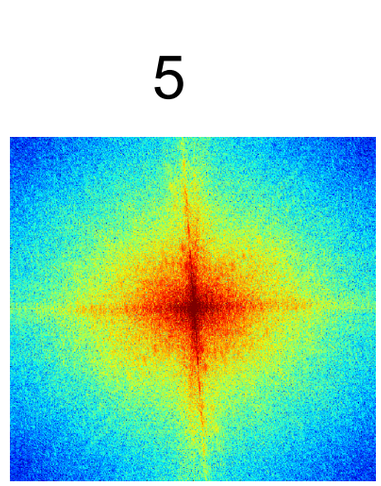
1. Match the spatial domain image to the Fourier magnitude image



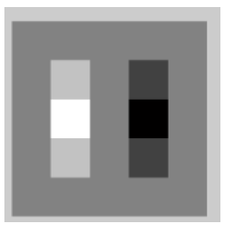








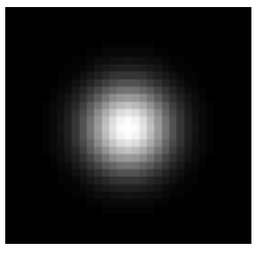
A



B

C

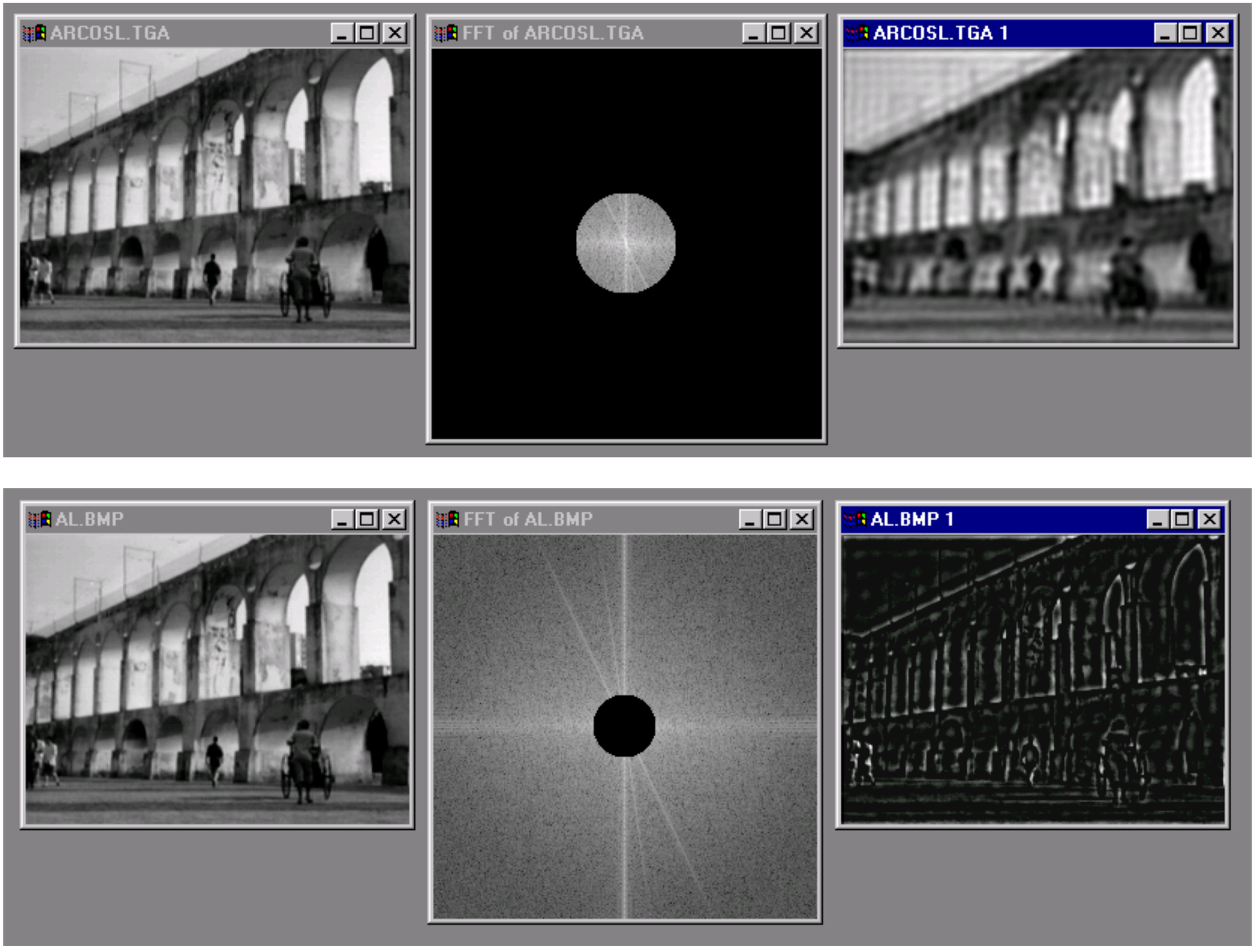
D



E

Source: Hoiem

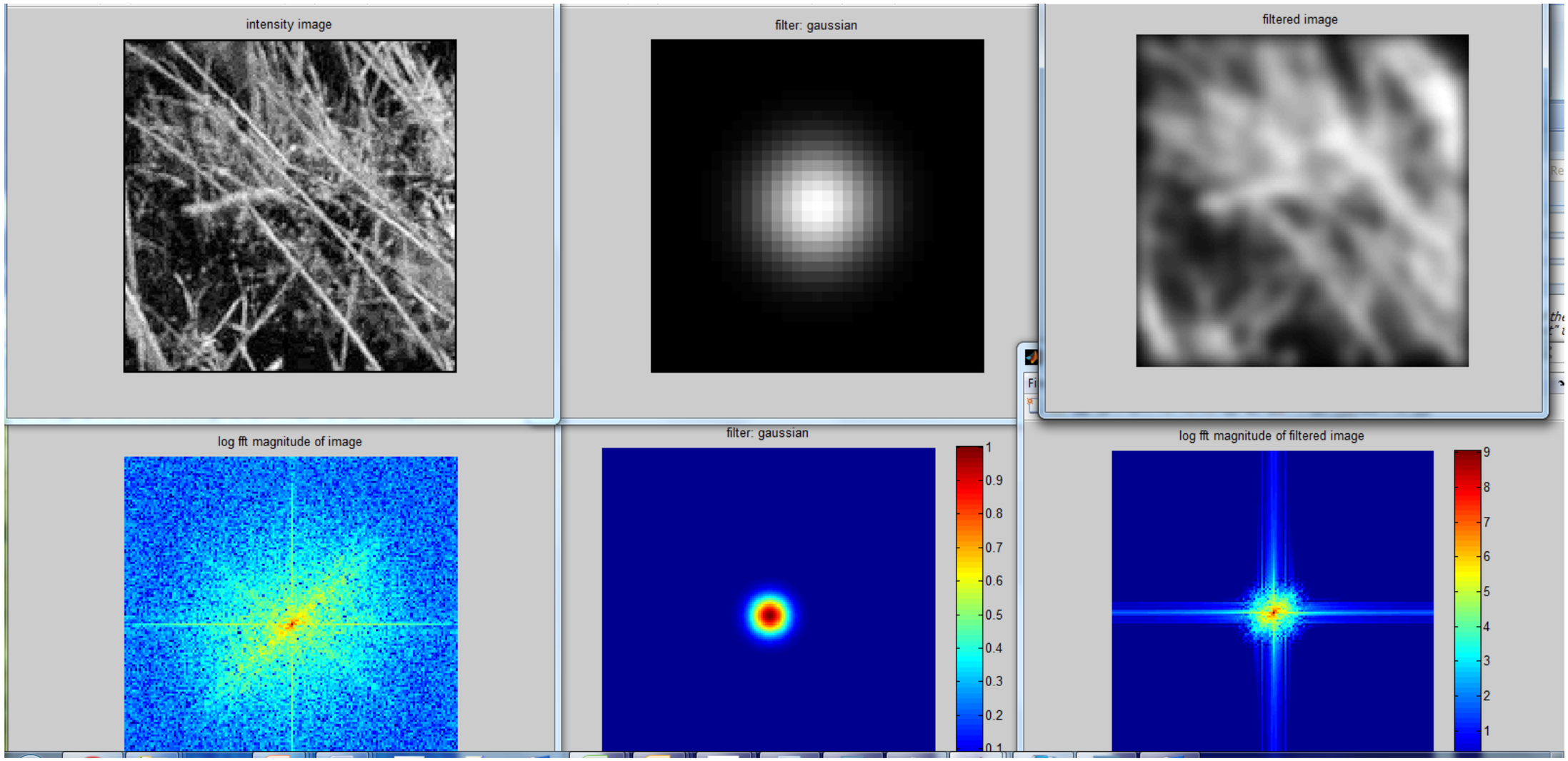
Low and high pass filtering



Source: J. Hays

Filters in frequency domain

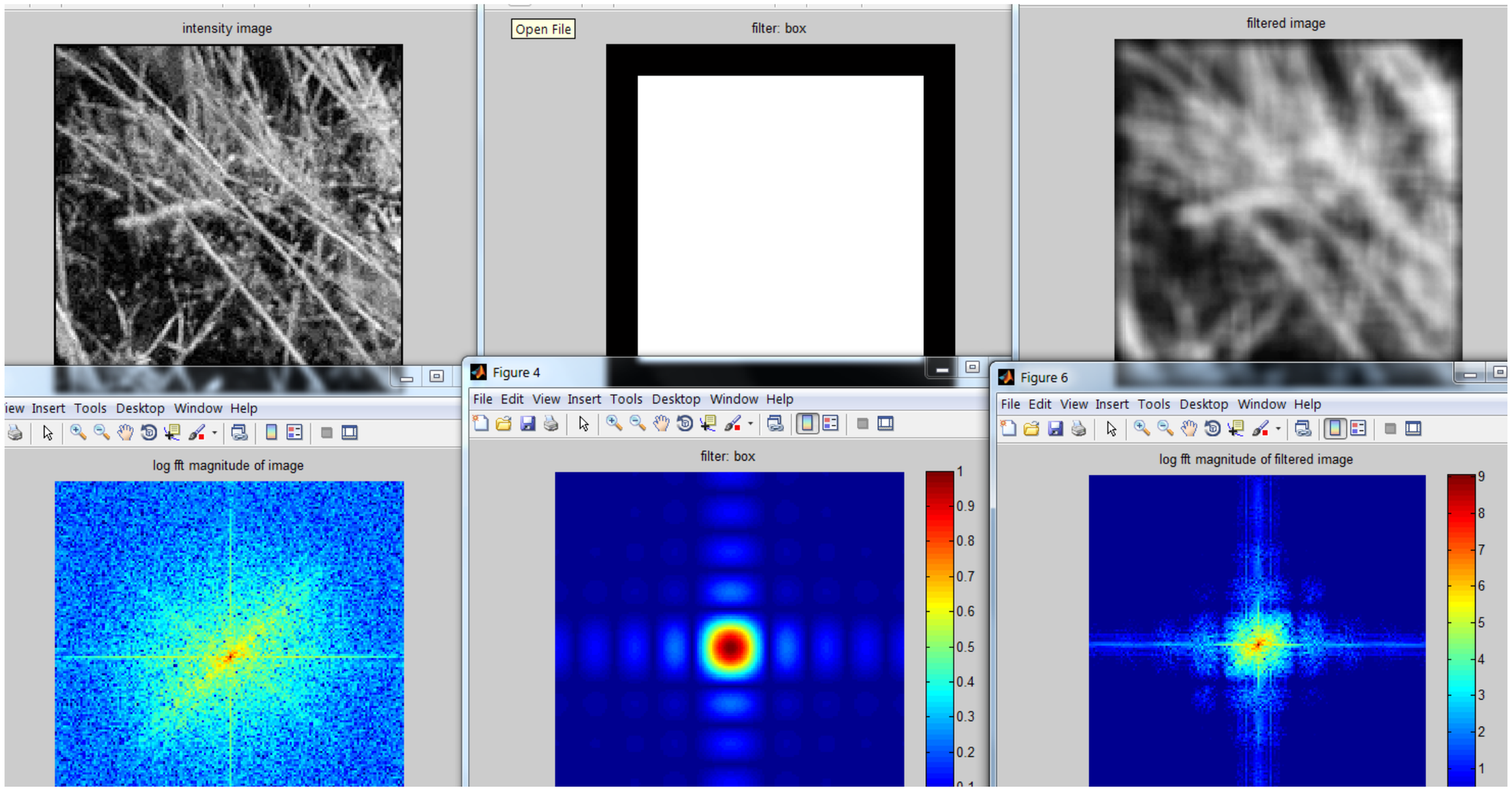
Gaussian



Source: J. Hays

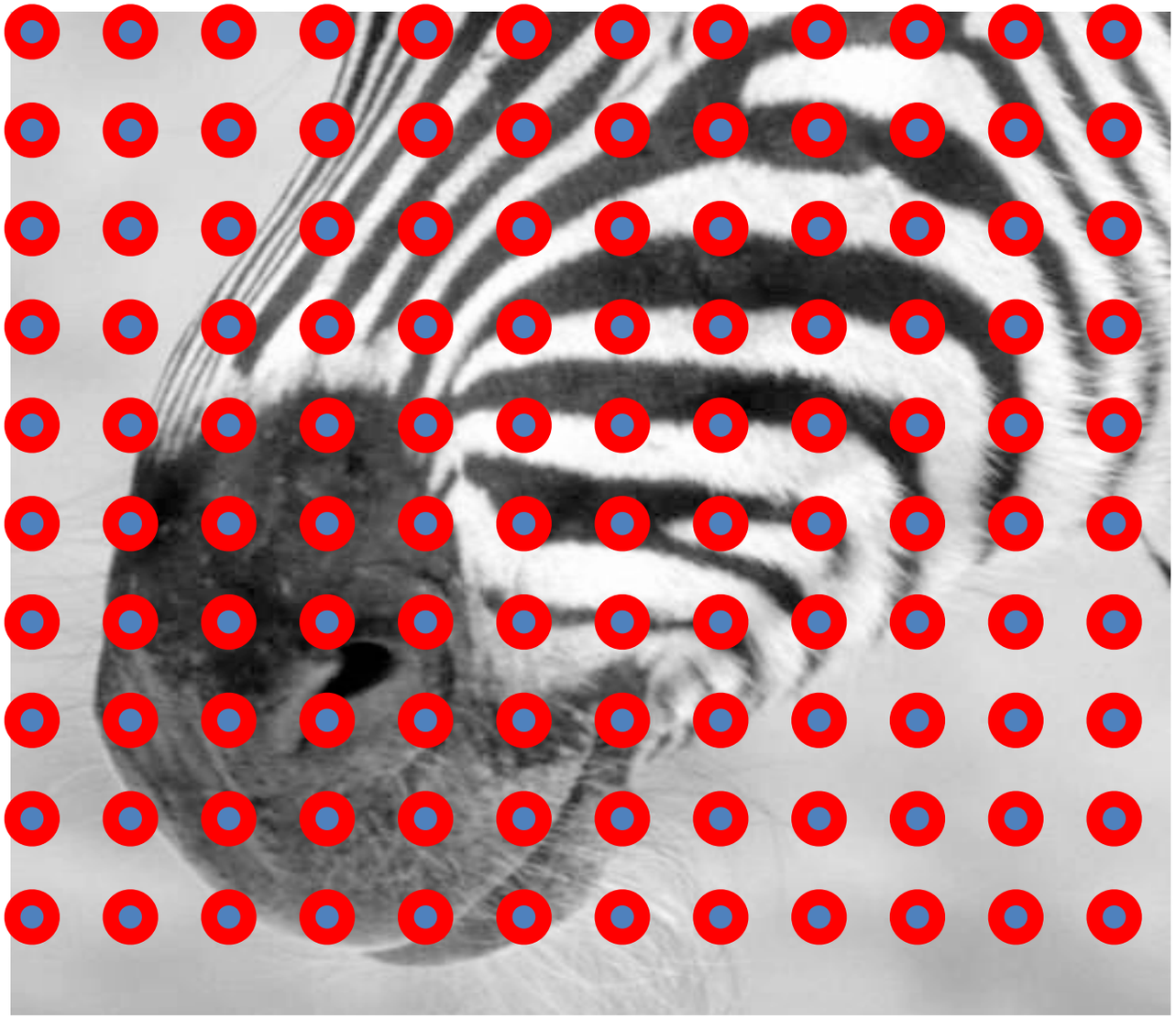
Filters in frequency domain

Box



Source: J. Hays

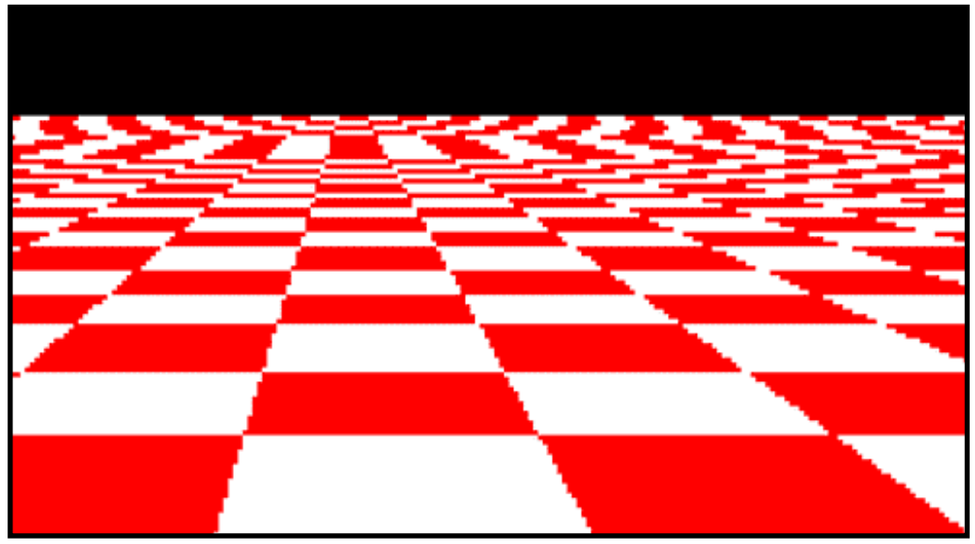
Subsampling by a factor of 2



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

Source: D. Hoiem

Aliasing

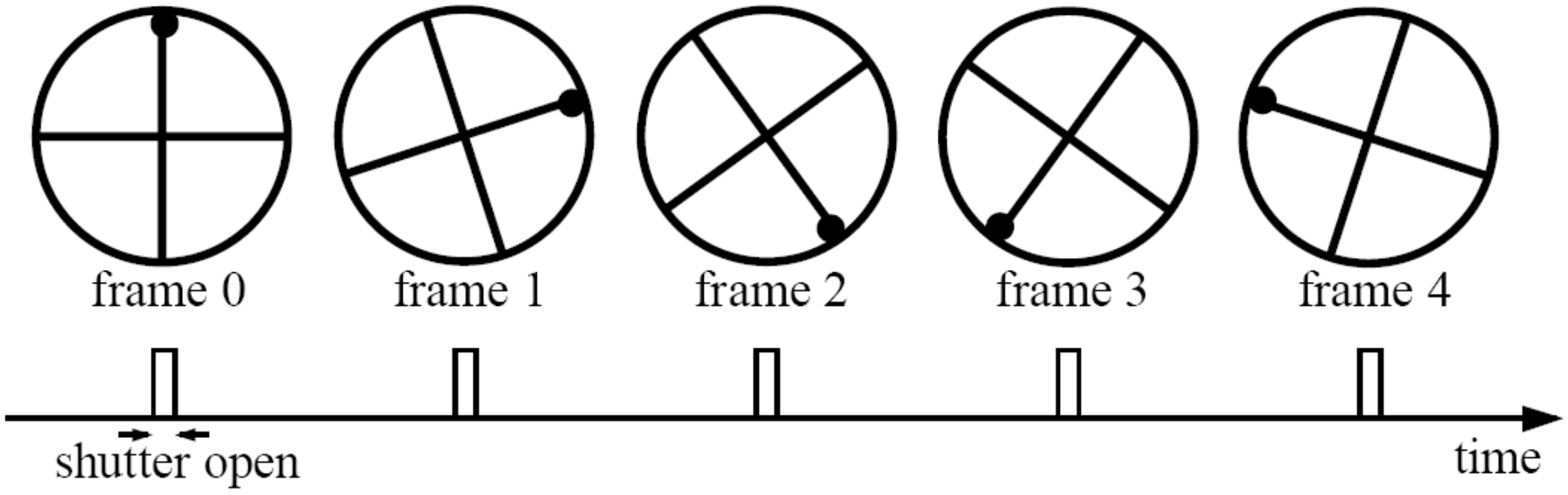


- Sub-sampling may be dangerous!
- Characteristic errors may appear:
  - Wagon wheels rolling the wrong way in movies
  - Checkerboards disintegrate in ray tracing
  - Striped shirts look funny on color television

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):

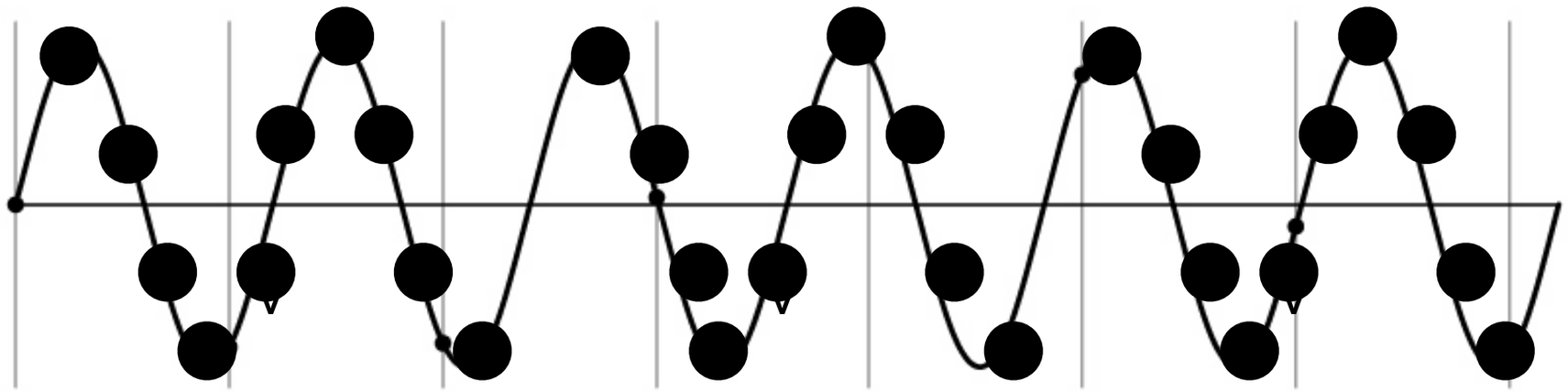


Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

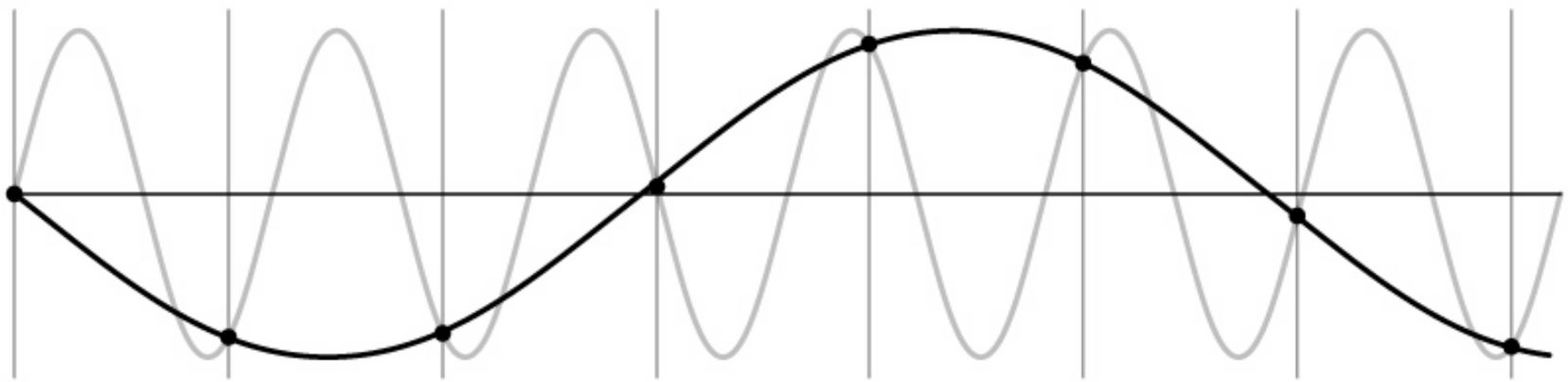
Slide by Steve Seitz

How to sample

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



good



bad

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.