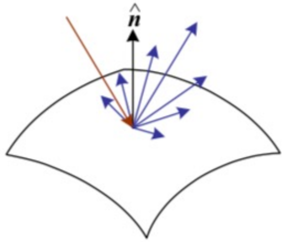


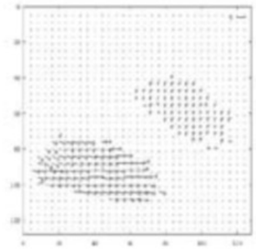
2. Image Formation

3. Image Processing

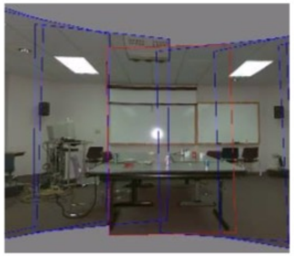
4. Features

5. Segmentation

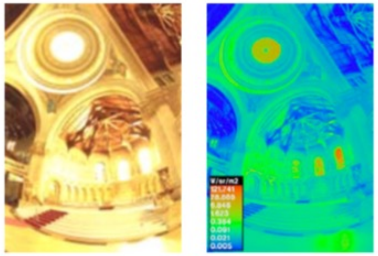
6-7. Structure from Motion



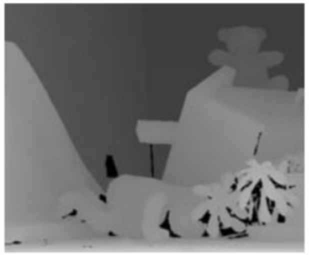
8. Motion



9. Stitching

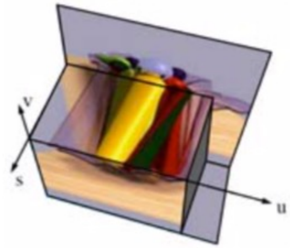


10. Computational Photography



11. Stereo

12. 3D Shape



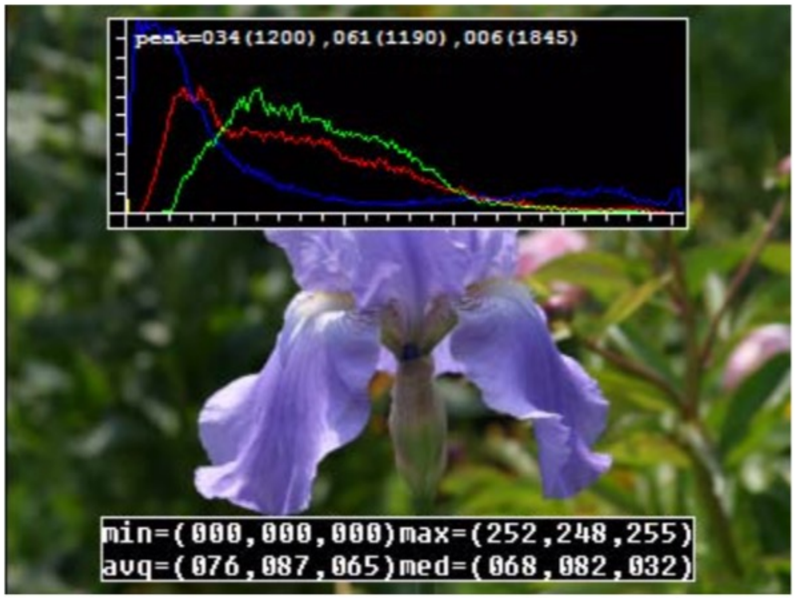
13. Image-based Rendering

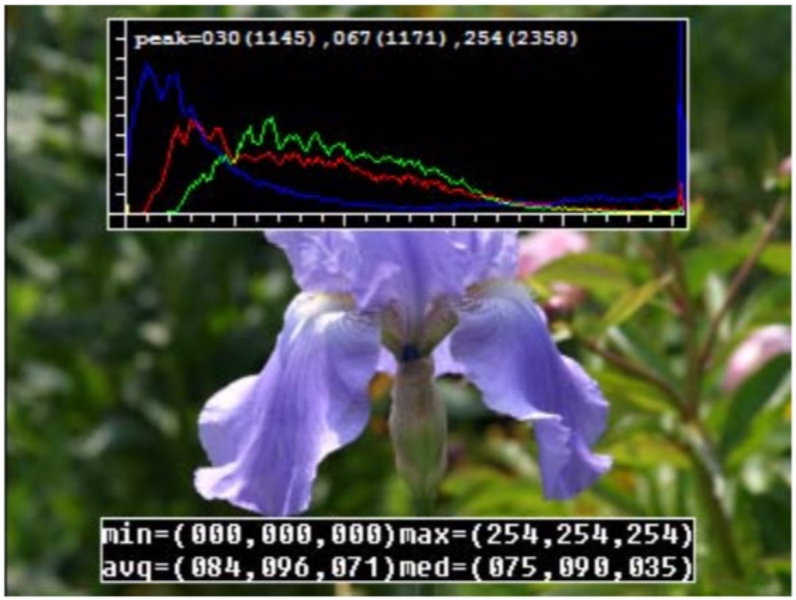
14. Recognition

3.1.1 Pixel transforms

- Contrast
- Brightness
- Gamma
- Histogram equalization
- Arithmetic
- Compositing

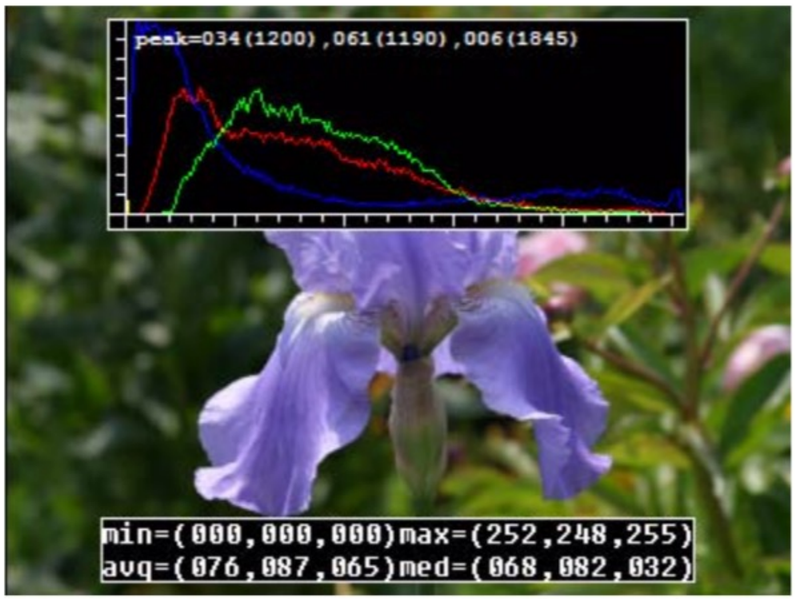
Contrast

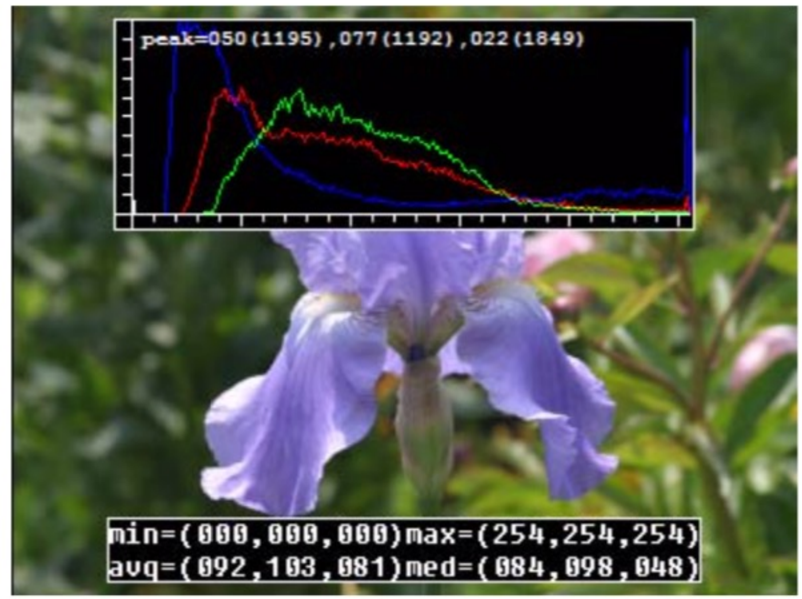




- g(x) = a f(x), a=1.1

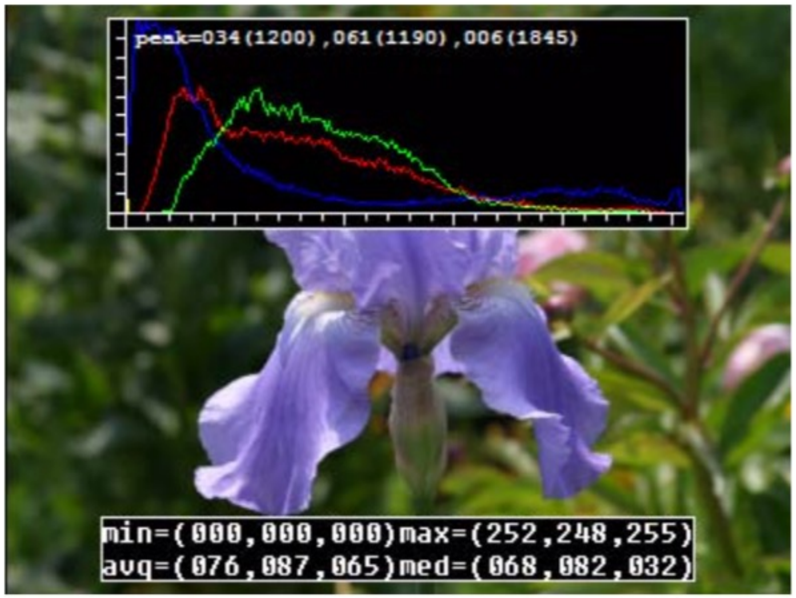
Brightness

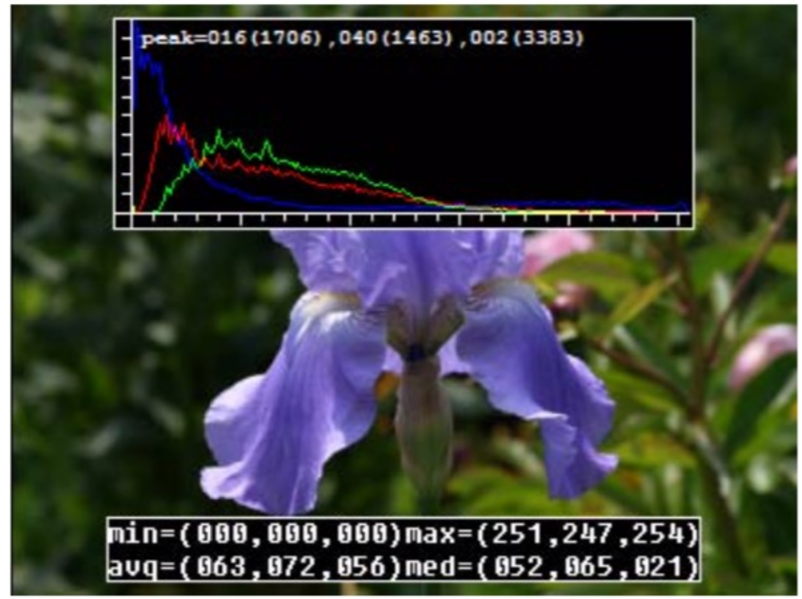




- $g(x) = f(x) + b$, $b=16$

Gamma correction

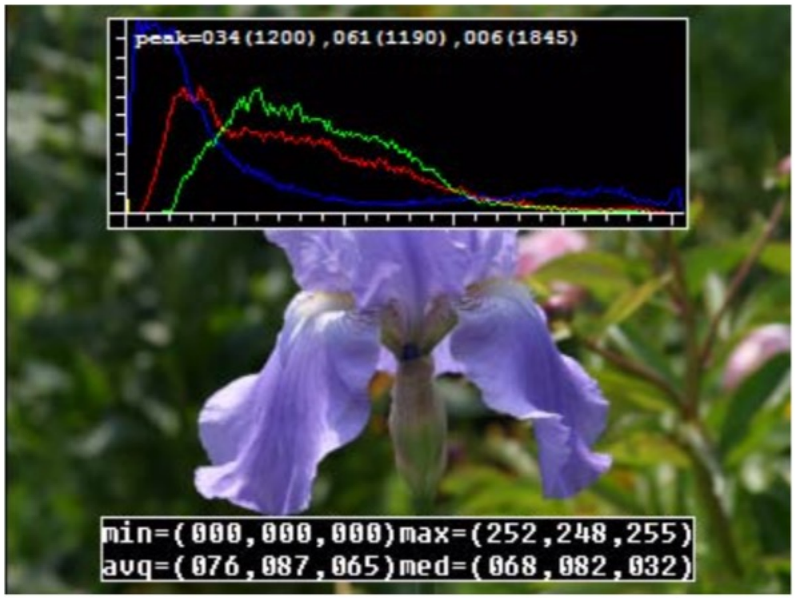


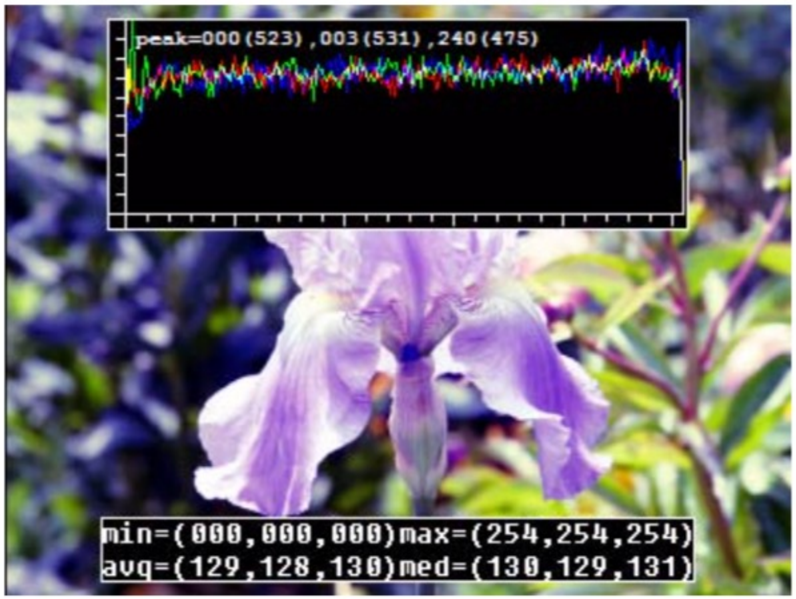


$$g(x) = [f(x)]^{1/\gamma}$$

- gamma = 1.2

Histogram Equalization





- Non-linear transform to make histogram flat
- Still a per-pixel operation $g(x) = h(f(x))$

Point-Process: Pixel/Point Arithmetic

120	122	140	142	143
121	120	141	144	147
122	121	144	146	11
125	121	144	145	10
126	121	145	147	13

+

120	122	140	142	143
121	80	40	144	10
122	81	40	0	151
125	80	40	0	152
126	70	40	0	153

=

240	244	280	284	286
121	200	181	288	157
122	202	184	146	162
125	201	184	145	164
126	191	185	147	166

120	122	140	142	143
121	120	141	144	147
122	121	144	146	11
125	121	144	145	10
126	121	145	147	13

–

120	122	140	142	143
121	80	40	144	10
122	81	40	0	151
125	80	40	0	152
126	70	40	0	153

=

0	0	0	0	0
0	40	101	0	137
0	40	104	146	-140
0	40	104	145	-142
0	191	185	147	-140

Pixel/Point Arithmetic: An Example

Image 1

-

Image 2

=

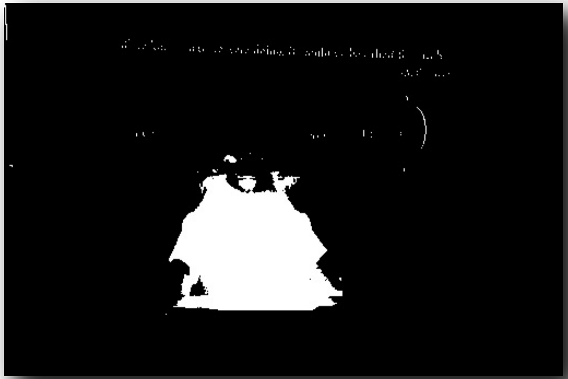


Image 1 - Image 2

Binary(Image 1 - Image 2)

Image filtering

- Image filtering: compute function of local neighborhood at each position
- Really important!
  - Enhance images
    - Denoise, resize, increase contrast, etc.
  - Extract information from images
    - Texture, edges, distinctive points, etc.
  - Detect patterns
    - Template matching
  - Convolutional Networks

Example: box filter

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

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

Slide credit: David Lowe (UBC)

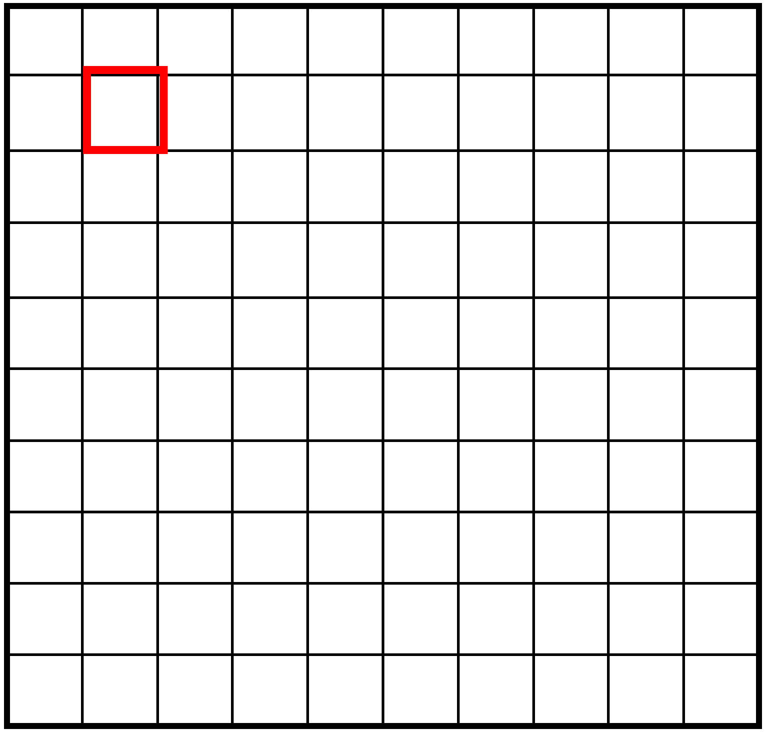
Image filtering

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

$f[.,.]$

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

$h[.,.]$



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

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

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

$f[.,.]$

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					

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

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

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

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	30				
						?			
				50					

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

Credit: S. Seitz

Image filtering

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

$f[.,.]$

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?

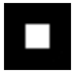
- Replaces each pixel with an average of its neighborhood
- Achieve smoothing effect (remove sharp features)

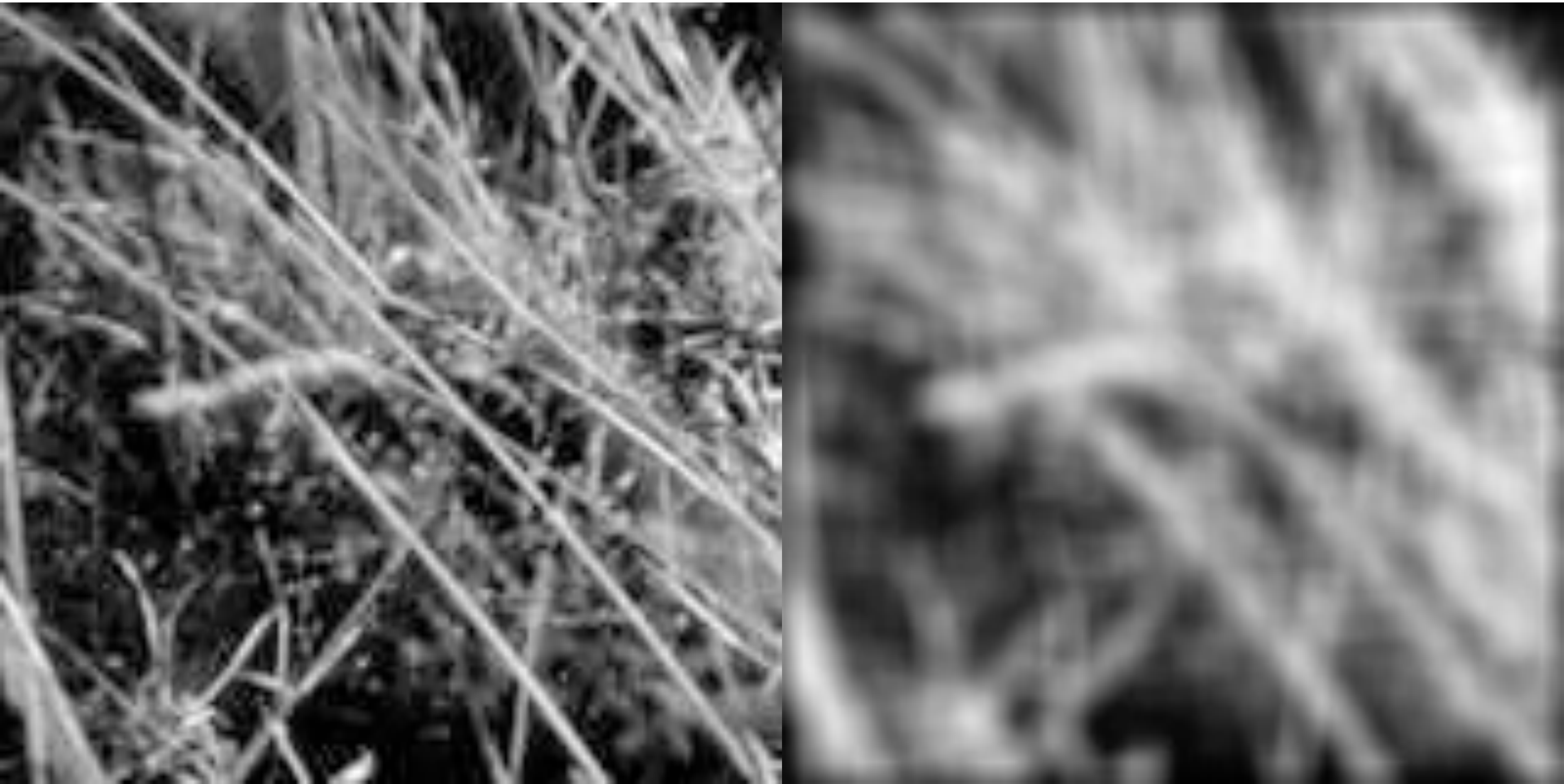
$$g[\cdot\,,\cdot\,]$$

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

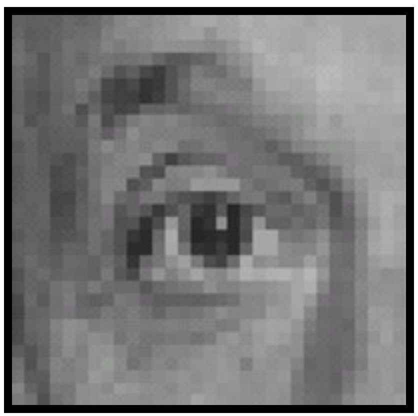
Slide credit: David Lowe (UBC)

Smoothing with box filter





Practice with linear filters



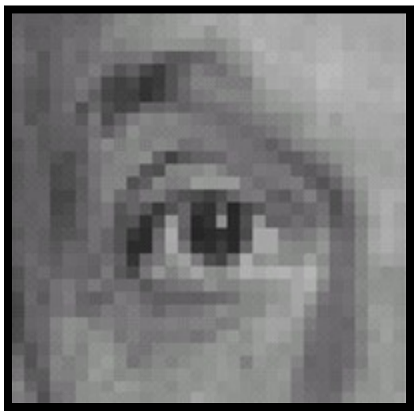
Original

0	0	0
0	1	0
0	0	0

?

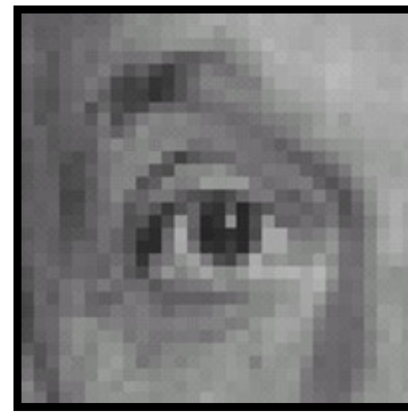
Source: D. Lowe

Practice with linear filters



Original

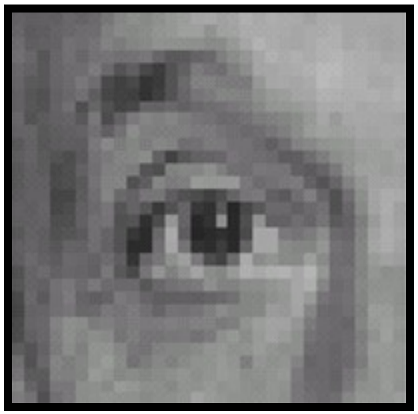
0	0	0
0	1	0
0	0	0



Filtered
(no change)

Source: D. Lowe

Practice with linear filters



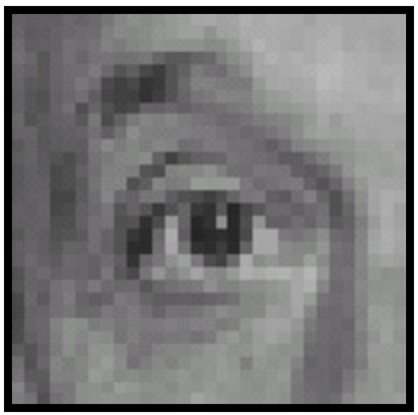
Original

0	0	0
0	0	1
0	0	0

?

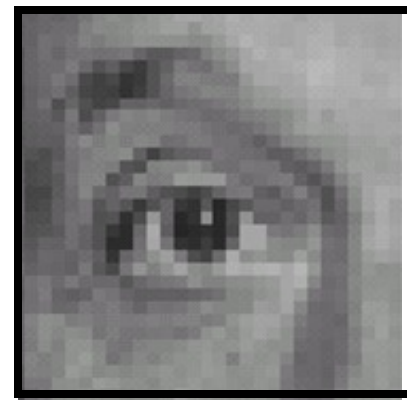
Source: D. Lowe

Practice with linear filters



Original

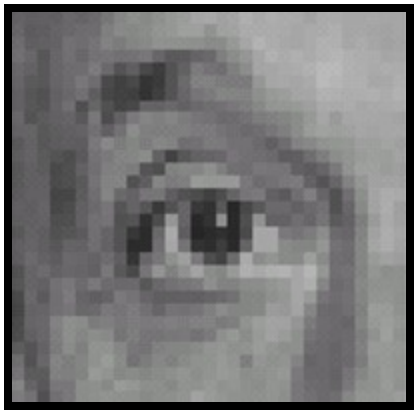
0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

Source: D. Lowe

Practice with linear filters



Original

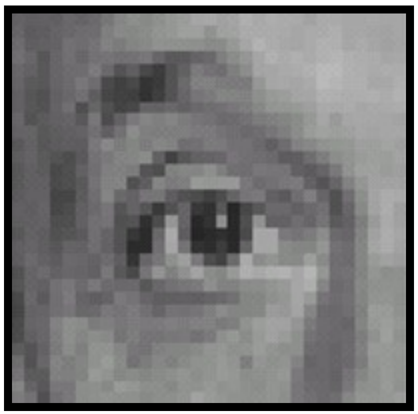
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

?

(Note that filter sums to 1)

Source: D. Lowe

Practice with linear filters



Original

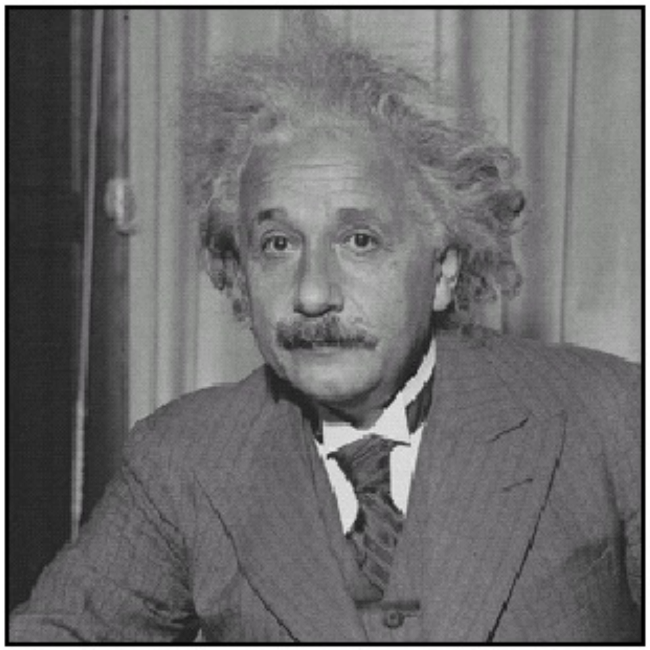
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Sharpening filter

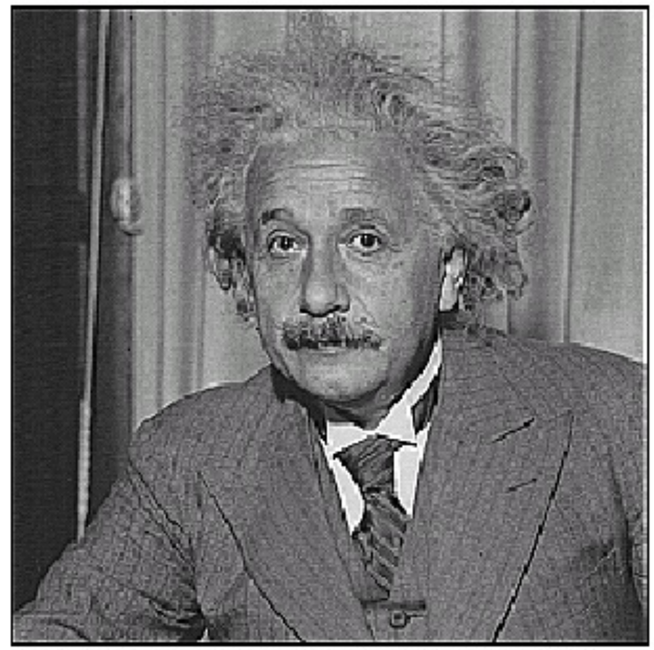
- Accentuates differences with local average

Source: D. Lowe

Sharpening



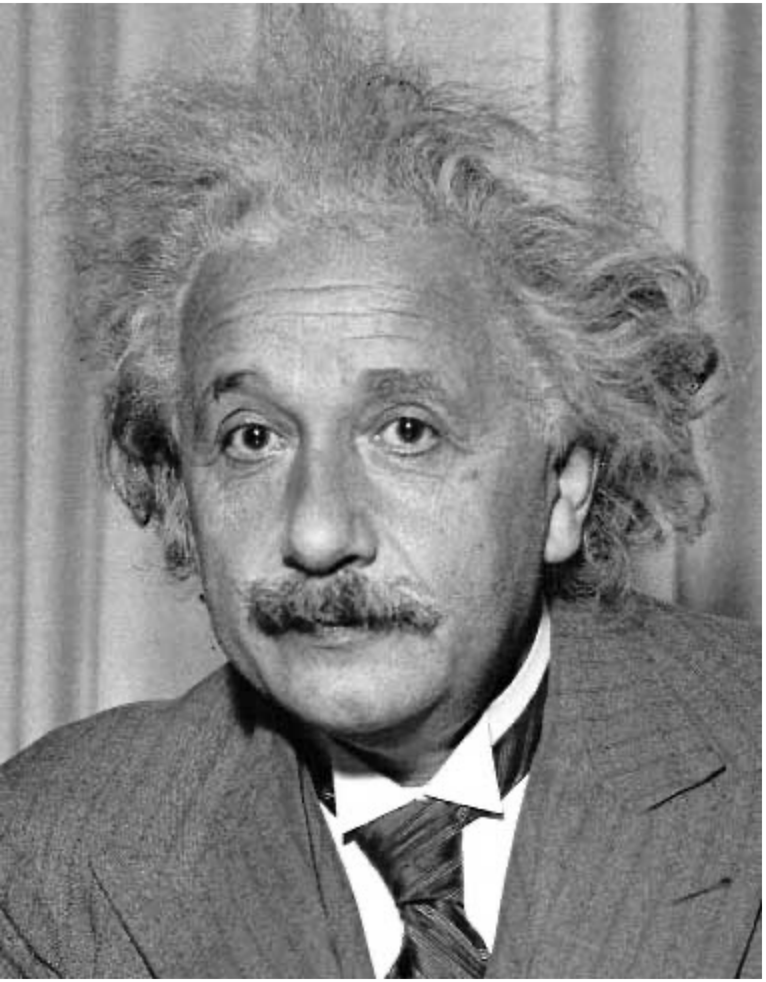
before



after

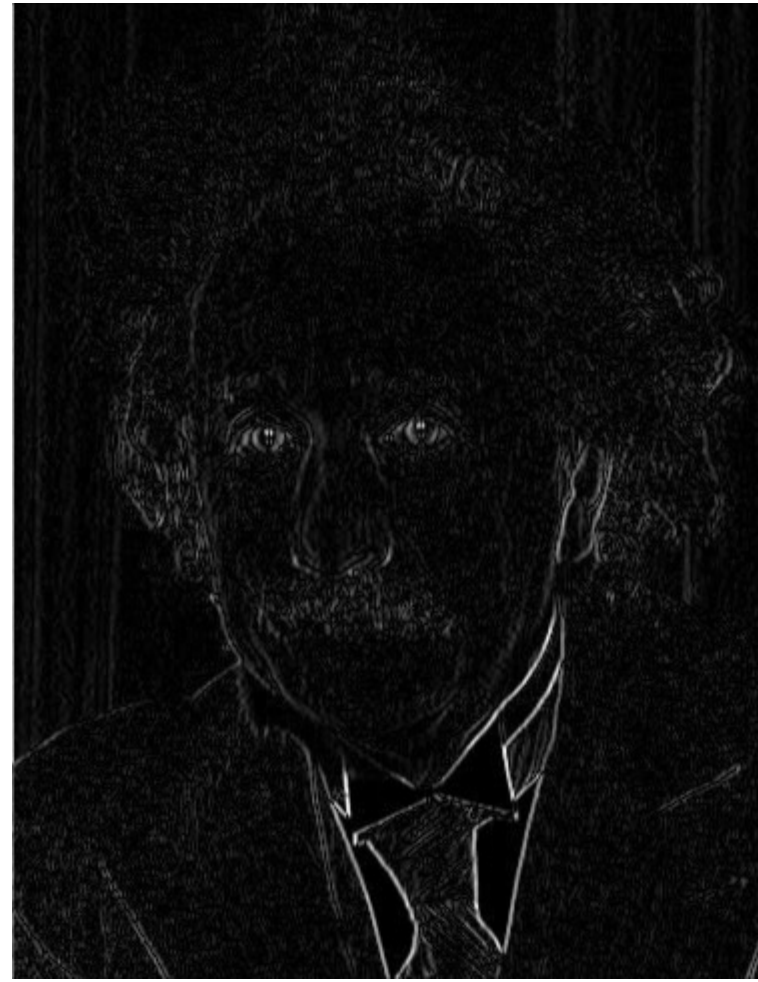
Source: D. Lowe

Other filters



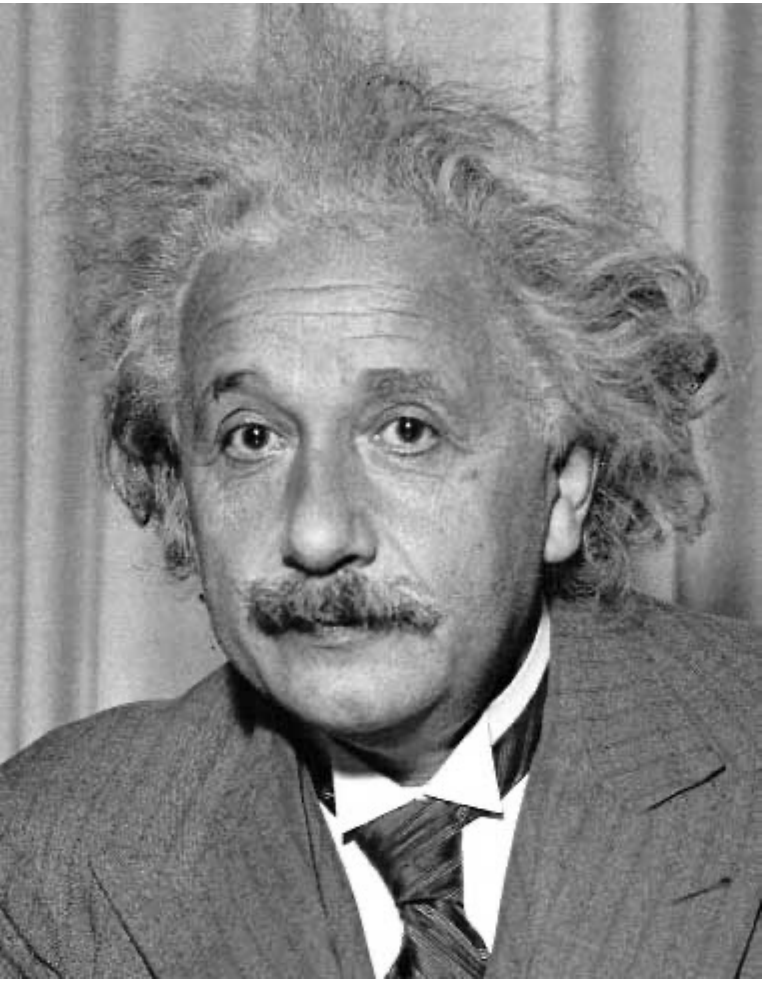
1	0	-1
2	0	-2
1	0	-1

Sobel



Vertical Edge
(absolute value)

Other filters



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

Sobel

Horizontal Edge
(absolute value)

Filtering vs. Convolution

- 2d filtering `f=filter` `I=image`
  - `h=filter2(f,I);` or `h=imfilter(I,f);`

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

- 2d convolution
  - `h=conv2(f,I);`

$$h[m,n]=\sum_{k,l} f[k,l]\,I[m-k,n-l]$$

Key properties of linear filters

Linearity:

```
imfilter(I, f₁ + f₂) =
   imfilter(I,f₁) + imfilter(I,f₂)
```

Shift invariance: same behavior regardless of pixel location

```
imfilter(I,shift(f)) = shift(imfilter(I,f))
```

Any linear, shift-invariant operator can be represented as a convolution

Source: S. Lazebnik

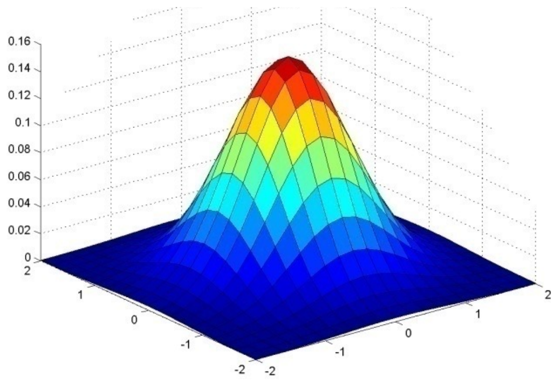
More properties

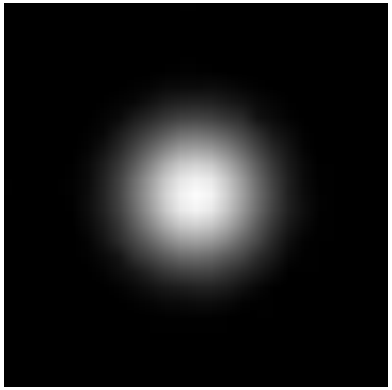
- Commutative: $a * b = b * a$
  - Conceptually no difference between filter and signal
  - But particular filtering implementations might break this equality
- Associative: $a * (b * c) = (a * b) * c$
  - Often apply several filters one after another: $(((a * b_1) * b_2) * b_3)$
  - This is equivalent to applying one filter: a * $(b_1 * b_2 * b_3)$
- Distributes over addition: $a * (b + c) = (a * b) + (a * c)$
- Scalars factor out: $ka * b = a * kb = k\,(a * b)$
- Identity: unit impulse e = [0, 0, 1, 0, 0],
  $a * e = a$

Source: S. Lazebnik

Important filter: Gaussian

- Weight contributions of neighboring pixels by nearness





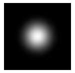
0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

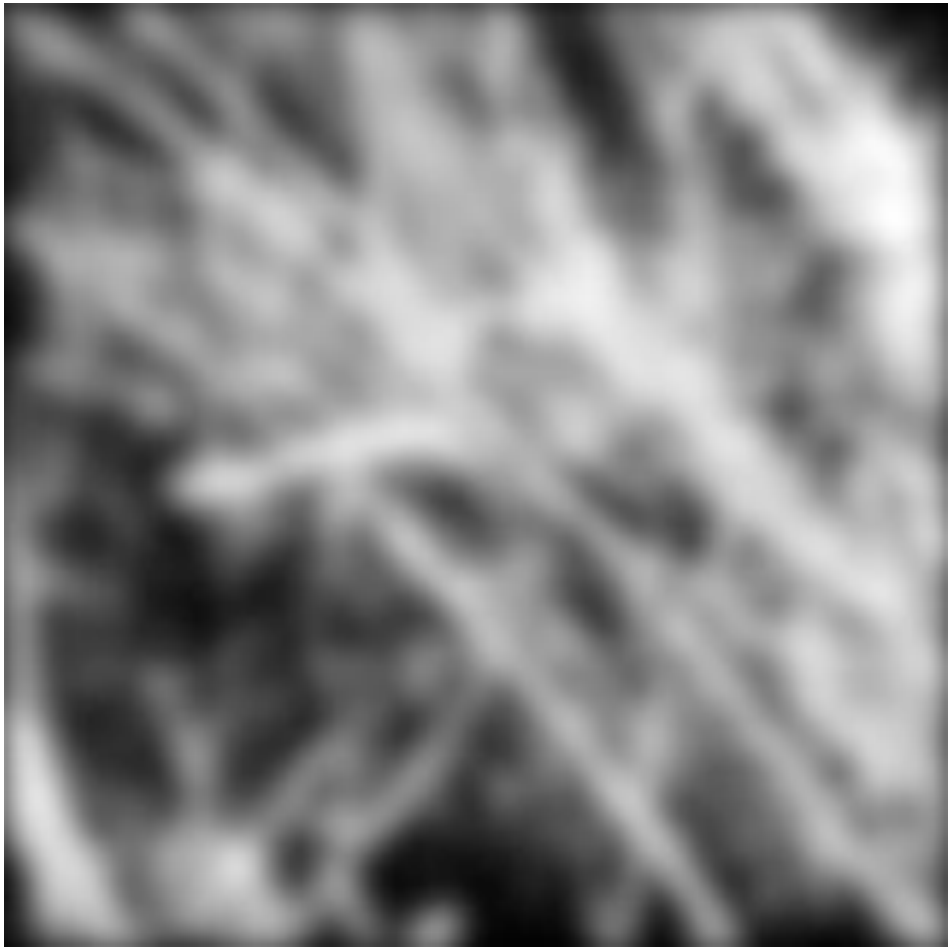
5 x 5, σ = 1

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

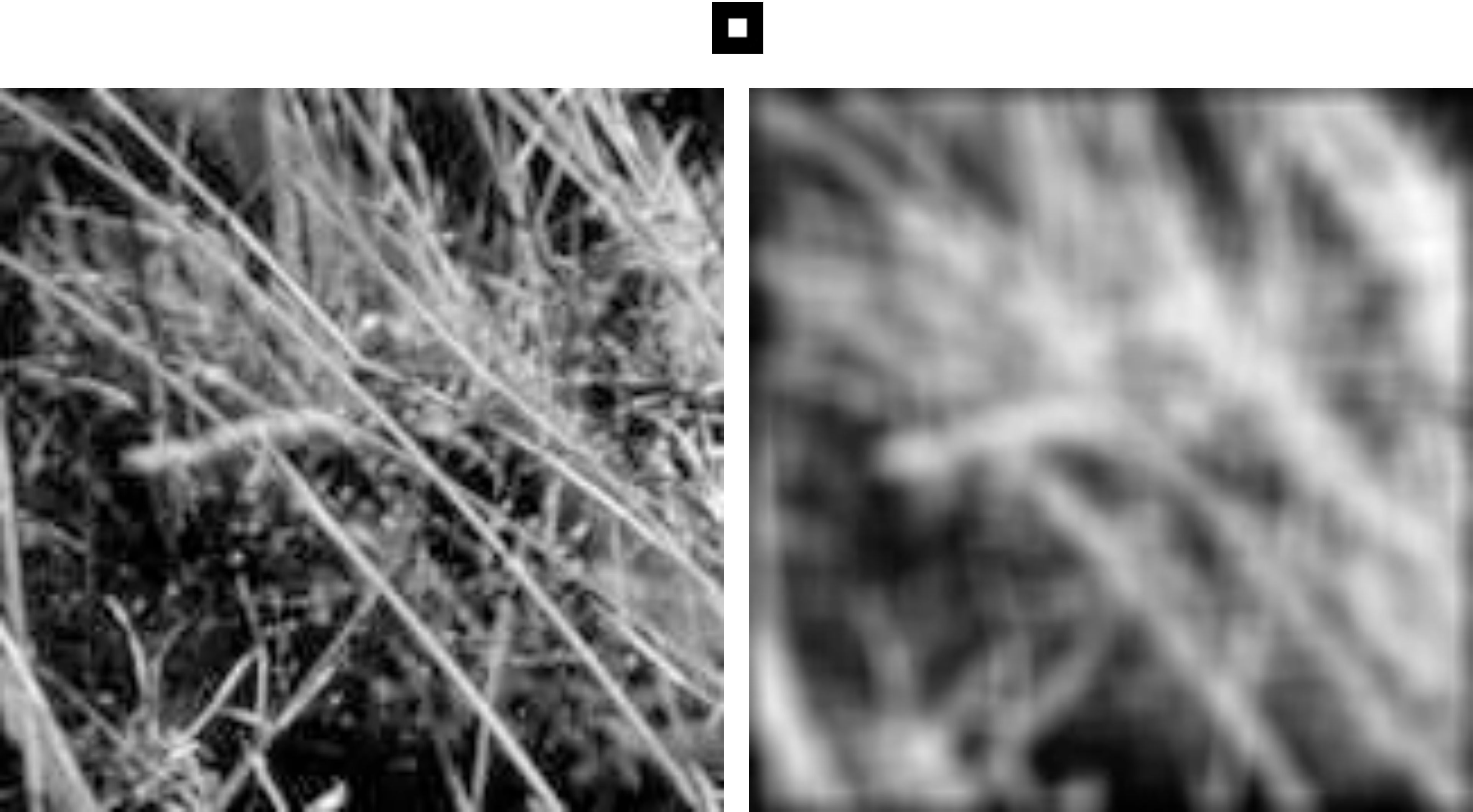
Slide credit: Christopher Rasmussen

Smoothing with Gaussian filter





Smoothing with box filter



Gaussian filters

- Remove "high-frequency" components from the image (low-pass filter)
  - Images become more smooth
- Convolution with self is another Gaussian
  - So can smooth with small-width kernel, repeat, and get same result as larger-width kernel would have
  - Convolving two times with Gaussian kernel of width σ is same as convolving once with kernel of width $\sigma\sqrt{2}$
- *Separable* kernel
  - Factors into product of two 1D Gaussians

Source: K. Grauman

Separability of the Gaussian filter

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

$$= \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{x^2}{2\sigma^2}}\right) \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{y^2}{2\sigma^2}}\right)$$

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

Separability example

2D convolution (center location only)

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 2 & 3 & 3 \\ 3 & 5 & 5 \\ 4 & 4 & 6 \end{bmatrix}$$

The filter factors into a product of 1D filters:

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

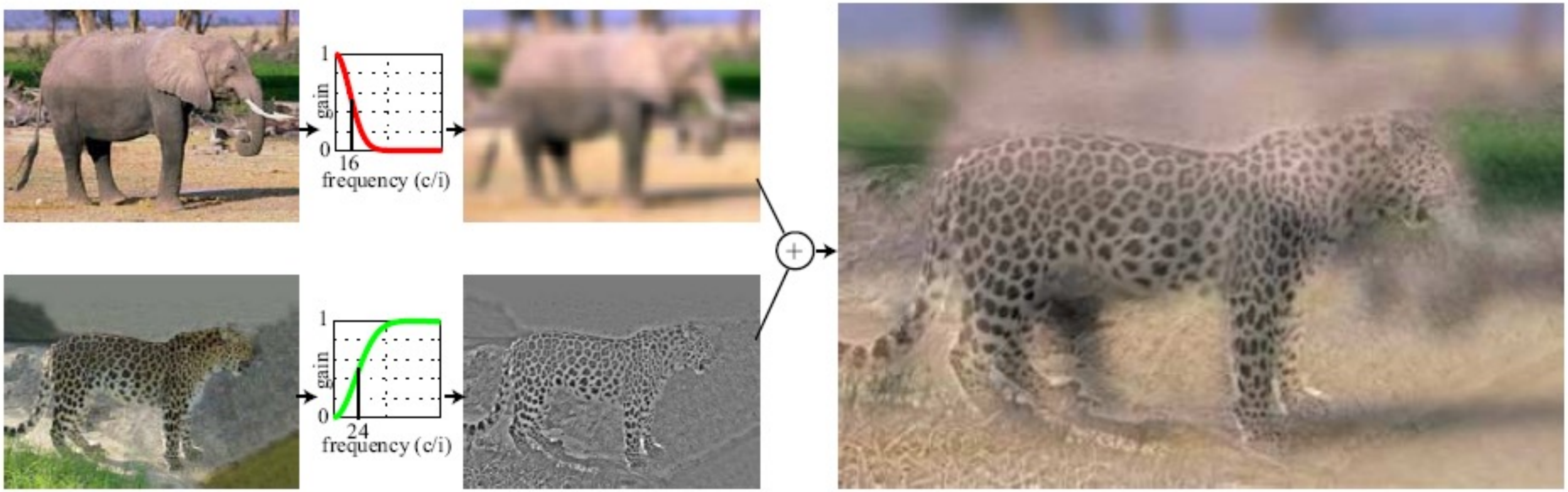
Perform convolution along rows:

$$\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 2 & 3 & 3 \\ 3 & 5 & 5 \\ 4 & 4 & 6 \end{bmatrix} = \begin{bmatrix} & 11 & \\ & 18 & \\ & 18 & \end{bmatrix}$$

Followed by convolution along the remaining column:

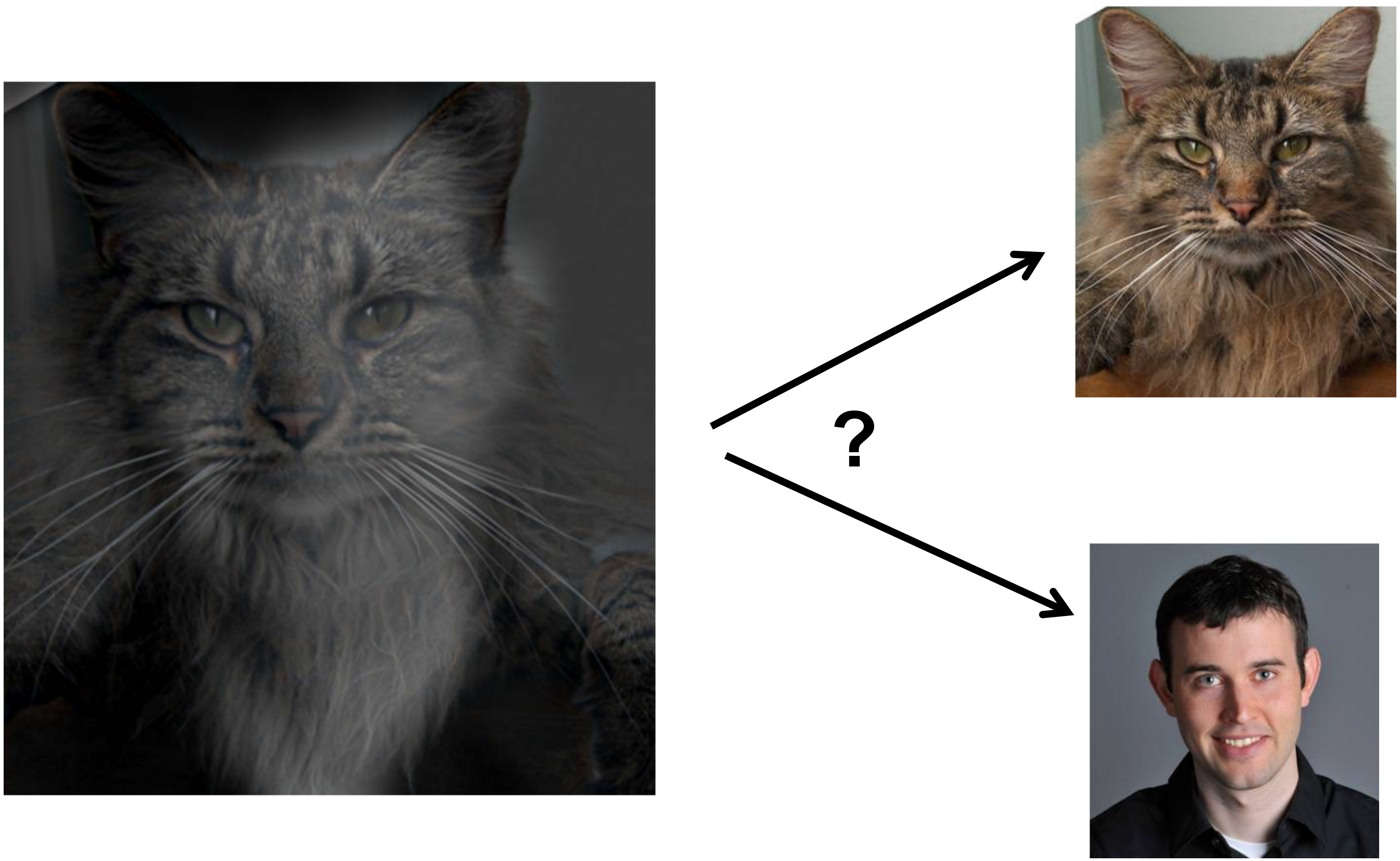
Source: K. Grauman

Hybrid Images

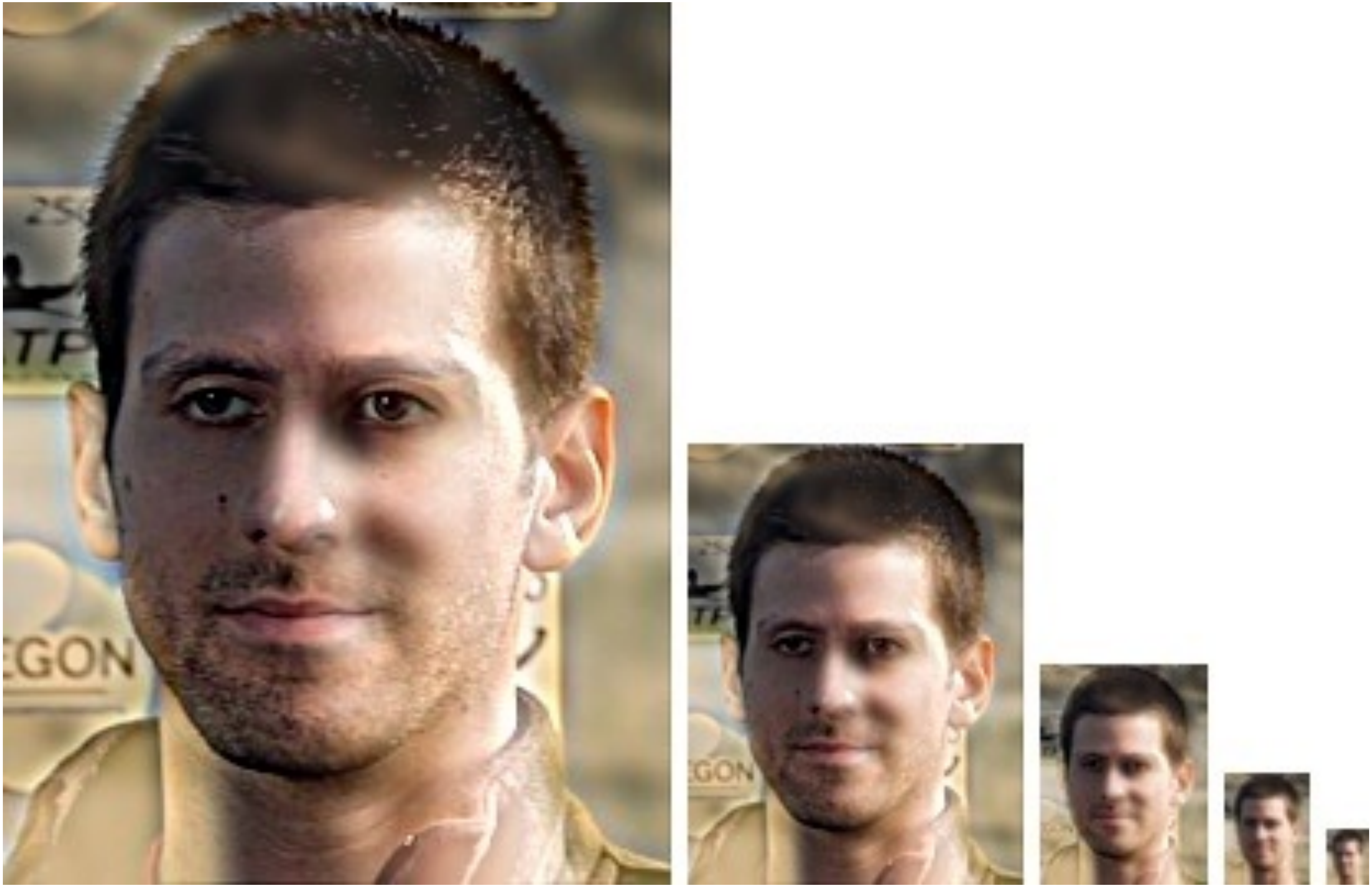


- A. Oliva, A. Torralba, P.G. Schyns, "Hybrid Images," SIGGRAPH 2006

Why do we get different, distance-dependent interpretations of hybrid images?



Slide: Hoiem

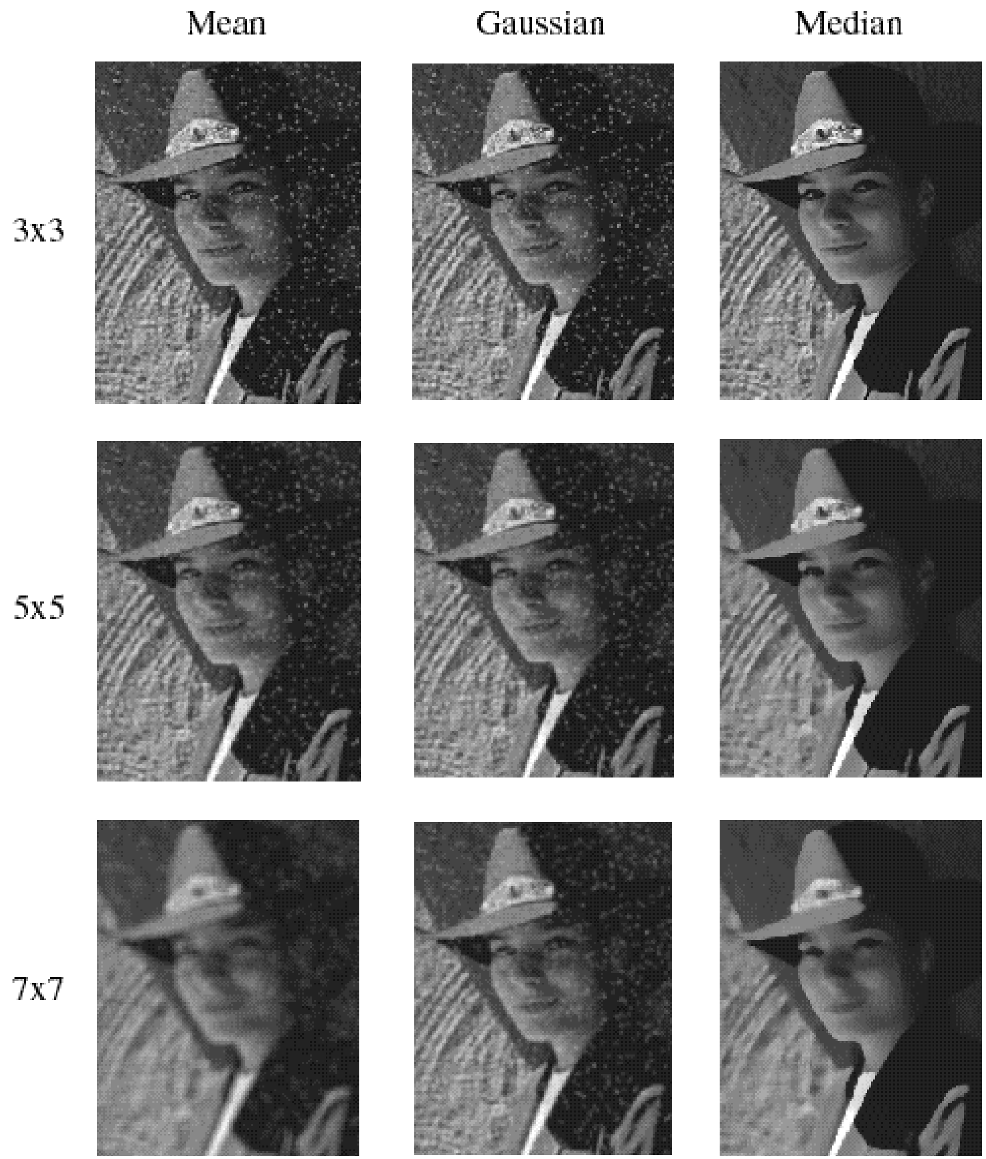


Median filters

- A **Median Filter** operates over a window by selecting the median intensity in the window.
- What advantage does a median filter have over a mean filter?
- Is a median filter a kind of convolution?

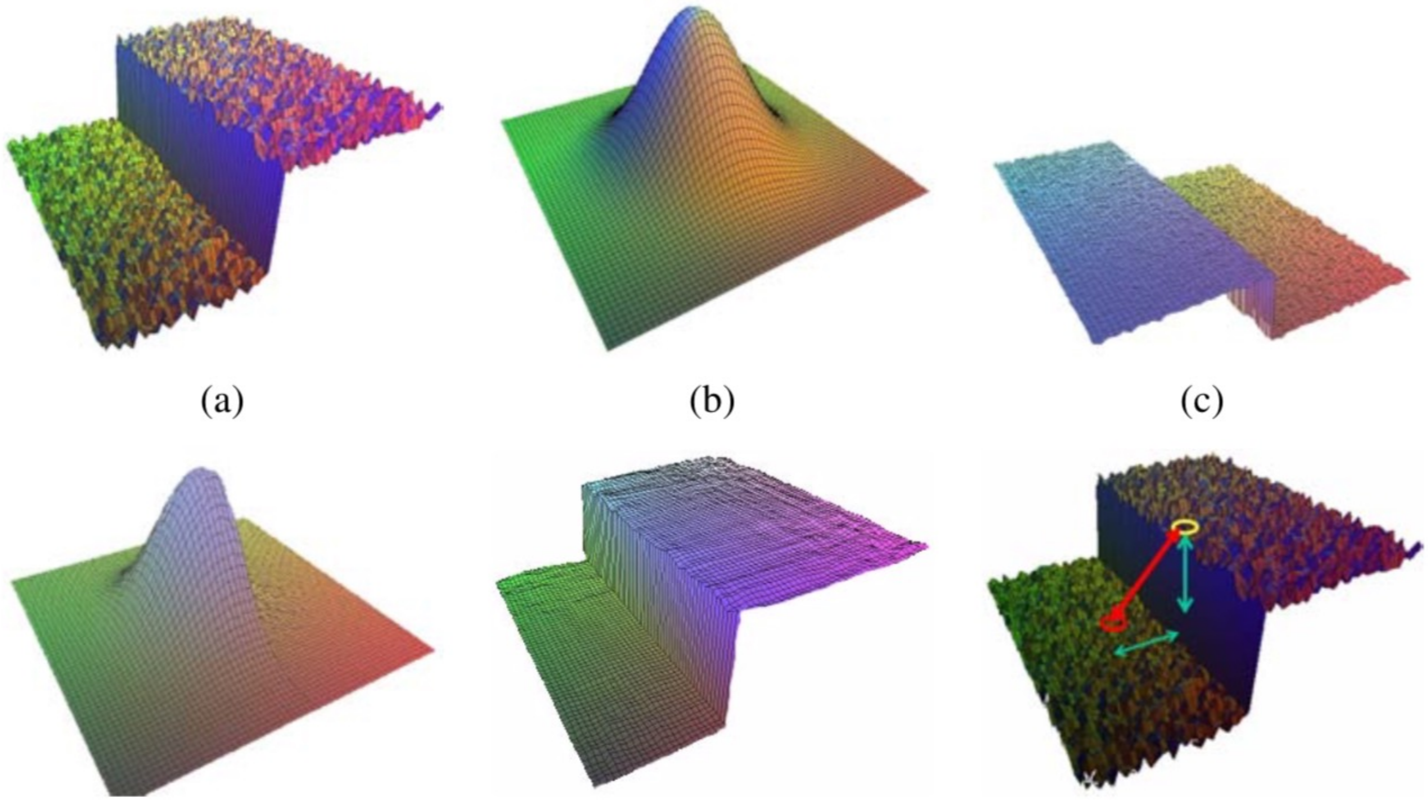
Slide by Steve Seitz

Comparison: salt and pepper noise



Slide by Steve Seitz

Bilateral filtering



(a) (b) (c)

(d) (e) (f)

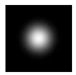
Figure 3.20 Bilateral filtering (Durand and Dorsey 2002) © 2002 ACM: (a) noisy step edge input; (b) domain filter (Gaussian); (c) range filter (similarity to center pixel value); (d) bilateral filter; (e) filtered step edge output; (f) 3D distance between pixels.

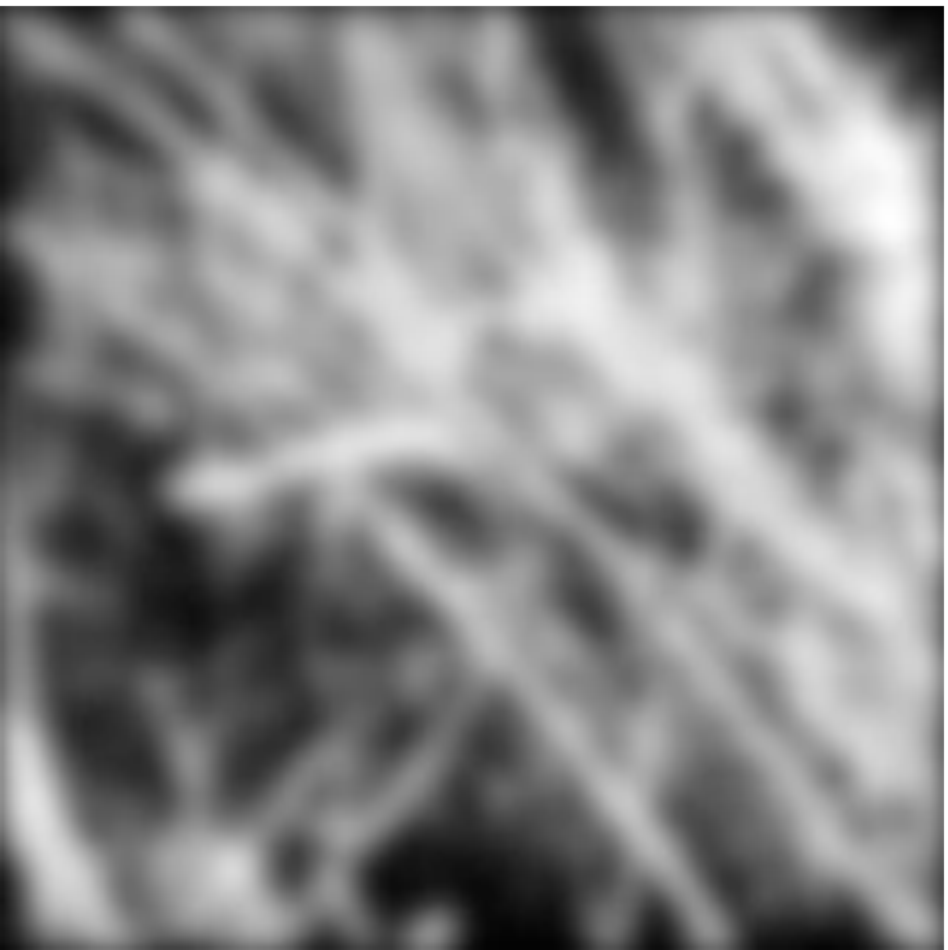
Morphological Operators

(a) (b) (c) (d) (e) (f)

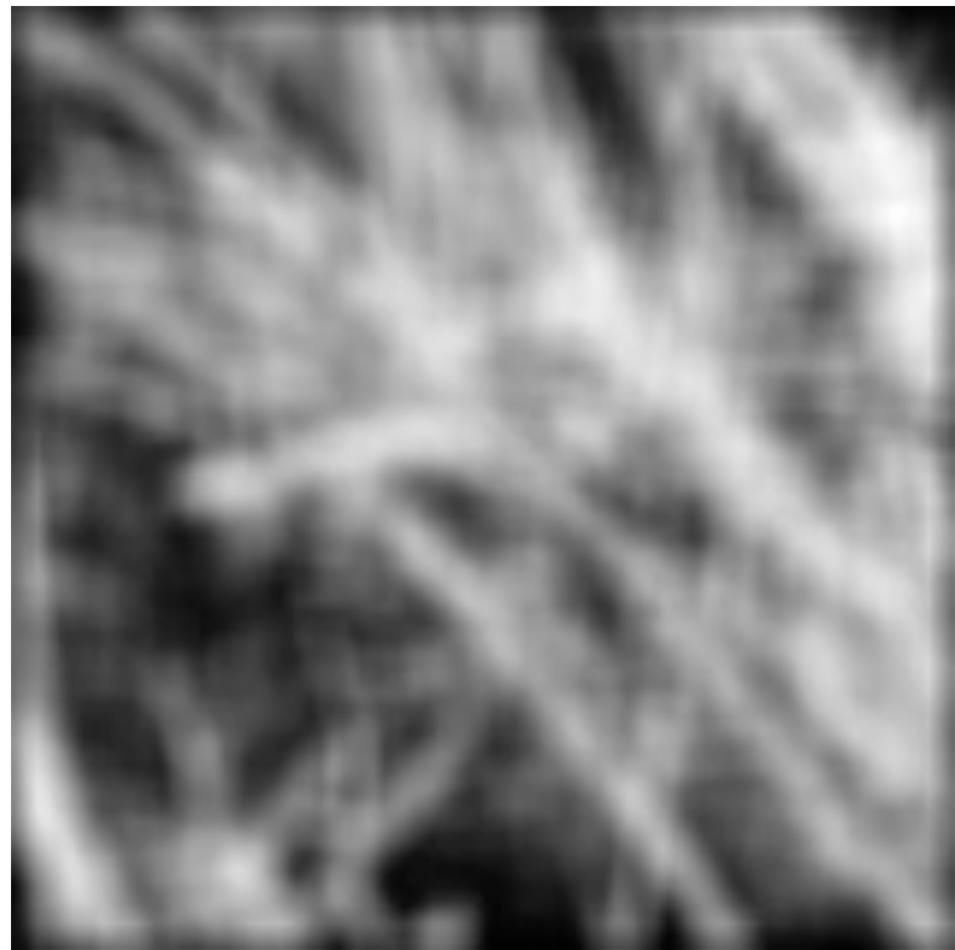
Figure 3.21 Binary image morphology: (a) original image; (b) dilation; (c) erosion; (d) majority; (e) opening; (f) closing. The structuring element for all examples is a 5×5 square. The effects of majority are a subtle rounding of sharp corners. Opening fails to eliminate the dot, since it is not wide enough.

Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?

Gaussian 



Box filter 



Why does a lower resolution image still make sense to us? What do we lose?

Image: http://www.flickr.com/photos/igorms/136916757/

Thinking in Frequency

Fourier, Joseph (1768-1830)

French mathematician who discovered that any periodic motion can be written as a superposition of sinusoidal and cosinusoidal vibrations. He developed a mathematical theory of heat in *Théorie Analytique de la Chaleur (Analytic Theory of Heat)*, (1822), discussing it in terms of differential equations.

Fourier was a friend and advisor of Napoleon. Fourier believed that his health would be improved by wrapping himself up in blankets, and in this state he tripped down the stairs in his house and killed himself. The paper of Galois which he had taken home to read shortly before his death was never recovered.

SEE ALSO: Galois

Additional biographies: MacTutor (St. Andrews), Bonn

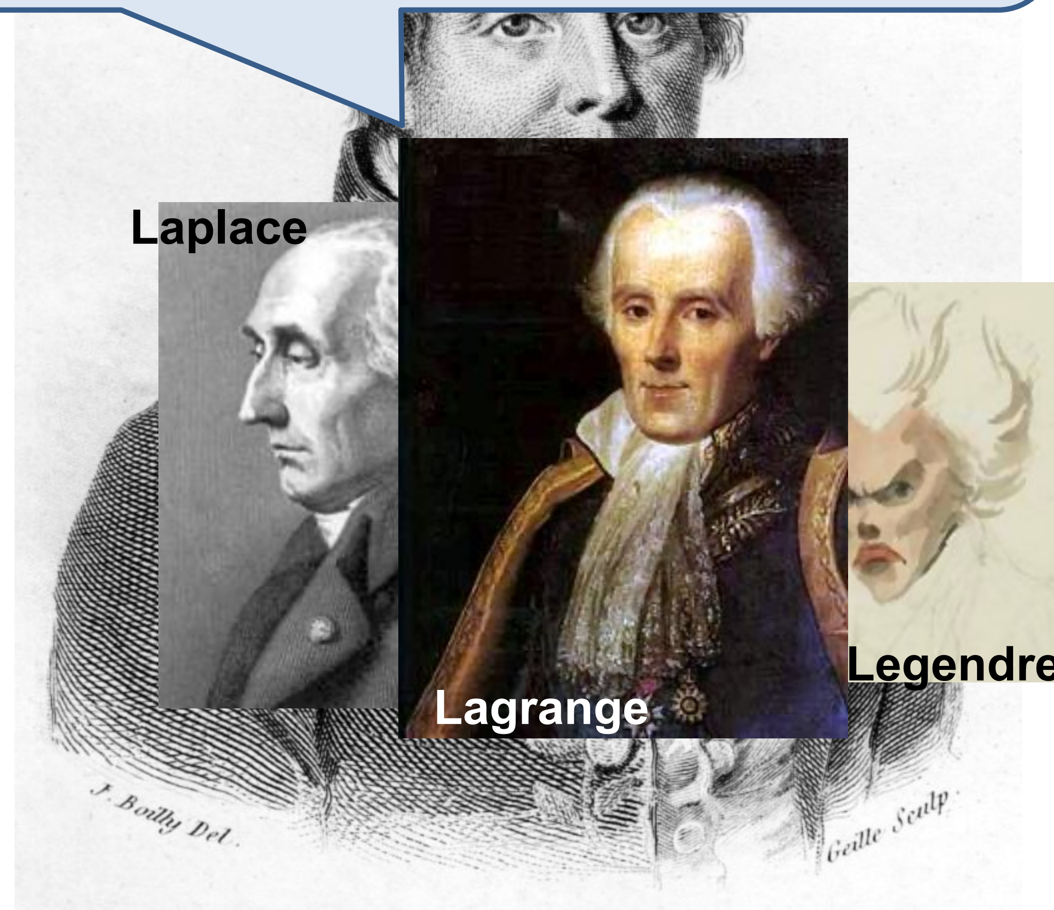
Jean Baptiste Joseph Fourier (1768-1830)

had crazy idea (1807):

...the manner in which the author arrives at these equations is not exempt of difficulties and...his analysis to integrate them still leaves something to be desired on the score of generality and even rigour.

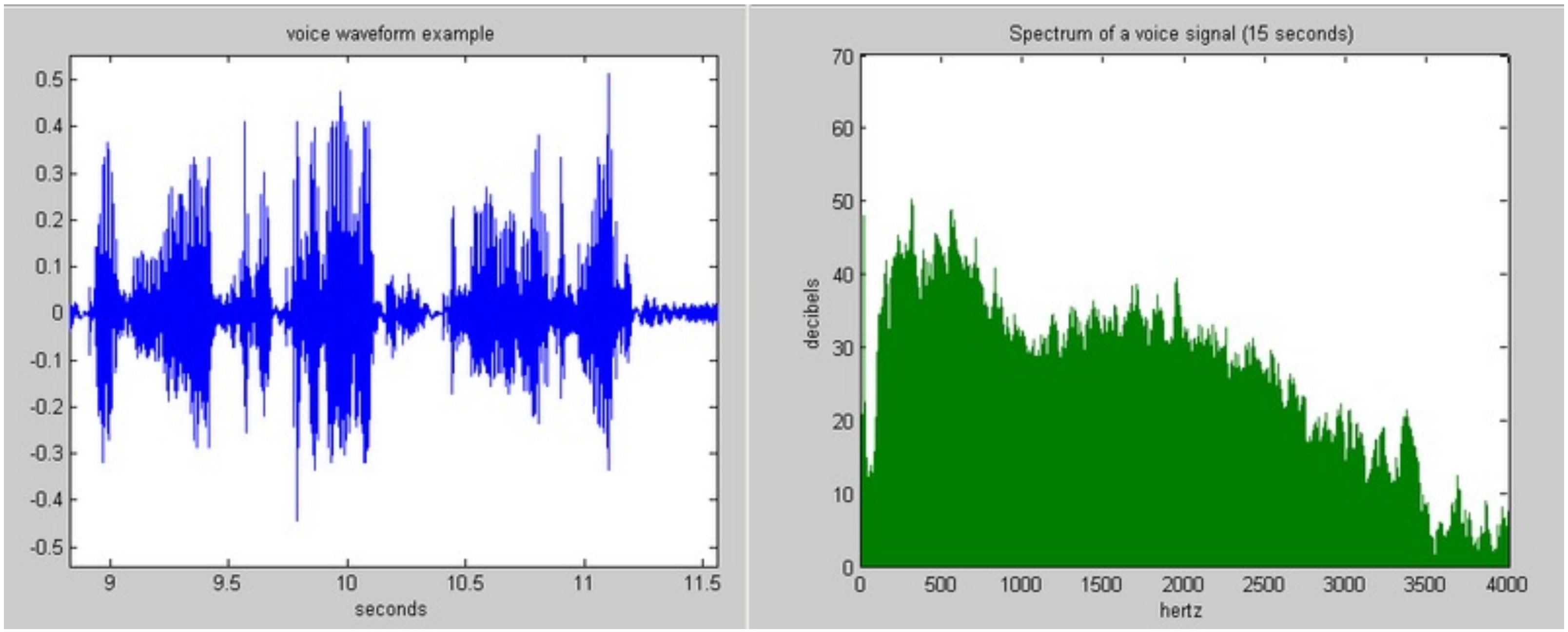
> ***Any** univariate function can*
> *rewritten as a weighted sum*
> *sines and cosines of different*
> *frequencies.*

- Don't believe it?
  - Neither did Lagrange, Laplace, Poisson and other big wigs
  - Not translated into English until 1878!
- But it's (mostly) true!
  - called Fourier Series
  - there are some subtle restrictions



Example: Music

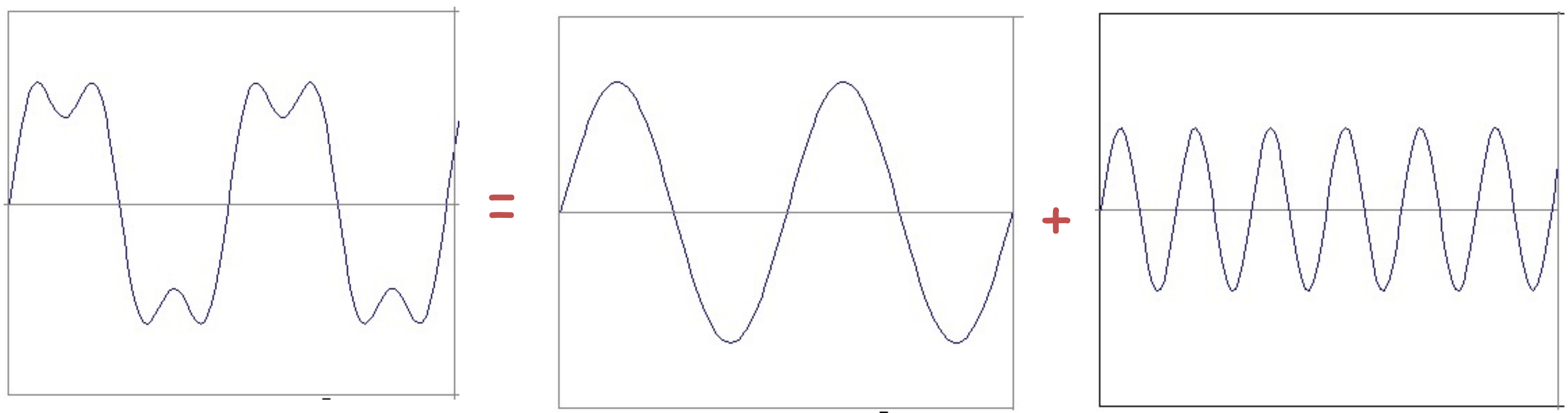
- We think of music in terms of frequencies at different magnitudes

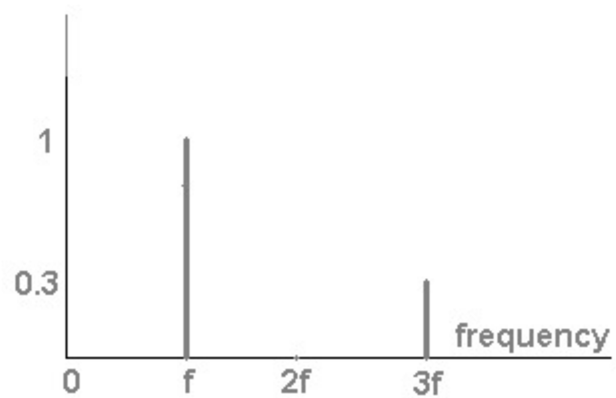


Slide: Hoiem

Frequency Spectra

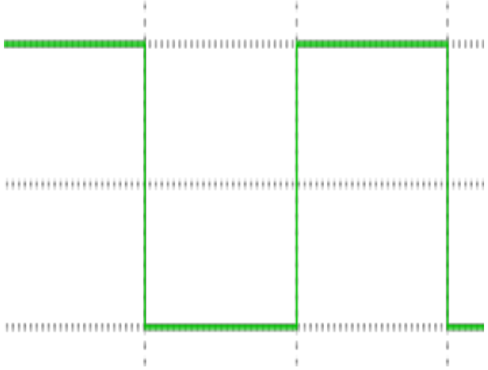
- example : $g(t) = \sin(2\pi f\, t) + (1/3)\sin(2\pi(3f)\, t)$



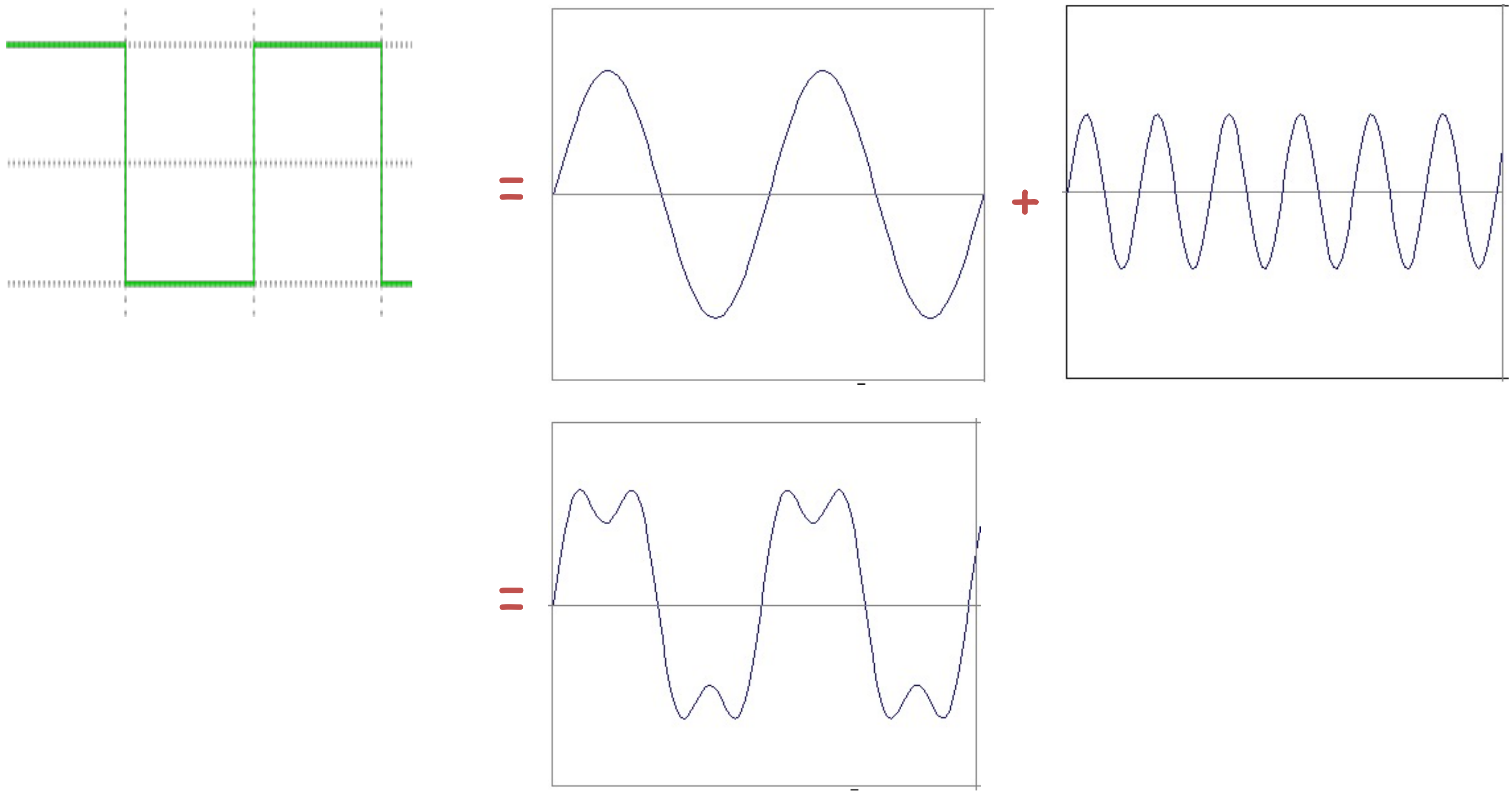


Slides: Efros

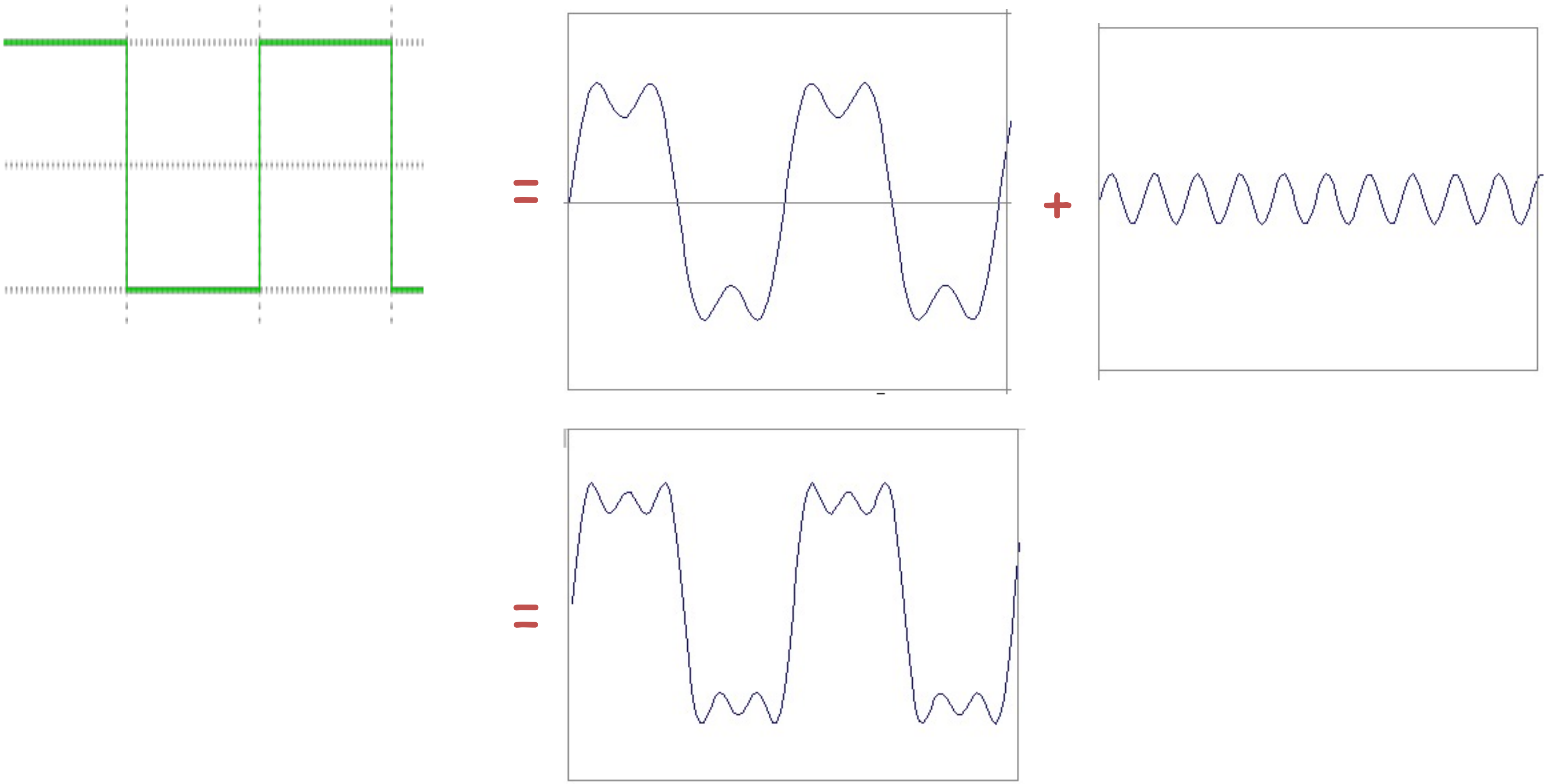
Frequency Spectra



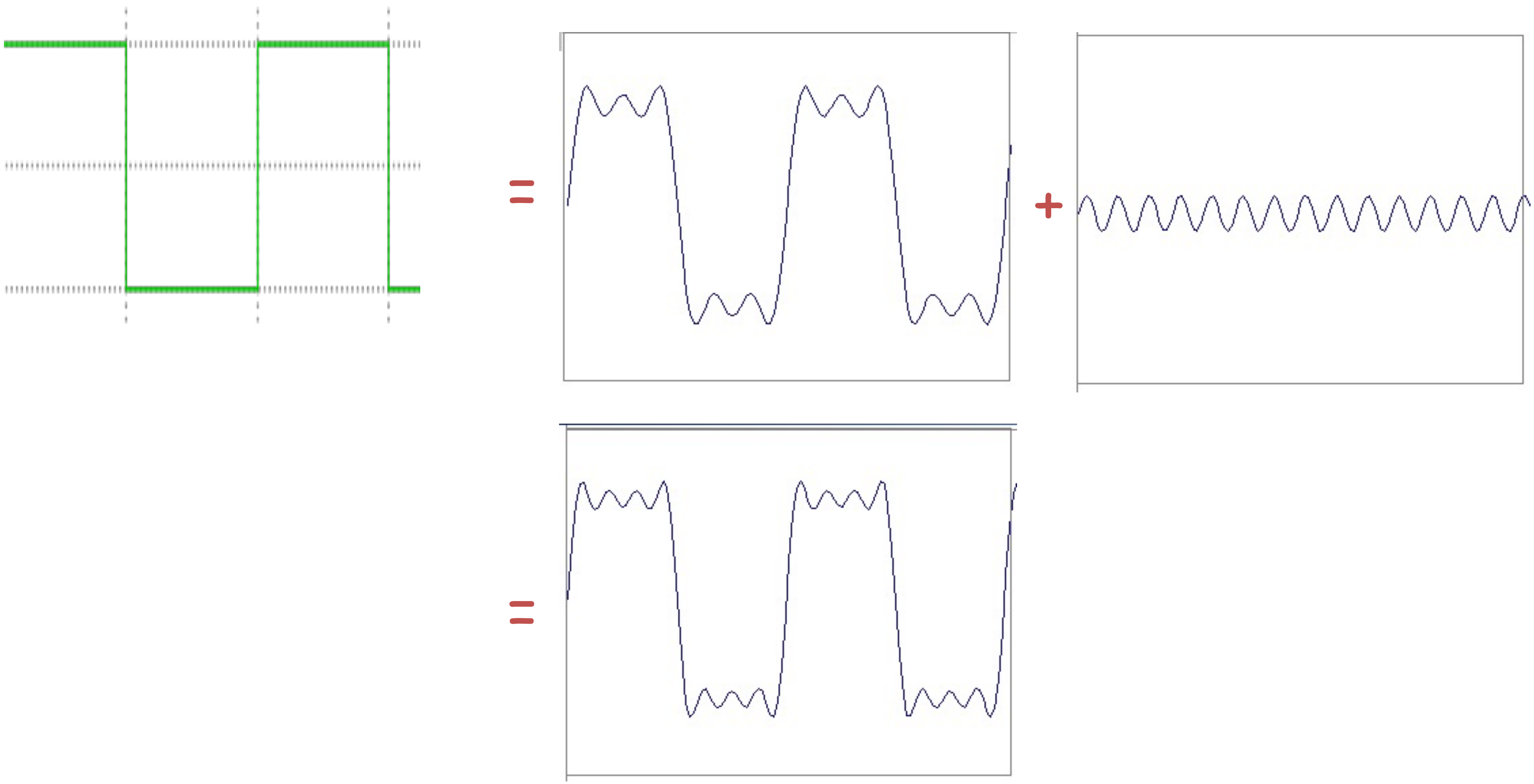
Frequency Spectra



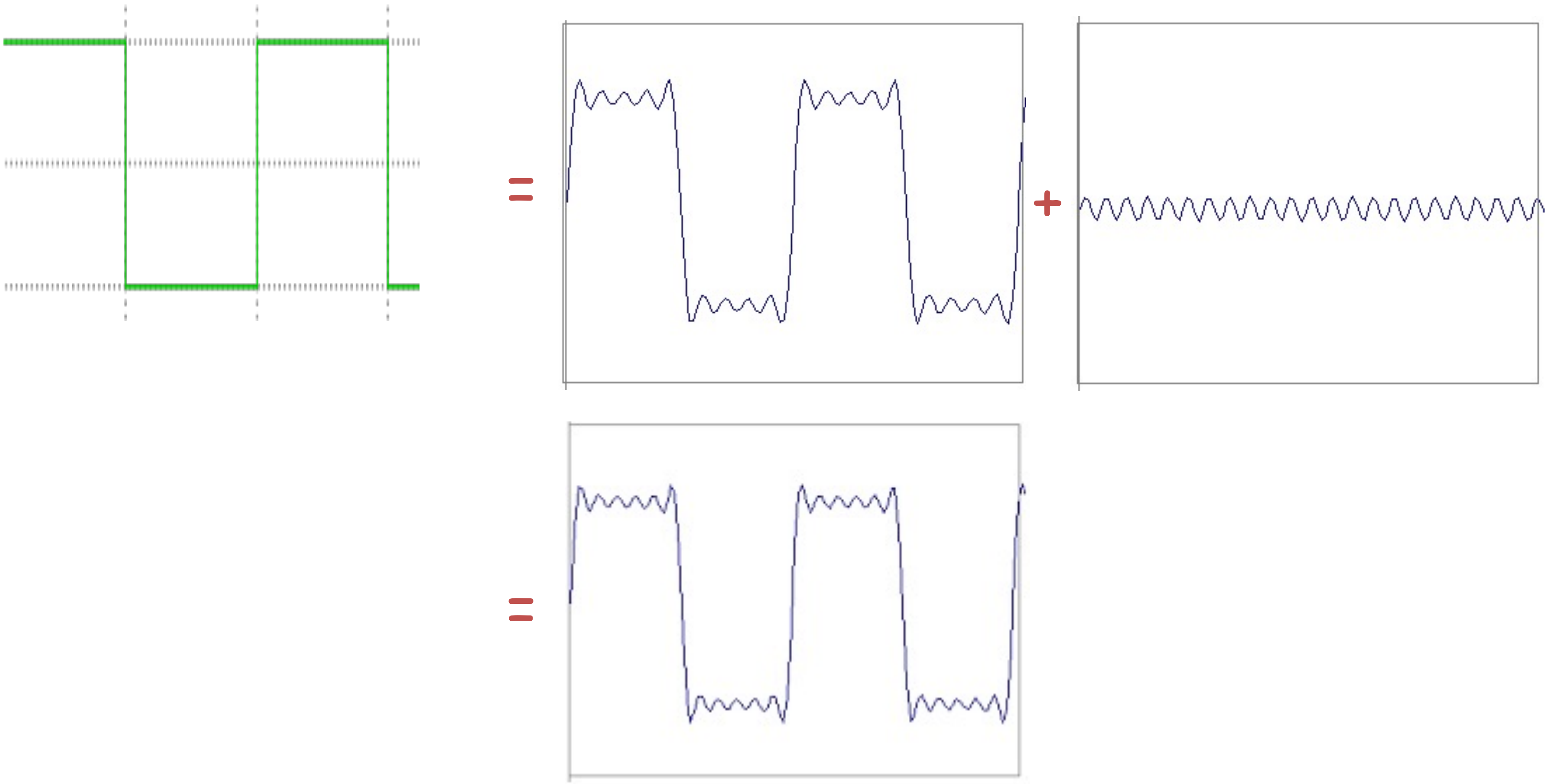
Frequency Spectra



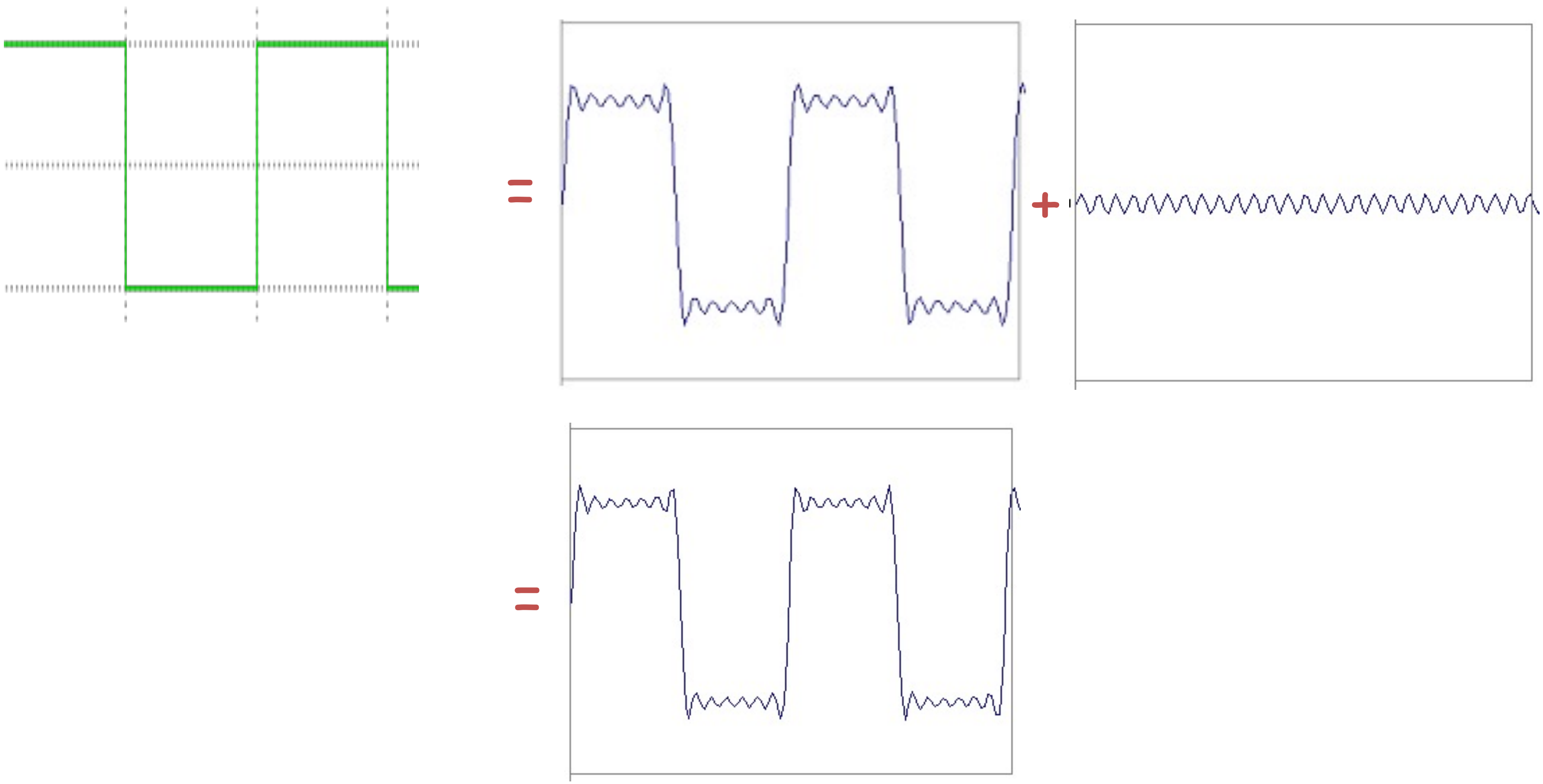
Frequency Spectra



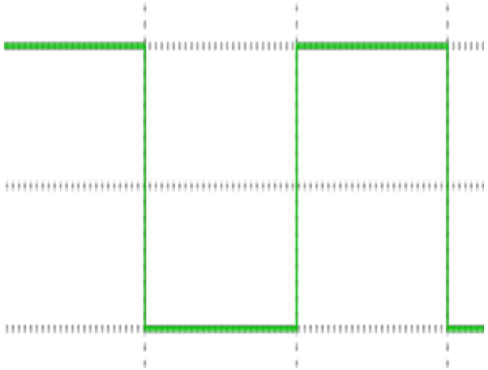
Frequency Spectra



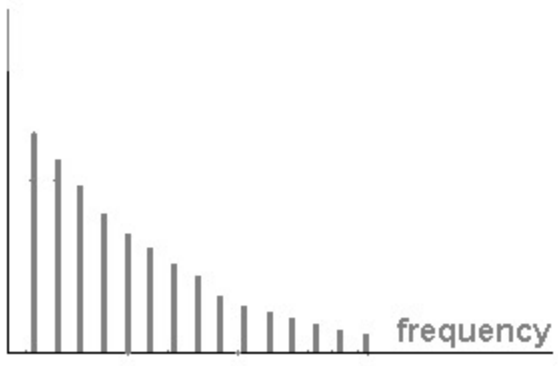
Frequency Spectra



Frequency Spectra



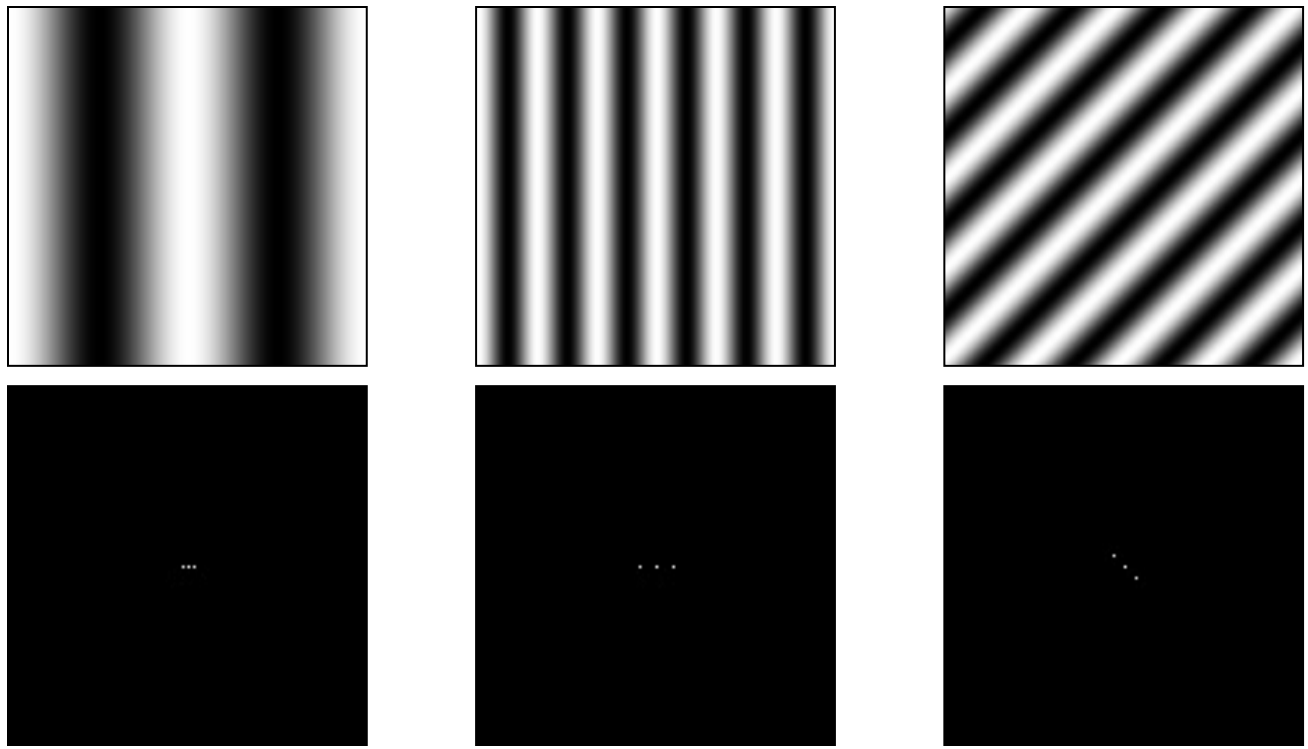
$$= \boxed{A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi kt)}$$



Fourier analysis in images

Intensity Image

Fourier Image



http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering

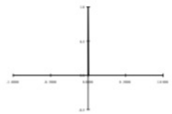
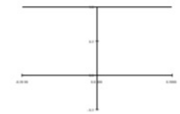
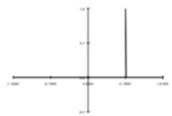
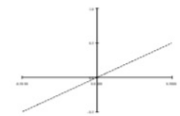
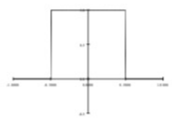
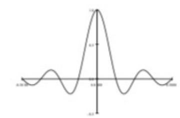
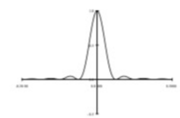
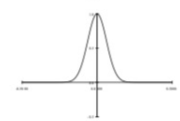
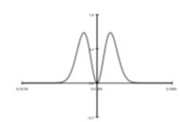
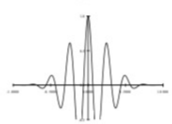
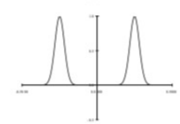
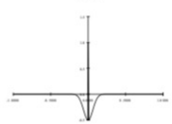
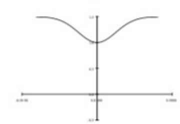
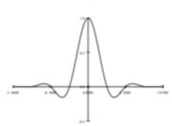
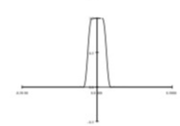
Fourier Transform

- Fourier transform stores the **magnitude** and **phase** at each frequency
  - **Magnitude** encodes how much signal there is at a particular frequency
  - **Phase** encodes spatial information (indirectly)
  - For mathematical convenience, this is often notated in terms of real and complex numbers

Amplitude: $$A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$$

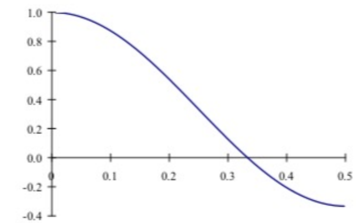
Phase: $$\phi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$$

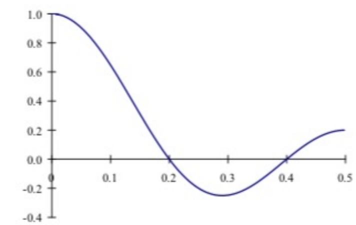
Fourier Transform Pairs

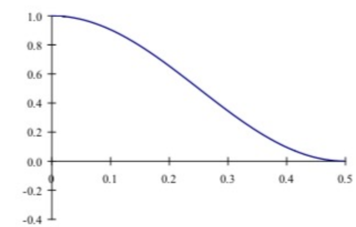
Name		Signal		Transform	
impulse		$\delta(x)$	$\Leftrightarrow$	1	
shifted impulse		$\delta(x-u)$	$\Leftrightarrow$	$e^{-j\omega u}$	
box filter		$\mathrm{box}(x/a)$	$\Leftrightarrow$	$a\mathrm{sinc}(a\omega)$	
tent		$\mathrm{tent}(x/a)$	$\Leftrightarrow$	$a\mathrm{sinc}^2(a\omega)$	
Gaussian		$G(x;\sigma)$	$\Leftrightarrow$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega;\sigma^{-1})$	
Laplacian of Gaussian		$(\frac{x^2}{\sigma^4}-\frac{1}{\sigma^2})G(x;\sigma)$	$\Leftrightarrow$	$-\frac{\sqrt{2\pi}}{\sigma}\omega^2 G(\omega;\sigma^{-1})$	
Gabor		$\cos(\omega_0 x)G(x;\sigma)$	$\Leftrightarrow$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega\pm\omega_0;\sigma^{-1})$	
unsharp mask		$(1+\gamma)\delta(x)$ $-\gamma G(x;\sigma)$	$\Leftrightarrow$	$(1+\gamma)-$ $\frac{\sqrt{2\pi}\gamma}{\sigma}G(\omega;\sigma^{-1})$	
windowed sinc		$\mathrm{rcos}(x/(aW))$ $\mathrm{sinc}(x/a)$	$\Leftrightarrow$	(see Figure 3.29)	

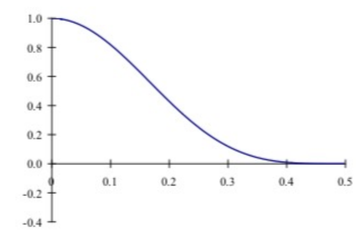
Fourier Transforms of Filters

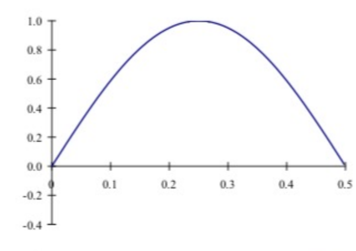
box-3	$\frac{1}{3}\begin{array}{\|c\|c\|c\|}\hline 1 & 1 & 1 \\ \hline\end{array}$	$\frac{1}{3}(1 + 2\cos\omega)$
box-5	$\frac{1}{5}\begin{array}{\|c\|c\|c\|c\|c\|}\hline 1 & 1 & 1 & 1 & 1 \\ \hline\end{array}$	$\frac{1}{5}(1 + 2\cos\omega + 2\cos 2\omega)$
linear	$\frac{1}{4}\begin{array}{\|c\|c\|c\|}\hline 1 & 2 & 1 \\ \hline\end{array}$	$\frac{1}{2}(1 + \cos\omega)$
binomial	$\frac{1}{16}\begin{array}{\|c\|c\|c\|c\|c\|}\hline 1 & 4 & 6 & 4 & 1 \\ \hline\end{array}$	$\frac{1}{4}(1 + \cos\omega)^2$
Sobel	$\frac{1}{2}\begin{array}{\|c\|c\|c\|}\hline -1 & 0 & 1 \\ \hline\end{array}$	$\sin\omega$



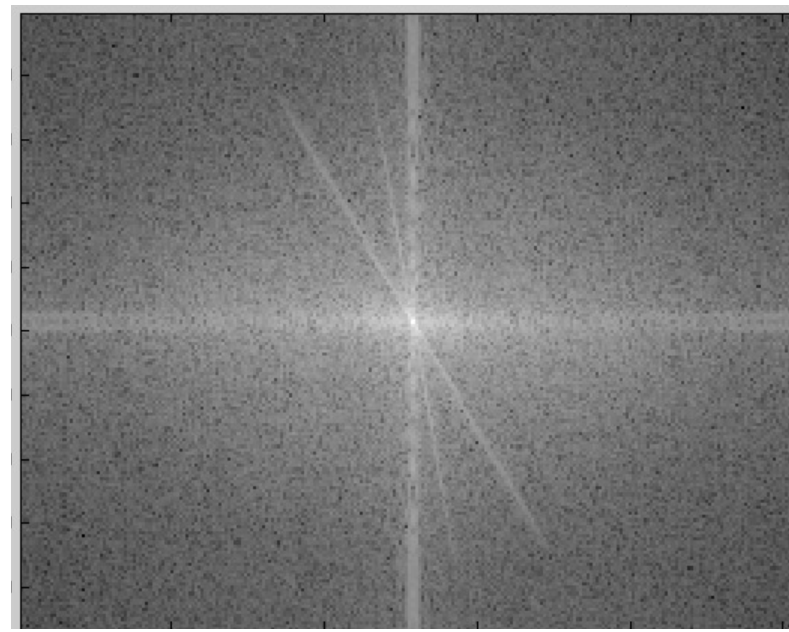






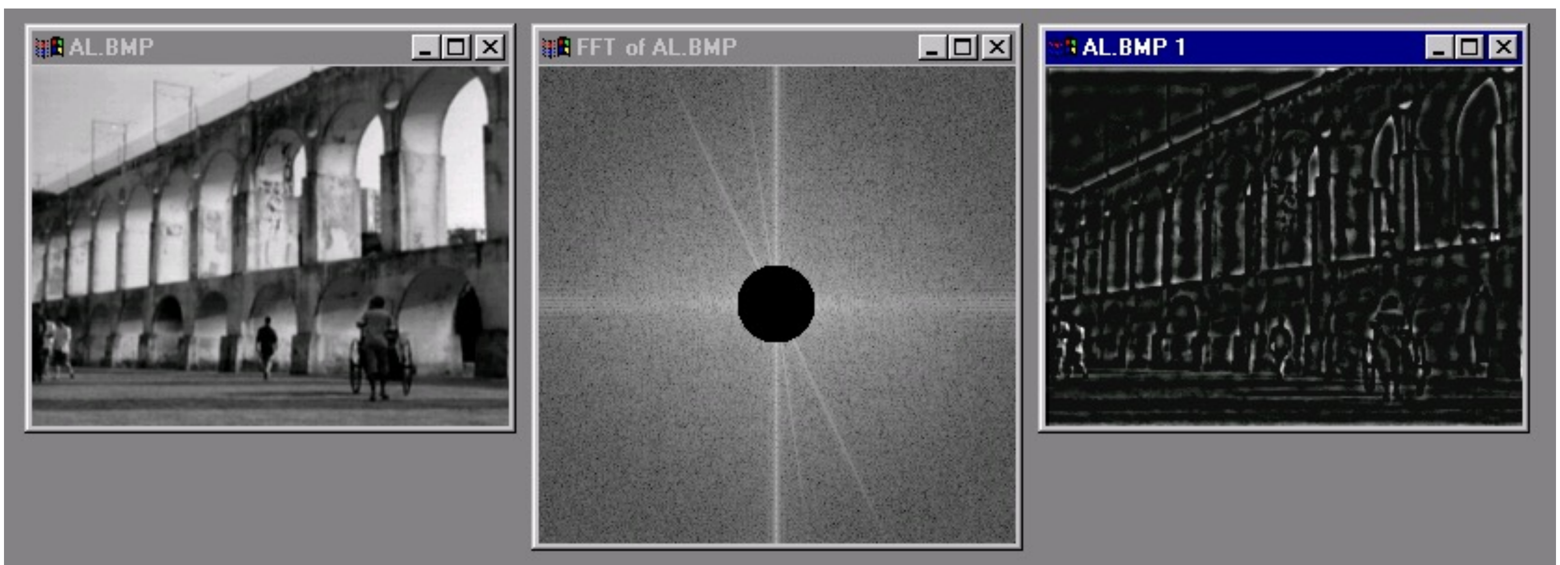


Man-made Scene



Can change spectrum, then reconstruct

Low and High Pass filtering



The Convolution Theorem

- The Fourier transform of the convolution of two functions is the product of their Fourier transforms

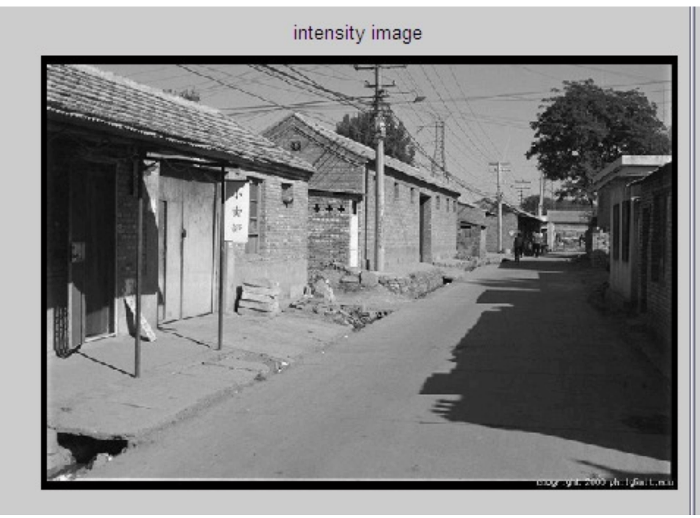
$$\mathrm{F}[g * h] = \mathrm{F}[g]\,\mathrm{F}[h]$$

- **Convolution** in spatial domain is equivalent to **multiplication** in frequency domain!

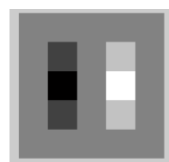
$$g * h = \mathrm{F}^{-1}[\mathrm{F}[g]\,\mathrm{F}[h]]$$

Filtering in spatial domain

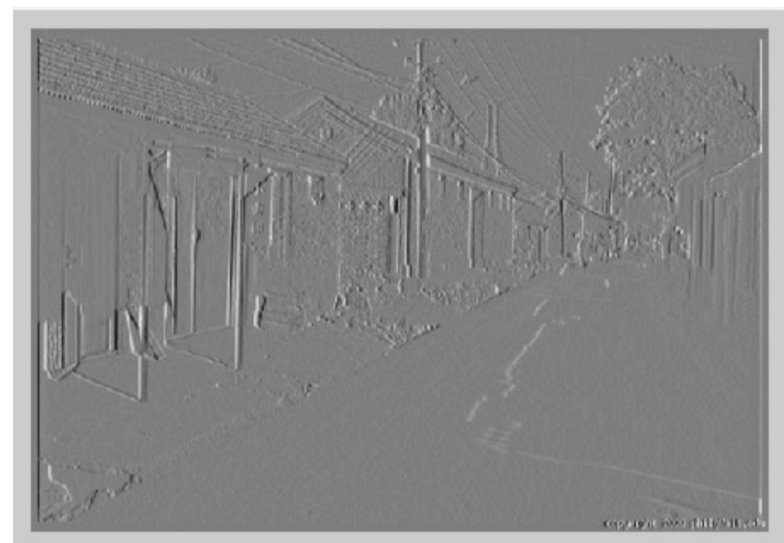
1	0	-1
2	0	-2
1	0	-1



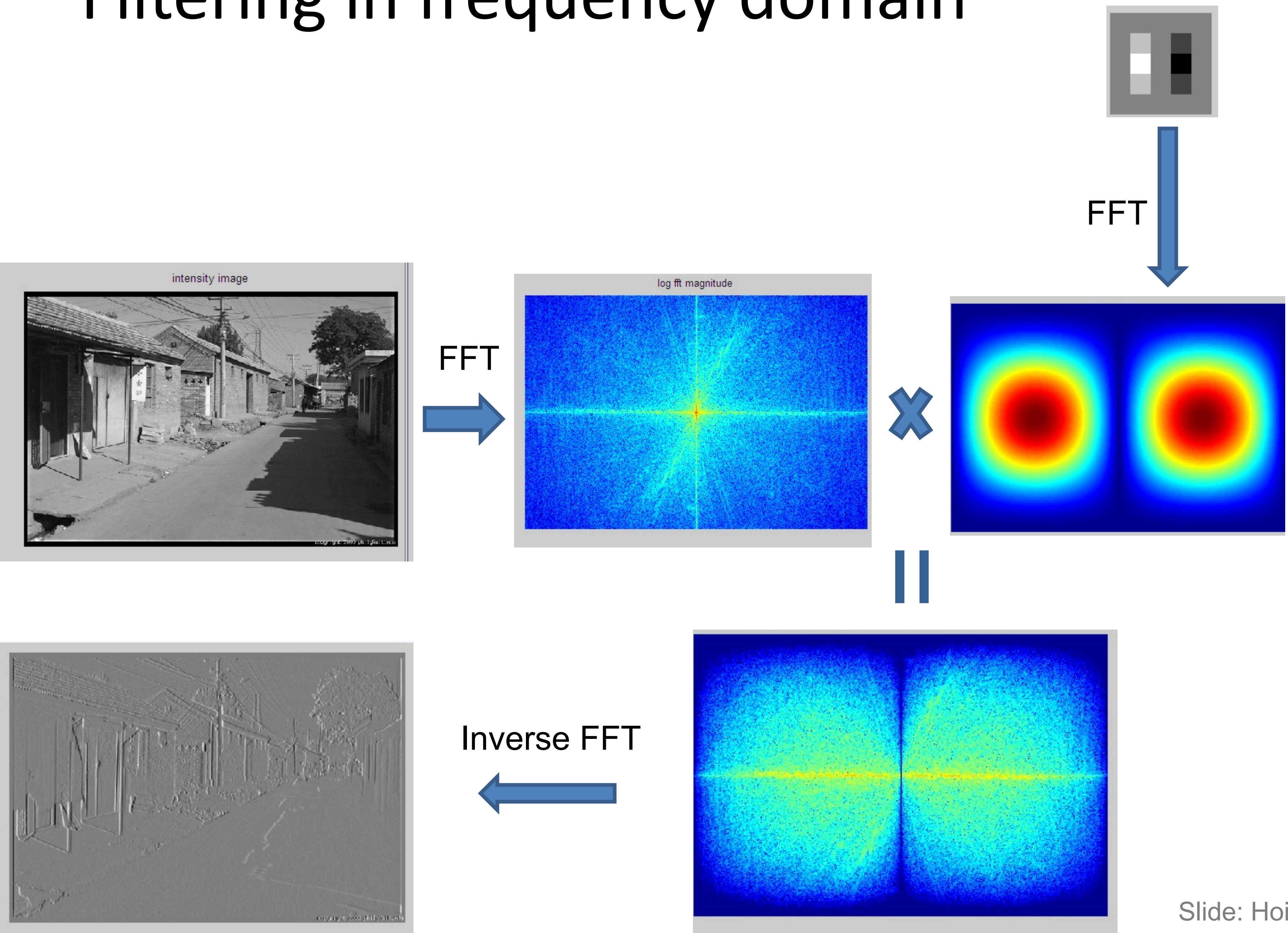
*



=



Filtering in frequency domain

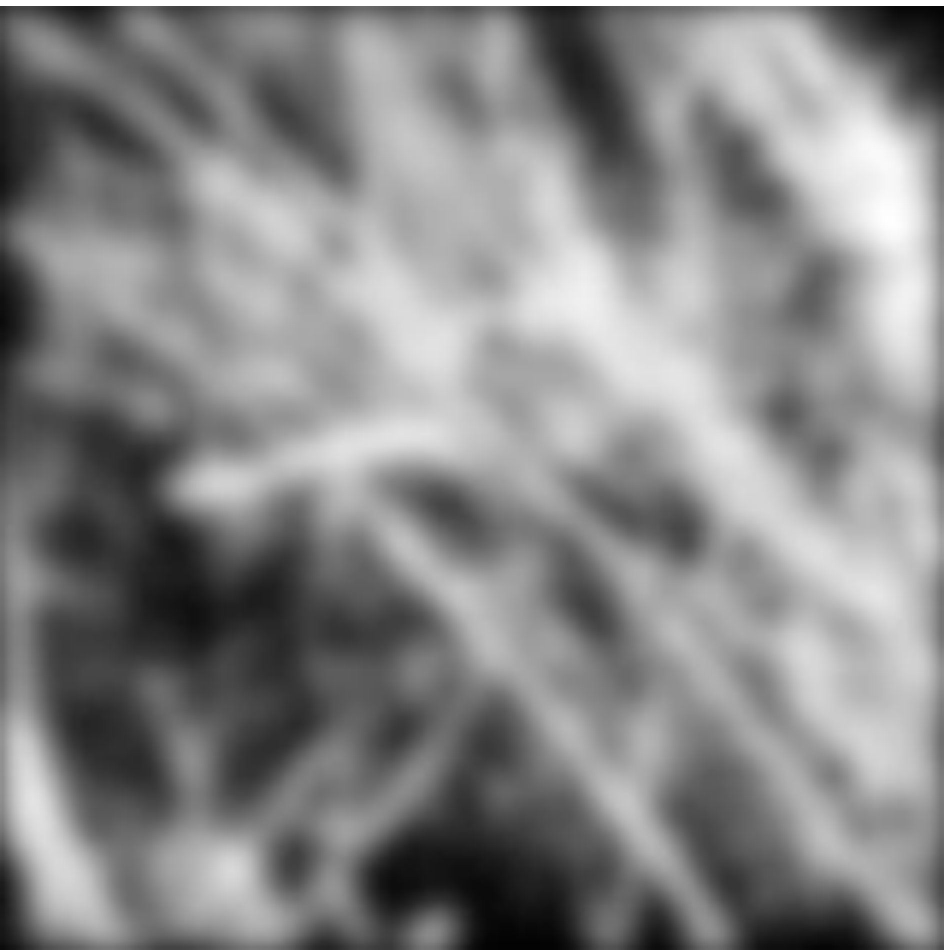


Slide: Hoiem

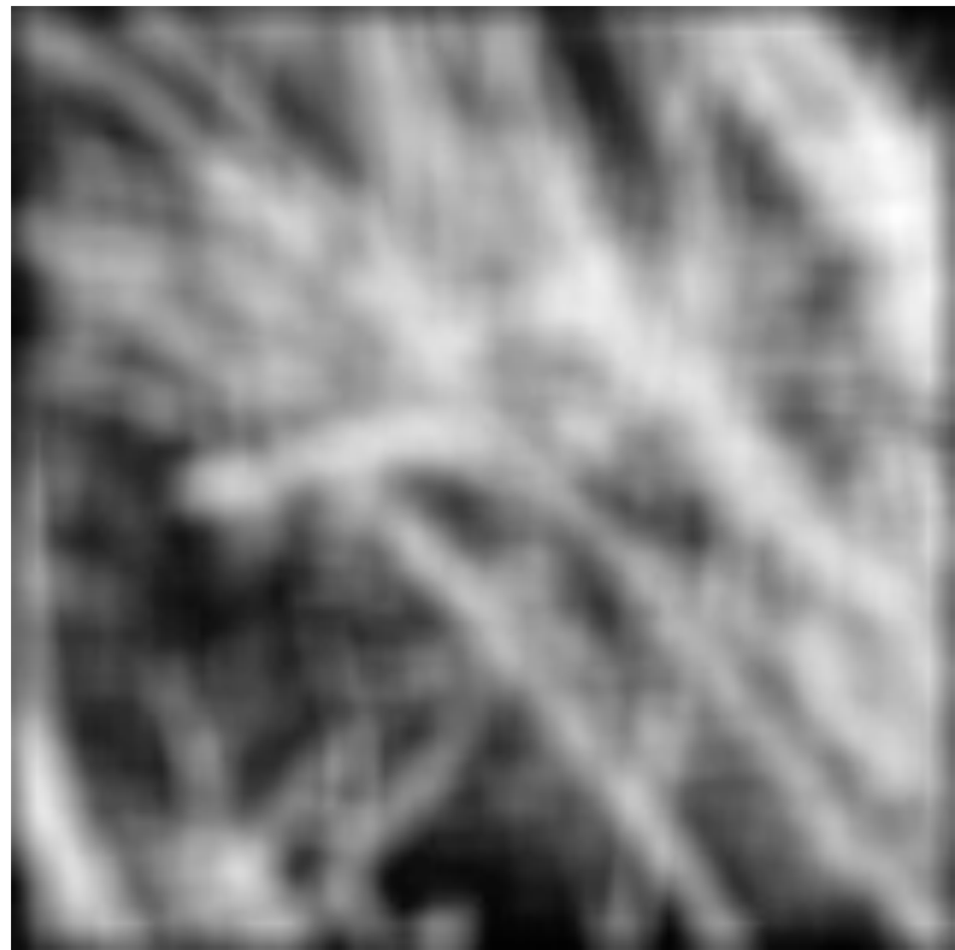
Filtering

Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?

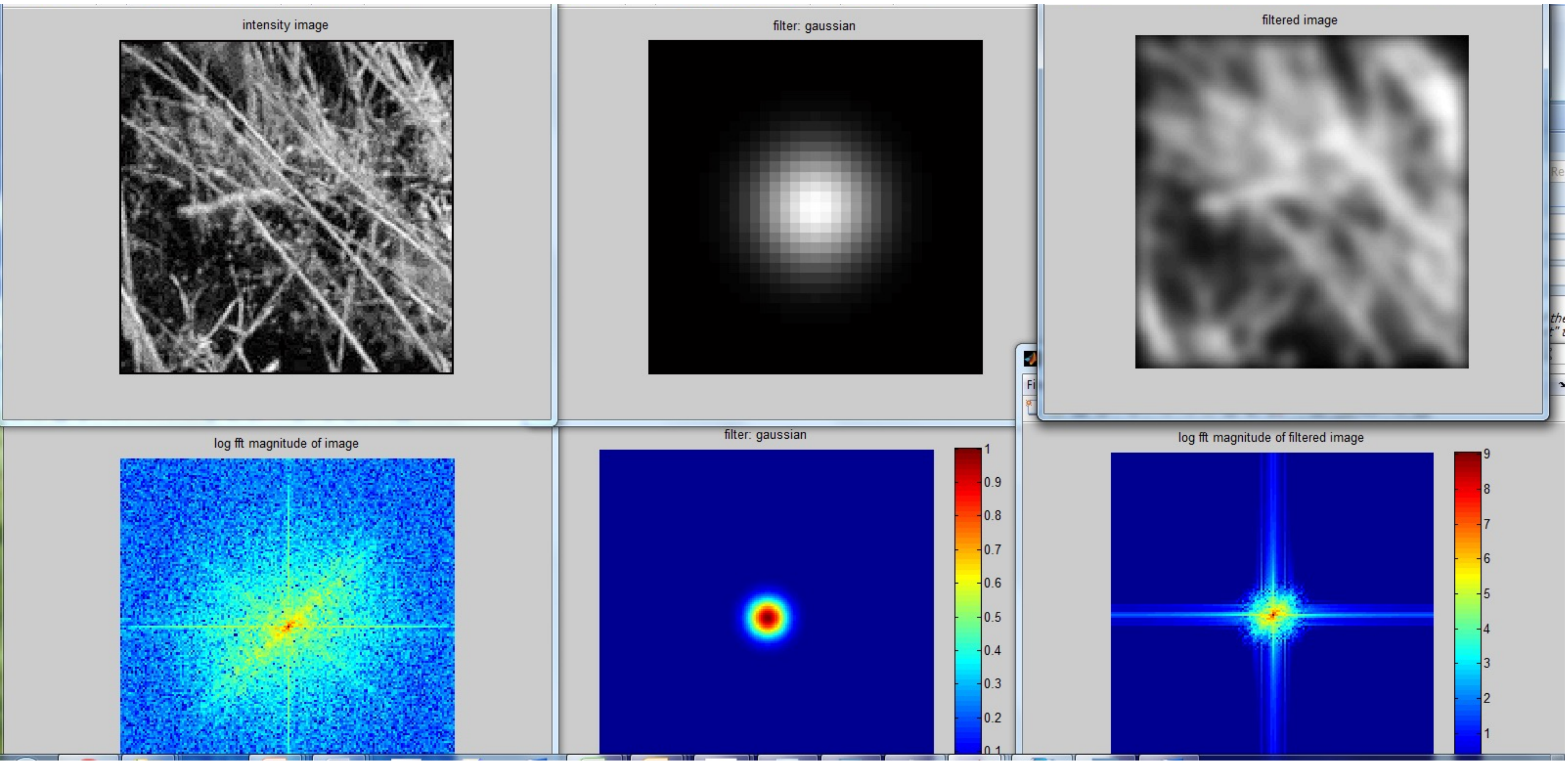
Gaussian 



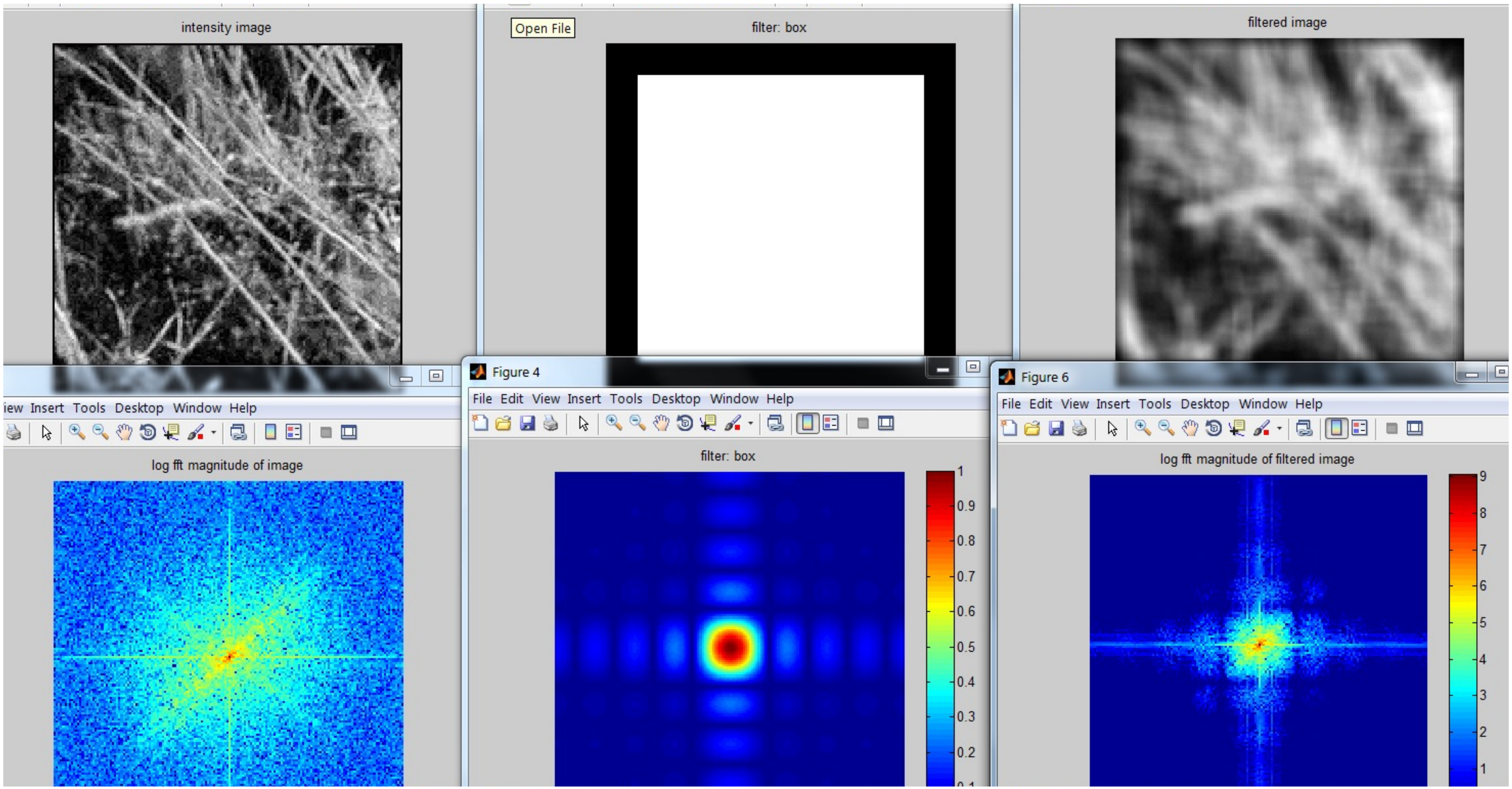
Box filter 



Gaussian


intensity image
filter: gaussian
filtered image
log fft magnitude of image
filter: gaussian
log fft magnitude of filtered image

Box Filter



intensity image
Open File
filter: box
filtered image
Figure 4
Figure 6
File Edit View Insert Tools Desktop Window Help
log fft magnitude of image
filter: box
log fft magnitude of filtered image
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
9
8
7
6
5
4
3
2
1