CG – T15 – Spatial Filters

L:CC, MI:ERSI

Miguel Tavares Coimbra

U. PORTO FC

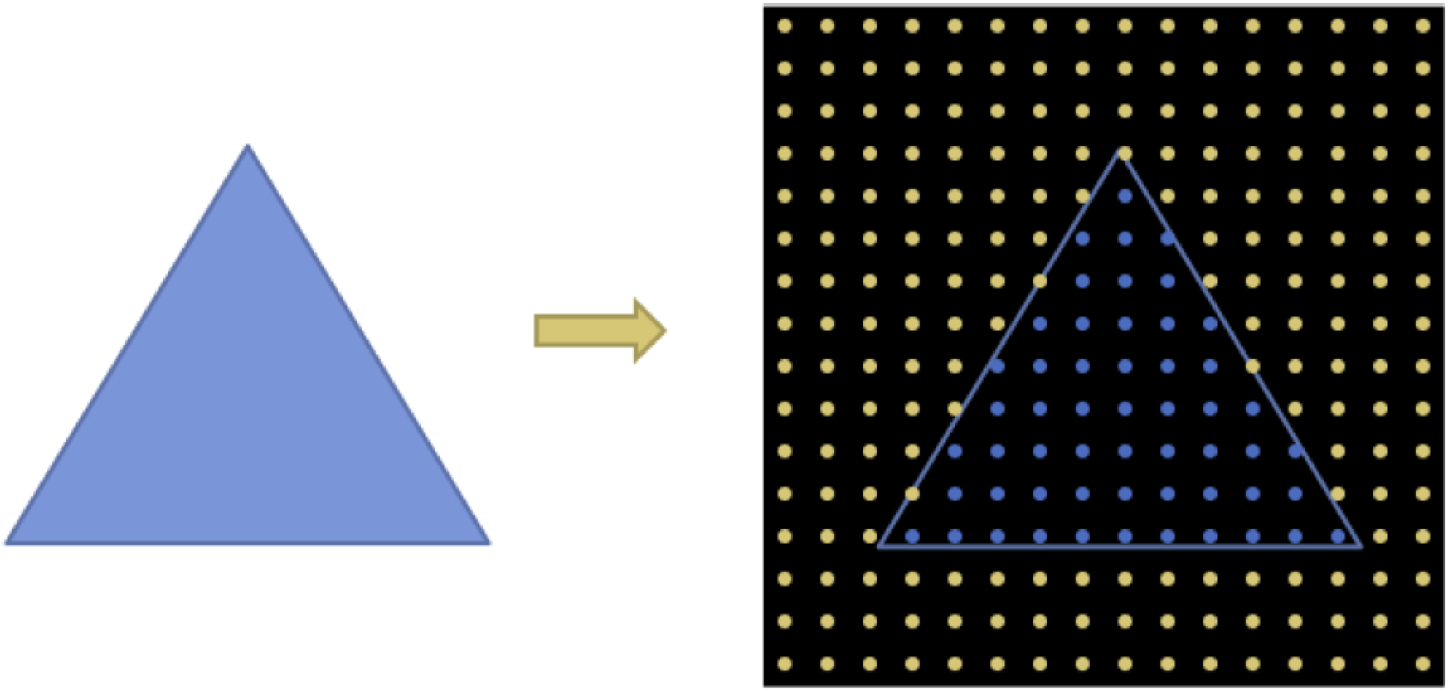
Suggested reading

- Gonzalez & Woods, “Digital Image Processing”, 3rd Edition, Prentice Hall
  - Chapter 3 – Intensity Transformations and Spatial Filtering

Various 2D images in the 3D pipeline

Fragments are 2D images

Rasterization



Textures are 2D images

Frame buffers are 2D images

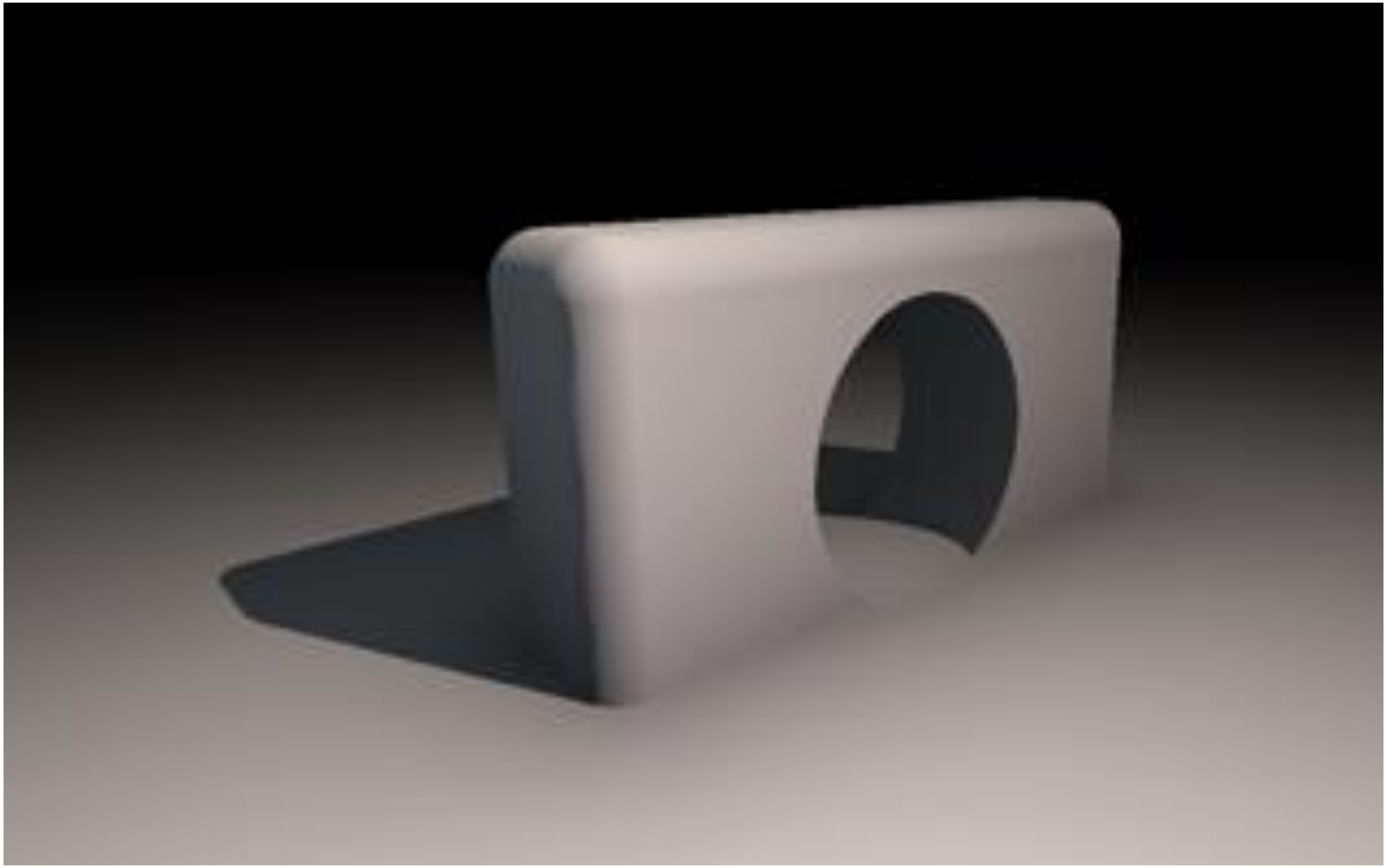


Image from http://www.iquilezles.org

What can I do with processing?

Blurring

Good for anti-aliasing!

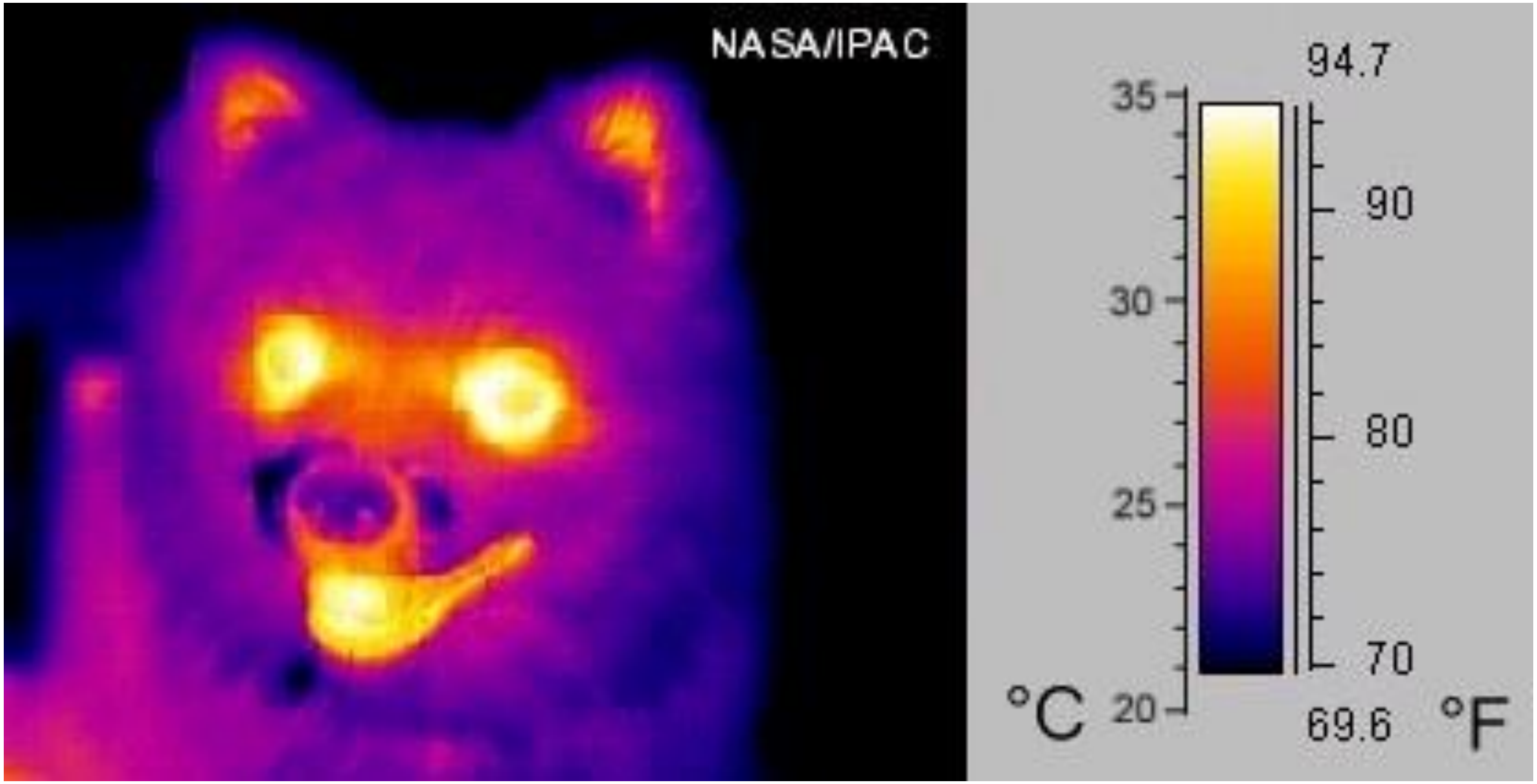
Edge Detection

And combine it with the image for unsharp filters

Depth of field effects

Image from http://images.dpchallenge.com/

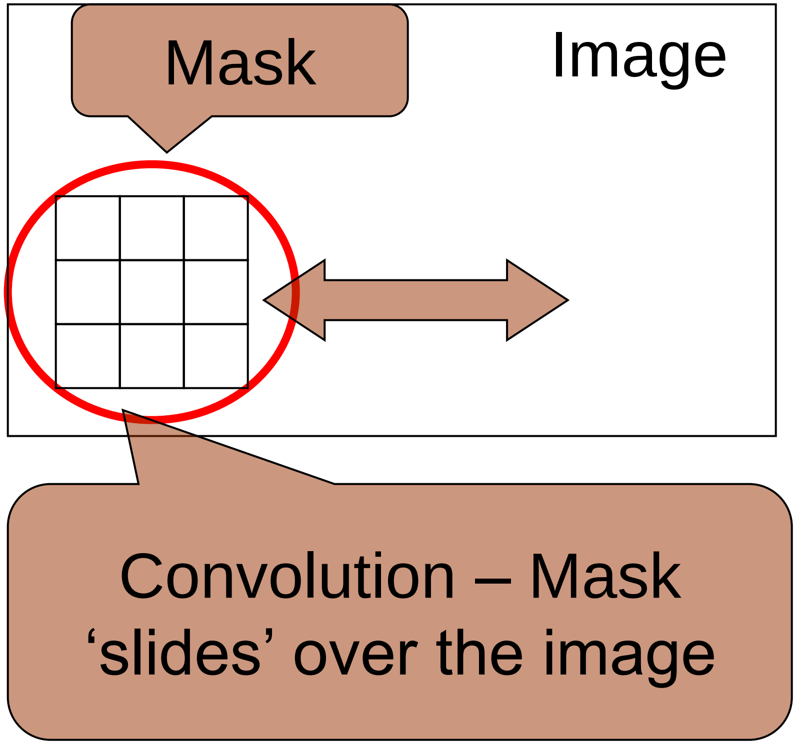
Pseudocolor



How can I do this?

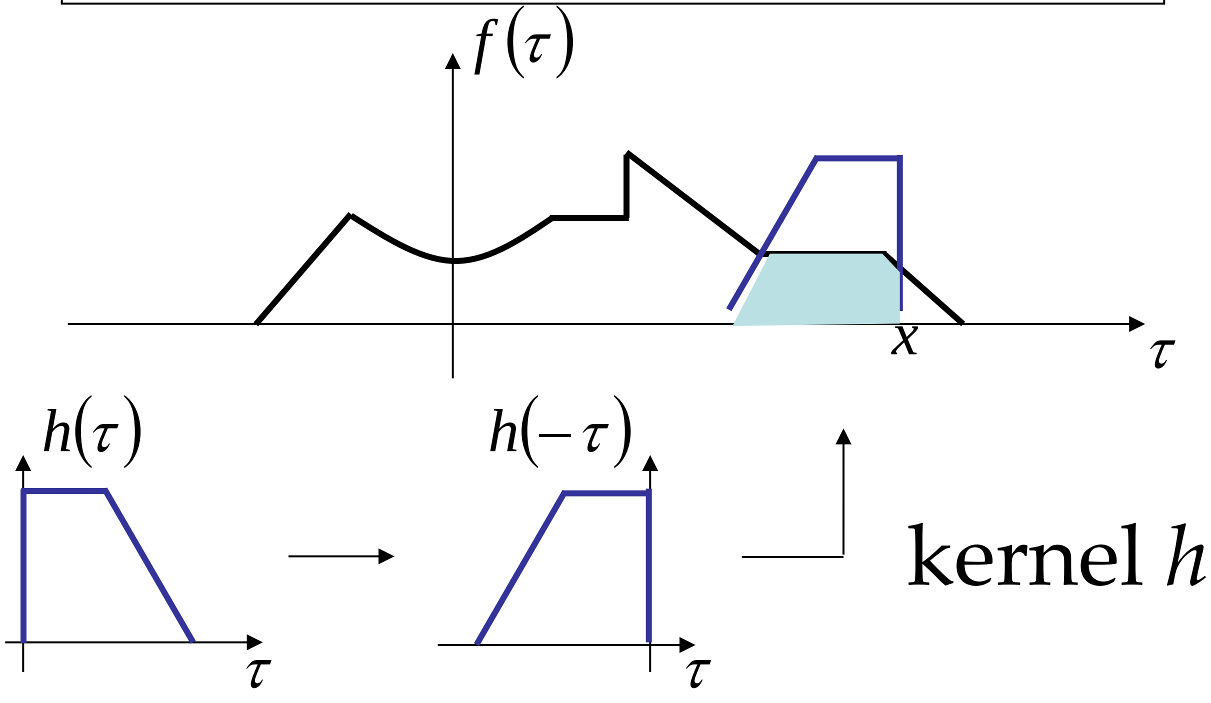
Convolution

- Simple way to process an image.
- Mask defines the processing function.
- Corresponds to a multiplication in frequency domain.

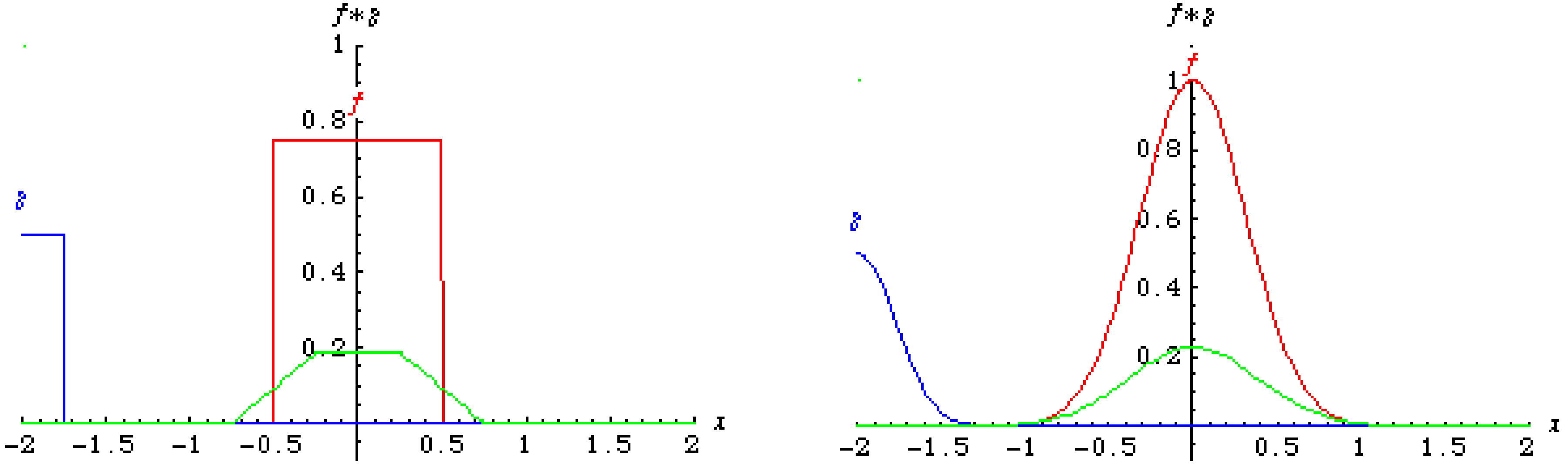


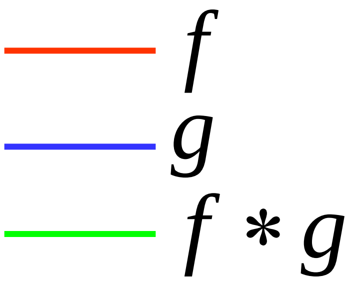
Convolution

$$g(x) = \int_{-\infty}^{\infty} f(\tau)\,h(x-\tau)\,d\tau \qquad g = f * h$$



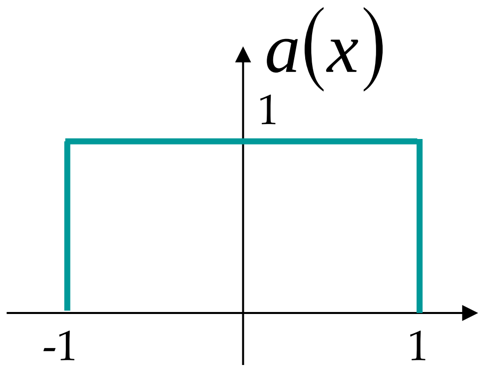
Convolution - Example

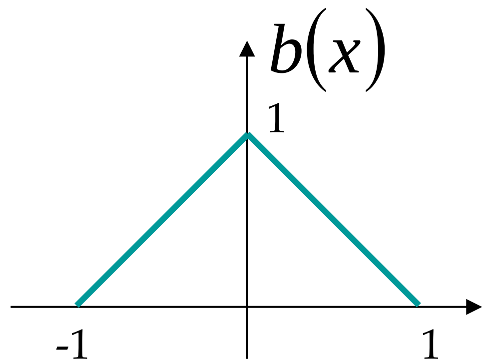




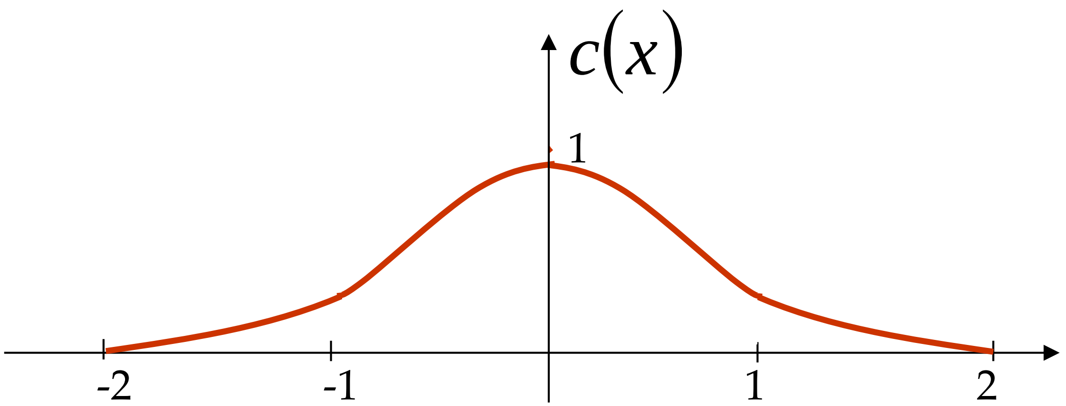
Eric Weinstein’s Math World

Convolution - Example





$$c = a * b$$



Properties of Convolution

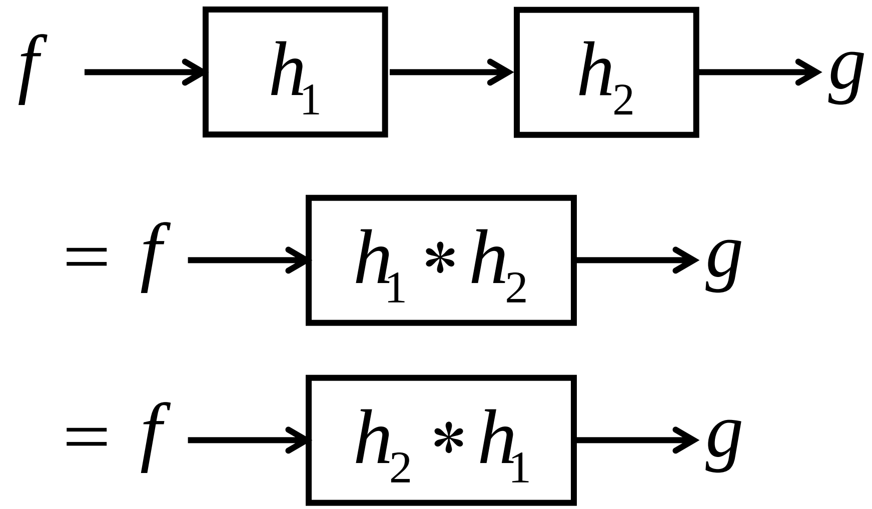
- Commutative

$$a * b = b * a$$

- Associative

$$(a * b) * c = a * (b * c)$$

- Cascade system



Example

- Each mask position has weight **w**.
- The result of the operation for each pixel is given by:

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

Mask

2	2	2
4	4	4
4	5	6

Image

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

=1*2+2*2+1*2+…
=8+0-20
=-12

Definitions

- Spatial filters
  - Use a **mask (kernel)** over an image region.
  - Work directly with pixels.
  - As opposed to: **Frequency filters**.
- Advantages
  - Simple implementation: **convolution** with the kernel function.
  - Different masks offer a **large variety of functionalities**.

Border Problem

$\frac{1}{16}$

1	2	1
2	4	2
1	2	1

How do we apply our mask to this pixel?

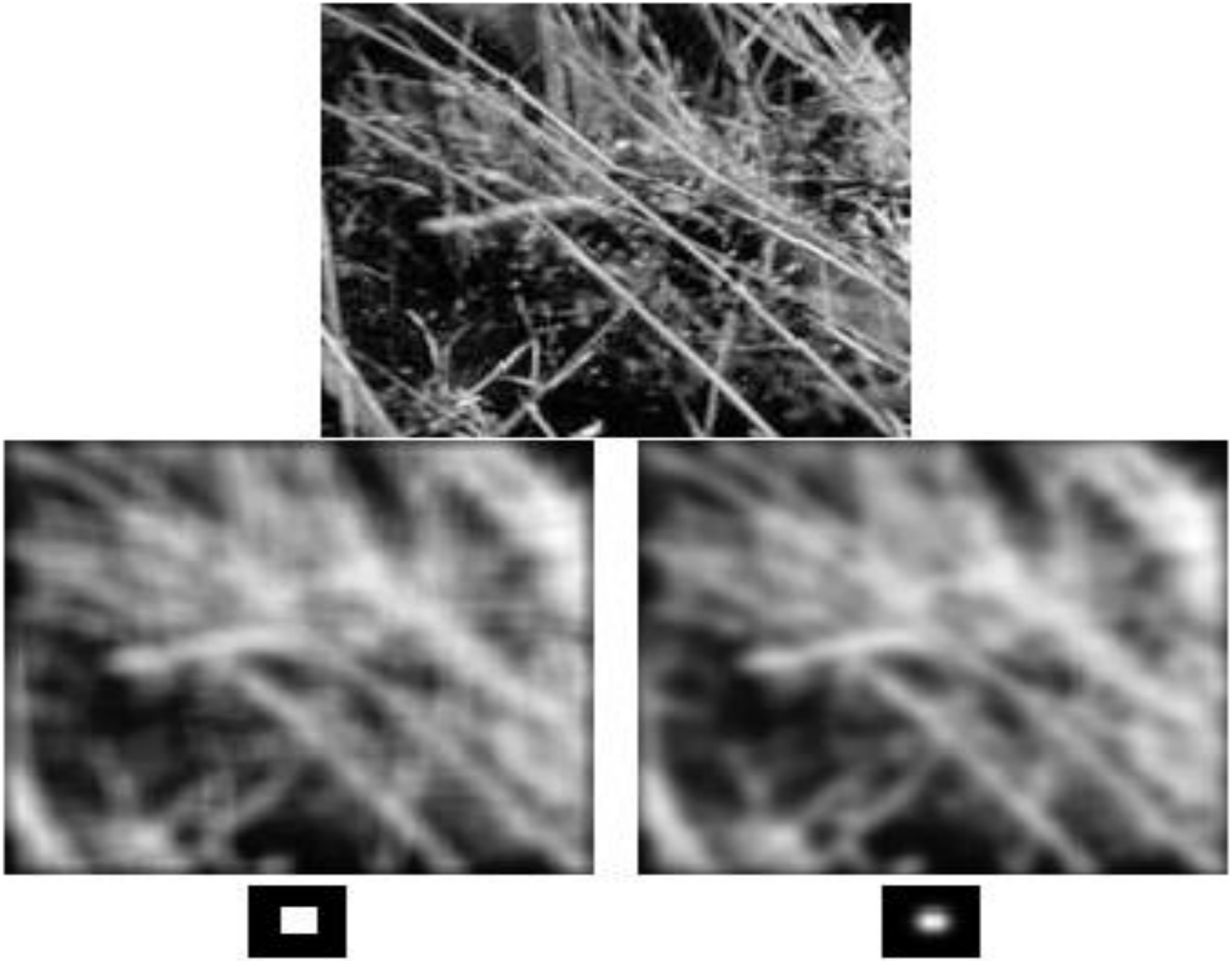
0	3	2	5	4	7	6	9	8
3	0	1	2	3	4	5	6	7
2	1	0	3	2	5	4	7	6
5	2	3	0	1	2	3	4	5
4	3	2	1	0	3	2	5	4
7	4	5	2	3	0	1	2	3
6	5	4	3	2	1	0	3	2
9	6	7	4	5	2	3	0	1
8	7	6	5	4	3	2	1	0

What a computer sees

Border Problem

- Ignore
  - Output image will be smaller than original
- Pad with constant values
  - Can introduce substantial 1st order derivative values
- Pad with reflection
  - Can introduce substantial 2nd order derivative values

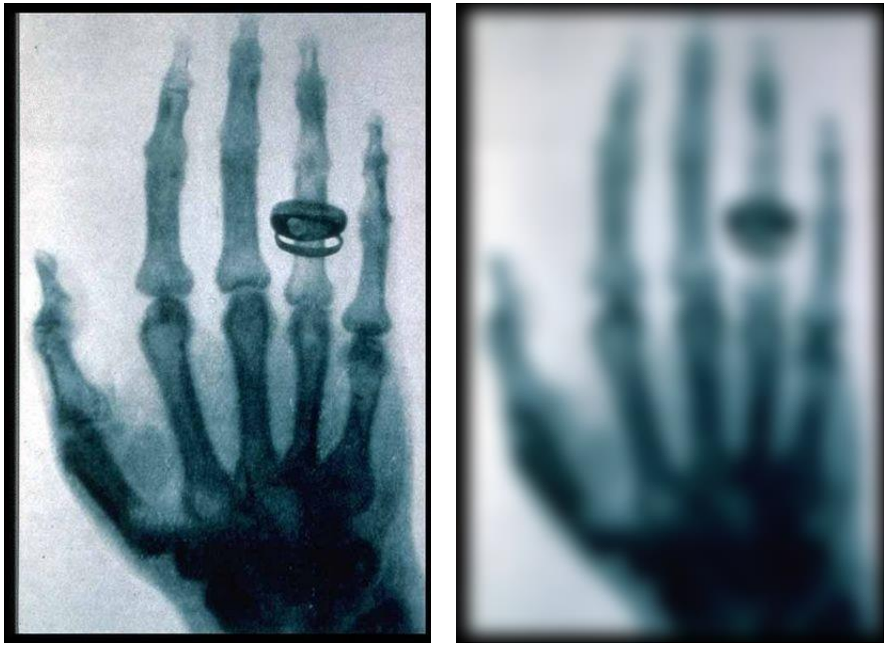
Smoothing



Mean filter Gaussian filter

Mean Filtering

- We are degrading the energy of the high spatial frequencies of an image (**low-pass filtering**).
  - Makes the image 'smoother'.
  - Used in noise reduction.
- Can be implemented with spatial masks or in the frequency domain.

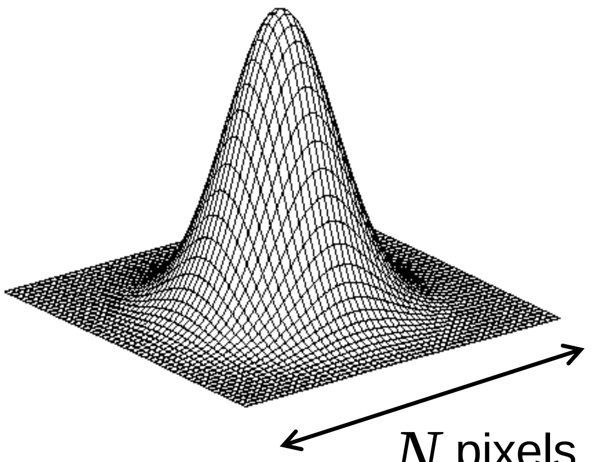


1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Gaussian Smoothing

Gaussian kernel

$$h(i, j) = \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2}\left(\frac{i^2+j^2}{\sigma^2}\right)}$$



Filter size $N \propto \sigma$ …can be very large

(truncate, if necessary)

$$g(i, j) = \frac{1}{2\pi\sigma^2} \sum_{m=1} \sum_{n=1} e^{-\frac{1}{2}\left(\frac{m^2+n^2}{\sigma^2}\right)} f(i-m, j-n)$$

2D Gaussian is separable!

$$g(i, j) = \frac{1}{2\pi\sigma^2} \sum_{m=1} e^{-\frac{1}{2}\frac{m^2}{\sigma^2}} \sum_{n=1} e^{-\frac{1}{2}\frac{n^2}{\sigma^2}} f(i-m, j-n)$$

Use two 1D Gaussian Filters!

Gaussian Smoothing

- A Gaussian kernel gives less weight to pixels further from the center of the window

$H[u, v]$

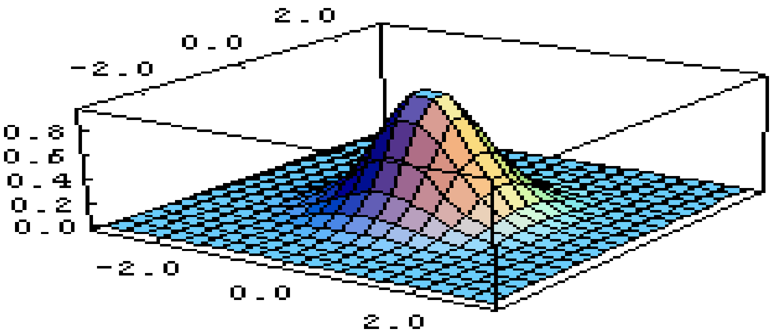
$\frac{1}{16}$

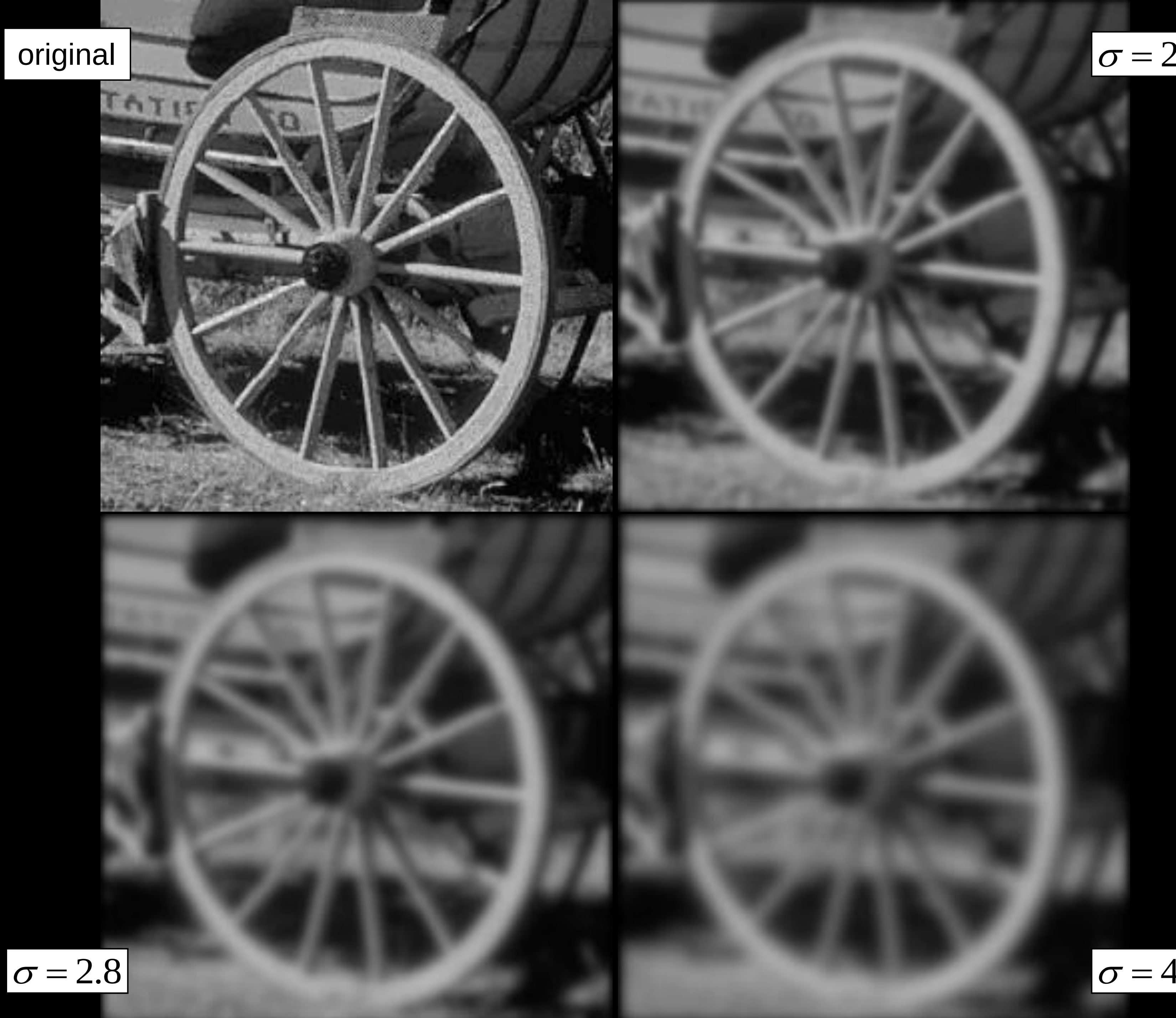
1	2	1
2	4	2
1	2	1

- This kernel is an approximation of a Gaussian function:

$F[x, y]$

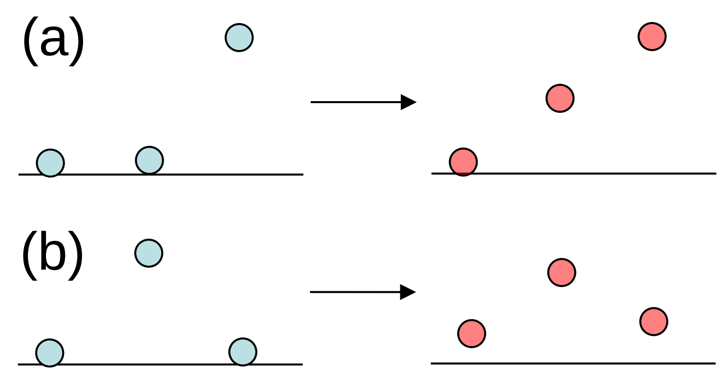
$$h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$



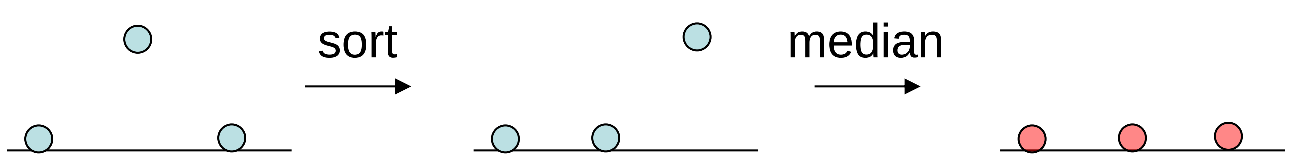

original
σ = 2
σ = 2.8
σ = 4

Median Filter

- Smoothing is averaging
  - (a) Blurs edges
  - (b) Sensitive to outliers

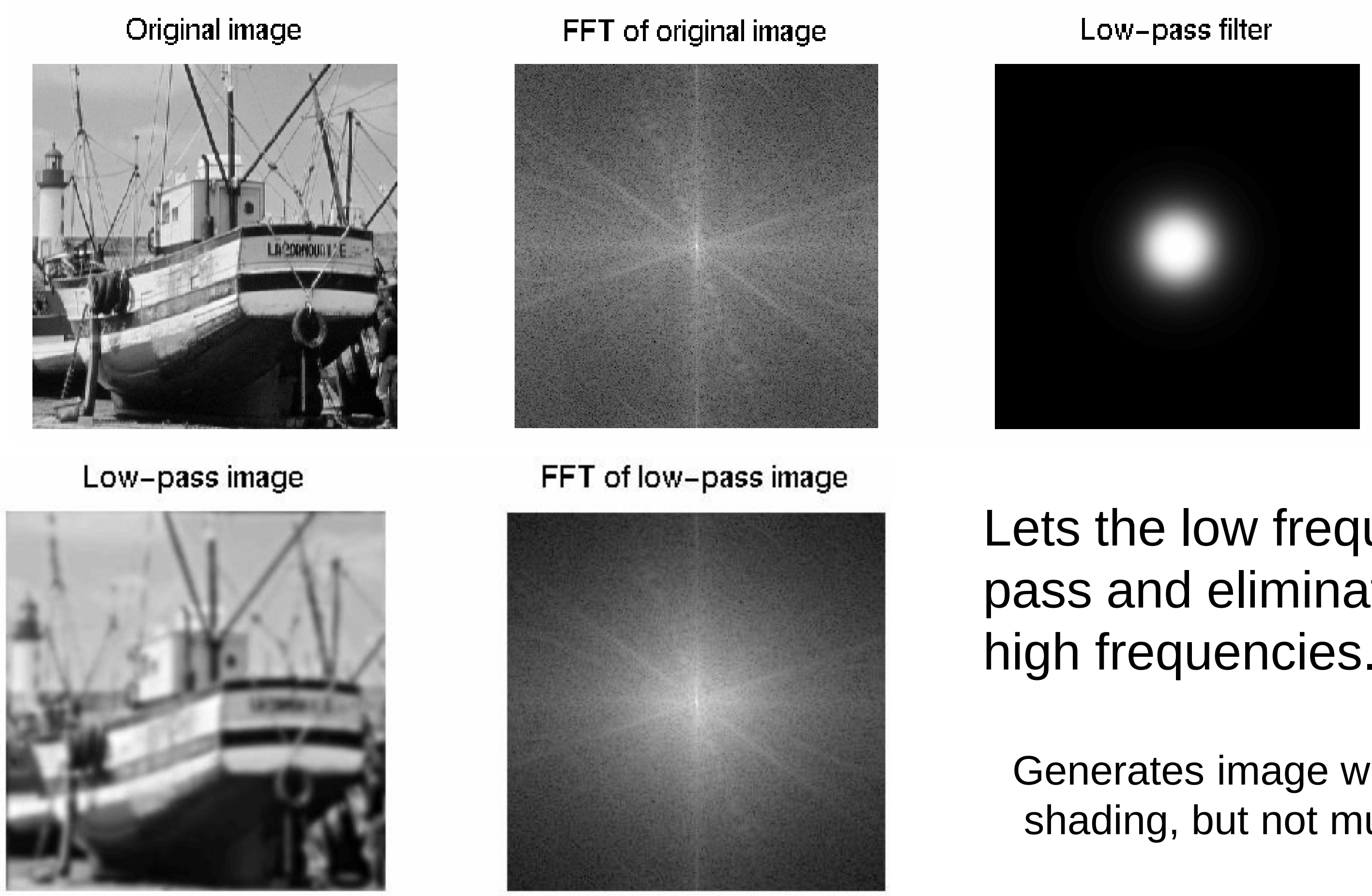


- Median filtering
  - Sort $N^2 - 1$ values around the pixel
  - Select middle value (median)



  - Non-linear (Cannot be implemented with convolution)

Low-pass Filtering



Lets the low frequencies pass and eliminates the high frequencies.

Generates image with overall shading, but not much detail

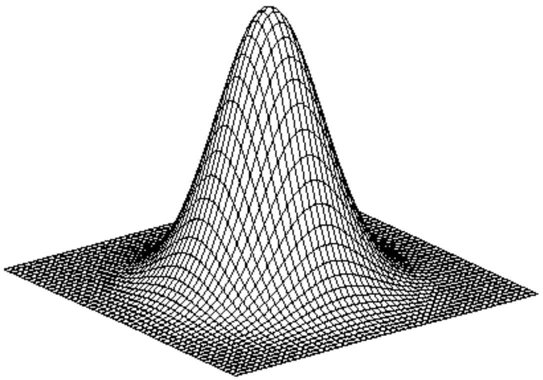
Edge Detection

The Sobel Operators

- Better approximations of the gradients exist
  - The *Sobel* operators below are commonly used

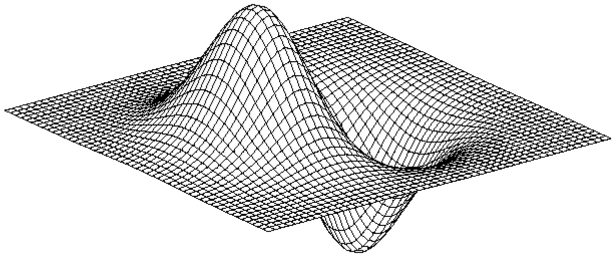
$$s_x = \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad s_y = \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Laplacian of Gaussian (LoG)



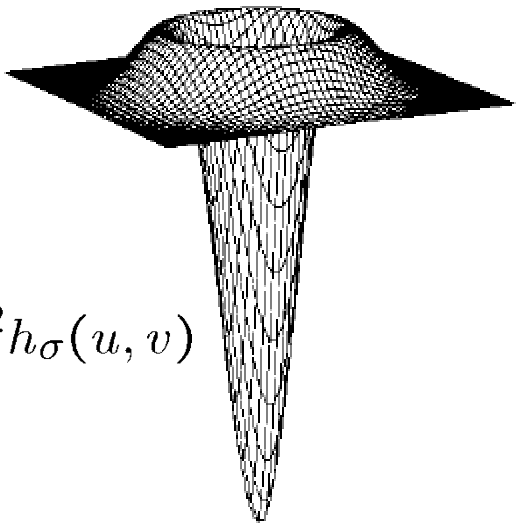
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

Gaussian



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

Derivative of Gaussian (DoG)



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

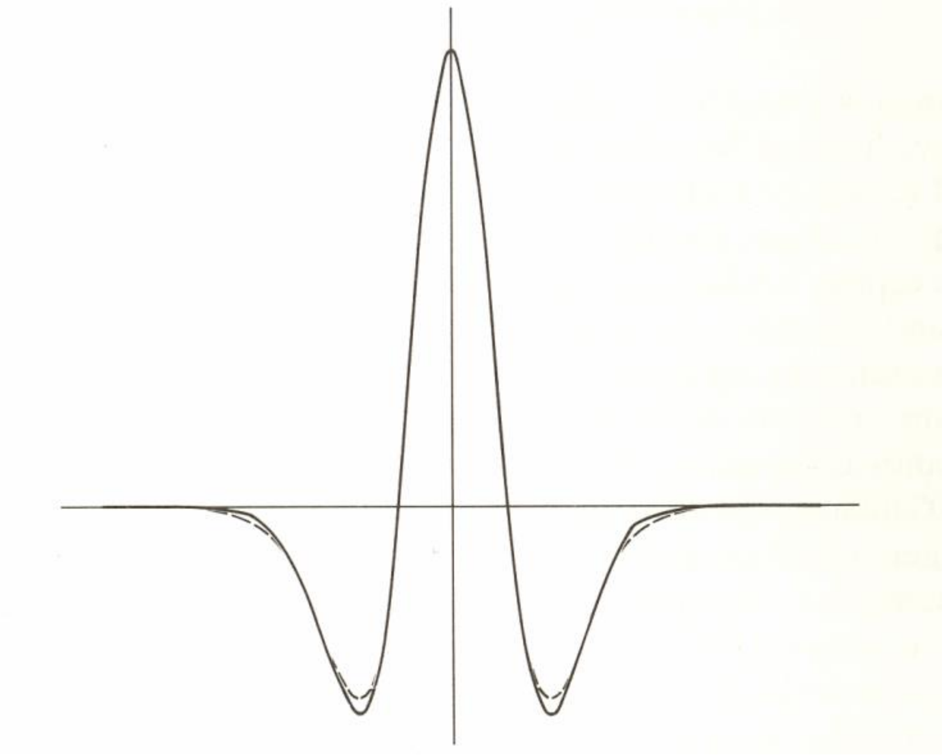
Laplacian of Gaussian

Mexican Hat (Sombrero)

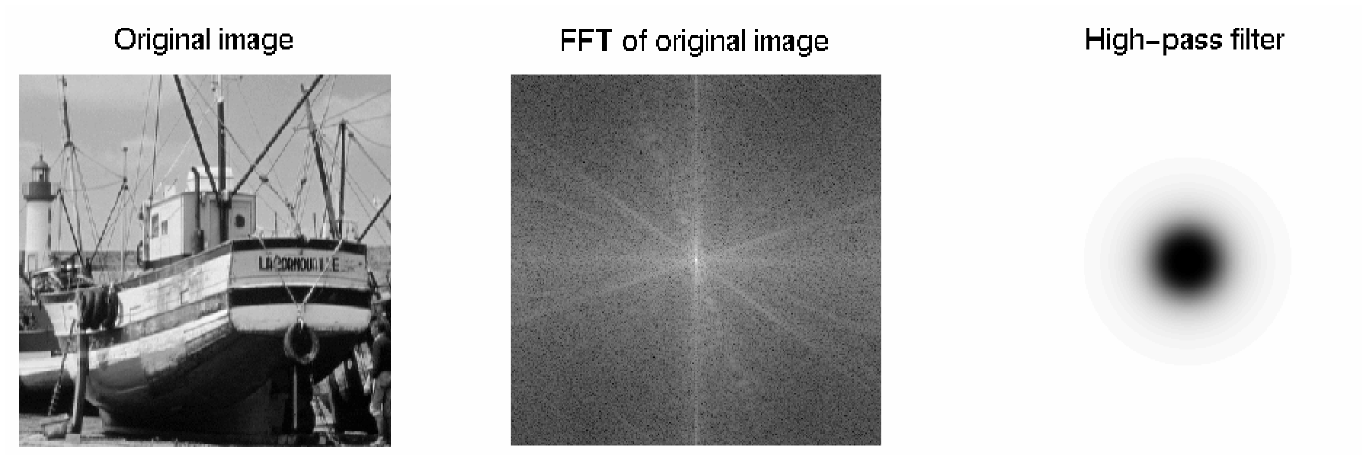
- ∇^2 is the **Laplacian** operator: $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

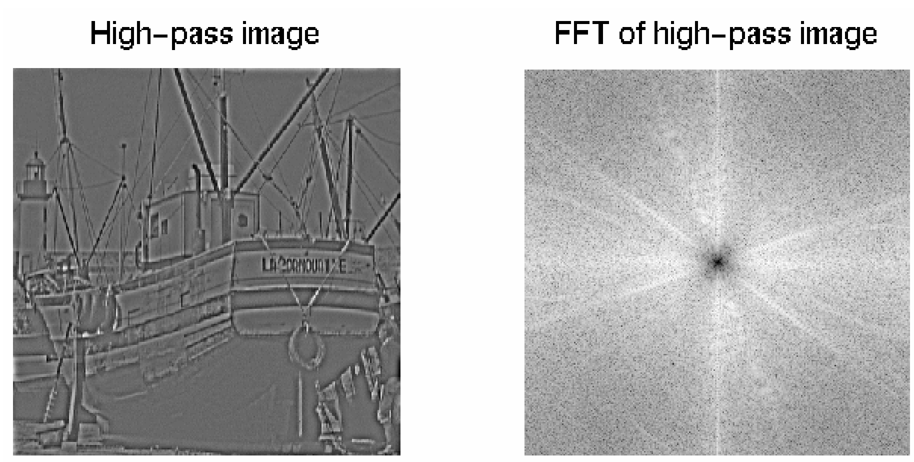
Difference of Gaussians (DoG)

- Laplacian of Gaussian can be approximated by the difference between two different Gaussians



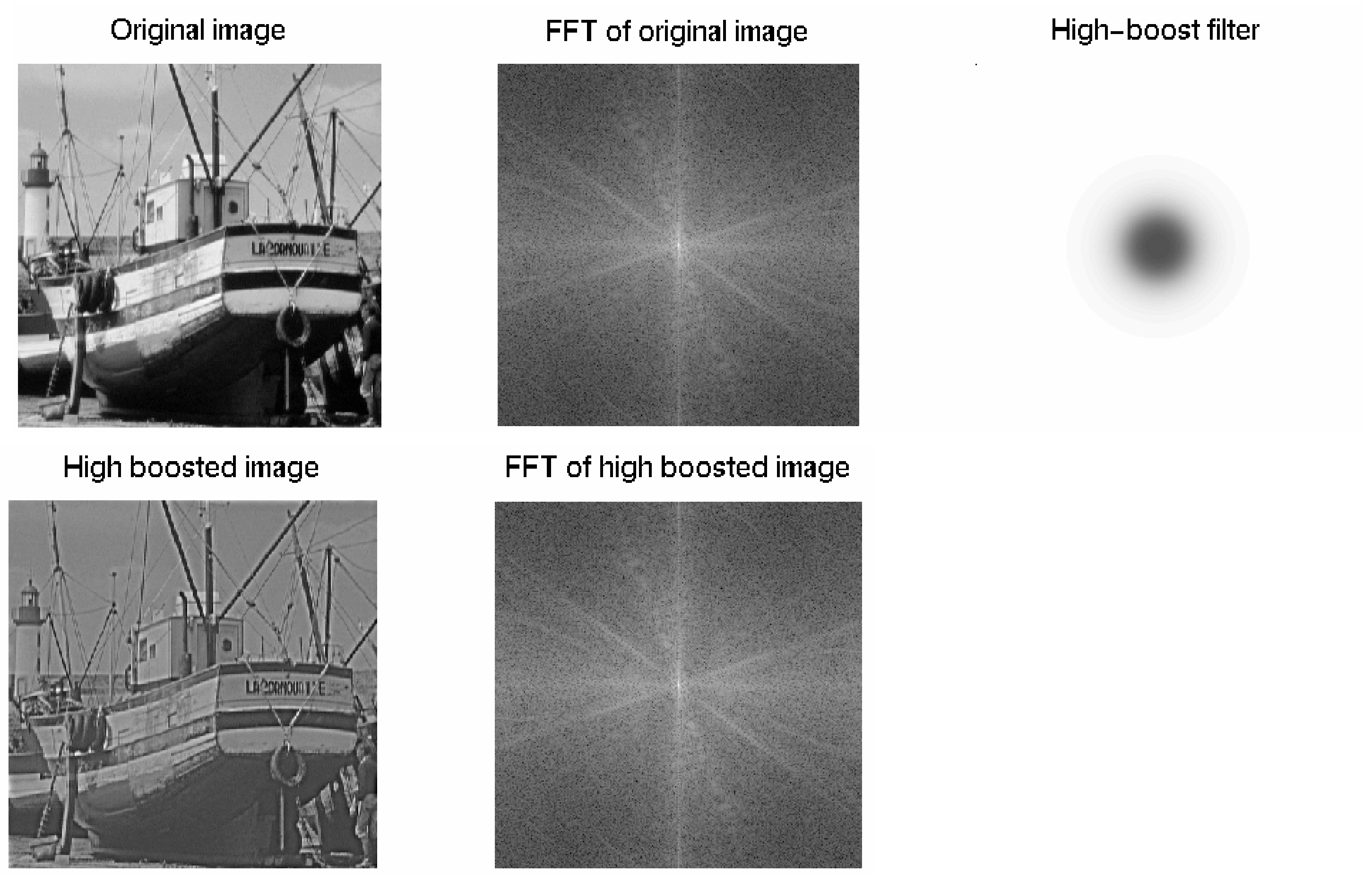
High-pass Filtering





Lets through the high frequencies (the detail), but eliminates the low frequencies (the overall shape). It acts like an edge enhancer.

Boosting High Frequencies



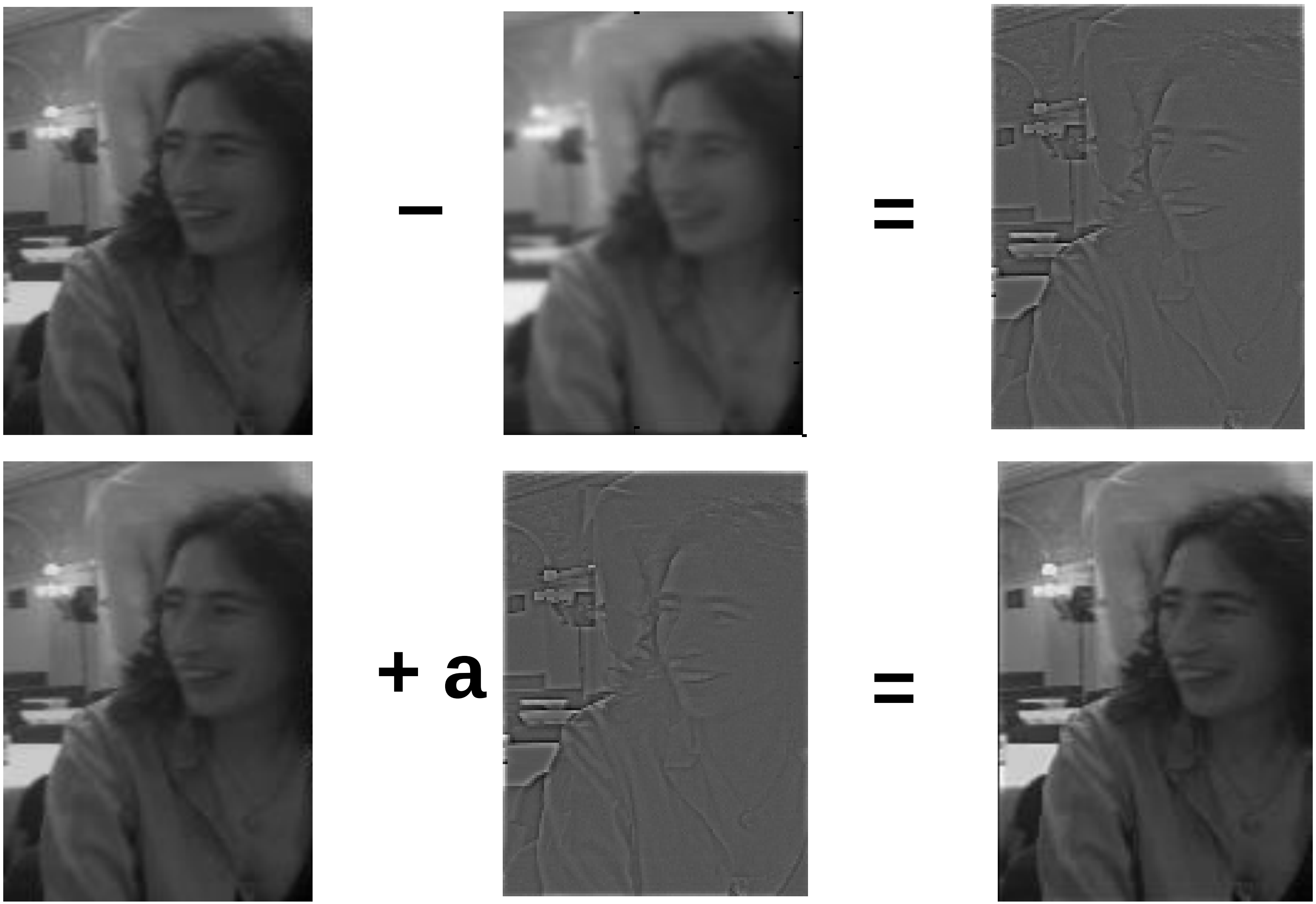
original image

magnitude of the gradient

DoG Edge Detection

(a) $\sigma = 1$ (b) $\sigma = 2$ (b)-(a)

Unsharp Masking



Summary

- Digital Filters are a useful tool to process 2D images
  - Rendered images, textures, frame buffers
- Convolution is a simple and powerful tool
- Various possible effects
  - Smoothing, edge detection, unsharp, field of view, motion blur, anti-aliasing