Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Computer Vision
Digital Filters – Spatial Domain

WS 2019/2020

Prof. Dr. Simone Frintrop

Computer Vision Group, Department of Informatics
University of Hamburg, Germany

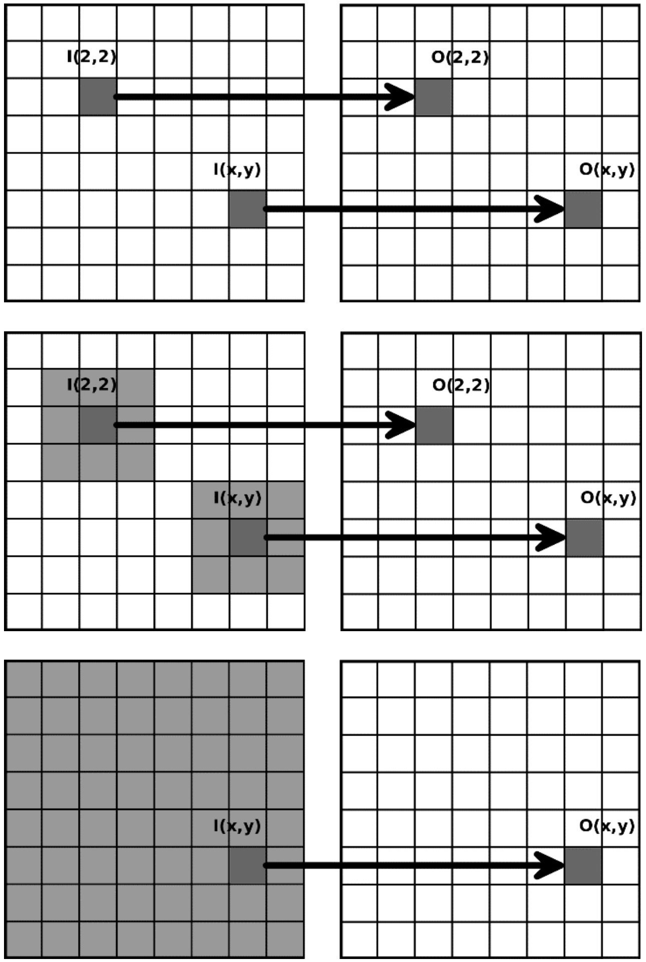
Content

- Spatial filters
- Correlation and Convolution
- Smoothing filters: Box filter, Gaussian
- Separable filters
- Difference-of-Gaussian Filters
- Image Pyramids
- Non-linear filters: median, max, min

Spatial Operations

Categories of spatial operations

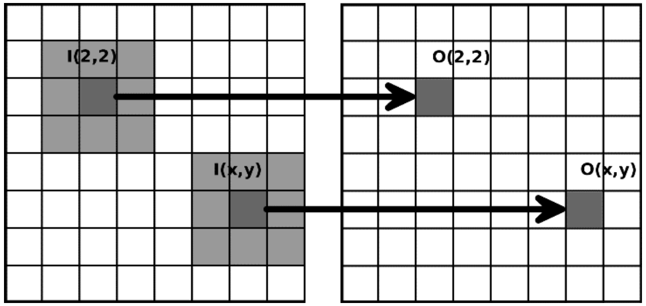
- **Point operations**
  (also: single-pixel operations).
  Area of support: single pixel.
  Transform image by adjusting single pixels
  E.g.: contrast adjustment, color transformation
- **Neighborhood operations:**
  Area of support: local neighborhood of pixel.
  change a pixel value according to a
  local neighborhood
  E.g.: smoothing or edge detection
- **Global operations:**
  Area of support: whole image.
  regard whole image to change pixel
  E.g.: histogram equalization or
  Fourier transformation



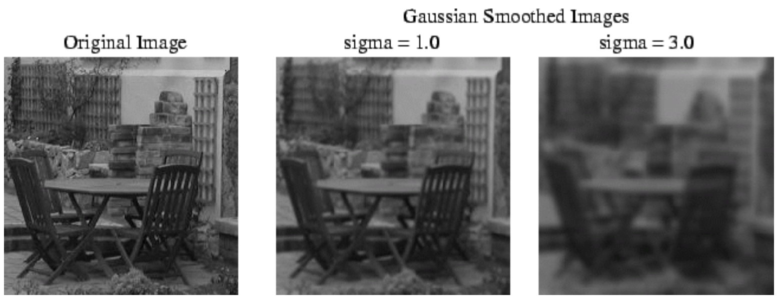
Spatial Operations

Today: Neighborhood operations: Spatial filters

- **Neighborhood operations:**
  Area of support: local neighborhood of pixel.
  change a pixel value according to a
  local neighborhood
  E.g.: smoothing or edge detection

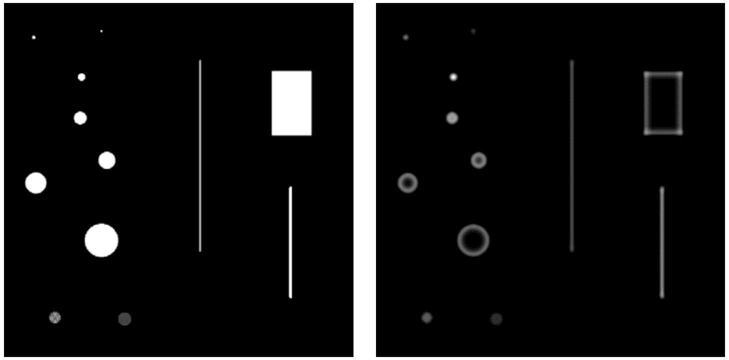


Why do we need digital filters?



Smooting

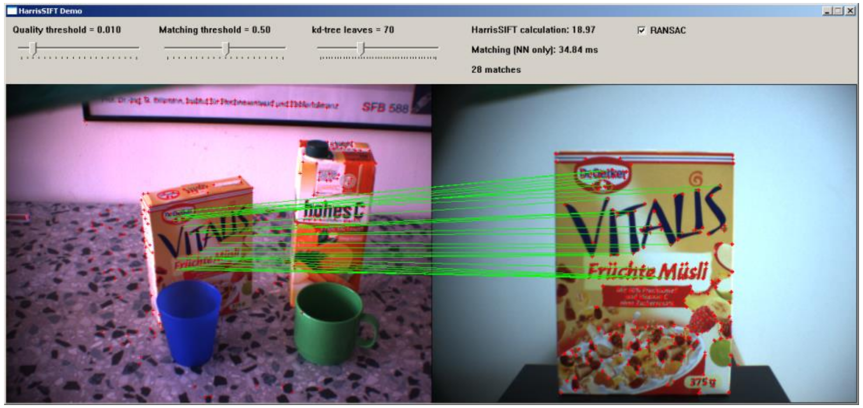
Edge Detection



Blob detection

Why do we need digital filters?

First step in many (most?) computer vision applications, e.g. keypoint-based object instance recognition:



Why do we need digital filters?

Essential components in Convolutional Neural Networks

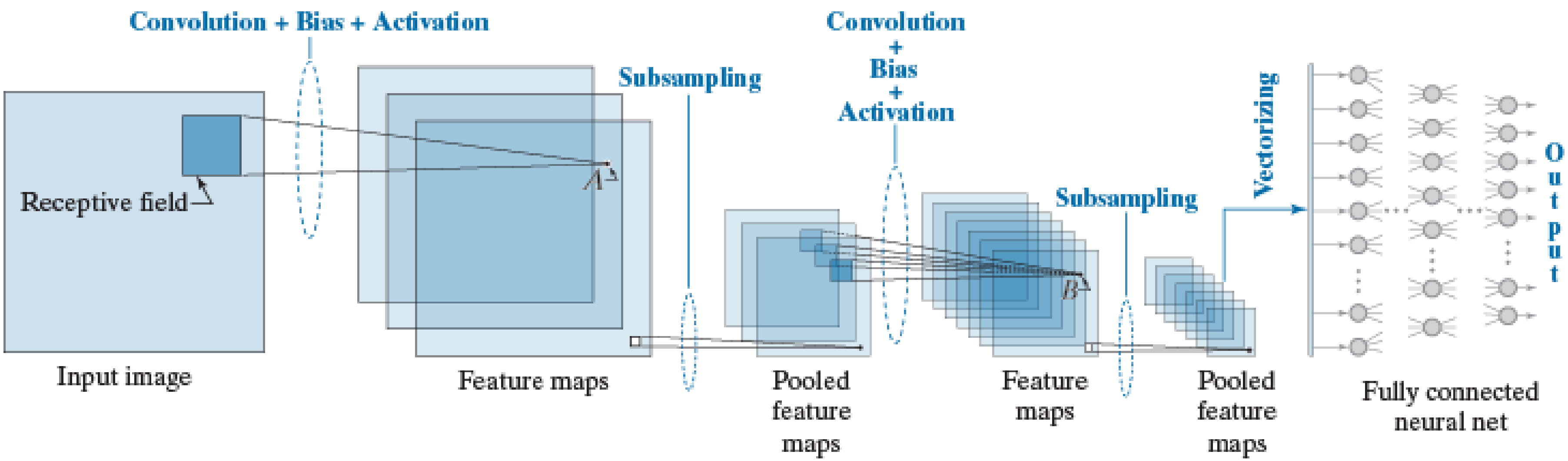


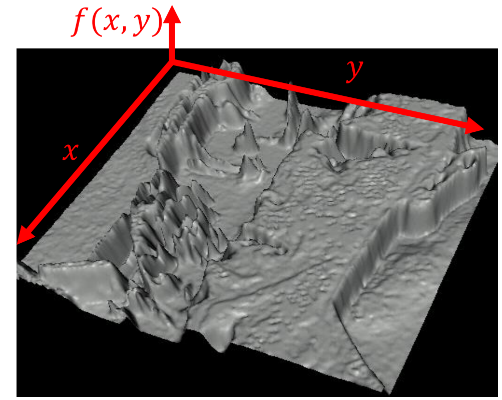
FIGURE 13.40 A CNN containing all the basic elements of a LeNet architecture. Points A and B are specific values to be addressed later in this section. The last pooled feature maps are vectorized and serve as the input to a fully connected neural network. The class to which the input image belongs is determined by the output neuron with the highest value.

[Gonzales/Woods 2018]

Image Processing Domains

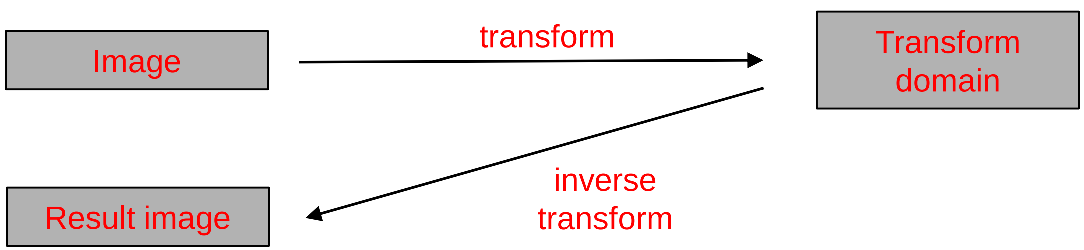
Images can be processed in the

- Spatial domain: the image plane itself



Today: the spatial domain

- Transform domain: e.g. the frequency domain



[Image: Steve Seitz]

Spatial filters

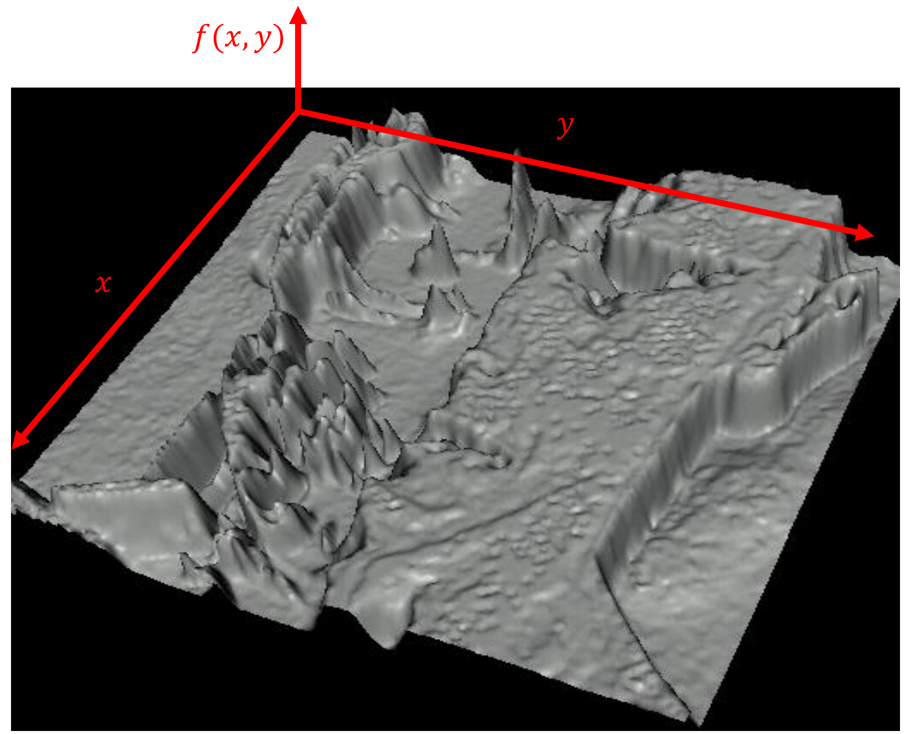
- Spatial filters: neighborhood operations
- Area of support: local neighborhood of pixel

- Why “filter”?
  They filter out some frequencies:
  - **Low-pass filter**: the low frequencies remain (“pass”), the high frequencies are removed. Effect: smoothing, removing noise
  - **High-pass filter**: the high frequencies remain, the low frequencies are removed. Effect: sharpening, edge detection
  - **Band-pass filter**: only frequencies within a certain “band” of frequencies remain. Effect: find edges, lines, blobs (corners)

The Image as Signal

Remember

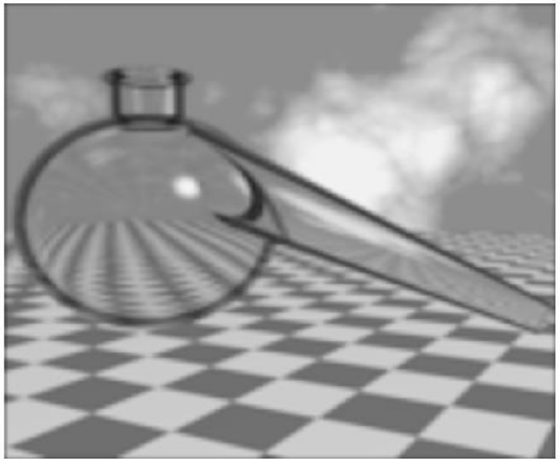
an image as a (continuous) signal or function $f(x, y)$:



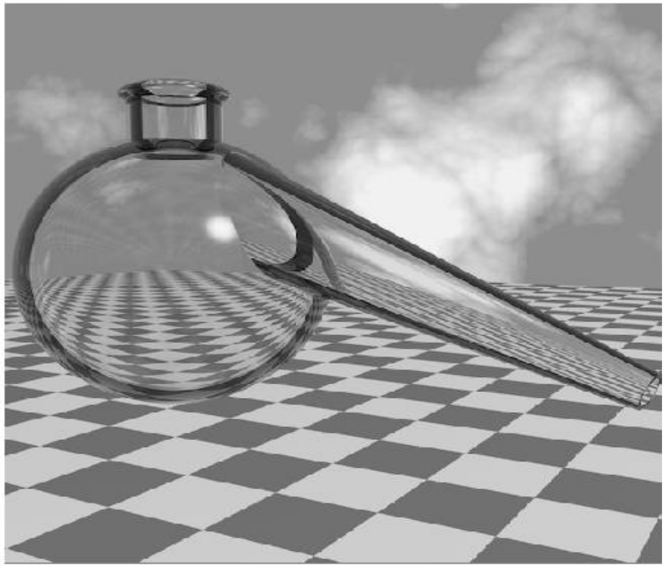
[Images: Steve Seitz]

Spatial filters

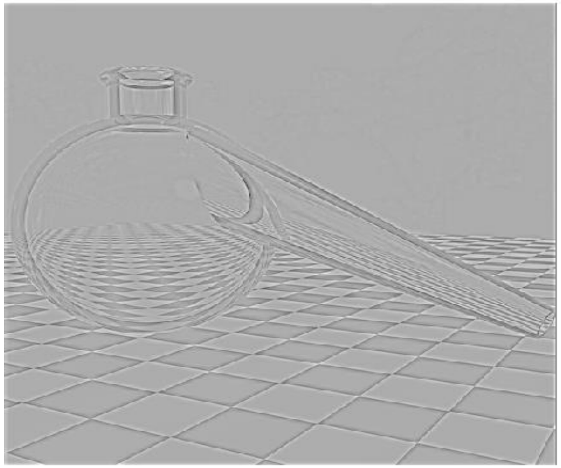
Low-pass vs. high pass



Low-pass filtered



Original image



High-pass filtered

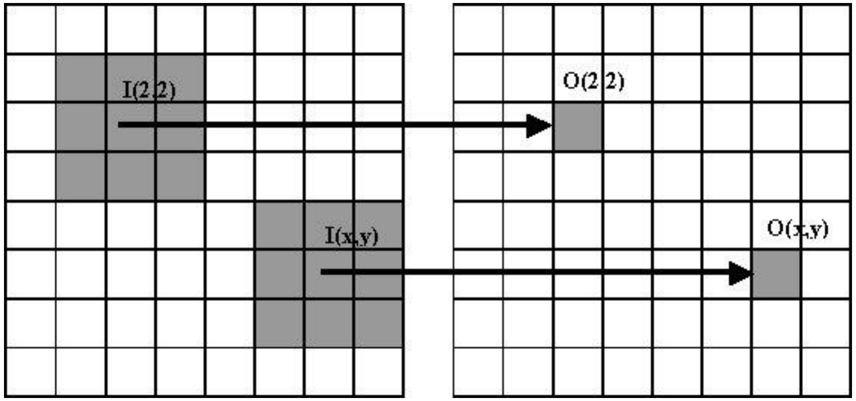
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Spatial filters

- Spatial (digital) filters consist of
  - A neighborhood
  - A predefined operation

- If the operation is linear, the filter is a linear filter

- Filter neighborhoods are also called filter masks, kernels, templates, or windows

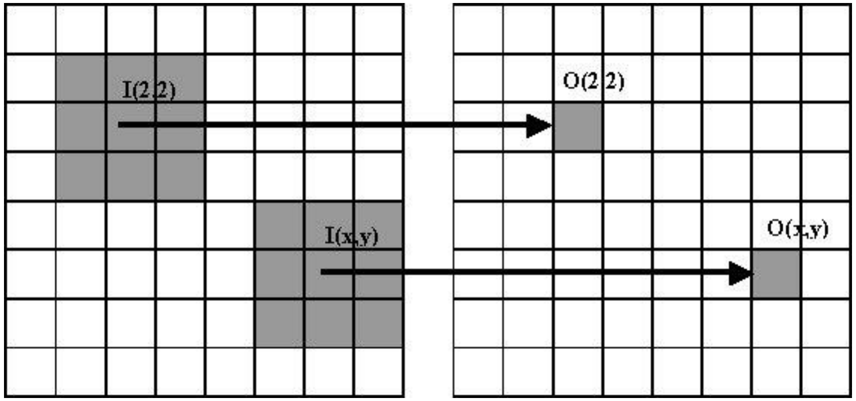


Spatial filters

- Filtering an image means to apply a spatial filter ***H*** successively to the whole image ***F***

- Two important operations:
  - Convolution and
  - Cross-correlation

- Both move a filter mask over image and compute a (weighted) average



Let's try to replace each pixel with an average of all the values in its neighborhood…

Moving Average in 2D

$F[x, y]$

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

$G[x, y]$

	0								

[Images: Steve Seitz]

Moving Average in 2D

$F[x,y]$

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

$G[x,y]$

	0	10							

[Images: Steve Seitz]

Moving Average in 2D

$F[x, y]$

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

$G[x, y]$

	0	10	20						

[Images: Steve Seitz]

Moving Average in 2D

$F[x, y]$

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

$G[x, y]$

	0	10	20	30					

[Images: Steve Seitz]

Moving Average in 2D

$F[x, y]$

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

$G[x, y]$

	0	10	20	30	30				

[Images: Steve Seitz]

Moving Average in 2D

$$F[x, y]$$

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

$$G[x, y]$$

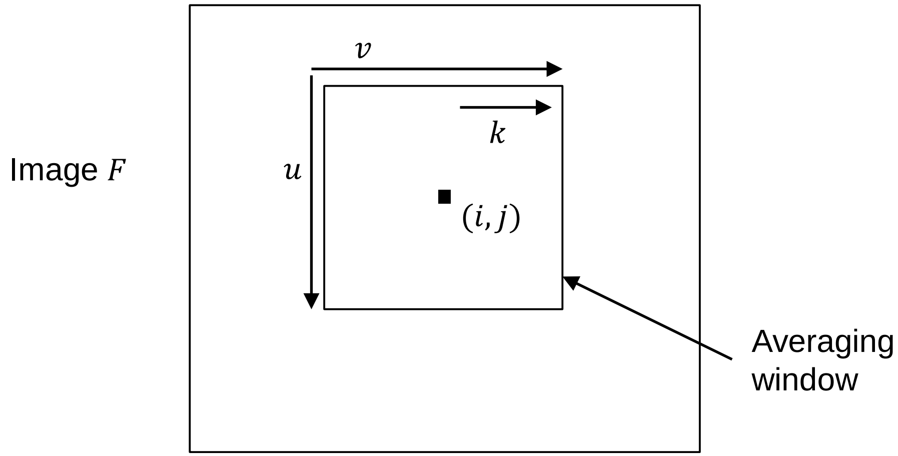
	0	10	20				20	10	
	0	20	40	60	60	60	40	20	
	0		60	90	90	90	60		
	0		50	80	80	90	60		
	0		50	80	80	90	60		
	0	20		50	50	60	40	20	
	10	20					20	10	
	10	10	10	0	0	0	0	0	

[Images: Steve Seitz]

Correlation Filtering

- Say the averaging window size is $2k+1 \times 2k+1$:

$$G[i,j] = \underbrace{\qquad\qquad\qquad}_{\text{Attribute uniform weight to each pixel}} \underbrace{\sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]}_{\text{Loop over all pixels in neighborhood around image pixel } F[i,j]}$$



Correlation Filtering

- Say the averaging window size is $2k+1 \times 2k+1$:

$$G[i,j] = \underbrace{\frac{1}{(2k+1)^2}}_{\text{Attribute uniform weight to each pixel}} \underbrace{\sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]}_{\text{Loop over all pixels in neighborhood around image pixel } F[i,j]}$$

- Now generalize to allow different weights depending on neighboring pixel's relative position:

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} \underbrace{H[u,v]}_{\text{Non-uniform weights}} F[i+u, j+v]$$

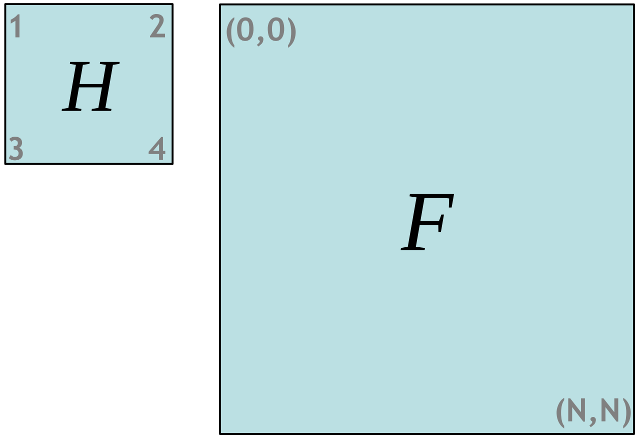
Correlation Filtering

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u, j+v]$$

- This is called cross-correlation
(or simply correlation), denoted as $G = H \otimes F$

The average filter
(box filter):

$$\left(\frac{1}{9}\right) * \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



Correlation Filtering

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v] F[i+u, j+v]$$

- This is called cross-correlation (or simply correlation), denoted as $G = H \otimes F$

The average filter (box filter):

$$\left(\frac{1}{9}\right) * \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

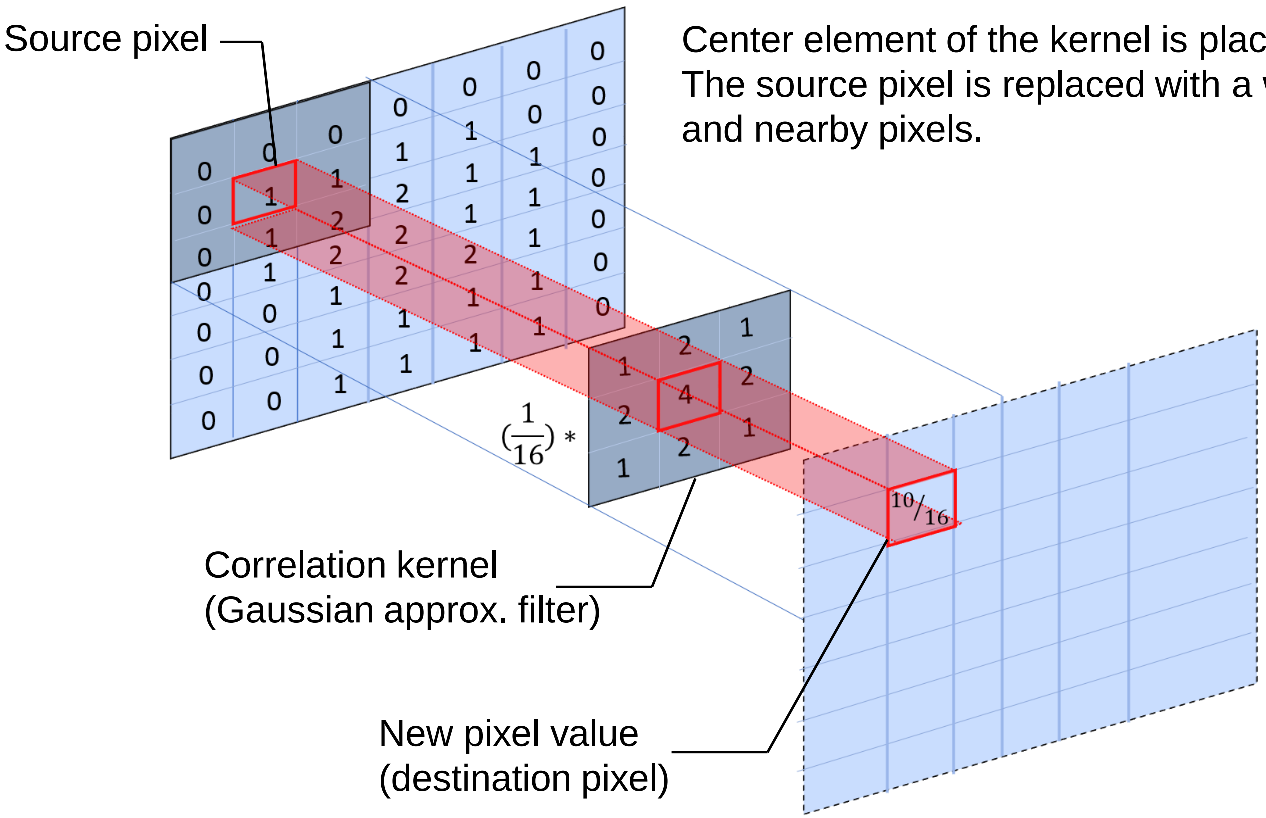
Smoothing filters have a normalization factor of: $\frac{1}{\sum_{u,v} H[u,v]}$
or the values are integrated into the filter mask $H[u,v]$:

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

Correlation Filtering

Correlation: a weighted sum where the filter provides the weights

Center element of the kernel is placed over the source pixel. The source pixel is replaced with a weighted sum of itself and nearby pixels.



$$
\begin{array}{rr}
 & (0 \times 1) \\
 & (0 \times 2) \\
 & (0 \times 1) \\
 & (0 \times 2) \\
 & (1 \times 4) \\
 & (1 \times 2) \\
 & (0 \times 1) \\
 & (1 \times 2) \\
+ & (2 \times 1) \\
\hline
 & 10
\end{array}
$$

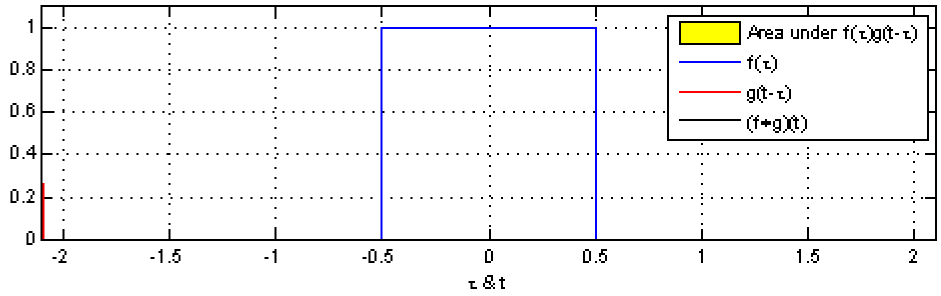
Normalize afterwards by 1/16

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Correlation

More general:

- Correlation of two signals multiplies its elements and sums up the results (sliding dot product).
- The result is a measure of how similar the two signals are. It is commonly used to search patterns in signals (e.g. images).
- This will be important for the Fourier analysis of signals, and for template matching of image patches



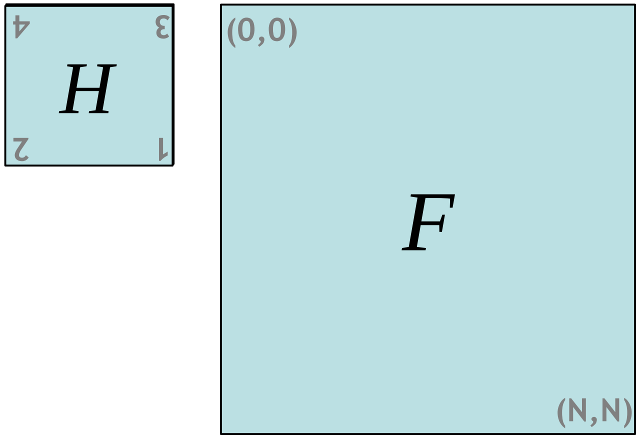
[Wikipedia: Convolution]

Convolution

- Convolution:
  - Flip the filter in both dimensions (bottom to top, right to left)
  - Then apply cross-correlation

$$G = H \star F$$

*Notation for convolution operator (often simply: H*F)*



This is the same as this equation:

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u, j-v]$$

Correlation vs. Convolution

- Correlation

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u, j+v]$$

$$G = H \otimes F$$

- Convolution

Note the difference!

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u, j-v]$$

$$G = H \star F$$

- Note:
If $H[-u,-v] = H[u,v]$, then correlation $\equiv$ convolution.

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Borders

What should we do at the borders?

1	1	1
1	1	1
1	1	1

Filter kernel g

1	12	10	20	10	10	10
10	10	11	10	10	10	10
10	10	11	11	11	1	10
10	20	1	11	11	10	0
0	4	1	1	1	10	10
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Input image f

Properties of convolution

f, g, h: signals (images or filters)
k: constant

- Commutativity: $f * h = h * f$
- Associativity: $(f * g) * h = f * (g * h)$
  - Often apply several filters h_i in sequence: $(((f * h_1) * h_2) * h_3)$
  - This is equivalent to applying one filter: $f * (h_1 * h_2 * h_3)$
- Distributivity: $h * (f_1 + f_2) = h * f_1 + h * f_2$
  (also additivity or superposition)
- Homogeneity: $k(f * g) = kf * g = f * kg$

  (also called: scaling property)

Properties of convolution

- Identity: $f * e = f$
  - for unit impulse $e = [\ldots, 0, 0, 1, 0, 0, \ldots]$.
- Differentiation: $\frac{\partial}{\partial x}(f \star g) = \frac{\partial f}{\partial x} \star g$

Linear operators

- Linear operator: an operator H is linear if it satisfies the properties for homogeneity and additivity (superposition)

$$H[kI_1 + kI_2] = H[kI_1] + H[kI_2] = kH[I_1] + kH[I_2]$$

(for images I_1 and I_2, and a constant k)

- Meaning: it does not matter whether we first apply the operator to each image independently and then add the two images, or if we first add the images and then apply the operator. Same for scaling with *k*.
- Important linear operators: correlation and convolution

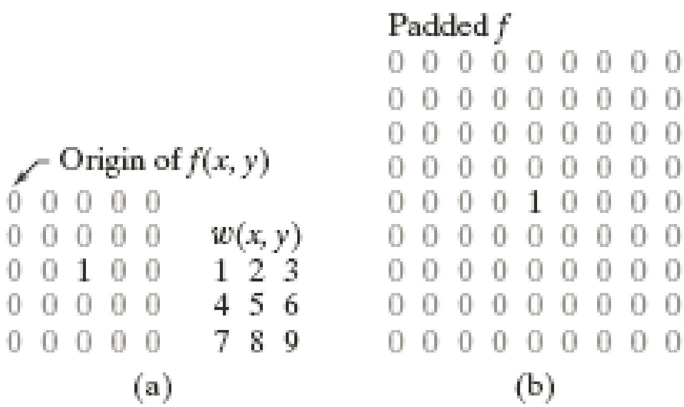
Properties of convolution

- Linear operators: Convolution and correlation are linear operators since they satisfy the properties for homogeneity and additivity (superposition)
  1. $k(f * g) = kf * g = f * kg$
  2. $h * (f_1 + f_2) = h * f_1 + h * f_2$

- Shift invariant: operator behaves the same everywhere, i.e., the value of the output depends on the pattern in the image neighborhood, not on the position of the neighborhood. Convolution and correlation are shift-invariant operators.

Properties of convolution

- Properties such as commutativity only apply if the signals are infinite
- To make them apply for finite signals such as images or kernels, we “simulate” infinity by padding all relevant items by zeros
- Q: How many lines of zeros do we need at each side for an $n \times n$ image f with a filter kernel g of size $m \times m$?
- A: $(m-1)$ zeros



[Gonzalez/Woods]

Spatial filters vs. cell response

Biology:

Cells respond to a spatially restricted area of the visual scene (the receptive field)

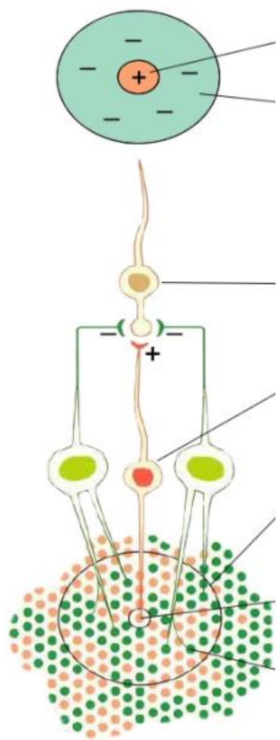
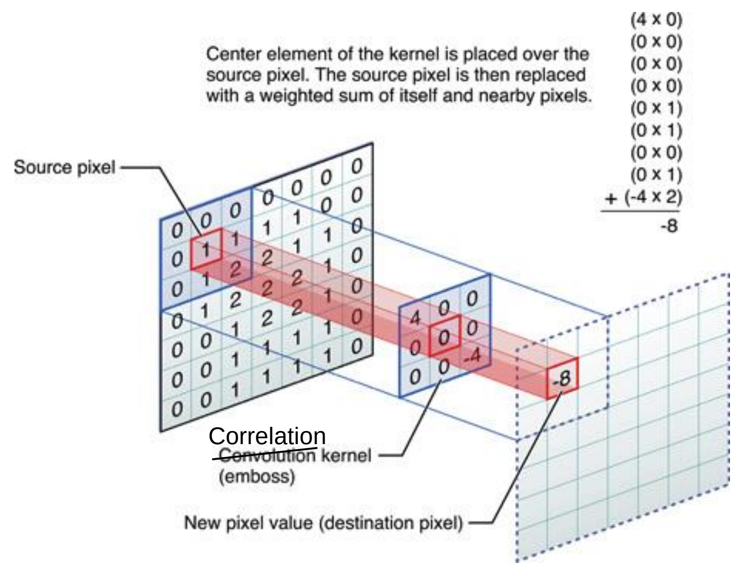


Image Processing:

Spatial Filters apply an operation (successively) to a local patch (neighborhood) of the image

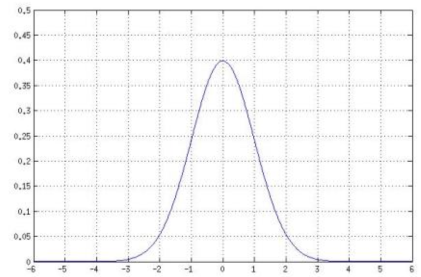


More in lecture Computer Vision II

The Gaussian Filter

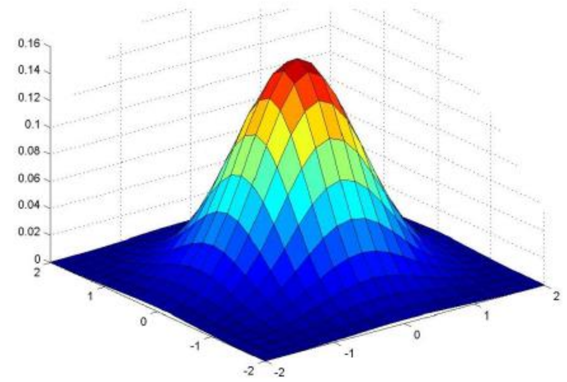
The 1D Gaussian:

$$g(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x^2)}{2\sigma^2}}$$



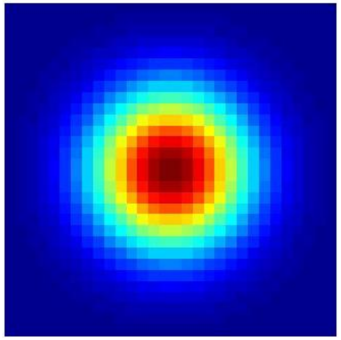
The 2D Gaussian:

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



The Gaussian Filter

The 2D Gaussian:



Look from the top

$$\frac{1}{273}$$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

Filter mask (5×5) for sigma 1.0

What parameters matter here?

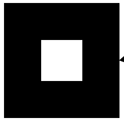
Remember

Spatial filters

- Remember: we have different types of spatial digital filters:
  - Low-pass filter: the low frequencies remain (“pass”), the high frequencies are removed. Effect: smoothing, removing noise
  - High-pass filter: the high frequencies remain, the low frequencies are removed. Effect: sharpening, edge detection
  - Band-pass filter: only frequencies within a certain “band” of frequencies remain. Effect: find edges, lines, blobs (corners)

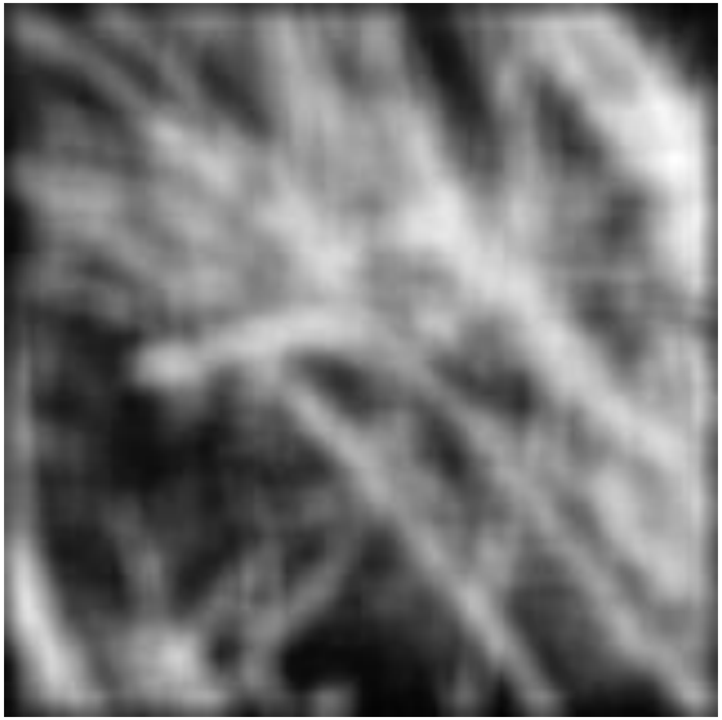
- Low pass filters:
  - most simple: box filter (moving average)
  - Better: Gaussian filter

Smoothing by Averaging



depicts box filter:
white = high value, black = low value

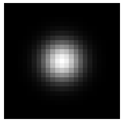
Original



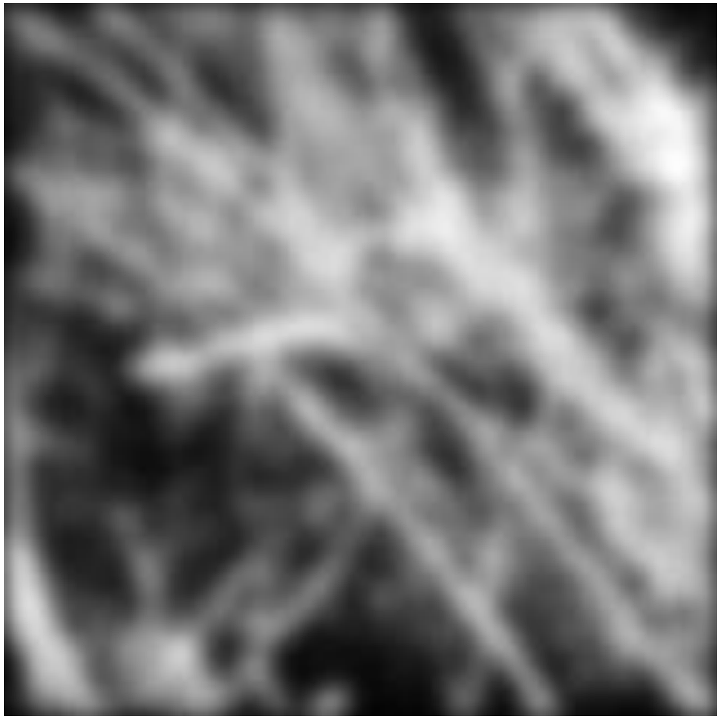
Filtered

[Image: Forsyth & Ponce]

Smoothing with a Gaussian



Original



Filtered

[Image: Forsyth & Ponce]

Successive Smoothing

- Applying Gaussians successively corresponds to applying one larger Gaussian, according to the equation:

$$G_3 = G_1 * G_2, \text{ with sigmas } \sigma_1,\ \sigma_2,\ \sigma_3, \text{ such that:}$$

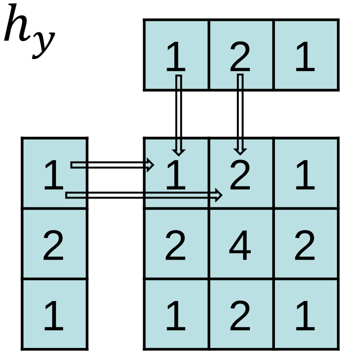
$$\sigma_3 = \sqrt{\sigma_1^2 + \sigma_2^2}$$

- Here, σ_3 is the *effective smoothing factor*.

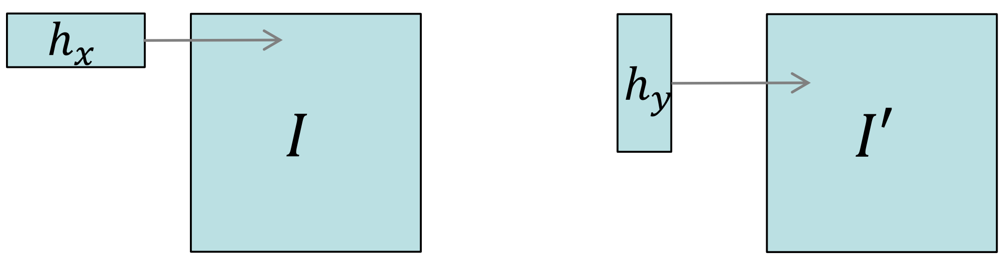
- Important e.g. for image pyramids

Separable filters

- **Definition:** A (2D) filter h is separable if it can be separated into two 1D filters h_x and h_y that can be combined to h by the outer product of h_x and h_y: $h = h_x \otimes h_y$

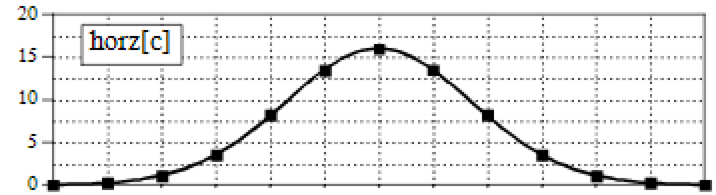


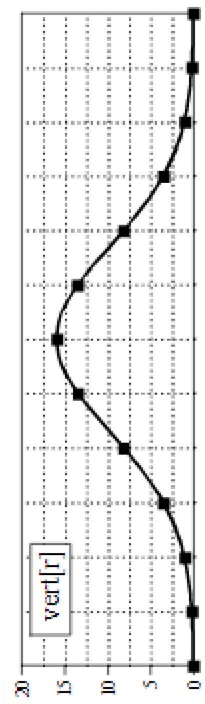
- Instead of applying h, apply first h_x and then h_y (faster!)



- The box filter and the Gaussian filter are separable

Separation of Gaussian



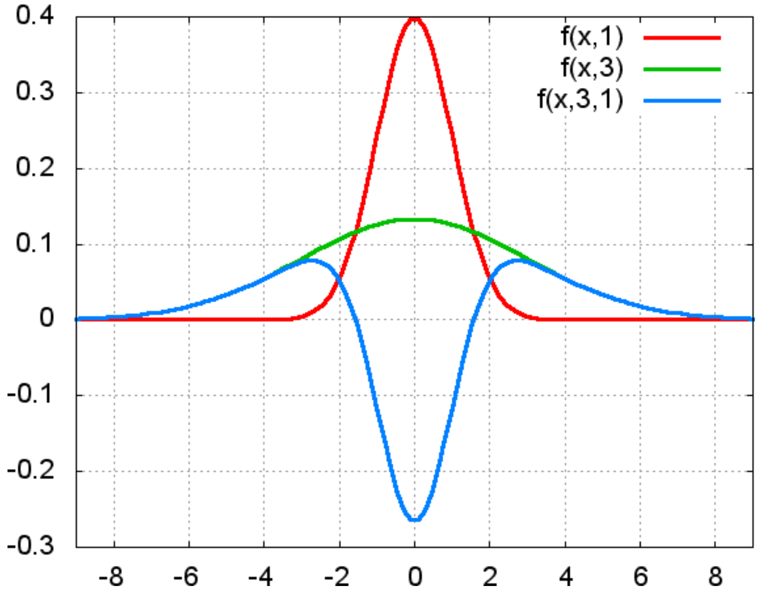


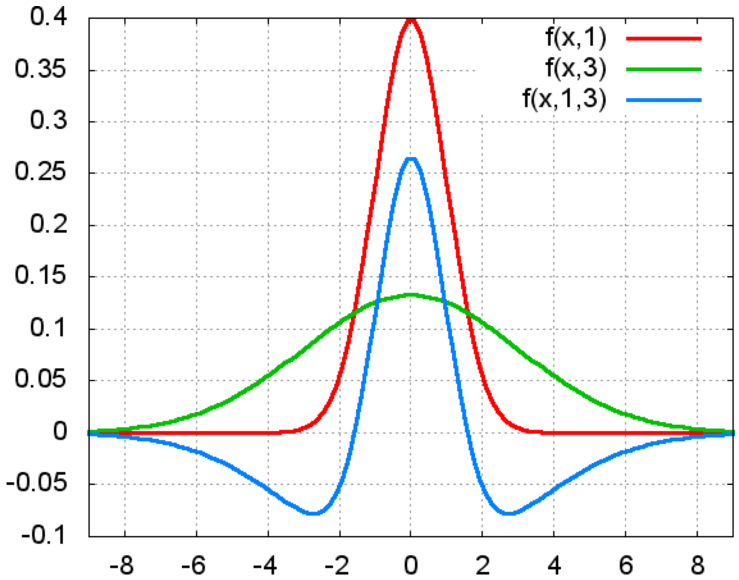
	0.04	0.25	1.11	3.56	8.20	13.5	16.0	13.5	8.20	3.56	1.11	0.25	0.04
0.04	0	0	0	0	0	1	1	1	0	0	0	0	0
0.25	0	0	0	1	2	3	4	3	2	1	0	0	0
1.11	0	0	1	4	9	15	18	15	9	4	1	0	0
3.56	0	1	4	13	29	48	57	48	29	13	4	1	0
8.20	0	2	9	29	67	111	131	111	67	29	9	2	0
13.5	1	3	15	48	111	183	216	183	111	48	15	3	1
16.0	1	4	18	57	131	216	255	216	131	57	18	4	1
13.5	1	3	15	48	111	183	216	183	111	48	15	3	1
8.20	0	2	9	29	67	111	131	111	67	29	9	2	0
3.56	0	1	4	13	29	48	57	48	29	13	4	1	0
1.11	0	0	1	4	9	15	18	15	9	4	1	0	0
0.25	0	0	0	1	2	3	4	3	2	1	0	0	0
0.04	0	0	0	0	0	1	1	1	0	0	0	0	0

[Smith 1997]

Difference of Gaussians (DoG)

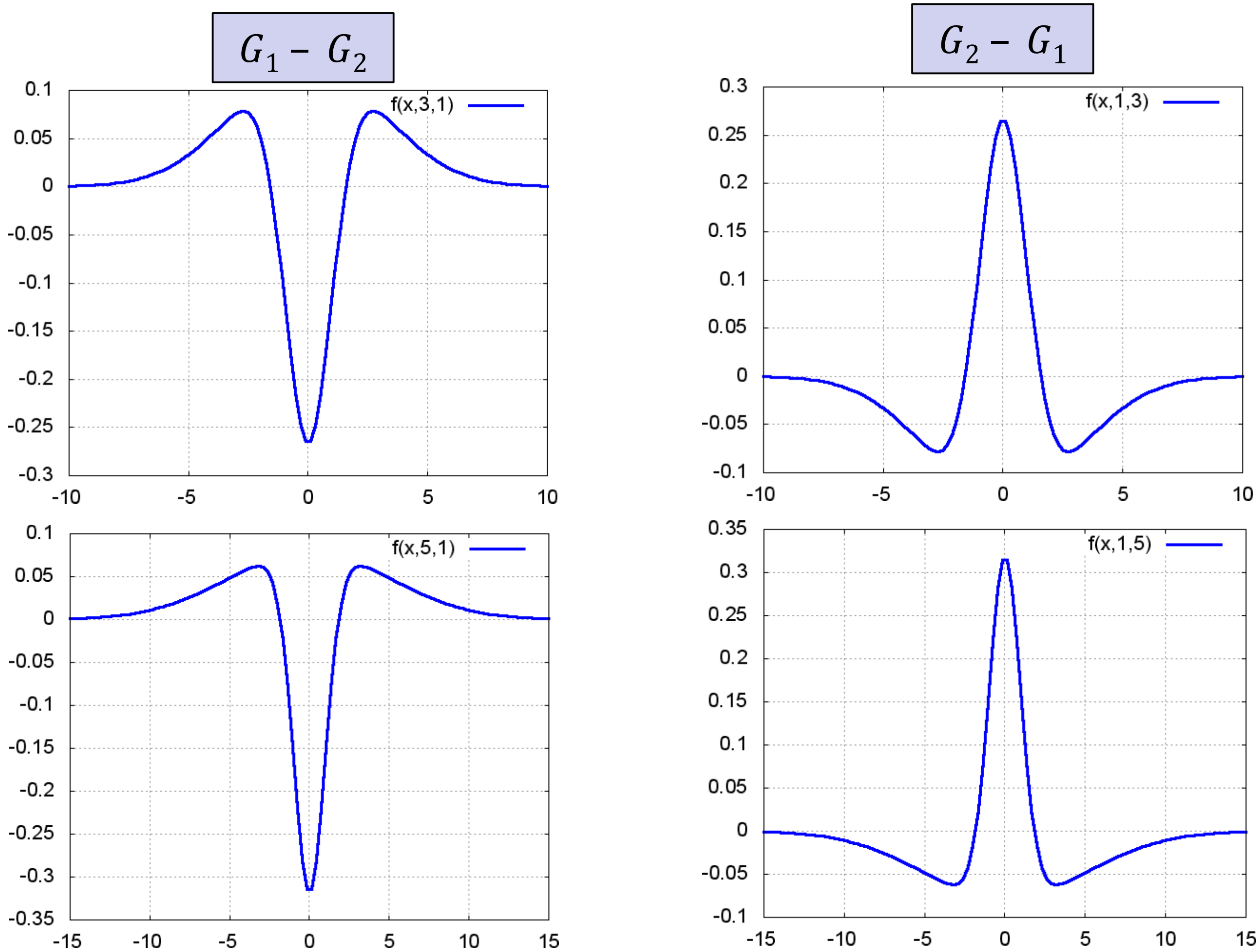
The Difference of Gaussian filter (DoG) is obtained by subtracting two Gaussians of different sizes:





$$f(x, \sigma_1, \sigma_2) = \frac{1}{\sqrt{2\pi}\sigma_1} exp\left(\frac{-x^2}{2\sigma_1^2}\right) - \frac{1}{\sqrt{2\pi}\sigma_2} exp\left(\frac{-x^2}{2\sigma_2^2}\right)$$

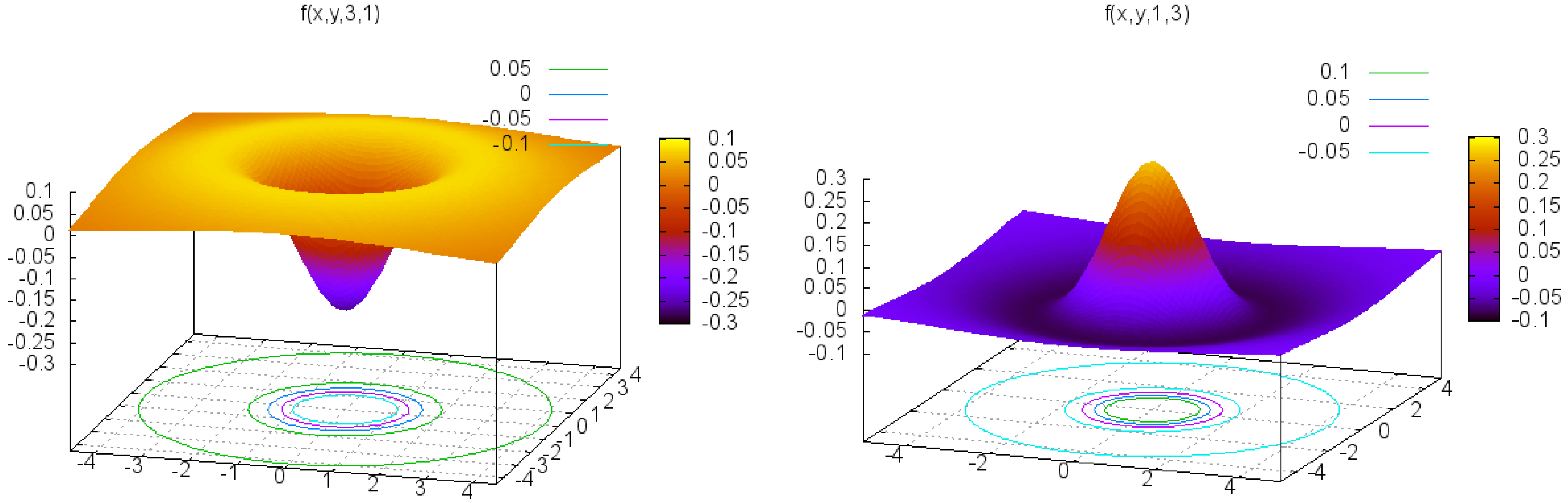
Difference of Gaussians



Note that the x-axis has a larger range in the bottom row. So, although the curve looks narrower, it is actually wider!

Difference of Gaussians

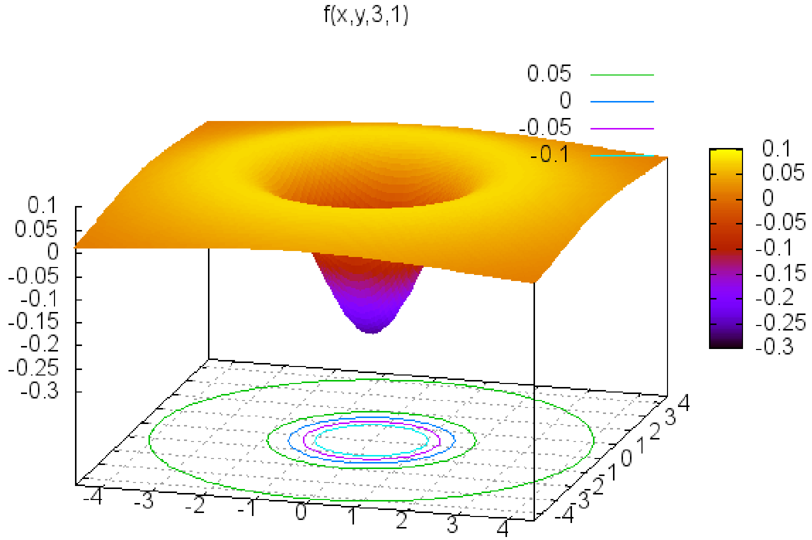
Two-dimensional case:



$$f(x,y,\sigma_1,\sigma_2) = \frac{1}{2\pi\sigma_1^2} exp\left(\frac{-(x^2+y^2)}{2\sigma_1^2}\right) - \frac{1}{2\pi\sigma_2^2} exp\left(\frac{-(x^2+y^2)}{2\sigma_2^2}\right)$$

Difference of Gaussians

Two-dimensional case:



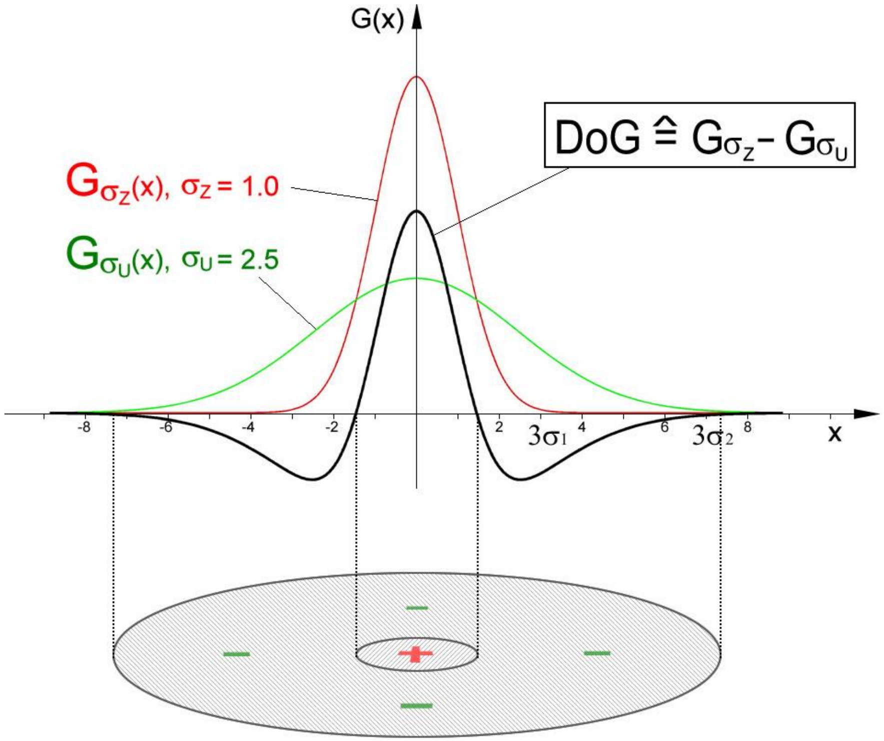
$$g(x,y) = \begin{pmatrix} 1 & 3 & 4 & 4 & 4 & 3 & 1 \\ 3 & 4 & 3 & 0 & 3 & 4 & 3 \\ 4 & 3 & -9 & -17 & -9 & 3 & 4 \\ 4 & 0 & -17 & -30 & -17 & 0 & 4 \\ 4 & 3 & -9 & -17 & -9 & 3 & 4 \\ 3 & 4 & 3 & 0 & 3 & 4 & 3 \\ 1 & 3 & 4 & 4 & 4 & 3 & 1 \end{pmatrix}$$

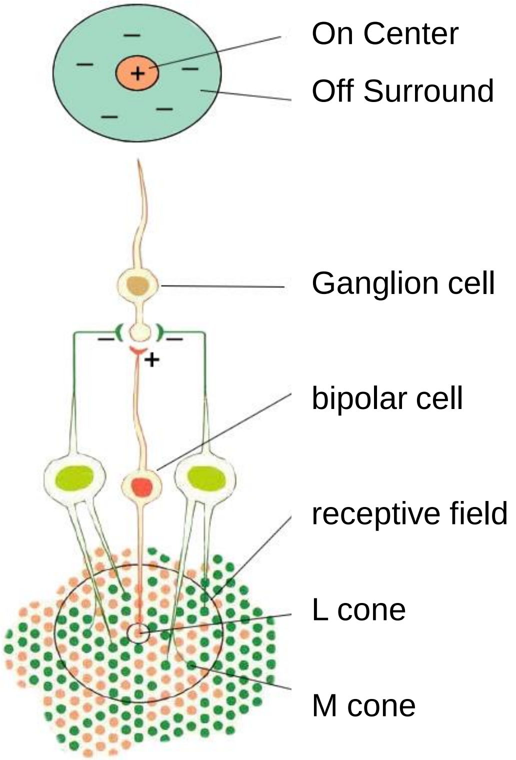
A DoG filter mask

$$f(x,y,\sigma_1,\sigma_2) = \frac{1}{2\pi\sigma_1^2} exp\left(\frac{-(x^2+y^2)}{2\sigma_1^2}\right) - \frac{1}{2\pi\sigma_2^2} exp\left(\frac{-(x^2+y^2)}{2\sigma_2^2}\right)$$

Difference of Gaussians

DoG filters simulate the processing of retinal ganglion cells in the human visual system (with $\sigma_2 \sim 5 * \sigma_1$)





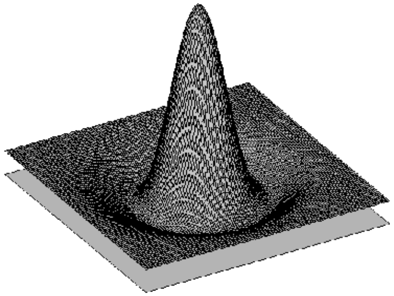
[Kaim 2006]

Difference of Gaussians

Instead of applying one large DoG filter, you can also (due to linearity):

- Smooth an image twice with different Gaussians G_1 and G_2
- Subtract resulting images

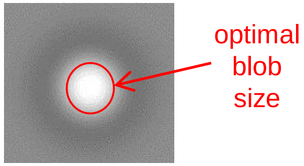
$$I \times (G_1 - G_2) = (I \times G_1) - (I \times G_2)$$



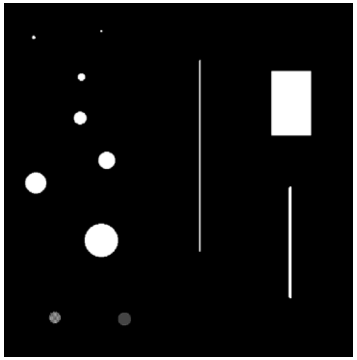
(This will be the basis for Laplacian pyramids!)

Difference of Gaussians

DoG filters respond best if the blob fits the filter size.



Filter



original image

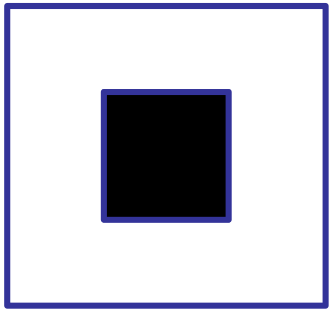
DoG responses with different filter sizes

Try it yourself at: http://matlabserver.cs.rug.nl
(on this page: [Centre-Surround cell (DoG) operator and Dot-pattern selective cell operator](#))

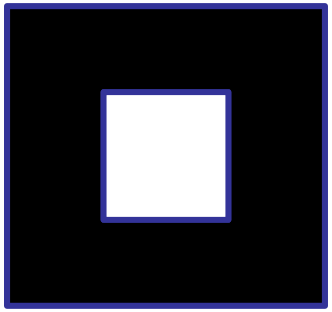
Difference of Gaussians

A (very) simple approximation to the DoG (and LoG):

- The DoB filter (Difference of Boxes) (or just center-surround filter):
- Computes difference between two mean filters of different sizes



surround – center

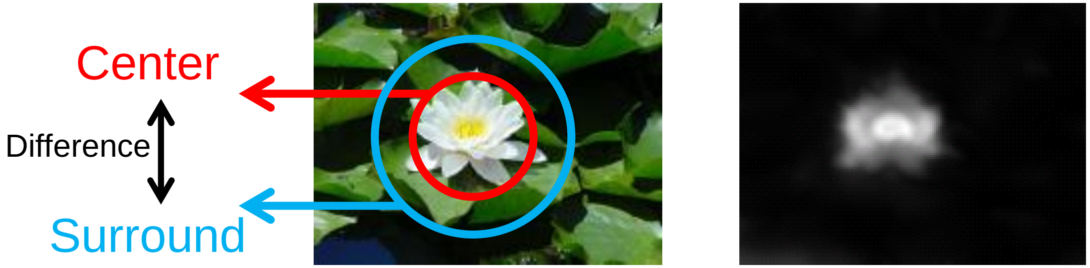


center – surround

Application: Saliency Detection

Difference-of-Gaussian filters are the core method in the VOCUS2 system for saliency detection [Frintrop et al. 2015]:

A *salient region* attracts human attention. VOCUS2 shows the saliency of image regions in a saliency map

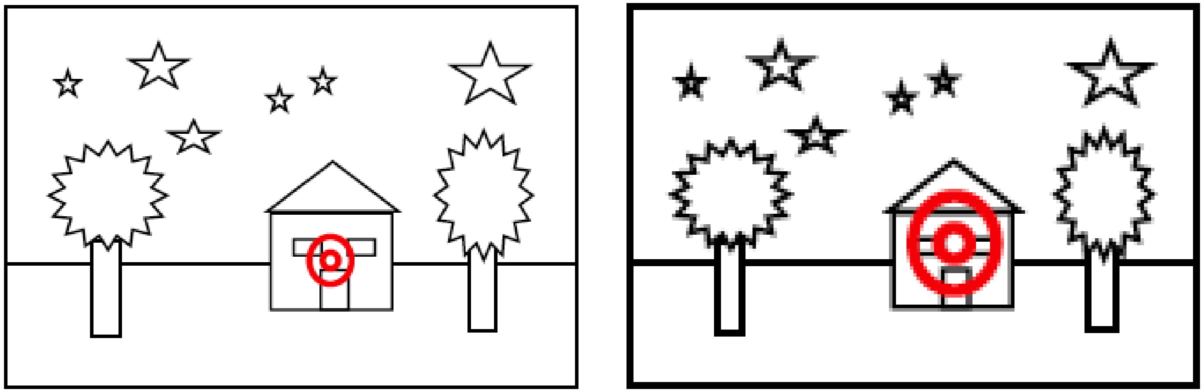


Saliency map

More in lecture “Computer Vision 2”, SS 2019

Image Pyramids

- Filters respond to items of different sizes
- For objects/patterns of different sizes, we could apply filters of many different sizes:

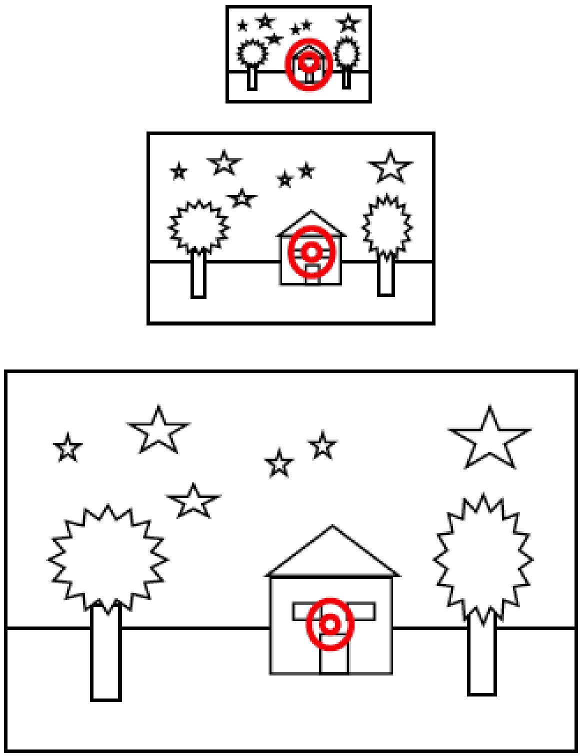


- But: Large filters are computationally expensive

[Images: Irani & Basri]

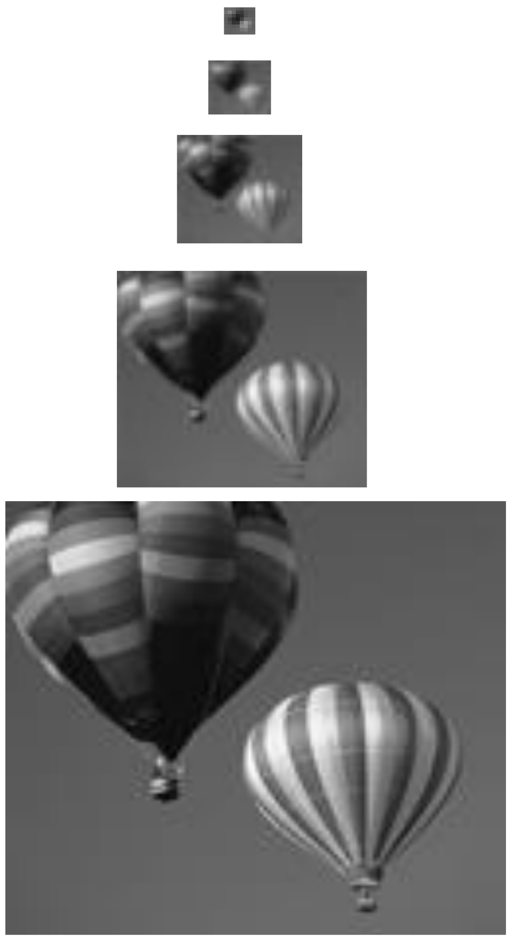
Image Pyramids

Idea: do not take a larger filter, but a smaller image:



[Images: Irani & Basri]

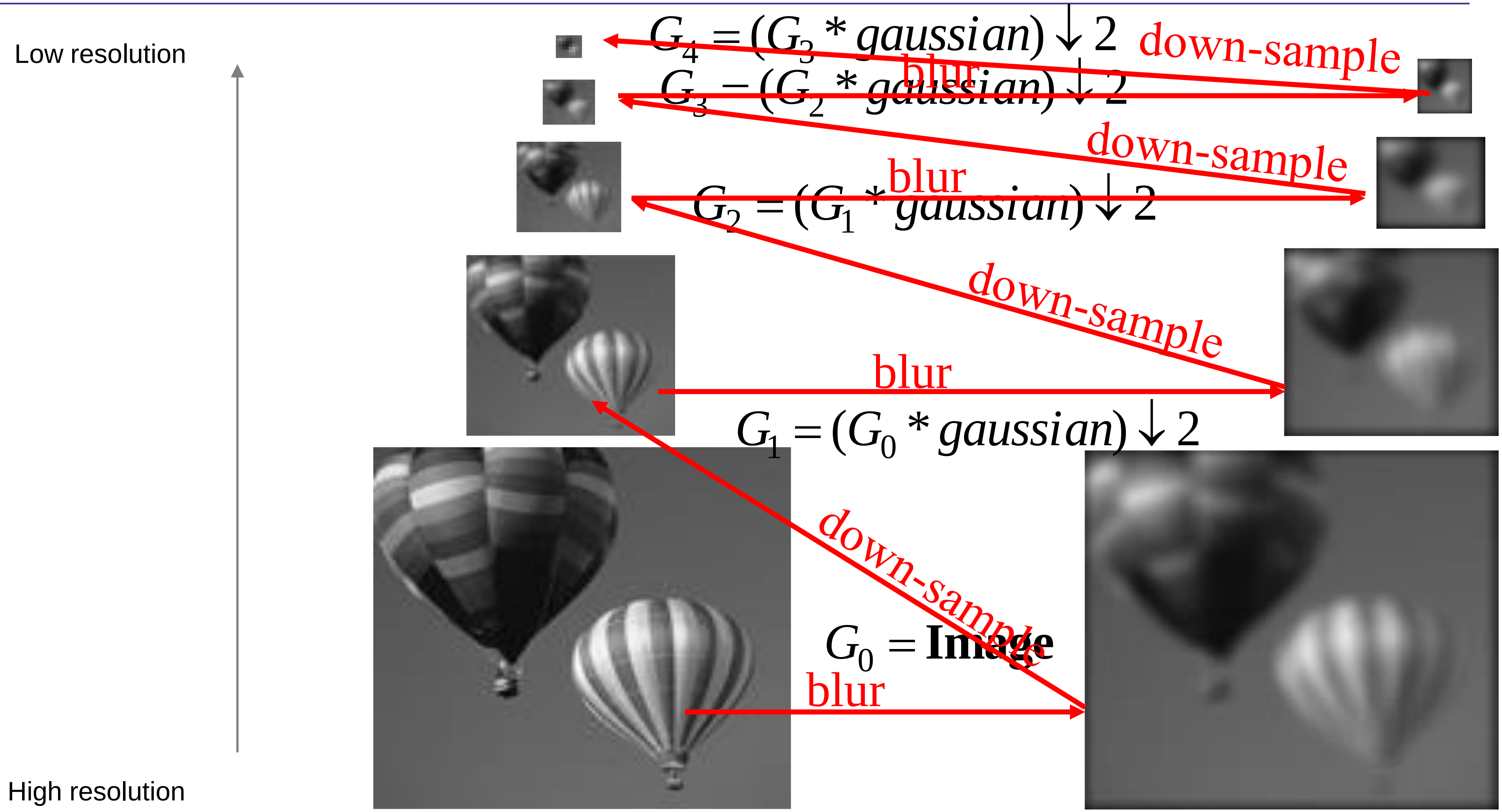
Image Pyramids



- An image pyramid is a scale space representation
- It contains n layers (scales)
- Layer n has usually half the size in each dimension as layer $n-1$ and is obtained by subsampling (take each second pixel)
- There are low-pass pyramids and band-pass pyramids
- The most famous low-pass pyramid is the **Gaussian pyramid**. It smooths each image with a Gaussian before subsampling.

[Images: Irani & Basri]

The Gaussian Pyramid



[Images: Irani & Basri]

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

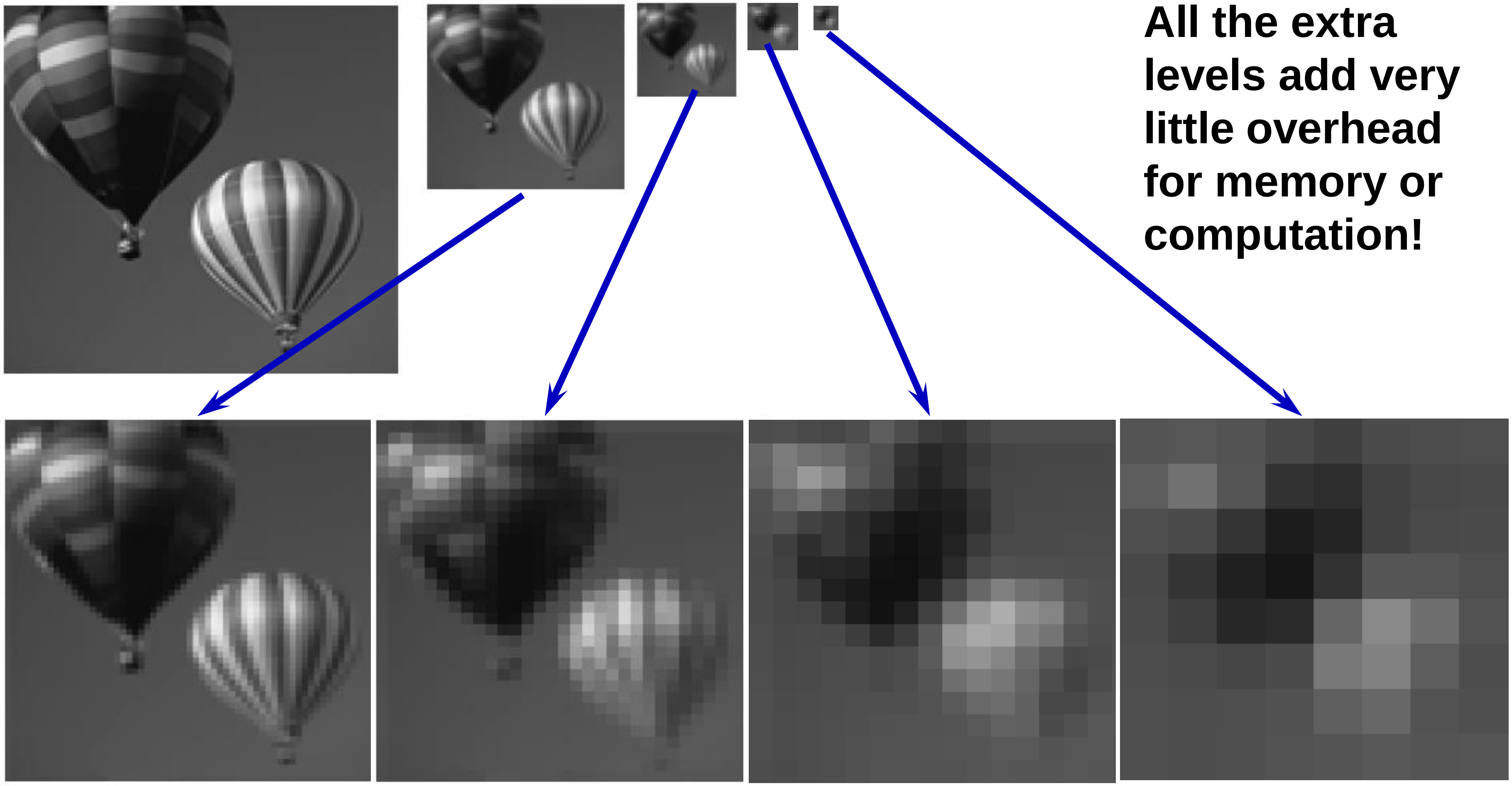
Another example of aliasing:

a b c

FIGURE 4.17 Illustration of aliasing on resampled images. (a) A digital image with negligible visual aliasing. (b) Result of resizing the image to 50% of its original size by pixel deletion. Aliasing is clearly visible. (c) Result of blurring the image in (a) with a 3×3 averaging filter prior to resizing. The image is slightly more blurred than (b), but aliasing is not longer objectionable. (Original image courtesy of the Signal Compression Laboratory, University of California, Santa Barbara.)

[Gonzalez/Woods]

Gaussian Pyramid – Stored Information

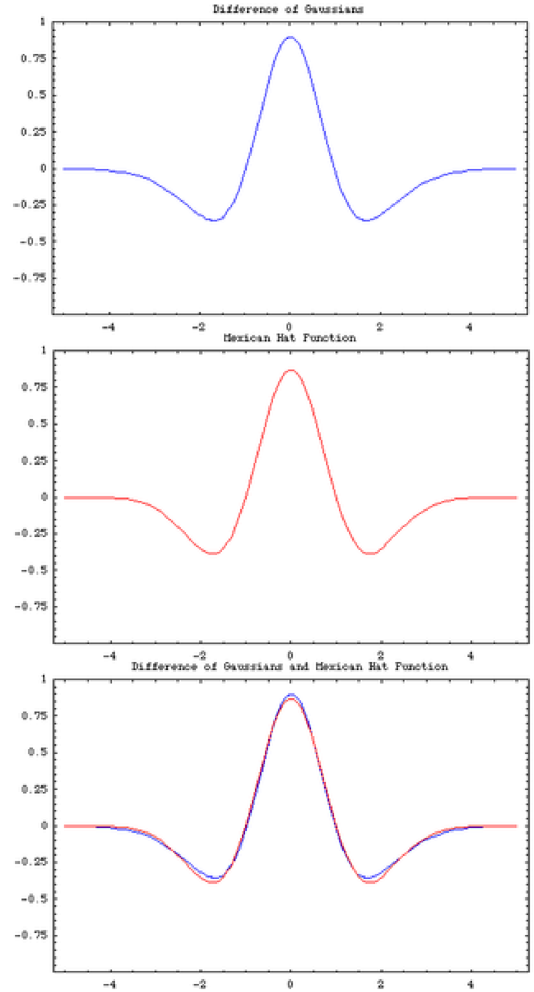


All the extra levels add very little overhead for memory or computation!

[Images: Irani & Basri]

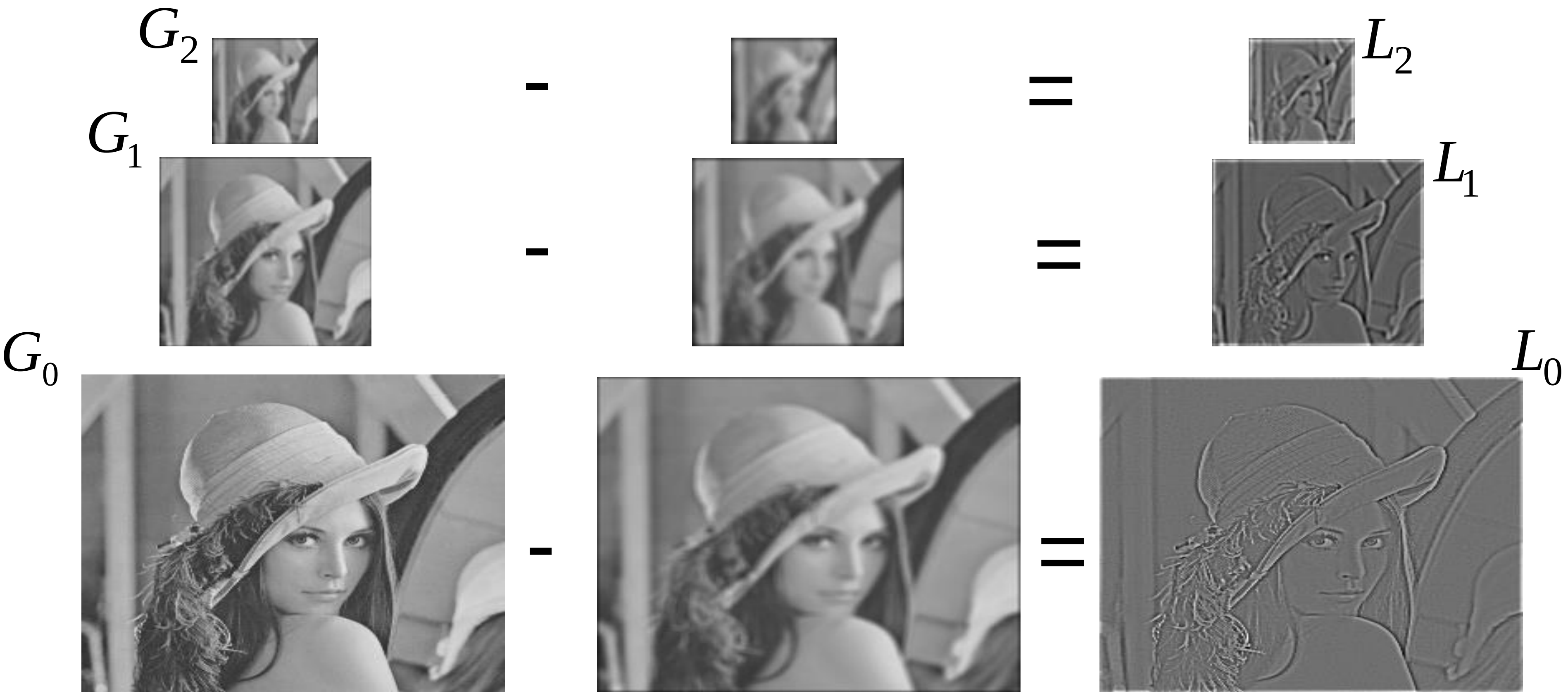
The Laplacian Pyramid

- The **Laplacian pyramid** is a band-pass pyramid
- It stores edge information of an image on different scales
- It is computed by subtracting adjacent layers of a Gaussian pyramid
- That means, it computes the difference of Gaussians
- It is called Laplacian Pyramid since the DoG function approximates the Laplacian-of-Gaussian function (see lecture on Edge Detection)



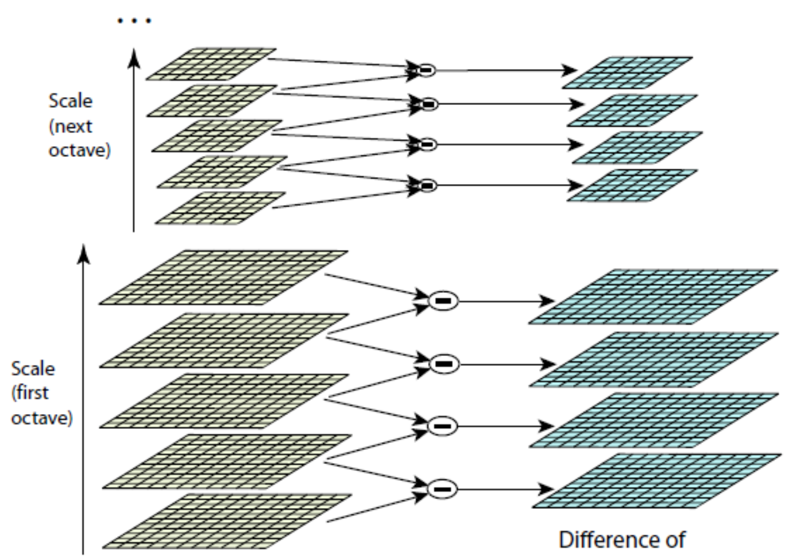
The Laplacian Pyramid

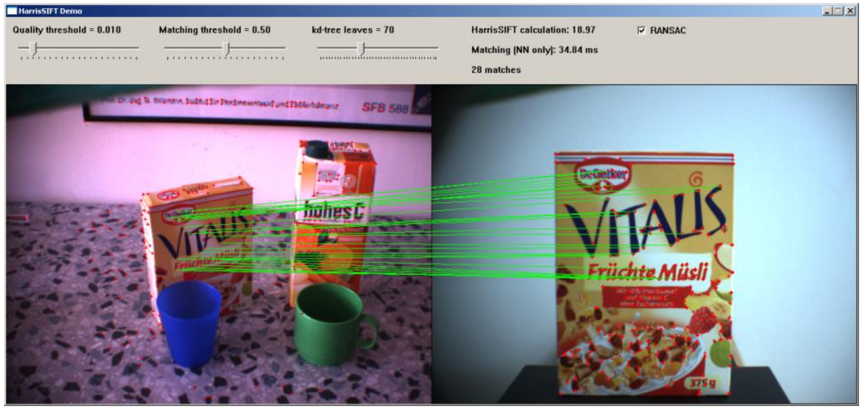
Gaussian Pyramid | Smoothed | Laplacian Pyramid



Application: Keypoint Detection for Object Recognition

Keypoints are detected in scale space (pyramids) to enable recognition of different object sizes (or the same object from different distances)





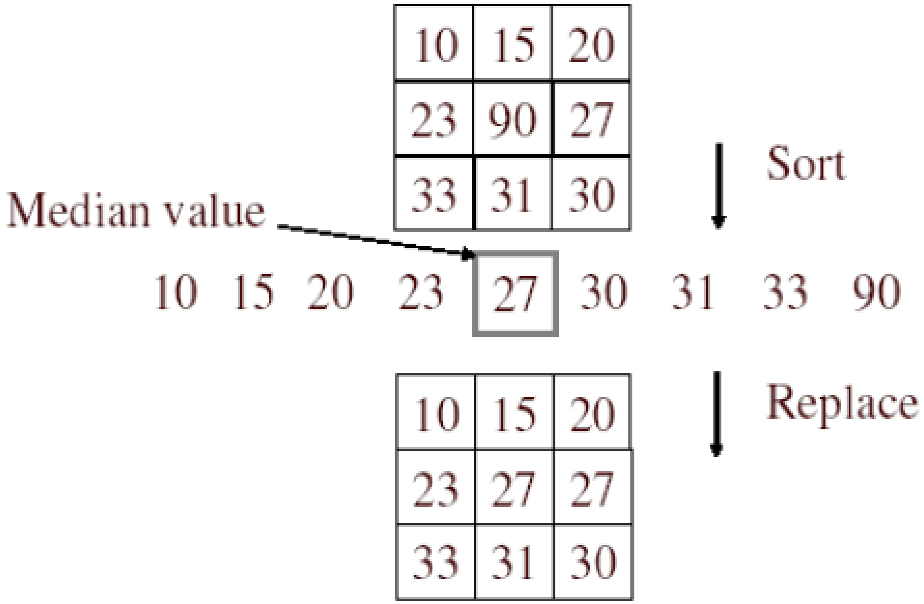
(see lecture on Features)

[Lowe 2004 & http://ivt.sourceforge.net/examples.html]

Non-linear operators

Typical non-linear operators:

- Median filter (replace each pixel by the median of its neighbors)

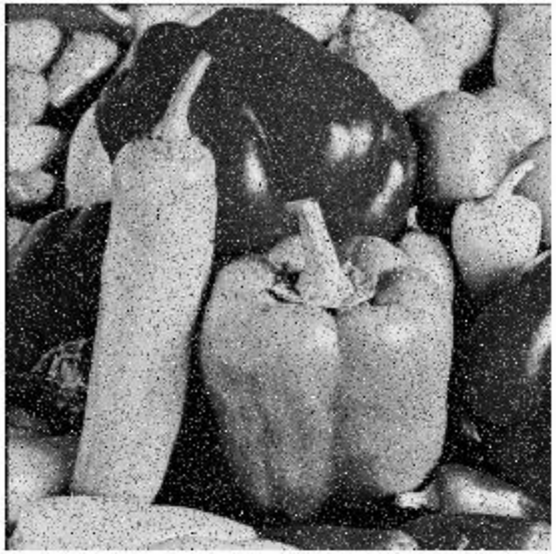


- Max filter (equivalently)
- Min filter (equivalently)
- Proof for non-linearity: see exercise

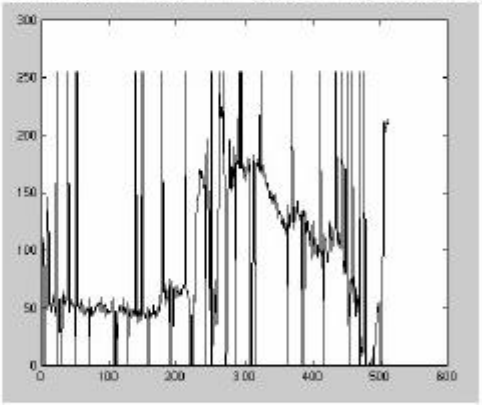
[Image: Kristen Grauman]

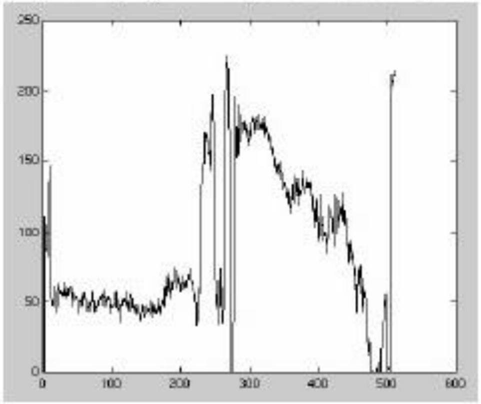
Median Filter

Salt and pepper noise



Median filtered



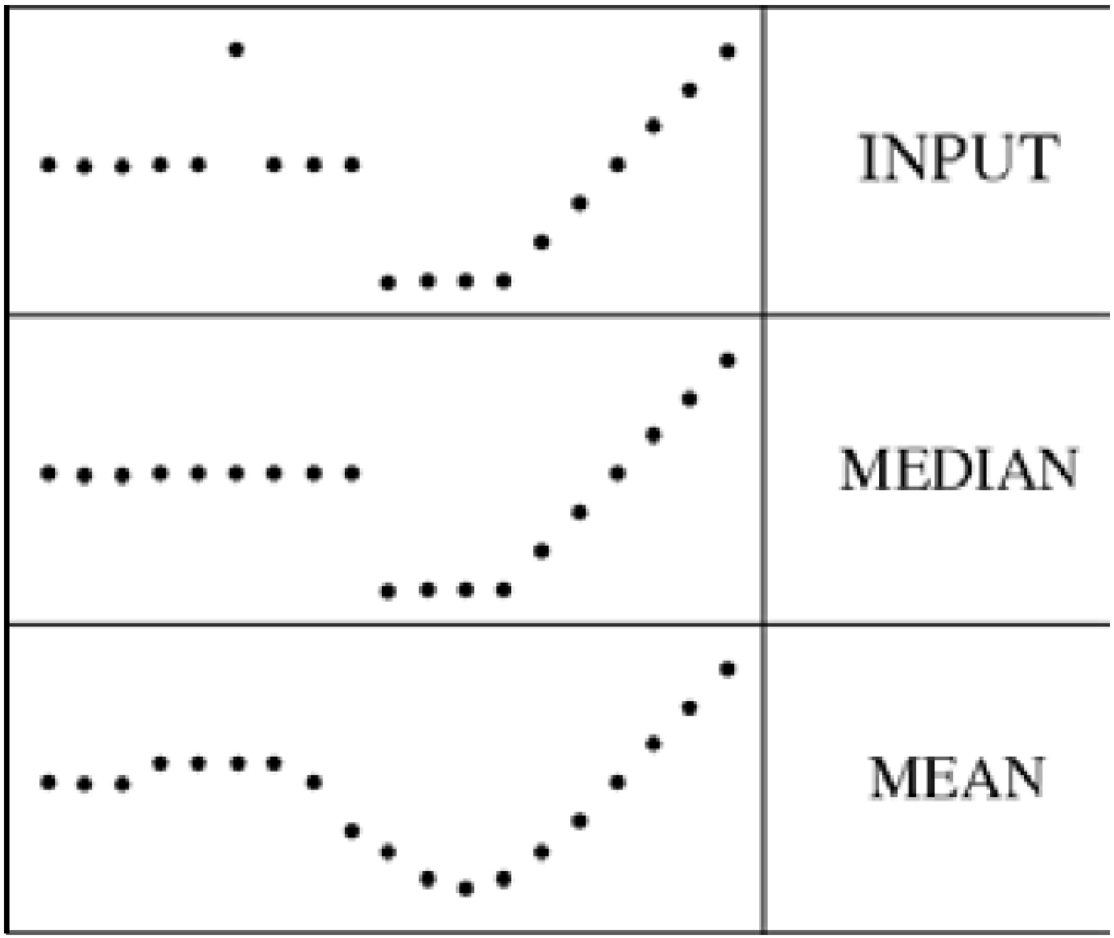


Plots of a row of the image

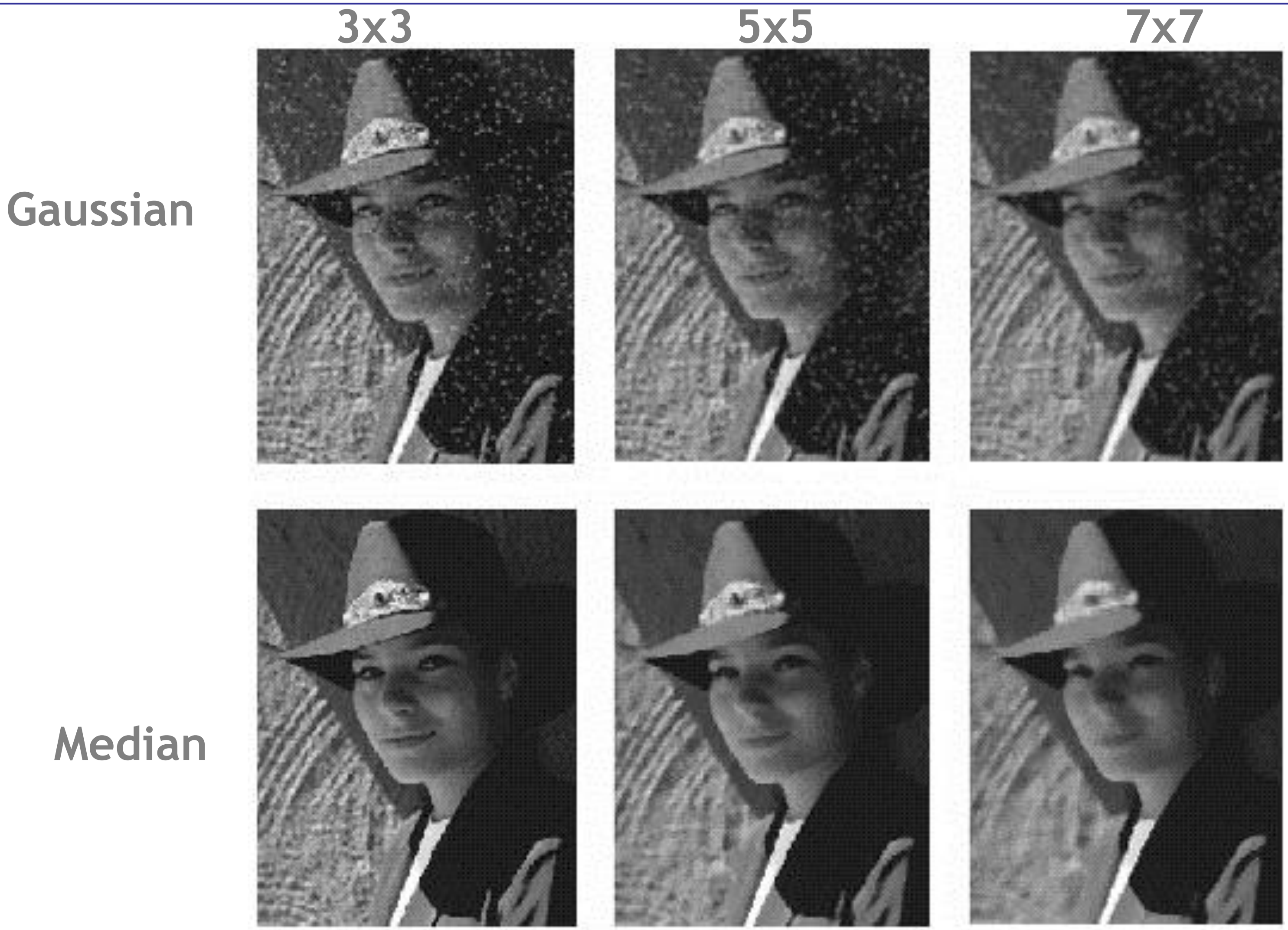
[Image: Martial Hebert]

Median Filter

The Median filter is edge preserving.



Median vs. Gaussian Filtering



Summary

- To extract visual information, we use filters.
- Filters can be applied in the spatial domain or in the frequency domain.
- Spatial filters consist of a neighborhood and an operation.
- Linear spatial filters are usually applied with the operations convolution or correlation.
- The Gaussian filter is the most common filter for smoothing.
- The Difference-of-Gaussian filter can be obtained by subtracting two Gaussian filters.
- Some filters are separable, which enables much faster computations.
- Image pyramids enable processing on different scales.
- Non-linear filters are for example the median, max, or min filter.

Primary Literature

- Szeliski: parts from chapters 3.2, 3.3, and 3.5
- Gonzalez/Woods: parts from chapters 3.4 and 3.5

Secondary Literature

- Forsyth/Ponce: “Computer Vision – A modern approach”, Prentice Hall, 2003
- Smith, Steven W. “The scientist and engineer's guide to digital signal processing.” (1997).
- Lowe, David G. “Distinctive image features from scale-invariant keypoints.” *International journal of computer vision* 60.2 (2004): 91-110.
- Simone Frintrop, Thomas Werner, and Germán Martín García: Traditional Saliency Reloaded: A Good Old Model in New Shape, *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), Boston, June 2015*