Computer Vision

-Spatial and Frequency Domain Processing-

Oliver Bimber

Course Schedule

Type	Date	Time	Room	Topic	Comment
Lec1	11.10.2022	12:00-13:30	H1	Introduction and Course Overview	
Lab1	10./11./12./13.10.2022	17:15-18:45	S3055	Introduction to Python	
Lec2	18.10.2022	12:00-13:30	HS 1	Spatial and Frequency Domain Processing	
Lab2	17./18./19./20.10.2022	17:15-18:45	S3055	Introduction to IP/CV Modules	
Lec3	25.10.2022	12:00-13:20	HS 1	Gradient Domain Processing	National Holiday (26.10.)
Lec4	08.11.2022	12:00-13:30	HS 10	Segmentation and Local Features	Allerheiligen (2.11.)
Lab3	07./08./08./10.11.2022	17:15-18:45	S3055	Project Introduction	
Lec5	15.11.2022	12:00-13:30	HS 1	Basics of Cameras	
Lec6	22.11.2022	12:00-13:30	HS 1	Geometric Camera Calibration	
Lab4	21./22./23./24.11.2022	17:15-18:45	S3055	Project Basics and Related Work	
Lec7	29.11.2022	12:00-13:30	HS 1	The Geometry of Multiple Views	
Lec8	06.11.2022	12:00-13:30	HS 1	Stereoscopic Depth Estimation	Mariä Empfängnis (8.12.)
Lec9	13.12.2022	12:00-13:30	HS 1	Range Scanning and Data Processing	
Lab5	12./13./14./15.12.2022	17:15-18:45	S3055	Presentation of Initial Ideas	Christmas Break
Lec10	10.01.2023	12:00-13:30	HS 1	Structure from Motion	
Lab6	09./10./11./12.01.2023	17:15-18:45	S3055	Presentation of Intermediate Results and Final Concepts	
Lec11	17.01.2023	12:00-13:30	HS 1	Computational Imaging	
Lec12	24.01.2023	12:00-13:30	HS 1	Recap and Q&A	
Lab7	23./24./25./26.01.2023	17:15-18:45	S3055	Final Project Presentations	
Ex1	31.01.2023	12:00-13:30	HS 1	Exam (Hauptklausur)	
Ex2	28.02.2023	15:30-17:00	TBA	Retry Exam (Nachklausur)	

Image Analysis

- One goal of image analysis is to identify distinct image features
- What are good features?

Image Analysis

- One goal of image analysis is to identify distinct image features
- What are good features?
    - for example, those image points that describe main characteristics of image content, such as edges or corners

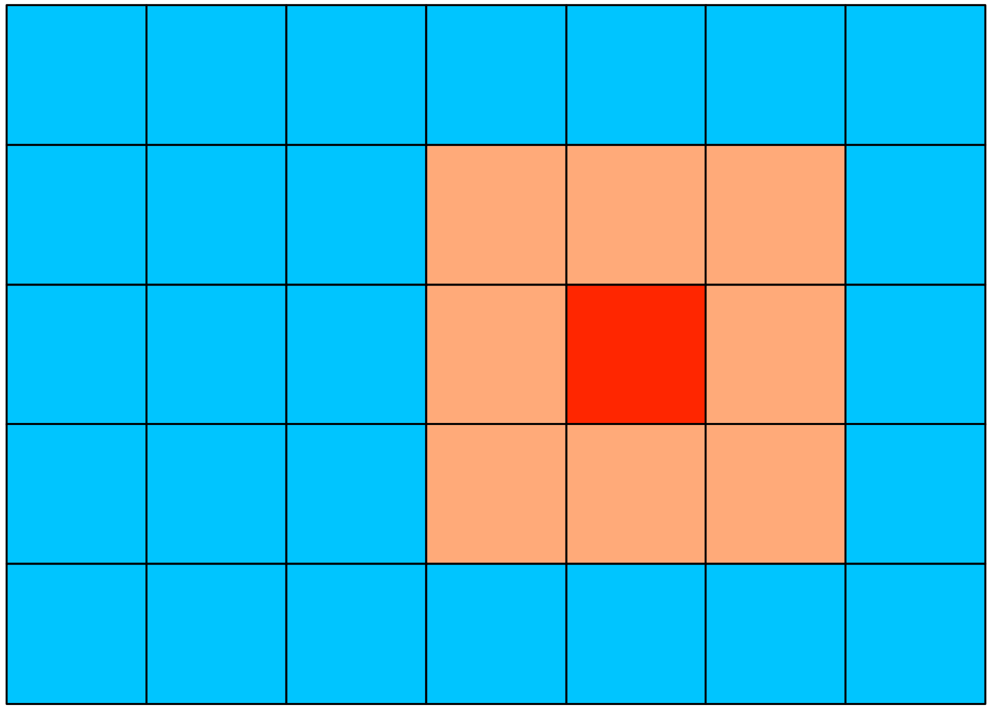
Image Analysis

- One goal of image analysis is to identify distinct image features
- What are good features?
  - for example, those image points that describe main characteristics of image content, such as edges or corners
  - but features should also be robust (invariant to rotation, translation, scaling, lighting, etc.), since we want to find the same features if the same content is captured under different conditions
- How do we compute (simple) features?

Linear Filtering

- Which kind of filtering operation is this?

$$R_{i,j} = \frac{1}{(2k+1)^2} \sum_{u=i-k}^{u=i+k} \sum_{v=j-k}^{v=j+k} F_{u,v}$$

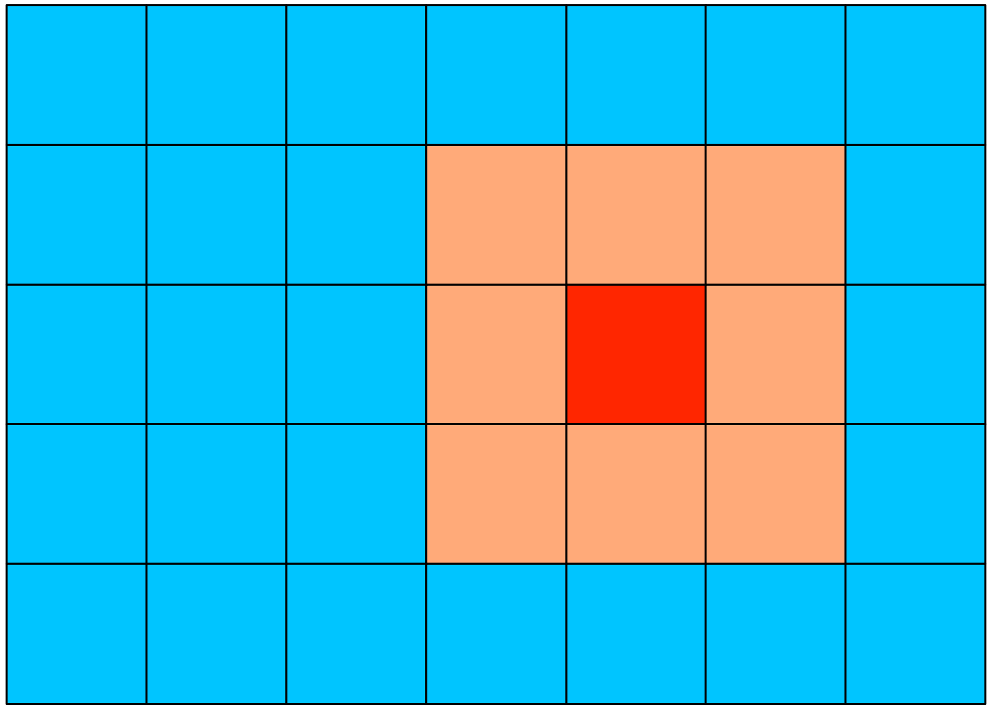


linear filtering

Linear Filtering

- Which kind of filtering operation is this?
  - local average
- Image filtering:
  - construct new image R of same size as original image F
  - fill each location in R with a weighted sum of the pixel values of corresponding location's neighbourhood in F
  - use the same weights each time
  - different sets of weights represent different filters
  - shift-invariant: filtered result depends on image neighbourhood – but not on position of neighbourhood
  - linear: filtering the sum of two images is equivalent as summing the two filtered images
- Shift invariant linear system (SILS)

$$R_{i,j} = \frac{1}{(2k+1)^2} \sum_{u=i-k}^{u=i+k} \sum_{v=j-k}^{v=j+k} F_{u,v}$$



linear filtering

Convolution

- The pattern of weights is usually referred to as kernel H
- The process of applying H to an image F is known as convolution
- Say: “The convolution of F with H results in R”
- The kernel H can be defined as a continuous function or a finite matrix of discrete weights
- What is the kernel for our local average example?

$$R_{i,j} = \frac{1}{(2k+1)^2} \sum_{u=i-k}^{u=i+k} \sum_{v=j-k}^{v=j+k} F_{u,v}$$

$$R_{i,j} = \sum_{u=i-k}^{u=i+k} \sum_{v=j-k}^{v=j+k} H_{i-u,j-v} \cdot F_{u,v}$$

$$R = H \otimes F$$

Convolution

- The pattern of weights is usually referred to as kernel H
- The process of applying H to an image F is known as convolution
- Say: “The convolution of F with H results in R”
- The kernel H can be defined as a continuous function or a finite matrix of discrete weights
- What is the kernel for our local average example?
- How large is it?

$$R_{i,j} = \frac{1}{(2k+1)^2} \sum_{u=i-k}^{u=i+k} \sum_{v=j-k}^{v=j+k} F_{u,v}$$

$$R_{i,j} = \sum_{u=i-k}^{u=i+k} \sum_{v=j-k}^{v=j+k} H_{i-u,j-v} \cdot F_{u,v}$$

$$R = H \otimes F$$

$$H_{i,j} = \frac{1}{(2k+1)^2}$$

Convolution

- The pattern of weights is usually referred to as kernel H
- The process of applying H to an image F is known as convolution
- Say: "The convolution of F with H results in R"
- The kernel H can be defined as a continuous function or a finite matrix of discrete weights
- What is the kernel for our local average example?
- How large is it?
    - $(2k+1)^2$ weights

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

"box filter"

$$R_{i,j} = \frac{1}{(2k+1)^2} \sum_{u=i-k}^{u=i+k} \sum_{v=j-k}^{v=j+k} F_{u,v}$$

$$R_{i,j} = \sum_{u=i-k}^{u=i+k} \sum_{v=j-k}^{v=j+k} H_{i-u,j-v} \cdot F_{u,v}$$

$$R = H \otimes F$$

$$H_{i,j} = \frac{1}{(2k+1)^2}$$

Example

Original

0	0	0
0	1	0
0	0	0

?

Example: Identity

Original

0	0	0
0	1	0
0	0	0

?

Original

0	0	0
0	1	0
0	0	0

Filtered
(no change)

Example

Original

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

?

Example: Local Average

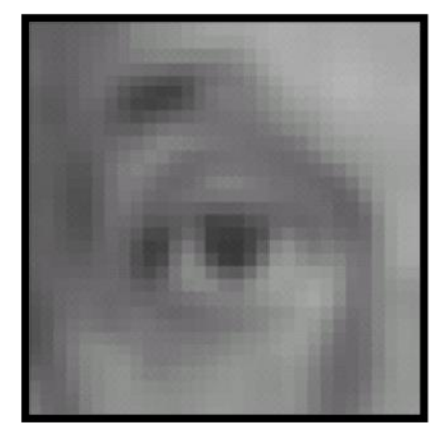
Original

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

?

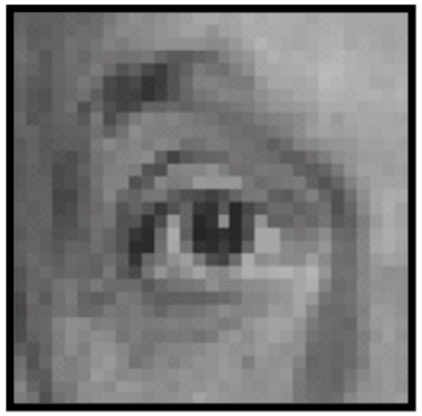
Original

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



Blur (with a box filter)

Example

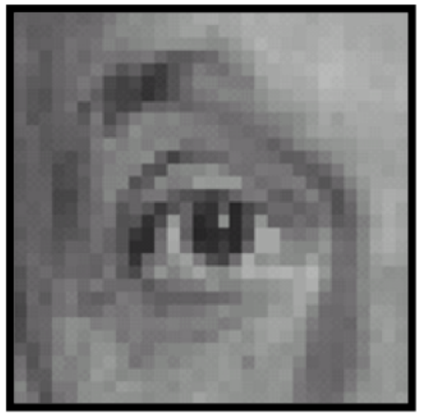


Original

$$\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} - \frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \quad ?$$

(Note that filter sums to 1)

Example: Sharpening



Original

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

(Note that filter sums to 1)

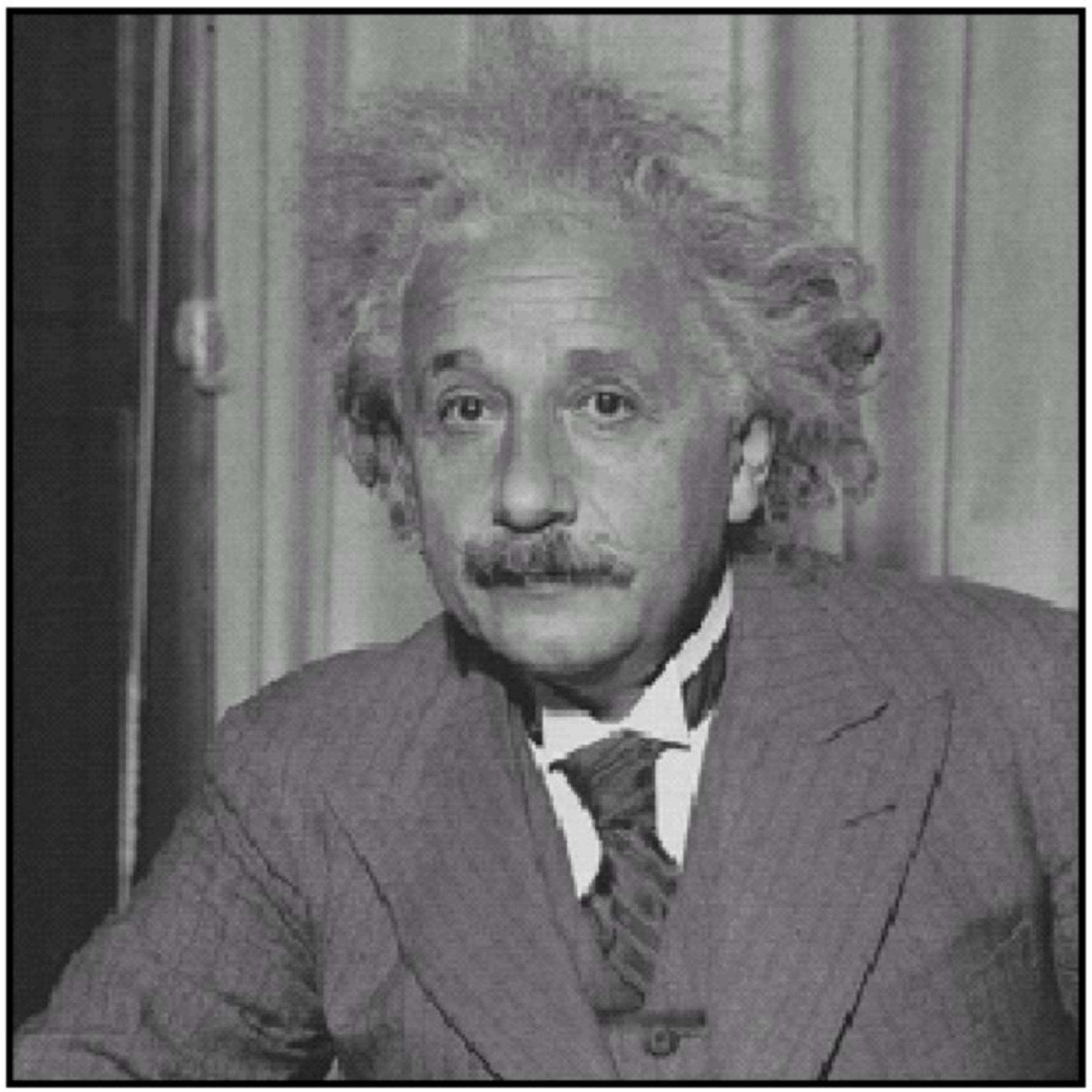
Original

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

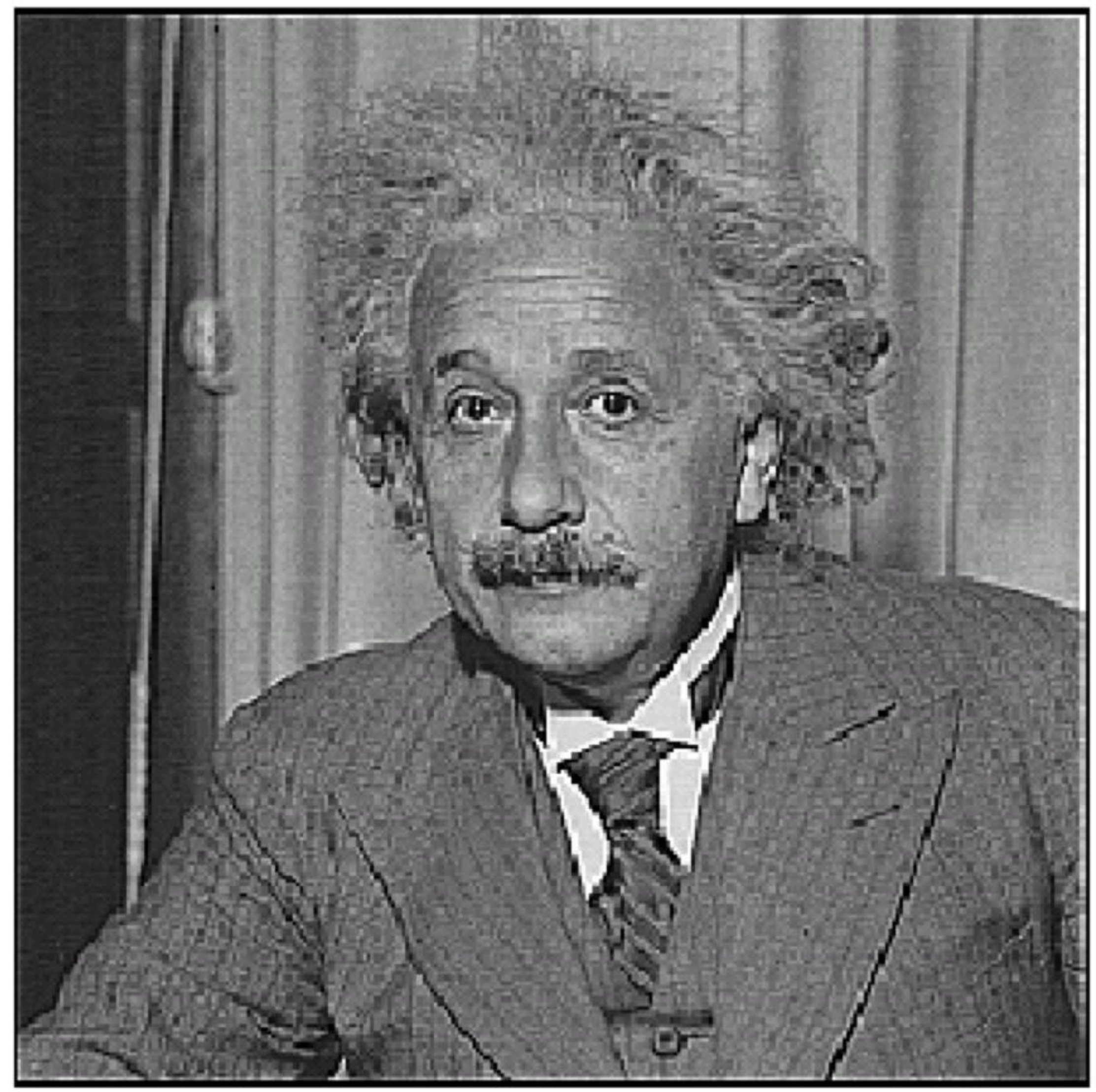
Sharpening filter

- Accentuates differences with local average

Example: Sharpening



before



after

Example: Classification

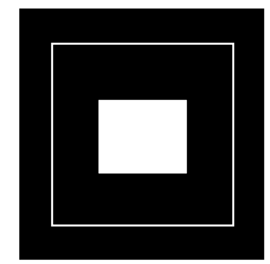
* ? =

Example: Classification

Example: Smoothening

- Convolving an average kernel with an image is known as smoothening or blurring
- Our local average kernel is a smoothening operator
- But it is quite unrealistic (its output does not look like the one of a defocused camera)
- Why?

$$R = H \otimes F$$

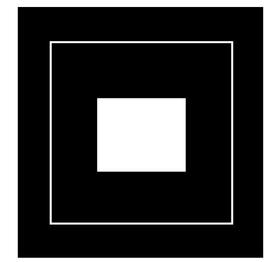


$$H_{i,j} = \frac{1}{(2k+1)^2}$$

Example: Smoothening

- Convolving an average kernel with an image is known as smoothening or blurring
- Our local average kernel is a smoothening operator
- But it is quite unrealistic (its output does not look like the one of a defocused camera)
- Why?
  - the intensity spread (point spread function - PSF) of a defocused point is not spatially constant, but follows a Gaussian distribution
- The standard deviation sigma σ controls the influence of the neighbours (small σ - neighbours have small weights)

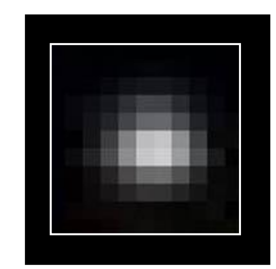
$$R = H \otimes F$$



$$H_{i,j} = \frac{1}{(2k+1)^2}$$

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

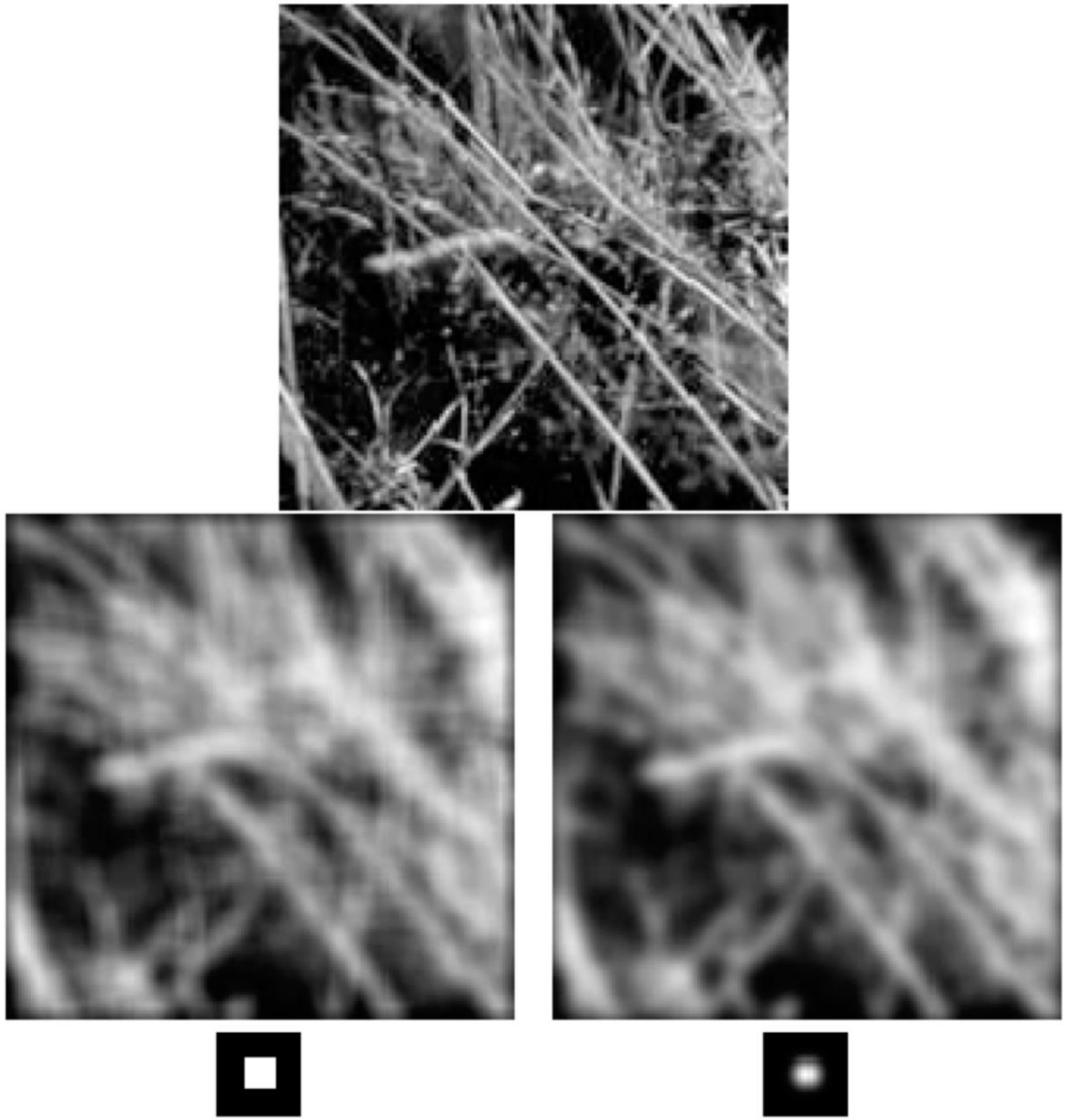
normalization
(sum of weights equals 1)



$$H_{i,j} =$$

$$\frac{1}{2\pi\sigma^2} \exp\left(-\frac{((i-k-1)^2+(j-k-1)^2)}{2\sigma^2} \right)$$

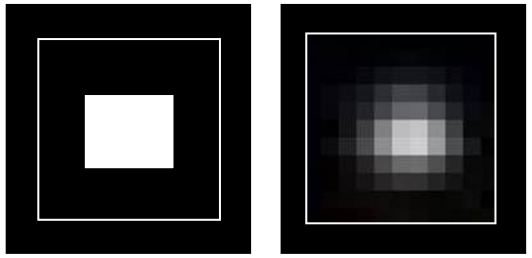
Example: Smoothening

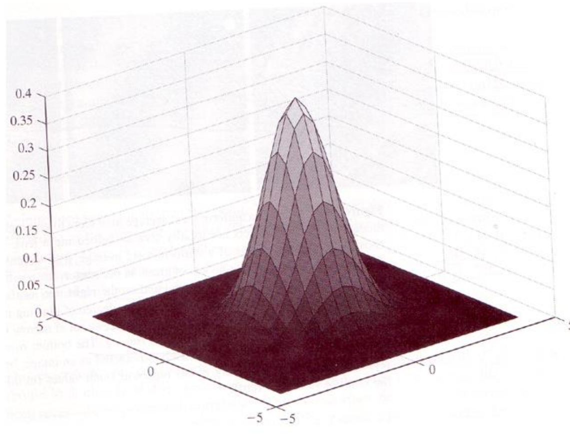


Example: Smoothening

- Convolving an average kernel with an image is known as smoothening or blurring
- Our local average kernel is a smoothening operator
- But it is quite unrealistic (its output does not look like the one of a defocused camera)
- Why?
  - the intensity spread (point spread function - PSF) of a defocused point is not spatially constant, but follows a Gaussian distribution
- The standard deviation sigma σ controls the influence of the neighbours (small σ - neighbours have small weights)

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

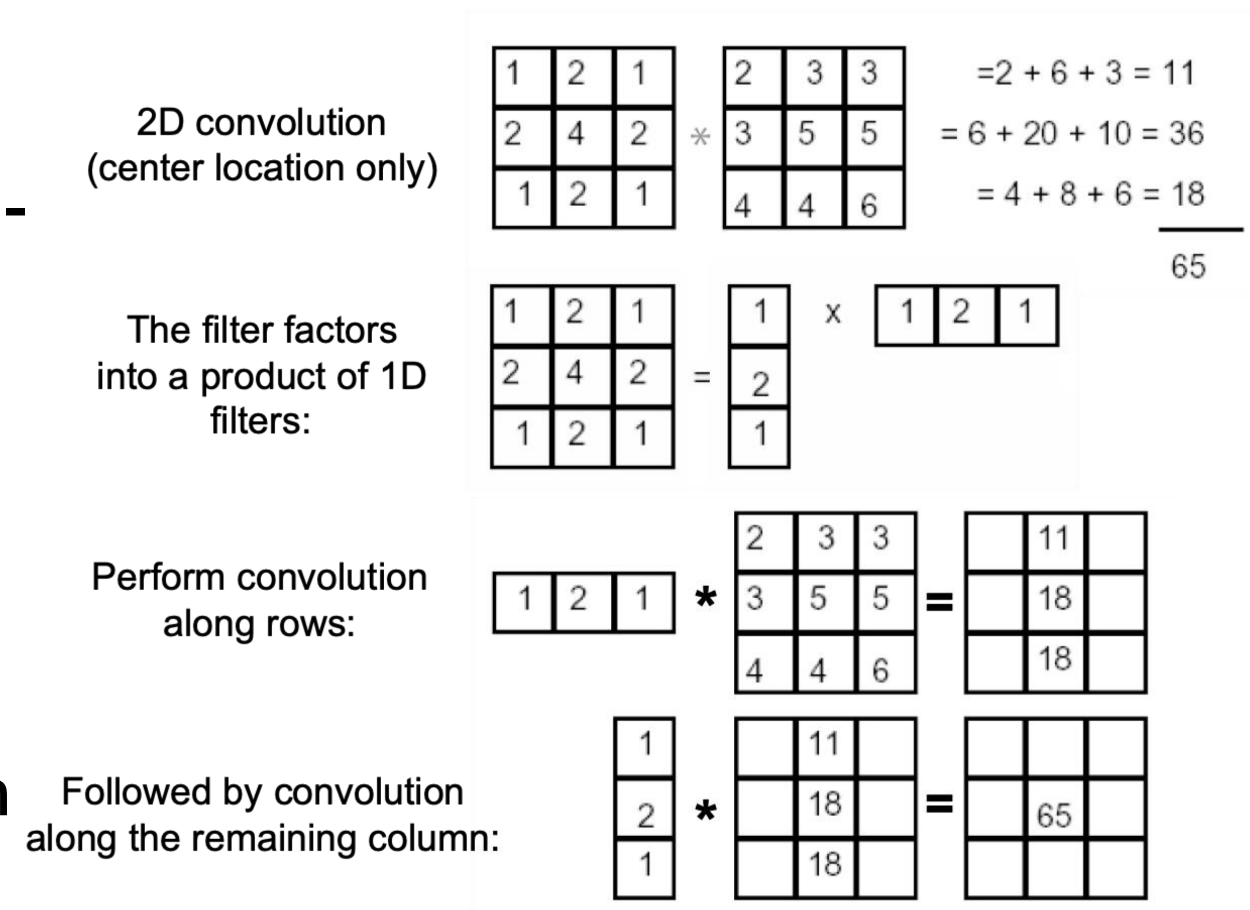




smoothening reduces noise

Properties of Gaussians

- Remove “high-frequency” components from the image (low-pass filter)
- Convolution with itself is another Gaussian
  - one can smoothen with small-width kernel multiple times, and get same result as smoothening with larger-width kernel
  - 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



Convolution Rules for SILSs

- Superimposition: the sum of two filtered images is equivalent to the filtered image with the sum of the two kernels (component-wise addition)

$$H_1 \otimes R + H_2 \otimes R = (H_1 + H_2) \otimes R$$

- Scaling: scaling of a filtered image is equivalent to filtering the image with a scaled kernel

$$(kH) \otimes R = k(H \otimes R)$$

- Symmetric: the convolution of two filters to an image is symmetric

$$H_1 \otimes (H_2 \otimes R) = H_2 \otimes (H_1 \otimes R)$$

- Associative: convolution is associative, which means that we can find a single kernel that behaves like the composition of two kernels

$$H_1 \otimes (H_2 \otimes R) = (H_1 \otimes H_2) \otimes R$$

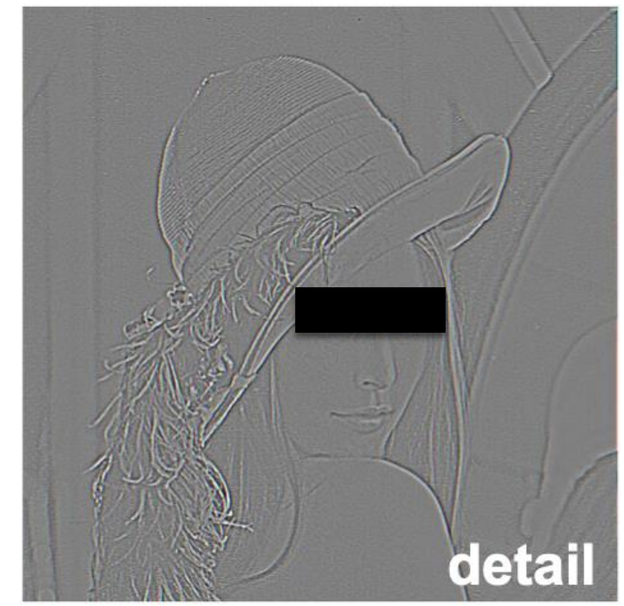
Example: Unsharp Masking

https://en.wikipedia.org/wiki/Lenna
https://de.wikipedia.org/wiki/Lena_(Testbild)

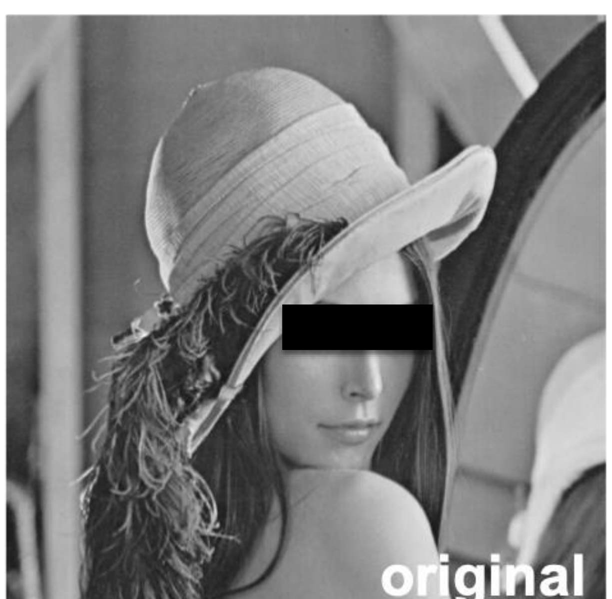
What does blurring take away?

–

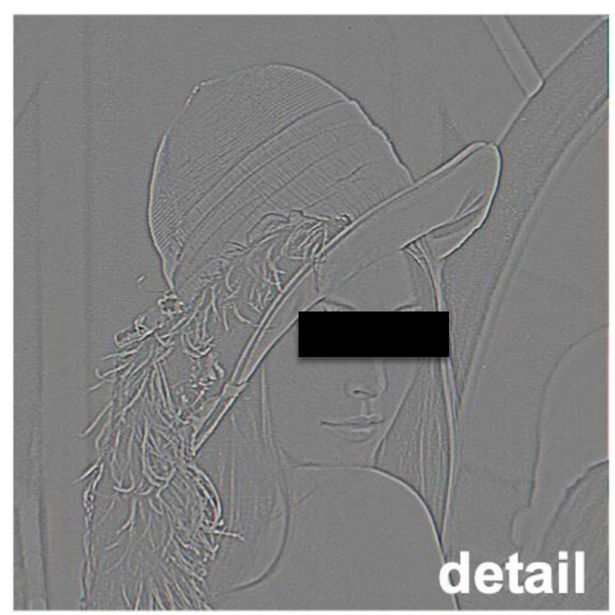
=



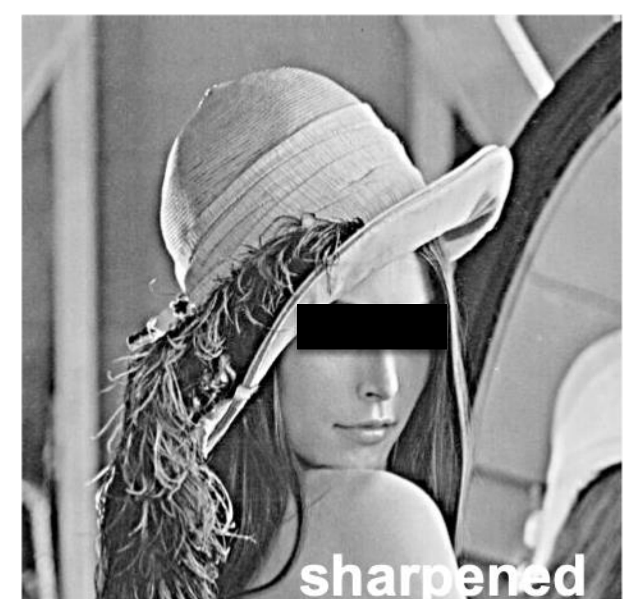
Let's add it back:



+ α



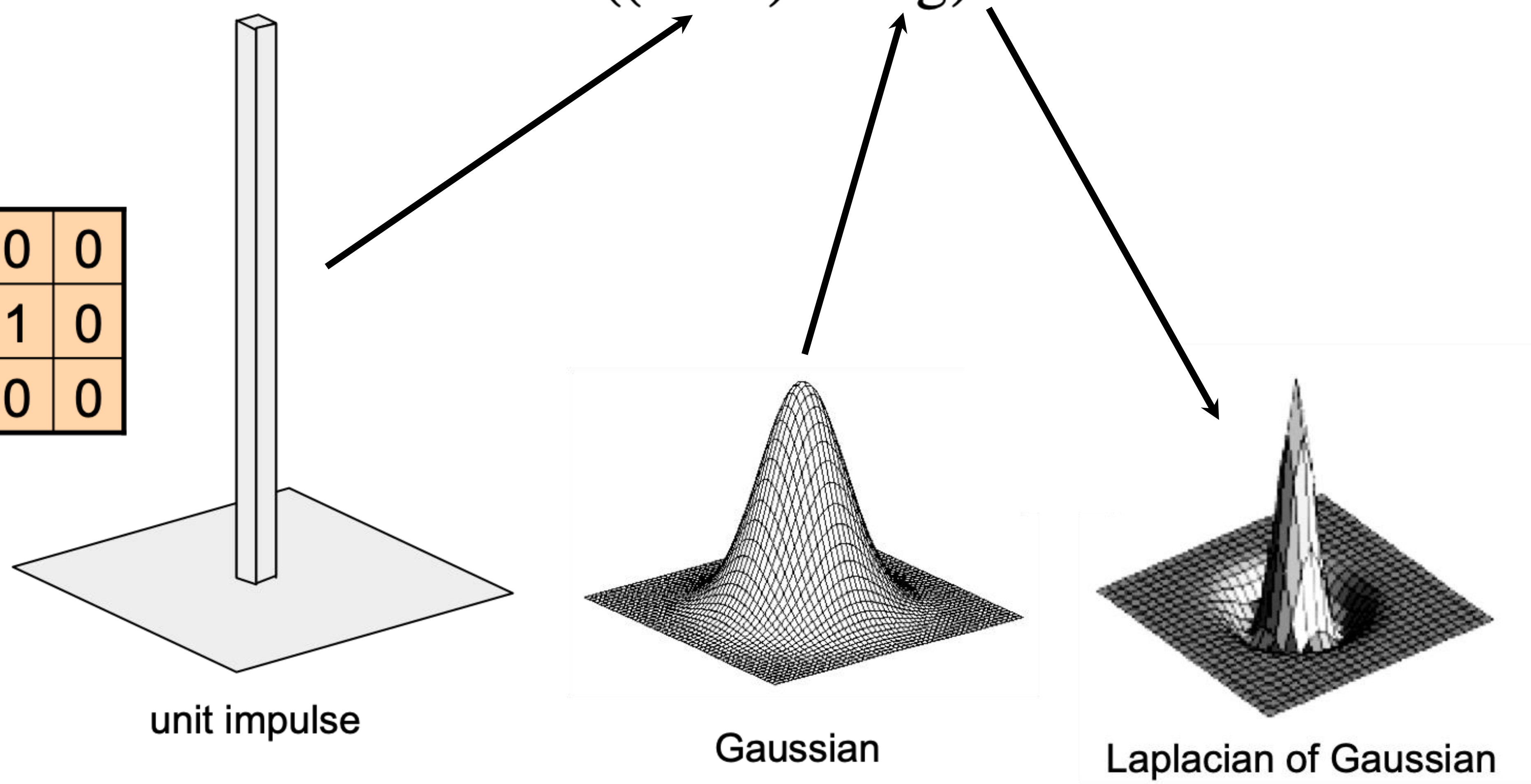
=



Example: Unsharp Masking

$$F + \alpha(F - F * g) =$$
$$F * e + \alpha(F * e - F * g) =$$
$$F * e + \alpha F * e - \alpha F * g =$$
$$(1 + \alpha)(F * e) - \alpha F * g =$$
$$F * ((1 + \alpha)e - \alpha g)$$

0	0	0
0	1	0
0	0	0



Example: Partial Derivations

- The partial derivation of a function f can be estimated as a symmetric finite difference

$$\frac{\partial f(x,y)}{\partial x} \approx f(x+1,y) - f(x-1,y)$$

- How does the corresponding filter kernel look like?

Example: Partial Derivations

- The partial derivation of a function f can be estimated as a symmetric finite difference

$$\frac{\partial f(x,y)}{\partial x} \approx f(x+1,y) - f(x-1,y)$$

$$H = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

- How does the corresponding filter kernel look like?
- When is the filter’s response large?

Example: Partial Derivations

- The partial derivation of a function f can be estimated as a symmetric finite difference
- How does the corresponding filter kernel look like?
- When is the filter's response large?
  - positive slope (from left to right)
- What about the partial derivation in y direction?

$$\frac{\partial f(x,y)}{\partial x} \approx f(x+1,y) - f(x-1,y)$$

$$H = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Example: Partial Derivations

- The partial derivation of a function f can be estimated as a symmetric finite difference

$$\frac{\partial f(x,y)}{\partial x} \approx f(x+1,y) - f(x-1,y)$$

$$H = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

- How does the corresponding filter kernel look like?
- When is the filter's response large?
  - positive slope (from left to right)
- What about the partial derivation in y direction?

$$\frac{\partial f(x,y)}{\partial y} \approx f(x,y+1) - f(x,y-1)$$

$$H = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad H = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

- Note that finite differences respond strongly to noise!

Example: Partial Derivations

- The partial derivation of a function f can be estimated as a symmetric finite difference
- How does the corresponding filter kernel look like?
- When is the filter's response large?
  - positive slope (from left to right)
- What about the partial derivation in y direction?
- Note that finite differences respond strongly to noise!

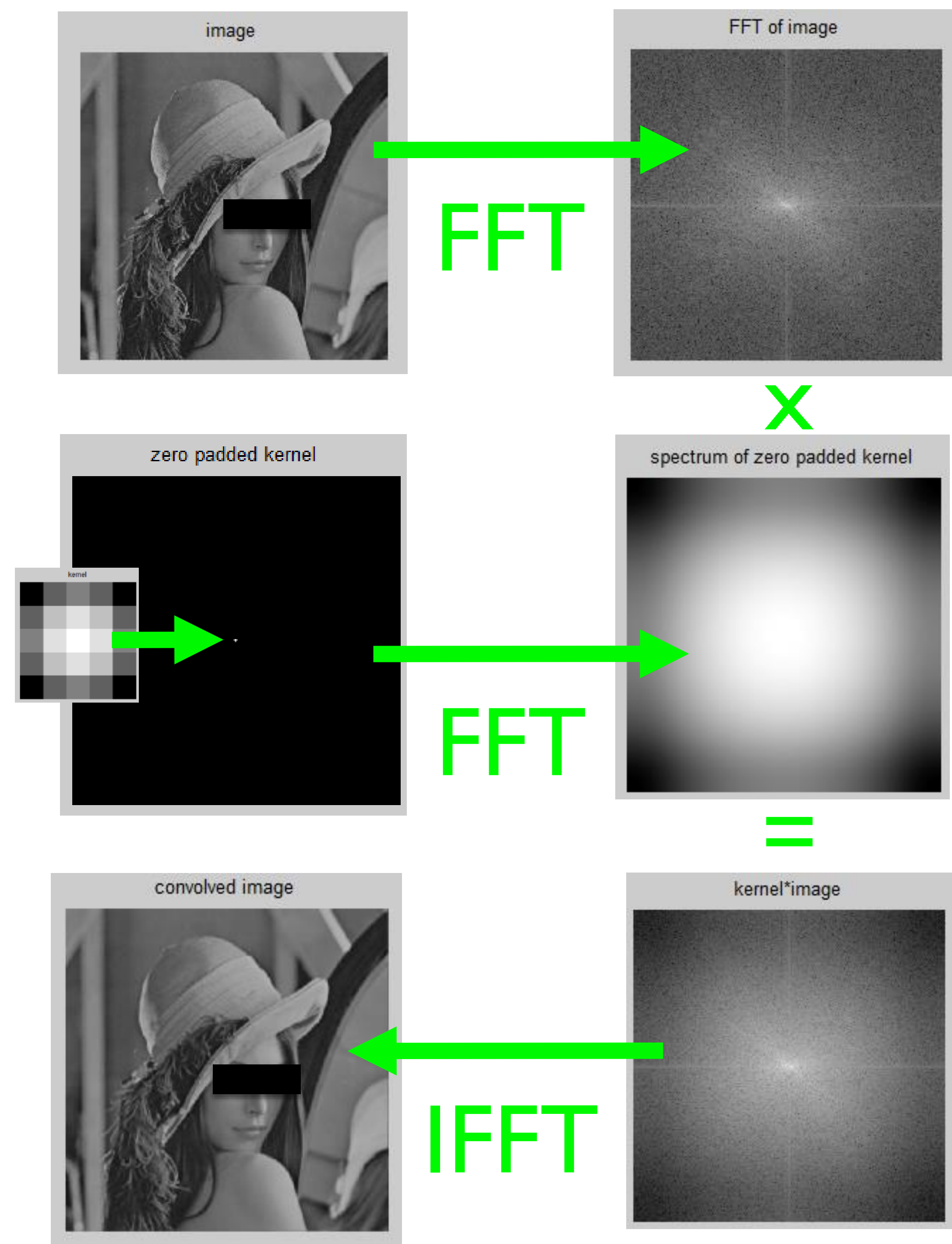
examples for partial derivations in horizontal and vertical directions

Frequency Decomposition

- An image can also be seen as a discrete two-dimensional signal
- As any signal, it can be decomposed into an integral of weighted basis functions
- A Discrete (Fast) Fourier Transformation applies cos+/-sin basis functions at different frequencies, while the weight coefficients represent the amplitude / magnitude of a particular frequency
- A Discrete Cosine Transformation (this is what JPEG compression uses) applies only a cosine basis functions for decomposition
- Thus, every image can be decomposed into a spectrum of frequency of a particular basis function
- This spectrum can be processed / analyzed as well
- The image can be composed from the spectrum through integration

Spatial vs. Frequency Domain

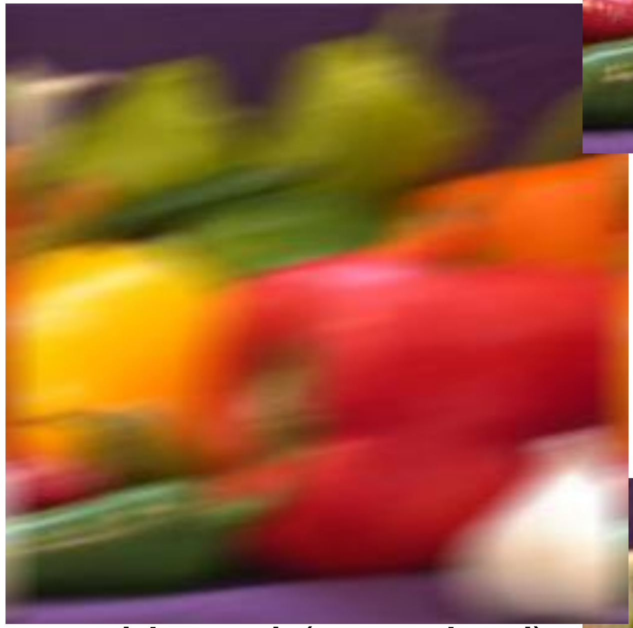
- Convolution theorem:
  - convolution in spatial domain equals a multiplication in frequency domain
  - a division in frequency domain equals the inverse operation, which is called deconvolution / inverse filtering
- In case of deconvolution with a Gaussian kernel, this is equivalent with sharpening (or deblurring)



Deconvolution / Inverse Filtering

- Given an image I that is convolved with signal K results in $I' = I*K+N$ (whereby N is noise)
- If noise N and convolution signal H are known, then original image F can be recovered with $I = K^{-1}*(I'-N)$
- This is known as deconvolution
- The challenge is to estimate K and N
- Example: image deblurring
  - we know that for camera lenses with round apertures the point spread function (PSF) of defocus is Gaussian – thus K is Gaussian
  - the scale of K depends on the amount of defocus
  - noise model can be given or not (then it has to be predicted)
- Wiener filter, regularized filter, Lucy-Richardson algorithm, blind deconvolution

original image



blurred (convolved) image

reconstructed (deconvolved) image

Deconvolution / Inverse Filtering

- Given an image I that is convolved with signal K results in $I' = I * K + N$ (whereby N is noise)
- If noise N and convolution signal H are known, then original image F can be recovered with $I = K^{-1} * (I' - N)$
- This is known as deconvolution
- The challenge is to estimate K and N
- Example: image deblurring
  - we know that for camera lenses with round apertures the point spread function (PSF) of defocus is Gaussian – thus K is Gaussian
  - the scale of K depends on the amount of defocus
  - noise model can be given or not (then it has to be predicted)
- Wiener filter, regularized filter, Lucy-Richardson algorithm, blind deconvolution

convolution in spatial domain:

$$I(x,y) * K_s(x,y) = I'(x,y)$$

convolution in frequency domain (Fourier transform + convolution theorem):

$$\hat{I}(f_x,f_y) \cdot \hat{K}_s(f_x,f_y) = \hat{I}'(f_x,f_y)$$

deconvolution in frequency domain:

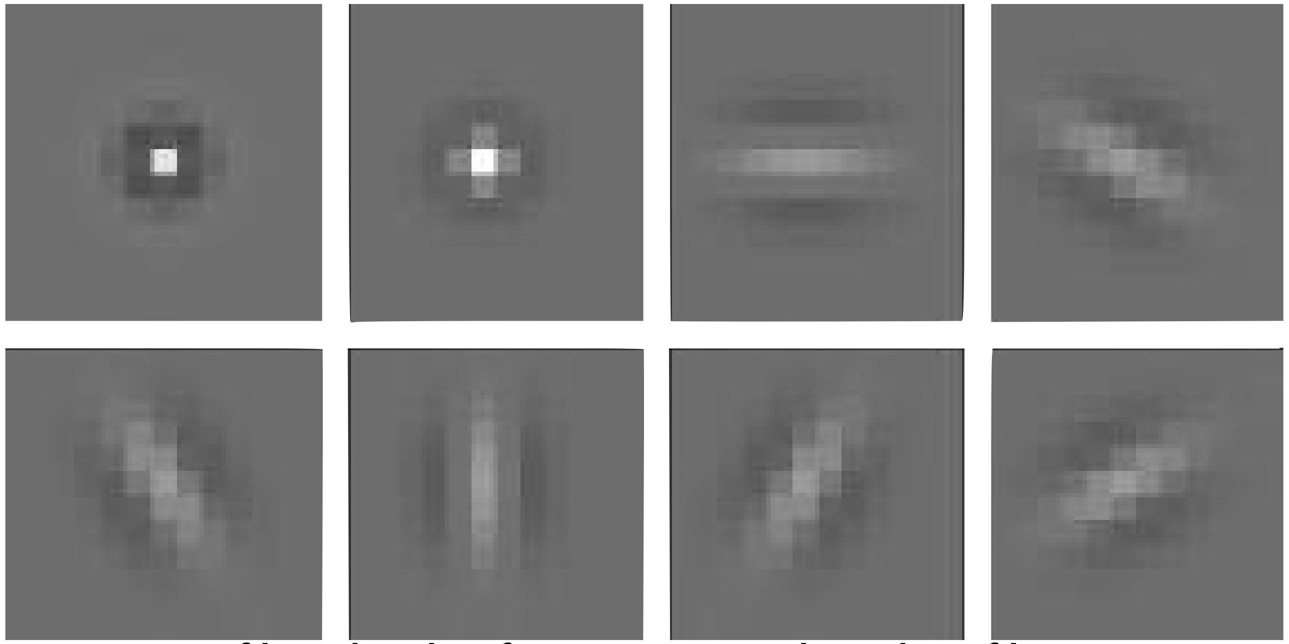
$$\hat{I}(f_x,f_y) = \frac{\hat{I}'(f_x,f_y)}{\hat{K}_s(f_x,f_y)}$$

This is simplified and does not consider a noise model. Division by small values in frequency domain leads to ringing artefacts in spatial domain. Better: apply regularized techniques.

ringing

Filter Banks

- What filters should be used for identifying important features?
  - a lot of research has (and is still) being carried out on this
  - some use filters that are adapted to the human visual system (spots and bars at different orientations)
  - others use simpler edge and line filters
- How do we filter features (e.g., spots or bars) at different scales?



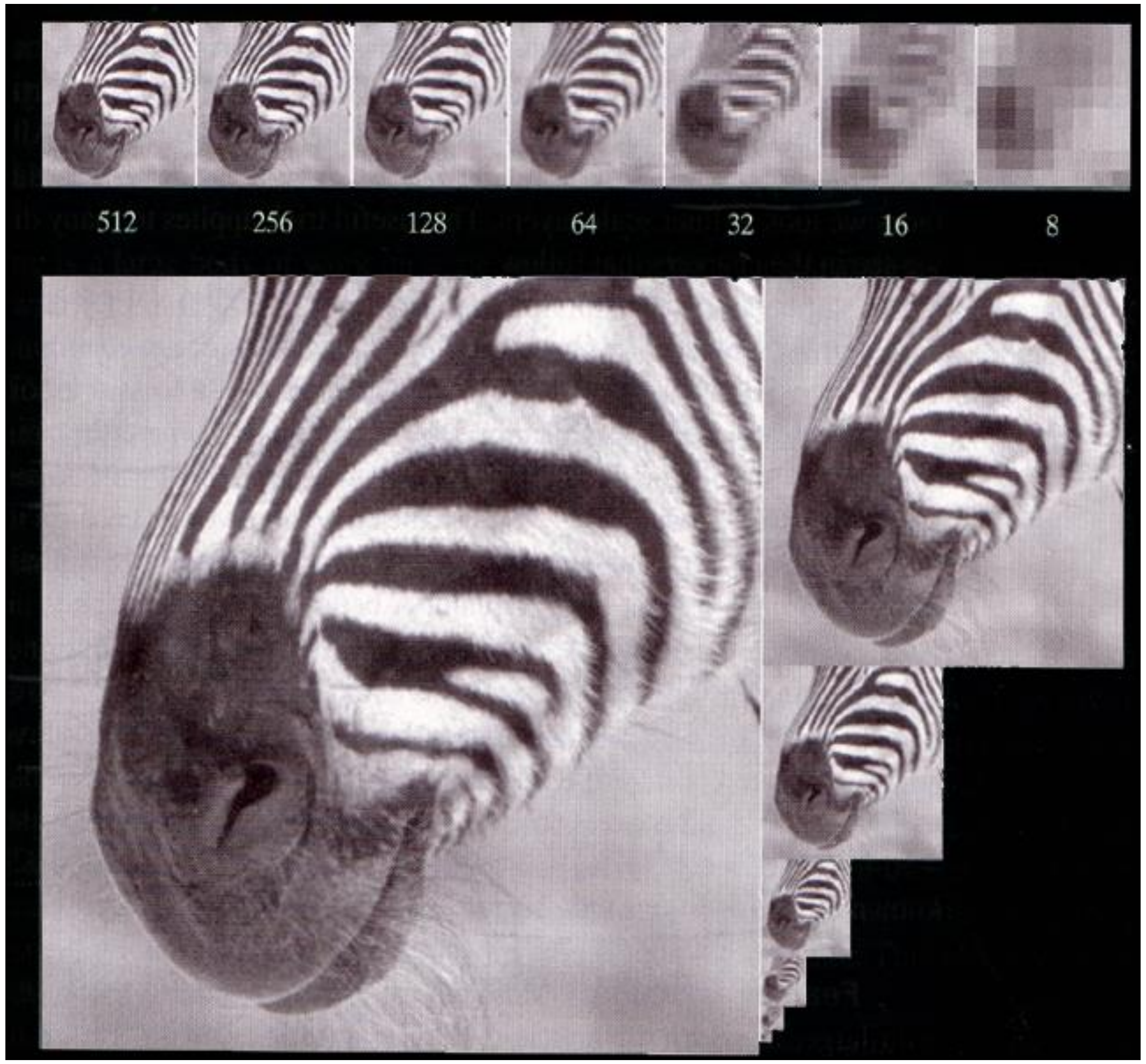
filter bank of two spot and six bar filters

Image Pyramids

- An image pyramid is a collection of representations of the same image at different scales
- Typically, each layer is half the width and half the height of the previous layer
- At each layer, a Gaussian pyramid stores a smoothened (Gaussian kernel) and down sampled version of the previous layer
- A Gauss filter is a low-pass filter
- In terms of signal processing, each level of the Gaussian pyramid is reduced by a subband of higher frequencies (going bottom up)
- The top entry (one pixel) represents the basis (average image intensity)

$$P_G^{n+1}(I) = \downarrow (G_s \otimes P_G^n(I))$$

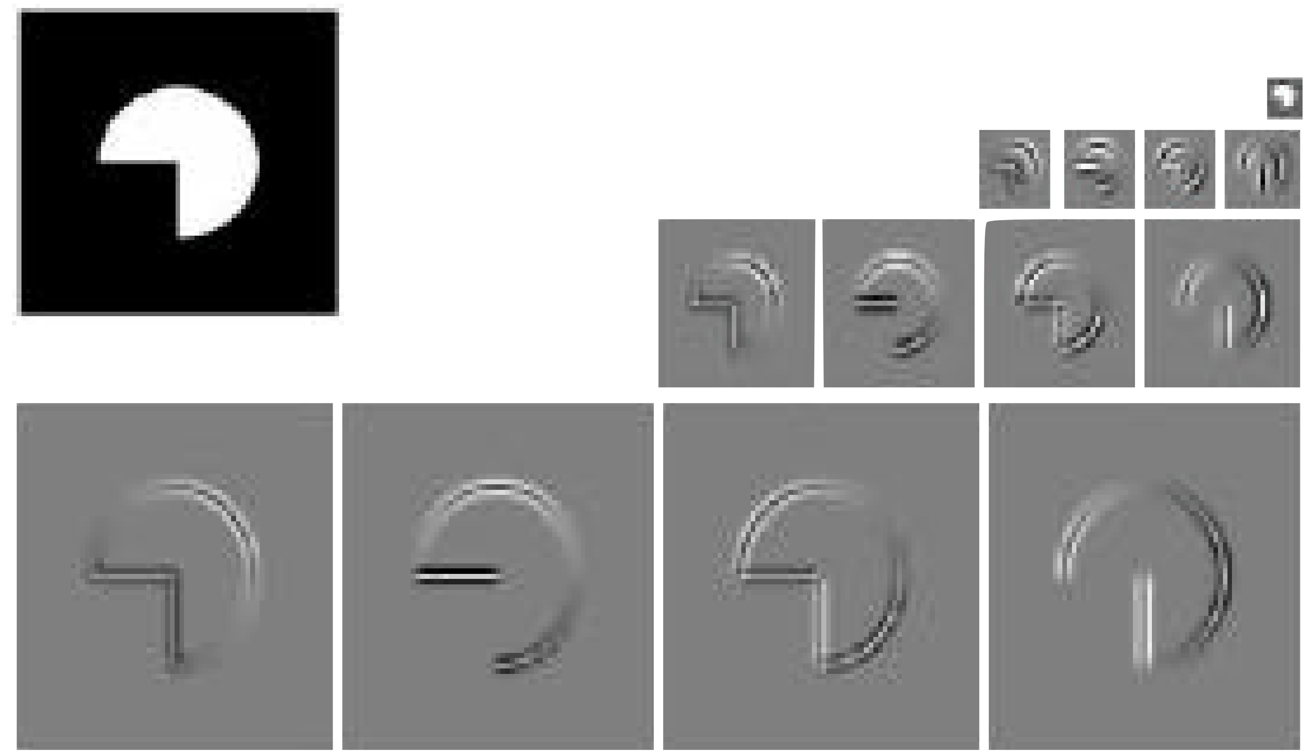
$$P_G^1(I) = I$$



Gaussian pyramid

Oriented Image Pyramids

- Apply filter bank to levels of pyramid (Gaussian or Laplacian)
- This is called oriented pyramid or steerable pyramid
- It contains the feature response at different scales for Gaussian (or different sub-bands for Laplacian)
- Multiple image pyramids can be the result of a texture analysis process



oriented filter bank of four orientation filters applied to levels of image pyramid pyramid

Decomposition using a Gaussian Basis

- A Gauss filter is a low-pass filter
- Consequently, Gaussian pyramid successively removes high frequencies
- What happens if two consecutive layers are subtracted?

$$P_L^n(I) = P_G^n(I) - \uparrow P_G^{n+1}(I)$$

Decomposition using a Gaussian Basis

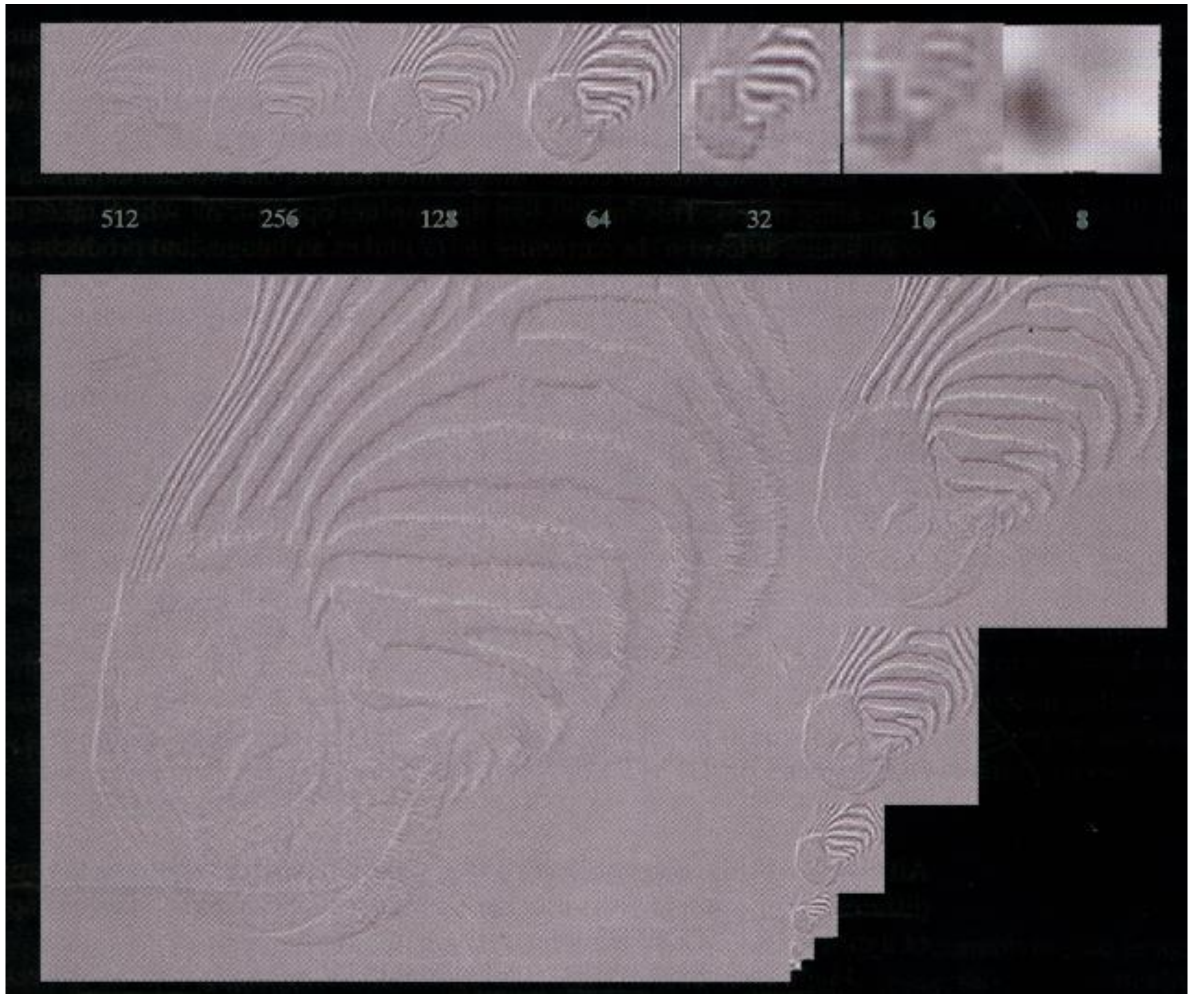
- A Gauss filter is a low-pass filter
- Consequently, Gaussian pyramid successively removes high frequencies
- What happens if two consecutive layers are subtracted?
  - the resulting image contains the removed (via Gauss filter) spatial frequencies (a sub-band)
- What happens if this is done for each layer of the Gaussian pyramid?

$$P_L^n(I) = P_G^n(I) - \uparrow P_G^{n+1}(I)$$

Decomposition using a Gaussian Basis

- A Gauss filter is a low-pass filter
- Consequently, Gaussian pyramid successively removes high frequencies
- What happens if two consecutive layers are subtracted?
    - the resulting image contains the removed (via Gauss filter) spatial frequencies (a sub-band)
- What happens if this is done for each layer of the Gaussian pyramid?
    - the result is another image pyramid, called Laplacian pyramid, that can be thought of as the response of a band-pass filter (a sub-band at each level)
- What happens when the pyramid is collapsed?

$$P_L^n(I) = P_G^n(I) - \uparrow P_G^{n+1}(I)$$



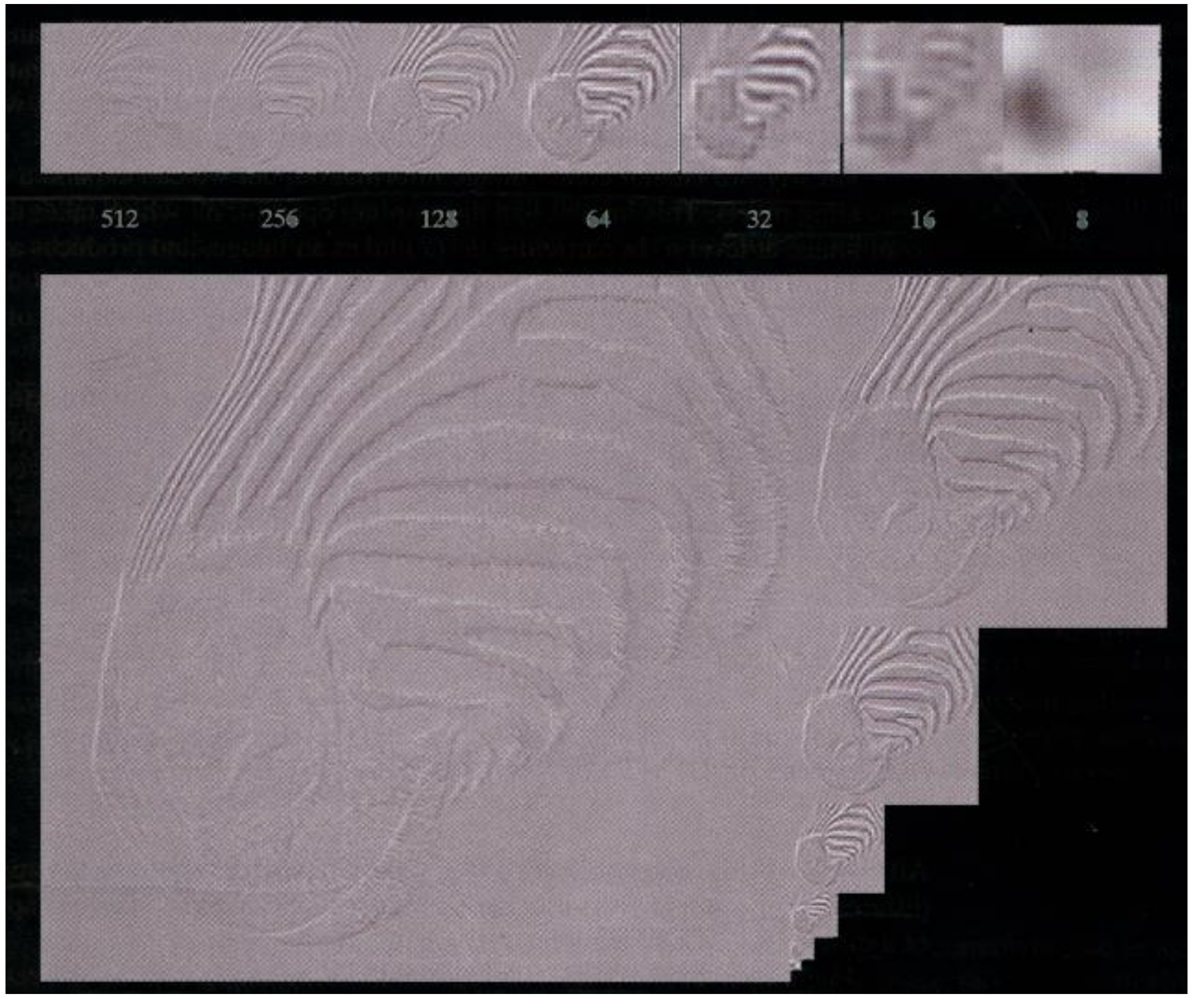
Laplacian pyramid

$$P_L^n(I) = P_L^n(I) + \uparrow P_L^{n+1}(I) \rightarrow I$$

Decomposition using a Gaussian Basis

- A Gauss filter is a low-pass filter
- Consequently, Gaussian pyramid successively removes high frequencies
- What happens if two consecutive layers are subtracted?
    - the resulting image contains the removed (via Gauss filter) spatial frequencies (a sub-band)
- What happens if this is done for each layer of the Gaussian pyramid?
    - the result is another image pyramid, called Laplacian pyramid, that can be thought of as the response of a band-pass filter (a sub-band at each level)
- What happens when the pyramid is collapsed?
    - the original image results

$$P_L^n(I) = P_G^n(I) - \uparrow P_G^{n+1}(I)$$



Laplacian pyramid

$$P_L^n(I) = P_L^n(I) + \uparrow P_L^{n+1}(I) \rightarrow I$$

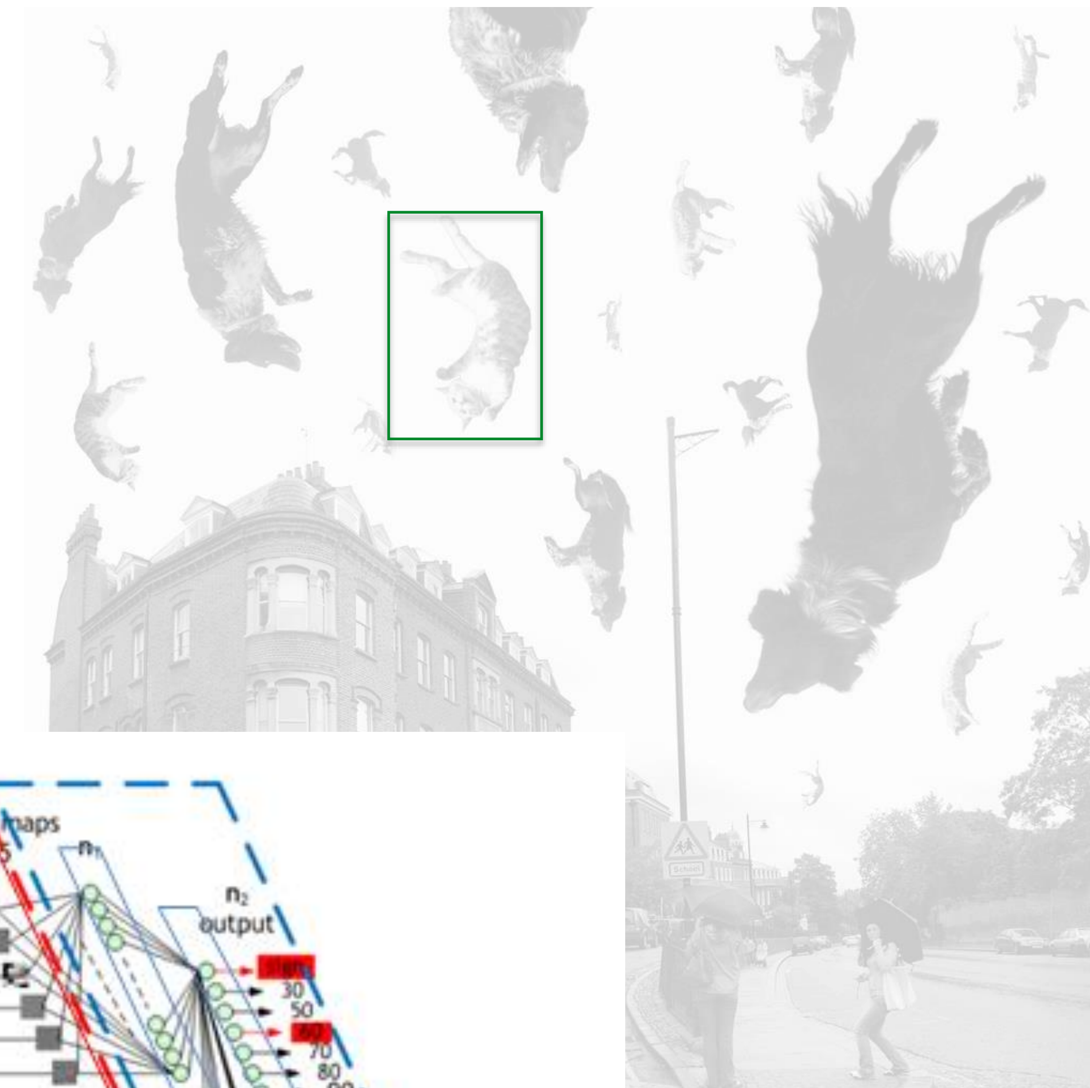
Recap. Example: Classification

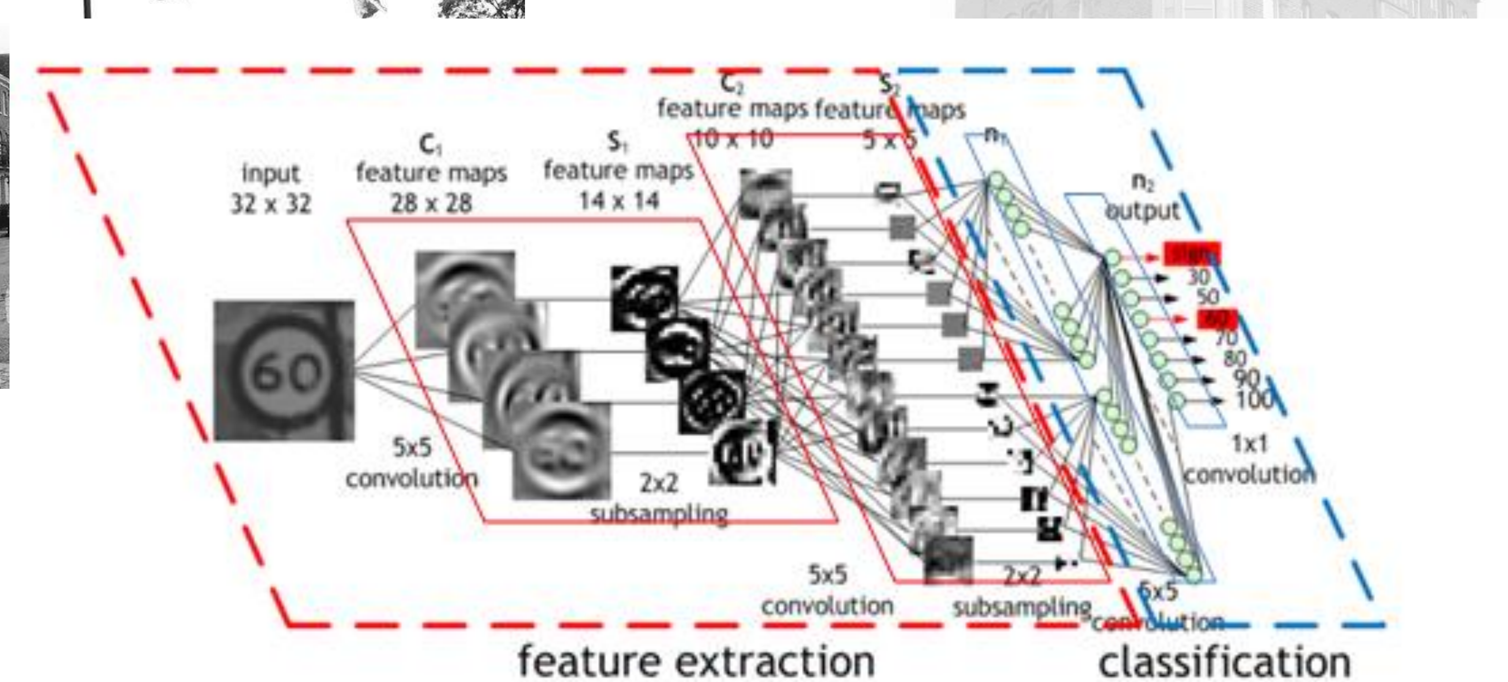
$* \; \boxed{?} \; =$

Recap. Example: Classification

*

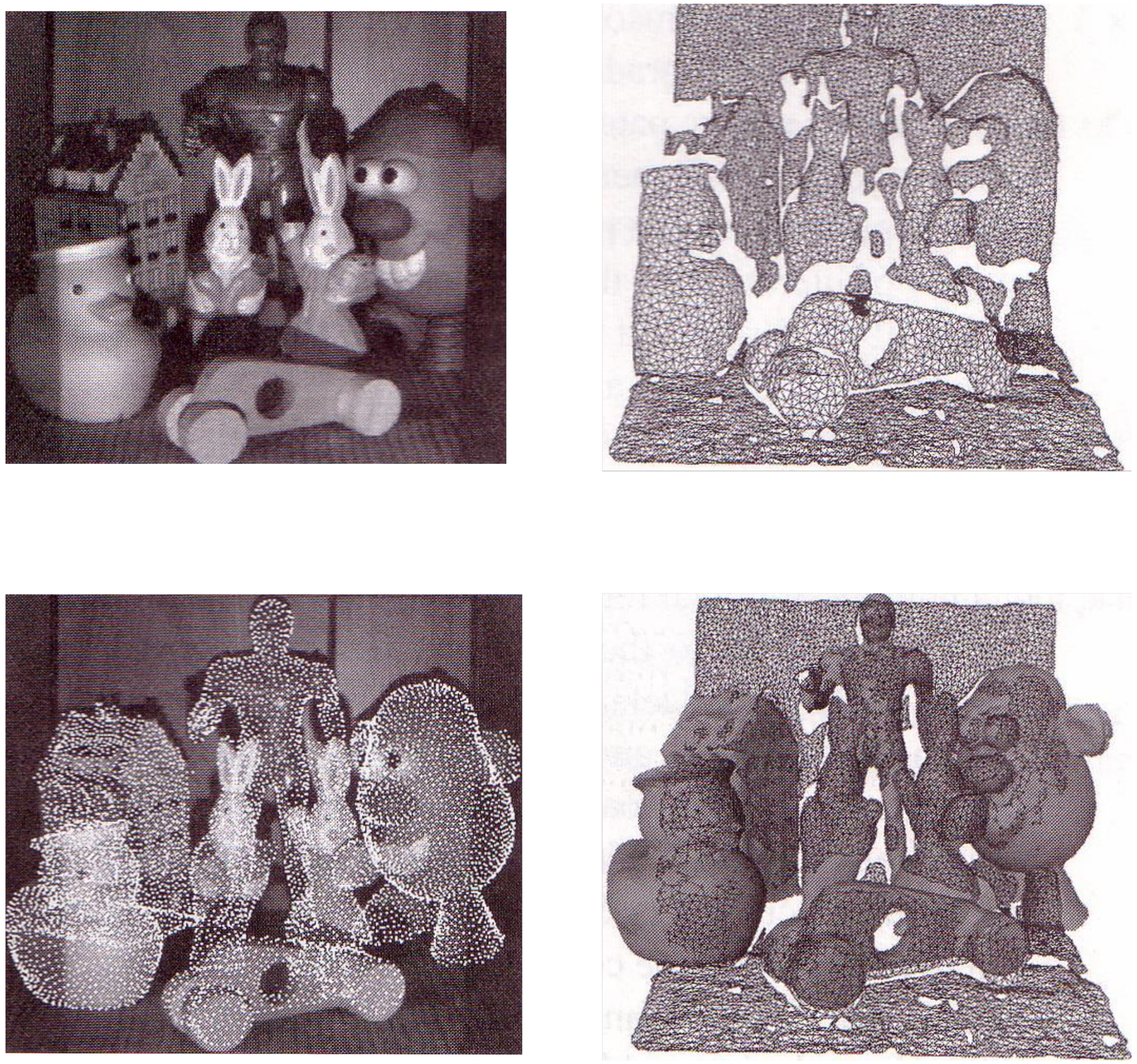
=



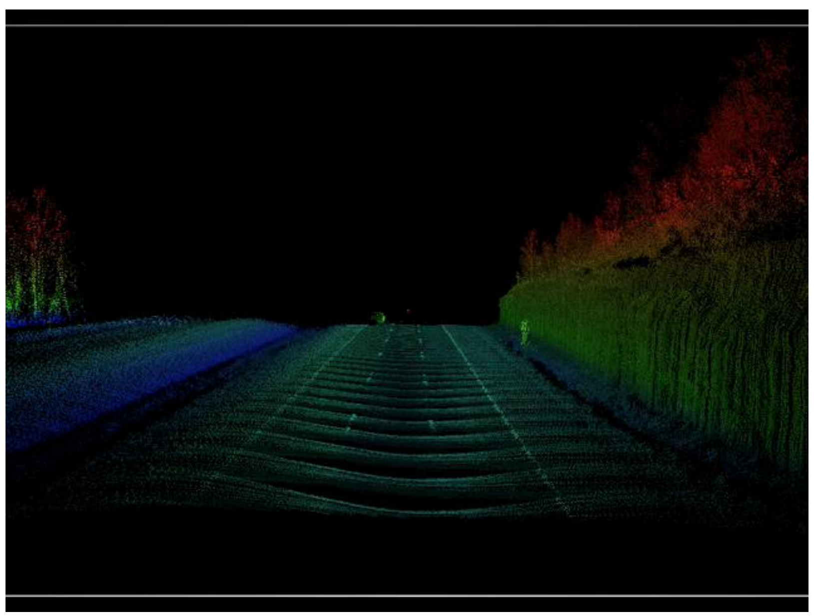


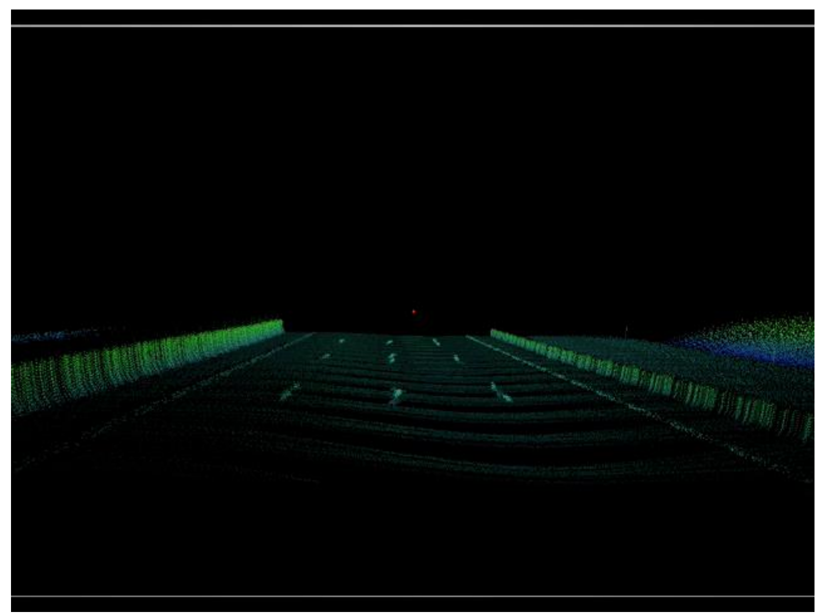
Processing Depth Data

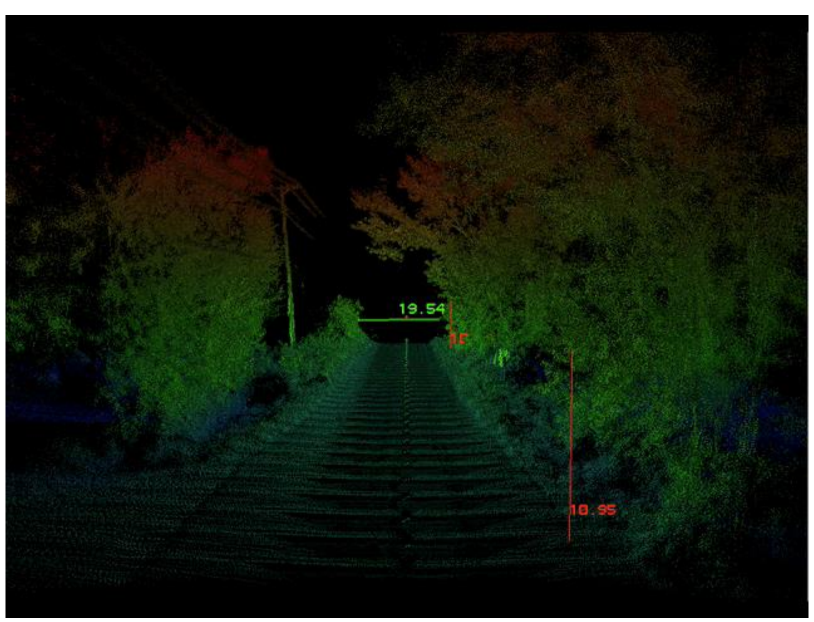
- In many cases, depth maps can be processed like images
- Image operators, such as filters, can be applied
- This, for instance, allows to:
  - handle discontinuities and overlaps
  - fill missing portions
  - smoothen geometric noise
  - analyze depth images
  - find features such as edges
  - segmentation of objects
  - 3D object recognition
  - etc.

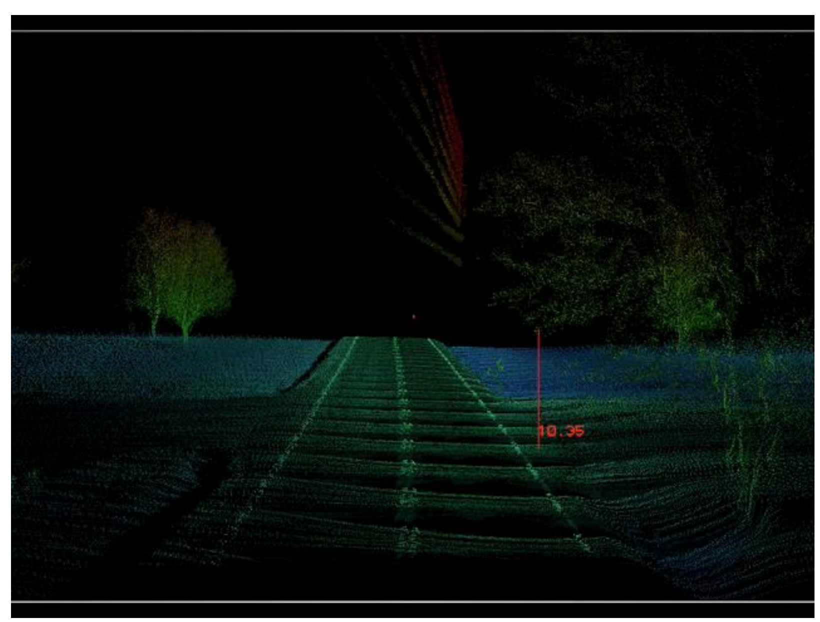


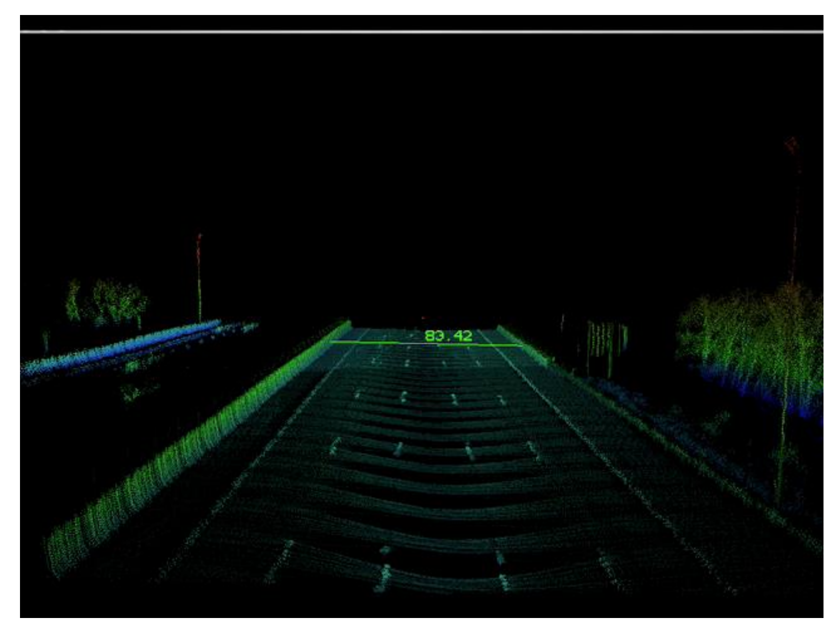
Example: LIDAR











Course Schedule

Type	Date	Time	Room	Topic	Comment
Lec1	11.10.2022	12:00-13:30	H1	Introduction and Course Overview	
Lab1	10./11./12./13.10.2022	17:15-18:45	S3055	Introduction to Python	
Lec2	18.10.2022	12:00-13:30	HS 1	Spatial and Frequency Domain Processing	
Lab2	17./18./19./20.10.2022	17:15-18:45	S3055	Introduction to IP/CV Modules	
Lec3	25.10.2022	12:00-13:20	HS 1	Gradient Domain Processing	National Holiday (26.10.)
Lec4	08.11.2022	12:00-13:30	HS 10	Segmentation and Local Features	Allerheiligen (2.11.)
Lab3	07./08./08./10.11.2022	17:15-18:45	S3055	Project Introduction	
Lec5	15.11.2022	12:00-13:30	HS 1	Basics of Cameras	
Lec6	22.11.2022	12:00-13:30	HS 1	Geometric Camera Calibration	
Lab4	21./22./23./24.11.2022	17:15-18:45	S3055	Project Basics and Related Work	
Lec7	29.11.2022	12:00-13:30	HS 1	The Geometry of Multiple Views	
Lec8	06.11.2022	12:00-13:30	HS 1	Stereoscopic Depth Estimation	Mariä Empfängnis (8.12.)
Lec9	13.12.2022	12:00-13:30	HS 1	Range Scanning and Data Processing	
Lab5	12./13./14./15.12.2022	17:15-18:45	S3055	Presentation of Initial Ideas	Christmas Break
Lec10	10.01.2023	12:00-13:30	HS 1	Structure from Motion	
Lab6	09./10./11./12.01.2023	17:15-18:45	S3055	Presentation of Intermediate Results and Final Concepts	
Lec11	17.01.2023	12:00-13:30	HS 1	Computational Imaging	
Lec12	24.01.2023	12:00-13:30	HS 1	Recap and Q&A	
Lab7	23./24./25./26.01.2023	17:15-18:45	S3055	Final Project Presentations	
Ex1	31.01.2023	12:00-13:30	HS 1	Exam (Hauptklausur)	
Ex2	28.02.2023	15:30-17:00	TBA	Retry Exam (Nachklausur)	

Thank You!