Computer Vision I

Basic Image Processing - 08.05.2013

TECHNISCHE UNIVERSITÄT DARMSTADT

vi visual inference

Announcements

TECHNISCHE UNIVERSITÄT DARMSTADT

- No office hours today:
  - Please see TAs for this week.

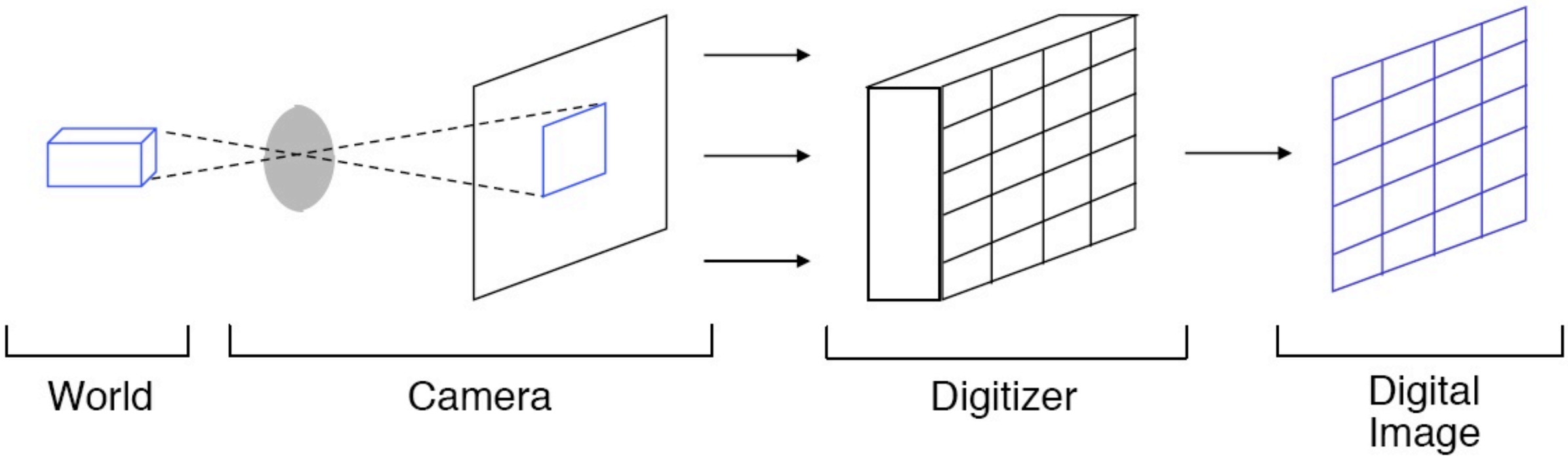
- Homework assignment 1:
  - Out later today

- Lab access:
  - Cards and logins will be distributed after the lecture

Overview

TECHNISCHE UNIVERSITÄT DARMSTADT

- Computer Vision: “invert” the imaging process
  - Today: 2D (2-dimensional) digital image processing
  - Later: ‘pattern recognition’ / 3D image analysis
  - Later: image understanding



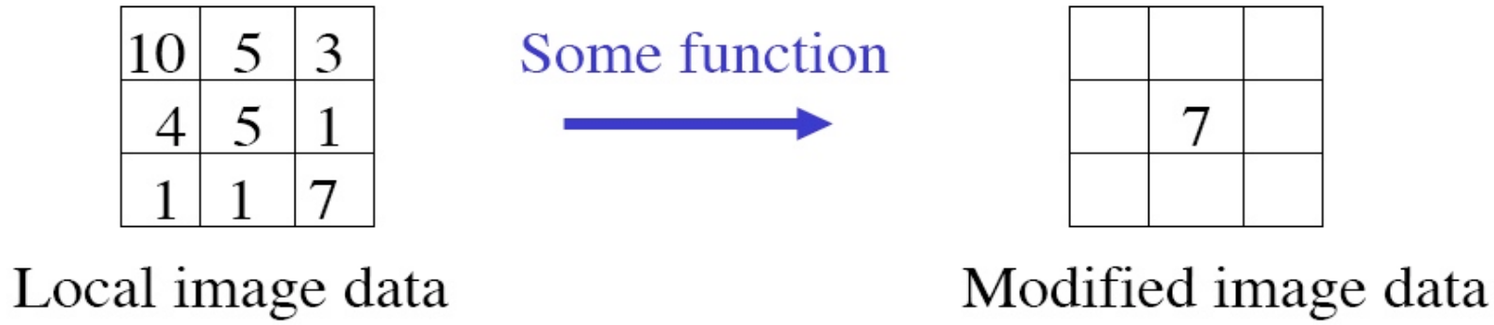
Digital Image Processing

TECHNISCHE UNIVERSITÄT DARMSTADT

- Some Basics
  - (digital signal processing, FFT, …)
  - Image Filtering

Credits: slides adapted from
Bernt Schiele
Michael Black

- Image Filtering
  - to reduce noise
  - to fill-in missing values/information
  - to extract image features (e.g. edges/corners)
  - …



Today - Basics of Digital Image Processing

TECHNISCHE UNIVERSITÄT DARMSTADT

- Images
- Filtering
  - Linear filtering
  - Non-linear filtering & morphology
- Multi-scale image representation
  - Gaussian pyramid
  - Laplacian pyramid
- Edge detection
  - ‘Recognition using line drawings’
  - Image derivatives (1st and 2nd order)

What is an image?

TECHNISCHE UNIVERSITÄT DARMSTADT

- We can think of the image as a function:

$$I(x, y), \quad I : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$$

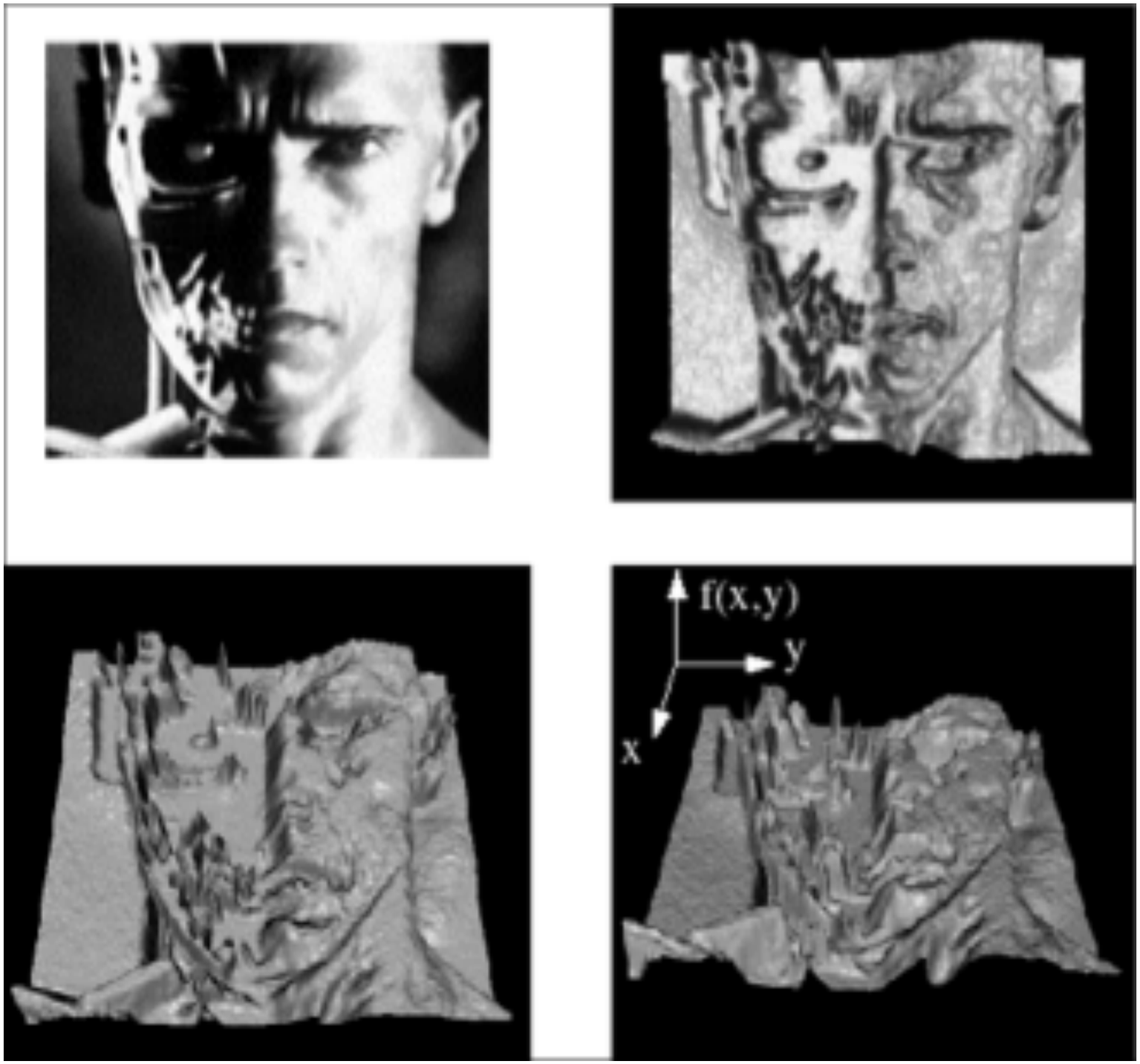
  - For every 2D point it tells us what the (light) intensity is.
  - Realistically, the size of the sensor is limited and so is the range of brightnesses it can capture:

$$I(x, y), \quad I : [a, b] \times [c, d] \rightarrow [0, m]$$

- Color images can be thought of as vector-valued functions:

$$\mathbf{I}(x, y) = \begin{pmatrix} I_R(x, y) \\ I_G(x, y) \\ I_B(x, y) \end{pmatrix}, \quad \mathbf{I} : [a, b] \times [c, d] \rightarrow [0, m]^3$$

Images as functions



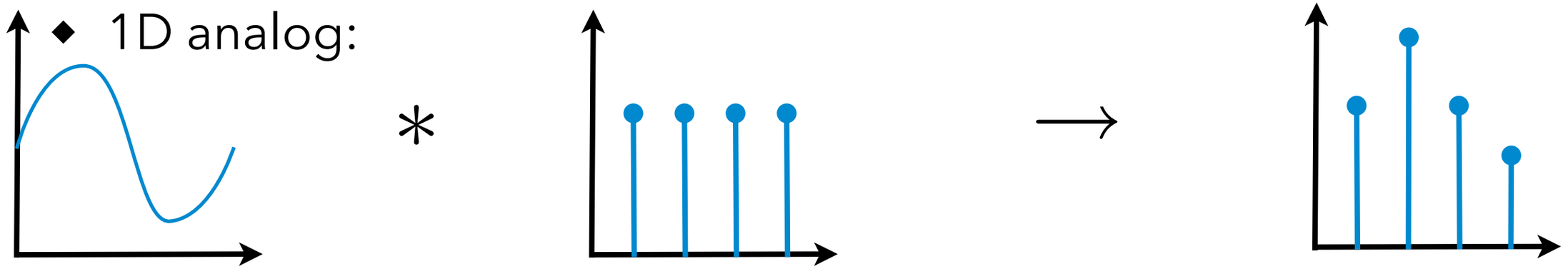
[from Steve Seitz]

What is a digital image?

TECHNISCHE UNIVERSITÄT DARMSTADT

- We usually do not work with spatially continuous functions, since our cameras do not sense in this way.
- Instead use (spatially) discrete images
  - Sample the 2D domain on a regular grid

- 1D analog:



continuous signal sampling operator sampled image

y =	x = 58	59	60	61	62	63	64	65	66	67	68	69	70	71	72
41	210	209	204	202	197	247	143	71	64	80	84	54	54	57	58
42	206	196	203	197	195	210	207	56	63	58	53	53	61	62	51
43	201	207	192	201	198	213	156	69	65	57	55	52	53	60	50
44	216	206	211	193	202	207	208	57	69	60	55	77	49	62	61
45	221	206	211	194	196	197	220	56	63	60	55	46	97	58	106
46	209	214	224	199	194	193	204	173	64	60	59	51	62	56	48
47	204	212	213	208	191	190	191	214	60	62	66	76	51	49	55
48	214	215	215	207	208	180	172	188	69	72	55	49	56	52	56
49	209	205	214	205	204	196	187	196	86	62	66	87	57	60	48
50	208	209	205	203	202	186	174	185	149	71	63	55	55	45	56
51	207	210	211	199	217	194	183	177	209	90	62	64	52	93	52
52	208	205	209	209	197	194	183	187	187	239	58	68	61	51	56
53	204	206	203	209	195	203	188	185	183	221	75	61	58	60	60
54	200	203	199	236	188	197	183	190	183	196	122	63	58	64	66
55	205	210	202	203	199	197	196	181	173	186	105	62	57	64	63

- Intensity/color values usually also discrete
- Quantize the values per channel (e.g. 8 bit)

Today - Basics of Digital Image Processing

TECHNISCHE UNIVERSITÄT DARMSTADT

- Images
- Filtering
  - Linear filtering
  - Non-linear filtering & morphology
- Multi-scale image representation
  - Gaussian pyramid
  - Laplacian pyramid
- Edge detection
  - ‘Recognition using line drawings’
  - Image derivatives (1st and 2nd order)

Linear Operations / Systems

TECHNISCHE UNIVERSITÄT DARMSTADT

- Basic properties:
  - homogeneity $T[a \cdot X] = a \cdot T[X]$
  - additivity $T[X + Y] = T[X] + T[Y]$
  - superposition $T[a \cdot X + b \cdot Y] = a \cdot T[X] + b \cdot T[Y]$
- Examples:
  - matrix-vector operations
  - convolutions

Convolution

TECHNISCHE UNIVERSITÄT DARMSTADT

- Replace each pixel by a linear combination of its neighbors (and itself).
- 2D convolution (discrete):

$$g = f * h \qquad g(i,j) = \sum_{k,l} f(i-k, j-l)h(k,l) = \sum_{k,l} f(k,l)h(i-k, j-l),$$

45	60	98	127	132	133	137	133
46	65	98	123	126	128	131	133
47	65	96	115	119	123	135	137
47	63	91	107	113	122	138	134
50	59	80	97	110	123	133	134
49	53	68	83	97	113	128	133
50	50	58	70	84	102	116	126
50	50	52	58	69	86	101	120

*

0.1	0.1	0.1
0.1	0.2	0.1
0.1	0.1	0.1

=

69	95	116	125	129	132
68	92	110	120	126	132
66	86	104	114	124	132
62	78	94	108	120	129
57	69	83	98	112	124
53	60	71	85	100	114

smaller output?

image $f(i,j)$ | filter (kernel) $h(i,j)$ | filtered image $g(i,j)$

Convolution / Correlation

TECHNISCHE UNIVERSITÄT DARMSTADT

- Convolution is
  - linear $h * (f_0 + f_1) = h * f_0 + h * f_1$
  - associative $(f * g) * h = f * (g * h)$
  - commutative $f * h = h * f$
  - shift-invariant $g(i,j) = f(i+k, j+l) \Leftrightarrow (h * g)(i,j) = (h * f)(i+k, j+l)$
  - can be represented as matrix-vector product $\boldsymbol{g} = \boldsymbol{H}\boldsymbol{f}$
  - Continuous versions exist, skip them here...
- Correlation:
  - Closely related, but do not mirror filter

$$g = f \otimes h \qquad g(i,j) = \sum_{k,l} f(i+k, j+l) h(k,l)$$

Filtering to Reduce Noise

TECHNISCHE UNIVERSITÄT DARMSTADT

- "Noise" is what we are not interested in
  - low-level noise: light fluctuations, sensor noise, quantization effects, finite precision, …
  - complex noise (not today): shadows, extraneous objects.

- Assumption:
  - The pixel's neighborhood contains information about its intensity

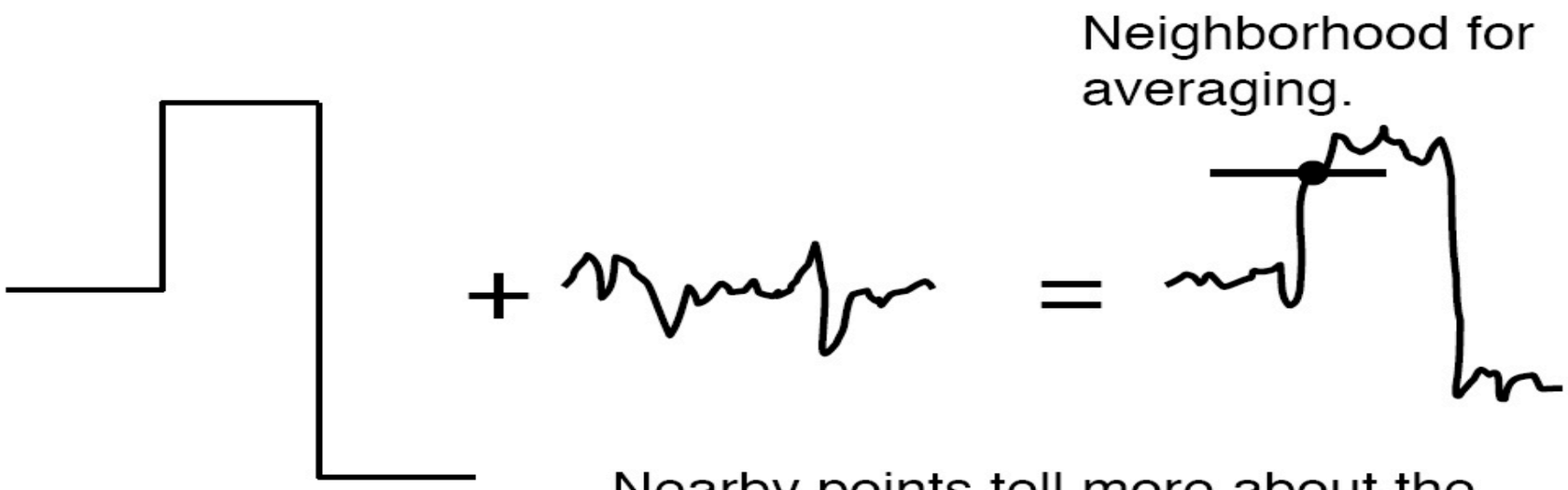
$$\begin{array}{|c|c|c|} \hline 2 & 3 & 3 \\ \hline 3 & 20 & 2 \\ \hline 3 & 2 & 3 \\ \hline \end{array} \rightarrow \begin{array}{|c|c|c|} \hline 2 & 3 & 3 \\ \hline 3 & 3 & 2 \\ \hline 3 & 2 & 3 \\ \hline \end{array}$$

Model: Additive Noise

TECHNISCHE UNIVERSITÄT DARMSTADT

Signal + Noise = Image

$$S + N = I$$



Model: Additive Noise

TECHNISCHE UNIVERSITÄT DARMSTADT

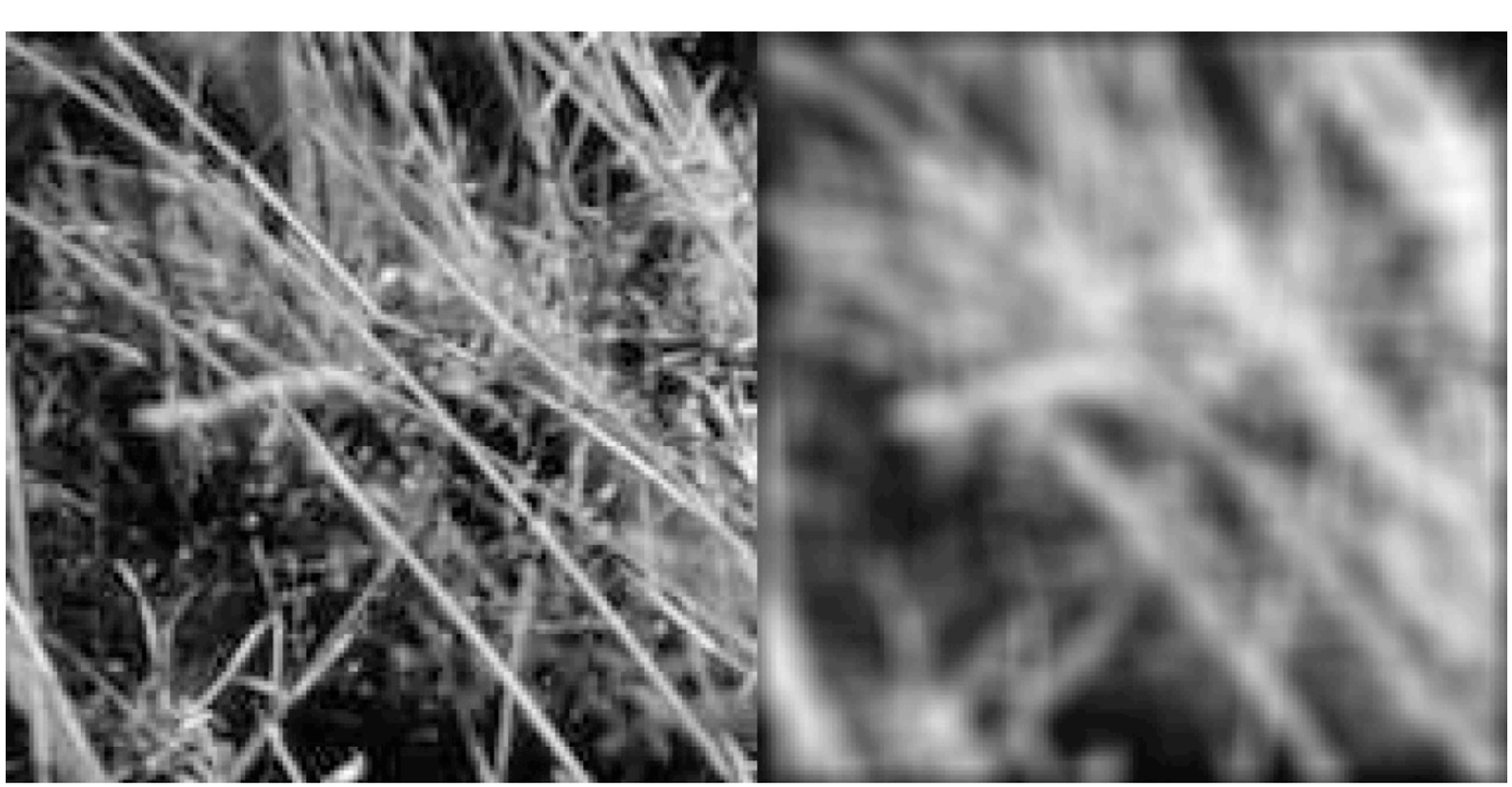
- Image I = Signal S + Noise N
  - I.e. noise does not depend on the signal
- We consider:
  - I_i : intensity of i-th pixel
  - $I_i = s_i + n_i$ with $E[n_i] = 0$
    - s_i deterministic
    - n_i, n_j independent for $i \neq j$
    - n_i, n_j i.i.d. (independent, identically distributed)
- Therefore:
  - Intuition: Averaging noise reduces its effect

Average Filter

TECHNISCHE UNIVERSITÄT DARMSTADT

- Average Filter
  - replaces each pixel with an average of its neighborhood
  - Mask with positive entries that sum to 1
- If all weights are equal, it is called a box filter

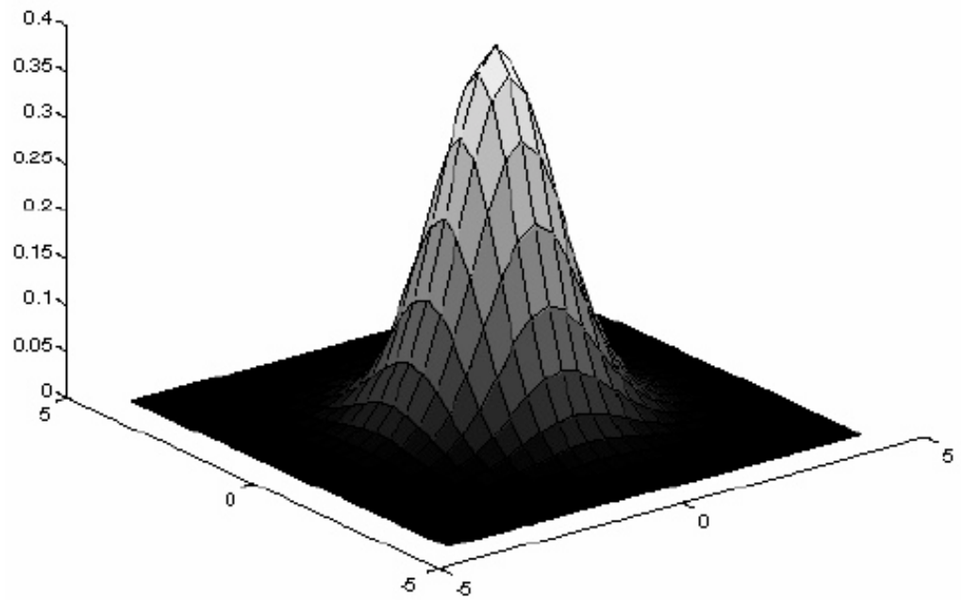
$$\frac{1}{9} \cdot \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$



Gaussian Averaging

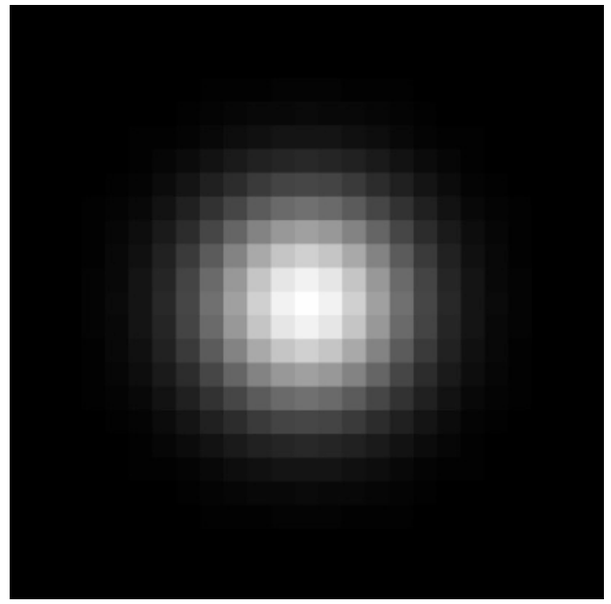
TECHNISCHE UNIVERSITÄT DARMSTADT

- Isotropic Gaussian (rotationally symmetric)
- Weighs nearby pixels more than distant ones



- Smoothing kernel proportional to

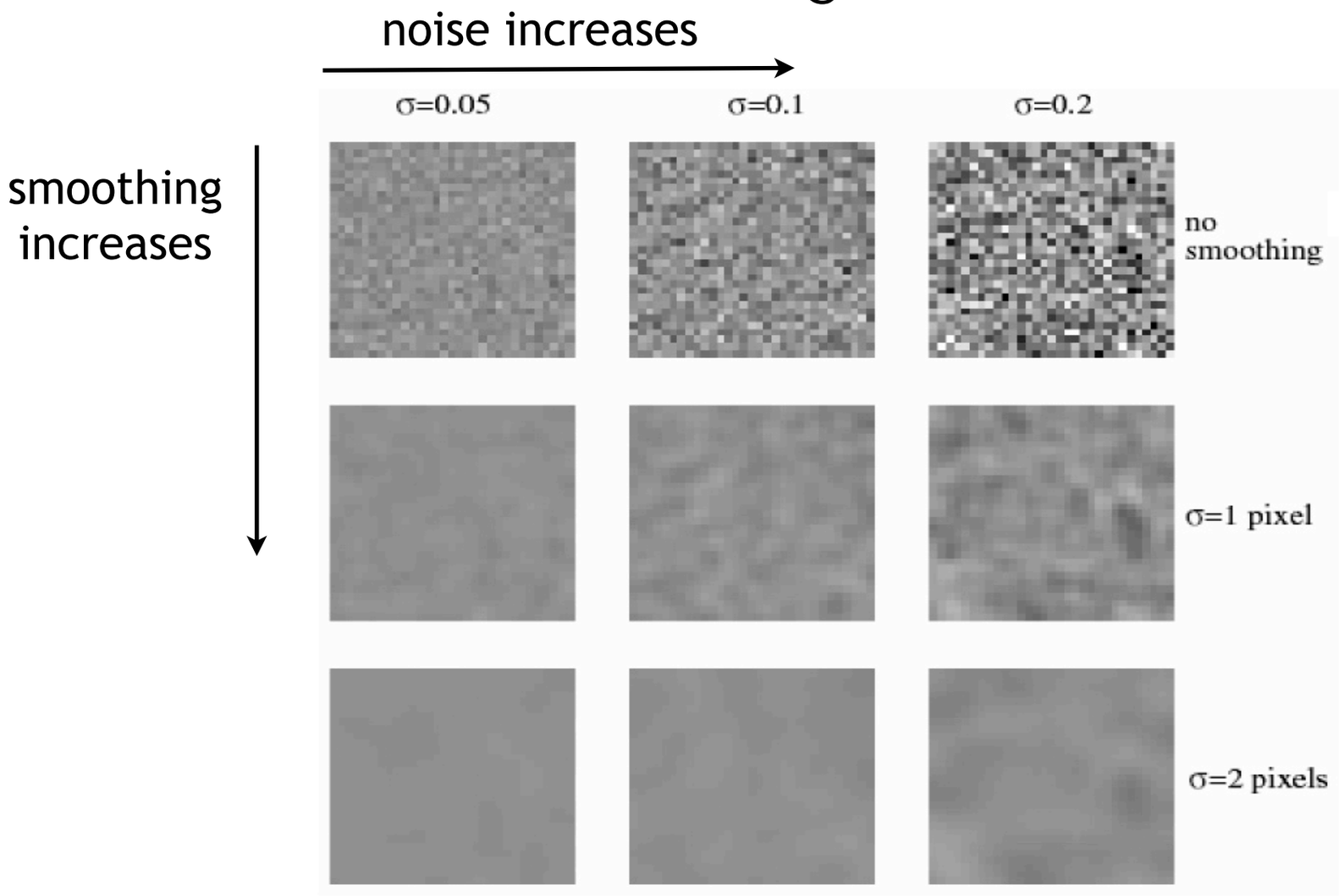
$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$



Smoothing with a Gaussian

TECHNISCHE UNIVERSITÄT DARMSTADT

- Effects of smoothing:
  - Each column shows realizations of an image of Gaussian noise
  - Each row shows smoothing with Gaussians of different width



Gaussians are everywhere!

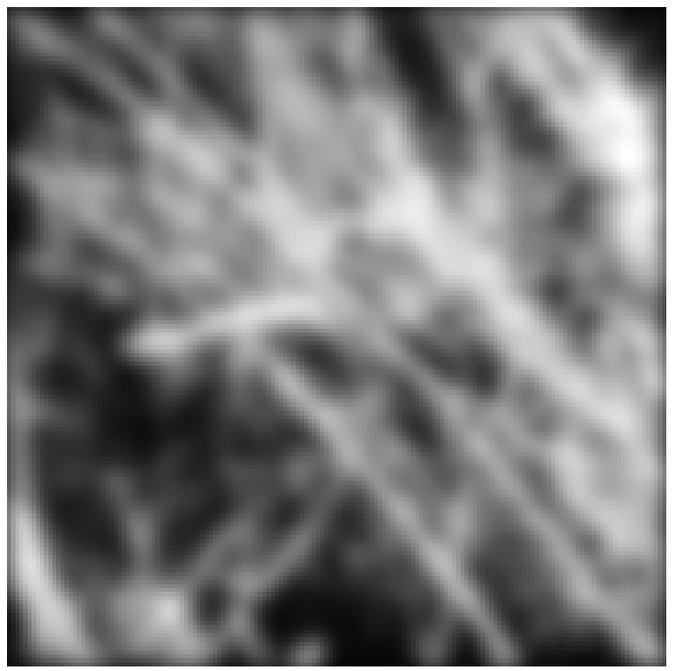
Smoothing with a Gaussian

TECHNISCHE UNIVERSITÄT DARMSTADT

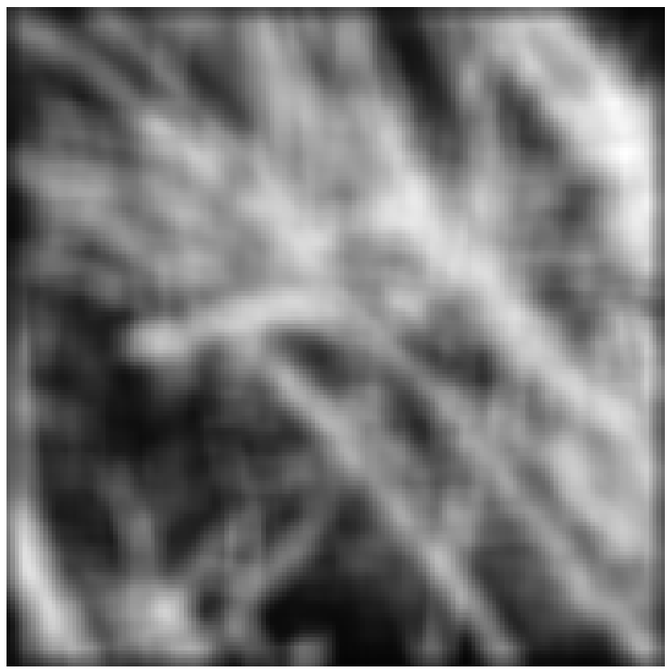
- Example:

Original image

Gaussian-filtered image



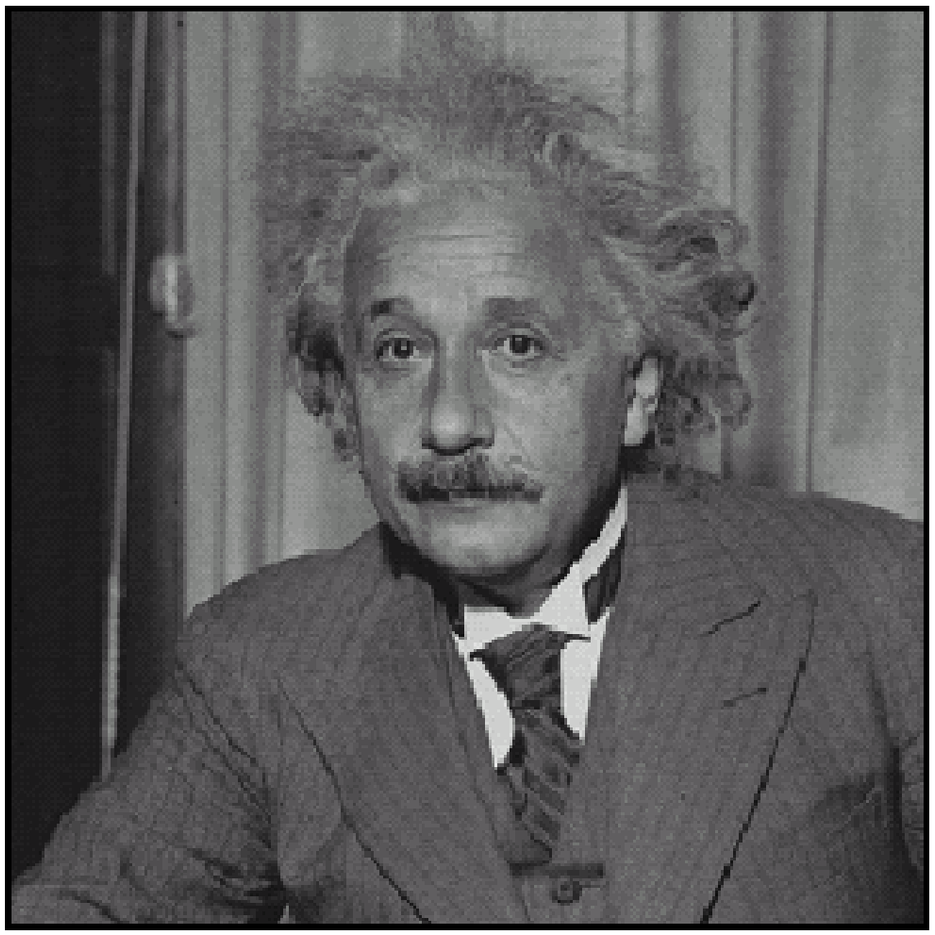
Box-filtered image



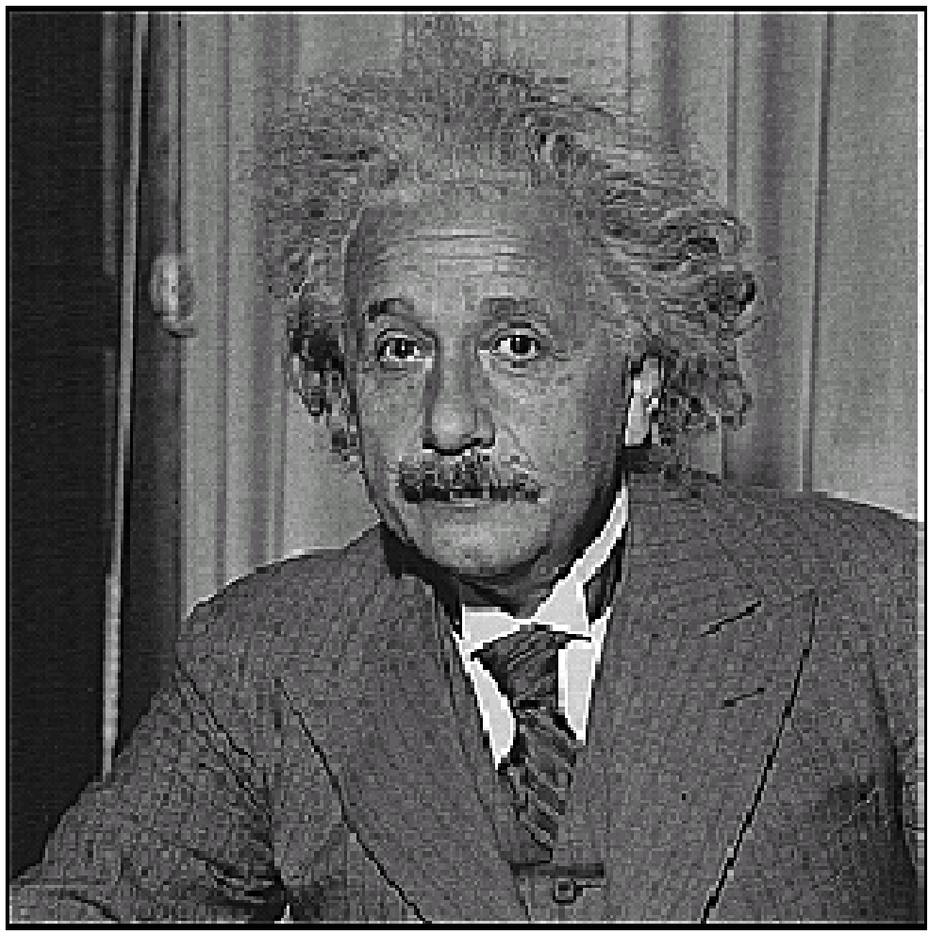
Only smoothing?

TECHNISCHE UNIVERSITÄT DARMSTADT

- We can also sharpen an image by amplifying what smoothing removes: $g_{\text{sharp}} = f + \gamma(f - h_{\text{blur}} * f)$



original



sharpened

[Bill Freeman]

Efficient Implementation

TECHNISCHE UNIVERSITÄT DARMSTADT

- Both, the box filter and the Gaussian filter are separable:
  - First convolve each row with a 1D filter
  - then convolve each column with a 1D filter

$$(f_x * f_y) * I = f_x * (f_y * I)$$

- Remember:
  - convolution is linear - associative and commutative
- Example: Separable box filter

$$f_x * f_y \qquad\qquad f_x \qquad\qquad f_y$$

$$\frac{1}{9} \cdot \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} = \frac{1}{3} \cdot \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline \end{array} * \frac{1}{3} \cdot \begin{array}{|c|}\hline 1 \\ \hline 1 \\ \hline 1 \\ \hline \end{array}$$

Example: Separable Gaussian

TECHNISCHE UNIVERSITÄT DARMSTADT

- Gaussian in x-direction

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

- Gaussian in y-direction

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

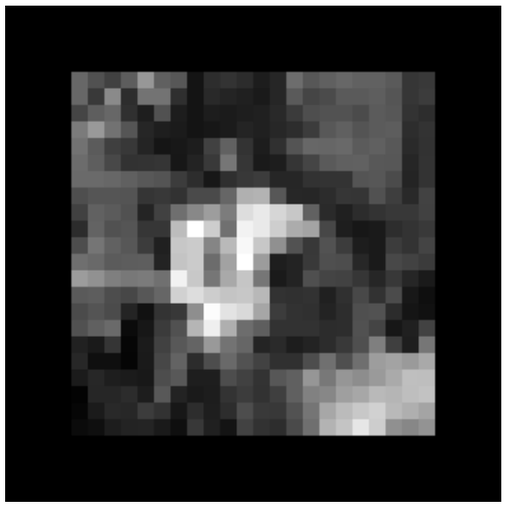
- Gaussian in both directions (product)

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

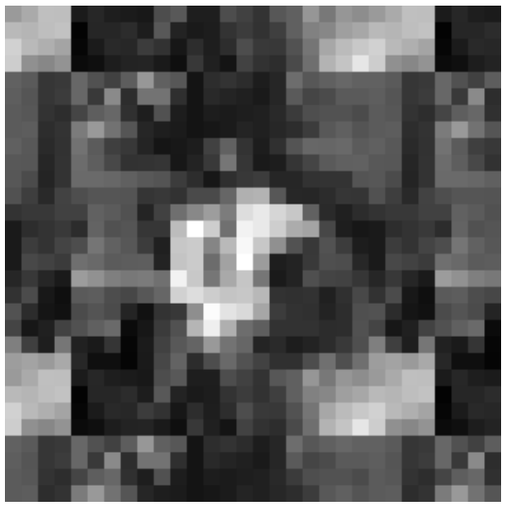
Boundary issues

TECHNISCHE UNIVERSITÄT DARMSTADT

- Need to be careful at the boundaries:
  - Boundary handling strategies

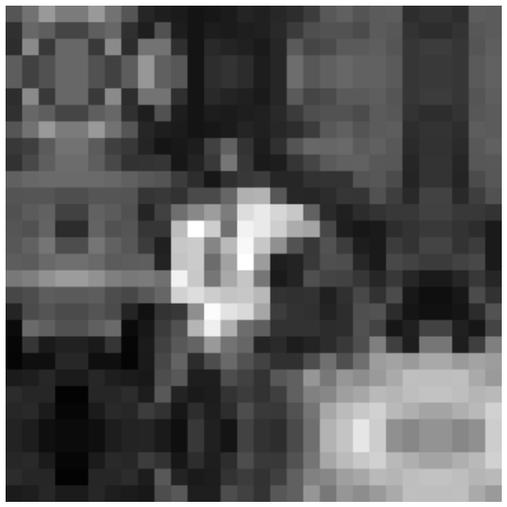


zero

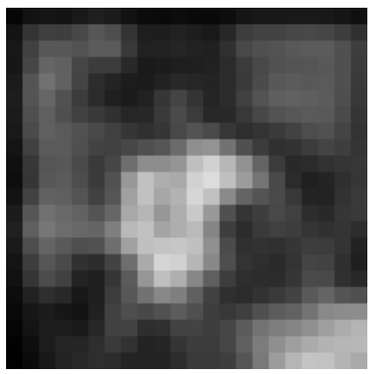


wrap

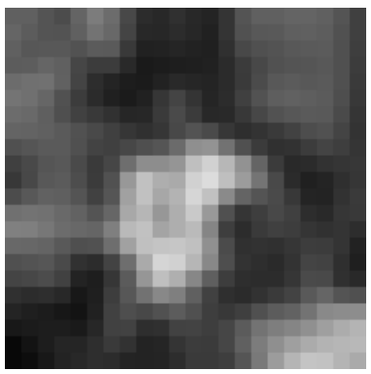
clamp



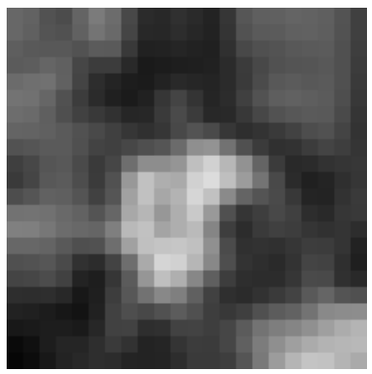
mirror



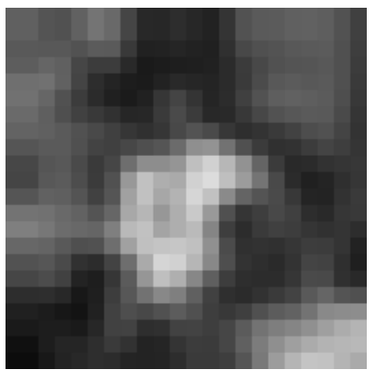
blurred zero



normalized zero



blurred clamp



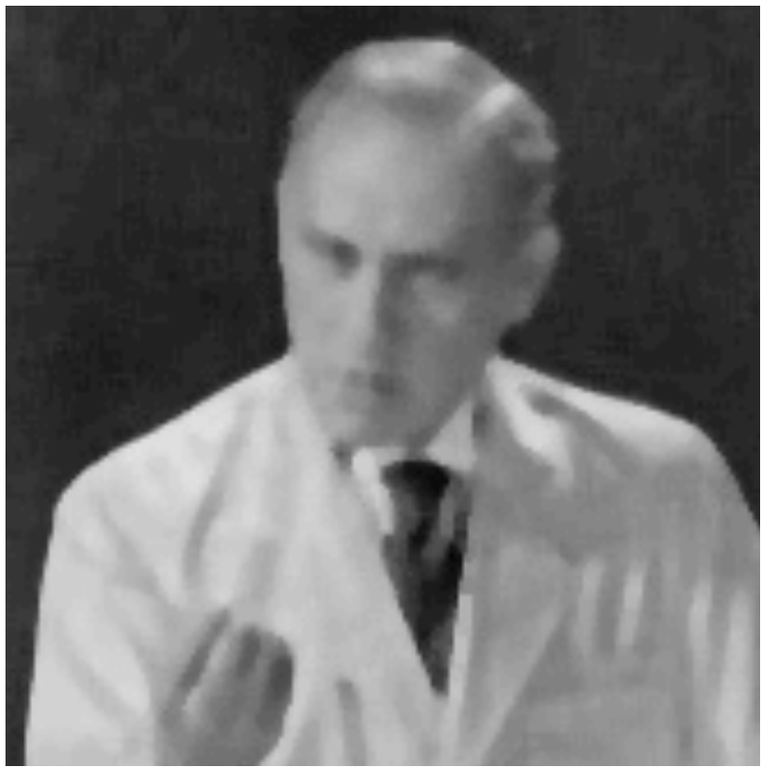
blurred mirror

Non-linear filtering

TECHNISCHE UNIVERSITÄT DARMSTADT

- There is a large variety of non-linear filters for noise removal.
- Simplest one: **Median filter**
  - Replace each pixel with the median in a neighborhood around it

Original



7x7 median filter

Special case: Binary Images

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Morphological operations:**
  - Perform convolution with a "structuring element" s
  - This is a binary mask (often circle or square)
  - Then perform thresholding to recover a binary image:

$$\theta(f,t) = \begin{cases} 1 & \text{if } f \geq t, \\ 0 & \text{else,} \end{cases}$$

original

dilation
$\theta(f * s, 1)$

erosion
$\theta(f * s, S)$

# of pixels in s

Today - Basics of Digital Image Processing

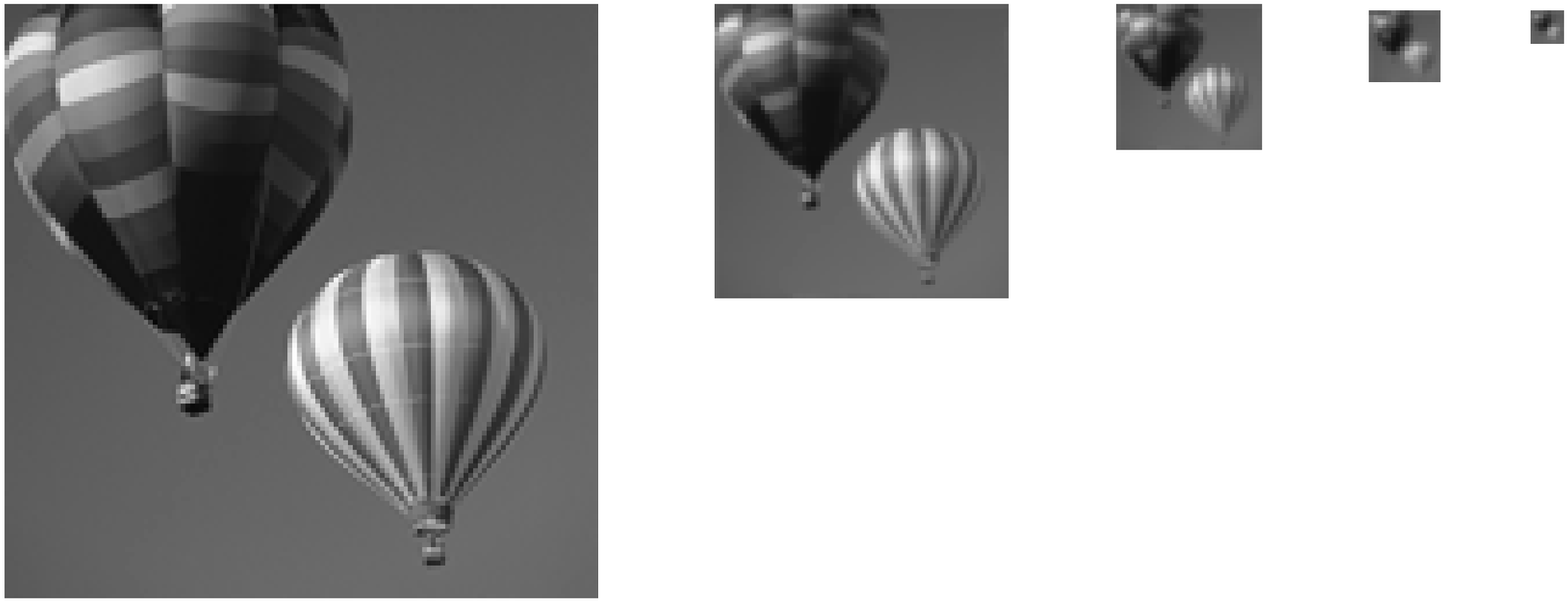
TECHNISCHE UNIVERSITÄT DARMSTADT

- Images
- Filtering
  - Linear filtering
  - Non-linear filtering & morphology
- Multi-scale image representation
  - Gaussian pyramid
  - Laplacian pyramid
- Edge detection
  - ‘Recognition using line drawings’
  - Image derivatives (1st and 2nd order)

Image pyramids

TECHNISCHE UNIVERSITÄT DARMSTADT

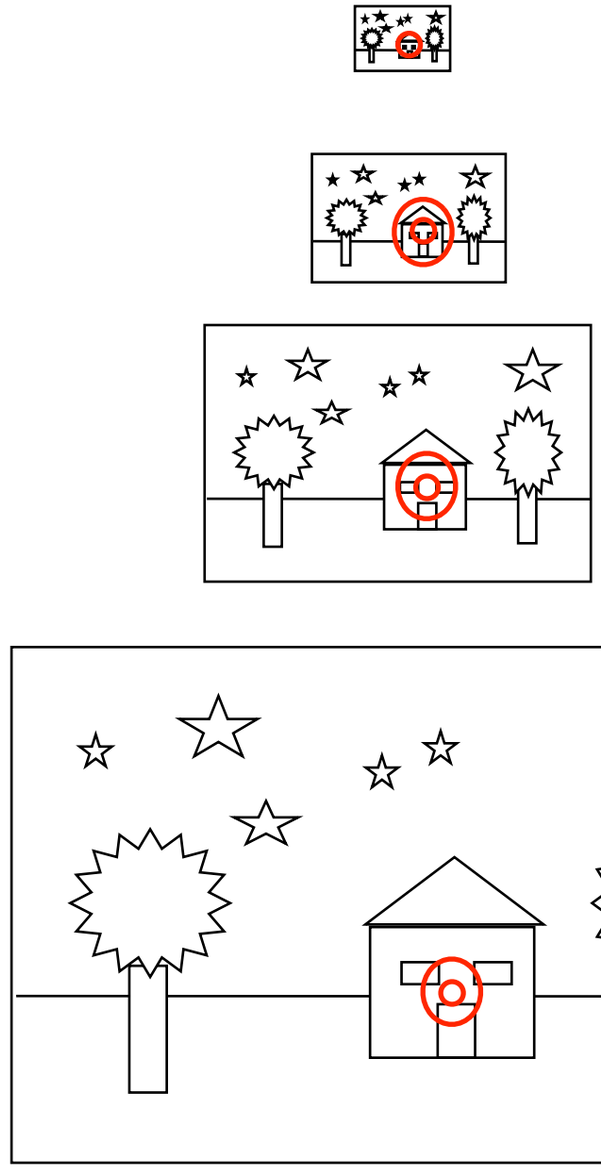
- Represent images at multiple scales (resolutions)
  - Stacked up like a pyramid.
  - E.g., Gaussian pyramid:



high resolution → low resolution

From Irani & Basri

Motivation: Search



search

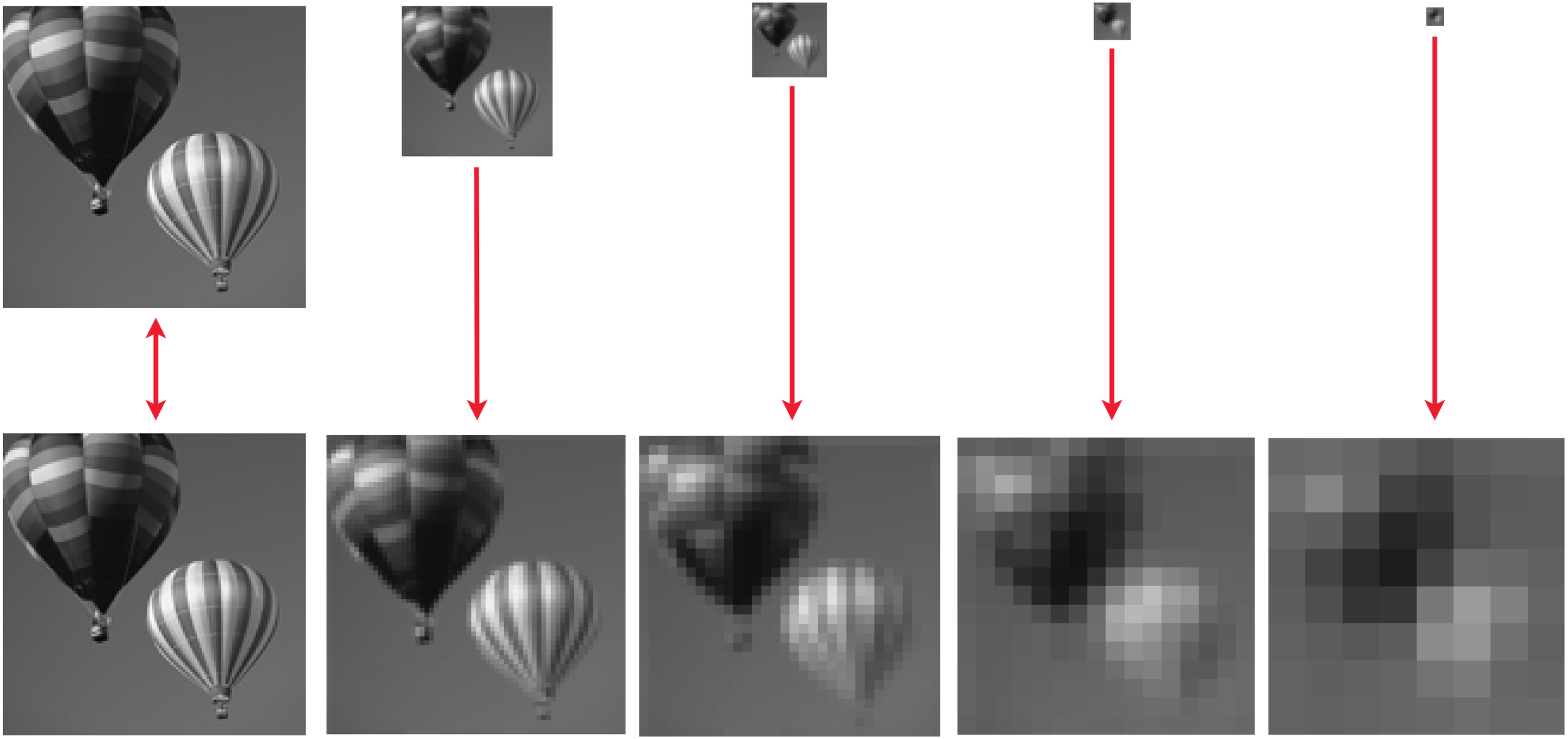
search

search

search

From Irani & Basri

Gaussian pyramids

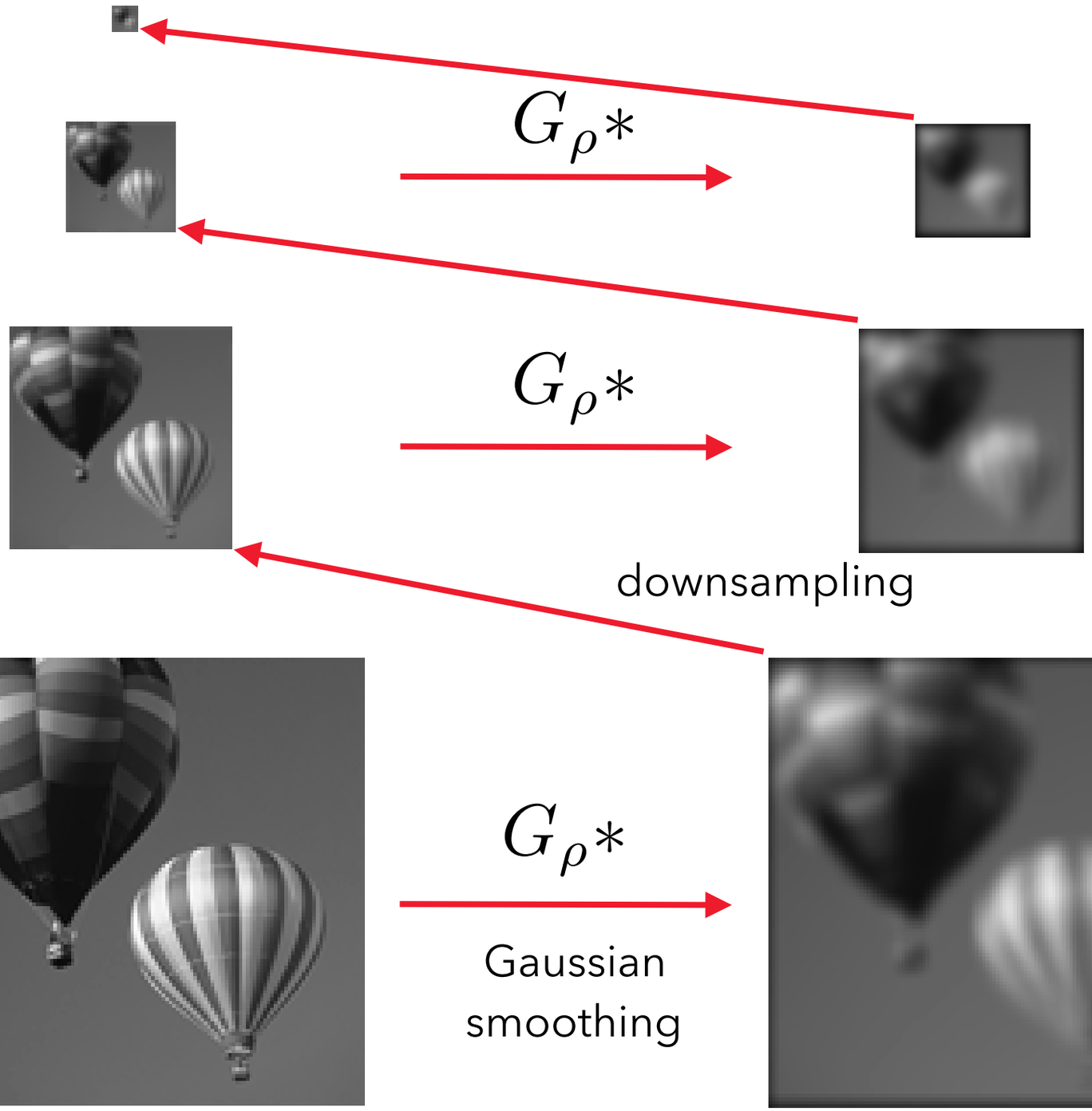


From Irani & Basri

Another Example

- A bar
  - in the big images is a hair (on the zebra's nose)
  - in smaller images, a stripe
  - in the smallest image, the animal's nose

Gaussian pyramids

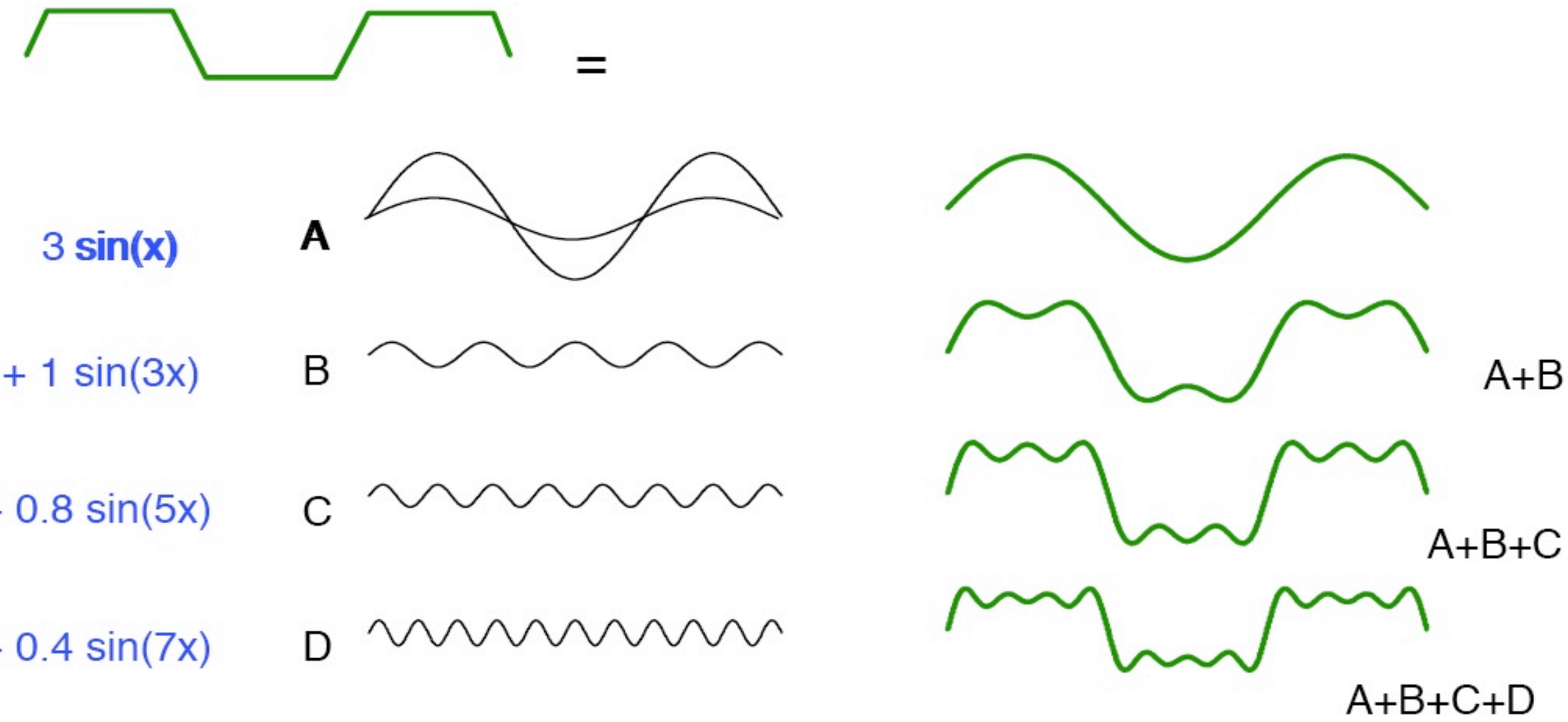


- Which information is preserved over ‘scales’?
- Which information is lost over ‘scales’?

Fourier Transform in Pictures

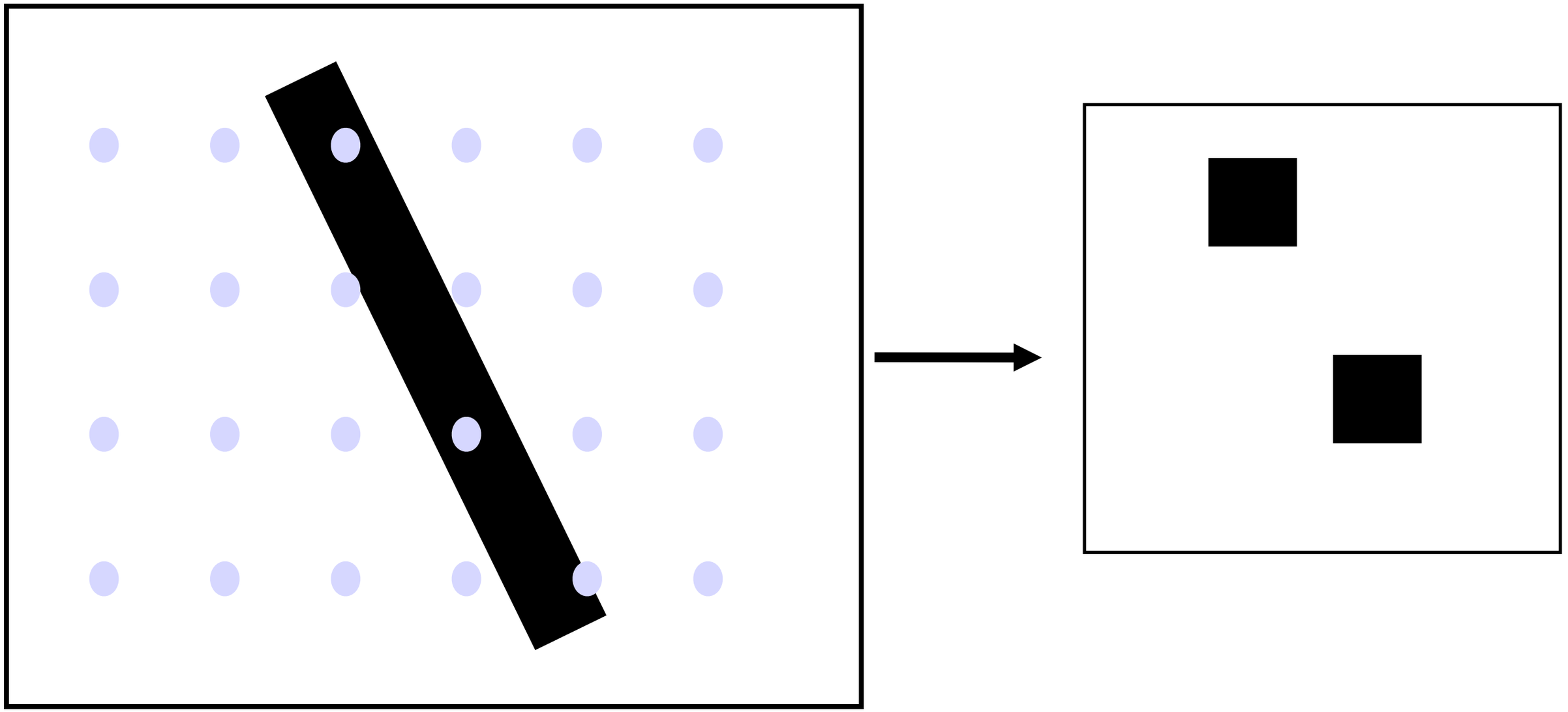
TECHNISCHE UNIVERSITÄT DARMSTADT

- A tiny bit about Fourier transform to talk about spatial frequencies… (refer to HCS)



Subsampling

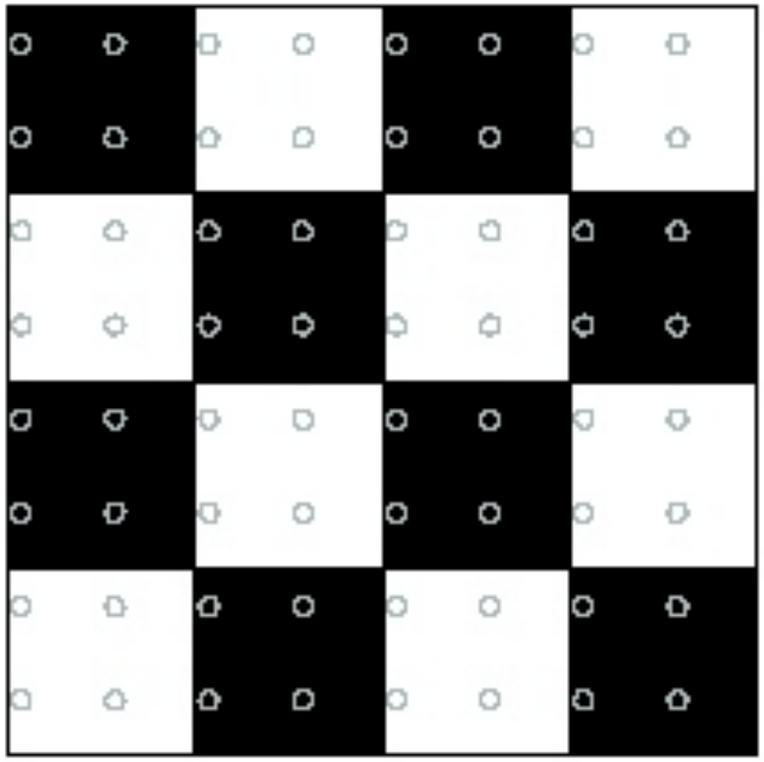
TECHNISCHE UNIVERSITÄT DARMSTADT

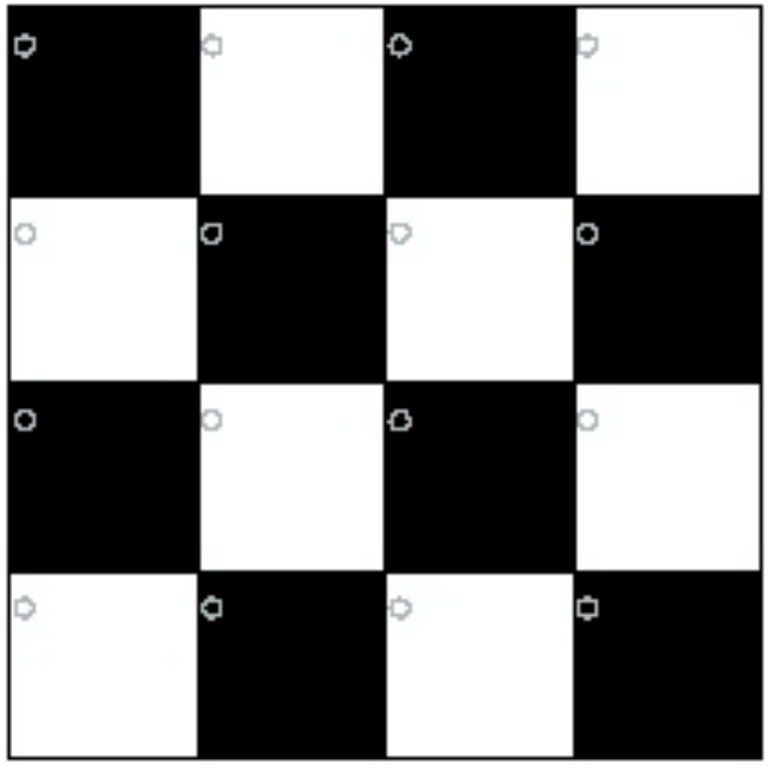


Aliasing

TECHNISCHE UNIVERSITÄT DARMSTADT

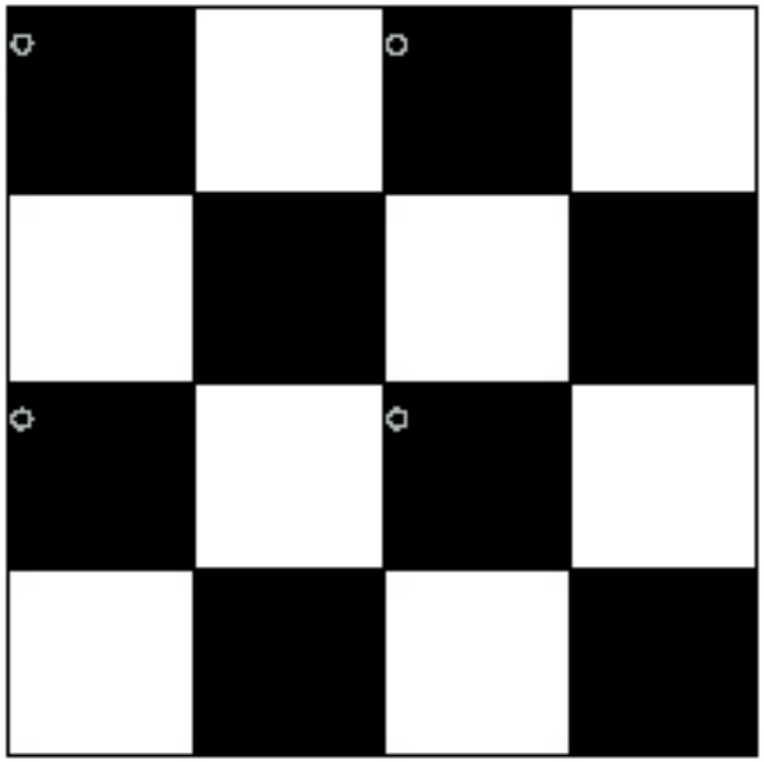
- Cannot shrink an image by taking every second pixel.
  - High frequencies cannot be represented anymore.
- If we do that anyway, characteristic errors appear:
  - Spatial frequencies are misinterpreted (**aliasing**)
  - Typically, small phenomena look bigger; fast phenomena can look slower.
  - Common examples:
    - Wagon wheels rolling the wrong way in movies.
    - Checkerboards misrepresented in ray tracing.
    - Striped shirts look strange on color television.

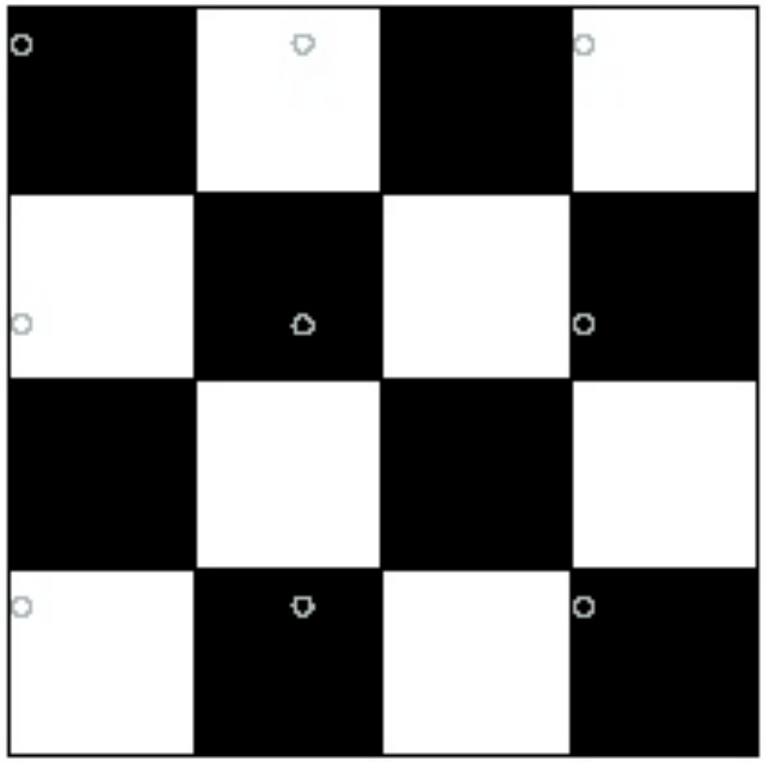




Resample the checkerboard by taking one sample at each circle.

In the case of the top left board, new representation is reasonable.

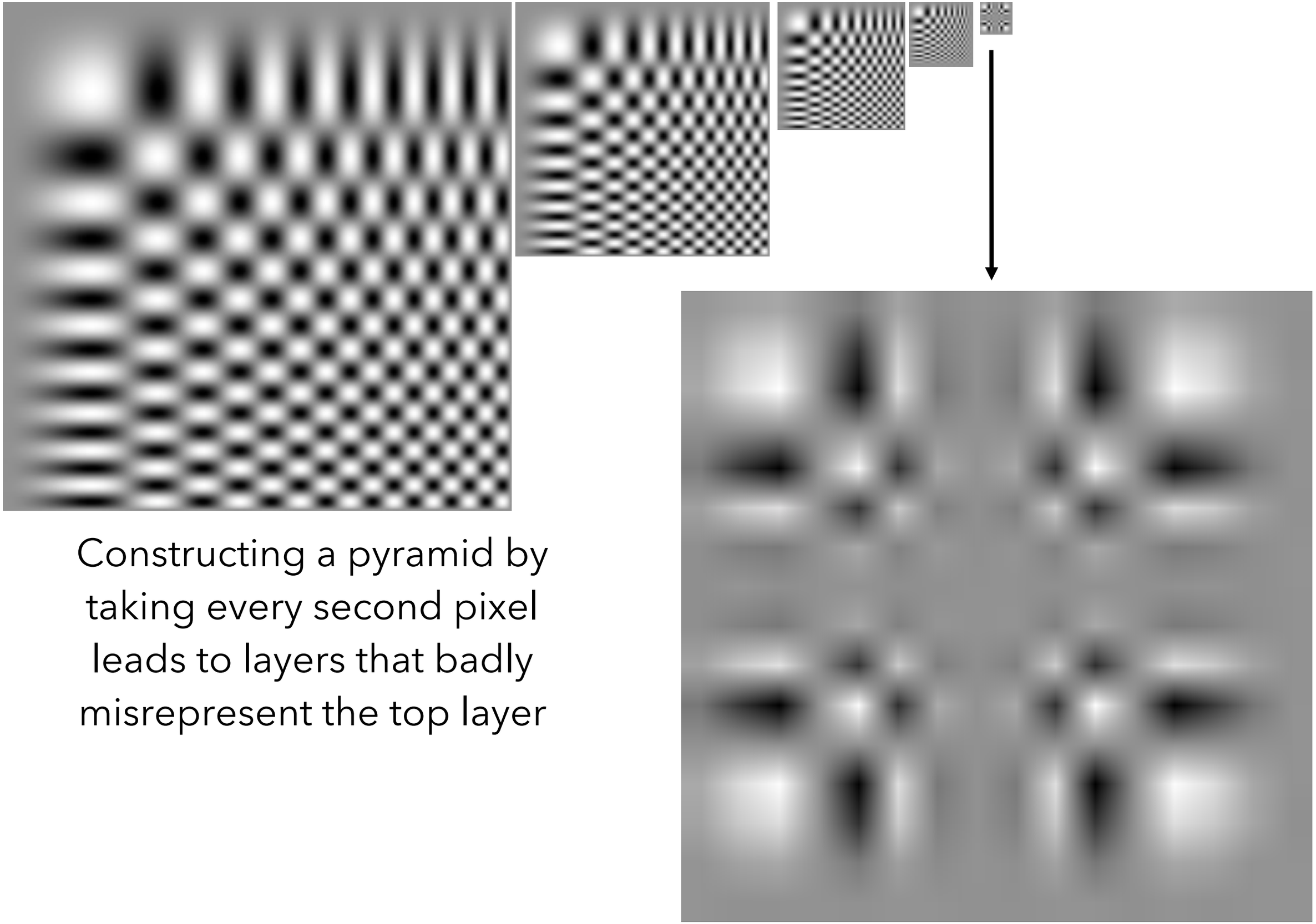




Top right also yields a reasonable representation.

Bottom left is all black (dubious) and bottom right has checks that are too big.

From [Ponce & Forsyth]



Constructing a pyramid by taking every second pixel leads to layers that badly misrepresent the top layer

From [Ponce & Forsyth]

Today - Basics of Digital Image Processing

TECHNISCHE UNIVERSITÄT DARMSTADT

- Images
- Filtering
  - Linear filtering
  - Non-linear filtering & morphology
- Multi-scale image representation
  - Gaussian pyramid
  - Laplacian pyramid
- Edge detection
  - ‘Recognition using line drawings’
  - Image derivatives (1st and 2nd order)

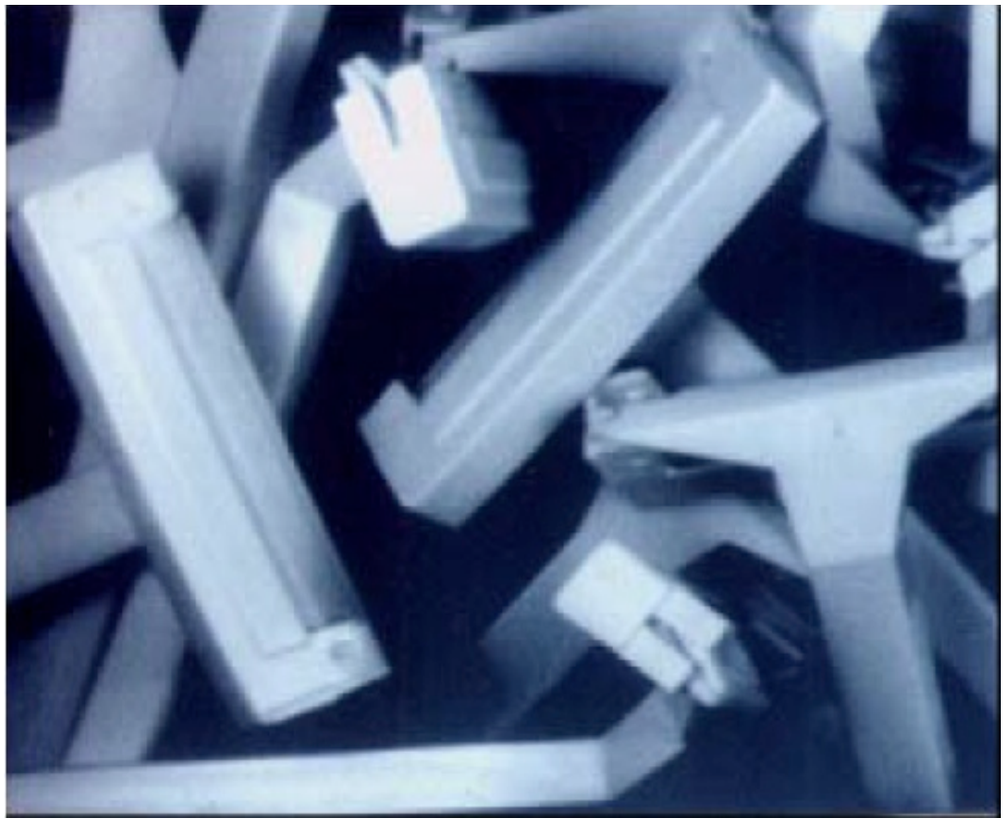
Line Drawings: Good Starting Point for Recognition?

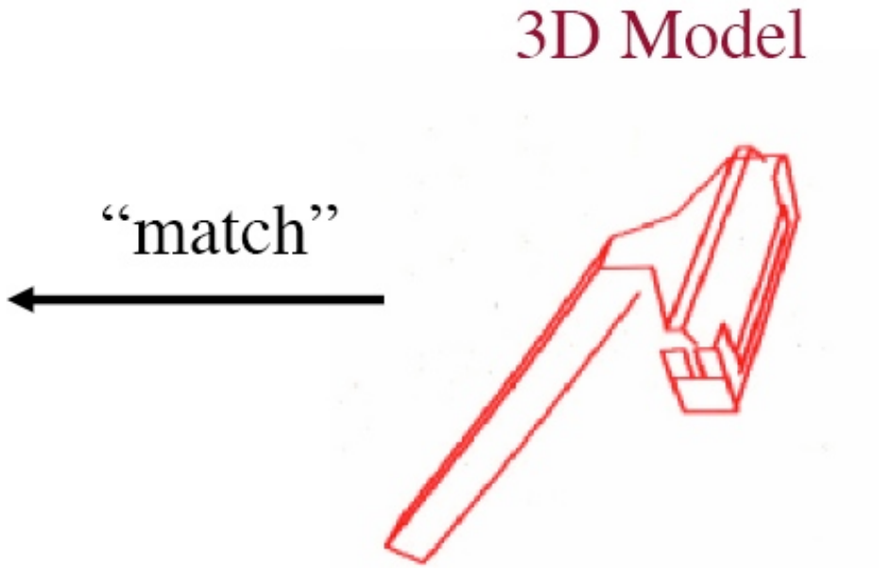
TECHNISCHE UNIVERSITÄT DARMSTADT

- Next time we will get more serious about recognition.
- Let's start...

Example of Recognition & Localization

- Simple approach [David Lowe]



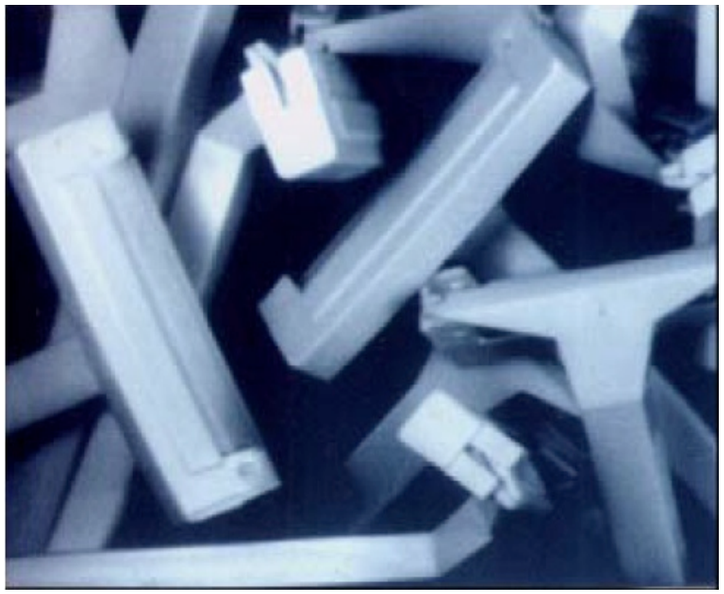


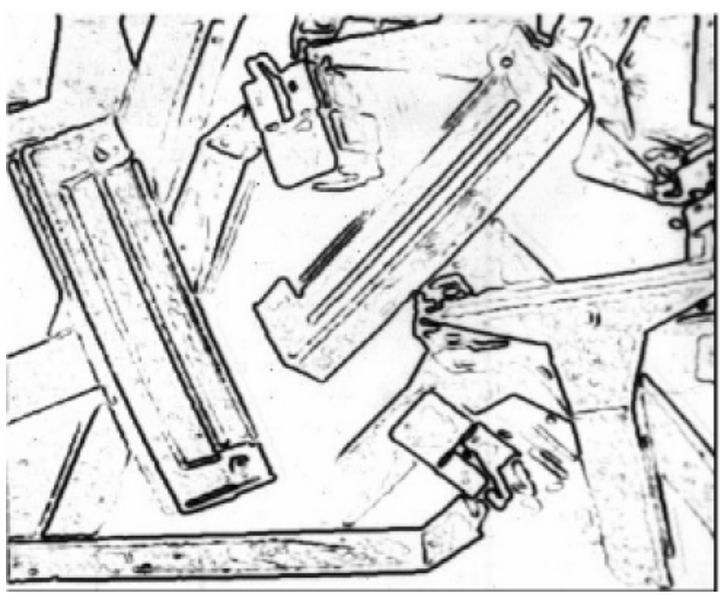
Parameters: 3D position and orientation

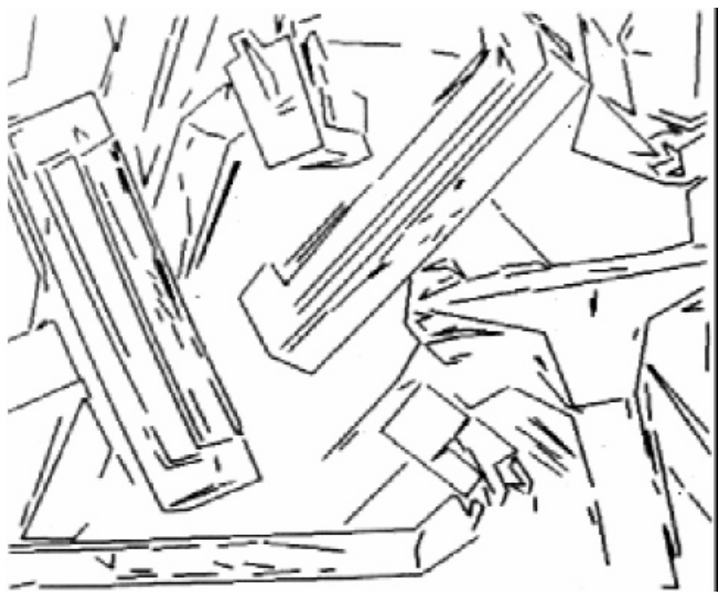
Example of Recognition & Localization

TECHNISCHE UNIVERSITÄT DARMSTADT

- Simple approach [David Lowe]
  - 1. “Filter” image to **find brightness changes**
  - 2. **“Fit” lines** to the raw measurements



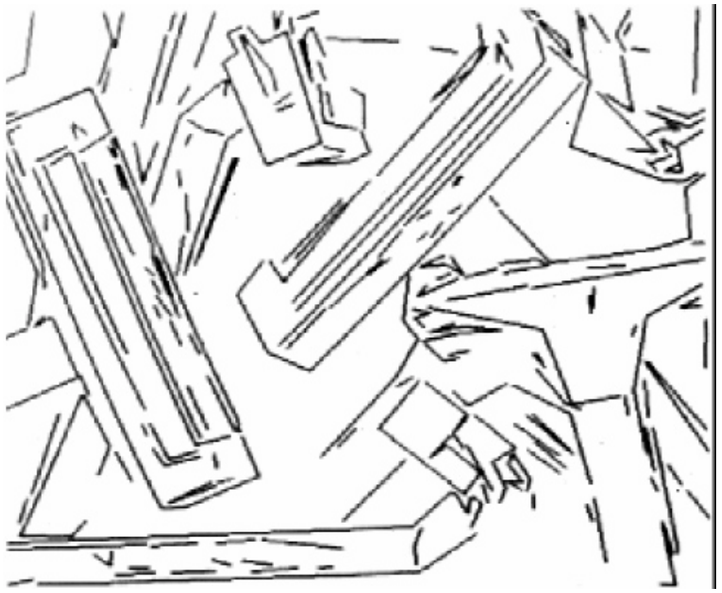


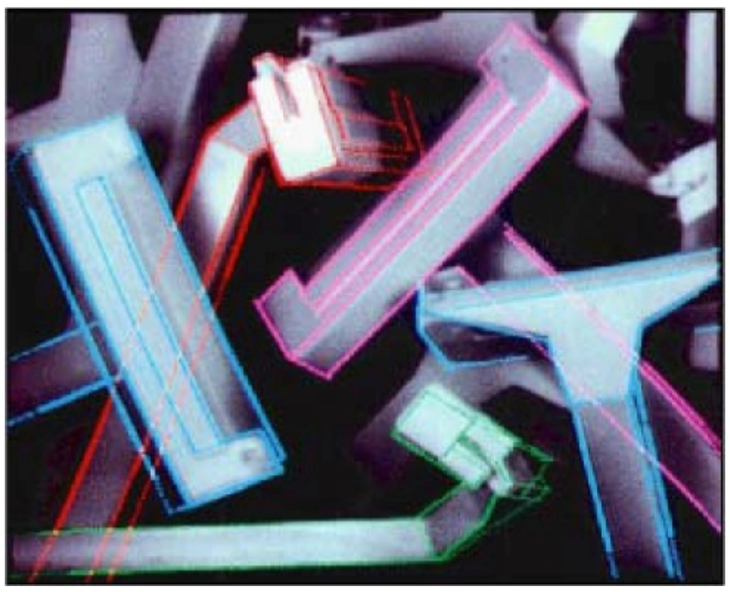


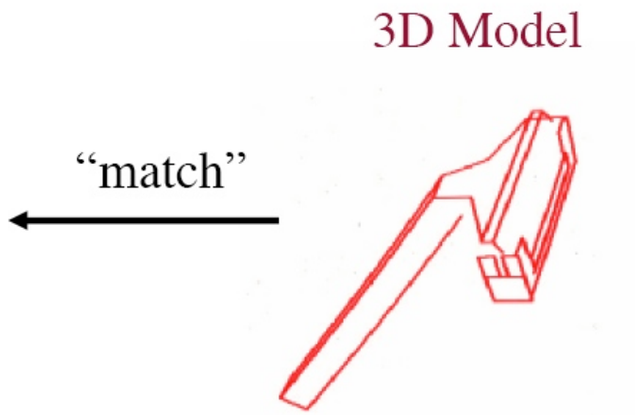
Example of Recognition & Localization

TECHNISCHE UNIVERSITÄT DARMSTADT

- Simple approach [David Lowe]
  - 3. "Project" model into the image and **"match" to lines** (solving for 3D pose)



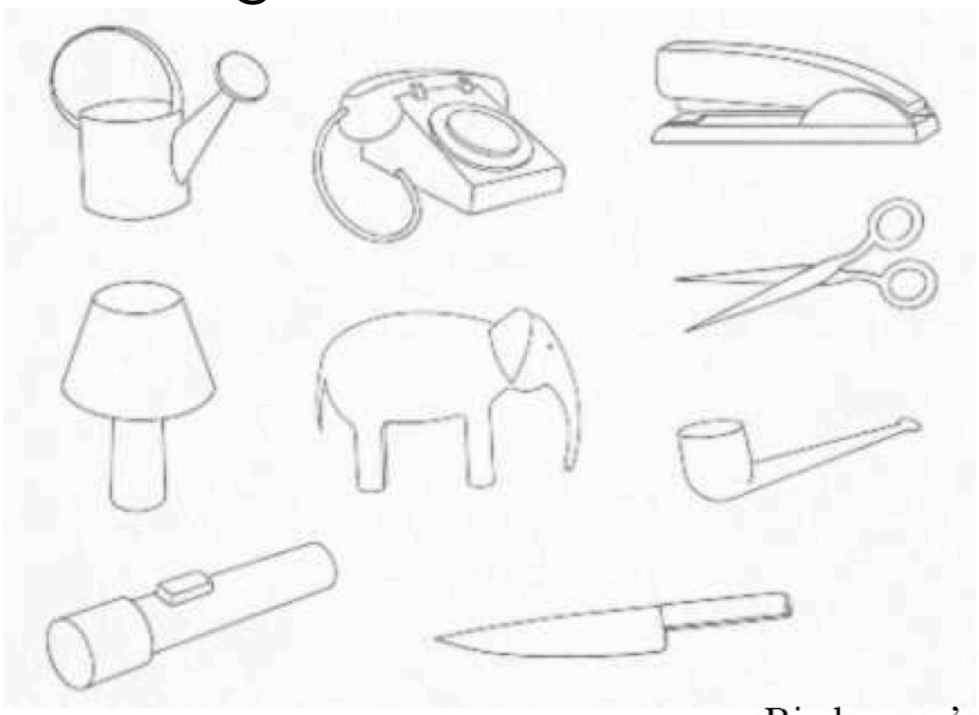




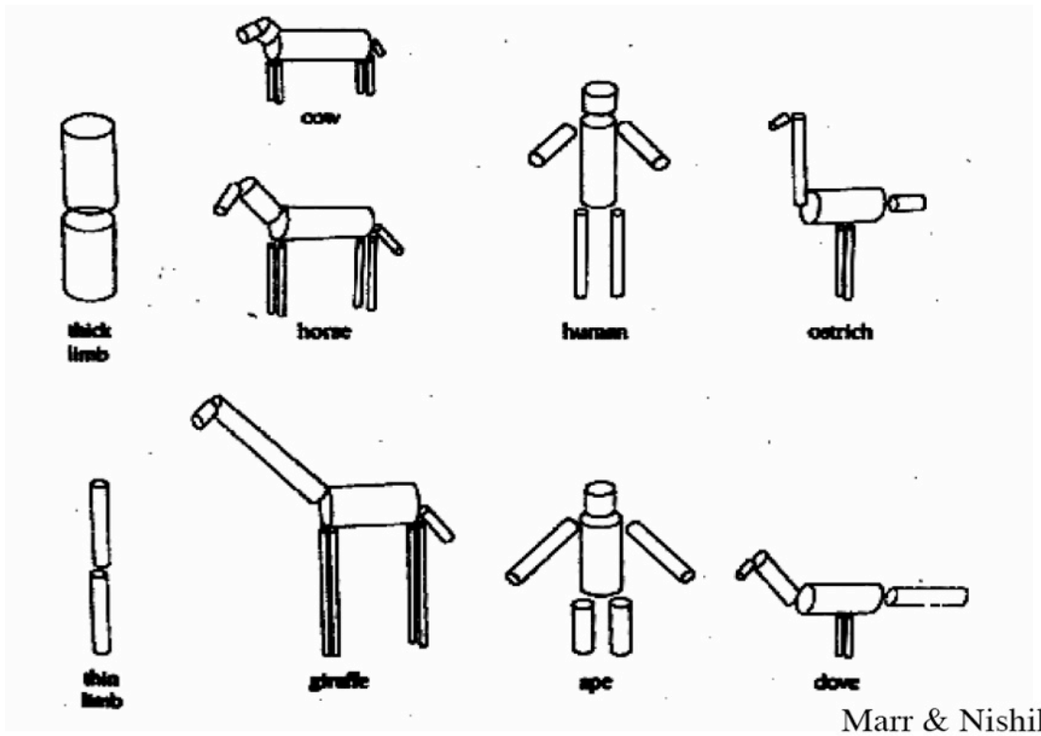
Object Models

TECHNISCHE UNIVERSITÄT DARMSTADT

- (Mostly) Historical idea & approach
  - Matching of models (wire-frame/geons/generalized cylinders...) to edges and lines



Biederman's Geons



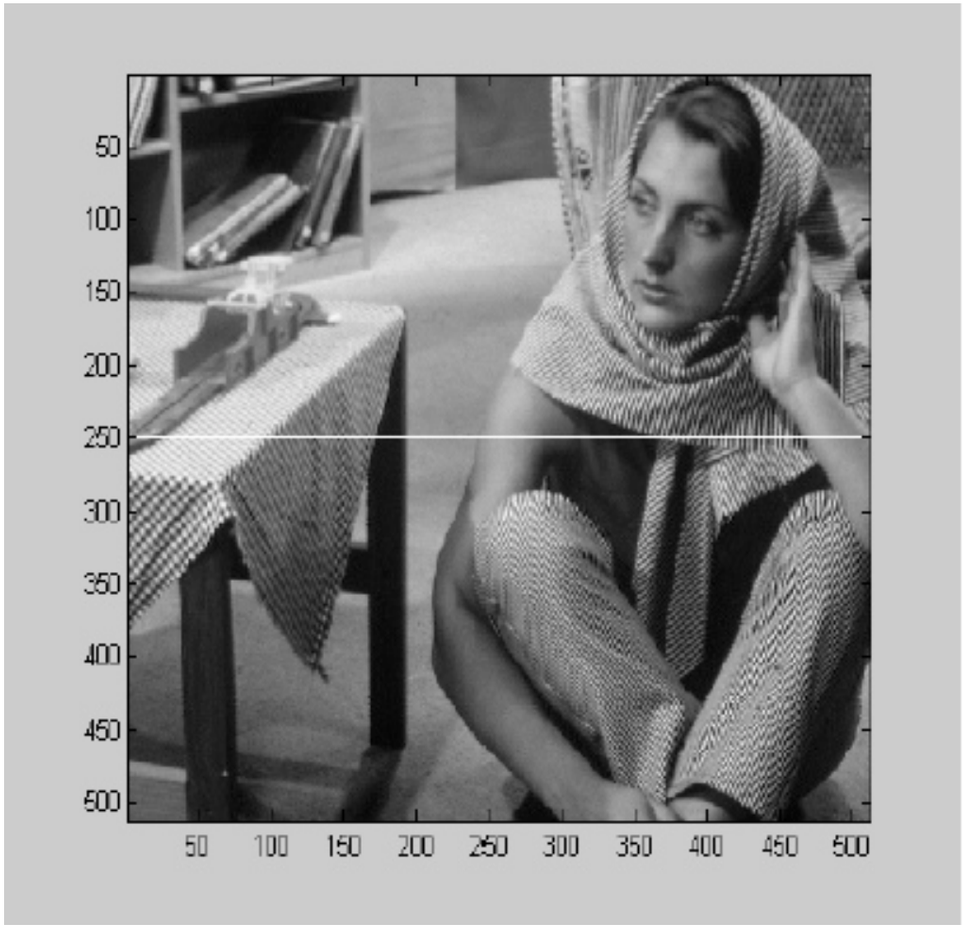
Marr & Nishihara

- The "only" remaining problem to solve is:
  - **reliably extract lines & edges** that can be matched to these models...

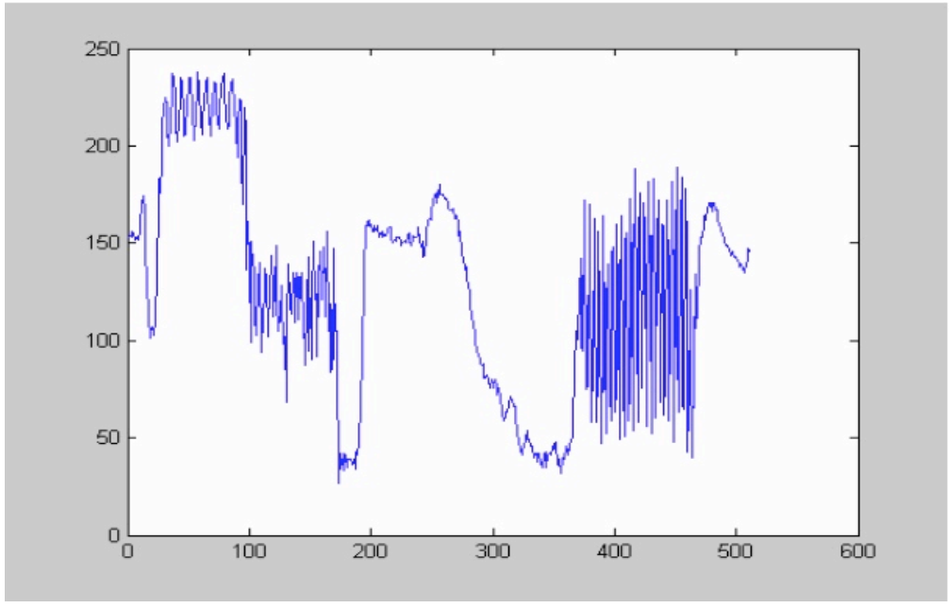
Image scanline

TECHNISCHE UNIVERSITÄT DARMSTADT

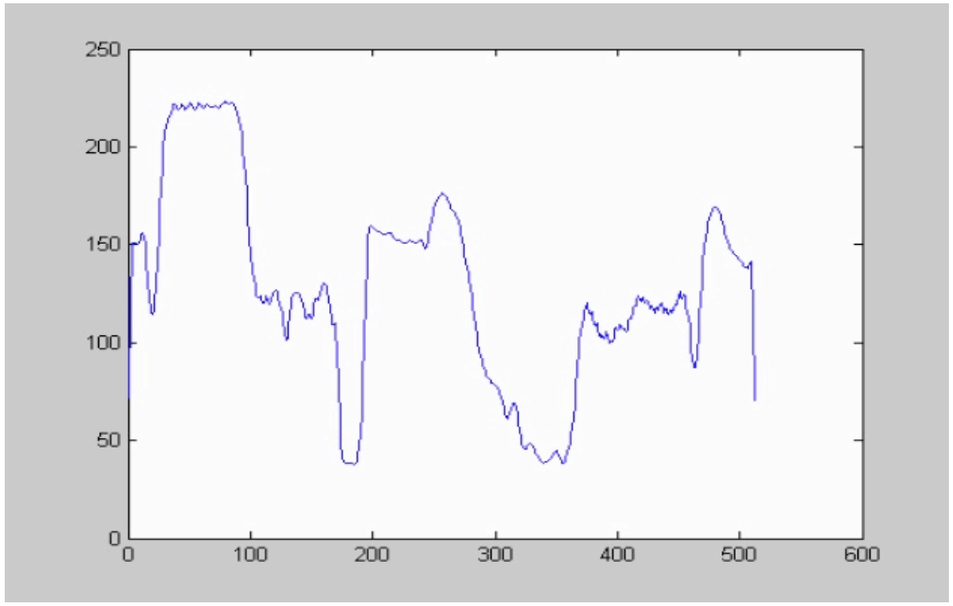
- Barbara image:
  - entire image



line 250:



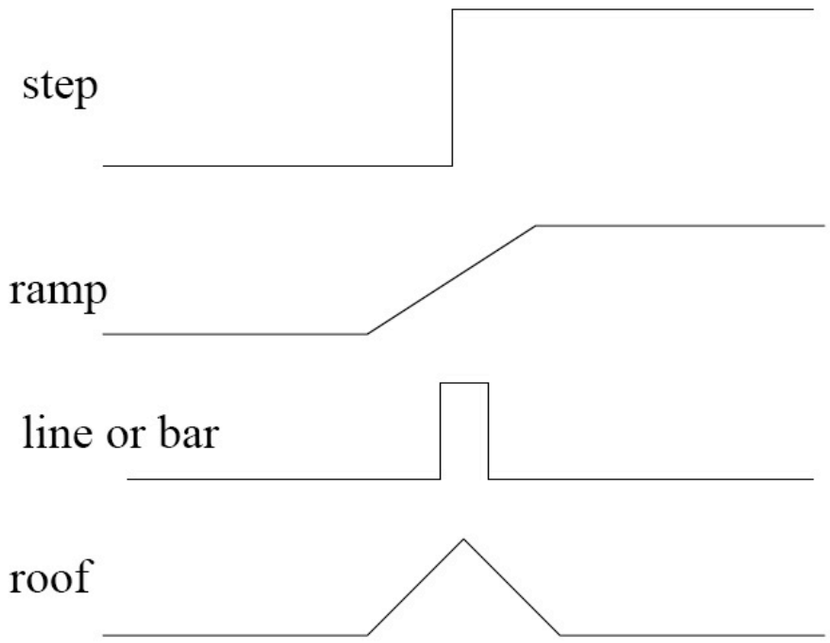
line 250 smoothed with a Gaussian:



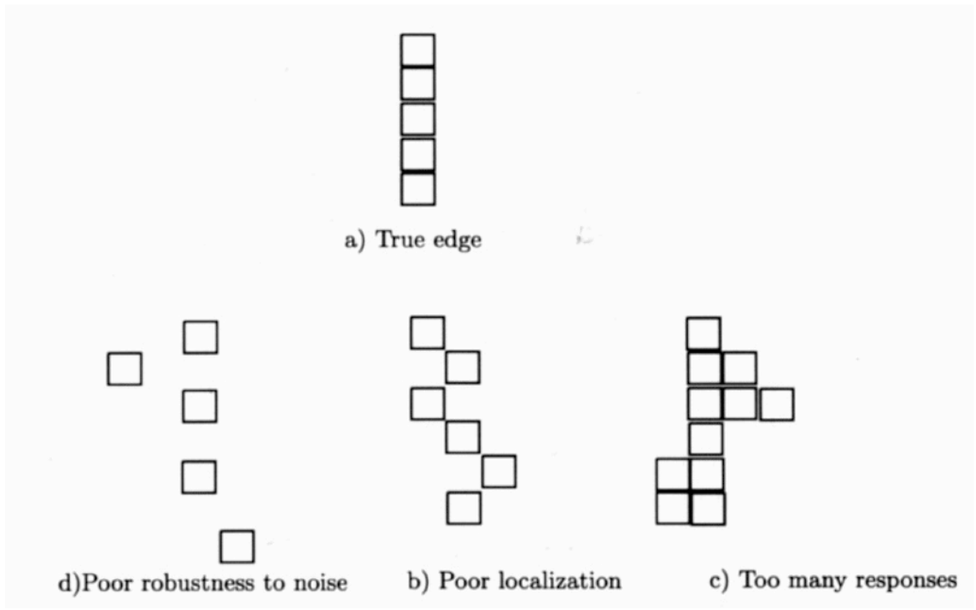
What are "edges" (1D)

TECHNISCHE UNIVERSITÄT DARMSTADT

- Idealized edge types:



- Goals of edge detection:
  - **good detection:** filter responds to edge, not to noise
  - **good localization:** detected edge near true edge
  - **single response:** one per edge

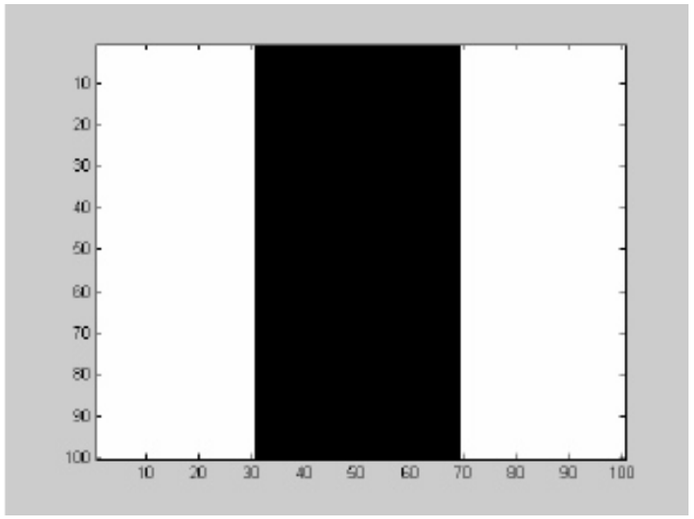


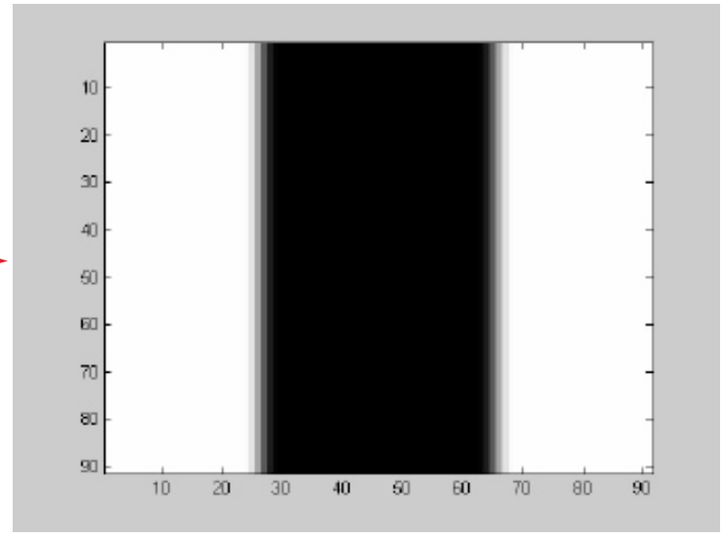
Edges

TECHNISCHE UNIVERSITÄT DARMSTADT

- correspond to fast changes
- where the magnitude of the derivative is large

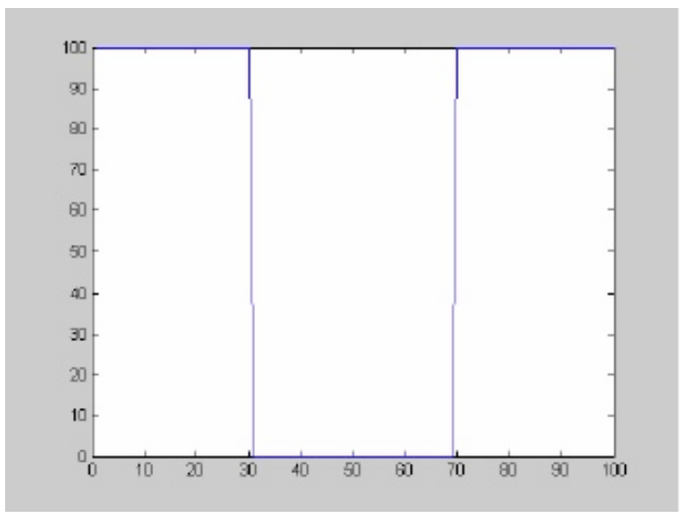
"image" of 2 step-edges

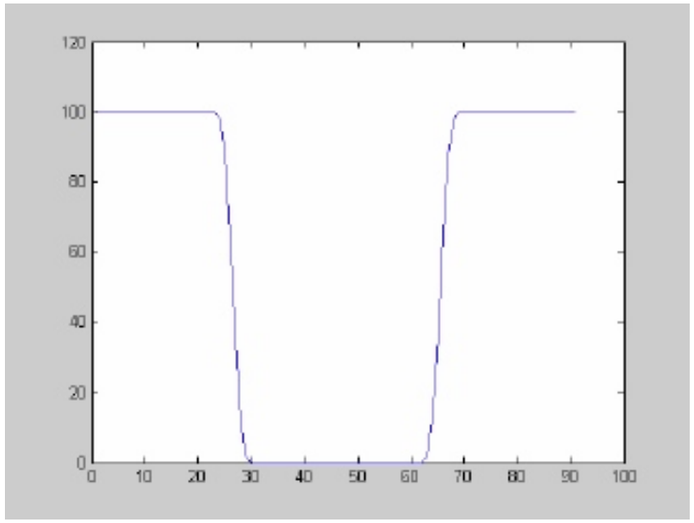




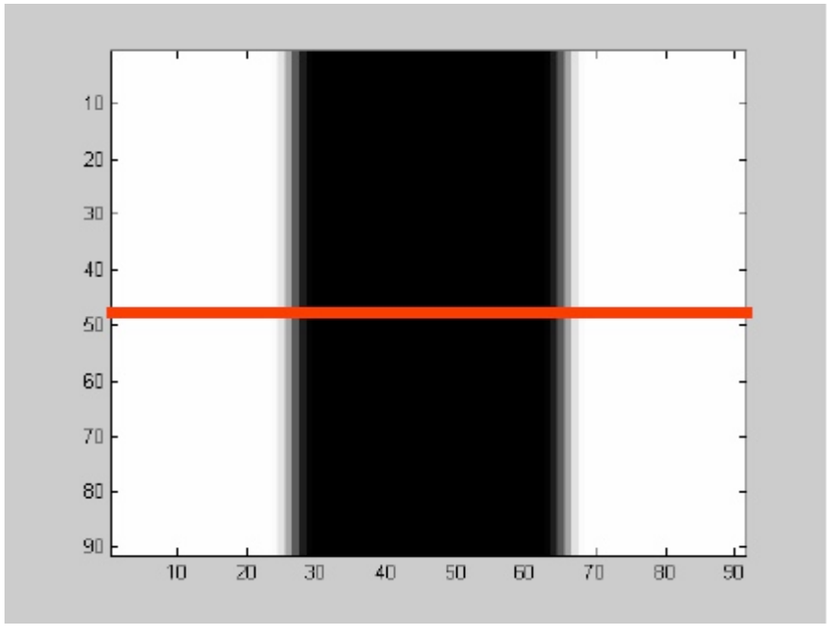
smoothing

single line of "image"

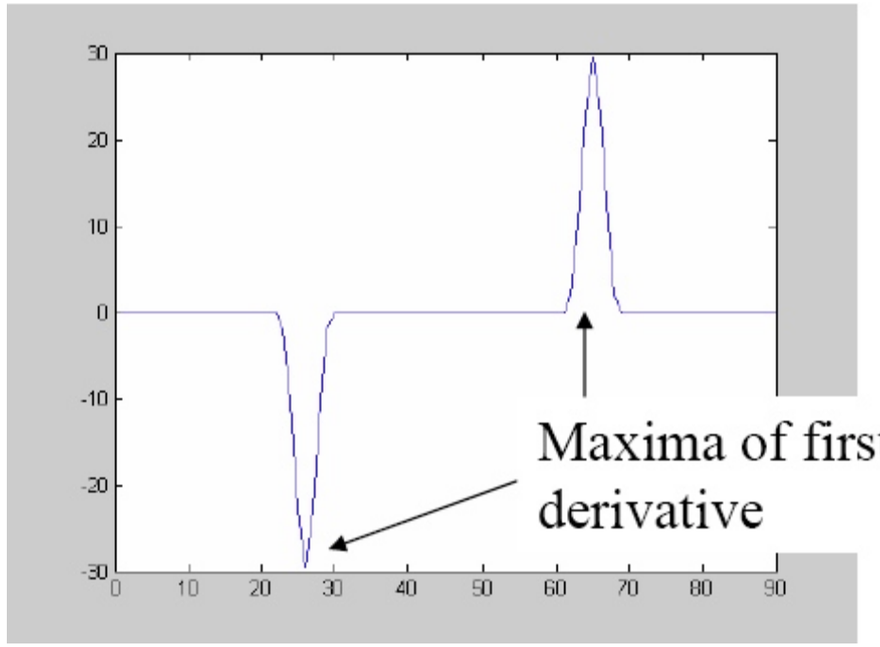


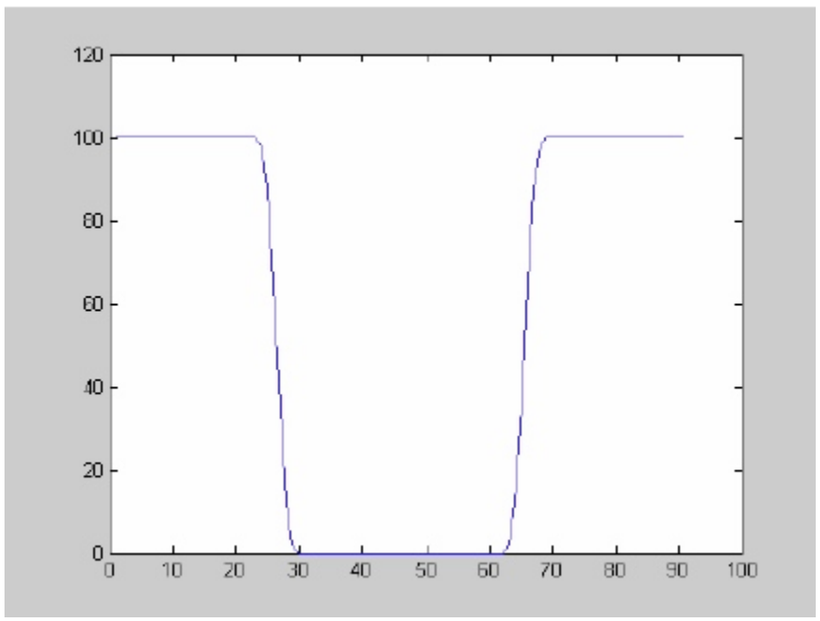


Edges & Derivatives…

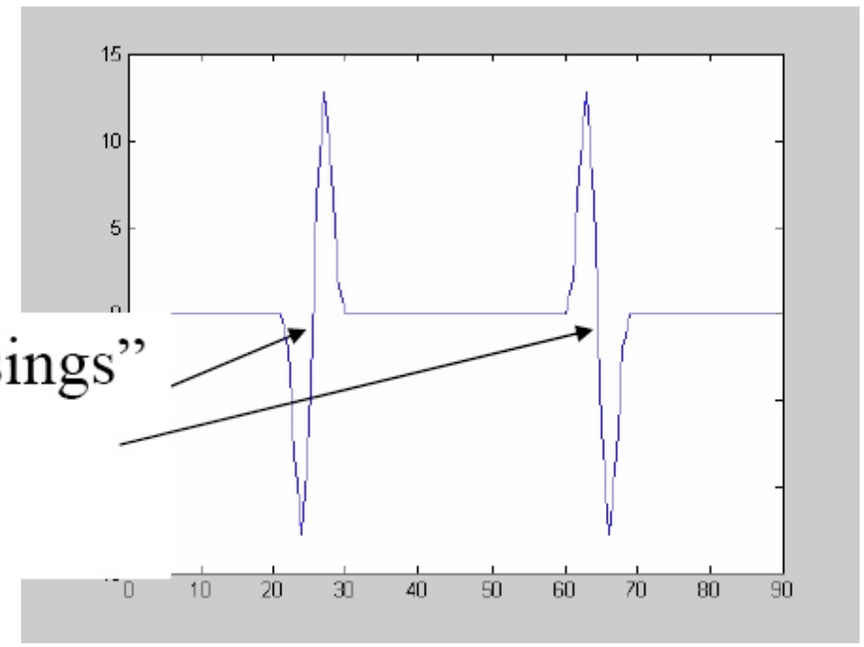


1st derivative





2nd derivative



Compute Derivatives

TECHNISCHE UNIVERSITÄT DARMSTADT

$$\frac{d}{dx} f(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \approx f(x+1) - f(x)$$

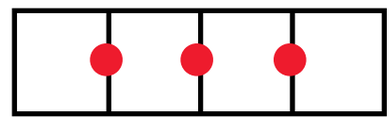
- We can implement this as a linear filter:
  - forward differences: (convolution filter)

1	−1

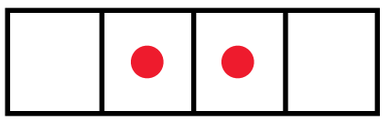
  - or central differences: $\frac{1}{2} \cdot$

1	0	−1

Where is the derivative computed?



in between pixels



at pixels

Edge Detection

TECHNISCHE UNIVERSITÄT DARMSTADT

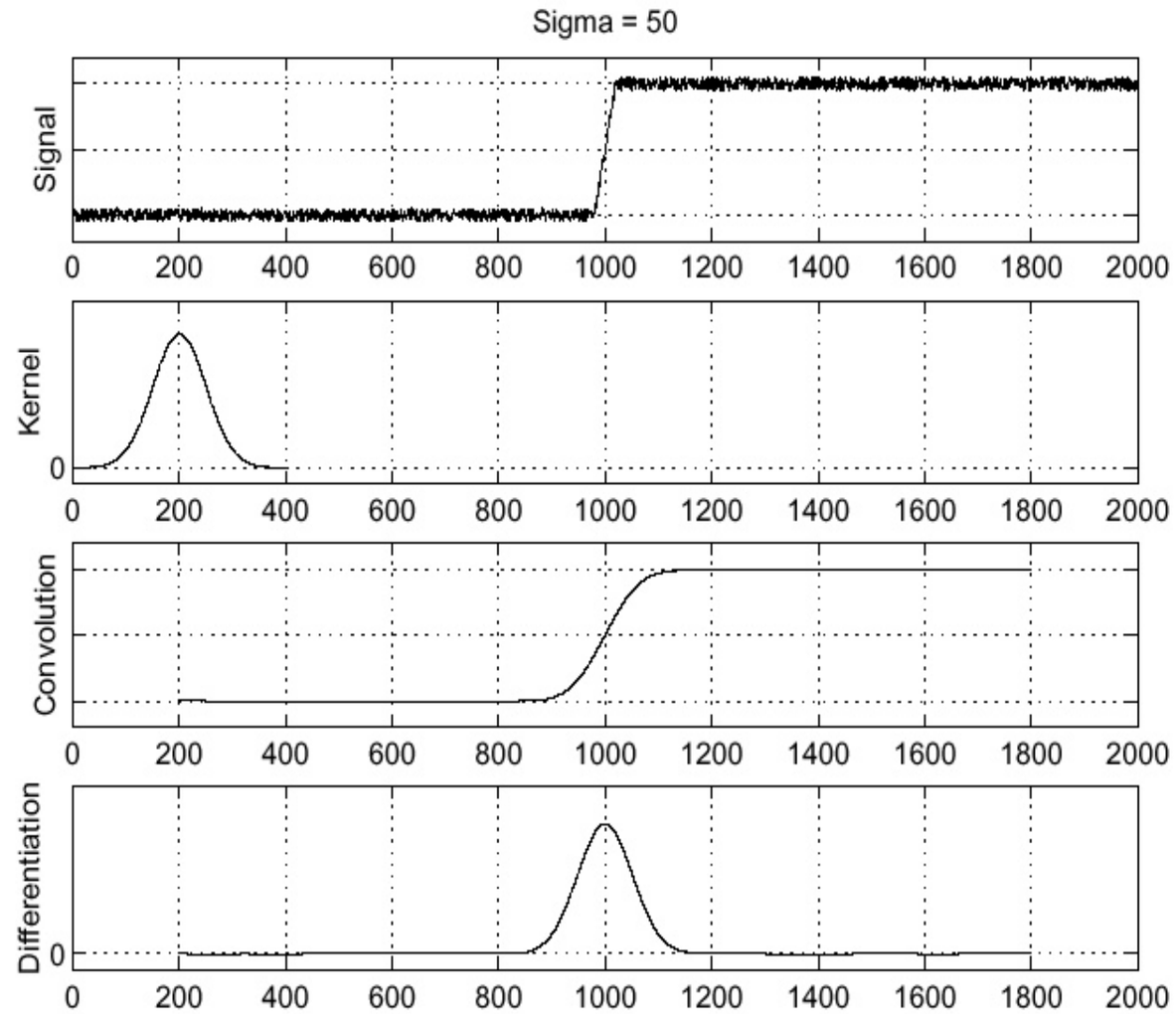
- based on 1st derivative:
  - **smooth** with Gaussian
  - calculate **derivative**
  - finds its **optima**

f

g

$g * f$

$$\frac{\mathrm{d}}{\mathrm{d}x}(g * f)$$



Edge Detection

TECHNISCHE UNIVERSITÄT DARMSTADT

- Simplification:

$$\frac{\mathrm{d}}{\mathrm{d}x}(g * f) = \left(\frac{\mathrm{d}}{\mathrm{d}x} g\right) * f$$

  - saves one operation

derivative of Gaussian

$$\frac{\mathrm{d}}{\mathrm{d}x} g$$

$$\left(\frac{\mathrm{d}}{\mathrm{d}x} g\right) * f$$

f

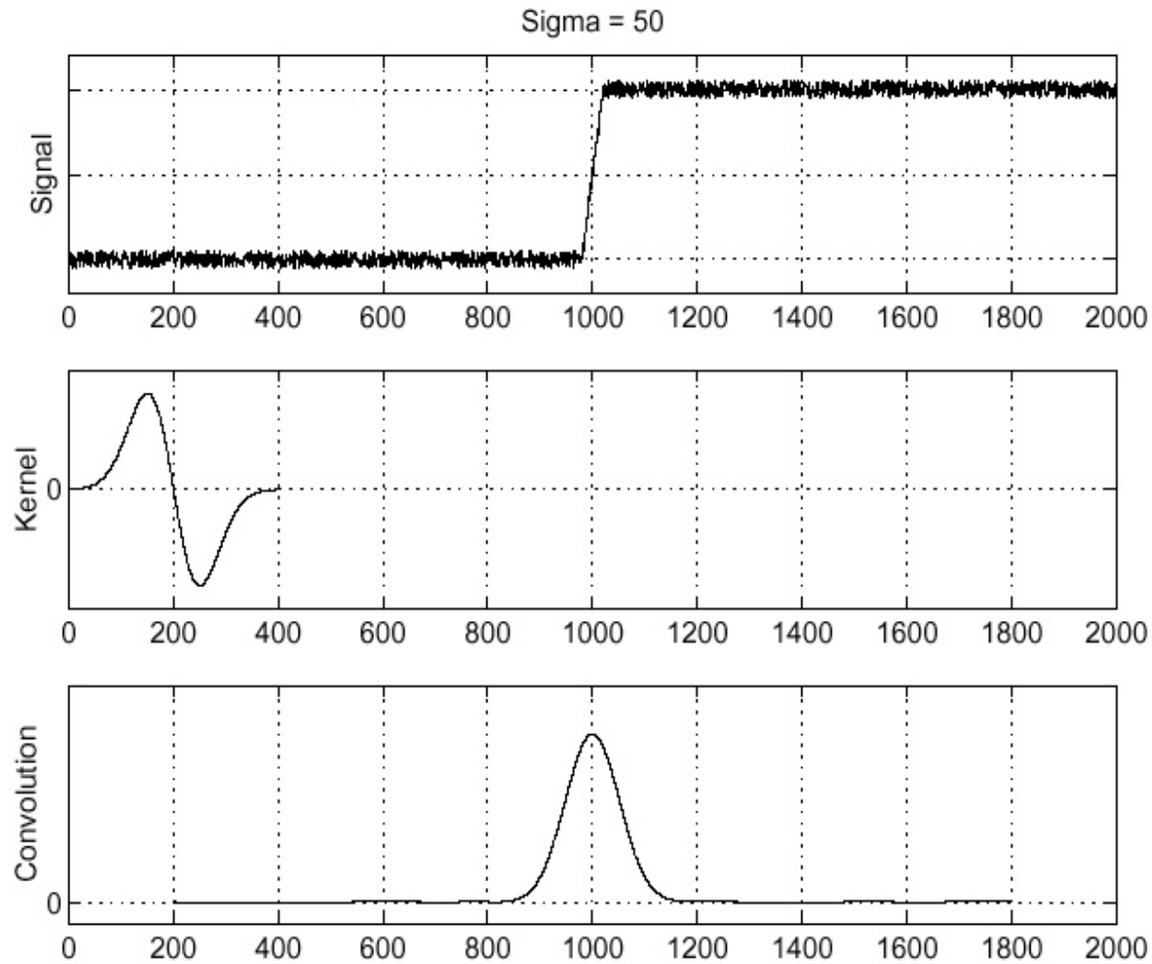
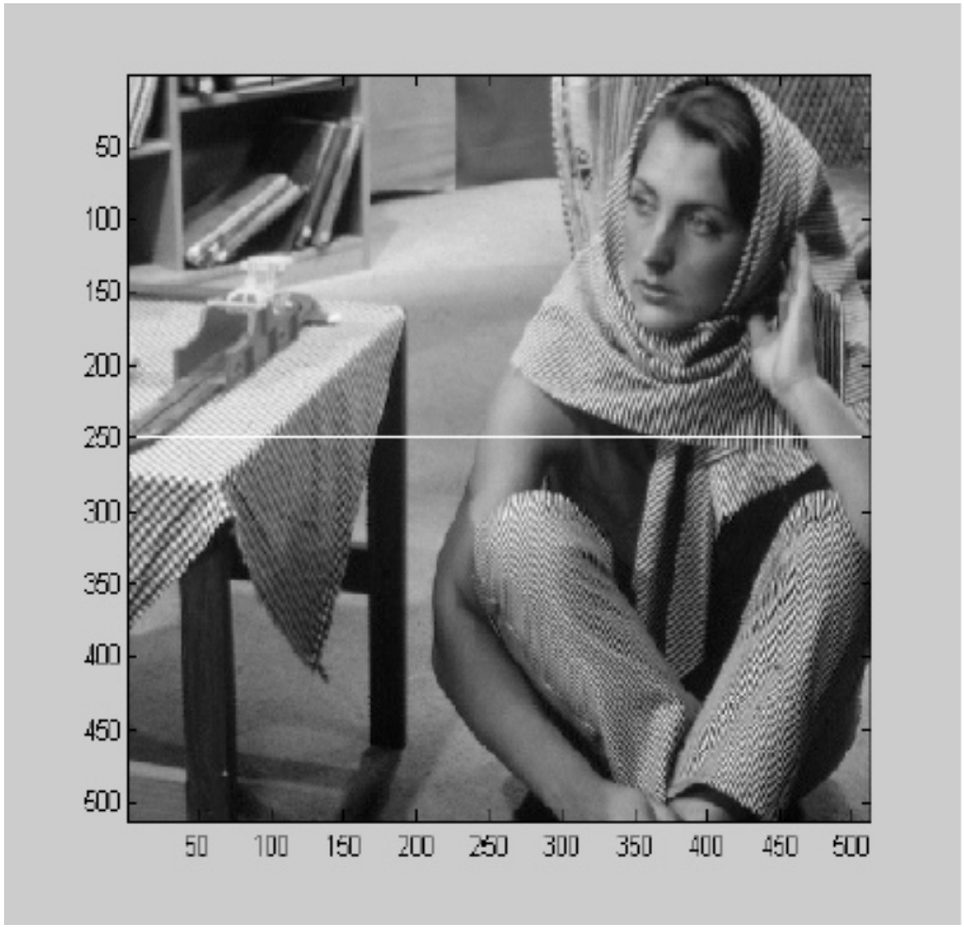


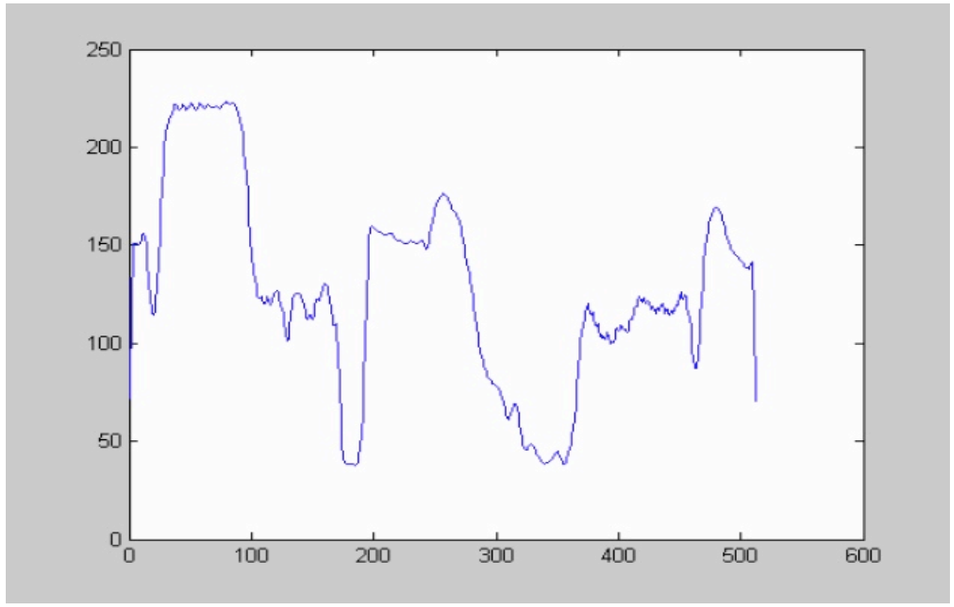
Image scanline

TECHNISCHE UNIVERSITÄT DARMSTADT

- Barbara image:
  - entire image



line 250 (smoothed):



1st derivative

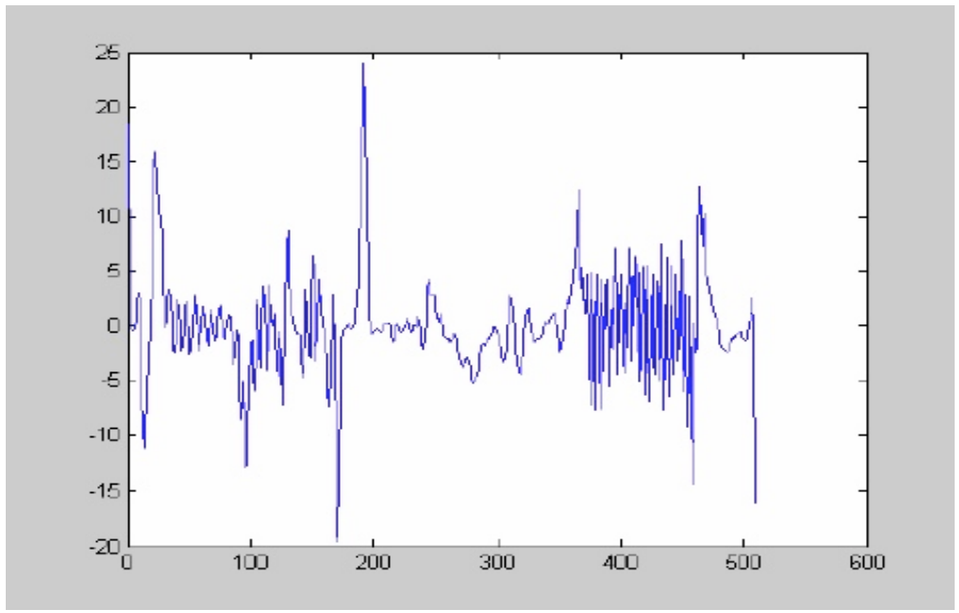
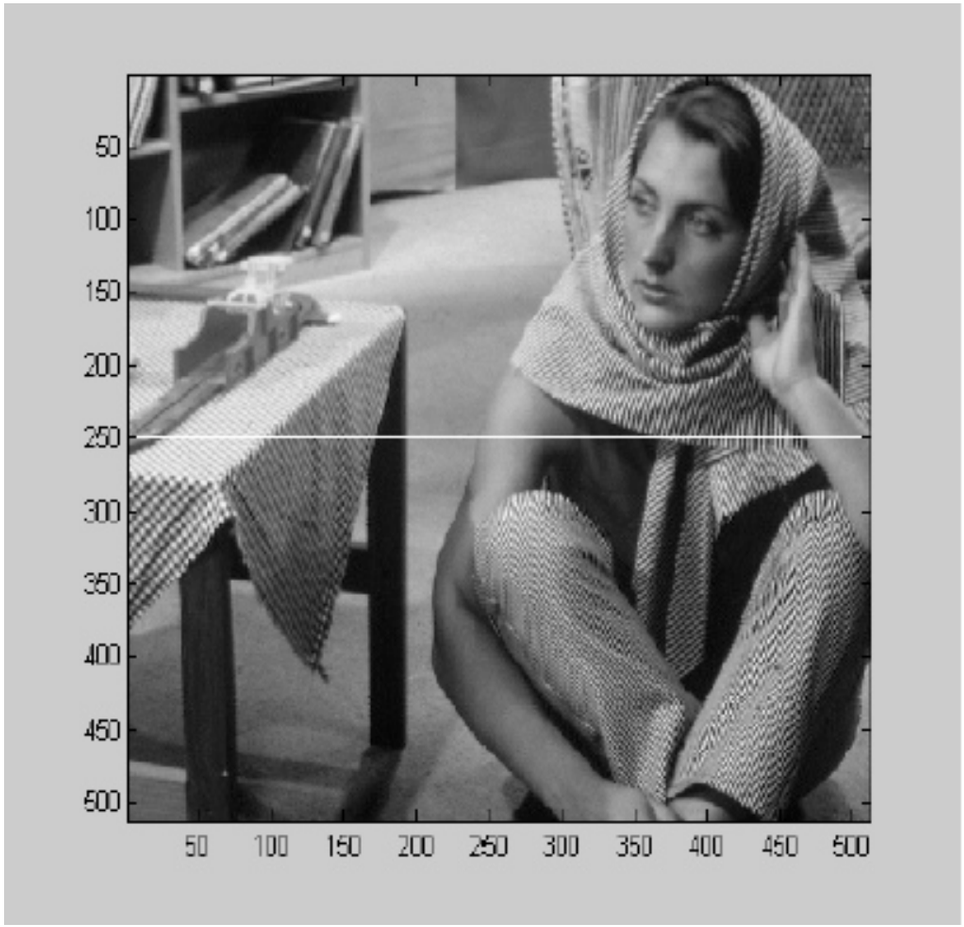


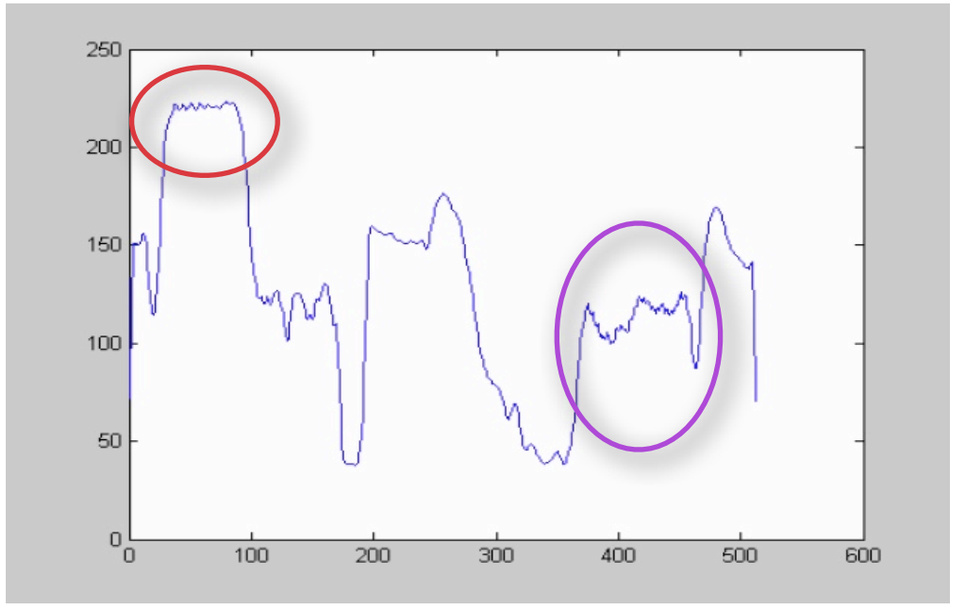
Image scanline

- Barbara image:
  - entire image

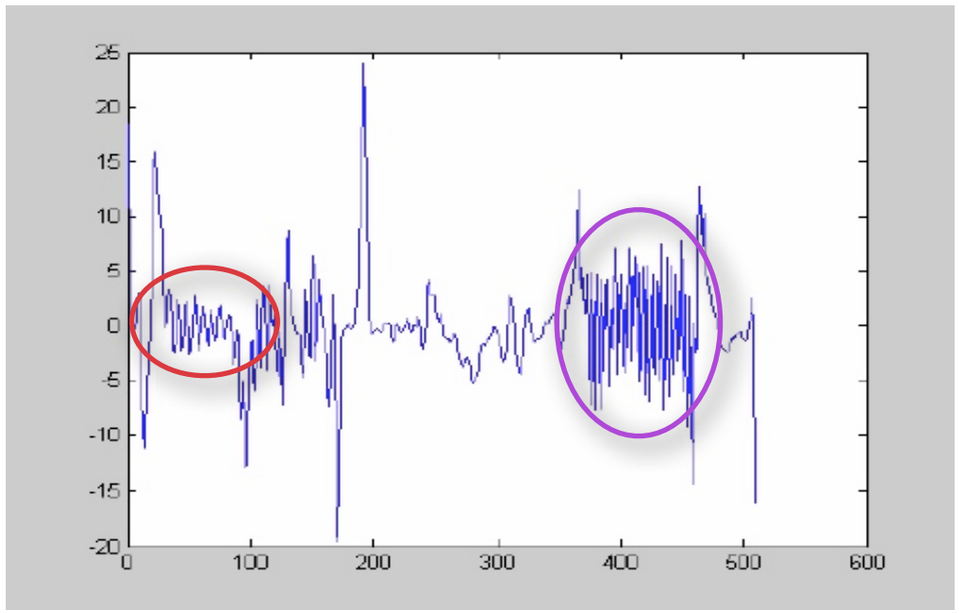


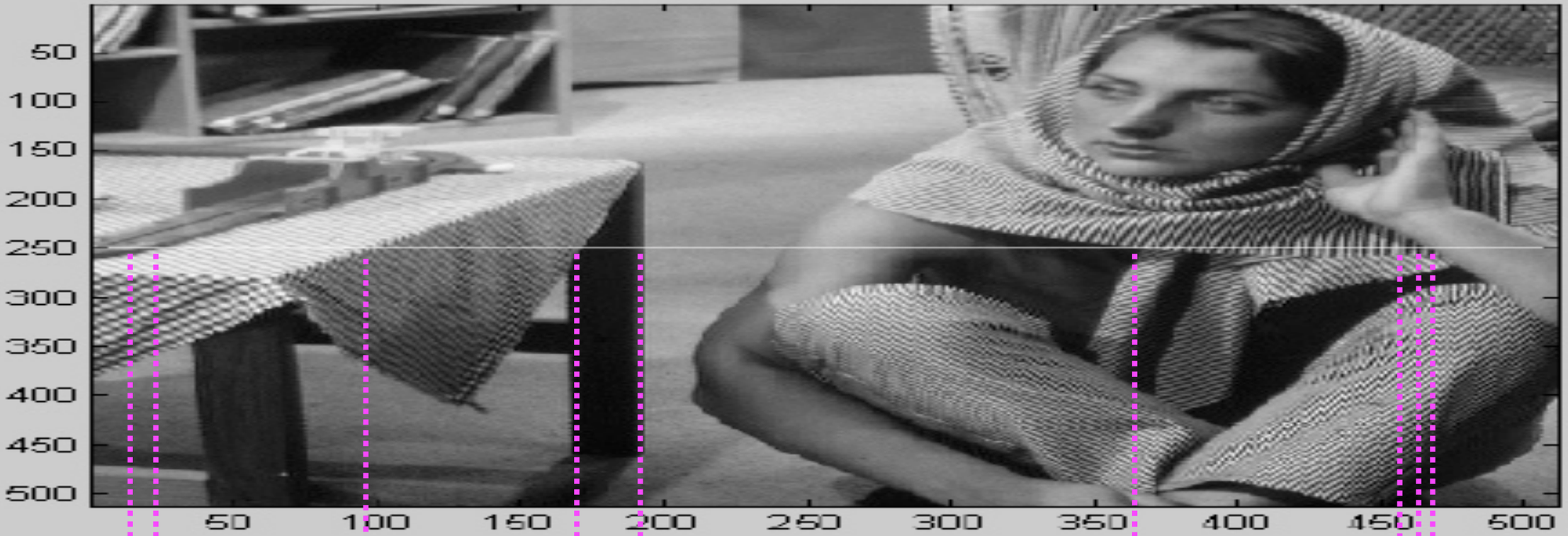
Amplification of small variations!

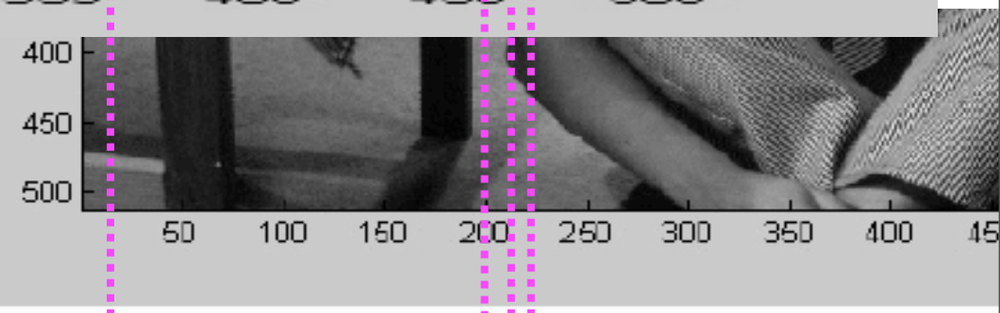
line 250 (smoothed):



1st derivative




50
100
150
200
250
300
350
400
450
500
50
100
150
200
250
300
350
400
450
500


400
450
500
50
100
150
200
250
300
350
400

25
20
15
10
5
0
-5
-10
-15
-20
0
100
200
300
400
500

Implementing 1D edge detection

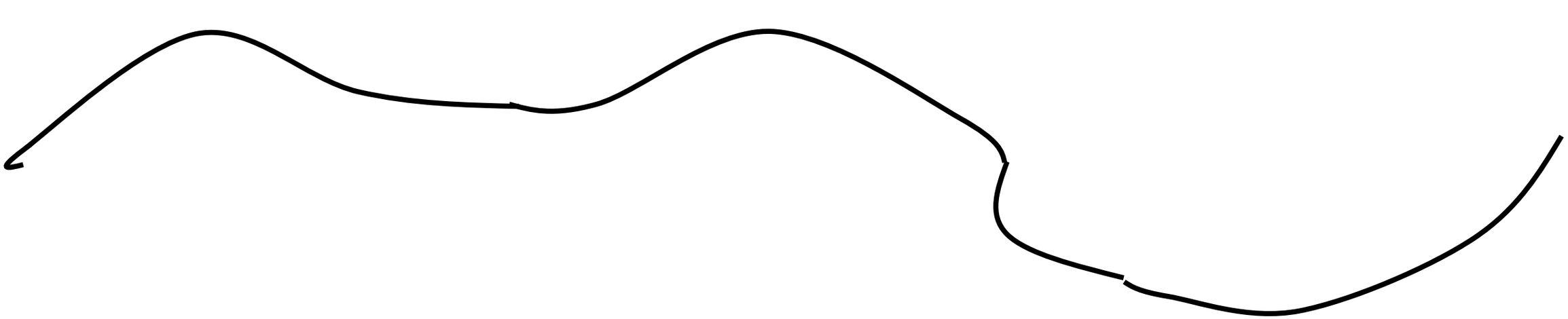
TECHNISCHE UNIVERSITÄT DARMSTADT

- Algorithmically:
  - find peak in the 1st derivative
- But
  - should be a local optimum
  - should be 'sufficiently' large
- Use 2 thresholds (called hysteresis)
  - high threshold (of absolute value) to start edge curve
  - low threshold to continue them
  - (really only makes sense in 2D...)

Hysteresis

TECHNISCHE UNIVERSITÄT DARMSTADT

- Check that maximum value of gradient value is sufficiently large
  - drop-outs? use hysteresis
    - use a high threshold to start edge curves and a low threshold to continue them.



Extension to 2D Edge Detection: Partial Derivatives

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Partial derivatives**
  - in x direction:

$$\frac{\mathrm{d}}{\mathrm{d}x} I(x,y) = I_x \approx D_x * I$$

  - in y direction:

$$\frac{\mathrm{d}}{\mathrm{d}y} I(x,y) = I_y \approx D_y * I$$

- often approximated with simple filters (finite differences):
  - first try

$$D_x = \frac{1}{6} \cdot \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline \end{array} \qquad D_y = \frac{1}{6} \cdot \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$$

Extension to 2D Edge Detection: Partial Derivatives

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Partial derivatives**
  - in x direction:

$$\frac{\mathrm{d}}{\mathrm{d}x} I(x,y) = I_x \approx D_x * I$$

  - in y direction:

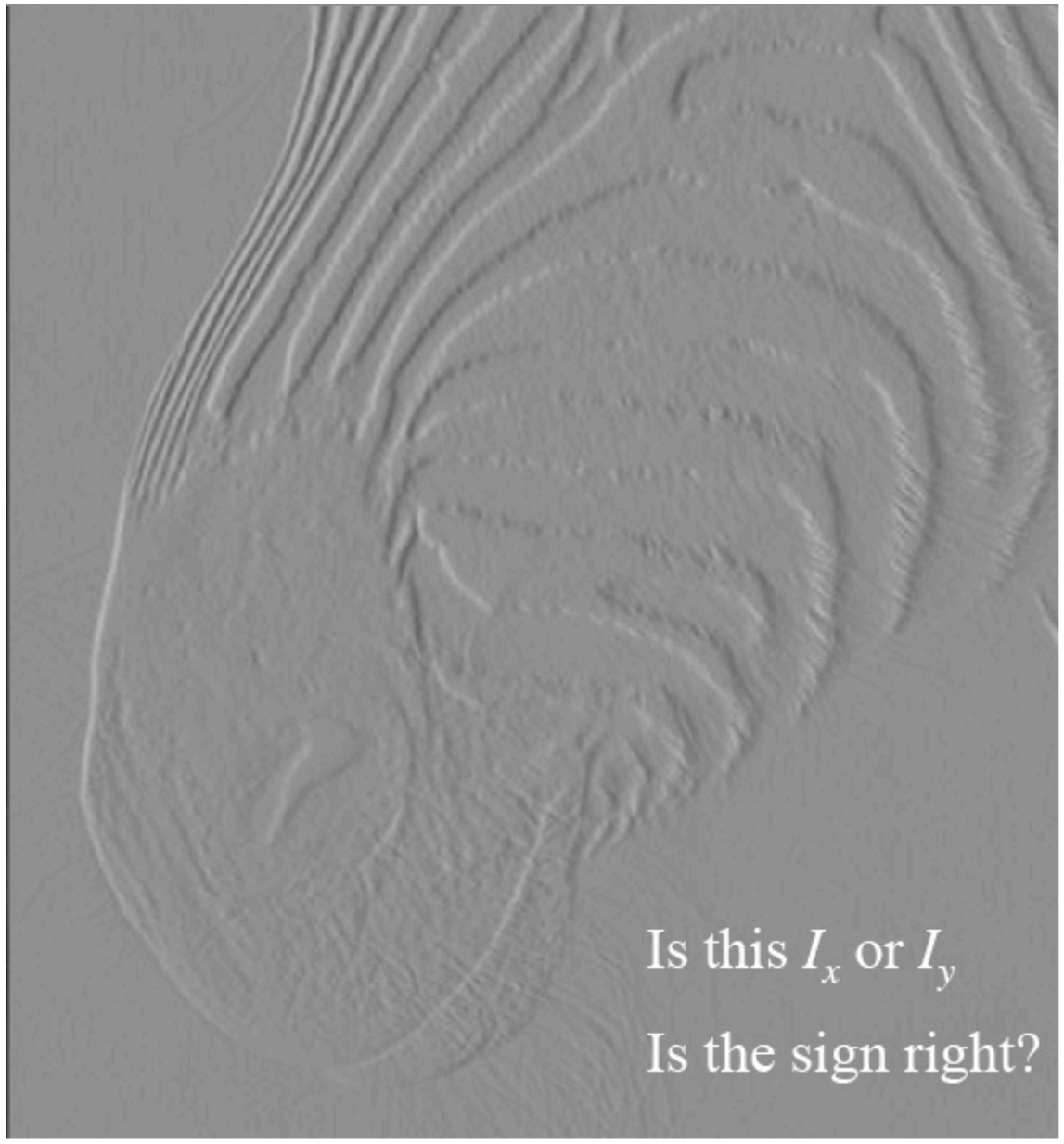
$$\frac{\mathrm{d}}{\mathrm{d}y} I(x,y) = I_y \approx D_y * I$$

- often approximated with simple filters (finite differences):
  - second try: **Sobel filters** (Gaussian smoothing in opposite direction)

$$D_x = \frac{1}{8} \cdot \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 2 & 0 & -2 \\ \hline 1 & 0 & -1 \\ \hline \end{array} \qquad D_y = \frac{1}{8} \cdot \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

Finite Differences

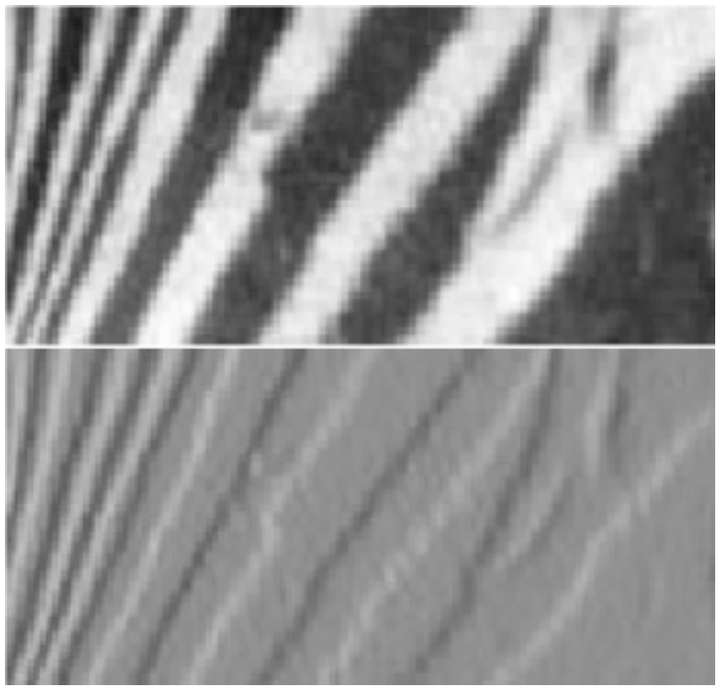
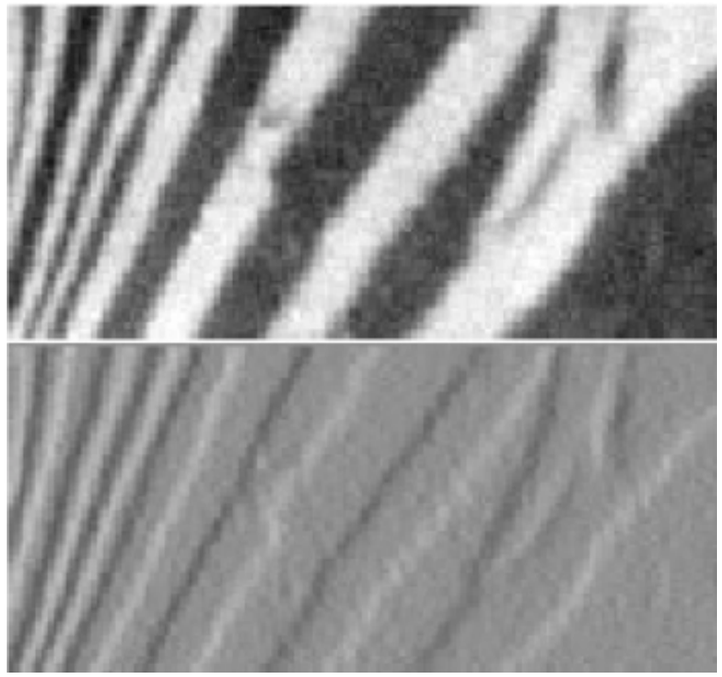
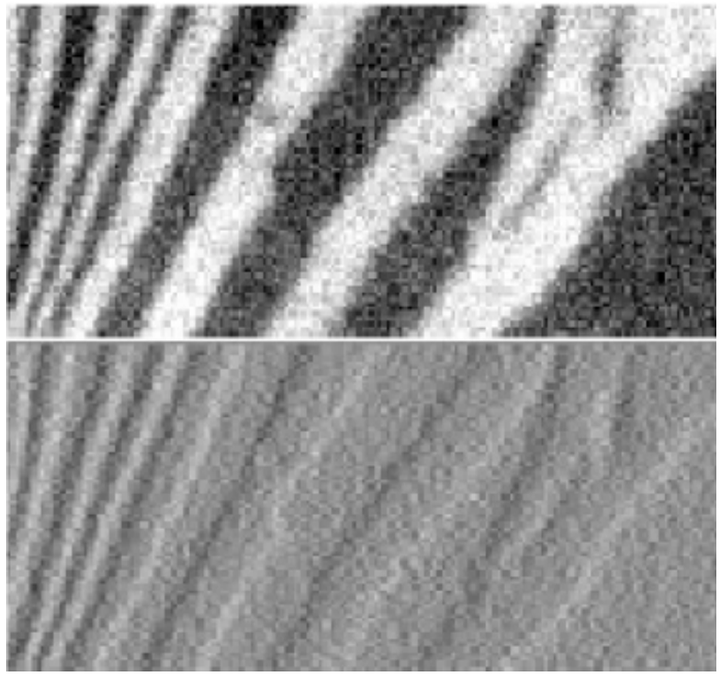
TECHNISCHE UNIVERSITÄT DARMSTADT



Finite differences responding to noise

TECHNISCHE UNIVERSITÄT DARMSTADT

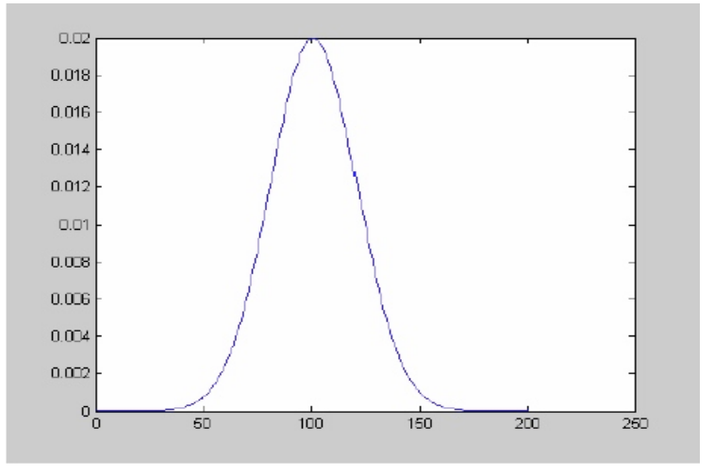
- Increasing noise level (from left to right)
  - noise: zero mean additive Gaussian noise

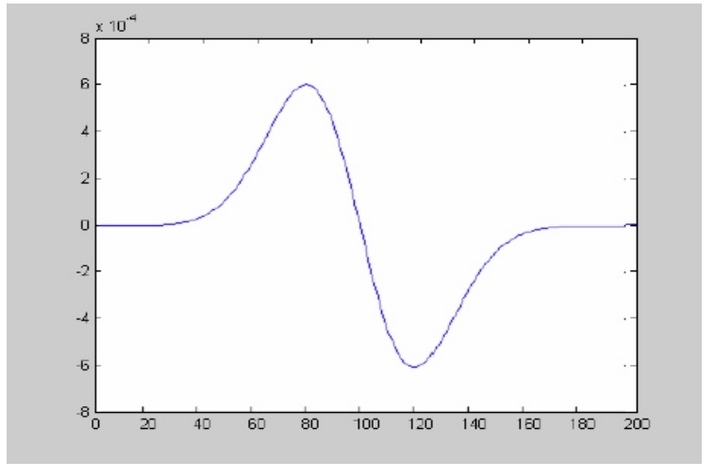




Again: Derivatives and Smoothing

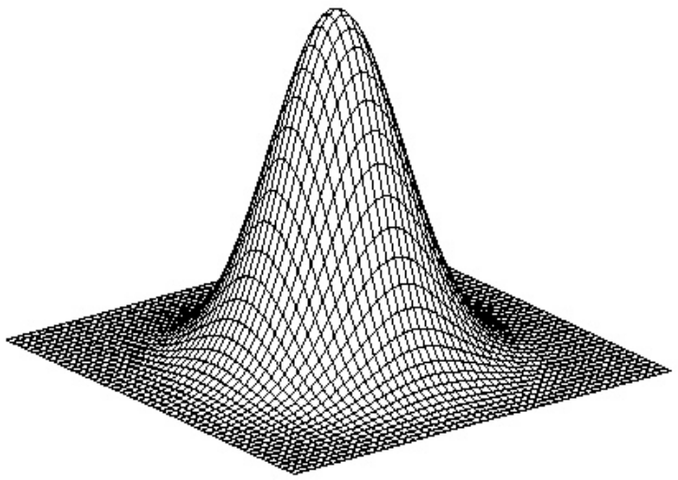
TECHNISCHE UNIVERSITÄT DARMSTADT

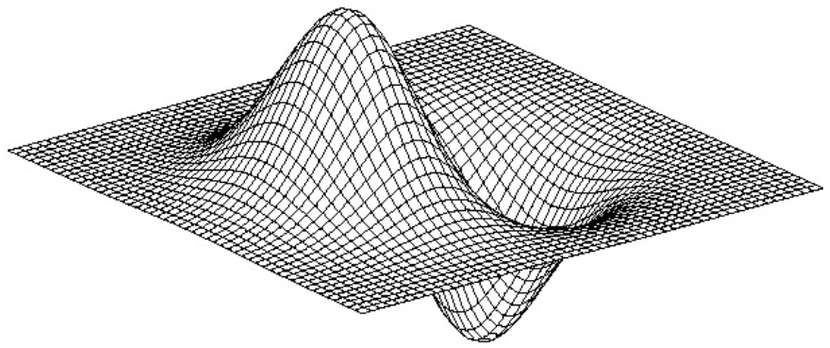
- Derivative in x-direction: $D_x * (G * I) = (D_x * G) * I$
  - in 1D:





  - in 2D:

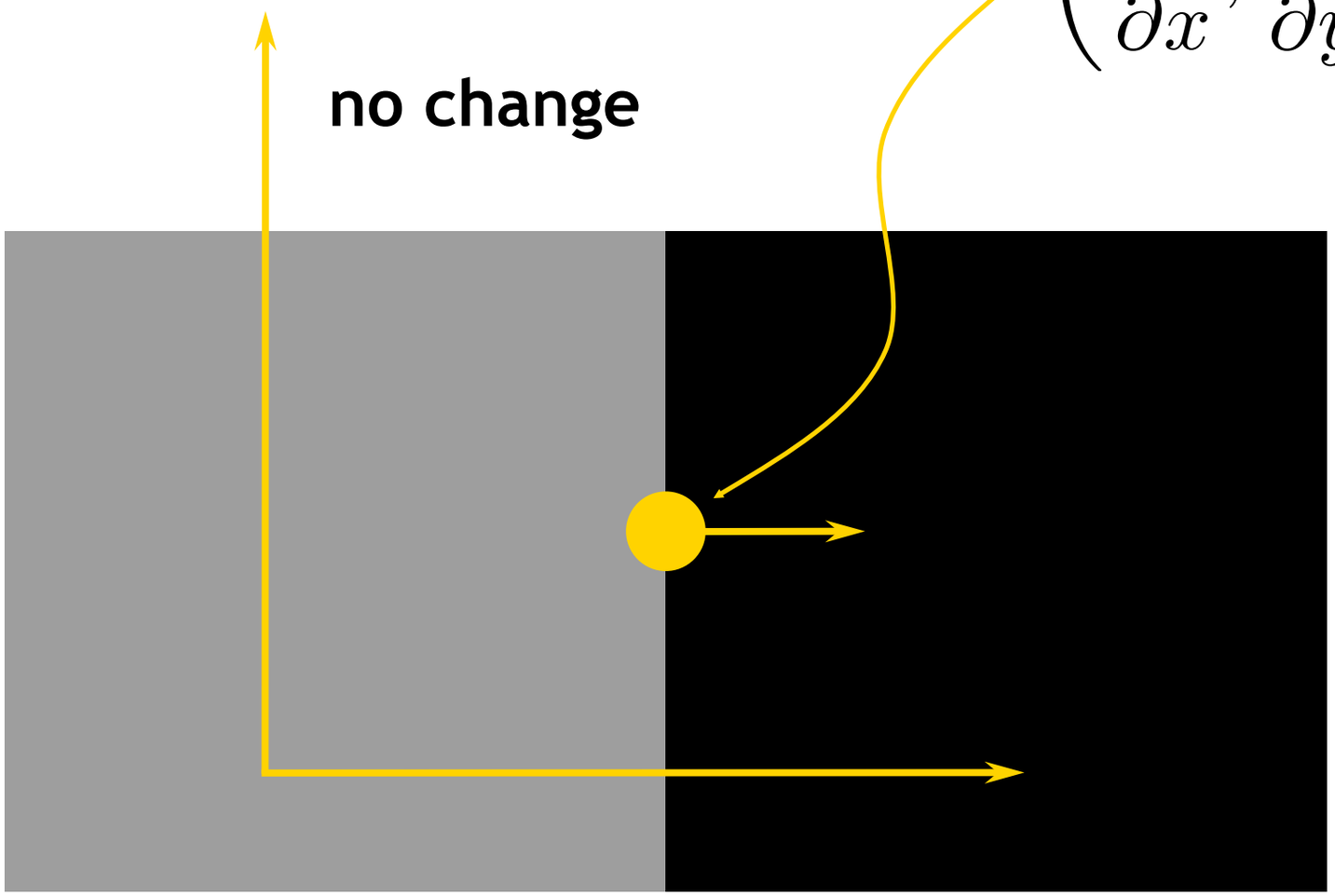




What is the gradient?

TECHNISCHE UNIVERSITÄT DARMSTADT

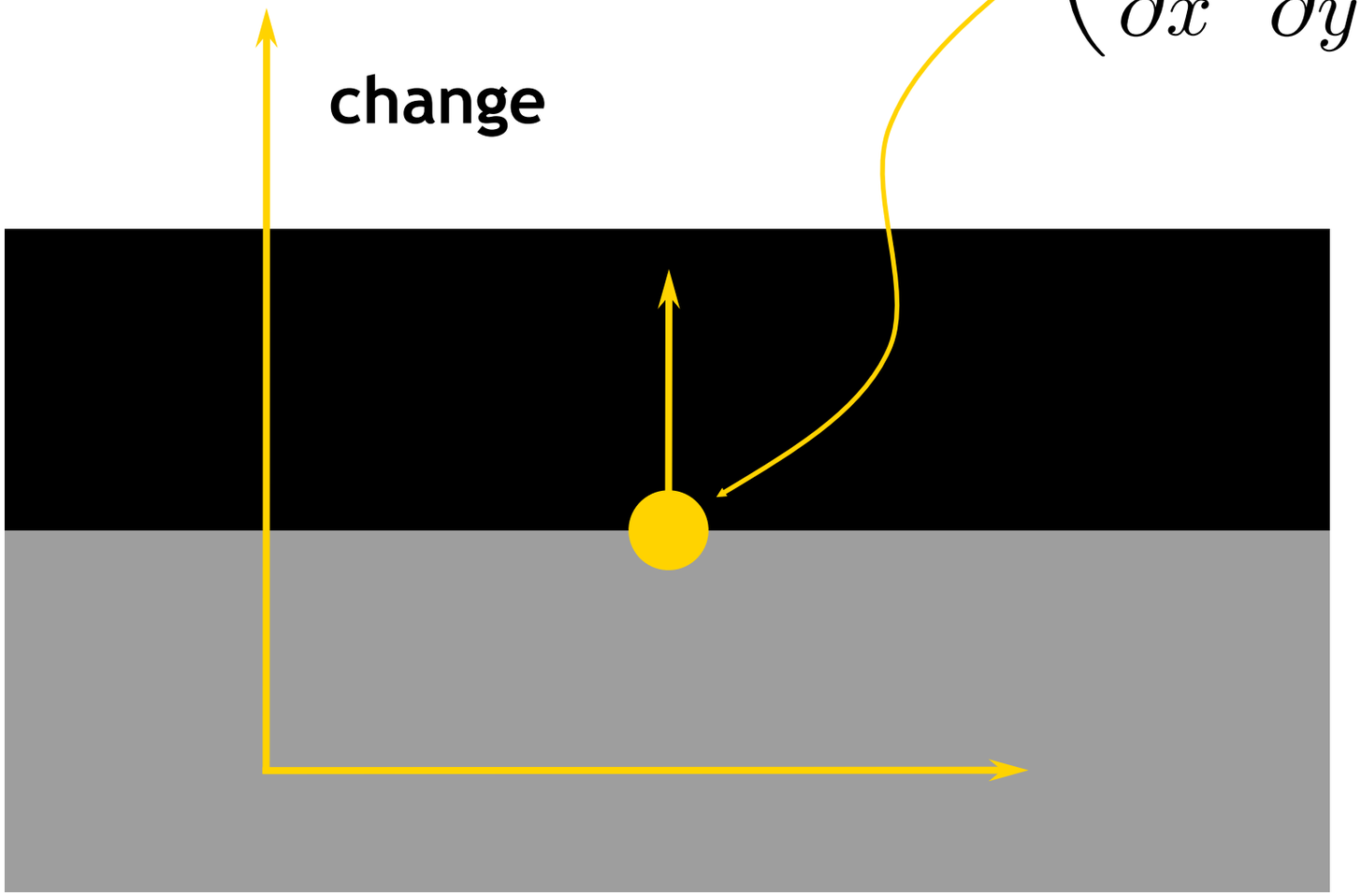
$$\left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right) = (k, 0)$$



What is the gradient?

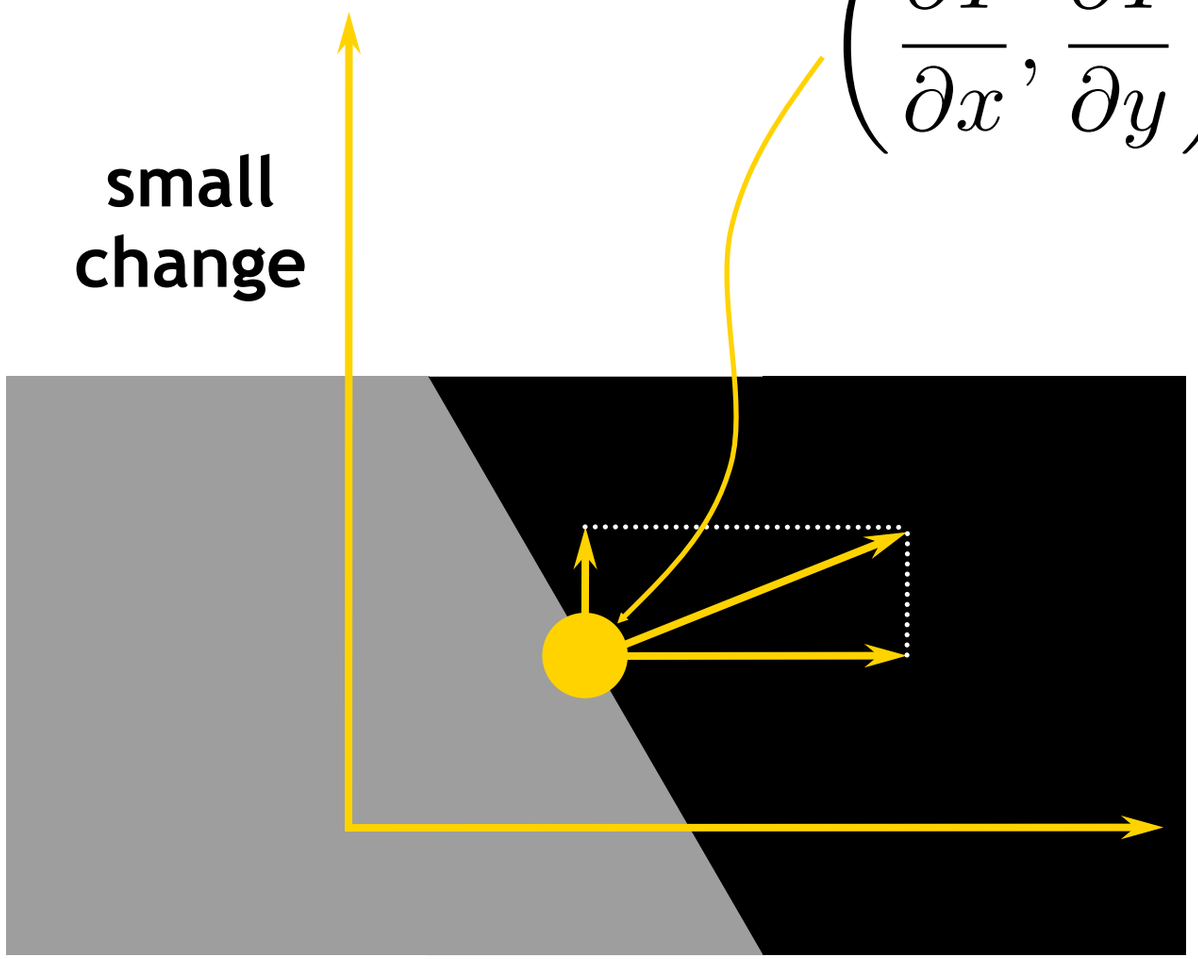
TECHNISCHE UNIVERSITÄT DARMSTADT

$$\left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right) = (0, k)$$



What is the gradient?

$$\left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right) = (k_x, k_y)$$



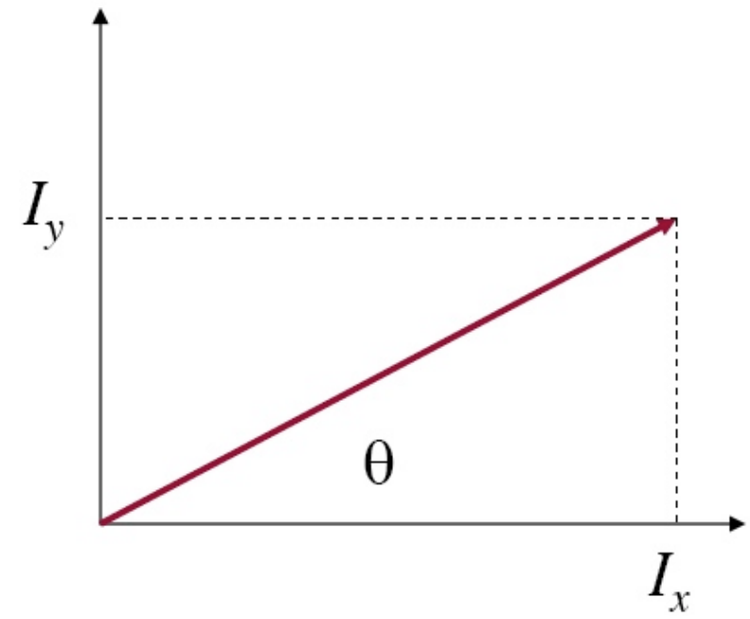
- gradient direction is perpendicular to edge
- gradient magnitude measures edge strength

2D Edge Detection

TECHNISCHE UNIVERSITÄT DARMSTADT

- Calculate derivative
  - use the **magnitude** of the gradient
  - the gradient is:

$$\nabla I = (I_x, I_y) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right)$$



  - the magnitude of the gradient is:

$$||\nabla I|| = \sqrt{I_x^2 + I_y^2}$$

  - the direction of the gradient is:

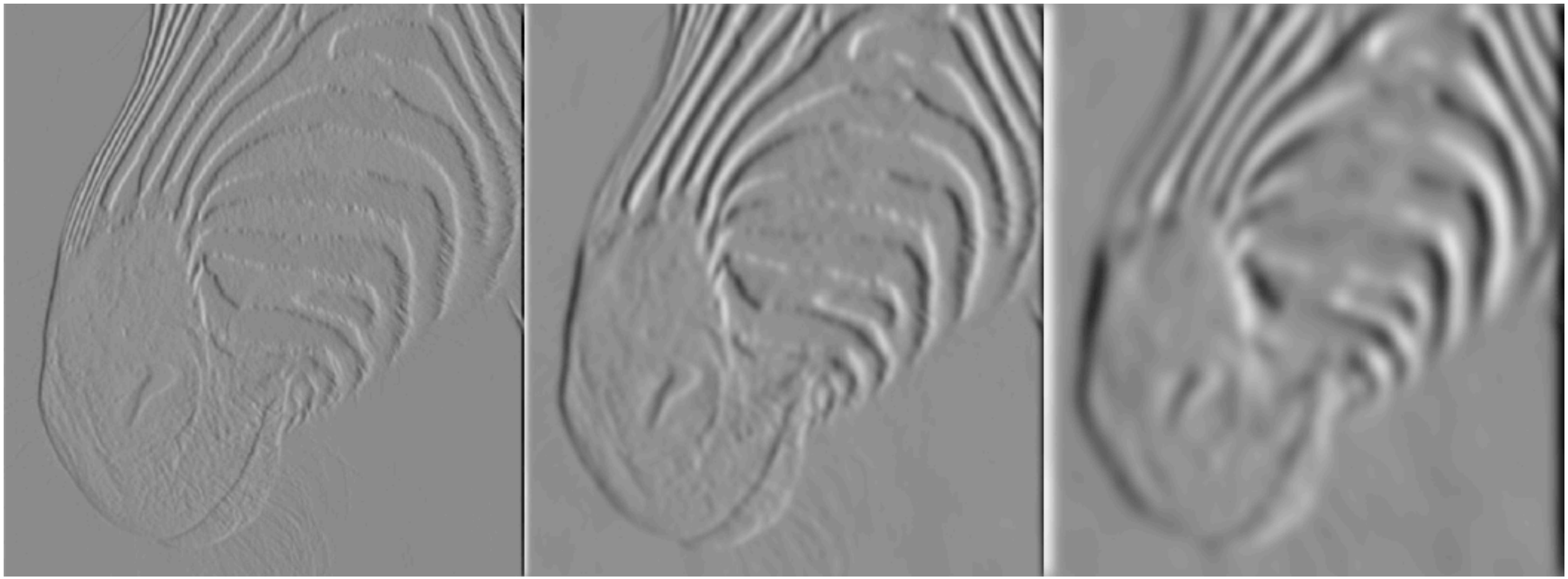
$$\theta = \text{atan}(I_y, I_x)$$

2D Edge Detection

TECHNISCHE UNIVERSITÄT DARMSTADT

- The **scale** of the smoothing filter affects derivative estimates, and also the semantics of the edges recovered
  - Note: strong edges persist across scales

1 pixel | 3 pixels | 7 pixels



‘Optimal’ Edge Detection: Canny

TECHNISCHE UNIVERSITÄT DARMSTADT

- Assume:
  - linear filtering
  - additive i.i.d. Gaussian noise
- Edge detection should have:
  - **good detection:** filter responds to edge, not to noise
  - **good localization:** detected edge near true edge
  - **single response:** one per edge
- Then: Optimal detector is approximately derivative of Gaussian
- Detection/localization tradeoff:
  - more smoothing improves detection
  - and hurts localization

The Canny edge detector

TECHNISCHE UNIVERSITÄT DARMSTADT

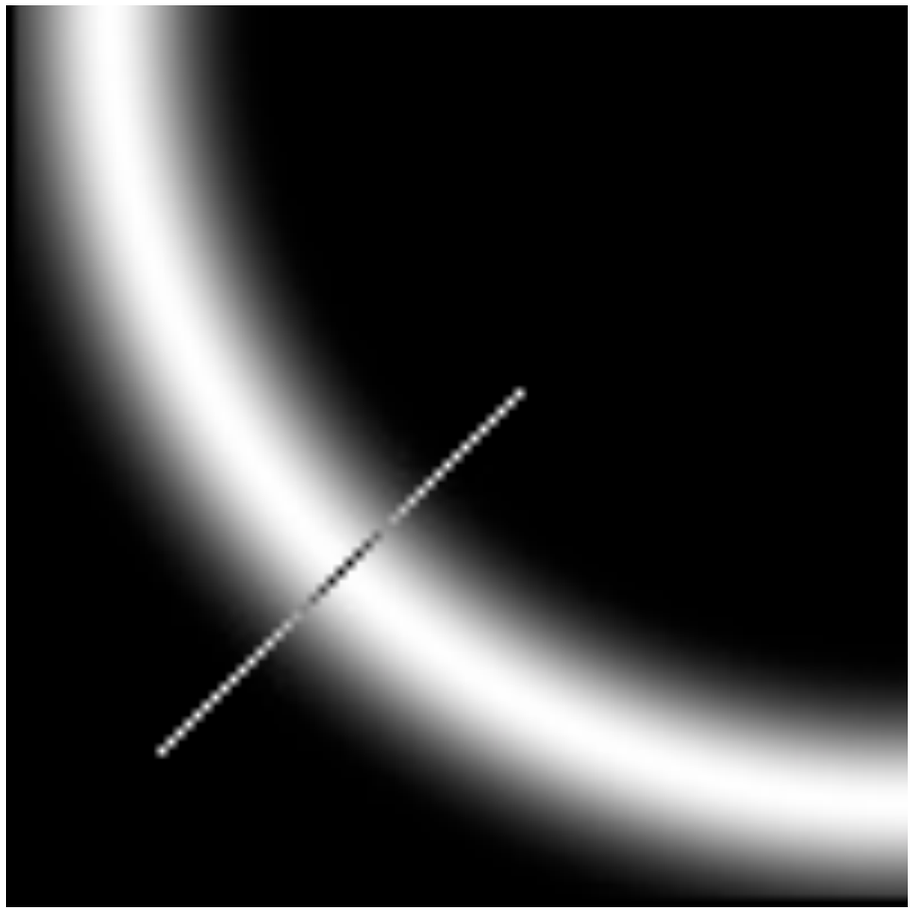
original image

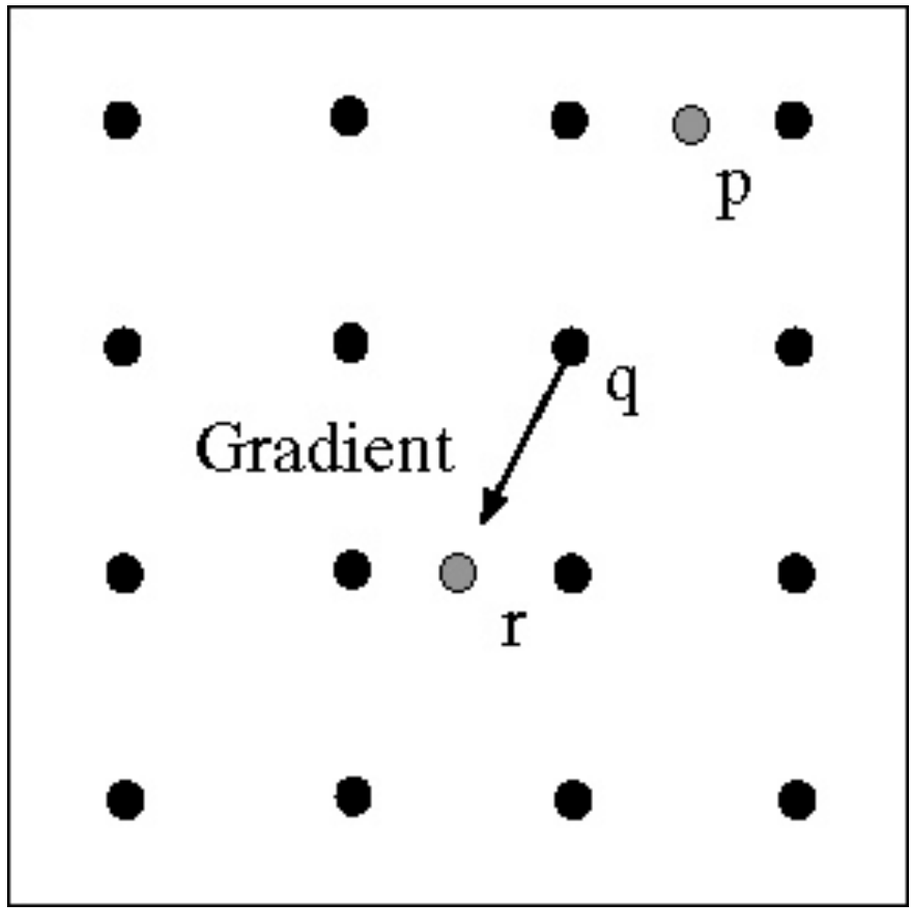
norm of the gradient

thinning: (non-maximum suppression)

thresholding

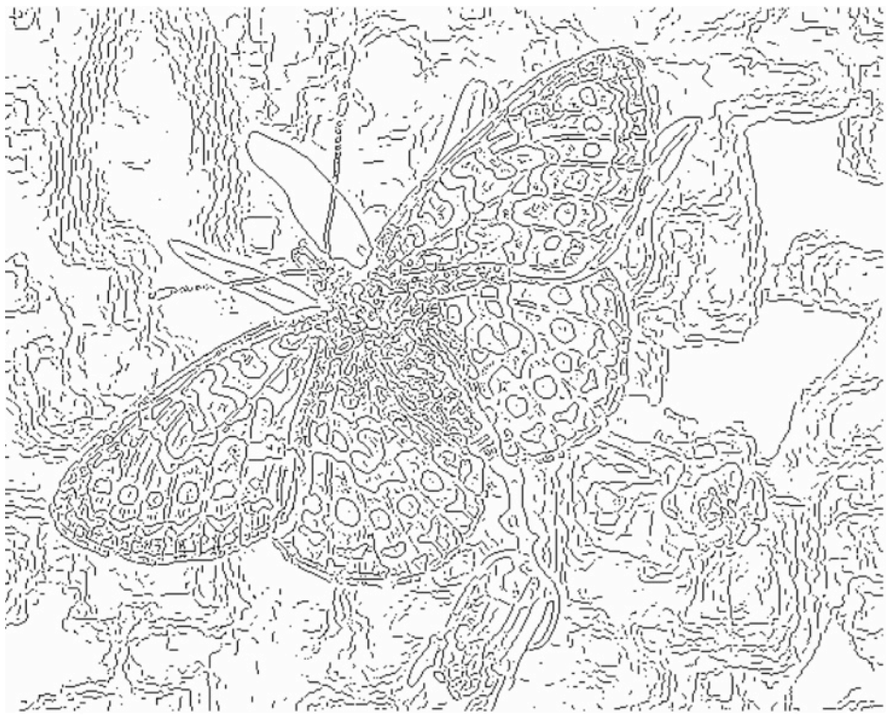
Non-maximum suppression





- Check if pixel is local maximum along gradient direction
  - choose the largest gradient magnitude along the gradient direction
  - requires checking interpolated pixels p and r

Butterfly Example [Ponce & Forsyth]



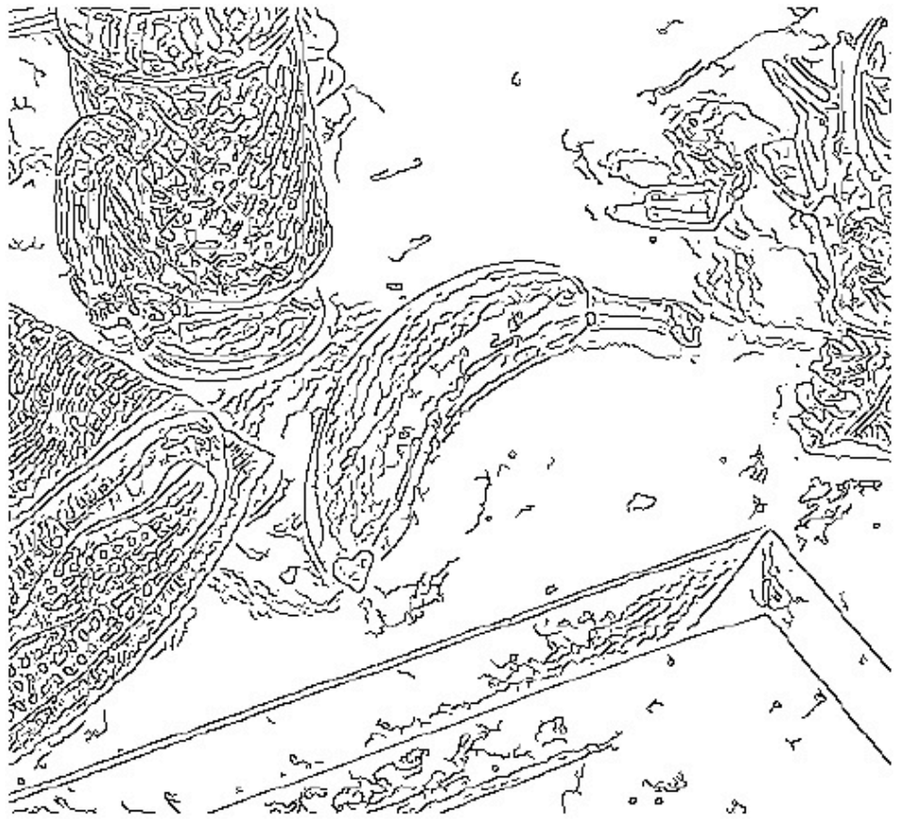
fine scale
high
threshold

coarse
scale
low
threshold

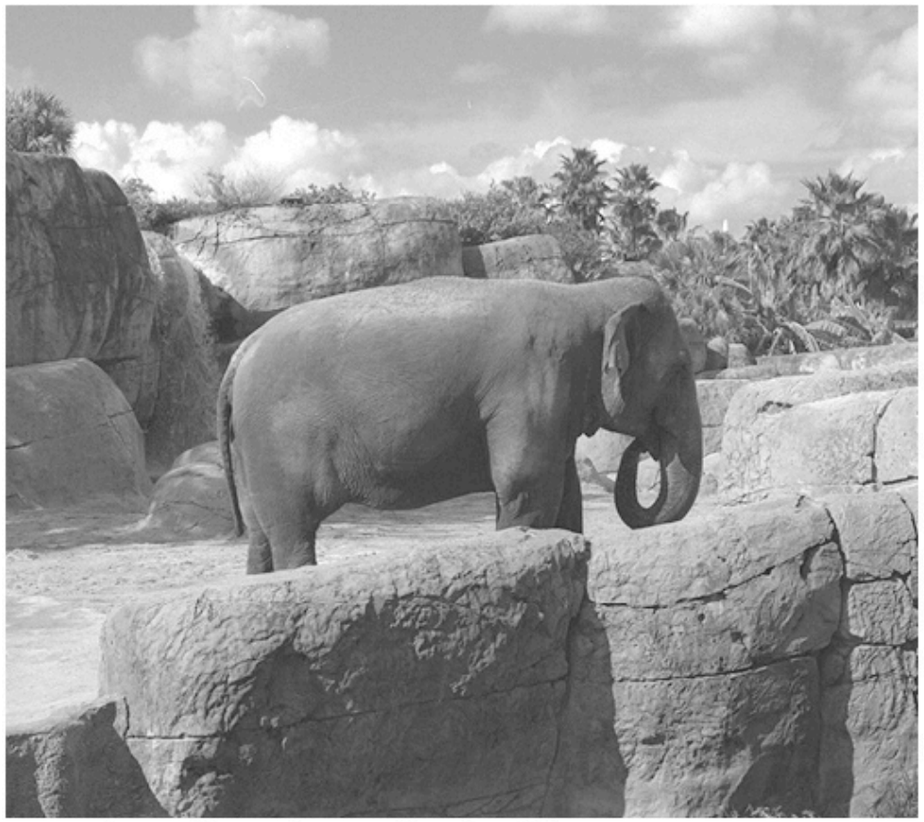
coarse
scale,
high
threshold

Line drawing vs. edge detection

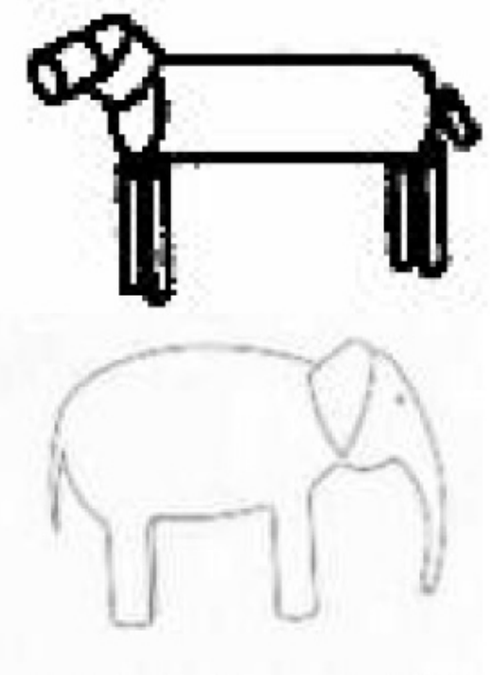
TECHNISCHE UNIVERSITÄT DARMSTADT



Models vs. edge detection



University of South Florida

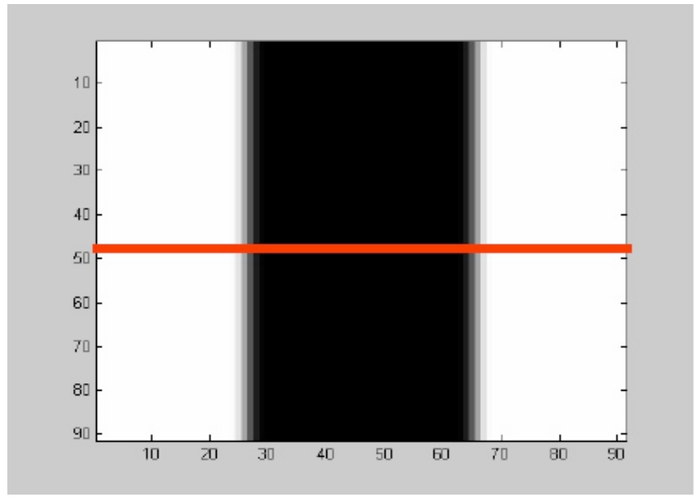


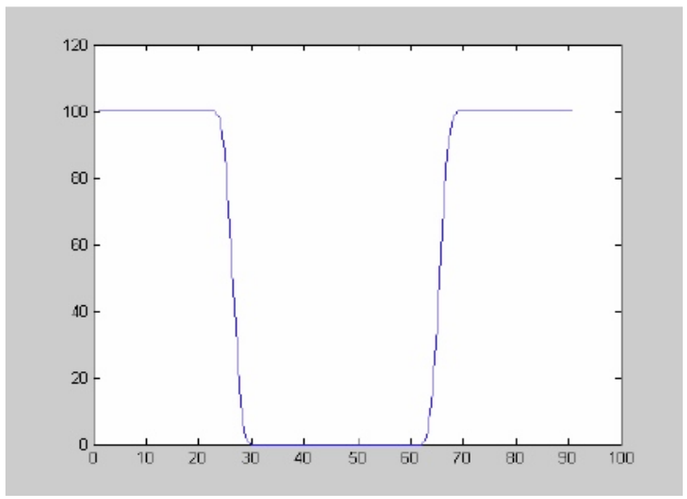
Match “model” to measurements?

Edges & Derivatives…

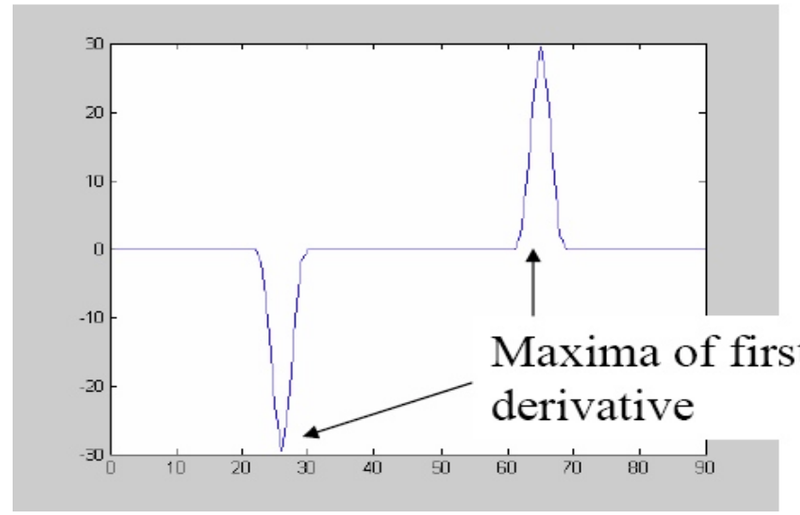
TECHNISCHE UNIVERSITÄT DARMSTADT

- Recall:
  - the zero-crossings of the second derivative tell us the location of edges

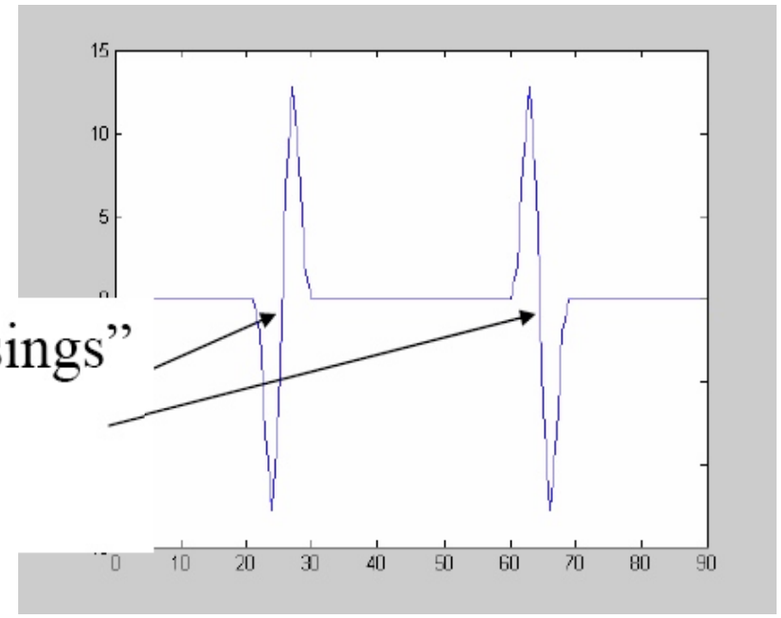




1st derivative



2nd derivative



"zero crossings" of second derivative

Compute 2nd order derivatives

TECHNISCHE UNIVERSITÄT DARMSTADT

- 1st derivative:

$$\frac{d}{dx} f(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \approx f(x+1) - f(x)$$

- 2nd derivative:

$$\frac{d^2}{dx^2} f(x) = \lim_{h \to 0} \frac{\frac{d}{dx} f(x+h) - \frac{d}{dx} f(x)}{h} \approx \frac{d}{dx} f(x+1) - \frac{d}{dx} f(x)$$

$$\approx f(x+2) - 2f(x+1) + f(x)$$

- mask for
  - 1st derivative: $\boxed{1 \mid -1}$ 2nd derivative: $\boxed{1 \mid -2 \mid 1}$

The Laplacian

TECHNISCHE UNIVERSITÄT DARMSTADT

- The Laplacian:

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

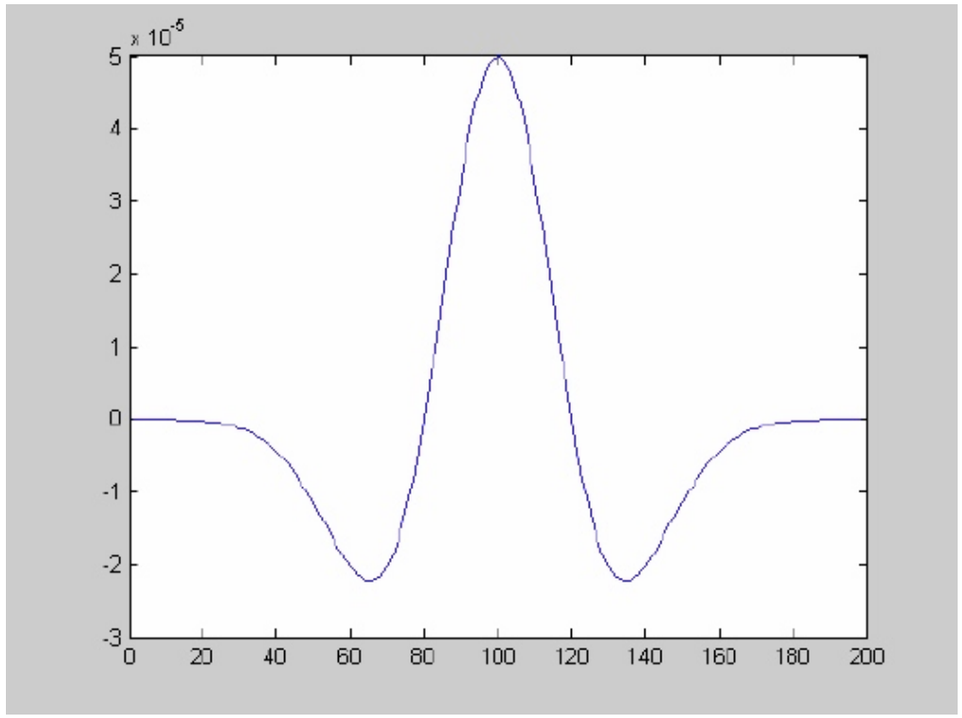
- just another linear filter:

$$\nabla^2 (G * I) = (\nabla^2 G) * I$$

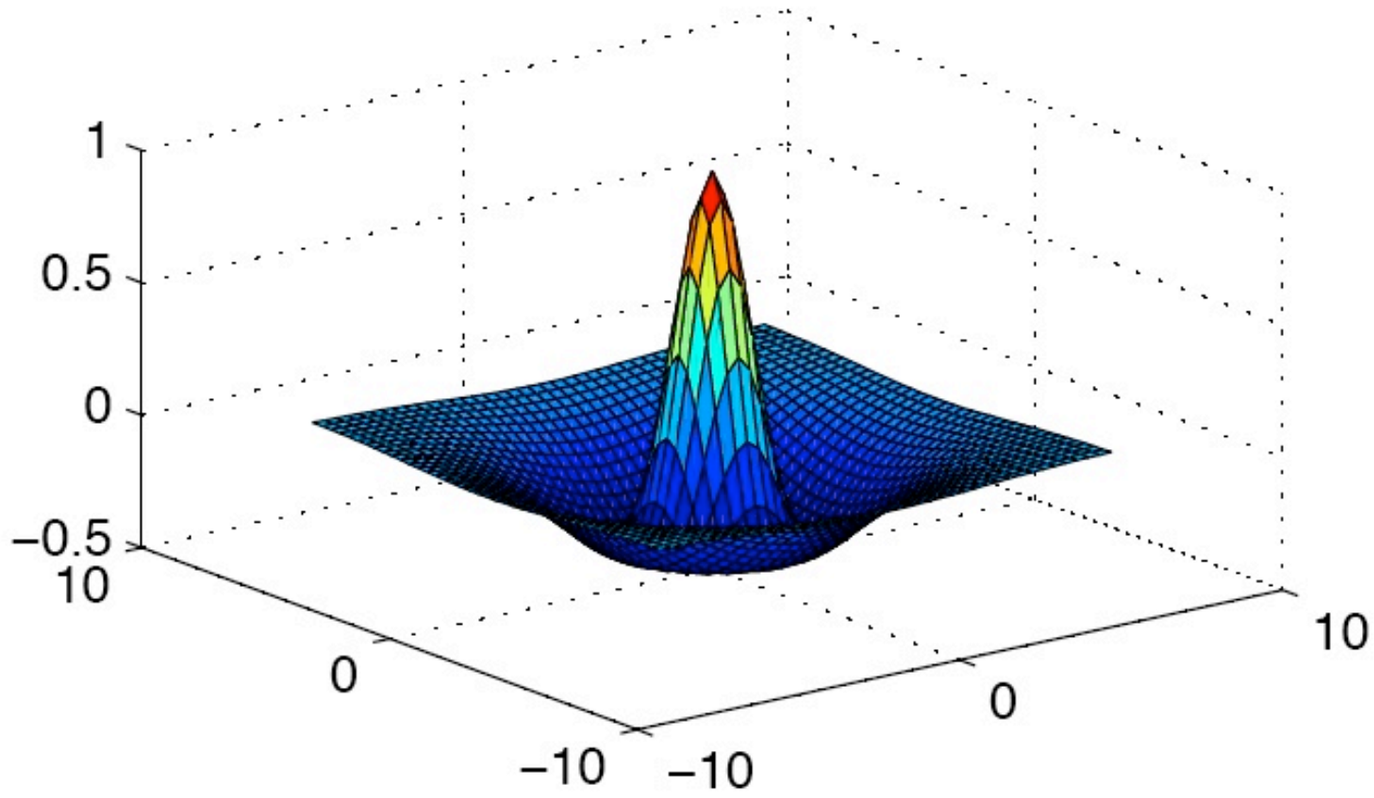
Second Derivative of Gaussian

TECHNISCHE UNIVERSITÄT DARMSTADT

- in 1D:



in 2D (mexican hat):



1D edge detection

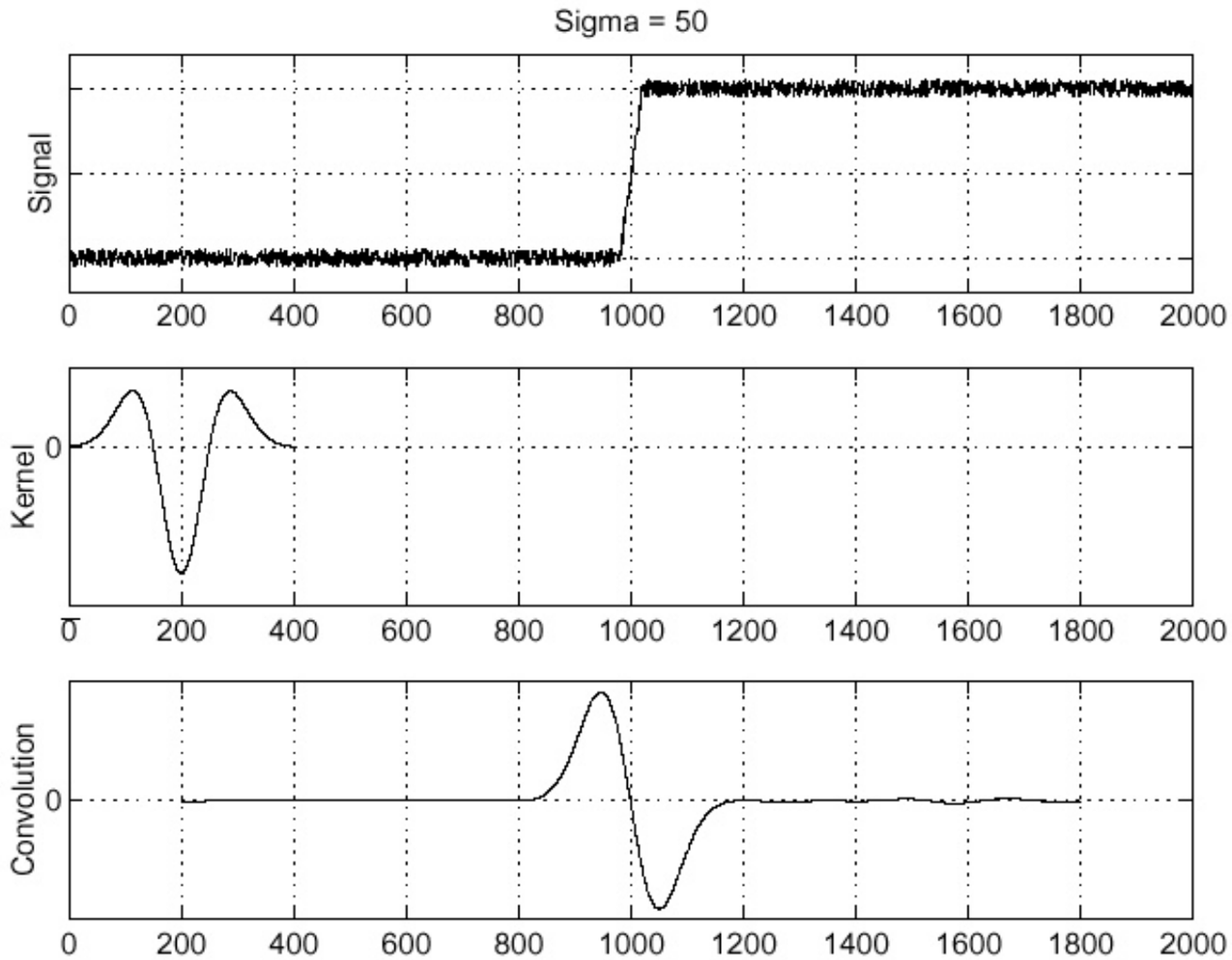
- using Laplacian

LoG - Laplacian of Gaussian operator

$$\frac{d^2}{dx^2} g$$

$$\left(\frac{d^2}{dx^2} g \right) \otimes f$$

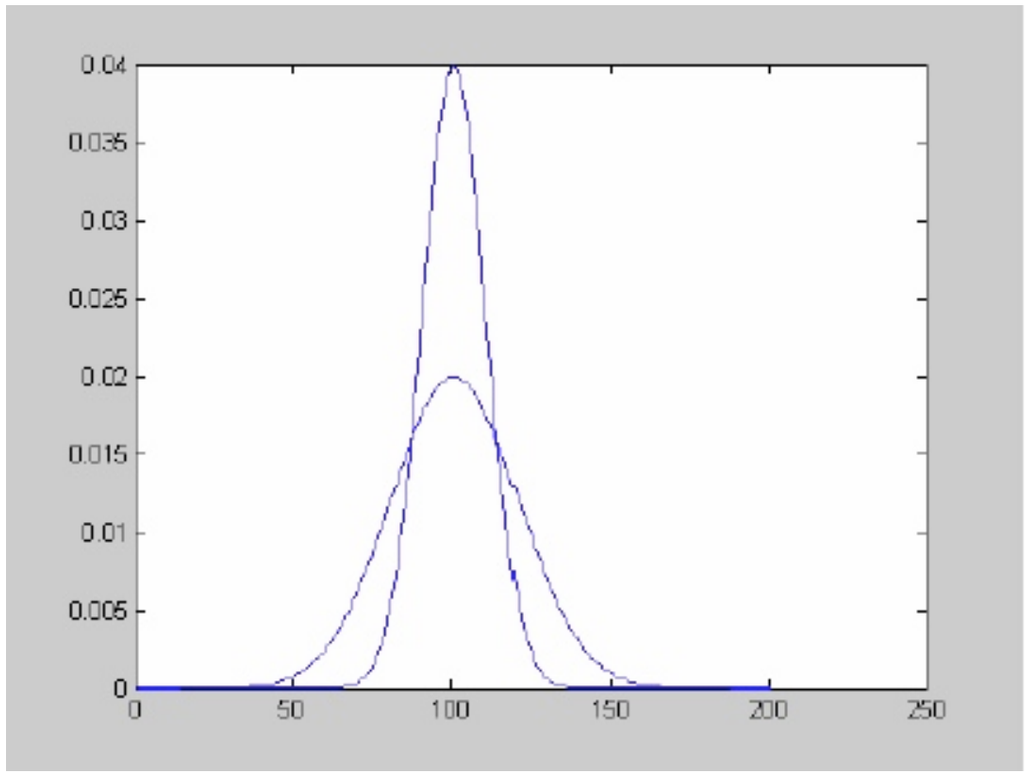
f

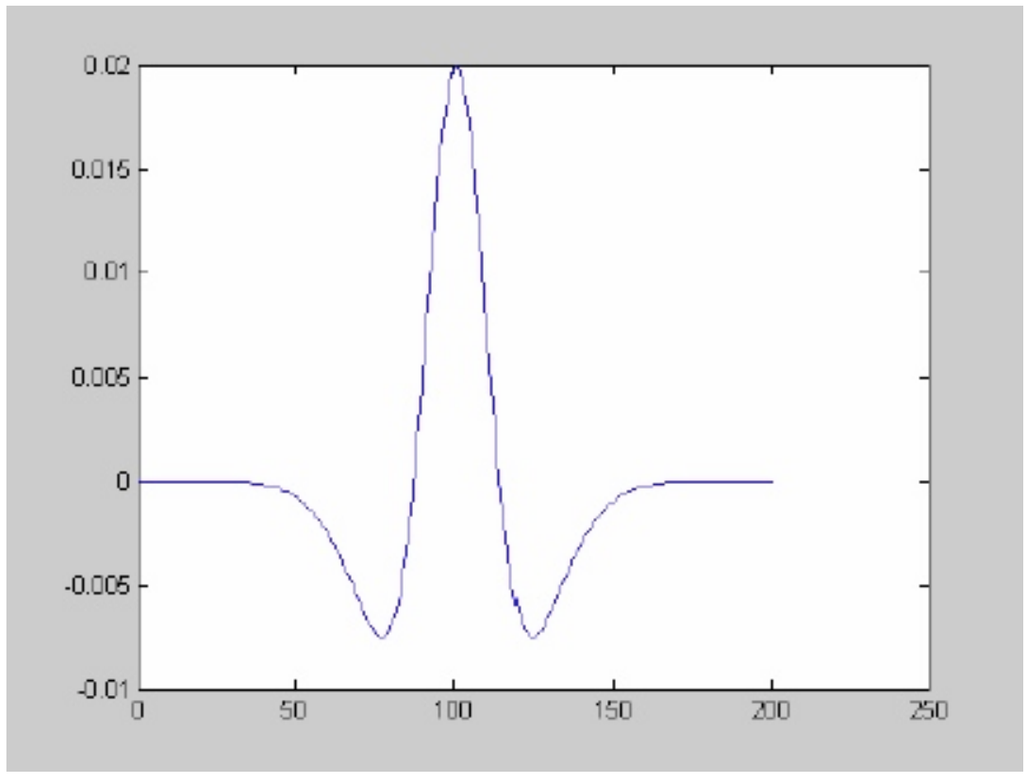


Approximating the Laplacian

TECHNISCHE UNIVERSITÄT DARMSTADT

- Approximate the LoG by a DoG...
- **Difference of Gaussians** (DoG) at different scales:

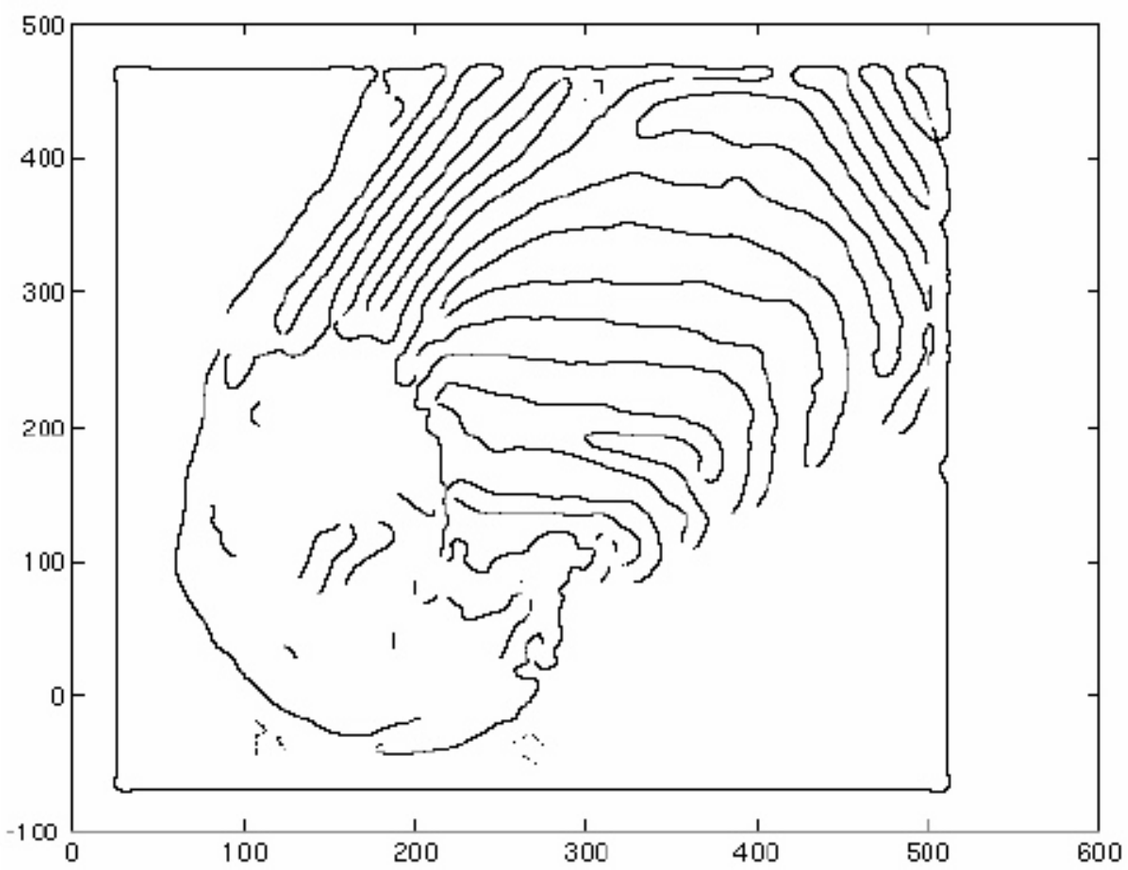




Edge Detection with Laplacian

TECHNISCHE UNIVERSITÄT DARMSTADT

sigma = 4



sigma = 2