h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

Visual Computing Digital Images

Darmstadt University

Prof. Dr Elke Hergenröther
Björn Frömmer
Prof. Dr Benjamin Meyer

CHAPTER 12

Image Processing - Local Image Operations

Image Processing - Local Image Operations

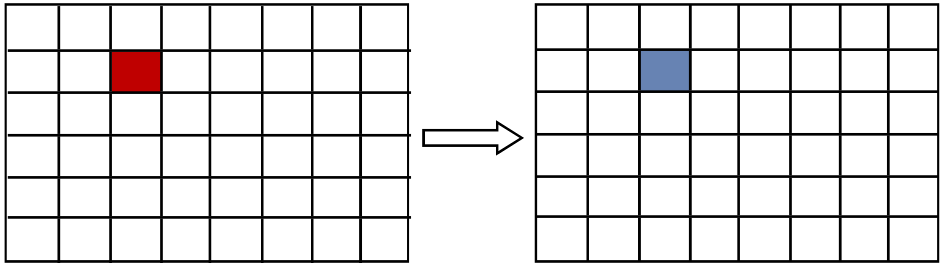
Point operators compared to local image operators Local image operators

- Blur: Mean operator, Gaussian filter
- Edge detectors: difference filter and Sobel operator, Laplace operator
- Contrast enhancement filter
- Ranking operators: Erosion, Dilation, Median as well as Opening and Closing
- Segmentation procedure

Image Processing - Local Image Operations

Point operators compared to local image operators

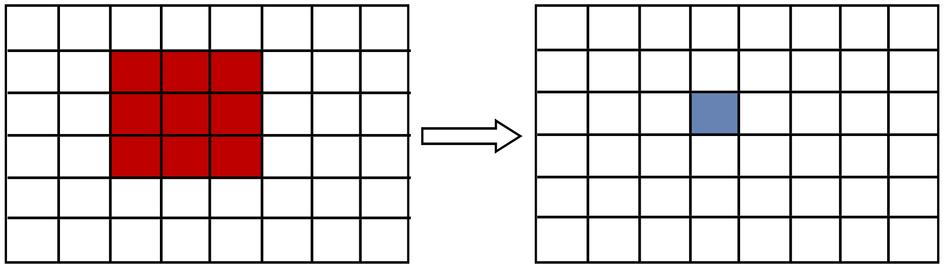
Point operator: Pixel by pixel is transformed without the neighbouring pixels being included in the transformation.



Examples:

- Brightness changes,
- Contrast changes,
- Gamma correction,
- Colour transformation, …

Local image operators: Each pixel is transformed in relation to its neighbouring pixels.



Examples:

- Blur / smear images,
- Detect edges, …

Image Processing - Local Image Operations

Local image operations allow the neighbouring pixels to be included in the calculation of the pixel formation (convolution) by using different kernels.

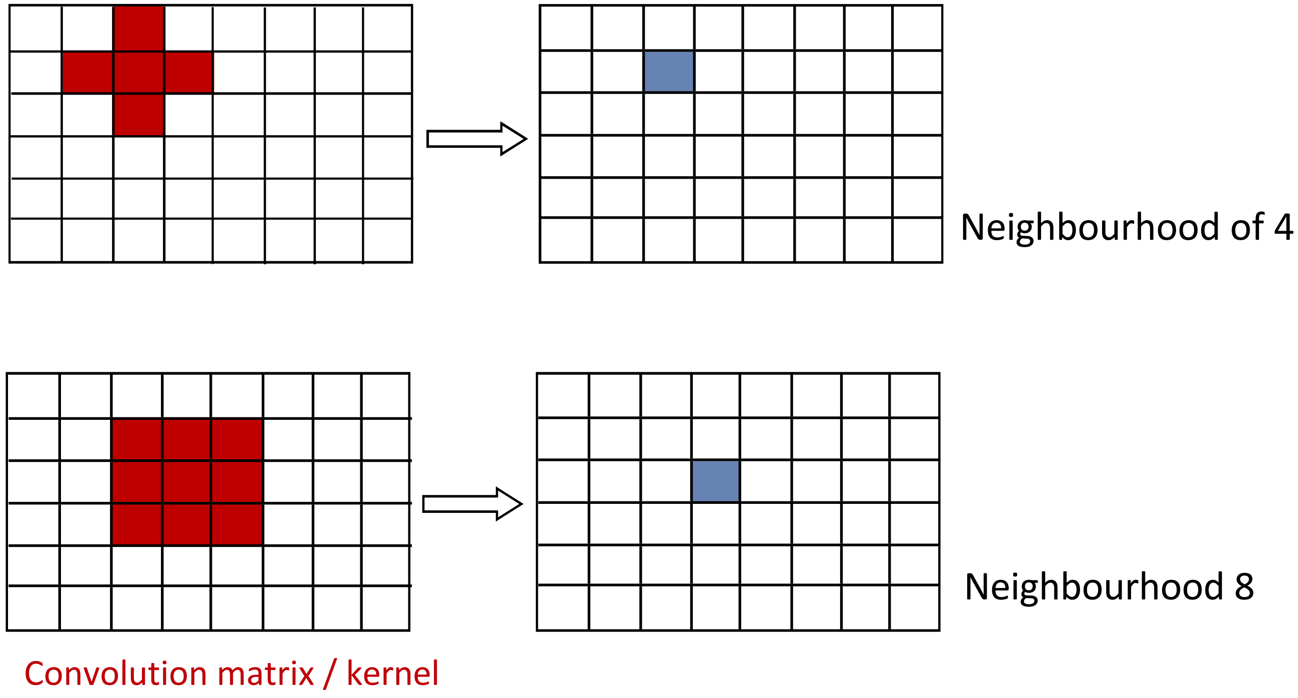
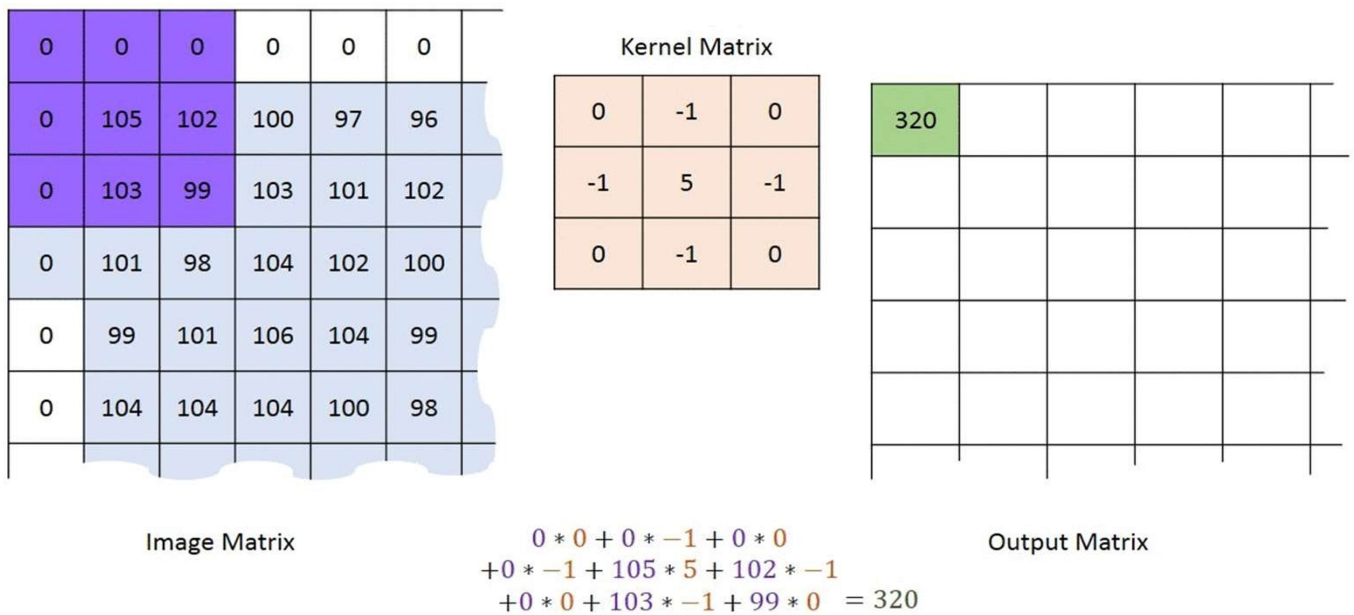


Image Processing - Local Image Operators

Example of a convolution with a local image operator using the N8 neighbourhood



from: https://i.stack.imgur.com/9OZKF.gif

Image Processing - Local Image Operators

Example of a convolution with a local image operator using the N8 neighbourhood

0	0	0	0	0	0
0	105	102	100	97	96
0	103	99	103	101	102
0	101	98	104	102	100
0	99	101	106	104	99
0	104	104	104	100	98

Image Matrix

Kernel Matrix

0	-1	0
-1	5	-1
0	-1	0

320				

Output Matrix

General formulation of the convolution operation:

$$e(i,j) = \sum_{l=0}^{2}\sum_{k=0}^{2}\{g(i-1+k,\, j-1+l) * f(k,l)\}$$

from: https://i.stack.imgur.com/9OZKF.gif

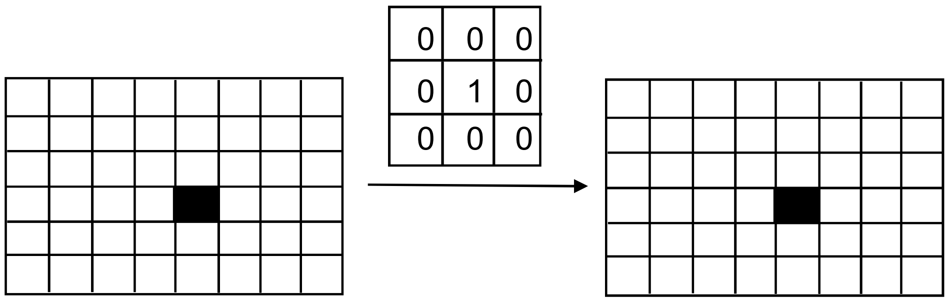
Image Processing - Local Image Operators

Convolution: **Identity operator**

Input image and convolution result are identical.

$$F = \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}$$

Kernel / convolution matrix



Input image

Convolution result

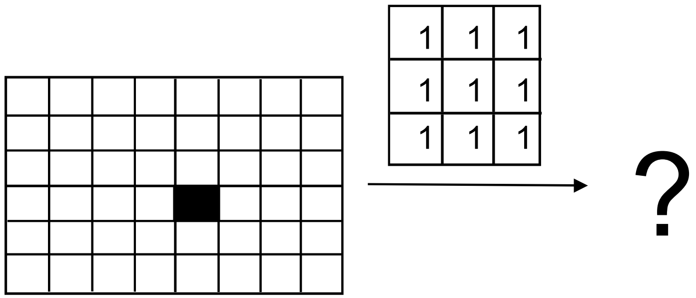
Image Processing - Local Image Operators

Convolution: **Mean value operator**

Forms the mean/average value from the neighbouring pixel values.

$$F = \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Kernel or convolution matrix



Input image is
8*6 pixels in size

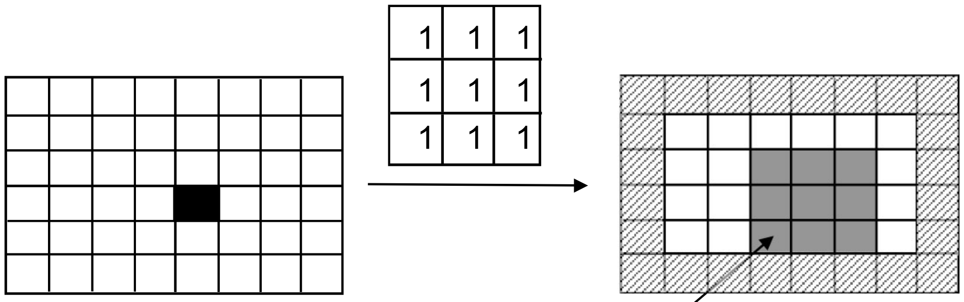
Image Processing - Local Image Operators

Convolution: **Mean value operator**

Forms the mean/average value from the neighbouring pixel values.

$$F = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Kernel or convolution matrix



Input image is 8*6 pixels in size

g'(i,j)= 255*8/9 = 227

Image Processing - Local Image Operators

Convolution: **different blurs - mean operator & Gaussian filter**

Fmean =

1	1	1
1	1	1
1	1	1

FGauss =

1	2	1
2	4	2
1	2	1

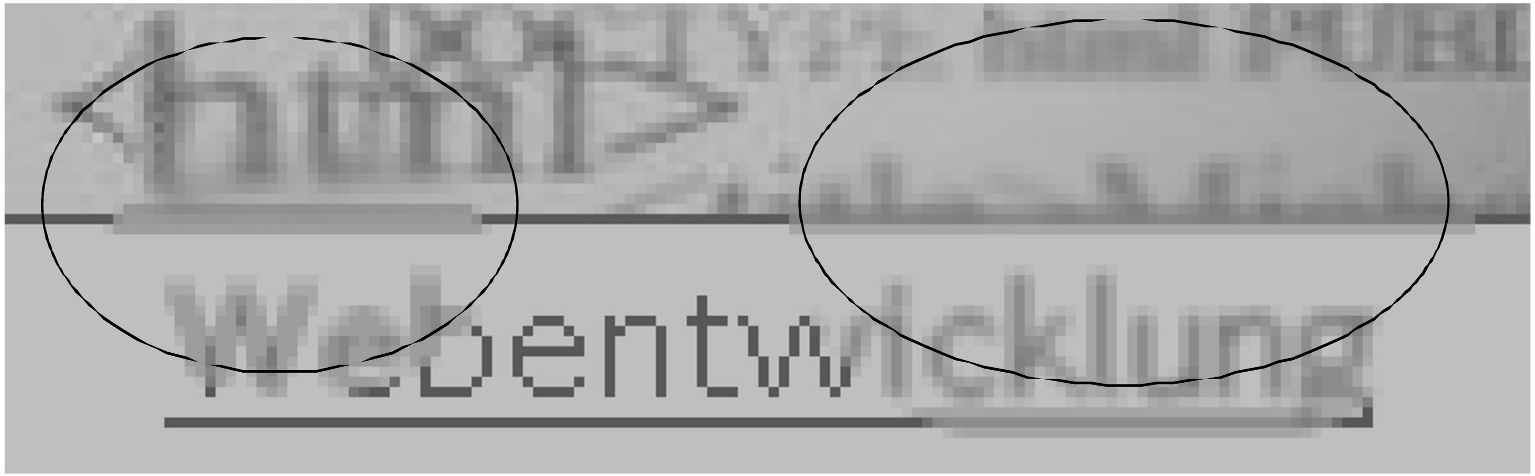


Image Processing - Local Image Operators

The calculation with the mean value operator or the Gaussian filter results in convolution results e(i, that lie outside the grey value range [0, ... 255].

To transform the convolution results back into the grey value range, a linear mapping to the value range [0, ... 255] must be carried out:

- Transformation after convolution with the mean value operator: $g'(i, j) = 1/9 \cdot e(i, j) + 0$
- Transformation after convolution with the Gaussian filter: $g'(i, j) = 1/16 \cdot e(i, j) + 0$

Image Processing - Local Image Operators

Folding coloured images / RGB images

1. Separating the different RGB colour channels
2. For each colour channel: filter corresponding greyscale image
3. Combine filtered images of the colour channels to form the colour image

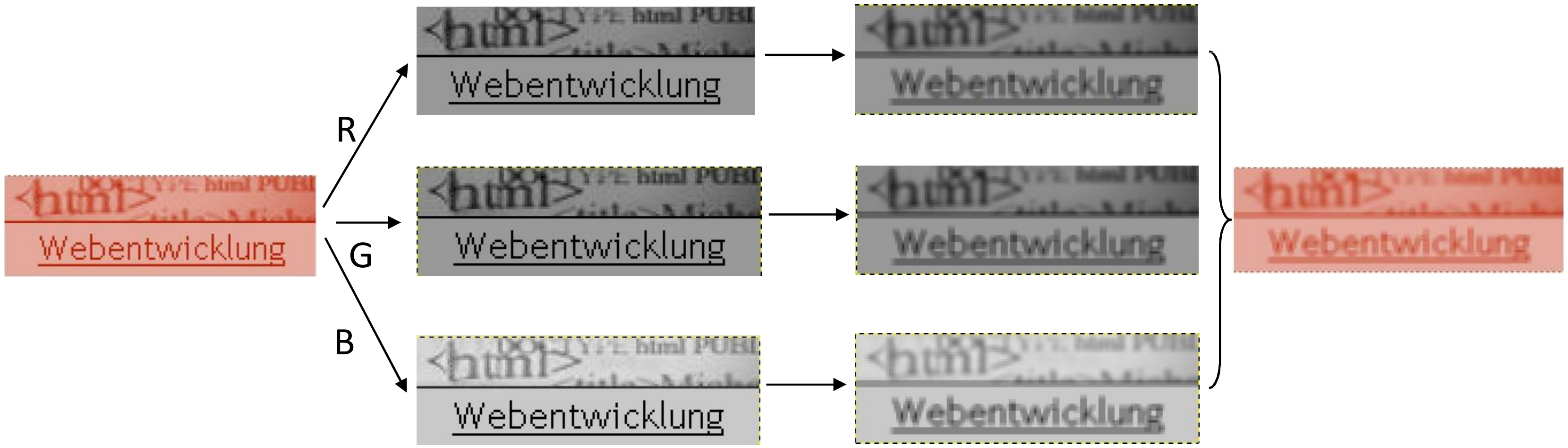


Image Processing - Local Image Operators

What effect do the following filter cores have?

$$\frac{1}{3}\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Mittelwertfilter in y-Richtung

$$\frac{1}{3}\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Mittelwertfilter in x-Richtung

From: "**Multimedia Signal Processing Image Processing: Filters",** Thorsten Thormählen https://www.mathematik.uni-marburg.de/~thormae/lectures/mmk/mmk_6_1_ger_web.html#1

Image Processing - Local Image Operators

Filter kernels can also have different sizes. The kernel size used depends on the resolution of the input image and the desired effect.

Ex. Gaussian filter:

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \quad \frac{1}{256}\begin{bmatrix}1 & 4 & 6 & 4 & 1\\ 4 & 16 & 24 & 16 & 4\\ 6 & 24 & 36 & 24 & 6\\ 4 & 16 & 24 & 16 & 4\\ 1 & 4 & 6 & 4 & 1\end{bmatrix} \quad \ldots$$

Original

3 x 3

5 x 5

7 x 7

21 x 21

From: "**Multimedia Signal Processing Image Processing: Filters",** Thorsten Thormählen https://www.mathematik.uni-marburg.de/~thormae/lectures/mmk/mmk_6_1_ger_web.html#1

Image Processing - Local Image Operations

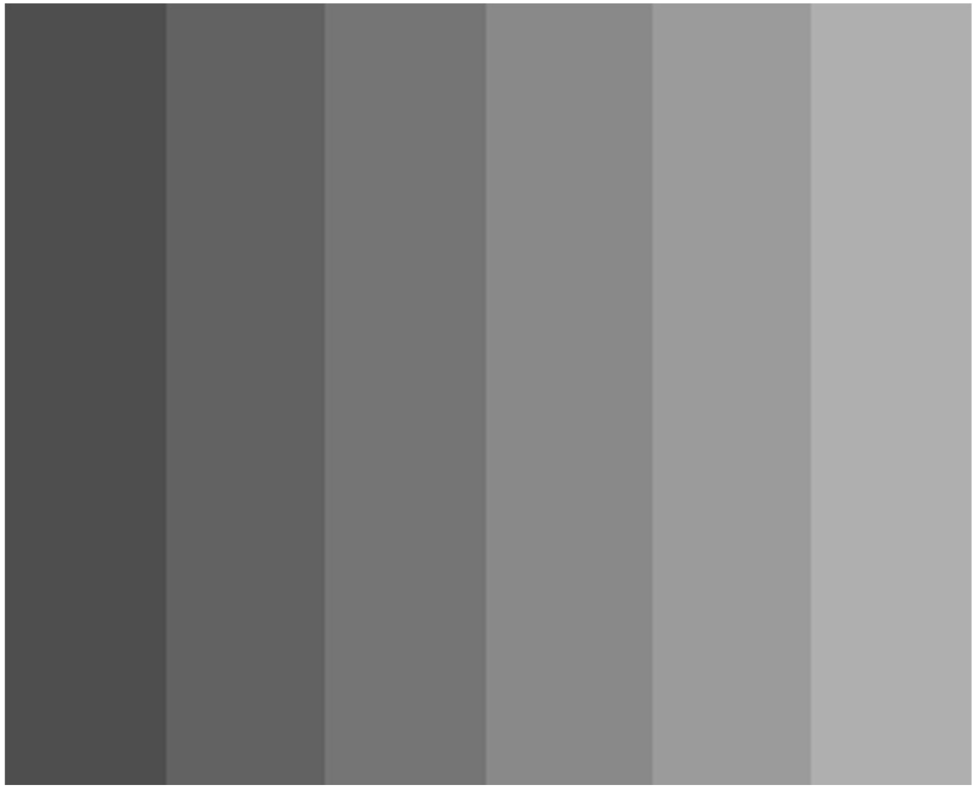
Point operators compared to local image operators Local image operators

- Blur: Mean operator, Gaussian filter
- Edge detectors: difference filter and Sobel operator, Laplace operator
- Contrast enhancement filter
- Ranking operators: Erosion, Dilation, Median as well as Opening and Closing
- Segmentation procedure

Image Processing - Local Image Operators

Our perception system is optimised for "edge detection": **Mach band effect**

The constant stimuli of the surfaces are muted and contrasts are exaggerated.

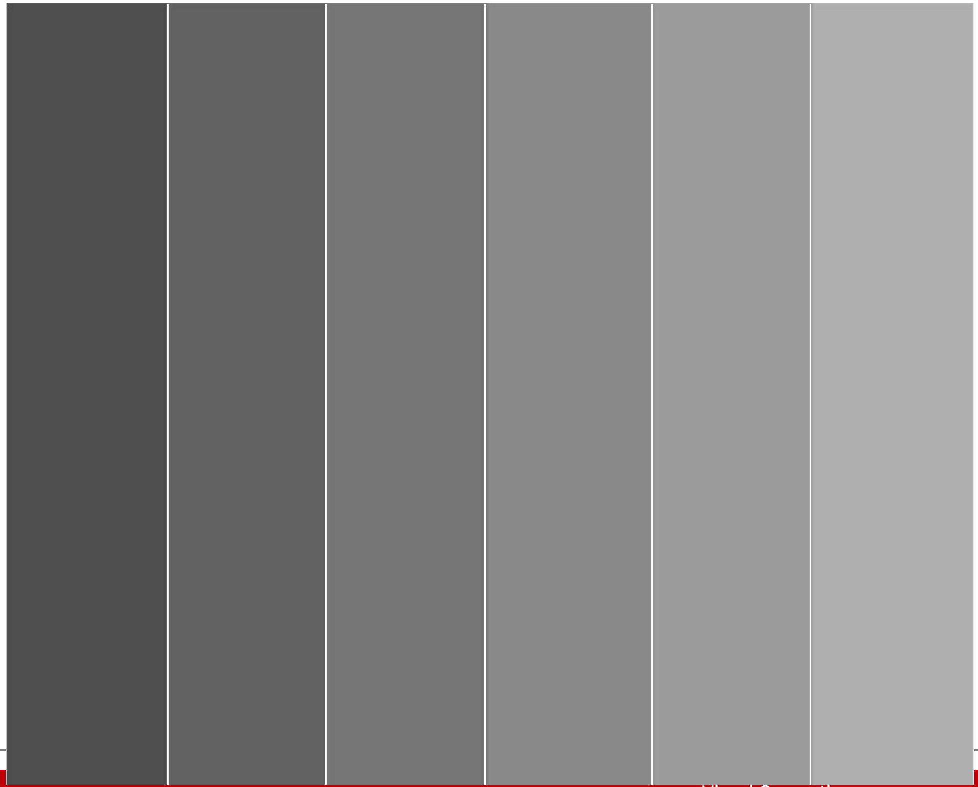


Picture from: https://upload.wikimedia.org/wikipedia/commons/9/97/Bandes_en_mach.PNG

Image Processing - Local Image Operators

Our perception system is optimised for "edge detection": **Mach band effect**

The constant stimuli of the surfaces are muted and contrasts are exaggerated.



Picture from: https://upload.wikimedia.org/wikipedia/commons/9/97/Bandes_en_mach.PNG

Image Processing - Local Image Operators

Our perception system is optimised for "edge detection": **Mach band effect**

Mach band effect can lead to misinterpretations in a radiological diagnosis. Light-dark contrasts are amplified and interpreted as caries, for example.

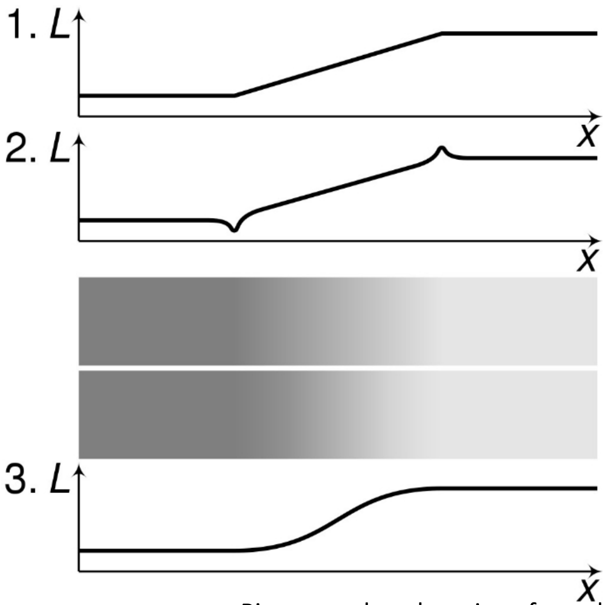
The effect was discovered by Ernst Mach in 1865.

A dark spot appears at this point, which could be interpreted as caries.

Image from: Radiographic projections of the objects, slide 11 https://slideplayer.org/slide/1330107/

Image Processing - Local Image Operators

Mach band effect: Brightness transitions are perceived with a higher contrast than is actually present. The figure describes the effect as a function of the luminance L, which corresponds to the perceived brightness without Mach band effect.



1. Luminance profile of abruptly changing grey values (see A)

2. Perceived brightness profile of A - including the Mach band effect

A. Grey value wedge with abruptly changing grey values

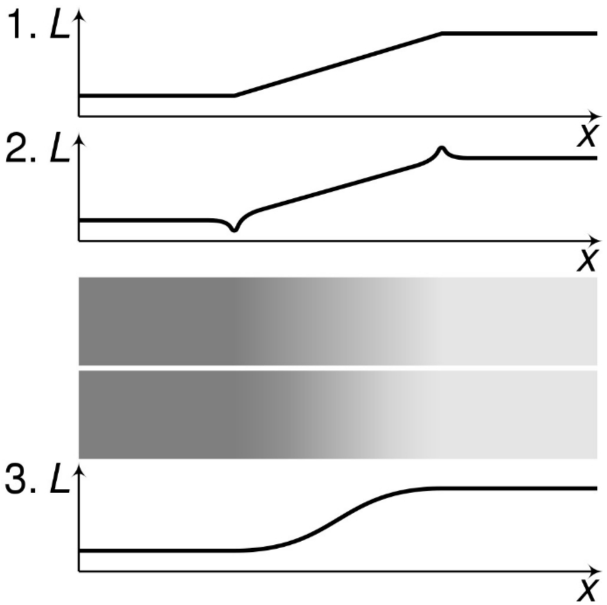
B. Grey value wedge with continuously changing grey values

3. Luminance profile of the continuously changing grey values (see B)

Picture and explanations from: https://en.wikipedia.org/wiki/Mach_bands#/media/File:Mach_bands_gradient_overshoot.svg

Image Processing - Local Image Operators

Mach band effect: Brightness transitions are perceived with a higher contrast than is actually present. The figure describes the effect as a function of the luminance L, which corresponds to the perceived brightness without Mach band effect.



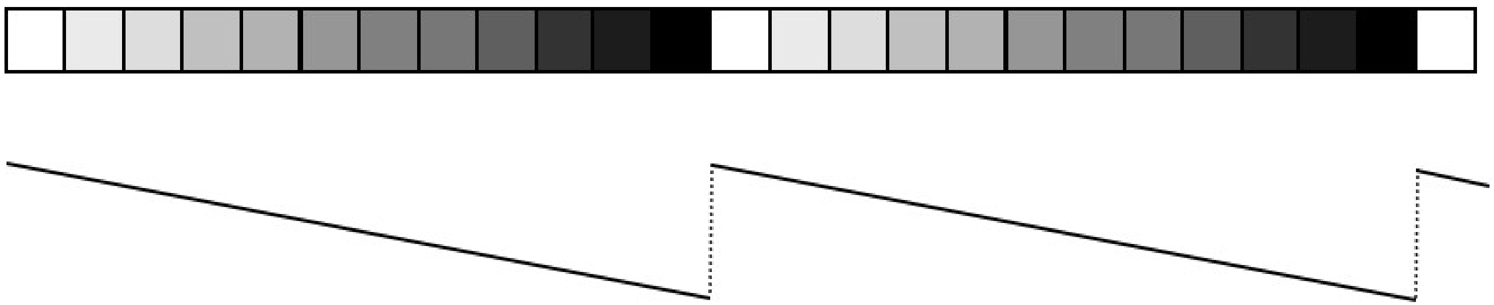
This effect was translated mathematically in order to develop a convolution operator for contrast enhancement. In other words, a convolution operator that amplifies edges in the way we are used to from our perception.

Picture and explanations from: https://en.wikipedia.org/wiki/Mach_bands#/media/File:Mach_bands_gradient_overshoot.svg

Image Processing - Local Image Operators

Convolution: **difference operator** - recognises edges.

1. Step for deriving the difference operator: Instead of an image, only one image line is considered!



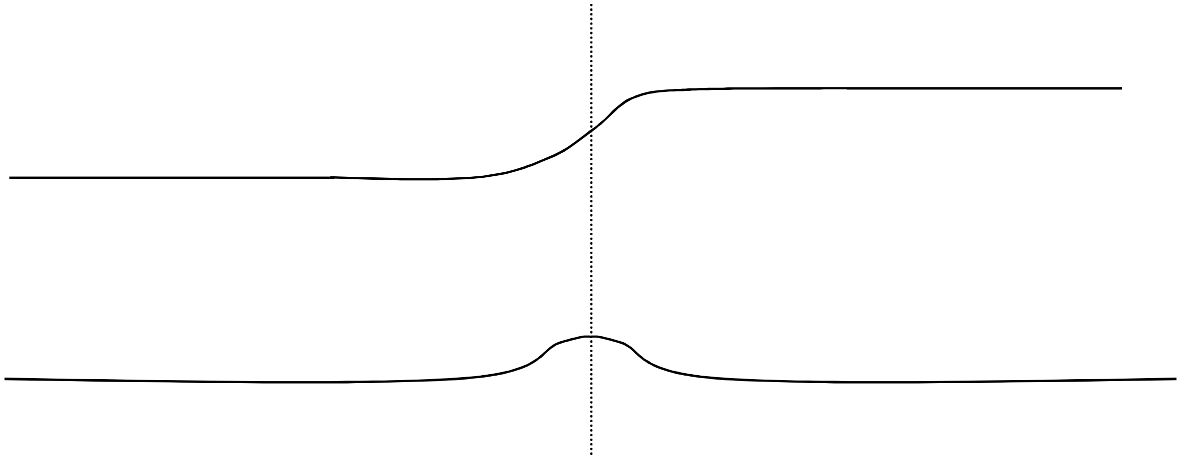
Grey value profile of the image line

Function s(x), which represents the grey value progression along the image line.

Image Processing - Local Image Operators

Convolution: **difference operator** - recognises edges.

2. Step to derive the difference operator: Derive the function s(x) to find the edge.



Grey value gradient of an image line s(x)

Calculate the 1st derivative s'(x)

In the 1st derivative, the non-zero values indicate the positions of the edges in the image line.

Image Processing - Local Image Operators

Convolution: **difference operator** - recognises edges.

2nd step to derive the difference operator: Derive the function s(x) , to find the edge.

The 1st derivative (slope) is defined by:

$$g'(x) = \lim_{\Delta x \to 0} \frac{g(x+\Delta x) - g(x)}{\Delta x}$$

with g(x) = grey value of the pixel at position x

Image Processing - Local Image Operators

Convolution: **difference operator** - recognises edges.

2nd step to derive the difference operator: Derive the function s(x) , to find the edge.

The 1st derivative (slope) is defined by:

$$g'(x) = \lim_{\Delta x \to 0} \frac{g(x+\Delta x) - g(x)}{\Delta x}$$

with g(x) = grey value of the pixel at position x

Transferred to an image that forms a discrete space due to the pixel representation, this means Δx, the distance between two pixels does not become infinitely small but assumes the value 1 in the minimum case.

The 1st derivative therefore corresponds to the difference between two neighbouring grey values:

$$\frac{g(x+1) - g(x)}{1} = g(x+1) - g(x)$$

Image Processing - Local Image Operators

Convolution: **difference operator**

Detects edges by calculating the difference between the grey values of two neighbouring pixels.

$$FDy=\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & -1 & 1 \\ \hline 0 & 0 & 0 \\ \hline\end{array}$$

Image Processing - Local Image Operators

Convolution: **difference operator**

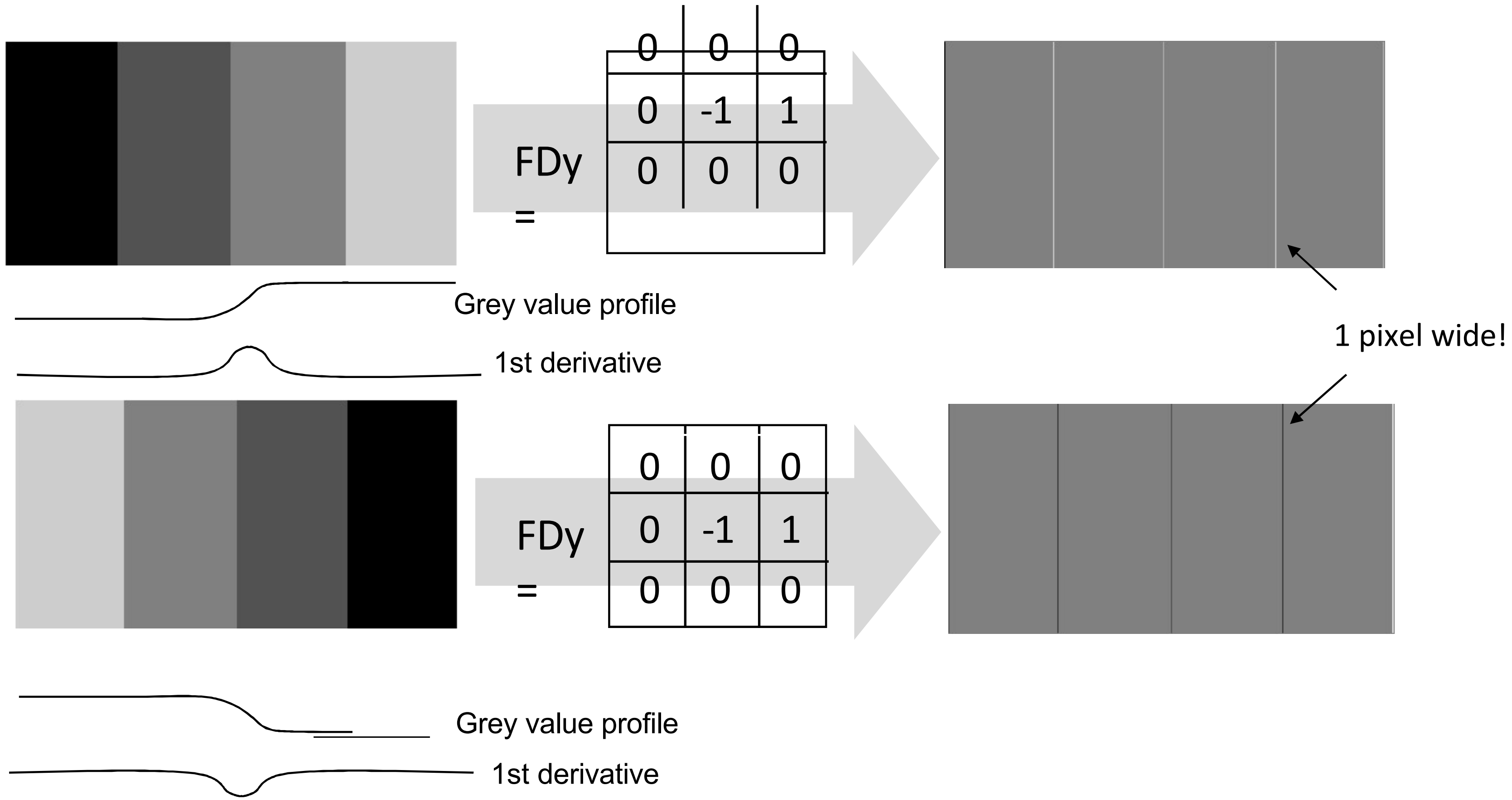


Image Processing - Local Image Operators

How do these edge images arise? The difference operator forms on a

Value range e(i, j) ∈ {-255, ... , 255} from

Convolution: **difference operator**

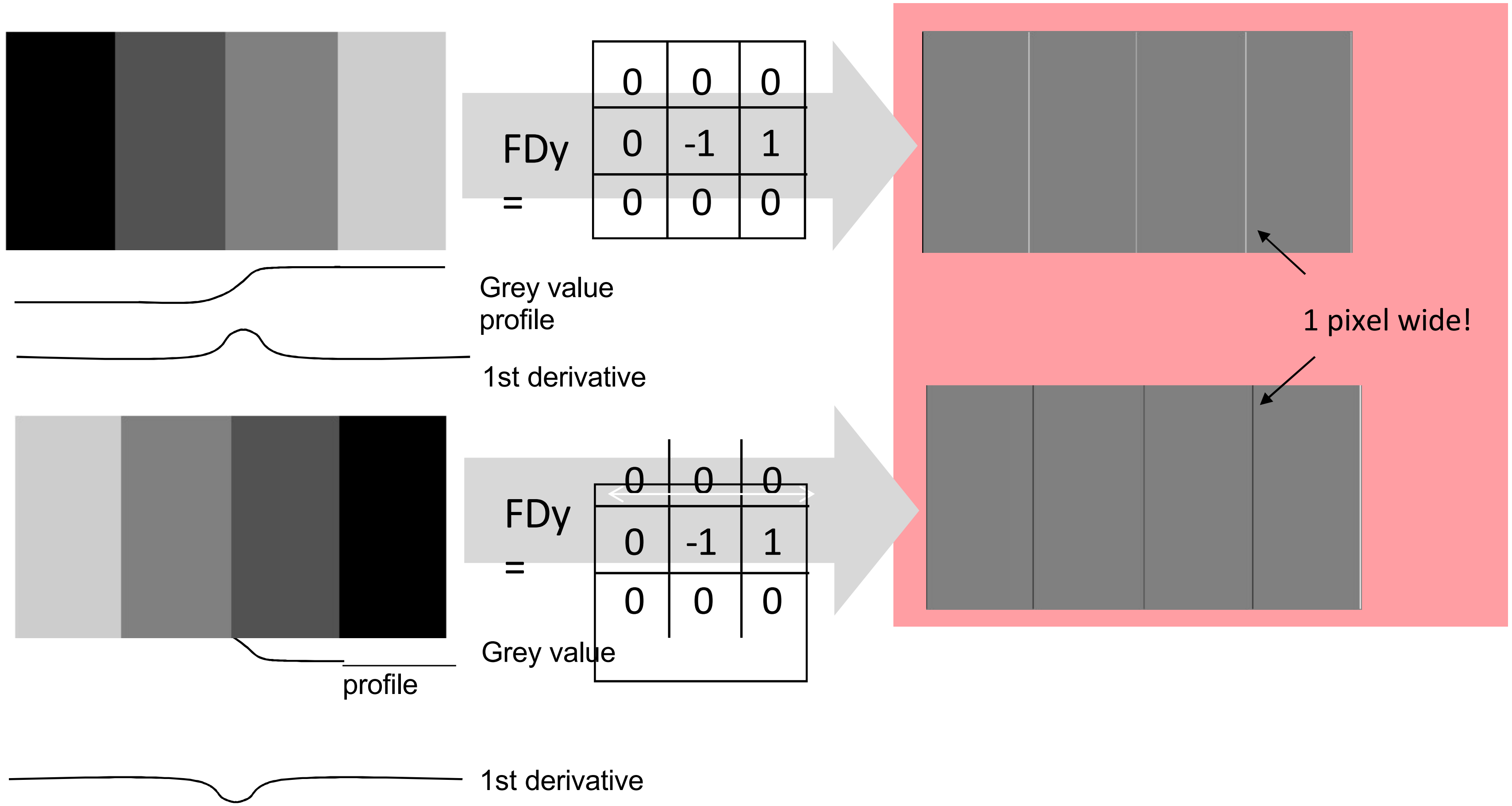


Image Processing - Local Image Operators

Convolution: **difference operator**

What to do if the results of the convolution are partially negative?

- After convolution with the difference operator, result values e(i, j) ∈ {-255, ... , 255} are obtained.
- Thus, a linear mapping is needed that maps the range of values of e(i, j) to therange g'(i, j) ∈ ... , 255}
- This linear mapping function is: g'(i,j)= e(i,j)· *0.5 + 127*

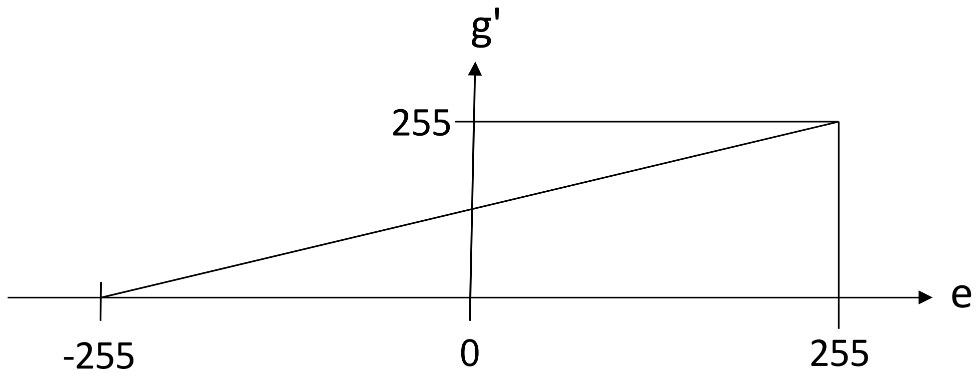


Image Processing - Local Image Operators

Convolution: **difference operator**

By applying the linear mapping function g'(i,j)= e(i,j) · 0.5 + 127 the convolution results are transformed into the representable grey value range.

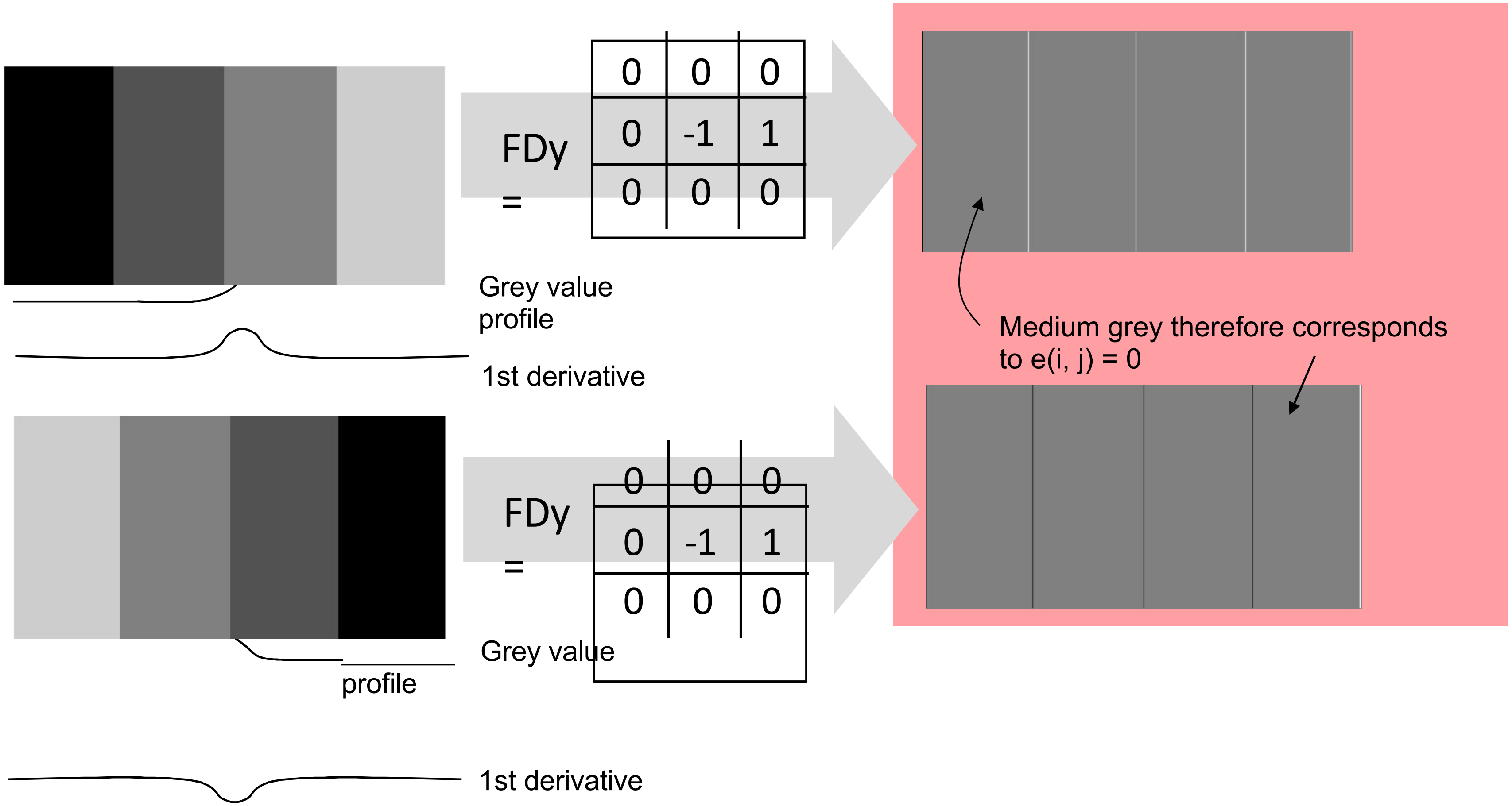
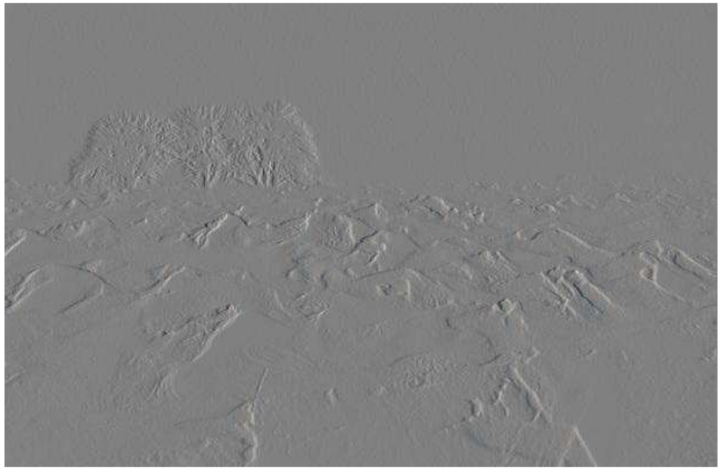


Image Processing - Local Image Operators

Convolution: **difference operator**

Detects edges by calculating the difference between the values of two neighbouring pixels.

$$FDy = \begin{array}{|c|c|c|} 0 & 0 & 0 \\ \hline 0 & -1 & 1 \\ \hline 0 & 0 & 0 \end{array}$$



In this example, only the vertically running edges were displayed.

What must a difference operator look like that finds the horizontally running edges?

Image Processing - Local Image Operators

Convolution: **difference operator**

Detects the horizontal edges by calculating the difference between the values of two neighbouring pixels in an image column.

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

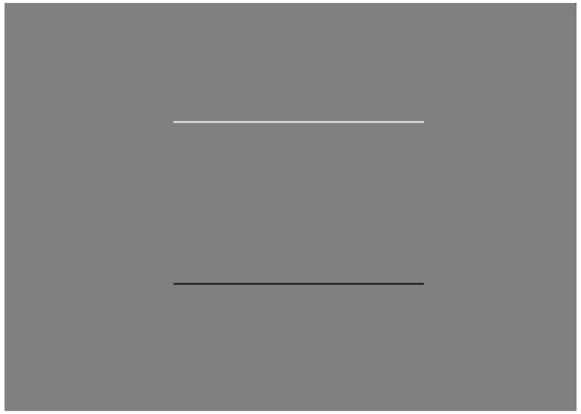
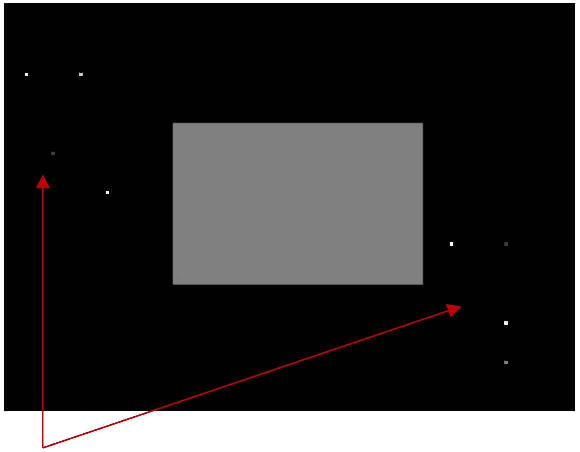


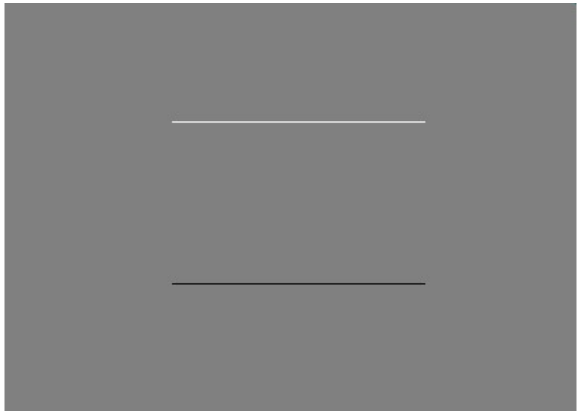
Image Processing - Local Image Operators

Convolution: **difference operator**

In order to make the edge detection robust and not to display an edge for every "faulty" pixel, the neighbouring pixels are included in the calculation.



$$FDy = \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$



Defective pixels become noticeable through noise in the image.

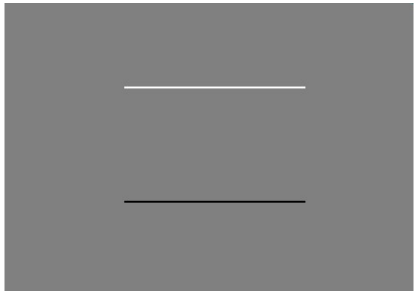
Image Processing - Local Image Operators

Convolution: **difference operator**

To make edge detection even more robust, a "buffer" line or column is inserted into the operator. This type of difference operator is called a Sobel operator.

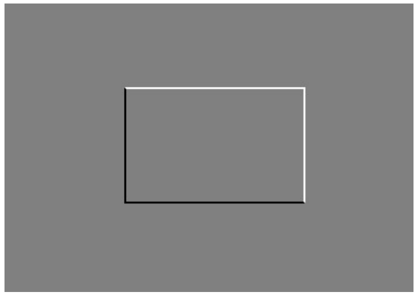
Convolution results of the different Sobel operators

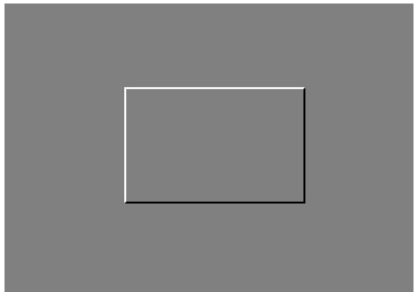
-1	0	1
-1	0	1
-1	0	1



-1	-1	-1
0	0	0
1	1	1

0	-1	-1
1	0	-1
1	1	0





-1	-1	0
-1	0	1
0	1	1

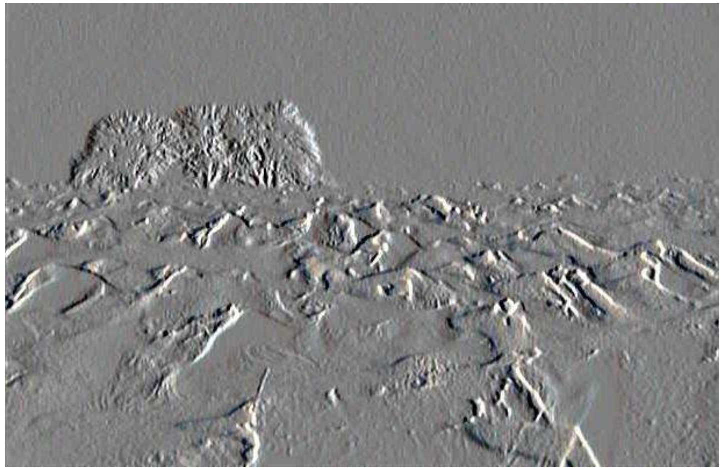
Image Processing - Local Image Operators

Convolution: **difference operator**

To make edge detection even more robust, a "buffer" line or column is inserted into the operator. This type of difference operator is called a Sobel operator.

Through the "buffer" column or row, the Sobel operator marks the edges with several pixels.

-1	0	1
-1	0	1
-1	0	1



-1	-1	-1
0	0	0
1	1	1

Image Processing - Local Image Operators

Convolution: Insert the result of the **difference operator** into the input image:

After applying an edge operator, only edges are visible in the result image. By adding the original image, a filter F is obtained that creates edges in the input image.

$$F = n +\cdot \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} \quad \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 1 & 1 \\ \hline \end{array} = \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline 0 & n & 0 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

F = n ·

+

=

Simplified representation without taking into account Consideration of the linear transformation

Image Processing - Local Image Operators

Convolution: Insert the result of the **difference operator** into the input image:

$$F = \begin{array}{|c|c|c|} \hline -1 & -1 & -1 \\ \hline 0 & n & 0 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

$$F = \begin{array}{|c|c|c|} \hline -1 & -1 & 0 \\ \hline -1 & n & 1 \\ \hline 0 & 1 & 1 \\ \hline \end{array}$$

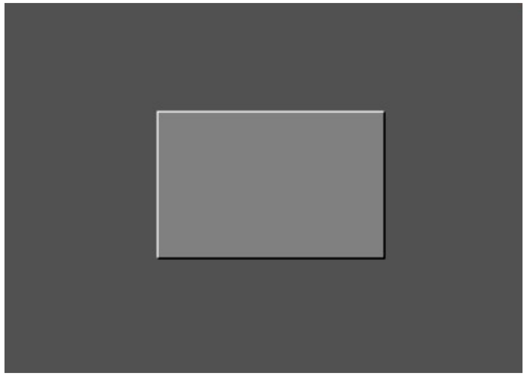


Image Processing - Local Image Operators

Application of the filter F with different n:

$$F = \begin{array}{|c|c|c|} \hline -1 & -1 & 0 \\ \hline -1 & n & 1 \\ \hline 0 & 1 & 1 \\ \hline \end{array}$$

Image Processing - Local Image Operators

Convolution: **Laplace operator** - recognises edges using the 2nd derivative.

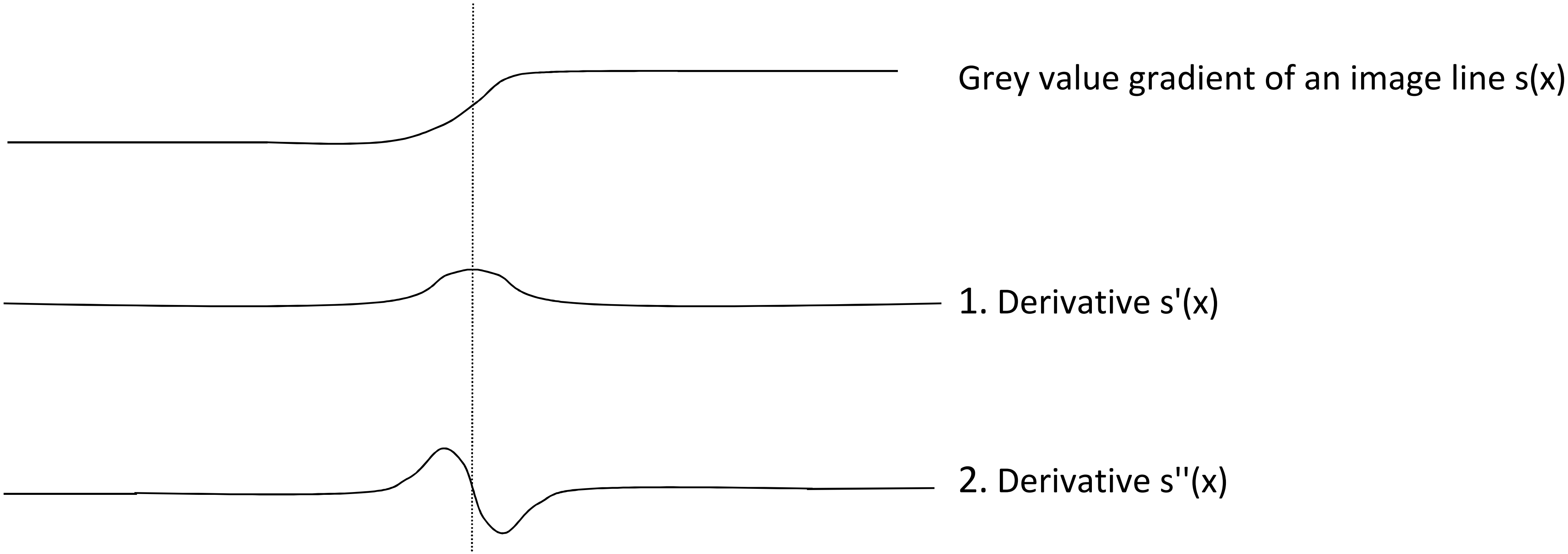


Image Processing - Local Image Operators

Convolution: **Laplace operator** - recognises edges using the 2nd derivative.

The 2nd derivative (slope) is defined by:

$$g''(x) = \lim_{\Delta X \to 0} \frac{(g(x + \Delta x) - g(x)) - (g(x) - g(x - \Delta x))}{\Delta X}$$

with ***g(x)*** = grey value of the pixel at position *x*

The following applies to raster images: Δx assumes the value 1 in the minimum case.

The 2nd derivative in a picture row or column: $g''(x) = g(x+1) - 2 \cdot g(x) + g(x-1)$

Image Processing - Local Image Operators

Convolution: **Laplace operator** - recognises edges using the 2nd derivative.

The 2nd derivative in a picture row or column: $g''(x) = g(x+1) - 2 \cdot g(x) + g(x-1)$

Laplace operator = $\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 1 & -2 & 1 \\ \hline 0 & 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 0 & -2 & 0 \\ \hline 0 & 1 & 0 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$

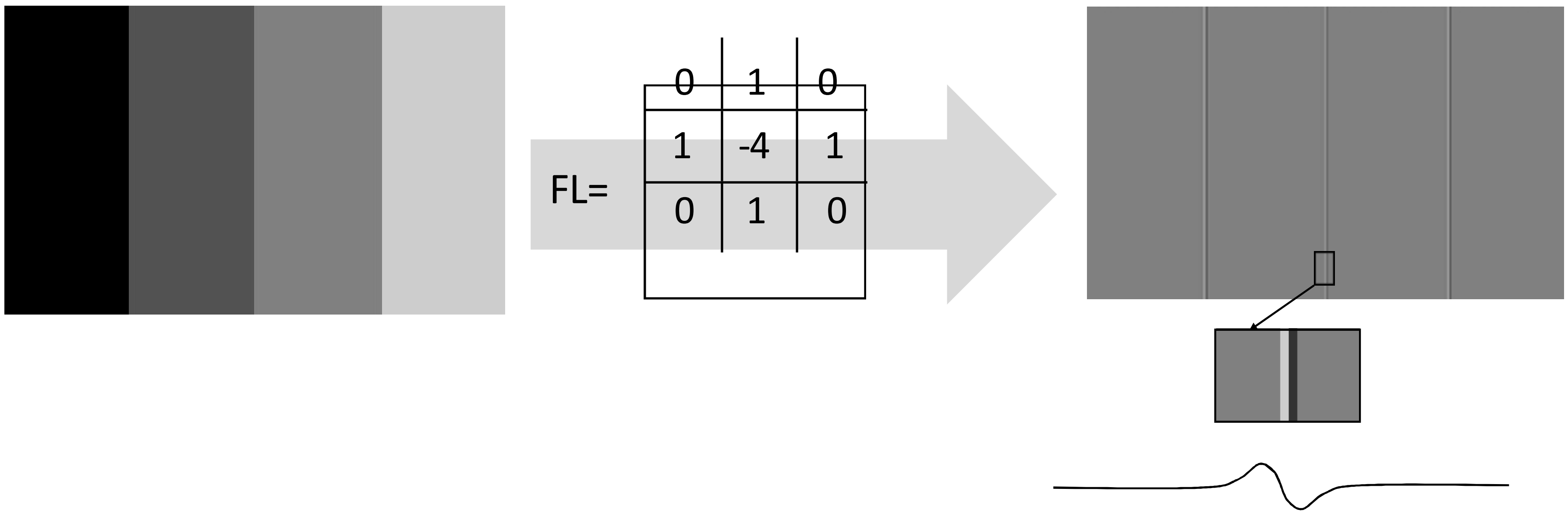
Edge detection in the image line

Edge detection in the image column

Edge detection in both the image column and image row

Image Processing - Local Image Operators

Convolution: **Laplace operator** - recognises edges using the 2nd derivative.

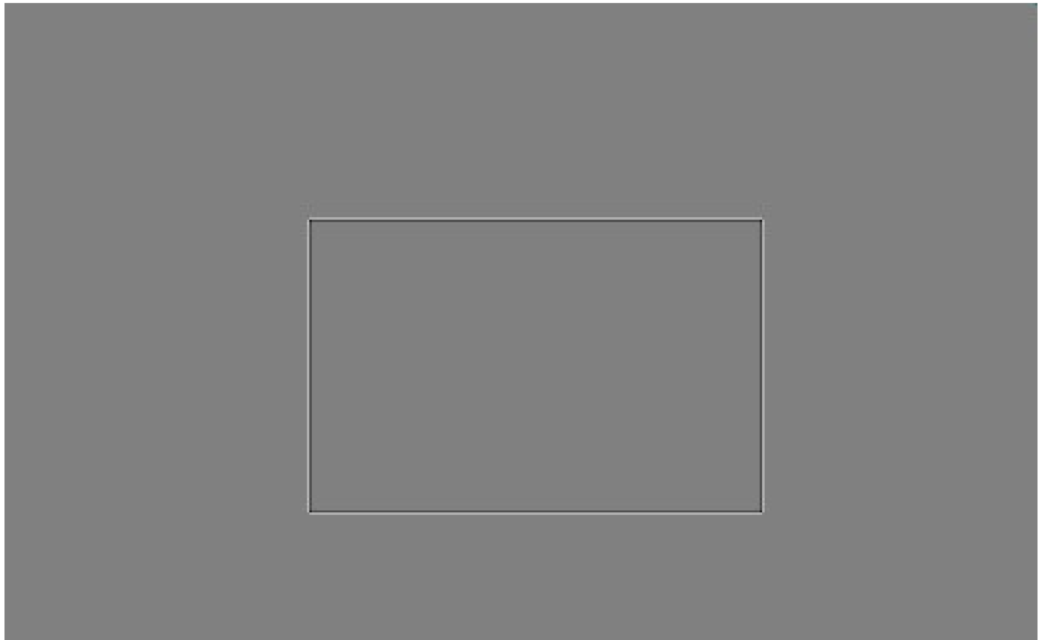


Corresponds to the course of the 2nd derivative

Image Processing - Local Image Operators

Convolution: **Laplace operator** - recognises edges using the 2nd derivative.

$$FL = \begin{array}{|c|c|c|} 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \end{array}$$



Edge image enlarged to make the edges more visible.

Image Processing - Local Image Operators

Convolution: **Laplace operator** - recognises edges using the 2nd derivative.

$$FL = \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$$

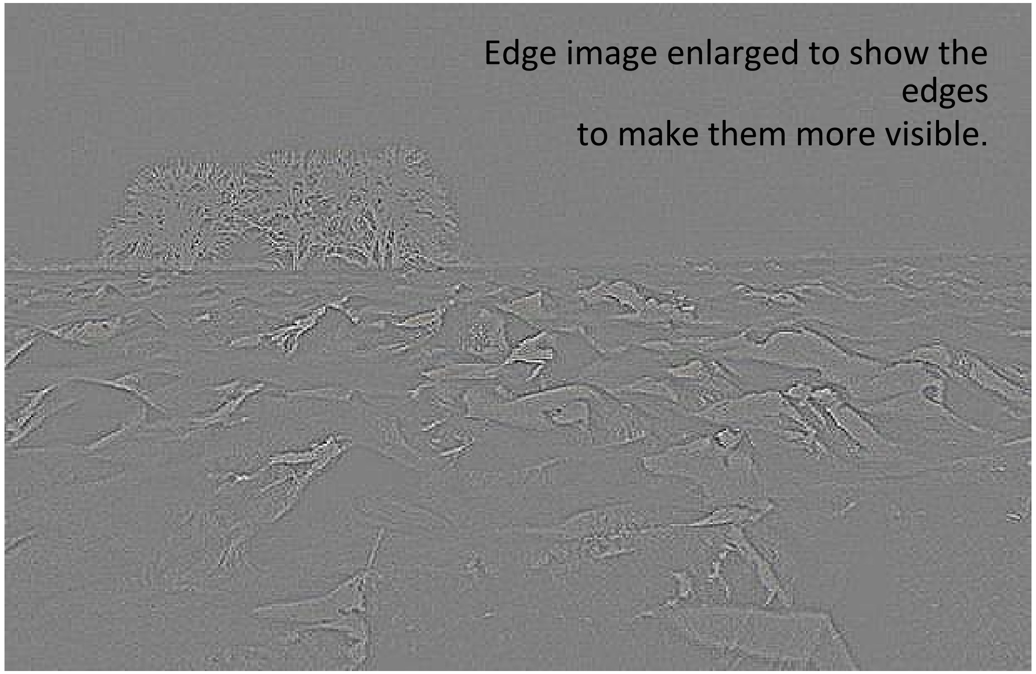
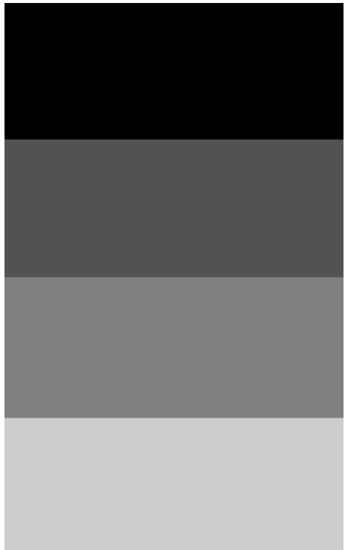


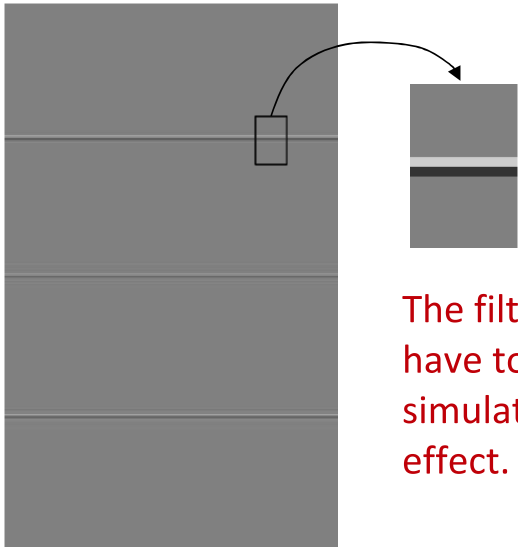
Image Processing - Local Image Operators

Convolution: from the **Laplace operator** to the **contrast enhancement filter**

A contrast enhancement filter emphasises the edge on the light side of the edge with lighter pixels and on the dark side of the edge with darker pixels - equivalent to the Mach band effect.



$$FL=\begin{array}{|c|c|c|}\hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline\end{array}$$

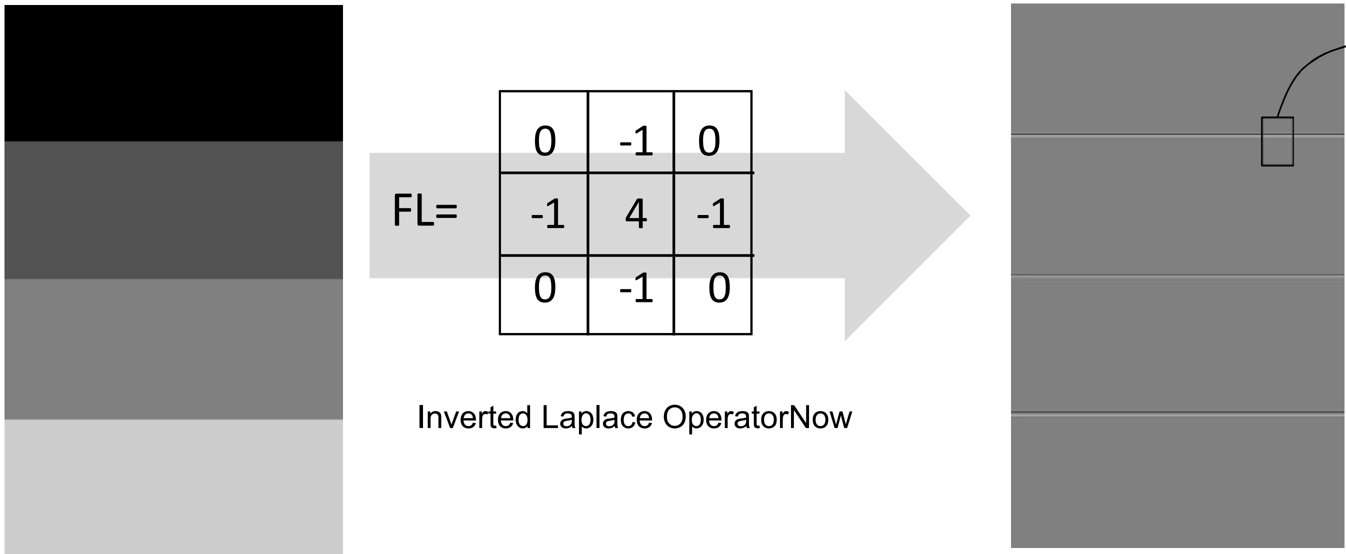


The filter result would have to be inverted to simulate the Mach band effect.

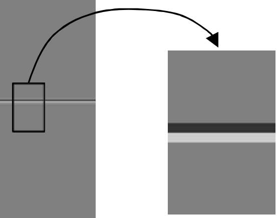
Image Processing - Local Image Operators

Convolution: from the **Laplace operator** to the **contrast enhancement filter**

A contrast enhancement filter emphasises the edge on the light side of the edge with lighter pixels and on the dark side of the edge with darker pixels - equivalent to the Mach band effect.



Inverted Laplace OperatorNow



only the input image has to be integrated into the convolution result.

Image Processing - Local Image Operators

Convolution: from the **Laplace operator** to the **contrast enhancement filter**

A contrast enhancement filter emphasises the edge on the light side of the edge with lighter pixels and on the dark side of the edge with darker pixels - equivalent to the Mach band effect.

The trick to integrate the input image into the convolution result is already known:

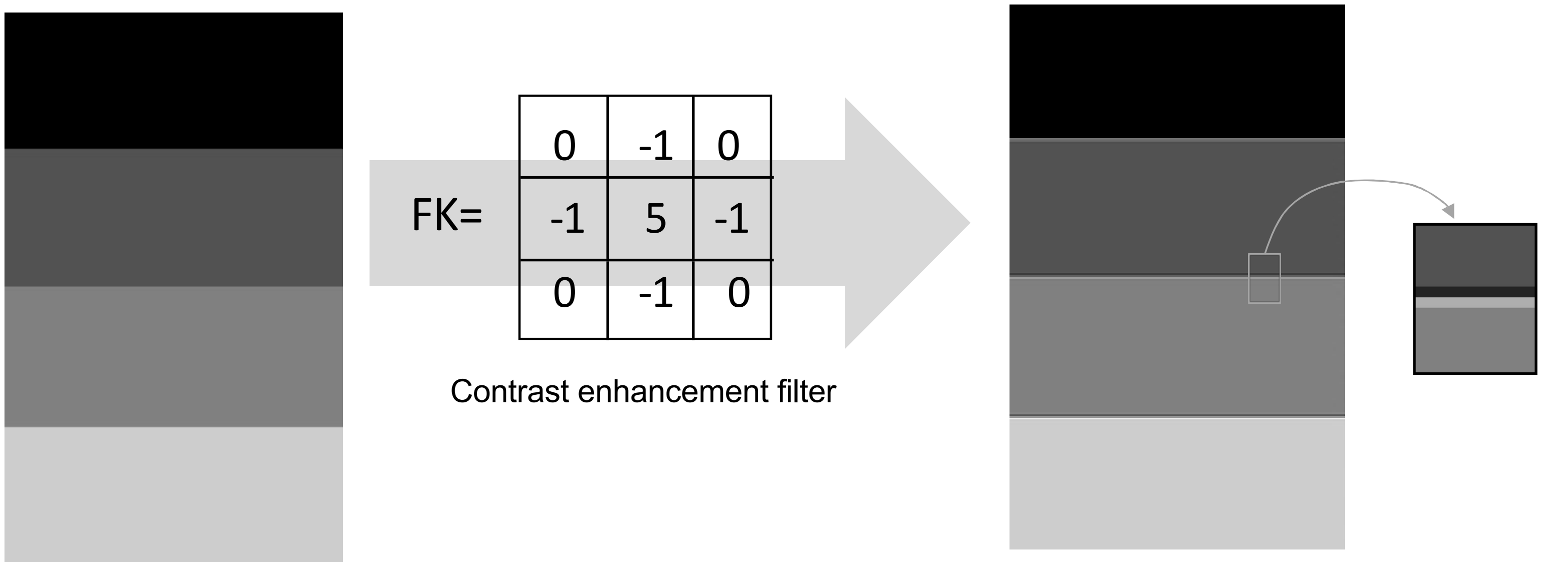
$$_{FK} = n \cdot \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array} = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & n+4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

Contrast enhancement filter

Image Processing - Local Image Operators

Convolution: from the **Laplace operator** to the **contrast enhancement filter**

A contrast enhancement filter emphasises the edge on the light side of the edge with lighter pixels and on the dark side of the edge with darker pixels - equivalent to the Mach band effect.



Contrast enhancement filter

Image Processing - Local Image Operators

Convolution: from the **Laplace operator** to the **contrast enhancement filter**

A contrast enhancement filter emphasises the edge on the light side of the edge with lighter pixels and on the dark side of the edge with darker pixels - equivalent to the Mach band effect.

$$FK = \begin{array}{|c|c|c|} \hline 0 & -1 & 0 \\ \hline -1 & 5 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

Contrast enhancement filter

Image Processing - Local Image Operations

Point operators compared to local image operators Local image operators

- Blur: Mean operator, Gaussian filter
- Edge detectors: difference filter and Sobel operator, Laplace operator
- Filter for contrast enhancement - already discussed with the edge detectors
- Ranking operators: Erosion, Dilation, Median as well as Opening and Closing
- Segmentation procedure

Image Processing - Local Image Operations

How are the filters used? Example lane detection.

Please click on the black border to start the video.

From:
https://youtu.be/IzvlqCEYjg4

Image Processing - Local Image Operations

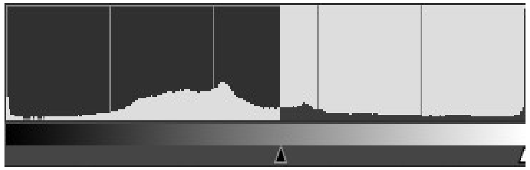
How are the filters used? Example lane detection.

- The aim is to find the red lane markings drawn in the JPEG image (left).
- The known filters are typically used to preprocess the image or video.

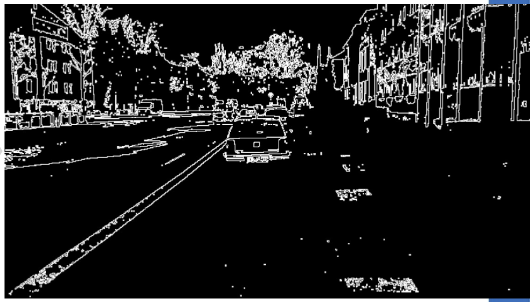
Image compressed with JPEG

1. Conversion into a greyscale image by extracting the brightness channel

2. Biniarisation



In the example image, the optimal threshold value is approx. 135



3. Edge extraction
   Here the Sobel operator was used

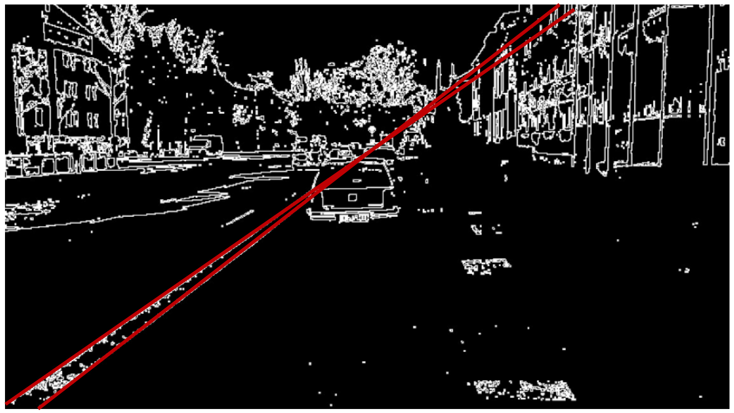
Image from
https://davidgruenewald.de/category/darmstadt/

Image Processing - Local Image Operations

How are the filters used? Example lane detection.

- The next step is to assign a content-related meaning to the pixels.
- In the lane detection example, only the edges are of interest.
- More precisely: the lines that are not perpendicular and that combine the most pixels.
- The **Hough Transformation is** suitable for this task

Input: **Edge image**



Result: extracted straight lines that are not vertical and have the most pixels. combine

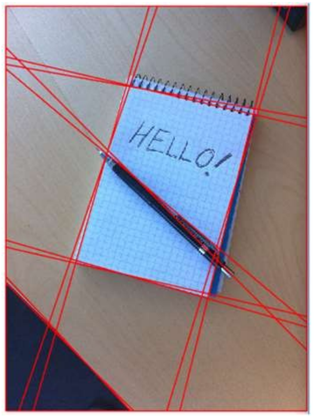
Original image from https://davidgruenewald.de/category/darmstadt/

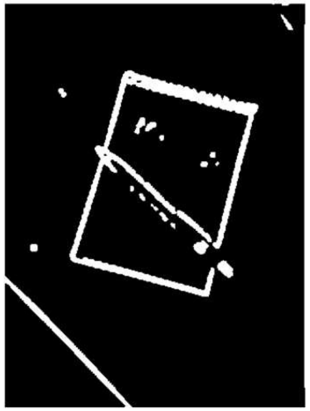
Image Processing - Local Image Operations

How are the filters used? Example lane detection.

The Hough transformation searches for straight lines in an edge image.

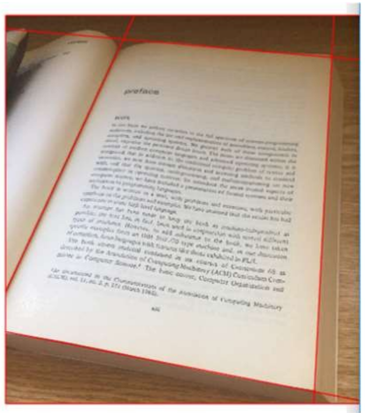
A.) Result of the Hough transformation based on an edge image whose edges are several pixels wide.

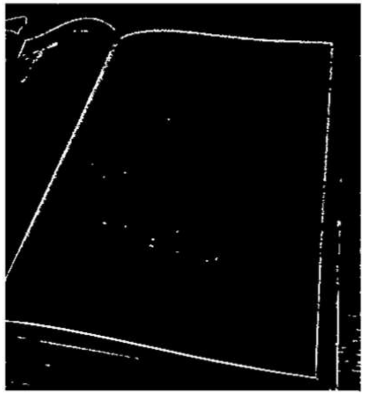




Two straight lines are recognised per edge, which combine approximately the same number of pixels in pairs.

B.) Result of the Hough transformation based on an edge image whose edges are <u>only one pixel wide.</u>





Only one straight line is recognised at a time.

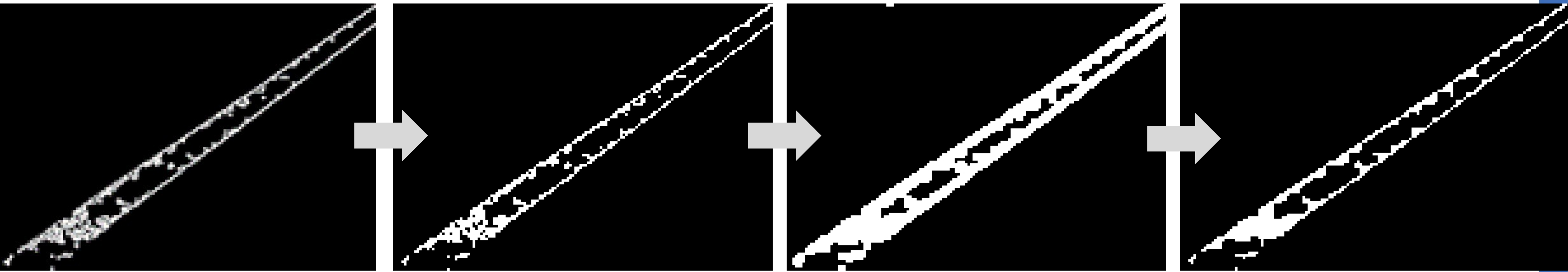
Conclusion: In pre-processing, care must be taken to ensure that the edge is only one pixel wide.

Images from: https://stackoverflow.com/questions/41474719/find-corners-of-a-page-after-applying-hough-transformation

Image Processing - Local Image Operations

How are the filters used? Example lane detection.

How can edges be thinned out? For example, through rank order operators.



Edge

imageBiniarisation of the edge image (optional, not absolutely necessary)

Dilatation = expansion of the foreground objects. The white pixels expand by one layer.

Erosion = The white pixel areas shrink by one pixel layer.

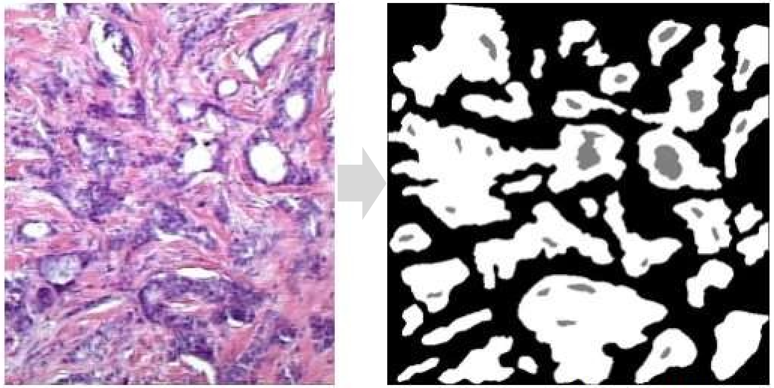
In practice, the Canny Edge Detector is used. It thins almost all edges to a thickness of one pixel. **The rank order operators are often used to prepare the segmentation**.

Image Processing - Local Image Operations

Definition: Segmentation

Segmentation is the interpretation of the content of an image at the pixel level [1]. Image segmentation recognises which pixels belong to an object based on the properties of the pixels. Colours or grey values and the neighbourhood of the pixels to each other are examples of properties that describe a pixel. Segmentation thus groups pixels with homogeneous properties [2].

Medical example image "Breast cancer tissue" [2]



Originalbild segmentiertes Bild

Segmentation of an aerial photograph, from

Originalbild segmentiertes Bild

[1] https://de.mathworks.com/solutions/image-video-processing/semantic-segmentation.html
[2] https://www.mathematik.uni-ulm.de/stochastik/lehre/ws05_06/seminar/ausarbeitung_sauter.pdf

Image Processing - Local Image Operations

Do "segmentation" and "semantic segmentation" mean the same thing?

Both are interpretations of the content of an image at the pixel level [1]. Semantic segmentation also divides the image into groups (classes). However, it is not the properties of a pixel that are decisive for the assignment to a class, but the affiliation of the pixel to an object class, such as tree, car or similar.

Example of semantic segmentation from [1]

belongs to the class Background (black)

Tree

Roadway

[1] https://de.mathworks.com/solutions/image-video-processing/semantic-segmentation.html

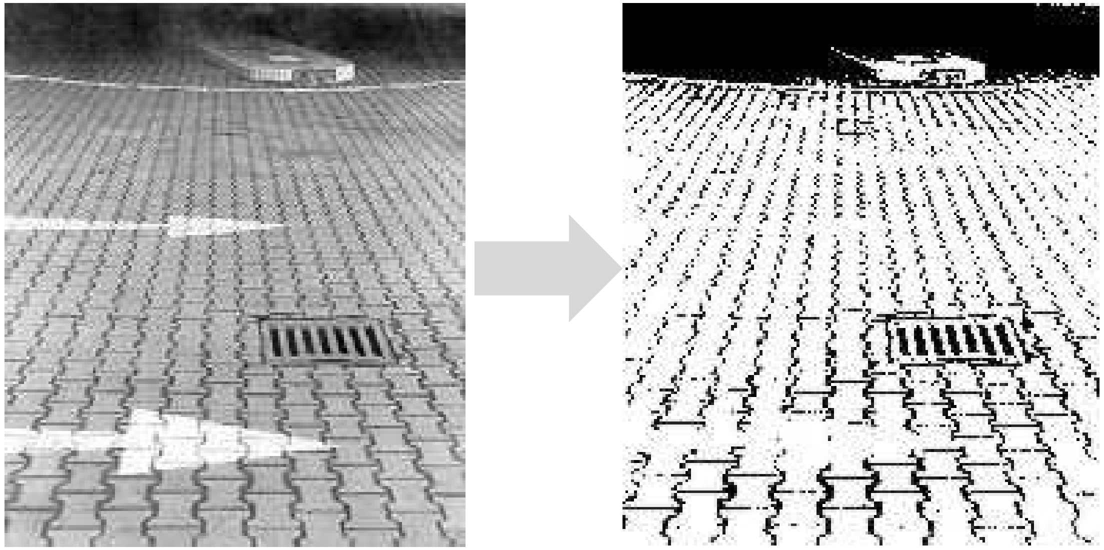
Image Processing - Local Image Operations

Segmentation: The task is to assign the pixels of the large arrow in the image foreground to a segment and to combine all other pixels as a background segment.

Image Processing - Local Image Operations

Segmentation: The task is to assign the pixels of the large arrow in the image foreground to a segment and to combine all other pixels as a background segment.

Step 1:



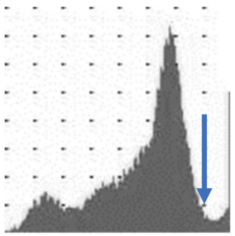
Biniarisation with threshold value 127

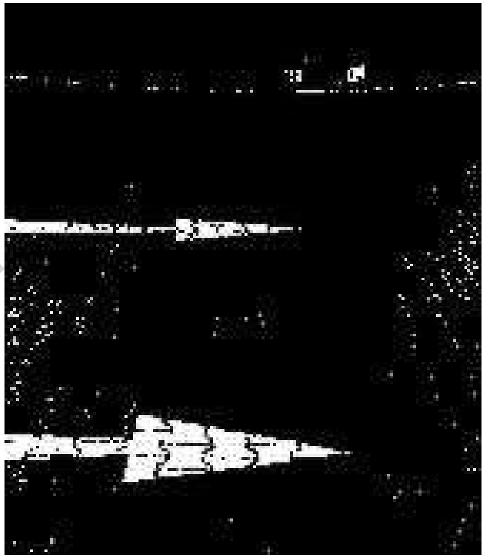
Image Processing - Local Image Operations

Segmentation: The task is to assign the pixels of the large arrow in the image foreground to a segment and to combine all other pixels as a background segment.

Step 1:

Histogram

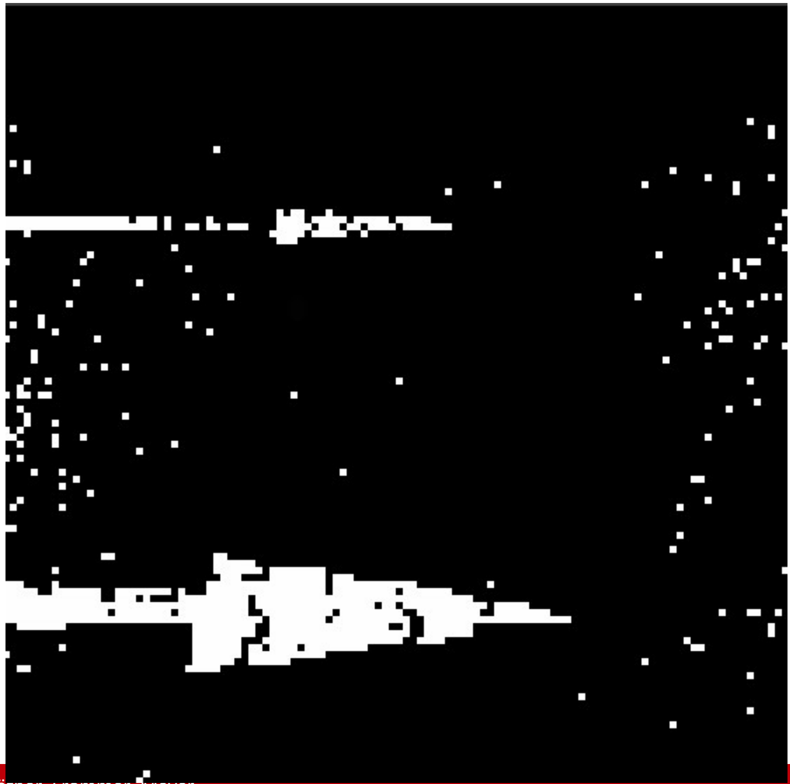




Biniarisation with the threshold value 225, which was determined on the basis of the histogram. 225 is the grey value that comes after the first minimum in the histogram, starting from white.

Image Processing - Local Image Operations

Segmentation: The task is to assign the pixels of the large arrow in the image foreground to a segment and to combine all other pixels as a background segment.

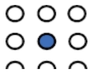


Remark:
The resolution of the binarised image was reduced to better show how the rank order operators work.

Question: How are rank order operators calculated?

Image Processing - Local Image Operations

Calculation of the rank order operators:

- the greyscale image is scanned pixel by pixel with a kernel, similar to convolution, but the kernel is now called a structural element and can take on all possible shapes.
- Calculation is exemplified by the structural element corresponding to the N8 neighbourhood: 
- the pixel *g(i,j)* to be transformed is in the middle of the structural element (marked blue)

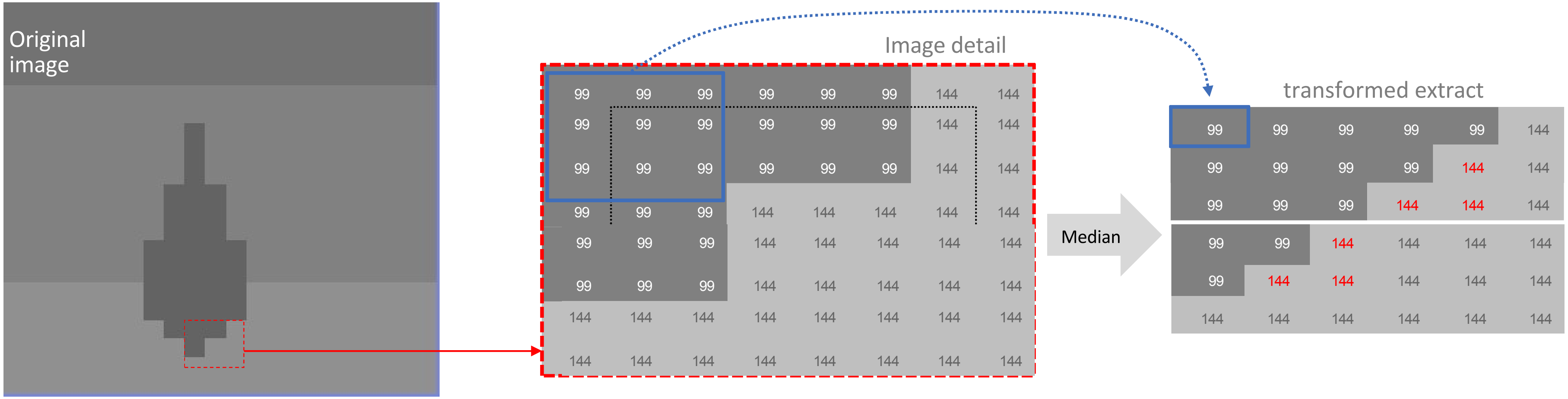
For any pixel transformation with a rank operator, the following applies:

- the grey values of the pixels covered by the structural element are sorted by size:
  $g0 \le g1 \le \ldots \le gn$
- Depending on the ranking operator, the grey value *g(i,j)* is replaced with a grey value at a specific position in the ranking...
  - For the median operator, this is the mean grey value. For the N8 neighbourhood: $g'(i,j) = g_4$
  - For dilation, this is the largest grey value. For the N8 neighbourhood: $g'(i,j) = g_8$
  - For erosion, this is the smallest grey value. For the N8 neighbourhood: $g'(i,j) = g_0$

Image Processing - Local Image Operations

Calculation of the ranking operators: **Median operator**

The grey values of the pixels covered by the structural element (N8 neighbourhood) are sorted by size. The median operator replaces the grey value of the pixel to be transformed with $g'(i,j) = g_4$.

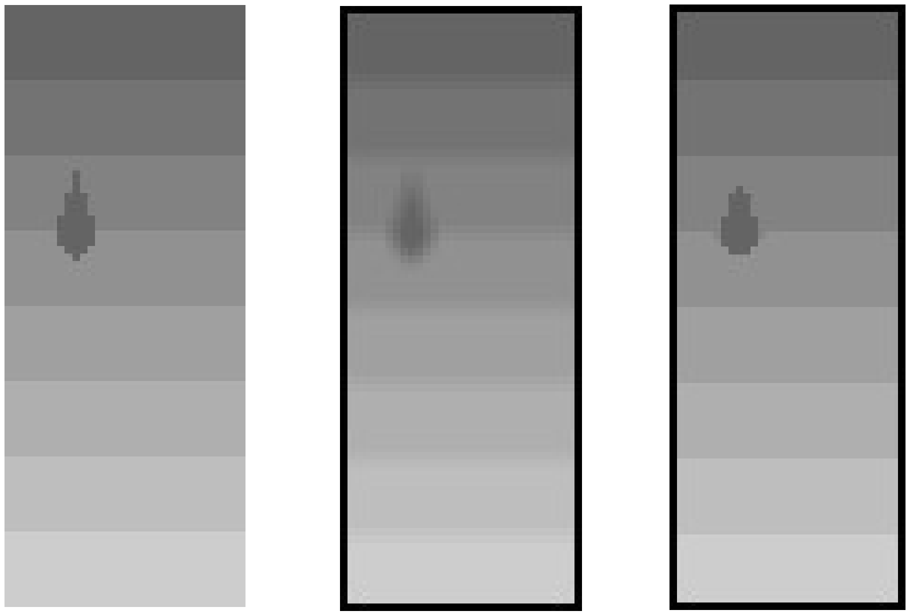


- Improves noisy images: Eliminates isolated, defective pixels
- Edges, however, are not washed out (compare mean value operator)

Image Processing - Local Image Operations

Comparison: **median operator and mean value**

- The median operator improves noisy images: Eliminates isolated, defective pixels
- Edges, however, are not washed out (compare mean value operator)



Original 3x3 Mean 3x3 Median

Image Processing - Local Image Operations

Calculation of the rank order operators: **Dilatation**

The grey values of the pixels covered by the structural element (N8 neighbourhood) are sorted by size. The dilation replaces the grey value of the pixel to be transformed with $g'(i,j) = g_8$, the largest grey value.

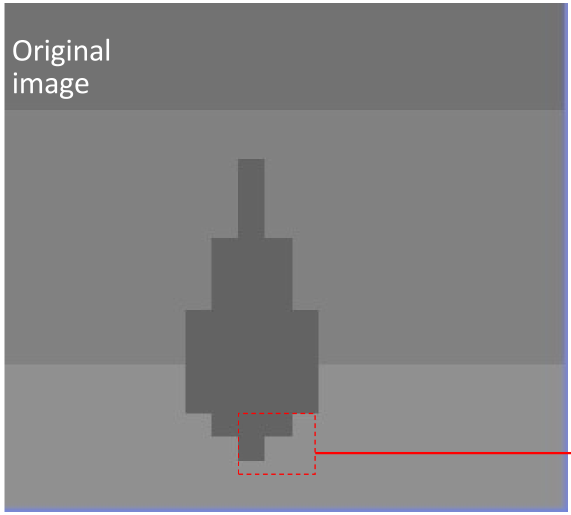


Image detail

99	99	99	99	99	99	144	144
99	99	99	99	99	99	144	144
99	99	99	99	99	99	144	144
99	99	99	144	144	144	144	144
99	99	99	144	144	144	144	144
99	99	99	144	144	144	144	144
144	144	144	144	144	144	144	144
144	144	144	144	144	144	144	144

Dilation

transformed extract

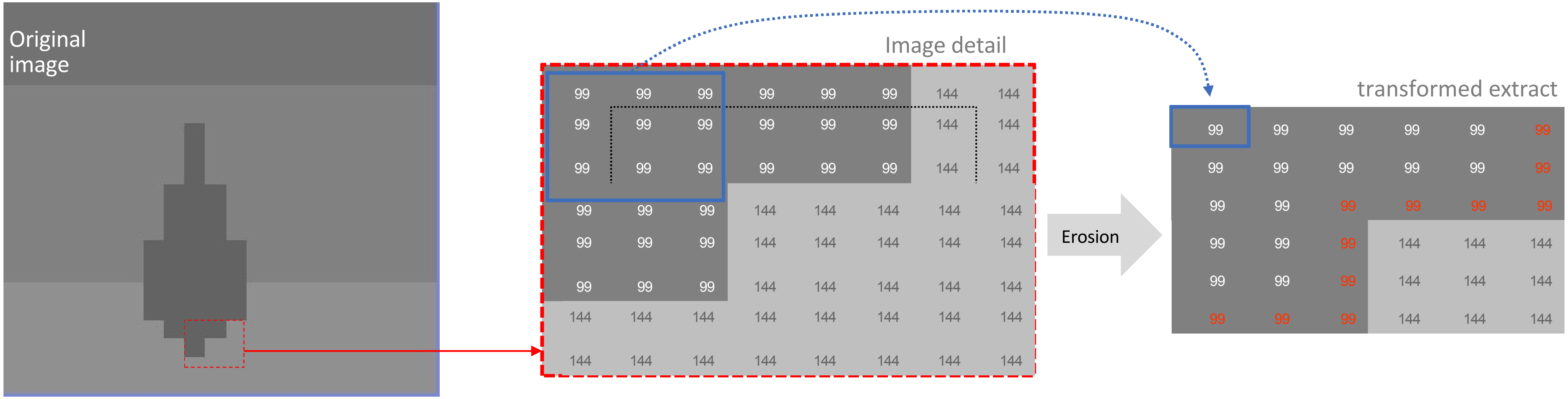
99	99	99	99	144	144
99	144	144	144	144	144
99	144	144	144	144	144
99	144	144	144	144	144
144	144	144	144	144	144
144	144	144	144	144	144

- The bright areas of the image expand.
- In general: $dil(x,y) = \max_{i,j}\{s_e(x+i, y+j) + k(i,j)\}$ with: i and j run over the scope of the Structural elements.

Image Processing - Local Image Operations

Calculation of the ranking operators: **Erosion**

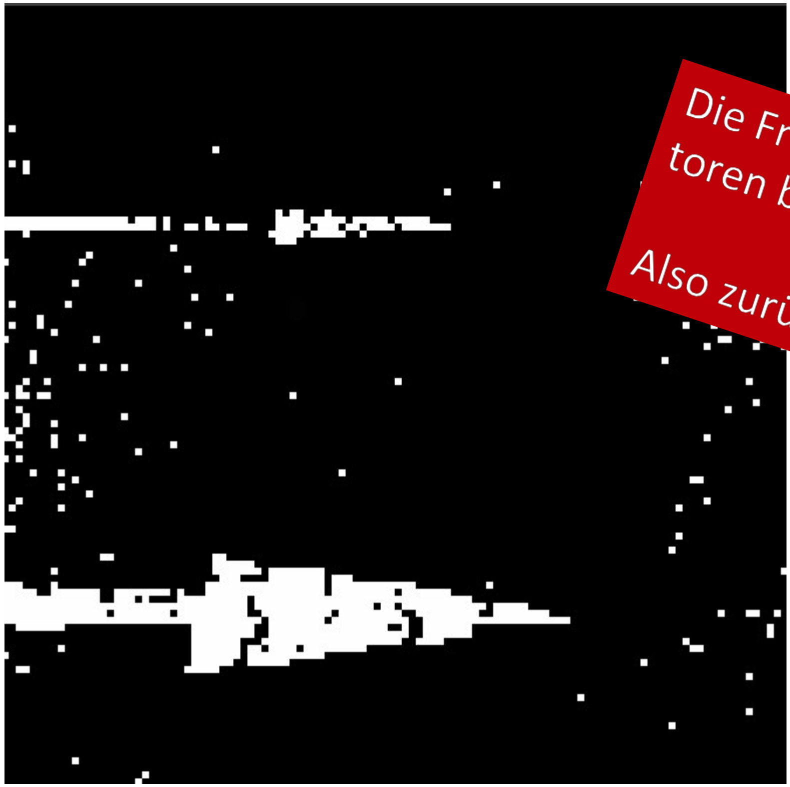
The grey values of the pixels covered by the structural element (N8 neighbourhood) are sorted by size. The erosion replaces the grey value of the pixel to be transformed with *$g'(i,j) = g_0$*, the smallest grey value.



- The "darker" areas of the picture expand.
- In general: $ero(x,y) = \min_{i,j}\{se(x+i, y+j) + k(i,j)\}$ with: i and j run across the scope of the Structural elements.

Image Processing - Local Image Operations

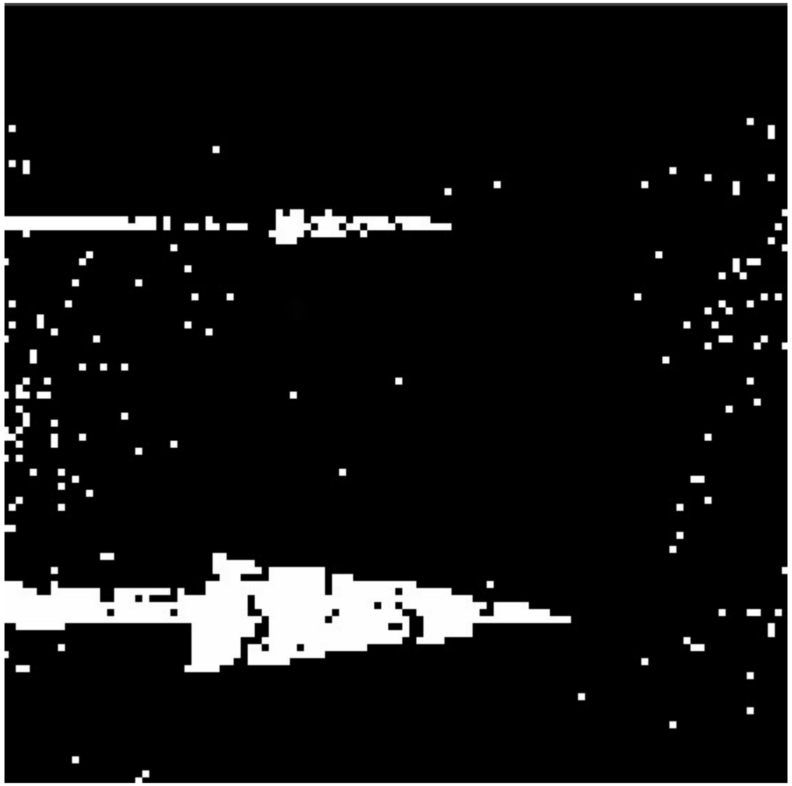
Segmentation: The task is to assign the pixels of the large arrow in the foreground to a segment. and all other pixels together as a background segment.



Die Frage: „Wie werden Rangfolgeopera-toren berechnet?“ ist nun beantwortet.
Also zurück zur eigentlichen Aufgabe.

Image Processing - Local Image Operations

Segmentation: The task is to assign the pixels of the large arrow in the foreground to a segment. and all other pixels together as a background segment.



The tasks can be divided into the following subtasks:

1. Eliminate noise
2. Close gaps within the arrow

Idea to eliminate the noise:

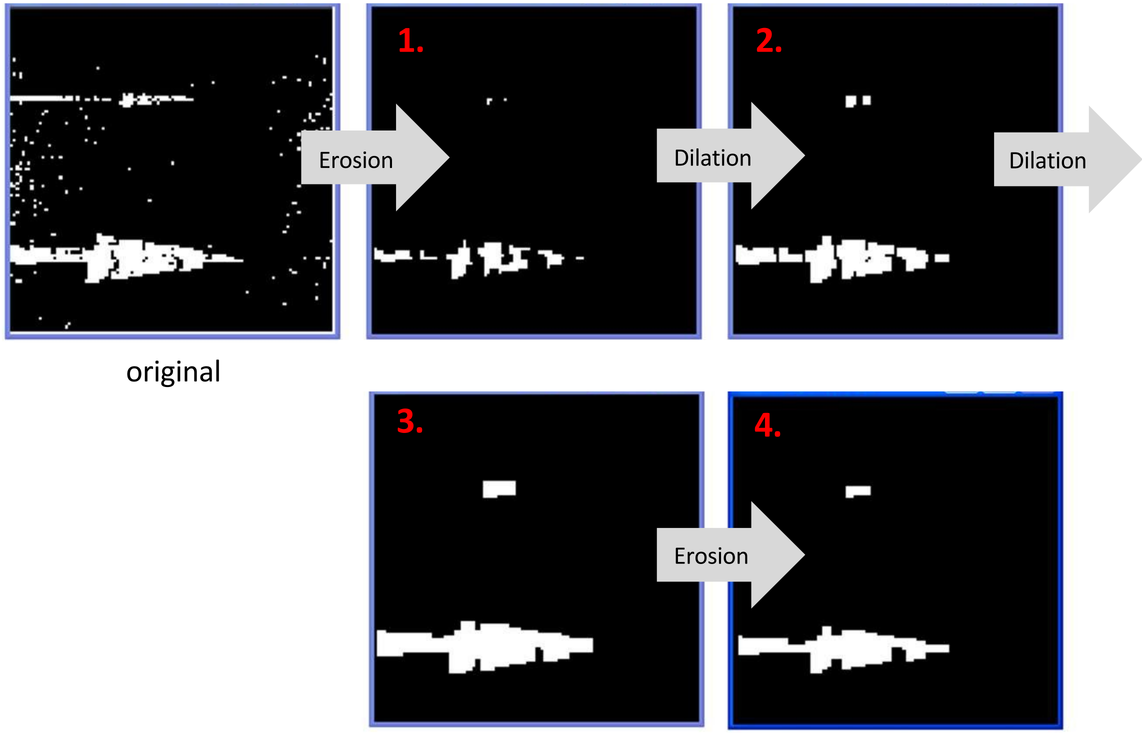
- 1 x erosion
- 1 x Dilatation

Idea for closing the arrow gaps:

- 1 x Dilatation
- 1 x erosion

Image Processing - Local Image Operations

Segmentation: The task is to assign the pixels of the large arrow in the image foreground to a segment and to combine all other pixels as a background segment.



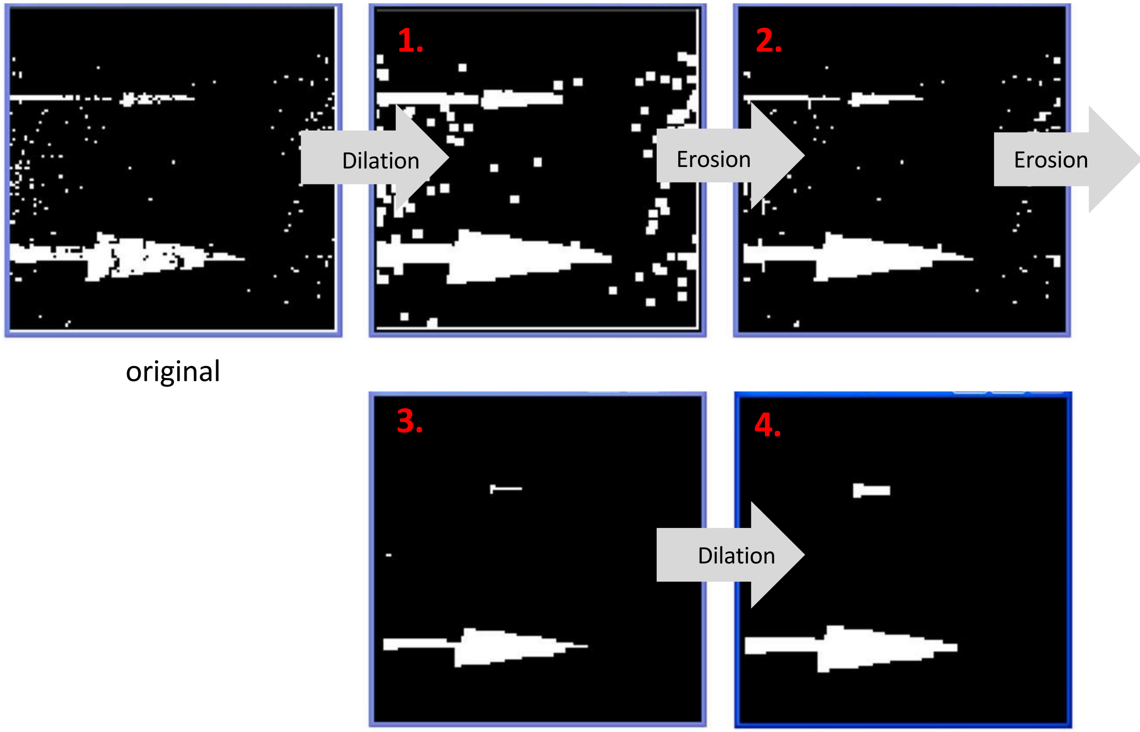
Variant A to solve the task:

1. Eliminate noise:
   - 1 x erosion
   - 1 x Dilatation
   - n· Erosion + n · Dilation = Opening

2. Close gaps within the arrow:
   - 1 x Dilatation
   - 1 x erosion
   - n· Dilation + n · Erosion = Closing

Image Processing - Local Image Operations

Segmentation: The task is to assign the pixels of the large arrow in the image foreground to a segment and to combine all other pixels as a background segment.



Variant B to solve the task:

1. Close gaps within the arrow:
- 1 x Dilatation
- 1 x erosion
- n· Dilation + n · Erosion = Closing

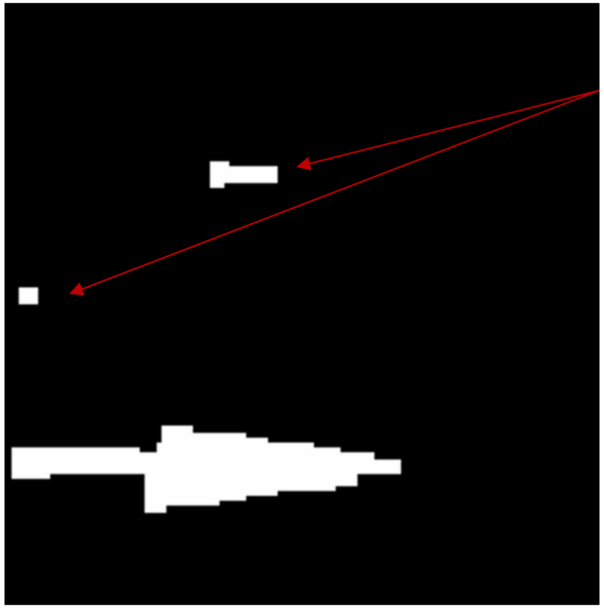
2. Eliminate noise:
- 1 x erosion
- 1 x Dilatation
- n· Erosion + n · Dilation = Opening

This variant is clearly the better one!

How do you get rid of the pixel group at the top?

Image Processing - Local Image Operations

Segmentation: The task is to assign the pixels of the large arrow in the image foreground to a segment and to combine all other pixels as a background segment.



Result after application of the rank order operators

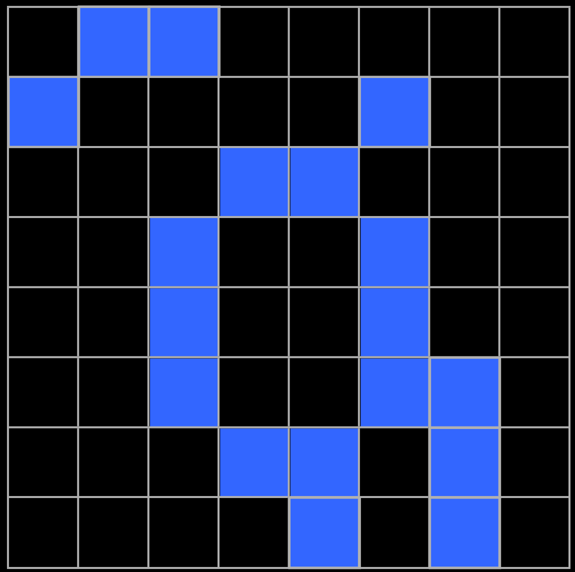
Question: How do you get rid of these two clusters of pixels?

Answer: You segment the image into one background object and three different foreground objects. Only the largest foreground object is kept. The pixels of all the other foreground objects flow into the background, i.e. they become black.

A greyscale image is segmented by forming correlation components.

Image Processing - Local Image Operations

Segmentation: What are contextual components (CC)?



The foreground (blue) contains:

7 N4 ZHK

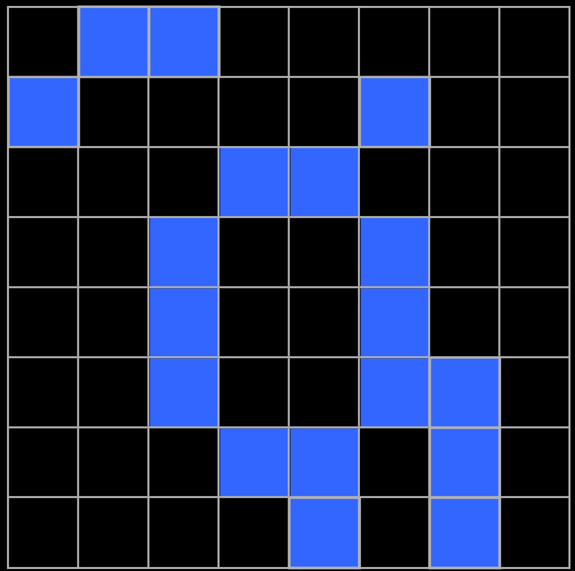
2 N8 ZHK

Foreground and background must always be determined with opposing neighbourhood ratios.

If the foreground objects are formed by N4 neighbourhoods, the corners of the pixels are not connected and the background can flow through. This means that the background pixels are connected via an N8 neighbourhood.

Image Processing - Local Image Operations

Segmentation: What are contextual components (CC)?



The background (black) contains:

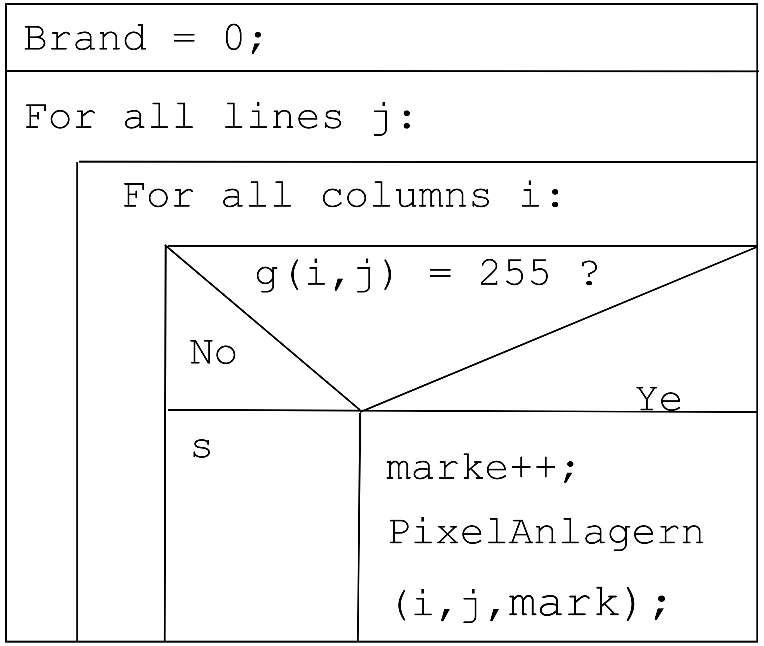
4 N4 ZHK

1 N8 ZHK

Image Processing - Local Image Operations

Segmentation: Recursive Flooding (Flood Fill Algorithm) - a way to build a coherence component

Recursive flooding()



PixelAnlagern(i,j, mark)

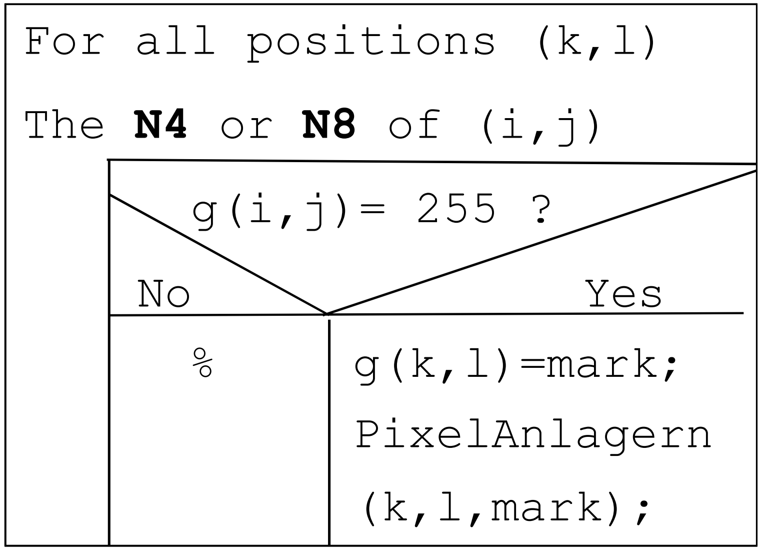


Image Processing - Local Image Operations

Segmentation: Recursive Flooding (Flood Fill Algorithm) - a way to build a coherence component

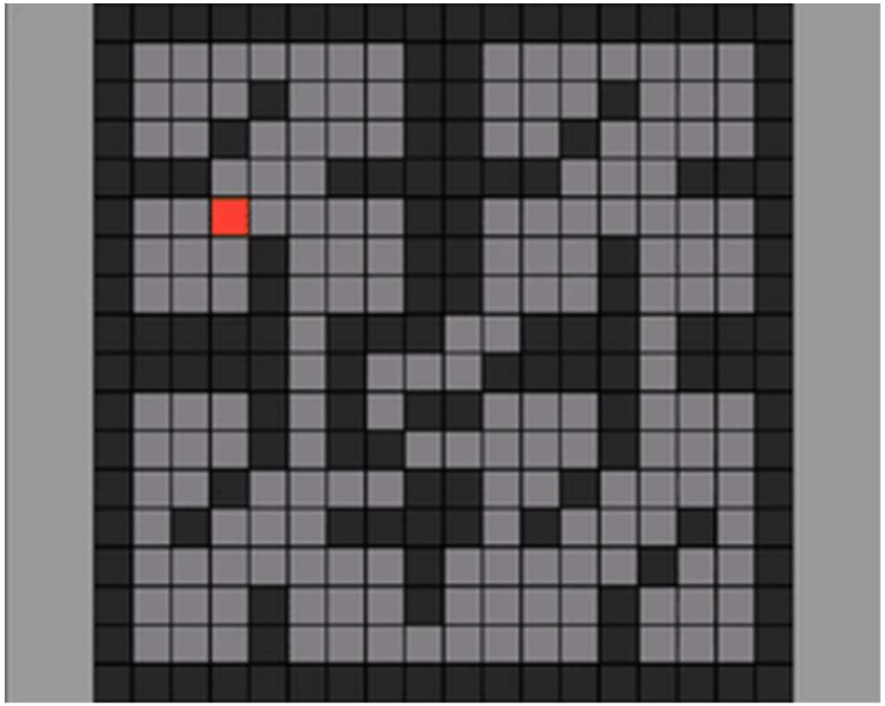
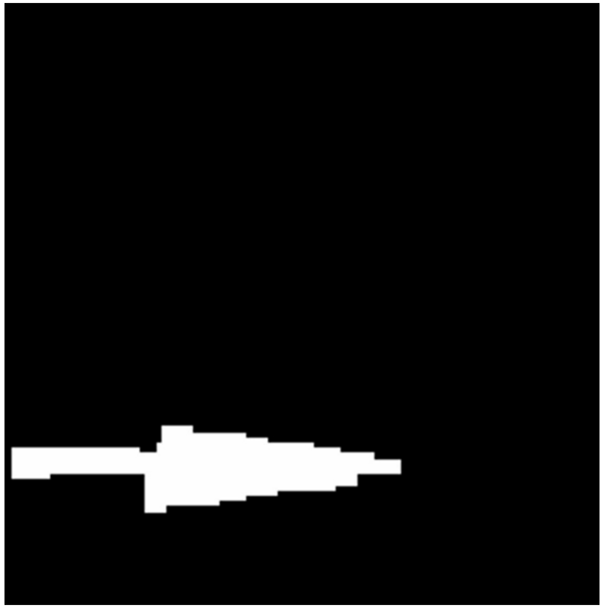


Image from: https://vidhu-verma.medium.com/everything-about-flood-fill-algorithm-graphs-28c133e516cf

Image Processing - Local Image Operations

Segmentation: The task is to assign the pixels of the large arrow in the image foreground to a segment and to combine all other pixels as a background segment.



Result after application of the rank order operators

The greyscale image is segmented by the formation of context components:

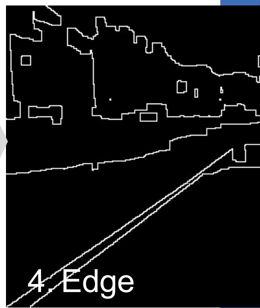
- There is one background segment and three foreground objects.
- The two small foreground objects are deleted.
- The largest foreground object remains. The pixels of this segment belong to the large arrow that was searched for. The task is solved

Image Processing - Local Image Operations

Summary: Sample image pre-processing

- happens before the content analysis of an image,
- i.e. before image contents are classified as lines, segments, etc.
- Segmentation and the Hough transformation are two examples for the interpretation of image content (see Step 4 and Step 5, respectively: This is where the content-related image analysis begins).

5. hough trans-formation for the detection of straight lines



Preprocessing takes place either on the brightness channel or on all colour channels.

1. Blur the image to make grey areas more homogeneous. Caution: Do not blur the edges too much! A selective blur was used here to preserve the edges.

2. Determine threshold value to biniarise the image

3. Use ranking operators to prepare the image for channel extraction or segmentation.

4. segmentation procedure

Image template from https://davidgruenewald.de/category/darmstadt/

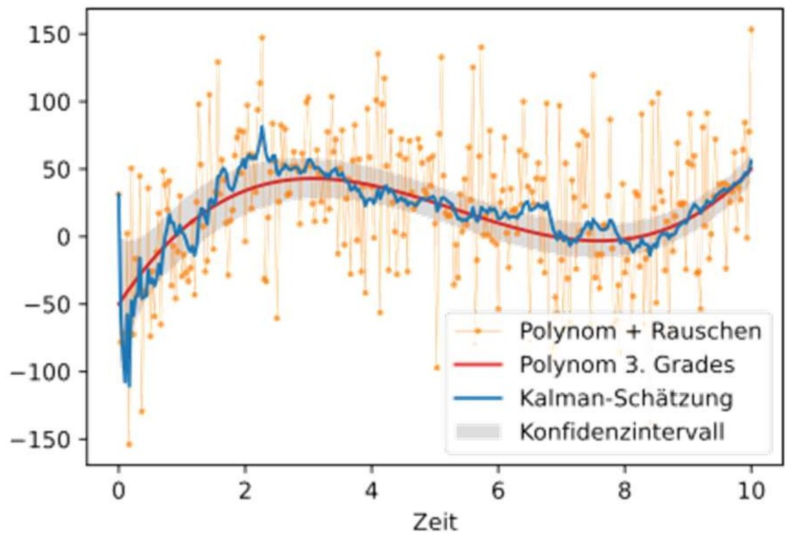
Image Processing - Local Image Operations

Little extra: How can you merge the pixels that mark the curved road into a line?

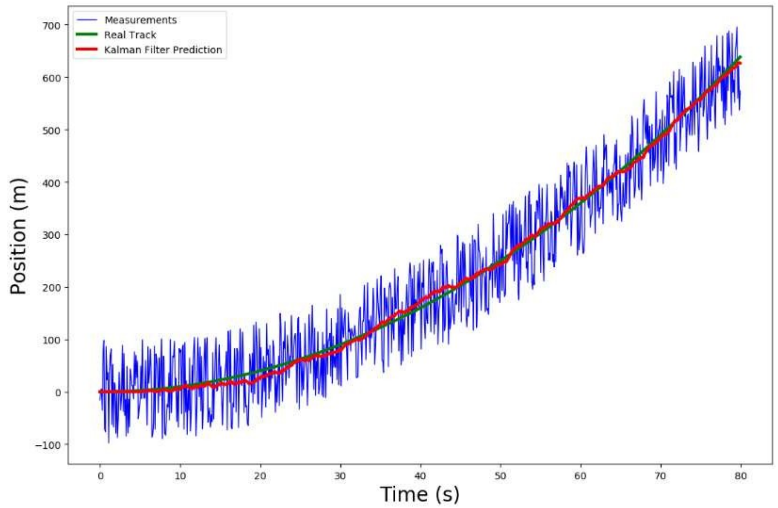
Image Processing - Local Image Operations

Little extra: How can you merge the pixels that mark the curved road into a line?

One possibility is to use the Kalman filter.
The Kalman filter is a mathematical method for the iterative estimation of parameters on the basis of observations containing errors.



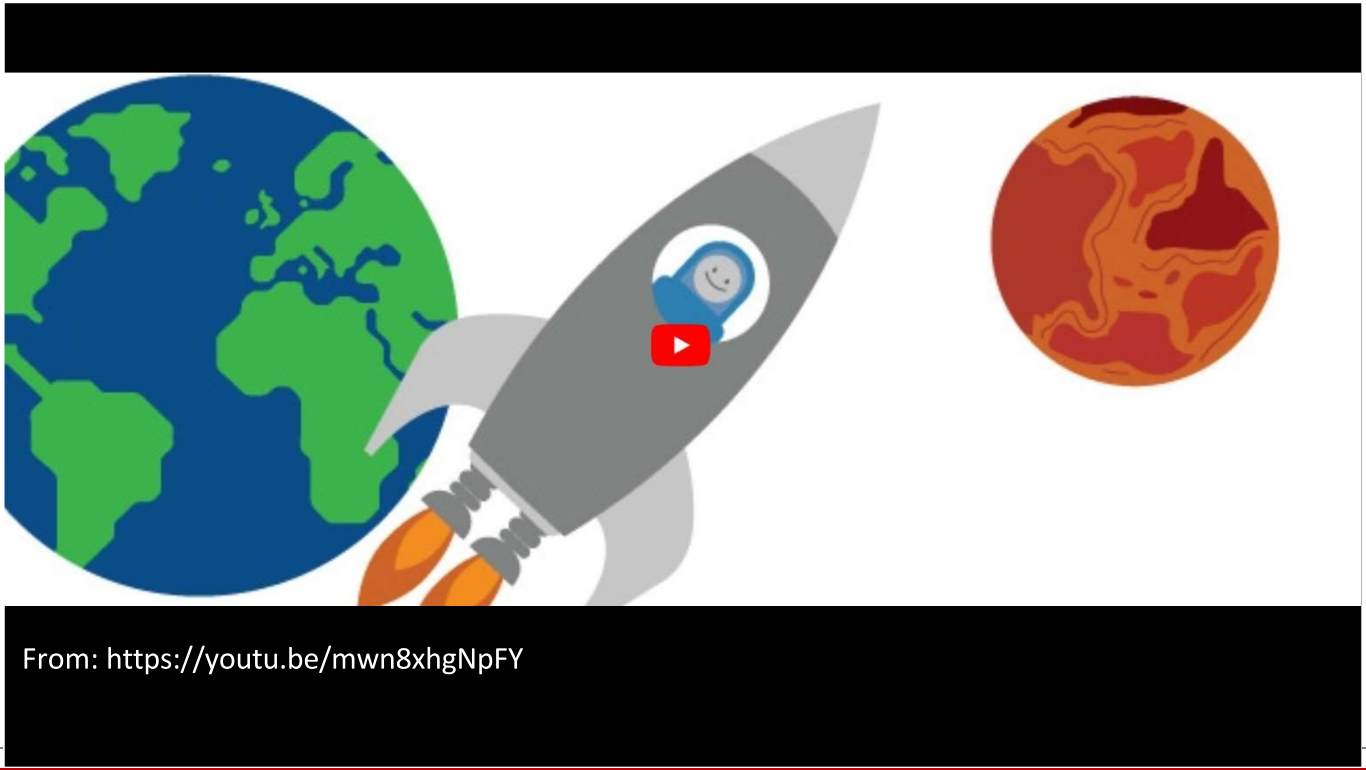
Example from: https://de.wikipedia.org/wiki/Kalman-Filter#/media/File:Kalman_Polynom_Test.svg



Example from: https://machinelearningspace.com/object-tracking-python/

Image Processing - Local Image Operations

A little extra: The Kalman filter is a mathematical method for the iterative estimation of parameters based on observations with errors.



Please click on the black border to start the video.

Image Processing - Local Image Operations

Point operators compared to local image operators Local image operators

- Blur: Mean operator, Gaussian filter
- Edge detectors: difference filter and Sobel operator, Laplace operator
- Contrast enhancement filter
- Ranking operators: Erosion, Dilatation, Median as well as Opening and Closing
- Segmentation procedure

Image Processing - Local Image Operations

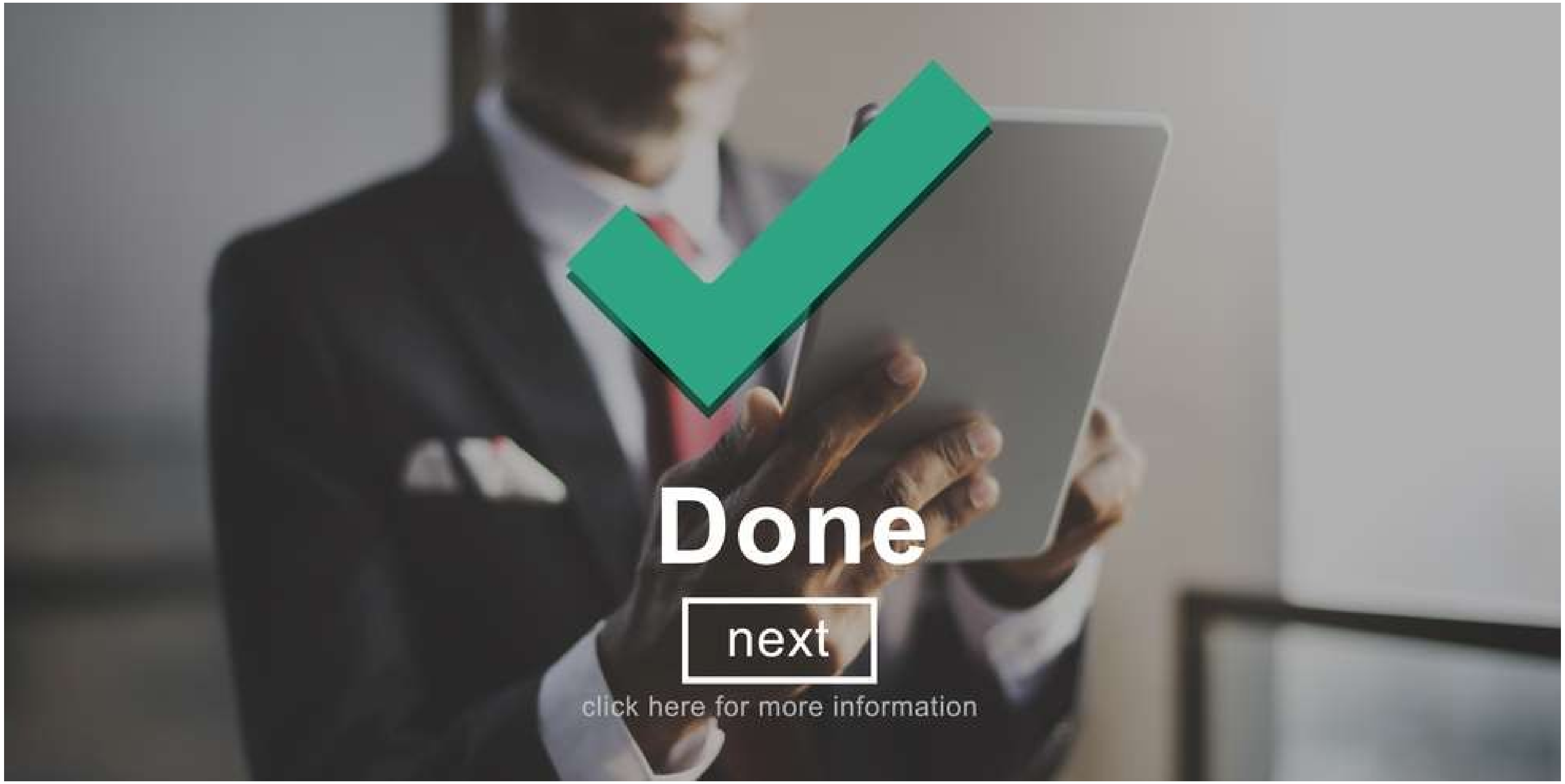


Image taken from: https://social-systems.de/blog/denkanstoesse/die-strategie-steht/