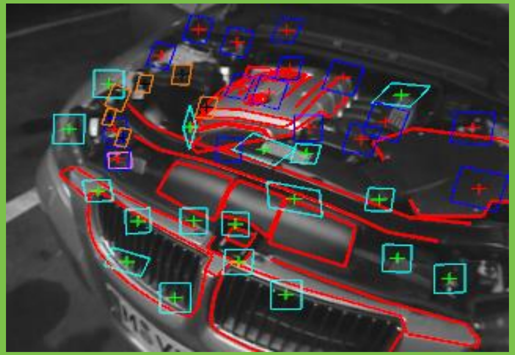
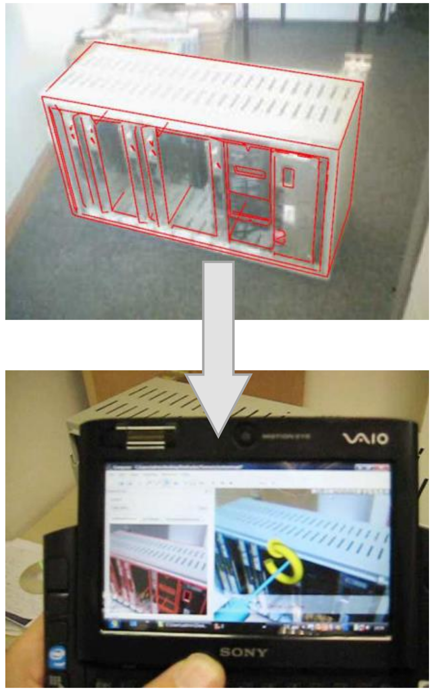
Visual Computing – Bildverarbeitung



E. Hergenröther, Y. Jung, B. Meyer

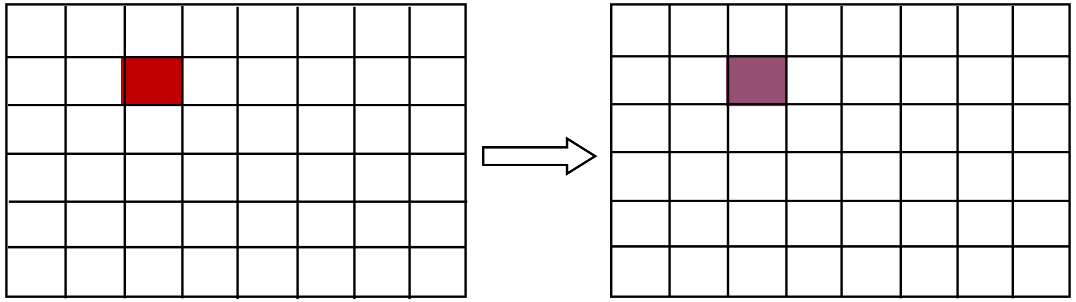
Übersicht

- Punktoperatoren im Vergleich zu lokalen Bildoperatoren
- Lokale Bildoperatoren (betrachten Nachbarschaften)
  - Weichzeichner: Mittelwertoperator, Gauß-Filter
  - Kantendetektoren: Differenzfilter und Sobel-Operator, Laplace-Operator
    - Filter zur Kontrastverbesserung
    - Rangfolgeoperatoren: Erosion, Dilation, Median sowie Opening und Closing
    - Segmentierungsverfahren



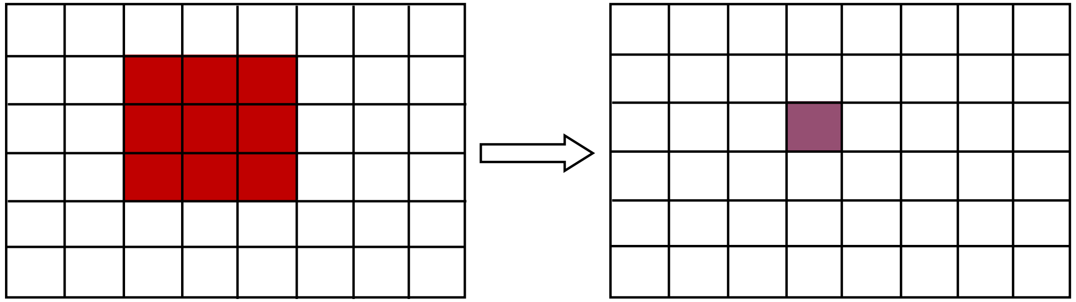
Punktoperatoren vs. lokale Bildoperatoren

Punktoperator: Pixel werden einzeln transformiert, ohne das Nachbarpixel in Transformation einfließen



Beispiele:
- Helligkeitsänderungen,
- Kontraständerungen,
- Gamma-Korrektur,
- Farbtransformation, …

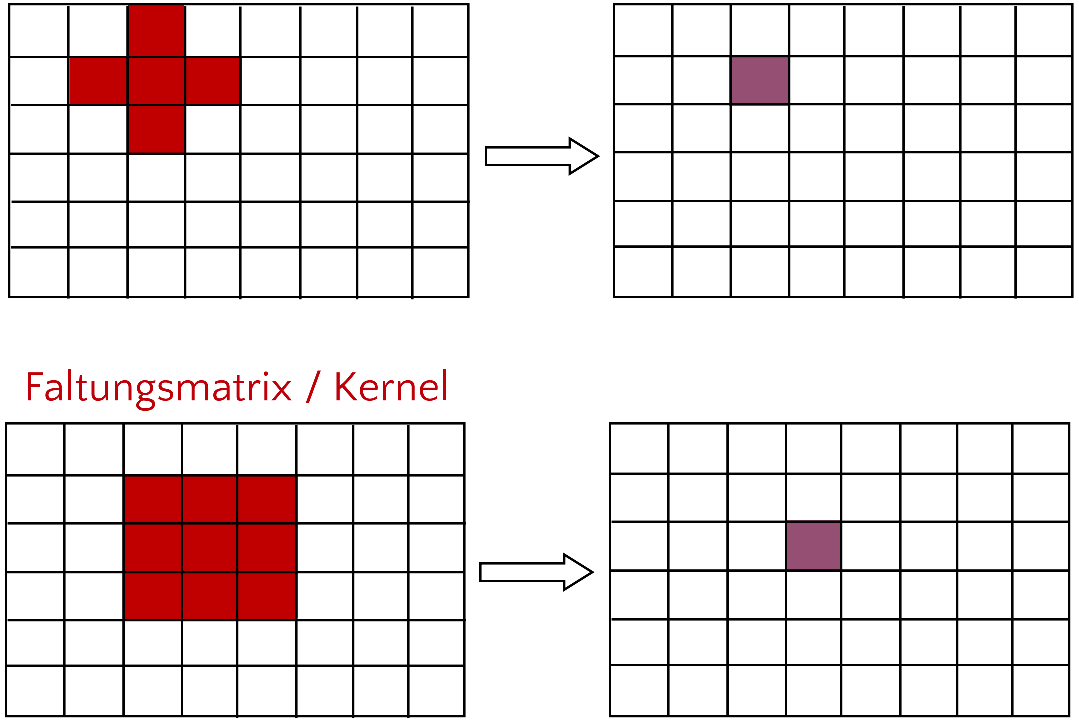
Lokale Bildoperatoren: Jeder Pixel wird in Abhängigkeit zu seinen benachbarten Pixel transformiert



Beispiele:
- Bilder weichzeichnen / verschmieren,
- Kanten detektieren, …

Lokale Bildoperationen

Lokale Bildoperatoren: Lassen Nachbarpixel durch Verwendung unterschiedlicher Kernels in Berechnung einfließen



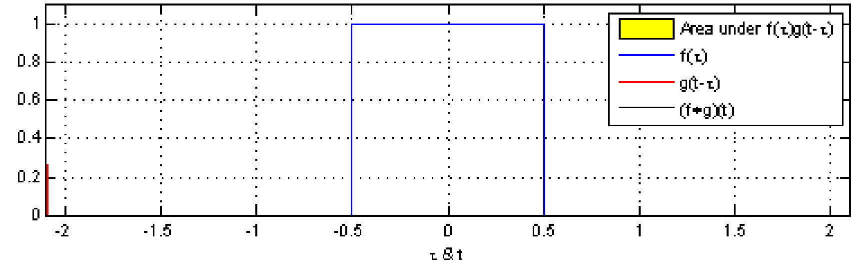
4er-Nachbarschaft (N4)

8er-Nachbarschaft (N8)

Faltung (Convolution)

HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

- Unter der Faltung zweier Funktionen *f(t), g(t)* versteht man folgendes Integral (mit Faltungsoperator „*“):
  - $y(t) = f(t) * g(t) = \int_{-\infty}^{\infty} f(\tau) g(t - \tau) d\tau$
  - Es gelten u.a. Kommutativ- und Assoziativgesetz
  - Bei diskreten Funktionen mit endlichem Definitionsbereich kann man die Faltung durch Multiplikation mit einer Matrix ausdrücken
    - Eine Funktion *f* kann man glätten, indem man sie mit einem sog. Glättungskern *g* faltet



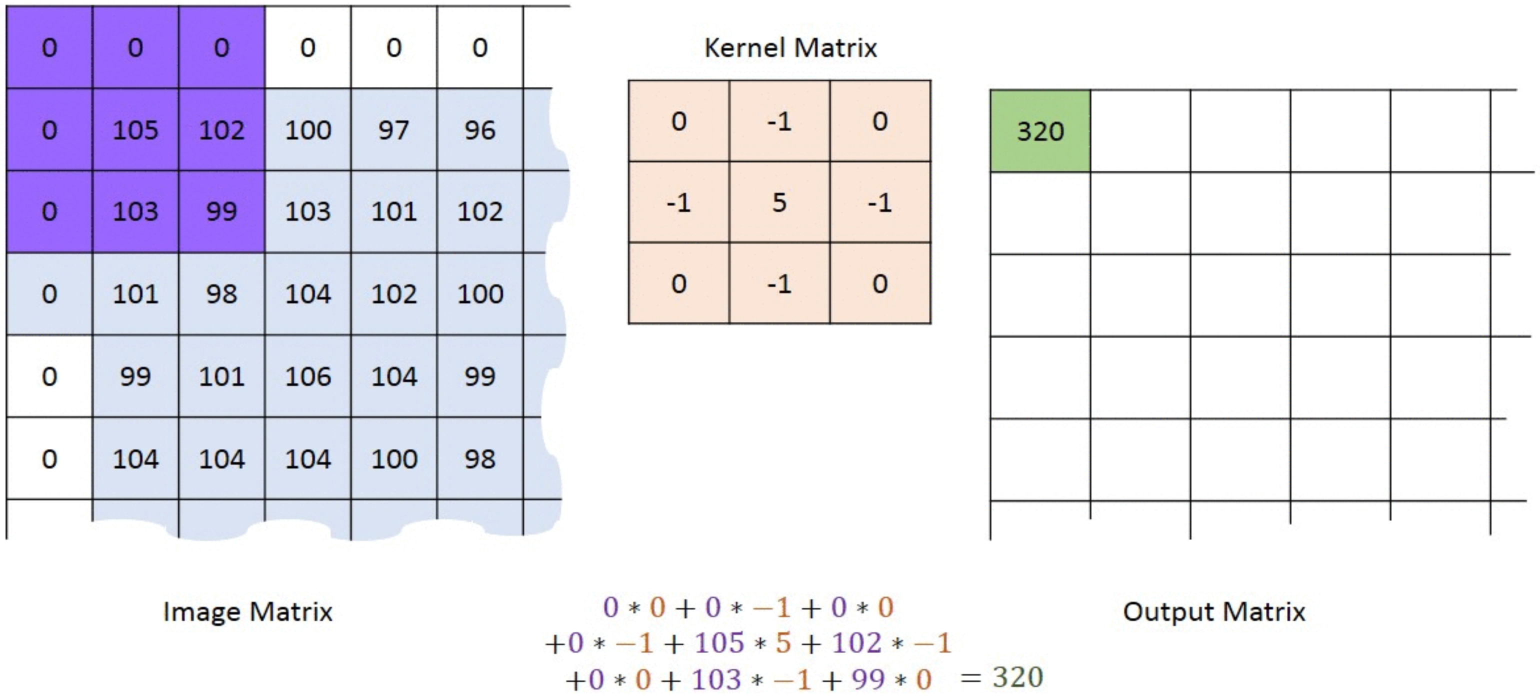
https://commons.wikimedia.org/w/index.php?curid=11003835

- Eng damit verwandt ist die Kreuzkorrelationsfunktion, die die Übereinstimmung zweier Funktionen in Abhängigkeit ihrer Verschiebung um *t* angibt
  - $\mathrm{r}(t) = \int_{-\infty}^{\infty} f(\tau) g(t + \tau) d\tau = f(-t) * g(t)$
  - Im Unterschied zur Faltung wird Funktion *g* hier nicht gespiegelt entlang von *f* verschoben

Diskrete Faltung am Beispiel

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

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

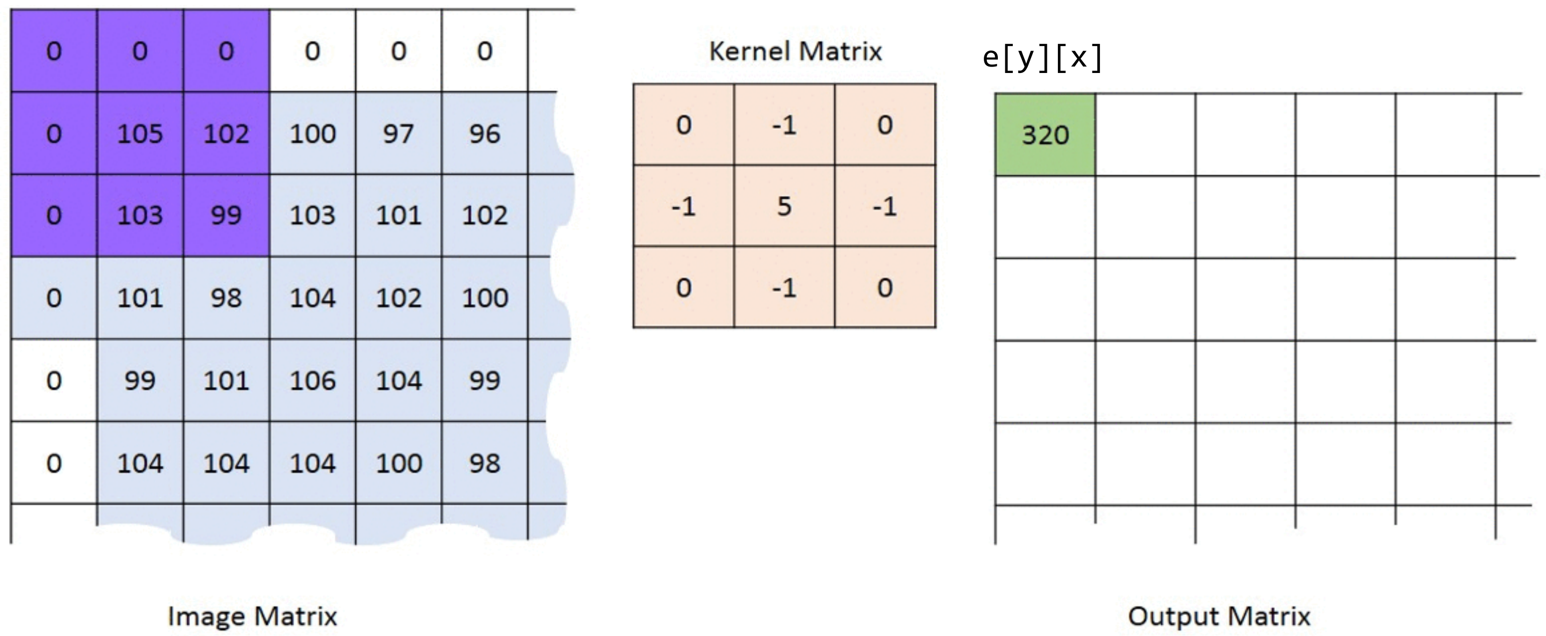


Array-/ Bildgrenzen beachten: meist wird mit 0, 255 oder dem Wert des Randpixels „aufgefüllt“

Faltung mit einem lokalen Bildoperator, der die sog. N8-Nachbarschaft nutzt

Diskrete Faltung am Beispiel

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



Kernel ***g*** *oft als eindimensionales Array gegeben – Zugriff auf g(l, k):*

```
g[l * 3 + k];
```

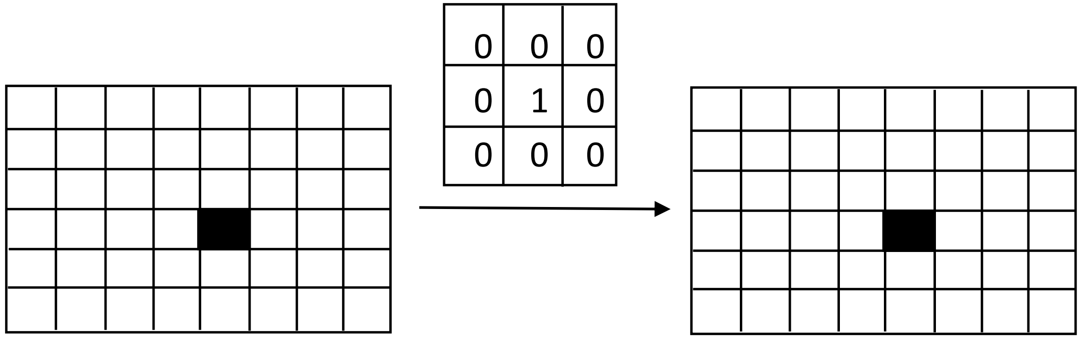
Allgemeine Formulierung der Faltungsoperation: $$e(y,x) = \sum_{l=0}^{2}\sum_{k=0}^{2}\{f(y+(l-1),x+(k-1))\cdot g(l,k)\}$$

Faltung: Identitätsoperator

Eingabebild und Faltungsergebnis sind identisch

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

(Filter-)Kernel / Faltungsmatrix



Eingabebild

Faltungsergebnis

Faltung: Mittelwertoperator

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

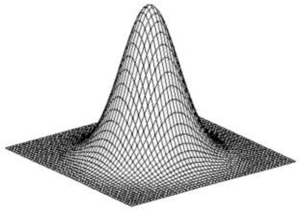
- Bildet Mittelwert bzw. Durchschnittswert aus benachbarten Pixelwerten

$F_{Mittelwert}$ =

1	1	1
1	1	1
1	1	1

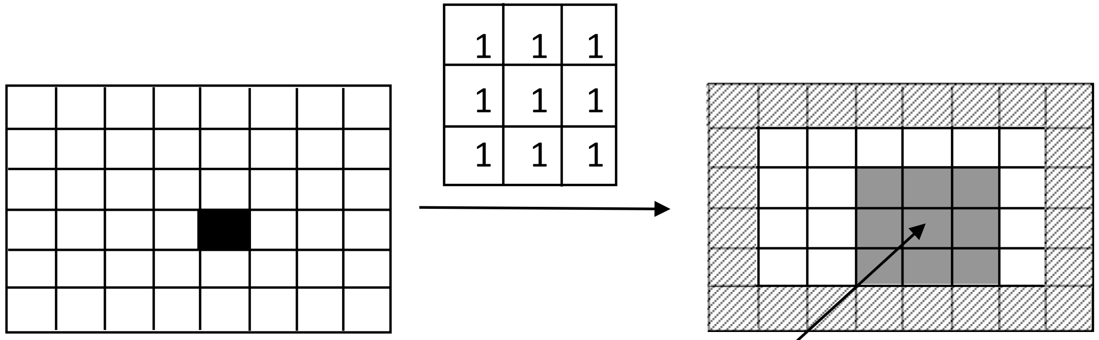
(Filter-)Kernel / Faltungsmatrix

- Unterschiedliche Weich-zeichner möglich
  - Gauß-Filter:



$F_{Gauß}$ =

1	2	1
2	4	2
1	2	1

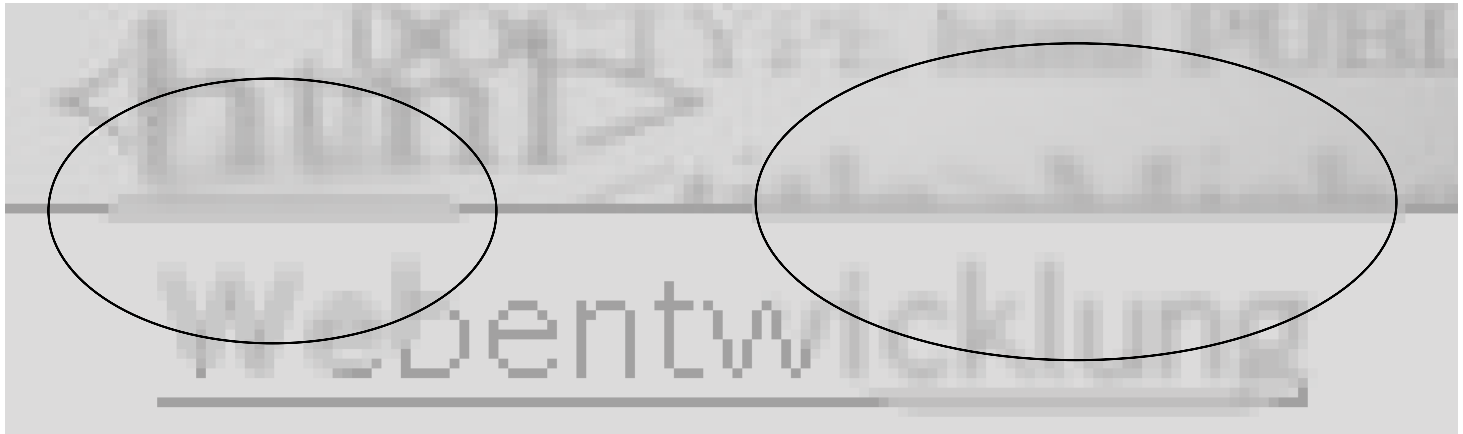


Eingabebild ist
8*6 Pixel groß

g‘(i,j)= 255*8 / 9. = 226

Zurück in Wertebereich abbilden

Mittelwertoperator



Gauß-Filter

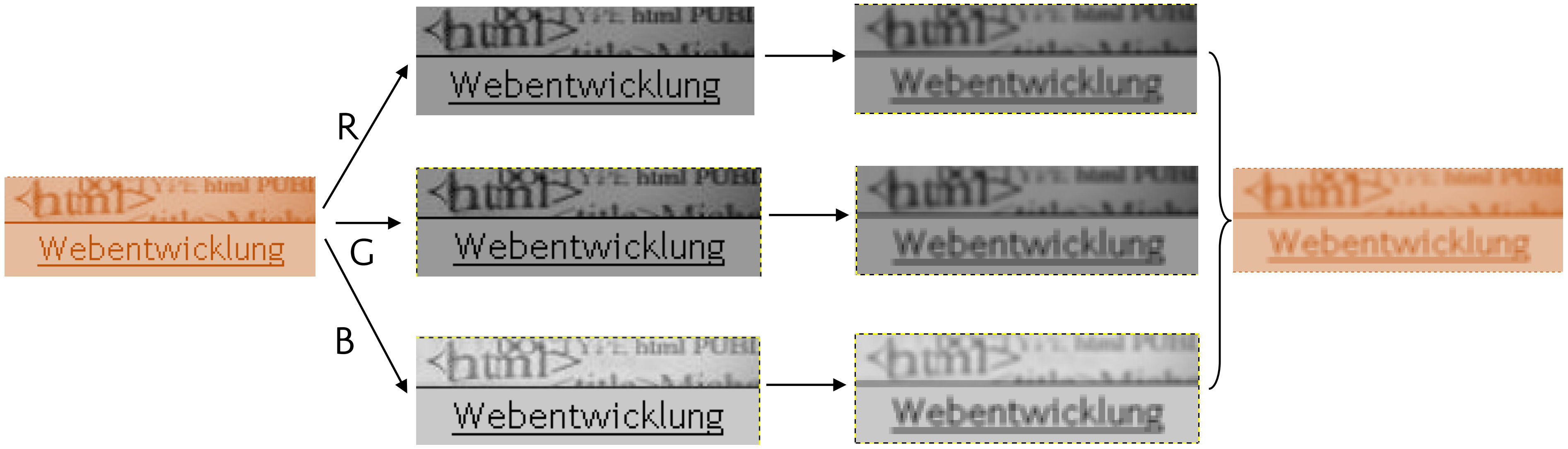
- Durch die Berechnung mit dem Mittelwertoperator bzw. dem Gauß-Filter ergeben sich Faltungsergebnisse **$e(i, j)$**, die außerhalb des Grauwertbereich $\{0, 1, \ldots, 255\}$ liegen
- Um die Faltungsergebnisse zur Anzeige wieder in den Grauwertbereich zu transformieren, muss eine lineare Abbildung auf den Wertebereich $\{0, 1, \ldots, 255\}$ durchgeführt werden:
  - Transformation nach Faltung mit dem Mittelwertoperator: $g'(i, j) = 1/9 \cdot e(i, j) + 0$
  - Transformation nach Faltung mit dem Gauß-Filter: $g'(i, j) = 1/16 \cdot e(i, j) + 0$

Farb-/RGB-Bilder falten

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

1. Trennen der unterschiedlichen RGB-Farbkanäle
2. Für jeden Farbkanal: entsprechendes Grauwertbild filtern
3. Gefilterte Bilder der Farbkanäle zum Farbbild kombinieren

Je nach Anwendungszweck (z.B. für Kantenerkennung) Bild ggfs. erst in Graustufen umwandeln und dann Graustufenbild falten



Unterschiedliche Filtergrößen

- Verwendete Kernelgröße ist abhängig von Auflösung des Eingabebilds und gewünschtem Effekt

Bsp. Gauß-Filter: $\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$ $\frac{1}{256}\begin{bmatrix}1 & 4 & 6 & 4 & 1\\ 4 & 16 & 24 & 16 & 4\\ 6 & 24 & 36 & 24 & 6\\ 4 & 16 & 24 & 16 & 4\\ 1 & 4 & 6 & 4 & 1\end{bmatrix}$ $\cdots$

Aus: „Multimediale Signalverarbeitung Bildverarbeitung: Filter“, Thorsten Thormählen
https://www.mathematik.uni-marburg.de/~thormae/lectures/mmk/mmk_6_1_ger_web.html#1

Beispiel für weitere Filter

- Welchen Effekt haben die folgenden Filterkerne?

Mittelwertfilter in y-Richtung

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

Mittelwertfilter in x-Richtung

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

Aus: „Multimediale Signalverarbeitung Bildverarbeitung: Filter“, Thorsten Thormählen
https://www.mathematik.uni-marburg.de/~thormae/lectures/mmk/mmk_6_1_ger_web.html#1

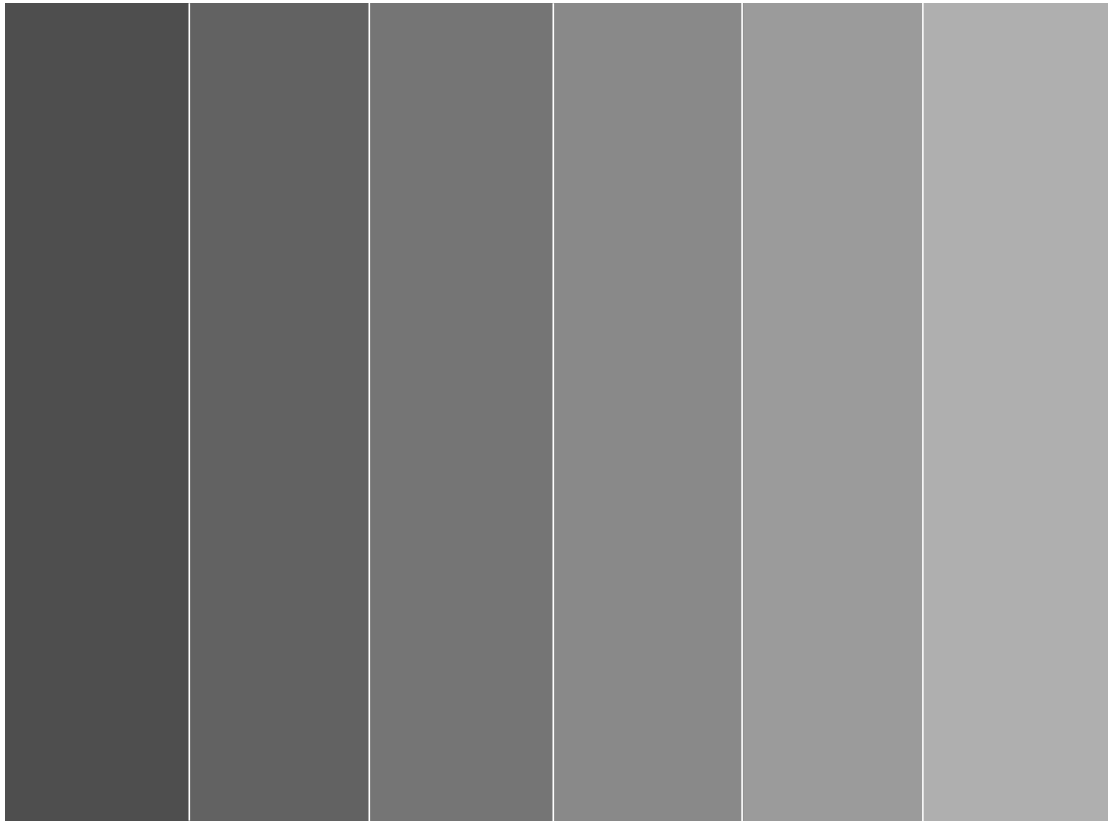
Kantenerkennung

Weitere Nachbarschaftsoperationen

Mach-Band-Effekt

- Menschliches Wahrnehmungssystem ist auf „Kantenerkennung“ optimiert

Die gleichbleibende Reize der Flächen werden gedämpft und Kontraste werden überzeichnet



Effekt wurde 1865 von Ernst Mach entdeckt

Bild aus:
https://upload.wikimedia.org/wikipedia/commons/9/97/Bandes_de_mach.PNG

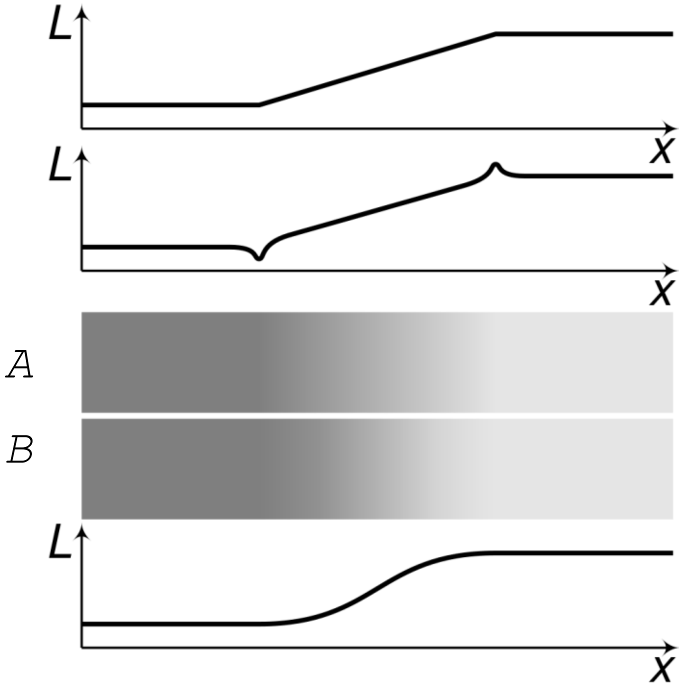
Mach-Band-Effekt

Effekt kann z.B. zu Fehlinterpretationen bei radiologischer Befundung führen: Hell-Dunkel-Kontraste werden verstärkt und bspw. als Karies interpretiert

Hier zeichnet sich ein dunkler Fleck ab, der ggfs. als Karies interpretiert werden könnte

Bild aus: Radiografische Projektionen der Objekte, Folie 11
https://slideplayer.org/slide/1330107/

Mach-Band-Effekt

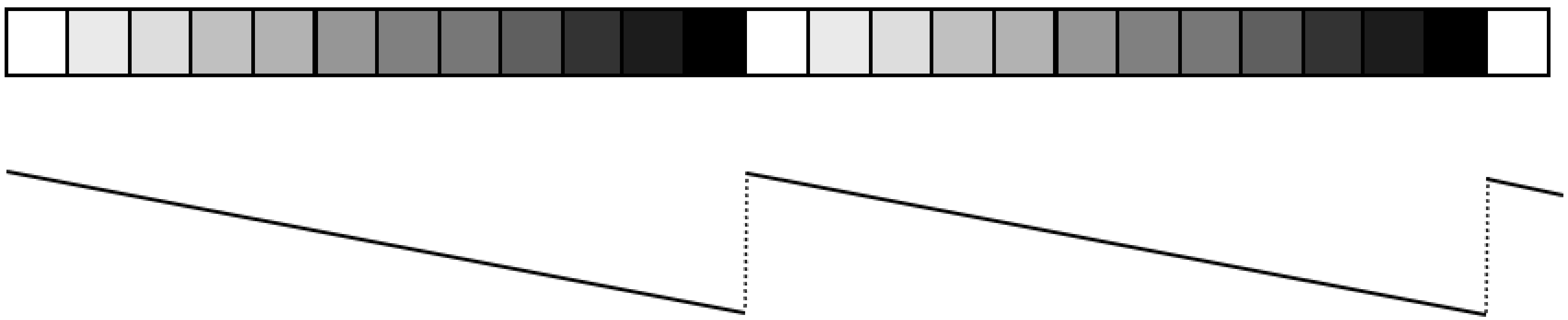


https://en.wikipedia.org/wiki/Mach_bands

- Helligkeitsübergänge werden mit höheren Kontrast wahrgenommen, als tatsächlich vorhanden ist
  - Abbildung beschreibt Effekt als Funktion der Leuchtdichte L
  1. Leuchtdichtenprofil sich abrupt ändernder Grauwerte (*A*)
  2. Von *A* wahrgenommenes Helligkeitsprofil
  3. Zum Vergleich Leuchtdichtenprofil sich kontinuierlich ändernder Grauwerte (*B*)
- Ähnlich, wie wir es von unserer Wahrnehmung gewohnt sind, arbeitet ein Faltungsoperator zur Kontrastverstärkung

Herleitung Kantenerkennung

- Differenzoperator erkennt Kanten
- 1. Schritt zur Herleitung:
  - Statt eines ganzen Bildes wird nur eine Bildzeile betrachtet



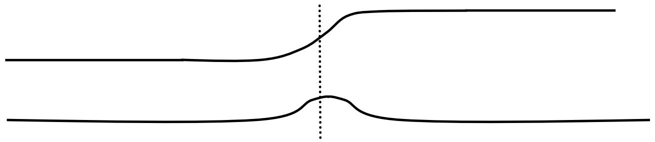
Grauwertprofil der Bildzeile

Funktion s(x), die den Grauwertverlauf entlang dieser Bildzeile darstellt

Herleitung Kantenerkennung

- 2. Schritt zur Herleitung:
  - Ableiten einer Funktion *g(x)*, um Kante zu finden
  - Die 1. Ableitung (Steigung) ist definiert durch:
    - $g'(x) = \lim_{\Delta x \to 0} \frac{g(x+\Delta x) - g(x)}{\Delta x}$
      - Mit g(x) = Grauwert des Pixel an Position x
    - Übertragen auf Bilder, die durch Pixel-Darstellung diskreten Raum bilden, heißt das, der Abstand Δx zwischen zwei Pixeln wird nicht unendlich klein, sondern ist mindestens 1
  - 1. Ableitung entspricht also der Differenz zweier benachbarter Grauwerte:
    - $\frac{g(x+1) - g(x)}{1} = g(x+1) - g(x)$
    - In 1. Ableitung zeigen Werte ungleich 0 die Positionen von Kanten in Bildzeile an

Grauwertverlauf einer Bildzeile s(x)



Berechnen der 1. Ableitung s‘(x)

Gradienten

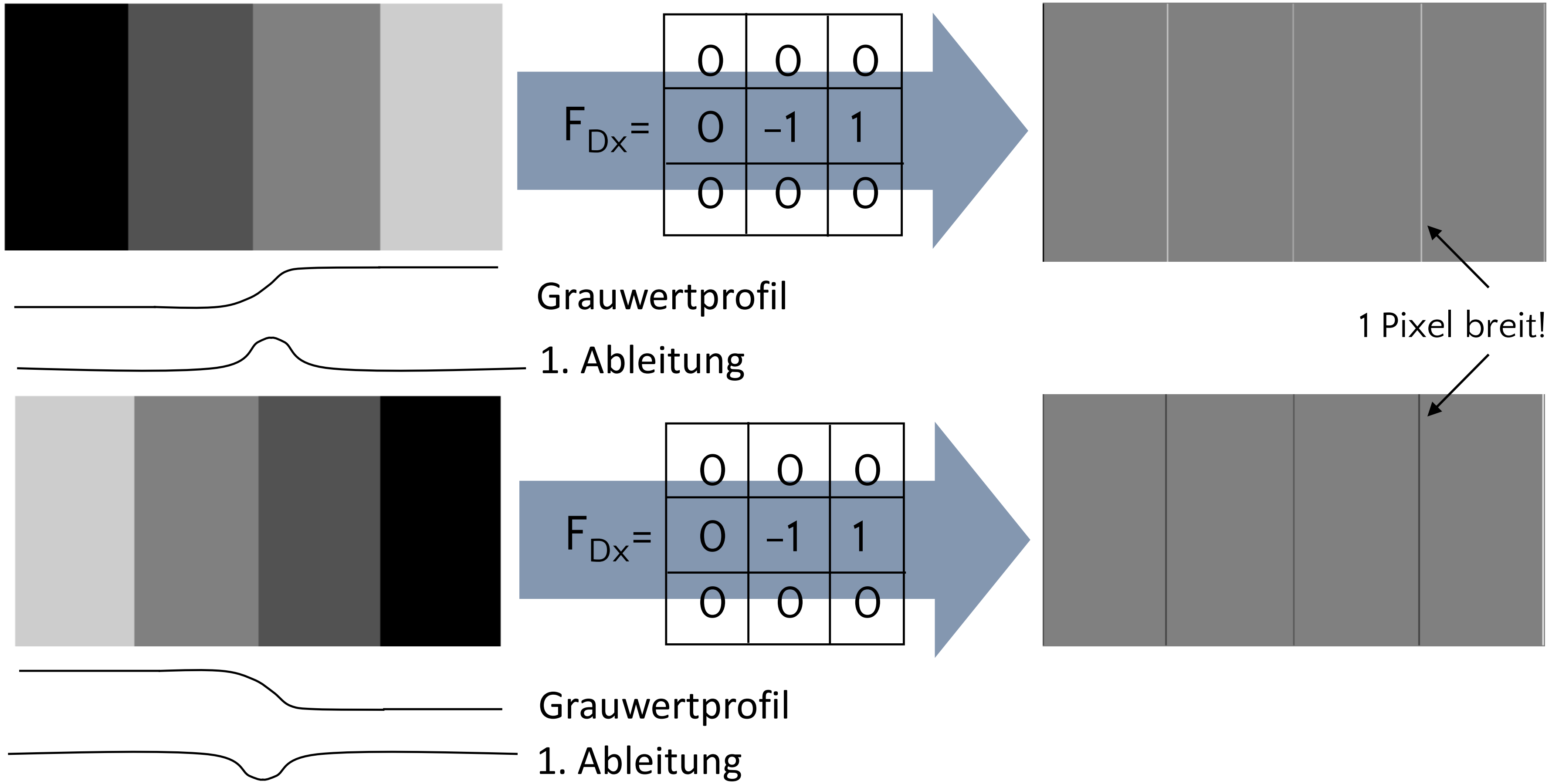
- Betrachte Bild als (nach *x* und *y*) differenzierbare Funktion *f*(*x*, *y*)
  - Der Wert von *f* an der Position (*x*, *y*) ist damit die Farbe oder Graustufe des Bildes
  - Da Bild jedoch diskrete Funktion von benachbarten Pixeln ist, müssen partielle Ableitungen (in horizontale und vertikale Richtung) approximiert werden
- Gradientenvektor ∇f an jedem Pixel bestimmen (i.d.R. durch Faltung des Bildes)
  - $\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix} = \begin{pmatrix} \lim\limits_{\Delta x \to 0} \frac{f(x+\Delta x, y) - f(x, y)}{\Delta x} \\ \lim\limits_{\Delta y \to 0} \frac{f(x, y+\Delta y) - f(x, y)}{\Delta y} \end{pmatrix} \underset{\substack{\Delta x = 1 \\ \Delta y = 1}}{\Longrightarrow} \begin{pmatrix} f(x+1, \ y) - f(x, \ y) \\ f(x, \ y+1) - f(x, \ y) \end{pmatrix}$
    - ∇f zeigt als Vektor in Richtung des stärksten Anstiegs von *f*
    - Eine Kante ist eine „Kurve“, entlang derer der Gradient des Bildes *f* stets in Normalenrichtung zeigt
    - Kantenpixel bei lokalen Extremstellen (Maximum oder Minimum) des Gradientenbetrags

Differenzoperator

- Erkennt Kanten, indem Differenz der Grauwerte zweier benachbarter Pixel berechnet wird (d.h. Gradientenbetrag)

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

Differenzoperator



Wie entstehen diese Kantenbilder?

Differenzoperator bildet ab auf einen Wertebereich, wo $e(i, j) \in \{-255, \dots , 255\}$

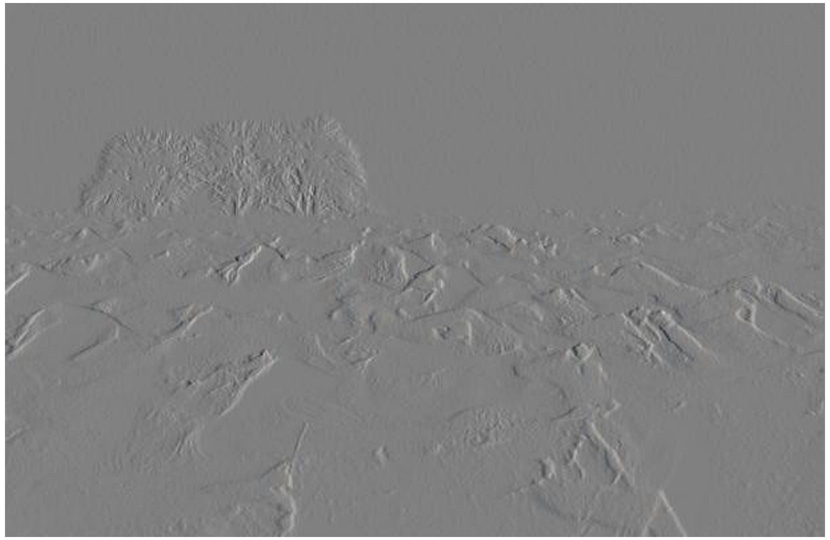
Ergebnis $e(i, j)$ nun noch auf Bereich $\{0, \dots , 255\}$ abbilden:

$g'(i, j) = (e(i, j) + 255) * 0.5$

Ergebnis damit darstellbarer Grauwert (Mittelgrau im Bild entspricht also $e(i, j) = 0$)

Differenzoperator

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

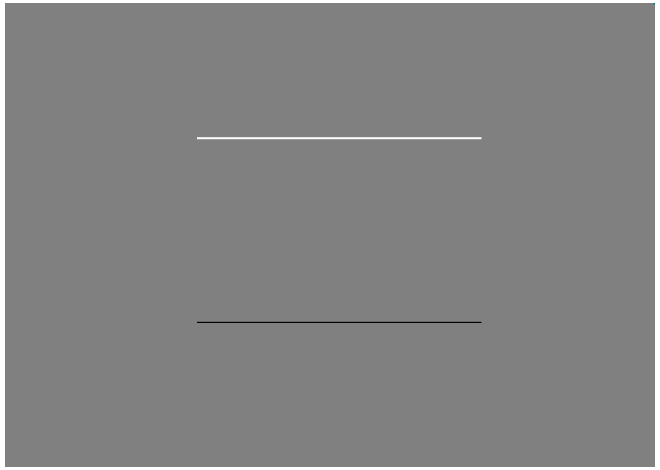


- In diesem Beispiel wurden nur die vertikal verlaufenden Kanten angezeigt
- Wie muss ein Differenzoperator aussehen, der die horizontal verlaufenden Kanten findet?

Differenzoperator

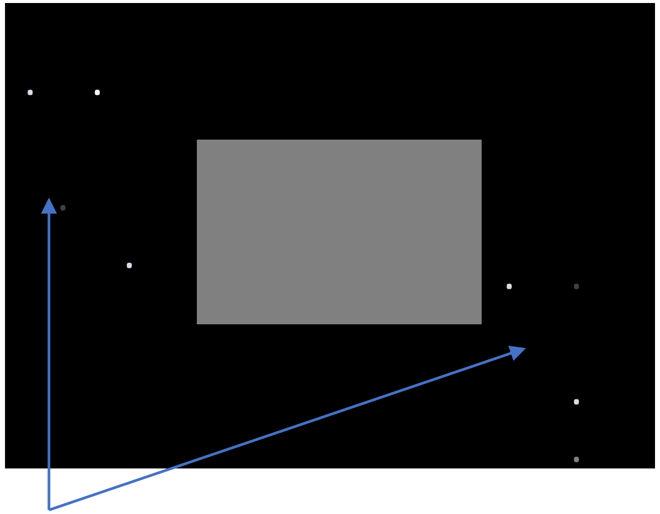
- Erkennt horizontal verlaufende Kanten, indem er die Differenz der Werte zweier benachbarter Pixel in einer Bildspalte berechnet

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

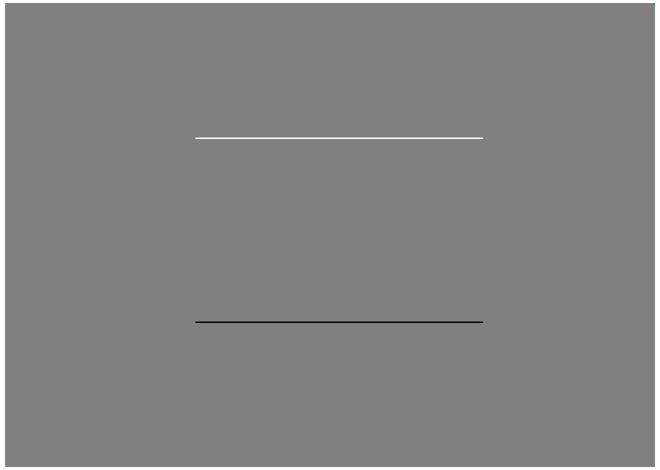


Differenzoperator

- Um Kantenerkennung robust zu machen und nicht bei jedem „fehlerhaften“ Pixel eine Kante anzuzeigen, werden benachbarte Pixel in Berechnung mit einbezogen



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



Fehlerhafte Pixel machen sich durch Rauschen im Bild bemerkbar

Weitere Differenzoperatoren

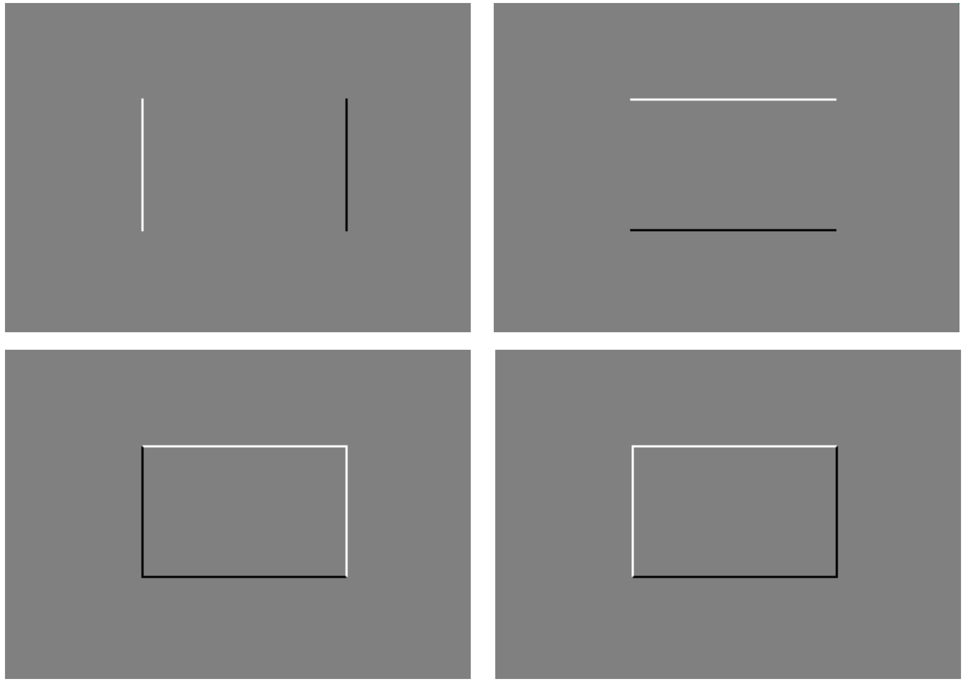
- Um Kantenerkennung noch robuster zu machen, wird „Puffer“-Zeile oder -Spalte in Operator eingefügt: Diese Differenzoperatoren heißen Prewitt-Operatoren

Faltungsergebnisse der unterschiedlichen Operatoren

–1	0	1
–1	0	1
–1	0	1

–1	–1	–1
0	0	0
1	1	1

Prewitt-Operator



0	–1	–1
1	0	–1
1	1	0

–1	–1	0
–1	0	1
0	1	1

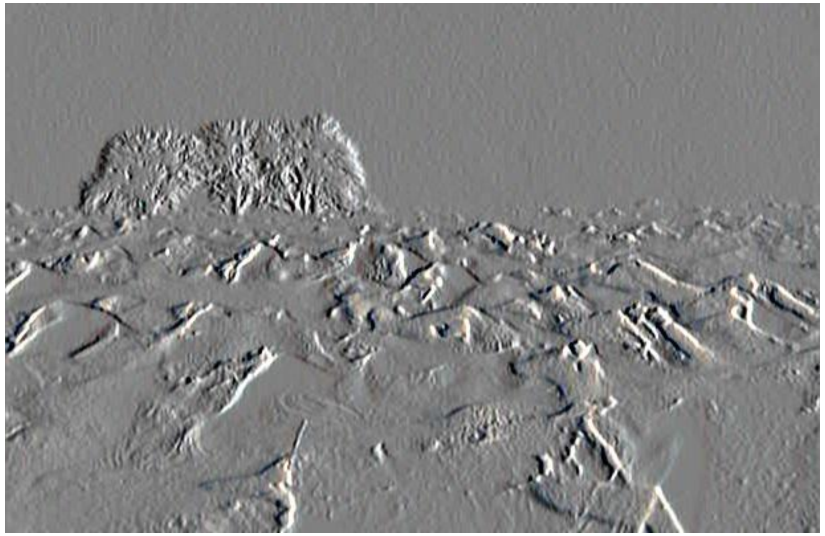
Compass Edge Detection (45°)

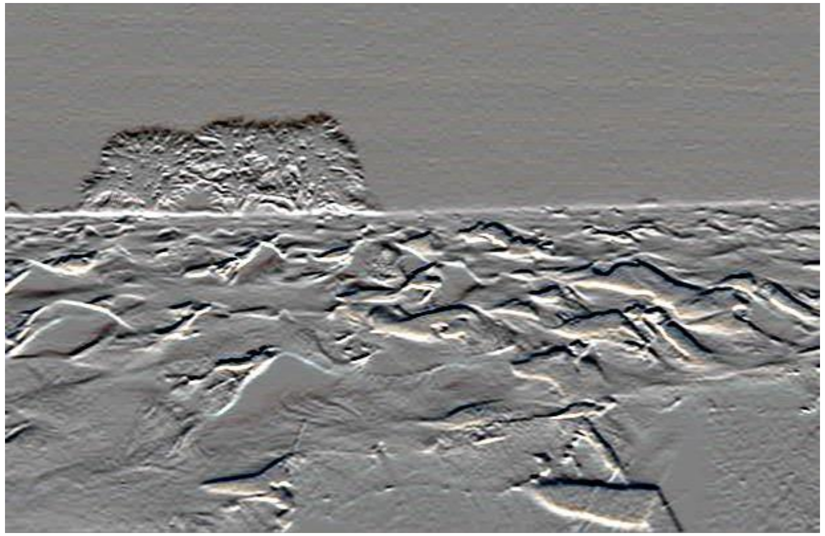
Prewitt-Operator

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

- Durch „Puffer“-Spalte bzw. -Zeile markiert Sobel-Operator die Kanten mit mehreren Pixeln (Bild grau durch Umrechnung in {0, 1, …, 255})
  - Filter separierbar, dadurch performantere Berechnung möglich

–1	0	1
–1	0	1
–1	0	1





–1	–1	–1
0	0	0
1	1	1

Sobel-Operator

- Verwendet ebenfalls zwei 3×3-Filter, die mit Originalbild gefaltet werden
- Kombination zentraler Differenzen (Prewitt-Filter) mit Gauß-Filter
  - Berechnet ebenfalls diskrete Ableitung in *x* und *y* (d.h. vertikal u. horizontal)
- Nutzt dazu nacheinander zwei Filter → Ausnutzen der Separierbarkeit
  - D.h. nacheinander in *x*- u. *y*-Richtung ausgeführte Operationen führen hier zum gleichen Ergebnis, als wären sie auf einmal erfolgt (dadurch schneller)

Horizontaler Filter: Blur 1D-Ableitungsfilter

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

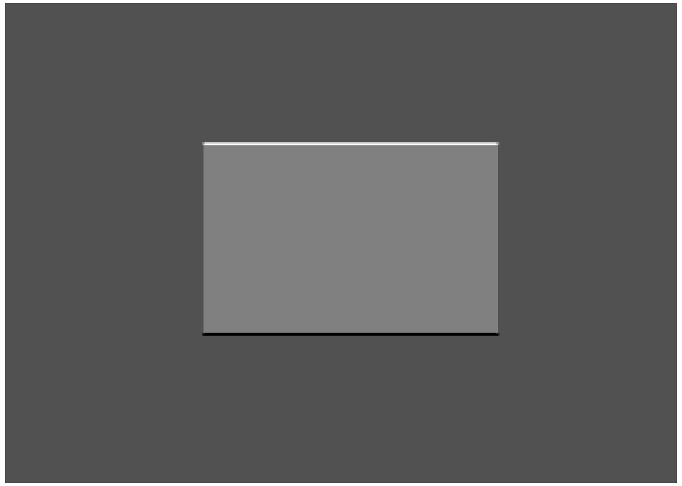
Vertikaler Filter: 1D-Ableitungsfilter Blur

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

Kanten in Originalbild hervorheben

- Ergebnis eines Differenzoperators zur Bildverbesserung ins Eingabebild einfügen
  - Nach Anwendung des Kantenoperators sind im Ergebnisbild nur Kanten sichtbar
  - Durch Addition des Originalbildes erhält man Filter F, der im Eingabebild Kanten erzeugt

$$F = n \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & n & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

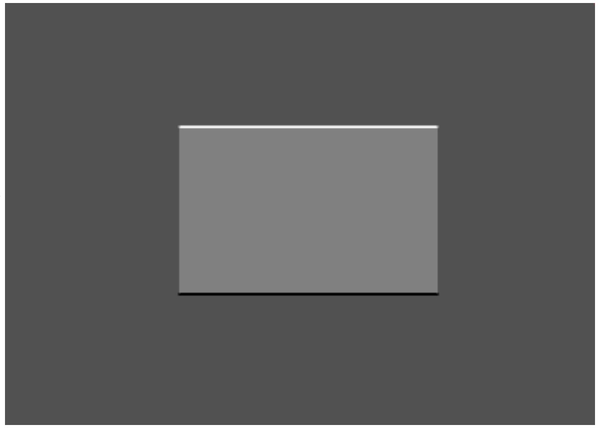
F = n ·  +  = 

Vereinfachte Darstellung ohne Berücksichtigung der linearen Transformation

Kanten in Originalbild hervorheben

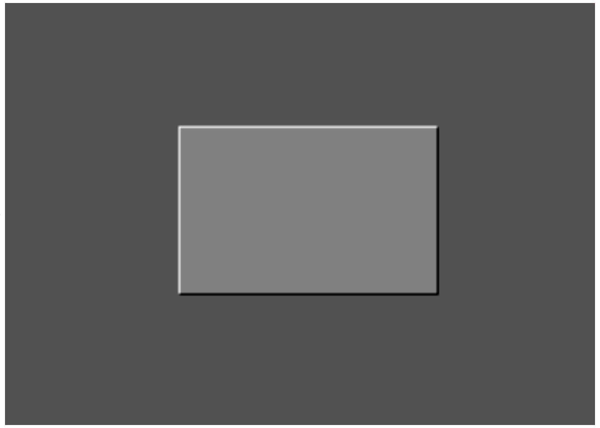
h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

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



Zur Bildverbesserung das Ergebnis eines Differenzoperators noch in Eingabebild einfügen (Addieren von n · Pixelfarbe)

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



Kombinationsfilter für mehr Kontrast

Anwendung des Filters *F* mit
je unterschiedlichem *n*:

F =

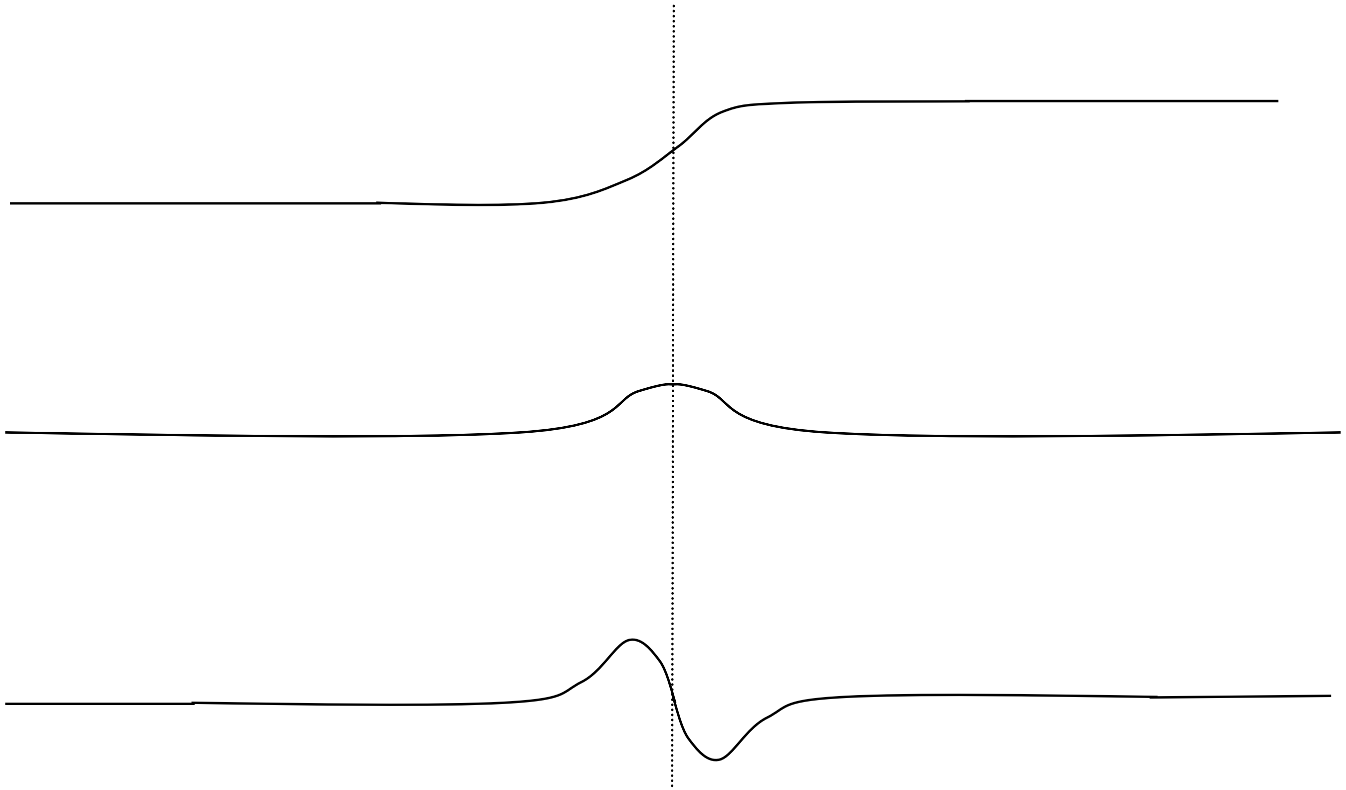
−1	−1	0
−1	n	1
0	1	1

Berechnung Kontrast zweier Objekte

- Differenz heller u. dunkler Leuchtdichte (L) geteilt durch Summe beider:
  $K = (L_{max} - L_{min}) / (L_{max} + L_{min})$

Laplace-Operator

- Erkennt Kanten anhand der 2. Ableitung



Grauwertverlauf einer Bildzeile s(x)

1. Ableitung s‘(x)

2. Ableitung s‘‘(x)

Laplace-Operator

- Die 2. Ableitung einer Funktion *g(x)* ist definiert durch:
  - $g''(x) = \lim_{\Delta x \to 0} \frac{(g(x+\Delta x) - g(x)) - (g(x) - g(x - \Delta x))}{\Delta x}$
    - Mit g(x) = Grauwert des Pixel an Position x
- Für Rasterbilder gilt: Δx nimmt im minimalen Fall Wert 1 an
  - Für die 2. Ableitung in einer Bildzeile oder -spalte gilt damit:
  - $g''(x) = g(x+1) - 2 \cdot g(x) + g(x-1)$

0	0	0
1	-2	1
0	0	0

Kantendetektion in Bildzeile

\+

0	1	0
0	-2	0
0	1	0

Kantendetektion in Bildspalte

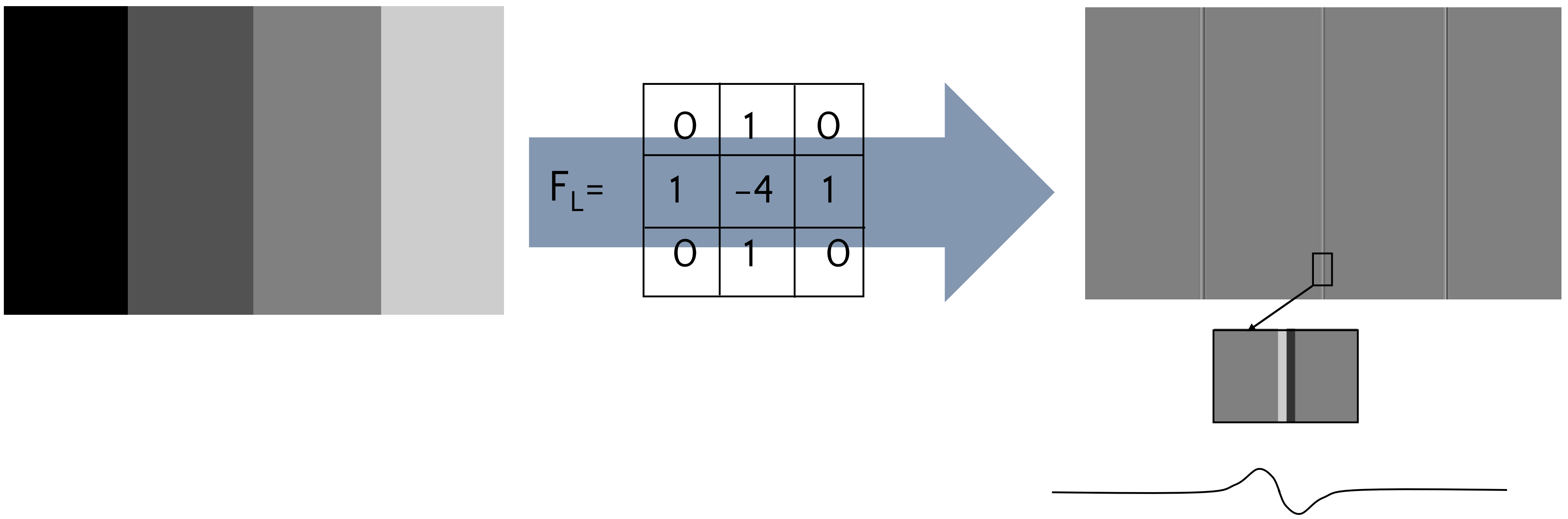
=

0	1	0
1	-4	1
0	1	0

Kantendetektion in Spalte u. Zeile

Laplace-Operator

Laplace-Operator



Entspricht dem Verlauf der 2. Ableitung

Laplace-Operator am Beispiel

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

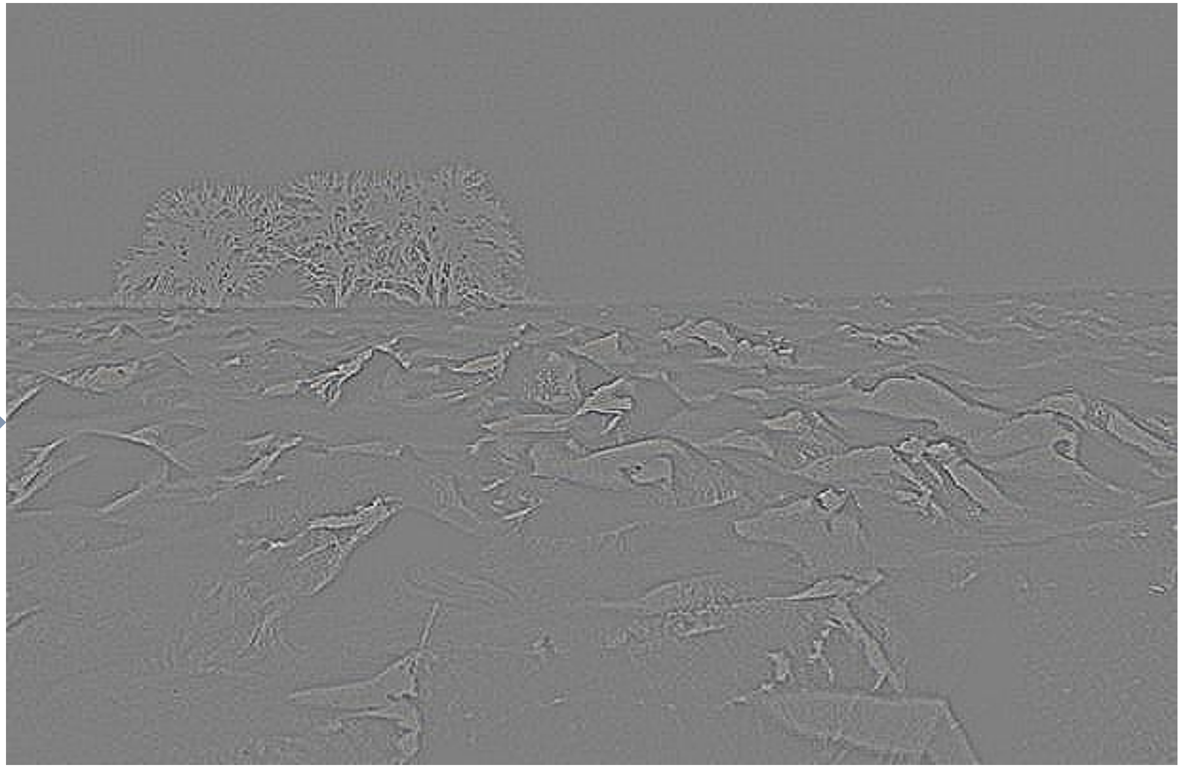
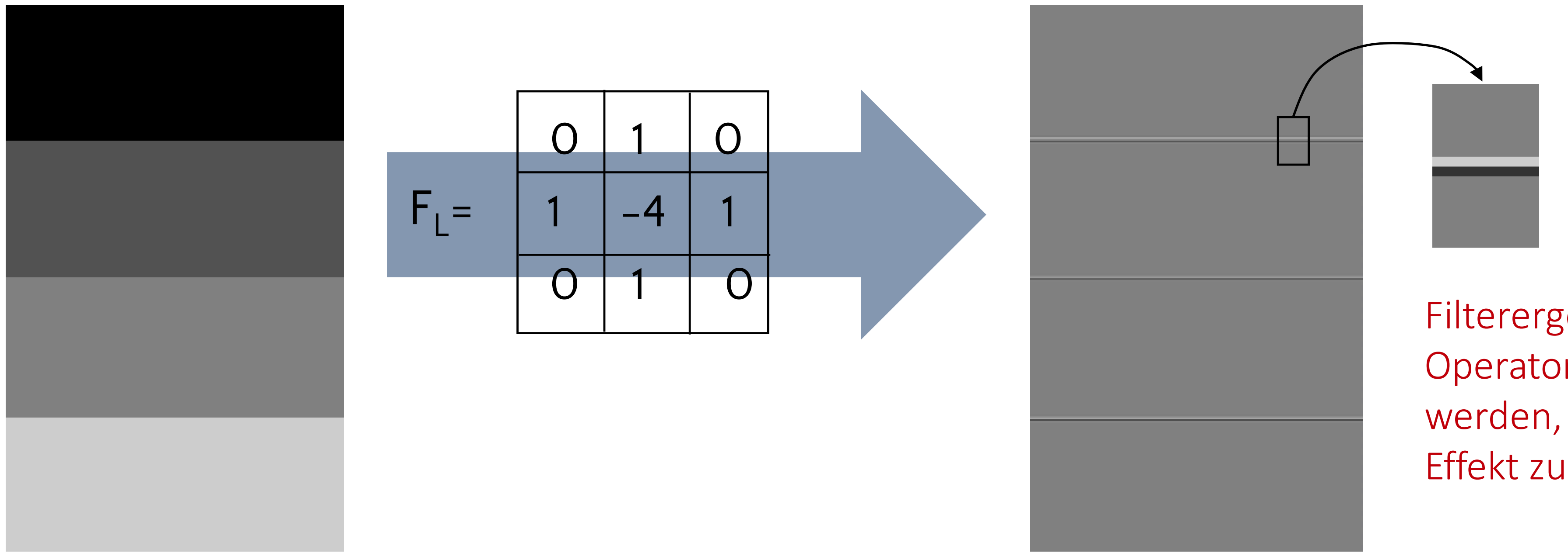


Bild vergrößert, um Kanten besser sichtbar zu machen

Kontrastverbesserungsfilter

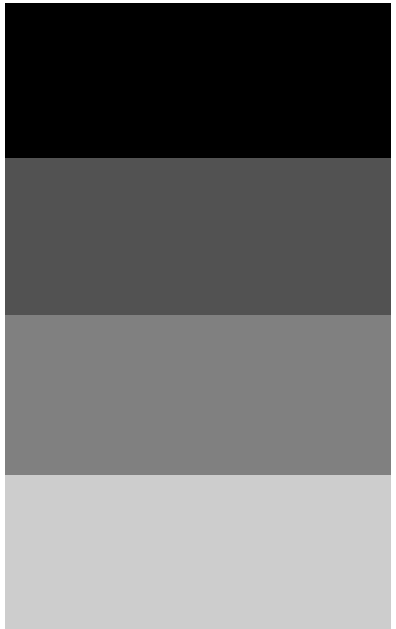
- Betont die Kante auf der hellen Seite der Kante durch hellere und auf der dunklen Seite der Kante durch dunklere Pixel
  - Entspricht damit dem Mach-Band Effekt



Filterergebnis des Laplace-Operators müsste invertiert werden, um Mach-Band Effekt zu simulieren

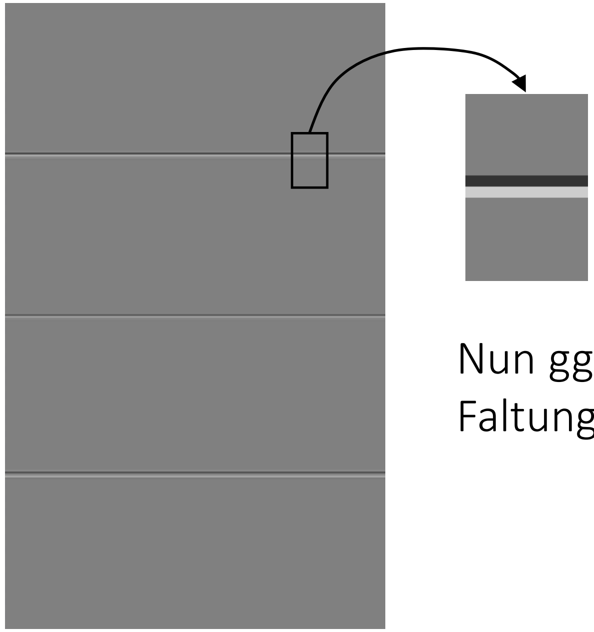
Kontrastverbesserungsfilter

- Betont die Kante auf der hellen Seite der Kante durch hellere und auf der dunklen Seite der Kante durch dunklere Pixel
  - Entspricht damit dem Mach-Band Effekt



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

Invertierter Laplace-Operator

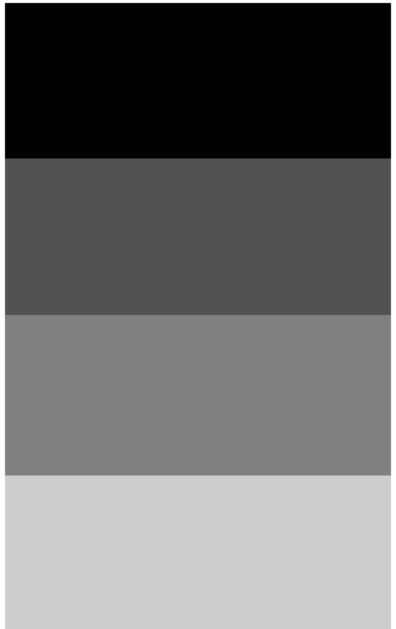


Nun ggfs. noch Eingabebild in Faltungsergebnis integrieren

Kontrastverbesserungsfilter

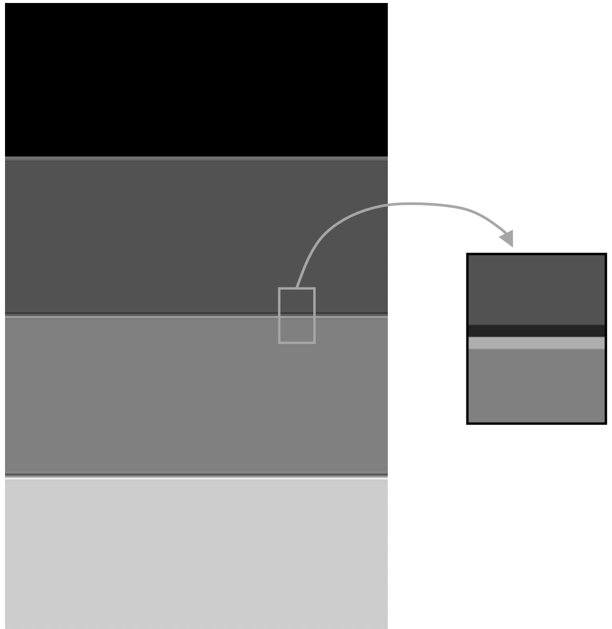
- Integration des Eingabebildes in das Faltungsergebnis:

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



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

Kontrastverbesserungsfilter



Beispiel Kontrastverbesserungsfilter

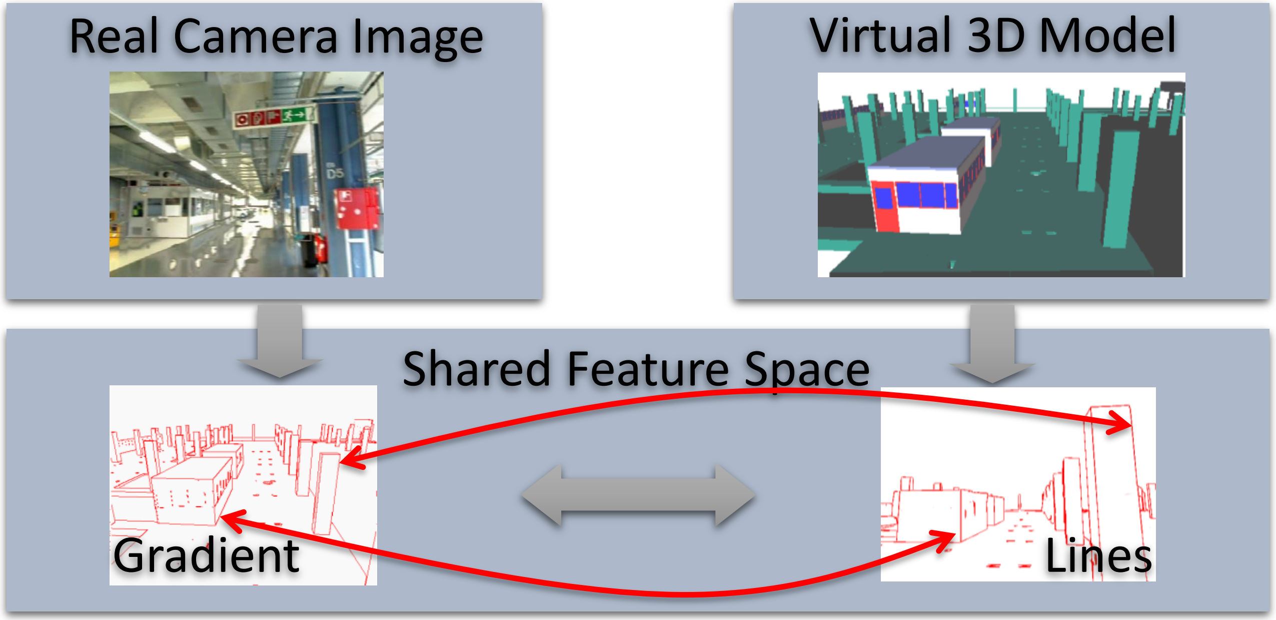
- Filter betont Kanten

$$F_K = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Kontrastverbesserungsfilter

Anwendung in Computer Vision

- Kantenerkennung ist oft Anfangsschritt eines Tracking-Algorithmus für Augmented Reality (AR)
- Kann u.a. zur Bestimmung der extrinsischen Kameraparameter dienen (sog. Kamera-Pose), d.h. Position und Orientierung der realen Kamera (sowohl initial als auch Frame-to-Frame Tracking)
  - Intrinsische Parameter (Projektionsmatrix, ggfs. Verzerrungen) mittels initialer einmaliger Kalibrierung

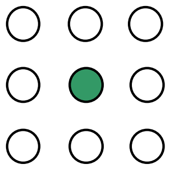


Rangfolgeoperatoren

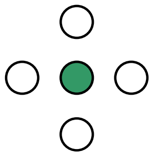
Median, Dilatation, Erosion

Rangfolgeoperatoren

- Aktuell betrachtete Bildposition: g(i, j)
  - Bei N8-Nachbarschaft:
    - Bezugspunkt:
    - Nachbarpunkt:



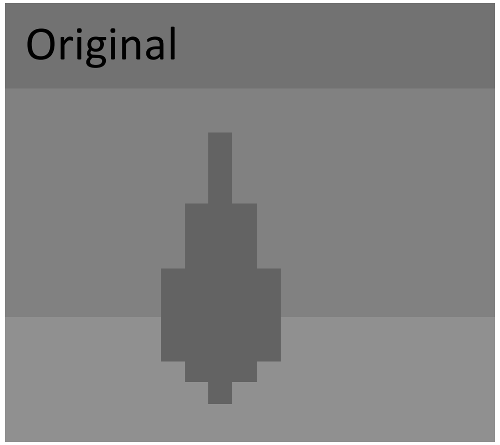
  - N4-Nachbarschaft:
    - Analog

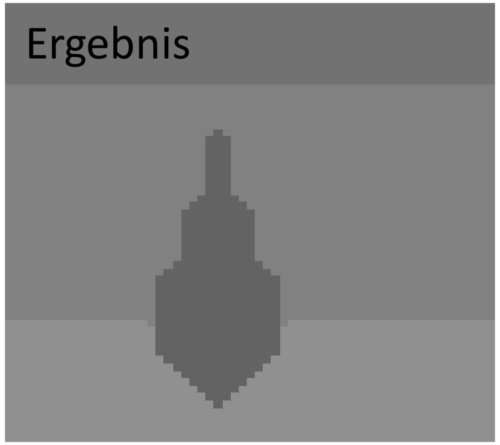


- g(i, j) und die Grauwerte der Nachbarschaft werden größenabhängig sortiert: $g_0 \leq g_1 \leq \ldots \leq g_n$
- Rangfolgeoperatoren wählen nun Wert an bestimmter Positionen dieser Sortierung aus...

Medianfilter

- Bezugspunkt nimmt *mittleren* Grauwert der Rangfolge an:
  1. Sortieren: $g_0 \leq g_1 \leq \ldots \leq g_8$
  2. Für eine N8-Nachbarschaft gilt: $g'(i,j) = g_4$
     - Nach Sortierung wird also mittlerer Wert ausgewählt
- Verbesserung von verrauschten Bildern
  - Eliminiert isolierte, fehlerhafte Bildpunkte
- Kanten werden jedoch nicht verwaschen
  - Anders als bei Mittelwertoperator



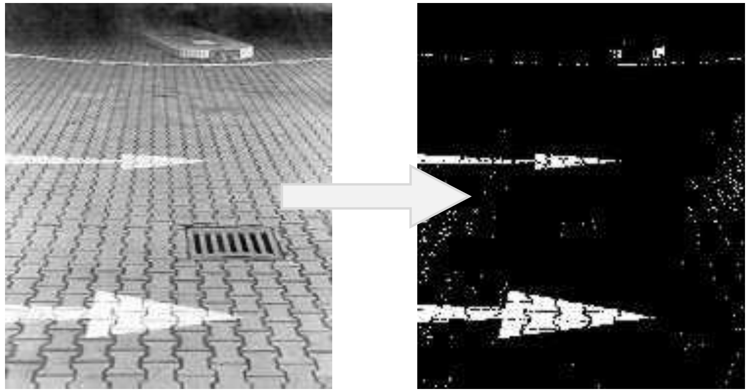


Dilatation & Erosion

- Dilatation: Bezugspunkt nimmt *maximalen* Grauwert der Rangfolge an
  1. Sortieren: $g_0 \leq g_1 \leq \ldots \leq g_8$
  2. Für eine N8-Nachbarschaft gilt: g‘(i,j) = g_8
- Folge: Ausdehnung der „helleren“ Bereiche (i.d.R. Bildvordergrund)

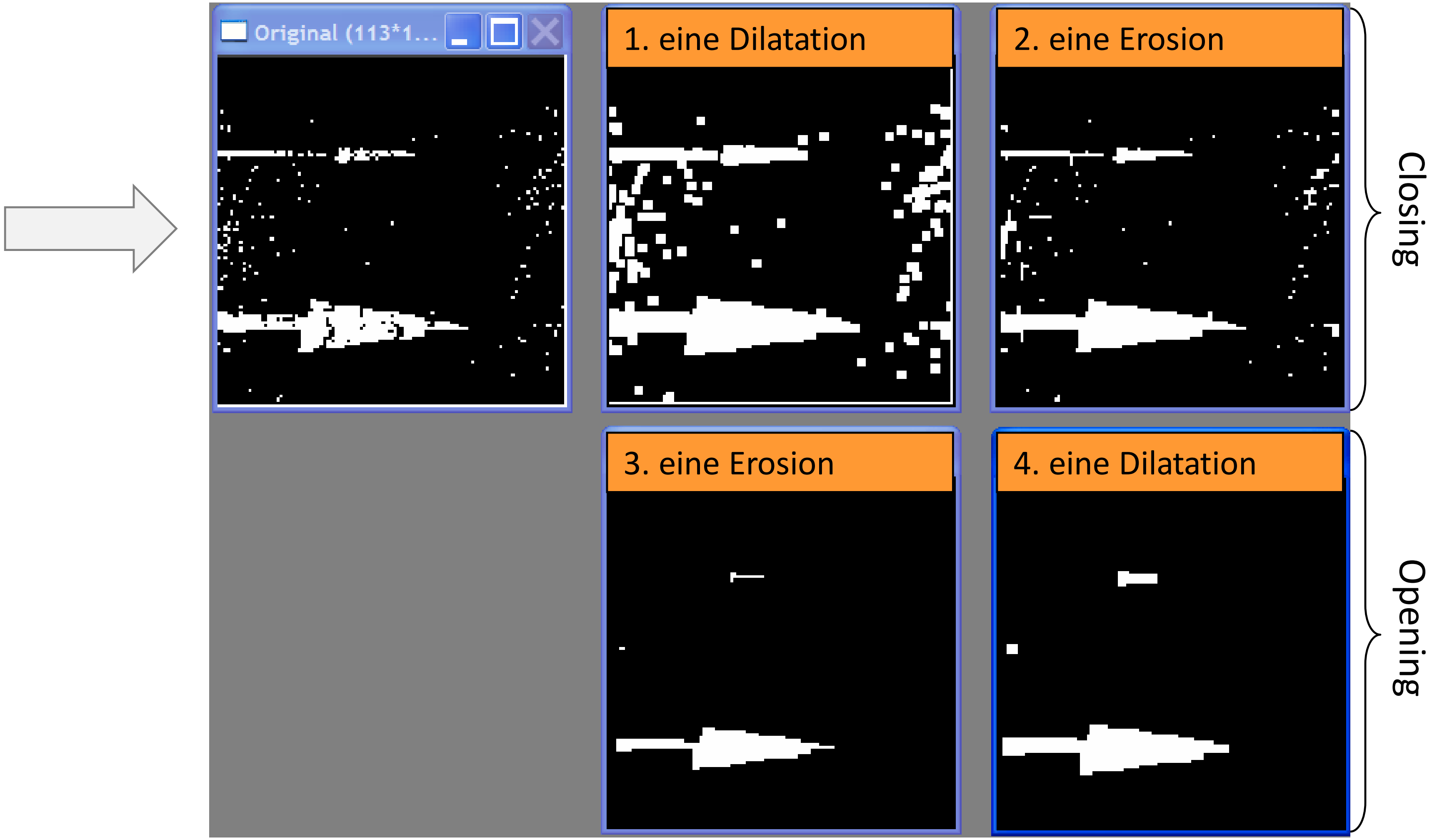
- Erosion: Bezugspunkt nimmt *minimalen* Grauwert der Rangfolge an
  1. Sortieren: $g_0 \leq g_1 \leq \ldots \leq g_8$
  2. Für eine N8-Nachbarschaft gilt: g‘(i,j) = g_0
- Folge: Ausdehnung der „dunkleren“ Bereiche (i.d.R. Bildhintergrund)

Opening & Closing



- Ziel: Segmentierung des Bildes
  - Zur Unterscheidung einzelner Objekte
- Originalbild erst geeignet binarisieren
  - Danach Rauschen eliminieren
  - Dann Lücken innerhalb der Objekte schließen
- Lösung zur Bildaufbereitung
  - Opening (erst Erosion, dann Dilatation, um Rauschen zu entfernen) gefolgt von Closing (erst Dilatation, dann Erosion)
  - Alternativ umgekehrt zuerst Closing (erst Dilatation, dann Erosion) gefolgt von Opening (erst Erosion, dann Dilatation)

Beispiel



Original (113*1...
1. eine Dilatation
2. eine Erosion
3. eine Erosion
4. eine Dilatation
Closing
Opening

Fazit und Ausblick

- Bildverarbeitung geht davon aus, dass in einem Bild Informationen stecken, die man daraus extrahieren will
  - Es werden keine Bilder manipuliert, sondern auf darin enthaltene Informationen überprüft
- In Computer Vision werden Bilder analysiert, um deren Bildinhalte zu verstehen und ggfs. geometrische Informationen zu extrahieren (z.B. für Tracking)
  - Nutzt i.d.R. auch Verfahren des maschinellen Lernens (Machine Learning)
    - Statt Regeln aufzustellen, um Antworten zu erhalten, werden bekannte Antworten genutzt, um generalisierte Regeln zu erhalten, die auch für unbekannte Daten Antworten liefern sollen
  - Computer Vision Anwendungen können mit Bildverarbeitungsverfahren und ML-Verfahren, z.B. Convolutional Neural Networks (CNN), realisiert werden
    - Bekannte Techniken aus Bildverarbeitung, wie z.B. Faltungsmatrizen zur Merkmalsdetektion, werden auch in CNNs verwendet
    - Objekte, die sich algorithmisch schwer definieren lassen, können so durch CNN erkannt werden

Vielen Dank!

Noch Fragen?