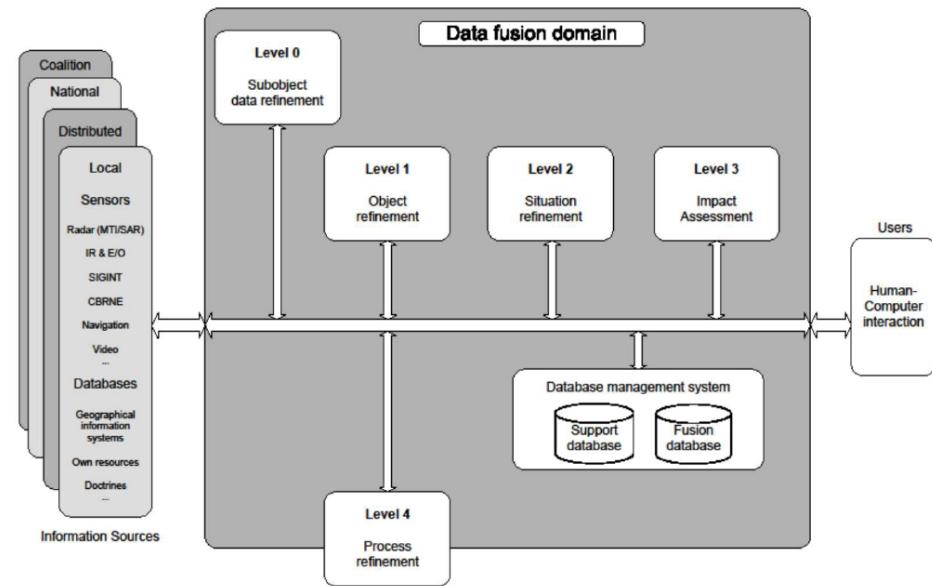
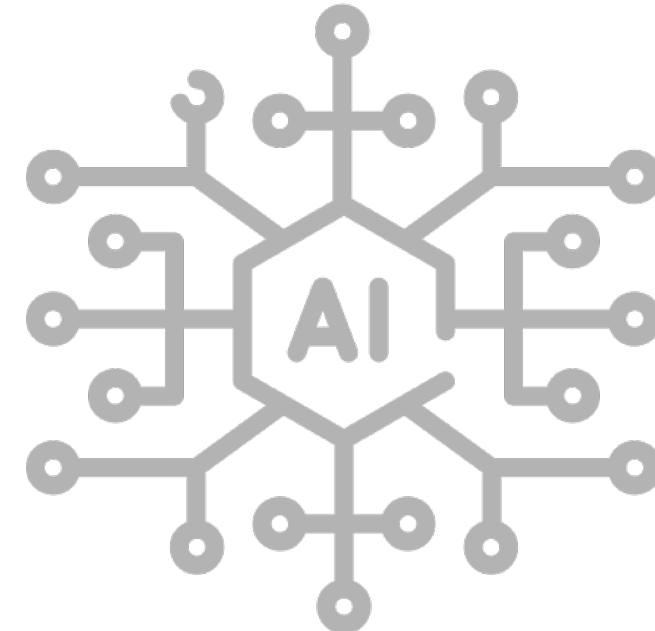


INTELLIGENTE SEHSYSTEME AUFGABEN, PROBLEME, BEGRIFFE



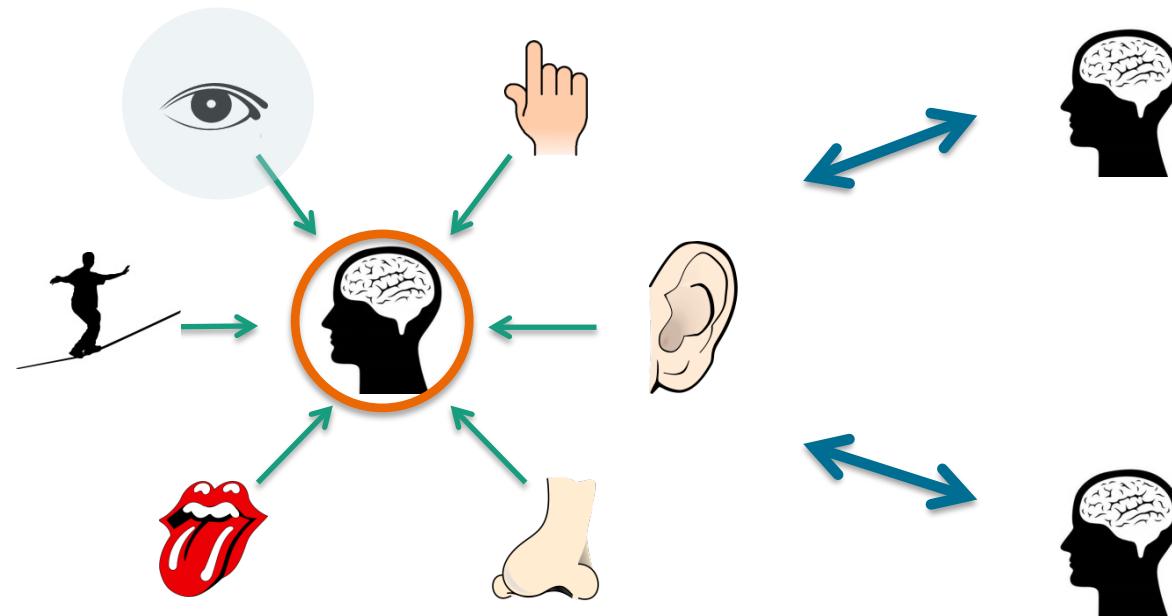
Joint Directors of Laboratories (JDL): Data Fusion Model



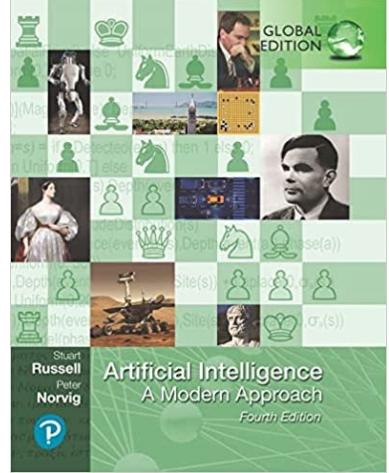
vor jeder technischen Umsetzung oder wissenschaftlichen Reflektion: Intelligentes Agieren als allgegenwärtiges Phänomen!

Alle Lebewesen verknüpfen Eindrücke einander ergänzender Sinne mit bisherigen Erfahrungen und Mitteilungen anderer Lebewesen. Daraus ergeben sich Umweltmodelle, die Voraussetzung für situationsgerechtes Verhalten.

Intelligente Sehsysteme:



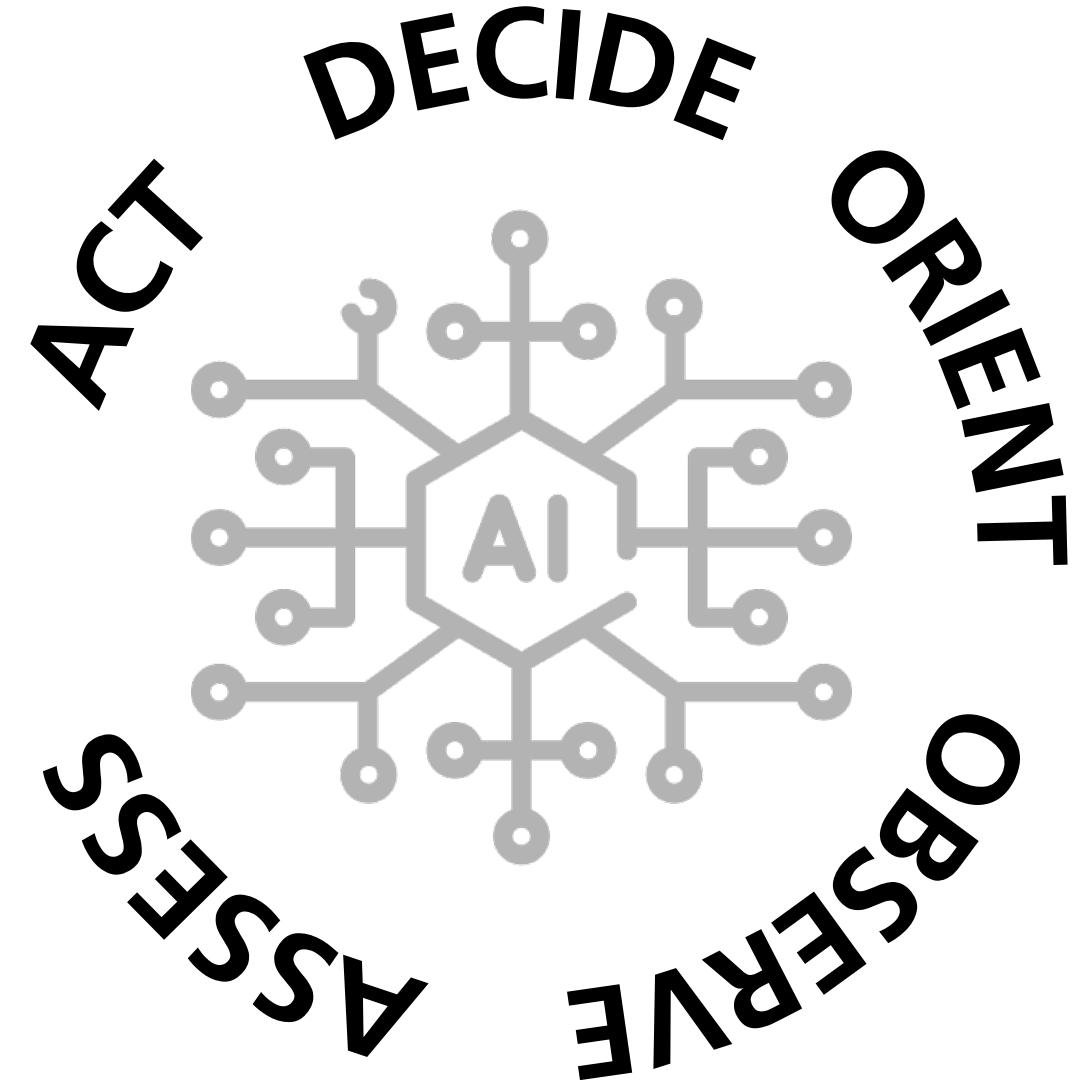
Wahrnehmen, Entscheiden, Wirken



modellbasiert

“sense”

Ressourcen-Management



zu erreichende Ziele, Umgebung

“AI is the ability of machines to perform tasks that normally require human intelligence. Even old technology can still be AI.”

US DoD AI Strategy

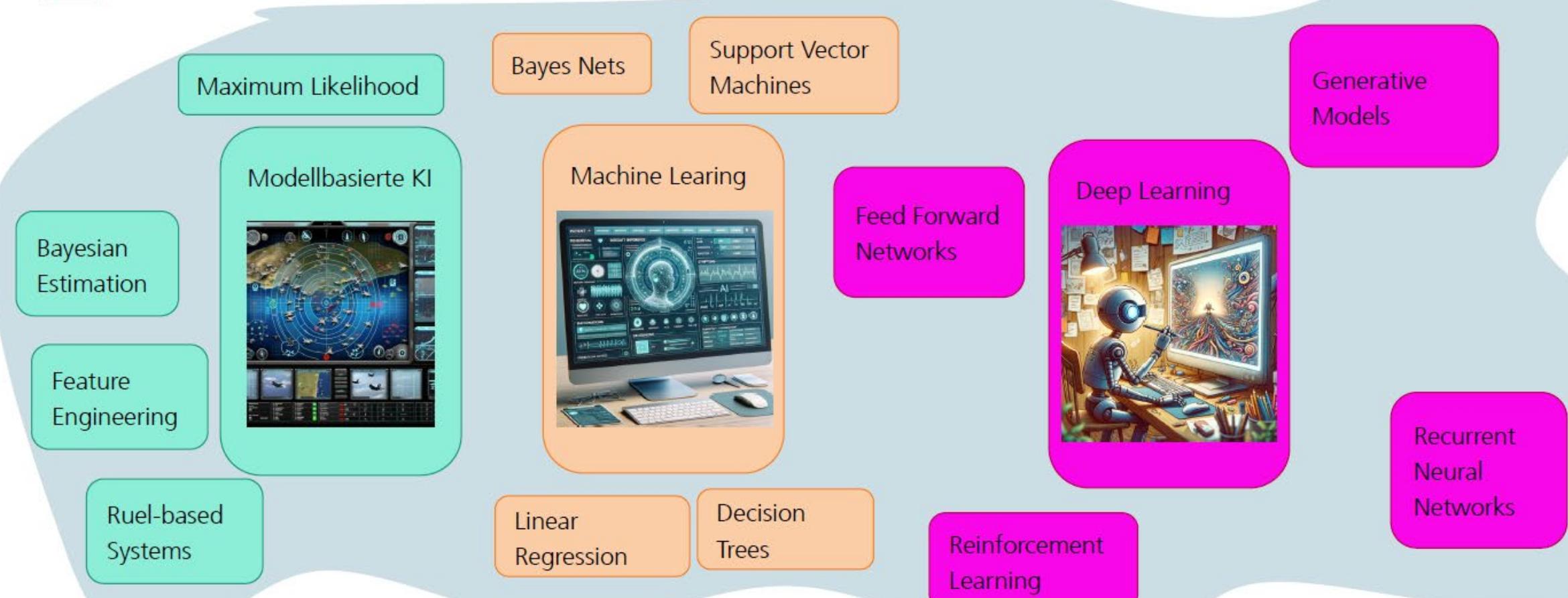
Multisensor-Datenfusion

datengetrieben

“sensibility”



Überaus grobe Landkarte der KI-Methoden



Understandable **Cognitive** and **Volitive** Assistance

- to evaluate imperfect and incomplete **mass data**,
- to fuse **context** knowledge with current data streams,
- to fuse complementary and heterogeneous sources,
- to estimate the **plausibility** of the information content,
- to enable **manned-unmanned teaming** and action, and
- to guarantee ethical, legal, and societal **compliance**.

Unburden persons so that they can do, what only persons can do:
acting **intelligently** and in **autonomous responsibility**.

responsible decision maker

volution:
why?

To what end?

In what way?

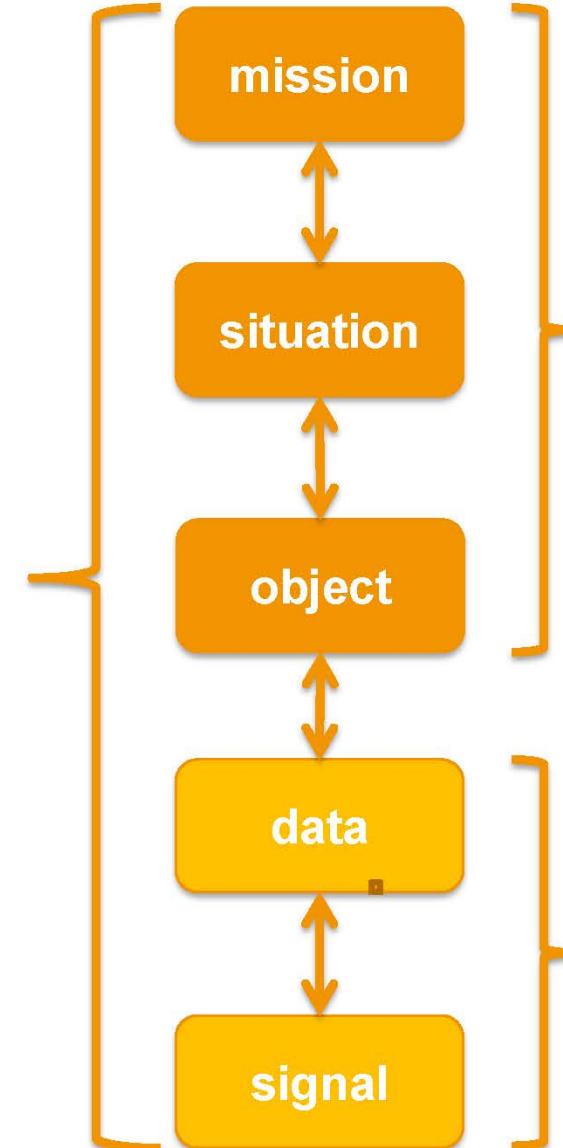
Which norm obeyed?

By what means?

**model-based
algorithms:
“causality”.**

Technical Automation

levels of perception



information levels data levels

Artificial Intelligence

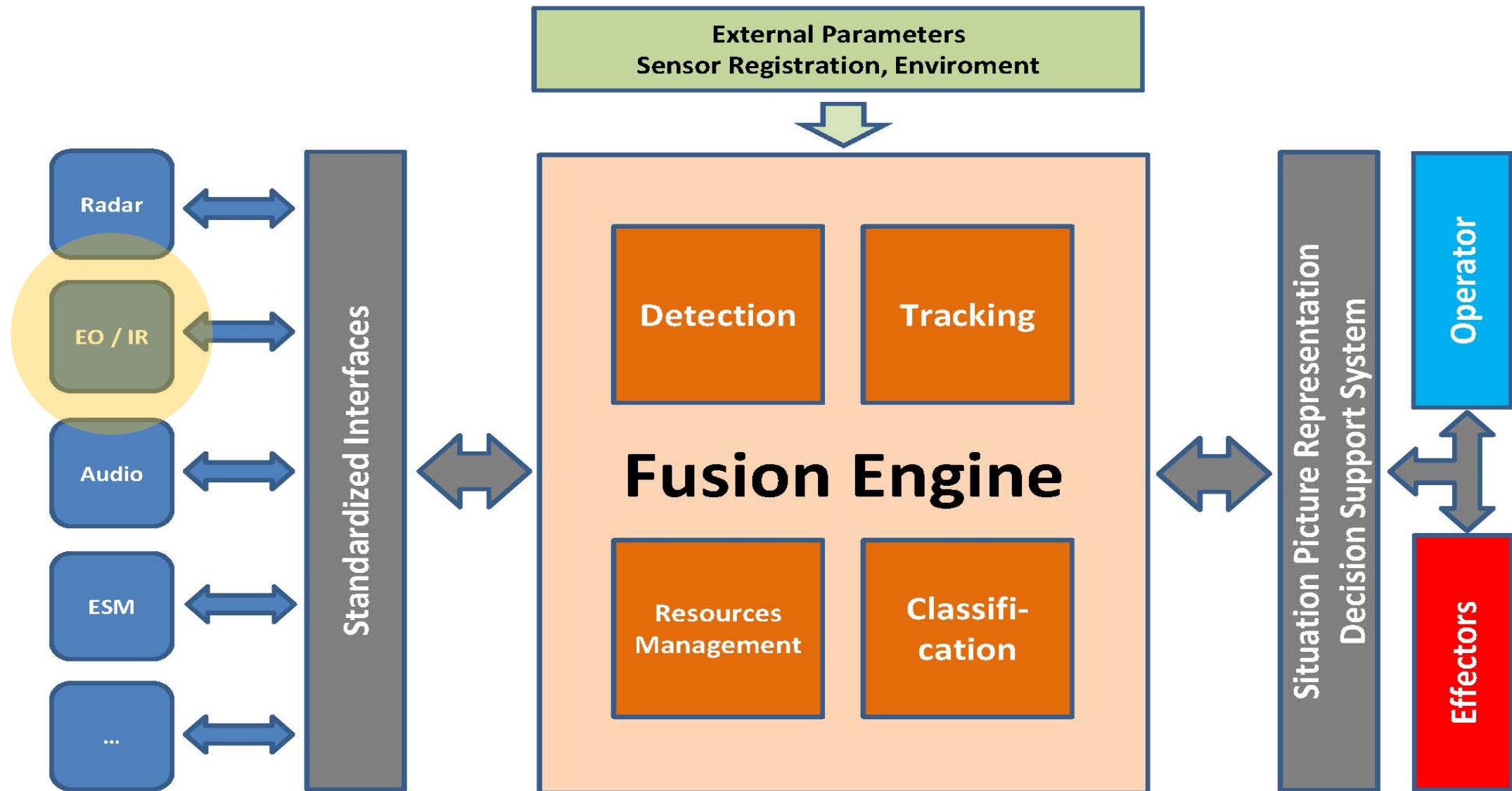
cognition:
what?

detection: existence
classification: essence
inter-object relations
decision relevance

**data-driven
algorithms:
“sensuality”.**

mission, environment

Fusion Engines – Link between Sensors, Context, Action



Intelligente Sehsysteme

Vorlesung:

Übungen:

Zeit und Ort:

Dienstags, 10¹⁵ – 11⁴⁵

Hörsaal 7 (HSZ) & online

Dozenten:

Prof. Wolfgang Koch

Dr. Felix Govaers

Roman Bartolosch

Florian Oßwald

Henry Hölzemann

Sprechstunden: n. Vereinbarung

Einordnung:

BA-INF 131

Intelligente Sehsysteme

6 LP / 2 + 2 SWS

Zeit und Ort:

Di, 08¹⁵ - 09⁴⁵, U.039

Mi, 08¹⁵ - 09⁴⁵, U.039

Mi, 10¹⁵ - 11⁴⁵, U.039

Do, 08¹⁵ - 09⁴⁵, U.039

TutorInnen:

- Benedikt Wude
s6bewude@uni-bonn.de
- Denis Jan Schafranski
s6dsscha@uni-bonn.de

Weitere Infos dazu dann:

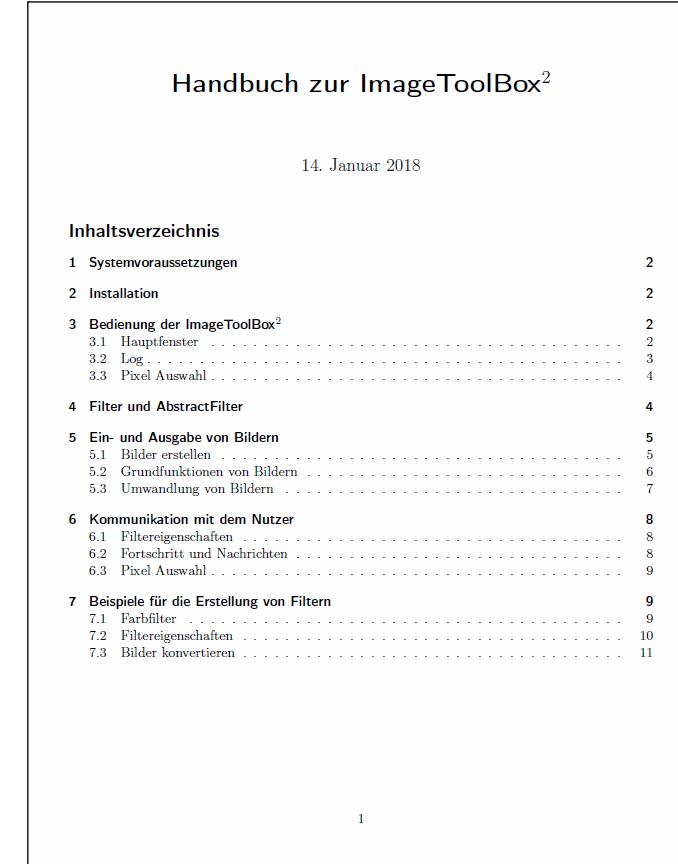
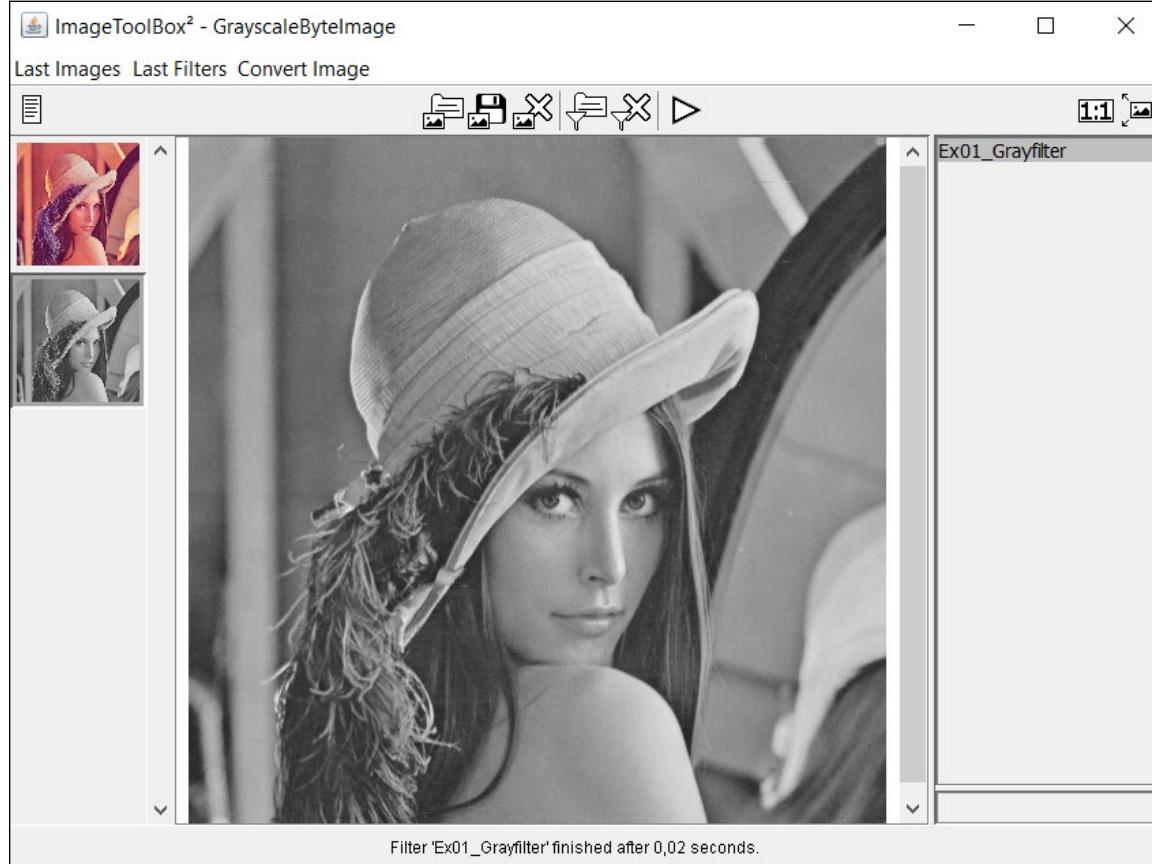
eCampus ISS Vorlesung

Themen, im Laufe der Vorlesung behandelt

- Phasen des Computersehens
- Histogrammbasierte Bildverarbeitung
- Lineare Filter
- Nichtlineare Filter
- Histogrammbasierte Segmentierung
- Homogenitätsbasierte Segmentierung
- Diskontinuitätsbasierte Segmentierung
- Modellbasierte Segmentierung
- Fourier-Transformation
- Skalenraum
- Interest Points
- Objekterkennung
- Merkmale
- 3D-Punktwolken
- Tiefe Konvolutionsnetze

ImageTool Box²:

ITB² für die Bearbeitung von Übungsaufgaben



Inhaltsverzeichnis

1	Systemvoraussetzungen	2
2	Installation	2
3	Bedienung der ImageToolBox ²	2
3.1	Hauptfenster	2
3.2	Log	3
3.3	Pixel Auswahl	4
4	Filter und AbstractFilter	4
5	Ein- und Ausgabe von Bildern	5
5.1	Bilder erstellen	5
5.2	Grundfunktionen von Bildern	6
5.3	Umwandlung von Bildern	7
6	Kommunikation mit dem Nutzer	8
6.1	Filtereigenschaften	8
6.2	Fortschritt und Nachrichten	8
6.3	Pixel Auswahl	9
7	Beispiele für die Erstellung von Filtern	9
7.1	Farbfilter	9
7.2	Filtereigenschaften	10
7.3	Bilder konvertieren	11

Literatur



A screenshot of a website page from Pearson STARK. At the top, the Pearson logo and the word "STARK" are displayed. The navigation menu includes links for Shop, Schule, Studium, Berufsleben, Englisch, International Schools, FAQ, Handel, Über uns, and a language selection for German. On the right side of the header are icons for user profile, shopping cart (with a red notification dot), and search. The main content area shows the book "Grundlagen der Bildverarbeitung" by Klaus D. Tönnies. It includes a thumbnail image of the book cover, the author's name, the title, and a brief description: "Tönnies führt in seinem Buch in die grundlegenden Aspekte der Bildverarbeitung ein. Dabei verknüpft er Motivation und... Mehr anzeigen". Below the description is a price of "23,99 €" with a note "Inkl. MwSt.", a green "Lieferbar" button with an info icon, and a link "Sofort zum Download". On the left side of the main content, there is a sidebar with links for Leseprobe, Inhaltsverzeichnis, Auf die Wunschliste, Feedback, Produktdetails, Artikelbeschreibung, and Extras Online.

<https://www.pearson.de/grundlagen-der-bildverarbeitung-9783863266370>

... sowie andere Quellen, die in jeweiliger Vorlesung genannt werden

Erforderliche Studienleistungen

Bearbeitung regelmäßig erscheinender Übungsblätter

- Die Bearbeitung soll in Gruppen von 3 Studierenden erfolgen.
- Insgesamt müssen mind. 50 % der Punkte erreicht werden.
- Jeder Student/jede Studentin muss **2**-mal die Lösung einer Aufgabe vorstellen. Die erste Vorstellung muss für eines der ersten fünf Übungsblätter erfolgen. Die zweite Vorstellung muss für eines der nächsten fünf Übungsblätter erfolgen.



Startseite	Aktuelles	Lehre	Arbeitsgruppen	Abteilung	Forschung	Verwaltung
Aktuelle Seite: Arbeitsgruppen Sensor Data and Information Fusion Lehre SS 2025 Einführung in die Sensordatenfusion						

Vorlesung: Einführung in die Sensordatenfusion

Sensordatenfusion verknüpft unvollständige und fehlerhafte, aber einander ergänzende Messdaten, so dass ein zugrundeliegendes Phänomen der Realität besser verstanden wird. Die Vorlesung vermittelt dazu benötigten Grundlagen, die anhand vieler Anwendungsbeispiele veranschaulicht werden. Die Studierenden lernen dadurch wichtiges Handwerkszeug der Schätz- und Filterungstheorie, der Simulation und Performance-Evaluation kennen, die auch in anderen Gebieten der Informatik nützlich sind. Die benötigten Grundbegriffe der Stochastik werden in der Vorlesung eingeführt. Freude an mathematischer Einsicht und Geschick bei der Implementierung von Algorithmen sind Voraussetzung. Geeignete Studierende können im 5. Semester im Fraunhofer FKIE an Projekten mitwirken und/oder ihre Bachelor-Arbeit schreiben. Im Master-Studiengang kann das Thema weiter vertieft werden. Informatikerinnen und Informatiker mit Kenntnissen der Sensordatenfusion sind sehr gesucht. Ein überaus fesselndes Arbeitsgebiet mit zahllosen Anwendungen wartet auf sie, das bisher nur angekratzt wurde: das „Internet der Sensoren“.

Veranstaltung:

- **Verantwortlich:** Prof. Dr. Wolfgang Koch
- **Beginn:** 11.04.2025
- **Zeiten:** Fr. 14:00 - 16:00, HSZ - HS3
- **Veranstaltungsnummer:** 614051014

Übung:

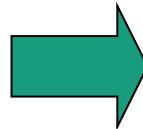
- **Verantwortlich:** Prof. Dr. Wolfgang Koch u. M.
- **Beginn:** 30.04.2025
- **Zeiten:** Mi. 16:00 -18:00, Uhr, HSZ - HS3

Weitere Informationen:

- **Studiengang:** Bachelor Informatik, Bachelor Cyber Security
- **Voraussetzungen:**
- **Fachbereich:** BA-INF 137
- **Aufwand:** 2V+2Ü / 6LP
- **Prüfungen:**

Intelligente Sehsysteme haben das Ziel, digitale Bilddaten zu interpretieren.

Bild: Helligkeiten, Farben



Bildquelle: Daimler AG

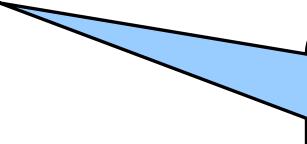
Interpretation of image scene: objects, interrelations:

- Near range:
pedestrians and car from aside on
the own lane
- Mid range:
bicycle from aside, traffic signs and
pedestrians on the own lane
- Own lane:
first a straight run,
then a slight left turn

Bildinterpretation als inverses Problem (1)

- **Ausgangspunkt:** Annahme einer Funktion f , welche die *Umwelt W* auf den *Sensorstimulus S* abbildet:

$$S = f(W)$$



Für die visuelle Wahrnehmung ist diese durch Physik und Optik definiert und i. W. durch die Computergrafik gelöst

- **Computersehen (engl. Computer Vision)** als Umkehrung der Computergraphik:

**Berechne die abgebildete Welt W aus gegebenem
Funktional f und Sensorstimulus S nach:**

$$W = f^{-1}(S)$$



Daher wird die Computer Vision auch als „inverse Computergrafik“ bezeichnet.

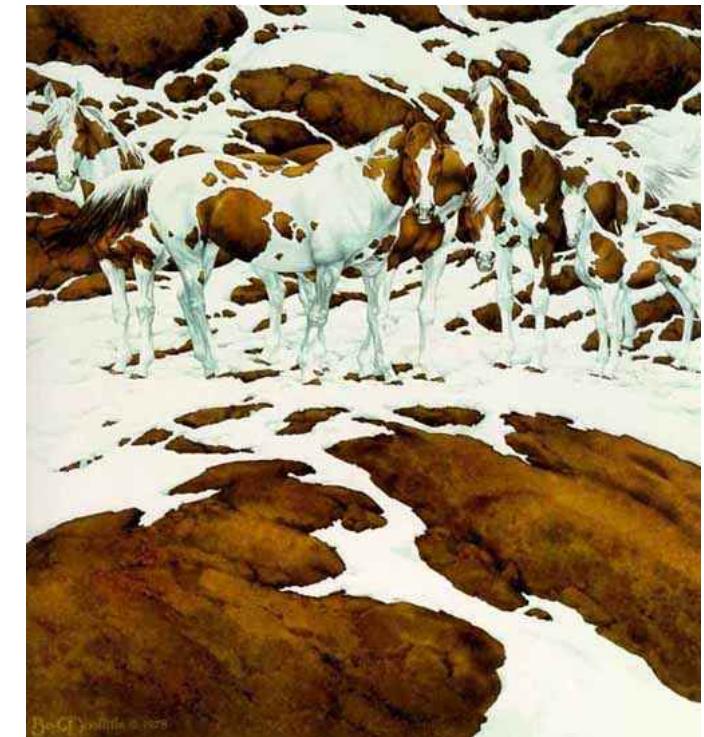
Bildinterpretation als inverses Problem (2)

Bildinterpretation als Rekonstruktion der abgebildeten Welt W für ein gegebenes **Abbildungsfunktional f** und einen Sensorstimulus S nach: $W = f^{-1}(S)$

ist ein *inverses Problem*: Schließen von der beobachteten *Wirkung (Abbildung)* eines Systems (Funktionals) auf die zugrunde liegenden *Ursachen* (abgebildete Welt).

Dieses Interpretationsproblem ist i.A. *unbestimmt* bzw. *schlecht gestellt* (engl. *ill-posed*), da die *Interpretation f^1* generell *mehrdeutig* ist.

Ein mathematisches Problem heißt *gut gestellt*, wenn gilt: (1) das Problem hat eine Lösung (Existenz), (2) die Lösung ist eindeutig (Eindeutigkeit), (3) die Lösung hängt stetig von den Eingangsdaten ab (Stabilität). Ist eine der Bedingungen nicht erfüllt, das Problem *schlecht gestellt*.

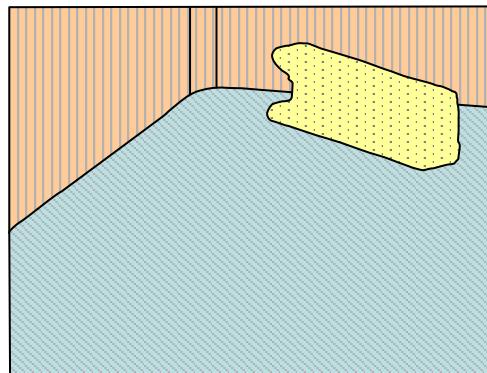


Bildquelle: *Pintos* von Bev Doolittle.

Klassische Phasen des Computersehens (1)

- 1) *Early Vision* oder *Low-Level Vision*: **Kontrastoptimierung**; **Glättung** des Rohbildes zur Eliminierung von Rauschen; Hervorhebung relevanter Bildpunkte, die z.B. Konturpunkte
- 2) *Mid-Level Vision*: z.B. Gruppierung von Konturpunkten zu **Konturlinien**; die Konturlinien zerlegen das Bild in flächenhafte Bereiche, sog. **Bildsegmente**
- 3) *High-Level Vision*: Zuordnung der Bildsegmente zu Objektklassen; eine inhaltliche Beschreibung als Interpretation des Bildes ist damit ableitbar

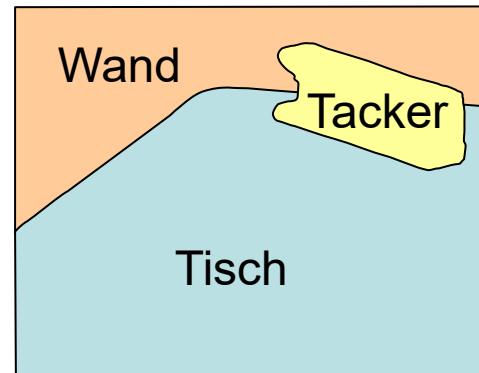
Klassische Phasen des Computersehens (2)



High-Level Vision



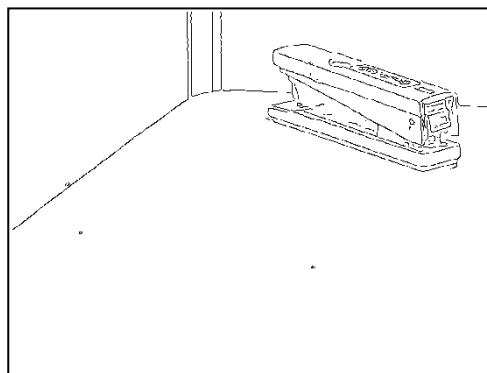
Segmente → Semantik



Mid-Level Vision



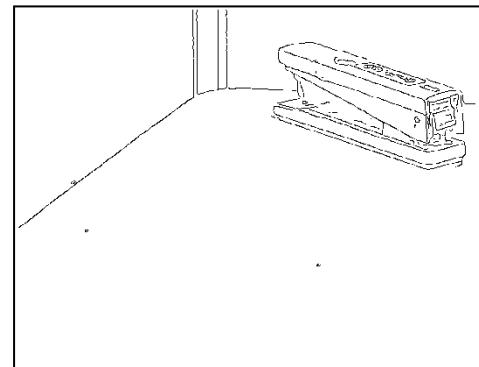
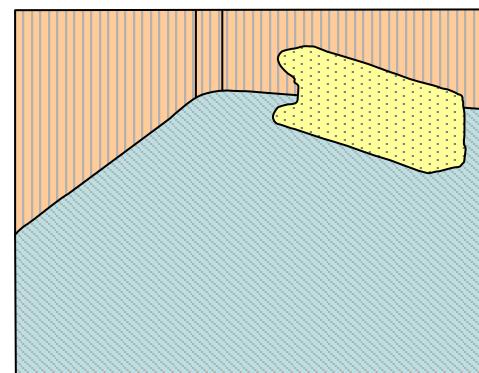
Raster → Segmente



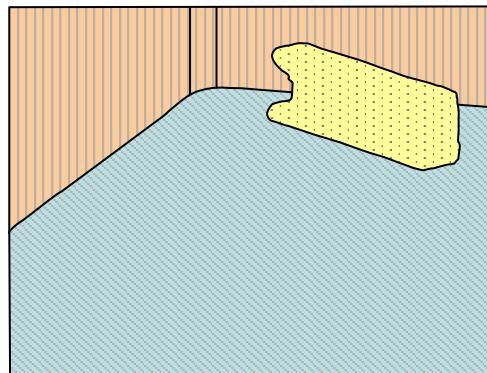
Low-Level Vision



Raster → Raster



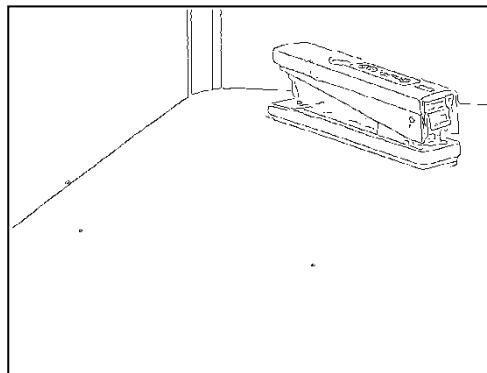
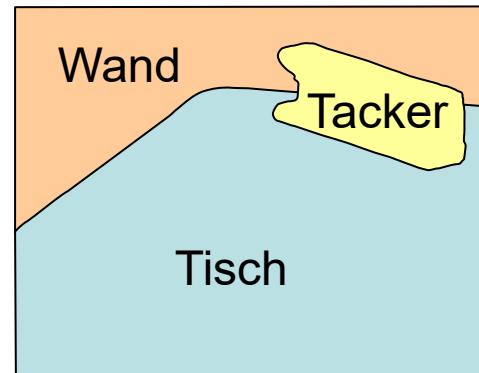
Klassische Phasen des Computersehens (3)



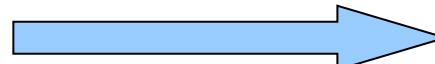
Klassifikation



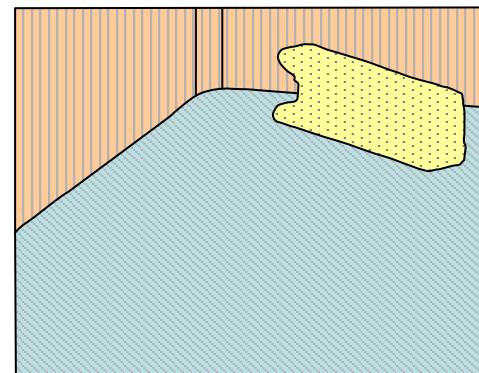
Segmente → Semantik



Segmentierung



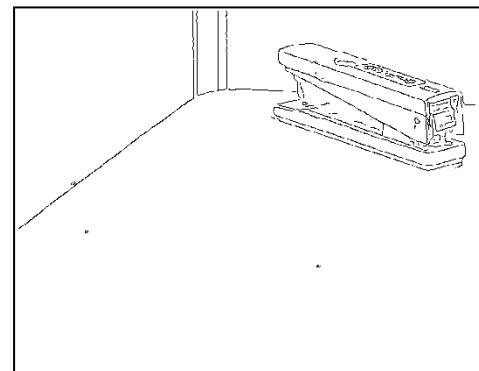
Raster → Segmente



Bildverarbeitung



Raster → Raster



Histogramme in der Bildverarbeitung

Histogramme in der Bildverarbeitung (Low Level Vision)

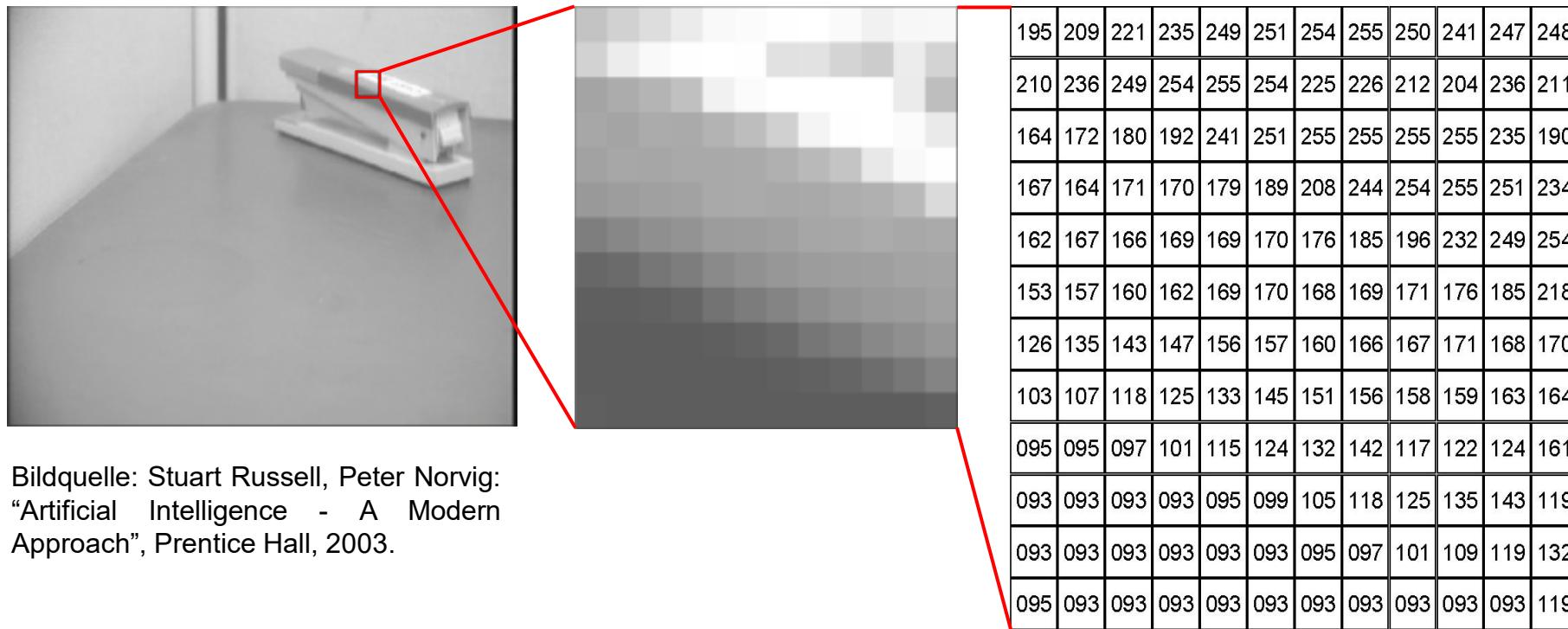
- stellen die Verteilung der Pixelwerte eines Digitalbildes dar
- erlauben die Manipulation dieser Verteilungen zur
 - Aufhellung/Abdunklung
 - Kontraststeigerung/Kontrastminderung



Histogramm-basierte
Aufhellung & Kontraststeigerung



Einkanalige Bilder

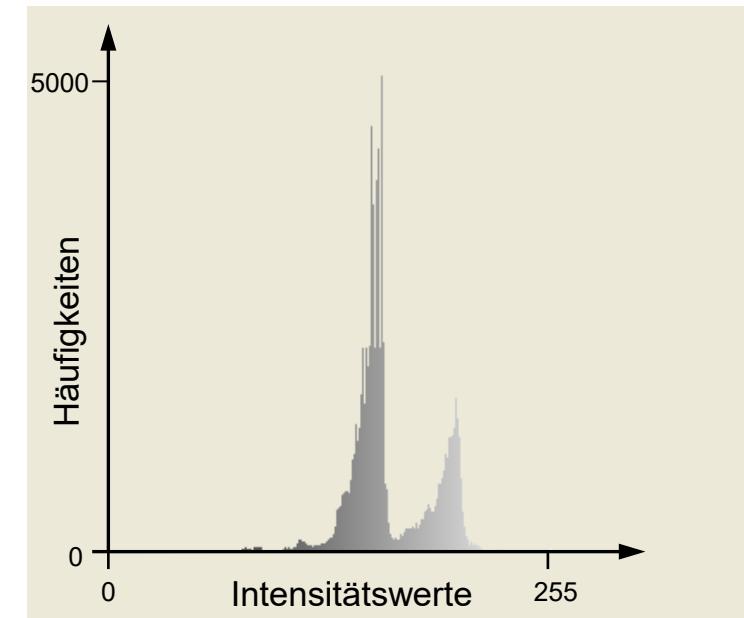
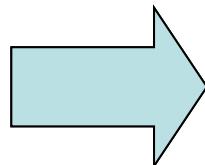


Bildquelle: Stuart Russell, Peter Norvig:
“Artificial Intelligence - A Modern
Approach”, Prentice Hall, 2003.

- **Einkanaliges (Grauwert)bild:** Matrix von Pixeln (engl. Abkürzung für Picture Elements); jedes Pixel hat einen Helligkeits- oder Intensitätswert
- Wird jedes Pixel mit 1 Byte kodiert, sind Helligkeitswerte von $I_{\min} = 0$ (schwarz) bis $I_{\max} = 255$ (weiß) darstellbar \rightsquigarrow Intensitätsspektrum $\{I_{\min}, \dots, I_{\max}\} = \{0, \dots, 255\}$

Histogramme einkanaliger Bilder (1)

- Für ein einkanaliges digitales Bild zeigt das **Intensitätshistogramm** die Helligkeitsverteilung im Bild als Balkendiagramm
- Beispiel: Histogramm des Tacker-Grauwertbildes mit $320 \times 256 = 81.920$ Pixeln



Bildquelle: Stuart Russell, Peter Norvig:
"Artificial Intelligence - A Modern
Approach", Prentice Hall, 2003.

Histogramm generiert mit *ImageToolBox*

Histogramme einkanaliger Bilder (2)

Das Intensitätshistogramm eines einkanaligen Bildes $I = [I(x,y)]$ mit Intensitätspektrum $\{0, \dots, I_{\max}\}$ ist eine diskrete Funktion $h_I(I)$, die für jeden Intensitätswert $I \in \{0, \dots, I_{\max}\}$ die Zahl n_I der Pixel im Bild I angibt, die diesen Wert aufweisen:

$$h_I(I) = n_I.$$

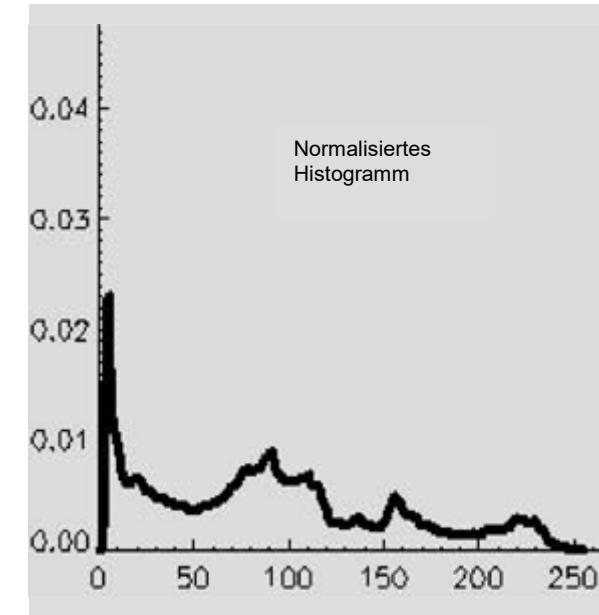
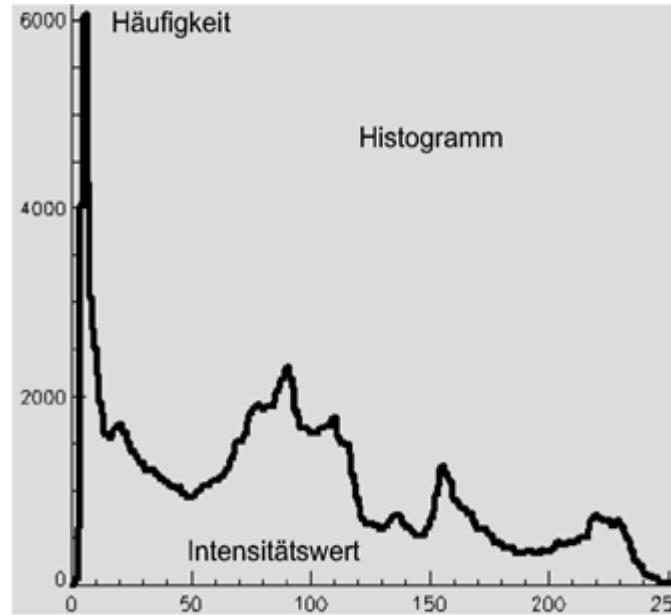
Das normalisierte Intensitätshistogramm* eines einkanaligen Bildes $I[x,y]$ skaliert die Einträge n_I für jeden Intensitätswert I durch die Gesamtzahl der Bildpixel $N = S \cdot Z$ (S Bildspalten, Z Bildzeilen):

$$p_I(I) = \frac{n_I}{S \cdot Z} .$$

* auch Histogramm der *relativen Häufigkeiten*

Histogramme einkanaliger Bilder (3)

Beispiel zu normalisierten Histogrammen:



Bildquellen: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Bemerkung: die Histogrammdarstellungen können zur Fehlinterpretation führen, dass es sich um kontinuierliche Kurven von Häufigkeitswerten handelt. Tatsächlich handelt es sich bei Histogrammen immer um Folgen von diskreten Werten.

Histogramme einkanaliger Bilder (4)

Mittelwert und mittlere quadrat. Abweichung eines einkanaligen Bildes $I = [I(x,y)]$ der Größe $N = S \times Z$ mit S Bildspalten und Z Bildzeilen lassen sich aus dem normalisierten Histogramm $p_I(I)$ berechnen:

- **Mittelwert:**

$$m_I = \frac{1}{N} \sum_{I=0}^{I_{\max}} I \cdot N \cdot p_I(I) = \sum_{I=0}^{I_{\max}} I \cdot p_I(I).$$

- **Mittlere quadratische Abweichung:**

$$q_I = \frac{1}{N} \sum_{I=0}^{I_{\max}} (I - m_I)^2 \cdot N \cdot p_I(I) = \sum_{I=0}^{I_{\max}} (I - m_I)^2 \cdot p_I(I).$$

Histogramme einkanaliger Bilder (5)

Werden die relativen Häufigkeiten $p_I(I)$ des normalisierten Intensitätshistogramms eines einkanaligen Bildes $I[x,y]$ aufsummiert, so erhält man das **kumulative Histogramm*** $s_I(I)$ eines einkanaligen Bildes $I[x,y]$.

Für jeden Intensitätswert $I \in \{0, \dots, I_{max}\}$ ergibt sich der Wert $s_I(I)$ durch Aufsummieren aller relativen Häufigkeiten $p_I(I')$ für $I' = 0, \dots, I$:



$$s_I(I) = \sum_{i=0}^I p_I(i), \quad I = 0, \dots, I_{max}.$$

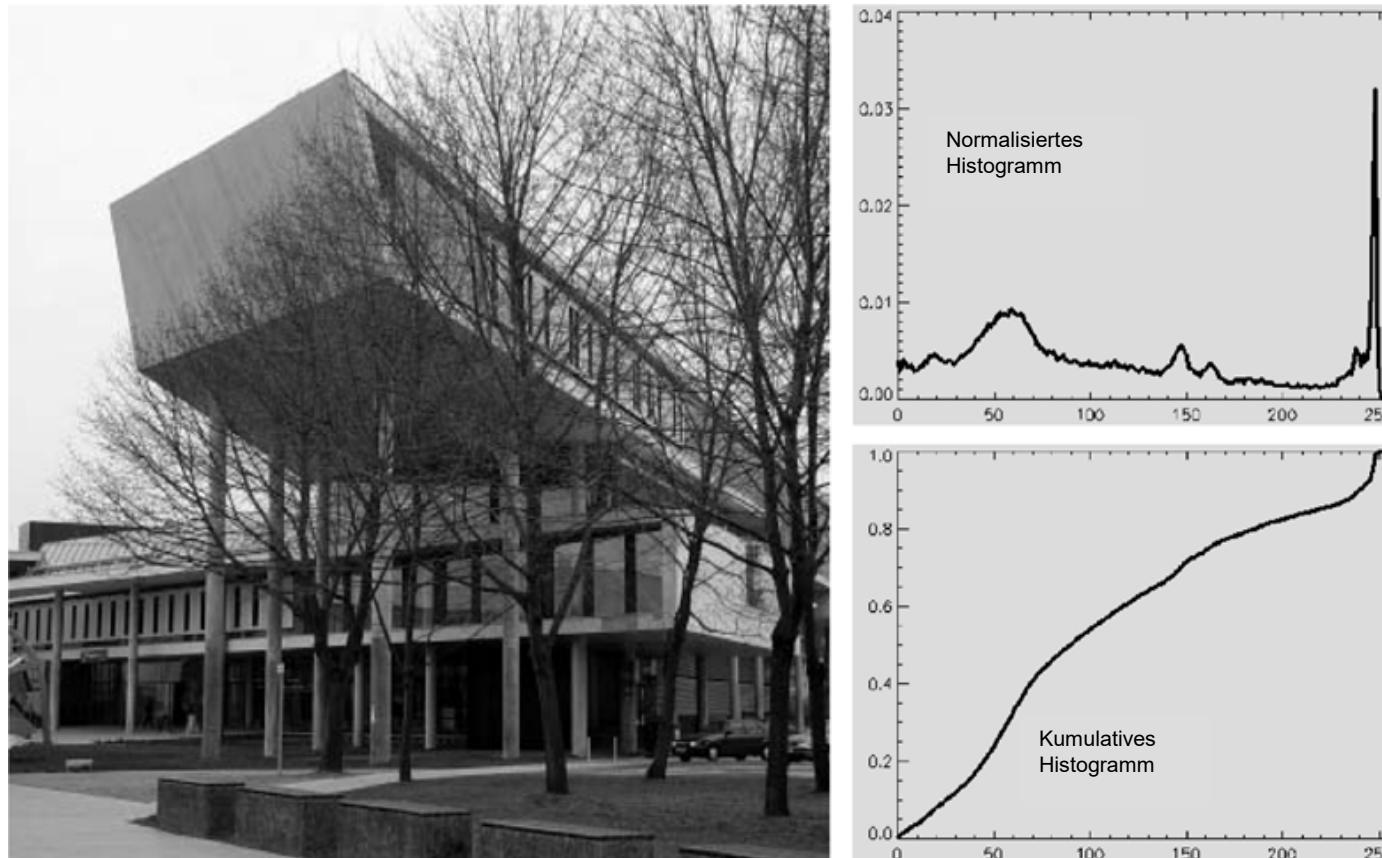
Für die relativen Summenhäufigkeiten $s_I(I)$ des **kumulative Histogramms** gilt:

$$0 \leq s_I(I) \leq 1.$$

* auch Histogramm der relativen Summenhäufigkeiten

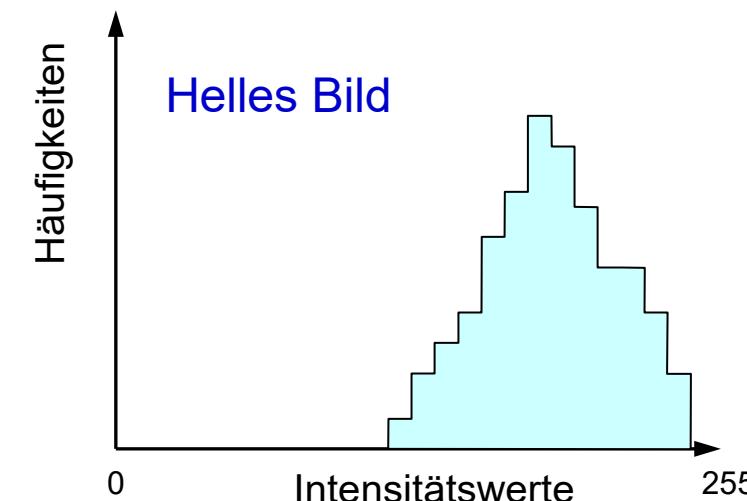
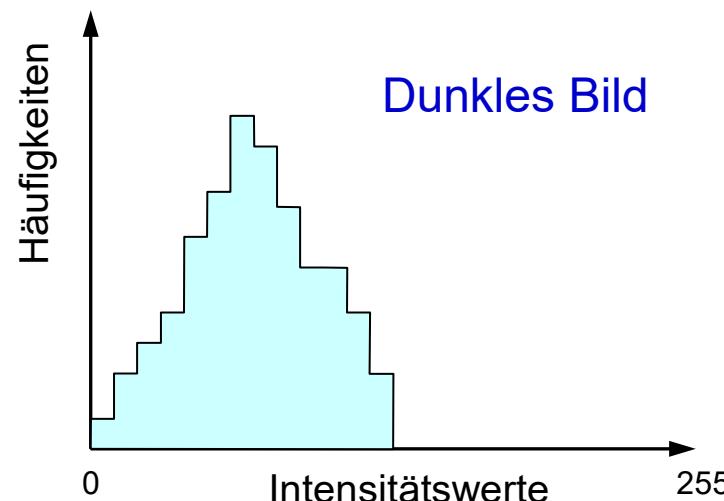
Histogramme einkanaliger Bilder (6)

Beispiel für die Ableitung des **kumulativen Histogramms $s, (I)$** aus dem normalisierten Intensitätshistogramm $p, (I)$ eines einkanaligen Bildes $I = [I(x,y)]$.



Zur Interpretation von Histogrammen (1)

- Für dunkle Bilder mit wenig Kontrast zeigen insbes. kleine Intensitätswerte I hohe Häufigkeiten $h_I(I)$ bzw. relative Häufigkeiten $p_I(I)$.
- Für helle Bilder mit wenig Kontrast zeigen insbes. die großen I -Werte hohe Häufigkeiten $h_I(I)$ bzw. relative Häufigkeiten $p_I(I)$.

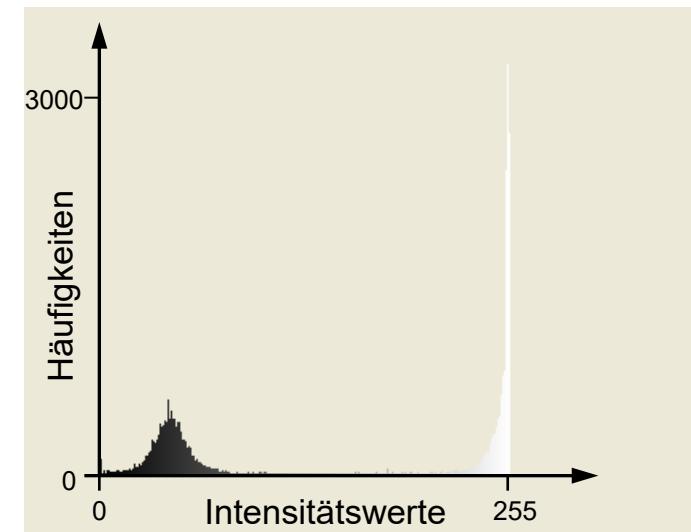
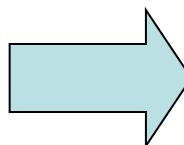


Zur Interpretation von Histogrammen (2)

- Ein Bild, das überwiegend einen dunklen und einen hellen Bereich enthält, zeigt ein sog. **bimodales Histogramm** mit zwei lokalen Maxima
- Bspiele: Scans von Dokumenten und der Anwendungsbereich Dokumentenanalyse

Bild

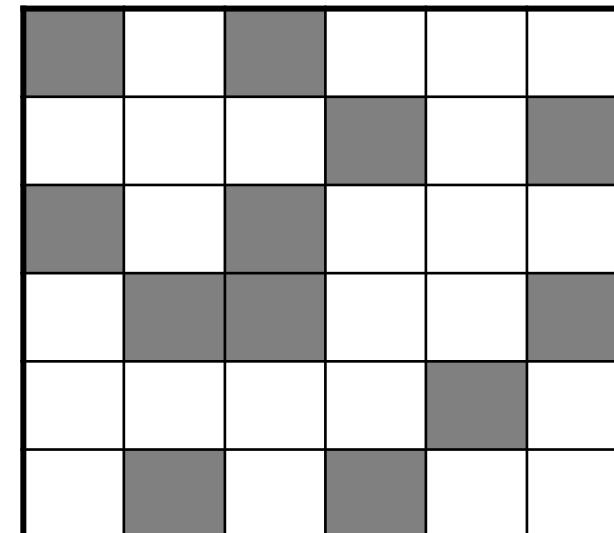
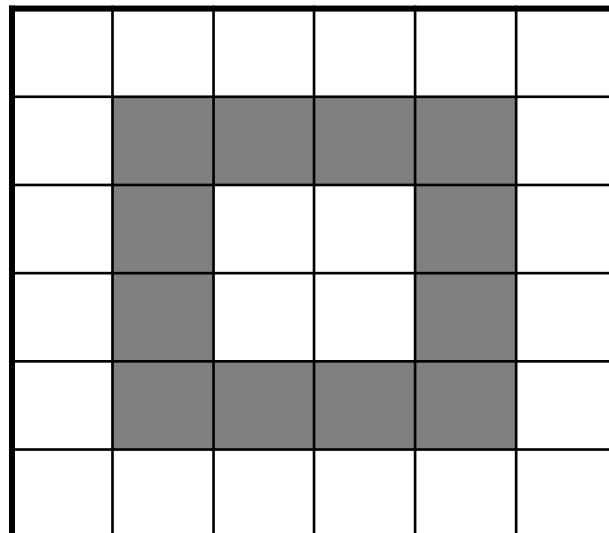
Eingescannter Text



Histogramm generiert mit *ImageToolBox*

Zur Interpretation von Histogrammen (3)

- Wichtig: Histogramme lassen keine Schlüsse auf die örtliche Anordnung der Intensitätswerte in der Bildmatrix $I = [I(x,y)]$ zu
- Beispiel: zwei Bilder mit demselben Histogramm



Histogramme mehrkanaliger Bilder

Geg.: Mehrkanalbild $I = [I(x,y,k)]$ mit S Spalten, Z Zeilen, K Kanälen und identischen Intensitätsspektren $\{0, \dots, I_{\max}\}$ für alle K Kanäle

Dessen Intensitätshistogramm ist eine diskrete Funktion $h_I(I_0, \dots, I_{K-1})$, die für jedes k -Tupel (I_0, \dots, I_{K-1}) von Intensitätswerten $I_0, \dots, I_{K-1} \in \{0, \dots, I_{\max}\}$ die Anzahl $n_{(I_0, \dots, I_{K-1})}$ der Pixel im vorliegenden Bild $I[I(x,y,k)]$ angibt, die dieses Wertetupel aufweisen:

$$h_I(I_0, \dots, I_{K-1}) = n_{(I_0, \dots, I_{K-1})}.$$

Das normalisierte Intensitätshistogramm von $I = [I(x,y,k)]$ skaliert die Einträge wieder durch die Gesamtzahl der Bildpixel $N = S \cdot Z$:

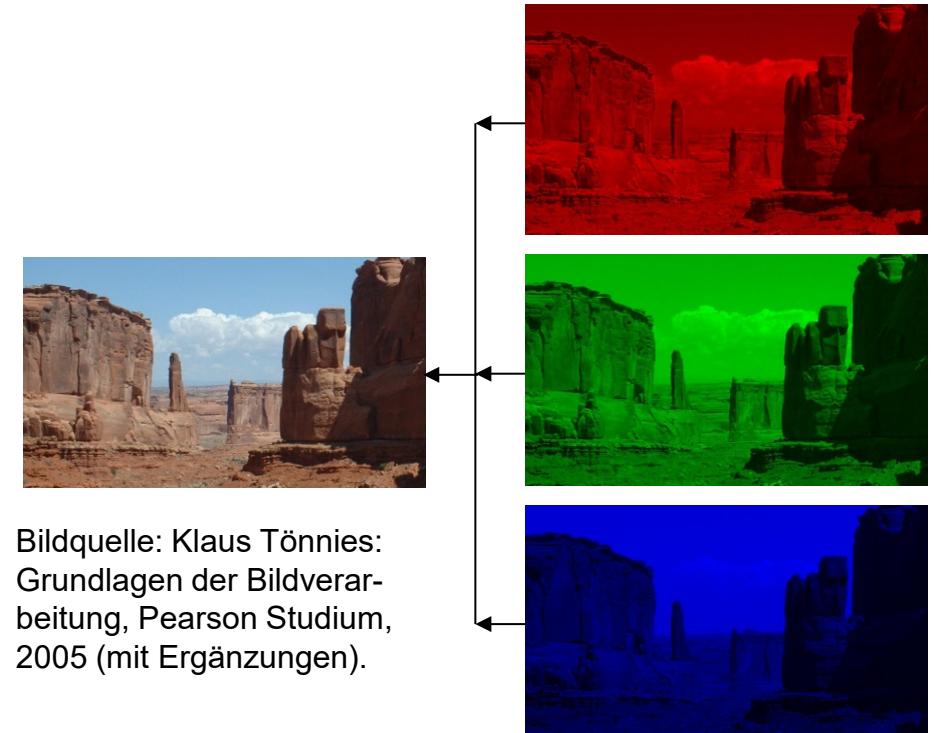
$$p_I(I_0, \dots, I_{K-1}) = \frac{n_{(I_0, \dots, I_{K-1})}}{N}.$$

Histogramme von RGB-Bildern

Geg.: RGB-Farbbild $I_{RGB} = [I(x,y,k)]$ mit
Intensitätsspektrum $\{0, \dots, I_{max}\}$ für jeden
Kanal $k \in \{R, G, B\}$.

Dessen **Intensitätshistogramm** ist eine
diskrete Funktion $h_I(I_R, I_G, I_B)$, die für jedes
Tripel (I_R, I_G, I_B) von Intensitätswerten $I_R, I_G, I_B \in$
 $\{0, \dots, I_{max}\}$ die Zahl $n_{(I_R, I_G, I_B)}$ der Pixel im Bild
angibt, die dieses Wertetripel aufweisen:

$$h_I(I_R, I_G, I_B) = n_{(I_R, I_G, I_B)}.$$

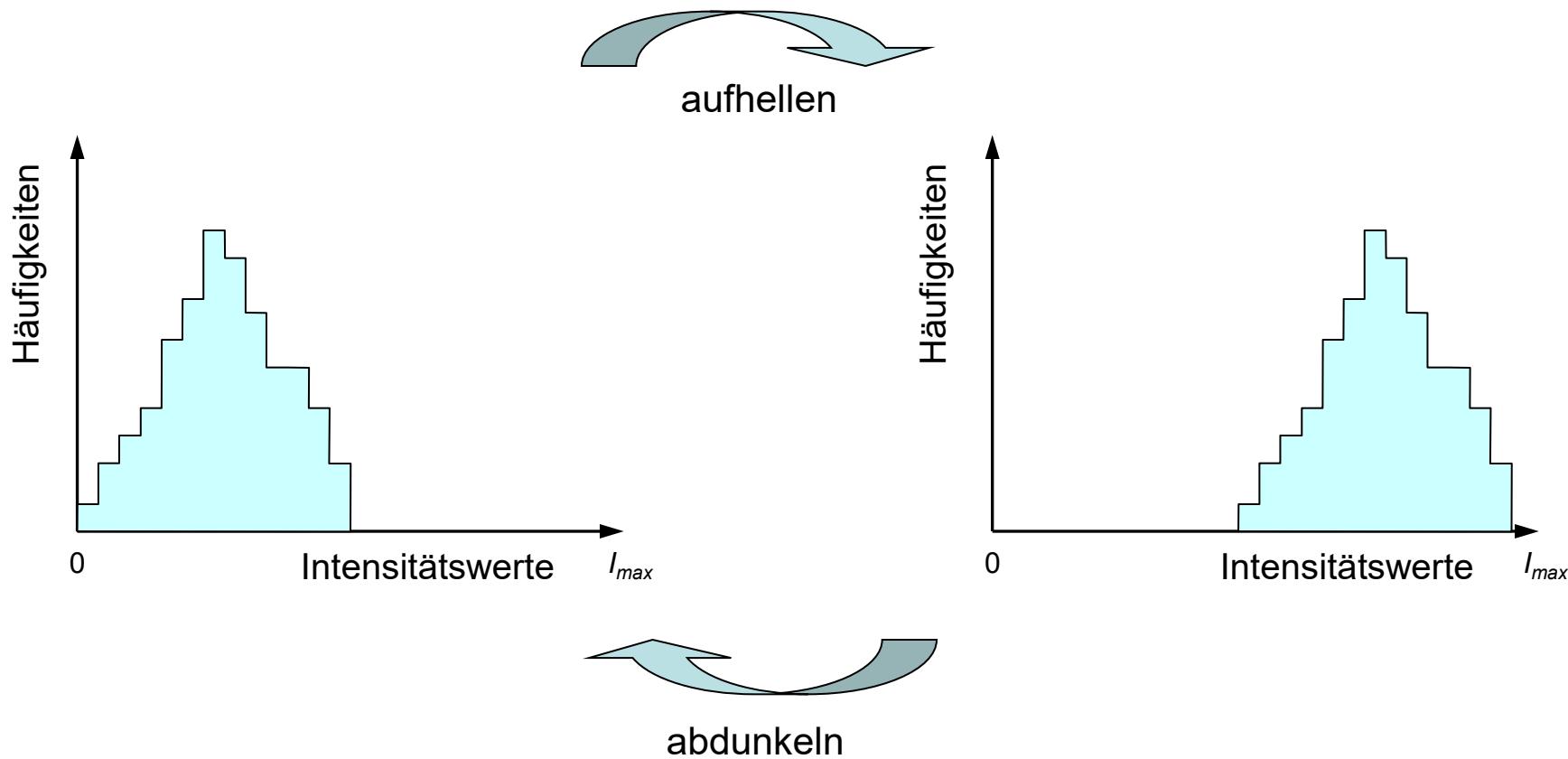


Bildquelle: Klaus Tönnies:
Grundlagen der Bildverar-
beitung, Pearson Studium,
2005 (mit Ergänzungen).

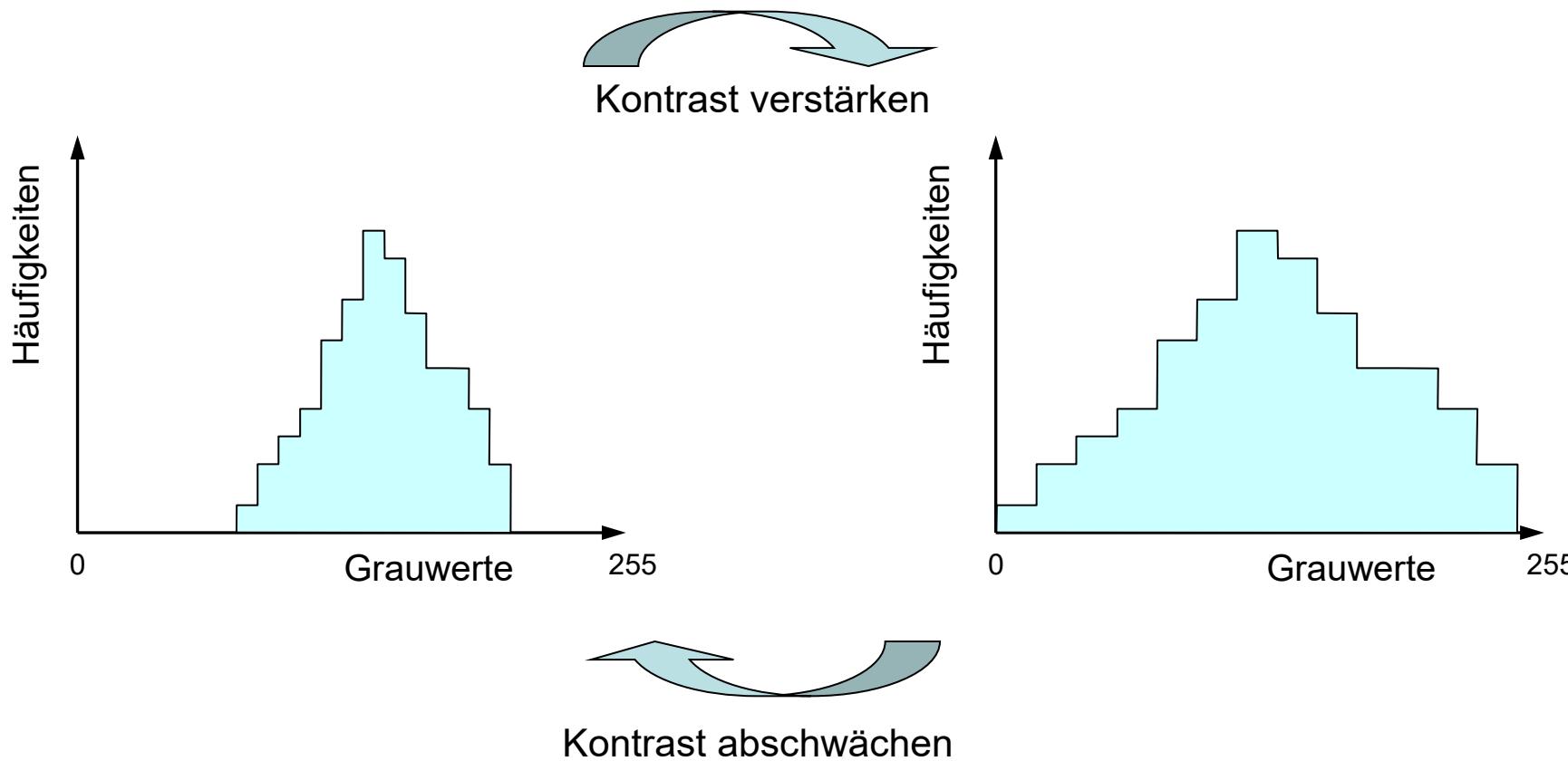
Das **normalisierte Intensitätshistogramm eines RGB-Farbbildes** $I_{RGB}[x,y]$
normalisiert die Einträge wieder durch die Gesamtzahl der Bildpixel $N = S \cdot Z$:

$$p_I(I) = \frac{n_I}{S \cdot Z}.$$

Histogrammbasierte Aufhellung/Abdunklung

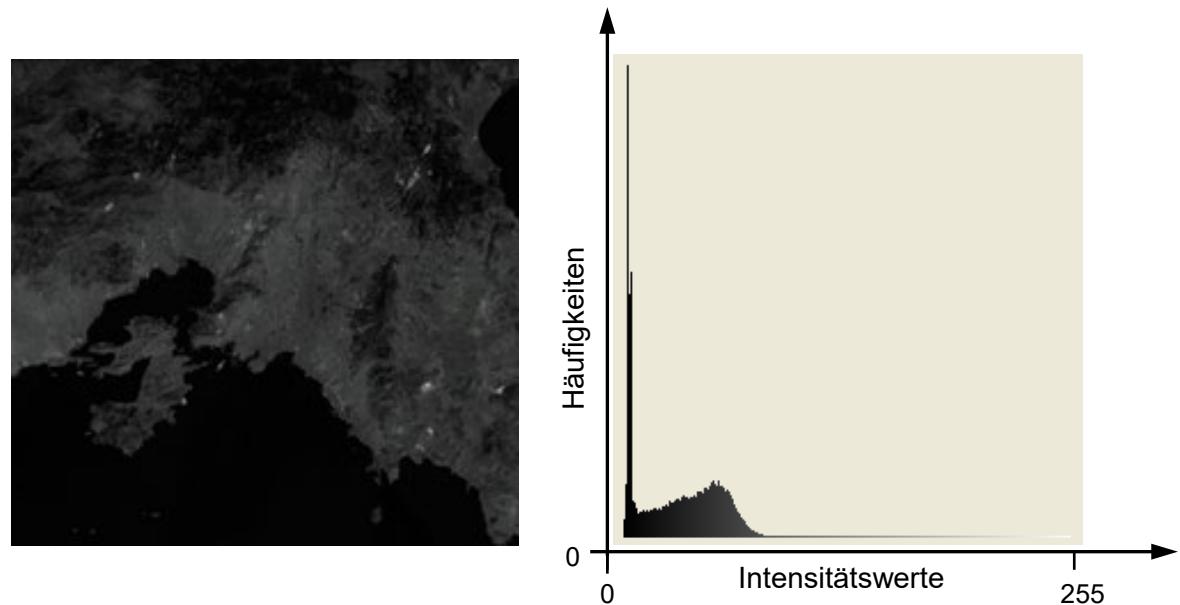


Histogrammbasierte Kontrast-verstärkung/-verminderung



Globaler Kontrast

- Def. [$I_{minGiven}, I_{maxGiven}$]
Der minimale bzw. maximale im Bild auftretende Intensitätswert werden als $I_{minGiven}$ bzw. $I_{maxGiven}$ notiert
- Bildbeispiel: $I_{minGiven} = 0$
 $I_{maxGiven} = 150$



Bildquelle links: Abteilung Fernerkundung der Univ.Trier: Kursbegleitung *Digitale Bildbearbeitung*

- Def. [globaler Kontrast C_{global}]
Der durch das geg. Intensitätsspektrum normalisierte Abstand zwischen minimalem Intensitätswert $I_{minGiven}$ und maximalem Intensitätswert $I_{maxGiven}$ im Bild wird als globaler Kontrast $C_{global} = (I_{maxGiven} - I_{minGiven}) / (I_{max} - I_{min})$ bezeichnet
- ~ Ein Bild, dessen globaler Kontrast nicht das vollständige Intensitätsspektrum von I_{min} bis I_{max} ausnutzt, zeigt einen suboptimalen Kontrast

Lineare Transferfunktionen

Aufhellung, Abdunklung, Kontrastverstärkung bzw. -minderung sind durch **lineare Transferfunktionen $T(I)$ auf den Intensitätswerten I mit $T(I) = (I + c_1) \cdot c_2$ umsetzbar.**

Je nach Belegung von c_1 und c_2 ergeben sich:

- $c_1 = 0$ und $c_2 = 1$: die identische Abbildung
- $c_1 > 0$: Aufhellung
- $c_1 < 0$: Abdunklung
- $c_2 > 1$: Kontraststeigerung
- $c_2 < 1$: Kontrastminderung



Bildquelle: Abteilung Fernerkundung der
Univ.Trier: Kursbegleitung *Digitale
Bildbearbeitung*

Bspl: Satellitenbilder nutzen den möglichen Dynamikbereich von 256 Grauwerten oft nur unvollständig, da die Sensoren so ausgelegt sind, dass sowohl sehr helle (z.B. Schnee) als auch dunkle Flächen in digitale Messwerte umsetzbar sind.

Lineare Grauwertspreizung

Die **lineare Grauwertspreizung** bzw. Intensitätsspreizung optimiert den globalen Kontrast C_{global} eines Einkanalbildes mit der folg. lineare Transferfunktion $T(I) = (I + c_1) \cdot c_2$:

- $c_1 = -I_{\text{minGiven}}$

- $c_2 = I_{\text{max}} / (I_{\text{maxGiven}} - I_{\text{minGiven}})$

$$\rightsquigarrow T(I) = [(I - I_{\text{minGiven}}) \cdot (I_{\text{max}} / (I_{\text{maxGiven}} - I_{\text{minGiven}}))]^*$$

Wenn $I_{\text{min}} = 0$. Bei $I_{\text{min}} > 0$:

$$c_1 = I_{\text{min}} - I_{\text{minGiven}}$$

$$c_2 = (I_{\text{max}} - I_{\text{min}}) / (I_{\text{maxGiven}} - I_{\text{minGiven}}).$$



mit maximalen und minimalen Intensitätswerten

I_{maxGiven} , I_{minGiven} im geg. Bild $I[x,y]$ und maximal darstellbarem Intensitätswert I_{max} **

Bildquelle: Abteilung Fernerkundung der Univ.Trier: Kursbegleitung *Digitale Bildbearbeitung*

* Äußerste Klammer steht für (kaufmännische) Rundung auf die nächstliegende ganze Zahl

** Z.B. $I_{\text{max}} = 255$ für 1-Byte-Grauwertbilder

- Die **Interpretation von Bildern** ist ein inverses und schlecht gestelltes Problem.
- **Computersehen** ist der Bereich der Informatik, der sich mit der automatisierten Interpretation von digitalen Bildern und anderen Sensordaten beschäftigt.
- **Computersehen** ist klassisch ein mehrstufiger Prozess, der sich in die Phasen der Low-Level Vision, Med-Level Vision und High-Level Vision untergliedert.

Helligkeit und Kontrast charakterisieren Bilder. Erste formale Bildcharakterisierungen werden durch **Mittelwert** und **quadrat. Abweichung** der Intensitätswerte.

Histogramme: Verteilung der Intensitätswerte. **Normierte Histogramme** sind normiert bzgl. der Bildgröße. **Kumulative Histogramme** dienen der Bildverbesserung.

Die lineare Grauwertspreizung dient zur Optimierung des globalen Kontrasts.

Contact Information

Wolfgang Koch

Prof. Dr. habil., Fellow IEEE

Fraunhofer FKIE

Chief Scientist FKIE

Head of *Sensor Data
and Information Fusion*

University of Bonn,
Institute of Informatics 4

HQ: Fraunhoferstr. 20
D-53343 Wachtberg

Branch: Zanderstraße 5
D-53177 Bonn
Germany

Phone +49 (228) 9435-373

Fax +49 (228) 9435-685

wolfgang.koch@fkie.fraunhofer.de
www.fkie.fraunhofer.de/sdf



Intelligente Sehsysteme

Fortsetzung:

1 Einführung & Histogrammbasierte Bildverarbeitung

Organisatorische & thematische Einführung

Histogrammbasierte Bildverbesserungen

Entropiemaximierende Bildverbesserungen

Henry Hölzemann

Nachtrag: Tutorienvergabe

- Die Vergabe der Tutorien findet durch das TVS bis 21. Oktober um 18 Uhr statt (siehe eCampus)
- Die Programmieraufgaben der Übungsblätter bauen erstmalig auf Python auf, nicht mehr auf der ImageToolBox und Java
- Die Abgabe der Übungen erfolgt in Gruppen von drei Studierenden
- Die Übungsblätter werden jeweils Dienstag nach der VL auf eCampus bereitgestellt
- Die Abgaben sind jeweils bis Sonntag, 12 Uhr, einzureichen
- 50% der Gesamtpunkte zur Zulassung erforderlich

Lineare Grauwertspreizung

Die **lineare Grauwertspreizung** bzw. Intensitätsspreizung optimiert den globalen Kontrast C_{global} eines Einkanalbildes mit der folg. linearen Transferfunktion
 $T(I) = (I + c_1) \cdot c_2$:

- $c_1 = -I_{\min\text{Given}}$

- $c_2 = I_{\max} / (I_{\max\text{Given}} - I_{\min\text{Given}})$

$$\rightsquigarrow T(I) = [(I - I_{\min\text{Given}}) \cdot (I_{\max} / (I_{\max\text{Given}} - I_{\min\text{Given}}))] ^*$$

mit maximalen und minimalen Intensitätswerten $I_{\max\text{Given}}$, $I_{\min\text{Given}}$ im geg. Bild $I[x,y]$ und maximal darstellbarem Intensitätswert I_{\max} **

Wenn $I_{\min} = 0$. Bei $I_{\min} > 0$:

$$c_1 = I_{\min} - I_{\min\text{Given}}$$

$$c_2 = (I_{\max} - I_{\min}) / (I_{\max\text{Given}} - I_{\min\text{Given}}).$$



Bildquelle: Abteilung Fernerkundung der Univ.Trier: Kursbegleitung *Digitale Bildbearbeitung*

* Äußerste Klammer steht für (kaufmännische) Rundung auf die nächstliegende ganze Zahl

** Z.B. $I_{\max} = 255$ für 1-Byte-Grauwertbilder

Lokaler Kontrast (1)

- Die Grauwertspreizung berücksichtigt nur den globalen Kontrast C_{global} , nicht aber die lokale Verteilung der Intensitäten im Bild
 - ↪ zwei Pixel p_1 und p_2 mit $I[p_1] = I_{\min}$ und $I[p_2] = I_{\max}$ reichen aus, um eine Grauwertspreizung zu verhindern – dennoch kann ein Großteil der Pixel ähnliche Intensitäten zeigen und das gesamte Bild daher kontrastarm sein
 - ↪ Der lokale Kontrast C_{local} wird als weitere Kenngröße definiert. Für ein Einkanalbild $I[I(x,y)]$ mit Z Zeilen und S Spalten:

$$C_{\text{local}}(I) = \frac{1}{Z \cdot S} \sum_{x=0}^{Z-1} \sum_{y=0}^{S-1} |I(x, y) - \bar{I}(x, y)|$$

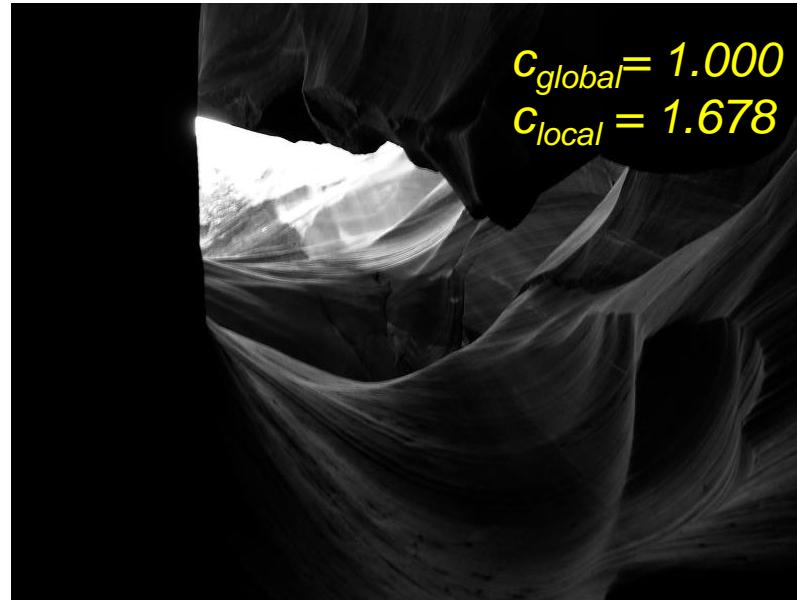
mit \bar{I} für den durchschnittlichen Wert der Intensitätsfunktion I in einer lokalen Nachbarschaft (z.B. der 8-Nachbarschaft) um (x,y)

Lokaler Kontrast (2)

Für ein Einkanalbild $I[(x,y)]$ mit Z Zeilen und S Spalten:

$$C_{global} = (I_{maxGiven} - I_{minGiven}) / (I_{max} - I_{min})$$

$$C_{local}(I) = \frac{1}{Z \cdot S} \sum_{x=0}^{Z-1} \sum_{y=0}^{S-1} |I(x, y) - \bar{I}(x, y)|.$$



Bildquelle: Klaus Tönnies

Lokaler Kontrast (3)

Nach Definition nimmt der lokale Kontrast C_{local} Bezug auf lokale Nachbarpixel.

Lokale Pixelnachbarschaften sind aber nicht in Histogrammen abgetragen!



↪ Somit scheint eine *histogrammbasierte* Verbesserung von C_{local} nicht möglich

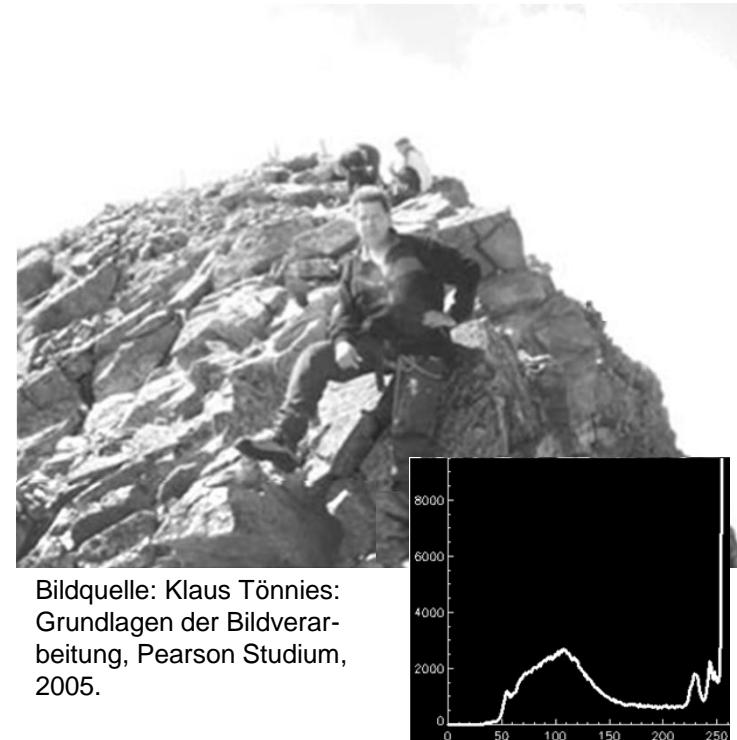
Aber:

Ausgangshypothese sei, dass Pixel mit häufig auftretenden Intensitätswerten auch häufig zueinander benachbart sind.

↪ Dann muss die Transferfunktion $T(I)$ von der Häufigkeit der Intensitätswerte abhängig sein.

Gamma-Korrektur (1)

- Annahme: die häufigsten Intensitätswerte finden sich entweder vollständig im niedrigen oder im hohen Wertebereich
 - ~ z.B. bei Unter- bzw. Überbelichtung
- Unter dieser Annahme kann die Gamma-Korrektur zu einer Verbesserung des lokalen Kontrastes führen
- Die **Gamma-Korrektur** ist eine nichtlineare Intensitäts- bzw. Grauwertspreizung auf I_{min} (i.A. = 0) bis I_{max} (mit $N_G = I_{max} + 1$):



Bildquelle: Klaus Tönnies:
Grundlagen der Bildverar-
beitung, Pearson Studium,
2005.

$$T_\gamma(I) = \left[N_G \left(\frac{I - I_{min}}{I_{max} - I_{min}} \right)^\gamma + I_{min} \right]$$

*

* Äußerste Klammer steht für (kaufmännische) Rundung auf die nächstliegende ganze Zahl.
Resultiert das Ergebnis mit N_G , so wird N_G ersetzt durch I_{max}

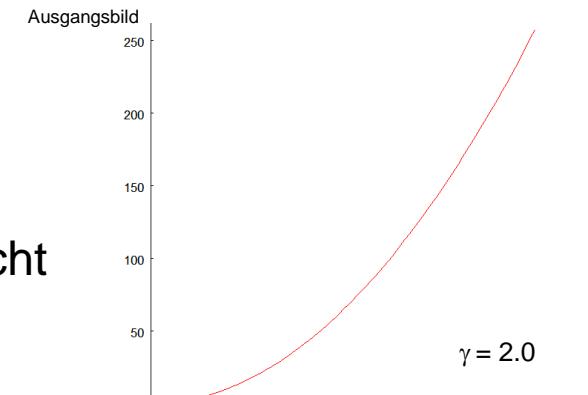
Gamma-Korrektur (2)

Anwendung der Gamma-Korrektur

$$T_{\gamma}(I) = \left[N_G \left(\frac{I - I_{min}}{I_{max} - I_{min}} \right)^{\gamma} + I_{min} \right]$$

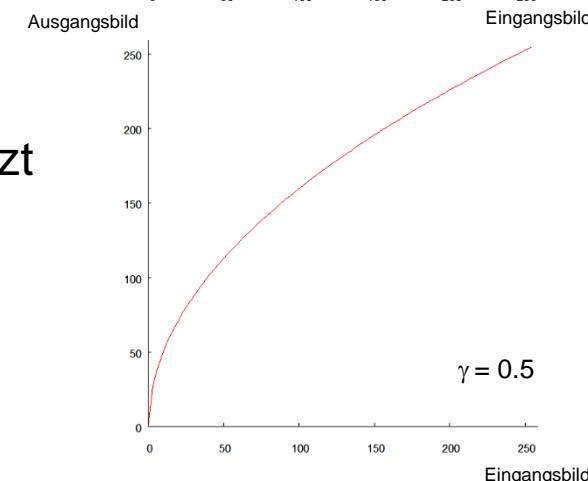
- Bei $\gamma > 1$:

- hohe Intensitätswerte werden gespreizt
- niedrige Intensitätswerte werden gestaucht
- ↗ Anwendung bei überbelichtetem Bild



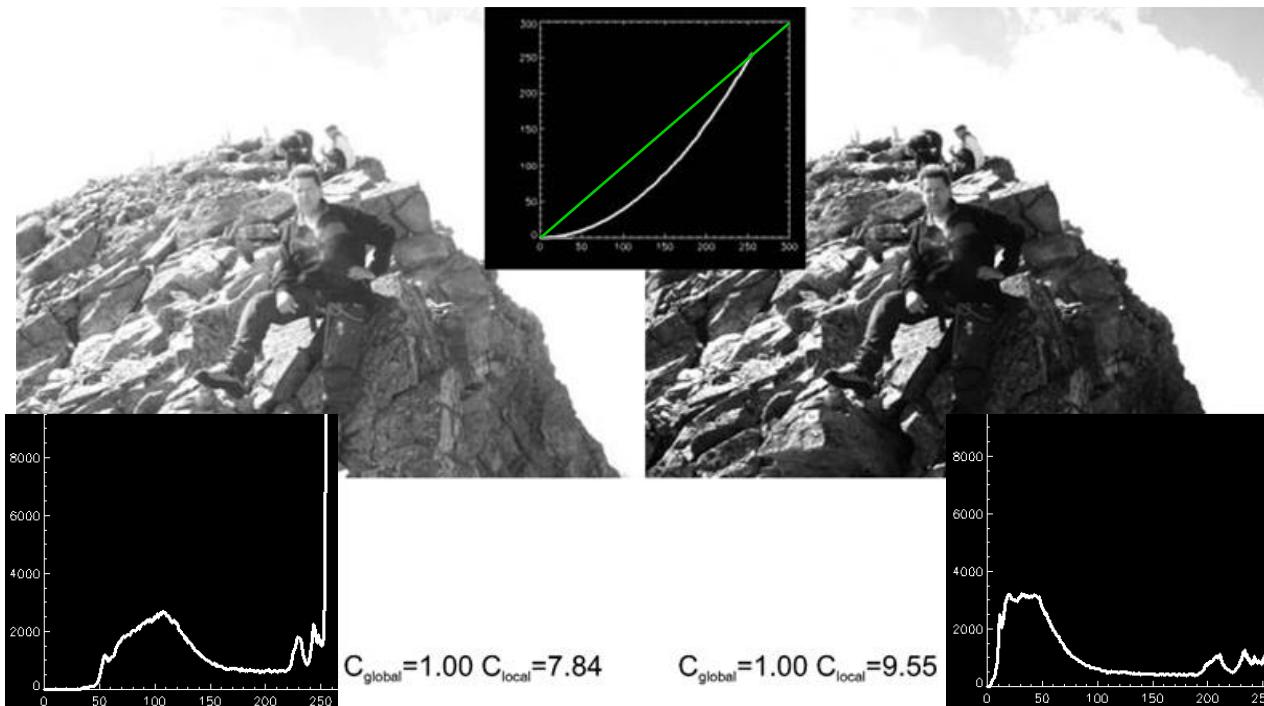
- Bei $\gamma < 1$:

- niedrige Intensitätswerte werden gespreizt
- hohe Intensitätswerte werden gestaucht
- ↗ Anwendung bei unterbelichtetem Bild



Gamma-Korrektur (3)

Bspl.: Gamma-Korrektur mit $\gamma = 2$



Bildquelle: modifiziert nach Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Entropie von Einkanalbildern (1)

Die Entropie

- ist ein Maß für den mittleren Informationsgehalt eines Bildes
- aus dem normalisierten Histogramm ableitbar

Für ein Einkanalbild $I = [I(x,y)]$

- mit Intensitätsspektrum $\{0, \dots, I_{\max}\}$ und
- normalisierten Histogramm mit relativen Häufigkeiten $p_I(I)$

ist die Entropie definiert als

$$H_I = - \sum_{I=0}^{I_{\max}} p_I(I) \cdot \log_2 p_I(I)$$

mit $0 \cdot \log_2 0 = 0$

Entropie von Einkanalbildern (2)

Beispiele:

$$\text{Geg.: } I = [I(x,y)], \quad H_I = - \sum_{I=0}^{I_{\max}} p_I(I) \cdot \log_2 p_I(I).$$

mit $0 \cdot \log_2 0 = 0$

- homogenes Bild $I = [I(x,y)] = I^*$:

$$H = - \sum_{I=I^*} p_I(I) \cdot \log_2(p_I(I)) - \sum_{I \neq I^*} p_I(I) \cdot \log_2(p_I(I)) = -1 \cdot 0 - 0 = 0$$

- Zweipegelbild $I = [I(x,y)]$ mit $p_I(I_1) = 50\%$ und $p_I(I_2) = 50\%$

$$H = - \sum_{I=I_1} p_I(I) \cdot \log_2(p_I(I)) - \sum_{I=I_2} p_I(I) \cdot \log_2(p_I(I)) = - \frac{1}{2} \cdot \log_2(\frac{1}{2}) - \frac{1}{2} \cdot \log_2(\frac{1}{2}) = 1$$

- gleich verteiltes Grauwertbild $I = [I(x,y)]$, z.B. mit $p_I(I) = 1/256$ für $I = 0, \dots, 255$

$$H = - \sum_{I=0, \dots, 255} p_I(I) \cdot \log_2(p_I(I)) = 256 \cdot (-1/256 \cdot \log_2(1/256)) = -1 \cdot -8 = 8$$



Maximierung der Entropie (1)

Die Entropie $H_I = -\sum_{I=0}^{I_{\max}} p_I(I) \cdot \log_2 p_I(I)$

- bildet neben globalem und lokalem Kontrast ein drittes Maß für den Kontrast
- eignet sich zur Spreizung von Intensitätswerten entsprechend ihrer Häufigkeit

Der Zusammenhang zwischen Entropie und Kontrast verwundert nicht, da

- die Entropie ein Maß für die Bildinformation darstellt ...
- ... und Kontrast für die Wahrnehmbarkeit von Objekten wichtig ist

Maximierung der Entropie (2)

Die Kontraststeigerung basiert jetzt also auf einer Maximierung der Entropie

$$H_I = - \sum_{I=0}^{I_{\max}} p_I(I) \cdot \log_2 p_I(I).$$

Die Entropie ist maximal, wenn die Häufigkeitseinträge für alle Intensitätswerte denselben konstanten Wert zeigen (vgl. Folie 42)



Maximierung der Entropie (3)

Die entspr. Transferfunktion ist einfacher ableitbar unter der Annahme, dass die Intensitätswerte *kontinuierlich* auf das Intervall $[0,1]$ skaliert sind

Dann folgt:

- 1) die Gesamtanzahl N aller Bildpixel ist das Integral über dem Histogramm
- 2) Für ein gleichverteiltes Histogramm ($h_I(I) = N/I_{\max}$) ist nach der Normalisierung das Integral von 0 bis zu einem beliebigen Intensitätswert I gerade I selbst:

$$\int_0^I p_I(i) di = \frac{1}{N} \cdot h_I(I) \cdot I = \frac{1}{N} \cdot \frac{N}{I_{\max}} \cdot I = I, \text{ da } I_{\max} = 1.$$

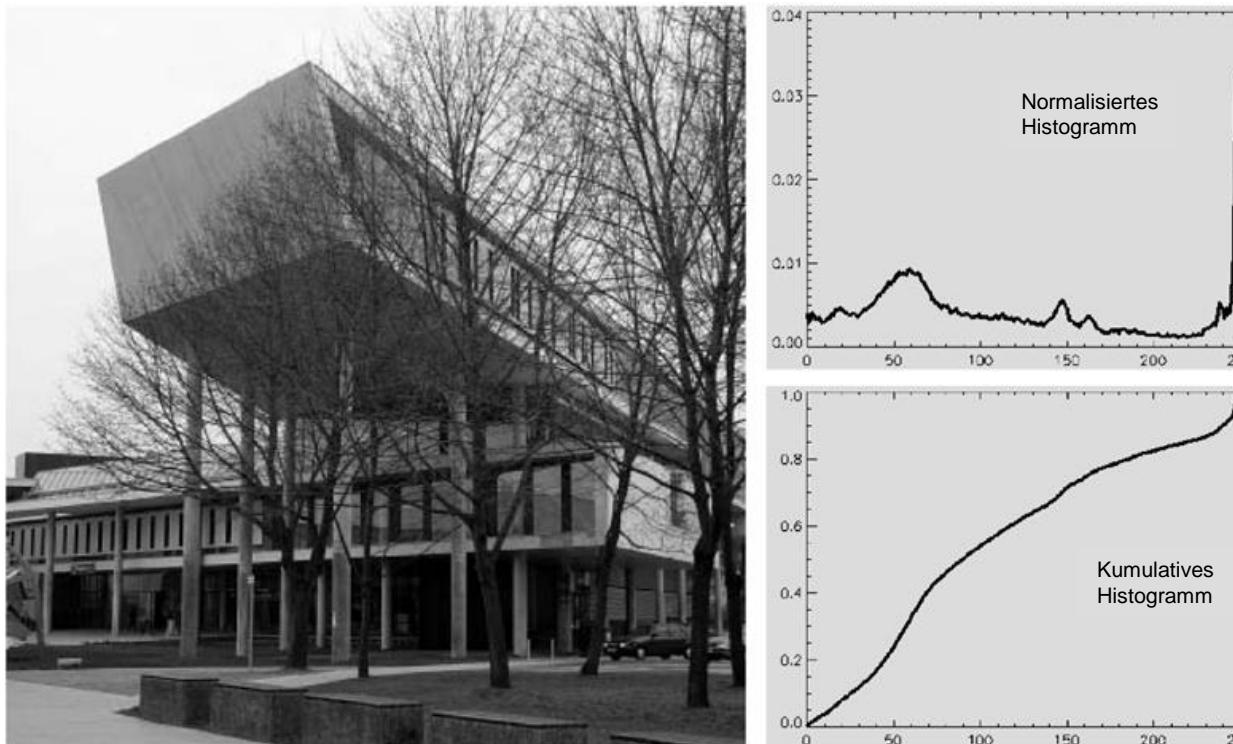
Um dies aus dem ursprünglichen Histogramm abzuleiten, muss diese Bedingung durch ebendiese Transferfunktion erzwungen werden:

$$T_H(I) = \int_0^I p_I(i) di.$$



Maximierung der Entropie (4)

Diese Transferfunktion $T_H(I) = \int_0^I p_I(i) di$ entspricht also einer kontinuierlichen Variante des kumulativen Histogramms und bewirkt eine Spreizung für besonders häufig vorkommende Intensitätswerte, weil das Integral mit steigendem I dort besonders rasch zunimmt.



Beispiel: Um die Entropie zu maximieren, wird das kumulative Histogramm (unten rechts) als Transferfunktion verwendet.

Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005 (Bild von K. Rink).

Maximierung der Entropie (5)

- Für die tatsächlichen Grauwerte
 - ist die Skalierung auf [0,1] rückgängig zu machen, indem mit der tatsächl. Anzahl N_G ($= I_{\max} + 1$) der Grauwerte multipliziert wird
 - ist die Integration wegen der Rasterung durch eine Summation zu ersetzen
 - sind die resultierenden reellen Intensitätswerte auf ganze Zahlen abzubilden:

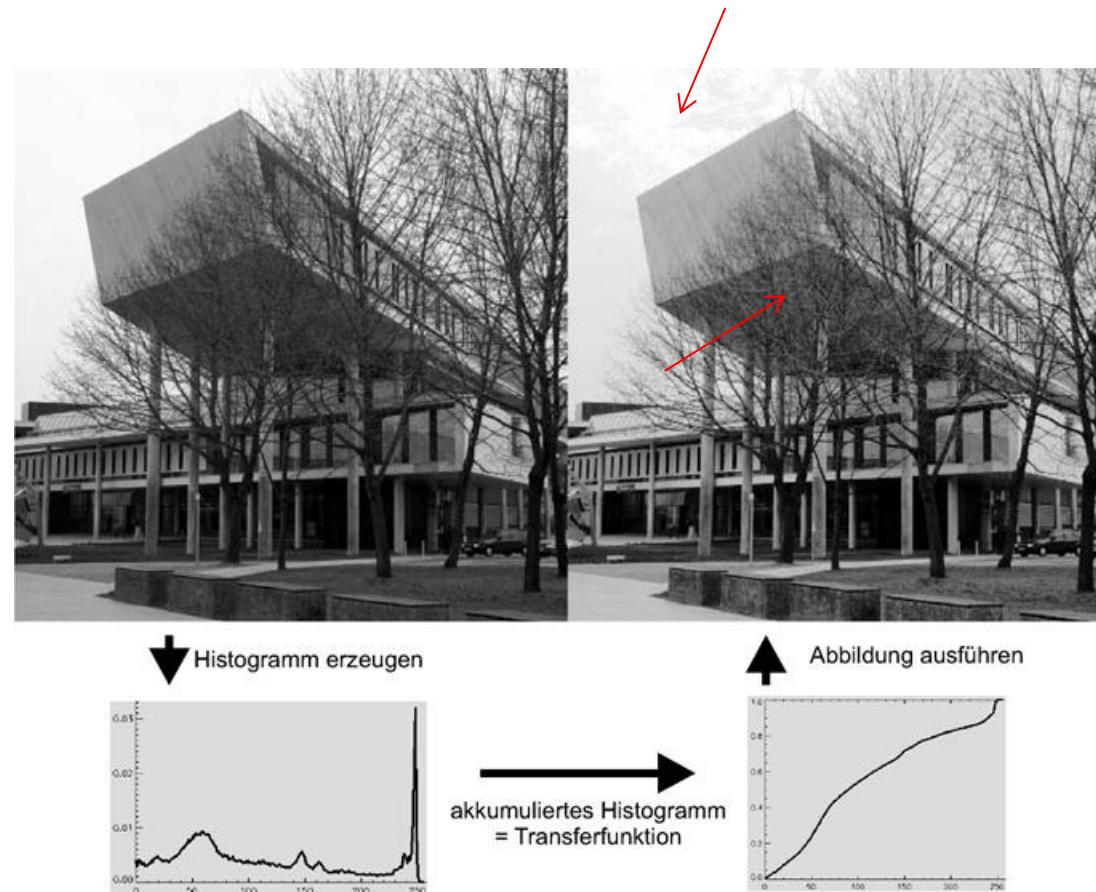
$$T_H(I) = \left[N_G \cdot \sum_{i=0}^I p_I(i) \right]^*$$

- Diese Operation heißt Histogrammlinearisation (engl. histogram equalisation)
 - das Ergebnis ist kein konstantes Histogramm, da die Summation die Integration nur approximiert
 - vielmehr kommt es zu einer Umverteilung im Histogramm: sehr häufige Intensitätswerte werden gespreizt, selten auftretende Werte werden zusammen gestaucht (z.T. in einzelne Werte überführt)

* Äußerste Klammer steht für (kaufmännische) Rundung auf die nächstliegende ganze Zahl.
Resultiert das Ergebnis mit N_G , so wird N_G ersetzt durch I_{\max}

Maximierung der Entropie (6)

Beispiel für die Histogrammlinearialisierung:



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005 (Bild von K. Rink).

Intelligente Sehsysteme

2 Bildstörungen und lineare Operatoren

Deterministische und stochastische Einflüsse,
verschiebungsinvariante, lineare Operatoren,
Konvolution, Bewegungsunschärfe, Rauschen,
lineare Glättungsfilter

Henry Hölzemann

- Deterministische Einflüsse und stochastische Einflüsse bei der Bildaufnahme
 - Photonenrauschen
 - Signal-Noise-Ratio
 - Impulsrauschen
- Lineare Filter zur Glättung
 - Mittelwertfilter
 - Gauß-Filter
 - Binomialfilter

Einflüsse bei der Bildaufnahme

Die Bildaufnahme geht i.A. mit **deterministischen und stochastischen Einflüssen** auf die abgebildete Information einher:

- **Deterministische Einflüsse** basieren auf berechenbaren und wiederholbaren Einwirkungen bei der Bildaufnahme
 - ~ Ziel ist hier, die Einwirkungen zu beschreiben, ggf. geeignet zu parametrisieren und rückgängig zu machen
- **Stochastische Einflüsse**, verändern **das Bild in statistisch beschreibbarer Weise**
 - ~ Ziel ist hier, die Parameter der statistischen Beschreibung zu ermitteln, damit der Einfluss bei der Interpretation berücksichtigt und näherungsweise rückgängig gemacht werden kann

Lineare Operatoren (1)

Viele determinist. Störungen sind sich durch lineare Operatoren beschreibbar:

- Ein Operator $O(\cdot)$ heißt linear, wenn für Funktionen f und g sowie $\lambda \in \mathbb{R}$ die Homogenität und Additivität gelten:
 - Homogenität: $O(\lambda \cdot f) = \lambda \cdot O(f)$
 - Additivität: $O(f + g) = O(f) + O(g)$
- Daraus folgt:
 - 1) ein linearer Operator ist unabhängig von den Funktionswerten
 - 2) ein linearer Operator kann nur die **gewichtete Summe aller Funktionswerte** sein kann

Lineare Operatoren (2)

Damit ist jeder lineare Operator als lineares Gleichungssystem formulierbar:

$$\mathbf{y} = \mathbf{A} \mathbf{x}^T$$

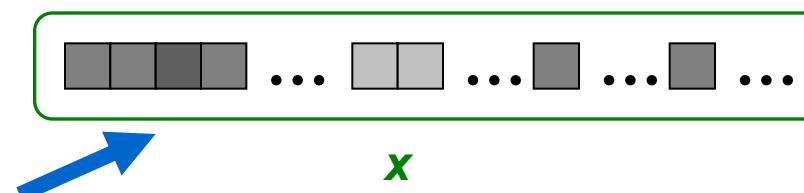
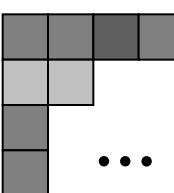
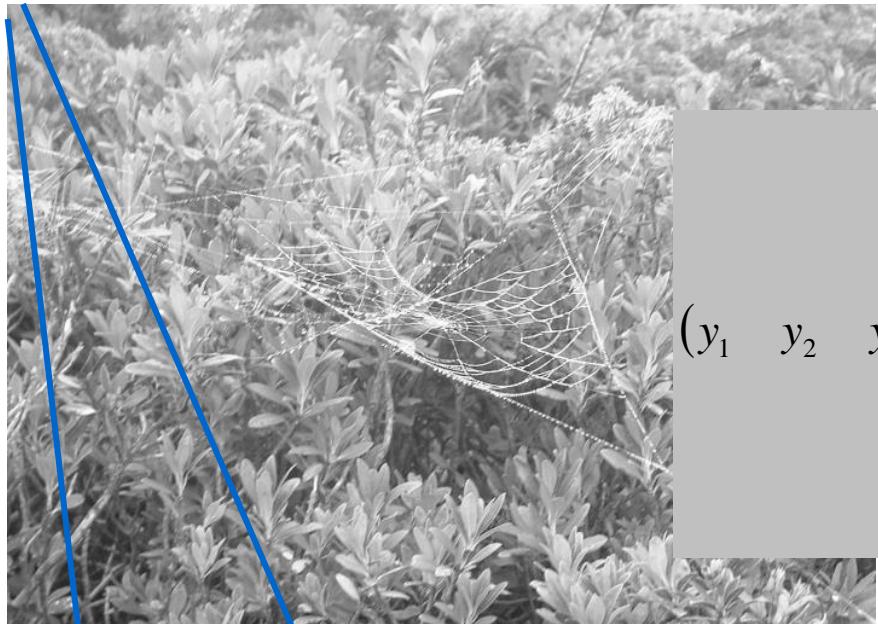
mit

- 1) Vektor \mathbf{x} der hintereinander geschriebenen Pixel des **ungestörten Bildes**
 - 2) Vektor \mathbf{y} der hintereinander geschriebenen Pixel des **gestörten Bildes**
 - 3) quadratischer Matrix \mathbf{A} , welche die **Störung** kodiert
- ↗ eine durch einen linearen Operator beschreibbare Störung ist **invertierbar**, wenn die Determinante von \mathbf{A} von Null verschieden ist

Lineare Operatoren (3)

Damit ist jeder lineare Operator als lineares Gleichungssystem formulierbar:

$$\mathbf{y} = \mathbf{A} \mathbf{x}^T.$$



$$\begin{pmatrix} y_1 & y_2 & y_3 & \dots & y_M \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \dots & a_{1,M} \\ a_{2,1} & a_{2,2} & a_{2,3} & & \\ a_{3,1} & a_{3,2} & a_{3,3} & & \\ \dots & & & & \dots \\ a_{M,1} & & & & a_{M,M} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_M \end{pmatrix}$$

Verschiebungsinvariante, lineare Operatoren (1)

- Bei linearen *verschiebungsinvarianten* Operatoren $O(\cdot)$ sind die Operatorergebnisse zusätzlich vom Ort der Operatoranwendung unabhängig:

$$O(S_{s,t}(f)) = S_{s,t}(O(f))$$

für Verschiebung $S_{s,t}$ mit $(s,t) \in \mathbb{Z}^2$. M.a.W.: verschieben wir ein Bild, ist das Ausgabebild identisch (mit Ausnahme der Verschiebung).

- Viele Einflüsse bei der Bildaufnahme sind unabhängig vom Ort der Pixel im Bild und daher durch verschiebungsinvariante, lineare Operatoren beschreibbar.

Beispiele:

- 1) gleichmäßige Bildunschärfe durch Defokussierung
- 2) gleichmäßige Bildunschärfe durch Kamerabewegung
(Bewegungsunschärfe)

Verschiebungsinvariante, lineare Operatoren (2)

Ein weiterer Vorteil:

- Ein verschiebungsinvarianter, lineare Operator ist auch dann bestimmbar, wenn die Ursache des Einflusses nicht bekannt ist
- Daher sind verschiebungsinvariante, lineare Operatoren für die Modellierung bzw. Beseitigung von Störungen sehr attraktiv

Konvolution

Verschiebungsinvariante, lineare Operatoren werden durch die **Konvolution** oder **Faltung** operationalisiert:

$$(f * g)(x,y) = \sum_{u \in D} \sum_{v \in D} f(u,v) \cdot g(x-u, y-v)$$

mit Menge D von ganzzahligen
Verschiebungsvektoren (u,v)

- Funktion f heißt **Konvolutionsfunktion** oder **Faltungsfunktion**
- Darstellung der **Konvolution** durch Operatorsymbol „*“
- Konvolutionsergebnis $(f * g)(x,y)$ ist also die gewichtete Summe der **Funktionswerte** $g(x-u, y-v)$ mit den Gewichten der nach (x,y) verschobenen **Konvolutionsfunktion** f

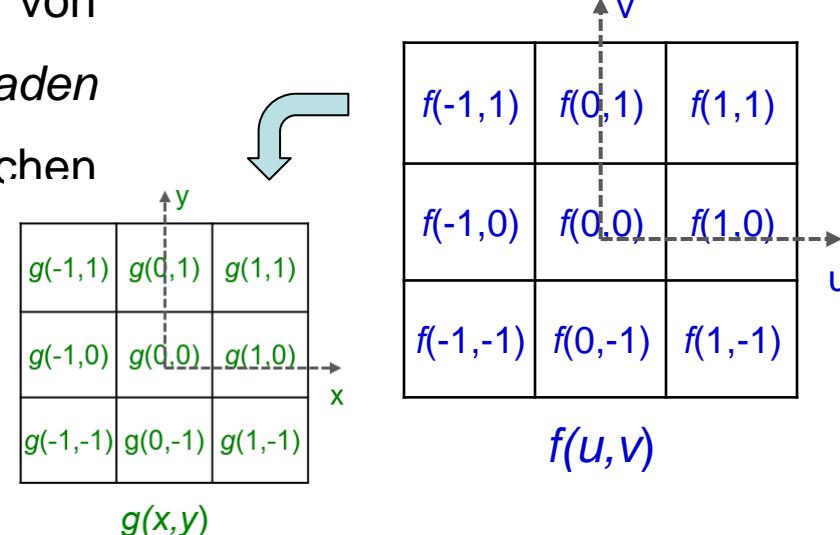
Konvolution im Computersehen

Bei der Anwendung in der Bildverarbeitung

- wird die Konvolution i. A. umgesetzt als

$$(f * g)(x,y) = \sum_u \sum_v f(u,v) \cdot g(x-u,y-v) \text{ mit } u,v = -(m-1)/2, \dots, (m-1)/2$$

- steht die **Funktion $g(x,y)$** für die **Grau- bzw. Intensitätswerte der Bildmatrix**
- definieren die Verschiebungsvektoren (u,v) einen begrenzten Argumentsbereich der **Konvolutionsfunktion f** von **quadratischer Form mit ungeraden und in Null zentrierten Wertebereichen**
(im Bsp. für $m = 3$)



Konvolutionsmaske

Die diskrete und begrenzte *Konvolutionsfunktion* $f(u,v)$ mit $u,v = -(m-1)/2, \dots, (m-1)/2$ ist also als Matrix von Konvolutions- bzw. Gewichtswerten darstellbar:

- diese werden als **Konvolutions- bzw. Faltungsmasken** oder **Konvolutions- bzw. Faltungskerne** bezeichnet
- der Parameter m wird als **Größe der Konvolutionsmaske** bezeichnet

Das Beispiel zeigt eine Konvolutionsmaske der Größe $m = 3$ mit den ganzzahligen Argumentwerten -1, 0, 1 für u und v

A 3x3 grid representing a convolution mask $f(u,v)$. The grid has three columns labeled $u = -1, 0, 1$ and three rows labeled $v = -1, 0, 1$. The values in the grid are: $f(-1,1), f(0,1), f(1,1)$ in the top row; $f(-1,0), f(0,0), f(1,0)$ in the middle row; and $f(-1,-1), f(0,-1), f(1,-1)$ in the bottom row. A dashed line connects the center cell $f(0,0)$ to the origin of both the u and v axes.

$f(-1,1)$	$f(0,1)$	$f(1,1)$
$f(-1,0)$	$f(0,0)$	$f(1,0)$
$f(-1,-1)$	$f(0,-1)$	$f(1,-1)$

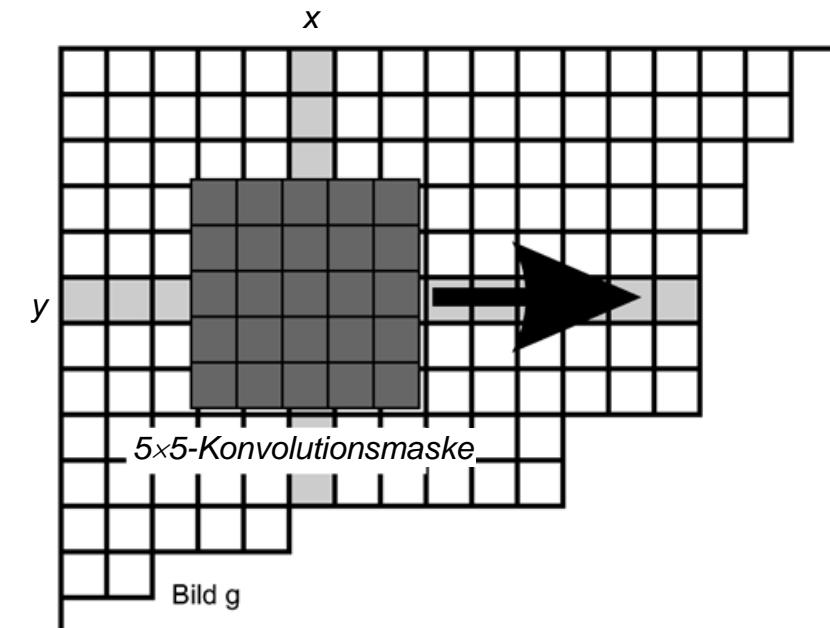
$f(u,v)$

Anwendung der Konvolution

- Die Konvolution $(f * g)(x,y)$ wird i. A. über zwei geschachtelte Schleifen (engl. *nested loops*) auf alle Pixel eines Bildes abgewendet:

```
for (int x = 0; x < height; x++) {  
    for (int y = 0; y < width; y++) { ...
```

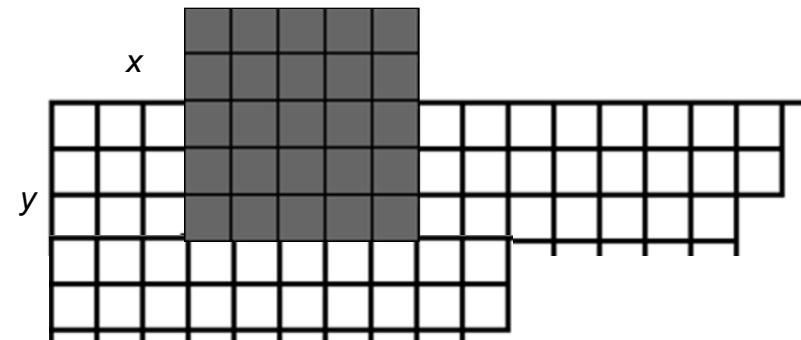
- Für jedes Pixel (x,y) mit Grauwert $g(x,y)$ wird das **Konvolutionsergebnis** $(f * g)(x,y)$ ermittelt *
- Das Bspl. zeigt eine Konvolutionsmaske der Größe $m=5$



* Die Ergebnisse $(f * g)(x,y)$ werden i.A. in separates Ausgabebild geschrieben

Randbehandlung bei der Konvolution

- Die naive Anwendung der Konvolution $(f * g)(x,y)$ führt an den Bildrändern zu undefinierten Intensitätswerten jenseits der Bildränder
- Vier gängige heuristische Lösungen:
 - 1) Speziallösungen für Randpixel, die nur die definierten Anteile der Konvolution nutzen
 - 2) Erweiterung der Bilder durch Kopieren der Randwerte jenseits der Ränder
 - 3) Erweiterung der Bilder durch Spiegelung der Randwerte jenseits der Ränder
 - 4) Vernachlässigung der Randpixel, die zu Ergebnisbildern führt, die in Höhe und Breite um die Größe der Konvolutionsmaske verkleinert sind



Nach Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Eigenschaften der Konvolution

Neben Linearität und Verschiebungsinvarianz gilt für die Konvolution:

- 1) die Kommutativität: $[f_1 * f_2](x,y) = [f_2 * f_1](x,y)$,
- 2) die Assoziativität: $f_1 * ([f_2 * f_3](x,y)) = [f_1 * f_2](f_3(x,y))$

für Operatoren f_1, f_2, f_3 .

Die Assoziativität ermöglicht effiziente Ausführung: anstelle von mehreren sukzessive auszuführenden Konvolutionen f_1, f_2, \dots, f_k auf ein Bild $I = [I(x,y)]$ kann ein Operator $f = f_1 * f_2 * \dots * f_k$ erzeugt werden und auf Bild I angewandt werden.

Beispl. für determin. Einfluss (1)

Störungen durch **Bewegungsunschärfe** entstehen, wenn die Kamera während der Belichtungszeit relativ zur beobachteten Szene bewegt wird.

Beispiel: eine ungestörte Aufnahme im Vergleich zu einer Aufnahme, bei der die Kamera in einem Winkel von 25° zur x-Achse des Kamerakoordinatensystems bewegt wurde



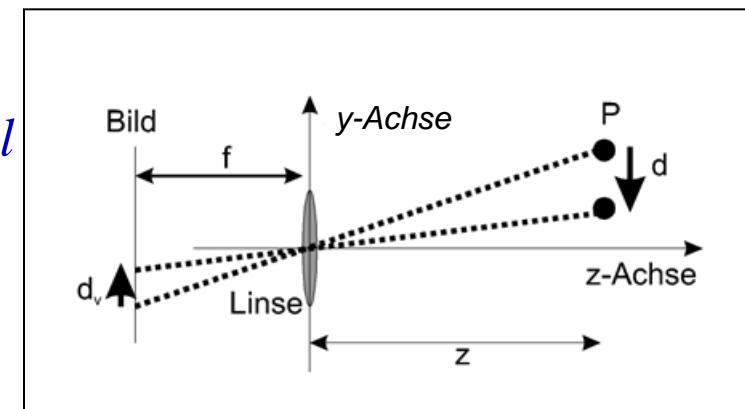
Beispl. für determin. Einfluss (2)

- Annahme: Bildfläche der digitalen Kamera wird für 30 ms belichtet, die Kamera wird in dieser Zeitspanne um $d = 2,5 \text{ cm}$ in y-Richtung verschoben
- Der Einfachheit halber sollen sich alle Szenenobjekte im Abstand $z = 1 \text{ m}$ zum Linsenzentrum befinden. Die Brennweite der Linse betrage $f = 8 \text{ mm}$. Die Pixelgröße sei $0,04 \text{ mm} \times 0,04 \text{ mm}$.
- Es folgt eine Bewegungsunschärfe durch die Verschiebung d_v (s. Abb.):

Aus $\frac{d_v}{d} = \frac{f}{z}$ folgt:

$$d_v = d \cdot \frac{f}{z} = 25 \cdot \frac{8}{10^3} = 0,2 \text{ mm} = 5 \text{ Pixel}$$

↗ dieselbe Lichtenergie, die ohne Verschiebung auf ein Pixel fiele, wird nun auf 5 Pixel verteilt



Nach Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Beispl. für determin. Einfluss (3)

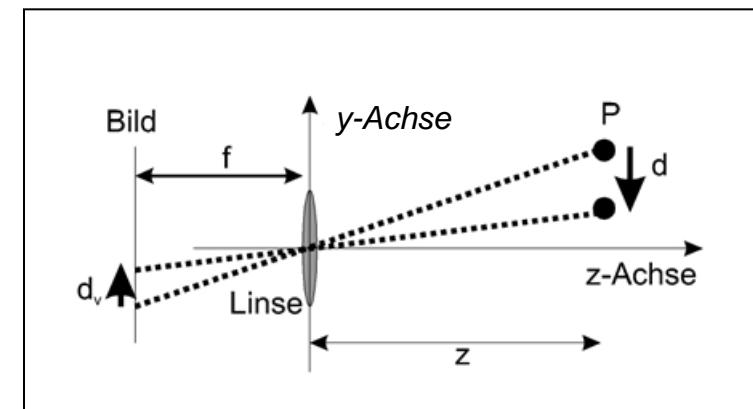
Die Verteilung der Lichtenergie eines Pixels auf 5 Pixel in y-Richtung führt zu folg. Konvolutionsfunktion als Modell dieser Störung einer Bewegungsunschärfe:

$$f(u,v) = \begin{cases} 0,2 & \text{für } -2 \leq v \leq 2 \wedge u = 0 \\ 0 & \text{sonst} \end{cases}$$

mit $m = 5$ und $u,v = -(m-1)/2, \dots, (m-1)/2$

bzw.

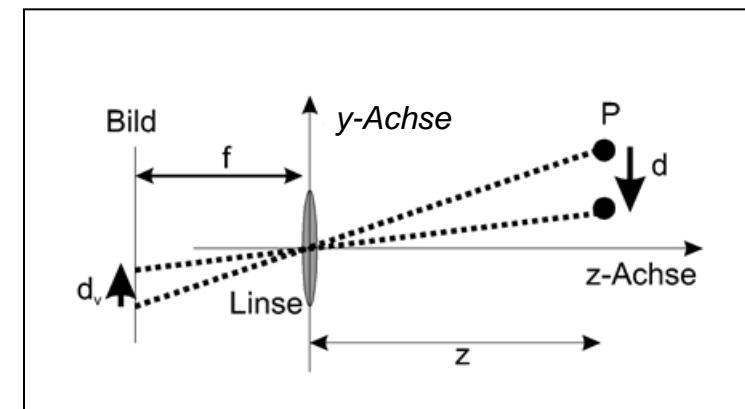
$$f(u,v) = \begin{bmatrix} 0 & 0 & 0,2 & 0 & 0 \\ 0 & 0 & 0,2 & 0 & 0 \\ 0 & 0 & 0,2 & 0 & 0 \\ 0 & 0 & 0,2 & 0 & 0 \\ 0 & 0 & 0,2 & 0 & 0 \end{bmatrix}$$



Nach Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Beispl. für determin. Einfluss (4)

- Die Funktion gilt aber auch approximativ für Szenenpunkte, die geringfügig kleinere oder größere Abstände als 1 m zur Kamera aufweisen
- Damit sind Bewegungsunschärfen für Szenarien mit kleinen Distanzunterschiede zwischen den abgebildeten Szeneobjekten modellierbar
- Zur Unterdrückung von (Bewegungs)-Unschärfen wird die **inverse Filterung** durch das **Wiener-Filter** umgesetzt, das den Fehler zwischen ungestörtem und gestörtem Bild minimiert
 - ↪ K. Tönnies: *Grundlagen der Bildverarbeitung*, 121ff, Pearson Studium, 2005. R. Gonzales, R. Woods: *Digital Image Processing*, 3rd Ed., 351 ff., Pearson Education, 2008



Nach Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Stochastische Einflüsse (1)

- Stochastische Einflüsse sind nicht deterministisch wiederholbar. Sie sind aber über Wahrscheinlichkeitsverteilungen beschreibbar
- Für Kameraaufnahmen ist insbes. das durch **Quantenrauschen** verursachte Bildrauschen von Interesse
- Prinzipiell wird eine **geradlinige Ausbreitung** von **Photonen** entspr. der **Strahlenoptik** angenommen
- **Quanteneffekte** bedingen eine teilweise Streuung von Photonen, die dann nicht an „der richtigen Stelle“ der Bildfläche auftreffen
- Dieser Effekt wird probabilistisch durch W'verteilungen beschrieben



Verrausches Bild einer Digitalkamera

Bildquelle: [http://en.wikipedia.org/
wiki/Image_noise](http://en.wikipedia.org/wiki/Image_noise) (24/10/2011)

Stochastische Einflüsse (2)

- Das **Photonenrauschen** röhrt i.W. aus Poisson-verteiltem Quantenrauschen und ist für hohe Photonenzahlen **durch eine Gauß-Verteilung approximierbar**

↗ Die Aufnahme eines Bildes $I = [I(x,y)]$ wird als **additiver Überlagerungsprozess von** einem **Signal** und **Rauschen** modelliert:

- Signal I_u = regulär verhaltende Photonen
- Rauschen η = zufallsverteilte Störung

↗ Damit gilt für die resultierende Intensität $I(x,y)$ an einer Bildkoordinate (x,y) :

$$I(x,y) = I_u(x,y) + \eta(x,y)$$

mit $\eta(x,y)$ = Gaußsche Normalverteilung mit Erwartungswert Null

Stochastische Einflüsse (4)

Die *Schätzung* des normalverteilten Rauscheinflusses erfolgt durch Bestimmung der *Varianz* σ^2 und basiert auf einer hinreichend großen Zahl von Stichproben mit bekanntem Erwartungswert:

Man nehme einen *homogenen Bildbereich* B mit Pixeln p . Deren ungestörter Intensitätswert sei $I_u(p) = I'$. Die Varianz σ^2 ist dann schätzbar durch die *korrigierte Stichprobenvarianz*

$$\sigma^2 = \frac{1}{|B|-1} \cdot \sum_{p \in B} (I(p) - I')^2$$

*Erwartungstreue
Schätzung der Varianz*

Dieser Wert wird für das gesamte Bild benutzt unter der Annahme, dass das Rauschen dieselbe Charakteristik über das gesamte Bild zeigt.

Stochastische Einflüsse (5)

Das Verhältnis zwischen Signalstärke und durchschnittl. Stärke des Rauschens wird als **Signal-Rausch-Verhältnis** (engl. **signal-to-noise ratio (SNR)**) bezeichnet.

Das SNR ist bei geg. Varianz σ^2 wie folgt bestimmbar:

Ist der Bildinhalt unbekannt,

- wird das Hintergrundsignal mit Null und das maximale Signal durch den höchsten im Bild auftretenden Intensitätswert $I_{\text{max_given}}$ geschätzt:

$$\text{SNR}_{\text{max}}(I) = I_{\text{max_given}}/\sigma$$

- Alternativ ist der mittlere Intensitätswert als Signalschätzung nutzbar:

$$\text{SNR}_{\text{avg}}(I) = (1/S \cdot Z) \sum_{x=0, \dots, S-1} \sum_{y=0, \dots, Z-1} I(x,y)/\sigma$$

Stochastische Einflüsse (6)

Das Verhältnis zwischen Signalstärke und durchschnittl. Stärke des Rauschens wird als Signal-Rausch-Verhältnis (engl. signal-to-noise ratio (SNR)) bezeichnet.

Das SNR ist bei geg. Varianz σ^2 wie folgt bestimmbar:

- Ist der Bildinhalt bekannt und damit das tatsächliche Signal (abgebildete relevante Objekte) vom tatsächl. Hintergrund unterscheidbar, kann für SNR_{OB} (**O**bject/**B**ackground) im Zähler der tatsächliche Abstand (Betrag) zw. gemittelten Signal und gemittelten Hintergrund stehen.

Stochastische Einflüsse (7)

Die dargestellten SNR-Varianten machen deutlich:

Bei der Nennung eines SNR-Wertes muss
dessen Herleitung genau spezifiziert werden.

Da die Objekt- bzw. Signalpixel nicht die hellsten Intensitäten zeigen müssen, kann z.B. SNR_{OB} erheblich von SNR_{max} oder SNR_{avg} abweichen.

Stochastische Einflüsse (7)

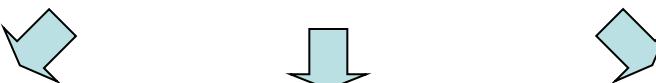
Beispiel: normalverteiltes Rauschen mit unterschiedlich hohem SNR_{\max} :



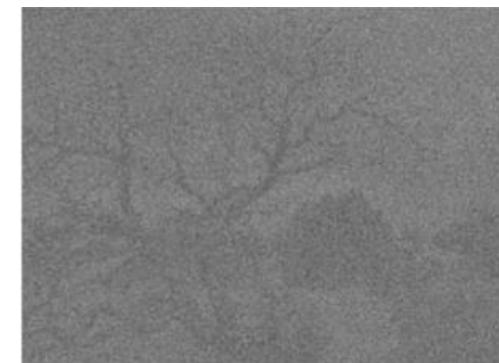
Original



$\text{SNR}_{\max} = 5$



$\text{SNR}_{\max} = 2$



$\text{SNR}_{\max} = 1$

Bilderquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Stochastische Einflüsse (8)

- Ggf. sind die zugrunde liegenden stochast. Störeinflüsse nicht durch das Gauß-verteilte Rauschmodell beschreibbar
- ↗ Ein Beispiel dafür ist das **Impulsrauschen**
(engl. *salt-and-pepper noise*)



Bildquelle: [http://en.wikipedia.org/
wiki/Image_noise](http://en.wikipedia.org/wiki/Image_noise) (26/10/2011)

Stochastische Einflüsse (9)

Impulsrauschen zeigt drei Eigenschaften:

- 1) die Veränderung der Intensitätswerte ist entweder max. positiv (hellster Grauwert 255 = „Salz“) oder max. negativ (dunkelster Grauwert 0 = „Pfeffer“)
- 2) nur wenige Pixel (z.B. 5%) sind gestört
- 3) in der lokalen Umgebung eines gestörten Pixels sind meist keine anderen Pixel gestört



Bildquelle: [http://en.wikipedia.org/
wiki/Image_noise](http://en.wikipedia.org/wiki/Image_noise) (26/10/2011)

Stochastische Einflüsse (10)

Modellierung von *Impulsrauschen*:

- Die geringe Zahl gestörter Pixel und die Lokalität der Störung macht die räumliche Korrelation schwer modellierbar, da diese eine summarische Betrachtung der Intensitätswerte in Abhängigkeit von den Abständen zwischen den Pixeln verlangt
- Dennoch werden wir bald ein Filter zur Unterdrückung des Impulsrauschen lernen



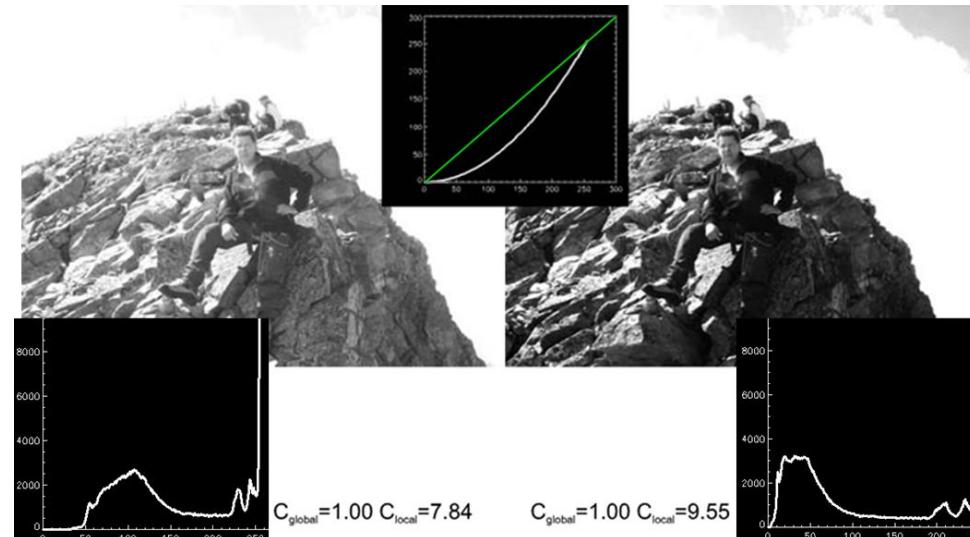
Bildquelle: [http://en.wikipedia.org/
wiki/Image_noise](http://en.wikipedia.org/wiki/Image_noise) (26/10/2011)

Histogrammbasierte Verfahren vs. Lineare Filter

Die erste Vorlesung zeigte histogrammbasierte Verfahren zur Bildverbesserung. Diese berücksichtigen nicht die räumliche Verteilung der Intensitätswerte im Bild.

So sind für eine Verbesserung des lokalen Kontrastes C_{local} eigentlich die lokalen Pixelnachbarschaften zu berücksichtigen.

Nur indirekt konnte C_{local} mit Hilfe der Gamma-Korrektur oder der adaptiven Histogrammlinearisierung verbessert werden.



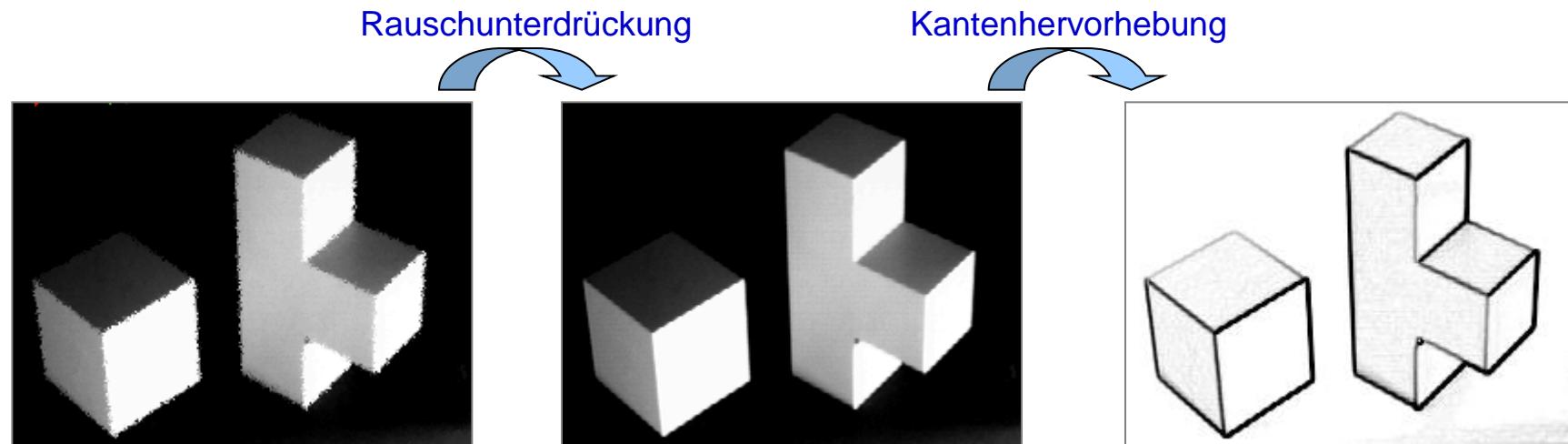
Bilderquelle: K. Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005

Die Konvolution kann hingegen Pixel in Abhängigkeit von ihrer lokalen Pixelnachbarschaft verändern. Damit sind lokale Bildverbesserungen möglich.

Lineare Filter (1)

Zwei Arten der Bildverbesserung sind durch Konvolution durchführbar:

- 1) durch **Rauschunterdrückung** (auch als **Glättung** bezeichnet) kann das Signal gegenüber stochastischem Rauschen verstärkt werden,
- 2) durch **Hervorhebung von Kantenpixeln**, die Grenzen zwischen Objektoberflächen abbilden, werden abgebildete Objekte und ihre Abgrenzungen besser erkennbar und durch anschließende Prozesse leichter erkennbar.



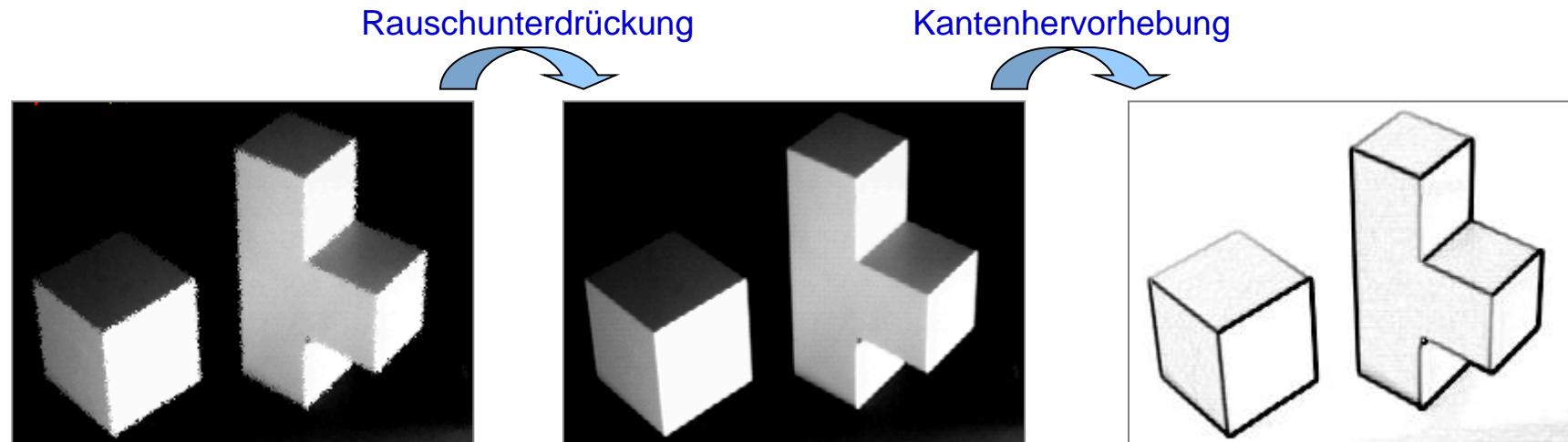
Lineare Filter (2)

Methoden der **Rauschunterdrückung** bzw. **Glättung** müssen die lokalen Pixelnachbarschaften berücksichtigen, weil Rauschen nur als statistische Eigenschaft einer Menge von Pixel charakterisierbar ist.

diese
Vorlesung

Methoden der **Kantenhervorhebung** benötigen die Information aus lokalen Pixelnachbarschaften, weil Kanten sich nur durch den Vergleich zwischen den Intensitäten benachbarter Pixel herausbilden.

nächste
Vorlesung



Lineare Filter zur Glättung (1)

Das erste einfache Glättungsfilter basiert auf dem additiven Überlagerungsmodell (vgl. Folie 20):

ein verrauschtes Bild $\mathbf{I} = [I(x,y)]$ zeigt resultierende Intensitäten $I(x,y)$ nach

$$I(x,y) = I_u(x,y) + \eta(x,y)$$

mit:

Signal I_u = ungestörtes Signal

Rauschen η = zufallsverteilte Störung

Lineare Filter zur Glättung (2)

- Bei diesem additiven Überlagerungsmodell

$$I(x,y) = I_u(x,y) + \eta(x,y)$$

wurde die W'keit einer Abweichung des gemessenen Intensitätswertes vom ungestörten Intensitätswert $I_u(x,y)$ durch eine **Gaußsche Normalverteilung mit Erwartungswert Null** modelliert: $E(\eta(x,y)) = 0$

- ↗ Nun gilt: mit **steigender Stichprobenzahl** ergibt sich eine zunehmend bessere Schätzung des Erwartungswertes für das Rauschen η

Lineare Filter zur Glättung (3)

Ausgangspunkt also: Mit steigender Stichprobenzahl ergibt sich eine zunehmend bessere Schätzung des Erwartungswertes für das Rauschen η

Dies ist zur Rauschunterdrückung wie folgt nutzbar:

- der Erwartungswert der ungestört. Intensitätsfunktion $I_u(x,y)$ ist konstant:

$$E(I_u(x,y)) = I_u(x,y)$$

*Erwartungswert der
Störung ist Null*

↗ damit gilt:

$$E(I(x,y)) = E(I_u(x,y)) + E(\eta(x,y)) = E(I_u(x,y)) = I_u(x,y)$$

↗ das Problem ist gelöst, sobald eine gute Schätzung für $E(I_u(x,y))$ vorliegt!

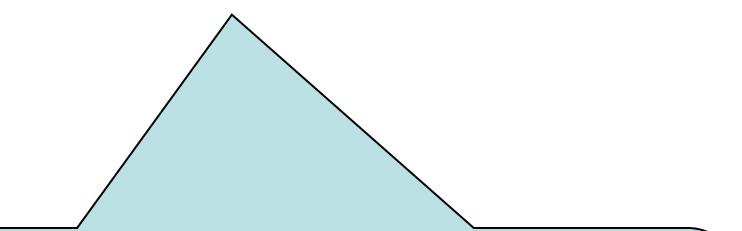
Lineare Filter zur Glättung (4)

Eine gute Schätzung für $E(I(x,y))$ wäre wie folgt ableitbar:

1) generiere eine Folge von k Bildern I_0, \dots, I_{k-1} derselben Szene

2) schätze $E(I(x,y))$ über das aus der Folge gemittelte Bild

↪ Also wäre das ungestörte Bild ableitbar durch eine Konvolution entlang der Zeitachse mit dem Konvolutionskern $f = (1/k, \dots, 1/k)$



Die Summe der Gewichtselemente des Konvolutionskerns ergibt 1. Diese Eigenschaft gilt für alle Konvolutionskerne zur Rauschunterdrückung!!!

Der Grund: es geht ja um die Schätzung von $E(I(x,y))$. Jede von 1 verschiedene Gewichtssumme würde eine Skalierung der Intensitäten der Bildfunktion ergeben.

Lineare Filter zur Glättung (5)

Die Mittelung über eine Folge von Einzelbildern ist i.A. nicht praktikabel.

Daher wird von $E(I(x,y))$ wie folgt geschätzt:

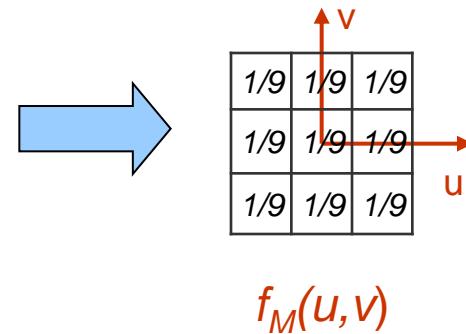
- 1) wir nehmen an, dass die Werte der ungestörten Bildfunktion $I_u(x,y)$ in einer lokalen Nachbarschaft jedes Pixels konstant ist
- 2) statt über eine Bildfolge wird $E(I(x,y))$ nun durch Mittelung über die Intensitätswerte innerhalb der lokalen Nachbarschaft geschätzt

Mittelwertfilter (1)

Der einfachste Konvolutionskern hierfür definiert das sog. *Mittelwertfilter* (engl. *boxcar filter* oder *shifted mean filter*).

Für die 3×3 -Nachbarschaft hat er folg. Form:

$$f_{M,3}(u,v) = \{(-1, 1) \rightarrow 1/9, (0, 1) \rightarrow 1/9, (1, 1) \rightarrow 1/9, \\ (-1, 0) \rightarrow 1/9, (0, 0) \rightarrow 1/9, (1, 0) \rightarrow 1/9, \\ (-1, -1) \rightarrow 1/9, (0, -1) \rightarrow 1/9, (1, -1) \rightarrow 1/9 \}$$

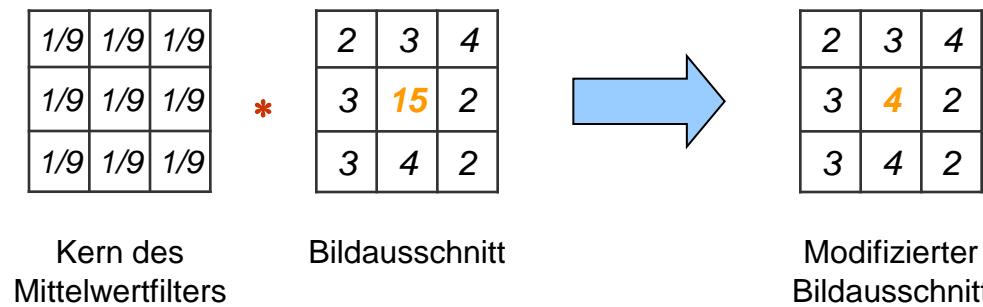


Mittelwertfilter (2)

Die Anwendung des **3×3-Mittelwertfilters** bzw. des **Mittelwertfilters der Größe 3** auf ein Pixel und seine 8-Nachbarschaft über die Konvolution nach

$$(f_{M,3} * g)(x,y) = \sum_u \sum_v f_{M,3}(u,v) \cdot g(x-u,y-v) \text{ mit } u,v = -1, 0, 1$$

am Beispiel:



Bemerkung: aus Effizienzgründen erfolgt die Implementierung i.A. so, dass alle neun Gewichte auf Eins gesetzt werden und die Summe durch 9 dividiert werden:

$$f(u,v) = 1/9 \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Mittelwertfilter (3)

Die Zuverlässigkeit der Schätzung von $E(I(x,y))$ steigt mit der Anzahl der Pixel, über die gemittelt wird.

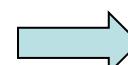
Mit einem **Mittelwertfilter der Größe 5** wird z.B. fast über die dreifache Zahl von Pixel gegenüber dem Mittelwertfilter der Größe 3 gemittelt:

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



0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04

Kern des 5×5-Mittelwertfilters



...

Kern des 3×3-Mittelwertfilters

Mittelwertfilter (4)

Grenzen des Mittelwertfilters:

Mit zunehmender Größe des Mittelwertfilters wird die Annahme unrealistischer, dass alle Pixel innerhalb der Nachbarschaft denselben ungestörten Intensitätswert $I_u(x,y)$ haben.

Das ist insbes. bei Kanten der Fall. So werden Artefakte bei größer werdenden Filtern immer stärker und die resultierenden Bilder wirken immer unschärfer.

Mittelwertfilter (5)

- Besonders kritisch wird die durch **Mittelwertfilter** resultierende Unschärfe, wenn **Impulsrauschen** bzw. ***Salt-and-Pepper-Noise*** vorliegt
- Der Erwartungswert des Impulsrauschen ist zwar ebenfalls 0, aber der überwiegende Anteil aller Pixel (z.B. 95%) ist vom Impulsrauschen nicht betroffen
 - ↗ Für die Filterung müsste die Umgebungsnachbarschaft sehr groß sein, um genügend positive und negative Impulse zu erfassen, damit ihr Einfluss sich in der Summation aufhebt
 - ↗ Damit ist das **Rauschen nicht zu entfernen**, ohne dass Kanten im Bild **unscharf werden**. Insofern kann praktikabel nur eine Abschwächung, nicht aber eine Unterdrückung des Impulsrauschens erzielt werden (s. Abb.)

Mittelwertfilter (6)

Beispiel für Anwendung von Mittelwertfiltern auf Impulsrauschen:



Bilderquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Gauß-Filter (1)

Die Anwendung von Mittelwertfiltern führt auch bei Photonenauschen zu einer Glättung von Bildkanten.

Das Photonenauschen wurde durch eine Gauß-Verteilung modelliert. Eine optimale Unterdrückung des Photonenauschens ist demnach durch einen nach einer Gauß-Funktion gewichteten Filter möglich.

Für eine Dimension:

$$f_{G,\sigma}(u) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(u^2)/2\sigma^2}.$$

Mit Standardabweichung σ der Gauß-Funktion. Je größer σ , desto stärker die Glättung des Gauß-Filters.

Die zweidimensionale Gauß-Funktion ergibt sich als Produkt zweier eindimensionaler Gauß-Funktionen*:

$$f_{G,\sigma}(u, v) = \frac{1}{2\pi\sigma^2} e^{-(u^2+v^2)/2\sigma^2}.$$

* siehe D. A. Forsyth, J. Ponce: *Computer Vision – A Modern Approach*. Pearson, 2003, Seite 173.

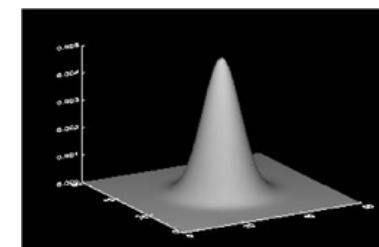
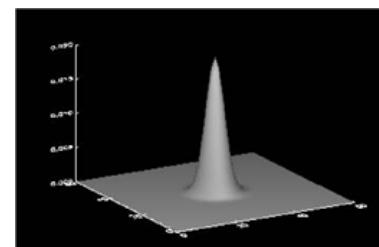
Gauß-Filter (2)

Separierbarkeit des zweidimensionalen Gauß-Filters:

- Die zweidimensionale Gauß-Funktion wurde als Produkt zweier eindimensionaler Gauß-Funktionen erzeugt.
- Analog ist die Konvolution mit dem zweidimensionalen Gauß-Filter durch zwei aufeinander folgende Konvolutionen mit eindimensionalen Gauß-Filtern in Zeilen- und Spaltenrichtung durchführbar.

Gauß-Filter (3)

Beispiel für Anwendung eines Gauß-Filters mit unterschiedl. σ :



Bilderquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Gauß-Filter (4)

Die Größe des Konvolutionskerns eines Gauß-Filters

- ist so zu wählen, dass die Werte der Gauß-Funktion hinreichend gut approximiert werden*
- ist von der Standardabweichung σ der Gauß-Funktion abhängig

Eine angemessene Filtergröße m ist z.B. $m = 2 \cdot \lceil 3\sigma \rceil + 1$:

- der Funktionswert am Rand des Filterkerns beträgt dann noch 1% des Maximums der Gauß-Funktion
- das Maximum in $f_{G,\sigma}(0,0)$ ist $(2\pi\sigma^2)^{-1}$

* Durch die Diskretisierung wird die Summe i.A. nicht mehr 1 ergeben. Entspr. muss die Gauß-Funktion durch die Summe aller Gewichte normiert werden.

Binomialfilter (1)

Binomialfilter sind sehr gute Approximationen der Gauß-Filter, die effizient mit ganzzahligen Operationen berechnet werden können. Sie heißen so, weil sich ihre Werte aus den Binomialkoeffizienten ergeben.

Binomialfilter werden zunächst als 1-dim. Filter entwickelt und dann auf 2-dim. Filter erweitert. Die *unnormierten* Koeffizienten der *eindimensionalen Binomialfilter B^k der Ordnung k und der Größe $k+1$* werden nach folgender Formel entwickelt:

$$b_i^k = \binom{k}{i}.$$

Also normiert:

$$B^1 = \frac{1}{2} (1 \ 1), \quad B^2 = \frac{1}{4} (1 \ 2 \ 1),$$

$$B^3 = \frac{1}{8} (1 \ 3 \ 3 \ 1), \quad B^4 = \frac{1}{16} (1 \ 4 \ 6 \ 4 \ 1),$$

...

Für die Filterung sind nur Binomialfilter von gerader Ordnung k brauchbar, da diese eine ungerade Anzahl $k+1$ von Filterkoeffizienten zeigen.

Binomialfilter (2)

Ein quadratisches, 2-dim. Binomialfilter $B^{k,k}$ ergibt sich durch Matrixmultiplikation der zwei 1-dim. Binomialfilter B^k gleicher Ordnung:

$$B^{k,k} = B^k \times (B^k)^T$$

Für die zweidimensionalen Binomialfilter der Ordnungen 2 und 4:

$$B^{2,2} = \frac{1}{4} \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} \times \frac{1}{4} \begin{pmatrix} 1 & 2 & 1 \end{pmatrix} = \frac{1}{16} \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix},$$

$$B^{4,4} = \frac{1}{16} \begin{pmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{pmatrix} \times \frac{1}{16} \begin{pmatrix} 1 & 4 & 6 & 4 & 1 \end{pmatrix} = \frac{1}{256} \begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}.$$

Binomialfilter (3)

Anmerkungen zu den Binomialfiltern

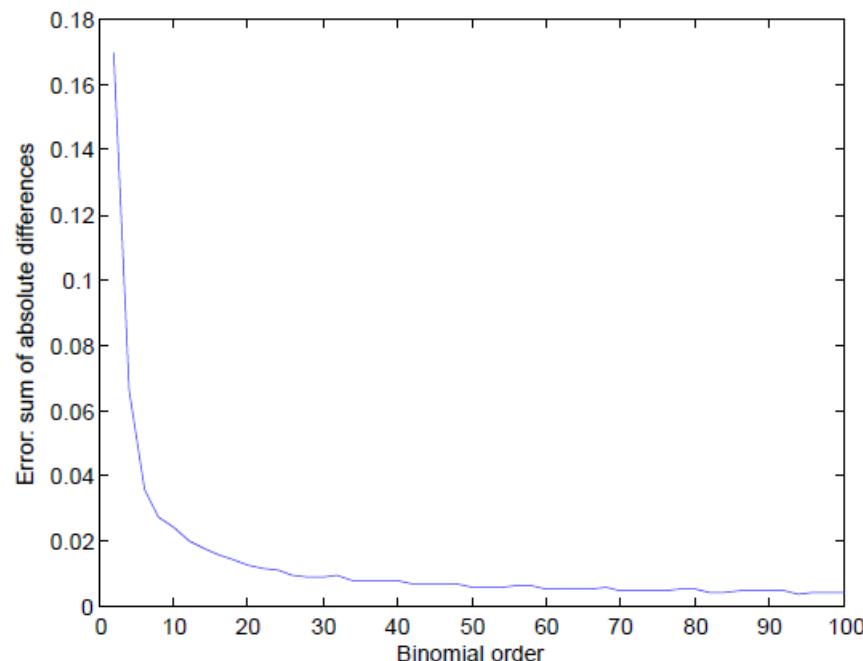
- Per Definition sind auch die zweidimensionalen Binomialfilter $B^{k,k}$ separabel
- Wenn die Ordnung k der Binomialfilter gegen Unendlich geht, ergibt sich das Gauß-Filter

Binomialfilter (4)

Zur approximativen Güte der Binomialfilter:

- Ein Binomialfilter der Ordnung k führt näherungsweise dieselbe Glättung durch wie ein Gauß-Filter mit $\sigma = k^{1/2}/2$.
- Die Approximation der Gauß-Funktion $G_{x,\sigma}$ (bis inkl. 3σ) durch die Binomialfilter B^k zeigt die Fehlerkurve $\sum |G_{x,\sigma} - B^k|$ mit $\sigma = k^{1/2}/2$.

Quelle: K.G. Derpanis: Overview of Binomial Filters. Lecture notes, March 2005, York Univ. (CA).



Binomialfilter (5)

Der Vergleich zwischen Mittelwertfilter und Binomialfilter zeigt, dass die Glättung beim gleichgroßen Binomialfilter nicht so stark ausfällt.

Im Ggs. zum Mittelwertfilter nimmt beim Binomialfilter – wie auch beim Gauß-Filter – der Glättungseffekt mit zunehm. Distanz zum Zentrum des Filterkerns ab.



Mittelwertfilter 5x5



Binomialfilter 5x5

Zusammenfassung

- Deterministische Einflüsse basieren auf wiederholbaren Einwirkungen bei der Bildaufnahme.
- Viele deterministische Einflüsse lassen sich durch lineare Operatoren und insbes. verschiebungsinvariante, lineare Operatoren beschreiben.
- Die Konvolution einer Signalfunktion mit einer Konvolutionsfunktion ist eine verschiebungsinvariante, lineare Operation, die in der Bildverarbeitung zur Rauschunterdrückung und Kantenhervorhebung benutzt wird.
- Bildstörungen durch Rauschen werden als additive Überlagerungen des Signals durch ein stochastisch modelliertes Rausches modelliert.
- Lineare Filterung zur Rauschunterdrückung lässt sich durch verschiedene Filter umsetzen wie Mittelwertfilter, Gauß-Filter und Binomialfilter.

Intelligente Sehsysteme

3 Hervorhebung von Kanten

Sobel-Operator

Konvolution und Korrelation

Laplace-Operator, LoG-Operator, DoG-Operator

Henry Hölzemann

- Gradienten der Bildfunktion
 - Sobel-Operator
- Konvolution und Korrelation
- 2. Ableitung der Bildfunktion
 - Laplace-Operator
 - Laplacian-of-Gaussian-Operator
 - Difference-of-Gaussian-Operator
 - Positionsbestimmung von Nulldurchgängen

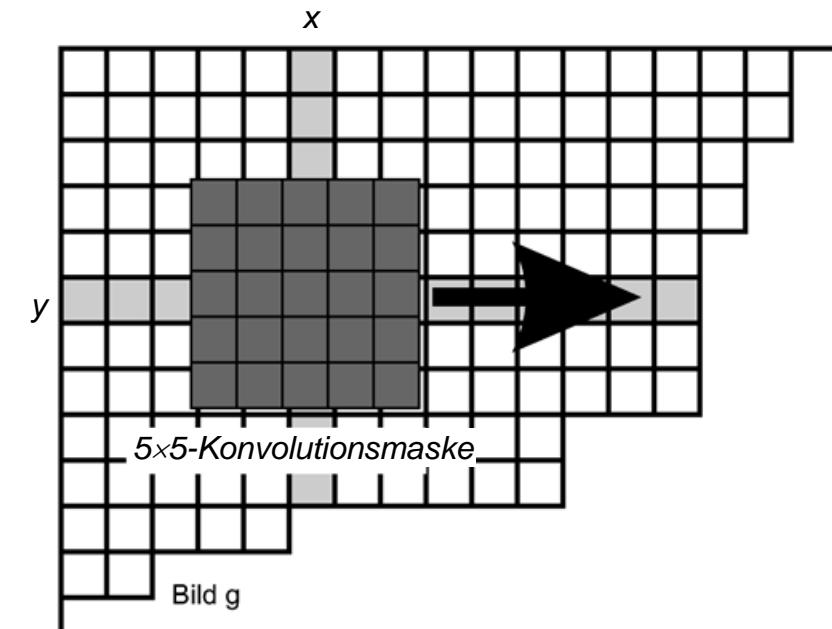
Lineare Filterung durch Konvolution (1)

Mit der Konvolution

$$(f * g)(x, y) = \sum_u \sum_v f(u, v) \cdot g(x-u, y-v) \text{ mit } u, v = -(m-1)/2, \dots, (m-1)/2$$

sind Pixel in Abhängigkeit von ihrer lokalen Pixelnachbarschaft veränderbar

- damit ist **Rauschunterdrückung** möglich (z.B. Mittelwert-, Gauß- und Binomialfilter)
- sowie die **Hervorhebung von Kantenpixeln**

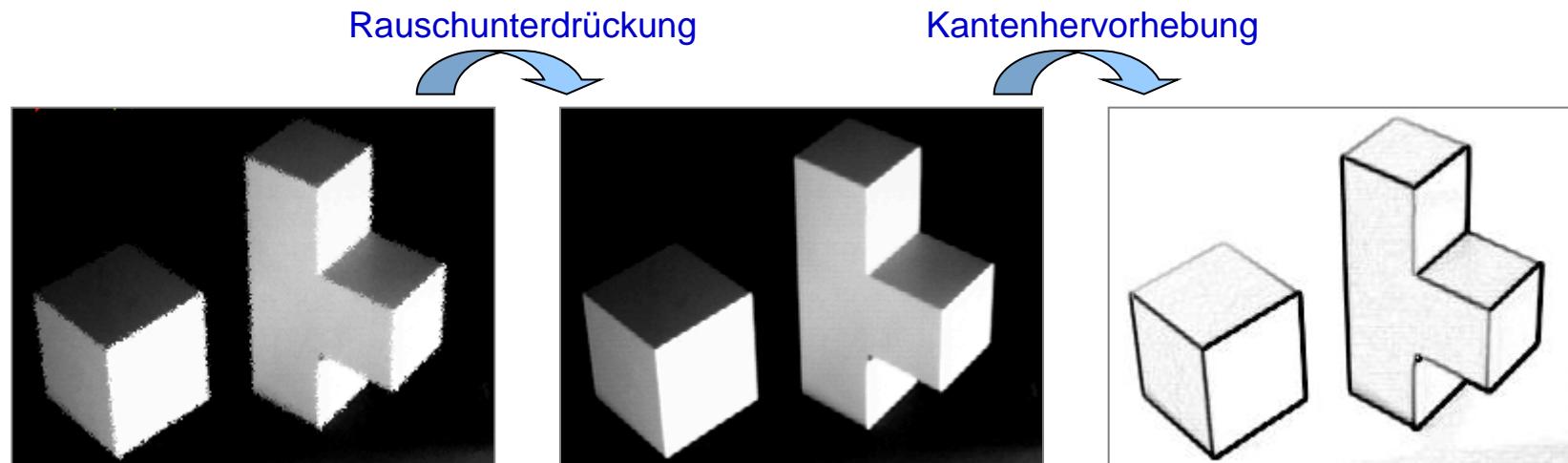


Nach Klaus Tönnies: Grundlagen der
Bildverarbeitung, Pearson Studium, 2005.

Lokale Nachbarschaft bei Konvolution

Beide Anwendungen der Konvolution basieren auf der Berücksichtigung der lokalen Pixelnachbarschaften:

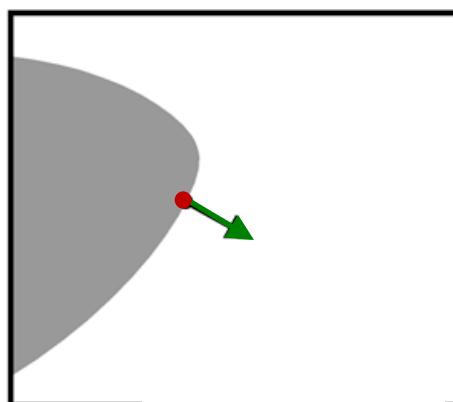
- bei **Rauschunterdrückung**, weil Rauschen nur als statistische Eigenschaft einer Menge von Pixel charakterisierbar ist
- bei **Kantenhervorhebung**, weil Kanten sich nur durch den Vergleich zwischen den Intensitäten benachbarter Pixel herausbilden



Zum Ergebnis einer Kantenhervorhebung

Die Erkennung von **Kantenpixel** kann prinzipiell resultieren in

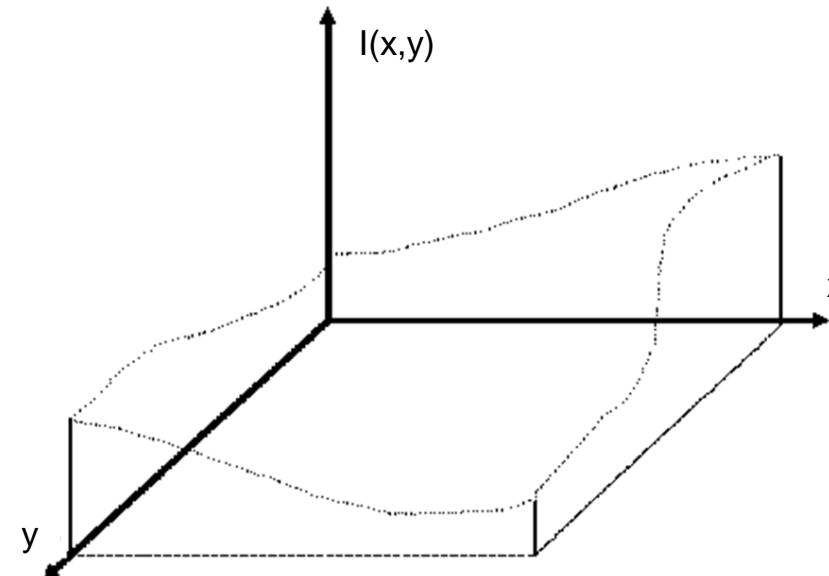
- einen **Positionswert** des Kantenpixels (Position des Pixels)
- einen **Betragswert** des Kantenpixels (Länge des Vektors)
- einen **Orientierungswert** des Kantenpixels (Richtung des Vektors)



Kantenmodell bei kontinuierlicher Bildfunktion (1)

Zurück zum Modell eines Grauwertbildes mit **reellwertiger Intensitätsfunktion** $I(x,y)$ in **reellwertigen Ortskoordinaten** x und y :

$$I: \mathbb{R}_{[x_{\min}, x_{\max}]} \times \mathbb{R}_{[y_{\min}, y_{\max}]} \rightarrow \mathbb{R}_{[l_{\min}, l_{\max}]}, \\ (x, y) \rightarrow I(x, y).$$



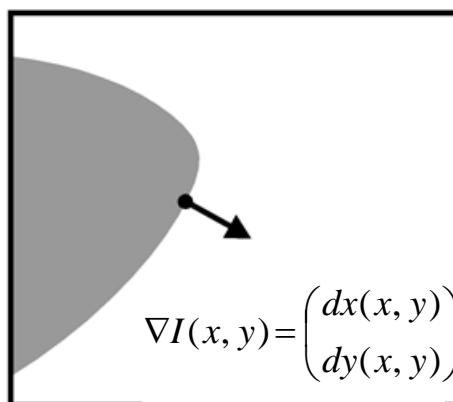
Bildquelle: Peter Haberäcker: Praxis der Digitalen Bildverarbeitung und Mustererkennung. Hanser, 1995.

- Die **Stärke einer Kante** nehmen wir als Intensitätsunterschied zwischen benachbarten Bildpunkten diesseits und jenseits der Kante wahr
 - ↪ bei einer kontinuierlichen Intensitätsfunktion entspricht eine Kante daher Stellen mit hohen Beträgen in der 1. Ableitung der Intensitätsfunktion

Kantenmodell bei kontinuierlicher Bildfunktion (2)

Der Gradient ist dann ein Vektor, dessen

- Länge die Kantenstärke repräsentiert und
- Richtung orthogonal zur Kantenrichtung verläuft



Kantenmodell bei diskreter Bildfunktion (1)

Bei diskreter Bildfunktion: [Approximation](#) der 1. Ableitung durch Ersetzen des Differentialquotienten für eine Pixelposition $p = (x,y)$ durch einen Differenzenquotienten:

$$\lim_{h \rightarrow 0} \frac{I(p+h) - I(p)}{h} \approx \frac{I(p+h) - I(p)}{h} \quad \text{für kleine } h$$

Kantenmodell bei diskreter Bildfunktion (2)

Implementierung der Approximation der 1. Ableitung der 2-dim. diskreten Bildfunktion durch Konvolution mit Differenzen in x- und y-Richtung:

$$\frac{\partial I(x, y)}{\partial x} \approx (-1 \quad 0 \quad 1) * I(x, y),$$

$$\frac{\partial I(x, y)}{\partial y} \approx \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} * I(x, y).$$

Die Konvolutionskerne bilden die Differenz in beiden Richtungen durch Subtraktion des Intensitätswertes vor dem aktuellen Pixel von dem Intensitätswert des nachfolgenden Pixels

Kantenmodell bei diskreter Bildfunktion (3)

Um a priori einer hohe Sensitivität gegenüber lokalem Rauschen vorzubeugen, können beide Konvolutionskerne jeweils mit einem Glättungsfilter kombiniert werden:

$$\text{Differenz in } x\text{-Richtung: } \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} \begin{pmatrix} -1 & 0 & +1 \end{pmatrix} = \begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix},$$

$$\text{Differenz in } y\text{-Richtung: } \begin{pmatrix} +1 \\ 0 \\ -1 \end{pmatrix} \begin{pmatrix} 1 & 2 & 1 \end{pmatrix} = \begin{pmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}.$$

Der Glättungsfilter ist jeweils der unnormierte 1-dim. Binomialfilter B^1 – jeweils orthogonal orientiert zum entspr. Differenzfilter

Filter des Sobel-Operators

- Diese Kombination von zwei Faltungen zur Gradientenapproximation mit kombinierter Glättung durch das B¹-Filter heißt **Sobel-Operator**
- Jede Konvolution besteht für sich wieder aus zwei separablen Operationen:
 - Glättung orthogonal zur Ableitungsrichtung
 - Differenzbildung in Ableitungsrichtung

$$\text{Horizontales Sobel - Filter } S_x : \begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} (-1 \ 0 \ +1)$$

$$\text{Vertikales Sobel - Filter } S_y : \begin{pmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} = \begin{pmatrix} +1 \\ 0 \\ -1 \end{pmatrix} (1 \ 2 \ 1)$$

Resultate des Sobel-Operators

Mit der Anwendung der beiden Sobel-Filter durch Konvolution ist die Operation noch nicht abgeschlossen. Das Ergebnis sind lediglich die Approximationen der beiden gerichteten Ableitungen.

Vgl. Folie 5

Der **Gradientenbetrag S** ergibt dann nach:

$$S \approx \sqrt{S_x(x, y)^2 + S_y(x, y)^2}.$$

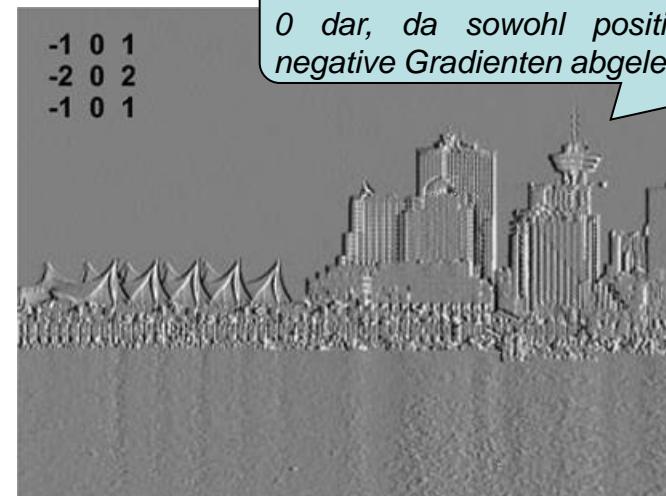
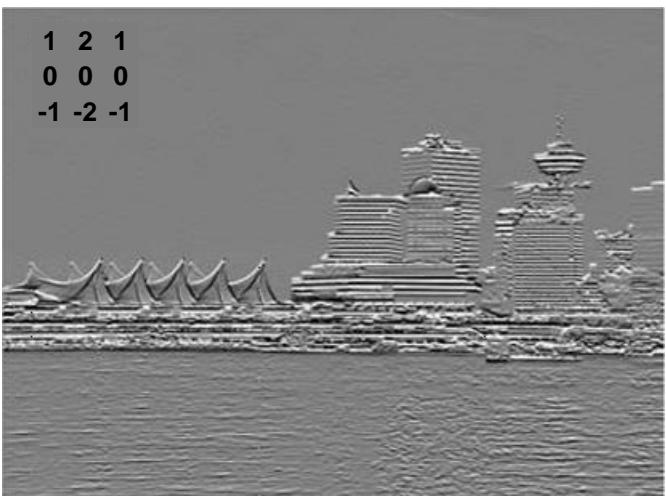
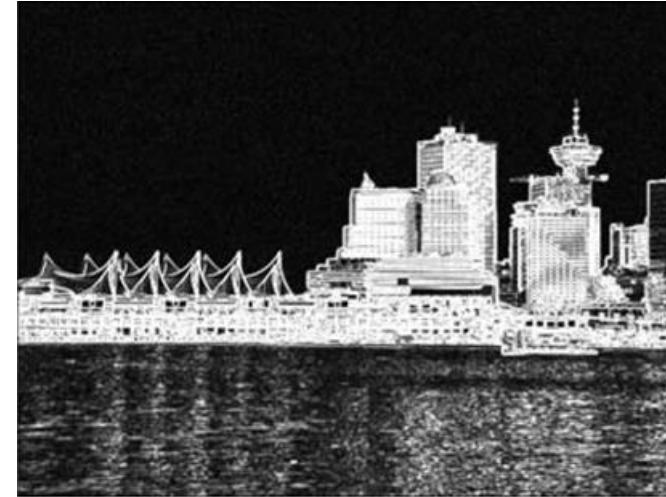
Die **Gradientenrichtung Θ** wird berechnet nach: *

$$\Theta \approx \begin{cases} \arctan(S_y(x, y) / S_x(x, y)) & \text{für } S_x(x, y) \neq 0, \\ 90^\circ & \text{für } S_x(x, y) = 0, S_y(x, y) \neq 0. \end{cases}$$

* Für $S_x(x, y) = S_y(x, y) = 0$ ist Θ undefiniert

Anwendung des Sobel-Operators (1)

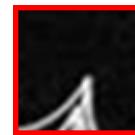
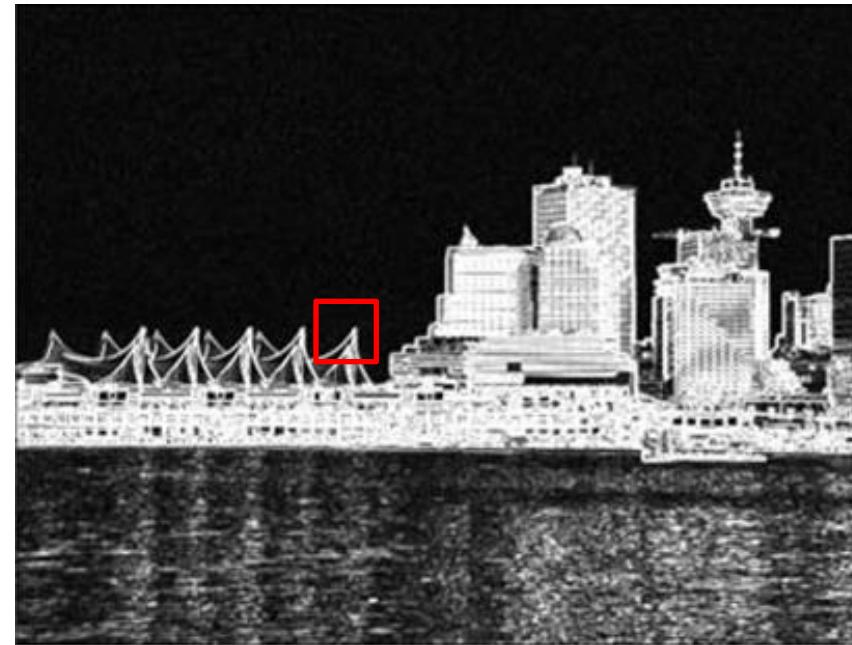
Beispiel: Das Originalbild (l.o.), die Gradientenbeträge (r.o.) sowie gerichteten Gradienten in vertikale (l.u.) und horizontale (r.u.) Richtung.



In der unteren Reihe stellt Grau den Wert 0 dar, da sowohl positive als auch negative Gradienten abgeleitet werden.

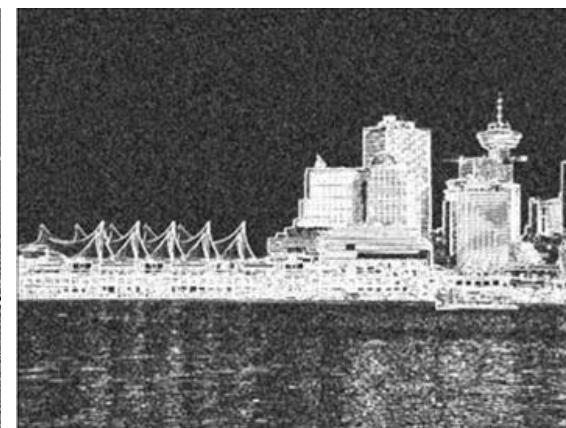
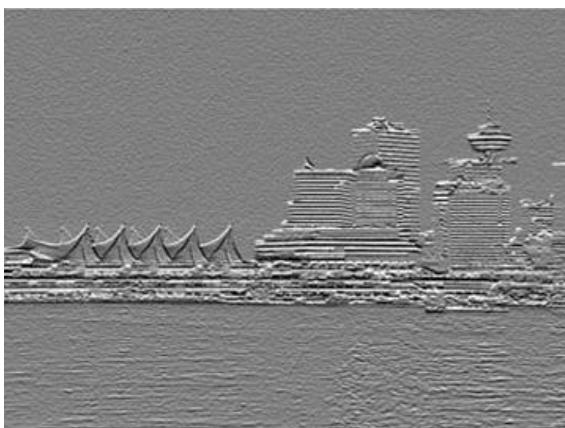
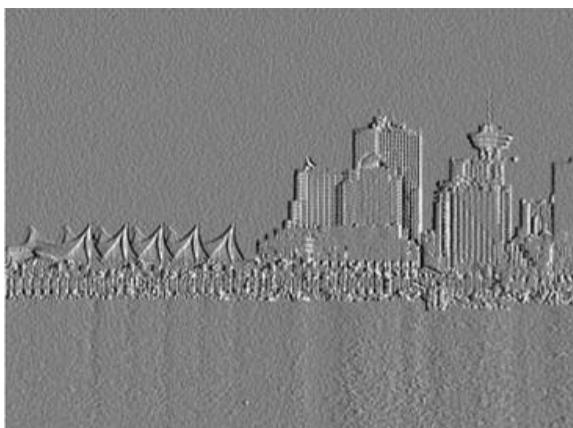
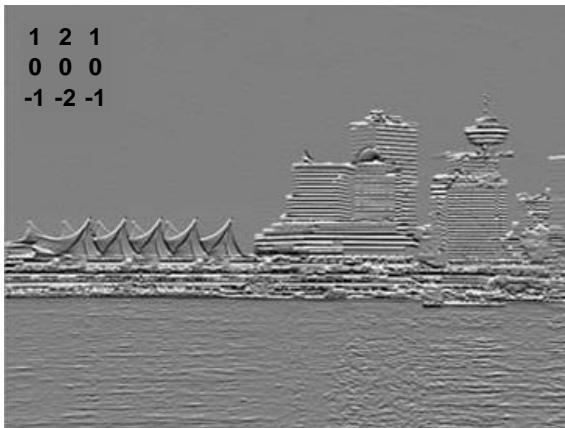
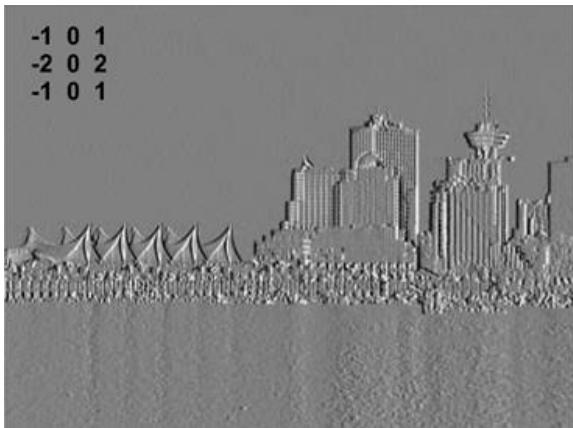
Anwendung des Sobel-Operators (2)

Beispiel: Gradientenbeträge skaliert auf [0,255]



Anwendung des Sobel-Operators (4)

Beide Komponenten des Sobel-Operators sowie der Betrag des Gradienten auf rauscharmer (oben) und verrauschter (unten) Bildversion:



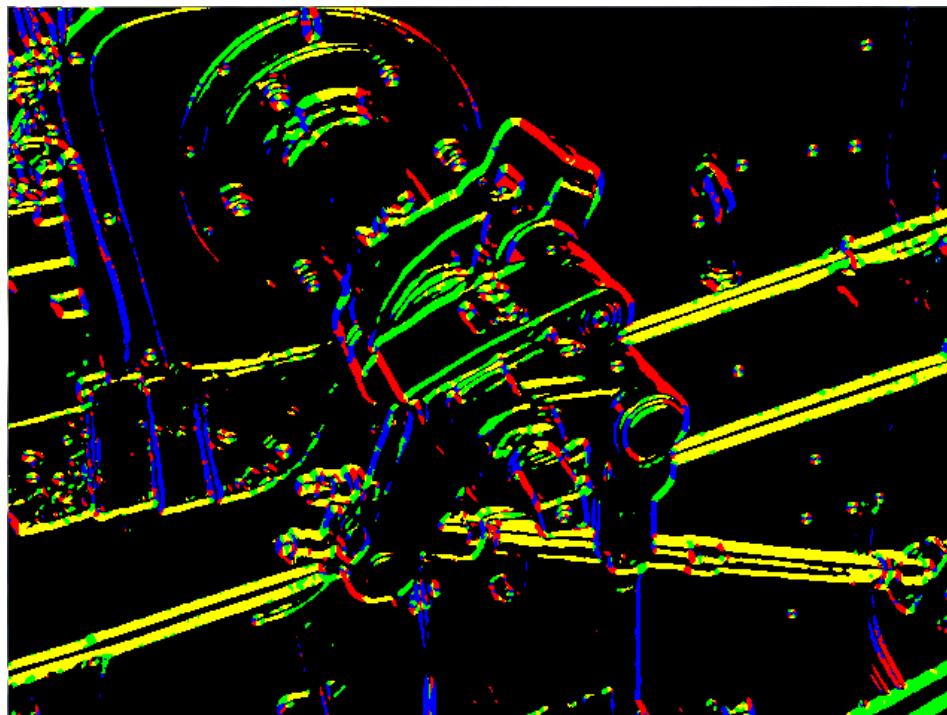
Anwendung des Sobel-Operators (5)

Für eine Visualisierung der Gradientenrichtung Θ bietet sich ein Binning der Winkelwerte an:

- Beim *Binning* wird der Wertebereich der Größe nach aufsteigend in Intervalle – sogenannte bins (engl. für Behälter) – eingeteilt
- Bspl.: Farbkodierung der Gradientenrichtung Θ in:^{*}
 - blau = $0^\circ \pm 22,5^\circ$
 - rot = $45^\circ \pm 22,5^\circ$
 - gelb = $90^\circ \pm 22,5^\circ$
 - grün = $135^\circ \pm 22,5^\circ$

* Einige Details dazu (s. Übungsblatt):

- (1) Die Intervalle der Bins müssen natürlich disjunkt sein ($\sim <, \geq; \leq, >$)
- (2) Die Richtungen unterscheiden hier nicht Übergänge von hell nach dunkel und umgekehrt
- (3) Die Behandlung der negativen Werte von Arkustangens und der 90° -Werte sind hier nicht berücksichtigt



Bildquelle: http://en.wikipedia.org/wiki/Canny_edge_detector (24.06.2011)

Sobel-Operator und Konvolution (1)

- Die Konvolution wendet die Filtermasken generell punktgespiegelt auf die Bildmatrix an – dies ergibt sich aus den negativen Vorzeichen der Verschiebungsvektoren u und v in der Konvolutionsformel:

$$(f * g)(x,y) = \sum_u \sum_v f(u,v) \cdot g(x-u, y-v) \text{ mit } u,v = -(m-1)/2, \dots, (m-1)/2$$

- Bei punktsymmetrischen Glättungsfilters ist dies ohne Auswirkung
- Bei orientierten Filtern wie den Sobel-Filtern sind Auswirkungen zu beachten

Sobel-Operator und Konvolution (2)

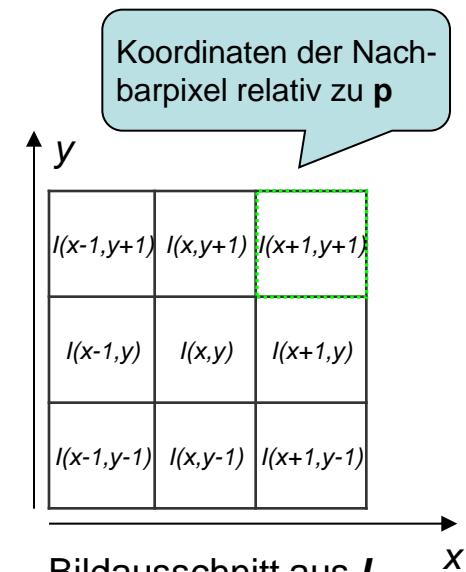
Konvolution mit den Sobel-Filtermasken auf ein Bildpixel $\mathbf{p} = (x,y)$:

-1	0	1
-2	0	2
-1	0	1

Horizontal. Sobel-Filter \mathbf{S}_x

1	2	1
0	0	0
-1	-2	-1

Vertikal. Sobel-Filter \mathbf{S}_y



$$\mathbf{S}_x * \mathbf{I} = I(x-1,y+1) + 2 \cdot I(x-1,y) + I(x-1,y-1) - I(x+1,y+1) - 2 \cdot I(x+1,y) - I(x+1,y-1),$$

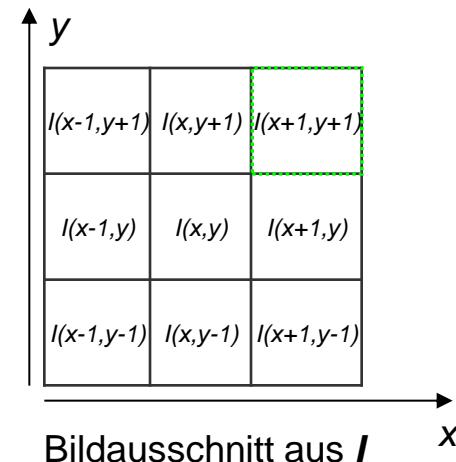
$$\mathbf{S}_y * \mathbf{I} = I(x+1,y-1) + 2 \cdot I(x,y-1) + I(x-1,y-1) - I(x+1,y+1) - 2 \cdot I(x,y+1) - I(x-1,y+1).$$

Konvolution und Korrelation (1)

Die punktgespiegelten Anwendung der Konvolution wird oft als verwirrend betrachtet, um die Koeffizienten der Konvolutionsmaske mit den Nachbarpixeln zu assoziieren, auf die sie angewendet werden:

-1	0	1
-2	0	2
-1	0	1

Horizontal. Sobel-Filter S_x



Konvolution und Korrelation (2)

Aus diesem Grund wird die Konvolution

$$(f * g)(x, y) = \sum_u \sum_v f(u, v) \cdot g(x-u, y-v) \text{ mit } u, v = -(m-1)/2, \dots, (m-1)/2$$

häufig durch die Korrelation ersetzt bzw. implementiert:

$$(f \oplus g)(x, y) = \sum_u \sum_v f(u, v) \cdot g(x+u, y+v) \text{ mit } u, v = -(m-1)/2, \dots, (m-1)/2.$$

Die Korrelation entspricht eher dem Bild einer gleitenden Maske mit korrespondierenden Stellen von Koeffizienten der Filtermaske einerseits und zu gewichtenden Bildpixeln andererseits.

Konvolution und Korrelation (3)

Die Anwendung der beiden Sobel-Filter durch die Korrelation zeigt die anschauliche Korrespondenz zwischen den Koeffizienten der Filterkerne einerseits und den Nachbarpixeln des untersuchten Pixels andererseits:

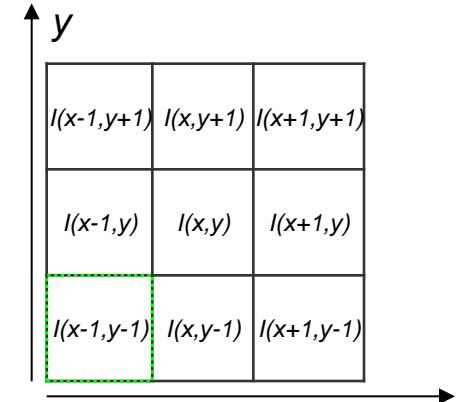
$$(f \oplus g)(x,y) = \sum_u \sum_v f(u,v) \cdot g(x+u, y+v) \text{ mit } u,v = -(m-1)/2, \dots, (m-1)/2.$$

-1	0	1
-2	0	2
-1	0	1

Horizontal. Sobel-Filter \mathbf{S}_x

1	2	1
0	0	0
-1	-2	-1

Vertikal. Sobel-Filter \mathbf{S}_y



Bildausschnitt aus I

$$\mathbf{S}_x \oplus I = I(x+1, y-1) + 2 \cdot I(x+1, y) + I(x+1, y+1) - I(x-1, y-1) - 2 \cdot I(x-1, y) - I(x-1, y+1),$$

$$\mathbf{S}_y \oplus I = I(x-1, y+1) + 2 \cdot I(x, y+1) + I(x+1, y+1) - I(x-1, y-1) - 2 \cdot I(x, y-1) - I(x+1, y-1).$$

Konvolution und Korrelation (4)

- Konvolution und Korrelation erzeugen bei orientierten Filtermasken unterschiedliche Ergebnisse.
- Daher ist anzugeben, mit welcher Operation eine Filtermaske angewendet wurde.
- Beispiel für horizontal. Sobel-Filter:

$$\mathbf{S}_x * \mathbf{I}$$

$$\begin{aligned} &= (1+2+1)-(100+200+100) \\ &= 4 - 400 = -396 \end{aligned}$$

$$\mathbf{S}_x \oplus \mathbf{I}$$

$$\begin{aligned} &= -(1+2+1)+(100+200+100) \\ &= -4 + 400 = 396 \end{aligned}$$

$\begin{array}{ c c c } \hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$	Horizont. Sobel-Filter \mathbf{S}_x	$\begin{array}{ c c c } \hline 1 & 20 & 100 \\ \hline 1 & 20 & 100 \\ \hline 1 & 20 & 100 \\ \hline \end{array}$	Bildausschnitt aus \mathbf{I}
--	---------------------------------------	--	---------------------------------

Bemerkung: Betrachten Sie nochmals die Abbildungen zu den gerichteten Gradienten in den Folien 13 - 16. Wurde die Konvolution oder die Korrelation angewandt?

Beispiel

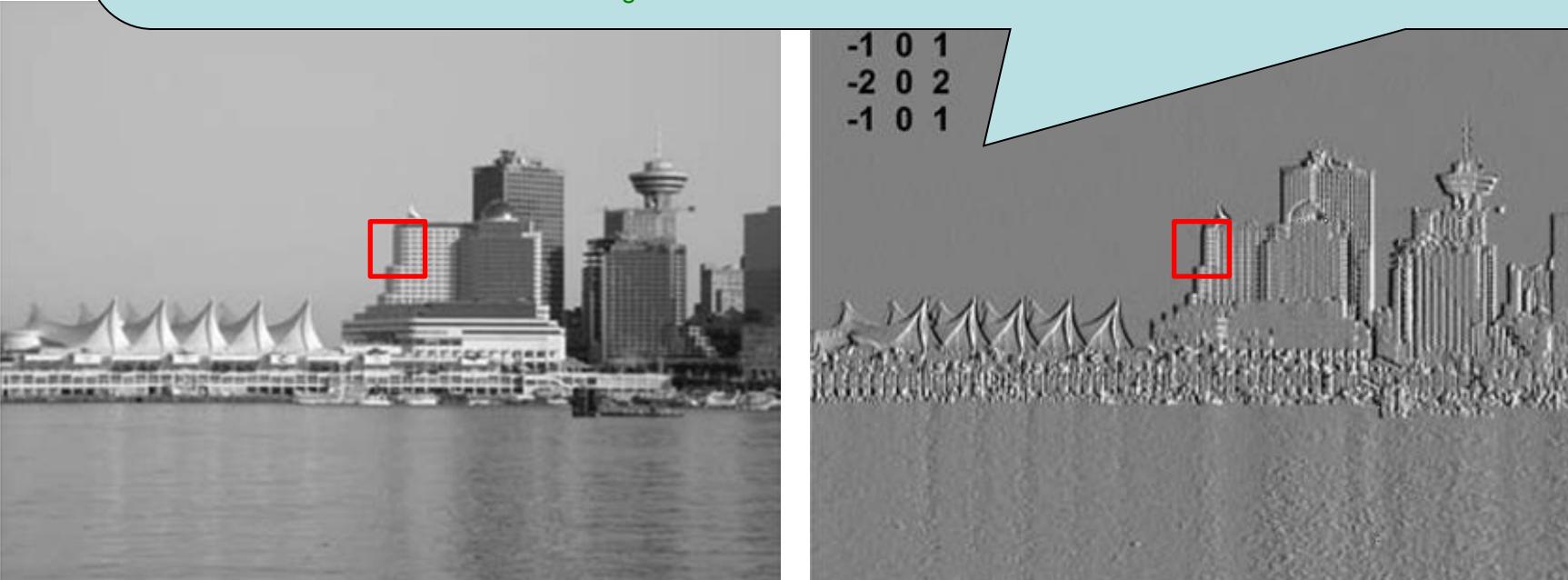
Übergang von hellem Himmel (links) zu dunkler Fassade (rechts):

1. Konvolution:

- dunkle Fassadenpixel (alle z.B. 50) werden negativ gewichtet: -200;
- helle Himmelpixel (alle z.B. 200) werden positiv gewichtet: 800;
- Gewichtete Summe: 600 \sim also positiver Gradient \sim helle Pixel im Gradientenbild
- **Widerspruch zu dunklen Pixeln im Gradientenbild**

2. Korrelation:

- dunkle Fassadenpixel (alle z.B. 50) werden positiv gewichtet: 200;
- helle Himmelpixel (alle z.B. 200) werden negativ gewichtet: -800;
- Gewichtete Summe: -600 \sim also negativer Gradient \sim dunkle Pixel im Gradientenbild
- **Übereinstimmung mit dunklen Pixeln im Gradientenbild**



$$\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$

∇



Normierung

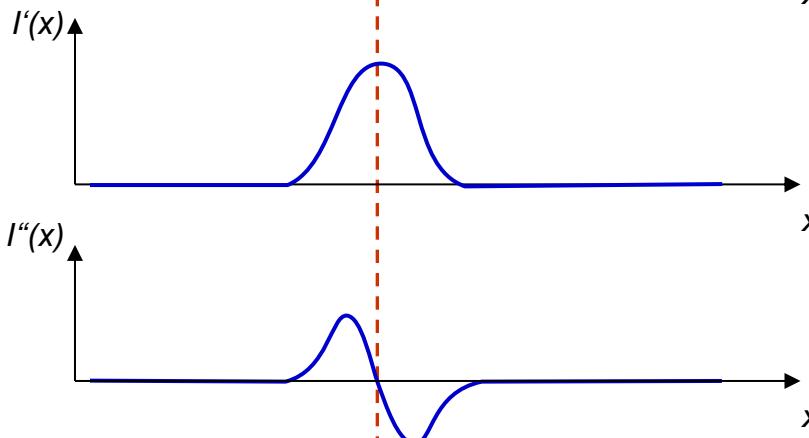
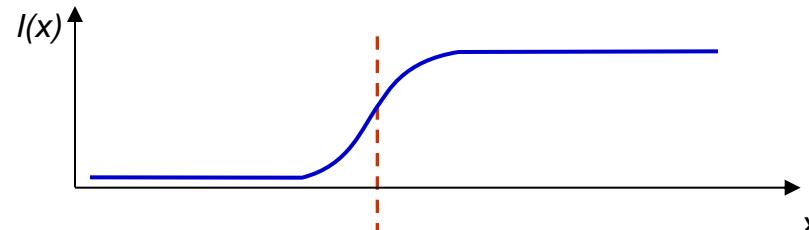
Das letzte Beispiel zeigte, dass die Ergebnisse des Sobel-Operators leicht die darstellbaren Intensitäten eines gegebenem Intensitätsspektrums (z.B. 0,...,255) unter- oder überschreiten können:

$$\begin{aligned} S_x \oplus I &= 396, \quad S_y \oplus I = 0, \\ S &= (S_x(x,y)^2 + S_y(x,y)^2)^{\frac{1}{2}} = 396. \end{aligned}$$

Allg. sind die Werte dann über *lineare Intensitätsspreizung* oder *Histogrammlinearisation* (s. 2. Vorl.) in das gegebene Intensitätspektrum zu transformieren.

2. Ableitung der Bildfunktion

- Alle bisherigen Operatoren zur Kantenhervorhebung basieren auf der 1. Ableitung der Bildfunktion
- Auch die 2. Ableitung der Bildfunktion ist zur Kantenhervorhebung nutzbar
- Aus dem Kantenmodell für kontinuierliche Bildfunktionen ergibt sich:
 - Kanten sind Orte mit maximalen Werten in der 1. Ableitung
 - dort zeigt die 2. Ableitung Vorzeichenwechsel – sog. Nulldurchgänge



2. Ableitung der Bildfunktion

Analog zur 1. Ableitung besteht die 2. Ableitung der Bildfunktion mit 2-dim. Definitionsbereich aus einer Menge von partiellen Ableitungen, nämlich vier partiellen 2. Ableitungen:

$$\frac{\partial^2 I}{\partial x^2}(x,y) , \quad \frac{\partial^2 I}{\partial y^2}(x,y) , \quad \frac{\partial^2 I}{\partial x \partial y}(x,y) , \quad \frac{\partial^2 I}{\partial y \partial x}(x,y) .$$

Diskr. Approximation der 2. Ableitung

Auch die 2. Ableitung der Bildfunktion durch Differenzen approximierbar:

- $\frac{\partial^2 I}{\partial x^2}(x, y)$ ist durch zweimaliges Anwenden eines Gradientenoperators in x-Richtung approximierbar
- $\frac{\partial^2 I}{\partial x \partial y}(x, y)$ ist durch Anwenden eines Gradientenoperators in x-Richtung gefolgt von der eines Gradientenoperators in y-Richtung approximierbar

Laplace-Operator (1)

Laplace-Operator

- basiert auf der Summe der partiellen 2. Ableitungen
- wird mit ∇^2 bezeichnet
- zeigt zwei Versionen:
 - Summe über beide zweimaligen Ableitungen in x- und y-Richtung:

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

- Summe über alle vier part. Ableitungen:

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

Laplace-Operator (2)

- Die diskrete Approximation des Laplace-Operators basiert auf die bereits vorgestellten einfachen Differenzopertoren zur partiellen Gradientenapproximation:

$$D_{x,3} = \begin{pmatrix} -1 & 0 & 1 \end{pmatrix}, \quad D_{y,3} = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$$

- Um kompakte 3×3 -Laplace-Operatoren zu erzeugen, werden aber noch kompaktere Differenzopertoren verwendet:

$$D_{x,2} = \begin{pmatrix} -1 & 1 \end{pmatrix}, \quad D_{y,2} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Laplace-Operator (3)

- Approx. Der 2. Ableitungen entlang der x- und der y-Richtung :

$$\begin{aligned}\frac{\partial^2 I}{\partial x^2} &\approx \frac{\partial D_{x,2}}{\partial x} \\ &= \frac{\partial(I(x+1,y) - I(x,y))}{\partial x} \\ &= \frac{\partial I(x+1,y)}{\partial x} - \frac{\partial I(x,y)}{\partial x} \\ &\approx (I(x+2,y) - I(x+1,y)) - (I(x+1,y) - I(x,y)) \\ &= I(x+2,y) - 2 \cdot I(x+1,y) + I(x,y)\end{aligned}$$

- Zentrum der Approximation ist Pixel $[x+1,y]$. Substitution von x mit $x-1$ führt zur Approximation für die zweite partielle Ableitung im Pixel $[x,y]$:

$$\frac{\partial^2 I}{\partial x^2} \approx I(x+1,y) - 2 \cdot I(x,y) + I(x-1,y)$$

Laplace-Operator (4)

- Analog zur 2. Ableitung in x-Richtung

$$\frac{\partial^2 I}{\partial x^2} \approx I(x+1, y) - 2 \cdot I(x, y) + I(x-1, y)$$

ergibt sich die 2. Ableitung in y-Richtung:

$$\frac{\partial^2 I}{\partial y^2} \approx I(x, y+1) - 2 \cdot I(x, y) + I(x, y-1)$$

- Kombination beider part. Ableitungen zu einem einzigen Operator führt zum sog. L₄-Filterkern zur Approx. des Laplace-Operators:

$$\begin{aligned}\nabla^2 I(x, y) &= \frac{\partial^2 I}{\partial x^2}(x, y) + \frac{\partial^2 I}{\partial y^2}(x, y) \\ &= I(x+1, y) + I(x-1, y) + I(x, y+1) + I(x, y-1) - 4 \cdot I(x, y)\end{aligned}$$

Laplace-Operator (5)

Der L_4 -Laplace-Operator

$$L_4 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} \xrightarrow{\text{häufige Variante}} \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

basiert nur auf der Approximation der 2. Ableitungen in x- und y-Richtung nach:

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

Die Approximation der 2. Version

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

führt zum L_8 -Laplace-Operator :

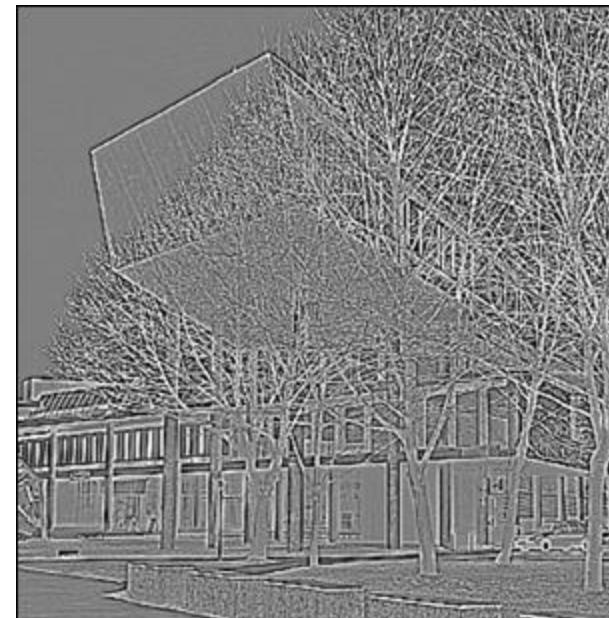
$$L_8 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix} \xrightarrow{\text{häufige Variante}} \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Laplace-Operator (6)

- Nach Anwendung des Laplace-Operators sind die Positionen der Nulldurchgänge im Bild abzuleiten.
- Für den L_4 -Laplace-Operator:
 - das Bild wird dazu zunächst jeweils um einen Pixel nach links und um einen Pixel nach unten verschoben
 - ein Nulldurchgang liegt vor, wenn das Ergebnis des L_4 -Laplace-Operators **unterschiedliche Vorzeichen im unverschobenen Bild und einer der beiden verschobenen Versionen** zeigt
- Für den L_8 -Laplace-Operator erfolgt eine analoge Untersuchung. Allerdings ist das Bild zusätzlich auch in beide Diagonalenrichtungen zu verschieben

Laplace-Operator (7)

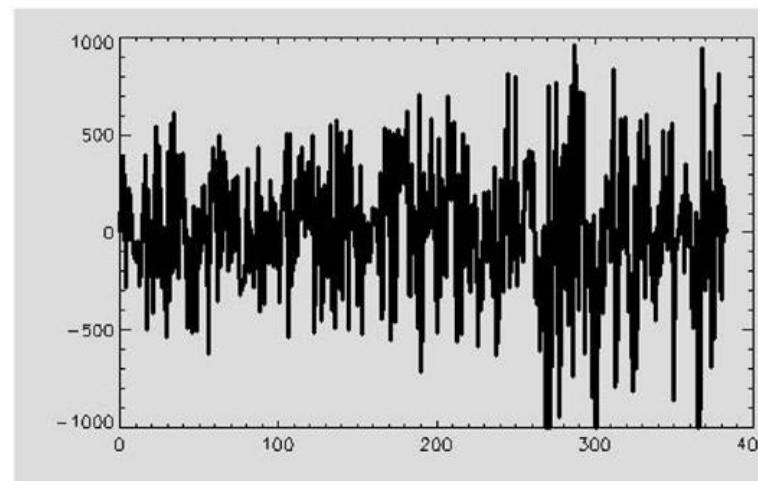
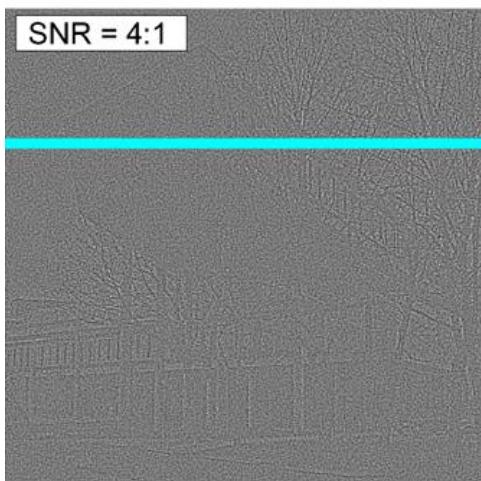
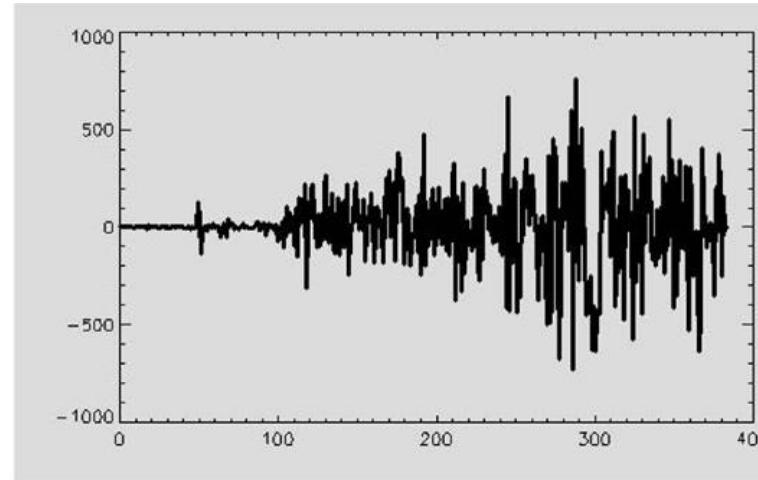
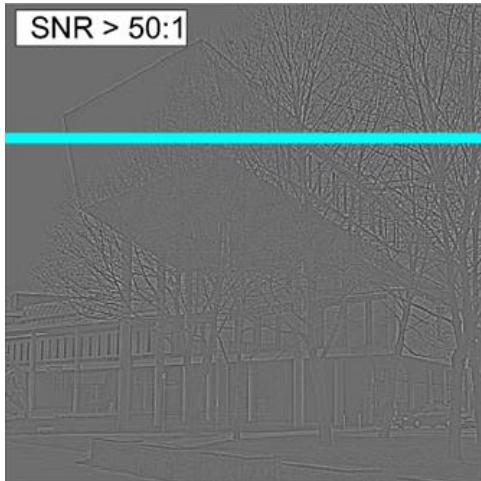
- Beispiel: gezeigt sind die Nulldurchgänge des Laplace-Operators:
 - dunkle Pixel bezeichnen negative Werte (Hell-Dunkel-Übergänge)
 - helle Pixel bezeichnen positive Werte (Dunkel-Hell-Übergänge)



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Laplace-Operator (8)

Der Laplace-Operator basiert auf einer zweimaligen Differentiation und ist daher sehr rauschempfindlich:



Filterantworten des Laplace-Operators in den blauen Bildzeilen.

Laplacian-of-Gaussian-Operator (1)

- Wegen der hohen Rauschempfindlichkeit des Laplace-Operators ist eine Kopplung mit einer vorherigen Glättung sinnvoll
- Der bekannteste Operator hierfür ist das Laplacian-of-Gaussian-Filter (kurz: [LoG-Filter](#))
- Das Filter entsteht durch Anwendung des Laplace-Operators auf die Gauß-Funktion:

$$f_{G,\sigma}(x, y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}.$$

Laplacian-of-Gaussian-Operator (2)

- Die Anwendung des Laplace-Operators auf die Gauß-Funktion ergibt :

$$LoG(x, y) = \nabla^2 f_{G,\sigma}(x, y)$$

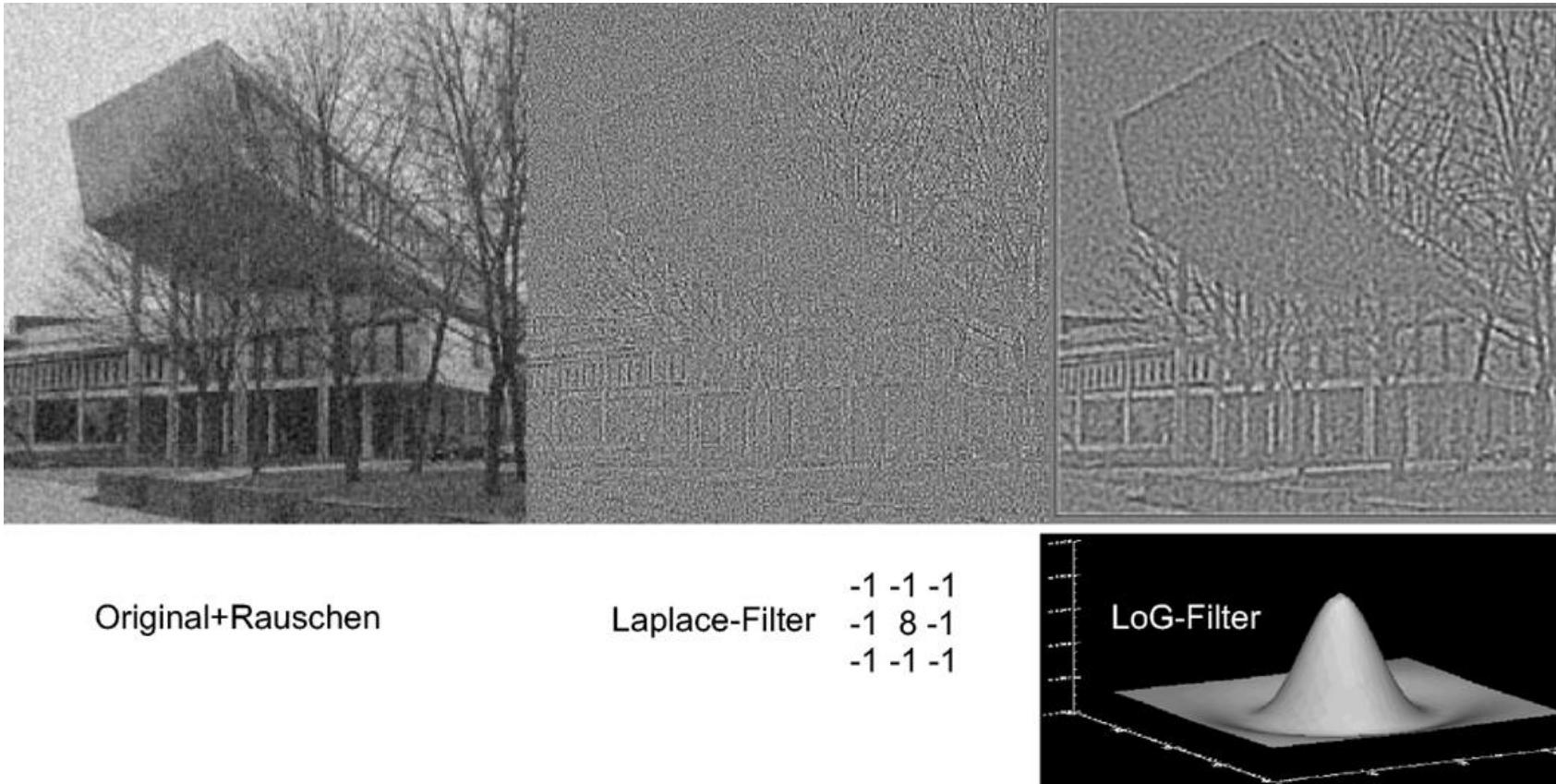
$$= \frac{\partial^2 f_{G,\sigma}}{\partial x^2}(x, y) + \frac{\partial^2 f_{G,\sigma}}{\partial y^2}(x, y)$$

$$= -\frac{1}{\pi\sigma^4} \left(1 - \frac{x^2 + y^2}{2\sigma^2}\right) e^{-\frac{x^2+y^2}{2\sigma^2}}.$$

- Wie bei der Diskretisierung der Gauß-Funktion ist auch beim LoG darauf zu achten, dass die Filtergröße k der Standardabweichung σ angemessen ist (z.B. $k = 2 \cdot \lceil 3\sigma \rceil + 1$)
- Durch Normierung ist zu sichern, dass die Summe der Filterwerte Null ergibt

Laplacian-of-Gaussian-Operator (3)

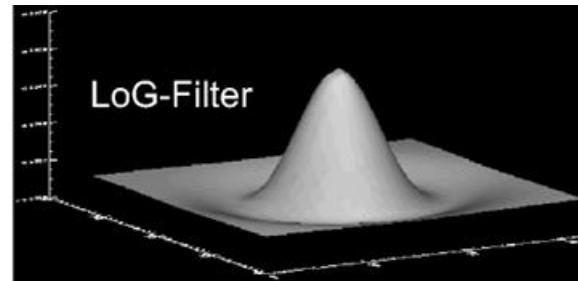
- Das LoG-Filter kann wegen der integrierten Rauschunterdrückung Nulldurchgänge von Kanten besser hervorzuheben als das Laplace-Filter:



Laplacian-of-Gaussian-Operator (4)

Das LoG-Filter ist auch bekannt als

- Mexican-Hat-Filter wegen seiner Form;
- Marr-Hildreth-Filter: der Informatiker David Marr und die Kognitionspsychologin Ellen Hildreth zeigten, dass die Signalverarbeitung in einer frühen Stufe der natürlichen visuellen Verarbeitung der LoG-Filterung ähnlich ist



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Laplacian-of-Gaussian-Operator (5)

- Zur Bestimmung von Nulldurchgängen werden auch oft Folgen von LoG-Filtern mit unterschiedlichen Standardabweichungen eingesetzt
- Orte, an denen Nulldurchgänge mit *verschiedenen* Filtergrößen erkannt werden, sind mit größerer W'keit Teil eines Kantenzuges als solche, die nur singulär mit einer bestimmten Standardabweichung erkannt werden
- Zusammen mit der integrierten Rauschunterdrückung führt diese Strategie bei verrauschten Bildvorlagen zu besseren Erkennungsraten

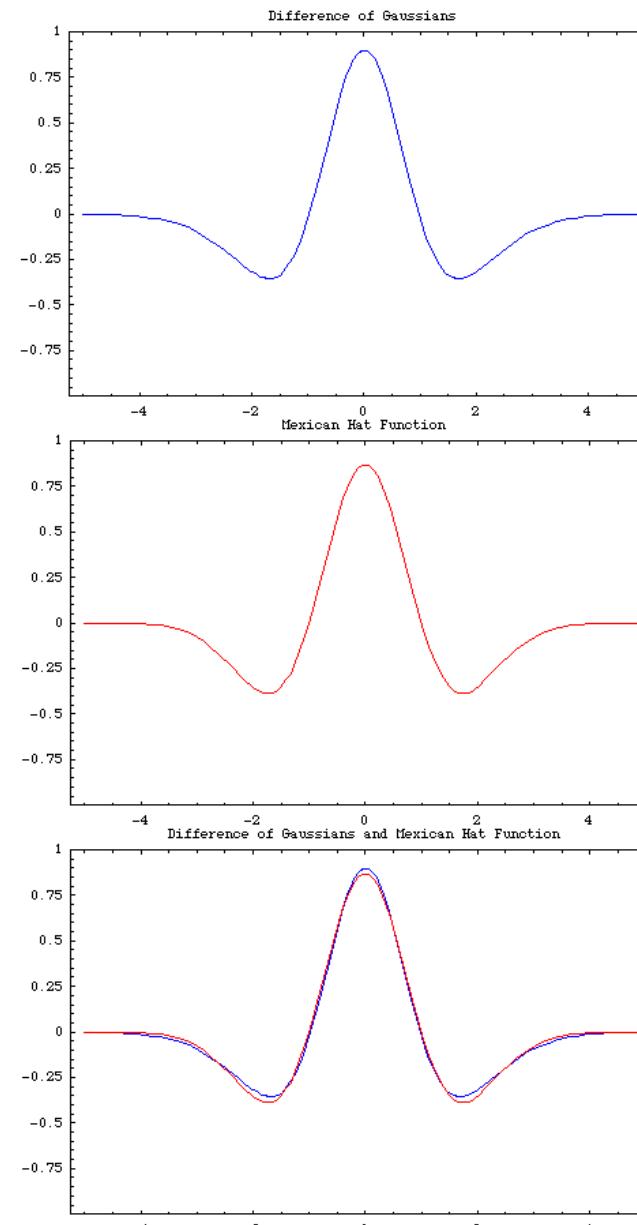
Difference-of-Gaussian-Operator (1)

- Der Difference-of-Gaussian-Operator (kurz: DoG) ist ein weiterer Operator, der die 2. Ableitung mit Rauschunterdrückung kombiniert
- Das Prinzip:
 - das Eingangsbild wird mit zwei Gauß-Filtern mit unterschiedlichen Standardabweichungen σ_1 bzw. σ_2 geglättet
 - Die beiden geglätteten Ergebnisbilder werden voneinander subtrahiert:

$$DoG(x, y) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}$$

Difference-of-Gaussian-Operator (2)

Die Differenz beider Gauß-Funktionen ist dem Mexican Hat und damit dem LoG-Filter ähnlich, da die Gauß-Funktion mit kleinerem σ ein höheres Maximum hat und schneller gegen Null geht als eine Gauß-Funktion mit größerem σ :



Bildquelle: http://en.wikipedia.org/wiki/Difference_of_Gaussians
(10.11.2011)

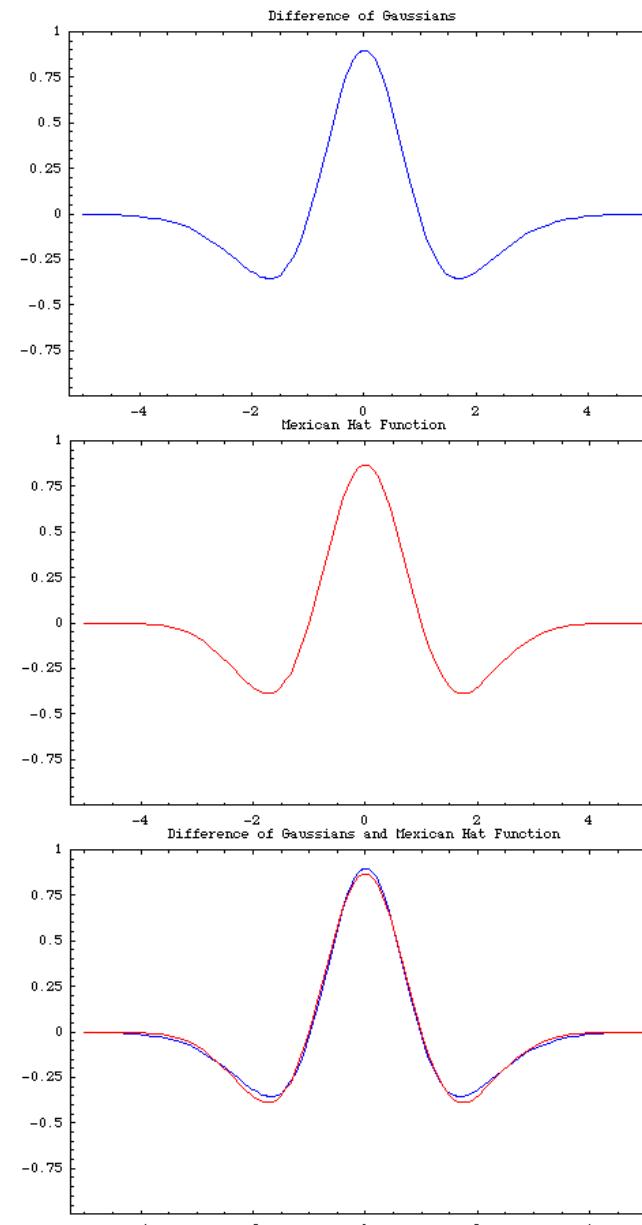
Difference-of-Gaussian-Operator (3)

Genauer erfolgt die Approximation eines LoG-Filters mit σ durch ein DoG-Filter nach

$$DoG(x, y, \sigma) =$$

$$\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} - \frac{1}{2\pi K^2\sigma^2} e^{-\frac{x^2+y^2}{2K^2\sigma^2}}$$

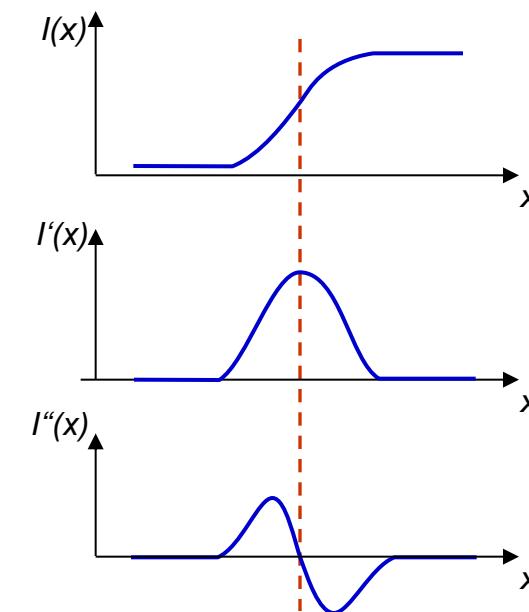
mit $K \sim 1,6$



Bildquelle: http://en.wikipedia.org/wiki/Difference_of_Gaussians
(10.11.2011)

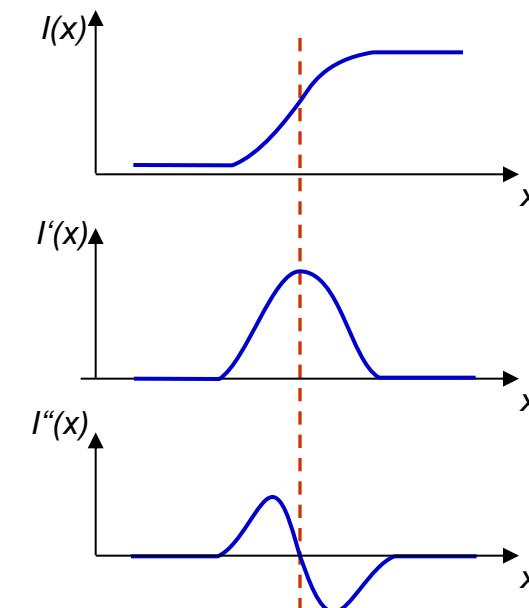
Positionsbestimmung von Nulldurchgängen (1)

- Die genaue Positionsbestimmung von Nulldurchgängen kann bei verrauschten Bildern trotz integrierter Rauschunterdrückung (LoG, DoG) schwierig sein.
- Hier kann ein Kantenmodell, das durch Rauschen bedingte Unschärfe berücksichtigt und damit anschaulich „verschmierten Kanten“ beschreibt, helfen.
- Das Modell geht davon aus, dass um einen gefundenen Nulldurchgang bzw. Kantenort eine lokale Umgebung in Gradientenrichtung existiert, in der sich *keine andere Kante befindet*.



Positionsbestimmung von Nulldurchgängen (2)

- Daher werden die Gradientenantworten innerhalb dieser Umgebung betrachtet und deren Schwerpunkt als Ort des Nulldurchgangs festgelegt.
- Innerhalb dieser Umgebung müssen die Integrale der 1. Ableitung in beiden Richtungen zu den Umgebungsgrenzen gleich sein.
- Also werden in diesem Ansatz die 1. Ableitung und die 2. Ableitung gemeinsam ausgewertet.



Positionsbestimmung von Nulldurchgängen (3)

Die subpixelgenaue Positionsbestimmung von Nulldurchgängen:

- 1) Berechnung von Gradienten *und* Nulldurchgängen
 - z.B. durch Gauß-Filterung gefolgt von *Sobel*- und *Laplace*-Filterung
- 2) An jedem Nulldurchgang
 - a) Bestimmung eines Umgebungsintervalls *in Gradientenrichtung* (*Sobel*-Operator) *um den Nulldurchgang* (ermittelt von *Laplace*-Operator)
 - b) Schwerpunktbestimmung der Gradientenbeträge im Umgebungsintervall
- 3) Der Schwerpunkt ist die gesuchte Position des Nulldurchgangs.

Fazit zur linearen Filterung (1)

- Kontrastverbesserung durch lineare Filterung erfolgt durch
 - Unterdrückung von Störsignalen,
 - Hervorhebung bzw. Extraktion von Kantensignalen.
- LEIDER führen die linearen Operatoren zur Rauschunterdrückung auch zur Kantenunterdrückung.
- DAHER muss in der Anwendung von linearen Operatoren häufig eine bzgl. des Anwendungsbereichs angemessene Balance zwischen Rauschunterdrückung und Kantenhervorhebung gewählt werden.

Fazit zur linearen Filterung (2)

- Die bei linearen Filtern notwendige Balance zwischen Rauschunterdrückung und Kantenhervorhebung ist für viele Anwendungen akzeptabel, da diese Balance hinsichtlich der Anwendung angepasst werden kann.
- Entsprechend gehören lineare Filter zum Standardmethodenarsenal des Computersehens und insbes. der *Low-Level-Vision* bzw. der *Bildverarbeitung*.

Zusammenfassung (1)

- Ein Ansatz der Kantenhervorhebung basiert auf der 1. Ableitung der Intensitätsfunktion eines Bildes. Für digitale Bilder wird die **1. Ableitung der zweidimensionalen diskreten Intensitätsfunktion** durch Differenzen in x- und y-Richtung **approximiert**. Dies führt z.B. zum **Sobel-Operator**.
- Prinzipiell ist die **Konvolution** auch durch eine **Korrelation** zu implementieren. Bei asymmetrischen Filterkernen sind dabei Richtungswechsel in den Orientierungswerten zu beachten.

Zusammenfassung (2)

- Ein zweiter Ansatz der Kantenhervorhebung basiert auf der [2. Ableitung](#) der [Bildfunktion](#) und sucht deren [Nulldurchgänge](#). Dies führt zum [Laplace-Operator](#).
- Wegen der Rauschempfindlichkeit des Laplace-Operators ist dessen Kopplung mit einer vorherigen Glättung sinnvoll. Dies führt zum [Laplacian-of-Gaussian-](#) sowie zum [Difference-of-Gaussian-Operator](#).
- Für die [genaue Positionsbestimmung von Nulldurchgängen](#) werden 1. und 2. Ableitung der Bildfunktion gemeinsam analysiert.

Intelligente Sehsysteme

4 Nichtlineare Filter und Filtern von Farbbildern

Rangordnungsfilter, Medianfilter, Diffusionsfilter

RGB- und HSI-Farbmodelle,

Glättung und Kantenhervorhebung von RGB- und HSI-Farbbildern

Henry Hölzemann

- Nichtlineare Filter
 - Rangordnungsfilter
 - Medianfilter
 - Kantenerhaltung beim Medianfilter
 - Diffusionsfilter
 - isotrope homogene Diffusion
 - isotrope inhomogene Diffusion
 - anisotrope Diffusion
- Filterung von Farbbildern
 - Glättung von Farbbildern
 - Kantenhervorhebung in Farbbildern

Rückblick auf lineare Filterung

- Kontrastverbesserung durch **lineare Filterung** erfolgt durch
 - Unterdrückung von **Störsignalen**
(Mittelwertfilter, Gauß-Filter, Binomialfilter)
 - Hervorhebung/Extraktion von **Kantensignalen**
(Sobel-Operator, Laplace-Operator, LoG-Operator, DoG-Operator)
- **LEIDER** führen lineare Operatoren zur **Rauschunterdrückung** auch zur **Kantenunterdrückung**.
- **DAHER** muss i. A. eine bzgl. der Anwendung **angemessene Balance** zwischen **Rauschunterdrückung** und **Kantenhervorhebung** gewählt werden.

Nichtlineare Filterung (1)

- Die bei linearen Filtern notwendige Balance zwischen Rauschunterdrückung und Kantenhervorhebung ist für viele Anwendungen akzeptabel, da diese Balance hinsichtlich der Anwendung angepasst werden kann.
- Entsprechend gehören lineare Filter zum Standardmethodenarsenal des Computersehens und insbes. der Low-Level-Vision bzw. der Bildverarbeitung.

Nichtlineare Filterung (2)

- Für schwierige Anwendungsfälle kann aber eine effizientere Trennung zwischen Rauschunterdrückung und Kantenhervorhebung notwendig sein.
- Zwei Filterklassen können diese Trennung zwischen Rauschunterdrückung und Kantenhervorhebung effizienter gestalten als lineare Filter:
 - Rangordnungsfilter
 - Diffusionsfilter
- Beide Filterklassen gehören zur Klasse der nichtlinearen Filter.

Rangordnungsfilter (1)

Rangordnungsfilter

- werden wie lineare Filter auf ein Pixel p und seine Nachbarpixel angewandt – wir gehen hier zunächst von Nachbarpixeln innerhalb einer um p zentrierten quadratischen Umgebung *ungerader Größen* (3×3 , 5×5 , usw.) aus.
- berechnen im Gegensatz zu linearen Filtern keine gewichteten Summen über den Intensitäten innerhalb der Umgebung, sondern
 - sortieren alle Pixel innerhalb der Umgebung gemäß ihrer Intensitätswerte
 - liefern als Ergebnis den an einem bestimmten Rang der sortierten Reihenfolge eingeordneten Wert

Rangordnungsfilter (2)

- Die Anzahl der Operationen ist für lineare Filter linear in der Anzahl der Pixel der quadratischen Umgebung. Die Zeitkomplexität ist also $O(n)$ für $n=m^2$ und Größe m des Filterkerns.
- Die Zeitkomplexität für Rangordnungsfilter ist wegen der Sortierung dagegen $O(n \log n)$ für $n = m^2$ und Größe m des Filterkerns.
- Für kleine Filterkerne ist dies nicht relevant. Für sehr große Filterkerne und sehr große Bilder kann es zu spürbaren Unterschieden in den Berechnungszeiten kommen.*

* Zumal sich für lineare Filter mit großen Filterkernen noch die FFT als effizienterer Weg anbietet. Die Konvolution eines Bildes der Größe $N \times N$ mit einem Faltungskern der Größe $m \times m$ hat die Zeitkomplexität $O(m^2 \cdot N^2)$. Über FFT ausgeführt ist sie $O(N^2 \cdot \log_2 N)$. Für $\log_2 N < m^2$ ist die Konvolution also über FFT effizienter ausführbar.

Medianfilter (1)

- Der **Median** ist der Wert in der mittleren Position einer sortierten Folge.
- Das **Medianfilter** ist ein Rangordnungsfilter, das als Ergebnis den Median liefert
- Median und Mittelwert einer sortierten Wertefolge sind i. A. ungleich, nähern sich aber für große Folgen normalverteilter Werte einander an
- Daraus folgt die Brauchbarkeit des Medianfilters für die Rauschunterdrückung
- Ist die ungestörte Bildfunktion $I_u(x,y)$ in der Filterumgebung konstant und der Erwartungswert des Rauschens Null, so entspricht der durch den Median angenäherte Erwartungswert der gesuchten ungestörten Bildfunktion.
- Ähnlich wie beim Mittelwertfilter wird die Näherung bei steigender Filtergröße des Medianfilters besser.

Medianfilter (2)

- Das Impulsrauschen (Salt-and-Pepper-Noise) ist ein Spezialfall des Rauschens
- Für das lineare Mittelwertfilter wären sehr große Bereiche nötig, um den Erwartungswert der ungestörten Bildfunktion anzunähern
- Da aber der Großteil der Intensitätswerte beim Impulsrauschen ungestört ist und damit dem Erwartungswert der ungestörten Bildfunktion entspricht, führt die Medianfilterung in homogenen Bereichen zur vollständigen Beseitigung des Impulsrauschen:



Bild mit Impulsrauschen

Mittelwertfilterung

Medianfilterung

Kantenerhaltung beim Medianfilter (1)

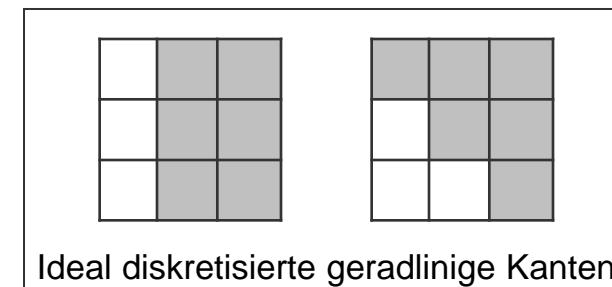
Unter den folgenden Umständen ist das **Medianfilter** kantenerhaltend:

- 1) die **Intensitätswerte** der ungestörten Bildfunktion $I_u(x,y)$
sind **auf beiden Seiten der Kante jeweils konstant**
- 2) der **Signalabstand**, d.h. der Wertunterschied an einer Kante,
ist **größer als die Rauschamplitude**
- 3) die Kante verläuft über die Fläche des Medianfilters **gerade**

Kantenerhaltung beim Medianfilter (2)

Diskussion:

- Das Medianfilter werde an einer Kante k angewandt
- In der sortierten Folge der Pixelintensitäten bezeichnen M_D bzw. M_H die Mengen der Pixel auf der dunkleren bzw. helleren Seite der Kante k
- Gilt Bedingung 1 (konstante Werte auf jeder Seite von k), werden alle Werte aus M_D vor allen Werten aus M_H eingesortiert
- Gilt Bedingung 2 (Kantensignal stärker als Rauschen), bleibt die o.g. Sortierung der Werte aus M_D und M_H auch bei überlagertem Rauschen erhalten
- Gilt Bedingung 3 (gerade Kante), so umfasst M_D mehr Pixel als M_H , wenn das zentrale Pixel zu M_D gehört – und umgekehrt



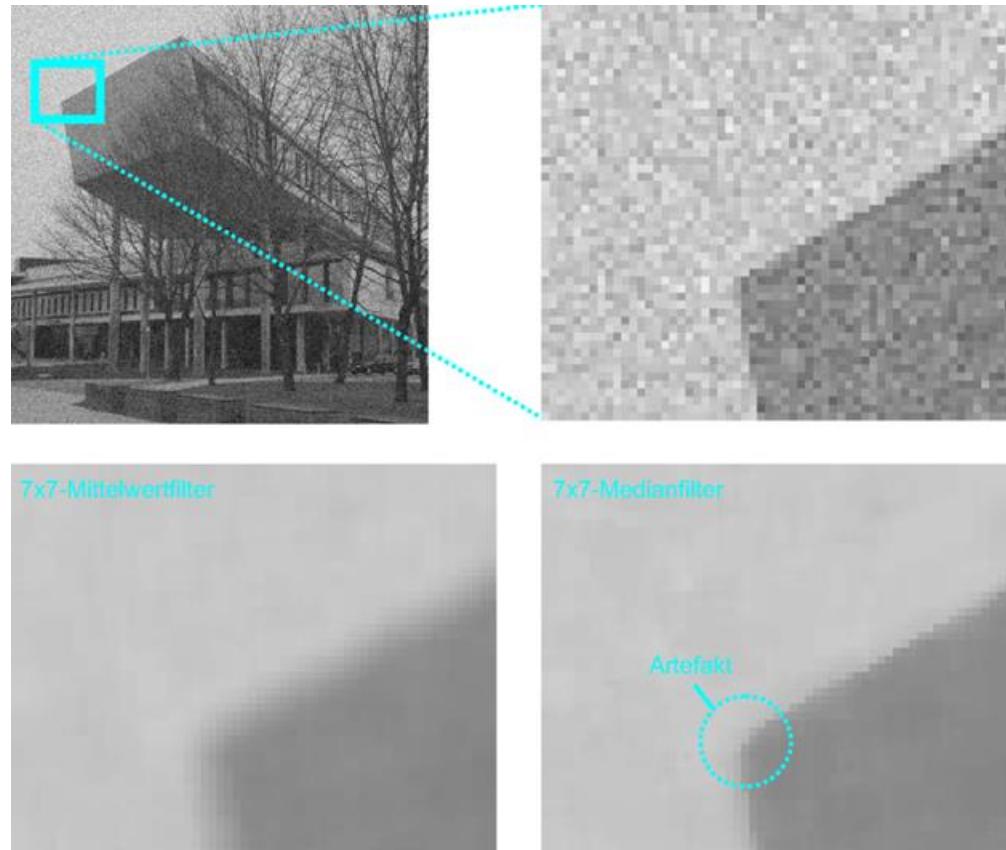
Bemerkung: das Medianfilter ist an einer Kante nicht mehr länger eine gute Näherung des Mittelwertfilters der Pixel einer homogenen Region, da er über Pixel aus zwei unterschiedl. Regionen gebildet wird. Aber genau das passt!

Kantenerhaltung beim Medianfilter (3)

- Von den drei Bedingungen ist die 3. Bedingungen für „Realweltszenen“ am schwierigsten einzuhalten:
 - Kantenzeuge verlaufen selten über längere Strecken geradlinig!
- Wird das Medianfilter jedoch klein genug gewählt, so wird die 3. Bedingungen jedoch in den meisten Bildbereichen angenähert.
 - Die einzige **Ausnahme** sind dann noch **Eckpunkte**:
 - So klein der Filterkern auch gewählt wird, der Kantenverlauf ist an einer Ecke per Definition nicht geradlinig!
 - Somit verursacht das Medianfilter zunächst Artefakte an Ecken.

Kantenerhaltung beim Medianfilter (4)

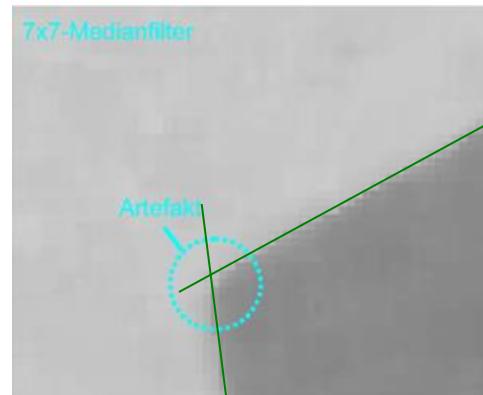
Das Medianfilter erhält Kanten nur, wenn sie innerhalb der Filterumgebung nahezu gerade verlaufen. Das ist an Ecken nicht der Fall:



Kantenerhaltung beim Medianfilter (5)

Die vom Medianfilter an Ecken erzeugten Artefakte sind ggf. (d.h. je nach Interpretation der Artefakte und nach Anwendung) aber auch kompensierbar:

- Wird das Artefakte als Positionsverschiebung der tatsächlich abgebildeten Objektecke interpretiert,
- so kann diese Eckenposition ggf. wieder durch die **Schnittbildung** der **extrahierten Kanten** zurück gewonnen werden:



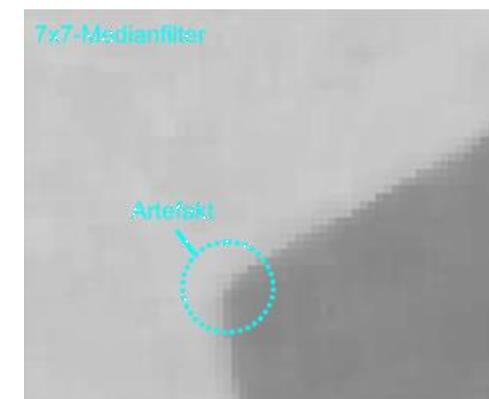
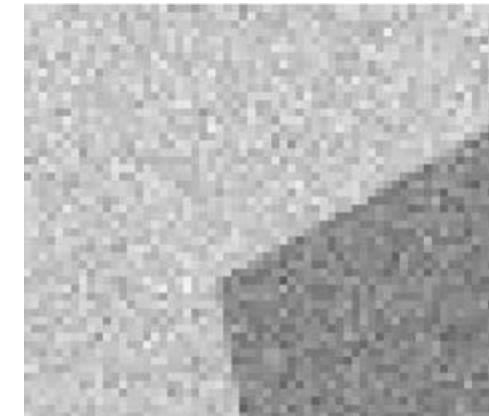
Kantenerhaltung beim Medianfilter (6)

Das Medianfilter wirkt also

- rauschunterdrückend in homogenen Bereichen und
- kantenerhaltend an Kanten.

Die Größe des Medianfilters sollte also eine Balance sein zwischen

- möglichst guter Annäherung an den Erwartungswert der ungestörten Bildfunktion $I_u(x,y)$ durch große Filter und
- einer Maximierung der Anzahl der Kantenpixel, die zu lokal geradlinig verlaufenden Kanten gehören.



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Intelligente Sehsysteme

4 Nichtlineare Filter und Filtern von Farbbildern

Rangordnungsfilter, Medianfilter, Diffusionsfilter

RGB- und HSI-Farbmodelle,

Glättung und Kantenhervorhebung von RGB- und HSI-Farbbildern

Henry Hölzemann

- Nichtlineare Filter
 - Rangordnungsfilter
 - Medianfilter
 - Kantenerhaltung beim Medianfilter
 - Diffusionsfilter
 - isotrope homogene Diffusion
 - isotrope inhomogene Diffusion
 - anisotrope Diffusion
- Filterung von Farbbildern
 - Glättung von Farbbildern
 - Kantenhervorhebung in Farbbildern

Rückblick auf lineare Filterung

- Kontrastverbesserung durch **lineare Filterung** erfolgt durch
 - Unterdrückung von **Störsignalen**
(Mittelwertfilter, Gauß-Filter, Binomialfilter)
 - Hervorhebung/Extraktion von **Kantensignalen**
(Sobel-Operator, Laplace-Operator, LoG-Operator, DoG-Operator)
- **LEIDER** führen lineare Operatoren zur **Rauschunterdrückung** auch zur **Kantenunterdrückung**.
- **DAHER** muss i. A. eine bzgl. der Anwendung **angemessene Balance** zwischen **Rauschunterdrückung** und **Kantenhervorhebung** gewählt werden.

Nichtlineare Filterung (1)

- Die bei linearen Filtern notwendige Balance zwischen Rauschunterdrückung und Kantenhervorhebung ist für viele Anwendungen akzeptabel, da diese Balance hinsichtlich der Anwendung angepasst werden kann.
- Entsprechend gehören lineare Filter zum Standardmethodenarsenal des Computersehens und insbes. der Low-Level-Vision bzw. der Bildverarbeitung.

Nichtlineare Filterung (2)

- Für schwierige Anwendungsfälle kann aber eine effizientere Trennung zwischen Rauschunterdrückung und Kantenhervorhebung notwendig sein.
- Zwei Filterklassen können diese Trennung zwischen Rauschunterdrückung und Kantenhervorhebung effizienter gestalten als lineare Filter:
 - Rangordnungsfilter
 - Diffusionsfilter
- Beide Filterklassen gehören zur Klasse der nichtlinearen Filter.

Rangordnungsfilter (1)

Rangordnungsfilter

- werden wie lineare Filter auf ein Pixel p und seine Nachbarpixel angewandt – wir gehen hier zunächst von Nachbarpixeln innerhalb einer um p zentrierten quadratischen Umgebung *ungerader Größen* (3×3 , 5×5 , usw.) aus.
- berechnen im Gegensatz zu linearen Filtern keine gewichteten Summen über den Intensitäten innerhalb der Umgebung, sondern
 - sortieren alle Pixel innerhalb der Umgebung gemäß ihrer Intensitätswerte
 - liefern als Ergebnis den an einem bestimmten Rang der sortierten Reihenfolge eingeordneten Wert

Rangordnungsfilter (2)

- Die Anzahl der Operationen ist für lineare Filter linear in der Anzahl der Pixel der quadratischen Umgebung. Die Zeitkomplexität ist also $O(n)$ für $n=m^2$ und Größe m des Filterkerns.
- Die Zeitkomplexität für Rangordnungsfilter ist wegen der Sortierung dagegen $O(n \log n)$ für $n = m^2$ und Größe m des Filterkerns.
- Für kleine Filterkerne ist dies nicht relevant. Für sehr große Filterkerne und sehr große Bilder kann es zu spürbaren Unterschieden in den Berechnungszeiten kommen.*

* Zumal sich für lineare Filter mit großen Filterkernen noch die FFT als effizienterer Weg anbietet. Die Konvolution eines Bildes der Größe $N \times N$ mit einem Faltungskern der Größe $m \times m$ hat die Zeitkomplexität $O(m^2 \cdot N^2)$. Über FFT ausgeführt ist sie $O(N^2 \cdot \log_2 N)$. Für $\log_2 N < m^2$ ist die Konvolution also über FFT effizienter ausführbar.

Medianfilter (1)

- Der **Median** ist der Wert in der mittleren Position einer sortierten Folge.
- Das **Medianfilter** ist ein Rangordnungsfilter, das als Ergebnis den Median liefert
- Median und Mittelwert einer sortierten Wertefolge sind i. A. ungleich, nähern sich aber für große Folgen normalverteilter Werte einander an
- Daraus folgt die Brauchbarkeit des Medianfilters für die Rauschunterdrückung
- Ist die ungestörte Bildfunktion $I_u(x,y)$ in der Filterumgebung konstant und der Erwartungswert des Rauschens Null, so entspricht der durch den Median angenäherte Erwartungswert der gesuchten ungestörten Bildfunktion.
- Ähnlich wie beim Mittelwertfilter wird die Näherung bei steigender Filtergröße des Medianfilters besser.

Medianfilter (2)

- Das Impulsrauschen (Salt-and-Pepper-Noise) ist ein Spezialfall des Rauschens
- Für das lineare Mittelwertfilter wären sehr große Bereiche nötig, um den Erwartungswert der ungestörten Bildfunktion anzunähern
- Da aber der Großteil der Intensitätswerte beim Impulsrauschen ungestört ist und damit dem Erwartungswert der ungestörten Bildfunktion entspricht, führt die Medianfilterung in homogenen Bereichen zur vollständigen Beseitigung des Impulsrauschen:



Bild mit Impulsrauschen

Mittelwertfilterung

Medianfilterung

Kantenerhaltung beim Medianfilter (1)

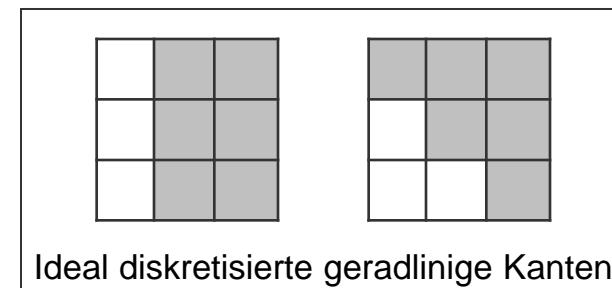
Unter den folgenden Umständen ist das **Medianfilter** kantenerhaltend:

- 1) die **Intensitätswerte** der ungestörten Bildfunktion $I_u(x,y)$
sind **auf beiden Seiten der Kante jeweils konstant**
- 2) der **Signalabstand**, d.h. der Wertunterschied an einer Kante,
ist **größer als die Rauschamplitude**
- 3) die Kante verläuft über die Fläche des Medianfilters **gerade**

Kantenerhaltung beim Medianfilter (2)

Diskussion:

- Das Medianfilter werde an einer Kante k angewandt
- In der sortierten Folge der Pixelintensitäten bezeichnen M_D bzw. M_H die Mengen der Pixel auf der dunkleren bzw. helleren Seite der Kante k
- Gilt Bedingung 1 (konstante Werte auf jeder Seite von k), werden alle Werte aus M_D vor allen Werten aus M_H eingesortiert
- Gilt Bedingung 2 (Kantensignal stärker als Rauschen), bleibt die o.g. Sortierung der Werte aus M_D und M_H auch bei überlagertem Rauschen erhalten
- Gilt Bedingung 3 (gerade Kante), so umfasst M_D mehr Pixel als M_H , wenn das zentrale Pixel zu M_D gehört – und umgekehrt



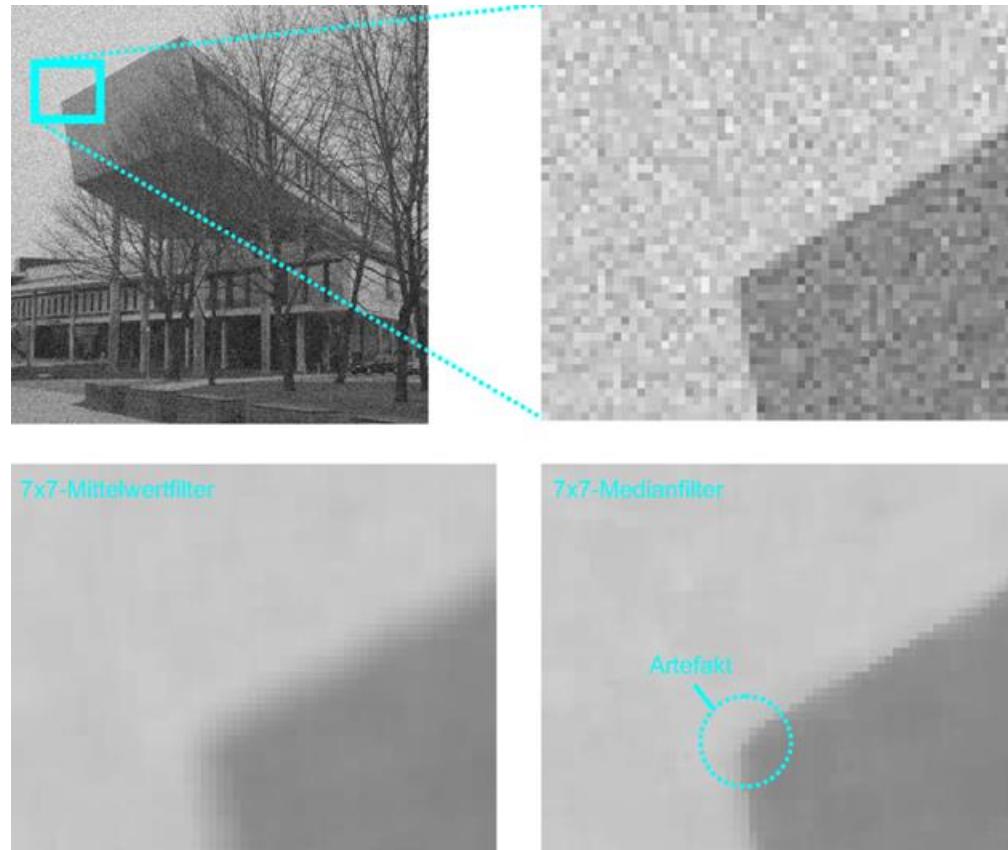
Bemerkung: das Medianfilter ist an einer Kante nicht mehr länger eine gute Näherung des Mittelwertfilters der Pixel einer homogenen Region, da er über Pixel aus zwei unterschiedl. Regionen gebildet wird. Aber genau das passt!

Kantenerhaltung beim Medianfilter (3)

- Von den drei Bedingungen ist die 3. Bedingungen für „Realweltszenen“ am schwierigsten einzuhalten:
 - Kantenzeuge verlaufen selten über längere Strecken geradlinig!
- Wird das Medianfilter jedoch klein genug gewählt, so wird die 3. Bedingungen jedoch in den meisten Bildbereichen angenähert.
 - Die einzige **Ausnahme** sind dann noch **Eckpunkte**:
 - So klein der Filterkern auch gewählt wird, der Kantenverlauf ist an einer Ecke per Definition nicht geradlinig!
 - Somit verursacht das Medianfilter zunächst Artefakte an Ecken.

Kantenerhaltung beim Medianfilter (4)

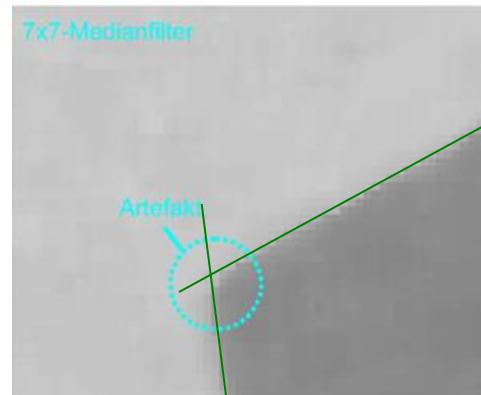
Das Medianfilter erhält Kanten nur, wenn sie innerhalb der Filterumgebung nahezu gerade verlaufen. Das ist an Ecken nicht der Fall:



Kantenerhaltung beim Medianfilter (5)

Die vom Medianfilter an Ecken erzeugten Artefakte sind ggf. (d.h. je nach Interpretation der Artefakte und nach Anwendung) aber auch kompensierbar:

- Wird das Artefakte als Positionsverschiebung der tatsächlich abgebildeten Objektecke interpretiert,
- so kann diese Eckenposition ggf. wieder durch die **Schnittbildung** der **extrahierten Kanten** zurück gewonnen werden:



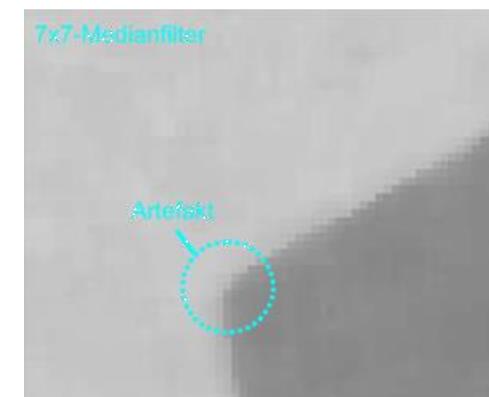
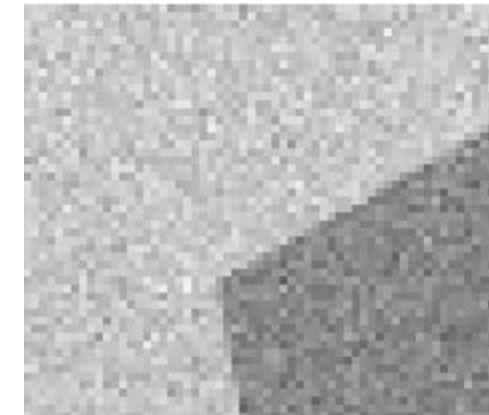
Kantenerhaltung beim Medianfilter (6)

Das Medianfilter wirkt also

- rauschunterdrückend in homogenen Bereichen und
- kantenerhaltend an Kanten.

Die Größe des Medianfilters sollte also eine Balance sein zwischen

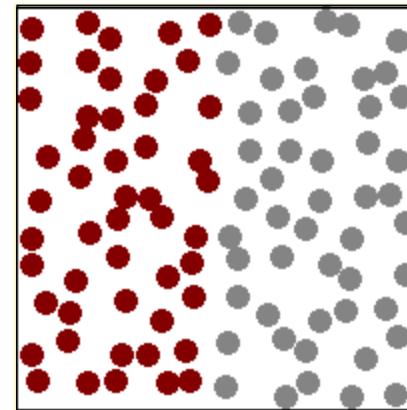
- möglichst guter Annäherung an den Erwartungswert der ungestörten Bildfunktion $I_u(x,y)$ durch große Filter und
- einer Maximierung der Anzahl der Kantenpixel, die zu lokal geradlinig verlaufenden Kanten gehören.



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Diffusionsfilter (1)

Die Idee der Diffusionsfilter basiert auf der [Simulation des physikalischen Diffusionsprozesses](#):



Diffusionsfilter (2)

Die Idee der Diffusionsfilter basiert auf der [Simulation des physikalischen Diffusionsprozesses](#):

- die Bildfunktion $I(x,y)$ wird als Konzentration einer Flüssigkeit interpretiert
- der Diffusionsprozess gleicht Konzentrationsvariationen, also eine inhomogen verteilte Konzentration, über die Zeit aus

Der Ansatz der Diffusionsfilter stammt von Perona und Malik (1990)*

* P. Perona, J. Malik (1990): Scale-Space and Edge Detection Using Anisotropic Diffusion. IEEE PAMI, 12(7).

Diffusionsfilter (3)

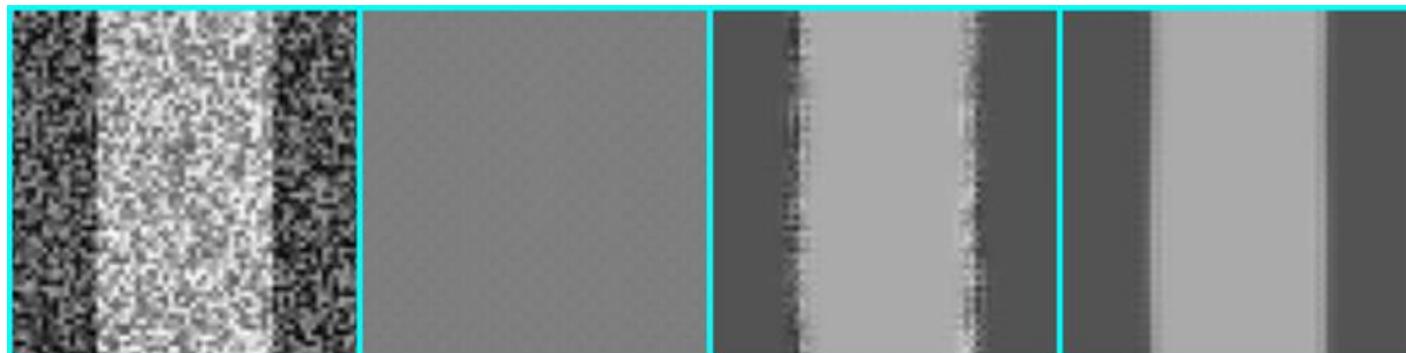
Es gibt drei verschiedene Diffusionsmodelle:

- 1) Isotrope homogene Diffusion führt den gleichmäßigen Ausgleich in alle Richtungen durch
- 2) Isotrope inhomogene Diffusion macht das Ausmaß der Diffusion vom Vorhandensein einer Kante in der Nachbarschaft ab
- 3) Anisotrope Diffusion findet auch an Kanten statt, aber nur parallel zur Kantenrichtung und nicht über die Kante hinaus

Diffusionsfilter (3)

Beispiel für die drei verschiedene Diffusionsmodelle (nach 500 Iterationen) an einem verrauschten Bild (links mit $SNR_{max} = 0,25:1$) :

- 1) Die **isotrope homogene Diffusion** löscht alle Kanten vollständig.
- 2) Die **isotrope inhomogene Diffusion** erhält den Unterschied zw. Vordergrund und Hintergrund durch Kantenbegrenzung, zeigt aber verrauschte Kanten.
- 3) Die **anisotrope Diffusion** erfolgt nur parallel zur Kantenrichtung und nicht über die Kante hinaus. Daher bleibt der Unterschied zwischen Vordergrund und Hintergrund erhalten *und* das Rauschen an den Kanten wird unterdrückt.



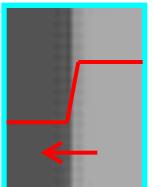
Diffusionsfilter (4)

- Die Diffusionsrate ist von einem *Diffusionskoeffizienten* D abhängig, der sich im mehrdimensionalen Fall als Diffusionstensor \mathbf{D} darstellt.*
- Der *Diffusionstensor* \mathbf{D} spezifiziert die Diffusionsrate in Abhängigkeit von der Richtung eines *Konzentrationsgradienten* ∇u .
- Soll durch die Diffusion das Rauschen in Bildern unterdrückt werden, so ist der Konzentrationsgradient der Intensitätsgradient der Bildfunktion $I(x,y)$ und der Diffusionstensor \mathbf{D} eine 2×2 -Matrix.
- Nach dem 1. *Fickschen Gesetz* ist der *Fluss* \vec{j} (bzw. die *Teilchenstromdichte*) proportional zum negativen Konzentrationsgradienten:

$$(1) \quad \vec{j}(x,y) = -\mathbf{D}(x,y) \times \nabla u(x,y) = -\boxed{\mathbf{D}(x,y)} \times \begin{pmatrix} \frac{\partial I(x,y)}{\partial x} \\ \frac{\partial I(x,y)}{\partial y} \end{pmatrix}$$

Diffusionskoeffizient

Konzentrationsgradient



* Ein Tensor ist eine multilinear Abbildung, d.h. eine Abbildung, die in jeder Variablen linear ist.

Diffusionsfilter (5)

- Der Fluss $\vec{j}(x,y,t)$ ändert sich über die Zeit t und verursacht zur Zeit t an der Stelle (x,y) eine Konzentrationsänderung:

$$\frac{\partial u(x,y,t)}{\partial t} = -\operatorname{div} \vec{j}(x,y,t) \quad (2)$$

- Der Divergenzoperator div
 - ist ein Differentialoperator
 - beschreibt anschaulich für jeden Raumpunkt p : „wie viel mehr fließt aus einer Umgebung von p hinaus als in sie hinein“
 - \rightsquigarrow Quellen zeigen eine Divergenz größer als Null
 - \rightsquigarrow Senken zeigen eine Divergenz kleiner als Null hat
 - Bei Divergenz gleich Null bezeichnet man das Feld als quellenfrei

Diffusionsfilter (6)

- Genauer: die Divergenz des Flusses \vec{j} an einem Punkt p ist der Grenzwert des Flusses über den Rand $\rho(R)$ einer den Punkt p umgebenden Region R (normiert durch die Größe $|R|$ von R) für das Schrumpfen von R auf den Punkt p :

$$\text{div } \vec{j}(p) = \lim_{|R| \rightarrow 0} \left(\frac{1}{|R|} \int_{\rho(R)} \vec{j} \cdot \vec{n} dS \right). \quad (3)$$

R ist eine beliebige Region (z.B. eine 3D-Kugel) mit Volumen $|R|$. Die Integration erfolgt über den Rand $\rho(R)$ von R . Vektor \vec{n} ist die nach außen gerichtete Normale und dS das zugehörige Flächenelement.

- Bei Spezialisierung auf infinitesimale Würfel erhält man eine Darstellung in kartesischen Koordinaten – hier 2-dimensional mit (x,y,t) für p :

$$\text{div } \vec{j}(x, y, t) = \left(\frac{\partial j_x(x, y, t)}{\partial x} + \frac{\partial j_y(x, y, t)}{\partial y} \right). \quad (4)$$

Dabei sind j_x und j_y die beiden Richtungskomponenten des Flussvektors.

Diffusionsfilter (7)

Die **Simulation** des Diffusionsprozesses

- basiert auf sukzessiven Berechnungen von Konzentrationen $u(x,y,t)$ für Zeiten t
- ausgehend von einer Anfangskonzentration $u(x,y,0)$, die dem zu bearbeitenden Bild entspricht: $u(x,y,0) = I(x,y)$.

ABER: für beliebige Diffusionstensoren ist dies nicht analytisch lösbar.

Daher: *iterative Schätzung* der Konzentration zur Zeit t_{i+1} aus der zur Zeit t_i :

$$u(x, y, t_{i+1}) = u(x, y, t_i) + (t_{i+1} - t_i) \frac{\partial u(x, y, t_i)}{\partial t}. \quad (5)$$

Die Schätzung geht von einer über dem Zeitintervall $(t_{i+1} - t_i)$ konstanten Rate $\partial u / \partial t$ aus und wird nur für Positionen (x,y) auf dem Pixelraster geschätzt.

Zusammenfassung: Alle Schritte der Diffusionsfilter

$$\rightarrow \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$$

- 1) Approximiere Intensitätsgradienten $\partial I(x,y,t)/\partial x$ und $\partial I(x,y,t)/\partial y$ durch Differenzen der Intensitäten in x- bzw. y-Richtung, z.B.: $I(x+1,y) - I(x-1,y)$, $I(x,y+1) - I(x,y-1)$

Noch zu klären!

Abhängig vom Modell des Diffusionsfilters

- 2) Berechne Diffusionstensor D

- 3) Rechne Fluss $\vec{j}(x,y,t) = (j_x(x,y,t), j_y(x,y,t)) = -D(x,y,t) \times \begin{pmatrix} \frac{\partial I(x,y)}{\partial x} \\ \frac{\partial I(x,y)}{\partial y} \end{pmatrix}$ (1)

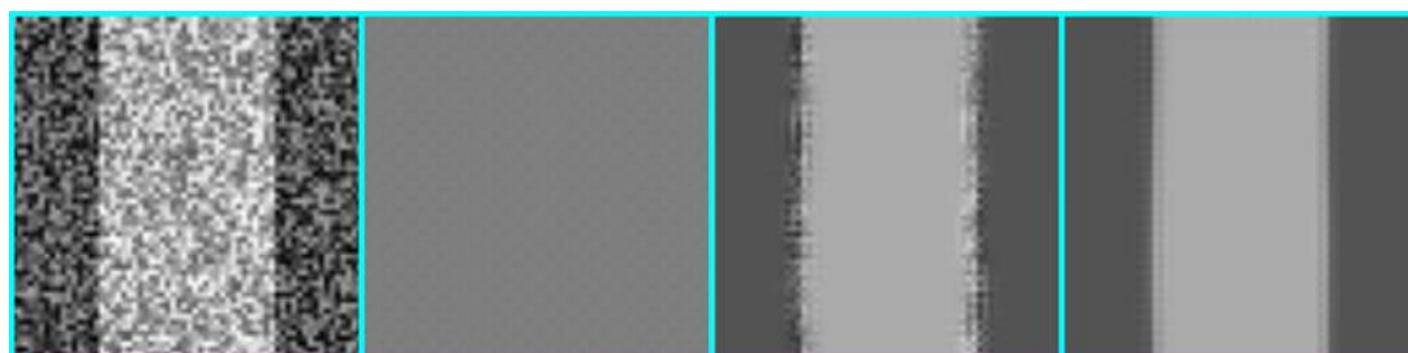
- 4) Approximiere Flussgradienten $\partial j_x(x,y,t)/\partial x$ und $\partial j_y(x,y,t)/\partial y$ durch Differenzen des Fluxes in x- bzw. y-Richtung, z.B.: $j_x(x+1,y) - j_x(x-1,y)$, $j_y(x,y+1) - j_y(x,y-1)$

- 5) Rechne Divergenz $\text{div } j(x,y,t) = \partial j_x(x,y,t)/\partial x + \partial j_y(x,y,t)/\partial y$ (4)

- 6) $I(x,y,t+1) = I(x,y,t) - \text{div } j(x,y,t) = I(x,y,t) - (\partial j_x(x,y,t)/\partial x + \partial j_y(x,y,t)/\partial y)$ (2),(5)

Diffusionsmodelle

- Verschied. Diffusionsfilter unterscheiden sich durch die Diffusionstensoren.
- Wir betrachten die drei genannten Diffusionsmodelle:
 - 1) isotrope homogene Diffusion mit gleichmäß. Ausgleich in alle Richtungen
 - 2) isotrope inhomogene Diffusion mit unterschiedl. Ausmaß der Diffusion
 - 3) anisotrope Diffusion, die nur parallel zur Kantenrichtung und nicht über Kante hinaus erfolgt



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Isotrope homogene Diffusion (1)

$$\curvearrowright \begin{pmatrix} \varepsilon_0 & 0 \\ 0 & \varepsilon_0 \end{pmatrix}$$

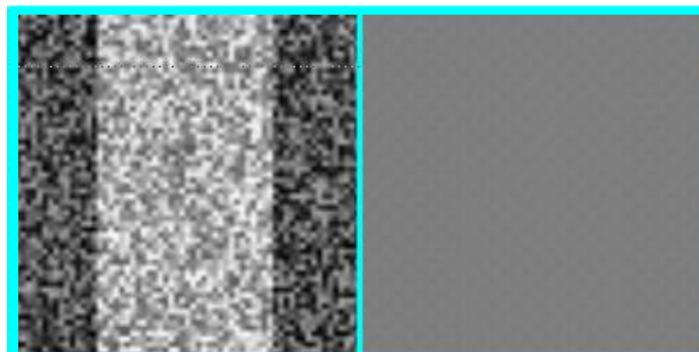
- Für die isotrope homogene Diffusion (Variante 1) ist \mathbf{D} die mit einem Skalar ε_0 multiplizierte Identitätsmatrix. Die Intensitätsveränderung $\partial u / \partial t$ ist also

$$\partial u(x,y,t) / \partial t = \varepsilon_0 (\partial u / \partial x + \partial u / \partial y). \quad (6)$$

- Die Intensitätsveränderung erfolgt also über einen konstanten Tensor und hängt nur vom lokalen Intensitätsgradienten ab!
- Hierfür gibt es eine analytische Lösung:
 - ☞ danach ist die Diffusion einer Anfangsverteilung $I(x,y)$ nach der Zeit t das Ergebnis einer Konvolution mit einer Gauß-Funktion mit $\sigma = (2t)^{1/2}$.

Isotrope homogene Diffusion (2)

- Erwartungsgemäß führt die isotrope homogene Diffusion daher bei hinreichender Zahl von Iterationen zum vollständigen Ausgleich der Intensitätswerte.
- Das Resultat ist ein geglättetes Bild mit unscharfen (ggf. nicht mehr erkennbaren) Kanten:



Isotrope homogene Diffusionsfilterung eines verrauschten Bildes nach 500 Iterationen.

Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

- Da die isotrope homogene Diffusion also der Gauß-Filterung entspricht, stellt sie keine Neuerung für die Bildverbesserung dar.

Isotrope inhomogene Diffusion (1)

Die **isotrope inhomogene Diffusion** (Variante 2) variiert den Diffusions-tensor lokal

- ~ Damit kann ein unterschiedliches Verhalten für Kanten und für abgebildete Objektflächen angesteuert werden
- ~ Im Idealfall
 - erwirkt die Diffusion den Ausgleich von Störungen auf abgebildeten homogenen Objektflächen
 - während an Kanten keine Diffusion statt findet

Isotrope inhomogene Diffusion (2)

Aber: der Verlauf von Kanten ist i.A. nicht a priori bekannt

↪ Daher sind Kanten für jede Zeit t zu schätzen – und zwar durch den Gradienten

$$\nabla u(x,y,t) = (\partial u / \partial x, \partial u / \partial y)$$

↪ je größer der Gradientenbetrag ist, desto geringer sollte die Diffusion sein

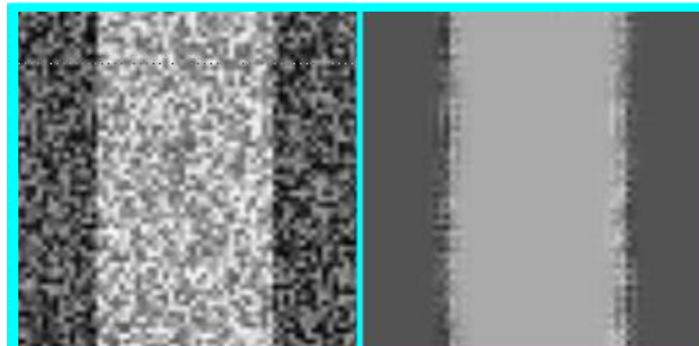
↪ dazu muss der Diffusionstensor den Gradientenbetrag berücksichtigen:

$$D = \begin{pmatrix} \varepsilon(\|\nabla u\|^2) & 0 \\ 0 & \varepsilon(\|\nabla u\|^2) \end{pmatrix} \text{ mit } \varepsilon(\|\nabla u\|^2) = \varepsilon_0 \frac{\lambda^2}{\|\nabla u\|^2 + \lambda^2}. \quad (7)$$

- Der Parameter λ steuert die Abnahme der Diffusion bei steigendem Gradienten:
 - großes λ : die inhomogene Diffusion nähert sich der homogenen Diffusion an
 - kleines λ : Diffusion sinkt rasch in Gebieten mit hohen Gradienten

Isotrope inhomogene Diffusion (3)

- Die isotrope inhomogene Diffusion ist nicht mehr analytisch lösbar, sondern erfordert die genannte iterative Variante (s. Gleichung 5).
- Für die Anwendung der isotropen inhomogenen Diffusion auf ein Bild mit Kanten sollte bei gut gewähltem λ gelten:
 - der Intensitätsausgleich und damit die Rauschunterdrückung erfolgt nur in den die Objektflächen abbildenden Bildregionen
 - hinreichend stark ausgeprägte Kanten bleiben dabei erhalten

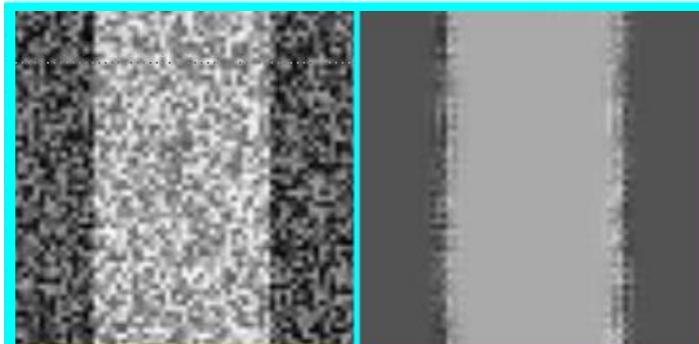


Isotrope inhomogene Diffusionsfilterung eines verrauschten Bildes nach 500 Iterationen.

Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Isotrope inhomogene Diffusion (4)

- Wie das Medianfilter führt die isotrope inhomogene Diffusion also zu einer Rauschunterdrückung auf abgebildeten homogenen Objektflächen und zu einer Erhaltung der abgebildeten Objektkanten.
- Dafür muss aber λ hinreichend klein gewählt werden, um die Diffusion an den Kanten zu reduzieren!
- Damit wird das Glätten der Kanten verhindert, aber auch das Rauschen nahe der Kanten nicht hinreichend unterdrückt.

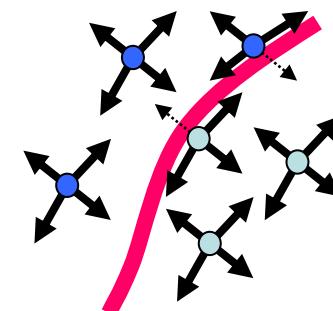


Isotrope inhomogene Diffusionsfilterung eines verrauschten Bildes nach 500 Iterationen.

Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Anisotrope inhomogene Diffusion (1)

- Bei der isotropen inhomogenen Filterung verblieb das Rauschen an den Kanten, weil die **richtungsunabhängige** Diffusion in Abhängigkeit der Gradienten an den Kanten gedämpft wurde.
- Dies Rauschen an den Kanten kann aber unterdrückt werden, wenn die Diffusion in Abhängigkeit vom Gradienten **nur in bestimmte Richtungen** zugelassen wird – nämlich bei Kanten nur **parallel** zur **Kantenrichtung!**
- Der dazu nötige Diffusionstensor wird aus einer Eigenwertzerlegung definiert. Diffusionstensoren sind positiv-definite, symmetrische Matrizen und haben daher zwei verschiedene Eigenwerte λ_1 und λ_2 .



Bildquelle: Klaus Tönnies:
Grundlagen der Bildverar-
beitung, Pearson Studium,
2005.

Anisotrope inhomogene Diffusion (2)

- Eigenwertzerlegung des Diffusionstensors:

$$\mathbf{D} = \begin{pmatrix} e_{1,1} & e_{2,1} \\ e_{1,2} & e_{2,2} \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} e_{1,1} & e_{1,2} \\ e_{2,1} & e_{2,2} \end{pmatrix} \quad (8)$$

mit Eigenwerten λ_1 und λ_2 und Eigenvektoren $(e_{1,1}, e_{1,2})^T$ bzw. $(e_{2,1}, e_{2,2})^T$

- ~ Die Eigenwerte λ_1 und λ_2 spezifizieren die Stärke der Diffusion
- ~ Zur Blockierung der Diffusion über Kanten muss einer der beiden Eigenvektoren in Gradientenrichtung zeigen und einen niedrigen Eigenwert haben
- ~ Der zweite Eigenvektor, der orthogonal zum ersten Eigenvektor ist, verläuft parallel zur Kante. Seine Diffusionsstärke soll unabhängig von der Kantenstärke sein und zeigt entsprechend einen konstanten Eigenwert

Anisotrope inhomogene Diffusion (3)

- Die folg. Eigenvektoren und -werte erfüllen die genannten Bedingungen:

$$\begin{pmatrix} e_{1,1} & e_{1,2} \end{pmatrix} = \frac{\nabla u}{\|\nabla u\|}, \quad \lambda_1 = \varepsilon \left(\|\nabla u\|^2 \right) = \varepsilon_0 \frac{\lambda^2}{\|\nabla u\|^2 + \lambda^2}, \quad (9)$$

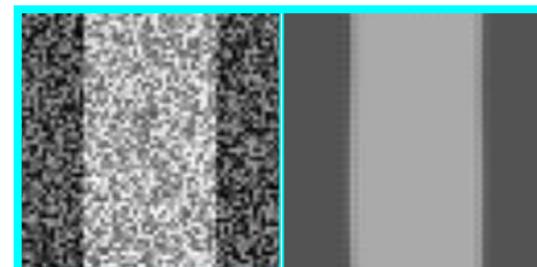
$$\begin{pmatrix} e_{2,1} & e_{2,2} \end{pmatrix} = \begin{pmatrix} e_{1,2} & -e_{1,1} \end{pmatrix}, \quad \lambda_2 = 1.$$

λ₁, λ₂ Eigenwerte
λ = Steuerparameter

- Erster Eigenvektor in Richtung des Gradienten und Eigenwert sinkt mit Gradientenstärke: keine Diffusion über Kanten hinweg.
- Zweiter Eigenvektor parallel zur Kante und Eigenwert konstant: Diffusion parallel zur Kante ist von der Kantenstärke unabhängig.

Anisotrope inhomogene Diffusion (4)

- Die anisotrope Diffusion ist wie die isotrope inhomogene Diffusion iterativ zu lösen (s. Gleichung 5).
- Die anisotrope Diffusion findet auch an Kantenpositionen (x,y) statt, wenn parallel zur Kantenrichtung Intensitätsunterschiede (dann $\nabla u(x,y) \neq 0$) vorliegen.
- Intensitätsunterschiede orthogonal zur Kantenrichtung gehen durch $\varepsilon(\|\nabla u\|^2)$ kaum ins Ergebnis ein.

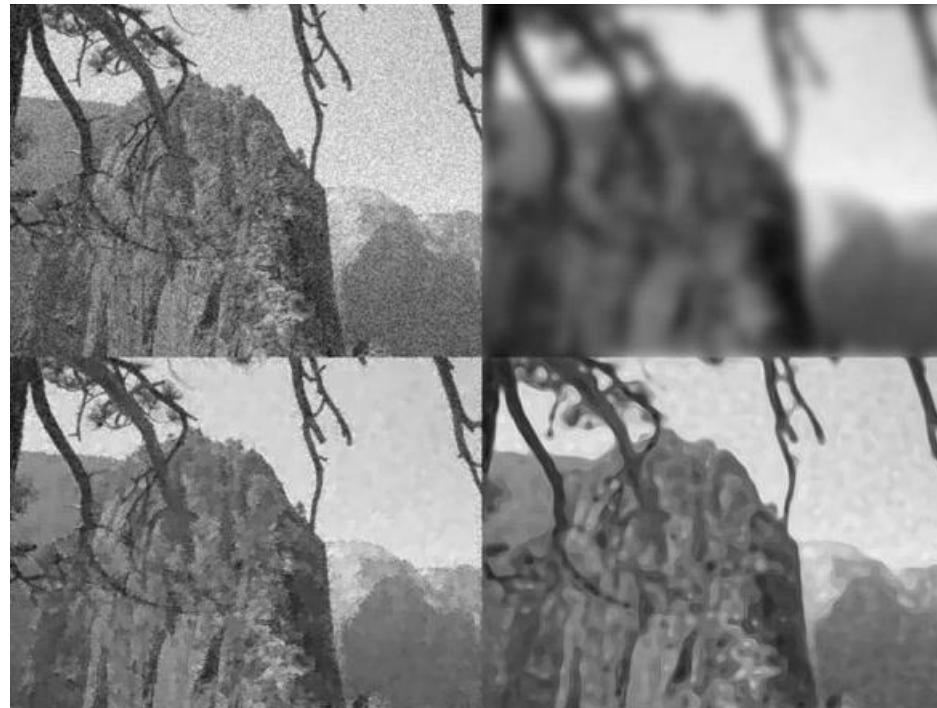


Anisotrope Diffusionsfilterung nach 500 Iterationen.

Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Anisotrope inhomogene Diffusion (5)

Verrausches Bild (oben links), homogene Diffusion (oben rechts), isotrope inhomogene Diffusion (unten links) und anisotrope Diffusion (unten rechts) nach 20 Iterationen:



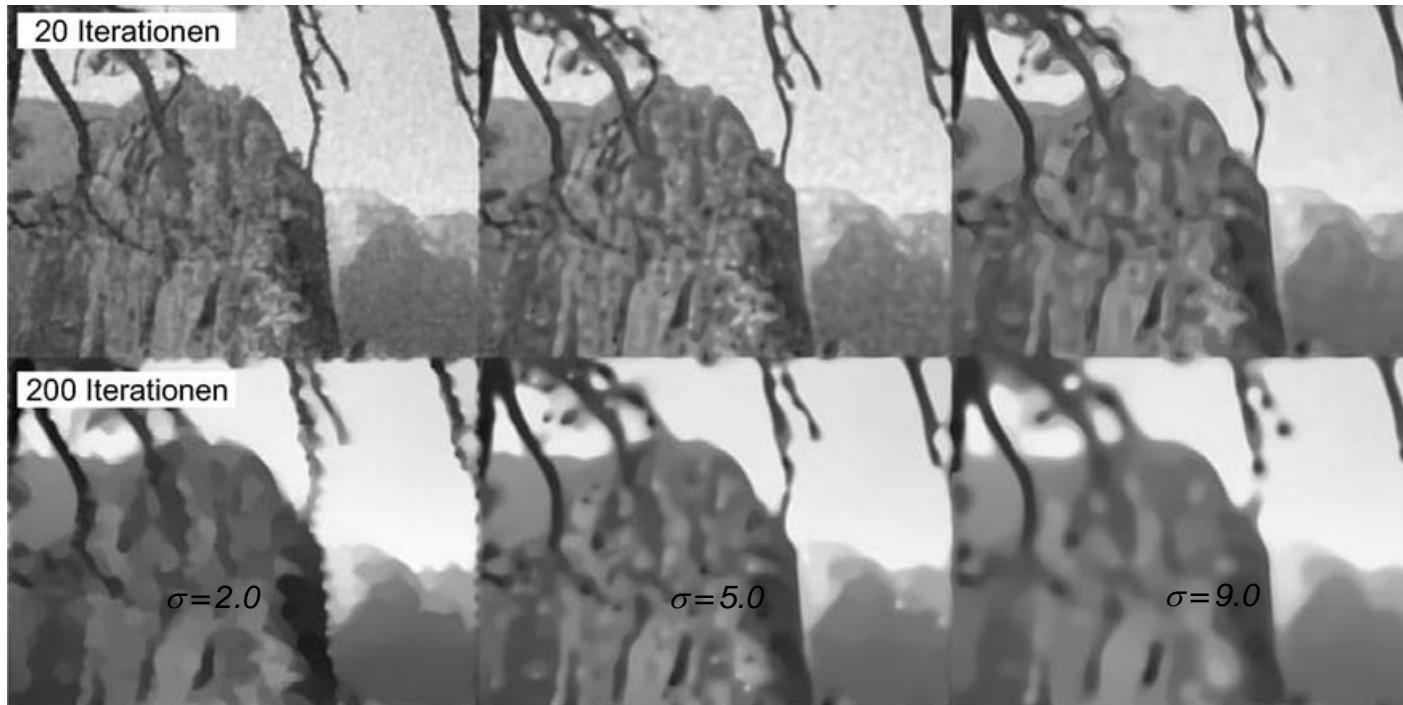
Anisotrope inhomogene Diffusion (6)

- Auch die anisotrope Diffusion wird nach vielen Iteration zu einer Kantenglättung, da der Gewichtungsterm $\varepsilon(\|\nabla u\|^2)$ nie vollständig gleich Null sein wird.
- Dem kann durch eine [Entkopplung der Modelle von Rauschen und Kanten](#) begegnet werden:
 - der Konzentrationsgradient ∇u wird durch einfache Differenzen abgeschätzt,
 - Im Diffusionstensor \mathbf{D} wird anstelle von ∇u in Gleichung (9) ein Kantengradient ∇k eingeführt, der durch Kombination eines Differenzfilters mit einem glättenden Gauß-Filter angenähert wird.
- Durch entsprechende Wahl vom σ des Gauß-Filters werden alle Kanten, die kleiner als die Filtergröße sind, als Rauschen betrachtet und durch Diffusion aufgelöst.
- Das σ des Gauß-Filters trennt also zwischen feinem Rauschen in hohem Frequenzbereich und deutlichen Kanten in niedrigerem Frequenzbereich.

Anisotrope Diffusion (6)

Anisotrope Diffusion für unterschiedliche Filtergrößen σ zur Bestimmung der Kantengradienten k .

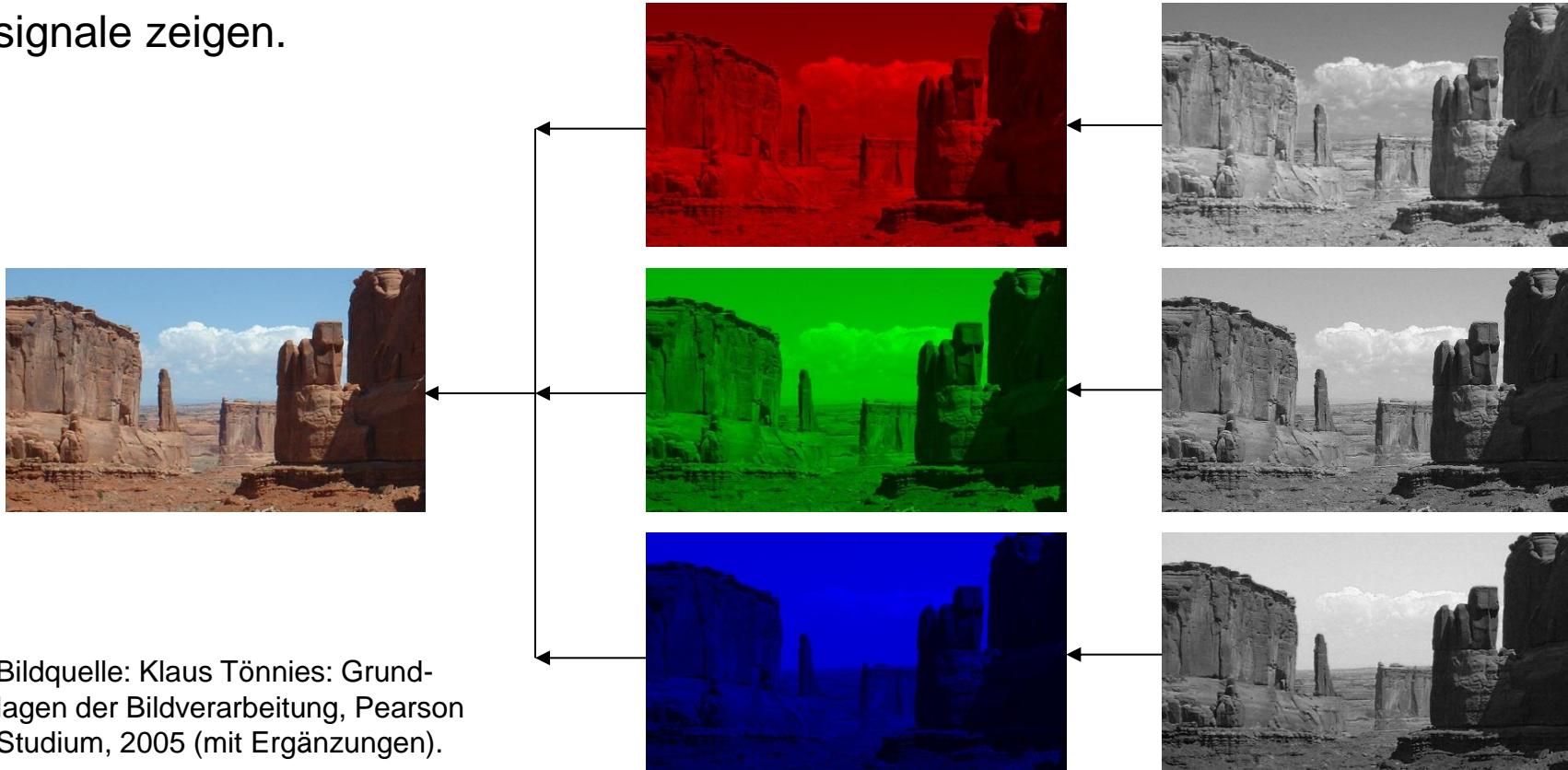
Die Filterung bewirkt, dass Details, die kleiner als die Filtergröße σ sind, als Rauschen betrachtet und durch Diffusion aufgelöst werden.



Filterung von Farbbildern (1)

Die verschiedenen Kanäle von Mehrkanalbildern zeigen i.A. verschiedene Intensitätswerte.

Für RGB-Farbbilder können also $I_R(x,y)$, $I_G(x,y)$ und $I_B(x,y)$ grundsätzlich unterschiedliche Intensitätswerte und damit auch unterschiedliche Stör- und Kanten- signale zeigen.

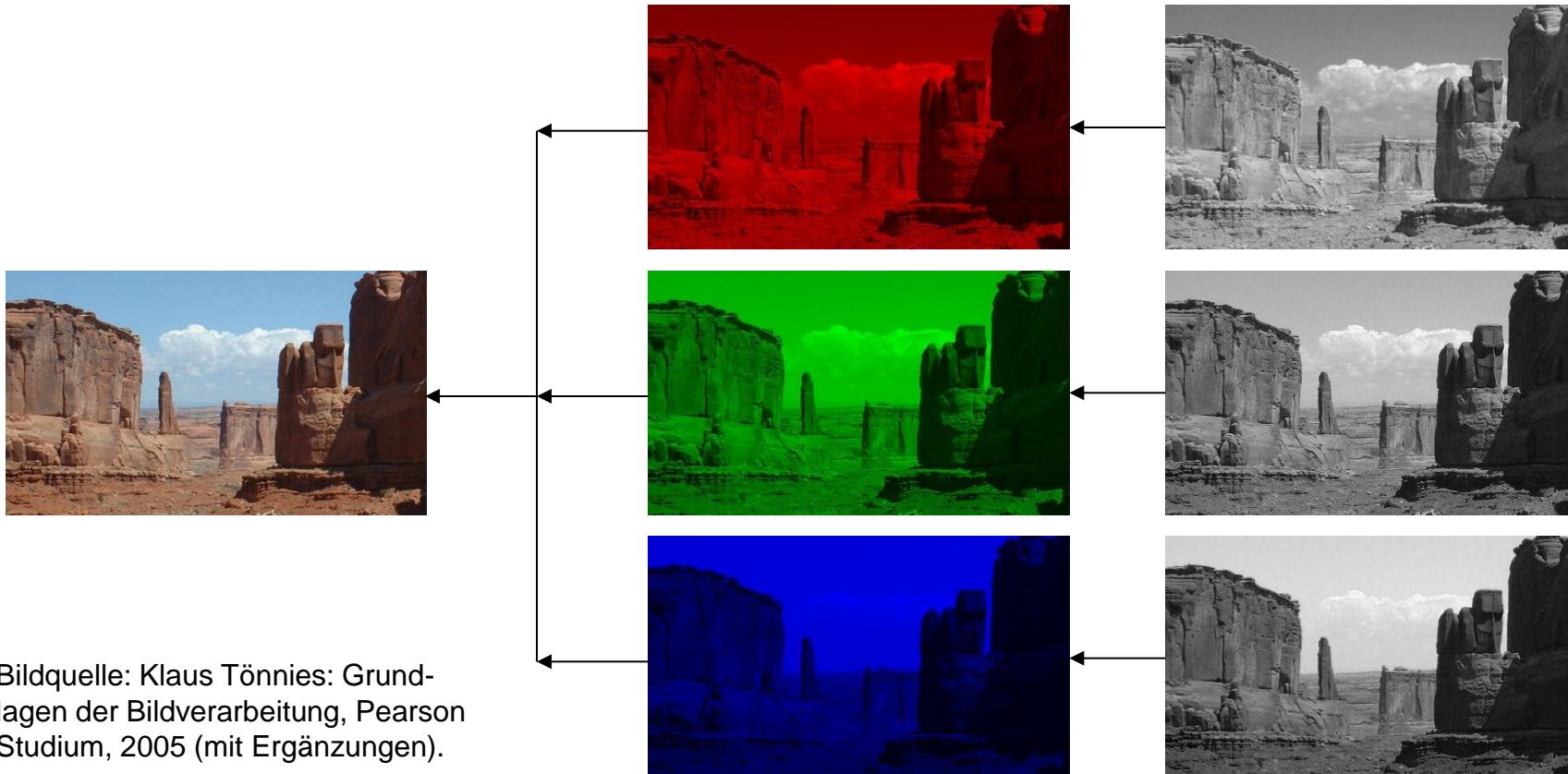


Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005 (mit Ergänzungen).

Filterung von Farbbildern (2)

In der Abbildung sind z.B. die Konturen der Wolken gegenüber dem Himmel im Blaukanal deutlich schwächer ausgeprägt.

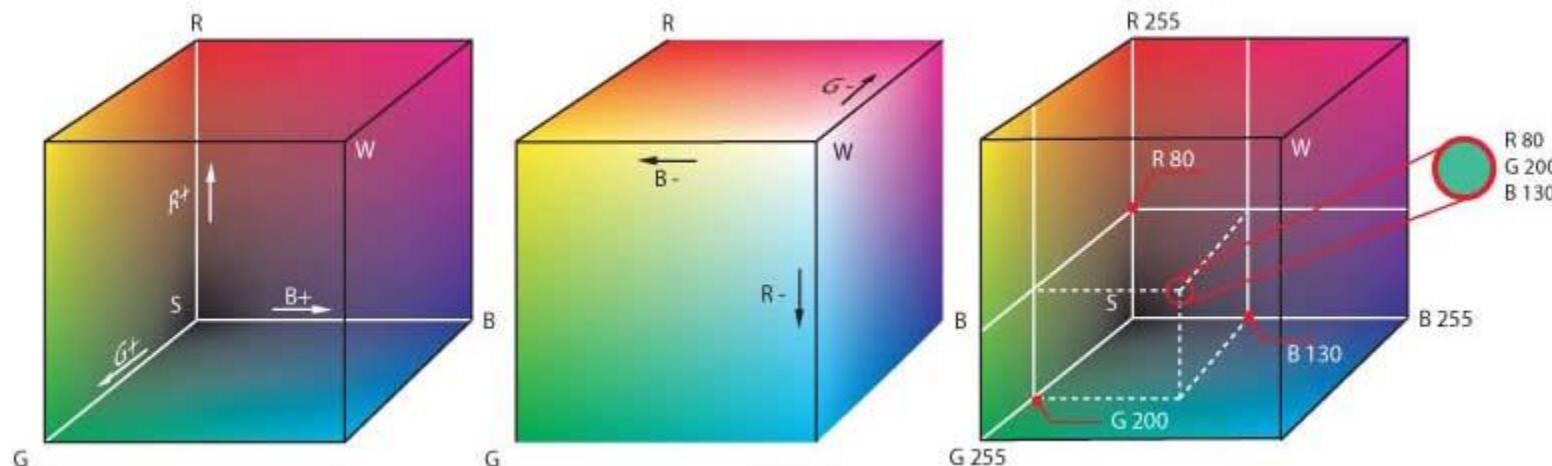
Damit stellt sich die Frage, in welcher Art Glättungs- und Kantenfilter auf Mehrkanalbildern, hier für RGB-Bilder, anzuwenden sind.



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005 (mit Ergänzungen).

Das RGB-Farbmodell

- Grundsätzlich gibt es verschiedene Farbmodelle, um die in Farbbildern enthaltene Information zu kodieren.
- Das RGB-Farbmodell ist ein additives Farbmodell, das Farbwahrnehmungen durch das additive Mischen der drei Grundfarben *Rot*, *Grün*, *Blau* nachbildet.
- Sind ein Rot, ein Grün und ein Blau in maximaler Intensität definiert, so kann eine bestimmte Farbe F durch ihren Rotanteil R, ihren Grünanteil G und ihren Blauanteil B beschrieben werden: $I_{RGB}(x,y) = (I_R(x,y), I_G(x,y), I_B(x,y))$.



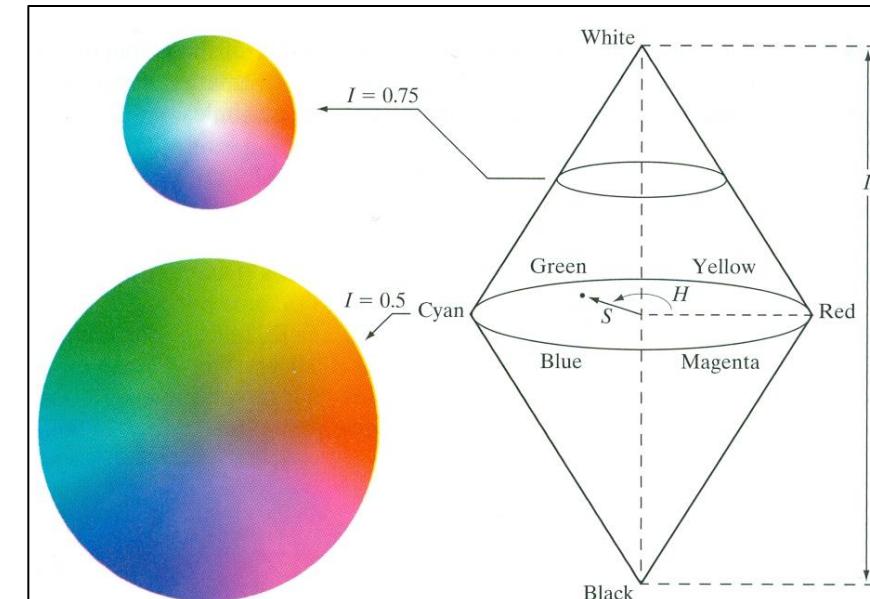
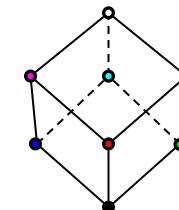
HSI-Farbmodell (1)

Das HSI-Farbmodell wird im Computersehen sowie in der Farbphotographie und Videotechnik verwendet und repräsentiert Farbbilder durch drei Größen:

- **H** für **hue** (Färbung, Tönung) $\in [0, 2\pi)$ *
- **S** für **saturation** (Sättigung, Chroma) $= 1 - 3 \cdot \min\{R, G, B\} / (R+G+B)$; S = 0 für Schwarz
- **I** für **intensity** (Intensität, Helligkeit) $= (R+G+B)/3$

Das HSI-Farbmodell entkoppelt also die Intensitätsinformation von der Farbinformation (Färbung und Sättigung)!

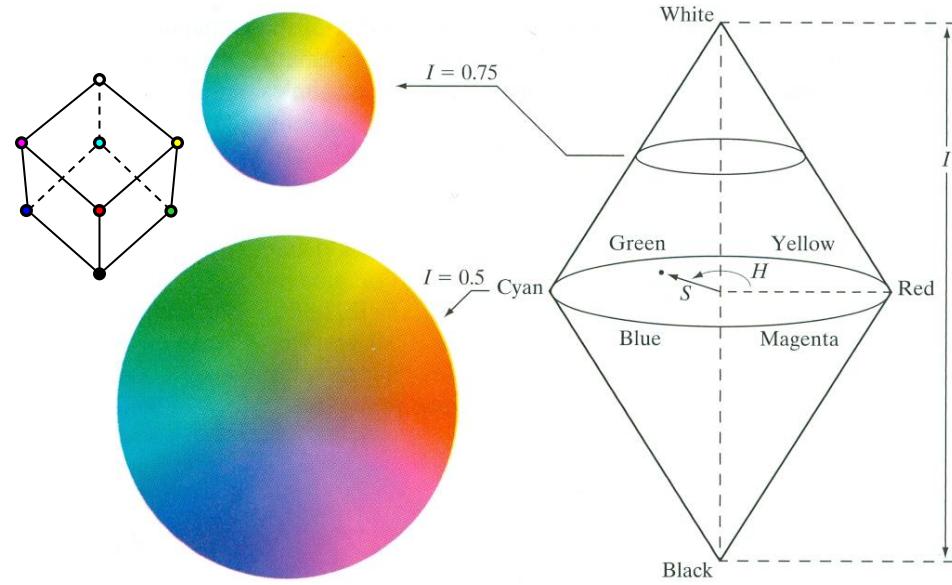
Ferner entspricht das HSI-Farbmodell besser der Farbempfindung



* s. Anhang für Konvertierungen

HSI-Farbmodell (2)

Color	R	G	B	H	s	i
Black	0	0	0	n.a.	0	0
Red	255	0	0	0	1	0.33
Green	0	255	0	120	1	0.33
Blue	0	0	255	240	1	0.33
Yellow	255	255	0	60	1	0.66
White	255	255	255	n.a.	0	1



R, G, B in [0, 255], H in [0, 360], s, i in [0, 1]

Bildquelle: R. C. Gonzales, R. E. Woods: Digital Image Processing (3rd Ed.). Pearson Ed. Int. 2008

- Der HSI-Farbraum zeigt eine vertikale Intensitätsachse sowie die Längen (Sättigungen) und Winkel (Farbtöne) zur Rotachse der Vektoren zu den Farbpunkten.
- Die Grundfarben Rot, Grün und Blau zeigen Winkelabstände von 120° und Intensitätswerte von 0,33.
- Die HSI-Wertetripel sind also nicht beliebig innerhalb ihrer Wertebereiche belegbar. Dies zeigen der Wertebereich der graphischen Farbraumdarstellung sowie die Herleitungen aus den RGB-Kodierungen.

Glättung von Farbbildern (1)

Die Glättung von Einkanalbildern wurde umgesetzt,

- indem ein Glättungsfilter (Mittelwertfilter, Gauß-Filter, Binomialfilter oder Medianfilter) auf jedes Pixel angewendet wird:

Dieser Ansatz ist auf RGB-Farbbilder übertragbar:

- indem das Filter jetzt nur nicht mehr auf einen Intensitätswert $I(x,y)$, sondern auf ein Intensitätstripel angewendet wird:

$$I_{RGB}(x,y) = [I_R(x,y), I_G(x,y), I_B(x,y)].$$

Glättung von Farbbildern (2)

Der Einfachheit halber betrachten wir das Mittelwertfilter:

Sei $N(x,y)$ die durch die Filtermaske definierte Nachbarschaftsumgebung mit K Pixeln und zentriert im Pixel $I_{RGB}(x,y)$ eines RGB-Farbbildes.

Der Mittelwertvektor der RGB-Intensitätsvektoren ist dann:

$$I_{RGB_{av}}(x,y) = 1/K \cdot \sum_{(s,t) \in N(x,y)} I(s,t).$$

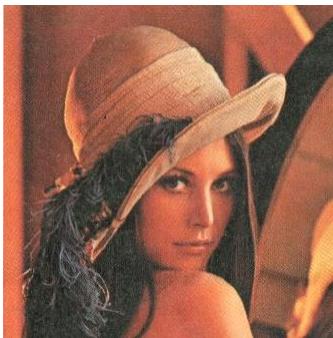
Nach den Gesetzen der Vektoraddition:

$$I_{RGB_{av}}(x,y) = [1/K \cdot \sum_{(s,t) \in N(x,y)} I_R(s,t), 1/K \cdot \sum_{(s,t) \in N(x,y)} I_G(s,t), 1/K \cdot \sum_{(s,t) \in N(x,y)} I_B(s,t)]$$

Glättung von Farbbildern (3)

- Die folgende Folie zeigt
 - das RGB-Farbbild *Lena*
 - seine RGB-Komponenten
 - seine HSI-Komponenten
- Jeder RGB-Kanal wurde separat mit einem 5×5 -Mittelwertfilter geglättet
 - ~ Die so geglätteten Farbkanäle werden zu einem resultierenden geglätteten RGB-Farbbild kombiniert
- Alternativ die HSI-Repräsentation ausschließlich im Intensitätskanal geglättet (die Tönungs- und Sättigungskanäle bleiben unverändert)
 - ~ Das Glättungsergebnis auf dem HSI-Intensitätskanal wird wieder in die RGB-Repräsentation konvertiert

Glättung von Farbbildern (4)



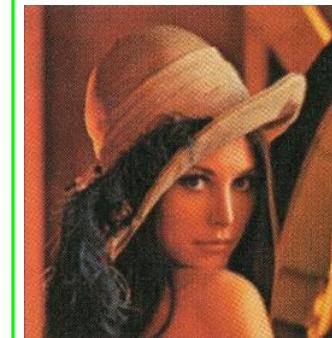
RGB Lena



red component



hue component



RGB smoothing



green component



saturation



blue component



intensity



HSI smoothing

Bildquelle: R. C. Gonzalez,
R. E. Woods: Digital Image
Processing (3rd Ed.). Pearson
Education Intern., pp.
441ff., 2008.

Glättung von Farbbildern (5)

Beide Ergebnisse der Mittelwertglättung sehen zunächst ähnlich aus.

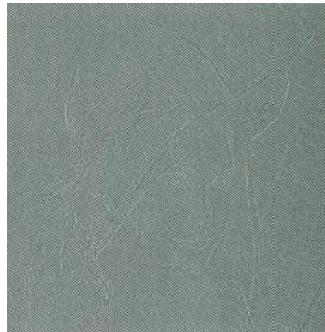
Das Differenzbild zeigt, dass beide Ergebnisse nicht gleich sind:

- Die RGB-Glättung konstituiert sich aus den *Mittelungen über den Farbwerten* der Pixel. Also ergibt sich i.A. eine veränderte Farbe für jeden Pixel.
- Bei der HSI-Glättung wird lediglich die Intensität gemittelt, nicht aber die Tönung und die Saturation. Die Farbe wird also nicht geändert!

Diese Unterschiede nehmen mit wachsender Filtergröße zu.



RGB smoothing



difference



HSI smoothing

Kantenhervorhebung in Farbbildern (1)

Analog zur Betrachtung der Glättung von Farbbildern wird die Erweiterung der Kantenhervorhebung auf Farbbildern betrachtet.

Der Einfachheit halber wird hier der L_4 -Laplace-Filter betrachtet:

$$L_4 = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}.$$

Aus den Gesetzen der Vektoraddition folgt:

$$\nabla^2 I_{RGB}(x,y) = [\nabla^2 I_R(x,y) , \nabla^2 I_G(x,y) , \nabla^2 I_B(x,y)]$$

Kantenhervorhebung in Farbbildern (2)

Wieder wird eine Laplace-Kantenhervorhebung auf den RGB-Kanälen mit einer Laplace-Kantenhervorhebung auf dem Intensitätskanal verglichen.

Wieder zeigt das Differenzbild Unterschiede.

Wieder erklärt das Mischen der Farbinformation bei der RGB-Kantenhervorhebung die Unterschiede zur HSI-Kantenhervorhebung, die ausschließlich auf dem Intensitätskanal ohne Manipulation der Farben durchgeführt wird.



RGB sharpening



difference



HSI sharpening

Fazit zur Filterung auf Farbbildern

- Sowohl für Glättung als auch für Kantenhervorhebung auf Farbbildern **ist das HSI-Farbmodell geeigneter.**
 - Beide Arten der Filterung können dort separat auf dem Intensitätskanal, aus dem sich die Gradienten für Störungen und Kanten ergeben, angewendet werden.
 - Dadurch wird die Farbinformation nicht manipuliert.
- In der Praxis wird auch oft auf Grauwertvarianten von Farbbildern gearbeitet. Diese entsprechen dem Intensitätskanal. Natürlich muss dann diese Vernachlässigung der Farbinformation mit der Aufgabenstellung vereinbar sein.

Zusammenfassung (1)

- Nichtlineare Filter können für schwierige Anwendungsfälle eine im Vergleich zu linearen Filtern effizientere Trennung zwischen Rauschunterdrückung und Kantenhervorhebung umsetzen.
- Zwei nichtlineare Filterklassen sind **Rangordnungsfilter** und **Diffusionsfilter**.
- Rangordnungsfilter sortieren die Intensitäten eines Pixels und der Pixel seiner lokalen Nachbarschaft und liefern den Intensitätswert, der an einem bestimmten Rang der sortierten Reihenfolge steht – z.B., erster, d. h. minimaler Wert, letzter, d.h. maximaler Wert, Median, ...
- Das **Medianfilter** liefert den Medianwert, unterdrückt effektiv das Impulsrauschen und ist bei lokal geradlinigem Kantenmodell kantenerhaltend.

Zusammenfassung (2)

- Diffusionsfilter basieren auf dem Modell der Diffusion. Dabei wird die Intensitätsfunktion eines Bildes als räumliche Konzentrationsfunktion interpretiert.
- Nach der isotropen homogenen Diffusion sind nach großer Iterationszahl leicht alle Kanten vollständig verschwunden. Sie entspricht einer Gauß-Filterung.
- Bei der isotropen inhomogenen Diffusion bleibt der Unterschied zwischen Vordergrund und Hintergrund durch die Kantenbegrenzungen erhalten, allerdings bleiben die Kanten verrauscht.
- Die anisotrope Diffusion erfolgt nur parallel zur Kantenrichtung und nicht über die Kante hinaus. Daher bleibt der Unterschied zwischen Vordergrund und Hintergrund erhalten und das Rauschen an den Kanten wird unterdrückt.
- Die Kantenerhaltung der anisotropen Diffusion ist durch Entkopplung der Modelle von Rauschen und Kanten verbesserbar, indem zwischen störungsbedingtem Konzentrationsgradient ∇u und Kantengradient ∇k unterschieden wird.

Zusammenfassung (3)

- Die Anwendung von Filtern zur Glättung und Kantenhervorhebung kann auf RGB-Farbbilder erweitert werden, indem dieselbe Filterung auf jedem Kanal separat angewendet wird.
 - Hierbei kommt es aber auch zur ungewollten Farbmanipulation.
- Die Anwendung von Filtern zur Glättung und Kantenhervorhebung auf HSI-kodierte Farbbilder zeigt diese Manipulation nicht, da die Intensitätsinformation, aus der sich Gradienten für Störungen und Kanten ergeben, von der Farb-information getrennt kodiert ist.

Anhang: RGB zu HSI

Converting colors from RGB to HSI

Given an image in RGB color format, the H component of each RGB pixel is obtained using the equation

$$H = \begin{cases} \theta & \text{if } B \leq G \\ 360 - \theta & \text{if } B > G \end{cases} \quad (6.2-2)$$

with

$$\theta = \cos^{-1} \left\{ \frac{\frac{1}{2}[(R - G) + (R - B)]}{[(R - G)^2 + (R - B)(G - B)]^{1/2}} \right\}.$$

The saturation component is given by

$$S = 1 - \frac{3}{(R + G + B)} [\min(R, G, B)]. \quad (6.2-3)$$

Finally, the intensity component is given by

$$I = \frac{1}{3} (R + G + B). \quad (6.2-4)$$

It is assumed that the RGB values have been normalized to the range $[0, 1]$ and that angle θ is measured with respect to the red axis of the HSI space, as indicated in Fig. 6.13. Hue can be normalized to the range $[0, 1]$ by dividing by 360° all values resulting from Eq. (6.2-2). The other two HSI components already are in this range if the given RGB values are in the interval $[0, 1]$.

The results in Eqs. (6.2-2) through (6.2-4) can be derived from the geometry shown in Figs. 6.12 and 6.13. The derivation is tedious and would not add significantly to the present discussion. The interested reader can consult the book's references or web site for a proof of these equations, as well as for the following HSI to RGB conversion results.

Anhang: HSI zu RGB (1)

Converting colors from HSI to RGB

Given values of HSI in the interval $[0, 1]$, we now want to find the corresponding RGB values in the same range. The applicable equations depend on the values of H . There are three sectors of interest, corresponding to the 120° intervals in the separation of primaries (see Fig. 6.13). We begin by multiplying H by 360° , which returns the hue to its original range of $[0^\circ, 360^\circ]$.

RG sector ($0^\circ \leq H < 120^\circ$): When H is in this sector, the RGB components are given by the equations

$$B = I(1 - S) \quad (6.2-5)$$

$$R = I \left[1 + \frac{S \cos H}{\cos(60^\circ - H)} \right] \quad (6.2-6)$$

and

$$G = 3I - (R + B). \quad (6.2-7)$$

Anhang: HSI zu RGB (2)

GB sector ($120^\circ \leq H < 240^\circ$): If the given value of H is in this sector, we first subtract 120° from it:

$$H = H - 120^\circ. \quad (6.2-8)$$

Then the RGB components are

$$R = I(1 - S) \quad (6.2-9)$$

$$G = I \left[1 + \frac{S \cos H}{\cos(60^\circ - H)} \right] \quad (6.2-10)$$

and

$$B = 3I - (R + G). \quad (6.2-11)$$

BR sector ($240^\circ \leq H \leq 360^\circ$): Finally, if H is in this range, we subtract 240° from it:

$$H = H - 240^\circ. \quad (6.2-12)$$

Then the RGB components are

$$G = I(1 - S) \quad (6.2-13)$$

$$B = I \left[1 + \frac{S \cos H}{\cos(60^\circ - H)} \right] \quad (6.2-14)$$

and

$$R = 3I - (G + B). \quad (6.2-15)$$

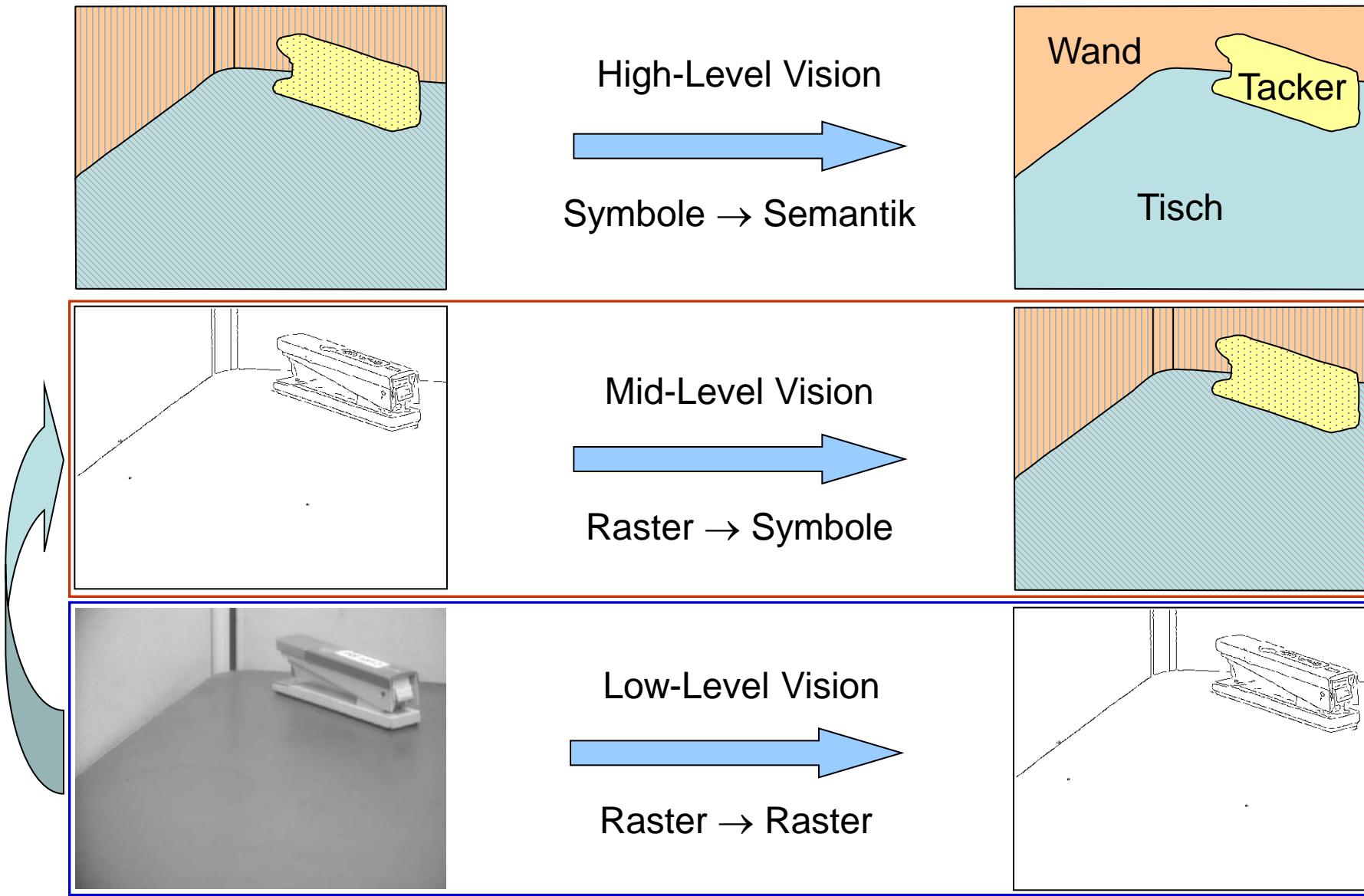
Intelligente Sehsysteme

5 Segmentierung

Histogrammbasierte, regionenbasierte und
texturbasierte Segmentierung

Florian Oßwald

Phasen des Computersehens (Wiederholung)



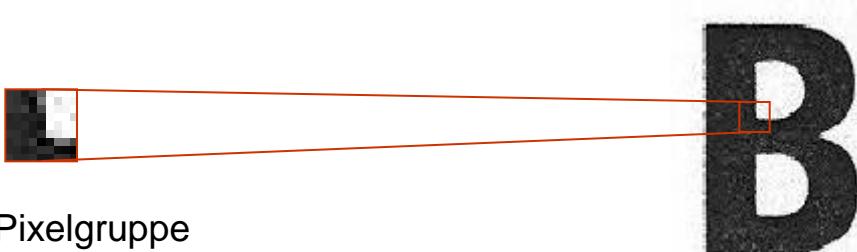
Bildquelle: Stuart Russell, Peter Norvig: "Artificial Intelligence - A Modern Approach", Prentice Hall, 2003.

- Einführung
- Histogrammbasierte „Segmentierung“
 - Methode von Otsu
- Region Labeling
 - Connected-Component Labeling & Flood Fill
- Segmentierung nach Homogenitätskriterien
 - Regionenbasierte Segmentierung
 - Region Merging & Split and Merge
 - Texturbasierte Segmentierung
 - Texturmaße und Texturbilder

Segmentierung (1)

- I. A. tragen einzelne Pixel nicht genügend Information für eine Bildinterpretation
- ↗ Pixel werden in zusammenhängende Segmente gruppiert, um die Bildinterpretation auf Eigenschaften dieser Segmente zu begründen
-

Beispiel: selbst lokale Pixelgruppen stellen i.A. keine hinreichende Grundlage für eine Bildinterpretation dar



Lokale Pixelgruppe
ist unzureichend für
Interpretation

Gesamte zusammenhängende Gruppe
von dunklen Pixeln wäre die optimale
Grundlage zur Zeichenerkennung

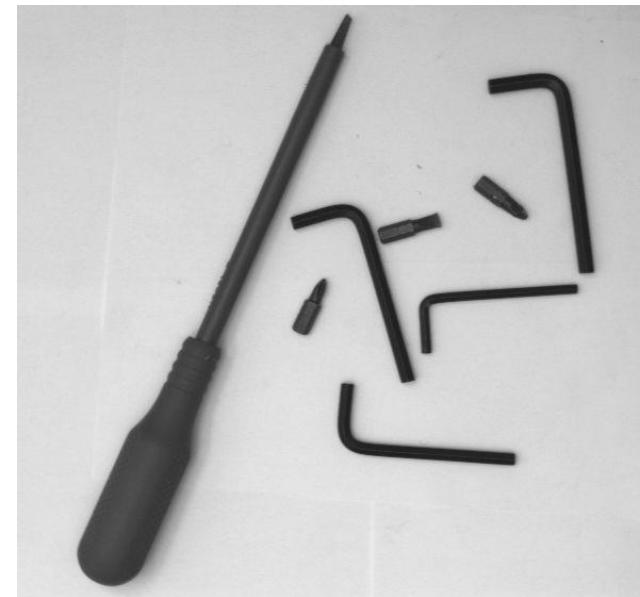
Segmentierung (2)

Typische Eigenschaften von Segmenten sind z.B. topologische Eigenschaften wie die Anzahl von Komponenten oder Löchern (der Großbuchstabe **B** hat z.B. zwei Löcher, der Kleinbuchstabe *i* hat z.B. zwei Komponenten) oder Formeigenschaften wie Kompaktheit oder Exzentrizität

Beispiel: sind die Pixel, die Werkzeuge abbilden, zu Pixelgruppen gruppiert, können die Werkzeuge z.B. über die Form der Pixelgruppen klassifiziert werden.

Informell (später genauer):

- Bit: kompakt und nicht gekrümmkt
- Inbusschlüssel: schmal, länglich und gekrümmkt
- Schraubendreher: schmal, länglich und nicht gekrümmkt



Segmentierung (3)

Der Gruppierungsprozess von Pixeln zu zusammenhängenden Segmenten heißt **Segmentierung**.

Def. [Segmentierung, Segmente]

Als Segmentierung bezeichnet man die Erzeugung von *zusammenhängenden Regionen* benachbarter Pixel entsprechend eines bestimmten **Homogenitätskriteriums**, so dass die daraus folgende Zerlegung des Bildes vollständig und überdeckungsfrei ist. Die so erzeugten zusammenhängenden Regionen werden als (Bild-) Segmente bezeichnet.

Die Segmentierung verbindet die subsymbolische Ebene der Low-Level Vision mit der semantischen Ebene der High-Level Vision.

Segmentierung (4)

In dieser Vorlesung werden drei Ansätze der Segmentierung vorgestellt:

- histogrammbasierte Segmentierung,
 - Segmentierung nach Homogenitätskriterien:
 - regionenbasierte Segmentierung
 - texturbasierte Segmentierung
-

In den Folgevorlesungen werden behandelt:

- Segmentierung nach Diskontinuitäten
- Multi-Skalen-Segmentierung
- modellbasierte Segmentierung

Histogrammbasierte Segmentierung (1)

Einfachster Fall: „Segmentierung“ durch Histogrammanalyse

- Z. B. gescannter Text: dunkle Zeichen vor hellem Hintergrund
- Allg.: die Vordergrundobjekte unterscheiden sich vom Hintergrund durch ihre Intensitätswerte



↗ das Histogramm ist ein **bimodales Histogramm** mit zwei ausgeprägten Maxima, dessen beide Modi für Vordergrund und Hintergrund stehen

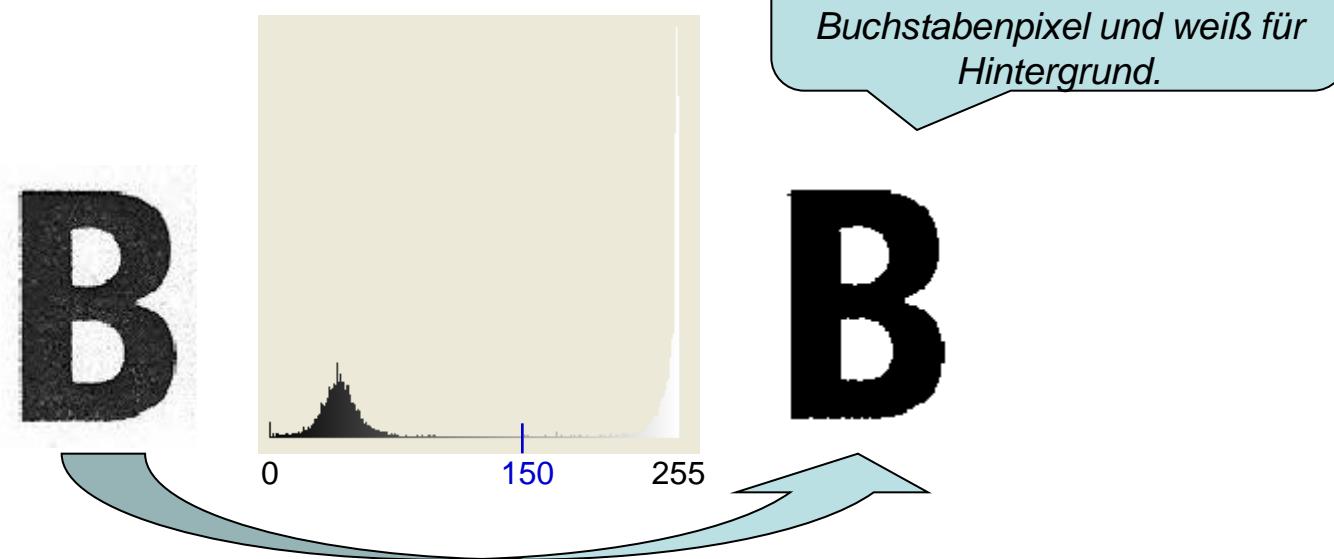
↗ Eine Trennung aller Pixel (x,y) in Vordergrundpixel und Hintergrundpixel entspricht einer **binären Klassifikation** über einen **Schwellwert t_B** (engl. *threshold*). Diese binären Klassifikation wird im Computersehen als **Binarisierung** bezeichnet:

$$I(x, y) \in \begin{cases} \text{Objekt:} & I(x, y) \leq t_B, \\ \text{Hintergrund:} & I(x, y) > t_B. \end{cases}$$

Histogrammbasierte Segmentierung (2)

Beispiel: Grauwertbild eines gescannten Großbuchstabens **B** und dessen bimodales Histogramm

Als Schwellwert t_B wird das lokale Minimum zwischen den beiden Maxima gewählt – hier: $t_B = 150$



Einfache Binarisierung eines Grauwertbildes über Schwellwert $t_B = 150$.

Histogrammbasierte Segmentierung (3)

Histogrammbasierte Segmentierung lässt sich von bimodalen Histogrammen auf **multimodale Histogramme** verallgemeinern:

- Grauwertbilder mit multimodalen Histogrammen zeigen $n \geq 2$ lokale Maxima mit entsprechend $n-1$ lokalen Minima
- Somit sind diese $n-1$ Minima zu suchen und entsprechende $n-1$ Schwellwerte t_i ($1 \leq i \leq n-1$) zu wählen
- Die Pixel, die dem k -ten Maximum zugehören ($1 \leq k \leq n$), werden der Klasse k zugeordnet und entsprechend markiert (engl.: *labeling*)

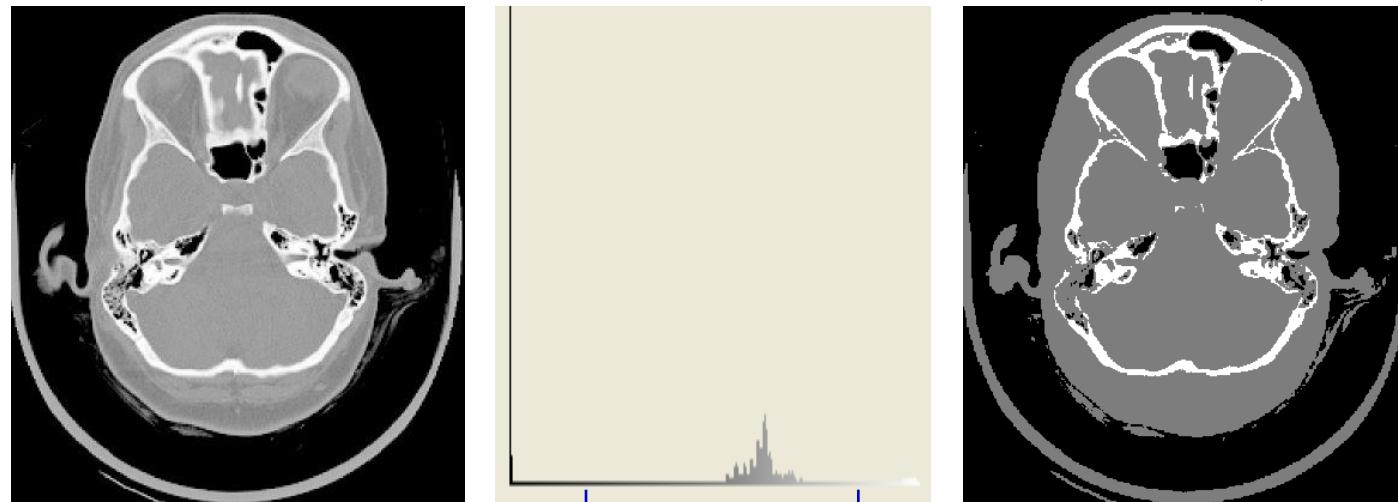
Wegen der schwellwertbasierten Zuordnung wird die histogrammbasierte Segmentierung auch als **Schwellwertsegmentierung** bezeichnet

Histogrammbasierte Segmentierung (4)

Beispiel: Grauwertbild des MRT-Bildes eines Kopfes und dessen multimodales Histogramm mit drei Maxima.

Trennung der Pixel in die Klassen *Hintergrund/Hohlräume* (schwarz), *Gewebe* (grau) und *Knochen* (weiß) durch die beiden Schwellwerte $t_1 = 50$ und $t_2 = 220$.

Visualisiert durch drei gut unterscheidbare Grauwerte



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Klassifikation eines MRT-Grauwertbildes über Schwellwerte $t_1 = 50$ und $t_2 = 220$.

Histogrammbasierte Segmentierung (5)

Bestimmung der Schwellwerte:

- interaktive Bestimmung, die ggf. auch [Anwendererfahrung](#) nutzen kann
- automatisierte [Suche](#) nach lokalen Minima im Histogramm

Hier: die [Clustering-Methode nach Otsu](#)

- Grundidee: Vordergrund- bzw. Objektpixel einerseits sowie Hintergrundpixel andererseits bilden zwei Cluster
- Methode: exhaustive Suche nach dem Schwellwert T_{opt} , der die Intraklassenvarianz (*within-class variance*) minimiert bzw. die Interklassenvarianz (*between-class variance*) maximiert

Histogrammbasierte Segmentierung nach Otsu (1)

Geg.: norm. Histogramm $p_I(I) = \frac{n_I}{S \cdot Z}$ für Bild $I = [I(x,y)]$

Intraklassenvarianz $\sigma_{Within}^2(T)$ für Schwellwert T :

$$\sigma_{Within}^2(T) = n_B(T) \cdot \sigma_B^2(T) + n_O(T) \cdot \sigma_O^2(T)$$



Quelle: https://en.wikipedia.org/wiki/Otsu's_method
(27.11.2017)

mit

$$n_B(T) = \sum_{i=0}^{T-1} p(i)$$

$$n_O(T) = \sum_{i=T}^{I_{\max}} p(i)$$

Hier dunkle Hintergrundpixel und helle Objektpixel

$\sigma_B^2(T)$ = Varianz der Hintergrundpixel $I(x,y)$ mit $I(x,y) < T$

$\sigma_O^2(T)$ = Varianz der Vordergrundpixel $I(x,y)$ mit $I(x,y) \geq T$

Histogrammbasierte Segmentierung nach Otsu (2)

Intraklassenvarianz $\sigma_{Within}^2(T)$ ist aufwändig zu berechnen

Interklassenvarianz $\sigma_{Between}^2(T)$ für Schwellwert T :

$$\begin{aligned}\sigma_{Between}^2(T) &= \sigma^2 - \sigma_{Within}^2(T) \\ &= n_B(T) \cdot |\mu_B(T) - \mu|^2 + n_O(T) \cdot |\mu_O(T) - \mu|^2\end{aligned}$$

mit Gesamtvarianz σ^2 basiert nur auf Mittelwerten

Mit $\mu = n_B(T) \cdot \mu_B(T) + n_O(T) \cdot \mu_O(T)$ folgt nach Vereinfachung:

$$\sigma_{Between}^2(T) = n_B(T) n_O(T) |\mu_B(T) - \mu_O(T)|^2$$

Histogrammbasierte Segmentierung nach Otsu (3)

Die Interklassenvarianz $\sigma_{Between}^2(T) = n_B(T) \cdot n_O(T) \cdot |\mu_B(T) - \mu_O(T)|^2$ ist also für jeden Schwellwert $T \in \{0, \dots, I_{\max}\}$ zu berechnen und T_{opt} mit maximaler Interklassenvarianz zu wählen.

Zur Effizienzsteigerung der iterativen Berechnungen tragen bei:

$$n_B(T+1) = n_B(T) + n_T$$

n_T = Anteil der
Pixel p mit $I(p) = T$

$$n_O(T+1) = n_O(T) - n_T$$

$$\mu_B(T+1) = \frac{\mu_B(T) \cdot n_B(T) + n_T \cdot T}{n_B(T+1)}$$

$$\mu_O(T+1) = \frac{\mu_O(T) \cdot n_O(T) - n_T \cdot T}{n_O(T+1)}$$

Histogrammbasierte Segmentierung nach Otsu (1)

Beispiel:



Bildquelle: https://en.wikipedia.org/wiki/Otsu's_method (27.11.2017)

Region Labeling (1)

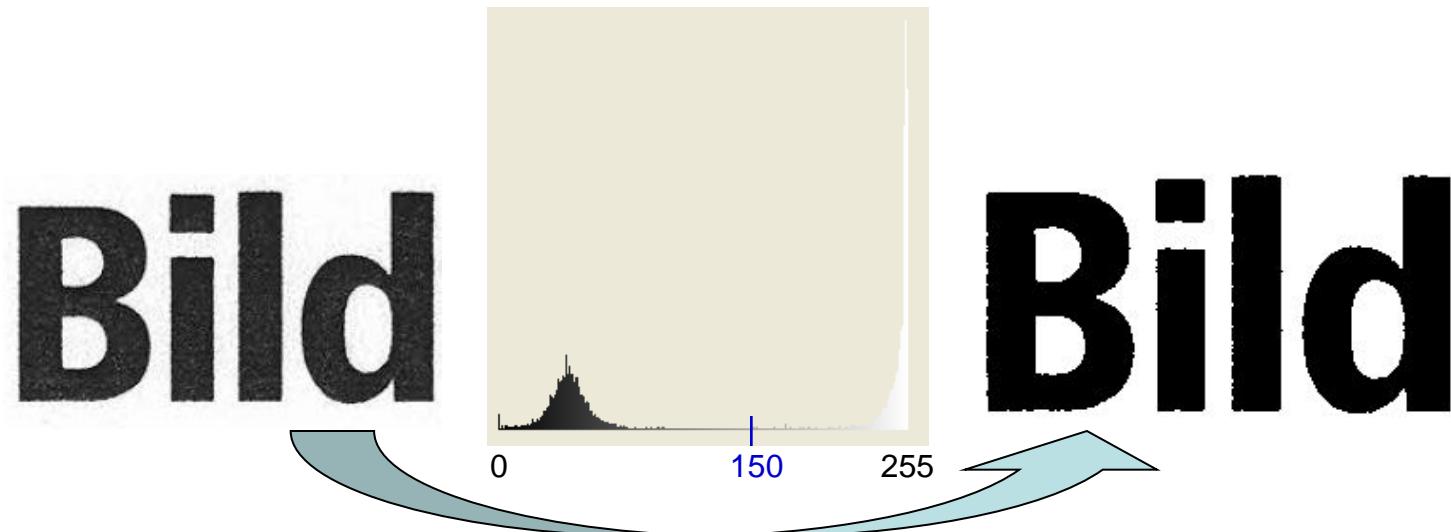
- Über die Schwellwertbildung erhalten alle Pixel in Abhängigkeit von ihrem Grauwert eine Klassenzuordnung (engl. *Label* \rightsquigarrow Klassen-Label)
- Jetzt sind zusammenhängende Regionen zu suchen, deren Pixel dasselbe Klassen-Label zeigen
- Jede Region erhält ein neues, eigenes Regionen-Label, das die Region als separates Segment identifiziert
 - Dieser Schritt wird als *Region Labeling* bezeichnet

Region Labeling (2)

Bspl. 1: nach Binarisierung das eingescannten Wortes ***Bild*** zeigen alle Pixel die Klassen-Label „Vordergrund“ (schwarz) bzw. „Hintergrund“ (weiß).

Ziel des *Region Labelings*: fünf Vordergrundsegmente (für $B, \iota, ;, l, d$) zu durch Zuordnung der Vordergrundpixel zu fünf **Regionen-Labels** bestimmen.

Diese so identifizierten Segmente sind dann Grundlage für Verfahren der High-Level-Vision. Hier z.B. einer Zeichen- und anschließenden Worterkennung.

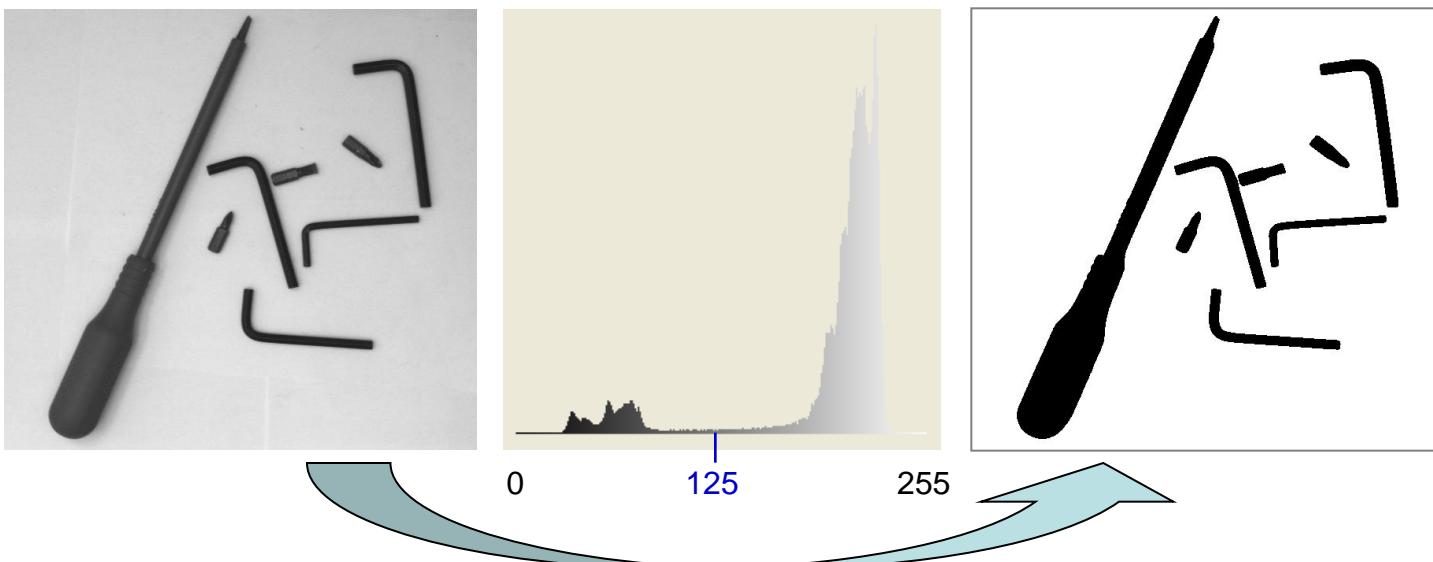


Einfache Binarisierung eines Grauwertbildes über Schwellwert $t_B = 150$.

Region Labeling (3)

Bspl. 2: für das Grauwertbild sind nach Binarisierung acht Vordergrundsegmente zu bestimmen.

Die identifizierten Segmente sind dann Grundlage für Verfahren der High-Level-Vision. Hier etwa einer Objektklassifikation in die Klassen *Schraubendreher*, *Inbusschlüssel* und *Bit*.



Einfache Binarisierung eines Grauwertbildes über Schwellwert $t_B = 125$.

Region Labeling (4)

Eine Umsetzung von Region Labeling ist [Connected-Component Labeling](#):

- Eingabe: binarisierter [Eingabebild Source](#) (schwarze Objektpixel, weiße Hintergrundpixel)
- Ausgabe: [Bild Destination](#), in dem die Labels (Regionennummern) durch gut unterscheidbare RGB-Codes visualisiert sind
- Verarbeitung: Zeilenweise Suche nach Objektpixeln in *Source*. Wird ein Objektpixel gefunden, wird überprüft, ob in seiner 8-Nachbarschaft bereits gelabelte Objektpixel sind. Ist dies der Fall, wird dem aktuellen Objektpixel das minimale Objektlabel eines Nachbarpixels zugeordnet. Sonst wird ihm ein neues Objektlabel im Ausgabebild *Destination* zugeordnet.
- Nachbearbeitung: Bei konkaven Objekten können deren Objektpixel noch unterschiedliche Labels zeigen. Daher werden in einer zweiten Schleife Äquivalenzen von Labels detektiert und notiert.

Region Labeling (5)

Connected-component labeling (first part: labeling loop)

begin

label $\leftarrow 0$;

all destination pixels $d(x,y)$ are set to zero in channel 1; // channel 1 is the label channel

for all source pixels $s(x,y)$ **do** row by row

if source pixel $s(x,y)$ is black **then begin** // black = object

if 8-neighborhood of corresponding destination pixel $d(x,y)$

 shows at least one pixel with label value $l \neq 0$ in channel 1

then begin

 find neighbor pixel with smallest label value l_{min} ;

 destination pixel $d(x,y)$ is set to l_{min} in channel 1;

 store the equivalence between neighboring labels in a list or an array of label equivalences;

end if-then;

else begin

label $\leftarrow i$ abel+1;

 destination pixel $d(x,y)$ is set to *label* in channel 1;

end else;

end if-then;

end for;

Region Labeling (6)

Connected-component labeling (second part: loop for checking for label equivalences)

...

select a sufficient number n of good distinguishable RGB colors $RGB[1], \dots, RGB[n]$;

for all destination pixels $d(x,y)$ **do** row by row

if destination pixel $d(x,y) = 0$ in channel 1 **then**

 destination pixel $d(x,y)$ is set to white // $(255,255,255)$ = background

else begin // $d(x,y) = k, k \neq 0$

 relabel $d(x,y)$ with smallest equivalent label k' ;

 destination pixel $d(x,y)$ is set to $RGB[k']$;

end for;

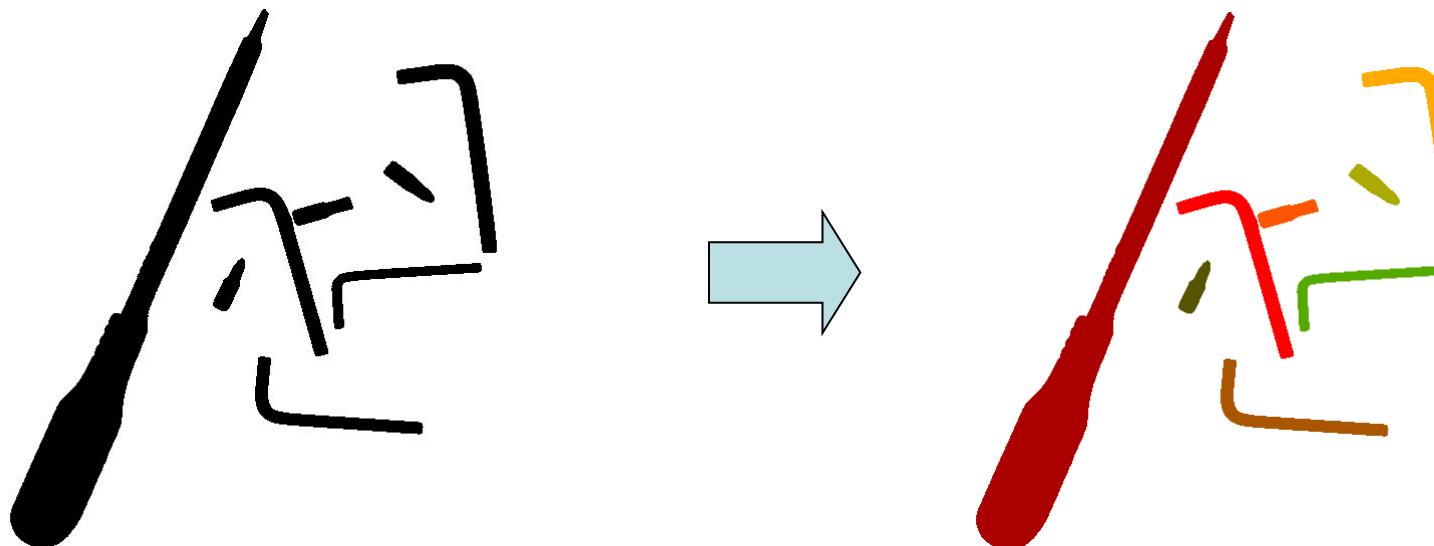
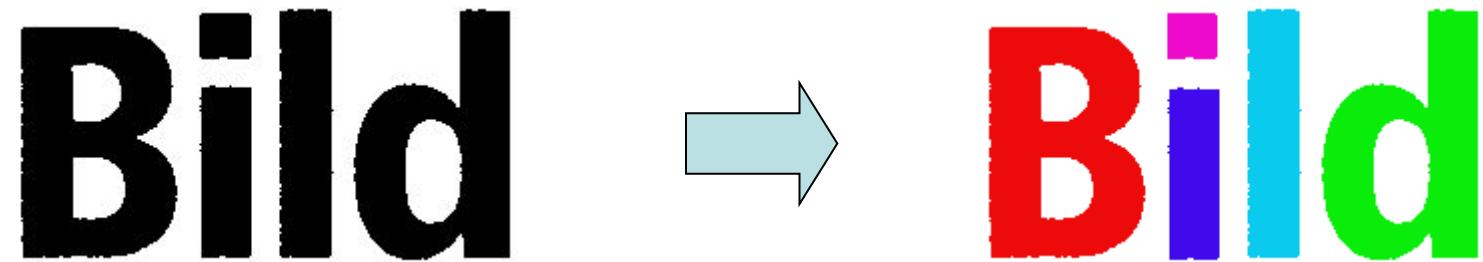
end; // Connected-component labeling

Diese leicht verständliche Grundvariante des Algorithmus kann in vielfältiger optimiert werden. So muss z.B. nicht die vollständige 8-Nachbarschaft überprüft werden, da die Pixel der nächsten Pixelreihe noch nicht gelabelt sind. Es ist auch möglich, alles in einem Durchlauf zu bearbeiten.

Der Algorithmus ist auch auf Bilder mit mehr als zwei Grauwerten erweiterbar.

Region Labeling (7)

Die Anwendung von [Connected-Component Labeling](#) auf die beiden Beispiele:



Region Labeling (8)

- Die beschriebene Version nennt man auch: *Connected component labeling based on label-equivalent-resolving*
- Es gibt eine verführerisch elegante rekursive Formulierung für das *Region Labeling* nach dem Prinzip *Connected component labeling based on label-propagation*, die anschaulich als *Flood Fill* bezeichnet wird:

- *Flood Fill* durchsucht das Bild einfach Reihe für Reihe nach Objektpixeln
- Sobald ein nicht gelabeltes Objektpixel gefunden wird, wird diesem ein neues Label zugeordnet
- Diese Labelzuordnung erfolgt nun rekursiv für alle Pixel der 8-Nachbarschaft

Ab einer bestimmten Bildgröße führt der rekursive *Flood Fill* aber leicht zu einem Stack-Overflow. Entsprechend sind iterative Versionen von *Flood Fill* zu bevorzugen.

Region Labeling (9)

Die [iterative Version von Flood Fill](#) geht von einem binären Bild mit schwarzen Objektpixeln und weißen Hintergrundpixeln aus und nutzt für das Labeling einen LIFO-Stack Q:

```
function flood_fill (binary image  $I_{bin}$ )
    l  $\leftarrow$  0
    for all image pixels p do
        if p is a black object pixel and not labeled then
            l  $\leftarrow$  l + 1;
            Q  $\leftarrow$  [p];
        while (Q is not empty) do
             $p_{buffer} \leftarrow pop_{last}(Q)$ 
            if  $p_{buffer}$  is a black object pixel and not labeled then
                label  $p_{buffer}$  with l;
                for all  $p_{neighbor}$  in  $N_8(p_{buffer})$  do pushlast(Q,  $p_{neighbor}$ );
        end;
```

Vorteile der schwwellwertbasierten Segmentierung

- Die schwwellwertbasierte Segmentierung besticht durch einfache und leicht verständliche Vorgehensweise
- Bei interaktiven Lösungen ist sie Benutzern leicht vermittelbar
- Benutzer können bei der Wahl der Schwellwerte auf einfache Weise Expertenwissen einbringen

Grenzen der schwellwertbasierten Segmentierung

- Globale, für das gesamte Bild, geltende Schwellwerte sind problematisch, wenn lokale Störungen im Bild auftreten. Diese können durch aufnahmebedingte Helligkeitsschwankungen oder durch die Objektvorlagen selbst bedingt sein
- Bspl.: Dokumente können lokal unterschiedlich auftretend ausbleichende Schrift und vergilbenden Hintergrund zeigen
 - Die Intensitäten von Schrift und Hintergrund variieren damit lokal. Ein einheitlicher Schwellwert ist dann nicht zu finden

Segmentierung nach Homogenitätskriterien

Im Vergleich zur starren schwellwertbasierten Segmentierung ist es flexibler, wenn ein *allgemeines Homogenitätskriterium* für alle Segmente definierbar ist, das die Zusammengehörigkeit mit Bezug zu den Pixeln der Segmente definiert.

Beispiele: die Intensitätsvarianz einer Region oder der Abstand zwischen maximalem und minimalem Intensitätswert der Region sind mögliche Homogenitätsfunktionen, auf denen ein Schwellwert als Kriterium anwendbar ist.

Mehrkanal-Bilder: Für die Segmentierung eines mehrkanaligen Bildes kann zunächst separat für jeden einzelnen Kanal ein Teilhomogenitätskriterium berechnet werden. Die Gesamthomogenität kann dann durch (gewichtete) Mittelung abgeleitet werden.

Region Merging (1)

Region Merging als einfachster Segmentierungsansatz über Homogenitätskriterien:

- 1) Start: jedes Pixel bildet ein eigenes Segment
- 2) Merging: zwei benachbarte Segmente werden zusammengefasst, wenn sie auch gemeinsam das Homogenitätskriterium erfüllen.
Zur Reihenfolge: i.A. werden immer die benachbarten Segmente zusammengefasst, die das Homogenitätskriterium am stärksten erfüllen.
- 3) Terminierung: die Segmentierung terminiert, wenn keine Segmente mehr zusammengefasst werden können.

Beachte: Da Region Merging mit Pixeln als Segmenten startet, muss das Homogenitätskriterium *für einzelne Pixel und für Regionen* berechenbar sein.

Region Merging (2)

Der Segmentierungsprozess wird durch den schrittweisen Aufbau eines **Regionenadjazenzgraphen** (Region Adjacency Graph – **RAG**) dokumentiert.

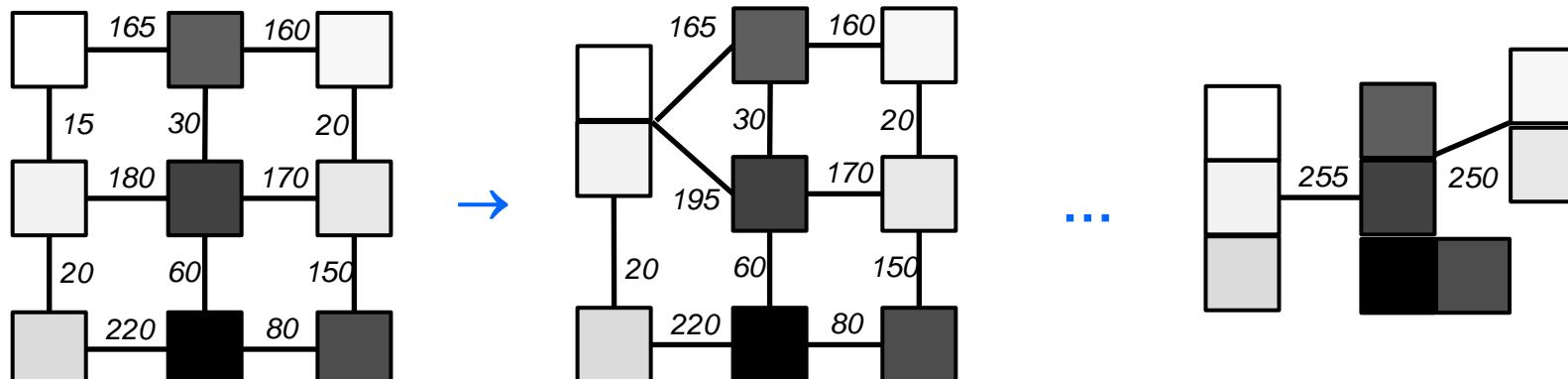
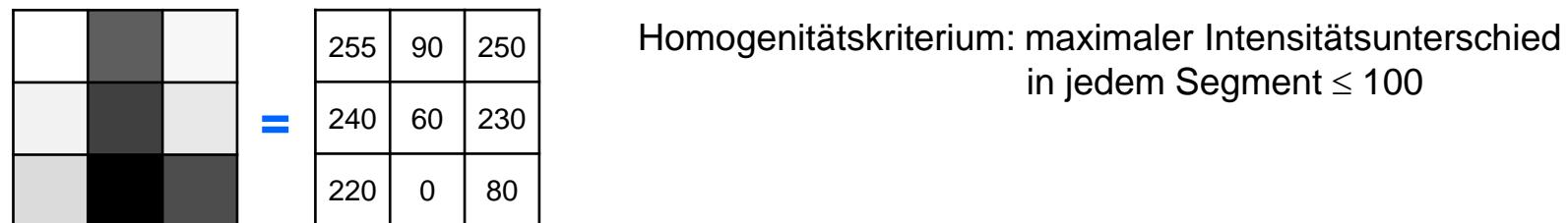
- Die **Knoten des RAG** repräsentieren die Regionen. Knotenwerte sind die für die Berechnung des Homogenitätskriteriums relevante Informationen.
- Eine **Kante des RAG** verbindet zwei Knoten gdw. die entsprechenden Regionen *direkt benachbart* sind. Kantenwerte geben den Homogenitätswert der Region wieder, die sich aus der Fusion der beiden den Knoten entsprechenden benachbarten Regionen ergäbe.

Das *Region Merging* fusioniert solange diejenigen benachbarten Regionen mit bester Kantenbewertung, bis keine Kante mehr existiert, deren Bewertung das Homogenitätskriterium erfüllt.

Region Merging (3)

Der schrittweise Aufbau eines [Regionenadjazenzgraphen](#) beim *Region Merging*:

- zunächst wird das Bild auf einen *Region Adjacency Graph* (RAG) abgebildet
- im RAG werden so lange Regionen fusioniert, bis keine zwei benachbarten Regionen das Homogenitätskriterium erfüllen.



Optimalität von Region Merging (1)

Das Ergebnis von *Region Merging* kann prinzipiell abhängig von der Verarbeitungsreihenfolge sein:

- Die Homogenitätswerte wie Mittelwerte und Varianzen, Abstände zu Maximal- und Minimalwerten etc. beziehen sich ja gerade auf die aktuell im Segment erfassten Pixel.
- Somit gehen Fusionen von Regionen i.A. mit Änderungen der Homogenitätswerte einher.

Optimalität von Region Merging (2)

Beispiel für die Reihenfolgeabhängigkeit von Region Merging:

z. B. kann die durch Fusion von zwei Regionen R_1 und R_2 erzeugte Region R_{12} aufgrund ihrer neuen Homogenitätswerte mit einer dritten Region R_3 fusioniert werden, obwohl R_3 mit keiner der beiden ursprünglichen Regionen R_1 und R_2 fusionierbar war.

Sicher ist aber, dass alle erzeugten Segmente die geforderten Homogenitätskriterien erfüllen.

Split-and-Merge (1)

Region-Merging erzeugt Segmente, indem es mit den kleinsten Segmenten, den Pixeln, beginnt und diese schrittweise zu größeren Segmenten fusioniert.

Split-and-Merge erzeugt Segmente, indem es mit dem größtmöglichen Segment, dem gesamten Bild, beginnt und dies schrittweise in kleinere Segmente unterteilt.

Split-and-Merge (2)

Split-and-Merge zeigt zwei Phasen:

- a) Die Unterteilung des Bildes in homogene Segmente ist der **Splitting-Teil** des Verfahrens.
 - Das Splitting kann prinzipiell zu einer **Übersegmentierung** führen, bei der benachbarte Segmente existieren, die ohne Verletzung des vorgegebenen Homogenitätskriteriums zusammen gefasst werden können.
- b) Diese entsprechende Zusammenfassung erfolgt im **Merging-Teil** des Verfahrens.

Split-and-Merge (3)

Split-and-Merge arbeitet mit folgenden Schritten:

- 1) **Start**: das gesamte Bild wird als Segment betrachtet.
- 2) **Start des Splittings**: ein Segment wird entlang der x- und der y-Achse wiederholt in vier gleich große Segmente zerlegt, wenn es einem vorgegebenen Homogenitätskriterium **nicht** genügt.
- 3) **Terminierung des Splittings**: die Zerlegung terminiert, wenn alle Segmente das Homogenitätskriterium erfüllen.
- 4) **Start des Mergings**: benachbarte Segmente werden fusioniert, wenn sie auch zusammenhängend das Homogenitätskriterium erfüllen.
- 5) **Terminierung des Mergings**: die Fusionierung terminiert, wenn keine benachbarten Segmente existieren, die auch nach Fusionierung das Homogenitätskriterium erfüllen würden.

Eigenschaften von Split-and-Merge (1)

- 1) Von Beginn an bestehen alle Regionen aus mehreren Pixeln.
 - Im Ggs. zum Region Merging, das prinzipiell mit einzelnen Pixeln als Segmenten startet, ist das Homogenitätskriterium gleich auf Pixelmengen definierbar und anwendbar.
 - Ferner sind damit probabilistische oder statistische Homogenitätskriterium unmittelbar einsetzbar (z.B. *statistisches Region Merging*).

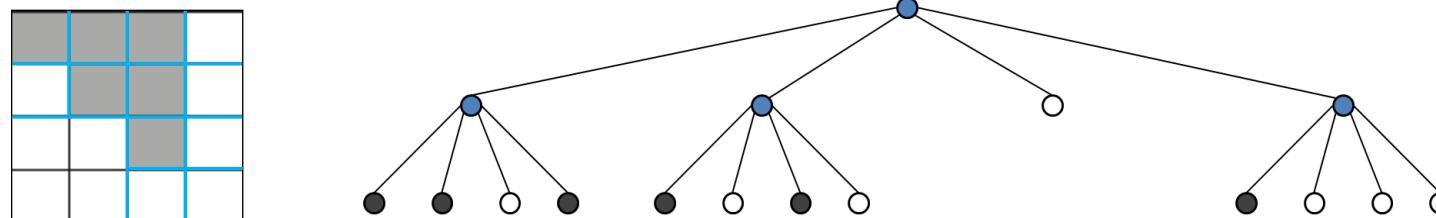
Eigenschaften von Split-and-Merge (2)

- 2) Der Splitting-Teil führt sehr selten zu Segmenten mit nur einem einzigen Pixel.
 - Der Merging-Teil hat also im Vergleich zum Region Merging i.A. weniger Regionen zu fusionieren. Das ist für die Rechenzeit relevant: Für jeden Merge-Schritt von n Segmenten ist die Zeitkomplexität $O(n)$. Im schlechtesten Fall muss von n Segmenten auf das ganze Bild fusioniert werden, also insges. $O(n^2)$.
- 3) Durch den Merging-Teil sind auch suboptimale Segmentierungen ableitbar in dem Sinne, dass nicht die minimale Zahl von Segmenten abgeleitet wird. Sicher ist aber, dass alle erzeugten Segmente die Homogenitätskriterien erfüllen.

Split-and-Merge (4)

Der Splitting-Teil lässt sich durch einen **Quadtree** repräsentieren. Daher wird der Ansatz auch als **Quadtree-Segmentation** bezeichnet:

- Der Wurzelknoten repräsentiert das gesamte Bild als Startsegment.
- Bei Inhomogenität eines Segments wird der entsprechende (innere) Knoten des Quadtrees in vier Kindsegmente unterteilt.
- Die Blattknoten entsprechen homogenen Segmenten.



Quadtree-Segmentierung bei Z-Durchlaufordnung.

Region Merging, Split-and-Merge, Clustering

Bemerkung: Region Merging und Split-and-Merge können auch in den Kontext der Clusteranalyse eingeordnet werden.

Dort wird innerhalb der sog. **hierarchischen Clusterverfahren** zwischen agglomerativen und divisiven Verfahren unterschieden.

- **Agglomeratives Clustering** geht von der feinsten Unterteilung aller Datenobjekte aus und bildet durch sukzessive Agglomeration Cluster mit ähnlichen Datenobjekten.
- **Divisives Clustering** startet mit der Gesamtmenge aller Datenobjekte und unterteilt diese sukzessive in homogene Cluster.

Somit setzt Region Merging ein agglomeratives Clustering von Bildregionen zur Segmentierung um, während Split-and-Merge ein bildbezogenes divisives Clustering von Bildregionen umsetzt.

Texturbasierte Segmentierung (1)

Als **Textur** einer Region versteht man eine gemeinsame Eigenschaft der Intensitätsverteilung dieser Region, die eine bestimmte **Bildungsregel** zugrunde liegt.

Verschiedene Regionen können sich danach auf zwei Arten unterscheiden:

- (1) die Parametrisierung derselben Bildungsregel ist verschieden,
- (2) die Bildungsregeln sind verschieden.

Ein Beispiel für (1): Bilder von verschiedenen Holzmaserungen, die sich im Abstand und der Dicke der Holzfasern unterscheiden.

Ein Beispiel für (2): Ein Bild von einer Holzmaserung und ein Bild von einem Wollstoff. Die Holzmaserung entspringt einer gänzlich anderen Bildungsregel als das Webmuster des Wollstoffs.

Texturbasierte Segmentierung (2)

Es gibt vielfältige Ansätze zur Definition von Textur :

- Eine **strukturelle Texturdefinition** beschreibt Textur als Zusammensetzung aus sog. Texturelementen (engl. *Texture Elements – texels*) wie z.B. linienförmigen Pixelgruppen.
- Eine **statistische Texturdefinition** beschreibt Textur über Charakterisierungen bzw. Maße von Intensitätsmustern.
- Eine **stochastische Texturdefinition** beschreibt Textur als Ergebnis eines parametrisierten, stochastischen Prozesses wie z.B. Gauß-verteilter Varianz.
- Eine **spektrale Texturdefinition** beschreibt Textur durch die Eigenschaften in einer Repräsentation des Frequenzraumes wie z.B. bei Fourier- oder Wavelet-transformationen.

Texturmaße nach Haralick

Die [Texturmaße nach Haralick](#) charakterisieren Intensitätsverteilungen auf Grundlage der sog. *Co-Occurrence-Matrix*.

Eine [Co-Occurrence-Matrix](#) $P_{\alpha,\Delta}[I_1, I_2]$ beschreibt das *gemeinsame Auftreten* (engl. *Co-Occurrence*)

- von Intensitätswerten I_1 und I_2
- mit Abstand Δ und
- mit Winkel α zur horizontalen Achse des Bildkoordinatensystems.

Co-Occurrence-Matrix

Die Einträge der Co-Occurrence-Matrix $\mathbf{P}_{\alpha,\Delta}[I_1, I_2]$ ergeben sich nach:

Für N Pixel $p = (x, y)$ einer Texturregion R ist die Co-Occurrence-Matrix $\mathbf{P}_{\alpha,\Delta}[I_1, I_2]$:

$$\mathbf{P}_{\alpha,\Delta}(I_1, I_2) = 1/N \sum_{p \in R} \delta_D(I(p) - I_1) \cdot \delta_D(I(p+d) - I_2) \text{ mit } d = \Delta \cdot (\cos \alpha, \sin \alpha). *$$

Die diskrete Variante δ_D der Dirac-Funktion liefert 1 für $\delta_D(0)$ und sonst 0.

Da Pixel über größere Positionsdistanzen meist nicht korrelieren, werden zur texturbasierten Klassifikation und Segmentierung i.A. Co-Occurrence-Matrizen für den Abstand von einem Pixel und über die vier Richtungen der 8-Nachbarschaft gerechnet: $\Delta = 1$, $\alpha = 0^\circ, 45^\circ, 90^\circ, 135^\circ$.

Haralicksche Texturmaße (1)

Zunächst sind die Einträge $P_{\alpha,\Delta}(I_1, I_2)$ zu normieren:

$$p_{\Delta,\alpha}(I_1, I_2) \xleftarrow{\text{Normierung}} \frac{1}{S} P_{\Delta,\alpha}(I_1, I_2) \text{ mit } S = \sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} P_{\Delta,\alpha}(I_1, I_2).$$

Die wichtigsten Haralickschen Texturmaße:

- Energie/Uniformität:

$$\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} p_{\Delta,\alpha}^2(I_1, I_2),$$

- Kontrast:

$$\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} (I_1 - I_2)^2 p_{\Delta,\alpha}(I_1, I_2),$$

- Entropie:

$$-\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} p_{\Delta,\alpha}(I_1, I_2) \cdot \log_2(p_{\Delta,\alpha}(I_1, I_2)),$$

mit $0 \cdot \log_2 0 = 0$
(s. auch Vorlesung 1)

- Homogenität/

inverse Differenz:

$$\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} \frac{p_{\Delta,\alpha}(I_1, I_2)}{1 + |I_1 - I_2|},$$

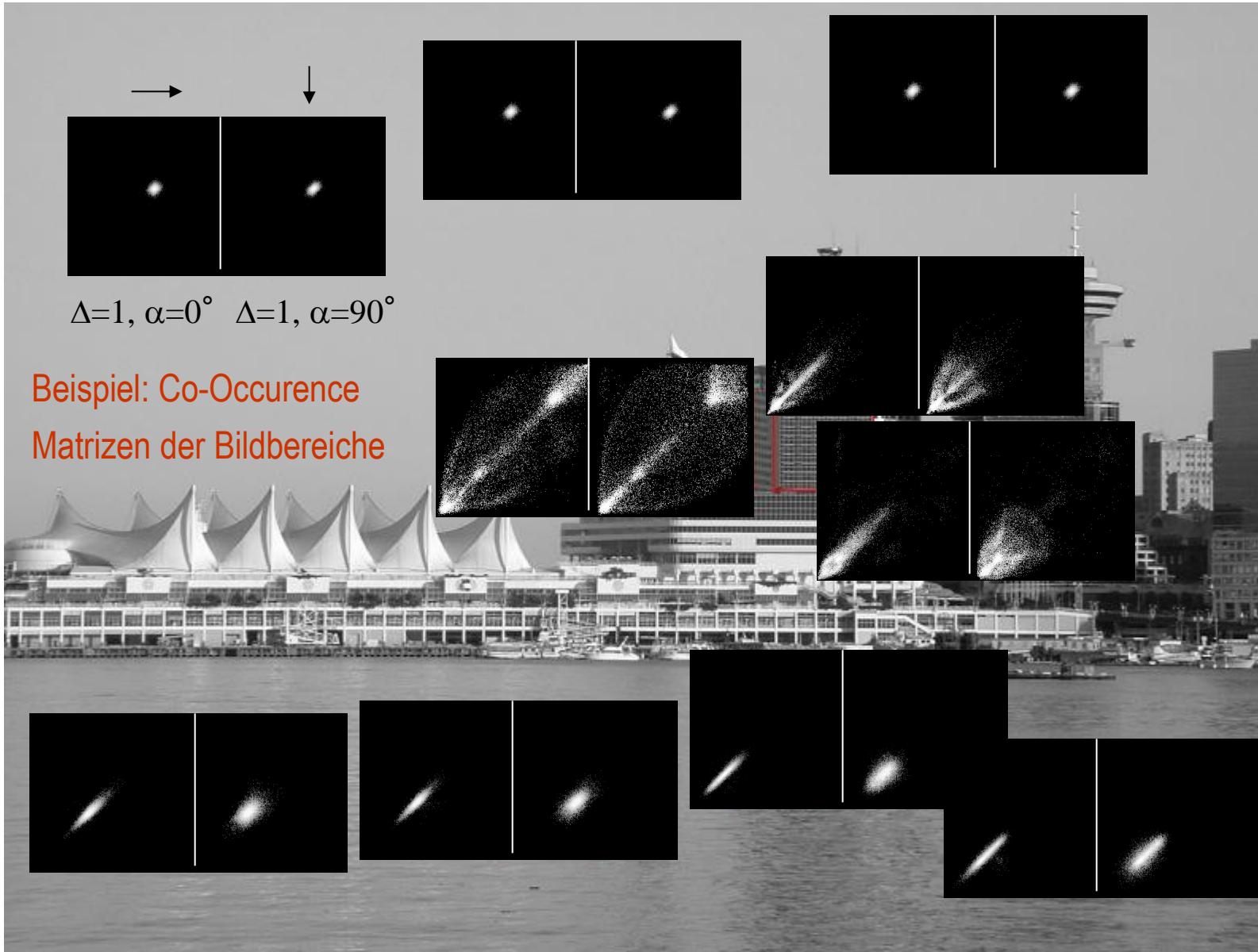
- Inv. Diff.-Moment:

$$\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} \frac{p_{\Delta,\alpha}(I_1, I_2)}{1 + (I_1 - I_2)^2}.$$

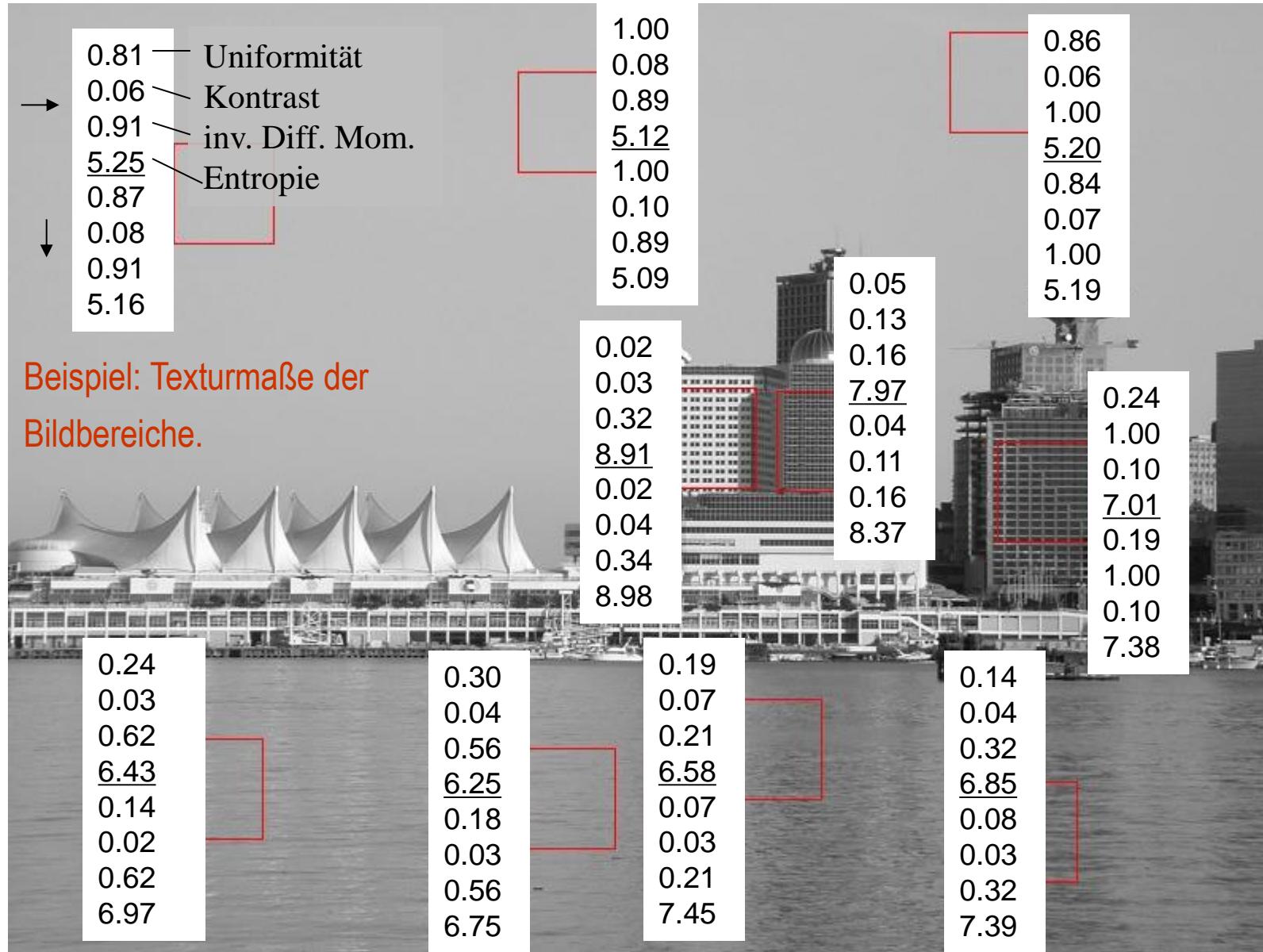
Haralicksche Texturmaße (3)



Haralicksche Texturmaße (4)



Haralicksche Texturmaße (6)



Texturbasierte Segmentierung mit Haralik-Maßen (1)

Anhand der Haralikschen Texturmaße sowie regionenbasierter Verfahren ist nun ein Verfahren zur **texturbasierten Segmentierung** umsetzbar.

- Gegeben sei ein Bild der Größe $Z \times S$ sowie die Annahme einer **Texturfrequenz** von n Pixeln (je horiz. und vertikal).

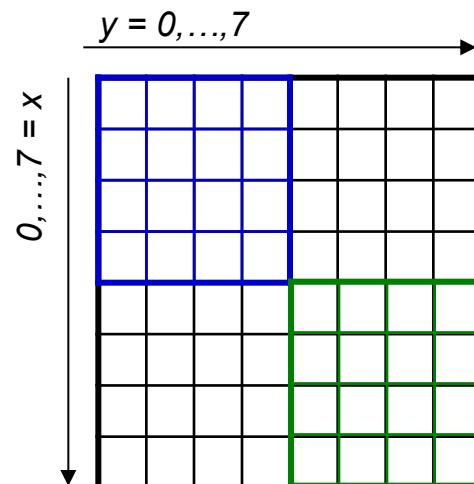
Pixelmengen, die dieselbe Textur zeigen - Schätzungen über Texturgrenzen führen zu Artefakten

 - Somit sind $n \times n$ -Bildblöcke zur Erfassung der Texturmaße nötig.
 - Im Bild der Größe $Z \times S$ lassen sich insgesamt $(Z-n+1)(S-n+1)$ überlappende Blöcke $B_{x,y}$ mit oberem linken Pixel an der Stelle (x,y) für $x = 0, \dots, Z-n+1$ und $y = 0, \dots, S-n+1$ erzeugen.

Texturbasierte Segmentierung mit Haralik-Maßen (2)

Beispiel:

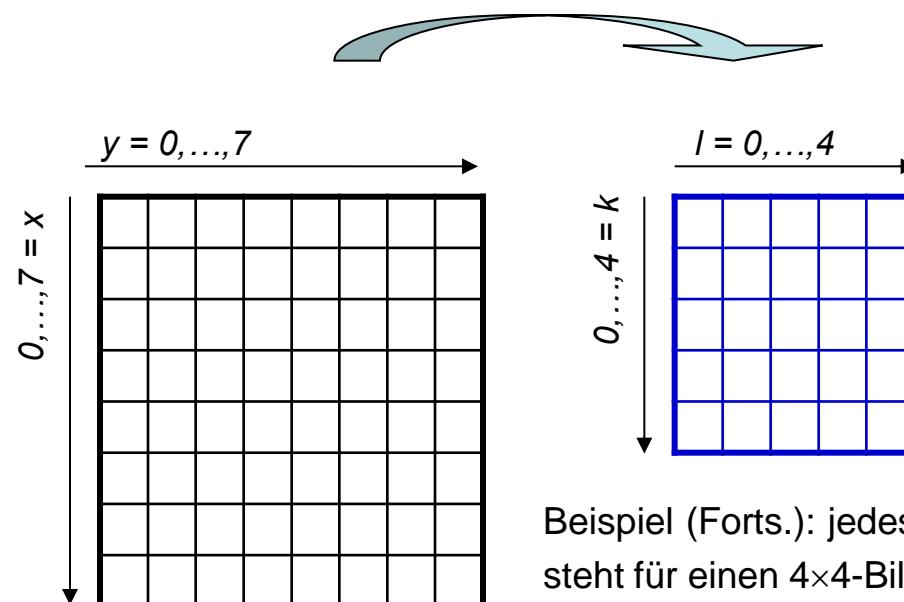
- Bild mit $Z = S = 8$ Bildzeilen und -spalten und 4×4 Bildblöcke für Texturanalyse.
- Dies ergibt 25 vollständig im Bild liegende und z. T. überlappende Bildblöcke von links oben nach rechts unten: erster Bildblock (blau) in (0,0); letzter Bildblock (grün) in (4,4).



Texturbasierte Segmentierung mit Haralik-Maßen (3)

Für jeden Block $B_{x,y}$ wird ein Merkmalsvektor $\mathbf{m}_{x,y} = (m_1, \dots, m_n)$ aus n Texturmaßen erzeugt, der die Textureigenschaft des Blocks $B_{x,y}$ beschreibt.

Die Merkmalsvektoren sind damit Homogenitätsmerkmale in einem neuen **Texturbild** der Größe $(Z-n+1)(S-n+1)$.



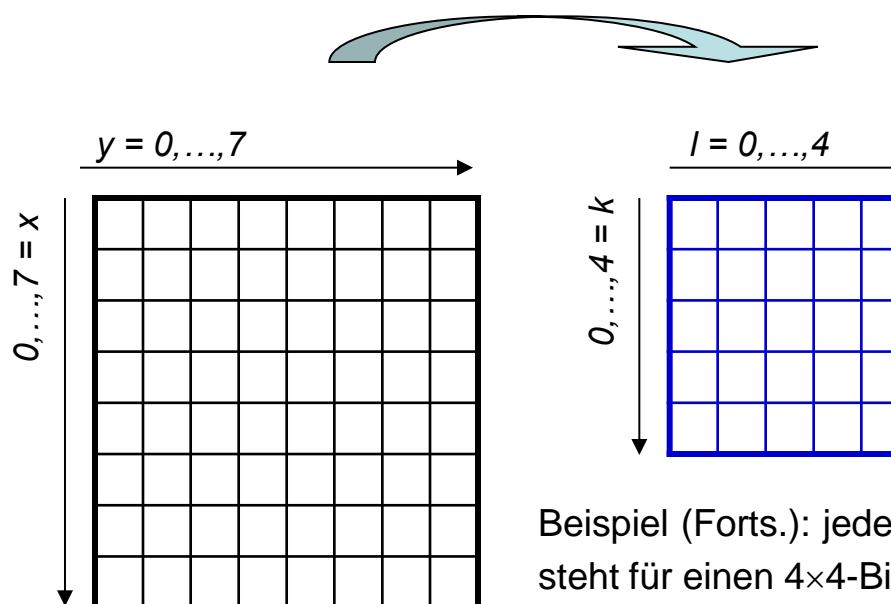
Beispiel (Forts.): jedes Pixel des 5×5 -Texturbildes steht für einen 4×4 -Bildblock im 8×8 -Originalbild.

Texturbasierte Segmentierung mit Haralik-Maßen (4)

Das Texturbild kann nun z.B. durch Split-and-Merge segmentiert werden.

Bspl.: Eine Region ist homogen, wenn die Euklidische Norm der Merkmalsvektoren zwischen allen Pixeln $\mathbf{p}_{x,y}$ der Region einen vorgegebenen Schwellwert d_{\min} unterschreitet:

$$\|\mathbf{m}(\mathbf{p}_i) - \mathbf{m}(\mathbf{p}_j)\| < d_{\min} \text{ für alle Pixel } \mathbf{p}_i \text{ und } \mathbf{p}_j \text{ eines Segments.}$$

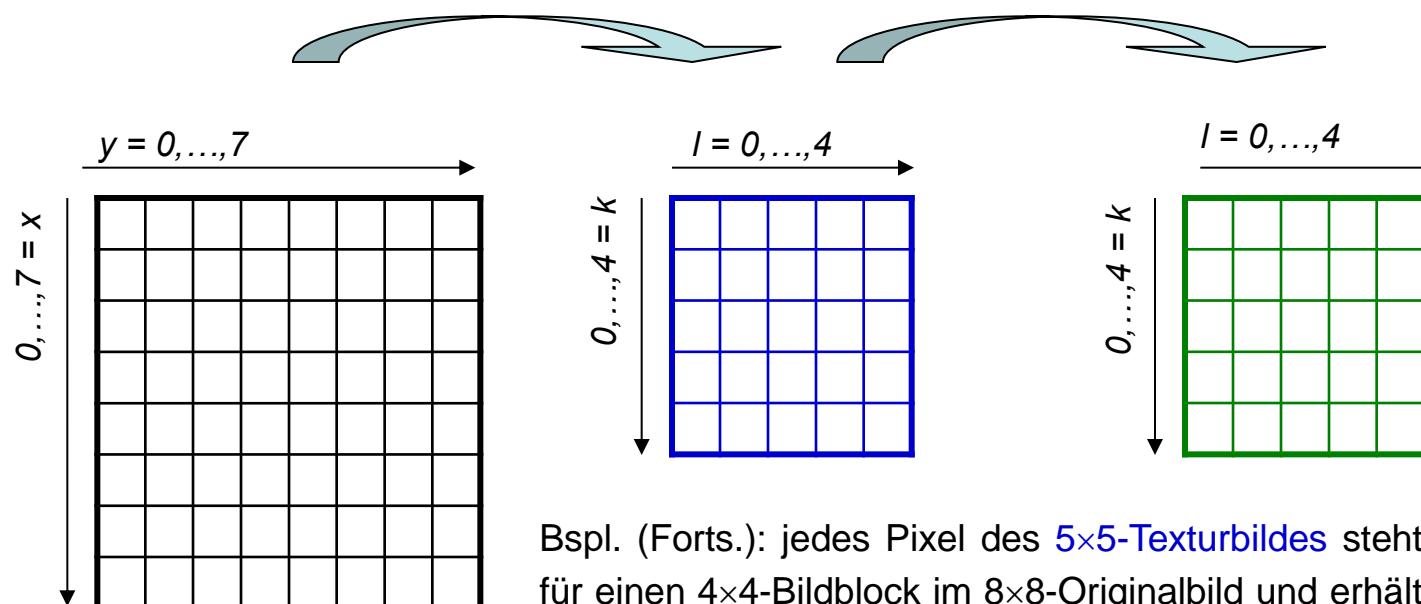


Beispiel (Forts.): jedes Pixel des **5x5-Texturbildes** steht für einen **4x4-Bildblock** im **8x8-Originalbild**.

Texturbasierte Segmentierung mit Haralik-Maßen (5)

Zur Auswertung der texturbasierten Analyse der $(Z-n+1)(S-n+1)$ Blöcke wird ein gleich großes **Label-Feld** L_{Block} der Größe $(Z-n+1)(S-n+1)$ erzeugt.

In L_{Block} werden die Label der erzeugten Segmente eingetragen.

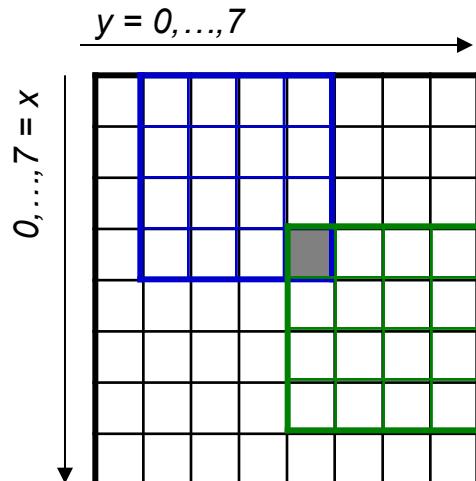


Bspl. (Forts.): jedes Pixel des **5x5-Texturbildes** steht für einen 4×4 -Bildblock im 8×8 -Originalbild und erhält ein Regionen-Label im Label-Feld L_{Block} .

Texturbasierte Segmentierung mit Haralik-Maßen (6)

Also erhält jedes Pixel (x,y) des Originalbildes Zuordnungen von Segmentlabeln aller das Pixel (x,y) überdeckenden Blöcke über das Label-Feld L_{Block} .

Genauer sind dies alle $n \times n$ -Blöcke des Texturbildes mit Koordinaten (k,l) mit $k = \max(0, x-n+1), \dots, \min(x, Z-n)$ und $l = \max(0, y-n+1), \dots, \min(y, S-n)$.



Beispiel (Forts.):

Zwei der insgesamt 16 überdeckenden
4x4 Bildblöcke für das Pixel (4,4) im
Originalbild.

Texturbasierte Segmentierung mit Haralik-Maßen (7)

Im Inneren des Originalbildes wird jedes Pixel maximal durch n^2 Blöcke überdeckt. Zu den Randpixeln hin werden es weniger überdeckende Blöcke bis hin zu den Eckpixeln, die nur einen überdeckenden Block aufweisen.

Jeder überdeckende Block stimmt für ein Segmentlabel aus dem segmentierten Regionenbild. Nach Normierung der Stimmenzahlen (durch die Gesamtzahl aller ein Pixel überdeckender Blöcke) ist das Stimmenergebnis als Label-Wahrscheinlichkeit interpretierbar.

Texturgrenzen (1)

Für Pixel, die nahe an einer Texturgrenze liegen, ergeben sich i.A. mehrere von Null verschiedene Label-Wahrscheinlichkeiten. Einige der Labels stehen für **wahre Texturen**, andere für **scheinbare Texturen**.

Scheinbare Texturen sind durch Blöcke verursacht, die verschiedene Texturen überlagern und dadurch eine „artifizielle Mischtextur“ ableiten. Diese Mischtexturen sind jedoch nicht lagestabil, d.h. sie ändern sich relativ stark in den verschobenen Nachbarblöcken.

Daher ist zu erwarten, dass alle Pixel, die hinreichend weit von einem Texturrand liegen, mehrheitlich Stimmen für das Label einer wahren Textur erhalten werden.

Texturgrenzen (2)

Das „Labeln“ der Originalpixel erfolgt also durch „Abstimmung“ der Bildblöcke für die Pixel, die sie umfassen.

Das Label, für das die meisten Stimmen ausfallen, wird als das Label mit höchster Wahrscheinlichkeit interpretiert.

Die Fehlzuordnungen an den Texturrändern können durch Nachbearbeitung korrigiert werden. Im einfachsten Fall etwa durch ein Glättungsfilter, das die i.A. schmalen fehlerhaften Bereiche „wegglättet“, oder durch Nachbarschaftsanalyse (z.B. Region Labeling oder Relaxation Labeling).

Texturfrequenz

Ein grundsätzliches Problem der texturbasierten Analyse ist natürlich die Festlegung der Texturfrequenz, aus der sich die Größe der Bildblöcke im Texturbild ableitet. Dies wird hier faktisch heuristisch entschieden.

Durch komplexere Multiskalenansätze kann hierfür aber auch systematisch nach der richtigen Frequenz gesucht werden. Dabei werden Segmentierungsergebnisse für verschiedene Frequenzen erzeugt und im Vergleich müsste die wahrscheinlichste ermittelt werden.

Zusammenfassung (1)

- Die Segmentierung verbindet die subsymbolische Ebene der Low-Level Vision mit der semantischen Ebene der High-Level Vision. Dazu werden die Pixel eines digitalen Bildes zu inhaltlich zusammenhängende Regionen gruppiert.
- Der einfachste Ansatz ist der der histogrammbasierten Segmentierung. Diese eignet sich für Bilder mit multimodalem Intensitätshistogramm, dessen Moden durch globale Schwellwerte separierbar sind und so eine intensitätsbasierte Einteilung aller entsprechenden Pixel erzielen. Aus den so klassifizierten Pixeln sind zusammenhängende Pixelregionen durch Region Labeling abzuleiten.

Zusammenfassung (2)

- Regionenbasierte Segmentierung nutzt **Homogenitätskriterien**, die Bezug auf die einzelnen zu bildenden Regionen nehmen und damit flexibler sind als die starren globalen Schwellwerte der histogrammbasierten Segmentierung.
- **Region Mergin** setzt diesen Ansatz bottom-up um, indem von den Pixeln als kleinstmöglichen Segmenten ausgehend Segmente schrittweise zu größeren Segmente fusioniert werden, solange die Fusion das jeweilige Homogenitätskriterium nicht verletzt.
- **Split-and-Merge** geht top-down vor und zerlegt vom gesamten Bild als größtmöglichem Segment ausgehend Segmente schrittweise in immer kleinere Segmente solange, bis alle so erzeugten Segmente das jeweilige Homogenitätskriterium erfüllen.

Zusammenfassung (3)

- Texturbasierte Segmentierung kann auf verschiedenen Texturdefinitionen basieren; z.B. auf strukturellen Texturdefinitionen, statistischen Texturdefinitionen, stochastischen Texturdefinitionen oder spektralen Texturdefinitionen.
- Gemeinsam ist die Aufgabe der Ableitung von Texturmaßen für die texturbasierte Segmentierung.
- Ferner ist mit fehlerhaften Segmentierungsergebnissen an Texturrändern zu rechnen. Diese sind i.A. einer Nachbearbeitung zu unterziehen.

Zusammenfassung (4)

- Die Haralickschen Texturmaße sind sehr bekannt und aus den Co-Occurrence-Matrizen von Bildregionen ableitbar.
- Die verwendeten Texturmaße definieren für Bildblöcke einer ausgewählten Größe einen Merkmalsvektor, der als Homogenitätskriterium in einer regionenbasierten Segmentierung einsetzbar ist.
- Problematisch ist die Bestimmung der Texturfrequenz und damit die Festlegung der Größe der Bildblöcke.
- Multiskalenverfahren bieten dazu eine Lösung.

Intelligente Sehsysteme

6 Segmentierung (II)

Edge Linking, Skelettierung, Canny Edge Operator,
optimale Kantenzüge, Wasserscheidentransformation

Florian Oßwald

Inhalt

- Segmentierung nach Diskontinuitätskriterien
vs. Segmentierung nach Homogenitätskriterien
- Edge Linking
- Linienverdünnung durch Skelettierung
- Canny Edge Operator
 - Non-Maxima-Unterdrückung und Hysterese-Schwellwertbildung
- Interaktive Suche nach optimalen Kantenzügen
- Wasserscheidentransformation

Segmentierung nach Homogenitätskriterien (1)

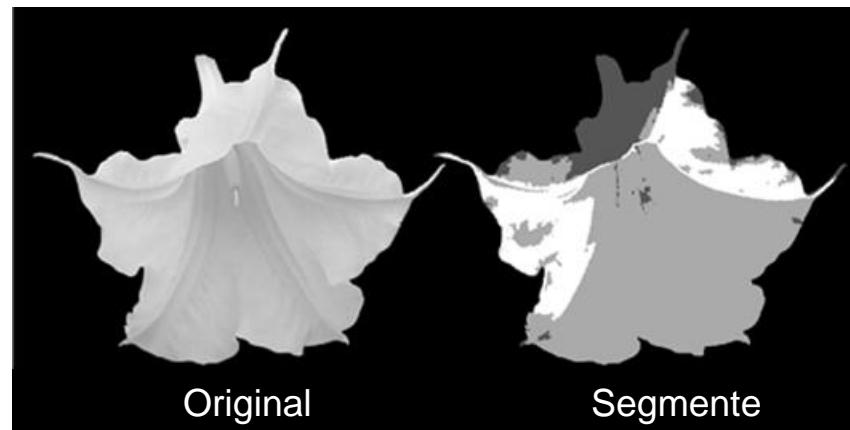
In der letzten Vorlesung: Segmentierungsansätze, die implizit oder explizit auf **Homogenitätskriterien** basieren, um benachbarte Pixel zu zusammenhängenden Segment zu gruppieren:

- histogrammbasierte Segmentierung: Homogenitätskriterien durch **globale** (d.h. für das gesamte Eingabebild festgelegte) **Schwellwerte**
- regionenbasierte Segmentierung: Homogenitätskriterien in Form von **regionenbezogenen Größen wie z.B. deren Intensitätsvarianzen**
- texturbasierte Segmentierung: Homogenitätskriterien durch **Texturen**

Segmentierung nach Homogenitätskriterien(2)

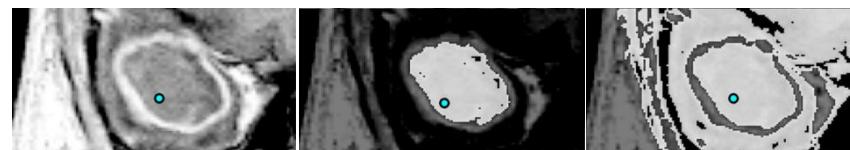
Zu beachten:

- das gewählte Homogenitätskriterium muss für *alle* Pixel des gesamten Segments gelten
- ↗ bei stärkeren Variationen der Segmentcharakteristik können Segmente dann fälschlicherweise in mehrere Segmente zerlegt werden
- ↗ es kommt zu einer Übersegmentierung: *



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

* Das Gegenteil heißt **Untersegmentierung** – s.
Bsp.: Originalbild, korrekte Segmentierung,
Untersegmentierung (v.l.n.r.)



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Segmentierung nach Diskontinuitätskriterien (1)

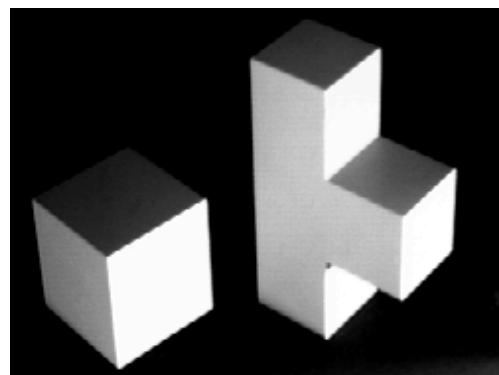
Bei stärkeren Variationen der Segmentcharakteristik bietet sich ein *komplementärer Ansatz* der Segmentierung an:

Segmentierung nach Diskontinuitätskriterien

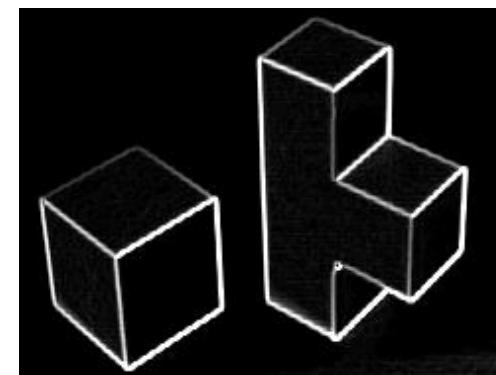
- Solche Ansätze zerlegen ein Bild nach Diskontinuitätskriterien an den *Segmenträndern* und sind daher a priori weniger anfällig gegenüber Variationen der Segmentcharakteristik
- Typische Diskontinuitätskriterien beziehen sich auf Intensitätsgradienten, die *Konturkanten* wiedergeben

Segmentierung nach Diskontinuitätskriterien (2)

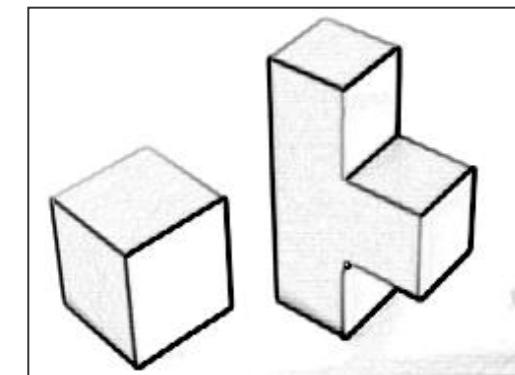
Kanteneigenschaften von Pixeln werden durch Operatoren der Bildverarbeitung (Low-Level Vision) wie z.B. durch den Sobel- oder Laplace-Operator hervorgehoben.



Grauwertbild



Antworten des
Sobel-Operators

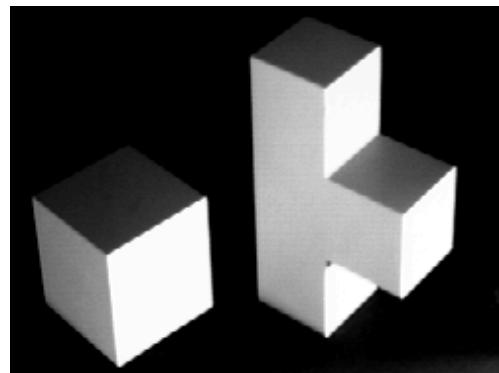


Invertierte Antworten des
Sobel-Operators

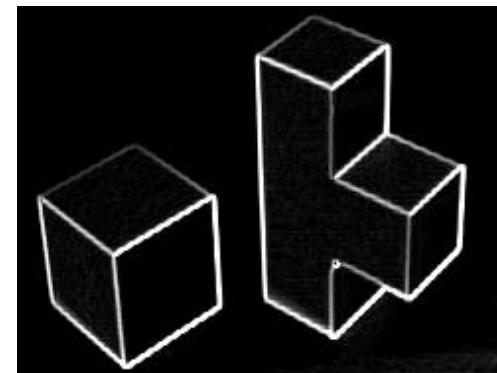
Segmentierung nach Diskontinuitätskriterien (3)

Segmente sind ableitbar, indem

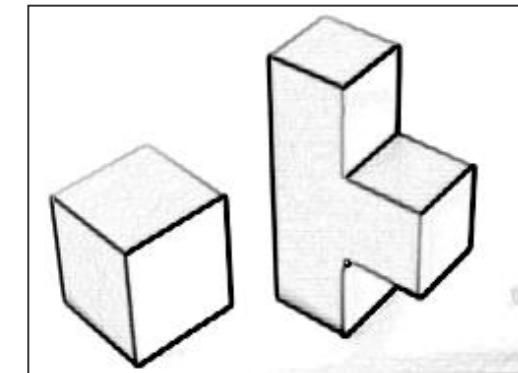
- aus allen Pixeln mit positiven Antworten des Kantenoperators vollständige und geschlossene Kantenzüge abgeleitet werden, die die Segmente über deren Ränder festlegen
- über eine Variante des *Region Labelings* ableitbar, bei der das Homogenitätskriterium festlegt, dass alle Pixel einer zusammenhängenden Region keine positiven Antworten des Kantenoperators zeigen



Grauwertbild



Antworten des
Sobel-Operators



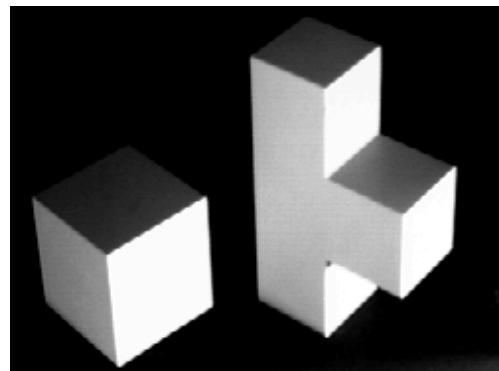
Invertierte Antworten des
Sobel-Operators

Segmentierung nach Diskontinuitätskriterien (4)

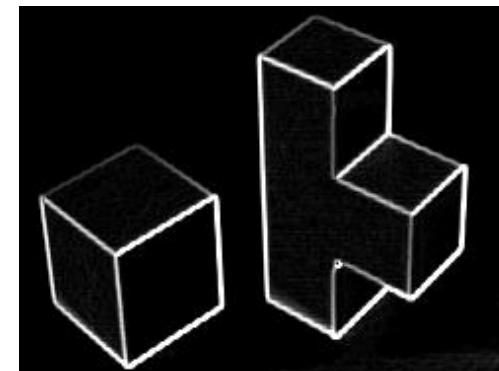
Kantenoperatoren geben wegen des Rauschens auch an solchen Stellen Operatorantworten, die keine Konturkanten abbilden.

Dem kann durch stärker angepasste Glättung begegnet werden.

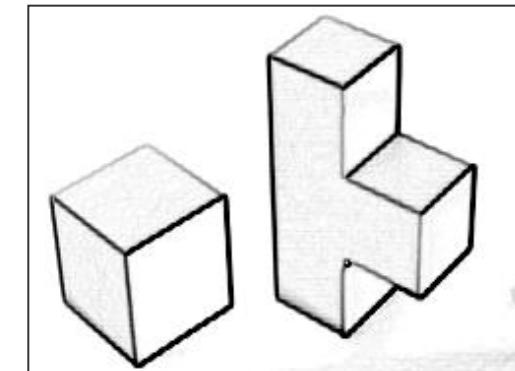
Zum anderen können die [Eigenschaften von Kantenzügen](#) genutzt werden wie z.B. beim [Edge Linking](#).



Grauwertbild



Antworten des
Sobel-Operators

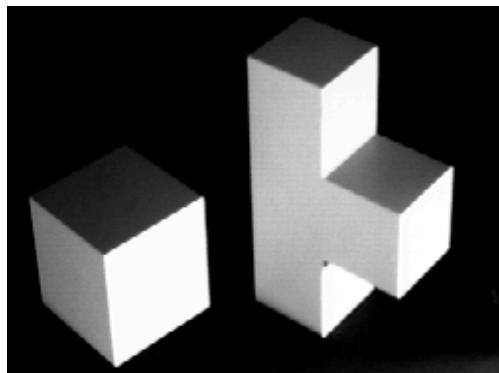


Invertierte Antworten des
Sobel-Operators

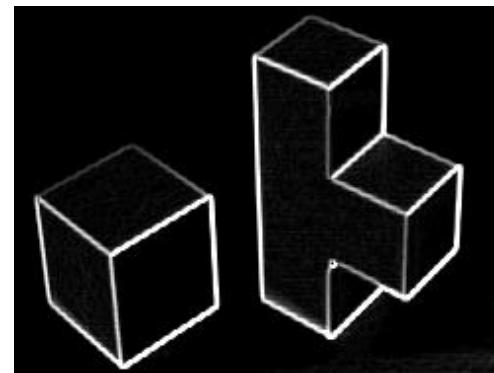
Eigenschaften von Kantenzügen (1)

(1) Der **Betrag des Gradienten** gibt die **Stärke der Kanteneigenschaft** an

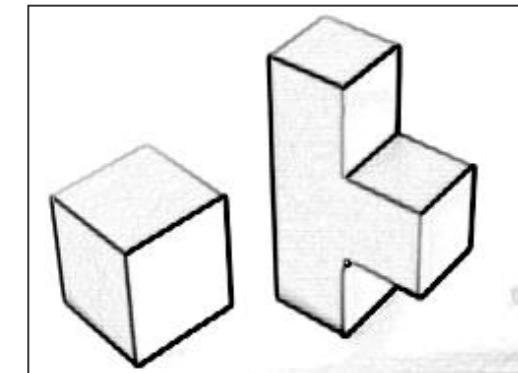
- ↪ Ist der Gradientenbetrag gering, so kann die zugrunde liegende Intensitätsänderung durch Rauschen bedingt sein.



Grauwertbild



Antworten des
Sobel-Operators

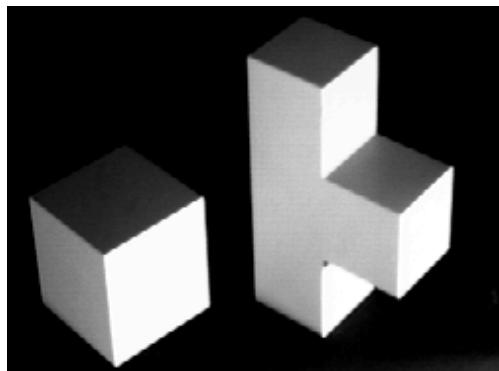


Invertierte Antworten des
Sobel-Operators

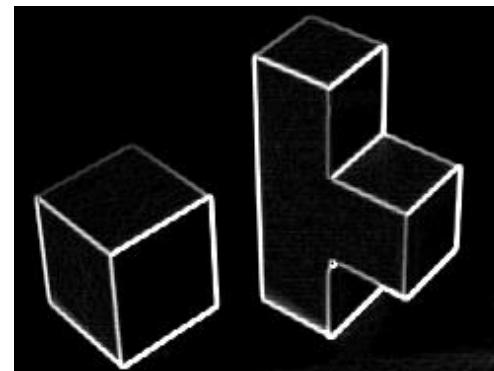
Eigenschaften von Kantenzügen (2)

(2) Aus der Richtung des Gradienten kann auf die lokale Richtung des Kantenzuges geschlossen werden.

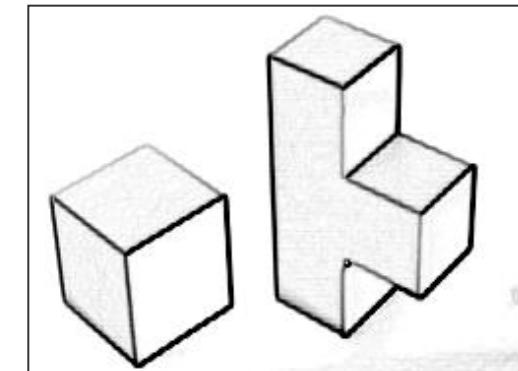
- ~ Die lokale Richtung des Kantenzuges sollte orthogonal zur Gradientenrichtung des betrachteten Kantenpixels sein.



Grauwertbild



Antworten des
Sobel-Operators

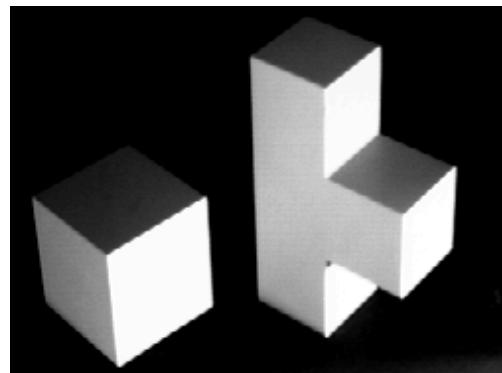


Invertierte Antworten des
Sobel-Operators

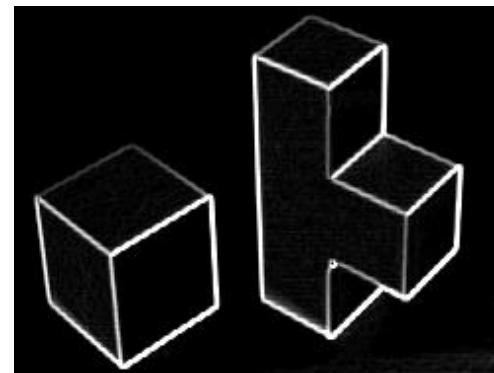
Eigenschaften von Kantenzügen (3)

(3) **Kantenzüge** sind i.A. kontinuierlich:

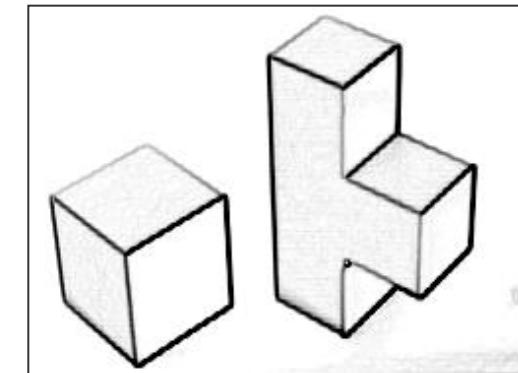
- ~ in der lokalen Umgebung eines Kantenpixels sollten weitere Kantenpixel mit ähnlichen Eigenschaften (genauer: Betrag und Richtung des Gradienten) zu finden sein.



Grauwertbild



Antworten des
Sobel-Operators



Invertierte Antworten des
Sobel-Operators

Binarisierung zur Selektion von Kantenpixeln (1)

Umsetzung von [Regel \(1\)](#): nur solche Pixel werden als Kantenpixel akzeptiert, deren Antwort auf einen Kantenoperator einen Schwellwert überschreitet.

Dies zeigt Ähnlichkeit zur Binarisierung durch einen Schwellwert t_B bei der histogrammbasierten Segmentierung:

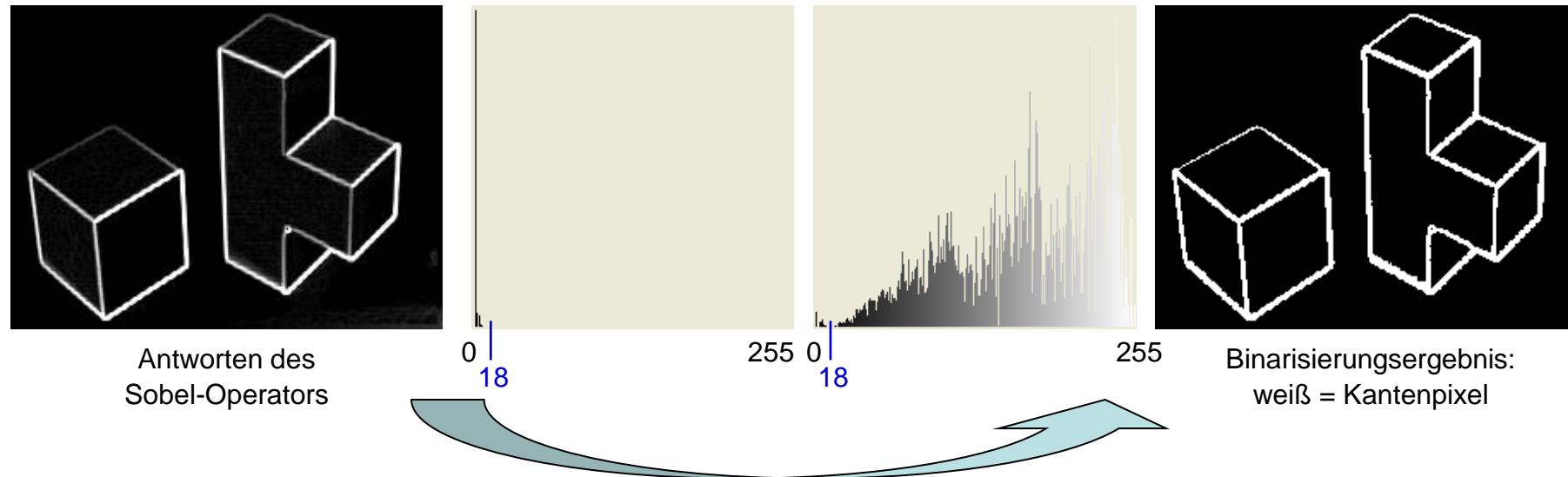
$$I(x, y) \in \begin{cases} \text{Objekt: } & I(x, y) \leq t_B, \\ \text{Hintergrund: } & I(x, y) > t_B. \end{cases}$$

Hier wird die [Binarisierung](#) aber *nicht* auf den originalen Intensitätswerten der Pixel, sondern [auf](#) deren [Gradientenbeträgen](#) zu deren Klassifikation in Kantenpixel und Flächen- bzw. Hintergrundpixel angewendet:

$$\nabla I(x, y) \in \begin{cases} \text{Kante: } & |\nabla I(x, y)| > t_B, \\ \text{Fläche/Hintergrund: } & |\nabla I(x, y)| \leq t_B. \end{cases}$$

Binarisierung zur Selektion von Kantenpixeln (2)

Für die gut ausgeleuchtete Laborszene führt das Ergebnis der Binarisierung zu hinreichend dichten und geschlossenen Ketten von Kantenpixeln :



Links die Gradientenbeträge der Intensitätswerte des Eingangsbildes. Das linke Intensitäts-histogramm zeigt die Häufigkeiten unskaliert. Im rechten Histogramm sind die Häufigkeitswerte mit den quadrierten Intensitätswerten gewichtet und zeigen besser das für die Binarisierung zu wählende Minimum beim Gradientenwert $t_B = 18$.* Rechts das Ergebnis der Binarisierung.

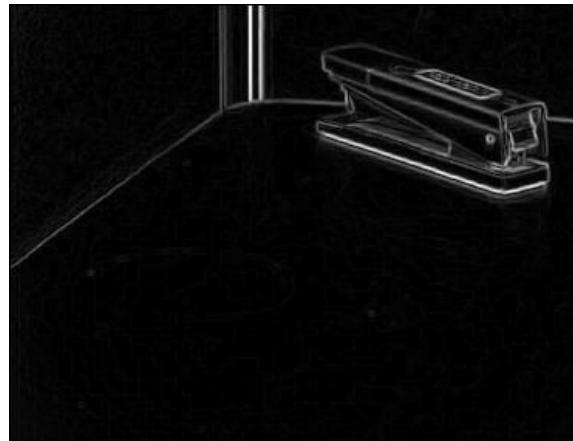
* Bei automatisierter Binarisierung wäre noch eine Glättung des bimodalen Histogramms zur automatisierten Bestimmung des Minimums angemessen. Siehe Vorlesung 7 zu histogrammbasierten Segmentierung.

Binarisierung zur Selektion von Kantenpixeln (3)

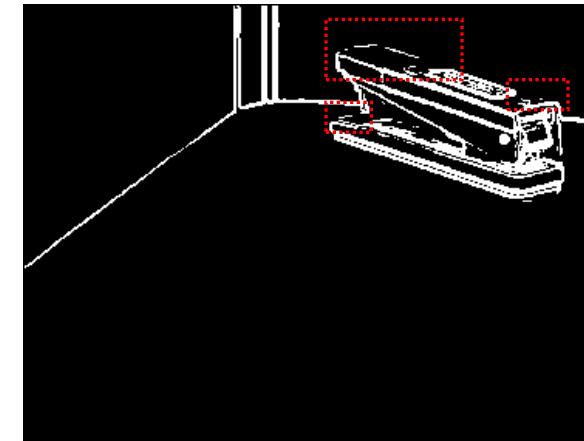
I. A. sind nach der schwellwertbasierten Binarisierung nicht mehr alle Kantenpixel über ihre Nachbarschaft miteinander verbunden, da es Bildbereiche gibt, in denen der Kontrast zwischen benachbarten Regionen zu gering ist (s. Bspl.).



Tacker-Grauwertbild*



Antworten Sobel-Operator



Binarisierungsergebnis für $t_B=40$

Edge Linking ist ein Ansatz, der Kantenpixel auch über **Lücken** in den Pixelnachbarschaften miteinander zu Kantenzügen verbinden kann.

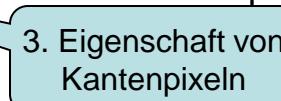
Bemerkung: im Bspl. hätte ein niedrigerer Schwellwert zwar zu weniger Lücken geführt, aber auch zu eher flächenhaften statt linienhaften Anordnungen von Kantenpixeln, was wiederum zu falschen Linienhypothesen führen kann. Prinzipiell kann aber der Kontrast zwischen zwei Regionen beliebig gering sein. Das Problem besteht also generell.

* Bildquelle: S. Russel, P. Norvig: Künstliche Intelligenz (2. Aufl.), Pearson Studium, 2004.

Algorithmus *Edge Linking*

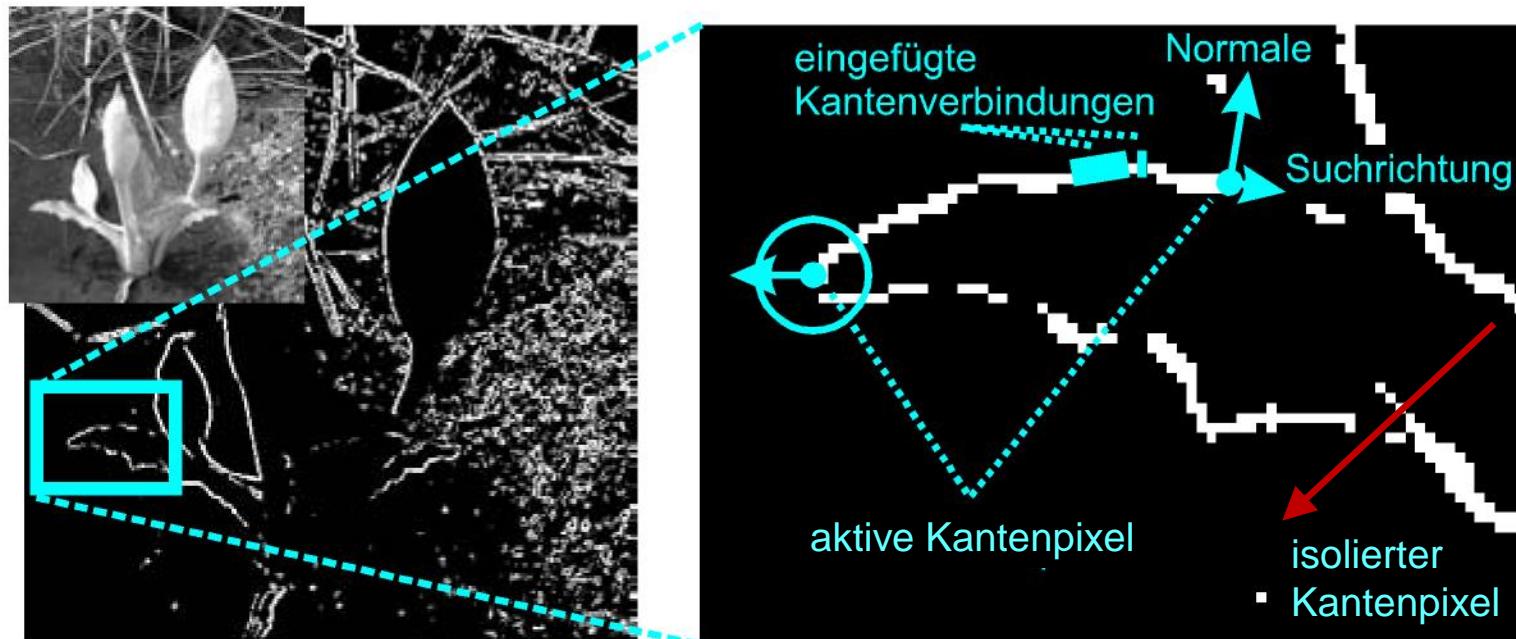
- 1) Markiere alle Pixel mit **hinreichend großem Gradientenbetrag** (Schwellwert) als *Kantenpixel*.

- 2) Markiere alle *Kantenpixel* als *unbearbeitet*.
- 3) Wähle nächstes *Kantenpixel*, das noch *unbearbeitet* ist und erkläre es als *aktives Pixel* p_a eines Kantenzuges k .

- 4) Wenn in der Umgebung $U(p_a)$ des *aktiven Pixels* p_a **in Kantenrichtung** (orthogonal zur Gradientenrichtung) *unbearbeitete Kantenpixel* p_i gefunden werden, die **ähnliche Gradientenrichtungen und -beträge** aufweisen, dann
 - a) markiere die Kantenpixel p_i als zum **Kantenzug k** gehörend,

 - b) erkläre die Kantenpixel p_i zu neuen *aktiven Pixeln*,
 - c) markiere das aktive Kantenpixel p_a als *bearbeitet*.
 - d) weiter mit Schritt 4., bis alle Kantenpixel als *bearbeitet* markiert sind.
- 5) Wenn in $U(p_a)$ Kantenpixel gefunden wurden, die bereits einem Kantenzug zugeordnet sind, dann wurde eine **Verzweigung** von Kanten gefunden.

Terminierung von *Edge Linking*

- Edge Linking terminiert, wenn alle Kantenpixel als *bearbeitet markiert sind.*
- Kantenpixel, die danach keinem Kantenzug zugeordnet wurden, werden als *Rauschen eingeordnet und entfernt.*



Parameter von *Edge Linking*

- Edge Linking zeigt drei heuristische Parameter in Schritt 4:
 - Größe der Umgebung $U(p_a)$ des *aktiven Pixels* p_a , in der Kanten-pixel mit *ähnlichen* Gradientenrichtungen und -beträgen gesucht werden
 - Maße bzw. Grenzwerte für diese Ähnlichkeiten von Gradientenrichtungen und -beträgen

Diese drei Größen müssen der Qualität der zu untersuchenden Bilder angepasst werden

Überbrückung von Lücken mit *Edge Linking*

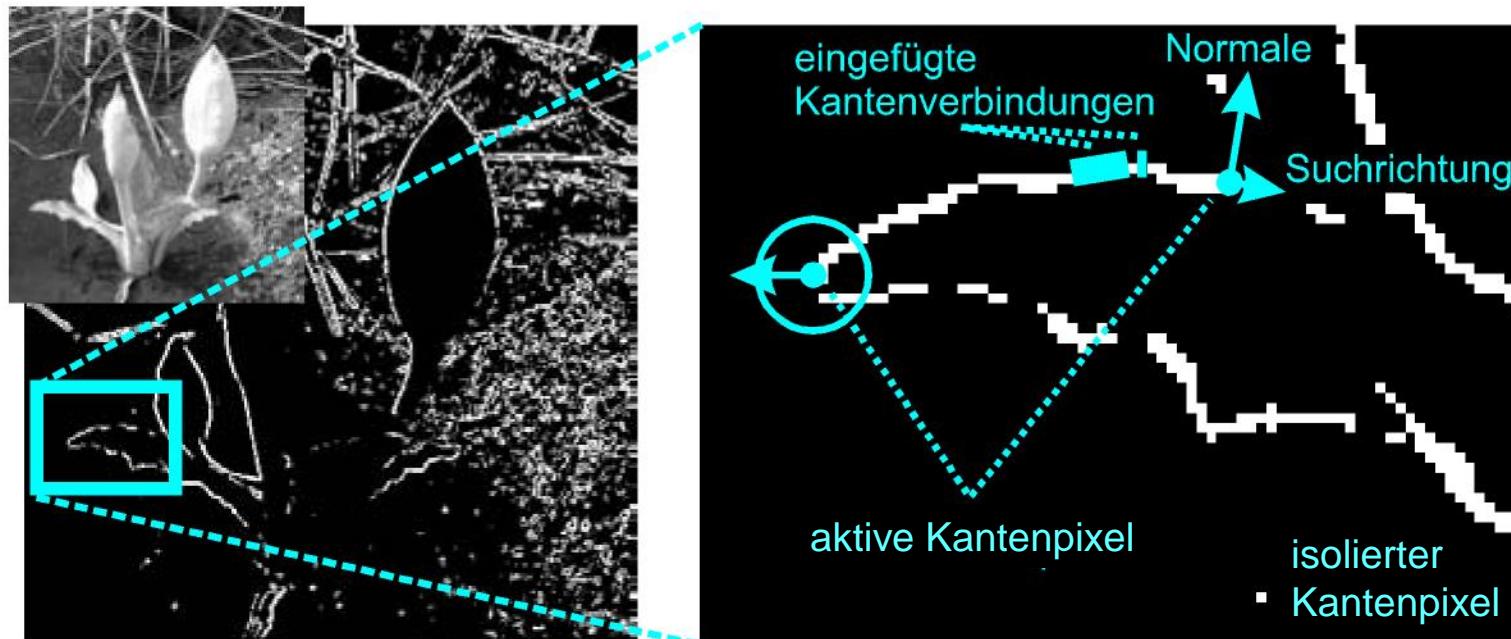
- Die Größe der Umgebung $U(p_a)$ definiert das Ausmaß, in dem Lücken in Kantenzügen überbrückt werden können
- Bsp!: die Umgebung $U(p_a)$ sei als ein in p_a zentriertes 5×5 -Pixelfeld definiert:
 - dann lässt dies Lücken von einem Pixel zu
 - Bsp. für Gradientenwerte in einem in p_a zentrierten 5×5 -Pixelfeld:

20	15	14	23	164
18	23	12	34	36
23	21	157	17	32
21	131	19	15	25
125	17	21	16	19

Einfügen von Kantenverbindungen in *Edge Linking*

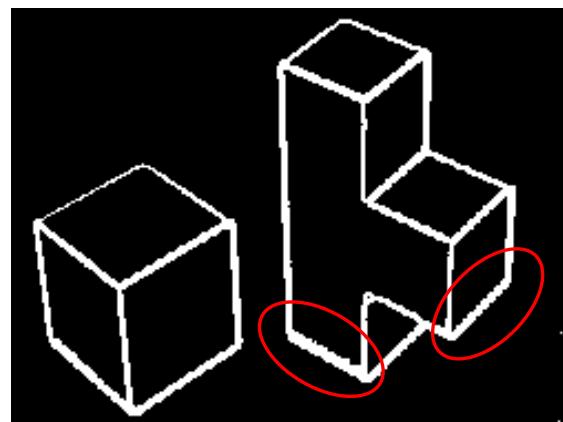
Die Überbrückung der Lücken erfolgt durch *lokale Kantenverbindungen*:

Die Konstruktion der Kantenverbindungen kann durch Linienalgorithmen aus der Computergrafik wie z.B. den Bresenham-Algorithmus oder durch Interpolation (z.B. lineare Interpolation oder Spline-Interpolation) erfolgen



Zum Ergebnis von *Edge Linking*

- Edge Linking erzeugt als Ergebnis Kanten im Sinne von linearen Gruppen von Kantenpixeln. Diese können breiter als ein Pixel sein.
- Um aus diesen „breiten“ Pixelgruppen symbolische Kantenbeschreibungen mit Parametern wie Länge, Orientierung und Krümmung abzuleiten, ist i.A. eine Nachbearbeitung durchzuführen.
- Dieser Schritt wird als *Kantenverdünnung* bezeichnet.



Breite Gruppen von Kantenpixel im Polyederbild und im Ausschnitt des Pflanzenbildes

Verdünnung durch Skelettierung

Eine Klasse von Verfahren, um breite lineare Pixelgruppen auf eine Breite von einem Pixel zu verdünnen, ist die sog. Skelettierungsverfahren.

Bei der **Skelettierung** wird aus einem flächenhaften Bildobjekt eine ein Pixel breite, innere Skelettlinie extrahiert.

Zeigt das Bildobjekt bereits eine eher linienhafte Form wie z.B. bei Buchstaben, so approximiert das **Skelett die Mittelachse des Objekts** (s. Abb. (a)).

Andernfalls ergeben sich keine angemessenen Ergebnisse in Hinblick auf eine Linienverdünnung (s. Abb. (b) und (c)).

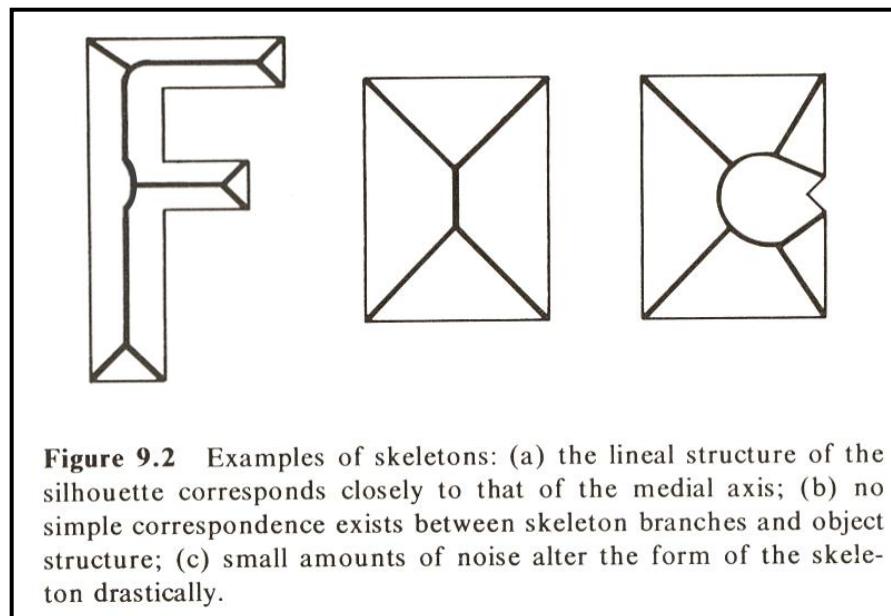


Figure 9.2 Examples of skeletons: (a) the lineal structure of the silhouette corresponds closely to that of the medial axis; (b) no simple correspondence exists between skeleton branches and object structure; (c) small amounts of noise alter the form of the skeleton drastically.

Bild korrigiert nach: Theo Pavlidis: *Algorithms for Graphics and Image Processing*. Springer, 1998.

Ziel der Skelettierung

Allgemein ist das Ziel der Skelettierung von Bildobjekten also

- a) die **Verdünnung**: das Erzeugen von dünnen (1 Pixel breiten) **Skelettlinien**, die in der Mitte der ursprünglichen Bildobjekten verlaufen
- b) die **Formerhaltung**: die Skelettlinien müssen die **ursprüngliche Form** des Bildobjekts widerspiegeln

Es gibt verschiedene Definitionen für das Skelett eines Bildobjekts.

Hier wird eine Definition verwendet, die auf den **Punkten des ursprünglichen Randes** des Bildobjekts und **deren Distanzen zum Skelett** basiert.

Definition von Skelett

Def. [Skelett]

Sei F eine planare Fläche mit Rand bzw. Grenze G und p ein Punkt in F .

Ein *nächster Grenzpunkt* g von p ist ein Punkt g in G derart, dass kein anderer Grenzpunkt g' in G existiert, dessen Abstand pg' kleiner als der Abstand pg ist.

Wenn p mehr als einen nächsten Grenzpunkt hat, heißt p *Skelettpunkt* von F .

Die Menge aller Skeletpunkte von F heißt *Skelett* oder *Mittelachse* von F .

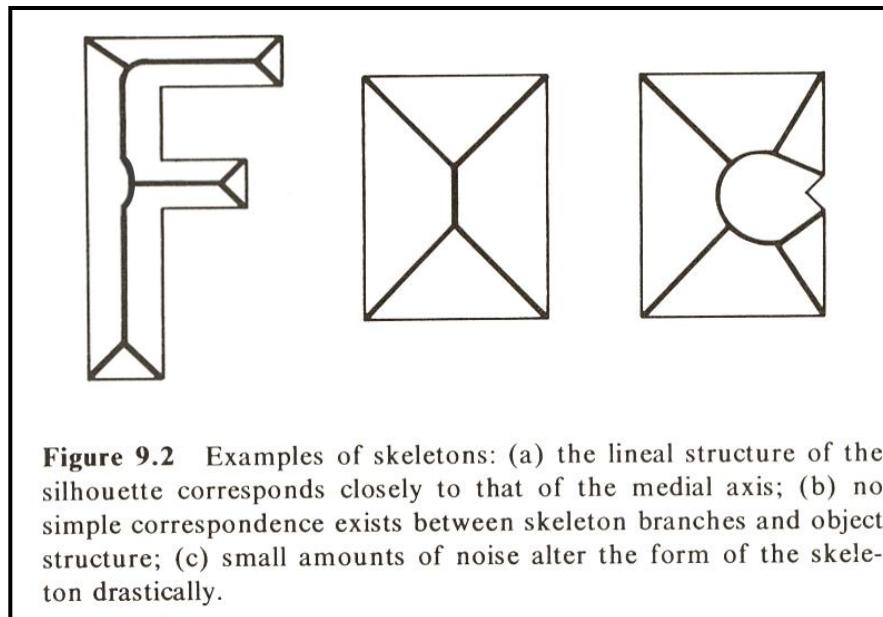


Figure 9.2 Examples of skeletons: (a) the lineal structure of the silhouette corresponds closely to that of the medial axis; (b) no simple correspondence exists between skeleton branches and object structure; (c) small amounts of noise alter the form of the skeleton drastically.

Bild korrigiert nach: Theo Pavlidis: *Algorithms for Graphics and Image Processing*. Springer, 1998.

Prinzip der Skelettierung

Prinzip der Skelettierung:

- Schrittweises Abtragen der Objektpixel vom Rand her bis zum Skelett
- Kern des Verfahrens: Entscheidung, welche Objektpixel zum Skelett gehören und nicht abzutragen sind, um den Zusammenhang des Bildobjekts zu gewährleisten.
- Die Erkennung der Skelettpixel erfolgt in vielen Ansätzen durch Inspektion der **8-Nachbarschaft**. Im Ansatz von Pavlidis* werden dazu zwei Muster überprüft:

A	A	A
0	P	0
B	B	B

A	A	A
A	P	0
A	0	1

* Theo Pavlidis: *Algorithms for Graphics and Image Processing*. Springer, 1998.

Skelettierung nach Pavlidis: die Muster

Semantik der Skelettpunktmasken:

- 0 = Pixel ist nicht gesetzt = Hintergrund
- 1 = Pixel ist gesetzt = Objekt
- in beiden Pixelgruppen A und B muss jeweils mindestens 1 Pixel gesetzt sein, also Objektpixel sein
- Evaluierung über beide Muster und ihre einmal bzw. dreimal um 90° rotierten Varianten:

A	A	A
0	P	0
B	B	B

A	A	A
A	P	0
A	0	1

- ~ Pixel P ist Skelettpixel, sobald P eines der beiden Muster bzw. einer rotierten Variante erfüllt
- ~ Skelettpixel dürfen nicht gelöscht werden

Skelettierung nach Pavlidis: die Methode

Prinzip der Skelettierung nach Pavlidis:

- Iteratives Abtragen der Kantenpixel von den Rändern her;
- Nicht abzutragen sind Skelettpixel:
 - Kantenpixel, deren Entfernung den Zusammenhang der Linie zerstört
 - Zur Erkennung der Skelettpixel sind lokale Inspektionen ausreichend
- Terminierung, wenn kein Kantenpixel mehr entfernt wurde; d.h.: alle verbliebenen Kantenpixel sind Skelettpixel.

Skelettierung nach Pavlidis: der Algorithmus

noch_Pixel_übrig := WAHR

SOLANGE *noch_PixelÜbrig* = WAHR

noch_PixelÜbrig := FALSCH

FÜR alle Nord-, Ost-, Süd- und West-Nachbarpixel *j*

FÜR alle Pixel *p* des Bildes

J WENN *p* ein Objektpixel ist und sein *j*-Nachbar nicht

J WENN *p* ein Skelett-Pixel ist

⊕

N

MARKIERE *p* als Skelett-Pixel

MARKIERE *p* als löschbares
Pixel

noch_Pixel_übrig := WAHR

⊕: hier
erfolgt der
Skeletttest
mit den
Masken

FÜR alle Pixel *p* des Bildes

J *p* ist löschbares Pixel

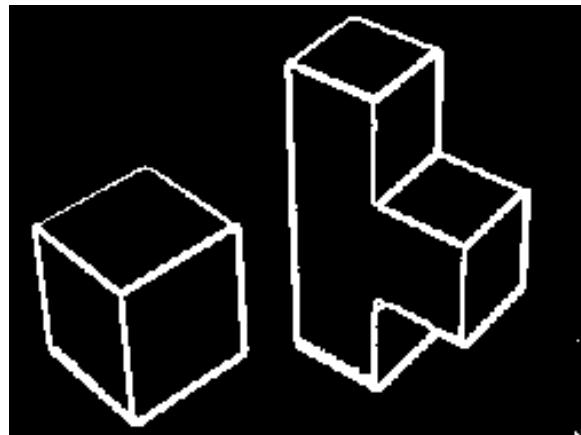
N

LÖSCHE *p*

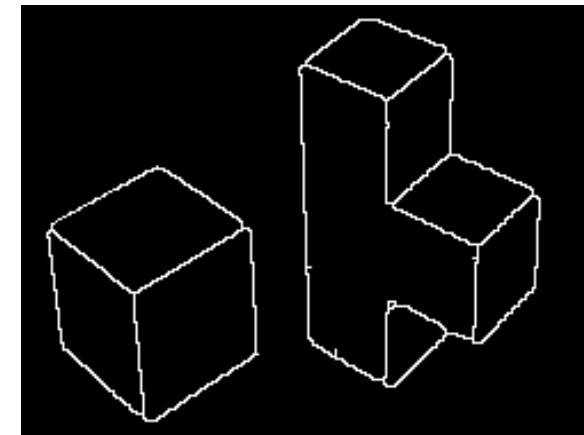
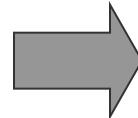
Struktogramm der Skelettierung nach Pavlidis

Skelettierung nach Pavlidis: Beispiel

Beispiel: Anwendung auf die Kantenpixel des Poylederbildes:



Kantenpixel



Extrahierte Skeletpixel
(nach Pavlidis)

Canny-Operator: Zielsetzung

Der Canny-Kantenoperator wurde 1986 von John Canny als **optimaler Kantendetektor** entwickelt*

Optimalität heißt dabei:

- hohe **Erkennungsrate** (hoher Recall):
Erkennung möglichst vieler Kanten der abgebildeten Szene
- hohe **Lokalisierungspräzision**:
möglichst ortsgenaue Lokalisierung der Kantenpixel
- **Eindeutigkeit**:
für jedes Kantenpixel soll es genau eine Detektorantwort geben

* Canny, J., *A Computational Approach To Edge Detection*, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

Canny-Operator: die Verarbeitungsstufen (1)

Mit der genannten Zielsetzung entwickelte Canny einen vierstufigen Algorithmus zur optimalen Kantendetektion.

- (1) Das Eingangsbild wird zunächst durch ein **Gauß-Filter** geglättet
- (2) Kantenpunkte werden durch den **Sobel-Operator** bestimmt
- (3) Breite Anordnungen von Kantenpunkten werden durch Unterdrückung von „Non-Maxima“-Kantenpunkten (engl. **non-maxima suppression**) verdünnt zu Linienbreiten von einem Kantenpunkt
- (4) Die in (2) erkannten und (3) selektierten Kantenpunkte werden durch Hysterese-Schwellwertbildung (engl. **hysteresis thresholding**) zu Kantenverläufen zusammengefügt

Canny-Operator: die Verarbeitungsstufen (2)

Der Canny-Algorithmus arbeitet mit drei Parametern σ , t_1 und t_2 :

- (1) Glättung durch Gauß-Filter mit Standardabweichung σ
- (2) Kantenerkennung mit Sobel-Operator:
 - (2.1) Gradientenbetrag: $|\nabla I(x,y)| \approx (\sqrt{S_x(x,y)^2 + S_y(x,y)^2})^{\frac{1}{2}}$
 - (2.2) Gradientenrichtung: $\Theta \approx \arctan(S_y(x,y) / S_x(x,y))$
gerundet zu 0° , 45° , 90° und 135° .
- (3) Non-Maxima-Unterdrückung: alle Kantenpixel, die in Gradientenrichtung Θ nicht ein lokales Maximum bilden, werden als solche verworfen
- (4) Fusion zu Kantenzügen: Kantenzüge werden mit Kantenpixeln begonnen, deren Gradient einen Schwellwert t_1 überschreitet. In Kantenrichtung (orthogonal zur Gradientenrichtung aus (2.2)) werden weitere Kantenpunkte des Kantenzuges erfasst. Die Kantenverfolgung bricht ab, wenn der Gradient des aktuellen Kantenpunktes unterhalb des zweiten Schwellwertes t_2 mit ($t_1 > t_2$) liegt.

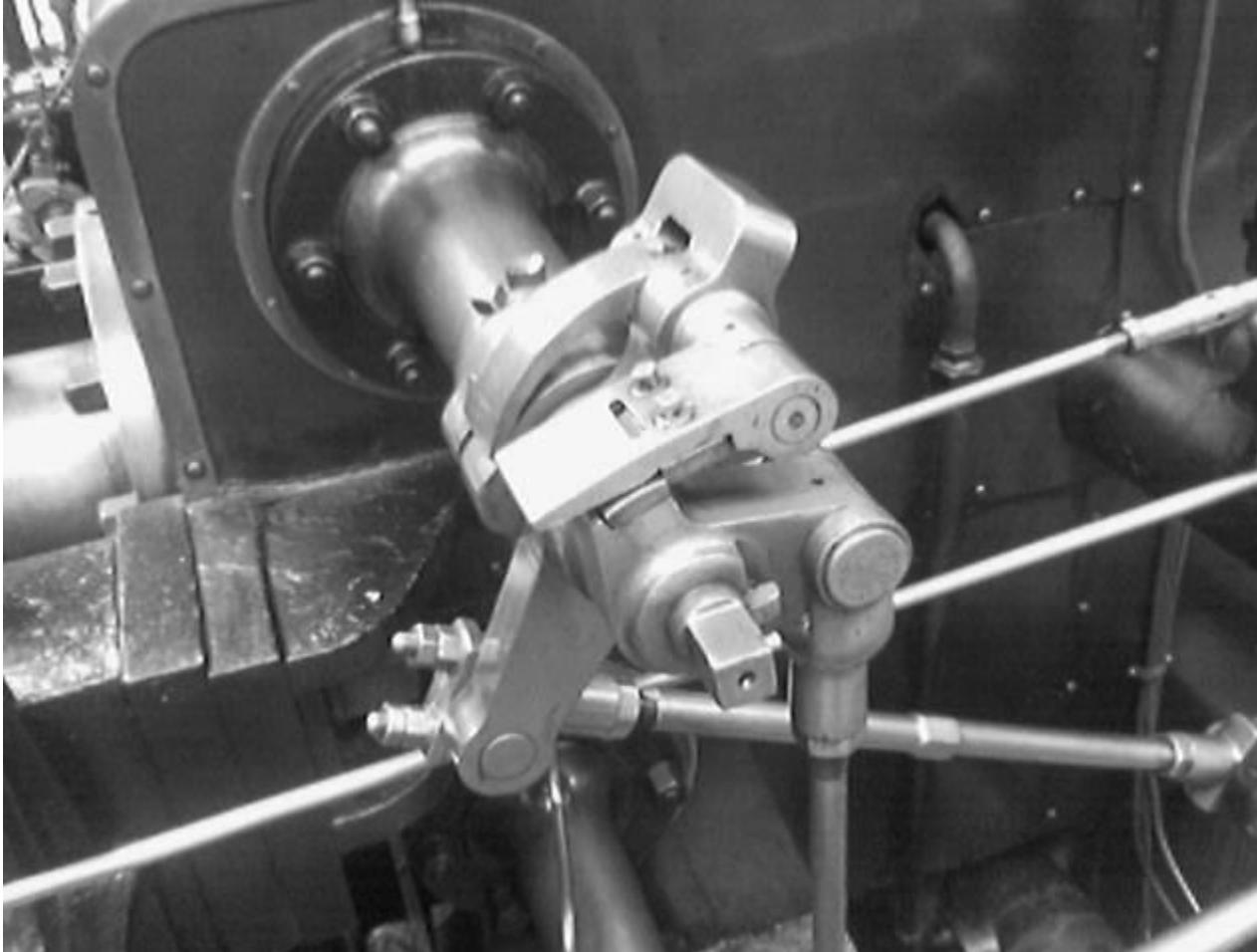
Canny-Operator: Hysterese

Bemerkung:

- Der vierte Schritt, also die Fusion zu Kantenzügen wird auch als **Hysterese-Verfahren** bezeichnet: „tracing edges with hysteresis thresholding“
- Als Hysterese bezeichnet man – insbes. in der Physik – ein Systemverhalten, bei dem die Ausgangsgröße nicht allein von der Eingangsgröße abhängt, sondern auch von dem vorherigen Zustand der Ausgangsgröße.
- Beim Canny-Operator ist die Binarisierung eben auch vom Zustand der Erkennung eines Kantenzuges abhängig: für den Start gilt Schwellwert t_1 , während der Gruppierung nach dem Start gilt Schwellwert t_2
- Hohe Werte von t_1 verhindern das Erkennen von schwach ausgebildeten Kanten. Kleinere Werte von t_2 verhindern Unterbrechungen der Kantenzüge. Canny empfiehlt in Abhängigkeit von erwarteten SNR ein Verhältnis von 3:1 bis 2:1 für die Schwellwerte t_1 und t_2

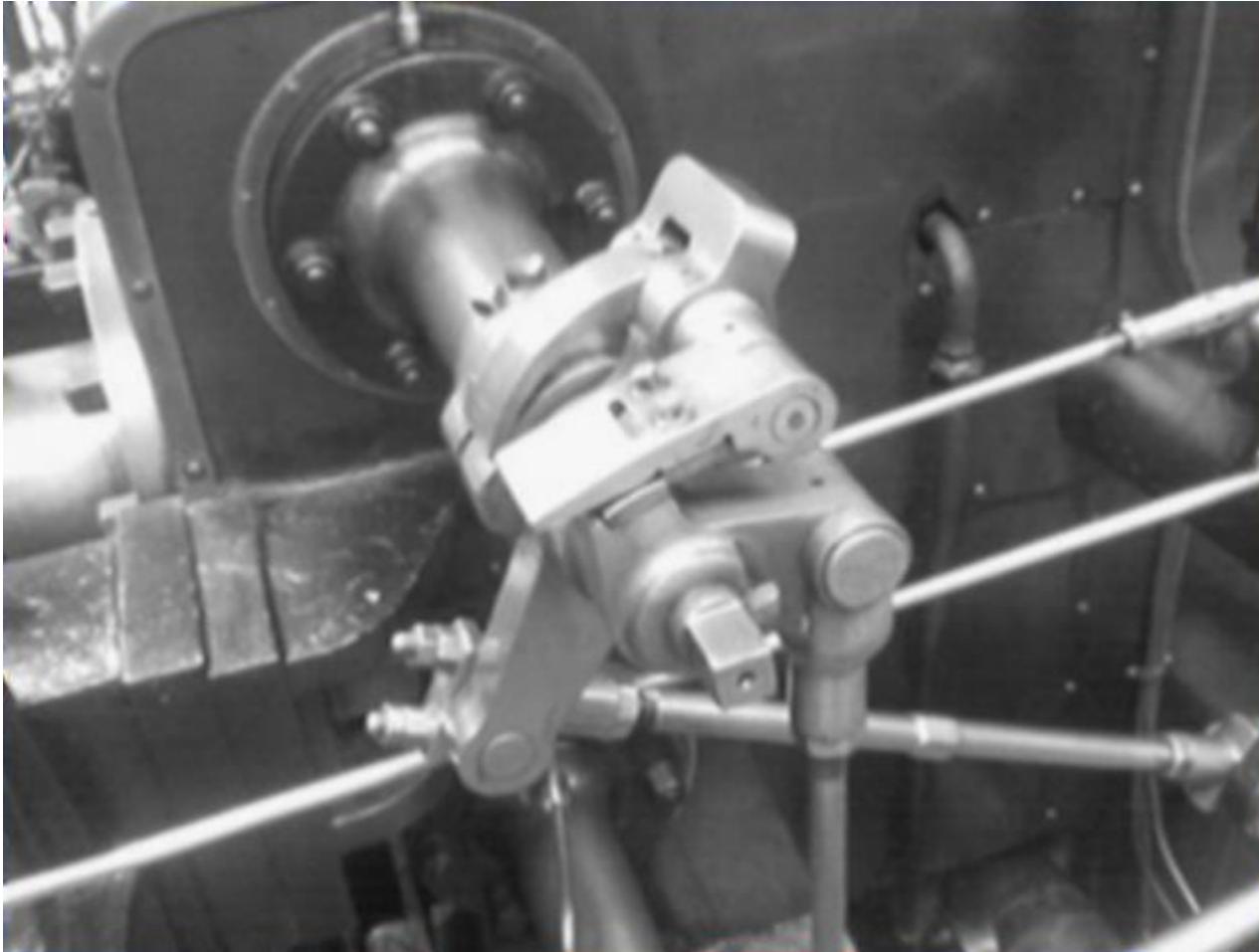
Canny-Operator: Beispiel (1)

Ausgangsbild für Canny-Opertor:



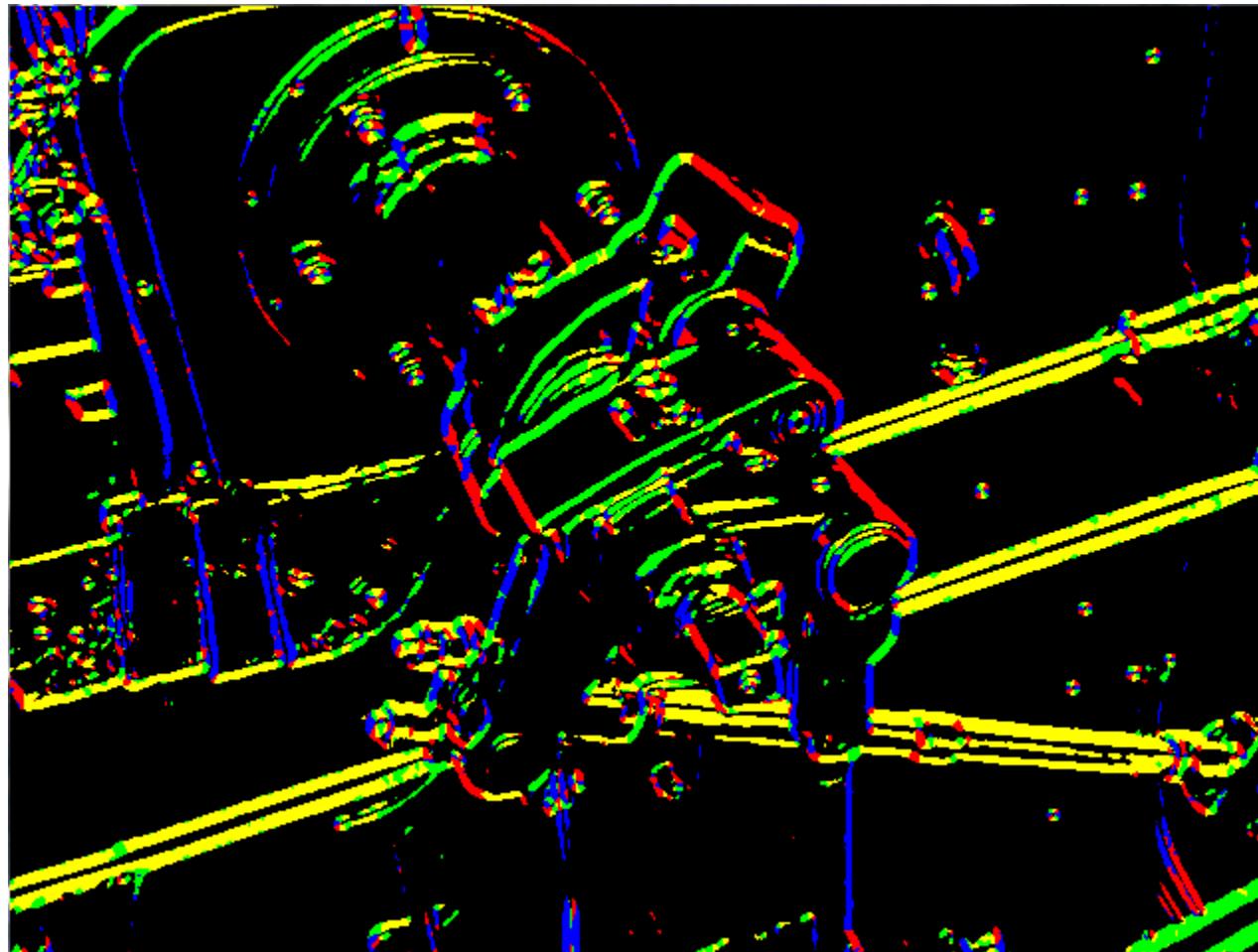
Canny-Operator: Beispiel (2)

Nach Anwendung von 5x5-Gauß-Filter mit $\sigma = 1,4$:



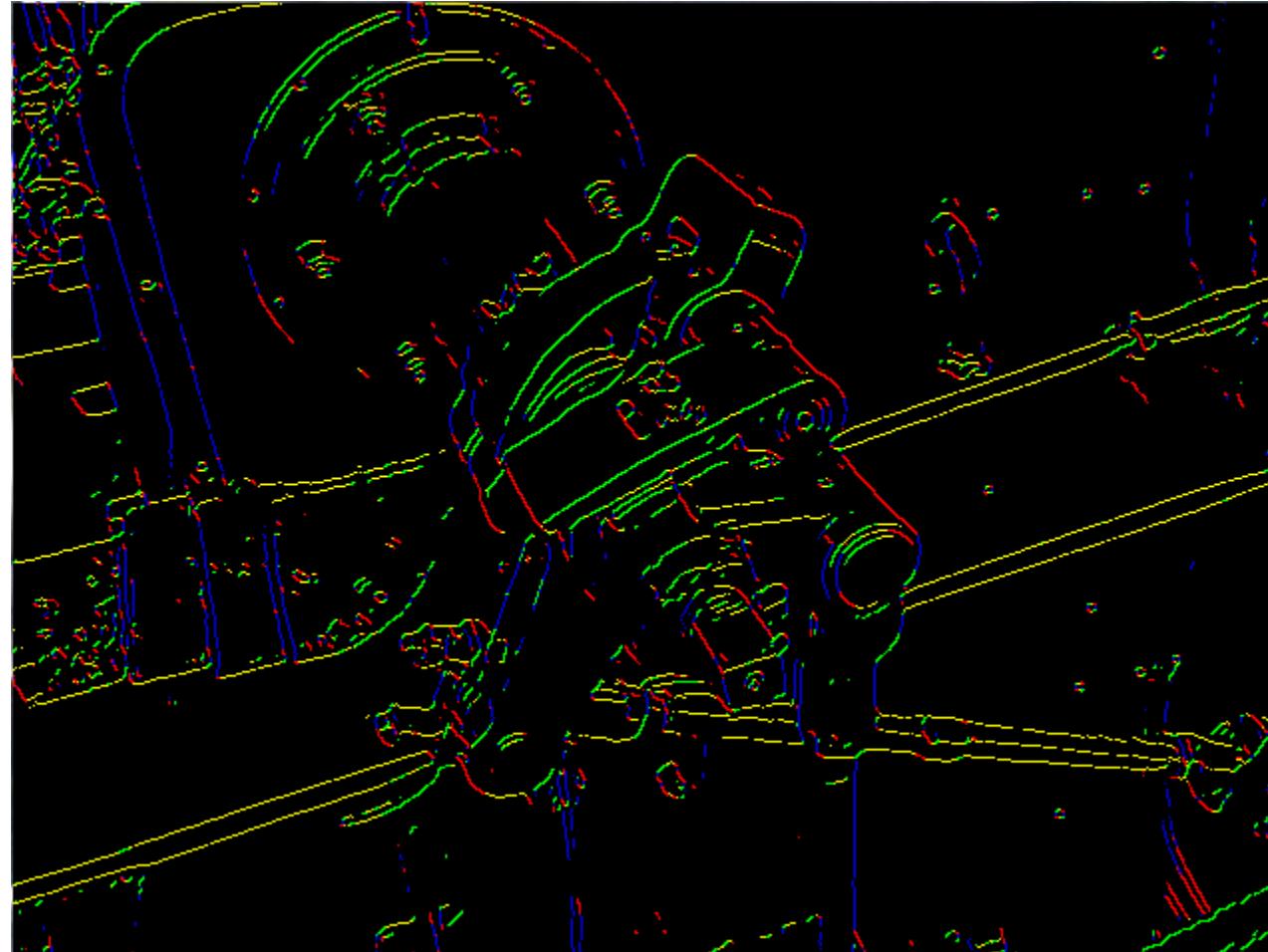
Canny-Operator: Beispiel (3)

Nach Anwendung von Sobel-Operator mit Farbkodierung der Klassifikation nach Kantenrichtungen: gelb = 0° , grün = 45° , blau = 90° , rot = 135°



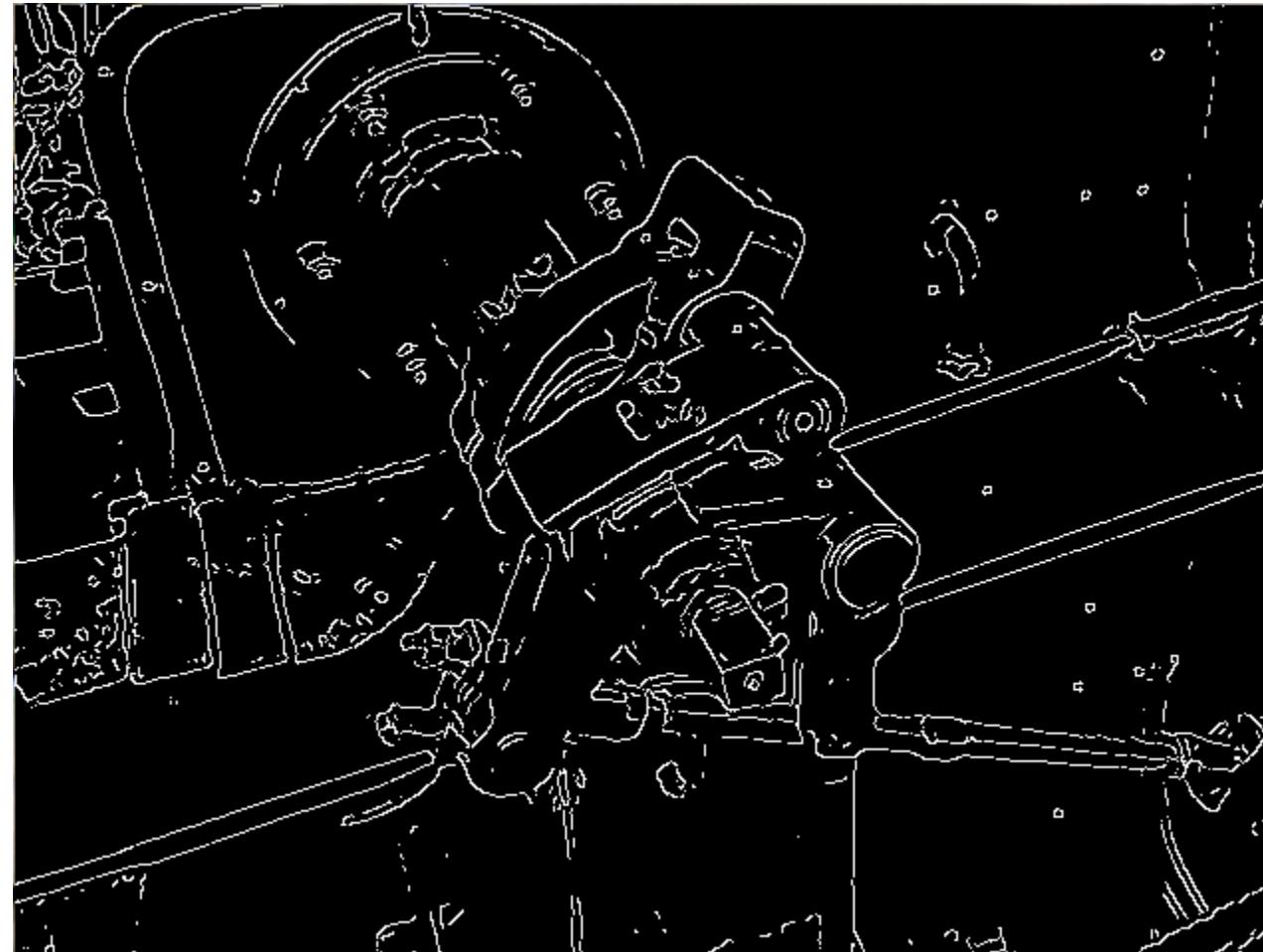
Canny-Operator: Beispiel (4)

Nach Anwendung der Non-Maximum-Unterdrückung in **Gradienten**richtung:



Low-Level-Vision: Canny-Operator (3.5)

Kantenzüge nach Hysterese-Schwellwertbildung:



Interaktive Suche nach optimalen Kantenzügen (1)

Wenn Start- *und* Endpunkt eines Kantenzuges bekannt sind,
dann ist ein optimaler Kantenzug zwischen beiden Orten als
Lösung eines Wegeproblems in einem Graphen ableitbar

Erzeugung des Graphen und Kostenfunktion:

- Überführe Bild $I = [I(x,y)]$ mit $Z \times S$ Pixeln in Graphen G_I mit $Z \times S$ Knoten
- Zwischen zwei Knoten k_1 und k_2 in G_I existiert eine Kante, wenn die entspr. Pixel p_1 und p_2 in I benachbart sind
- Jedem Knoten k werden Kosten $c(k)$ zugeordnet, die von den Eigenschaften des gesuchten Kantenzuges abhängen

Interaktive Suche nach optimalen Kantenzügen (2)

Die Kostenfunktion $c(k)$ sollte auf Gradienteninformation basieren:

- der Gradientenbetrag ist ungeeignet, weil er zu maximieren wäre und dabei alle nötigen, aber auch unnötigen Kantenpixel mitnehmen würde
- Tönnies (2005)* schlägt ein Abweichungsmaß von einer Modellgröße vor, z.B. einem Modellgradienten. Dieser kann als mittlerer Gradient von Startpixel s und Endpixel e definiert werden: $g_{\text{mod}} = \frac{1}{2} \cdot (|\nabla I(e)| + |\nabla I(s)|)$.
- Die **Kosten** $c(k)$ eines Knotens k ergeben sich dann als Abweichung seines **Gradientenbetrages** $g(k) = |\nabla I(k)|$ von g_{mod} :

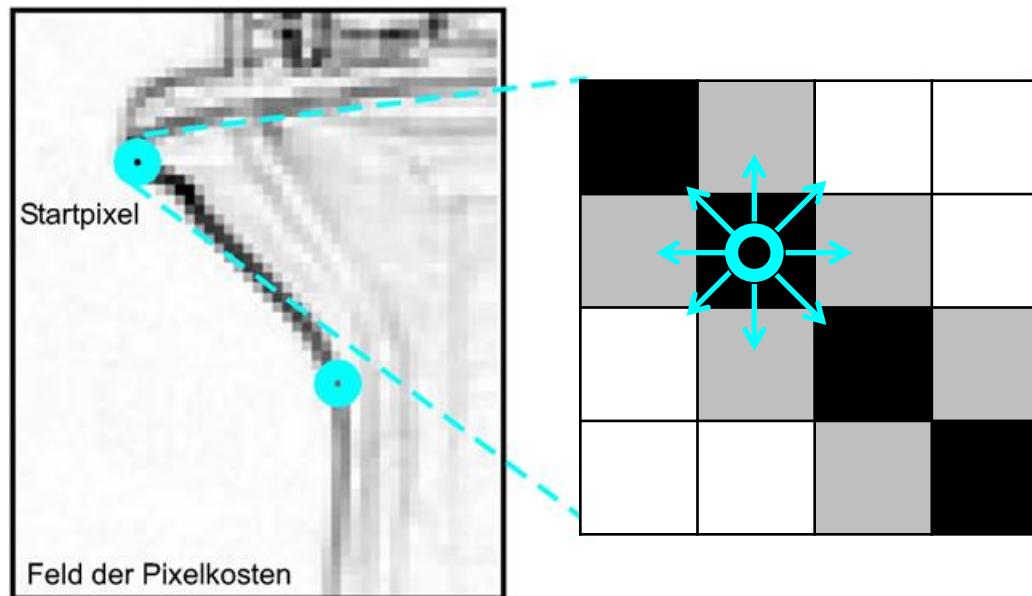
$$c(k) = | g(k) - g_{\text{mod}} |$$

* Es sind komplexere Kostenfunktionen anwendbar. In der Originalarbeit wird eine Kombination aus Nulldurchgängen der 2. Ableitung, den Gradientenbeträgen und Richtungsabweichungen der Gradienten eingesetzt (E. N. Mortensen, W. A. Barrett: Intelligent Scissors for Image Composition, ACM SIGGRAPH 1995, 191-198)

Interaktive Suche nach optimalen Kantenzügen (3)

Prinzip des Verfahrens:

Bei der Suche nach einem optimalen Kantenzug zwischen zwei Pixeln wird ausgehend vom Startpixel eine Suche nach einem Pfad mit minimalen Kosten zum Endpixel durchgeführt



Interaktive Suche nach optimalen Kantenzügen (4)

- Gegeben:
 - zusammenhängender Graph G_I ,
dessen Knoten mit den Pixeln des Bildes $I = [I(x,y)]$ korrespondieren
 - positive Knotenkosten,
die dem Abstand zu einem Modellgradienten entsprechen
 - Startknoten s und Endknoten e
 - Gesucht:
 - Pfad von Startknoten s zu Endknoten e mit minimalen Pfadkosten
- ↗ Umsetzung durch den Dijkstra-Algorithmus durchführbar

Interaktive Suche nach optimalen Kantenzügen (5)

Dijkstra-Algorithmus zur Suche nach optimalen Kantenzügen:

- Jedem Knoten k sind **Knotenkosten** $c(k)$ und **Pfadkosten** $p(k)$ zugeordnet.
Die Pfadkosten $p(k)$ sind die Kosten vom Startknoten s zum Knoten k
- Die Pfadkosten $p(s)$ des Startknotens s sind dessen Knotenkosten $c(s)$.
- Alle anderen Knoten $k \neq s$ sind mit $p(k) = \text{max_cost} = \sum_k c(k)+1$ initialisiert.
Diese sind höher als die höchsten Kosten, um k von s aus zu erreichen.
- Die Liste *active_nodes* wird mit dem Startknoten s initialisiert.
- Aus *active_nodes* wird immer der Knoten k_{min} mit minimalen Pfadkosten ausgewählt und entfernt.
- Die Pfadkosten aller Nachbarn von k_{min} werden bestimmt. Sind diese geringer als ihre bisherigen Pfadkosten, werden diese korrigiert und die entspr. Knoten in *active_nodes* eingefügt.

Interaktive Suche nach optimalen Kantenzügen (6)

- **Terminierung:** Auswahl des Endknotens e aus $active_nodes$:
 - Da alle Pfadkosten positiv sind und immer der Knoten mit minimalen Pfadkosten aus $active_nodes$ gewählt wird, können die Kosten jedes anderen Pfades zum Endknoten nur höher sein
- **Pfadausgabe:** Zurückverfolgung des Kantenzuges vom Endknoten e zum Startknoten s :
 - Die Rückwärtsverfolgung (*backtracking*) des gefundenen Pfades findet die Vorgängerknoten durch Vergleich der Pfadkosten mit den Pfadkosten an den Vorgängerknoten und den Knotenkosten:
Vorgängerknoten v zum Knoten k kann nur der Knoten sein mit

$$p(v) + c(k) = p(k)$$

Interaktive Suche nach optimalen Kantenzügen (7)

- Listing des Dijkstra-Algorithmus:

```
function Dijkstra (s,e);
    active_nodes = {s};
    p(s) = c(s);
    for all nodes v ≠ s do p(v) = max_cost;
    while v_min ≠ e do
        v_min = select_min_cost(active_nodes);
        active_nodes = active_nodes\{v_min};
        neighbours = get_neighbours(v_min);
        for all w ∈ neighbours do
            if [ p(v_min) + c(w) < p(w) ] then
                p(w) = p(v_min) + c(w);
                active_nodes = active_nodes ∪ {w};
            endif
        endfor;
    endwhile;
    return;
end;
```

Interaktive Suche nach optimalen Kantenzügen (8)

- Listing der Rückwärtsverfolgung:

```
function backtrack (s,e);  
    k = e;  
    nodelist = {e};  
    while k ≠ s do  
        neighbours = get_neighbours(k);  
        for all v ∈ neighbours do  
            if ( p(v) + c(k) = p(k) ) then  
                prev_node = v;  
            endif  
        endfor;  
        nodelist = nodelist ∪ {prev_node};  
        k = prev_node;  
    endwhile;  
    return nodelist ;  
end;
```

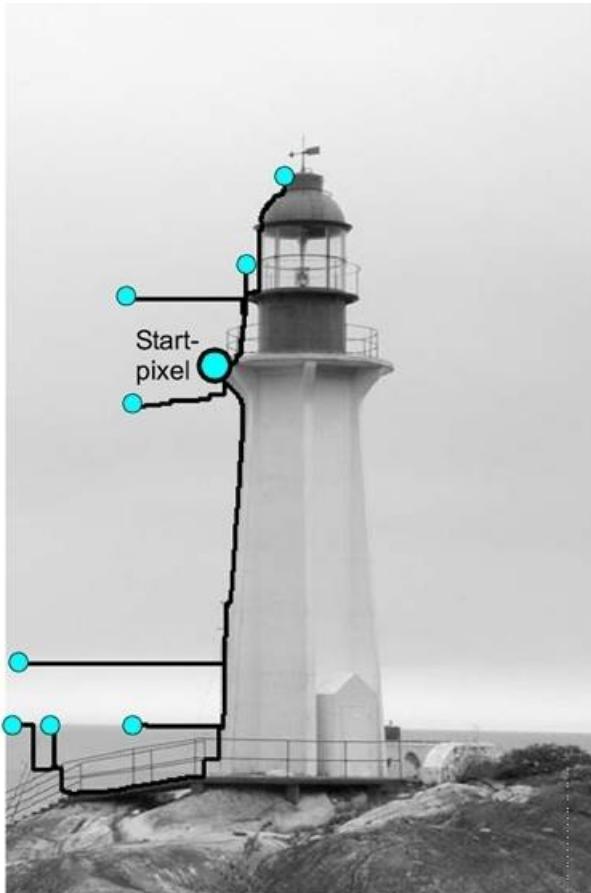
Interaktive Suche nach optimalen Kantenzügen (9)

Komplexität:

- Wird *active_nodes* als einfache Liste implementiert, ist die Zeitkomplexität des Dijkstra-Algorithmus $O(n^2+m)$ mit Knotenzahl n und Kantenzahl m . Bei Umsetzung von *active_nodes* in einer Baumstruktur folgt $O(n \cdot \log n + m)$.
- Die Rückwärtsverfolgung hat lineare Laufzeit in der Pfadpixelzahl und kann daher in Echtzeit umgesetzt werden. Dies führt zur sog. Variante des *Livewire*.

Interaktive Suche nach optimalen Kantenzügen (10)

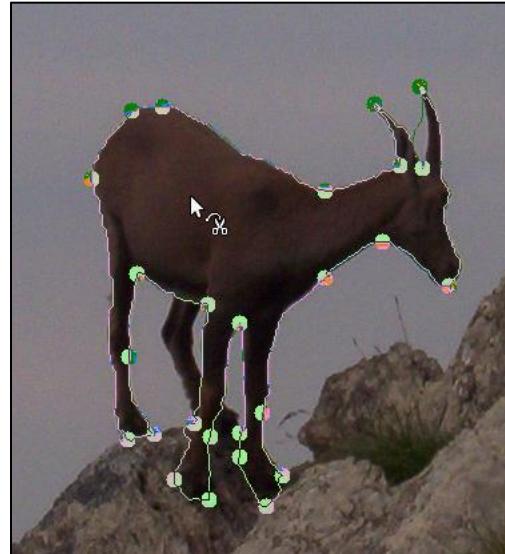
Die Livewire-Variante terminiert, wenn *active_nodes* leer ist. Daher werden kostenminimale Pfade vom Startknoten s zu beliebigen Endpunkten gesucht.



Interaktive Suche nach optimalen Kantenzügen (11)

Erweiterung des Verfahrens auf Folge von Kontrollpunkten:

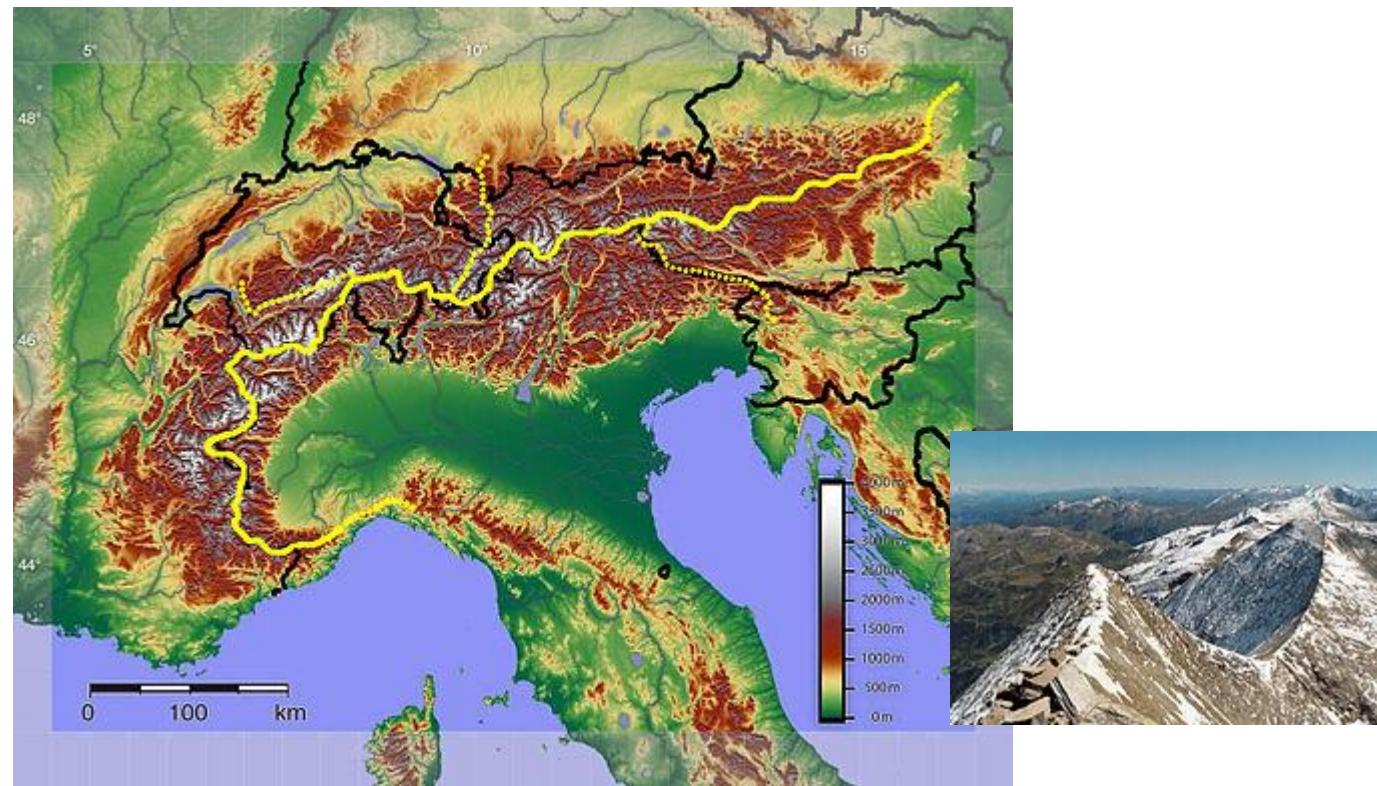
Werden aufeinanderfolgend angeklickte Cursor-Positionen als Folge von Start- und Endpunkten gewählt, ergibt sich ein interaktives Segmentierungsverfahren, das bekannt ist als *Intelligent Scissors*



"Each time you left-click with the mouse, you create a new control point, which is connected to the last control point by a curve that tries to follow edges in the image."

Wasserscheidentransformation (1)

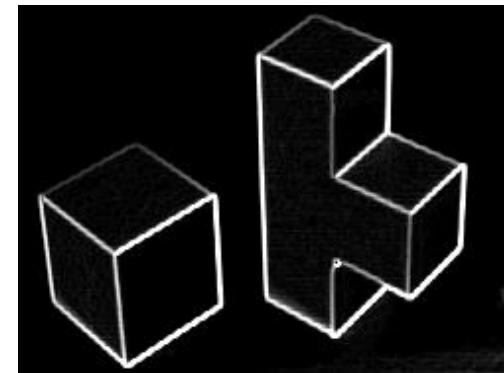
- In der Geographie trennt eine Wasserscheide Gebiete, die in unterschiedliche Senken entwässert werden
- ↗ Daher verlaufen Wasserscheiden i. A. entlang von Bergkämmen



Wasserscheidentransformation (2)

- Die Wasserscheidentransformation (WST) interpretiert die Gradienteninformation als Höheninformation.

Also entspricht die Identifikation der Wasserscheiden zwischen angrenzenden „Abflussgebieten“ der Ableitung der Grenzkonturen der Bildsegmente.



Gradienteninformation durch Sobel-Operator abgeleitet

- Der Flutungsalgorithmus erzeugt die Segmentierung des Bildes durch sukzessive Flutung des Gradientengebirges beginnend mit den lokalen Gradientenminima bzw. Senken bzw. Quellen.

Flutungsalgorithmus (1)

Jedem Pixel \mathbf{p} wird dessen lokaler Gradientenbetrag als Höhe $h(\mathbf{p})$ zugeordnet.

Für jede Flutungshöhe $h_{\text{aktuell}} = 0, \dots, h_{\text{max}}$

Für jedes neu überflutete Pixel \mathbf{p} mit $h(\mathbf{p}) = h_{\text{aktuell}}$

Wenn \mathbf{p} isoliert ist („Quelle“)

(d.h. \mathbf{p} ist nicht benachbart zu anderen bereits gelabelten überfluteten Pixeln \mathbf{p}' mit $h(\mathbf{p}') \leq h_{\text{aktuell}}$),
dann vergabe neues Segment-Label für \mathbf{p} .

Wenn \mathbf{p} eine Überflutungsregion erweitert

(d.h. \mathbf{p} ist benachbart zu bereits überfluteten und mit identischem Segmentlabel markierten Pixeln \mathbf{p}' mit $h(\mathbf{p}') \leq h_{\text{aktuell}}$),
dann übernehme dieses Segment-Label für \mathbf{p} .

Wenn \mathbf{p} Teil einer Wasserscheide ist

(d.h. \mathbf{p} ist benachbart zu überfluteten, aber mit unterschiedl. Segmentlabeln markierten Pixeln \mathbf{p}' mit $h(\mathbf{p}') \leq h_{\text{aktuell}}$),
dann vergabe das Label „Wasserscheide“ für \mathbf{p} .

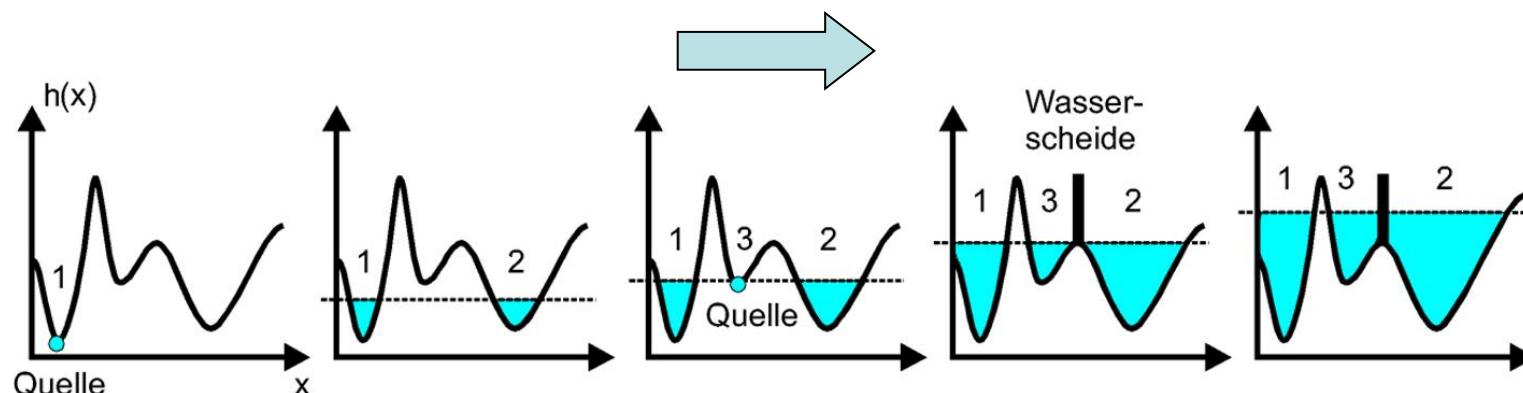
Label „Wasserscheide“
ist kein Segmentlabel

Flutungsalgorithmus (2)

Wenn die Höhe $h_{\text{aktuell}} = h_{\max}$ erreicht ist, haben alle Pixel ein Label erhalten: entweder ein Segment-Label oder das Label „Wasserscheide“.

Die Wasserscheidenpixel können in einer optionalen Nachbearbeitung einer der angrenzenden Regionen zugeordnet werden.

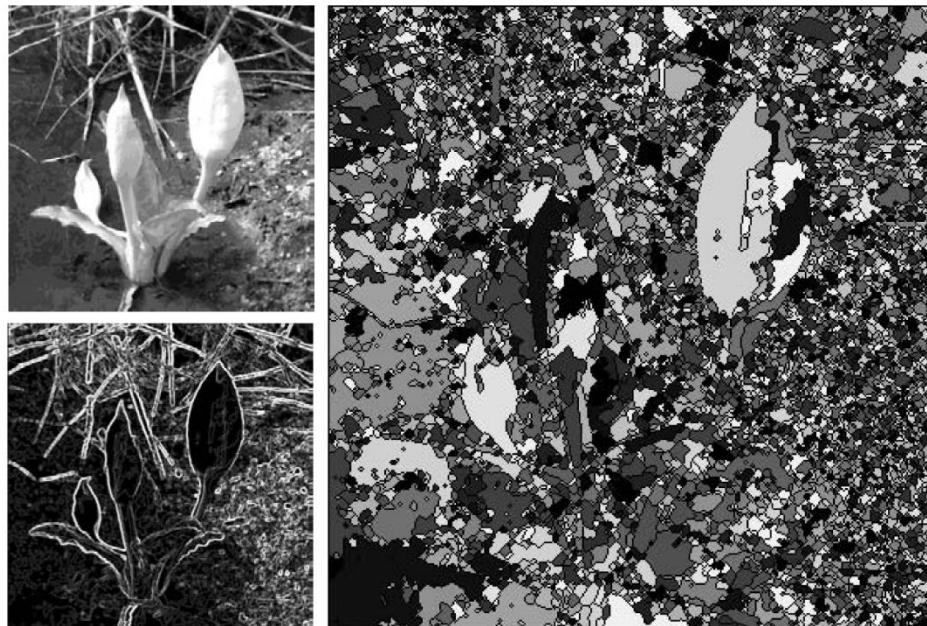
Die Abb. zeigt einige Stationen einer WST durch Flutung:



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Ergebnisse der Wasserscheidentransformation

Die WST führt leicht zu einer Übersegmentierung.

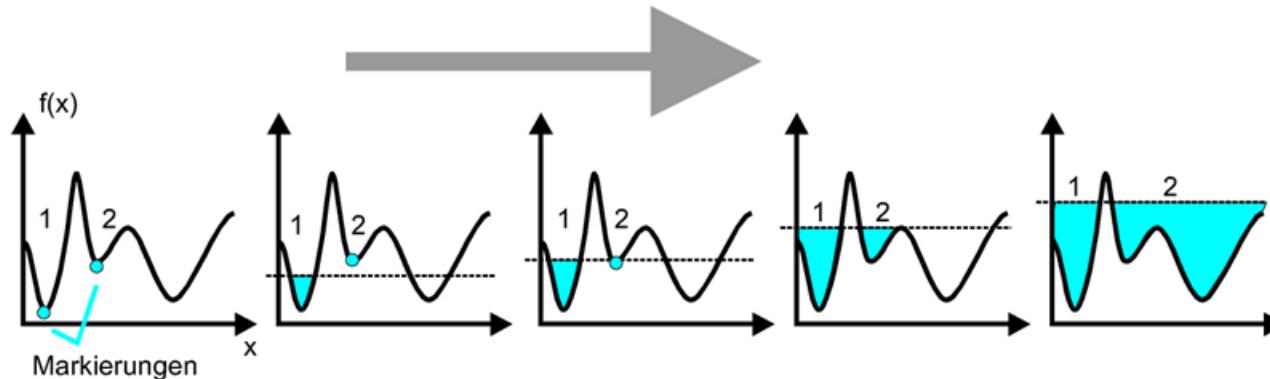


Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Hierarchische Wasserscheidentransformation (1)

Der Übersegmentierung durch WST kann auf zwei Arten begegnet werden:

- Hierarchische WST:
 - Wiederholte WST auf jeweils vorherigen WST-Resultaten erzeugt sukzessive größere Segmente, bis die Größe der Segmente den Erwartungen entspricht
 - Es entsteht eine **Hierarchie von Segmenten**, in der Segmente einer Stufe Segmente der früheren Stufen zusammenfassen
- Interaktive WST: Interaktive Markierungen erzeugen das Labeling vor dem Flutungsprozess



Zusammenfassung (1)

- Alternativ zu den Segmentierungsansätzen nach Homogenitätskriterien bietet sich ein komplementärer Ansatz der Segmentierung an, nämlich die Segmentierung nach Diskontinuitätskriterien.

Solche Ansätze zerlegen ein Bild nach Diskontinuitätskriterien an den Segmenträndern und sind daher a priori weniger anfällig gegenüber Variationen der Segmentcharakteristik.

- Eine schwellwertbasierte Binarisierung kann zur Auswahl von Kantenpixeln genutzt werden. I.A. werden wegen schwacher Kontraste u.Ä. jedoch nicht alle Kantenpixel über ihre Nachbarschaft miteinander verbunden sein.

Hierfür ist Edge Linking ein Ansatz, der Kantenpixel auch über Lücken in den Pixelnachbarschaften miteinander zu Kantenzügen verbinden kann.

Zusammenfassung (2)

- Edge Linking erzeugt als Ergebnis Kantenzüge im Sinne von linearen Gruppen von Kantenpixeln. Diese können breiter als ein Pixel sein.
- Eine Klasse von Verfahren, um breite lineare Pixelgruppen auf eine Breite von einem Pixel zu verdünnen, ist die sog. Skelettierung.
- Der Canny-Kantenoperator wurde 1986 von John Canny mit dem Ziel entwickelt, einen optimalen Kantendetektor darzustellen. Der Canny-Kantenoperator kombiniert Glättung, Kantenhervorhebung, Edge-Linking und Verdünnung.

Zusammenfassung (3)

- Werden Start- und Endpunkt eines Kantenzuges interaktiv bestimmt, dann ist ein optimaler Kantenzug zwischen beiden Orten als Lösung eines [Wegeproblems](#) in einem Graphen mit dem [Dijkstra-Algorithmus](#) ableitbar.

Mit diesem Ansatz sind dann die Livewire-Variante sowie [*Intelligent Scissors*](#) ableitbar.

- Die [Wasserscheidentransformation](#) ([WST](#)) extrahiert lokale Maxima der Gradientenbeträge als Orte trennender Wasserscheiden und leitet damit Bildsegmente als „Abflussgebiete“ zwischen angrenzenden Wasserscheiden ab.

Die hierarchische WST sowie die interaktive WST begegnen einer möglichen Übersegmentierung durch die normale WST.

Intelligente Sehsysteme

7 Fourier-Transformation und Skalenräume

Orts- und Frequenzraum, Fourier-Transformation
Multiskalenstrategien und -räume

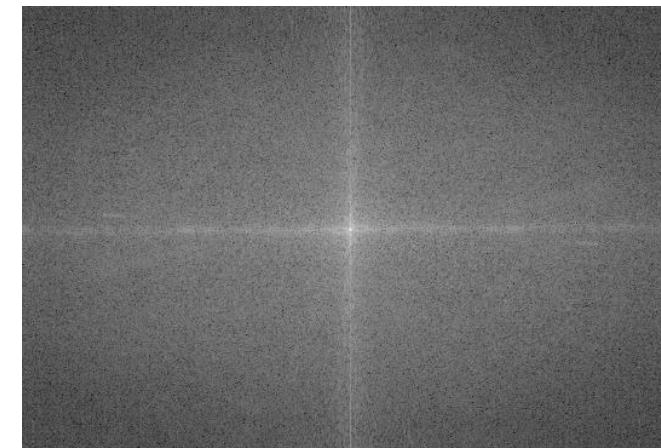
Florian Oßwald

Inhalt

- Basisfunktionen
- Orts- und Frequenzraum
- Fourier-Transformation
- Filter im Frequenzraum
- Multiskalen-Strategien
- Multiskalenraum
- Gauß-Pyramide
- Laplace-Pyramide

Fourier-Transformation

- Fourier-Transformation (FT): Beschreibung einer beliebigen Funktion als Summe von gewichteten periodischen Funktionen (Basisfunktionen) mit unterschiedlichen Frequenzen (Frequenzraumrepräsentation).
- Anwendungen sind u.a.:
 - Beschreibung des Informationsverlusts bei Digitalisierung
 - Restauration von linearen Störungen
 - Schnelle Filterung
- hier: Motivation von Multiskalen-ansätzen



Basisfunktionen

- Bilder wurden bereits als zweidimensionale Funktionen beschrieben.
- Allgemein:

Jede Funktion $f(n)$ kann als Summe von N gewichteten Basisfunktionen b_u aufgefasst werden:

$$f(n) = \sum_{u=0}^{N-1} w_u \cdot b_u(n)$$

Die Gewichte w_u bilden eine neue Funktion $w(u)$, die $f(n)$ zusammen mit den Basisfunktionen genau beschreibt.

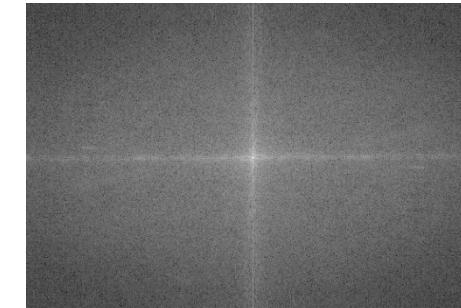
- Die Fourier-Transformation ist die Transformation von einer Ortsbasis in eine Frequenzbasis. Was heißt das?

Ortsraum und Frequenzraum

- Die Basisfunktionen der FT sind periodische Funktionen unterschiedlicher Frequenz und Amplitude. Durch diese wird die transformierte Bildfunktion repräsentiert. Man spricht von einer Repräsentation im **Frequenzraum**.
- Das ursprüngliche Bild wird Repräsentation im **Ortsraum** genannt.
- Funktionen im **Ortsraum** werden i.A. durch kleine Buchstaben (z.B. $f(i,j)$) und korrespondierende **Frequenzraumfunktionen** durch große Buchstaben (z.B. $F(u,v)$) bezeichnet.



Repräsentation im Ortsraum



Repräsentation im Frequenzraum

Basisfunktionen im Ortsraum

- Jede Basisfunktion ist ein Bild, das in genau einem Pixel den Wert 1 und in allen anderen Pixeln den Wert 0 hat.

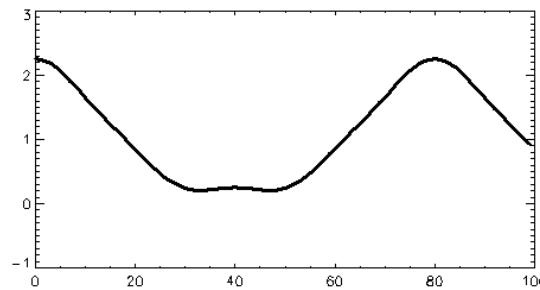
↗ Zwei Basisfunktionen b_k und b_l unterscheiden sich dadurch, dass sie den „1-Pixel“ an unterschiedlichen Stellen haben.

↗ Die vorgeschlagene Basis ist vollständig und eindeutig.

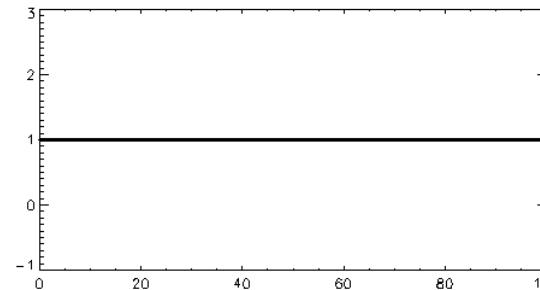
Basisfunktionen	Gewichtung
	\times <input type="text"/>
$=$	
	\times
$+$	
	\times
$+$	
	\times
$+$	
	\times <input type="text"/>
$+$	
	\times <input type="text"/>
$+$	
	\times
$+$	
	\times
\dots	
$+$	
	\times <input type="text"/>
$b_n(i,j)$	w_n

$$I = \sum w_n \cdot b_n$$

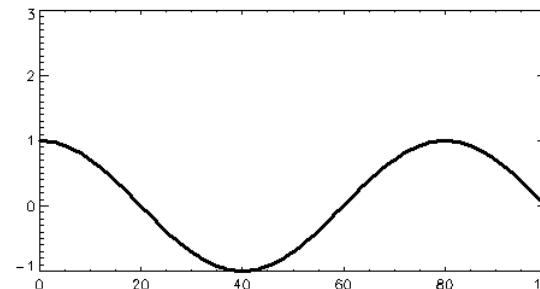
Basisfunktionen im Frequenzraum (1-D)



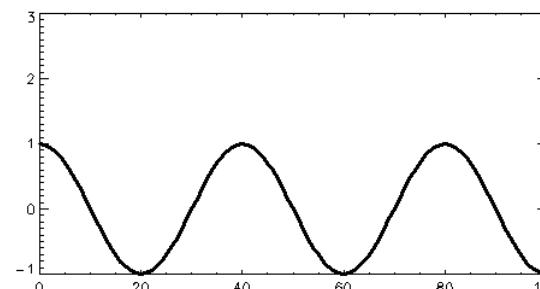
= $1.0 \times$



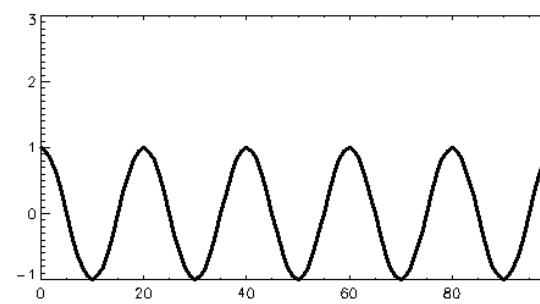
+ $1.0 \times$



+ $0.2 \times$

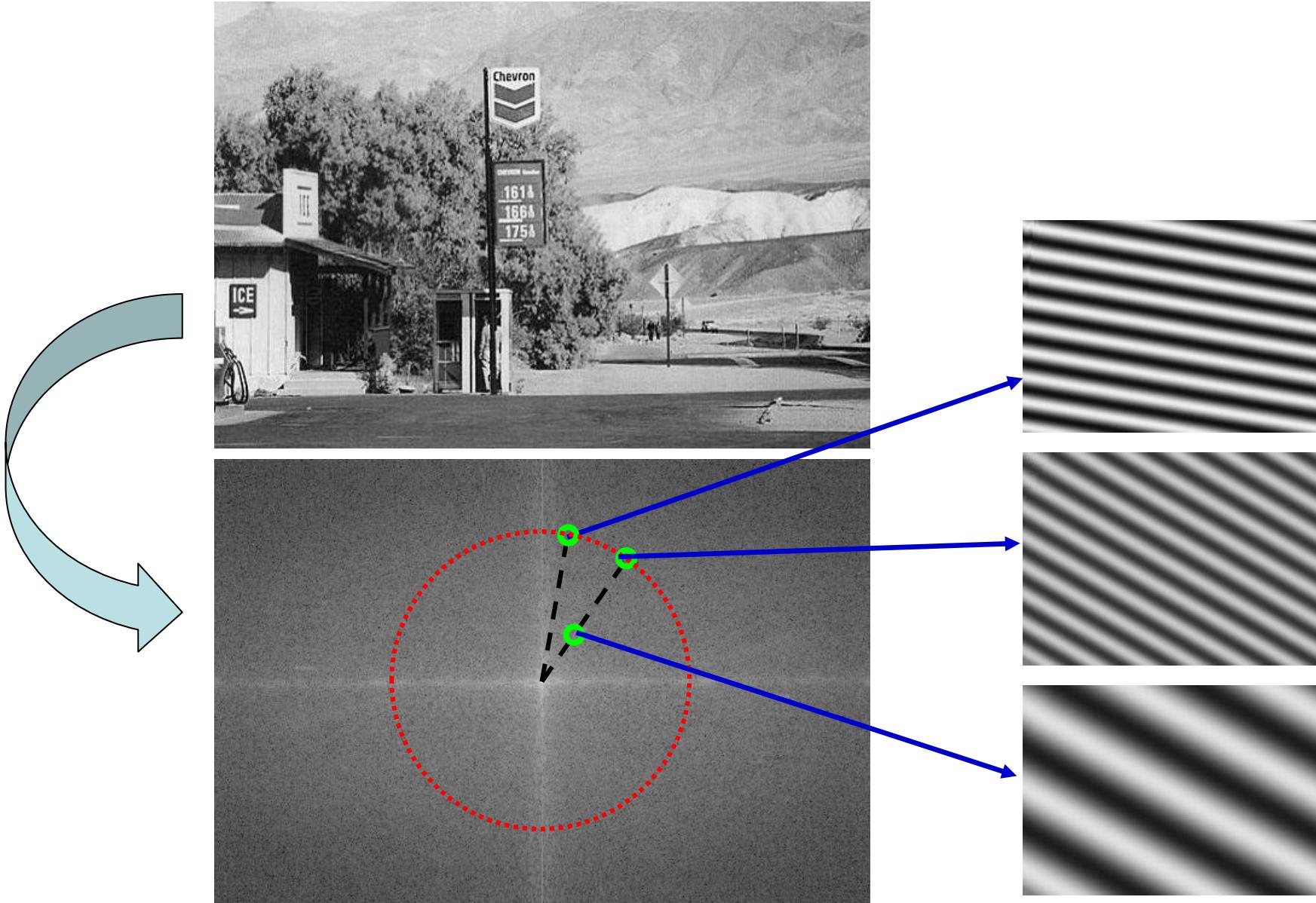


+ $0.05 \times$



Bildquelle: Klaus Tönnies: Grundlagen der
Bildverarbeitung, Pearson Studium, 2005.

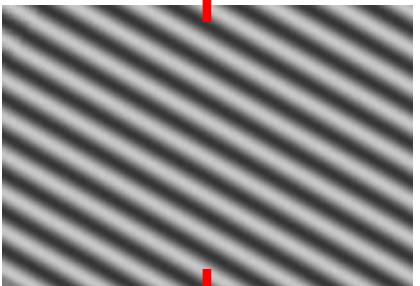
Basisfunktionen im Frequenzraum (2-D)



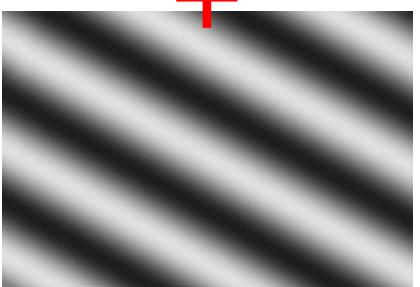
Frequenzbasis (2-D) – Forts.



+



+



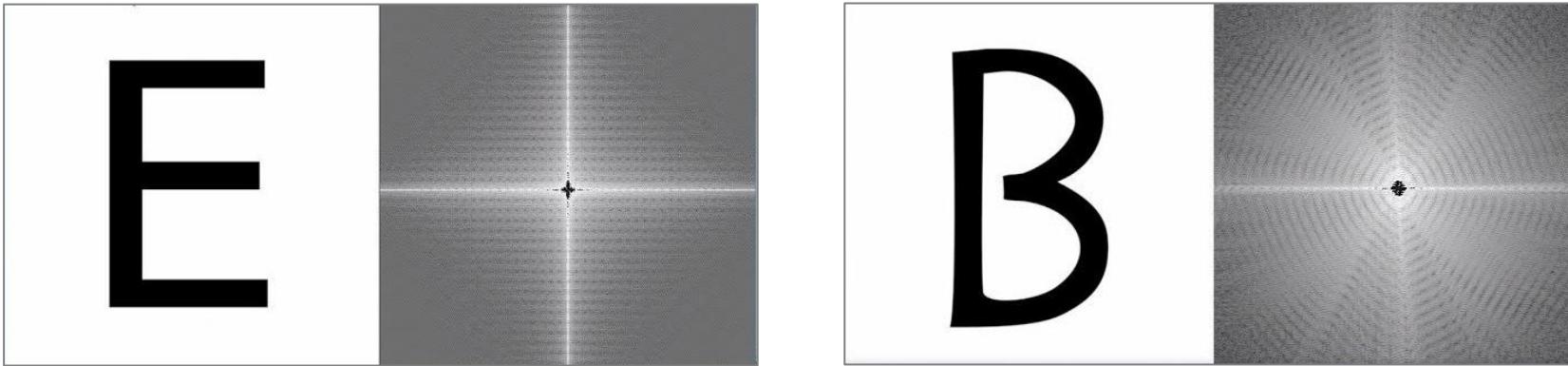
+

... =

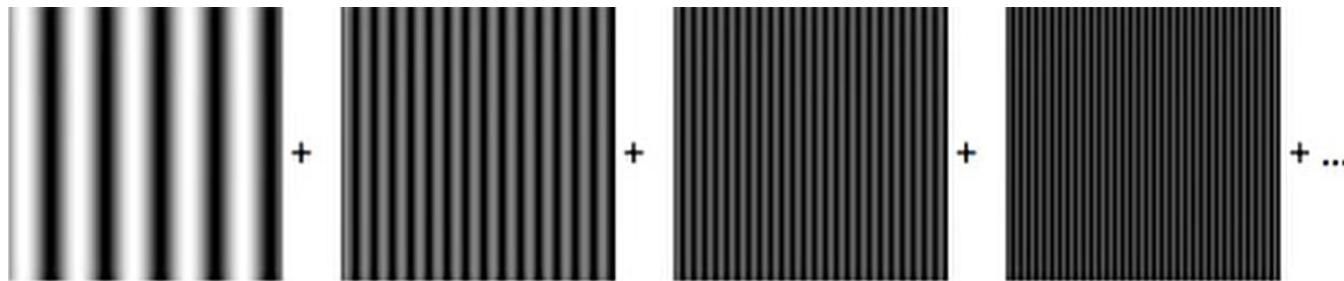


Bildquelle: Klaus Tönnies:
Grundlagen der Bildverarbeitung,
Pearson Studium, 2005.

Frequenzbasis (2-D) – Forts.



Bildquelle: Sergio Canu, PySource: Fourier Transform – OpenCV 3.4 with python 3 Tutorial 35



Summed image



Bildquelle: Michelle Dunn, Swinburne Univ. of Technology, Fourier transforms in image processing

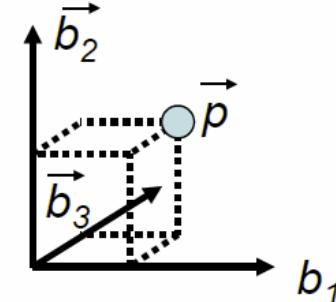


Invertierbarkeit

- Die Repräsentation und Verarbeitung von Bildern im Frequenzraum sollte auch im Ortsraum wiedergebbar sein \leadsto Invertierbarkeit der Transformation
- Allg.:
 - Transformation: „Projektion“ der Bildfunktion auf die neue Basis
 - Inverse Transformation: „Rückprojektion“ auf die alte Basis
 - Eine Transformation ist invertierbar,
wenn die Basisfunktionen eine **orthogonale Basis** bilden
 - 1) Was ist Orthogonalität für Funktionen?
 - 2) Wann bilden Basisfunktionen eine orthogonale Basis?
- Orthogonalität, Projektion, Transformation etc. sind Begriffe aus der **Vektoralgebra** und können auch für Funktionen genutzt werden

Orthogonale Vektorbasis

- N Vektoren bilden eine Basis für einen N -dim. Raum, wenn sie alle **orthogonal** zueinander sind.



- Zwei Vektoren sind **orthogonal**, wenn ihr **Skalarprodukt** Null ist.

$$\vec{b}_i \cdot \vec{b}_j = 0, \quad \text{für } i, j = 1, 2, 3 \wedge i \neq j$$

- Transformation: Projektion der Funktion auf die neue Basis.

Die Projektion eines Vektors \vec{p} auf einen Basisvektor \vec{b}_i ist durch das normierte **Skalarprodukt** zwischen ihnen gegeben.

$$\vec{p}^B = \left(\frac{\vec{p} \cdot \vec{b}_1}{\|\vec{b}_1\|}, \frac{\vec{p} \cdot \vec{b}_2}{\|\vec{b}_2\|}, \frac{\vec{p} \cdot \vec{b}_3}{\|\vec{b}_3\|} \right) \circ$$

Bildquelle: Klaus Tönnies:
Grundlagen der Bildverarbeitung,
Pearson Studium, 2005.

Funktionen statt Vektoren

- Eine Funktion $f(x)$ sei durch vollständige Aufzählung aller Werte eindeutig definiert: $f(x) = \{f(x_1), f(x_2), \dots\}$
- Skalarprodukt zwischen Funktionen f_1 und f_2 :
 - Reeller, unbeschränkter Definitionsbereich: $\int_{-\infty}^{\infty} f_1(x)f_2(x)dx$
 - Reeller, beschränkter Definitionsbereich: $\int_{x_{\min}}^{x_{\max}} f_1(x)f_2(x)dx$
 - Ganzzahliger, unbeschränkter Definitionsbereich: $\sum_{n=-\infty}^{\infty} f_1(n)f_2(n)$
 - Ganzzahliger, beschränkter Definitionsbereich: $\sum_{n=0}^{N-1} f_1(n)f_2(n)$

hier relevant
- Orthogonale Basen können wie für Vektoren definiert und genutzt werden.

Für mehrdimensionale Funktionen

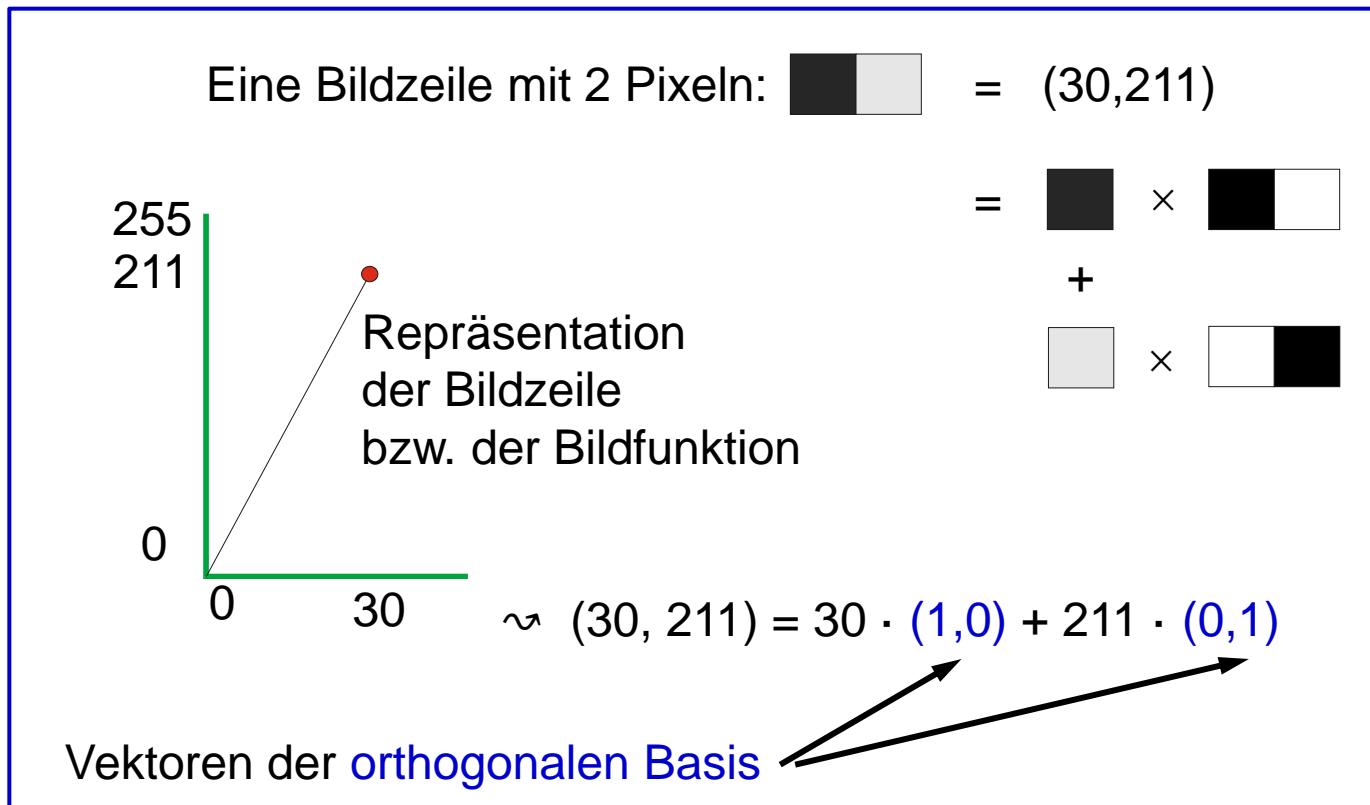
- Konzept der Basisfunktionen ist auf mehrdim. Funktionen erweiterbar.
Nur das Skalarprodukt muss definiert werden.
- Für zwei Funktionen $f_1(m,n)$ und $f_2(m,n)$ mit ganzzahligem, beschränktem
Definitionsbereich $(0,\dots,N-1)(1,\dots,M-1)$ ist das **Skalarprodukt** definiert als:

$$f_1 \bullet f_2 = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f_1(m,n) \cdot f_2(m,n).$$

- Das Konzept lässt sich auf Funktionen mit reellwertigen und unbeschränkten Definitionsbereichen erweitern. Dies ist hier für digitale Bilder nicht relevant.

Funktion als Vektor (Beispiel)

- Ein Bild wird also als Funktion mit *endlichem Definitionsbereich* aufgefasst.
 - Anzahl der Pixel = Anzahl der Funktionswerte = Vektordimension
 - Intensitätsspektrum = Wertebereich für jedes Pixel.



Basis-Transformation und Invertierbarkeit

- Die Transformation T_b in eine andere Basis ist invertierbar,
 - wenn sie **orthogonal** ist
 - wenn die **Anzahl der Basisvektoren gleich** ist.
- Orthogonalität von zwei ganzzahligen Basisfunktionen mit beschränktem
Definitionsbereich : $[b_1 \cdot b_2](n) = \sum_{n=0}^{N-1} b_1(n) \cdot b_2(n) = 0$
- Jede orthogonale Basis kann durch Rotation (und Translation) aus jeder anderen Basis für die gleiche Funktion erzeugt werden.

Transformation entspricht einer

Rotation (und ggf. Translation):

$$T_b[f](u) = \sum_{n=0}^1 f(n) \cdot b_u(n)$$
$$b_1(n) = (\cos \gamma \quad \sin \gamma)$$
$$b_2(n) = (-\sin \gamma \quad \cos \gamma)$$



Basis-Transformation: Vektor-Matrix-Darstellung

- Eine Funktion kann als Vektor \vec{f} ihrer Funktionswerte repräsentiert werden:

$$\vec{f} = (\ f(0) \ f(1) \dots f(N-1))$$

- Basisfunktionen b_u können in quadrat. Matrix **B** zusammengefasst werden:

$$\mathbf{B} = \begin{pmatrix} b_0(0) & b_1(0) & \dots & b_{N-1}(0) \\ b_0(1) & b_1(1) & & \dots \\ \dots & & & \\ b_0(N-1) & \dots & & b_{N-1}(N-1) \end{pmatrix}$$

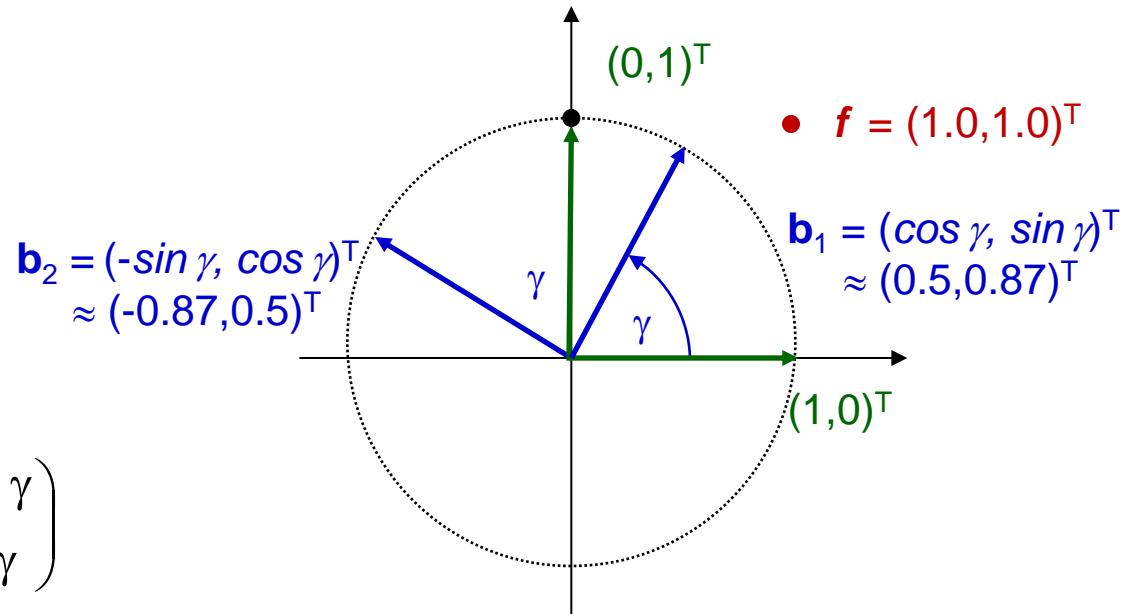
- Transformation auf die neue Basis ist die Multiplikation von \vec{f} mit **B**:

$$\vec{f}' = \vec{f} \times \mathbf{B}$$

Basis-Transformation: Beispiel

Neue Basisvektoren $\mathbf{b}_1, \mathbf{b}_2$ als um Winkel $\gamma = 60^\circ$ gegen den UZS rotierte Version der alten Basis:

$$\mathbf{b}_1 \cdot \mathbf{b}_2 = \begin{pmatrix} 0.5 \\ 0.87 \end{pmatrix} \cdot \begin{pmatrix} -0.87 \\ 0.5 \end{pmatrix} = 0$$

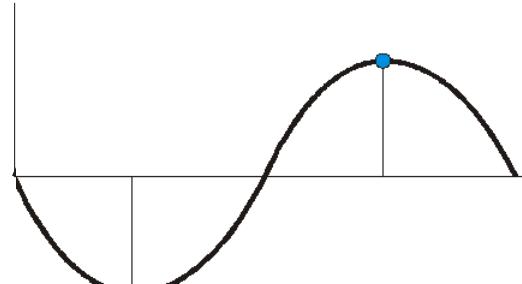


$$\mathbf{B} = (\mathbf{b}_1 \quad \mathbf{b}_2) = \begin{pmatrix} \cos \gamma & -\sin \gamma \\ \sin \gamma & \cos \gamma \end{pmatrix}$$

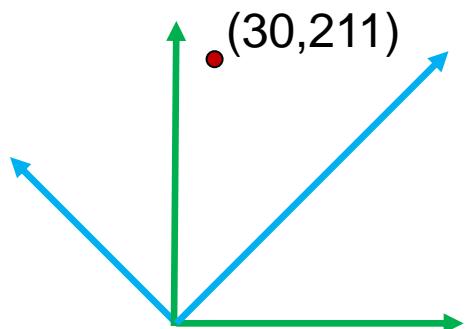
$$\mathbf{f}' = \mathbf{f} \cdot \mathbf{B} = (1 \quad 1) \cdot \begin{pmatrix} 0.5 & -0.87 \\ 0.87 & 0.5 \end{pmatrix} = (1.37 \quad -0.37)$$

Orthogonale periodische Funktionen (1)

Zwei Basisfunktionen, die an zwei *diskreten Orten* definiert sind:



Basis:
 $(1,1)$
 $(-1,1)$



Transformation:

$$(30, 211) \cdot \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} = (241, 181)$$

Rücktransformation:

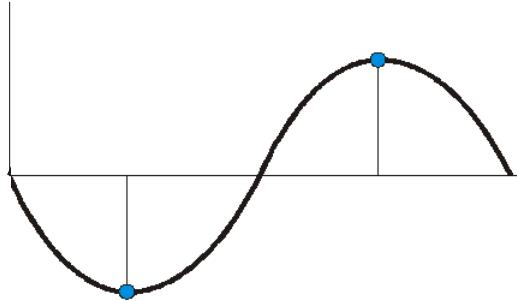
$$(241, 181) \cdot \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} = (60, 422)$$

Anmerkung:

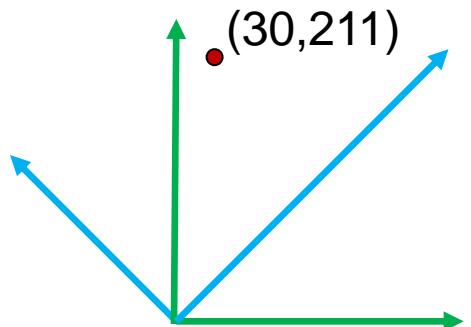
Das Resultat der Rücktransformation muss skaliert werden, weil die Basis nicht *normiert* ist.

Orthogonale periodische Funktionen (2)

Dieselben zwei Basisfunktionen:



Orthonormalbasis:
(0.7071, 0.7071)
(-0.7071, 0.7071)



Transformation:

$$(30, 211) \cdot \begin{pmatrix} 0.707 & -0.707 \\ 0.707 & 0.707 \end{pmatrix} = (170.4, 128.0)$$

Rücktransformation:

$$(170.4, 128.0) \cdot \begin{pmatrix} 0.707 & 0.707 \\ -0.707 & 0.707 \end{pmatrix} = (30, 211)$$

Vollständige Basis für den Frequenzraum (1)

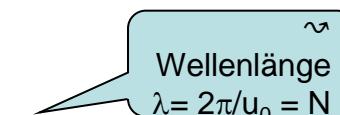
Gegeben seien zwei ungleiche Kosinusfunktionen $\cos(u_1 \cdot n)$ und $\cos(u_2 \cdot n)$:

- diese sind an diskreten Orten $n = 0, \dots, N-1$ definiert (\rightsquigarrow diskrete FT)
- die Frequenzen u_1 und u_2 sind ganzzahl. Vielfache einer Basisfrequenz u_0

Dann gilt:

Ist die Basisfrequenz u_0 gleich $2\pi/N$,

so sind $\cos(u_1 \cdot n)$ und $\cos(u_2 \cdot n)$ orthogonal



Vollständige Basis für den Frequenzraum (2)

Bspl.: $N = 4$, $u_0 = 2\pi/N = \frac{1}{2}\cdot\pi$, $u_1 = 1 \cdot u_0$, $u_2 = 2 \cdot u_0$

$$f_1(n) = \cos(u_1 \cdot n) = \cos(2 \cdot \pi \cdot n/4) = \cos(\frac{1}{2} \cdot \pi \cdot n) \text{ für } n = 0, \dots, 3$$

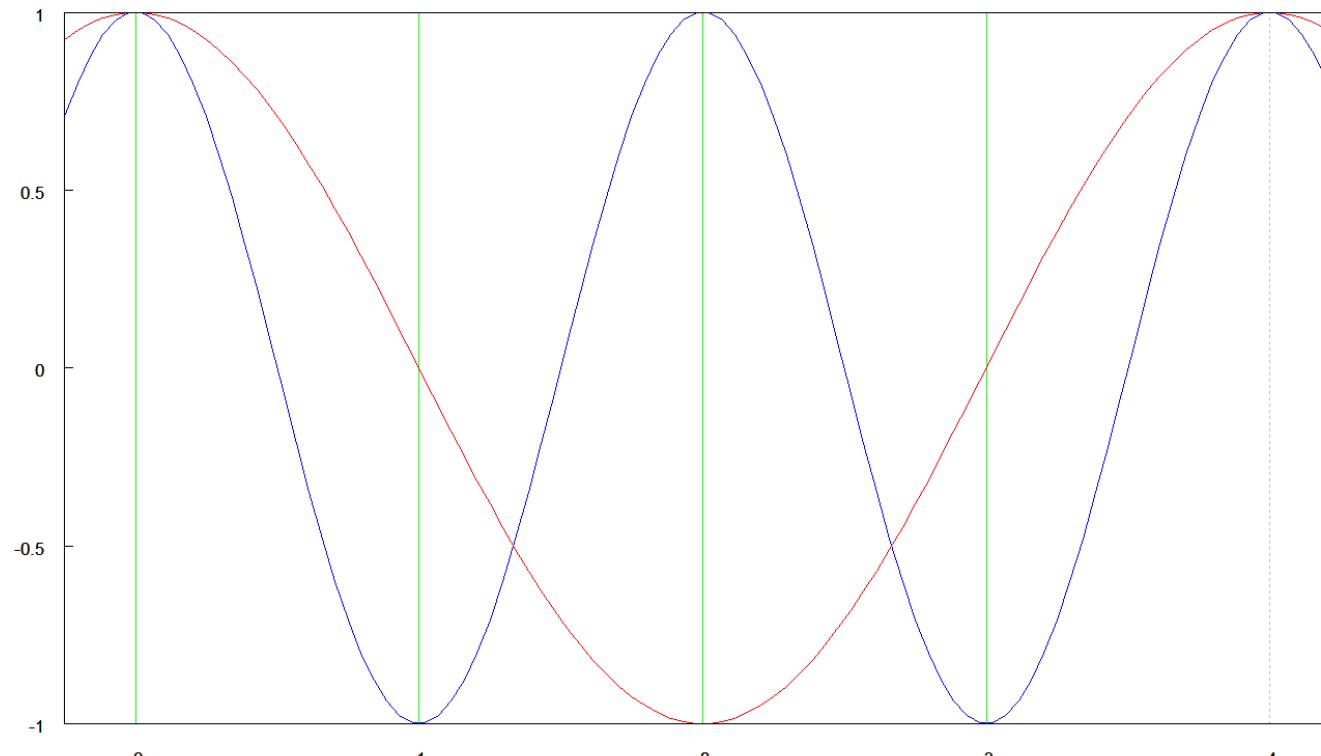
$$f_2(n) = \cos(u_2 \cdot n) = \cos(2 \cdot 2 \cdot \pi \cdot n/4) = \cos(\pi \cdot n) \text{ für } n = 0, \dots, 3$$

$\rightsquigarrow f_1(n)$ und $f_2(n)$ sind orthogonal: $[f_1 \cdot f_2](n) = \sum_{n=0}^{N-1} f_1(n) \cdot f_2(n) = 0$

○

n	$f_1(n)$	$f_2(n)$	$f_1(n) \cdot f_2(n)$
0	1	1	1
1	0	-1	0
2	-1	1	-1
3	0	-1	0
Σ			0

Abszisse : n
Ordinate: $f_k(n)$



Vollständige Basis für den Frequenzraum (3)

Aber für zwei an N diskreten Orten definierte Funktionen $\cos(u_1 \cdot n)$ und $\cos(u_2 \cdot n)$ mit $\frac{u_1}{u_0} = N - \frac{u_2}{u_0}$ gilt: $\cos(u_1 \cdot n) = \cos(u_2 \cdot n)$ für alle $n = 0, \dots, N-1$

Wellen mit Frequenzen $\leq N/2$ sind damit (an den definierten Stellen) nicht unterscheidbar von derart korrespondierenden Wellen mit Frequenzen $> N/2$ für alle $n = 0, \dots, N-1$

Damit stehen *nicht* hinreichend viele Kosinus-Funktionen
für eine Basis zur Verfügung

Vollständige Basis für den Frequenzraum (4)

Bspl.: $N = 4$, $u_0 = 2\pi/N = \frac{1}{2}\cdot\pi$, $u_1 = 3\cdot u_0$, $u_2 = 1\cdot u_0$

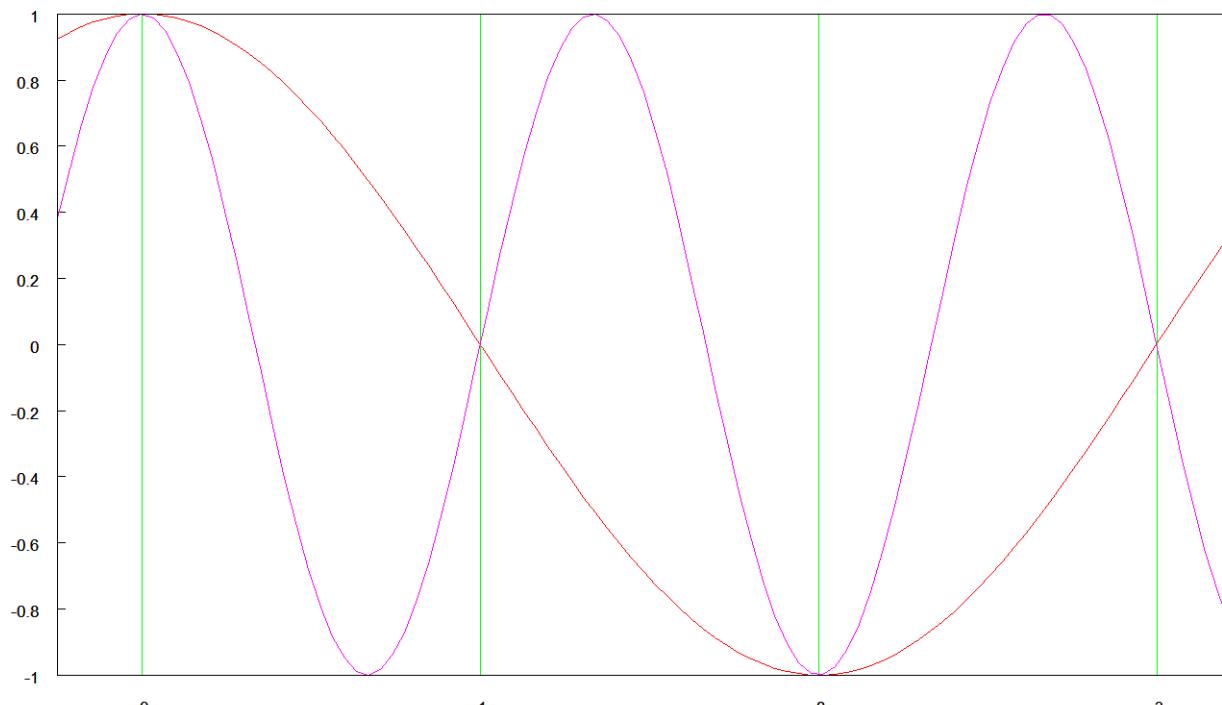
$$u_1/u_0 = 3 = N - u_2/u_0 = 4 - 1$$

$$f_1(n) = \cos(u_1 \cdot n) = \cos(3 \cdot 2 \cdot \pi \cdot n/4) = \cos(1\frac{1}{2} \cdot \pi \cdot n) \text{ für } n = 0, \dots, 3$$

$$f_2(n) = \cos(u_2 \cdot n) = \cos(1 \cdot 2 \cdot \pi \cdot n/4) = \cos(\frac{1}{2} \cdot \pi \cdot n) \text{ für } n = 0, \dots, 3$$

$\leadsto f_1(n)$ und $f_2(n)$ sind nicht unterscheidbar für $n = 0, \dots, 3$

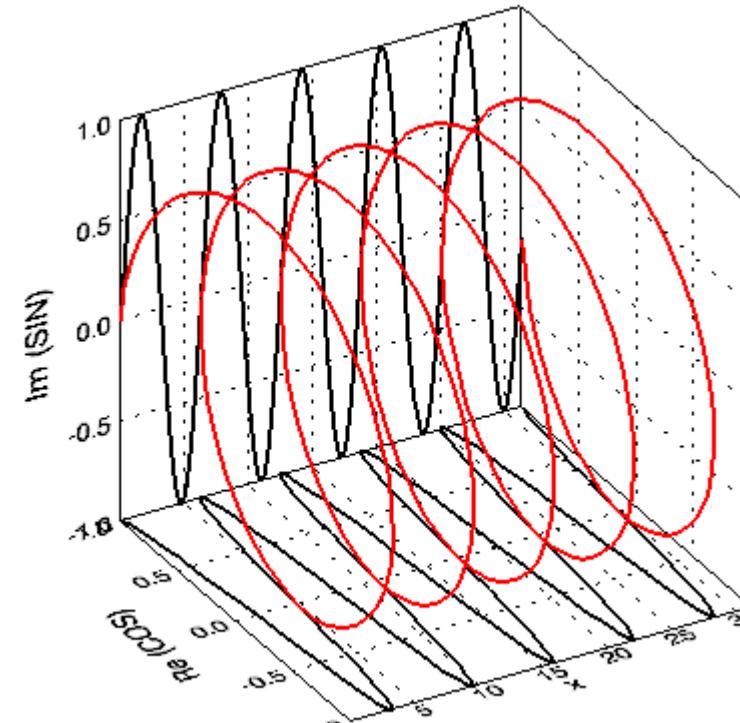
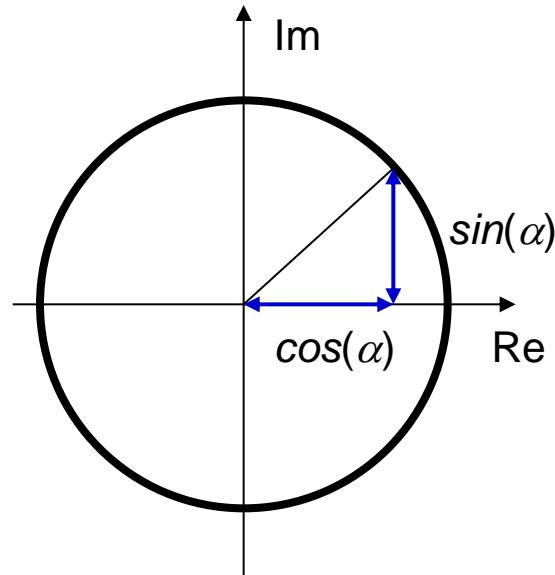
n	$f_1(n)$	$f_2(n)$
0	1	1
1	0	0
2	-1	-1
3	0	0



$$f_1(n) = \cos(u_2 \cdot n) = \cos(3 \cdot 2 \cdot \pi \cdot n/4) = \cos(1\frac{1}{2} \cdot \pi \cdot n) \text{ und } f_2(n) = \cos(u_1 \cdot n) = \cos(2 \cdot 2 \cdot \pi \cdot n/4) = \cos(\frac{1}{2} \cdot \pi \cdot n)$$

Vollständige Basis für den Frequenzraum (5)

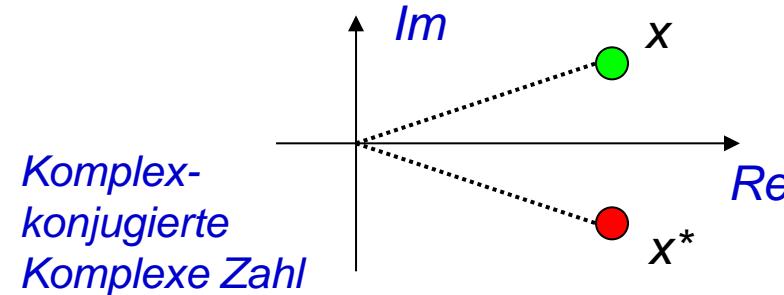
- Für eine vollständige orthogonale Basis sind also weitere Basisfunktionen zu finden, die möglichst der Semantik der Kosinuswellen folgen
- Anstelle reellwertiger Funktionen $\cos(u \cdot n)$ werden komplexe Funktionen $\cos(u \cdot n) + i \cdot \sin(u \cdot n)$ gewählt



Alle Werte für komplexe Zahlen der Form $\cos(\alpha) + i \sin(\alpha)$ liegen auf einem Kreis mit Abstand 1 in der komplexen Ebene.

Vollständige Basis für den Frequenzraum (6)

- Basisfunktionen sind also $\vec{b}_u(n) = [b_{u,\cos} \ b_{u,\sin}] = [\cos(n \cdot u \cdot 2\pi/N) \ \sin(n \cdot u \cdot 2\pi/N)]$
- Das Skalarprodukt zwischen zwei Vektoren mit komplexen Elementen:
 - Summe über den Produkten der Komponenten des ersten Vektors mit den komplex-konjugierten Komponenten des zweiten Vektors.
 - Die komplex-konjugierte komplexe Zahl zu $x = a + i \cdot b$ ist $x^* = a - i \cdot b$.
 - Also:



$$\vec{x} \bullet \vec{y} = \sum_{i=0}^{N-1} x_i \cdot y_i^* = \sum_{i=0}^{N-1} (\operatorname{Re}(x_i) + i \operatorname{Im}(x_i))(\operatorname{Re}(y_i) - i \operatorname{Im}(y_i))$$

Vollständige Basis für den Frequenzraum (7)

- Analog zu den reellwertigen Kosinusfunktionen lässt sich zeigen:
zwei komplexe Wellen $\cos(u_1 \cdot n) + i \sin(u_1 \cdot n)$ und $\cos(u_2 \cdot n) + i \sin(u_2 \cdot n)$
sind für ganzzahlige Vielfache $u_1 \neq u_2$ einer Basisfrequenz $u_0 = 2\pi/N$ im
Definitionsbereich $n = 0, \dots, N-1$ zueinander orthogonal
 - Zusätzlich sind jetzt durch die zusätzlichen Sinuswellen bis zu N unter-
schiedliche Wellenlängen bei N diskreten Definitionstellen auf dem
Intervall unterschiedlich repräsentierbar
- Damit spannen die N Funktionen den gewünschten N -dim. Raum auf

Eindimensionale Fourier-Transformation (1)

- Basisfunktionen sind also $\vec{b}_u(n) = [b_{u,\cos} \ b_{u,\sin}] = [\cos(n \cdot u \cdot 2\pi/N) \ \sin(n \cdot u \cdot 2\pi/N)]$
- Bei der Fourier-Transformation wird eine Funktion $f(n)$ durch Konvolution mit den Basisfunktionen auf den N-dim. Frequenzraum projiziert gemäß:

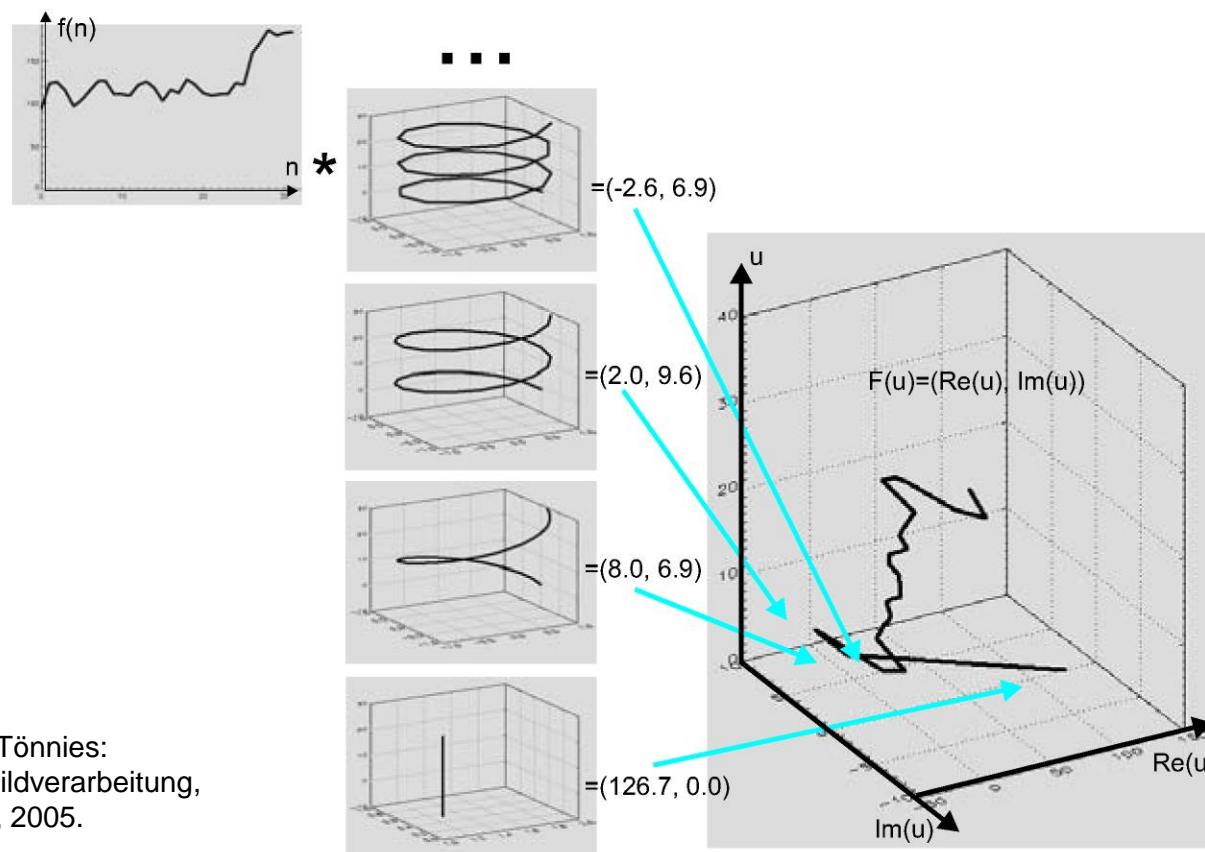
$$F(u) = \frac{1}{s} \sum_{n=0}^{N-1} f(n) \left[\cos\left(\frac{2\pi u n}{N}\right) - i \cdot \sin\left(\frac{2\pi u n}{N}\right) \right].$$

- Durch die Skalierung s werden die Basisfunktionen normiert. Der Betrag der Basisfunktionen ist dabei:

$$s = \left\| \vec{b}_u \right\| = \sqrt{\sum_{n=0}^{N-1} \cos^2\left(\frac{2\pi u n}{N}\right) + \sin^2\left(\frac{2\pi u n}{N}\right)} = \sqrt{N}.$$

Eindimensionale Fourier-Transformation (2)

Bspl.: Projektion einer 1-dim. Funktion auf eine Folge von orthogonalen, komplexen, periodischen Basisfunktionen. Das Ergebnis jeder einzelnen Projektion ist eine komplexe Zahl. Die Folge aller komplexen Zahlen ergibt die transformierte Funktion.



Eindimensionale FT mit komplexen Basisfunktionen (1)

- Taylorreihenentwicklung für Kosinus und Sinus:

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots \quad \sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots$$

- Taylorreihenentwicklung für e^{ix} :

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots \Rightarrow e^{ix} = 1 + \frac{ix}{1!} + \frac{(ix)^2}{2!} + \frac{(ix)^3}{3!} + \dots$$

- Wegen $i^2 = -1$ gilt daher die Eulersche Formel:

$$\cos(x) + i \cdot \sin(x) = e^{ix}$$

Eindimensionale FT mit komplexen Basisfunktionen (2)

Formuliert über Exponentialfunktion:

- Bildfunktion: $f(n), n = 0, \dots, N-1$,

also: N Basisfunktionen

$$b_u(n) = e^{-i \cdot n \cdot u \cdot 2\pi/N} \text{ mit Frequenzfaktoren } u = 0, \dots, N-1,$$

z.B. $b_0(n) = [(1,1), (1,1), \dots, (1,1)]$

- Transformation FT: $\mathbf{FT}(f) = \mathbf{F} = f \cdot \mathbf{B}$ (Vektor-Matrix-Schreibweise)

$$F(u) = \frac{1}{\sqrt{N}} \cdot \sum_{n=0}^{N-1} f(n) \cdot \exp(-i \cdot n \cdot u \cdot \frac{2\pi}{N}), u = 0, \dots, N-1$$

- Rücktransformation \mathbf{FT}^{-1} : $\mathbf{FT}(f) = \mathbf{F} = f \cdot \mathbf{B}$ (Vektor-Matrix-Schreibweise)

$$f(n) = \frac{1}{\sqrt{N}} \cdot \sum_{u=0}^{N-1} F(u) \cdot \exp(i \cdot n \cdot u \cdot \frac{2\pi}{N}), n = 0, \dots, N-1$$



Skalierungsfaktoren, weil die Basisfunktionen nicht normiert sind

Zweidimensionale FT mit komplexen Basisfunktionen

Basisfunktionen sind **Wellen**:

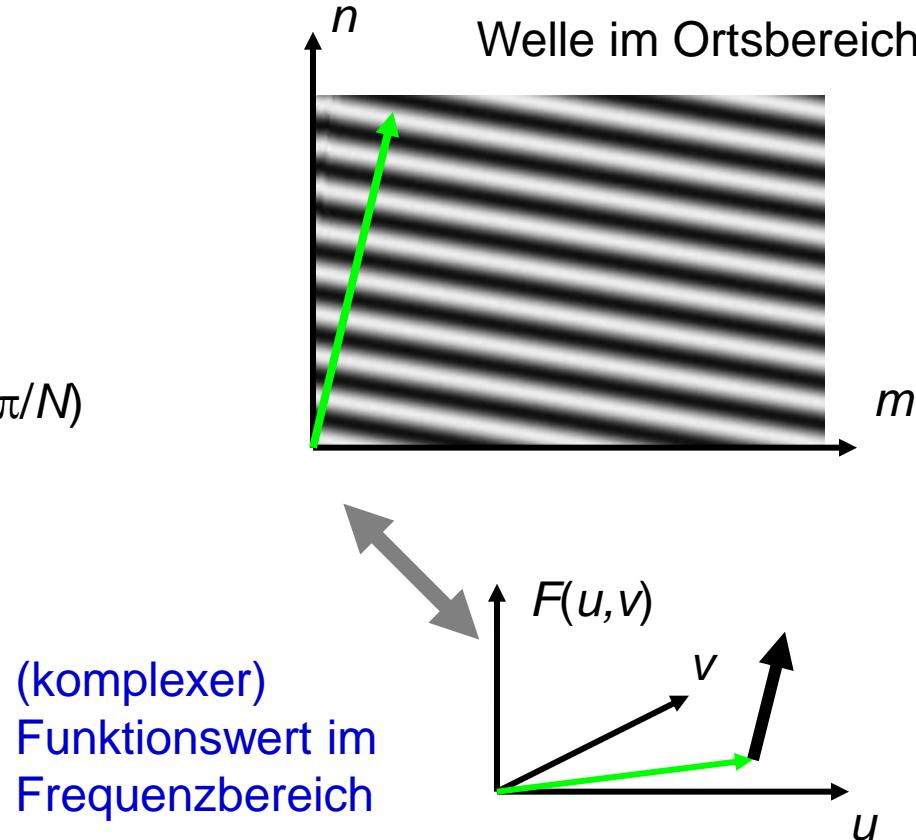
$$\exp((mu+nv) \cdot i \cdot 2\pi/N)$$

Richtung ist durch Vektor $(u v)$ gegeben.

Die Basisfunktionen der 2-D Fourier-Transformation sind **zerlegbar**:

$$\begin{aligned}\exp((m \cdot u + n \cdot v) \cdot i \cdot 2\pi/N) &= \\ \exp(m \cdot u \cdot i \cdot 2\pi/N) \cdot \exp(n \cdot v \cdot i \cdot 2\pi/N)\end{aligned}$$

Bildquelle: Klaus Tönnies:
Grundlagen der Bildverarbeitung,
Pearson Studium, 2005.



2D Fourier-Transformationspaar

$$F(u, v) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) \cdot e^{-i \cdot 2\pi \left(\frac{um}{M} + \frac{vn}{N} \right)}$$

$$f(m, n) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) \cdot e^{i \cdot 2\pi \left(\frac{um}{M} + \frac{vn}{N} \right)}$$

Transformationspaar für
Bilder der Größe $M \times N$

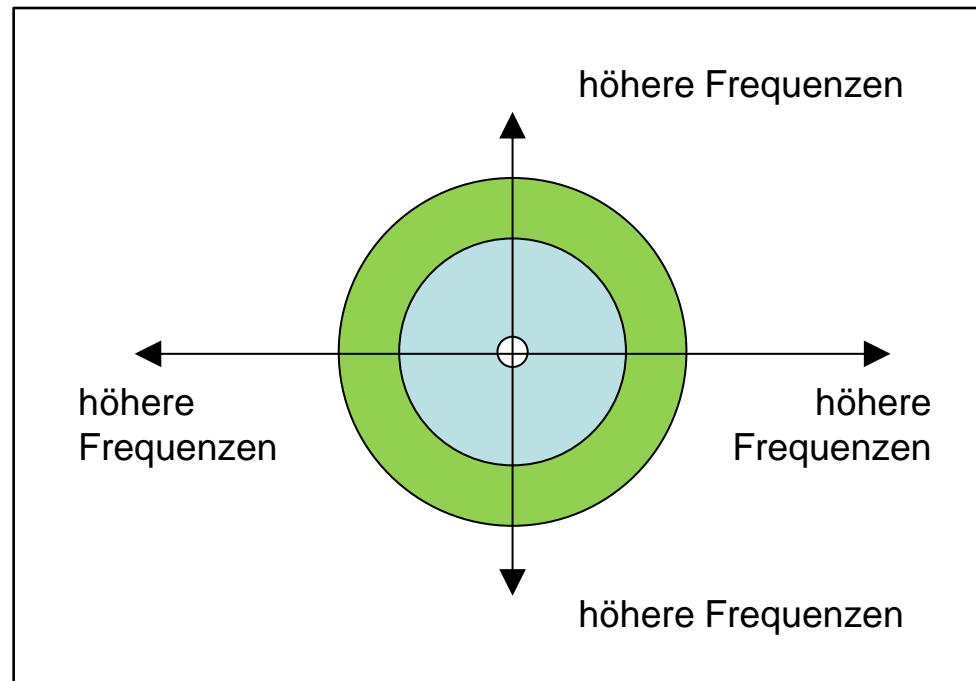
Transformationspaar für quadrat.
Bilder der Größe $N \times N$

$$F(u, v) = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m, n) \cdot e^{-i \cdot \frac{2\pi}{N} \cdot (um + vn)}$$

$$f(m, n) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) \cdot e^{i \cdot \frac{2\pi}{N} \cdot (um + vn)}$$

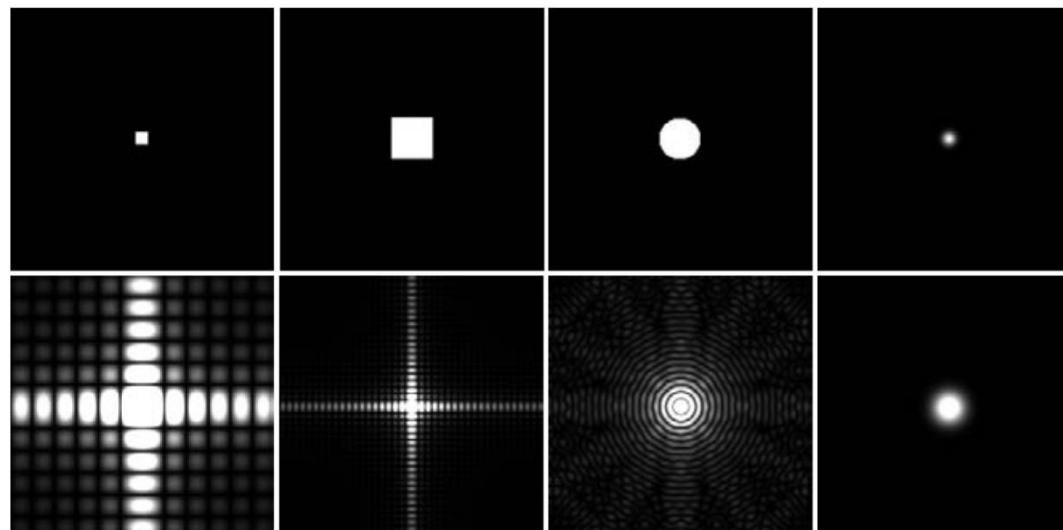
Bilder im Orts- und Frequenzraum (1)

- Das Ergebnis der FT ist komplex, daher wird zur Visualisierung des Amplitudenspektrums der Betrag genutzt: $|F(u, v)| = \sqrt{\left(Re(F(u, v))\right)^2 + \left(Im(F(u, v))\right)^2}$
- Bei der Darstellung des Amplitudenspektrums werden die Amplituden der tiefen Frequenzen in den Bildmittelpunkt verschoben
- Das Resultat ist eine im Bildmittelpunkt zentrierte Darstellung, bei dem der Ursprung in der Mitte und die hohen Frequenzen außen sind



Bilder im Orts- und Frequenzraum (2)

- Horizontale Strukturen im Originalbild resultieren in vertikalen Komponenten des Fourier-Spektrums. Vertikale Anordnungen im Originalbild resultieren in horizontalen Komponenten (z.B. Richtungspräferenz nach Ausführung der FT auf den beiden Bildern mit Quadraten).
- Die Repräsentation von scharfen Kanten erfordert sehr viele hohe Frequenzanteile. Nur im Bild rechts außen, das keine harte Kante zwischen Vordergrund und Hintergrund besitzt, sinken die Werte im Frequenzraum rasch mit steigender Frequenz.



Bildquelle: Klaus Tönnies:
Grundlagen der Bildverarbeitung, Pearson Studium,
2005.

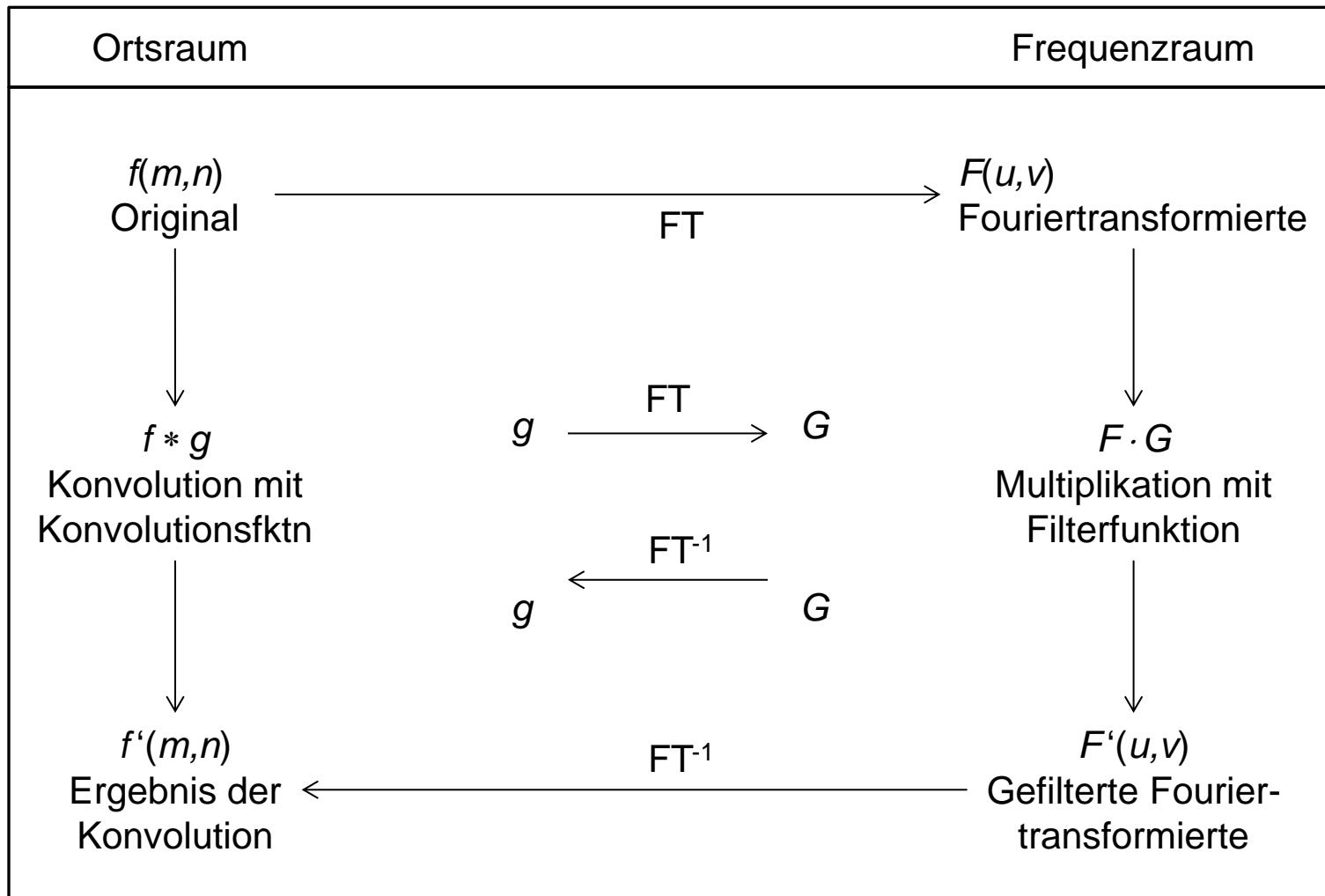
Beträge der Fourier-Transformierten (unten) für verschiedene einfache Bilder (oben). ☺

Schnelle FT = Fast FT = FFT

- Geg.: Bild der Größe $N \times N$:
 - Berechnungsaufwand der FT für ein Pixel: $O(N^2)$
 - Berechnungsaufwand der FT für gesamtes Bild: $O(N^4)$
- Die FFT basiert auf einer Divide-and-Conquer-Strategie unter Nutzung der Periodizität ($F(u) = F(u + N)$, $f(n) = F(n + N)$), Symmetrie ($F(u) = {}^*F(-u)$, $f(u, v) = {}^*F(-u, -v)$) und Separabilität (2D-FT und 2D-FT⁻¹ als Produkt von jeweils zwei 1D-FTs)
 - Berechnungsaufwand der FFT für ein Pixel: $O(\log N)$
 - Berechnungsaufwand der FFT für gesamtes Bild: $O(N^2 \log N)$

Konvolution und FT (1)

Die Fourier-Transformierte des Faltungsprodukts $f'(m,n) = f(m,n) * g(m,n)$ ist das Produkt der Fourier-Transformierten der einzelnen Funktionen (s. Anh.):



Konvolution und FT (2)

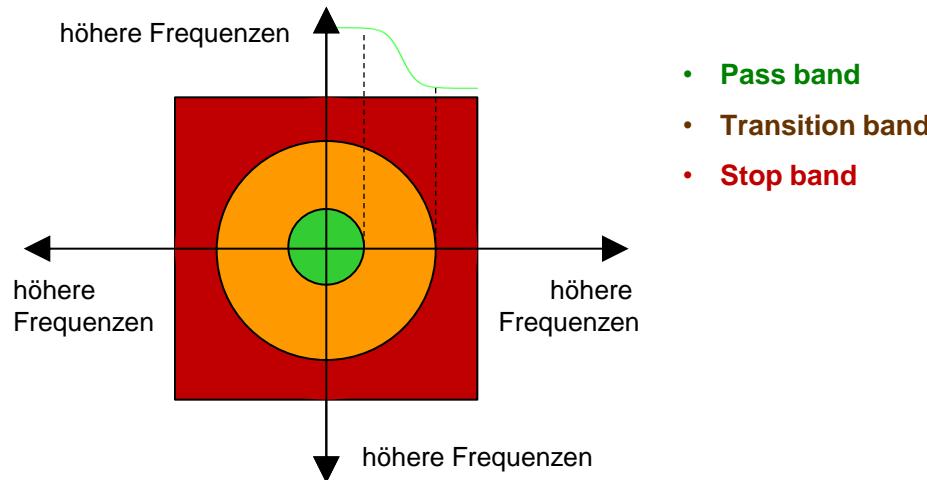
- Damit ist ein wichtiges Einsatzgebiet der FT angesprochen: Eine rechenintensive Faltung (Integration) im Ortsraum kann durch eine einfache Multiplikation im Frequenzraum durchgeführt werden
- Die Konvolution des Bildes der Größe $N \times N$ mit einem Faltungskern der Größe m (also $m \times m$ Gewichte) hat die Zeitkomplexität $O(m^2 \cdot N^2)$
- Über FFT ausgeführt braucht die Konvolution des Bildes $O(N^2 \cdot \log_2 N)$
- Für $\log_2 N < m^2$ ist die Konvolution also über FFT effizienter ausführbar
- Beispiele:
 - 1024×1024 -Bild: $\log_2 1024 = 10$, 512×512 -Bild: $\log_2 512 = 9$, 256×256 -Bild: $\log_2 256 = 8$
 - 3×3 -Filter: $3^2 = 9$, 5×5 -Filter: $5^2 = 25$, 7×7 -Filter: $7^2 = 49$, ...

Filterung im Frequenzbereich

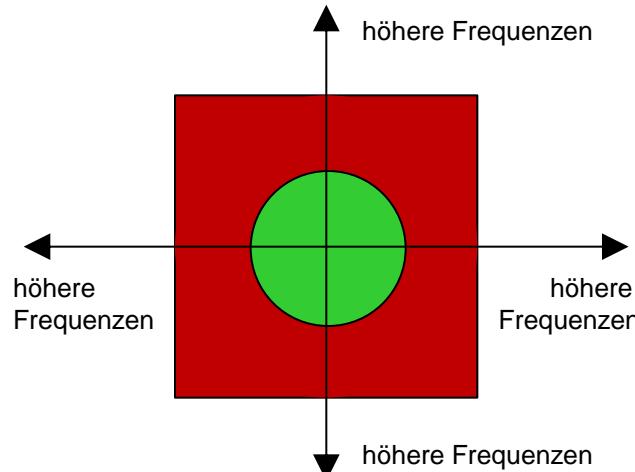
- **Tiefpass**: lässt tiefe Frequenzen passieren, reduziert oder eliminiert hohe Frequenzen
- **Ideales Tiefpassfilter**: Alle Frequenzen kleiner/gleich dem *Cut Off* werden ohne Abschwächungen übernommen, alle anderen vollkommen unterdrückt
- **Hochpass**: lässt hohe Frequenzen passieren, reduziert oder eliminiert tiefe Frequenzen
- **Ideales Hochpassfilter**: Alle Frequenzen kleiner/gleich dem *Cut Off* werden ohne Abschwächungen übernommen, alle anderen vollkommen unterdrückt
- **Bandpass**: Kombination von Tiefpass und Hochpass

Tiefpass im Frequenzbereich

- Tiefpass:

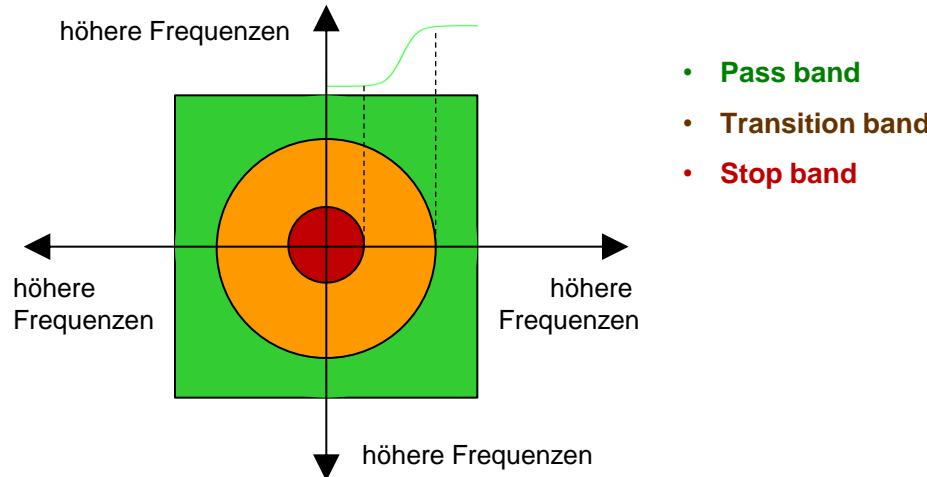


- Idealer Tiefpassfilter:

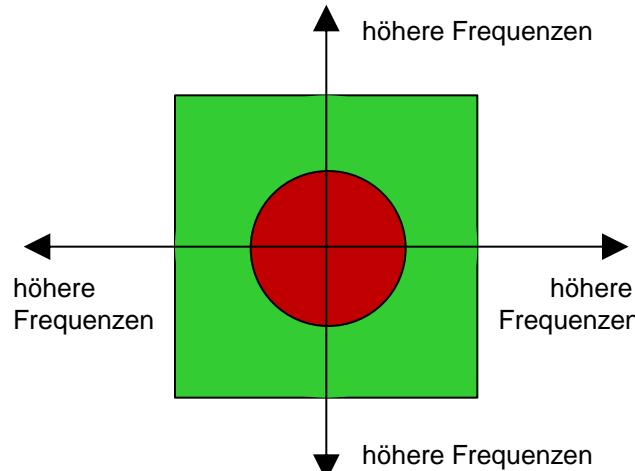


Hochpass im Frequenzbereich

- Hochpass:

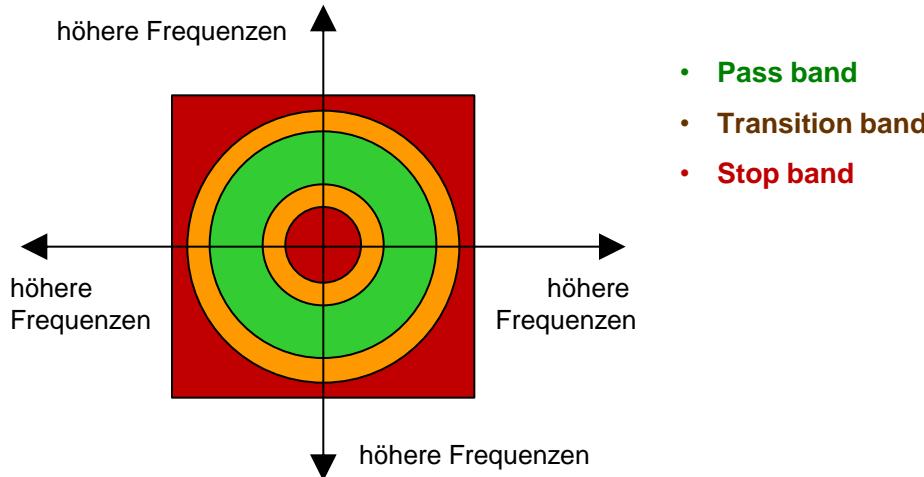


- Idealer Hochpassfilter:

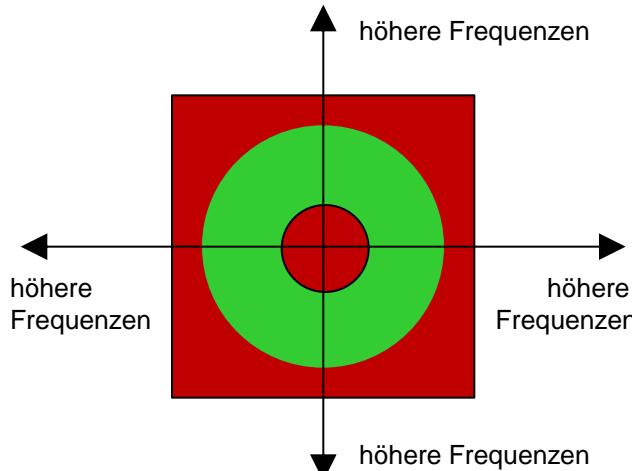


Bandpass im Frequenzbereich

- Bandpass:

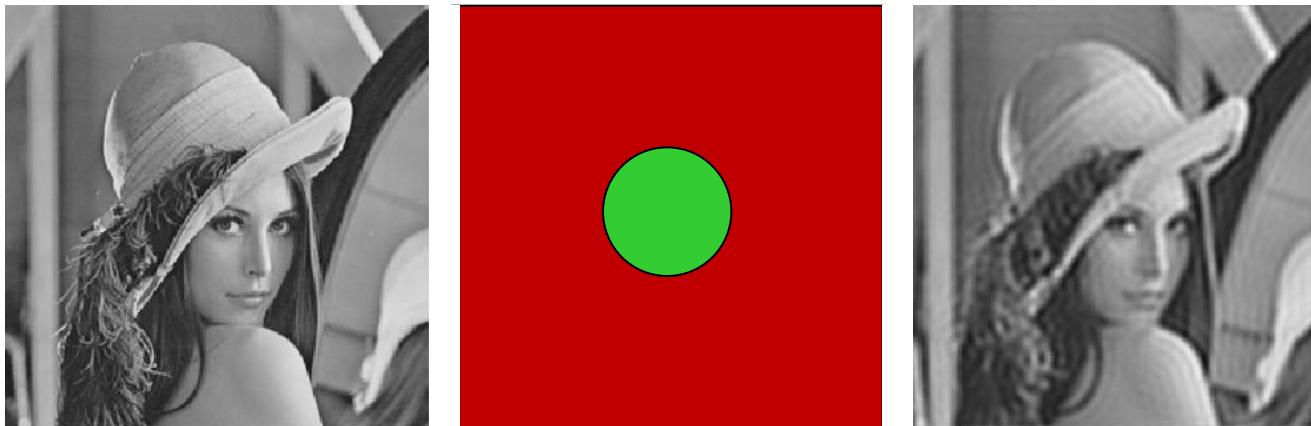


- Idealer Bandpassfilter:



Anwendung von Tiefpass

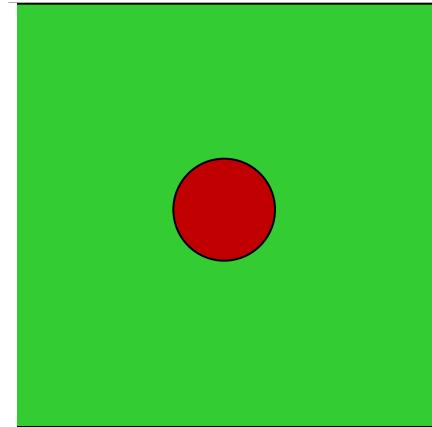
- Idealer Tiefpassfilter



- Tiefpassfilter sind glättend (entsprechen Glättungsfilters im Ortsraum)
~ Verschmieren des Bildes und Abschwächen der Kanteninformation

Anwendung von Hochpass

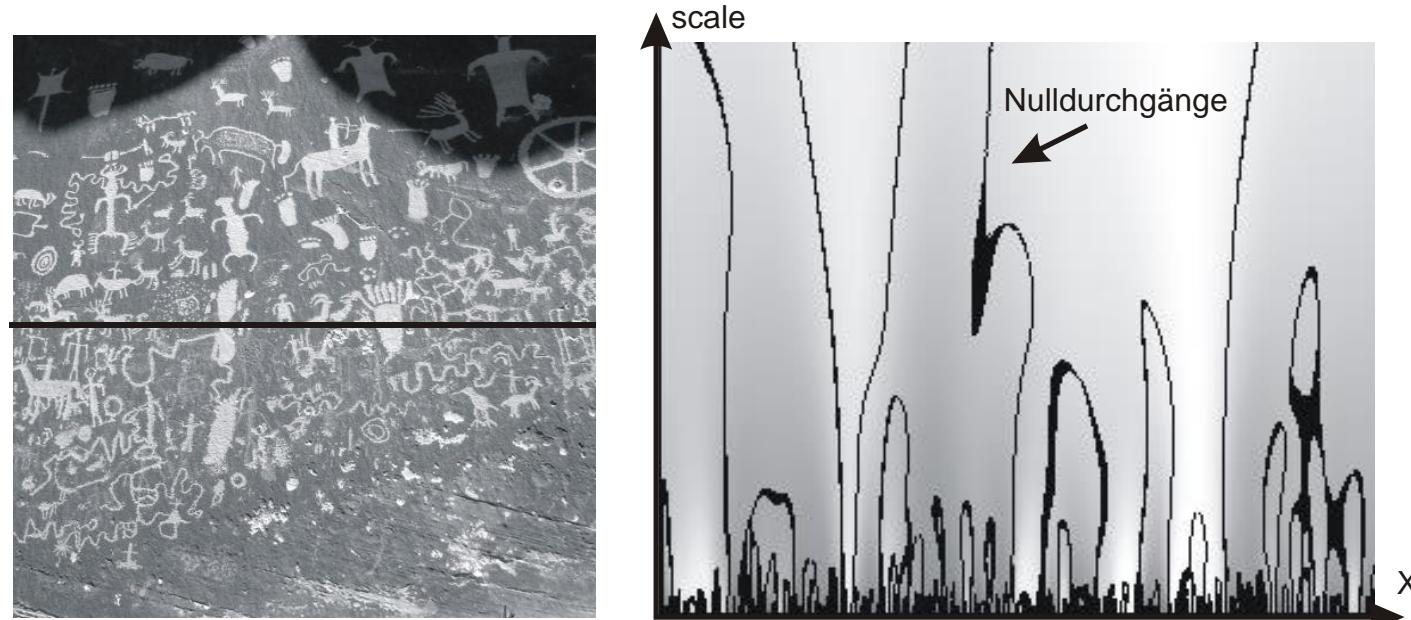
- Idealer Hochpassfilter



- Hochpassfilter heben Konturen hervor (entspr. Kantenfiltern im Ortsraum)
~ Hervorhebung der Kanteninformation

Multiskalenstrategien (1)

- Relative Kriterien für Homogenität (und damit auch Diskontinuitäten) können über unterschiedliche Entferungen (Skalen) verschieden wirken.
- Segmentierung nach Multiskalenstrategie wertet Kriterien auf unterschiedlichen Skalierungen aus.



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Multiskalenstrategien (2)

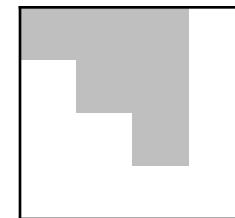
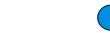
Multiskalenstrategie sind umsetzbar:

- auf einer expliziten Multiskalenrepräsentation
- oder*
- implizit durch Integration in den Segmentierungsalgorithmus

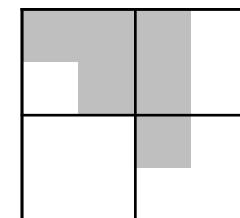
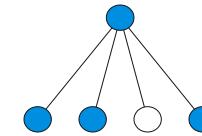
Implizit integrierte Multiskalenstrategien

Split-and-Merge (s. Vorl. 5) ist ein Beispiel für die implizite Integration einer Multiskalenstrategie in den Segmentierungsalgorithmus:

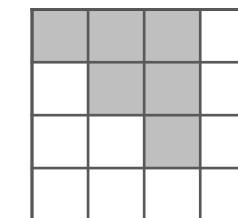
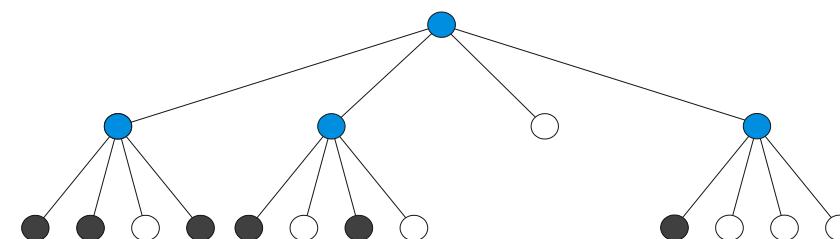
1. Skala:



2. Skala:



3. Skala:



Explizite Multiskalenrepräsentation (1)

- Expliziten Multiskalenrepräsentationen bilden das Originalbild in einen **Multiskalenraum** ab
- Die Idee des Multiskalenraums basiert auf der Erkenntnis, dass die menschliche Wahrnehmung Information in Detailebenen verwertet
- Anschaulich kann die Wahrnehmung von Detailebenen über unterschiedliche Entfernungen (Skalen) imaginiert werden

Die Seitenansicht desselben Autos wird bei unterschiedlichen Beobachtungsdistanzen unterschiedliche Detailebenen zeigen:

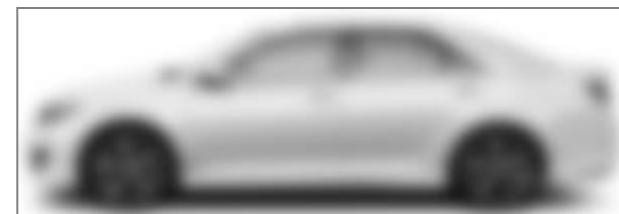


Seitenansicht eines PKWs in 307×108 bzw. 76×27 Pixeln (Bildquelle: Toyota Motor Corp.)

Explizite Multiskalenrepräsentation (2)

- Aber Ebenen unterschiedlicher Detailgenauigkeit sind oft über Modellwissen definiert. Die Wissen steht bei der Segmentierung i.A. nicht zur Verfügung.
- Ein Multiskalenraum ist ohne Modellwissen über spezifische Frequenzbänder bzw. spezifische Ebenen unterschiedlicher Auflösung definierbar.

Dazu können Glättungsfilter (im Ortsraum) bzw. Tiefpassfilter (im Frequenzraum) systematisch genutzt werden.

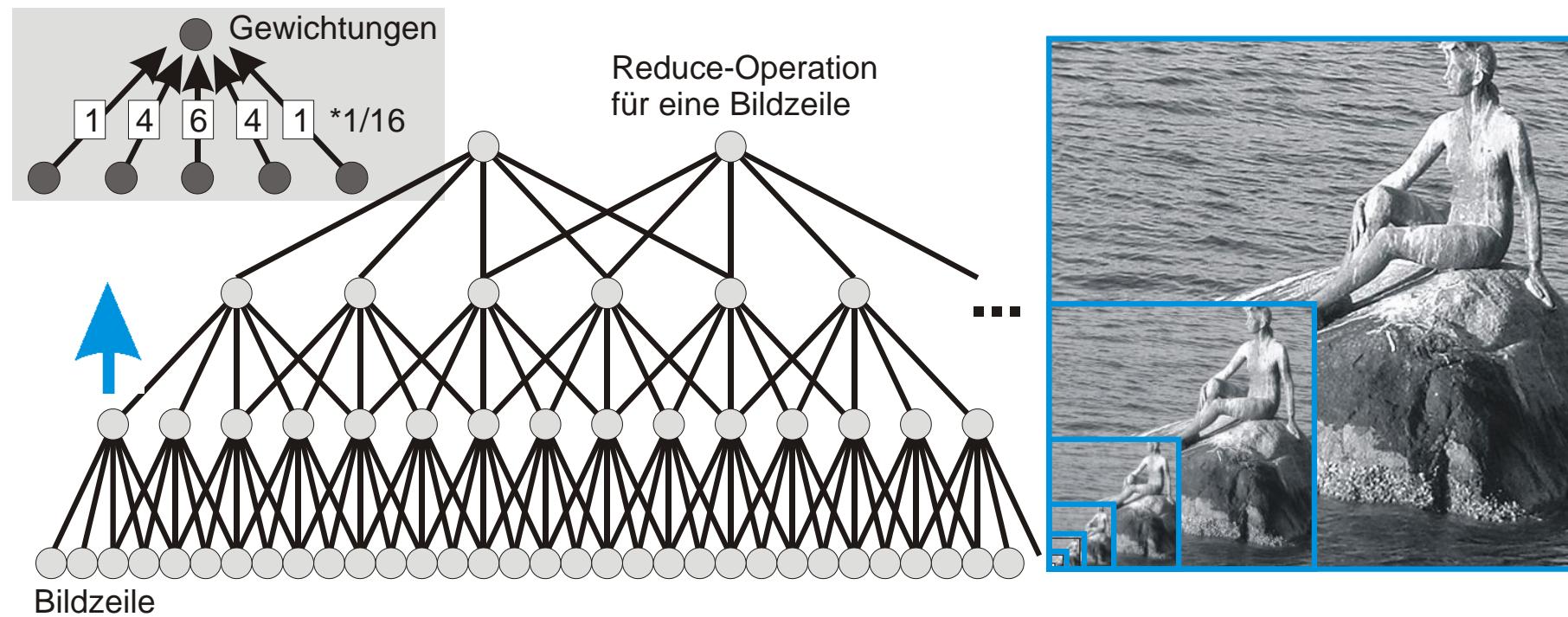


Seitenansicht eines PKWs in 307×108 bzw. 76×27 Pixeln sowie Ergebnis einer Tiefpassfilterung auf dem Originalbild (Bildquelle: Toyota Motor Corp.)

Gauß-Pyramide (1)

Eine Gauß-Pyramide ist eine Möglichkeit zur Abbildung eines Bildes in einen Multiskalenraum:

- Das Originalbild wird fortlaufend durch eine „reduce“-Operation in seiner Anzahl von Pixeln in jeder Spalte und jeder Zeile halbiert
- Jedes Pixel der nächsten Skalierungsstufe repräsentiert 4 Pixel der aktuellen Stufe

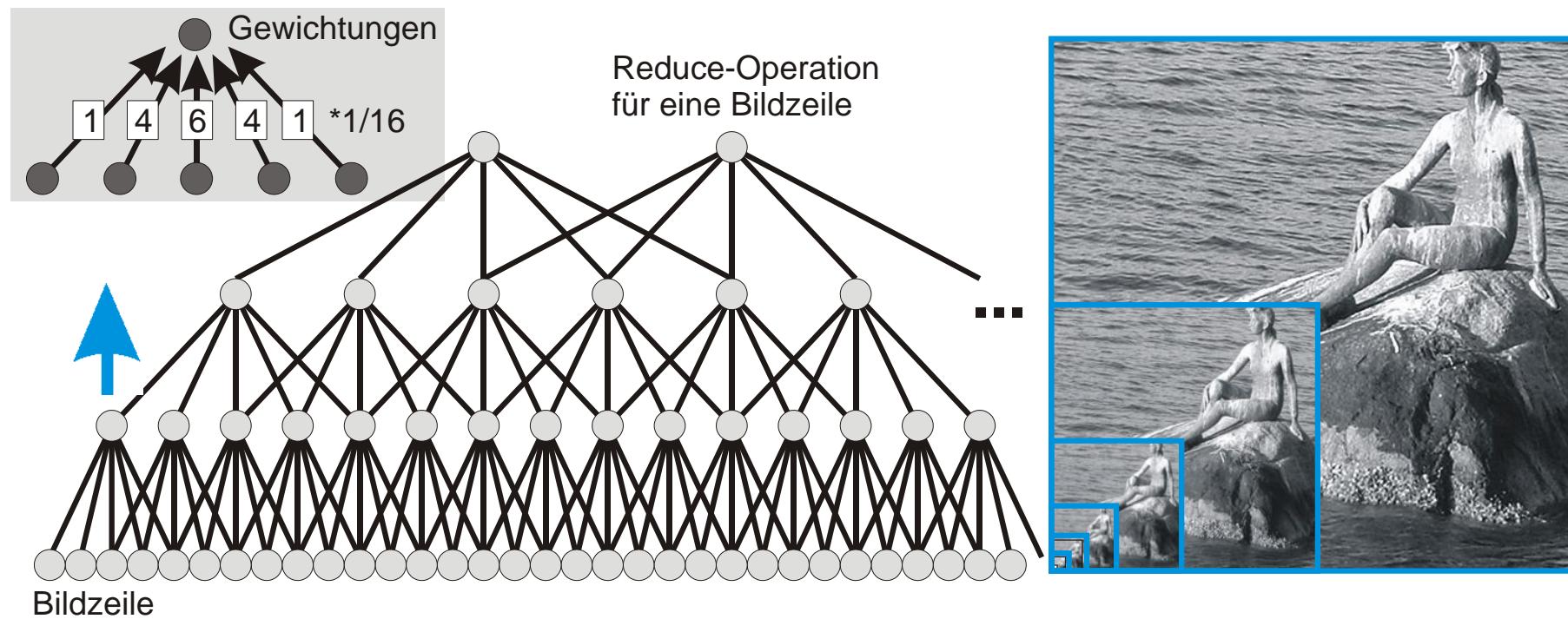


Gauß-Pyramide (2)

Vor der Reduktion wird der Frequenzumfang durch Filterung vermindert. Die Gauß-Filterung mit $\sigma = 1$ (bzw. deren Approx.) ist in der Praxis bewährt:

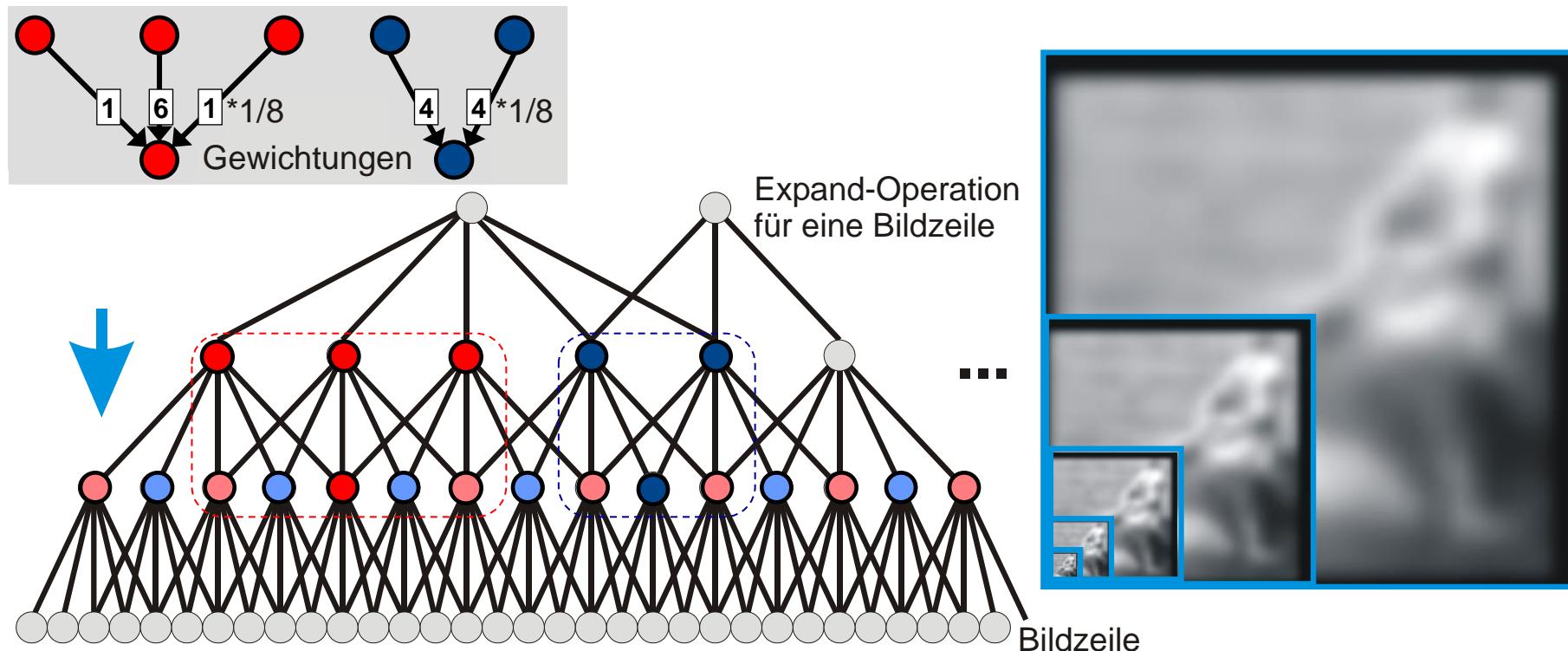
Gaußfilter: $\frac{1}{16}(0.87 \quad 3.91 \quad 6.44 \quad 3.91 \quad 0.87)$

Binomialfilter: $\frac{1}{16}(1 \quad 4 \quad 6 \quad 4 \quad 1)$



Gauß-Pyramide (3)

- Für Vergleiche verschiedener Skalierungsstufen müssen reduzierte Stufen wieder auf vorherige Stufen abbildbar sein
- Um die vorherige Skalierungsstufe aus der aktuellen Stufe zu erzeugen, wird daher eine „expand“-Operation definiert



Gauß-Pyramide (4)

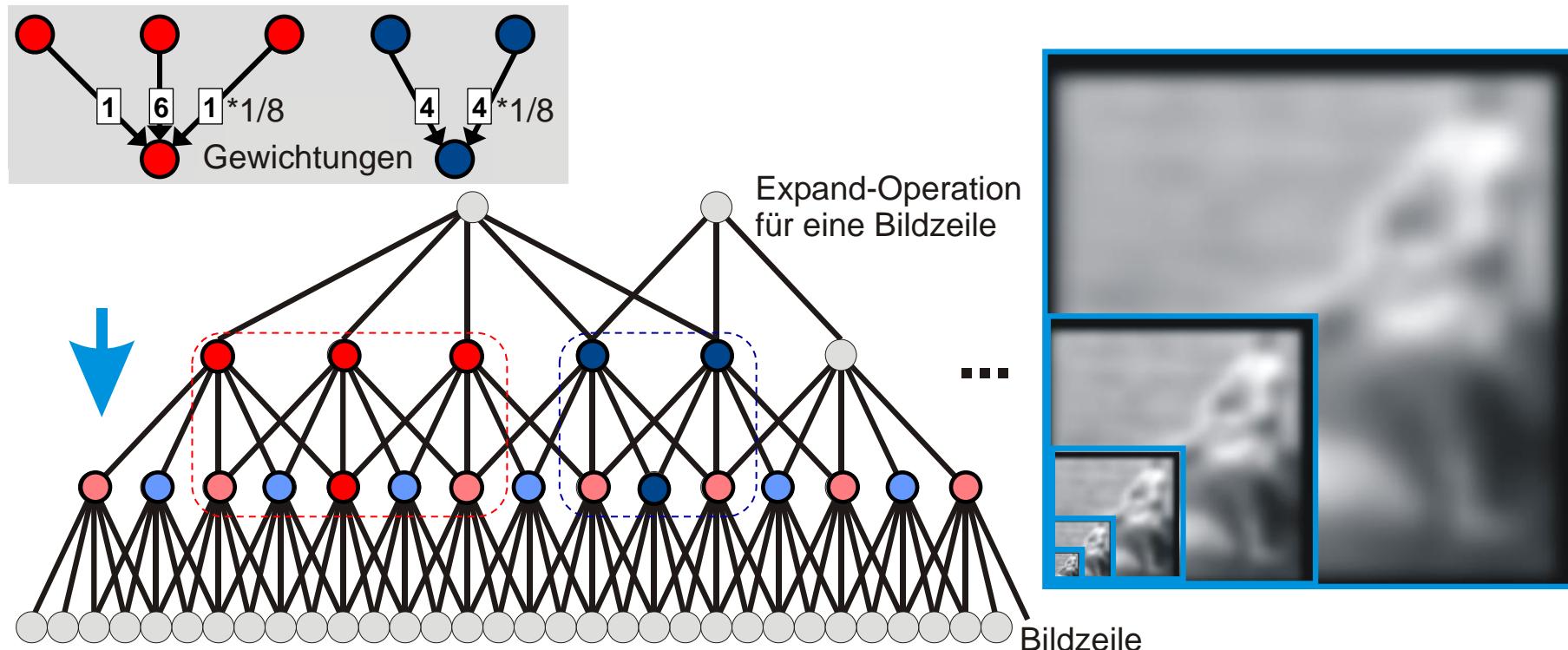
- Pixel der neuen Skalierungsstufe werden durch Interpolation erzeugt:

- Pixelorte, die auf beiden Skalierungsstufen existieren:

$$\frac{1}{8.18} (0.87 \quad 6.44 \quad 0.87) \text{ bzw. } \frac{1}{8} (1 \quad 6 \quad 1)$$

- Pixelorte, die nur auf der vorherigen Skalierungsstufe existieren:

$$\frac{1}{7.82} (3.91 \quad 3.91) \text{ bzw. } \frac{1}{8} (4 \quad 4)$$



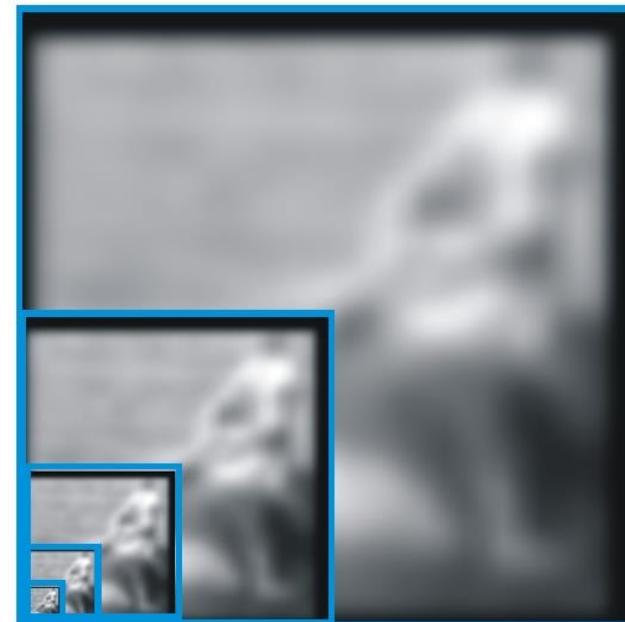
Gauß-Pyramide (5)

Der Aufbau der Gauß-Pyramide kann also wie folgt formal beschrieben werden:

$$\mathbf{G}_0 = \text{Originalbild } \mathbf{I}[x,y]$$

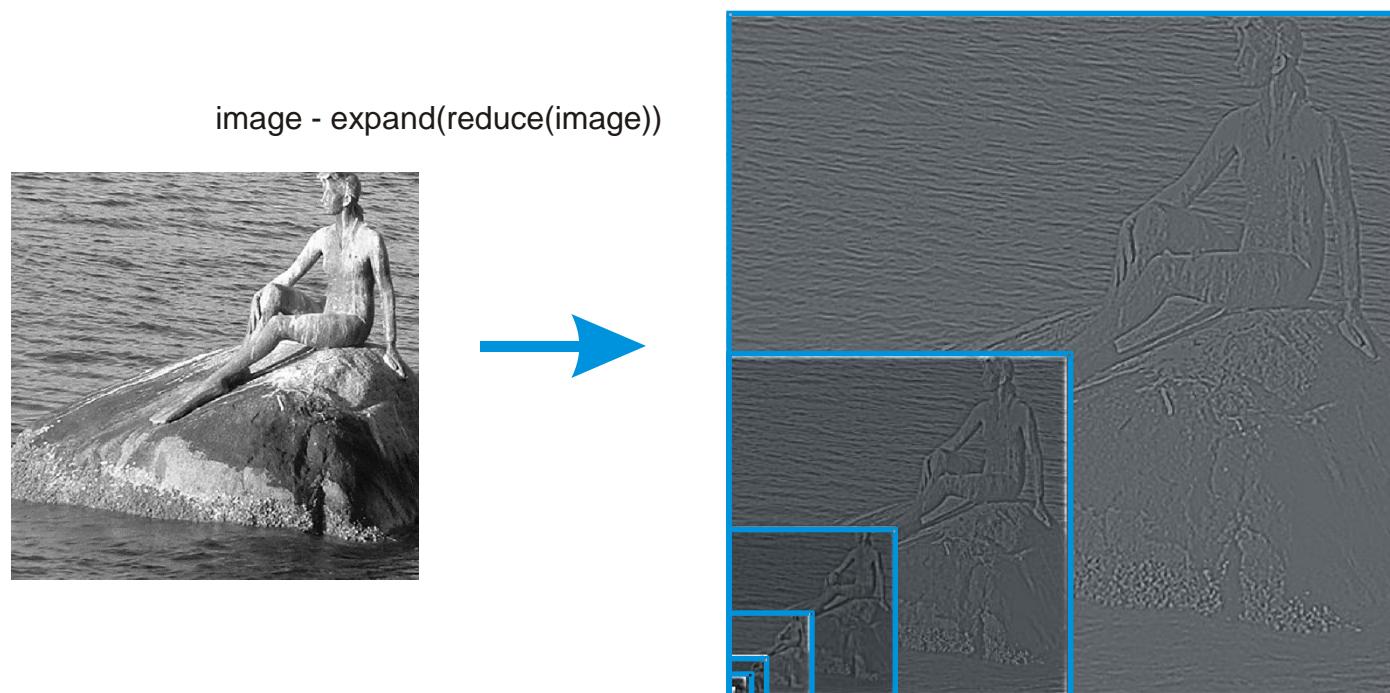
$$\mathbf{G}_{k+1} = \text{Reduce}(\mathbf{G}_k)$$

für Stufen $k = 0, 1, \dots, r$



Laplace-Pyramide (1)

Eine nahezu redundanzfreie Repräsentation erhält man, wenn auf jeder Skalenstufe k nicht das reduzierte Bild \mathbf{G}_k , sondern das Differenzbild zwischen der Expansion des nächstreduzierten Bildes und G_k selbst gespeichert wird:



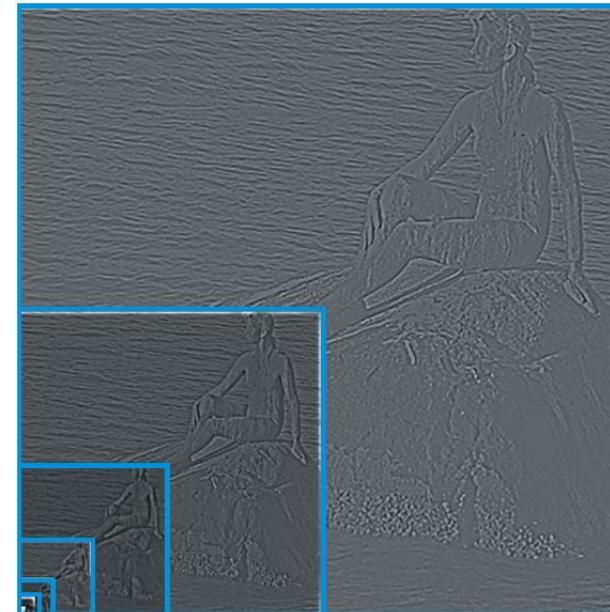
Laplace-Pyramide (2)

Der Aufbau der Laplace-Pyramide kann also wie folgt formal beschrieben werden:

$$\mathbf{L}_r = \mathbf{G}_r$$

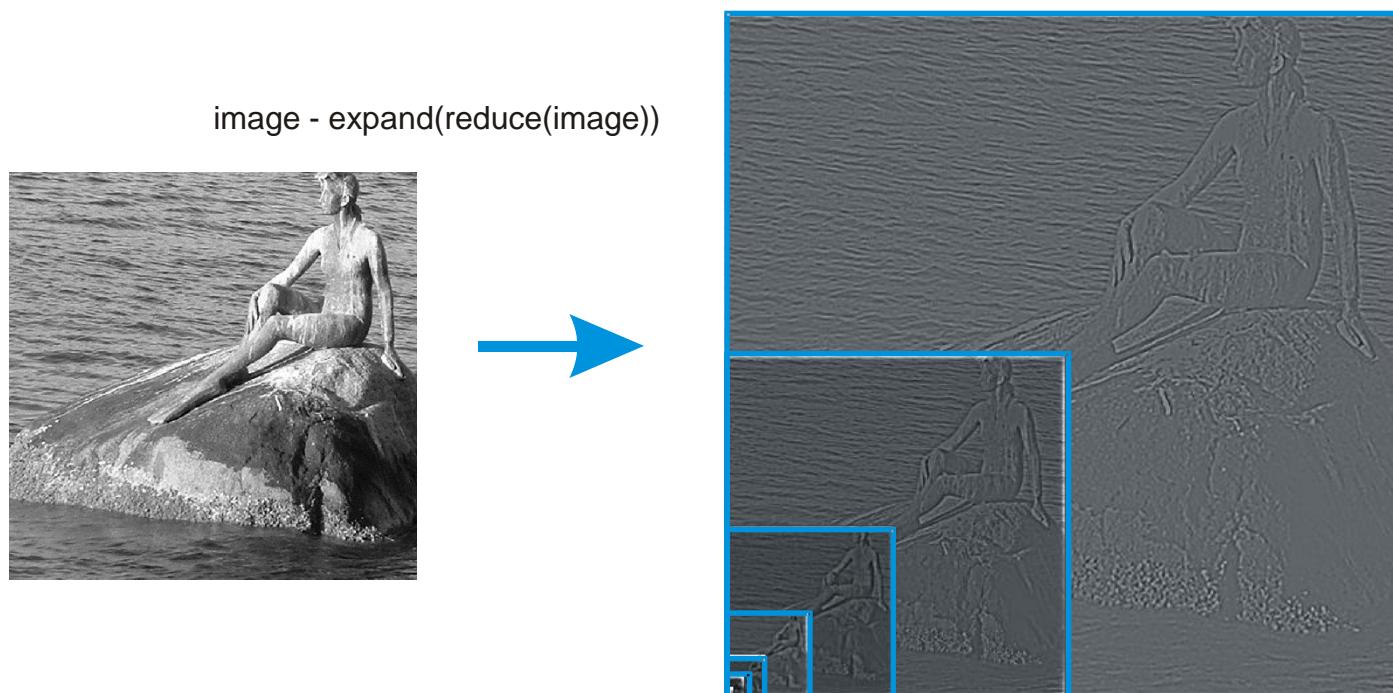
$$\mathbf{L}_k = \mathbf{G}_k - \text{Expand}(\mathbf{G}_{k+1})$$

für Stufen $k = 0, 1, \dots, r-1$



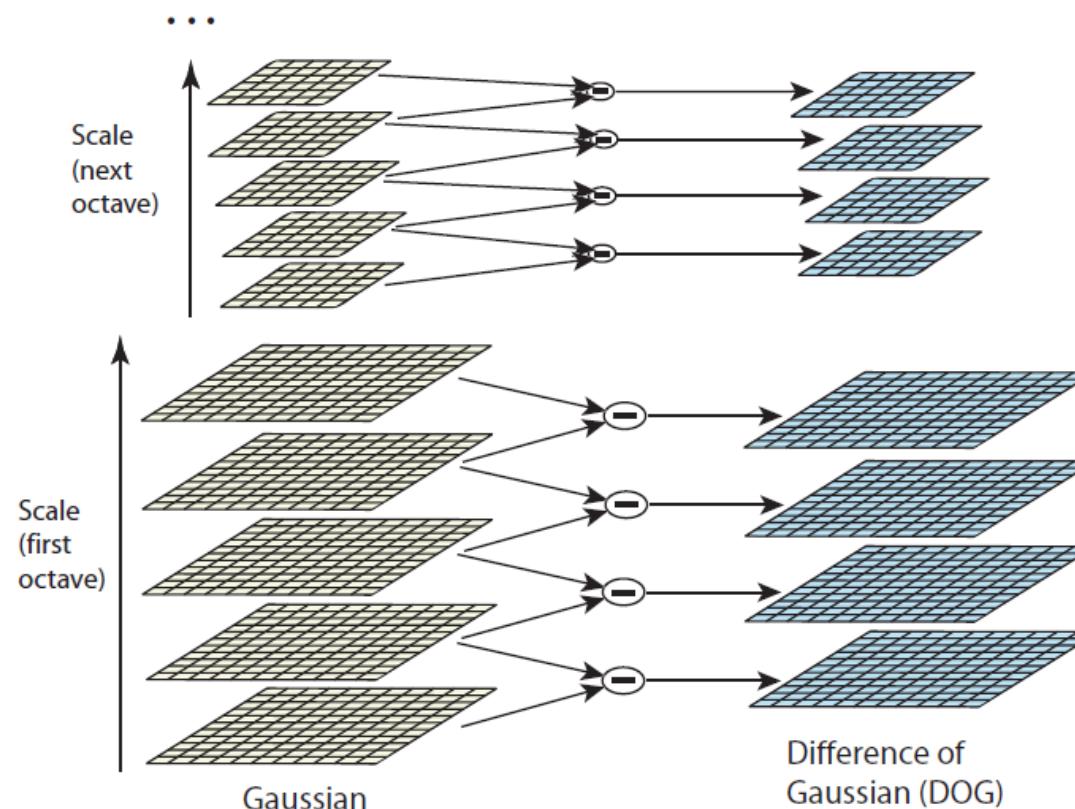
Laplace-Pyramide (3)

Faktisch wird hier der Difference-of-Gaussian-Operator (DoG) als Approximation des Laplacian-of-Gaussian-Filter (LoG-Filter) eingesetzt (s. Vorlesung 3).



Laplace-Pyramide (4)

- Gauß- und DoG-Hierarchien bilden wiederum die Basis von skaleninvarianten Ansätzen für das Finden und Beschreiben von sog. *Interest Points*. Einen grundlegenden Ansatz dazu stellt die Arbeit von D. Lowe dar.



Explizite Multiskalenrepräsentation im Ansatz von D. Lowe: *Distinctive image features from scale-invariant keypoints*. Intern. Journal of Computer Vision 60(2), 91-110, 2004.

Zusammenfassung (1)

- (Diskrete) Bildfunktionen (und damit Bilder) sind durch Basisfunktionen im Orts- und Frequenzraum darstellbar.
- Die diskrete Fourier-Transformation setzt periodische komplexe Basisfunktionen der Form $\cos(u \cdot n) + i \cdot \sin(u \cdot n)$ ein.
- Die Konvolution von Bildern mit Glättungs- und Kantenfilter im Ortsraum kann durch Multiplikation der entspr. Fourier-transformierten Bildfunktionen effizient umgesetzt werden.

Zusammenfassung (2)

- Multiskalen-Strategien versuchen, Merkmale oder Segmente in unterschiedlichen Frequenzbändern bzw. Auflösungsstufen zu ermitteln. Dies kann implizit erfolgen wie etwa in *Split-and-Merge* und der *hierarch. Wasserscheidentransformation*.
- Andere Multiskalen-Strategien setzen explizite Repräsentation von Multiskalenräumen ein.
- Die bekannteste Ansätze zur expliziten Repräsentation von Multiskalenräumen setzen Gauß-Pyramiden und Laplace-Pyramiden ein.

Anhang: Konvolution und FT

Multiplikation im Frequenzraum entspricht Konvolution im Ortsraum:

$$\begin{aligned} F(u) \cdot G(u) &= \sum_k f(k) \cdot \exp(-i2\pi uk/N) \cdot \sum_m g(m) \cdot \exp(-i2\pi um/N) \\ &= \sum_m \sum_k [f(k) \cdot \exp(-i2\pi uk/N) \cdot g(m) \cdot \exp(-i2\pi um/N)] \\ &= \sum_m \sum_k [f(k) \cdot \textcolor{blue}{g(m)} \cdot \exp(-i2\pi uk/N) \cdot \exp(-i2\pi um/N)] \end{aligned}$$

(Verschiebeeigenschaft $\textcolor{blue}{g(m)} \cdot \exp(-i2\pi uk/N) = g(m-k)$ in obigem Term)

$$\begin{aligned} &= \sum_m \sum_k f(k) \cdot \textcolor{orange}{g(m-k)} \cdot \exp(-i2\pi um/N) \\ &= \sum_m [\sum_k f(k) \cdot g(m-k)] \cdot \exp(-i2\pi um/N) \\ &= \mathbf{FT}[\sum_k f(k) \cdot g(m-k)] = \mathbf{FT}[[f^*g](m)] \end{aligned}$$