Computergrafik 2: Segmentierung

Prof. Dr. Michael Rohs, Dipl.-Inform. Sven Kratz

michael.rohs@ifi.lmu.de

MHCI Lab, LMU München

Folien teilweise von Andreas Butz, sowie von Klaus D. Tönnies
(Grundlagen der Bildverarbeitung. Pearson Studium, 2005.)

Vorlesungen

Datum	Thema
24.4.	Einführung, Organisatorisches (Übungen, Klausur)
1.5./8.5.	keine Vorlesungen (wegen 1. Mai und CHI-Konferenz)
15.5.	Abtastung von Bildern, Punktbasierte Verfahren der Bildverbesserung
22.5.	Licht, Farbe, Farbmanagement
30.5.	Konvolution, Filterung im Ortsraum (Verschiebung wegen Pfingstdienstag)
5.6.	Fouriertransformation: Grundlagen
12.6.	Filterung im Frequenzraum
19.6.	Kanten, Linien, Ecken
27.6.	Segmentierung
3.7.	Segmentierung, Morphologische Operationen
10.7.	Klassifikation
17.7.	Image Matching
24.7.	Klausur (Hörsaal M 018 im Hauptgebäude, 14-16 Uhr)

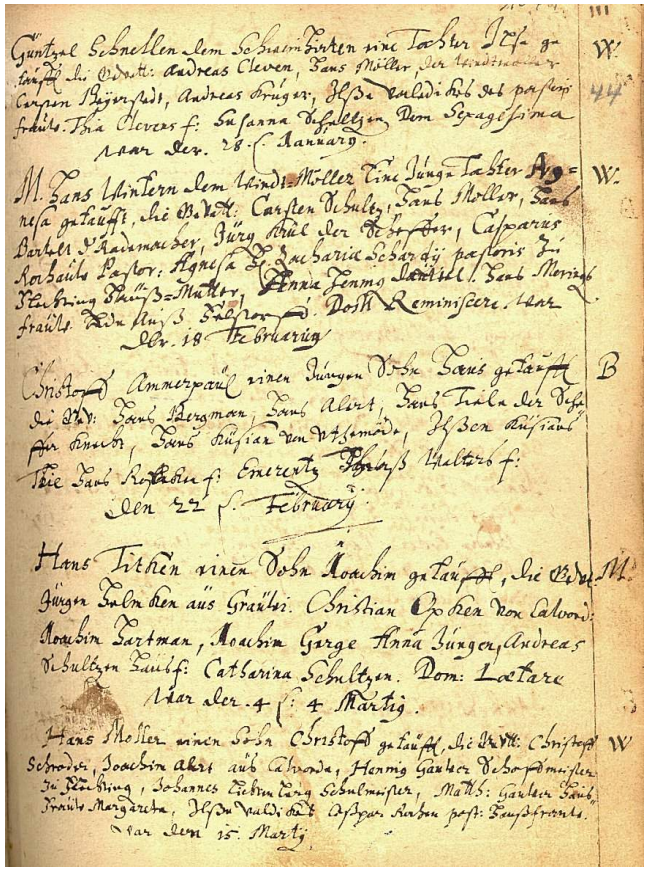
Themen heute

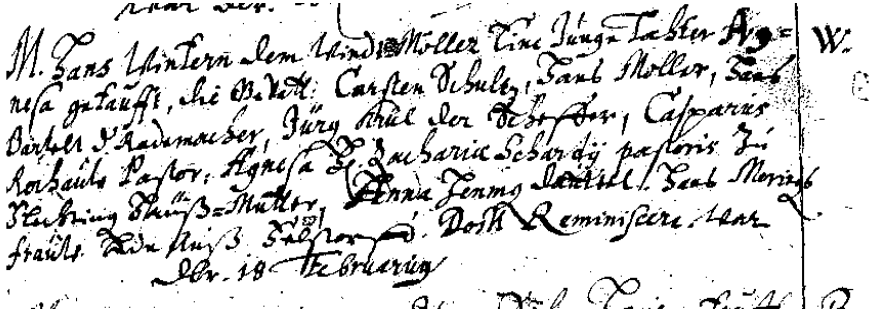
- Histogrammbasierte Segmentierung
  - optimaler Schwellenwert: Algorithmus von Otsu
  - Variable Schwellenwerte
  - Region labeling, flood fill
- Regionenbasierte Segmentierung
  - Region Merging, Split & Merge
  - Textursegmentierung
- Kantenbasierte Segmentierung
  - Wasserscheidentransformation
  - Verbesserung der Wasserscheidentransformation durch Marker
- Modellbasierte Segmentierung
  - Region Growing
  - Kantenverfolgung

SEGMENTIERUNG

Segmentierung

- Zerlegung eines Bildes in semantische Einheiten
- **Segmente:** Träger von Bedeutung der Strukturen eines Bildes
- Eigenschaften
  - **vollständig**: jedes Pixel ist einem Segment zugeordnet
  - **überdeckungsfrei**: ein Pixel ist höchstens einem Segment zugeordnet
  - **zusammenhängend**: jedes Segment bildet ein zusammenhängendes Gebiet

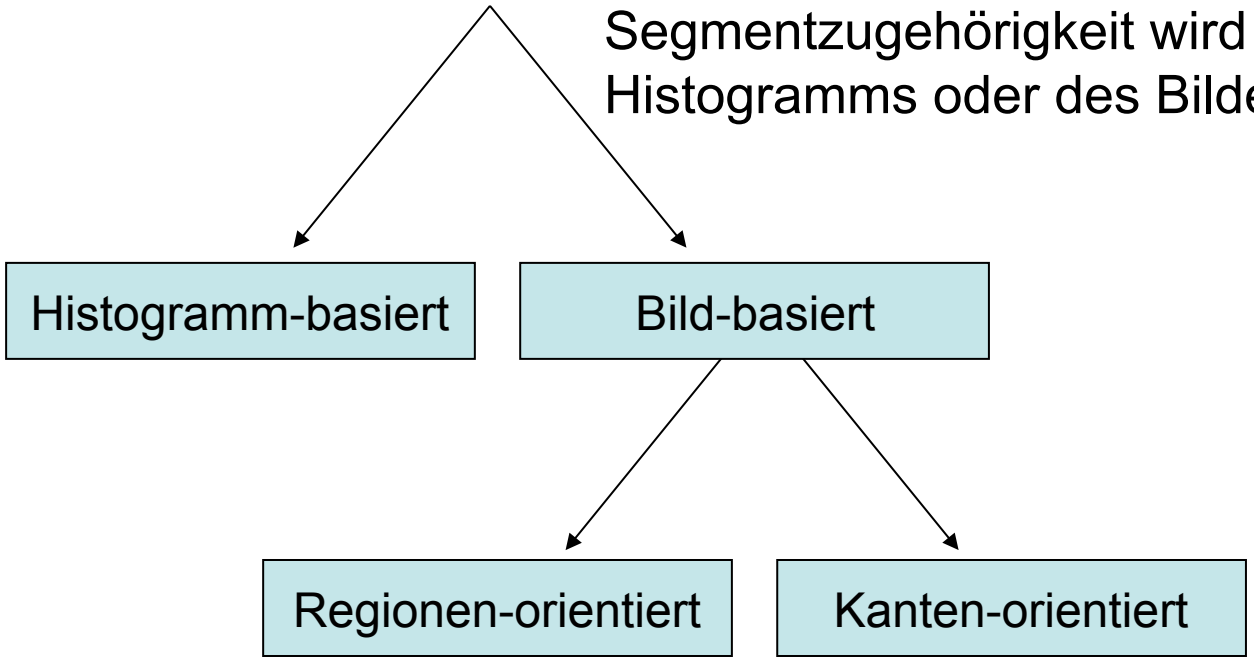




Segmentierungsmethoden

Histogramm-basierte oder
Bild-basierte Segmentierung:

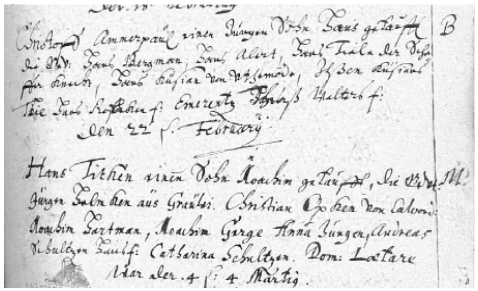
Segmentzugehörigkeit wird anhand des Histogramms oder des Bildes entschieden

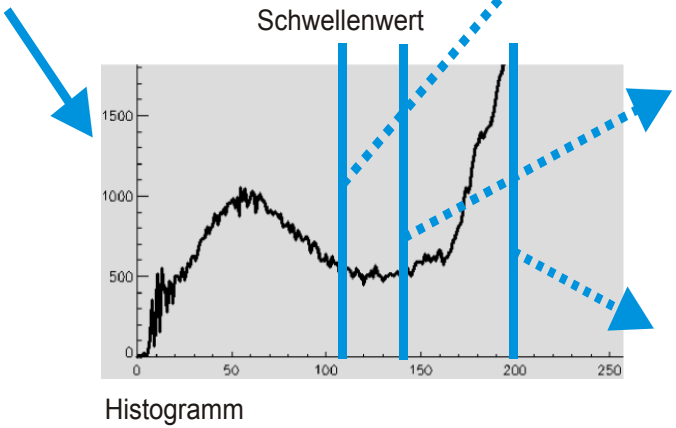


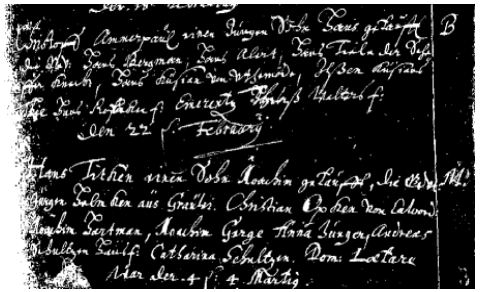
Regionen- oder Kantenorientierung:

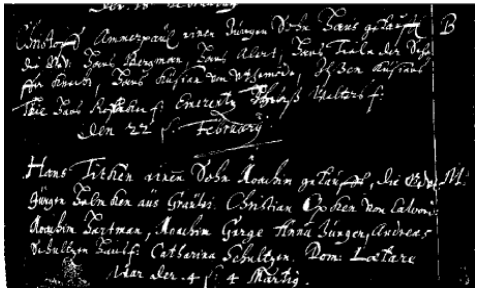
Segmente werden durch ihr Inneres oder ihre Grenzen definiert

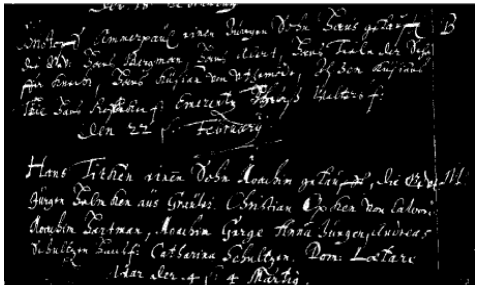
Histogrammbasierte Segmentierung











Annahme: Bild besteht aus zwei Anteilen (Vordergrund, Hintergrund) die sich durch Grauwert unterscheiden

Aufgabe: Schwellenwert zwischen Vorder- und Hintergrund finden

$$b(x,y) = \begin{cases} 0 & falls\ f(x,y) > T \\ 1 & sonst \end{cases}$$

Optimaler globaler Schwellenwert: Das Verfahren von Otsu

- Bilde Klassen C_1 (dunkle Pixel) und C_2 (helle Pixel)
- Optimaler Schwellenwert k* maximiert der Varianz zwischen den Klassen

- Annahmen über Bild:
- MxN Pixel
- Grauwertstufen {0, 1, 2, ..., L-1}
- n_i Pixel mit Grauwert i (also MN = $n_0 + n_1 + ... + n_{L-1}$)
- Normalisiertes Histogramm: $p_i = \frac{n_i}{MN}, \quad \sum_{i=0}^{L-1} p_i = 1, \; p_i \geq 0$

Optimaler globaler Schwellenwert: Das Verfahren von Otsu

- Schwellenwert k zur Segmentierung in Klassen C_1 und C_2 so dass Pixel mit Grauwerten [0,k] in C_1 und [k+1,L-1] in C_2
- $P_1(k)$: Wahrscheinlichkeit, dass Pixel in C_1:

$$P_1(k) = \sum_{i=0}^{k} p_i \qquad P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k)$$

- Durchschnittlicher Grauwert in C_1: $m_1(k) = \frac{1}{P_1(k)} \sum_{i=0}^{k} i \cdot p_i$
- Durchschnittlicher Grauwert in C_2: $m_2(k) = \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} i \cdot p_i$
- Durchschnittlicher Grauwert des Bildes: $m_G = \sum_{i=0}^{L-1} i \cdot p_i$

Optimaler globaler Schwellenwert: Das Verfahren von Otsu

- Güte des Schwellenwerts k: $\eta(k) = \frac{\sigma_B^2(k)}{\sigma_G^2}$

- σ_B^2 ist Varianz zwischen C_1 und C_2:

$$\sigma_B^2(k) = P_1(k) \cdot (m_1(k) - m_G)^2 + P_2(k) \cdot (m_2(k) - m_G)^2$$

- σ_G^2 ist globale Varianz der Pixelgrauwerte:

$$\sigma_G^2(k) = \sum_{i=0}^{L-1} p_i \cdot (i - m_G)^2$$

- optimaler Schwellenwert:

$$k^* = \arg\max_{0 \le k \le L-1} \sigma_B^2(k)$$

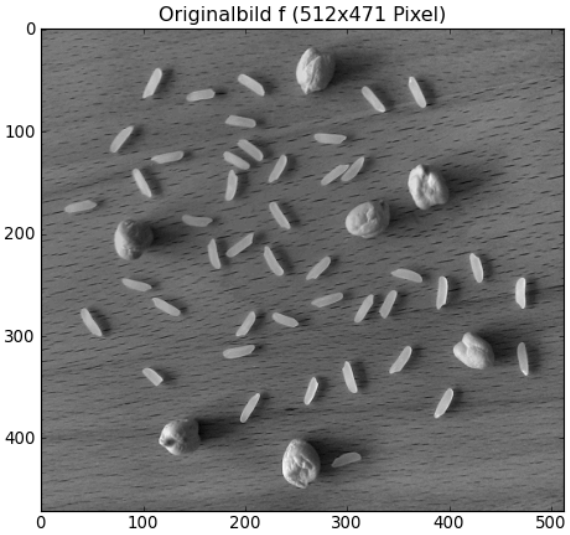
Segmentierung:

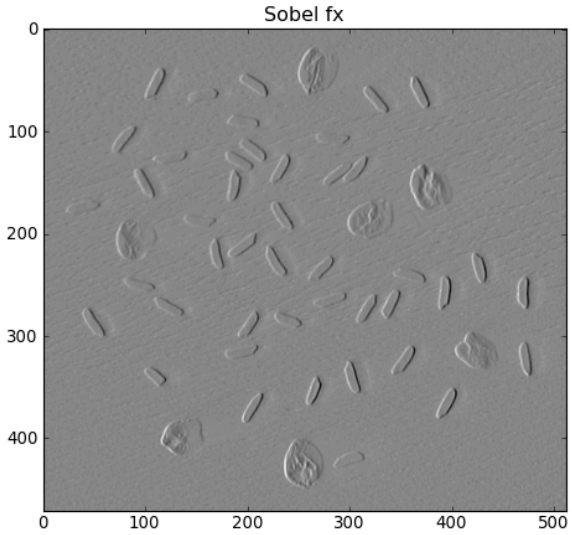
$$g(x,y) = \begin{cases} 1 & falls\ f(x,y) > k^* \\ 0 & sonst \end{cases}$$

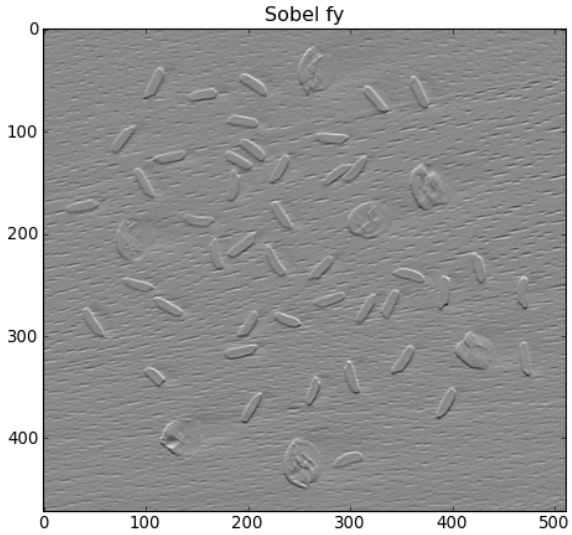
Verbesserung der histogrammbasierten Segmentierung durch Kanten

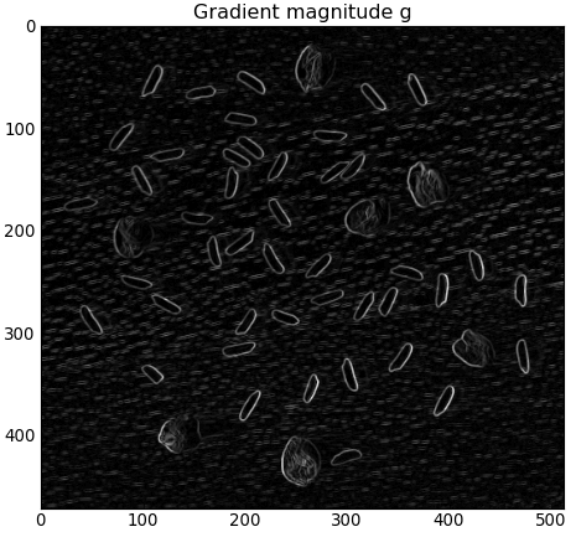
- Idee: Nur Pixel in Histogramm eintragen, die auf (oder nahe an) Kanten liegen
  - gleiche Wahrscheinlichkeit, dass solche Pixel auf Vorder- oder Hintergrund liegen
  - Gradienten- oder Laplace-Operatoren zum Finden der Kanten (aber: nicht alle Kanten separieren Vorder- und Hintergrund)
- Algorithmus
  - Kantenbild g von f berechnen (Gradienten-/Laplace-Operatoren)
  - Pixel (x,y) auf Kantenbild g auswählen mit g(x,y) > T
  - Histogramm h von f mit ausgewählten Pixeln berechnen
  - Methode von Otsu auf h zur globalen Segmentierung anwenden

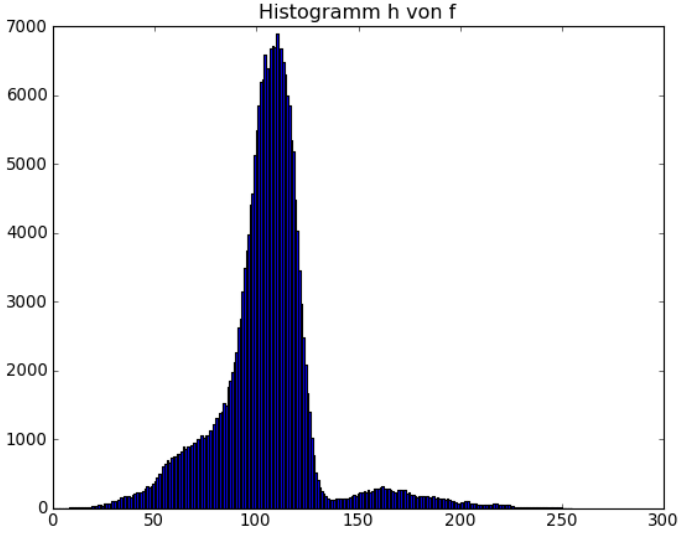
Histogramm aller Pixel

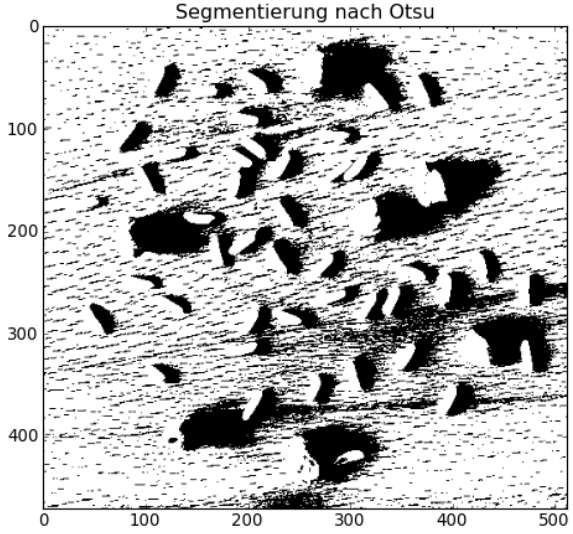




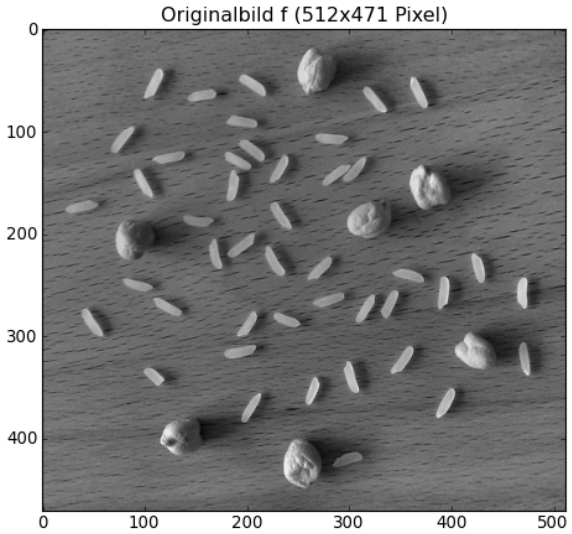


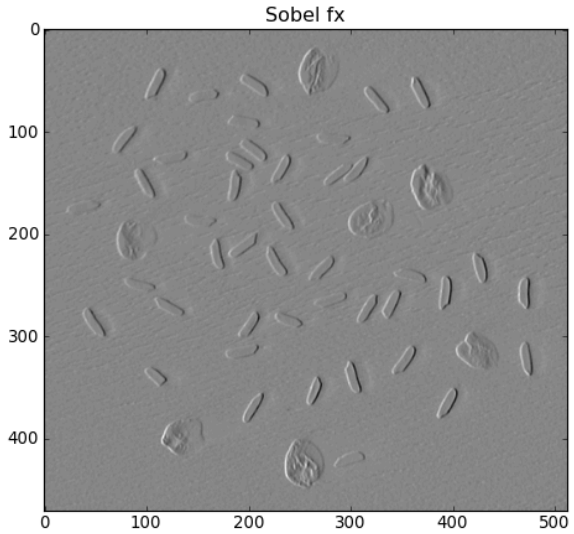


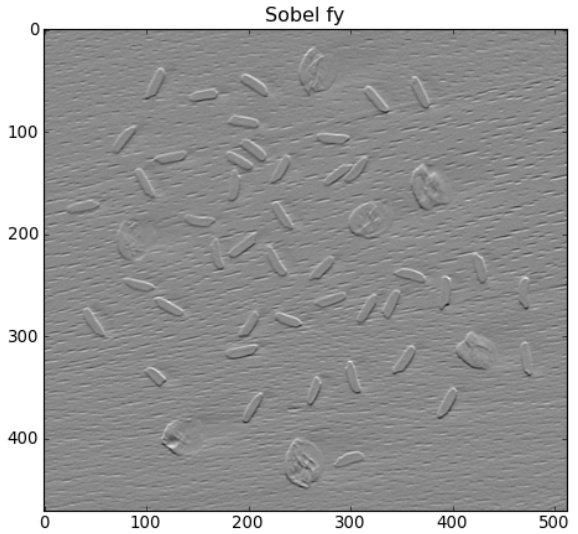


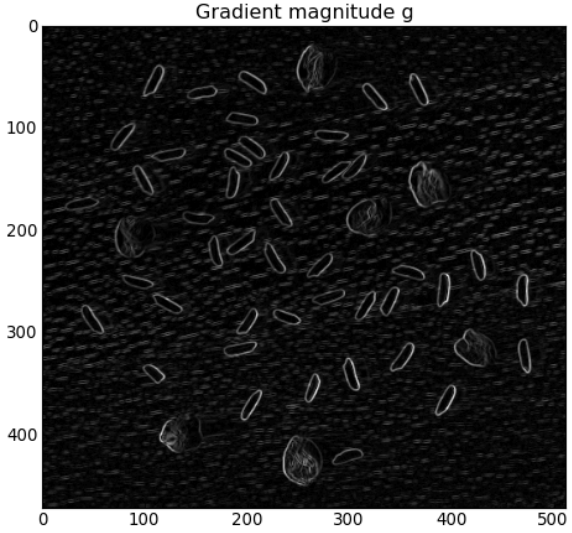


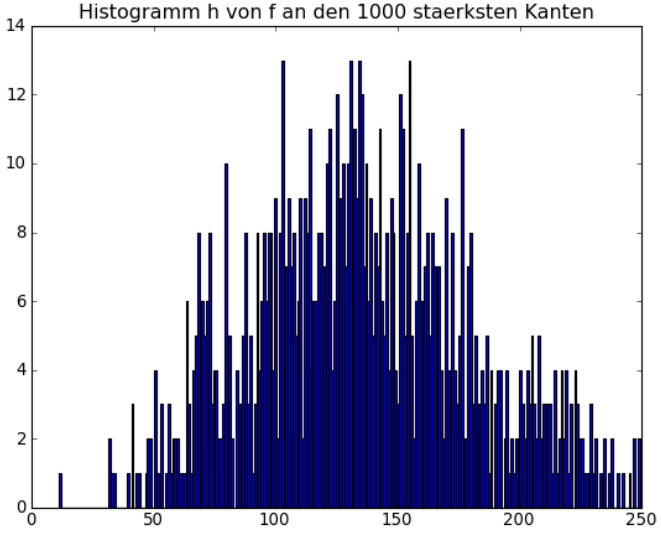
Histogramm der Pixel an den Stellen der 1000 längsten Gradienten

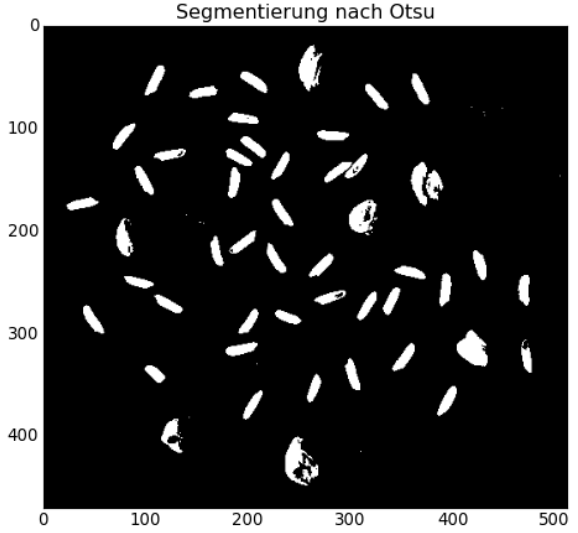












Region Labeling per Flood Fill

- Schwellenwert zerlegt das Bild in Vordergrund- und Hintergrundsegmente
- **Region Labeling** bestimmt Ort und Anzahl aller zusammenhängenden Gebiete im Binärbild *b*:

```
initialise() // Region der Größe M,N erzeugen und
label=0                    // mit 0 initialisieren, Startlabel=1
for (i,j) = (0,0) to (M-1,N-1) do // Doppelschleife über i und j
  if labels(i,j) == 0 then         // dieser Ort ist noch nicht
                                   // Teil einer Region
    label = label + 1              // neues Label vergeben
    flood_fill(i,j,label)          // zusammenhängendes
                                   // Gebiet um (i,j) mit
                                   // Label füllen
```

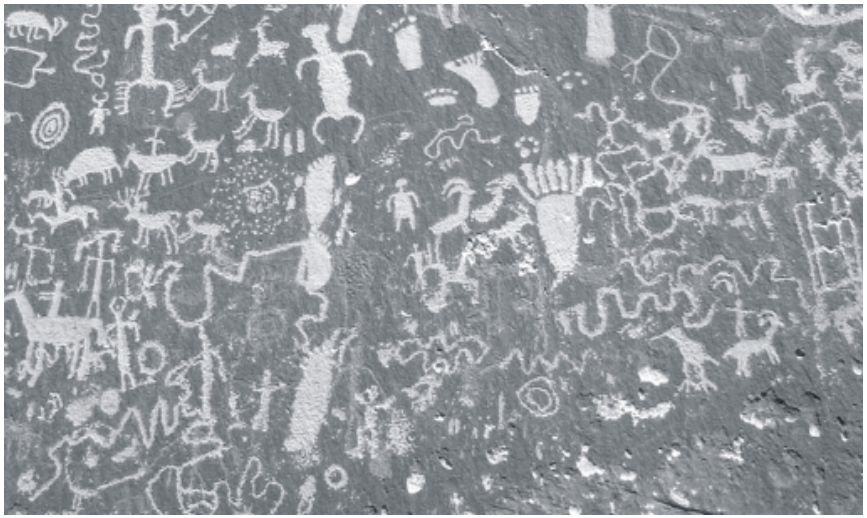
Region Labeling per Flood Fill

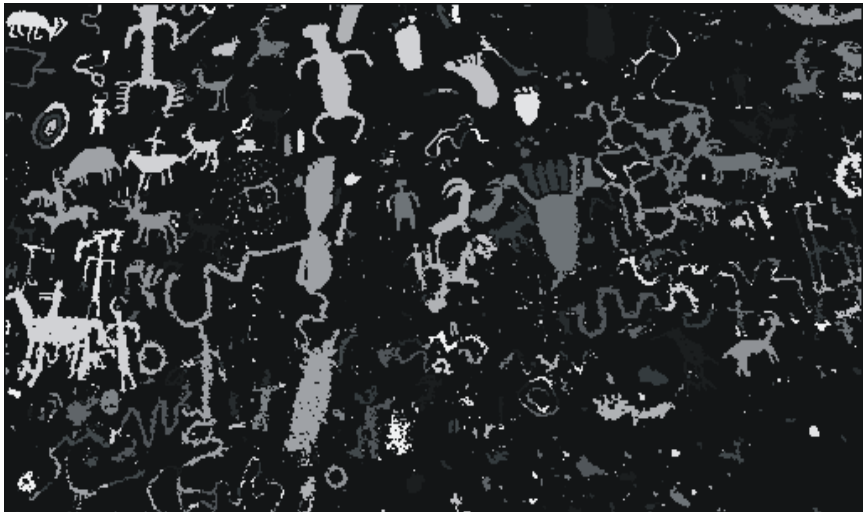
```
flood_fill(i,j,label)  // Variablen zur Auswertung der
        Zusammenhangsbedingung sind global verfügbar
if f(i,j) erfüllt Zusammenhangsbedingung then
    region(i,j) = label      // Region an (i,j) mit Label
                             // versehen
    flood_fill(i-1,j,label) // Nachbarpixel untersuchen
    flood_fill(i,j-1,label)
    flood_fill(i+1,j,label)
    flood_fill(i,j+1,label)
```

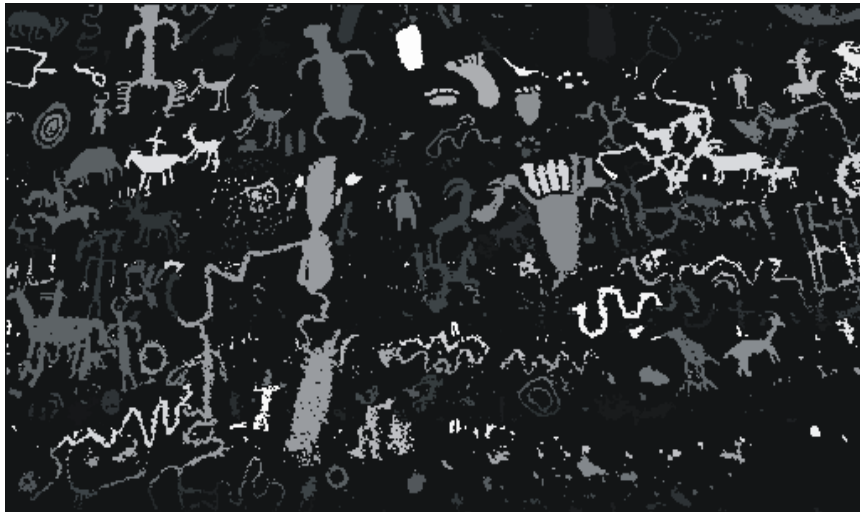
Region labeling:
- vollständige Segmentierung?
- überdeckungsfrei?
- zusammenhängend?

Bsp. für **Zusammenhangsbedingung**: hat den gleichen Grauwert wie Saatpunkt

Resultat







(einfache) Merkmale der Regionen

- Größe, Bounding Box
- Schwerpunkt
- Länge und Orientierung der Haupt- und Nebenachsen
  - Eigenwerte und Eigenvektoren der Kovarianzmatrix M mit

$$M(segment) = \begin{pmatrix} f_{xxSum}(segment) & f_{xySum}(segment) \\ f_{xySum}(segment) & f_{yySum}(segment) \end{pmatrix}$$

mit

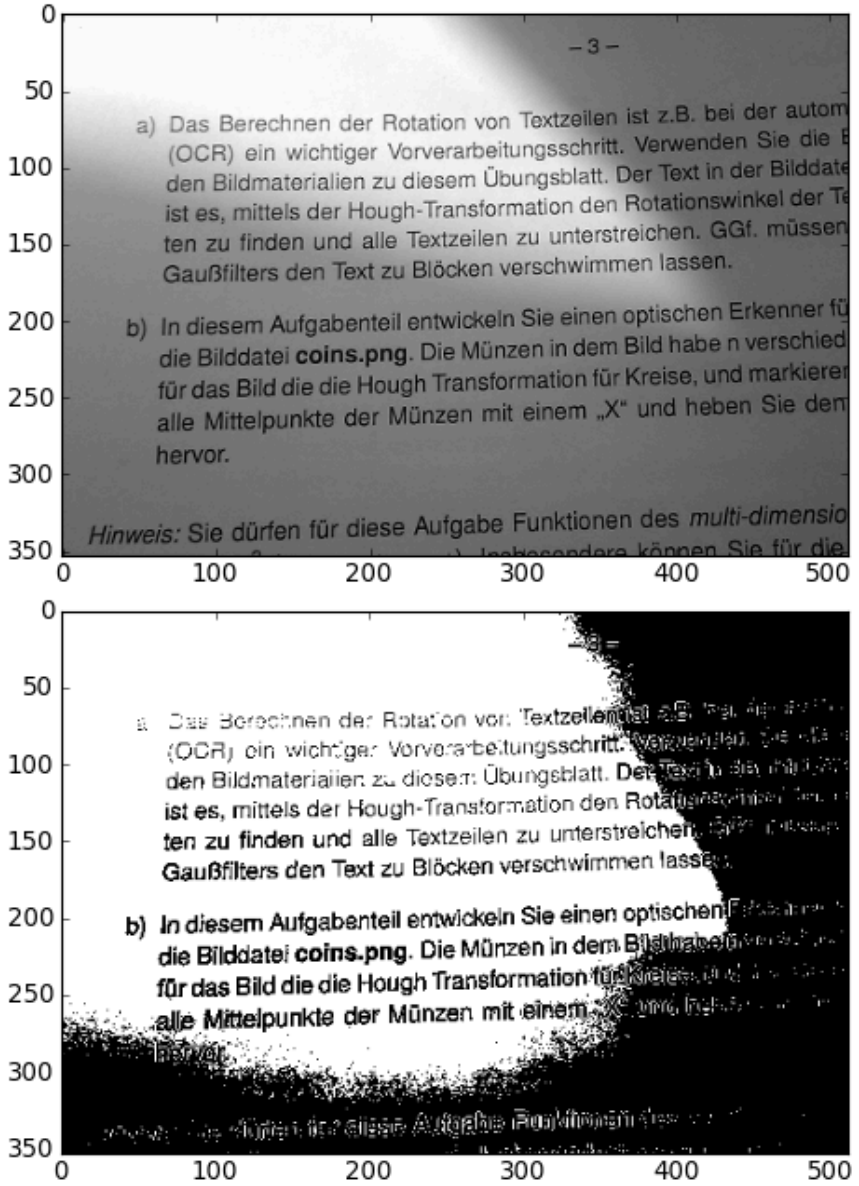
$$f_{xxSum}(segment) = \frac{1}{segment.size} \sum_{(x,y) \in segment} (x - segment.center.x)^2$$

$$f_{yySum}(segment) = \frac{1}{segment.size} \sum_{(x,y) \in segment} (y - segment.center.y)^2$$

$$f_{xySum}(segment) = \frac{1}{segment.size} \sum_{(x,y) \in segment} (x - segment.center.x)(y - segment.center.y)$$

Shading

- Helligkeitsvariationen zerstören die bimodale Verteilung der Häufigkeiten
  - ungleichmäßige Beleuchtung
  - Schatten
  - unterschiedliche Reflexionseigenschaften der Oberfläche
- Schwellenwert ist nicht mehr global (für das gesamte Bild) definierbar

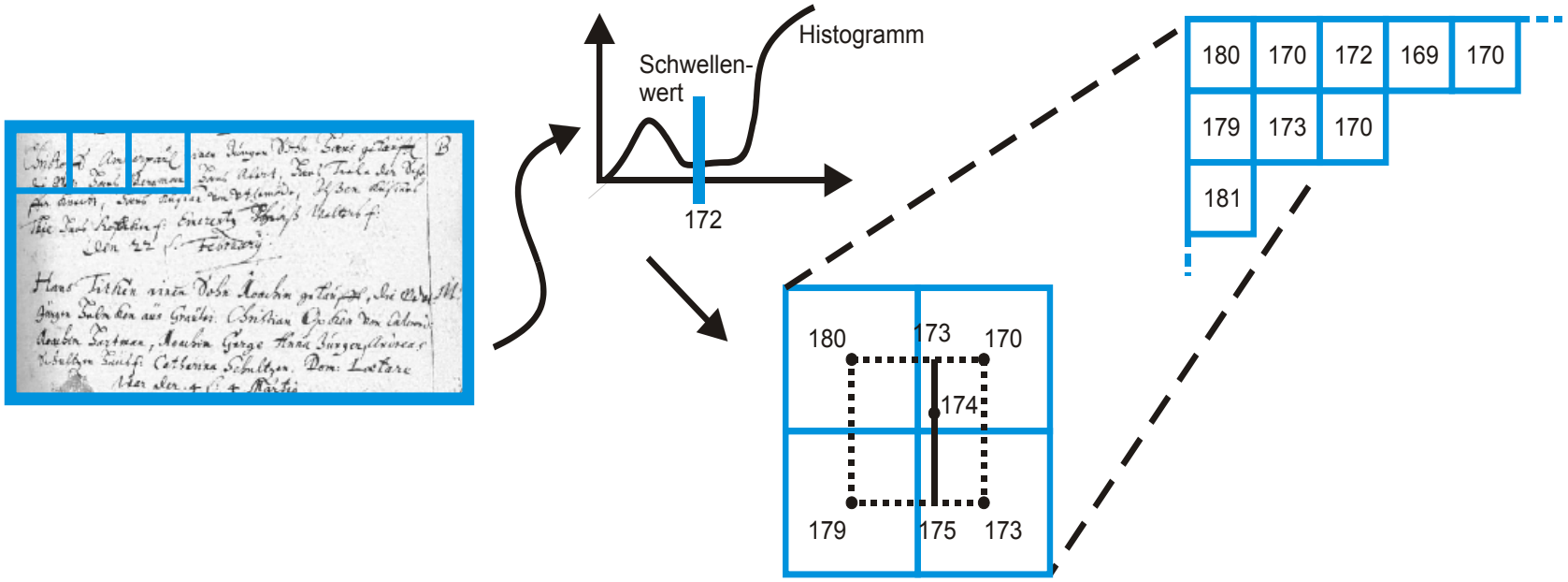


Berücksichtung von Shading

- Homogenes Bild unter gleichen Bedingungen aufnehmen
- Shading-Bild aus dem Bild selbst bestimmen
- Lokale Schwellenwerte

- Shading invertieren/subtrahieren

Lokale Schwellenwerte



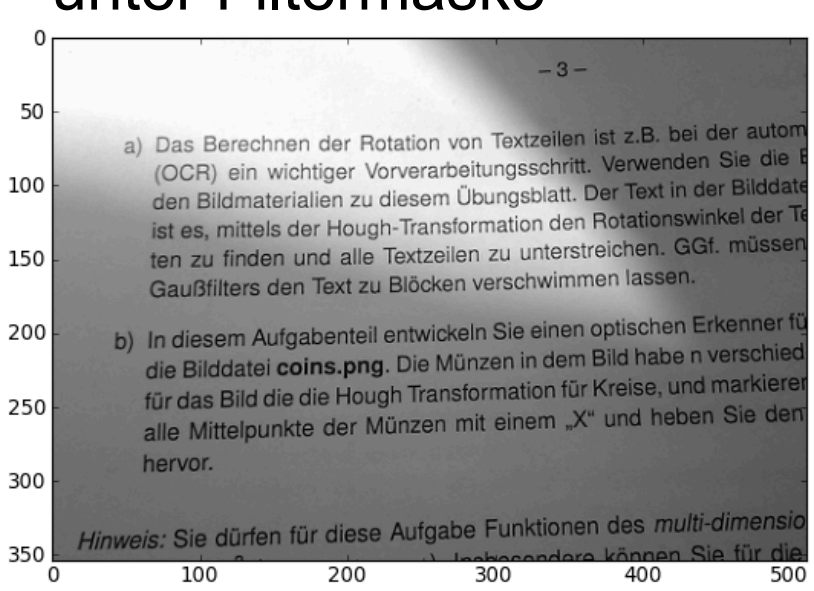
Lokale Schwellenwerte aus Histogrammen in Teilregionen

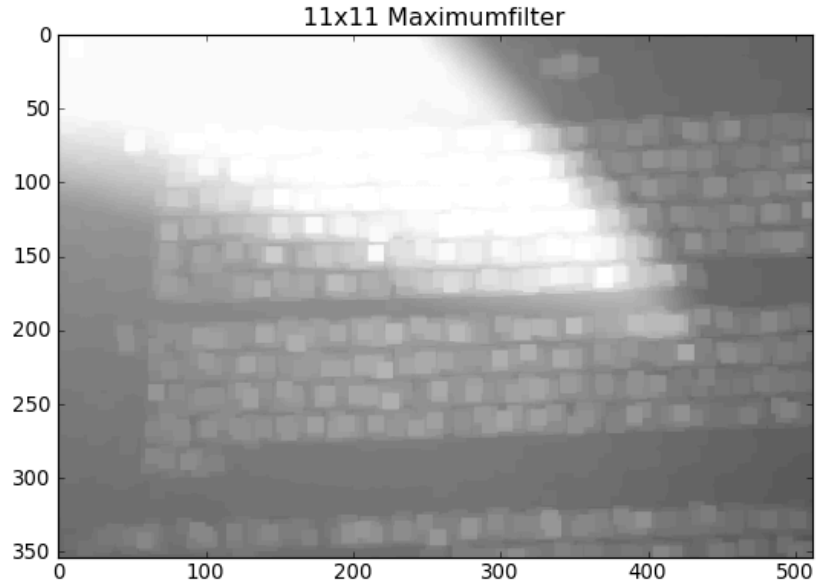
(Lineare) Interpolation von Schwellenwerten $T(i,j)$ an allen anderen Punkten

Segmentierung durch $f(i,j) > T(i,j)$

Bestimmung des Shading-Bilds

- Falls überwiegender Anteil des Bildes Vorder- oder Hintergrundpixel: Shading-Bild durch Rangordnungsfilter (Minimum- bzw. Maximumfilter) erzeugbar
- Rangordnungsfilter mindestens so groß, dass immer mindestens ein Vordergrund- und ein Hintergrundpixel unter Filtermaske

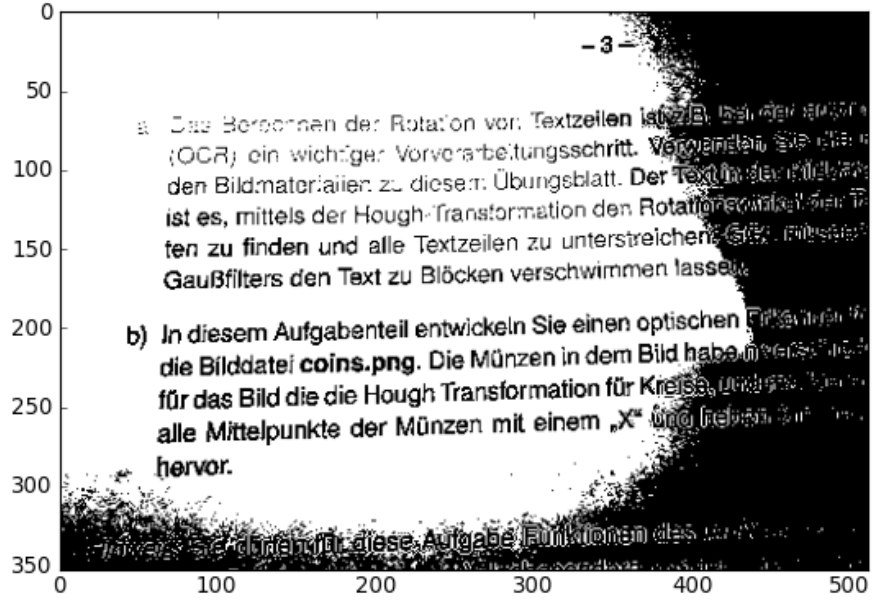




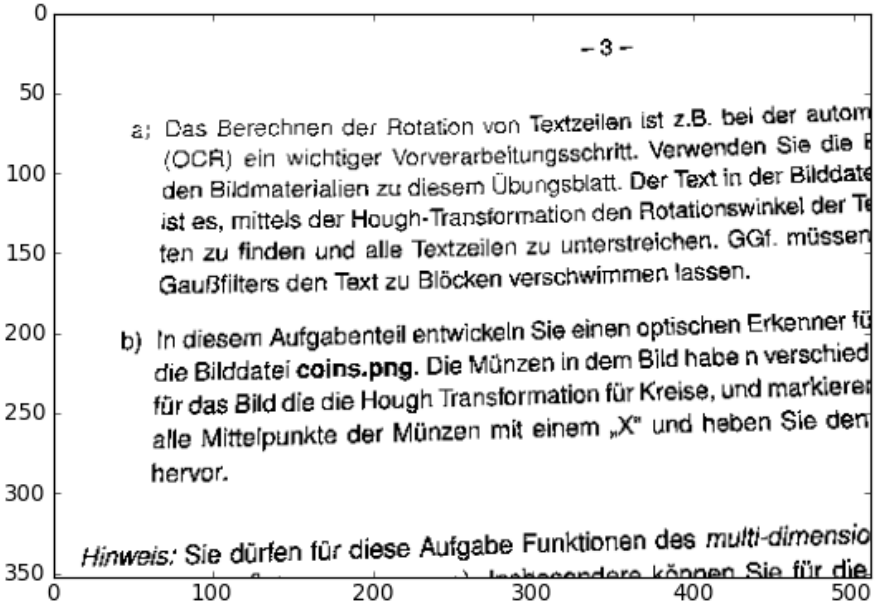
Shading-Korrektur

- Berechne aus dem Hintergrundbild Shading-Funktion $s(i,j)$
- Korrektur:
  $g(i,j) = f(i,j) - s(i,j)$
  bzw.
  $g(i,j) = f(i,j) / s(i,j)$
- Segmentierung auf dem korrigierten Bild

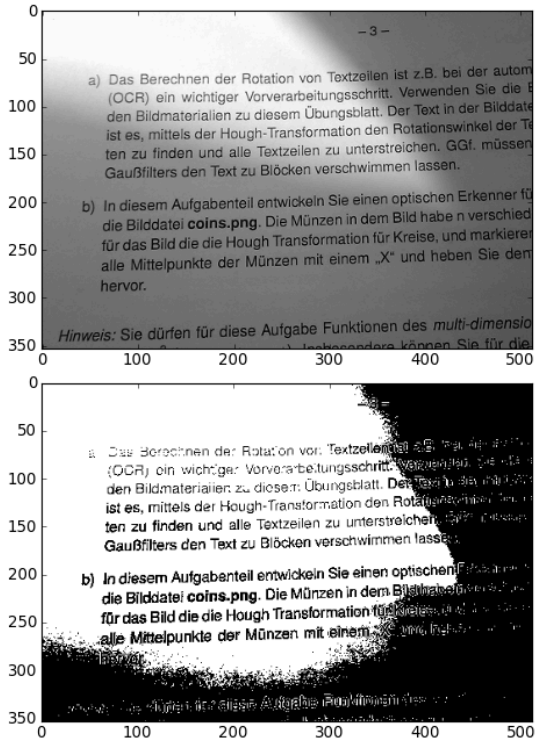
ohne Korrektur



mit Korrektur



Adaptive Schwellenwerte

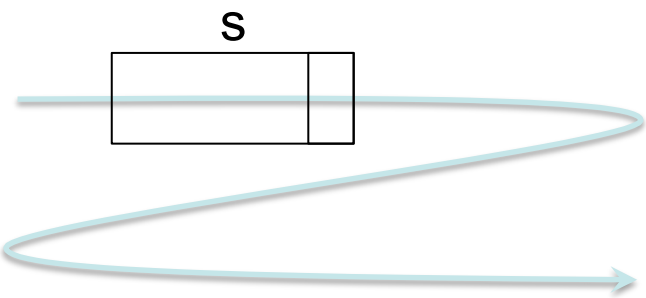


- Globale Schwellenwerte problematisch
  - bei Shading nicht oft mehr global (für das gesamte Bild) definierbar
- Aufteilung des Bildes auch problematisch
  - Aufteilung klein genug, dass Beleuchtung (nahezu uniform), aber nicht so klein, dass nur Hintergrund- oder nur Vordergrundpixel
- Adaptive Schwellenwerte
  - Schwellenwert wird an jedem Punkt im Bild neu berechnet
  - typische Strategie: zeilenweises Durchlaufen im zick-zack, gleitender Durchschnitt der letzten n Grauwerte

Wellner's Adaptive Thresholding

- Zeilenweises Durchlaufen im Zick-zack
- (*s*-facher) Durchschnittsgrauwert der letzten *s* Pixel:

$$g_s(n) = g_s(n-1) \cdot \left(1 - \frac{1}{s}\right) + p_n, s = \frac{w}{8}$$



- Initialisierung: $g_s(0) = s\frac{c}{2}$
- Schwellenwert:

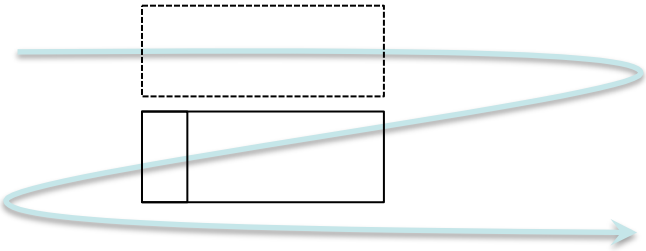
$$T(n) = \begin{cases} 0 & falls\ p_n < \frac{g_s(n)}{s} \cdot \frac{100-t}{100} \\ 1 & sonst \end{cases}, t = 15$$

Pierre D. Wellner. Adaptive thresholding for the DigitalDesk. Technical Report EPC-93-110, Rank Xerox Research Centre, Cambridge, UK, 1993.

Wellner's Adaptive Thresholding

- Verbesserung: Berücksichtigung des durchschnittlichen (s-fachen) Grauwerts über dem aktuellen Pixel

$$h_s(n) = \frac{1}{2}\left(g_s(n) + g_s(n-w)\right)$$



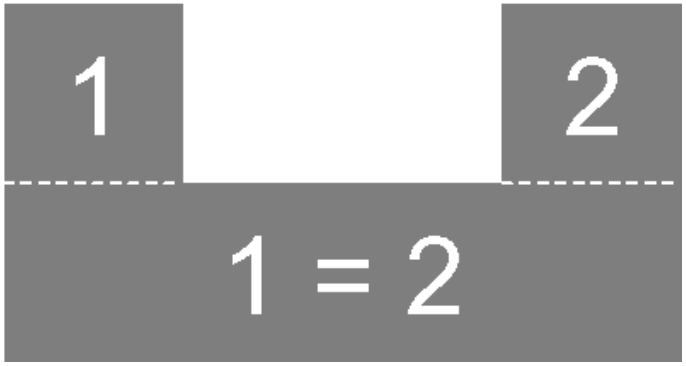
$$T(n) = \begin{cases} 0 & falls\ p_n < \dfrac{h_s(n)}{s} \cdot \dfrac{100-t}{100} \\ 1 & sonst \end{cases}, t = 15$$

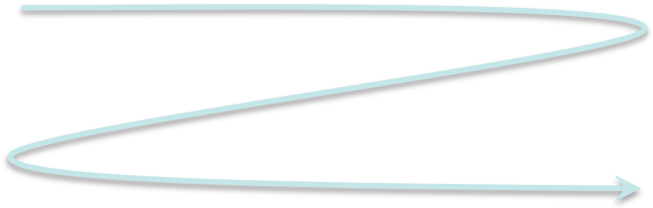
Pierre D. Wellner. Adaptive thresholding for the DigitalDesk. Technical Report EPC-93-110, Rank Xerox Research Centre, Cambridge, UK, 1993.

Region Labeling 1. Phase: Vorläufige Label

- Basiert auf 4-Nachbarschaft
- Falls auf Vordergrund-Pixel p gestoßen, betrachte bearbeitete Nachbarpixel darüber und links davon (falls aktuelle Bildzeile v.l.n.r. durchlaufen) bzw. rechts davon (sonst)
- Falls beide Nachbarn (darüber, links/rechts) Vordergrund, dann setze Label(p) = Label(darüber)
  - falls Label(darüber) ≠ Label(links/rechts), dann markiere beide Label als äquivalent (Äquivalenzketten)
- sonst, falls einer der Nachbarn (darüber, links/rechts) Vordergrund, dann setze Label(p) auf dessen Label
- sonst vergebe neues Label für p
  - füge in eigene Äquivalenzkette ein

Region Labeling 2. Phase: Auflösen der Äquivalenzen

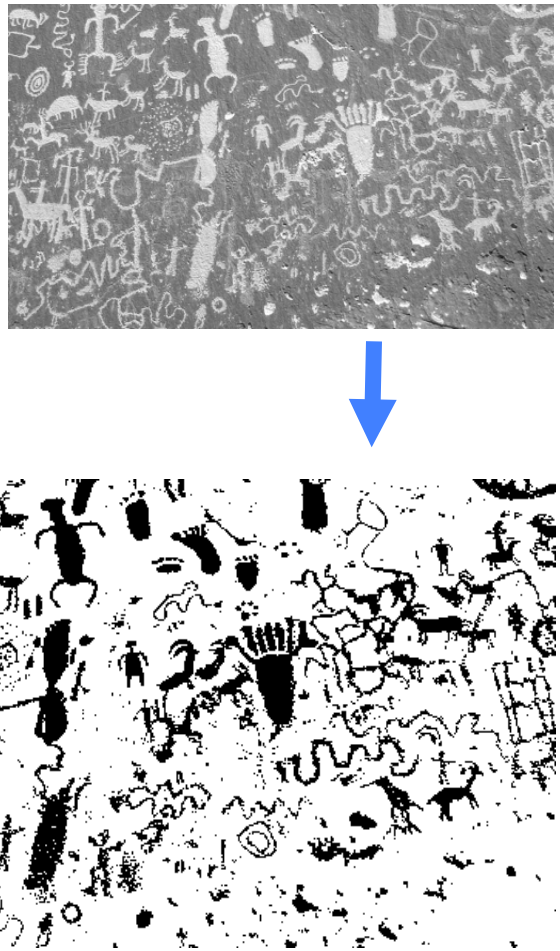




- Label-Äquivalenzen werden möglicherweise erst später erkannt (→ Durchlauf zeilenweise)
- Labeling 2. Phase: Auflösen der Äquivalenzen
  - Druch Äquivalenzkette laufen und jedem Pixel in der gleichen Äquivalenzkette das gleiche finale Label geben

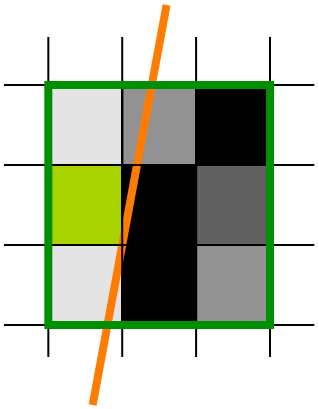
Nachverarbeitung

- Trennung nach Grauwerten nicht perfekt
- Schwellenwertbild enthält falsche Regionen
  - kleine fälschlicherweise als Segmente identifizierte Regionen
  - Störungen am Rand von Regionen
- Nachverarbeitung
  - Medianfilterung auf den Labeln
  - Entfernung von zu kleinen Regionen



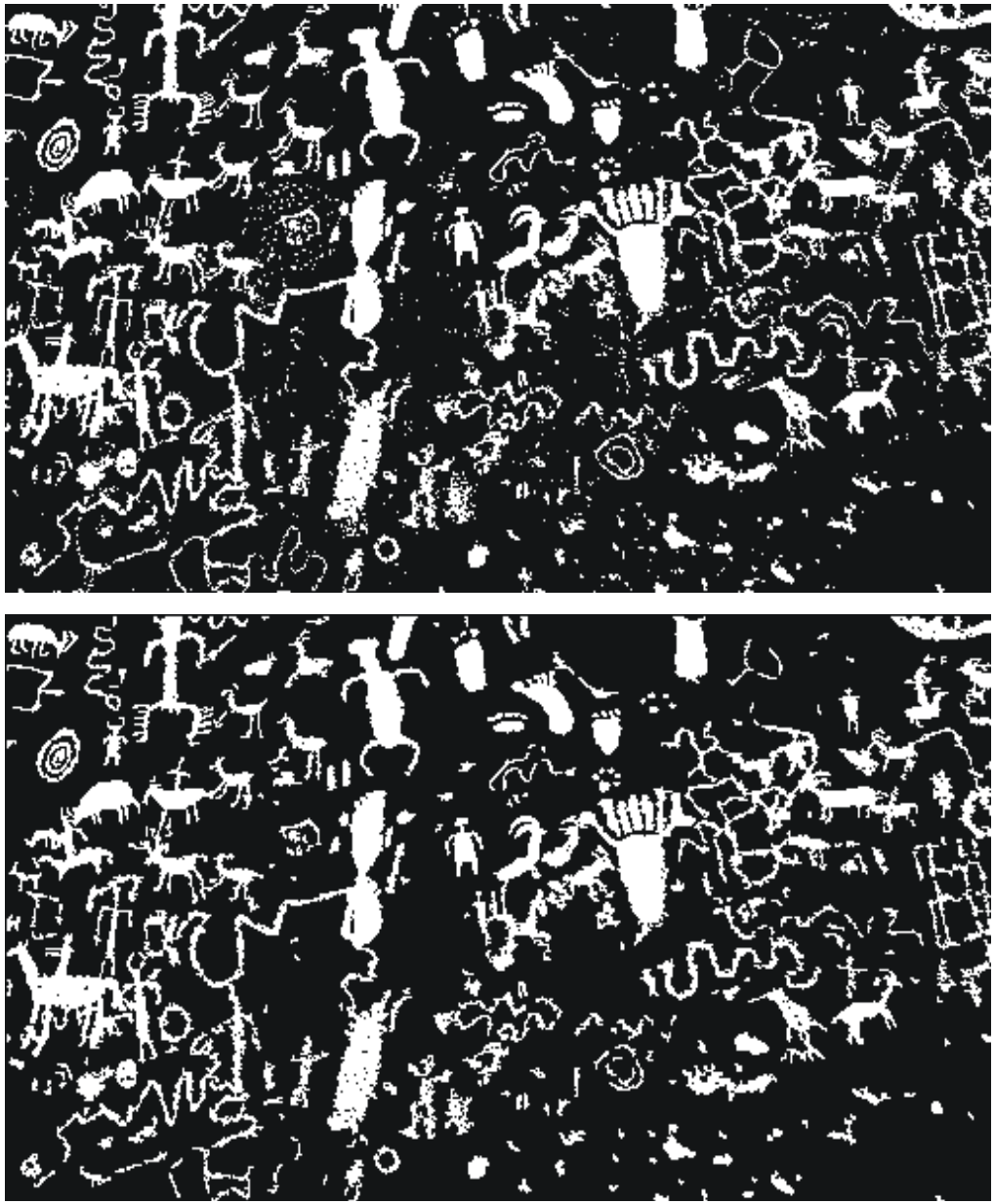
Medianfilterung auf Labeln

Wieso geht das überhaupt?
→ siehe Def. Medianfilter



Entfernung kleiner Gebiete

- Morphologische Operationen
  → nächste Vorlesung

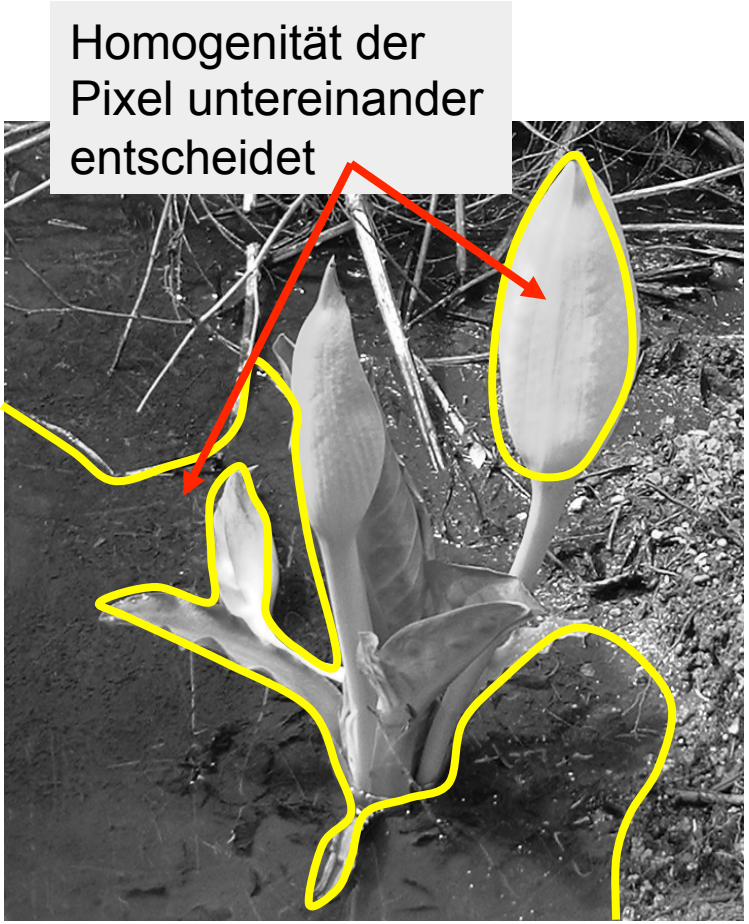


Regionenbasierte Segmentierung

- Region Merging
- Multiskalenstrategie
- Split-and-Merge
- Textursegmentierung

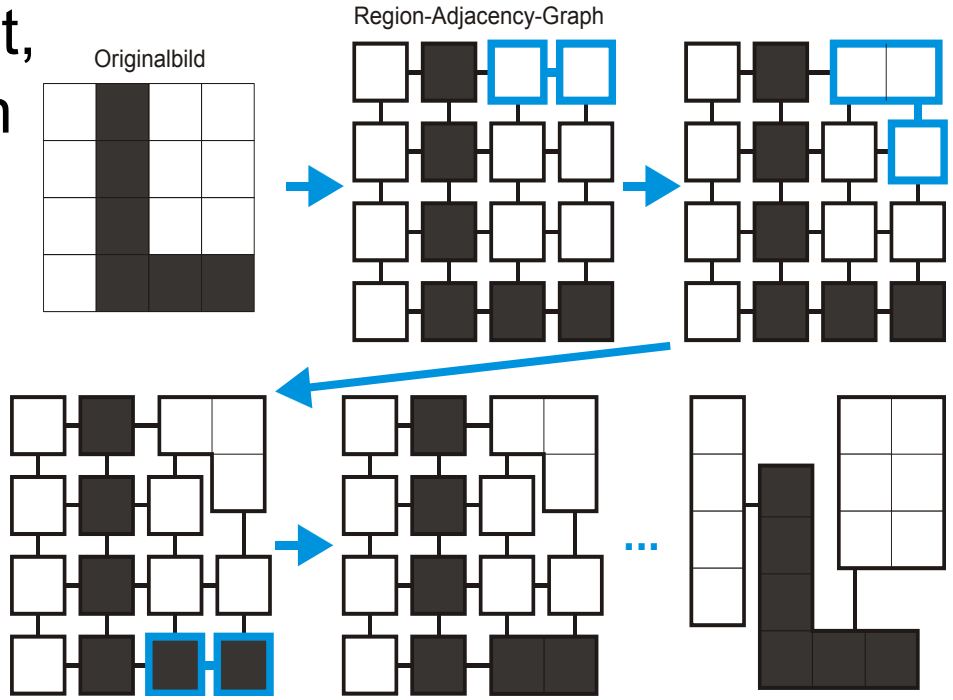
Regionenbasierte Segmentierung

- Homogenität im Inneren des Segments
- Homogenitätsbedingung wird bei der Segmentierung ausgewertet
- Globale Zusammenhänge über Multiskalenstrategie



Region Merging

- Initial wird jedes Pixel zu einem Segment erklärt
- Zwei benachbarte Regionen werden zusammengefasst, wenn sie auch gemeinsam das Homogenitätskriterium erfüllen
- Segmentierung ist beendet, wenn keine zwei Regionen mehr zusammengefasst werden können
- Zwischenergebnisse werden in einem Region Adjacency Graph (RAG) gespeichert



Region Merging

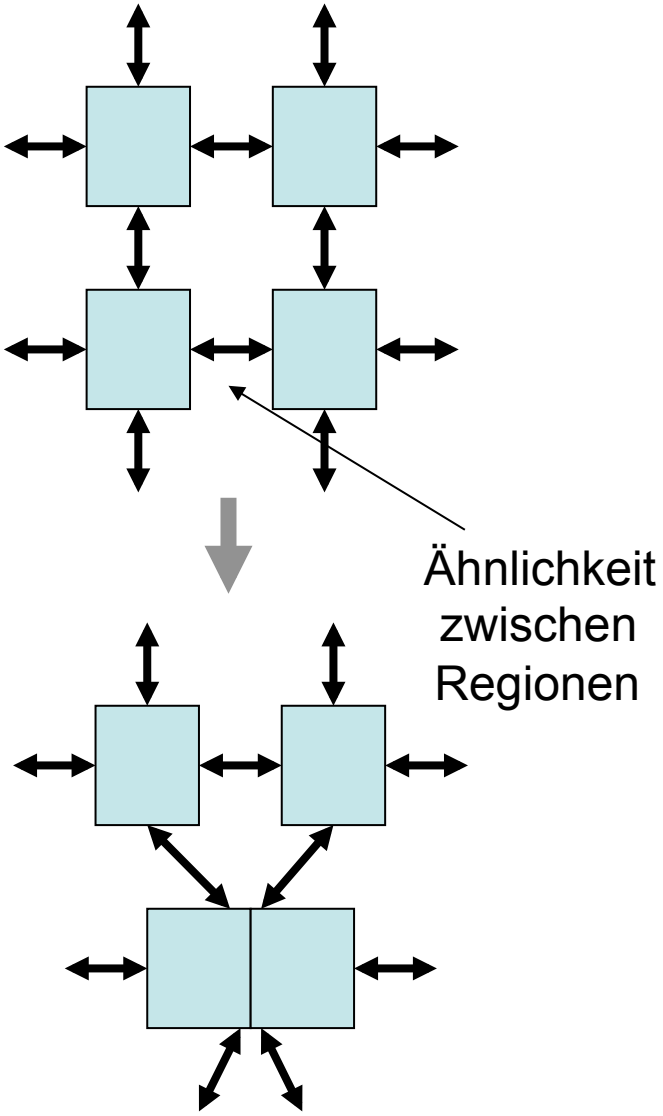
„von Pixeln zu Regionen“:

```
stopMerge = false
while not stopMerge do
  (r1,r2) = MaxSimilarity(region)
  if sim(r1,r2) > T then
    region.merge(r1,r2)
  else
    stopMerge=true
```

Bsp. f. Ähnlichkeitskriterium:

maximaler Grauwertunterschied zwischen Pixeln von r1 und r2

Region Labeling kann in den Prozess integriert werden



Region Merging und Multiskalenstrategie

Modellannahme:

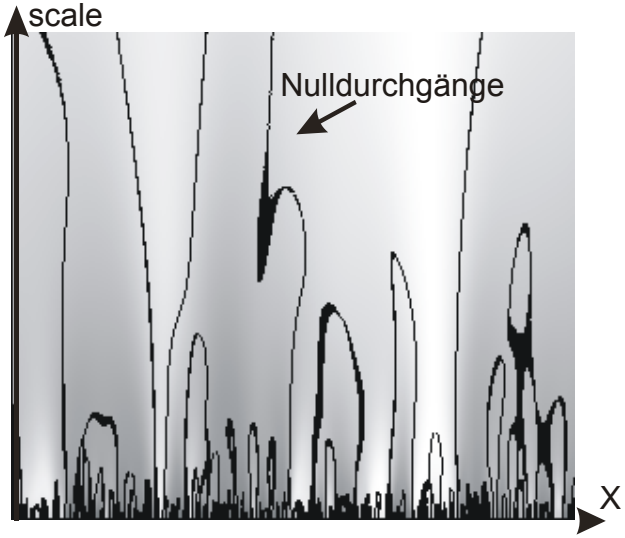
Die gröbste Skalierungsstufe, auf der sich segmentierungs-relevante Eigenschaften zeigen, ist bekannt

Prozess:

- Region Merging auf grober Skalierung
- Übertragung des Resultats auf die nächstfeinere Stufe
- Alle Pixel, die zu Pixeln eines anderen Segments benachbart sind, werden nochmals geprüft
- Verfahren endet, wenn die feinste Skalierungsstufe erreicht ist

Multiskalenstrategie

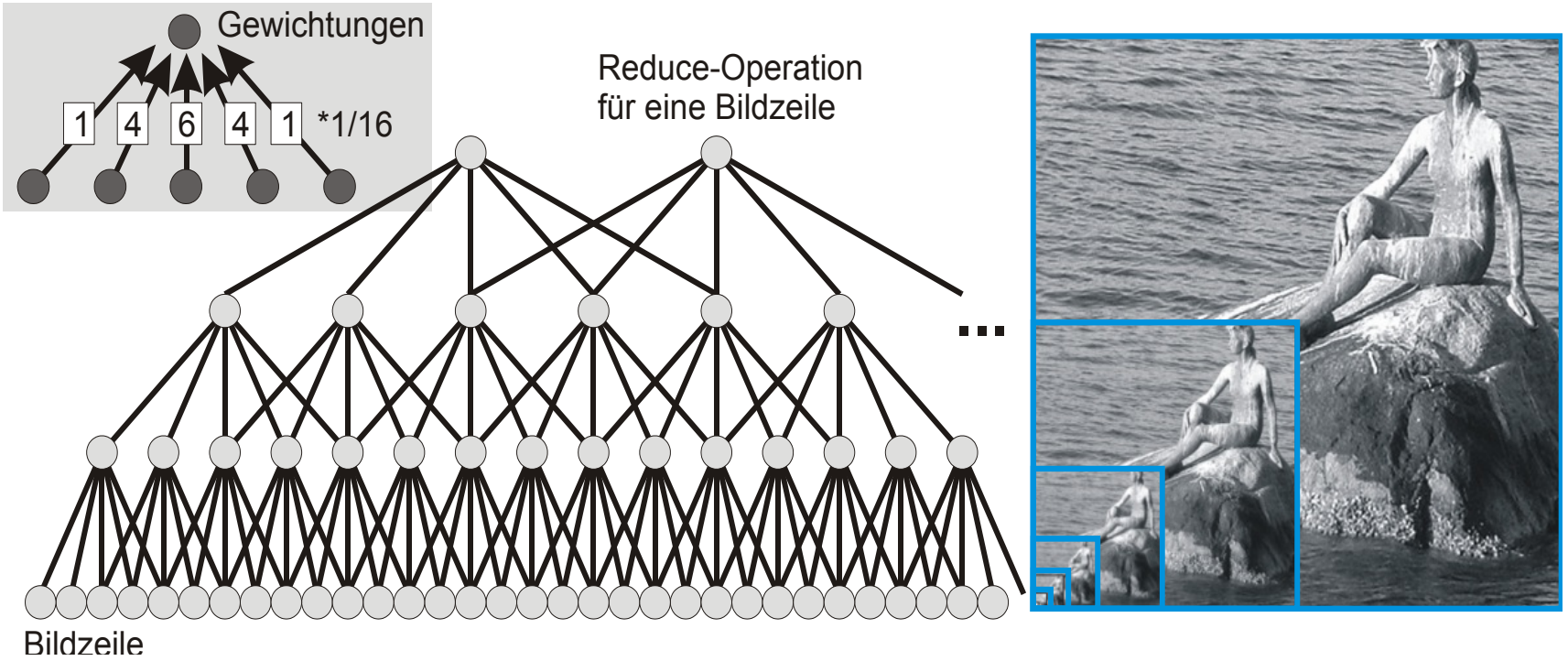
- Relative Kriterien für Homogenität können über unterschiedliche Entfernungen verschieden wirken
- Segmentierung nach Multiskalenstrategie wertet Kriterien auf unterschiedlichen Skalierungen aus



Gaußpyramide

- Das Originalbild wird fortlaufend durch eine reduce-Operation skaliert
- Jedes Pixel der nächsthöheren Skalierungsstufe repräsentiert 5 Pixel der aktuellen Stufe
- Vor der Reduktion wird der Frequenzumfang durch Filterung vermindert:
  - Gaußfilter $\frac{1}{16}\begin{pmatrix}0.87 & 3.91 & 6.44 & 3.91 & 0.87\end{pmatrix}$
  - Binomialfilter $\frac{1}{16}\begin{pmatrix}1 & 4 & 6 & 4 & 1\end{pmatrix}$

Gaußpyramide



Expand-Operation

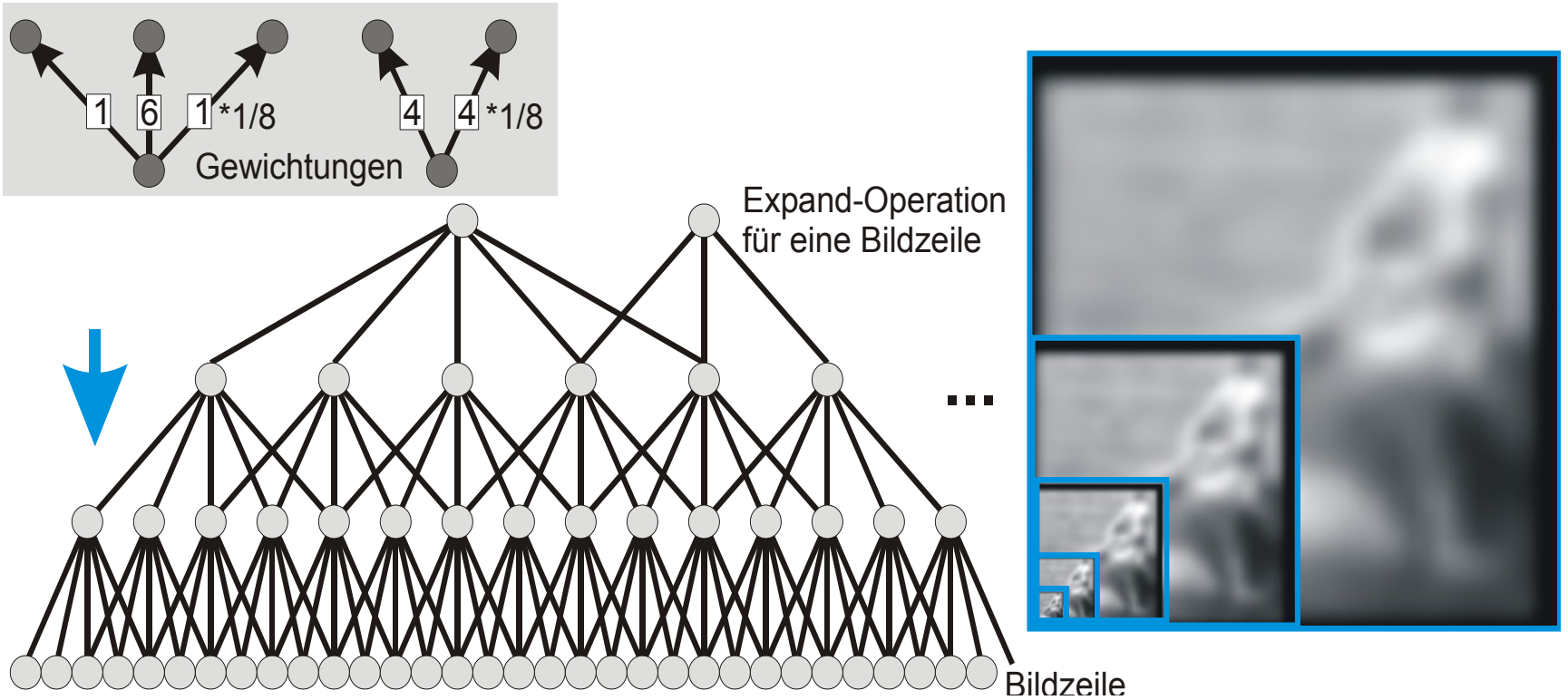
- Um die vorherige Skalierungsstufe zu erzeugen, wird eine expand-Operation definiert
- Pixel der neuen Skalierungsstufe werden durch Interpolation erzeugt
  - Pixelorte, die auf beiden Skalierungsstufen existieren:

$$\frac{1}{8.18}\begin{pmatrix}0.87 & 6.44 & 0.87\end{pmatrix} \text{ bzw. } \frac{1}{8}\begin{pmatrix}1 & 6 & 1\end{pmatrix}$$

  - Pixelorte, die nur auf der vorherigen Skalierungsstufe existieren:

$$\frac{1}{7.82}\begin{pmatrix}3.91 & 3.91\end{pmatrix} \text{ bzw. } \frac{1}{8}\begin{pmatrix}4 & 4\end{pmatrix}$$

Expand-Operation

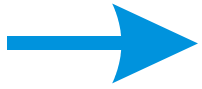


Die Expand-Operation ist nicht verlustfrei

Laplace-Pyramide

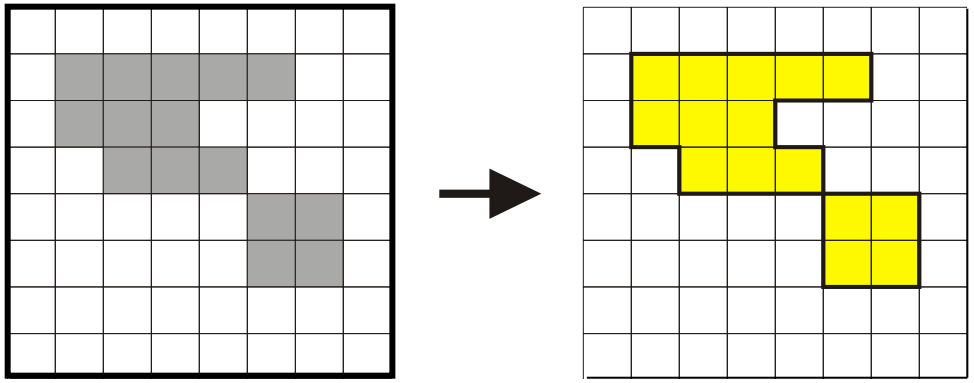
- Jede Skalierungsstufe s enthält nur den Unterschied f_s - expand(reduce(f_s))
- redundanzfreie Repräsentation

expand(reduce(image))-image



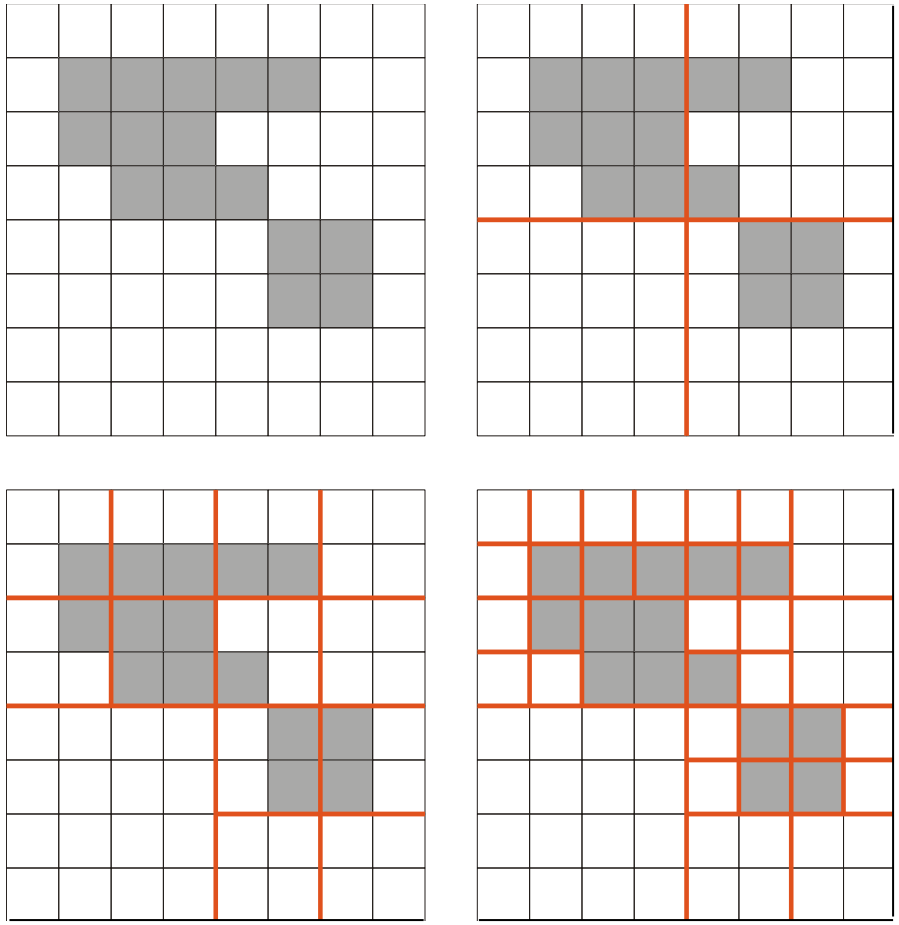
Split & Merge-Algorithmus

Regionenbasiertes Verfahren



- Startbedingung: Das gesamte Bild ist ein Segment
- Jedes Segment wird solange in 4 Untersegmente zerlegt, wie es ein gegebenes Homogenitätskriterium nicht erfüllt
- Benachbarte Segmente werden zusammenfasst, wenn sie auch zusammengenommen homogen sind
- Resultat ist eine vollständige, überdeckungsfreie Zerlegung des Bildes (Segmentierung gemäß Definition)

Zerlegungsschritt

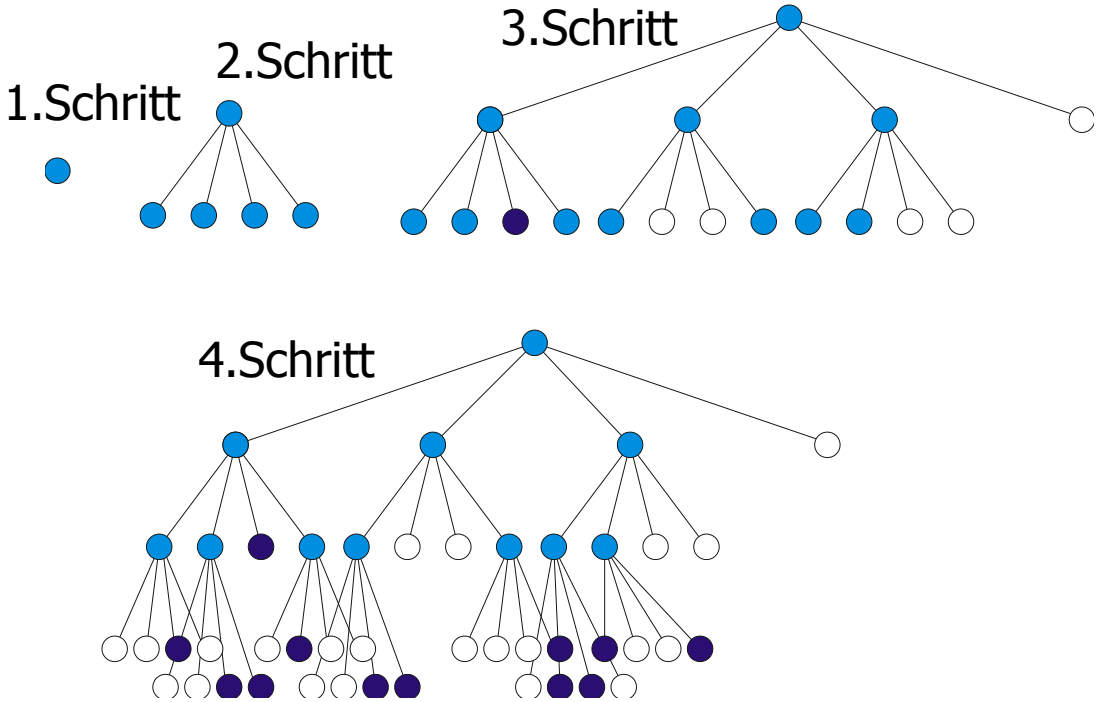


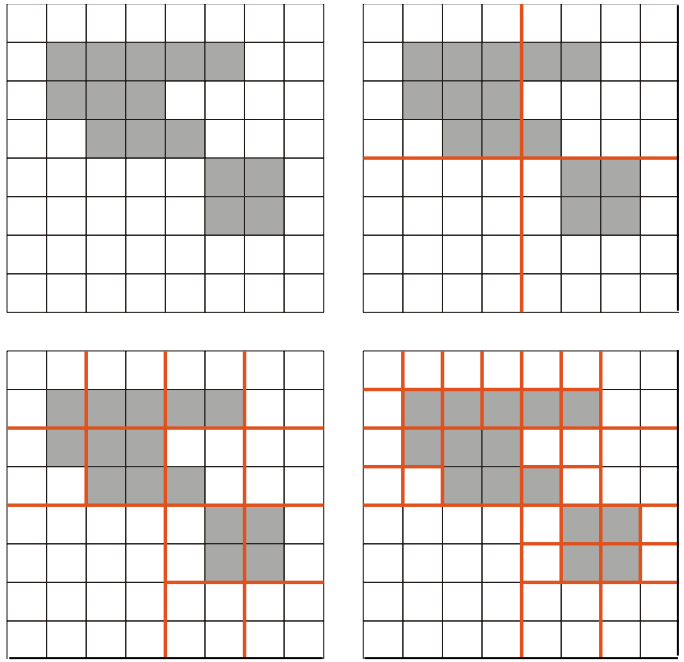
Zerlegung terminiert spätestens auf Pixelebene

Problem: Datenstruktur zur Dokumentation der aktuellen Zerlegung
→ Quad-Tree

Zerlegungsschritt (Quad-Tree Repräsentation)

Wert des Homogenitätsmerkmals einer Region wird im entsprechenden Blatt des Quad-Tree abgelegt

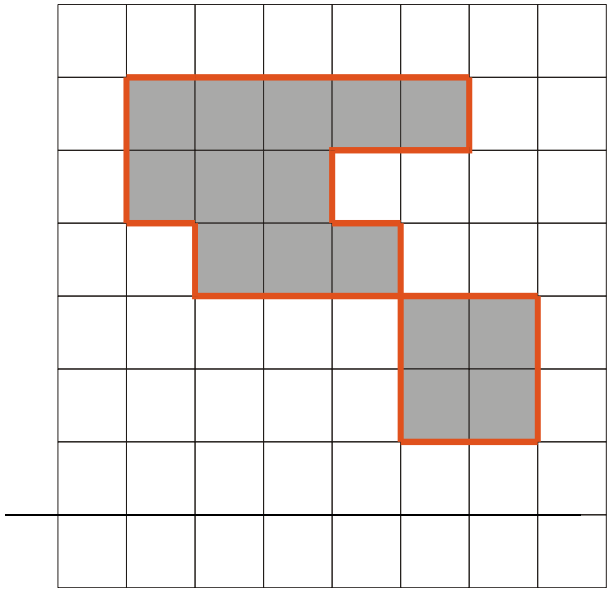


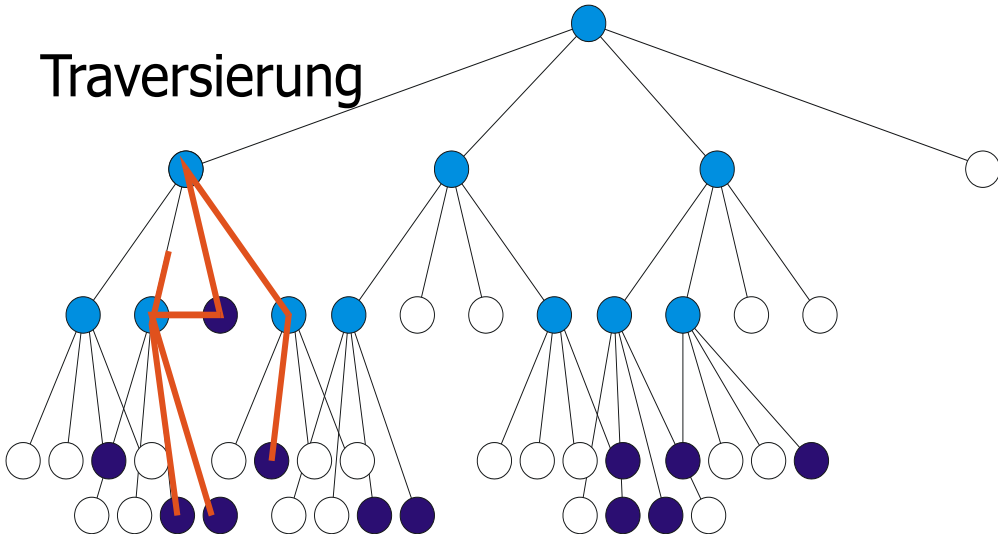


Merging

Quadtree wird traversiert und in einen RAG überführt

Auf dem RAG wird ein Region Merging durchgeführt



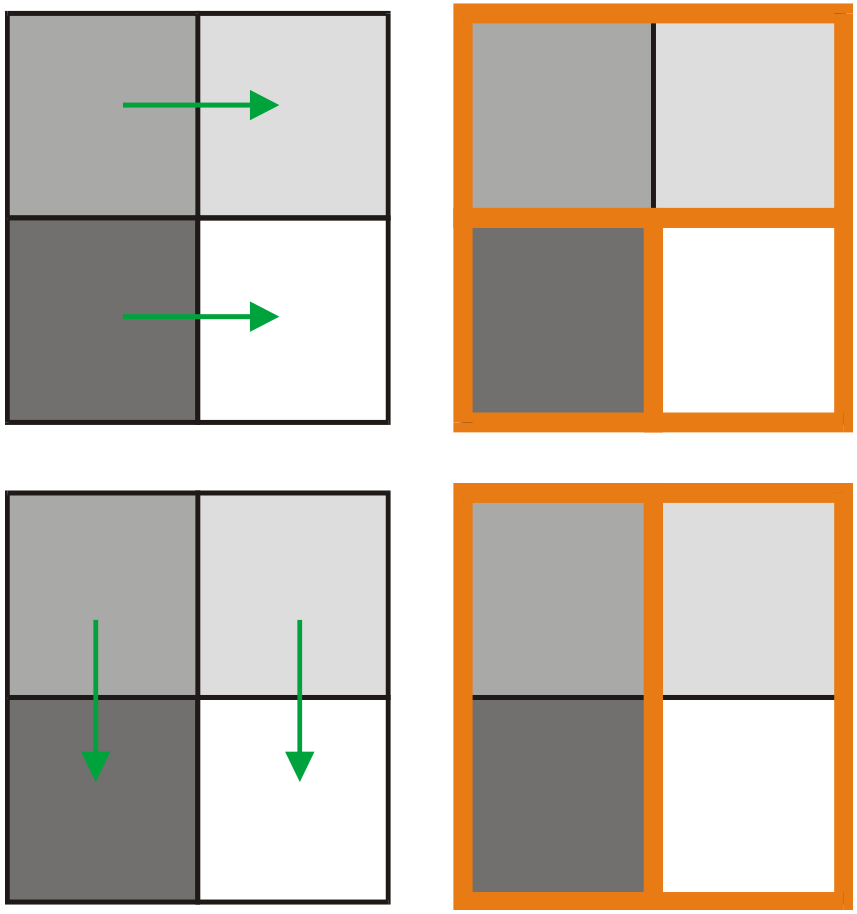


Split & Merge

Resultat: Zerlegung des gesamten Bilds in Regionen

Multiskalenstrategie ist integriert

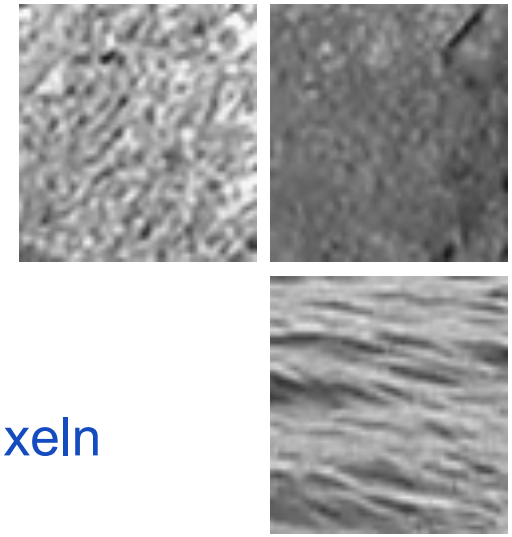
Homogenitätsmerkmale wie bei Region Merging



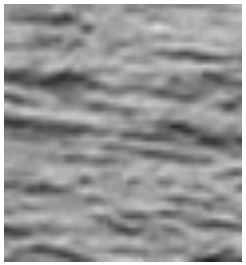
Probleme (Region Merging und Split & Merge):

- Merge-Schritt ist bei relativem Homogenitätsmerkmal nicht immer eindeutig
- Minimale Segmentzahl wird nicht immer gefunden

Textur als Homogenitätsmerkmal



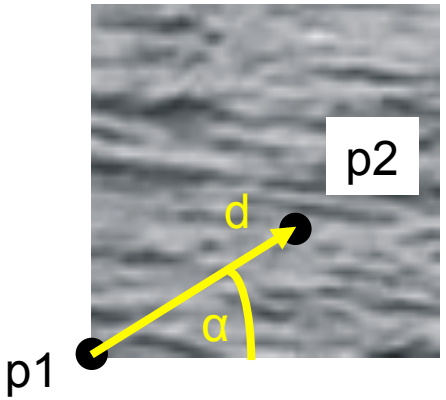
- Textur – Musterung der Oberfläche
  - es existiert keine Definition von Textur
  - es gibt eine große Anzahl von Texturmaßen
  - Textur ist inhärent skalenabhängig
  - Textur ist eine Eigenschaft einer Gruppe von Pixeln
- Texturmaße
  - strukturell (Zusammensetzung aus Texturelementen – texeln)
  - stochastisch (eine charakterisierbare Grauwertverteilung)
  - spektral (charakteristische Frequenzattribute)

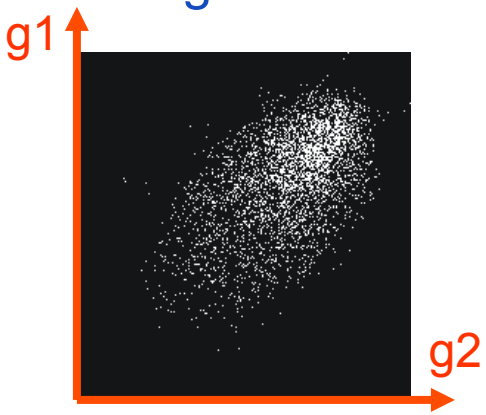


Haralick'sche Texturmaße

Robert Haralick

- **Co-Occurrence-Matrix** = 2D-Histogramm für Pixelpaare
  - Pixel p1 und p2 sind ein Paar, wenn Abstand *d* haben und auf Linie mit Winkel α zur *x*-Achse liegen
- Repräsentiert Korrelation zwischen Pixeln
  - Wahrscheinlichkeit, dass p1 und p2 Grauwerte g1 und g2 haben
  - Meist sind Pixel nicht über große Entfernungen korreliert, daher *d*=1 oder *d*=2 üblich
  - Für Korrelation über größere Entfernung → Multiskalenstrategie





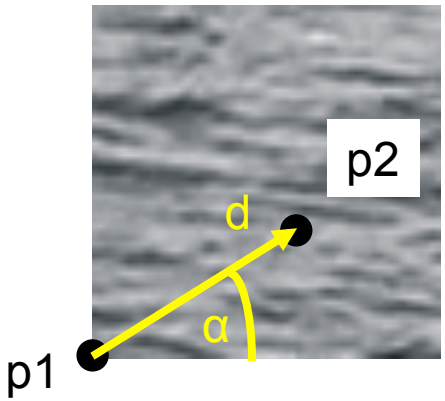
Haralick'sche Texturmaße

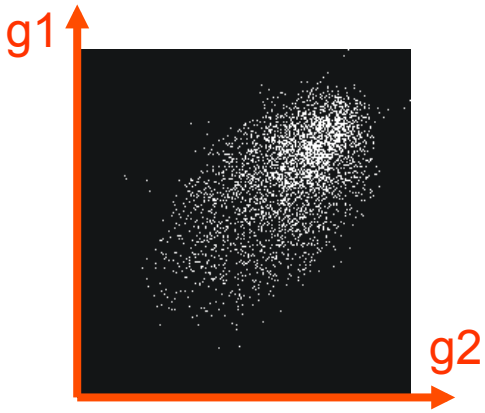
Robert Haralick

- Co-Occurrence-Matrix für Region R, |R| = K

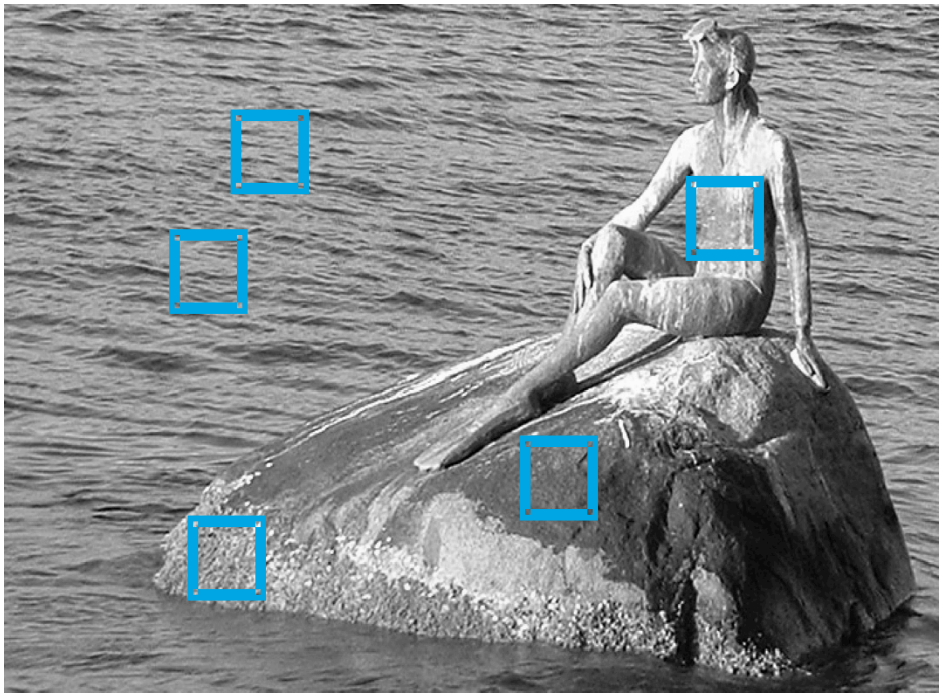
$$P_{\alpha,\Delta}(g_1,g_2) = \frac{1}{|R|}\sum_{\vec{x}\in R}\delta_D\left(f(\vec{x})-g_1\right)\cdot\delta_D\left(f(\vec{x}+\vec{d})-g_2\right)$$

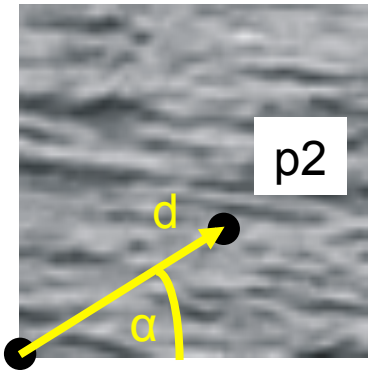
$$\vec{d} = \Delta\cdot\begin{pmatrix}\cos(\alpha)\\ \sin(\alpha)\end{pmatrix}$$

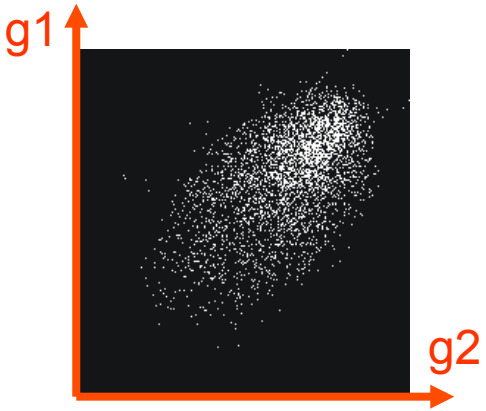


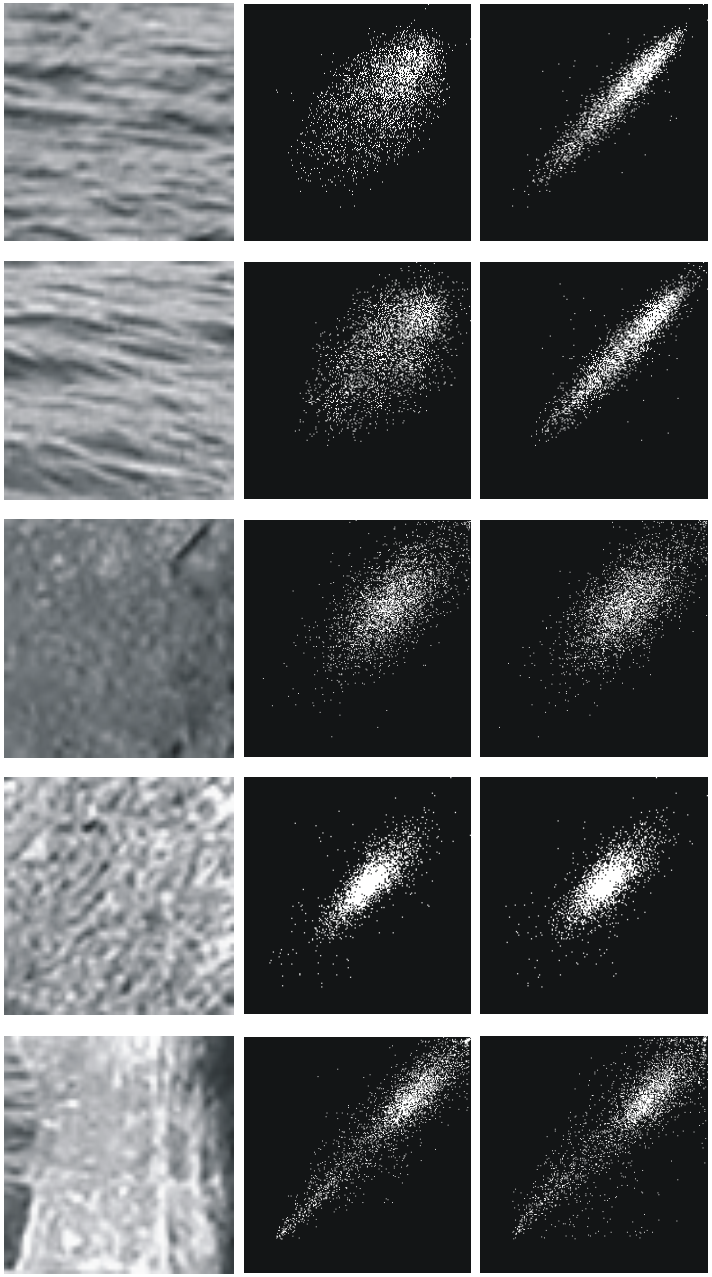


Co-Occurrence-Matrix









d=1, α=90°, α=0°

Haralick'sche Texturmaße

zunächst $P_{\Delta,\alpha}$ normieren: $P_{\Delta,\alpha} := \frac{1}{s} P_{\Delta,\alpha}$ mit $s = \sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2)$

Energie / Uniformität $\sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} P_{\Delta,\alpha}^2(g_1, g_2)$

Kontrast $\sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} (g_1 - g_2)^2 \cdot P_{\Delta,\alpha}(g_1, g_2)$

Entropie $-\sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2) \cdot \log_2 \left[P_{\Delta,\alpha}(g_1, g_2) \right]$

Homogenität / inverse Differenz $\sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} \frac{P_{\Delta,\alpha}(g_1, g_2)}{1 + |g_1 - g_2|}$

- liefern aussagekräftige Kennwerte für Texturen
- zur Segmentierung
  - Berechnung für Δ = 1 und α = 0°, 45°, 90°, 135°
  - Merkmalsvektor aus Texturmaßen
  - empfohlene Merkmale zur Texturklassifikation: Entropie, Kontrast, Korrelation

Haralick'sche Texturmaße (weitere)

zunächst $P_{\Delta,\alpha}$ normieren: $P_{\Delta,\alpha} := \frac{1}{s} P_{\Delta,\alpha}$ mit $s = \sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2)$

Korrelation $$\sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} \frac{(g_1 - \mu_1) \cdot (g_2 - \mu_2) \cdot P_{\Delta,\alpha}(g_1, g_2)}{\sigma_1 \sigma_2}$$

mit

$$\mu_1 = \sum_{g_1=0}^{K-1} g_1 \sum_{g_2=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2) \qquad \sigma_1 = \sqrt{\sum_{g_1=0}^{K-1} (g_1 - \mu_1)^2 \sum_{g_2=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2)}$$

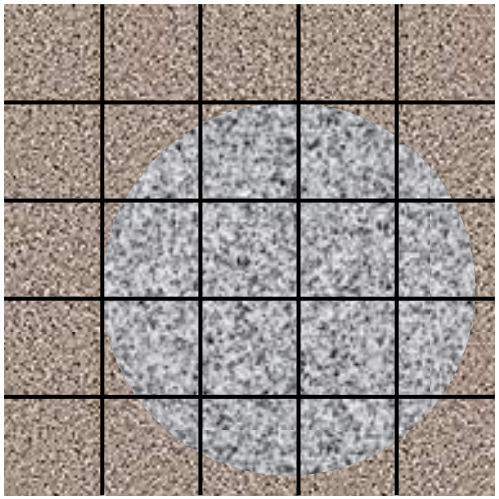
$$\mu_2 = \sum_{g_2=0}^{K-1} g_2 \sum_{g_1=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2) \qquad \sigma_2 = \sqrt{\sum_{g_2=0}^{K-1} (g_2 - \mu_2)^2 \sum_{g_1=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2)}$$

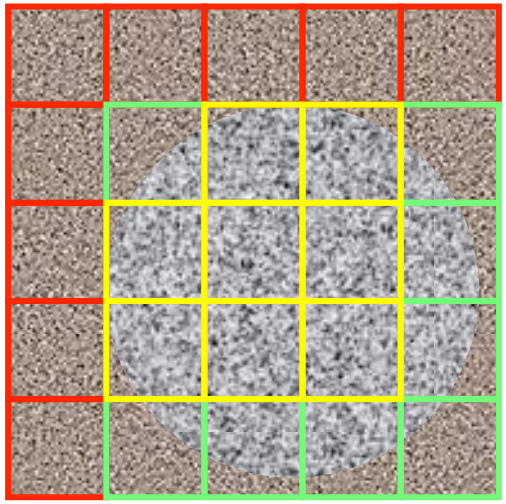
inverse difference moment $$\sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} \frac{P_{\Delta,\alpha}(g_1, g_2)}{1 + (g_1 - g_2)^2}$$

Unähnlichkeit $$\sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2) \cdot |g_1 - g_2|$$

Besonderheiten Texturmerkmal

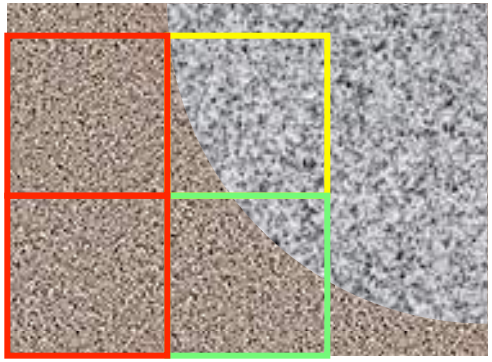
- Berechnung der Texturmerkmale auf der Basis willkürlicher Regionen
- Segmentierung
- Berechnung der Zuverlässigkeit
- Erneute Segmentierung

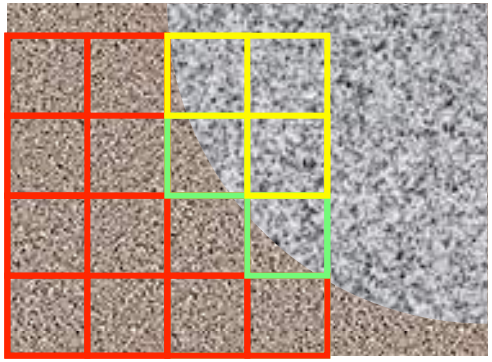




Zuverlässigkeit Texturmerkmal

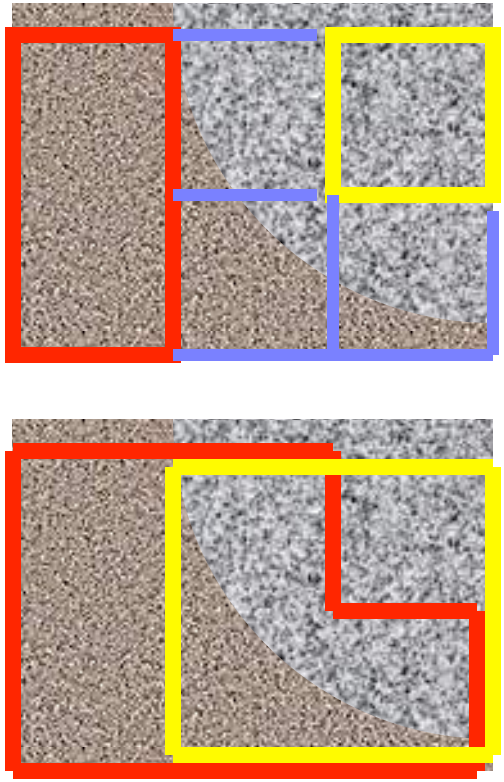
- Persistenz:
- Falls eine Region in Teile zerlegt wird und das Merkmal in den Teilregionen berechnet werden kann, dann sollte das Texturmerkmal in der Teilregion dieselben Werte annehmen, wie in der Ursprungsregion
- Achtung: In der Regel ist das berechnete Maß eine Schätzung, deren Güte von der Größe der Region abhängt





Strategie: Erneute Segmentierung

- Texturmerkmale der Gesamtsegmente erneut berechnen
- Segmente mit hoher Unzuverlässigkeit
  - allen benachbarten zuverlässigen Segmenten zuordnen



Die beiden Extremfälle

Naive Textursegmentierung

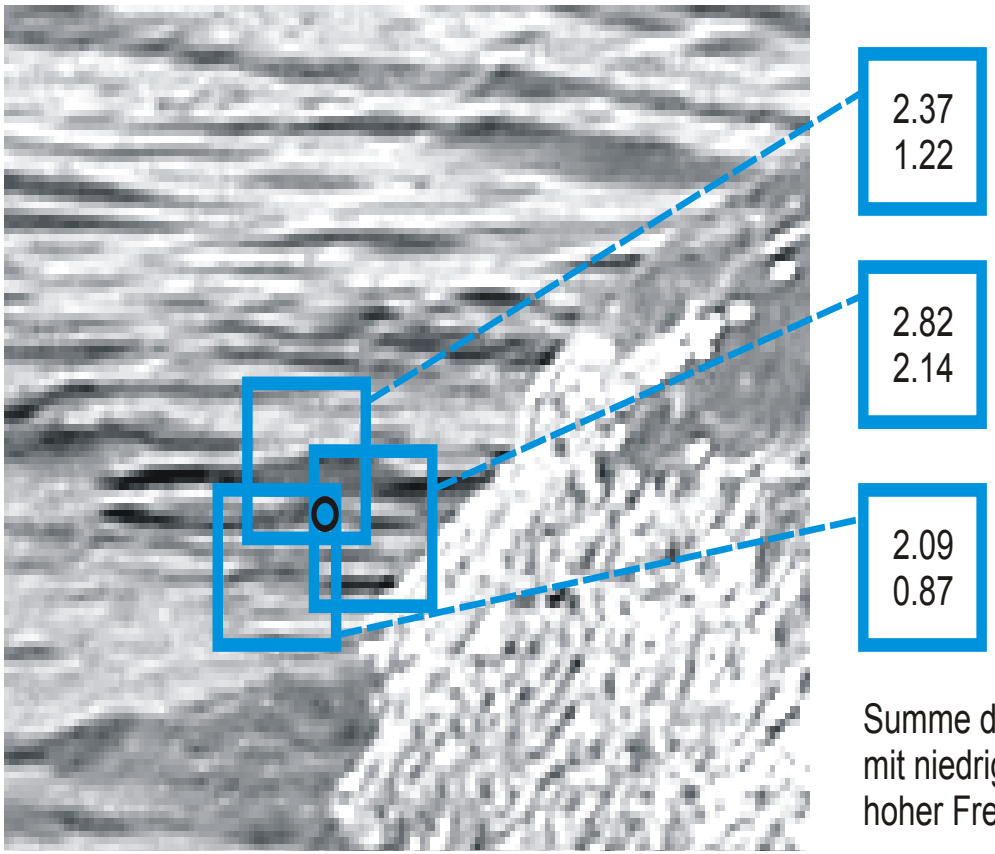
Modell:

- Skalierungsstufe der Textur, Blockgröße, sowie geeignetes Texturmaß seien bekannt

Algorithmenskizze:

- Berechne für überlappende Blöcke das Texturmaß
- Führe eine homogenitätsbasierte Segmentierung auf den Texturmaßen durch (z.B. durch Split-and-Merge)
- Ein Pixel erhält von jedem Block, der es überdeckt eine Stimme für ein Label
  - inhomogene Blöcke an Texturgrenzen
- Nachverarbeitung wie bei Histogrammsegmentierung

Textursegmentierung (naiv)



Summe der Amplituden mit niedriger und mit hoher Frequenz

Beispiel:

Jedes Pixel wird von mehreren Blöcken überdeckt

Sind die Texturen unterschiedlich, dann erhalten diese Blöcke unterschiedliche Label

Kantenbasierte Segmentierung

- Edge Linking und Canny Edge Operator
- Nulldurchgänge
- Wasserscheidentransformation

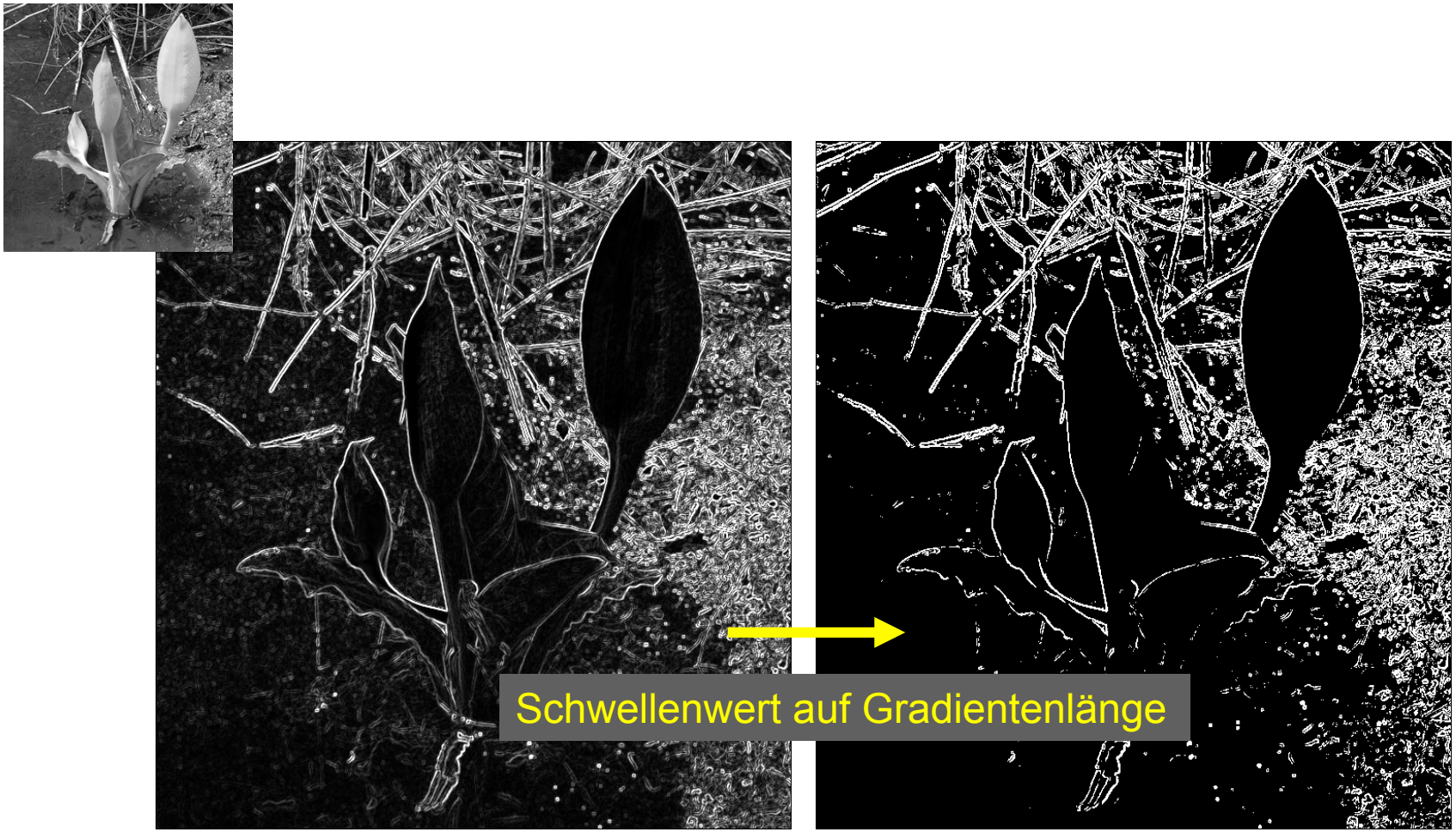
Segmentierung durch Kantenerkennung

Vorteil: Kantenmerkmale sind robuster gegenüber Shading

Einfache Methode:

- Gradientenberechnung
- Kantenpunktdetektion (z.B. Schwelle auf Gradientenlänge)
- Region Labeling basierend auf Kantenpunkten

Segmentierung durch Kantenerkennung



Problem: Kantenpunkte sind nicht Ränder zusammenhängender Gebiete

WASSERSCHEIDEN-TRANSFORMATION

Wasserscheidentransformation

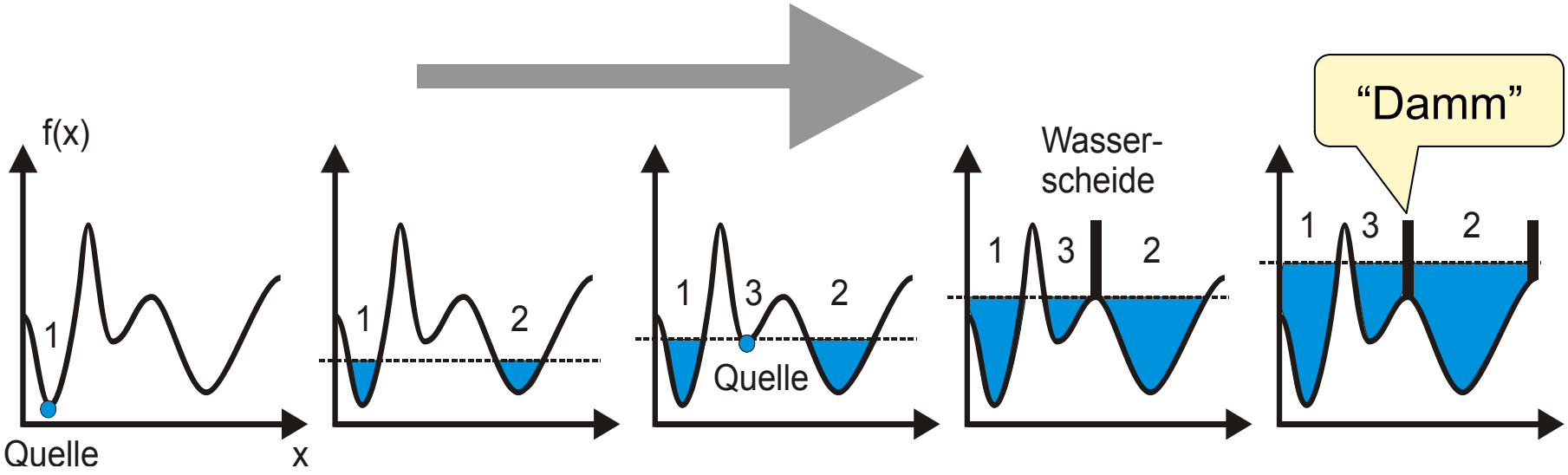
- Wasserscheide: Grenzen der Entwässerung in unterschiedliche Senken
  - Beispiel: Wasserscheide zwischen Nordsee und Mittelmeer entlang des Kamms der Berner Alpen
- Wasserscheide in der Segmentierung: Generiere ein Höhenprofil so, dass die Wasserscheiden die gesuchten Segmentgrenzen sind

Wasserscheiden

- Wasserscheiden sollen an Kanten verlaufen
- Wasserscheiden sind „Gebirgskämme“
- Wasserscheiden sind die Längen der Grauwertgradienten

Wasserscheidentransformation

- **Beregnung:** Es fällt „Regen“ auf jedes Pixel. Gradienten entschieden, wohin der Regen entwässert wird
- **Flutung:** Die „Welt“ wird von den Senken her geflutet. Wenn Wasser aus zwei Senken zusammen fließt, wird ein Damm gebaut bzw. entsteht eine Wasserscheide



Animation von Serge Beucher

- Partitionierung des Bildes durch Flutung von den Minima aus bei Vermeidung des Zusammenfließens verschiedener Quellen
  - zwei Mengen: Niederschlagsgebiete und Wasserscheidenlinien
- Transformation auf Gradientenbild: Niederschlagsgebiete entsprechen homogenen Regionen

Copyright © 2010, Serge Beucher

http://cmm.ensmp.fr/~beucher/wtshed.html

Flutungsalgorithmus (Skizze)

Jedes bei Höhe $h_{aktuell}$ neu überflutete Pixel (m_f,n_f) ist

- **in Isolation**:

  Es nicht zu anderen überfluteten Pixeln der Höhen $h<h_{aktuell}$ benachbart.

  Isolierte Pixel sind Kerne von neuen Segmenten.

- **Erweiterung**:

  Es ist zu anderen überfluteten Pixeln der Höhen $h<h_{aktuell}$ mit gleichem Label benachbart.

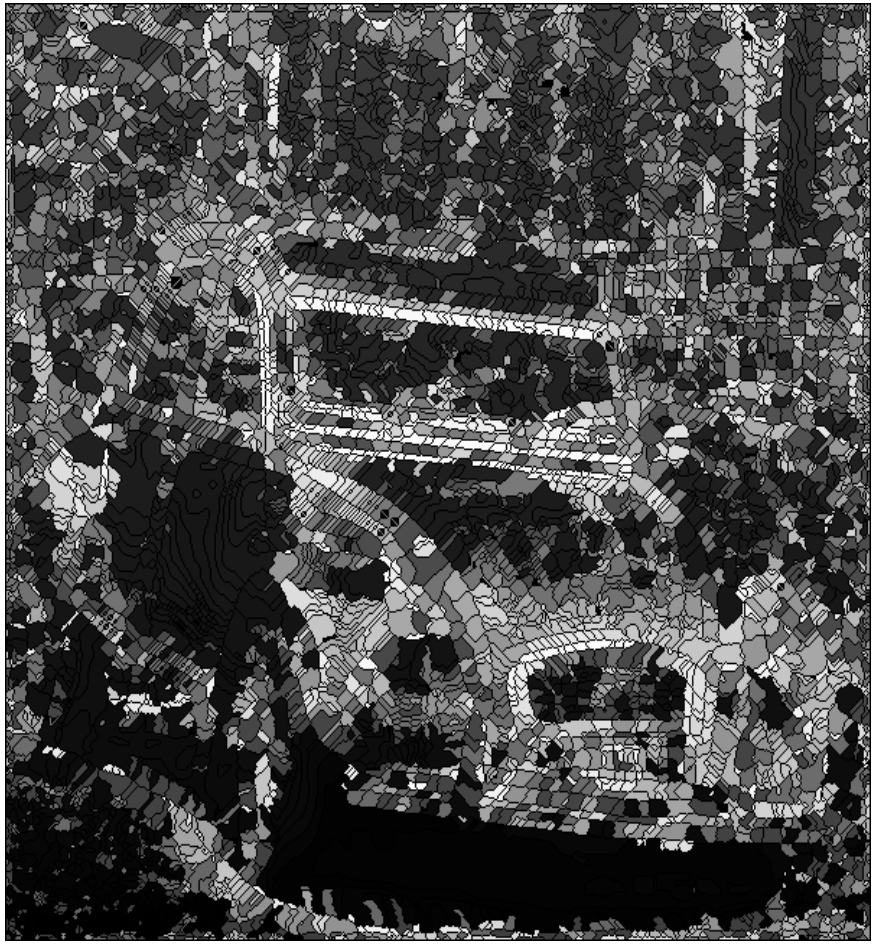
  Das Pixel wird dem Segment mit diesem Label zugeordnet.

- **Wasserscheide**:

  Es ist zu überfluteten Pixeln von mindestens zwei Regionen benachbart.

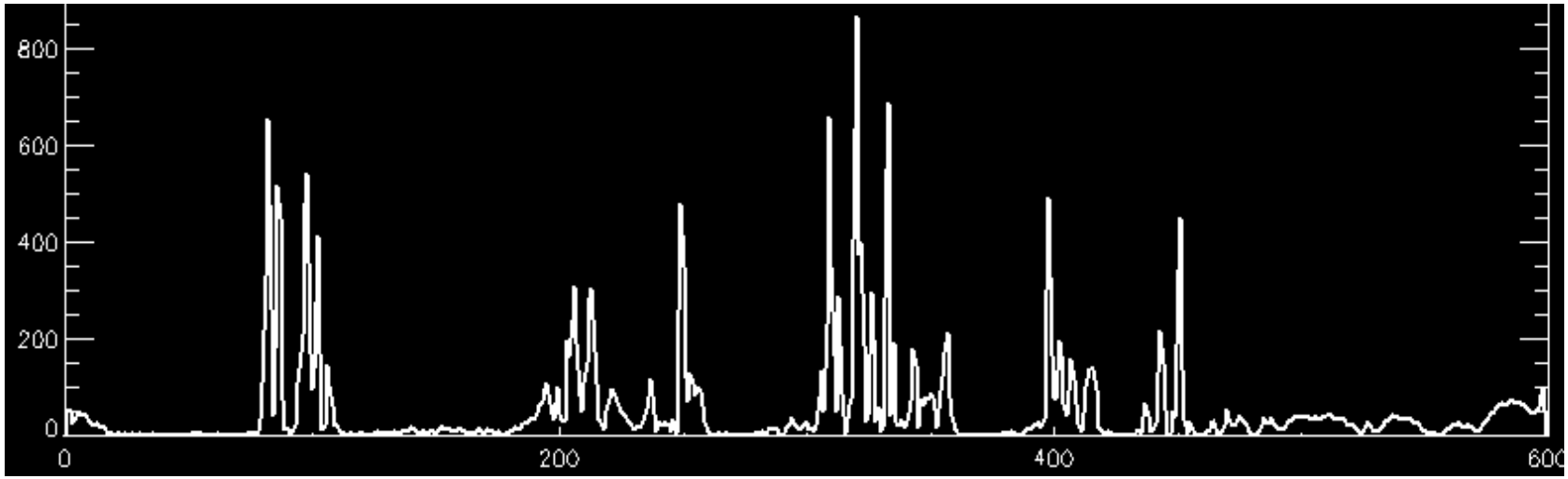
  Dem Pixel wird das Label „Wasserscheide“ zugeordnet.

Resultat der WST



Problem Übersegmentierung

- Für die WST ist jedes lokale Minimum eine Senke
- Die meisten Senken werden durch Rauschen verursacht
- Senken durch Rauschen sind weniger tief als die von Kanten

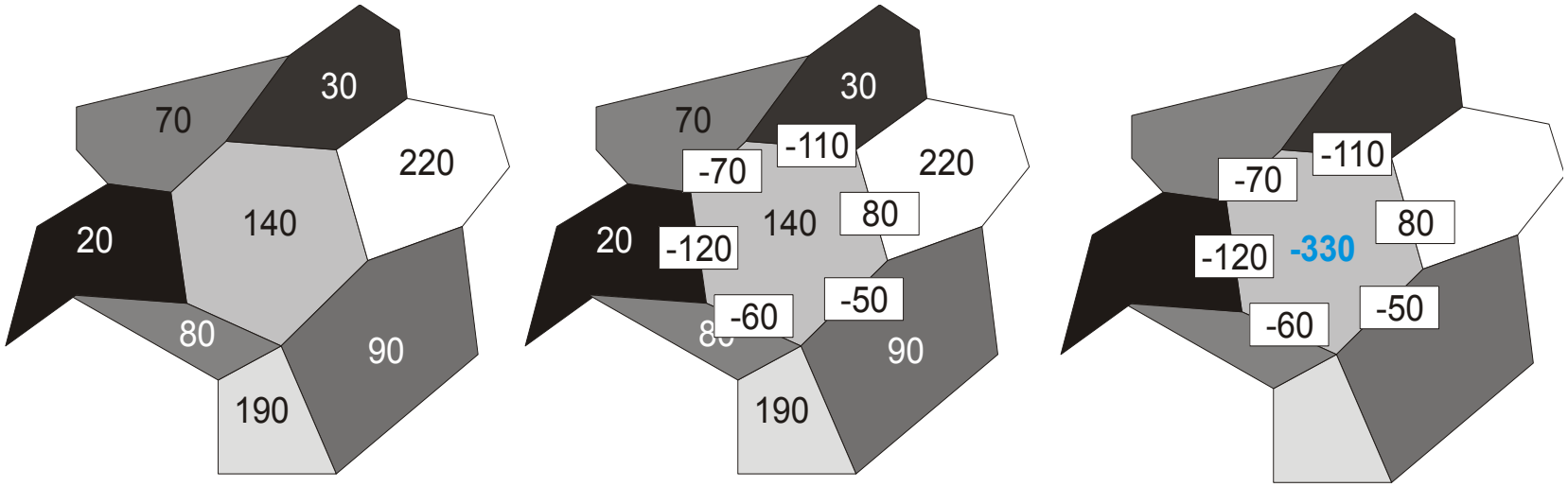


Hierarchische WST

Multiskalenstrategie:

- Wasserscheidentransformation auf dem WST-Resultat
- Jede Region erhält ihren durchschnittlichen Grauwert als Funktionswert
- Die erste WST wird hauptsächlich durch Rauschen verursachte Senken finden
- „Wahre“ Senken sollten über mehrere Stufen der Hierarchie erhalten bleiben

Gradienten für die hierarchische WST



- Differenz zu jeder benachbarten Regionen berechnen
- Gradientenlänge berechnen als durchschnittliche Differenz zu allen Regionen (evtl. mit Fläche gewichten)
- Problem: Kanten werden breiter

Verwendung von Markern bei der WST

- Marker = Region im Bild
  - interne Marker markieren Vordergrundobjekte
  - externe Marker markieren den Hintergrund
- Strategien zur Erzeugung von Markern
  - interaktives Setzen der Marker
  - Bild zur Vorverarbeitung glätten Marker auf geglättetem Bild
  - Textur
- angepasster Algorithmus
  - isoliertes Pixel: Label „unbekannt“
  - Pixel erweitert Region: Label zuordnen
  - Pixel verbindet zwei Regionen: falls eine „unbekannt“ und andere endgültiges Label, dann Label auch für die andere Region

MODELLBASIERTE SEGMENTIERUNG

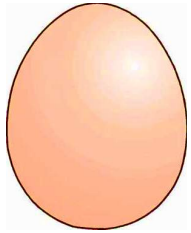
Modellbasierte Segmentierung

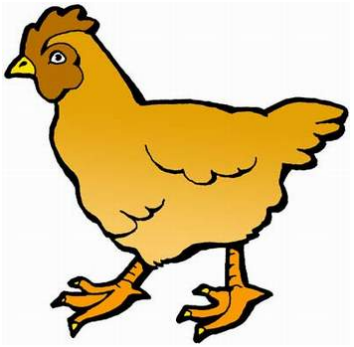
- Vollständige Suche
  - Template Matching
  - Hough Transformation
- Interaktive Suche
  - Region Growing
  - Kantenverfolgung
  - Markerbasierte WST

Modellbasierte Segmentierung

- Segmentierung: Generierung von Symbolen (Bedeutungsträgern) aus Pixeln
- Modell: Erwartete Bedeutung

►Henne-Ei-Problem??





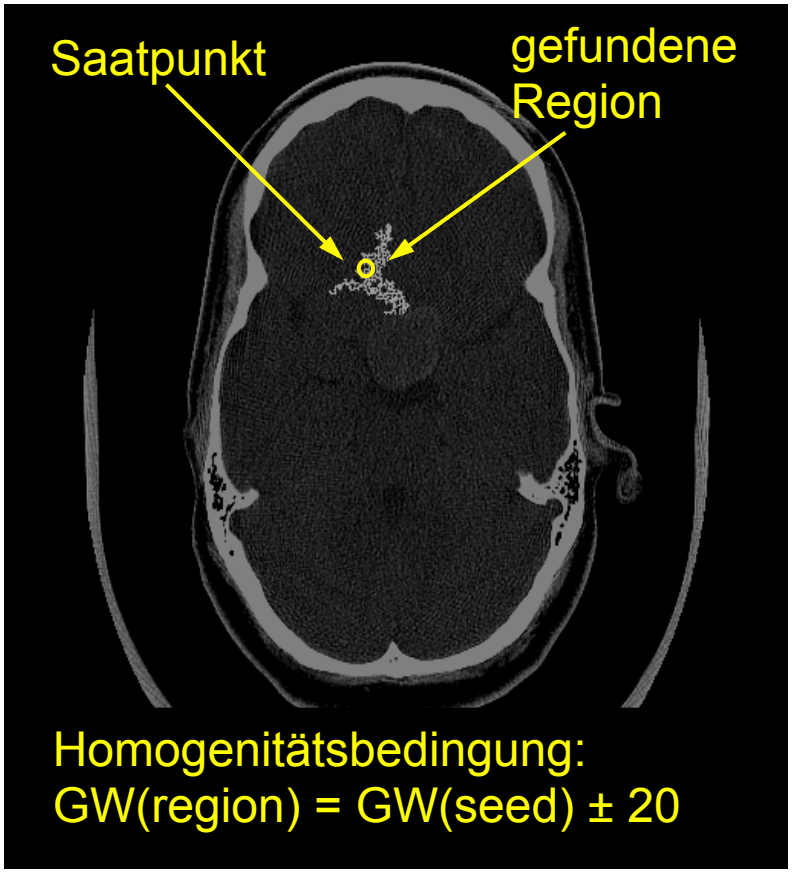
Modellbasierte Segmentierung: Mix aus Segmentierung und Analyse
Mit Vorwissen über Objekt wird nach Instanzen gesucht

- Interaktive Suche: Benutzer gibt Vorwissen interaktiv ein
- Vollständige Suche: Instanzen eines Modells mit wenigen Parametern werden gesucht
- Iterative Suche: Instanzen eines Modells mit vielen Parametern werden gesucht

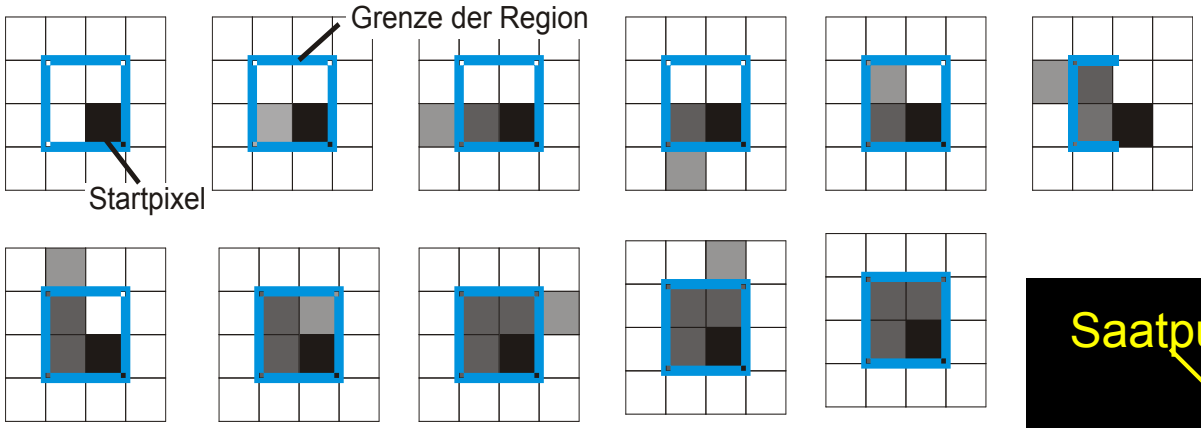
Region Growing

= Flood Fill auf Grauwertbild für (vorgegebenen) Saatpunkt

- Homogenitätsbedingungen
  - Grauwertbereich
  - Grauwertschwankung
- Annahmen und Ziel
  - Gebiet ist intern von größerer Homogenität als an den Grenzen
  - Selektion eines einzigen Gebiets

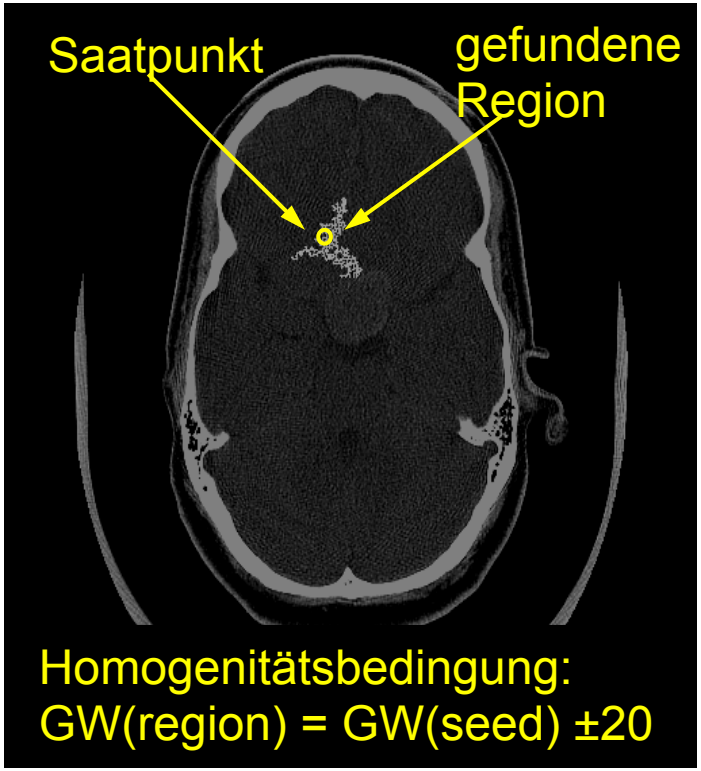


Region Growing

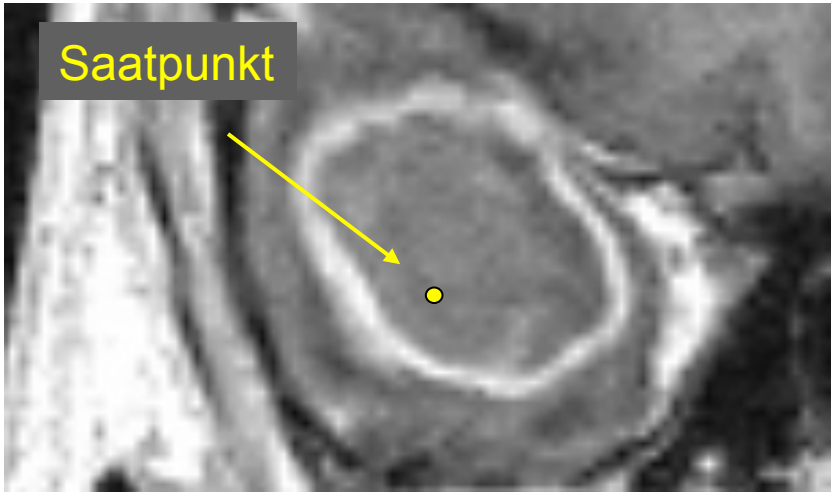


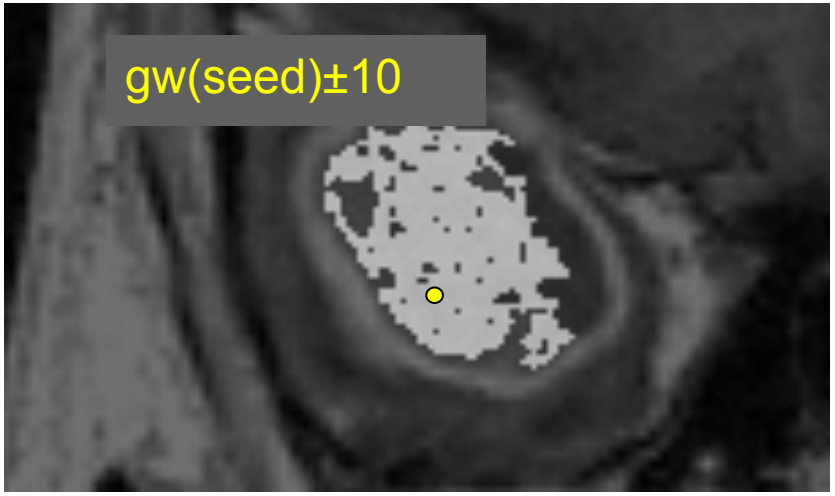
Probleme:

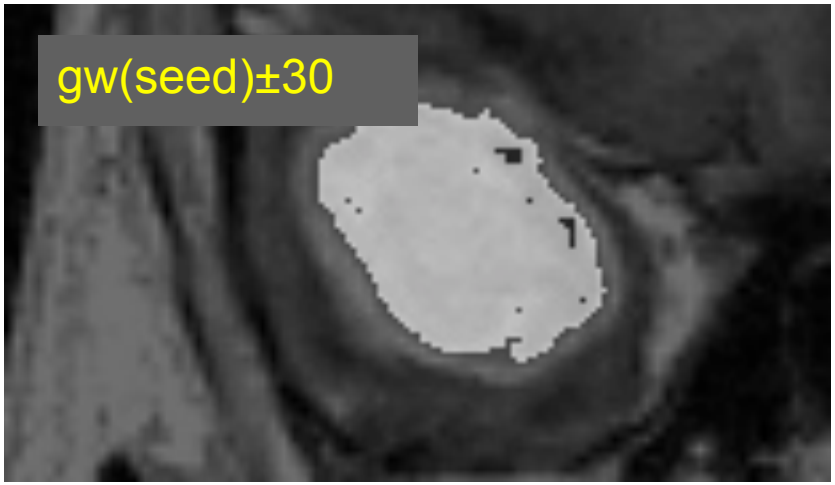
- „Auslaufen“ der Regionen
- zu kleine Regionen
- Rauschanfälligkeit
- Shading im Bild

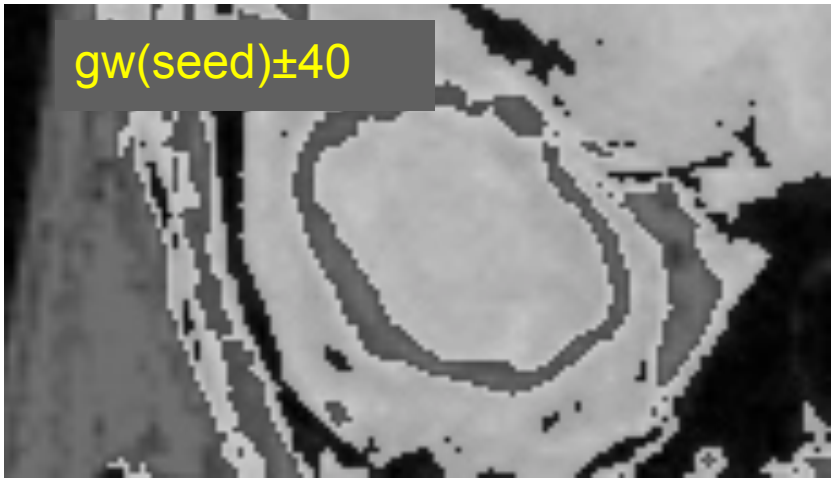


Auslaufen von Regionen





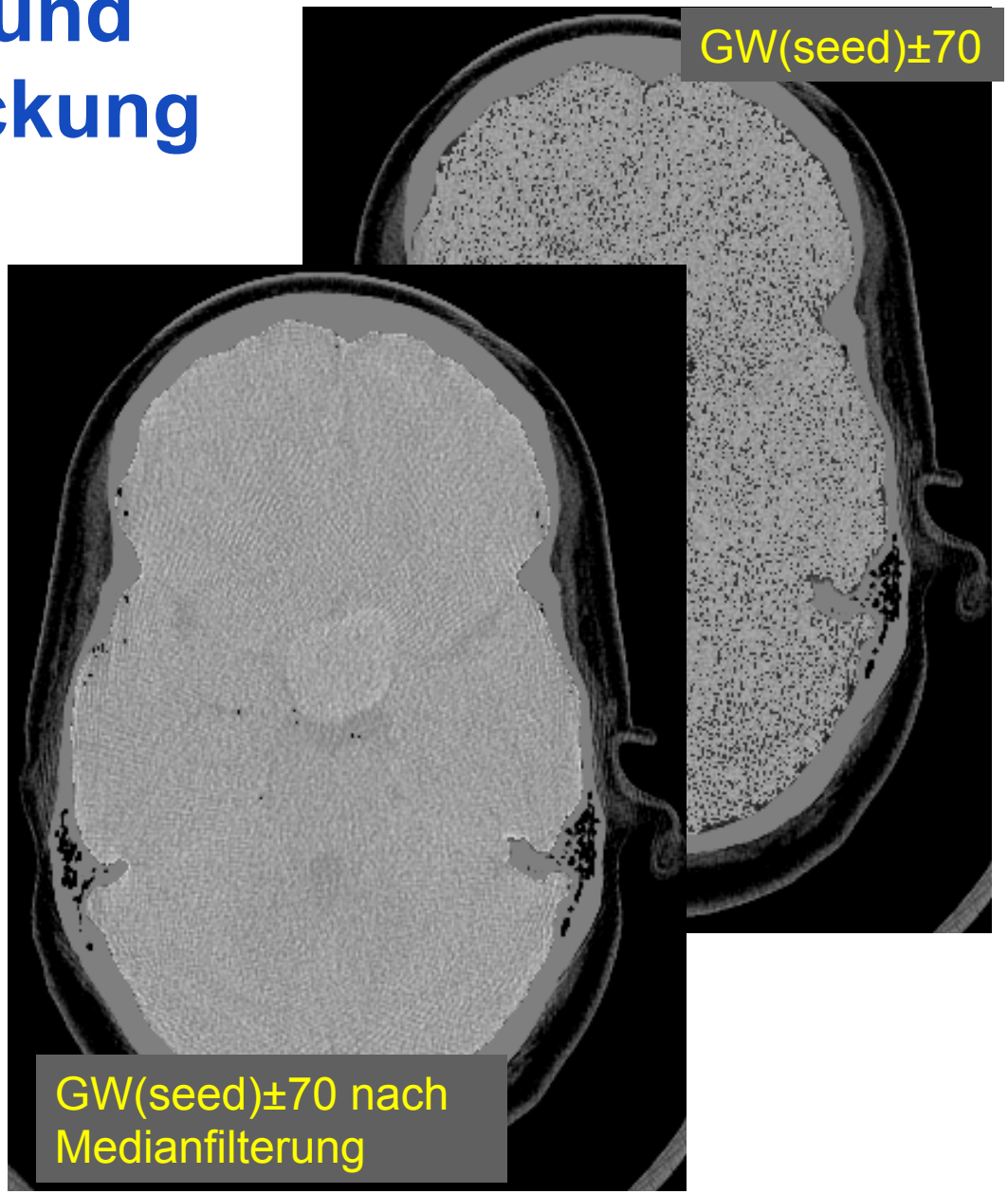




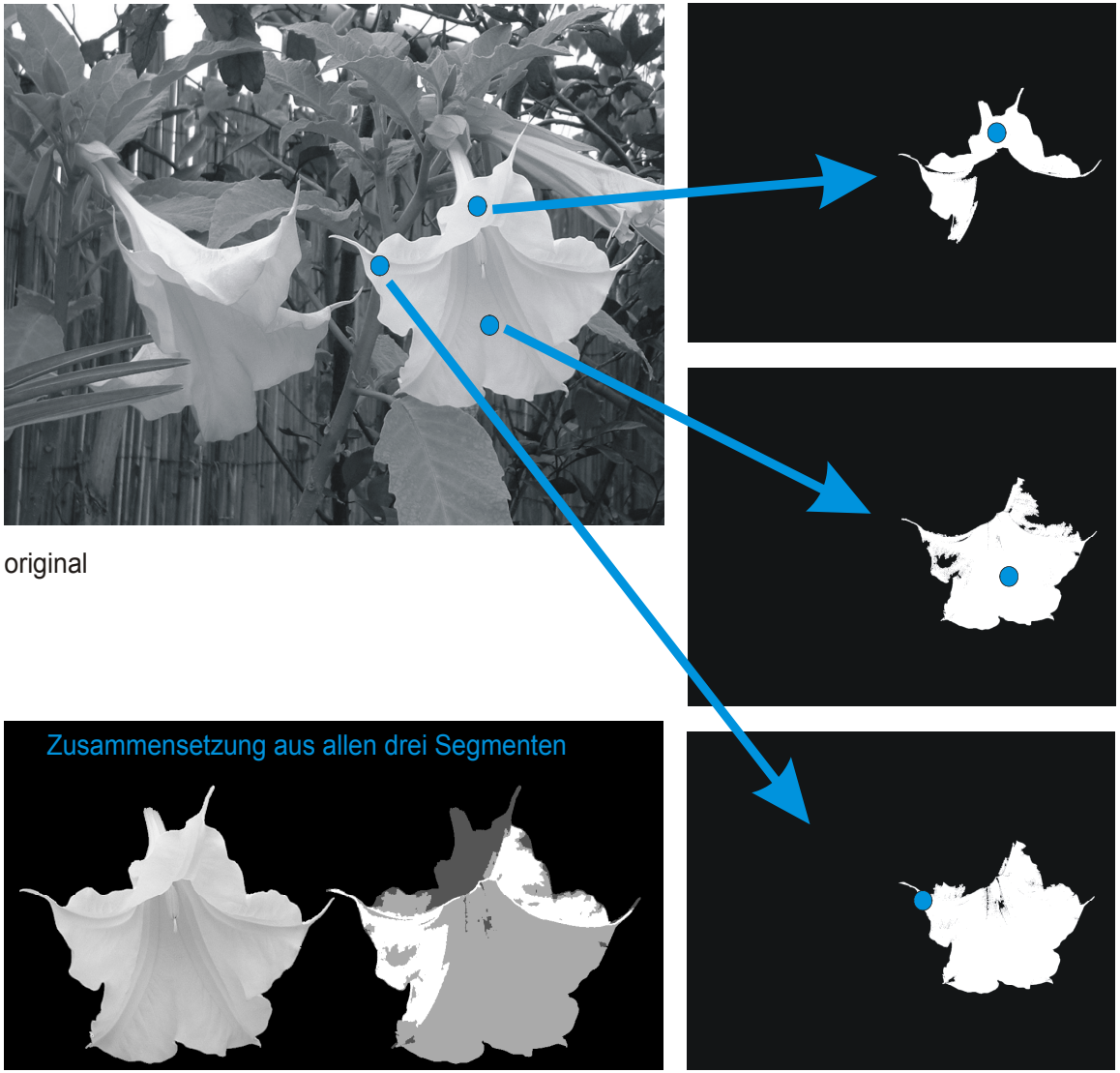
Region Growing und Rauschunterdrückung

Rauschen führt zu Anfälligkeit der Methode

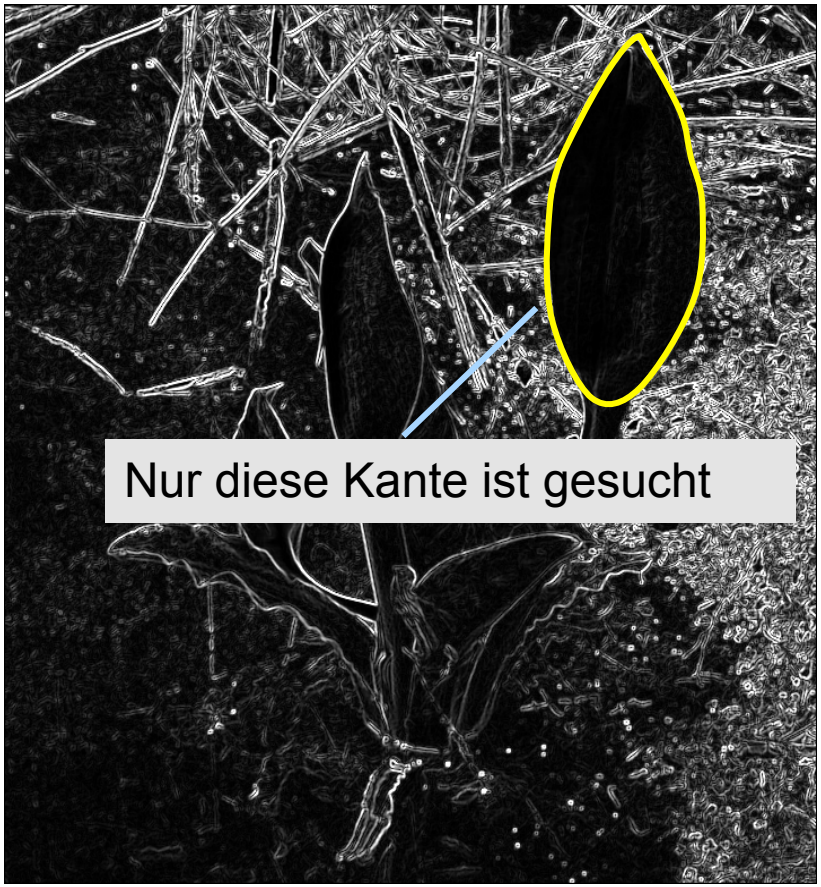
➔ vorherige Rauschunterdrückung oder Nachverarbeitung



Mehrfaches Region Growing



Gezielte Kantensuche



Resultat der Kantenfilter:

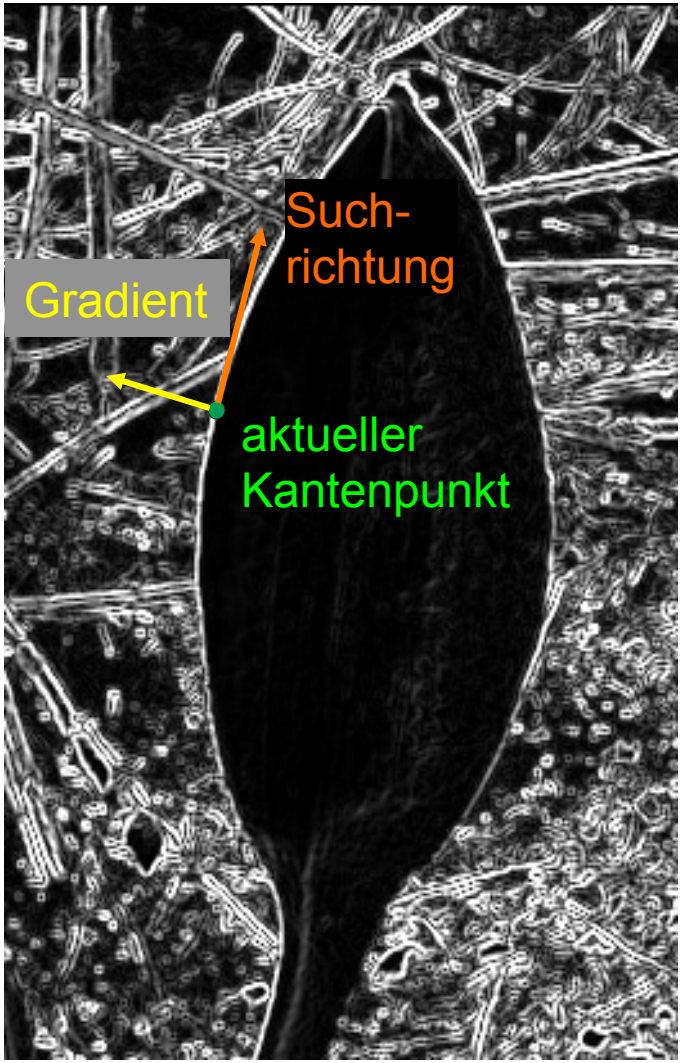
Liste von Kantenpunktkandidaten

- nicht alle Kandidaten gehören zur gesuchten Regionengrenze
- die Regionengrenze kann Lücken aufweisen

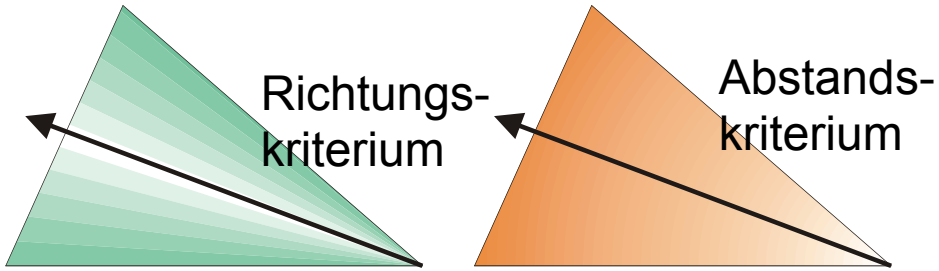
Strategien mit Nutzerinteraktion

- Kantenverfolgung
- Optimale Kantenzüge
- Hough Transformation

Kantenverfolgung



- Anfangspunkt bestimmen
- suche senkrecht zur aktuellen Gradientenrichtung
- Kantenpunkt ist gefunden, wenn
  - in hinreichend kurzer Entfernung,
  - mit hinreichend geringer Richtungsabweichung
  - ein hinreichend großer Gradient existiert



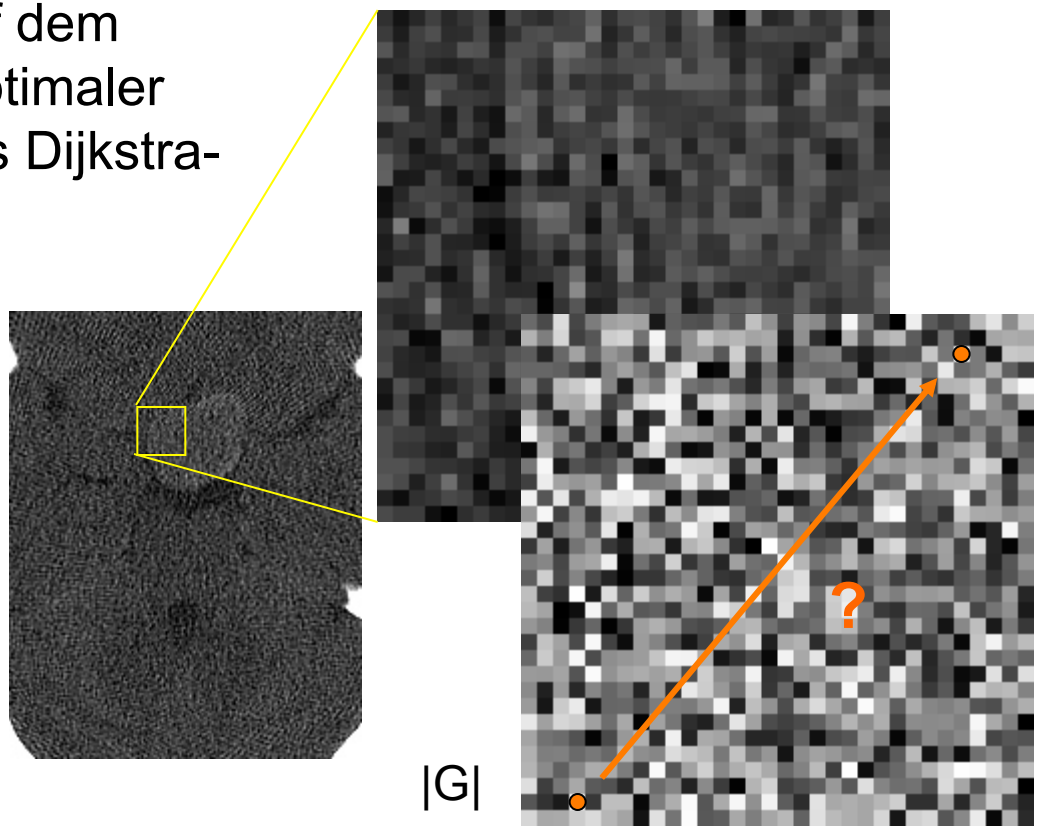
Optimale Kantenzüge (Graphensuche)

Das Resultat des Gradientenoperators wird als Graph aufgefasst, auf dem zwischen zwei Punkten ein optimaler Pfad gesucht wird (z.B. mittels Dijkstra-Algorithmus).

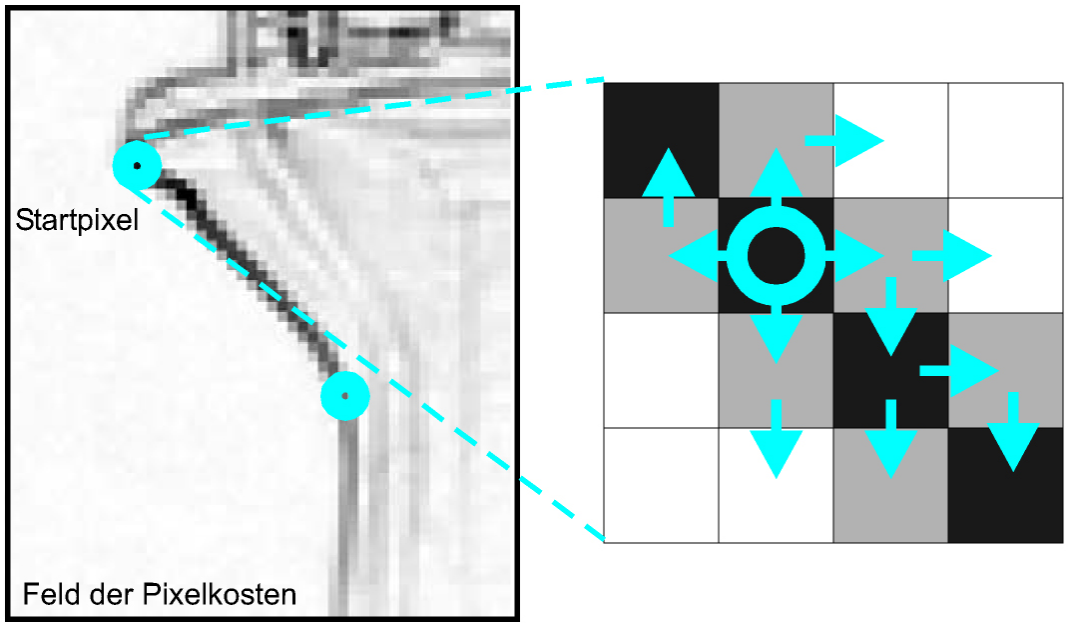
Vorteil:

Globale Aspekte des Kantenzugs können eingebracht werden.
(z.B. Glattheitsbedingungen)

Nachteil:
Aufwand ist größer als bei Kantenverfolgung.



Optimale Kantenzüge (Graphensuche)



Optimalitätskriterien:

- Maximierung der (durchschnittlichen) Gradientenlänge
- Minimierung der Pfadlänge
- Minimierung der Richtungsänderungen
- Minimierung der Grauwertänderungen

Resultate