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)

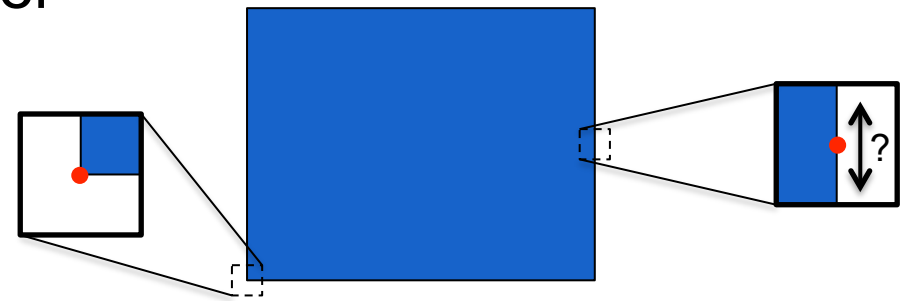
Themen heute

- Ecken (2)
- Segmentierung
  - Schwellenwertbasiert
  - Shading-Korrektur
  - optimaler Schwellenwert: Algorithmus von Otsu
  - Region labeling, flood fill
  - Relaxation labeling
- Regionenbasierte Segmentierung
  - Split-and-Merge
  - Textursegmentierung
- Kantenbasierte Segmentierung
  - Wasserscheidentransformation

ECKEN (2)

Erkennung von Ecken

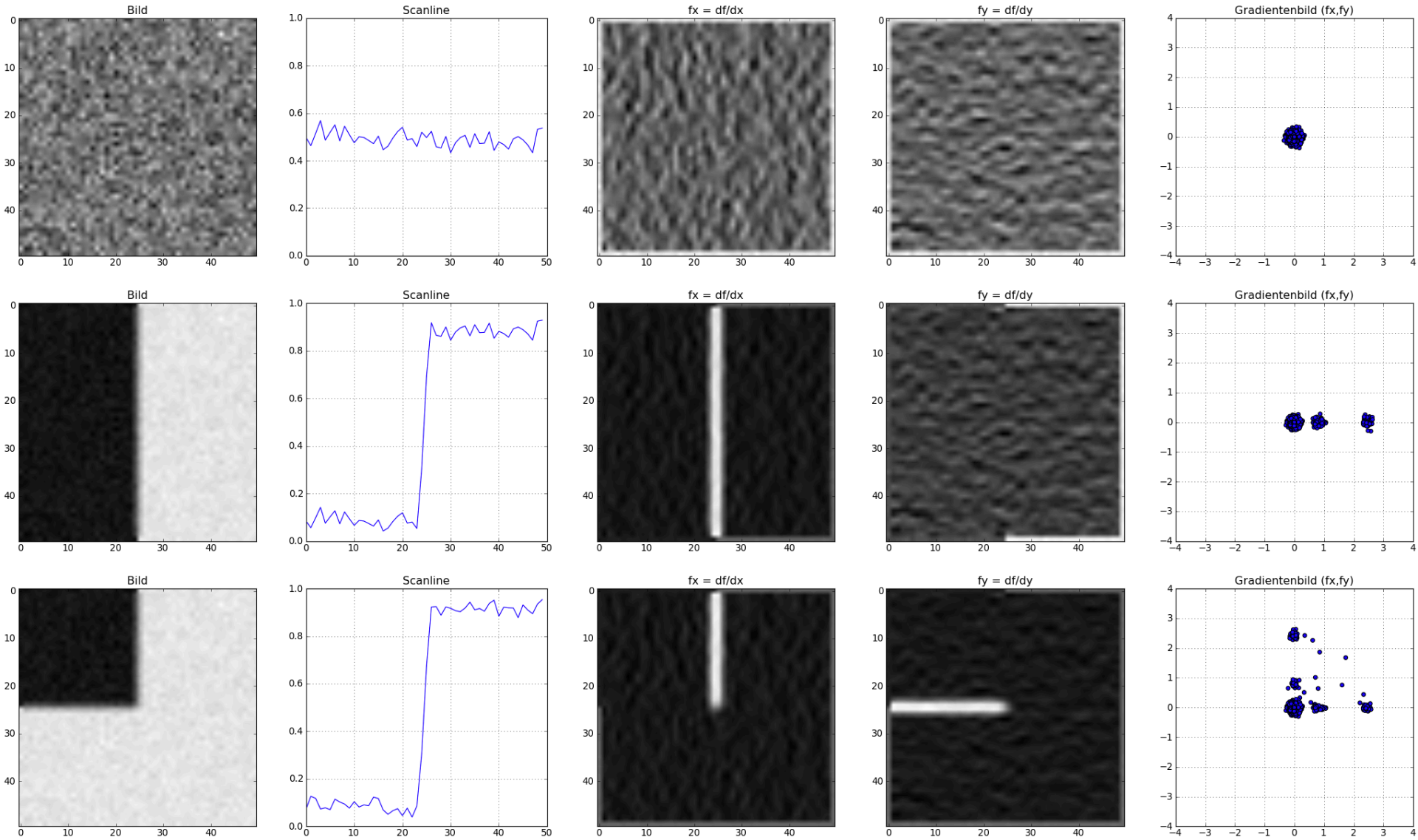
- Viele Bilderkennungsalgorithmen benötigen Merkmale, die eine stabile Position in (x,y) haben
- Kanten sind nur in einer Richtung lokalisiert
  → Ecken in zwei



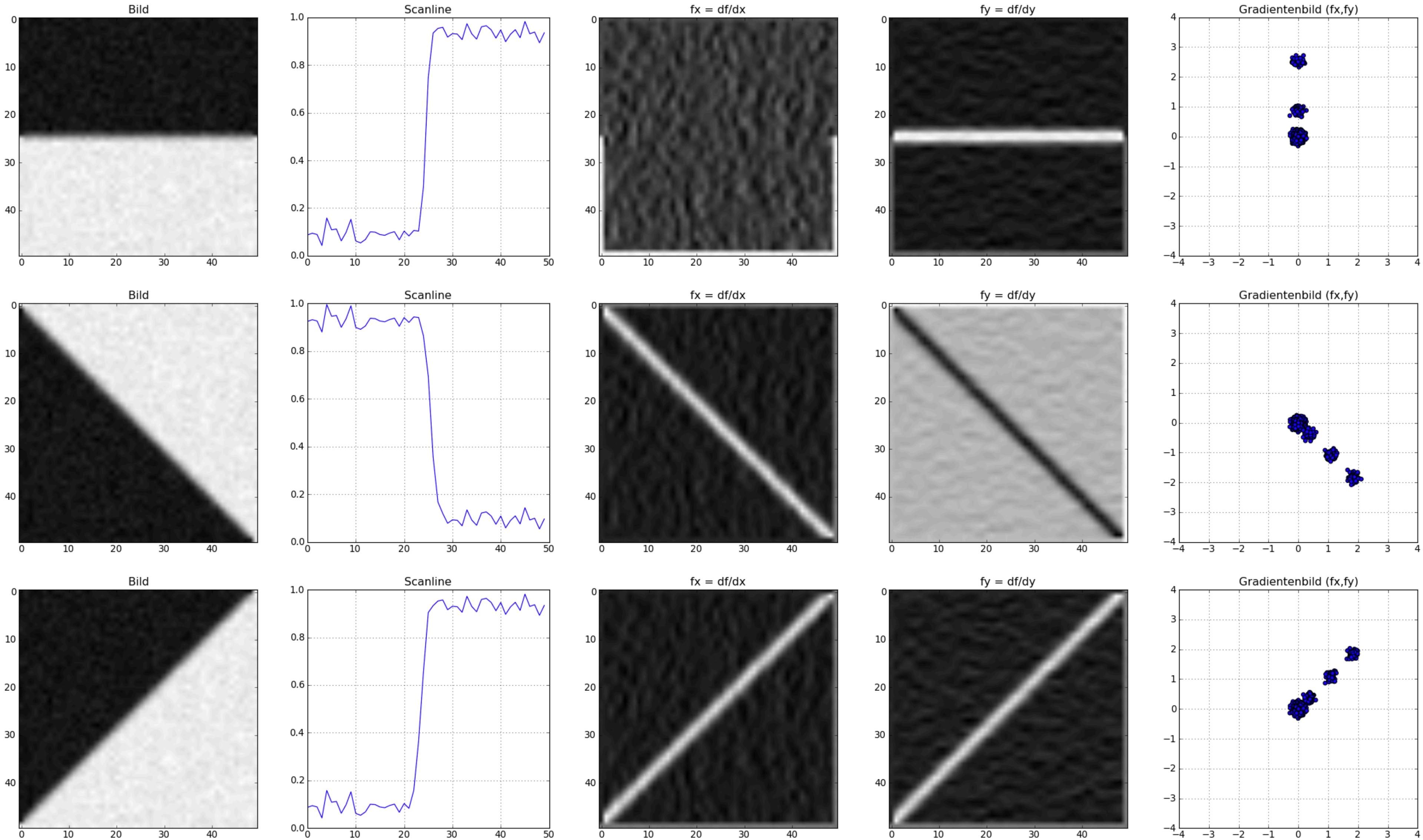
- Gewünschte Eigenschaften von Merkmalen (in verschiedenen Bildern vom gleichem Objekt / der gleichen Szene)
  - Genaue Lokalisierbarkeit
  - Invarianz gegenüber Rotation, Skalierung, Helligkeitsänderung
  - Robust gegenüber Rauschen

Slide and illustration adapted from Bernd Girod, Digital Image Processing

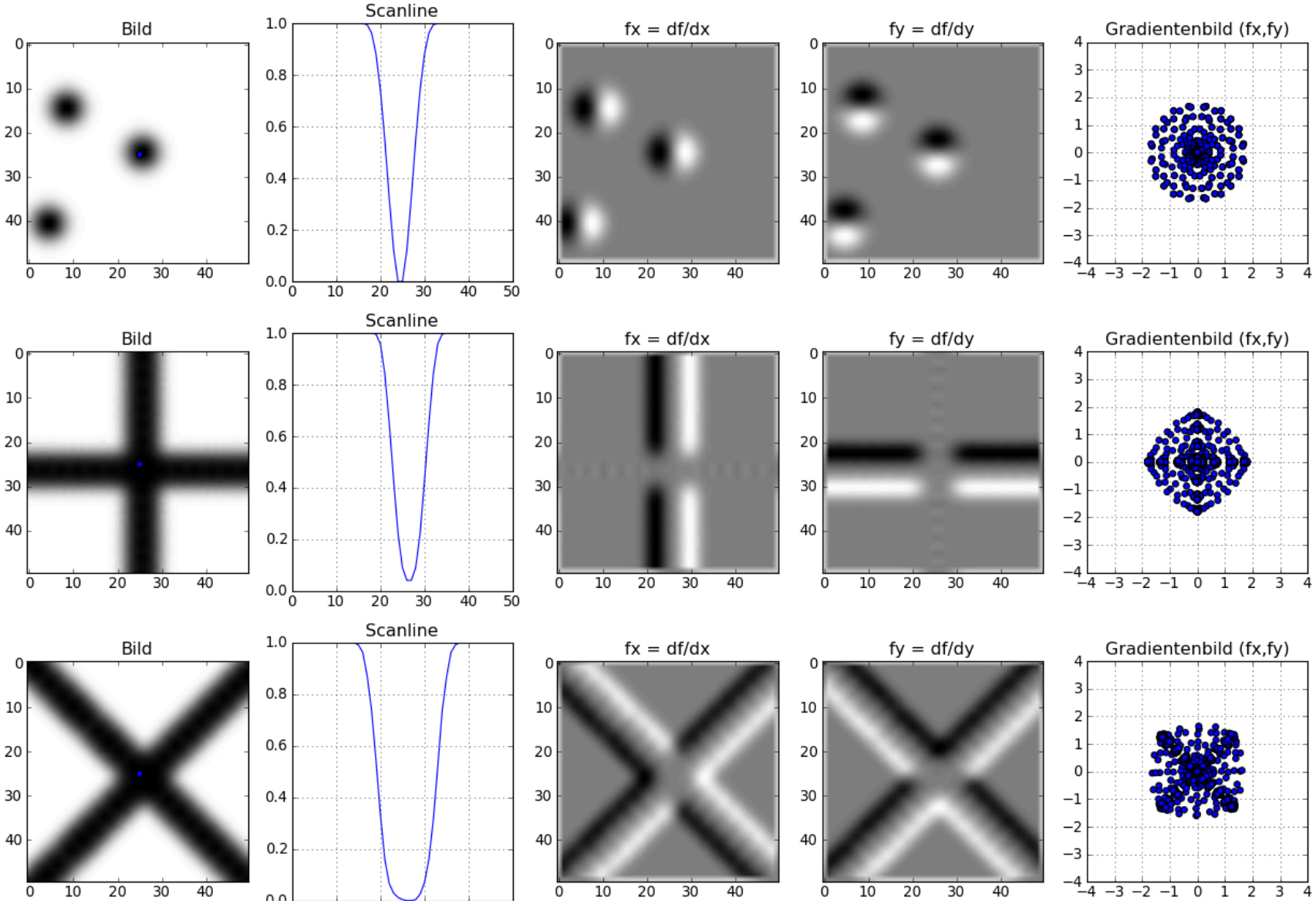
Verteilung der Gradienten



Verteilung der Gradienten

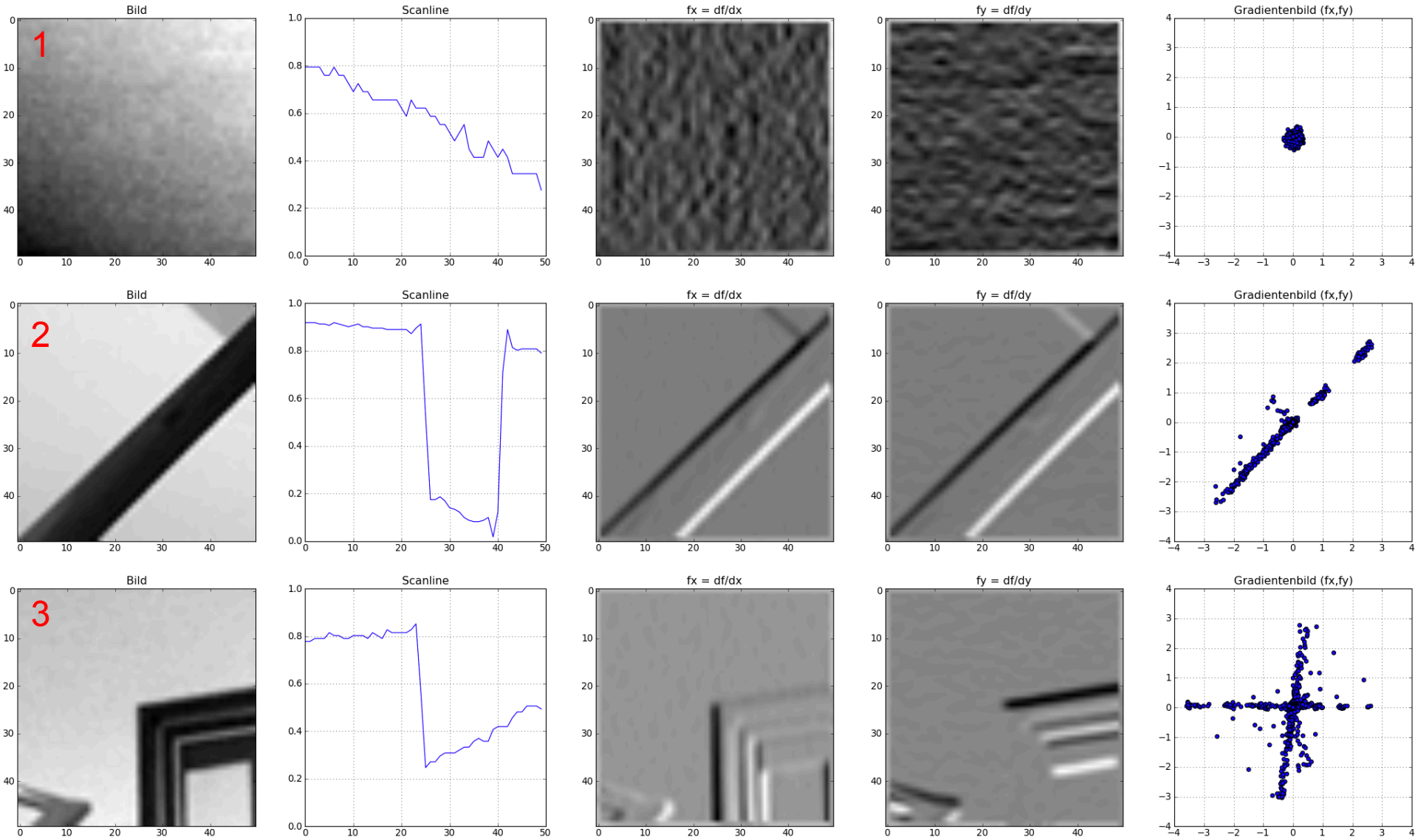


Verteilung der Gradienten



Verteilung der Gradienten in realem Bild

Verteilung der Gradienten in realem Bild



Verschiebung

- Effekt einer Verschiebung um (kleine) Δx, Δy
  - Flache Region: keine Änderung im Erscheinungsbild
  - Kante: keine Änderung bei Verschiebung entlang der Kante
  - Ecke: große Änderung in jeder Richtung
- Harris-Corner-Detektor beschreibt diese Fälle
- Intensitätsänderung bei Verschiebung um Δx, Δy

$$s(\Delta x, \Delta y) = \sum_{(x,y) \in window} \left[f(x,y) - f(x+\Delta x, y+\Delta y) \right]^2$$

- Lineare Approximation für (kleine) Δx, Δy

$$f(x+\Delta x, y+\Delta y) \approx f(x,y) + f_x(x,y)\Delta x + f_y(x,y)\Delta y$$

  - f_x, f_y sind Gradienten in x- bzw. y-Richtung

Verschiebung

- Lineare Approximation für (kleine) Δx, Δy

$$s(\Delta x, \Delta y)$$

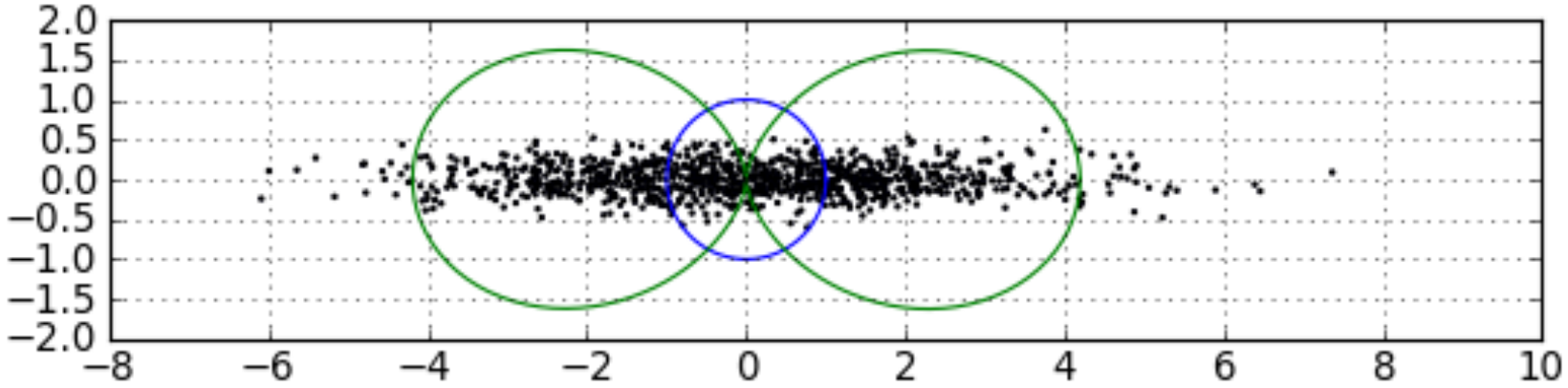
$$\approx \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \begin{pmatrix} \sum_{(x,y)\in window} f_x^2 & \sum_{(x,y)\in window} f_x f_y \\ \sum_{(x,y)\in window} f_x f_y & \sum_{(x,y)\in window} f_y^2 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

$$= \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} M \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

- M: Kovarianzmatrix, „autocorrelation matrix“, „second moment matrix“

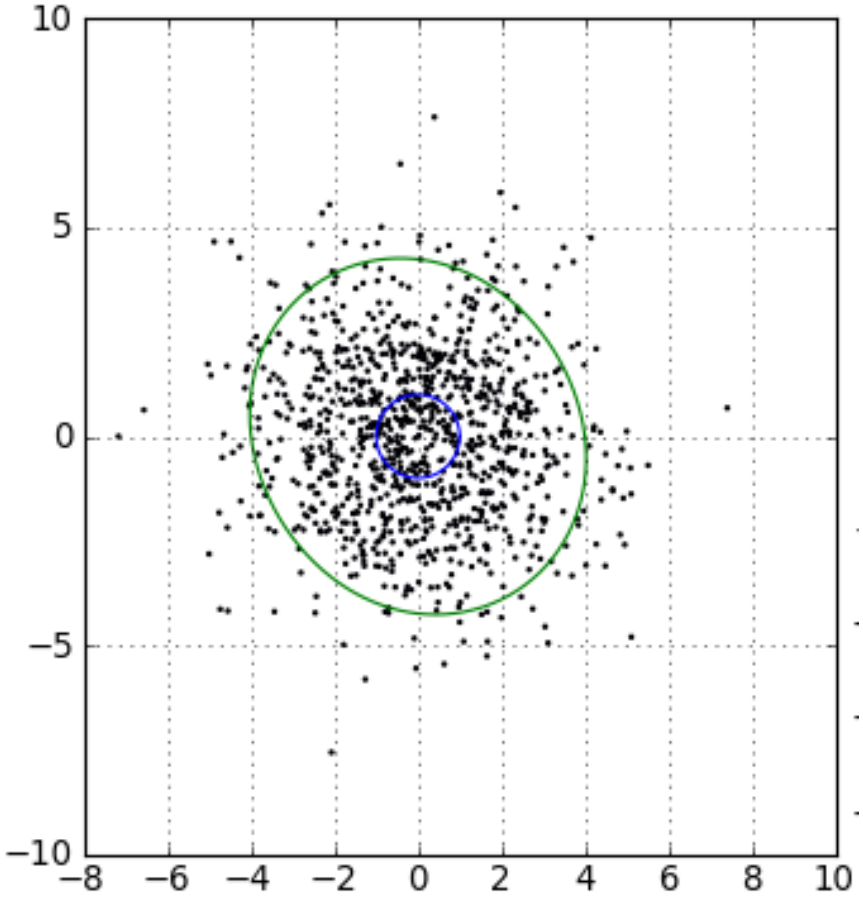
Intensitätsänderung bei Verschiebung

- σ_x=2.0, σ_y=0.2
- blauer Kreis: (Δx, Δy)
- grüne Kurve: s(Δx, Δy) • $(\Delta x, \Delta y)^T$



Intensitätsänderung bei Verschiebung

- σ_x=2.0, σ_y=2.0
- blauer Kreis: (Δx, Δy)
- grüne Kurve:
- $s(\Delta x, \Delta y) \bullet (\Delta x, \Delta y)^T$



Harris Corner Detektor (Harris, Stephens, 1988)

- Kovarianzmatrix, „autocorrelation matrix“, „second moment matrix“

$$M = \sum_{(x,y)\in window} w(x,y) * \begin{pmatrix} f_x^2(x,y) & f_x(x,y)f_y(x,y) \\ f_x(x,y)f_y(x,y) & f_y^2(x,y) \end{pmatrix}$$

w(x,y): Gewichtungsfunktion, z.B. Gauß-Funktion

- Maß für die “Stärke” der Ecke

$$C(x,y) = \det(M) - k(trace(M))^2 = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$$

k = 0.04..0.06

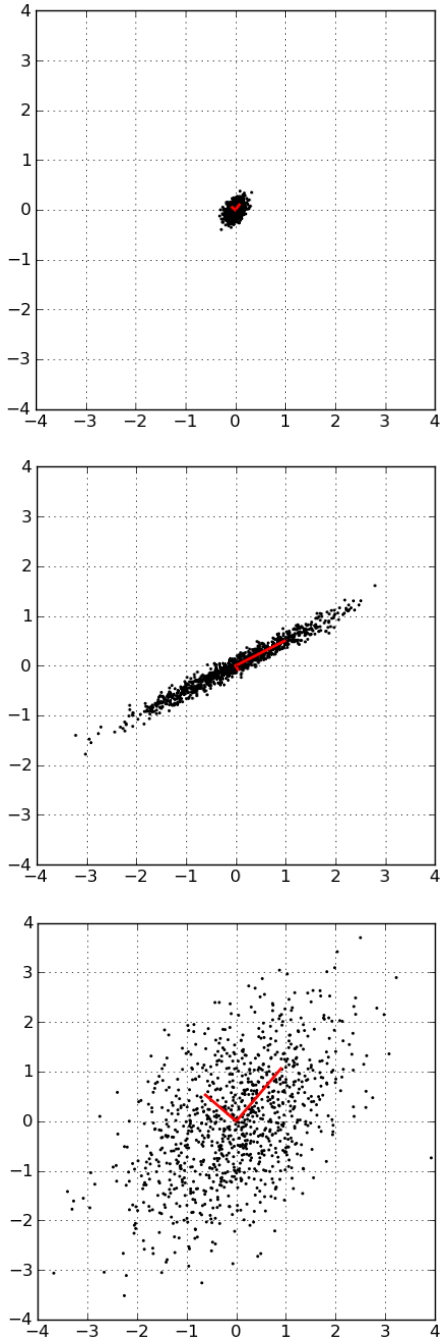
Harris Matrix

- Eigenvektoren v_1, v_2; Eigenwerte λ_1, λ_2
  - Richtung: Hauptachsen der Verteilung
  - Länge: Varianz entlang der Hauptachsen
- Keine dominante Orientierung
  - $\lambda_1 = 0.017$
  - $\lambda_2 = 0.006$

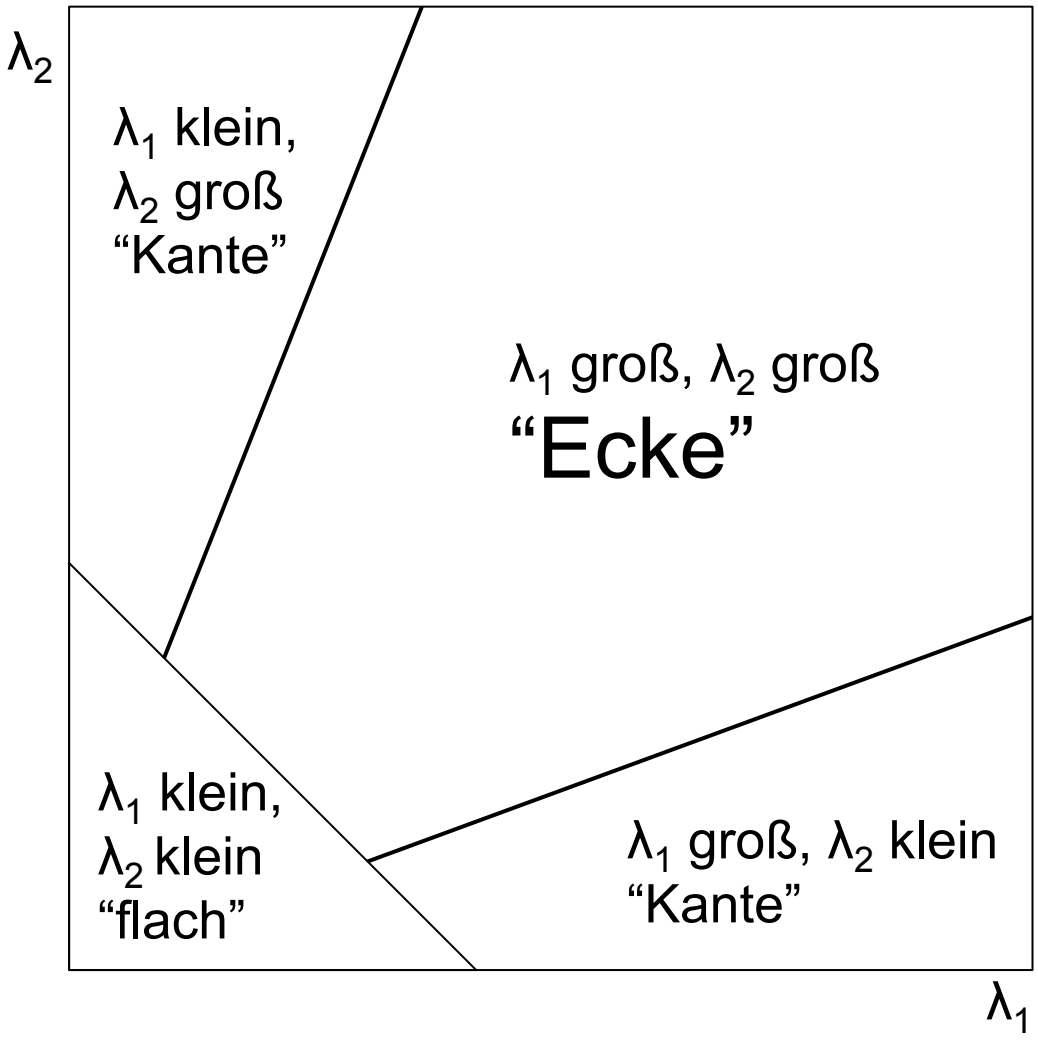
  → λ_1 klein, λ_2 klein
- Nur eine dominante Orientierung
  - $\lambda_1 = 1.313$
  - $\lambda_2 = 0.008$

  → λ_1 groß, λ_2 klein
- Mehrere dominante Orientierungen
  - $\lambda_1 = 1.936$
  - $\lambda_2 = 0.669$

  → λ_1 groß, λ_2 groß



Klassifikation über Eigenvektoren



Harris Corner Detection – Algorithmus

- Berechne die Ableitungen f_x und f_y in x- und y-Richtung
  - z.B. per Konvolution mit dem Sobel-Operator
- Berechne die elementweisen Produkte der Ableitungen

  $f_x^2 = f_x f_x, \quad f_y^2 = f_y f_y, \quad f_{xy} = f_x f_y$
- Berechne $f_{xxSum} = f_x^2 * w$, $f_{yySum} = f_y^2 * w$, $f_{xySum} = f_{xy} * w$, wobei „*“ der Konvolutionsoperator ist und

$$w = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

- Definiere für jedes Pixel (x,y) die Matrix M(x,y)

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

Harris Corner Detection – Algorithmus

- Berechne für jedes Pixel das Maß der Eckenstärke

$$C(x,y) = \det(M) - k(trace(M))^2 = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$$

  - k = 0.04..0.06
- Wende eine untere Schranke und non-maximum Unterdrückung an
  - z.B. im Radius 2 um den jeweils betrachteten Punkt

Beispiel

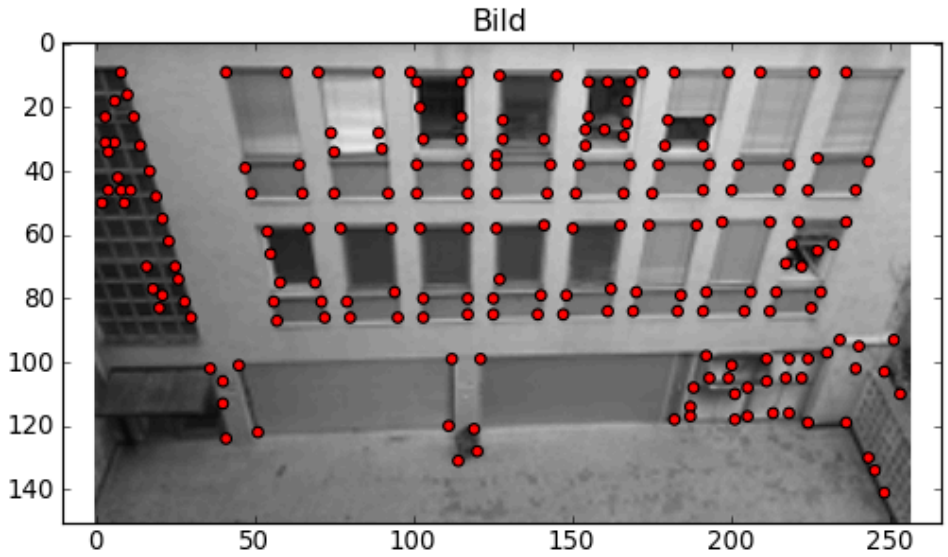


Bild
0
20
40
60
80
100
120
140
0
50
100
150
200
250

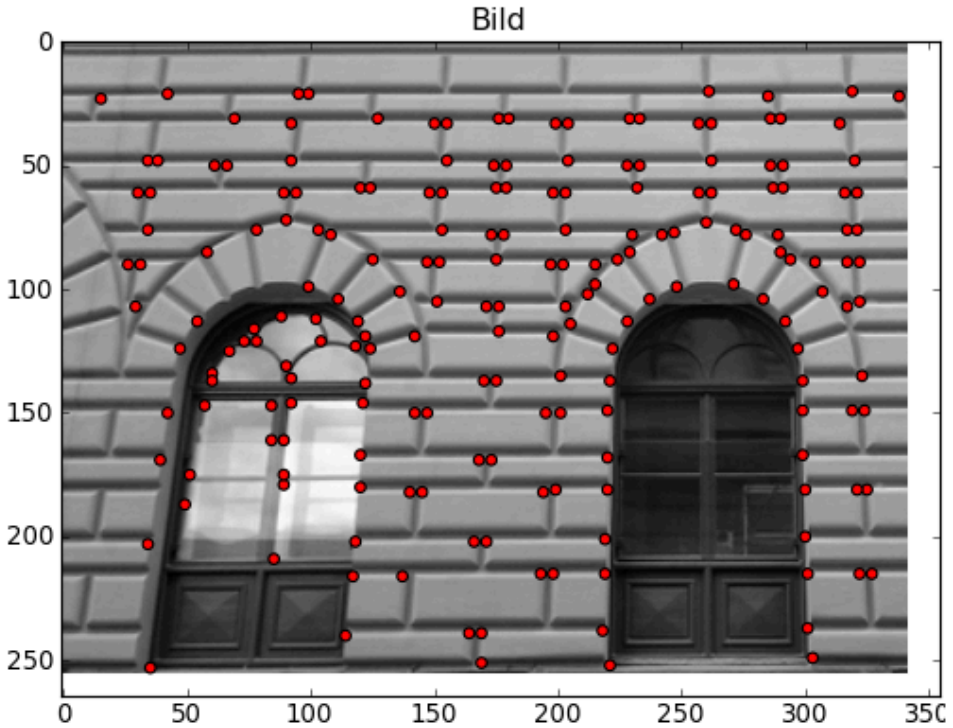
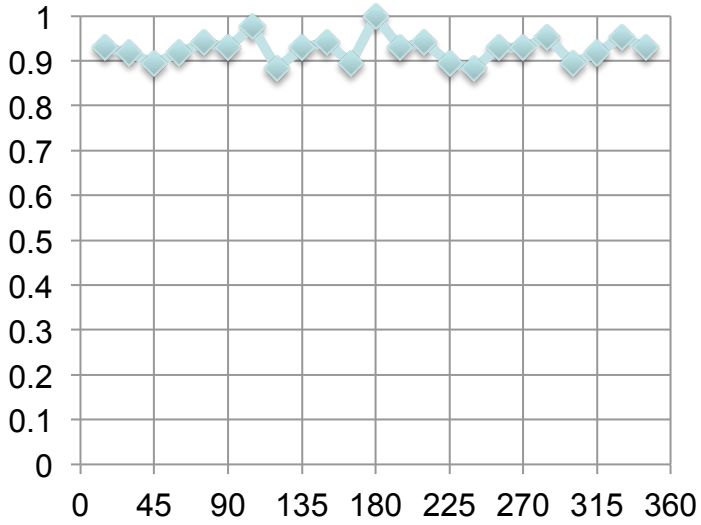
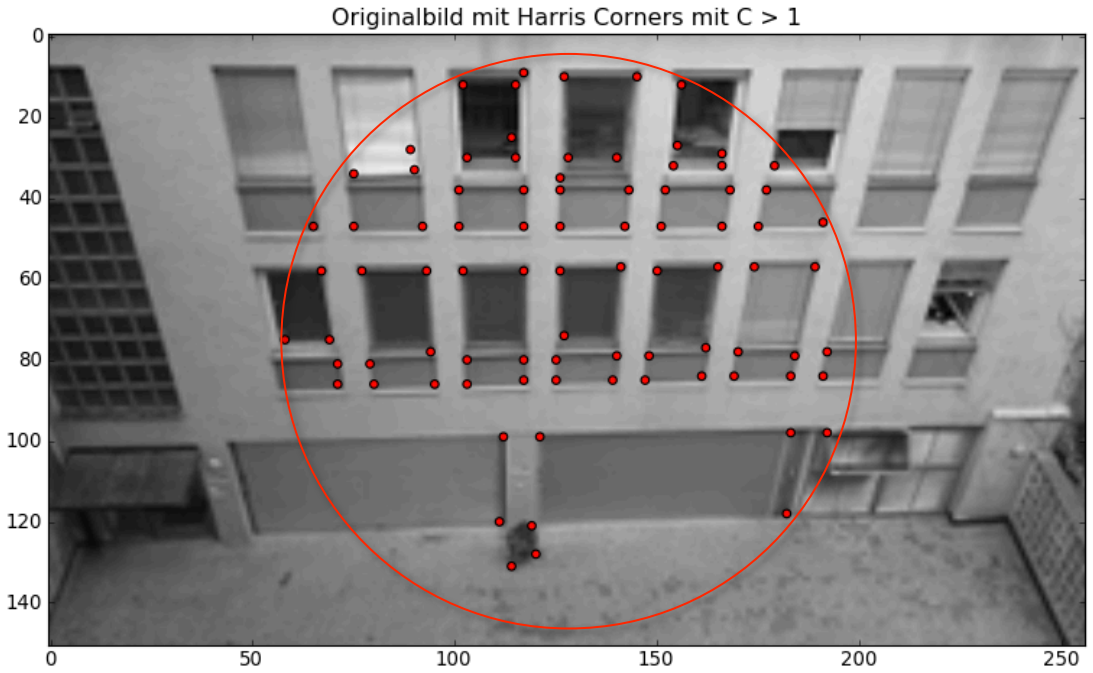


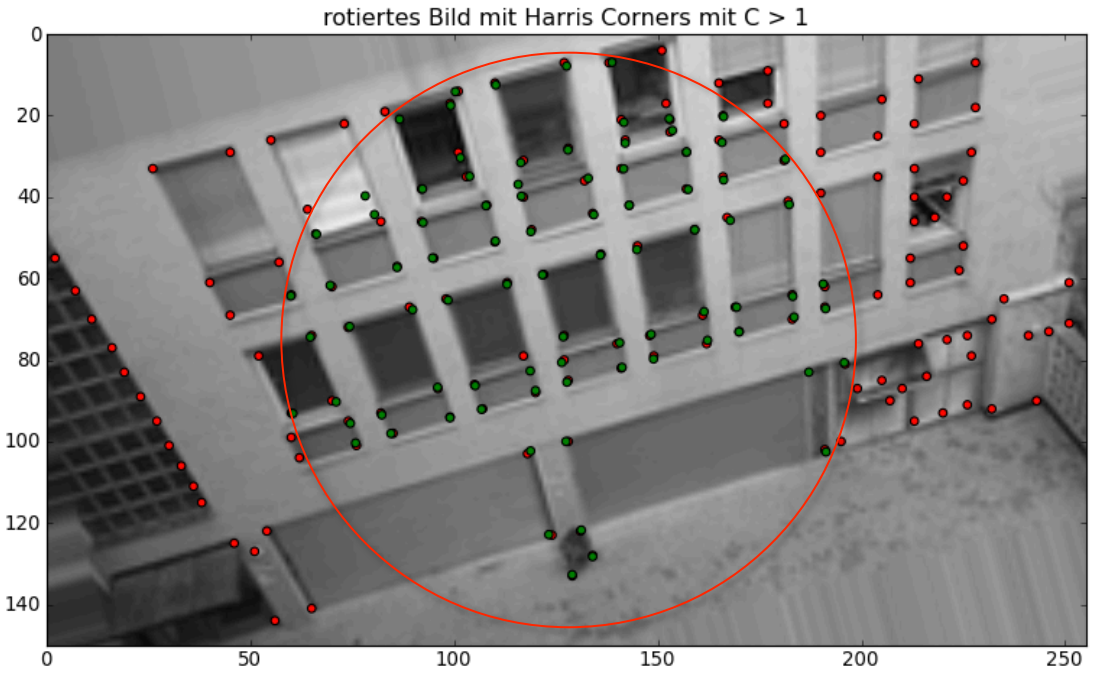
Bild
0
50
100
150
200
250
0
50
100
150
200
250
300
350

Repeatability

- Invarianz des Eckenmaßes auf Rotation, Skalierung, Änderung der Perspektive
- hier Rotation:

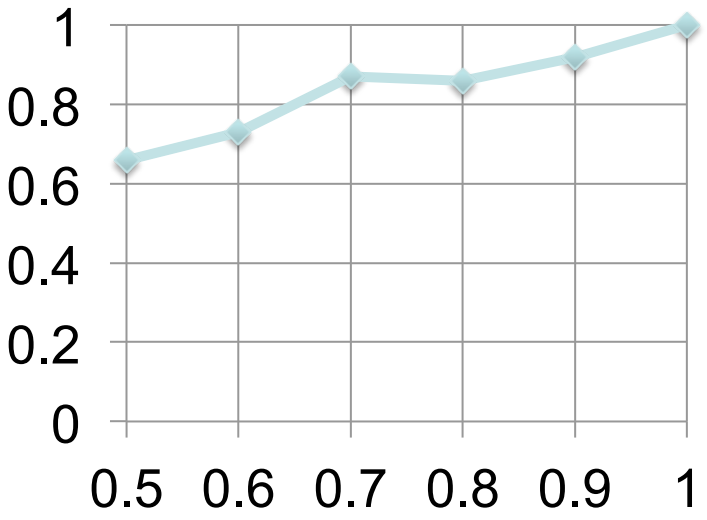


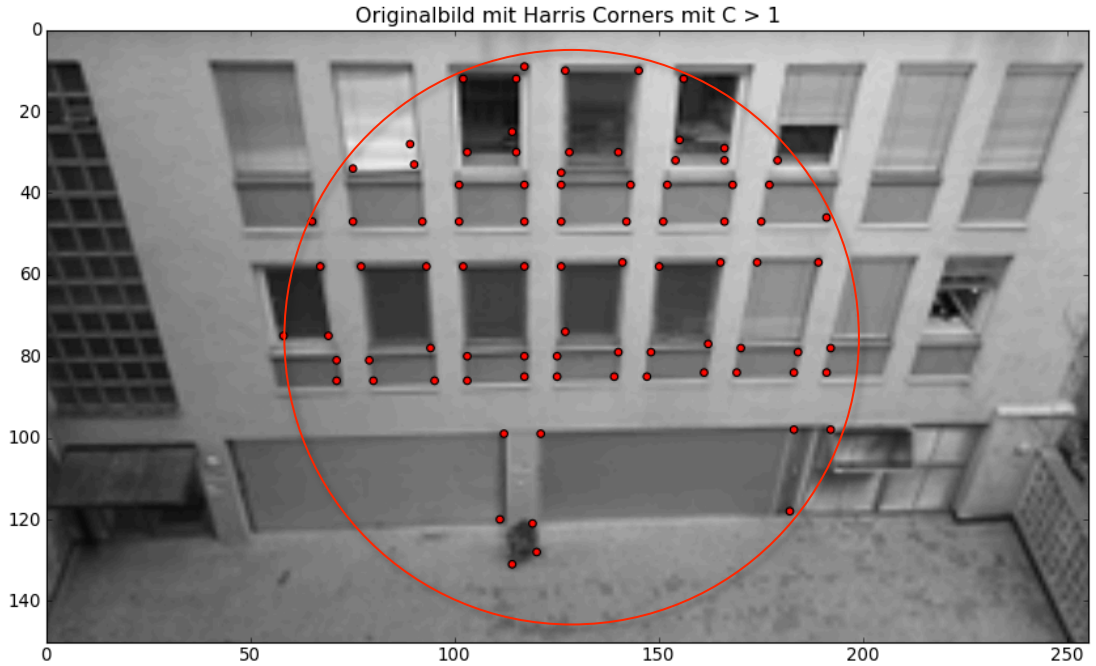


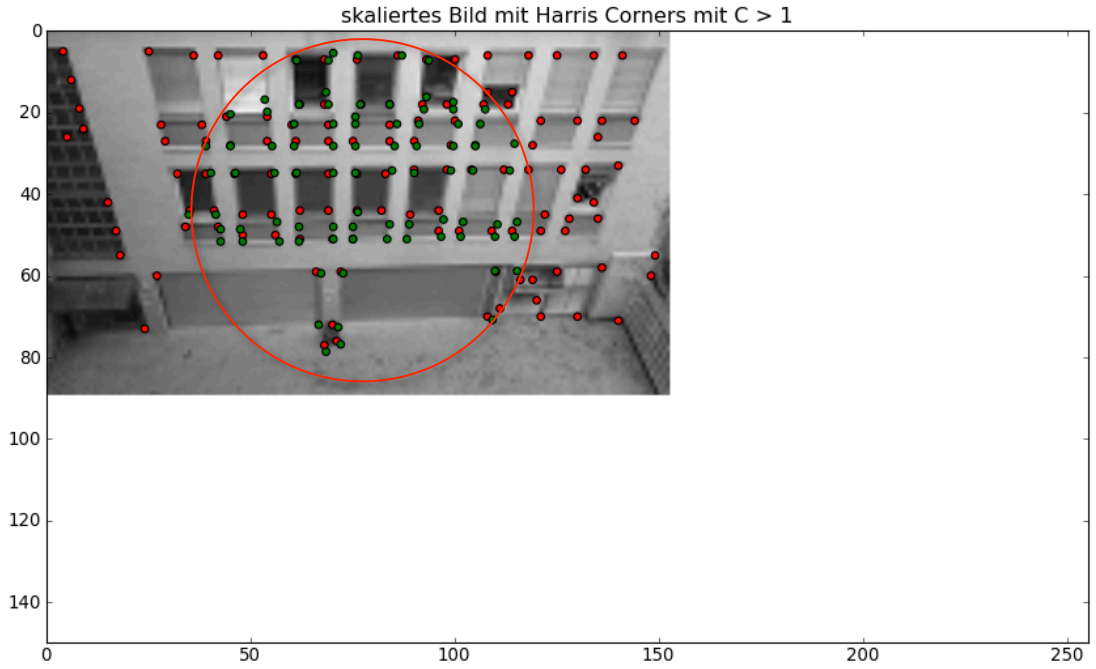


Repeatability

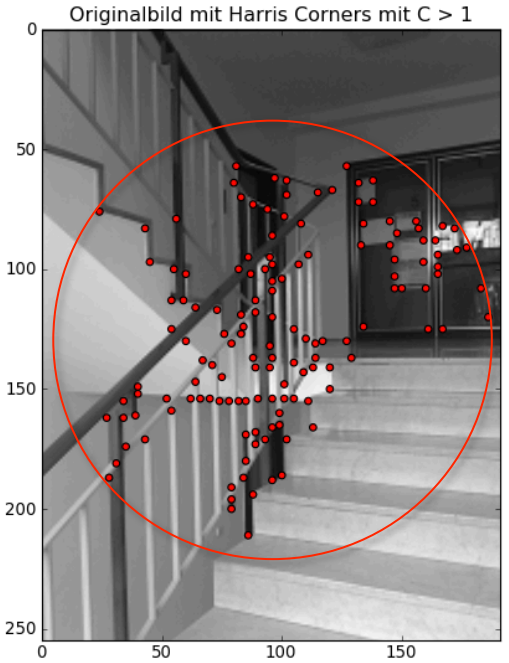
- Invarianz des Eckenmaßes auf Rotation, Skalierung, Änderung der Perspektive
- hier Skalierung:



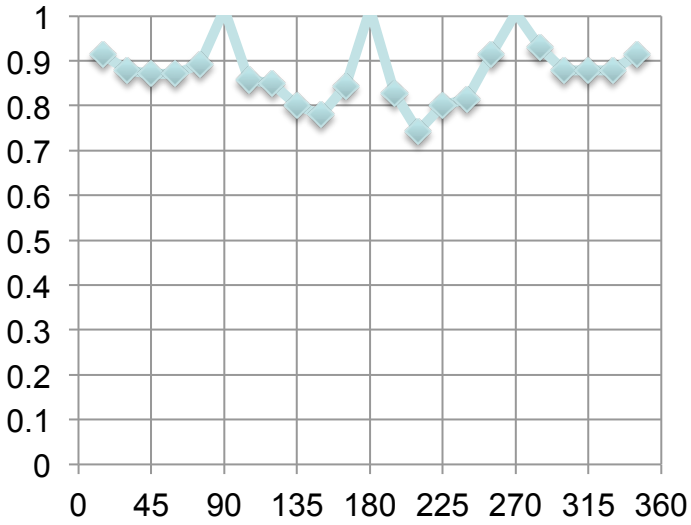


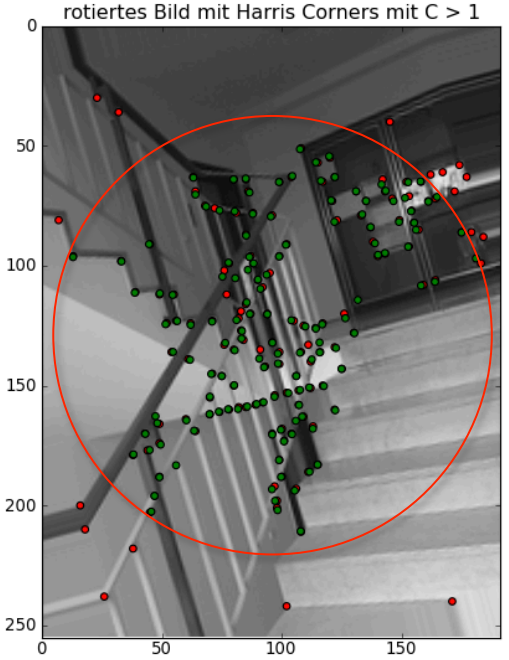


Repeatability



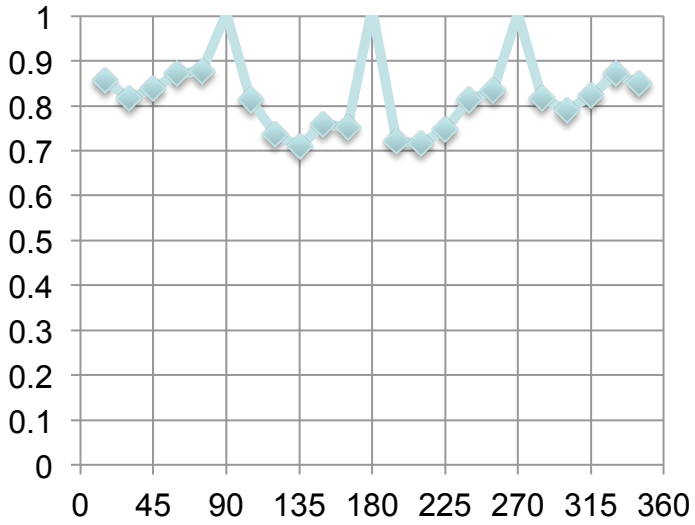
- Rotation:

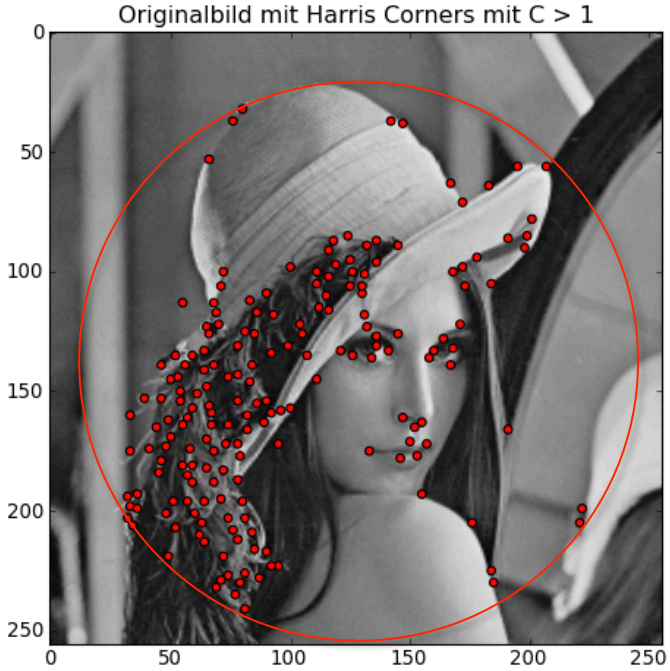


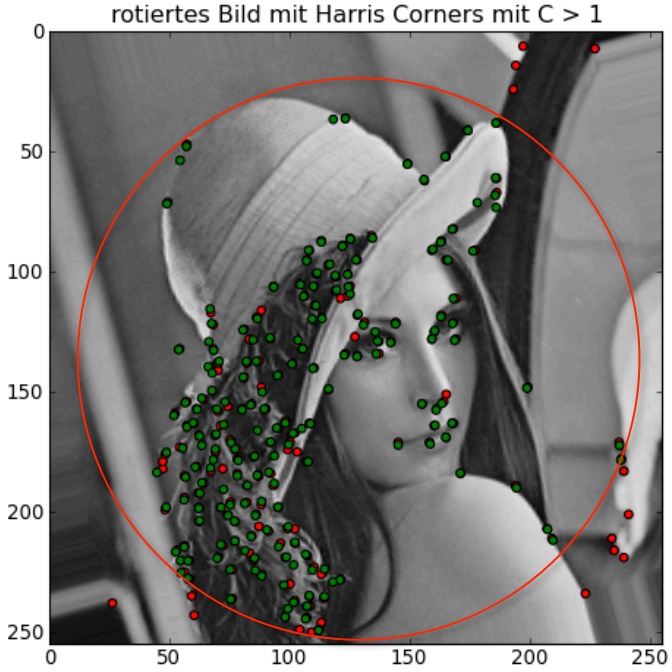


Repeatability

- Rotation:





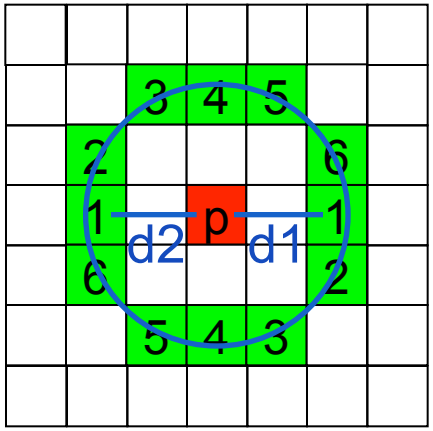


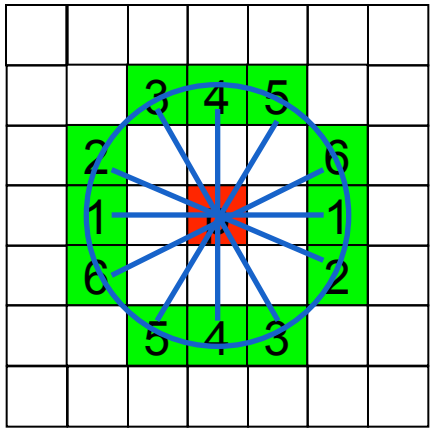
Simple Corner Detector

- Corner response of pixel p = (x,y)

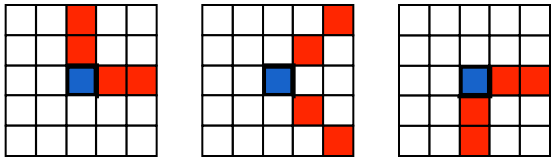
```
// I(p) = intensity (grayscale value) of pixel p
dmin = ∞
for all opposite points (p1,p2) on circle
    d1 = abs(I(p) - I(p1))
    d2 = abs(I(p) - I(p2))
    d = max(d1, d2)
    dmin = min(dmin, d)
cornerResponse = dmin
```

- Non-maximum suppression
- Threshold to generate ~150 corners

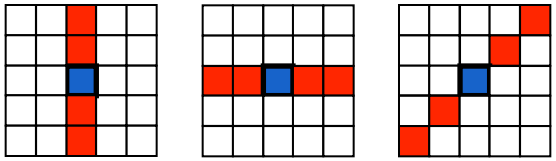




Ecken:



Keine Ecken:



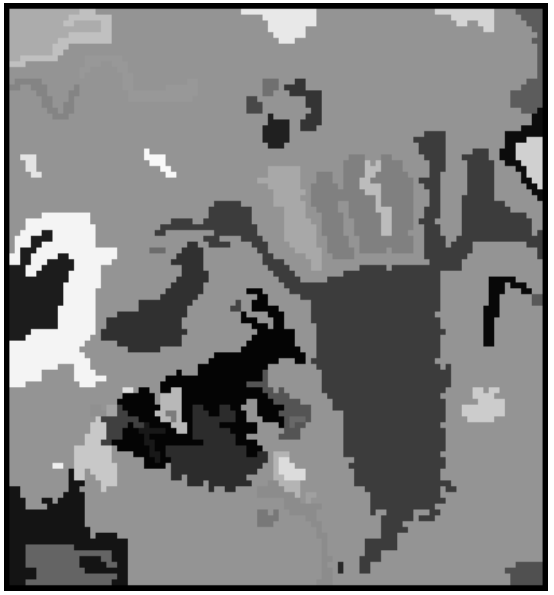
FAST Corner Detector

- auf voriger Folie: vereinfachte Version des FAST Corner Detectors
  - FAST: Features from Accelerated Segment Test
- E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In Proc. of European Conference on Computer Vision, May 2006.
- Algorithmus: für jedes Pixel p
  - konstruiere Bresenham-Kreis mit Radius r = 3
  - früher Abbruch möglich, falls d_{min} früh unter Schwellenwert

SEGMENTIERUNG

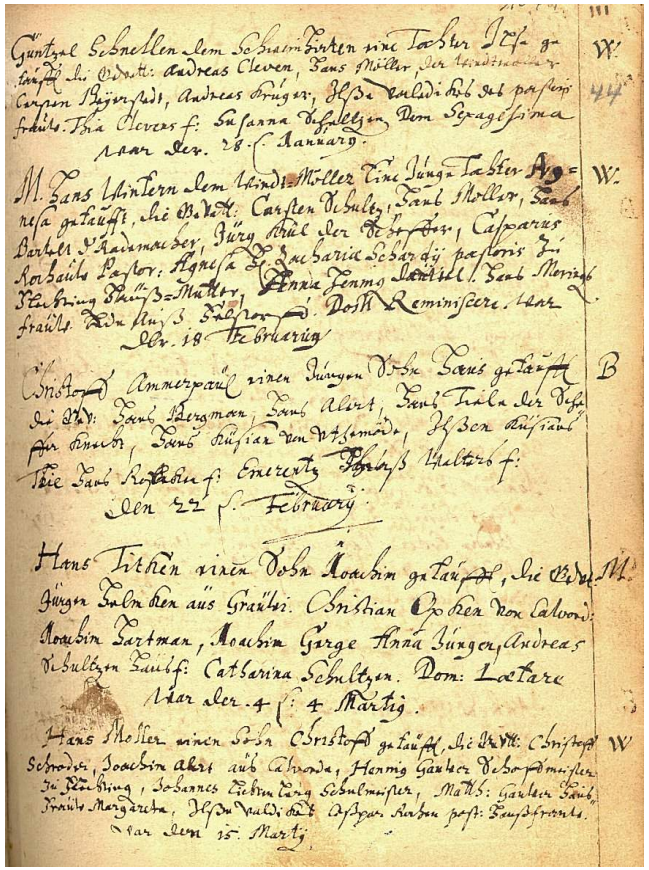
Segmentierung

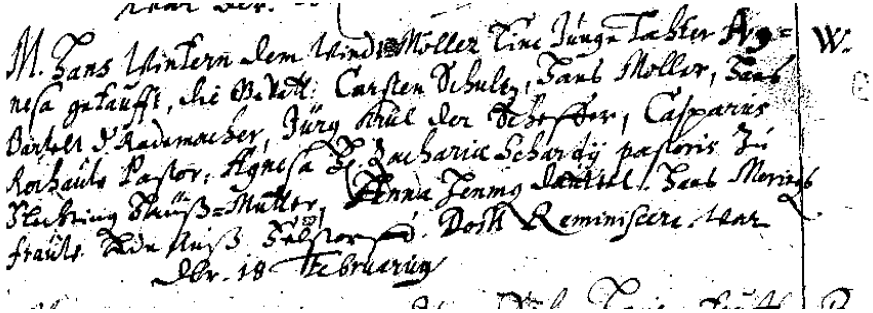
- Ziel: Pixelaggregate schaffen, denen Bedeutung zugeordnet werden kann
  - Vordergrund vom Hintergrund trennen
- Pixel gehören zusammen, wenn sie einem zusammenhängenden Gebiet angehören
- Homogenitätsbedingung ist vom Bildinhalt unabhängig
  - Segmentierung ist datenbasiert
- Modellbasierte Segmentierung: Zwitter zwischen Segmentierung und Analyse
  - z.B. Kreise als Modell, Rest als Hintergrund



Segmentierung

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

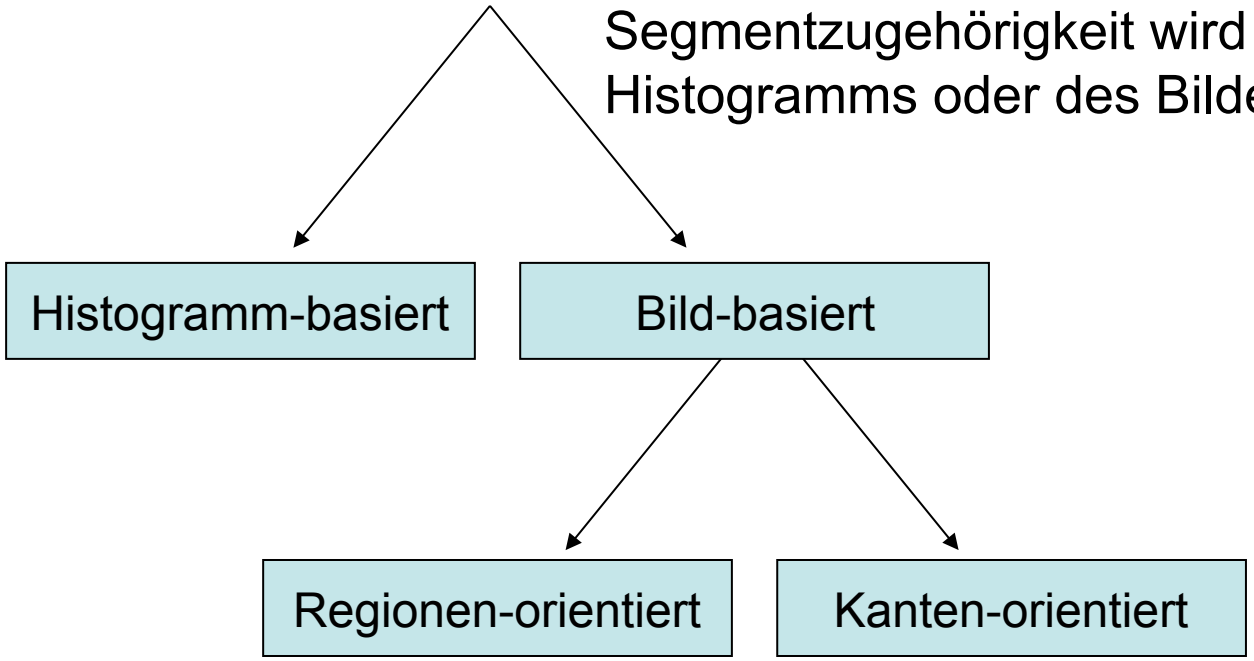




Segmentierungsmethoden

Histogramm-basierte oder
Bild-basierte Segmentierung:

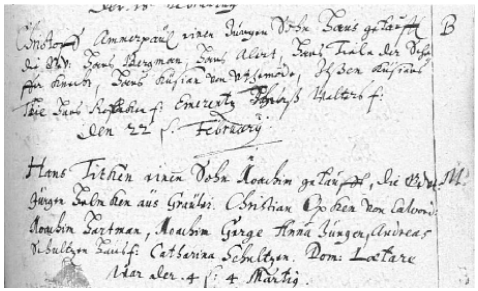
Segmentzugehörigkeit wird anhand des Histogramms oder des Bildes entschieden

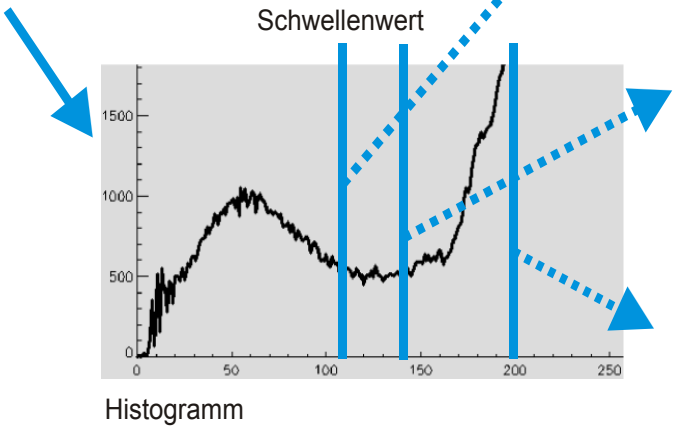


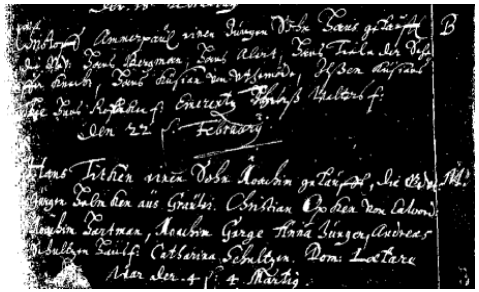
Regionen- oder Kantenorientierung:

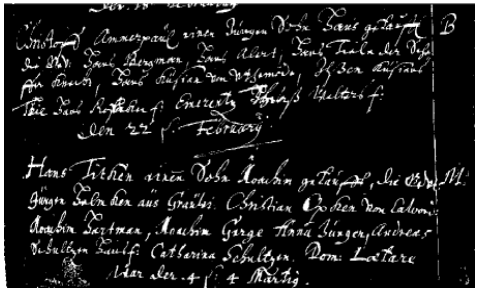
Segmente werden durch ihre Grenzen oder ihr Inneres definiert

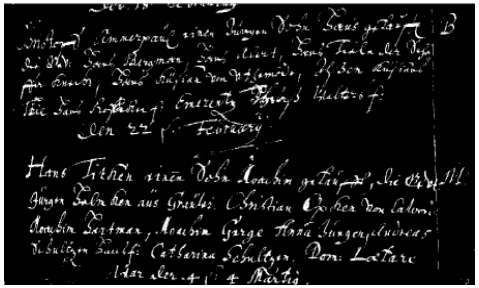
Histogramm-basierte Segmentierung











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

Aufgabe: Schwellenwert zwischen Vorder- und Hintergrund finden

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

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, bzw. in C_2:

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

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

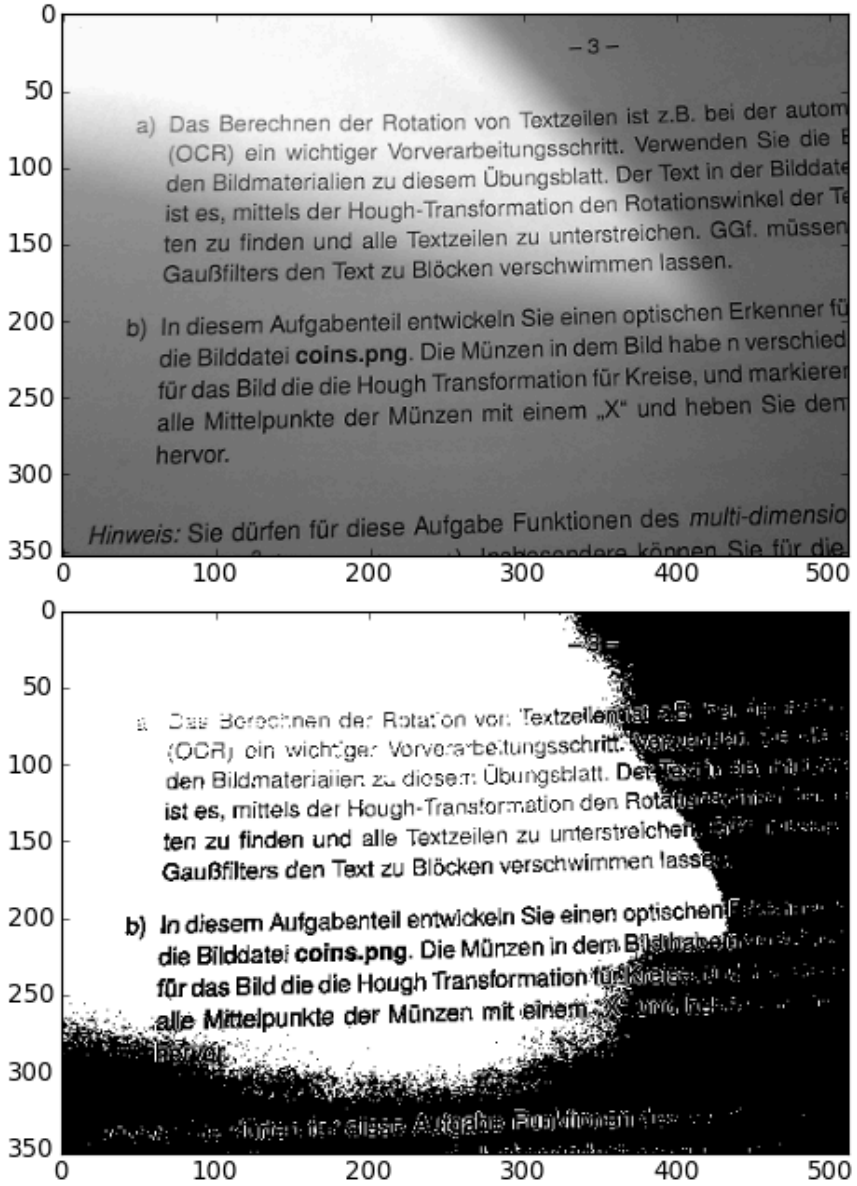
$$k^* = \arg\max_{0 \le k \le L-1} \sigma_B^2(k)$$

Segmentierung:

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

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



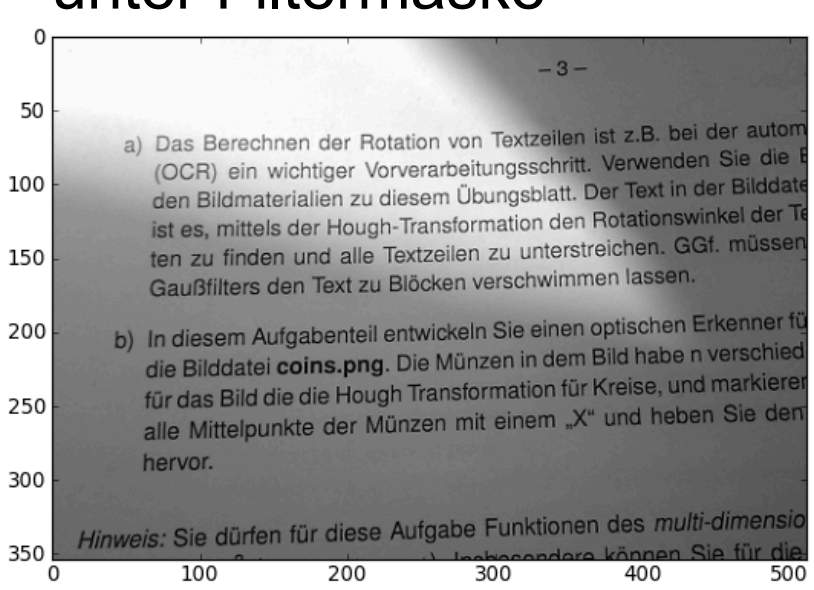
Berücksichtung von Shading

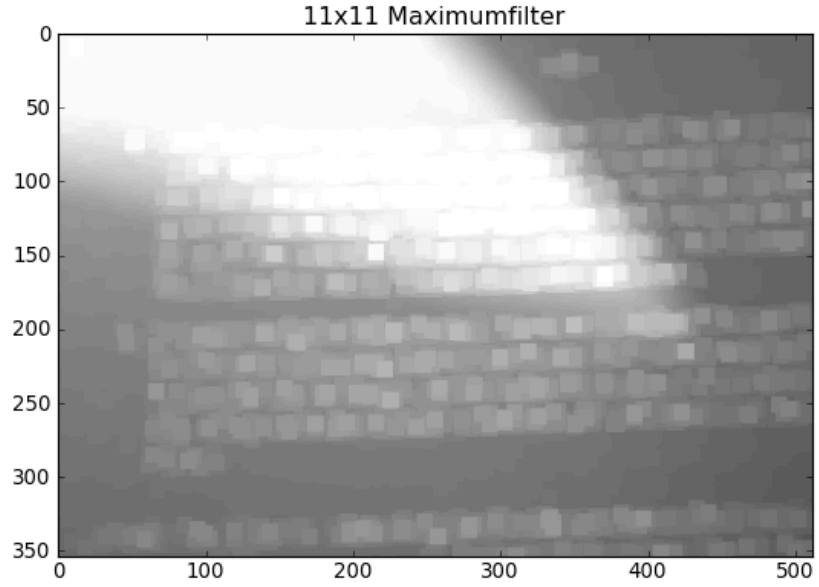
- Homogenes Bild unter gleichen Bedingungen aufnehmen
- Shading-Bild aus dem Bild selbst bestimmen
- Variierende Schwellenwerte

- Shading invertieren/subtrahieren

Bestimmung des Shading-Bilds

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





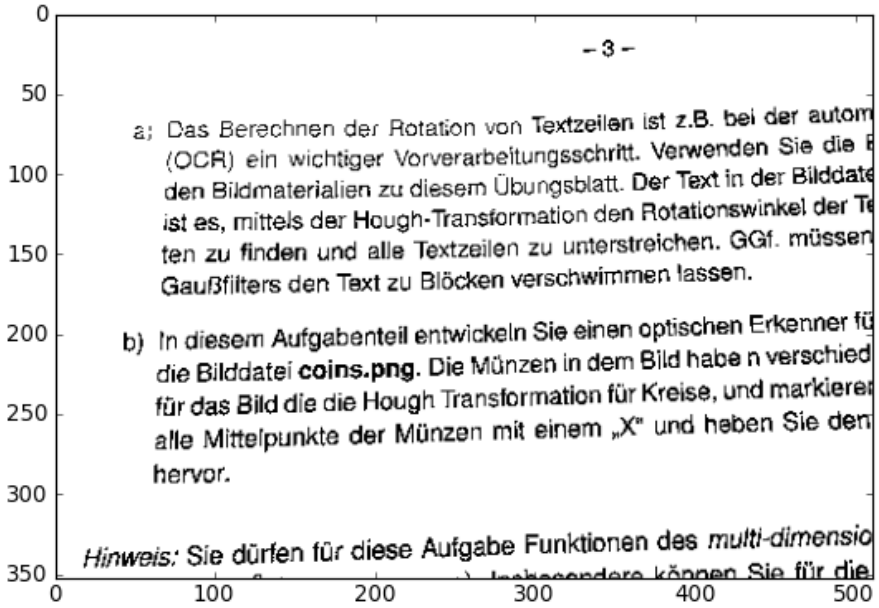
Shading-Korrektur

- Berechne aus dem Hintergrundbild Shading-Funktion $s(i,j)$

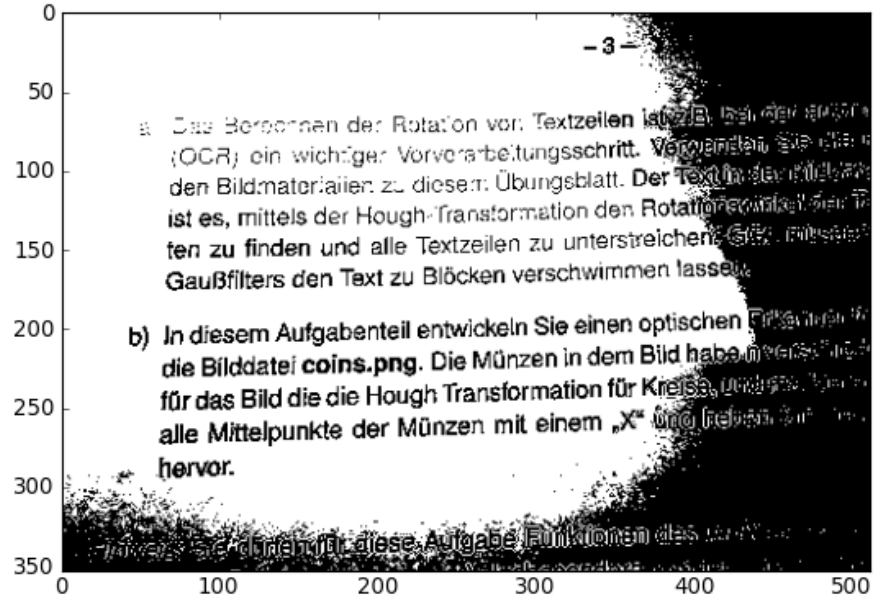
- Korrektur:
  $g(i,j) = f(i,j) / s(i,j)$
  oder
  $g(i,j) = f(i,j) - s(i,j)$

- Segmentierung auf dem korrigierten Bild

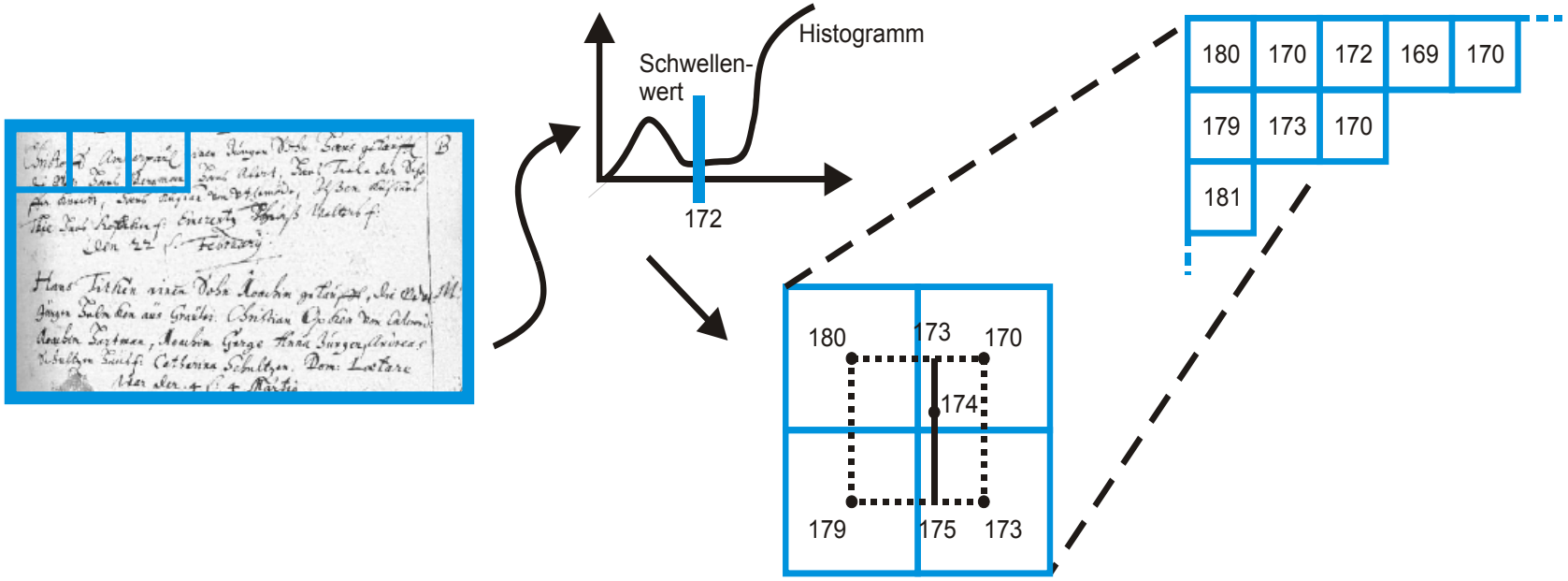
mit Korrektur



ohne Korrektur



Variierende Schwelle



Lokale Schwellenwerte aus Histogrammen in Teilregionen

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

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

Region Labeling

- 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
```

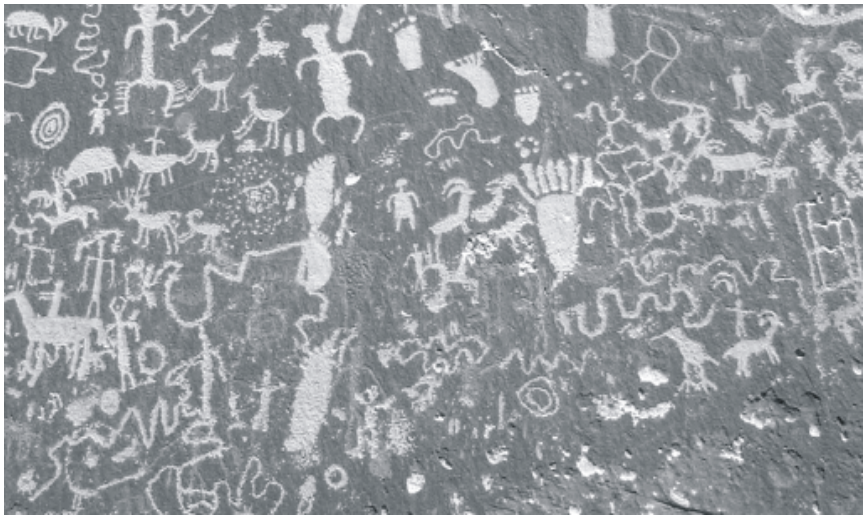
Flood Fill

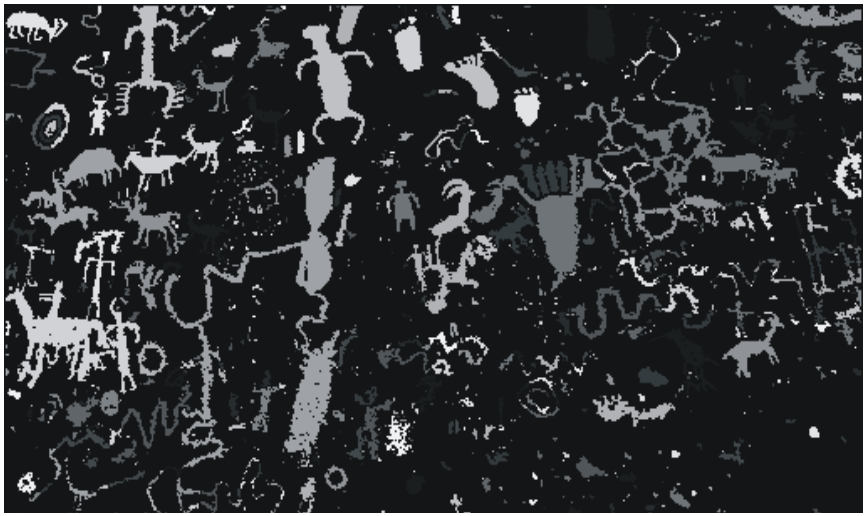
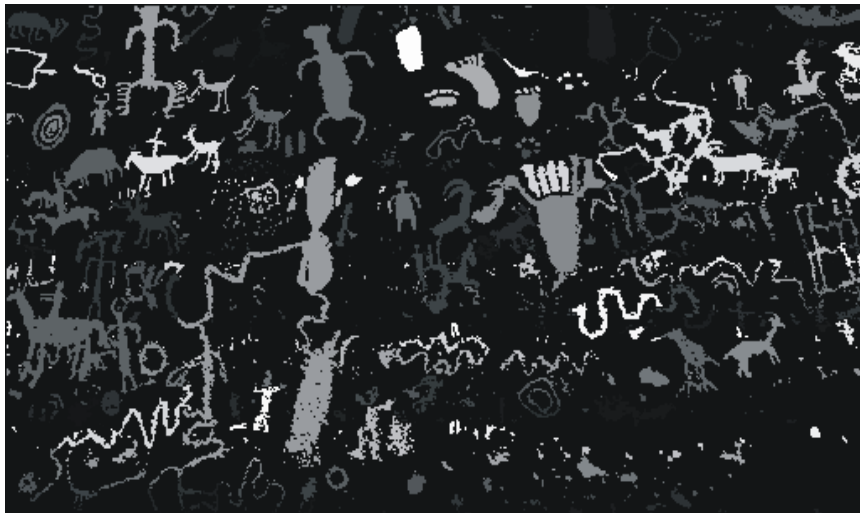
```
flood_fill(i,j,label)  // Variablen zur Auswertung der
        Zusammenhangsbedingung sind global verfügbar
if f(i,j) erfüllt Zusammenhangsbedingung then
    region(i,j) = label      // Region an (i,j) mit Label
                             // versehen
    flood_fill(i-1,j,label) // Nachbarpixel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

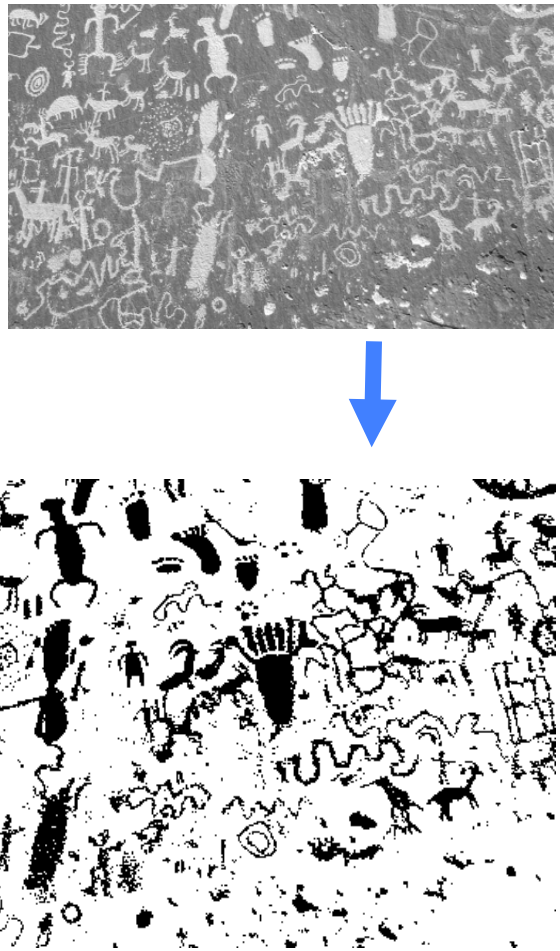
Resultat



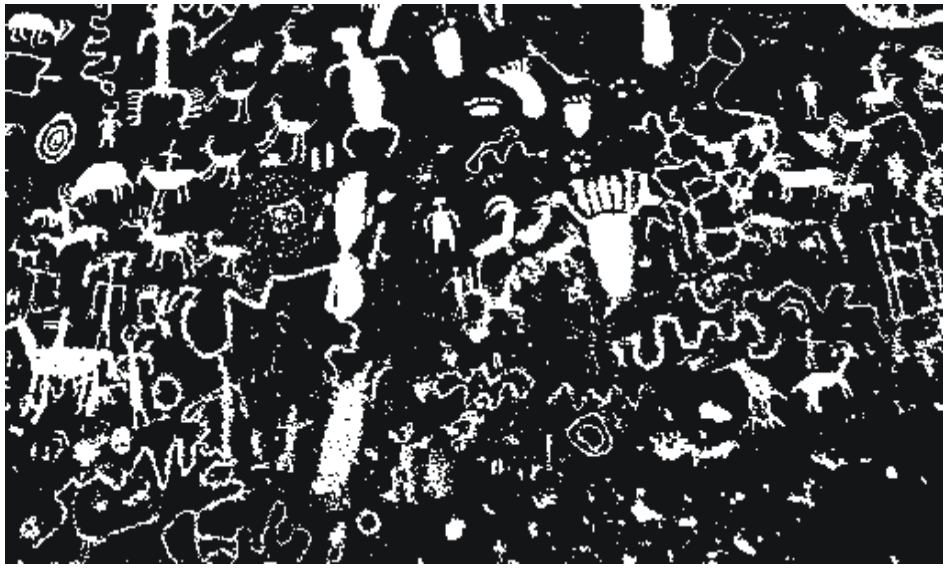



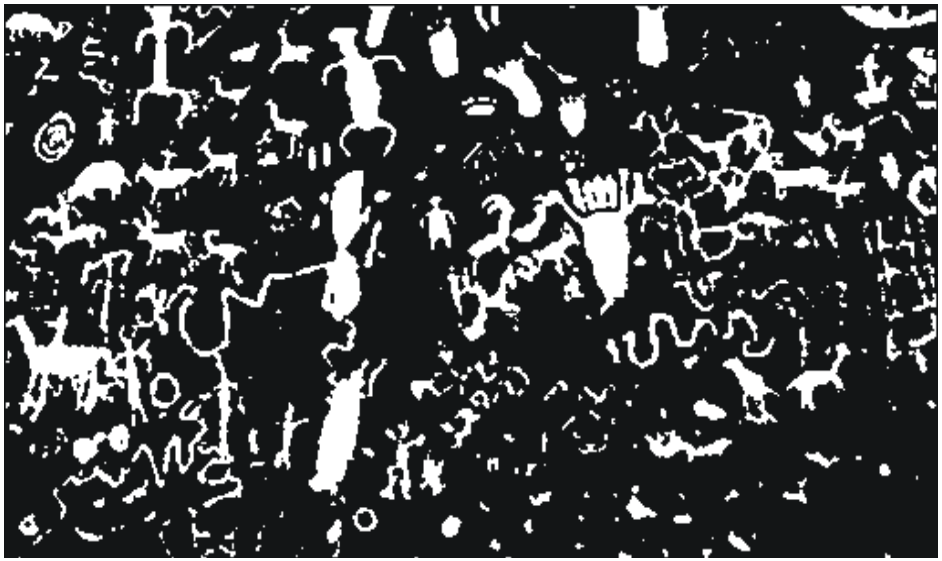
Nachverarbeitung

- 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
  - Relaxation Labeling



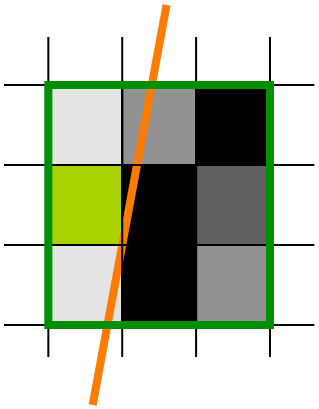
Medianfilterung auf Labeln

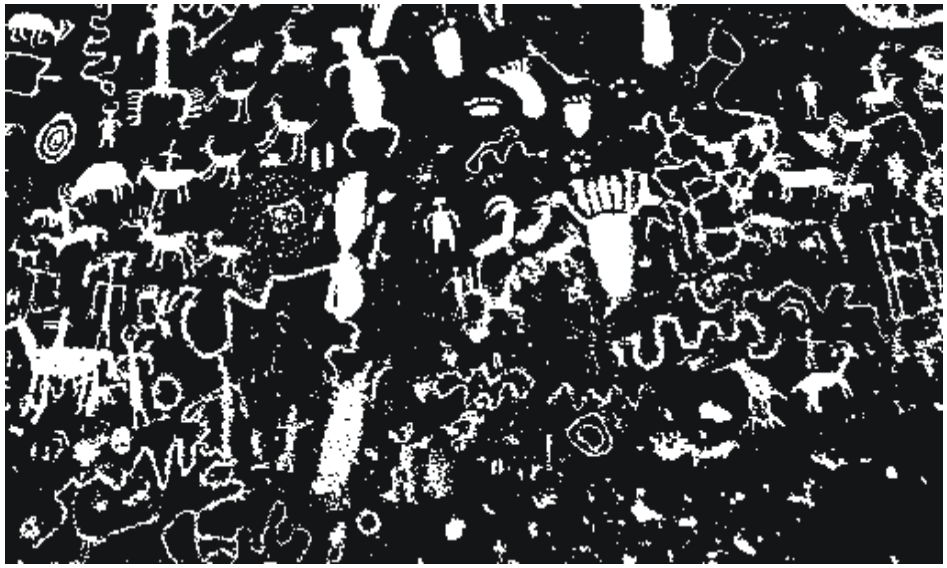


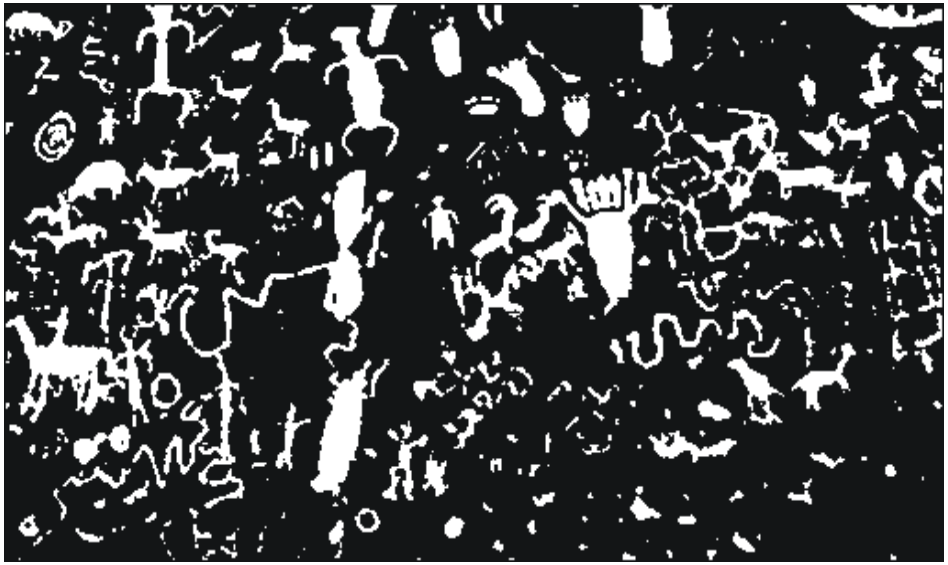


Medianfilterung auf Labeln

Wieso geht das überhaupt?
→ siehe Def. Medianfilter

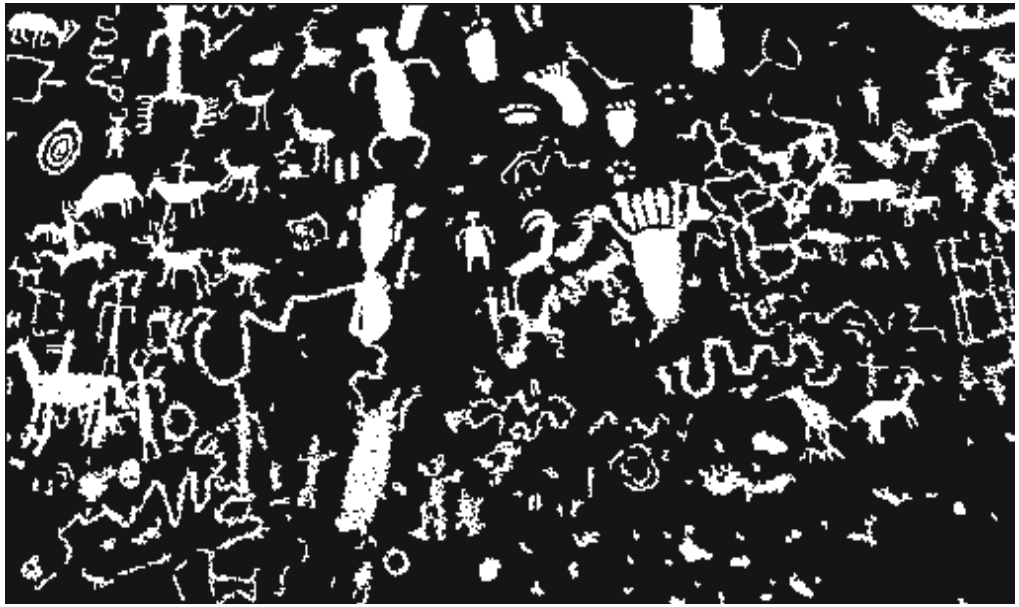






Entfernung kleiner Gebiete

- Morphologische Operationen



Relaxation Labeling

- Jedes Pixel p_i erhält jedes Label l_k mit einer initialen Wahrscheinlichkeit $P(p_i,l_k)$
  - Beispiel: Schwellenwertresultat gegeben;
    weiße Pixel: $P(p_i,\text{“weiß“})=0.8$, $P(p_i,\text{“schwarz“})=0.2$;
    schwarze Pixel: $P(p_i,\text{“weiß“})=0.2$, $P(p_i,\text{“schwarz“})=0.8$
- Benachbarte kompatible Pixel unterstützen sich
- **Relaxationsprozess:** Zuordnungswahrscheinlichkeiten ändern sich mit dem Maß der Unterstützung
- Am Ende für jedes Pixel das Label mit höchster Wahrscheinlichkeit auswählen
- Zu definieren: Kompatibilität, Einfluss der Kompatibilität auf die Label-Wahrscheinlichkeiten

Labelwahrscheinlichkeit

- Aufzählung aller Pixel in Liste $p_0, p_1, \ldots, p_{NM-1}$
- K verschiedene Label
- Initiale Label-Wahrscheinlichkeit $P^{(0)}$ für jedes Pixel, und jedes Label l_k vergeben, z.B.

$$P^{(0)}(p_i, l_k) = \begin{cases} 0.8 & , \text{falls } l_k = l(p_i) \\ 0.2 & , \text{sonst} \end{cases}$$

Segmentierungsverfahren hat Label $l(p_i)$ vergeben

Label-Wahrscheinlichkeit darf nicht 0 oder 1 sein (Gewissheiten werden nicht verändert)

Kompatibilität

- Label l_k von Pixel p_i hat Kompatibilität r mit Label l_l von Pixel p_j: $r\left(\left(p_i, l_k\right), \left(p_j, l_l\right)\right)$

- Kompatibilitätskoeffizient für Binärbilder (K=2 Label) z.B.

$$r\left(\left(p_i, l_k\right), \left(p_j, l_l\right)\right) = r\left(l_k, l_l\right) = \begin{cases} 1 & , \text{falls } l_k = l_l \\ 0 & , \text{sonst.} \end{cases}$$

  - (d.h., nur gleiche Label unterstützen sich)

- Ideal: r ist bedingte Wahrscheinlichkeit, dass Pixel p_i label l_k trägt, wenn Pixel p_j Label l_l trägt

$$r\left(\left(p_i, l_k\right), \left(p_j, l_l\right)\right) = P\left(p_i, l_k \mid p_j, l_l\right)$$

Unterstützung eines Pixels

- Unterstützung $q^{(n)}$ von Label l_k des Pixel p_i durch Pixel p_j zur Iteration n

$$q_j^{(n)}(p_i, l_k) = \sum_{l=0}^{K-1} P^{(n)}(p_j, l_l) \cdot r((p_i, l_k), (p_j, l_l))$$

- Unterstützung von p_i durch alle Pixel

$$Q^{(n)}(p_i, l_k) = \sum_{j=0}^{NM-1} c_{ij} q_j^{(n)}(p_i, l_k)$$

mit Einflussparameter c_{ij}, z.B.

$$c_{ij} = \begin{cases} \frac{1}{8} & , \text{falls } p_j \in N_8(p_i) \\ 0 & , \text{sonst} \end{cases}$$

Iterationsschritt

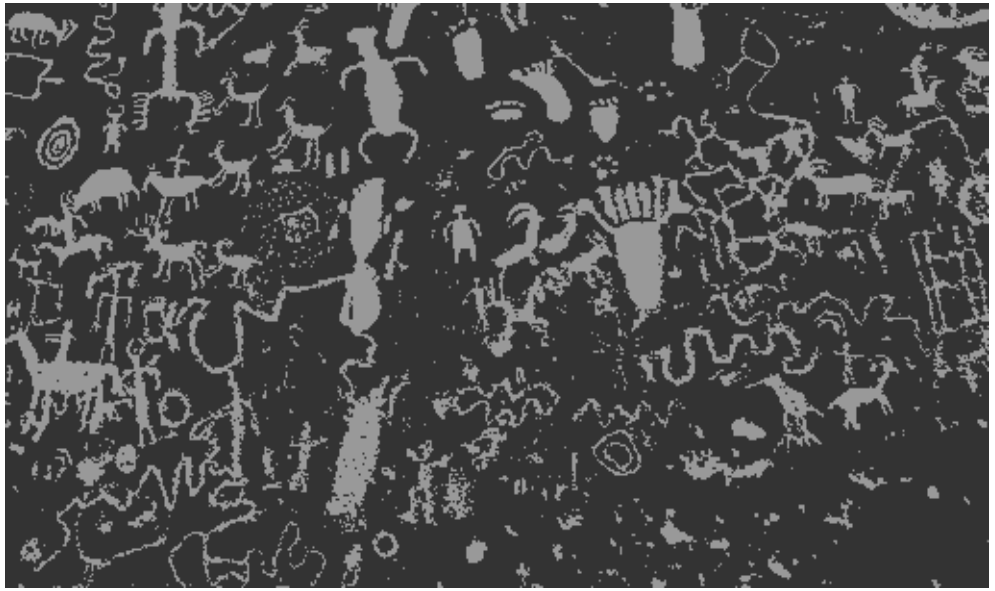
- Wahrscheinlichkeit, dass Pixel p_i Label l_k trägt, wurde (in Iteration n) bereits berechnet
- Label-Wahrscheinlichkeit der Iteration n+1

$$P^{(n+1)}\left(p_i, l_k\right) = \alpha \frac{P^{(n)}\left(p_i, l_k\right)\left[1 + Q^{(n)}\left(p_i, l_k\right)\right]}{\sum_{l=0}^{K-1} P^{(n)}\left(p_i, l_l\right)\left[1 + Q^{(n)}\left(p_i, l_l\right)\right]}$$

- Nenner normiert über alle Label von Pixel p_i
- α: Schrittweitenparameter, z.B. α = 1

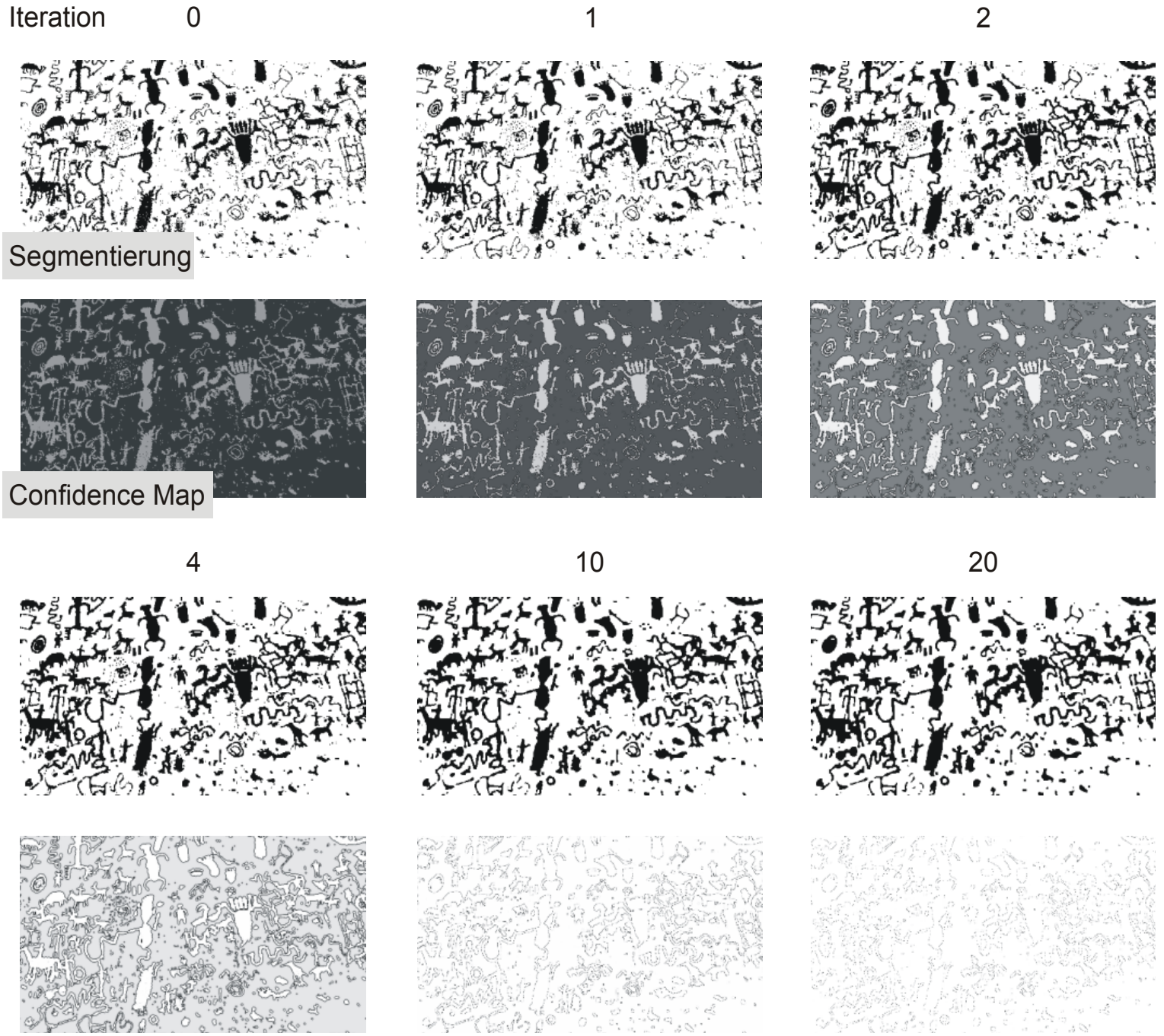
Confidence Map

- Confidence Map: Gibt für jedes Pixel die Zuverlässigkeit der derzeitigen Entscheidung an (weiß = sicher)
  - Confidence Map erlaubt Beobachtung des Konvergenzverhaltens
- Quotient der Wahrscheinlichkeiten des wahrscheinlichsten Labels zum zweitwahrscheinlichsten Label
  - dunkel
    = kleiner Quotient
    = geringes Vertrauen
  - hell
    = großer Quotient
    = hohes Vertrauen



Initiale Confidence Map

Konvergenz

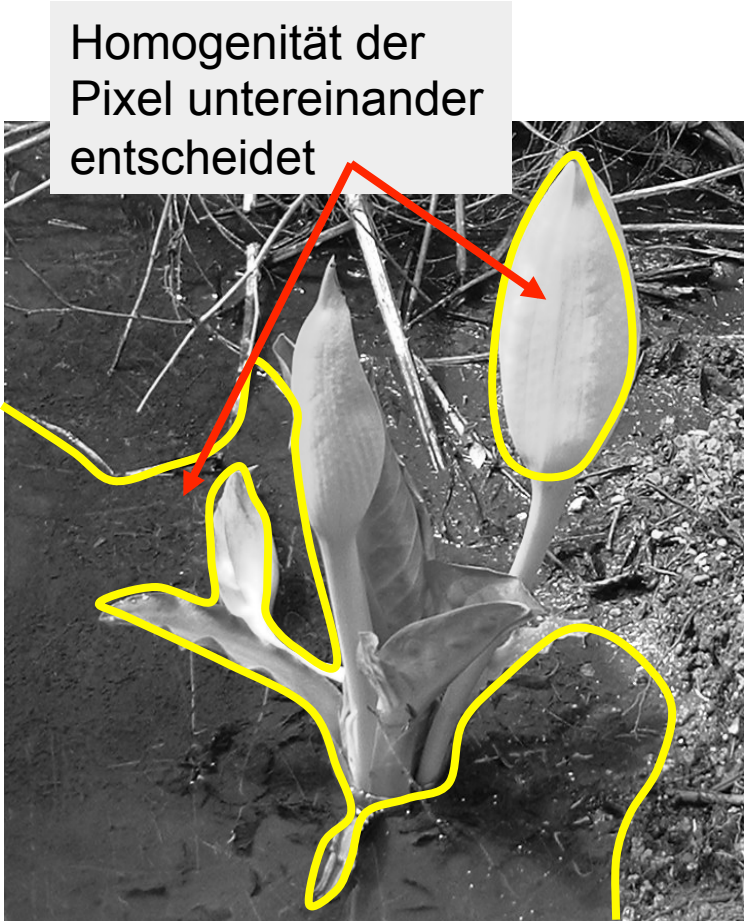


Regionenbasierte Segmentierung

- Multiskalenstrategien
- Region Merging
- Split-and-Merge
- Textursegmentierung

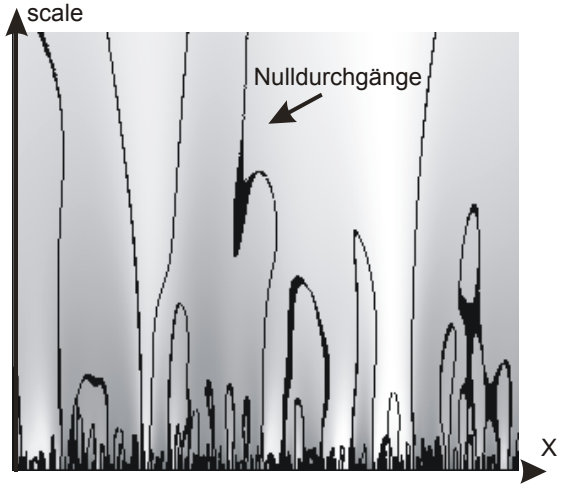
Regionenbasierte Segmentierung

- Homogenität im Inneren des Segments
- Homogenitätsbedingung wird bei der Segmentierung ausgewertet
- Homogenität ist relativ zu den Attributen eines Segments definiert
- Globale Zusammenhänge über Multiskalenstrategie



Multiskalenstrategie

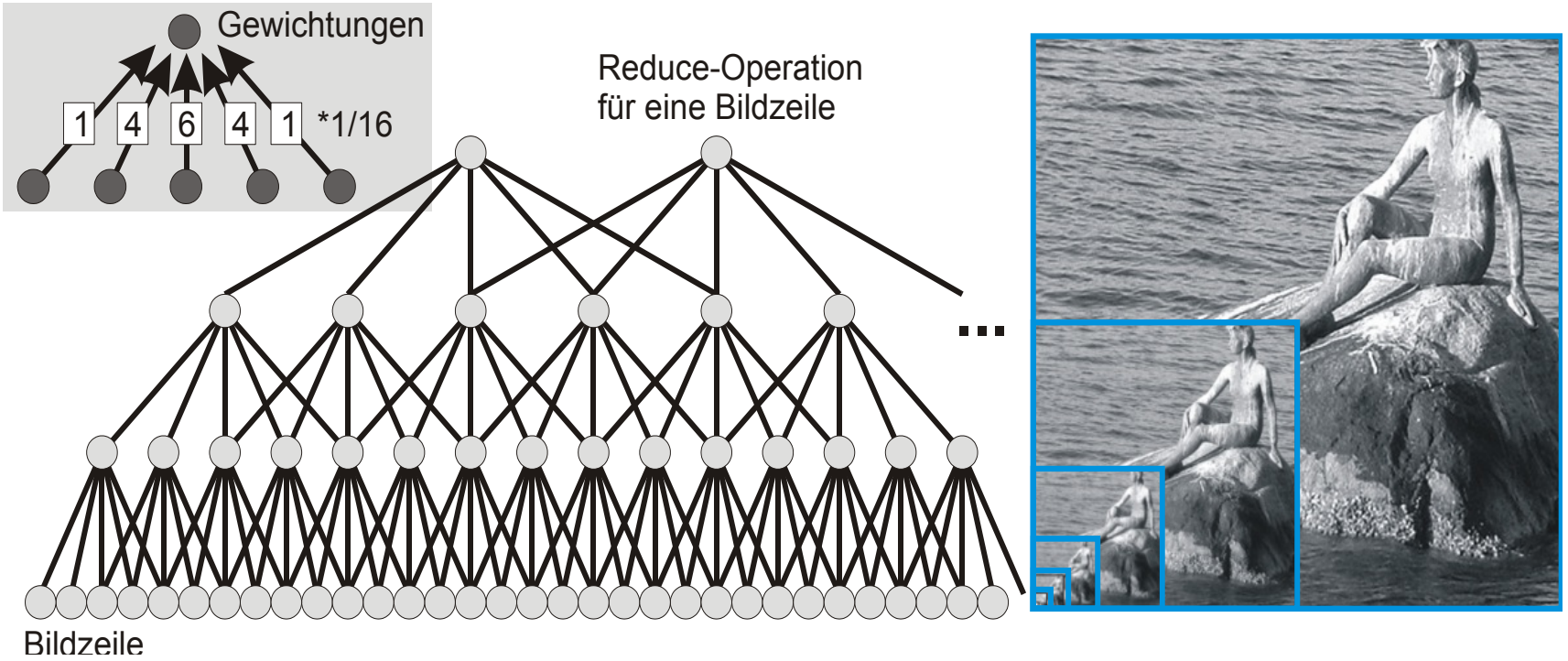
- Relative Kriterien für Homogenität können über unterschiedliche Entfernungen verschieden wirken
- Segmentierung nach Multiskalenstrategie wertet Kriterien auf unterschiedlichen Skalierungen aus



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

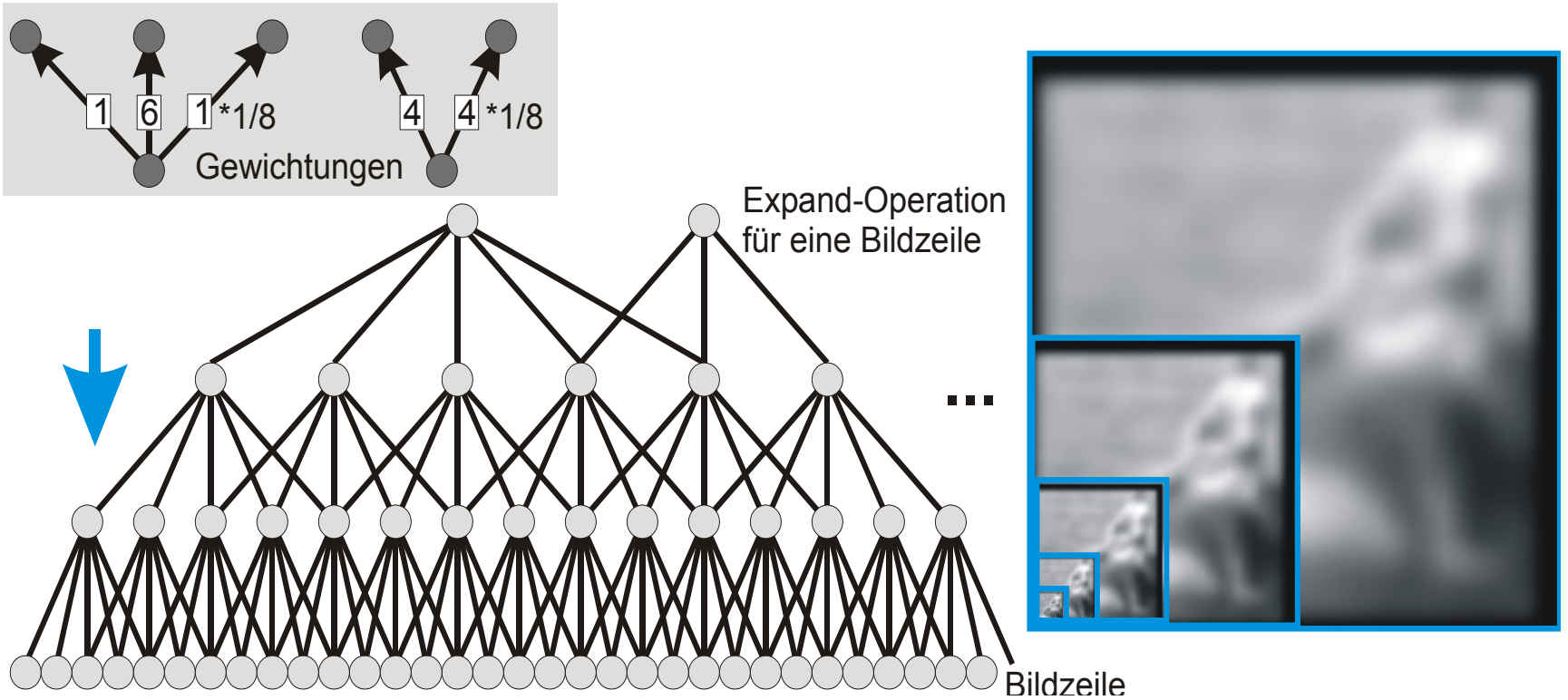
- Um die vorherige Skalierungsstufe zu erzeugen, wird eine expand-Operation definiert
- Pixel der neuen Skalierungsstufe werden durch Interpolation erzeugt
  - Pixelorte, die auf beiden Skalierungsstufen existieren:

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

  - 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

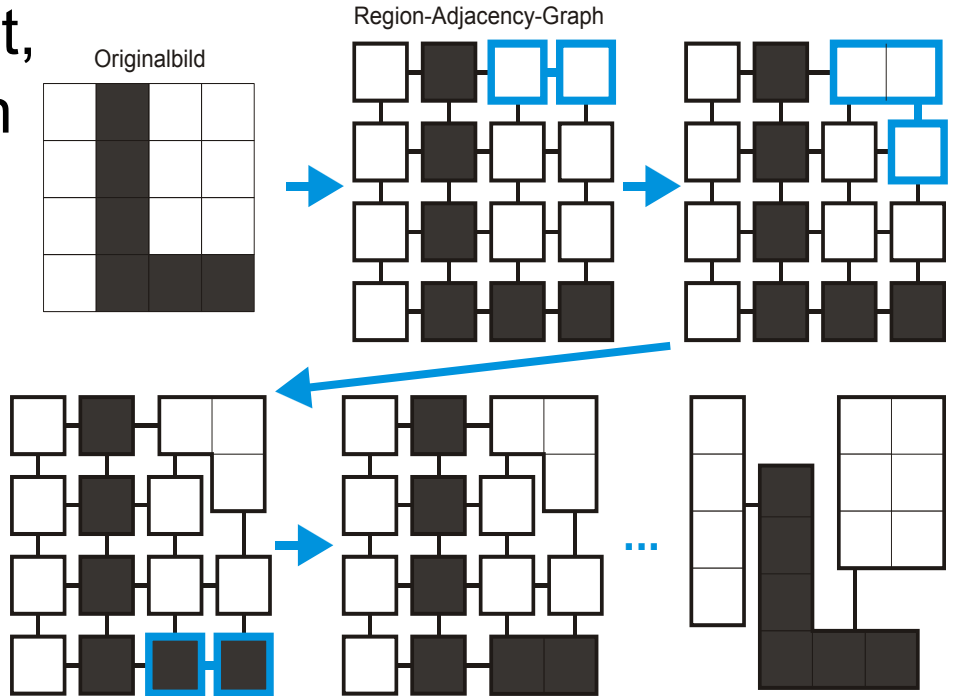
Laplace-Pyramide

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

expand(reduce(image))-image

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

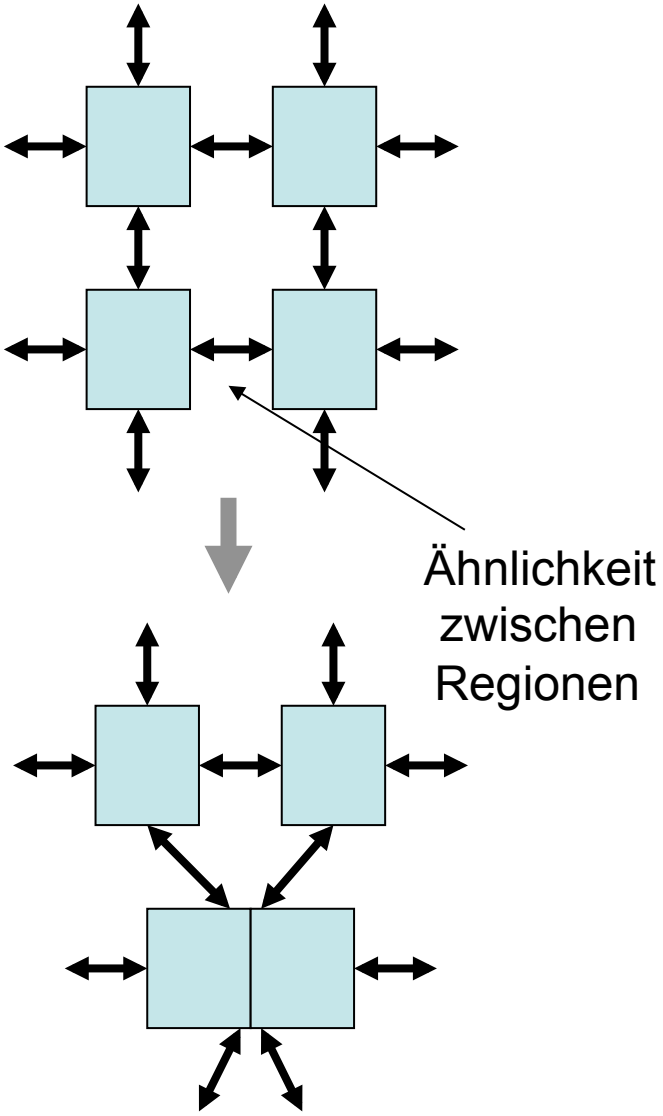
„von Pixeln zu Regionen“:

```
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

- Homogenitätskriterium
  - Grauwertdifferenz innerhalb der Region
  - Wahrscheinlichkeit, dass beide Regionen die gleichen Grauwertverteilungen haben

Region Merging und Multiskalenstrategie

Modellannahme:

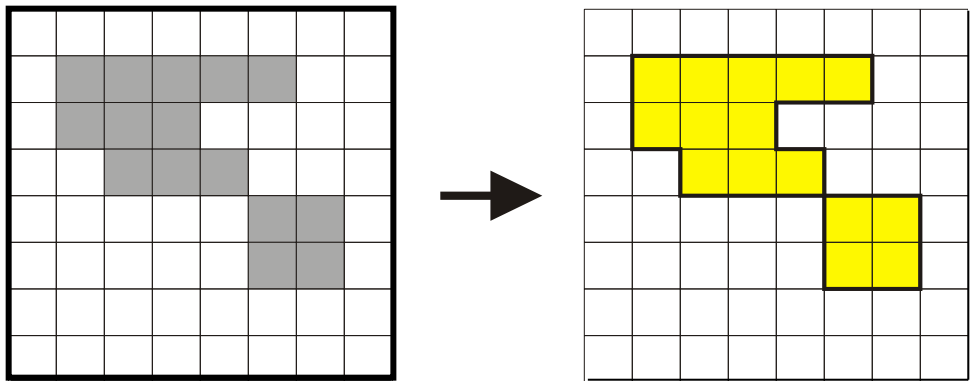
Die gröbste Skalierungsstufe, auf der sich segmentierungs-relevante Eigenschaften zeigen, ist bekannt

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

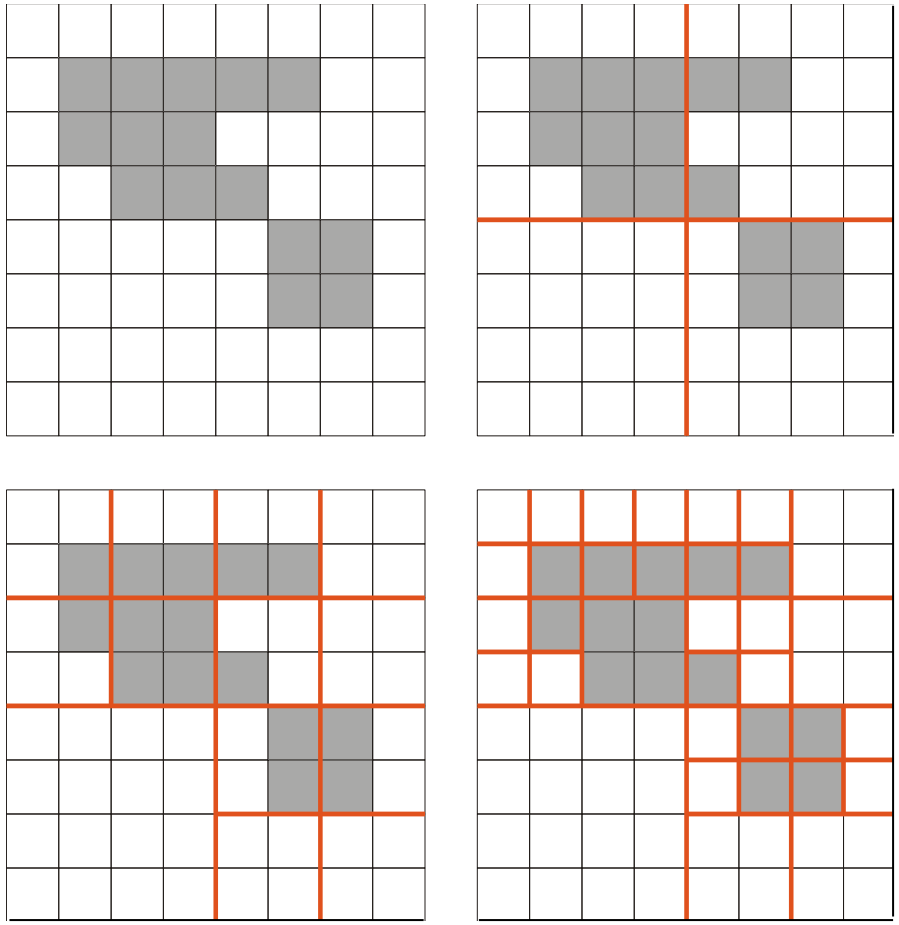
Split & Merge-Algorithmus

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

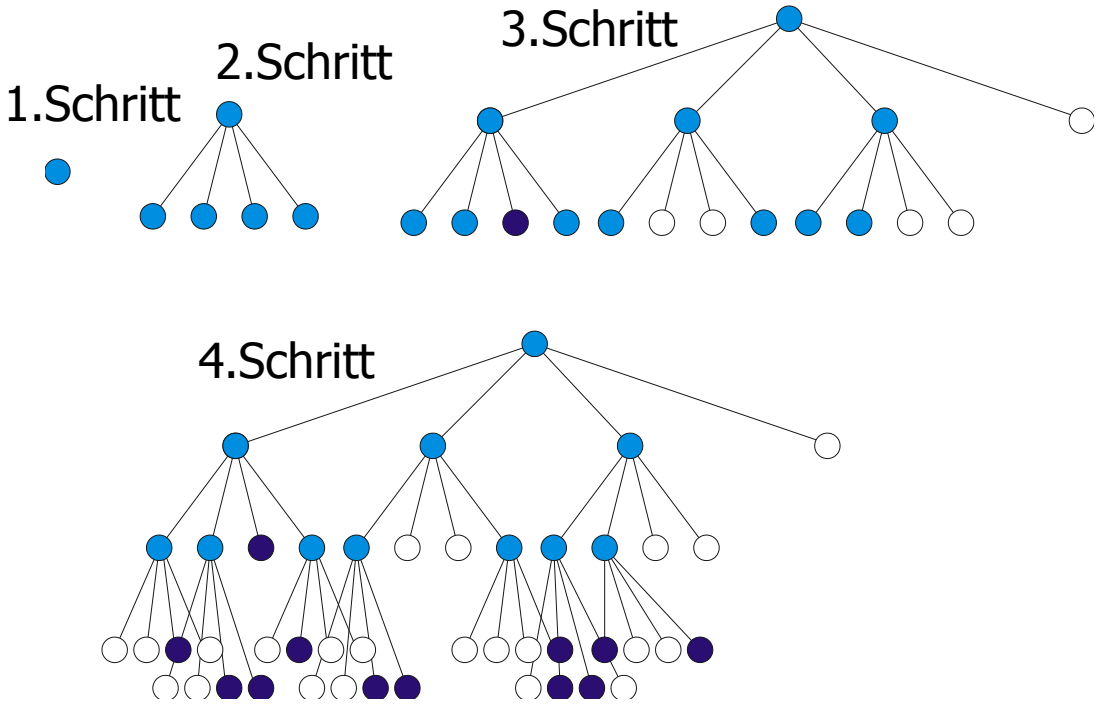


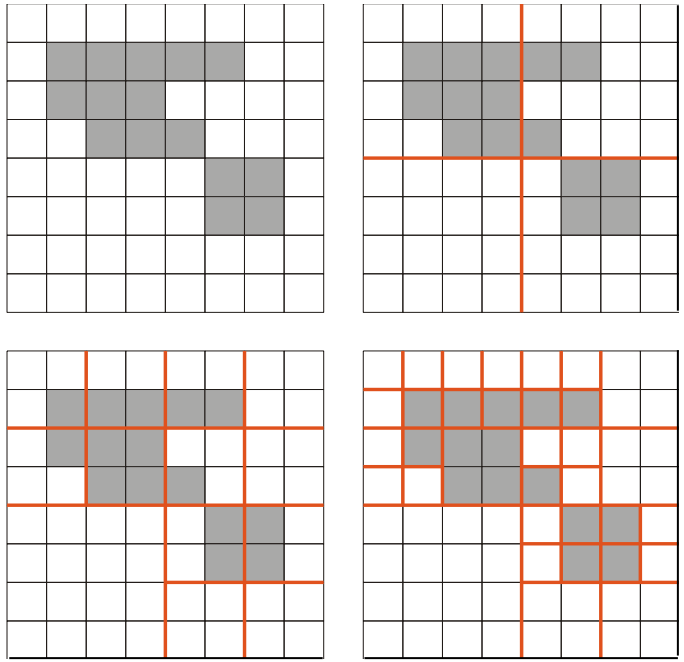
Zerlegung terminiert spätestens auf Pixelebene

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

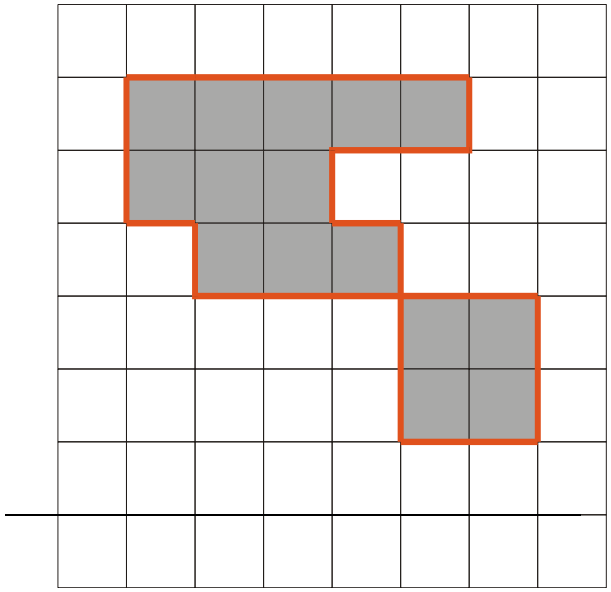


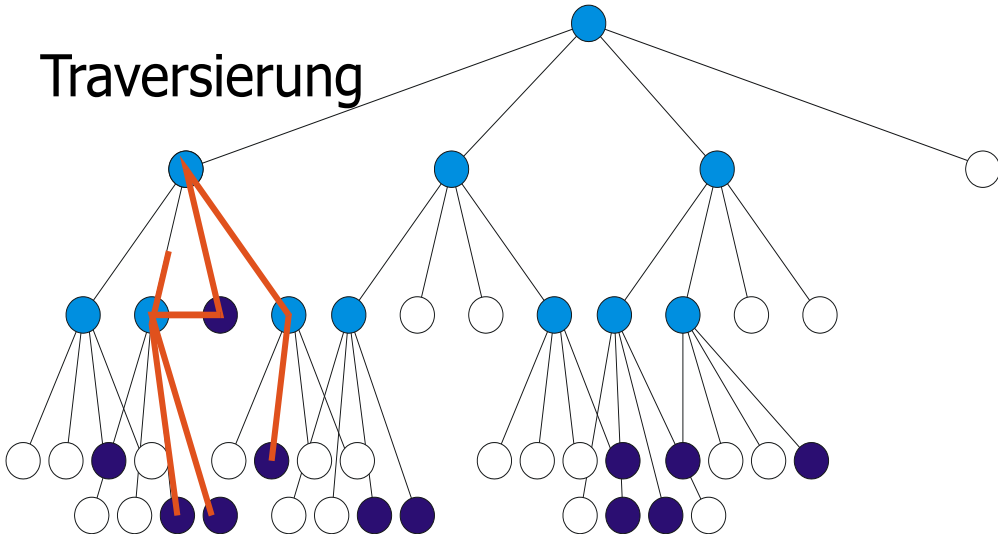


Merging

Quadtree wird traversiert und in einen RAG überführt

Auf dem RAG wird ein Region Merging durchgeführt



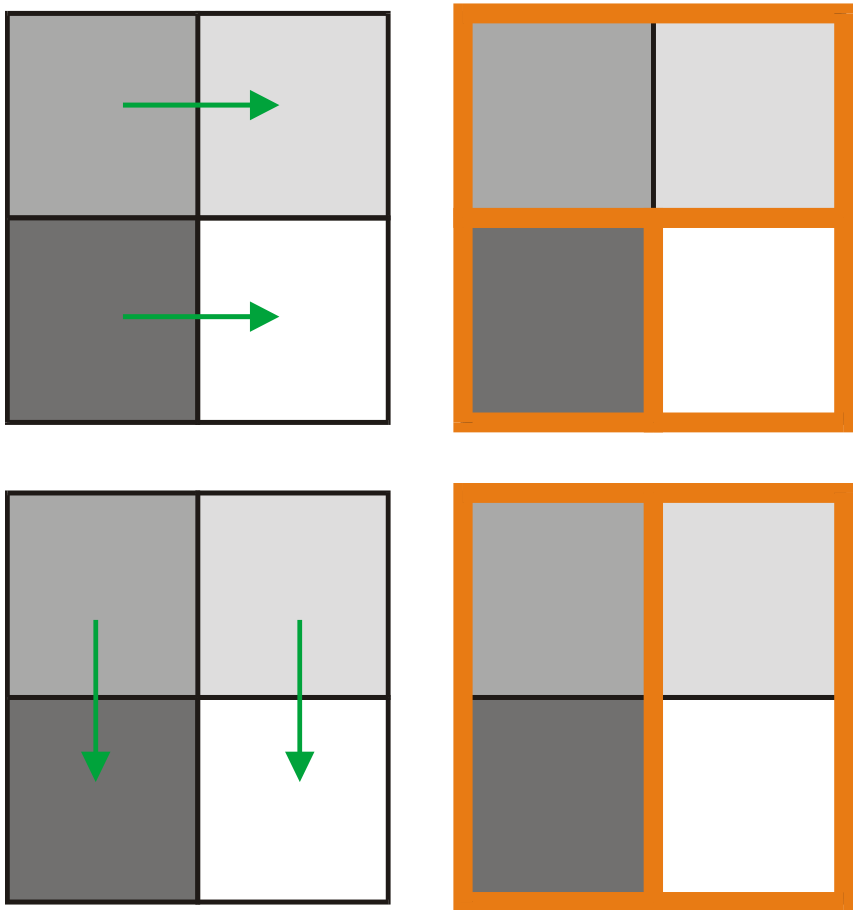


Split & Merge

Resultat: Zerlegung des gesamten Bilds in Regionen

Multiskalenstrategie ist integriert

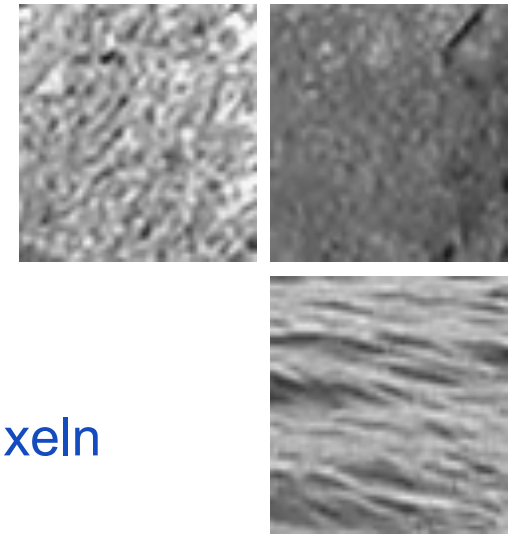
Homogenitätsmerkmale wie bei Region Merging



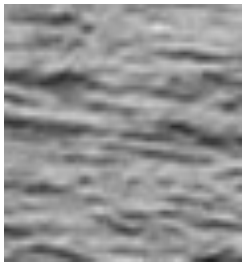
Probleme (Region Merging und Split & Merge):

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

Textur als Homogenitätsmerkmal



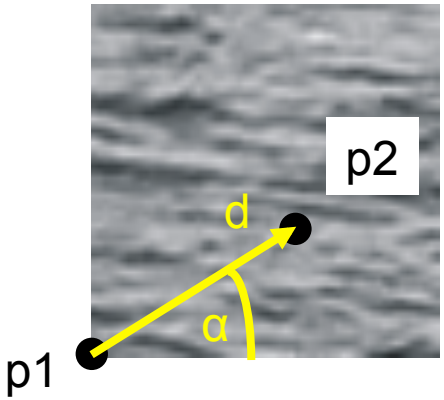
- Textur – Musterung der Oberfläche
  - Es existiert keine Definition von Textur
  - Es gibt eine große Anzahl von Texturmaßen
  - Textur ist inhärent skalenabhängig
  - Textur ist eine Eigenschaft einer Gruppe von Pixeln
- Texturmaße
  - strukturell (Zusammensetzung aus Texturelementen – texeln)
  -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

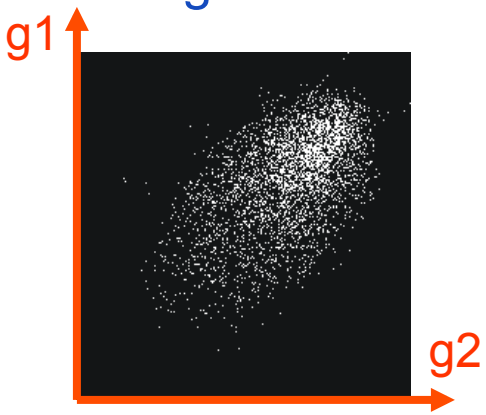


Haralick'sche Texturmaße

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





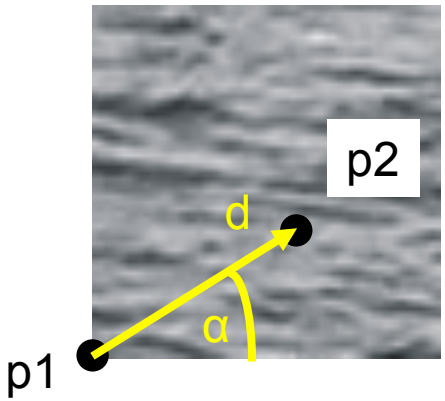
Haralick'sche Texturmaße

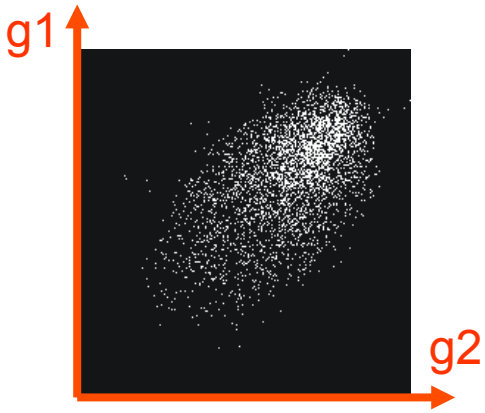
Robert Haralick

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

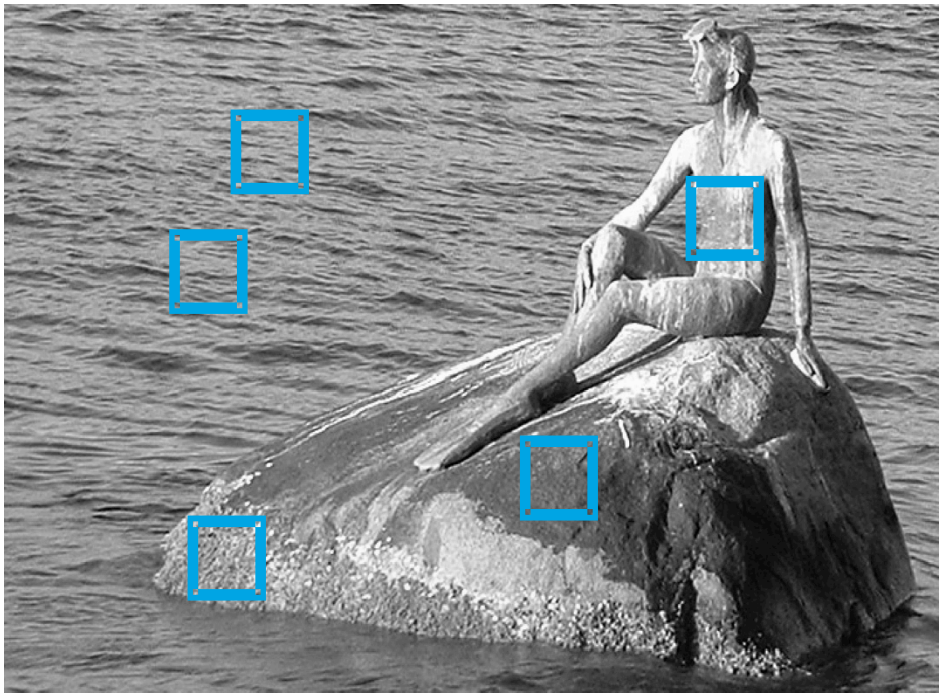
$$P_{\alpha,\Delta}(g_1,g_2)=\frac{1}{|R|}\sum_{\vec{x}\in R}\delta_D\left(f(\vec{x})-g_1\right)\cdot\delta_D\left(f(\vec{x}+\vec{d})-g_2\right)$$

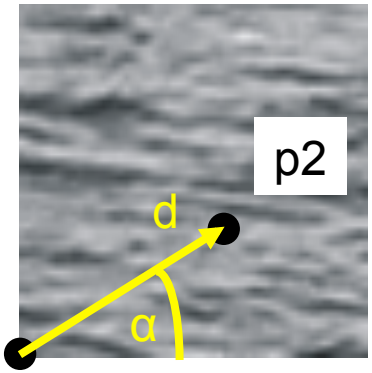
$$\vec{d}=\Delta\cdot\begin{pmatrix}\cos(\alpha)\\ \sin(\alpha)\end{pmatrix}$$

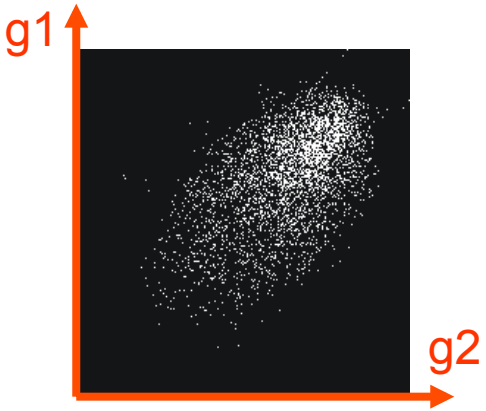


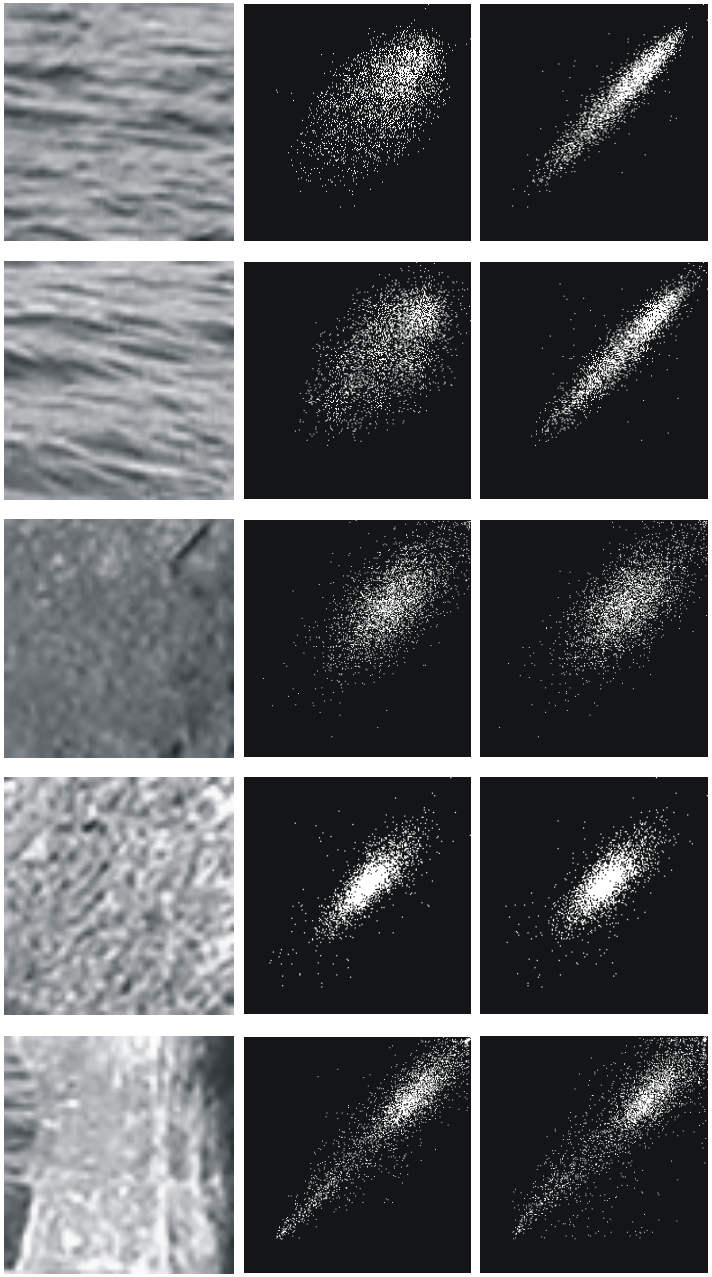


Co-Occurrence-Matrix









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

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

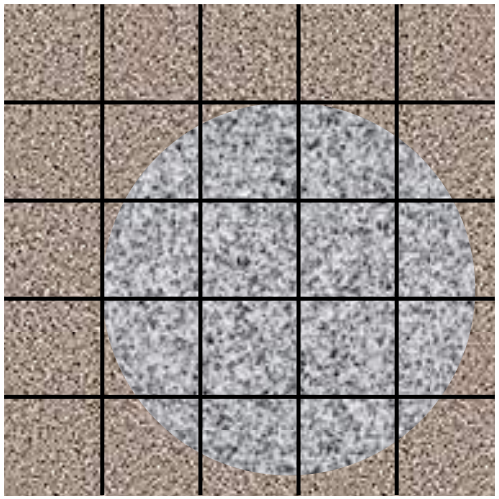
$$\mu_2 = \sum_{g_2=0}^{K-1} g_2 \sum_{g_1=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2) \qquad \sigma_2 = \sqrt{\sum_{g_2=0}^{K-1} (g_2 - \mu_2)^2 \sum_{g_1=0}^{K-1} P_{\Delta,\alpha}(g_1, g_2)}$$

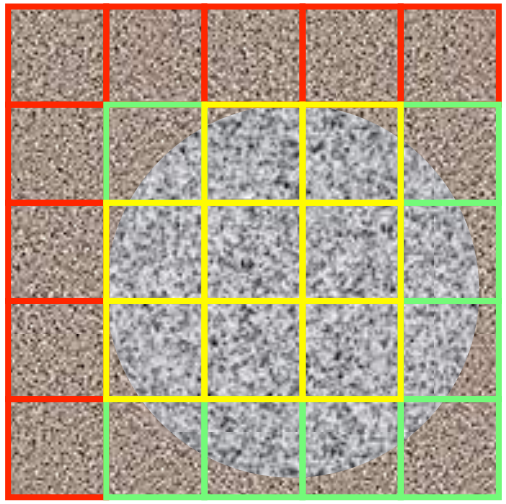
inverse difference moment $$\sum_{g_1=0}^{K-1} \sum_{g_2=0}^{K-1} \frac{P_{\Delta,\alpha}(g_1, g_2)}{1 + (g_1 - g_2)^2}$$

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

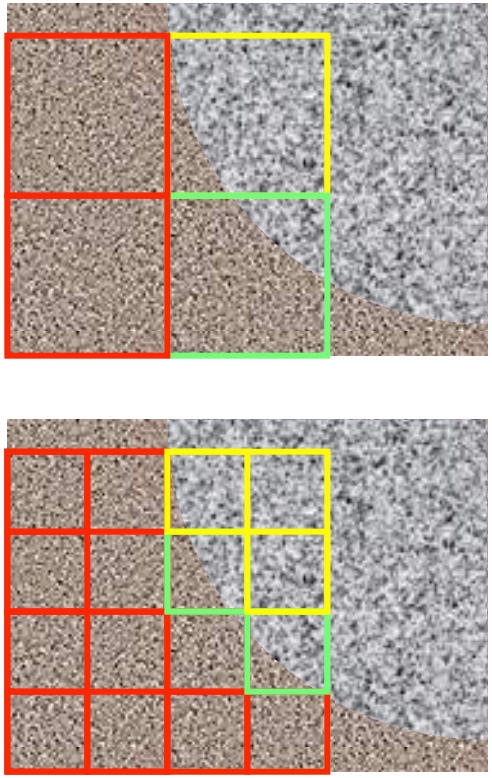
- Berechnung der Texturmerkmale auf der Basis willkürlicher Regionen
- Segmentierung
- Berechnung der Zuverlässigkeit
- Erneute Segmentierung





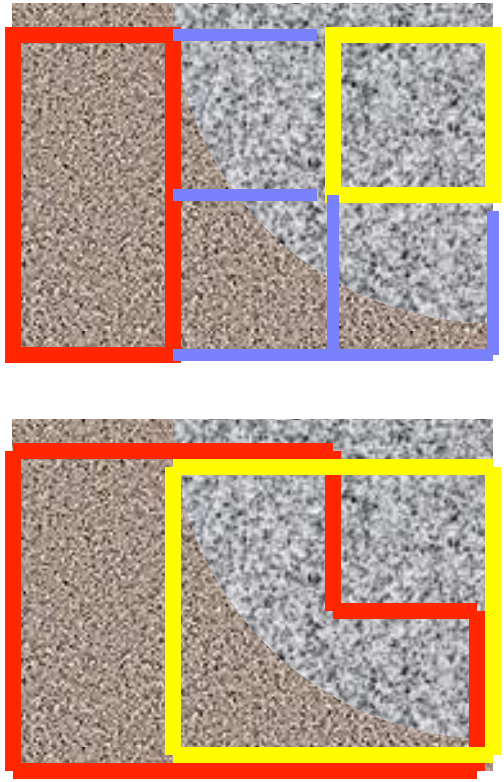
Zuverlässigkeit Texturmerkmal

- 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
  - ähnlichste Kombination selektieren
- Berechenbarkeit von Texturmerkmalen beliebiger Regionen ist vorteilhaft
- Kann über mehrere Auflösungsstufen erfolgen



Die beiden Extremfälle

Naive Textursegmentierung

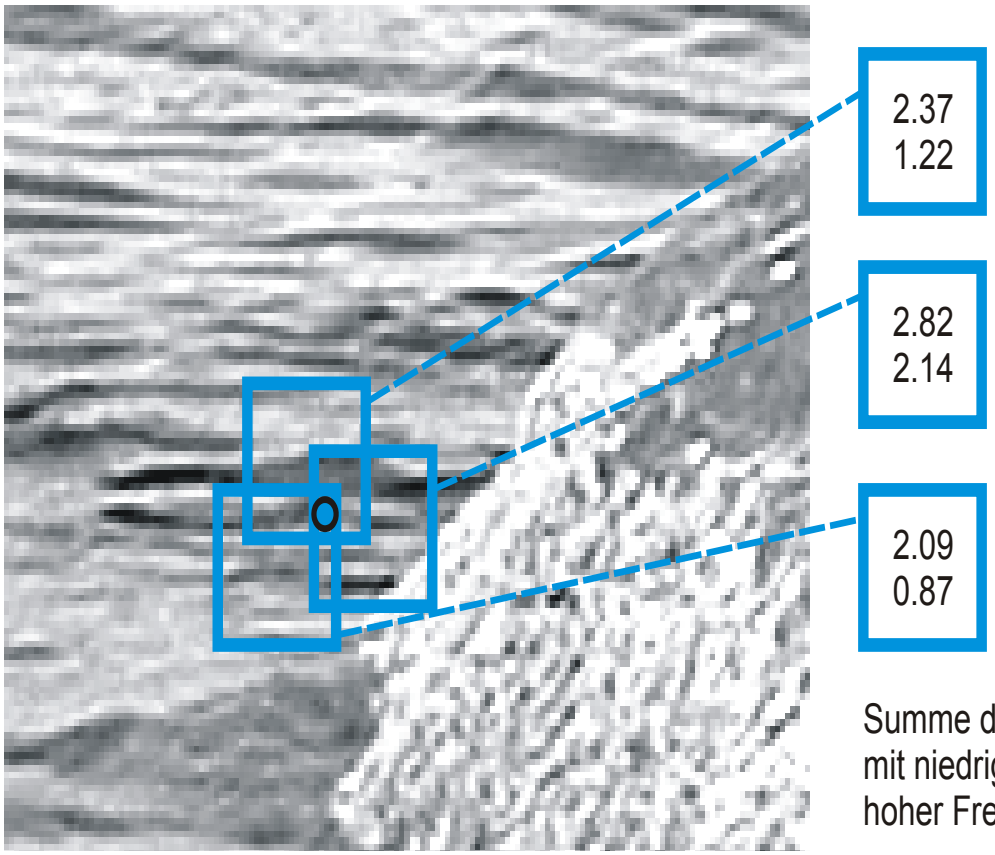
Modell:

- 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 Schwellenwertsegmentierung

Textursegmentierung (naiv)



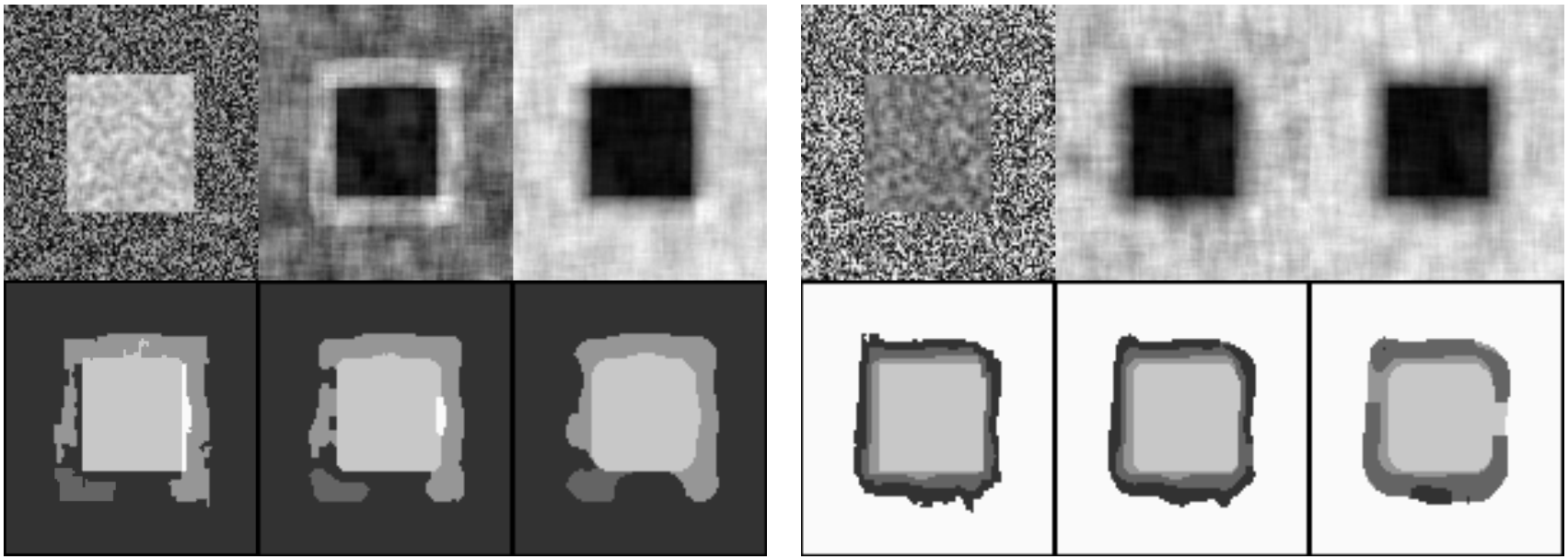
Beispiel:

Jedes Pixel wird von mehreren Blöcken überdeckt

Sind die Texturen unterschiedlich, dann erhalten diese Blöcke unterschiedliche Label

Nachverarbeitung Medianfilterung

Segmentierung von zwei verschiedenen Texturen



Resultat Median 5×5 Median 11×11 Resultat Median 5×5 Median 11×11

Nachverarbeitung Relaxation Labeling