

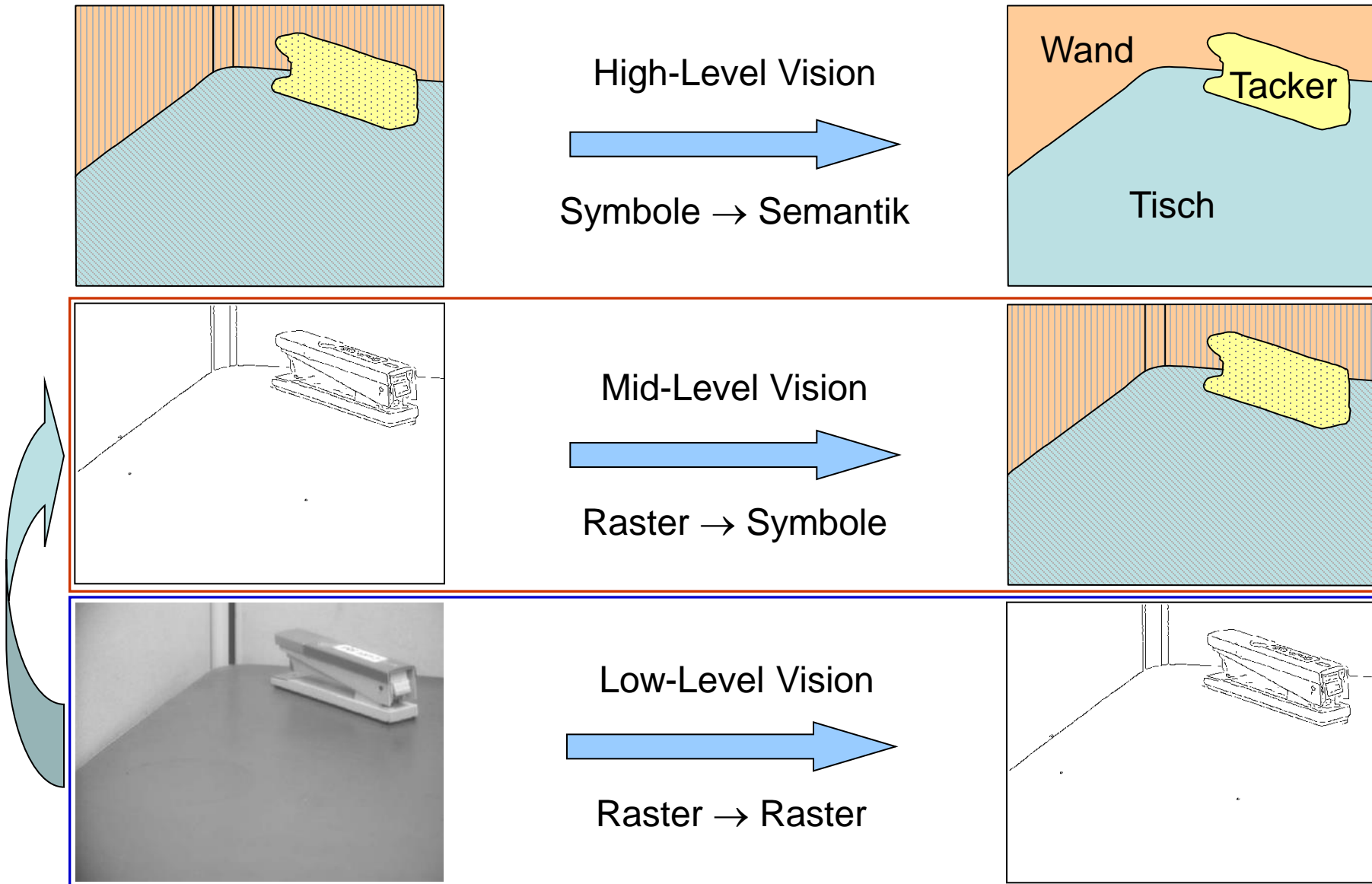
Intelligente Sehsysteme

5 Segmentierung

Histogrammbasierte, regionenbasierte und
texturbasierte Segmentierung

Florian Oßwald

Phasen des Computersehens (Wiederholung)



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

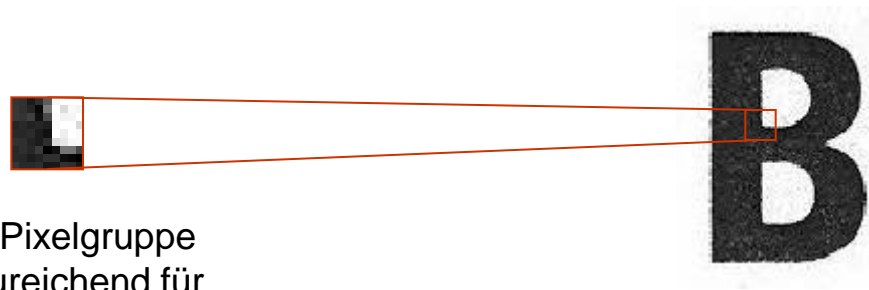
Inhalt

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

Segmentierung (1)

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

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



Lokale Pixelgruppe
ist unzureichend für
Interpretation

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

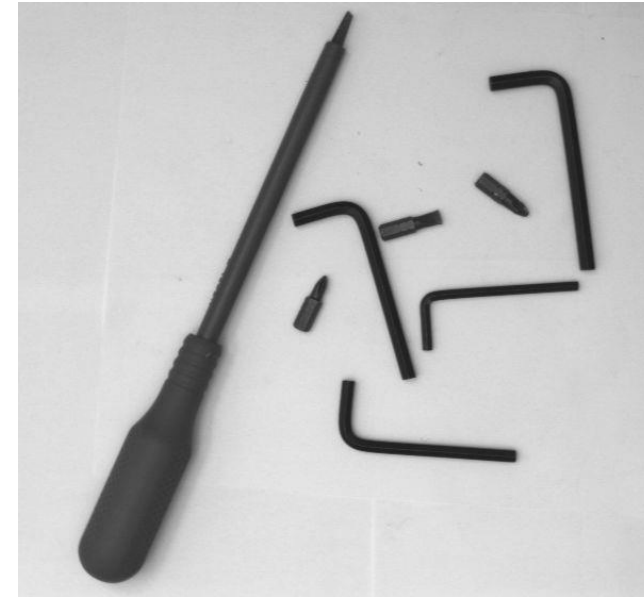
Segmentierung (2)

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

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

Informell (später genauer):

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



Segmentierung (3)

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

Def. [[Segmentierung](#), [Segmente](#)]

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

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

Segmentierung (4)

In dieser Vorlesung werden drei Ansätze der Segmentierung vorgestellt:

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

In den Folgevorlesungen werden behandelt:

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

Histogrammbasierte Segmentierung (1)

Einfachster Fall: „Segmentierung“ durch Histogrammanalyse

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



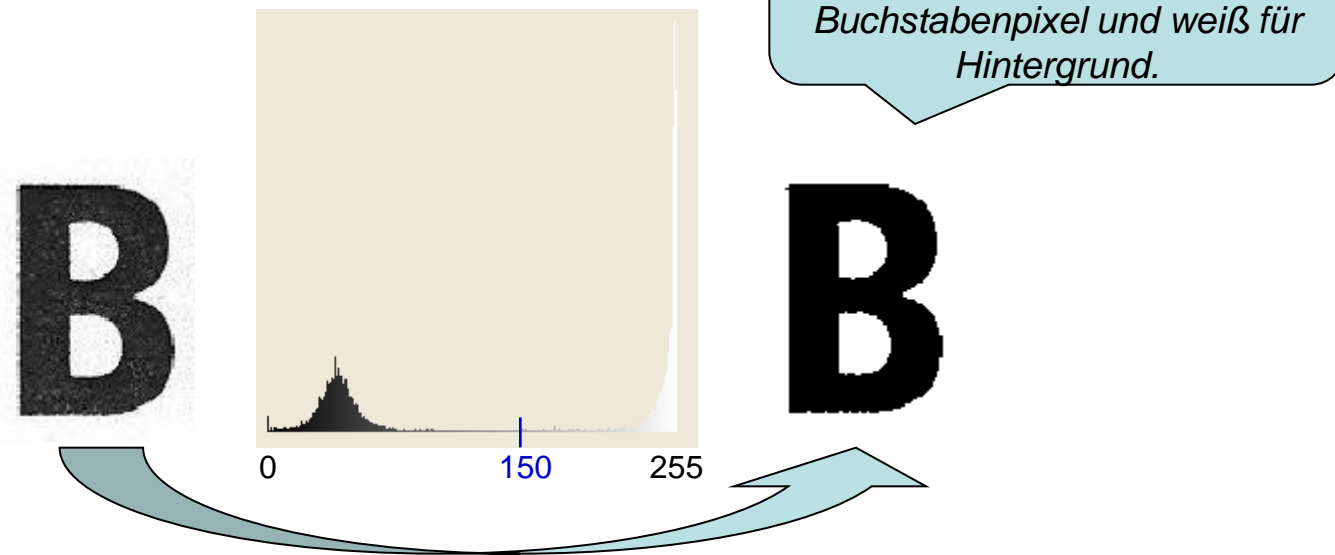
- ↪ das Histogramm ist ein **bimodales Histogramm** mit zwei ausgeprägten Maxima, dessen beide Modi für Vordergrund und Hintergrund stehen
- ↪ Eine Trennung aller Pixel (x,y) in Vordergrundpixel und Hintergrundpixel entspricht einer **binären Klassifikation über einen Schwellwert t_B** (engl. *threshold*). Diese binären Klassifikation wird im Computersehen als **Binarisierung** bezeichnet:

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

Histogrammbasierte Segmentierung (2)

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

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



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

Histogrammbasierte Segmentierung (3)

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

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

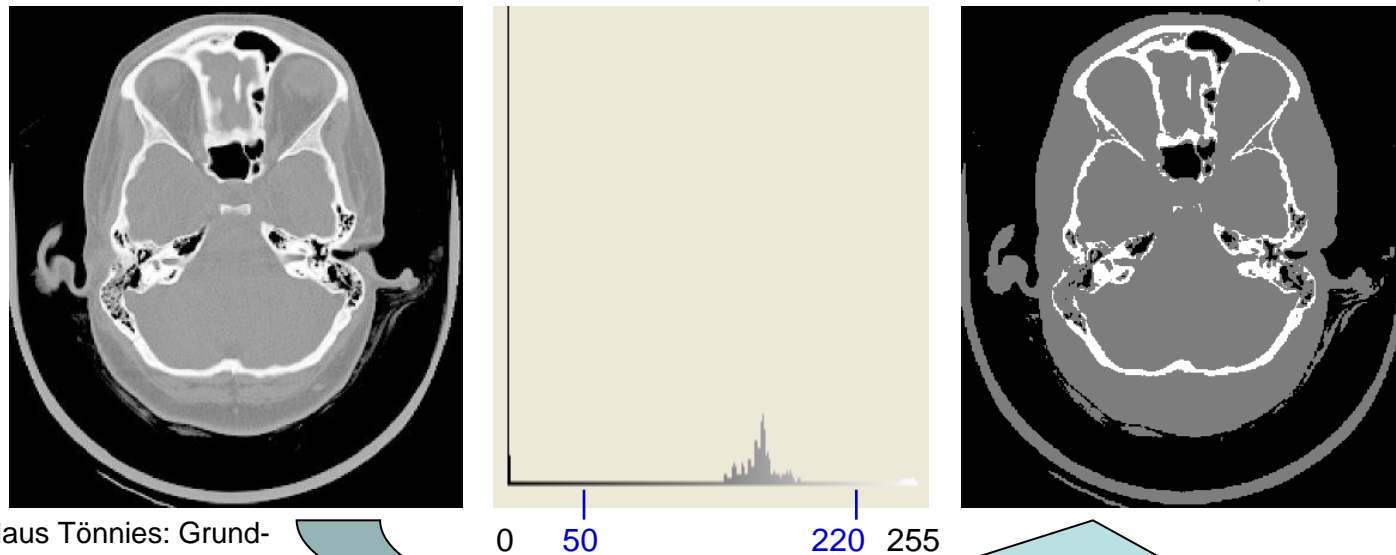
Wegen der schwellwertbasierten Zuordnung wird die histogrammbasierte Segmentierung auch als **Schwellwertsegmentierung** bezeichnet

Histogrammbasierte Segmentierung (4)

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

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

Visualisiert durch drei gut unterscheidbare Grauwerte



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

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

Histogrammbasierte Segmentierung (5)

Bestimmung der Schwellwerte:

- interaktive Bestimmung, die ggf. auch Anwendererfahrung nutzen kann
- automatisierte Suche nach lokalen Minima im Histogramm

Hier: die Clustering-Methode nach Otsu

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

Histogrammbasierte Segmentierung nach Otsu (1)

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

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

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

mit

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

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

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

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



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

*Hier dunkle Hintergrund-
pixel und helle Objektpixel*

Histogrammbasierte Segmentierung nach Otsu (2)

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

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

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

mit Gesamtvarianz σ^2 basiert nur auf Mittelwerten

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

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

Histogrammbasierte Segmentierung nach Otsu (3)

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

Zur Effizienzsteigerung der iterativen Berechnungen tragen bei:

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

n_T = Anteil der
Pixel \mathbf{p} mit $I(\mathbf{p}) = T$

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

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

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

Histogrammbasierte Segmentierung nach Otsu (1)

Beispiel:



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

Region Labeling (1)

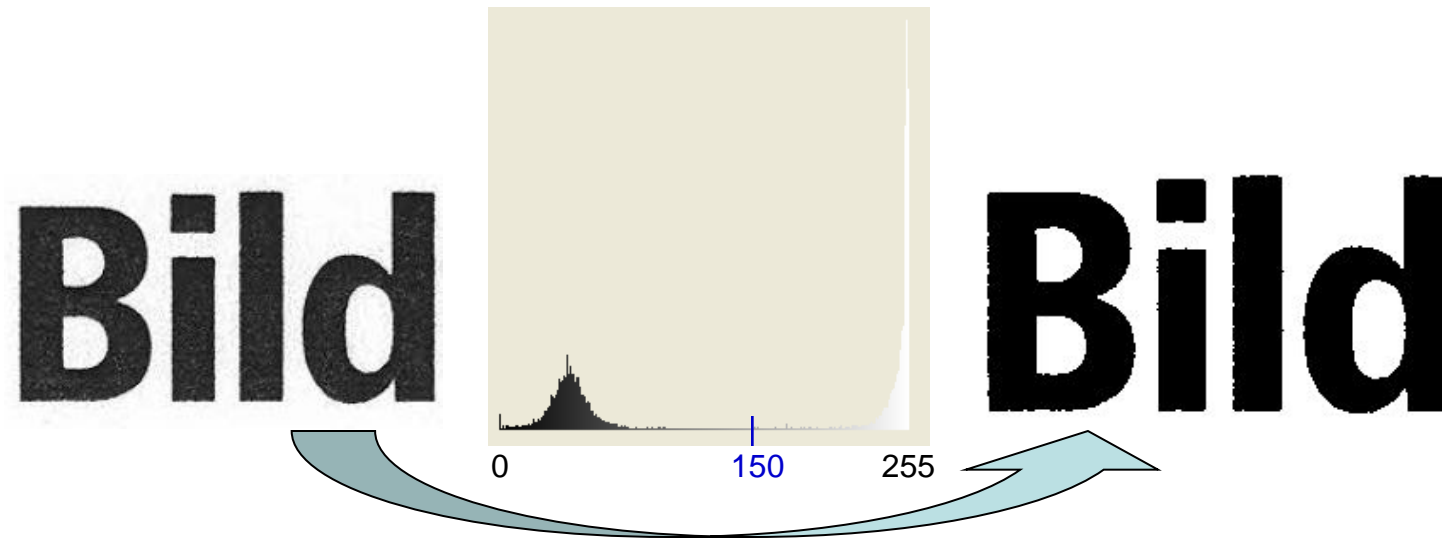
- Über die Schwellwertbildung erhalten **alle Pixel** in Abhängigkeit von ihrem Grauwert eine **Klassenzuordnung** (engl. *Label* \leadsto Klassen-Label)
- Jetzt sind **zusammenhängende Regionen** zu suchen, deren Pixel dasselbe Klassen-Label zeigen
 - Jede Region erhält ein neues, eigenes Regionen-Label,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

Region Labeling (2)

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

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

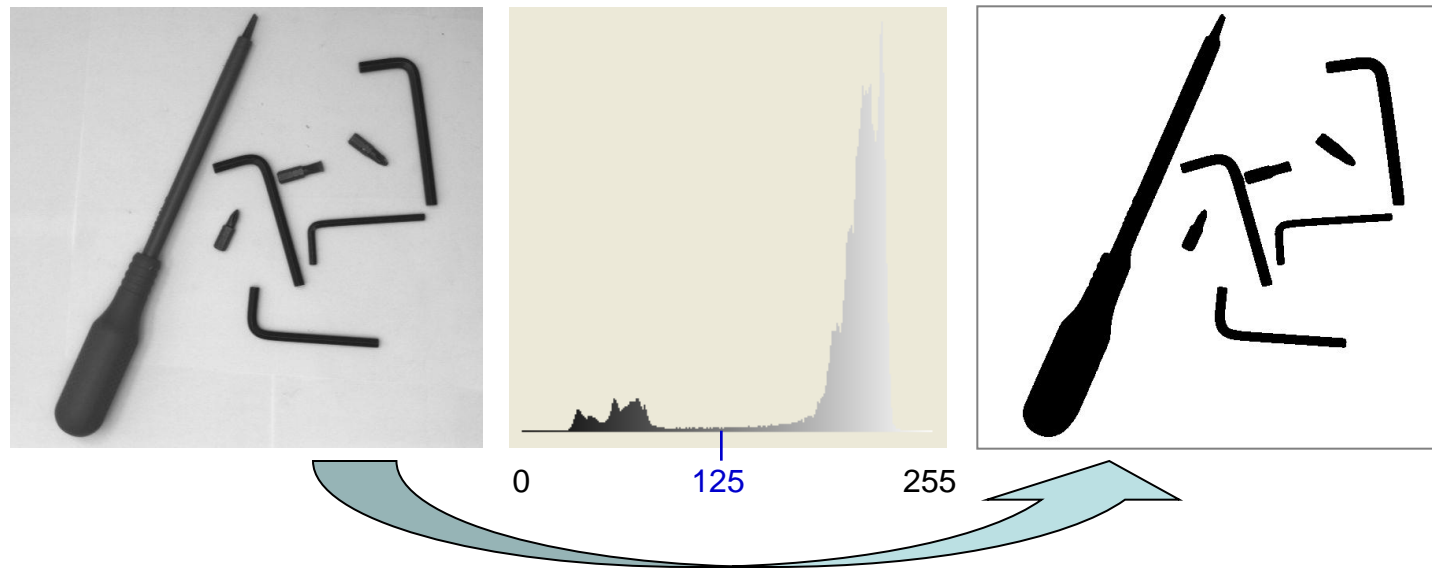
Diese so identifizierten Segmente sind dann Grundlage für Verfahren der High-Level-Vision. Hier z.B. einer Zeichen- und anschließenden Worterkennung.



Region Labeling (3)

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

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



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

Region Labeling (4)

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

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

Region Labeling (5)

Connected-component labeling (first part: labeling loop)

begin

$label \leftarrow 0;$

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

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

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

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

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

then begin

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

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

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

end if-then;

else begin

$label \leftarrow label+1;$

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

end else;

end if-then;

end for;

Region Labeling (6)

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

...

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

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

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

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

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

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

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

end for;

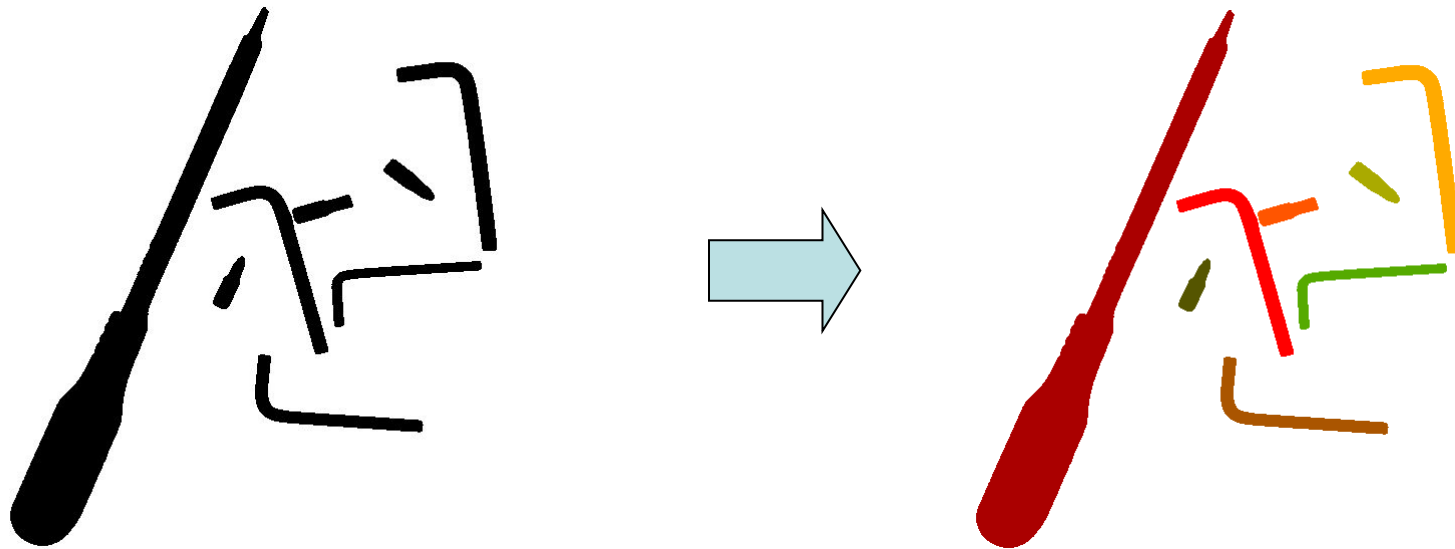
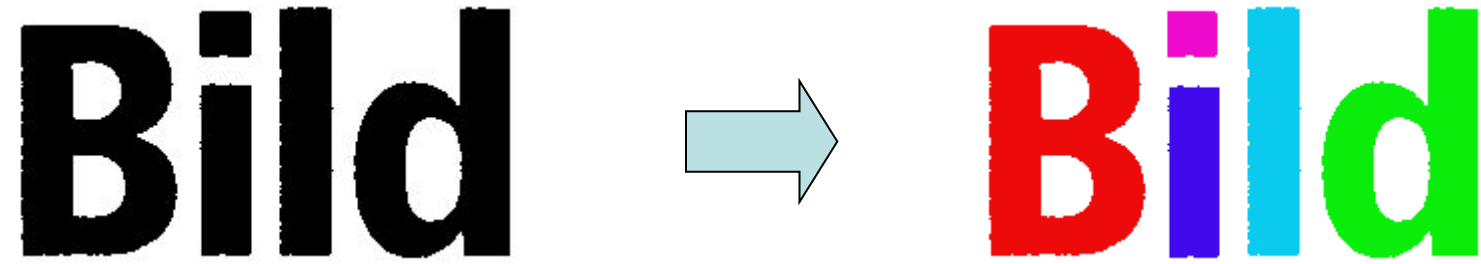
end; // Connected-component labeling

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

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

Region Labeling (7)

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



Region Labeling (8)

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

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

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

Region Labeling (9)

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

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

Vorteile der schwellwertbasierten Segmentierung

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

Grenzen der schwellwertbasierten Segmentierung

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

Segmentierung nach Homogenitätskriterien

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

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

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

Region Merging (1)

Region Merging als einfachster Segmentierungsansatz über Homogenitätskriterien:

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

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

Region Merging (2)

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

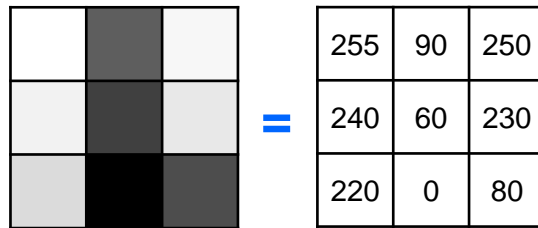
- Die **Knoten des RAG** repräsentieren die Regionen. Knotenwerte sind die für die Berechnung des Homogenitätskriteriums relevante Informationen.
- Eine **Kante des RAG** verbindet zwei Knoten gdw. die entsprechenden Regionen *direkt benachbart* sind. Kantenwerte geben den Homogenitätswert der Region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

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

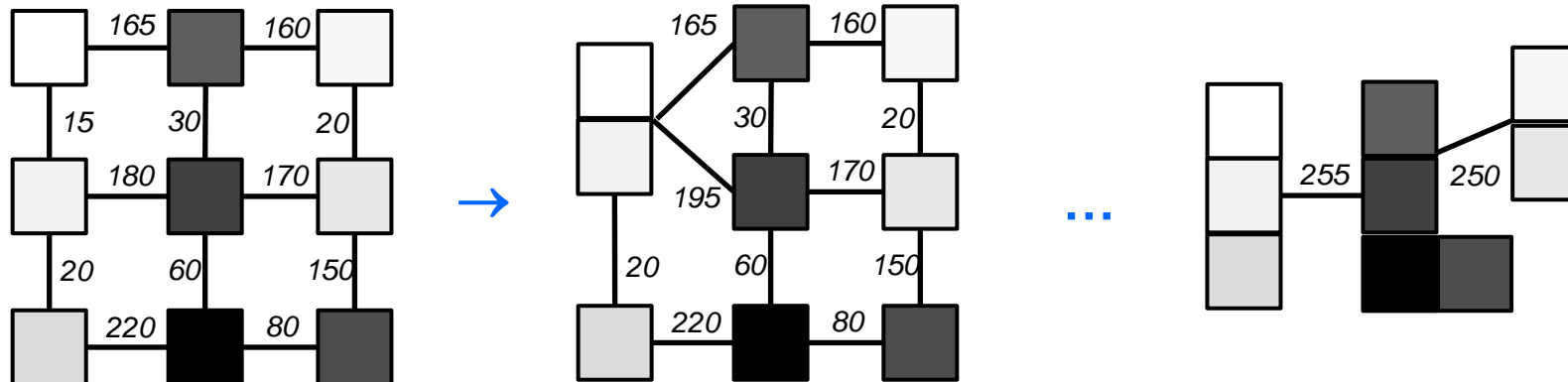
Region Merging (3)

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

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



Homogenitätskriterium: maximaler Intensitätsunterschied in jedem Segment ≤ 100



Optimalität von Region Merging (1)

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

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

Optimalität von Region Merging (2)

Beispiel für die Reihenfolgeabhängigkeit von Region Merging:

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

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

Split-and-Merge (1)

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

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

Split-and-Merge (2)

Split-and-Merge zeigt zwei Phasen:

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

Split-and-Merge (3)

Split-and-Merge arbeitet mit folgenden Schritten:

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

Eigenschaften von Split-and-Merge (1)

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

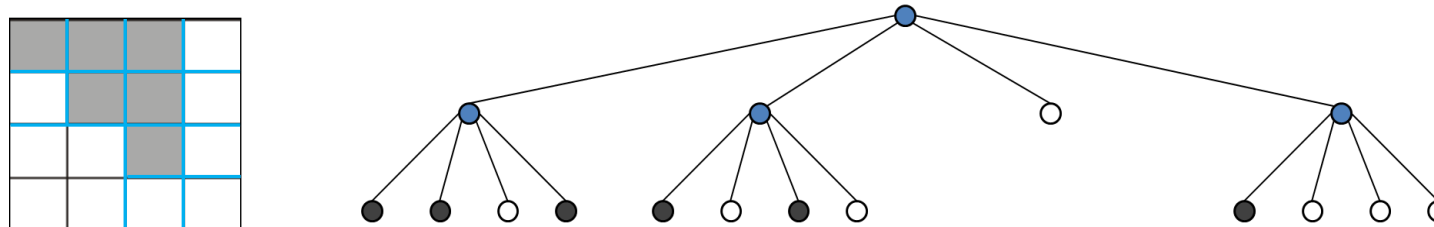
Eigenschaften von Split-and-Merge (2)

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

Split-and-Merge (4)

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

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



Region Merging, Split-and-Merge, Clustering

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

Dort wird innerhalb der sog. [hierarchischen Clusterverfahren](#) zwischen agglomerativen und divisiven Verfahren unterschieden.

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

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

Texturbasierte Segmentierung (1)

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

Verschiedene Regionen können sich danach auf zwei Arten unterscheiden:

(1) die Parametrisierung derselben Bildungsregel ist verschieden,

(2) die Bildungsregeln sind verschieden.

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

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

Texturbasierte Segmentierung (2)

Es gibt vielfältige Ansätze zur Definition von Textur :

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

Texturmaße nach Haralick

Die **Texturmaße nach Haralick** charakterisieren Intensitätsverteilungen auf Grundlage der sog. *Co-Occurrence-Matrix*.

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

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

Co-Occurrence-Matrix

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

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

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

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

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

Haralicksche Texturmaße (1)

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

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

Die wichtigsten Haralickschen Texturmaße:

- Energie/Uniformität: $\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} p_{\Delta,\alpha}^2(I_1, I_2),$
- Kontrast: $\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} (I_1 - I_2)^2 p_{\Delta,\alpha}(I_1, I_2),$
- Entropie: $-\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} p_{\Delta,\alpha}(I_1, I_2) \cdot \log_2(p_{\Delta,\alpha}(I_1, I_2)),$
- Homogenität/
inverse Differenz: $\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} \frac{p_{\Delta,\alpha}(I_1, I_2)}{1 + |I_1 - I_2|},$
- Inv. Diff.-Moment: $\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} \frac{p_{\Delta,\alpha}(I_1, I_2)}{1 + (I_1 - I_2)^2}.$

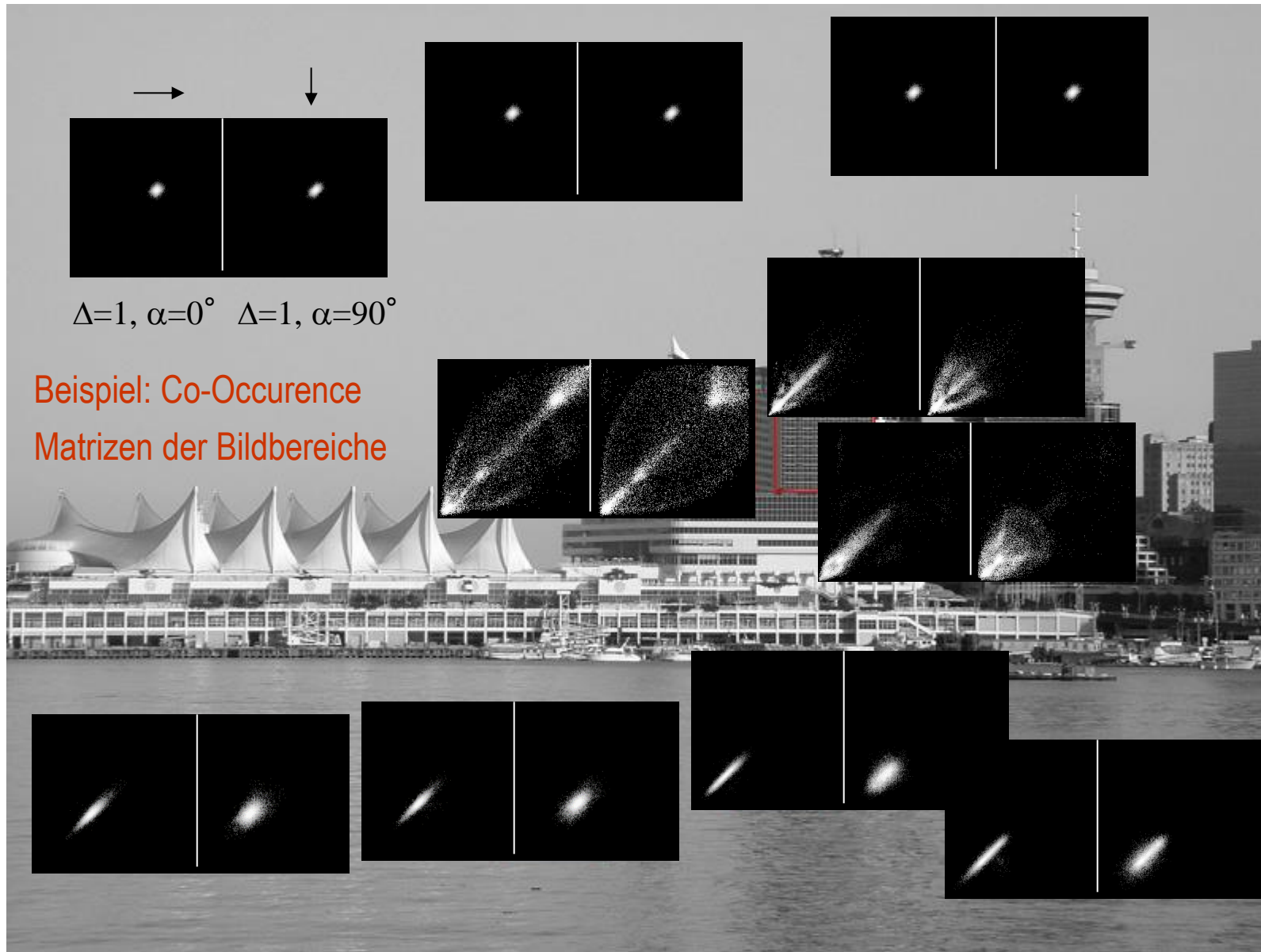
mit $0 \cdot \log_2 0 = 0$
(s. auch Vorlesung 1)

Haralicksche Texturmaße (3)

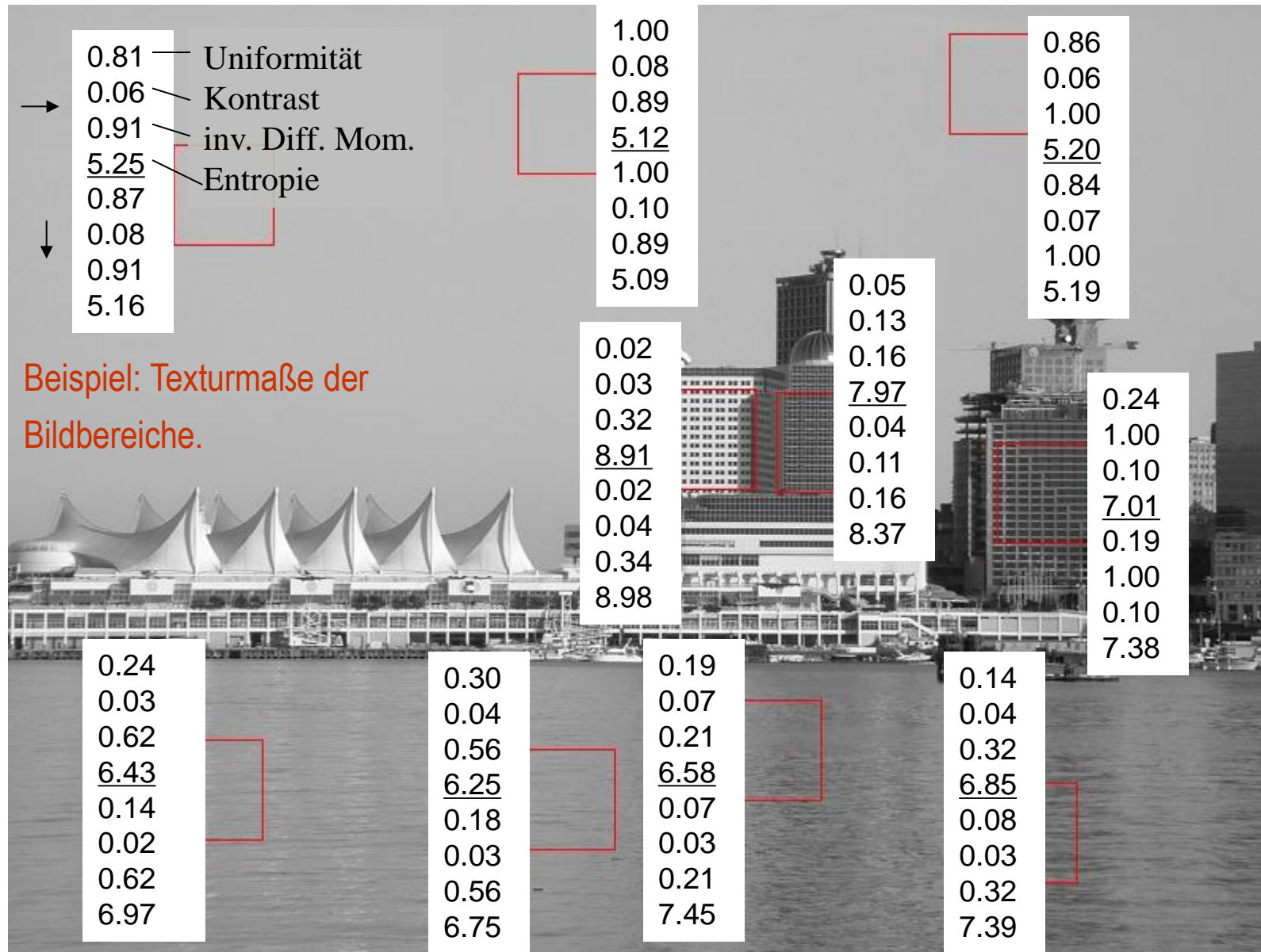


Beispiel: ausgewählte
Bildbereiche.

Haralicksche Texturmaße (4)



Haralicksche Texturmaße (6)



Texturbasierte Segmentierung mit Haralik-Maßen (1)

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

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

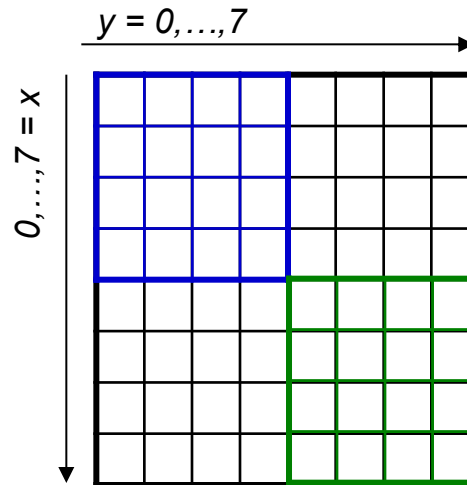
Pixelmengen, die dieselbe Textur zeigen - Schätzungen über Texturgrenzen führen zu Artefakten

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

Texturbasierte Segmentierung mit Haralik-Maßen (2)

Beispiel:

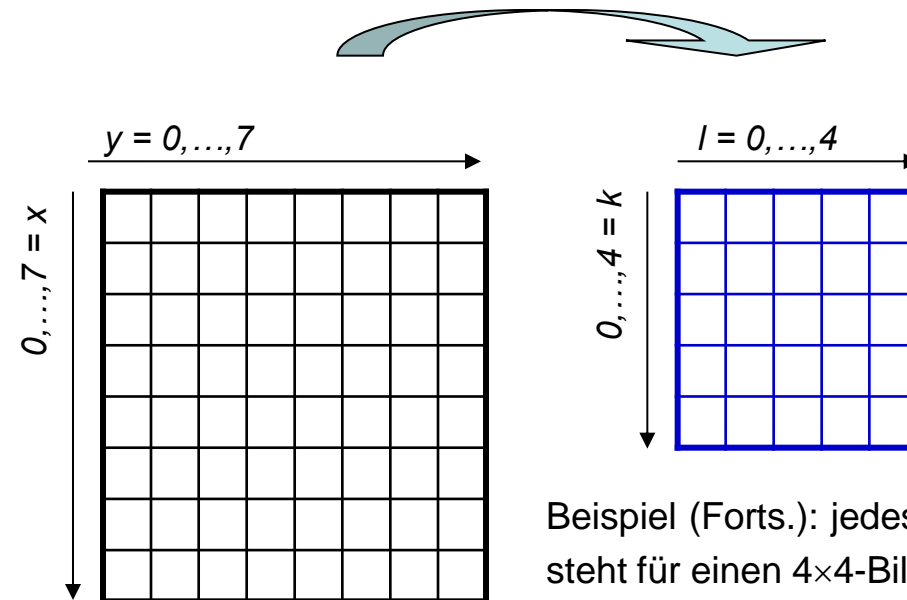
- Bild mit $Z = S = 8$ Bildzeilen und -spalten und 4×4 Bildblöcke für Texturanalyse.
- Dies ergibt 25 vollständig im Bild liegende und z. T. überlappende Bildblöcke von links oben nach rechts unten: erster Bildblock (blau) in $(0,0)$; letzter Bildblock (grün) in $(4,4)$.



Texturbasierte Segmentierung mit Haralik-Maßen (3)

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

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

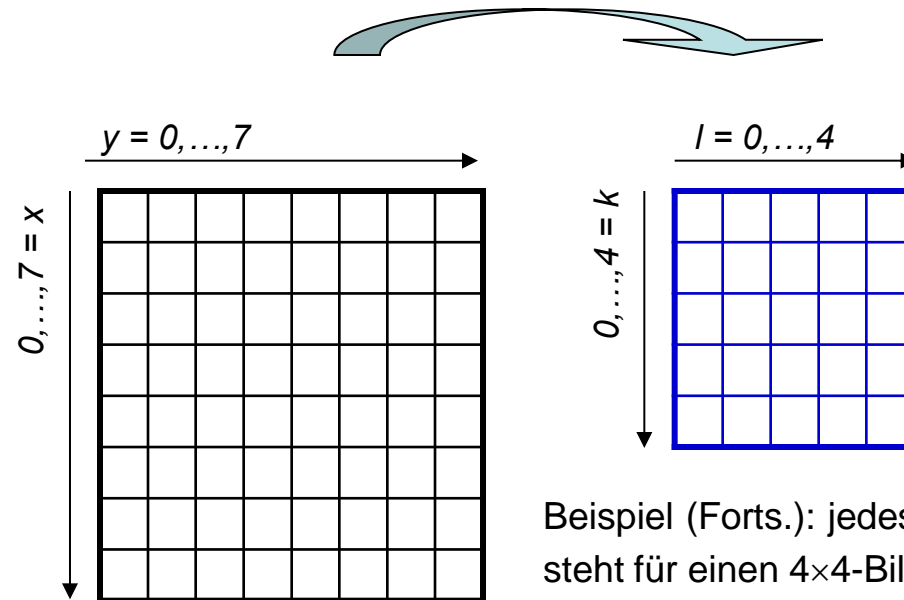


Beispiel (Forts.): jedes Pixel des **5x5-Texturbildes** steht für einen 4x4-Bildblock im 8x8-Originalbild.

Texturbasierte Segmentierung mit Haralik-Maßen (4)

Das Texturbild kann nun z.B. durch Split-and-Merge segmentiert werden.
Bspl.: Eine Region ist homogen, wenn die Euklidische Norm der Merkmalsvektoren zwischen allen Pixeln $\mathbf{p}_{x,y}$ der Region einen vorgegebenen Schwellwert d_{\min} unterschreitet:

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

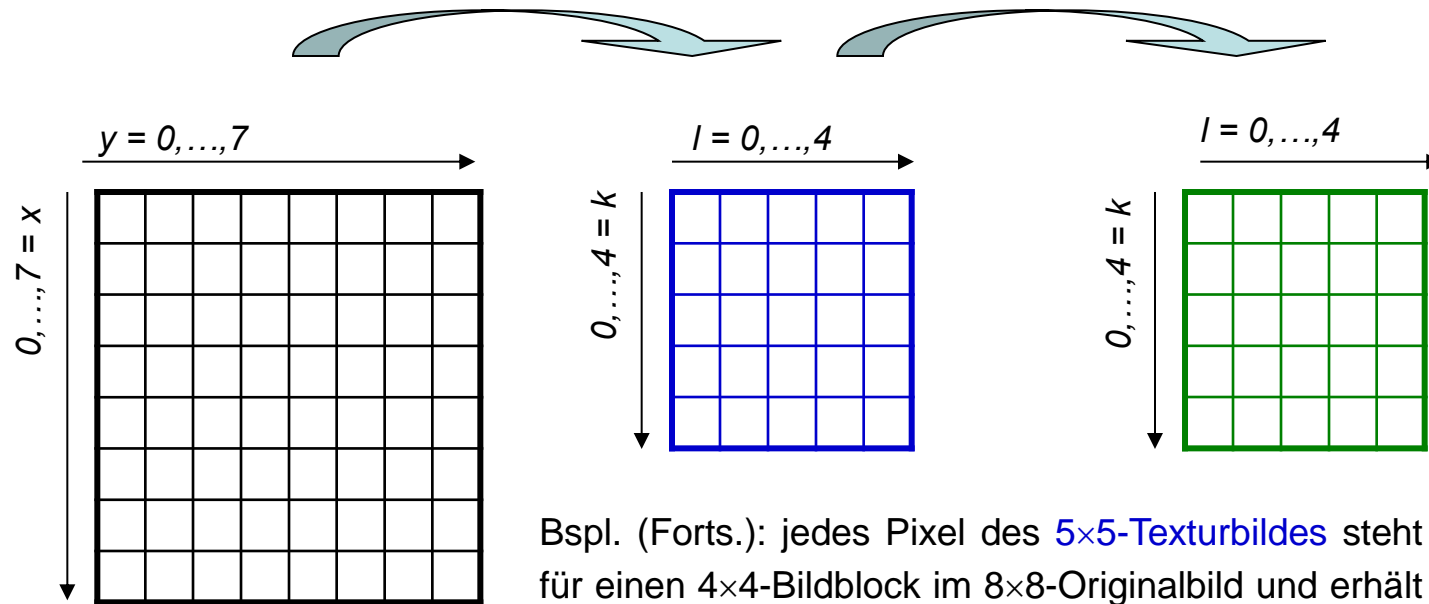


Beispiel (Forts.): jedes Pixel des **5x5-Texturbildes** steht für einen 4x4-Bildblock im 8x8-Originalbild.

Texturbasierte Segmentierung mit Haralik-Maßen (5)

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

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

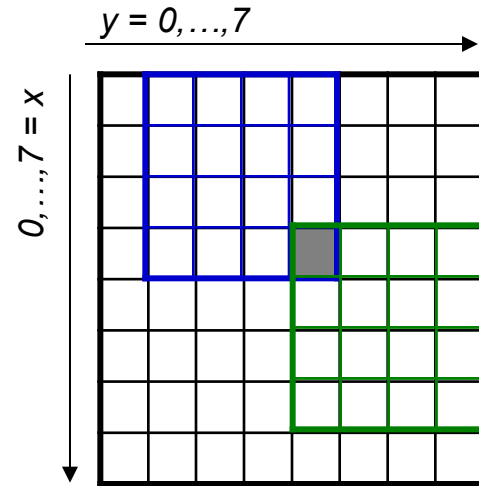


Bspl. (Forts.): jedes Pixel des **5x5-Texturbildes** steht für einen 4x4-Bildblock im 8x8-Originalbild und erhält ein Regionen-Label im Label-Feld L_{Block} .

Texturbasierte Segmentierung mit Haralik-Maßen (6)

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

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



Beispiel (Forts.):

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

Texturbasierte Segmentierung mit Haralik-Maßen (7)

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

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

Texturgrenzen (1)

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

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

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

Texturgrenzen (2)

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

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

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

Texturfrequenz

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

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

Zusammenfassung (1)

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

Zusammenfassung (2)

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

Zusammenfassung (3)

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

Zusammenfassung (4)

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