

Intelligente Sehsysteme

6 Segmentierung (II)

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

Volker Steinhage

Inhalt

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

Segmentierung nach Homogenitätskriterien (1)

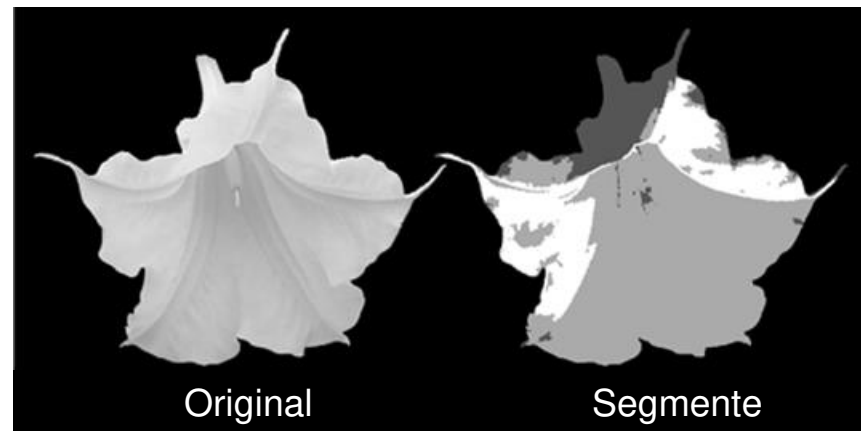
In der letzten Vorlesung: Segmentierungsansätze, die implizit oder explizit auf **Homogenitätskriterien** basieren, um benachbarte Pixel zu zusammenhängenden Segment zu gruppieren:

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

Segmentierung nach Homogenitätskriterien(2)

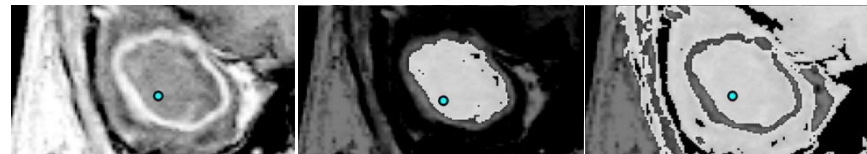
Zu beachten:

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



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

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



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

Segmentierung nach Diskontinuitätskriterien (1)

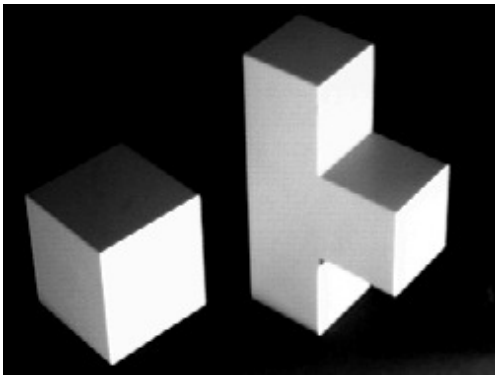
Bei stärkeren Variationen der Segmentcharakteristik bietet sich der *komplementärer Ansatz* der Segmentierung an:

Segmentierung nach Diskontinuitätskriterien

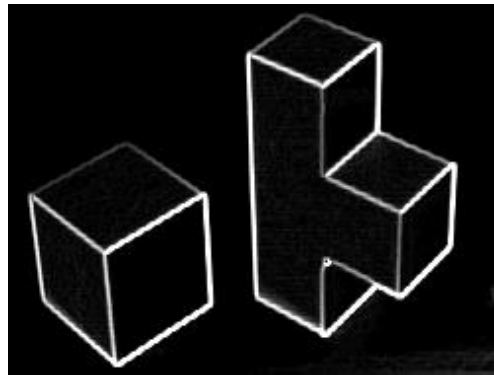
- Solche Ansätze zerlegen ein Bild nach Diskontinuitätskriterien an den *Segmenträndern* und sind daher a priori weniger anfällig gegenüber Variationen der Segmentcharakteristik
- Typische Diskontinuitätskriterien beziehen sich auf Intensitätsgradienten, die *Konturkanten* wiedergeben

Segmentierung nach Diskontinuitätskriterien (2)

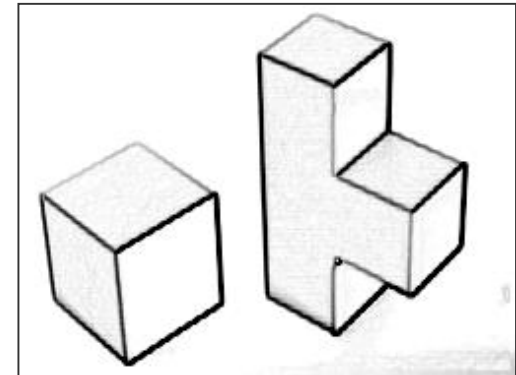
Kanteneigenschaften von Pixeln werden durch Operatoren der Bildverarbeitung (Low-Level Vision) wie z.B. durch den Sobel- oder Laplace-Operator hervorgehoben.



Grauwertbild



Antworten des
Sobel-Operators

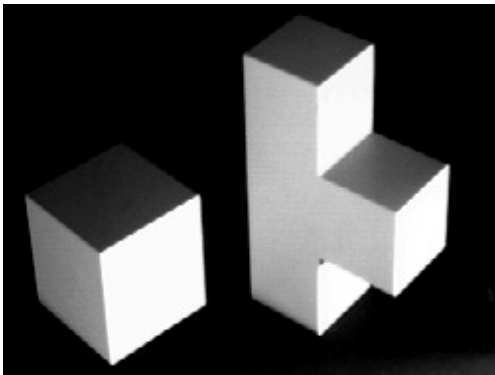


Invertierte Antworten des
Sobel-Operators

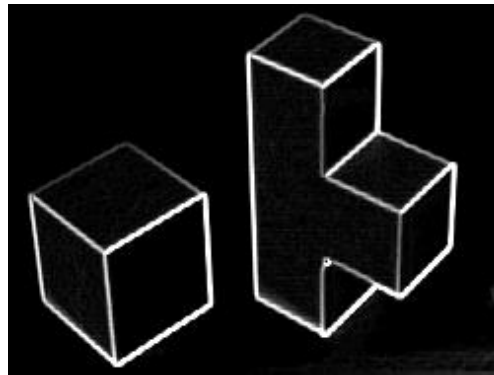
Segmentierung nach Diskontinuitätskriterien (3)

Segmente sind ableitbar, indem

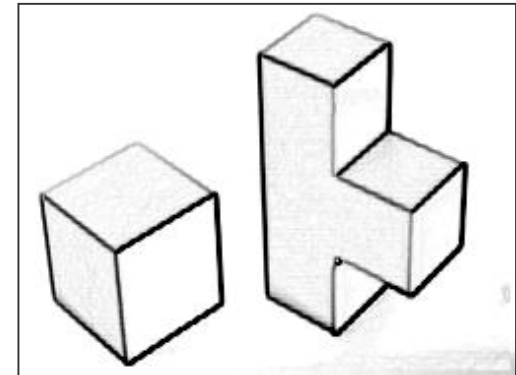
- aus allen Pixeln mit positiven Antworten des Kantenoperators vollständige und geschlossene Kantenzügen abgeleitet werden, die die **Segmente über deren Ränder festlegen**
- über eine Variante des *Region Labelings* ableitbar, bei der das Homogenitätskriterium festlegt, dass alle Pixel einer zusammenhängenden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



Grauwertbild



Antworten des
Sobel-Operators



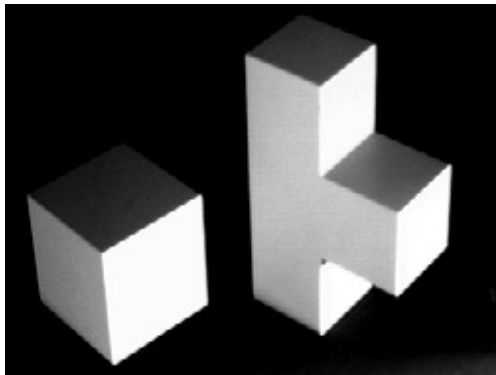
Invertierte Antworten des
Sobel-Operators

Segmentierung nach Diskontinuitätskriterien (4)

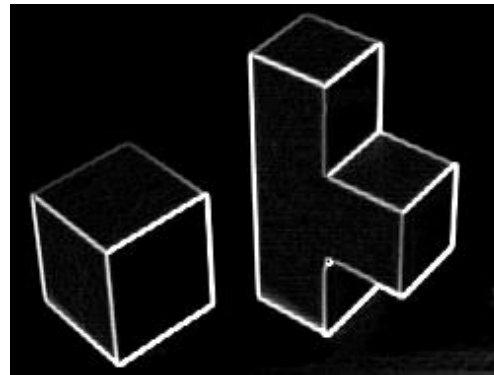
Kantenoperatoren geben wegen des Rauschens auch an solchen Stellen Operatorantworten, die keine Konturkanten abbilden.

Dem kann durch stärker angepasste Glättung begegnet werden.

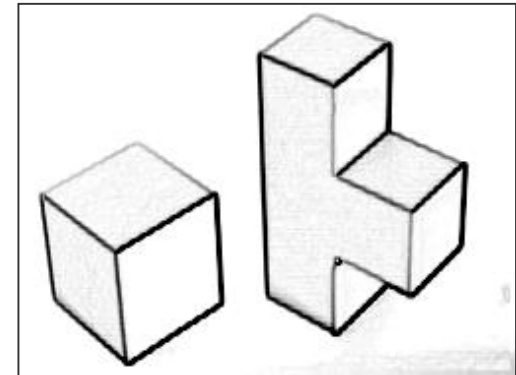
Zum anderen können die **Eigenschaften von Kantenzügen** genutzt werden wie z.B. beim **Edge Linking**.



Grauwertbild



Antworten des
Sobel-Operators

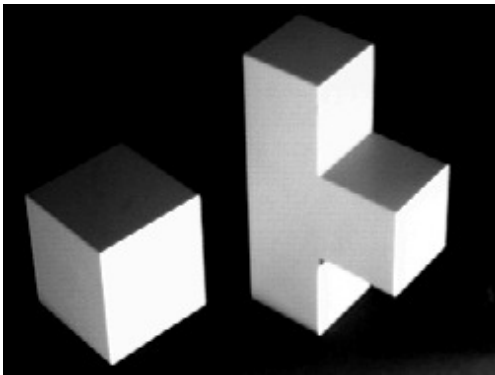


Invertierte Antworten des
Sobel-Operators

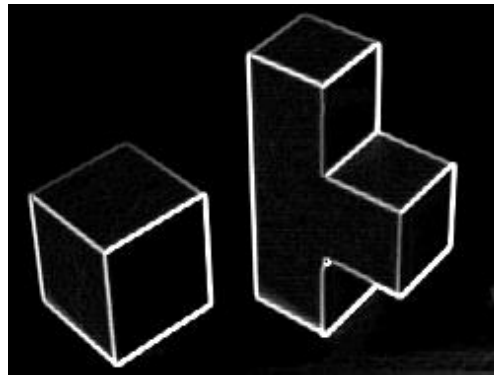
Eigenschaften von Kantenzügen (1)

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

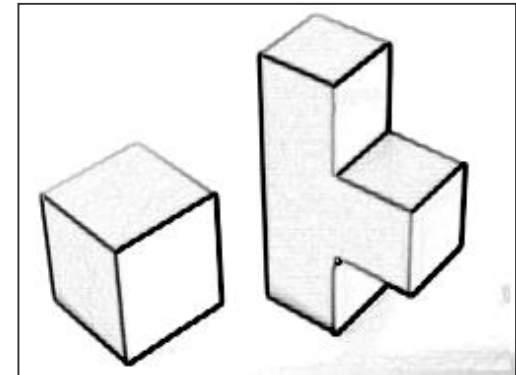
→ Ist der Gradientenbetrag gering, so kann die zugrunde liegende Intensitätsänderung durch Rauschen bedingt sein.



Grauwertbild



Antworten des
Sobel-Operators

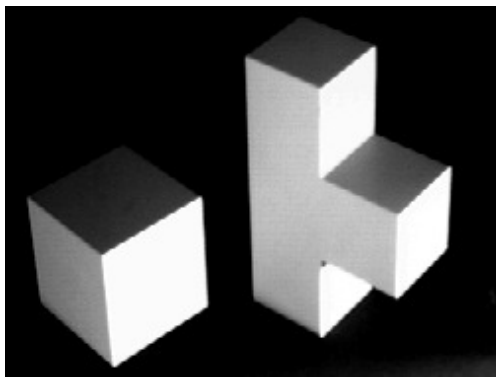


Invertierte Antworten des
Sobel-Operators

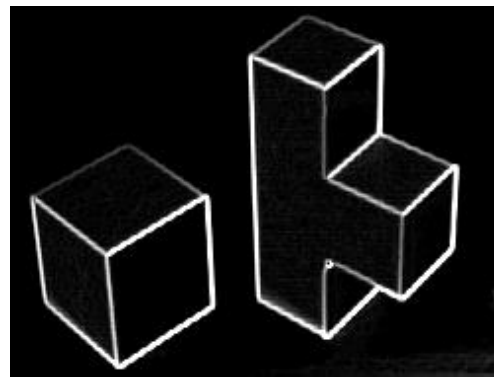
Eigenschaften von Kantenzügen (2)

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

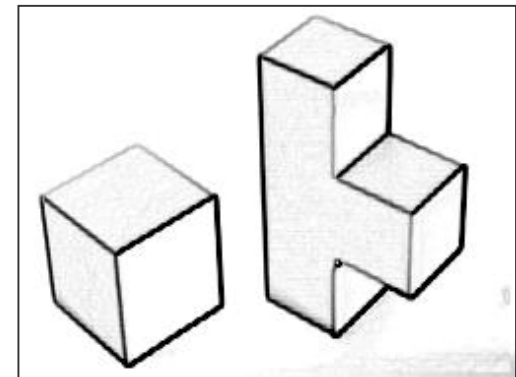
→ Die lokale Richtung des Kantenzuges sollte orthogonal zur Gradientenrichtung des betrachteten Kantenpixels sein.



Grauwertbild



Antworten des
Sobel-Operators

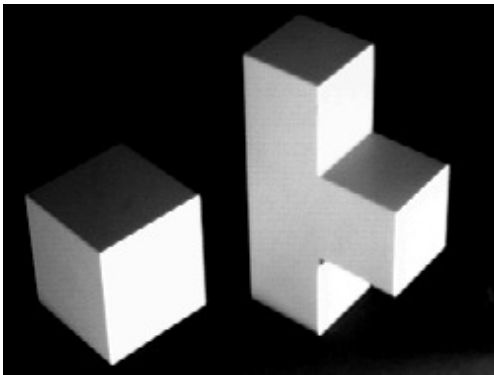


Invertierte Antworten des
Sobel-Operators

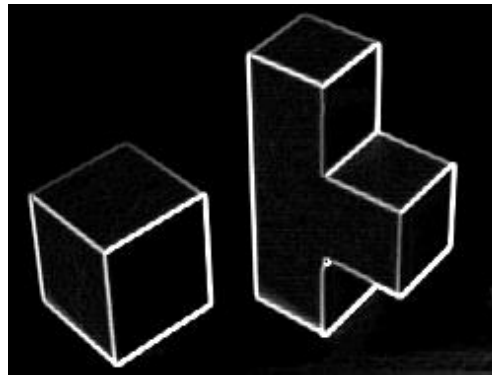
Eigenschaften von Kantenzügen (3)

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

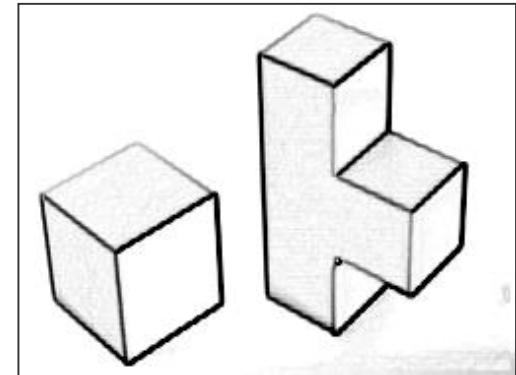
→ in der **lokalen Umgebung** eines **Kantenpixels** sollten weitere **Kantenpixel mit ähnlichen Eigenschaften** (genauer: Betrag und Richtung des Gradienten) zu finden sein.



Grauwertbild



Antworten des
Sobel-Operators



Invertierte Antworten des
Sobel-Operators

Binarisierung zur Selektion von Kantenpixeln (1)

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

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

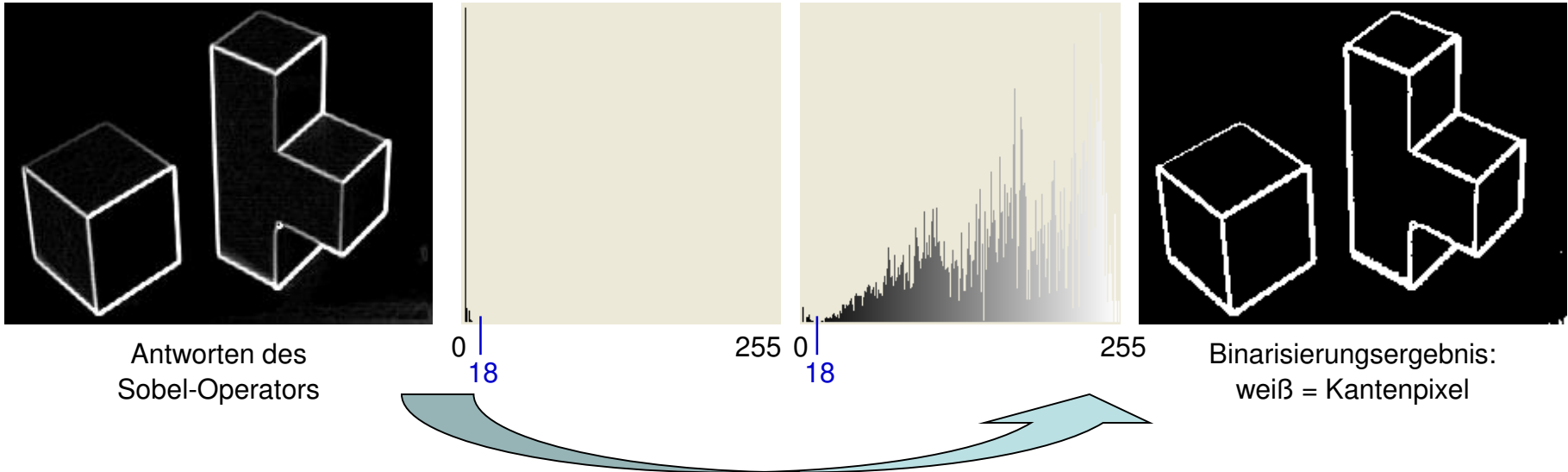
$$I(x, y) \in \begin{cases} \text{Objekt:} & I(x, y) \leq t_B, \\ \text{Hintergrund:} & I(x, y) > t_B. \end{cases}$$

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

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

Binarisierung zur Selektion von Kantenpixeln (2)

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



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

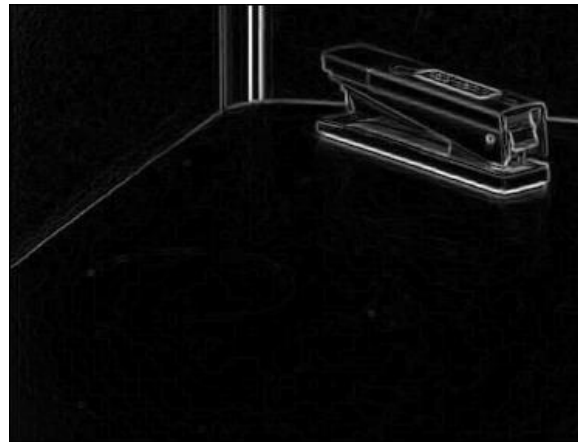
^{*} Bei automatisierter Binarisierung wäre noch eine Glättung des bimodalen Histogramms zur automatisierten Bestimmung des Minimums angemessen. Siehe Vorlesung 7 zu histogrammbasierter Segmentierung.

Binarisierung zur Selektion von Kantenpixeln (3)

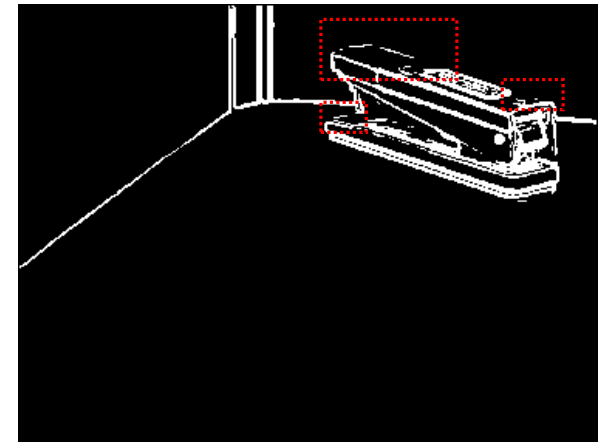
I. A. sind nach der schwellwertbasierten Binarisierung nicht mehr alle Kantenpixel über ihre Nachbarschaft miteinander verbunden, da es Bildbereiche gibt, in denen der Kontrast zwischen benachbarten Regionen zu gering ist (s. Bspl.).



Tacker-Grauwertbild*



Antworten Sobel-Operator



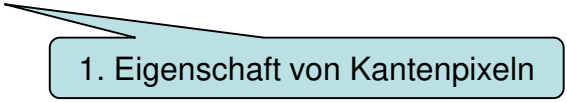
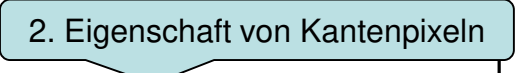

Binarisierungsergebnis für $t_B=40$

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

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

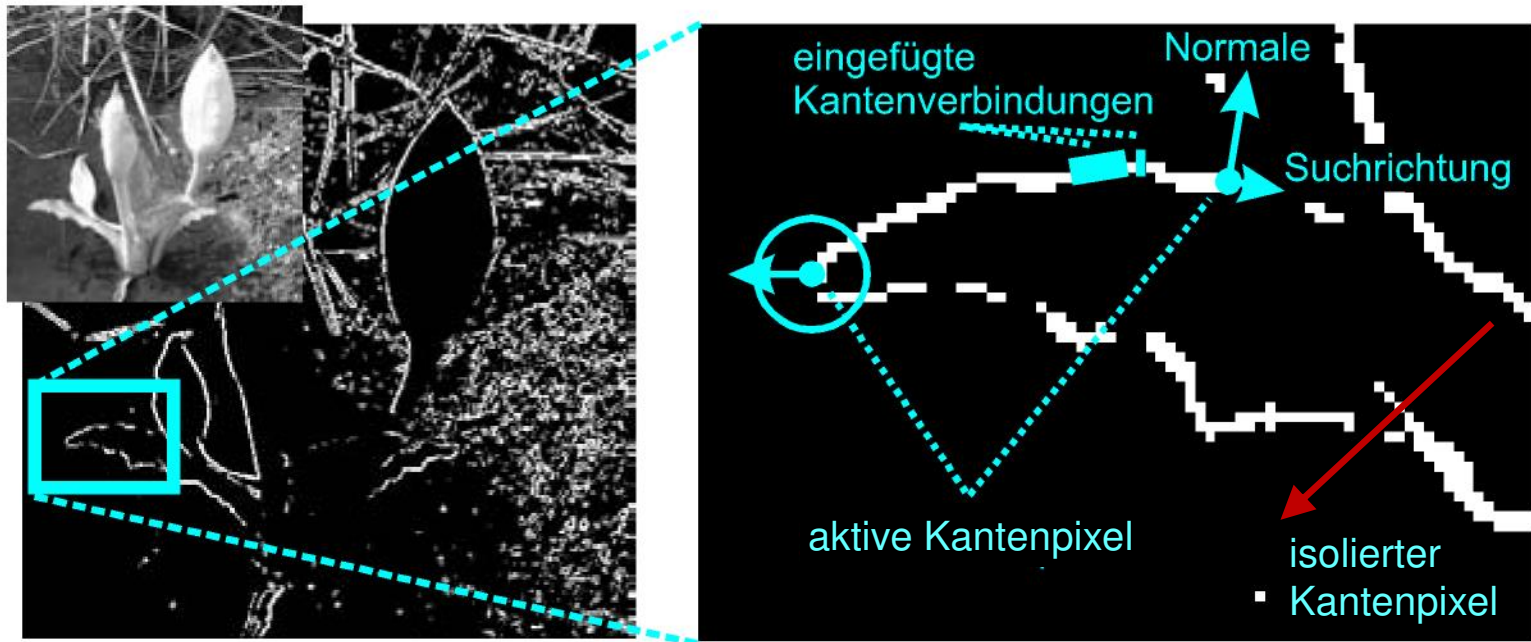
* Bildquelle: S. Russel, P. Norvig: Künstliche Intelligenz (2. Aufl.), Pearson Studium, 2004.

Algorithmus *Edge Linking*

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

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

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

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

Terminierung von *Edge Linking*

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



Parameter von *Edge Linking*

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

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

Überbrückung von Lücken mit *Edge Linking*

- Die *Größe* der Umgebung $U(p_a)$ definiert das Ausmaß, in dem **Lücken** in Kantenzügen überbrückt werden können
- Bspl: die Umgebung $U(p_a)$ sei als ein in p_a zentriertes 5×5-Pixelfeld definiert:

→ dann lässt dies Lücken von einem Pixel zu

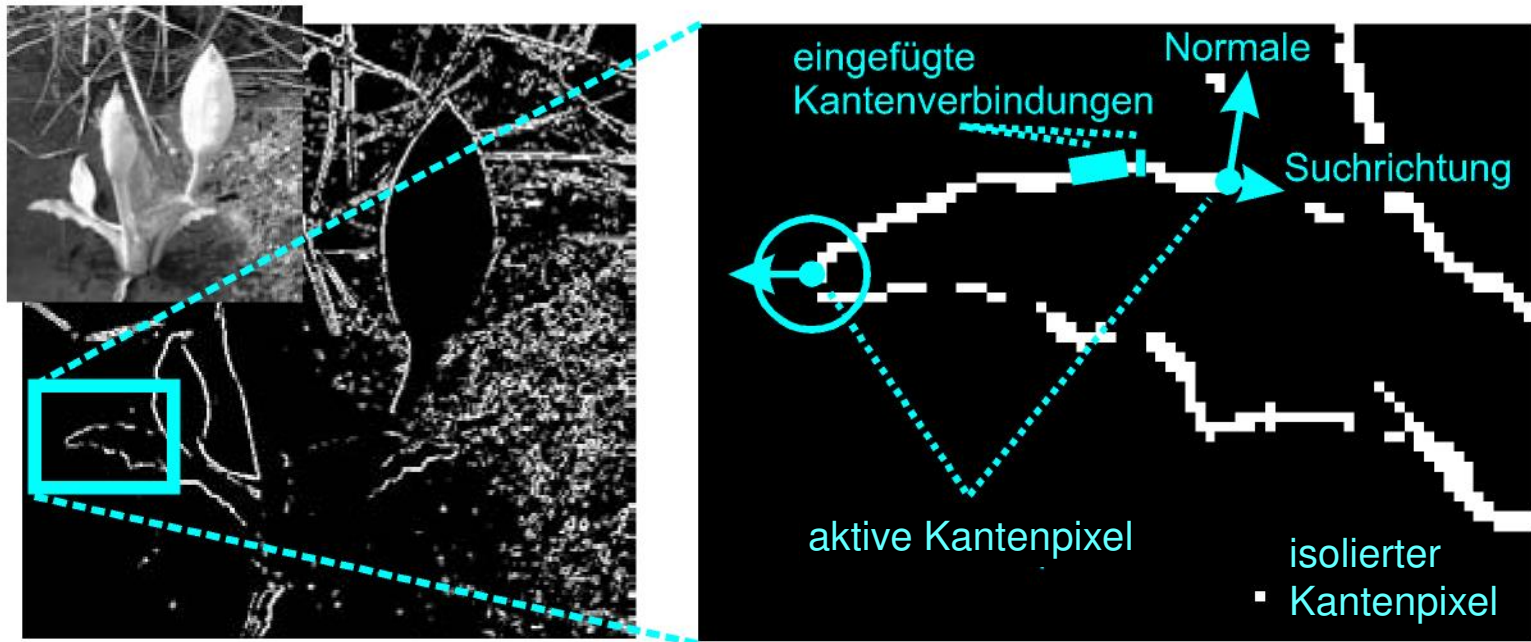
- Bspl. für Gradientenwerte in einem
in p_a zentrierten 5×5-Pixelfeld:

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

Einfügen von Kantenverbindungen in *Edge Linking*

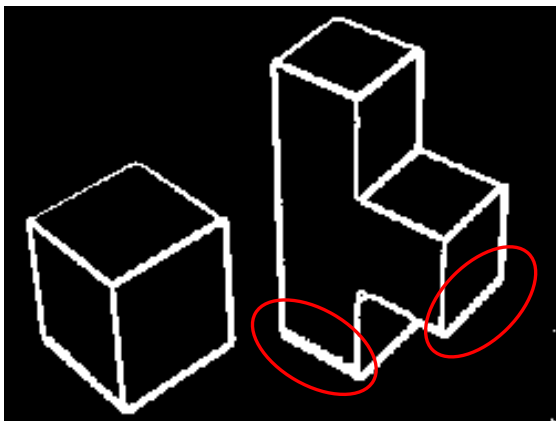
Die Überbrückung der Lücken erfolgt durch *lokale Kantenverbindungen*:

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



Zum Ergebnis von *Edge Linking*

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



Breite Gruppen von Kantenpixel im Polyederbild und im Ausschnitt des Pflanzenbildes

Verdünnung durch Skelettierung

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

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

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

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

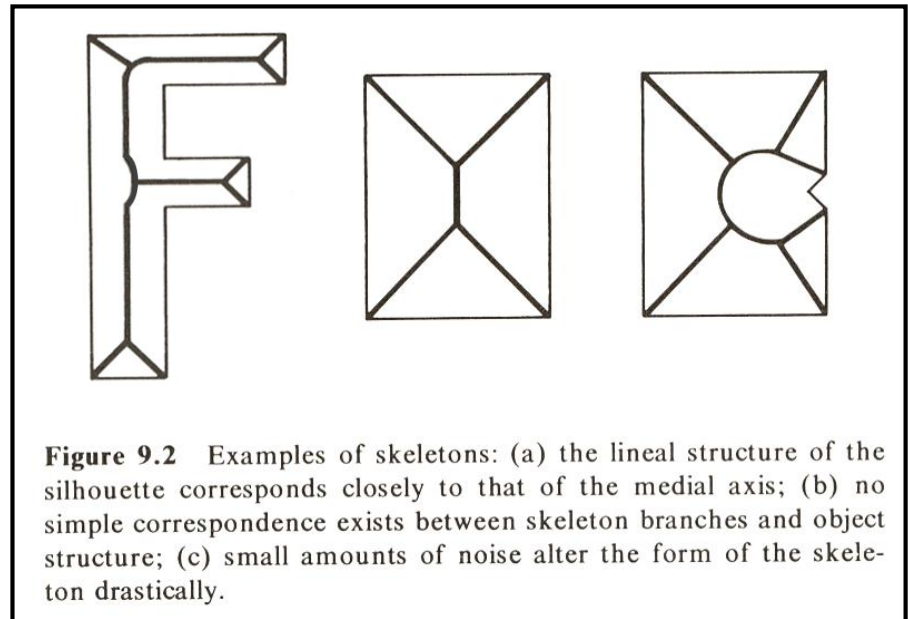


Bild korrigiert nach: Theo Pavlidis: *Algorithms for Graphics and Image Processing*. Springer, 1998.

Ziel der Skelettierung

Allgemein ist das Ziel der Skelettierung von Bildobjekten also

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

Es gibt verschiedene Definitionen für das Skelett eines Bildobjekts.

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

Definition von Skelett

Def. [Skelett]

Sei F eine planare Fläche mit Rand bzw. Grenze G und p ein Punkt in F . Ein *nächster Grenzpunkt* g von p ist ein Punkt g in G derart, dass kein anderer Grenzpunkt g' in G existiert, dessen Abstand pg' kleiner als der Abstand pg ist.

Wenn p mehr als einen nächsten Grenzpunkt hat, heißt p *Skelettpunkt* von F .

Die Menge aller Skelettpunkte von F heißt *Skelett* oder *Mittelachse* von F .

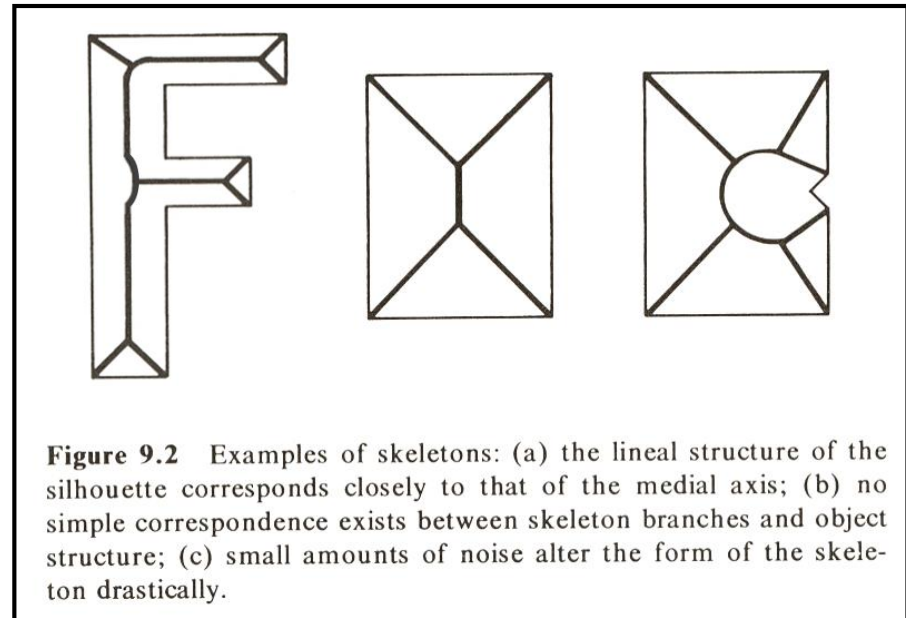


Bild korrigiert nach: Theo Pavlidis: *Algorithms for Graphics and Image Processing*. Springer, 1998.

Prinzip der Skelettierung

Prinzip der Skelettierung:

- **Schrittweises Abtragen der Objektpixel vom Rand** her bis zum Skelett
- Kern des Verfahrens: Entscheidung, welche Objektpixel zum Skelett gehören und nicht abzutragen sind, um den Zusammenhang des Bildobjekts zu gewährleisten.
- Die Erkennung der Skelettpixel erfolgt in vielen Ansätzen durch Inspektion der **8-Nachbarschaft**. Im Ansatz von Pavlidis* werden dazu zwei Muster überprüft:

A	A	A
0	P	0
B	B	B

A	A	A
A	P	0
A	0	1

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

Skelettierung nach Pavlidis: die Muster

Semantik der Skelettpunktmasken:

- 0 = Pixel ist nicht gesetzt = Hintergrund
- 1 = Pixel ist gesetzt = Objekt
- in beiden Pixelgruppen A und B muss jeweils mindestens 1 Pixel gesetzt sein, also Objektpixel sein
- Evaluierung über beide Muster und ihre einmal bzw. dreimal um 90° rotierten Varianten:

A	A	A
0	P	0
B	B	B

A	A	A
A	P	0
A	0	1

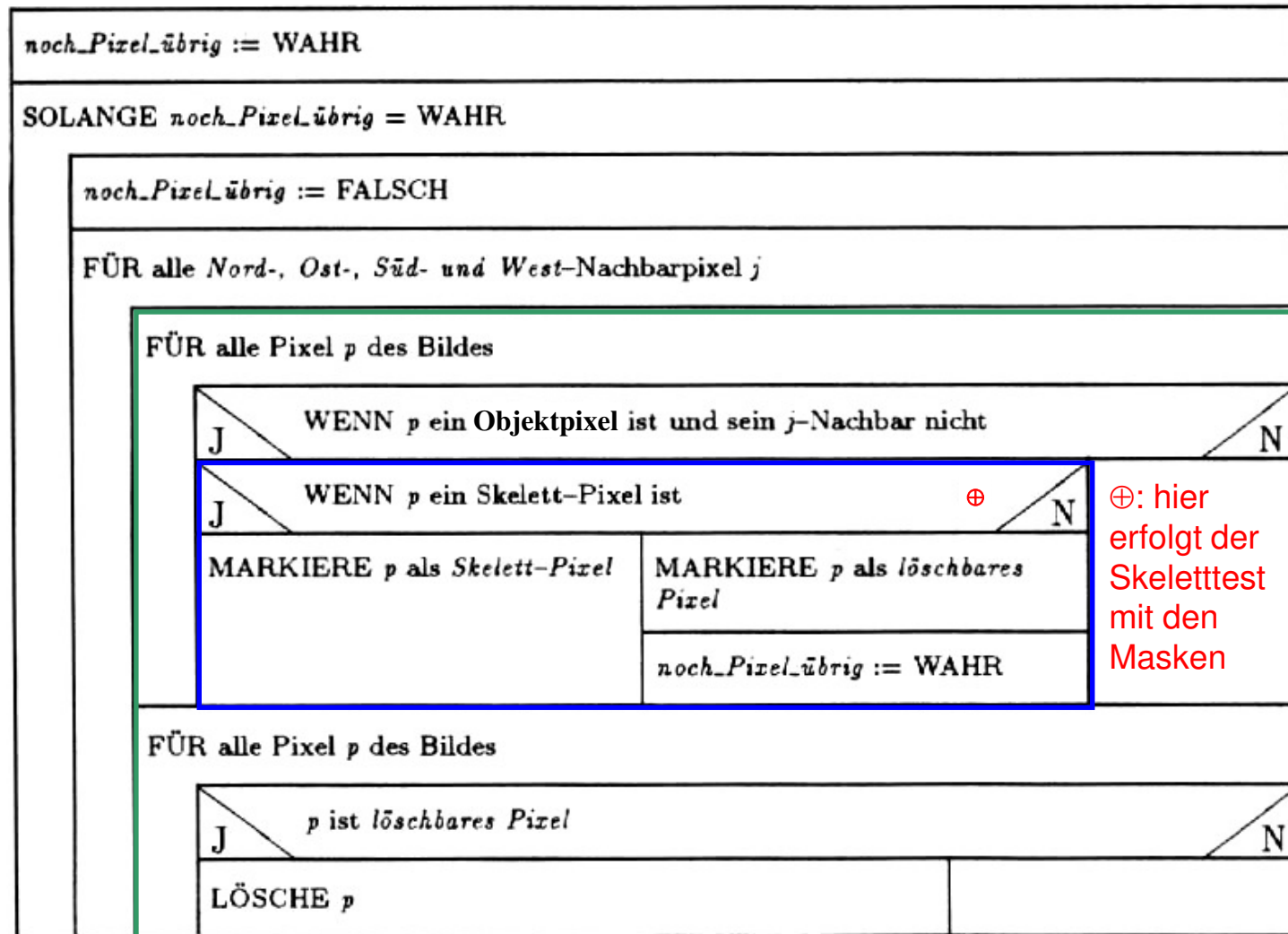
- Pixel P ist Skelettpixel, sobald P eines der beiden Muster bzw. einer rotierten Variante erfüllt
- Skelettpixel dürfen nicht gelöscht werden

Skelettierung nach Pavlidis: die Methode

Prinzip der Skelettierung nach Pavlidis:

- Iteratives Abtragen der Kantenpixel von den Rändern her;
- Nicht abzutragen sind Skelettpixel:
 - Kantenpixel, deren Entfernung den Zusammenhang der Linie zerstört
 - Zur Erkennung der Skelettpixel sind lokale Inspektionen ausreichend
- Terminierung, wenn kein Kantenpixel mehr entfernt wurde; d.h.: alle verbliebenen Kantenpixel sind Skelettpixel.

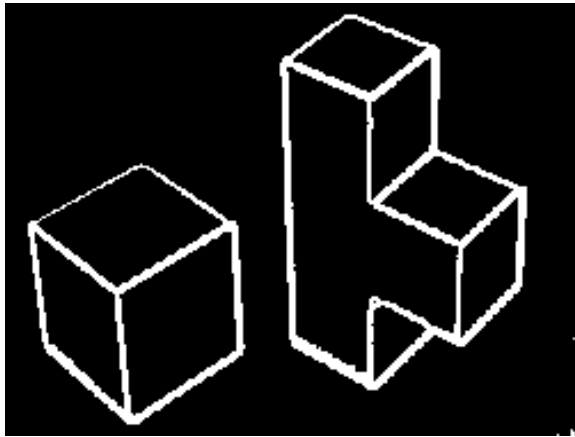
Skelettierung nach Pavlidis: der Algorithmus



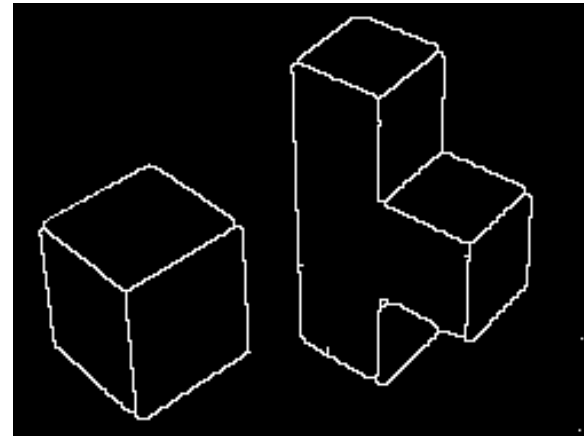
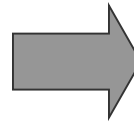
Struktogramm der Skelettierung nach Pavlidis

Skelettierung nach Pavlidis: Beispiel

Beispiel: Anwendung auf die Kantenpixel des Poylederbildes:



Kantenpixel



Extrahierte Skelettpixel
(nach Pavlidis)

Canny-Operator: Zielsetzung

Der Canny-Kantenoperator wurde 1986 von John Canny als **optimaler Kantendetektor** entwickelt*

Optimalität heißt dabei:

- **hohe Erkennungsrate** (hoher Recall):
Erkennung möglichst vieler Kanten der abgebildeten Szene
- **hohe Lokalisierungspräzision**:
möglichst ortsgenaue Lokalisierung der Kantenpixel
- **Eindeutigkeit**:
für jedes Kantenpixel soll es genau eine Detektorantwort geben

* Canny, J., *A Computational Approach To Edge Detection*, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

Canny-Operator: die Verarbeitungsstufen (1)

Mit der genannten Zielsetzung entwickelte Canny einen vierstufigen Algorithmus zur optimalen Kantendetektion.

- (1) Das Eingangsbild wird zunächst durch ein **Gauß-Filter** geglättet
- (2) Kantenpunkte werden durch den **Sobel-Operator** bestimmt
- (3) Breite Anordnungen von Kantenpunkten werden durch Unterdrückung von „Non-Maxima“-Kantenpunkten (engl. **non-maxima suppression**) verdünnt zu Linienbreiten von einem Kantenpunkt
- (4) Die in (2) erkannten und (3) selektierten Kantenpunkte werden durch Hysterese-Schwellwertbildung (engl. **hysteresis thresholding**) zu Kantenverläufen zusammengefügt

Canny-Operator: die Verarbeitungsstufen (2)

Der Canny-Algorithmus arbeitet mit drei Parametern σ , t_1 und t_2 :

- (1) **Glättung** durch **Gauß-Filter** mit Standardabweichung σ
- (2) **Kantenerkennung** mit **Sobel-Operator**:
 - (2.1) Gradientenbetrag: $|\nabla I(x,y)| \approx (S_x(x,y)^2 + S_y(x,y)^2)^{1/2}$
 - (2.2) Gradientenrichtung: $\Theta \approx \arctan(S_y(x,y) / S_x(x,y))$
gerundet zu 0° , 45° , 90° und 135° .
- (3) **Non-Maxima-Unterdrückung**: alle Kantenpixel, die in Gradientenrichtung Θ nicht ein lokales Maximum bilden, werden als solche verworfen
- (4) **Fusion zu Kantenzügen**: Kantenzüge werden mit Kantenpixeln begonnen, deren Gradient einen Schwellwert t_1 überschreitet. In Kantenrichtung (orthogonal zur Gradientenrichtung aus (2.2)) werden weitere Kantenpunkte des Kantenzuges erfasst. Die Kantenverfolgung bricht ab, wenn der Gradient des aktuellen Kantenpunktes unterhalb des zweiten Schwellwertes t_2 mit ($t_1 > t_2$) liegt.

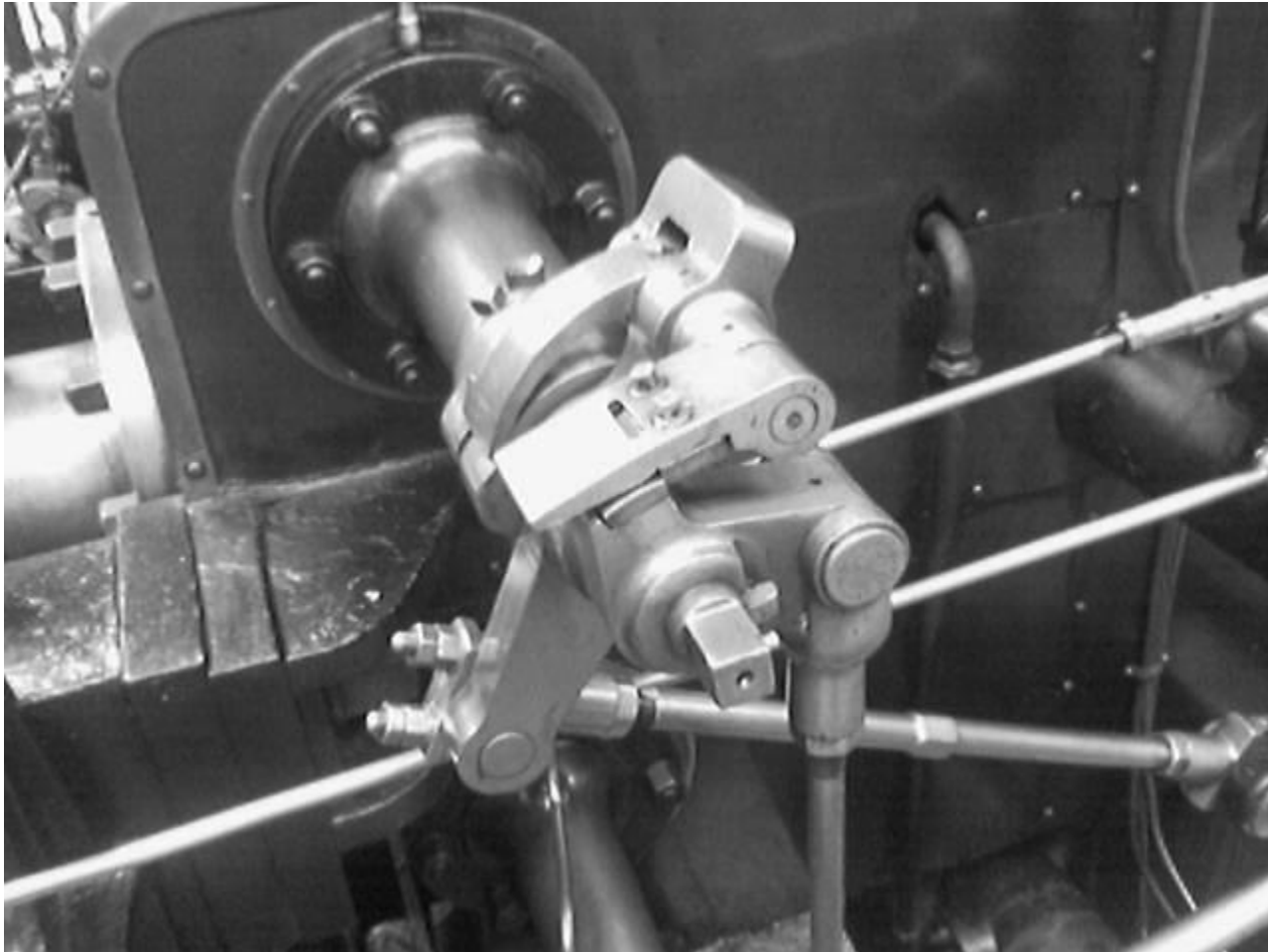
Canny-Operator: Hysterese

Bemerkung:

- Der vierte Schritt, also die Fusion zu Kantenzügen wird auch als **Hysterese-Verfahren** bezeichnet: „tracing edges with hysteresis thresholding“
- Als Hysterese bezeichnet man – insbes. in der Physik – ein Systemverhalten, bei dem die Ausgangsgröße nicht allein von der Eingangsgröße abhängt, sondern auch von dem vorherigen Zustand der Ausgangsgröße.
- Beim Canny-Operator ist die Binarisierung eben auch vom Zustand der Erkennung eines Kantenzuges abhängig: für den Start gilt Schwellwert t_1 , während der Gruppierung nach dem Start gilt Schwellwert t_2
- Hohe Werte von t_1 verhindern das Erkennen von schwach ausgebildeten Kanten. Kleinere Werte von t_2 verhindern Unterbrechungen der Kantenzüge. Canny empfiehlt in Abhängigkeit von erwarteten SNR ein Verhältnis von 3:1 bis 2:1 für die Schwellwerte t_1 und t_2

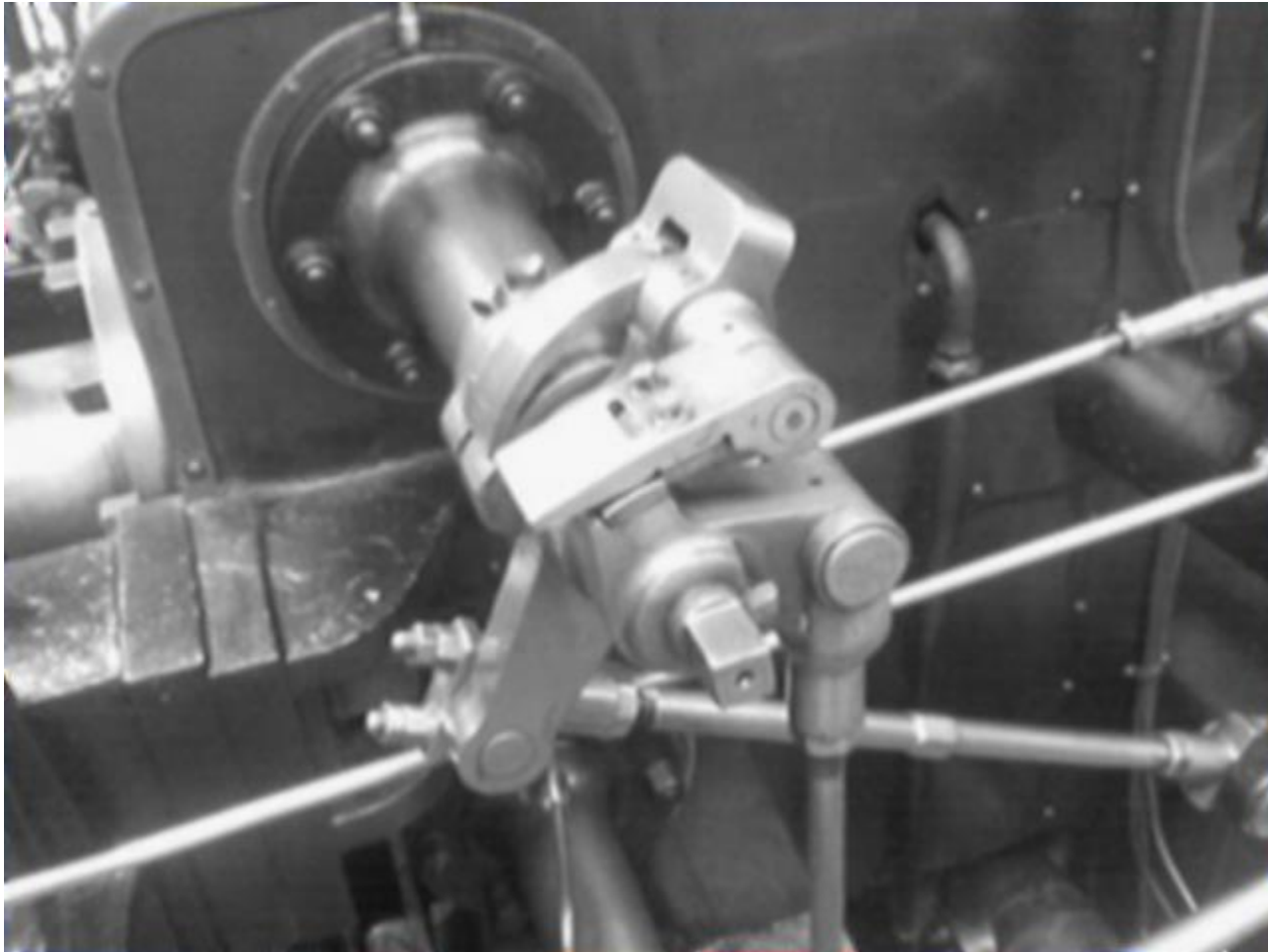
Canny-Operator: Beispiel (1)

Ausgangsbild für Canny-Operator:



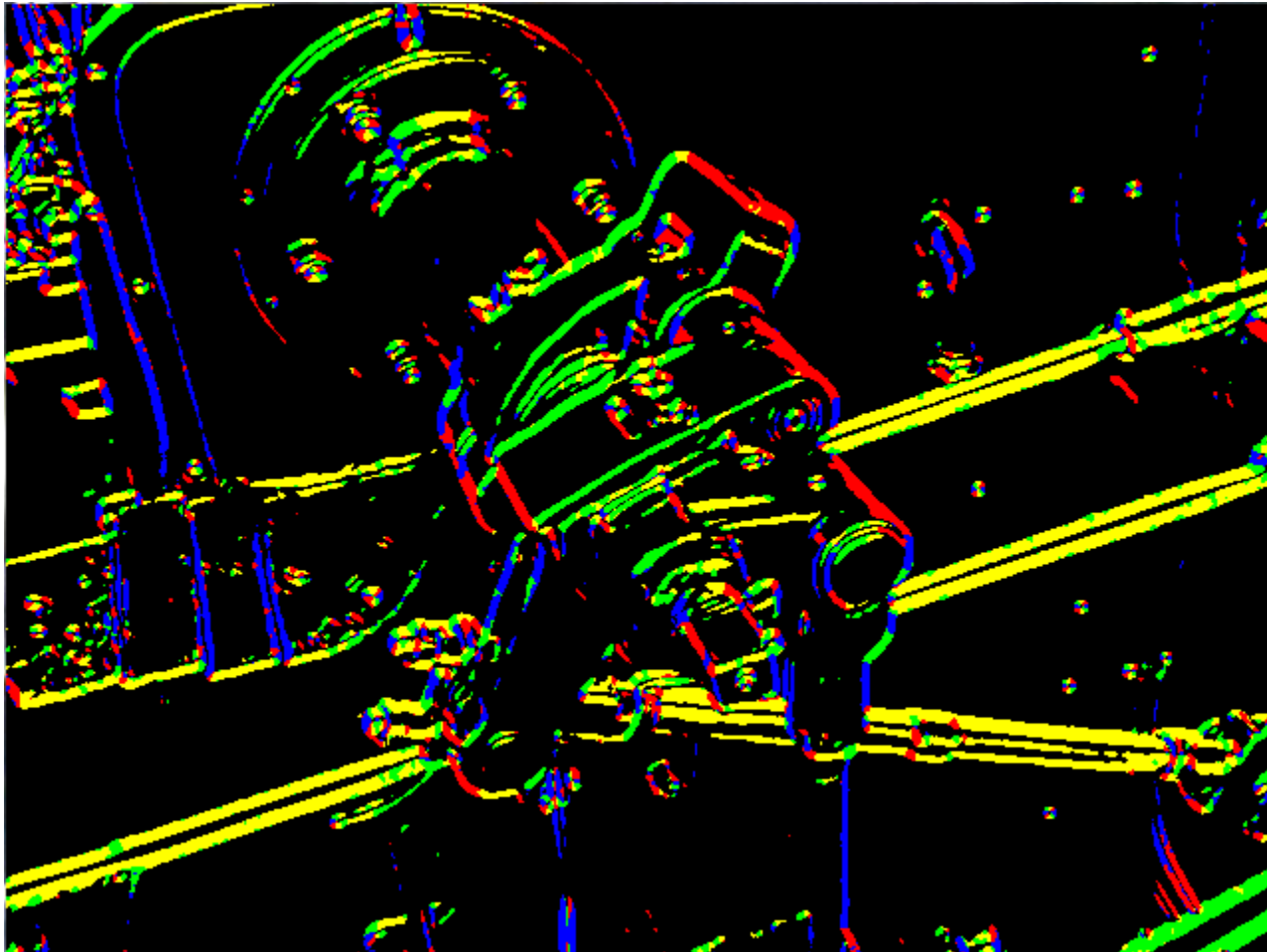
Canny-Operator: Beispiel (2)

Nach Anwendung von 5x5-Gauß-Filter mit $\sigma = 1,4$:



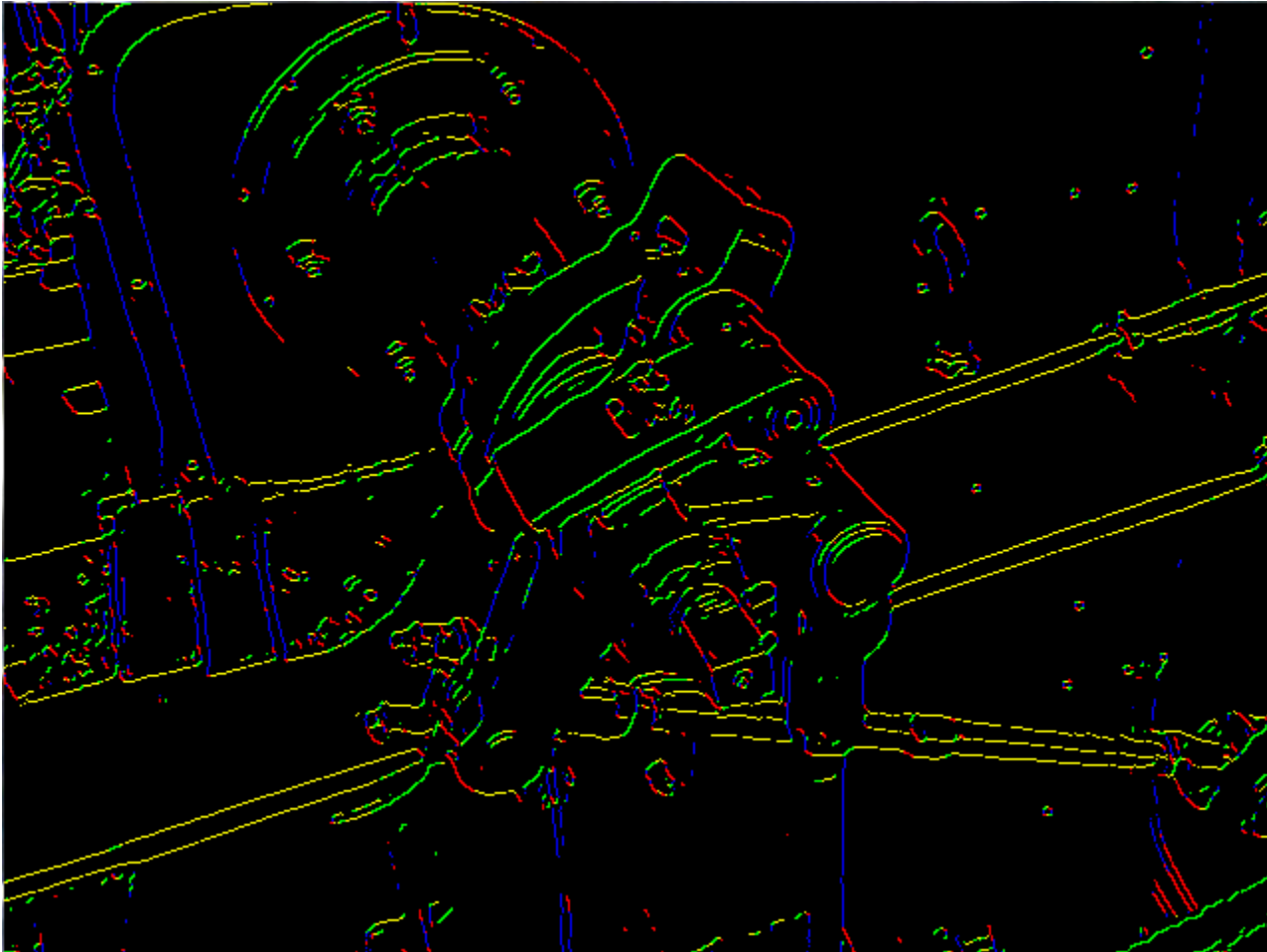
Canny-Operator: Beispiel (3)

Nach Anwendung von Sobel-Operator mit Farbkodierung der Klassifikation nach **Kantenrichtungen**: gelb = 0° , grün = 45° , blau = 90° , rot = 135°



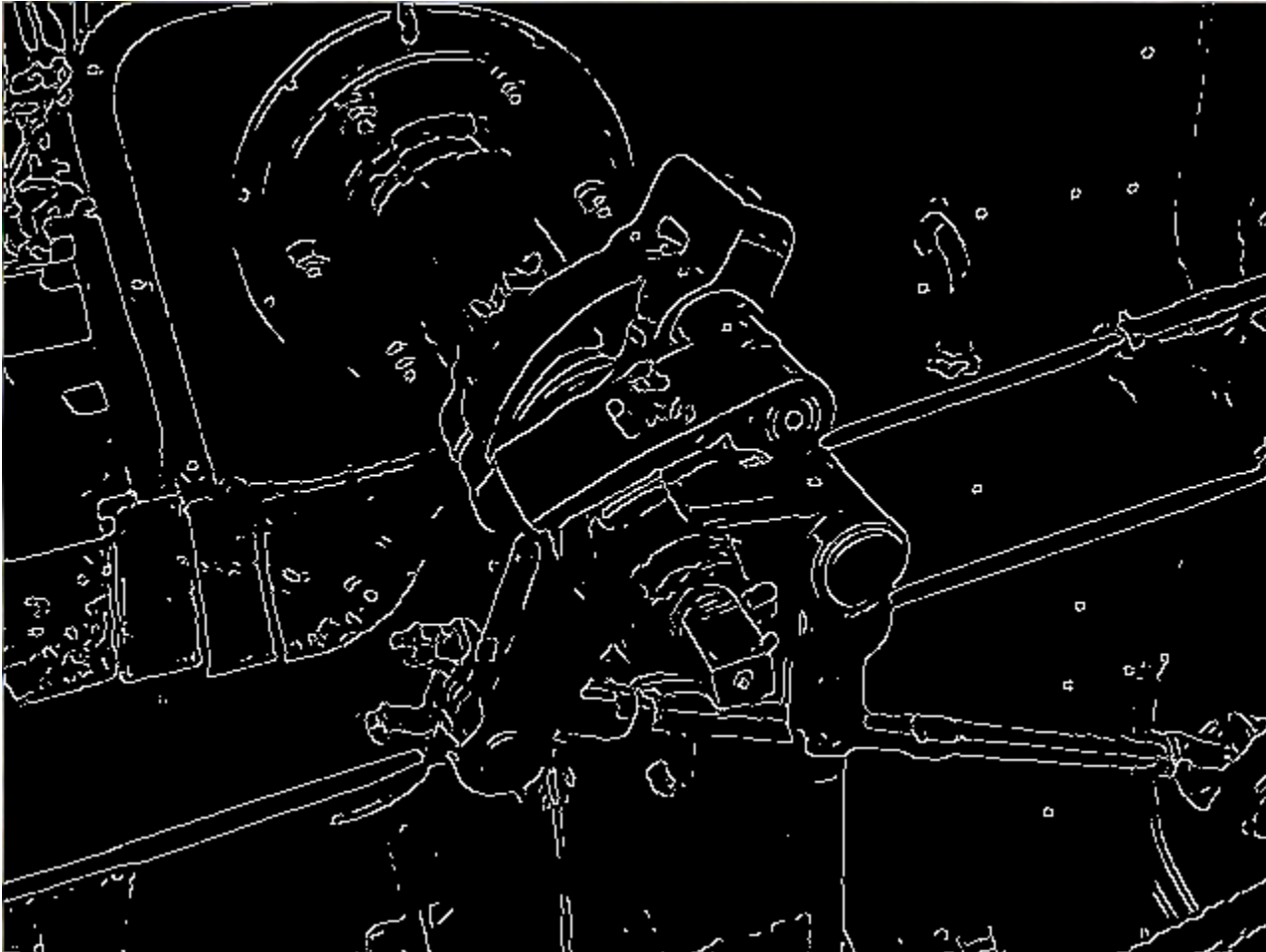
Canny-Operator: Beispiel (4)

Nach Anwendung der Non-Maximum-Unterdrückung in **Gradienten**richtung:



Low-Level-Vision: Canny-Operator (3.5)

Kantenzüge nach Hysterese-Schwellwertbildung:



Interaktive Suche nach optimalen Kantenzügen (1)

Wenn Start- *und* Endpunkt eines Kantenzuges bekannt sind, **dann** ist ein optimaler Kantenzug zwischen beiden Orten als Lösung eines **Wegeproblems** in einem Graphen ableitbar

Erzeugung des Graphen und Kostenfunktion:

- Überführe Bild $I = [I(x,y)]$ mit $Z \times S$ Pixeln in Graphen G_I mit $Z \times S$ Knoten
- Zwischen zwei Knoten k_1 und k_2 in G_I existiert eine Kante, wenn die entspr. Pixel p_1 und p_2 in I benachbart sind
- Jedem Knoten k werden Kosten $c(k)$ zugeordnet, die von den Eigenschaften des gesuchten Kantenzuges abhängen

Interaktive Suche nach optimalen Kantenzügen (2)

Die Kostenfunktion $c(k)$ sollte auf Gradienteninformation basieren:

- der Gradientenbetrag ist ungeeignet, weil er zu maximieren wäre und dabei alle nötigen, aber auch unnötigen Kantenpixel mitnehmen würde
- Tönnies (2005)* schlägt ein Abweichungsmaß von einer Modellgröße vor, z.B. einem Modellgradienten. Dieser kann als mittlerer Gradient von Startpixel s und Endpixel e definiert werden: $g_{\text{mod}} = \frac{1}{2} \cdot (|\nabla I(e)| + |\nabla I(s)|)$.
- Die **Kosten** $c(k)$ eines Knotens k ergeben sich dann als Abweichung seines **Gradientenbetrages** $g(k) = |\nabla I(k)|$ von g_{mod} :

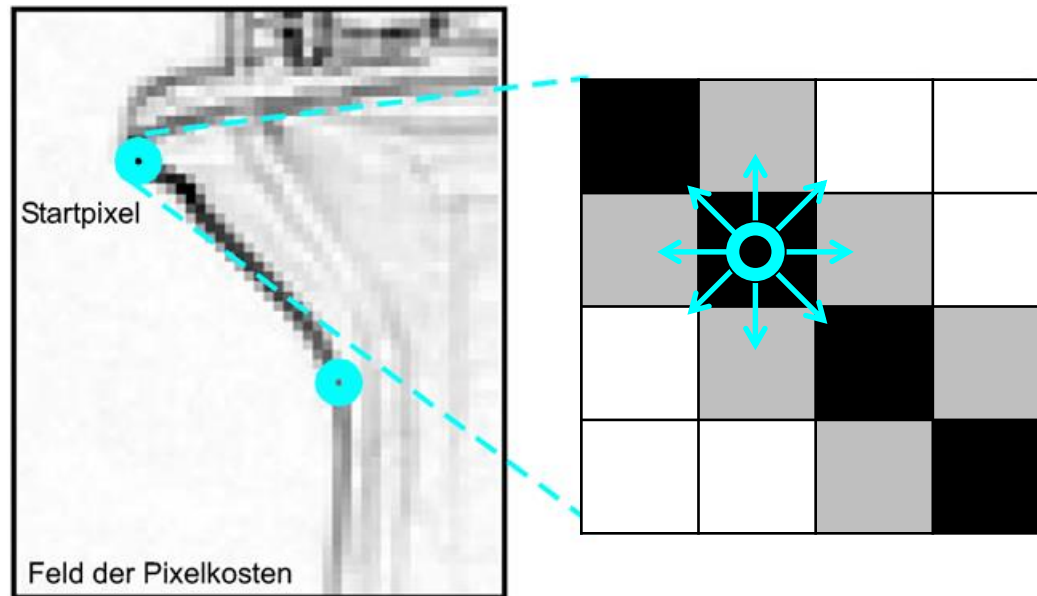
$$c(k) = |g(k) - g_{\text{mod}}|$$

* Es sind komplexere Kostenfunktionen anwendbar. In der Originalarbeit wird eine Kombination aus Nulldurchgängen der 2. Ableitung, den Gradientenbeträgen und Richtungsabweichungen der Gradienten eingesetzt (E. N. Mortensen, W. A. Barrett: Intelligent Scissors for Image Composition, ACM SIGGRAPH 1995, 191-198)

Interaktive Suche nach optimalen Kantenzügen (3)

Prinzip des Verfahrens:

Bei der Suche nach einem optimalen Kantenzug zwischen zwei Pixeln wird ausgehend vom Startpixel eine Suche nach einem Pfad mit minimalen Kosten zum Endpixel durchgeführt



Interaktive Suche nach optimalen Kantenzügen (4)

- Gegeben:
 - zusammenhängender Graph G_I ,
dessen Knoten mit den Pixeln des Bildes $I = [I(x,y)]$ korrespondieren
 - positive Knotenkosten,
die dem Abstand zu einem Modellgradienten entsprechen
 - Startknoten s und Endknoten e
 - Gesucht:
 - Pfad von Startknoten s zu Endknoten e mit minimalen Pfadkosten
- Umsetzung durch den [Dijkstra-Algorithmus](#) durchführbar

Interaktive Suche nach optimalen Kantenzügen (5)

Dijkstra-Algorithmus zur Suche nach optimalen Kantenzügen:

- Jedem Knoten k sind **Knotenkosten** $c(k)$ und **Pfadkosten** $p(k)$ zugeordnet. Die Pfadkosten $p(k)$ sind die Kosten vom Startknoten s zum Knoten k
- Die Pfadkosten $p(s)$ des Startknotens s sind dessen Knotenkosten $c(s)$.
- Alle anderen Knoten $k \neq s$ sind mit $p(k) = \text{max_cost} = \sum_k c(k) + 1$ initialisiert. Diese sind höher als die höchsten Kosten, um k von s aus zu erreichen.
- Die Liste *active_nodes* wird mit dem Startknoten s initialisiert.
- Aus *active_nodes* wird immer der Knoten k_{min} mit minimalen Pfadkosten ausgewählt und entfernt.
- Die Pfadkosten aller Nachbarn von k_{min} werden bestimmt. Sind diese geringer als ihre bisherigen Pfadkosten, werden diese korrigiert und die entspr. Knoten in *active_nodes* eingefügt.

Interaktive Suche nach optimalen Kantenzügen (6)

- **Terminierung:** Auswahl des Endknotens e aus *active_nodes*:
 - Da alle Pfadkosten positiv sind und immer der Knoten mit minimalen Pfadkosten aus *active_nodes* gewählt wird, können die Kosten jedes anderen Pfades zum Endknoten nur höher sein
- **Pfadausgabe:** Zurückverfolgung des Kantenzuges vom Endknoten e zum Startknoten s :
 - Die Rückwärtsverfolgung (*backtracking*) des gefundenen Pfades findet die Vorgängerknoten durch Vergleich der Pfadkosten mit den Pfadkosten an den Vorgängerknoten und den Knotenkosten:
Vorgängerknoten v zum Knoten k kann nur der Knoten sein mit

$$p(v) + c(k) = p(k)$$

Interaktive Suche nach optimalen Kantenzügen (7)

- Listing des Dijkstra-Algorithmus:

```
function Dijkstra (s,e);  
    active_nodes = {s};  
    p(s) = c(s);  
    for all nodes  $v \neq s$  do  $p(v) = \text{max\_cost}$ ;  
    while  $v\_min \neq e$  do  
         $v\_min = \text{select\_min\_cost}(\text{active\_nodes})$ ;  
         $\text{active\_nodes} = \text{active\_nodes} \setminus \{v\_min\}$ ;  
         $\text{neighbours} = \text{get\_neighbours}(v\_min)$ ;  
        for all  $w \in \text{neighbours}$  do  
            if [  $p(v\_min) + c(w) < p(w)$  ] then  
                 $p(w) = p(v\_min) + c(w)$ ;  
                 $\text{active\_nodes} = \text{active\_nodes} \cup \{w\}$ ;  
            endif  
        endfor;  
    endwhile;  
    return;  
end;
```

Interaktive Suche nach optimalen Kantenzügen (8)

- Listing der Rückwärtsverfolgung:

```
function backtrack (s,e);  
    k = e;  
    nodelist = {e};  
    while k  $\neq$  s do  
        neighbours = get_neighbours(k);  
        for all v  $\in$  neighbours do  
            if ( p(v) + c(k) = p(k) ) then  
                prev_node = v;  
            endif  
        endfor;  
        nodelist = nodelist  $\cup$  {prev_node};  
        k = prev_node;  
    endwhile;  
    return nodelist ;  
end;
```

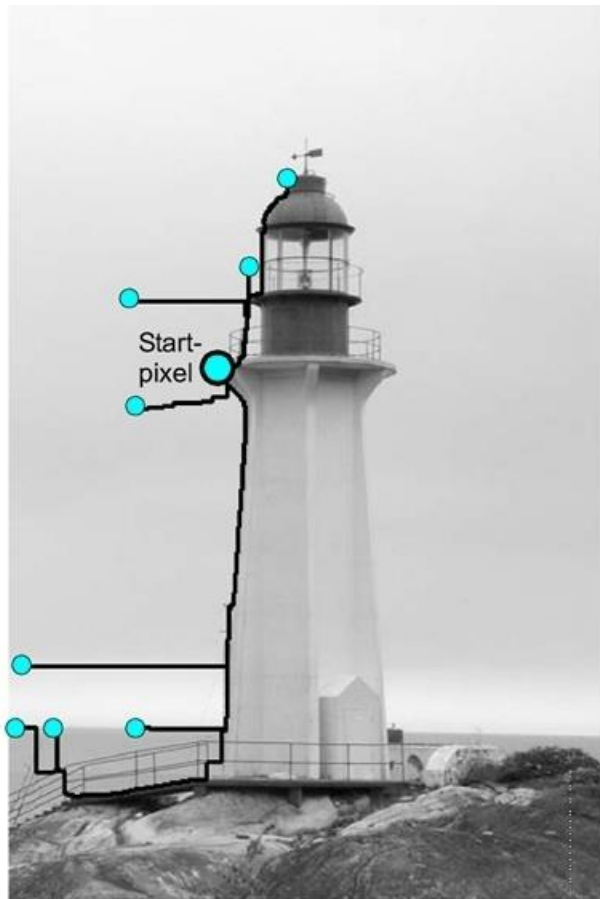
Interaktive Suche nach optimalen Kantenzügen (9)

Komplexität:

- Wird *active_nodes* als einfache Liste implementiert, ist die Zeitkomplexität des Dijkstra-Algorithmus $O(n^2+m)$ mit Knotenzahl n und Kantenzahl m . Bei Umsetzung von *active_nodes* in einer Baumstruktur folgt $O(n \cdot \log n + m)$.
- Die Rückwärtsverfolgung hat lineare Laufzeit in der Pfadpixelzahl und kann daher in Echtzeit umgesetzt werden. Dies führt zur sog. Variante des *Livewire*.

Interaktive Suche nach optimalen Kantenzügen (10)

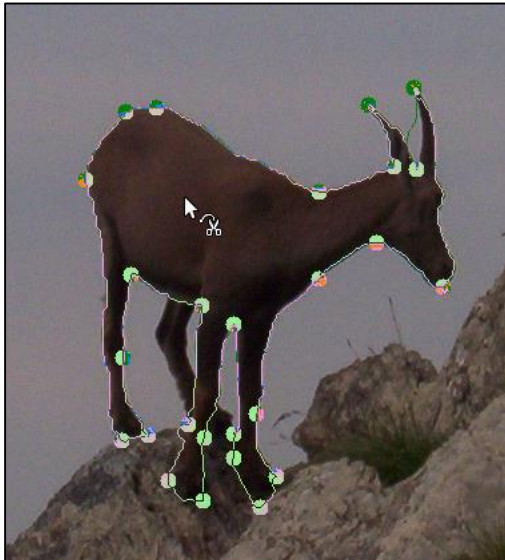
Die Livewire-Variante terminiert, wenn *active_nodes* leer ist. Daher werden kostenminimale Pfade vom Startknoten *s* zu beliebigen Endpunkten gesucht.



Interaktive Suche nach optimalen Kantenzügen (11)

Erweiterung des Verfahrens auf Folge von Kontrollpunkten:

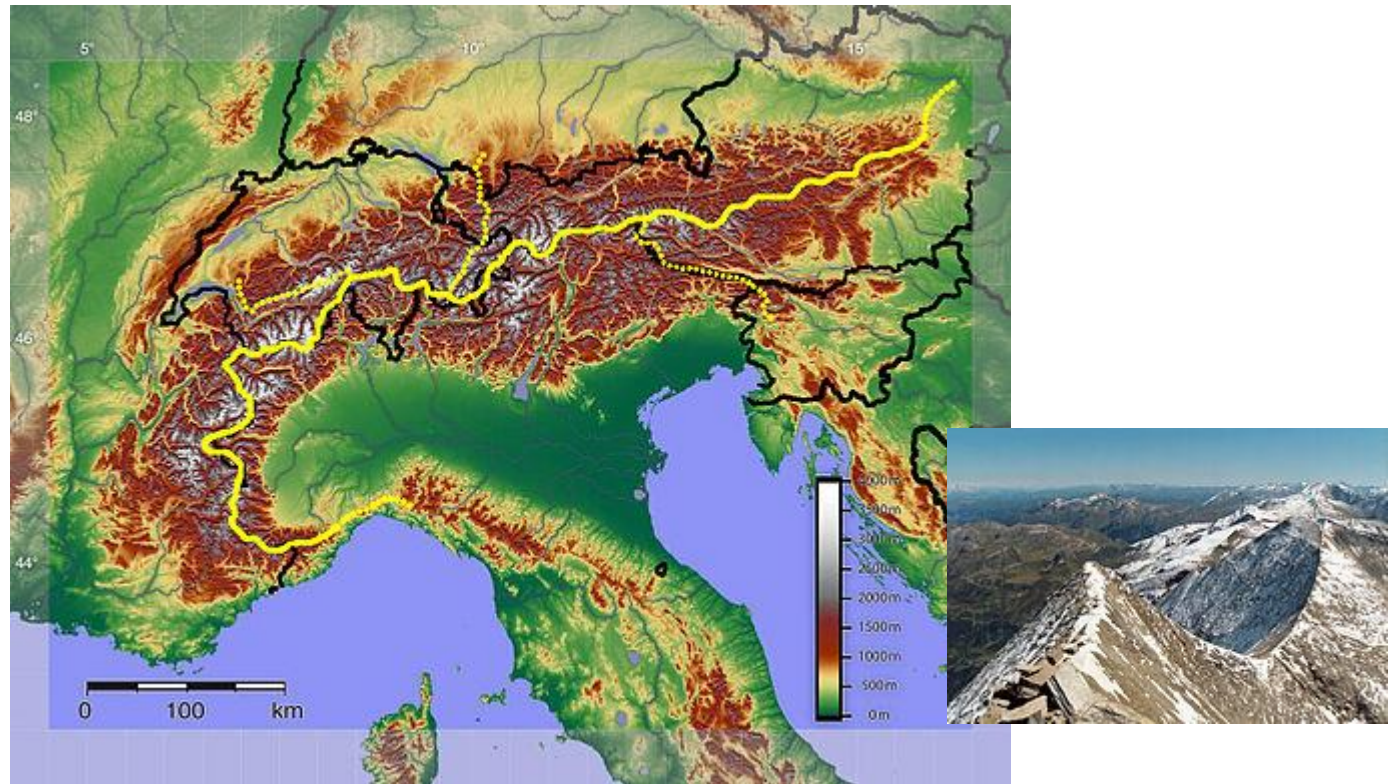
Werden aufeinanderfolgend angeklickte Cursor-Positionen als Folge von Start- und Endpunkten gewählt, ergibt sich ein interaktives Segmentierungsverfahren, das bekannt ist als *Intelligent Scissors*



“Each time you left-click with the mouse, you create a new control point, which is connected to the last control point by a curve that tries to follow edges in the image.”

Wasserscheidentransformation (1)

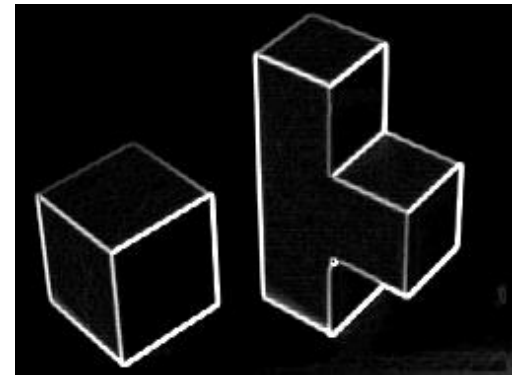
- In der Geographie trennt eine Wasserscheide Gebiete, die in unterschiedliche Senken entwässert werden
- Daher verlaufen Wasserscheiden i. A. entlang von Bergkämmen



Wasserscheidentransformation (2)

- Die **Wasserscheidentransformation (WST)** interpretiert die **Gradienteninformation als Höheninformation**.

Also entspricht die Identifikation der Wasserscheiden zwischen angrenzenden „Abflussgebieten“ der Ableitung der Grenzkonturen der Bildsegmente.



Gradienteninformation durch Sobel-Operator abgeleitet

- Der **Flutungsalgorithmus** erzeugt die Segmentierung des Bildes durch sukzessive Flutung des Gradientengebirges beginnend mit den lokalen Gradientenminima bzw. Senken bzw. Quellen.

Flutungsalgorithmus (1)

Jedem Pixel \mathbf{p} wird dessen lokaler Gradientenbetrag als Höhe $h(\mathbf{p})$ zugeordnet.

Für jede Flutungshöhe $h_{\text{aktuell}} = 0, \dots, h_{\text{max}}$

Für jedes *neu* überflutete Pixel \mathbf{p} mit $h(\mathbf{p}) = h_{\text{aktuell}}$

Wenn \mathbf{p} isoliert ist („Quelle“)

(d.h. \mathbf{p} ist nicht benachbart zu anderen bereits gelabelten überfluteten Pixeln \mathbf{p}' mit $h(\mathbf{p}') \leq h_{\text{aktuell}}$),
dann **vergebe neues Segment-Label** für \mathbf{p} .

Wenn \mathbf{p} eine Überflutungsregion erweitert

(d.h. \mathbf{p} ist benachbart zu bereits überfluteten und mit *identischem Segmentlabel* markierten Pixeln \mathbf{p}' mit $h(\mathbf{p}') \leq h_{\text{aktuell}}$),
dann **übernehme dieses Segment-Label** für \mathbf{p} .

Wenn \mathbf{p} Teil einer Wasserscheide ist

(d.h. \mathbf{p} ist benachbart zu überfluteten, aber mit *unterschiedl. Segmentlabeln* markierten Pixeln \mathbf{p}' mit $h(\mathbf{p}') \leq h_{\text{aktuell}}$),
dann **vergebe das Label „Wasserscheide“** für \mathbf{p} .

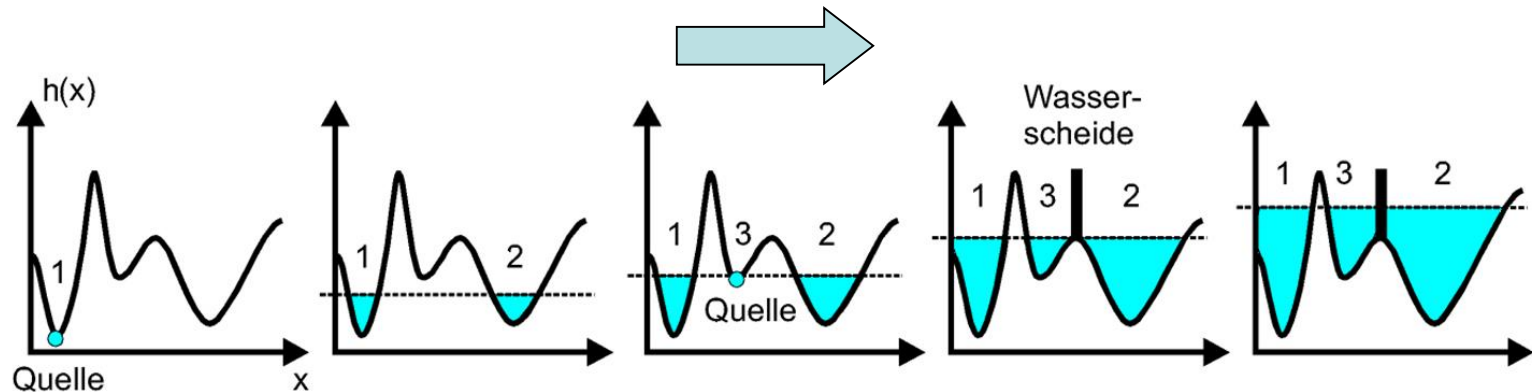
Label „Wasserscheide“
ist kein Segmentlabel

Flutungsalgorithmus (2)

Wenn die Höhe $h_{\text{aktuell}} = h_{\text{max}}$ erreicht ist, haben alle Pixel ein Label erhalten: entweder ein Segment-Label oder das Label „Wasserscheide“.

Die Wasserscheidenpixel können in einer optionalen Nachbearbeitung einer der angrenzenden Regionen zugeordnet werden.

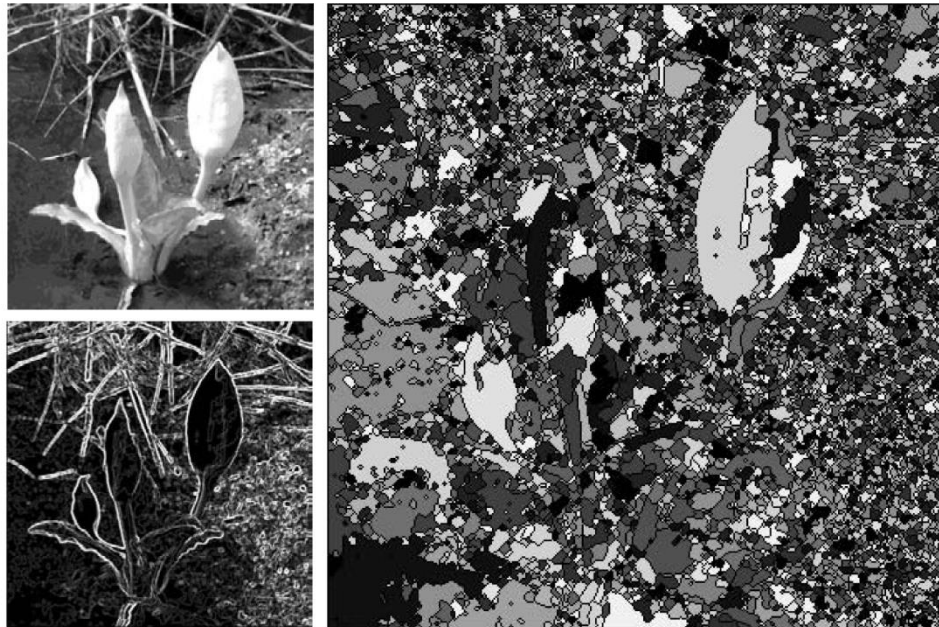
Die Abb. zeigt einige Stationen einer WST durch Flutung:



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

Ergebnisse der Wasserscheidentransformation

Die WST führt leicht zu einer **Übersegmentierung**.

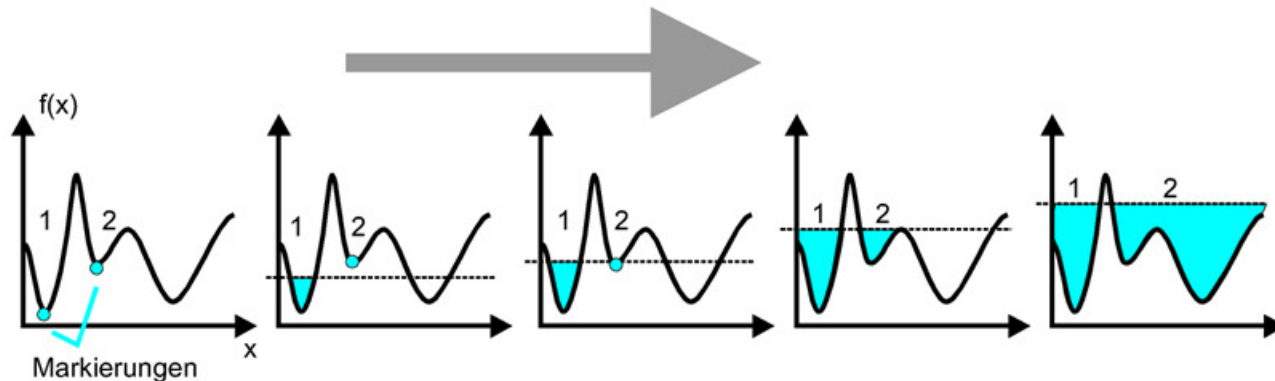


Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005.

Hierarchische Wasserscheidentransformation (1)

Der Übersegmentierung durch WST kann auf zwei Arten begegnet werden:

- Hierarchische WST:
 - Wiederholte WST auf jeweils vorherigen WST-Resultaten erzeugt sukzessive größere Segmente, bis die Größe der Segmente den Erwartungen entspricht
 - Es entsteht eine **Hierarchie von Segmenten**, in der Segmente einer Stufe Segmente der früheren Stufen zusammenfassen
- Interaktive WST: Interaktive Markierungen erzeugen das Labeling vor dem Flutungsprozess



Zusammenfassung (1)

- Alternativ zu den Segmentierungsansätzen nach **Homogenitätskriterien** bietet sich ein komplementärer Ansatz der Segmentierung an, nämlich die **Segmentierung nach Diskontinuitätskriterien**.

Solche Ansätze zerlegen ein Bild nach Diskontinuitätskriterien an den Segmenträndern und sind daher a priori weniger anfällig gegenüber Variationen der Segmentcharakteristik.

- Eine **schwellwertbasierte Binarisierung** kann zur **Auswahl von Kantenpixeln** genutzt werden. I.A. werden wegen schwacher Kontraste u.Ä. jedoch nicht alle Kantenpixel über ihre Nachbarschaft miteinander verbunden sein.

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

Zusammenfassung (2)

- Edge Linking erzeugt als Ergebnis Kantenzüge im Sinne von linearen Gruppen von Kantenpixeln. Diese können breiter als ein Pixel sein.
- Eine Klasse von Verfahren, um breite lineare Pixelgruppen auf eine Breite von einem Pixel zu verdünnen, ist die sog. Skelettierung.
- Der Canny-Kantenoperator wurde 1986 von John Canny mit dem Ziel entwickelt, einen optimalen Kantendetektor darzustellen. Der Canny-Kantenoperator kombiniert Glättung, Kanten hervorhebung, Edge-Linking und Verdünnung.

Zusammenfassung (3)

- Werden Start- und Endpunkt eines Kantenzuges interaktiv bestimmt, dann ist ein optimaler Kantenzug zwischen beiden Orten als Lösung eines **Wegeproblems** in einem Graphen mit dem **Dijkstra-Algorithmus** ableitbar.

Mit diesem Ansatz sind dann die Livewire-Variante sowie *Intelligent Scissors* ableitbar.

- Die **Wasserscheidentransformation (WST)** extrahiert lokale Maxima der Gradientenbeträge als Orte trennender Wasserscheiden und leitet damit Bildsegmente als „Abflussgebiete“ zwischen angrenzenden Wasserscheiden ab.

Die hierarchische WST sowie die interaktive WST begegnen einer möglichen Übersegmentierung durch die normale WST.