

Skelettierung

Topologie von Bildobjekten



Wolfgang Heiden © 2014-22, mit Beiträgen von R. Herpers, F. Mannuß, B. Kahl, G. Heisenberg

Wolfgang Heiden © 2014-22 wolfgang.heiden@fh-bonn-rhein.sieg.de

-- auf der Grundlage einer Lehrveranstaltung von Prof. Dr. Rainer Herpers sowie Folien von Florian Mannuß 2011, Dr. Björn Kahl 2012, Prof. Dr. G. Heisenberg, 2013 --

Fachbereich Informatik (Dpt. Computer Science)

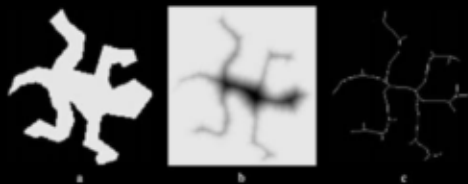
Hochschule Bonn-Rhein-Sieg – University of Applied Sciences,

53754 Sankt Augustin

Germany

Skelettierung von Bildobjekten

- **Objekt-Topologie**
- Reduktion eines **Objekts** auf eine **Linie** (1 Pixel) als Repräsentant des Objekts
- anwendbar auf **Binärbilder**
- **Objektpixel** (1), **Hintergrundpixel** (0)

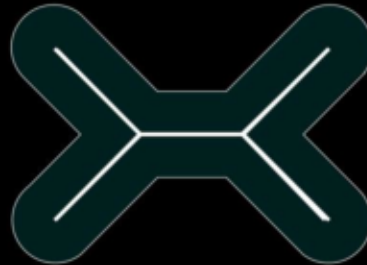
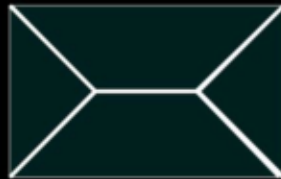


Im Bildbeispiel sind die Objektpixel in Bild a schwarz, die Hintergrundpixel weiß.

Bild b stellt eine Abstandskarte (distance map) für Bild a dar, wobei Hintergrundpixel schwarz dargestellt werden und große Abstände eines Pixels zum Objektrand heller und kleine Abstände dunkler dargestellt werden.

Bild c zeigt das sich daraus ergebende „Skelett“ des Bildobjekts in Form von Linien mit 1 Pixel Dicke.

Uneindeutige Skelettierung



Dass die Skelettierung von Bildobjekten nicht zwingend eindeutig sein muss, ist daran zu erkennen, dass hier zwei deutlich unterschiedliche Objektformen bei der Skelettierung zum selben Ergebnis führen.

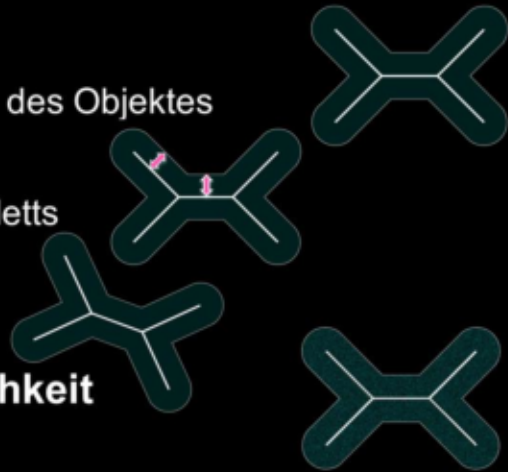
- Merkmalsextraktion
- Objekterkennung
 - z.B. OCR, Tracking, ...
- Animation



Typische Anwendungsgebiete für die Skelettierung finden sich in der Extraktion von typischen Merkmalen, z.B. zur Vorbereitung einer semantischen Bildanalyse, der Objekterkennung, z.B. zur Texterkennung (Optical Character Recognition = OCR) in gescannten Buchseiten oder zur Vorbereitung von Animationen durch automatische Definition eines Skeletts für zu animierende Objektformen.

Ideale Skelett-Eigenschaften

- **Homotopie**
 - Erhalten der Topologie des Objektes
- **Mittigkeit**
 - Zentrale Lage des Skeletts
- **Rotationsinvarianz**
- **Dicke: 1 Pixel**
- **Rauschunempfindlichkeit**
- **Geschwindigkeit**
 - schnelle Berechnung



Methoden zur Skelettierung

Voraussetzung:
nur **ein Objekt** im Bild vorhanden

- **Distanztransformation**
 - z.B. Mittelachsentransformation
- **Thinning**
 - Zhang & Suen
 - u.a.
- **Voronoi-Skelettierung**

verschiedene Algorithmen
→ verschiedene Skelette

- **Mittelachse:**
Menge der Pixel, deren zwei **kürzeste Abstände zum Objektrand** gleich sind

- Berechnung über **Distanztransformation**
 - mittels **Distanzfunktionen**
 - oder durch mehrfach angewandte **Erosion**

- Graustufenwert eines Objektpixels = kürzeste Distanz zum Objektrand

BLUM MAT

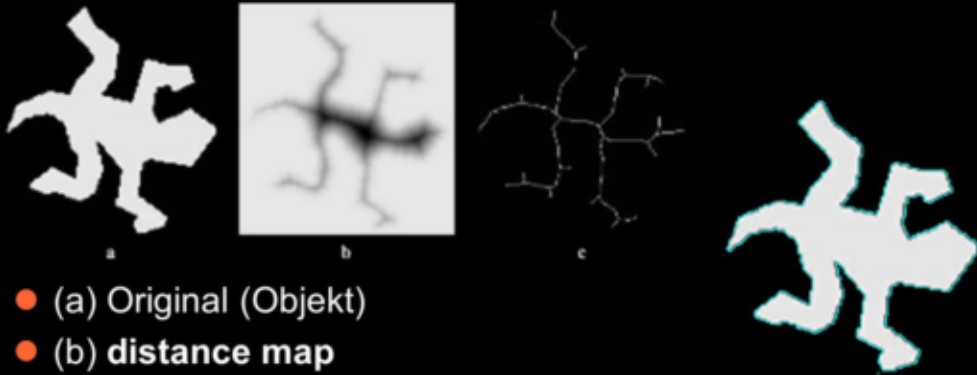
- For any point P in the object
 - Locate the closest boundary point, if there is more than one at this minimum distance – It's a skeletal point



Für jeden Punkt im Bild, der zum Objekt gehört, den Abstand zum Rand berechnen → Grauwert (Hintergrund schwarz)

Falls es für ein Pixel mehr als einen Randpunkt mit dem minimalen Abstand gibt, ist das ein Punkt auf dem Skelett.

Beispiel Mittelachsentransformation



- (a) Original (Objekt)
- (b) **distance map**
 - Je heller der Graustufenwert eines Pixels, desto größer ist der Abstand zum **Objektrand**.
- (c) Skelett
 - Gradientenzentrum (vgl. Canny: non-maximum suppression)

In Punkt (c) offenbart sich ein Unterschied zur vorherigen Folie: Dort erfolgt die Berechnung zweier Abstände zum Objektrand, woraus dann über Zugehörigkeit oder nicht-Zugehörigkeit zum Skelett entschieden wird.

Alternativ entsteht das Skelett hier durch einen Vorgang ähnlich der non-maximum suppression beim Canny-Kantenfilter-Algorithmus. Nur die höchsten Werte entlang eines lokalen Helligkeitsgradienten bleiben als Skelettpixel erhalten.

Distanztransformation: Berechnung

- Distanztransformation:
Kürzester Abstand zum Objektrand

$$D(p) = \min\{d(p, q) | I(q) = 0\}$$

für alle Pixel p des Objekts

$$O = \{p | I(p) = 1\}$$

- *Beispiel:*

- (a) Objekt = alle schwarzen Pixel

- (b) distance map: $D(p)$

Alle Hintergrundpixel q werden schwarz gezeichnet.



- Distanzfunktion $d(p, q)$

(a): (Bemerkung: Das Bild ist der Übersichtlichkeit wegen invertiert. D.h. schwarze Pixel gehören zum Objekt, weiße nicht.)

Distanzfunktion: abhängig von gewählter Abstandsmetrik

Nachbarschafts-Metriken (Wiederholung)

- **Nachbarschaften** eines Bildpunktes (x, y) :

- 4-er Nachbarschaft (gemeinsame Kante)
- 8-er Nachbarschaft (Kante od. Ecke)



- **Abstand** zwischen Bildpunkten (x_1, y_1) und (x_2, y_2) :

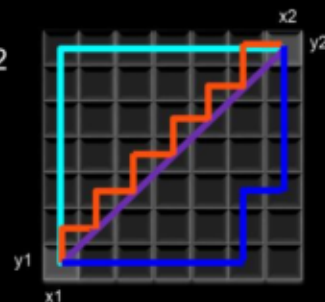
- **Euklidische Metrik**

- 4-er $d_e=1$ und diagonale Nachbarn in 8-er $d_e \leq \sqrt{2}$

- **Cityblock-Metrik**

- 4-er $d_c=1$ und diagonale Nachbarn in 8-er $d_c = 2$

- **Schachbrett-Metrik**, 8-er $d_s=1$



Bei der Betrachtung der Nachbarschaft eines Bildpunktes (z.B. bei Maskenoperationen) kann es sich als wichtig erweisen, den Einfluss benachbarter Pixel von deren Abstand zum Referenzpixel abhängig zu machen. Je nach Betrachtungsweise bzw. Entstehungskontext des Bildes stehen unterschiedliche Metriken zur Berechnung dieser Abstände zur Verfügung.

Cityblock-Metrik = Manhattan-Metrik = Taxi-Metrik: Distanz zweier Punkte als Summe der absoluten Differenzen der Einzelkoordinaten \rightarrow diskrete Ortspunkte, "Umfahren von Häuserblocks"

d_e = Euclidian distance (diagonal: Pythagoras)

d_c = cityblock distance (in ganzen Schritten von Pixelzentrum zu Pixelzentrum entlang der Bildachsen)

d_s = chessboard distance (vgl. Zugoptionen des „Königs“ als Schachfigur)

Distanzfunktionen

- **Euklidische Distanz** $d_{\text{euklid}}(p, q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$
- **City-Block-Distanz** $d_{\text{city}}(p, q) = |p_x - q_x| + |p_y - q_y|$
- **Chessboard-Distanz** $d_{\text{chess}}(p, q) = \max\{|p_x - q_x|, |p_y - q_y|\}$

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	$\sqrt{2}$	2	$\sqrt{2}$	1	0
0	1	2	$\sqrt{8}$	2	1	0
0	1	$\sqrt{2}$	2	$\sqrt{2}$	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	2	2	2	1	0
0	1	2	3	2	1	0
0	1	2	2	2	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	2	1	1	0
0	1	2	2	2	1	0
0	1	1	2	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Binärbild

Euklid

City

Chessboard

Für ein einfaches 7x7-Binärbild wird nach 3 verschiedenen gebräuchlichen Abstandsmetriken die jeweilige Distance map bestimmt.

Jedes Ergebnis-Pixel trägt als Wert den kleinsten Abstand gemäß der jeweiligen Metrik zum Objektrand. Hintergrundpixel des Originalbilds behalten den Wert 0.

Distance Map via Erosion

- Mehrmaliges Ausführen der **Erosion** mit 3x3-Strukturelement
- Anzahl bisher durchgeführter Erosionen → Graustufenwert des zu löschenden Pixels
- Distance map ist berechnet, sobald alle Objektpixel gelöscht sind.



0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0



0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	2	1	1	0
0	1	2	2	2	1	0
0	1	1	2	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0



Alternativ zur Mittelachsen-Transformation mit anschließender Definition von Skelettpixeln über Doppel-Abstände oder Unterdrückung von Nicht-Maxima (non-maximum suppression, „nms“) können Abstandskarten auch durch Protokollieren eines iterativen Erosionsvorgangs erstellt werden. Die Übertragung in ein Formskelett ähnelt dann der nms, indem nämlich nur die Pixel mit dem höchsten Zähler als dem Skelett zugehörig angesehen werden.

Mittelachsentransformation auf Distance Map

- Anwenden des **Laplace-Operators** auf distance map

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

0	1	0
1	-4	1
0	1	0

Laplace Operator:
Faltungskern für die 2.
Ableitung

- **Schwellwertberechnung** für Skelettextraktion oder Clipping
- **Thinning** mit morphologischen Operationen

Aus der Distance Map, die zunächst als Graustufenbild mit an- und absteigenden Gradienten vorliegt, lässt sich durch eine geeignete Operation die zentral gelegene Achse herausarbeiten.

Für eine solche Operation zur „Mittelachsentransformation“ kann z.B. der Laplace-Operator eingesetzt werden, der die 2. Ableitung der Ortsfunktion berechnet und auch als Faltungsfiler zum Hervorheben von Kanten in Bildern gebräuchlich ist.

Alternativ kann die zentrale Achse auch über einen Schwellwert aus der Distance Map geschält werden oder durch Ausdünnungsoperationen mithilfe von morphologischen Filtern.

Laplace-Operator (Wiederholung)

$$\nabla^2 I(u, v) = I(u+1, v) + I(u-1, v) - 4I(u, v) + I(u, v+1) + I(u, v-1)$$

Filtermatrix:

$$h_{lp}(i, j) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Drehung um 45° :

$$h_{lp_{45}}(i, j) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$



Laplace-Operator: Kantenerkennung über 2. Ableitung der Ortsfunktion

Thinning

- morphologische Operation
- Randpixel iterativ abtragen
- Ziel: Reduktion des Objekts auf Linienzüge (1 Pixel breit)
 - Erkenne Pixel, die zum Skelett gehören und ...
 - entferne die restlichen Pixel.
- Abbruch der Iteration: stabiler Zustand
 - Anwenden der Thinning-Operation, bis keine Veränderung zwischen Eingangsbild und Ergebnisbild mehr erfolgt.

Wir erinnern uns an den Vorgang beim Morphologischen Thinning:
Zentralpunkt entfernen, wo Hit&Miss erfolgreich

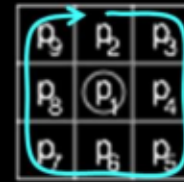
Iterativ angewandt, kann Thinning zur Skelettierung genutzt werden.

Thinning-Algorithmen

- Verschiedene Thinning-Algorithmen liefern unterschiedliche Skelette.
- Bsp.: Algorithmus von **Zhang und Suen**
 - T. Y. Zhang and C. Y. Suen. 1984. A fast parallel algorithm for thinning digital patterns. Commun. ACM 27, 3 (March 1984), 236-239.
DOI=10.1145/357994.358023
<http://doi.acm.org/10.1145/357994.358023>
- *weitere Algorithmen siehe Erhard, Kap. 9*

Thinning-Algorithmus nach Zhang & Suen (1)

- schnell und einfach
- gut parallelisierbar
- Strukturelement 3x3
- je zwei Subschritte pro Iteration
- Terminierung durch Erreichen eines stabilen Zustandes
- qualitativ gute Skelettierung
- typisches Einsatzgebiet: Schrifterkennung
 - OCR = Optical Character Recognition



Der Thinning-Algorithmus nach Zhang und Suen beruht auf einem mehrstufigen Thinning-Verfahren.

In dem 3x3-Strukturelement, das den zentralen Hot Spot (p_1) aus den Nachbarschaftsbetrachtungen ausspart, werden die Nachbarpixel im Uhrzeigersinn durchnummeriert (von p_2 bis p_9).

Mithilfe dieses Strukturelements werden zwei Check-Parameter ermittelt: $A(p_1)$ und $B(p_1)$.

Thinning-Algorithmus nach Zhang & Suen (2)

- Abfragen verschiedener Bedingungen via 3x3-Strukturelement



- $B(p_1)$: Anzahl der Objektpixel unter $p_2 - p_9$



$$B(p_1) = 0$$



$$B(p_1) = 3$$



$$B(p_1) = 5$$

Der Parameter $B(p_1)$ für das Verfahren ergibt sich aus der Anzahl zum Objekt gehöriger Pixel unter den Nachbarzellen des Strukturelements (d.h. min. 0, max. 8).

Thinning-Algorithmus nach Zhang & Suen (3)

- $A(p_1)$: Anzahl der $0 \rightarrow 1$ -Übergänge von p_2 nach p_2 via p_9 (im Uhrzeigersinn)**

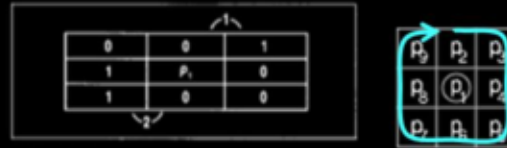


FIGURE 2. Counting the 01 patterns in the ordered set $p_2, p_1, p_8, \dots, p_6, p_5$.



$$A(p_1) = 0$$



$$A(p_1) = 2$$



$$A(p_1) = 1$$



$$A(p_1) = 0$$



$$A(p_1) = 4$$



$$A(p_1) = 1$$

Der Parameter $A(p_1)$ wird bestimmt als Anzahl der Übergänge von Nicht-Objektpixeln zu Objektpixeln im Uhrzeigersinn um den zentralen Hot Spot, p_1 , herum.

Übergänge ($0 \rightarrow 1$), die p_1 enthalten, zählen dabei nicht.

Die Betrachtung erfolgt zyklisch im Uhrzeigersinn (beginnend mit p_2), der letzte Schritt ($p_9 \rightarrow p_2$) schließt den Kreis.

Thinning-Algorithmus nach Zhang & Suen (4)

• Erste Subiteration:

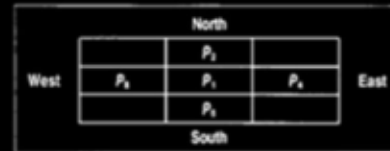
- von links oben nach rechts unten
- löscht südliche und östliche Randpixel
- löscht nord-westliche Eckpixel



nord-westlicher Eckpixel

östlicher Randpixel

südlicher Randpixel



Die Auswirkungen der einzelnen Teilschritte dieses Thinning-Verfahrens lassen sich leichter beschreiben, wenn die Position der Nachbarpixel des zentralen Hot Spots über Himmelsrichtungen benannt werden.

Die erste Subiteration wirkt sich so aus, als würde der Abtrag von Pixeln das Objekt von links oben nach rechts unten abschleifen. Die andere Diagonale bleibt unangetastet.

In den Beispielbildern zählen schwarze Pixel zum Objekt, weiße nicht. In den drei Beispielsituationen wird jeweils der runde schwarze Pixel durch das Verfahren gelöscht.

Thinning-Algorithmus nach Zhang & Suen (5)

• Erste Subiteration:

Durchlaufe Bild von links oben nach rechts unten

• Lösche Pixel p_1 , wenn folgende Bedingungen erfüllt:

- $2 \leq B(p_1) \leq 6$
- $A(p_1) = 1$
- $p_2 \cdot p_4 \cdot p_6 = 0$
- $p_4 \cdot p_6 \cdot p_8 = 0$



$$A(p_1) \neq 1$$



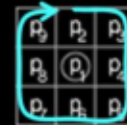
$$B(p_1) = 1$$



$$p_2 \cdot p_4 \cdot p_6 \neq 0$$

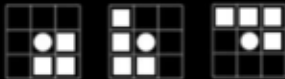


$$p_4 \cdot p_6 \cdot p_8 = 0$$



$A(p_1)$: 01-Übergänge
von p_2 nach p_9

$B(p_1)$: Objektpixel
unter $p_2 - p_9$



→ lösche Pixel p_1

Auch hier gilt: Übergänge ($0 \rightarrow 1$ od. $1 \rightarrow 0$), die p_1 enthalten, zählen nicht.

Die hier dargestellten Regeln weichen teilweise von Darstellungen an anderer Stelle (durch andere Autoren) ab, orientieren sich aber an der Original-Publikation und haben sich (im Gegenteil zu manchen anderen Varianten) nach gründlicher Überprüfung als funktionstüchtig erwiesen.

[abweichend von Literatur: Ehrhardt, S. 202: Dort steht: $(p_2=0) \wedge (p_4=0) \wedge (p_6=0)$ und entsprechend mit p_4, p_6, p_8 :

Multiplikation wäre „oder“; nach Ehrhardt „und“; im Original-Paper steht die „or“-Verknüpfung via Multiplikation]

Subiteration 1

$$- 2 \leq B(p_1) \leq 6$$

$$- Cn(p_1) = 1$$

$$- (p_2 = 0) \vee (p_4 = 0) \vee (p_6 = 0)$$

$$- (p_4 = 0) \vee (p_6 = 0) \vee (p_8 = 0)$$

Wenn alle Pixel des Bildes bearbeitet sind, beginnt Subiteration 2 (mit dem Ergebnis von SI1) – Details siehe spätere Folie

Thinning-Algorithmus nach Zhang & Suen (6)

● Zweite Subiteration:

- von rechts unten nach links oben
- löscht nördliche und westliche Randpixel
- löscht süd-östliche Eckpixel



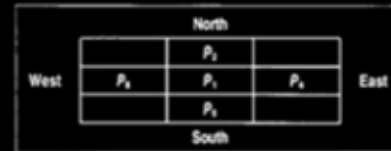
süd-östlicher Eckpixel



westlicher Randpixel



nördlicher Randpixel



Die zweite Subiteration nutzt als Input das Ergebnis der ersten und geht (hinsichtlich des Ergebnisses) in umgekehrter Richtung vor:

Subiteration 2 (mit dem Ergebnis von SI1)

- $2 \leq B(p1) \leq 6$
- $Cn(p1) = 1$
- $(p2 = 0) \vee (p4 = 0) \vee (p8 = 0)$
- $(p2 = 0) \vee (p6 = 0) \vee (p8 = 0)$

Thinning-Algorithmus nach Zhang & Suen (7)

● Zweite Subiteration:

- Durchlaufe Ergebnisbild der ersten Subiteration von rechts unten nach links oben

● Lösche Pixel p_1 , wenn folgende Bedingungen erfüllt:

- $2 \leq B(p_1) \leq 6$
- $A(p_1) = 1$
- $p_2 \cdot p_4 \cdot p_8 = 0$
- $p_2 \cdot p_6 \cdot p_8 = 0$



$$A(p_1) \neq 1$$



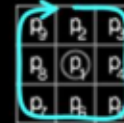
$$B(p_1) = 1$$



$$p_2 \cdot p_4 \cdot p_8 \neq 0$$



$$p_2 \cdot p_6 \cdot p_8 = 0$$



$A(p_1)$: 01-Übergänge von p_2 nach p_9

$B(p_1)$: Objektpixel unter $p_2 - p_9$



→ lösche Pixel p_1

wichtig: Die zweite Subiteration verwendet das Ergebnis der ersten als Ausgangsbild.
(Sonst wäre es auch sinnlos, zwei getrennte Subiterationen nacheinander durchzuführen, statt beide Kriterienkombinationen in einem Durchlauf zu prüfen.)

Thinning-Algorithmus nach Zhang & Suen (8)

- *Beispiel:*

- Führe mit dem Ergebnisbild der zweiten Subiteration die Berechnung erneut durch, bis keine Änderungen mehr auftreten.



Das Beispiel zeigt die Unempfindlichkeit des Algorithmus' bei der Skelettierung geringfügig voneinander abweichenden Abbildungen des Buchstaben „H“.

Eine sehr grobe Darstellung einer humanoiden Figur wird ebenfalls überzeugend skelettisiert.

Probleme bei Skelettierung

- **Bildrauschen**
(variiert zwischen
verschiedenen
Algorithmen)
- Qualität der **Segmentierung**
ausschlaggebend



Ein verrauschtes Bild oder Glanzlichter in der komplexen Form eines Telefonhörers können gravierende Auswirkungen auf das Ergebnis einer Skelettierung haben.

Kompetenzcheck

- **Skelettierung von Bildobjekten**
 - Prinzip, Anwendung
 - Eigenschaften
- **Distanztransformation**
 - Distance Map
 - Mittelachsentransformation
- **Thinning**
 - Prinzip
 - Thinning-Algorithmus nach Zhang & Suen

