Übersicht der Vorlesung

1. Einführung
2. Bildverarbeitung
3. Morphologische Operationen
4. Bildsegmentierung
5. Merkmale von Objekten
6. Klassifikation
7. Dreidimensionale Bildinterpretation
8. Bewegungsanalyse aus Bildfolgen
9. PCA (Hauptkomponentenanalyse)
10. ICA (Independent Component Analysis – Unabhängigkeitsanalyse)

4. Bildsegmentierung

4.6 Kantenorientierte Segmentierung

4.6.1 Überblick

Überblick

- **Ziel:** Finden der Randkanten von Gebieten

  → Sobald man die Kanten gefunden hat, erhält man auch eine Segmentierung des Bildes.

Kern – Rand

Nachbarschaftsstruktur: $[P, N]$ $M \subseteq P$ $p \in M$

p Kernpunkt von M: $N(p) \subseteq M$

Nachbarn von p

Menge aller Kernpunkte von M: $K(M)$

Randpunkt von M: $q \in M$, $q \notin K(M)$

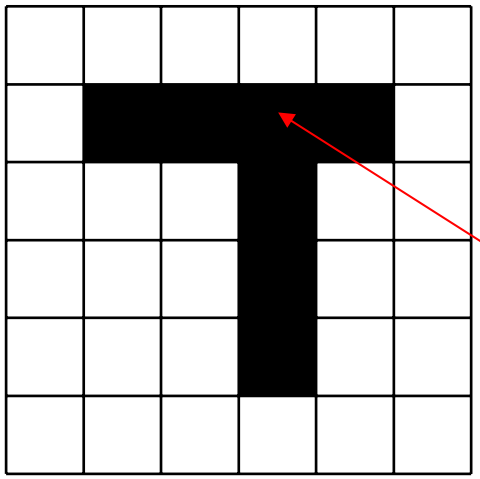
Menge aller Randpunkte von M: $R(M)$

eindimensionale Menge (Kante)

Nachbarschaftsstruktur: $[P, N]$ $K \subseteq P$

zusammenhängend

K heißt eindimensional: $\forall p \in K: \; |N(p) \cap K| \leq 2$

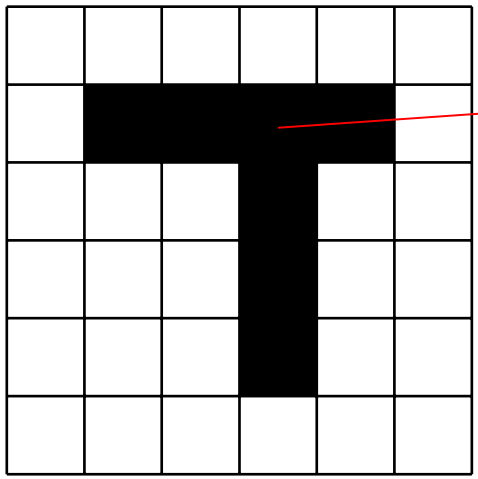


Problem bei verzweigten Kanten

Ein Pixel hat 3 Nachbarn

eindimensional – andere Möglichkeit für Bildraster

$\forall p \in K:$ keine zwei Punkte der Menge $N_8(p) \cap K$ sind benachbart bezüglich N_4



erfüllt

Kantenorientierte Segmentierung

Nachbarschaftsstruktur: $[P, N]$

Kantenorientierte Segmentierung: $\{K_1, K_2, \ldots, K_n\}$ $K_i \cap K_j = \emptyset, \; i \neq j$

eindimensional

existiert Zerlegung von *P*: $Z = \{X_1, \ldots, X_m\}$

$$\bigcup_{j=1}^{m} R(X_j) = \bigcup_{i=1}^{n} K_i$$

3 Teilaufgaben

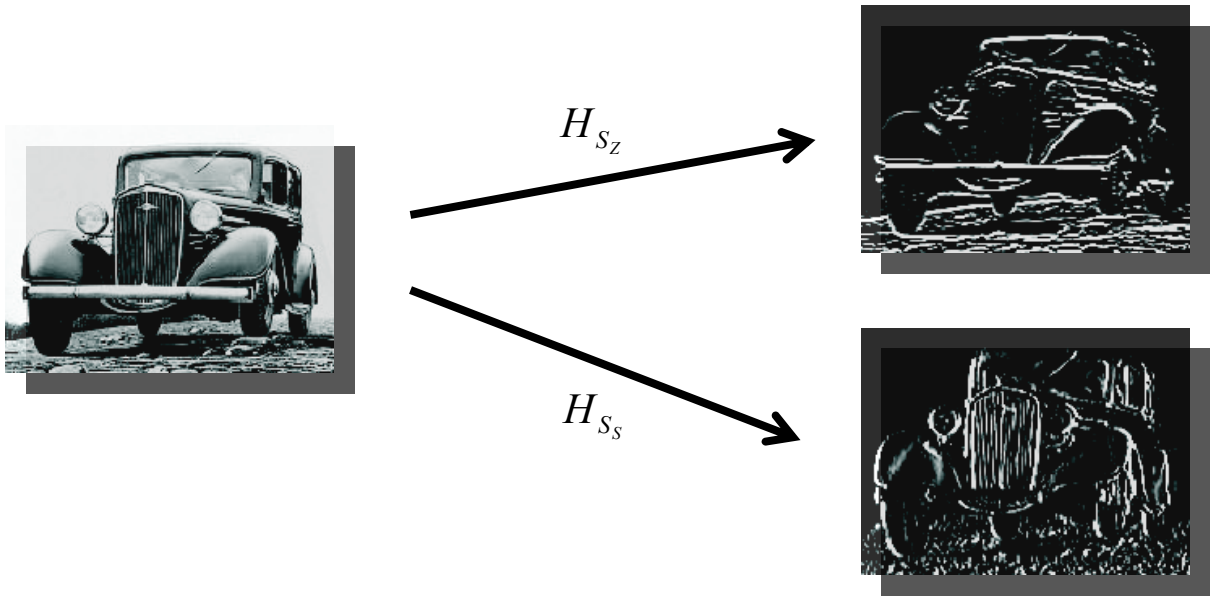
Im Folgenden sei *[P,N]* immer ein Bild.

Ziel: $\{K_1, K_2, \ldots, K_n\}$

- Kantendetektion
  - man erhält eine Menge *K* kantenverdächtiger Punkte (Kapitel 2)
- Kantenverdünnung
  - es wird eine eindimensionale Teilmenge von *K* erzeugt
- Kantenverfolgung
  - bisher gefundene Kantensegmente werden verlängert bzw. geschlossen

4.6.2 Kantendetektion

Kantendetektion

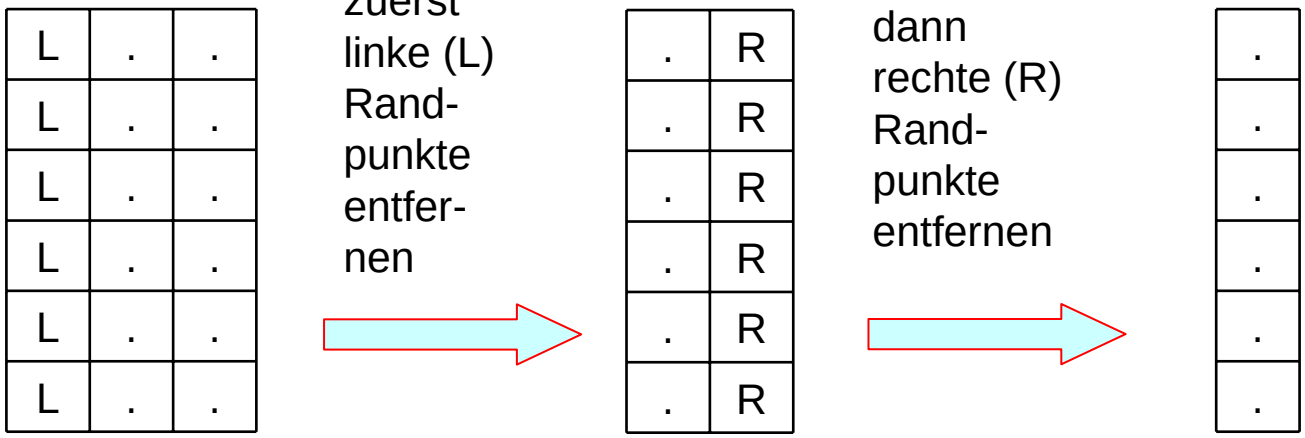


4.6.3 Kantenverdünnung

Kantenverdünnung

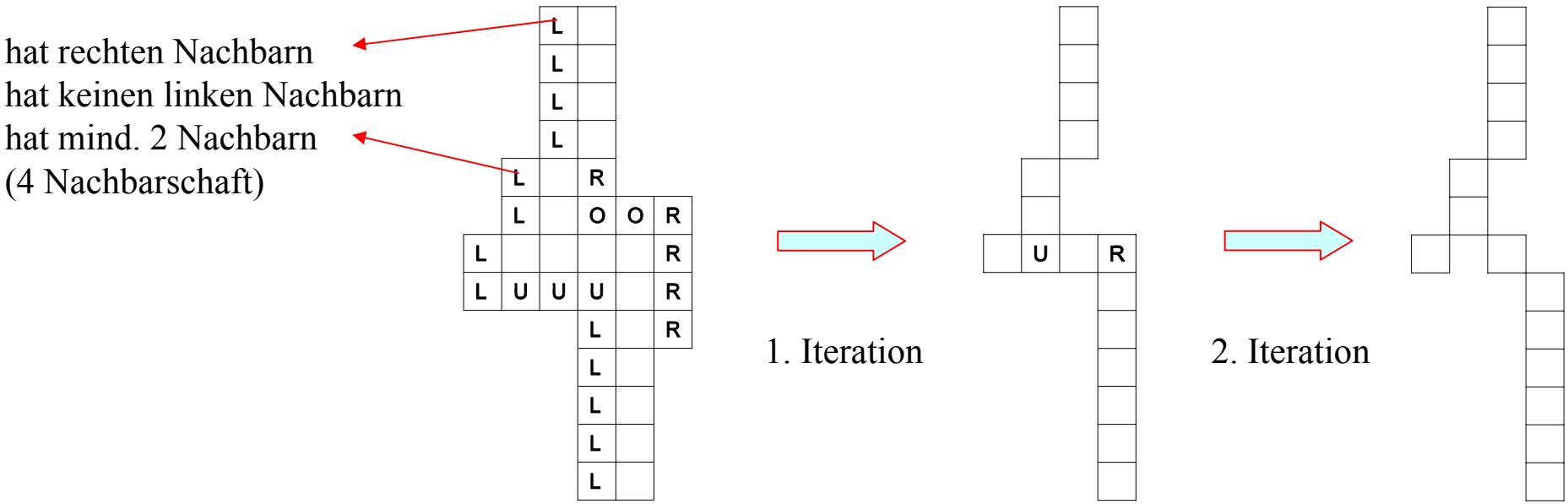
- Die Menge *K* der kantenverdächtigen Punkte enthält i.a. noch Anhäufungen von Kantenpunkten, die Kanten von mehr als 1 Bildpunkt Breite erzeugen (*K* ist nicht eindimensional).
- Der Zweck der Kantenverdünnung ist es, Punkte aus *K* so zu eliminieren, dass eindimensionale Kantensegmente entstehen.

einfaches Beispiel



weiteres Beispiel

- die Randpunkte werden in der Reihenfolge L (Links), R (Rechts), U (Unten), O (Oben) entfernt
- andere Reihenfolgen sind möglich
- Resultat ist von dieser Reihenfolge abhängig
- bei der angegebenen Reihenfolge werden vertikale Kantensegmente bevorzugt



4.6.4 Skelettierung

Skelettierung

- allgemeinere Möglichkeit der Kantenverdünnung
- mit morphologischen Operationen (Kapitel 3)
- Algorithmus von Lü und Wang für Binärbilder

Algorithmus von Lü und Wang

3 x 3 Maske:

P_1	P_2	P_3
P_8	P	P_4
P_7	P_6	P_5

$$P_i \in \{0,1\}$$

P gehört immer zum Segment (alle Einsen) und hat den Wert 1 (Binärbilder)

A(P) – Anzahl der Übergänge von 0 → 1 , wenn die Punkte $P_1,\ldots, P_8, P_1$ einmal durchlaufen werden.

0	0	1
1	1	0
1	0	0

A(P)=2
B(P)=3

B(P) – Anzahl der 1 unter den Punkten $P_1,\ldots, P_8$.

1	0	1
0	1	1
0	1	0

A(P)=3
B(P)=4

Algorithmus von Lü und Wang

Wir schieben die Maske in mehreren Iterationen über das Bild.

P wird im Segment gelöscht, wenn:

$$3 \le B(P) \le 6$$

$$A(P) = 1$$

$$P_4 \wedge P_6 \wedge (P_2 \vee P_8) = false$$

bei gerader Iteration (2.,4.,…)

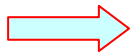
$$P_2 \wedge P_8 \wedge (P_4 \vee P_6) = false$$

bei ungerader Iteration (1.,3.,…)

Bei den letzten 2 Bedingungen betrachten wir P_i
als logische Variable mit den Werten true und false.

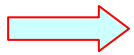
Beispiele

0	0	0
0	P	1
0	0	0



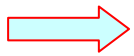
$B(P) = 1 \Rightarrow$ P wird nicht entfernt

1	0	0
1	P	1
0	0	0



$A(P) = 2 \Rightarrow$ P wird nicht entfernt

0	1	0
1	P	1
0	1	0



$P_4 \wedge P_6 \wedge (P_2 \vee P_8) = true$

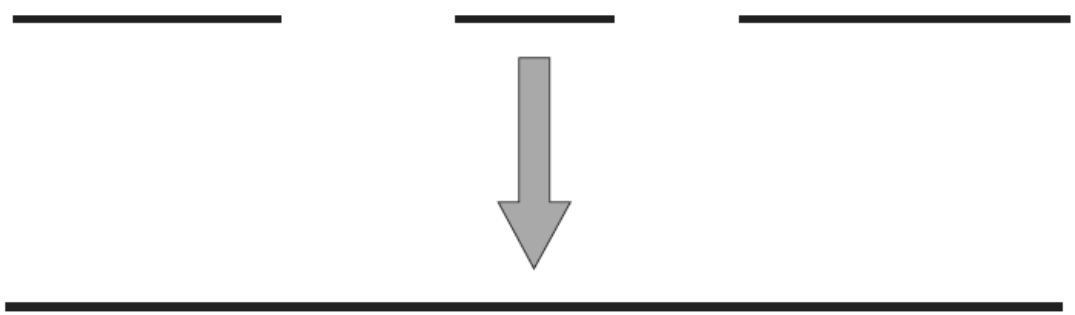
$P_2 \wedge P_8 \wedge (P_4 \vee P_6) = true$

P wird bei keiner Iteration entfernt

4.7 Kantenverfolgung

Kantenverfolgung

bisher gefundene Kantensegmente verlängern bzw. schließen
(kleinere Teile werden eventuell auch entfernt)



- 2 Aufgaben:
  - Wahl eines Startpunktes
  - Sukzessive wird versucht, weitere Punkte der Kante zu finden
- Informationen zur Beurteilung, ob ein weiterer Punkt zur Kante gehört:
  - aus Bild (z.B. Grauwertänderung)
  - a priori (z.B. Form der Kante)

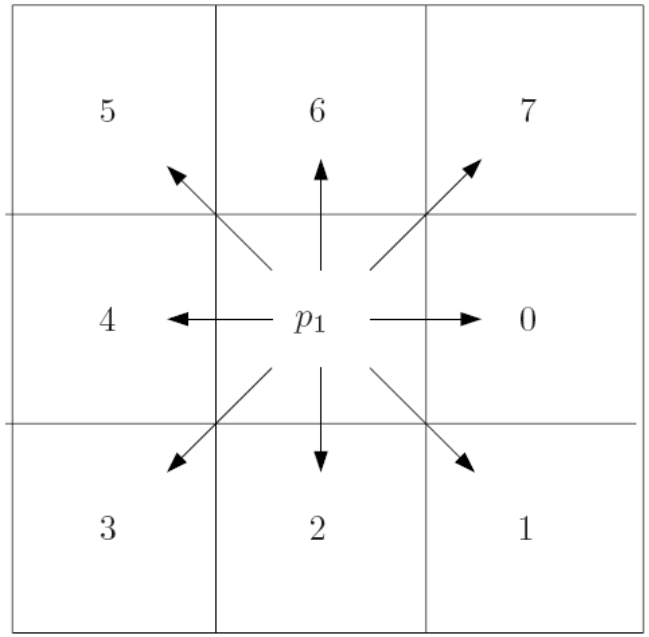
4.7.1 Freemancode

Freemancode (Richtung)

$$p_1, p_2 \in P \qquad (p_1, p_2) \in N_8$$

$$P = \{(i, j) : \; i = 0, \ldots, I-1, \;\; j = 0, \ldots, J-1\}$$

$$r(p_1, p_2) = \begin{cases} 0 & i_1 = i_2 \wedge j_1 = j_2 - 1 \\ 1 & i_1 = i_2 + 1 \wedge j_1 = j_2 - 1 \\ 2 & i_1 = i_2 + 1 \wedge j_1 = j_2 \\ 3 & i_1 = i_2 + 1 \wedge j_1 = j_2 + 1 \\ 4 & i_1 = i_2 \wedge j_1 = j_2 + 1 \\ 5 & i_1 = i_2 - 1 \wedge j_1 = j_2 + 1 \\ 6 & i_1 = i_2 - 1 \wedge j_1 = j_2 \\ 7 & i_1 = i_2 - 1 \wedge j_1 = j_2 - 1 \end{cases} \quad falls$$



Addition, Subtraktion von Freemancodes

2 Codes: $c_1, c_2 \in \{0,1,2,3,4,5,6,7\}$

Addition: $$c_1 \oplus c_2 = (c_1 + c_2) \bmod 8$$

Subtraktion: $$c_1 \ominus c_2 = (c_1 - c_2) \bmod 8$$

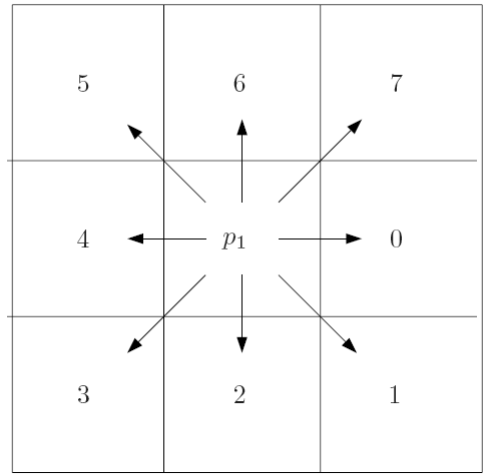
Winkeldifferenz

$c_1, c_2 \in \{0,1,2,3,4,5,6,7\}$

$$c_1 \angle c_2 = \begin{cases} (c_1 - c_2) \bmod 8 & falls \quad (c_1 - c_2) \bmod 8 \leq 4 \\ -[(c_2 - c_1)] \bmod 8 & \quad\quad\quad (c_1 - c_2) \bmod 8 > 4 \end{cases}$$

Beispiele:

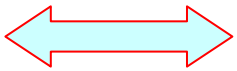
$$0 \angle 1 = -[1 \bmod 8] = -1$$
$$1 \angle 0 = 1 \bmod 8 = 1$$
$$0 \angle 7 = -7 \bmod 8 = 1$$
$$7 \angle 0 = -[-7 \bmod 8] = -1$$



Beschreibung eines Weges

Weg: $w = (p_0, p_1, \ldots, p_n) \quad p_i \in P \ (i = 0, \ldots, n)$

$(p_i, p_{i+1}) \in N_8 \ (i = 0, \ldots, n-1)$



$$w = (p_0, C) \quad C = (c_1, c_2, \ldots, c_n)$$

$$c_i = r(p_{i-1}, p_i) \ (i = 1, \ldots, n)$$

Startpunkt des Weges

Länge des Weges

$|w| = n$

Drehung eines Weges um einen Punkt

$$w = (p_0, C) \quad C = (c_1, c_2, \ldots, c_n)$$

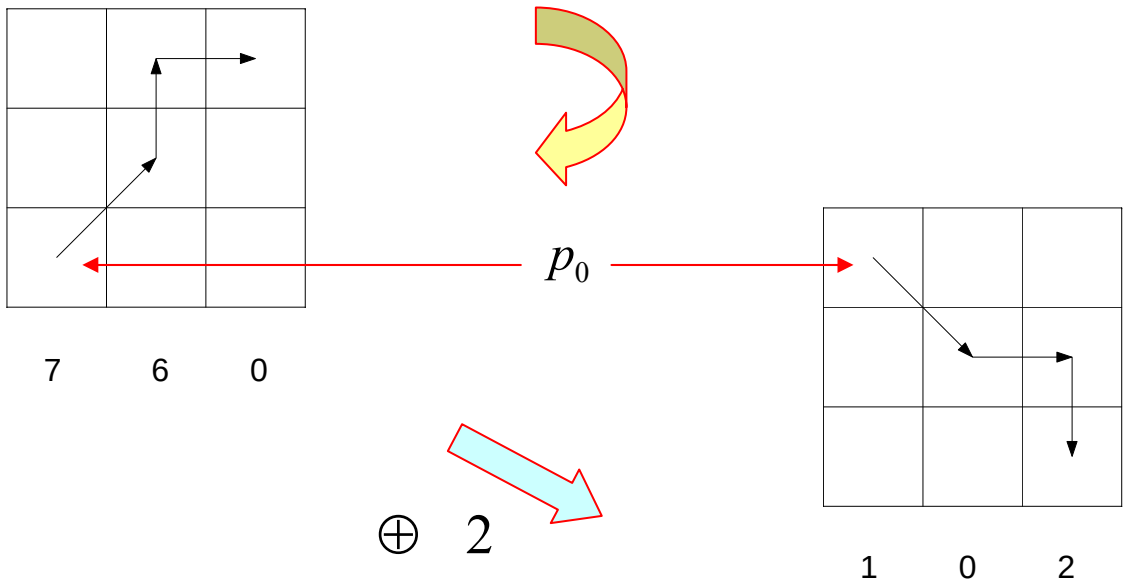
Drehpunkt

Drehwinkel: $m \cdot \frac{\pi}{4}$

$$w_D = (p_0, C_D) \quad C_D = (c_1 \oplus m, c_2 \oplus m, \ldots, c_n \oplus m)$$

Beispiel

Beispiel: Drehung um 90° (m=2)



4.7.2 Kantenverfolgung – Suchen von Wegen

Suchen von Wegen

- Kantenverfolgung kann als Finden eines Weges beschrieben werden
- vom Startpunkt p_0 werden immer wieder neue Nachbarpunkte bezüglich der 8 – Nachbarschaft zum Weg hinzufügt, bis ein geeigneter Weg gefunden ist
- Endpunkt des Weges kann vorgegeben sein oder man sucht einen Weg einer bestimmten Länge k

- insgesamt 8^k mögliche Wege der Länge k
  (Suchmenge muss eingeschränkt werden)
  - Einschränken der lokale Wegkrümmung
  - eingeschränkter Winkelbereich um den Startpunkt

Einschränken der lokalen Wegkrümmung

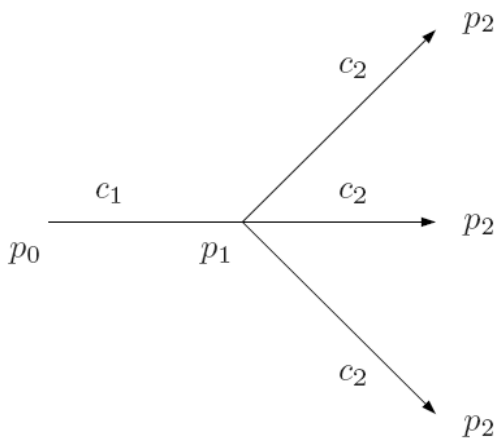
$$w = (p_0, C) \quad C = (c_1, c_2, \ldots, c_k)$$

$$c_i = r(p_{i-1}, p_i) \ (i = 1, \ldots, k)$$

$$\left| c_{i-1} \angle c_i \right| \leq a \in \{1,2\} \quad i = 2, \ldots, k$$

Beispiel:

$$a = 1, c_1 = 0, k = 2$$



		P_2
P_0	P_1	P_2
		P_2

3 Wege

Einschränken der lokalen Wegkrümmung

Beispiel:

$a = 1, c_1 = 0, k = 3$

				P_3
			P_2	P_3
	P_0	P_1	P_2	P_3
			P_2	P_3
				P_3

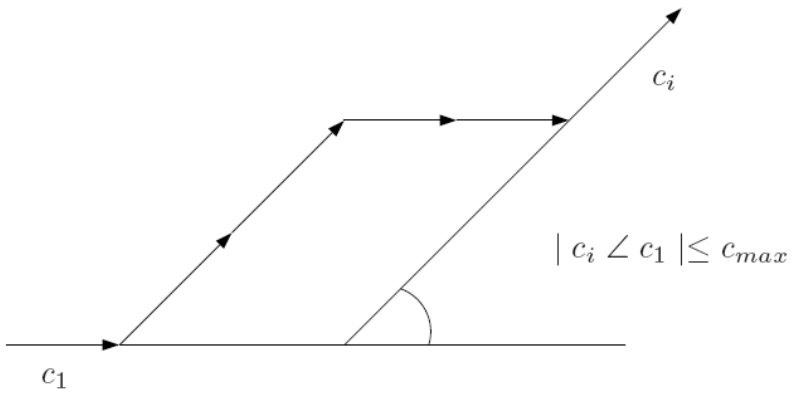
9 Wege

eingeschränkter Winkelbereich um den Startpunkt

$$w = (p_0, C) \quad C = (c_1, c_2, \ldots, c_k)$$

$$c_i = r(p_{i-1}, p_i) \quad (i = 1, \ldots, k)$$

$$\left| c_i \angle c_1 \right| \leq c_{\max} \in \{1,2,3\} \quad i = 2, \ldots, k$$



4.7.3 Aufbau einer Kostenfunktion und Anwendung von Suchverfahren

Kostenfunktion

- Knotenmenge $\{p_0, \cdots, p_k\}$
  und Kantenmenge $\{c_1, \cdots, c_k\}$
  bilden Graph
- Gesucht ist ein möglichst optimaler Weg bezüglich einer Kostenfunktion von p_0 zu p_k
- Die Kostenfunktion enthält Angaben über die Grauwertänderung entlang des Weges und über die Form des Weges

Beispiel einer Kostenfunktion

$$K(w) = \frac{g_1}{k} \sum_{i=1}^{k} \frac{1}{1+|\alpha_i|} + \frac{g_2}{k-d+1} \sum_{i=1}^{k-d+1} |k_i|$$

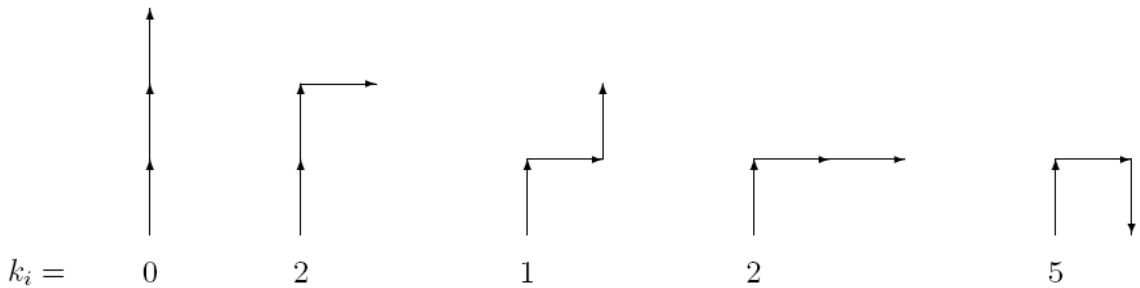
möglichst gering

Weglänge

Gewichtsfaktoren

Grauwertdifferenz am *i*-ten Wegelement

lokale Krümmung am *i*-ten Wegelement
z.B. für $d = 3$ (5 Möglichkeiten)



4.8 Gebietsnachbarschaftsgraph

Gebietsnachbarschaftsgraph

- Zusammenhang zwischen den Segmenten
- Beschreibung mittels Graphen
- Knoten entsprechen den Segmenten
- Kanten beschreiben geometrische Relationen zwischen den Segmenten
  - benachbart
  - umgibt
  - links (rechts) von
  - über
- diese Graphen werden bei der Objektklassifikation benutzt

Gebietsnachbarschaftsgraph

Nachbarschaftsstruktur: $[P, N]$

Homogenitätsfunktion: $h : 2^P \rightarrow \{true, false\}$

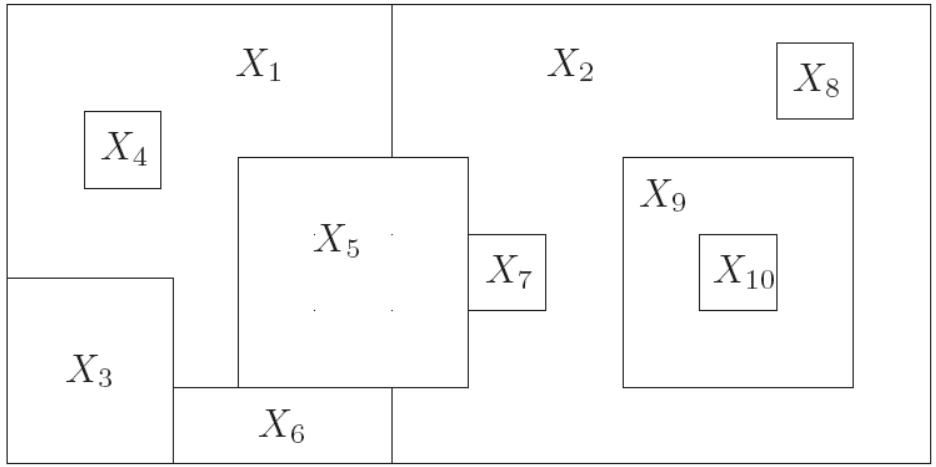
Segmentierung bezüglich *h*: $Z_S = \{X_1, \ldots, X_n\}$

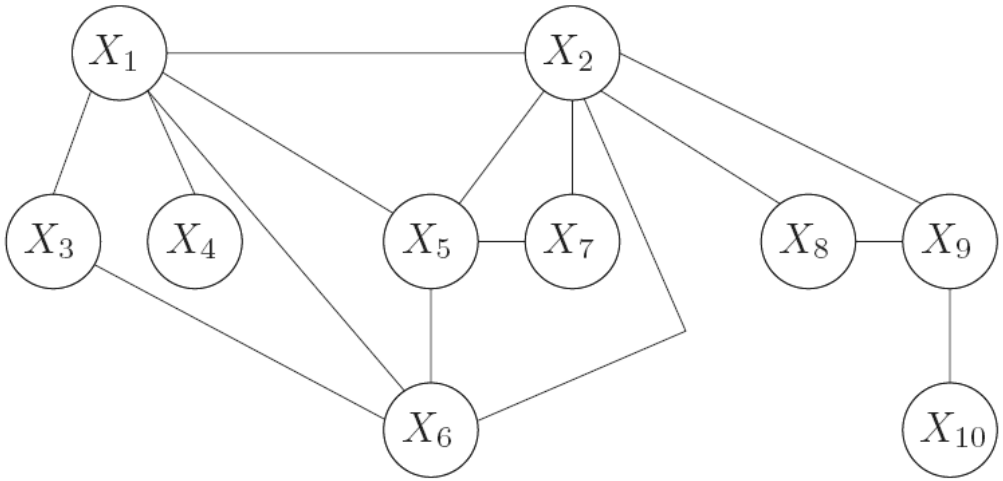
Gebietsnachbarschaftsgraph: $G_N = [Z_S, K_N]$

Knoten Kanten

$(X_i, X_j) \in K_N \leftrightarrow X_i$ und X_j sind benachbart

Beispiel





Gebietshierarchie

Nachbarschaftsstruktur: $[P, N]$

$P = \{(i, j) : i = 0, \ldots, I-1, \; j = 0, \ldots, J-1\}$ $N = N_4$ oder N_8

Homogenitätsfunktion: $h : 2^P \rightarrow \{true, false\}$

Segmentierung bezüglich h: $Z_S = \{X_1, \ldots, X_n\}$

Gebietshierarchie: $G_H = [Z_S, K_H]$

$(X_i, X_j) \in K_H \leftrightarrow (X_i, X_j) \in K_N$ und X_i umgibt X_j

exakte Definition noch notwendig

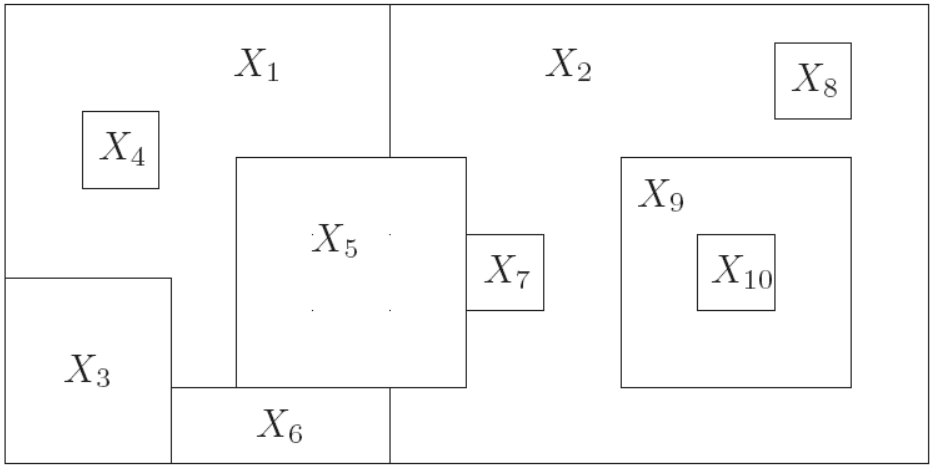
Umgibtgraph

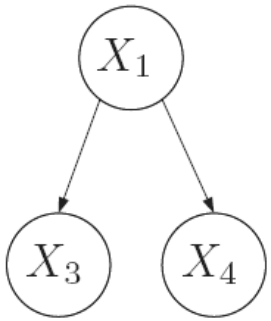
Gebietshierarchie: $G_H = [Z_S, K_H]$

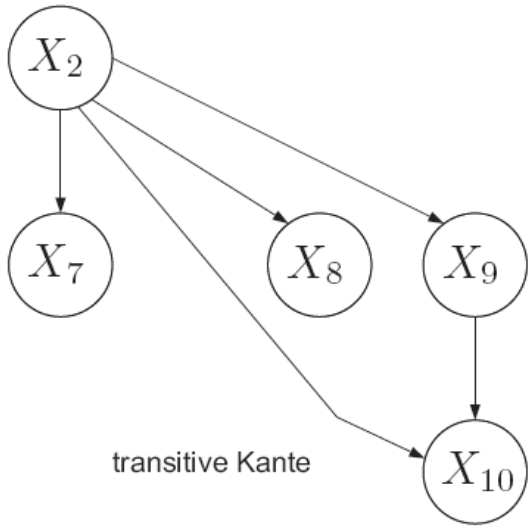
Umgibtgraph: $G_U = [Z_S, K_U]$

transitive Hülle von K_H

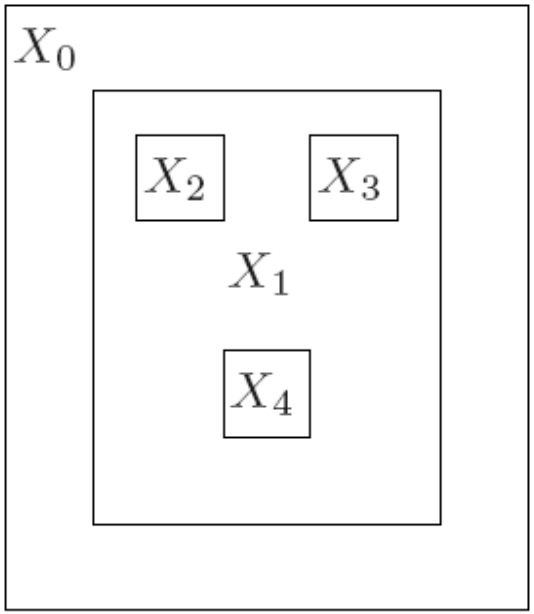
Beispiel

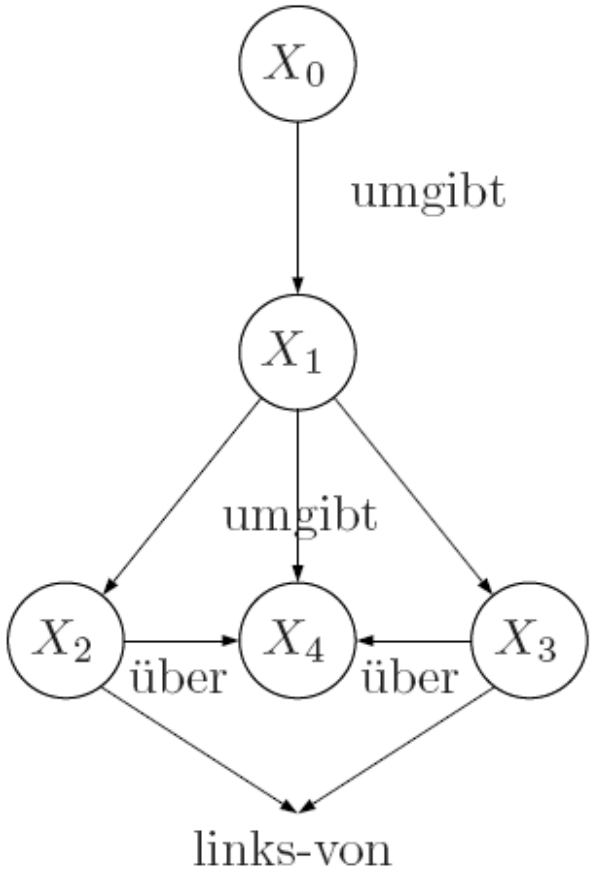

X_1
X_2
X_8
X_4
X_5
X_9
X_7
X_{10}
X_3
X_6


X_1
X_3
X_4


X_2
X_7
X_8
X_9
transitive Kante
X_{10}

Weitere Relationen





4.9 Modellabhängige Verfahren zur Segmentierung

Modellabhängige Verfahren zur Segmentierung

- Wissen über die geometrische Form der gesuchten Segmente wird benutzt

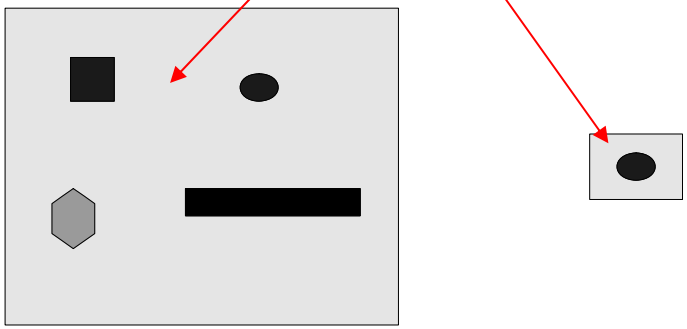
- Verfahren:
  - Matchen
  - Hough – Transformation

4.9.1 Matchen

Matchen

Eingabebild: $G_E = (g_E(i,j)),\ i = 0,\ldots,I-1,\ j = 0,\ldots,J-1$

Muster: $A = (a(i_1, j_1)),\ i_1 = 0,\ldots,a_1,\ j_1 = 0,\ldots,a_2$



$$(l,m):\ l = 0,\ldots,I-1,\ m = 0,\ldots,J-1$$

$$d(l,m) = \sum_{(i,j)\in I_{lm}} [g_E(i,j) - a(i-l, j-m)]^2$$

$$I_{lm} = \{(i,j) : i-l \in \{0,\cdots,a_1\}\ \ j-m \in \{0,\cdots,a_2\}\}$$

Muster ist im Bild vorhanden und befindet sich am Ort (l,m), wenn:

$$d(l,m) < T_0$$

T_0: vorgegebener Schwellwert

4.9.2 Hough – Transformation

Hough – Transformation

- wird oft zur Detektion von Geraden im Bild benutzt
- dadurch kann man Segmente finden, die durch geradlinige Kanten begrenzt werden
- aber auch Kreise oder Ellipsen können gefunden werden

Allgemeines Vorgehen

$$p = (p_1, \ldots, p_n) \in R^n$$

Vektor von Parametern zur Beschreibung eines Objektes

repräsentiert ein konkretes Objekt,
z.B. eine Gerade oder einen Kreis in der Ebene R^2

binäres Eingabebild:

$$G_E = (g_E(i,j)),\ i = 0,\ldots,I-1,\ \ j = 0,\ldots,J-1$$

$$g_E(i,j) \in \{0,1\}$$

Die Punkte (i,j) mit $g_E(i,j)=1$ können z.B. kantenverdächtige Punkte sein, die man mit Operatoren der Bildverarbeitung bereits herausgefiltert hat.

Allgemeines Vorgehen

$f(i,j,p)$ beliebige Funktion mit Werten aus R

$A \subseteq R^n = \{(p_1,\ldots,p_n)\}$ endliche Teilmenge (z.B. ein Gitter)

$\forall p \in A$

$$d(p) = \sum_{(i,j)\ mit\ g_E(i,j)=1} H(i,j,p)$$

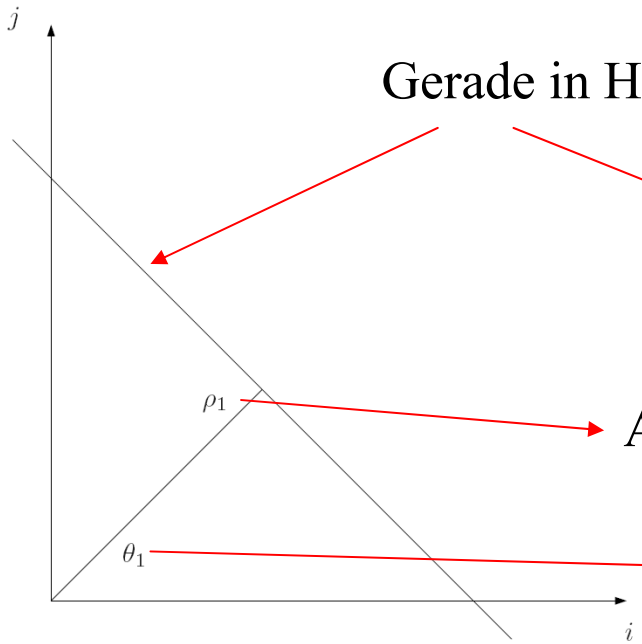
$$H(i,j,p) = \begin{cases} 1 & \text{falls } f(i,j,p) = 0 \\ 0 & \text{sonst} \end{cases}$$

$d(p) > T_1$ ⇨ Objekt mit der Parameterkombination p gefunden

T_1: vorgegebener Schwellwert

Geraden

$n = 2$ $\quad p = (\rho, \theta) \in R^2$



Gerade in Hessescher Normalform

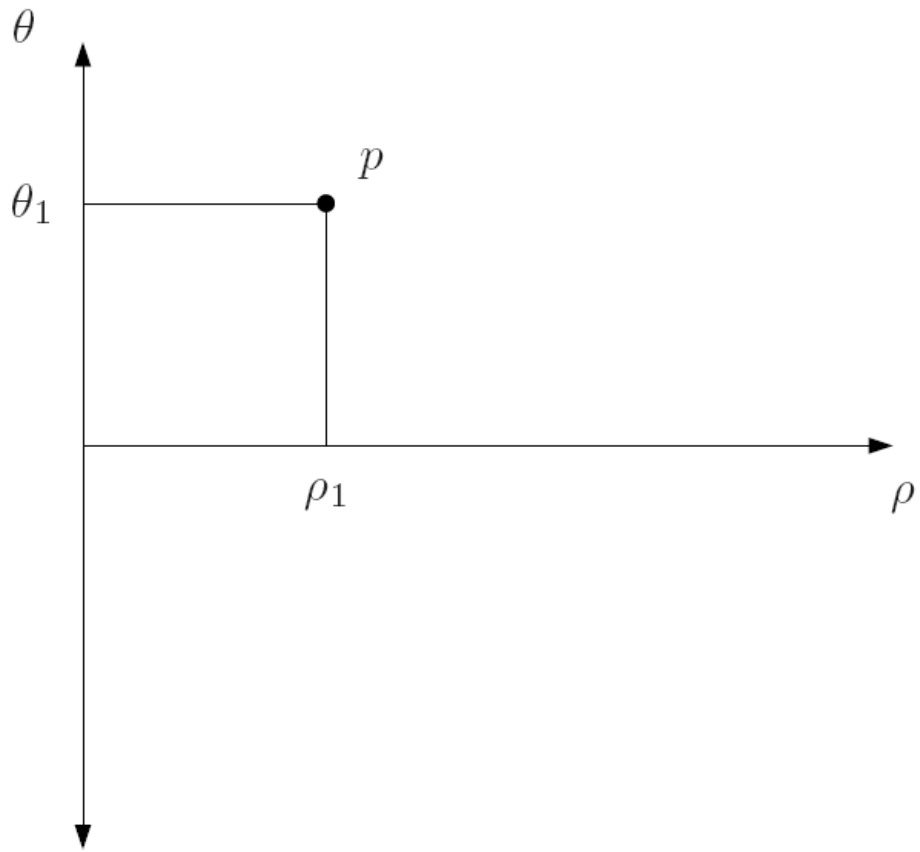
$$\rho = i\cos(\theta) + j\sin(\theta)$$

Abstand vom Nullpunkt zur Geraden

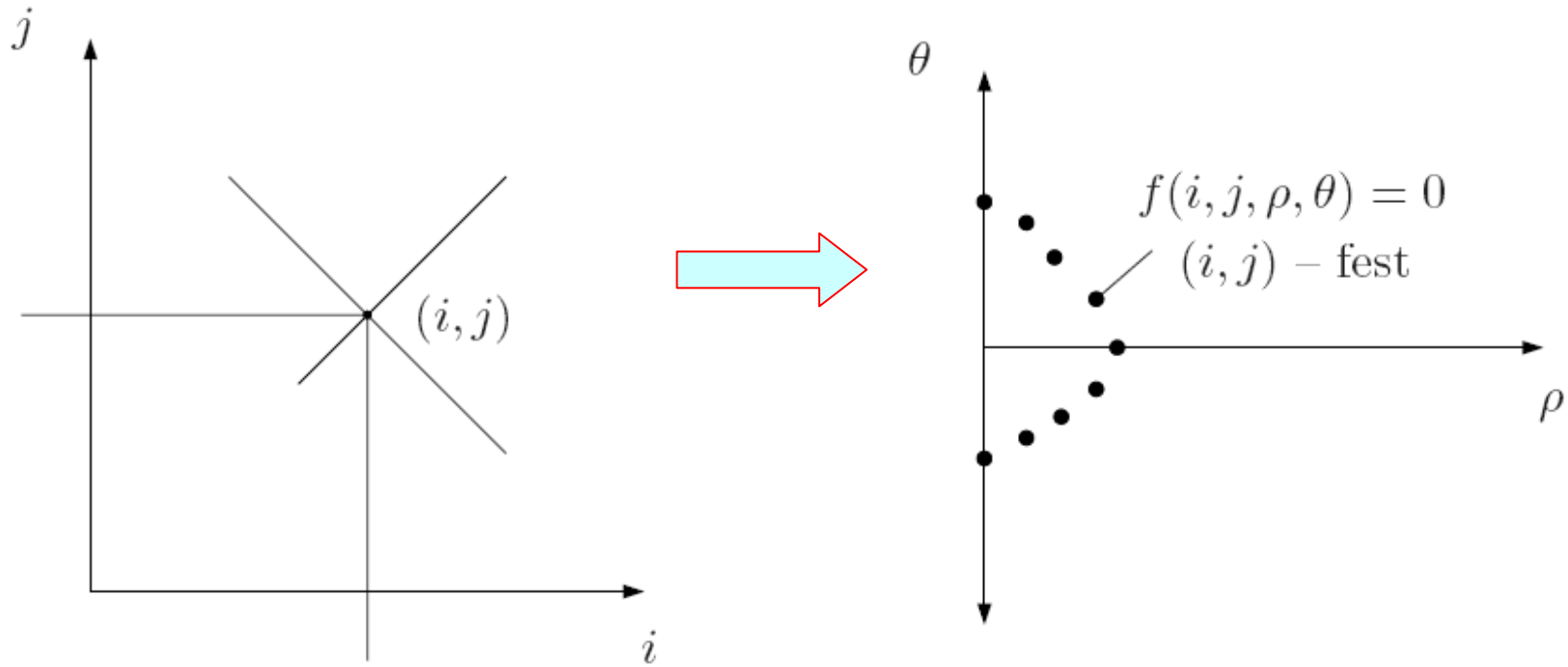
Winkel zwischen der i – Achse und dem Lot vom Nullpunkt auf die Gerade

$$f(i, j, \rho, \theta) = i\cos(\theta) + j\sin(\theta) - \rho$$

Geraden – Parameterraum

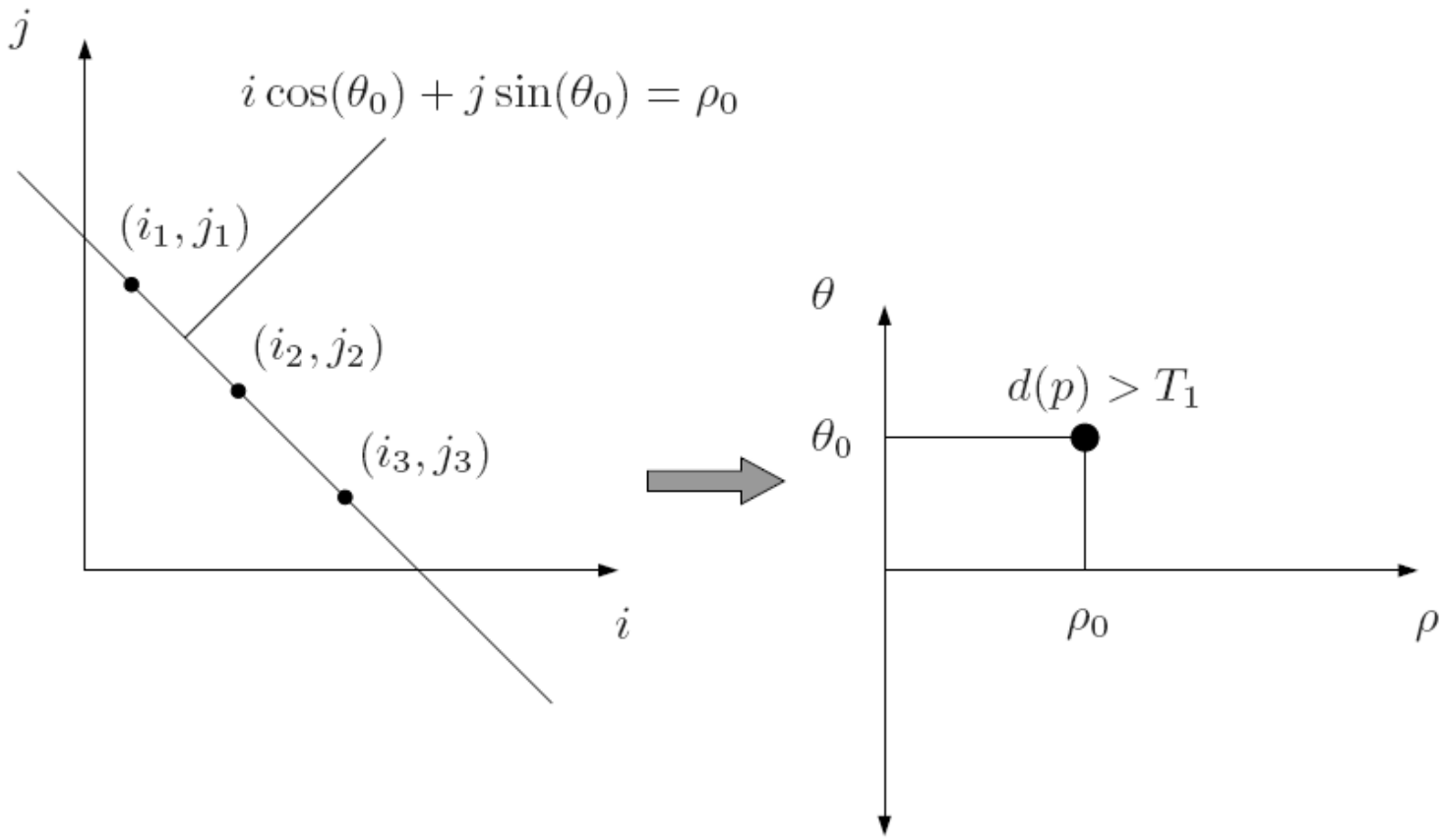


Geraden – Parameterraum



Eine Schar von Geraden durch einen Punkt (*i,j*) erscheint im Parameterraum als eine Punktmenge.

Geraden – Parameterraum



Eine Gerade durch 3 Punkte erscheint nur an einer Stelle *p* im Parameterraum. Dafür ist der Wert von *d*(*p*) aber schon 3.

Geraden – Algorithmus

Parameter ρ und θ werden diskretisiert, z.B. in Form eines Gitters

$$\rho = \rho_1, \ldots, \rho_k \qquad \theta = \theta_1, \ldots, \theta_l$$

$$d(p) = \sum_{(i,j)\ mit\ g_E(i,j)=1} H(i,j,p)$$

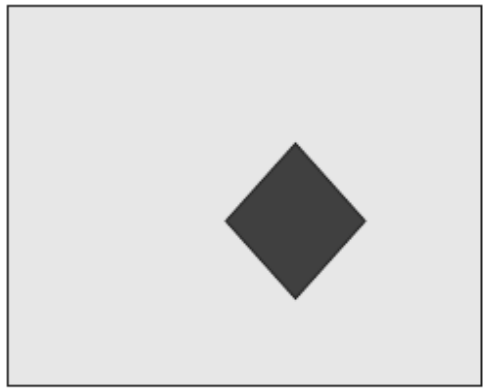
$$\forall p = (\rho, \theta),\ \rho = \rho_1, \ldots, \rho_k,\ \theta = \theta_1, \ldots, \theta_l$$

$$H(i,j,p) = \begin{cases} 1 & \text{falls} \quad f(i,j,\rho,\theta) = i\cos(\theta) + j\sin(\theta) - \rho = 0 \\ 0 & \text{sonst} \end{cases}$$

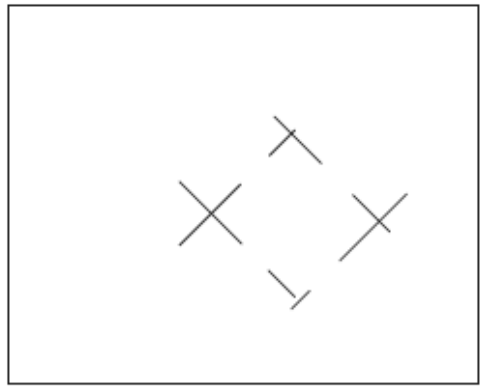
$$d(p) > T_1 \quad \Rightarrow$$

Gerade mit der Parameterkombination p gefunden

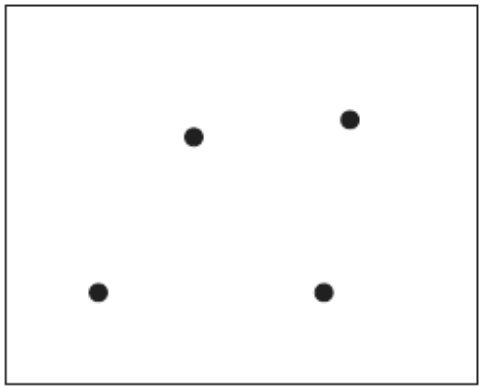
Beispiel



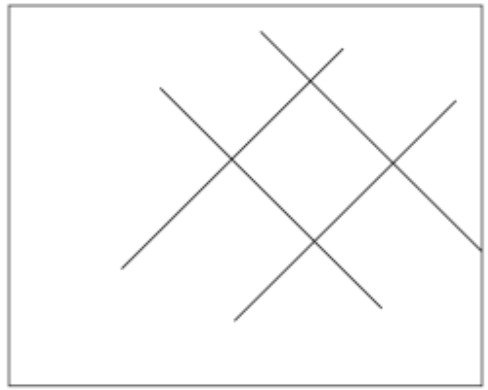
Eingabebild



Kantendetektion

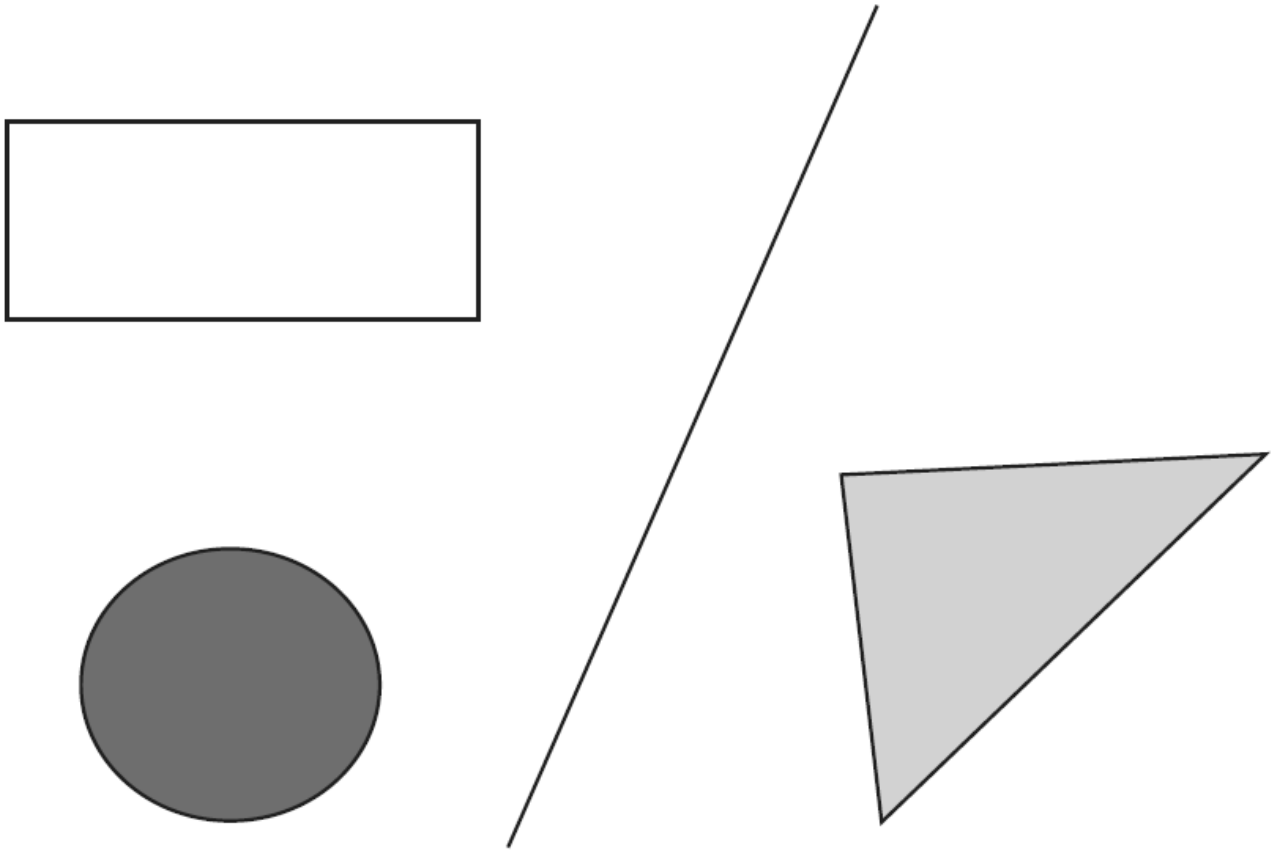


Parameterraum
(4 markante Punkte)



gefundene Geraden

Beispiel

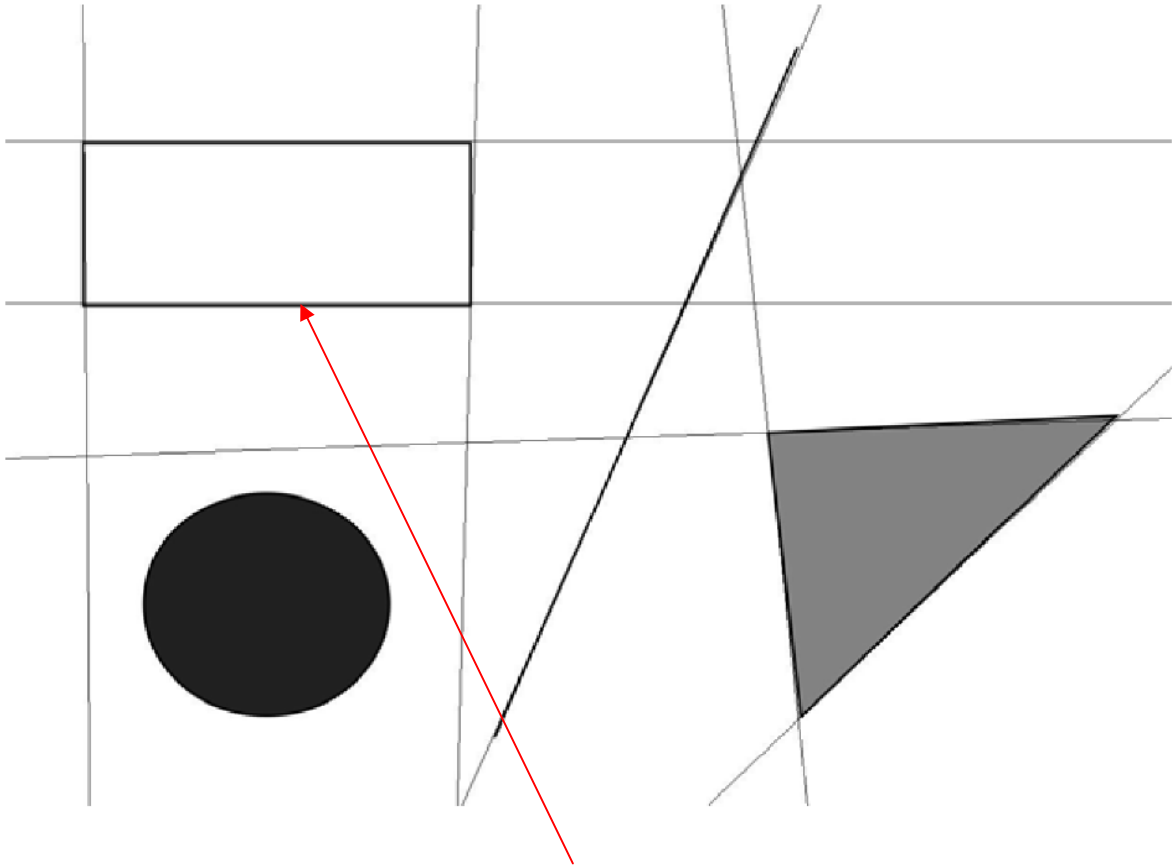


Beispiel

Die Werte *d(p)* können im Parameterraum bildlich dargestellt werden.
d(p) kann als Grauwert interpretiert werden.

Im Parameterraum finden wir 8 markante Einträge
für die 8 Geraden im Eingabebild.

Beispiel



Um die geradlinig begrenzten Segmente zu finden, ist aber noch eine Nachbearbeitung nötig, da man nicht die gesamten Geraden benötigt, sondern nur Teilstrecken dieser Geraden.

Kreise

R_0 - gegeben (x, y) - gesucht (Mittelpunkt)

$n = 2$ $p = (x, y)$

$$f(i, j, x, y) = (i - x)^2 + (j - y)^2 - R_0^2$$

R – kann auch noch ein dritter Parameter sein

Bei mehr als 2 Parametern kann allerdings der Rechenaufwand schon sehr groß werden.