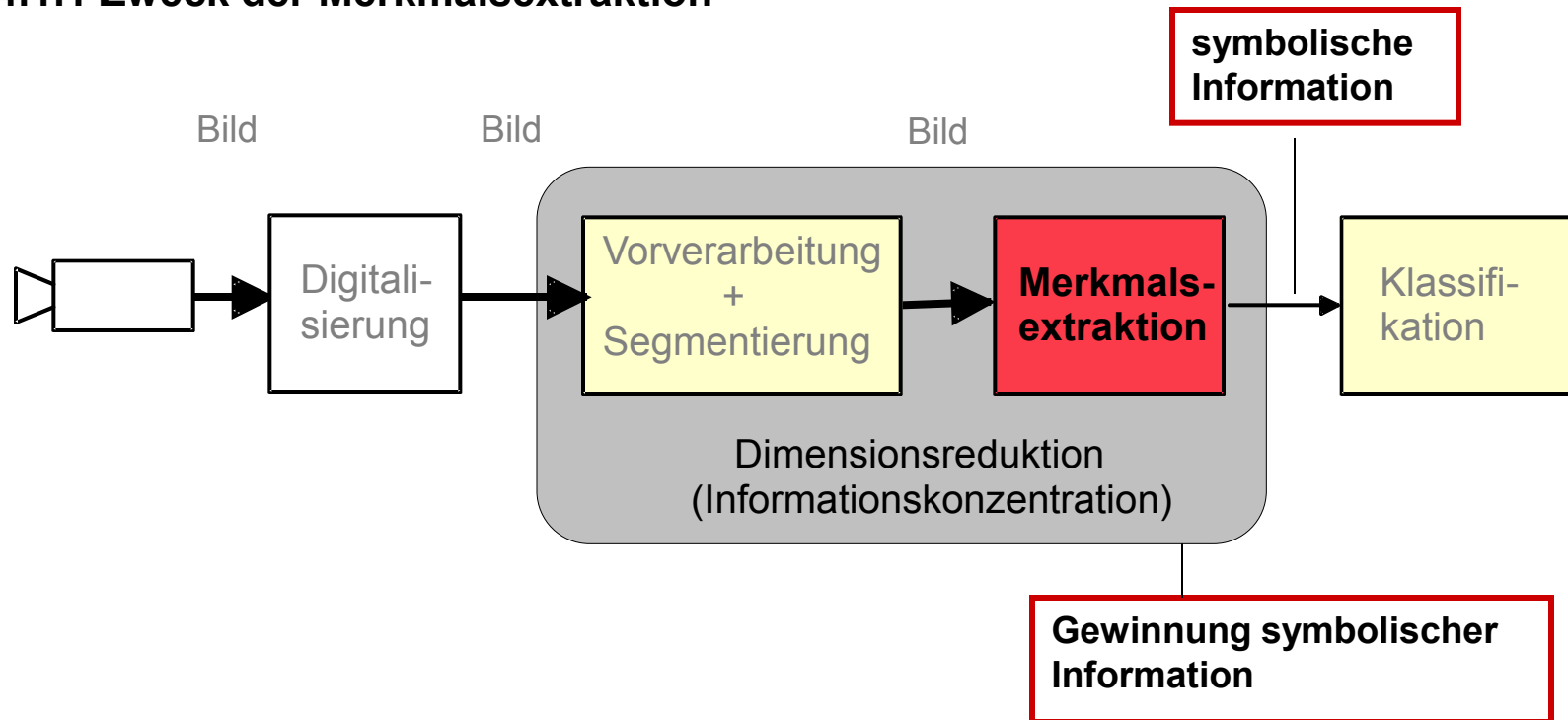




4. Merkmalsextraktion

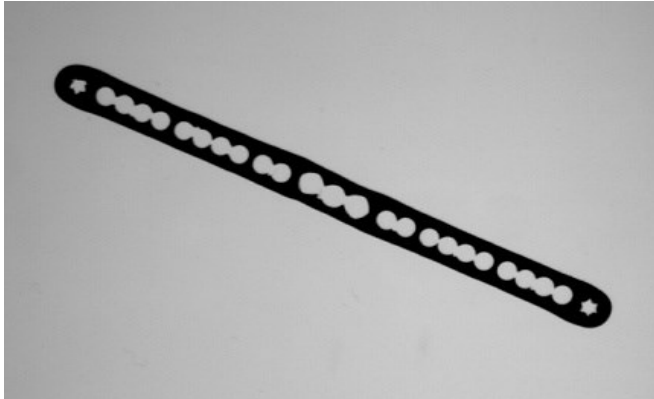
4.1 Einführung

4.1.1 Zweck der Merkmalsextraktion



Ziel: Extraktion der für die Klassifikation bedeutsamen Merkmale eines Musters.

4.1.2 Behandelte Aufgabenstellungen

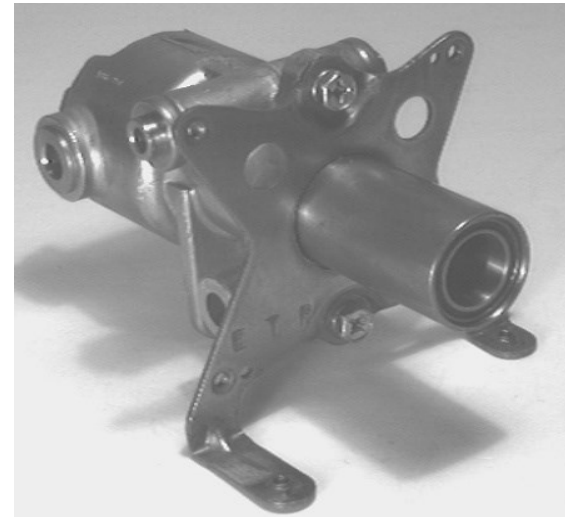


Kennzeichen:

Zusammenhängende Bildflächen
(= Regionen)

Die zusammenhängenden Bildflächen
(*Silhouette, Aussen-/Innenkontur*)
beinhalten die relevante Information.

→ **regionenbasierte** Verfahren



Kennzeichen:

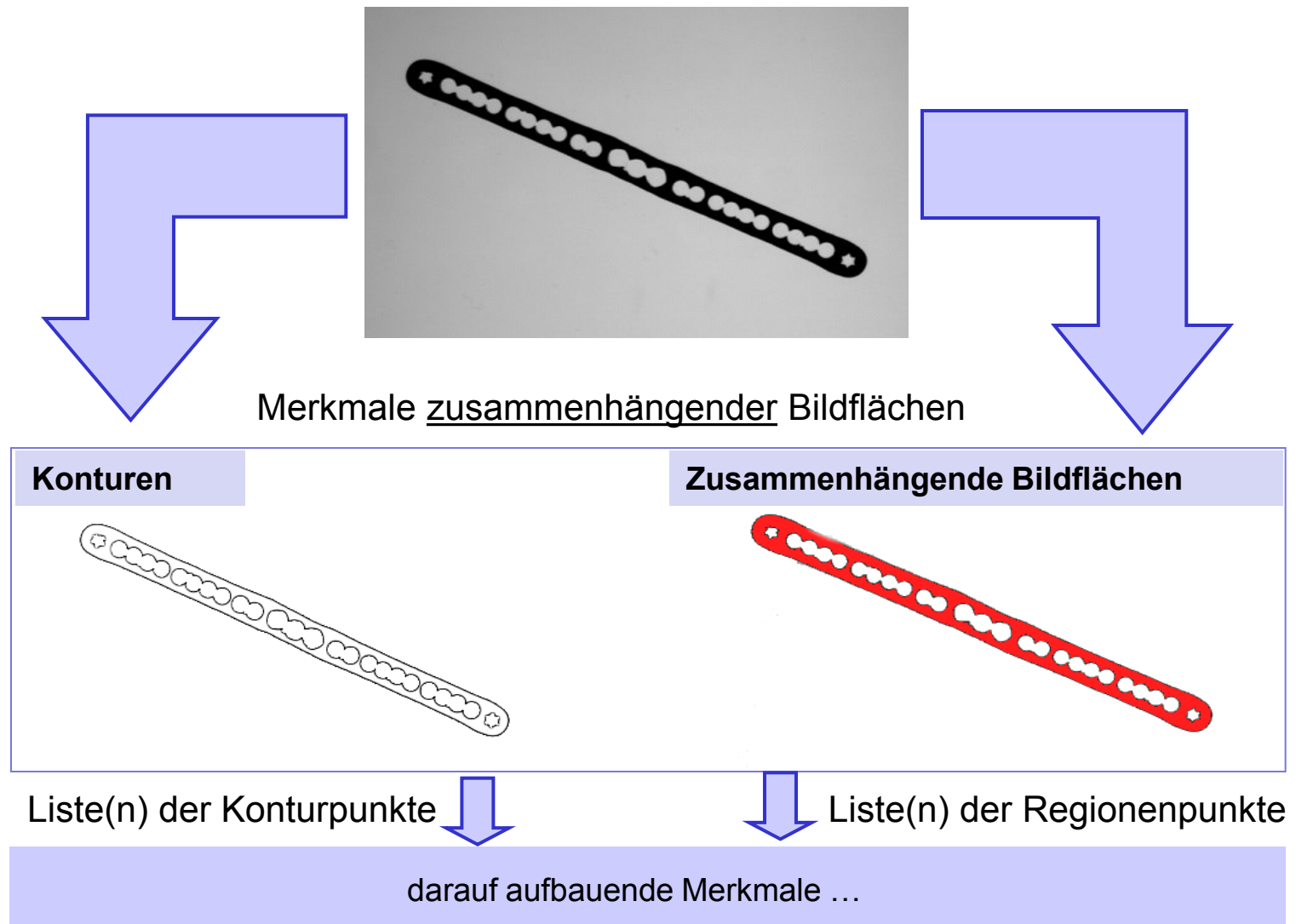
Zusammenhängende Bildflächen
sind nicht ohne weiteres extrahierbar.

Nicht die zusammenhängenden
Bildflächen stellen nicht die relevante
Information dar, sondern die Kanten.

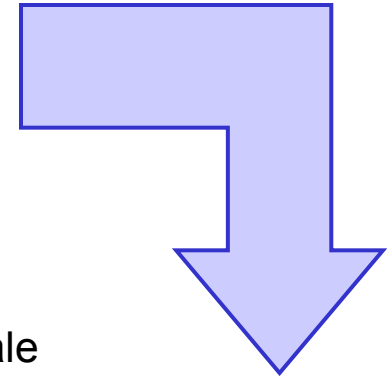
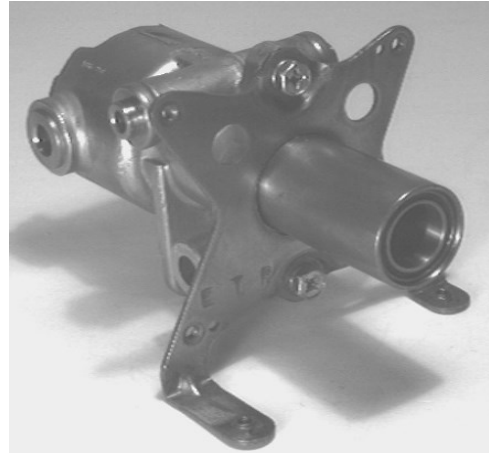
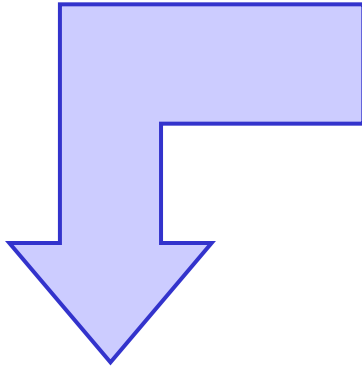
→ **kantenbasierte** Verfahren



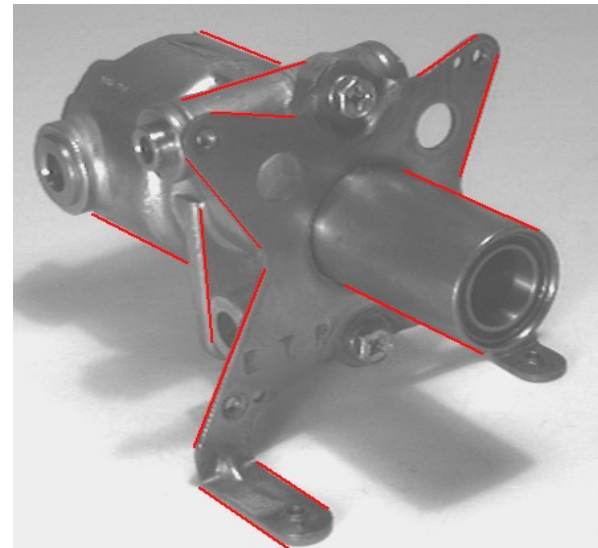
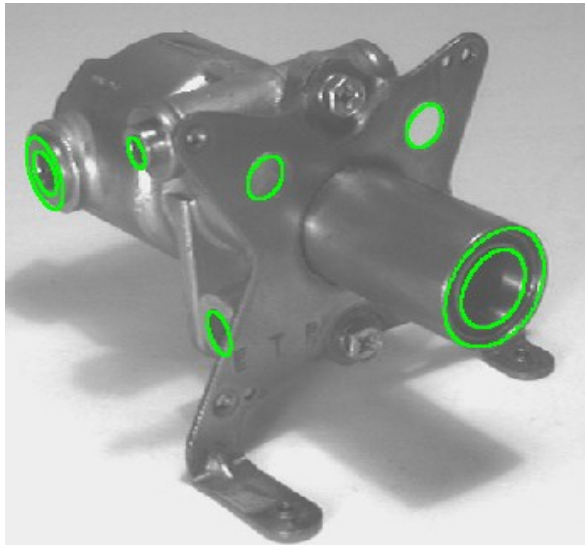
4.1.3 Regionenbasierte Verfahren



4.1.4 Kantenbasierte Verfahren



Beispiele kantenbasierter Merkmale





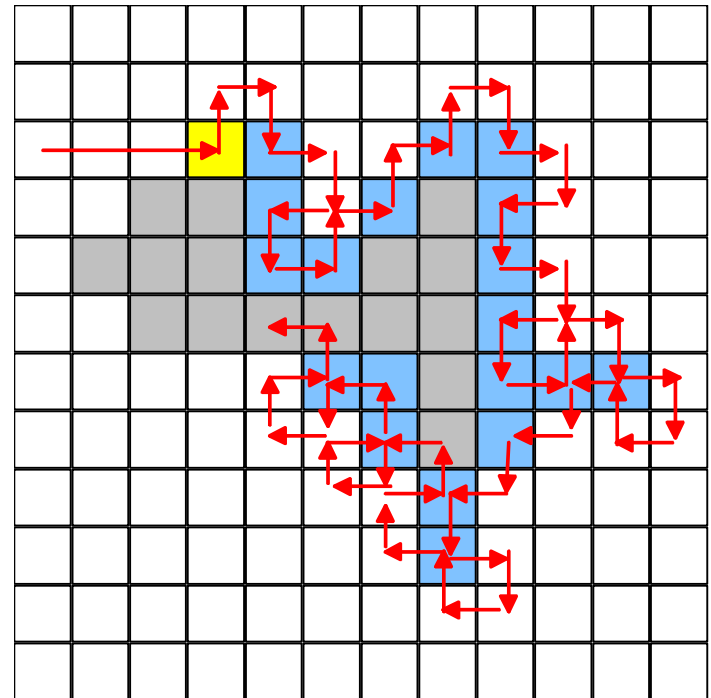
4.2 Regionenbasierte Verfahren anhand von Konturen

4.2.1 Konturextraktion aus Binärbildern

Ziel der Konturextraktion ist es, eine Liste von Objektpunkten zu erhalten, die auf der Randlinie des Objektes liegen.

Algorithmus 1:

1. Suche zeilenweise nach dem ersten Objektpunkt.
Dies ist der Startpunkt .
2. **do**
3. **if**(Objektpunkt gefunden)
4. biege nach links ab 
5. **else**
6. biege nach rechts ab 
7. **endif**
8. **while** (aktueller Punkt NOT Startpunkt)

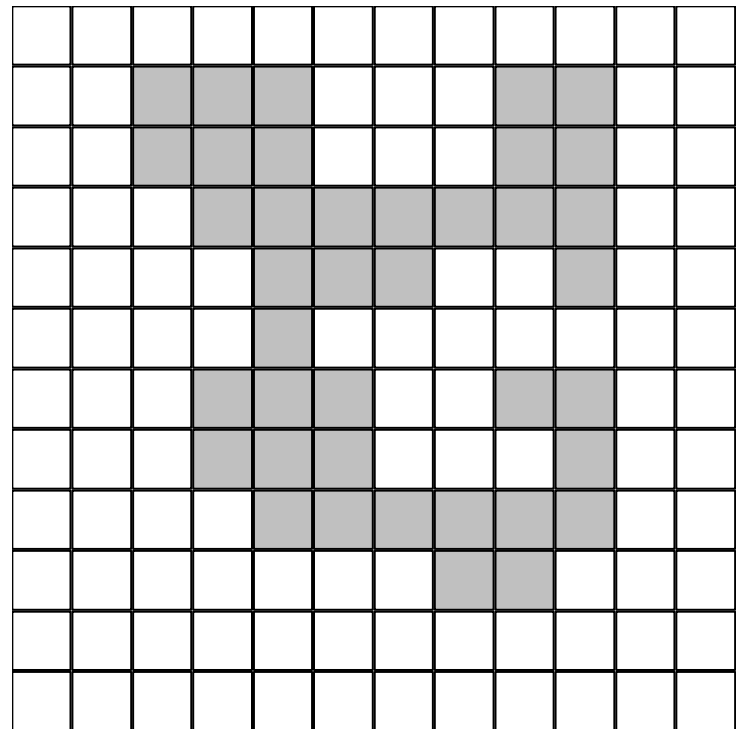
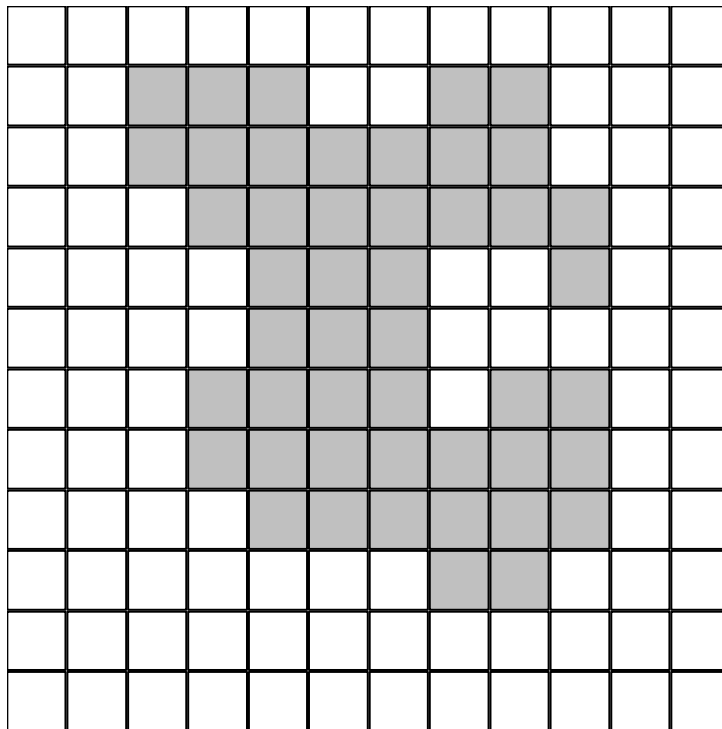




ÜBUNG: Konturextraktion 1

Zeigen Sie, wie Konturalgorithmus 1 die Kontur findet. Geben Sie die Liste der extrahierte Punkte an (Anm.: Bildursprung oben links).

Wo könnten Probleme entstehen?





Algorithmus 2 (*Pavlidis*):

1. Suche zeilenweise nach dem ersten Objektpunkt. Dies ist der Startpunkt A .
2. Setze: aktuellen Punkt $C = A$, Suchrichtung $S = 6$, Flag $first = true$.
3. **while** ($C \neq A$ OR $first == true$) // ein Umlauf um das Objekt
4. Setze: Flag $found = false$.
5. **while** ($found == false$) // suche in verschiedene Richtungen nach Objektpunkt
6. **if** (in $Richtung(S-1)$ ist ein Objektpunkt (B_{S-1}))
7. Setze: $C = B_{S-1}$, $S = Richtung(S-2)$, $found = true$.
8. **else if** (in $Richtung(S)$ ist ein Objektpunkt (B_S))
9. Setze: $C = B_S$, $found = true$.
10. **else if** (in $Richtung(S+1)$ ist ein Objektpunkt (B_{S+1}))
11. Setze: $C = B_{S+1}$, $found = true$.
12. **else**
13. Setze: $S = Richtung(S+2)$
14. **endif**
15. **end while**
16. Setze: $first = false$.
17. **end while**

3	2	1
4	C	0
5	6	7

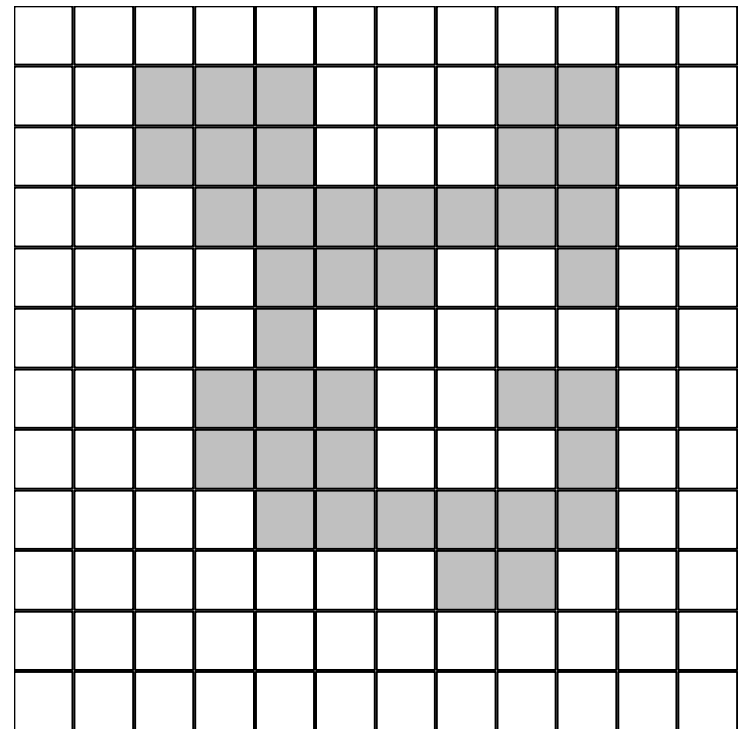
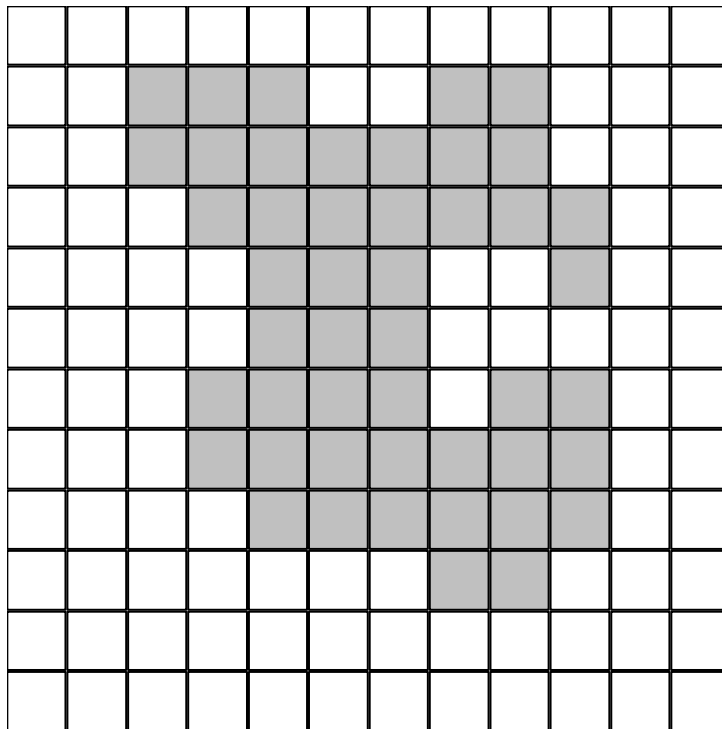
Suchrichtung S

Anm.: $Richtung(S)$ { **return** $(S+8) \bmod 8$ }

ÜBUNG: Konturextraktion 2

Zeigen Sie, wie Konturalgorithmus 2 die Kontur findet. Geben Sie die Liste der extrahierten Punkte an (Anm.: Bildursprung oben links).

Wo könnten Probleme entstehen?



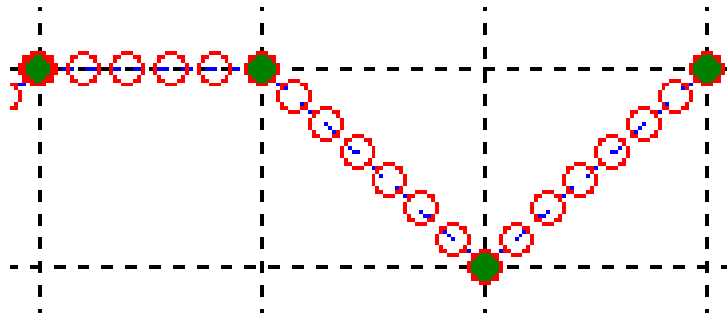


4.2.2 Maßnahmen zur Konturverbesserung

4.2.2.1 Resampling = äquidistante Unterabtastung

Für die nachfolgende Verarbeitung kann es von Nachteil sein, dass die Konturpunkte nicht äquidistant sind :

Abstand horizontaler/vertikaler Nachbarn : 1
Abstand diagonalen Nachbarn : $\sqrt{2}$



$$\sqrt{2} : 1 \approx 1.4 : 1 = 7 : 5$$

Einen (fast) äquidistanten Konturpunktabstand erhält man Unterteilung der

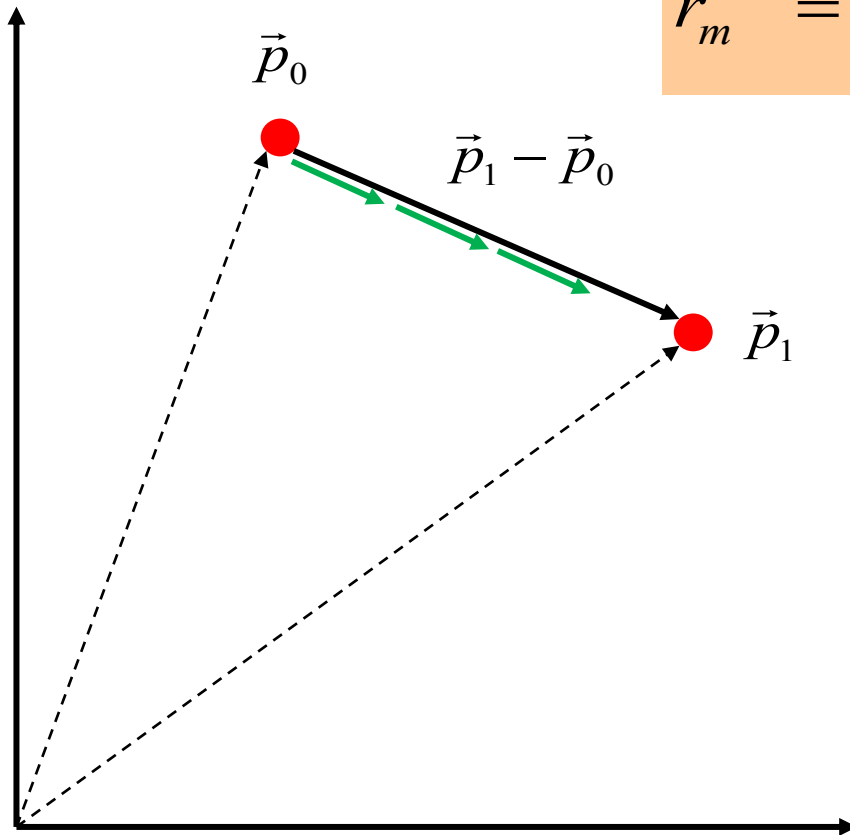
- horizontalen/vertikalen Konturschritte in 5 Unterschritte und der
- diagonalen Konturschritte in 7 Unterschritte .



Die T unterabgetasteten Punkte zwischen 2 Punkten erhält man mit:

$$\vec{r}_m = \vec{p}_0 + m \cdot \frac{(\vec{p}_1 - \vec{p}_0)}{T}$$

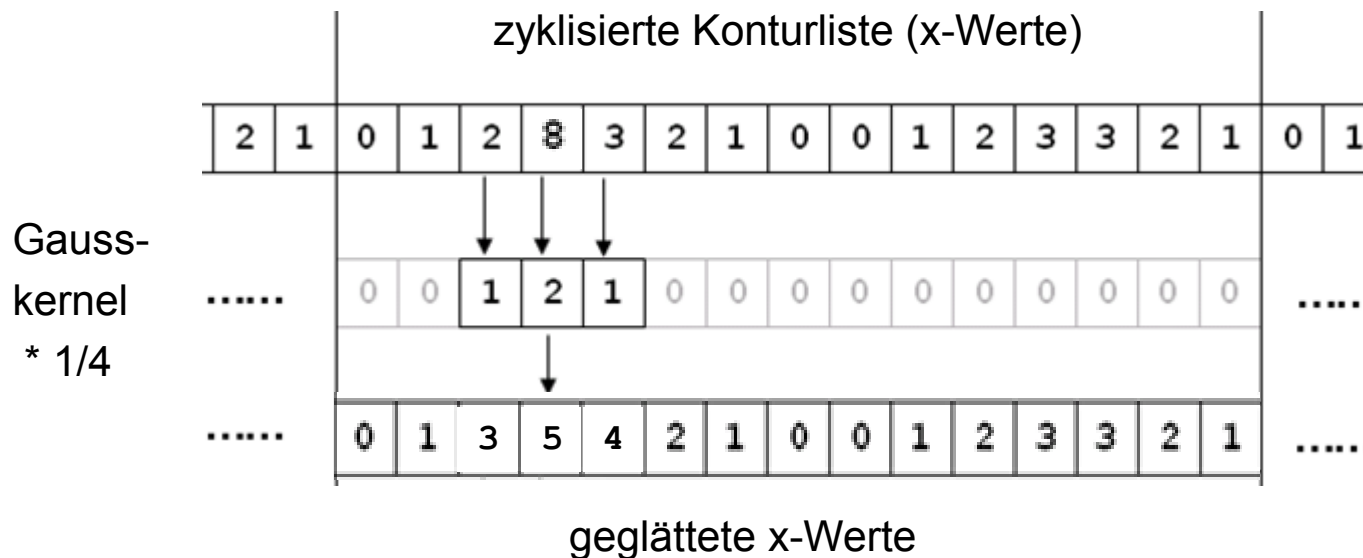
$$m = [0, T - 1)$$

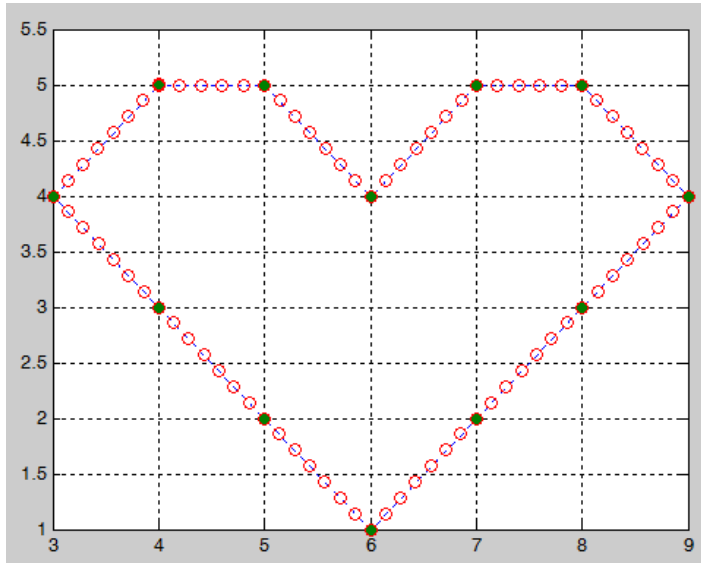




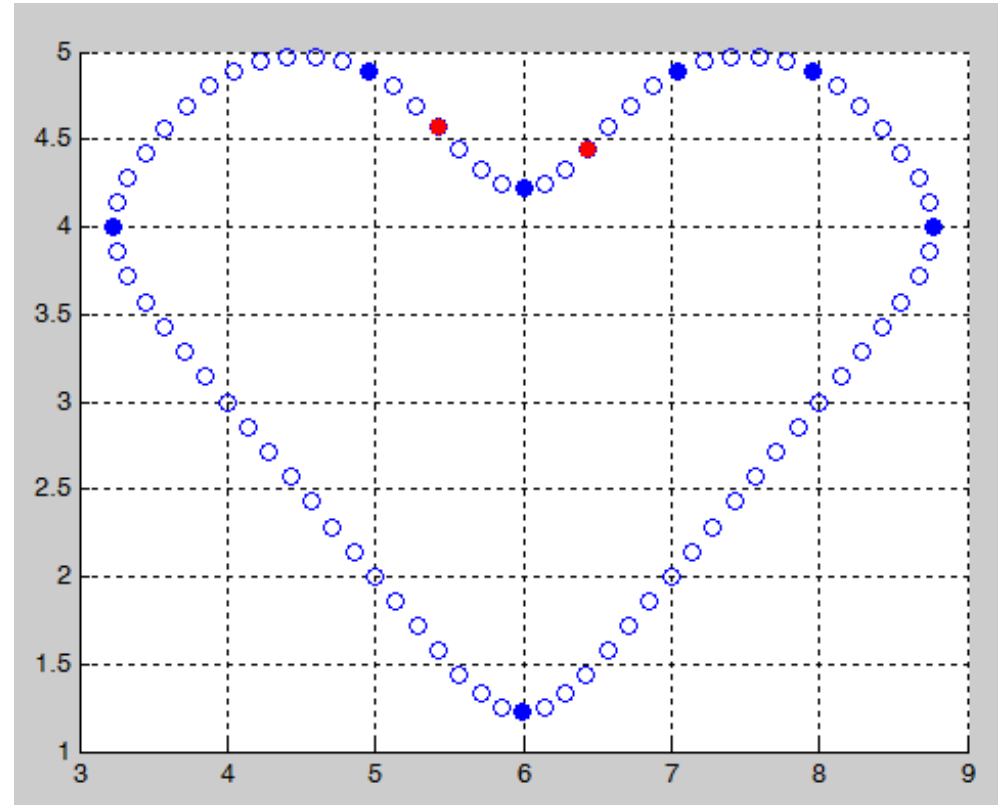
4.2.2.2 Konturglättung

Zur Glättung der Kontur können die (zyklisierten) x- und y-Konturwerte mit einem (1-dimensionalen) Gauss-Kernel gefaltet werden.





äquidistant neu abgetastete
Konturliste



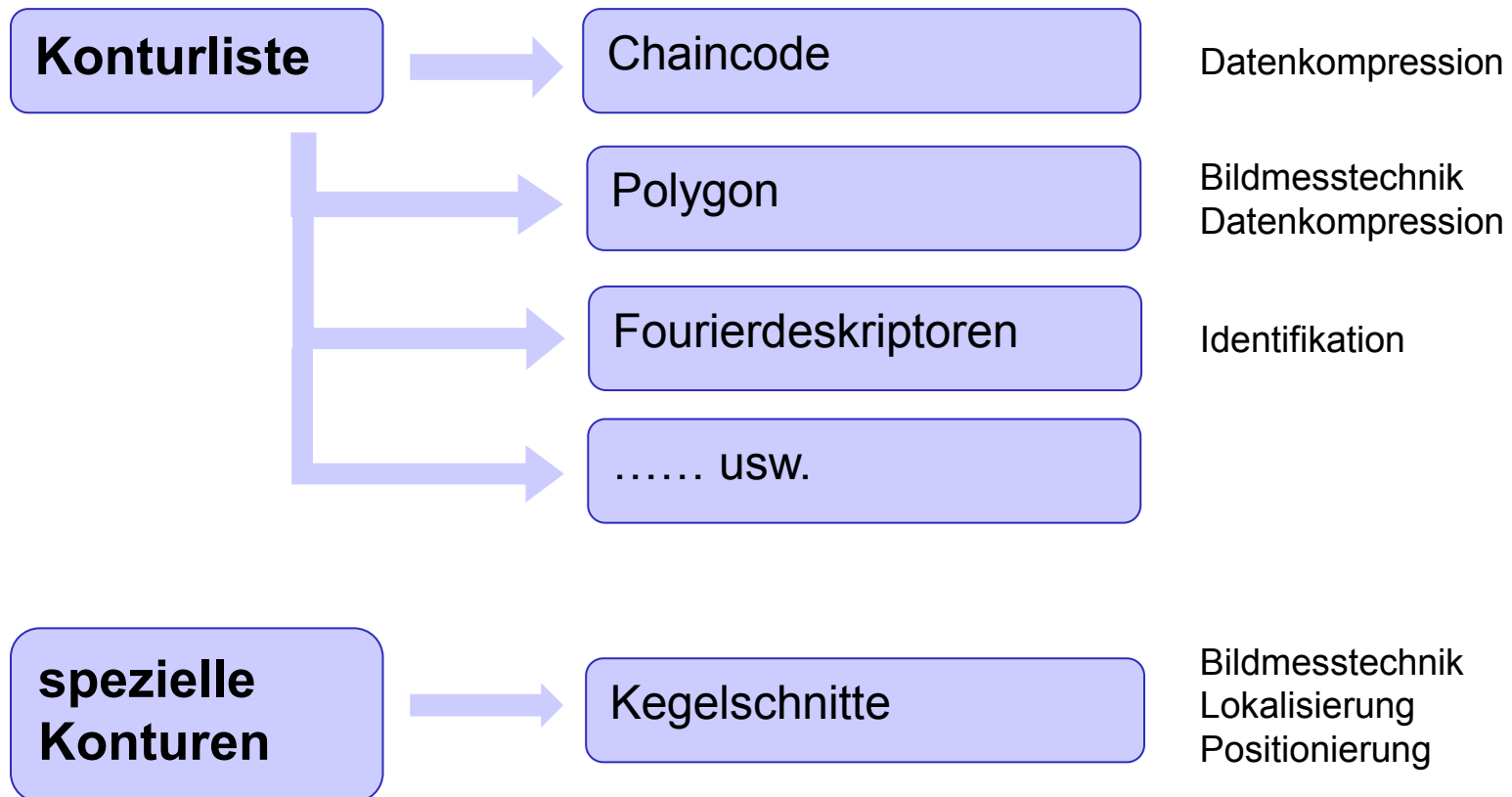
geglättete Konturliste
(13-Element-Gausskernel)



4.2.3 Konturbeschreibung

4.2.3.1 Einführende Gedanken

Für die weitere Verarbeitung (Merkmalsextraktion) werden die Konturpunkte i. Allg. in andere Darstellungsweisen umgewandelt.





4.2.3.2 Chaincode

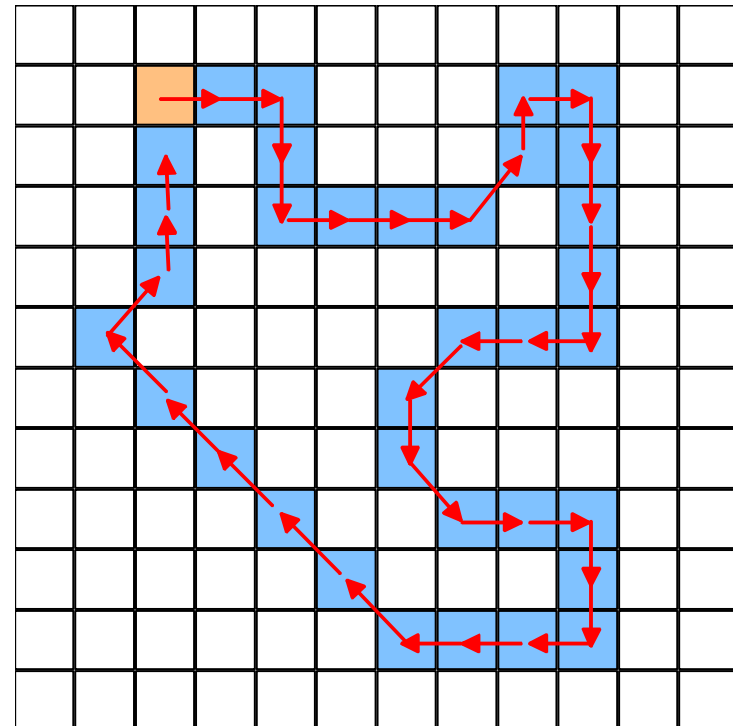
Beginnend bei einem Startpunkt S, wird lediglich die Richtung des jeweils nächsten Punktes angegeben (8-er Nachbarschaft). Pro Punkt sind somit nur 3 bit (Werte: 0...7) notwendig.

3	2	1
4	C	0
5	6	7

Suchrichtung S

Beispiel: Chaincode

00660001206664456700....





4.2.3.3 Polygonbeschreibung

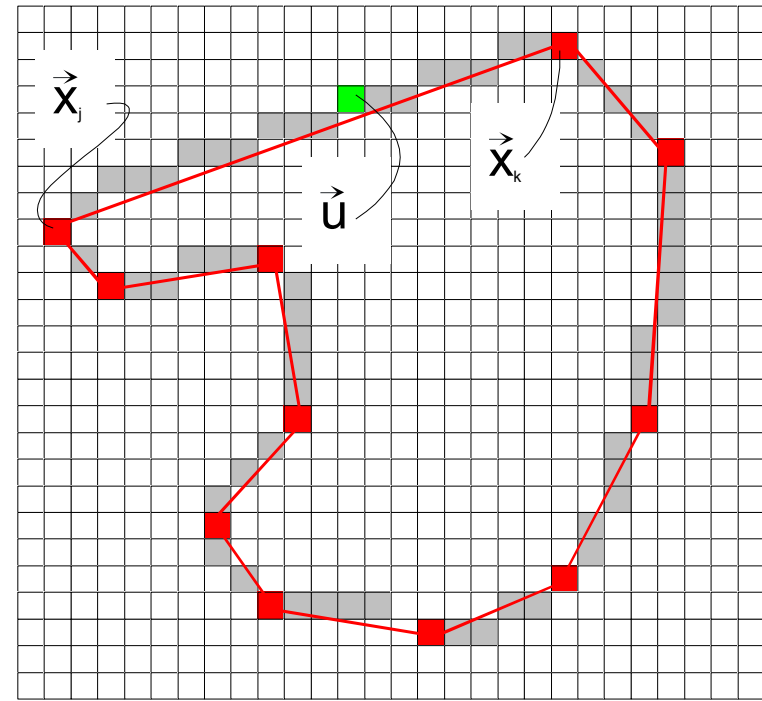
Für viele Anwendungsfälle (z.B. Konturidentifikation) ist es wünschenswert, die Konturpunktmenge stückweise linear so zu approximieren, dass ein vorgegebener Maximalfehler nicht überschritten wird (→ [Split & Merge - Algorithmus](#)).

Für die Gleichung einer Geraden zwischen zwei Punkten (x_j, y_j) und (x_k, y_k) gilt:

$$x(y_j - y_k) + y(x_k - x_j) + y_k \cdot x_j - y_j \cdot x_k = 0 \quad (1)$$

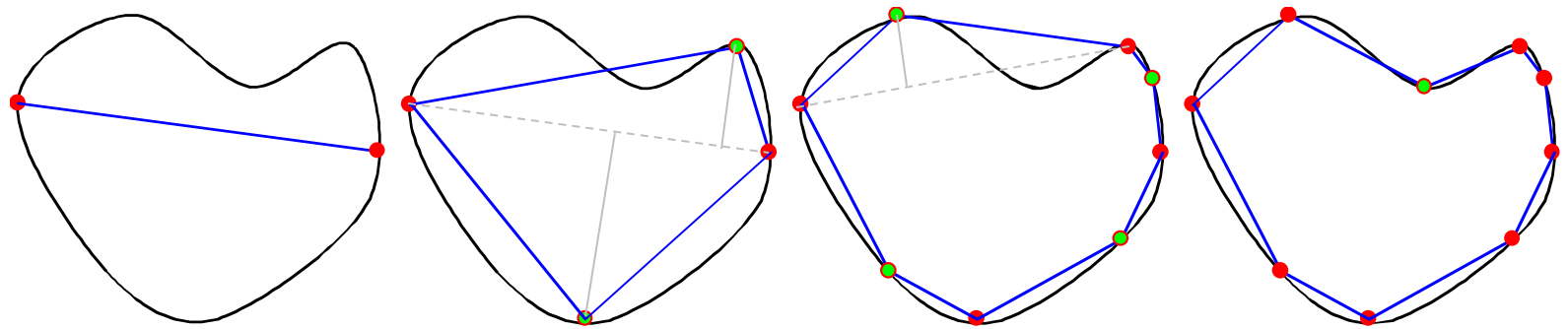
Für den Abstand eines Punktes (u, v) zu dieser Geraden gilt:

$$d = \frac{u(y_j - y_k) + v(x_k - x_j) + y_k \cdot x_j - y_j \cdot x_k}{\sqrt{(y_j - y_k)^2 + (x_k - x_j)^2}} \quad (2)$$



Polygonapproximation durch Zerlegung (Split)

1. Bestimme: Anfangspunkt (x_a, y_a) und Endpunkt (x_e, y_e) der ersten Geraden.
2. WHILE (Fehler mindestens einer Gerade größer F_{\max})
3. FOR EACH (Gerade mit zu großem Fehler)
4. Ermittle den Punkt P mit dem größten Abstand zur Geraden
5. Ersetze die alte Gerade durch zwei neue Geraden durch P
6. END
7. END WHILE





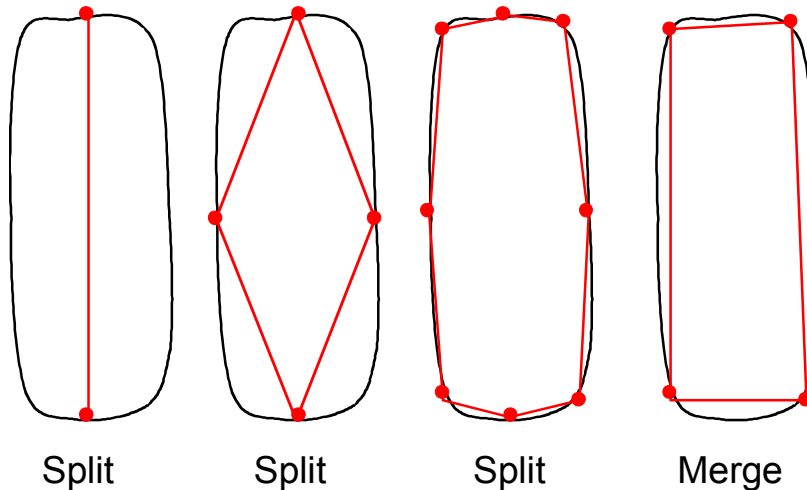
Verbesserte Polygonapproximation (Split & Merge)

Bei komplexeren Konturen führt die Polygonapproximation durch das Splittingverfahren häufig

- zu einer zu großen Anzahl von Segmenten und
- die Zerlegung ist stark abhängig vom (zufälligen) Startpunkt.

Eine deutliche Verbesserung bringt der folgende Algorithmus:

1. Führe eine Anfangszerlegung durch Splitting durch.
2. Vereinige alle Geradenpaare, deren Vereinigung unter Beibehaltung des max. erlaubten Fehlers möglich ist.





ÜBUNG: Math. Beschreibung von Geraden in der Ebene 1

1. Diskutieren / interpretieren Sie verschiedene Varianten der Geradenbeschreibung:

a) Steigungs-Achsenabschnittsform $y = mx + b$

b) Achsenabschnittsform

$$\frac{x}{a} + \frac{y}{b} = 1$$

$$Ax + By = 1$$

c) Zwei-Punkte-Form

$$x(y_j - y_k) + y(x_k - x_j) + y_k \cdot x_j - y_j \cdot x_k = 0$$

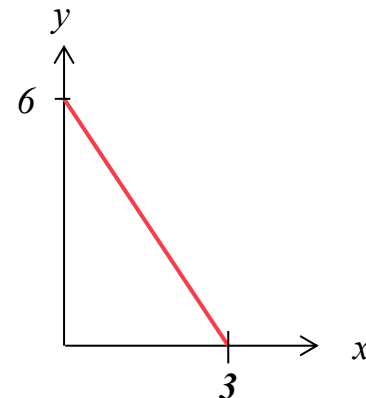
d) Hesse'sche Normalform

$$r = x \cdot \cos \Theta + y \cdot \sin \Theta$$

2. Wie lautet die Geradengleichung der nebenstehenden Gerade

a) in der Achsenabschnittsform

b) in der Steigungs-Achsenabschnittsform?





ÜBUNG: Math. Beschreibung von Geraden in der Ebene 2

s. Tafel: kurze Wiederholung zu Vektoren

3. Leiten Sie die "*2-Punkte-Form der Geradengleichung*" her, also :

$$x(y_j - y_k) + y(x_k - x_j) + y_k \cdot x_j - y_j \cdot x_k = 0$$

4. Eine Gerade geht durch die Punkte $(x_1, y_1) = (1, 3)$ und $(x_2, y_2) = (6, 8)$.

a) Geben Sie die "*2-Punkte-Form der Geradengleichung*" an.

b) Beschreiben Sie die Gerade in der Form $y = mx + b$.

c) Wie weit ist der Punkt $(u, v) = (2, 6)$ von der Gerade entfernt?



ÜBUNG: Math. Beschreibung von Geraden in der Ebene 3

5 . Leiten Sie die "*Hesse'sche Normalform der Geradengleichung*" her, also :

$$r = x \cdot \cos \theta + y \cdot \sin \theta$$

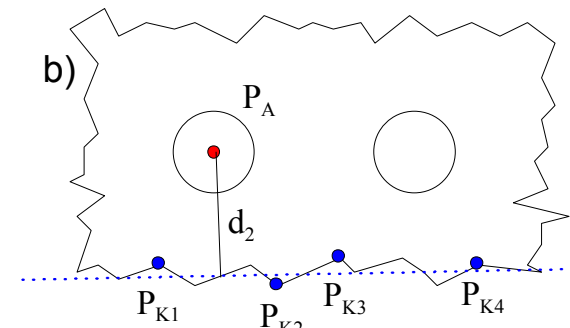
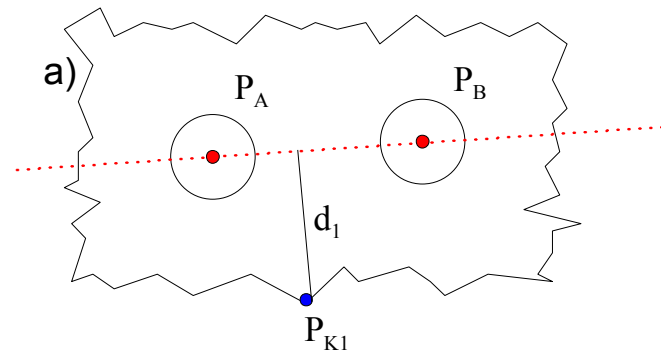
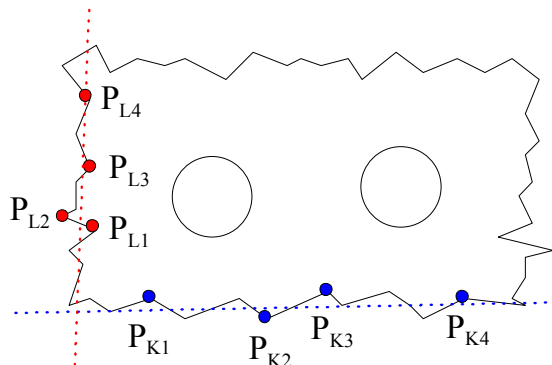
6. Gegeben ist eine Gerade $y = -2x + 4$.

- a) Beschreiben Sie die Gerade in der Form $Ax + By = I$.
- b) Beschreiben Sie die Gerade in der Hesse'schen Normalform.

ÜBUNG: Anwendung in der „Bildmesstechnik“

Gegeben ist das abgebildete Werkstück (Kantenrauhigkeit übertrieben gezeichnet).

- Zu bestimmen ist der kürzeste Abstand d_1 zwischen der Verbindungsgeraden P_A - P_B (Bohrungszentren) und dem Kantenpunkt P_{K1} .
- Zu bestimmen ist der kürzeste Abstand d_2 zwischen der Ausgleichsgeraden durch die Kantenpunkte und dem Bohrungszentrum P_A .
- Zu bestimmen ist der Winkel zwischen den beiden Ausgleichsgeraden K und L.



Skizzieren und diskutieren Sie die Lösungen.

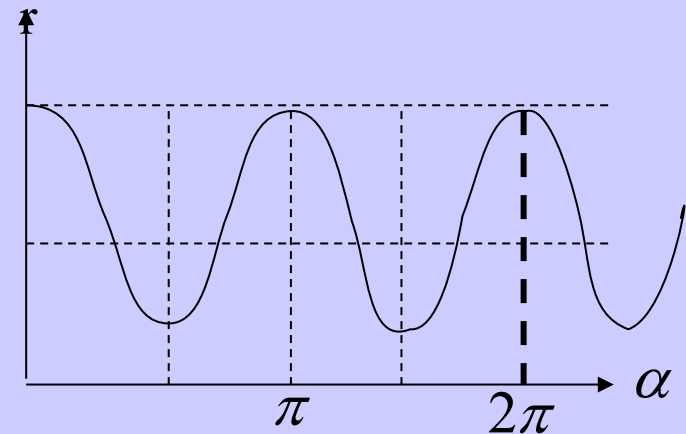
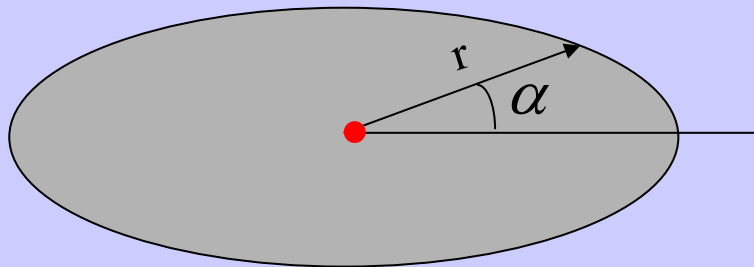
4.2.3.4 Periodische Konturfunktionen

Für die Objektidentifikation ist die Umwandlung der Konturliste in eine periodische Konturfunktion vorteilhaft.

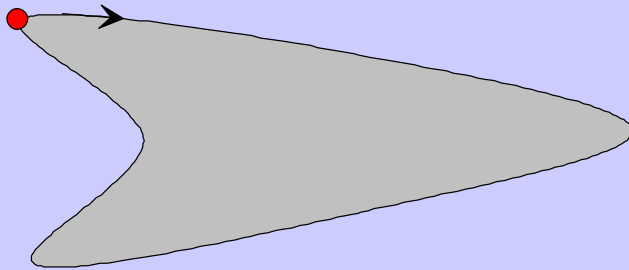
Grundidee:

1. Kontur in eine eindimensionale, periodische Funktion umwandeln.
2. Beschreibung der periodischen Funktion durch die Fourierkoeffizienten.

Beispiel 1: Schwerpunktabstand $r(\alpha)$



Anm.: nur eingeschränkt anwendbar (warum?)

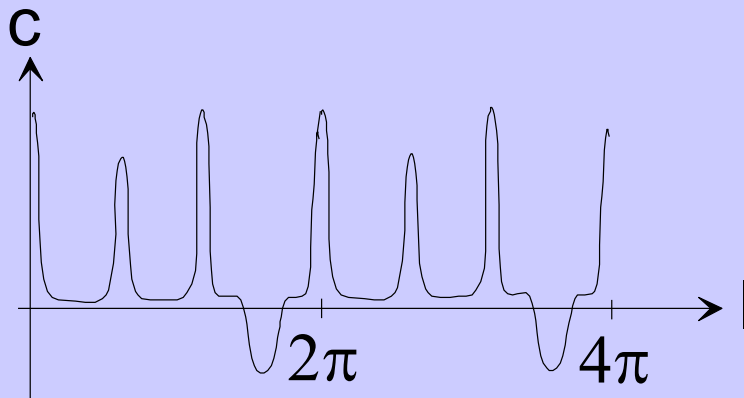
**Beispiel 2:** Krümmungsfunktion κ 

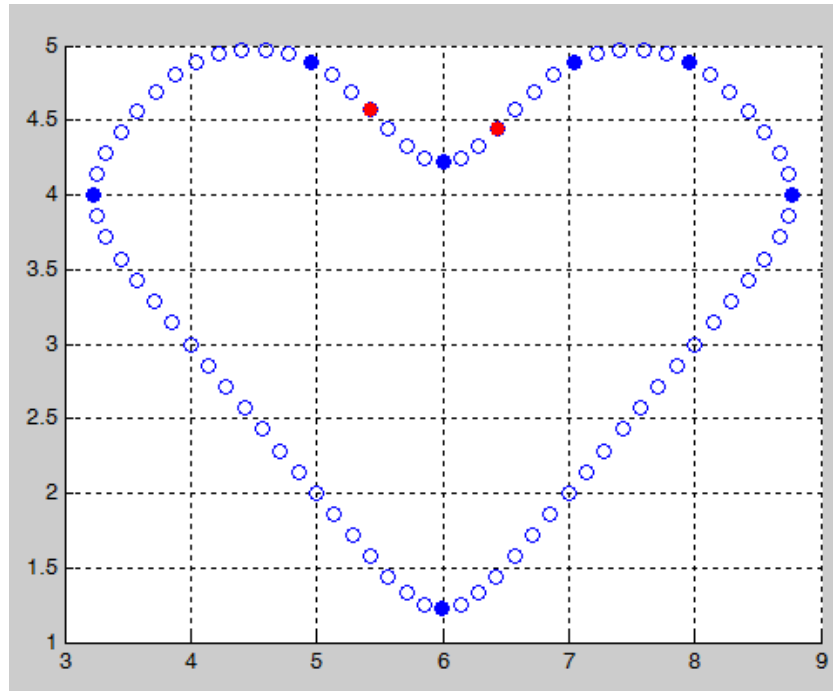
$$\kappa = \frac{x' \cdot y'' - x'' \cdot y'}{(x'^2 + y'^2)^{3/2}}$$

mit den Ableitungen

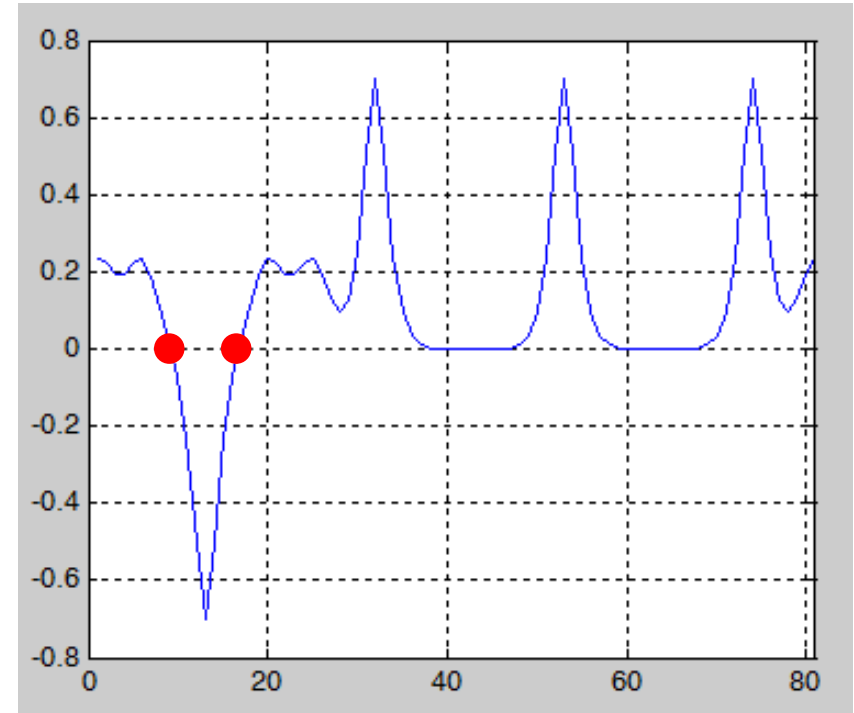
$$x'(n) = \frac{1}{2} \cdot [x(n+1) - x(n-1)]$$

$$x''(n) = x(n+1) - 2x(n) + x(n-1)$$





geglättete Kontur



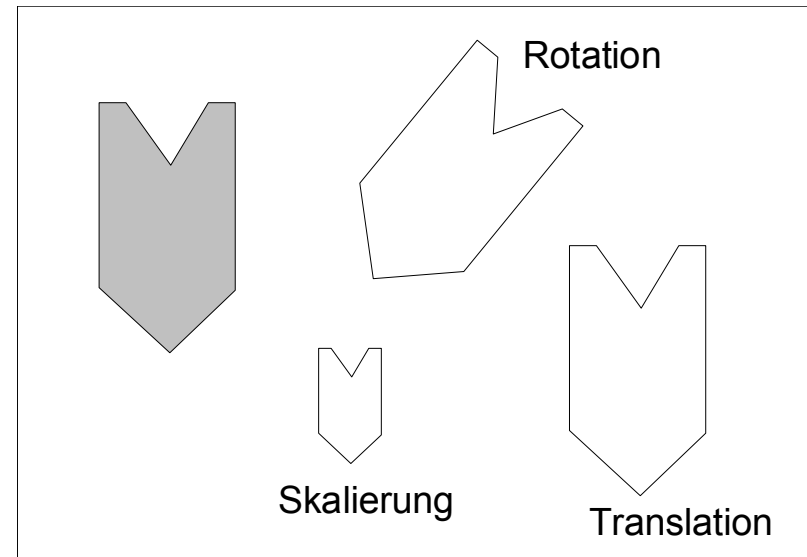
Krümmungsfunktion mit Krümmungswendepunkten (Zerocrossings)



4.2.3.5 Konturidentifikation

Anforderungen :

Ähnliche Konturen müssen
ähnliche Ergebnisse erzeugen !



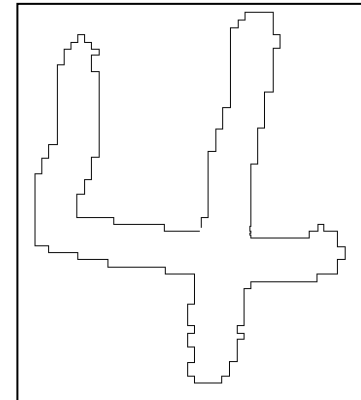
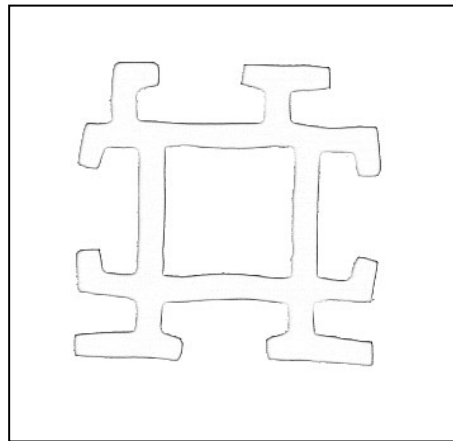
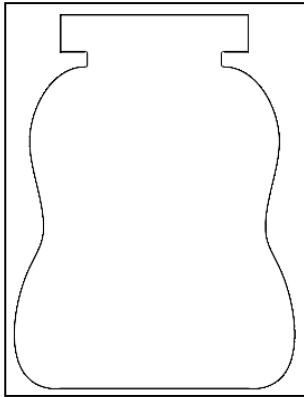
Die Identifikation muss unabhängig sein von/vom

- Bildmaßstab (*Skalierungsinvarianz*)
- Verdrehung im Bild (*Rotationsinvarianz*)
- Verschiebung im Bild (*Translationsinvarianz*)



Anwendungen:

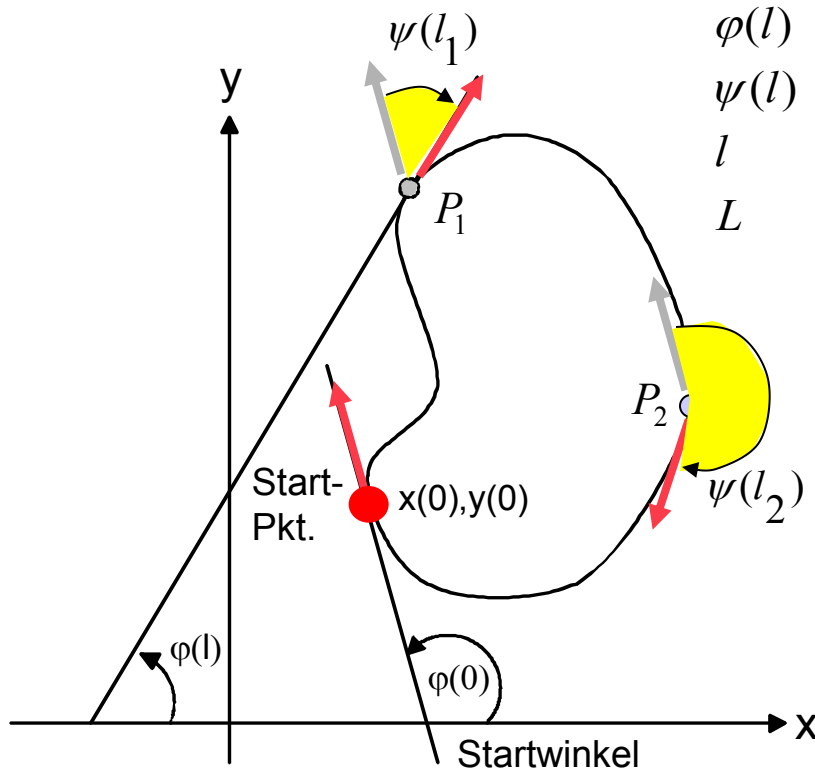
Anwendbar auf alle geschlossenen Konturen.



Teilweise aus: Gonzalez/Woods, Digital Image Processing, Prentice Hall

Eine Möglichkeit - Konturidentifikation nach Zahn und Roskies

Eine Verfahrensvariante (Verfahren von *Zahn & Roskies*) basiert auf dem sog. "kumulativen Tangentialwinkel", (=periodische Konturfunktion).



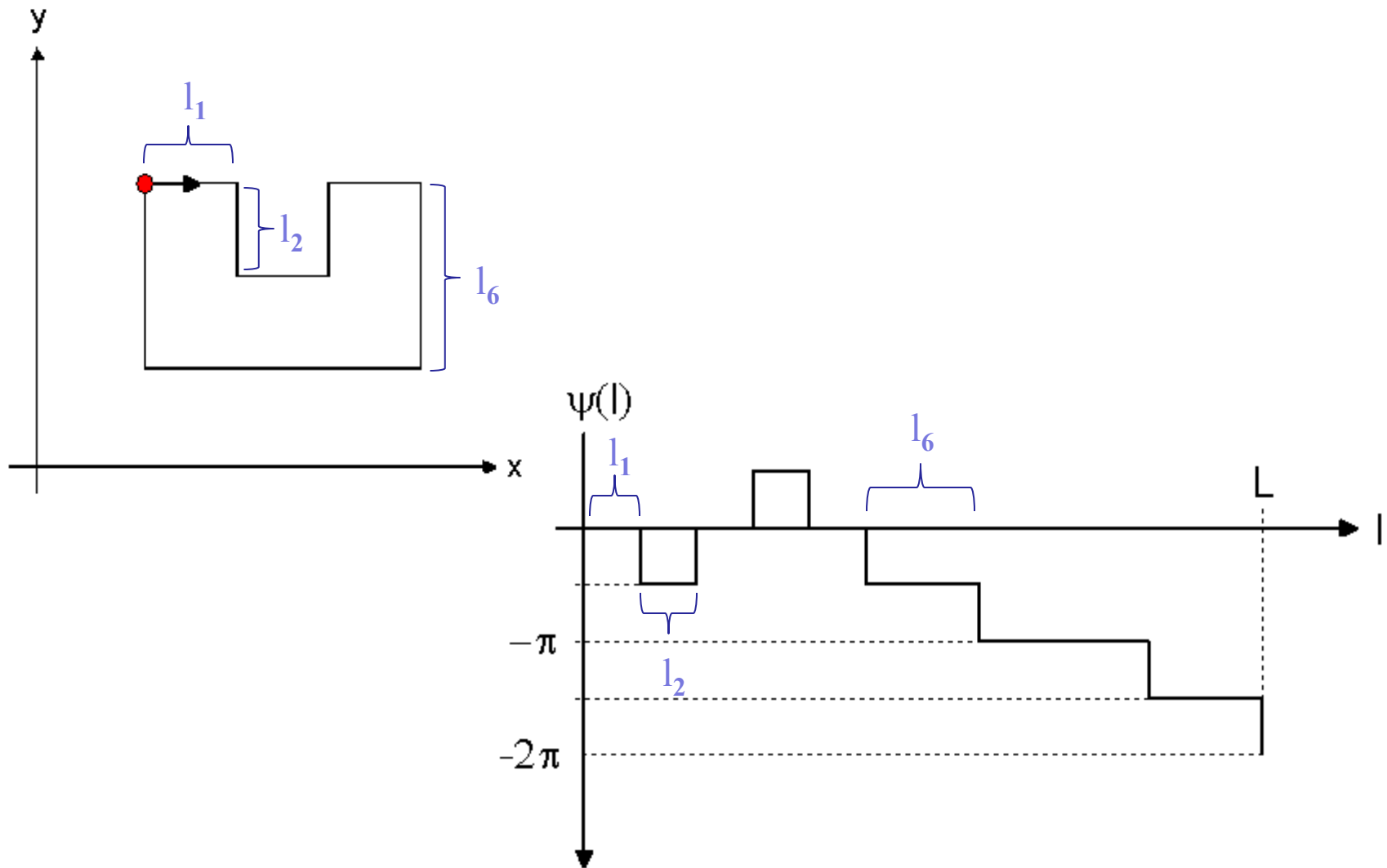
- $\varphi(l)$: absoluter Tangentialwinkel
- $\psi(l)$: kumulativer Tangentialwinkel
- l : Bogenlänge der Kontur, vom Startpunkt aus
- L : Gesamtkonturlänge

$$\psi(l) = \varphi(l) - \varphi(0)$$

Für ein Umlauf gilt dann: $\psi(L) = -2\pi$



BEISPIEL: Kumulativer Tangentialwinkel





Entwicklung einer periodischen Konturfunktion $\psi^*(l)$

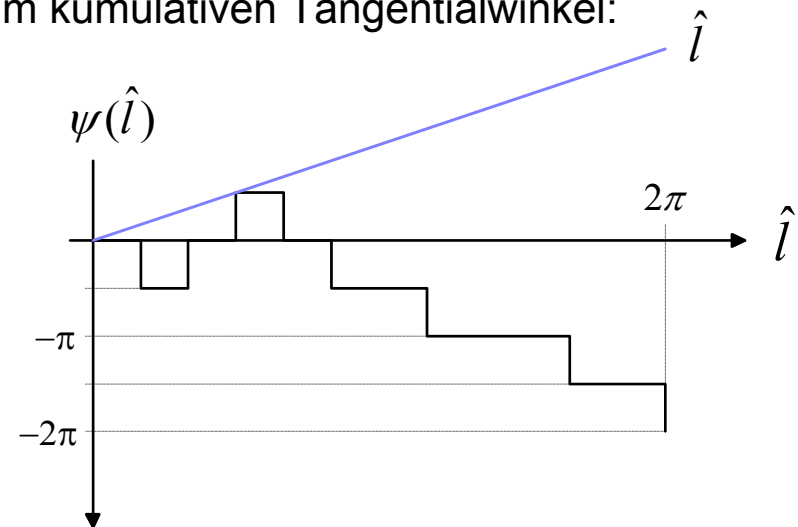
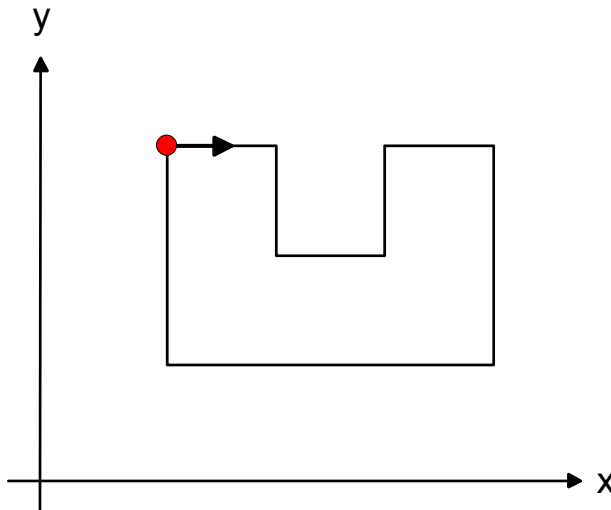
1. Bogenlänge l durch die *normierte Bogenlänge* $\hat{l} = \frac{l}{L} \cdot 2\pi$ ersetzen.
Für einen Umlauf gilt dann: $\hat{l}_{Uml} = 2\pi$

→ Tafel

2. Addition der normierten Bogenlänge zum kumulativen Tangentialwinkel:

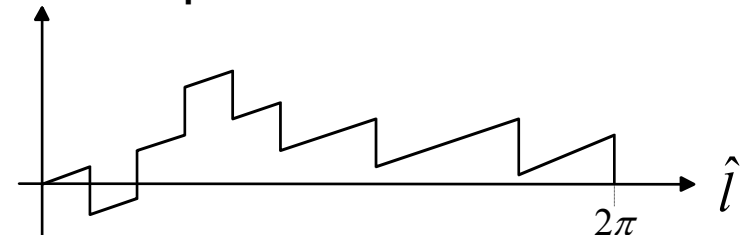
$$\psi^*(\hat{l}) = \psi(\hat{l}) + \hat{l}$$

→ periodische Funktion,
für die gilt: $\psi^*(0) = \psi^*(2\pi) = 0$



$$\psi^*(\hat{l})$$

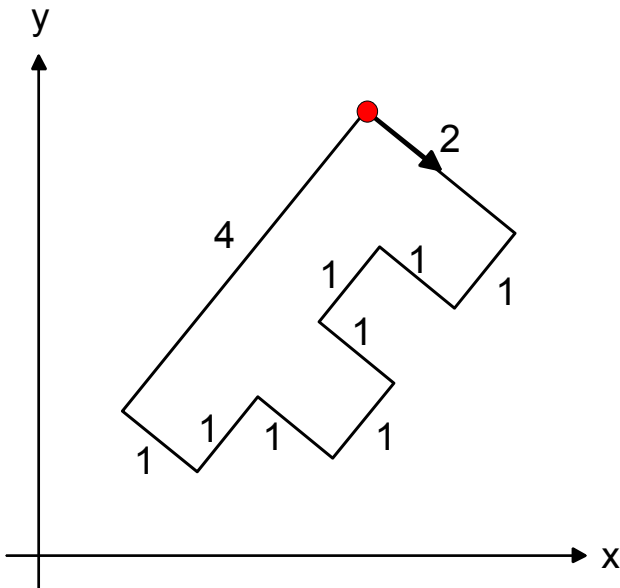
periodische Funktion !





ÜBUNG: Kumulativer Tangentialwinkel und Konturfunktion $\psi^*(\hat{l})$

Zeichnen Sie den kumulativen Tangentialwinkel und die Konturfunktion.



Wie ändert sich die Funktion

- a) bei Rotation,
- b) bei Skalierung,
- c) bei Verschiebung des Objektes,
- d) bei Startpunktverschiebung.



Fourierzerlegung der Konturfunktion

$\psi^*(\hat{l})$ ist invariant gegen Translation, Rotation und Skalierung der Kontur.
Es besteht lediglich eine Startpunktabhängigkeit.

Da $\psi^*(\hat{l})$ eine periodische Funktion ist, kann sie als Fourierreihe dargestellt werden:

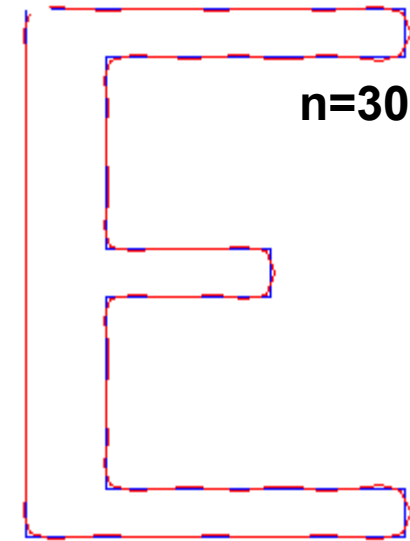
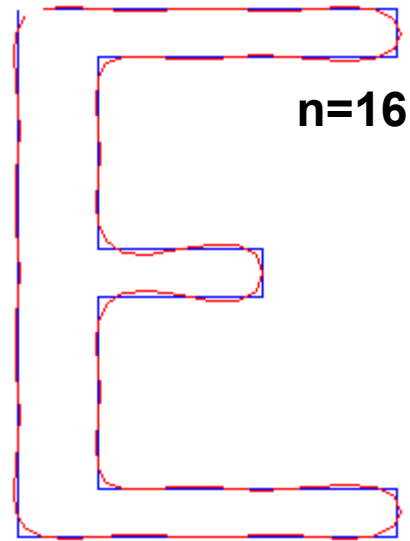
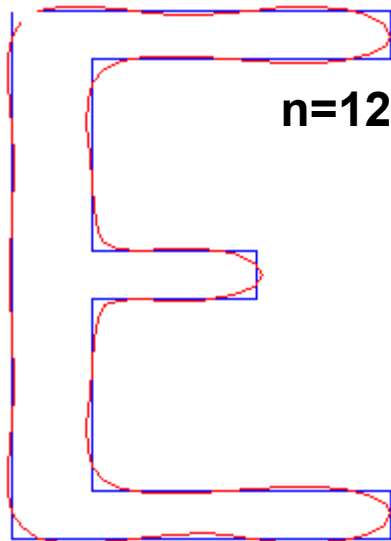
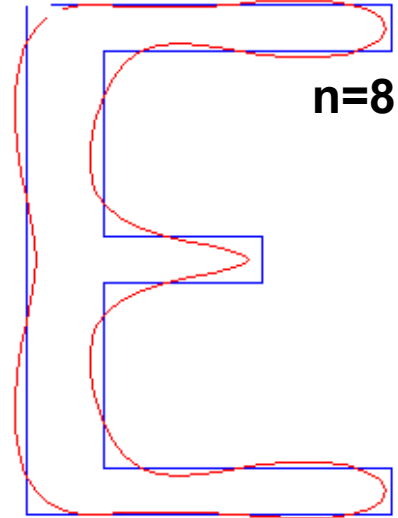
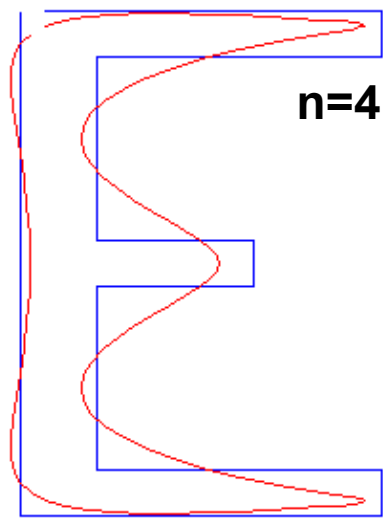
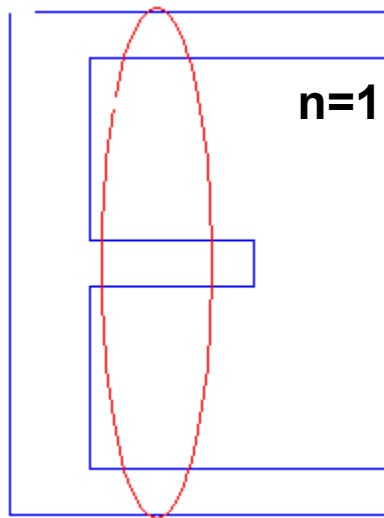
$$\begin{aligned}\psi^*(\hat{l}) &= \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos n\hat{l} + b_n \sin n\hat{l}) \quad \text{mit} \quad A_n = \sqrt{a_n^2 + b_n^2} \\ &= A_0 + \sum_{n=1}^{\infty} A_n \cos(n\hat{l} - \beta_n)\end{aligned}$$
$$\beta_n = \arctan(b_n / a_n)$$

Die Form der Kontur kann daher durch wenige Kennzahlen (A_n, β_n), die sog. Fourierdeskriptoren, dargestellt werden.

Eine Startpunktverschiebung wirkt sich nur auf A_0 und die Winkel β_n aus.



Rekonstruktion der Kontur aus den Fourierkoeffizienten



Berechnung der Fourierdeskriptoren aus polygonalen Konturfunktionen

Liegt die Konturbeschreibung als Polygonzug vor, so lassen sich die Fourierkoeffizienten (bzw. F.-Deskriptoren) besonders einfach berechnen/implementieren (o. Bew.):

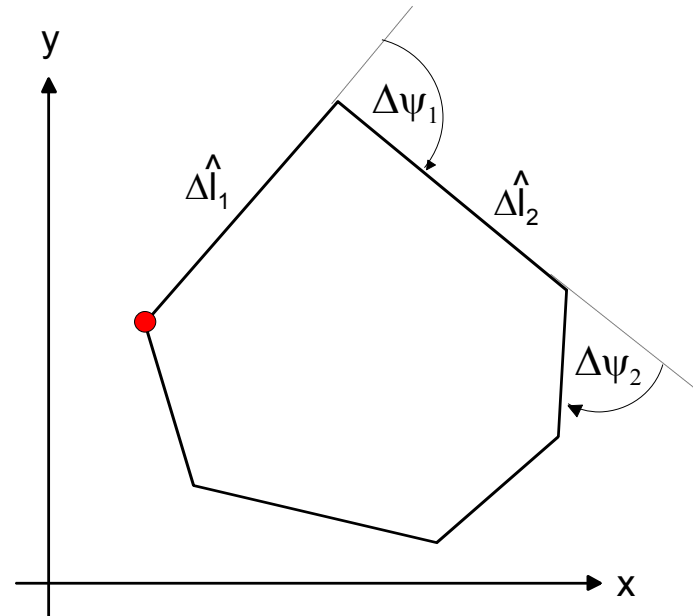
$$a_n = -\frac{1}{n\pi} \sum_{j=1}^m \Delta\psi_j \cdot \sin(n \cdot \hat{l}_j)$$

$$b_n = +\frac{1}{n\pi} \sum_{j=1}^m \Delta\psi_j \cdot \cos(n \cdot \hat{l}_j)$$

$$A_n = \sqrt{a_n^2 + b_n^2}$$

m: Gesamtzahl der Segmente

mit $\hat{l}_j = \sum_{i=1}^j \Delta\hat{l}_i \rightarrow$ normierte Teilkonturlänge



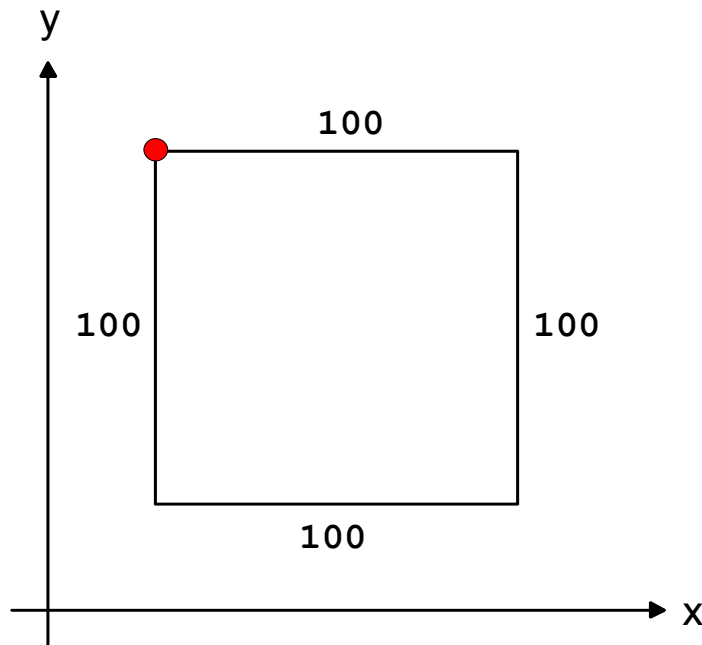


ÜBUNG: Fourierdeskriptoren

Schreiben Sie ein C-Programm zur Berechnung der Fourierdeskriptoren.

Berechnen Sie damit zu folgender Kontur (Längenangabe in Pixel) die Werte der ersten 15 Fourierkoeffizienten.

Variieren Sie die Längen- und Winkel geringfügig.

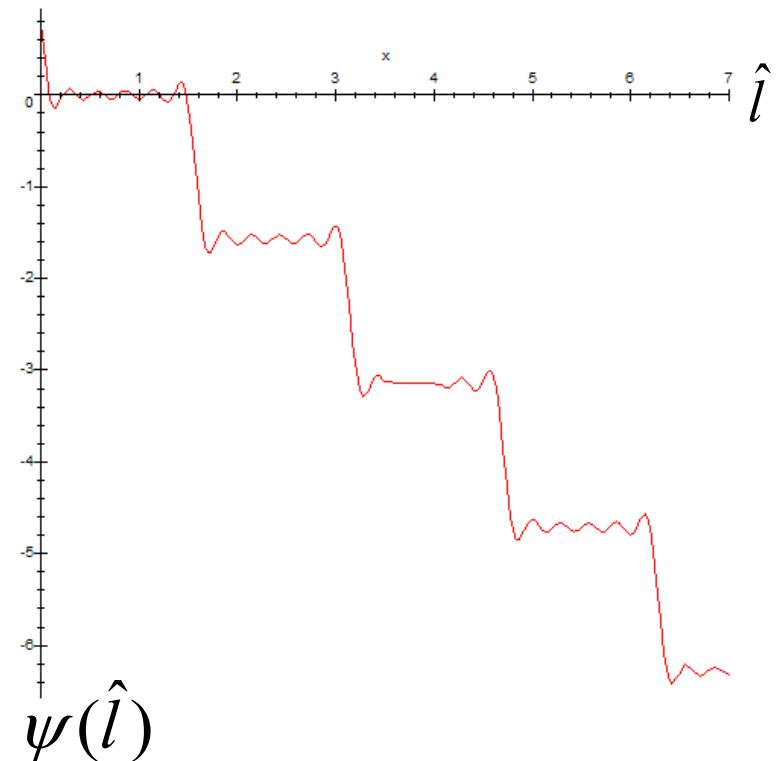
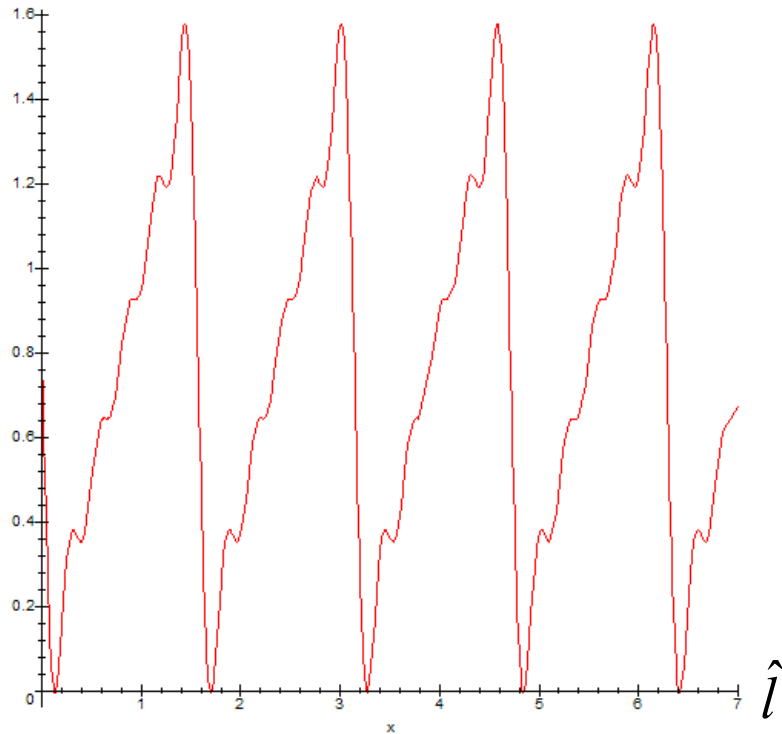




Ergebnis: Kumulativer Tangentialwinkel

$$\psi^*(\hat{l}) = \frac{1}{4}\pi - \frac{1}{2}\sin(4\hat{l}) - \frac{1}{4}\sin(8\hat{l}) - \frac{1}{6}\sin(12\hat{l}) - \frac{1}{8}\sin(16\hat{l}) - \frac{1}{10}\sin(20\hat{l})$$

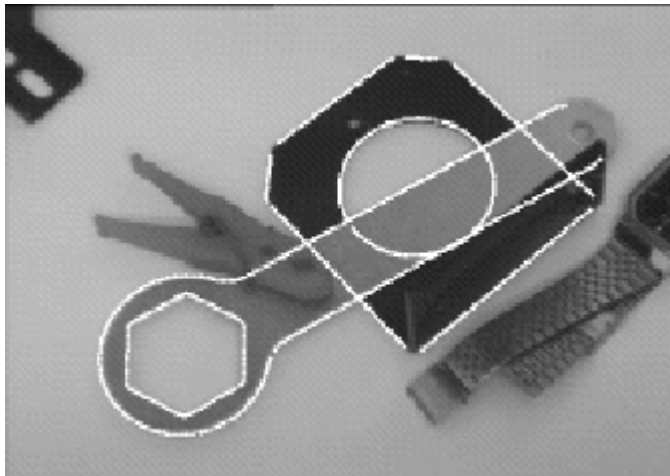
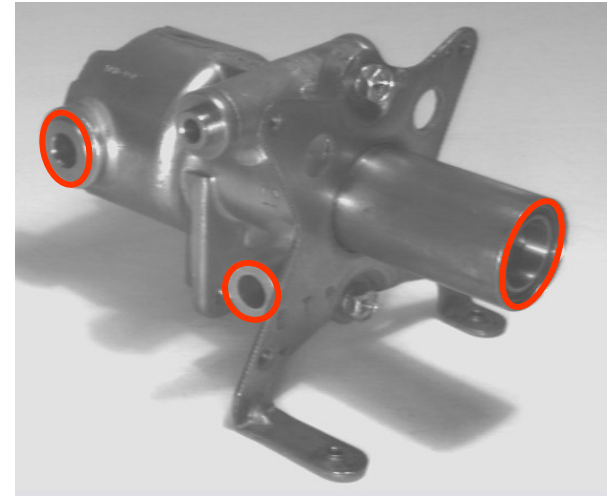
$$\psi^*(\hat{l})$$



4.2.3.5 Spezielle Konturen: Kegelschnitte

Kegelschnitte beschreiben

- Geraden
- Parabeln
- Kreise und Ellipsen
- Hyperbeln



Ein Kegelschnitt wird
beschrieben durch:

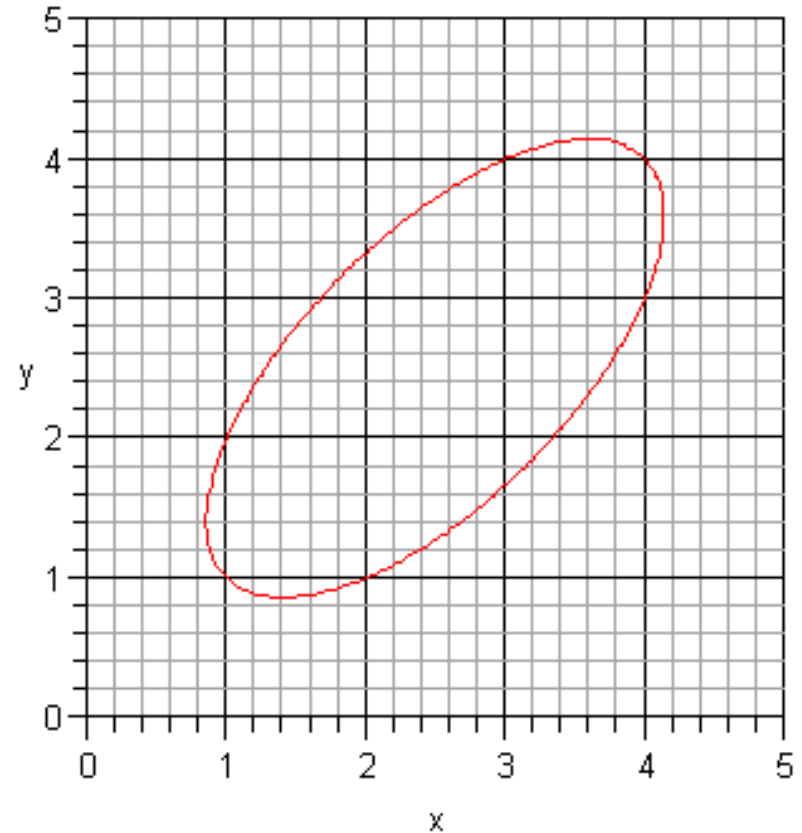
$$a \cdot x^2 + b \cdot xy + c \cdot y^2 + d \cdot x + e \cdot y + f = 0$$



ÜBUNG: Kegelschnittgleichung

$$x^2 - \frac{4}{3} \cdot xy + y^2 - \frac{5}{3} \cdot x - \frac{5}{3} \cdot y + \frac{8}{3} = 0$$

- a) Ändert sich die Ellipse, wenn die Kegelschnittparameter mit einer Zahl $k \neq 0$ multipliziert wird?
- b) Es ist zu zeigen, dass die obige Kegelschnittgleichung die dargestellte Ellipse beschreibt.





Beispiel: Berechnung der Kegelschnittparameter aus 5 Punkten

Gegeben sind 5 Punkte, die auf einer unbekannten Ellipse liegen.

$$\mathbf{x}_1 = (1, 2)^T \quad \mathbf{x}_2 = (2, 1)^T \quad \mathbf{x}_3 = (4, 3)^T$$

$$\mathbf{x}_4 = (4, 4)^T \quad \mathbf{x}_5 = (3, 4)^T$$

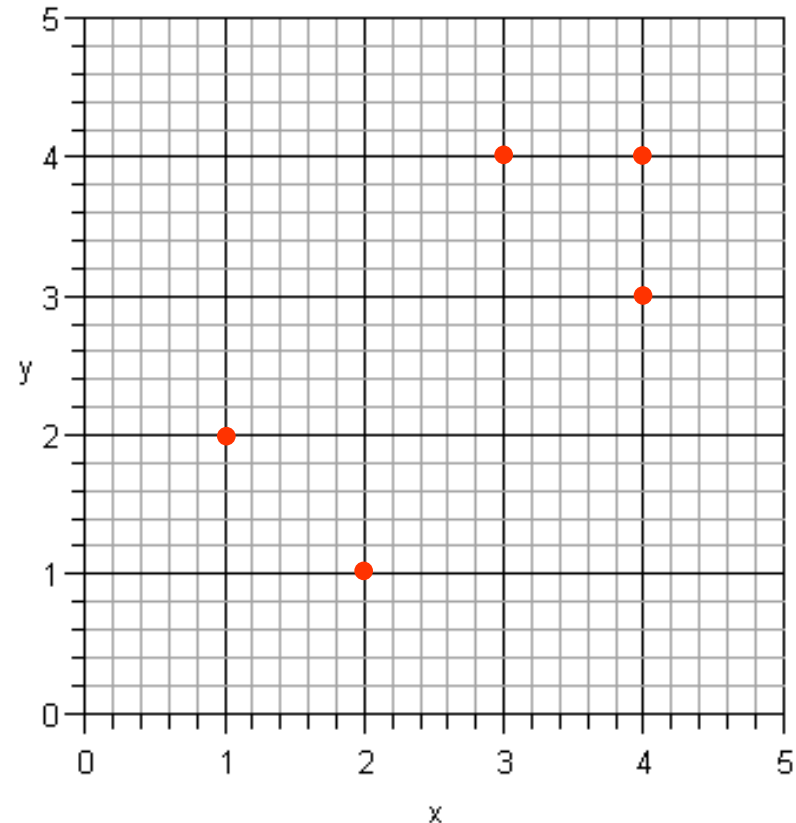
Die Parameter $a \dots f$ der Ellipse sollen bestimmt werden:

$$a \cdot x^2 + b \cdot xy + c \cdot y^2 + d \cdot x + e \cdot y + f = 0$$

Durch Division der Gleichung durch a wird aus der Kegelschnittgleichung:

$$\Rightarrow b^* \cdot xy + c^* \cdot y^2 + d^* \cdot x + e^* \cdot y + f^* = -x^2$$

→ Tafel





Für 5 Punkte und in Matrixform erhält man :

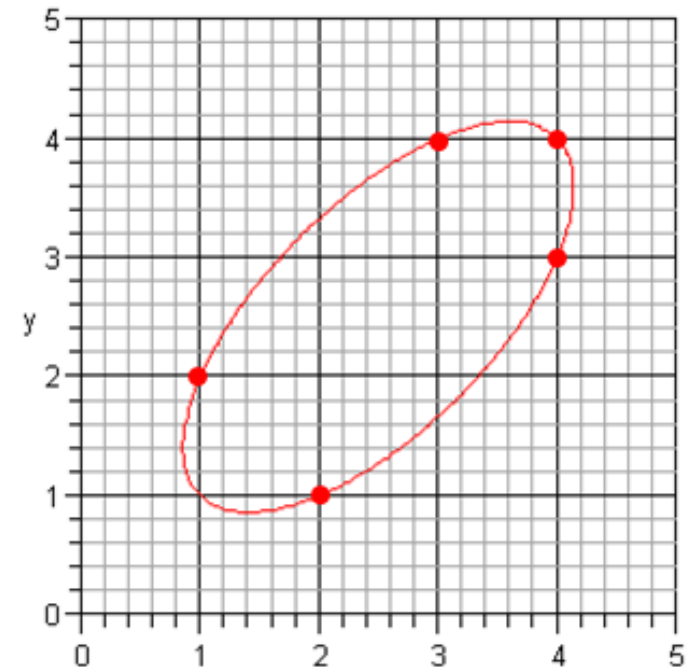
$$\begin{bmatrix} x_1 y_1 & y_1^2 & x_1 & y_1 & 1 \\ x_2 y_2 & y_2^2 & x_2 & y_2 & 1 \\ x_3 y_3 & y_3^2 & x_3 & y_3 & 1 \\ x_4 y_4 & y_4^2 & x_4 & y_4 & 1 \\ x_5 y_5 & y_5^2 & x_5 & y_5 & 1 \end{bmatrix} \cdot \begin{pmatrix} b^* \\ c^* \\ d^* \\ e^* \\ f^* \end{pmatrix} = - \begin{pmatrix} x_1^2 \\ x_2^2 \\ x_3^2 \\ x_4^2 \\ x_5^2 \end{pmatrix} \Rightarrow \begin{bmatrix} 2 & 4 & 1 & 2 & 1 \\ 2 & 1 & 2 & 1 & 1 \\ 12 & 9 & 4 & 3 & 1 \\ 16 & 16 & 4 & 4 & 1 \\ 12 & 16 & 3 & 4 & 1 \end{bmatrix} \cdot \begin{pmatrix} b^* \\ c^* \\ d^* \\ e^* \\ f^* \end{pmatrix} = \begin{pmatrix} -1 \\ -4 \\ -16 \\ -16 \\ -9 \end{pmatrix}$$

Durch Lösen des Gleichungssystems erhält man für $b^* \dots f^*$

$$\left[-\frac{4}{3} \quad +1 \quad -\frac{5}{3} \quad -\frac{5}{3} \quad +\frac{8}{3} \right]$$

Die Kegelschnittgleichung ist damit:

$$x^2 - \frac{4}{3} \cdot xy + y^2 - \frac{5}{3} \cdot x - \frac{5}{3} \cdot y + \frac{8}{3} = 0$$





ÜBUNG: Kreis

1. Zeigen Sie den Zusammenhang zwischen der Kreisgleichung

$$(x - x_0)^2 + (y - y_0)^2 = r^2$$

und der Kegelschnittgleichung:

$$a \cdot x^2 + b \cdot xy + c \cdot y^2 + d \cdot x + e \cdot y + f = 0$$

2. Geben Sie die Parameter x_0, y_0, r desjenigen Kreises an, der durch die Punkte $(2,1)$, $(-1,1)$ und $(0, -3)$ geht.



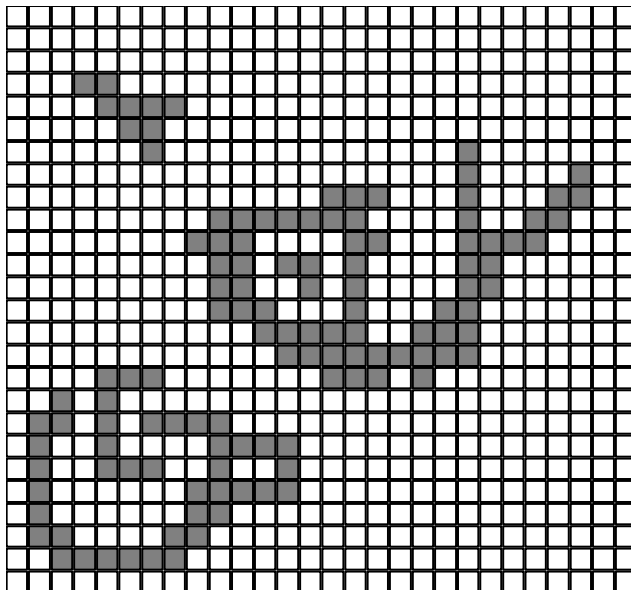
4.3 Regionenbasierte anhand von zusammenhängenden Flächen

4.3.1 Flächenextraktion in Binärbildern

4.3.1.1 Connected Components Labeling

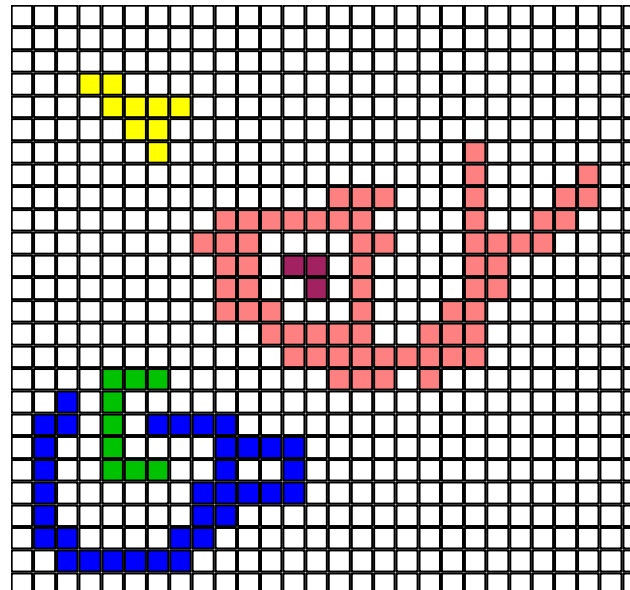
Ziel: Identifikation zusammenhängender Bildregionen in Binärbildern

Quellbild: Binärbild



□ = 0
■ = 1

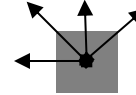
Ergebnisbild: „gelabeltes“ Bild



□ = 0 □ = 2 □ = 4
■ = 1 ■ = 3 ■ = 5



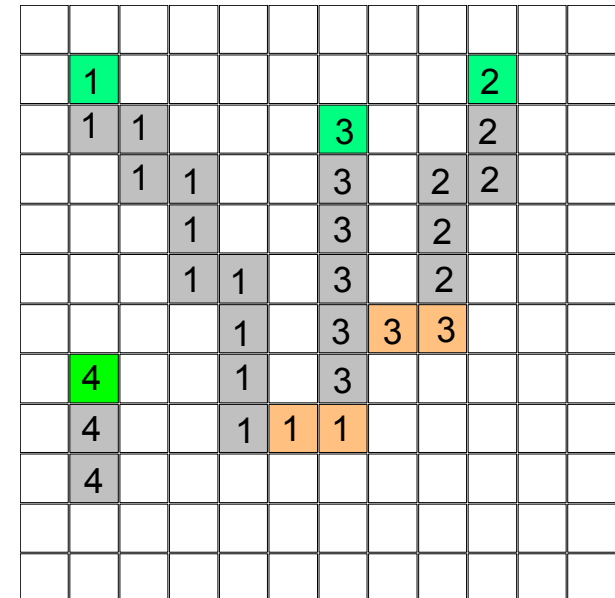
Def.: Nbr = Nachbarn in folgenden Richtungen



Algorithmus: Initiales Labeling (1. Schritt)

1.	Label = 0	1			
2.	for y = 0 to rows-1	1	1		
3.	for x = 0 to columns -1			1	1
4.	if Q[x][y] ≠ 0 (Objektpunkt gefunden)				1
5.	if alle 4 Nbr = 0				1
6.	Label = Label +1	4			
7.	Q[x][y] = Label	4			
8.	else if Nbr ≠ 0 und NbrLabel gleich	4			
9.	Q[x][y] = Label von Nbr				
10.	else Nbr ≠ 0 und NbrLabel verschieden				
11.	Q[x][y] = Label von einem Nbr				
12.	die anderen Labels als „äquivalent“ merken				
13.	end if				
14.	end if				
15.	end for				
16.	end for				

nach Ablauf des Algorithmus



```
Lab2=Lab3
Lab1=Lab3
```



Jetzt: Ersetzen äquivalenter Label durch ein Label in 3 Schritten:

Schritt 1: Eintragen der Label-Äquivalenzen in die *Äquivalenz-Matrix* (L).

Lab2=Lab3
Lab1=Lab3

initiales L \longrightarrow i

	1	2	3	4	5
1	1		1		
2		1	1		
3	1	1	1		
4				1	
5					

j \downarrow

nach Schritt 1

Die leeren Felder haben
den Wert 0
(aus Übersichtlichkeits-
gründen weggelassen).



Schritt 2: Auflösen der Äquivalenzen mit dem „Floyd-Warshall-Algorithmus“.

Algorithmus: Äquivalenzen auflösen (2. Schritt: Floyd-Warshall-Alg.)

```

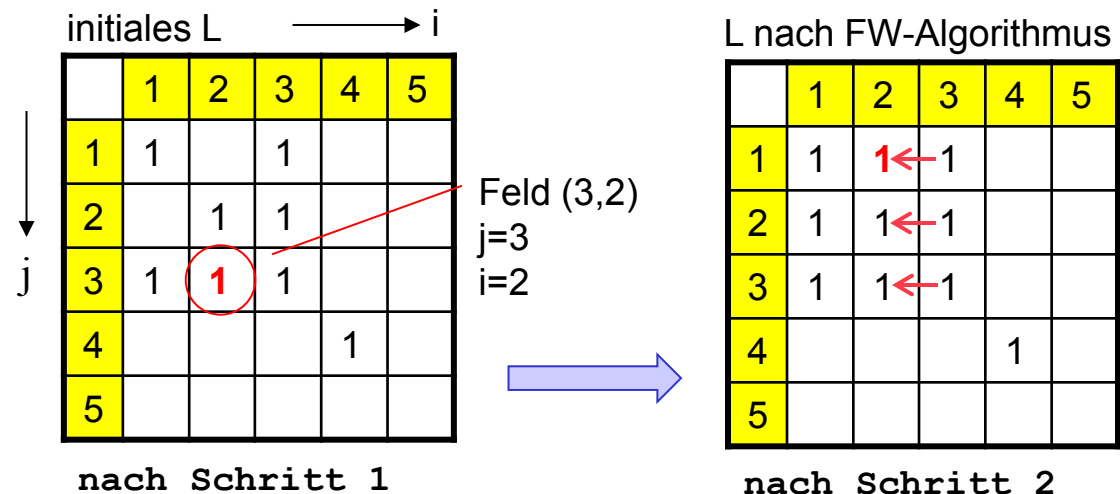
1.  for j = 1 to n
2.      for i = 1 to n
3.          if  $L_{ji} = 1$ 
4.              for k = 1 to n
5.                   $L_{ki} = L_{ki} \text{ OR } L_{kj}$ 
6.              end for
7.          end if
8.      end for
9.  end for

```

Algorithmus in Worten:

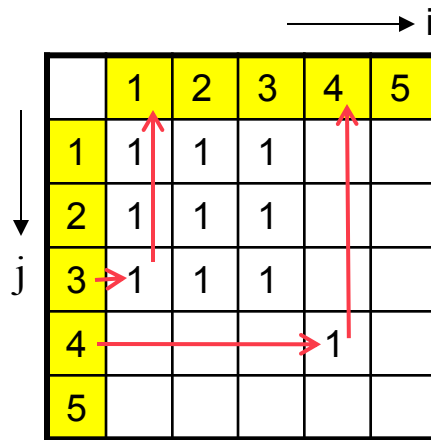
Wenn ein Punkt $L_{ji}=1$ ist, dann werden die gesetzten Felder (Wert=1) der Spalte **j** in die Felder der Spalte **i** kopiert.

Lab2=Lab3
Lab1=Lab3





Schritt 3: Ersetze im initial gelabelten Bild die alten Bildpunktlabel j durch die neuen Label i mit Hilfe der Funktion „Ersetze_Labelwert_ j _durch_ i (j)“



```

1.  Ersetze_Labelwert_ $j$ _durch_ $i$  ( $j$ ) {
2.      for  $i = 1$  to  $n$ 
3.          if  $L_{ji} = 1$ 
4.              return  $i$ 
5.          end if
6.      end for
7.  }
```

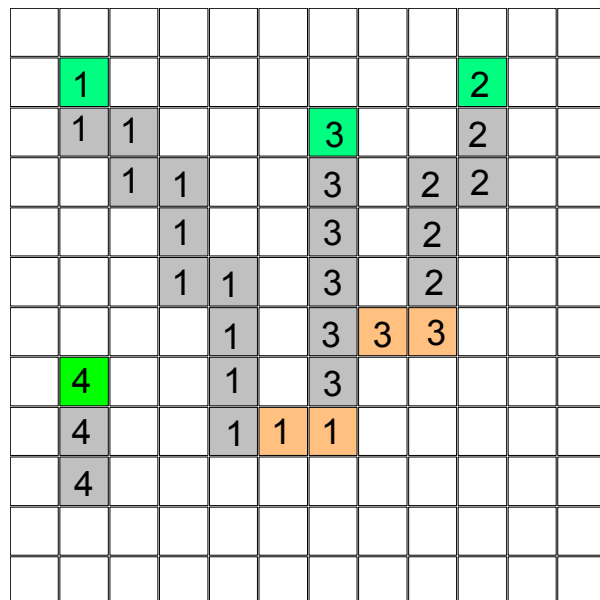
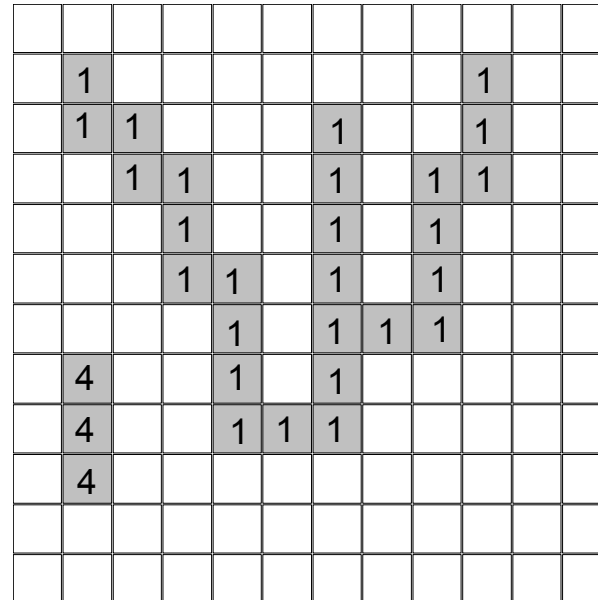


Bild nach Schritt 3





ÜBUNG: Floyd-Warshall-Algorithmus

Berechnen Sie die Äquivalenzmatrix mit Hilfe des FW-Algorithmus wenn folgende Äquivalenzen gegeben sind:

Lab 1 = Lab 3
Lab 2 = Lab 5
Lab 3 = Lab 4
Lab 5 = Lab 7
Lab 4 = Lab 8
Lab 7 = Lab 6

→ i

	1	2	3	4	5	6	7	8
1								
2								
3								
4								
5								
6								
7								
8								

↓ j

	1	2	3	4	5	6	7	8
1								
2								
3								
4								
5								
6								
7								
8								

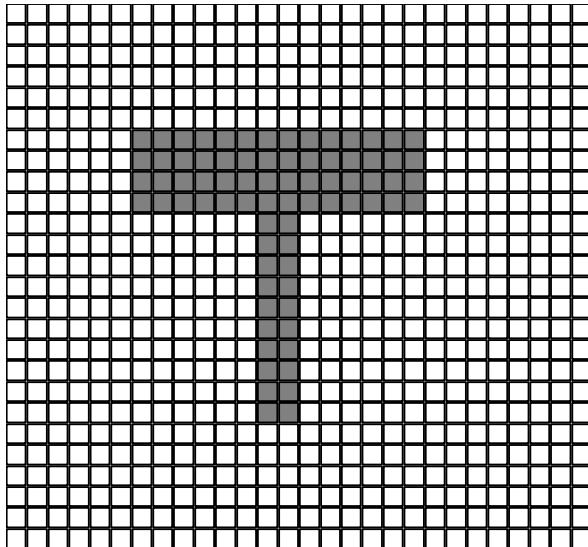


4.3.2 Flächenmerkmale

4.3.2.1 Geometrische Momente

Geometrische Momente werden in der Mechanik zur Beschreibung mechanischer Eigenschaften verwendet, die aus Querschnittsflächen abgeleitet werden können (Beispiel: Flächenträgheitsmoment).

Das Konzept der Flächenbeschreibung durch geom. Momente lässt sich für die (einfache) Mustererkennung zweidimensionaler Objekte nutzen.



Vorteil dieses Ansatzes ist, dass einige dieser Momente (Schwerpunkt, Flächenträgheitsmoment) anschaulich interpretiert werden können.



4.3.2.2 Definition der geometrischen Momente

Ein *geometrisches Moment* eines zweidimensionalen Objektes der Ordnung (p,q) ist definiert als:

$$m_{pq} = \sum_x \sum_y f(x, y) \cdot x^p \cdot y^q \quad \text{mit} \quad \begin{array}{l} f(x,y) : \text{Grauwert an der Stelle } (x,y) \\ x,y : \text{Koordinaten des Objektpunktes} \end{array}$$

So ist der Schwerpunkt (\bar{x}, \bar{y}) eines zweidimensionalen Objektes:

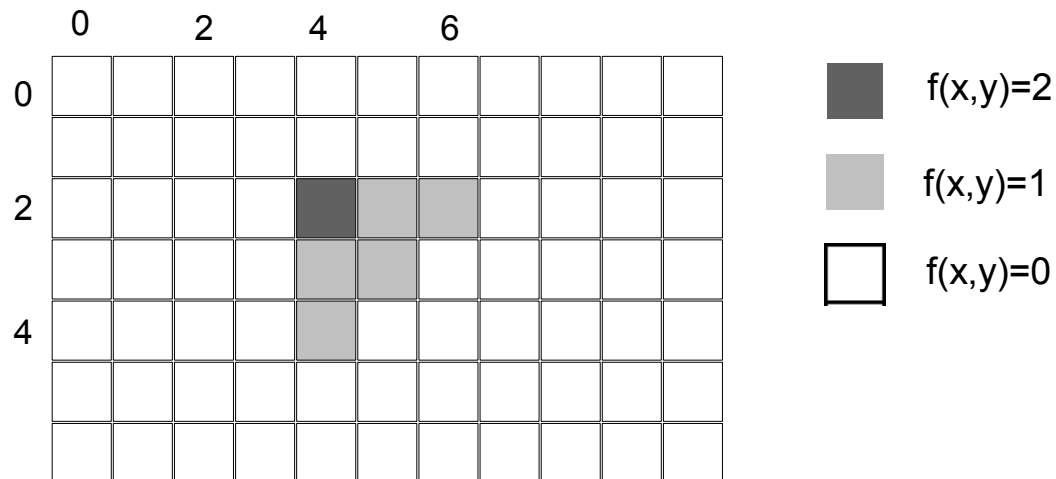
$$\bar{x} = \frac{m_{10}}{m_{00}} \quad \bar{y} = \frac{m_{01}}{m_{00}}$$

Die so definierten Momente sind jedoch abhängig vom Ort des zweidimensionalen Objektes (d.h. nicht translationsinvariant). → Zentralmomente



ÜBUNG: Berechnung des Objektschwerpunktes

a) Geben Sie des Schwerpunkt des Objektes an.



b) Geben Sie das Moment m_{20} des Objektes an.



4.3.2.3 Definition der Zentralmomente

Ein *Zentralmoment* der Ordnung (p,q) ist definiert als:

$$\mu_{pq} = \sum_x \sum_y f(x,y) \cdot (x - \bar{x})^p \cdot (y - \bar{y})^q \quad \text{mit} \quad \begin{array}{ll} f(x,y) & : \text{Grauwert an der Stelle } (x,y) \\ x,y & : \text{Koordinaten des Objektbildpunktes} \\ \bar{x}, \bar{y} & : \text{Objektschwerpunkt} \end{array}$$

Die Zentralmomente sind translationsinvariant,
aber nicht rotations- und skalierungsinvariant.

→ *normalisierte Zentralmomente*



4.3.2.4 Definition der normalisierten Zentralmomente

Die *normalisierten Zentralmomente* der Ordnung (p,q) sind definiert als:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \quad \gamma = \frac{p+q}{2} + 1$$

Die normalisierten Zentralmomente sind translations- und skalierungsinvariant, aber nicht rotationsinvariant (o.Bew.).

→ *invarianten Momente* (Hu, 1962)



4.3.2.5 Definition der invarianten Momente

Hu (1962) hat sieben Momente definiert, mit deren Hilfe sich geometrische Eigenschaften von Bildregionen translations-, skalierungs- und rotationsinvariant darstellen lassen.

$$p + q = 2$$

$$\phi_1 = \eta_{20} + \eta_{02}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$p + q = 3$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (\eta_{03} - 3\eta_{21})^2$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{03} + \eta_{21})^2$$

$$\begin{aligned} \phi_5 = & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & + (\eta_{03} - 3\eta_{21})(\eta_{03} + \eta_{21})[(\eta_{03} + \eta_{21})^2 - 3(\eta_{12} + \eta_{30})^2] \end{aligned}$$

$$\begin{aligned} \phi_6 = & (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ & + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{03} + \eta_{21}) \end{aligned}$$

$$\begin{aligned} \phi_7 = & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & + (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[(\eta_{03} + \eta_{21})^2 - 3(\eta_{30} + \eta_{12})^2] \end{aligned}$$

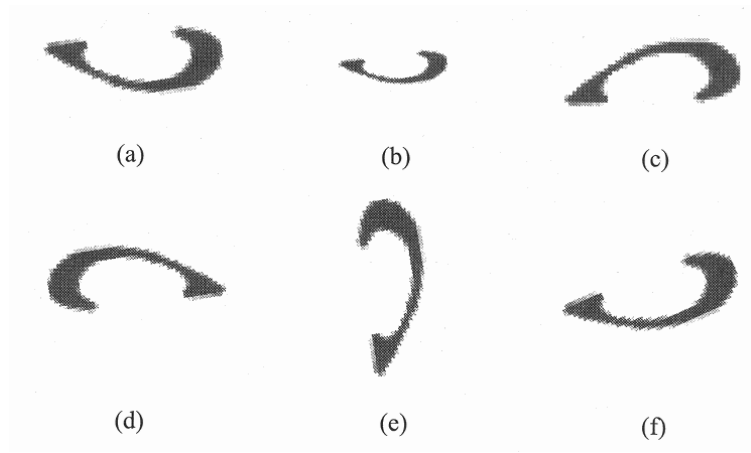
Formeln z.B. aus:

(1) Gonzalez/Woods, Digital Image Processing, Prentice Hall

(2) Theodoridis/Koutroumbas, Pattern Recognition, Academic Press



Beispiel: Invariante Momente von Binärobjekten



Moments	0°	Scaled	180°	15°	Mirror	90°
ϕ_1	93.13	91.76	93.13	94.28	93.13	93.13
ϕ_2	58.13	56.60	58.13	58.59	58.13	58.13
ϕ_3	26.70	25.06	26.70	27.00	26.70	26.70
ϕ_4	15.92	14.78	15.92	15.83	15.92	15.92
ϕ_5	3.24	2.80	3.24	3.22	3.24	3.24
ϕ_6	10.70	9.71	10.70	10.57	10.70	10.70
ϕ_7	0.53	0.46	0.53	0.56	-0.53	0.53

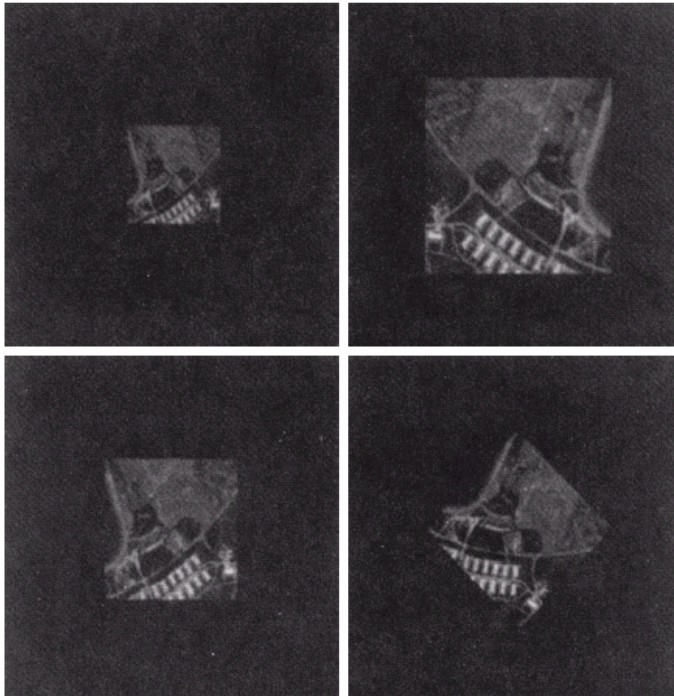
Beispiel aus:
Theodoridis/Koutroumbas, Pattern Recognition, Academic Press



Beispiel: Invariante Momente von Grauwertbildern



Invariant (Log)	Original	Half Size	Mirrored	Rotated 2°	Rotated 45°
ϕ_1	6.249	6.226	6.919	6.253	6.318
ϕ_2	17.180	16.954	19.955	17.270	16.803
ϕ_3	22.655	23.531	26.689	22.836	19.724
ϕ_4	22.919	24.236	26.901	23.130	20.437
ϕ_5	45.749	48.349	53.724	46.136	40.525
ϕ_6	31.830	32.916	37.134	32.068	29.315
ϕ_7	45.589	48.343	53.590	46.017	40.470



Beispiel aus:
Gonzalez/Woods, Digital Image Processing, Prentice Hall



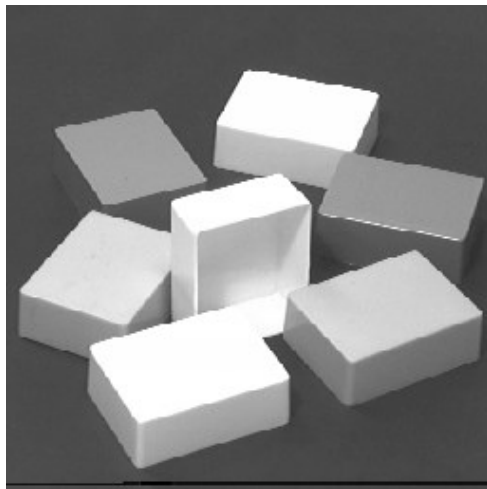
4.3.2.6 Beispielanwendungen (aus Literatur)

- Lesen chinesischer Schriftzeichen
- Inhaltsbezogender Zugriff auf Bilddatenbanken
- Bestimmung der Bewegungsparameter von Blutkörperchen
- Gestenerkennung
- Lageregelung für Helikopter anhand von Landemarkierungen (auf Schiffen)

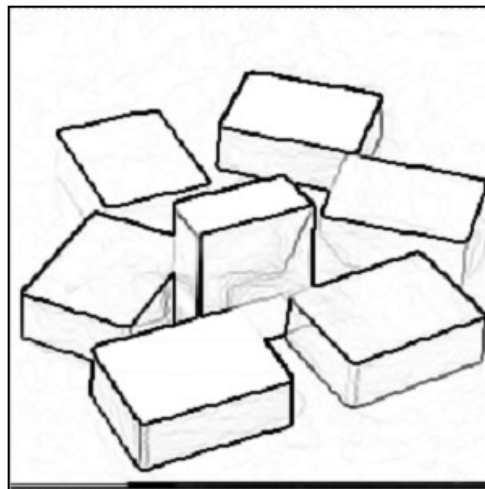
4.4 Kantenbasierte Verfahren

4.4.1 Geradenextraktion in Grauwertbildern durch Houghtransformation

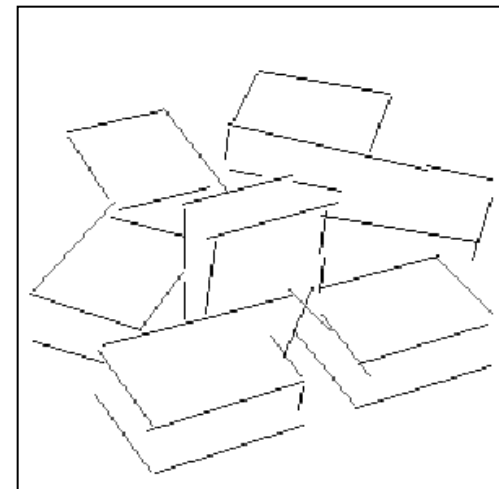
Ziel der Houghtransformation ist die Detektion und Lokalisierung von linearen Bildstrukturen, z.B. Objektkanten.



Grauwertbild



Gradientenbild
(Sobel)
invers dargestellt

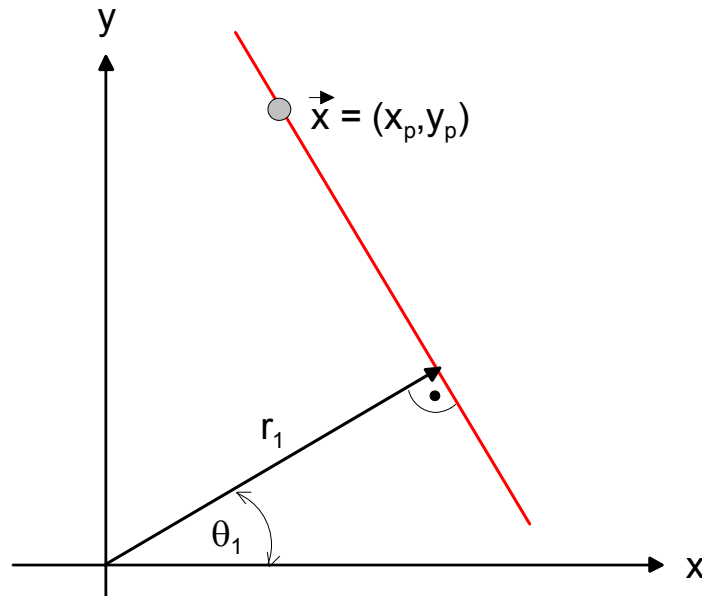


Extrahierte Bildstrecken
mit Houghtransformation

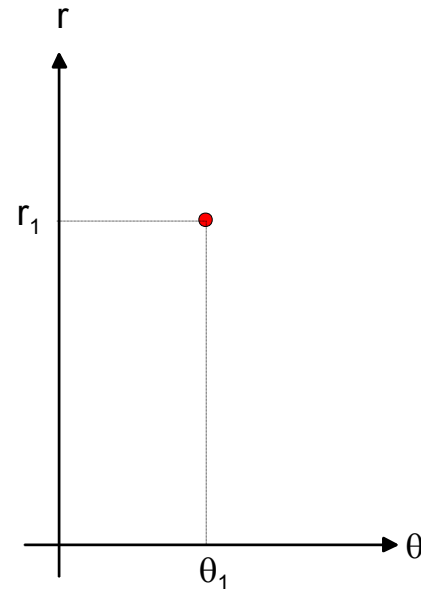


Mathematische Grundlagen

Grundlage der Houghtransformation ist die *Hesse'sche Normalform* der Geradengleichung.



$$r = x_p \cdot \cos \theta + y_p \cdot \sin \theta$$



Beschreibung der Gerade im Parameterraum (Houghraum)

Interpretation: Alle Punkte (x_p, y_p) , die auf der Gerade (rot) liegen, genügen der Hesse'schen Normalform. Die Gerade lässt sich durch die beiden Werte (r, θ) eindeutig beschreiben.



ÜBUNG: Hessesche Normalform 1

Prüfen Sie, welche Punkte auf der Gerade mit den folgenden Parametern liegen:

$$r = \frac{5}{\sqrt{2}}, \quad \theta = 45^\circ$$

a) $(x_1, y_1) = (1, 4)$

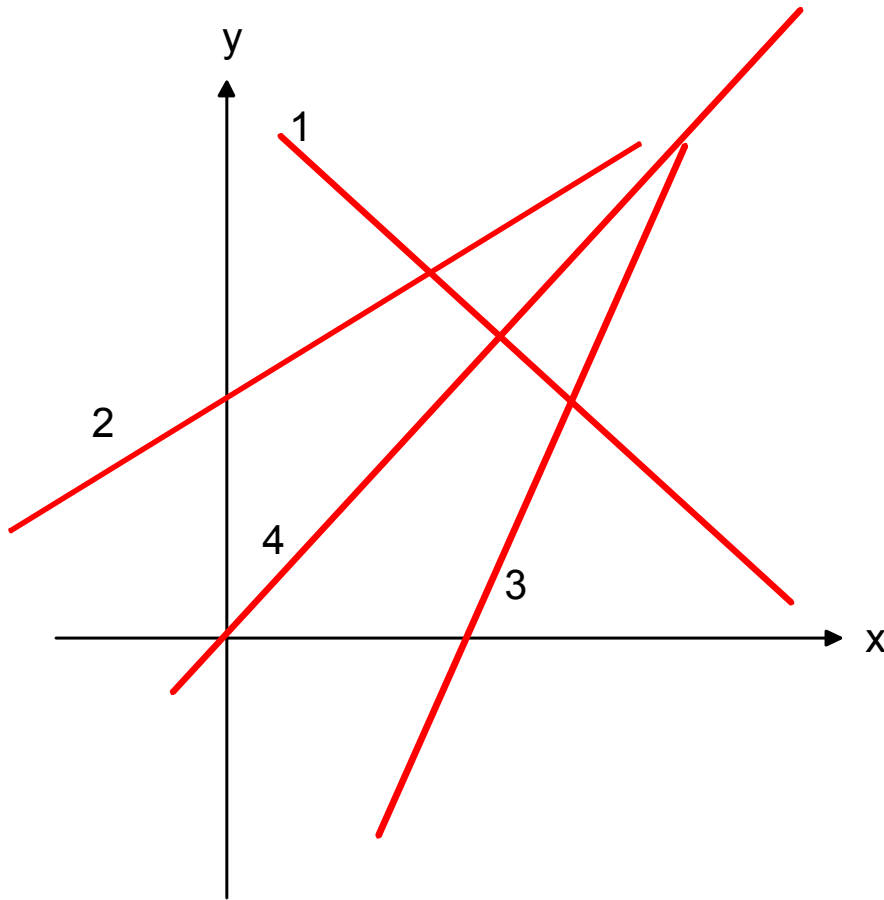
b) $(x_2, y_2) = (2, 2)$

c) $(x_3, y_3) = (3, 2)$



ÜBUNG: Hessesche Normalform 2

Skizzieren Sie für folgende Geraden den Parameterraum (Houghraum).



Frage:

Unter der Einschränkung, dass ein Teil der Gerade im 1. Quadranten liegt:

Wie ist der Wertebereich von θ ?

ÜBUNG: Liniendetektion – Diskussion der Aufgabenstellung

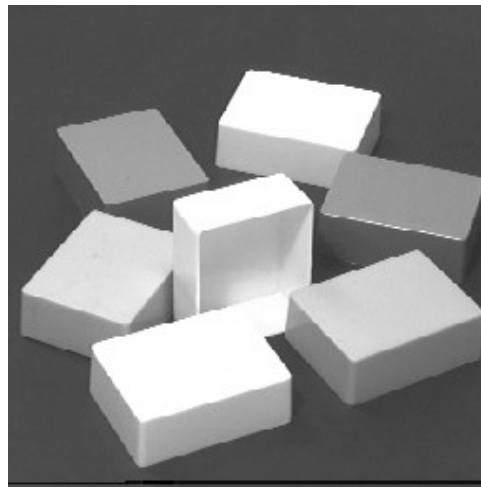
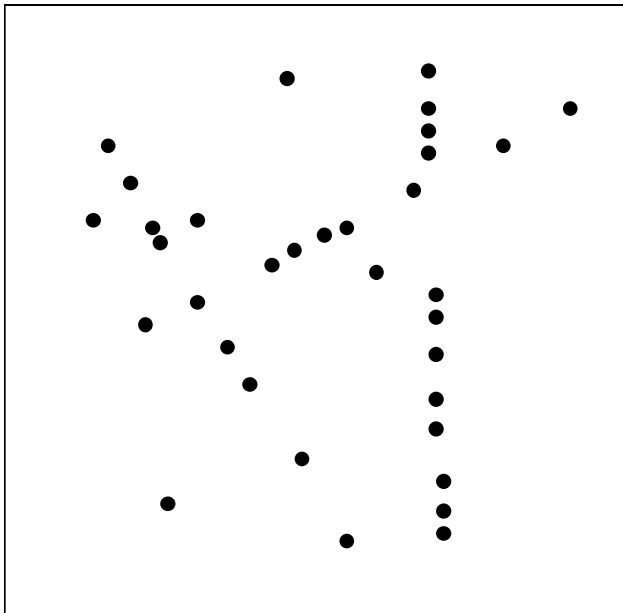
Folgende Aufgabenstellung sei gegeben:

In einem Bild sind einzelne Bildpunkte gesetzt (sw). Es ist zu prüfen, ob mehrere dieser Punkte auf einer Gerade liegen. Diskutieren Sie Lösungsmöglichkeiten.

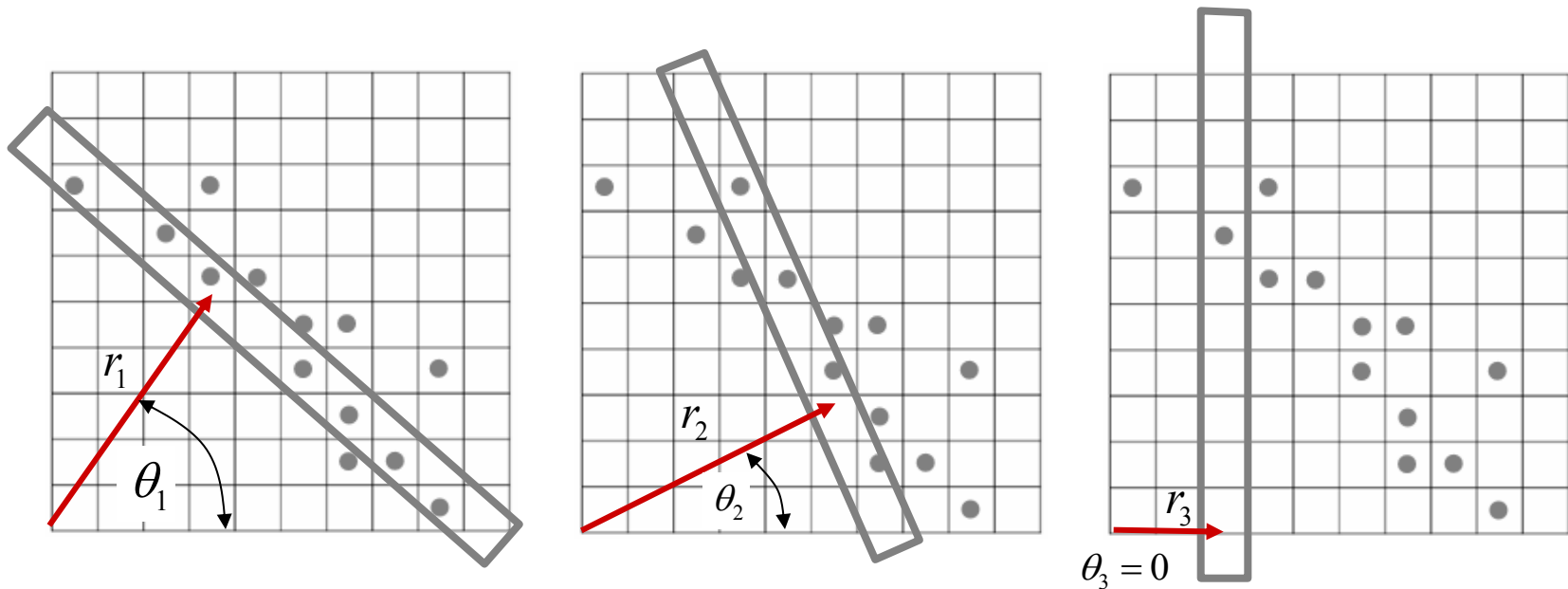
Hintergrund:

Diese Aufgabenstellung liegt z.B. dann vor, wenn die Kanten von geradlinig begrenzten Bildobjekten zu bestimmen sind.

Durch Kantenfilterung lässt sich die in der Aufgabenstellung beschriebene Situation herstellen.



ÜBUNG: Detektion kollinearier Bildpunkte mit Hilfe des Parameterraumes

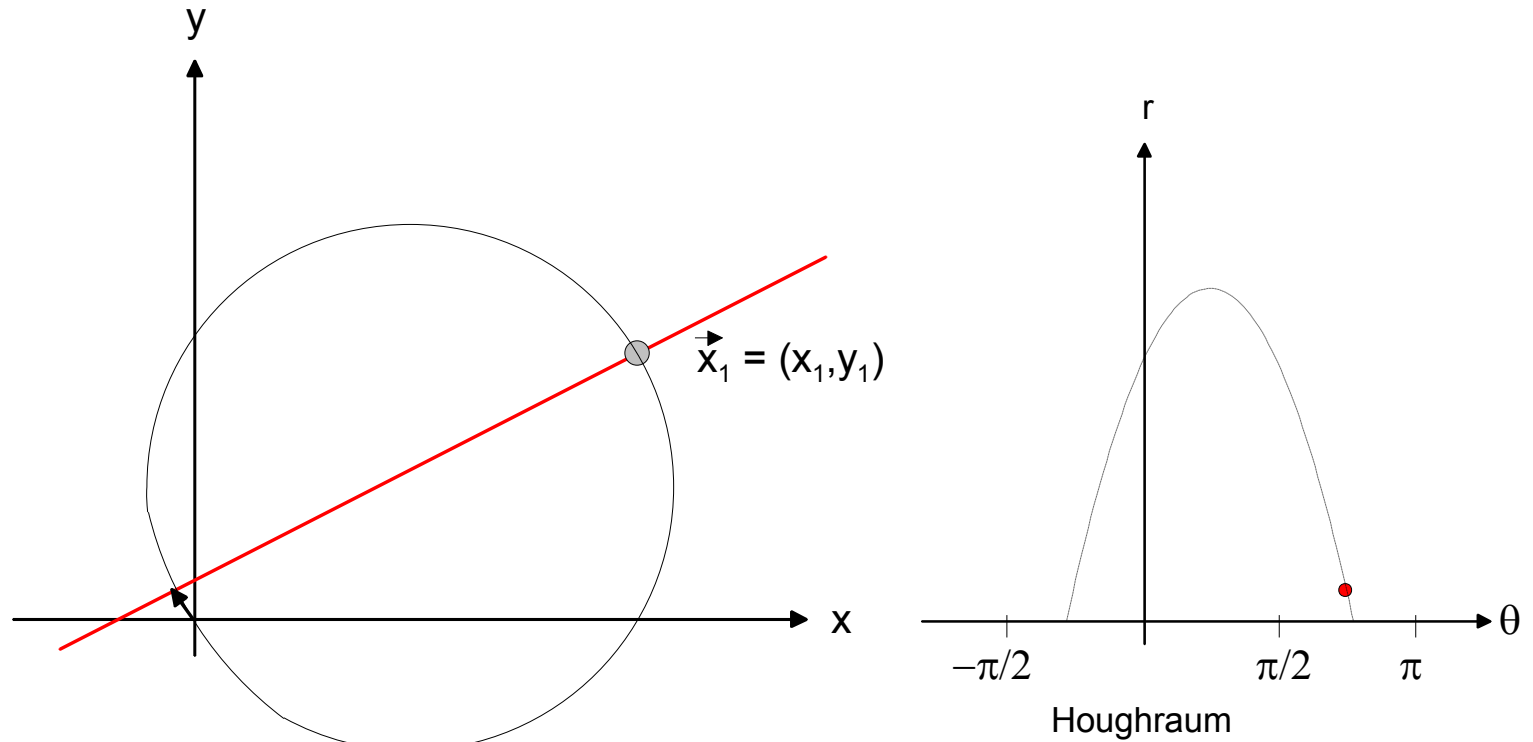


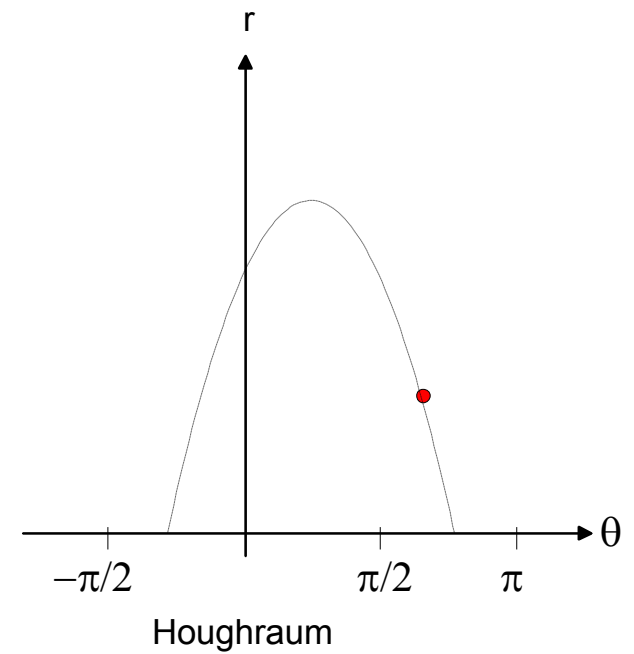
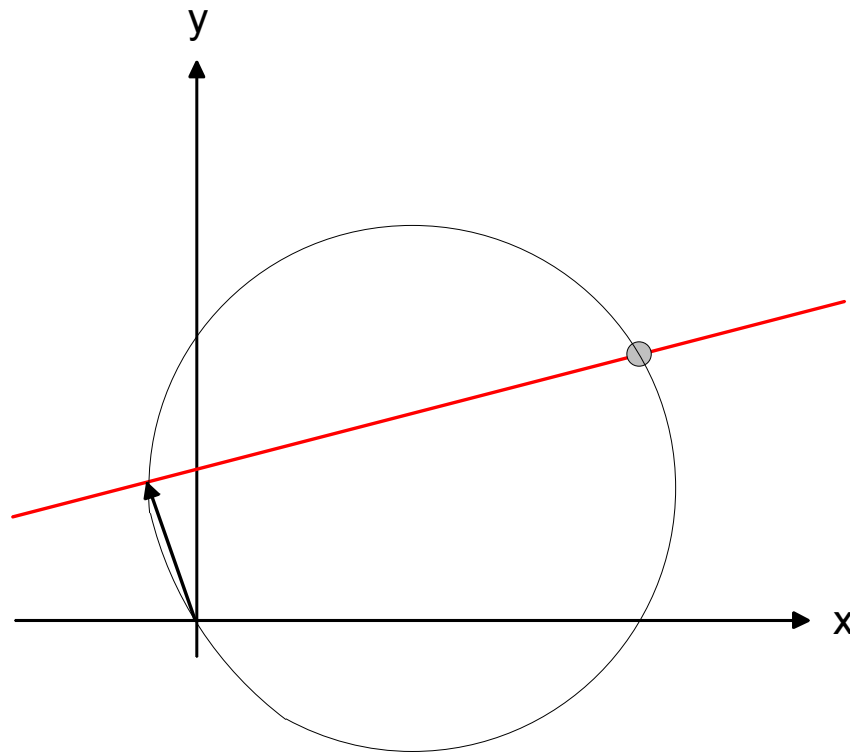
Welche Einträge erzeugen die Bildpunkte auf folgenden Geraden im Parameterraum ?

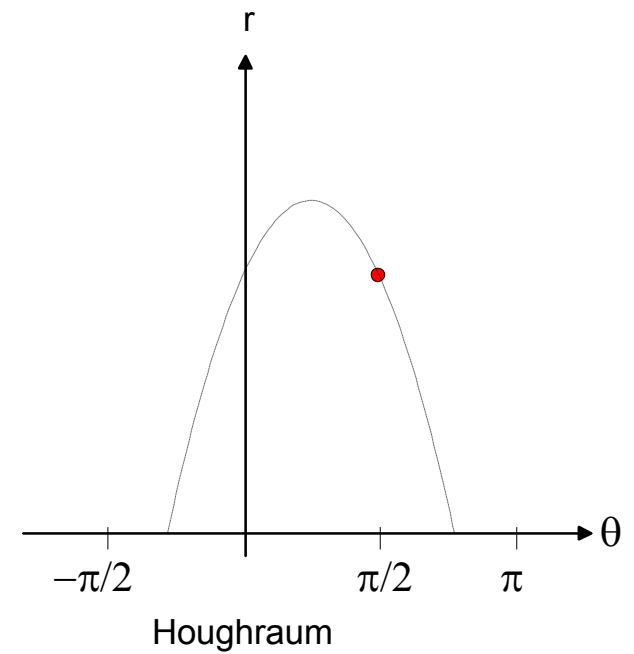
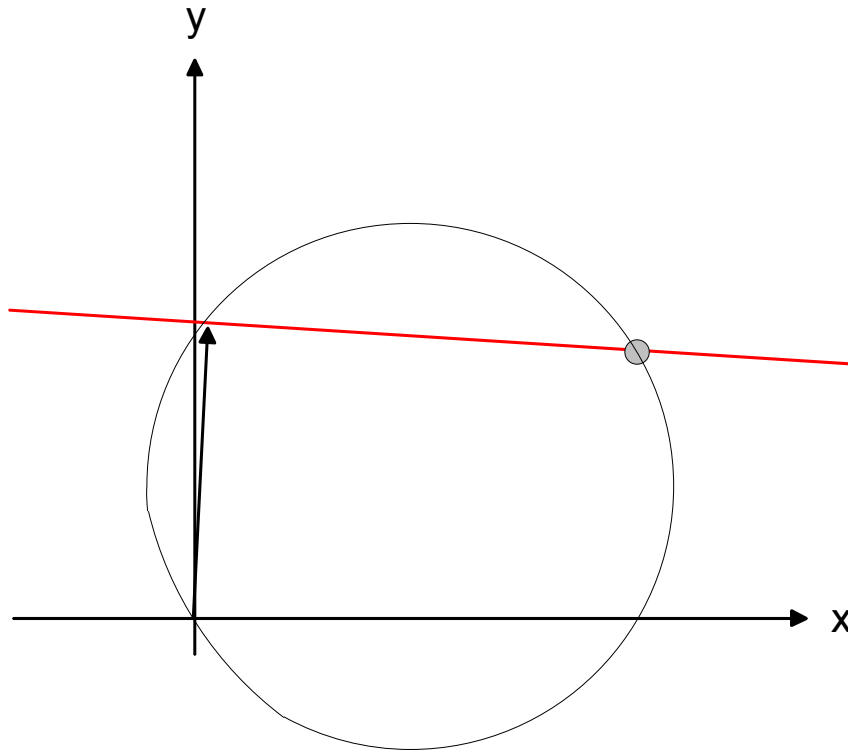
Wie könnte man den vollständigen Parameterraum bestimmen?

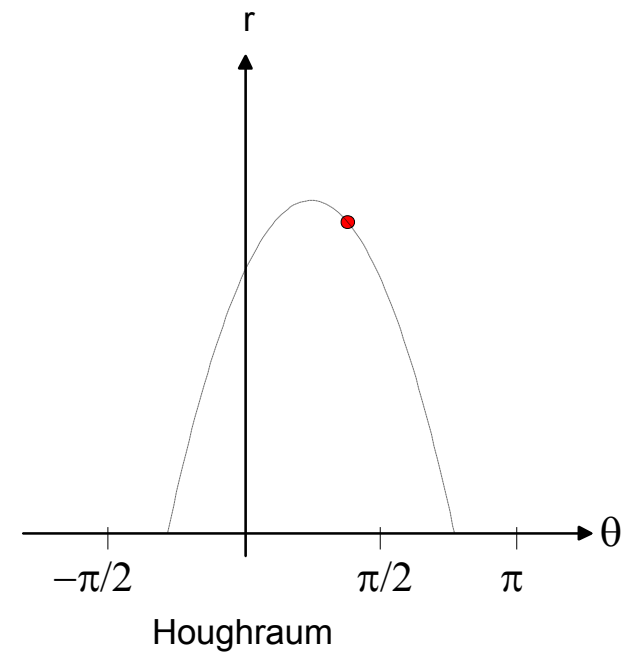
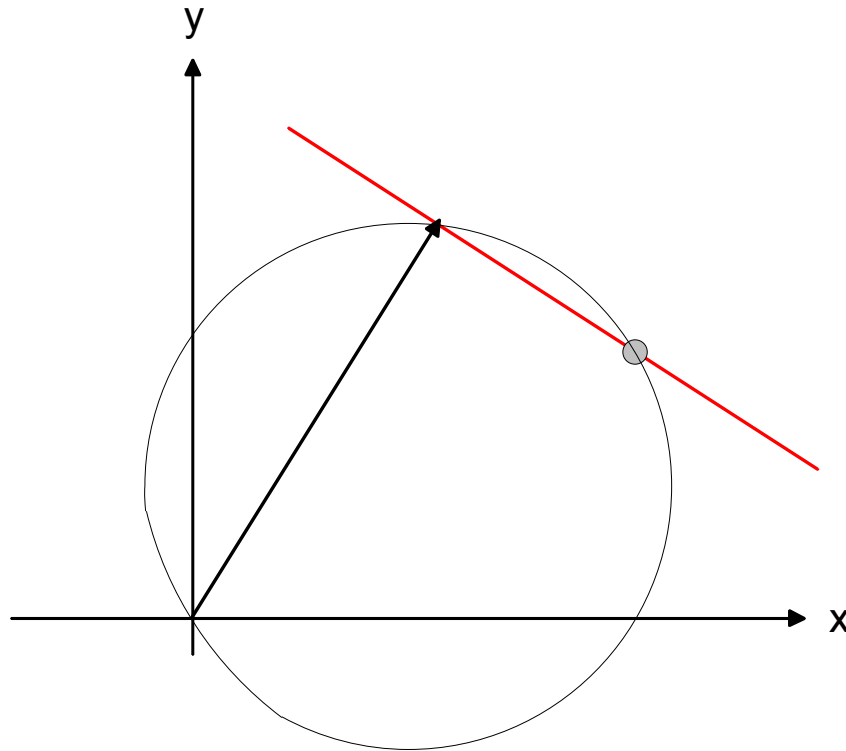


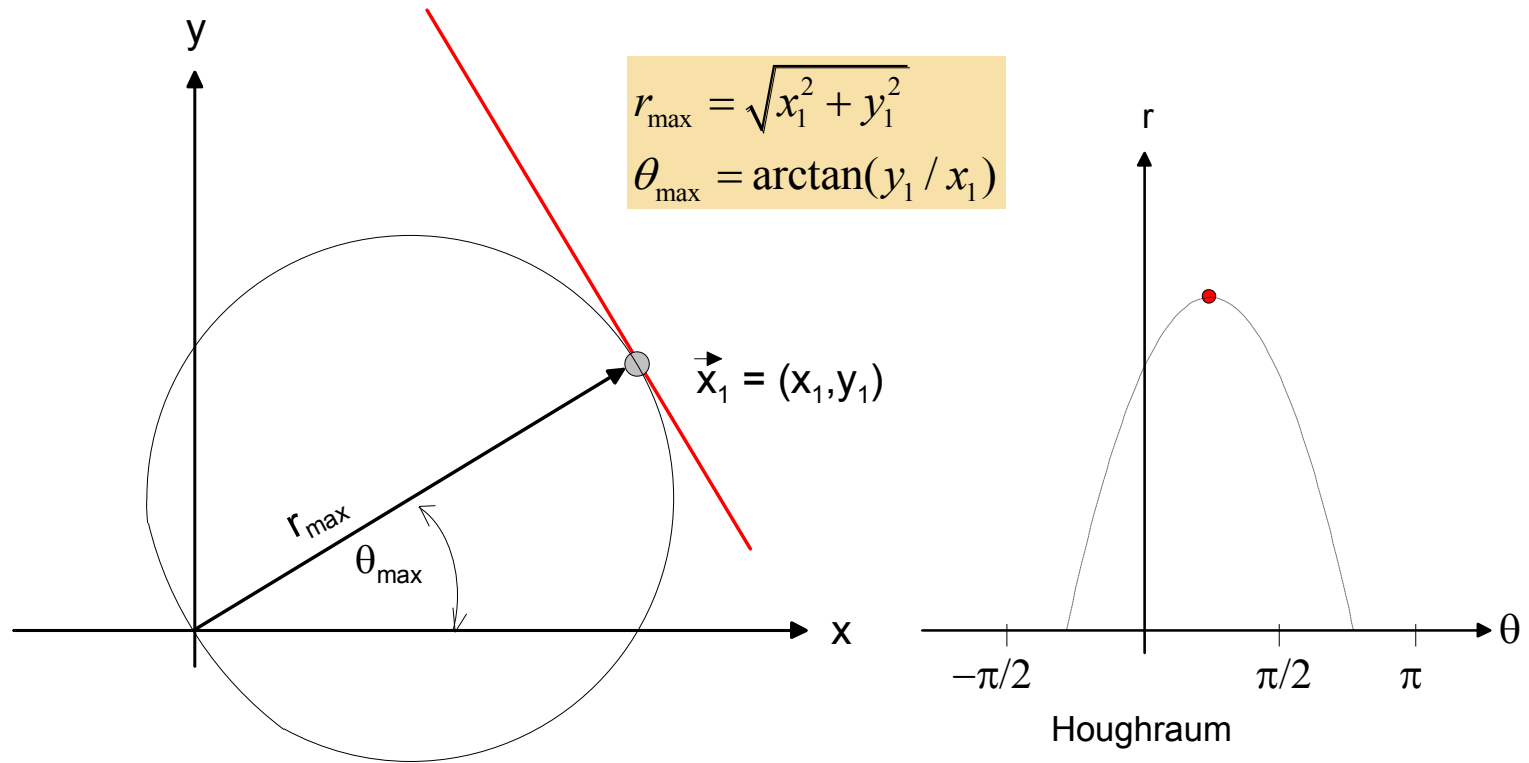
Gesamtheit aller durch einen Punkt gehenden Geraden

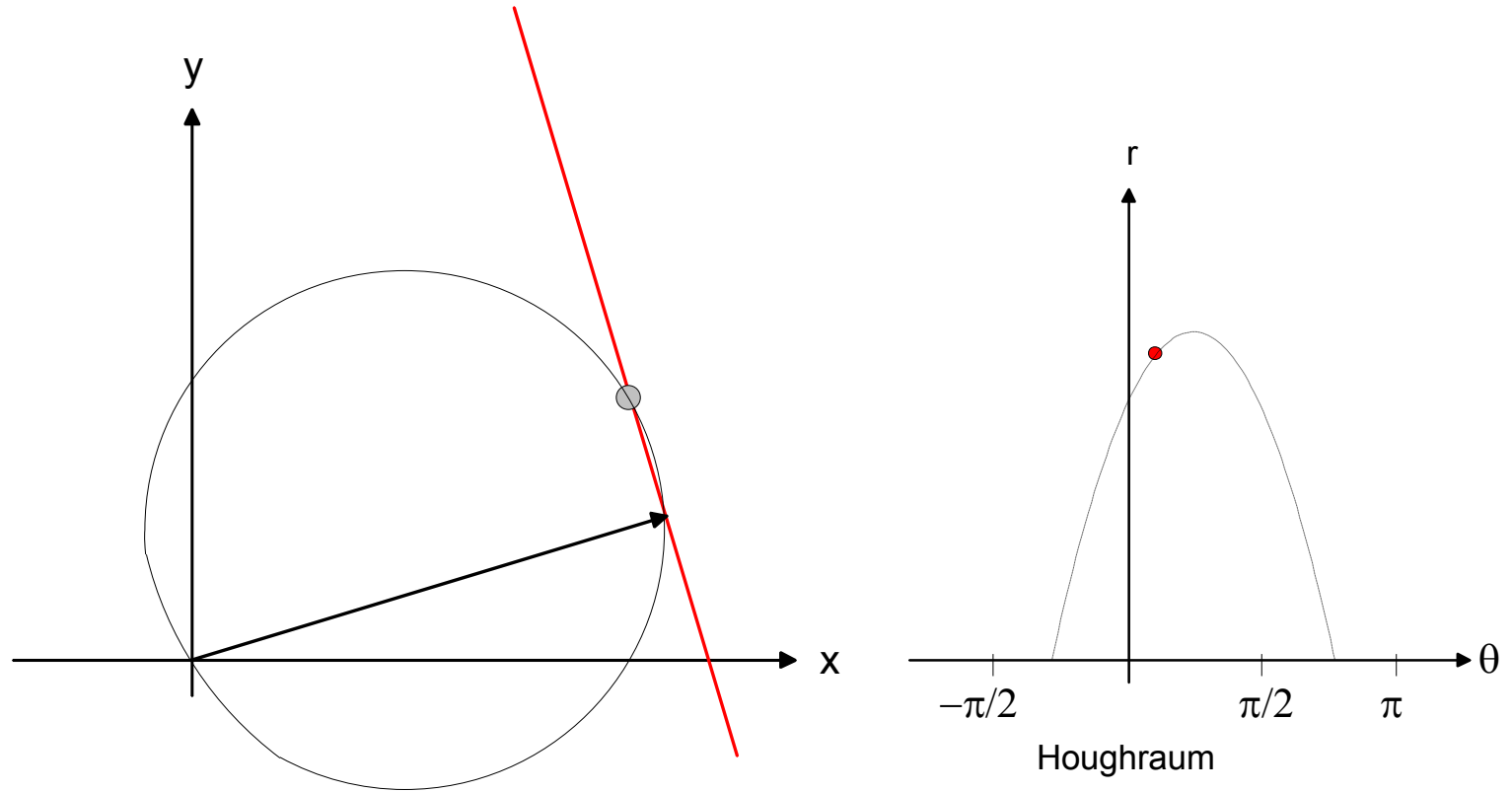


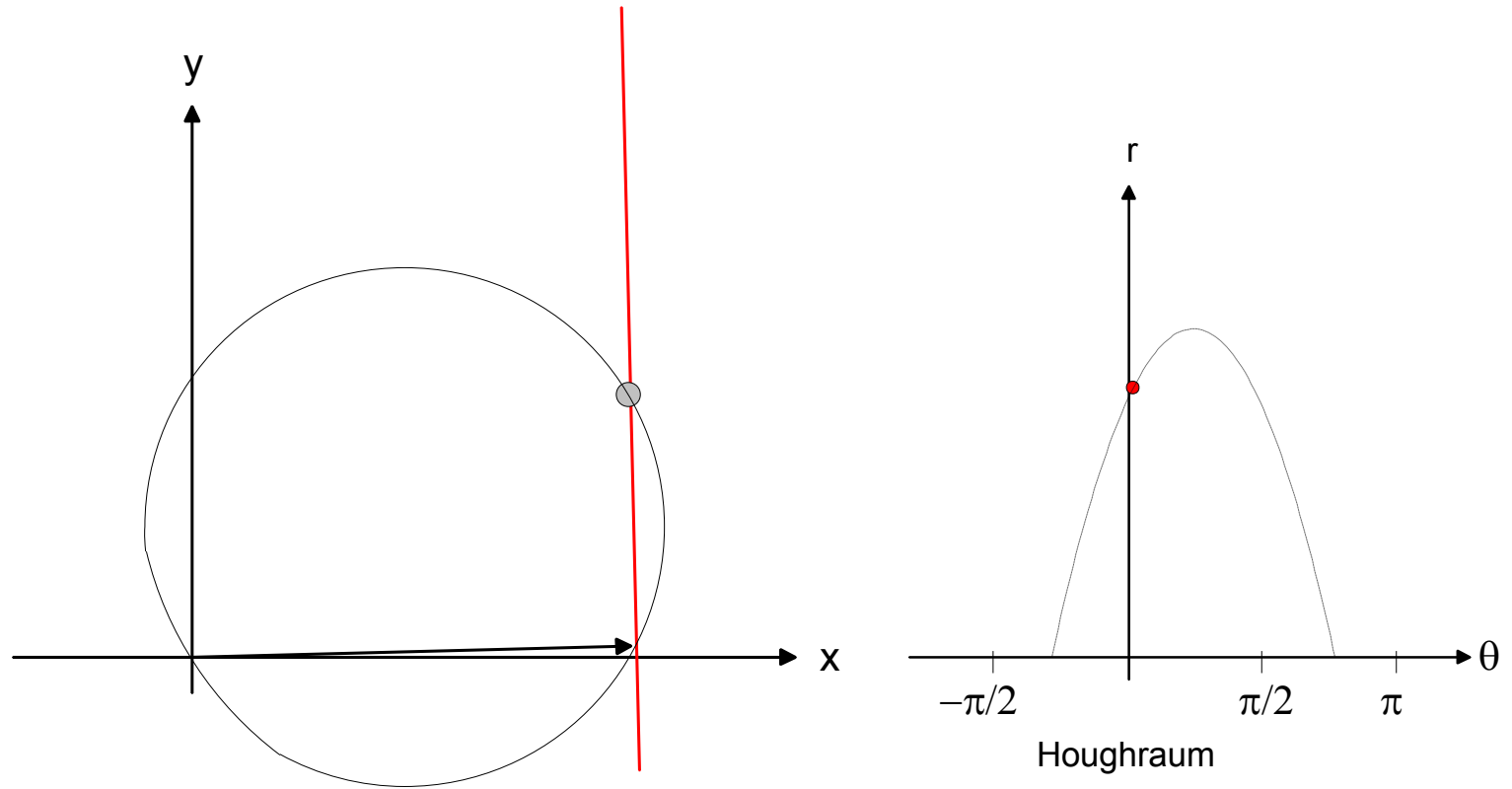


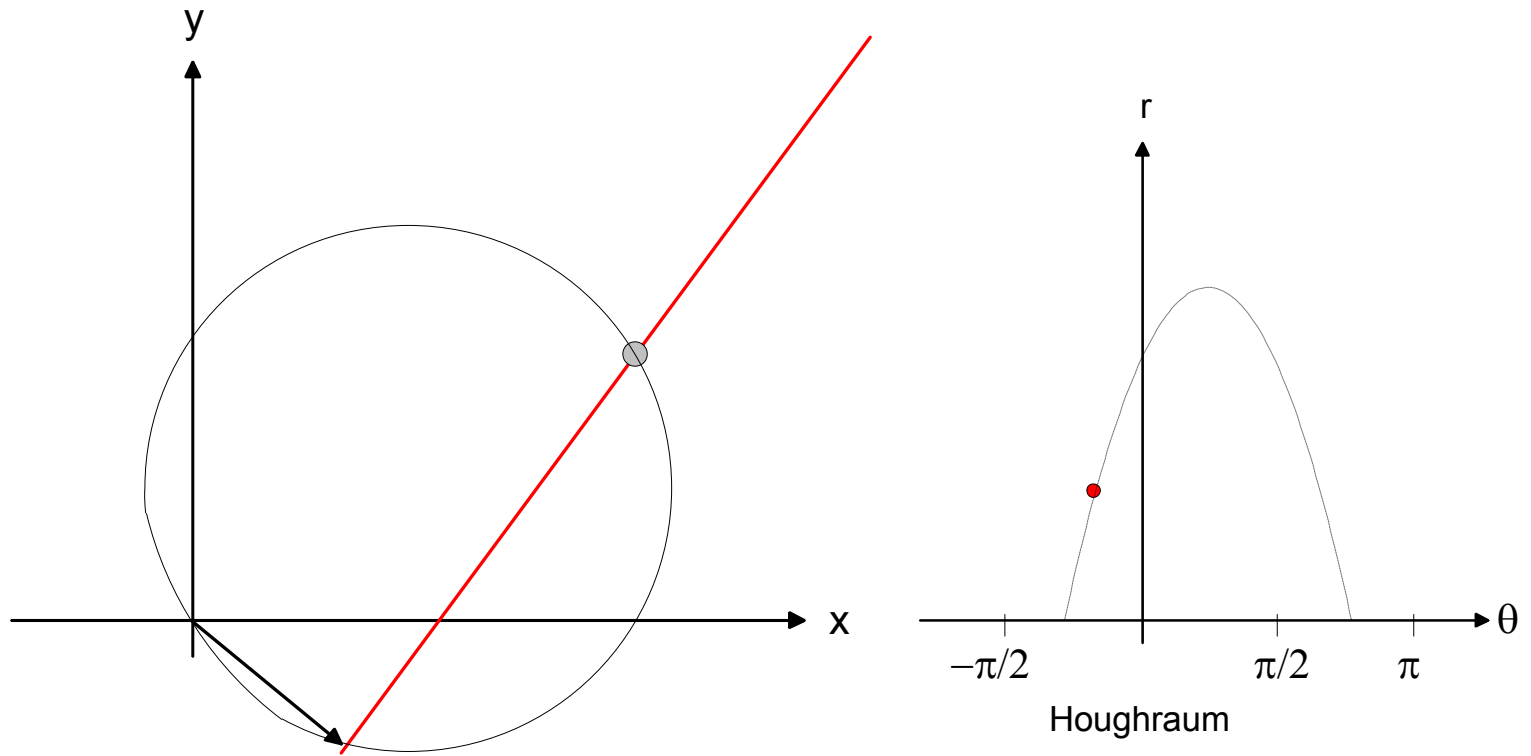


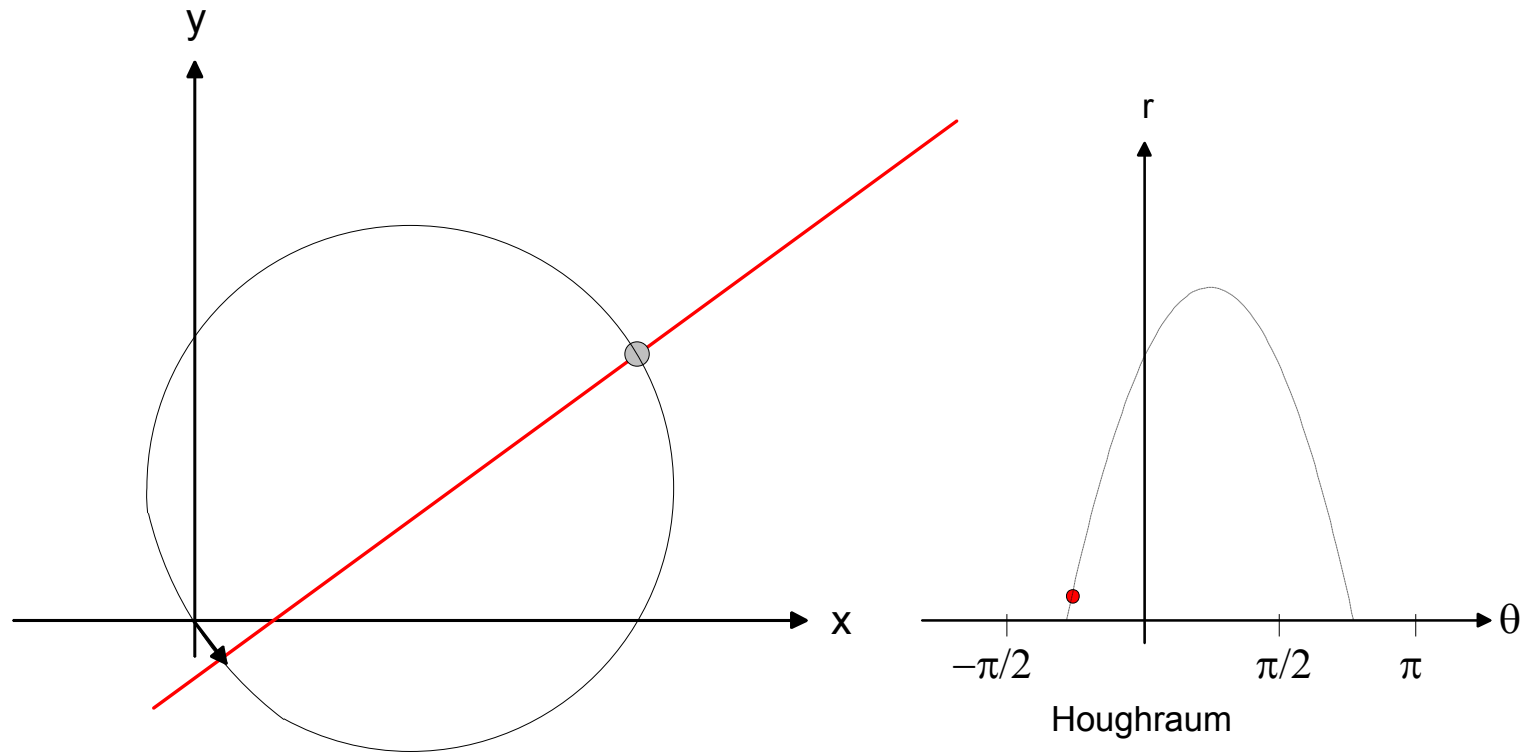














Zusammenfassung

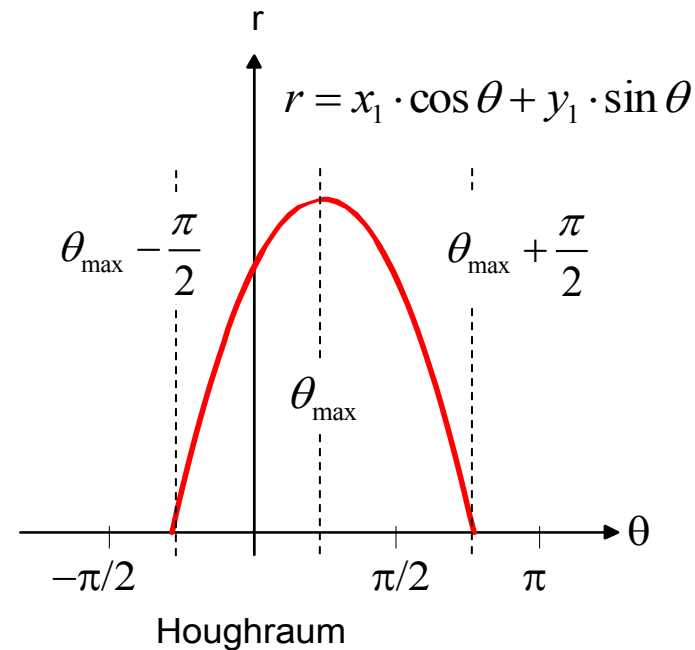
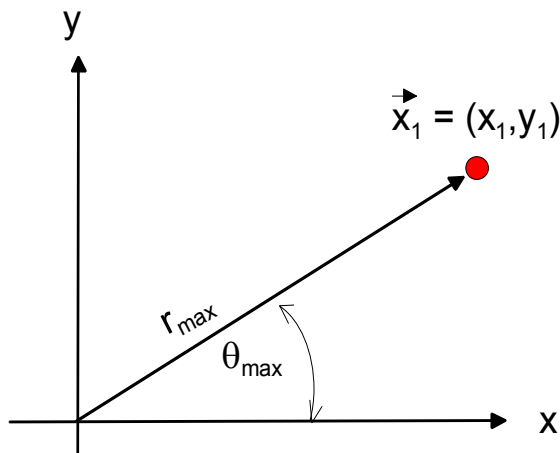
Die Menge aller durch einen Punkt (x_1, y_1) gehenden Geraden, wird beschrieben durch die Gleichung

$$r = x_1 \cdot \cos \theta + y_1 \cdot \sin \theta$$

mit

$$\theta \in [\theta_{\max} - \pi/2, \theta_{\max} + \pi/2]$$

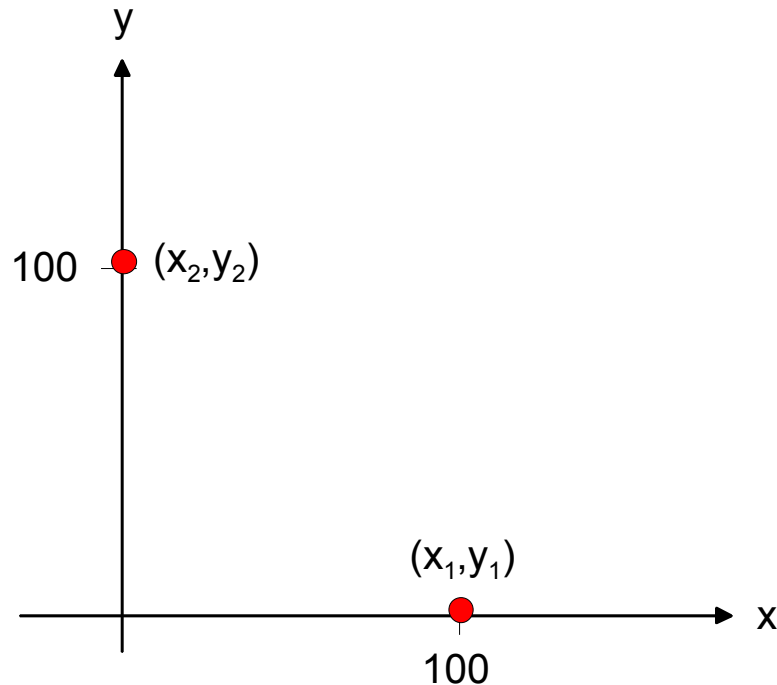
$$\theta_{\max} = \arctan(y_1 / x_1)$$





ÜBUNG: Houghtransformation

Zeichnen Sie den Parameterraum (Houghraum) und tragen Sie für die beiden gegebenen Punkte die Menge aller durch diese Punkte möglichen Geraden ein:

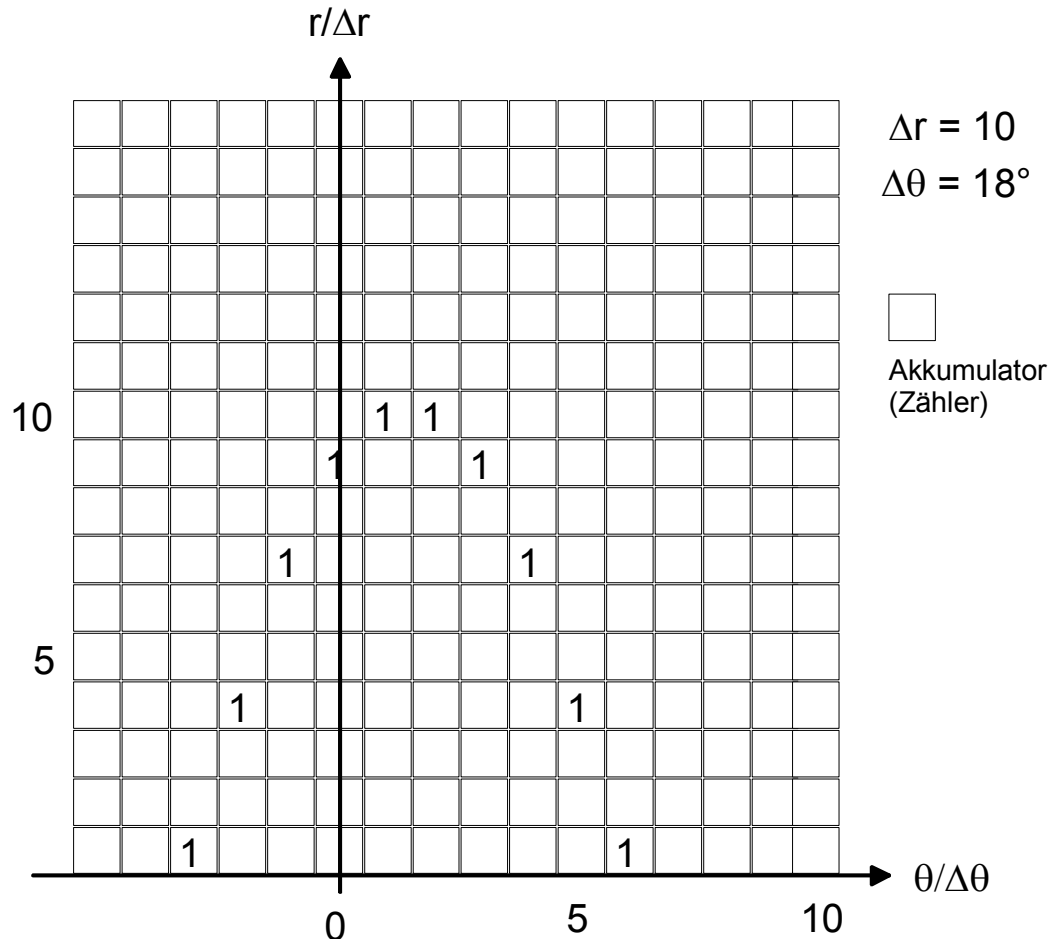




Algorithmische Umsetzung der Houghtransformation (1)

Problem: Ein kontinuierlicher Houghraum (R) ist nicht realisierbar.

Lösung: Diskretisierung des Houghraumes (= Zählerarray).



Anmerkung:

Typ. Diskretisierungen:

$$\Delta r = 1$$

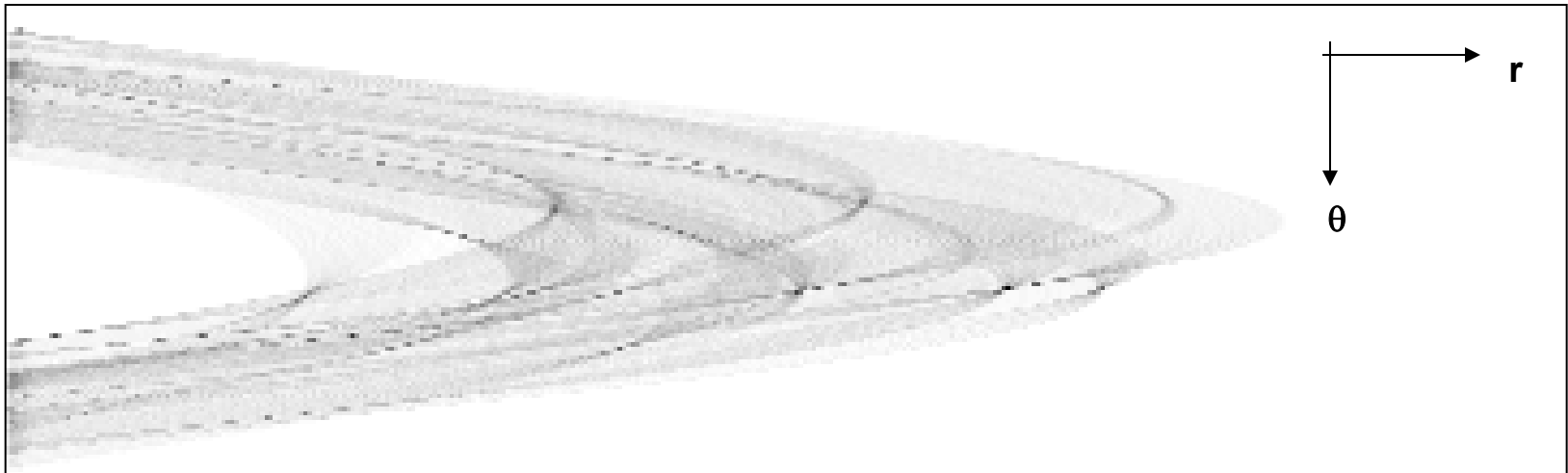
$$\Delta q = 2.5^\circ$$



Algorithmische Umsetzung der Houghtransformation (2)

Algorithmus "Houghtransformation"

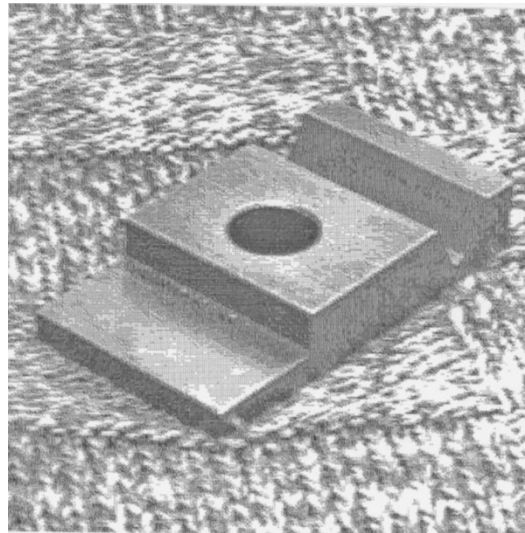
```
FOR j=1, n DO          /* n : Anzahl der gesetzten Bildpunkte */  
     $\theta_{\max} = \arctan(y_j/x_j)$   
    FOR  $\theta = \theta_{\max} - \pi/2 \dots \theta_{\max} + \pi/2$  STEP  $\Delta\theta$  DO  
         $r = \text{round}(x_j \cdot \cos(\theta) + y_j \cdot \sin(\theta))$   
         $A(r, \theta) = A(r, \theta) + 1$   
    ENDFOR  
ENDFOR
```



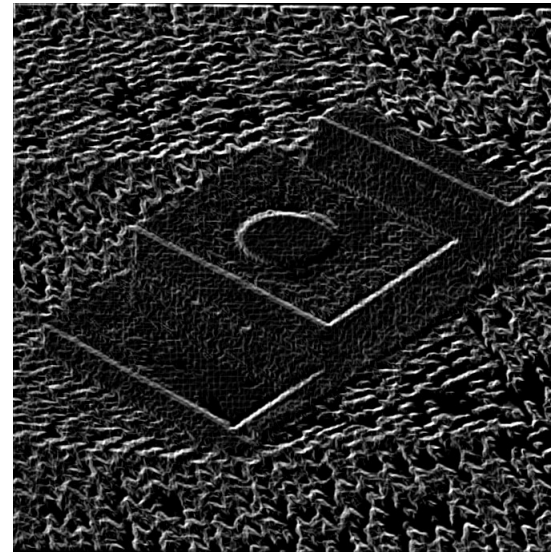
Eigenschaften der Houghtransformation

Problem: Es werden keine Geraden gesucht, sondern Bildstrukturen, die auf einer Gerade liegen. Dieser Sachverhalt kann auch vorteilhaft sein, z.B. im Fall teilweise verdeckter Kanten.

Problematisches Bild, da nahezu auf allen möglichen Geraden nennenswerte Grauwertsummen (im kantengefilterten Bild) zustande kommen werden.



Sobel
→

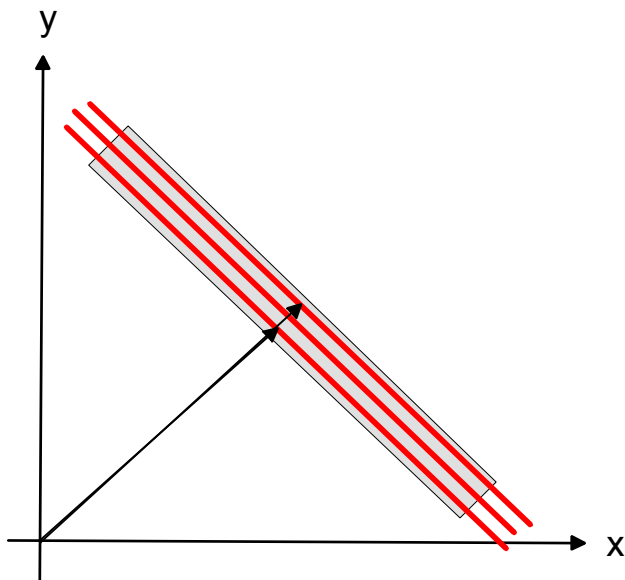


Lösung: Bildrauschen sollte durch eine Bildglättung unterdrückt werden. Durch eine geeignete Vorfilterung, sollten möglichst alle nicht-linienförmigen Bildstrukturen unterdrückt werden.

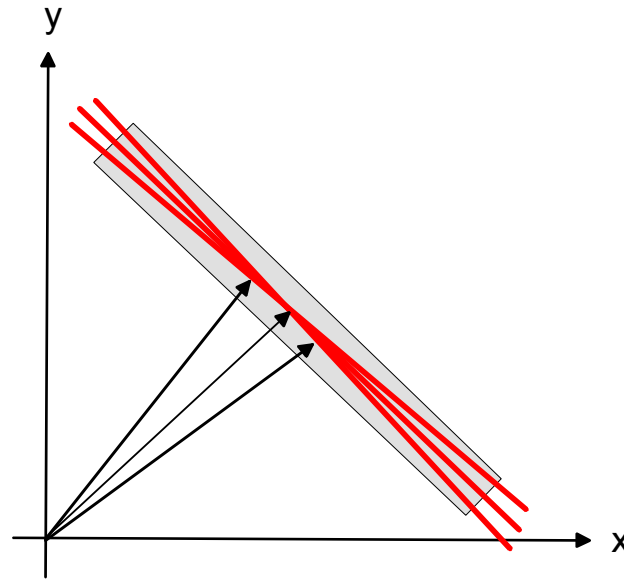


Eigenschaften der Houghtransformation

Problem: Breite Kanten führen zu mehrdeutigen Maxima.



Geraden mit unterschiedlichem r passen in die Bildkante (grau).



Geraden mit unterschiedlichem θ passen in die Bildkante (grau).

Verbesserung:

Eine zur Vorverarbeitung passende Quantisierungsstufe wählen.

5	2	0	0	0	0
10	11	0	0	0	0
14	17	20	3	0	0
13	20	25	20	6	0
12	11	18	24	13	5
6	2	0	15	18	10
2	0	0	1	16	11
0	0	0	0	8	12
0	0	0	0	2	12

Houghraum im Bereich des Maximums



Verbesserungen des klassischen Ansatzes

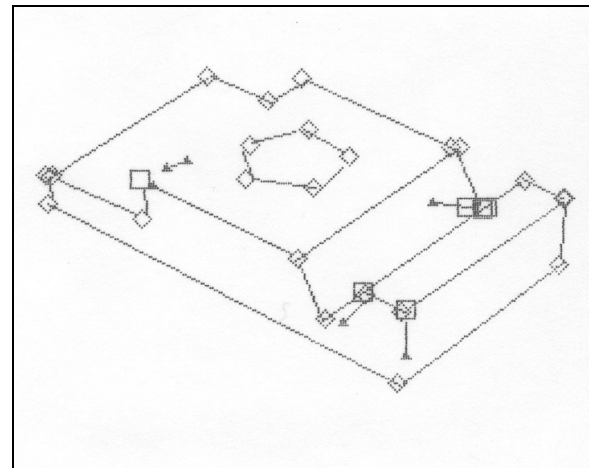
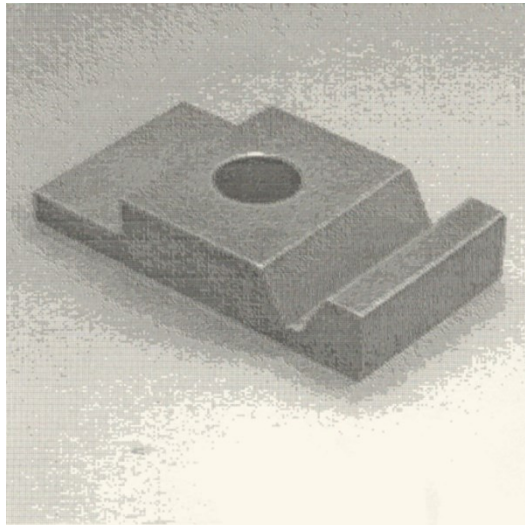
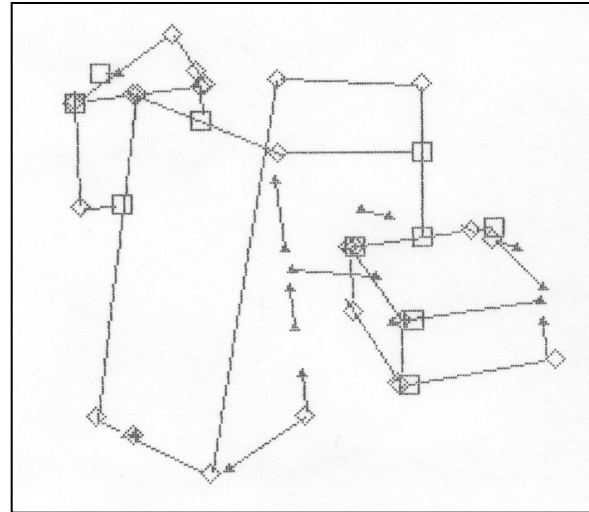
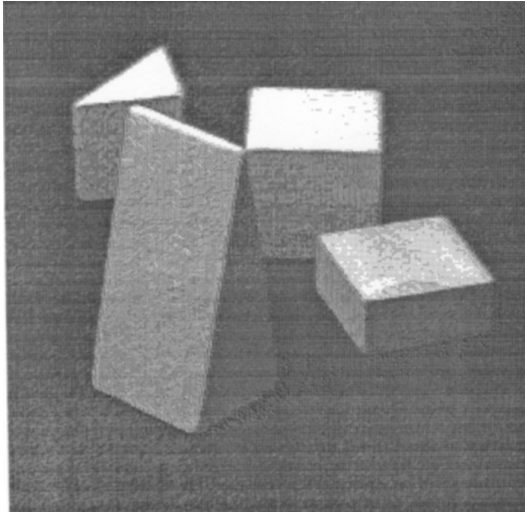
→ Grauwert-Houghtransformation

Eine Verbesserung erreicht man dadurch, dass die Hough-Akkumulatoren für gesetzte Bildpunkte nicht einfach nur hochgezählt werden, sondern dass stattdessen die Grauwerte der kantengefilterten Bildpunkte aufsummiert werden.

→ "Inverse Houghtransformation"

1. WHILE (nennenswerte Maxima im Houghraum)
2. Es wird das absolute Maximum im Houghraum gesucht.
3. Die zu diesem Maximum gehörenden Bildpunkte werden aus dem Bild und aus dem Houghakkumulator entfernt.
4. END WHILE

Beispiele: Houghtransformation





4.5 Weitergehende Verfahren

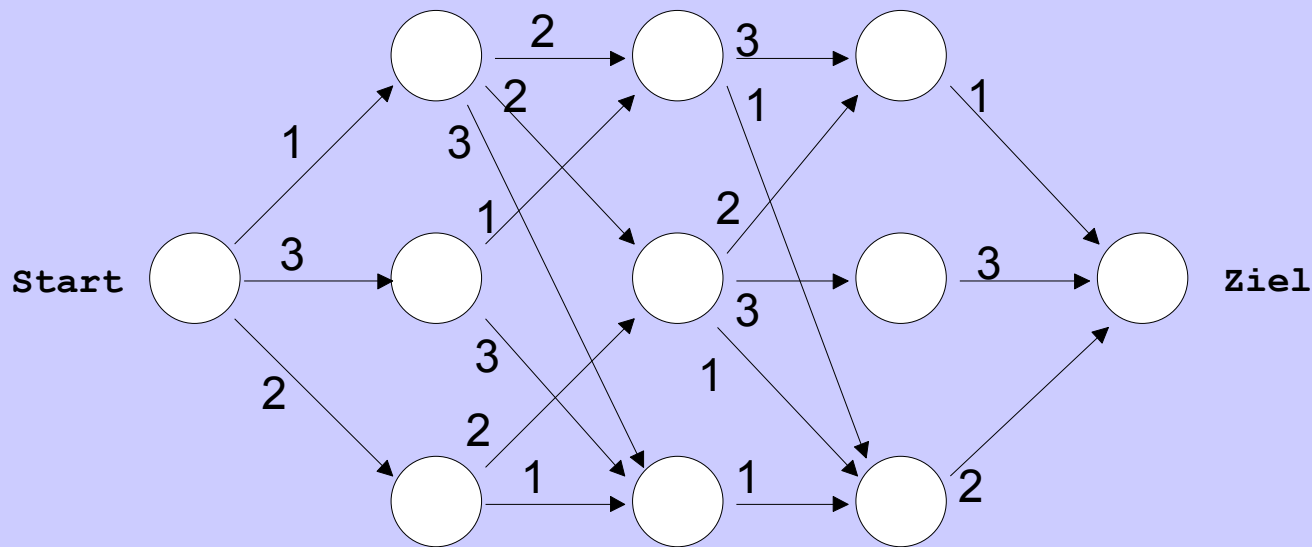
4.5.1 Dynamische Programmierung

4.5.1.1 Aufgabenstellung

Gegeben ist ein Graph, dessen Kanten gewichtet sind.

Auf welchem Weg ist die Summe der Gewichte am kleinsten (oder größten)?

Welcher Weg ist der beste ?



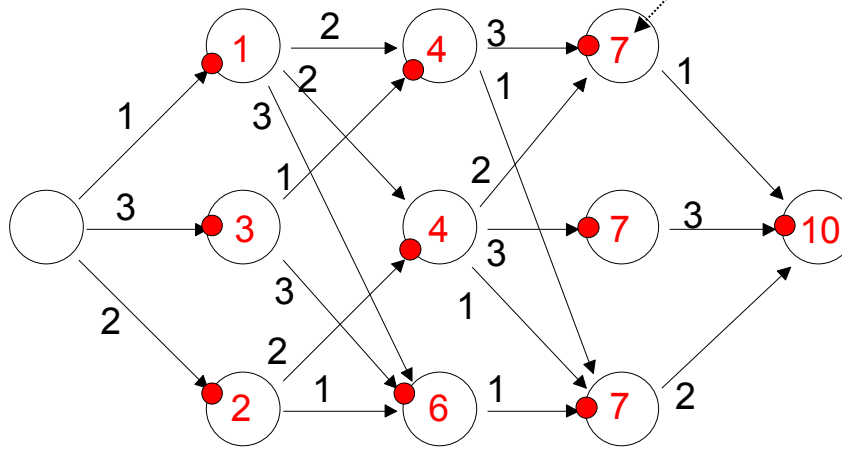


4.5.1.2 Lösungsprinzip

Anmerkung:

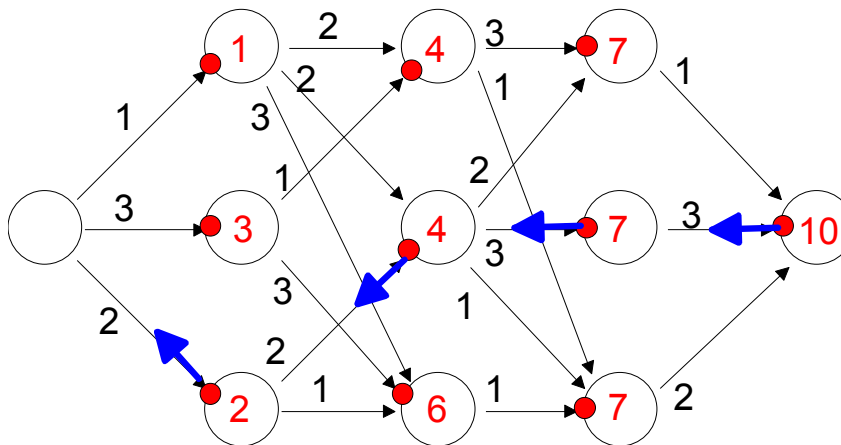
Ziel ist eine max.
Gewichstsumme

Knotenwert



Schritt 1:

1. Gestartet wird beim linken Knoten.
2. Bei jedem Nachfolgeknoten wird gespeichert:
 - der Weg zum besten Vorgänger ●
 - die Summe aus dem Knotenwert des besten Vorgängers und dem Verbindungsknoten.



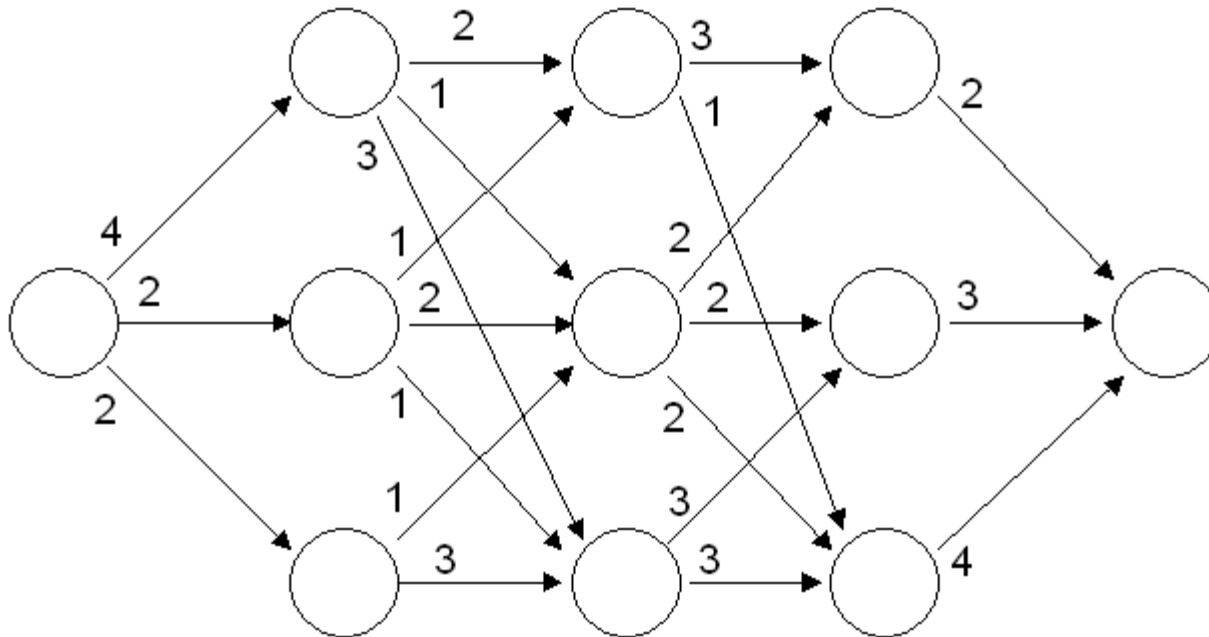
Schritt 2: (Backtracking)

1. Gestartet wird beim Endknoten.
2. Von Knoten zu Knoten folgt man der vorgeschlagenen Richtung.



ÜBUNG: Dynamische Programmierung 1

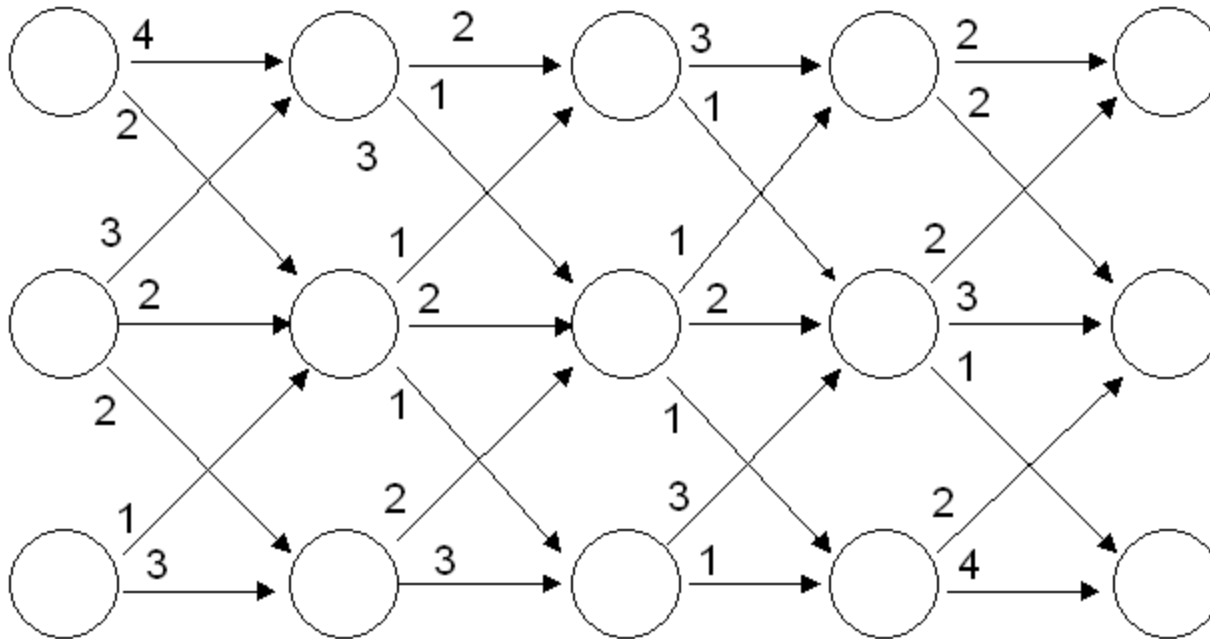
Gegeben ist folgender Graph. Finden Sie mit Hilfe der dyn. Programmierung den Weg mit der minimalen Gewichtsumme.





ÜBUNG: Dynamische Programmierung 2

Gegeben ist folgender Graph. Finden Sie mit Hilfe der dyn. Programmierung den Weg von links nach rechts mit der minimalen Gewichtsumme.

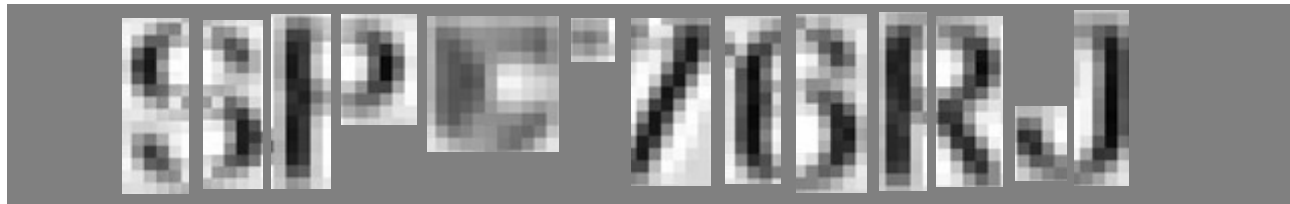
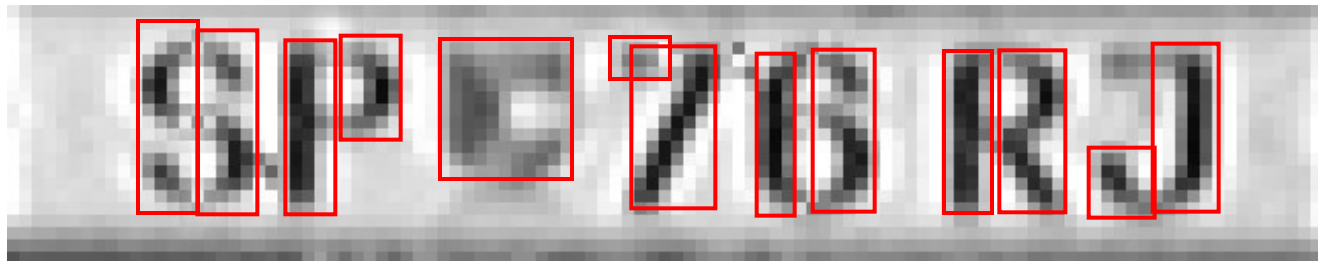




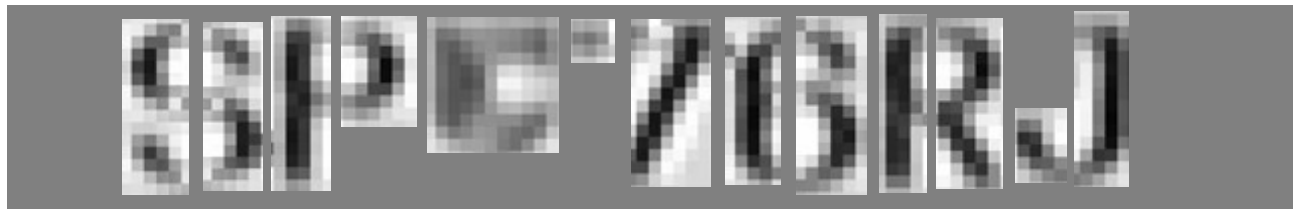
4.5.1.3 Anwendungsbeispiel : Lesen von Autokennzeichen



überkritische Segmentierung



1 2 3 4 5 6 7 8 9 A B C D



1 2 3 4 5 6 7 8 9 A B C D

