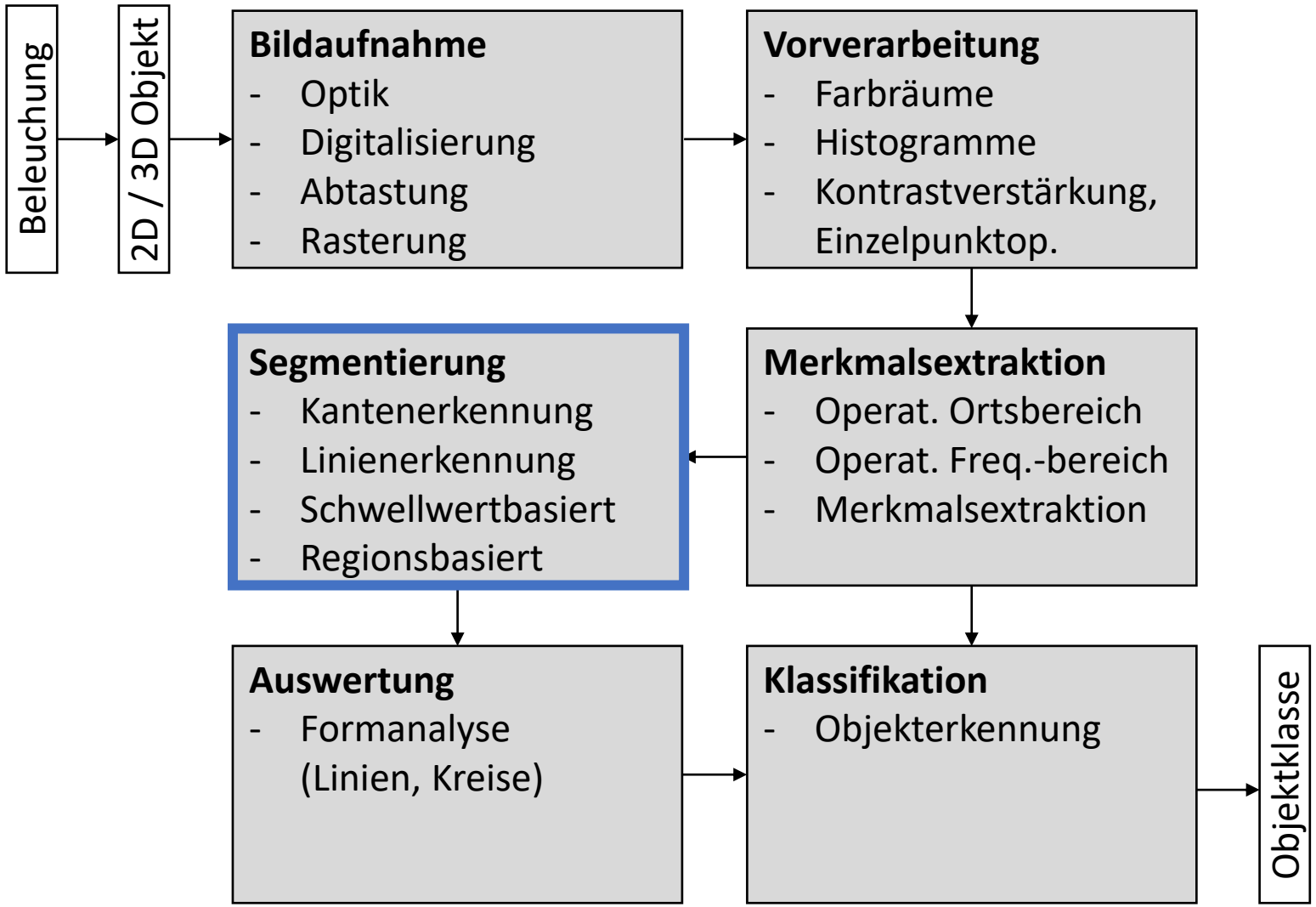
Digitale Bildverarbeitung

DHBW Stuttgart

Übersicht – Struktur der Vorlesung



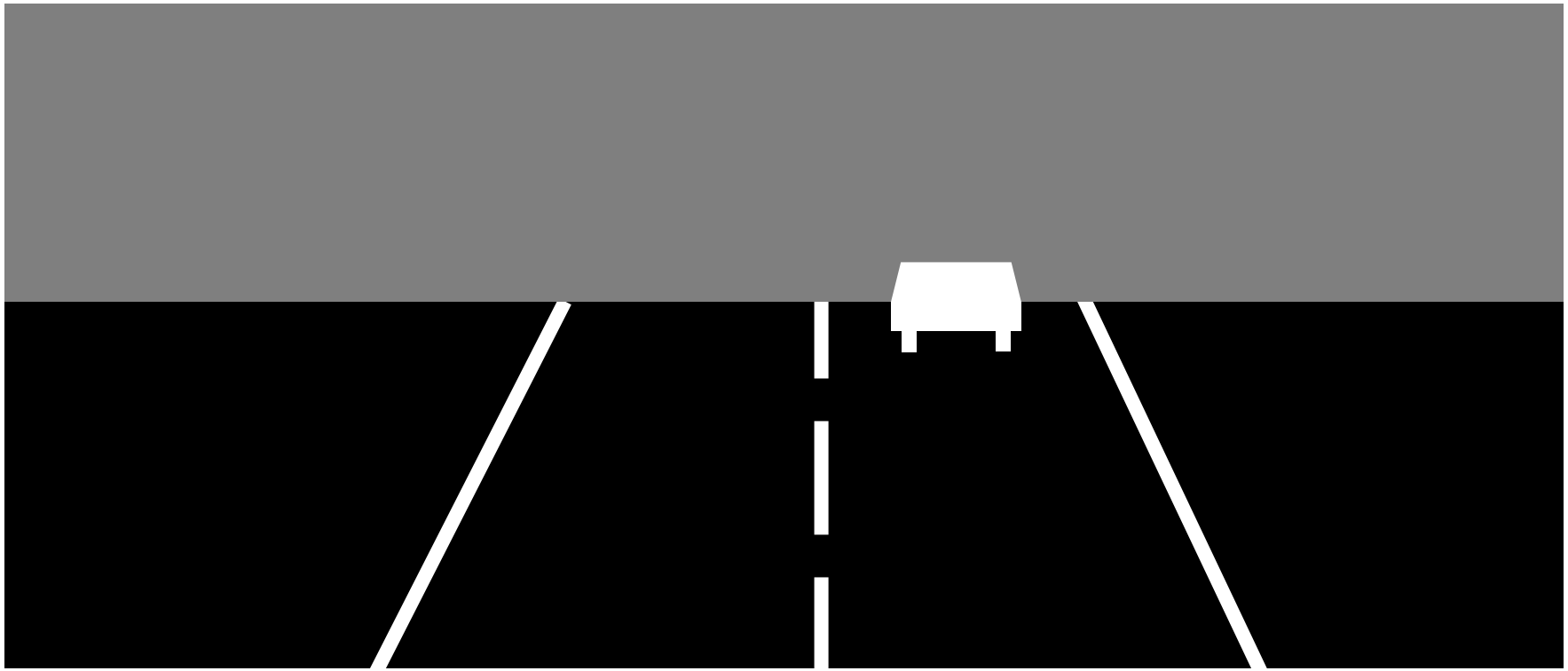
Quelle: Haasdonk, Digitale Bildverarbeitung, Uni Freiburg

Lernziele Segmentierung

- Motivation
- Verfahren zur Kantenerkennung (Gradient, Rang, Canny)
- Schwellwertbasierte Segmentierung (global, adaptiv)
- Regionsbasierte Segmentierung (wachsen, teilen und zusammenführen, Watershed)

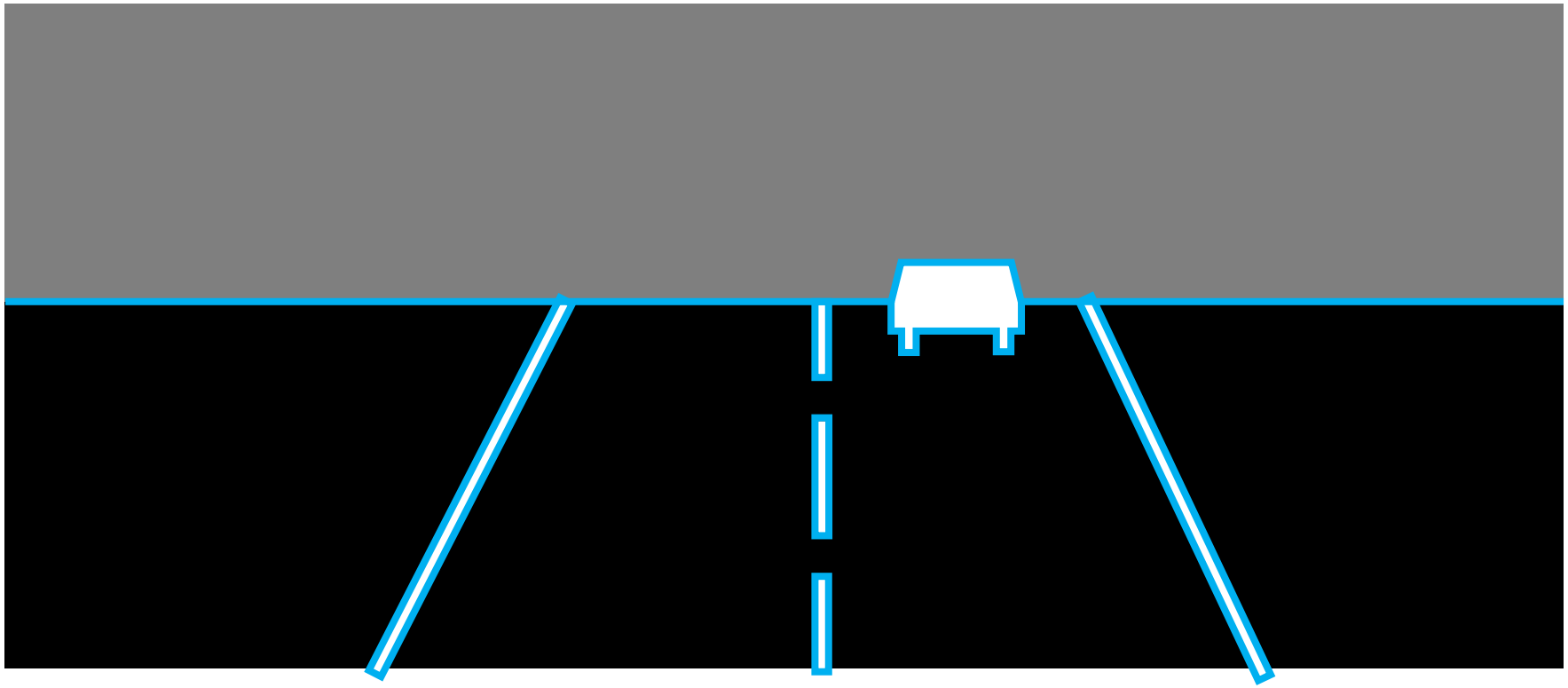
Motivation

- Punkt- / Linien- / Kanten- / …-Erkennung Grundlage für Segmentierung



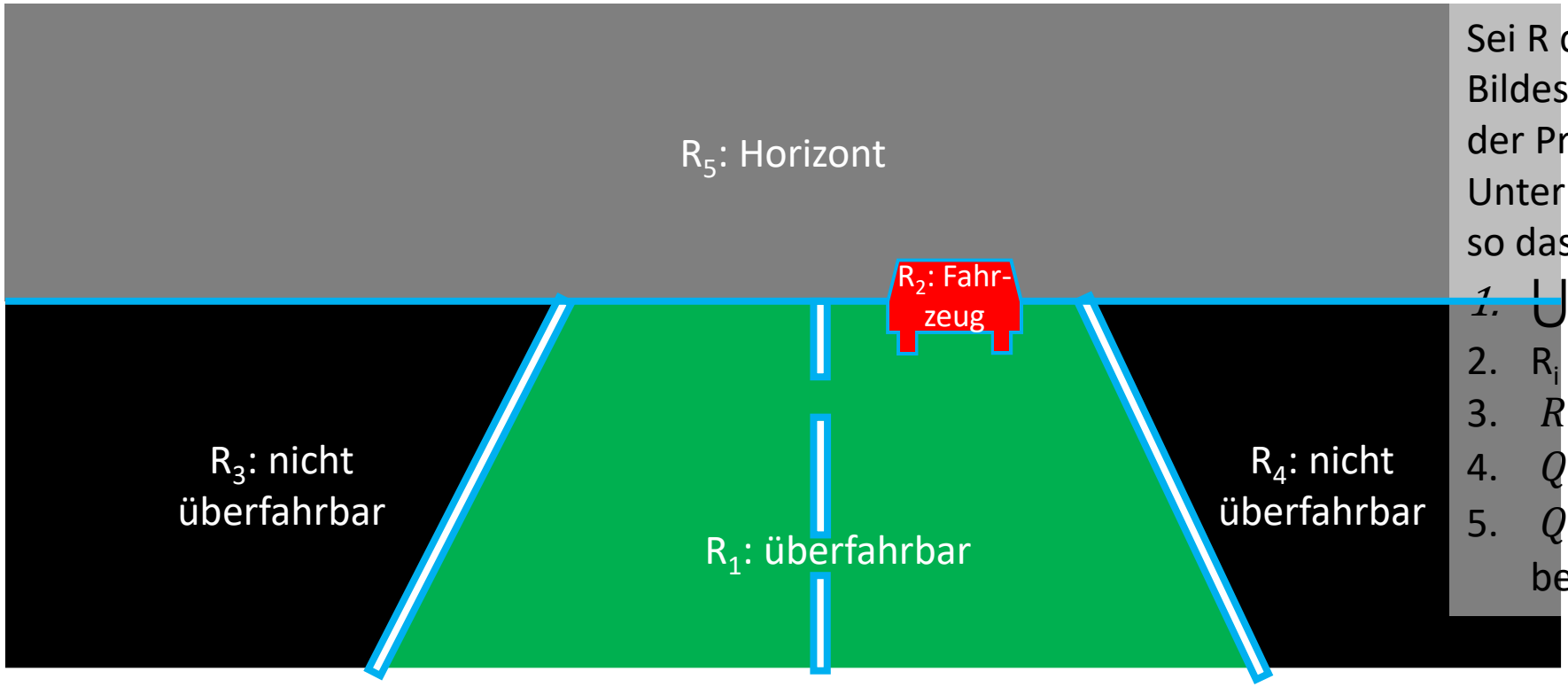
Motivation

- Punkt- / Linien- / Kanten- / …-Erkennung Grundlage für Segmentierung



Motivation

- Punkt- / Linien- / Kanten- / …-Erkennung Grundlage für Segmentierung

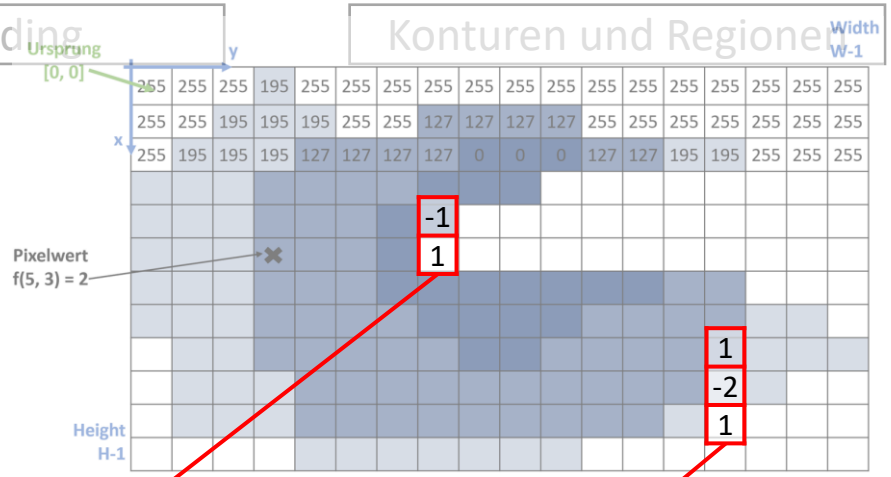


Sei R die gesamte Region eines Bildes. Bildsegmentierung ist dann der Prozess, der R in n Unterregionen R_1, R_2, …, R_n einteilt, so dass …

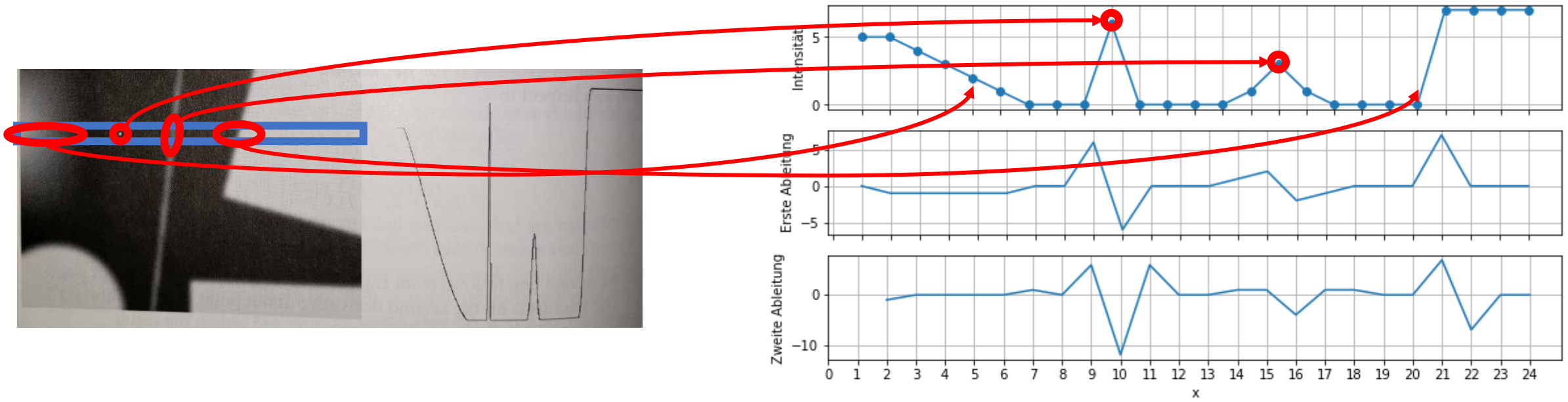
1. $\bigcup_{i=1}^{n} R_i = R$
2. R_i ist eine verbundene Menge
3. $R_i \cap R_j = \emptyset$ für i ≠ j
4. $Q(R_i) = TRUE$
5. $Q(R_i \cup R_j) = FALSE$ für benachbarte Regionen i und j

$Q(R_i) = TRUE$: Pixel in Region R_i erfüllen selbe Eigenschaft (z.B. Intensität) → logisches Prädikat

Hintergrund, Arten von Kanten

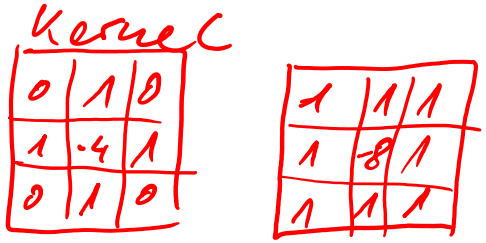


- Grundlagen
  - Ableitung erster Ordnung $\frac{\partial f}{\partial x} = f(x+1) - f(x)$
  - Ableitung zweiter Ordnung $\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$



w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Detektion isolierter Punkte



- Definiere die Antwort der Maske mit R (Response):

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_9 z_9 = \sum_{k=1}^{9} w_k z_k$$

1. Laplacian zur Erkennung von Punkten

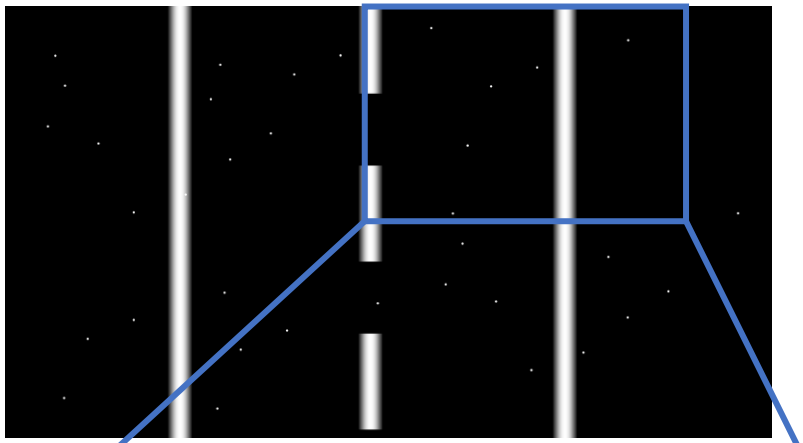
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

-1	-1	-1
-1	8	-1
-1	-1	-1

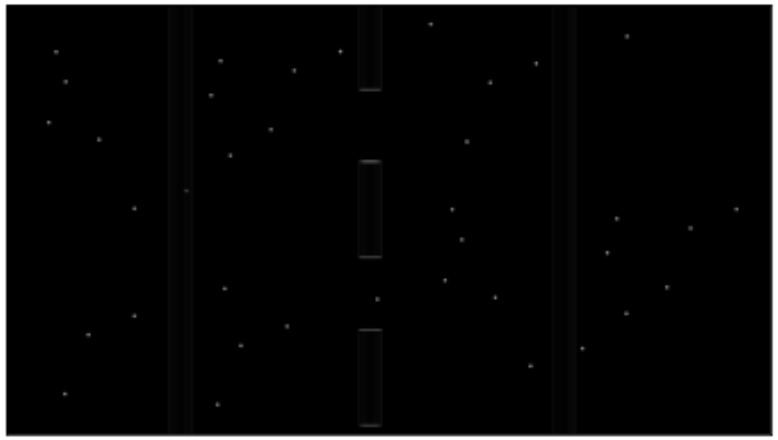
2. Anwendung eines Schwellwerts

$$g(x, y) = \begin{cases} 1, \text{ if } |R(x, y)| \geq T \\ 0, \text{andernfalls} \end{cases}$$

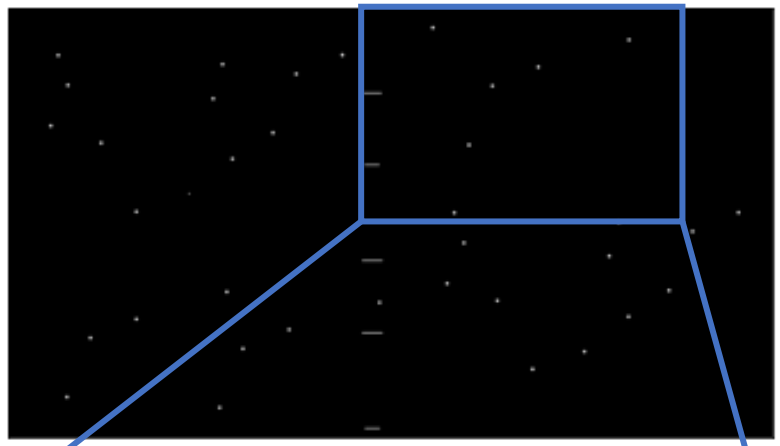
Detektion isolierter Punkte: Beispiel



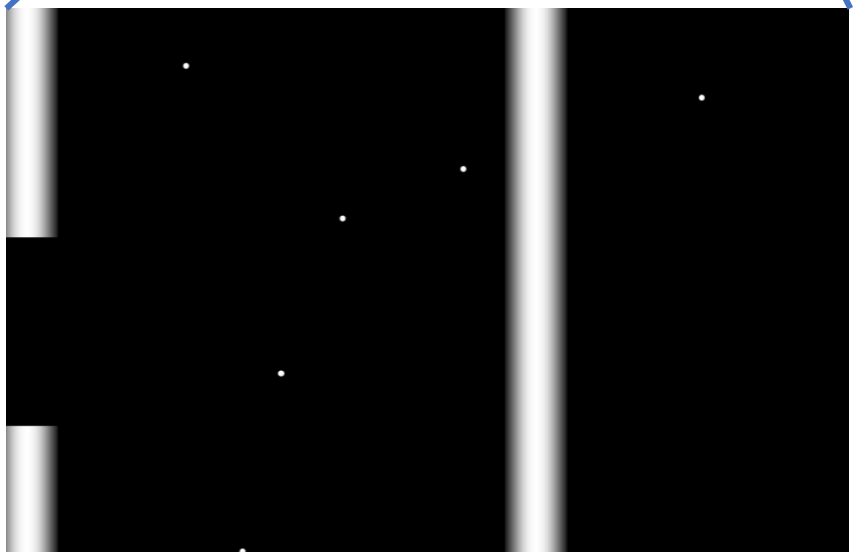
Original

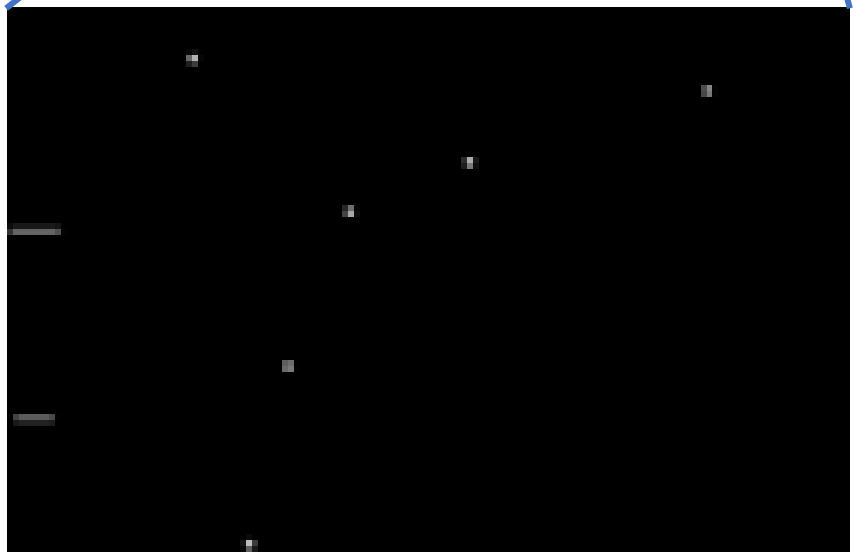


Laplacian

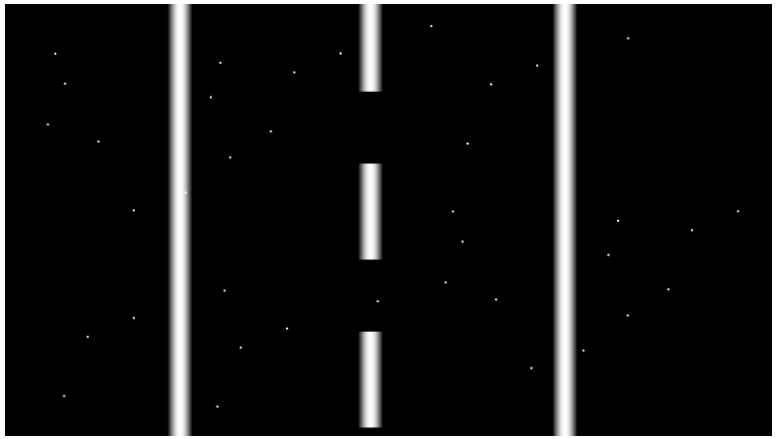


Schwellwert + Dilatation





Detektion isolierter Punkte: Beispiel



Original

–

Schwellwert + Dilatation

=

Detektion von Linien

- Masken zur Liniendetektion

-1	-1	-1
2	2	2
-1	-1	-1

Horizontal

2	-1	-1
-1	2	-1
-1	-1	2

Diag. (45°)

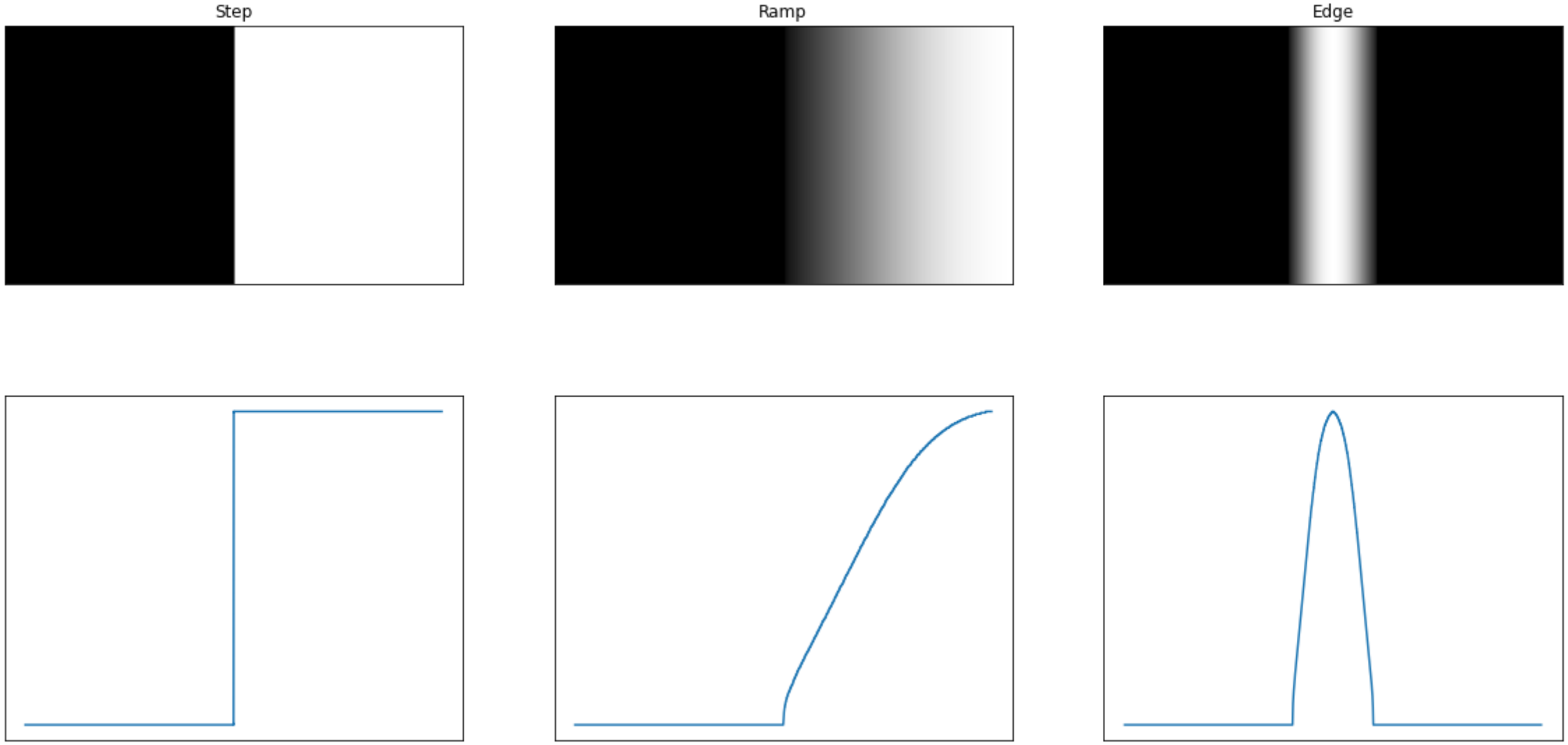
-1	2	-1
-1	2	-1
-1	2	-1

Vertikal

-1	-1	2
-1	2	-1
2	-1	-1

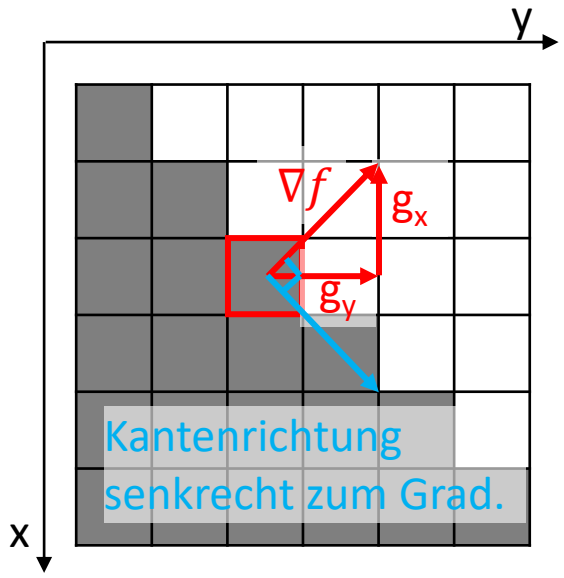
Diag. (-45°)

Detektion von Kanten



Detektion von Kanten mithilfe des Gradienten

- Gradient $$\nabla f = grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix} = \begin{bmatrix} f(x+1,y) - f(x,y) \\ f(x,y+1) - f(x,y) \end{bmatrix}$$
- Betrag $$\mathrm{M}(x,y) = \mathrm{mag}(\nabla f) = \sqrt{{g_x}^2 + {g_y}^2} \approx |g_x| + |g_y|$$
- Winkel $$\alpha(x,y) = tan^{-1} \begin{bmatrix} g_x \\ g_y \end{bmatrix}$$

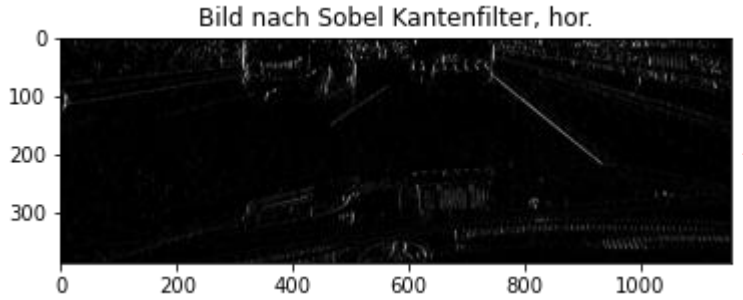


Detektion von Kanten mithilfe des Gradienten

*

-1	0	1
-2	0	2
-1	0	1

=



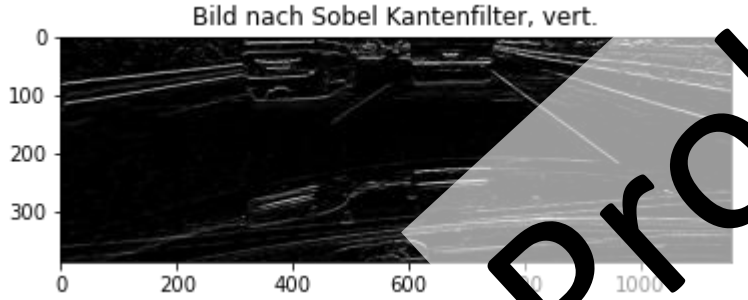
g_x

$$M(x,y) = \text{mag}(\nabla f) = \sqrt{{g_x}^2 + {g_y}^2}$$

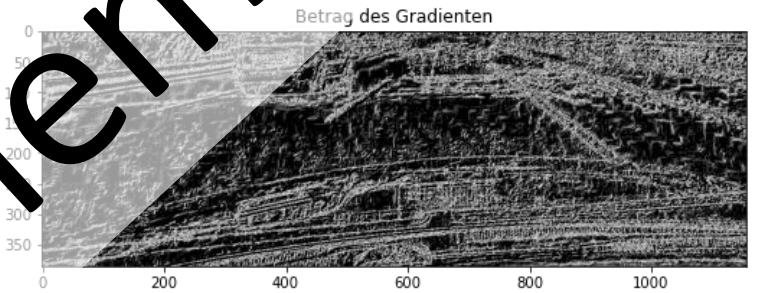
*

-1	-2	-1
0	0	0
-1	-2	-1

=



g_y



Probleme?

Eingangsbild | Filter | Hor. + vert. Kanten | Ergebnisbild

Detektion von Kanten mit Canny Edge Det.

1. **Rauschreduzierung**
   Kantenerkennung ist Rausch-anfällig, daher wird Rauschen mit 5x5-Gauß-Filter gefiltert
2. **Berechnung des Gradienten**

$$\mathrm{M}(\mathrm{x},\mathrm{y}) = \mathrm{mag}(\nabla f) = \sqrt{{g_x}^2 + {g_y}^2}; \alpha(x,y) = tan^{-1}\begin{bmatrix} g_x \\ g_y \end{bmatrix}$$

Winkel des Gradienten senkrecht zur Kante

3. **Non-Maximum Suppression**
   Größe und Winkel des Grad. wird in lok Nachbarschaft verglichen
   Mittlerer Wert relevant, wenn Größe > beide relevanten Nachbarn

131	162	232
104	~~93~~	139
243	~~26~~	252

α=90°

93 > 26 → ignoriere 26
93 < 162 → ignoriere 93

131	162	232
104	~~93~~	139
243	26	252

α=180°

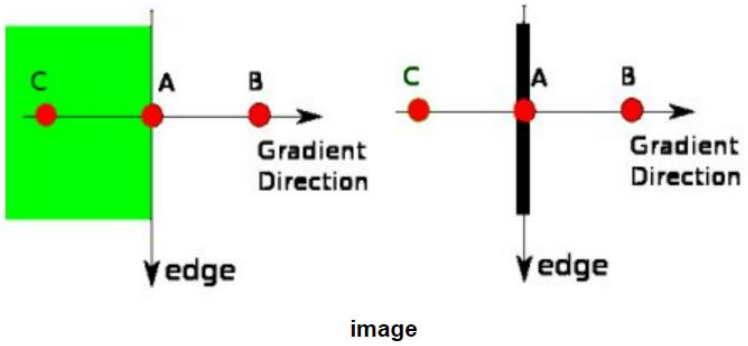
93 < 104 → ignoriere 93
93 < 139 → ignoriere 93

OpenCV Edge Detection (cv2.Canny) - PyImageSearch

Detektion von Kanten mit Canny Edge Det.

3. Non-Maximum Suppression (cont.)

Größe und Winkel des Grad. wird in lok Nachbarschaft verglichen
Mittlerer Wert relevant, wenn Größe > beide relevanten Nachbarn

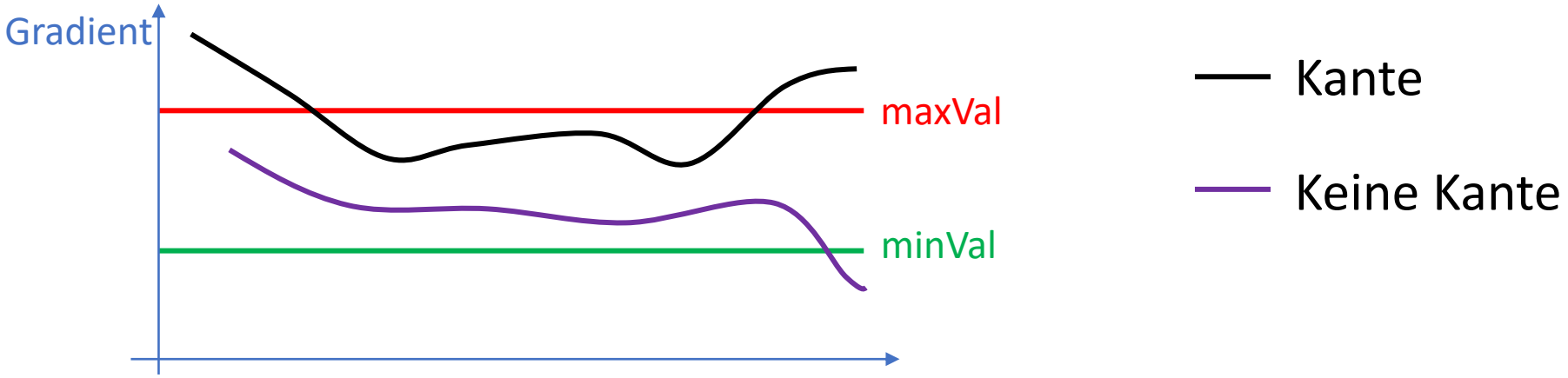


- Punkt A ist auf einer Kante in vertikaler Richtung des Bildes
- Richtung des Gradienten ist senkrecht („normal“) zur Kante
- Punkte B und C liegen in der Richtung des Gradienten
- Punkt A wird verglichen mit B und C
  → falls lokales Maximum, behalte ihn für nächsten Schritt
  → falls nicht, ignoriere ihn (weise ihm „0“ zu)

Detektion von Kanten mit Canny Edge Det.

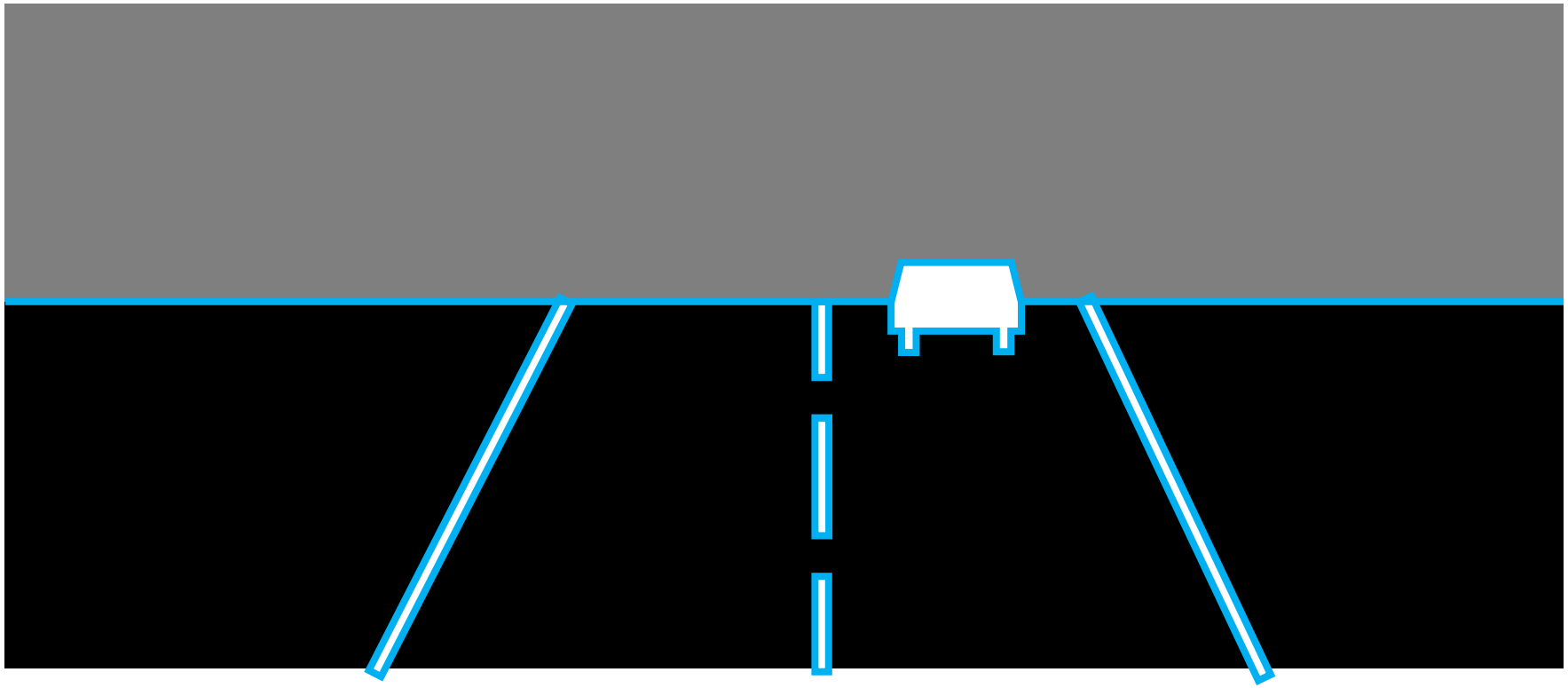
4. Schwellwert mit Hysterese

- definiere unteren und oberen Schwellwert minVal und maxVal
- jede Kante mit Gradienten > maxVal ist **„sichere Kante“**
- jede Kante mit Gradienten < minVal ist sicher **„keine Kante“**
- jede Kante mit minVal < Gradient < maxVal wird entsprechend ihrer Verbindung gewertet:
  - falls Kante verbunden mit „sicherer Kante“ → Teil der Kante
  - falls Kante verbunden mit „keiner Kante“ → kein Teil der Kante



Motivation

- Punkt- / Linien- / Kanten- / …-Erkennung Grundlage für Segmentierung



Schwellwertbasierte und Regionsbasierte Segm.

Schwellwertbasierte Segmentierung

- global definierter Schwellwert
- adaptiver Schwellwert

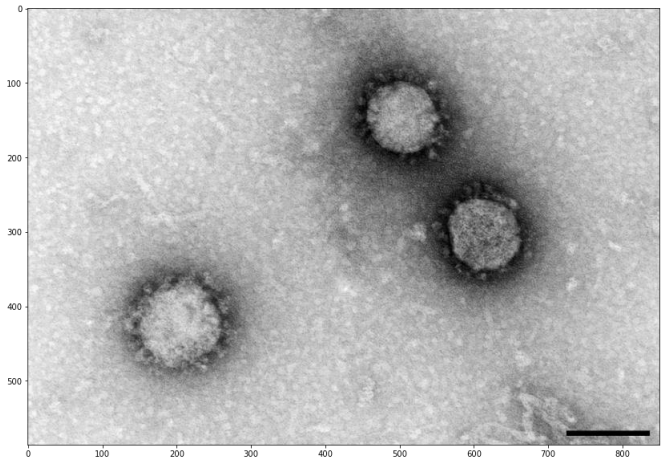
Regionsbasierte Segmentierung

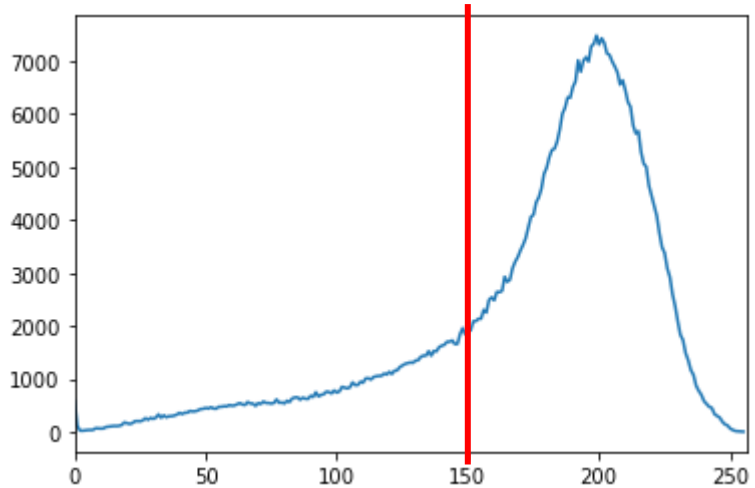
- Region Growing

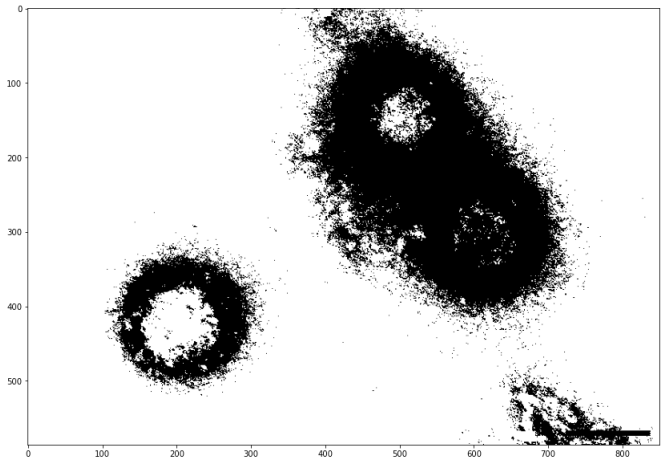
Schwellwertbasierte Segmentierung: global

- $g(x,y) = \begin{cases} 1, \text{ if } f(x,y) > T \\ 0, \text{andernfalls} \end{cases}$

Problem: einzelne Zellen können nicht getrennt werden.







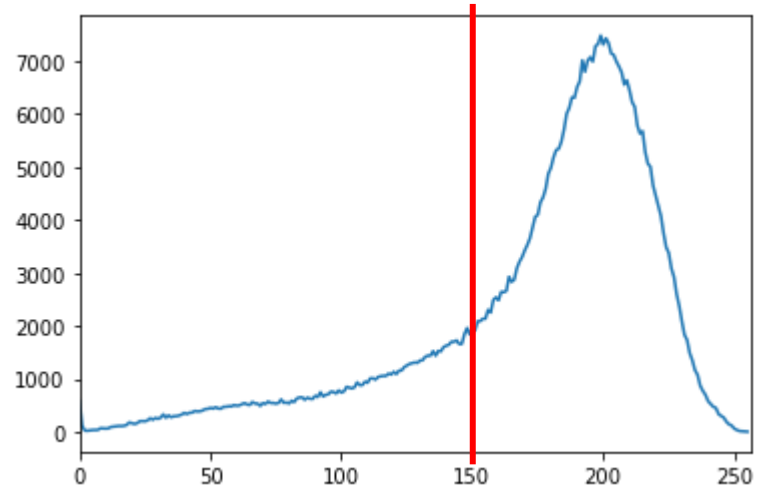
1. Vorverarbeitung (z.B. HSV)

2. Treshold definieren

3. Threshold anwenden

Schwellwertbasierte Segmentierung: global

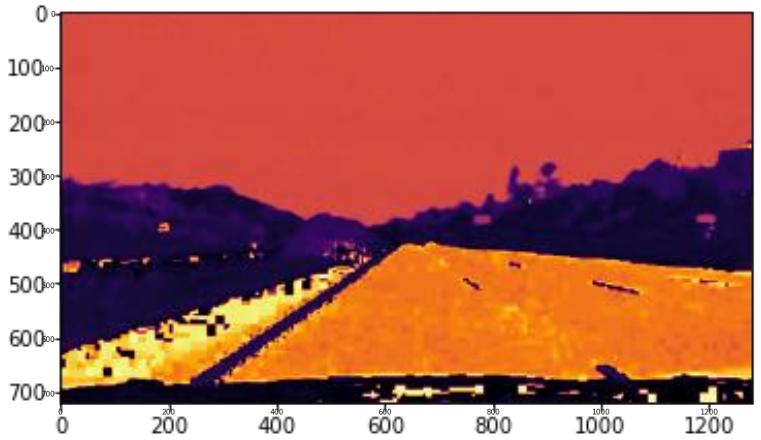
- $g(x,y) = \begin{cases} 1, \text{ if } f(x,y) > T \\ 0, \text{andernfalls} \end{cases}$

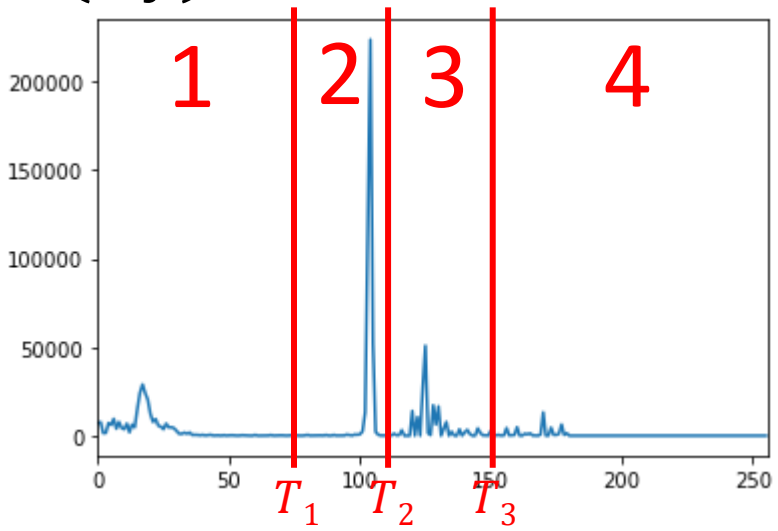


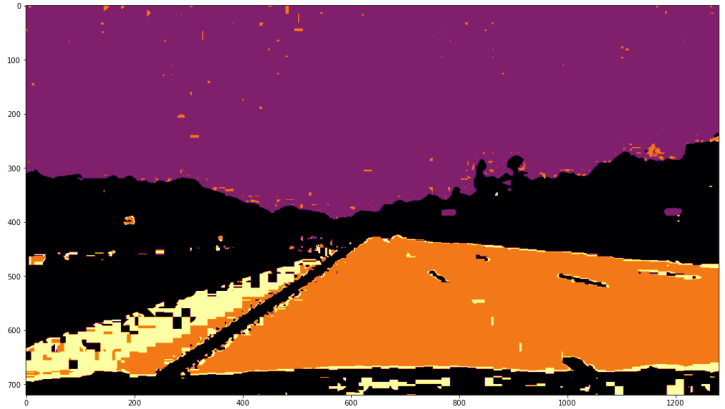
- Automatische Bestimmung des Schwellwerts:
  1. wähle initialen Schwellwert T (z.B. mittig = 128)
  2. teile Bild in zwei Segmente: R_1 besteht als allen Pixeln mit Wert > T, R_2 aus Pixeln $\leq$ T und berechne die Mittelwerte m_1 und m_2 der Intensitäten in den Bereichen
  3. berechne neuen Schwellwert: $T = {}^1/_2\,(m_1 + m_2)$
  4. wiederhole Schritte 2 – 4, bis Veränderung von T in zwei aufeinanderfolgenden Iterationen < ΔT

Schwellwertbasierte Segmentierung: global

- Mehrere Schwellwerte T_x
- $$g(x,y) = \begin{cases} 0, \text{ if } f(x,y) \leq T_1 \\ 0.25, if\ T_1 < f(x,y) \leq T_2 \\ 0.5, if\ T_2 < f(x,y) \leq T_3 \\ 1, \text{ if } f(x,y) > T3 \end{cases}$$





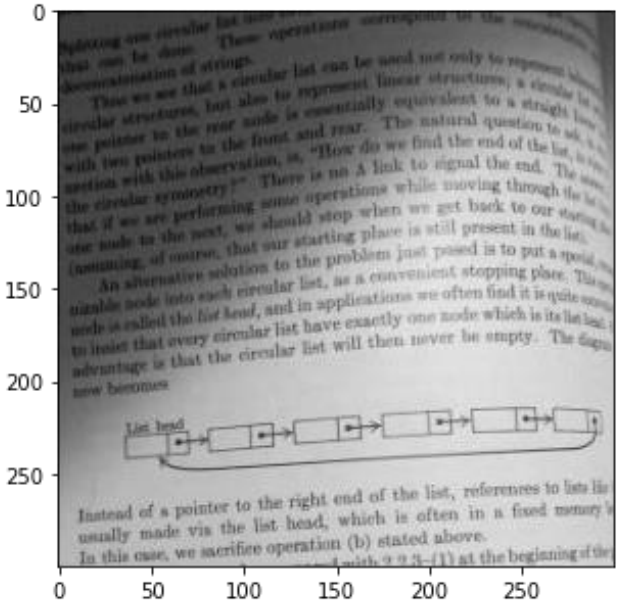


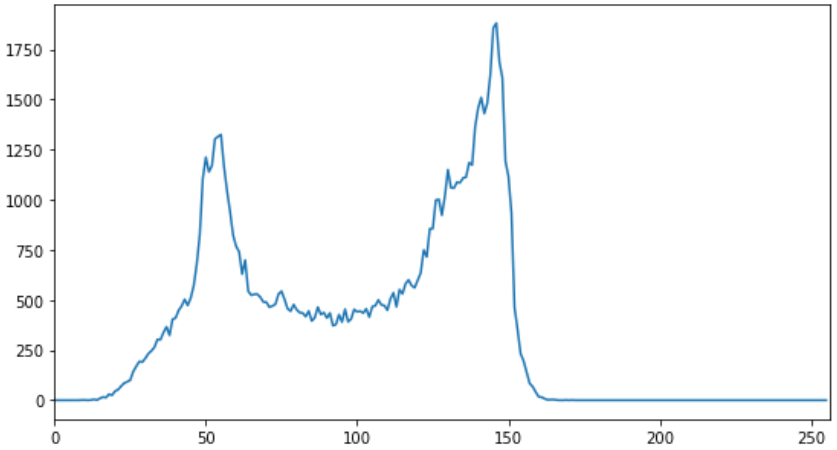
1. Vorverarbeitung (z.B. HSV)

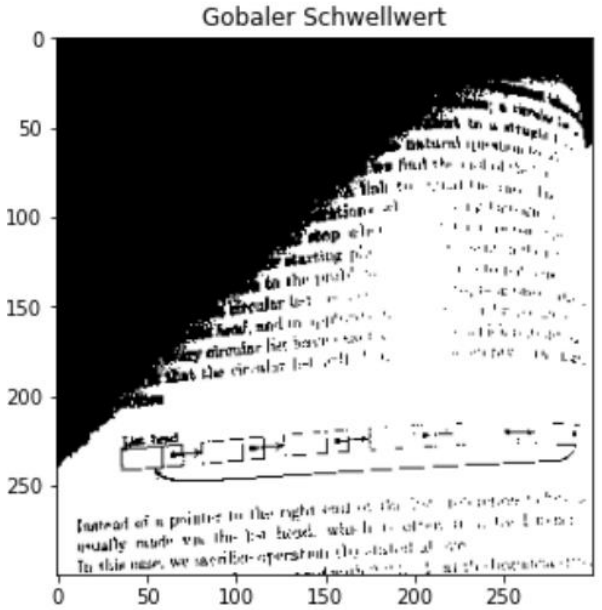
2. Treshold definieren

3. Threshold anwenden

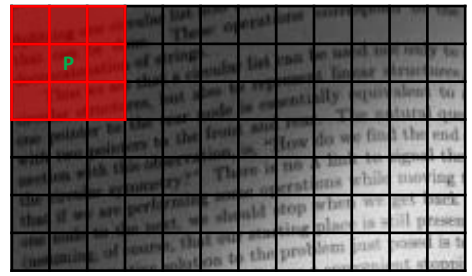
Schwellwertbasierte Segmentierung: global





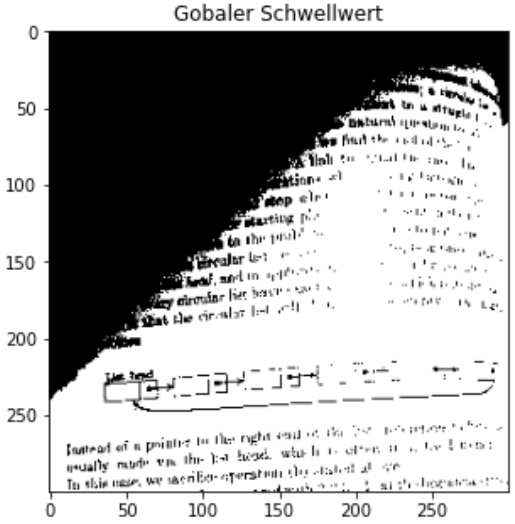


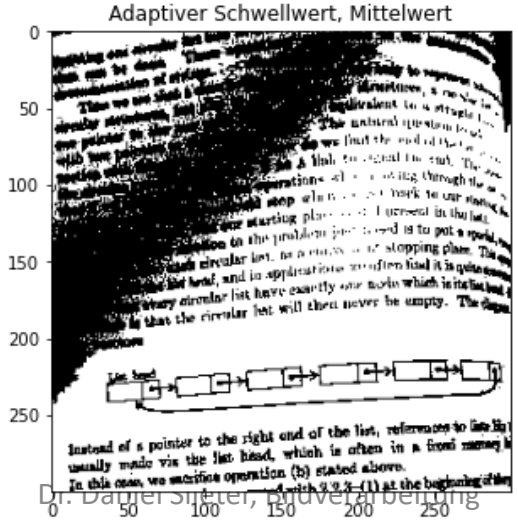
Schwellwertbasierte Segmentierung: adaptiv

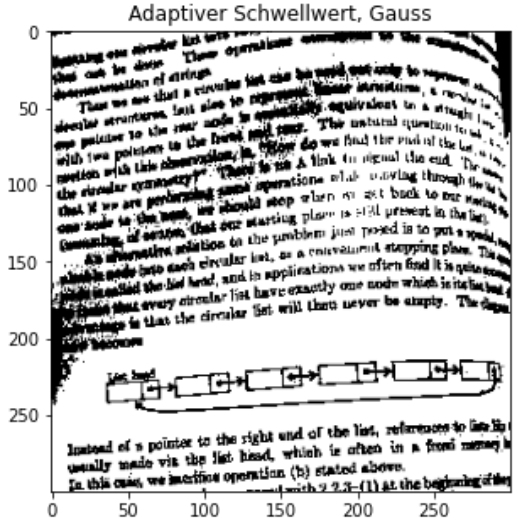


- Annahme: Bild ist nicht gleichmäßig belichtet

1. definiere zu betrachtende Nachbarschaft (3, 5, 7, …) eines Pixels
2. berechne für jeden Pixel den Schwellwert abhängig von der Nachbarschaft
   1. Mittelwert der angegebenen Nachbarschaft
   2. mit Gaußscher Verteilung gewichtete Summe der Nachbarschaft



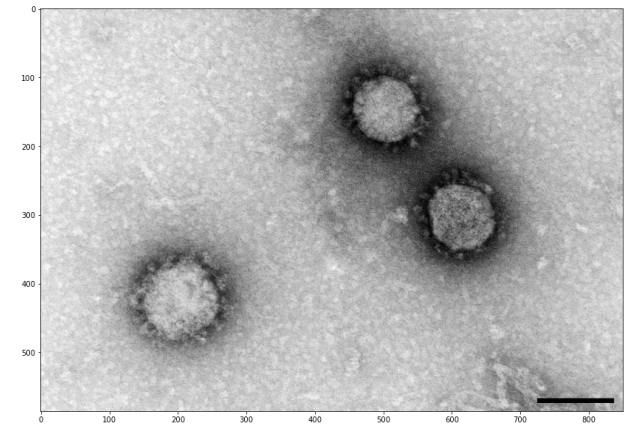




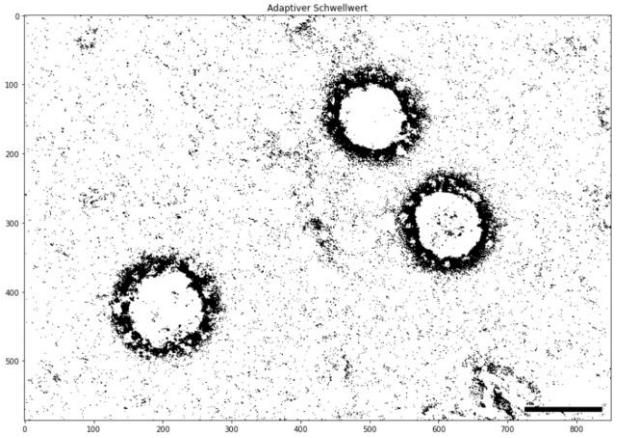
Schwellwertbasierte Segmentierung: adaptiv

Beispiel

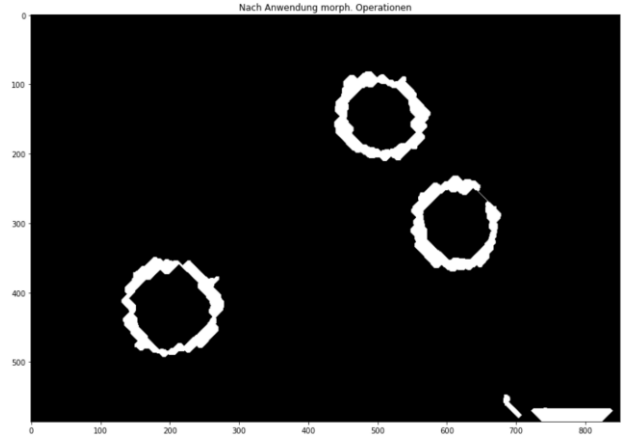
Einzelne Zellen können getrennt werden, automatische Zählung und Auswertung (Größe, Form, etc. möglich



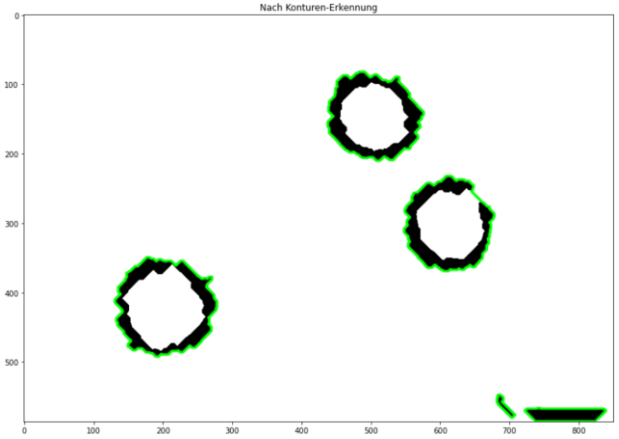
1. Vorverarbeitung (z.B. HSV)



2. Adaptiver Schwellwert



3. Morph. Operationen

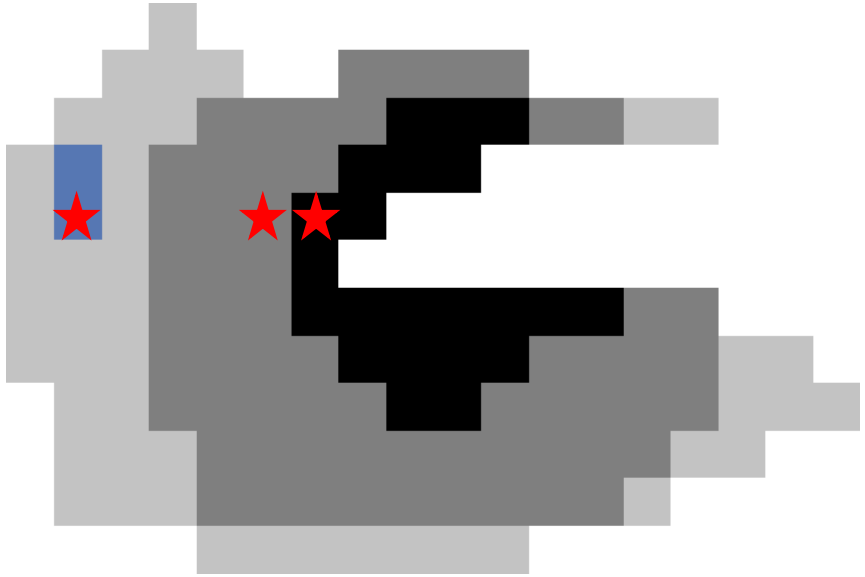


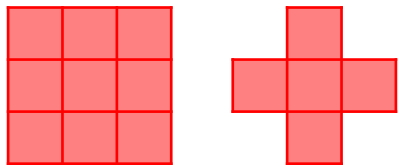
4. Konturen erkennen

Regionsbasierte Segmentierung

Region Growing

- wähle ein geeignetes Ähnlichkeitskriterium (hier: Grauwert)
- wähle initiale Startpunkte der Segmente (sog. Seed-Points)
- für jedes Segment:
  - überprüfe Nachbarn des (Seed-)Punktes
  - falls Ähnlichkeitskriterium erfüllt, füge Punkt zu Segment hinzu





Welche Nachbarschaft wurde hier gewählt?

if $|f(neighbor) - f(seed)| < Threshold$,
$g(neighbor) = 1$

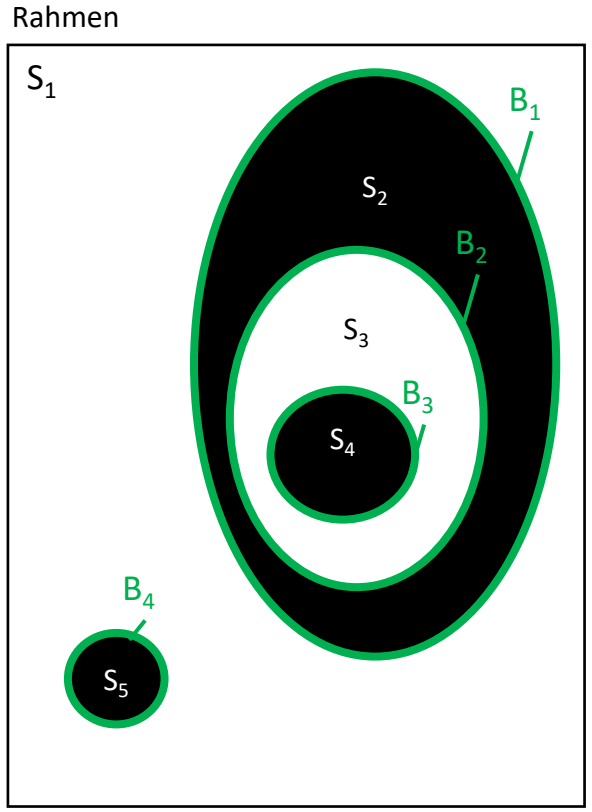
Regionsbasierte Segmentierung

Segmentierung, Konturen und Regionen

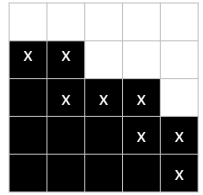
- **Segmentierung** ist der Prozess, ein Bild in verschiedene **Regionen** einzuteilen
- **Konturen** sind kontinuierliche Linien oder Kurven, die einzelne Objekte eines Bildes umranden oder ausfüllen.

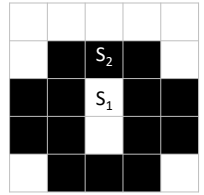
Regionsbasierte Segmentierung: Konturen

Konturerkennung = Kantenverfolgung in binären Bildern (Verfahren cv.findContours nach Suzuki und Abe)



Definitionen:



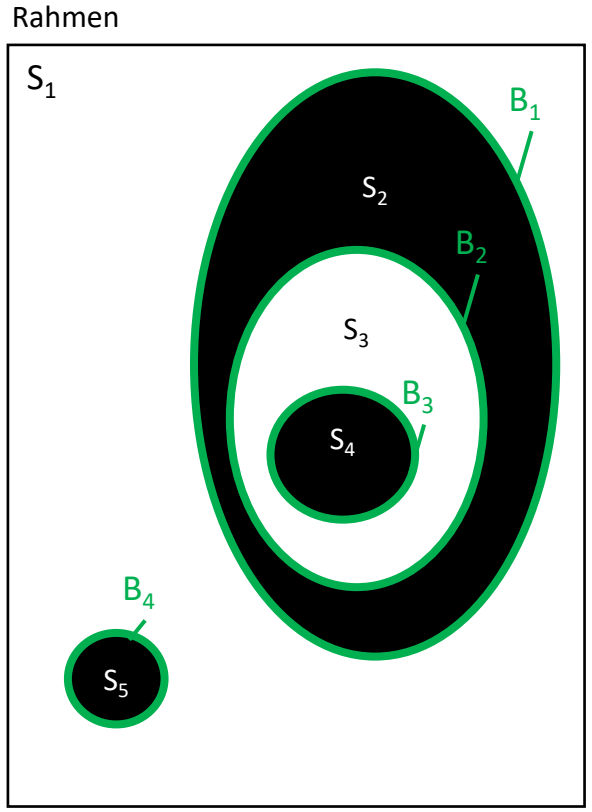


- **Randpunkt:** 1-Pixel (i,j) besitzt einen 0-Pixel (p,q) in seiner Nachbarschaft
- **Umgebung von verbundenen Segmenten:** seien S_1 und S_2 zwei verbundene Segmente. Falls ein Pixel aus S_2 für jeden 4er-Pfad eines Pixels aus S_1 existiert, so umgibt das Segment S_2 das Segment S_1. Falls S_2 das Segment S_1 umgibt und zwischen beiden Segmente mindestens ein Grenzpunkt existiert, so umgibt S_2 S_1 direkt.
- **Äußerer Rand und Lochrand:** Ein äußerer Rand ist definiert als die Menge von Randpunkten zwischen einer willkürlichen 1-Komponente und einer 0-Komponente, welche es direkt umgibt. Die Menge an Randpunkten zwischen einem Loch und der 1-Komponente, welche das Loch umgibt, wird als Lochrand bezeichnet. Der Begriff Rand wird für beide Arten verwendet.

Suzuki, Abe: Topological Structural Analysis of Digitized Binary Images by Border Following, Computer Vision, Graphics and Image Processing, 1985

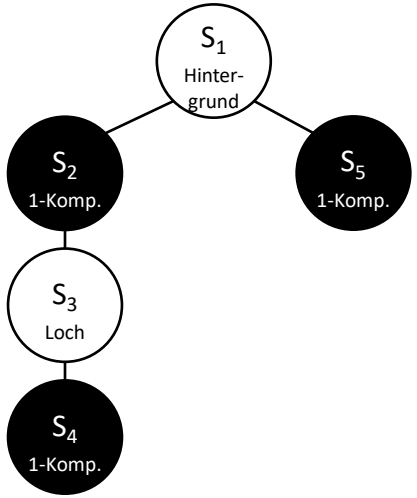
Regionsbasierte Segmentierung: Konturen

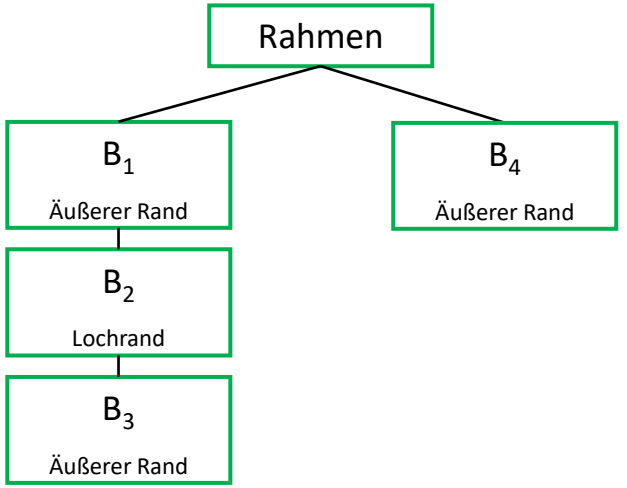
Konturerkennung = Kantenverfolgung in binären Bildern (Verfahren cv.findContours nach Suzuki und Abe)

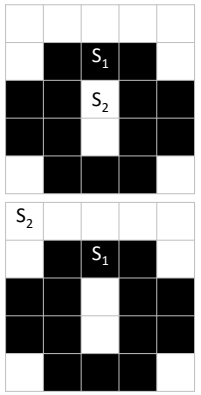


Definitionen:

- **Elterlicher Rand:** Ein elterlicher Rand eines äußeren Randes zwischen einer 1-Komponente S_1 und der 0-Komponente S_2, welche S_1 direkt umgibt, ist definiert als:
  - die Lochkante zwischen S_2 und der 1-Komponente welche S_2 umgibt, falls S_2 ein Loch ist
  - der Rahmen des Bildes, falls S_2 der Hintergrund ist



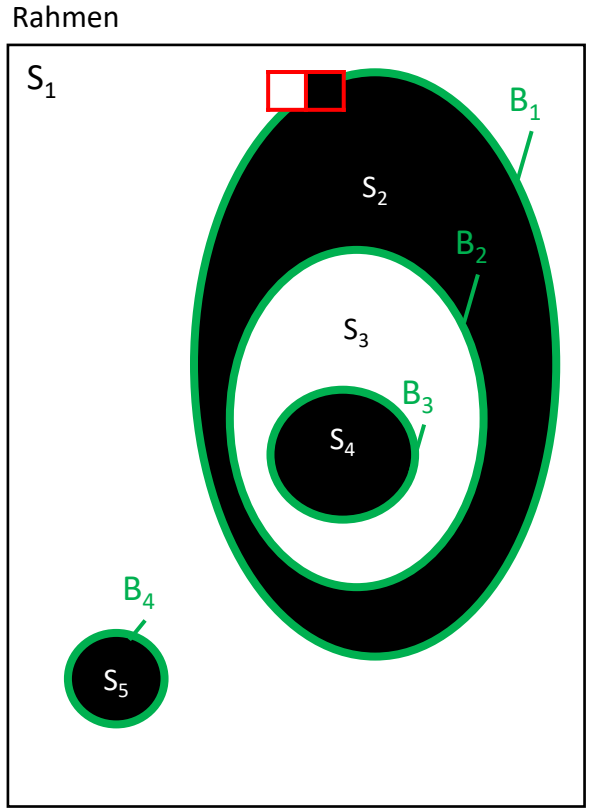




Suzuki, Abe: Topological Structural Analysis of Digitized Binary Images by Border Following, Computer Vision, Graphics and Image Processing, 1985

Regionsbasierte Segmentierung: Konturen

Konturerkennung = Kantenverfolgung in binären Bildern (Verfahren cv.findContours nach Suzuki und Abe)



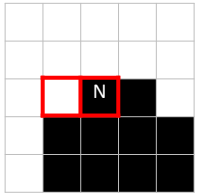
Algorithmus zur Kantenverfolgung für topologische Analysen:

- Iteriere durch das Bild, bis ein Pixel (i,j) gefunden wurde, der einer äußeren oder einer Lochkante entspricht. Falls beide Bedingungen (äußere und Lochrand) erfüllt sind, so ist Pixel (i,j) als Startpunkt des äußeren Randes zu betrachten. Weise diesem Rand eine eindeutige ID (=NBD) zu.
- Finde den elterlichen Rand heraus: Während der Iterationen wird die ID des zuletzt aufgetretenen (äußeren oder Loch-) Rands als LNBD gespeichert. Der elterliche Rand des neu entdeckten Randes B hängt von der eigenen Art und der Randart des zuletzt aufgetretenen Randes B‘ ab:

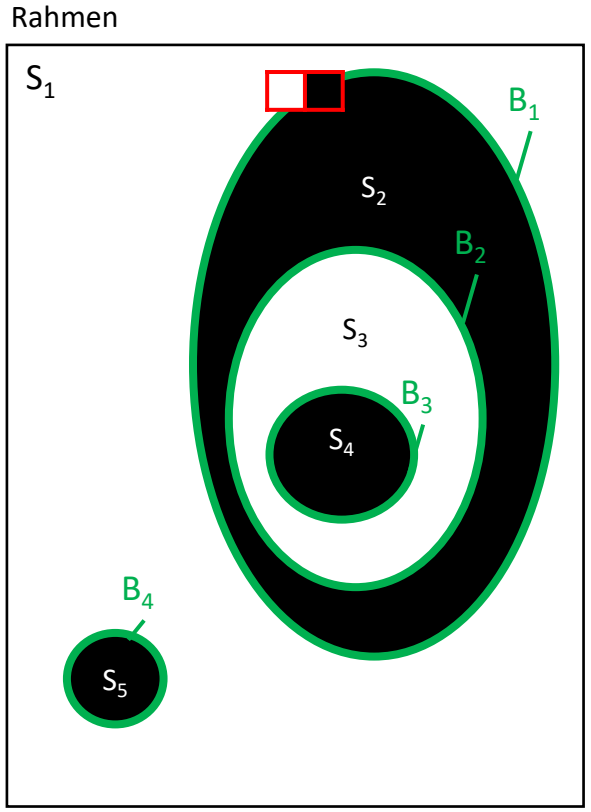
		Art des Randes von B‘	
		Äußerer Rand	Lochrand
Art des Randes von B	Äußerer Rand	Elterlicher Rand von B‘	Rand B‘
	Lochrand	Rand B‘	Elterlicher Rand von B‘

Suzuki, Abe: Topological Structural Analysis of Digitized Binary Images by Border Following, Computer Vision, Graphics and Image Processing, 1985

Regionsbasierte Segmentierung: Konturen

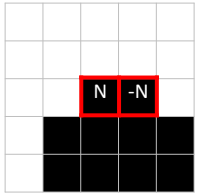


Konturerkennung = Kantenverfolgung in binären Bildern (Verfahren cv.findContours nach Suzuki und Abe)



Algorithmus zur Kantenverfolgung für topologische Analysen:

- Folge dem gefundenen Rand, beginnend beim Startpunkt:
  - falls der aktuell zu folgende Rand zwischen einer 0-Komponente, welche das Pixel #2 (p, q+1) und die 1-Komponente, welche das Pixel #1 (p,q) enthält, so weise dem Pixel (p,q) –N(BD) zu
  - andernfalls weise dem Pixel (p,q) den Wert N(BD) zu, außer (p,q) ist bereits Teil eines Randes
- Setze danach die iterative Suche nach weiteren Rändern fort, bis letztes Pixel erreicht wurde

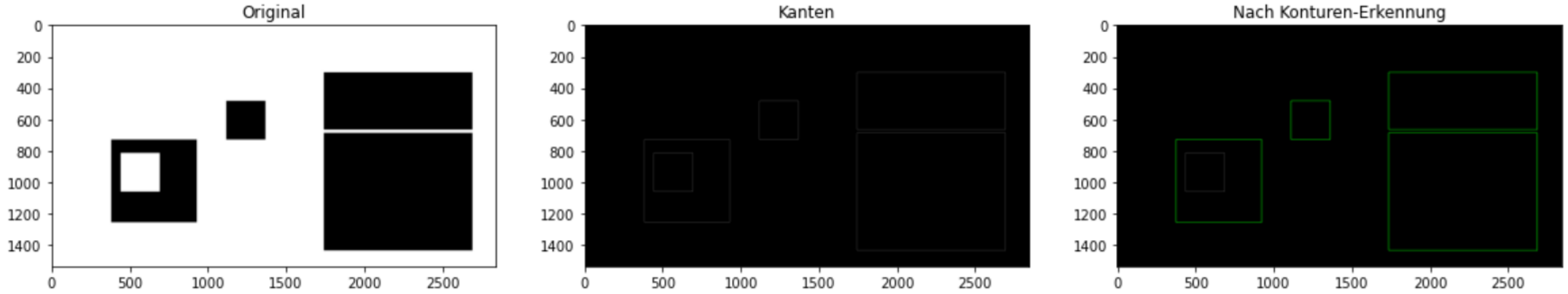


Suzuki, Abe: Topological Structural Analysis of Digitized Binary Images by Border Following, Computer Vision, Graphics and Image Processing, 1985

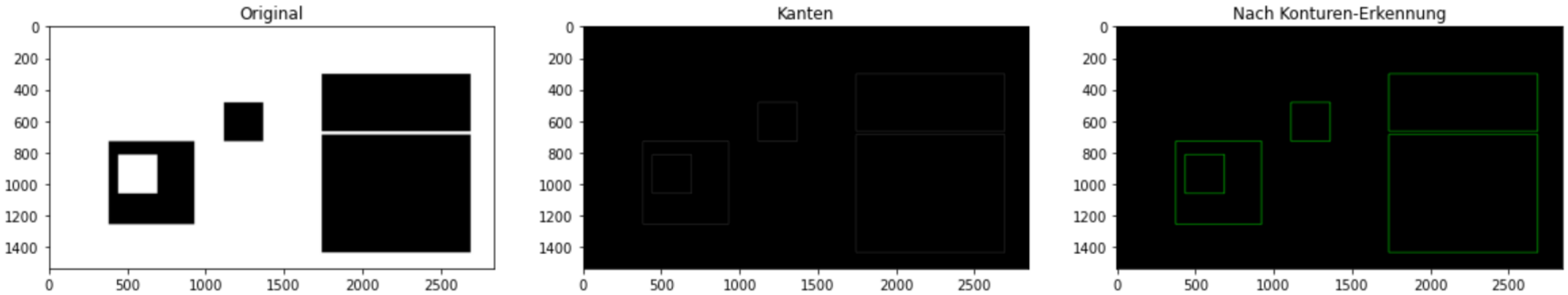
Regionsbasierte Segmentierung: Konturen

- Beispiel:

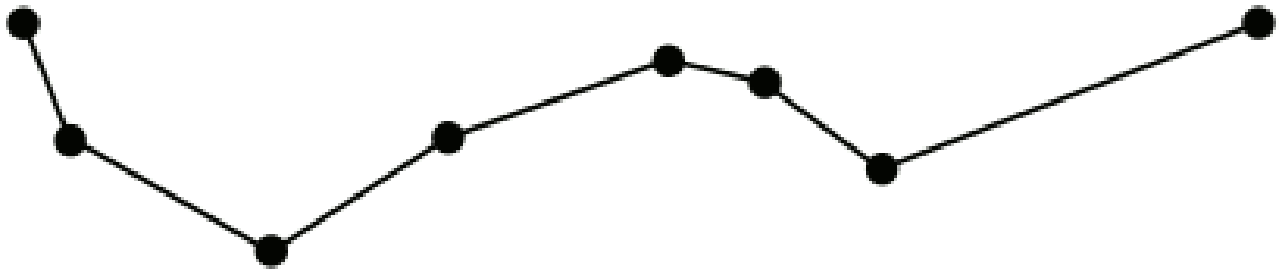
RETR_EXTERNAL



RETR_LIST



Annäherung Polygonale Kurve



Ramer–Douglas–Peucker algorithm

The starting curve is an ordered set of points or lines and the distance dimension $\varepsilon > 0$.

The algorithm recursively divides the line. Initially it is given all the points between the first and last point. It automatically marks the first and last point to be kept. It then finds the point that is farthest from the line segment with the first and last points as end points; this point is obviously farthest on the curve from the approximating line segment between the end points. If the point is closer than ε to the line segment, then any points not currently marked to be kept can be discarded without the simplified curve being worse than ε.

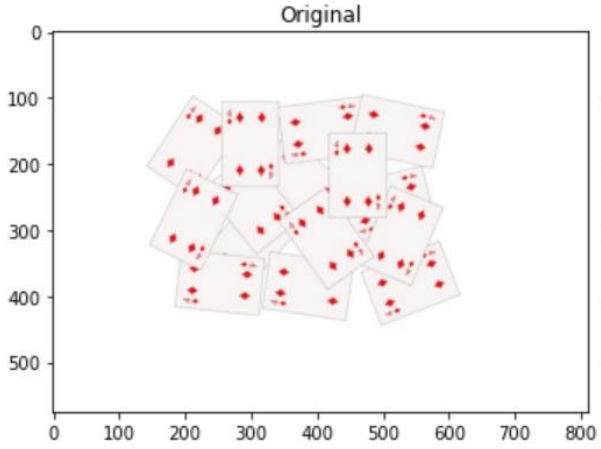
If the point farthest from the line segment is greater than ε from the approximation then that point must be kept.

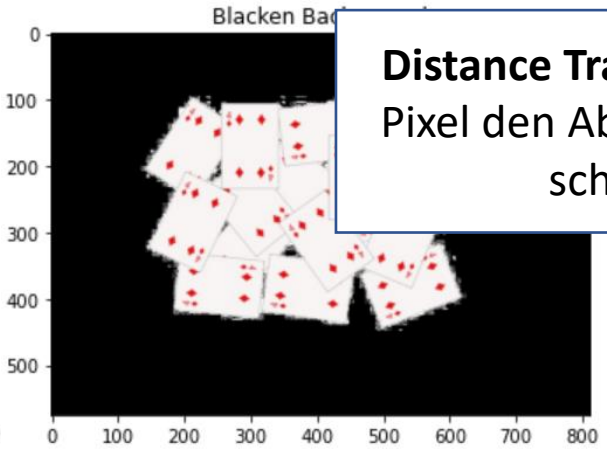
The algorithm recursively calls itself with the first point and the farthest point and then with the farthest point and the last point, which includes the farthest point being marked as kept.

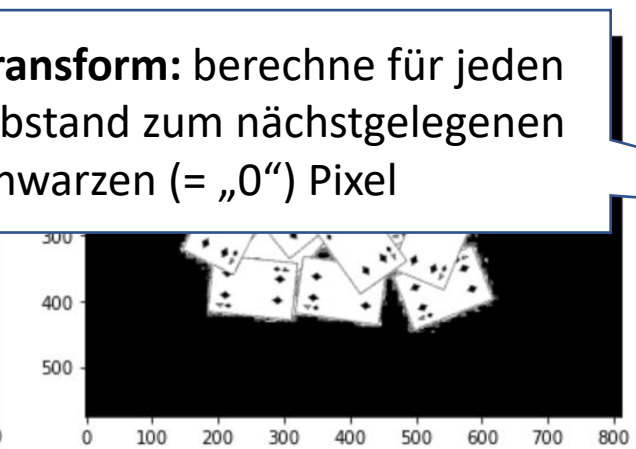
When the recursion is completed a new output curve can be generated consisting of all and only those points that have been marked as kept.

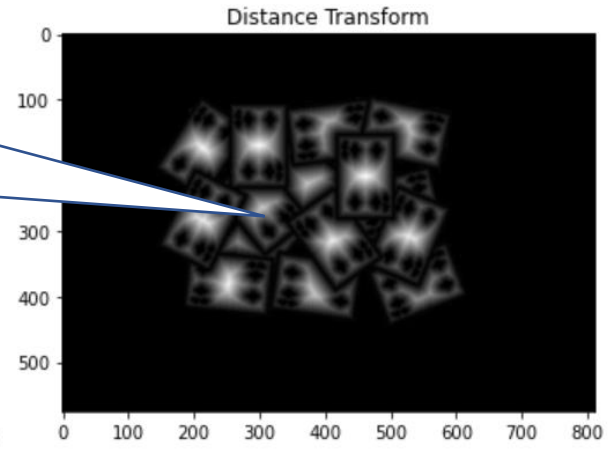
Ramer–Douglas–Peucker algorithm - Wikipedia

Watershed Algorithmus

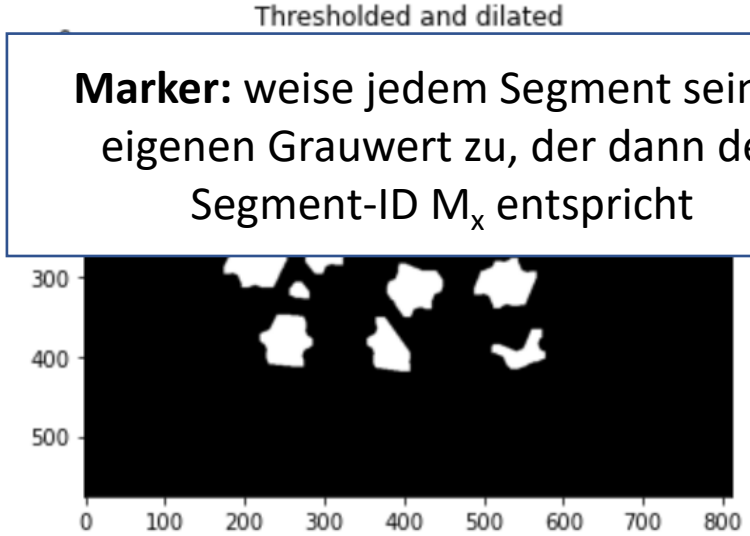








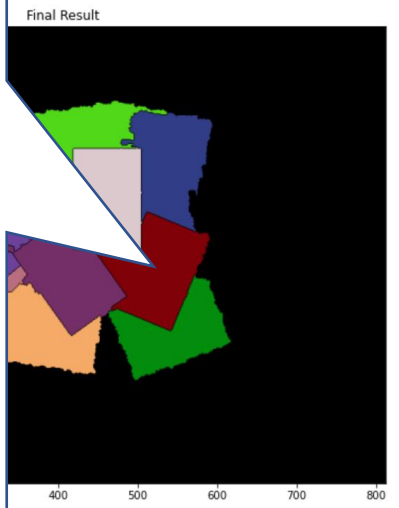
Distance Transform: berechne für jeden Pixel den Abstand zum nächstgelegenen schwarzen (= „0“) Pixel



Marker: weise jedem Segment seinen eigenen Grauwert zu, der dann der Segment-ID M_x entspricht

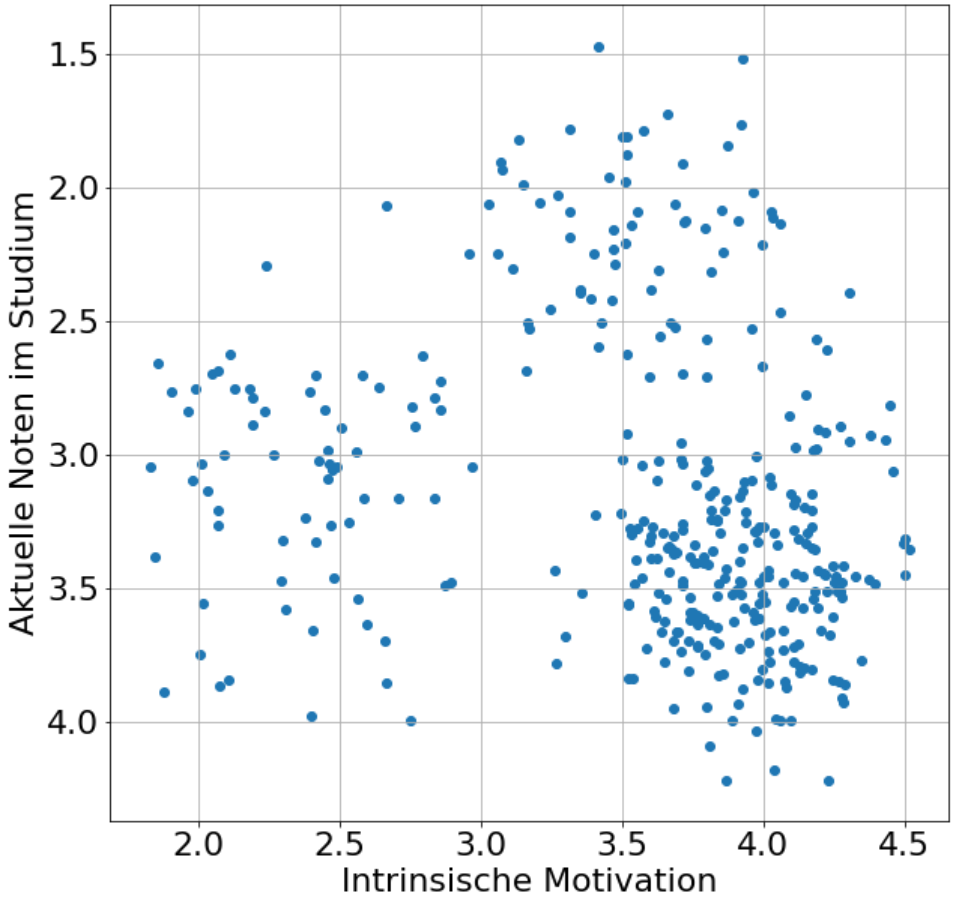
Watershed:

1. füge die benachbarten Pixel für jede markierte Region in eine Priority Queue
   Prio = Absolutwert des Gradienten
2. Pixel mit höchster Prio wird der Queue entnommen
   Falls benachbarte Pixel bereits einer gleichen Region zugewiesen wurden, so weise neuem Pixel Region zu
   Alle noch nicht markierten Nachbarn, die bisher noch nicht in der Priority Queue sind, werden hinugefügt



Segmentierung mit K-Means

Erkennung von Clustern mit K-Means

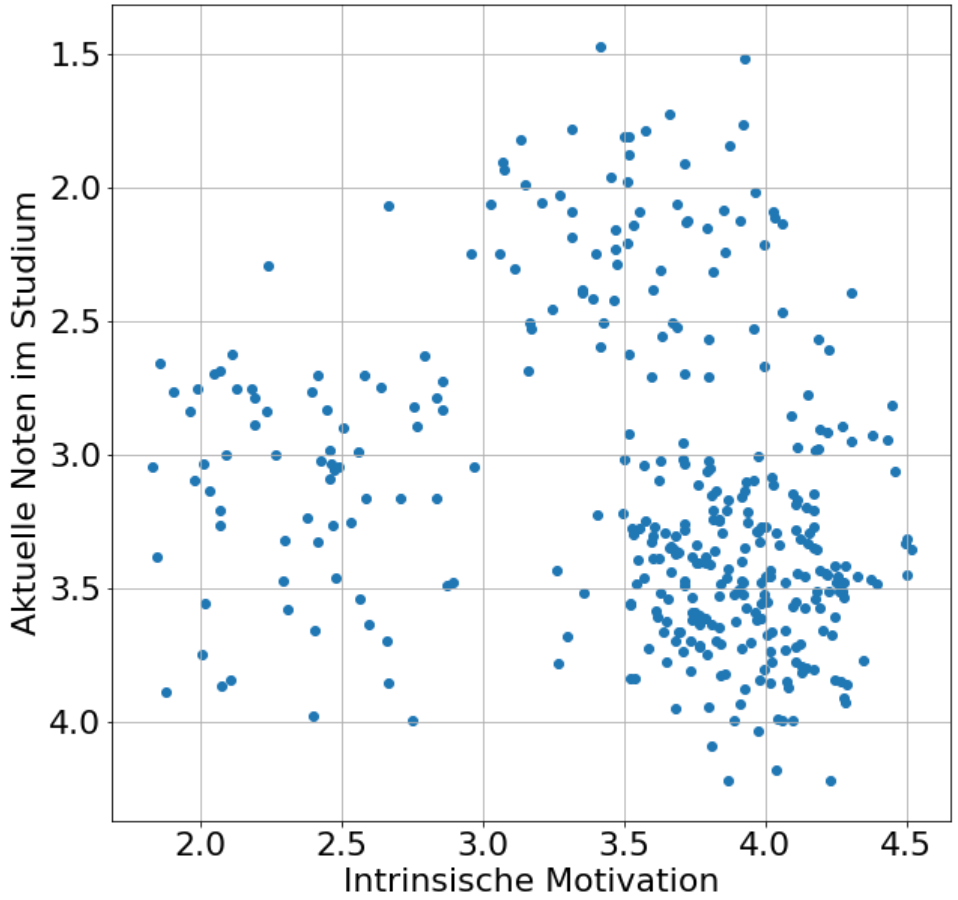


Gegeben
- eine Menge $\mathcal{T} = \{\overrightarrow{x_1}, \overrightarrow{x_2}, \dots, \overrightarrow{x_n}\} \subset X$ von Beobachtungen
- eine Anzahl K zu erkennender „Cluster“
- eine Abstandsfunktion $d(\overrightarrow{x_i}, \overrightarrow{x_j})$
- eine Qualitätsfunktion

Finde
- Cluster $C_1, C_2, \dots, C_K$, so dass alle $\vec{x} \in X$ einer Gruppe zugeordnet sind
- und die Qualitätsfunktion optimiert wird
  - Abstand zwischen Beobachtungen derselben Gruppe minimal
  - Abstand zwischen Beobachtungen anderer Gruppen maximal

Segmentierung mit K-Means

Erkennung von Clustern mit K-Means



1. initialisiere die Cluster-Mittelpunkte $\overrightarrow{\mu_1}, \overrightarrow{\mu_2}, \dots, \overrightarrow{\mu_K} \in \mathcal{T}$ zufällig

2. berechne die Abstände zwischen der Beobachtung $\overrightarrow{x_i}$ zu allen Cluster-Mittelpunkten und wähle das Cluster mit dem geringsten Abstand d

$$c_i = \arg\min_j ||\overrightarrow{x_i} - \overrightarrow{\mu_j}||$$

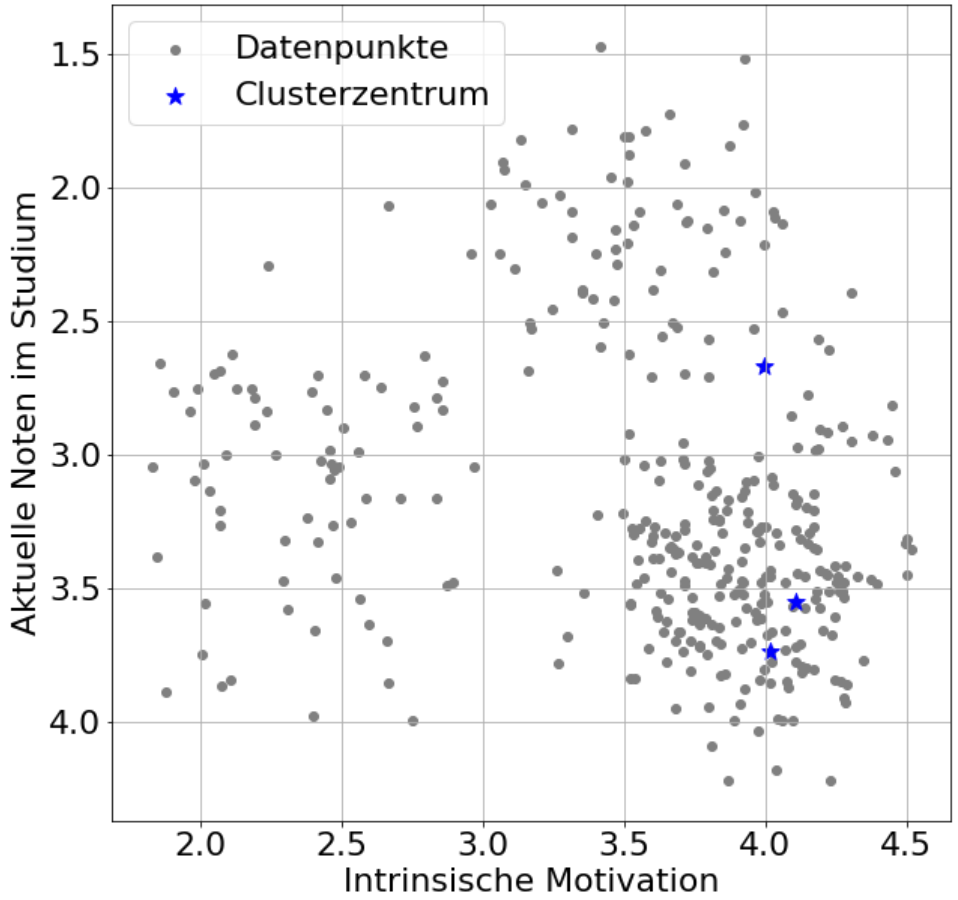
3. berechne die neuen Mittelpunkte $\overrightarrow{\mu_j}$ entsprechend der Cluster-Zuordnung der Punkte

$$\overrightarrow{\mu_j} = \frac{\sum_{i=1}^{n} 1\{c_i = j\}\, \overrightarrow{x_i}}{\sum_{i=1}^{n} 1\{c_i = j\}}$$

4. Wiederhole Schritte 2 und 3, bis $\overrightarrow{\mu_j}$ konvergieren

Segmentierung mit K-Means

Erkennung von Clustern mit K-Means



1. initialisiere die Cluster-Mittelpunkte $\overrightarrow{\mu_1}, \overrightarrow{\mu_2}, \ldots, \overrightarrow{\mu_K} \in \mathcal{T}$ zufällig

2. berechne die Abstände zwischen der Beobachtung $\overrightarrow{x_i}$ zu allen Cluster-Mittelpunkten und wähle das Cluster mit dem geringsten Abstand d

$$c_i = \arg\min_j ||\overrightarrow{x_i} - \overrightarrow{\mu_j}||$$

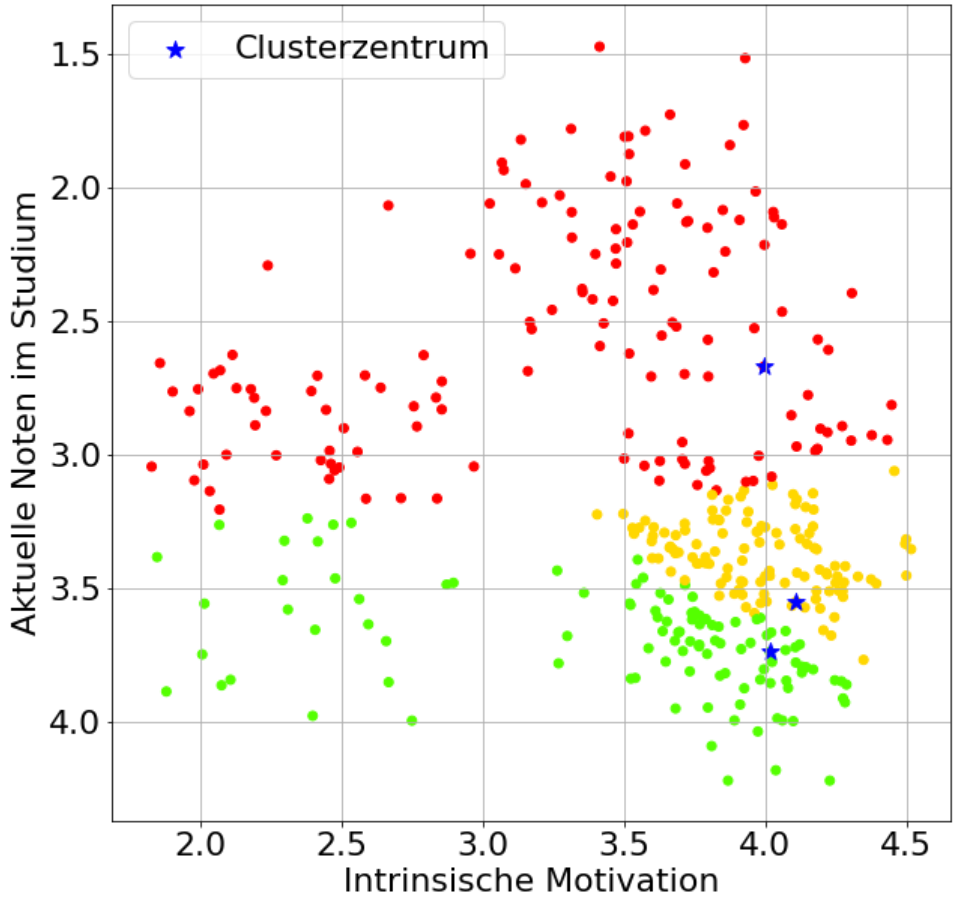
3. berechne die neuen Mittelpunkte $\overrightarrow{\mu_j}$ entsprechend der Cluster-Zuordnung der Punkte

$$\overrightarrow{\mu_j} = \frac{\sum_{i=1}^{n} 1\{c_i = j\}\,\overrightarrow{x_i}}{\sum_{i=1}^{n} 1\{c_i = j\}}$$

4. Wiederhole Schritte 2 und 3, bis $\overrightarrow{\mu_j}$ konvergieren

Segmentierung mit K-Means

Erkennung von Clustern mit K-Means



1. initialisiere die Cluster-Mittelpunkte $\overrightarrow{\mu_1}, \overrightarrow{\mu_2}, \ldots, \overrightarrow{\mu_K} \in \mathcal{T}$ zufällig

2. berechne die Abstände zwischen der Beobachtung $\overrightarrow{x_i}$ zu allen Cluster-Mittelpunkten und wähle das Cluster mit dem geringsten Abstand d

$$c_i = \arg\min_j ||\overrightarrow{x_i} - \overrightarrow{\mu_j}||$$

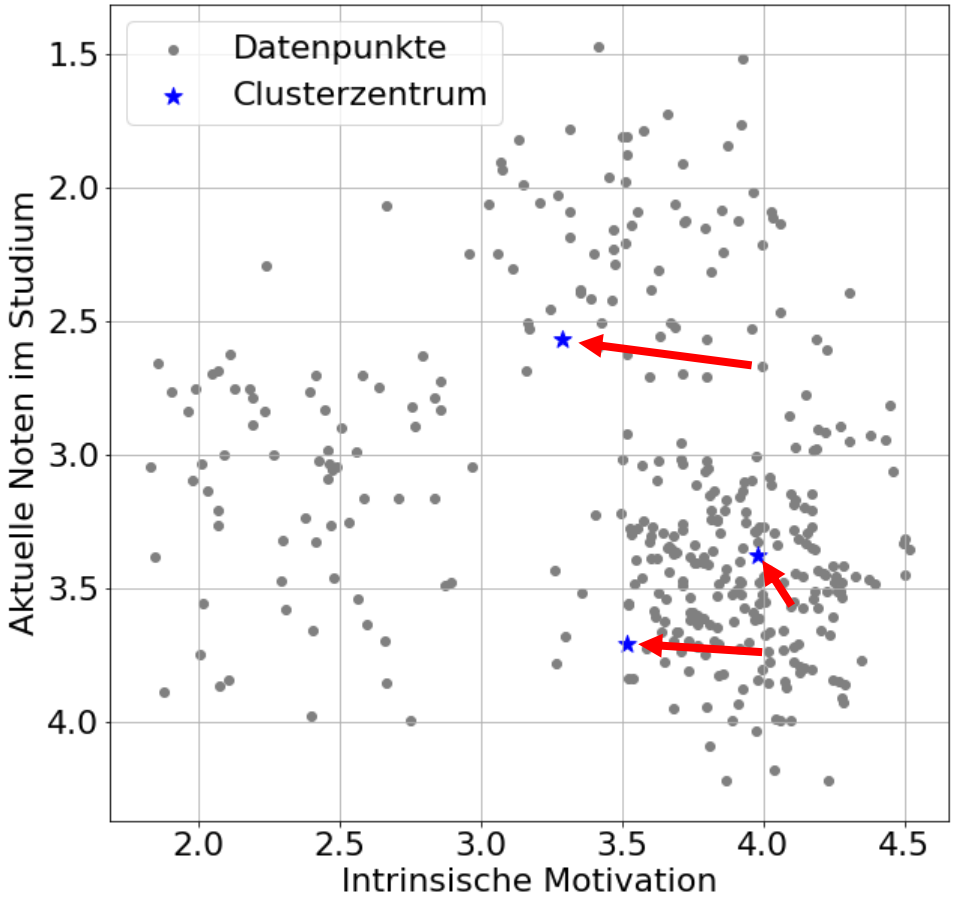
3. berechne die neuen Mittelpunkte $\overrightarrow{\mu_j}$ entsprechend der Cluster-Zuordnung der Punkte

$$\overrightarrow{\mu_j} = \frac{\sum_{i=1}^{n} 1\{c_i = j\}\,\overrightarrow{x_i}}{\sum_{i=1}^{n} 1\{c_i = j\}}$$

4. Wiederhole Schritte 2 und 3, bis $\overrightarrow{\mu_j}$ konvergieren

Segmentierung mit K-Means

Erkennung von Clustern mit K-Means



K-Means Clustering

1. initialisiere die Cluster-Mittelpunkte $\overrightarrow{\mu_1}, \overrightarrow{\mu_2}, \ldots, \overrightarrow{\mu_K} \in \mathcal{T}$ zufällig

2. berechne die Abstände zwischen der Beobachtung $\overrightarrow{x_i}$ zu allen Cluster-Mittelpunkten und wähle das Cluster mit dem geringsten Abstand d

$$c_i = \arg\min_j ||\overrightarrow{x_i} - \overrightarrow{\mu_j}||$$

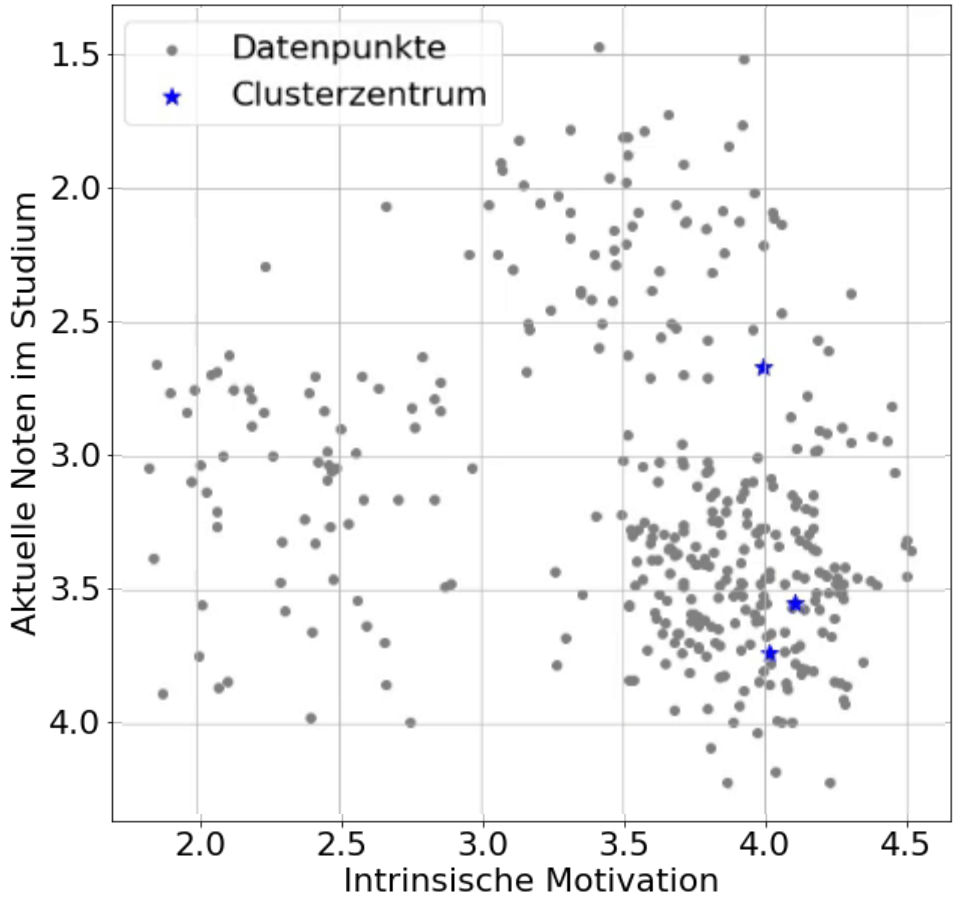
3. berechne die neuen Mittelpunkte $\overrightarrow{\mu_j}$ entsprechend der Cluster-Zuordnung der Punkte

$$\overrightarrow{\mu_j} = \frac{\sum_{i=1}^{n} 1\{c_i = j\}\,\overrightarrow{x_i}}{\sum_{i=1}^{n} 1\{c_i = j\}}$$

4. Wiederhole Schritte 2 und 3, bis $\overrightarrow{\mu_j}$ konvergieren

Segmentierung mit K-Means

Erkennung von Clustern mit K-Means



1. initialisiere die Cluster-Mittelpunkte $\overrightarrow{\mu_1}, \overrightarrow{\mu_2}, \ldots, \overrightarrow{\mu_K} \in \mathcal{T}$ zufällig

2. berechne die Abstände zwischen der Beobachtung $\overrightarrow{x_i}$ zu allen Cluster-Mittelpunkten und wähle das Cluster mit dem geringsten Abstand d

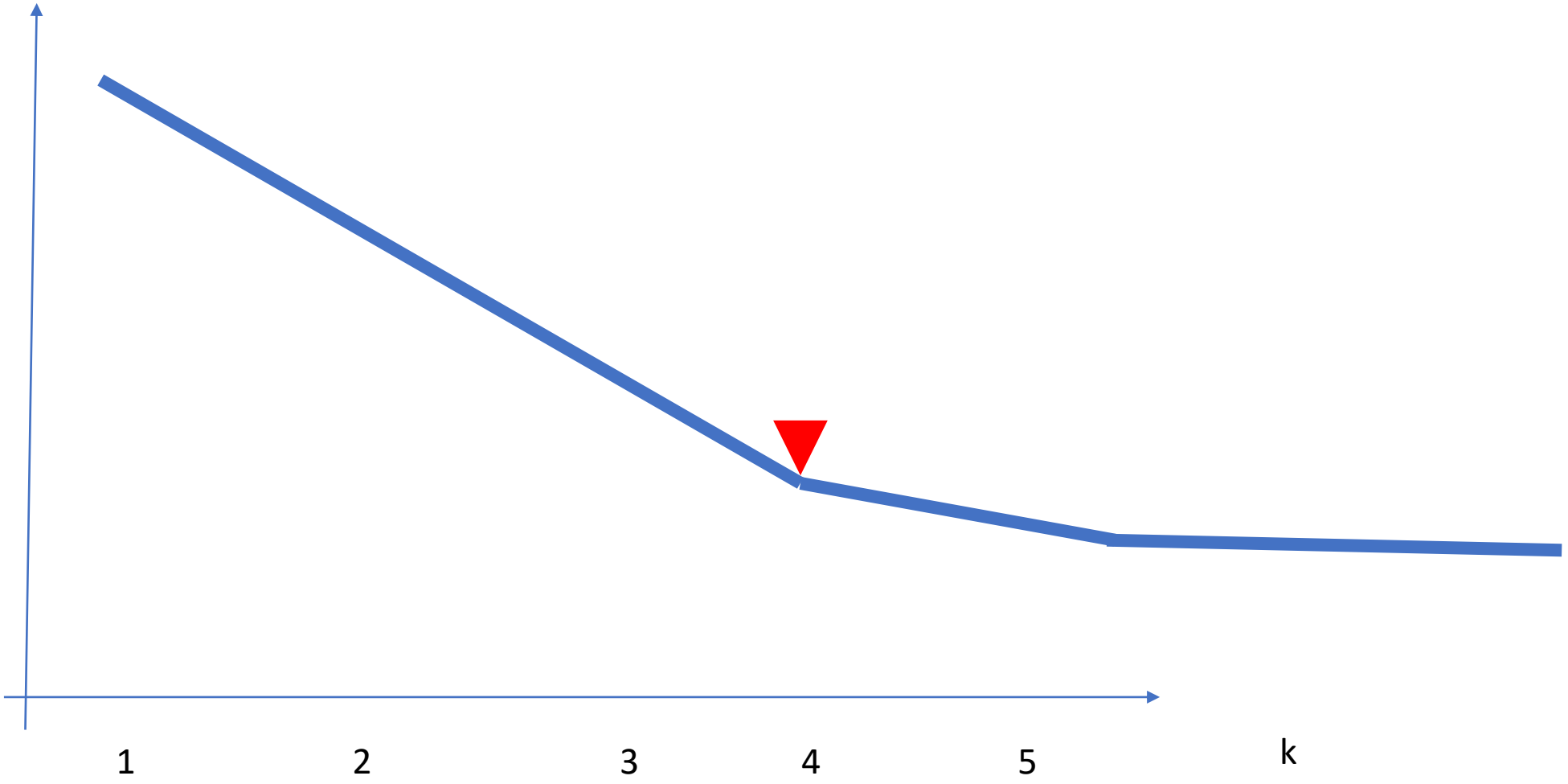
$$c_i = \arg\min_j ||\overrightarrow{x_i} - \overrightarrow{\mu_j}||$$

3. berechne die neuen Mittelpunkte $\overrightarrow{\mu_j}$ entsprechend der Cluster-Zuordnung der Punkte

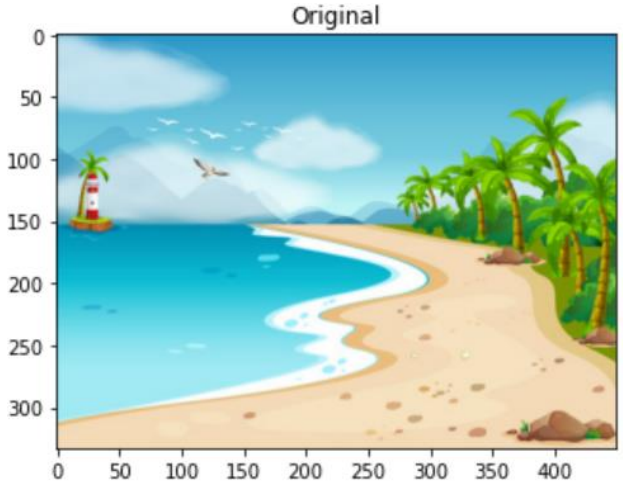
$$\overrightarrow{\mu_j} = \frac{\sum_{i=1}^{n} 1\{c_i = j\}\,\overrightarrow{x_i}}{\sum_{i=1}^{n} 1\{c_i = j\}}$$

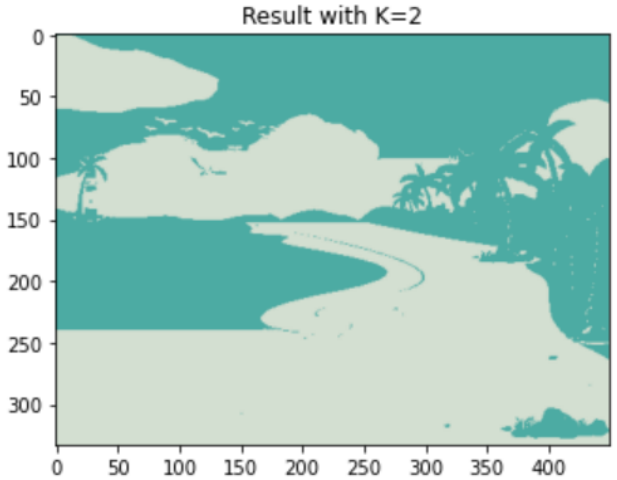
4. Wiederhole Schritte 2 und 3, bis $\overrightarrow{\mu_j}$ konvergieren

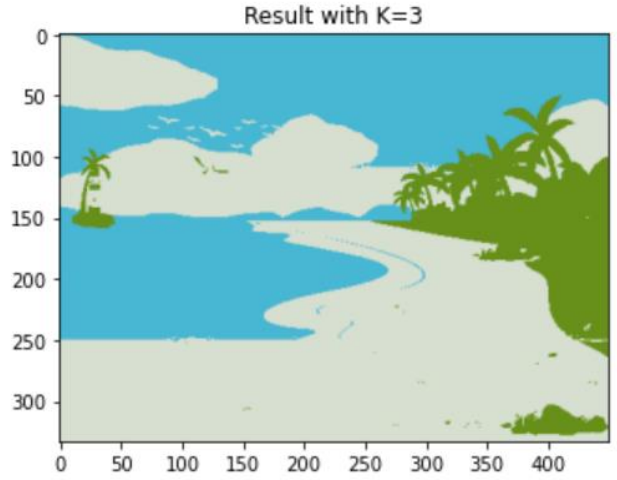
Bestimmung K

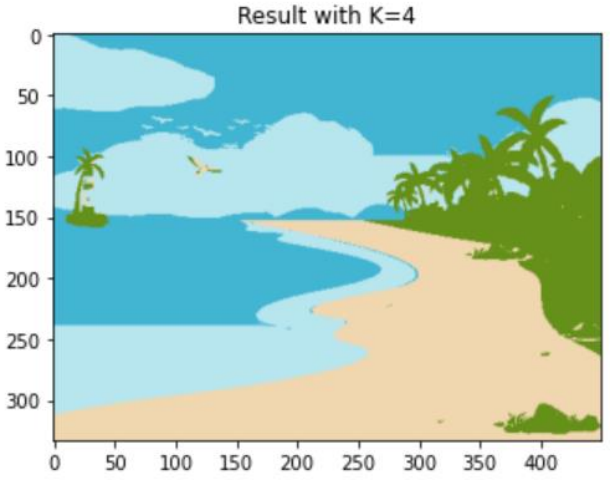


Segmentierung mit K-Means









Zusammenfassung

- **Segmentierung** ist der Prozess, ein Bild in verschiedene **Regionen** einzuteilen
- Trennung zwischen Regionen kann durch
  - Punkte, Kanten oder Konturen vorgenommen werden
  - Farbwerte vorgenommen werden

Beispielbilder