

I. Einführung und Histogrammbasierte Bildverarbeitung

Grundlagen intelligenter Sehsysteme

Definition: Intelligentes Sehsystem

Intelligente Sehsysteme sollen digitale Bilddaten interpretieren um z.B. Objekte zu erkennen.

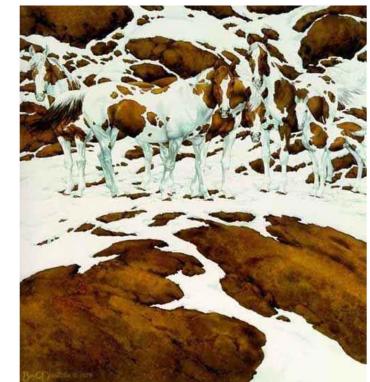
Die Bilddaten können als Funktion f angenommen werden, die die Welt W auf einen Sensorstimulus S abbildet:

$$S = f(W)$$

Computersehen berechnet die Welt aus S :

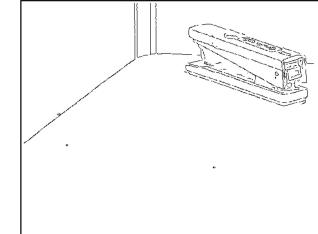
$$W = f^{-1}(S)$$

→ dieses inverse Problem f^{-1} ist i.A. unterbestimmt und mehrdeutig

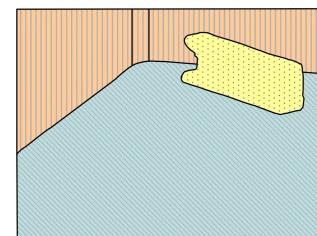
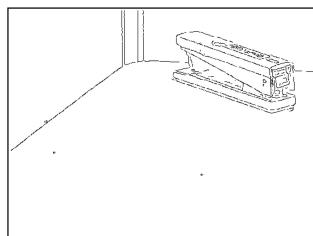


Phasen des Computersehens

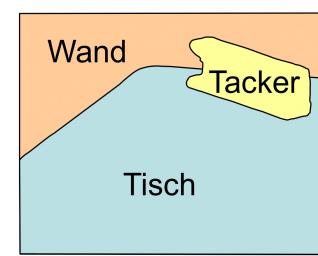
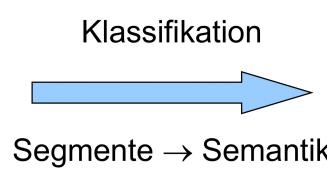
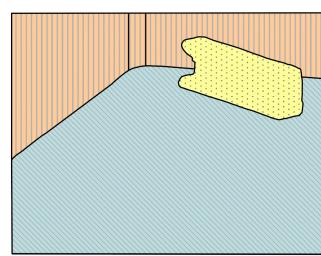
Low-Level Vision: Kontrastoptimierung, Glättung, Hervorhebung relevanter Punkte



Mid-Level Vision: Gruppierung von Konturpunkten zu Konturlinien, Zerlegung in Bildsegmente durch Konturlinien

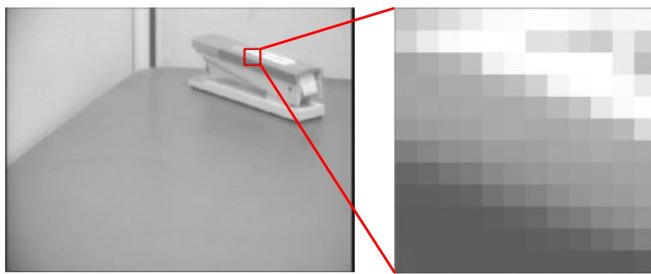


High-Level Vision: Zuordnung zu Objektklassen, inhaltliche Beschreibung



Einkanalige Bilder

Grauwertbilder werden durch eine Matrix von Pixeln dargestellt, wobei die Werte die Intensität beschreiben



195	209	221	235	249	251	254	255	250	241	247	248
210	236	249	254	255	254	225	226	212	204	236	211
164	172	180	192	241	251	255	255	255	255	235	190
167	164	171	170	179	189	208	244	254	255	251	234
162	167	166	169	169	170	176	185	196	232	249	254
153	157	160	162	169	170	168	169	171	176	185	218
126	135	143	147	156	157	160	166	167	171	168	170
103	107	118	125	133	145	151	156	158	159	163	164
095	095	097	101	115	124	132	142	117	122	124	161
093	093	093	093	095	099	105	118	125	135	143	119
093	093	093	093	093	093	095	097	101	109	119	132
095	093	093	093	093	093	093	093	093	093	093	119

Bildquelle: Stuart Russell, Peter Norvig:
"Artificial Intelligence - A Modern Approach", Prentice Hall, 2003.

mit: Intensitätsspektrum $\{I_{\min}, \dots, I_{\max}\} = \{0, \dots, 255\}$ $I_{\min} = 0$ (schwarz)
 $I_{\max} = 255$ (weiß)

Mehrkanalige Bilder

Jeder Pixel besteht aus einem k-Tupel mit identischem Intensitätsspektrum: $I = [I(x,y,k)]$

Histogrammbasierte Bildverarbeitung

Definition: Intensitätshistogramm einhnmaliger Bilder

Histogramme stellen die Verteilung der Pixelwerte dar und sind Bestandteil der Low-Level Vision

Intensitätshistogramm: ist eine diskrete Funktion, die jedem Intensitätswert $I \in \{0, \dots, I_{\max}\}$ die Zahl der Pixel im Bild zuordnet:

$$h_I(I) = n_I.$$

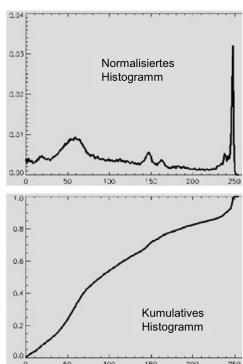
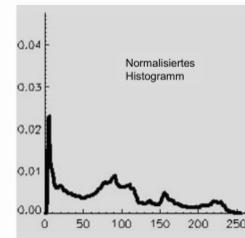
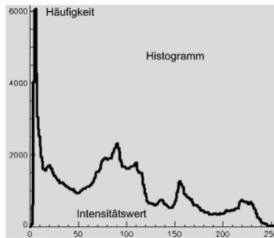
normalisiertes Intensitätshistogramm: normalisiert die Einträge mit der Anzahl an Pixeln:

$$p_I(I) = \frac{n_I}{S \cdot Z}$$

kumulatives Intensitätshistogramm: für $s_I(I)$ werden die relativen Häufigkeiten $p_I(I')$ mit $I' < I$ aufsummiert:

$$s_I(I) = \sum_{i=0}^I p_I(i), I = 0, \dots, I_{\max}$$

Es gilt: $0 \leq s_I(I) \leq 1$



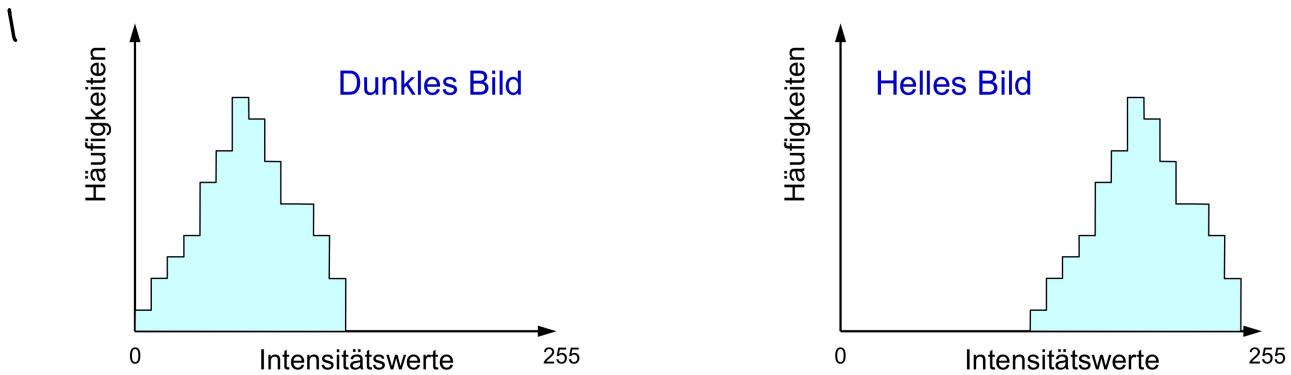
Mittelwert:

$$m_I = \frac{1}{N} \sum_{I=0}^{I_{\max}} I \cdot N \cdot p_I(I) = \sum_{I=0}^{I_{\max}} I \cdot p_I(I)$$

Mittlere quadratische Abweichung:

$$q_I = \frac{1}{N} \sum_{I=0}^{I_{\max}} (I - m_I)^2 \cdot N \cdot p_I(I) = \sum_{I=0}^{I_{\max}} (I - m_I)^2 \cdot p_I(I)$$

Interpretation von Histogrammen



→ Beide Bilder weisen einen geringen Kontrast auf

2.



→ Bimodales Histogramm kann auf einen Scan deuten

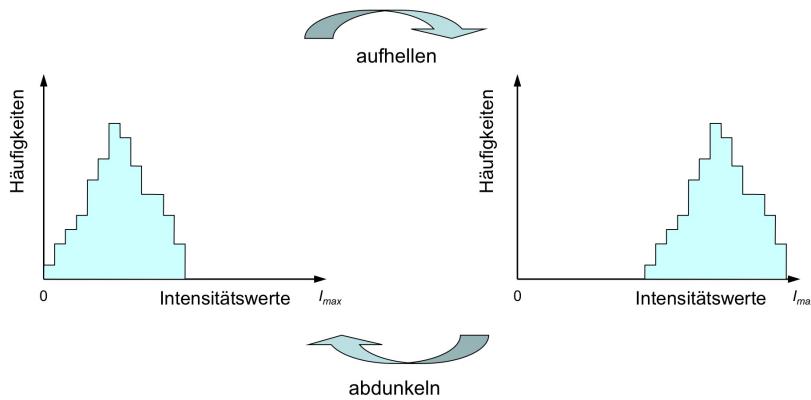
Definition: Intensitätshistogramm mehrkanaliger Bilder

Intensitätshistogramm: eine diskrete Funktion die jedem k -Tupel die Anzahl an Pixeln zuweist, die genau dieses Tupel aufweisen: $h_I(I_0, \dots, I_{K-1}) = n_{(I_0, \dots, I_{K-1})}$

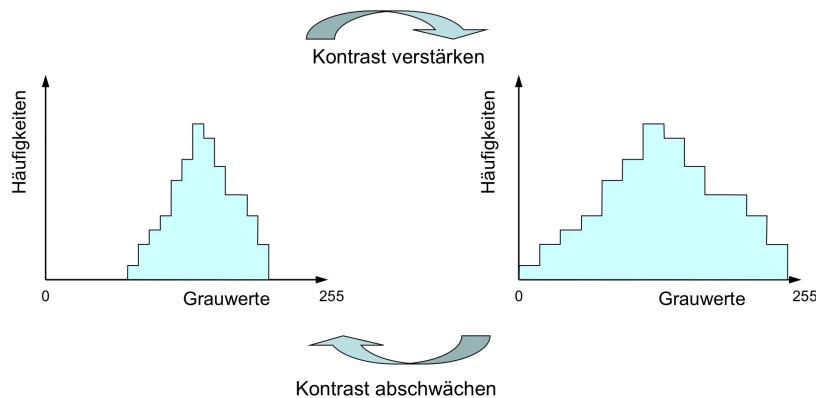
Normalisiertes Intensitätshistogramm: normalisiert die Werte durch die Anzahl der Pixel:

$$p_I(I_0, \dots, I_{K-1}) = \frac{n_{(I_0, \dots, I_{K-1})}}{N}$$

Histogrammbasierte Aufhellung / Abdunklung



Histogrammbasierte Kontrastverstärkung / -verminderung



Definition: Globaler Kontrast

Abstand zwischen dem minimalen und maximalen Intensitätswert in einem Bild.

$$C_{global} = (I_{maxGiven} - I_{minGiven}) / (I_{max} - I_{min})$$

→ ein Bild, das nicht das gesamte Intensitätsspektrum nutzt, hat einen suboptimalen Kontrast

Definition: Lokaler Kontrast

Kontrast mit Bezug auf die lokale Nachbarschaft

$$C_{local}(I) = \frac{1}{Z \cdot S} \sum_{x=0}^{Z-1} \sum_{y=0}^{S-1} |I(x, y) - \bar{I}(x, y)|$$

\bar{I} : durchschnittlicher Intensitätswert in der lokalen Nachbarschaft

Lineare Transferfunktionen

Funktionen zur Aufhellung/Abdunklung oder Kontrastverstärkung/-verminderung:

$$T(I) = (I + c_1) \cdot c_2$$

Koeffizientenwahl: $c_1 = 0$ und $c_2 = 1$: die identische Abbildung

$c_1 > 0$: Aufhellung

$c_1 < 0$: Abdunklung

$c_2 > 1$: Kontraststeigerung

$c_2 < 1$: Kontrastminderung



Lineare Grauwertspezierung:

$$T(I) = [(I - I_{minGiven}) \cdot (I_{max} / (I_{maxGiven} - I_{minGiven}))]$$

Mit: $c_1 = -I_{minGiven}$
 $c_2 = I_{max} / (I_{maxGiven} - I_{minGiven})$

Wenn $I_{min} = 0$. Bei $I_{min} > 0$:
 $c_1 = I_{min} - I_{minGiven}$
 $c_2 = (I_{max} - I_{min}) / (I_{maxGiven} - I_{minGiven})$.

Gamma-Korrektur

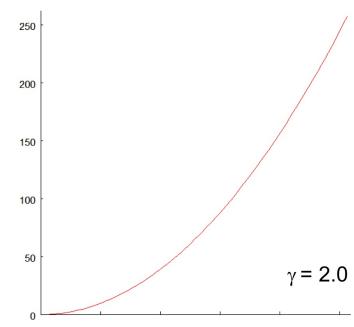
Durch z.B. Über- oder Unterbelichtung sind die häufigsten Intensitätswerte im hohen oder niedrigen Bereich. Gamma-Korrektur kann so den lokalen Kontrast verbessern.

$$T_\gamma(I) = \left[N_G \left(\frac{I - I_{min}}{I_{max} - I_{min}} \right)^\gamma + I_{min} \right] \quad N_G = I_{max} + 1$$

Gammawahl:

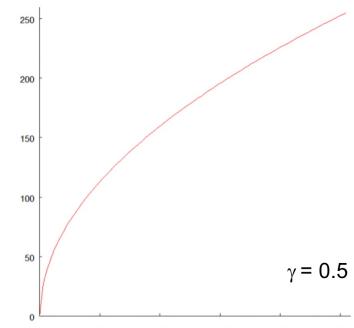
Bei $\gamma > 1$:

- hohe Intensitätswerte werden gespreizt
- niedrige Intensitätswerte werden gestaucht
- Anwendung bei überbelichtetem Bild



Bei $\gamma < 1$:

- niedrige Intensitätswerte werden gespreizt
- hohe Intensitätswerte werden gestaucht
- Anwendung bei unterbelichtetem Bild



Definition: Entropie

Die Entropie ist ein Maß für den mittleren Informationsgehalt eines Bildes. Sie bildet ein drittes Maß für den Kontrast

$$H_I = - \sum_{I=0}^{I_{\max}} p_I(I) \cdot \log_2 p_I(I)$$

Beispiele: homogenes Bild $I = [I(x,y)] = I^*$:

$$H = - \sum_{I=I^*} p_I(I) \cdot \log_2(p_I(I)) - \sum_{I \neq I^*} p_I(I) \cdot \log_2(p_I(I)) = -1 \cdot 0 - 0 = 0$$

Zweipegelbild $I = [I(x,y)]$ mit $p_I(I_1) = 50\%$ und $p_I(I_2) = 50\%$

$$H = - \sum_{I=I_1} p_I(I) \cdot \log_2(p_I(I)) - \sum_{I=I_2} p_I(I) \cdot \log_2(p_I(I)) = -\frac{1}{2} \cdot \log_2(\frac{1}{2}) - \frac{1}{2} \cdot \log_2(\frac{1}{2}) = 1$$

gleich verteiltes Grauwertbild $I = [I(x,y)]$, z.B. mit $p_I(I) = 1/256$ für $I = 0, \dots, 255$

$$H = - \sum_{I=0, \dots, 255} p_I(I) \cdot \log_2(p_I(I)) = 256 \cdot (-1/256 \cdot \log_2(1/256)) = -1 \cdot -8 = 8$$

Maximierung der Entropie

Durch die Maximierung wird eine Kontraststeigerung erreicht. Die Entropie ist maximal, wenn alle Intensitätswerte dieselbe Häufigkeit annehmen.

1. normalisiertes Histogramm erzeugen

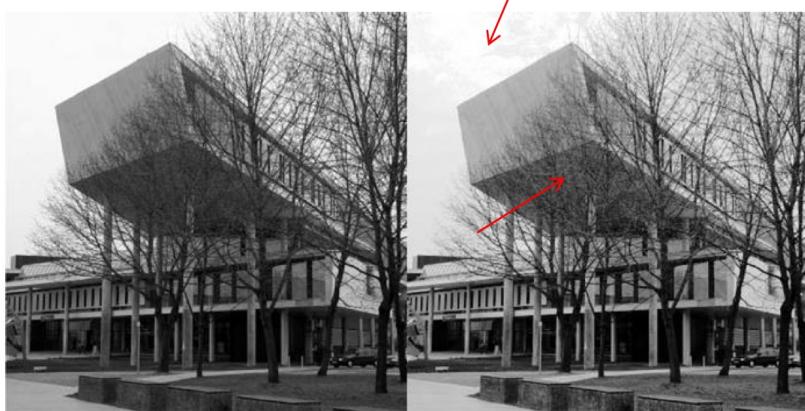
→ Intensitätswerte werden kontinuierlich auf $[0,1]$ skaliert

2. akkumuliertes Histogramm mit der Transferfunktion erzeugen

$$T_H(I) = \int_0^I p_I(i) di$$

3. tatsächliche Grauwerte erzeugen durch Histogrammlinearisierung

$$T_H(I) = \left[N_G \cdot \sum_{i=0}^I p_i(i) \right]$$



Histogramm erzeugen

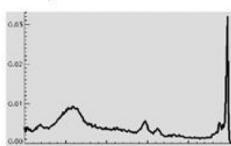
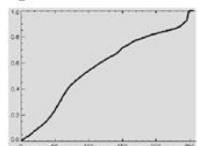


Abbildung ausführen



2. Bildstörungen und lineare Operatoren

Definition: Deterministische Einflüsse

Diese Bildstörungen basieren auf berechenbaren und wiederholbaren Einwirkungen bei der Bildaufnahme.

Diese können ggf. parametrisiert und durch lineare Operatoren rückgängig gemacht werden.

Beispiel:

Bewegungsunschärfe wird erzeugt, wenn die Kamera während der Belichtungszeit bewegt wird.

Modell: Annahme: Bildfläche der digitalen Kamera wird für 30 ms belichtet, die Kamera wird in dieser Zeitspanne um $d = 2,5 \text{ cm}$ in y-Richtung verschoben

Der Einfachheit halber sollen sich alle Szenenobjekte im Abstand $z = 1 \text{ m}$ zum Linsenzentrum befinden. Die Brennweite der Linse betrage $f = 8 \text{ mm}$.

Die Pixelgröße sei $0,04 \text{ mm} \times 0,04 \text{ mm}$.

Es folgt eine Bewegungsunschärfe durch die Verschiebung d_v (s. Abb.):

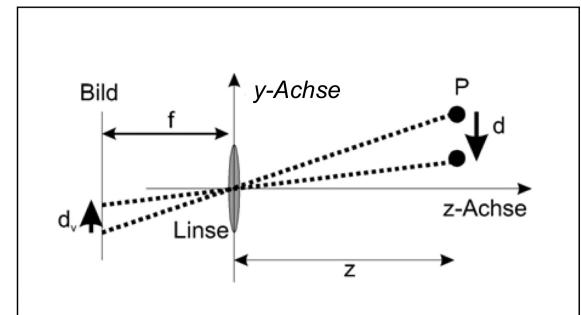
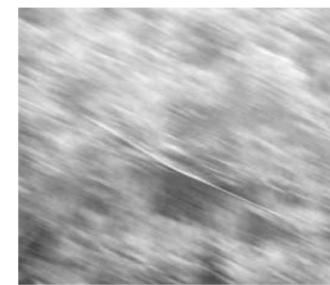
Aus $\frac{d_v}{d} = \frac{f}{z}$ folgt:

$$d_v = d \cdot \frac{f}{z} = 25 \cdot \frac{8}{10^3} = 0,2 \text{ mm} = 5 \text{ Pixel}$$

Rausfilterung mit Hilfe einer Konvolutionsmaske anhand der obigen Erkenntnisse:

$$f(u, v) = \begin{bmatrix} 0 & 0 & 0,2 & 0 & 0 \\ 0 & 0 & 0,2 & 0 & 0 \\ 0 & 0 & 0,2 & 0 & 0 \\ 0 & 0 & 0,2 & 0 & 0 \\ 0 & 0 & 0,2 & 0 & 0 \end{bmatrix}.$$

→ Wiener-Filter



Definition: Stochastische Einflüsse

Eine Bildstörung, die das Bild in statistisch beschreibbarer Weise verändert.

Diese kann näherungsweise rückgängig gemacht werden, indem die Parameter einer statistischen Beschreibung approximiert werden.

Beispiele:

1 Quantenrauschen:

Quanteneffekte bedingen, dass Photonen sich nicht geradlinig ausbreiten und so an der falschen Stelle im Bild gemessen werden.

2. Photonerauschen:

Photonerauschen beschreibt das Poisson-verteilte Quantenrauschen, welches durch eine Gauß-Verteilung approximiert wird.



Modell: Bild wird als additive Überlagerung von Signal und Rauschen interpretiert:
Signal I_u = regulär verhaltende Photonen
Rauschen η = zufallsverteilte Störung

$$I(x,y) = I_u(x,y) + \eta(x,y)$$

$\eta(x,y)$ = Gaußsche Normalverteilung mit Erwartungswert Null

Schätzung des Rauscheinflusses erfolgt durch die Bestimmung der Varianz σ^2 in einem homogenen Bildbereich B durch die korrigierte Stichprobenvarianz

$$\sigma^2 = \frac{1}{|B|-1} \cdot \sum_{p \in B} (I(p) - I')^2$$

$I_u(p) = I'$: ungestörter Intensitätswert

3. Impulsrauschen

Einzelne Pixel werden entweder zum höchsten oder niedrigsten Intensitätswert verändert.
Es sind nur wenige Pixel betroffen ($\sim 5\%$) und in der lokalen Nachbarschaft sind meist keine weiteren betroffen.

→ schwer modellierbar



Signal-to-noise ratio (SNR)

Verhältnis zwischen Signalstärke und durchschnittlicher Stärke des Rauschens

$$\text{SNR}_{\max}(I) = I_{\max_given}/\sigma$$

$$\text{SNR}_{\text{avg}}(I) = (1/S \cdot Z) \sum_{x=0, \dots, S-1} \sum_{y=0, \dots, Z-1} I(x, y)/\sigma$$

Ist der Bildinhalt bekannt:

→ Abstand zwischen gemitteltem Intensitätswert des Bildes und des Hintergrunds

$$\text{SNR}_{\text{OB}} = \frac{|\bar{I}_{\text{obj}} - \bar{I}_{\text{Hin}}|}{\sigma}$$

Lineare Operatoren

Viele durch **deterministische Störungen** gestörte Bilder können durch **lineare Operatoren** modelliert werden:

$$\mathbf{y} = \mathbf{A} \mathbf{x}^T$$

Vektor \mathbf{x} der hintereinander geschriebenen Pixel des **ungestörten Bildes**

Vektor \mathbf{y} der hintereinander geschriebenen Pixel des **gestörten Bildes**

quadratischer Matrix \mathbf{A} , welche die **Störung** kodiert

Die Störung ist **invertierbar** und kann so durch einen linearen Operator rückgängig gemacht werden, wenn die Determinante von \mathbf{A} ungleich 0 ist.

Verschiebungsinvarianz: Operator ist unabhängig vom Ort der Anwendung.

$$O \circ f(x+a, y+b) = [O \circ f](x+a, y+b)$$

→ Operator auch bestimmbar, wenn die Ursache der Störung nicht bekannt ist

Konvolution

Operationalisiert verschiebungsinvariante, lineare Operatoren:

$$(f * g)(x, y) = \sum_{u \in D} \sum_{v \in D} f(u, v) \cdot g(x-u, y-v)$$

f: Konvolutionsfunktion

g: Funktions-/Intensitätswerte

$f(-1, 1)$	$f(0, 1)$	$f(1, 1)$
$f(-1, 0)$	$f(0, 0)$	$f(1, 0)$
$f(-1, -1)$	$f(0, -1)$	$f(1, -1)$

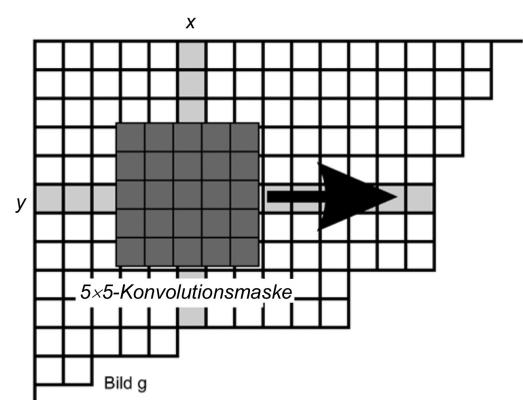
$g(-1, 1)$	$g(0, 1)$	$g(1, 1)$
$g(-1, 0)$	$g(0, 0)$	$g(1, 0)$
$g(-1, -1)$	$g(0, -1)$	$g(1, -1)$

Implementierung:

Die Konvolutionsfunktion ist diskret und begrenzt. Sie wird als Konvolutionsmaske/-kern bezeichnet mit Größe m. Beachte, dass die zu verrechneten Werte invertiert zu den Pixeln ist.

im Programm:

```
for (int x = 0; x < height; x++) {  
    for (int y = 0; y < width; y++) { ...
```



Korrelation

Funktioniert analog zur Konvolution nur dass der Filterkern nicht invertiert auf das Bild angewendet wird.

$$(f \oplus g)(x,y) = \sum_u \sum_v f(u,v) \cdot g(x+u, y+v) \text{ mit } u,v = -(m-1)/2, \dots, (m-1)/2.$$

→ korrespondiert ggf. besser mit dem ursprünglichen Bild

Lineare Filter zur Rauschunterdrückung / Glättung

Prinzip:

Da der Erwartungswert des Rauschens gleich Null ist, kann das ungestörte Signal durch einen geschätzten Erwartungswert des gestörten Signals approximiert werden.

$$E(I(x,y)) = E(I_u(x,y)) + E(\eta(x,y)) = E(I_u(x,y)) = I_u(x,y)$$

Dies geschieht entweder durch eine Bilderserie der selben Szene oder, wie im Folgenden, durch Betrachtung bzw. Mittelung der Intensitätswerte der direkten Nachbarschaft, angenommen die Intensitätswerte der Nachbarschaft sind gleich im ungestörten Bild.

Mittelwertfilter

Berechnet durch Konvolution den Mittelwert der benachbarten Pixel.

$$f_{M,m}(u,v) = \frac{1}{m^2}$$

3x3-Mittelwertfilter:

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

5x5-Mittelwertfilter:

0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04
0.04	0.04	0.04	0.04	0.04

Anmerkungen:

- Desto größer der Filter wird, desto unrealistischer die Annahme, dass die Nachbarschaft gleich ist.
- Kanten werden mehr und mehr unscharf
- starke Unschärfe beim Impulsrauschen

Gauß-Filter

Ein nach einer Gauß-Funktion gewichteter Filter erreicht eine optimale Unterdrückung des Photonenrauschen

Eindimensional:

$$f_{G,\sigma}(u) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(u^2)/2\sigma^2}$$

Zweidimensional:

$$f_{G,\sigma}(u,v) = \frac{1}{2\pi\sigma^2} e^{-(u^2+v^2)/2\sigma^2}.$$

- Kann auch durch eine hintereinander Ausführung der Konvolution mit dem eindimensionalen Filter erreicht werden (Separierbarkeit)
- Werte müssen durch die Summe aller Gewichte normiert werden

Optimale Größe des Konvolutionskerns:

$$m = 2 \cdot \lceil 3\sigma \rceil + 1$$

→ Gauß-Funktion wird hinreichend gut approximiert

Binomialfilter

Gute Approximation des Gauß-Filters, der effizient mit ganzzahligen Operationen berechnet werden kann.

Der eindimensionale Filter ergibt sich aus den Binomialkoeffizienten und wird dann auf zwei Dimensionen erweitert.

Eindimensional: $b_i^k = \binom{k}{i}$

Normierung durch $\frac{1}{2^k}$:

$$B^1 = \frac{1}{2} (1 \ 1), \quad B^2 = \frac{1}{4} (1 \ 2 \ 1),$$

$$B^3 = \frac{1}{8} (1 \ 3 \ 3 \ 1), \quad B^4 = \frac{1}{16} (1 \ 4 \ 6 \ 4 \ 1).$$

Zweidimensional: $B^{k,k} = B^k \times (B^k)^T$

$$B^{2,2} = \frac{1}{4} \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} \times \frac{1}{4} \begin{pmatrix} 1 & 2 & 1 \end{pmatrix} = \frac{1}{16} \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix},$$

$$B^{4,4} = \frac{1}{16} \begin{pmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{pmatrix} \times \frac{1}{16} \begin{pmatrix} 1 & 4 & 6 & 4 & 1 \end{pmatrix} = \frac{1}{256} \begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}$$

Anmerkung:

- Filter ist separabel
- $k \rightarrow \infty \Rightarrow$ Gauß-Filter
- Binomialfilter mit Größe k hat näherungsweise dieselbe Glättung, wie ein Gauß-Filter mit: $\sigma = k^{1/2}/2$.



Mittelwertfilter 5x5



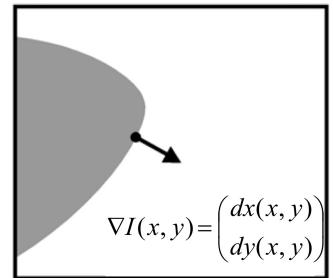
Binomialfilter 5x5

3. Hervorhebung von Kanten

Definition Kantenpixel

Ein Kantenpixel wird durch folgende Werte definiert:

1. **Positionswert:** Position des Pixels im Bild
2. **Betragswert:** Länge des Gradienten
3. **Orientierungswert:** Richtung des Gradienten



Kantenpixel sind Stellen mit großem Gradienten

Betragswert im diskreten Fall:

$$\lim_{h \rightarrow 0} \frac{I(p+h) - I(p)}{h} \approx \frac{I(p+h) - I(p)}{h} \quad \text{für kleine } h \quad \text{für } p = (x, y)$$

Konvolutionskern:

$$\frac{\partial I(x, y)}{\partial x} \approx (-1 \ 0 \ 1) * I(x, y),$$
$$\frac{\partial I(x, y)}{\partial y} \approx \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} * I(x, y).$$

Visualisierung des Orientierungswertes

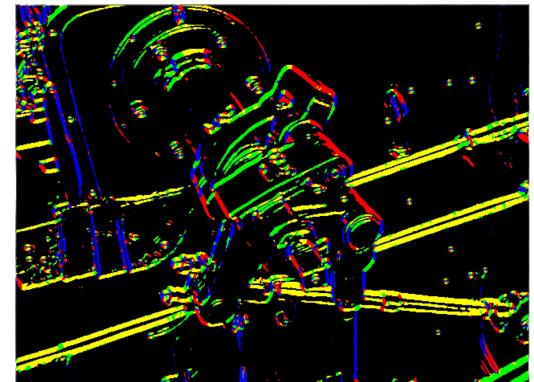
Durch das Binning werden die Gradientenrichtungen in Intervalle eingeteilt und entsprechend gefärbt

blau = $0^\circ \pm 22,5^\circ$

rot = $45^\circ \pm 22,5^\circ$

gelb = $90^\circ \pm 22,5^\circ$

grün = $135^\circ \pm 22,5^\circ$



Sobel - Filter

Der Sobel-Operator ist stabiler gegenüber lokalem Rauschen durch einen kombinierten GlättungsfILTER.

$$\text{Horizontales Sobel - Filter } S_x : \begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} (-1 \ 0 \ +1)$$

$$\text{Vertikales Sobel - Filter } S_y : \begin{pmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} = \begin{pmatrix} +1 \\ 0 \\ -1 \end{pmatrix} (1 \ 2 \ -1)$$

Betragswert. Gradientenbetrag S :

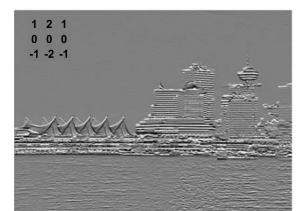
$$S \approx \sqrt{S_x(x, y)^2 + S_y(x, y)^2}.$$

Orientierungswert. Gradientenrichtung Θ :

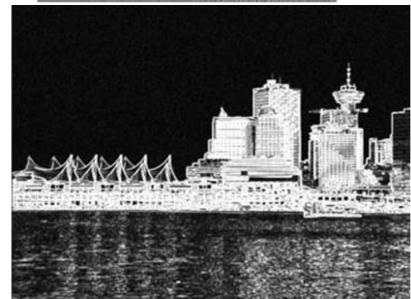
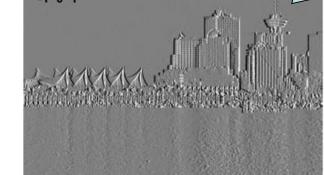
$$\Theta \approx \begin{cases} \arctan(S_y(x, y) / S_x(x, y)) & \text{für } S_x(x, y) \neq 0, \\ 90^\circ & \text{für } S_x(x, y) = 0, S_y(x, y) \neq 0. \end{cases}$$

Anmerkung:

- Korrelation und Koeffizienten erzeugen verschiedene Ergebnisse und muss daher angegeben werden. Korrelation korrespondiert ggf. besser mit den ursprünglichen Bildpixeln
- Werte müssen durch lineare Intensitätsverstärkung oder Histogrammumlinearisierung für das Intensitätsspektrum normiert werden.



In der unteren Reihe stellt Grau den Wert 0 dar, da sowohl positive als auch negative Gradienten abgeleitet werden.



Laplace-Filter

Der **Laplace-Operator** basiert auf der Summe der partiellen 2. Ableitungen. An Nulldurchgängen der 2. Ableitungen befindet sich eine Kante. Der Filter ist sehr empfindlich gegenüber Rauschen.

L_4 -Laplace-Operator:

Basiert nur auf den 2. Ableitungen in x- und y-Richtung

$$\nabla^2 I(x, y) = \frac{\partial^2 I}{\partial x^2}(x, y) + \frac{\partial^2 I}{\partial y^2}(x, y) \quad | \text{ mit: } \frac{\partial^2 I}{\partial x^2} \approx I(x+1, y) - 2 \cdot I(x, y) + I(x-1, y)$$

$$\frac{\partial^2 I}{\partial y^2} \approx I(x, y+1) - 2 \cdot I(x, y) + I(x, y-1)$$

Konvolutionskern:

$$L_4 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} \xrightarrow{\text{häufige Variante}} \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

Anwendung:

1. Es werden zwei zusätzliche Bilder erzeugt. Das eine Bild wird um einen Pixel nach links verschoben und das Andere um einen Pixel nach unten
2. Es liegt ein Nulldurchgang, eine Kante, vor, wenn der L_4 -Laplace-Operator verschiedene Vorzeichen im original Bild gegenüber einem der verschobenen Bilder aufweist

L_8 -Laplace-Operator

Nutzt alle partiellen 2. Ableitungen

$$\nabla^2 I(x, y) = \frac{\partial^2 I}{\partial x^2}(x, y) + \frac{\partial^2 I}{\partial y^2}(x, y) + \frac{\partial^2 I}{\partial x \partial y}(x, y) + \frac{\partial^2 I}{\partial y \partial x}(x, y)$$

Konvolutionskern:

$$L_8 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix} \xrightarrow{\text{häufige Variante}} \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Anwendung:

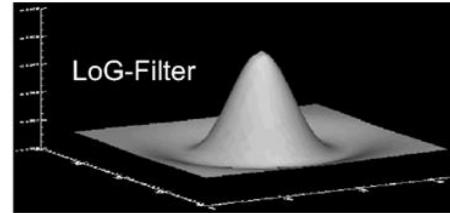
1. Es werden vier zusätzliche Bilder erzeugt. Zusätzlich zu denen beim L_4 wird noch in beide Diagonalrichtungen verschoben
2. Es liegt ein Nulldurchgang, eine Kante, vor, wenn der L_4 -Laplace-Operator verschiedene Vorzeichen im original Bild gegenüber einem der verschobenen Bilder aufweist

Laplacian-of-Gaussian-Filter

Laplace-Operator wird mit einer Glättung gekoppelt um vor einer hohen Rauschempfindlichkeit zu schützen.

$$LoG(x, y) = \nabla^2 f_{G,\sigma}(x, y)$$

$$\begin{aligned} &= \frac{\partial^2 f_{G,\sigma}}{\partial x^2}(x, y) + \frac{\partial^2 f_{G,\sigma}}{\partial y^2}(x, y) \\ &= -\frac{1}{\pi\sigma^4} \left(1 - \frac{x^2 + y^2}{2\sigma^2}\right) e^{-\frac{x^2+y^2}{2\sigma^2}}. \end{aligned}$$



Anmerkung:

- Die Filtergröße ist entsprechend der Standardabweichung zu wählen: $k = 2 \cdot \lceil 3\sigma \rceil + 1$
- Normierung durch die Summe der Filterwerte ist erforderlich
- Häufig werden Folgen von LoG-Filters verwendet mit unterschiedlichen Standardabweichungen zur besseren Kantenerkennung

Difference-of-Gaussian-Filter

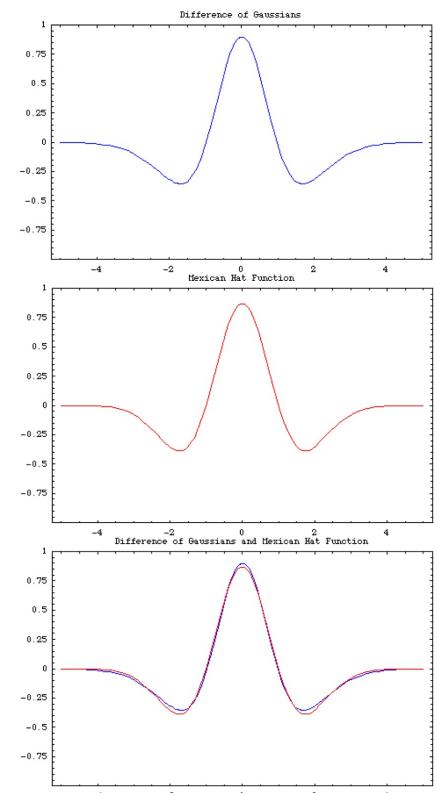
Zwei Gauß-Funktionen mit unterschiedlichen Standardabweichungen werden voneinander subtrahiert. Das Bild wird geglättet und die Nulldurchgänge können berechnet werden.

$$DoG(x, y) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}$$

$$DoG(x, y, \sigma) =$$

$$\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} - \frac{1}{2\pi K^2 \sigma^2} e^{-\frac{x^2+y^2}{2K^2\sigma^2}}$$

mit $K \sim 1,6$



Subpixelgenaue Positionsbestimmung von Nulldurchgängen

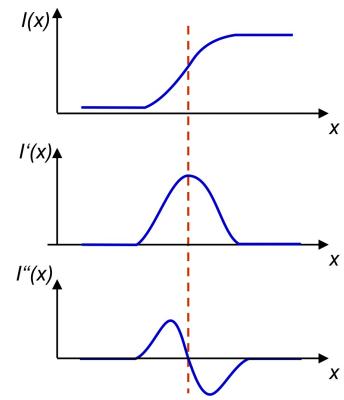
1) Berechnung von Gradienten und Nulldurchgängen

- z.B. durch Gauß-Filterung gefolgt von Sobel- und Laplace-Filterung

2) An jedem Nulldurchgang

- Bestimmung eines Umgebungsintervalls *in Gradientenrichtung* (Sobel-Operator) *um den Nulldurchgang* (ermittelt von Laplace-Operator)
- Schwerpunktbestimmung der Gradientenbeträge im Umgebungsintervall

3) Der Schwerpunkt ist die gesuchte Position des Nulldurchgangs.



4. Nichtlineare Filter und Filtern von Farbbildern

Nichtlineare Filter

Gegenüber linearen Filtern, die eine Balance zwischen Rauschunterdrückung und Kantenhervorhebung finden müssen, können nichtlineare Filter dies effizient trennen.

Rangordnungsfilter

Der Filter betrachtet ebenfalls die Nachbarschaft eines Pixels

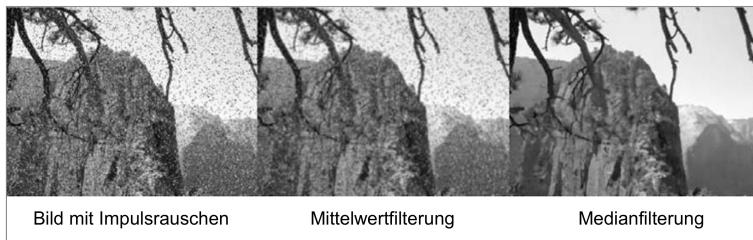
1. Sortiere alle Pixel der Umgebung gewäß ihrer Intensitätswerte
2. als Ergebnis wird der Intensitätswert eines bestimmten Rauges geliefert

Zeitkomplexität: $O(n \log n)$ für $n = m^2$

Medianfilter

Das Medianfilter ist ein **Rangordnungsfilter**, der als Ergebnis den Median liefert. Unter der Annahme, dass die Intensität konstant ist und der Erwartungswert des Rauschens Null ist, ist das Filter zur Rauschunterdrückung nutzbar.

- besonders Effektiv bei **Impulsrauschen**
- **kantenerhaltend** unter gewissen Umständen



Das Medianfilter ist **kantenerhaltend** unter folgenden Umständen

- 1) die Intensitätswerte der ungestörten Bildfunktion $I_u(x,y)$

sind auf beiden Seiten der Kante jeweils *konstant*

Alle Pixel der helleren Seite M_H werden vor die der dunkleren Seite M_D sortiert.

- 2) der **Signalabstand**, d.h. der Wertunterschied an einer Kante,

ist **größer** als die **Rauschamplitude**

Die Sortierung der Pixel M_D und M_H bleibt bei überlagertem Rauschen **erhaltenden**

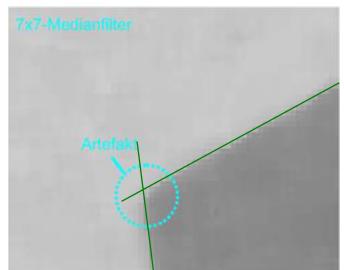
3) die Kante verläuft über die Fläche des Medianfilters gerade

M_D bzw. M_H umfasst mehr Pixel als M_H bzw. M_D , wenn das zentrale Pixel zu M_D bzw. M_H gehört.

Wird das Medianfilter klein genug gewählt, kann dies in den meisten Fällen angenähert werden.

Ausnahme sind Ecken an denen Artefakte der Ecken erzeugt werden.

Eckpositionen können durch Schnittbildung der extrahierten Kanten zurückgewonnen werden

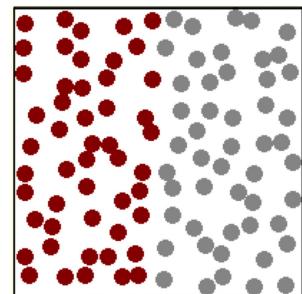


Anmerkung:

Die Größe des Medianfilters sollte eine Balance zwischen einem großen Filter zur Rauschunterdrückung und einem möglichst kleinen um die Anzahl an geradlinigen Kantenzygen zu maximieren, damit Kanten erhalten bleiben.

Diffusionsfilter

Der Diffusionsfilter orientiert sich an dem physikalischen Diffusionsprozess. Dabei wird die Intensitätsfunktion als Konzentration einer Flüssigkeit interpretiert. Die Pixel gleichen im Diffusionsprozess ihre Intensitätswerte aus.

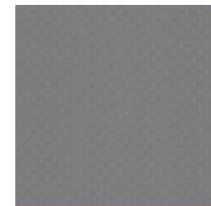


Diffusionsmodelle:

→ unterscheiden sich durch den im Diffusionsprozess angewendeten Diffusionstensor

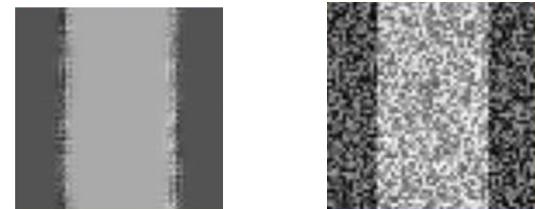
1. Isotrope homogene Diffusion

- führt zu einem Ausgleich in alle Richtungen
- Löscht alle Kanten vollständig
- kann analytisch durch eine Konvolution mit einer Gauß-Funktion mit $\sigma = (2t)^{1/2}$ berechnet werden



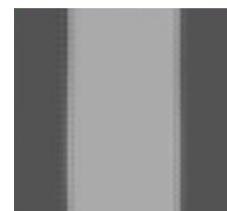
2. Isotrope inhomogene Diffusion

- die Stärke der Diffusion hängt von dem Vorhandensein einer Kante in der Nachbarschaft ab
- Erhält den Unterschied zwischen Vorder- und Hintergrund, jedoch mit verschwommenen Kanten



3. Anistrope Diffusion

- Diffusion findet an Kanten nur parallel zur Kantenrichtung statt
- Kanten bleiben erhalten und das Rauschen wird unterdrückt
- bei hoher Iterationenanzahl werden Kanten ebenfalls geglättet



Diffusionsprozess (pro Iteration):

1. Approximiere Intensitätsgradienten durch Differenzenbildung:

$$\partial I(x,y,t)/\partial x = I(x+1,y) - I(x-1,y) \longrightarrow \begin{pmatrix} -1 & 0 & 1 \end{pmatrix}$$

$$\partial I(x,y,t)/\partial y = I(x,y+1) - I(x,y-1) \longrightarrow \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$$

2. Berechne Diffusionstensor \mathbf{D} :

Isotrope homogene Diffusion:

$$\mathbf{D} = \begin{pmatrix} \varepsilon_0 & 0 \\ 0 & \varepsilon_0 \end{pmatrix} \text{ bzw. } \varepsilon_0 \quad \text{Die Diffusion hängt nur von einem konstanten Shalar ab}$$

Intensitätsveränderung: $\partial u(x,y,t)/\partial t = \varepsilon_0(\partial u/\partial x + \partial u/\partial y) = \text{div } \mathbf{j}(x,y,t)$

→ dadurch fällt die weitere iterative Lösung weg, springe zu 6.

Isotrope inhomogene Diffusion:

$$\boxed{\mathbf{D} = \begin{pmatrix} \varepsilon(\|\nabla u\|^2) & 0 \\ 0 & \varepsilon(\|\nabla u\|^2) \end{pmatrix} \text{ mit } \varepsilon(\|\nabla u\|^2) = \varepsilon_0 \frac{\lambda^2}{\|\nabla u\|^2 + \lambda^2} \quad \nabla u(x,y,t) = (\partial u/\partial x, \partial u/\partial y)}$$

Parameter λ :

großes λ : Diffusion ist schwach von ∇u abhängig
→ homogene Diffusion

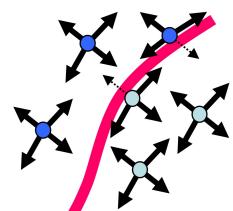
kleines λ : Diffusion ist schwach in Gebieten mit kleinen Gradienten

gutes λ : Diffusion erfolgt nur in Objektflächen und Kanten bleiben erhalten

Anistrophe Diffusion:

$$\mathbf{D} = \begin{pmatrix} e_{1,1} & e_{2,1} \\ e_{1,2} & e_{2,2} \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} e_{1,1} & e_{1,2} \\ e_{2,1} & e_{2,2} \end{pmatrix}$$

mit Eigenwerten λ_1 und λ_2 und Eigenvektoren $(e_{1,1}, e_{1,2})^\top$ bzw. $(e_{2,1}, e_{2,2})^\top$



$$1. \lambda_1 = \varepsilon(\|\nabla u\|^2) = \varepsilon_0 \frac{\lambda^2}{\|\nabla u\|^2 + \lambda^2} \quad (e_{1,1} \quad e_{1,2}) = \frac{\nabla u}{\|\nabla u\|}$$

$$2. \lambda_2 = 1 \quad (e_{2,1} \quad e_{2,2}) = (e_{1,2} \quad -e_{1,1}).$$

Bedingung an die Eigenvektoren:

1. Ein Vektor zeigt in Gradientenrichtung und hat einen kleinen Eigenwert
2. Der andere Vektor ist orthogonal zum Ersten, also parallel zur Kante und hat einen konstanten Eigenwert

3. Fluss berechnen für jedes Pixel:

$$\vec{j}(x,y,t) = (j_x(x,y,t), j_y(x,y,t)) = -\mathbf{D}(x,y,t) \times \begin{pmatrix} \frac{\partial I(x,y)}{\partial x} \\ \frac{\partial I(x,y)}{\partial y} \end{pmatrix}$$

4. Approximiere Flussgradienten:

$$\partial j_x(x,y,t)/\partial x = j_x(x+1,y) - j_x(x-1,y)$$

$$\partial j_y(x,y,t)/\partial y = j_y(x,y+1) - j_y(x,y-1)$$

5. Berechne Divergenz:

$$\operatorname{div} j(x,y,t) = \partial j_x(x,y,t)/\partial x + \partial j_y(x,y,t)/\partial y$$

6. Intensität für $t+1$ berechnen:

$$I(x,y,t+1) = I(x,y,t) - \operatorname{div} j(x,y,t) = I(x,y,t) - (\partial j_x(x,y,t)/\partial x + \partial j_y(x,y,t)/\partial y)$$

Filterung von Farbbildern

RGB-Farbmodell

Darstellung: $I_R(x,y)$, $I_G(x,y)$ und $I_B(x,y)$

Filterung: Filter werden separat auf jeden Intensitätswert angewandt.

HSI-Farbmodell

Darstellung:

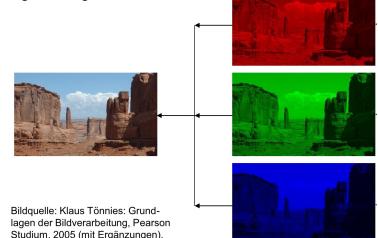
$$H = \begin{cases} \theta & \text{if } B \leq G \\ 360 - \theta & \text{if } B > G \end{cases}$$

$$\theta = \cos^{-1} \left\{ \frac{\frac{1}{2}[(R-G)+(R-B)]}{[(R-G)^2+(R-B)(G-B)]^{1/2}} \right\}.$$

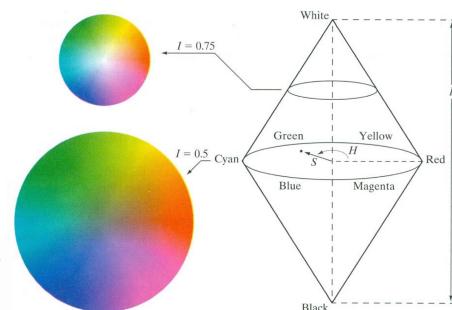
S für saturation (Sättigung, Chroma) = $1 - 3 \cdot \min\{R,G,B\}/(R+G+B)$; S = 0 für Schwarz

I für intensity (Intensität, Helligkeit) = $(R+G+B)/3$

signale zeigen.



Bildquelle: Klaus Tönnies: Grundlagen der Bildverarbeitung, Pearson Studium, 2005 (mit Ergänzungen).



Color	R	G	B	H	s	i
black	0	0	0	n.a.	0	0
red	255	0	0	0	1	0.33
green	0	255	0	120	1	0.33
blue	0	0	255	240	1	0.33
yellow	255	255	0	60	1	0.66
white	255	255	255	n.a.	0	1

Filterung: Filter werden auf lediglich einen Intensitätswert angewendet, der auch dem Graubild Intensitätswert entspricht.

→ Farben bleiben hingegen zum RGB-Farbmodell bei der Filterung besser erhalten

→ besser geeignet zur Glättung und Kantenhervorhebung



RGB Lena



red component



blue component



green component



saturation



intensity

Bildquelle: R. C. Gonzalez, R. E. Woods: Digital Image Processing (3rd Ed.), Pearson Education Intern., pp. 441ff., 2008.



HSI smoothing

5. Segmentierung

Definition: Segmentierung

Mit Segmentierung wird der Gruppierungsprozess entsprechend eines Homogenitätskriteriums von benachbarten Pixeln zu zusammenhängenden Segmenten bezeichnet.
Ist ein zentraler Bestandteil der Mid-Level Vision.

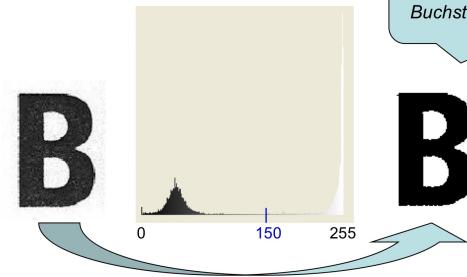
Histogrammbasierte Segmentierung/Schwellwertsegmentierung

bimodales Histogramm:

Binarisierung beschreibt die binäre Klassifikation aller Pixel über einen Schwellenwert t_B :

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

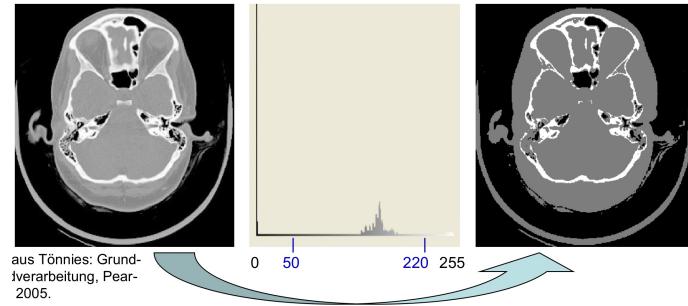
$$t_B = 150$$



Binarisierung visualisiert durch binäres Bild mit schwarz für Buchstabenpixel und weiß für Hintergrund.

multimodales Histogramm:

Histogramm besitzt $n \geq 2$ Maxima. Schwellenwerte entsprechen den $n-1$ Minima



aus Tönnies: Grundverarbeitung, Pearl 2005.

Bestimmung der Schwellenwerte:

1. interaktive Bestimmung, z.B. durch Anwendererfahrung
2. automatisierte Suche nach Minima im Histogramm

Clustering-Methode nach Otsu

Geg.: norm. Histogramm $p_I(I) = \frac{n_I}{S \cdot Z}$ für Bild $I = [I(x,y)]$

Für alle $T \in \{0, \dots, I_{\max}\}$:

1. Minimiere Intraklassenvarianz $\sigma_{\text{Within}}^2(T)$

$$\sigma_{\text{Within}}^2(T) = n_B(T) \cdot \sigma_B^2(T) + n_O(T) \cdot \sigma_O^2(T)$$

$$n_B(T) = \sum_{i=0}^{T-1} p(i)$$

$$n_O(T) = \sum_{i=T}^{I_{\max}} p(i)$$

Hier dunkle Hintergrundpixel und helle Objektpixel

$\sigma_B^2(T) = \text{Varianz der Hintergrundpixel } I(x,y) \text{ mit } I(x,y) \leq T$

$\sigma_O^2(T) = \text{Varianz der Vordergrundpixel } I(x,y) \text{ mit } I(x,y) > T$

2. Maximiere Interklassenvarianz $\sigma_{\text{Between}}^2(T)$

$$\begin{aligned} \sigma_{\text{Between}}^2(T) &= \sigma^2 - \sigma_{\text{Within}}^2(T) \\ &= n_B(T) \cdot |\mu_B(T) - \mu|^2 + n_O(T) \cdot |\mu_O(T) - \mu|^2 \end{aligned}$$

$$\sigma_{\text{Between}}^2(T) = n_B(T) \cdot n_O(T) \cdot |\mu_B(T) - \mu_O(T)|^2$$

$$n_B(T+1) = n_B(T) + n_T$$

$$n_O(T+1) = n_O(T) - n_T$$

$n_T = \text{Anteil der Pixel } p \text{ mit } I(p) = T$

$$\mu_B(T+1) = \frac{\mu_B(T) \cdot n_B(T) + n_T \cdot T}{n_B(T+1)}$$

$$\mu_O(T+1) = \frac{\mu_O(T) \cdot n_O(T) - n_T \cdot T}{n_O(T+1)}$$

Region Labeling

Anschließendes Schritt nach der Segmentierung um zusammenhängende Regionen aus Pixeln mit gleichem Klassenlabel zu suchen um Segmente zu identifizieren

Connected-Component Labeling

Eingabe: binarisches Eingabebild Source

Ausgabe: Bild Destination, in dem den Pixeln Labels durch RGB-Codes zugeordnet sind

Verfahren: Das Bild wird zeilenweise durchgegangen, wird ein Objektpixel gefunden, wird die 8-Nachbarschaft nach einem Label durchsucht. Das minimale Label wird gewählt oder ein Neues erstellt, falls kein Label vorhanden ist.

Anschließende Nachbearbeitung um äquivalente Label zusammenzufassen

Algorithmus:

```
begin
label ← 0;
all destination pixels  $d(x,y)$  are set to zero in channel 1; // channel 1 is the label channel
for all source pixels  $s(x,y)$  do row by row
    if source pixel  $s(x,y)$  is black then begin // black = object
        if 8-neighborhood of corresponding destination pixel  $d(x,y)$ 
            shows at least one pixel with label value  $l \neq 0$  in channel 1
        then begin
            find neighbor pixel with smallest label value  $l_{min}$ ;
            destination pixel  $d(x,y)$  is set to  $l_{min}$  in channel 1;
            store the equivalence between neighboring labels in a list or an array of label equivalences;
            end if-then;
        else begin
            label ← label+1;
            destination pixel  $d(x,y)$  is set to  $label$  in channel 1;
            end else;
        end if-then;
    end for;
```

2.

Connected-component labeling (second part: loop for checking for label equivalences)

```
...
select a sufficient number  $n$  of good distinguishable RGB colors  $RGB[1], \dots, RGB[n]$ ;
for all destination pixels  $d(x,y)$  do row by row
    if destination pixel  $d(x,y) = 0$  in channel 1 then
        destination pixel  $d(x,y)$  is set to white // (255,255,255) = background
    else begin //  $d(x,y) = k, k \neq 0$ 
        relabel  $d(x,y)$  with smallest equivalent label  $k'$ ;
        destination pixel  $d(x,y)$  is set to  $RGB[k']$ ;
    end for;
end; // Connected-component labeling
```



Flood Fill

rekursive Version:

Flood Fill durchsucht das Bild einfach Reihe für Reihe nach Objektpixeln

Sobald ein nicht gelabeltes Objektpixel gefunden wird, wird diesem ein neues Label zugeordnet

Diese Labelzuordnung erfolgt nun rekursiv für alle Pixel der 8-Nachbarschaft

iterative Version:

```
function flood_fill (binary image  $I_{bin}$ )
 $I \leftarrow 0$ 
for all image pixels  $p$  do
    if  $p$  is a black object pixel and not labeled then
         $I \leftarrow I + 1$ ;
         $Q \leftarrow [p]$ ;
        while ( $Q$  is not empty) do
             $p_{buffer} \leftarrow pop_{last}(Q)$ 
            if  $p_{buffer}$  is a black object pixel and not labeled then
                label  $p_{buffer}$  with  $I$ ;
                for all  $p_{neighbor}$  in  $N_8(p_{buffer})$  do pushlast( $Q$ ,  $p_{neighbor}$ );
        end;
```

Segmentierung nach Homogenitätshriterien

Zur Segmentierung wird ein allgemeines **Homogenitätshriterium** verwendet, was die Zusammengehörigkeit von Pixeln untereinander definiert. Pixel werden zu einem Segment zusammengefasst wenn sie dieses Kriterium erfüllen.

Region Merging

Regionenadjazenz graph (RAG)

Knoten: Repräsentieren Regionen und enthalten den Wert zum Berechnen des Homogenitätshriteriums

Kanten: Verbinden Knoten, wenn die entsprechenden Regionen benachbart sind. Der Wert entspricht dem Homogenitätswert, würden die beiden Regionen fusioniert werden.

Verfahren:

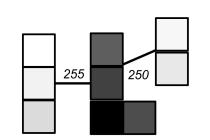
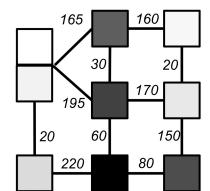
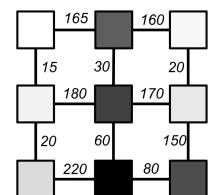
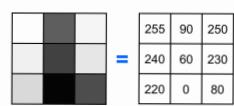
1. Stelle das Bild als Regionenadjazenzgraphen dar

2. Fusioniere solange Regionen bis keine zwei benachbarten Regionen das Homogenitätshriterium erfüllen.

Nehme immer zuerst Regionen, die das Kriterium am stärksten erfüllen

3. Extrahierte Regionen entsprechen den Segmenten

→ Ergebnis ist abhängig von der Verarbeitungsreihenfolge



Split-and-Merge

Größere Segmente, was gegenüber dem Region Merging eine Laufzeitverbesserung hervorrufen kann.

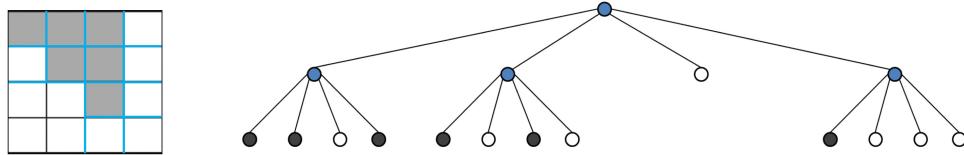
Verfahren:

1. Das gesamte Bild wird als ein Segment betrachtet

2. Splitting-Teil:

Ein Segment wird entlang der x- und y-Achse in vier gleich große Segmente zerlegt, wenn es nicht dem Homogenitätskriterium genügt, solange bis alle Segmente das Kriterium erfüllen.

Darstellung: Quadtree



3. Merging-Teil:

Betrachtete Segmente werden fusioniert, falls sie das Homogenitätskriterium erfüllen, solange bis keine Segmente mehr fusioniert werden können.

Hierarchische Clusterverfahren

Agglomeratives Clustering:

Gehört von der kleinst möglichen Unterteilung aus und bildet durch sukzessive Aggregation Cluster aus ähnlichen Objekten

→ Region Merging

Divisives Clustering:

Startet mit der Gesamtmenge, die sukzessive in homogene Cluster unterteilt wird.

Textrbasierte Segmentierung

Definition: Textur

Die Textur einer Region ist die gemeinsame Eigenschaft der Intensitätsverteilung der Region, die durch eine Bildungsregel beschrieben wird.

Formale Definitionen:

Eine **strukturelle Texturdefinition** beschreibt Textur als Zusammensetzung aus sog. Texturelementen (engl. *Texture Elements – texels*) wie z.B. linienförmigen Pixelgruppen.

Eine **statistische Texturdefinition** beschreibt Textur über Charakterisierungen bzw. Maße von Intensitätsmustern.

Eine **stochastische Texturdefinition** beschreibt Textur als Ergebnis eines parametrisierten, stochastischen Prozesses wie z.B. Gauß-verteilter Varianz.

Eine **spektrale Texturdefinition** beschreibt Textur durch die Eigenschaften in einer Repräsentation des Frequenzraumes wie z.B. bei Fourier- oder Wavelet-transformationen.

→ Texturfrequenz ist in dem Ansatz nicht sicher zu bestimmen

Textrbasierte Segmentierung nach Haralih-Maßen

Haralihsche Texturmäße

1. Co-Occurrence-Matrix $P_{\alpha, \Delta}[I_1, I_2]$

→ Beschreibt das gemeinsame Auftreten I_1 und I_2 in Bezug auf Abstand Δ und Winkel α zur horizontalen Bildachse

$$P_{\alpha, \Delta}(I_1, I_2) = 1/N \sum_{p \in R} \delta_D(I(p) - I_1) \cdot \delta_D(I(p+d) - I_2) \text{ mit } d = \Delta \cdot (\sin \alpha, \cos \alpha).$$

$$\delta_D(x) = \begin{cases} 1 & x=0 \\ 0 & \text{sonst} \end{cases} \quad \Delta = 1, \alpha = 0^\circ, 45^\circ, 90^\circ, 135^\circ.$$

→ nur in 8-Nachbarschaft

2. Normierung

$$p_{\Delta, \alpha}(I_1, I_2) \leftarrow \underset{\text{Normierung}}{\frac{1}{S}} P_{\Delta, \alpha}(I_1, I_2) \text{ mit } S = \sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} P_{\Delta, \alpha}(I_1, I_2).$$

3. Haralih-Maße

Energie/Uniformität: $\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} p_{\Delta, \alpha}^2(I_1, I_2),$

Kontrast: $\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} (I_1 - I_2)^2 p_{\Delta, \alpha}(I_1, I_2),$

Entropie: $-\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} p_{\Delta, \alpha}(I_1, I_2) \cdot \log_2(p_{\Delta, \alpha}(I_1, I_2)),$

Homogenität/
inverse Differenz: $\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} \frac{p_{\Delta, \alpha}(I_1, I_2)}{1 + |I_1 - I_2|},$

Inv. Diff.-Moment: $\sum_{I_1=0}^{I_{\max}} \sum_{I_2=0}^{I_{\max}} \frac{p_{\Delta, \alpha}(I_1, I_2)}{1 + (I_1 - I_2)^2}.$

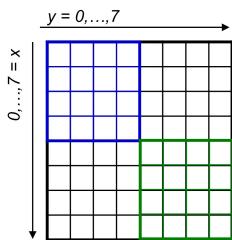
Verfahren:

Geg.: $Z \times S$ großes Bild, Texturfrequenz von n Pixeln (vert. und hor.)

1. Erzeuge Blöcke $B_{x,y}$

Oberer linker Pixel (x, y)

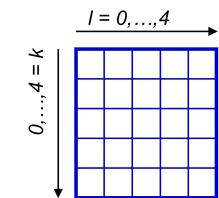
Anzahl überlappender Blöcke: $(Z-n+1)(S-n+1)$



2. Erzeuge für jeden Block den Merkmalsvektor

$$m_{x,y} = (m_1, \dots, m_n)$$

→ Besteht aus den Haralik-Massen



3. Segmentierung durch z.B. Split-and-Merge

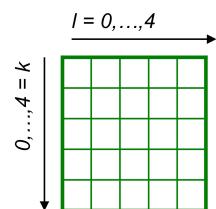
Homogenitätskriterium:

Unterschied der Haralik-Massen aller Pixel unterschreitet einen Schwellwert

$$\|m(p_i) - m(p_j)\| < d_{\min} \text{ für alle Pixel } p_i \text{ und } p_j \text{ eines Segments.}$$

4. Erzeuge ein Labelfeld L_{Block} für jeden Block

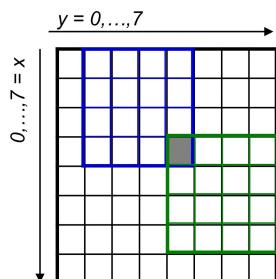
Das Label-Feld jedes Bloches weist jedem Pixel das Label des erzeugten Segmentes zu



5. Label-Wahrscheinlichkeiten berechnen

Anhand des Labelfeldes stimmt jeder den Pixel überdeckenden Block für ein Label.

Die Stimmzahl wird durch die Anzahl der überdeckenden Blöcke normiert und anschließend das mit höchster Wahrscheinlichkeit gewählt.



Segmentierung nach Diskontinuitätskriterien

Diese Ansatz zerlegen Bilder in Segmente an den Segmenträndern anhand von **Diskontinuitätskriterien**, wie der Intensitätsgradienten an Konturkanten. Dieses Vorgehen ist stabiler bei starker Variation der Segmentcharakteristik.

Segmente werden über Kantenzüge und ihren Eigenschaften über Kantenoperatoren identifiziert.

Kantenzüge

Zusammenhängende Kantenpixel, die die Segmentränder darstellen
Eigenschaften:

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

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

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

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

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

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

Edge Linking

Extrahiert zusammenhängende Kanten im Sinne von linearen Gruppen von Kantenpixeln

Verfahren:

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

1. Eigenschaft von Kantenpixeln

Binarisierung über die Intensitätsgradienten des Bildes zum Finden von Kantenpixel (Vorab Sobeloperator)

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

- 2) Markiere alle Kantenpixel als unbearbeitet.
- 3) Wähle nächstes Kantenpixel, das noch unbearbeitet ist und erkläre es als aktives Pixel p_a eines Kantenzuges k .
- 4) Wenn in der Umgebung $U(p_a)$ des aktiven Pixels p_a in Kantenrichtung (orthogonal zur Gradientenrichtung) unbearbeitete Kantenpixel p_i gefunden werden, die ähnliche Gradientenrichtungen und -beträge aufweisen, dann
 - a) markiere die Kantenpixel p_i als zum Kantenzug k gehörend,
 - b) erkläre die Kantenpixel p_i zu neuen aktiven Pixeln,
 - c) markiere das aktive Kantenpixel p_a als bearbeitet.
 - d) weiter mit Schritt 4., bis alle Kantenpixel als bearbeitet markiert sind.

2. Eigenschaft von Kantenpixeln

3. Eigenschaft von Kantenpixeln

Größe der Umgebung $U(p_a)$:

Bestimmt, wie große Lücken in Kantenzügen überbrückt werden können

Ahnlichkeit von Gradientenrichtungen und -beträgen:

Bestimmt wie gut Kanten erkannt und Rauschen ignoriert werden.

- 5) Wenn in $U(p_a)$ Kantenpixel gefunden wurden, die bereits einem Kantenzug zugeordnet sind, dann wurde eine Verzeigung von Kanten gefunden.

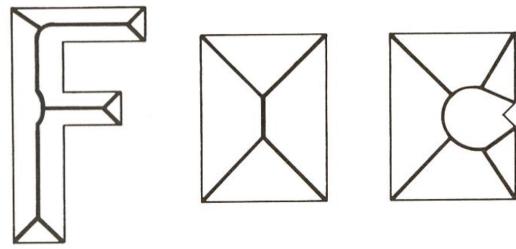
- 6) Gehe zu Schritt 3)

Terminiere, wenn alle Kantenpixel als bearbeitet markiert sind
→ Kantenpixel, die dann keinem Kantenzug zugeordnet sind, werden als Rauschen interpretiert.
→ Anschließend eventuell Kantenverdünnung durch Skelettierung

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

Skelettierung

Extrahiert aus Bildflächen eine ein Pixel breite **Skelettlinie**. Bei linienhaften Objekten, wie Kanten wird die Mittelachse approximiert. Dabei sollte die Form des ursprünglichen Objekts erhalten bleiben.



Definition: Skelett

Sei F eine planare Fläche mit Rand bzw. Grenze G und p ein Punkt in F .

Ein **nächster Grenzpunkt** g von p ist ein Punkt g in G derart, dass kein anderer Grenzpunkt g' in G existiert, dessen Abstand pg' kleiner als der Abstand pg ist.

Skeletpunkt: p ist ein Skeletpunkt, wenn es mehr als ein nächsten Grenzpunkt hat

Skelett: Menge aller Skeletpunkte

Skelettierung nach Pavlidis

Prinzip:

Iteratives Abtragen der Kantenpixel von den Rändern her

Kantenpixel werden nicht abgetragen, wenn für den Pixel eines der folgenden Muster oder die um einmal oder dreimal um 90° rotierten Muster erfüllt sind:

A	A	A
0	P	0
B	B	B

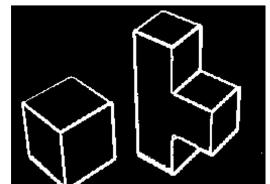
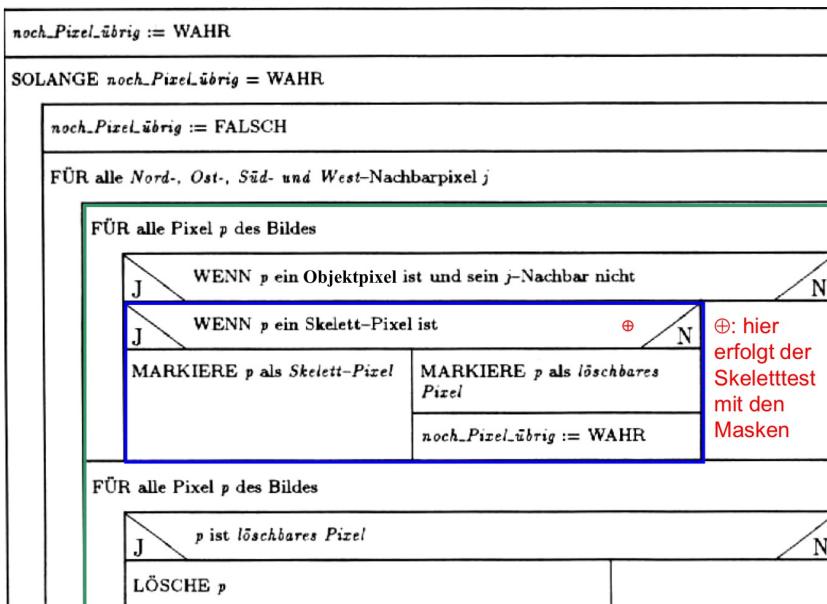
A	A	A
A	P	0
A	0	1

0 = Pixel ist nicht gesetzt = Hintergrund

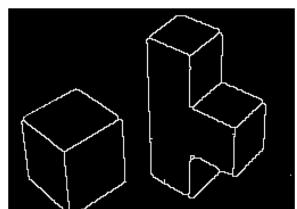
1 = Pixel ist gesetzt = Objekt

In A und B muss jeweils mind. eine 1 vorhanden sein

Verfahren:



Kantenpixel



Extrahierte Skeletpixel
(nach Pavlidis)

Canny-Operator

Optimaler Kantendetektor

1. Hohe Erkennungsrate
2. Hohe Lokalisierungspräzision
3. Eindeutigkeit

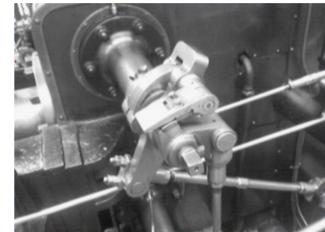
→ genaue eine Detektorantwort für jedes Kantenpixel

Verfahren:

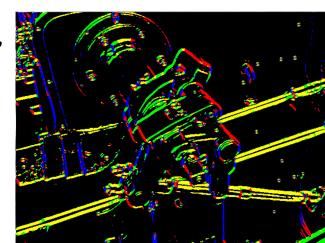
Parameter: σ , t_1 und t_2

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

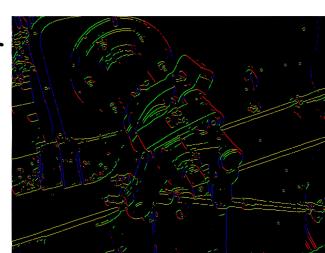
1.



2.



3.

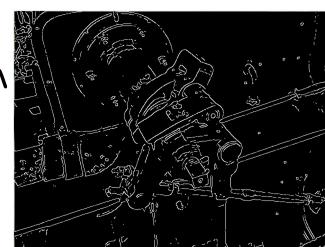


→ auch Hysterese-Verfahren

Hohes t_1 : verhindert das Erkennen von schwach ausgeprägten Kanten

Kleines t_2 : verhindert Unterbrechung von Kantenzygen

Verhältnis $t_1:t_2$: 3:1 - 7:1, abhängig von SNR



Interaktive Suche nach optimalen Kantenzügen

Suche nach Kanten mit Hilfe von Graphentheorie und finden eines kürzesten Weges.

Komplexität: $O(n \cdot \log n + m)$.

Verfahren:

1. Erzeugung eines Graphens und Kostenfunktion

Knoten: Jedes der $I \times S$ Pixel entspricht einem Knoten

Kanten: Benachbarte Pixel erhalten eine Kante im Graphen

Kostenfunktion: Jeder Knoten werden Kosten zugewiesen

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

mit

$$g_{\text{mod}} = \frac{1}{2} \cdot (|\nabla I(e)| + |\nabla I(s)|)$$

s: Start

e: Ende

$$g(k) = |\nabla I(k)|$$

Afadkostenfunktion: Kosten von Knoten s zu Knoten k

Initialisierung: $p(s) = c(s)$, sonst $p(k) = \text{max_cost} = \sum_k c(k) + 1$

2. Definiere Start- und Endpunkt der Kante

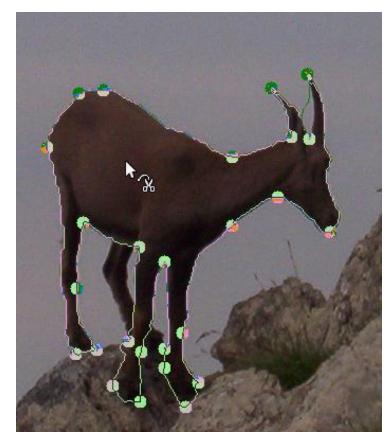
3. Berechnung des minimalen Pfads

Dijkstra-Algorithmus:

```
function Dijkstra (s,e);
    active_nodes = {s};
    p(s) = c(s);
    for all nodes v ≠ s do p(v) = max_cost;
    while v_min ≠ e do
        v_min = select_min_cost(active_nodes);
        active_nodes = active_nodes \ {v_min};
        neighbours = get_neighbours(v_min);
        for all w ∈ neighbours do
            if [ p(v_min) + c(w) < p(w) ] then
                p(w) = p(v_min) + c(w);
                active_nodes = active_nodes ∪ {w};
            endif
        endfor;
    endwhile;
    return;
end;
```

4. Bestimmung der Knoten des Pfads durch Backtracking

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



Wasserscheidentransformation

In dieser Methode wird die Gradienteninformation als Höheninformation. Mit dem **Flutungsalgorithmus** werden Bilder durch fluten der Gradientengebiete segmentiert.
→ führt leicht zu einer Übersegmentierung

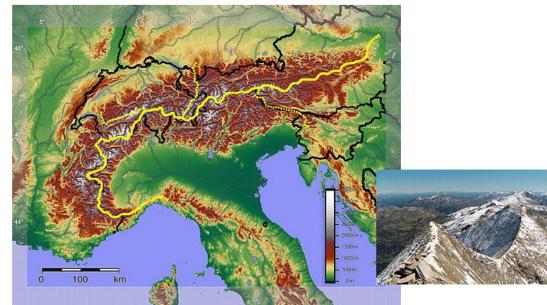
Jedem Pixel p wird dessen lokaler Gradientenbetrag als Höhe $h(p)$ zugeordnet.

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

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

Wenn p isoliert ist („Quelle“)

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



Wenn p eine Überflutungsregion erweitert

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

Wenn p Teil einer Wasserscheide ist

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

Label „Wasserscheide“
ist kein Segmentlabel

Hierarchische WST:

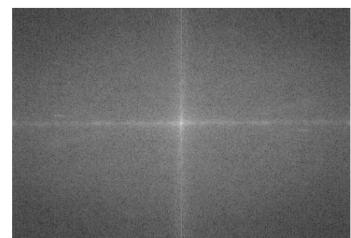
Wiederholte Anwendung des WST um Übersegmentierung zu verhindern.

7. Fourier-Transformation und Skalenräume



Definition: Ortsraum

Die Bilder, wie sie in ursprünglicher Form sind, sind Repräsentationen im Ortsraum.



Definition: Frequenzraum

Darstellung des Bildes durch einzelne Frequenzen. Das Bild wird durch eine Transformation in einer anderen Basis dargestellt

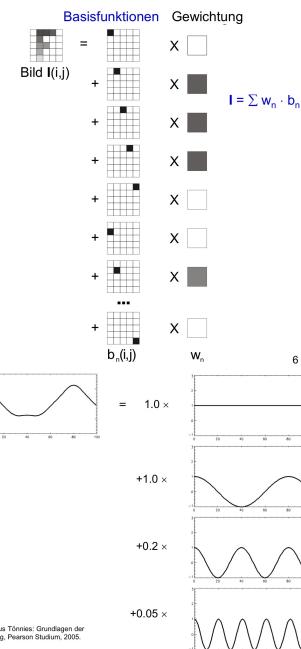
Basisfunktion

Jede Funktion $f(n)$ kann als Summe von N gewichteten Basisfunktionen b_u aufgefasst werden:

$$f(n) = \sum_{u=0}^{N-1} w_u \cdot b_u(n)$$

im Ortsraum:

Jede Basisfunktion ist ein Bild mit einem 1-Pixel, die mit einem Gewicht verrechnet wird.



im Frequenzraum:

Jede Basisfunktion entspricht einer Frequenz mit Amplitude 1, die mit einem Skalar gewichtet wird.

Transformation: Projektion auf eine neue Basis



Theorem: Jede orthogonale Basis kann durch Rotation (und Translation) aus jeder anderen Basis für die gleiche Funktion erzeugt werden.

$$\begin{aligned} T_b[f](n) &= \sum_{u=0}^N f(n) \cdot b_u(n) \\ b_1(n) &= (\cos \gamma \quad \sin \gamma) \\ b_2(n) &= (-\sin \gamma \quad \cos \gamma) \end{aligned}$$

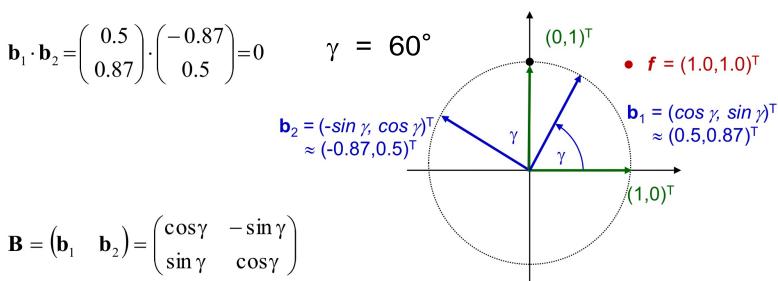


Vektorielle Darstellung:

$$\vec{f} = (f(0) \ f(1) \ \dots \ f(N-1))$$

$$\mathbf{B} = \begin{pmatrix} b_0(0) & b_1(0) & \dots & b_{N-1}(0) \\ b_0(1) & b_1(1) & & \dots \\ \dots & & & \\ b_0(N-1) & \dots & & b_{N-1}(N-1) \end{pmatrix}$$

$$\text{Transformation: } \vec{f}' = \vec{f} \times \mathbf{B}$$



$$\mathbf{f}' = \mathbf{f} \cdot \mathbf{B} = (1 \ 1) \cdot \begin{pmatrix} 0.5 & -0.87 \\ 0.87 & 0.5 \end{pmatrix} = (1.37 \ -0.37)$$

Inverse Transformation:

Transformation ist invertierbar \Leftrightarrow

1. Die Basisfunktion bildet eine orthogonale Basis

$$[b_1 \cdot b_2](n) = \sum_{n=0}^{N-1} b_1(n) \cdot b_2(n) = 0$$

2. Die Anzahl der Basisvektoren ist gleich

Invertierte Transformation = transponierte Matrix

Vollständige Basen für den Frequenzraum

1. Seien $\cos(u_1 \cdot n)$ und $\cos(u_2 \cdot n)$ mit Basisfrequenz u_0 definiert an diskreten Orten $n = 0, \dots, N-1$

1. Gilt $u_0 = 2\pi/N \Rightarrow \cos(u_1 \cdot n)$ und $\cos(u_2 \cdot n)$ sind orthogonal

2. Gilt $\frac{u_1}{u_0} = N - \frac{u_2}{u_0} \Rightarrow \cos(u_1 \cdot n) = \cos(u_2 \cdot n)$ für alle $n = 0, \dots, N-1$

\Rightarrow nicht genug Kosinusfunktion um alle Wellenlängen darzustellen

\rightarrow nicht vollständig

2. $\cos(u \cdot n) + i \cdot \sin(u \cdot n)$

$$\vec{b}_u(n) = [b_{u,\cos} \ b_{u,\sin}] = [\cos(n \cdot u \cdot 2\pi/N) \ \sin(n \cdot u \cdot 2\pi/N)]$$

Durch die zusätzliche Sinusfunktion können bis zu N unterschiedliche Wellenlängen dargestellt werden. Spannt N -dimensionalen Raum auf

\rightarrow Vollständig

Fourier-Transformation

Transformation in eine Darstellung im Frequenzraum durch Basisfunktionen, die unterschiedliche Frequenzen repräsentieren

Basisfunktionen: $\vec{b}_u(n) = [b_{u,\cos} \ b_{u,\sin}] = [\cos(n \cdot u \cdot 2\pi/N) \ \sin(n \cdot u \cdot 2\pi/N)]$

Umgewandelt (Eulersche Formel):

$$b_u(n) = e^{-i \cdot n \cdot u \cdot 2\pi/N} \text{ mit Frequenzfaktoren } u = 0, \dots, N-1$$

Transformation: $F(u) = \frac{1}{\sqrt{N}} \cdot \sum_{n=0}^{N-1} f(n) \cdot \exp(-i \cdot n \cdot u \cdot 2\pi/N), u = 0, \dots, N-1$

Rücktransformation: $f(n) = \frac{1}{\sqrt{N}} \cdot \sum_{u=0}^{N-1} F(u) \cdot \exp(i \cdot n \cdot u \cdot 2\pi/N), n = 0, \dots, N-1$

Anwendung in Bildern:

für $M \times N$ -Bilder:

$$F(u, v) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) \cdot e^{-i \cdot 2\pi \left(\frac{um}{M} + \frac{vn}{N} \right)}$$
$$f(m, n) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) \cdot e^{i \cdot 2\pi \left(\frac{um}{M} + \frac{vn}{N} \right)}$$

für $M \times M$ -Bilder:

$$F(u, v) = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m, n) \cdot e^{-i \cdot \frac{2\pi}{N} \cdot (um + vn)}$$
$$f(m, n) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) \cdot e^{i \cdot \frac{2\pi}{N} \cdot (um + vn)}$$

Komplexität: Berechnungsaufwand der FT für ein Pixel: $O(N^2)$

Berechnungsaufwand der FT für gesamtes Bild: $O(N^4)$

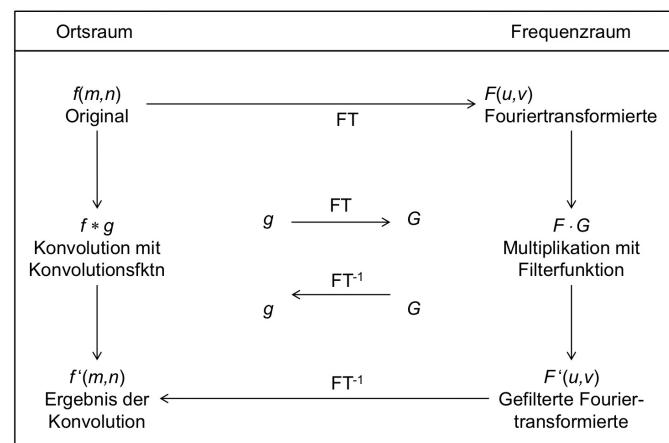
Fast Fourier-Transformation

Die FFT basiert auf einer Divide-and-Conquer-Strategie unter Nutzung der Periodizität ($F(u) = F(u+N)$, $f(n) = F(n+N)$), Symmetrie ($F(u) = {}^*F(-u)$, $f(u, v) = {}^*F(-u, -v)$) und Separabilität (2D-FT und 2D-FT⁻¹ als Produkt von jeweils zwei 1D-FTs)

- Berechnungsaufwand der FFT für ein Pixel: $O(\log N)$
- Berechnungsaufwand der FFT für gesamtes Bild: $O(N^2 \log N)$

Vorteil:

Komplexe Konvolution im Ortsraum kann durch einfache Multiplikation im Frequenzraum ersetzt werden.

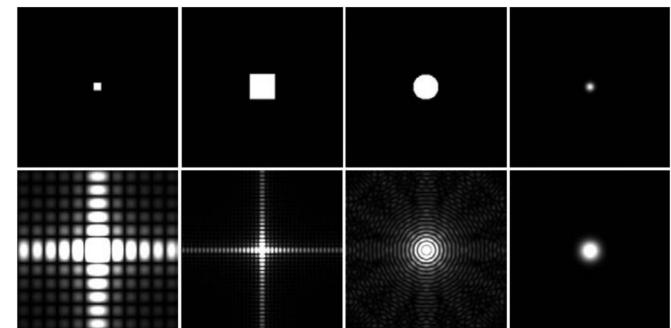
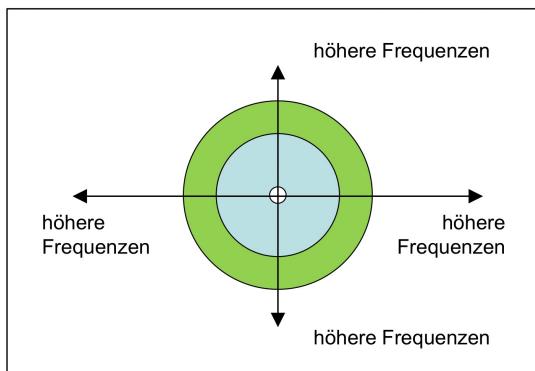


Vergleich Bilder im Orts- und Frequenzraum

Darstellung des Amplitudenspektrums

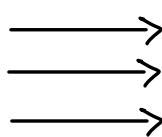
Betrag der FT:

$$|F(u, v)| = \sqrt{(Re(F(u, v)))^2 + (Im(F(u, v)))^2}$$



Originalbild

Horizontale Strukturen
Vertikale Strukturen
Scharfe Kanten



Fourier-Spektrum

Vertikale Komponenten
Horizontale Komponenten
viele hohe Frequenzen

Filter im Frequenzbereich

Tiefpass: tiefe Frequenzen passieren,
hohe Frequenzen werden reduziert
oder eliminiert

Ideal: unter dem Cut Off wird nicht
abgeschwächt, höhere Frequenzen
werden eliminiert.

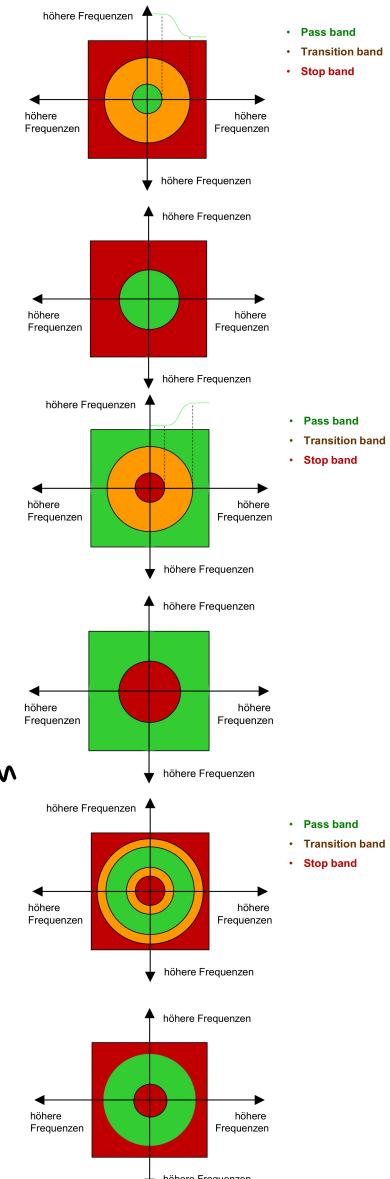
→ Glättungsfilter

Hochpass: hohe Frequenzen passieren,
tiefe Frequenzen werden reduziert
oder eliminiert

Ideal: über dem Cut Off wird nicht
abgeschwächt, tiefere Frequenzen
werden eliminiert.

→ Hervorhebung von Konturen/Konteninformation

Bandpass: Kombination aus Tief- und Hochpass-
filter



Multiskalenstrategien

Da relative Kriterien in der Segmentierung auf verschiedenen Skalierungen verschiedene Ergebnisse zeigen, versucht der Multiskalen-Ansatz Merkmale oder Segmente in unterschiedlichen Auflösungsstufen bzw. Frequenzbändern zu extrahieren.

Implizite Multiskalenstrategien

- 1 Split-and-Merge im Segmentierungsalgorithmus
- 2 hierarchische Wasserscheidenttransformation

Explizite Multiskalenstrategien

Bilden des Originalbild in Multiskalenräume ab mit verschiedenen Detailstufen

Gauß-Pyramide

G_0 = Originalbild $I[x,y]$

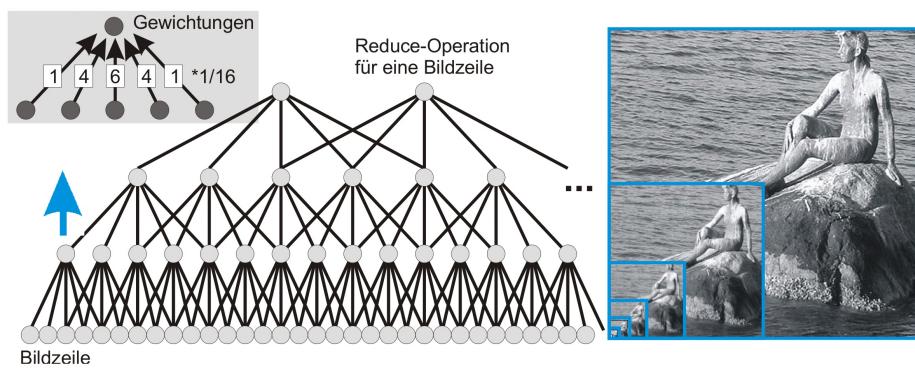
G_{k+1} = $\text{Reduce}(G_k)$
für Stufen $k = 0, 1, \dots, r$

1. reduce - Operation

→ Anzahl an Pixeln wird in jeder Spalte und Zeile fortlaufend halbiert

Filtrierung: Gaußfilter: $\frac{1}{16}(0.87 \ 3.91 \ 6.44 \ 3.91 \ 0.87)$

Binomialfilter: $\frac{1}{16}(1 \ 4 \ 6 \ 4 \ 1)$



5

2. expand - Operation

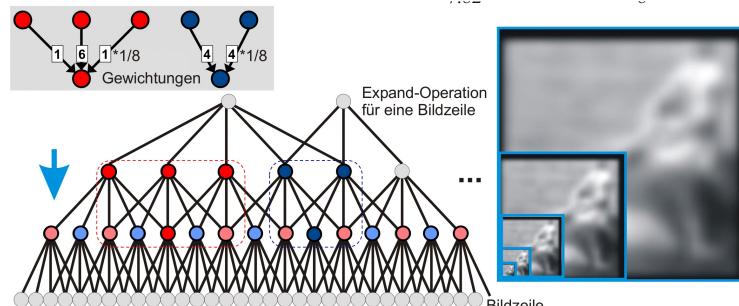
→ zum Vergleich müssen die reduzierten Stufen wieder auf die Vorherige abgebildet werden

Filtrierung: Pixelorte, die auf beiden Skalierungsstufen existieren:

$$\frac{1}{8.18}(0.87 \ 6.44 \ 0.87) \text{ bzw. } \frac{1}{8}(1 \ 6 \ 1)$$

Pixelorte, die nur auf der vorherigen Skalierungsstufe existieren:

$$\frac{1}{7.82}(3.91 \ 3.91) \text{ bzw. } \frac{1}{8}(4 \ 4)$$



5

Laplace-Pyramide

Jede Stufe k entspricht dem Differenzbild zwischen G_k und der Expansion des nächstrreduzierten Bildes
→ Difference-of-Gaussian verwenden um Laplacian-of-Gaussian zu approximieren

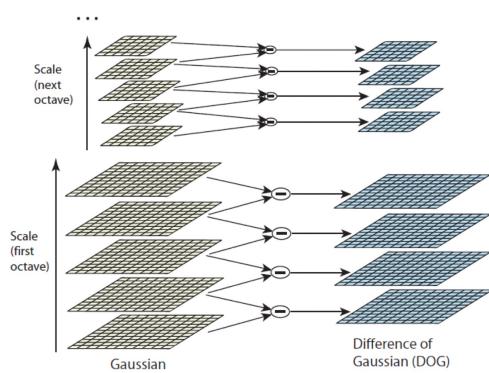
$$L_r = G_r$$

$$L_k = G_k - \text{Expand}(G_{k+1})$$

für Stufen $k = 0, 1, \dots, r-1$



Gauß- und DoG-Hierarchien bieten einen skaleninvarianten Ansatz zum Finden und Beschreiben von Interest Points.



8 Skaleninvariante Erkennung und Beschreibung von markanten Punkten

Definition: **Markanter Punkt** (interest point)

Ein markanter Punkt hat eine wohldefinierte Position und weist eine spezielle Eigenschaft auf, die robust identifiziert werden kann:

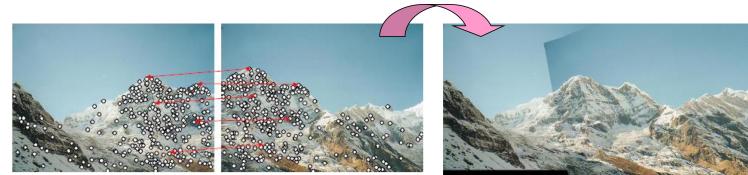
Ecken

isolierte Punkte mit lokalem Intensitätsmaximum oder -minimum

Kreuzungspunkte

Linienenden

lokale Krümmungsmaxima bei gekrümmten Linien oder Kanten



Meist werden Ecken als markante Punkte extrahiert mit Hilfe von Eckenoperatoren

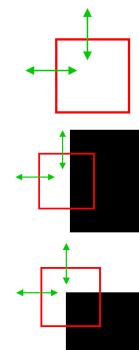
→ Die meisten Eckenoperatoren extrahieren markante Punkte und nicht ausschließlich Ecken

Harris Corner Detector

Extrahiert rotationsinvariant, aber nicht skaleninvariant, **wichtige Punkte**

Prinzip: Bewegen eines Fensters und Interpretation der Intensitätsverteilung

1. **homogener Bildbereich:** kaum Veränderung
2. **Kante:** geringe Veränderung entlang der Kante
große Veränderung senkrecht zur Kante
3. **Ecke.** große Veränderung in jede Richtung



Verfahren:

1. Aufstellen der Harris Matrix

$$\mathbf{A} = \sum_{u,v} w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix}$$

$$\text{mit: } w(u,v) = e^{-\frac{(u^2+v^2)}{2\sigma^2}} \quad I_x = \frac{\partial I}{\partial x} \approx I * (-1,0,1)$$

$$I_y = \frac{\partial I}{\partial y} \approx I * \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$$

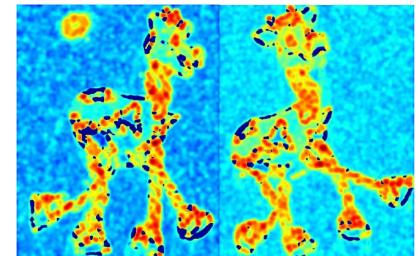


Darstellung als Funktion:

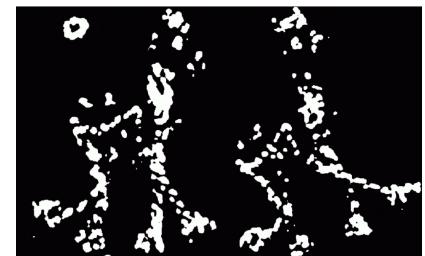
$$S(x,y) \approx \sum_{u,v} w(u,v) [I_x(u,v)x + I_y(u,v)y]^2$$

2. Berechnen der Antwortfunktion

$$\begin{aligned} R &= \det(\mathbf{A}) - \kappa \operatorname{trace}^2(\mathbf{A}) \\ &= \langle I_x^2 \rangle \cdot \langle I_y^2 \rangle - \langle I_x \cdot I_y \rangle^2 - \kappa \cdot (\langle I_x^2 \rangle + \langle I_y^2 \rangle)^2 \end{aligned}$$



Mit: $0.04 \leq \kappa \leq 0.15$



3. Optional: Binarisierung über Schwellwert

4. Optional Non-Maxima-Unterdrückung
→ siehe Canny-Operator



Multiskalen-Vergleich von Regionen

Um markante Punkte in Bildern robust zu identifizieren und zuzuordnen, ist eine skaleninvariante Erkennung von Interest Regions nötig.

→ Händisches Vergleichen von Descriptoren in verschiedenen Skalen zu aufwändig

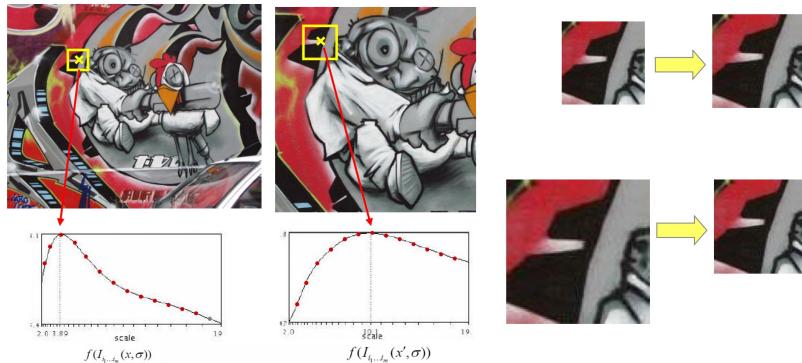


Automatische Skalensuche

Einsatz einer auf Regionen skaleninvarianten Bewertungsfunktion f , wie der Intensitätsmittelwert

Lösung: lokales Maximum der Funktion f

→ Regionengröße ist skaleninvariant



Blob Detection

Definition: Blob

Ein Blob ist eine kompakte Bildregion, die sich bzgl. einer bestimmten Eigenschaft (z.B. Farbe, Helligkeit) von der Umgebung unterscheidet.

Laplacian-of-Gaussian Filter

1. Generierung eines Skalenraums

Konvolution mit einer 2D-Gauß Funktion auf multiplen Skalen

$$g(u, v, t) = \frac{1}{2\pi t} e^{-(u^2+v^2)/2t}$$

Skalenraumrepräsentation: $G(x, y, t) = g(x, y, t) * I(x, y)$

2. Anwendung des Laplace-Operators für jede Skala t :

$$\nabla^2 G(x, y, t) = \frac{\partial^2 G}{\partial x^2}(x, y, t) + \frac{\partial^2 G}{\partial y^2}(x, y, t)$$

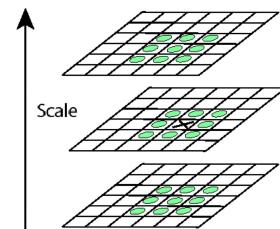
automatische Skalenwahl:

$$\begin{aligned}\nabla_{\text{norm}}^2 G(x, y, t) &= t \left(\frac{\partial^2 G}{\partial x^2}(x, y, t) + \frac{\partial^2 G}{\partial y^2}(x, y, t) \right) \\ &= \sigma^2 \left(\frac{\partial^2 G}{\partial x^2}(x, y, \sigma^2) + \frac{\partial^2 G}{\partial y^2}(x, y, \sigma^2) \right)\end{aligned}$$

→ Normierung durch Skala
→ Scale-space Blob detector

3. Erkennung von lokalen Extrema

$G(x, y, t)$ ist lokales Maximum/Minimum, wenn $\nabla_{\text{norm}}^2 G(x, y, t)$ größer/kleiner den ∇_{norm}^2 -Werten aller 26 Nachbarn ist



Maximum: dunkle Blobs mit Radius $r = \sqrt{2t} = \sigma\sqrt{2}$

Minimum: helle Blobs mit Radius $r = \sqrt{2t} = \sigma\sqrt{2}$

4. Automatische Bestimmung der charakteristischen Skala t_c eines Blobs durch das entsprechende Extremum in Skala t_c

SIFT (Scale-invariant Feature Transform)

SIFT kann Schwellungsinvariant Interest Regions identifizieren und schwellungs- und rotationsinvariante Region Descriptor erstellen.

Verfahren:

1. Interest Regions extrahieren

1. Gauß-Pyramide erstellen

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y) \text{ mit}$$

→ Konvolution

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-(u^2+v^2)/2\sigma^2}$$

2. Difference-of-Gaussian Filter anwenden

$$\begin{aligned} D(x,y,\sigma) &= (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) \\ &= L(x,y,k\sigma) - L(x,y,\sigma) \end{aligned}$$

→ Nach jeder Oktave downsampling durch Faktor 2

Parameter k:

Anzahl s an Schalen pro Oktave: $k = 2^{1/s}$

→ $s + 3$ Ebenen in der Gauß-Pyramide $L(x,y,\sigma)$ pro Oktave

$s < 3 \rightarrow$ steigende Zahl stabiler Keypoints

$s = 3 \rightarrow$ maximale Zahl stabiler Keypoints

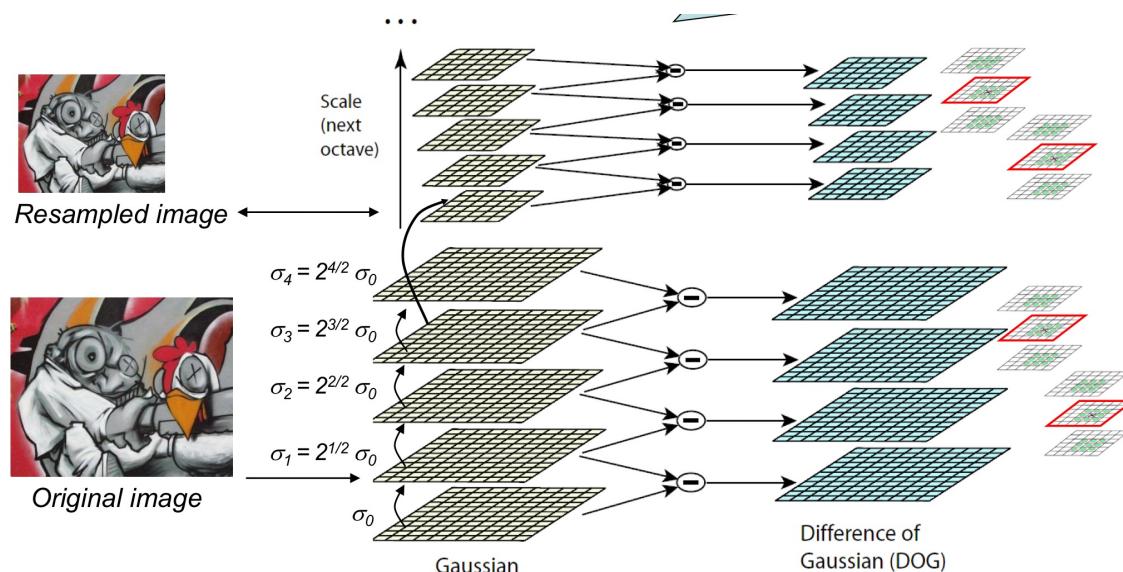
$s > 3 \rightarrow$ sinkende Zahl stabiler Keypoints

3. Lokale Extrema von $\nabla^2_{\text{norm}} G(x,y,t)$ finden

$G(x,y,t)$ ist lokales Maximum/Minimum,

wenn $\nabla^2_{\text{norm}} G(x,y,t)$ größer/kleiner den

∇^2_{norm} -Werten aller 26 Nachbarn ist



4. Post-Processing der Extrema

Berechne solange den Offset und verschiebe den Keypoint in diese Richtung, solange $\text{Offset} > 0.5$

$$\hat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2} \frac{\partial D}{\partial \mathbf{x}} \quad \text{mit } \mathbf{x} = (x, y, \sigma)^T$$

Finaler Offset wird aufaddiert

5. Eliminierung von Kantenextrema

Verwirf Kantenextrema, für die gilt:

$$\frac{\text{trace}(H_{x,y})^2}{\det(H_{x,y})} < \frac{(r+1)^2}{r} \quad \text{mit } H_{x,y} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix} \quad r = 10$$

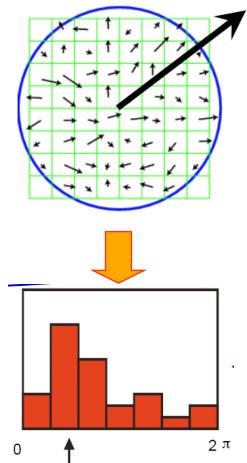
2. Keypoint Descriptor erstellen

1. Dominante Orientierung bestimmen

Beträge und Orientierung des Gradienten aller Pixel im Radius $r_\sigma = 1,5 \cdot \sigma$ in $L(x, y, \sigma)$ für Skala σ ableiten

$$\text{Betrag} := m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2}$$

$$\text{Orientg} := \Theta(x, y) = \tan^{-1}((L(x, y+1) - L(x, y-1)) / (L(x+1, y) - L(x-1, y)))$$



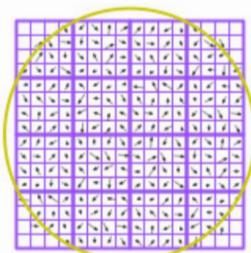
Erstelle Histogramm mit 36 Bins, Maximum ist dominante Orientierung

2. Generierung des Keypoint Descriptor

1. Leite die Gradienten der benachbarten Pixel in $L(x, y, \Theta)$ ab und gewicht diese mit einer Gaußfunktion mit Θ .

2. Gradienten in Sample-Array schreiben und in Teilregionen aufteilen

16×16 Sample-Array. $4 \times 4 = 16$ Teilregionen



3. Dominante Orientierung bestimmen, Orientierungen in 8 Bins mit Beträgen aufteilen

4. Daraus ergibt sich der Keypoint Descriptor

$16 \cdot 8 = 128$ Werte

*	*	*	*
*	*	*	*
*	*	*	*
*	*	*	*

Keypoint descriptor

9. Modellbasierte Segmentierung

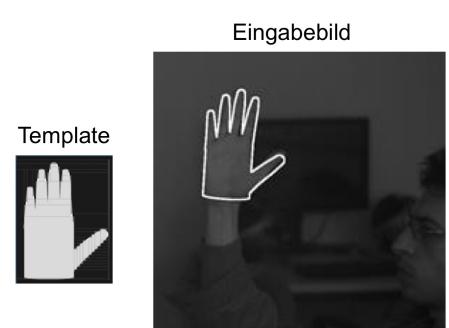
Definition: Modellbasierte Segmentierung

Bei der Segmentierung werden auf Bilder anwendbare Repräsentation eines Modells genutzt, welches eine Klasse von Szenenobjekten beschreibt. So kann ggf. Segmentierung und Objekterkennung gemeinsam geschehen.

Template Matching

Ein Muster beschreibt das gesuchte Segment und den Hintergrund in der unmittelbaren Umgebung.

Feld $t(p,q)$ mit $p = 0, \dots, P-1$, $q = 0, \dots, Q-1$



Größe des Templates:

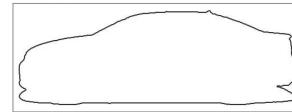
Das Template sollte groß genug gewählt werden, sodass neben dem Zielobjekt auch der Hintergrund erfasst wird um die Form des Zielobjekts zu erfassen.

Varianten

1. Intensitätswerte des Bildes
2. Gradientenbeträge
→ entspricht der Objektsilhouette



Bildquelle: Toyota Motor Corp.



Robustheit:

→ wird auch durch das gewählte Abstandsmass beeinflusst

Formtoleranz:

kleine Formabweichungen werden toleriert, solange die große Mehrheit der Pixel dem Muster entspricht

Skalierungs- und Rotationsabhängigkeit:

Template Matching ist empfindlich gegenüber Rotation und Skalierung. Kann durch das gewählte Abstandsmass kompensiert werden.

Lösung:

Parameterraum der Such wird um s (Skalierung), α (Rotation) erweitert.
→ deutlich aufwändiger Suche

Verfahren:

Suche im Bild Positionen x_k und y_k , sodass der Bildausschnitt $I_{PQ}(x_k, y_k)$ mit $x = x_k, \dots, x_k + P-1$ und $y = y_k, \dots, y_k + Q-1$ mit dem Muster t optimal übereinstimmt.

Bewertung anhand eines Abstandsmaßes

Abstandsmaße:

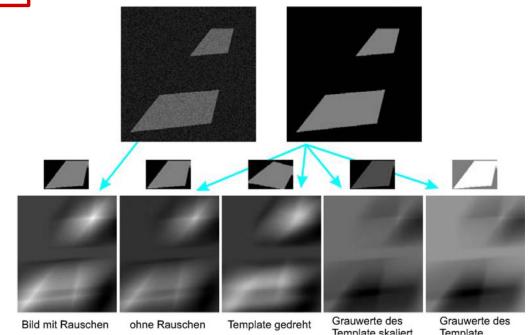
Durchschnittlicher quadratischer Abstand (msd):

$$msd(x_k, y_k) = \frac{1}{PQ} \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} (I(x_k + p, y_k + q) - t(p, q))^2$$

Durchschnittlicher Abstand der Beträge (mad):

$$mad(x_k, y_k) = \frac{1}{PQ} \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} |I(x_k + p, y_k + q) - t(p, q)|$$

- robust gegenüber Rauschen
- empfindlich gegenüber Rotation
- schlechtere Ergebnisse, wenn sich die durchschnittliche Helligkeit im Bild unterscheidet oder der Kontrast im Bild unterscheidet



Korrelationskoeffizient

$$cc_{I,t}(x_k, y_k) = \frac{\sigma_{I,t}(x_k, y_k)}{\sqrt{\sigma_I^2(x_k, y_k) \cdot \sigma_t^2(p, q)}}$$

Kovarianz:

$$\sigma_{I,t}(x_k, y_k) = \frac{1}{PQ} \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} ((I(x_k + p, y_k + q) - \bar{I}(x_k + p, y_k + q)) (t(p, q) - \bar{t}(p, q)))$$

Ergebnis:

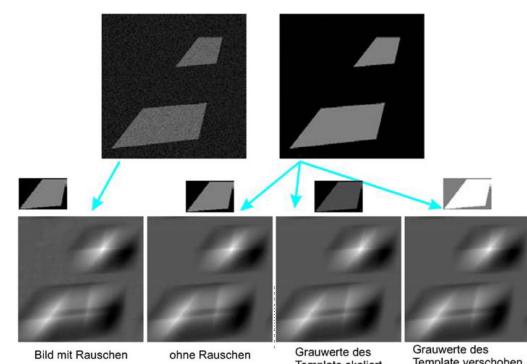
$cc_{I,t}(x_k, y_k) = 1$: Bildausschnitt und Muster sind sich sehr ähnlich mit $I_{PQ}(x_k, y_k) = s \cdot t(p, q) + d$ $s > 0$

$cc_{I,t}(x_k, y_k) = -1$: Bildausschnitt ist der invertierten Variante des Musters sehr ähnlich mit $I_{PQ}(x_k, y_k) = s \cdot t(p, q) + d$ $s < 0$.

$cc_{I,t}(x_k, y_k) = 0$: Bildausschnitt und Muster sind sich nicht ähnlich.

keine lin. Abhängigkeit besteht zw. $I_{PQ}(x_k, y_k)$ und $t(p, q)$

- robust gegenüber Rauschen, Skalierung und Verschiebung der Grauwerte
- Relation der Intensitätswerte der einzelnen Komponenten des Musters muss erhalten bleiben



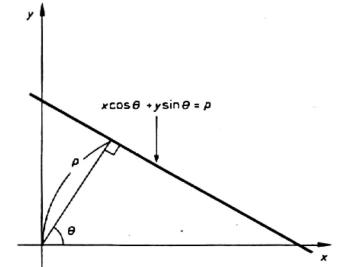
Hough-Transformation

Ahnlich zu Template Matching, nur dass Muster durch Kantenstücke, die durch Parameterbeschreibung definiert sind, beschrieben sind.

Prinzip:

Darstellung der Geraden in Hessescher Normalform:

$$x \cdot \cos(\theta) + y \cdot \sin(\theta) = \rho \quad \text{mit } (\theta, \rho)$$



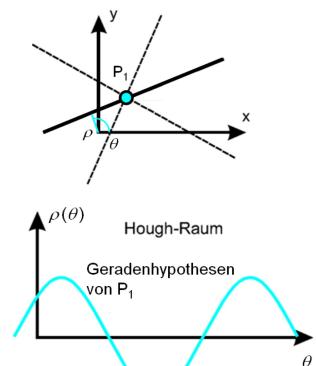
Darstellung von Kreisen:

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \quad \text{mit } (x_0, y_0, r)$$

Abstimmungsprinzip

Kantapixel votieren für alle Geraden über (θ, ρ) bzw. für alle Kreise über (x_0, y_0, r) auf denen sie theoretisch liegen können.

Die Stimmen werden im jeweiligen Hough-Raum dargestellt

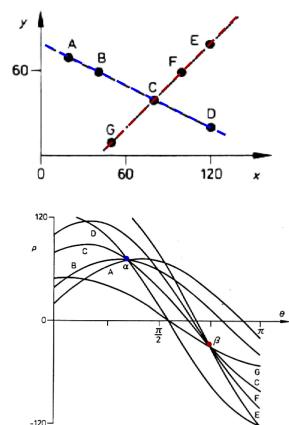


Abstimmung: $\rho(\theta) = x \cos(\theta) + y \sin(\theta)$.

Berechne $\rho(\theta)$ für alle Kantapixel
mit $0^\circ \leq \theta < 180^\circ$

Ergebnis:

Lokale Maxima der Stimmen an Stelle (ρ, θ) bzw. (x_0, y_0, r) deuten eine Gerade bzw. einen Kreis mit den Parametern an
→ Schnittpunkt der Kurven



Verfahren (für diskrete Bilddaten):

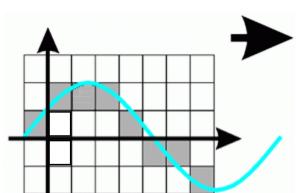
Wahl der Parameter:

Geradengleichung

Winkelparameter θ : $0 \leq \theta < \pi$

Diskretisierung: z.B. Schritte von 1°

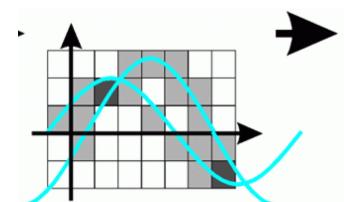
Abstandsparameter ρ : $-\sqrt{x_{\max}^2 + y_{\max}^2} \leq \rho \leq +\sqrt{x_{\max}^2 + y_{\max}^2}$



Diskretisierung: z.B. durch Pixelgröße
Kreisgleichung:

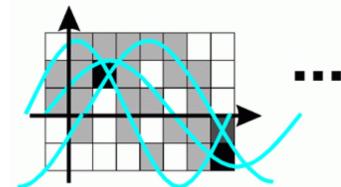
x_0, y_0 : Begrenzt durch Bildränder

r : Begrenzt durch Anwendung $[r_{\min}, r_{\max}]$



Diskretisierung erzeugt einen Hough-Raum bestehend aus Akkumulatorzellen

Überdeckt eine Kurve eine Zelle erhält diese eine Stimme



Pseudocode:

```
function discrete_HT (i:image);
    ρ_max = ( (image_width)2 + (image_height)2 )½;
    ρ_min = -1 * ρ_max;
    HoughSpace [0...π][ρ_min ... ρ_max] = 0; // all cells set to 0
    for all contour pixels p do
        for (θ = 0; θ < π, θ++) do // incrementing by 1°
            ρ = p.x · cos(θ) + p.y · sin(θ);
            if (ρ_min ≤ ρ ≤ ρ_max) then HoughSpace[θ, ρ]++;
        endfor;
    endfor;
    return;
end;
```

Vermeidung von Fehlstimmen

Ein Kantenpixel gibt 180 Stimmen ab, wobei nur eine Richtig ist. → 179 Fehlstimmen
→ kann zu Fehlinterpretationen führen

Lösung:

Kantenpixel dürfen nur für Kanten abstimmen, wenn die Gradientenrichtung mit einem gewissen Toleranzbereich orthogonal zur Kantenrichtung ist

$$\theta_p \pm \phi$$

Polarisationswinkel θ_p : wird in Gradientenrichtung gewählt

Toleranzwinkel ϕ : Abschätzung der Orientierungsunsicherheit ($5^\circ, 10^\circ$)

Lücken in Kantenzügen

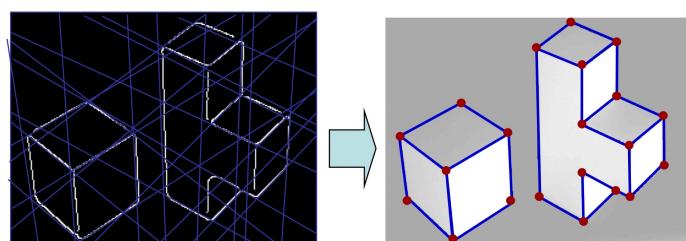
Die Hough-Transformation ist robust gegenüber Lücken, solange genügend andere Kantenpixel für die Kante votieren.

Begrenzte Kantenzüge

Die Hough-Transformation erzeugt Kanten unendlicher Länge.

Endliche Kanten werden erzeugt durch das zurückprojizieren der Kanten auf Kantenpixel (hohe Gradientenbeträge, orthogonale Gradientenrichtung).

→ Lückentoleranz einbauen



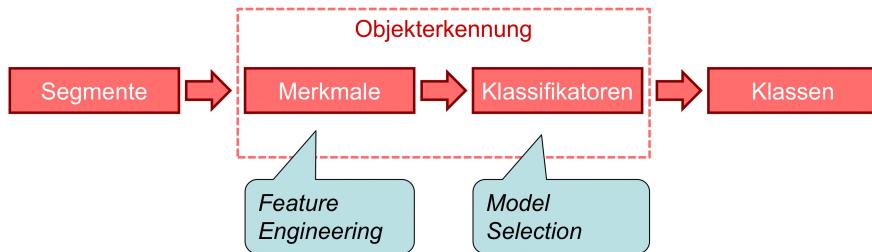
Geradenhypothesen

Validierte Linienabschnitte mit
End- bzw. Verbindungspunkten

10. Objekterkennung

Definition: Objekterkennung

Zuordnung von Segmenten zu Objektklassen durch Segmentmerkmale im Merkmalsraum

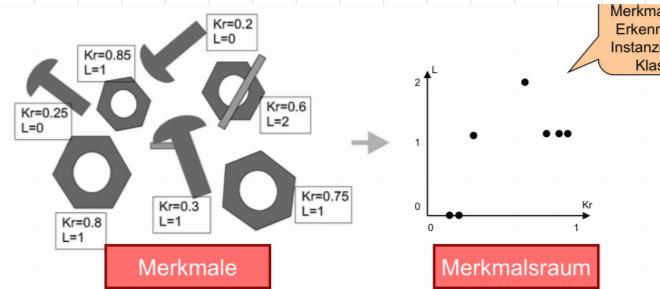


Durch Klassifikation soll Segment eine Bedeutung zugeordnet werden mit einer Qualitätsangabe der Klassifikation. Fehlzuordnungen sollen minimiert werden.

Merkmale und Merkmalsräume

Jedes Segment wird durch einen Merkmalsvektor $m = (m_1, \dots, m_M)$ beschrieben.

→ bildet auf einen Ort im Merkmalsraum ab



Wahl von Merkmalen

Sei T eine Menge von klassifizierten Segmenten nach K Klassen c_0, \dots, c_{K-1} , die T in K disjunkte Mengen T_0, \dots, T_{K-1} unterteilen. Dann rechnet sich

Within-class scatter S_w :

Durchschnittliche Varianz der Merkmalsvektoren einer Klasse sollte minimiert werden

$$S_w = \frac{1}{|T|} \sum_{k=0}^{K-1} \sum_{s \in T_k} \|(\mathbf{m}(s) - \boldsymbol{\mu}_k)\|^2$$

Between-class scatter S_b :

Mittlere Abstand zwischen den Mittelwerten der Klassen sollte maximiert werden

$$S_b = \frac{1}{n} \sum_{k_1=0}^{K-1} \sum_{k_2>k_1}^{K-1} \|(\boldsymbol{\mu}_{k_1} - \boldsymbol{\mu}_{k_2})\|^2$$

mit $\boldsymbol{\mu}_k = \frac{1}{|T_k|} \sum_{s \in T_k} \mathbf{m}(s)$ und $n = \binom{K}{2}$.

Regionenbasierte Merkmale

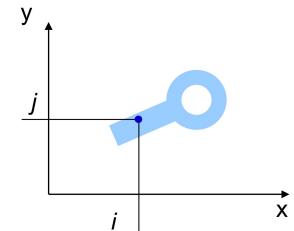
Texturen zum Charakterisieren von Segmenten (Grauwert, Farbe)
→ Haralische Texturmaße
→ durchsch. Grauwert und Varianz

Formbasierte Merkmale

Formmerkmale definieren ausschließlich über den Objektrand
 → anfällig gegenüber leichten Segmentierungsfehlern

Flächeninhalt: $F(s) = \text{Zahl aller Pixel von } s$
 $= |\{(i,j) | (i,j) \in s\}|$

→ translations- und rotationsinvariant



Randlänge: $R(s) = \text{Zahl der Pixel von } s, \text{ die mind. ein Nachbarpixel}$
 in ihrer 8-Nachbarschaft außerhalb von s haben,
 $= |\{(i,j) \in s \text{ mit } (k,l) \in N_8(i,j) \wedge (k,l) \notin s\}|$

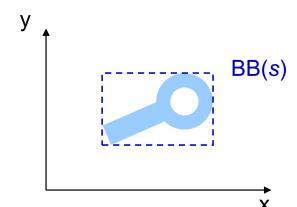
Kompattheit: $K(s) = \text{Verhältnis von Flächeninhalt von } s$
 zum Quadrat der Randlänge von s
 $= F(s) / R(s)^2$

→ gewisse Skalierungsinvarianz

Kreisähnlichkeit: $Kr(s) = \text{Verhältnis vom } 4\pi\text{-fachen Flächeninhalt von } s$
 zur Randlänge von s
 $= (F(s) \cdot 4\pi) / R(s)$

Größe der Bounding Box: $F_{BB}(s) = \text{Höhe}_{BB}(s) \cdot \text{Breite}_{BB}(s)$

→ Orientierungsabhängig



Differenz zw. Fläche und Fläche der konvexen Hülle:

$$F_{Diff_ConvHull}(s) = F_{ConvHull}(s) - F(s)$$

Schwerpunkt: $(i_\mu, j_\mu) = \text{Schwerpunktkoordinaten von } s \text{ mit } i_\mu = \frac{1}{F(s)} \cdot \sum_{(i,j) \in s} i,$
 $j_\mu = \frac{1}{F(s)} \cdot \sum_{(i,j) \in s} j.$

Trägheitsmoment durch μ :

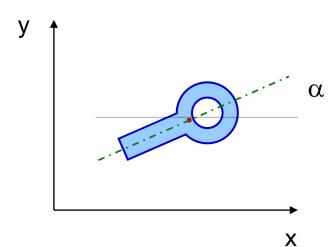
X-Richtung: $m_x(s) = \sum_{(i,j) \in s} (j - j_\mu)^2$

Y-Richtung: $m_y(s) = \sum_{(i,j) \in s} (i - i_\mu)^2$

gemischt: $m_{xy}(s) = \sum_{(i,j) \in s} (i - i_\mu)(j - j_\mu)$

$m_{xy}(s) = 0$, wenn eine Bezugssachsenrichtung
 eine Symmetrieachse darstellt

Andernfalls ist Drehwinkel α berechenbar nach $\tan(2\alpha) = 2m_{xy}/(m_y - m_x)$.



MAP-klassifikation

Maximum-a-posteriori-Methode

A-Posteriori-Wahrscheinlichkeit: $P(s = c_k | m(s))$

Zuordnung zur Klasse c_k , die die A-Posteriori-Wahrscheinlichkeit maximiert:

$$P(s = c_k | m(s)) \geq P(s = c_j | m(s)) \text{ für alle } j \neq k.$$

Bestimmung der A-Posteriori-Wahrscheinlichkeit

- Merkmalsverteilung $P(m(s) | s = c_i)$ innerhalb einer Klasse ist leichter zu bestimmen
- Satz von Bayes

$$P(S = C_i | m(S)) = \frac{P(m(S) | S = C_i)P(S = C_i)}{\sum_{k=0}^{K-1} P(m(S) | S = C_k)P(S = C_k)}.$$

Normierung kann weggelassen werden, wenn nur das Maximum und nicht die W'heit gesucht wird

A-Priori-Wahrscheinlichkeit: $P(s = c_i)$

Merkmalsverteilung: $P(m(s) | s = c_i)$

Schätzung der A-Priori-Wahrscheinlichkeit

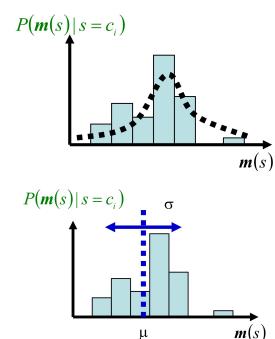
Es gilt: $\sum_{k=0, \dots, K-1} P(s = c_k) = 1$

1. $P(s = c_i)$ sind aus der Anwendungsdomäne bekannt
2. $P(s = c_i)$ wird aus der Verteilungsstatistik der Trainingsmenge $T = T_1 \cup T_2 \cup \dots \cup T_K$ geschätzt, die zu K -klassen zugeordnete Segmente enthält

Schätzung der klassenspezifischen Merkmalsverteilung

Schätzung einer M -dimensionalen Funktion abgeleitet aus den Histogrammen der Merkmalsvektoren aus der Trainingsmenge

1. Interpolation der Funktion durch die Trainingsmenge
2. Schätzung der Parameter einer vorgegebenen Verteilungscharakteristik



Bei unkorrelierten Merkmalen:

1. Erstelle für jedes Merkmal m_j ein normiertes Histogramm

Ggf.: Binning in B Bins $[m_{j,min}, m_{j,max}] \rightarrow [m_{j,min}, m_{j,1}], [m_{j,1}, m_{j,2}], \dots, [m_{j,B-1}, m_{j,max}]$
mit: $N = |T|$ Stichproben, $B = N^{\frac{1}{2}}$

2. $P(m_j(s) | s = c_k)$ bestimmen aus den normierten Histogrammen

$$3. P(m(s) | s = c_k) = P(m_1(s) | s = c_k) \cdot P(m_2(s) | s = c_k) \cdot \dots \cdot P(m_M(s) | s = c_k)$$

Bei korrelierten Merkmalen:

Kann nicht aus dem Produkt geschätzt werden

Nach der verallgemeinerten Daumenregel sind bei $N = |\mathcal{T}|$ Stichproben

$B = N^{1/(M+1)}$ Intervalle jeder der M Achsen des Merkmalsraumes zu wählen.

→ hohe Stichprobenzahl erforderlich

→ Charakteristik der Verteilung ist nötig

Beispiel:

Beispiel für die Schätzung von unkorrelierten und normalverteilten klassen-abhängigen Merkmalsverteilungen (1):

- Ausgangspunkt seien segmentierte Grauwertbilder von Äpfeln und Birnen.

- Ausgewählte Merkmale seien:

- 1) Die Kreisähnlichkeit $Kr(s)$ des Konturrandes: $Kr(s) = (F(s) \cdot 4\pi) / R(s)$ mit Fläche $F(s)$ und Randlänge $R(s)$ – vgl. Folien 16, 17 und 19.

Annahme: die Kreisähnlichkeit ist für Äpfel größer als für Birnen.

- 2) Der mittlere Intensitätswert des Segments:

Annahme: Birnen eher gelb bis grün, Äpfel eher rot bis grün, daher hellere Grauwerte bei Birnen.



Bildquelle: Colourbox (15.01.21)

die Merkmalsverteilungen für die Klassen c_1 und c_2 ergeben sich aufgrund der unkorrelierten Merkmale als Produkte

$$P((m_1(s), m_2(s)) | s = c_k) = P(m_1(s) | s = c_k) \cdot P((m_2(s) | s = c_k), k = 1, 2.$$

Aus je 10 Stichproben werden Erwartungswert E und Varianz σ^2 geschätzt:

$$E_k(m_i) \approx \frac{1}{10} \sum_{j=0}^9 m_i(s_j), \sigma_k^2(m_i) \approx \frac{1}{9} \sum_{j=0}^9 (m_i(s_j) - E(m_i))^2, i = 1, 2, k = 1, 2.$$

Daraus folgt (für $k = 1, 2$):

Zur Erinnerung:

$$P(S = C_i | \mathbf{m}(S)) = \frac{P(\mathbf{m}(S) | S = C_i) P(S = C_i)}{\sum_{k=0}^K P(\mathbf{m}(S) | S = C_k) P(S = C_k)}$$

$$P(c_k | (m_1(s), m_2(s))) \propto \frac{1}{\sigma_k(m_1)\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{m_1(s)-E_k(m_1)}{\sigma_k(m_1)}\right)^2} \cdot \frac{1}{\sigma_k(m_2)\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{m_2(s)-E_k(m_2)}{\sigma_k(m_2)}\right)^2}.$$

Die 10 Stichproben m_j ($0 \leq j \leq 9$) für Klasse c_1 (Äpfel) mit Merkmalen $m_{j1} = Kr(s)$ und $m_{j2} = i_{average}(s)$:

$$\vec{m}_j = \begin{pmatrix} m_{j1} \\ m_{j2} \end{pmatrix} = \begin{pmatrix} 0,95 \\ 0,62 \end{pmatrix}, \begin{pmatrix} 0,92 \\ 0,64 \end{pmatrix}, \begin{pmatrix} 0,84 \\ 0,64 \end{pmatrix}, \begin{pmatrix} 0,99 \\ 0,72 \end{pmatrix}, \begin{pmatrix} 0,91 \\ 0,70 \end{pmatrix}, \begin{pmatrix} 0,92 \\ 0,65 \end{pmatrix}, \begin{pmatrix} 0,94 \\ 0,59 \end{pmatrix}, \begin{pmatrix} 0,98 \\ 0,67 \end{pmatrix}, \begin{pmatrix} 0,99 \\ 0,68 \end{pmatrix}, \begin{pmatrix} 0,83 \\ 0,71 \end{pmatrix}$$

ergeben für c_1 : $E_1(m_1) = 0,93$, $\sigma_1^2(m_1) = 0,0032$, $E_1(m_2) = 0,66$, $\sigma_1^2(m_2) = 0,0017$.

Die 10 Stichproben für c_2 (Birnen):

$$\vec{m}_j = \begin{pmatrix} m_{j1} \\ m_{j2} \end{pmatrix} = \begin{pmatrix} 0,72 \\ 0,80 \end{pmatrix}, \begin{pmatrix} 0,80 \\ 0,83 \end{pmatrix}, \begin{pmatrix} 0,77 \\ 0,72 \end{pmatrix}, \begin{pmatrix} 0,60 \\ 0,91 \end{pmatrix}, \begin{pmatrix} 0,67 \\ 0,83 \end{pmatrix}, \begin{pmatrix} 0,72 \\ 0,85 \end{pmatrix}, \begin{pmatrix} 0,84 \\ 0,74 \end{pmatrix}, \begin{pmatrix} 0,85 \\ 0,77 \end{pmatrix}, \begin{pmatrix} 0,78 \\ 0,79 \end{pmatrix}, \begin{pmatrix} 0,77 \\ 0,88 \end{pmatrix}$$

ergeben für c_2 : $E_2(m_1) = 0,75$, $\sigma_2^2(m_1) = 0,0058$, $E_2(m_2) = 0,81$, $\sigma_2^2(m_2) = 0,0036$.

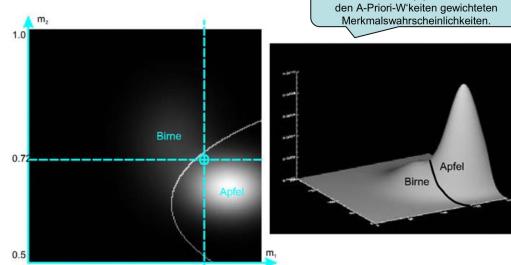
Die Berechnungen der A-Posteriori-W'keiten liefern:

$$P(c_1 | (0,83, 0,72)) = 0,65,$$

$$P(c_2 | (0,83, 0,72)) = 0,35.$$

Also wird Segment s_{test} der Klasse c_1 der Äpfel zugeordnet.

Die Abbildungen zeigen die Position des Merkmalsvektors $\vec{f}(S)$ im Raum der mit den A-Priori-W'keiten gewichteten Merkmalswahrscheinlichkeiten.



II. Merkmale (Features)

Ansätze des Feature Engineering

- Traditional Pattern Recognition: Fixed/Handcrafted Feature Extractor



- Mainstream Modern Pattern Recognition: Unsupervised mid-level features



- Deep Learning: Representations are hierarchical and trained



Local Binary Patterns

Der LBP-Dekcriptor ist ein histogrammbasierter Dekcriptor, der das zentrale Pixel mit den Nachbarpixeln vergleicht, startend mit dem "Ost"-Pixel, und daraus eine Binärzahl produziert

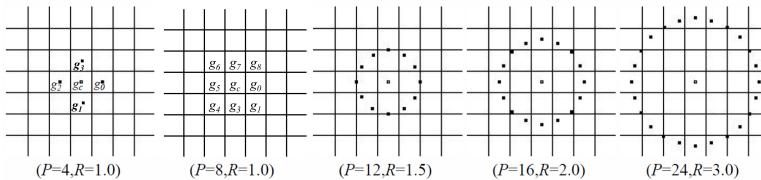
$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c) 2^p$$

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

P: Anzahl Pixel
R: Radius

60	70	80
50	50	10
40	30	20

$$\Rightarrow 11110000 \Rightarrow 2^7 + 2^6 + 2^5 + 2^4 = 240$$



Rotation Invariant Local Binary Patterns

Durch bitweise Rechtsshift werden alle Rotationsvarianten auf die Variante mit der geringsten Binärzahl projiziert

$$LBP_{P,R}^{ri} = \min\{ROR(LBP_{P,R}, i) \mid i = 0, 1, \dots, P-1\}$$

where $ROR(x, i)$ performs a circular bit-wise right shift on the P -bit number x i times

$$\begin{array}{c} \bullet \circ \circ \\ \vdots \quad \cdot \quad \cdot \\ \cdot \cdot \cdot \end{array} \Rightarrow [11000000] \Rightarrow 2^6 + 2^7 = 192 \quad \begin{array}{c} \bullet \circ \circ \\ \vdots \quad \cdot \quad \cdot \\ \cdot \cdot \cdot \end{array} \Rightarrow [10000001] \Rightarrow 2^0 + 2^7 = 129$$

Uniform Rotational Invariant Local Binary Patterns

Nur Muster mit maximal zwei Wechsel zwischen 0 und 1 leisten einen Beitrag zur Erkennung. Muster werden auf ihre rotationsinvariante Darstellung gebracht.

$$LBP_{8,R}^{riu2} \text{ codes: } \begin{array}{ccccccccccccccccccccccccc} \bullet & \bullet \\ \cdot & 0 & \cdot & 1 & \circ & 2 & \circ & 3 & \circ & 4 & \circ & 5 & \circ & 6 & \circ & 7 & \circ & 8 & \circ \\ \bullet & \bullet \end{array}$$

Momente

$$m_{pq}(s) = \sum_{(i,j) \in s} i^p j^q$$

→ Moment der Ordnung $p+q$
 → nicht translationsinvariant & skalierungsinvariant

Für Grauwert-Bilder: $m_{pq}(s) = \sum_{(i,j) \in s} i^p j^q g(i,j)$

Fläche: m_{00}

Schwerpunkt: $(i_\mu, j_\mu) = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right)$

Zentrale Momente

$$\mu_{pq}(s) = \sum_{(i,j) \in s} (i - i_\mu)^p (j - j_\mu)^q \quad \text{mit Schwerpunkt } (i_\mu, j_\mu)$$

→ translationsinvariant, aber nicht skalierungsinvariant

Fläche: zentr. Moment 0-ter Ordnung $\mu_{00}(s)$

Trägheitsmomente: zentr. Momenten 2-ter Ordnung $\mu_{ij}(s)$

Normalisierte zentrale Momente

$$\eta_{pq}(s) = \frac{\mu_{pq}(s)}{[\mu_{00}(s)]^\gamma} \quad \text{mit } \gamma = \frac{p+q}{2} + 1 \text{ für } p+q \geq 2$$

→ translations- und skalierungsinvariant

Hu-Moment

→ invariant bzgl. Translation, Skalierung, Spiegelung, Rotation

$$\begin{aligned} \phi_1 &= \eta_{2,0} + \eta_{0,2} \\ \phi_2 &= (\eta_{2,0} - \eta_{0,2})^2 + 4 \cdot \eta_{1,1}^2 \\ \phi_3 &= (\eta_{3,0} - 3 \cdot \eta_{1,2})^2 + (3 \cdot \eta_{2,1} - \eta_{0,3})^2 \\ \phi_4 &= (\eta_{3,0} + \eta_{1,2})^2 + (\eta_{2,1} + \eta_{0,3})^2 \\ \phi_5 &= (\eta_{3,0} - 3 \cdot \eta_{1,2})(\eta_{3,0} + \eta_{1,2})[(\eta_{3,0} + \eta_{1,2})^2 - 3 \cdot (\eta_{2,1} - 3 \cdot \eta_{0,3})^2] \\ &\quad + (3 \cdot \eta_{3,0} + \eta_{1,2})(\eta_{2,1} + \eta_{0,3})[3 \cdot (\eta_{3,0} + \eta_{1,2})^2 - (\eta_{2,1} + \eta_{0,3})^2] \\ \phi_6 &= (\eta_{2,0} - \eta_{0,2})[(\eta_{3,0} + \eta_{1,2})^2 - (\eta_{2,1} + \eta_{0,3})^2] \\ &\quad + 4 \cdot \eta_{1,1}(\eta_{3,0} + \eta_{1,2})(\eta_{2,1} + \eta_{0,3}) \\ \phi_7 &= (3 \cdot \eta_{2,1} - \eta_{0,3})(\eta_{3,0} + \eta_{1,2})[(\eta_{3,0} + \eta_{1,2})^2 - 3 \cdot (\eta_{2,1} + \eta_{0,3})^2] \\ &\quad + (3 \cdot \eta_{1,2} - \eta_{3,0})(\eta_{0,3} + \eta_{2,1})[3 \cdot (\eta_{3,0} + \eta_{1,2})^2 - (\eta_{2,1} + \eta_{0,3})^2] \end{aligned}$$

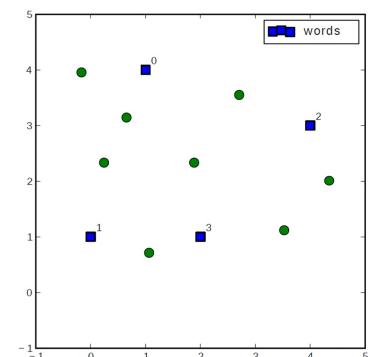
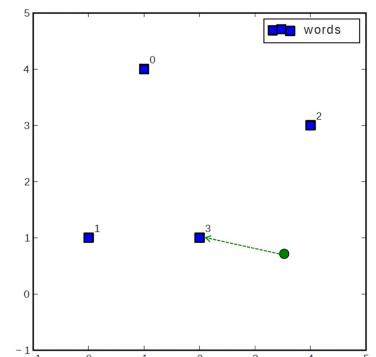
Anschließende Skalierung: $\phi_i' = \text{sign}(\phi_i) \log_{10}(|\phi_i|)$

Bag of Visual Words

Erzeugt Mid-Level-Features auf Basis von handcrafted Features
→ Ansatz aus der Dokumentenverarbeitung

1. Erzeugen eines Dictionary D bestehend aus k Visual Words
→ meist Vorgegeben
→ sonst Clustering mit k Clustern anhand von Merkmälern
2. Ordne die Descriptoren der Segmente den Visual Words zu durch Nächste-Nachbar-Zuordnung
3. Histogramm des Auftretens der Visual Words erstellen
4. BoW-Repräsentation $\text{BoW}(s)$ eines Segments erstellen

Normalisiertes Histogramm über die Zuordnung der Descriptoren zu den Visual Words



Principal Component Analysis

PCA ist eine orthogonale Transformation die k linear korrelierte Merkmale (z_1, \dots, z_n) in m linear **dekorrelierte** Merkmale (w_1, \dots, w_m)

w_j : Hauptkomponente

Erste Hauptkomponente hat die größte Varianz, danach weiter absteigend. Varianzen der letzten Hauptkomponenten sind so gering, dass sie nicht zur Charakterisierung der Segmente beitragen.

Variationsmodi: m Komponenten mit hinreichend großer Varianz

$$\text{Wahl von } m: \sum_{j=0}^m \lambda_j \geq 0.95 \cdot \sum_{j=0}^k \lambda_j$$

Transformation:

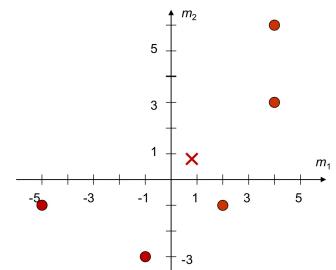
1. Daten in ihrem Schwerpunkt zentrieren

Für alle n Segmente s_i ($i = 1, \dots, n$) mit k Merkmalen

Für jedes Merkmal z_j ($j = 1, \dots, k$)

$$1) \text{ Berechnung von Mittelwert } \mu_j = \frac{1}{n} \sum_{i=1}^n z_j(s_i)$$

$$2) \text{ Ersetzung von } z_j(s_i) \text{ durch } z_j(s_i) - \mu_j \rightarrow E(\mathbf{z}) = 0$$

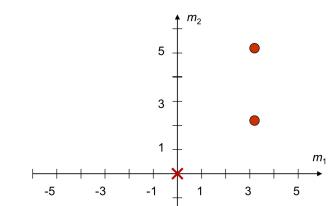


2. Kovarianzmatrix aufstellen

Für $\mathbf{z}_i = (z_{i1}, \dots, z_{ik})$:

$$\mathbf{C}(\mathbf{z}) = \begin{pmatrix} \sigma_1^2 & \cdots & \sigma_1 \sigma_k \\ \vdots & \ddots & \vdots \\ \sigma_k \sigma_1 & \cdots & \sigma_k^2 \end{pmatrix}$$

$$\sigma_j \sigma_{j'} = \frac{1}{n} \sum_{i=1}^n (z_{ij} - \mu_j)(z_{ij'} - \mu_{j'}) \text{ mit } \mu_l = \frac{1}{n} \sum_{i=1}^n (z_{il})$$



3. Transformation zu dekorrelierten Merkmalen

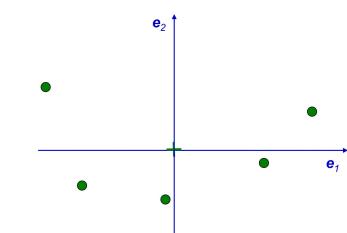
→ Richtung der größten Varianz

1. Eigenwerte durch charakteristisches Polynom bestimmen

$$\det(\mathbf{M} - \lambda_k \mathbf{Id}) = 0$$

2. Eigenvektoren bestimmen

$$\mathbf{M} \times \mathbf{e}_k = \lambda_k \mathbf{e}_k$$



3. Punkte mit Transformationsmatrix transformieren

$$(z_{i1}, \dots, z_{ik}) \begin{pmatrix} e_{11} & \cdots & e_{k1} \\ \vdots & \ddots & \vdots \\ e_{1k} & \cdots & e_{kk} \end{pmatrix} = (w_{i1}, \dots, w_{ik}) \rightarrow \mathbf{C}(\mathbf{w}) = \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_k \end{pmatrix}$$

Rücktransformation

1. Inverse Transformationsmatrix aufstellen

$$\mathbf{E}^{-1} = \mathbf{E}^T$$

2. Rücktransformation berechnen

$$\mathbf{z} = \mathbf{E}(\mathbf{z}) + \mathbf{w} \times \mathbf{E}^T$$

$$\mathbf{z} = (z_1, \dots, z_k) = \mathbf{E}(\mathbf{z}) + (w_1, \dots, w_m) \begin{pmatrix} e_{11} & \cdots & e_{1k} \\ \vdots & \ddots & \vdots \\ e_{m1} & \cdots & e_{mk} \end{pmatrix}$$