Computergrafik 2: Kanten, Linien, Ecken

Prof. Dr. Michael Rohs, Dipl.-Inform. Sven Kratz

michael.rohs@ifi.lmu.de

MHCI Lab, LMU München

Folien teilweise von Andreas Butz, sowie von Klaus D. Tönnies
(Grundlagen der Bildverarbeitung. Pearson Studium, 2005.)

Vorlesungen

Datum	Thema
24.4.	Einführung, Organisatorisches (Übungen, Klausur)
1.5./8.5.	keine Vorlesungen (wegen 1. Mai und CHI-Konferenz)
15.5.	Abtastung von Bildern, Punktbasierte Verfahren der Bildverbesserung
22.5.	Licht, Farbe, Farbmanagement
30.5.	Konvolution, Filterung im Ortsraum (Verschiebung wegen Pfingstdienstag)
5.6.	Fouriertransformation: Grundlagen
12.6.	Filterung im Frequenzraum
19.6.	Kanten, Linien, Ecken
26.6.	Segmentierung
3.7.	Segmentierung, Morphologische Operationen
10.7.	Klassifikation
17.7.	Image Matching
24.7.	Klausur (Hörsaal M 018 im Hauptgebäude, 14-16 Uhr)

WARUM ERZEUGT DER GAUSSFILTER KEINE RINGING-ARTEFAKTE?

Fourier-Transformation einer Gauß-Funktion

$$f(t) = e^{-at^2}$$

$$F(u) = \int_{-\infty}^{\infty} f(t) \cdot e^{-i2\pi ut}\, dt = \int_{-\infty}^{\infty} e^{-at^2} \cdot e^{-i2\pi ut}\, dt$$

$$= \int_{-\infty}^{\infty} e^{-at^2} \cdot \left[\cos(2\pi ut) - i\sin(2\pi ut)\right] dt$$

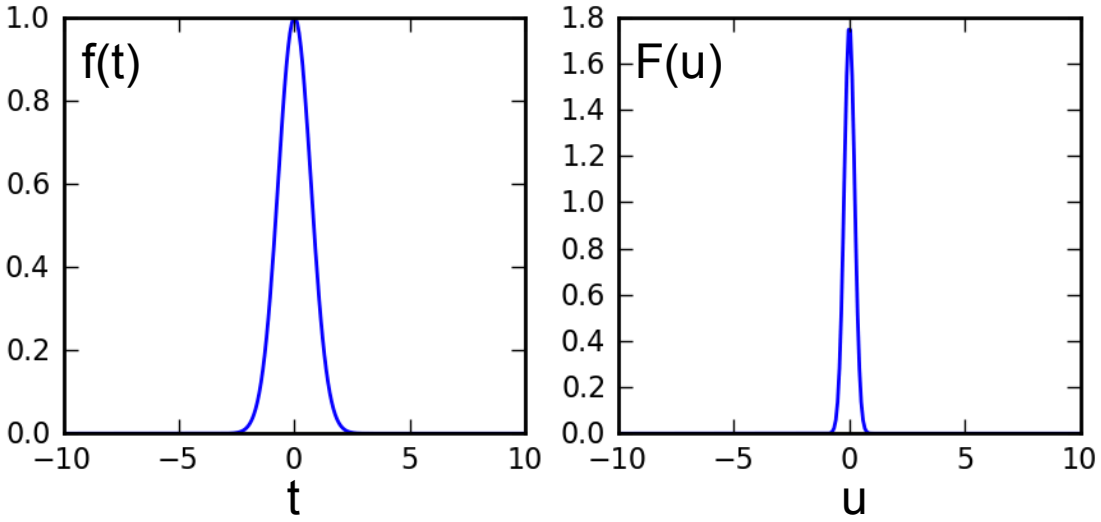
$$= \int_{-\infty}^{\infty} e^{-at^2} \cdot \cos(2\pi ut)\, dt - i\int_{-\infty}^{\infty} e^{-at^2} \cdot \sin(2\pi ut)\, dt$$

$$= \int_{-\infty}^{\infty} e^{-at^2} \cdot \cos(2\pi ut)\, dt - 0$$

$$= \sqrt{\frac{\pi}{a}} e^{-\pi^2 u^2 / a}$$

sin ist punktsymmetrisch zum Ursprung, insgesamt ist zweiter Integrand ungerade, symmetrische Integrationsgrenzen: 0

erstes Integral: Handbook of Mathematical Functions...



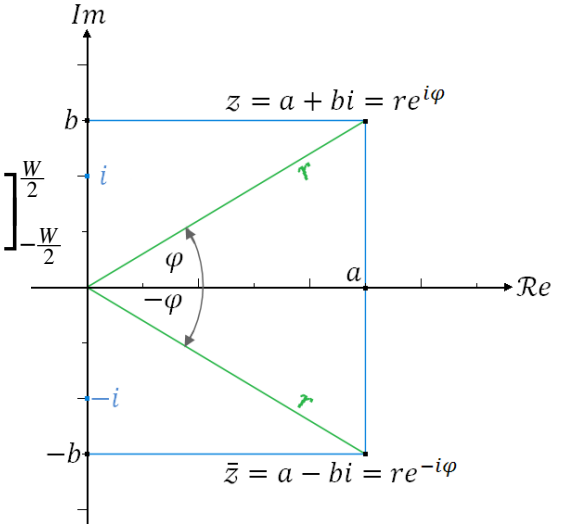
Fourier-Transformation einer Box-Funktion

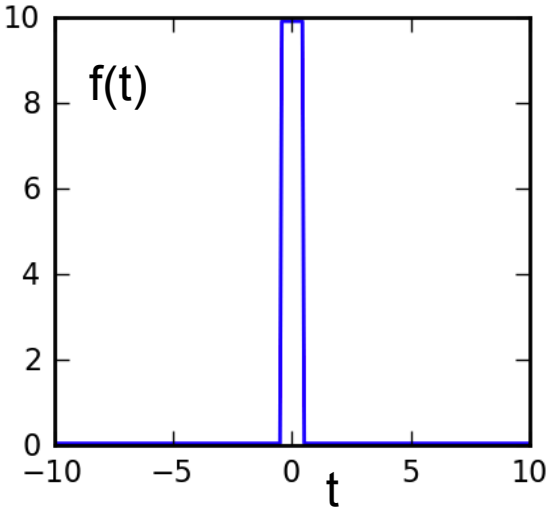
$$f(t) = \begin{cases} A & falls\ -\frac{W}{2} \le t \le \frac{W}{2} \\ 0 & sonst \end{cases}$$

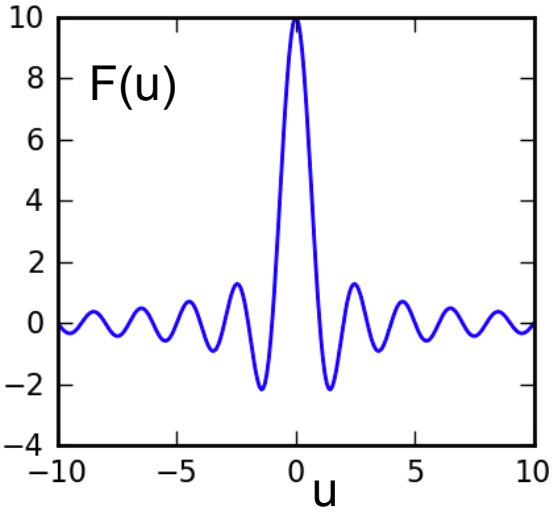
$$F(u) = \int_{-\infty}^{\infty} f(t) \cdot e^{-i2\pi ut}\, dt = \int_{-\frac{W}{2}}^{\frac{W}{2}} A \cdot e^{-i2\pi ut}\, dt = -\frac{A}{i2\pi u}\left[e^{-i2\pi ut}\right]_{-\frac{W}{2}}^{\frac{W}{2}}$$

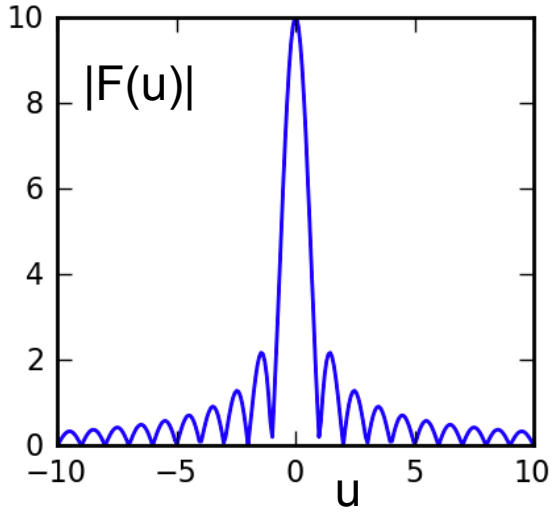
$$= -\frac{A}{i2\pi u}\left[e^{-i2\pi\frac{W}{2}} - e^{i2\pi u\frac{W}{2}}\right] = \frac{A}{i2\pi u}\left[e^{i\pi uW} - e^{-i\pi uW}\right]$$

$$= \frac{A}{i2\pi u} i2\sin(\pi uW) = AW\frac{\sin(\pi uW)}{\pi uW}$$ "Ringing-Artefakt"



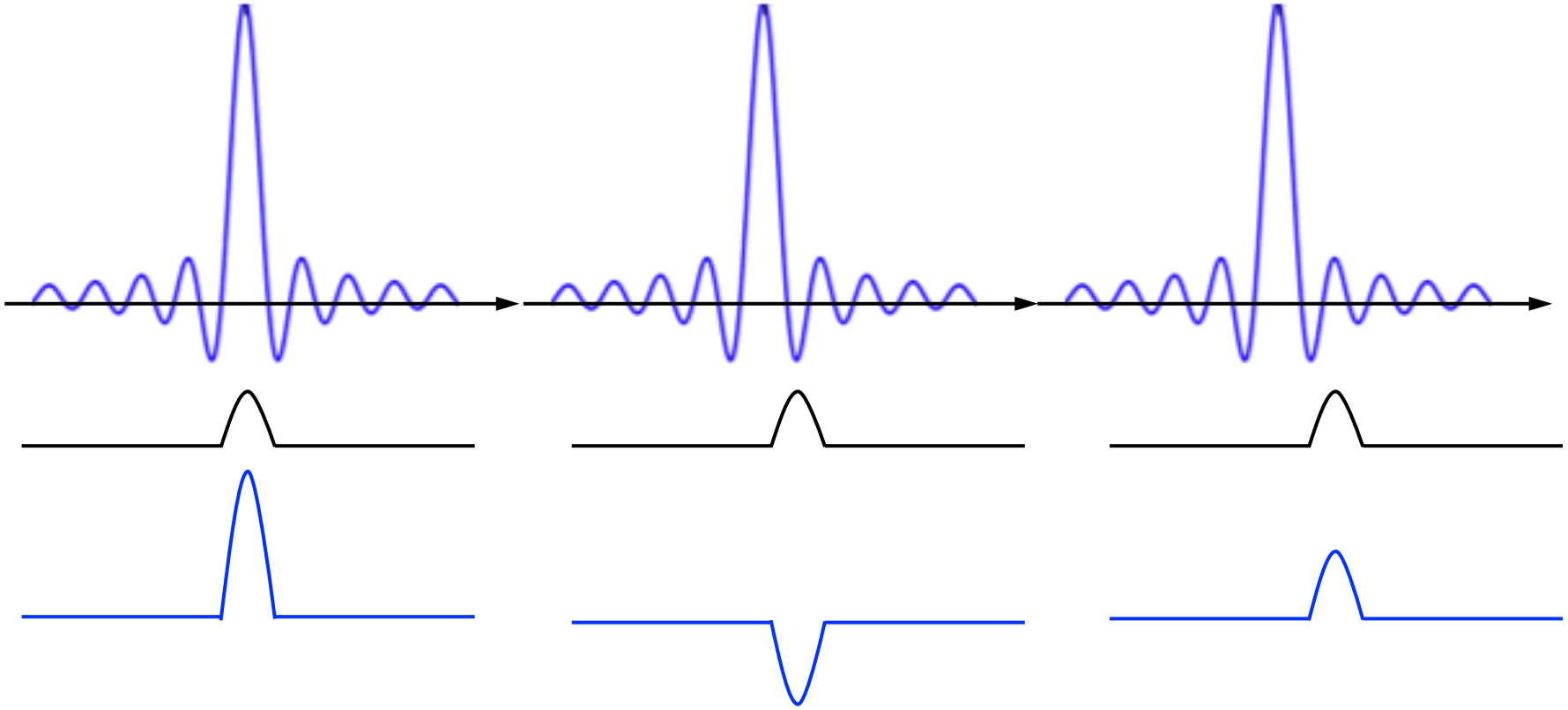






Konvolution im Ortsraum

- elementweise Multiplikation im Frequenzraum entspricht Konvolution im Ortsraum
- Konvolution ist gewichtete Mittelwertbildung



Themen heute

- Kanten
  - Kantenoperatoren
  - Kantenattribute
  - Kanten und Rauschen
- Canny-Kantenerkennung
- Hough-Transformation
- Harris-Eckenerkennung

Erkennung von Merkmalen

- Merkmal (engl. „feature“): wichtiges/interessantes lokales Muster
- Erkennung von Merkmalen wichtiger Schritt beim Lokalisieren oder Erkennen von Objekten in Bildern
- Beispiele für Merkmale
  - Kanten
  - Linien
  - Ecken
  - spezifische lokale Muster
  - Texturen

KANTEN

Kanten

- Kanten grenzen homogene Regionen ein → Segmentierung
- Kantenverstärkung kann Bildwahrnehmung verbessern
- Detektion von Kantenpunkten
- Finden von Kantenzügen
- Starke lokale Intensitätsänderungen
  - „Ableitung“
- Stärke der Intensitätsänderung
  - Betrag / Gradient der Ableitungsfunktion
- Diskrete Bilder
  - Ableitung angenähert durch Differenzenquotient

Erste und zweite Ableitung von Bildern

- Erste Ableitung:

$$\frac{\partial f}{\partial x}(x) = f(x+1) - f(x)$$

- Zweite Ableitung:

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

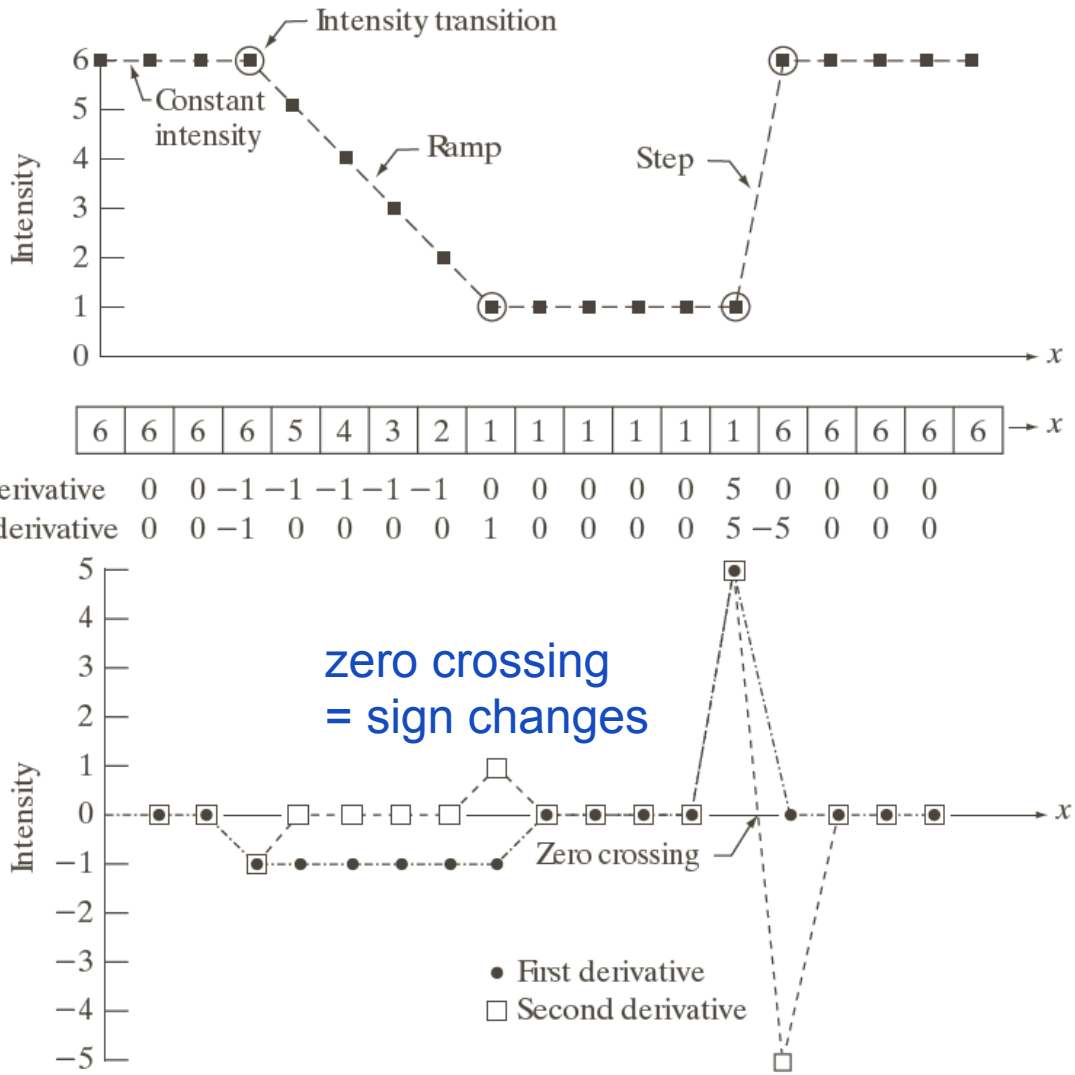


Abbildung: © R. C. Gonzalez & R. E. Woods, Digital Image Processing

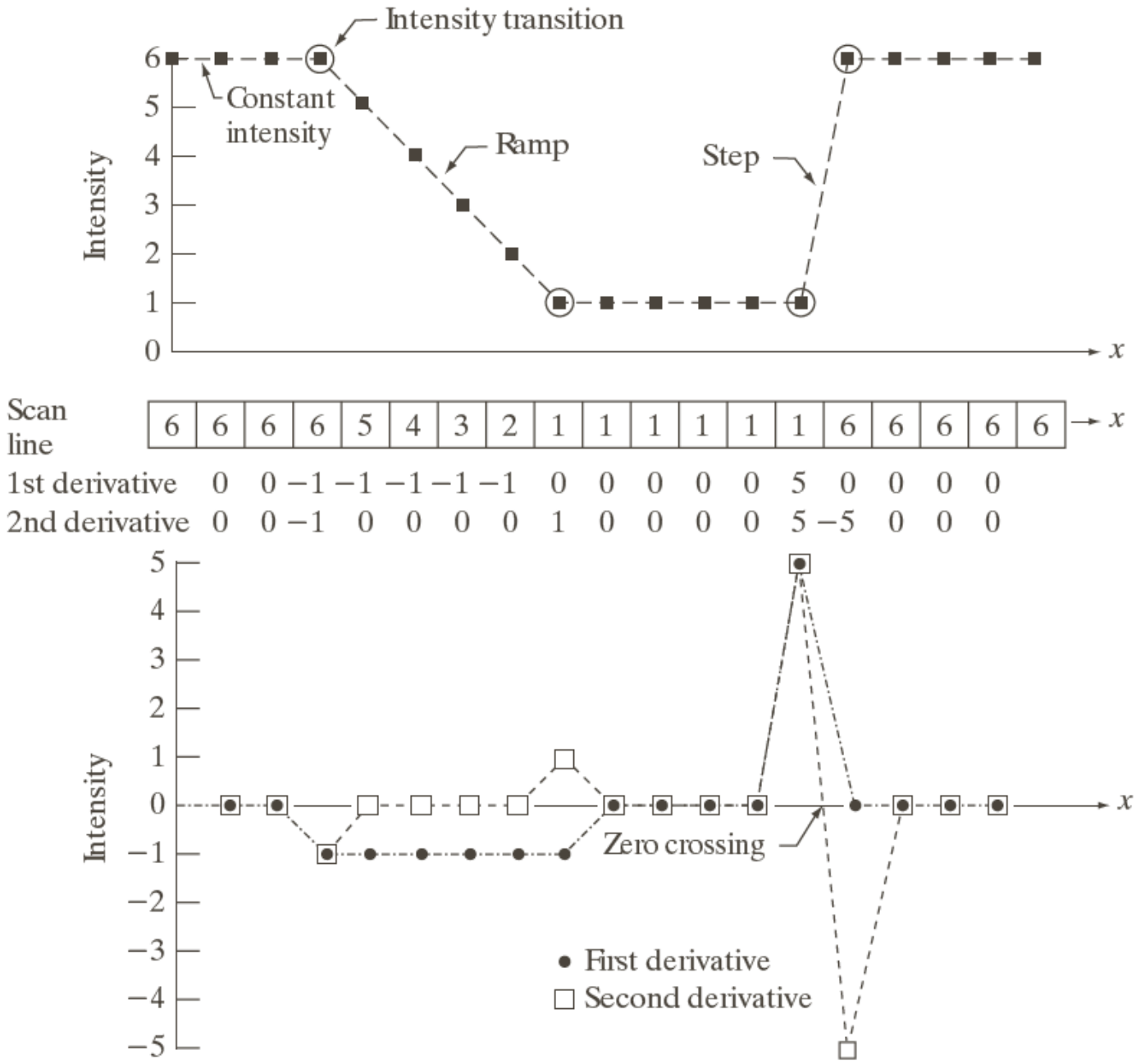
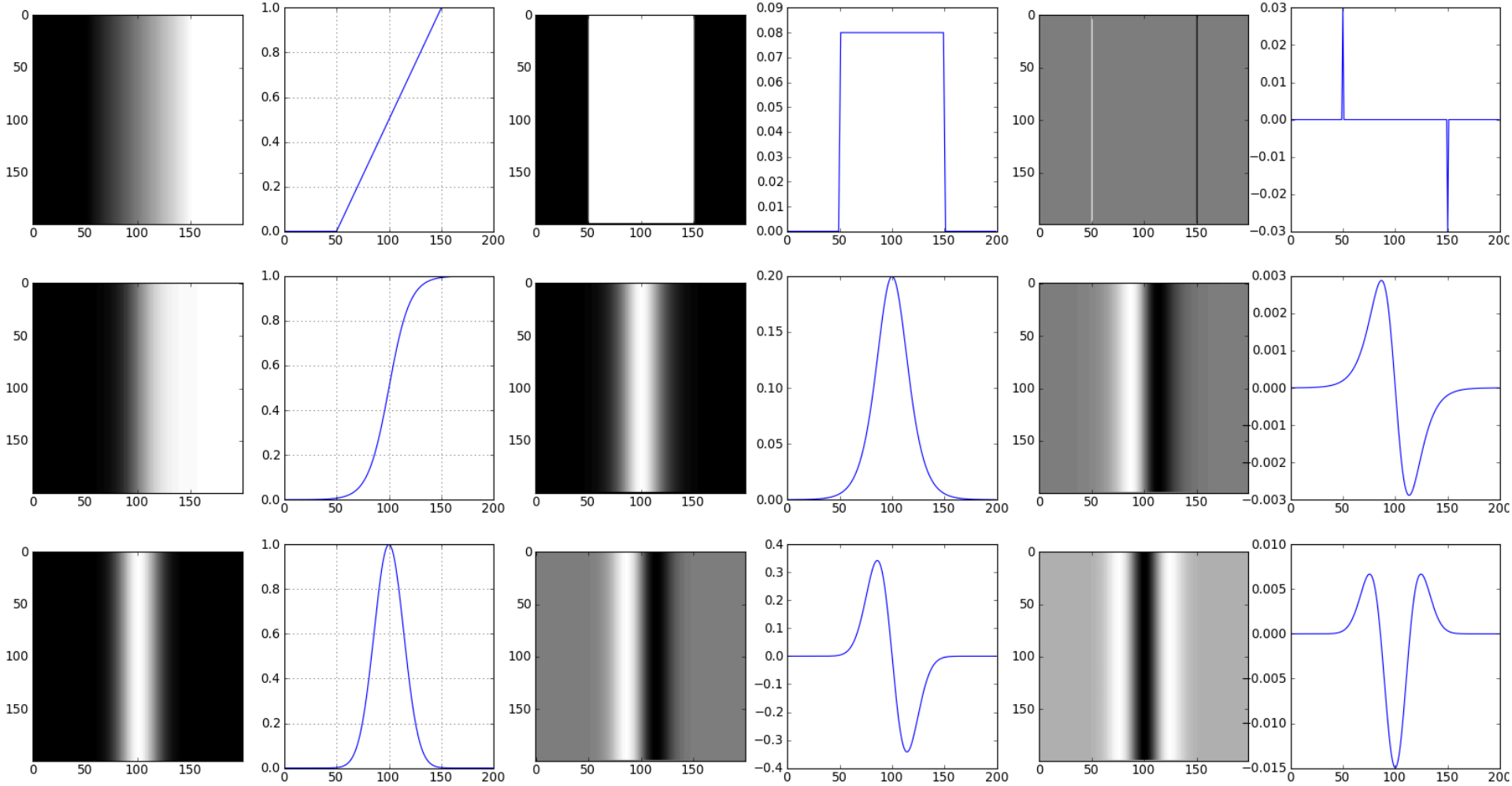


Abbildung: © R. C. Gonzalez & R. E. Woods, Digital Image Processing

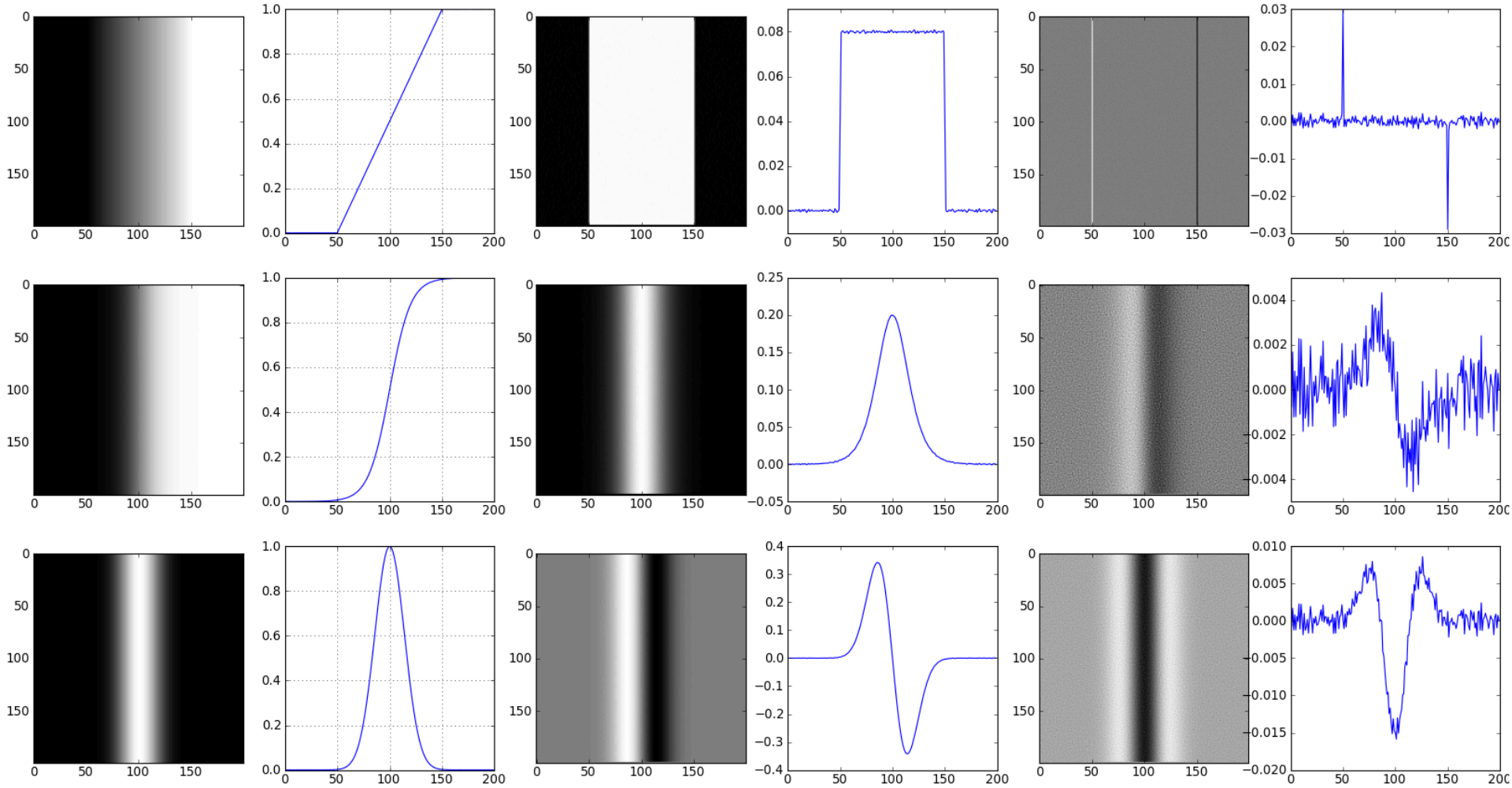
Intensitätsprofil, erste und zweite Ableitung

Intensitätsprofil | erste Ableitung | zweite Ableitung



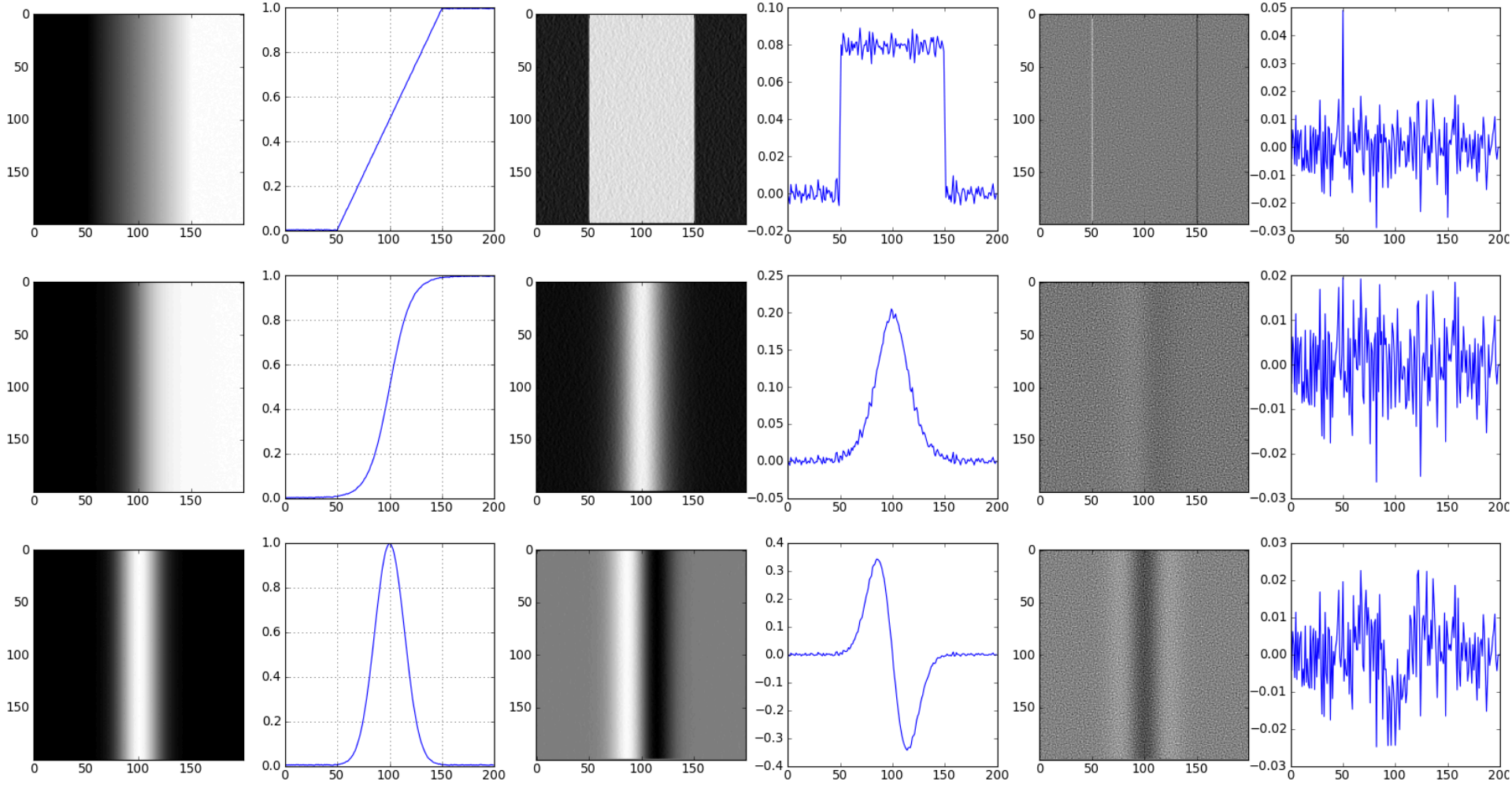
Normalverteiltes Rauschen, σ = 0.0001

Intensitätsprofil | erste Ableitung | zweite Ableitung



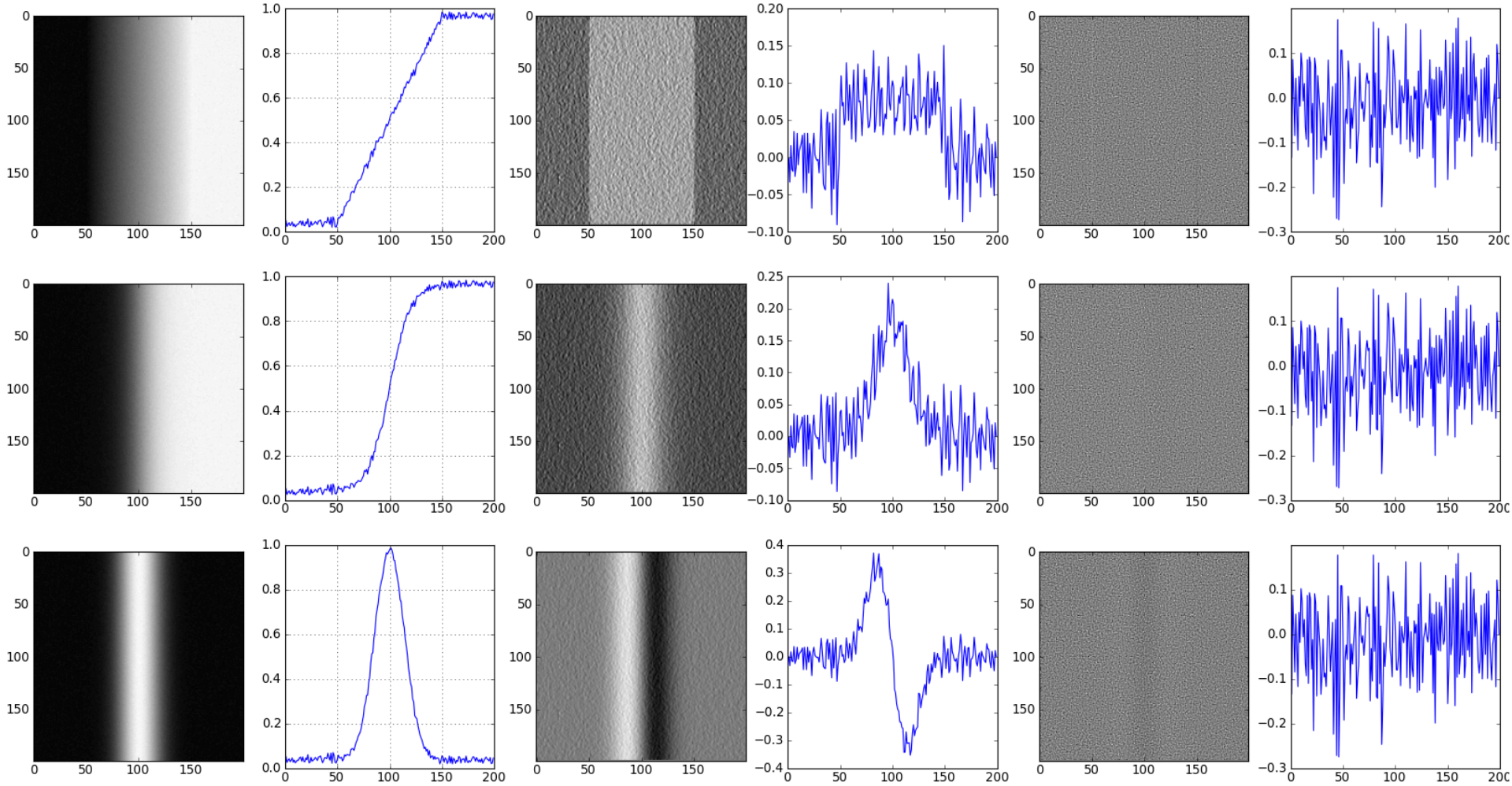
Normalverteiltes Rauschen, σ = 0.001

Intensitätsprofil | erste Ableitung | zweite Ableitung



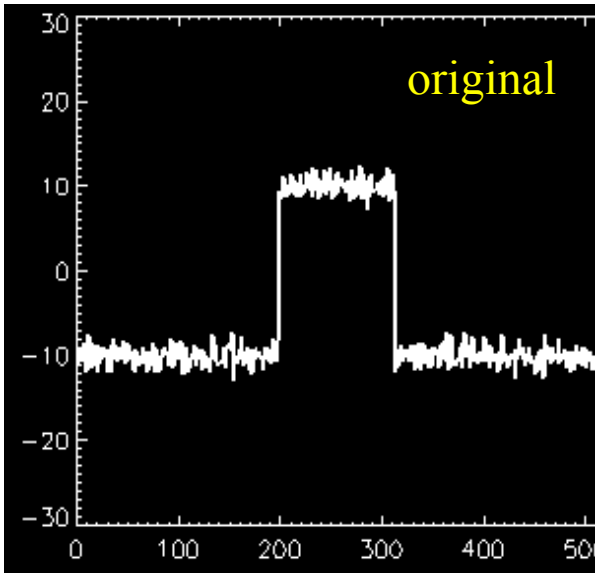
Normalverteiltes Rauschen, σ = 0.01

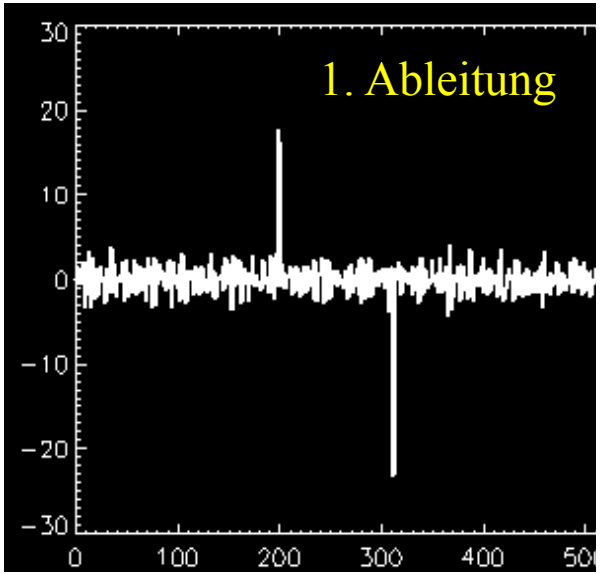
Intensitätsprofil | erste Ableitung | zweite Ableitung

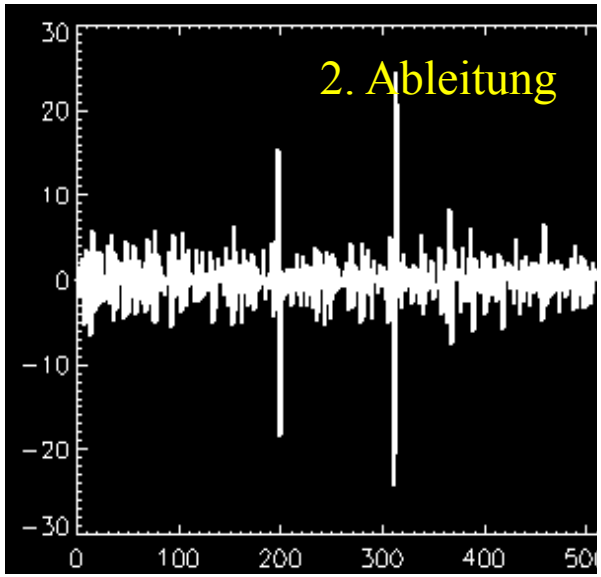


Kanten im Ortsraum

Kanten und Rauschen haben ähnliche Charakteristika im Frequenzraum ➔
Kantendetektor verstärkt Rauschen







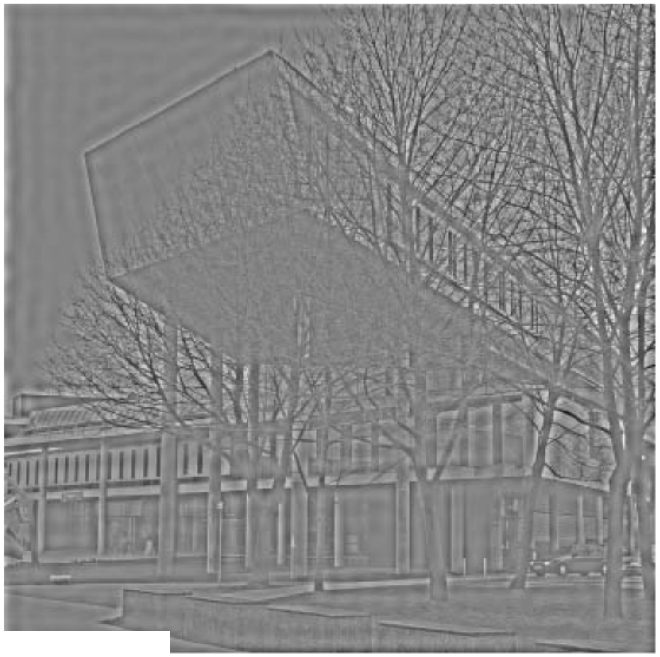
Kantenhervorhebung durch Frequenzraumfilterung

- Kanten weisen mehr hochfrequente Anteile auf als homogene Gebiete

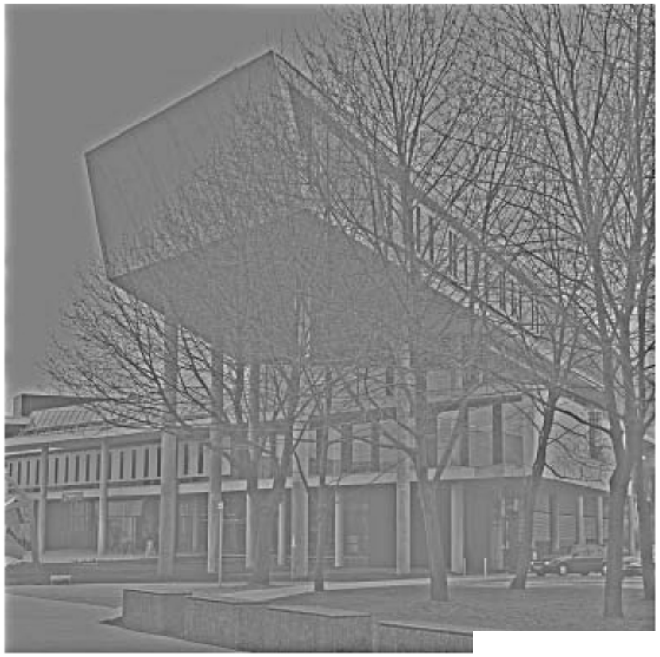
►Hochpassfilterung

$$H_{F_{\max}}(u,v) = \begin{cases} 1 & \text{, falls } u^2 + v^2 \geq F_{\max}^2 \\ 0 & \text{, sonst.} \end{cases}$$

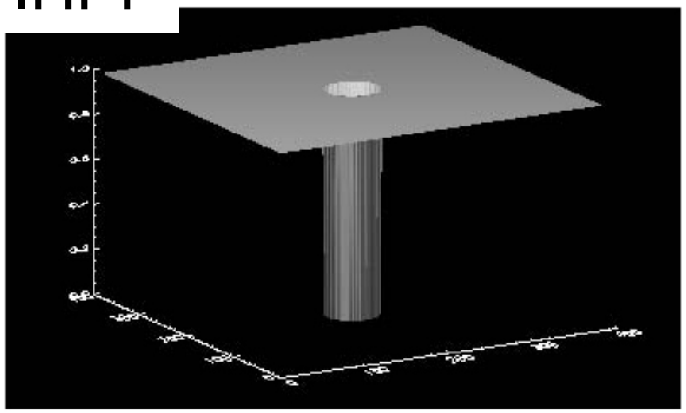
Filterung mit idealem Hochpass- und Butterworth-Hochpassfilter

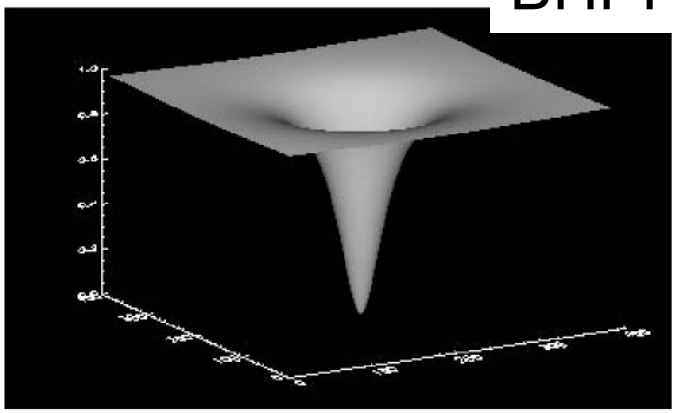


IHPF



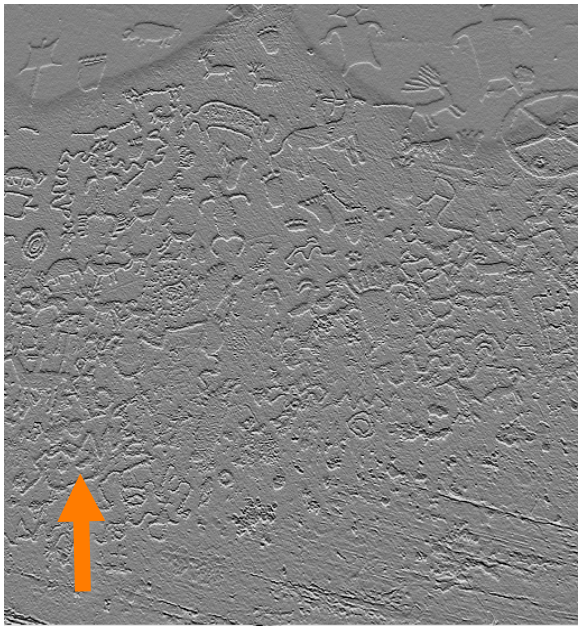
BHPF



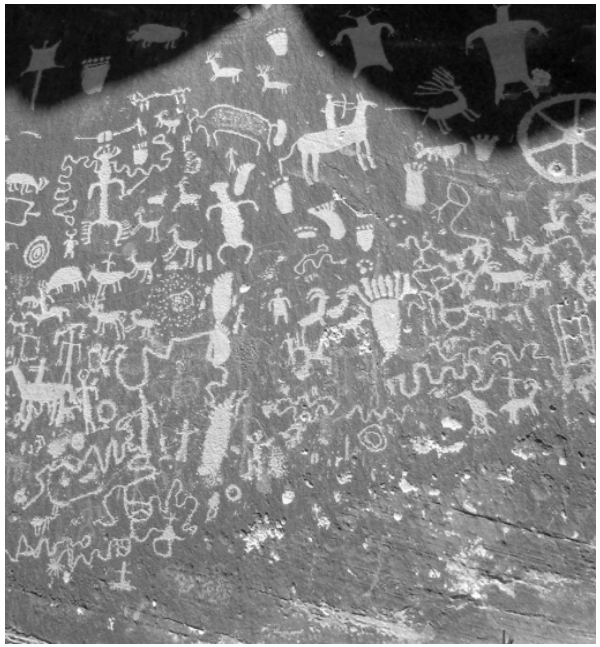


Kanten im 2-D Raum

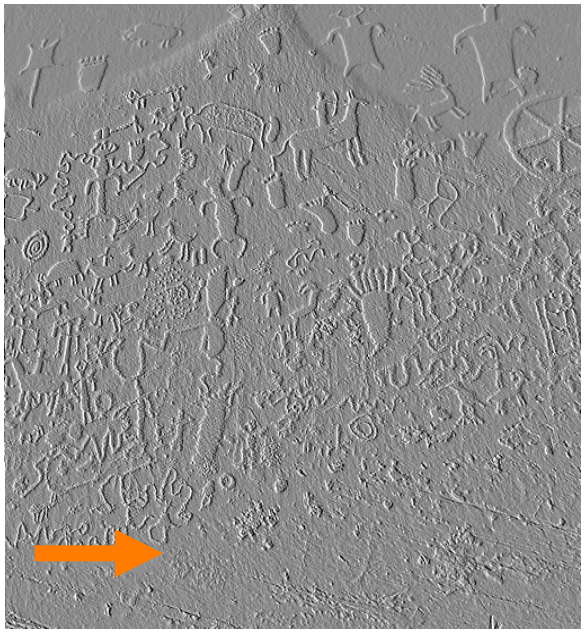
- Kanten können beliebig orientiert sein
- Differenzbildung in zwei orthogonalen Richtungen



Differenzbildung in *m*-Richtung



Pseudo-3D Eindruck



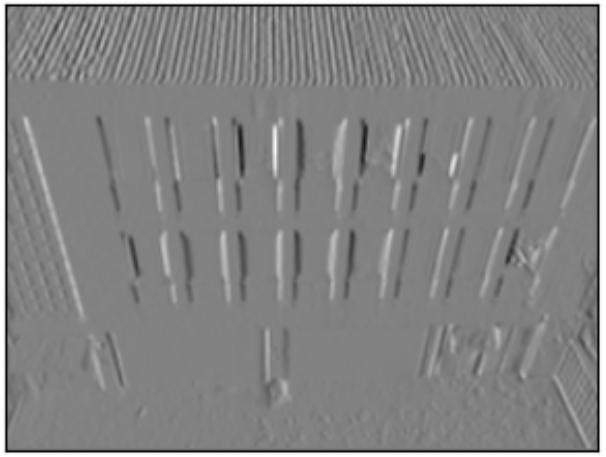
Differenzbildung in *n*-Richtung

Gradienten finden: Konvolution mit dem Sobel-Operator

$*$

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

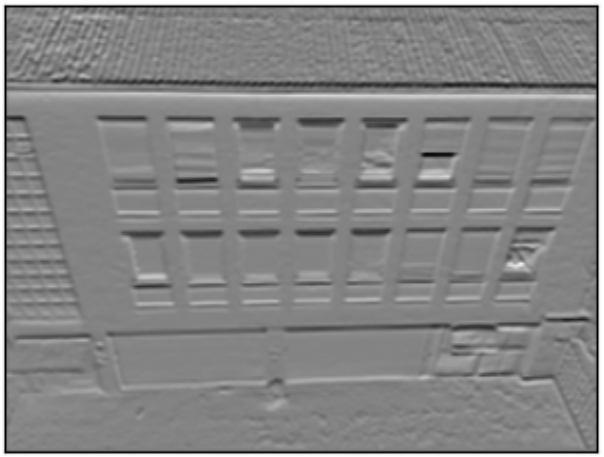
$=$



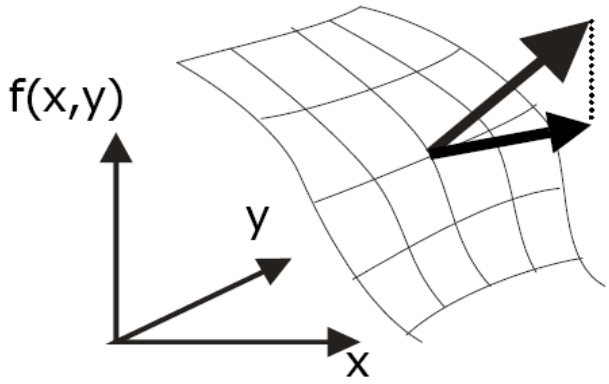
$*$

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

$=$



Kanten im 2D-Raum: Gradienten



Richtung: Richtung der größten Steigung.

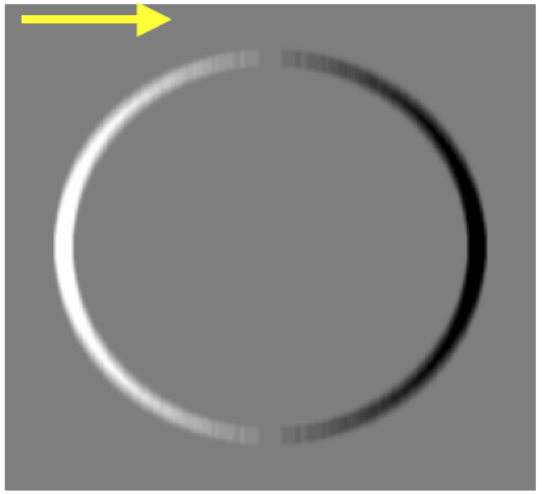
Länge: Stärke der stärksten Steigung.

- Gradient im kontinuierlichen Raum (x,y): Vektor der partiellen Ableitungen der Bildfunktion in x- und y-Richtung: $(f(x,y)) = (\partial f/\partial x \; \partial f/\partial y)$
- Approximation des Gradienten: Differential wird durch Differenz approximiert:

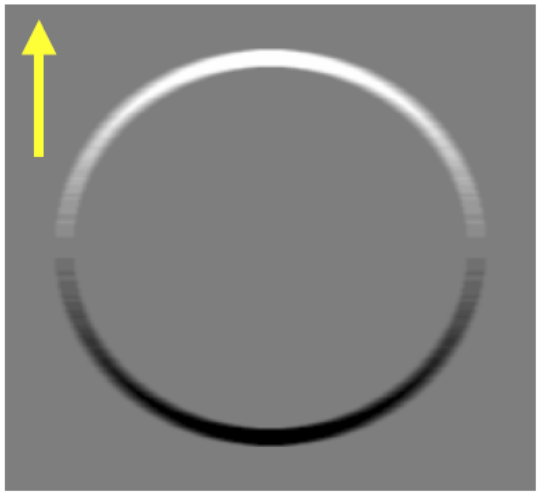
$$\vec{G}(f)(m,n) \approx [G_x(m,n) \quad G_y(m,n)] = [f(m,n)\text{-}f(m\text{-}1,n) \quad f(m,n)\text{-}f(m,n\text{-}1)]$$

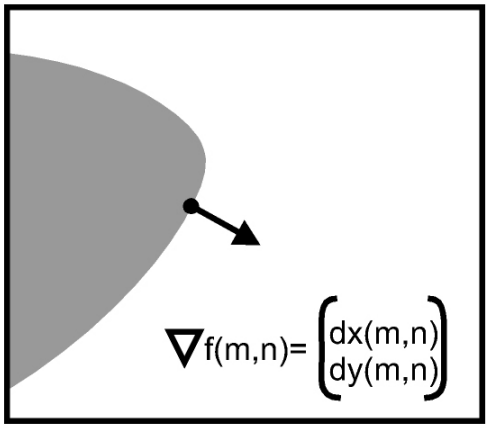
- Die Länge des Gradienten ist sein Betrag $|G(f)|$ oder näherungsweise $|Gx|+|Gy|$.

Elemente des Gradienten



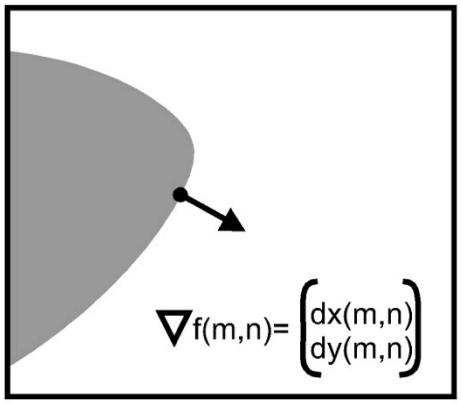
Betrag: $\text{sqrt}(G_x^2 + G_y^2)$
Richtung: $\tan^{-1}(G_y / G_x)$





Gradientenschätzung über Konvolution

- Gradient im *N*-dimensionalen Raum ist ein *N*-dimensionaler Vektor aus *N* partiellen Ableitungen
- Jede partielle Ableitung kann durch eine Differenz abgeschätzt werden, die durch Konvolution berechnet werden kann



- Beispiele:

$$\vec{G}(m,n) = \begin{pmatrix} G_x(m,n) \\ G_y(m,n) \end{pmatrix}, \quad G_x(m,n) \approx [f * g_x](m,n), \quad G_y(m,n) \approx [f * g_y](m,n).$$

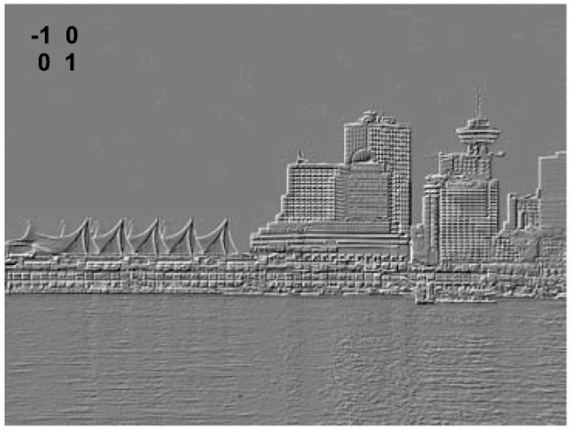
$$g_x = \begin{bmatrix} -1 & 1 \end{bmatrix} \quad g_y = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \qquad g_{R1} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad g_{R2} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

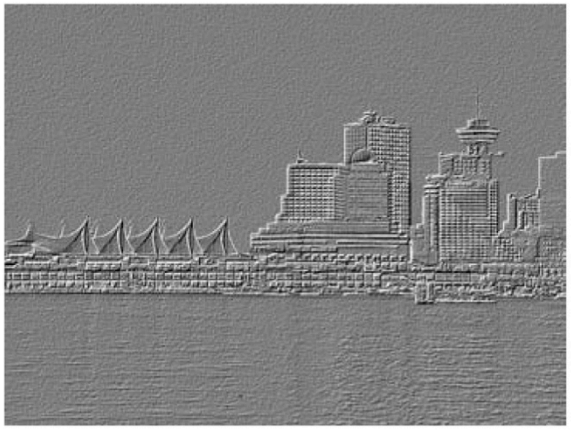
$$g_x = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad g_y = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

(g_{R1}, g_{R2}: Robert's Gradient)

Roberts Gradient

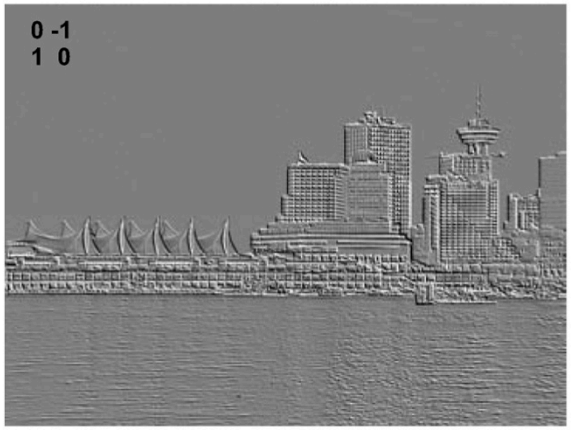
$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

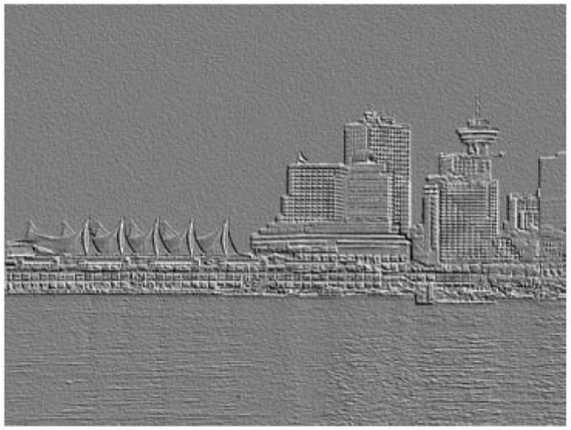


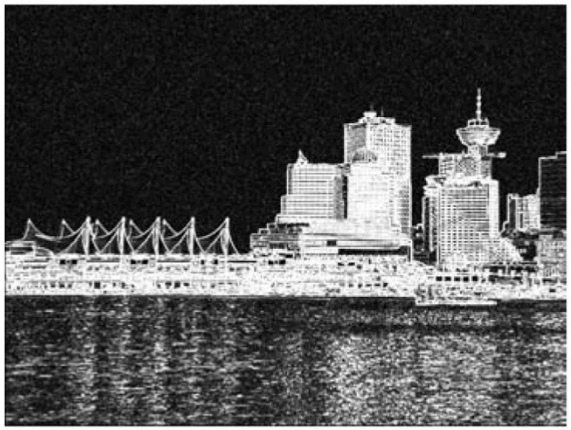


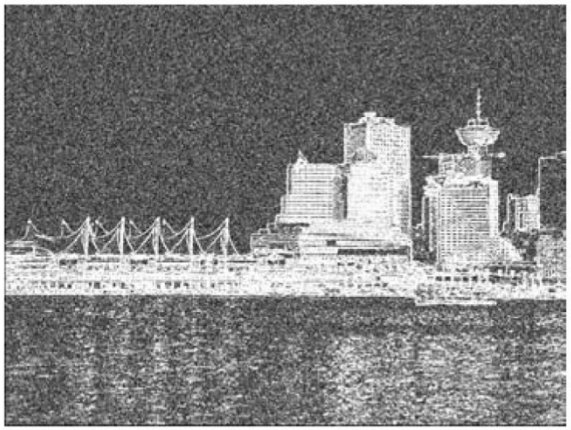
- rauscharmes Bild (links)
- verrauschtes Bild (rechts)
- Betrag des Gradienten (unten)
- Roberts-Operator rauschempfindlich

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$







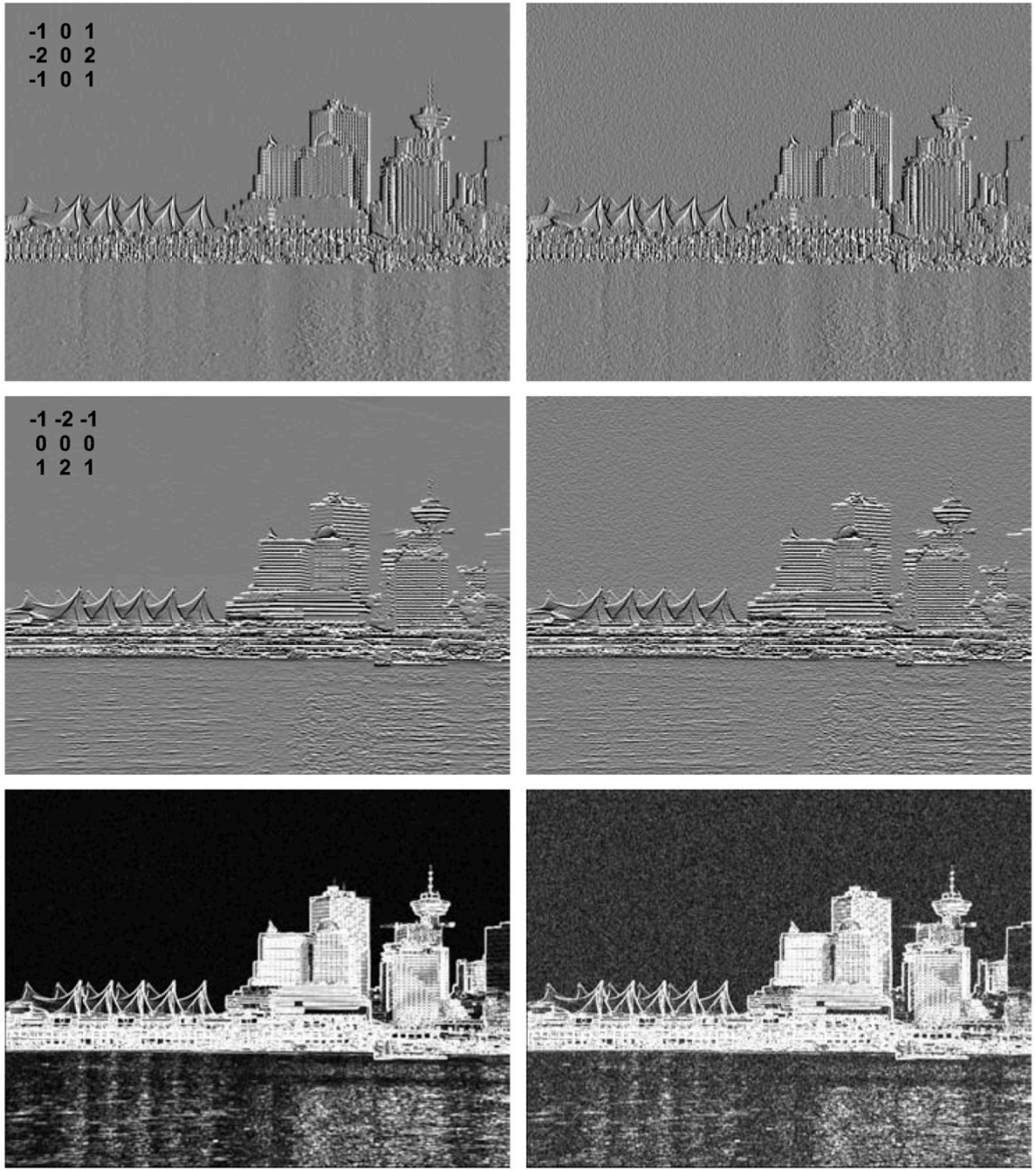


Sobel Operator

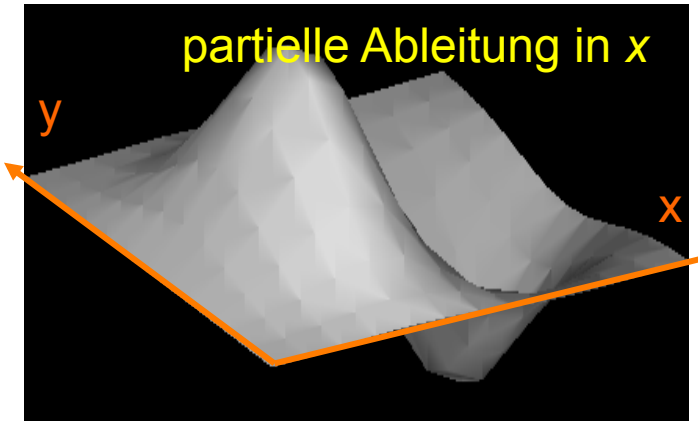
$$\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix} \quad \text{und} \quad \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$$

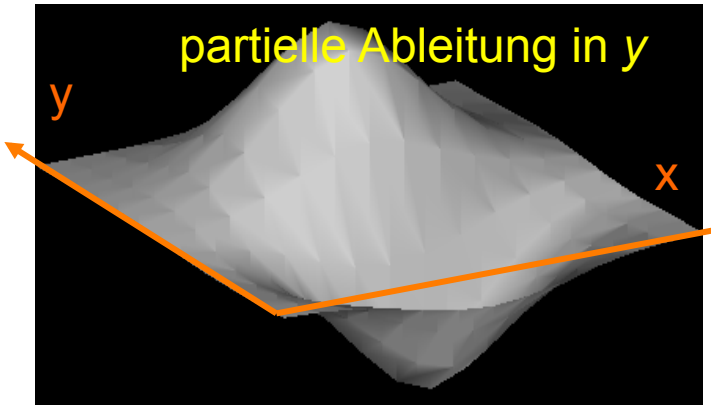
Faltungskern kombiniert Glättung und Differenzierung in einem Operator

Nicht perfekt rotationsinvariant



Ableitungen der Gaußfunktion





Ableitung der Gaußfunktion

- 2D-Gaußfunktion: $G(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$

- kombiniert Glättung und Ableitung
- erste Ableitung der Gaußfunktion in x-Richtung

$$\frac{\partial G(x,y)}{\partial x} = \frac{1}{2\pi\sigma^2}\left[\frac{-2x}{2\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right] = -\frac{x}{2\pi\sigma^4}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

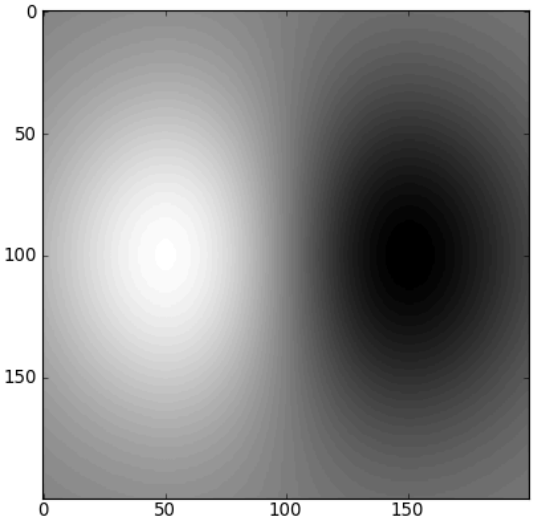
- erste Ableitung der Gaußfunktion in y-Richtung

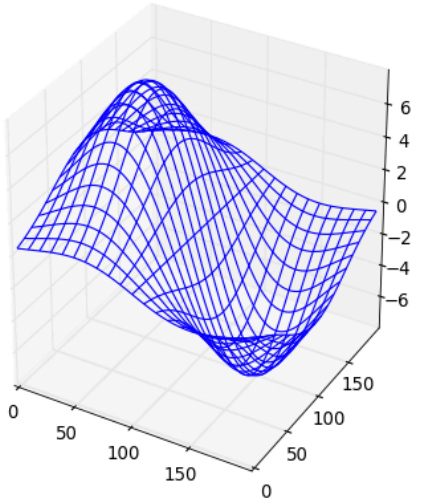
$$\frac{\partial G(x,y)}{\partial y} = \frac{1}{2\pi\sigma^2}\left[\frac{-2y}{2\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right] = -\frac{y}{2\pi\sigma^4}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

- separierbar?

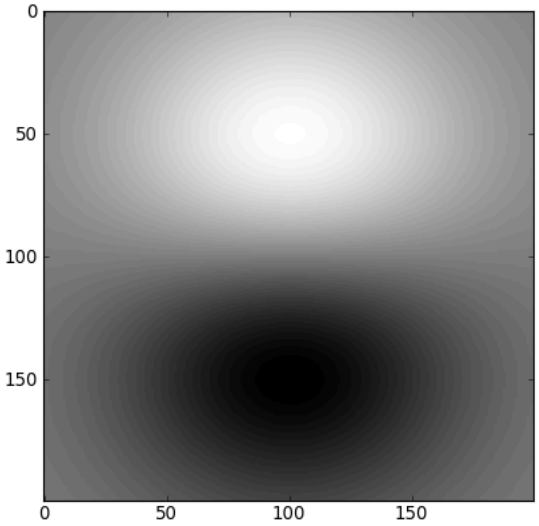
Ableitung der Gaußfunktion

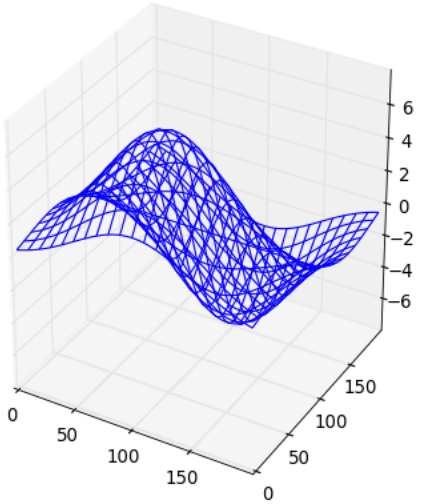
- erste Ableitung der Gaußfunktion in x-Richtung





- erste Ableitung der Gaußfunktion in y-Richtung





Sobel vs. Gauß

Was ist das?

Kirschoperator

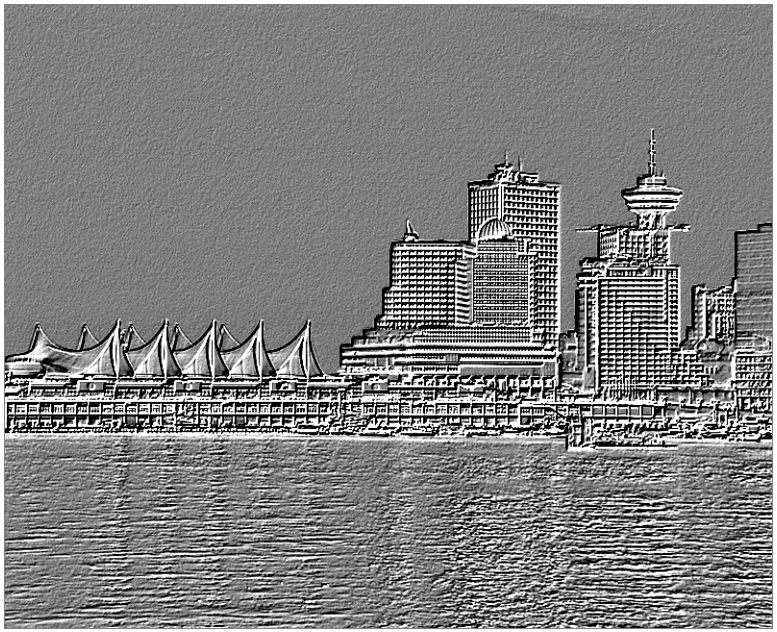
Kompassfilter

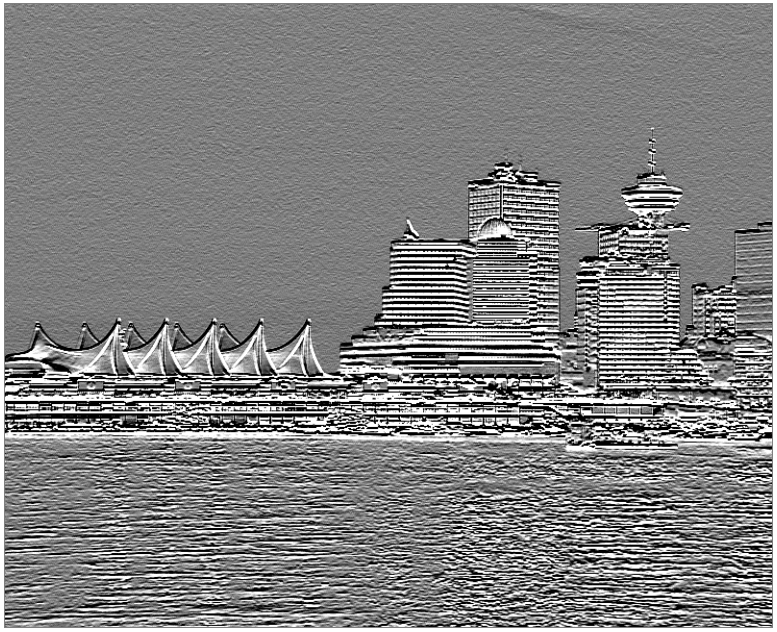
- Dienen der Hervorhebung von Kanten in einer bestimmten Richtung
- Kirsch-Operator
  - nicht-linear
  - sucht maximale Kantenstärke

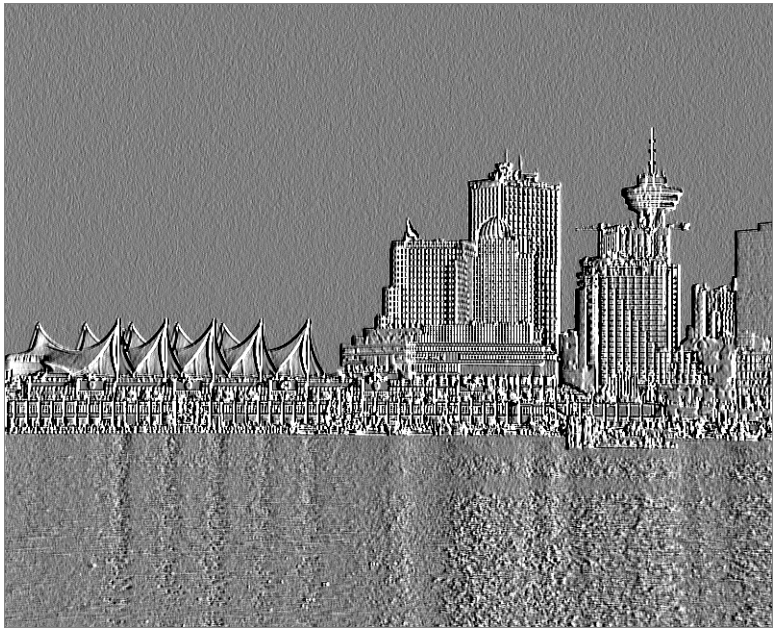
$$h_1 = \begin{pmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{pmatrix}, \quad h_2 = \begin{pmatrix} 3 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & -5 & 3 \end{pmatrix}, \quad h_3 = \begin{pmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{pmatrix} \ldots \quad h_8 = \begin{pmatrix} 3 & 3 & 3 \\ 3 & 0 & -5 \\ 3 & -5 & -5 \end{pmatrix}$$

$$h(n,m) = \max_{z=1,\ldots,8} \sum_{i=-1}^{1} \sum_{j=-1}^{1} h_z(i,j) \cdot f(n+i, m+j)$$

Kirschoperator



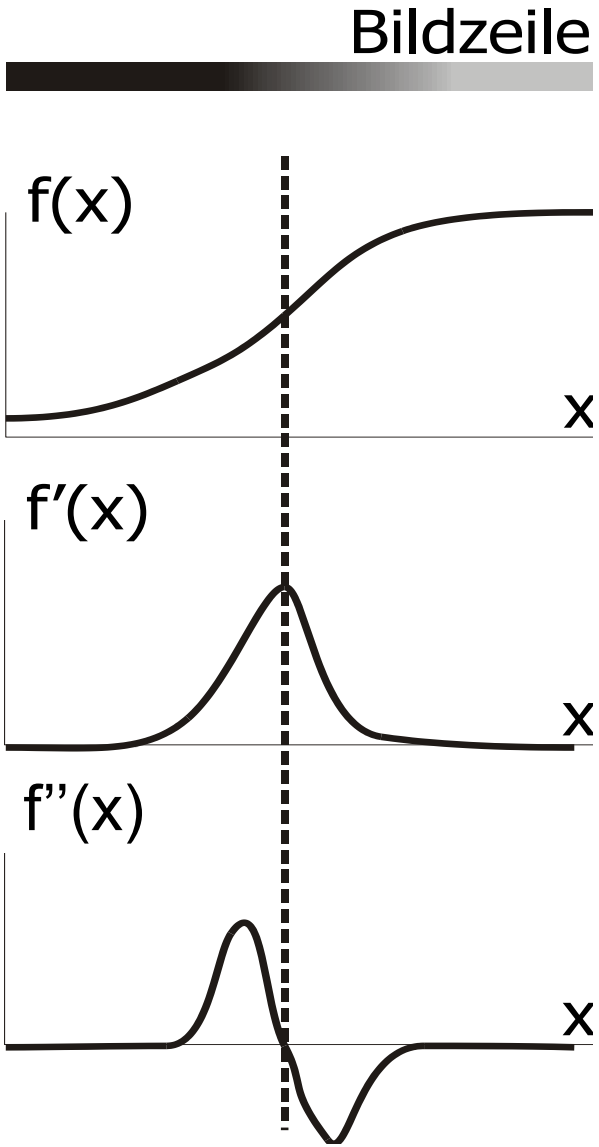




ZWEITE ABLEITUNG

Zweite Ableitung

- Vorzeichenwechsel ist leichter zu erkennen, als ein Minimum oder Maximum
  - Gradient (Länge) als Maß für die Wichtigkeit einer Kante
  - zweite Ableitung für den Ort der Kante (Nulldurchgang)
- Operatoren zur Berechnung der zweiten Ableitung
  - Laplace Filter
  - Marr-Hildreth Filter (LoG Filter, Mexican Hat)
  - DoG (Difference of Gaussians)



Laplace-Funktion

- Summe der partiellen zweiten Ableitungen

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

- Nulldurchgänge der Laplace-Funktion produzieren zusammenhängende Kurven entlang von Bildkanten
- Approximation durch Kombination einer doppelten Differenzbildung in x- und y-Richtung

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

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

Laplace-Funktion

- zweite Ableitung in x-Richtung

$$\frac{\partial^2 f}{\partial^2 x}(x) = \frac{\partial f}{\partial x}(x+1) - \frac{\partial f}{\partial x}(x)$$
$$= (f(x+1) - f(x)) - (f(x) - f(x-1))$$
$$= f(x-1) - 2f(x) + f(x+1) \Longrightarrow \begin{pmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

- zweite Ableitung in y-Richtung

$$\frac{\partial^2 f}{\partial^2 y}(y) = \frac{\partial f}{\partial y}(y+1) - \frac{\partial f}{\partial y}(y)$$
$$= (f(y+1) - f(y)) - (f(y) - f(y-1))$$
$$= f(y-1) - 2f(y) + f(y+1) \Longrightarrow \begin{pmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

Laplace-Funktion

- Summe der partiellen zweiten Ableitungen

$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

- Summe aller partiellen Ableitungen, um rotationsinvarianten Operator zu erhalten

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} + \frac{\partial^2 f(x,y)}{\partial x \partial y} + \frac{\partial^2 f(x,y)}{\partial y \partial x} : \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 1 \\ 0 & -2 & 0 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

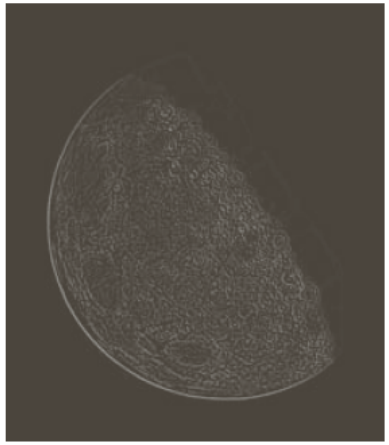
Laplace-Operator

- Nulldurchgänge des Laplace-Operators
- mittleres grau = 0, dunkle Pixel < 0, helle Pixel > 0

Schärfen eines Bildes mit Lapace-Operator

a
b c
d e

FIGURE 3.38
(a) Blurred image of the North Pole of the moon. (b) Laplacian without scaling. (c) Laplacian with scaling. (d) Image sharpened using the mask in Fig. 3.37(a). (e) Result of using the mask in Fig. 3.37(b). (Original image courtesy of NASA.)



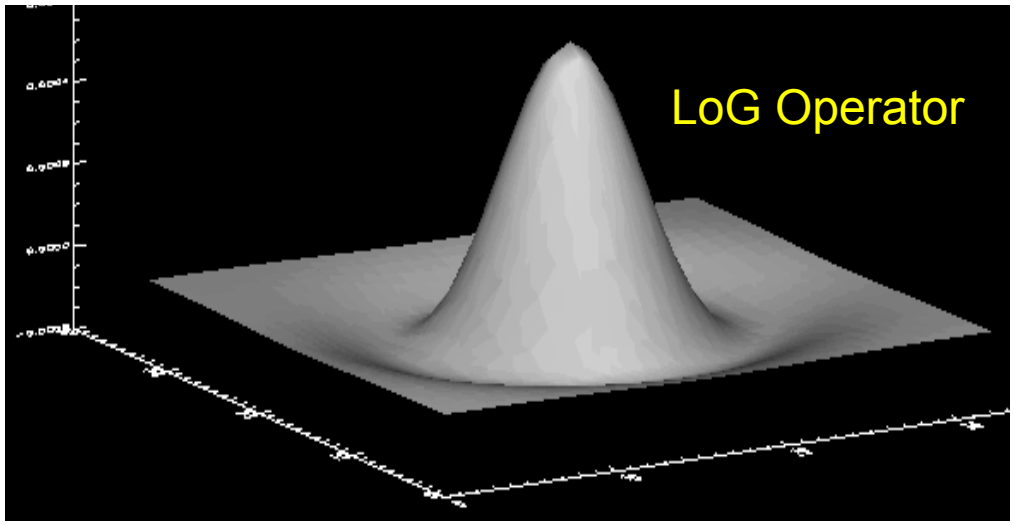
$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

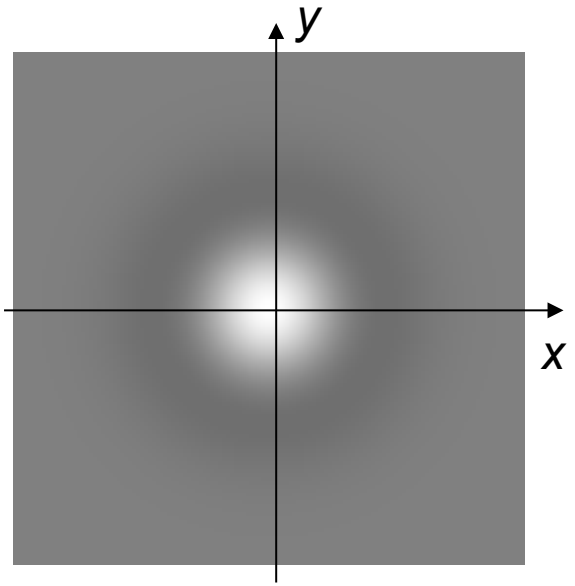
$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Abbildung: © R. C. Gonzalez & R. E. Woods, Digital Image Processing

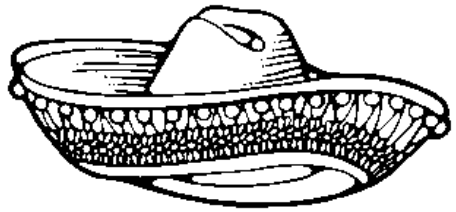
Marr-Hildreth-Filter = LoG-Filter

LoG-Filter: Laplacian-of-Gaussian,
d.h. der Faltung mit dem Laplacefilter geht eine Glättung mit einer Gaußfunktion voraus





$$LoG_\sigma(x,y) = -\frac{1}{\pi\sigma^4}\left(1-\frac{x^2+y^2}{2\sigma^2}\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$



Auch genannt: „Mexican hat“ filter

Laplace-Funktion der Gaußfunktion

- 2D-Gaußfunktion: $G(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$

- Laplace-Funktion der Gaußfunktion (zweite Ableitung)

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

$$= \frac{1}{2\pi\sigma^2}\frac{\partial}{\partial x}\left[\frac{-2x}{2\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right] + \frac{1}{2\pi\sigma^2}\frac{\partial}{\partial y}\left[\frac{-2y}{2\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right]$$

$$= \frac{1}{2\pi\sigma^2}\left[\frac{x^2}{\sigma^4} - \frac{1}{\sigma^2}\right]\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) + \frac{1}{2\pi\sigma^2}\left[\frac{y^2}{\sigma^4} - \frac{1}{\sigma^2}\right]\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

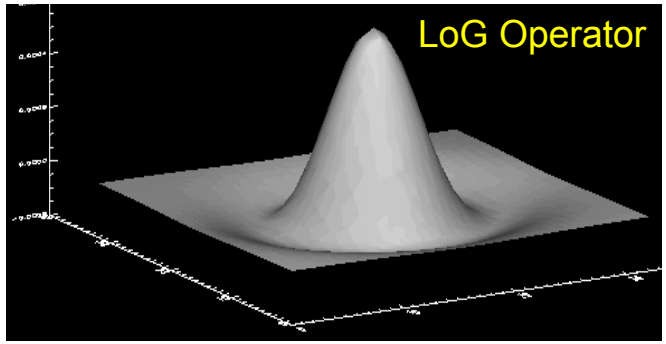
$$= \frac{1}{\pi\sigma^4}\left[\frac{x^2+y^2}{2\sigma^2} - 1\right]\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

Marr-Hildreth-Filter = LoG-Filter

- Vorteile der Laplace-Funktion der Gaußfunktion
  - Glättung und Kantenhervorhebung in einem Operator
  - wegen Glättung weniger empfindlich bei Rauschen
  - skalierbar
  - rotationsinvariant
  - separierbar
- Marr-Hildreth-Algorithmus zur Kantenerkennung

$$g(x,y) = \left[\nabla^2 G(x,y)\right] * f(x,y) = \nabla^2 \left[G(x,y) * f(x,y)\right]$$

1. nxn Gaußfilter anwenden
2. 3x3 Laplace-Filter anwenden
   (-8 umrahmt von 1en)
3. Zero-crossings finden
   (threshold verwenden)



Nulldurchgänge

Die Orte der Nulldurchgänge der zweiten Ableitung sind Ränder zusammenhängender Gebiete

Methode:

- Laplace-Operator
- Nulldurchgänge bestimmen:

  $\nabla^2(f(i,j)) \cdot \nabla^2(\text{shift}(f(i,j))) \leq 0$

  (shift: Verschiebung des Bildes um ein Pixel in x- und/ oder y-Richtung)

Laplace-Operator:

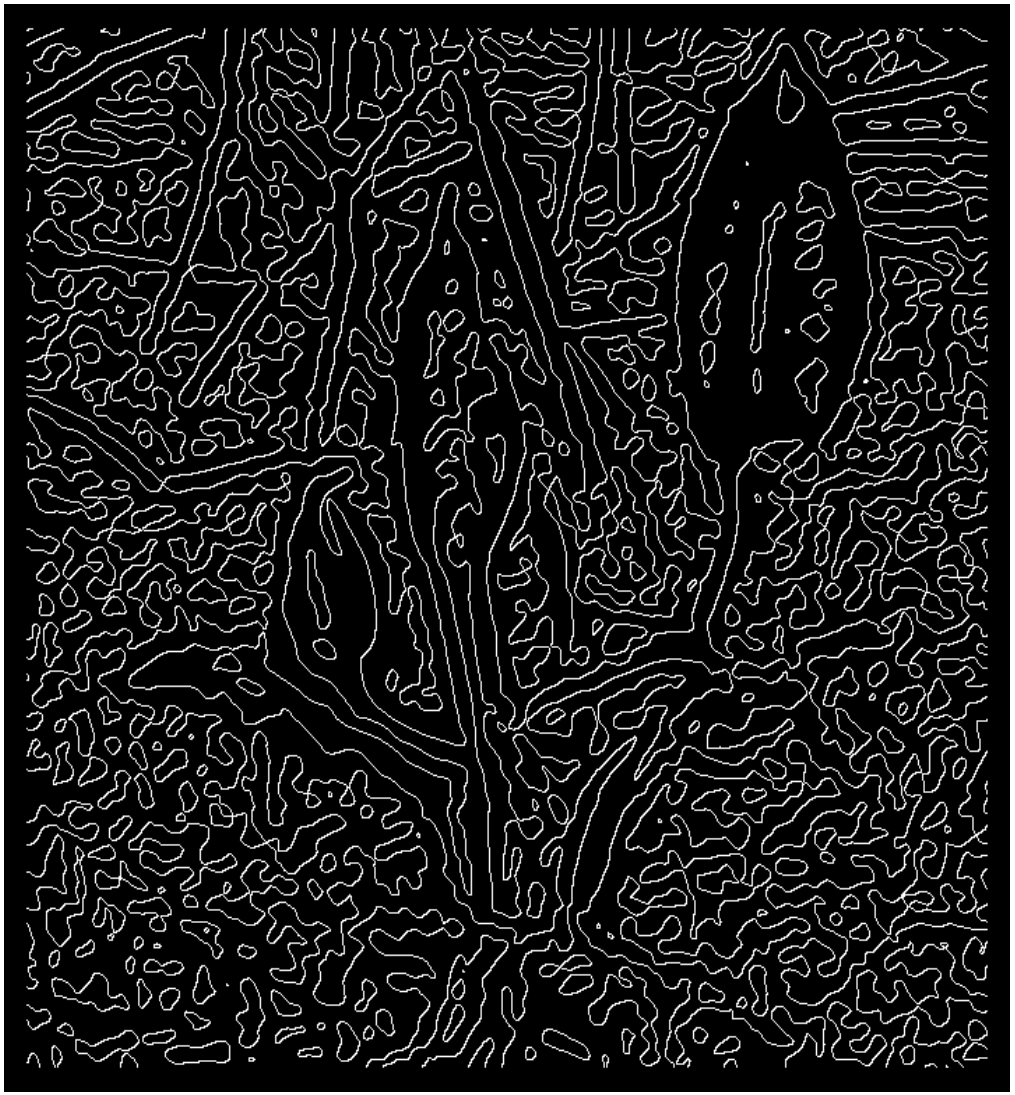
$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Nulldurchgänge

$$LoG_\sigma(x,y) = \ldots$$

Kombination des Laplace-Operators mit Glättungsoperator (z.B. als LoG-Operator) reduziert die Anzahl der Nulldurchgänge

Threshold auf Gradientenlänge reduziert Anzahl der Nulldurchgänge ebenfalls

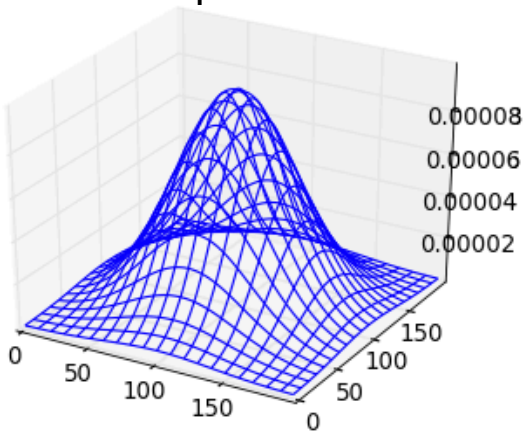


DoG (Difference of Gaussians)

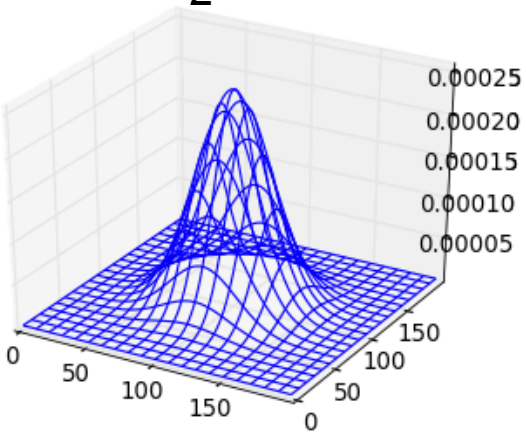
- Approximation des LoG durch DoG
- Subtraktion zweier Gaußfilter unterschiedlicher Varianz

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

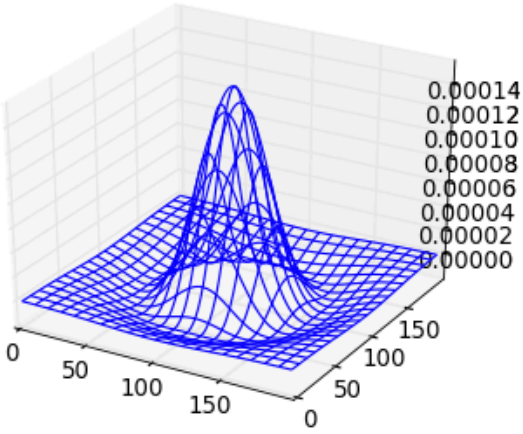
σ_1=40.0



σ_2=25.0



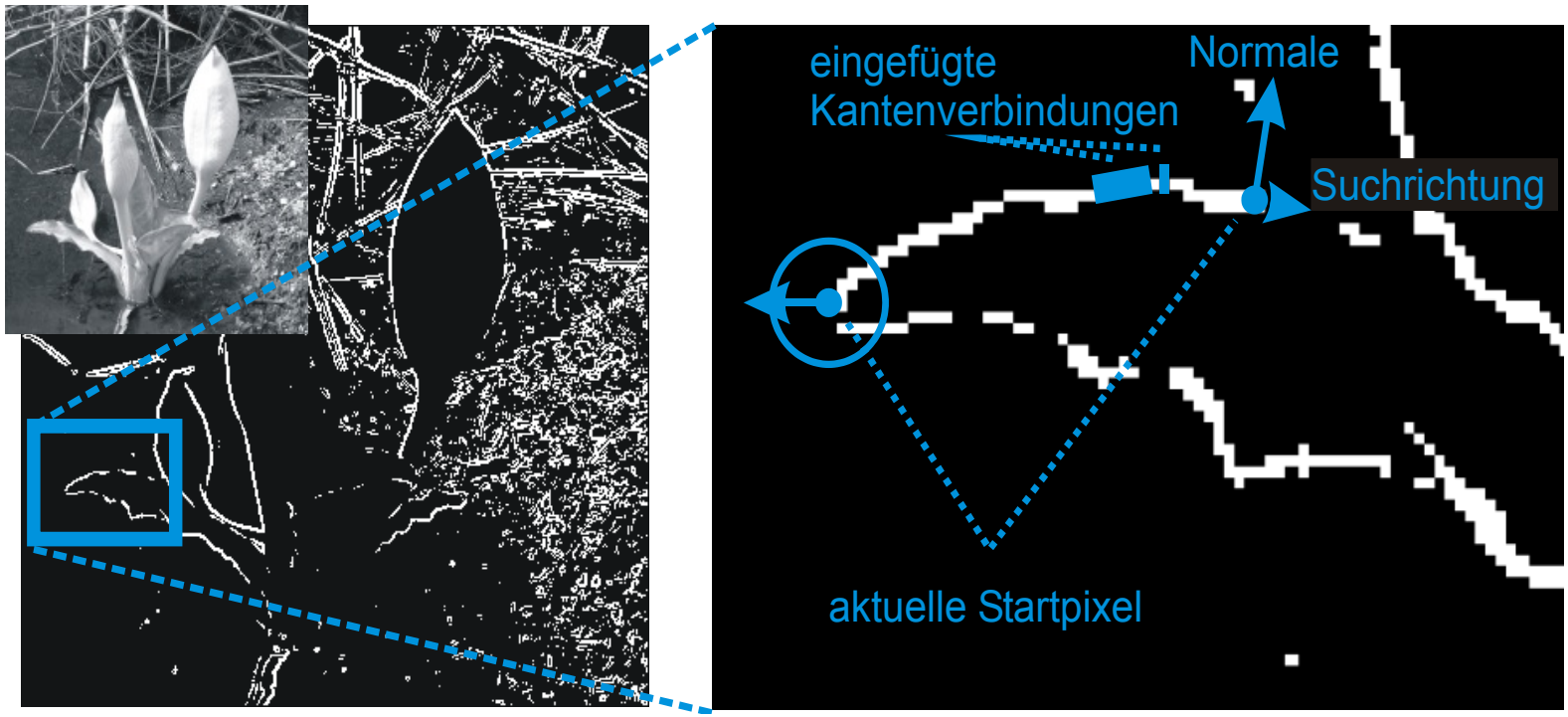
DoG



CANNY EDGE DETECTION

Edge Linking verbindet Kantenpixel zu Kantenzügen

- Starte Kantenzug mit starker Kante
- Betrachte Nachbarpixel orthogonal zur Gradientenrichtung
- Setzte Kantenzug fort auch bei schwacher Kante fort



Canny Edge Operator

- Ziele
  - Erkennung: alle Kanten finden und nur Kanten finden, d.h. vom Hintergrund unterscheiden
  - Lokalisierung: Kanten genau lokalisieren
  - Ansprechverhalten: für jede Kante genau eine Detektorantwort
- Canny Operator besteht aus Kantenhervorhebung und Erzeugung von Kantenzügen
  - Startpixel können nur Pixel sein, deren Gradientenlänge oberhalb einer Signifikanzschwelle T1 liegt
  - Neue Kantenpixel werden in den Kantenzug eingefügt, wenn ihre Gradientenlänge größer als T2 ist (T2<T1)

Canny Edge Detection – Algorithmus

1. Bild mit Gaußfilter glätten
2. Gradientenoperator (Sobel, Prewit, etc.) anwenden
   – ergibt Gradientenlänge und -richtung
3. Non-maxima Unterdrückung anwenden
   a. Unterdrücke Punkt, falls ein 8-Nachbar in Gradientenrichtung (0°, 45°, 90°, 135°) größere Gradientenlänge hat
4. Doppeltes Tresholding anwenden ($T_1 > T_2$)
   a. Wähle Punkt als Startpunkt eines Kantenzugs, falls Gradientenlänge ≥ T_1 (starke Kante)
   b. Nimm nächsten Punkt entlang der Kante in Kantenzug auf, falls dessen Gradientenlänge ≥ T_2 (schwache Kante, falls < T_1)

- Verhalten hängt von T_1, T_2, sowie Gaußfilter ab

Non-Maxima Unterdrückung

- Quantisieren der Kantenorientierung in eine von vier Richtungen: 0°, 45°, 90°, 135°
- Falls Gradientenlänge kleiner als einer seiner Nachbarn in Kantenorientierung, dann unterdrücken, sonst behalten

Doppeltes Thresholding (Hysterese)

- starke Kanten: $L[x,y] \geq T_1$
- schwache Kante: $T_1 > L[x,y] \geq T_2$

- typischerweise $T_1 / T_2 = 2..3$

- schwache Kanten überleben nur, wenn sie starke Kanten fortsetzen

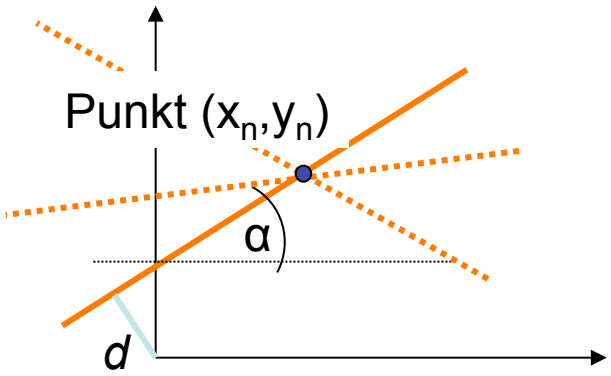
HOUGH TRANSFORMATION

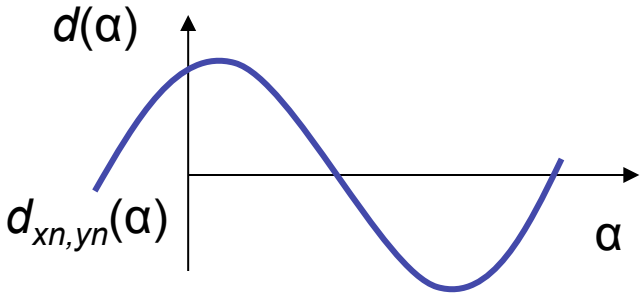
Hough Transformation (HT)

- Modellbasierte Suche
  - Top-down: Modell einer Regionengrenze (z.B. Linie, Kreis) ist vorgegeben
  - Bottom-up: Information über mögliche Kantenorte ist gegeben (z.B. durch Gradientenlänge)
  - Zielsetzung: Orte finden, an denen Modell im Bild auftritt
- Hough-Transformation
  - entwickelt für Geraden, erweiterbar für beliebige Formen
  - Voting(Abstimmungs)-Mechanismus: jeder Ort stimmt abhängig von Gradientenlänge für Modell; 2D-Histogramm
  - Hough-Raum ist definiert durch Parameter des Modells (bei Kreis z.B. Mittelpunkt und Radius)
  - Anzahl/Stärke der Votes für einen Satz von Parametern bestimmt Evidenz für Auftreten der Instanz des Modells

Hough Transformation für Geraden

Suche von Geraden in einem Binärbild
Geradenrepräsentation: $x \cos(\alpha) + y \sin(\alpha) - d(\alpha) = 0$





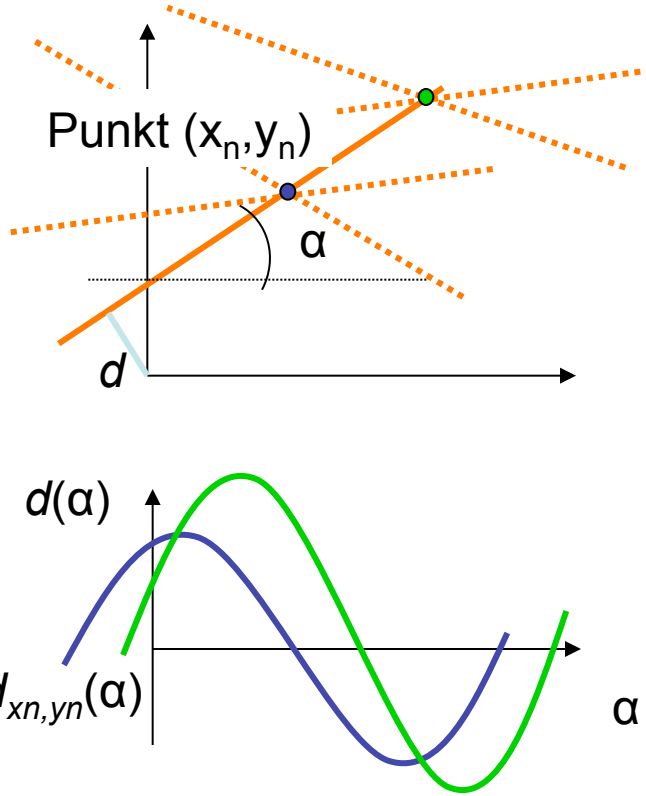
Hough Transformation:

Suche alle Parameter (α,d) für Geraden, die durch einen Punkt (x_n,y_n) gehen

$$d(\alpha) = x_n \cos(\alpha) + y_n \sin(\alpha)$$

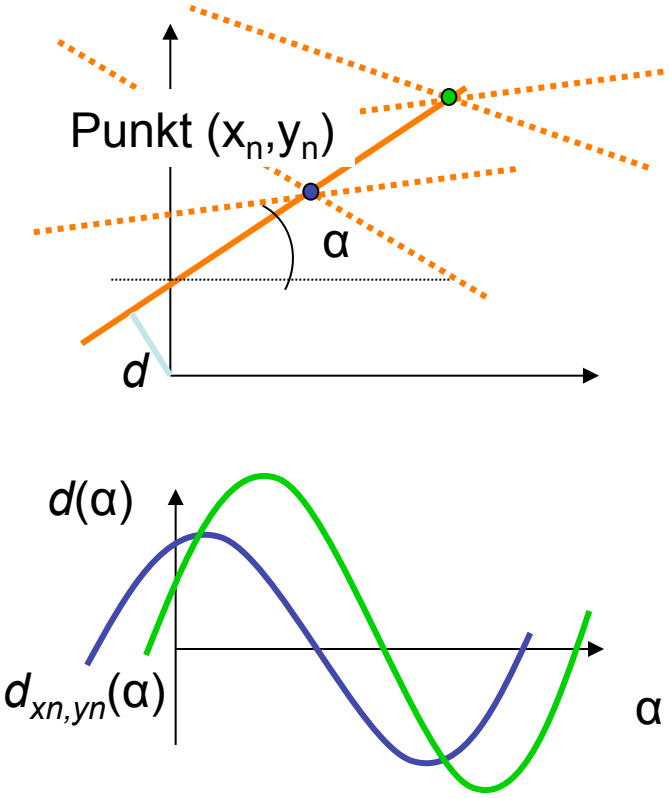
Der Raum, der durch (α,d) aufgespannt wird, heißt Hough-Raum

Berechnung der HT für Binärbilder



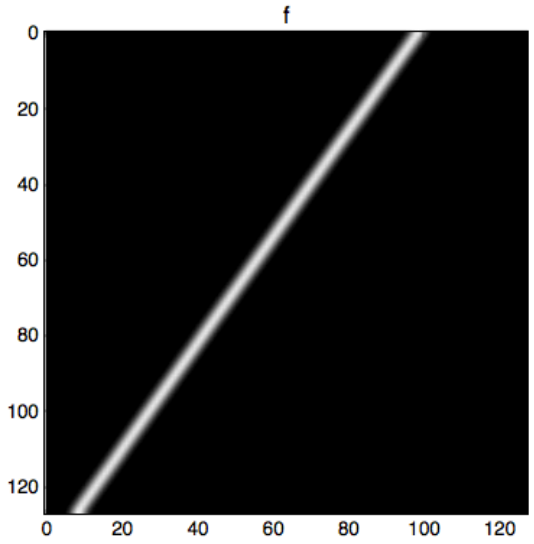
- Erzeugung eines SW-Kantenbilds durch Schwellenwertsetzung auf Gradientenlängen
- Diskretisierung des (α,*d*)-Raums (Zerlegung in Akkumulatoren)
- Für jeden Punkt x_n,y_n wird eine Kurve im (α,*d*)-Raum diskretisiert
- Jeder Akkumulator wird inkrementiert, sobald eine Kurve durch ihn verläuft
- Parameter von Linien im Ortsraum sind durch (α,*d*)-Kombinationen gegeben, deren Wert (Stimmenanzahl, votes) nach Ausführung der Transformation am höchsten sind

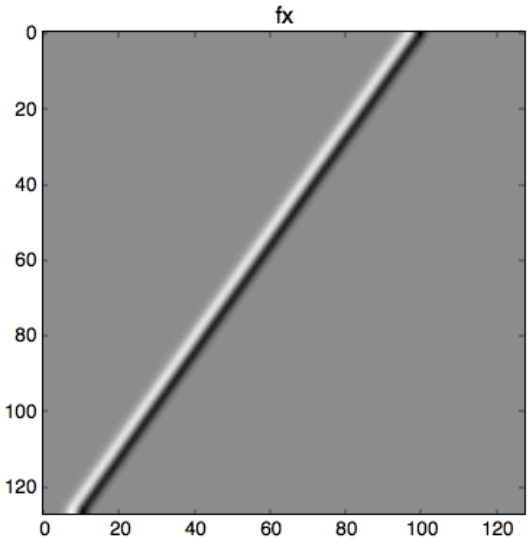
Berechnung der HT für Grauwertbilder

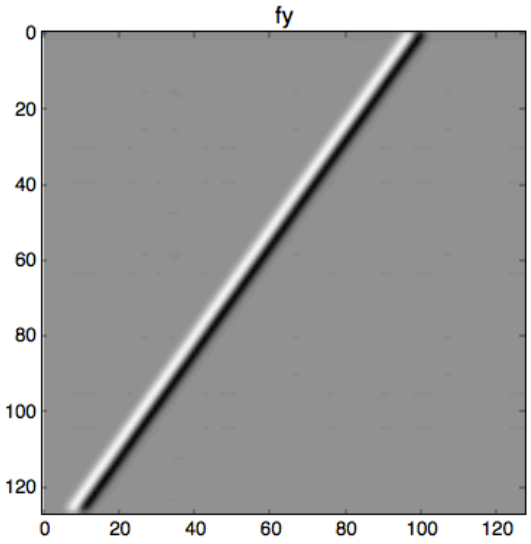


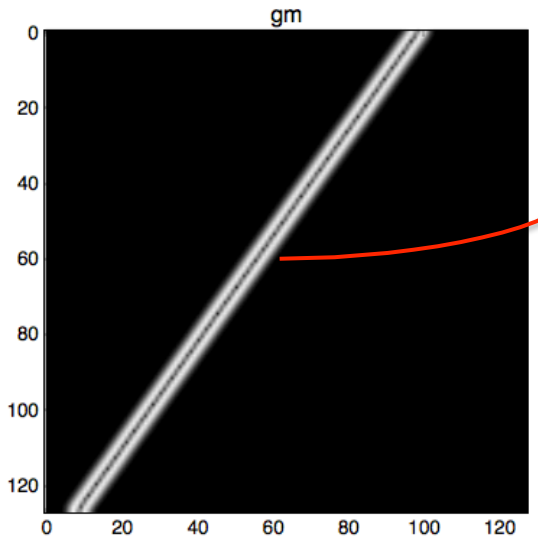
- Erzeugung eines Kantenbilds durch Anwendung des Sobel-Operators
- Diskretisierung des (α,*d*)-Raums (Zerlegung in Akkumulatoren)
- Für jeden Punkt x_n,y_n wird eine Kurve im (α,*d*)-Raum diskretisiert
- Für jeden Punkt werden entsprechend der Gradientenlänge die passenden Akkumulatoren erhöht
- Parameter von Linien im Ortsraum sind durch (α,*d*)-Kombinationen mit hohen Werten gegeben

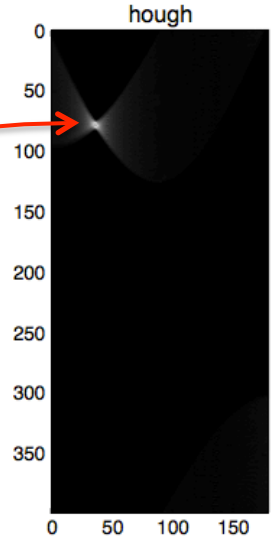
α = 0..179

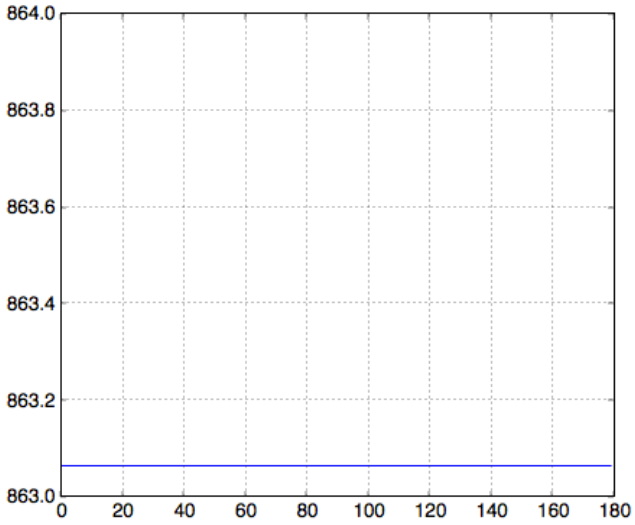

f
0
20
40
60
80
100
120
0 20 40 60 80 100 120


fx
0
20
40
60
80
100
120
0 20 40 60 80 100 120

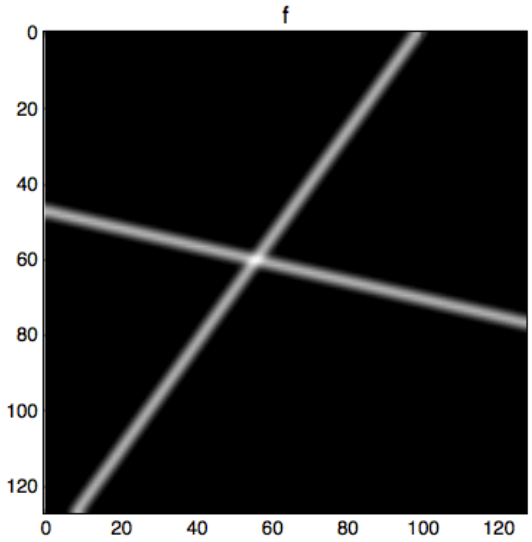

fy
0
20
40
60
80
100
120
0 20 40 60 80 100 120

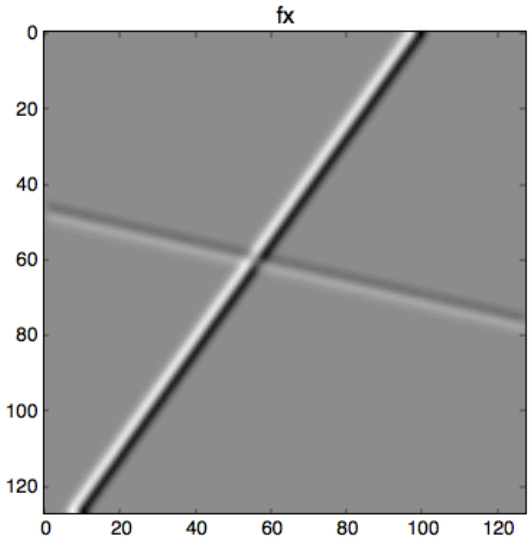

gm
0
20
40
60
80
100
120
0 20 40 60 80 100 120

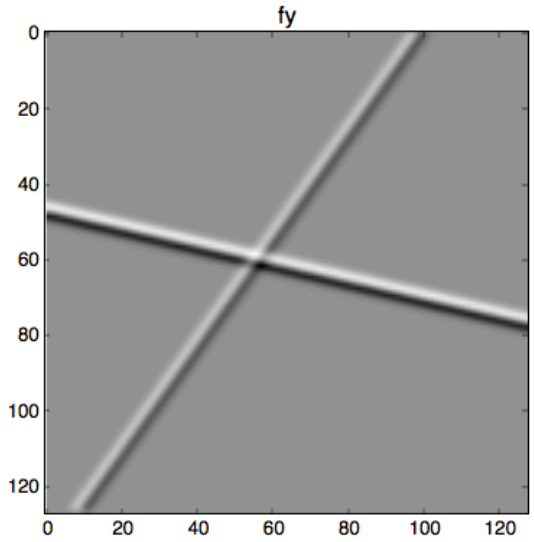

hough
0
50
100
150
200
250
300
350
0 50 100 150

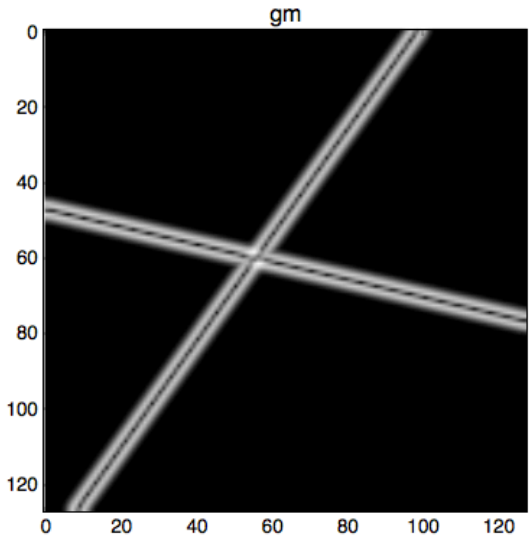

864.0
863.8
863.6
863.4
863.2
863.0
0 20 40 60 80 100 120 140 160 180

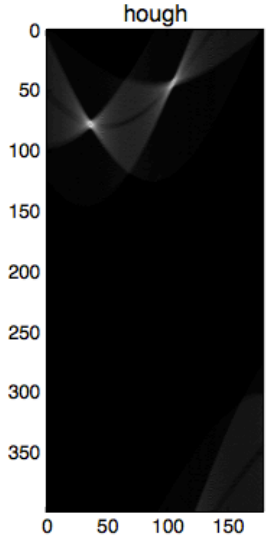
α = 0..179

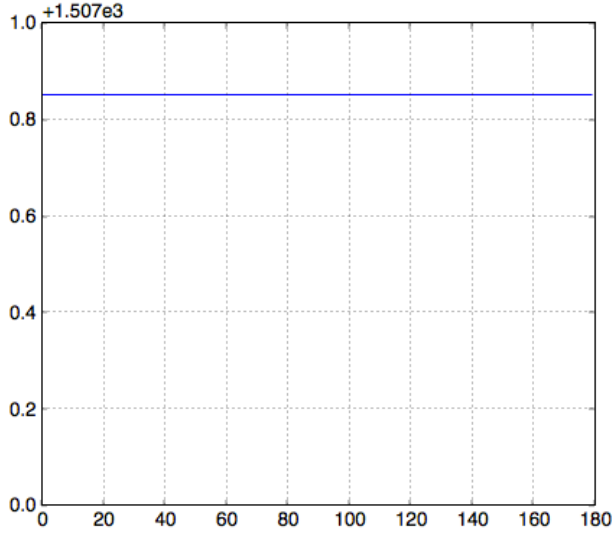

f
0
20
40
60
80
100
120
0
20
40
60
80
100
120


fx
0
20
40
60
80
100
120
0
20
40
60
80
100
120

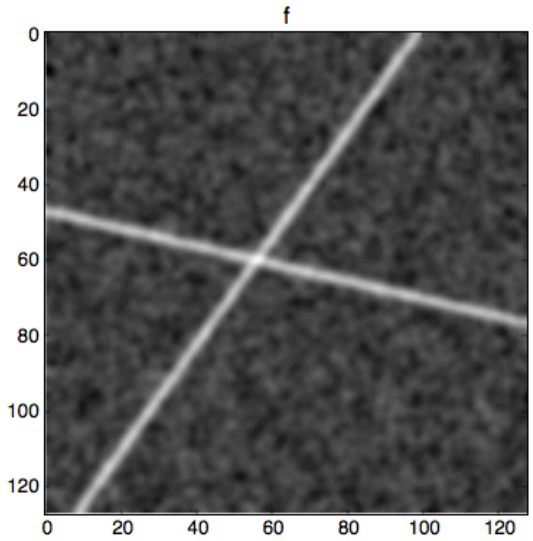

fy
0
20
40
60
80
100
120
0
20
40
60
80
100
120

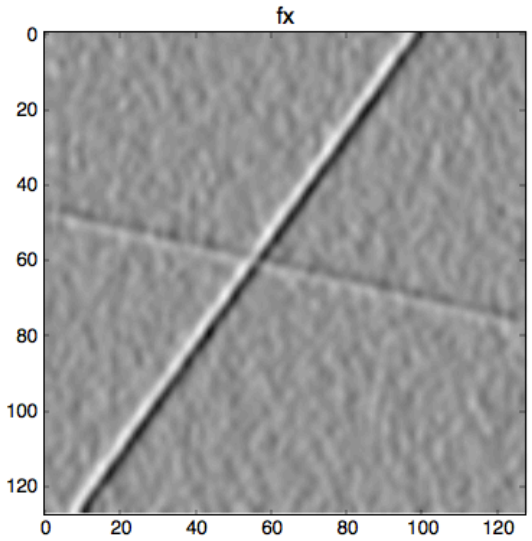

gm
0
20
40
60
80
100
120
0
20
40
60
80
100
120

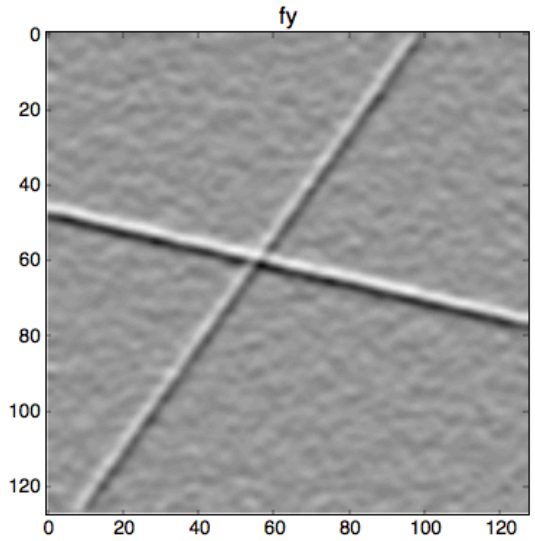

hough
0
50
100
150
200
250
300
350
0
50
100
150

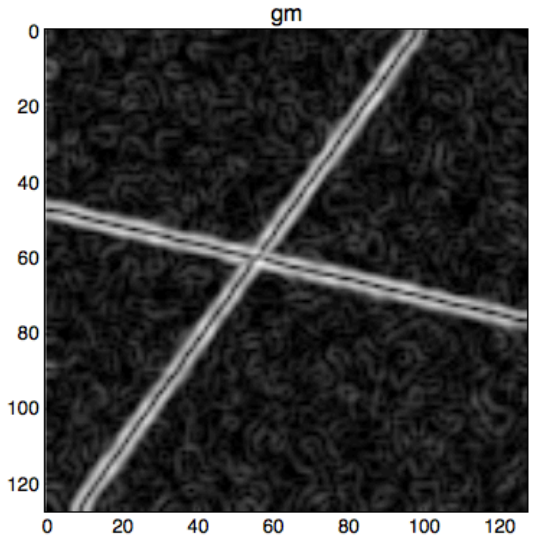

+1.507e3
1.0
0.8
0.6
0.4
0.2
0.0
0
20
40
60
80
100
120
140
160
180

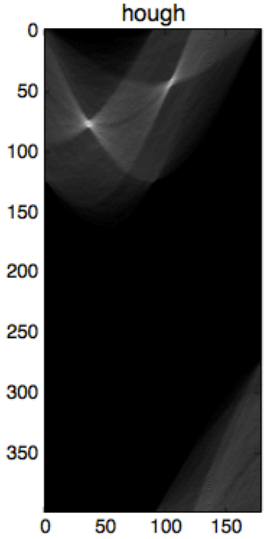
α = 0..179, Rauschen

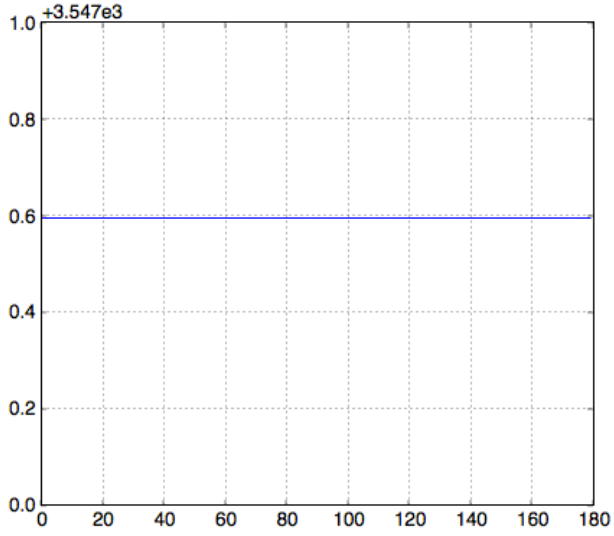




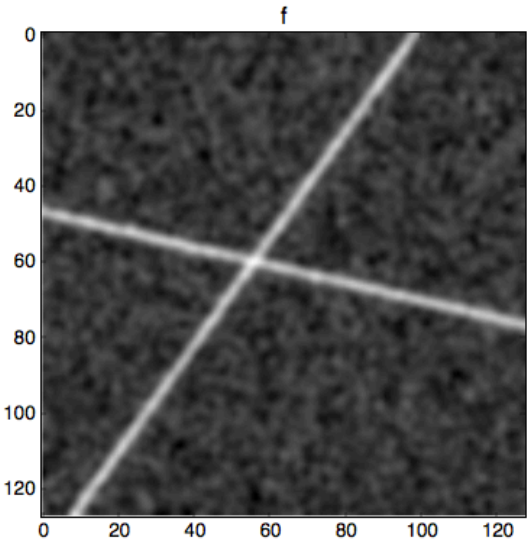




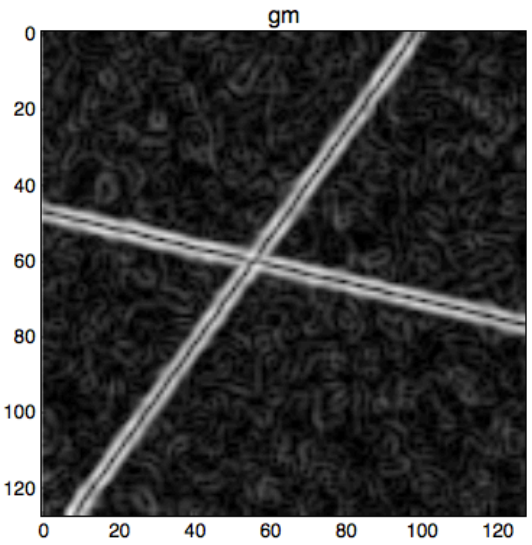


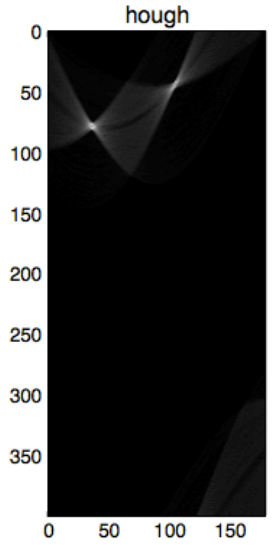


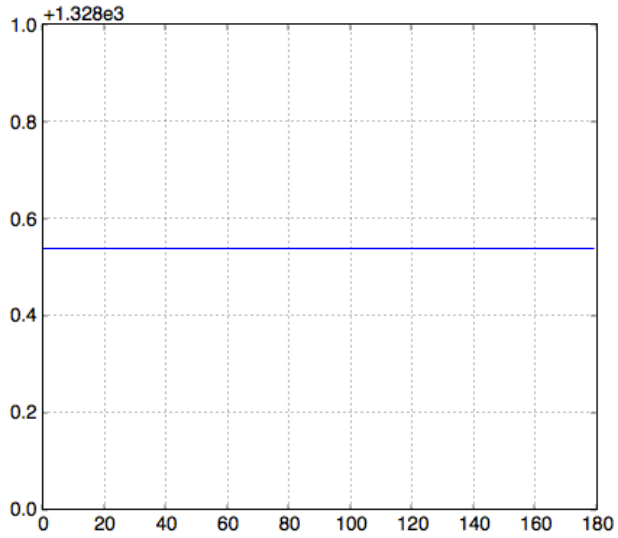
α = 0..179, Rauschen, gm > $2gm_{mean}$



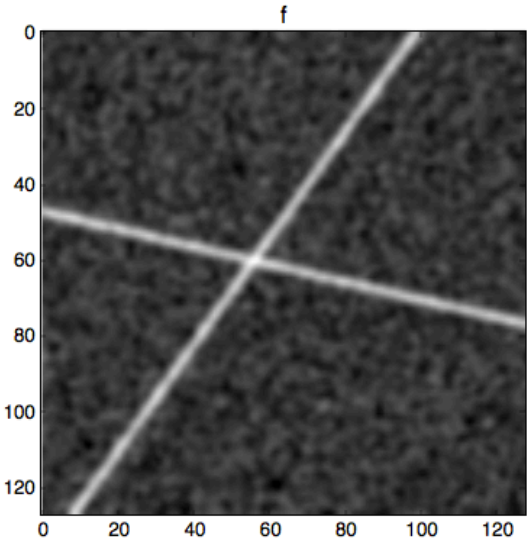
- Nur Gradienten, mit Länge größer der doppelten Durchschnittslänge berücksichtigen
- gm > $2gm_{mean}$





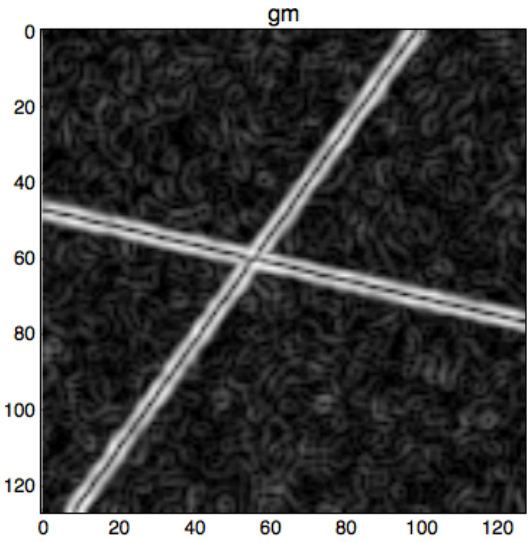


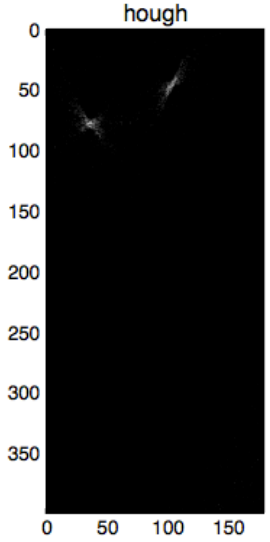
α = 0..179, Rauschen, Orientierung

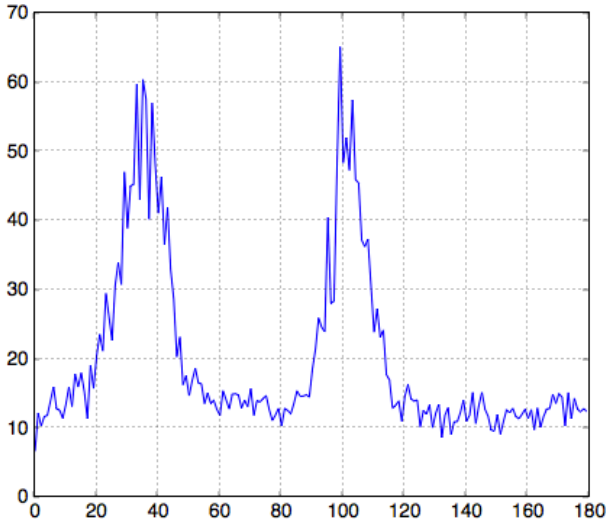


- Berücksichtigung der Gradienten-orientierung (go)
- Jeder Punkt erzeugt einen Eintrag im Houghraum (gm = gradient magnitude)

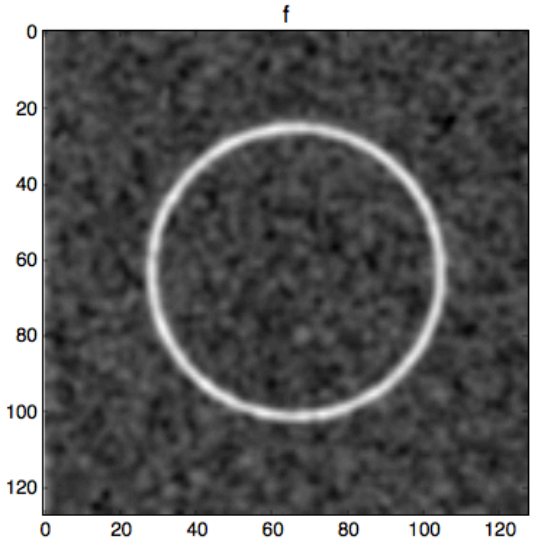
  hough[go[x,y]] = hough[go[x,y]] + gm(x,y)



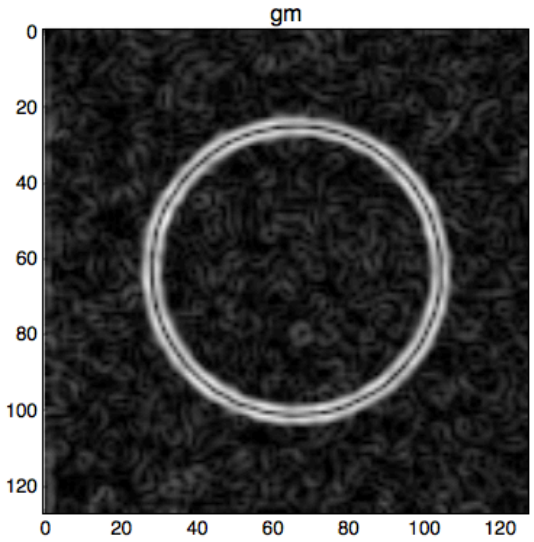


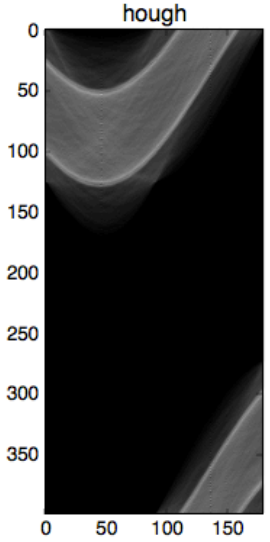


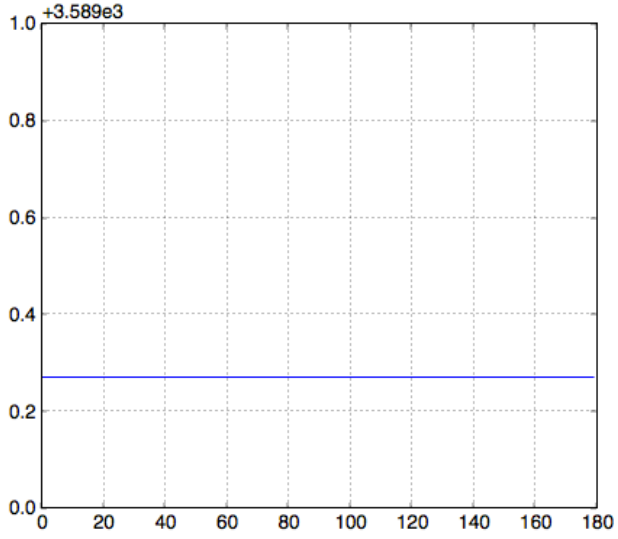
α = 0..179



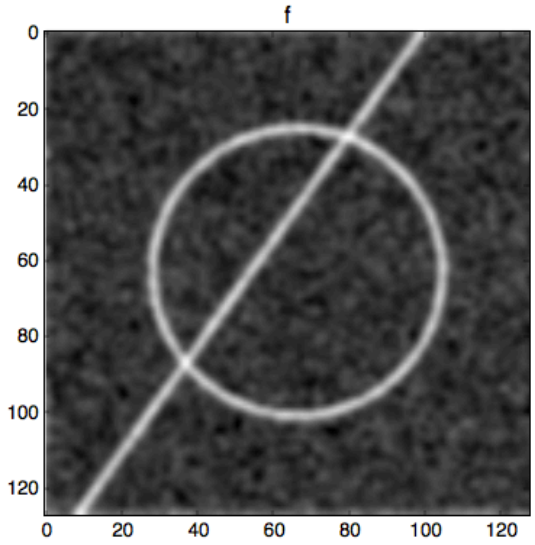
- Modell (Gerade) stimmt nicht mit Daten überein
- kein Peak im Houghraum

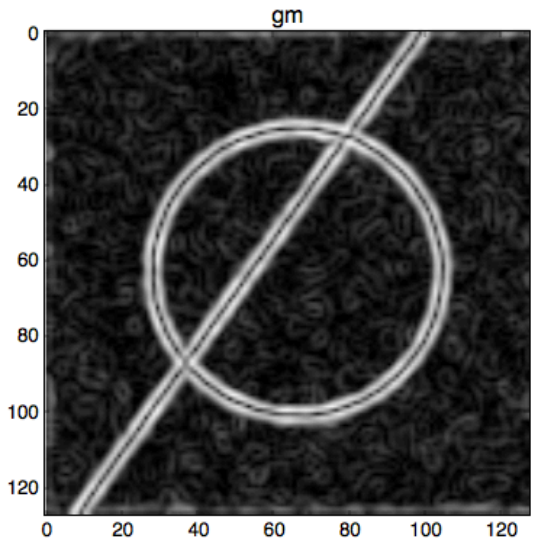


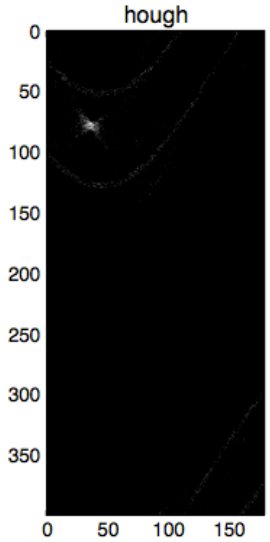


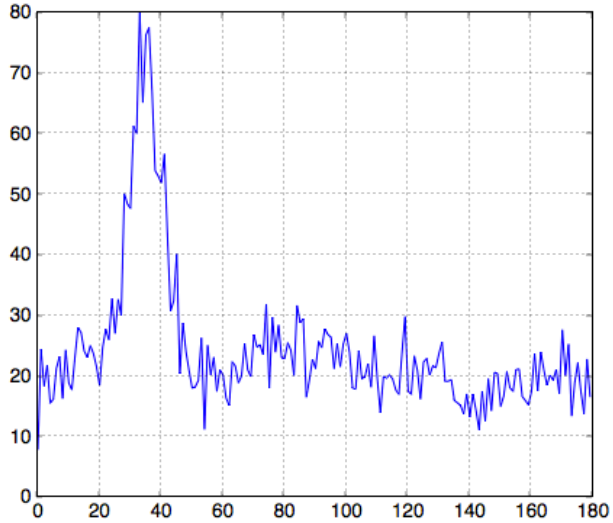


α = 0..179

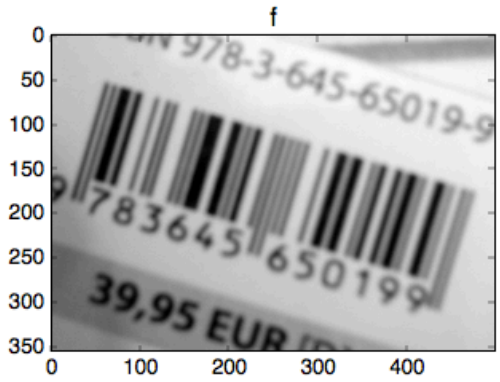

f
0
20
40
60
80
100
120
0
20
40
60
80
100
120

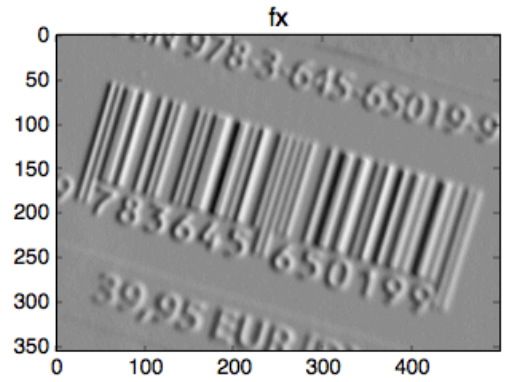

gm
0
20
40
60
80
100
120
0
20
40
60
80
100
120

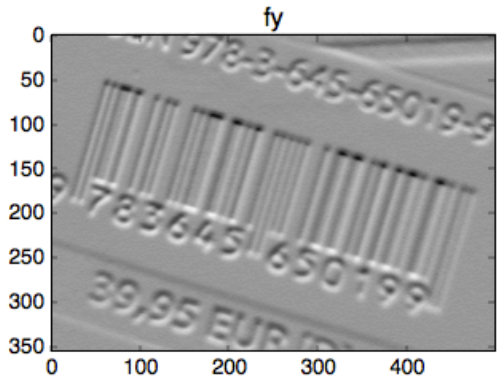

hough
0
50
100
150
200
250
300
350
0
50
100
150

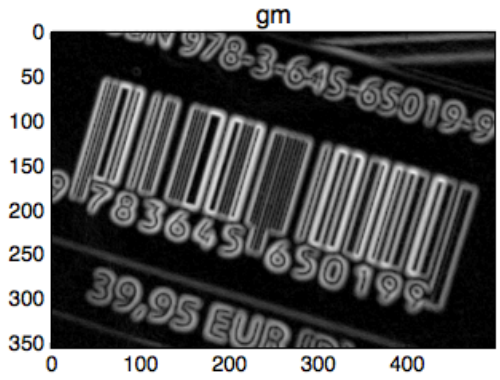

80
70
60
50
40
30
20
10
0
0
20
40
60
80
100
120
140
160
180

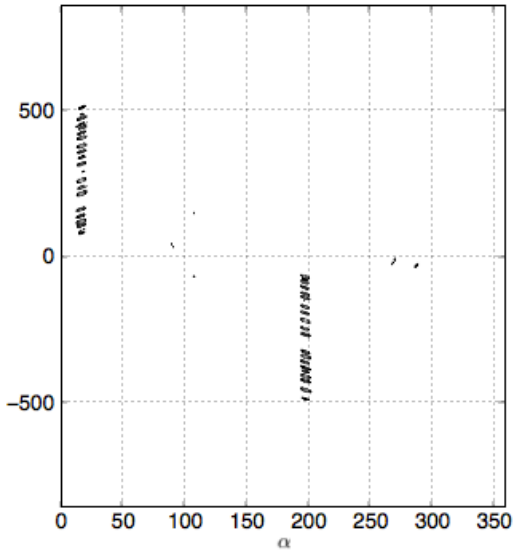
Hough Transformation von Barcodes

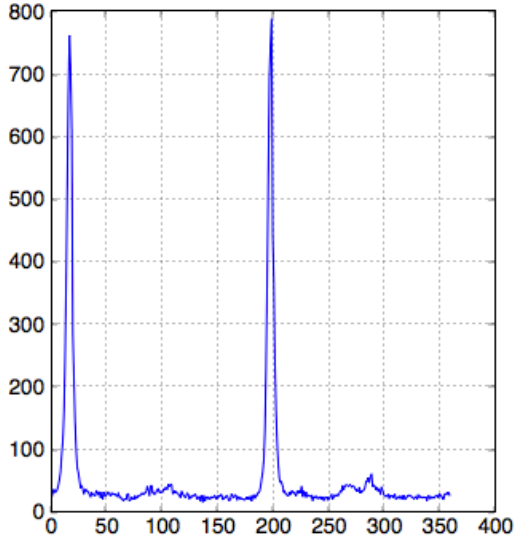






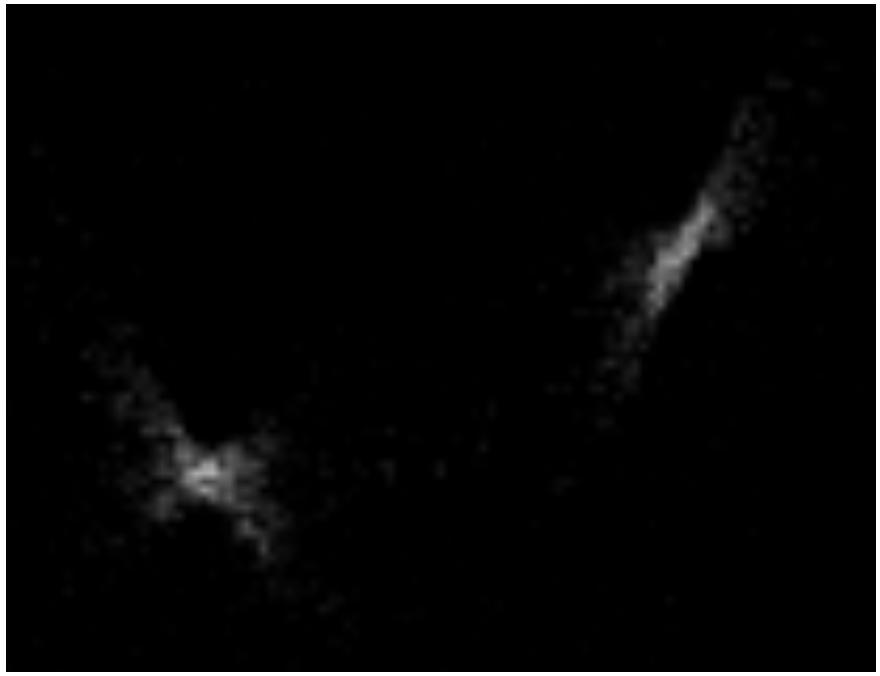




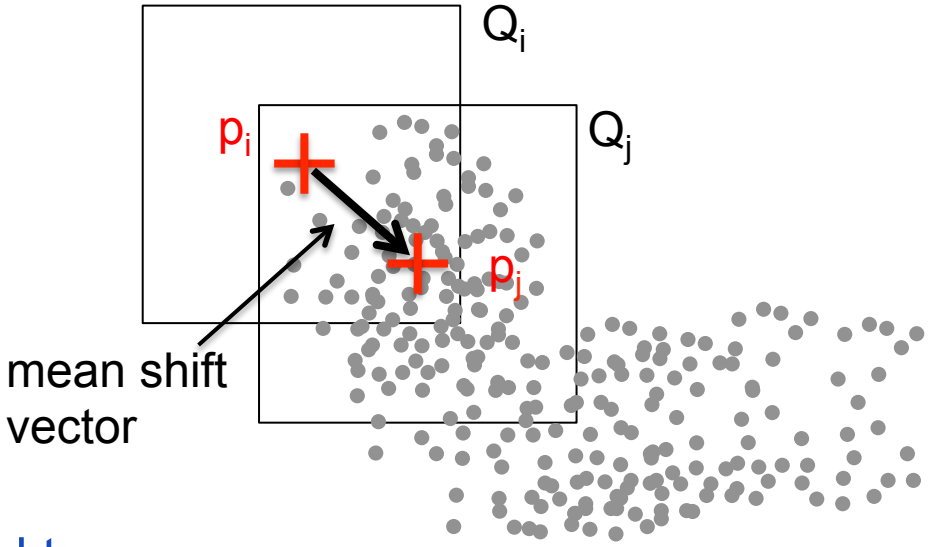


Lokalisieren von Maxima im Hough-Raum

- Lokalisieren der Maxima-Zentren im Hough-Raum
- Mean-Shift Algorithmus zum lokalisieren der Maxima (mode detection) einer Wahrscheinlichkeitsdichte von der diskrete Samples gegeben (Fukunaga & Hostetler, 1975)



Mean-Shift

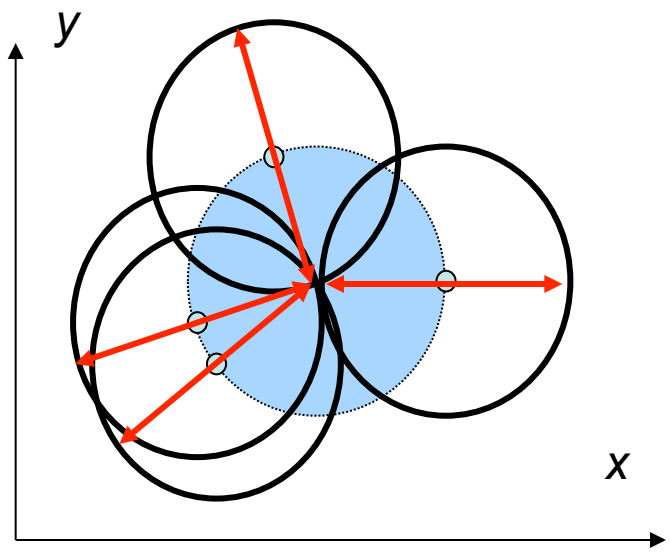


- Bewegen eines Punktes zum Schwerpunkt einer Umgebung
- Algorithmus (in 2D)
  1. Zentriere Quadrat Q_i an Punkt p_i
  2. Berechne Schwerpunkt p_j aller Punkte in Q_i
  3. falls $||p_i - p_j|| > \delta$, dann $p_i := p_j$ und gehe zu 1
- Verbesserung durch Gewichtung mit Gaußkern

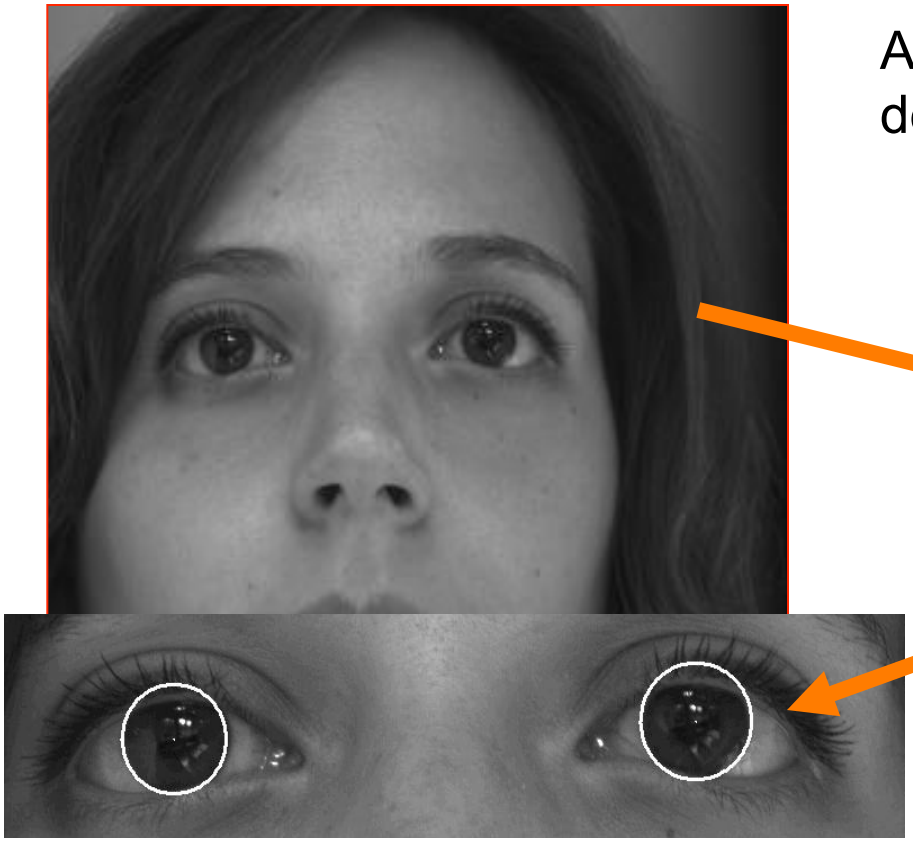
$$p_x(x_i, y_i) = \frac{\sum_{(x,y) \in Q_i(x_i,y_i)} G(x - x_i, y - y_i) \cdot x \cdot h(x,y)}{\sum_{(x,y) \in Q_i(x_i,y_i)} h(x,y) \cdot G(x - x_i, y - y_i)}, \quad G(x,y) = \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

Hough Transformation für Kreise

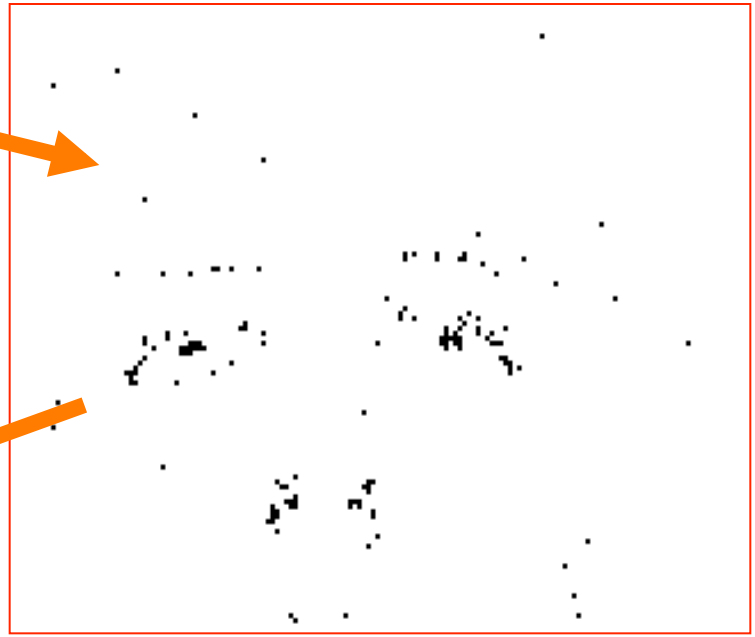
- Kreisgleichung für Kreis mit Mittelpunkt (x_c, y_c) und Radius r: $(x-x_c)^2 + (y-y_c)^2 - r^2 = 0$
- Falls der Radius bekannt ist, ist nur der Verschiebevektor (x_c, y_c) gesucht
  - Hough-Raum = Ortsraum
  - Um jeden Kantenpunkt wird ein Kreis mit Radius r diskretisiert
- Beschleunigung: Akkumulator wird nur in Distanz r in und entgegen der Gradientenrichtung inkrementiert



Hough Transformation für Kreise



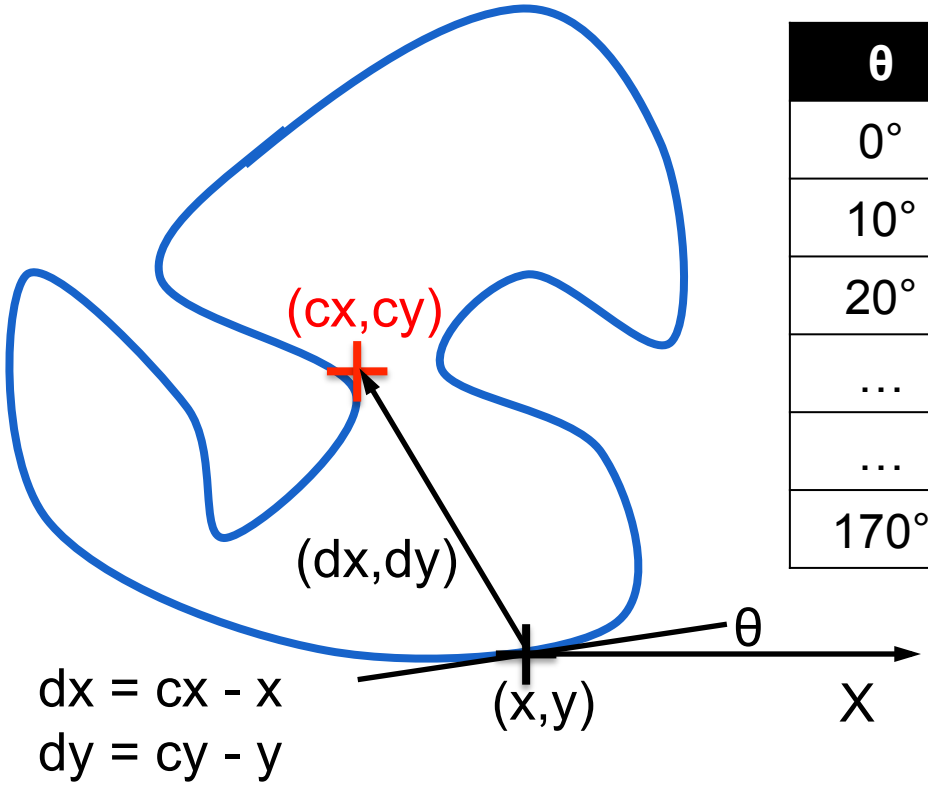
Akkumulatorzellen mit mehr als 33% der maximalen Stimmenanzahl



selektierte Kreise mit r = 4.5mm

Generalisierte Hough Transformation

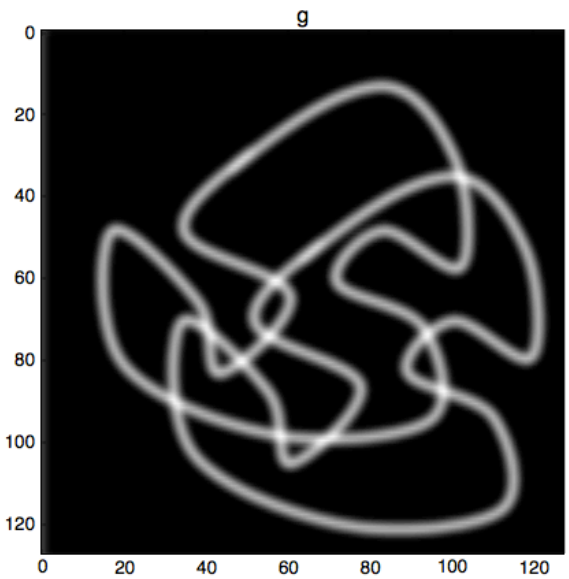
- Hough Transformation für beliebige Formen

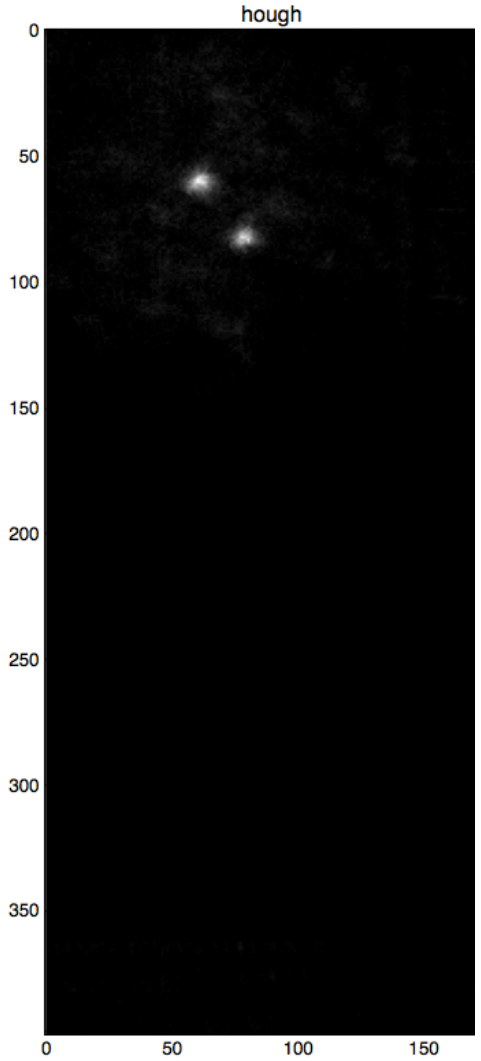


θ	R
0°	(dx_1,dy_1)
10°	(dx_2,dy_2), (dx_3,dy_3), (dx_5,dy_6)
20°	
…	…
…	…
170°	(dx_7,dy_7), (dx_8,dy_8)

D.H. Ballard: Generalizing the Hough Transform to Detect Arbitrary Shapes. Pattern Recognition, Vol. 13, No. 2, pp. 111-122, 1981

Generalisierte Hough Transformation

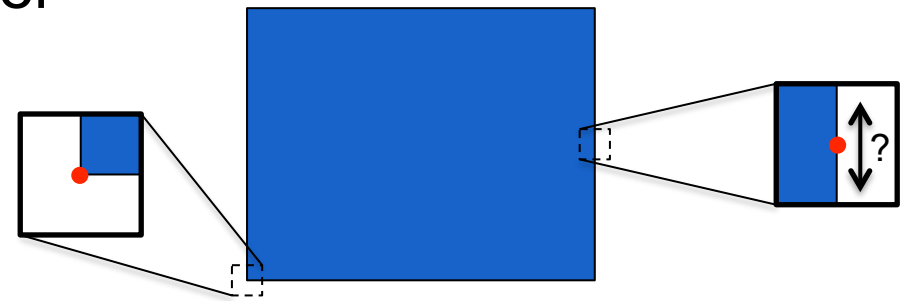




ECKEN

Erkennung von Ecken

- Viele Bilderkennungsalgorithmen benötigen Merkmale, die eine stabile Position in (x,y) haben
- Kanten sind nur in einer Richtung lokalisiert
  → Ecken in zwei



- Gewünschte Eigenschaften von Merkmalen (in verschiedenen Bildern vom gleichem Objekt / der gleichen Szene)
  - Genaue Lokalisierbarkeit
  - Invarianz gegenüber Rotation, Skalierung, Helligkeitsänderung
  - Robust gegenüber Rauschen

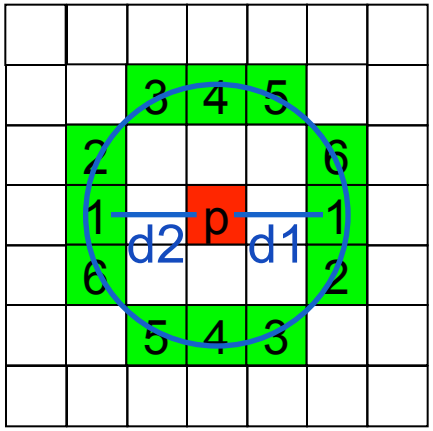
Slide and illustration adapted from Bernd Girod, Digital Image Processing

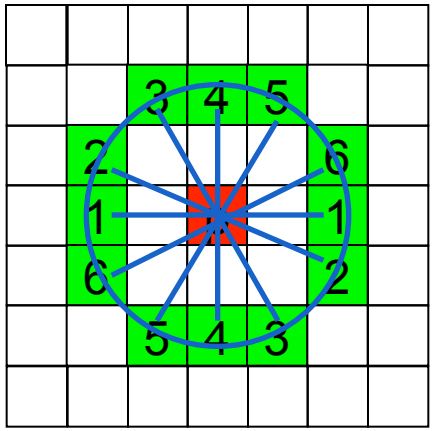
Simple Corner Detector

- Corner response of pixel p = (x,y)

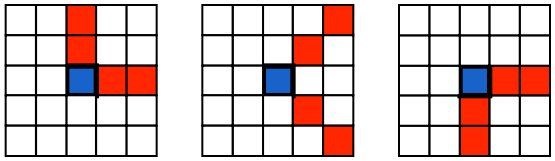
```
// I(p) = intensity (grayscale value) of pixel p
dmin = ∞
for all opposite points (p1,p2) on circle
    d1 = abs(I(p) - I(p1))
    d2 = abs(I(p) - I(p2))
    d = max(d1, d2)
    dmin = min(dmin, d)
cornerResponse = dmin
```

- Non-maximum suppression
- Threshold to generate ~150 corners

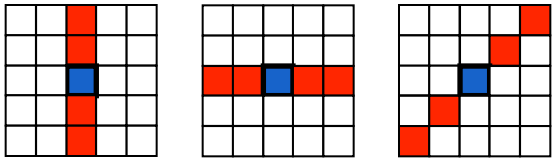




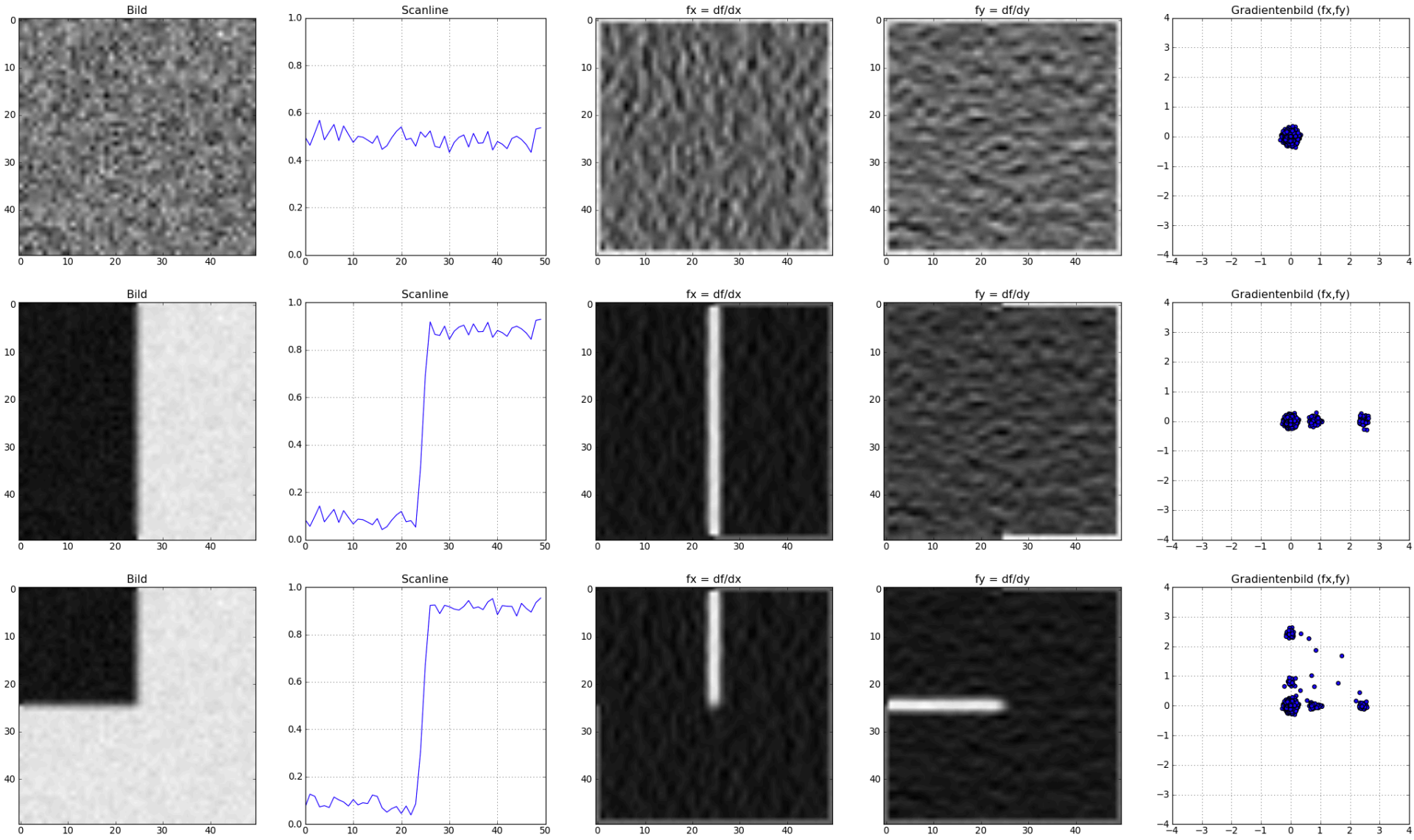
Ecken:



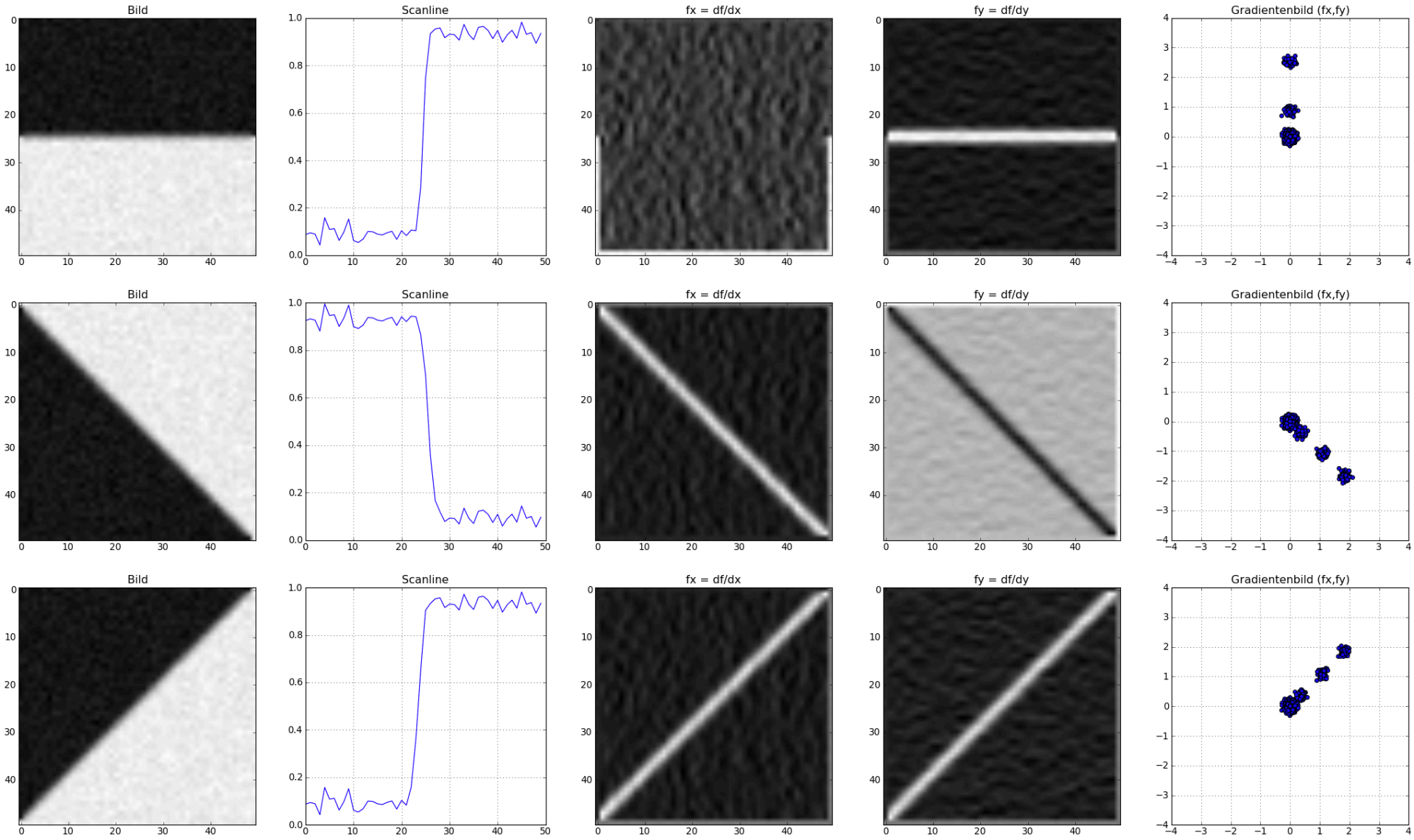
Keine Ecken:



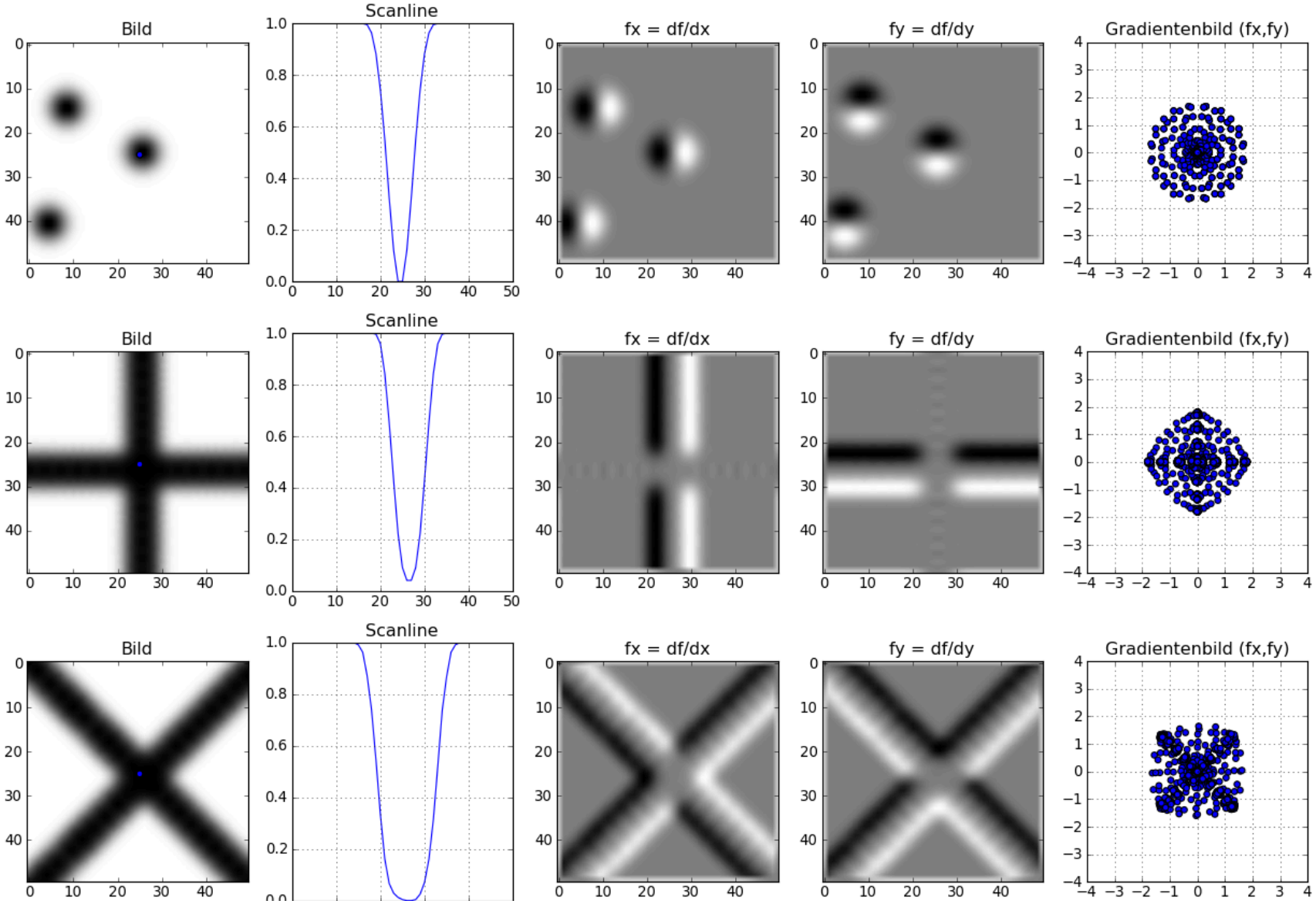
Verteilung der Gradienten



Verteilung der Gradienten

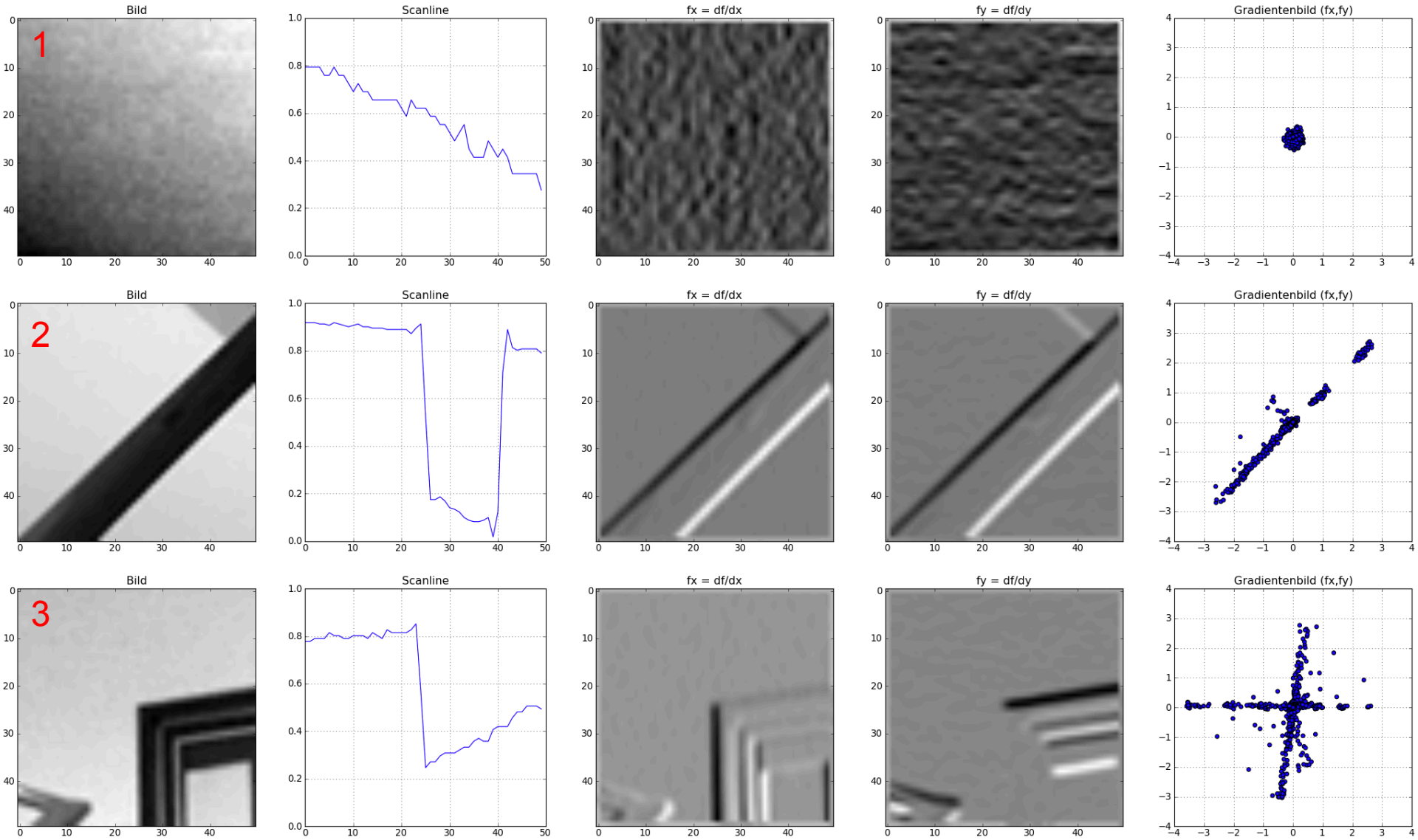


Verteilung der Gradienten



Verteilung der Gradienten in realem Bild

Verteilung der Gradienten in Bild



Verschiebung

- Effekt einer Verschiebung um (kleine) Δx, Δy
  - Flache Region: keine Änderung im Erscheinungsbild
  - Kante: keine Änderung bei Verschiebung entlang der Kante
  - Ecke: große Änderung in jeder Richtung
- Intensitätsänderung bei Verschiebung um Δx, Δy

$$s(\Delta x, \Delta y) = \sum_{(x,y) \in window} \left[f(x,y) - f(x+\Delta x, y+\Delta y) \right]^2$$

- Lineare Approximation für (kleine) Δx, Δy

$$f(x+\Delta x, y+\Delta y) \approx f(x,y) + f_x(x,y)\Delta x + f_y(x,y)\Delta y$$

  - f_x, f_y sind Gradienten in x- bzw. y-Richtung

Verschiebung

- Lineare Approximation für (kleine) Δx, Δy

$$f(x+\Delta x, y+\Delta y) \approx f(x,y) + f_x(x,y)\Delta x + f_y(x,y)\Delta y$$

$$s(\Delta x, \Delta y) \approx \sum_{(x,y)\in window} \left[f_x(x,y)\Delta x + f_y(x,y)\Delta y \right]^2$$

$$= \sum_{(x,y)\in window} \left[\begin{pmatrix} f_x(x,y) & f_y(x,y) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right]^2$$

$$= \sum_{(x,y)\in window} \left[\begin{pmatrix} f_x(x,y) & f_y(x,y) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right] \left[\begin{pmatrix} f_x(x,y) & f_y(x,y) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right]$$

$$= \sum_{(x,y)\in window} \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \begin{pmatrix} f_x(x,y) & f_y(x,y) \end{pmatrix}^T \begin{pmatrix} f_x(x,y) & f_y(x,y) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

Verschiebung

$$= \sum_{(x,y)\in window} \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \begin{pmatrix} f_x(x,y) & f_y(x,y) \end{pmatrix}^T \begin{pmatrix} f_x(x,y) & f_y(x,y) \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

$$= \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \left[\sum_{(x,y)\in window} \begin{pmatrix} f_x(x,y) & f_y(x,y) \end{pmatrix}^T \begin{pmatrix} f_x(x,y) & f_y(x,y) \end{pmatrix} \right] \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

$$= \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \left[\sum_{(x,y)\in window} \begin{pmatrix} f_x^2 & f_x f_y \\ f_x f_y & f_y^2 \end{pmatrix} \right] \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

$$= \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \begin{pmatrix} \sum_{(x,y)\in window} f_x^2 & \sum_{(x,y)\in window} f_x f_y \\ \sum_{(x,y)\in window} f_x f_y & \sum_{(x,y)\in window} f_y^2 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} M \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

Verschiebung

- Lineare Approximation für (kleine) Δx, Δy

$$s(\Delta x, \Delta y)$$

$$\approx \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} \begin{pmatrix} \sum_{(x,y)\in window} f_x^2 & \sum_{(x,y)\in window} f_x f_y \\ \sum_{(x,y)\in window} f_x f_y & \sum_{(x,y)\in window} f_y^2 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

$$= \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} M \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

- M: Kovarianzmatrix, „autocorrelation matrix“

Animation: http://www.aiaccess.net/English/Glossaries/GlosMod/e_gm_covariance_matrix.htm#Animation_covariance%20matrix

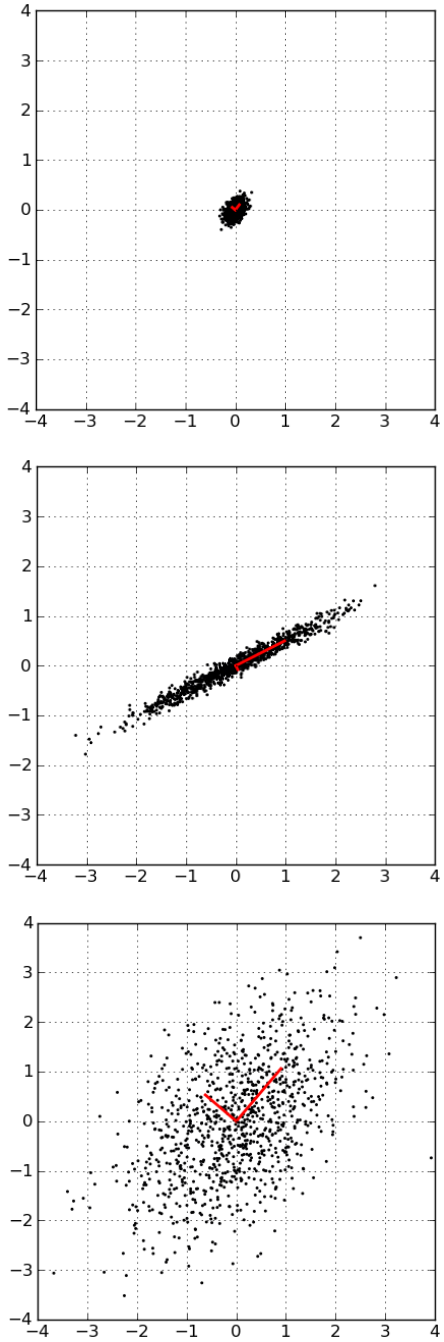
Eigenwerte der Kovarianzmatrix

- Eigenvektoren v_1, v_2; Eigenwerte λ_1, λ_2
  - Richtung: Hauptachsen der Verteilung
  - Länge: Varianz entlang der Hauptachsen
- Keine dominante Orientierung
  - $\lambda_1 = 0.017$
  - $\lambda_2 = 0.006$

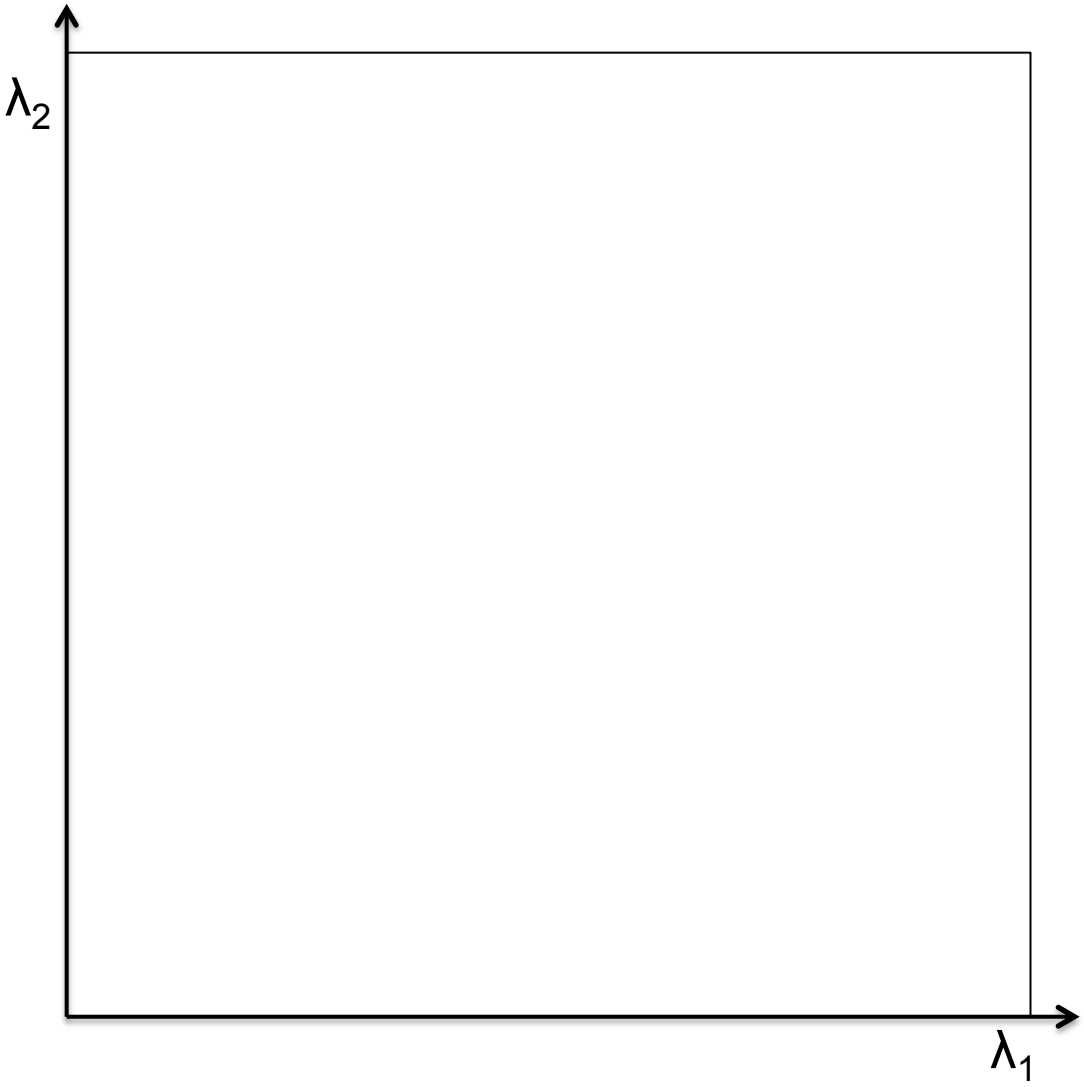
  → λ_1 klein, λ_2 klein
- Nur eine dominante Orientierung
  - $\lambda_1 = 1.313$
  - $\lambda_2 = 0.008$

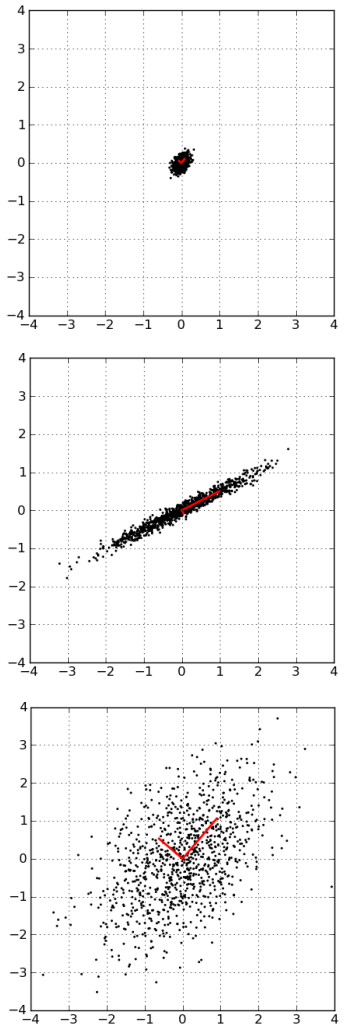
  → λ_1 groß, λ_2 klein
- Mehrere dominante Orientierungen
  - $\lambda_1 = 1.936$
  - $\lambda_2 = 0.669$

  → λ_1 groß, λ_2 groß

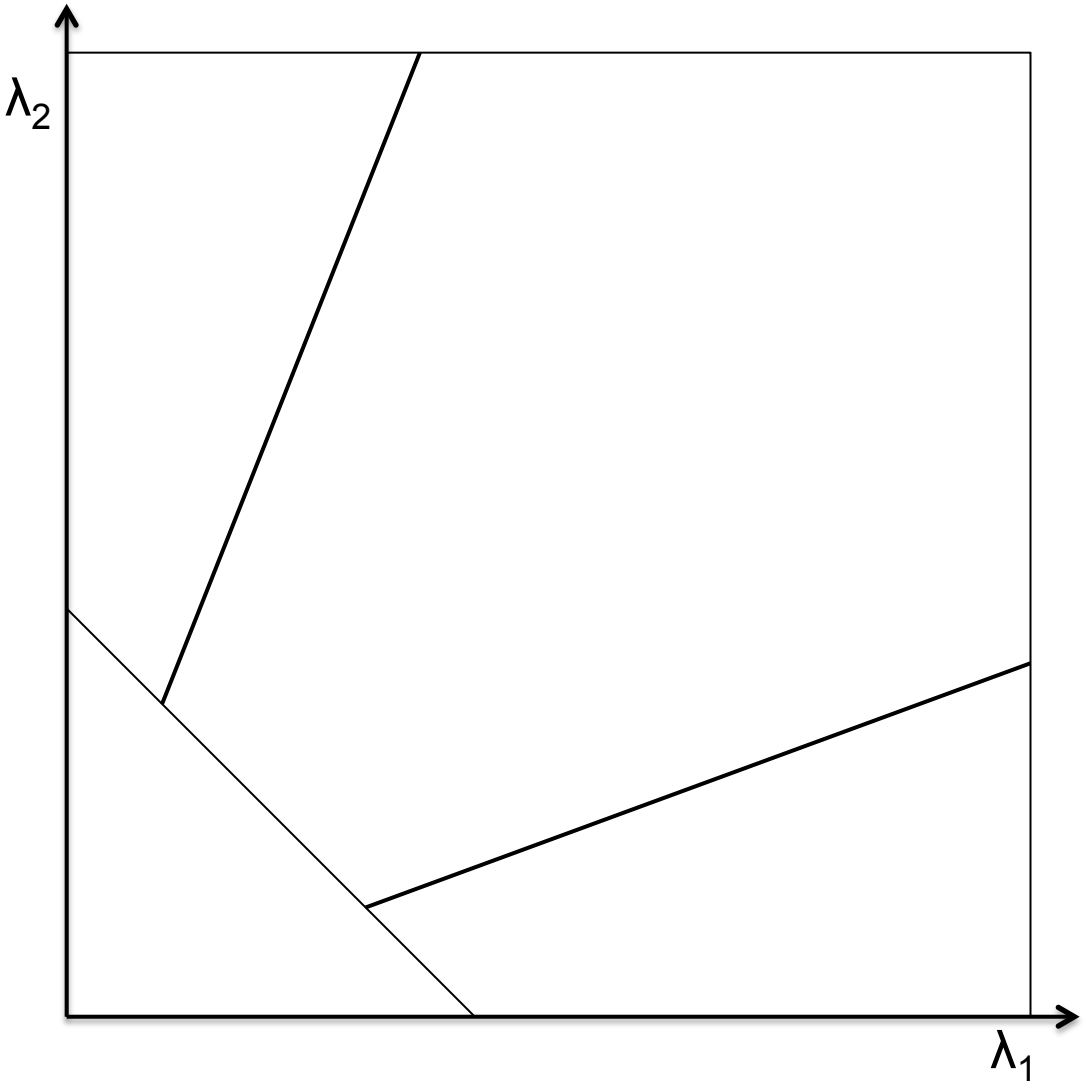


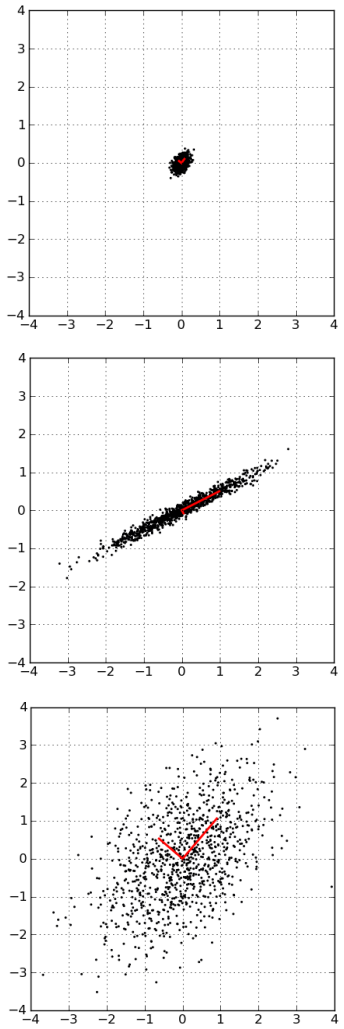
Klassifikation über Eigenvektoren



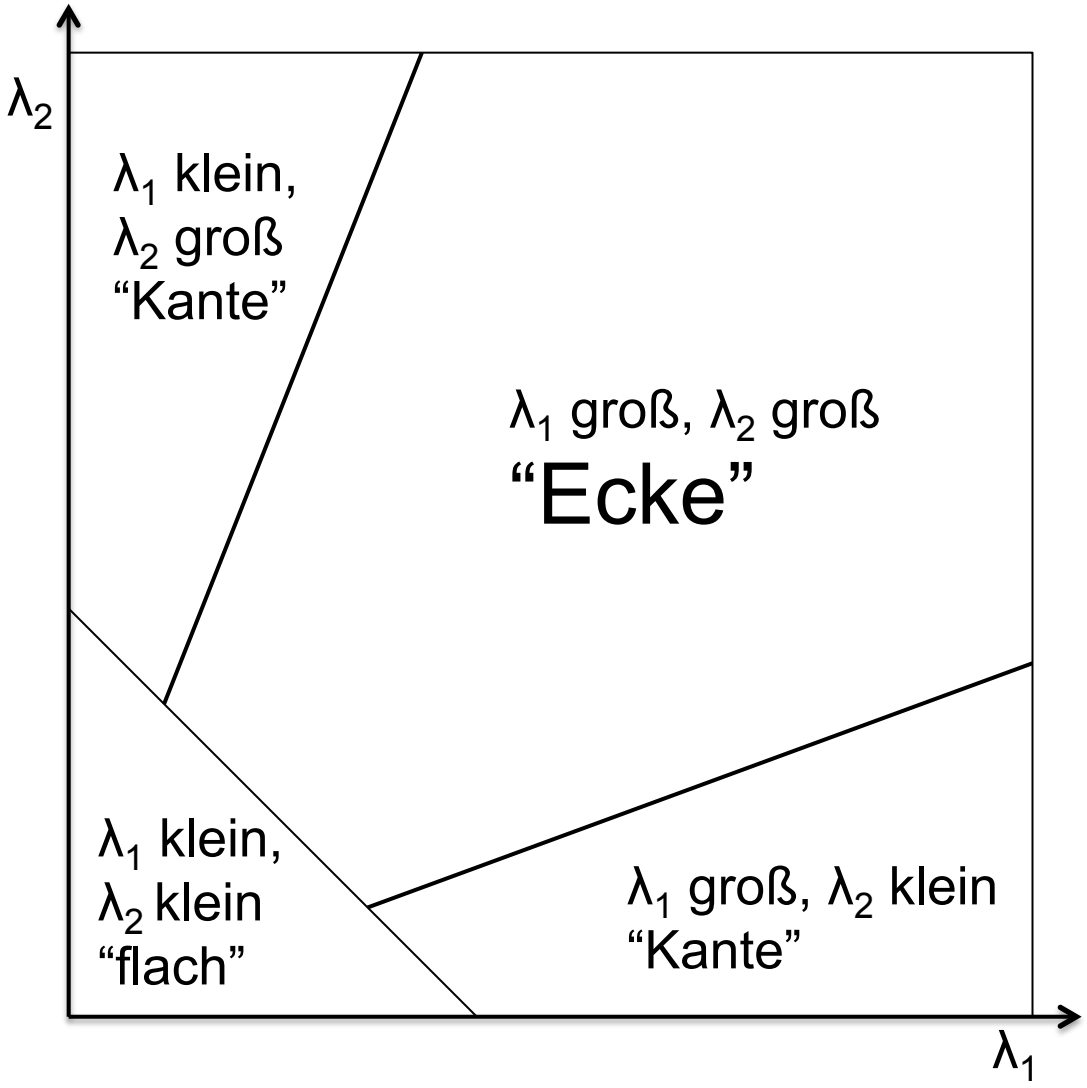


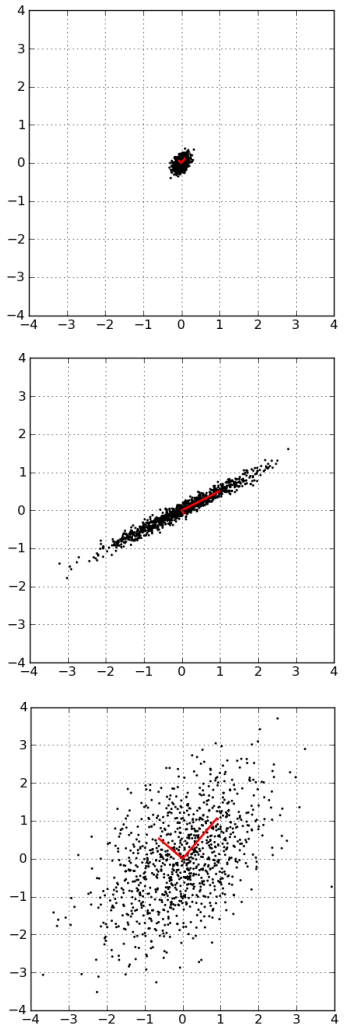
Klassifikation über Eigenvektoren





Klassifikation über Eigenvektoren





Harris Corner Detektor (Harris, Stephens, 1988)

- Kovarianzmatrix, „autocorrelation matrix“, „second moment matrix“

$$M = \sum_{(x,y)\in window} w(x,y) * \begin{pmatrix} f_x^2(x,y) & f_x(x,y)f_y(x,y) \\ f_x(x,y)f_y(x,y) & f_y^2(x,y) \end{pmatrix}$$

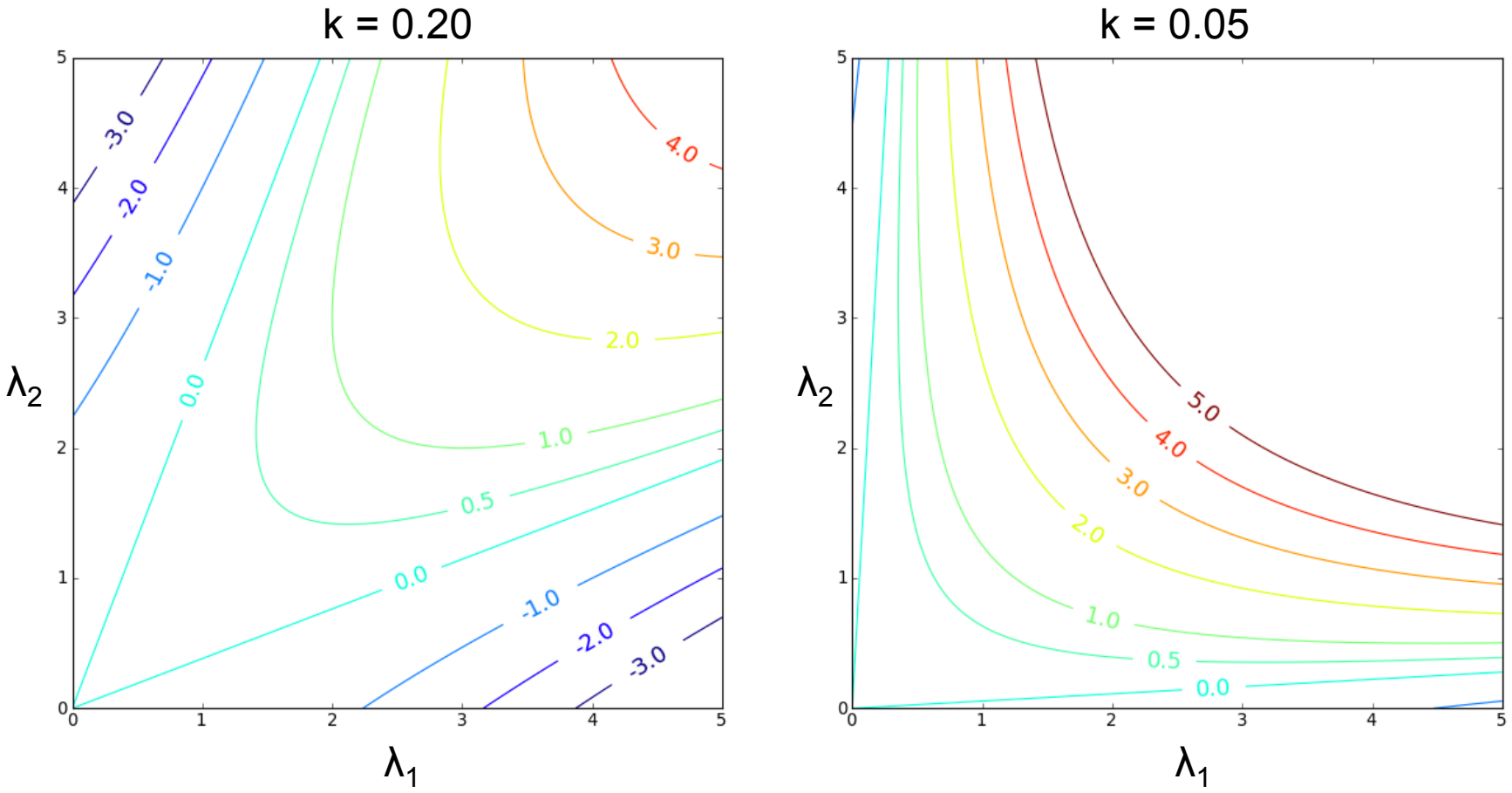
w(x,y): Gewichtungsfunktion, z.B. Gauß-Funktion

- Maß für die “Stärke” der Ecke

$$C(x,y) = \det(M) - k(trace(M))^2 = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$$

k = 0.04..0.06

Harris Corner Detektor: Parameter “k”



- $C = \det(M) - k\ \mathrm{trace}(M)^2 = \lambda_1 \lambda_2 - k\ (\lambda_1 + \lambda_2)$

Harris Corner Detection – Algorithmus

- Berechne die Ableitungen f_x und f_y in x- und y-Richtung
- Berechne für jedes Pixel die Produkte der Ableitungen

  $f_x^2 = f_x f_x, \quad f_y^2 = f_y f_y, \quad f_{xy} = f_x f_y$
- Berechne für jedes Pixel (x,y) die gewichtete Summe im Fenster um (x,y)
- Definiere für jedes Pixel (x,y) die Matrix M(x,y)
- Berechne für jedes Pixel das Maß der Eckenstärke
- Wende eine untere Schranke und non-maximum Unterdrückung an

Harris Corner Detection – Algorithmus

- Berechne die Ableitungen f_x und f_y in x- und y-Richtung
  - z.B. per Konvolution mit dem Sobel-Operator
- Berechne die elementweisen Produkte der Ableitungen

  $f_x^2 = f_x f_x, \quad f_y^2 = f_y f_y, \quad f_{xy} = f_x f_y$
- Berechne $f_{xxSum} = w * f_x^2, \quad f_{yySum} = w * f_y^2, \quad f_{xySum} = w * f_{xy}$, wobei * der Konvolutionsoperator ist und (z.B.)

$$w = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

- Definiere für jedes Pixel (x,y) die Matrix M(x,y)

$$M(x,y) = \begin{pmatrix} f_{xxSum}(x,y) & f_{xySum}(x,y) \\ f_{xySum}(x,y) & f_{yySum}(x,y) \end{pmatrix}$$

Harris Corner Detection – Algorithmus

- Berechne für jedes Pixel das Maß der Eckenstärke

$$C(x,y) = \det(M) - k(trace(M))^2 = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$$

k = 0.04..0.06

- Wende eine untere Schranke und non-maximum Unterdrückung an
  - z.B. im Radius 2 um den jeweils betrachteten Punkt

Beispiele

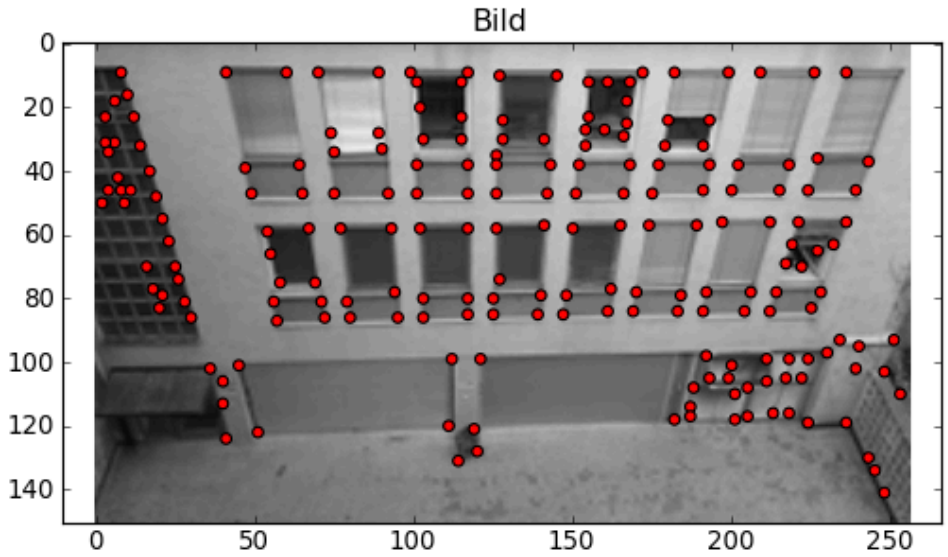


Bild
0
20
40
60
80
100
120
140
0
50
100
150
200
250

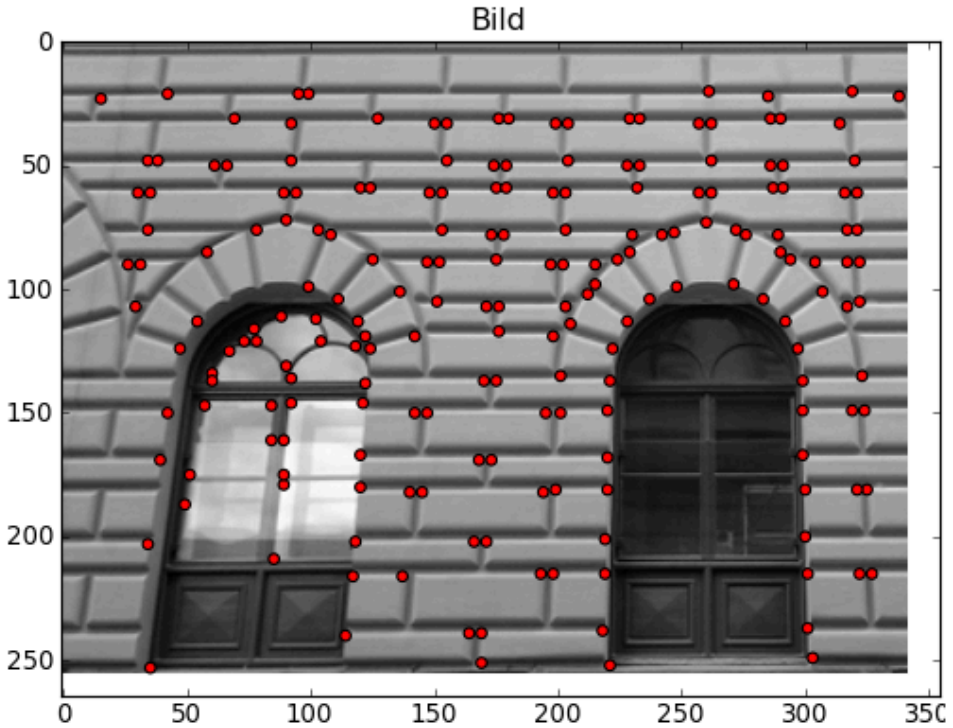
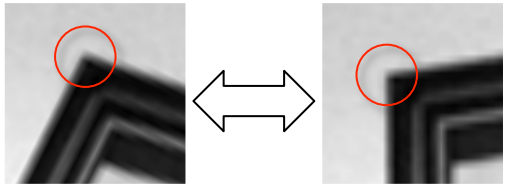


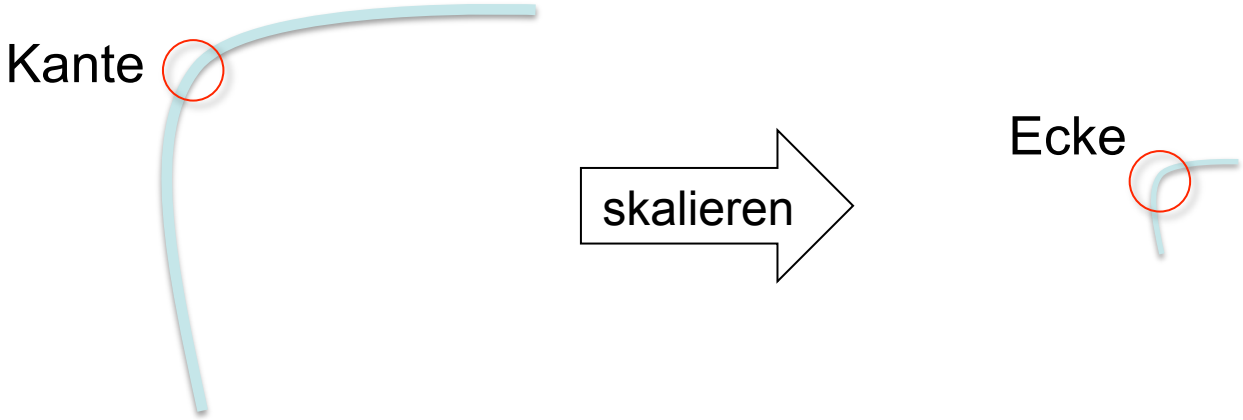
Bild
0
50
100
150
200
250
0
50
100
150
200
250
300
350

Robustheit des Harris Corner Detektors

- Invariant gegenüber Helligkeitsänderungen
- Invariant gegenüber Translation und Rotation



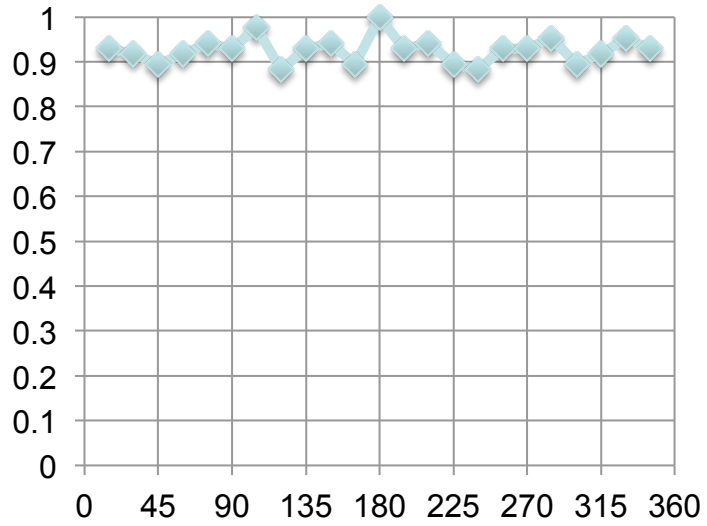
- Nicht invariant gegenüber Skalierung

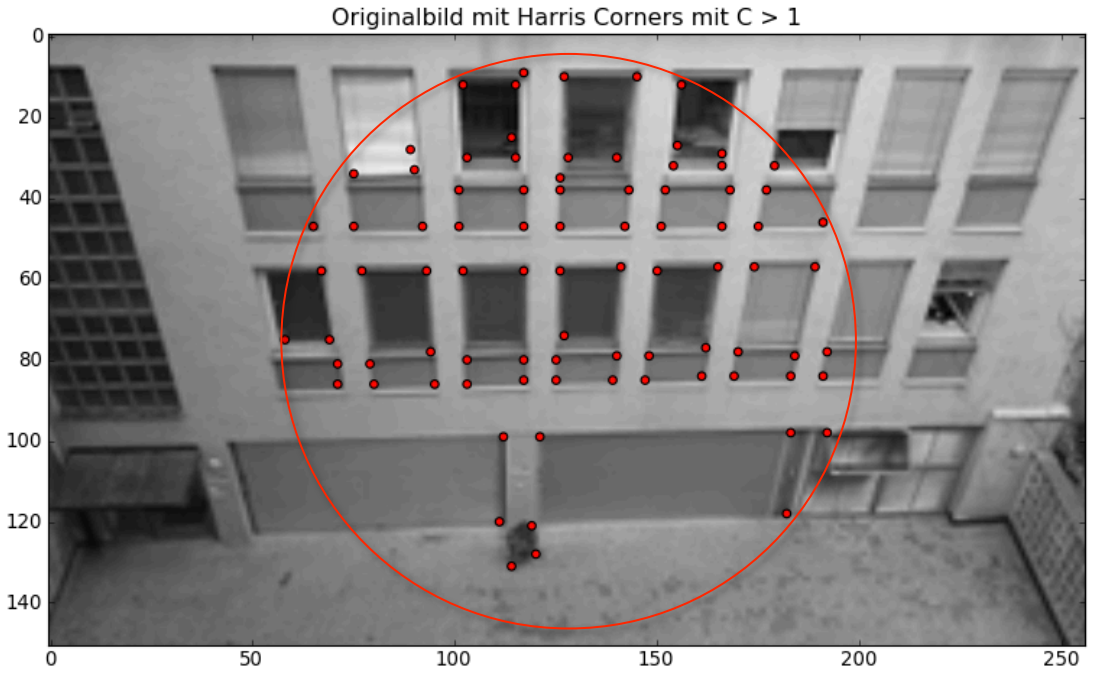


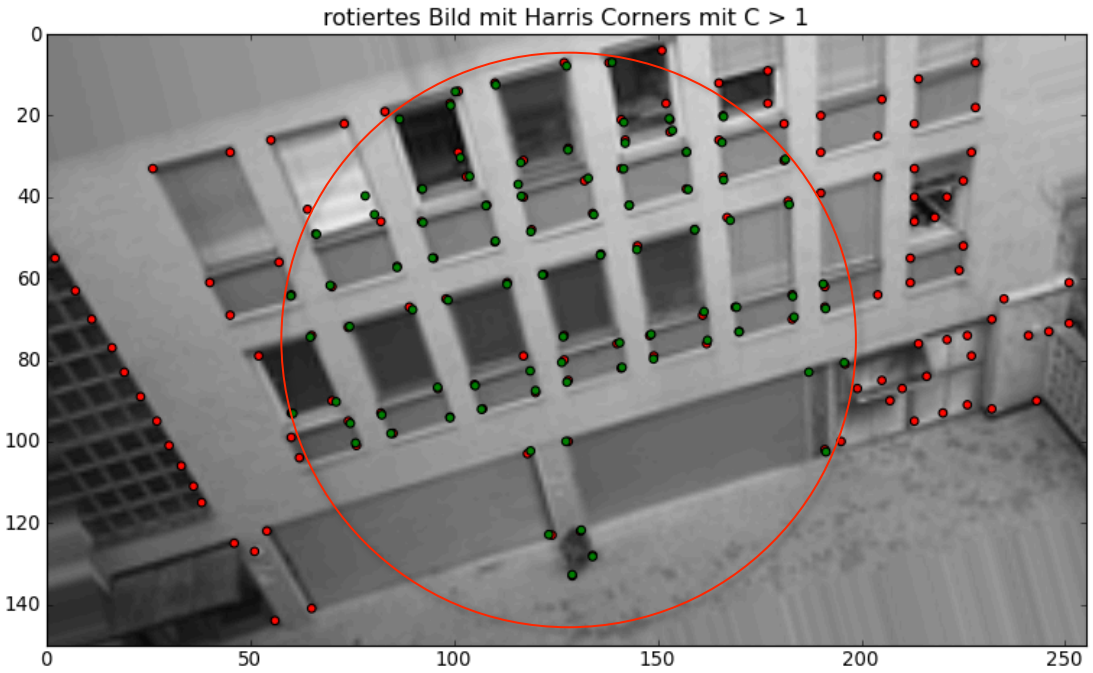
Slide and illustration adapted from Bern Girod, Digital Image Processing

Repeatability

- Robustheit gegenüber Rotation, Skalierung, Änderung der Perspektive
- hier Rotation:







Repeatability

- Robustheit gegenüber Rotation, Skalierung, Änderung der Perspektive
- hier Skalierung: