Kantenextraktion

Klassische Verfahren

Christoph Wagner

30. Januar 2006

Vortrag zum Seminar
„Bildsegmentierung und Computer Vision“

Gliederung

Gliederung

Zielstellung

Vorher

Nachher

Was sind Kanten?

- Stellen mit abrupter Helligkeitsänderung
- Optisch prägnante Bildmerkmale
- Ermöglichen, Umrisse von Menschen, Objekten, etc. wahrzunehmen
- spielen wichtige Rolle beim menschlichen Sehen
- gehören zu den wichtigsten Bildinformationen (Extremfall: selbst Strichzeichnung genügen, um Personen, Objekte usw. zu erkennen)

Eigenschaften von Kanten

- Lokal starke Intensitätsänderung entlang ausgeprägter Richtung
- d.h. es gibt starken An– oder Abstieg auf relativ kleinem Raum
- Daher naheliegend, Ableitung der Bildfunktion zu betrachen
- Kanten lassen sich dann als lokale Maxima der Ableitung charakterisieren

Gliederung

Anforderungen an Kantenfilter

Verschiebungsfreiheit:

- Kantendetektion invariant unter Verschiebungen

Isotropie (Invarianz unter Bilddrehungen):

- (ermittelte) Kantenintensität soll nicht von der Kantenrichtung abhängen
- mit anderen Worten: Bilddrehung ändert Kantenintensitäten nicht
- Schwachpunkt fast aller klassischen Kantendetektionsverfahren

Gliederung

Darstellung von Bildern, Notationen

Wie üblich:

- Betrachten nur Graustufenbilder
- Bei Farbbildern Verwendung nur eines Kanals (z.B. Helligkeit)
- Bild ist dann eine Funktion $f : \mathbb{R}^2 \rightarrow \mathbb{R}$
- Annahme: eigentliches Bild ist stetige Funktion
- Kennen jedoch nur deren Werte an auf einem diskreten Gitter
- Manchmal auch besser, das Bild als Matrix $\mathbf{I}$ aufzufassen

Lineare Filter

- Die klassischen Verfahren lassen sich mithilfe von sog. *Linearen Filtern* implementieren
- einfaches, vielseitiges Hilfsmittel zur Bildfilterung
- Prinzip: lineare Verknüpfung von Pixeln in kleinen Nachbarschaften
- Jedes Pixel sowie alle Pixel in einer gewissen Nachbarschaft werden mit einem Koeffizienten gewichtet und dann aufsummiert
- dies ergibt dann den neuen Wert des Pixels an dieser Stelle

Faltung

- Lineare Filter lassen sich als Matrizen darstellen
- Anwendung mittels *Faltung* mit der Bildmatrix

Definition

Sei $\mathbf{I}(x, y)$ die Bildmatrix und $\mathbf{H}(x, y)$ die Faltungsmatrix an der Stelle (x, y). Dann ist die **diskrete Faltung** $\mathbf{I} * \mathbf{H}$ definiert als

$$(\mathbf{I} * \mathbf{H})(x, y) := \sum_{i=-n}^{n} \sum_{j=-n}^{n} \mathbf{I}(x + i, y + j)\mathbf{H}(n + i, n + j)$$

wobei $\mathbf{H} \in \mathbb{R}^{(2n+1)\times(2n+1)}$

Beispiel

$$\begin{array}{c|c|c} 1 & 5 & 1 \\ \hline 1 & 5 & 1 \\ \hline 1 & 5 & 1 \end{array} \ast \begin{array}{c|c|c} 0 & 0 & 0 \\ \hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \end{array} = \begin{array}{c|c|c} 6 & 7 & 6 \\ \hline 6 & 7 & 6 \\ \hline 6 & 7 & 6 \end{array}$$

- Hier wurde außerhalb der Matrix der Wert 0 angenommen
- Es gibt jedoch auch andere Möglichkeiten:
    - Erweitern
    - Umfalten

Beispiel (Gauß'scher Glättungsfilter, 3×3)

$$\mathbf{G} = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- Diskretisierung einer 2D-Normalverteilung auf 3×3-Matrix
- Parameter: σ und Filterradius (meistens $\approx 3\sigma$)
- Normalerweise mit Normalisierung (hier weggelassen)
- Verringert Bildrauschen
- Weniger Konturverwaschung als z.B. mit Mittelwertfilter

Gliederung

Herleitung (1)

- Betrachten den Gradienten der Bildfunktion, d.h.

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{pmatrix}$$

- Betrag des Gradienten $|\nabla f| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$ ist Maß für die Stärke des Anstiegs
- Richtung des Anstiegs: $\Phi = \arctan\left(\frac{\partial f/\partial y}{\partial f/\partial x}\right)$
- Approximation der Ableitung durch diskrete Differenzen:

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

- analog für die zweite Koordinate

Herleitung (2)

Es ergeben sich folgende Faltungskerne (ohne Nullzeilen bzw. -spalten):

- horizontal: $\mathbf{H}_x = \begin{bmatrix} -0.5 & 0 & 0.5 \end{bmatrix} = 0.5 \cdot \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$
- vertikal: $\mathbf{H}_y = \begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix} = 0.5 \cdot \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$
- Schätzung des Gradienten in horizontaler und vertikaler Richtung: $\mathbf{D}_x = \mathbf{I} * \mathbf{H}_x$, $\mathbf{D}_y = \mathbf{I} * \mathbf{H}_y$

Beispiel (Gradientenbilder)

$* \mathbf{H}_x =$

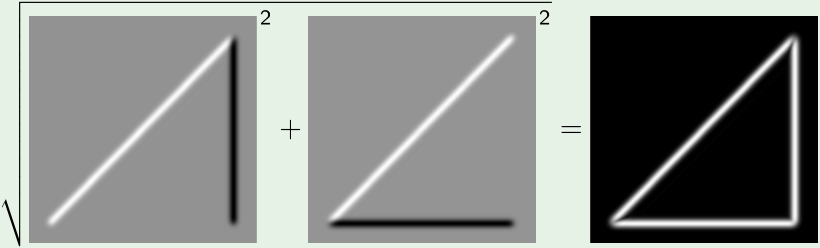
$* \mathbf{H}_y =$

Herleitung (3)

Errechnen nun Kantenbild aus $\mathbf{D}_x$ und $\mathbf{D}_y$ folgendermaßen:

- schätzen den Betrag des Gradienten in jedem einzelnen Bildpunkt
- berechnen also $\mathbf{D}(i,j) := \sqrt{\mathbf{D}_x^2(i,j) + \mathbf{D}_y^2(i,j)}$
- Schreiben dafür einfacher: $\mathbf{D} := \sqrt{\mathbf{D}_x^2 + \mathbf{D}_y^2}$
- Wurzelberechnung jedoch recht aufwendig
- Daher häufig Annäherung durch $\tilde{\mathbf{D}} := |\mathbf{D}_x| + |\mathbf{D}_y|$

Beispiel (Kantenbild)



Prewitt-Operator

- Motivation: Verringern der Rauschanfälligkeit
- Dazu wird orthogonal zur Ableitungsrichtung geglättet
- Es ergeben sich folgende Filtermasken:
  - $\mathbf{H}_x^P = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$
  - $\mathbf{H}_y^P = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

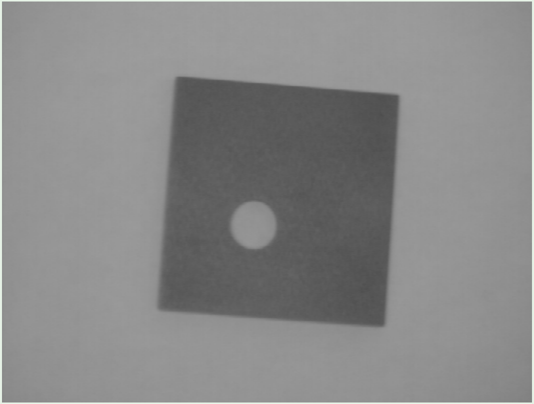
Sobel-Operator

- Stärkere Gewichtung der zentralen Matrixspalte bzw. –zeile
- Die Filtermatrizen sehen dann wie folgt aus:
  - $\mathbf{H}_x^S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$
  - $\mathbf{H}_y^S = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
- Liefert in den meisten Fällen sehr brauchbare Ergebnisse
- Ist praktisch in allen Bildbearbeitungsprogrammen zu finden

Optimierter Sobel-Operator

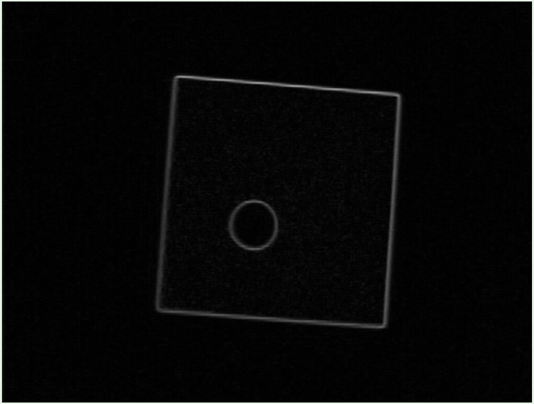
- Durch geschickte Wahl der Gewichte kann die Isotropie verbessert werden
- Man kann zeigen, dass folgende Variante des Sobel-Filters optimal ist:
  - $\tilde{\mathbf{H}}_x^S = \begin{bmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{bmatrix}$
  - $\tilde{\mathbf{H}}_y^S = \begin{bmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{bmatrix}$

Beispiel (Prewitt- und Sobel-Operator)



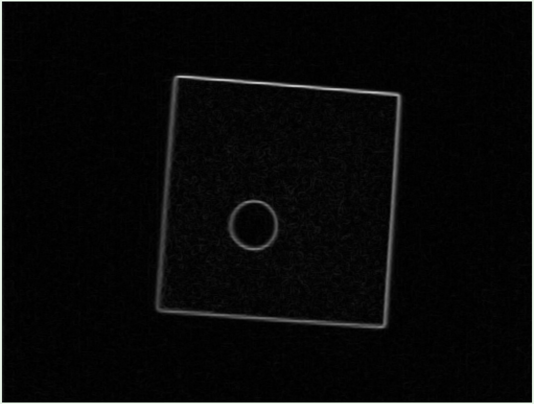
Ausgangsbild

Beispiel (Prewitt- und Sobel-Operator)



Mit Prewitt-Operator

Beispiel (Prewitt- und Sobel-Operator)



Mit Sobel-Operator

Kompassfilter
Motivation

- Problem bei den bisher betrachteten Filtern: Anisotropie
- Stärkeres Ansprechen bei horizontale bzw. vertikale Kanten
- Möglicher Ausweg: zusätzliche Filter zur Erkennung diagonal verlaufender Kanten → „Kompassfilter“
- Ein Beispiel ist der sog. *Kirsch-Operator*
- dieser verwendet acht Filtermatrizen, für Kanten in 45°-Schritten

Kirsch-Operator
Faltungsmasken

$$\mathbf{H}_0^K = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \mathbf{H}_4^K = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$\mathbf{H}_1^K = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix} \qquad \mathbf{H}_5^K = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix}$$

$$\mathbf{H}_2^K = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad \mathbf{H}_6^K = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\mathbf{H}_3^K = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \qquad \mathbf{H}_7^K = \begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix}$$

Durchführung

- Offenbar müssen nur vier der acht Filter berechnet werden
- die anderen unterscheiden sich lediglich durch das Vorzeichen
- es werden also folgende Teilbilder berechnet:

$$\begin{array}{llll} \mathbf{D}_0 = \mathbf{I} * \mathbf{H}_0 & \mathbf{D}_1 = \mathbf{I} * \mathbf{H}_1 & \mathbf{D}_2 = \mathbf{I} * \mathbf{H}_2 & \mathbf{D}_3 = \mathbf{I} * \mathbf{H}_3 \\ \mathbf{D}_4 = -\mathbf{D}_0 & \mathbf{D}_5 = -\mathbf{D}_1 & \mathbf{D}_6 = -\mathbf{D}_2 & \mathbf{D}_7 = -\mathbf{D}_3 \end{array}$$

- das Filterergebnis ist definiert als das Maximum der Einzelfilter: $\mathbf{D}^K := \max_{i=0\ldots7} \mathbf{D}_i = \max_{i=0\ldots3} |\mathbf{D}_i|$
- Praktischer Vorteil dieses Filters jedoch recht gering
- Kommt aber ohne Wurzelberechnung aus

Gliederung

Verfahren zweiter Ordnung

- Bisher: Kanten als Extrema der ersten Ableitung
- Jetzt: Nullstellen der zweiten Ableitung
- An einer Extremstelle der ersten Ableitung hat die zweite einen Nulldurchgang
- Betrachten den *Laplace-Operator*

$$\nabla^2 f := \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Laplace-Operator (1)

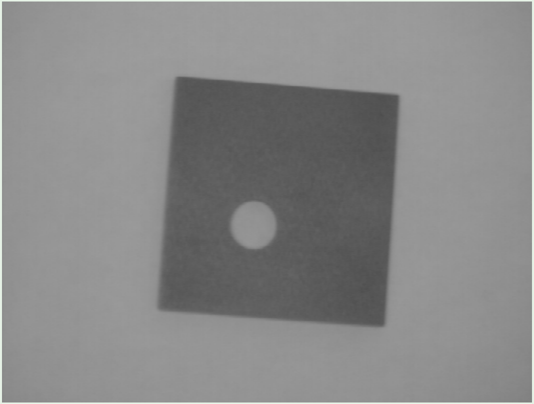
- Verwenden folgende Diskretisierung der zweiten Ableitung

$$
\begin{aligned}
\frac{\partial^2 f(x,y)}{\partial x^2} &\approx \frac{\partial(f(x+1,y)-f(x,y))}{\partial x} \\
&\approx f(x+1,y)-f(x,y)-(f(x,y)-f(x-1,y)) \\
&= f(x+1,y)-2f(x,y)+f(x-1,y)
\end{aligned}
$$

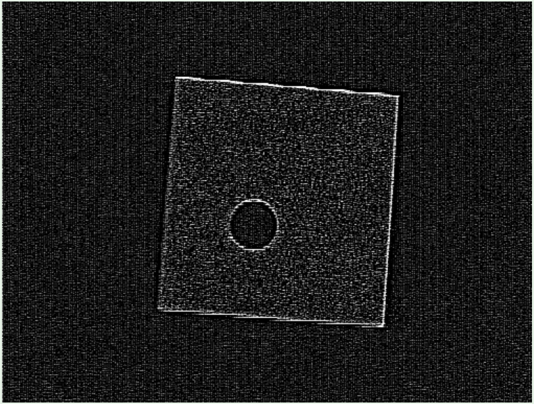
- Analog wird die Ableitung nach y gebildet
- Dann ergibt sich als Faltungskern

$$
\mathbf{H}^L = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}
$$

Beispiel (Laplace-Operator)



Beispiel (Laplace-Operator)



Laplace-Operator (2)

- Eine gebräuchliche Variante ist folgender Faltungskern

$$\tilde{\mathbf{H}}^L = \begin{bmatrix} 1 & 2 & 1 \\ 2 & -12 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \left(= 2\mathbf{H}^L + \begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix} \right)$$

- Nach Anwendung der Maske kann noch nach Nulldurchgängen gesucht werden

Algorithmus (Suchen nach Nulldurchgängen)

Für jedes Pixel:

- Betrachte paarweise gegenüberliegende Nachbarpixel
- Besitzt ein Paar unterschiedliche Vorzeichen, setze Pixel auf 1
- Ansonsten setze Pixel auf 0

Laplace-Operator (2)

- Eine gebräuchliche Variante ist folgender Faltungskern

$$\tilde{\mathbf{H}}^L = \begin{bmatrix} 1 & 2 & 1 \\ 2 & -12 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \left(= 2\mathbf{H}^L + \begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix} \right)$$

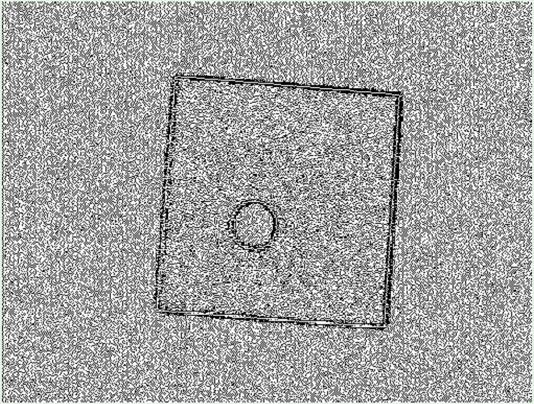
- Nach Anwendung der Maske kann noch nach Nulldurchgängen gesucht werden

Algorithmus (Suchen nach Nulldurchgängen)

Für jedes Pixel:

- Betrachte paarweise gegenüberliegende Nachbarpixel
- Besitzt ein Paar unterschiedliche Vorzeichen, setze Pixel auf 1
- Ansonsten setze Pixel auf 0

Beispiel (Laplace mit Suche nach Nulldurchängen)



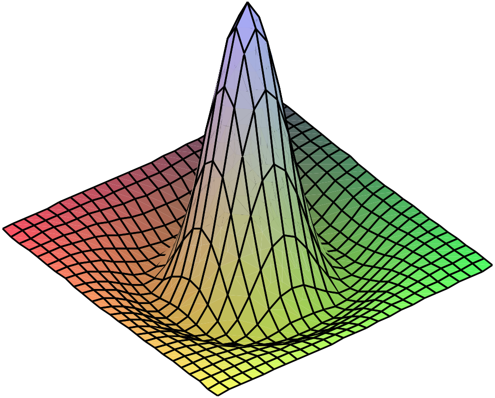
Laplacian of Gaussian (1)

- Wegen der Rauschanfälligkeit liegt es nahe, das Bild zu glätten
- Dazu kann wieder der Gauß-Filter verwendet werden
- Da die Faltung assoziativ ist, kann man beides kombinieren
- Dieser Filter wird *Laplacian of Gaussian* (LoG) genannt
- Die Faltungsmatrix ergibt sich wie folgt
- Sei $\phi(x,y) = \frac{1}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$
- Dann gilt

$$\nabla^2 \phi(x,y) = \frac{1}{\sigma^4} \left(\frac{x^2+y^2}{\sigma^2} - 2 \right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Laplacian of Gaussian (2)

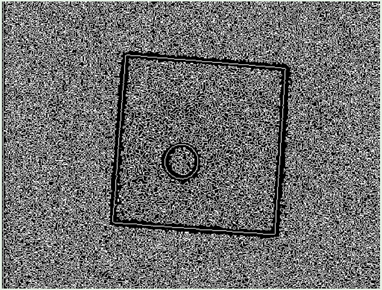
Die Funktion $-\nabla^2\phi(x, y)$ wird wegen ihrer Form auch „Mexican Hat“ genannt:



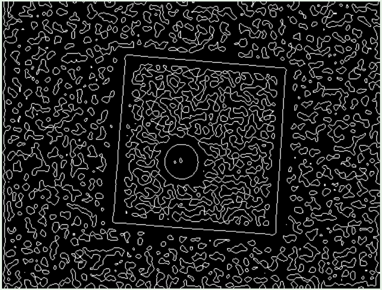
Laplacian of Gaussian (3)

- Radius der Maske muss in Abhängigkeit von σ gewählt werden
- Auch hier wird wieder meistens $r = 3\sigma$ gewähl
- Die enstehenden Filtermatrizen werden jedoch recht groß! (z.B. $\sigma = 1 \Rightarrow 7 \times 7$-Filtermatrix)
- Man kann zeigen, dass die detektierten Kanten immer geschlossen sind, wenn sie nicht am Bildrand enden
- Der Filter wird auch als *Marr-Hildreth-Operator* bezeichnet

Beispiel (Laplacian of Gaussian)



$\sigma = 1$



$\sigma = 3$

Gliederung

Canny-Algorithmus

- Ziel der Entwicklung: Optimalität in Bezug auf drei Kriterien
  - **Erkennung:** alle tatsächlichen Kanten sollen gefunden werden, aber keine falschen
  - **Lokalisierung:** Abstand zwischen tatsächlicher und erkannter Kante möglichst klein
  - **Ansprechverhalten:** keine Mehrfacherkennung ein- und derselben Kante
- Verwendet anders als bisher auch die Kantenrichtungen
- Ergebnis: Binärbild ($1 =$ Kante / $0 =$ keine Kante)
- Wird allgemein als bestes der klassischen Verfahren erachtet

Canny-Algorithmus
Übersicht

Der Algorithmus besteht aus mehreren Phasen:

1. **Glättung:** um Rauschen zu unterdrücken, wird mit einem Gauß-Filter geglättet
2. **Kantendetektion:** berechne Kantenstärken und –richtungen
3. **Unterdrückung von Nicht-Maxima:** Nur lokale Maxima der Kantenstärke werden als potenzielle Kantenpixel zugelassen
4. **Hysterese:** Unterdrückung nichtrelevanter Kanten mittels eines Zwei-Schwellenwerteverfahrens

Beispiel (Ausgangsbild)

Canny-Algorithmus

Glättung und Kantendetektion

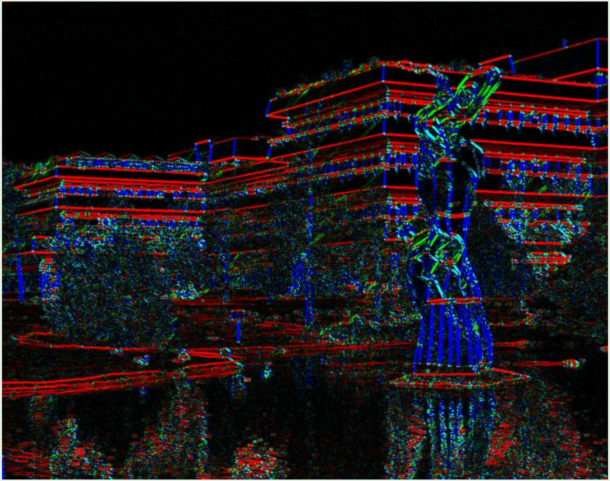
Zu Beginn werden folgende Schritte ausgeführt:

- Glättung mit einem Gauß-Filter ($\rightarrow$ Parameter: σ)
- Berechnung des Sobel-Operators in x– und y–Richtung
- Daraus Berechnung des Kantenbildes $\mathbf{D} = \sqrt{\mathbf{D}_x^2 + \mathbf{D}_y^2}$
- Berechnung der Gradientenrichtung für jedes Pixel: $\Phi(x, y) = \arctan\left(\frac{\mathbf{D}_y(x,y)}{\mathbf{D}_x(x,y)}\right)$
- Runden auf $0, 45, 90, 135$ Grad, d.h. waagerecht, senkrecht oder diagonal

Beispiel (Kantenbild)

Kantenbild ($\sigma = 1$)

Beispiel (Kantenbild)



Gradientenrichtungen (nach Kantenstärke)

Canny-Algorithmus

Unterdrückung von Nicht-Maxima

- Unterdrückung aller Pixel, die keine lokalen Maxima in Gradientenrichtung sind
- Dadurch werden Kanten ausgedünnt (auf ein Pixel Breite)
- Wichtig für Lokalisierung der Kanten
- Kante wird dort vermutet, wo Filterantwort am stärksten

Algorithmus

Führe für alle Pixel folgende Schritte durch

- Betrachte beide Nachbarpixel in Gradientenrichtung
- Besitzt eines einen größeren Pixelwert: lösche aktuelles Pixel
- Falls nicht (lok. Maximum): behalte Pixelwert bei

Canny-Algorithmus

Unterdrückung von Nicht-Maxima

- Unterdrückung aller Pixel, die keine lokalen Maxima in Gradientenrichtung sind
- Dadurch werden Kanten ausgedünnt (auf ein Pixel Breite)
- Wichtig für Lokalisierung der Kanten
- Kante wird dort vermutet, wo Filterantwort am stärksten

Algorithmus

Führe für alle Pixel folgende Schritte durch

- Betrachte beide Nachbarpixel in Gradientenrichtung
- Besitzt eines einen größeren Pixelwert: lösche aktuelles Pixel
- Falls nicht (lok. Maximum): behalte Pixelwert bei

Beispiel (Nach Unterdrückung der Nicht-Maxima)

Canny-Algorithmus

Hysterese

- Verwendet zwei Schwellenwerte, T_1 und T_2, mit $T_1 \leq T_2$
- Unterdrückt schwache bzw. „falsche“ Kanten
- „Zerreißt“ weniger Kanten als einfache Schwellenwertverfahren

Algorithmus

- Markiere alle Pixel mit Werten größer T_2 als Kantenpixel
- Setze alle Pixel mit Werten kleiner T_1 auf 0
- Beginnend bei jedem Kantenpixel:
  - Verfolge alle angrenzenden Kanten, solange Wert $>= T_1$
  - Markiere alle dazugehörigen Pixel als Kanten
- Setze alle noch nicht als Kante markierten Pixel auf 0

Canny-Algorithmus

Hysterese

- Verwendet zwei Schwellenwerte, T_1 und T_2, mit $T_1 \leq T_2$
- Unterdrückt schwache bzw. „falsche“ Kanten
- „Zerreißt“ weniger Kanten als einfache Schwellenwertverfahren

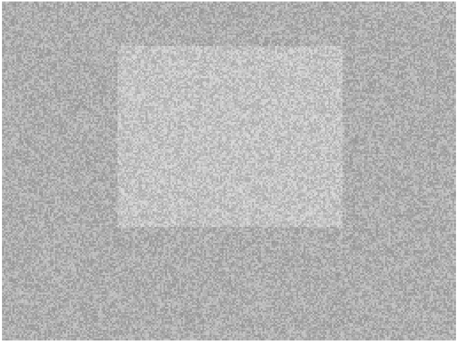
Algorithmus

- Markiere alle Pixel mit Werten größer T_2 als Kantenpixel
- Setze alle Pixel mit Werten kleiner T_1 auf 0
- Beginnend bei jedem Kantenpixel:
  - Verfolge alle angrenzenden Kanten, solange Wert $>= T_1$
  - Markiere alle dazugehörigen Pixel als Kanten
- Setze alle noch nicht als Kante markierten Pixel auf 0

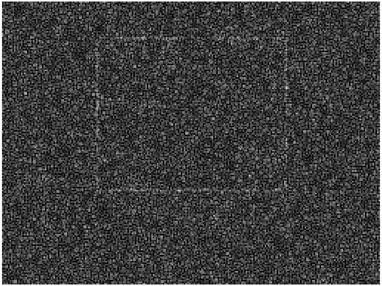
Beispiel (Endgültiges Bild)

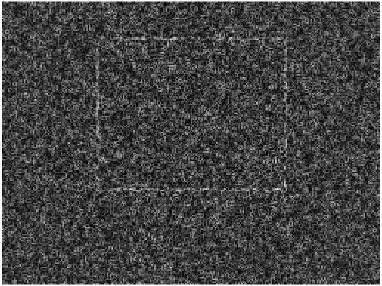
Gliederung

Vergleich der Verfahren
Ausgangsbild

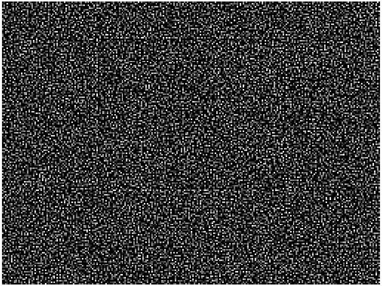


Vergleich der Verfahren
Prewitt und Sobel

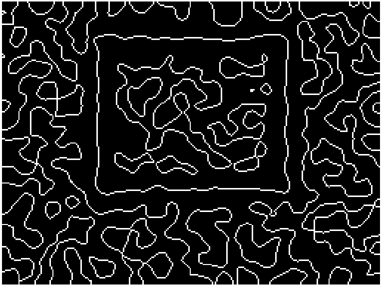

Prewitt-Operator


Sobel-Operator

Vergleich der Verfahren
Laplace, LoG

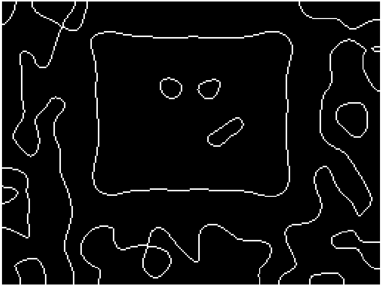


Laplace

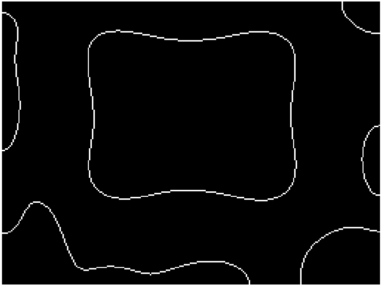


LoG ($\sigma = 5$)

Vergleich der Verfahren
Laplace, LoG

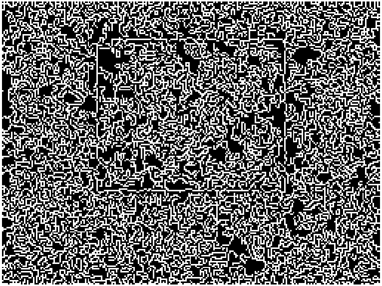


LoG ($\sigma = 10$)

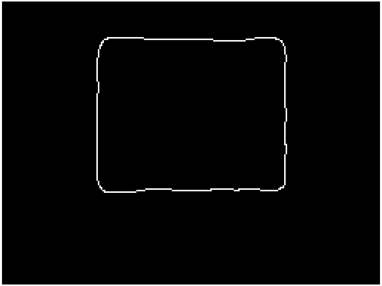


LoG ($\sigma = 20$)

Vergleich der Verfahren
Canny-Algorithmus



$\sigma = 1,\ T_1 = 0.05,\ T_2 = 0.5$



$\sigma = 5,\ T_1 = 0.5,\ T_2 = 0.8$

Gliederung

Zusammenfassung

Prewitt-, Sobel- und Kirschoperator

- Einfach zu implementieren
- Brauchbare Ergebnisse
- Detektieren jedoch häufig falsche Kanten
- Bevorzugen deutlich horizontale bzw. vertikale Kanten

Zusammenfassung

Laplace, LoG

- Weniger richtungsabhängig
- Dafür rauschempfindlicher
- LoG bei großem σ aufwendig zu berechnen

Zusammenfassung

Canny-Filter

- „Optimal“ unter den klassischen Verfahren
- Basis vieler Verbesserungen
- Starke Abhängigkeit von den Parametern
- Daher kein „unbeaufsichtigter“ Einsatz möglich
- Texturierte Obeflächen erzeugen u.U. „falsche“ Kanten
- Dafür gibt es jedoch bereits Lösungsansätze („Surround Suppression“, s. Literaturhinweise)

Ausblick

- Information über Kanten kann zur Objekterkennung benutzt werden
- Dazu kann man versuchen, die Kanten nachzuverfolgen
- Zusammenhängende Kanten können aber nicht immer als solche detektiert werden
- Andere Möglichkeit: sog. *Hough-Transformation*
- Weiteres Problem: keine einfache Verallgemeinerung auf Mehrkanalbilder möglich
- Verschiedene Kanäle können widersprüchliche Kanteninformationen liefern

Literatur

B. Jähne
Digitale Bildverarbeitung, Kapitel 11
Springer, 2002

W. Burger, M. J. Burge
Digitale Bildverarbeitung, Kapitel 7
Springer, 2005

J. Canny
A Computational Approach To Edge Detection
IEEE Transactions on Pattern Analysis and Machine Intelligence, 8:679-714 (1986)

URLs

- Canny-Filter online (eigene Bilder hochladbar)
  `http://matlabserver.cs.rug.nl/cannyedgedetectionweb/web/index.html`
- Bilder z.T. von `http://homepages.inf.ed.ac.uk/rbf/HIPR2/`
- ImageJ Homepage: `http://rsb.info.nih.gov/ij/`
- FeatureJ (enthält Canny- und LoG-Filter)
  `http://www.imagescience.org/meijering/software/featurej/`

Zur Vertiefung

- C. Grigorescu, N. Petkov, M. A. Westenberg, *Contour and boundary detection improved by surround suppression of texture edges*, Image and Vision Computing 22 (2004), 609–622.
  `http://www.cs.rug.nl/~petkov/publications/2004ivc_contour.pdf`

Vielen Dank für die Aufmerksamkeit!