FRIEDRICH-SCHILLER-
UNIVERSITÄT
JENA
Fakultät für Mathematik
und Informatik

Institut für Informatik
Lehrstuhl für Digitale Bildverarbeitung
Prof. Dr.-Ing. Joachim Denzler
http://www.inf-cv.uni-jena.de

M.Sc. Niklas Penzel

Wintersemester

Übung zur Vorlesung

Rechnersehen 1

Übungsblatt 2: Histogramme, Filterung im Ortsbereich

Aufgabe 1 Histogramme (2 Punkte)

Das Histogramm h eines Bildes stellt die Verteilung der Intensitäten im Bild dar. Schreiben Sie eine Python-Funktion, die ein Grauwertbild lädt, dessen Histogramm ermittelt und dieses graphisch darstellt!

Aufgabe 2 Histogrammlinerarisierung (2 Punkte)

Damit der zur Verfügung stehende Grauwertbereich optimal ausgenutzt wird, kann das Histogramm eines Bildes linearisiert werden. Dadurch wird der Kontrast verstärkt und das Bild qualitativ besser. Bei der Linearisierung wird die Quantisierungskennlinie optimal an die in einem Bild auftretenden Helligkeitswerte angepasst, d.h. Bereiche mit seltenen Grauwerten werden im Histogramm enger „zusammengerückt“, Bereiche mit häufigen Grauwerten werden gestreckt (Abbildung 1).

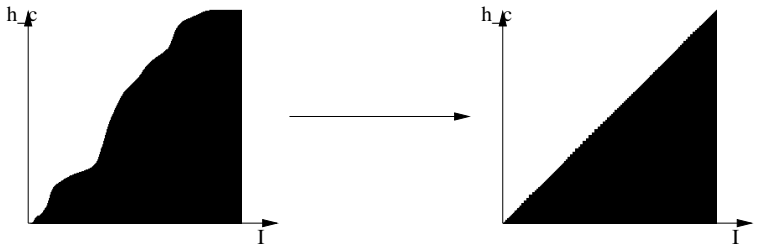

Abbildung 1: Histogrammlinearisierung: Darstellung der kumulierten Histogramme

Um das Histogramm eines Bildes zu linearisieren, wird zunächst das kumulierte Histogramm

$$h_c(I) = \sum_{i=0}^{I} h(i). \tag{1}$$

berechnet, das zu jedem Grauwert I die Häufigkeit von Intensitäten unterhalb dieses Grauwertes angibt. Jedem Pixel im Bild mit dem Grauwert I wird dann ein neuer Grauwert $I' = h_c(I)$ zugewiesen, wobei eine Skalierung der Werte von h_c auf den Wertebereich der Grauwerte vorgenommen wird.

Schreiben Sie eine Python-Funktion, die die Histogrammlinearisierung auf Grauwertbildern durchführt! Testen Sie diese auf den im Ordner `Bilder` bereitgestellten Beispielbildern!

Aufgabe 3 Faltung (2 Punkte)

Ein wichtiger Basisalgorithmus in der Bildverarbeitung ist die diskrete Faltung

$$\boldsymbol{I}_A(i,j) = (\boldsymbol{I} * \boldsymbol{A})(i,j) = \sum_{h=-\lfloor \frac{m}{2} \rfloor}^{\lfloor \frac{m}{2} \rfloor} \sum_{k=-\lfloor \frac{n}{2} \rfloor}^{\lfloor \frac{n}{2} \rfloor} \boldsymbol{A}(h,k) \cdot \boldsymbol{I}(i-h, j-k). \tag{2}$$

Hierbei wird eine Faltungsmaske $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ verwendet und über das Eingabebild $\boldsymbol{I}$ „geschoben“. Dabei wird für jedes Pixel des Eingabebildes eine mit den korrespondierenden Einträgen der Faltungsmaske gewichtete Linearkombination der Nachbarschaftspixel berechnet. Verschiedene Faltungsmasken haben dabei unterschiedliche Auswirkungen auf das Bild.

Eine einfache Faltungsmaske ist z. B. der Mittelwertfilter $\boldsymbol{A}_{\text{avg}}$, der für $m = n = 3$ die folgende Form hat:

$$\boldsymbol{A}_{\text{avg}} = \frac{1}{9} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}. \tag{3}$$

Schreiben Sie eine Python-Funktion, die eine beliebige Filtermaske $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ auf ein Bild anwendet und testen Sie diese mit obigem Mittelwertfilter $\boldsymbol{A}_{\text{avg}}$! Überlegen Sie sich ein Vorgehen bei der Behandlung der an den Bildrändern gelegenen Pixeln!

Aufgabe 4 Gauß-Filter (2 Punkte)

Der vorgestellte Mittelwertfilter bewirkt eine Glättung des Bildes, die zur Rauschunterdrückung verwendet werden kann. Ein anderer Faltungsoperator mit demselben Zweck ist der Gauß-Filter $\boldsymbol{A}_{\text{Gauß}} \in \mathbb{R}^{m \times m}$. Dabei sind die Koeffizienten der Faltungsmaske gegeben durch

$$\boldsymbol{A}_{\text{Gauß}}(h,k) = \mathrm{e}^{-\frac{h^2+k^2}{2\sigma^2}}. \tag{4}$$

Die Standardabweichung σ wird in Abhängigkeit von der Fenstergröße üblicherweise auf $\sigma = \frac{m}{5}$ gesetzt.

Berechnen Sie für verschiedene Fenstergrößen (z. B. $m \in \{3, 5, 7\}$) die Koeffizienten des Gaußfilters! Achten Sie dabei darauf, dass die Summe der Koeffizienten auf 1 normiert wird, damit bei der Faltung das Bild insgesamt nicht heller oder dunkler wird! Verwenden Sie die errechneten Gaußfilter als Ersatz für den Mittelwertfilter aus der vorherigen Teilaufgabe und vergleichen Sie die Ergebnisse!

Aufgabe 5 Separierbarkeit der Gauß-Filterung (2 Punkte)

Als effizientere Variante kann eine zweidimensionale lineare Filterung auf zwei eindimensionale lineare Filterungen reduziert werden.

Ein linearer zweidimensionaler Filter $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ heißt separierbar, wenn er durch Faltung zweier eindimensionaler Filter dargestellt werden kann:

$$\boldsymbol{A} = \boldsymbol{D}_1 * \boldsymbol{D}_2, \qquad \text{mit } \boldsymbol{D}_1 \in \mathbb{R}^m, \boldsymbol{D}_2 \in \mathbb{R}^n \tag{5}$$

$$= \boldsymbol{D}_1 \cdot \boldsymbol{D}_2^\top, \qquad (\text{da } \boldsymbol{D}_1, \boldsymbol{D}_2 \text{ Vektoren}). \tag{6}$$

Somit ergibt sich die Faltung zu

$$\boldsymbol{I} * \boldsymbol{A} = \boldsymbol{I} * (\boldsymbol{D_1} * \boldsymbol{D_2}) \tag{7}$$

$$= (\boldsymbol{I} * \boldsymbol{D_1}) * \boldsymbol{D_2} \qquad (\text{Assoziativität der Faltung}) \tag{8}$$

Implementieren Sie nun die 2-D-Gaußfilterung als Hintereinanderausführung je eines Gaußfilters in vertikaler und horizontaler Richtung! Vergleichen Sie die Laufzeiten mit der nicht-separierten Filterung für verschiedene Größen der Filtermaske!

Viel Spaß und Erfolg!