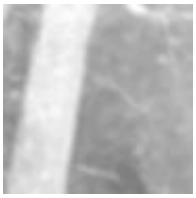
Praktikum 9

Non-maximum Suppression

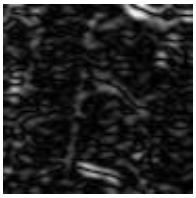
M.Thaler, 8/2014, ZHAW

1 Einführung

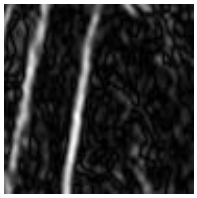
Bei der Bestimmung der Kanten in einem Bild, wird oft in einem ersten Schritt der Betrag des Gradienten bestimmt und in einem zweiten Schritt mit Hilfe eines Thresholds ein Binärbild erzeugt. Hier der Ausschnitt aus einem Bild:



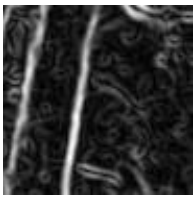
a) Original



b) Gx



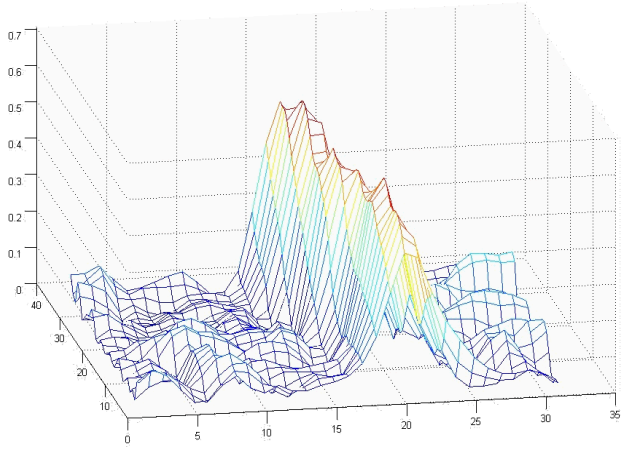
c) Gy



d) Betrag von G

e) Threshold von G

Nachteilig an diesem Verfahren ist, dass die Kante mehrere Pixel breit ist und damit auch die genaue Lage der Kante nicht bekannt ist. Eine detailliertere Analyse zeigt, dass der Betrag des Gradienten G, d.h. die Kante einen *Gebirgskamm* darstellt, wie folgendes 3-d Bild zeigt (Ausschnitt des Gradientenbildes von oben):



Gemäss Definition steht der Gradient rechtwinklig zu einer Kante im Bild und hat den grössten Betrag im Mittelpunkt der Kante. Werden nun entlang einer Kante nur die Pixel mit dem grössten Gradienten ausgewählt, erhält man den exakten, ein Pixel breiten Verlauf der Kante.

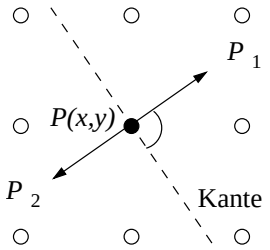
2 Non-maximum Suppression

Um den Verlauf einer Kante genauer bestimmen zu können, müssen diejenigen Pixel ausgewählt werden, die auf dem Kamm des Gradientenbildes liegen, resp. alle Pixel die nicht auf dem Kamm liegen müssen unterdrückt werden. Dies nennt man *non-maximum suppression*. Im folgenden wird ein einfaches Verfahren zum Finden der Gradientenkämme vorgestellt.

2.1 Vorgehen

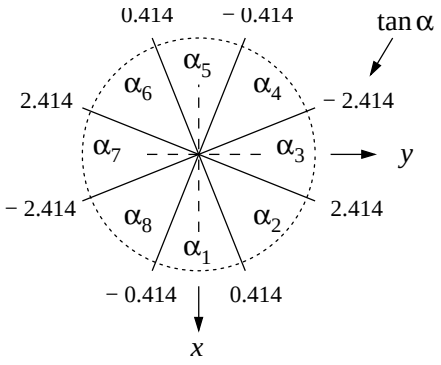
Gehen Sie für jedes Pixel eines Bildes wie folgt vor:

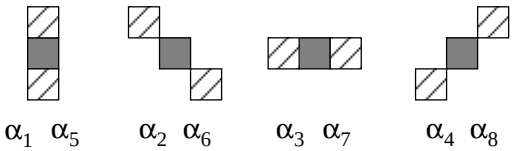
1. Suchen Sie zwei Pixel (P_1 und P_2) in der 8er-Nachbarschaft des aktuellen Pixels $P(x, y)$, so dass sie rechtwinklig zur Kantenrichtung liegen (d.h. in Richtung des Gradienten):



2. Bestimmen Sie in den zwei Pixeln P_1 und P_2 den Betrag des Gradienten.
3. Ist der Betrag des Gradienten in einem der Pixel P_1 oder P_2 grösser als im aktuellen Pixel $P(x, y)$, gehört das Pixel $P(x, y)$ nicht zur Kante und kann auf Null gesetzt werden, andernfalls wird das Pixel mit seinem Gradientenwert $G(x, y)$ übernommen.

Die Gradienten in den Pixeln P_1 und P_2 lassen sich auf verschiedenste Art und Weise bestimmen, z.B. durch lineare Interpolation, aber auch durch entsprechende Nachbarschaften. In diesem Fall reicht es aus festzustellen, in welchem Bereich der Winkel α des Gradienten liegt. Für eine 8er-Nachbarschaft müssen folgende Winkelbereiche in Betracht gezogen werden, wobei die schraffierten Nachbarn den Pixeln P_1 resp. P_2 entsprechen:





Der Tangens des Winkels α lässt sich aus dem Gradienten im Pixel $P(x, y)$ wie folgt bestimmen:

$$\tan\alpha = \frac{Gy(x,y)}{Gx(x,y)}$$

dabei ist zu beachten, dass x Null werden kann. Mit Hilfe des $\tan\alpha$ kann dann die entsprechende Nachbarschaft ausgewählt werden und bei den beiden Pixeln P_1 und P_2 der Gradient bestimmt werden.

2.2 Aufgabenstellung

Gegeben ist das Bild `arterie.jpg`. Bestimmen Sie für dieses Bild die ausgedünnten Kanten mit Hilfe der non-maximum Suppression. Gehen Sie dazu wie folgt vor:

1. Lesen Sie das Bild in Matlab ein, *casten* Sie es auf den Datentyp `double` und skalieren Sie es auf den Wertebereich $[0, 1]$.
2. Filtern Sie das Bild mit einem Gauss Tiefpass der Grösse 7x7 und $\sigma = 1.0$, damit wird das Rauschen reduziert.
3. Bestimmen Sie die Gradientenkomponenten G_x und G_y, sowie den Betrag des Gradienten G mit Hilfe von Sobel-Masken:
$$G = \sqrt{G_x^2 + G_y^2} \text{ resp. } G = |G_x| + |G_y|$$
4. Bestimmen Sie das Gradientenbild mit Hilfe der non-maximum Suppression. Vergleichen Sie das Gradientenbild mit und ohne non-maximum Suppression.

3 Nachbearbeitung

Die non-maximum Suppression liefert ein Bild mit Grauwerten proportional zur Stärke des Gradienten. Dieses Bild muss noch in ein schwarz-weiss Bild umgewandelt werden. Das kann z.B. mit Hilfe eines Thresholds und anschliessendem Verbinden von Nachbarschaftpixeln mit kleinerem *Grauwert* erreicht werden.

3.1 Aufgabenstellung

Verwenden Sie im folgenden das mit non-maximum Suppression vorverarbeitete Gradientenbild.

1. Skalieren Sie das Gradientenbild auf den Bereich $0, 1$ und erzeugen Sie ein Binärbild P_b, vorerst mit einem Threshold $th_1 = 0.3$.
2. Durchlaufen Sie nun das Binärbild P_b von oben links nach unten rechts und wählen Sie bei allen Pixeln $P_b(x, y) == 1$ diejenigen Pixel in der Umgebung $U_{b1}(x, y)$ als zusätzliche Kantenpixel, deren Grauwert im Gradientenbild grösser als der Threshold th_2 ist (setzen Sie vorerst $th_2 = 0$). Umgebungspixel $U_{b1}(x, y)$ von $P_b(x, y)$ sind hier Pixel an den Positionen $(x, y + 1)$, $(x + 1, y)$, $(x + 1, y + 1)$ und $(x + 1, y - 1)$.
3. Durchlaufen Sie das oben ergänzte Binärbild von unten rechts nach oben links und wählen Sie bei allen Pixeln $P_b(x, y) == 1$ diejenigen Pixel in der Umgebung $U_{b2}(x, y)$ als zusätzliche Kantenpixel, deren Grauwert im Gradientenbild grösser als der Threshold th_2 ist (setzen Sie vorerst $th_2 = 0$). Umgebungspixel $U_{b2}(x, y)$ von $P_b(x, y)$ sind hier Pixel an den Positionen $(x, y - 1)$, $(x - 1, y - 1)$, $(x - 1, y)$ und $(x - 1, y + 1)$.
4. Vergleichen Sie Ihr Resultat mit dem Resultat eines Canny-Edge Detektors, den Sie mit dem Matlab-Befehl `edge(f,'canny')` berechnen können. Variieren Sie bei Ihrem Verfahren die beiden Thresholds und diskutieren Sie die Resultate. Was müsste ev. noch verbessert werden?

Anmerkung: Sie haben hier einen etwas vereinfachten Canny-Edge Detektor implementiert. Wie beurteilen Sie Ihren Detektor?