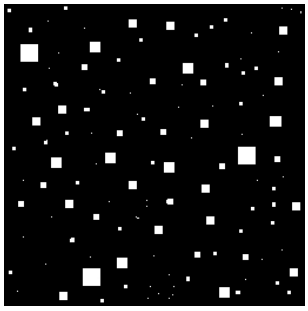
Praktikum 6

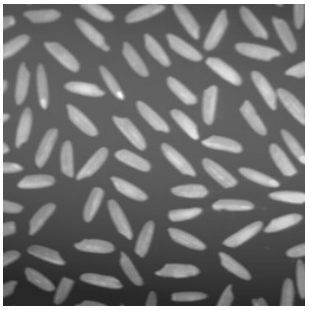
Morphologische Operationen und Component Labeling

M.Thaler, 8/2014, ZHAW

1 Einführung

Oft ist es notwendig verschiedene Objekte in einem Bild zu lokalisieren und für die einzelnen Objekte Parameter wie Grösse, Orientierung, Form, etc. zu bestimmen. Dazu müssen die Objekte in einem ersten Schritt extrahiert und markiert werden. Mit diesem Praktikum sollen die grundlegenden Schritte aufgezeigt werden, die für die Erkennung, Lokalisierung und Identifikation von Objekt in Bildern notwendig sind.
In einem ersten Beispiel werden Sie ausschliesslich mit morphologischen Operationen a priori bekannte Objekte (Quadrate der Grösse 5×5 aus einem Bild extrahieren und diese Komponenten zählen. In einem zweiten Beispiel werden Sie mit *component labeling* in einem realen Bild die Anzahl Reiskörner zählen, sowie einige einfache Objektmerkmale wie Gröss etc. berechnen und anschliessend farbig darstellen.





2 Morphologische Operationen

Das Bild `squares.tif` enthält verschieden grosse Objekte, unter anderem auch Quadrate der Grösse 5×5 Pixel.

2.1 Unerwünschte Objekte eliminieren

Eliminieren Sie im Bild `squares.tif` alle Objekte mit morphologischen Operationen, die nicht Quadrate der Grösse 5×5 sind. Verwenden Sie dazu die MATLAB-Funktionen `imerode()`, `imdilate()`, resp. `bwmoprh()`, sowie logische Operationen für Bilder (siehe on-line Help für weitere Informationen).

2.2 Quadrate lokalisieren

Im folgenden sollen die übrig gebliebenen 5×5 Quadrate lokalisiert, gezählt und nacheinander aus dem Bild entfernt werden. Das könnte wegen der bekannten Geometrie der Objekte sehr einfach implementiert werden, wir möchten hier aber einen anderen Weg beschreiten und einen Algorithmus implementieren, der auch mit Objekten beliebiger Form arbeitet. *Anmerkung*: Sie werden diesen Algorithmus im Zusammenhang mit dem Zählen der Reiskörner benötigen.

2.2.1 Algorithmus

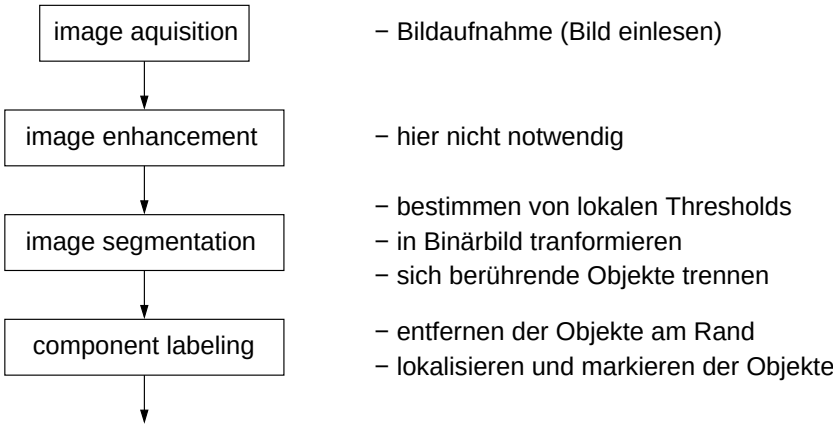
Verwenden Sie für das Extrahieren der Quadrate den in der Vorlesung vorgestellten Algorithmus zum Lokalisieren von *connected components*. Ein so genannter *Seed Point* wird dabei solange innerhalb seines Objektes mit dem Strukturelement B dilatiert, bis keine neuen Pixel mehr dazukommen:

$$X_k = (X_{k-1} \oplus B) \cap A \quad \text{mit } B = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Die und-Verknüpfung mit dem Bild A beschränkt die Dilatation auf das Objekt selbst. Für die Implementation steht das Rahmenprogramm in `morph.m` zur Verfügung, das die Funktion `extract()` aufruft (definiert als Rahmen in `extract.m`). Sie müssen also nur die Funktion `extract()` implementieren, die als Parameter das Binärbild, sowie einen Seed Point erwartet.

3 Component Labeling

Im folgenden sollen die Anzahl Reiskörner im Bild `rice.tif` gezählt werden. Mit diesem Beispiel soll auch der typische Ablauf eines Bildverarbeitungsalgorithmus aufgezeigt werden.
Folgende Schritte müssen dabei grundsätzlich durchlaufen werden:



Der Schritt *image enhancement* kann entfallen, weil die Kompensation der ungleichmässigen Bildausleuchtung zusammen mit der Segmentierung (Thresholding) durchgeführt werden kann (adaptiver Threshold).
Im folgenden werden die einzelnen Schritte anhand des Reiskorn-Beispiels erklärt und schrittweise implementiert.

3.1 Segmentierung

3.1.1 Umwandlung in Binärbild

Das Reiskornbild ist nicht gleichmässig ausgeleuchtet, d.h. es muss ein adaptives Threshold-Verfahren verwendet werden. Die Intensität ändert sich nur vertikal, aus diesem Grund kann das Bild in Streifen unterteilt werden und pro Bildstreifen ein Threshold bestimmt werden.

a) Unterteilen Sie das Bild in Streifen mit 16-Pixel Breite und bestimmen Sie pro Streifen den minimalen und maximalen Grauwert G_{min} resp. G_{max}.

b) Bestimmen Sie den Threshold $th = (G_{max} + G_{min})/2$ für jeden Bildstreifen und wandeln Sie das Grauwertbild in ein Binärbild um, indem Sie den jeweiligen lokalen Threshold verwenden.

3.1.2 Randobjekte entfernen

Die Objekt am Rand lassen sich entfernen, indem vorgängig zum Lokalisieren der Objekte alle Randpixel auf *weiss* gesetzt werden (alle Randobjekte sind damit verbunden). Diese Objekte können nun mit dem oben beschriebenen Extraktionsverfahren entfernt werden.

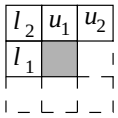
3.1.3 Label Map erstellen

Die Reiskörner könnten nun auch mit dem oben beschriebenen Exatraktionsverfahren gefunden werden. Da jedoch relativ viele Reiskörner vorhanden sind ist dieses Verfahren langsam (pro Objekt muss das mehrmals durchlaufen werden, speziell bei Objekten mit konkaven und konvexen Randabschnitten).

Im folgenden implementieren wir einen effizienteren Algorithmus, bei dem das Bild nur zweimal durchlaufen werden muss. In einem ersten Durchlauf werden alle Objekte mit einem einfachen Algorithmus markiert, wobei ein Objekt vorerst auch mit mehr als einem Label[1] markiert kann, also durch eine Menge von Labeln beschrieben wird. Im zweiten Durchlauf werden die Objekte dann mit eindeutigen Labeln versehen.

1. Durchlauf

Zuerst wird das Label-Bild, das die gleiche Grösse wie das Binär-Bild hat, mit Nullen initialisiert. Dann wird das Binär-Bild von oben links nach unten rechts durchlaufen und für jedes Objekt-Pixel ein Label im Label-Bild aufgrund der vier Nachbar-Pixel l_1, l_2,u_1, u_2 gewählt:



Das Label für das noch nicht markierte Pixel wird nach folgenden Regeln bestimmt:

a) alle vier Nachbarpixel gehören nicht zum Objekt (haben also den Wert 0) → dem Objekt-Pixel wird ein neues Label zugewiesen

b) einige der vier Nachbarpixel gehören zum Objekt und haben das gleiche Label → dem Objekt-Pixel wird dieses gleiche Label zugewiesen

c) mindestens zwei der Nachbarpixel gehören zum Objekt und haben verschiedene Labels → dem Objekt-Pixel wird irgend eines der Label zugewiesen, alle Labels, die zum Objekt gehören, werden in eine Gruppen-Liste eingetragen

Die Gruppen-Liste kann mit einem zweidimensionalen Array realisiert werden. Zwei Label $label_1$ und $label_2$ gehören dann zu einer Label-Gruppe, wenn in der Gruppen-Liste an der Stelle $(label_1, label_2)$ und resp. oder $(label_2, label_1)$ eine 1 eingetragen wird. Beim Aufdatieren der Gruppen-Liste wird im Fall c) für alle vier Nachbarn, die von 0 verschieden sind, ein Eintrag gemacht. Fall b) kann genau gleich wie Fall c) behandelt werden, die Einträge in der Gruppen-Liste liegen dann einfach

[1] Ein Label ist eine ganze Zahl, mit der jedes Objekt eindeutig gekennzeichnet wird

auf der Diagonalen. Eine Fallunterscheidung wäre aufwendiger als die Label einfach in die Liste einzutragen.

Hinweis Die Gruppen-Liste kann als kleiner (z.B. 10×10 grosser Array) initialisiert werden, beim Adressieren von Arrayelementen die ausserhalb dieses Bereichs liegen, wird der Array von Matlab automatisch erweitern.

In einem nächsten Schritt muss aus der Gruppen-Liste eine Label-Map erzeugt werden. Dazu steht die Matlab-Funktion `getLabelMap()` zur Verfügung, die als Parameter eine Gruppen-Liste erwartet und eine Label-Map zurückgibt.

2. Durchlauf

Im **zweiten Durchgang** kann nun aufgrund der Label-Map, die endgültige Label-Zuweisung erfolgen, dabei dient das Label eines Objektes als Index in die Label-Map, der ausgelesene Wert entspricht dem endgültigen Label.

3.1.4 Aufgabe

Implementieren Sie den Connected-Component Labeling Algorithmus wie oben beschrieben und stellen Sie das Label-Bild als Graustufen-Bild dar. Wie viele Reiskörner sind auf dem Bild zu finden?

3.2 Bestimmen von Objektmerkmalen

a) Bestimmen Sie die Grösse (Anzahl Pixel) der einzelnen Reiskörner, sowie die mittlere Grösse der Reiskörner mit Hilfe der Label Map. Markieren Sie anschliessend alle Reiskörner mit rot, die kleiner als der Mittelwert sind und mit grün alle Körner die grösser oder gleich dem Mittelwert sind.

b) Erstellen Sie ein Histogramm resp. die Verteilungsfunktion der Reiskorngrössen, wählen Sie selbst eine sinnvolle Klassengrenze.