Automatisches Lokalisieren, Lesen und Verifizieren von Beschriftungen auf Frachtcontainern in Farbbildern

Andreas Kuleschow, Marco Flachmann, Klaus Spinnler, Robert Schmidt

Fraunhofer Institut für Integrierte Schaltungen, Projektgruppe Industrielle Bildverarbeitung und Medizintechnik
Spk@iis.fhg.de

Abstract. An improved method for text lines retrieval is applied to identify containers. A new three-steps approach allows a good selection of characters. Natural interferences due the details of the container's surface are suppressed. An OCR-Task is simplified and processing time is reduced. The successful selection up to 99% of characters is obtained.

Einführung

Die Aufgabe, eine Aufschrift von unbekannter Farbe und Größe auf unterschiedlichsten Hintergründen automatisch und sicher zu lokalisieren, zu segmentieren und zu interpretieren ist eine herausfordernde Aufgabe in der Bildverarbeitung. Als Ansatz zu einer robusten Lösung bieten sich Split-and-Merge Verfahren an, bei der die Farbbilder zunächst in homogene Farbregionen aufgeteilt werden. Anschließend werden mehrere homogene Regionen mit ähnlichen Eigenschaften wieder vereint. Die spezielle Problematik der Schrifterkennung auf Containern liegt darin, dass u.a. Rostflecken, Verschmutzungen, Schatten oder profilierte Oberflächen die Farbe der Aufschrift so stark abweichen lassen, dass diese nur teilweise als homogen bezeichnet werden kann.

Stand der Technik

Die bekanntesten Methoden für das Teilen eines Farbbildes in homogene Regionen benutzen entweder eine Kombination mehrerer eindimensionaler Histogramme oder ein einzelnes mehrdimensionales Histogramm (Farbgramm). Für den ersteren Fall, der häufig bei schwarzweißen Bildern angewandt wird [1], ist die direkte Übertragung auf Farbbilder jedoch nicht praktikabel, da entweder für jeden Farbkanal ein eigenes Histogramm angefertigt und deren Ergebnisse zusammengefasst werden müssten [2], oder aber ein Histogramm mit bis zu 256^3 Farben angelegt werden müsste. In [3] wird eine Methode beschrieben, die bei Farbbildern grundsätzlich ein Intensitäts – Histogramm benutzt, aber in dieses nur solche Pixel einfügt, die

entsprechende statistische Eigenschaften besitzen (mittlere Abweichung, Gradient der Helligkeit, ...). Die Spitzen auf diesem sog. „Homogenitäts-Histogramm“ entsprechen der Intensität einer homogenen Region. In einem folgenden Schritt werden Subregionen mit ähnlichen Farbtönen in zusammenhängende Regionen vereint. Dieses Verfahren besitzt jedoch für unsere Aufgabenstellung zwei Nachteile: Zum einen ist der Zeitaufwand für die statistischen Berechnungen sehr groß, und zum anderen stellt die geeignete Definition des Kriteriums für die Ähnlichkeit der Farben ein Problem dar.
Eine andere Möglichkeit ein Farbbild zu segmentieren besteht darin, sog. mehrdimensionale Histogramme (Farbgramme) zu berechnen, und dort lokale Maxima zu finden. Alle ähnlichen Pixel können danach einer bestimmten Farbe zugeordnet werden, wobei als Ähnlichkeitskriterium der geometrische Abstand zwischen aktueller und Referenzfarbe verwendet werden kann [4]

Eigene Lösung

In unserem Verfahren verwenden wir den Ansatz der Farbgramm-Untersuchung sowohl für die Farbsegmentierung als auch für die Lokalisation der Beschriftungs-Region im Bild. Zur Reduktion des Rechenaufwands und Speicherbedarfs wird in unserem Ansatz das Farbgramm mit circa 20 - 25 Farbleveln je Stufe quantisiert. Die Verarbeitung von RGB-Bildern erzeugt dabei das Farbgramm in einem 3-D-Raum und ergibt dann 9^3 – 16^3 Bins. Im folgenden werden die entsprechend den Quantisierungsstufen in den Bins zusammengefasste Farbwerte als „Farbe“ bezeichnet.
Die Lokalisation der Schrift-Region ist ein wichtiger Bestandteil des Gesamtverfahrens. Je größer ein Bildfenster für die Generierung des Farbgramms ist, desto schwieriger ist es, darauf die Referenzfarben von einander zu trennen (das gleiche Problem kommt oft auch bei schwarz-weißen Bildern vor). Insgesamt gliedert sich das Verfahren in 5 Schritte:

1. Kontrastreiche Regionen (d.h. Kandidaten für Beschriftung) lokalisieren.
2. So gefundene Regionen (AOI) binarisieren.
3. In der AOI Objekte finden und gefundene Objekte in Zeilen zusammenfassen.
4. Zeichenerkennung auf Objekten in den Zeilen anwenden.
5. Die gefundenen Zeichenketten durch syntaktische Überprüfung mittels Vorwissen verifizieren.

Schritt 1 – Kandidaten für Beschriftungsregion lokalisieren

Im ersten Schritt der Schriftlokalisation werden auf dem Bild AOI-Kandidaten gesucht, die mit hoher Wahrscheinlichkeit Schriftzeichen enthalten. Die Suche nach solchen Kandidaten erfolgt über die Berechnung des Farbgramms in einem Fenster, das in einem äquidistanten Raster über das Bild geschoben wird („Sliding-Window“). Als Fensterhöhe empfiehlt sich etwa die Hälfte der zu erwartenden minimalen

Zeichenhöhe. In jedem Fenster werden die dominierenden Farben als Maxima des Farbgramms gesucht. Zwischen diesen Maxima dient nun die größte euklidische Distanz im Farbgramm als Maß für den höchsten Farbkontrast. Diese Distanz wird dann als Merkmal in eine Matrix (Merkmalsbild) eingetragen. Da wir eine Schrittweite des sliding window entsprechend der Fenstergröße verwenden, ergibt sich ein in der Auflösung entsprechend reduziertes Merkmalsbild. Da Schriftzüge i. d. R. einen hohen Kontrast gegenüber dem Hintergrund aufweisen, findet man mit Hilfe des Merkmalsbildes Bereiche, die mit hoher Wahrscheinlichkeit Schriftzüge enthalten. Die AOI-Kandidaten für die gesuchte Containerbeschriftung sind somit gefunden. Die Bereiche mit niedrigem Farbkontrast dagegen sind Kandidaten für den Hintergrund.

Fig. 1. Merkmalsbild (vergrößert) in der oberen linken Ecke des Originalbildes. Es sind hier nur ungefähr 2/3 des Originalbildes zu sehen.

Fig. 2. Die erste AOI (im Originalbild rot eingerahmt) ist gefunden.

Mit Hilfe des Merkmalsbildes lässt sich in einem weiteren Schritt gut die Textorientierung analysieren, dazu wird nach Binarisierung des Merkmalsbildes die Anzahl markierter Pixel pro Zeile ausgewertet. Zeilenweise Beschriftungen spiegeln

sich in der Maske als horizontale Strukturen wieder und lassen sich auf diese Weise erkennen. Bei senkrechten Beschriftungen ergeben sich dagegen vertikale Strukturen in der Maske (eine ähnliche Methode ist in [5] beschrieben).
In Fig.1 ist eine typische Containerbeschriftung (fast störungsfrei) gezeigt. Links oben in der Ecke ist das mit Hilfe des Farbgramms berechnete Merkmalsbild dargestellt. In Fig.2 ist rechts neben dem eingeblendeten Merkmalsbild eine Grafik gezeigt, die die Anzahl der markierten Pixel pro Zeile visualisiert. In diesem Bild stellt die Aufgabe, eine Zeile zu detektieren, kein Problem mehr dar. (Die oberste Zeile in Fig.2 ist schon gefunden und wurde deshalb schon aus dem Merkmalsbild gelöscht). Für Programmschritt 1 ergibt sich also ein Schema wie in Fig.3 gezeigt.

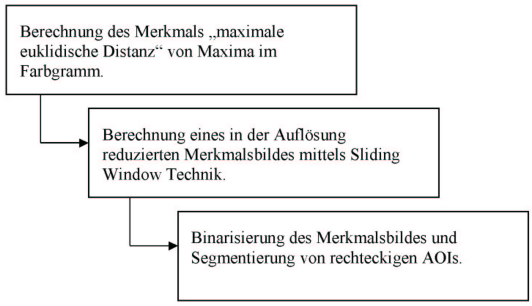

Fig. 3. Das Schema der Lokalisation der Beschriftungsregionen.

Schritt 2 - Binarisieren des Farbbildes.

In diesem Verfahrensschritt wird das gleiche Verfahren wie in Schritt 1 angewendet, jedoch mit einer unterschiedlichen Parametrierung. Dabei wird nur noch auf den in Schritt 1 ermittelten AOIs gearbeitet. Es wird wieder ein Farbgramm gebildet, diesmal für jede gefundene AOI, dabei entspricht die Fenstergröße der Größe der jeweiligen AOI. Wiederum werden die dominierenden Farben als lokale Maxima gefunden. Damit wird nun eine Pseudo-Farben Kopie der AOI erstellt, wobei jedes Pixel gemäß einem Index der dominierenden Farben gesetzt wird. Farben geringer Häufigkeit werden dabei benachbarten Farbwerten zugeschlagen, damit die Anzahl der Indizes gering bleibt.
Wenn die Anzahl dominierenden Referenzfarben zwei ist, dann kann man sofort die Farbe, die mit geringerer Häufigkeit auftritt als die Farbe der Zeichen interpretieren und das Bild entsprechend binarisieren. Normalerweise ist jedoch die Anzahl der vorhandenen Referenzfarben wesentlich größer als Zwei. In diesem Fall wird Anzahl der Farben weiter gemäß einem heuristischen Verfahren reduziert.
Dabei werden 2 Regeln aufgestellt:

1. Ist ein Objekt zu groß ist oder ragt über eine AOI-Grenze hinaus, ist es als Hintergrund zu interpretieren. Dies liegt darin begründet, dass die maximale und

minimale Größe der erkennbaren Zeichen direkt abhängig ist von der gewählten Fenstergröße und damit der Auflösung des Merkmalsbildes.

2. Ähnliche Farben sind zu einer zusammenzufassen. Es werden dafür 2 Schwellen verwendet, dabei ist die erste Schwelle konstant und niedrig. Die zweite Schwelle ist höher und wird aus dem Kontrast des Bildes berechnet und ist damit variabel. Es ergeben sich damit zwei Fallunterscheidungen:
   a) **euklidische Distanz < erste Schwelle**.
   Die Farben sind sehr ähnlich und werden zu einer Farbe zusammengefasst.
   b) **erste Schwelle < euklidische Distanz < zweite Schwelle**.
   Die Farben werden nur unter ganz bestimmten Bedingungen zusammengefaßt. Beispielsweise betrachtet man die Grenzen zwischen gefundenen Objekten, d.h. zusammenhängende Gebiete des gleichen Farbindex. Zwei Objekte mit verschiedenen Farben werden nur dann zusammengefasst, wenn sie eine gemeinsame Grenze haben, die eine geforderte Mindestlänge besitzen.

Fig. 4. Echte Farben werden auf der AOI durch Pseudofarben (Referenzfarben-Code) ersetzt.

Bevor der Schritt der Segmentierung abgeschlossen wird, muss geprüfet werden, ob die so gefundenen Objekte als Zeichen betrachtet werden können. Dafür werden für jedes Objekt Merkmale wie folgt berechnet. Für jedes Objekt, d.h. jeden Kandidaten für ein Zeichen wird ein zeilenweiser Lauflängen-Code berechnet und diese gefundenen Lauflängen jeder Zeile in ein Histogramm eingetragen.
Handelt es sich bei dem betrachteten Objekt um ein Zeichen, so tritt im Histogramm ein ausgeprägtes Maximum auf, das mit der Strichbreite des Zeichens korrespondiert. Bei einem Artefakt zufälliger Form tritt ein solches Maximum nicht auf.
Alle Objekte mit bislang noch unterschiedlichen Farbindices, die nach dieser Prüfung Zeichen enthalten können, werden im folgenden Binarisierungsschritt einheitlich als Objekt, d.h. Vordergrund, markiert, der Rest wird gelöscht, bzw. als Hintergrund betrachtet. Fig.4 zeigt 3 lokalisierte Zeilen der Beschriftung. Auf der oberen AOI sind schon die dominierenden fünf Referenzfarben gefunden und eine Pseudo-Farbbild der AOI, hier in Grauwert-Darstellung, erstellt. Fig. 5 zeigt Ergebnisse der Binarisierung. Außerhalb der AOIs werden alle Pixel schwarz gesetzt. Diese Methode ist sehr robust und kann Zeilen mit einer unterschiedlichen Färbung der Zeichen extrahieren, diese allerdings mit geringerer Sicherheit.

Schematisch funktioniert diese Stufe wie in Fig. 6 gezeigt. Damit ist die Phase der Segmentierung abgeschlossen.

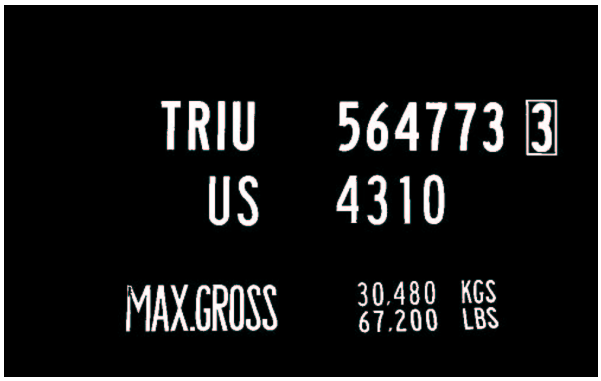

Fig. 5. Das binarisierte Bild.

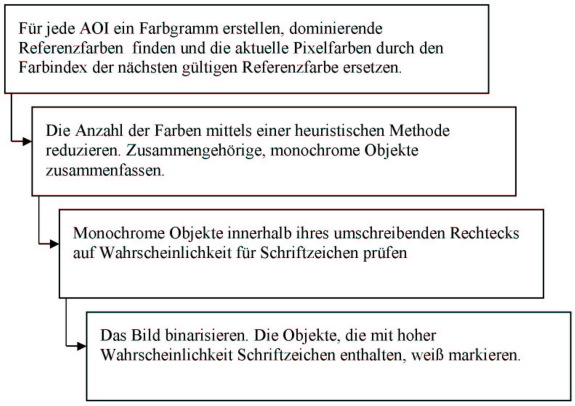

Fig. 6. Binarisieren des Farbbildes (Schema).

Schritt 3 - Objekte in Zeilen zusammenfassen.

Der letzte, noch verbleibende Schritt ist das Zusammenfassen aller segmentierten Objekte in Zeilen. In diesem Schritt fließt bereits Vorwissen über mögliche

Anordnungen der Beschriftung ein. Dabei werden nochmals einige Objekte aussortiert, andere zusammengefasst. Für diese Buchstabenverkettung kann angenommen werden, dass alle Zeichen (zumindest in einer Zeile) ungefähr die gleiche Größe besitzen. Abgesehen von geometrischen Verzerrungen kann auch angenommen werden, dass die korrespondierenden Grenzen von Zeichen einer Zeile bzw. einer Spalte annähernd horizontal bzw. vertikal ausgerichtet sind.
Dieses Gruppierung der Zeichen erfolgt iterativ. Zuerst werden Koordinaten der oberen (für waagerechte Zeilen) oder linken (für senkrechte Zeilen) Grenzen des umschreibendem Rechtecks der Zeichen untersucht. Falls diese Koordinaten z.B. alle einen ähnlichen Y-Wert aufweisen, wird angenommen, dass eine waagrechte Zeile vorliegt. Alle Zeichen mit einem umschreibendem Rechteck nahe dieses Y-Wertes werden zur Zeile hinzugefügt. Dann berechnen wir statistische Parameter der einzelnen umschreibenden Rechtecke (mittlere Breite und Höhe) und alle Objekte, die stark von diesen Werten abweichen, werden ausgeschlossen. Anschließend werden diejenigen Objekte gesucht, die nicht ganz die korrekte Position haben, aber deren mittleren Parameter mit denen der aktuellen Zeile übereinstimmen.

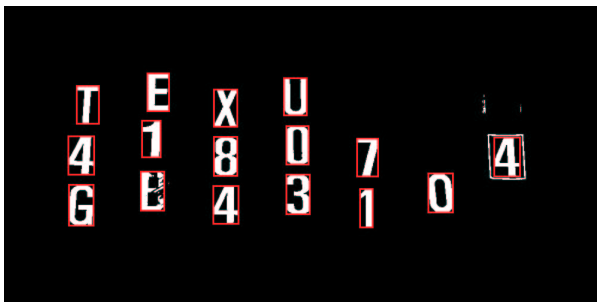

Fig. 7. Beispiel der Zeilenzusammenfassung. Das Zeichen „B“ ist richtig erfasst.

Fig. 8. Beispiel der Zeilenzusammenfassung. Das Zeichen “S” ist richtig erfasst.

Diese werden auch noch in die Zeile eingetragen. Dadurch wird erreicht, dass die Zeile durch diejenigen Objekte aufgefüllt wird, die durch zufällige Schäden vergrößert oder fragmentiert sind.
Fig. 7 und 8 zeigen Ergebnisse der Zeilenerfassung. Stark beschädigte Zeichen wie „B“ auf Fig. 7 und „S“ auf Fig. 8 wurden trotzdem richtig erfasst.
Das Schema des dritten Programmschrittes ist in Fig. 9 gezeigt.

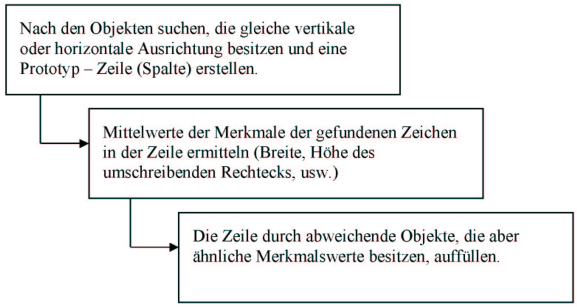

Fig. 9. Vorgehensweise bei der Zeilenzusammenfassung (das Schema).

Die segmentierten Zeichen einer Schriftzeile werden einem Template-Matching basierten OCR-Verfahren zugeführt. An die Zeichenerkennung der einzelnen Zeichen schließt sich eine Validierung der Zeichenkette mittels der erkannten Prüfziffer an.

Experimentelle Untersuchungen.

Das Verfahren wurde an realen Containerbildern getestet. Anhand einer Stichprobe aus 24 Bildern wurden folgende Ergebnisse erzielt (Tabelle 1):

Anz. Zeichen insgesamt	Richtig extrahiert	Extrahiert mit Schäden	Zeichen verloren	Artefakte hinzugefügt	Richtig gelesen	Falsch gelesen
261	257	2	2	0	258	1
100%	98,4%	0,8%	0,8%	0%	98,8%	0,4%

Tabelle 1. Ergebnisse und Erkennungsleistung

„Extrahiert mit Schäden“ bedeutet: die Zeichen wurden zwar an der richtigen Position gefunden, hatten aber Abweichungen von ihrer Form, die durch fehlerhafte Segmentierung verursacht wurde. Die Zeichen, die schon von Anfang durch Rostflecken usw. verzerrt waren, aber durchaus korrekt segmentiert wurden, sind in der Spalte „Richtig extrahiert“ eingetragen. Die extrahierten, beschädigten Zeichen

konnten dennoch zum Teil richtig gelesen werden. Eine Übersicht über die verwendeten Testbilder ist in Fig. 10 dargestellt.
Die Größe der relevanten Zeichen lag in unseren Bildern zwischen 35 und 120 Pixel, d.h. es trat ein Faktor 3,5 im Größenunterschied der verschiedenen Schriftzüge auf. Es wurden dennoch alle Zeichen ohne jegliche Veränderung der Verfahrens-Parameter gefunden.

Fig. 10. Die verwendete Stichprobe.

Ausblick

Das Einfügen rekursiver Prozeduren könnte das beschriebene Verfahren noch verbessern. So ist derzeit die Anzahl der Bins beim Farbgramm auf einen festen Wert eingestellt. Dadurch ist die Erkennungsleistung des Programms auf hohe Kontraste beschränkt: Das Bildbeispiel mit dem Schriftzug „GATU 404651“ lässt sich bei der Farbquantisierung in 9^3 Bins gerade noch richtig auswerten. Durch eine feinere Quantisierung könnte hier die Erkennungsleistung weiter gesteigert werden. Allerdings steigt bei einer Erhöhung der Farbauflösung die Anzahl der Bins im Farbgramm

mit der dritten Potenz und führt so sehr schnell zu erheblich mehr Rechenaufwand. Zur Minimierung dieses Problems könnte eine iterative Analyse des Bildes, beginnend mit sehr grober Quantisierung und entsprechend schneller Auswertung, erfolgen. Dabei würde die Quantisierung nur verfeinert werden, solange das erwartete Segmentierungs-Ergebnis nicht gefunden wird.
Mit dem vorgestellten Verfahren wurde auf der präsentierten Stichprobe eine hohe Erkennungsleistung erzielt. Durch die Implementierung verschiedener Validierungs-mechanismen und Einführung von Rückweisungsklassen konnte eine 100%ige Vermeidung von Falschklassifikationen erreicht werden.

References

1. N. Pal and S. Pal „A review on image segmentation techniques“ *Pattern Recognition* vol. 26, no. 9, pp. 1277-1294, 1993.
2. C.K. Yang and W.H. Tsai “Reduction of color space dimensionality by moment – preserving thresholding and its application for edge detection in color images”, *Pattern Recognit. Lett.*, vol. 17, pp. 481 – 490, 1996.
3. Heng-Da Cheng and Ying Sun “A Hierarchical Approach to Color Image Segmentation Using Homogeneity”, *IEEE Trans. on Image Processing,* vol. 9, no. 12, pp. 2071 – 2082, 2000.
4. Y. Zhong, K. Karu and A. K. Jain “Locating Text in complex color images”, *Pattern Recognition*, vol.28, no. 10, pp. 1523 – 1535, 1995
5. S. Messelodi, C.M. Modena “Automatic identification and skew estimation of text lines in real scene images” *Pattern Recognition*, vol. 32 no. 5, pp. 791 – 810, 1999.