

OCR am Einzahlungsschein

Empfangsschein / Récépissé / Ricevuta	Einzahlung Giro	Versement Virement	Versamento Girata
<p>Einzahlung für / Versement pour / Versamento per Berner Kantonalbank 3001 Bern</p> <p>Zugunsten von / En faveur de / A favore di My Company Ltd. Company Address Rd. 23 3001 City</p> <p>Konto/Compte/Conto 01-200000-7 CHF</p> <p>100 . 00</p> <p>Einbezahlt von / Versé par / Versato da 99 99990 00000 00000 00011 11111 Heinz Müller Payer Court Street 23 3072 Payertown Switzerland</p> <p>Die Annahmestelle L'office de dépôt L'ufficio d'accettazione</p>	<p>Einzahlung für / Versement pour / Versamento per Berner Kantonalbank 3001 Bern</p> <p>Zugunsten von / En faveur de / A favore di My Company Ltd. Company Address Rd. 23 3001 City</p> <p>Konto/Compte/Conto 01-200000-7 CHF</p> <p>100 . 00</p> <p>609</p>	<p>Keine Mitteilungen anbringen Pas de communication Non agglungele comunicazioni</p> <p>Referenz-Nr./N° de référence/N° di riferimento 99 99990 00000 00000 00011 11111</p> <p>Einbezahlt von / Versé par / Versato da Heinz Müller Payer Court Street 23 3072 Payertown Switzerland</p>	

0100000100009>999999900000000000000001111111+ 012000007>



Christian Kaufmann

Lars Piller

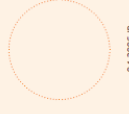



BFH Maschinelle Bildverarbeitung

Inhalt

Problemstellung.....	2
Lösungsansatz	2
Herausforderungen.....	2
Finden, Ordnen und Ausschneiden.....	3
Region of Intrest (ROI)	3
Filterung.....	4
Binarisieren	4
Konturen finden, Koordinaten berechnen und Zwischenspeichern.....	4
Sortieren und als einzelne Bilder speichern	4
Zeichenerkennung	5
Ablauf der Bilderkennung	5
Präprozessor	5
Trainingsbilder lesen und Matrix [CvKNearest] schreiben	5
Zeichen erkennen	5
In den Zwischenspeicher kopieren	6
Fazit.....	6

Problemstellung

Aus einem Bild eines Einzahlungsscheins soll die ESR-Zeile ausgelesen werden und als Text in die Zwischenablage gespeichert werden. So kann sie beliebig weiterverwendet werden (E-Banking, Textdokument, etc.). Es ist nur die ESR-Zeile nötig, da sie, wie in folgender Grafik gezeigt, alle wichtigen Informationen über den Inhalt des Einzahlungsscheins enthält.

Empfangsschein / Récépissé / Ricevuta	Einzahlung Giro	Versement Virement	Versamento Girata
<small>Einzahlung für / Versement pour / Versamento per</small> Robert Schneider SA Grands magasins Case postale 2501 Biel/Bienne	<small>Einzahlung für / Versement pour / Versamento per</small> Robert Schneider SA Grands magasins Case postale 2501 Biel/Bienne	<small>Keine Mitteilungen anbringen Pas de communications Non aggiunte comunicazioni</small>	 04.2006 IR
<small>Konto / Compte / Conto</small> CHF 01-162-8  3949 75 <small>Einbezahlt von / Versé par / Versato da</small> 120000000000234478943216899	<small>Konto / Compte / Conto</small> CHF 01-162-8  3949 75 609	<small>Referenz-Nr./N° de référence/N° di riferimento</small> 12 00000 00000 23447 89432 16899 <small>Einbezahlt von / Versé par / Versato da</small> Rutschmann Pia Marktgasse 28 9400 Rorschach	442.05
 Die Annahmestelle L'office de dépôt L'ufficio d'accettazione			

0100003949753>120000000000234478943216899+ 010001628>

- Belegart (codiert) s. 3.5.5.3
- Betrag (+ Prüfziffer)
- Referenznummer (+ Prüfziffer)
- Teilnehmernummer

In erster Instanz soll dies mit Scan's von Einzahlungsscheinen geschehen weil dabei die Gefahr mangelnder Bildqualität, Drehung und Verzerrung nicht besteht. Erweiternd soll mit Bildern von Kameras (Handykamera, Webcam des Notebooks) gearbeitet werden.

Lösungsansatz

Die ESR-Zeile soll als ganze Zeile gefunden und segmentiert werden. Anschliessend sollen die einzelnen Zeichen mittels Konturerkennung aus der gesamten Zeile separiert, ausgeschnitten und zwischengespeichert werden.

Die Zeichenerkennung soll mittels k-nearest neighbour realisiert werden. Diese Funktion schaut das Bild an und sucht in den Musterbildern nach dem Nächstgelegenen. Jedoch muss für diese Funktion jedes Zeichen am selben Ort des Bildes sein und dieselbe Auflösung haben.

Herausforderungen

Die erste grosse Hürde war es, die OpenCV library unter Visual Studio 2010 zum Laufen zu bringen. Als wir dies geschafft hatten, mussten wir die Grundlagen von OpenCV erarbeiten, was sehr zeitaufwendig war. Somit dauerte es schon sehr lange, bis wir eine funktionierende

Entwicklungsumgebung hatten. Ebenfalls mussten wir feststellen, dass der Aufwand grösser wird als zu Anfang gedacht.

Die nächste grosse Herausforderung war die Bildqualität. Die Bildqualität muss einen gewissen Level haben, sonst ist es gar nicht möglich die Zeichen zu erkennen. Zum Beispiel, wenn die Auflösung zu niedrig ist, können die Zahlen eins und sieben nicht mehr voneinander unterschieden werden.

Finden, Ordnen und Ausschneiden

Im ersten Teil unseres Programms wird die ESR-Zeile auf dem Einzahlungsschein gesucht und so vorverarbeitet, dass im zweiten Teil (Zeichenerkennung) mittels Template Matching aus den Bildern die Zeichen erkannt und wiedergegeben werden können. Dies geschieht in folgenden Schritten:

Region of Interest (ROI)

Da für unser Vorhaben nur die ESR-Zeile von Bedeutung ist, definieren wir diese als ROI. Aufgrund der Tatsache dass mehrere Formatmöglichkeiten für Einzahlungsscheine existieren, selektieren wir diese ROI per Auswahl mit der Maus aus unserem Originalbild. Alle folgenden Schritte werden dann nur noch auf dieses selektierte Bildsegment angewandt.

Empfangsschein / Récépissé / Ricevuta	Einzahlung Giro	Versement Virement	Versamento Girata
Empfänger für / Versament pour / Versamento per Berner Kantonalbank 3001 Bern	Empfänger für / Versament pour / Versamento per Berner Kantonalbank 3001 Bern	Keine Mitteilungen anbringen Pas de communication Non applegiare comunicazioni	
Zugunsten von / En faveur de / A favore di My Company Ltd. Company Address Rd. 23 3001 City	Zugunsten von / En faveur de / A favore di My Company Ltd. Company Address Rd. 23 3001 City	Referenz-Nr. der Referenz-Nr. / riferimento 99 99990 00000 00000 00011 11111	
Konto/Compte/Conto CHF 01-200000-7 100 . 00	Konto/Compte/Conto CHF 01-200000-7 100 . 00	Einbezahl von / Verse par / Versato da Heinz Müller Payer Court Street 23 3072 Payertown Switzerland	
Einbezahl von / Verse par / Versato da 99 99990 00000 00000 00011 11111 Heinz Müller Payer Court Street 23 3072 Payertown Switzerland			
Die Annahmestelle L'office de dépôt L'ufficio d'accettazione			

0100000100009>999999000000000000000001111111+ 012000007>

609

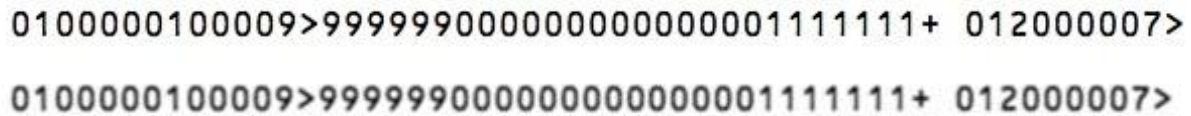
0100000100009>999999000000000000000001111111+ 012000007>

Wie Sie der Grafik entnehmen können, ist unser neues Source Image nur noch die ESR-Zeile auf weissem Grund. Dies würde auch für andere Formate des Einzahlungsscheins so sein, da sich an der Tatsache dass sich die Zeile auf weissem Grund befindet nichts verändert. Danach wird die ROI mittels der OpenCV Funktion „cvCvtColor(..., ..., CV_RGB2GRAY)“ in ein Graustufenbild umgewandelt.

Filterung

Nun werden die Konturen in der ROI mit einem Filter verwischt um die Grundlage für ein besseres Binärbild zu schaffen. Dies ist nötig weil die Zeichen zum Teil sehr dünne Linien enthalten, die sonst eventuell beim Binarisieren verloren gehen könnten.

Wir haben dazu die OpenCV Funktion „cvSmooth()“ in Kombination mit einem Gauss’schen Filterkern verwendet.



Hier die ROI vor (oben) und nach (unten) dem Filter.

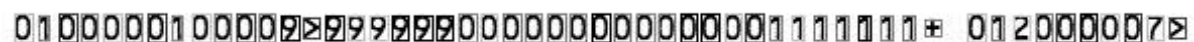
Binarisieren

Jetzt wird aus dem gefilterten Graustufenbild ein Binärbild erstellt. Die OpenCV Funktion „cvThreshold()“ löst dieses Problem indem sie den Grauwert jedes Pixels mit einem gewählten Grenzwert vergleicht und je nachdem, ob dieser Wert grösser oder kleiner als der Grenzwert ist, auf weiss oder schwarz setzt. Das daraus erhaltene Resultat sieht wie folgt aus.



Konturen finden, Koordinaten berechnen und Zwischenspeichern

Da für das Template Matching jedes Zeichen ein einzelnes Bild sein muss, werden diese Zeichen nun mit den OpenCV Funktionen „cvFindContours()“ und „cvBoundingRect()“ gefunden und eingegrenzt. „cvFindContours()“ findet alle schwarz-weiss Übergänge und speichert deren Koordinaten in einer Struktur. „cvBoundingRect()“ berechnet aus diesen Konturpunkten für jedes Zeichen das kleinste, aufrechtstehende Rechteck und speichert die Daten dieser Rechtecke auch wieder in einer Struktur. Zur Veranschaulichung können diese Rechtecke gezeichnet werden.



Sortieren und als einzelne Bilder speichern

Das Problem von „cvFindContours()“ ist, dass die Koordinaten der gefundenen Konturen nicht anhand ihrer Positionen sortiert gespeichert werden. Deshalb haben wir uns dazu entschieden diesen Teil mit dem Bubblesort-Algorithmus in einer separaten Schleife zu bewerkstelligen. Dazu werden die Daten der Rechtecke in ein 2D Array gepackt und anschliessend anhand der X Koordinate ihrer oberen linken Eckpunkte in aufsteigender Reihenfolge geordnet. Anschliessend werden die Segmente die von den Rechtecken umschlossen sind von links nach rechts ausgeschnitten und in einer weiteren Struktur als einzelne Bilder mit aufsteigenden Indizes gespeichert.

An diesem Punkt im Programm sind die Einzelbilder so vorbereitet, dass mit der Zeichenerkennung begonnen werden kann.

Zeichenerkennung

Ablauf der Bilderkennung

Zuerst werden die Testbilder mit einer for-Schleife durch den Präprozessor geschickt. Anschliessend werden die Bilder in einer Matrix von Typ [CvKNearest] mit einer Klasse gespeichert. Pro Klasse wird ein Bild eingelesen. Für die Zeichenerkennung wird das Bild, welches im vorherigen Teil ausgeschnitten wurde, zuerst durch den Präprozessor gelassen und anschliessend mit der Funktion k-nearest neighbour dem entsprechenden Zeichen zugewiesen. Zum Schluss wird die ganze ESR-Zeile als Zeichenreihe in den Zwischenspeicher kopiert, damit sie der Anwender mit ctrl + V, zum Beispiel beim Onlinebanking, einfügen kann.

Präprozessor

Der Präprozessor bereitet das Bild so auf, dass die Auflösung 40x40 Pixel beträgt, das Zeichen nicht verzogen ist und in der Mitte des Bildes liegt. Dies wird wie folgt bewerkstelligt. Zuerst wird von links nach rechts mit der Funktion „cvGetCol()“ jede Spalte einzeln herausgelesen und anhand eines Vergleiches entschieden, ob sich ein schwarzes Pixel darin befindet oder nicht. Sobald das erste schwarze Pixel auftaucht wird diese Spalte in eine Variable geschrieben. Das ganze Prozedere wird weitergeführt und sobald nur noch weisse Pixel vorkommen wird die Spalte auch in eine Variable geschrieben. Dasselbe wird für die Y-Achse wiederholt. Anschliessend wird das so randlos ausgeschnittene Zeichen in die Mitte des Bildes kopiert, auf die Auflösung 40x40 skaliert und zurück an die Funktion übergeben.



Das quadratische 40x40 Bild vom Präprozessor

Trainingsbilder lesen und Matrix [CvKNearest] schreiben

Zuerst wird das Trainingsbild aus einem definierten Ordner geladen und von einer for Schleife durchlaufen. In dieser for Schleife wird zuerst der Präprozessor durchlaufen, anschliessend werden die binären Daten des Bildes in einen Vektor geschrieben, welcher somit nur aus Nullen und Einsen besteht. Am Schluss der for Schleife werden der Klassenvektor und der Bilddatenvektor mittels der Funktion „CvKNearest()“ in die Matrix „knn“ geschrieben. Diese for Schleife hat im Fall unseres Zeichensatzes 12 Iterationen (Ziffern 0 bis 9 und Zeichen + sowie >).

Zeichen erkennen

Die Bilder, welche vom ersten Programmteil (Finden, Ordnen und Ausschneiden) als [IplImage] in einer Struktur zur Verfügung gestellt werden, werden wie die Trainingsbilder aufbereitet. Anschliessend werden sie mit der Funktion „find_nearest()“ und der „knn“ Matrix klassifiziert und es wird ihnen anhand der Klasse das entsprechende Zeichen zugewiesen. Diese Zeichen sind der

Rückgabewert der Erkennungsfunktion. Für die Erkennung wurde die Funktion `k nearest neighbours` gewählt. Dies ist eine sehr einfache Funktion, welche das Bild mit den Testbildern vergleicht und die nächstgelegene Klasse zurückgibt. Der Nachteil ist, dass diese Funktion einen hohen Rechenaufwand hat und viel Arbeitsspeicher benötigt. Für unsere Anwendung spielt das aber keine grosse Rolle, da der Zeichensatz und somit die Anzahl Trainingsbilder eher klein ist.

In den Zwischenspeicher kopieren

Die ganze Zeichenkette wird mit Hilfe einer Windowsfunktion in den Zwischenspeicher kopiert. Hierbei ist zu beachten dass die Variable, in der die Zeichenkette gespeichert werden soll, leer initialisiert wird. Ansonsten kann es zu unangenehmen Überraschungen kommen.

Fazit

OpenCV bietet sehr viele Funktionen und Möglichkeiten. Einerseits kann man viele Probleme elegant lösen. Hingegen entwickelt es sich vor allem für Beginner zu einem Dickicht aus Funktionen und Datentypen, dass auch mit einem grossen Buschmesser nur langsam durchschritten werden kann.

Unser Grundziel wurde erreicht und das Programm ist erweiterbar. Der nächste Schritt wäre zum Beispiel, dass das Programm die ESR Zeile selber findet.

Unsere Arbeitsteilung war sinnvoll. Wir definierten eine klare Schnittstelle, konnten somit unabhängig voneinander arbeiten und am Schluss die Teile zusammenfügen. Das hat sehr gut funktioniert.

Es war für uns eine lehrreiche und interessante Erfahrung. Wir konnten einen Teil des gelernten vom Fache „Funktionale Softwareentwicklung“ anwenden und vertiefen. Wir erhielten einen Einblick in die Bildverarbeitung und stellten fest, dass es sich um ein sehr umfangreiches Gebiet handelt mit dem man sich jahrelang beschäftigen kann.