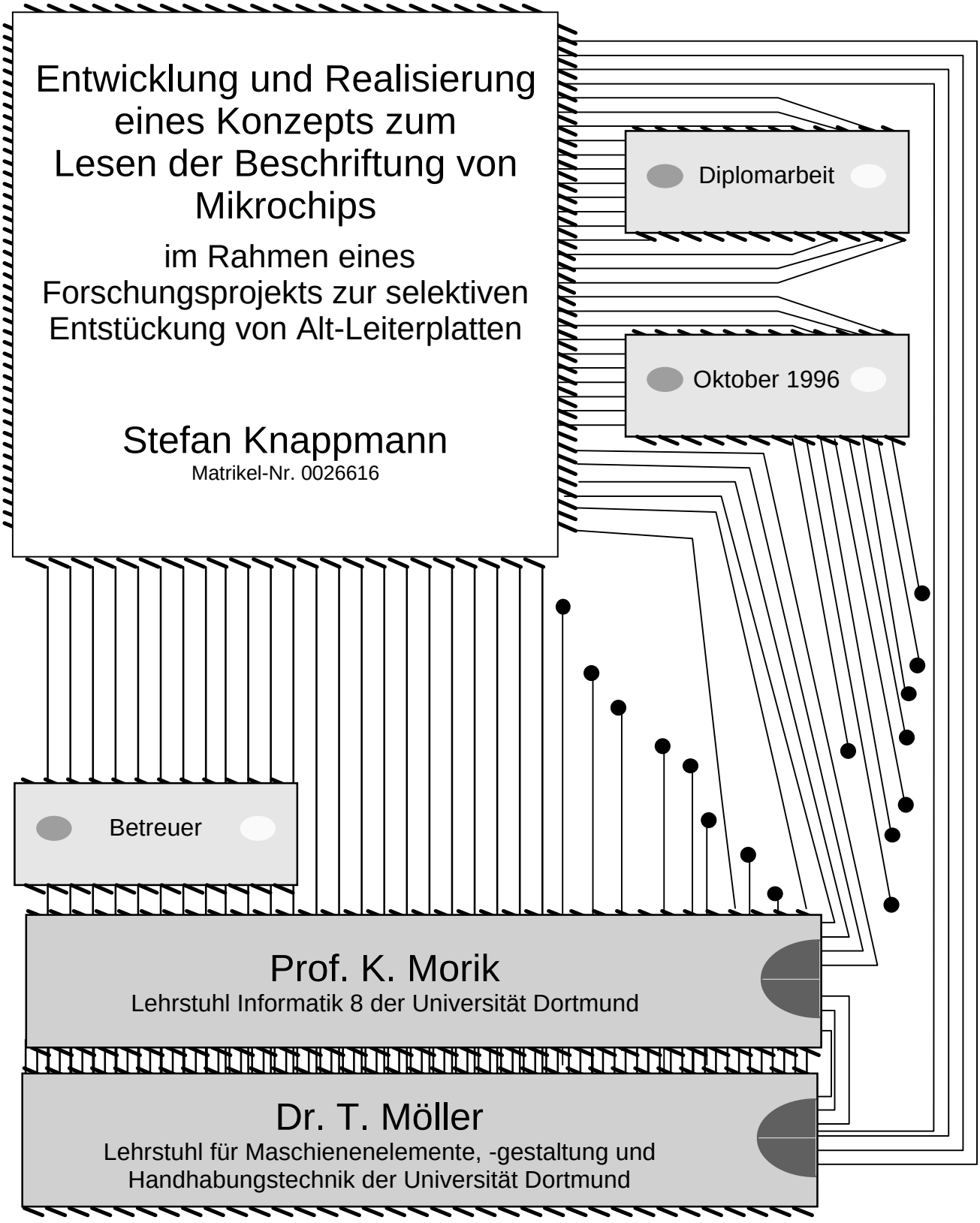
Entwicklung und Realisierung eines Konzepts zum Lesen der Beschriftung von Mikrochips

im Rahmen eines Forschungsprojekts zur selektiven Entstückung von Alt-Leiterplatten

Stefan Knappmann

Matrikel-Nr. 0026616



Inhaltsverzeichnis

1 EINLEITUNG 4

1.1 Eine vollautomatische selektive Entstückungsanlage 4
1.2 Aufgabenbeschreibung 6
1.3 Übersicht über den Inhalt 7

2 ENTWICKLUNG EINES KONZEPTES 8

2.1 Digitalisierung 9
2.2 Vorverarbeitung 10
2.2.1 Detailunterdrückungsfilter 11
2.2.2 Detailverstärkungsfilter 11
2.3 Vorsegmentierung 13
2.4 Binarisierung 15
2.4.1 Schwellwertbestimmungsverfahren 16
2.4.2 Pixelaggregation 17
2.4.3 Invertierung 18
2.5 Segmentierung 18
2.5.1 Punkte zu Zeichenhypothesen 19
2.5.2 Zeichenhypothesen zu Worthypothesen 20
2.5.3 Worthypothesen zu Linienhypothesen 21
2.5.4 Entfernen von Störungen, Logos und Gehäuseeinbuchtungen 21
2.6 Bestimmung der Orientierung 22
2.6.1 Achsenorientierung 22
2.6.2 Richtungsorientierung 24
2.7 Zeichenklassifizierung 25
2.7.1 Merkmale 26
2.7.2 Klassifikationsverfahren 33
2.8 Trennen von Zeichen 36
2.9 Zusammenfassung des daraus abgeleiteten Konzepts 37

3 ERGEBNISSE DER REALISIERUNG UND VERBESSERUNGEN 40

3.1 Digitalisierung 40
3.2 Vorverarbeitung 40
3.3 Vorsegmentierung 46
3.4 Binarisierung 48
3.4.1 Schwellwertbestimmungsverfahren 48
3.4.2 Pixelaggregation 51
3.4.3 Invertierung 53
3.5 Segmentierung 53
3.5.1 Punkte zu Zeichenhypothesen 54

3.5.2 Zeichenhypothesen zu Worthypothesen 55
3.5.3 Worthypothesen zu Linienhypothesen 58
3.5.4 Linienhypothesen zu Blockhypothesen 58
3.6 Bestimmung der Orientierung 60
3.6.1 Achsenorientierung 60
3.6.2 Richtungsorientierung 66
3.7 Zeichenklassifizierung 67
3.7.1 Merkmale 67
3.7.2 Klassifikationsverfahren 73
3.8 Trennen von Zeichen 78
3.8.1 Fehlende Trennstellen 79
3.8.2 Falsche Trennstellen 80
3.8.3 Trennreihenfolge 81
3.9 Zusammenfassung des realisierten Konzepts 82

4 ZUSAMMENFASSUNG UND AUSBLICK 85

5 ANHANG: NEGATIVE HU-INVARIANTEN 88

6 WORTERKLÄRUNGEN 91

7 LITERATUR 94

1 Einleitung

Die rasende Weiterentwicklung der Mikroelektronik eröffnet dieser zum einen immer neue Anwendungsfelder, mit dementsprechend immer größeren Stückzahlen, zum anderen veralten hierdurch elektrotechnische Geräte in nur wenigen Jahren. Beides zusammen führt zu großen Mengen Elektronikschrott, dessen fachgerechte Entsorgung Probleme bereitet. Um die Schrottmenge zu reduzieren, hat die Europäische Union ein Forschungsprojekt genehmigt, mit dem Ziel: schadstoffhaltige bzw. wiederverwertbare Bauteile auf Leiterplatten automatisch zu erkennen, zu entnehmen und zu entsorgen bzw. wiederzuverwerten.[1] Am Lehrstuhl für Maschinenelemente, -gestaltung und Handhabungstechnik der Universität Dortmund von Prof. Dr. Ing. W. Kreis und dem Lehrstuhl Informatik 8 von Prof. K. Morik der Universität Dortmund, wird an der Lösung dieser Aufgabe gearbeitet.

Nach Ideen von Dr. T. Möller [Möll96] wird eine vollautomatischen Anlage zur selektiven Entstückung von Leiterplatten entwickelt. Um eine Einordnung dieser Arbeit in das Gesamtsystem zu ermöglichen, soll zuerst in Abschnitt 1.1 der Aufbau und Ablauf der gesamten Entstückungsanlage beschrieben werden. In Abschnitt 1.2 wird dann die Aufgabenstellung dieser Arbeit genauer erläutert. Abschnitt 1.3 gibt einen Überblick über den Aufbau der restlichen Arbeit.

Einige gängige Begriffe aus der Bildverarbeitung wurden im Text der Diplomarbeit nicht mehr definiert. Der Text wird hierdurch flüssiger. Damit auch Neulinge auf dem Gebiet der Bildverarbeitung mit dieser Arbeit zurechtkommen, werden im Kapitel 6 „Worterklärungen“ diese Begriffe definiert. Alle Begriffe, die dort zu finden sind, sind bei ihrem ersten Auftreten in jedem Kapitel kursiv geschrieben.

1.1 Eine vollautomatische selektive Entstückungsanlage

Die Entstückungsanlage besteht aus drei modularen Arbeitsstationen: Vision I, Vision II und einer Entlötstation. Die drei Stationen sind über ein Fließband verbunden, so daß die zu entstückenden Leiterplatten automatisch zu jeder Arbeitsstation transportiert werden können. Den Aufbau der Anlage zeigt schematisch *Abbildung 1.1*.
Die Arbeitsstation Vision I besteht aus einer Kamera und einer passenden Beleuchtung. Die Kamera macht ein Bild von der gesamten Leiterplatte. Das Bild wird untersucht und die Lage von Bauteilen bestimmt. Werden mehrere ungefähr gleich große Bauteile gefunden, werden deren Positionen auf eine regelmäßige Anordnung untersucht. Alle ermittelten Positionen und Regelmäßigkeiten in der Anordnung werden an die Arbeitsstation Vision II übermittelt. Die Leiterplatte wird dann zu dieser Station transportiert. Für eine

[1] "Automatic reuse of electronic components on printed circuit boards and separation of toxic components."

ausführlichere Beschreibung der Station Vision I sei auf die Arbeit von S. Albers [Albe95] verwiesen.

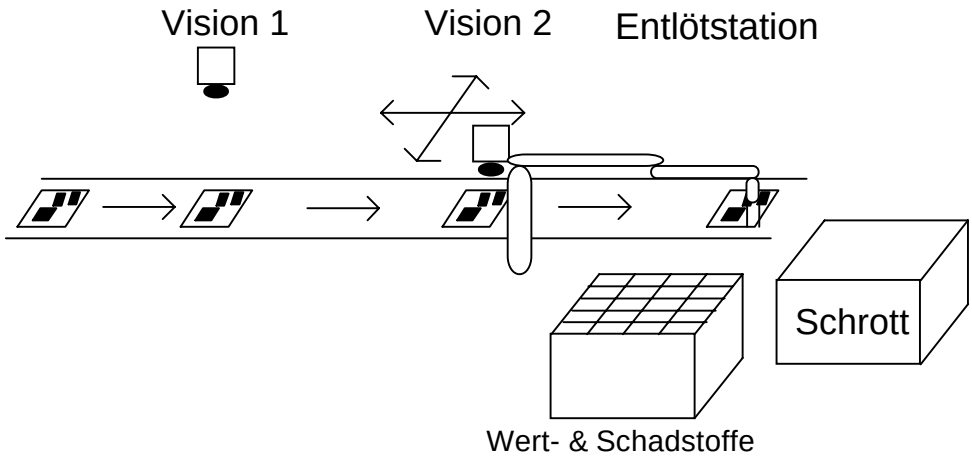


Abbildung 1.1: Schematischer Aufbau der Entstückungsanlage

An der Arbeitsstation Vision II befindet sich eine zweite Kamera mit Beleuchtung. Kamera und Beleuchtung sind in X- und Y-Richtung verfahrbar. Die Kamera fährt alle von der Arbeitsstation Vision I erhaltenen Positionen an und macht eine Nahaufnahme des dortigen Bauteils. Die genaue Größe, Position und Bauart werden bestimmt.
Die gesammelten Informationen werden mit den Einträgen in einer Datenbank verglichen. In der Datenbank sind alle schadstoffhaltigen und wertvollen Bauteile gespeichert. Wertlose, schadstofffreie und damit uninteressante Bauteile sind in der Datenbank nur enthalten, wenn ihr Vorhandensein zur besseren Abgrenzung zu den interessanten Bauteilen sinnvoll ist.
Wird kein passender Eintrag in der Datenbank gefunden, wird die aktuelle Position als uninteressant eingestuft und ignoriert. Es handelt sich dann bei der aktuellen Position um ein wertloses und schadstofffreies Bauteil oder um einen von Vision I falsch eingeschätzten Bereich. Wurde genau ein passender Eintrag in der Datenbank gefunden, ist das Bauteil eindeutig identifiziert. Wurden mehrere passende Einträge gefunden, muß die Beschriftung des Bauteils gelesen werden, um zu einem eindeutigen Ergebnis zu kommen. Hierzu wird die ermittelte Bauteiloberfläche an ein Schrifterkennungssystem übergeben. Das Ergebnis der Schrifterkennung, die gelesene Aufschrift, wird mit den fraglichen Datenbankeinträgen verglichen. Hierdurch wird das Bauteil entweder eindeutig identifiziert oder, wenn kein zur Beschriftung passender Datenbankeintrag vorhanden ist, als uninteressant eingestuft.
Wurde ein Bauteil eindeutig identifiziert, so gibt die Datenbank Aufschluß ob es schadstoffhaltig oder wertvoll ist. Sind von regelmäßig angeordneten Bauteilen die Ersten vom selben Typ wird angenommen, daß alle weiteren Bauteile ebenfalls von diesem Typ sind und werden nicht weiter von Vision II

inspiziert. Wurden alle Bauteile untersucht, bzw. ihre Untersuchung aufgrund ihrer Anordnung ausgelassen, wird die Leiterplatte zur Entstückungsstation befördert. Die genaue Position und der Typ jedes wertvollen oder schadstoffhaltigen Bauteils wird an die Entlötstation weitergeleitet.

An der Entlötstation befindet sich ein Roboter mit wechselbaren Entlötwerkzeugen und Greifern. In einer Datenbank ist das Wissen gespeichert mit welchen Werkzeugen und Verfahren ein Bauteiltyp bearbeitet werden muß. Die als wertvoll oder schadstoffhaltig eingestuften Bauelemente werden entlötet, entnommen und, nach Typen sortiert, in bereitstehenden Kästen abgelegt. Abschließend verläßt die Leiterplatte die Entstückungsanlage. Sie muß als Restmüll entsorgt werden.

1.2 Aufgabenbeschreibung

Durch geeignete Verfahren werden an der Arbeitsstation Vision II Lage, Abmessungen und Bauform der Bauteile auf einer Leiterplatte bestimmt. Um einen Chip eindeutig zu identifizieren, reichen diese Daten oft nicht aus; seine Aufschrift muß gelesen werden.

Die Komplexität des Lesens der Aufschrift ergibt sich dabei aus der Fülle verschiedener Chips unterschiedlichster Hersteller. So sind Schrifttyp, -position, -größe und -orientierung größtenteils unbekannt. Gehäuseeinbuchtungen und Firmenlogos müssen von den Schriftzeichen unterschieden werden. Schriftfarbe und Hintergrundfarbe sind von Chip zu Chip verschieden, ebenso wie die Oberflächenstruktur (matt, glänzend, rauh, etc.). Während kleine Chips vom Platzmangel geprägt sind, die Zeichen sehr eng beieinander stehen und ihre Separierung hohe Anforderungen stellt, spiegelt bei großen, teuren Chips (CPUs) das Schriftlayout die Komplexität des Schaltkreislayouts wieder.

Im Rahmen dieser Diplomarbeit wird ein Konzept entwickelt, mit dem sich die Beschriftung auf Mikrochips lesen läßt. Anschließend wird das Konzept als Prototyp realisiert, um die Machbarkeit und die Grenzen des Vorhabens zu verdeutlichen.

Das zu entwickelnde Konzept wird auf den vorhandenen Verfahren ähnlicher Anwendungen, die am ehesten für diese Anwendung geeignet scheinen, basieren. Für Problembereiche in denen es noch keine Lösungen gibt, oder die vorhandenen Lösungen unzureichend für diese Anwendung erscheinen, wird nach neuen Lösungsansätzen geforscht.

Die Ergebnisse und Erfahrungen mit dem realisierten Konzept geben dann Aufschluß über die generelle Machbarkeit der Schrifterkennung auf Mikrochips. Die Schwächen der verwendeten Verfahren sollten aufgezeigt und mögliche Verbesserungen vorgeschlagen werden, bzw. weiterer Forschungsbedarf festgestellt werden.

Wie in der Projektbeschreibung erklärt, erhält die Schrifterkennung die Eckkoordinaten der zu untersuchenden Chipoberfläche als Eingabe. Ausgabe

der Schrifterkennung ist die erkannte Chipaufschrift. Eine Fehlerkorrektur der gefunden Aufschrift mittels eines Wörterbuches oder ein Abgleich mit einer Datenbank zur Bestimmung des Chiptyps ist nicht mehr Aufgabe dieser Diplomarbeit.

1.3 Übersicht über den Inhalt

Ausgehend vom Kamerabild werden Konzepte erarbeitet, zur Bildverbesserung und Binarisierung, zur Separierung der Beschriftung vom Hintergrund, zur Isolierung der Zeichen, zur Ermittlung der Schriftorientierung und zur Erkennung der einzelnen Schriftzeichen und Firmenlogos.

In der Einleitung zu Kapitel 2 wird ein allgemeiner Überblick über das Vorgehen von Schrifterkennungssystemen gegeben. In den Unterkapiteln von Kapitel 2 werden zu jedem Bearbeitungsschritt gängige Lösungsansätze diskutiert und aus diesen ein Konzept zur Lösung der gestellten Aufgabe entwickelt. In Kapitel 3 werden die Ergebnisse des realisierten Konzeptes vorgestellt. Ausgehend von den Ergebnissen werden Verbesserungsvorschläge gemacht. Soweit die Verbesserungen umgesetzt wurden, werden auch deren Ergebnisse vorgestellt. In Kapitel 4 werden die wichtigsten Ergebnisse und offenen Probleme zusammengefaßt.

2 Entwicklung eines Konzeptes

Die Lösung einer Schrifterkennungsaufgabe läßt sich in mehrere Teilaufgaben gliedern. Unabhängig vom gewählten Lösungsweg sind immer die folgenden Teilprobleme zu bewältigen:

- Digitalisierung
- Vorverarbeitung
- Vorsegmentierung
- Binariesierung
- Segmentierung
- Orientierungsbestimmung
- Zeichenerkennung
- Nachbearbeitung

Jedes Schrifterkennungssystem hat zuerst eine Hardwarekomponente zur **Digitalisierung** einer Vorlage. Dies kann ein Scanner oder eine Kamera sein. Danach folgt eine **Vorverarbeitung** zur Korrektur von Digitalisierungsfehlern bzw. zur grundsätzlichen Bildverbesserung. Häufige Digitalisierungsfehler sind ungleichmäßige Beleuchtung und Bildrauschen. In der Vorverarbeitung werden in der Regel Filter eingesetzt. Je nach Filter wird das Bild geglättet oder Kanten hervorgehoben.
Anschließend erfolgt eine **Vorsegmentierung**. Aufgabe der Vorsegmentierung ist es, zusammengehörende Bildbereiche zu erkennen und voneinander abzugrenzen. Solche Bereiche sind z.B. Grafiken, Photos, Textblöcke oder Freiräume.
Die vorverarbeiteten Bildbereiche mit Text werden dann **binarisiert**. Im Binärbild sind schwarze Punkte Zeichenpunkte und weiße Punkte Teil des Hintergrundes. Um dies zu erreichen, muß das Bild unter Umständen auch invertiert werden.
In der **Segmentierung** werden alle Punkte, die zusammen ein Zeichen bilden, zusammengefaßt. Ebenso werden alle Zeichen, die ein Wort bilden, und alle Wörter, die eine Linie bilden, zusammengefaßt. Die Segmentierung legt damit die Lesereihenfolge der Zeichen fest.
Die Bestimmung der **Orientierung** der Zeichen ist Voraussetzung für das richtige Erkennen der Zeichen. In vielen Anwendungen wird von einer festen Orientierung ausgegangen, so daß dieser Schritt entfällt. In anderen Anwendungen wird dieser Schritt übergangen, indem orientierungsunabhängige Verfahren zur Zeichenklassifikation verwendet werden.
Mit der **Zeichenklassifizierung** wird den Zeichenpunkten eine Bedeutung zugeordnet.
Würden alle vorgenannten Schritte perfekt funktionieren, so wäre eine abschließende Fehlerkorrektur überflüssig. In der Praxis treten aber immer verklebte und zerfallene Buchstaben auf. Eine **Nachbearbeitung** der Segmentierung und Binarisierung von nicht erkannten Zeichen mit anschließendem erneuten Versuch der Erkennung ist daher unumgänglich.

Zur Nachbearbeitung gehört auch die Korrektur des Leseergebnisses mittels eines Wörterbuches, falls für eine Anwendung vorhanden. Dieser letzte Schritt ist aber nicht mehr Bestandteil dieser Diplomarbeit.

Obiges Schema ist nicht bindend. Manche Autoren erkennen erst die Zeichen und segmentieren dann diese in Worte und Linien. Die Vorsegmentierung übernimmt manchmal schon das Einteilen in Linien. Diese Linien werden dann in der Segmentierung in Worte und diese wiederum in Zeichen zerlegt.
In den folgenden Unterkapiteln werden die einzelnen Schritte genauer beschrieben, verschiedene Lösungsmöglichkeiten aufgezeigt und deren Vor- und Nachteile erläutert. Aus der Diskussion der einzelnen Schritte ergibt sich dann ein Konzept zur Lösung der gestellten Aufgabe.

2.1 Digitalisierung

In der Projektvorstellung in Kapitel 1.1 wurde die verwendete Apparatur zur Digitalisierung schon angedeutet. Für die genaue Vermessung der Mikrochips und die Schrifterkennung steht nur eine Schwarzweiß-Kamera zur Verfügung. Diese ist über eine Verfahreinheit beliebig über der Leiterplatte in X- und Y-Richtung positionierbar. Der Abstand zwischen Kamera und Leiterplatte wurde so gewählt, daß die größten bekannten Chips noch ganz ins Bild passen, denn sonst könnten sie nicht mehr vermessen werden.
Es wurde eine spezielle Beleuchtung entwickelt: Ein Ring bestückt mit drei Reihen Infrarot-Leuchtdioden. Dieser ist ebenfalls an der Verfahreinheit befestigt und wird mit der Kamera verfahren. Damit in die Kamera auch nur Infrarotlicht einfällt, ist sie mit einem Infrarotfilter ausgestattet. Die Beleuchtung mit Infrarotlicht verhindert, daß unterschiedliche äußere Lichtverhältnisse, bedingt durch Tageszeit und Aufbauort der Apparatur, das Kamerabild beeinflussen. Wird mit normalem Licht gearbeitet ist es notwendig, Bildverarbeitungsanlagen gegen äußere Lichteinflüsse abzuschirmen.
Die Kamera ist so eingestellt, daß das Bild wenige Millimeter über der Leiterplatte scharf ist. Da Chips unterschiedlich hoch sind, sind nur die Bilder von flachen Chips scharf, je höher ein Chip, desto unschärfer wird das Bild. Dies ist nur ein Kompromiß, besser wäre eine Autofokuskamera. Der Wahl des Brennpunktes knapp oberhalb der Leiterplatte liegt die Annahme zugrunde, daß hohe Chips auch größer sind und damit, auf Grund des größeren Platzangebotes, vermutlich auch die Schrift auf ihnen größer ist. Eine geringe Unschärfe ist bei großer Schrift eher akzeptabel, als bei kleiner Schrift. Leider trifft diese Annahme nicht immer zu. Eproms sind z.B. sehr hoch, haben aber eine sehr kleine Schrift, da der größte Teil der Chipoberfläche vom Löschfenster eingenommen wird. Welchen Einfluß die fehlende Schärfe in der Praxis hat wird sich zeigen.
Neben einem Autofokus wäre auch ein elektronisches Zoom-Objektiv wünschenswert. Da zur Schrifterkennung die Größe der Chips bereits bekannt ist, könnte die Bildausschnittsgröße immer an die aktuelle Chipgröße angepaßt werden. Bei sehr kleinen Chips wäre hierdurch eine weitaus

bessere Erkennung zu erwarten. Leider stehen diese zusätzlichen Gerätschaften nicht zur Verfügung.

2.2 Vorverarbeitung

Nachdem Bilder von der Kamera gemacht wurden, wird in einem ersten Schritt versucht, diese zu verbessern und erste Störungen zu unterdrücken. Es ist davon auszugehen, daß die Beleuchtung des Chips gleichmäßig und ausreichend hell ist; die Beleuchtung wurde extra für diese Anlage konstruiert und sollte daher gute Ergebnisse erzielen. Da im Infrarot-Lichtbereich gearbeitet wird, kann die Gleichmäßigkeit der Ausleuchtung auch nicht durch andere Lichtquellen (Tageslicht, Raumbeleuchtung) gestört werden. Auf einen Verarbeitungsschritt zum Ausgleich einer ungleichmäßigen Ausleuchtung des Chips kann daher verzichtet werden. Da nur eine Schwarzweiß-Kamera verwendet wird, sind auch keine Farbtemperatur-abgleiche und ähnliches erforderlich.

Zur Unterdrückung von Bildrauschen und zur Kantenverstärkung werden in der Regel Filter verwendet. Filter können sowohl im Frequenzbereich als auch im Bildbereich angewandt werden. Allgemein läßt sich nicht sagen, daß die eine Methode besser als die andere ist, sondern die Ergebnisse sind immer abhängig von der Anwendung. Soll im Frequenzbereich gearbeitet werden, muß das Bild zuerst fouriertransformiert werden und nach der Filterung zurücktransformiert werden. Da dies zusätzlicher Zeitaufwand ist, sollen zu Beginn nur Filter im Bildbereich getestet werden. Nur falls diese sich als unzureichend erweisen, werden auch Filter im Frequenzbereich getestet. Für eine Beschreibung von Filtern im Frequenzbereich sei auf [GoWo92] verwiesen. Die Berechnung von Filtern im Bildbereich wird im folgenden beschrieben. Für darüber hinausgehende Informationen kann ebenfalls auf [GoWo92] zurückgegriffen werden.
Ein Filter ist eine Funktion, die auf jeden Punkt eines Bildes angewandt wird. Der Wert der Filterfunktion für einen Punkt ist dabei abhängig von den Grauwerten des Punktes selber und der benachbarten Punkte. Es können beliebig viele Nachbarpunkte hinzugezogen werden, oft werden die acht direkt angrenzenden Nachbarn verwendet. Die Filterfunktion hat damit neun Parameter, die verknüpft werden um den neuen Grauwert eines Punktes zu berechnen. Es werden lineare und nicht-lineare Filter unterschieden. Bei linearen Filtern ergibt sich der neue Grauwert eines Punktes aus der Summe der einzeln gewichteten Parameter:

$$g^{neu}_{x,y} = \begin{matrix} w_0 g_{x-1,y-1} & +w_1 g_{x,y-1} & +w_2 g_{x+1,y-1} \\ +w_3 g_{x-1,y} & +w_4 g_{x,y} & +w_5 g_{x+1,y} \\ +w_6 g_{x-1,y+1} & +w_7 g_{x,y+1} & +w_8 g_{x+1,y+1} \end{matrix}$$

Die Summe der Gewichte w_i sollte eins sein, um sicher zu sein, daß der neue Grauwert im Wertebereich für Grauwerte liegt.

Bei nicht-linearen Filtern werden die Parameter anders verknüpft. Das Ergebnis kann z.B. der minimale Grauwert der Parameter sein oder ihr *Median*.

Parallel zur Unterscheidung von linearen und nicht-linearen Filtern können Filter auch nach ihrer Funktion unterschieden werden. Es werden Detailunterdrückungs- und Detailverstärkungsfilter unterschieden.

2.2.1 Detailunterdrückungsfilter

Filter zur Detailunterdrückung werden meistens zur Verringerung des Bildrauschens eingesetzt. Die häufigsten Filter sind Tiefpaß- und Medianfilter.
Der Tiefpaßfilter ist ein linearer Filter bei dem alle Gewichte positiv sind. Der neue Grauwert ist damit ein Durchschnittswert aller benachbarten Grauwerte. Im einfachsten Fall sind alle Gewichte 1/9. Alle Nachbarn werden dann gleich gewichtet. Beliebige andere Gewichte sind möglich, z.B. eine stärkere Gewichtung des mittleren Parameters, gegenüber den umliegenden. Hierdurch wird die Detailunterdrückung abgeschwächt.

$$\mathrm{TP} = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Abbildung 2.1: Tiefpaßfilter

Eine Folge der Tiefpaßfilterung von Bildern ist das Verschwimmen von Kanten und feinen Details. Soll dies vermieden werden und nur Rauschen ausgefiltert werden, so kann ein Medianfilter besser sein. Insbesondere wenn das Rauschen im Bild aus vereinzelten Extremwerten besteht. Der Medianfilter ist ein nicht-linearer Filter. Er ersetzt einen Punkt durch den Median der umliegenden Punkte.

2.2.2 Detailverstärkungsfilter

Detailverstärkungsfilter sind Hochpaßfilter und Ableitungsfilter.
Bei Hochpaßfiltern sind alle umliegenden Gewichte negativ und das Gewicht am Punkt selbst ist positiv. Ist die Summe aller Gewichte null und haben die Punkte innerhalb einer 3x3 Matrix ungefähr alle den gleichen Grauwert, ist der neue Grauwert in der Mitte null. Ist der alte Grauwert in der Mitte größer als die umliegenden, ist der neue Grauwert positiv, ansonsten negativ. Da negative Grauwerte meistens außerhalb des Definitionsbereiches liegen, müssen die Grauwerte neu auf den gültigen Wertebereich transformiert werden. Ist die Summe der Gewichte größer null, wird das hochpaßgefilterte Bild mit dem Ursprungsbild überlagert. Die tiefen Frequenzen bleiben dann

im Bild erhalten. Auf eine Neuskalierung des Wertebereichs kann dann meistens verzichtet werden.

$$HP = \frac{1}{9}\begin{pmatrix} -1 & -1 & -1 \\ -1 & w & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Abbildung 2.2: Hochpaßfilter mit $w = 9A\text{-}1$. Ist $A = 1$, werden tiefe Frequenzen komplett unterdrückt. Mit $A > 1$ wird das Hochpaßbild mit dem Ursprungsbild überlagert.

Ableitungsfilter berechnen den Gradienten an einem Punkt. Der Gradient ist dann der neue Grauwert. Wie bei Hochpaßfiltern kann das Gradientenbild mit dem Ursprungsbild überlagert werden, indem das Gewicht des mittleren Punktes größer null gewählt wird. Für eine Herleitung der Gewichte eines Gradientenfilters sei auf [GoWo92] verwiesen. Gradientenfilter bestehen meistens aus einer horizontalen und einer vertikalen Filtermatrix, deren absoluten Werte addiert werden. Charakteristisch für einen Gradientenfilter ist, daß die Gewichte der äußeren Zeilen oder Spalten entgegengesetzte Vorzeichen haben.

$$Sobel = abs\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} + abs\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

Abbildung 2.3: Sobelfilter(Ableitungsfilter)

Die verschiedenen Filter lassen sich beliebig kombinieren. In der Praxis wird oft zuerst ein Detailunterdrückungsfilter und anschließend noch ein Detailverstärkungsfilter verwendet.
Um den für meine Anwendung besten Filter, bzw. die beste Filterkombination zu finden, werde ich bei der Realisierung mit folgenden Filtern und Filterkombinationen experimentieren:

- ohne Filterung
- Hochpaß
- Tiefpaß
- Hochpaß / Tiefpaß
- Tiefpaß / Hochpaß
- Ableitungsfilter
- Tiefpaß / Ableitungsfilter
- Median
- Median / Hochpaß

2.3 Vorsegmentierung

Bei vielen Anwendungen sind Text und Grafik gemischt in einem Bild enthalten. Nicht-Textkomponenten sollten so früh wie möglich aussortiert werden. Zum Einen wird hierdurch der Verarbeitungsaufwand für die Texterkennung minimal gehalten. Zum Anderen können Nicht-Textkomponenten die weitere Verarbeitung des Textes behindern. Es ist z. B. einfacher ein Bild automatisch zu binarisieren in dem sich nur Zeichenvordergrundpunkte und Zeichenhintergrundpunkte befinden, als wenn sich dort auch noch Grafiken befinden. Die Aufgabe der Vorsegmentierung ist also die frühzeitige Trennung des Textes von anderen Komponenten im Bild.
Einen Großteil dieser Aufgabe wird bereits in der ersten Stufe von Vision II erledigt. Hier wird der Bildausschnitt des Kamerabildes bestimmt, welcher der Chipoberfläche entspricht. Umliegende Leiterbahnen und andere Bauteile werden hierdurch ausgesondert. Auf Mikrochips sind neben Zeichen auch andere Objekte zu finden. Dies sind insbesondere Firmenlogos, Gehäuseeinbuchtungen und unbedruckte Flächen. Sofern möglich sollten diese von den Textzeichen im Rahmen der Vorsegmentierung getrennt werden.
Vorsegmentierungsverfahren werden hauptsächlich in der Dokumentanalyse eingesetzt. In [Hara94] findet sich eine Zusammenstellung von Verfahren, die speziell für die Dokumentanalyse erdacht wurden und für meine Anwendung ungeeignet sind. Die dort erläuterten Verfahren erwarten alle eine große Menge an Zeichen.
Scherl gibt in [Sche87] einen besseren und allgemeineren Überblick über Möglichkeiten der Vorsegmentierung. In Anlehnung an seine Arbeit sollen diese kurz vorgestellt und ihre Vor- und Nachteile diskutiert werden.

Projektionsverfahren

Die *Projektion* faßt die Grauwerte der Vorlage in horizontaler (oder vertikaler) Richtung entlang einer Vorlagenzeile (-spalte) in einem Histogramm zusammen. Dieses Histogramm hat ein charakteristisches Aussehen für Textzeilen, Bilder oder Grafiken. Bei verdrehten Vorlagen, können keine Textzeilen mehr erkannt werden, sondern maximal noch Textblöcke. Das Verfahren versagt, sobald Text, Grafik und Bilder beliebig gemischt auftreten. Vorteilhaft ist die Unempfindlichkeit gegen Störungen.

Filterverfahren

Filterverfahren können Muster mit einem ausgeprägten einheitlichen Erscheinungsbild erkennen und klassifizieren. Filter sind aber immer auf eine Mustergröße und -orientierung zugeschnitten. Bei unbekannter Größe und Orientierung sind sie ungeeignet. Vorteilhaft ist ihr integrierender Effekt, so daß z.B. Linien mit kleinen Löchern noch erkannt werden. Dies kann sich aber auch nachteilig auswirken, wenn nah benachbarte Muster auch integriert und nicht mehr separiert werden. Aufgrund der Nähe von einzelnen Schriftzeichen zueinander, insbesondere auch bei kursiver Schrift können

Filterverfahren in der Regel nur Textzeilen erkennen und nicht einzelne Zeichen.

Blow-Shrink-Verfahren

Diese Verfahren werden in der Literatur auch als Erosions- und Dilatationsverfahren bezeichnet. Durch Ausdehnung der weißen und schwarzen Punkte, soll aus der zerhackten Struktur einer Textzeile ein monolithischer Block entstehen. Die Blöcke können dann mit statistischen Verfahren klassifiziert werden. Blow-Shrink-Verfahren sind den Filterverfahren ähnlich und haben die selben Vor- und Nachteile.

Lauflängenverfahren

Lauflängenverfahren suchen die Zeilen und Spalten nach aufeinanderfolgenden weißen oder schwarzen Punkten ab, deren Anzahl eine vorgegebene Mindestanzahl überschreitet (siehe *Lauflängenkodierung*). Durch eine logische Verknüpfung der horizontalen und vertikalen Lauflängen entstehen voneinander isolierte Blöcke, die noch als Bild-, Grafik- oder Textbereich mit statistischen Verfahren zu klassifizieren sind. Durch die vorzugebende Längenschwelle haben diese Verfahren ähnliche Probleme wie die Filterverfahren.

Analyserechtecke

Bei diesen Verfahren wird das Dokument in gleiche, fest positionierte, sich überlappende Rechtecke eingeteilt. Die einzelnen Rechtecke werden mit statistischen Verfahren unabhängig voneinander klassifiziert. Durch Nachbearbeitung entstehen zusammenhängende Text-, Grafik- und Bildbereiche. Die Rechteckgröße muß zur Zeichengröße passen. Mustergrenzen entsprechen in der Regel nicht den Rechteckgrenzen, so daß Falschklassifizierungen möglich sind. Aufgrund der statistischen Auswertung der Rechtecke sind diese Verfahren recht unempfindlich gegenüber kleinen Störungen.

Die Klassifikation der Bereiche durch statistische Verfahren wie Mittelwert, Streuung und *Momenten* höherer Ordnung auf der Basis von Grauwert-, Lauflängen- oder Gradientenstatistiken hat Vor- und Nachteile. Durch die integrierende Wirkung sind statistische Verfahren sehr unempfindlich gegen Detailstörungen und im allgemeinen auch drehlagenunabhängig. Die für eine statistische Auswertung erforderliche Mindestdatenmenge steht einer Detailanalyse im Wege.

Laut Scherl setzen alle diese Verfahren ein Vorwissen über die zu untersuchende Vorlage voraus. Es wird erwartet, daß die Vorlage und damit auch die Zeichen nicht verdreht vorliegen, daß sich die Größe der Zeichen nur in einem geringen Rahmen bewegt, oder daß eine Mindestmenge von Zeichen vorhanden ist, um eine statistische Auswertung zu ermöglichen.

Beschreibungsverfahren

Scherls Meinung nach sind die einzigen allgemein verwendbaren Verfahren zur Vorsegmentierung und auch zur Segmentierung Beschreibungsverfahren. Sie werden in der Literatur auch als syntaktische oder nichtnumerische Verfahren bezeichnet und klassifizieren ein Muster nicht aufgrund von charakteristischen Merkmalen, sondern zerlegen das Muster in Teilmuster und klassifizieren das Muster aufgrund der Teile und deren Beziehungen untereinander. Für die theoretischen Grundlagen syntaktischer Verfahren sei auf [LiEn89] verwiesen. Als atomares Teil wird der Umriß gleich heller zusammenhängender Punkte verwendet. Der Umriß der einzelnen Teile kann z.B. durch einen Kettenkode beschrieben werden.
Mit Beschreibungsverfahren können Muster beliebiger Größe, Lage und Orientierung segmentiert und klassifiziert werden. Werden Beschreibungsverfahren verwendet, so kann auf eine Vorsegmentierung in der Regel verzichtet werden. Durch die Segmentierung mittels Beschreibungsverfahren wird die Aufgabe der Vorsegmentierung, das Trennen von Text- und Nicht-Textbereichen, vollzogen. Denn bei Verwendung von Beschreibungsverfahren werden durch die Segmentierung all die Bereiche als Text klassifiziert, in denen Objekte gefunden werden, die sich zu Worten und Linien zusammensetzen lassen. Bei Bedarf können die restlichen Bereiche noch weiter unterschieden werden.
Ein Nachteil von Beschreibungsverfahren ist ihre Empfindlichkeit gegenüber Störungen. Ein zusammenhängendes Muster kann in Teile zerfallen oder mehrere Muster können verkleben. Diese Probleme können durch eine angepaßte Beschreibung und Auswertung behoben bzw. gemildert werden. Insbesondere verkleben Muster bei Beschreibungsverfahren nur dann, wenn sie in der Vorlage bereits verklebt sind. Bei Filterverfahren verkleben aufgrund des integrierenden Charakters auch Muster, die in der Vorlage noch separiert sind.

Wegen des zeichenähnlichen Erscheinungsbildes von Firmenlogos und Gehäuseeinbuchtungen erscheint es schwierig diese bereits im Rahmen der Vorsegmentierung auszusondern. Aufgrund des geringen Vorkommens solcher Objekte ist es vertretbar diese erst im Rahmen der Segmentierung auszusondern. Auf eine Vorsegmentierung wird daher verzichtet.

2.4 Binarisierung

Aufgabe der Binarisierung ist es Textpunkte vom Hintergrund zu trennen. Dazu wird aus dem vorverarbeiteten Bild ein neues Bild erzeugt, in dem alle Vordergrundpunkte eines Zeichen schwarz sind und alle anderen Punkte weiß.
Ganz allgemein ist diese Aufgabe nicht zu lösen, sondern es muß eine Annahme getroffen werden. Es wird davon ausgegangen, daß es einen Helligkeitswert S gibt für den gilt, daß alle Vordergrundpunkte eine Helligkeit kleiner oder gleich S haben und daß alle Hintergrundpunkt eine Helligkeit größer S haben. S wird als Schwellwert bezeichnet. Nun reicht es aus, S zu

bestimmen und die Binarisierung ist trivial. Bei manchen Anwendungen kann der gleiche Schwellwert für alle Bilder verwendet werden. S kann dann einmal manuell festgelegt werden. Aufgrund der unterschiedlichen Schrift- und Chipfarben muß für meine Anwendung ein Schwellwert für jedes Bild automatisch ermittelt werden. Das eigentliche Problem bei der Binarisierung ist daher die Bestimmung von S.
Nicht auf allen Chips haben die Vordergrundpunkte eine kleinere Helligkeit als die Hintergrundpunkte. Dies kann auch genau invers sein. Das Bild muß dann vor der Binarisierung invertiert werden. Das Problem hierbei ist es zu erkennen, ob eine Invertierung nötig ist oder nicht.
Verfahren zur Bestimmung von S heißen Schwellwertbestimmungsverfahren. In Abschnitt 2.4.1 werden drei Verfahren vorgestellt. Sie gehen davon aus, daß die Schrift und der Hintergrund jeweils einfarbig sind. Ein Verfahren, daß bei mehrfarbigen Bildern verwendet werden kann, ist die Pixelaggregation. Dieses Verfahren wird in Abschnitt 2.4.2 beschrieben. Mit Fragen der Invertierung beschäftigt sich Abschnitt 2.4.3.

2.4.1 Schwellwertbestimmungsverfahren

Drei in der Literatur weit verbreitete Verfahren werden im folgenden vorgestellt.

2.4.1.1 Rekursives Binarisierungsverfahren

Diesem Verfahren liegt die Idee zugrunde, den Schwellwert in die Mitte zwischen der mittleren Vordergrund- und Hintergrundhelligkeit zu plazieren. Da diese auch nicht bekannt sind, wird iterativ[2] vorgegangen. Es werden zuerst der größte und der kleinste vorkommende Helligkeitswert bestimmt. Der Schwellwert wird zu Beginn auf den mittleren Helligkeitswert zwischen diesen Extremwerten gesetzt. Für diesen Schwellwert, werden die mittleren Vordergrund- und Hintergrundhelligkeiten bestimmt. Anschließend kann ein neuer Schwellwert berechnet werden, der genau in der Mitte der beiden Helligkeitswerte liegt. Für diesen neuen Schwellwert wird das Verfahren wiederholt. Dies wird solange getan, bis der neu berechnete Schwellwert vom vorhergehenden nur um eine minimale Spanne abweicht. Das Verfahren konvergiert laut Klette und Zamperoni [KlZa95] nach einigen Iterationen und erzielt für viele Anwendungen eine gute Trennung zwischen Objekt und Hintergrund, insbesondere dann, wenn Objekt und Hintergrund ungefähr die gleiche Bildfläche bedecken.

2.4.1.2 Binarisierungsverfahren nach der Diskriminanzanalyse

Jeder Schwellwert S bestimmt eine Gruppe von Vordergrund- und Hintergrundpunkten. Für diese beiden Gruppen kann dann der Erwartungswert und die Streuung der Helligkeit in der Gruppe bestimmt

[2] Mir ist nicht bekannt, warum das Verfahren rekursiv heißt, obwohl es rein iterativ ist.

werden. Otsu schlug in [Otsu79] vor, den Schwellwert als ideal zu betrachten, für den die gewichtete Summe der Varianzen der beiden Gruppen minimal ist. Als Gewichte werden die Wahrscheinlichkeiten der Gruppen verwendet. Die Wahrscheinlichkeit einer Gruppe ist die relative Auftrittshäufigkeit der Grauwerte der Gruppe im Bild. Sind die Anteile von Vordergrund- und Hintergrundpunkten stark unterschiedlich, führt diese Methode nicht notwendigerweise zum besten Ergebnis (laut [HaSh92]).

2.4.1.3 Binarisierung nach der Kullback-Distanz

Dieses Verfahren stammt von Kittler und Illingworth. Ihm liegt die Annahme zugrunde, daß die Helligkeitsverteilung des Bildes die Superposition von zwei Gaußverteilungen ist. Die Helligkeitsverteilung der Vordergrund- und Hintergrundpunkte ergibt sich aus dem Histogramm in Abhängigkeit von einem angenommenen Schwellwert. Aus den Helligkeitsverteilungen können Erwartungswerte und Varianzen derselben bestimmt werden. Durch diese Erwartungswerte und Varianzen sind Gaußkurven charakterisiert. Es wird der Schwellwert gewählt bei dem die Ähnlichkeit zwischen den superpositionierten Gaußkurven und der Helligkeitsverteilung des Bildes am geringsten ist. Die Ähnlichkeit wird mittels der Kullback-Distanz gemessen.

Für eine ausführlichere Beschreibung des ersten Verfahren sei auf [KlZa95] verwiesen. Die anderen beiden Verfahren werden in [HaSh92] beschrieben.
Neben diesen Verfahren gibt es noch viele andere. Kein Verfahren ist jedoch ideal und es muß für jeden Anwendungsfall ausprobiert werden, welches Verfahren die besten Ergebnisse erzielt. Ich werde daher mit allen drei Verfahren experimentieren.

2.4.2 Pixelaggregation

Die Pixelaggregation führt keine Binarisierung durch, sondern bestimmt Mengen von zusammenhängenden, ungefähr gleich hellen Punkten. Dazu wird zuerst ein Punkt gesucht, dessen acht direkte Nachbarn den gleichen Helligkeitswert plus, minus einer anzugebenden Toleranz haben. Wurde solch ein Punkt gefunden, wird er und alle seine Nachbarn als zusammengehörend betrachtet. Jeder an diese Punkte angrenzende Punkt wird dann untersucht, ob er ebenfalls zu den Ausgangspunkten gehört. Dies ist der Fall, wenn die Helligkeitswerte aller zusammengehörenden Punkte und des angrenzenden Punktes eine anzugebende Schwankungsbreite nicht übersteigt und wenn die Helligkeit des angrenzenden Punktes weniger als eine anzugebende Differenz von der Helligkeit der umliegenden zugehörigen Punkte abweicht. Wird ein Punkt in die Menge der zusammengehörenden Punkte aufgenommen, so werden auch die an ihn angrenzenden Punkte untersucht, ob sie obige Kriterien erfüllen und in die gleiche Menge gehören. Der Nachteil an diesem Verfahren ist, daß drei Grenzwerte anzugeben sind. Die Wahl dieser Grenzwerte entscheidet über die Qualität der erzielten Ergebnisse. Leider sind keine Verfahren bekannt, mit denen diese Grenzwerte automatisch bestimmt werden können. Die Werte müssen daher

von Hand eingestellt und für alle Bilder verwendet werden. Ob Werte existieren, die für alle Bilder in meiner Anwendung brauchbar sind, können erst praktische Versuche zeigen.

2.4.3 Invertierung

Nach der Binarisierung sollen Vordergrundpunkte schwarz und Hintergrundpunkte weiß sein. Hat das Ausgangsbild weiße Schrift auf schwarzem Hintergrund, muß das Bild invertiert werden. Ich werde davon ausgehen, daß dieser Fall vorliegt, wenn der ermittelte Schwellwert heller ist, als der am häufigsten vorkommende Helligkeitswert.
Diesem Kriterium liegen die Annahmen zugrunde, daß auf einem Chip mehr Hintergrundpunkte vorkommen als Vordergrundpunkte, daß die Verteilung der Vorder- und Hintergrundhelligkeiten jeweils gaußförmig ist und daß sich die Gaußkurven nur geringfügig überlappen. Der häufigste Helligkeitswert ist damit der Erwartungswert der größeren Gaußkurve. Die größere Gaußkurve entspricht der häufigeren Punktmenge und damit, laut Annahme, den Hintergrundpunkten.
Die Annahmen gaußförmiger Verteilungen und geringer Überlappungen derselben sind realistisch, da die der Literatur entnommenen Schwellwert-bestimmungsverfahren von ähnlichen Vorraussetzungen ausgehen.

Wird die Pixelaggregation verwendet, so ist der erste Schritt der Segmentierung bereits geschehen, das Finden von zusammengehörenden Punkten. Wurde jedoch ein Binarisierungsverfahren verwendet, so wird im folgenden Kapitel erklärt wie daraus Mengen von zusammengehörenden Punkten bestimmt werden können.

2.5 Segmentierung

Durch die Binarisierung wurde bereits eine wichtige Gruppierung der Bildpunkte in Zeichenpunkte und Hintergrundpunkte erreicht. Die Segmentierung gruppiert die Punkte nun weiter. In einem ersten Schritt werden Zeichenpunkte zusammengefaßt, die zu einem Zeichen gehören. Anschließend können diese Zeichen zu Worten und Linien gruppiert werden. Diese Gruppierung gibt dann Aufschluß über die Achsenorientierung. Eine dritte Aufgabe der Segmentierung ist das Aussortieren von nicht Textelementen, die sich noch im Bild befinden und nicht durch die Vorsegmentierung entfernt wurden. Dies können je nach Anwendung z.B. Grafiken, Diagramme, Photos und Störungen sein. Speziell bei meiner Anwendung sind dies Firmenlogos, Gehäuseeinbuchtungen und natürlich Störungen.
Für die Segmentierung können die gleichen Verfahren verwendet werden, wie für die Vorsegmentierung. Für einen Überblick über die Verfahren und ihre Vor- und Nachteile sei daher auf das Kapitel 2.3 verwiesen.
Bei allen mir bekannten Anwendungen mit Zeichen beliebiger Lage, Größe und Orientierung werden Beschreibungsverfahren zur Segmentierung

verwendet (siehe z.B. [Sche87] und [CCLT94]). Ich werde daher ebenfalls ein solches verwenden.

2.5.1 Punkte zu Zeichenhypothesen

Der erste Schritt der Segmentierung ist die Bildung von Zeichenhypothesen. Zusammenhängende Punkte werden zu einer Punktmenge zusammengefaßt. Für die nachfolgenden Schritte wird prinzipiell angenommen, daß jede Menge von zusammenhängenden Punkten einem Zeichen entspricht. Eine solche Menge von Punkten wird daher im folgenden als Zeichenhypothese bezeichnet. Es muß aber berücksichtigt werden, daß Zeichen auch aus mehreren Teilen bestehen können oder zerfallen sein können oder daß mehrere Zeichen verklebt sind. Eine Zeichenhypothese kann daher auch nur ein Teil eines Zeichens sein oder mehrere Zeichen enthalten.
Wann Punkte zusammenhängen und wie die Punktmengen berechnet werden, wird im Abschnitt 2.5.1.1 erklärt. Da die Weiterverarbeitung von Punktmengen umständlich ist, wird in Abschnitt 2.5.1.2 beschrieben welche Merkmale der Punktmengen geeignet sind um diese für die weitere Segmentierung zu charakterisieren.

2.5.1.1 Zusammenfassen von Punkten

Der erste Schritt der Bildung einer Zeichenhypothese ist das Zusammenfassen von zusammenhängenden Zeichenpunkten. Zwei Punkte hängen zusammen, wenn sie benachbart sind, oder wenn der erste Punkt benachbart ist zu einem dritten Punkt, der zusammenhängt mit dem zweiten Punkt. Nachbarschaft läßt sich definieren als horizontal oder vertikal angrenzend zum betrachteten Punkt (vier Nachbarn) oder zusätzlich auch diagonal (acht Nachbarn). Ich werde Nachbarschaft mit nur vier Nachbarn definieren. Dieser Definition liegt die Annahme zugrunde, daß die Linien eines Zeichens immer mindestens zwei Punkte breit sind. Ist dies der Fall, so sind die gefundenen Zeichenmengen bei vier und bei acht Nachbarn identisch. Ist dies nicht der Fall, so vermute ich, daß eine Erkennung der Zeichen unmöglich ist, da sie viel zu klein sind. Anderseits können zwei benachbarte Zeichenmengen auf Grund von Störungen leichter bei Achternachbarschaft als bei Vierernachbarschaft verkleben. Desweiteren ist das Zusammensetzen von zerfallenen Zeichen einfacher, als das Trennen von Verklebten. Da ich mit sehr gestörten Zeichen und geringen Zeichenabständen rechne, ziehe ich die Vierernachbarschaft vor.
Zusammenhängende Punkte lassen sich einfach bestimmen. Sie können in Form eines Kettenkodes platzsparend gespeichert werden. Die Speicherung als Kettenkode ermöglicht auch die effiziente und anschauliche Darstellung der gefundenen Zeichenmengen. Dies ist besonders wichtig zum Testen der verschiedenen Filter und Binarisierungsverfahren. Es gab auch Versuche die Kettenkodierung zur Zeichenerkennung zu verwenden. Die Ergebnisse waren aber unzureichend (vergl. [HöDe90]).

2.5.1.2 Abstraktion von der genauen Punktmenge

Im folgenden müssen Zeichenhypothesen zu Worthypothesen und diese zu Linienhypothesen zusammengefaßt werden. Das Arbeiten mit der exakten Menge von zusammenhängenden Punkten wäre zu umständlich. Deshalb wird mit einem alle Punkte umschließenden Rechteck oder Kreis gearbeitet, oder nur mit dem Mittelpunkt der Punktmenge. Die Seiten des Rechtecks können entweder parallel zum Bildrand sein, oder so gelegt werden, daß die Fläche des umschließenden Rechtecks minimal wird. Letzteres lohnt sich nur, wenn die Zeichen beliebig verdreht sein können; ist aber auch dann nicht unbedingt erforderlich (vergl. [Sche87] und [CCLT94]). Die Verwendung eines Kreises oder des Schwerpunktes ist zwar eine Drehwinkel unabhängige Darstellung, aber weniger aussagekräftig. Die meisten Zeichen sind nicht rund, sondern eher rechteckig und besser durch eine Breiten- und Höhenangabe charakterisiert als durch einen Radius, auch wenn Breiten- und Höhenangabe aufgrund einer Verdrehung verfälscht sind.
Da davon ausgegangen wird, daß die Beschriftung auf einem Chip immer parallel zum Rand erfolgt, ist die einfachste Lösung gleichzeitig auch die Beste: Randparallele Rechtecke sind einfach zu berechnen und beschreiben die Zeichen am Genauesten. Ich werde daher mit diesen arbeiten.

Im folgenden wird öfters von der Höhe und der Breite einer Hypothese die Rede sein. Damit ist dann immer die Höhe und Breite des umschließenden Rechtecks gemeint. Auch Abstand und Überlappung zweier Hypothesen bezieht sich immer auf das umschließende Rechteck der Hypothesen.

2.5.2 Zeichenhypothesen zu Worthypothesen

Die Gruppierung von Zeichenhypothesen zu einer Worthypothese erfolgt aufgrund des Abstandes der Zeichenhypothesen zueinander, der Überlappung der Zeichenseiten und der Zeichenhöhen. Die Verwendung der Breite in irgendeiner Form ist nicht möglich, da durch das Verkleben von Zeichen beliebige Breiten entstehen können. Abstand und Überlappung von Zeichenhypothesen müssen immer relativ zur Höhe betrachtet werden, um größenunabhängige Grenzwerte festlegen zu können. Einen solchen Ansatz verwenden Scherl und Consorti in [Sche87] und [CCLT94].
Einen anderen Ansatz verfolgen Fletcher und Kasturi in [FlKa88]. Sie bestimmen zuerst alle Zeichenhypothesen deren Mittelpunkte auf einer Geraden liegen. Hierzu verwenden sie die *Hough Transformation*. Aufgrund des Abstandes und der Höhe werden Zeichenhypothesen, die auf einer Geraden liegen, dann zu Worten zusammengefaßt. Das Verfahren ist im Kapitel 2.6.1 „Achsenorientierung“ genauer beschrieben und auch seine Nachteile gegenüber dem ersten Verfahren dargelegt.
Ich werde die Einteilung der Zeichenhypothesen in Worthypothesen nach dem Verfahren von Scherl durchführen und auch seine Grenzwerte verwenden. Inwieweit eine Anpassung der Grenzwerte und des Verfahrens an mein Anwendungsgebiet erforderlich ist, muß dann die Praxis zeigen.

Nach Scherl gehört ein Zeichen zu einem horizontalen Wort, wenn es die folgenden drei Bedingungen erfüllt:

1. Höhenschwankung: $0{,}55 < h_z / h_w < 1{,}8$
2. relativer Abstand: $\min(x_{1z}-x_{2w}, x_{1w}-x_{2z})/ \max(h_z, h_w) < 0{,}5$
3. relative vertikale Überlappung: $0{,}35 < \min(y_{1z}-y_{2w}, y_{1w}-y_{2z}) /\max(h_z, h_w) < 1{,}0$

Zeichen und Wort sind durch das umschließende Rechteck bestimmt. Die linke obere Ecke des Rechtecks hat die Koordinaten (x_1,y_1), die rechte untere Ecke die Koordinaten (x_1,y_2). Es gilt $x_2 \geq x_1$ und $y_2 \geq y_1$. Die Höhe h ist $h = y_2-y_1+1$. Der Index z bezeichnet die Zeichenhypothese, w die Worthypothese.
Die Formeln sind nur für horizontale Worthypothesen angegeben. Durch Vertauschen der Indizes x und y erhält man die Formeln für vertikale Worthypothesen.
Meistens wird es möglich sein eine Zeichenhypothese sowohl einer horizontalen als auch einer vertikalen Worthypothese zuzuordnen. Da an dieser Stelle noch nicht gesagt werden kann, welche Zuordnung richtig ist, werden Worthypothesen für beide Möglichkeiten erstellt. Die Entscheidung, welche Zuordnung richtig ist, wird bei der Bestimmung der Achsenorientierung (Abschnitt 2.6) getroffen. Auch die nachfolgenden Linienhypothesen werden einmal für horizontale und einmal für vertikale Linien aus entsprechenden Worthypothesen erstellt.

2.5.3 Worthypothesen zu Linienhypothesen

Nachdem Worthypothesen erzeugt wurden, können Linienhypothesen generiert werden. Die Formeln sind hier ähnlich, erlauben nur einen größeren Abstand, eine geringere Differenz der Höhen und erwarten eine größere Überlappung:

1. Höhenschwankung: $0{,}77 < h_l / h_w < 1{,}3$
2. Abstand: $\min(x_{1l}-x_{2w},x_{1w}-x_{2l})/ \max(h_l, h_w) < 2{,}5$
3. relative vertikale Überlappung: $0{,}5 < \min(y_{1z}-y_{2w}, y_{1w}-y_{2z}) /\max(h_z, h_w) < 1{,}0$

Der Index l bezeichnet eine Linie.

2.5.4 Entfernen von Störungen, Logos und Gehäuseeinbuchtungen

Firmenlogos und Gehäuseeinbuchtungen sind anwendungsabhängige Probleme. Es gibt daher keine Literatur zu deren Behandlung. Störungen treten in jeder Anwendung auf und werden durch eine Vorverarbeitung reduziert. Störungen, die auch nach der Vorverarbeitung noch vorhanden sind, können auch nur durch anwendungsabhängige Maßnahmen erkannt und entfernt werden. Auch hierzu findet sich daher keine Literatur.
Wie schon im Kapitel über die Vorsegmentierung angesprochen, sind Firmenlogos und auch Gehäuseeinbuchtungen in ihrer Form Zeichen recht ähnlich. Sie sollen daher erst durch die Zeichenklassifizierung erkannt und

ausgesondert werden. Ist die Aufschrift heller und sind die Gehäuseeinbuchtungen dunkler als die Chipoberfläche, sind im binarisierten Bild die Einbuchtungen zu Hintergrundpunkten geworden. Die Einbuchtungen werden nicht als eigene Objekte gefunden und stören nicht. Nur in den seltenen Fällen mit dunklerer Aufschrift als Oberfläche werden Gehäuseeinbuchtungen gefunden und zum Problem. Für diese seltenen Fälle müssen sie durch die Zeichenerkennung bearbeitet werden.

Kleine Störungen sollen anhand ihrer Größe erkannt werden. Es sollte möglich sein eine minimale Größe für ein erkennbares Zeichen empirisch zu ermitteln. Alle kleineren Zeichenhypothesen können dann vor der Bildung der Worthypothesen gelöscht werden. Größere Störungen können erst nach dem fehlgeschlagenen Versuch der Klassifizierung entfernt werden.

Nach der Segmentierung sind zwei gegensätzliche Systeme von Linienhypothesen entstanden: Zum einen ein System aus horizontalen Linienhypothesen, bestehend aus horizontalen Wörtern, zum anderen ein System aus vertikalen Linienhypothesen bestehend aus vertikalen Wörtern. Da beide Systeme auf den gleichen Zeichenhypothesen beruhen, kann nur eines der beiden Systeme richtig sein. Welche Kriterien zur Entscheidung herangezogen werden, beschreibt das folgende Kapitel.

2.6 Bestimmung der Orientierung

Auf Mikrochips kann die Beschriftung prinzipiell in vier verschiedenen Richtungen vorkommen. Es stellt sich daher die Frage, wie kann die Orientierung bestimmt werden.
Bei den meisten Anwendungen mit Schrifterkennung ist die Orientierung vorgegeben. Texte werden immer entsprechend der Leserichtung verarbeitet. Formulare die automatisch weiterverarbeitet werden sollen, haben Markierungen, die Aufschluß über die Orientierung geben.
Erst bei Anwendungen mit beschrifteten Grafiken, wie Konstruktionszeichnungen, Bauplänen etc. in denen Beschriftungen in verschiedenen oft beliebigen Richtungen und Winkeln auftreten, stellt sich das Problem der Ermittlung der Orientierung. Das Problem wird dabei in der Regel in zwei Schritten gelöst. In einem ersten Schritt wird eine Linie AB ermittelt auf der die Zeichen liegen, z.B. eine horizontale oder vertikale Linie. Im zweiten Schritt wird dann bestimmt, ob die Schreibrichtung entlang der Linie von A nach B oder von B nach A ist, also ob z. B. von links nach rechts oder von rechts nach links beschriftet wurde. Zur besseren Unterscheidung wird in dieser Arbeit der erste Schritt als Bestimmung der Achsenorientierung bezeichnet, der Zweite als Bestimmung der Richtungsorientierung.

2.6.1 Achsenorientierung

Zum Bestimmen der Achsenorientierung werden in der Praxis zwei Verfahren verwendet.

Bei dem einen Verfahren werden Zeichen aufgrund von Nachbarschaft und Ähnlichkeit (in der Regel: Abstand, Überlappung und Höhe) als zu einem Wort und zu einer Zeile gehörend klassifiziert. Aus dem Verlauf der Schwerpunkte der Zeichen einer Zeile, kann dann die Linie bestimmt werden auf der die Zeichen liegen. Auf die Verwendung der Breite wird immer verzichtet, da aufgrund von verklebten Zeichen diese beliebig schwankt. Dieses Verfahren wurde erfolgreich von Scherl und Consorti verwendet und ist in [Sche87] und [CCLT94] beschrieben.
Beim zweiten Verfahren wird die *Hough Transformation* auf die Schwerpunkte aller Zeichen angewandt. Zeichen, die auf der selben Linie liegen, werden durch die Hough Transformation auf Linien abgebildet, die sich alle im selben Punkt schneiden. Dieser Schnittpunkt bestimmt die Parameter der Linie auf der die Zeichen liegen. Zeichen liegen leider nie ganz genau auf einer Linie, sondern alleine aufgrund der unterschiedlichen Zeichenhöhen von Groß- und Kleinbuchstaben müssen Abweichungen bis zu einem gewissen Grad toleriert werden.
Die entstehenden Linien müssen aber nicht notwendigerweise richtig sein. Sind viele Zeichen verklebt, so daß eine Zeile nur noch aus zwei oder drei verklebten Zeichen besteht, können mehr Schwerpunkte von Zeichen aus verschiedenen Zeilen auf einer Linie liegen, als von Zeichen in einer Linie. Dies ist insbesondere zu erwarten, da auf Mikrochips mehrzeilige Beschriftungen mit sehr kleinen Zeichenabständen häufig sind. Bei anderen Anwendungen ist die Bildung falscher Zeilen unwahrscheinlicher. Ich vermute daher, daß mit der Hough Transformation nicht immer eine korrekte Bestimmung der Orientierung möglich ist. Ein weiterer Nachteil dieses Verfahrens ist, daß nach dem Finden der Linie trotzdem noch die Zeichen untersucht werden müssen um sie in Worte einzuteilen und ihre Reihenfolge festzulegen.
Für eine ausführlichere Beschreibung dieses Verfahrens sei auf die Arbeit von Fletcher und Kasturi verwiesen [FlKa88].

Aufgrund des höheren Rechenaufwandes und der fraglichen Anwendbarkeit beim zweiten Verfahren, habe ich mich für ein Vorgehen nach dem ersten Verfahren entschieden.
Baupläne, beschriftete Grafiken und ähnliches enthalten nur wenige Textzeilen. Diese sind meist räumlich weit voneinander entfernt, da der Text nicht in mehreren Zeilen untereinander angeordnet ist, sondern über das gesamte Dokument in der Nähe der grafischen Dokumentelemente verstreut ist. Die Zuteilung von Zeichen zu einem Wort ist daher meist eindeutig durch die Abstands- und Höhenregeln gegeben. Scherl und Consorti haben daher kein Kriterium entwickelt, um aus gegensätzlichen Zuordnungen die richtige zu bestimmen.
Auf Mikrochips wird ein solches Kriterium benötigt. Der Text ist in mehreren kurzen Linien untereinander angeordnet. Es kann davon ausgegangen werden, daß die Beschriftung parallel zum Rand und entweder horizontal oder vertikal ist. Die Zuteilung aufgrund der Nachbarschaft ist jedoch nicht eindeutig. Während der Segmentierung wird daher einmal ein System aus horizontalen Hypothesen und ein System aus vertikalen Hypothesen erstellt.

Es muß entschieden werden welches richtig ist. Es wird das System als richtig akzeptiert, welches wahrscheinlicher erscheint. Die Wahrscheinlichkeit eines Systems soll ermittelt werden aufgrund der Anzahl der Linien, der Anzahl der Wörter pro Linie, der Anzahl der Zeichen pro Wort, der Gleichmäßigkeit der Höhen und Abstände der Zeichen in einem Wort und der Wörter in einer Linie. Ich habe mir folgende Formeln zur Berechnung von Wahrscheinlichkeiten von Wörtern, Linien und Liniensystemen überlegt:

```
P(Wort) =   (1 - (Varianz(Zeichenhöhen) / 10)
            * (1 - (Varianz(Zeichenabstände) /  10)
            * g(Zeichenanzahl)
mit g(Zeichenanzahl) =1.0     ; Zeichenanzahl >= 4
                       0.95   ; Zeichenanzahl = 3
                       0.8    ; Zeichenanzahl = 2
                       0.7    ; sonst

P(Linie) =   (1 - (Varianz(Worthöhen) /  10 )
            * (1 - (Varianz(Wortabstände) / 10)
            * Mittelwert(Wortwahrscheinlichkeiten)
            * f(Wortanzahl)
mit f(Wortanzahl) = 1.0  ; Wortanzahl >=2
                    0.9  ; sonst

P(Liniensystem) =  (1 - (Varianz(Linienabstände) /  10)
                   * Mittelwert((Linienwahrscheinlichkeiten)
```

Das Liniensystem mit der größeren Wahrscheinlichkeit wird als richtig angenommen und demzufolge auch die Orientierung dieses Liniensystems.
Sollte die Annahme, daß Chips nur entweder horizontal oder vertikal bedruckt sind falsch sein und auch andere Winkel vorkommen, so wird das obige Vorgehen ineffizient. Es ist dann empfehlenswerter die Hough-Transformation zur Bestimmung der Orientierung zu verwenden.

2.6.2 Richtungsorientierung

Wurde die Achsenorientierung ermittelt, muß nun die Richtungsorientierung bestimmt werden. Für diese Aufgabe habe ich in der Literatur leider keine Lösung gefunden. Auch die bereits oben zitierten Autoren geben keinen Hinweis darauf, wie sie dieses Problem gelöst haben, obwohl es bei ihnen mit Sicherheit auftritt. Einige Autoren geben zwar an rotationsunabhängige Verfahren zur Zeichenerkennung zu verwenden, so daß vordergründig das Problem damit gelöst ist. Praktisch kann ich darin aber keine Lösung des Problems sehen, denn für ein wirkliches Erkennen der Schrift muß auch die Schreibrichtung exakt ermittelt werden. Es kann nicht ausreichen zu ermitteln, daß die Ziffern 1, 2, 3, 4 und 5 in dieser Reihenfolge auftreten. Es muß bestimmt werden, ob dort die Zahl 12.345 oder 54.321 steht. Ich vermute, daß alle Autoren für dieses Problem die trivial Lösung verwenden: beide

Richtungen ausprobieren und die Richtung als richtig annehmen, bei der mehr Zeichen erkannt werden. Diese Methode werde ich daher verwenden.
Theoretisch wären vielleicht noch andere Methoden denkbar, z.B. stehen Großbuchstaben immer am Anfang eines Wortes und nicht am Ende, sofern das Wort nicht nur aus Großbuchstaben besteht. Des weiteren stehen Satzzeichen in der Regel am Ende eines Wortes. Diese Regeln erfordern jedoch immer spezielle Zeichen um die Orientierung zu erkennen und diese sind nicht immer vorhanden. Insbesondere sind Mikrochips nur mit Großbuchstaben und ohne Satzzeichen beschriftet.
Für ein flexibeles Erkennungssystem und insbesondere zur Lesung der Beschriftung von Mikrochips sind diese theoretischen Möglichkeiten daher ohne praktische Bedeutung.
Ich möchte nicht unerwähnt lassen, daß auch die triviale Lösung, beide Lesemöglichkeiten auszuprobieren, nicht immer zu einem Ergebnis führen muß. Bei der folgenden Zeichenfolge ist z.B. ohne weitere Kontextinformation nicht entscheidbar, ob sie von links nach rechts oder von rechts nach links gelesen werden muß:

009966II

Es ist jedoch zu erwarten, daß solche absolut zweideutigen Zeichenfolgen in der Praxis kaum auftreten. Die Ermittlung der Orientierung wird jedoch erschwert, da viele Zeichen aufgrund ihrer Symmetrie zur Orientierungsbestimmung unbrauchbar sind und eine Aussage über die Orientierung eines Wortes nur gemacht werden kann, wenn mindestens ein unsymmetrisches Zeichen enthalten ist und erkannt wird.

2.7 Zeichenklassifizierung

Durch die Vorverarbeitung wurde ermittelt, wo sich Zeichen befinden und welche Abmessungen sie haben. Auch wurden die Zeichen bereits zueinander in Beziehung gesetzt, so daß Wort- und Zeilenzugehörigkeiten bereits bekannt sind.
Als letzten Schritt in einem Zeichenerkennungssystem muß die Klassifizierung erfolgen, die Zuordnung jeder Zeichenhypothese zu einer Bedeutungsklasse.
Die bisherigen Schritte waren durch die Anwendung geprägt: Das Lesen der Beschriftung von Mikrochips. Die Zeichenklassifizierung ist, aufgrund der vorhergehenden Schritte, zu einem anwendungsunabhängigen Problem geworden: Der Zuordnung einer Bedeutungsklasse zu einer rechteckigen Punktmenge. Gleichzeitig ist die Zeichenklassifizierung ein Problem, dessen Komplexität an die Komplexität aller anderen Schritte zusammen heranreicht. Der Zeichenklassifizierung soll und kann daher nur weniger Aufmerksamkeit gewidmet werden, als dies zur idealen Lösung der Aufgabe erforderlich wäre. Im Rahmen dieser Arbeit muß ein einfaches Klassifizierungsverfahren genügen, mit dem sich der Gesamtansatz und insbesondere die anwendungsspezifischen Schritte empirisch verifizieren lassen.

Mögliche Ansätze zur Zeichenklassifizierung werden im folgenden vorgestellt und diskutiert. Abschließend wird dann ein passender Ansatz ausgewählt.

Die Klassifizierung erfolgt in zwei Schritten: Als erstes müssen aus den Helligkeitswerten in der Zeichenregion Merkmale gewonnen werden. In einem zweiten Schritt dienen diese Merkmale als Eingabe für ein Klassifizierungsverfahren, welches die Zuordnung zu einer Bedeutungsklasse vornimmt.

2.7.1 Merkmale

Im Laufe der Zeit wurden viele Merkmale zur Zeichenerkennung vorgeschlagen, und es kommen immer wieder neue hinzu.
Merkmale sollten prinzipiell folgende Kriterien erfüllen (nach [SiPö92]):

- Die Beschreibung des Zeichens durch die Merkmale sollte kompakter sein, als die Beschreibung durch die Helligkeitswerte. Der Werteraum sollte also kleiner sein.
- Durch die Merkmale sollte Wesentliches hervorgehoben und Unwesentliches entfernt werden. Für die Zeichenerkennung gelten Größe, Helligkeit, Verdrehung, Strichstärke, etc. als unwesentlich. Auch Störungen durch Rauschen und Kratzer sind unwesentlich und sollten unterdrückt werden. Was wesentlich an einem Zeichen ist, ist leider nicht genau bekannt, sonst würde es nicht immer wieder neue Vorschläge für Merkmale von Zeichen geben. Einige dieser Vorschläge werden im folgenden noch erläutert.
- Die Merkmale sollten in einer anwendungsunabhängigen Form darstellbar sein, z.B. durch einen Ganzzahlenvektor. Hierdurch kann ein Klassifikationsverfahren für verschiedene Anwendungen verwendet werden, bzw. Klassifikationsverfahren anderer Anwendungen oder Standardklassifikatoren können Verwendung finden.

Mori und Yamamoto geben in [MoSY92] einen Überblick, über die Entwicklung der Zeichenerkennung von ihren Anfängen bis heute. Ihr Fazit ist, daß bisher noch kein mathematisches Modell gefunden wurde, welches die Form eines Zeichens beschreibt. Alle bekannten Methoden sind bisher mehr oder weniger eine Ansammlung von empirischem Wissen. Eine Methode, die als die beste Methode für alle Anwendungen betrachtet werden kann, gibt es nicht. Ihrer Meinung nach werden Systeme, die sehr gute Erkennungsraten erfordern, mehrere Methoden kombinieren.
Sie teilen die bisher erfundenen Methoden in zwei große Gruppen ein: Musterbildabgleich (Template Matching Approach) und Strukturanalyse.

2.7.1.1 Musterbildabgleich

Beim Musterbildabgleich wird das Punkteraster eines Zeichens als ganzes betrachtet und mittels einer oder mehrerer Funktionen in einen eindimensionalen Merkmalvektor umgewandelt. Es wird davon ausgegangen, daß ein Zeichen immer einem bekannten Musterzeichen gleicht und daher auch immer den gleichen Vektor liefert. Der entstehende Vektor wird dann mit

dem Vektor eines bekannten Musterbildes von jedem Zeichen verglichen. Im folgenden werden einige Verfahren des Musterbildabgleichs vorgestellt.

Grauwertmatrix

Im einfachsten Fall eines Musterbildabgleichs werden die Helligkeitswerte des zweidimensionalen Bildes in einen eindimensionalen Vektor kopiert, in dem alle Zeilen nacheinander folgen. Diese Methode führt zu keiner kompakteren Beschreibung. Auch wird unwesentliches nicht unterdrückt und wichtiges nicht hervorgehoben. Wird das Zeichenbild vor dem Musterabgleich größennormiert und hierbei verkleinert, so findet eine geringe Komprimierung statt. Auch wird hierdurch eine Größenunabhängigkeit erreicht und kleine Störungen werden nivelliert. Diese Methode wird im Rest der Arbeit als Verwendung der Grauwertmatrix bezeichnet. Eine Grauwertmatrix der Größe $X \times Y$ bedeutet, daß das Bild größennormiert wird auf eine Breite von X Punkten und eine Höhe von Y Punkten.

„Peak-Hole"

Bei der „Peak-Hole"-Methode werden aus dem Zeichenbild nur einige als signifikant angenommene Punkte verwendet. Hier findet schon eine größere Komprimierung statt, als bei der Verwendung der Grauwertmatrix. Das Verfahren ist aber auch bedeutend anfälliger gegen Störungen und geringste Verdrehungen. Zu entscheiden bei dieser Methode ist insbesondere, wieviele Punkte verwendet werden sollen und welche Punkte signifikant sind. Dies wird dann schwierig zu entscheiden, wenn nicht nur Zeichen zu klassifizieren, sondern auch Störungen und verklebte Zeichen sicher abzulehnen sind. Welche Punkte erlauben es sicher Störungen oder verklebte Zeichen von einzelnen Zeichen zu unterscheiden? Diese Methode ist daher nur einsetzbar, wenn Störungen und Verklebungen ausgeschlossen werden können.

Caliper-Distanz

Die *Caliper-Distanz* wird meistens bei der Suche nach Trennstellen verwendet. Sie kann aber auch zur Zeichenklassifizierung verwendet werden. Für jede der vier Seiten eines Zeichens wird die Caliper-Distanz berechnet. Die berechneten Werte aller Seiten bilden dann zusammen den Merkmalvektor. Damit der Merkmalvektor auch bei unterschiedlich großen Zeichen eine konstante Länge hat, wird der Vektor durch Stauchung oder Streckung größennormiert.

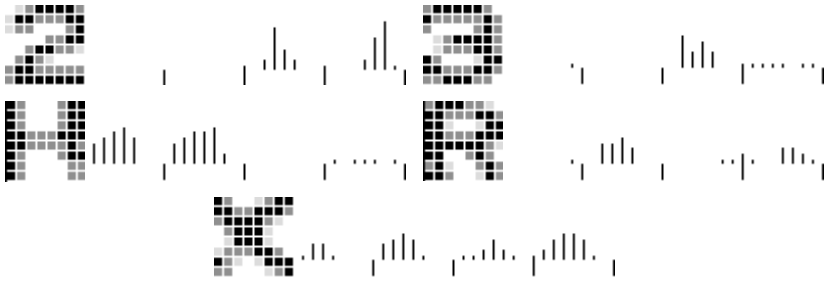


Abbildung 2.4: Einige Zeichen und rechts daneben der Merkmalvektor aus der Caliper-Distanz mit 8 Merkmalen pro Seite.

Abbildung 2.4 zeigt einige Zeichen und ihre Caliper-Distanzen. Der Merkmalvektor aus der Caliper-Distanz ist in vier Abschnitte unterteilt, entsprechend den vier Zeichenseiten. Die Grenzen der Abschnitte sind durch einen senkrechten Strich nach unten angedeutet. Der erste Abschnitt ist die Caliper-Distanz berechnet von oben, der zweite von unten, der dritte von links und der vierte von rechts. In den Grauwertmatrizen zählen hellgraue Punkte noch zum Hintergrund. Dies erklärt warum die Caliperdistanzen nicht mit dem optischen Eindruck der Zeichen übereinstimmen. So ist z.B. beim H die von oben berechnete Caliperdistanz nicht konstant, obwohl beim Betrachten des Zeichens diese Erwartung entsteht. Die Merkmalvektoren aus den Caliper-Distanzen unterscheiden sich deutlich bei den verschiedenen Zeichen. Es werden aber auch Schwächen offensichtlich. Theoretisch ist das X ein symmetrisches Zeichen, alle vier Caliper-Distanzen sollten daher identisch sein. In der Praxis sind sie recht unterschiedlich ausgeprägt, haben aber zumindest alle den selben Verlauf: Ansteigend zur Mitte, dann abfallend.

Momente

Momente sind eine sehr beliebte Methode, vergl. die Arbeiten von Hu [Hu62], Prokop und Reeves [PrRe92], Kim und Yuan [KiYu94], Cash und Hatamian [CaHa87] und Desai und Cheng [DeCh94]. Der Merkmalvektor wird kompakt und ist, je nach verwendeten Momenten, invariant gegenüber Änderungen von Größe, Lage, Verdrehung und Helligkeit. Momente werden zur Klassifizierung von Zeichen wie auch von Gebäuden, Brücken, Flugzeugen und Werkstücken eingesetzt. Sie sind daher auch bestens geeignet, um Firmenlogos zu klassifizieren. Zernike und pseudo-Zernike Momente gelten als die besten Momente. Sie erfordern aber auch den größten Rechenaufwand. Am schnellsten zu berechnen sind Moment-Invarianten und Standardmomente.
Obige Autoren erzielen mit Momenten in Verbindung mit dem *k-Nearest-Neighbour* Klassifizierungsalgorithmus Erkennungsraten von weit über 90% bis zu 99%. Jedoch haben viele Autoren ihre Experimente zur Zeichenerkennung nur im "Labor", ohne praktische Anwendung, gemacht. Welche Ergebnisse, bei einer praktischen Anwendung zu erwarten sind, ist daher unklar. Auch machen die Autoren keine Angaben, ob es auch eine Klasse von abzulehnenden Zeichen gab.

Fourier-Deskriptoren

Fourier-Deskriptoren werden aus dem Umriß eines Zeichens berechnet. Sie sind daher abhängig vom Startpunkt der Umrißbeschreibung und gelten als sehr anfällig für Störungen. Zerfällt ein Zeichen in mehrere Teile, können sie dieses nicht mehr erkennen, es sei denn die Teile werden wieder zusammengesetzt. Sie erfordern das Bestimmen der Innenräume von Zeichen, um zuverlässig zu klassifizieren. Mit zusätzlichem Aufwand können sie invariant gegenüber Änderungen von Lage, Orientierung, Größe und Startpunkt gemacht werden. Bei den von Mori und Yamamoto erläuterten Experimenten, wurde eine Punktmatrix von 64x32 Punkten verwendet. Die Erkennungsrate lag bei 98%. Aufgrund der geringeren Zeichengröße bei meiner Anwendung, rechne ich mit einer weitaus geringeren Erkennungsrate. Insbesondere kann ich aufgrund der geringeren Größe nicht sicher sein, daß auch die Innenräume von Zeichen gefunden werden. Die Verwendung der Innenräume führt dazu, daß die Anzahl der Merkmale nicht bei allen Zeichen einheitlich ist, nicht alle Zeichen haben gleich viele Innenräume. Diese Varianz verhindert den Einsatz von Standardklassifikationsverfahren wie *k-Nearest-Neighbour*, etc.

Bei einem Vergleich der vorgestellten Musterbildabgleich-Methoden scheinen mir Momente am vielversprechendsten. Sie erfüllen am besten, die an Merkmale gestellten Bedingungen. Sie sind kompakt, erlauben die Verwendung von Standardklassifikationsverfahren und abstrahieren von unwesentlichem. Letzteres kann behauptet werden, da Momente größtenteils größe-, lage- und rotationsunabhängig sind.
Welche Art von Momenten die Beste ist, läßt sich nicht so einfach beantworten. Denn die zuverlässigeren Methoden erfordern wieder mehr Rechenzeit. Es muß also abgewogen werden zwischen Erkennungsrate und Rechenzeit. Da die Erkennungsraten verschiedener Verfahren für meine Anwendung noch unbekannt sind, kann erstmal nur eine provisorische Entscheidung für ein schnelles und einfaches Verfahren getroffen werden. Erweist sich dieses als zu unzuverlässig, kann ein besseres verwendet werden. Eine schnelles und einfaches Verfahren sind die sieben Invarianten von Hu [Hu62].

Methoden des Musterbildabgleichs sind nur für Druckbuchstaben einsetzbar. Handschriftliche Zeichen weichen zu stark vom Musterbild ab. Für handschriftliche Zeichen kommen nur Methoden der Strukturanalyse in Frage.

2.7.1.2 Strukturanalyse

Der Strukturanalyse liegt die Idee zugrunde, daß Menschen Zeichen als linien- und kreisförmige Gebilde betrachten. Ziel der Strukturanalyse ist daher, in dem Zeichenbild solche oder ähnliche Formen zu finden und die Beziehungen zwischen diesen Strukturen zu beschreiben. Merkmale in der

Strukturanalyse sind dementsprechend Formen und deren Beziehungen untereinander.
Mori, Suen und Yamamoto unterteilen die Methoden der Strukturanalyse in 6 Gruppen, bei denen die Grenzen jedoch mehr oder minder fließend sind:

- Thinning Line Analysis
- Bulk Decomposition
- Stream Following Analysis
- Vectorization
- Contour Following Analysis
- Background Analysis

Eine Empfehlung für eine der Methoden geben sie aber nicht. Auch gibt es keine direkten Vergleiche zwischen den Methoden. Eine Angabe der Vor- und Nachteile der einzelnen Methoden fehlt ebenfalls. Als Beispiel für Methoden der Strukturanalyse seien die "Thinning Line Analysis" und "Vectorization" erläutert.

Thinning Line Analysis

Bei der "Thinning Line Analysis" (Linien-Ausdünnungs-Methode) werden Zeichen als linienhafte Gebilde betrachtet. In einem ersten Arbeitsschritt werden die Linien ausgedünnt, so daß sie anschließend nur noch einen Punkt breit sind. Dies wird auch als Skelettierung bezeichnet und dient teilweise nur als Vorverarbeitung zur Eliminierung von Unterschieden in der Strichbreite. Linienendpunkte und Linienschnittpunkte werden ermittelt und als Klassifizierungsmerkmale verwendet.
Mindy Bokser gibt in [Boks92] Beispiele, in denen die Skelettierung dazu führt, daß Zeichen nicht mehr unterscheidbar sind, obwohl sie vor der Skelettierung sich noch deutlich unterscheiden. Sie lehnt daher eine Skelettierung ab.

Vectorization

Dies ist eine Methode, die in gewisser Weise Skelettierung, "Bulk Decomposition" und "Stream Following" vereint bzw. enthält. Liegen die Zeichen als vertikal *lauflängenkodiertes* Bild vor, so wird jeder Lauf in der Zeichenfarbe ein Knoten eines Graphen. Läufe aus aufeinanderfolgenden Spalten die sich überlappen werden durch eine Kante zwischen den Knoten der Läufe im Graphen markiert. Der Graph kann dann nach einigen Normierungen als Merkmal zur Klassifizierung verwendet werden. Alternativ lassen sich aus dem Graphen Informationen gewinnen über die Anzahl und Lage der Löcher, die konkave Hülle und das umschließende Rechteck des Zeichens, sowie Anzahl und Lage von Schnitt- und Endpunkten. Diese Informationen werden dann zur Klassifizierung verwendet. Kahan Pavlidis und Baird beschreiben in [KaPB87] ein System, welches so einen Graphen erzeugt. Den Graphen verwenden sie zur Skelettierung der Zeichen. Lage und Richtung der durch die Skelettierung entstandenen Linien verwenden sie als Klassifikationsmerkmale.

Die Methoden der Strukturanalyse führen dazu, daß Zeichen unterschiedlich viele Merkmale haben, z. B. unterschiedlich viele Schnittpunkte und Linienendpunkte oder unterschiedlich große Graphen. Die Folge sind aufwendigere Verfahren zur Klassifizierung. Insbesondere können keine Standardverfahren wie *k-Nearest-Neighbour* oder *Neuronale Netze* verwendet werden. Sondern es müssen Verfahren des *maschinellen Lernens* angewendet werden, oder es muß über Umwege wieder eine Darstellung erreicht werden, die eine konstante Anzahl von Merkmalen hat (vergl. [KaPB87]). Weiter erfordern Strukturanalyse-Verfahren zuerst die Auswahl von gewünschten Merkmalen und anschließend müssen Methoden gefunden und realisiert werden, um diese Merkmale zu gewinnen. Bei Musterbildabgleich-Methoden werden die Merkmale mittels einfacher mathematischer Funktionen gewonnen.
Strukturanalyse-Verfahren sind dementsprechend in der Realisierung aufwendiger als Musterbildabgleich-Methoden. Beim Musterbildabgleich können zur Klassifizierung fertige Programmpakete verwendet werden.

2.7.1.3 Entscheidung für eine Methode

Alle vorgestellten Methoden haben Stärken und Schwächen. Die besten Ergebnisse werden sicherlich erzielt, wenn mehrere der Methoden parallel verwendet werden und bei unterschiedlichen Ergebnissen das häufigste Ergebnis als richtig angenommen wird. Im Rahmen dieser Diplomarbeit kann nur eine einfache Methode realisiert werden. Aufgrund der Vielzahl von Methoden und des Mangels an direkten Vergleichen zwischen diesen fällt eine Entscheidung für ein Verfahren nicht leicht und kann nur subjektiv erfolgen.
Da in meiner Anwendung nur Druckbuchstaben vorkommen und für diese die Leistungsfähigkeit von Musterbildabgleich-Verfahren ausreicht, ist ein Nutzen des Mehraufwandes zur Realisierung eines Strukturanalyse-Verfahrens fraglich. Hinzu kommt der Vorteil der Musterbildabgleich-Verfahren, daß diese auch zur Erkennung von Firmenlogos verwendet werden können. Diese sind nicht zwingend linien- und kreisförmige Gebilde, damit ist nicht sicher, daß diese von Strukturanalyse-Verfahren behandelt werden können, ohne diese Verfahren dafür speziell zu erweitern oder zu verbessern.
Aufgrund obiger Überlegungen habe ich mich entschieden mit einer Musterbildabgleich-Methode zu arbeiten. Bei der Realisierung wird jedoch darauf geachtet, daß das Programmodul zur Zeichenerkennung einfach durch ein anderes ersetzt werden kann, so daß später bessere Verfahren verwendet werden können, falls die Leistungsfähigkeit der gewählten Musterbildabgleich-Methode unzureichend ist. Wie bereits beim Vergleich der Musterbildabgleich-Methoden beschrieben, scheinen Momente für diese Anwendung am Geeignetsten zu sein.
Das zu erstellende Erkennungssystem wird daher mit den sieben Invarianten-Momenten als Klassifizierungsmerkmal arbeiten. Es werden sowohl die alphanumerischen Zeichen als auch die Firmenlogos mit Momenten klassifiziert.

An dieser Stelle sei darauf hingewiesen, daß die Formel für das siebte Moment in dem Buch von Gonzalez und Woods [GoWo92] und auch in dem Artikel von Prokop und Reeves [PrRe92] falsch ist. Im Vergleich zur richtigen Formel von Hu [Hu62] ist einmal ein Index vertauscht und in der anderen Quelle ein Vorzeichen falsch. Wer mit Momenten Arbeiten will sollte also auf den Originalarikel von Hu zurückgreifen.

Momente können sowohl im Binärbild als auch in einem Grauwertbild berechnet werden. Soll ein Grauwertbild verwendet werden, so muß eine Helligkeitsnormierung vorgenommen werden. Dieses Problem wird im folgenden behandelt.

2.7.1.4 Helligkeitsnormierung

Da ein Grauwertbild mehr Informationen enthält und damit ein Zeichen auch besser beschreibt als ein Binärbild, werde ich die Momente im Grauwertbild berechnen. Die Verwendung von Binärbildern bei vielen Autoren hat meistens historische Gründe. Speicher und Rechenleistung waren vor einigen Jahren noch knapp und so war es in den meisten Anwendungen nicht möglich mit Grauwerten zu arbeiten, da hierzu der vorhandene Speicher nicht ausreichte. Heute ist das kein Problem mehr.

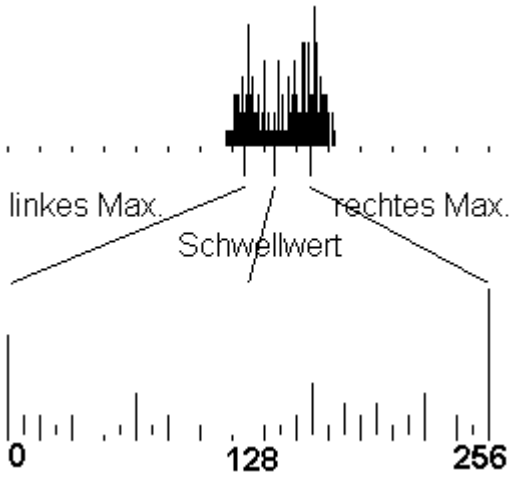


Abbildung 2.5: Helligkeitsnormierung mittels des Histogramms. Oben das Histogramm des unnormierten Bildes, unten das des normierten. Die drei Striche zwischen den Histogrammen verdeutlichen die Aufteilung der Helligkeitswerte. Der mittlere Strich entspricht im oberen Histogramm dem Schwellwert und im unteren immer 128. Werte im oberen Histogramm ausserhalb der Striche werden auf die Extremwerte 0 und 255 abgebildet. Werte zwischen den Strichen werden auf den gesammten Wertebereich des unteren Histogramms umgerechnet.

Soll mit einem Graubild gearbeitet werden, so muß das Bild helligkeitsnormiert werden; sonst führen unterschiedliche Zeichenfarben und

Chipoberflächen auch zu unterschiedlichen Merkmalvektoren. Zur Normalisierung der Helligkeit wird im Zeichenbild der Schwellwert mittels des Diskriminanz-Verfahrens bestimmt (vergl. Abschnitt 2.4.1.2). Zusätzlich werden links und rechts von diesem Schwellwert die Grauwerte bestimmt, die am häufigsten im Zeichenbild vorkommen. Alle Grauwerte zwischen dem linken Maximum und dem Schwellwert werden dann auf die normierten Helligkeiten 0 bis 127 linear abgebildet. Alle Grauwerte kleiner dem linken Maximum werden zu Null. Analog werden alle Grauwerte größer dem rechten Maximum zu 255. Alle Grauwerte zwischen dem Schwellwert und dem rechten Maximum werden auf den Wertebereich 128-255 abgebildet. Diesem Verfahren liegt die Annahme zugrunde, daß die Helligkeiten der Zeichen- und Hintergrundpunkte eine gaußförmige Verteilung haben (vergl. 2.4.1.3). Der Erwartungswert dieser Verteilungen sollte daher ungefähr in der Nähe der ermittelten Maxima liegen. Punkte deren Grauwert extremer ist, als die Erwartungswerte, können sicher als Zeichen- oder Hintergrundpunkt angesehen werden und bekommen auch im normierten Zeichenbild die extremsten Werte. Nur bei Punkten die zwischen den beiden Erwartungswerten liegen, ist die Zuordnung als Zeichen- oder Hintergrundpunkt anzweifelbar. Besonders wenn der Grauwert dem Schwellwert relativ nahe kommt. Die Grauwerte zwischen den Maxima werden daher entsprechend ihrer Nähe zum Schwellwert abgestuft. Die *Abbildung 2.5* verdeutlicht dieses Vorgehen.

Nachdem eine Entscheidung für ein Merkmalextraktionsverfahren getroffen wurde, muß nun ein passendes Klassifikationsverfahren gewählt werden. Im folgenden wird ein Überblick über Klassifikationsverfahren gegeben.

2.7.2 Klassifikationsverfahren

Die Aufgabe eines Klassifikationsverfahrens ist es, für einen eingegebenen Merkmalvektor eine Ausgabe zu erzeugen, so daß für alle Merkmalvektoren mit gleicher Bedeutung genau die gleiche Ausgabe und für alle Merkmalvektoren mit einer anderen Bedeutung auch eine andere Ausgabe erzeugt wird. Anhand der Ausgabe kann dann auf die Bedeutung eines Merkmalvektors geschlossen werden.
Damit ein Klassifikationsverfahren dies kann, muß es zuerst mit einer Menge von Vektoren angelernt werden. Diese Vektoren heißen Lern- oder Trainingsdaten. Die Leistungsfähigkeit des Klassifikationsverfahrens wird dann mittels einer weiteren von den Lerndaten disjunkten Menge von Merkmalvektoren überprüft. Was beim anlernen genau passiert, ist bei jedem Klassifikationsverfahren unterschiedlich.

Es werden drei große Gruppen von Klassifikationsverfahren unterschieden:
- neuronale Netze
- maschinelles Lernen
- statistische Verfahren

Ein Vergleich verschiedenster Verfahren aus allen drei Bereichen und bei verschiedensten Anwendungen findet sich in [MiST94]. Dort finden sich auch Verweise auf weiterführende Literatur.

2.7.2.1 Neuronale Netze

Der Ansatz zu neuronalen Netzen stammt von dem Versuch, die Arbeitsweise des menschlichen Gehirns zu simulieren. Es gibt viele Arten von neuronalen Netzen, die meisten bestehen aus einer Menge von untereinander verbundenen Knoten. Die Knoten berechnen aus ihrer Eingabe mittels meistens nichtlinearer Funktionen eine Ausgabe. Die Eingabe eins Knotens ist entweder die Ausgabe eines anderen Knotens oder ist Teil der Eingabedaten an das neuronale Netz. Die Ausgabe einiger Knoten wird als Ergebnis des neuronalen Netzes verwendet. Ein großer Nachteil Neuronaler Netze ist, daß die von ihnen gelieferten Ergebnisse für den Menschen nicht nachvollziehbar sind. Ungünstig ist auch, daß sie immer eine konstante Anzahl von Eingaben erfordern.

2.7.2.2 Maschinelles Lernen

Verfahren des maschinellen Lernens erzeugen aus gegebenen Beispielen Regeln, die diese Beispiele korrespondierend zu ihren Klassen unterscheiden. Die Regeln können auch verwendet werden, um neue, ungelernte Beispiele zu klassifizieren. Der Vorteil von Verfahren des Maschinellen Lernens ist, daß die entstehenden Regeln für einen Menschen verständlich sind und daß variabel große Merkmalvektoren verarbeitet werden können.

2.7.2.3 Statistische Verfahren

Den Statistischen Verfahren liegt ein Wahrscheinlichkeitsmodell zugrunde. Es wird einem Merkmalvektor eine Wahrscheinlichkeit für die Zugehörigkeit zu einer Klasse zugeordnet. Der Vektor wird zur wahrscheinlichsten Klasse zugehörig klassifiziert.
Das am häufigsten verwendete statistische Verfahren ist das k-Nearest-Neighbour Verfahren. Ein Merkmalvektor wird als zu der Klasse zugehörig angenommen, die am häufigsten vertreten ist unter den k Trainingsvektoren, die zu dem untersuchten Merkmalvektor den kleinsten Abstand haben. Als Maß für den Abstand wird meistens der Euklidische-Abstand verwendet. Andere Maße sind aber denkbar und auch sinnvoll, wenn der Merkmalvektor unterschiedliche Merkmaltypen enthält, die unterschiedlich gewertet werden müssen. Da es immer einen Trainingsvektor gibt, der den kleinsten Abstand zum zu klassifizierenden Merkmalvektor hat, werden Merkmalvektoren immer einer Klasse zugeordnet und nie zurückgewiesen. Um Merkmalvektoren auch zurückweisen zu können, wird ein maximaler Abstand definiert. Ist der Abstand vom Merkmalvektor zum nächsten Trainingsvektor größer als der maximale Abstand, wird der Merkmalvektor als nicht klassifizierbar

zurückgewiesen. Statistische Verfahren können nur mit einem Merkmalvektor konstanter Größe arbeiten.
Zwei weitere statistische Verfahren sind das Hyperrectangle und das Hypersphere Verfahren. Das Hyperrectangle Verfahren erzeugt für jeden n-dimensionalen Merkmalvektor, mit dem gelernt wird, ein n-dimensionales Rechteck. Dieses Rechteck wird verkleinert, falls die Rechtecke von Merkmalvektoren anderer Klassen sich mit diesem überlappen. Die Rechtecke werden dann so verkleinert, daß sie sich nicht mehr überlappen. Liegt ein Merkmalvektor im Zentrum eines Rechtecks eines anderen Merkmalvektors der gleichen Klasse, so wird für diesen Merkmalvektor kein eigenes Rechteck erzeugt. Das Hypersphere Verfahren funktioniert wie das Hyperrectangle Verfahren. Hier werden aber Kreise an Stelle von Rechtecken erzeugt.
Wettscherek und Diettrich [WeDi95] haben das Hyperrectangle und das k-Nearest-Neighbour Verfahren miteinander verglichen. Sie sind zu dem Ergebnis gekommen, daß das k-Nearest-Neighbour-Verfahren fast immer besser ist als das Hyperrectangle Verfahren. Nur wenn die Verteilung der Merkmalvektoren im Merkmalraum tatsächlich rechteckig ist, schneidet das Hyperrectangle Verfahren besser ab. Bei Wettscherek und Diettrich ist das k-Nearest-Neighbour Verfahren deutlich besser bei der Zeichenerkennung, als das Hyperrectangle Verfahren. Ein Vergleich zwischen Hypersphere und k-Nearest-Neighbour ist mir nicht bekannt.

2.7.2.4 Entscheidung für ein Verfahren

Die Wahl für ein Klassifikationsverfahren fällt leicht, nachdem entschieden wurde, daß Momente als Merkmale verwendet werden sollen. Alle mir bekannten Autoren, die Momente verwenden, benutzen den k-Nearest-Neighbour Algorithmus zur Klassifizierung. Ich werde diesen auch verwenden, um meine Ergebnisse vergleichbar mit den ihren zu halten. Es ist jedoch möglich jedes andere Klassifikationsverfahren mit Momenten zu verwenden.
Am Lehrstuhl ist eine Software-Bibliothek vorhanden, die neben dem k-Nearest-Neighbour auch das Hypersphere Verfahren beinhaltet. Hierdurch ist es kein großer Aufwand, die Zeichenerkennung auch einmal mit dem Hypersphere Verfahren zu testen. Ich werde daher einen Vergleich zwischen k-Nearest-Neighbour und Hypersphere Verfahren durchführen.

Sowohl für den k-Nearest-Neighbour als auch für das Hypersphere Verfahren müssen Parameter bestimmt werden. Beim k-Nearest-Neighbour das k und der maximale Abstand, beim Hypersphere Verfahren der anfängliche Kugelradius. Für die Parameter werde ich verschiedene Werte probieren und miteinander vergleichen.
Neben den Parametern müssen noch Lerndaten und Testdaten erzeugt werden. Hierzu wird eine Serie von Chipbildern gemacht und die Zeichen als Lerndaten gespeichert. Eine weitere Serie von Chipbildern von anderen Chips gleicher Bauart wird verwendet, um eine erste Testdatenmenge von Zeichen zu erzeugen. Eine zweite Testdatenmenge wird von Chips erzeugt,

die nicht in der Lernserie enthalten waren. Mit der ersten Testdatenmenge kann die Leistungsfähigkeit des Zeichenerkennungssystems bei bekannten Schriftarten getestet werden. Die zweite Testdatenmenge testet das Verhalten bei unbekannten Schriftarten. Für die Lerndaten werden nur Chips verwendet, von denen mehrere Exemplare vorhanden sind, sonst könnte die erste Testdatenmenge nicht erzeugt werden. Ansonsten erfolgt die Chipauswahl zufällig. Wieviele Chips hierfür erforderlich sind, wird sich ergeben. Die Klassifikatoren sollen nicht nur mit Zeichen getestet werden, sondern auch mit Abzulehnendem, wie Störungen, verdrehten und verklebten Zeichen und Zeichenbruchstücken.

2.8 Trennen von Zeichen

Eine perfekte Segmentierung wird selten erreicht, teilweise sind Zeichen schon auf dem Chip verklebt, teilweise verkleben sie aufgrund einer zu geringen Kameraauflösung, teilweise aufgrund der Vorverarbeitung. Ein Verfahren zum Trennen verklebter Zeichen ist daher unbedingt erforderlich. Doch wie sollen Stellen bestimmt werden, an denen Verklebtes getrennt wird, wenn nicht bekannt ist welche Zeichen vorliegen und dementsprechend welche Breite die Zeichen haben? Drei Verfahren werden vorgestellt.

Berechnung der vertikalen Projektion

Der einfachste Ansatz zur Lösung dieses Problems geht von der Annahme aus, daß zwischen allen Zeichen ein Spalt existiert, dieser jedoch durch Störungen überbrückt wurde. Durch Berechnung der *vertikalen Projektion* in jeder Spalte, werden die Spalten als mögliche Trennstellen angesehen, deren vertikale Projektion minimal ist. Leider finden sich Minima mit dieser Methode nicht nur zwischen Zeichen, sondern auch innerhalb einzelner Zeichen. Die gefundenen Trennstellen müssen daher überprüft werden. Zu diesem Zweck werden sie der Reihe nach ausprobiert, indem die einzelnen entstehenden Abschnitte klassifiziert werden. Wird eine Abfolge von Trennstellen gefunden, die für alle Abschnitte eine Zuordnung zu einer Zeichenklasse bestimmt, werden diese Trennstellen als richtig betrachtet.
Simon Kahan und Theo Pavlidis schlagen vor, statt der vertikalen Projektion dessen Ableitung zu verwenden, da eine echte Trennstelle häufig durch einen Ansprung in der vertikalen Projektion gekennzeichnet ist. In ihren Experimenten erhielten sie hierdurch bessere Ergebnisse als bei Verwendung der vertikalen Projektion (siehe [KaPB87]).

Nachbarverfahren

Shuichi Tsujimoto und Haruo Asada schlagen alternativ die Verwendung folgendes Kriteriums vor, das ich als Nachbarverfahren bezeichnen werde. Für jede Spalte wird die Anzahl der schwarzen Punkte ermittelt, die auch in der nachfolgenden Spalte noch schwarz sind. Mögliche Trennstellen sind dann alle Spalten, die eine minimale Anzahl solcher Punkte enthalten. Trennstellen, die nahe einem starken Ansprung liegen, werden als

wahrscheinlicher angesehen und bevorzugt betrachtet. Sie zeigen an einem theoretischen Beispiel, daß diese Methode besser ist, als die Verwendung der vertikalen Projektion (siehe [TsAs92]).

Caliper-Distanz

Ein weiteres Kriterium für Trennstellen ist die Verwendung der *Caliper-Distanz*. Dabei werden die vertikalen Caliper-Distanzen vom oberen und unteren Rand addiert. Sind beide Einzelwerte gleich der Zeichenhöhe, das bedeutet es befindet sich kein Vordergrundpunkt in dieser Spalte, so wird ihre Summe gleich der Zeichenhöhe definiert. Dies führt zu konsistenteren Caliper-Distanz Werten. Eine erweiterte Version der Caliper-Distanz verwenden H. Fujisawa, Y. Nakano und K. Kurino mit großem Erfolg bei verklebten handgeschriebenen Zahlen.(vgl. [FuNK92])

Bei den beiden ersten Verfahren sind alle Minima und die Randwerte mögliche Trennstellen. Ein Minimum liegt auch dann vor, wenn der Vorgänger- bzw. Nachfolgerwert gleich dem aktuellen Wert ist und der Nachfolger- bzw. Vorgängerwert größer als der aktuelle Wert ist.
Bei der Caliper-Distanz sind alle Maxima und die Randwerte mögliche Trennstellen. Zur Vereinheitlichung sollte daher nicht direkt mit der Caliper-Distanz gearbeitet werden, sondern mit der Invertierung: Zeichenhöhe - Caliper-Distanz. Wird im folgenden von der Caliper-Distanz im Zusammenhang mit der Zeichentrennung gesprochen, so ist der invertierte Wert gemeint.

Alle Verfahren liefern zu viele Trennstellen. Mit ihnen kann nur gearbeitet werden, wenn ein Klassifikator vorhanden ist, der Zeichenbruchstücke sicher als nicht klassifizierbar bewertet.
In meiner Anwendung kommt keine kursive Schrift und nur lateinische Großbuchstaben und arabische Ziffern vor. Bei kursiver Schrift versagen alle Verfahren. H. Fujisawa, Y. Nakano und K. Kurino beschreiben in [FuNK92], wie das Caliper-Verfahren für kursive Schrift erweitert werden kann. Kleinbuchstaben sollten die Leistungsfähigkeit der Verfahren nicht beeinträchtigen. Bei anderen Alphabeten, z. B. dem japanischen, könnten alle Verfahren schlecht sein. Hier müßte die Anwendbarkeit genauer untersucht werden.

Die Überlegenheit des Nachbarverfahrens gegenüber der vertikalen Projektion wurde bereits durch Tsujimoto und Asada gezeigt. Ob das Nachbarverfahren auch besser ist als die Caliper-Distanz, kann erst ein direkter Vergleich zeigen. Da ich Keinen in der Literatur finden konnte, die Verfahren aber nicht sehr aufwendig sind, werde ich beide realisieren und miteinander vergleichen.

2.9 Zusammenfassung des daraus abgeleiteten Konzepts

Das angestrebte Erkennungssystem hat damit den in *Abbildung 2.8* gezeigten Aufbau. Der Übersicht halber sind die Zeichenerkennung in die *Abbildung 2.6*

und das Trennen in die *Abbildung 2.7* ausgelagert. Die Abbildungen zeigen den Datenfluß. Neben den Pfeilen stehen die Daten in kursiver Schrift. Zwischen den Pfeilen stehen die Funktionen in fetter Schrift. Pfeile, die auf eine Funktion zeigen, symbolisieren die Eingaben für diese Funktion, wegführende Pfeile die Ausgaben. Die Ein- und Ausgaben eines gesamten Datenflußbildes sind eingerahmt. Die verwendeten oder möglichen Algorithmen stehen hinter den Funktionen in Klammern und normaler Schrift. Firmenlogos werden mit dem gleichen Zeichenerkennungsverfahren erkannt, wie Ziffern und Buchstaben.

↓ *helligkeitsnormiertes Graubild* ↓ *Orientierung* ↓ *Zeichenkoordinaten*
korrekt gedrehtes Zeichenbild erstellen ↓ *Zeichenbild* **Merkmale berechnen** (7 invariante Momente nach Hu) ↓ *Merkmalvektor* **Klassifizierung** (k-Nearest-Neighbour, Hypersphere)
↓ *Interpretation bzw. nicht klassifizierbar*

Abbildung 2.6: Ablauf der geplanten Zeichenerkennung

↓ *helligkeitsnormiertes Graubild* ↓ *Orientierung* ↓ *Zeichenkoordinaten*
Trennstellen bestimmen (Nachbar- und Caliper-Verfahren) ↓ *Trennstellen* **Zeichenhypothesen mittels Trennstellen erstellen** ↓*neue Zeichenhypothesen* **Zeichenerkennung bis gesamter Ausschnitt klassifiziert werden kann** **(**Ausprobieren der Hypothesen von links nach rechts)
↓ *Interpretation bzw. nicht klassifizierbar*

Abbildung 2.7: Ablauf der geplanten Trennung

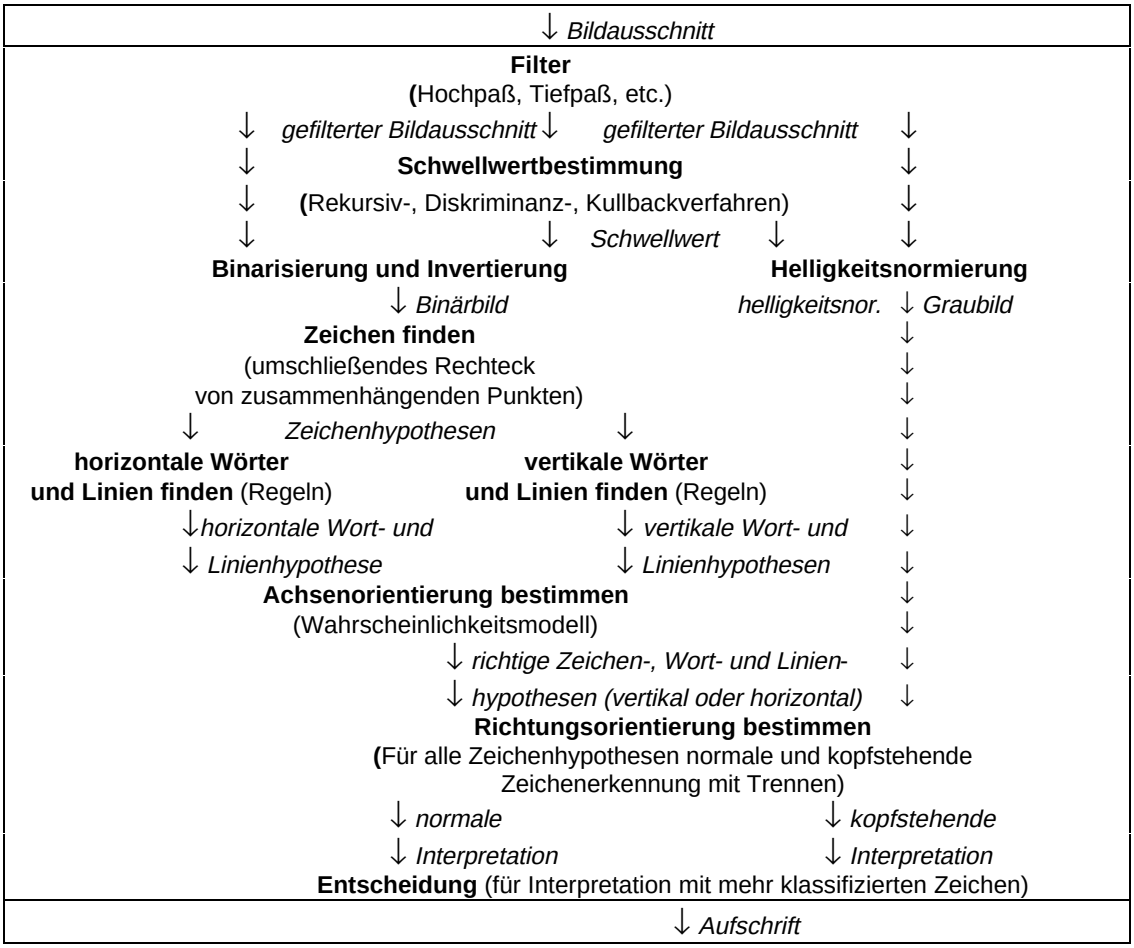


Abbildung 2.8: Aufbau des erarbeiteten Konzeptes

3 Ergebnisse der Realisierung und Verbesserungen

Im vorangegangenen Kapitel wurde ein Konzept hergeleitet. Grundlage der Herleitung waren theoretische Überlegungen und Erfahrungen und Ergebnisse aus anderen Arbeiten. In diesem Kapitel werden die Erfahrungen mit der Realisierung des Konzeptes geschildert. Die gefundenen Schwächen werden erläutert und vorgenommene oder mögliche Verbesserungen und Ergänzungen beschrieben.

Das Konzept wurde für das Betriebssystem Windows 3.1 realisiert. Die grafische Benutzeroberfläche wurde in Visual BASIC, die eigentlichen Programmroutinen in der Programmiersprache C implementiert. Ein Softwarepaket mit dem k-Nearest-Neighbour Verfahren und dem Hypersphere Verfahren war vorhanden und wurde benutzt. Alle Laufzeitangaben der Realisierung wurden auf einem PC mit Pentium 75 Prozessor gemessen.

In den folgenden Unterkapiteln werden die Etappen der Bildverarbeitung einzeln in der Reihenfolge ihrer Durchführung besprochen. Zum Abschluß des Kapitels wird das realisierte Gesamtsystem zusammengefaßt.

3.1 Digitalisierung

Die Aufgabe der Digitalisierung ist es, ein Grauwertbild von den Chips zu erzeugen, welches von einem Rechner weiterverarbeitet werden kann. In der Beschreibung der Apparatur im Kapitel 2.1 wurden bereits zwei mögliche Verbesserungen angesprochen, ein Autofokus und ein elektronisches Zoom-Objektiv. In der Praxis zeigte sich noch ein mögliches Beleuchtungsproblem bei gravierter Schrift. Dieses soll erst im Kapitel 3.4 über die Binarisierung genauer besprochen werden.
Ein spürbarer Nachteil durch unscharfe Bilder, mangels Autofokus, konnte nicht festgestellt werden. Einzig bei Eproms scheitert die Schrifterkennung aufgrund der kleinen Schrift und der Höhe der Chips. Dies ist vernachläßigbar, da Eproms bereits durch ihr Löschfenster identifiziert werden. Auf ein Lesen der Aufschrift wird bei ihnen verzichtet.
Bei sehr kleinen Chips, ist auch die Schrift oft nicht mehr entzifferbar. Sollen diese auch verarbeitet werden können, so ist ein Zoom-Objektiv unumgänglich. In der Praxis haben so kleine Chips aber keinen Wert. Ihr Recycling wäre teurer, als die Neuproduktion. Die wirtschaftlich interessanten Chips können alle mit dem vorhandenen Objektiv gelesen werden. Ein Zoom-Objektiv würde trotzdem der Zuverlässigkeit der Schrifterkennung zu Gute kommen.

3.2 Vorverarbeitung

Ziel der Vorverarbeitung ist die Verbesserung des Bildes, so daß Störungen unterdrückt werden und eine Unterteilung in Zeichenpunkte und Hintergrundpunkte leichter möglich ist.
Es wurden verschiedene Filter und Filterkombinationen ausprobiert und ihr Einfluß auf die Qualität der anschließenden Binarisierung untersucht. Um die

Ergebnisse beurteilen zu können, muß definiert werden, was unter Qualität verstanden wird. Durch eine Filterung erhalten die Grauwerte eines Bildes einen neuen Wert. Nach der Filterung sollten alle Punkte, die zu Zeichen gehören, einen Wert größer einem Schwellwert haben und alle Punkte, die zum Hintergrund gehören, einen Wert kleiner einem Schwellwert. In der Praxis gelingt dies jedoch nicht, einige Punkte haben weiterhin einen zu niedrigen oder zu hohen Grauwert. Die Anzahl dieser Punkte ist dabei abhängig von dem gewählten Schwellwert. Die Qualität der Filterung kann nicht über die Anzahl solcher Punkte definiert werden, sondern eine wichtigere Rolle spielt die Lage dieser Punkte im Bild und der durch sie verursachte Fehler bei der Segmentierung.

Hintergrundpunkte, die Zeichen zugeordnet werden, verursachen folgende Fehler:

- Zeichen werden großflächiger als sie wirklich sind.
- Nebeneinander liegende Zeichen verkleben.
- Untereinander liegende Zeichen verkleben.
- Im Hintergrund werden Zeichenpunktmengen gefunden obwohl dort keine sind.

Zeichenpunkte, die dem Hintergrund zugeordnet werden, verursachen folgende Fehler:

- Zeichen werden dünner, skeletthafter.
- Zeichen zerfallen in mehrere Mengen.
- Teile von Zeichen werden nicht gefunden.

Wie schwerwiegend die einzelnen Fehler für das Gesamtkonzept sind soll im folgenden dargestellt werden. Die Fehler sind hier entsprechend ihrer Tragweite sortiert. Die gravierendsten Fehler werden zuerst erläutert.
Irreparabel ist der letztgenannte Fehler. Fehlen Teile von Zeichen, können diese Teile durch die nachfolgenden Schritte nicht rekonstruiert werden, denn nach der Binarisierung wird nur noch mit den Zeichenpunktmengen gearbeitet. Alle Punkte die zu Zeichen gehören und nicht in diesen Mengen enthalten sind, werden auch danach nicht mehr berücksichtigt. Die Zeichen können damit auch nicht erkannt werden.
Folgenschwer ist auch das Verkleben von untereinander liegenden Zeichen. In meinem Konzept ist bisher kein Verarbeitungsschritt vorgesehen, der solche Fehler erkennt und behebt. Falls sich dieser Fehler nicht vermeiden läßt, müßte ich mein Konzept um eine weitere Komponente erweitern.
Zerfallen einige Zeichen in mehrere Teile, ist dies durchaus reparabel und bei einigen Zeichen wie Umlauten oder Doppelpunkt der Normalfall. Der Anteil der zerfallenen Zeichen an der Anzahl aller Zeichen sollte aber gering sein. Zeichen sollten auch schlimmstenfalls in drei Teile zerfallen, sonst ist ein zusammenfügen fast unmöglich. Zerfallen Zeichen in mehrere nebeneinander liegende Teile, die alle die volle Höhe haben, ist dies einfacher zu reparieren, als der Zerfall in mehrere untereinander liegende Teile, denn die Höhe ist das wichtigste Kriterium bei der Zuordnung der Zeichen zu Wörtern.
Das Finden von Zeichenpunktmengen im Hintergrund ist solange unschädlich, wie die Anzahl dieser Mengen deutlich in der Minderheit bleibt und somit die Zusammenstellung von Wörtern und Linien und die daraus folgende Bestimmung der Achsenorientierung nicht verfälscht. Wichtig ist ein

Zeichenklassifikator, der diese Störmengen sicher als nicht klassifizierbar zurückweist.
Das Verkleben von nebeneinander liegenden Zeichen ist der am wenigsten störende Fehler. Viele Verklebungen verlängern nur die Auswertungszeit. Ein Trennen ist jedoch fast immer möglich, sofern der Zeichenklassifikator verklebte Zeichen sicher zurückweist. Um die Auswertungszeit nicht unnötig in die Länge zu treiben, sollte natürlich auch dieser Fehler möglichst selten vorkommen. Einige Zeichen sollten aber immer unverklebt gefunden werden können. Dies ist zur Bestimmung der Richtungsorientierung unbedingt erforderlich (siehe Abschnitt 3.6.2).
Die Verdünnung und Vergrößerung der Zeichen beschränkt sich in der Regel auf den Rand der Zeichen. Hier werden einzelne Punkte aus dem Zeichenrand entfernt oder hinzugefügt. Solange es bei einzelnen Punkten bleibt, ist dies vernachlässigbar. Werden es zu viele Punkte, folgt daraus meistens einer der anderen Fehler.
Aus der Auflistung und Tragweite der Fehler wird deutlich, daß die Zuordnung von Hintergrundpunkten zu Zeichenmengen meistens weniger gravierend ist, als der umgekehrte Fall.

Die Qualität einer Filterung wird anhand der Art und Häufigkeit der möglichen Fehler bewertet. Hierzu werden von jedem gefilterten Bild drei Binärbilder erzeugt. Es werden drei Bilder erstellt, da drei Schwellwertverfahren bei der Binarisierung verwendet werden können und noch nicht untersucht wurde, welches Verfahren das geeignetste ist. In den Binärbildern werden die Kettencodes bestimmt. Die Qualität der entstandenen Kettenkodes und damit der Filterung wurde dann anhand der obigen Kriterien bewertet und verglichen. Zuerst wurde für jeden Filter das Bild mit den besten Kettencodes aus den drei Vorhandenen ausgewählt. Anschließend wurden die Filter untereinander anhand ihrer besten Kettencodes verglichen. Der Filter mit dem besten Ergebnis bekam drei Punkte für diesen Chip, der zweitbeste Filter zwei Punkte und der Dritte einen Punkt. Waren mehrere Filter gleich gut, so bekamen sie alle die gleiche Punktzahl. Die nachfolgenden Plätze wurden dann auch noch weiter mit Punkten versehen.

Erste Experimente mit dem Sobelfilter als Ableitungsfilter zeigten, daß dieser sehr schlechte Ergebnisse liefert. Es verkleben fast immer alle Zeichen. Das Verkleben läßt sich auf das Verwenden der absoluten Werte der Gradienten zurückführen. *Abbildung 3.1* zeigt das Auffüllen eines Spaltes durch den Sobelfilter. An den weißen Stellen im Ausgangsbild werden die beiden Nachbarn betrachtet, um den neuen Grauwert zu berechnen. Diese sind je einmal schwarz und weiß. Der neue Grauwert des dritten Punktes wird damit zu g_3=abs(-2*1 + 2*0) = 2. Analog g_4 = abs(-2*0 + 2*1) = 2. Beide Punkte werden schwarz. Würde nicht der absolute Wert verwendet, wäre g_3= -2 und bliebe weiß.

Abbildung 3.1: Spaltauffüllung durch Sobelfilter. Schwarze Punkte = 1, weiße Punkte = 0. Links: Ausgangsbild; Rechts: Bild nach der Filterung

Aufgrund dieser Erfahrungen wurde noch ein zusätzlicher einfachere Ableitungsfilter entworfen und getestet:

$$eA = \frac{1}{2}\begin{pmatrix} 0 & 1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

Der Filter wird als einfacher bezeichnet, da er nur eine Matrix hat. Dieser erhöht oder erniedrigt den Grauwert eines Punktes entsprechend der Stärke des Helligkeitswechsels seiner Umgebung. Der in *Abbildung 3.1* gezeigte Spalt bleibt bei diesem Filter erhalten.

Abbildung 3.2 - *Abbildung 3.5* zeigen die Ergebnisse einiger Chips mit den verschiedenen Filtern. Die Bilder sind repräsentativ für den Gesamteindruck der Filterergebnisse aus 50 Bildern. Die verwendeten Filter sind, entsprechend der Reihenfolge in den Bildern:

- ohne Filterung
- Hochpaß
- Tiefpaß
- Ableitungsfilter (einfach)
- Ableitungsfilter (Sobel)
- Hochpaß / Tiefpaß
- Tiefpaß / Hochpaß
- Tiefpaß / Ableitungsfilter (einfach)
- Median
- Median / Hochpaß

Der Hochpaßfilter hat den Parameter A gleich 10/9. Die Parameter des Tiefpaßfilters sind alle 1 (Vergleiche Kapitel 2.2). Die nach der Binarisierung gefundenen Zeichenpunktmengen sind in den Bildern grün umrandet.

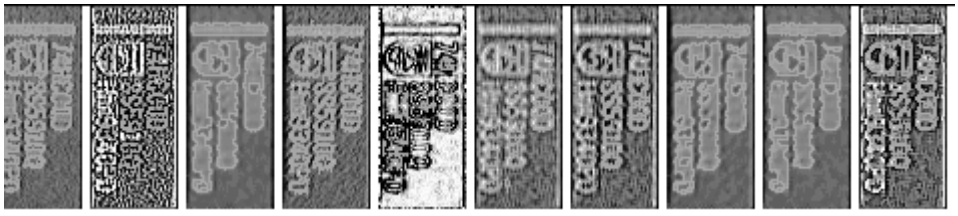


Abbildung 3.2: Ergebnisse verschiedener Filterverfahren. Hochpaß-, Sobelfilter und Hochpaßfilterkombinationen finden nicht alle Zeichenpunkte. Beim Tiefpaß-, Tiefpaß-/ einfacher Ableitungs- und Medianfilter sind alle Zeichen verklebt.

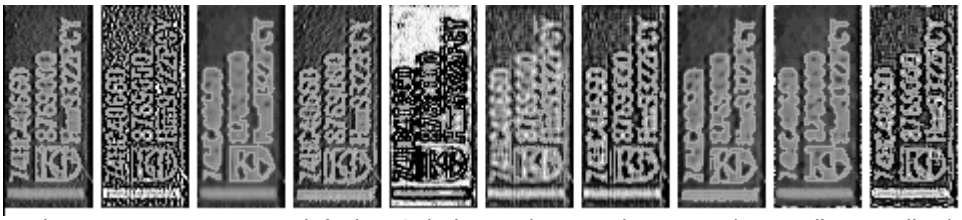


Abbildung 3.3: Ergebnisse verschiedener Filterverfahren. Beim Tiefpaß-, Tiefpaß-/ einfacher Ableitungs- und Medianfilter sind alle Zeichen verklebt.

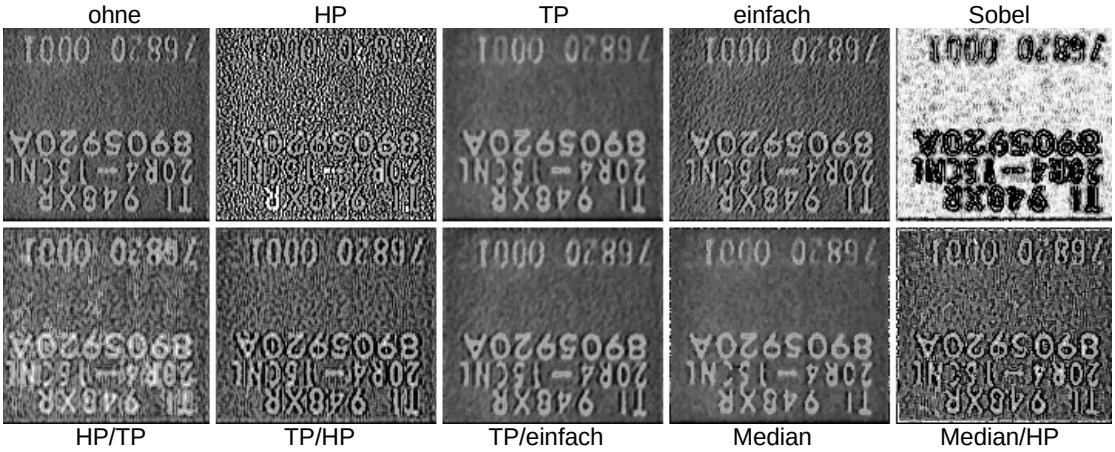


Abbildung 3.4: Ergebnisse verschiedener Filterverfahren. Ungefiltertes Kamerabild, Tiefpaß- und Medianfilter finden die Zeichenpunkte nur ungenügend.

Die Ergebnisse des Sobelfilters sind sehr schlecht, wie bereits vorausgehend beschrieben. Obwohl die Kanten der Zeichen deutlich dunkler sind als der Hintergrund, gibt es oft keinen Schwellwert, der diese trennt. Gibt es einen, dann verkleben fast alle Zeichen.
Das Problem, daß es manchmal keinen trennenden Schwellwert gibt, hat auch der Hochpaßfilter und Kombinationen mit diesem.
Der Tiefpaß und der Medianfilter haben dieses Problem selten. Hier nur in *Abbildung 3.4*. In diesem Bild finden sie nicht alle Zeichenpunkte. Gleiches gilt auch für das ungefilterte Kamerabild. Der einfache Ableitungsfilter schneidet hier am besten ab. Hat aber auch geringe Probleme. In *Abbildung 3.2* und *Abbildung 3.3* sind, nach der Tiefpaß oder Median-Filterung, alle Zeichen einer Zeile miteinander verklebt. Dies ist äußerst ungünstig, da für die Bestimmung der Richtungsorientierung einige wenige unverklebte Zeichen benötigt werden. Der einfache Ableitungsfilter trennt die meisten Zeichen voneinander. Dies wird auch in *Abbildung 3.5* deutlich. In dieser Abbildung schaffen es nur der Hochpaßfilter und der einfache Ableitungsfilter das Firmenlogo und das "S" zu trennen, dies verdeutlichen die roten Markierungen, die den Umriß einer gefundenen Zeichenpunktmenge darstellen. Bei allen anderen Filtern sind Logo und „S“ verklebt. Beim Ableitungsfilter zerfällt das Firmenlogo zwar in zwei Teile, dies ist aber weniger problematisch, als das Verkleben von Zeichen verschiedener Zeilen.

Abbildung 3.5: Ergebnisse verschiedener Filterverfahren. Nur der Hochpaß- und einfache Ableitungsfilter trennen das Firmenlogo und das „S“. Bei allen anderen Filtern sind diese verklebt. Die rote Markierung ist der Umriß einer als zusammengehörig gefundenen Punktmenge.

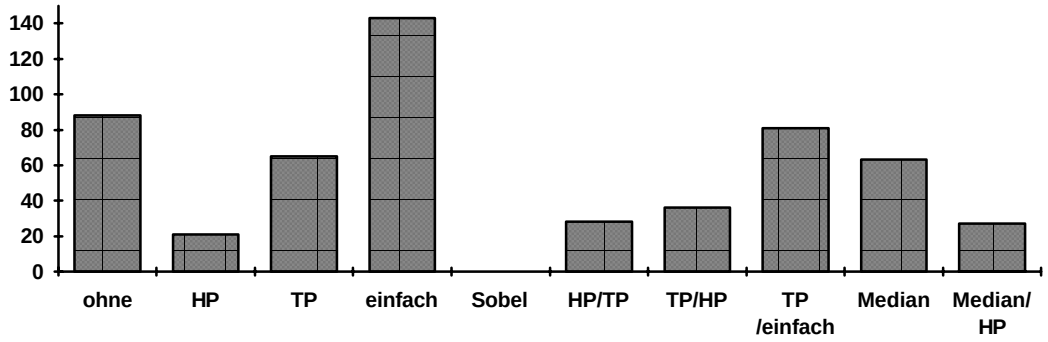


Abbildung 3.6: Erreichte Punktzahl der verschiedenen Filter

In *Abbildung 3.6* sind die erreichten Punktwerte der verschieden Filter für 50 Bilder dargestellt. Folgende Bilanz kann gezogen werden: Der Sobelfilter, der Hochpaßfilter und Kombinationen mit dem Hochpaßfilter sind für diese Anwendung ungeeignet. Da sie oft nicht alle Zeichen finden. Tiefpaß-, Medianfilter und auch das ungefilterte Bild sind häufig, insbesondere bei großen Zeichen zufriedenstellend. Am häufigsten überzeugt jedoch der einfache Ableitungsfilter. Zum einen durch das geringste Verkleben, zum Anderen durch das Finden der meisten Zeichen.
Trotzdem kann es vorkommen, daß bei dem einen oder anderen Chip ein anderes Verfahren bessere Ergebnisse erzielt als der einfache

Ableitungsfilter. Wünschenswert wäre ein Algorithmus, der das jeweils günstigste Filterverfahren für ein Bild ermittelt.
Aufgrund der Ergebnisse dieses Kapitels wird im weiteren Verlauf der Arbeit mit dem einfachen Ableitungsfilter gearbeitet. Dabei muß noch auf einen Nachteil dieses Filters hingewiesen werden. Der Filter ist richtungsabhängig. Je nachdem ob von rechts nach links oder von links nach rechts gefiltert wird, sind die Ergebnisse unterschiedlich, gleiches gilt für die vertikale Richtung.
Dies liegt daran, daß der Filter nicht spiegelsymetrisch ist und keine absoluten Werte berechnet werden. Die Filterrichtung ist immer gleich, von oben nach unten und von links nach rechts. Da die Zeichen aber beliebig orientiert sein können und demzufolge später gedreht werden müssen, macht sich diese Richtungsabhängigkeit doch bemerkbar. Um sie zum Teil auszugleichen, wird die Größe des umschließenden Rechtecks eines Zeichens vor der Drehung am rechten und unteren Rand um eins reduziert, bzw. falls das Bild nach der Filterung noch invertiert wurde am oberen und linken Rand. Die *Abbildung 3.7* verdeutlicht die Richtungsabhängigkeit und ihre Korrektur. Die Korrektur verhindert vor allem, daß das Zeichen verschoben wird.

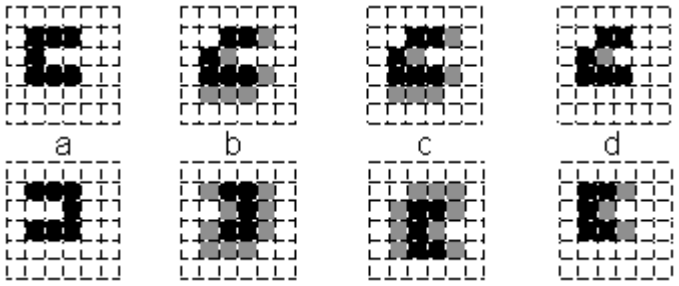


Abbildung 3.7: Filterung eines Zeichens mit dem einfachen Ableitungsfilter.
(a) Oben Zeichen mit normaler Orientierung; Unten das gleiche Zeichen um 180° verdreht. (b) Zeichen nach der Filterung. (c) Ergebnis nach der Drehung des unteren Bildes ohne Korrektur. (d) Ergebnis mit Korrektur vor der Drehung.

3.3 Vorsegmentierung

Im Konzept ist keine Vorsegmentierung vorgesehen. Nur bei einem Chip hat sich dies so negativ ausgewirkt, daß er nicht mehr bearbeitet werden konnte.
Die im folgenden Abschnitt verwendeten Schwellwertverfahren zur Binarisierung sind nur für zweifarbige Bilder konzipiert. Wie die *Abbildung 3.8* zeigt, sind Chips mit Gehäuseeinbuchtungen dreifarbig. Neben der Zeichen- und Oberflächenfarbe haben die Gehäuseeinbuchtungen einen eigenen, dunkleren Helligkeitswert. In den meisten Fällen stört dies jedoch die Ermittlung eines Schwellwertes nicht. Die Bestimmung des Schwellwertes funktioniert bei dem obersten Chip in *Abbildung 3.8* ebenso wie bei allen verfügbaren Baugleichen, bis auf eine Ausnahme, diese zeigt das mittlere Bild.
Der problematische Chip läßt sich besser Segmentieren, wenn das Histogramm zur Schwellwertbestimmung nur aus dem kreuzförmigen Bereich bestimmt wird, der die Gehäuseeinbuchtungen ausschließt. Dies ist in der dritten Zeile der *Abbildung 3.8* verdeutlicht. Ein solches Vorgehen ist ohne

Probleme bei allen quadratischen Chips möglich, es würde nur dann scheitern, wenn ein Chip ausschließlich in den ausgesparten Ecken beschriftet ist. Dies ist unwahrscheinlich und mir ist kein solcher Chip bekannt. Nach der Bestimmung des Schwellwertes wird die gesamte Chipoberfläche inklusive der Ecken binarisiert. Die Seitenlänge der auszusparenden Ecken entspricht 25% der Seitenlänge des Chips. Dieser Wert wurde empirisch ermittelt.

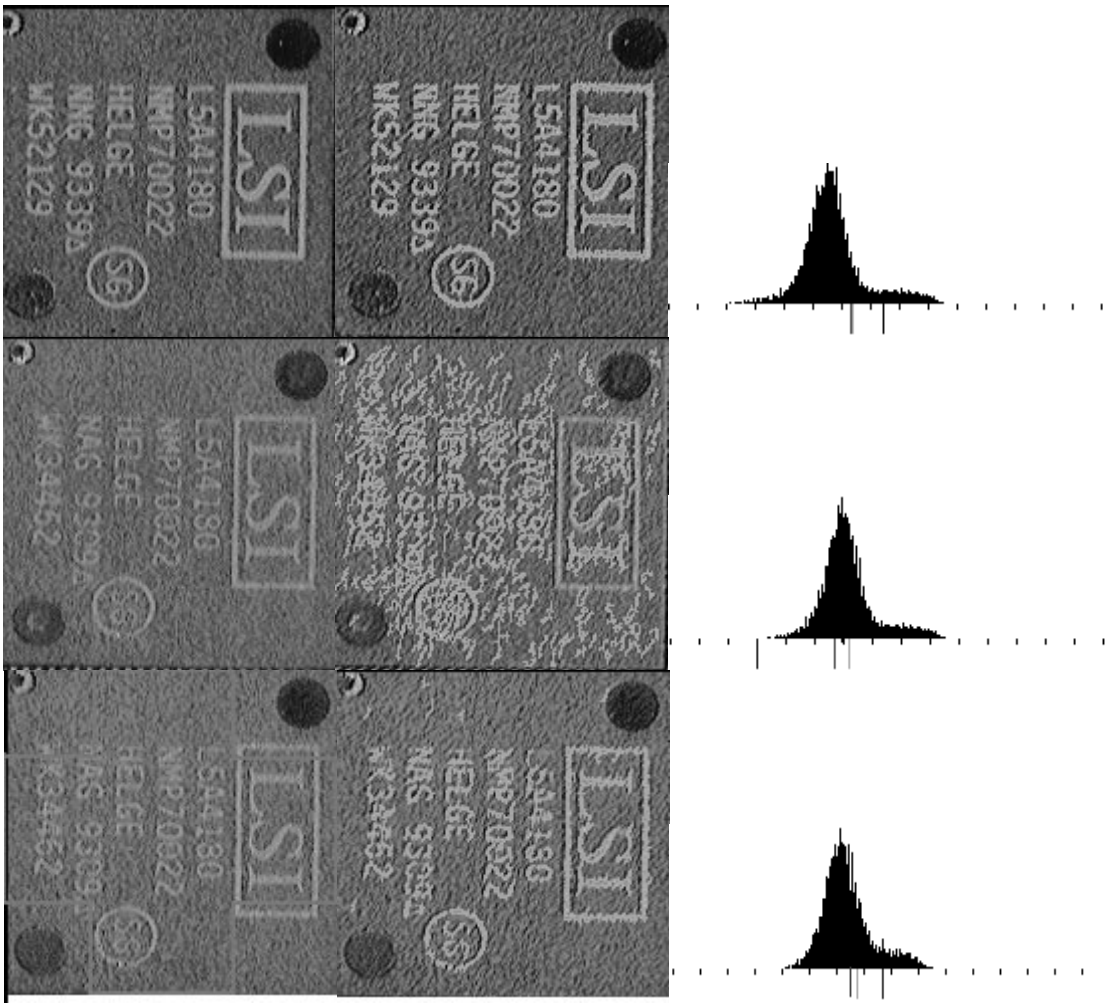


Abbildung 3.8: Gelingen und Scheitern an Chips gleicher Bauart und Verbesserung des Schwellwertes durch kreuzförmigen Bereich zur Histogrammberechnung. In den Histogrammen verdeutlichen die farbigen Striche die gefundenen Schwellwerte der drei Verfahren: (rot) Rekursives-, (grün) Diskriminanz- und (blau) Kullback-Verfahren. Die mit dem Schwellwert des Diskriminanz-Verfahrens ermittelten Zeichenpunktmengen sind in den Chipbildern grün umrandet.

Da dieses Problem ein Einzelfall ist, wurde weder die oben beschriebene einfache Lösung für diesen Chip implementiert, noch eine allgemeinere Vorsegmentierung nachträglich erstellt. Es ist jedoch zu vermuten, daß in den meisten Fällen eine Vorsegmentierung die den Chipbereich ermittelt der die Schrift enthält die ermittelten Schwellwerte zur Binarisierung verbessern würde. Ob dies insgesamt bessere Ergebnisse nach sich ziehen würde, ist eine andere Frage.

3.4 Binarisierung

Nach der Vorverarbeitung kann das Bild binarisiert werden. Hierzu ist ein Schwellwert erforderlich. Erste Versuche zeigten, daß verschiedene Chips auch unterschiedliche Schwellwerte benötigen. Es muß also ein automatisches Verfahren zur Schwellwertbestimmung verwendet werden. Drei Verfahren wurden getestet. Zusätzlich wurde mit der Pixelaggregation ein anderer Ansatz untersucht, der eine Binarisierung erübrigt und direkt zusammenhängende Punktmengen bestimmt.

3.4.1 Schwellwertbestimmungsverfahren

Alle drei Verfahren liefern gute Ergebnisse, wenn das Bild einen guten Kontrast hat. Bei Bildern mit schlechtem Kontrast zeigen sich dann die Stärken und Schwächen der Verfahren.
Die in diesem Abschnitt gezeigten Abbildungen verwenden die Farben rot für das Rekursive-, grün für das Diskriminanz- und blau für das Kullback-Verfahren. In Histogrammen sind durch diese Farben die gefundenen Schwellwerte der Verfahren markiert. Auf Chipbildern sind die mit diesen Schwellwerten gefundenen Zeichenpunktmengen entsprechend farbig umrandet.

3.4.1.1 Kullback-Distanz-Verfahren

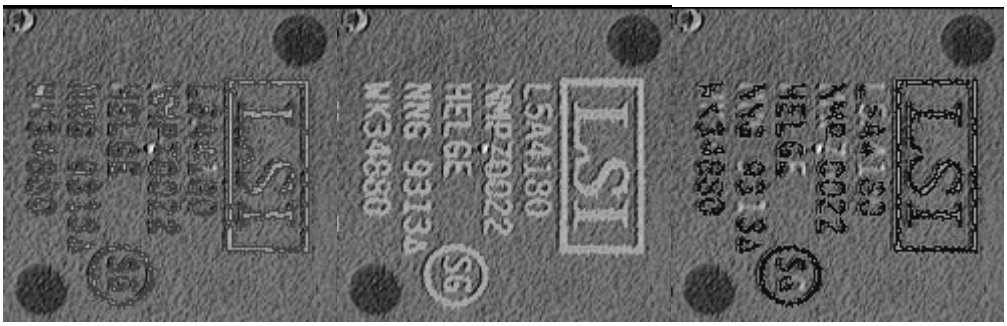
Das Kullback-Distanz-Verfahren neigt dazu den Schwellwert zu hoch anzusetzen. Von den Zeichen fehlen dann einzelne Linienzüge, wie in *Abbildung 3.9* dargestellt. Auf der anderen Seite verkleben nur sehr selten Buchstaben bei diesem Verfahren und Störungen werden größtenteils unterdrückt. Ein anderes Problem dieses Verfahrens ist die Überbewertung einiger weniger extremer Helligkeitspunkte. Dies ist in *Abbildung 3.10* verdeutlicht. Aufgrund der Überbewertungen und den oft zu hohen Schwellwerten ist dieses Verfahren für diese Anwendung nicht einsetzbar.

3.4.1.2 Diskriminanzanalyse-Verfahren

Das Diskriminanzanalyse-Verfahren liefert im Mittel bessere Ergebnisse als das Kullback-Distanz-Verfahren. Trotzdem ist der ermittelte Schwellwert des öfteren etwas zu niedrig, so daß mehr Buchstaben verkleben als nötig und Störungen im Hintergrund auftreten. Der ermittelte Schwellwert ist aber immer brauchbar. Verklebte Buchstaben können in der Regel auch später immer getrennt werden.

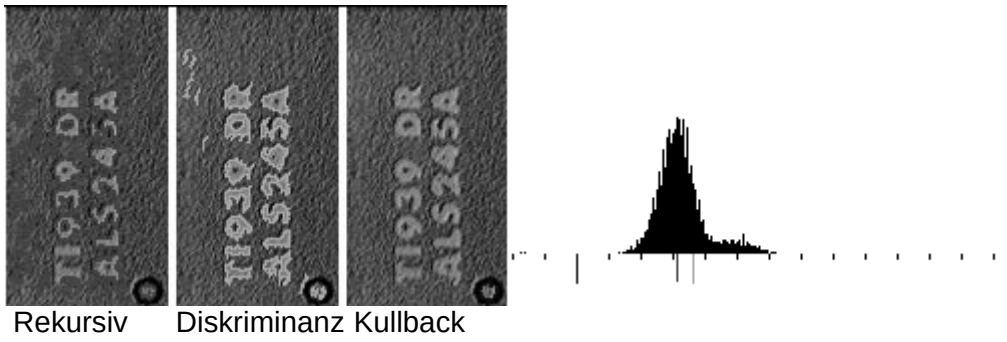
3.4.1.3 Rekursives-Verfahren

Das Rekursive-Verfahren scheitert an einigen Chips. Es scheitert immer dann, wenn der Helligkeitsunterschied zwischen Vordergrund und Hintergrund zu gering wird. Es scheitert jedoch auch in anderen Situationen (siehe *Abbildung 3.10*). In den meisten Fällen liefert es ähnliche Ergebnisse wie das Diskriminanzanalyse-Verfahren. Da es nicht für alle Chips einsetzbar ist, kann es nicht verwendet werden.



Rekursives-Verfahren Diskriminanz-Verfahren Kullback-Verfahren

Abbildung 3.9: Vergleich der Schwellwertverfahren Teil 1. Rekursives- und Diskriminanz-Verfahren liefern beide den Schwellwert 113 und damit das identische linke und mittlere Bild. Das Kullback-Verfahren liefert den Schwellwert 123. Das Ergebnis ist im rechten Bild dargestellt. Der Wert ist zu hoch. Einzelne Zeichen und Zeichenteile werden nicht gefunden.



Rekursiv Diskriminanz Kullback

Abbildung 3.10: Vergleich der Schwellwertverfahren Teil 2. Das Diskriminanz-Verfahren im mittleren Bild findet alle Zeichen. Der ermittelte Schwellwert ist im Histogramm grün dargestellt. Das Rekursive-Verfahren im linken Bild scheitert komplett. Der bereits nach einer Iteration ermittelte Schwellwert, im Histogramm rot markiert, entspricht fast dem am häufigsten auftretenden Grauwert. Das Kulback-Verfahren setzt den Schwellwert (blau) so, daß die sehr schwarze Gehäuseeinbuchtung in der unteren Ecke gefunden wird. Es gewichtet die wenigen schwarzen Punkte am linken Rand des Histogramms stärker, als die große Anzahl grauer Zeichenpunkte.

Keines der Verfahren ist immer das beste, sondern jedes liefert dann und wann die besseren Ergebnisse. Von allen drei getesteten Verfahren ist nur das Diskriminanzanalyse-Verfahren zuverlässig. Die beiden anderen Verfahren finden oft keinen brauchbaren Schwellwert. Die mit dem Diskriminanz-Verfahren erzielten Ergebnisse sind nicht optimal aber ausreichend. Die Ergebnisse können verbessert werden, durch eine erneute Binarisierung nach der Bestimmung der Zeichenhypothesen. Bei der erneuten Binarisierung wird für jede Zeichenhypothese ein eigener Schwellwert berechnet. Dabei werden nur die Punkte berücksichtigt, die innerhalb des umschließenden Rechtecks der Zeichenhypothese liegen. Nach der erneuten Binarisierung sind nur noch solche Zeichen verklebt, die auch im Kamerabild verklebt sind.

Bei einigen Chips sind alle drei Verfahren unzureichend. Zum einen werden bei sehr kleinen, Fingernagel großen und kontrastarmen Chips oft nicht alle Zeichen gefunden. Zum anderen gibt es Probleme mit eingravierter Schrift.

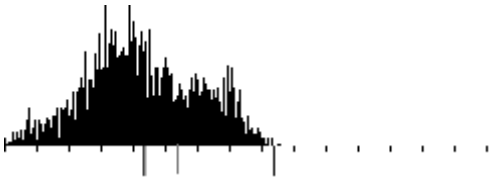


Abbildung 3.11: Histogramm eines kleinen Chips in dem keine sinnvollen Schwellwerte gefunden werden. Der lila Strich markiert den durch manuelles Ausprobieren besten gefundenen Schwellwert.

Bei kleinen Chips ist das Histogramm sehr breit gestreut und keine zwei Bereiche zu erkennen, wie in *Abbildung 3.11* dargestellt. Der lila Strich markiert den optimalen von Hand ermittelten Schwellwert. Da solch kleine Chips für eine Wiederverwertung uninteressant sind, wurden keine Anstrengungen zu ihrer besseren Binarisierung unternommen, bzw. die Ursachen für das schlechte Ergebnis genauer untersucht.

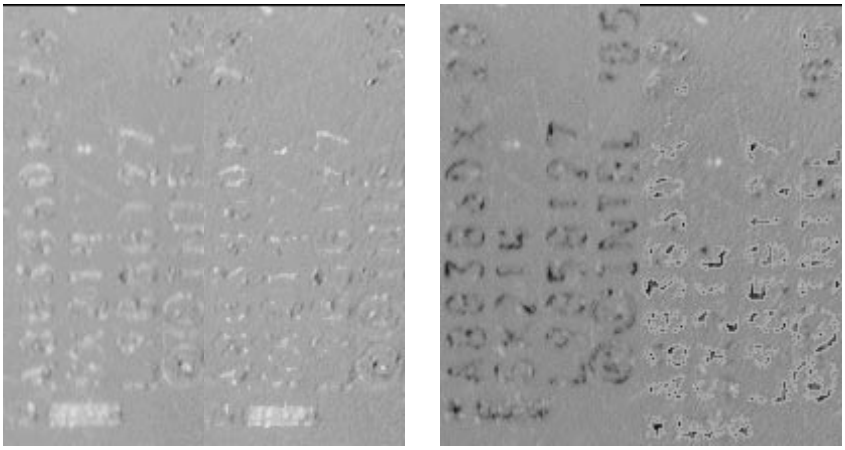


Abbildung 3.12: Einfluß der Beleuchtung auf gravierte Schrift. Links: Chip direkt beleuchtet. Die Zeichen erscheinen weiß. Rechts: Indirekte Beleuchtung die Zeichen erscheinen schwarz, ihre wirkliche Farbe.

Neben der Beschriftung von Chips durch einen Farbaufdruck, sind manche Chips durch eingravieren gekennzeichnet. Diese Chips können durch keines der getesteten Schwellwertverfahren binarisiert werden. Bei allen Verfahren sind die Schwellwerte zu niedrig. Die Zeichen zerfallen in viele kleine Zeichenhypothesen. Auch durch manuelles Ausprobieren ist es nicht möglich, einen passenden Schwellwert zu finden. Zum lesen dieser Chips muß daher ein anderer Lösungsansatz gefunden werden, oder an Kameraaufbau und Beleuchtung „gedreht“ werden. Das hier insbesondere eine andere Beleuchtung hilfreich sein könnte, zeigt die *Abbildung 3.12*. Für das linke Bild wurden der Chip direkt unter die Kamera und damit ins Zentrum der Beleuchtung gelegt. Für das rechte Bild wurde der Chip an den Rand des Kameraausschnittes gelegt und dementsprechend anders beleuchtet. Die aufgenommene Zeichenfarbe ändert sich hierdurch von weiß nach schwarz. Eine Binarisierung gelingt jedoch immer noch nicht. Neben dem Problem des

Findens einer besseren Beleuchtung und einer besseren Binarisierung, stellt sich das Problem des automatischen Erkennens eines Chips mit eingravierter Schrift. Nur dann kann automatisch von der normalen Beleuchtung auf die Beleuchtung für gravierte Schrift umgestellt werden. Aus zeitlichen Gründen konnte für diese Probleme keine Lösung erarbeitet werden.

In der Vorverarbeitung wurde das Kamerabild gefiltert. Das gefilterte Bild soll jetzt binarisiert werden. Der Schwellwert könnte sowohl im ungefilterten wie auch im gefilterten Bild berechnet werden. Durch Experimente haben ich festgestellt, daß bessere Ergebnisse zu erzielen sind, wenn der Schwellwert im ungefilterten Bild berechnet wird. Dies läßt sich folgendermaßen erklären. Zur Ermittlung des Schwellwertes werden Erwartungswerte und Streuungen berechnet. Implizit wird angenommen, daß die Helligkeitswerte eine bestimmte, kamerabedingte Verteilung haben. Durch die Filterung erhalten die Helligkeitswerte eine neue Verteilung. Diese neue Verteilung weicht stärker von der Verteilung ab, die den Schwellwertverfahren zugrunde liegt, als die Helligkeitsverteilung im ungefilterten Bild. Die ermittelten Schwellwerte sind dementsprechend schlechter.

3.4.2 Pixelaggregation

Neben den Schwellwertverfahren wurde auch die Pixelaggregation zum Finden von zusammengehörenden Punkten verwendet. Die Ergebnisse waren aber unzureichend. Die Gründe hierfür sind hauptsächlich der geringe Kontrast zwischen Zeichen und Hintergrund und die rauhen Chipoberflächen. Ersteres verbietet eine Tiefpaßfilterung des Bildes und letzteres erfordert genau eine solche Filterung. Eine rauhe Chipoberfläche bedeutet, daß die Helligkeitswerte des Hintergrundes stark schwanken. Damit durch die Pixelaggregation nicht hunderte von kleinen zusammengehörenden Punktmengen im Hintergrund gefunden werden, muß entweder die Parametereinstellung der Pixelaggregation relativ große Helligkeitsunterschiede in einer Punktmenge zulassen, oder das Bild durch eine Tiefpaßfilterung geglättet werden.
Werden relativ große Helligkeitsunterschiede erlaubt, wird der größte Teil des Hintergrundes als eine Punktmenge zusammengefaßt. Aber die Grenzen zwischen den Punkten einzelner Zeichen und auch zwischen Zeichen- und Hintergrundpunkten werden überwunden. Viele Zeichen verkleben, die gefundenen Zeichenpunkte enthalten auch Punkte die zum Hintergrund gehören und Teile der Zeichen werden dem Hintergrund zugeordnet.
Der andere Ansatz, der Glättung des Bildes durch eine Tiefpaßfilterung vor der Pixelaggregation führt zu einem verschwimmen der Grenzen zwischen Zeichen- und Hintergrundpunkten. Die Ergebnisse entsprechen denen des ersten Ansatzes.
Eine vorhergehende oder nachträgliche Hochpaßfilterung führt ebenfalls zu keinen besseren Ergebnissen. Die Hochpaßfilterung ändert die Helligkeitswerte an den Schnittstellen zwischen Vordergrund und Hintergrund auf sehr hohe oder sehr niedrige Werte. Der Abstand dieser Werte zu den Helligkeitswerten der sonstigen Vordergrund und Hintergrundpunkten ist so groß, daß die Punkte an den Schnittstellen nicht mehr zugehörig zu einer Gruppe von sonstigen Punkten interpretiert werden. Die Punkte an den

Schnittstellen werden zu eigenen Punktmengen. Die Zeichen zerfallen damit in mehrere Punktmengen. Bei vielen Chips ist es, trotz optimaler von Hand ermittelter Aggregationsparameter, nicht möglich die Punkte richtig in Zeichen- und Hintergrundpunkte einzuteilen.
Anhand zweier Histogramme in *Abbildung 3.13* soll dies noch mal verdeutlicht werden. Das obere Histogramm stammt von einem ungefilterten Chipbild, das untere vom selben Chip nach einer Tiefpaßfilterung. Im oberen Histogramm sind deutlich die beiden Farbbuckel der Zeichen- und Hintergrundpunkte zu sehen. Der grüne Strich markiert den vom Diskriminanzverfahren ermittelten Schwellwert, der eine einwandfreie Binarisierung ermöglicht. Damit durch die Pixelaggregation alle zusammenhängenden Zeichenpunkte auch als zusammenhängend erkannt werden, muß die erlaubte Schwankung der Helligkeitswerte eines Zeichens der Breite des Zeichenpunktbuckels entsprechen. Analog gilt dies für die Hintergrundpunkte, die ebenfalls als eine zusammenhängende Gruppe interpretiert werden sollen. Die Buckel sind jedoch breiter, als das Tal zwischen ihnen. Damit ist ein Auslaufen der Zeichenpunktmengen in den Helligkeitsbereich des Hintergrundes und umgekehrt, der Hintergrundpunkt-mengen in den Helligkeitsbereich der Zeichen, möglich. Dies soll verhindert werden, indem der Helligkeitsunterschied benachbarter Punkte begrenzt wird. Aufgrund der rauhen Chipoberfläche und den dementsprechend hohen Unterschieden der Helligkeit benachbarter Punkte ist eine solche Begrenzung jedoch nur eingeschränkt möglich. Der Hintergrund würde wieder in zahllose Einzelgruppen zerfallen. Andererseits ist das Tal zwischen den Buckeln nicht leer und die Helligkeitswerte zwischen den Buckeln befinden sich im Chipbild genau an den Grenzstellen zwischen Hintergrund- und Zeichenpunkten. Damit ist ein Übergang vom Hintergrund in den Helligkeitsbereich der Zeichen und umgekehrt möglich, auch wenn der Helligkeitsunterschied benachbarter Punkte stark eingegrenzt wird. Abschließend läßt sich feststellen, daß es nicht möglich ist, Parameter für die Pixelaggregation zu finden, die ein solches Bild richtig binarisieren.

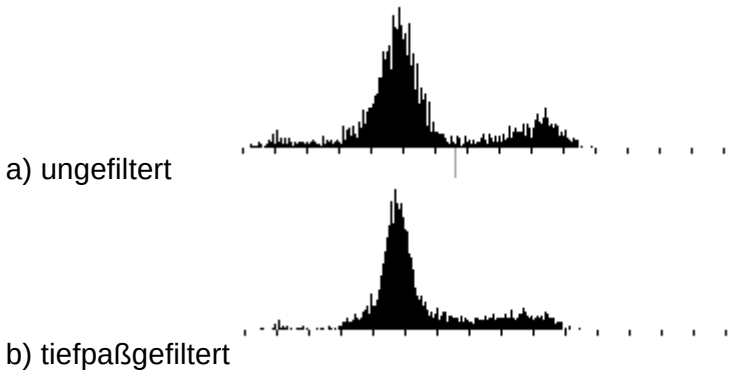


Abbildung 3.13: Histogramm eines a) ungefilterten und b) tiefpaßgefilterten Chipbildes.

Wie steht es nun mit einem tiefpaßgefilterten Bild? Das untere Histogramm in *Abbildung 3.13* stammt vom tiefpaßgefilterten Bild. In diesem Histogramm ist der linke Buckel, der den Hintergrundpunkten entspricht, deutlich schmaler geworden. Dies würde eine Reduzierung der Schwankungsbreite der Helligkeitswerte innerhalb einer Punktmenge erlauben, wenn dies ebenfalls für den Buckel der Zeichenpunkte gelten würde. Doch dieser ist breiter

geworden und dadurch fast verschwunden. Das Tal zwischen den Buckeln ist ebenfalls verschwunden und damit eine Grenze zwischen Zeichen- und Hintergrundhelligkeit. Es gibt damit auch keinen Wert für den maximalen Helligkeitsunterschied benachbarter Punkte, der erfolgreich die Punktmengen trennen könnte. Die Pixelaggregation scheitert auch an diesem Bild.

3.4.3 Invertierung

Nach der Binarisierung sollen Vordergrundpunkte schwarz und Hintergrundpunkte weiß sein. Hat das Ausgangsbild weiße Schrift auf schwarzem Hintergrund, muß das Bild invertiert werden. Problematisch ist nicht das Invertieren selber, sondern das Bestimmen ob ein Bild invertiert werden muß oder nicht. Das hierfür entwickelte Kriterium hat sich im Verlauf der praktischen Arbeit als zuverlässig und ausreichend gezeigt. Während der gesamten Entwicklung und Anwendung der Schrifterkennung wurde richtig erkannt, ob eine Invertierung nötig war oder nicht.

Die Ergebnisse dieses Abschnitts lassen sich folgendermaßen zusammenfassen. Der Schwellwert für die Binarisierung wird mittels des Diskriminanz-Verfahrens im ungefilterten Bild berechnet. Anschließend wird das gefilterte Bild binarisiert und falls nötig invertiert. Das Kriterium für die Entscheidung, ob zu invertieren ist, hat sich als zuverläßig erwiesen. Die Pixelaggregation ist für diese Anwendung ungeeignet. Die Binarisierung von gravierter Schrift ist ein ungelöstes Problem.

3.5 Segmentierung

Aufgabe der Segmentierung ist die Zuordnung der Vordergrundpunkte zu ihrem Kontext. Es werden Punkte zu Zeichen, Zeichen zu Worten und Worte zu Linien zusammengefaßt. Zusätzlich zum ursprünglichen Konzept werden Linien auch noch zu Blöcken zusammengefaßt. Dies war erforderlich, da sonst die Bestimmung der Achsenorientierung nicht immer funktionierte.

Die Segmentierung war schwieriger als erwartet. Die Kriterien von Scherl zur Zusammenfassung von Hypothesen konnten unverändert übernommen werden. Es mußten aber weitere Kriterien gefunden werden um Störungen und andere Eigenarten von Chiplayouts auszugleichen.

Es werden zwei Hypothesensysteme erstellt, ein vertikales und ein horizontales. Die Bildung von Wort-, Linien- und Blockhypothesen erfolgt jeweils für beide Achsenorientierungen. Bei der Bestimmung der Achsenorientierung werden die richtigen Hypothesen ausgewählt. Im folgenden sind der Übersichtlichkeit halber nur die Formeln und Schritte für die Bestimmung von horizontalen Hypothesen angegeben. Die Formeln für vertikale Hypothesen ergeben sich analog.
Am Ende dieses Kapitels werden die Ergebnisse der wichtigsten Segmentierungsschritte an einem Beispiel gezeigt.

Eingangs sollen noch einige grundsätzliche Definitionen gegeben werden. Die Höhe eines horizontalen Zeichens z ist $h_z=Y_{2z}-Y_{1z}+1$. Es gilt immer $Y_2 \geq Y_1$. Da Y_2 auch gleich Y_1 sein kann, muß eins addiert werden, denn ein Zeichen der Höhe null macht keinen Sinn. Analog wird die Breite eines horizontalen Zeichens z definiert als $b_z = X_{2z}-X_{1z}+1$. Für vertikale Zeichen ist die Definition genau umgekehrt. D. h. die Breite eines horizontalen Zeichens ist die Höhe eines vertikalen Zeichens.
Die Höhe h_w eines Wortes w ist der Mittelwert der Höhen der Zeichen des Wortes w. Die Breite b_w eines Wortes w ist hingegen $b_w=X_{2max}-X_{1min}+1$ mit X_{2max} dem größten X_2 aller im Wort enthaltenen Zeichen und X_{1min} dem kleinsten X_1 aller Zeichen im Wort. Höhe und Breite einer Linie sind analog zur Breite eines Wortes definiert. Die Höhe ist also die Gesamthöhe der Worte und nicht die mittlere Höhe.
Der relative horizontale Abstand zweier Rechtecke i und j ist definiert als:

$$rhA(i, j) = \max(X_{1i} - X_{2j}, X_{1j} - X_{2j}) / \max(h_i, hj)$$

bzw. sollte dies negativ sein als:

$$rhA(i, j) = (\max(X_{1i} - X_{2j}, X_{1j} - X_{2j})-1) / \min(b_i, b_j)$$

Der Wertebereich wird eingeschränkt auf $rhA(i, j) \geq -1$. Der zweite Teil der Formel tritt in Kraft, wenn die Rechtecke sich schon überlappen.
Die Formel für den relativen vertikalen Abstand rvA ergibt sich aus obiger Formel, indem X durch Y ersetzt wird und Breiten und Höhen vertauscht werden.
Die relative Überlappung $rÜ(i, j)$ zweier Rechtecke i und j ist:

$$rÜ(i, j) = -1*rA(i, j)$$

Die Definitionen werden des öfteren in den folgenden Abschnitten über die Schritte der Segmentierung gebraucht.

3.5.1 Punkte zu Zeichenhypothesen

Der von Gonzalez und Woods in [GoWo92] beschriebene Algorithmus zur Berechnung von Kettenkodes wird verwendet, um den Umriß von zusammenhängenden Punktmengen zu bestimmen. Kettenkodes haben den Vorteil, daß die gefundenen Mengen einfach und anschaulich sichtbar gemacht werden können.
Aus den Kettenkodes kann sehr einfach das umschließende Rechteck bestimmt werden. Die Seiten des Rechtecks werden parallel zum Chipgehäuse berechnet. Die Eckpunkte des Rechteckes ergeben sich aus den Minima und Maxima aller auftretender X- und Y-Koordinaten der Umrißpunkte.
Rand- und Kleinkettenkodes werden gelöscht. Ein Kettenkode, der kürzer als 17 Schritte ist, hat eine maximale Fläche von $4x4$ Punkten. Dies ist eindeutig zu klein, um ein Zeichen zu sein, bzw. um daraus noch ein Zeichen zu erkennen. In der Regel handelt es sich bei solchen Ketten um Störungen im Hintergrund. Selten sind dies Bruchstücke von Zeichen, die hierdurch verloren gehen. Die Bruchstücke sind aber so klein, daß das zugehörige Zeichen auch so erkannt werden kann. Problematisch wird es nur, wenn ein Zeichen nur aus solchen Bruchstücken besteht. Dann wird dieses Zeichen durch diesen Schritt komplett gelöscht. Dies ist vertretbar, da die nachfolgenden Schritte nicht in der Lage sind, solch ein Zeichen wieder richtig zusammenzusetzen.

Ketten, die den Rand berühren, können nur dann zu Zeichen gehören, wenn die an das Schrifterkennungssystem übergebene Chipoberfläche nicht ganz korrekt war. Normalerweise wird kein Chip direkt am Rand ohne jeglichen Abstand bedruckt. Ketten, die den Rand berühren, sind entweder normale Hintergrundstörungen, Gehäuseeinbuchtungen oder Chipbeinchen. Chipbeinchen können nur im Bildausschnitt liegen, wenn wiederum die übergebene Chipfläche falsch war. Diese Störungen werden durch das Löschen der Randkettenkodes behoben. Bei einigen Chiptypen liegen die Gehäuseeinbuchtungen direkt am Rand. Zwar werden diese selten gefunden, da sie noch dunkler sind als der Hintergrund und der Vordergrund meistens heller ist als der Hintergrund. Werden sie aber trotzdem gefunden, so werden sie hier sofort wieder gelöscht.

Von alle Kettenkodes, die jetzt noch übrig sind, wird erstmal angenommen, daß sie Zeichen sind. Aus jedem Kettenkode wird eine Zeichenhypothese, die durch das umschließende Rechteck charakterisiert ist.
Sich vollständig überlappende Zeichenhypothesen werden zu einer Hypothese zusammengefaßt. Viele Zeichen haben geschlossene Innenränder, für diese werden eigene Kettenkodes gefunden. Die Ziffer 8 besteht z.B. aus drei Kettenkodes, einem äußerem und zwei für Ränder der Innenkreise. Die umschließenden Rechtecke der Ketten der Innenkreise liegen vollständig innerhalb des umschließenden Rechtecks des Außenrandes. Da es sich aber nur um ein Zeichen handelt, müssen diese Ketten zusammengefaßt werden. Zwei Ketten werden zu einer Zeichenhypothese zusammengefaßt, wenn sie sich sowohl in horizontaler wie vertikaler Richtung um mindestens 7/8 überlappen. Das bedeutet, daß pro 8 Punkte Höhe ein Punkt außerhalb des jeweils anderen Rechtecks liegen darf. Dieses Verhältnis hat sich empirisch bewährt, da Zeichen häufig mindestens 8 Punkte breit oder hoch sind und eine Abweichung von 1 Punkt regelmäßig auftritt, sogar normal ist, denn hier ist ja die Auflösungsgrenze der Kamera erreicht worden. Sind die Rechtecke jedoch kleiner als 8 Punkte, ist es fraglich, ob es überhaupt Zeichen sind, dann wird eine 100% Überlappung gefordert.

3.5.2 Zeichenhypothesen zu Worthypothesen

Das Zusammenfassen von Zeichenhypothesen zu Worthypothesen erfolgt genau entsprechend den Regeln, die von Scherl aufgestellt wurden und die auch schon im Kapitel 2.5 beschrieben wurden. Der Übersicht halber sind die Regeln hier noch mal aufgeführt:

1. Höhenschwankung: $0{,}55 < h_z / h_w < 1{,}8$
2. relativer horizontaler Abstand: $\mathrm{rhA}(z,w) < 0{,}5$
3. relative vertikale Überlappung: $0{,}35 < \mathrm{rvÜ}(z,w) < 1{,}0$

Die Formeln sind dabei nur für horizontale Worthypothesen angegeben. Für vertikale Worthypothesen wird der vertikale Abstand und die horizontale Überlappung verwendet.
Erfüllt eine Zeichenhypothese z zusammen mit einer Worthypothese w alle drei Bedingungen, wird z in w eingefügt. Findet sich für eine Zeichenhypo-

these z keine passende Worthypothese, wird ein neue Worthypothese erzeugt, in welche z eingefügt wird.
Wurde so jede Zeichenhypothese einer Worthypothese zugeordnet werden sich vollständig überlappende Worthypothesen zusammengefaßt. Der Punkt eines kleinen „i“ und andere Teile von Zeichen, die aus mehreren Teilen bestehen oder in solche zerfallen sind, werden beim Finden der Wörter keinem Wort zugeordnet, sondern wird zu einem eigenen Wort. Dies liegt daran, daß die Schwankung der Höhen zwischen dem restlichen Wort und dem I-Punkt zu groß ist. Das Rechteck des Wortes des I-Punktes liegt dann aber vollständig im Rechteck des Wortes, in dem der Strich des I’s und die umliegenden Zeichen zusammengefaßt wurden. Daher werden in diesem Schritt alle Worte auf vollständige Überlappung in beide Richtungen überprüft. Tritt solch ein Fall auf, werden alle Zeichen des zweiten Wortes in das erste Wort eingefügt und das zweite Wort gelöscht.

*Abbildung 3.14:*Die zerfallene 2 (letztes Zeichen der oberen Zeile) wird zusammengesetzt.

Durch den vorhergehenden Schritt sollten alle Teile eines Zeichens dem selben Wort zugeordnet worden sein. Nun muß aus allen Teilen eine einzige Zeichenhypothese erstellt werden. Dazu werden Zeichenhypothesen, die zu dem gleichen Wort gehören, vereint, wenn sie übereinander oder ineinander liegen. Übereinander liegen zwei horizontale Hypothesen, wenn ihre relative horizontale Überlappung größer 7/8 ist und ihr relativer vertikaler Abstand kleiner 0,6 ist. Ineinander liegen zwei horizontale Hypothesen, wenn ihre relative horizontale Überlappung größer 0,5 und ihre relative vertikale Überlappung größer 0,2 sind. In ungefähr fünf Prozent der Worthypothesen tritt der erste Fall auf. Vom zweiten Fall sind ungefähr vier Prozent betroffen. Die angegebenen Grenzwerte wurden durch meine Erwartungen zur Struktur von Zeichen festgelegt und haben sich bisher bewährt. Die *Abbildung 3.14* zeigt das Zusammensetzen eines zerfallenen Zeichens mit den obigen Regeln. Das linke Bild zeigt die Zeichenhypothesen vor der Wortbildung, das Rechte die Zeichenhypothesen nach der Wortbildung und der Kontrolle der Zeichenhypothesen in jeder Worthypothese gemäß den obigen Regeln. Im linken Bild wird auch deutlich, daß sich die Zeichenteile nicht hundertprozentig überlappen, so daß ein Grenzwert kleiner 1 (hier 7/8) für die relative horizontale Überlappung erforderlich ist.

Eine spezielle Regel ist nötig, um mehrteilige Firmenlogos zu einem Zeichen zusammenzufassen. Ein Firmenlogo ist in der Regel ein einzelnes Zeichen, und es bleibt auch das einzige Zeichen in einen Wort. Besteht ein Firmenlogo jedoch aus mehreren übereinander liegenden unverbundenen Teilen, werden mehrere Wörter mit jeweils einer Zeichenhypothese gefunden. Diese sollten zusammengefaßt werden, um das Logo anschließend erkennen zu können. *Abbildung 3.15* zeigt zwei Beispiele.

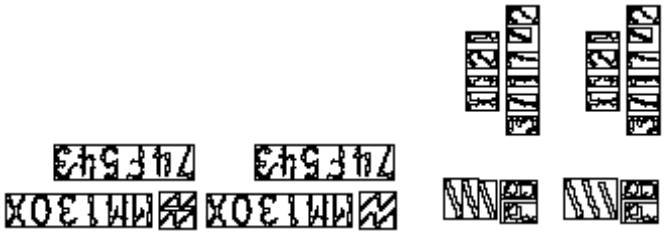


Abbildung 3.15: Zwei mehrteilige Logos, die zusammengefaßt werden.

Es werden zwei Worthypothesen zu einer zusammengefaßt, wenn beide nur aus einer Zeichenhypothese bestehen, der relative vertikale Abstand kleiner 0,5 ist, die relative horizontale Überlappung größer 0,5 und die Differenz der Breiten beider Wörter zwischen 0,75 und 1/0,75 liegt. Diese Regeln beinhalten, daß die Wörter ungefähr gleich breit sind, sie übereinander liegen mit einem Abstand von maximal der halben Worthöhe und sich ihre Breitseiten mindestens zur Hälfte überlappen. Nicht nur die beiden Worthypothesen werden zu einer zusammengefaßt, sondern auch die Zeichenhypothesen, denn es liegt nur ein Logo vor.
Diese Regel ist teilweise zu restriktiv, und nicht immer werden alle Logos zusammengefaßt. Sie muß jedoch sehr restriktiv sein, um zu verhindern, daß Zeichen in übereinander liegenden Zeilen auch zusammengefaßt werden. Diese Gefahr besteht immer dann, wenn alle Zeichen zweier übereinander liegender Wörter verklebt sind. Dann sind alle Zeichen jedes Wortes in nur einer Zeichenhypothese enthalten und die Worte bestehen dann auch nur aus je einer Zeichenhypothese, siehe z. B. *Abbildung 3.19* auf Seite 65. Für diese Fälle verhindern die restriktiven Forderungen bezüglich Breitenverhältnis und Überlappung das Zusammenfassen dieser Hypothesen.
Es ist vertretbar, daß diese Regel nicht immer zum Erfolg führt, da die Firmenlogos nicht unbedingt gebraucht werden zur Typbestimmung eines Chips. Die Regel hat ihren Sinn eher darin, in den meisten Fällen Klassifikationsversuche zu verhindern, die scheitern müssen, da kein vollständiges Logo vorliegt. Da in solchen Fällen nach Trennstellen gesucht wird und dann erneute Klassifikationsversuche gestartet werden, sind diese Versuche sehr zeitaufwendig und müssen dennoch immer erfolglos bleiben, da kein vollständiges Logo vorliegt.

Alternativ zu den oben beschrieben Regeln zur Wortbildung wurde zwischenzeitlich auch mit einem optimistischen, weniger restriktiven Ansatz gearbeitet.
Anstatt den Höhenunterschied zu begrenzen, den die Zeichen in einem Wort maximal haben dürfen, wurde zuerst keine Höhenbegrenzung für das Zusammensetzen von Wörtern verwendet. In einem nachfolgenden Schritt sollten dann die entstandenen Wörter untersucht werden. Falls das entstandene Wort unwahrscheinlich war, sollte es in zwei Worte zerlegt werden. Der Vorteil dieses Vorgehens ist, daß Satzzeichen dem richtigen Wort zugeordnet werden, während sie bei dem pessimistischeren Ansatz mit Höhenbeschränkung erstmal ausgeschlossen sind. Das Problem bei diesem Schritt war die Definition von unwahrscheinlich und das Finden einer geeigneten Stelle, um das Wort zu zerlegen.

Die Idee war ein Wort dann als unwahrscheinlich anzusehen, wenn viele Zeichenhypothesen in einem Wort übereinander lagen anstatt nebeneinander, Wieviele Zeichenhypothesen sind aber viele? Sind Zwei schon viele? Die *Abbildung 3.19* zeigt einen Chip, auf dem nur drei Zeichenhypothesen gefunden wurden. Bei Verwendung der optimistischen Regeln zum erstellen von Worten, werden im vertikalen Fall alle drei Zeichen zu einem Wort zusammengefaßt. Die beiden verklebten Zeichenketten liegen dann übereinander. In diesem Fall wäre Zwei bereits viel. In anderen Fällen aber nicht.
Insgesamt gelang es nicht ein stabiles Regelsystem zum Erkennen und Zerlegen von falsch zusammengesetzten Worten aufzustellen. Zu viele unterschiedliche Fälle traten auf, die eigene Spezialregen benötigten, gleichzeitig waren die einzelnen Fälle zu selten, um die Regelparameter auf empirische Daten abstützen zu können.
Dieser Ansatz wurde daher fallengelassen. Letztendlich hat sich sein Fehlen auch selten negativ bemerkbar gemacht. Bemerkbar wird das Weglassen dieses Ansatzes nur auf einigen Chips, bei diesen werden Satzzeichen keinem Wort zugeordnet. Wie ich meine ein tollerierbarer Fehler, der in *Abbildung 3.17* konkret zu sehen ist. Das Minuszeichen wird im horizontalen System keinem Wort zugeordnet. Es bleibt am Ende als eigener Block übrig. Ähnliches passiert auch im vertikalen Fall. Das „l“ in der letzten, horizontalen Zeile wird nicht dem richtigen Wort zugeordnet. Eine weitere zusätzliche Regel könnte diese Fehler sicher noch nachträglich beheben.

3.5.3 Worthypothesen zu Linienhypothesen

Das Zusammenfassen von Worthypothesen zu Linienhypothesen erfolgt nach ähnlichen Regeln, wie das Zusammenfassen von Zeichenhypothesen zu Worthypothesen. Die Abstände zwischen den Worthypothesen dürfen größer sein, als zwischen Zeichenhypothesen, die Höhen der Worthypothesen müssen jedoch ähnlicher, die Überlappung stärker sein. Folgende Bedingungen müssen erfüllt sein:

1. Höhenschwankung: $0{,}77 < h_l / h_w < 1{,}3$
2. relativer horizontaler Abstand: $rhA(w,l) < 2{,}5$
3. relative vertikale Überlappung: $0{,}5 < rvÜ(w,l) < 1{,}0$

Erfüllt eine Worthypothese w zusammen mit einer Linienhypothese l alle drei Bedingungen, wird w in l eingefügt. Findet sich für eine Worthypothese w keine passende Linienhypothese, wird eine neue Linienhypothese erzeugt, in welche w eingefügt wird.

3.5.4 Linienhypothesen zu Blockhypothesen

Die Zuordnung der Wörter zu Linien, erwies sich als nicht ausreichend, um zuverlässig entscheiden zu können, ob eine vertikale oder horizontale Beschriftung vorliegt. Ein wichtiges Kriterium bei der Beurteilung der Achsenorientierung ist die Regelmäßigkeit der Abstände der Linien. Diese

Regelmäßigkeit wird jedoch zerstört, wenn große Logos, die sich neben den Textzeilen befinden, wie normale Linien behandelt werden. *Abbildung 3.16* verdeutlicht dies. Durch die Aufteilung in Blöcke sind die Abstände der Linien mit 4, 5, 5, 5 und 5 sehr einheitlich. Würde auf Blöcke verzichtet, ergäben sich die Abstände 4, 5, -1, -19, 5 und 5; die Einheitlichkeit wäre dahin. Ein anderes Beispiel für die Notwendigkeit der Blöcke zeigt die *Abbildung 3.17*. Ohne die Blöcke, würde dieser Chip nicht mehr als horizontal orientiert erkannt.

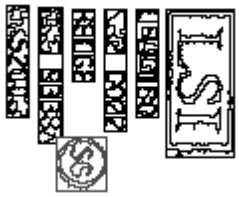


Abbildung 3.16: Das umkreiste SG wird als eigener Block eingestuft. Der restliche Text ist ebenfalls ein Block.

Es werden daher übereinander liegende Linien zu einem Block zusammengefaßt. Nebeneinander liegende Linien kommen in verschiedene Blöcke. eine Linie gehört zu einem Block, wenn sich die Breitseiten von Block und Linie fast komplett überlappen, wenn der absolute Abstand von der Linie zum Block kleiner als die doppelte Linienhöhe ist und die Linie sich mit keiner anderen Linie im Block um mehr als eine halbe Linienhöhe überlappt. Formal gehört eine horizontale Linie zu einem Block, wenn:

1. Überlappung : $rhÜ(l,b) > 0{,}9$
2. Abstand: $rvA(l,b)/ \max(h_l, h_b) * h_l < 2$
3. Überlappung: $\forall\, l' \ni b$ gilt $rvÜ(l, l') < 0{,}5$

Jeder Block enthält genau ein Liniensystem, in dem beliebig viele Linien enthalten sein können. Ein Blocksystem ist die Menge von Blöcken, die für eine Achsenorientierung zusammengestellt wurde. Welche Achsenorientierung richtig ist, wird über die Wahrscheinlichkeit des zugehörigen Blocksystems entschieden.

Die meisten Konstanten der Regeln zur Segmentierung wurden intuitiv gewählt und falls nötig von Hand angepaßt, bis die erzielten Ergebnisse zufriedenstellend waren. Eine Kritik dieses Vorgehens und mögliche Alternativen werden im Abschnitts 3.6.1.8 „Kritik des gewählten Ansatzes“ erörtert.

Die *Abbildung 3.17* zeigt die Ergebnisse der wichtigsten Schritte zur Segmentierung an einem Chip. Es werden sowohl die horizontalen als auch die vertikalen Hypothesen dargestellt. Die entstandenen vertikalen Linien ergeben durchaus ein sinnvolles Ganzes. Das „S“ und auch besonders der vertikale Strich passen sehr gut zu diesen Linien. Die Linien sind alle deutlich länger als hoch, der Abstand zwischen den Linien ist ungefähr gleich und die

Linien sind ungefähr alle gleich hoch. Damit sollte deutlich werden wie schwierig es ist die Achsenorientierung eines Chips aus den ermittelten Blöcken, Linien und Wörtern zu bestimmen. Wie dies trotzdem gelingt, wird im folgenden Abschnitt erläutert.

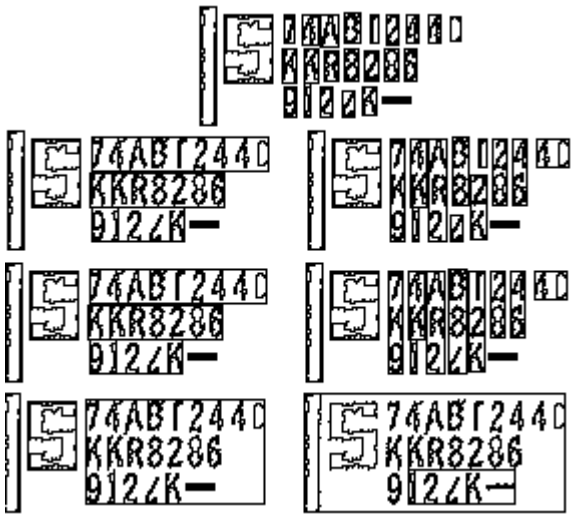


Abbildung 3.17: Beispiel einer Segmentierung. Im obersten Bild die Zeichenhypothesen, dann jeweils links die horizontale und rechts die vertikale Version der Wort-, Linien-, und Blockhypothesen, jeweils durch das umschließende Rechteck dargestellt. Das Zusammenfassen der „4“ und des „D“ bei den vertikalen Wörtern kommt durch die Regel zum Zusammenfassen von Firmenlogos. Der kleinere der beiden vertikalen Blöcke besteht nur aus dem „I“ und dem Minus. Da das umschließende Rechteck gezeichnet wird, entsteht der Eindruck, daß auch die dazwischen liegenden Zeichen „22K“ in diesem Block sind, betrachtet man die Wort-hypothesen, wird deutlich, daß diese Zeichen nicht zu dem Block gehören können. Bei den horizontalen Blöcken, sind der Strich, das „S“ und ins-besondere auch das Minus eigene Blöcke!

3.6 Bestimmung der Orientierung

In den vorausgegangenen Schritten, wurden alle gefundenen, zusammenhängenden Punktmengen, ähnlich einem Puzzle, zu Zeichen-, Wort-, Linien- und Blockhypothesen zusammengefaßt. Insgesamt wurden jedoch zwei Lösungen für das Puzzlespiel berechnet: eine mit horizontaler und eine mit vertikaler Beschriftung. Die richtige Lösung und damit die Achsenorientierung wird anhand von Wahrscheinlichkeiten bestimmt. Anschließend wird die Richtungsorientierung bestimmt.

3.6.1 Achsenorientierung

Zwei Hypothesensysteme wurden erstellt. Um das Richtige zu finden, wird jeder Hypothese eine Wahrscheinlichkeit zugeordnet. Die Verknüpfung dieser Wahrscheinlichkeiten ergibt die Gesamtwahrscheinlichkeit jeder Lösung. Dabei handelt es sich nicht um echte Wahrscheinlichkeiten, denn es werden auch Werte größer 1 vergeben, um besonders schöne Hypothesen mit einem Bonus zu versehen und die Wahrscheinlichkeiten nicht endlos klein werden

zu lassen. Die wahrscheinlichere Lösung wird als die Richtige angesehen. Die Kriterien, die sich für die Berechnung der Wahrscheinlichkeit der verschieden Hypothesenarten bewährt haben, werden in den folgenden Abschnitten beschrieben. Die anfänglichen Formeln waren größtenteils unzureichend, die Konstanten in den Formen mußten teilweise mühselig optimiert werden.

3.6.1.1 Wahrscheinlichkeit einer Zeichenhypothese

Im ursprünglichen Konzept sollten alle Zeichenhypothesen gleich wahrscheinlich sein und den Wahrscheinlichkeitswert 1 erhalten. In praktischen Versuchen wurden aber folgende sinnvollen Kriterien gefunden: Zeichenhypothesen mit so geringer Höhe, daß keine Erkennung mehr möglich ist werden nur mit Wahrscheinlichkeit 0,5 bewertet. Eine Erkennung wird ausgeschlossen, wenn ein Zeichen keine 5 Punkte hoch ist. Dieses Kriterium wertet um 90 Grad verdrehte I's ab.
Alle anderen Zeichen erhalten eine Wahrscheinlichkeit $P = Breite / Höhe + 0{,}7$. P wird noch eingeschränkt auf einen Wertebereich von $0{,}7 - 1$. Dieses Kriterium bestraft Zeichenhypothesen, die mindestens dreimal höher als breit sind. Hierdurch sind, fälschlicher Weise, I's betroffen. Gute Dienste leistet das Kriterium, aber bei der Abwertung von Hypothesen, die verklebte Zeichen enthalten und verdreht betrachtet werden. Ansonsten ist es neutral und liefert für beide möglichen Achsenorientierungen meistens den Wert 1.

3.6.1.2 Wahrscheinlichkeit einer Worthypothese

Anfänglich wurde die Wahrscheinlichkeit eines Wortes abhängig von der Varianz der Zeichenabstände, der Varianz der Zeichenhöhen und der Anzahl der Zeichenhypothesen berechnet. Die Anzahl der Zeichenhypothesen erwies sich aber als nicht aussagekräftig. Oft verkleben alle Zeichen eines Wortes, das Wort besteht dann nur aus einer Zeichenhypothese. Anstatt mit der Anzahl der Zeichenhypothesen, wird daher mit dem Verhältnis zwischen Breite und Höhe gearbeitet. Je Breiter ein Wort, desto wahrscheinlicher ist es auch. Auch die Varianz der Abstände der Zeichen in einem Wort ist ein unbrauchbares Kriterium. Die Abstandsstreuung ist oft geringer in der falschen Achsenorientierung, als in der Richtigen. In der falschen Achsenorientierung, wird der Zeilenabstand als Zeichenabstand angesehen. Dieser ist oft größer und damit exakter und einheitlicher zu bestimmen. Der Abstand wird daher nicht verwendet. Die Formel für die Höhenstreuung konnte fast so beibehalten werden. Sie wurde aber noch um einen Divisor ergänzt, da sonst Worte mit vielen Zeichenhypothesen gegenüber Worten mit sehr wenigen benachteiligt waren. Getreu dem Motto: Ausnahmen bestätigen die Regel, tanzt bei einem Wort mit vielen Zeichenhypothesen eine Zeichenhypothese meistens aus der Reihe. Die Varianz ist daher bei Wörtern mit vielen Zeichenhypothesen häufig höher, als bei Wörtern mit wenigen Zeichenhypothesen. Durch die Division der Varianz durch die Anzahl der Hypothesen wird die Varianz entsprechend relativiert. Dieser Trend wurde auch bei allen anderen Anwendungen der Varianz beobachtet und durch diese Division korrigiert. Eine weitere Änderung an der Formel für die Höhenstreuung war die Erhöhung des Maximalwert von 1,0 auf 1,1.

Höhenstreuung = 1,1 - (Varianz(Zeichenhöhen) / 10 / Anzahl(Zeichen)); mindestens 0,3
Breiten-Höhen-Verhältnis = Breite / Höhe; mindestens 0,80; maximal 1,25

P(Worthypothese) = Höhenstreuung
* Breiten-Höhen-Verhältnis
*Mittelwert(Zeichenhypothesenwahrscheinlichkeit)

3.6.1.3 Wahrscheinlichkeit einer Linienhypothese

Die Höhenstreuung bei Linien wurde genauso abgewandelt wie bei Worten. Während bei Worten die Abstandsstreuung erfolglos war, ist sie hier, aufgrund der größeren Distanz zwischen Worten als zwischen Zeichen, anwendbar. Sie wurde aber ebenfalls um die Anzahl der Worthypothesen als zusätzlichen Divisor ergänzt und der Maximalwert von 1,0 auf 1,25 angehoben.
Die Anzahl der Worthypothesen als Kriterium wurde, wie bei Worten, durch das Breiten-Höhen-Verhältnis ersetzt. Es wird für Linien aber ein noch krasseres Verhältnis erwartet, als für Worte.

Höhenstreuung = 1,10 - (Varianz(Worthöhen) / 10 / Anzahl(Worthypothesen)); mindestens 0,3
Abstandsstreuung = 1,25 - (Varianz(Wortabstände) / 10 / Anzahl(Worthypothesen)); mindestens 0,8; maximal 1,0
Breiten-Höhen-Verhältnis = Breite / Höhe /3 ; mindestens 0,80; maximal 1,25

P(Liniehypothese) = Höhenstreuung
* Abstandsstreuung
* Breiten-Höhen-Verhältnis
* Mittelwert(Worthypothesenwahrscheinlichkeiten)

3.6.1.4 Wahrscheinlichkeit eines Liniensystems

Die ursprüngliche Formel wurde analog zu den anderen Wahrscheinlichkeiten angepaßt. Der Maximalwert für die Abstandsstreuung wurde auf 1.25 erhöht. Die Varianz wurde durch die Anzahl der Linien dividiert.

Abstandsstreuung = 1,25 - (Varianz(Linienabstände) / 10 / Anzahl(Linien)); mindestens 0,8
P(Liniensystem) = Abstandsstreuung
* Mittelwert(Linienhypothesenwahrscheinlichkeit)

3.6.1.5 Wahrscheinlichkeit eines Blocks

Blöcke waren im ursprünglichen Konzept nicht vorgesehen. Sie wurden eingeführt, um die Störung eines Liniensystems durch neben den Textzeilen befindliche Logos zu korrigieren. Solche Logos führten sonst zu riesigen Abstandsstreuung in Liniensystemen und folglich zu Fehlentscheidungen. Die

Wahrscheinlichkeit eines Blockes selber ist die Wahrscheinlichkeit des in ihm enthaltenen Liniensystems.

3.6.1.6 Wahrscheinlichkeit eines Blocksystems

Die Wahrscheinlichkeit eines Blocksystems ist:

P(Blocksystem) = Mittelwert(Blockwahrscheinlichkeiten))

Die Wahrscheinlichkeit des Blocksystems entscheidet darüber, welche Achsenorientierung für richtig erachtet wird. Bisher konnte bei allen untersuchten Chips mit diesen Formeln für die Wahrscheinlichkeiten die Achsenorientierung richtig bestimmt werden. Die Konstanten in den Formeln wurden anfänglich intuitiv gewählt. Dort wo die anfänglichen Werte falsche Ergebnisse lieferten, wurden die Werte empirisch angepaßt. Die Vielzahl der Formeln und ihrer Konstanten macht deutlich, daß die Bestimmung der Achsenorientierung der Beschriftung eines Mikrochips recht komplex ist. Aufgrund der Vielzahl der Formeln, ist der Einfluß der einzelnen Formeln für das Gesamtergebnis auch nicht mehr überschaubar. Dies erschwert insbesondere die Optimierung der Parameter.

3.6.1.7 Wahrscheinlichkeitslose Alternative

Probehalber wurde untersucht, ob auch die Bestimmung der Achsenorientierung mittels einfachen Ausprobierens sicher möglich ist. Es wird also keine Wahrscheinlichkeit für die beiden ermittelten Hypothesensysteme berechnet, sondern für alle vier Richtungen wird versucht, die Zeichenhypothesen aus dem jeweiligen Hypothesensystem zu klassifizieren. Das Kriterium für die Akzeptanz einer Orientierung bleibt dabei das gleiche wie oben beschrieben. Es müssen drei Zeichen mehr in einer Richtung gefunden werden als in alle anderen. Auch auf diese Art kann die Orientierung sicher ermittelt werden. Zwar kommt es öfters auch zu Fehlklassifikationen von verklebten Zeichen, dies ist aber eher auf ein unzureichendes Zeichenerkennungsverfahren oder unzureichendes antrainieren zurückzuführen. Der Methode zur Bestimmung der Orientierung können diese Misklassifikationen nicht angelastet werden. So werden verklebte Zeichen, die um 90 Grad gedreht und sehr viel höher als breit sind, häufig als I erkannt. Ein anderes Beispiel von Misklassifikation zeigt *Abbildung 3.18*.

Abbildung 3.18: Verklebte Zeichenfolge AA2. Durch die Drehung um 90 Grad und die Größennormierung auf 8x8 entsteht ein E.

Der Nachteil des wahrscheinlichkeitslosen Vorgehens ist der höhere Zeitaufwand. Die gesamte Bearbeitung eines Chips dauert zwischen 10 und 20 Sekunden, bei Verwendung einer 8x8 großen Grauwertmatrix als Klassifikationsmerkmale, einem *3-Nearest-Neighbour*-Klassifikator und einer angemessenen großen Menge von Lernzeichen. Die Berechnung der

Wahrscheinlichkeiten dauert eine zehntel Sekunde, die Bestimmung der Orientierung durch Klassifikation bei zwei Orientierungen je nach Chip zwischen 1,5 und 3 Sekunden, bei vier Orientierungen doppelt so lange. Die Bearbeitungszeit eines Chips erhöht sich damit um ca. 15%.
Die Zuhilfenahme von Wahrscheinlichkeiten ist damit deutlich vorteilhaft, wenn Geschwindigkeit eine Rolle spielt. Der Nachteil von Wahrscheinlichkeiten ist der einmalige vorhergehende Aufwand zur Bestimmung von Formeln um die Wahrscheinlichkeiten zu ermitteln.

3.6.1.8 Kritik des gewählten Ansatzes

An die Segmentierung und die Bestimmung der Achsenorientierung bin ich mit der Erwartung herangetreten, daß sie größtenteils mit den Kriterien und Ideen von Scherl [Sche87] und Consorti [CCLT94] bereits gelöst ist. Ich wurde eines besseren belehrt und mußte viel Zeit investieren, bis Segmentierung und Bestimmung der Achsenorientierung einwandfrei für beliebige Chips funktionierten. Immer wieder erforderten neue Chips neue Regeln oder Regeländerungen. Die Anpassung der Konstanten und Regeln an neue Chips war mühevolle Handarbeit und erforderten eine ausgiebige erneute Überprüfung aller anderen Chips. Dieser Lösungsansatz ist daher in der Realisierung sehr aufwendig. Aber, er funktioniert zuverlässig, und die Segmentierung und die Bestimmung der Achsenorientierung sind sehr schnell.
Das Bestimmen der Parameter von Hand ist zu aufwendig und kann nicht zur Nachahmung empfohlen werden. Besser wäre eine automatische Bestimmung der Parameter mittels Verfahren des maschinellen Lernens. Damit wäre eine Anpassung der Segmentierung an andere Anwendungen auch ohne Probleme möglich.
Eine andere Lösung wären alternative Ansätzen zur Behandlung des Problems, die ohne viele Parameter auskommen. In Kapitel 2.5 wurde ein, von Fletcher und Kasturi in [FlKa88] verwendeter, anderer Ansatz basierend auf der *Hough-Transformation* diskutiert. Auch nach meinen Erfahrungen mit der Segmentierung bin ich weiterhin überzeugt, daß dieser Ansatz für meine Anwendung nicht besser geeignet ist, als der verwendete. Insbesondere benötigt der Ansatz mit der Hough-Transformation größtenteils unverklebte Zeichen. Solche liegen aber bei einem Großteil der Chips nicht vor. Für alle diese Chips, wäre der Ansatz mit der Hough-Transformation daher nicht anwendbar. *Abbildung 3.19* zeigt einen Chip in dem nur drei Zeichenhypothesen gefunden wurden. Hier kann mit der Hough-Transformation keine Aussage über die Achsenorientierung gewonnen werden. Es müßten andere Regeln gefunden werden. Wenn jedoch Regeln benötigt werden, können diese gleich für alle auftretenden Fälle entwickelt werden. Der Einsatz der Hough-Transformation für einige Chips mit großer Schrift und der Einsatz von Regeln für alle anderen Chips verkompliziert das Problem nur. Es muß dann ein Kriterium gefunden werden, wann die Ergebnisse der Hough-Transformation zuverlässig sind und wann auf die Regeln ausgewichen wird.

Abbildung 3.19: Chip mit nur drei Zeichenhypothesen aufgrund massiver Verklebungen. Die Ermittlung der Achsenorientierung alleine mittels der Hough-Transformation ist unmöglich.

Das Vorziehen der Zeichenklassifikation vor die Segmentierung, wäre eine weitere Möglichkeit. Es würden also nur Zeichenhypothesen erzeugt und diese dann klassifiziert. Erst danach würden Zeichen zu Worten und Linien zusammengefaßt. Die Segmentierung würde hierdurch entsprechend vereinfacht, da Orientierung und Bedeutung jedes Zeichens bereits bekannt wären. Die Berechnung von Wahrscheinlichkeiten und die Formeln derselben könnten entfallen. Alle Störungen wären bereits durch die Klassifikation beseitigt worden. Regeln zum Zusammenfassen von mehreren Hypothesen zu einem Zeichen wäre aber weiterhin nötig, denn zerfallene Zeichen sollen klassifiziert werden. Kann erwartet werden, daß zuverlässig funktionierende Regeln für diese Aufgabe einfacher sind als Regeln zur Gruppierung von Zeichen zu Wörtern und Wörter zu Linien? Nach meinen Erfahrungen mit der Segmentierung, denke ich hier eher pessimistisch. Aufgrund der fehlenden Kontextinformation müssen bei diesem Ansatz sicher mehr Möglichkeiten der Zusammenfassung von Hypothesen probiert werden, als bei einem Ansatz mit vorausgehender Segmentierung. Gegen diesen Ansatz spricht daher der zusätzliche Zeitaufwand bei der Klassifikation. Während der Klassifikation müßten doppelt so viele mögliche Orientierungen getestet werden. Alleine dies würde schon mehr Zeit in Anspruch nehmen, als die gesamte momentane Segmentierung. So dauert die Segmentierung und die Bestimmung der Achsenorientierung momentan maximal eine Sekunde. Das anschließende Bestimmen der genauen Richtungsorientierung dauert eineinhalb bis zwei Sekunden. Alleine diese Zeit würde sich verdoppeln. Aber auch die restliche Klassifizierung würde deutlich länger dauern. Kann eine Hypothese nicht klassifiziert werden, müssen zeitaufwendige Trennversuche mit vier Orientierungen gemacht werden, führt dies nicht zum Erfolg müssen ebenso aufwendige Vereinigungsversuche gemacht werden. Bei meinem Ansatz reicht eine Orientierung aus. Das Vorziehen der Zeichenklassifikation vor die Segmentierung ist daher aus Geschwindigkeitsgründen nicht akzeptabel.

Abschließend halte ich weiterhin den gewählten Ansatz zur Segmentierung und Bestimmung der Achsenorientierung für den Besten. Die Bestimmung von Regeln und Konstanten sollte aber einem Verfahren des maschinellen Lernens überlassen werden und nicht von Hand angegangen werden. Aus meinen Erfahrungen weiß ich, daß die Hoffnung, hinter der aktuellen

Bergkuppe das Paradies zu finden, immer wieder von neuem trügt. Nach jeder Lösung eines Segmentierungsproblems, taucht nur das nächste auf. Das Paradies, die perfekte Lösung, bleibt immer unerreicht. Die Lösung der Segmentierung von Hand gleicht einer Sisyphusarbeit.
Ob Methoden des maschinellen Lernens aber genauso gute Ergebnisse erzielen wie mein gefundene Lösung, können erst praktische Versuche zeigen.

3.6.2 Richtungsorientierung

Bisher ist nur die Achsenorientierung bekannt, ob die Orientierung der Schrift horizontal oder vertikal ist. Es muß noch die Richtungsorientierung ermittelt werden. Der gewählte Ansatz besteht aus einfachem Ausprobieren beider Möglichkeiten. Anfänglich wurden alle Zeichenhypothesen getestet und zum Schluß die Richtungsorientierung akzeptiert, in der die meisten Zeichen erkannt wurden. Später wurde ein optimistischeres Kriterium verwendet, so daß eine Richtung als richtig akzeptiert wird, sobald in dieser Richtung drei Zeichen mehr erkannt wurden, als in der anderen Richtung. Wurden alle Zeichen klassifiziert ohne das eine Richtung einen solchen Vorsprung erreichte, wird die Richtung genommen, welche die meisten Zeichen erkannte. Im Extremfall, daß beide Richtungen gleich viele oder keine Zeichen erkannt haben, scheitert das Verfahren. Dieser Extremfall trat in der Praxis bisher nie auf. Er wäre auch eher das Produkt eines mangelhaften Zeichenklassifikators oder falscher Segmentierung. Eine Reduzierung des nötigen Vorsprungs von drei auf nur zwei Zeichen führt manchmal zu falschen Ergebnissen und ist daher nicht möglich.
Bei der Klassifizierung der Zeichenhypothesen zur Bestimmung der Richtungsorientierung wurden anfänglich nicht klassifizierbare Zeichenhypothesen als verklebte Zeichen angenommen und ein Trennungsversuch unternommen. Der Versuch der Trennung von Zeichenhypothesen zur Orientierungsbestimmung führt aber häufig zu falschen Ergebnissen. Viele Zeichen, die auf dem Kopf stehen, können durch das Trennen in mehrere richtig orientierte Zeichen zerlegt werden. *Abbildung 3.20* verdeutlicht diesen Effekt.

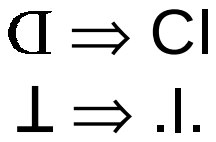


Abbildung 3.20: Zerlegung von kopfstehenden Zeichen in mehrere richtig orientierte Zeichen.

Auch bei mehreren verklebten kopfstehenden Zeichen sind solche Trennungen in richtig orientierte Zeichen möglich. Hierdurch werden in der kopfstehenden Richtung mehr Zeichen erkannt, als in der richtigen Richtung. Die Bestimmung der Orientierung schlägt fehl. Aus diesen Gründen wird auf eine Trennung von nicht klassifizierten Zeichen während der Orientierungsbestimmung verzichtet. Hiermit wird deutlich, wie wichtig es ist, daß bei der Binarisierung unverklebte Zeichen erhalten bleiben.

Denkbar wäre zwar eine Verfeinerung der Regel, wann getrennt wird. Z.B. nur, wenn in beide Richtungen eine Zeichenhypothese nicht klassifiziert werden konnte und nach einer Trennung wird nur das Trennungsergebnis akzeptiert, das aus weniger Zeichen besteht. Die Bestimmung der Richtungsorientierung alleine aus den ohne Trennung klassifizierten Zeichen erwies sich aber als ausreichend zuverlässig. Auf ein zeitintensives Trennen während der Bestimmung der Achsenorientierung wird daher ganz verzichtet.
Der verwendete Klassifikator misklassifizierte anfangs des öfteren kopfstehende Zeichen. So wurde  oft als  ;  als  oder  klassifiziert. Durch antrainieren des Klassifikators mit kopfstehenden Zeichen konnten solche Fehlklassifikationen unterbunden werden. Bei besseren Verfahren zur Zeichenerkennung kann möglicherweise auf ein solches Antrainieren verzichtet werden. Im meinen Fall konnte die Richtungsorientierung erst nach diesem zusätzlichem Training sicher ermittelt werden.

3.7 Zeichenklassifizierung

Die Zeichenklassifizierung läuft in zwei Schritten ab. Zuerst werden aus dem Zeichenbild Merkmale berechnet. Im zweiten Schritt werden diese Merkmale als Eingabe für Klassifikationsverfahren verwendet. Die Klassifikationsverfahren ordnen die Merkmale und damit das Zeichenbild einer Bedeutungsklasse zu.
In Abschnitt 3.7.1 werden die Erfahrungen mit den untersuchten Merkmalen geschildert. In Abschnitt 3.7.2 wird auf die Klassifikatoren eingegangen.

3.7.1 Merkmale

Zur Zeichenklassifizierung sollten Momente als Merkmale eingesetzt werden. Leider erfüllten die Momente nicht die in sie gesetzten Erwartungen. Alternativ wurden daher noch Grauwertmatrizen und Caliper-Distanzen als Merkmale getestet. Beide Verfahren sind aber Notlösungen und zu primitiv für eine industrielle Anwendung. Für den im Rahmen der Diplomarbeit zu erstellenden Prototyp sollten sie aber ausreichen. Für einen späteren industriellen Einsatz sollte ein anderes Verfahren verwendet werden. Hierzu würde ich das in [KaPB87] beschriebene Verfahren von Kahan, Pavlidis und Baird empfehlen, da ihr Ansatz überzeugend klingt und ihre Ergebnisse auch vielversprechend sind. Da ihr Ansatz aufwendiger ist, konnte ich ihn nicht realisieren. Ob ihr Verfahren hält was es verspricht, kann natürlich erst eine Realisierung zeigen.
Im Abschnitt 3.7.1.1 werden die Erfahrungen mit Momenten geschildert. Abschnitt 3.7.1.2 beschreibt die mit den Grauwertmatrizen und Caliper-Distanzen gemachten Tests. In Abschnitt 3.7.1.3 werden die Ergebnisse der Grauwertmatrizen und in Abschnitt 3.7.1.4 die Ergebnisse der Caliper-Distanzen erläutert.

3.7.1.1 Momente

Was sich in der Theorie sehr einfach anhörte, erwies sich in der Praxis als doch problematisch, gemeint ist die Benutzung von *Momenten*.
Probleme bereiten die Berechnung der angegebenen Formeln, und der große Wertebereich der Zeichen. Prokop und Reeves [PrRe92], Gonzalez und Woods [GoWo92] und Kim und Yuan [KiYu94] geben folgende Formel zur Berechnung der Standardmomente eines diskreten Bildes an:

$$m_{pq} = \sum_x \sum_y x^p y^q f(x,y)$$

Prokop und Reeves geben als Definitionsbereich für x $0 \leq x \leq N-1$ und für $0 \leq y \leq M-1$ an, bei einer Bildgröße von $N*M$. Alle anderen Autoren verzichten auf eine Angabe. Für p und q wird von keinem der Autoren ein Definitionsbereich explizit angegeben. Mögliche Werte sind aber alle $p, q \geq 0$.
Sind x und p gleichzeitig 0 oder auch y und q, ist die Potenz 0^0 zu berechnen. Diese ist aber nicht definiert. Eine Anwendung der Formel mit obigem Definitionsbereich ist daher mathematisch unmöglich. Keiner der Autoren weißt auf dieses Problem hin, geschweige denn erklärt, wie die Momente denn zu berechnen sind.
Das gleiche Problem stellt sich auch bei der Berechnung der Zentralmomente:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x,y)$$

Hier entsteht die Potenz 0^0, wenn x gleich $\bar{x}$ und p gleich 0 ist, bzw. y gleich $\bar{y}$ und q gleich 0 ist. Bei den Standardmomenten kann die Berechnung erfolgen, indem der Definitionsbereich von $[0;N-1]$ nach $[1;N]$ verändert wird, analog auch für y. Diese Veränderung ist unproblematisch. Bei den Zentralmomenten muß die Null übersprungen werden. Der Definitionsbereich wird zu $x = -\bar{x}, \ldots -1, 1, \ldots N-\bar{x}$. Eine andere Möglichkeit ist die Festlegung, daß 0^0 gleich 1 ist. Beide Methoden sind denkbar, aber mit Mathematik haben sie wenig zu tun. Ich habe für meine Berechnungen beide Methoden probiert, konnte aber keinen Unterschied in der Qualität der Ergebnisse feststellen.

Das zweite Problem ist der große Wertebereich der auftreten kann und auch auftritt. So entstehen Wertdifferenzen von bis zu 10^7. Der verwendete Klassifikator kann aber nur ganzzahlige Werte im Bereich -32768 bis 32767 verarbeiten, also ein Definitionsbereich kleiner 10^5. Bei der Anpassung der Werte an den Definitionsbereich muß in Kauf genommen werden, daß alle Werte kleiner 10^2 zu Null werden, oder alle Werte größer 10^5 gleich dem Maximalwert werden. Beides bedeutet einen erheblichen Informationsverlust. Dies ist insbesondere deshalb problematisch, da die auftretenden Werte nicht gleichmäßig über den Wertebereich verteilt sind, sondern oft entweder sehr klein oder sehr groß sind und durch die Umrechnung der Werte auf den Definitionsbereich des Klassifikators sehr viele Werte zu Null werden, bzw. zum Maximalwert. Der Merkmalvektor der Zeichen besteht dann häufig nur aus Nullen und Maximalwerten, ab und zu kommt auch ein anderer Wert vor. Die Differenz zwischen zwei großen Werten zweier Bilder kann dabei die Summe der Differenzen aller anderen Werte übertreffen und damit alleine ausschlaggebend für die Klassifizierung sein. Die Werte verschiedener Bilder

eines Zeichens schwanken recht stark. Gonzalez und Woods verwenden den Logarithmus der berechneten Werte um die Schwankungsbreite zu reduzieren. Hierbei gibt es jedoch wieder ein Problem. Die zu logarithmierenden Werte können auch negativ sein, das wird in einem Anhang auf Seite 88 gezeigt. Für negative Werte ist der Logarithmus aber nicht definiert. Wie Gonzalez und Woods daher logarithmieren können ist für mir unverständlich.
Für meine Berechnungen wurde auf eine Logarithmierung verzichtet und die große Schwankungsbreite in Kauf genommen.

Nachdem die Realisierung eines Momentverfahrens sich schon als schwieriger erwies als erwartet, enttäuschten auch die Ergebnisse. Die Erkennungsrate lag bei wenigen Prozent. Ich vermutete die Ursache hierfür an der zu geringen Zeichengröße, die auf den Chips auftreten, und machte Tests mit besonders großen Zeichen. Hierzu wurden die alphanumerischen Zeichen mehrerer Schriftarten in verschiedenen Größen ausgedruckt und anschließend wie ein normaler Chip mit der üblichen Kameraapparatur verarbeitet. Die so erzeugten Zeichen hatten eine Größe zwischen 25x25 und 40x40 Punkten. Die Zeichen wurden klassifiziert mittels der 7-Invarianten von Hu als Merkmalvektor und dem Hypersphereverfahren und zum Vergleich mittels der Grauwertmatrix der Größe 8x8 des Zeichenbildes und dem 1-Nearest-Neighbour-Verfahren. Daß unterschiedliche Klassifikationsverfahren verwendet wurden, hat einen Grund. Anhand der Ergebnisse des Hypersphere Verfahrens lassen sich leicht Rückschlüsse auf die Qualität der Merkmale ziehen, dies wird weiter unten noch erläutert. Bei guter Qualität der Merkmale liefert das Nearest-Neighbour-Verfahren meistens bessere Ergebnisse als das Hypersphere-Verfahren und ist unempfindlicher gegenüber den Parametereinstellungen. Beide Klassifikatoren wurden mit durchschnittlich vier Beispielen je Klasse antrainiert. Zum Test wurden im Mittel drei Zeichen je Klasse verwendet. Die Testbeispiele unterschieden sich in Schriftart oder Größe oder Beidem von den Lernbeispielen. Die *Tabelle 3-1* zeigt die Ergebnisse. Beim Hypersphere Verfahren kann ein zu klassifizierendes Zeichen innerhalb der Kugeln mehrerer Klassen liegen. In einem solchen Fall wird die Klassifikation als unsicher richtig bezeichnet, wenn die richtige Klasse dabei ist; ansonsten ist die Klassifikation unsicher falsch. Beim k-Nearest-Neighbour Verfahren können keine unsicheren Klassifikationen auftreten.

Der Vergleichsklassifikator ist der einfachste denkbare Klassifikator und erzielt hierfür akzeptable Ergebnisse. Diese wären noch verbesserbar durch Anpassung der Parameter des Nearest-Neighbour Algorithmus.
Der Momentklassifikator verwendet die Hypersphere Methode mit einem sehr großen Anfangsradius. Bei einem sehr großen Anfangsradius ist folgendes Verhalten zu erwarten: Die Kugeln sind so groß, daß alle zu klassifizierende Zeichen in einer Kugel liegen und daß sich die Kugeln verschiedener Klassen überlappen, so daß es zu sehr vielen unsicheren aber richtigen Klassifizierungen kommt.

	unsicher	sicher	gesamt
7-Invarianten von Hu und der Hypersphere Klassifikator			
richtig erkannt	39%	2%	41%
falsch erkannt	31%	12%	44%
nicht klassifizierbar		15%	15%
8x8 Grauwertmatrix und der 1-Nearest-Neighbour Klassifikator			
richtig erkannt	0%	91%	91%
falsch erkannt	0%	2%	2%
nicht klassifizierbar		7%	7%

Tabelle 3-1: Vergleich von Momenten mit Grauwertmatrizen

Die Ergebnisse zeigen, daß zumindest der zweite Teil der Erwartungen erfüllt ist. Nur wenige Zeichen werden sicher richtig erkannt, die meisten unsicher. Der Kugelradius müßte vermindert werden, damit aus den unsicheren Klassifikationen sichere werden können. Der erste Teil der Erwartungen, daß alle Zeichen klassifiziert werden, ist aber nicht erfüllt. Viele Zeichen werden falsch oder nicht erkannt. Dies erfordert eigentlich eine Vergrößerung des Kugelradius, so daß diese Zeichen auch in eine richtige Kugel fallen. Aus diesen gegensätzlichen Forderungen läßt sich das Fazit ziehen, daß die Werte der Momente zu weit variieren, um die Zeichen richtig klassifizieren zu können. Auch die Anwendung eines anderen Klassifikators kann über diesen Tatbestand nicht hinweghelfen. Die Ergebnisse mit einem 1-Nearest-Neighbour Klassifikator sind noch schlechter.
Die Verwendung von Standardmomenten oder Zentralmomenten anstatt der Invarianten von Hu brachte auch keine besseren Ergebnisse. Gleiches gilt für die Verwendung anderer Normierungsmethoden. Weder die Normierung nach Abo-Zaid (vergl. [PrRe92]) noch eine Skalierung der Zeichen auf eine einheitliche Größe von 8x8, 16x16 oder 64x64 vor der Momentberechnung führte zu akzeptablen Ergebnissen. Auch die Verwendung von 3, 4, 15, 25 oder 45 Momenten brachte keine Verbesserung. Egal ob die Momente aus einem Graubild mit 265 oder 16 Graustufen oder einem Binärbild berechnet wurden, egal ob 0^0 als 1 oder 0 definiert wurde oder übersprungen wurde, egal wie die Umrechnung der Momentwerte auf den Definitionsbereich des Klassifikators erfolgte, gerade mal 30% wurden maximal richtig klassifiziert. Die Ergebnisse bei der Erkennung größerer Zeichen waren sowohl mit 7 als auch mit 25 Momenten geringfügig besser, als der Durchschnitt, jedoch weiterhin miserabel.
Ich habe lange nach den Ursachen für die schlechten Ergebnisse gesucht, konnte aber keine Erklärung finden. Zur Überprüfung meines Programms, habe ich die Momente für ein Zeichen von Hand mit einem Taschenrechner nachgerechnet und bin zu den selben Werten gekommen. Karl A. Nyberg hat mir seine Implementierung der sieben Momente von Hu zur Verfügung gestellt, dafür möchte ich ihm an dieser Stelle noch einmal herzlich Danken. Seine Implementierung verfolgt einen anderen, optimierten Ansatz zur Momentberechnung, ermittelte aber immer die selben Werte wie meine Implementierung. Mein Programm sollte also richtig sein. Da die Klassifikatoren mit der Grauwertmatrix funktionieren, sollten auch diese Implementierungen korrekt sein.

Meine Vermutung, daß die Zeichen auf den Chips zu klein und daher die Schwankungen zu groß sind, wurde nicht bestätigt, da das Verfahren auch bei Zeichen der Größe 40x40 nur unzureichend ist. Möglicherweise funktionieren Momente erst bei utopischen Zeichengrößen von 100x100 oder mehr. Die Autoren, die Momente erfolgreich verwenden, machen leider keine Angaben über ihre Zeichengrößen. Dementsprechend sind die Ergebnisse von Hu und seinen Nachahmern für mich nicht nachvollziehbar.
Aufgrund der katastrophalen Ergebnisse mit den sieben Invarainten-Momenten von Hu, wurden keine Versuche mit anderen Momentarten unternommen. Diese sind zwar dafür bekannt, daß sie etwas besser sind als die Momente von Hu, aber Wunder gegenüber den Hu Momenten erwarte ich mir von ihnen auch nicht.

3.7.1.2 Testbedingungen

Zum Testen der verschieden Merkmalvektoren wurden *1-Nearest-Neighbour* Klassifikatoren verwendet. Der maximale Abstand des Nearest-Neighbour wurde auf unendlich gesetzt, so daß alle Zeichen klassifiziert wurden und keine Zeichen zurückgewiesen wurden. Die Klassifikatoren wurden mit 3000 Zeichen angelernt. Es wurde ein Test mit 1300 Zeichen bekannter Chips gemacht. Also mit Chips, von denen Chips gleicher Bauart in der Trainingsmenge enthalten waren. Ein weiterer Test wurde mit 250 Zeichen unbekannter Chips und dementsprechend auch möglichen unbekannten Schriftarten gemacht. Die Auswahl der Zeichen erfolgte rein zufällig.
Die Test und Trainingsmengen enthielten Zeichen aus 44 Klassen. Dies waren die zehn Ziffern, 24 Großbuchstaben, I und O wurden mit 1 bzw. 0 zusammengefaßt, da sie, auch für den Menschen, meistens nur im Kontext unterscheidbar sind. Weitere Klassen waren der Punkt, der Bindestrich, das kaufmännische Und, der Schrägstrich, das Copyright-Zeichen, vier Firmenlogos und eine Klasse „Störungen“ in der neben Störungen auch verklebte oder verdrehte Buchstaben und Buchstabenbruchstücke enthalten waren.

3.7.1.3 Grauwertmatrizen

Die Ergebnisse der Grauwertmatrizen sind in *Abbildung 3.21* und *Abbildung 3.22* dargestellt. Mit Grauwertmatrizen werden Erkennungsraten von 94% bei bekannten und 88% bei unbekannten Chips erreicht. Keine Matrixgröße konnte als eindeutig beste ermittelt werden. Die besten Ergebnisse bei bekannten Chips wurden mit Matrizen der Größen 7x7, 8x8, 9x8 und 9x10 erreicht. Bei unbekannten Chips erzielten die Matrizen 5x5, 6x5, 6x7, 6x8, 8x4, 8x8, 9x4, 9x5 und 9x7 die höchsten Erkennungsraten. Auffallend ist, daß im zweiten Fall viele kleine Matrizen beste Ergebnisse erzielen. Ein Grund hierfür wird sein, daß die Zeichen bei der zweiten Testreihe im Mittel kleiner waren als bei der ersten.

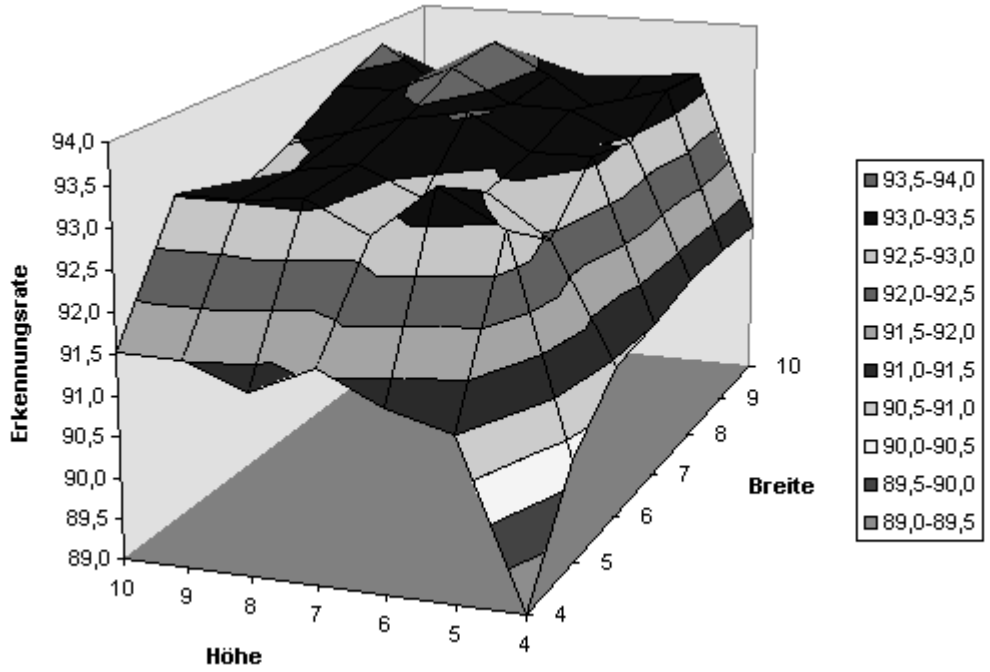


Abbildung 3.21: Erkennungsrate bei verschiedenen Grauwertmatrixgrößen bei bekannten Chips

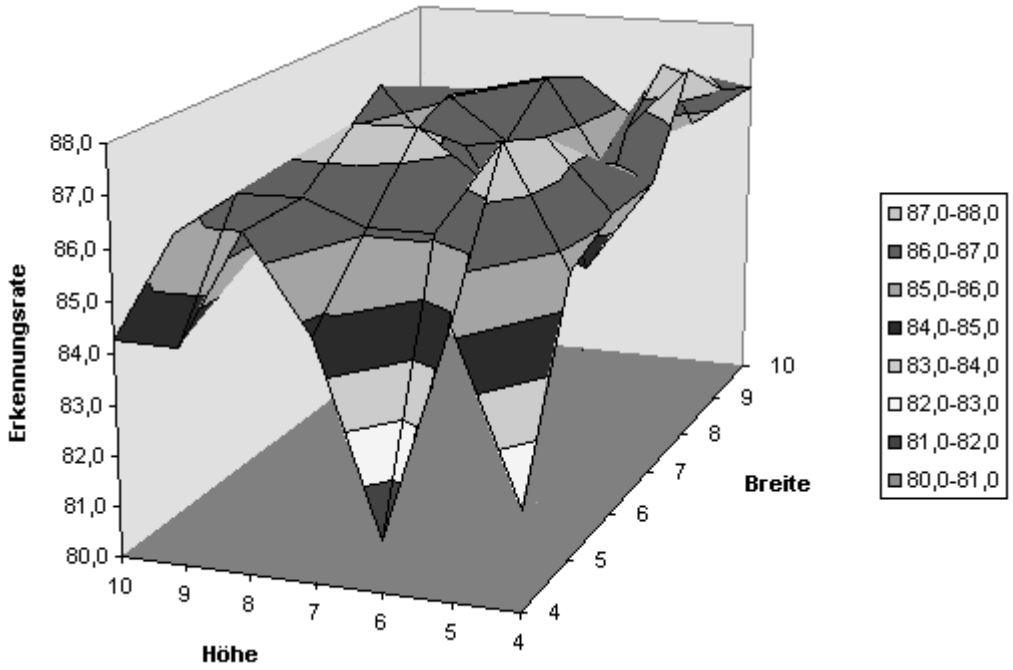


Abbildung 3.22: Erkennungsrate bei verschiedenen Grauwertmatrixgrößen bei unbekannten Chips

Die Verwendung von Matrizen mit einer Seitenlänge von mehr als 10 ist nicht sinnvoll. Die Ergebnisse sind bis auf eine Ausnahme (9x10) zum Rand immer abfallend. Noch größerer Matrizen würden zu inakzeptable langen Klassifikationszeiten führen.

Werden die beiden Testreihen zusammengefaßt ist die 8x8 Matrix eine der Besten. Interessant wäre auch die Verwendung einer 5x5 Matrix, die zwar eine um 1% niedrigere Erkennungsrate als die 8x8 Matrix hat, da sie aber mit weniger als der Hälfte der Merkmale auskommt, ist die Zeichenerkennung mit ihr doppel so schnell.

3.7.1.4 Caliper-Distanzen als Klassifikationsmerkmale

Abbildung 3.23 zeigt die Ergebnisse der Caliper-Distanz für bekannte Chips. Auf eine zweite Testreihe mit unbekannten Chips wurde aufgrund der schlechteren Ergebnisse im Vergleich zu Grauwertmatrizen verzichtet.
Bei der Caliper-Distanz ist der Verlauf der Erkennungsrate sehr schwankend. Maximal 78% der Zeichen werden richtig klassifiziert. Eine Erhöhung der Merkmalanzahl führt nicht unbedingt zu einer besseren Erkennungsrate. Die in der Testdatenmenge enthaltenen Zeichen 0, © und M-Kreis (ein Firmenlogo) sind alle kreisrund und unterscheiden sich nur durch ihr Innenleben. Sie sind damit mit der Caliper-Distanz nicht zu unterscheiden. Ungefähr 4% der Fehlklassifikationen lassen sich so erklären. Bei gleicher Merkmalanzahl sind die Ergebnisse der Caliper-Distanz, auch unter Berücksichtigung dieser 4%, deutlich schlechter als die Ergebnisse der Grauwertmatrix.

Abschließend kann festgehalten werden: Die Erkennungsraten sind mit Grauwertmatrizen deutlich besser als mit Caliper-Distanzen. Die 8x8 Matrix hat sich als eine der besten Matrixgrößen herausgestellt. Im weiteren Verlauf wird daher mit dieser gearbeitet.
Im folgen Abschnitt wird untersucht, ob sich die Erkennungsrate der 8x8 Grauwertmatrix durch einen anderen Klassifikator oder andere Klassifikatorparameter noch steigern läßt.

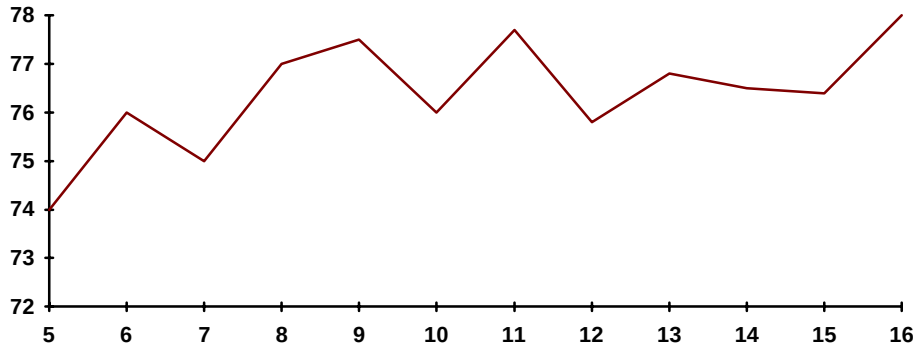


Abbildung 3.23: Erkennungsrate mit der Caliper-Distanz, bei Merkmalvektor-größen von 5 bis 16 Merkmalen pro Seite

3.7.2 Klassifikationsverfahren

Im vorausgegangenen Abschnitt wurde ermittelt, daß die einfachste untersuchte Lösung, die Verwendung der Grauwertmatrix, für diese

Anwendung die Beste ist. Im folgenden wird der Einfluß des Klassifikations-verfahrens auf die Ergebnisse untersucht.
Es wurden zwei Klassifikatoren getestet. Der k-Nearest-Neighbour und der Hypersphere Klassifikator. Als Klassifikationsmerkmal wurde eine 8x8 Grauwertmatrix verwendet. Es wurden die gleichen Klassen, Trainings- und Testmengen verwendet, wie beim Testen der Merkmalvektoren (siehe Abschnitt 3.7.1.2). Mit jedem Klassifikator wurde also ein Test mit bekannten und unbekannten Chips durchgeführt.
Die Tests wurden verwendet, um die optimalen Parameter für die Klassifikatoren zu bestimmen. Zur Ermittlung der realistischen Erkennungsrate der Klassifikatoren wäre eigentlich eine weitere unabhängige Testmenge erforderlich. Da die Erkennungsrate, auch bei unterschiedlichen Parametern, nur sehr geringfügig schwankte, wurde auf einen abschließenden, erneuten Test verzichtet und die Erkennungsrate aus den Tests zur Parameteroptimierung als Erkennungsrate der Klassifikatoren angegeben.

3.7.2.1 k-Nearest-Neighbour

Beim Nearest-Neighbour wurde die Anzahl der betrachteten Nachbarn und der maximale Abstand variiert. Dabei wurde zuerst die Anzahl Nachbarn variiert und der maximale Abstand auf unendlich gesetzt, so daß alle Zeichen irgendeiner Klasse zugeordnet werden. Wie *Abbildung 3.24* und *Abbildung 3.25* zeigen, wurden die besten Ergebnisse mit drei Nachbarn erzielt. Bei bekannten Chips ergab sich eine Erkennungsrate von 93,85% bei unbekannten Chips 87,5%. Die Varianz der Anzahl der Nachbarn führte nur zu geringfügigen Unterschieden in den Ergebnissen. Die häufigsten Fehlklassifikationen sind das Klassifizieren von Oberflächenkonturen oder verklebten Zeichen zu einer Klasse anstatt diese Eingaben abzulehnen. Von allen falsch Klassifikationen sind 25%-45% Klassifikationen von Störungen.

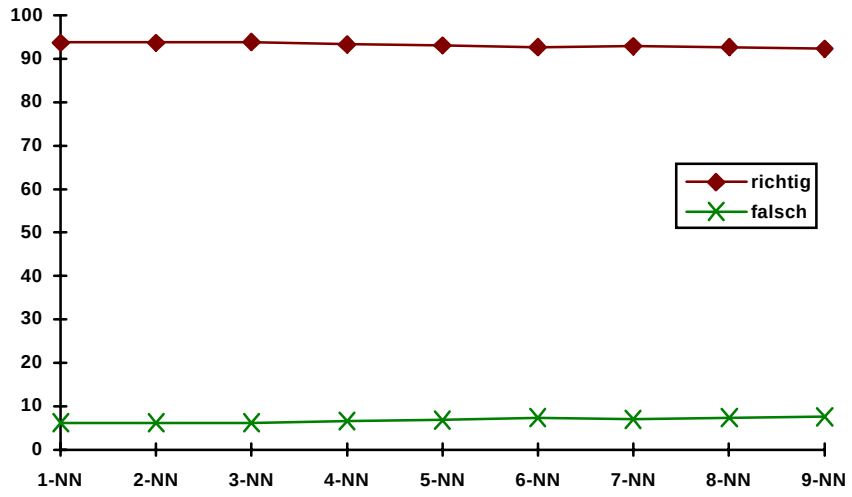


Abbildung 3.24: Erkennungsrate des k-Nearest-Neighbours in Abhängigkeit von der Anzahl der betrachteten Nachbarn bei bekannten Chips

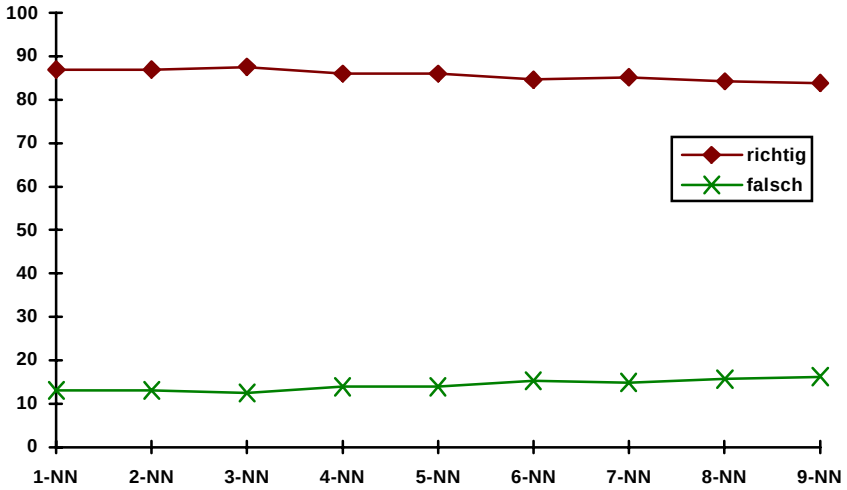


Abbildung 3.25: Erkennungsrate des k-Nearest-Neighbours in Abhängigkeit von der Anzahl der betrachteten Nachbarn bei unbekannten Chips

Am Häufigsten werden Störungen als „1“ erkannt. Fast jede zweite Störung wird klassifiziert anstatt zurückgewiesen zu werden. Dies zeigt, daß eine Grauwertmatrix als Merkmal zur Zeichenklassifikation unzureichend ist. Mit einer Grauwertmatrix können zwar Zeichen zuverlässig unterschieden werden, das Erkennen und Zurückweisen von Störungen gelingt jedoch nicht.

Nachdem die beste Anzahl Nachbarn ermittelt worden war, wurde der maximale Abstand für diese Anzahl variiert. Die Dokumentation zu dem verwendeten k-Nearest-Neighbour-Paket gibt leider keinen Aufschluß darüber wie der Abstand zweier Merkmalvektoren berechnet wird. Die hier ermittelten besten Abstandswerte sind daher nur aussagekräftig für dieses k-Nearest-Neighbour Paket. Bei Verwendung einer anderen Implementierung würden vermutlich andere Abstandswerte ermittelt.

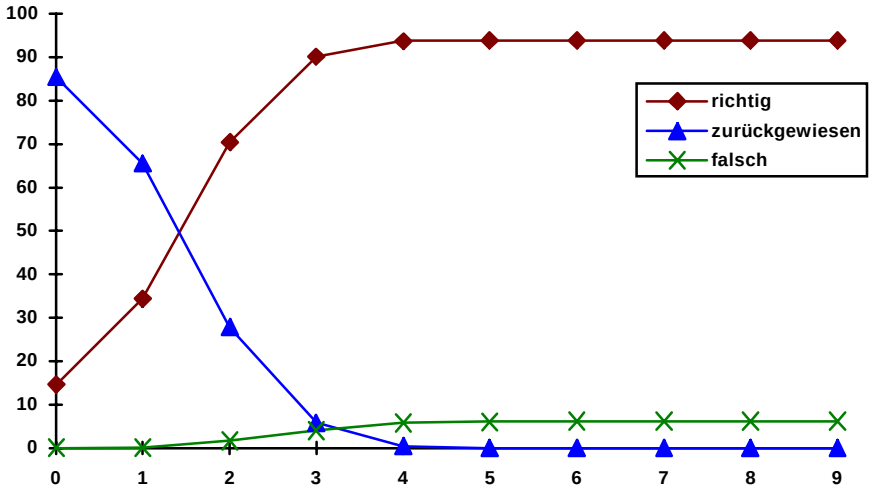


Abbildung 3.26: Einfluß des maximalen Abstandes auf die Erkennungsrate bei bekannten Chips, bei Verwendung des 3-Nearst-Neighbours.

Abbildung 3.26 und *Abbildung 3.27* zeigen, daß hier nur Werte von 0-5 interessant sind. Alle größeren Werte, auch sehr große Werte von 100 oder 1000, führen zu den selben Ergebnissen wie der Wert 5. In beiden Bildern wird deutlich, daß mit größer werdendem maximalen Abstand, die Erkennungsrate und Fehlerrate ansteigen und die Zurückweisungssrate abfällt. Bei unbekannten Chips, hat die Erkennungsrate ein Maximum beim maximalen Abstand 4, fällt dann wieder ganz leicht ab und bleibt auf einem konstanten Wert. Bei den bekannten Chips wird die maximale Erkennungsrate ab einem maximalen Abstand von 4 erreicht.

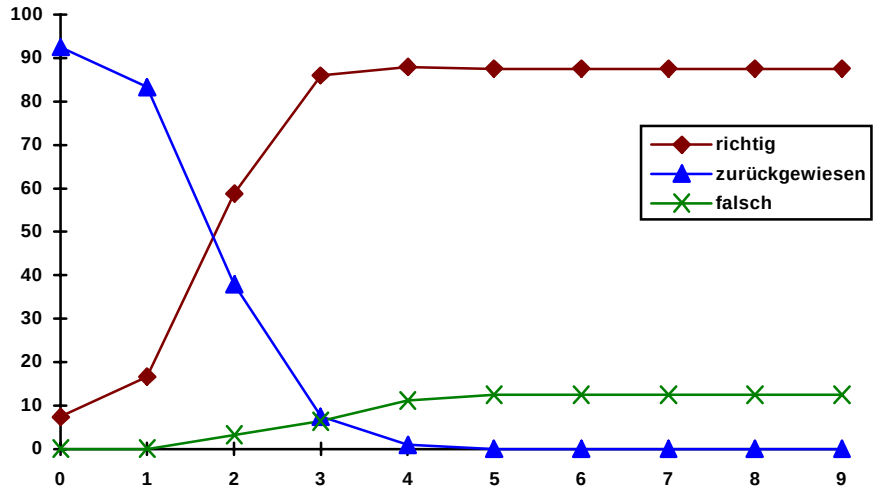


Abbildung 3.27: Einfluß des maximalen Abstandes auf die Erkennungsrate bei unbekannten Chips, bei Verwendung des 3-Nearest-Neighbours.

Welcher maximale Abstand als der beste angesehen wird, hängt von den zugrunde gelegten Kriterien ab. Ist es wichtig, daß möglichst wenig Zeichen falsch klassifiziert werden und wünschenswerter, daß ein Zeichen lieber gar nicht klassifiziert wird als falsch, so ist ein maximaler Abstand von 3 der Beste. Bei diesem ist die Fehlerquote recht gering (5%), dafür können aber weitere 5-10% der Zeichen gar nicht klassifiziert werden. Ist eine möglichst hohe Erkennungsrate gewünscht und sind die Folgen von Fehlklassifikation und Nicht-Klassifikation gleich oder ähnlich, so kann ein maximaler Abstand von 4 verwendet werden. Bei diesem ist die Erkennungsrate etwas höher als beim Abstand 3, die Zurückweisungsrate null, aber gleichzeitig auch die Fehlerrate bei bis zu 10%. Ob eine geringe Fehlerrate oder eine größere Erkennungsrate vorteilhafter ist, hängt von den nachbearbeitenden Schritten ab. Dies sind insbesondere die Trennung von verklebten Zeichen und der Abgleich des Gelesenen mit einer Datenbank. Wird in einem Trennvorgang ein Zeichen als nicht klassifizierbar zurückgewiesen, so werden alle verklebten Zeichen dieses Trennvorgangs nicht erkannt. Bei der Trennung wird also besonderer Wert auf eine hohe Erkennungsrate gelegt. Die Fehlklassifikation eines Zeichens ist dabei weniger tragisch, als wenn der ganze zu trennende Zeichenblock zurückgewiesen wird, weil ein einziges Zeichen nicht klassifiziert werden konnte. Eine hohe Fehlerrate beim Trennen erhöht aber auch die Wahrscheinlichkeit, daß an falschen Stellen getrennt wird und Zeichenteile als ein Zeichen erkannt werden. Solche falschen Trennungen können aber nur bestehen, wenn mehrere falsche Trennungen

vorgenommen werden, so daß insgesamt die falsch getrennten Bereiche genau die Lage und Breite eines oder mehrerer Zeichen haben. Falsche Trennungen werden daher meistens als solche erkannt und verworfen, da der restliche zu trennende Bereich nicht komplett getrennt werden kann. Falsche Trennungen verlängern daher meistens nur die Zeit bis die richtige Trennung gefunden wird. Für die Trennung scheint mir daher eine möglichst hohe Erkennungsrate vorteilhafter als eine möglichst geringe Fehlerrate.

Der andere nachbearbeitende Schritt, der fehlertolerante Abgleich der gelesenen Aufschrift mit den Einträgen von Chipbeschriftungen in einer Datenbank, ist nicht mehr Bestandteil dieser Diplomarbeit. Von den in diesem Schritt verwendeten Algorithmen wird es abhängen, ob Zurückweisungen vorteilhafter sind, als Fehlklassifikationen. Handelt es sich hierbei um optimistische Algorithmen, die erwarten, daß erkannte Zeichen auch richtig sind und bei Zeichen, die nicht sicher bestimmt werden konnten, diese als nicht klassifizierbar eingestuft wurden, so sollte eine konservativerer maximaler Abstand von 3 verwendet werden. Sind die Algorithmen zum Datenbankabgleich konservativ und gehen bei jedem erkannten Zeichen davon aus, daß es falsch sein könnte, so kann ein optimistischer maximaler Abstand von 4 gewählt werden.

Solange noch kein Nachbearbeitung der gelesenen Schrift erfolgt, habe ich die freie Wahl für oder gegen einen optimistischen maximalen Abstand. Ich werde daher einen optimistischen maximalen Abstand von 4 verwenden, um eine möglichst hohe Erkennungsrate zu erzielen.

3.7.2.2 Hypersphere Klassifikator

Die Ergebnisse mit dem Hypersphere Klassifikator sind in *Abbildung 3.28* und *Abbildung 3.29* dargestellt. Es wurden die gleichen Trainings- und Testzeichen verwendet wie beim k-Nearest-Neighbour Verfahren. Während das k-Nearest-Neighbour Verfahren sich immer für maximal eine zugehörige Klasse pro Zeichen entscheidet, werden beim Hypersphere Verfahren auch mehrere Klassen für ein Testzeichen ermittelt. Die Klassen sind dann gleich passend für das klassifizierte Zeichen. Wurden mehrere Klassen für ein Zeichen ermittelt und war die richtige Klasse unter diesen Klassen enthalten, so ist dies in den Abbildungen als „fast richtig“ bezeichnet. Die besten Ergebnisse erzielt der Hypersphere Algorithmus für beide Testmengen bei einem Kugelradius von 250. Die Ergebnisse sind jedoch deutlich schlechter, als beim k-Nearest-Neighbour und zwar auch dann, wenn die fast richtig klassifizierten Zeichen den richtig klassifizierten Zeichen zugerechnet werden. Die Erkennungsrate liegt dann für bekannte Chips bei 84,6% und bei unbekannten Chips bei 79,2%. In der praktischen Anwendung ist eine fast richtige Klassifikation aber weitaus schlechter und nichtssagender als eine richtige.

Hypersphere und 3-Nearest-Neighbour Klassifikator benötigen gleich viel Speicherplatz. Der 3-Nearest-Neighbour Klassifikator ist recht langsam. Er klassifiziert maximal 2-3 Zeichen pro Sekunde. Der Hypersphere Klassifikator ist ungefähr 30% schneller. Dieser minimale Geschwindigkeitsvorteil kann aber kein Grund sein die geringere Erkennungsrate in Kauf zu nehmen. Eine

bessere Lösung wäre der Einsatz von optimierten Nearest-Neighbour Verfahren. Verweise auf solche Verfahren finden sich bei der Beschreibung des k-Nearest-Neighbour Verfahrens in [MiST94].

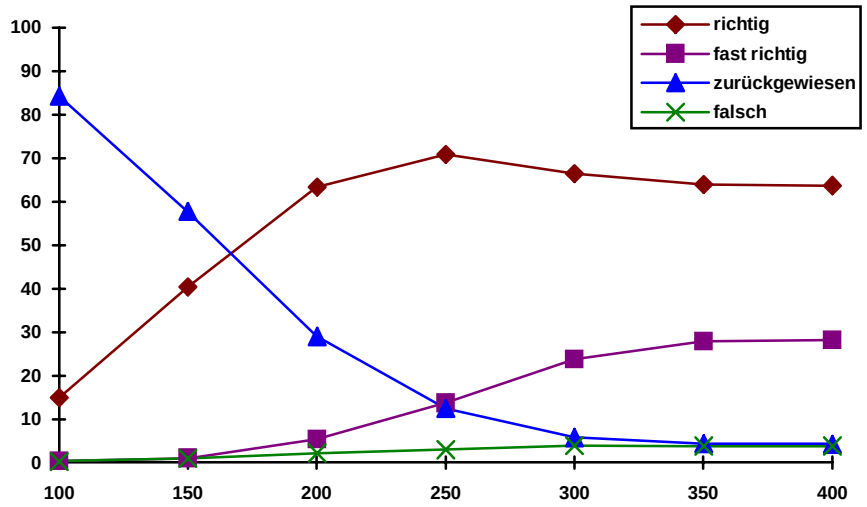


Abbildung 3.28: Einfluß des Kugelradius beim Hypersphere Klassifikator auf die Erkennungsrate bei bekannten Chips.

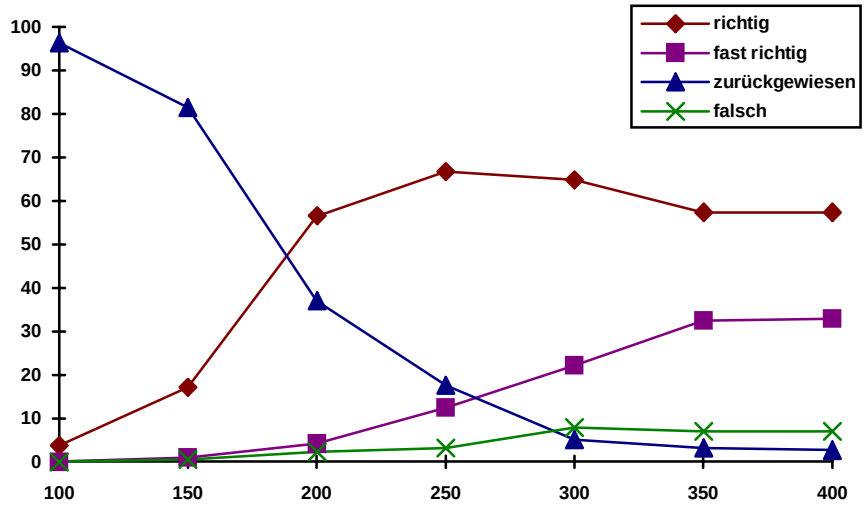


Abbildung 3.29: Einfluß des Kugelradius beim Hypersphere Klassifikator auf die Erkennungsrate bei unbekannten Chips.

3.8 Trennen von Zeichen

Es wurden zwei Verfahren getestet, um verklebte Zeichen zu trennen. Das von Shuichi Tsujimoto und Haruo Asada vorgeschlagene Nachbarverfahren und die Verwendung der *Caliper-Distanz*.
Beide Verfahren sind nicht perfekt. Zum einen finden die Verfahren unter bestimmten Bedingungen eine Stelle nicht, an der getrennt werden muß. Diese Stellen werden im folgenden als fehlende Trennstellen bezeichnet und im Abschnitt 3.8.1 behandelt. Zum anderen schlagen sie Trennstellen

innerhalb von Zeichen vor, also an Stellen an denen nicht zu trennen ist. Diese Trennstellen werden im folgenden als falsche Trennstellen bezeichnet und im Abschnitt 3.8.2 untersucht. Falsche Trennstellen verzögern den Trennvorgang, fehlende Trennstellen machen eine Trennung unmöglich. Entscheidend für die Qualität eines Trennverfahrens ist daher hauptsächlich die Häufigkeit von fehlenden Trennstellen. Einen großen Einfluß auf das Trennergebnis hat auch die Reihenfolge, in der Trennstellen getestet werden. Mit der Trennreihenfolge beschäftigt sich Abschnitt 3.8.3.

Abbildung 3.30: Fehlende Trennstellen. In der ersten Bildspalte sind zwei verklebte X zu sehen, in der Zweiten ein verklebtes E mit einem F. In beiden Fällen findet das Nachbarverfahren die Trennstellen während beim Caliper-Verfahren diese fehlen. In der dritten Bildspalte sind zwei O und eine C verklebt. Hier versagt das Nachbarverfahren während das Caliperverfahren diese Stellen erkennt. In der letzten Spalte versagen beide Verfahren bei der verklebten Zeichenfolge MNP. Die gefundenen Trennstellen sind durch einen verlängerten Strich nach unten markiert.

3.8.1 Fehlende Trennstellen

Abbildung 3.30 zeigt fehlende Trennstellen für beide Verfahren. Bei den fehlenden Trennstellen gibt es solche, die nur von einem Verfahren nicht gefunden werden und solche, bei denen beide Verfahren versagen. Das Caliper-Verfahren versagt nur im Zusammenhang mit vertikalen Strichen. Das Nachbarverfahren versagt bei runden und schrägen Zeichen, insbesondere auch bei den Zahlenfolgen 00 und 88. Da auf Chips mehr Ziffern als Buchstaben zu finden sind, ist das Caliper-Verfahren für meine Anwendung zuverlässiger als das Nachbarverfahren. Die Fälle in denen das Caliper-Verfahren versagt sind in meiner Anwendung selten.

Mit zwei Erweiterungen müßte das Caliper-Verfahren in der Lage sein alle Trennstellen zu finden. Zum einen durch Verwendung des Nachbarverfahrens an den Stellen an denen die Caliper-Distanz null ist. Das sind die Stellen, an denen direkt am oberen und unteren Zeichenrand bereits Zeichenpunkte vorhanden sind, z.B. bei vertikalen Strichen ober über die ganze Breite eines großen „E“. Die zweite Erweiterung wäre das Einfügen einer zusätzlichen Trennstelle in der Mitte zwischen zwei gefunden Trennstellen, wenn die Fläche zwischen den beiden gefundenen Trennstellen sowohl in der Höhe als auch in der Breite komplett schwarz ist. Die erste Zusatzregel findet die Trennstellen, die vom Nachbarverfahren gefunden werden, aber nicht vom Caliper-Verfahren. Dabei wird der Abschnitt in dem mit dem Nachbarver-

fahren gesucht wird auf den Bereich limitiert, in dem das Caliper-Verfahren versagen kann. Die zweite Zusatzregel findet die Stellen, die bei beiden Verfahren fehlen. Sie führt jedoch zu einer großen Zahl zusätzlicher falscher Trennstellen. Ideal wäre, wenn die normale vertikale Strichbreite der Zeichen bekannt wäre und die zweite Regel nur angewendet wird, wenn der Abstand zwischen zwei Trennstellen diese Breite übersteigt. Die Ermittlung der normalen Breite ist nicht trivial, da sie nicht aus der *vertikalen Projektion* gewonnen werden kann, denn die Zeichen sind ja verklebt und es gibt Zeichen mit doppelter Strichbreite, eben jene die getrennt werden sollen. Ob es auch Zeichen mit einer normaler Strichbreite gibt ist nicht sicher. Mit diesen Erweiterungen sollte das Caliper-Verfahren alle Verklebungen trennen können. Aus zeitlichen Gründen konnte diese Erweiterung nicht mehr realisiert werden.

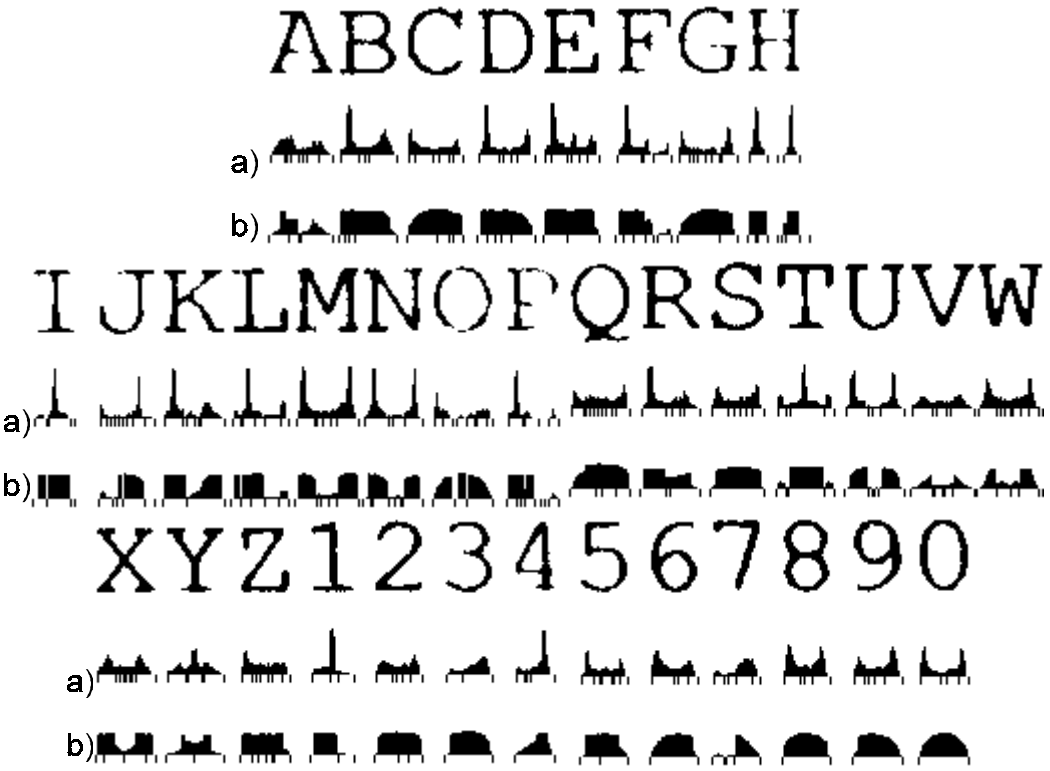


Abbildung 3.31: Vergleich a) Nachbarverfahren und b) Caliper-Distanzverfahren anhand eines eingescannten Alphabets. Die Striche markieren die gefundenen Trennstellen. Das Nachbarverfahren ermittelt 167 falsche Trennstellen (4,64 pro Zeichen), das Caliperverfahren nur 96 (2,67 pro Zeichen). Bei Ziffern ist das Caliperverfahren besonders erfolgreich: nur 7 falsche Trennstellen (0,7/Z.), gegenüber 33 beim Nachbarverfahren (3,3/Z.).

3.8.2 Falsche Trennstellen

Das Caliper-Verfahren liefert im Mittel weniger falsche Trennstellen als das Nachbarverfahren. Dies ist in *Abbildung 3.31* dargestellt. Die Ergebnisse schwanken je nach Schriftart und Vorlagenqualität. Das Caliper-Verfahren bleibt jedoch dem Nachbarverfahren überlegen. Dies ist ein weiterer Grund das Caliper-Verfahren dem Nachbarverfahren vorzuziehen.

3.8.3 Trennreihenfolge

Sind mögliche Trennstellen gefunden worden, muß eine Reihenfolge festgelegt werden, in der die Stellen ausprobiert werden. Eine gute Reihenfolge ist nicht nur wichtig um das zeitaufwendige Probieren falscher Trennstellen zu reduzieren, sondern auch um Übertrennung oder falsche Trennung zu verhindern. Die *Abbildung 3.32* zeigt Beispiele für beide Fälle.
Übertrennung entsteht durch das Abarbeiten der Zeichen von links nach rechts. Das heißt es wird zuerst versucht den Bereich zu klassifizieren, der zwischen dem linken Rand und der linkesten Trennstelle liegt. Ist diese Trennung nicht erfolgreich, wird der Bereich zwischen dem linken Rand und der zweitlinkesten Trennstelle klassifiziert, usw. Beim Abarbeiten der Trennstellen von rechts nach links, wird zuerst der Bereich zwischen dem linken Rand und der am weitesten rechts liegenden Trennstelle klassifiziert. Bei Mißerfolg wird der Bereich zwischen dem linken Rand und der zweitrechtesten Trennstelle probiert, usw. Dieses Vorgehen kann zu falschen Trennungen führen.

M ⇒ IVI CK ⇒ OC

Abbildung 3.32: links: Abarbeitung der Trennstellen von links nach rechts und die entstehende Übertrennung. Rechts: Abarbeitung von rechts nach links und falsche Trennung.

Eine falsche Trennung kann durch ein besseres Zeichenerkennungsverfahren verhindert werden. Insbesondere wäre es hilfreich Wahrscheinlichkeiten für die Richtigkeit einer Klassifikation vom Klassifikationsverfahren zu erhalten. Es könnten dann solange Trennstellen ausprobiert werden, bis die ermittelte Wahrscheinlichkeit wieder sinkt. Die Klassifikation mit der größten Wahrscheinlichkeit würde akzeptiert. Die Übertrennung ist ein Problem, das im Aufbau der Zeichen begründet liegt. Ein Abarbeiten der Trennstellen von rechts nach links ist daher dem umgekehrten Fall vorzuziehen.
Sind viele Zeichen verklebt, bedeutet ein Abarbeiten von rechts nach links das Ausprobieren von vielen unwahrscheinlichen Trennstellen, denn die entstehenden Zeichen haben eine viel zu große Breite. Um diese unwahrscheinlichen Trennstellen anfangs zu umgehen, wird die erste Trennstelle, die ausprobiert wird, bei der Breite gesucht, die der Zeichenhöhe entspricht und dann von dort bis zum linken Rand die Trennstellen ausprobiert. Konnte bis dahin keine Klassifizierung erfolgen, wird noch bis zu einer Breite, die der dreifachen Zeichenhöhe entspricht, nach einer passenden Trennstelle gesucht. Kann bis dahin keine erfolgreiche Trennung vorgenommen werden, wird die übergebene Zeichenfolge als nicht klassifizierbar eingestuft. Die Grenzwerte einfache und dreifache Zeichenhöhe wurden so gewählt, daß sie etwas oberhalb der mittleren und maximal aufgetretenen Höhenbreitenverhältnisse für die Buchstaben M und W liegen.

Die Reihenfolge der Trennstellen könnte noch weiter optimiert werden. Möglich wäre eine dynamische Anpassung der Einstiegsbreite und der Maximalbreite aufgrund der Breite schon klassifizierter Buchstaben im gleichen Wort. Eine andere Möglichkeit wäre die Definition einer Reihenfolge

aufgrund der Minimalität des Trennvektors an den Trennstellen. Es wird zuerst die Trennstelle versucht, die den geringsten Wert im Trennvektor hat, dann die Stelle mit dem zweitgeringsten usw. Anstelle des geringsten Wertes im Trennvektor kann auch die Verwendung des Gradienten an der Trennstelle im Trennvektor sinnvoll sein. Ob diese Reihenfolgen besser sind als die verwendeten, müßten entsprechende Tests erst zeigen.

Abschließend lassen sich die Ergebnisse dieses Kapitels folgendermaßen zusammenfassen. Die Caliper-Distanz ist deutlich besser als das Nachbarverfahren. Sie hat sowohl weniger fehlende Trennstellen als auch falsche Trennstellen. Es wurde eine Erweiterung der Caliper-Distanz vorgeschlagen, mit der sich alle Trennstellen finden lassen müßten.
Die Trennreihenfolge hat nicht nur einen Einfluß auf die Trenndauer sondern auch auf das Trennergebnis. Es ist vorteilhafter die Trennstellen von breiteren zu schmaleren Zeichen auszuprobieren als das umgekehrte Vorgehen. Durch Berücksichtigung der Werte an den Trennstellen ließe sich die Trennreihenfolge möglicherweise noch verbessern.

3.9 Zusammenfassung des realisierten Konzepts

Analog zum Kapitel 2.9 ist das realisierte Schrifterkennungssystem in diesem Kapitel erneut in drei Datenflußdiagrammen dargestellt. Die Symbolik der Diagramme wurde bereits in Kapitel 2.9 erklärt, auf eine Wiederholung wird verzichtet.
Während in den anderen Diagrammen noch mehrere zur Auswahl stehende Algorithmen angeführt waren, ist in diesen Diagrammen nur noch der verwendete aufgeführt. Ansonsten hat es folgende Änderungen zwischen Konzept und Realisierung gegeben.
Im Diagramm des Gesamtsystems wird der Schwellwert im ungefilterten Bild mit dem Diskriminanz-Verfahren berechnet. Zusätzlich zu Zeichen-, Wort- und Linienhypothesen werden auch Blockhypothesen ermittelt. Die Bestimmung der Richtungsorientierung wird ohne Trennung durchgeführt. Anstatt alle Zeichen mit zwei Richtungen zu klassifizieren, wird die Bestimmung der Richtung bereits erfolgreich abgebrochen, wenn in einer Richtung drei Zeichen mehr erkannt wurden als in der anderen. Da nach der Richtungsorientierungsbestimmung noch nicht alle Zeichen klassifiziert sind, gibt es einen nachfolgenden Schritt, der die restlichen Zeichen klassifiziert, dieser unternimmt auch Trennungsversuche.
Für die Zeichenerkennung werden keine Momente verwendet, sondern die 8x8 Grauwertmatrix. Als Klassifikator wird ein 3-Nearest-Neighbour verwendet.
Trennstellen werden nicht von links nach rechts durchprobiert, sondern von einer mittleren Position zum linken Rand und dann wieder von der mittleren Position nach rechts. Die Trennstellen werden mit der Caliper-Distanz bestimmt.

Die Zeit zur Erkennung eines gesamten Chips schwankt zwischen 10 und 25 Sekunden auf einem Rechner mit Pentium 75 CPU. Die Zeit ist abhängig von der Chipoberflächengröße und vor allem von der Anzahl der Zeichen auf dem Chip. 10% der Zeit werden durchschnittlich für die ersten Schritte bis zur

Bestimmung der Achsenorientierung benötigt. Die restlichen 90% der Zeit verbrauchen die Bestimmung der Richtungsorientierung und die restliche Zeichenerkennung mit Trennen. Diese 90% lassen sich noch zerlegen in 15% für Normierungen, Berechnungen und ähnliches und 75% in reine Klassifizierungszeit des k-Nearest-Neighbour Verfahrens. Das größte Potential zur Beschleunigung der Anwendung liegt damit in einer Optimierung des k-Nearest-Neighbour Verfahrens.

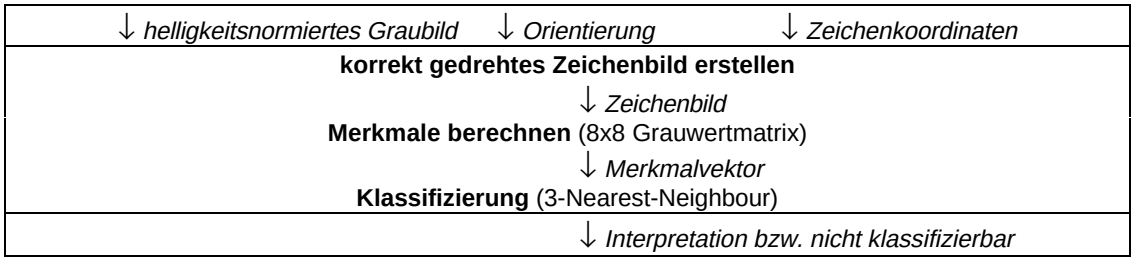


Abbildung 3.33: Ablauf der realisierten Zeichenerkennung, Logos werden wie Zeichen behandelt

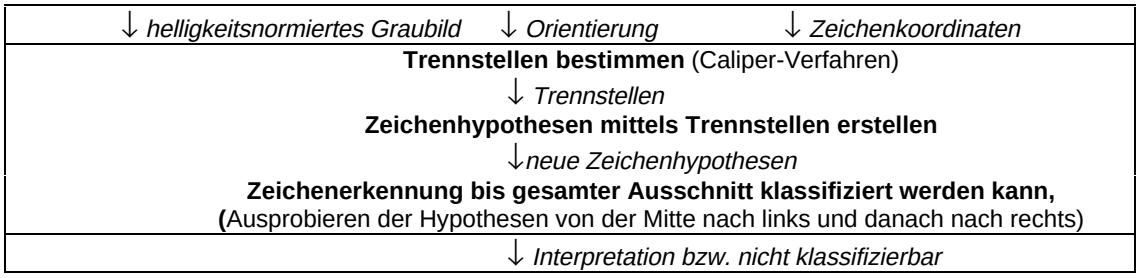


Abbildung 3.34: Ablauf der realisierten Trennung

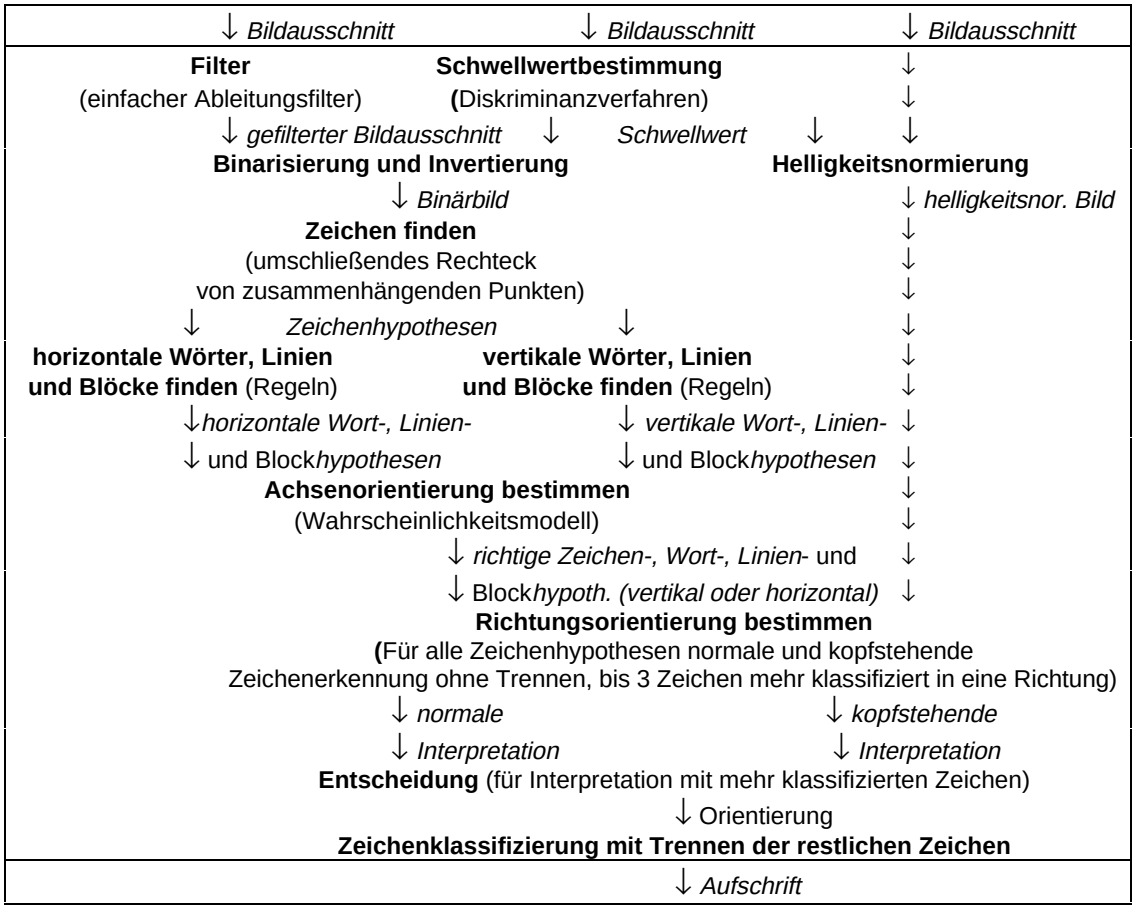


Abbildung 3.35: Aufbau des realisierten Schrifterkennungssystems, dargestellt anhand eines Datenflußdiagramms

4 Zusammenfassung und Ausblick

Das erstellte Konzept hat sich größtenteils als praktikabel und zuverlässig erwiesen. Das realisierte System erfüllt seine Aufgabe. Die Aufschrift von Chips beliebiger Größe und Bauart kann gelesen werden!
Die Fehlerrate ist mit ungefähr 6% - 12% für einen Prototyp akzeptabel. Die sich im Gesamtprojekt an die Schrifterkennung anschließende fehlertolerante Datenbanksuche kann mit den so erkannten Beschriftungen die Chips identifizieren.
Die Verarbeitungszeit für einen Chip von 10 bis 25 Sekunden ist etwas langsam. 75% dieser Zeit verbraucht das verwendete Klassifikationsverfahren. Die Verarbeitungszeit könnte vermutlich durch Ersetzung dieses Moduls (welches zugekauft wurde) durch ein optimiertes, um den Faktor 3 bis 4 gesteigert werden.

Im Folgenden werden die Ergebnisse und offenen Probleme der einzelnen Bearbeitungsschritte zusammengefaßt.

Die Digitalisierung könnte durch ein elektrisches Zoom-Objektiv und einen Autofokus verbessert werden.
Die Ausstattung der Kamera mit einem elektronischen Zoom-Objektiv würde die sichere Bearbeitung von beliebig kleinen Chips ermöglichen. Zusätzlich würde dies sicher auch der Zuverlässigkeit des Systems für größere Chips steigern. Bisher wird die volle Kameraauflösung nur bei den größten existierenden Chips ausgenutzt. Bei allen anderen Chips zeigt der größte Teil des Kamerabildes die uninteressante Umgebung des zu untersuchenden Chip. Ein Zoom-Objektiv würde die Anpassung des Bildausschnittes an die bekannte Chipgröße ermöglichen, es würde immer die volle Kameraauflösung genutzt werden.
Hilfreich wäre auch ein Autofokus der Kamera, um den Brennpunkt den unterschiedlichen Chiphöhen anpassen zu können.

Im Rahmen der Vorverarbeitung wurden verschiedene Filter und Filterkombinationen zur Verbesserung der Bildqualität getestet. Ein selbst entwickelter einfacher Ableitungsfilter erwies sich als am zuverlässigsten.
Ein ungelöstes Problem sind bisher Chips mit eingravierter Schrift. Die verwendete Vorverarbeitung ermöglicht bisher keine Auswertung von Gravuren. Es wurden Anzeichen gefunden, daß hierfür nicht nur eine andere Vorverarbeitung, sondern auch eine andere Beleuchtung benötigt wird. Insgesamt bleibt hier noch ein Gebiet auf dem geforscht werden muß.

Auf eine Vorsegmentierung wird bisher verzichtet. Nur bei einem einzigen Chip führte dies zum Scheitern des Lesens der Beschriftung. Die Entwicklung einer Vorsegmentierung hat damit eine sehr geringe Priorität.

Für die Binarisierung wurden drei Verfahren zur Schwellwertbestimmung miteinander verglichen. Die überzeugendsten Ergebnisse lieferte das Diskriminanz-Verfahren.

Alternativ zu Schwellwertbestimmungsverfahren wurde auch mit der Pixelaggregation experimentiert. Sie ist für diese Anwendung aber unzureichend.
Ein Kriterium wurde vorgeschlagen zur Ermittlung, ob helle Schrift auf dunklem Hintergrund vorliegt oder der inverse Fall. Dieses Kriterium hat sich in der Praxis als sehr zuverlässig erwiesen.

Die Segmentierung wurde durch ein Regelsystem realisiert. Anhand des Regelsystems werden Vordergrundpunkte zu Zeichen, Zeichen zu Worten, Worte zu Linien und Linien zu Blöcken zusammengefaßt. Den entstehenden Zeichen, Worten, Linien und Blöcken werden Wahrscheinlichkeiten zugeordnet.
Die Achsenorientierung wird anhand dieser Wahrscheinlichkeiten bestimmt. Das Regelsystem und die Wahrscheinlichkeitsformeln arbeiten zuverlässig und schnell. Ihre Entwicklung per Hand kann aber nicht zur Nachahmung empfohlen werden. Etliche Regeln mußten erstellt und viele Konstanten überlegt und optimiert werden, bevor die Segmentierung funktionierte. In der Hoffnung, daß mit der jeweils nächsten neuen Regel die Segmentierung geschafft sei, wurde dies von Hand gemacht und eine Automatisierung nicht in Angriff genommen. Rückblickend war dies ein Fehler. Die Segmentierung ist ein sehr komplexes Problem und läßt sich nicht ad hoc lösen. Die Anwendung von Methoden des maschinellen Lernens, zur Bestimmung von Regeln und Konstanten, scheint mir die besten Möglichkeiten zu bieten, dieses Problem in Zukunft zuverlässig, flexibel und schnell zu lösen.

Die Richtungsorientierung wird mittels einfachen Ausprobierens beider möglicher Richtungen ermittelt. Dabei hat es sich gezeigt, daß während der Bestimmung der Richtungsorientierung auf Trennversuche verzichtet werden muß. Ansonsten kommt es teilweise zu falschen Ergebnissen.

Zur Zeichenklassifizierung wurden Experimente mit verschiedenen Merkmalen und Klassifikationsverfahren gemacht.
Als Merkmale wurden Momente, Grauwertmatrizen und Caliper-Distanzen getestet. Grauwertmatrizen haben sich als die deutlich besten Merkmale herausgestellt. Ihre Unterscheidungsfähigkeit von Ziffern, Buchstaben und auch Firmenlogos ist gut. Ihr Potential Störungen zurückzuweisen ist aber unbefriedigend. Die besten Ergebnisse wurden mit Matrizen der Größe 8x8 erzielt. Für eine Prototypen sind Grauwertmatrizen als Merkmale durchaus ausreichend, für industrielle Anwendungen sollte aber auf andere Merkmale zurückgegriffen werden.
Eine große Enttäuschung war die Verwendung von 7 Moment-Invarianten nach Hu als Merkmalvektoren zur Zeichenerkennung. Die erzielten Ergebnisse passen eher zu dem Sprichwort: „Auch ein blindes Huhn findet mal ein Korn“, als daß von Erkennen gesprochen werden kann. Trotz sorgfältiger Analyse der Implementierung und der Ergebnisse konnte keine Erklärung für die katastrophalen Ergebnisse von Momenten bei dieser Anwendung gefunden werden.

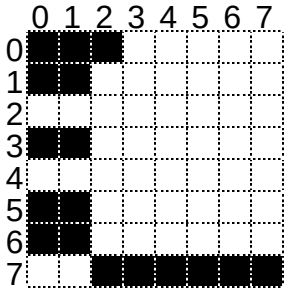
Es wurden zwei Klassifikationsverfahren getestet: das k-Nearest-Neighbour und das Hyper-Sphere Verfahren. Das k-Nearest-Neighbour Verfahren erzielt deutlich bessere Ergebnisse, als das Hyper-Sphere Verfahren. In puncto Geschwindigkeit gibt es geringfügige Vorteile beim Hyper-Sphere Verfahren.

Diese sind aber nicht ausreichend, um die geringere Erkennungsrate in Kauf zu nehmen.
Die beste Erkennungsrate von 94% bei bekannten Chips und 88% bei unbekannten Chips wurde bei der Berücksichtigung von 3 Nachbarn und einem maximalen Abstand von 4 erzielt.
Das verwendete k-Nearest-Neighbour Verfahren ist die Bremse des Systems. Durchschnittlich 75% der gesamten Verarbeitungszeit für einen Chip werden für die Bestimmung der nächsten Nachbarn verwendet. Der Einsatz eines optimierten Nearest-Neighbour Verfahrens würde sich daher lohnen.

Zum Trennen von verklebten Zeichen wurde das Nachbarverfahren und die Caliper-Distanz getestet. Die Caliper-Distanz erwies sich als deutlich besser, insbesondere bei Anwendungen mit überwiegend numerischen Zeichen. Bei Ziffern liefert die Caliper-Distanz im Mittel 0,7 falsche Trennstellen pro Ziffer, während das Nachbarverfahren 3,3 erzeugt.
In einigen seltenen Fällen versagt die Caliper-Distanz. Für diese Fälle wurde eine Erweiterung der Caliper-Distanz entwickelt, so daß für beliebig verklebte Zeichen eine Trennung möglich ist.
Die Reihenfolge, in der die möglichen Trennstellen ausprobiert werden, hat nicht nur Einfluß auf den Zeitbedarf sondern auch auf den Erfolg einer Trennung. Durch jede Trennstelle wird ein neues Zeichen unterschiedlicher Breite erzeugt. Es ist empfehlenswerter die Trennstellen so abzuarbeiten, daß erst die breiteren Zeichen getestet werden und anschließend die schmaleren. Ansonsten kommt es zu Übertrennungen. Inwieweit der Wert der Caliper-Distanz an jeder Trennstelle für die Festlegung der Reihenfolge verwendet werden kann, müssen noch weitere Untersuchungen zeigen.

5 Anhang: Negative Hu-Invarianten

Die sieben Invarianten nach Hu [Hu62] haben einen sehr großen Wertebereich. Um diesen zu reduzieren, verwenden einige Autoren den Logarithmus der Invarianten (siehe [GoWo92]). Bei meinen Tests mit Momenten wurden die Invarianten jedoch manchmal negativ. Da der Logarithmus für negative Werte nicht definiert ist, kann nicht logarithmiert werden.
An dieser Stelle wird an einem Beispiel gezeigt, daß die Hu-Momente negativ werden können. Gegeben sei das nachfolgend dargestellte 8x8 Punkte große Binärbild. Die schwarzen Kästchen haben den Wert 1, die weißen den Wert 0. Das Bild sieht nicht sehr sinnvoll aus, es gleicht keinem Zeichen. Da aber auch Störungen klassifiziert werden müssen, treten solche Bilder in der Praxis auf. Aufgrund der geringen Anzahl schwarzer Punkte ist ein Nachrechnen der *Momente* per Hand einfacher möglich.



Die Numerierung der Bildkoordinaten geht von 0-7; 0^0 ist als 1 definiert. Alternativ könnten die Koordinaten auch von 1-8 definiert werden, dann würden sich andere Momente ergeben. Die Zentralmomente, die anschließend aus den Momenten berechnet werden, sind jedoch für beide Fälle identisch, ebenso die Hu-Invarianten.

Allgemein ist ein Moment (p+q)-ter Ordnung definiert als:

$$m_{pq} \equiv \sum_{y=0}^{M-1} \sum_{x=0}^{N-1} x^p y^q g(x,y)$$

Dann ist der Wert des Momentes (p+q)-ter Ordnung für dieses Bild:

$$m_{pq} = \begin{pmatrix} 0^p \cdot 0^q & +1^p \cdot 0^q & +2^p \cdot 0^q & +0 & +0 & +0 & +0 & +0 \\ +0^p \cdot 1^q & +1^p \cdot 1^q & +0 & +0 & +0 & +0 & +0 & +0 \\ +0 & +0 & +0 & +0 & +0 & +0 & +0 & +0 \\ +0^p \cdot 3^q & +1^p \cdot 3^q & +0 & +0 & +0 & +0 & +0 & +0 \\ +0 & +0 & +0 & +0 & +0 & +0 & +0 & +0 \\ +0^p \cdot 5^q & +1^p \cdot 5^q & +0 & +0 & +0 & +0 & +0 & +0 \\ +0^p \cdot 6^q & +1^p \cdot 6^q & +0 & +0 & +0 & +0 & +0 & +0 \\ +0 & +0 & +2^p \cdot 7^q & +3^p \cdot 7^q & +4^p \cdot 7^q & +5^p \cdot 7^q & +6^p \cdot 7^q & +7^p \cdot 7^q \end{pmatrix}$$

Hieraus ergeben sich die Werte:

$m_{00} = 17$ $m_{10} = 34$ $m_{20} = 148$ $m_{30} = 796$
$m_{01} = 72$ $m_{11} = 204$ $m_{21} = 988$
$m_{02} = 436$ $m_{12} = 1394$
$m_{03} = 2796$

Die Zentralmomente sind definiert als:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x,y) \text{ mit } \bar{x} = \frac{m_{10}}{m_{00}}, \bar{y} = \frac{m_{01}}{m_{00}}$$

Sie können aber auch über die oben berechneten Momente ermittelt werden:

$\mu_{00} = m_{00} = 17$
$\mu_{10} = 0$
$\mu_{01} = 0$
$\mu_{11} = m_{11} - \bar{y} m_{10} = 60$
$\mu_{20} = m_{20} - \bar{x} m_{10} = 80$
$\mu_{02} = m_{02} - \bar{y} m_{01} = 131{,}1$
$\mu_{12} = m_{12} - 2\bar{y} m_{11} - \bar{x} m_{02} + 2\bar{y}^2 m_{10} = 13{,}76$
$\mu_{21} = m_{21} - 2\bar{x} m_{11} - \bar{y} m_{20} + 2\bar{x}^2 m_{01} = 121{,}2$
$\mu_{30} = m_{30} - 3\bar{x} m_{20} + 2\bar{x}^2 m_{10} = 180$
$\mu_{03} = m_{03} - 3\bar{y} m_{02} + 2\bar{y}^2 m_{01} = -160{,}7$

Die normalisierten Zentralmomente sind definiert als:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \text{ mit } \gamma = \frac{p+q}{2} + 1 \quad \text{für } p + q = 2,3,\ldots$$

Damit ergeben sich folgende Werte:

$\eta_{11} = 0{,}2076$
$\eta_{20} = 0{,}2768$
$\eta_{02} = 0{,}4535$
$\eta_{21} = 0{,}1017$
$\eta_{12} = 0{,}01155$
$\eta_{30} = 0{,}1511$
$\eta_{03} = -0{,}1349$

Nach den Formeln von Hu ergeben sich dann die folgenden sieben Invarianten:

$\phi_1 = 0{,}7303$
$\phi_2 = 0{,}2036$
$\phi_3 = 0{,}2071$
$\phi_4 = 0{,}02754$
$\phi_5 = -0{,}0007046$
$\phi_6 = -0{,}008960$
$\phi_7 = 0{,}0008818$

Die Formeln zur Berechnung der Hu-Invarianten können in [Hu62] nachgelesen werden, beispielhaft sei hier die Formel für das fünfte Moment angegeben:

$$\phi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2]$$
$$+(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

In diesem Beispiel sind die Werte aller Invarianten recht klein, bei anderen Bildern sind auch Werte bis zu 10^6 möglich. Aufgrund dieser großen Werte erklärt sich der Wunsch, den Wertebereich durch logarithmieren zu reduzieren.
Die Werte für das fünfte und sechste Moment sind in diesem Beispiel negativ. Wie kann daher logarithmiert werden?

6 Worterklärungen

Einige Begriffe aus der Bildverarbeitung, die in dieser Diplomarbeit verwendet, aber nicht definiert werden, werden hier erklärt und Verweise auf weiterführende Literatur gegeben.
Alle Worte, die im Text der Diplomarbeit kursiv sind, werden in diesem Kapitel genauer beschrieben.

Caliper-Distanz: →horizontale Caliper-Distanz

Fourier-Deskriptoren: Gegeben sei eine Menge von N Punkten (x_k, y_k) $0 <= k < N$, die den Umriß einer Figur in der XY-Ebene beschreiben. Betrachtet man die Punkte als Real- und Imaginärteile komplexer Zahlen s_k mit $s_k = x_k + jy_k$, können die Punkte mittels der diskreten Fourier Transformation in eine Folge von komplexen Zahlen $a(u)$, Fourier Deskriptoren genannt, transformiert werden,

mit $a(u) = \frac{1}{N}\sum_{k=0}^{N-1} s(k)\exp[-j2\pi uk/N]$ u = 0, 1, 2.., N-1

Durch die Inverse Fourier Transformation können die ursprünglichen s_k wieder berechnet werden. Die Fourier Transformation hat die Eigenschaft, daß die $a(u)$ mit kleinem u die niederfrequenten Komponenten des Umrisses enthalten, also die Teile, die das gesamte, globale Aussehen der Figur beschreiben. Die $a(u)$ mit großen u hingegen enthalten die höherfrequenten Komponenten mit den kleinen, lokalen Details der Figur. Dementsprechend kann das globale Aussehen einer Figur, unter weglassen der Details, durch einige wenige $a(u)$ beschrieben werden. Fourier Deskriptoren können daher verwendet werden, um eine variable lange Folge von Punkten einer Umrißbeschreibung in eine konstante Anzahl von komplexen Zahlen zu verwandeln, die charakteristisch sind für diesen Umriß.
Für eine weiterführende Beschreibung und ein Beispiel sei auf [GoWo92] verwiesen. Dort finden sich auch Ausführungen zum Verhalten von Fourier Deskriptoren bei Verdrehung, Verschiebung und Skalierung.

horizontale Caliper-Distanz HC: Sei HC(y) die Anzahl der Hintergrundpunkte vom Rand der y'ten Zeile bis zum ersten Vordergrundpunkt. HC ist der Vektor aller HC(y). Die Caliper-Distant kann jeweils für den linken und rechten Rand berechnet werden. Analog ist die vertikale Caliper-Distanz CV über die Spalten definiert.

horizontale Projektion H: Sei H(y) die Anzahl der schwarzen Punkte in der y'ten Zeile eines Bildes. H ist der Vektor aller H(y) eines Bildes. Analog ist die vertikale Projektion V über die Spalten definiert.

Hough Transformation: Mittels der Hough Transformation kann effizient ermittelt werden, welche Punkte auf einer Geraden liegen. Ein Punkt (x_i, y_i) liegt auf unendlich vielen Geraden, die alle durch die Geradengleichung: $y_i = a*x_i + b$ gegeben sind. Durch Umformung der Geradengleichung nach $b = -x_i*a + y_i$ wird aus dem Punkt (x_i, y_i) in der ab-Ebene eine einzige Gerade. Ein weiterer Punkt (x_j, y_j) wird analog zu einer weiteren Geraden in der ab-

Ebene. Die beiden Geraden der Punkte schneiden sich in der ab-Ebene in genau einem Punkt (a', b'). a' ist die Steigung und b' der Y-Abschnitt der Geraden auf der die beiden Punkte (x_i, y_i) und (x_j, y_j) in der XY-Ebene liegen. Verallgemeinert gilt: Alle Punkte die auf einer Geraden mit Steigung a und Y-Abschnitt b liegen, werden in der ab-Ebene abgebildet auf Geraden, die sich alle im Punkt (a, b) schneiden. Zur Suche nach Punkten, die auf einer Geraden liegen, wird die ab-Ebene in ein zweidimensionales Raster eingeteilt und für jeden Punkt und für jedes a das entsprechende b berechnet. Die Zelle im Raster mit den meisten Punkten entspricht dann der Geraden, auf der die meisten Punkte in der XY-Ebene liegen.
Da Steigung und Y-Abschnitt für vertikale Geraden gegen unendlich gehen, empfiehlt es sich nicht mit der Geradengleichung $b = -x*a + y$ zu arbeiten, sondern besser die Gleichung $\rho = x \cos\theta + y \sin\theta$ zu verwenden, mit $-90 \le \theta < 90$ und $\rho \in \pm\sqrt{2}D$, D ist die Länge der Bilddiagonalen.
Die Hough Transformation kann nicht nur für Geraden, sondern für beliebige Formen verwendet werden, siehe [Ball81] und [DuHa72]. Zur Vertiefung und für ein Beispiel sei auf [GoWo92] verwiesen.

k-Nearest-Neighbour: siehe Seite 34

Lauflängenkodierung: Dies ist eine Methode zur kompakten Speicherung eines Binärbildes. Anstatt ein Bild als lange Folge von Nullen und Einsen zu beschreiben, wird nur gespeichert wieviele Nullen nacheinander folgen, anschließend wieviele Einsen, dann wieder die Anzahl der Nullen usw. Die Binärfolge „0001111011100000" würde lauflängenkodiert „34135" lauten. Ein lauflängenkodiertes Bild ist nicht nur meistens kompakter als das Originalbild, sondern kann auch zur Analyse des Bildes verwendet werden. So korrespondieren große Werte im lauflängenkodierten Bild zu langen weißen oder schwarzen Strichen. Ähnlich wie mit der Berechnung von Kettenkodes lassen sich lauflängenkodierte Bilder auch zur Bestimmung von zusammenhängenden Punktmengen nutzen.

Median: Der Median M von n Werten ist der Wert für den gilt, daß genau n/2 Werte größer oder gleich M sind.

Momente: Momente werden in der Statistik benutzt, um die Verteilung von Zufallszahlen zu charakterisieren. In der Statistik haben die hierzu verwendeten Momente besondere Namen: Erwartungswert und Varianz. Das zweidimensionale kartesische Momente m_{pq} der Ordnung p+q einer diskreten Dichtefunktion f(x,y) ist definiert als:

$$m_{pq} \equiv \sum_{y=0}^{M-1}\sum_{x=0}^{N-1} x^p y^q g(x,y)$$

Ein Erwartungswert ist damit das Moment 1. Ordnung einer eindimensionalen Dichtefunktion.
Hu [Hu62] bewies mit seinem Eindeutigkeit-Theorem: Wenn f(x,y) stückweise kontinuierlich und nur in einem endlichen Bereich ungleich Null ist, existieren die Momente aller Ordnung. Sie sind eindeutig durch f(x, y) bestimmt und umgekehrt ist f(x, y) eindeutig durch alle seine Momente bestimmt. Digitale Bilder erfüllen diese Bedingungen und es bietet sich daher an, Bilder durch ihre Momente zu beschreiben. Es können dazu natürlich nicht alle Momente

verwendet werden, sondern es muß eine Menge von als besonders charakterisierend bekannten Momenten ausgewählt werden. Wie in der Statistik, können Momente von Bildern auch interpretiert werden. Das Moment m_{00} gibt die Masse des Bildes an. Die Momente m_{01} und m_{10} jeweils dividiert durch die Masse sind die Koordinaten des Masseschwerpunktes eines Bildes. Werden Momente relativ zum Masseschwerpunkt berechnet, so werden die Momente als Zentralmomente bezeichnet und es ergibt sich folgende Formel:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x,y) \text{ mit } \bar{x} = \frac{m_{10}}{m_{00}}, \bar{y} = \frac{m_{01}}{m_{00}}$$

Die Varianz ist damit das Zentralmoment 2ter Ordnung einer eindimensionalen Dichtefunktion.

Für eine ausführlichere Darstellung und die Interpretation weiterer Momente sei auf die sehr gute Arbeit von Prokop und Reeves [PrRe92] verwiesen. In dieser Arbeit sind auch die verschieden auf Momenten basierende Ansätze zur Bilderkennung zusammengefaßt und erläutert. Prokop und Reeves unterscheiden fünf Momenttechniken: Moment-Invarianten, Rotationsmomente, Orthogonalmomente, Komplexemomente und Standardmomente. Die zu den Orthogonalmomenten zählenden Zernike und pseudo-Zernike Momente gelten als die Besten. Sie erfordern aber auch den größten Rechenaufwand. Alle Momenttechniken sind invariant gegenüber Änderungen von Größe, Lage und Orientierung.

Neuronales Netz: siehe Seite 34

Projektion: →horizontale Projektion

vertikale Caliper-Distanz: →horizontale Caliper-Distanz

vertikale Projektion: →horizontale Projektion

7 Literatur

[Albe95] Stefan Albers: „Entwicklung und Implementierung eines Bildverarbeitungssystems zur Identifizierung recyclingrelevanter Regionen auf Leiterplatten in ein automatisiertes Gesamtsystem zur Entstückung von Leiterplatten“; Diplomarbeit Universität Dortmund Fachbereich Elektrotechnik Lehrstuhl Steuer- und Regelungstechnik 1995

[Ball81] D. H. Ballard: "Generalizing the Hough transform to detect arbitrary shapes."; in Pattern Recognition Vol. 13, S. 111-122, 1981

[Boks92] Mindy Bokser: "Omnidocument Technologies"; in [PIEEE80] S. 1066-1078

[CaHa87] G. L. Cash, M. Hatamian: "Optical character recognition by the method of moments"; in Comput. Vision Graphics Image Process. 39, S. 291-310, 1987

[CCEE94] V. Consorti, L. P. Cordella, V. Eramo, D. Perifano: "Knowledge Based Search of Character Strings in Line Drawings"; in [IAPR94b] S. 589

[CCLT94] V. Consorti, L.P. Cordella, T. Lavorgna, F. Tortorella: "Singling out Character Strings in Map Analysis"; in "Progress in Image Anaysis and Processing III" S. Impedevo Editor, World Scientific Publ. S. 63-70, 1994

[CVGIP54] "Graphical Models and Image Processing": CVGIP Vol. 54, Nr. 5, September 1992

[CVPR94] "IEEE Computer Society Conference on computer vision and pattern recognition": Prcoeedings CVPR 1994; ISBN 0-8186-5825-8

[DeCh94] Rutvik Desai, H. D. Cheng: "Pattern Recognition by Local Radial Moments"; in [IAPR94b], S. 168-172

[DuHa72] R. O. Duda, P. E. Hart: "Use of the Hough transform to detect lines and curves in pictures"; in Graphics and Image Processing Vol. 15, S. 11-15, 1972

[FlKa88] LLoyd Alan Fletcher, Rangachar Kasturi: "A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images"; in [PAMI10] S.910-918

[FuNK92] H. Fujisawa, Y. Nakano, K. Kurino: "Segmentation Methods for Character Recognition: From Segmentation to Document Structure Analysis" in [PIEEE80], S. 1079-1092

[GoWo92] Rafael C. Gonzalez, Richard E. Woods: "Digital Image Processing"; Addison-Wesley, Massachsetts 1992; ISBN 0-201-50803-6

[Groß90] R. E. Großkopf (Hrsg.): „Mustererkennung“ 12. DAGM-Symposium; Springer-Verlag 1990; ISBN 3-540-53172-6

[Hara94] R. M. Haralick: "Document Image Understanding: Geometric and Logical Layout"; in [CVPR94] S. 385

[HaSh92] R. M. Haralick, L. G. Shapiro: „Computer and Robot Vision“ Volume I + II; Addison-Wesley Publishing Company 1992, Massachusetts; ISBN 0-201-10877-1 (Volume I) 0-201-56943-4 (Volume II)

[HöDe90] Frank Hönes, Andreas Dengel: „ Ein kontext-restriktiver Ansatz zur Texterkennung“ in [Groß90]

[Hu62] M. K. Hu: "Visual pattern recognition by moment invariants"; in IRE Transaction on Information Theory Februar 1962, S. 179-187

[IAPR94b] "12th International Conference on Pattern Recognition" 1994 Volume 2; ISBN 0-8186-6270-0

[KaPB87] Simon Kahan, Theo Pavlidis, Henry S. Baird: "On the Recognition of Printed Characters of Any Font and Size"; in [PAMI9] S.274-287

[KiYu94] Whoi-Yul Kim, Po Yuan: "A Practikal Pattern Recognition System for Translation, Scale and Rotation Invariance"; in [CVPR94], S. 391-396

[KlZa95] Reinhard Klette, Piero Zamperoni: „Handbuch der Operatoren für die Bildverarbeitung“; Vieweg 1995, Braunschweig; ISBN 3-528-16431-X

[LiEn89] C.-E. Liedtke, M. Ender: "Wissensbasierte Bildverarbeitung"; Springer Verlag 1989; ISBN 3-540-50641-1

[MiST94] D. Michie, Spiegelhalter, C. Taylor: „Machine Learning, Neural and Statistical Classification“; Ellis Horward Series in Artifical Intelligence 1994, New York; ISBN 0-13-106360-X

[Möll96] Thomas Möller: „Beitrag zur automatisierten selektiven Entstückung von Leiterplatten“; Shaker Verlag, Aachen 1996; ISBN 3-8265-1544-7

[MoSY92] Shunji Mori, Ching Y. Suen, Kazuhiko Yamamoto: "Historical Review of OCR Research and Development" in [PIEEE80], S. 1029-1058

[Otsu79] N. Otsu: "A threshold selection method from gray-level histograms."; in IEEE Trans. SMC-9, Seite 62-66; 1979

[PAMI9] "IEEE Transaction on Pattern Analysis and Machine Intelligence."; Volume PAMI-9, No. 2, March 1987

[PAMI10] "IEEE Transaction on Pattern Analysis and Machine Intelligence."; Volume PAMI-10, No. 6, November 1988

[PIEEE80] Proceedings of the IEEE Vol. 80, No. 7, July 1992

[PrRe92] Richard J. Prokop, Anthony P. Reeves: "A Survy of Moment-Based Techniques for Unoccluded Object Representation and Recognition"; in [CVGIP54], S. 438-460

[Sche87] W. Scherl: "Bildanalyse allgemeiner Dokumente"; Informatik Fachberichte 131; Springer Verlag 1987; ISBN 3-540-17214-9

[SiPö92] H. W. Simon, S. Pölt: Endbericht der Projektgruppe „Lamex-Z“; Universität Dortmund 1992

[TsAs92] S. Tsujimoto, H. Asada: "Major Components of a Complete Text Reading System" in [PIEEE80], S. 1133-1149

[WeDi95] D. Wettscherek, T. G. Dietterich: „An Experimental Comparsion of the Nearest Neighbour and Nearest Hyperrectangle Algorithms“; in Machine Learning Volume 19 Number 1 April 1995

[Yuech92] Yuechiro Amzai: „Pattern Recognition and Machine Learning“; Academic Press, Inc. 1992, Boston; ISBN 0-12-058830-7