Ein intuitives Verfahren zur adaptiven merkmalsgestützten Segmentierung

Inauguraldissertation

zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften
der Universität Mannheim

vorgelegt von

Dipl.-Inf. Michael Beller
aus Mannheim

Mannheim, 2005

Dekan:	Professor Dr. Matthias Krause, Universität Mannheim
Referent:	PD Dr. Jürgen Hesser, Universität Mannheim
Korreferent:	Professor Dr. Hartmut Gemmeke, Forschungszentrum Karlsruhe

Tag der mündlichen Prüfung: 27. Januar 2006

Zusammenfassung

Bei der Automatisierung einer Objekterkennungsaufgabe wird eine Prozeßkette bestehend aus Vorverarbeitung, Segmentierung, Merkmalsextraktion, Merkmalsselektion und Klassifikation aufgebaut. Im Allgemeinen werden hierfür manuell die geeigneten Algorithmen und ihre Parameter ausgewählt und konfiguriert, bis man mit den Ergebnissen zufrieden ist. Als Parameter werden meist spezifische Eigenschaften oder Merkmale der Objekte verwendet.

Die Auswahl insbesondere der Parameter der Algorithmen sollte auf einer ausreichend großen, repräsentativen Stichprobe stattfinden, um für alle zu erkennenden Objekte einer Domäne gültig zu sein. Dies ist durch die manuelle Auswahl meist nicht gegeben. Zwar kann der Experte intuitiv gut segmentieren und klassifizieren, jedoch die benötigten Parameter nicht mathematisch präzise formulieren. Ein Algorithmus kann ohne den Experten nicht so wie von diesem gewünscht segmentieren, die Parameter aus einer vorliegenden Segmentierung aber einfach ermitteln – er ist ihm in der Extraktion und statistischen Selektion quantitativ meßbarer Merkmale überlegen.

Im Rahmen dieser Arbeit wurde eine Methodik entwickelt, bei der ein Experte intuitiv Beispiele präsentiert, aus denen der Algorithmus problembezogene Segmentierungsparameter ableitet. Indem ausreichend Beispiele gegeben werden, lernt der Algorithmus, die zu erkennenden Objekte zur Zufriedenheit des Experten zu segmentieren und ihn bei der Automatisierung zu unterstützen.

Diese neue Methodik wurde erfolgreich auf verschiedene Probleme angewendet. Für künstliche Bilder ergaben sich Übereinstimmungen der Segmentierungen zu Referenzsegmentierungen von im Mittel 95 %. Für die Brustkrebserkennung konnten Detektionsraten von bis zu 97 % erzielt werden. Für die Segmentierung von Kaumuskeln auf MRT-Daten konnten automatisch Parameter gefunden werden, mit denen die manuelle Segmentierung zu 99 % reproduziert werden konnte.

Weiterhin wurde die Methode zur Segmentierung natürlicher und künstlicher Szenen verwendet und bei der Detektion von Lunkern in Spritzgußwerkstücken getestet. Die Ergebnisse konnten hier nur subjektiv bewertet werden.

Bisher kommen nur lokale Merkmale von Grauwertbildern zum Einsatz. Aufgrund ihres generalisierenden Ansatzes können mit der entwickelten Methodik beliebige lokale Merkmale (z.B. Farbe) der Objekte als Parameter verwendet werden.

Danksagung

An dieser Stelle danke ich allen, die am Gelingen dieser Arbeit durch ihren fachlichen Rat und ihre Hilfe oder durch Motivation beigetragen haben. Insbesondere gilt mein Dank

- Prof. Dr. Hartmut Gemmeke für die mir gebotene Möglichkeit, diese Arbeit am Forschungszentrum Karlsruhe anzufertigen, das in mich gesetzte Vertrauen und das Korreferat dieser Arbeit.

- Dr. Jürgen Hesser für die wissenschaftliche Unterstützung, die konstruktiven Hilfen zur Ausarbeitung und die Bereitschaft, diese Arbeit zu bewerten.

- Dr. Rainer Stotzka für die fachlichen Diskussionen und die Unterstützung in allen Bereichen.

- Dipl.-Inform. Tim Müller für jegliche Hilfe in softwaretechnischen Fragen, unsere Diskussionen und den Cappuccino.

- Den studentischen Mitarbeitern Michael Sutter und Torsten Hopp, die durch Diplom- und Studienarbeiten wichtige Beiträge leisteten und Teile der Software erstellten.

- Allen Mitarbeitern des Instituts für Prozeßdatenverarbeitung und Elektronik, insbesondere der Abteilung Softwaremethoden für die vielfältige Unterstützung.

Mein besonderer Dank gilt meinen Eltern und meiner Familie für ihre Liebe, ihre Unterstützung und ihr Vertrauen. Außerdem danke ich all meinen Freunden, die mir bewußt und unbewußt in so vielfältiger Weise bei dieser Arbeit geholfen haben.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Symbolverzeichnis

Allgemeine Symbole

(x, y)	Koordinaten im $\mathfrak{L}_2$
(x, y, z)	Koordinaten im $\mathfrak{L}_3$
$f(x, y)$	Bildpunkt an der Stelle (x, y)
$f(x, y, z)$	Bildpunkt an der Stelle (x, y, z)
$g(x, y)$	Grauwert an der Stelle (x, y)
$g(x, y, z)$	Grauwert an der Stelle (x, y, z)
$r(x, y)$	Regionsnummer an der Stelle (x, y)
$r(x, y, z)$	Regionsnummer an der Stelle (x, y, z)
C_D	Dice-Coeffizient
E	Schätzfehler
E_T	Trainingsfehler
E_{Te}	Testfehler
E_G	Generalisierungsfehler

Abkürzungen

Merkmalsextraktoren

Die hier beschriebenen Abkürzungen beziehen sich auf die englischen Namen der Merkmale, deren Definitionen in Anhang A zu finden sind. Die Namen wurden konsistent zur Literatur in Englisch belassen, da oft keine allgemein gültige deutsche Übersetzung bekannt ist.
S kennzeichnet ein statistisches Merkmal, **T** ein Texturmerkmal.

Statistische Merkmale

SMean	Mean
SVar	Variance
SStdDev	Standard Deviation
SMin	Minimum
SMax	Maximum
SSkew	Skewness
SSkewMeas	Skewness Measure
SKurt	Kurtosis
SKurtMeas	Kurtosis Measure
SEner	Energy
SEntr	Entropy
SAnisoCoeff	Anisotropic Coefficient
SGS	Grey Spread
SVarCoeff	Variation Coefficient
SNumGV	Number Grey Value
SModeGV	Mode Grey Value
SMed	Median

Texturmerkmale

TCov	Covarianz
TAbsVal	Absolute Value
TEner	Energie Cooccurrence Matrix
TEntr	Entropy Cooccurrence Matrix
TMaxProb	Maximum Probability
TMean	Mean
TVar	Variance
TStdDev	Standard Deviation
TCorr	Correlation
TCont	Contrast
TSumAvg	Sum Average
TSumEntr	Sum Entropy
TSumVar	Sum Variance
TDiffVar	Difference Variance
TDiffEntr	Difference Entropy
TIMCorr1	Information Measure Correlation 1
TIMCorr2	Information Measure Correlation 2
TMaxCorrCoeff	Max Correlation Coefficient
TClustShade	Cluster Shade
TClustProm	Cluster Prominence
THom	Homogeneity
TInvDiffMom	Inverse Difference Moment
TRlGLDist	Run-length Grey Level Distribution
TRlDist	Run-length Distribution
TRlRunPerc	Run-length Run Percentage
TRlSRE	Run-length Short Run Emphasis
TRlLRE	Run-length Long Run Emphasis

Kapitel 1

Einleitung

In vielen Anwendungen im Bereich der digitalen Bildverarbeitung ist die Erkennung spezifischer Muster ein wichtiger Bestandteil [1], der üblicherweise von menschlichen Experten durchgeführt wird. Der Experte ist aufgrund seines Wissens und seiner Erfahrung in der Lage, auf Bildern vorhandene Muster oder Objekte zu erkennen, sie vom Rest des Bildes abzugrenzen und einer Kategorie zuzuweisen. Für große Datenmengen ist diese manuelle Form der Bearbeitung sehr zeitaufwendig; die Ergebnisse verschiedener Experten können sich signifikant unterscheiden [2, 3, 4]. Da Maschinen im Allgemeinen reproduzierbar und objektiv arbeiten, wird in vielen Fällen die Automatisierung der Erkennung angestrebt. Als Beispiele hierfür seien die industrielle Sichtprüfung und die Tumordetektion in der Medizin genannt. Abbildung 1.1 zeigt das prinzipielle Aussehen eines automatischen Systems zur Erkennung von Mustern.

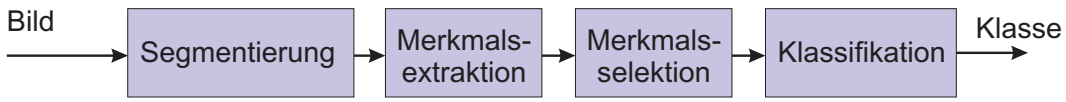


Abbildung 1.1: *Prinzipieller Aufbau einer Bildverarbeitungskette. Ausgehend von einem digitalen Bild wird eine Vorverarbeitung durchgeführt. Die Segmentierung grenzt Objekte voneinander ab, deren Merkmale berechnet werden. Wichtige Merkmale werden selektiert und die Objekte von einem Klassifikator kategorisiert, so daß als Endergebnis die Klassenzugehörigkeit jedes auf dem Bild enthaltenen Objektes steht.*

Mit bekannten Daten (Trainingsdaten) wird eine Verarbeitungskette aus verschie-

denen Algorithmen zusammengestellt. Nach einer Vorverarbeitung, beispielsweise mit digitalen Filtern, legt die Segmentierung die Objektgrenzen fest und trennt Objekte vom Hintergrund. Aus segmentierten Objekten wird mit der Merkmalsextraktion eine formale Beschreibung abgeleitet. Eine besonders zutreffende Beschreibung wird durch die Merkmalsselektion ermittelt. Im Klassifikationsschritt wird jedes Objekt einer Klasse zugewiesen. Die Algorithmen der Kette werden solange optimiert, bis der Experte mit den Ergebnissen zufrieden und die Erkennungsrate möglichst hoch ist. Die erstellte Kette kann auf unbekannte Bilder (Testdaten) angewendet werden. Der Mustererkennungsprozeß ist somit automatisiert.

Der menschliche Experte kann mit seinem Wissen und seiner Erfahrung sehr gut segmentieren und klassifizieren. Ein Rechner ist dem Experten jedoch in der Extraktion und Selektion numerischer Merkmale überlegen. Damit ein System selbsttätig segmentieren und klassifizieren kann, müssen für jedes Problem geeignete Algorithmen und Parameter verwendet werden [5]. Da ein universeller, für alle Erkennungsprobleme gleichsam geeigneter Algorithmus nicht existiert [6], muß für die meisten Probleme ein neues System entworfen werden.

Während in der Vorverarbeitung einfache Verfahren, z.B. zur Rauschunterdrükkung, zum Einsatz kommen, wird in der Segmentierung bereits eine Semantik induziert, d.h. es wird zwischen verschiedenen Mustern im Bild unterschieden, z.B. Objekte und Hintergrund. Eine gute Erkennung dieser Muster ist somit von einer guten Segmentierung abhängig [7, 8]. Auch umgekehrt ist die Segmentierung eines Musters von seiner Erkennbarkeit abhängig. Die bisherige Art und Weise des Aufbaus der Bildverarbeitungkette spiegelt diese Dualität nicht wider.

1.1 Problemstellung

Aufgrund ihrer Bedeutung wird die Segmentierung als wichtigster Schritt der Kette betrachtet, von dessen Ergebnis alle Folgeschritte abhängen [7, 8]. Die Qualität einer automatischen Segmentierung läßt sich an ihrer Übereinstimmung zur manuellen – als korrekt erachteten – Segmentierung eines Experten (Goldstandard) messen. Die manuelle Segmentierung ist ein zeitaufwendiger Prozeß mit Ergebnissen, die abhängig vom Experten stark schwanken können [2, 3, 4].

Eine automatische Segmentierung hingegen könnte reproduzierbare Ergebnisse liefern. Fehler in der Segmentierung ziehen sich in Form falscher Merkmale oder ungenauer Werte der Merkmale durch die Kette und können das Klassifikationsergebnis stark verfälschen. Eine optimale Segmentierung ist die Voraussetzung für ein optimales Klassifikationsergebnis.

Ein System soll derart segmentieren und klassifizieren können, daß der Experte zufrieden ist. Hierfür existieren viele verschiedene Algorithmen für die einzelnen Stufen der Bildverarbeitungskette. Die Auswahl geeigneter Algorithmen ist abhängig vom jeweiligen Erkennungsproblem [5]. In der Literatur erfolgt diese Auswahl für jede Stufe manuell. Das Erkennungsproblem und die Funktionsweise des einzusetzenden Algorithmus müssen bekannt sein, um die jeweiligen Parameter korrekt einzustellen [5, 9, 10], z.B. Schwellwerte für die Segmentierung. Auch dem Experten ist es kaum möglich, komplexe Bildinhalte mathematisch präzise zu beschreiben. Die zur Erkennung der Muster benötigten Merkmale werden subjektiv ausgewählt; häufig geschieht dies auf Basis der menschlichen Wahrnehmung der Muster (z.B. in [11, 12, 13]). Mit dieser Form der Parameterwahl kann nicht garantiert werden, daß alle benötigten Parameter ermittelt worden sind.

Die Auswahl der benötigten Parameter sollte auf einer ausreichenden Menge von bekannten, vorsegmentierten Objekten erfolgen [14]. Ist die Variabilität der Objekte sehr hoch, ist diese Menge meist zu gering, um den entsprechenden Anwendungsfall adäquat zu repräsentieren. Eine korrekte Adaption des Klassifikators und eine gute Generalisierung können dann nicht ermöglicht werden. In der Praxis sind diese Trainingsdaten aufgrund des zeitlichen Aufwands selten verfügbar, weshalb das Segmentierungsmodell nicht zum kompletten Datensatz paßt.

Benötigt wird hier eine Methodik, die den Experten interaktiv unterstützt, Trainingsdaten für den jeweiligen Anwendungsfall zu gewinnen. Zusätzlich soll die Methodik in der Lage sein, die geeigneten Parameter automatisch und objektiv auszuwählen.

Die Optimierung der eingesetzten Algorithmen erfolgt in der Literatur fast durchweg isoliert voneinander [15]; Zusammenhänge zwischen den einzelnen Verarbeitungsstufen und insbesondere die Abhängigkeit von der Segmentierung werden vernachlässigt. So entsteht eine komplexe Verarbeitungskette, die nur auf ein einzelnes Problem anwendbar ist. Für jedes neue Problem müssen die Algorithmen

der einzelnen Stufen aufwendig angepaßt oder komplett neu entworfen werden. Notwendig ist hier eine flexible Methodik, welche die einzelnen Stufen problemabhängig modifiziert, dabei Zusammenhänge zwischen den einzelnen Stufen besser beachtet und so den gesamten Ablauf hinsichtlich der Segmentierung und daraus folgend der Klassifikationsergebnisse optimiert.

1.2 Gliederung der Arbeit

Kapitel 2 *Stand der Technik* gibt einen Überblick über den aktuellen Stand der Technik. Zunächst wird der grundsätzliche Aufbau von Mustererkennungssystemen erläutert. Daran anschließend wird auf Algorithmen und Systeme aus der Literatur eingegangen und daraus die Ziele der Arbeit abgeleitet.
In Kapitel 3 *Grundlagen* werden die zum Aufbau des eigentlichen Systems benutzten Konzepte zur Segmentierung, Merkmalsextraktion, -selektion und Klassifikation sowie Lernkonzepte erläutert. Daneben wird ein Framework beschrieben, innerhalb dessen die Softwareimplementierung stattfindet.
In Kapitel 4 wird ein System zur Mustererkennung entwickelt, das durch einfache Extraktion lokaler Muster und einfachem, interaktivem Lernen von Beispielen auf vielfältige Erkennungsprobleme trainiert werden kann.
Es besteht hauptsächlich aus einem neu entwickelten adaptiven Segmentierungsalgorithmus, der auf Basis der überwachten Extraktion lokaler Muster die benötigten Segmentierungsparameter lernt. Desweiteren wird die mögliche Kombination mit einem Objekterkennungsschritt mittels globaler Eigenschaften erläutert.
Kapitel 5 *Erzielte Ergebnisse* zeigt diejenigen Ergebnisse, die mit dem entwickelten System für vielfältige Fragestellungen erzielt werden können. Dabei werden synthetische und natürliche Erkennungsprobleme betrachtet.
Eine Diskussion findet in Kapitel 6 *Diskussion und Ausblick* statt. Dort werden der Ansatz kritisch betrachtet und die Einsatzmöglichkeiten rekapituliert. Die Arbeit schließt mit einer Zusammenfassung der Beschränkungen des Ansatzes und einem Ausblick für weitere Arbeiten.
In Anhang *A* finden sich mathematischen Berechnungsvorschriften der verwendeten Merkmale.

Kapitel 2

Stand der Technik

Um automatische, reproduzierbare Segmentierungsergebnisse zum Zwecke der Mustererkennung zu erzielen, muss insbesondere sichergestellt werden, daß geeignete Algorithmen und Parameter verwendet werden. Diese Ergebnisse sollen den Anforderungen des Experten gerecht werden. Die Güte dieser Ergebnisse kann in vielen Fällen nur subjektiv vom Experten bewertet werden. Es wäre daher wünschenswert, über ein System zu verfügen, welches die Auswahl automatisch und abhängig vom jeweiligen Erkennungsproblem treffen kann. Um das Erkennungsproblem gut zu generalisieren, sollte das System zusätzlich über ausreichende Trainingsdaten verfügen. In den folgenden Abschnitten werden kurz grundlegende Anforderungen an ein solches System zusammengetragen und die in der Literatur verfügbaren Methodiken und Systeme diskutiert. Abschließend werden daraus die Ziele der Arbeit abgeleitet.

2.1 Anforderungen

Aufgrund der geschilderten Problemstellung (vgl. 1.1) lassen sich die Anforderungen an ein System, das Muster automatisch im Sinne des Experten erkennt, folgendermaßen zusammenfassen:

- **Automatische Segmentierung und Erkennung der Muster auf Bildern:** Das System soll Muster automatisch „gut" erkennen. Dazu soll es die Muster optimal im Bild isolieren und die benötigten Eigenschaften extrahieren, um ein optimales Klassifikationssystem für die Muster aufzubauen.

- **Automatische Wahl geeigneter Parameter:** Um die Verarbeitungskette für alle Muster eines Problemkreises anwenden zu können, müssen die Parameter der einzelnen Schritte entsprechend ermittelt werden [9, 10]. Dies ist nur dann möglich, wenn eine hinreichende Beschreibung der Muster oder eine ausreichende Zahl von Trainingsmustern zur Verfügung steht [14].

- **Lernfähigkeit:** Häufig sind weder eine Beschreibung der Muster noch eine ausreichende Anzahl von Trainingsmustern verfügbar. Die Informationen über die Muster können zudem nach und nach einfließen. Das System soll daher lernfähig sein [4, 7].

- **Interaktivität:** Der Experte präsentiert dem System in einem überwachten Lernvorgang [9] Beispiele der vorgegebenen Objektklassen. Um den Lernfortschritt bewerten und korrigierend eingreifen zu können, soll das System interaktiv sein.

- **Allgemeiner Ansatz:** Ein Mustererkennungssystem sollte nicht auf eine Anwendung beschränkt, sondern für viele verschiedene Probleme einsetzbar sein. Daher sollte die Auswahl der Algorithmen der einzelnen Verarbeitungsstufen (Segmentierung, Merkmalsextraktion, Merkmalsselektion und Klassifikation) und deren Optimierung aufeinander automatisiert werden. Ein möglichst allgemeiner Ansatz kann diese Anforderung erfüllen.

2.2 Bestehende Systeme und Methoden

In der Literatur finden sich einige Ansätze zur Erstellung eines solchen Mustererkennungssystems. Häufig soll damit die Verarbeitungskette optimiert werden. Das Hauptaugenmerk liegt hier auf den Segmentierungsalgorithmen (vgl. 1.1).

2.2.1 Systeme

Zwei generalisierende Ansätze, mit denen die dargelegten Erfordernisse erfüllt sein sollen, vertreten die Systeme *CYCLOPS* [16] und *Gipsy* [17, 18]. Beide Systeme stellen Umgebungen zur Verfügung, innerhalb derer die komplette Verar-

beitungskette modelliert werden kann. Konzipiert sind sie für die medizinische Bildverarbeitung.

Cyclops soll unter anderem die automatische Erkennung von Objekten in Bildern durchführen [19]. Es kontrolliert vollständig die Auswahl, Parametrisierung und Koordination der Bildverarbeitungsverfahren. Dies geschieht mit statischen und dynamischen Wissensbasen. Anhand von globalen und lokalen Attributen entscheidet eine Inferenzmaschine (eine Maschine zur Anwendung von Wissen), welche Verfahren und Parameter für ein Bild am besten geeignet sind. Dies geschieht mit Hilfe von Fallbasen und Regeln. Die eigentliche Optimierungsstrategie bleibt unklar. Das System wurde speziell für medizinische Bilddaten entworfen. Es bleibt offen, ob es in anderen Kontexten angewendet werden kann. Eine Strategie, um aus Fehlern zu lernen, gibt es nicht.

Gipsy ist ein System zur wissensbasierten Adaption eines echtzeitfähigen Sichtprüfsystems [17]. Aufgrund der Aufgabenstellung wählt Gipsy aus einer Wissensbasis die verfügbaren Bildverarbeitungsoperatoren aus. Es werden Verarbeitungsketten konfiguriert, welche die gestellte Aufgabe voraussichtlich am besten erfüllen. Die Parameter der gewählten Operatoren werden iterativ verändert, bis die Ergebnisse für die Aufgabenstellung optimal sind. Möglichst repräsentative Lernbilder werden mit speziellen Operatoren segmentiert und die Merkmale der segmentierten Objekte mit Benutzervorgaben verglichen. Die Adaption der Parameter erfolgt mit hochsprachlichen Beschreibungen und WENN-DANN-Regeln (z.B. WENN Kompaktheit zu stark, DANN verringere Operatorfenstergröße). Aufgrund des Vergleichs mit Benutzervorgaben werden hochsprachliche Merkmale verwendet. In diesem System müssen sehr viele Regeln implementiert sein, um alle möglichen Einflüsse derart komplexer Regeln überprüfen zu können. Desweiteren bleibt unklar, ob Regeln in dieser Form numerisch hinreichend genau implementiert werden können, um eine vollständige automatisierte Objekterkennung zu gewährleisten. Methoden zur Erweiterung des Regelsatzes für komplexe Anwendungen sind nicht beschrieben.

Andere Systeme [2, 14] sind fallbasierte Entscheidungssysteme (Case-Based Reasoner (CBR)). Solche Systeme wählen aufgrund von Merkmalen den naheliegendsten bereits gelösten Fall aus und verwenden dessen Parameter für das aktuelle Problem. Die verwendeten Algorithmen und Parameter sind fest vorgegeben. Bei

unzureichenden Ergebnissen muß der Benutzer die einzelnen Parameter selbsttätig optimieren, bevor diese in die Fallbasis einfließen.
Im folgenden werden Methoden zur Optimierung der Segmentierung und Objekterkennung diskutiert.

2.2.2 Methoden zur Segmentierung

Bei der Segmentierung wird ein Bild in eine Menge von Regionen partitioniert, von denen jede ein semantisches Objekt enthält. Eine der gebräuchlichsten Methoden der Segmentierung ist das Regionenwachstum (engl. Seeded Region Growing) [20]. Ausgehend von einem Saatpunkt werden zwei benachbarte Pixel zusammengefaßt, wenn ihr Gradient unter einer bestimmten Schwelle liegt [21, 22, 23]. Die entstehenden Regionen sind dann homogen im Sinne des Gradienten. Die Verwendung des Gradienten limitiert die Methode sehr stark. Viele semantische Objekte zeichnen sich nicht durch einen homogen Gradienten aus. Durch geschickte Wahl der Schwellwerte (z.B. in Abhängigkeit von der Position im Bild) wird versucht, die durch die Verwendung des Gradienten bestehende Limitierung zu durchbrechen. Der Gradient bleibt für komplexe Bildinhalte ungeeignet.
Das Verfahren kann jedoch einfach erweitert werden, um diese Schwäche zu eliminieren. Es existieren Methoden, ein Regionenwachstum mit (zusätzlichen) vom Gradienten verschiedenen Randbedingungen zu versehen. Dadurch entstehen andere Segmentierungen als beim normalen Region Growing. Diese Form der Segmentierung wird bisher als Constrained Region Growing (CRG) [12, 20, 22] bezeichnet. Warfield et al. beschreiben ein solches CRG in [12]. Sie verwenden zusätzlich zu beobachteten Unterscheidungsmerkmalen Expertenwissen in Form eines anatomischen Atlasses zur Formulierung ihrer Randbedingungen (engl.: constraints).
Pohle et al. [24] schätzen mittels des Gradienten und einfacher morphologischer Eigenschaften die Homogenität während der Segmentierung ständig neu und passen die Segmentierungsparameter dementsprechend an.
Paclík et al. [25] benutzen spektrale und spatiale Merkmale für eine unüberwachte Textursegmentierung von Rückstreuungsbildern von Reinigungsmitteln. Aus der Initialsegmentierung werden iterativ neue Klassenzugehörigkeitswahrschein-

lichkeiten für jedes Pixel berechnet. Der Pixelklassifikator der letzten Iteration wird zur Initialsegmentierung weiterer Bilder mit ähnlicher spektraler Signatur verwendet.

Tan et al. [26] extrahieren Regionen aus natürlichen Bildern mit einem globalen Schwellwert für verschiedene photometrische Merkmale. Die Benutzerinteraktion beschränkt sich auf das Markieren der interessanten Objekte in Form von Einkreisen oder Anklicken. Die globale Segmentierung wird mit einer linearen Diskriminanzanalyse lokal verfeinert.

Wooley und Smith [13] verwenden sechs Texturmerkmale als Randbedingung für ihr Region Growing. Weichen Mittelwert und Standardabweichung der Merkmale von Benutzervorgaben ab, liegt eine neue Region vor. Die entstehenden Regionen werden mit einem unüberwachten Bayes'schen Clustering kategorisiert.

Die beschriebenen CRG–Algorithmen sind mit nur wenigen Randbedingungen aufgebaut, welche zudem subjektiv für die Problemstellung ausgewählt werden. Die Kontrollmöglichkeiten für den Benutzer sind stark spezialisiert und es bleibt unklar, ob sie ausreichend und für weitergehende Problemstellungen geeignet sind.

Es existieren weitere Segmentierungsverfahren, in denen Merkmale und sonstige Randbedingungen zu finden sind. Metzler et al. [27] segmentieren wissensbasiert: Aus einer Datenbasis von Regionenbildern werden mit evolutionären Algorithmen neue Regionenbilder erzeugt. Dazu fassen sie die Segmentierung als hochdimensionales Optimierungsproblem auf und kombinieren ein globales und ein lokales Optimierungsverfahren zu einem iterativen Schema, das verschiedene Bildeigenschaften als Randbedingungen für die Segmentierungen verwendet.

Iivarinen et al. [28] verwenden lokale Texturmerkmale zusammen mit self-organising maps (SOM) zur Schätzung der Eigenschaften von intaktem Papier. Segmentiert werden Regionen, deren lokale Eigenschaften stark von den geschätzten abweichen. Eine Kombination von Form-, Farb- und Texturmerkmalen hilft bei der Erkennung dieser Regionen als tatsächliche Defekte. Sie schlagen zudem vor, permanent die Eigenschaften der Defekte zu überwachen und gegebenenfalls anzupassen, da nicht alle Defekte a priori bekannt sind.

Konturorientierte Verfahren (Snakes, ...) orientieren sich an der Dissimilarität an den Grenzen zwischen zwei Regionen. In der ursprünglichen Form wird hier

von starken Änderungen des Gradienten ausgegangen, die als Kante einer Region interpretiert wird. Im Gegensatz dazu beschränken andere Verfahren die Segmentierungsmöglichkeiten durch statistische Modelle der äußeren Form, die von Trainingssets abgeleitet sind [4, 29, 30]. Objekte, von denen die Grundgesamtheit ihrer Formen nicht geschätzt werden kann oder solche, deren Form nicht ausschlaggebend ist, können auf diese Weise nicht segmentiert werden [4].
Desweiteren wird vorgeschlagen, Segmentierungs- und Klassifikationstechniken mit Lernmethoden zu vereinen, so daß Prozesse an bestimmte Aufgabenstellungen angepaßt werden können. Lee et al. [4] lassen ihr System unüberwacht lernen, um so für eine bestimmte Aufgabe klassenspezifische Templates für die Segmentierung mit Snakes zu lernen. Dabei werden die Templates durch Mittelung über jede Klasse von Konturen generiert.
Die Kombination eines Segmentierungsalgorithmus mit einem genetischen Algorithmus, der die Parameter zur Segmentierung von Synthetic Aperture Radar (SAR) Bildern optimiert, wird von Bhanu et al. [31] vorgeschlagen. In einem anderen Beitrag [7] wird ein allgemeiner Ansatz zur Mustererkennung präsentiert, welcher die Segmentierungsparameter an sich verändernde Bildaufnahmebedingungen anpassen kann. Die Anwendung wird allerdings auf Objekte, deren Modell bekannt ist, beschränkt.
Legal-Ayala et al. [32] schlagen eine lernende Segmentierung vor, die aus ideal segmentierten Bildern lokale Merkmale berechnet und daraus eine Entscheidungsmatrix für die Pixel konstruiert. Für jedes Pixel eines unbekannten Bildes wird mit einer abgewandelten Nearest–Neighbour–Strategie die beste Entscheidung in der Matrix gesucht. Die Methodik verwendet allerdings nur vier speziell ausgesuchte Merkmale und ist auf die Segmentierung von Handschrift in Dokumenten beschränkt. Der Hintergrund dieser Dokumente ist zudem extrem homogen. Außerdem wird nur mit sehr wenigen Beispielen gelernt. Aufgrund dieser Beschränkungen und der Notwendigkeit idealer Segmentierungen, die oft nicht in ausreichender Zahl verfügbar sind, scheint die Methodik nicht gut für andere Segmentierungsprobleme geeignet zu sein.
Oft wird darauf hingewiesen, daß aufgrund der zu geringen Anzahl an Trainingsbildern für eine verläßliche Parameterschätzung inkrementell gelernt werden sollte, sobald neue Informationen (Bilder) zur Verfügung stehen [4, 7].

2.3 Fazit

Die in 2.1 gestellten Anforderungen werden von keinem der diskutierten Ansätze vollständig erfüllt.
Die verfügbaren Systeme, die komplette Bildverarbeitungspfade modellieren, sind in der Regel zu einfach angesetzt. Es wird vermutet, daß sie nur in speziellen Anwendungsfällen funktionieren. Ergebnisse und Implementierungen konnten in der Literatur nicht gefunden werden.
Einfache Segmentierungsverfahren sind für komplexe Bildinhalte unzureichend. Mit zusätzlichen Parametern lassen sich komplexere Bildinhalte besser modellieren. Die in der Literatur beschriebenen Algorithmen verwenden jedoch alle subjektiv oder empirisch ausgewählte Parameter und sind auf spezielle Probleme optimiert. Es ist kein Algorithmus bekannt, der für ein beliebiges Segmentierungsproblem die benötigten Parameter findet und optimiert.
Die meisten existierenden Methoden sind somit durch die meist ungenügenden Modellannahmen oder die unzureichenden Möglichkeiten, sich an neue Daten anzupassen, limitiert.
Es gibt zurzeit in der Literatur kein System, das einen Experten bei der Lösung eines unbekannten Mustererkennungsproblems ausreichend unterstützt. Die Möglichkeit zum interaktiven Training und der daraus resultierenden Adaptionsfähigkeit des Systems ist bisher nicht ausreichend gegeben. Der Experte kann nicht korrigierend eingreifen, wenn die Ergebnisse nicht seinen Wünschen entsprechen.

2.4 Ziele

Das Ziel dieser Arbeit ist es, dem menschlichen Experten ein System zur Verfügung zu stellen, welches er mit wenig Aufwand auf die selbsttätige und zuverlässige Erkennung von Objekten auf digitalen Bildern trainieren kann. Dazu soll ein adaptives System entwickelt und implementiert werden, das die Segmentierung der Objekte automatisch und reproduzierbar durchführt. Geeignete Parameter sollen automatisch ausgewählt werden, so daß die Segmentierungsergebnisse den Anforderungen des Experten gerecht werden.
Das System soll für möglichst viele verschiedene Mustererkennungsprobleme ge-

eignet sein. Daher ist ein allgemeiner Ansatz notwendig, der innerhalb eines gemeinsamen Frameworks die benötigten Algorithmen zur Segmentierung, Merkmalsextraktion, -selektion und Klassifikation automatisch auswählen und auf das jeweilige Problem adaptieren kann.
Das System soll einem menschlichen Experten die Auswahl der Systemparameter abnehmen und die Auswahl selbsttätig in Abhängigkeit von der jeweiligen Anwendung treffen. Durch die Eliminierung der subjektiven Parameterwahl sollen die Ergebnisse objektiv und reproduzierbar werden. Insbesondere für viele Segmentierungsprobleme sollen so geeignete Parameter zur Beschreibung komplexer Bildinhalte gefunden werden.
Der Experte soll bei der Automatisierung eines Mustererkennungsprozesses unterstützt werden. Dazu gehört die Erstellung einer geeigneten Stichprobe, die die Verläßlichkeit der Parameterwahl gewährleisten soll. Das System soll in der Lage sein, sich an Veränderungen der Daten anzupassen und diese zu lernen. Dieses Lernen soll interaktiv und auf einfache Art und Weise durch das Präsentieren von Beispielen im Interaktion mit dem Experten geschehen.
Der Experte soll in das System korrigierend eingreifen und die Ergebnisse seinen Wünschen anpassen können.

2.5 Zusammenfassung

In diesem Kapitel wurden die Anforderungen an ein System zusammengetragen, das automatisch reproduzierbare Segmentierungsergebnisse zum Zwecke der Mustererkennung erzielen soll. Nach dem Stand der Technik existiert kein System, welches die Anforderungen an ein solches System (vgl. 2.1) erfüllt. Daraus resultieren folgende Ziele: die Entwicklung eines allgemeinen Ansatzes für ein automatisches System zur Segmentierung und Erkennung von Mustern auf Bildern, die automatische Wahl geeigneter Parameter, Lernfähigkeit und Interaktivität.

Kapitel 3

Grundlagen

In diesem Kapitel werden die wichtigsten Konzepte zur computergestützten Erkennung von Mustern kurz eingeführt. Dabei wird die Reihenfolge der Verarbeitungsschritte in der Bildverarbeitungskette verwendet (vgl. 1.1).

3.1 Mustererkennung

Mustererkennung ist die computergestützte Detektion und Analyse von Mustern in Daten. In der Bildverarbeitung versteht man darunter die Analyse, Beschreibung, Identifikation und Klassifikation von Objekten [33]. Im Allgemeinen besteht ein System zur Mustererkennung aus mehreren Stufen, wie in Abb. 1.1 dargestellt.
Man unterscheidet [6, 33, 34] Template Matching, statistische Klassifikation, syntaktisches (strukturelles) Matching und Neuronale Netze.
Beim Template Matching werden zu erkennende Muster mit bekannten Templates des selben Typs verglichen und der Klasse desjenigen Templates zugewiesen, mit dem der höchste Grad des Vergleichsmaßes, z.B. Korrelation, erzielt werden kann.
Bei der statistischen Klassifikation wird jedes Muster als Punkt in einem hochdimensionalen Merkmalsraum betrachtet. Hierbei liegen die Ziele in der geschickten Auswahl derjenigen Merkmale, die es den Mustern verschiedener Kategorien gestatten, kompakte und disjunkte Regionen im Merkmalsraum einzunehmen [6] und in der Bestimmung der Grenzen, die diese Regionen voneinander trennen.

Das syntaktische Matching folgt einem hierarchischen Ansatz und betrachtet komplexe Muster als Komposita einfacherer Muster, sog. Primitive. Dieses Konzept formt Analogien zwischen der Struktur eines Musters und der Syntax einer Sprache. Somit können große Mengen komplexer Muster von einer kleinen Anzahl von Primitiven und grammatikalischen Regeln dargestellt werden [6, 33].
Mit künstlichen neuronalen Netzen, bei denen einzelne Neuronen netzartig verknüpft sind, wird versucht, die Lernfähigkeit des menschlichen Gehirns nachzubilden und Eigenschaften wie sequentielle Trainierbarkeit und Adaptivität zu erreichen [35]. Durch wiederholtes Anlegen von Beispielmustern und durch Anwendung eines Lernalgorithmus speichert das Netz Informationen. Dadurch können komplexe Muster gelernt werden, ohne die zugrundeliegenden Regeln zu abstrahieren. Als nachteilig erweist sich, daß die Logik nicht mehr aus dem Netz ermittelt werden kann [33]. Das Training als Lösung eines hochdimensionalen nichtlinearen Optimierungsproblems und der Hang zum Auswendiglernen (Overfitting) bei schlechter Wahl der Netzarchitektur sind weitere schwerwiegende Nachteile. Neuronale Netze sollten aufgrund ihrer Komplexität erst dann zum Einsatz kommen, wenn einfachere Verfahren versagen [35].

3.1.1 Segmentierung

Das Ziel einer Segmentierung ist es, ein Bild so zu partitionieren, daß jede Region ein semantisches Objekt enthält. Dabei wird auch der Bildhintergrund als Objekt betrachtet. Die Modellannahme geht davon aus, daß eine Homogenität oder Ähnlichkeit innerhalb des Objektes und eine Diskontinuität zwischen den Objekten besteht [36].
Eine Region ist eine durch die Segmentierung entstandene, zusammenhängende Ansammlung von Pixeln. Sie ist nicht zwingend semantisch korrekt. Ein Objekt hingegen ist zwingend semantisch korrekt.
Eine Segmentierung ist normalerweise dann optimal, wenn sie nicht von einer bei synthetischen Bildern bekannten oder einer benutzerspezifischen Segmentierung abweicht.
Die vielen bekannten Segmentierungsverfahren lassen sich in pixel-, kanten- und regionenbasierte Verfahren einteilen [22, 33]. Zusätzlich unterscheidet man mo-

dell- und texturbasierte Verfahren. Die Grenzen zwischen den Verfahren sind oft fließend [33].

3.1.1.1 Pixelbasierte Verfahren

Bei pixelbasierten Verfahren lautet die Modellannahme, daß für jeden einzelnen Bildpunkt die Entscheidung über seine Objektzugehörigkeit unabhängig von seinen Nachbarn getroffen wird. Das einfachste Verfahren ist das globale Schwellwertverfahren, bei dem über den Vergleich des Grauwertes (oder eines anderen eindimensionalen Merkmals, z.B. der spektralen Signatur) mit einem Schwellwert entschieden wird, ob das Pixel zum Objekt oder zum Hintergrund gehört [21, 22, 23, 33].
Problematisch ist die automatische Wahl des Schwellwertes, insbesondere für Bilder, bei denen das Histogramm des verwendeten Merkmals multimodal ist. Adaptive Schwellwertverfahren können hier helfen. Nachteilig ist die Vernachlässigung globaler Zusammenhänge [37].

3.1.1.2 Kantenbasierte Verfahren

Das Modell bei diesen Verfahren nimmt an, daß Objekte durch Kanten voneinander getrennt sind. Diese Kanten entsprechen Diskontinuitäten der Homogenität. Die Detektion dieser Diskontinuitäten ist das Ziel der kantenbasierten Verfahren [37].
Da viele Algorithmen keine geschlossenen Kantenzüge liefern, müssen diese zum Objekteinschluß mit weiteren Verfahren zusammengefügt werden [22]. Beispiele für kantenbasierte Verfahren sind Sobel- und Laplace-Operator sowie die Gradientensuche. Bei diesen können unvollständige Kanten entstehen. Zusammenhängende Kanten werden z.B. mit der Wasserscheidentransformation, Live-Wires oder Aktiven Konturen (Active Shape Models oder Snakes) erzeugt [22, 33]. Die am häufigsten verwendete Aktive Kontur sind Snakes. Dabei wird a priori Wissen durch Definition und Parametrisierung eines Energieterms sowie durch Vorgabe einer Kontur in die Bilddaten eingebracht. Der Energieterm setzt sich aus innerer und äußerer Energie zusammen. Dabei stellt die Krümmungsenergie der Kurve die innere und die Bildenergie (Gradient) die äußere Energie dar [38]. Die Kontur

wird solange iterativ modifiziert, bis ihre Energie minimal ist. Bei den Live-Wires wird ebenfalls a priori Wissen in die Segmentierung eingebracht: Zwischen gegebenen Stützpunkten im Bild wird ein kostenoptimaler Pfad gesucht, wobei die Kostenfunktion auf dem lokalen Gradienten basiert [39].

3.1.1.3 Regionenbasierte Verfahren

Regionenorientierte Verfahren modellieren die Objektzugehörigkeit eines Punktes in Abhängigkeit von seiner Umgebung [22]. Dazu wird ein Distanzmaß benötigt, mit dem die Entscheidung der Zugehörigkeit getroffen werden kann. Ein Distanzmaß ist ein Maß für die Ähnlichkeit von zwei Regionen, die im einfachsten Fall nur einen Punkt enthalten. Ein typisches Distanzmaß ist der Grauwertabstand zweier Pixel [37].

$$d(f_1, f_2) = |g_1 - g_2| \tag{3.1}$$

f_1 und f_2 werden zu einer Region verschmolzen, wenn $d \leq \vartheta$, wobei ϑ ein frei wählbarer Schwellwert ist.
Neben divisiven (Split and Merge [37]) und hierarchischen (Pyramid Linking [22]) Verfahren ist das Region Growing (engl. (Seeded) Region Growing) der häufigste Vertreter [20]. Ausgehend von einem Saatpunkt (seed) werden zwei benachbarte Pixel zusammengefaßt, wenn ihr Gradient unter einer bestimmten Schwelle liegt [21, 22, 23]. Die entstehenden Regionen sind dann homogen im Sinne des Gradienten.
Der einfache Gradient ist in den meisten Fällen nicht ausreichend, die Homogenität von Regionen zu beschreiben. Das Constrained Region Growing (CRG) [12, 20, 22] – also das Regionenwachstum unter Randbedingungen – verwendet komplexere Distanzfunktionen, die je nach Anwendungsfall modelliert werden müssen.

3.1.1.4 Modellbasierte Verfahren

Bei diesen Verfahren wird Wissen über ein Bild eingesetzt und ein Modell der gesuchten Objekte zugrundegelegt, beispielsweise die Form. Weiterhin werden statistische Modelle und die Segmentierung über Templates (Template-Matching) verwendet [22, 33].

3.1.1.5 Texturorientierte Verfahren

Die meisten natürlichen Strukturen besitzen keine einheitliche Farbe, sondern eher eine einheitliche Textur. Die bisherigen Ansätze sind zur Segmentierung solcher Strukturen unzureichend. Es wird versucht, die Segmentierungsparameter dahingehend einzustellen, daß das gewünschte Texturmuster segmentiert wird. Beispiele für solche Parameter sind Cooccurrence-Matrizen von Haralick [40], Lauflängenmatrizen (Run-Length-Matrix) [41], Texturenergiemaße (Texture Energy Measure) [33] und fraktale Dimensionen [33].

3.1.1.6 Fazit

Die verschiedenen Segmentierungskonzepte sind nicht universell einsetzbar, da sie unterschiedliche Eigenschaften der zu segmentierenden Objekte modellieren. In zahlreichen Publikationen werden daher Erweiterungen, Verbesserungen und Kombinationen der bestehenden Segmentierungskonzepte vorgeschlagen. Diese werden in Kapitel 2 diskutiert.

3.1.2 Merkmale

Als Merkmal (auch Attribut genannt) werden gemeinhin die Eigenschaften von Objekten, beispielsweise Farbe, Form und Struktur bezeichnet [6]. Der menschliche Sprachgebrauch kennt zudem subjektive und schwer quantifizierbare Eigenschaften wie z.B. warm, kalt, rauh, glatt, hell, dunkel, ... Als Merkmal bezeichnet man daher eine deterministische Rechenvorschrift für die Messung der Eigenschaften eines (segmentierten) Objekts [42].
Es gibt viele verschiedene Merkmale (z.B. statistisch, morphologisch und texturiell) aus unterschiedlichen Kategorien (z.B. numerisch oder nominal). Ein gemessenes Merkmal wird als Observation (Beobachtung) bezeichnet.

3.1.3 Merkmalsextraktion

Die Berechnung eines Merkmales wird als Merkmalsextraktion bezeichnet [43]. In der Literatur wird als Merkmalsextraktion auch die Ableitung neuer Merkmale aus existierenden Merkmalen [6] bezeichnet. Diese Arbeit verwendet die erstere

Definition; für letzteren Sachverhalt wird vielmehr der Begriff der Merkmalstransformation verwendet. Die gemessenen Merkmale eines Objektes bezeichnet man als Muster oder Observation.

3.1.4 Merkmalsselektion

Es existieren Merkmale oder Kombinationen von Merkmalen, die besser geeignet sind, Objekte zu beschreiben als andere [35, 44]. Die Auswahl einer solchen Kombination bezeichnet man als Merkmalsselektion [35, 45]. Bekannte Verfahren zur Merkmalsselektion sind die Varianz- und die Diskriminanzalanlyse [33]. Weitere gebräuchliche Verfahren wie die Forward Selection und die Backward Elimination sind in [35, 45] beschrieben.

3.1.5 Klassifikation

Als Klassifikation bezeichnet man die eindeutige Zuordnung einer Observation zu einer Gruppe mit gemeinsamen Eigenschaften [46]. Diese Gruppe wird als Klasse bezeichnet. Ein Verfahren, das diese Zuordnung durchführt, heißt Klassifikationsverfahren. Ein Klassifikationsverfahren wird mit repräsentativen Beispielen trainiert. Ein solches Beispiel besteht aus einer Beobachtung und der dazugehörigen Klasse. Die Menge von Beispielen ist eine Teilmenge aller möglichen Ausprägungen und wird als Stichprobe bezeichnet. Je besser und größer die Stichprobe ist, desto erfolgversprechender ist das Training. Je repräsentativer die Stichprobe ist, desto generischer ist die Klassifikation unbekannter Beobachtungen [46]. Prinzipiell lassen sich Klassifikationsverfahren in zwei Klassen einteilen [6]:

- **Überwachte Klassifikation**: Die Klassenzugehörigkeit der als Lernbeispiele dienenden Observationen ist bekannt. Das Ziel besteht in der Zuordnung unbekannter Observationen zu den bekannten Klassen mit einem möglichst geringen Klassifikationsfehler. Die Zuordnungsvorschrift wird während des Trainings aus den Mustern bestimmt. Überwachtes Lernen wird auch als Diskriminanzanalyse bezeichnet [47].

- **Unüberwachte Klassifikation**: Ohne Vorwissen über die Klassenzugehörigkeit der Observationen ist es das Ziel, Anhäufungen der Daten im Merk-

malsraum zu identifizieren und die Observationen Clustern zuzuordnen. Dieses Vorgehen wird daher auch als Clustering [47] bezeichnet. Anschließend können weitere Beobachtungen den festgelegten Clustern zugeordnet werden.

Der zentrale Algorithmus, der die Zuordnung von Observationen zu Klassen vornimmt, ist der Klassifikator. Abhängig von den Merkmalen, der Anzahl der Observationen und dem Klassifikationsalgorithmus, wird nicht immer die korrekte Klasse vorhergesagt. Dies wird Klassifikationsfehler genannt. Das Verhalten des Klassifikators auf unbekannten Mustern wird durch den Generalisierungsfehler ausgedrückt [35, 45].

3.1.6 Fazit zur Mustererkennungskette

Die beschriebene Vorgehensweise zur Mustererkennung in der digitalen Bildverarbeitung folgt einer starren Abfolge von Algorithmen zur Segmentierung des Musters, der Extraktion von dessen Merkmalen, der Auswahl guter Merkmale und der Zuweisung des Musters in eine a priori bekannte Klasse.
Damit diese Vorgehensweise für ein beliebiges Mustererkennungsproblem funktioniert, müssen Wissen und Erfahrung eines Experten über das Erkennungsproblem entsprechend in Algorithmen umgesetzt werden. Da dies sehr selten möglich ist (vgl. 1.1), benötigt man ein System, in dem eine geeignete Lernstrategie zur Anwendung kommt.
Es existieren verschiedene Methoden zum maschinellen Lernen. Maschinelles Lernen ist die künstliche Generierung von Wissen aus Erfahrung [33, 35]. Die Einteilung der Methoden erinnert stark an die der Klassifikation (vgl. 3.1.5).

- Überwachtes Lernen (engl. supervised learning): Das System lernt dabei aus gegebenen Paaren von Aufgaben und Lösungen. Ein externer Lehrer gibt dabei die jeweils korrekte Lösung vor. Die maschinelle Klassifikation ist ein Teilgebiet des überwachten Lernens.

- Unüberwachtes Lernen (engl. unsupervised learning): Für eine gegebene Menge von Aufgaben wird ein Modell erzeugt, das die Aufgaben miteinander vergleicht und in Gruppen einteilt. Dabei kann die Anzahl der Gruppen

a priori bekannt sein, die Art der Gruppe (Klasse) hingegen ist unbekannt.

- Bestärkendes Lernen (engl. reinforcement learning): Durch externes Feedback in Form von Belohnung und Bestrafung lernt das System, wie es sich in bestimmten Situationen zu verhalten hat.

Das zu entwerfende System soll einen Experten unterstützen, der notfalls korrigieren eingreifen können soll. Daher kommt am ehesten die Form des bestärkenden Lernens in Frage.

3.2 Softwareumgebung

Diese Arbeit ist in die Komponentensoftware *ICE - Integrated Component Environment* [46] integriert. Darin eingebettet ist *WEKA - Waikato Environment for Knowledge Analysis* [45]. Im folgenden werden diese beiden Programme näher erläutert.

3.2.1 ICE

ICE ist eine Komponentensoftware, vergleichbar mit Khoros [48] oder LabView [49]. Ihr Aufbau beruht auf dem Prinzip eines Baukastens. Dadurch wird die einfache Erweiterung mit neuen Komponenten sowie die Interaktion bestehender und neuer Komponenten erleichtert. ICE ist in JAVA implementiert und daher plattformunabhängig und streng objektorientiert. Ursprünglich wurde ICE für die automatische Erstellung von Klassifikationssystemen entwickelt [50]: Mit verschiedenen einschränkenden Annahmen und geschicktem Resampling der Trainingsdaten können mit der in ICE implementierten sog. *Blasensynthese* in kurzer Zeit optimale Kombinationen aus *Selektions- und Klassifikationsalgorithmen* gefunden werden [46]. Für gängige Klassifikationsprobleme wurden mit ICE bessere Kombinationen gefunden als zuvor in der Literatur beschrieben. Daher soll die Blasensynthese in das zu entwerfende System miteinfließen. Für große Datenmengen und eine große Anzahl verfügbarer Selektions- und Klassifikationsverfahren steigt die Rechenzeit stark an und limitiert so die Anwendbarkeit in einem interaktiven System.

ICE ist nicht auf die Klassifikation beschränkt, so wie beispielsweise Khoros auf die Bildverarbeitung. Für eine Vielzahl von Problemstellungen können Komponenten entwickelt werden. Bereits implementiert sind bisher Komponenten für die Bildverarbeitung und Klassifikation sowie Datenimport und -export.
Alle Komponenten werden von einer Wurzel-Komponente abgeleitet, in welcher sich alle für die Interaktion in ICE notwendigen Schnittstellen befinden. Komponenten können über Datenflußleitungen Daten austauschen und interagieren. Die Syntax und Semantik der Schnittstellen wird durch abstrakte globale Datentypen definiert; dies gewährleistet eine vollständige Transparenz der Komponenten. Die Entwicklung neuer Komponenten und Datentypen wird durch verschiedene Hilfsmittel zur Generierung von Quelltext erleichtert [46].
Container-Komponenten enthalten andere Komponenten und organisieren Abläufe hierarchisch in Form von Baumstrukturen. Diese Container liegen auf einer Art Metaebene, durch die ICE in der Lage ist, selbsttätig Komponenten zu instanziieren und so die Programmierung komplexer Abläufe zu vereinfachen.
ICE stellt eine graphische Benutzeroberfläche zur Verfügung (vgl. Abb.3.1), welche die Funktionalität und das Zusammenspiel sowohl bestehender, als auch neuer Komponenten erheblich erleichtert. Jede Komponente muss auf einem Desktop abgelegt werden. Es entsteht ein graphisches Symbol, welches den aktuellen Status der Komponente repräsentiert. Für jede Komponente existiert ein sogenanntes „Property Sheet", in dem die notwendigen Parameter der Komponente eingestellt werden. Eine ausführlichere Beschreibung zum Aufbau von ICE, seiner Funktionalität und Bedienung findet sich in [46].

3.2.2 WEKA

WEKA wurde speziell für das maschinelle Lernen entwickelt. Es besteht aus einer großen Sammlung von Klassifikatoren und Merkmalsselektoren. Zusätzlich stehen Import und Export sowie einige Visualisierungsmöglichkeiten zur Verfügung. WEKA ist ebenfalls in JAVA implementiert, konnte daher leicht in ICE integriert werden und dient als Algorithmensammlung für das automatische Auffinden optimaler Selektor-Klassifikatorkombinationen mit ICE.
WEKA verwendet für die Observationen ein spezielles Dateiformat, die sog. .arff-

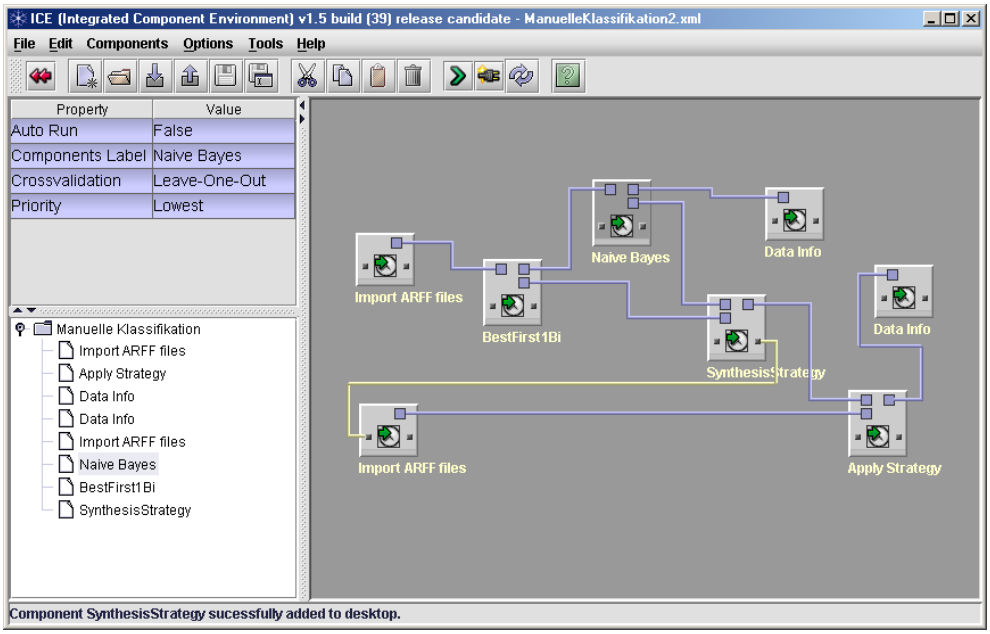


Abbildung 3.1: *ICE. Beispielhafter Aufbau eines Klassifikationssystems in ICE mit verschiedenen Komponenten. Ein Trainingsdatensatz wird mit Import ARFF (vgl. Tab. 3.1) eingelesen. Der Merkmalsselektor BestFirst1Bi wählt Merkmale aus. Danach wird ein NaiveBayes- Klassifikator aufgebaut. Aus dem Selektor und dem Klassifikator wird eine sog. Strategie synthetisiert, die auf unbekannte Daten angewendet wird. DataInfo dient hier der Visualisierung der Klassifikationsfehler. Die blauen Linien stellen Datenpfade dar, die gelbe Linie eine Synchronisation, so daß die untere Import-Komponente erst dann automatisch gestartet wird, wenn die Strategie synthetisiert wurde.*

Dateien. Tabelle 3.1 zeigt den prinzipiellen Aufbau einer solchen Datei.

Nach dem Einlesen einer .arff-Datei können Selektions- und Klassifikationsalgorithmen auf die Daten angewendet werden. Erst durch die Integration in ICE wurde es möglich, die erstellten Selektoren und Klassifikatoren als Binärdaten abzuspeichern.
Neuerdings stellt WEKA eine graphische Oberfläche mit komponenten-ähnlichem Verhalten zur Verfügung, die jedoch bei weitem nicht so intuitiv und komfortabel zu bedienen ist wie ICE. Eine ausführliche Beschreibung zu WEKA findet sich in [51, 45].

% Kommentar		
@relation	Name des Datensatzes	
@attribute	Merkmal 1	Typ{numerisch,nominal}
@attribute	Merkmal 2	Typ{numerisch,nominal}
...		
@attribute	Merkmal n	Typ{numerisch,nominal}
@attribute	Klasse	{Klasse 1, Klasse 2, ...}
@data		
x_{11},x_{21},. . . ,x_{n1},Klasse		
x_{12},x_{22},. . . ,x_{n2},Klasse		
...		
x_{1m},x_{2m},. . . ,x_{nm},Klasse		

Tabelle 3.1: *Prinzipieller Aufbau einer .arff-Datei.*

3.3 Zusammenfassung

In diesem Kapitel wurden die grundlegenden Begriffe der Mustererkennung und der dabei verwendeten Algorithmen erläutert. Dabei wurde auf Segmentierung, Merkmale, Merkmalsextraktion und -selektion sowie die Klassifikation eingegangen. Die Schlußfolgerung legt nahe, daß starre Systeme nur begrenzt einsatzfähig sind. Daher wurde kurz beschrieben, welche Lernmethoden existieren, mit denen starre Systeme flexibilisiert werden können. Weiterhin wurden die verwendeten Softwarepakete ICE und WEKA kurz vorgestellt.

Kapitel 4

Adaptive Segmentierung

Um die geschilderte Problemstellung zu lösen, wird in diesem Kapitel ein System entworfen, das vom menschlichen Experten mit Hilfe von Beispielen auf viele beliebige Mustererkennungsprobleme trainiert werden kann. Es gelten folgende Einschränkungen:

- Ein Experte muß in der Lage sein, das gestellte Mustererkennungsproblem zu lösen, d.h. er kennt die Objekte und deren Klassenzugehörigkeit.

- Die verwendeten Daten bestehen aus $1-N$ Bildern in $1-M$ Dimensionen. Im weiteren werden nur zweidimensionale Grauwertbilder betrachtet.

4.1 Idee

Die Grundidee besteht darin, daß der Experte dem System Beispiele der zu erkennenden Objekte präsentiert. Aus diesen Beispielen leitet der Rechner automatisch geeignete Maße zur Erkennung ab. Als Maße dienen die lokalen Eigenschaften der Objekte. A priori sind die geeigneten Eigenschaften unbekannt, daher steht dem System eine Sammlung von Berechnungsvorschriften lokaler Merkmale zur Verfügung. Aus diesen wählt es die geeigneten Merkmale aus und segmentiert damit das Bild. Die Segmentierung wird vom Experten begutachtet und gegebenenfalls korrigiert. Auf diesem Weg wird das Wissen des Experten auf objektive Weise in eine für den Rechner verständliche Form umgesetzt.

4.2 Eigener Ansatz

Zusätzlich zu den o.g. Einschränkungen werden die folgenden Annahmen getroffen:

- Die Güte der Objekterkennung hängt von der Segmentierungsgenauigkeit ab. Die Genauigkeit der Segmentierung wiederum wird dadurch bestimmt, daß bekannt ist, um welche Objekte es sich handelt.
- Der Experte kann intuitiv geeignete Beispiele für die Objektklassen angeben.
- Es handelt sich immer um 2-Klassen-Probleme, d.h. ein Bild setzt sich nur aus 2 Objekttypen (im allgemeinen Objekt und Hintergrund) zusammen.
- Die Objekte lassen sich anhand charakteristischer lokaler Eigenschaften beschreiben.

Als lokale Eigenschaften werden Merkmale betrachtet, die aus einem einzelnen Pixel und/oder einer definierten Umgebung um ein einzelnes Pixel herum gemessen und diesem Pixel zugeordnet werden können (vgl. Abb. 4.1). Diese Umgebungen werden im folgenden als **gleitende Fenster** bezeichnet. Prinzipiell handelt es sich dabei um ein lineares Filter.

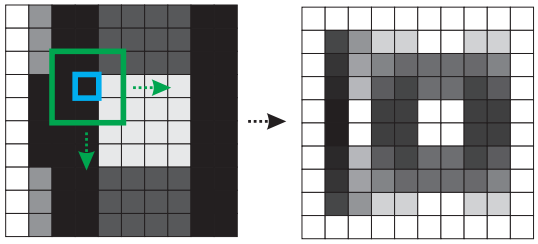


Abbildung 4.1: *Beispiel für die Extraktion eines lokalen Merkmals. Links: Aus einer $n \times n$ Pixel großen Umgebung (grün), im folgenden gleitendes Fenster genannt, wird für $n = 3$ die Streuung der Grauwerte (Formel (A.12)) bestimmt und dem Zentralpixel (blau) zugewiesen. Danach wird das Fenster verschoben. Dunkle Farben entsprechen niedrigen Grauwerten. Rechts: Das transformierte (gefilterte) Bild. Der äußere Rand hat die halbe Breite bzw. Höhe des gleitenden Fensters, da keine Werte von außerhalb des Bildes gemessen werden können. Dunkle Farben entsprechen höheren Streuungen.*

4.2.1 Methodik

Abbildung 4.2 zeigt den grundlegenden Aufbau des Systems.

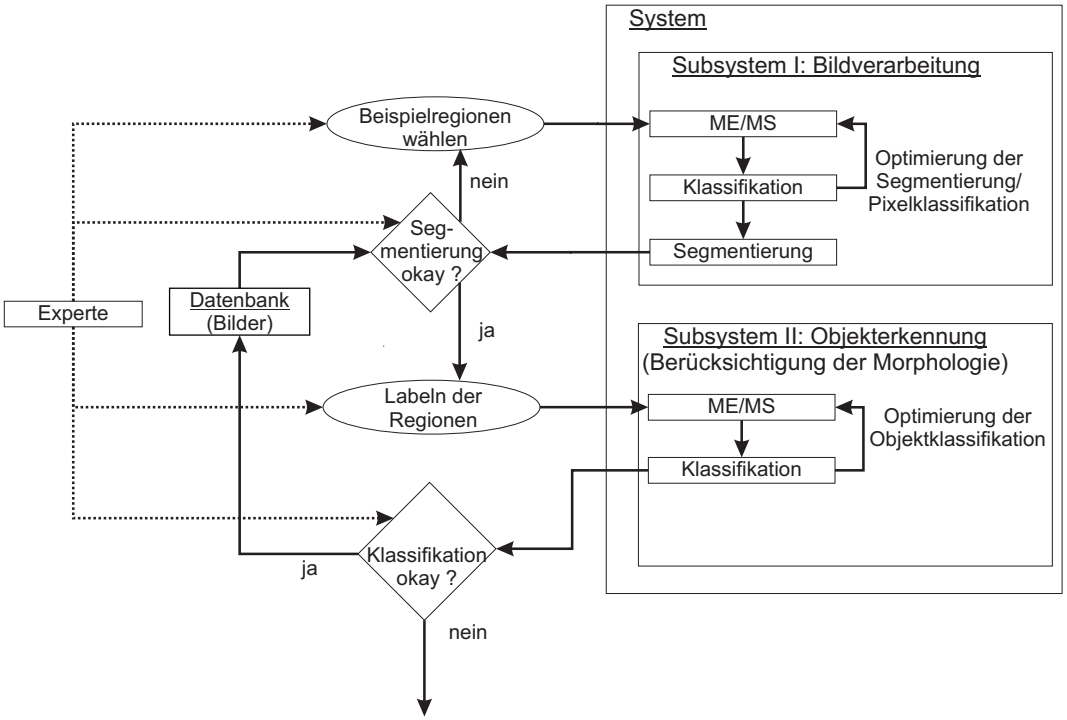


Abbildung 4.2: *Das zweiteilige Objekterkennungssystem. Subsystem I ist für die Segmentierung und Integration des Expertenwissens zuständig, während sich Subsystem II mit der eigentlichen Objekterkennung beschäftigt. Außerhalb des Systems stehen der Experte und seine Entscheidungen zur Segmentierungsgüte und Klassenzugehörigkeit der Objekte. Der Experte wählt Beispielregionen auf einem Bild aus einer Datenbasis. Das System berechnet die lokalen Eigenschaften der Beispielregionen, erstellt ein Klassifikationssubsystem für die Pixel des Bildes und schlägt auf dessen Basis eine Segmentierung vor. Der Experte korrigiert diese Segmentierung – wenn nötig – mit weiteren Beispielen, mit denen wie beschrieben verfahren wird. Außerdem weist er gut segmentierten Regionen eine Objektklasse zu, so daß das System ein Klassifikationssystem für die globalen Objekteigenschaften erstellen kann.*

Die **fett** gedruckten Worte beziehen sich im folgenden auf die Interaktionsmöglichkeiten des Experten: '**Beispielregionen wählen**', '**Segmentierung okay ?**', '**Labeln der Regionen**' und '**Klassifikation okay ?**'. Im folgenden wird nur das Subsystem I zur Bildverarbeitung detailliert betrachtet.
Prinzipiell steht eine Datenbank mit nicht vorsegmentierten und unklassifizierten Bildern einer Domäne zur Verfügung, d.h. die Bilder haben ähnliche Inhalte bzw.

es sind ähnliche Objekte abgebildet. Das gesamte Verfahren wird im folgenden Schritt für Schritt anhand eines Beispiels (s. Abb. 4.3) detailliert illustriert.

Abbildung 4.3: *Ein Zebra. Anhand dieses Bildes werden die einzelnen Schritte des Verfahrens erläutert.*

Aus der Datenbank wird das o.g. Bild ausgewählt. Dem System ist zu diesem Zeitpunkt keinerlei Information über das Problem bekannt. Das Bild ist nicht segmentiert, daher muß **Segmentierung okay ?** mit Nein beantwortet werden. Der Experte definiert eine oder mehrere Regions-Of-Interest (ROIs) für die Objektklasse (Zebra (weiß)) und die Hintergrundklasse (restlicher Bildinhalt (schwarz)) (s. Abb. 4.4).

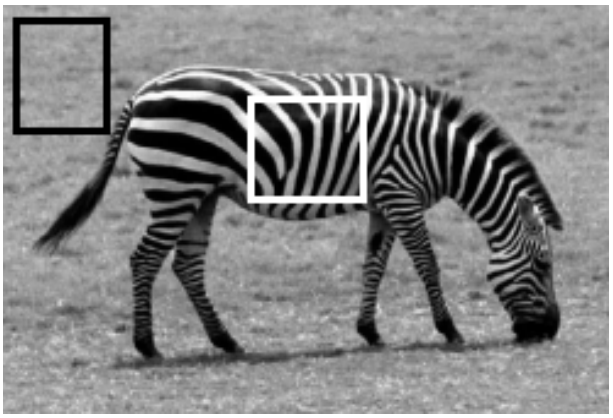


Abbildung 4.4: *Das Zebra mit ROIs für Objekt (Zebra (weiß)) und sonstigen Bildinhalt (schwarz).*

Aus diesen Beispielen soll das System geeignete Segmentierungsparameter bestimmen, um das Bild selbsttätig hinreichend gut zu segmentieren. Das System kennt weder die zur Trennung von Objekt und Hintergrund am besten geeigneten Merkmale, noch die Größe der gleitenden Fenster, aus denen die lokalen Merkmale extrahiert werden müssen.
Subsystem I zur Bildverarbeitung analysiert die Beispielregionen wie folgt: Ein gleitendes Fenster der Größe 3×3 Pixel wird über 100 äquidistante Punkte pro Klasse geschoben (s. Abb. 4.5) und für jeden dieser Punkte werden 45 Statistik- und Texturmerkmale extrahiert. Es entstehen 200 Observationen mit 45 Merkmalen und einer Klassenzugehörigkeit. Diese Observationen werden in einer .arff-Datei abgespeichert. Dieser Vorgang wird mit Fenstergrößen von 5×5, 7×7, 9×9, 11×11, 16×16, 32×32 und 64×64 wiederholt.

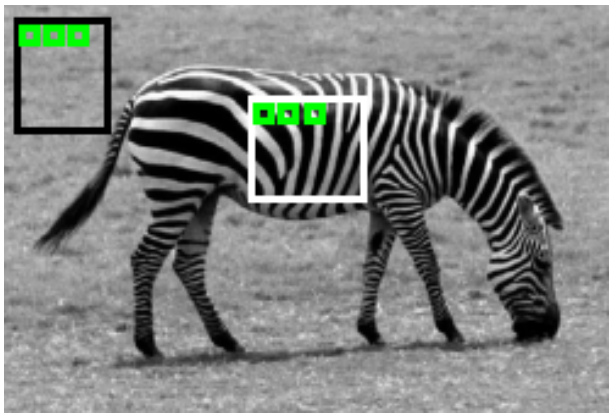


Abbildung 4.5: *Extraktion lokaler Merkmale aus den ROIs mit einem gleitenden Fenster (grün). Exemplarisch hat das Fenster die Größe* 5×5 *Pixel und es werden nur die jeweils ersten 3 Positionen des Fensters gezeigt.*

Für jede Fenstergröße wird ein Klassifikationssystem – im folgenden **Klassifikations*sub*system** genannt – aufgebaut: Aus einer Menge zur Verfügung stehender Merkmalsselektoren und Klassifikatoren wird mit der Blasensynthese automatisch diejenige Kombination ausgewählt (vgl. 3.2.1 (ICE)), die den geringsten Klassifikationsfehler aufweist. Gleichzeitig werden so die geeigneten Merkmale bestimmt. Für das Beispiel werden ein 11×11 Pixel großes Fenster, ein BestFirstForward-Selektor und ein Klassifikationsbaum (J48) bei einem Klassifikationsfehler von 0,3 % ausgewählt. Das ermittelte Merkmal ist die Kovarianz (TCov).
Mit dieser Fenstergröße und dem Klassifikationssubsystem wird die automatische

Segmentierung durchgeführt.
Zur Segmentierung wird das gesamte Bild betrachtet. Um jedes Pixel herum werden in der ermittelten Fenstergröße die vom Selektor bestimmten Merkmale extrahiert. Der erstellte Klassifikator bestimmt die Klasse des Pixels auf Basis der extrahierten lokalen Merkmale. Benachbarte Pixel, die der gleichen Klasse angehören, werden danach zu einer Region zusammengefaßt (s. Abb. 4.6).

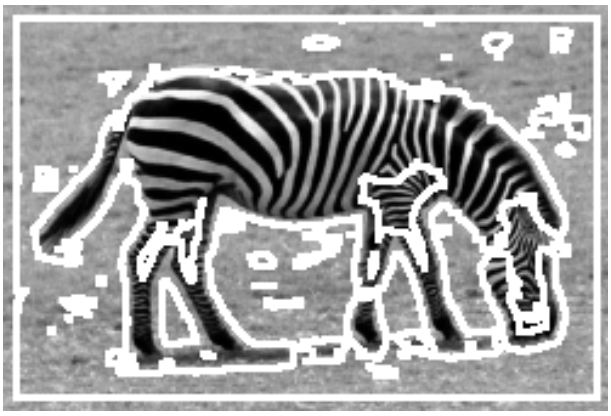


Abbildung 4.6: *Das segmentierte Zebra. Mit den gegebenen und analysierten Beispielen wurde das Zebra segmentiert. Das Bild ist deutlich übersegmentiert. Zudem werden Teile des Kopfes und der vorderen Oberschenkel nicht dem Tier zugerechnet. Ebenso werden Teile von Boden und Schatten dem Tier zugewiesen.*

Der Experte begutachtet die automatische Segmentierung und beantwortet die Frage **Segmentierung okay ?**:

- Nein: Das Zebra ist nicht zur Zufriedenheit des Experten segmentiert worden (vgl. Abb. 4.6). Der Experte wird aufgefordert, weitere ROIs anzugeben. Diese sollten sich vorzugsweise in den Regionen des Bildes befinden, die falsch segmentiert wurden (s. Abb. 4.7).
  Nach diesen Korrektureingriffen iteriert das System: Erneut werden aus den (alten und neuen) ROIs sämtliche verfügbaren Merkmale für alle Fenstergrößen extrahiert, neue Klassifikationssubsysteme aufgebaut, das beste ausgewählt und in die Segmentierung zurückgekoppelt. Das Merkmal ist jetzt der Variationskoeffizient (SVarCoeff), die Fenstergröße liegt weiterhin bei 11×11 und der Klassifikationsfehler bei 0,33 %.
  Die neue Segmentierung (s. Abb. 4.8) wird vom Experten begutachtet. In

diesem Fall ist das Zebra zufriedenstellend segmentiert.
Anderenfalls iteriert das System solange, bis der Experte mit der automatischen merkmalsgesteuerten Segmentierung zufrieden ist.

- Ja: Der Experte ist mit der Segmentierung zufrieden. Das Bild, die ROIs, die extrahierten Merkmale und das erstellte Klassifikationssubsystem, d.h. der Selektor und der Klassifikator, werden abgespeichert.

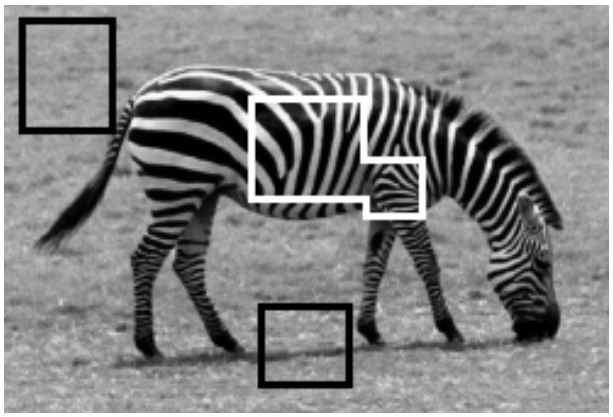


Abbildung 4.7: *Das Zebra mit ROIs für Objekt (Zebra (weiß)) und sonstigen Bildinhalt (schwarz). Zusätzlich zu den alten ROIs wurden in falsch segmentierten Bereichen neue ROIs definiert.*

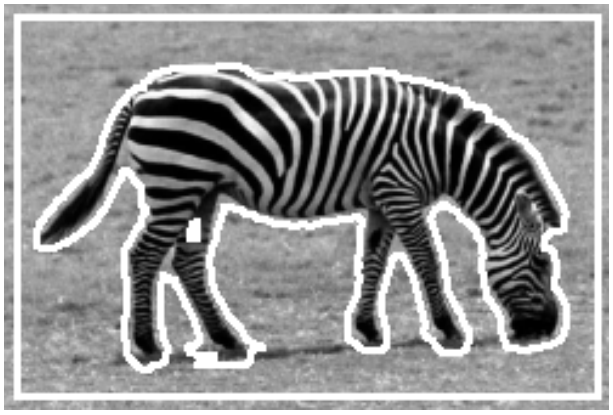


Abbildung 4.8: *Das segmentierte Zebra. Nach den zusätzlichen Beispielen wird das Zebra zur Zufriedenheit des Experten segmentiert.*

Die Bearbeitung des ersten Bildes ist damit abgeschlossen. Das erstellte Klassifikationssubsystem kann auf das zweite Bild angewendet werden (s. Abb. 4.9).

Diese Segmentierung wird ebenfalls vom Experten überprüft. Ist sie zufriedenstellend, war das bisher ermittelte Klassifikationssubsystem ausreichend und wird für das nächste Bild weiterverwendet. Ist die Segmentierung fehlerhaft und der Experte muß korrigierend eingreifen, so geschieht dies nach oben beschriebenem Muster durch die Definition vom ROIs für falsch segmentierte Bildausschnitte.

Abbildung 4.9: *Ein segmentiertes Zebra. Das bisher ermittelte Klassifikationssubsystem wird hier auf ein weiteres Bild angewendet, auf dem ein anderes Zebra abgebildet ist.*

Nach einem Korrektureingriff des Experten auf dem zweiten Bild werden aus definierten ROIs sämtliche Merkmale in der bereits festgelegten Fenstergröße berechnet. Diese extrahierten Merkmale werden mit denen des vorhergehenden Bildes (bzw. der vorhergehenden Bilder) kombiniert. Mit diesen Daten wird ein neues Klassifikationssubsystem aufgebaut, welches in der Lage sein soll, beide Objekte gut zu segmentieren. Der Segmentierung folgt eine erneute Überprüfung durch den Experten mit den geschilderten Möglichkeiten der Interaktion.
Nach dieser Methode wird für alle weiteren Bilder verfahren, die aus der Datenbank ausgewählt werden.

Sofern genug Bilder verarbeitet werden, zieht das System allgemeine Schlüsse über die zu erkennenden Objekte [6] und ist in der Lage, ohne Benutzerinteraktion mit dem bis dahin erreichten Klassifikationsfehler zu arbeiten [2, 3] sowie eine im Mittel gute Segmentierung aller Bilder zu gewährleisten [14].

4.2.2 Diskussion der Methodik

Die Methodik basiert auf der Annahme, daß jedes Objekt bzw. jede Region in sich in einer bestimmten Form homogen ist. Der Begriff der inneren Homogenität bezieht sich auf die Ähnlichkeit der Bestandteile einer Region [36], z.B. ähnliche Helligkeit, Kontrast, Textur etc. Die äußere Homogenität hingegen beschreibt die Ähnlichkeit zwischen zwei verschiedenen Regionen.
Prinzipiell ist es möglich, verbal zu beschreiben, weshalb zwei Regionen unterschiedlich sind. Prinzipiell ist es auch immer möglich, ein einziges Merkmal zu finden, in dem sich zwei Regionen unterscheiden. Dazu müssen im schlimmsten Fall unendlich viele Merkmale ausprobiert werden. Dies ist praktisch nicht durchführbar. Eine ausreichende mathematische Beschreibung der jeweiligen Homogenität wird durch eine Kombination verschiedener Merkmale erreicht.
Die Auswahl geeigneter Merkmale muß optimiert werden, da je nach Problemstellung andere komplexe Merkmalskombinationen benötigt werden. Die vorgestellte Methodik ermittelt eine solche geeignete Kombination von Merkmalen, mit denen die innere Homogenität der Regionen modelliert werden kann.

Um eine optimale Parameterwahl für jeden Anwendungsfall zu erreichen, steht dem System eine Sammlung von Merkmalen zur Verfügung. Zur Auswahl der geeigneten Parameter benötigt das System Vorwissen über die zu erkennenden Objekte. Dieses Vorwissen wird über die Interaktion mit dem Experten erlangt. Der Experte präsentiert Beispiele und Gegenbeispiele der zu erkennenden Objekte. Er muß die Objekte nicht aufwendig manuell segmentieren, sondern mit rechteckigen Regions-Of-Interest (ROIs) lediglich grob markieren. Hierbei ist er dafür verantwortlich, keine Hintergrundpixel als Vordergrund zu markieren und umgekehrt. Das System bestimmt automatisch geeignete Parameter, mit denen die Bilder so segmentiert werden können, daß der Experte mit dem Ergebnis zufrieden ist. Segmentiert das System falsch, muß der Experte korrigierend eingreifen, indem er zusätzliche Beispiele präsentiert (also ROIs definiert). Lokale Merkmale werden als Segmentierungsparameter verwendet.

Eine Segmentierung mit Kombinationen von Merkmalen bezeichnet man als Constrained Region Growing [12] (CRG). Jedes berechenbare Merkmal ist eine mögliche Randbedingung für das CRG. Merkmale, die geeignet sind, die innere Ho-

mogenität einer Region zu beschreiben, sollten als Segmentierungsparameter verwendet werden. Im Unterschied zur Literatur wird neben den Merkmalen noch ein echtes Klassifikationssubsystem eingesetzt. Dieses dient dazu, aus der verfügbaren Menge von Merkmalen automatisch eine geeignete Kombination zu wählen und auf Basis der vom Experten gegebenen Beispiele eine optimale Entscheidung über die Regionenzugehörigkeit eines Pixels abhängig von seinen lokalen Eigenschaften treffen zu können. Die Segmentierung wird somit als lokales Klassifikationsproblem betrachtet, bei dem die entsprechende Terminologie zum Einsatz kommt.

- Die Berechnung der lokalen Eigenschaften wird als *Merkmalsextraktion* betrachtet.
- Die Auswahl geeigneter Merkmale erfolgt durch die *Merkmalsselektion*.
- Die Trennung von Objekt- und Hintergrundpixeln erfolgt durch einen *Klassifikator*.

Mit diesen Begriffen läßt sich z.B. das herkömmliche Regionenwachstum wie folgt beschreiben: Die Berechnung des Gradienten wird als Extraktion des Merkmals "Grauwertgradient" aus einer eng begrenzten Region, d.h. dem Saatpixel und einem Nachbarpixel, aufgefaßt. Eine Merkmalsselektion findet angesichts der Tatsache, daß nur ein Merkmal verfügbar ist, nicht statt. Die Schwellwertentscheidung, zu welcher Region das Nachbarpixel gehört, entspricht einer einfachen Klassifikationsaufgabe.
Die Segmentierungsparameter bestehen somit aus den lokalen Eigenschaften und einem Klassifikationssubsystem aus einer geeigneten Selektor–Klassifikator–Kombination. Durch die vorgestellte Art der Segmentierung entstehen in erster Linie Regionen, die in den selektierten Merkmalen homogen sind. Ihre Pixel gehören potentiell zum gesuchten Objekt oder zum Hintergrund.
Nachdem ausreichend Beispiele präsentiert wurden, soll das System einen guten Parametersatz zur Segmentierung der Objekte bestimmt haben. Hierbei ist zu berücksichtigen, daß ein guter Parametersatz für alle Objekte nicht bedeutet, daß jedes Bild gut segmentiert wird. Ein guter Parametersatz garantiert vielmehr einen *average best fit* über alle Bilder [14], d.h. daß die Bilder im Mittel gut segmentiert werden.

Es kommt ständig zum Wechsel von automatischer Segmentierung und menschlicher Interaktion. Das System könnte dadurch in einen instabilen Zustand kommen. Dies würde bedeuten, daß jedes zusätzliche vom Experten gegebene Beispiel eine schlechtere Segmentierung nach sich zieht und in der letzten Iteration korrekt segmentierte Regionen im aktuellen Schritt falsch segmentiert werden. Die Folge wären Korrektureingriffe in bereits definierten ROIs.
Diese Inkonsistenz ist möglich, wird jedoch abgefangen: Aufgrund der begrenzten Bildgröße kann die Anzahl der vom Experten gegebenen Beispiele die Zahl der Pixel im Bild nicht übersteigen. Im schlimmsten Fall müßte also jedes Pixel explizit gelabelt werden. Dies entspricht einer manuellen Segmentierung. Das System würde das bestmögliche Klassifikationssubsystem bestimmen, welches die manuelle Segmentierung zu reproduzieren versucht. Der unvermeidbare Klassifikationsfehler dieses Systems begrenzt die Güte der automatischen Segmentierung.
In einem zweiten Schritt kann der Experte den Regionen Objektklassen zuweisen. Mit diesen kann das System die globalen Eigenschaften der entstandenen Regionen analysieren und einen entsprechenden Objektklassifikator aufbauen.

4.2.3 Implementierung

Die beschriebene Methodik wurde als Java–Beans in die Komponentensoftware ICE (vgl. 3.2.1) integriert. Damit folgt die Implementierung dem Konzept von ICE bezüglich der Wiederverwendbarkeit von Funktionalitäten. Bereits in ICE implementierte Selektoren und Klassifikatoren können mit sehr geringem Aufwand wiederverwendet werden. Implementiert wurden

- Komponenten
  - Adaptive Segmentierung
  - Container für gesamten Verarbeitungsvorgang
  - Merkmalsextraktion
  - Zusatzkomponenten
    - Erzeugen und Spalten von Labels
    - Aneinanderhängen von .arff-Dateien

    - Visualisieren von Bildern in Merkmalsräumen

- Notwendige Datenstrukturen: Label für Segmentierungen, ...
- Merkmalsextraktoren

4.2.3.1 Segmentierungskomponente

Die Segmentierungskomponente *Constrained Region Growing* erhält als Eingangsdaten das zu segmentierende Bild, die vorklassifizierten Observationen in Form einer .arff-Datei und einen Klassifikator. Weitere Parameter sind die Größe des gleitenden Fensters und seine Verschiebungsschrittweite in x, y, z. Außerdem sind verschiedene Segmentierungsarten verfügbar:

- Regionenwachstum: Hierbei wird ein normaler Wachstumsalgorithmus eingesetzt. Das Zentralpixel des Bildes wird als Saatpixel betrachtet.
- Kontursuche: Hierbei wird ausgehend vom Zentralpixel in Richtung des oberen Bildrandes nach einer Kontur, also einem Wechsel der Pixelklasse, gesucht. Danach wird dieser Klassenwechsel im Uhrzeigersinn verfolgt, bis die Kontur geschlossen ist.
- Windowed: Hierbei wird das gleitende Fenster zeilenweise über das Bild geschoben, das Zentralpixel klassifiziert und der Bereich unter dem Fenster der ermittelten Klasse zugewiesen. Mit dieser Methode können schnell grobe Übersichtssegmentierungen erzeugt werden.
- Saatpunktsuche und Regionenwachstum: Es wird ein Gitter mit der eingestellten Fenstergröße über das gesamte Bild gelegt und jedes Zentralelement klassifiziert. Ausgehend von diesen Saatpunkten wird dann ein Regionenwachstum eingeleitet.

Diese Segmentierungsarten können explizit vom Benutzer eingestellt werden. Das System verwendet standardmäßig den ersten Punkt, das 'Regionenwachstum'.
Aus der bereitgestellten .arff-Datei werden die enthaltenen Merkmale bestimmt und die jeweiligen Merkmalsextraktoren instanziiert.

Die Komponente liefert folgende Daten zurück:

- Originalbild
- Regionenbild
- Vektor mit der Regionen- und Klassenzugehörigkeit jedes Pixels (für Visualisierungszwecke)
- Vektor der Observationen aller Pixel (abhängig von der verwendeten Segmentierungsform). Dieser kann beliebig weiterverarbeitet werden, etwa zu Visualisierungszwecken.

4.2.3.2 Containerkomponente ORC

Die Komponente **ORC** (*Object Recognition Container*) ist eine Komponente auf der Metaebene von ICE (vgl. 3.2.1). Sie dient der Steuerung in ihr enthaltener Komponenten bezüglich des Datenflusses und der Parameter. Die Containerkomponente kann auf alle Methoden und Variablen der untergeordeten Komponenten unter Beachtung der Zugriffsregeln (public, private) zugreifen und sie gegebenenfalls modifizieren. Sie ist dafür verantwortlich, die einzelnen Komponenten zu instanziieren, ihnen Daten zur Verfügung zu stellen und deren Ergebnisse zu interpretieren und zu verwalten.
Die hier implementierte Containerkomponente hat somit folgende Aufgaben (siehe auch Abb. 4.2):

- Führung des Experten durch die Verarbeitungsstufen des Systems mittels Dialogfenster
- Instanziierung aller benötigten Subkomponenten
- Import der Bilddaten
- Verwaltung:
  - ROIs
  - Observationsvektoren
  - Klassifikationsergebnisse

- Auswahl des besten Klassifikationssubsystems
- Einstellung der Parameter für die Segmentierung:
  - Bild, Selektor und Klassifikator
  - Größe und Verschiebungsvektor des gleitenden Fensters
  - Segmentierungsmethode
- Export der Ergebnisse als Bilder, .arff-Dateien und binäre Selektor- und Klassifikatordateien

Der Komponente stehen drei Operationsmodi zur Verfügung: *Train, Apply* und *Resume*. Mit *Train* kann das System mit neuen Daten trainiert werden. Das Training kann jederzeit beendet werden. Nach jedem bearbeiteten Bild werden die folgenden zugehörigen Daten abgespeichert:

- ROIs
- automatische Segmentierung
- .arff-Datei mit den Observationen
- .arff-Datei mit den Observationen der selektierten Merkmale

Zusätzlich werden

- Selektor
- Klassifikator
- .arff-Datei mit den Observationen **aller** Bilder
- .arff-Datei mit den Observationen der selektierten Merkmale **aller** Bilder

als aktueller Systemzustand abgespeichert.
Ein trainiertes System kann mit dem Modus *Apply* auf entsprechende Testdaten angewendet werden. Dazu werden der erstellte Selektor und Klassifikator geladen. Mit diesem Klassifikationssubsystem werden alle Bilder des Testdatensatzes segmentiert. Die resultierenden Segmentierungen werden abgespeichert.

Ein abgebrochener Trainingsvorgang kann mit *Resume* wiederaufgenommen werden. Dazu wird der zuletzt abgespeicherte Zustand des Systems wieder geladen. Dieser besteht aus Selektor, Klassifikator und den (selektierten) Observationen aller bisher bearbeiteten Bilder der Domäne. Danach erfolgt ein ganz normales Training (s.o.), welches auf den bestehenden Daten aufbaut.

4.2.3.3 Sonstige Komponenten

Die Datenstruktur für die Segmentierungen wird als *Label* bezeichnet und besteht prinzipiell aus zwei Bildern, von denen das eine für die Bildinformation und das andere für die Regionenzugehörigkeiten der einzelnen Pixel verwendet wird. Die Komponente *CreateLabel* fügt zwei gleichgroße Bilder zu einem Label zusammen. *SplitLabel* spaltet ein Label wieder in zwei Bilder auf. Jede Segmentierung erzeugt ein solches Label.
Mit der Komponente zur Merkmalsextraktion werden lokale oder globale Merkmale eines Bildes extrahiert. Dazu wird das o.g. Label verwendet, mit dem Bildinformation und Regionenzugehörigkeit verfügbar sind. Die Parameter der Komponente sind die zu extrahierenden Merkmale, die Fenstergröße und die Nummer der Region, aus der extrahiert werden soll. Die Komponente liefert einen Vektor aus Observationen der extrahierten Merkmale in Form einer .arff-Datei zurück.

4.3 Bewertungsmaße

4.3.1 Bewertung der Segmentierung

Zur Bewertung der Qualität der erzielten Segmentierungen wird der sogenannte 'Dice-Coeffizient' C_D verwendet. Er beschreibt das Verhältnis zwischen Schnitt und Vereinigung zweier Mengen. Mit ihm wird die Übereinstimmung zweier Segmentierungen R, R' des selben Objekts verglichen. Der Dice-Koeffizient liegt zwischen 0 (keine Übereinstimmung) und 1 (vollständige Übereinstimmung). Er ist wie folgt definiert

$$C_D = \frac{R \cap R'}{R \cup R'} \tag{4.1}$$

und wird nur dann verwendet, wenn eine Referenzsegmentierung existiert.
In der Literatur wird zur Bewertung vereinzelt das Verhältnis falsch klassifizier-

ter Pixel zur Gesamtzahl zu klassifizierender Pixel verwendet. In dieser Arbeit kann es nicht zum Einsatz kommen, da der durch die Verwendung des gleitenden Fensters entstehende Rand nicht mit einberechnet werden kann und das Ergebnis verfälschen könnte.
Eine weitere Möglichkeit zur Bewertung der Segmentierungsergebnisse besteht darin, die globalen Eigenschaften der segmentierten Regionen zu berechnen und mit denen einer Referenzsegmentierung (falls vorhanden) zu vergleichen.
Falls keine Referenzsegmentierung vorhanden ist, existiert kein objektives Maß zur Bewertung der Segmentierungsqualität. Allein der Experte kann hier subjektiv über die Güte der Segmentierung entscheiden.

4.3.2 Bewertung des Klassifikationssubsystems

Zum Vergleich der automatisch generierten Klassifikationssubsysteme eignet sich der Klassifikationsfehler. Er gibt an, wieviele Pixel von Objekt- und Hintergrundklasse der jeweils richtigen Klasse zugeordnet werden. Somit wird immer das Klassifikationssubsystem gewählt, dessen Fehler minimal ist.
Prinzipiell gibt auch der Klassifikationsfehler Auskunft über die Genauigkeit einer Segmentierung, da er ebenfalls die Anzahl der fälschlicherweise dem Hintergrund zugeordneten Objektpixel beinhaltet. Jedoch vernachlässigt er die Position der Pixel im Bild, weswegen der Dice-Coeffizient zur Bewertung der Segmentierung besser geeignet ist.

4.3.3 Bewertung der Einfachheit

Mit der Einfachheit wird der Aufwand für den Experten bewertet. Es läßt sich nur subjektiv beurteilen, wie hoch der Aufwand für die Vorgabe der Beispiele und die Korrektureingriffe ist. Das Markieren mit der Maus ist eine einfache und intuitive Form der Interaktion. Eine objektive Bewertung erfolgt durch die Häufigkeit der Interaktionen, also der Zahl der Korrektureingriffe durch den Experten. Eine konkrete Aussage läßt sich auch bei diesem Kriterium nicht ableiten, da die Zahl der Korrektureingriffe maßgeblich vom Problem und der Repräsentativität der Beispiele abhängt. Bei weniger repräsentativen Beispielen oder einer hohen Variabilität der Bildinhalte werden mehr Korrekturen nötig sein.

4.3.4 Statistische Bewertung

Die mittlere Segmentierungsqualität ist ein Hinweis darauf, ob die gefundenen Segmentierungsparameter dem Problem angemessen sind. Die Stichhaltigkeit dieses Hinweises läßt sich statistisch überprüfen. Dabei muß zwischen Trainings- und Testset unterschieden werden. Wird zur Berechnung der Genauigkeit der Parameterschätzung angenommen, daß sich die gemessenen Dice-Coeffizienten innerhalb eines Sets normalverteilen, dann genügt der Schätzfehler E folgender Formel [52]

$$E = z_{\frac{\alpha}{2}} \frac{\sigma}{\sqrt{N}} \tag{4.2}$$

mit α als Signifikanzniveau, $\frac{\alpha}{2}$ als rechtem Ausläufer der Standard-Normalverteilung, $z_{\frac{\alpha}{2}}$ als kritischem Wert, σ als Streuung der Grundgesamtheit und N als Stichprobengröße. Der tatsächliche Mittelwert μ_x liegt im Konfidenzintervall

$$\mu_x \epsilon \, [\bar{x} - E, \bar{x} + E] \tag{4.3}$$

zum Signifikanzniveau α mit $\bar{x}$ als gemessenem Mittelwert der Verteilung. Im allgemeinen wird $\alpha = 0,05\,\%$ als Signifikanzniveau verwendet. Daraus ergibt sich aus der Standardnormalverteilung $z_{\frac{\alpha}{2}} = 1,96$.

4.4 Zusammenfassung und Fazit

In diesem Kapitel wurde das in dieser Arbeit entwickelte System vorgestellt. Sein Kern ist ein adaptiver Segmentierungsalgorithmus, der auf dem Constrained Region Growing basiert. Dieses wurde um beliebige Randbedingungen erweitert. Ein echtes Klassifikationssubsystem – bestehend aus einer geeigneten Kombination von Merkmalsselektor und Klassifikator – entscheidet über die Regionenzugehörigkeit einzelner Pixel. Die lokalen Eigenschaften der zu segmentierenden Objekte werden als Randbedingungen verwendet. Das System adaptiert sich automatisch an die Vorgaben, die ihm interaktiv vom Benutzer gemacht werden. Es wählt die Randbedingungen automatisch aus einer Vielzahl von Randbedingungen aus und ist den bisher in der Literatur beschriebenen Ansätzen überlegen, bei denen diese Randbedingungen und die zugehörigen Pixelklassifikationsvorschriften manuell und subjektiv ausgewählt werden müssen.

Es wurden Maße zur Bewertung der einzelnen Systemteile und die Form der Implementierung in Java–Beans erläutert.
Zur Segmentierung werden die lokalen Eigenschaften der Objekte verwendet. Ein regionenorientiertes Verfahren ist hierfür besonders geeignet. Dieses Verfahren funktioniert nur, sofern die innere Homogenität der Objekte mit den zur Verfügung stehenden Merkmalen modelliert werden kann.
Das System ermittelt aus den zur Verfügung stehenden Parametern eine geeignete Parameterkombination. Die Plausibilität der gefundenen Parameter wird mit dem Klassifikationsfehler gemessen, der sich jedoch nicht direkt auf die Segmentierungsgüte abbilden läßt, da er positionsunabhängig ist. Er stellt jedoch eine sinnvolle Kenngröße für die Qualität des ermittelten Klassifikationssubsystems dar. Per definitionem ist der Dice-Coeffizient zur Bewertung der Segmentierungsgüte bzgl. einer Referenzsegmentierung geeignet. Mit beiden Maßen zusammen läßt sich angeben, ob die gefundenen Parameter das Problem zufriedenstellend lösen.

Kapitel 5

Erzielte Ergebnisse

In diesem Kapitel wird das System an vielen verschiedenen Beispielen getestet:

- Künstlich generierte Datensätze mit bekannten Eigenschaften
  - Texturdaten 1 (20.000)
  - Texturdaten 2 (100)
- Medizinische Datensätze
  - Spiculierte Massen (72)
  - Kaumuskeln (2)
- Diverse Bilder aus der Literatur
  - Künstliche Texturen (18)
  - Natürliche Szenen (6)
  - Berkeley Bilddatenbank (36)

Die Zahl in Klammern gibt die Anzahl der verfügbaren Bilder an. Mit diesen Beispielen sollen insbesondere die Einsatzmöglichkeiten der vorgestellten Methodik aufgezeigt werden. Dabei stellt sich zunächst die Frage, ob das System in der Lage ist, geeignete Segmentierungsparameter zu ermitteln. Dies soll mit künstlich generierten Datensätzen untersucht werden.
In der Literatur existiert kein Standarddatensatz, mit dem Segmentierungsalgorithmen miteinander verglichen werden können [53]. Zwar gibt es Ansätze [54] und

einige Datensätze für spezielle Anwendungsfälle [55], jedoch sind viele Methodiken für spezielle Anwendungen konzipiert und Vergleiche häufig nicht möglich. Im folgenden werden für jeden Datensatz zuerst die Ziele erläutert und anschließend die erzielten Ergebnisse aufgezeigt und diskutiert.

5.1 Synthetische Datensätze

Das System verfügt über drei freie Parameter für die Segmentierung: die Größe des gleitenden Fensters, den Merkmalsselektor und den Klassifikator. Es soll überprüft werden, ob das System diese Parameter passend zum Problem bestimmt.

5.1.1 Datensatz 1

Ziel dieses Experiments ist es, herauszufinden, ob das System in der Lage ist, für einen bekannten Datensatz die optimale Fenstergröße und das optimale Merkmal zu ermitteln. Diese sind die ersten wichtigen Parameter für die spätere Segmentierung.
Es wird eine Menge von Bildern mit bekannten Merkmalen erzeugt. Um das Problem einfach und überschaubar zu halten, wird als Merkmal nur die „Standardabweichung" ausgewählt.
Für eine aussagekräftige Statistik wird ein 2-Klassen-Problem mit 10.000 Bildern pro Klasse generiert. Jedes Bild ist 64×64 Pixel groß und besitzt den konstanten Grauwert $g(x, y) = 128$. Zur Unterscheidung der beiden Klassen wird auf jedem Bild Gauß'sches Rauschen erzeugt und es gilt:

$$\sigma\epsilon\,(0, 1) \Leftrightarrow \text{Bild gehört zu Klasse 0} \tag{5.1}$$

$$\sigma\epsilon\,(1, 2) \Leftrightarrow \text{Bild gehört zu Klasse 1} \tag{5.2}$$

Das Merkmal „Streuung" (SStdDev) dient somit der Trennung der beiden Klassen. Die Klassenzugehörigkeit wird für das Zentralpixel festgelegt und separat gespeichert. Das Rauschen ist über die Menge der Bilder ungefähr gleichverteilt. Da die Bilder endlich groß sind, entsteht bei der Generierung des Rauschens eine Ungenauigkeit, so daß die o.g. Intervallgrenzen um weniger als 8 % aufge-

weicht werden[1]. Extrahiert man das Merkmal „Streuung" um das Zentralpixel herum, ist die Schätzung der Verteilungsfunktion umso besser, je mehr Pixel in die Schätzung miteinbezogen werden. Mit einem 3×3 Pixel großen Fenster ist die Schätzung schlechter als mit einem 64×64 Pixel großen Fenster. Daher soll das System die (maximale) Fenstergröße von 64×64 Pixeln ermitteln.
Aus allen Bildern wird jeweils nur um das Zentralpixel in allen Größen des gleitenden Fensters eine Extraktion aller verfügbaren 45 lokalen Merkmale durchgeführt. Mit den erhaltenen Daten wird für jede Fenstergröße mittels der Blasensynthese das optimale Klassifikationssubsystem gesucht.
Das automatisch ausgewählte Klassifikationssubsystem besteht aus einer *Bidirectional Forward selection* und einem *J48 Pruned Tree*. Tabelle 5.1 zeigt die Trainings- und Generalisierungsfehler der verschiedenen Fenstergrößen mit dem gewählten Klassifikationssubsystem, die Namen der selektierten Merkmale sowie das Intervall der Streuung in der jeweiligen Fenstergröße.
Bei einer Fenstergröße von 64×64 Pixeln sind die Fehler am geringsten. Das bestimmte Merkmal ist in diesem Fall die Varianz.
Nachfolgend werden die in Tabelle 5.1 verwendeten Abkürzungen erläutert. **S** kennzeichnet ein statistisches Merkmal, **T** ein Texturmerkmal (vgl. Anhang A). Englische Bezeichnungen aus der Literatur werden beibehalten.

Abkürzung	vollständige Bezeichnung
SMin, SMax	minimaler und maximaler Grauwert
SStdDev	Standardabweichung
SEner, SEntr	Energie, Entropie
SGS, SNumGV	Greyspread, Number of Greyvalues
SModeGV	Mode Greyvalues
SVarCoeff	Variationscoeffizient
TEntr, TSumEntr, TDiffEntr	Entropie, Summenentropie, Differenzentropie
TIMCorr2,	Information Measure Correlation 2
TInvDiffMom	Inverse Difference Moment
TRunLRE	Run-length Long-run emphasis

[1] Für Klasse 0 ist $0 \leq \sigma \leq 1,07$; für Klasse 1 ist $1,02 \leq \sigma \leq 2,08$

Fenster-größe	Trainings-fehler [%]	Generali-sierungs-fehler [%]	automatisch ausgewählte(s) Merkmal(e)	Intervall der Streuung σ [*Grauwert*]
3×3	11,43	11,45	SMin, SGS, SNumGV, SModeGV	(0 ; 3,53)
5×5	6,13	6,41	SStdDev	(0 ; 8,23)
7×7	3,61	4,48	SMin, SMax, SEner, SEntr, SVarCoeff, SModeGV TEntr, TRunLRE, TSumEntr, TDiffEntr, TIMCorr2	(0 ; 2,66)
9×9	3,61	4,48	SMin, SMax, SEner, SEntr, SVarCoeff, SModeGV TEntr, TSumEntr, TDiffEntr	(0 ; 2,5)
11×11	2,22	2,58	SMin, SMax, SEner, SVarCoeff, SModeGV TSumEntr, TInvDiffMom	(0 ; 2,35)
16×16	1,80	1,92	SStdDev	(0 ; 2,27)
32×32	0,81	0,89	SMax, SEntr TEntr	(0 ; 2,2)
64×64	0,42	0,43	SStdDev	(0 ; 2,08)

Tabelle 5.1: *Die Trainings- und geschätzten Generalisierungsfehler in Prozent und die verwendeten Merkmale für verschiedene Fenstergrößen. Zur Kontrolle finden sich in der ganz rechten Spalte die in den verschiedenen Fenstergrößen berechneten Intervallgrenzen der Streuung des Gesamtdatensatzes. Die englischen Bezeichnungen aus der Literatur wurden beibehalten.*

5.1.1.1 Schlußfolgerungen aus Datensatz 1

Das Klassifikationssubsystem hat die erwartete Fenstergröße von 64×64 Pixeln ermittelt. Das gewählte Merkmal ist wie erwartet die Standardabweichung (Streuung).
Dieses einfache Beispiel läßt die Annahme zu, daß die Auswahl der Fenstergröße und der Merkmale dem jeweiligen Problem angemessen ist. Daher soll im folgenden untersucht werden, wie gut die Segmentierungsqualität ist, die mit dem vorgestellten System auf einem bekannten Datensatz zu erreichen ist.

5.1.2 Datensatz 2

Mit Hilfe dieses Experiments soll herausgefunden werden, wie gut das System ein vorgegebenes Segmentierungsproblem löst. Insbesondere soll die Segmentierungsgüte in Form des Dice-Coeffizienten gemessen werden. Hierfür wird ein Datensatz mit künstlichen Objekten erstellt: Der Datensatz besteht aus 100 Bildern der Kantenlänge 75×75 Pixel. In jedes Bild ist zentriert ein Quadrat von 25×25 Pixeln Kantenlänge eingebettet. Der Bildhintergrund wird mit der Brodatz-Textur D57 [56] gefüllt, das Quadrat mit D9. Aus diesem Bild werden 99 weitere erzeugt, indem das Quadrat in $3{,}6\,^\circ$ Schritten um den Bildmittelpunkt rotiert wird. Dadurch werden viele verschiedene Übergänge zwischen beiden Texturen erzeugt, um ein Auswendiglernen der Übergänge zu vermeiden. Zu jedem Bild ist die optimale Segmentierung bekannt. Diese 100 Bilder stellen die statistische Grundgesamtheit dar.
Es soll herausgefunden werden, wie gut der entworfene Algorithmus das vorliegende Segmentierungsproblem lösen kann. Aus Zeitgründen[2] können nicht sämtliche Bilder zur Merkmalsextraktion verwendet sowie das Klassifikationssubsystem optimiert werden. Unter der Annahme, daß sich aufgrund der ähnlichen Bildinhalte die Klassifikationsfehler in den verschiedenen Fenstergrößen für jedes Bild ähnlich verhalten, wird nur mit dem ersten Bild (s. Abb. 5.1 (a)) eine Fenstergröße festgelegt: Mit ICE [46] wird ein Klassifikationssubsystem aufgebaut, welches das

[2] Bei 20 Pixeln pro Sekunde dauert die Merkmalsextraktion für 100 Bilder à 75×75 Pixel in 7 Fenstergrößen bei 45 Merkmalen etwa 54h. Der zusätzliche Rechenaufwand für die Bestimmung des besten Klassifikationssubsystems beträgt etwa 2500 h [46].

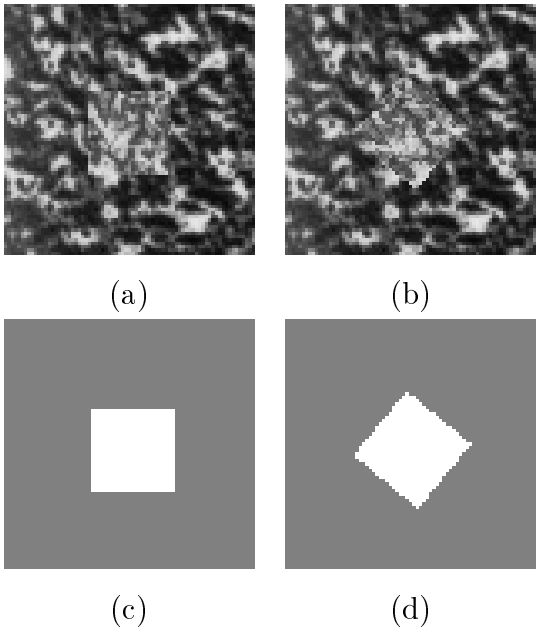


Abbildung 5.1: *Beispielbilder. Oben: Zwei Originalbilder bei 0° und 50,4° Rotation. Unten: Die jeweilige optimale Segmentierung. Die Rotation fand ohne Anti-Aliasing statt, um die Originaldaten nicht zu verändern.*

Problem bestmöglich beschreibt: Für die Merkmalsselektion wird eine bidirektionale BestFirst-Suche verwendet [35] und als Klassifikator ein *Pruned Tree* (J48) [45]. Beide haben sich in der Praxis häufig als gute Kombination erwiesen[3]. Die ausgewählten Merkmale sind in Tab. 5.2 aufgelistet. Es handelt sich dabei um drei Merkmale, die aus der Cooccurrence-Matrix berechnet werden und vier histogrammbasierte Merkmale.

[3] Tests mit der Blasensynthese auf Teildatensätzen aller Bilder bekräftigen die Wahl des Klassifikationssubsystems.

Nr.	Merkmal
1	TCorr
2	TDiffVar
3	TClustShade
4	SMin
5	SSkew
6	SModeGV
7	SMed

Tabelle 5.2: *Durch die bidirektionale BestFirst-Suche ausgewählte Merkmale.* ***S*** *kennzeichnet ein Merkmal des Histogrammes,* ***T*** *ein aus der Cooccurrence-Matrix bestimmtes Merkmal.*

Blockgröße	NaiveBayes [%]	J48 [%]
3×3	19,25/19,25	7,96/10,04
5×5	12,28/12,32	2,88/6,01
7×7	11,07/10,94	1,85/4,79
9×9	10,40/10,45	0,98/4,08
11×11	7,91/7,91	0,68/3,17
16×16	4,69/4,72	3,75/4,36
32×32	9,04/9,24	1,96/5,37

Tabelle 5.3: *Die Trainings- und geschätzten Generalisierungsfehler in Prozent für verschiedene Fenstergrößen und Klassifikatoren. Die jeweils linke Zahl gibt den Trainingsfehler an, die rechte den Generalisierungsfehler, der mit zehnfacher Kreuzvalidierung ermittelt wurde.*

Tabelle 5.3 zeigt die Trainings- und geschätzten Generalisierungsfehler[4] des ersten Bildes bei verschiedenen Blockgrößen. Diese beiden Fehler sind bei einer Fenstergröße von 11×11 Pixeln mit $0,68\,\%$ bzw. $3,17\,\%$ am geringsten. Daher wird 11×11 Pixel als Fenstergröße gewählt.
In dieser Fenstergröße werden aus allen 100 Bildern jeweils alle Merkmale extrahiert und ein Klassifikationssubsystem mit o.g. Selektor-Klassifikator-Kombination

[4] geschätzt mit zehnfacher Kreuzvalidierung

aufgebaut. Dieses wird zum Segmentieren aller Bilder verwendet.

5.1.2.1 Segmentierung

Abbildung 5.2 zeigt neben den Originalbildern auch die Ränder der automatisch segmentierten zentralen Quadrate. Die mittlere Segmentierungsgüte μ_{C_D} liegt bei 0,945, die Streuung σ_{C_D} bei 0,008. Als geringster Wert für C_D wird 0,924 und als größter 0,965 ermittelt.

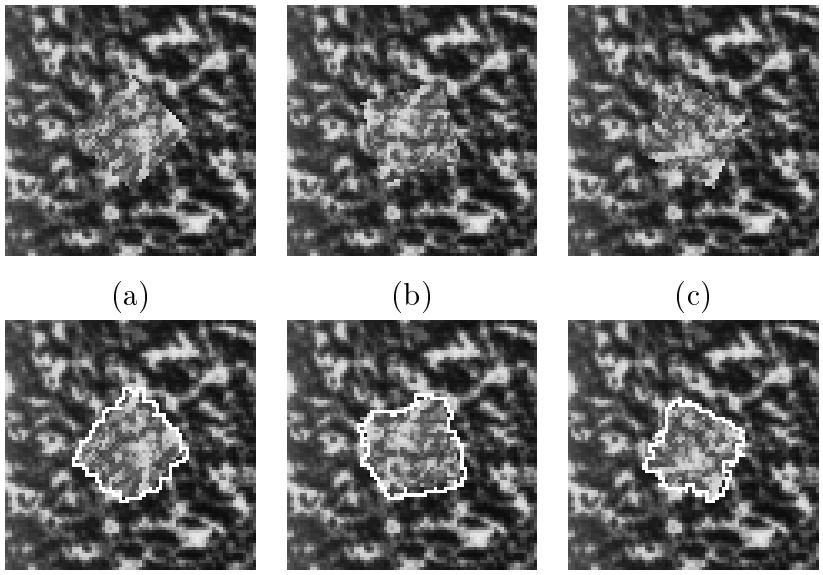


Abbildung 5.2: *Beispielhafte Segmentierungsergebnisse. Oben die Originalbilder, unten die jeweilige automatische Segmentierung. Die weiße Umrandung kennzeichnet die Trenngrenze zwischen den Regionen. (a) zeigt die minimale Segmentierungsgüte ($C_D = 0,924$), (b) die mittlere ($C_D = 0,945$) und (c) die maximale ($C_D = 0,965$). Die zentralen Quadrate sind um 46,8°, 284,4° bzw. 64,8° rotiert.*

Der Klassifikationsfehler des ermittelten Klassifikationssubsystems stellt den in dieser Konfiguration minimal zu erreichenden Klassifikationsfehler dar. Er soll als Referenz für den folgenden Versuch dienen.

5.1.2.2 Inkrementelle Betrachtung

Es soll untersucht werden, wie sich das System verhält, wenn die Bilder dem System erst nach und nach zur Verfügung gestellt werden. Insbesondere soll er-

mittelt werden, ob die Segmentierungsgüte mit steigender Anzahl von Trainingsdaten zunimmt. Es wird erwartet, daß die Segmentierungsgüte anfänglich niedrig ist und im Laufe des Lernens stets zunimmt. Für den Generalisierungsfehler wird erwartet, daß er zu Beginn des Trainings hoch ist, sich aber dem Trainingsfehler annähern und abnehmen wird.
Die Daten werden in verschiedene Trainings- und Testsets aufgespaltet: Das erste Trainingsset besteht nur aus dem ersten Bild. Das dazugehörige Testset besteht aus den Bildern 2–100. Für das zweite Trainingsset werden die Bilder 1 und 2 verwendet; das Testset besteht entsprechend nur noch aus den Bildern 3–100, usw. Mit jedem Trainingsset wird ein Klassifikationssubsystem (s.o.) erstellt und auf das zugehörige Testset angewendet. Durch die Kenntnis des Datensatzes kann berechnet werden, wie gut das jeweilige Klassifikationssubsystem der bearbeiteten Bilder bezogen auf die Gesamtdaten ist. Somit kann für diesen inkrementellen Lernvorgang jederzeit die Güte von Klassifikationsfehler und Segmentierung bestimmt werden.

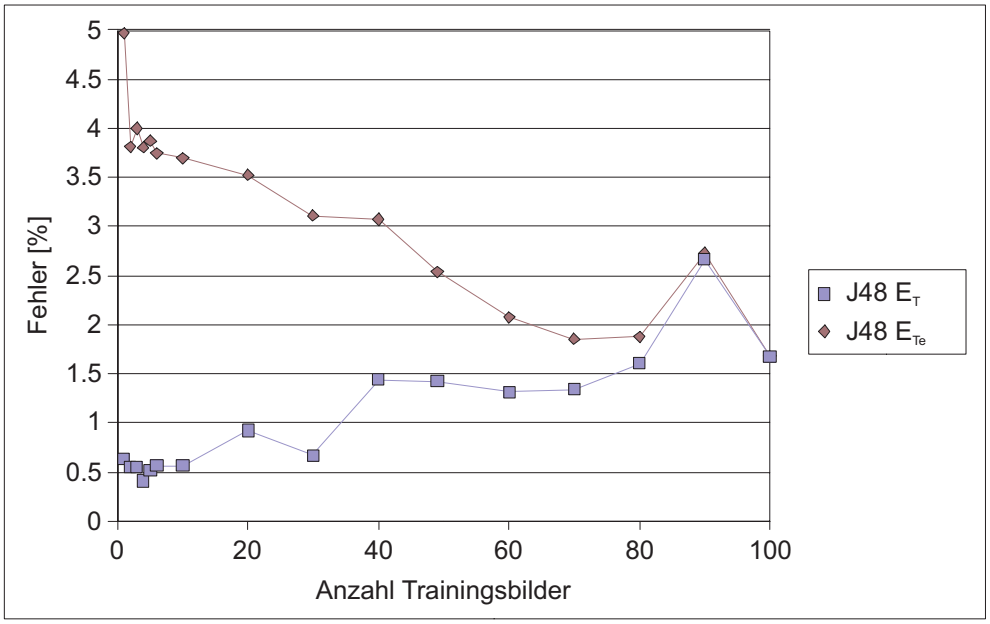


Abbildung 5.3: *Entwicklung von Trainings- und Testfehler für den J48-Klassifikator über die wachsende Stichprobe. E_T bezeichnet den Trainings-, E_{Te} den Testfehler.*

Abbildung 5.3 zeigt den Verlauf von Trainings- und Testfehler für eine steigende Anzahl von Trainingsbildern. Während die beiden Kurven anfangs noch 4,3 %

auseinanderlagen, nähern sie sich erwartungsgemäß immer weiter an, bis sie sich bei 100 Bildern schließlich treffen.
Betrachtet man die sequentielle Verarbeitung der einzelnen Bilder, so ergibt sich bei steigender Zahl von Trainingsbildern eine bessere mittlere Segmentierungsgüte (s. Abb. 5.4). Bei 91 Trainingsbildern zeigt sich ein Abfall beider Kurven um 3 %. Der Schätzfehler E sinkt von anfänglich 0,016 bei einem Trainingsbild auf 0,0016 bei 99 Trainingsbildern.

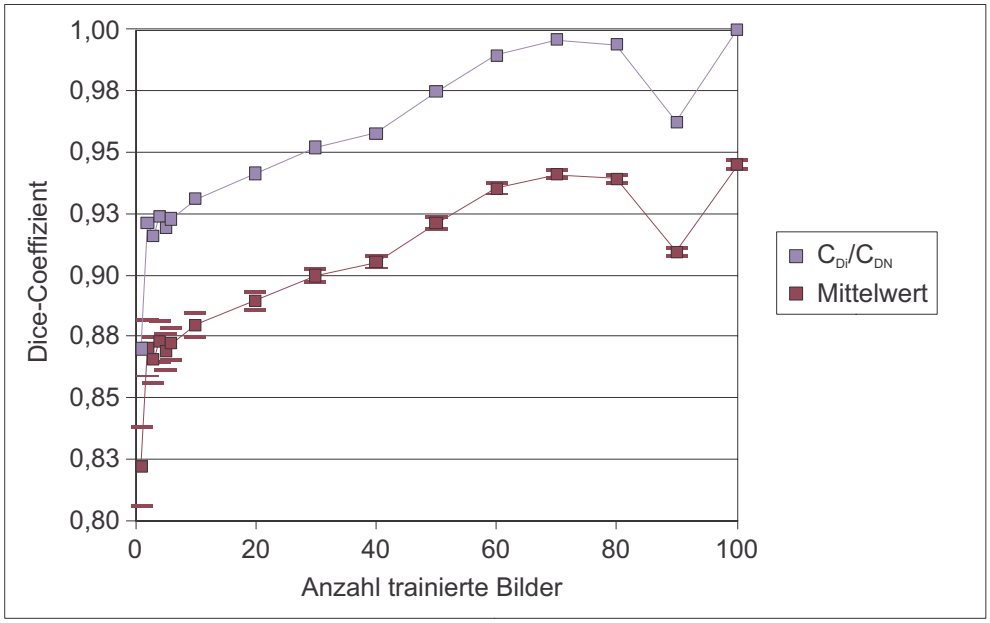


Abbildung 5.4: *Statistik der Segmentierungsgüte. Über die Anzahl der Trainingsbilder aufgetragen sind der mittlere Dice-Coeffizient C_D als Maß für die Segmentierungsgüte und das Verhältnis der mittleren Segmentierungsgüte nach i gelernten Bildern C_{Di} zur erreichbaren Güte C_{DN}. Um die Meßpunkte der Mittelwertskurve ist das jeweilige Konfidenzintervall eingezeichnet.*

In Abbildung 5.5 werden die Unterschiede der mittleren Segmentierungsgüten zwischen Trainings- und Testset deutlich. Im Trainingsset ist sie immer höher als im Testset. Die Standardabweichung liegt im Mittel bei 0,03. Die Spannweite[5] von C_D des Testsets verringert sich von 0,16 auf 0,04.

[5] Die Spannweite ist die Differenz zwischen Maximum und Minimum

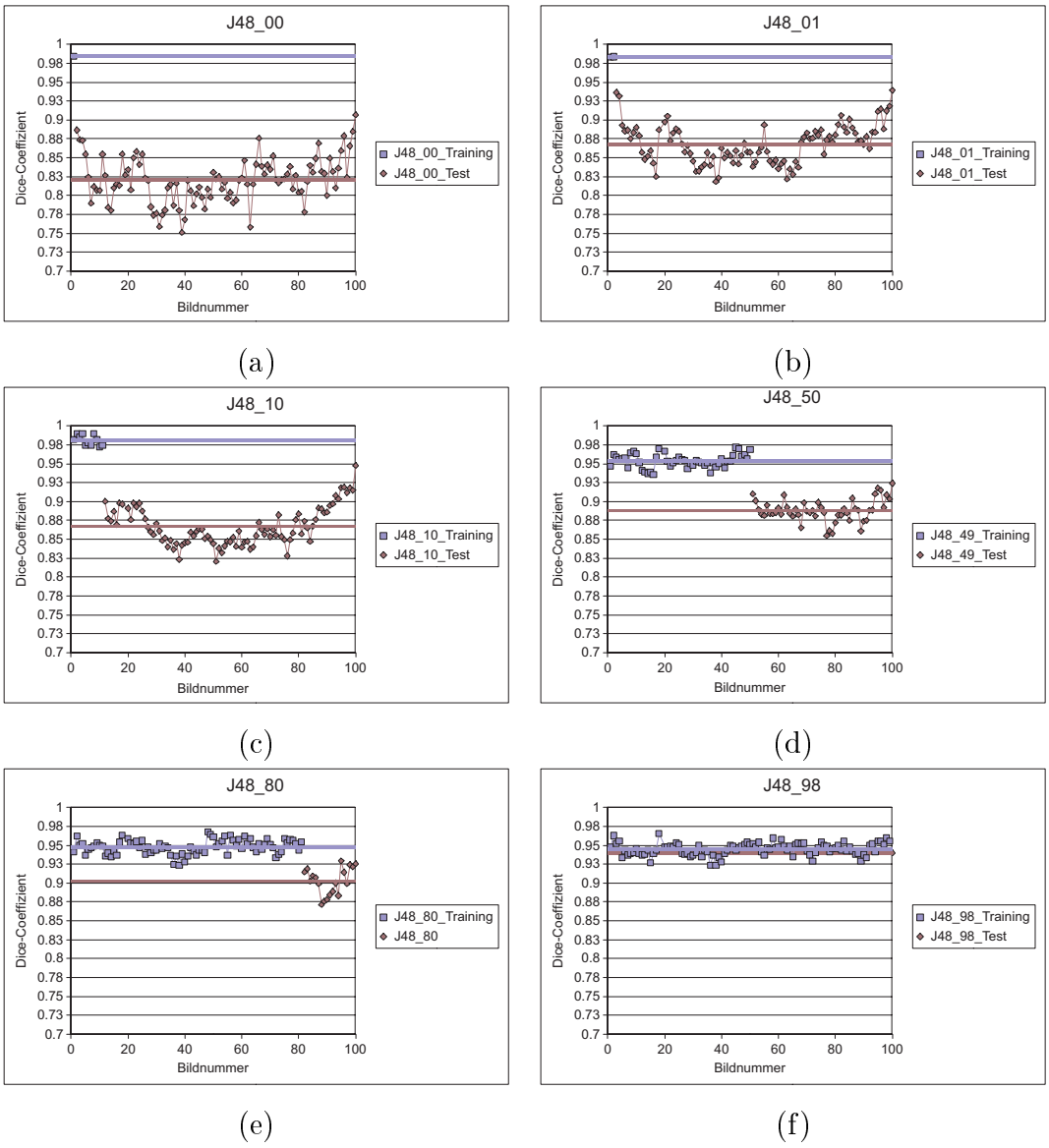


Abbildung 5.5: *Verlauf der Segmentierungsgüte beispielhaft ausgewählter Trainingsvorgänge. (a) Mit dem ersten Bild trainiert und auf den folgenden getestet ergibt sich die Kurve J48_00. (c) - (f) zeigen das Training mit 2, 11, 51, 81 und 99 Bildern und Test auf den jeweils restlichen Bildern. Parallel zur x-Achse sind für Trainings- und Testset in der jeweiligen Farbe die mittleren Segmentierungsgüten mit aufgetragen.*

5.1.2.3 Schlußfolgerungen aus Datensatz 2

Das System hat Segmentierungsparameter ermittelt, mit denen das vorliegende Textursegmentierungsproblem gelöst werden kann. Der mittlere Dice-Coeffizient zeigt pro hinzugefügter Trainingsmenge (1 bis 10 Bilder) signifikante Veränderungen: Insgesamt steigt er von 0,822 auf 0,945. Mit dem aufgebauten Klassifikationssubsystem wird ein sehr gutes mittleres Segmentierungsergebnis erreicht. Das System lernt von Bild zu Bild mehr über das Segmentierungsproblem. Trainings- und Testfehler nähern sich mit steigender Zahl von Trainingsdaten erwartungsgemäß aneinander an. Den gleichen Sachverhalt zeigt auch die Annäherung der jeweiligen Mittelwertsgeraden in Abb. 5.5.
Die Generalisierungsfähigkeit steigt demzufolge mit zunehmender Trainingsmenge. Dadurch wird der Abfall der Segmentierungsgüte, der beim Übergang vom Trainings- ins Testset auftritt, im Mittel von $0,164$ auf $0,019$ verringert.
Die Ergebnisse werden mit einer festen Selektor-Klassifikator-Kombination erstellt, die möglicherweise suboptimal ist.
Mit dem entwickelten Algorithmus werden automatisch Parameter ausgewählt, mit denen sehr gute Segmentierungergebnisse erzielt werden können. Dies wurde an einem Texturproblem nachgewiesen.

5.1.2.3.1 Probleme bei 91 Trainingsbildern Das Absinken der Segmentierungsgüte um 3 % bei 91 Trainingsbildern (vgl. Abb. 5.4) scheint durch den verwendeten Selektor beeinflußt zu werden. Im 91. Bild ist offensichtlich Information enthalten, die der Selektor nicht ausreichend analysieren kann. Dadurch verwirft er einige Merkmale. Fügt man manuell diejenigen Merkmale hinzu, die er bei Bild Nr. 90 und Nr. 92 selektiert, so sinken Trainings- und Testfehler an diesem Punkt auf 1,72 % und 1,82 %. Der Ausreißer fügt sich damit gut in die Kurve ein.
Die manuelle Auswahl des Selektionsalgorithmus erzeugt im vorliegenden Fall kein optimales Ergebnis. Mit der Blasensynthese könnte dieses Problem umgangen werden[6].
Das alleinige Auftreten des Problems bei diesem Bild deutet auf einen zufälligen

[6] Die für die Blasensynthese benötigte Rechenzeit auf einem 2GHz PC wird auf ca. 360 h geschätzt [46].

statistischen Ausreißer hin. Eine Mittelung über die Klassifkationsfehler mit permutierten Trainingsdaten würde eine genauere Fehlerkurve hervorbringen. Aus zeitlichen Gründen ist diese jedoch nicht berechenbar[7]. Somit bleibt festzuhalten, daß ein Abwärtstrend der Testfehler deutlich erkennbar ist.

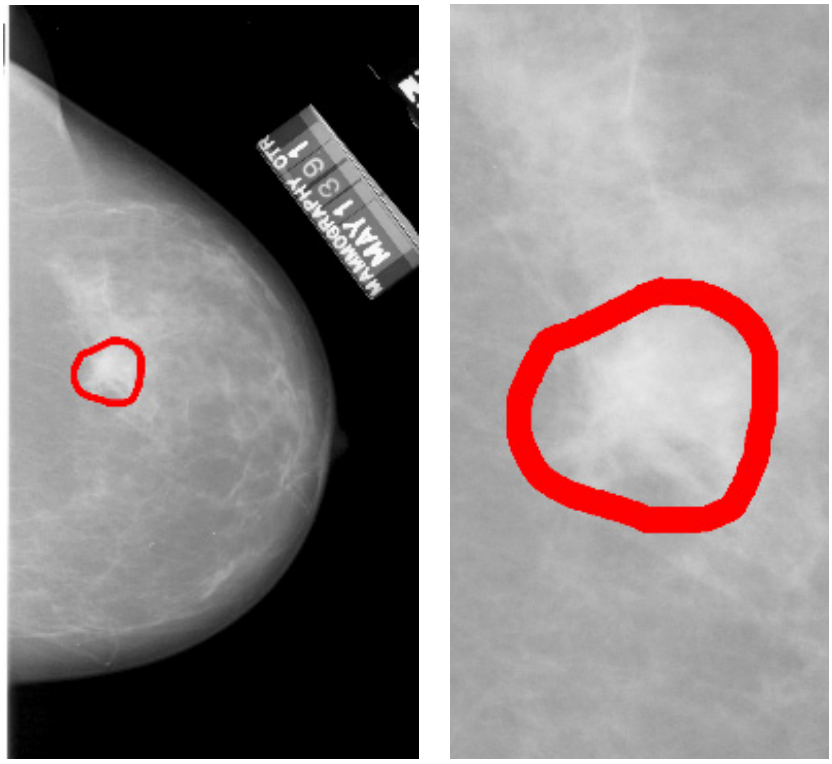
5.2 Zwischenergebnis

Die Ergebnisse der beiden synthetischen Probleme zeigen, daß sich die vorgestellte Methodik zum interaktiven Training auf Segmentierungsprobleme sehr gut eignet. Es werden geeignete Fenstergrößen, Merkmale, Selektor- und Klassifikatorkombinationen gefunden und gute Segmentierungen erzielt. Es wird angenommen, daß ähnlich gute Ergebnisse auch auf natürlichen Bildern erzielt werden können. Diese Ergebnisse werden im folgenden beschrieben. Dabei ist anwendungsabhängig jeweils darauf zu achten, ob mit der Methodik eine pixelgenaue Segmentierung generiert oder bestimmte Objekte zuverlässig detektiert werden sollen. Die Segmentierungsaufgabe wird in diesen Fällen (z.B. Spiculierte Massen) wegen des noch sehr hohen Rechenaufwands in eine Detektionsaufgabe umgewandelt.

[7] Eine sinnvolle Mittelung sollte wenigstens aus 10 Messungen des Klassifikationsfehlers bestehen. Die Rechenzeit für eine Messung beträgt etwa 2 Wochen.

5.3 Spiculierte Massen

Spiculierte Massen (engl. *Stellate Lesions*) sind wichtige Indikatoren für etwa 75 % aller bösartigen Brusttumore. Sie definieren sich durch eine unförmige zentrale Masse, von der Gewebestränge (Spiculi) sternförmig wegführen [57]. Ihre automatische Erkennung ist eine der schwierigsten Aufgaben in der medizinischen Bildverarbeitung. Das Aussehen der Massen und insbesondere der Spiculi variiert von Fall zu Fall, da sie auf dem Mammogramm von umliegendem Gewebe überlagert werden. In den meisten Fällen ist der menschliche Experte in der Lage, den Tumor zu erkennen und zumindest die zentrale Masse zu markieren (s. Abb. 5.6). Eine akkurate Segmentierung ist auch dem Experten unmöglich, da die Spiculi auch in gesundes Gewebe eindringen.



(a) (b)

Abbildung 5.6: *Röntgenmammogramme mit einer markierten Spiculierten Masse (rot). (a) Die äußere Erscheinung der Masse unterscheidet sich optisch nur wenig von gesundem Kontrollgewebe. (b) Es wird deutlich, daß keine vollständige manuelle Segmentierung angegeben werden kann.*

Das Ziel ist die Detektion der Spiculierten Massen, so daß die detektierten Regionen zu den verfügbaren manuellen Markierungen passen. Die Massen sollen aufgrund der Rechenzeit und der Schwierigkeit, die Segmentierungsergebnisse zu evaluieren, nur detektiert werden.

Im folgenden werden die verwendeten Daten und die Verarbeitung der digitalen Mammogramme beschrieben. Die erzielten Ergebnisse werden diskutiert und mit denen aus der Literatur verglichen.

Verwendet wird der BCRP-Teil (Breast Cancer Research Program) der DDSM-Datenbank (Digital Database for Screening Mammography) [58]. Er besteht aus digitalisierten Mammogrammen und deren manuellen Segmentierungen. Die Bilder haben eine Größe von $\sim 4000 \times 8000$ Pixeln@12bit und eine Auflösung von $43.5 \frac{\mu m}{Pixel}$. Die manuellen Segmentierungen decken einen Teil der zentralen Massen und in einigen Fällen auch Regionen mit Spiculi ab. Die Datenbank ist geteilt in ein Trainings- und ein Testset mit 39 bzw. 40 Bildern. Aus dem Trainingsteil konnten jedoch nur 32 Bilder fehlerfrei dekomprimiert werden. Die Bildgröße wurde auf 25 % reduziert, so daß 1 Pixel 174 μm entspricht.
Die Bilder werden sequentiell in der vorliegenden Reihenfolge bearbeitet: Auf dem ersten Bild definiert der Experte ROIs für die Läsion und gesundes Kontrollgewebe. Aufgrund der zu erwartenden großen Datenmenge wird der Pixelklassifikator nicht optimiert. Stattdessen werden wieder die bidirektionale BestFirst-Suche und der Klassifikationsbaum J48 verwendet. Mit diesen Angaben bestimmt das System automatisch die benötigten Segmentierungsparameter. Statt einer pixelgenauen Segmentierung werden hier nur alle Regionen detektiert, die zu einer spiculierten Masse gehören könnten. Zu diesem Zweck wird mit der Segmentierungsvariante „Windowed" eine grobe Segmentierung erzeugt. Der Verschiebungsvektor des gleitenden Fensters liegt zwischen 8 und 20 Pixeln. Diese groben Regionen werden vom Experten begutachtet und gegebenenfalls korrigiert. Die gefundenen Parameter werden auf das Folgebild angewendet und der Experte verfährt wie in Kapitel 4 beschrieben.
Mit den verwendeten Algorithmen kann ein gutes Klassifikationssystem erstellt werden. Die Detektionsergebnisse sind in Tabelle 5.4 dargestellt. Die Detektionsrate im Training liegt bei 91 % und im Test bei 70 %. Der gesamte Prozeß der Definition der ROIs und der Überprüfung der Zwischenergebnisse dauerte

auf einem Pentium IV 1,6 GHz mit 1 GB RAM etwa 4 h, die Berechnung der Parameter zusätzlich ungefähr 60 h. Die Anzahl der Interaktionen lag im Mittel bei drei pro Bild. Die Segmentierungsparameter mußten dreimal pro Bild neu berechnet werden.

	Detektions-rate für Läsionen	Klassifika-tionsfehler (Pixel)	Falsch Positive (Läsionen)	Falsch Positive pro Bild	Inter-aktionen pro Bild	Itera-tionen pro Bild
Training	91%	1.1%	267	6	3.4	2.7
Test	70%	2.4%	325	8	0	0

Tabelle 5.4: *Der Klassifikationsfehler im Testfall wurde mit zehnfacher Kreuzvalidierung aus 37823 Observationen à 45 lokalen Merkmalen bestimmt.*

Während der sequentiellen Verarbeitung der Bilder variiert die Anzahl der benötigten Merkmale zwischen 1 und 12; am Ende werden 7 Merkmale als besonders wichtig erachtet: SMean, SVar, SMin, SMax, TSumAvg, THom und TRunGLD. Durch die inkrementelle Bearbeitung der Bilder wird eine Genauigkeit von 97 % im Training erreicht. Nachdem das fertige Klassifikationssubsystem erstellt ist, wird es erneut auf alle Trainingsbilder angewendet. Zwei Trainingsmassen wurden nicht mehr detektiert, so daß sich die Genauigkeit um 6 % auf 91 % verringert.

	Methode	Heath [58]	Kobatake [59]	Kupinski [60]	Li [61]
TP Training	97 % (91 %)	88 %	kA	kA	kA
TP Test	70 %	86 %	92–96 %	90.4 %	70-90 %
avg. FP	8	8	kA	1.3	1.6

Tabelle 5.5: *Vergleich der Echt Positiven (True Positives (TP)) in Trainings- und Testfall sowie der mittleren Falsch Positiven Detektionen pro Bild mit Literaturangaben.*

5.3.1 Diskussion der Spiculierten Massen

Das System kann mit wenig Aufwand für den Benutzer erfolgreich auf die Detektion Spiculierter Massen angepaßt werden und zeigt eine gute Generalisierungsfähigkeit.

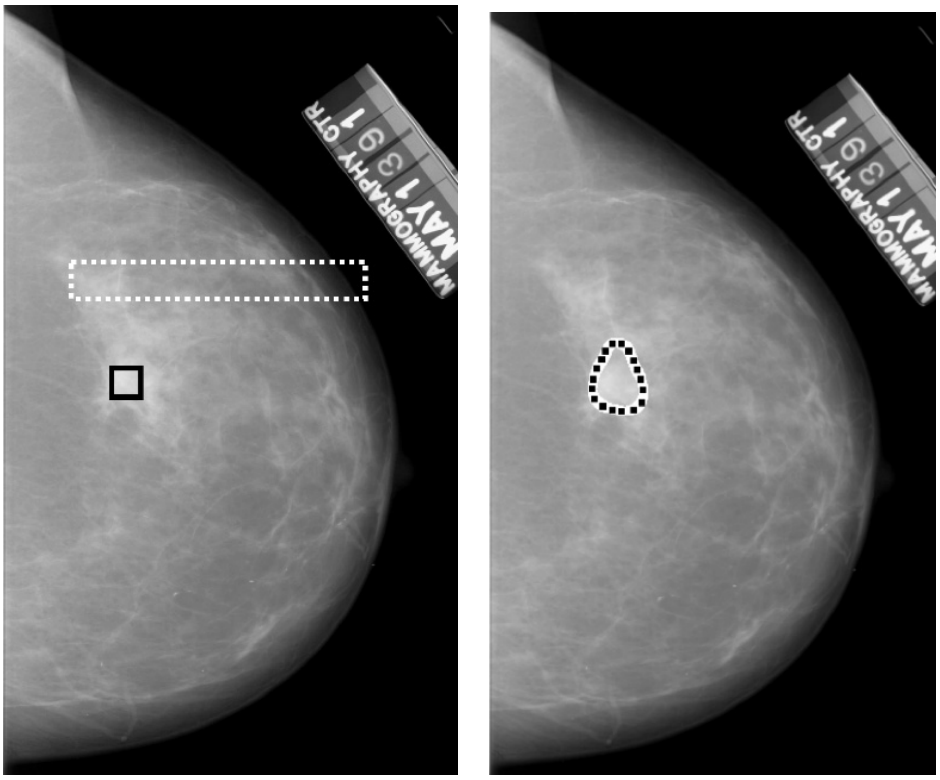


Abbildung 5.7: *Links: Das Originalmammogramm mit den vom Experten definierten ROIs für Läsion (schwarz) und gesundes Kontrollgewebe (weiß gepunktet). Rechts: Die Läsionsgrenze nach der Segmentierung.*

Die Detektions- und Segmentierungsergebnisse sind subjektiv zufriedenstellend. Eine quantitative Bewertung der Segmentierung wird als nicht sehr sinnvoll erachtet, da nur eine manuelle Segmentierung pro Bild zur Verfügung steht, die in einigen Fällen sogar unzureichend zu sein scheint. Grundsätzlich ist eine pixelgenaue Segmentierung zur quantitativen Bewertung der Segmentierungsergebnisse unter Berücksichtigung des extrem hohen Rechenaufwands möglich.
30 % der Massen des Testsets können nicht klar detektiert werden. Es zeigt sich, daß diese Massenformen nicht im Trainingsset repräsentiert sind. Daher ist es dem Lernalgorithmus nicht möglich, eine korrekte Generalisierung aus des Trainingsdaten abzuleiten. Folglich ist das Trainingsset mit nur 33 Mammogrammen nicht ausreichend repräsentativ für das Problem.
Im Vergleich mit bekannten Methoden sind die erzielten Ergebnisse ähnlich, jedoch ist die vorgestellte Methodik nicht speziell für die Erkennung Spiculierter

Massen entwickelt worden. Daher ist sie den anderen Methoden bei vergleichbaren Ergebnissen überlegen.
Obwohl die Zahl der Interaktionen und Iterationen sehr gering ist und zeigt, daß die Ergebnisse mit wenig Aufwand erzielt werden können, ist die Rechenzeit noch sehr hoch.
Die verwendete Selektor–Klassifikator–Kombination ist nicht zwingend optimal. Eine bessere Kombination kann die Ergebnisse noch verbessern.
In diesem Versuch wurden nur die lokalen Eigenschaften zur Segmentierung benutzt. Bei Einbeziehung der Formmerkmale der Läsionen könnte die Erkennungsrate weiter gesteigert und die Zahl der Falsch Positiven verringert werden.

5.4 Weitere Beispiele

Die vorgestellte Methodik wurde an vielen weiteren Bildern sehr unterschiedlichen Inhalts getestet. Einige dieser Beispiele werden im folgenden vorgestellt: verschiedene synthetische Texturbilder [62, 21, 22] und natürliche Szenen [54, 62] aus der Literatur und dem Internet.
Im Unterschied zu den bisher gezeigten Ergebnissen sind die folgenden Daten nicht als komplette Datensätze zu verstehen, d.h. die Bilder haben keine ähnlichen Inhalte und stehen auch sonst in keiner Beziehung zueinander. Jedes Bild ist für sich ein abgeschlossenes Segmentierungs-/Erkennungsproblem. Insbesondere entstehen Trainingsdaten nur aus jeweils einem Bild; Testdatensatz ist der Rest des Bildes selbst. Die gefundenen Parameter sind für andere der folgenden Bilder im allgemeinen nicht nutzbar.
Ein Klassifikationssubsystem aus bidirektionalem BestFirst-Selektor und J48-Baumklassifikator hat sich in der Praxis als akzeptabler und schneller Ansatz erwiesen und wird daher für die folgenden Beispiele eingesetzt.
Zur besseren Darstellung wurden die Ränder der Segmentierungen dilatiert und über das Originalbild gelegt. So kann ein rein visueller Eindruck der Segmentierungsqualität gewonnen werden.

5.4.1 Kaumuskeln

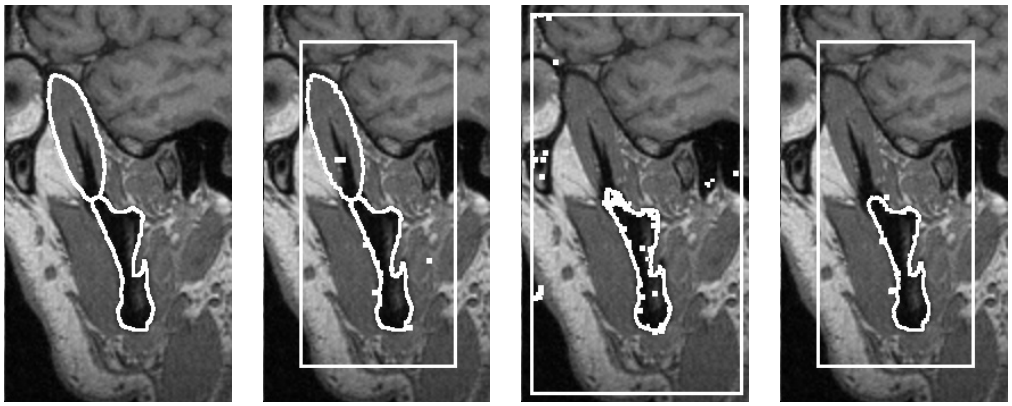
Hierbei handelt es sich um zwei Schichtbilder aus einem T1-gewichteten MRT-Datensatz des menschlichen Schädels. Es liegen manuelle Segmentierungen einer Expertin von Kaumuskulatur und Unterkieferknochen vor (s.Abb. 5.8 & 5.9). Ziel ist es, Parameter zu finden, die die manuelle Segmentierung möglichst gut reproduzieren. Da in den Datensätzen Kieferknochen, zwei Muskeln und Hintergrund zu finden sind, werden Mehrklassenprobleme (bis zu vier Klassen) untersucht. Der Knochen ist für die Expertin zusätzlich von Interesse, daher werden auch die 2-Klassen-Probleme (Knochen und Hintergrund (incl. Muskel)) betrachtet.

5.4.1.1 Datensatz 1

In diesem Datensatz sind ein Muskel, Knochen und Hintergrund abgebildet (vgl. Abb. 5.8 (a)). Betrachtet man zunächst das 3-Klassen-Problem der Trennung von Knochen und Muskel zum Hintergrund ergibt sich eine gewählte Fenstergröße von 64×64 Pixeln bei einem Trainings- bzw. Generalisierungsfehler von $0,25\,\%$ bzw. $1,15\,\%$. Die ermittelten Merkmale waren TMaxProb, TCorr, TSumAvg, TDiffVar, TIMCorr2, TClustShade, SMean, SMin, SMax, SModeGV und SMed. Als 2-Klassen-Problem der Trennung von Knochen zum Hintergrund ergibt sich eine Fenstergröße von 16×16 Pixeln, Trainings- bzw. Generalisierungsfehler von $0,45\,\%$ bzw. $0,85\,\%$ und die Merkmale SMean, SMin, SModeGV, SMed.

Klassen	Auflösung	Anzahl Merkmale	Fehler (E_T ; E_G)	C_D Knochen	C_D Muskel
3	64×64	12	0,25 % ; 1,15 %	0,992	0,993
2	16×16	4	0,45 % ; 0,85 %	0,959	k.A.

Tabelle 5.6: *Die ermittelte Auflösung, die Zahl der verwendeten Merkmale und der Dice-Coeffizient für die Klassen Knochen und Muskel.*



(a) (b) (c) (d)

Abbildung 5.8: *Überlagerte Segmentierungen von Knochen (unten) und Kaumuskelgewebe (oben) über das Originalbild. (a) Manuelle Segmentierung. (b) Automatische Segmentierung. (c) Automatische Segmentierung mit Parametern des 2-Klassen-Falls. (d) Segmentierung mit Fenstergröße und Merkmalen aus (b) und adaptiertem Klassifikator.*

5.4.1.1.1 Fazit Für beide Probleme werden geeignete Parameter gefunden und sehr gute Segmentierungsergebnisse erzielt. Verwunderlich sind der um 0,2 % bis 0,3 % größere Trainings- und Generalisierungsfehler im 2-Klassen-Fall: Durch das Zusammenfassen zweier Klassen sollte das Klassifikationsproblem eigentlich vereinfacht werden. Der Selektor scheint nicht in der Lage zu sein, dies zu erkennen. Dies zeigt sich in den unterschiedlichen Merkmalsmengen. Die Texturmerkmale finden keine Beachtung mehr. Auch die Segmentierungsergebnisse sind im 2-Klassen-Fall deutlich schlechter; der C_D ist 0,033 verringert. Eine manuelle Anpassung der Auflösung und der Merkmale verbessert die Ergebnisse deutlich:

Klassen	Auflösung	Anzahl Merkmale	Fehler (E_T ; E_G)	C_D Knochen	C_D Muskel
2	64×64	12	0,14 % ; 0,72 %	0,991	k.A.

Tabelle 5.7: *Die eingestellte Auflösung, die Zahl der verwendeten Merkmale und der Dice-Coeffizient für die Klassen Knochen und Muskel.*

Trainings- und Generalisierungsfehler von $0,14\,\%$ bzw. $0,72\,\%$ liegen nun erwartungsgemäß unter denen des 3-Klassen-Falls. Mit ihnen läßt sich ein C_D von $0,991$ erzielen. Die zugehörige Segmentierung ist in Abb. 5.8 (d) abgebildet.
Die feste Auswahl des Selektors war in diesem Fall ungeeignet, da ein anderer Selektor – in diesem Fall ein „manueller" Selektor – deutlich bessere Ergebnisse erzielt. Die Verwendung der Blasensynthese könnte eine bessere Kombination hervorbingen[8], mit der auch der C_D auf über 0,992 steigen könnte.

5.4.1.2 Datensatz 2

In diesem Datensatz sind zwei Muskeln, ein Knochen und Hintergrund abgebildet (vgl. Abb. 5.9 (a)). Betrachtet man zunächst das 4-Klassen-Problem der Trennung von Knochen und der zwei Muskelgruppen zum Hintergrund ergab sich eine gewählte Fenstergröße von 64×64 Pixeln bei einem Trainings- bzw. Generalisierungsfehler von $0,25\,\%$ bzw. $1,15\,\%$. Die ermittelten Merkmale waren TMaxProb, TCorr, TSumAvg, TDiffVar, TIMCorr2, TClustShade, SMean,

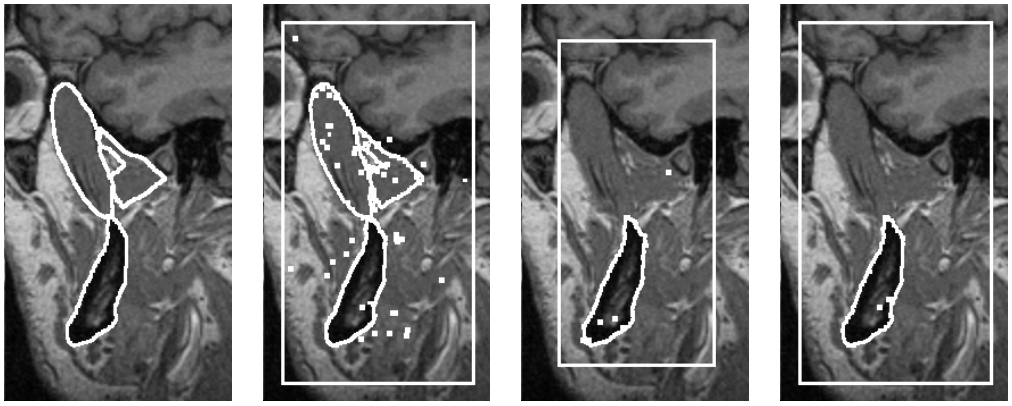
[8] Aufgrund der hohen Zahl an Observationen von ca. 50.000 pro Datensatz wird die Berechnungsdauer auf etwa 235 h geschätzt [46]

SMin, SMax, SModeGV und SMed.

Als 2-Klassen-Problem der Trennung von Knochen und Hintergrund (incl. der Muskeln) ergab sich eine Fenstergröße von 64×64 Pixeln, ein Trainings- bzw. Generalisierungsfehler von $0,12\,\%$ bzw. $0,56\,\%$ und die Merkmale TCorr, TSumAvg, TIMCorr1, TRlPerc, SMean, SMin, SKurt, SVarCoeff.

Klassen	Auflösung	Anzahl Merkmale	Fehler (E_T ; E_G)	C_D Knochen	$\varnothing C_D$ Muskeln
4	32×32	10	0,32 % ; 1,34 %	0,993	0,990
2	64×64	8	0,12 % ; 0,56 %	0,991	k.A.

Tabelle 5.8: *Die ermittelte Auflösung, die Zahl der verwendeten Merkmale und der Dice-Coeffizient für die Klassen Knochen und Muskel 1 & 2.*



(a) (b) (c) (d)

Abbildung 5.9: *Überlagerte Segmentierungen von Knochen (unten) und Kaumuskelgewebe (oben) über das Originalbild. (a) Manuelle Segmentierung. (b) Automatische Segmentierung. (c) Automatische Segmentierung mit Parametern des 2-Klassen-Falls. (d) Automatische Segmentierung mit Parametern aus (b) und adaptiertem Klassifikator.*

5.4.1.2.1 Fazit Für beide Probleme werden geeignete Parameter gefunden und sehr gute Segmentierungsergebnisse erzielt. Trainings- und Generalisierungsfehler sind im 2-Klassenfall um 0,2 % bis 0,78 % geringer. Ein besseres Segmentierungsergebnis wird jedoch nicht erreicht. Der Grund hierfür wird wieder im

Selektor gesucht, da die Anzahl der Merkmale und die resultierende Fenstergröße verändert wurden. Eine manuelle Anpassung der Auflösung und der Merkmale verbessert zwar Trainings- und Generalisierungsfehler deutlich, jedoch das Segmentierungsergebnis nicht meßbar:

Klassen	Auflösung	Anzahl Merkmale	Fehler (E_T ; E_G)	C_D Knochen	$\varnothing C_D$ Muskeln
2	32×32	10	0,09 % ; 0,38 %	0,991	k.A.

Tabelle 5.9: *Die eingestellte Auflösung, die Zahl der verwendeten Merkmale und der Dice-Coeffizient für die Klassen Knochen und Muskel 1 & 2.*

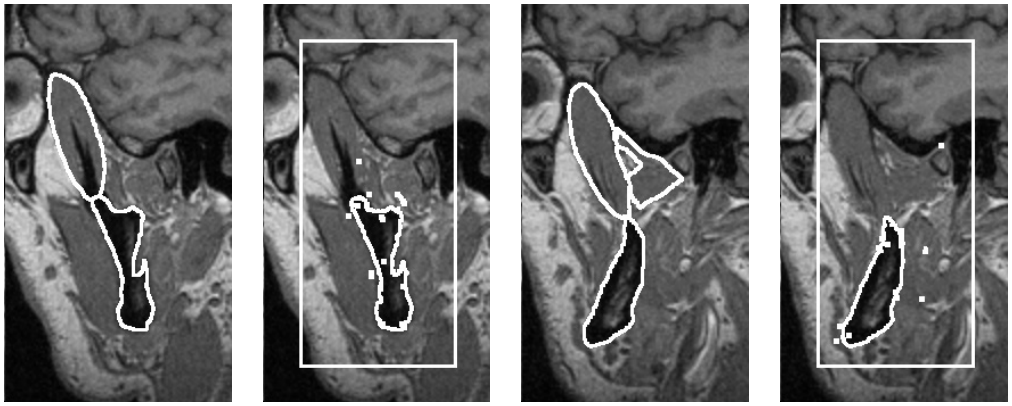
Trainings- und Generalisierungsfehler von $0,09\,\%$ bzw. $0,38\,\%$ liegen erwartungsgemäß unter denen des 4-Klassen-Falls. Mit ihnen läßt sich ein C_D von $0,991$ erzielen. Die zugehörige Segmentierung ist in Abb. 5.9 (d) abgebildet.
Die feste Auswahl des Selektors wirkte sich in diesem Fall nicht negativ auf die Segmentierungsgüte aus, obwohl der Klassifikationsfehler deutlich geringer wurde. Dies zeigt, daß der Klassifkationsfehler keine direkte Schlußfolgerung auf die Segmentierungsgüte zuläßt.

5.4.1.3 Gemeinsame Betrachtung

Neben der isolierten Betrachtung des jeweiligen Datensatzes wird auch untersucht, welche Ergebnisse bei einer inkrementellen Bearbeitung der Daten erzielt werden können. Dazu werden die Parameter des 2-Klassen-Falls von Datensatz 1 auf Datensatz 2 angewendet und die Segmentierungsqualität gemessen. Danach wird ein gemeinsamer Parametersatz für beide Bilder gemeinsam erstellt und wiederum die Segmentierungsqualität gemessen. Verwendet man die Parameter des ersten Datensatzes zur Segmentierung des zweiten, so sind die Ergebnisse sehr schlecht. Der Dice-Coeffizient liegt bei nur 0,525.
Trainiert man mit beiden manuellen Segmentierungen, ergeben sich Trainings- und Generalisierungsfehler von $0,22\,\%$ bzw. $0,83\,\%$. Wendet man die gefundenen Parameter auf beide Bilder an, so ergibt sich für Datensatz 1 ein C_D von 0,984 und für Datensatz 2 ein C_D von 0,993.

Auflösung	Anzahl Merkmale	Fehler (E_T ; E_G)	C_D Knochen (Bild1 ; Bild2)
64×64	7	0,22 % ; 0,83 %	0,984 ; 0,993

Tabelle 5.10: *Die ermittelte Auflösung, die Zahl der verwendeten Merkmale und der Dice-Coeffizient für die Klasse Knochen.*



(a) (b) (c) (d)

Abbildung 5.10: *Überlagerte Segmentierungen von Knochen (unten) und Kaumuskelgewebe (oben) über das Originalbild. (a) & (b) Manuelle und automatische Segmentierung von Datensatz 1. (c) & (d) Manuelle und automatische Segmentierung von Datensatz 2.*

5.4.1.4 Fazit Kaumuskeln

Das System ist in der Lage, Parametersätze zu finden, mit denen die manuellen Segmentierungen in hohem Maße reproduziert werden können. Das Ziel des Experiments wird damit erreicht.
Für Datensatz 1 gilt, daß die Qualität der automatisch ermittelten Parameter im 3-Klassen-Fall höher als bei nur zwei betrachteten Klassen ist. Für den 2-Klassen-Fall wird eigentlich erwartet, daß die Klassifikationsergebnisse besser als im 3-Klassen-Fall werden, da zwei Klassen zusammengefaßt werden. Der verwendete Selektionsalgorithmus ist nicht optimal. Die manuelle Auswahl der Parameter bringt neben deutlichen Verbesserungen des Klassifikationsergebnisses auch eine höhere Segmentierungsgüte.
Bei Datensatz 2 zeigt sich ein ähnliches Bild. Im 2-Klassen-Fall werden uner-

warteterweise andere Parameter gefunden als im 4-Klassen-Fall – obwohl drei Klassen zusammengefaßt werden. Hierbei sind die Unterschiede der Segmentierungsergebnisse jedoch nicht meßbar. Ein besseres Klassifikationsubsystem kann mit den zur Verfügung stehenden Algorithmen nicht gefunden werden.
Die Anwendung der Parameter von Datensatz 1 auf Datensatz 2 ergibt eine stark verminderte Segmentierungsgüte von $C_D = 0,525$ für den Knochen. Ein einzelnes MRT-Bild ist nicht repräsentativ für die Problemstellung. Wird das zweite MRT-Bild in das Training miteinbezogen, kann eine mittlere Segmentierungsqualität von $C_D = 0,989$ erreicht werden. In Anlehnung an die bereits betrachteten Datensätze (vgl. Abschnitte 5.1 und 5.3) wird erwartet, daß sich die Ergebnisse bei größeren Trainings- und Testdaten ähnlich gut verhalten.

5.4.2 Diverse Bilder

Die im folgenden Abschnitt verwendeten Bilder stammen aus der Literatur und dem Internet [21, 22, 54, 62]. Jedes stellt für sich ein Segmentierungs-/Mustererkennungsproblem dar. Zwischen den Bildern besteht kein Zusammenhang.
Für jedes Bild wird wenigstens eine ROI pro Klasse manuell definiert. Die Anzahl der Observationen beträgt (wenn möglich) 100 pro Klasse. Verfahren wird wie in Kapitel 4 beschrieben mit einer bidirektionalen BestFirst-Suche und dem J48-Klassifikator. Für die Bilder existieren keine Referenzsegmentierungen, daher kann die Segmentierungsgüte nur visuell evaluiert werden. Die Abbildungen 5.11 – 5.15 zeigen jeweils das Originalbild mit den manuell definierten ROIs für Objekt (weiß) und Hintergrund (schwarz) sowie der automatischen Segmentierung.
Die Segmentierungsergebnisse reichen rein visuell betrachtet von sehr gut (siehe Abb. 5.11 – 5.13) bis mangelhaft (siehe Abb. 5.14 – 5.15). Stellenweise zeigen sich starke Artefakte etwa in Form kompletter Fehlsegmentierungen.

5.4.2.1 Fazit

Es zeigt sich, daß das System mit wenig Aufwand Parameter für sehr gute Segmentierungen, insbesondere für synthetische Texturbilder bestimmen kann. Gleichzeitig gibt es Bilder, bei denen keine sinnvollen Segmentierungsergebnisse erzielt werden können. Aufgrund fehlender Referenzsegmentierungen können kei-

ne quantitativen Vergleichsmaße berechnet werden.

Segmentierungsprobleme treten besonders bei Bildbereichen auf, in denen die Objektgrenzen auch mit dem Auge schwer zu erkennen sind, aber auch dann, wenn sich Objekt und Hintergrund sehr ähnlich sehen. Es wird vermutet, daß die zur Verfügung stehenden Merkmale oder ihre Kombination nicht ausreichen, um das Problem angemessen zu beschreiben [35]. Bessere Merkmale könnten hier Abhilfe schaffen; genauso könnte jedoch eine bessere Selektor-Klassifikator-Kombination zu besseren Ergebnissen führen. Inwiefern gefundene Parameter für ähnliche Bilder nutzbar sind, kann aufgrund der beschränkten Menge von Daten nicht untersucht werden. Im Hinblick auf die vorangegangenen Untersuchungen kann angenommen werden, daß das System dazu genutzt werden kann, komplette Bilddatensätze mit ähnlichem Inhalt wie die Untersuchten inkrementell zu lernen und im Mittel gut zu segmentieren.

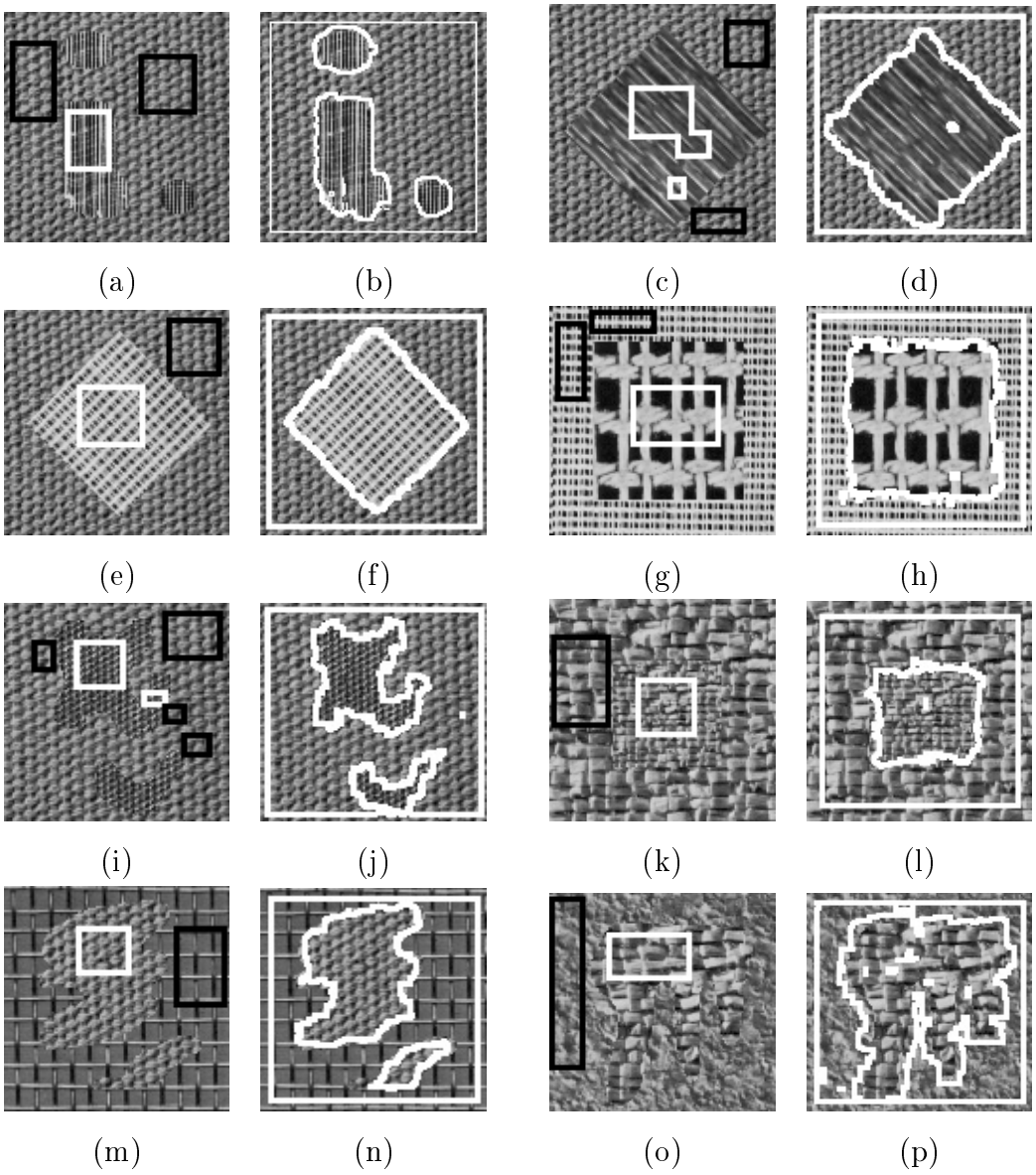


Abbildung 5.11: *Beispiele für gute Segmentierungen synthetischer Objekte. (Paarweise abgebildet) Jeweils links: Die synthetischen Texturbilder mit ROIs für Objekt (weiß) und Hintergrund (schwarz). Jeweils rechts: Die Grenzen der automatischen Segmentierungen (weiß).*

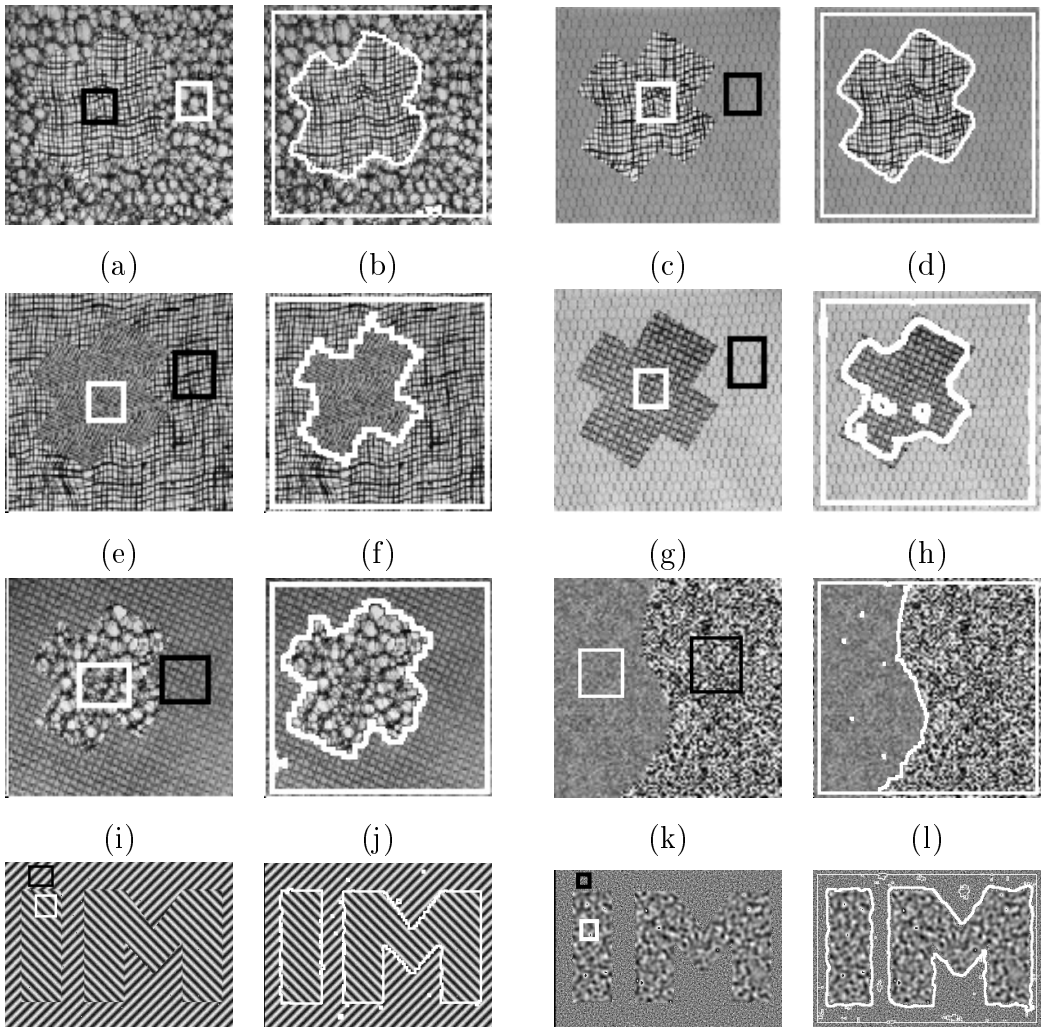


Abbildung 5.12: *Beispiele für gute Segmentierungen synthetischer Objekte. (Paarweise abgebildet) Jeweils links: Die synthetischen Texturbilder mit ROIs für Objekt (weiß) und Hintergrund (schwarz). Jeweils rechts: Die Grenzen der automatischen Segmentierungen (weiß).*

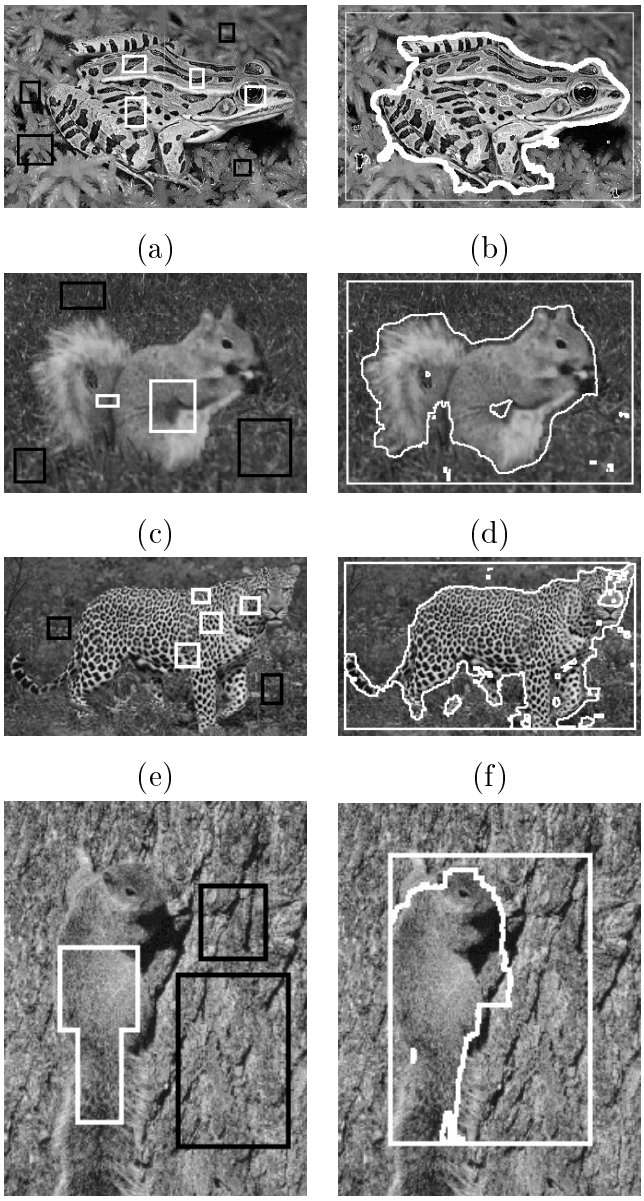
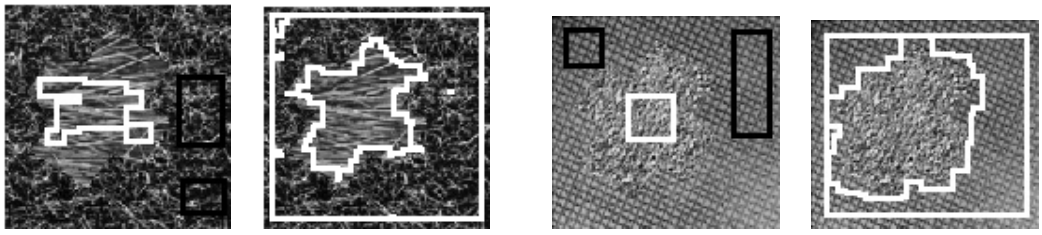
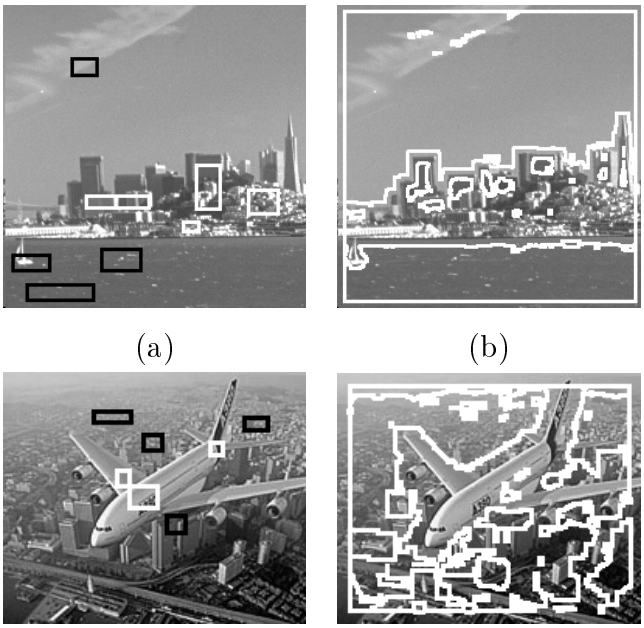


Abbildung 5.13: *Beispiele für gute Segmentierungen natürlicher Objekte. (Von oben nach unten) Links: Das ursprüngliche Bild eines Frosches auf Moos, eines Eichhörnchens auf einer Wiese, eines Leopard im Dschungel und eines getarnten Eichhörnchens am Baum. ROIs für Objekt (weiß) und Hintergrund (schwarz). Rechts: Die Grenzen der automatischen Segmentierungen (weiß).*



(a) (b) (c) (d)

Abbildung 5.14: *Beispiele für schlechte Segmentierungen synthetischer Objekte. (Paarweise abgebildet) Jeweils links: Die synthetischen Texturbilder mit ROIs für Objekt (weiß) und Hintergrund (schwarz). Jeweils rechts: Die Grenzen der automatischen Segmentierungen (weiß).*



(a) (b)

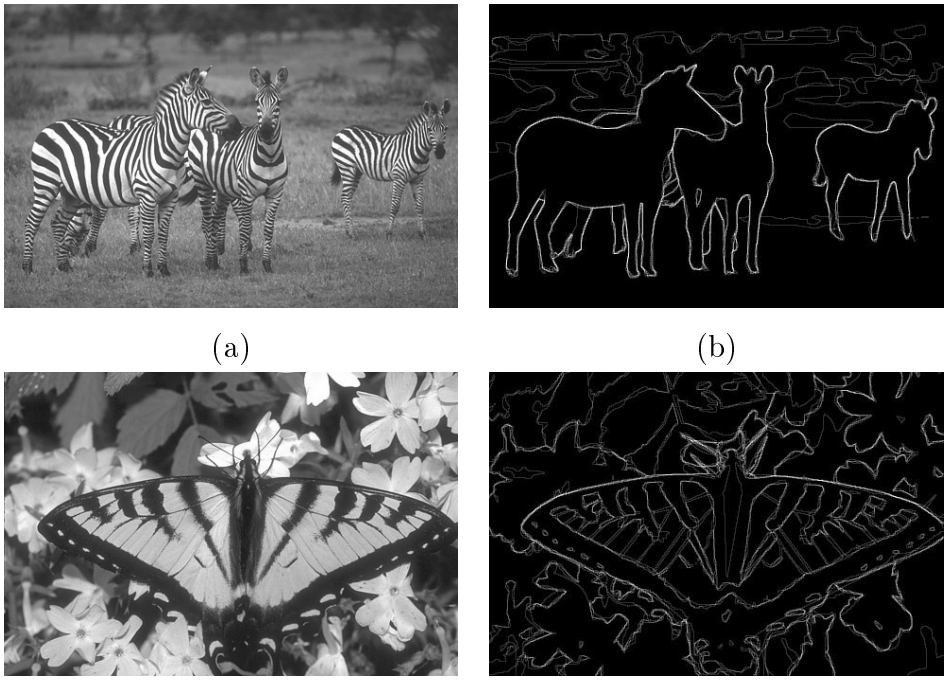
(c) (d)

Abbildung 5.15: *Beispiele für schlechte Segmentierungen natürlicher Objekte. (Von oben nach unten) Links: Das ursprüngliche Bild einer Stadt am Meer und eines Flugzeugs über einer Großstadt. ROIs für Objekt (weiß) und Hintergrund (schwarz). Rechts: Die Grenzen der automatischen Segmentierungen (weiß).*

5.4.3 Die Berkeley Bilddatenbank

Diese Datenbank besteht aus etwa 12.000 manuellen Segmentierungen, die von 30 Menschen für 1.000 Bilder des Corel-Datensatzes durchgeführt wurden. Diese Segmentierungen wurden als empirische und wissenschaftliche Grundlage für die Forschung an Segmentierung und Detektion von Regionenkanten [54] erstellt. Zur Hälfte waren die Bilder farbig, zur anderen Hälfte in Graustufen. Frei verfügbar waren 300 Bilder, davon sind 200 zum Training und 100 zum Test vorgesehen.
Jedes Bild der Datenbank wurde von verschiedenen Personen segmentiert, weshalb die Kanten der Regionen – und auch die Regionen selbst – nicht einheitlich sind. Daher wurden die Kanten als Wahrscheinlichkeiten interpretiert: Je mehr Menschen dieselbe Kante zwischen zwei Regionen zeichneten, desto wahrscheinlicher war sie eine Kante. Trotzdem war jede eingezeichnete Kante bzw. Region gültig.
Die Datenbank ist vornehmlich dazu erstellt worden, kantenorientierte Ansätze miteinander zu vergleichen, bei denen nicht nur Objektkanten interessant sind. Vielmehr sollen alle Kanten, die von menschlichen Segmentierern markiert werden, auch automatisch auffindbar sein. Trotzdem werden Teile der vorliegenden Datenbank verwendet, weil sie eine große Vielzahl natürlicher Szenen abdecken und damit Aufschluß auf die vielfältigen Anwendungsmöglichkeiten der vorgestellten Methodik liefern könnten.

Die von den Entwickler der Datenbank vorgeschlagene Bewertung eines Segmentierungsalgorithmus ging davon aus, daß der Algorithmus ebenfalls Wahrscheinlichkeiten für jedes Pixel ausgibt, zu einer Kante zu gehören. Hierin lag die Schwierigkeit, die Ergebnisse des in dieser Arbeit entwickelten Algorithmus mit der Datenbank zu vergleichen. Das System produzierte mit den meisten Klassifikationssystemen eine binäre Kante, d.h. ein Pixel gehörte entweder zu einer Kante oder nicht. Das Bewertungsmaß der Datenbank konnte daher nicht zum Einsatz kommen.
Es werden nur 2-Klassenprobleme betrachtet. Die menschlichen Segmentierungen wurden als Entscheidungshilfe verwendet, welches Objekt das System vordergründig segmentieren sollte. Mit einfachen ROIs wurden Teile der entsprechenden Objekte ausgewählt und seitens des System eine Segmentierung vorgeschlagen.



(a) (b)

(c) (d)

Abbildung 5.16: *(a) Das Originalbild mehrerer Zebras. (b) Die menschlichen Segmentierungen. Dieses Bild konnte nicht verwendet werden, weil das Objekt Zebra nicht einzeln vor dem Hintergrund erscheint. Die Flächen mit Zebramuster konnten als Ganzes sehr gut identifiziert werden (c) Das Originalbild eines Schmetterlings. (d) Die menschlichen Segmentierungen. Dieses Bild konnte nicht verwendet werden, weil das Objekt Schmetterling übersegmentiert war. Je höher der Grauwert einer Kante in (b) und (d), von desto mehr Segmentierern wurde sie eingezeichnet.*

Zur Bewertung der Segmentierungsgüte wurden die manuellen segmentierten Regionen überlagert und der Dice-Coeffizient mit der automatischen Segmentierung berechnet. Es konnten nur solche Bilder berücksichtigt werden, bei denen das eigentliche Objekt als Ganzes und nicht von allen Menschen zerstückelt segmentiert war (vgl. Abb. 5.16(d)). Außerdem sollten die Objekte voneinander getrennt sein, da der Segmentierer nicht darauf ausgelegt ist, aneinander grenzende gleiche Objekte zu trennen (vgl. Abb. 5.16(b)). Weiterhin sollte, möglichst viel Abstand der Objekte zum Bildrand gegeben sein, anderenfalls können Objektteile aufgrund der Fensterung fehlen (vgl. Abb. 5.16(d)). Es können daher nur 35 Bilder der Datenbank verwendet werden. Die Abbildungen 5.17 und 5.18 zeigen Beispiele von

Bildern aus der Datenbank sowie deren manuelle und automatische Segmentierungen. In Tabelle 5.11 ist eine statistische Auswertung der Ergebnisse zu finden.

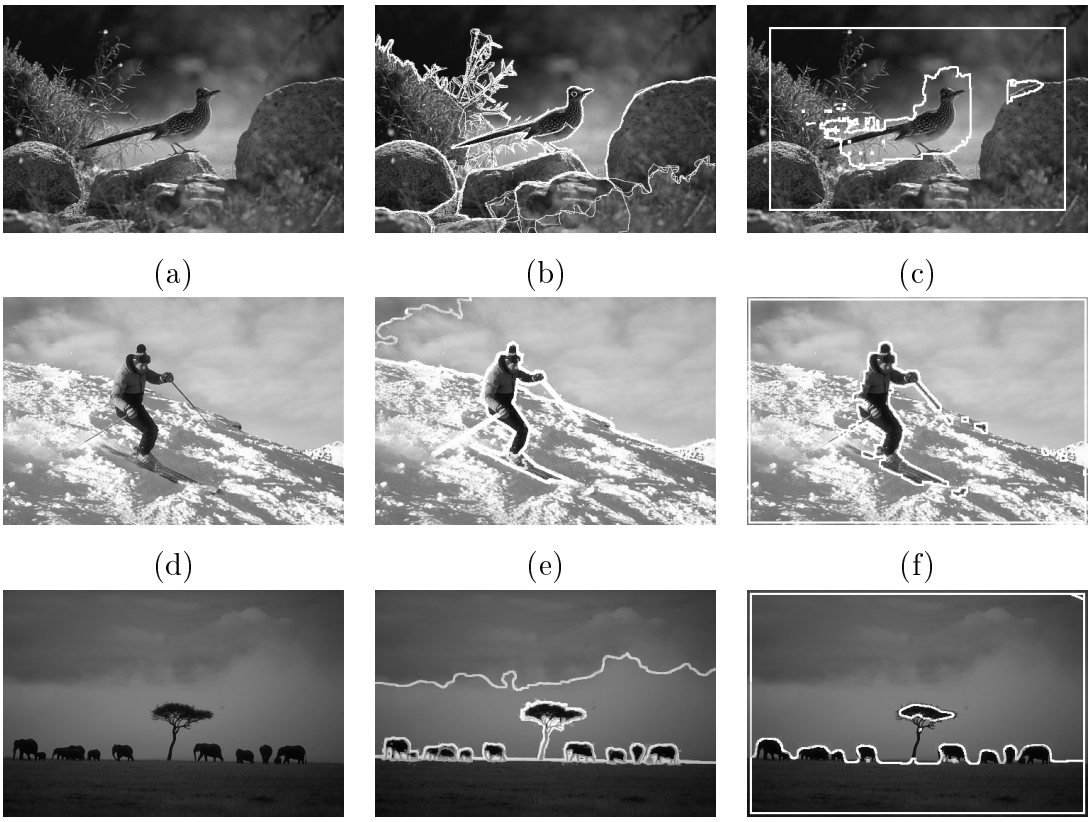


Abbildung 5.17: *Trainingsdaten. Gezeigt werden diejenigen Bilder, die der minimalen (erste Zeile), mittleren (zweite Zeile) und maximalen (dritte Zeile) automatischen Segmentierungsgüte entsprechen. (a),(d),(g) Originalbilder. (b),(e),(h) Dem Bild überlagerte menschliche Segmentierungen. (c),(f),(i) Dem Bild überlagerte automatische Segmentierungen.*

Die mittlere Segmentierungsgüte des Gesamtdatensatzes liegt bei $0,818$ bei einer Streuung von $0,105$ und einer Schätzgüte von $0,018$. Auch visuell betrachtet sind die Segmentierungen im Mittel gut bis sehr gut. Einige Bilder können nur mittel bis sehr schlecht segmentiert werden. Zwischen dem Trainings- und dem Testset gibt es Unterschiede (vgl. Tab. 5.11). Der Mittelwert des Trainingssets ist mit $0,823 \pm 0,02$ nicht signifikant höher als der des Testsets mit $0,800 \pm 0,03$. Minimale und maximale Segmentierungsqualität werden nur im Trainingsset erreicht, dafür

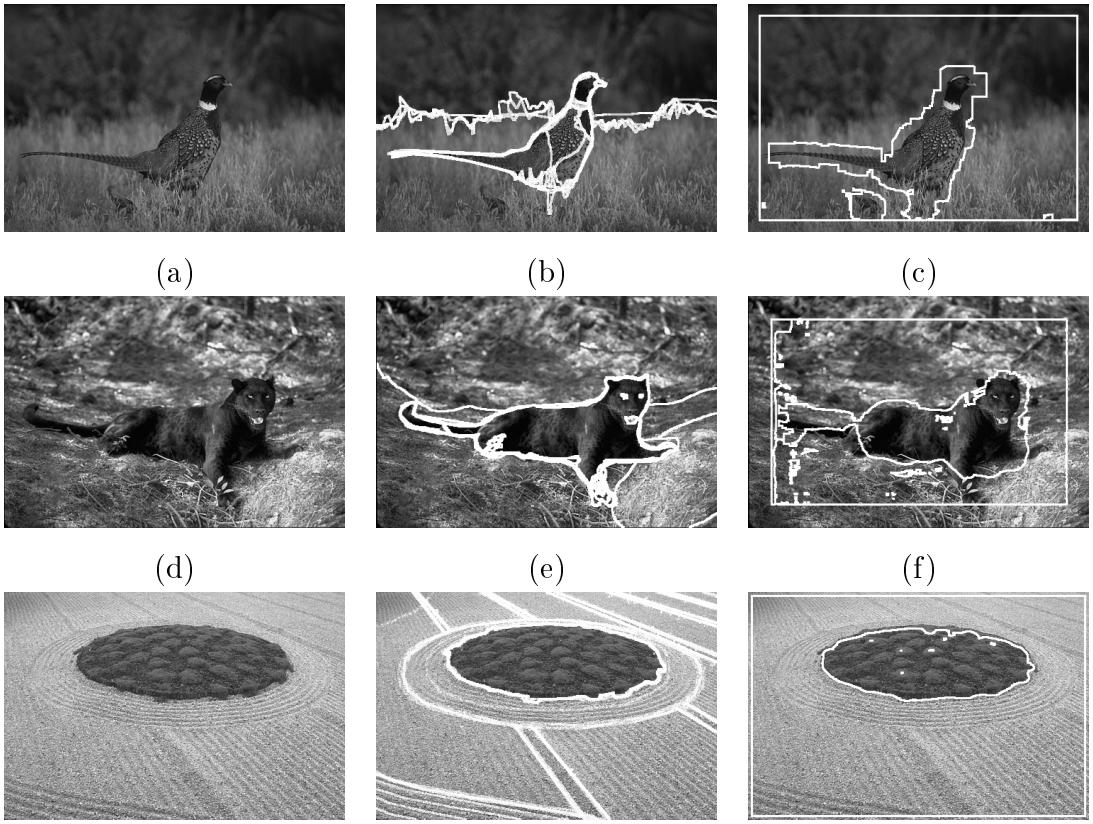


Abbildung 5.18: *Testdaten. Gezeigt werden diejenigen Bilder, die der mittleren (erste Zeile), minimalen (zweite Zeile) und maximalen (dritte Zeile) automatischen Segmentierungsgüte entsprechen. (a),(d),(g) Originalbilder. (b),(e),(h) Dem Bild überlagerte menschliche Segmentierungen. (c),(f),(i) Dem Bild überlagerte automatische Segmentierungen.*

ist die Spannweite im Testset halb so groß wie im Trainingsset.

5.4.3.1 Diskussion Berkeley

An 36 Bildern der Berkeley Datenbank konnte gezeigt werden, daß mit der vorgestellten Methodik im Mittel zufriedenstellende Segmentierungsergebnisse von $C_D = 0,8227$ erzielt werden können. Die hohe Spannweite von 0,5128 ist ein deutlicher Hinweis darauf, daß die verwendeten Merkmale oder Klassifikationssubsysteme nicht immer gut geeignet waren, die Segmentierungsprobleme zu beschreiben. Bessere Merkmale oder Selektor-Klassifikator-Kombinationen könnten

	Gesamt	Trainingsset	Testset
Mittelwert $\bar{x}$	0,8177	0,8227	0,8001
Streuung s	0,1049	0,1093	0,0921
Minimum	0,4531	0,4531	0,6652
Maximum	0,9659	0,9659	0,9482
Spannweite	0,5128	0,5128	0,2830
Schätzfehler	0,0175	0,0207	0,0325
Anzahl Bilder	36	28	8

Tabelle 5.11: *Statistik der Segmentierungsergebnisse für den Gesamtdatensatz, das Trainings- und Testset.*

hier Abhilfe schaffen [35].
Die Beschränkung der Methodik auf primär 2-Klassen-Probleme hat zur Folge, daß nicht alle Bilder verwendet bzw. zur Evaluierung herangezogen werden.

5.5 Zusammenfassung

In diesem Kapitel wurde anhand von künstlichen und natürlichen Daten empirisch gezeigt, welche Segmentierungsergebnisse mit der vorgestellten Methodik erreicht werden können.
Mit zwei künstlichen Datensätzen wurden sowohl die Bestimmung geeigneter Parameter als auch das inkrementelle Lernen für die Segmentierung überprüft. Die positiven Ergebnisse ließen auf vielversprechende Segmentierungsergebnisse für natürliche Bilder hoffen. Diese Hoffnung hat sich in vielen Fällen erfüllt, wie die Beispiele der Detektion von Spiculierten Massen und zahlreiche Segmentierungprobleme aus der Literatur veranschaulichen. In einigen Fällen hingegen konnten keine guten Ergebnisse erzielt werden. Die Gründe hierfür werden in der für die Beispiele verwendeten fest eingestellten Selektor-Klassifikator-Kombination vermutet. Der Einfluß der Selektion ließ sich am Beispiel der Kaumuskeln (Datensatz 1) nachweisen. Eine manuelle Korrektur der Merkmalsselektion resultierte in deutlich verbesserten Ergebnissen. Die mit der Blasensynthese ermittelten Klassifikationssubsysteme für zwei synthetische Datensätze zeigen, daß dieses Problem

vermeidbar ist. Allerdings können sich die begrenzte Zahl und Mächtigkeit der verwendeten Merkmale, Selektoren und Klassifikatoren nachteilig auf die Ergebnisse auswirken.
Die Güte der Segmentierung ist stark von den benutzerdefinierten ROIs abhängig. Daher sollten die Inhalte der ROIs repräsentativ für die zu trennenden Bildinhalte sein.

Kapitel 6

Diskussion und Ausblick

Das Ziel dieser Arbeit war es, dem menschlichen Experten ein System zur Verfügung zu stellen, welches er mit wenig Aufwand auf die selbsttätige und zuverlässige Erkennung von Objekten auf digitalen Bildern trainieren kann.
Es wurde ein interaktiv lernendes System entwickelt und implementiert, das vornehmlich aus einem adaptiven Segmentierungsalgorithmus besteht. Es wählt automatisch geeignete Parameter aus, so daß die Segmentierungsergebnisse den Anforderungen des Experten gerecht werden. Mit dem vorliegenden System wird das Ziel erreicht. Dies konnte an zwei künstlichen und drei natürlichen Datensätzen sowie zahlreichen weiteren Einzelbildern gezeigt werden.

6.1 Die adaptive Segmentierung

Die Hauptannahme bestand darin, daß gute Mustererkennungsergebnisse auf einer guten Segmentierung beruhen. Daher beschränkt sich diese Arbeit auf die Entwicklung einer adaptiven Segmentierung. Weiterhin wurde angenommen, daß Objekte und Bildhintergrund in einer anwendungsspezifischen Form homogen sind und sich diese Homogenität durch entsprechende Parameter beschreiben läßt. Um die Adaptivität zu erreichen und die Beschränkungen bekannter Methoden zu überwinden, wurde ein automatisch generiertes Klassifikationssubsystem zur Segmentierung und ein Feedback zur Bildverarbeitungskette hinzugefügt und ein frei konfigurierbares Parameterset beliebiger Größe verwendet. Das Parameterset besteht aus lokalen Merkmalen und einer geeigneten Kombination von Merkmals-

selektor und Klassifikator.

Zur Zeit sind etwa 45 verschiedene lokale Merkmale implementiert, die in den meisten Fällen ausreichend gute Ergebnisse liefern. Mit nur 45 Merkmalen können keine guten Segmentierungen für jedes beliebige Problem garantiert werden. Bei schlechten Ergebnissen sind möglicherweise weitere Merkmale zur Verbesserung nötig [35].

Mit der vorgestellten Methodik kann jedes beliebige lokale Merkmal verwendet werden. Die Merkmalsselektions- und Klassifikationsalgorithmen waren für viele der vorgestellten Probleme fest auf eine bestimmte Kombination eingestellt. In der verwendeten Software ICE steht jedoch ein breites Spektrum von zur Zeit etwa 100 Selektoren und 50 Klassifikatoren zur Verfügung [45]. Eine Methode, die eine gute Kombination aus beiden in sinnvoller Zeit auffindet, wurde von Müller et al. [46, 50] vorgestellt und kann mit nur geringem Implementierungsaufwand in das System eingefügt werden.

Momentan liegt die Zeit, die zur Berechnung der Parameter aus 200 Observationen benötigt wird, d.h. die Extraktion der Merkmale, ihre Selektion und Klassifikation, bei bis zu 10 Minuten. Sie ist abhängig von der Zahl der Observationen und der Komplexität des Klassifikators. Die Zeit für die Segmentierung eines Bildes beträgt bis zu 20 Minuten[1]. Die Verteilung der Berechnungen auf einem GRID könnte einen deutlichen Geschwindigkeitsvorteil erzielen.

Das System wurde auf künstlichen und natürlichen Datensätzen unterschiedlicher Inhalte getestet. Die Segmentierungsqualität ist bei rein visueller Betrachtung in vielen Fällen sehr gut. Da die perfekte Segmentierung meist nur von synthetischen Bildern bekannt ist, wird der geschätzte Generalisierungsfehler als Evaluierungskriterium für die Zuordnung einzelner Pixel zu den verschiedenen Regionen verwendet. Der geschätzte Generalisierungsfehler ist das einzig gute Kriterium, das Klassifikationssubsystem zu bewerten, während die Güte der Segmentierung damit nicht eindeutig zu bestimmen ist. Daher wird in den Fällen, in denen die Segmentierung bekannt ist, der Dice-Coeffizient vorgezogen.

Als Endergebnis liegt eine JAVA-Implementierung der vorgestellten Methodik vor. Die Implementierung folgt dem komponentenbasierten Konzept von ICE [46]

[1] Für ein 400×400 Pixel großes Bild und 40 zu berechnende Merkmalen auf einem Standard-PC mit 3 GHz und 1 GB RAM.

und ist vollständig in ICE eingebettet. Dadurch können neue in ICE integrierte Selektions- und Klassifikationsalgorithmen von der vorgestellten Methodik ohne zusätzlichen Aufwand genutzt werden.
Eine einfache Version der Software könnte als „Webservice" über das Internet zugänglich gemacht werden. Mit diesem könnten Benutzer weltweit eigene Bilder testen, ohne die Methodik reimplementieren zu müssen.

6.2 Einschränkungen

Für den menschlichen Experten wurde ein komfortabler Weg aufgezeigt, durch das interaktive Präsentieren von Beispielen und Gegenbeispielen ein Trainingsset plausibler Größe und Repräsentativität zu erstellen und das System mit einer gewissen Fehlerrate auf bestimmte Anwendungen zu adaptieren. Die Performance des Systems kann iterativ verbessert werden, indem weitere Beispiele zur Verfügung gestellt werden.
Das vorgestellte Verfahren funktioniert nur dann, wenn die innere Homogenität der Objekte mit den zur Verfügung stehenden Merkmalen modelliert werden kann. Dabei ist offensichtlich, daß ein Training des Klassifikationssubsystems auf die innere Homogenität der Regionen zur Folge haben kann, daß die erkannten Grenzen nicht zwingend die wahren Regionsgrenzen widerspiegeln müssen. Dies kann verhindert werden, indem explizit die Regionsgrenzen trainiert werden, z.B. durch die Auswahl aneinander grenzender Beispiele aus verschiedenen Klassen.
Auch werden die Merkmale derzeit nicht rotationsinvariant berechnet. Dies kann erreicht werden, indem die quadratischen Fenster durch Kreise ersetzt werden. Eine Kombination aller Berechnungsformen könnte weitere Verbesserungen nach sich ziehen, da auf diese Weise mehr Merkmale zur Verfügung stehen, wodurch sich die Wahrscheinlichkeit erhöht, ein geeignetes Merkmal zu finden. Der Rechenaufwand stiege dadurch jedoch stark an.
Die letztendliche Fenstergröße wird auf genau eine Größe beschränkt. Die Kombination verschiedener Merkmale in verschiedenen Fenstergrößen[2] könnte die Ergebnisse weiter verbessern. Der Rechenaufwand steigt dadurch ebenfalls stark an.

[2] z.B. der Mittelwert in einem 3×3 und die Kovarianz in einem 7×7 Pixel großen Fenster

Zur Zeit werden Merkmale nur von Grauwertbildern berechnet. Die Methodik ist darauf angelegt, mit beliebigen Merkmalen umzugehen, daher können auch Merkmale von Farbbildern verwendet werden.
Die Segmentierungsergebnisse sind sehr stark von den benutzerdefinierten ROIs abhängig. Der Benutzer muß ROIs definieren, die repräsentativ für die zu trennenden Bereiche sind. Anderenfalls können die Segmentierungsergebnisse unzureichend sein. Insbesondere sollte der Benutzer davon Abstand nehmen, explizite Fehlinformationen in das System einzubringen. Die Auswirkungen wären in jedem Fall hohe Klassifikationsfehler. Prinzipiell ist es möglich, den Benutzer darauf hinzuweisen, sobald er eine Region des Bildes einmal als Objekt und ein anderes Mal als Hintergrund markiert. Dies wurde jedoch nicht implementiert, da immer davon ausgegangen wird, daß der Experte an der Lösung des vorliegenden Problems interessiert ist und explizite Fehleingaben daher vermeidet.
In dieser Arbeit wurden nur zweidimensionale Ergebnisse gezeigt. Die Methodik funktioniert auch im Dreidimensionalen, indem statt eines Rechtecks ein Kubus als ROI definiert wird.
Das entwickelte System verarbeitet die Bilder iterativ. Änderungen an der Segmentierung bereits verarbeiteter Bilder, die durch noch zu bearbeitende Bilder verursacht werden und einen Korrektureingriff erfordern könnten, werden momentan noch nicht berücksichtigt. Hier sei darauf verwiesen, daß eine gute Generalisierungsfähigkeit nicht bedeutet, daß jedes Bild gut segmentiert ist, sondern daß die Segmentierungsqualität im Mittel über den gesamten Datensatz gut ist (*average best fit*). Dabei ist schwer absehbar, nach wievielen Beispielen der Lernvorgang eingestellt werden kann und die Verteilung der gesammelten Observationen repräsentativ genug ist, um als Grundgesamtheit zu fungieren. Nimmt man an, daß die Grundgesamtheit normalverteilt ist, so kann mit einer statistischen Konfidenzanalyse (vgl. Formel 4.3) die Güte der Schätzung berechnet werden.
Die parallele Verarbeitung der Bilder ist das entscheidende Mittel, um die Verarbeitungsgeschwindigkeit zu erhöhen. Die Methode ist wegen der eventorientierten Bearbeitung sehr leicht parallelisierbar.

6.2.1 Einschränkung durch die Implementierung

Die Implementierung der vorgestellten Methodik in der Komponentensoftware ICE ist nicht nur von Vorteilen gekennzeichnet. Da eine Komponentensoftware darauf basiert, daß eine Komponente genau eine Funktionalität hat, sind komplexe Vorgänge schwierig zu handhaben. Als Beispiel sei hierfür die interaktive Visualisierung genannt, bei der ein Bild oder Volumen visualisiert und dabei editiert wird. In einer Komponentensoftware ist dies strikt getrennt: Eine Komponente visualisiert, eine andere gestattet das Editieren. Damit ist die Kommunikation zwischen zwei Komponenten nur über Datenpfade möglich. Eine direkte Kommunikation gibt es nicht.
Diese Hindernisse wurden in der vorliegenden Implementierung ignoriert und teilweise mehrere zusammengehörende Funktionalitäten in einer Komponente kombiniert.
Weitere Probleme ergeben sich durch den erhöhten Speicherbedarf einer Komponentensoftware, in der jede Komponente ihre eigenen Datenstrukturen bereit hält. Große Bilder liegen somit als Kopien einmal je Komponente vor. Insbesondere bei Volumina kann es vorkommen, daß der verfügbare Speicher schnell ausgenutzt ist. Dies kann zu Lasten der Rechnerperformance gehen.
Ein möglicher Kritikpunkt ist die lange Rechenzeit der Merkmalsextraktoren. Eine parallele Extraktion der Merkmale etwa in einem GRID könnte die Rechenzeit erheblich verringern. Einzig die Erstellung eines Klassifikators ist bisher nicht parallelisierbar.
Das inkrementelle Erweitern des jeweiligen Klassifikators ist derzeit noch sehr unhandlich, da nach jeder Änderung ein neuer Trainingsdatensatz vorhanden ist, für den ein neuer Klassifikator erstellt werden muss. Manche Klassifikatoren, z.B. Bayes'sche, könnten mit *Codebook Vektoren* [63] einfacher aktualisiert werden.

6.3 Schlußfolgerung

Die in dieser Arbeit entwickelte Methodik gestattet dem Experten das interaktive Training eines Systems auf bestimmte Mustererkennungsprobleme.
Bislang werden für Mustererkennungsprobleme, insbesondere für die benötigte

Segmentierung, stets neue Algorithmen entwickelt. Diese Algorithmen werden speziell für ein Segmentierungsproblem entwickelt. Ihre Parameter sind meist manuell und subjektiv ausgewählt worden. Spätere Anpassungen sind meist unmöglich. Mit der vorliegenden Methodik werden diese Probleme umgangen. Dies wird durch eine flexibles, lernendes System mit automatischer, problembezogener Parameterwahl erreicht.

Zu diesem Zweck bestimmt das System automatisch geeignete Parameter zur Lösung des Problems, insbesondere für einen lernenden Segmentierungsschritt. Der Experte kann jederzeit mit seinem Wissen Einfluß auf das System nehmen und den Lernvorgang nach seinen Wünschen beeinflussen.

Für den Fall, daß ein trainiertes System zukünftigen Anforderungen nicht genügt, kann das Training jederzeit wieder aufgenommen werden, um neues Expertenwissen zu integrieren. Die Qualität der mit der Methodik erzielten Ergebnisse ist subjektiv und objektiv bewertet in den meisten Fällen sehr hoch, obwohl die Methode noch nicht optimiert wurde (s. 6.2).

Abschließend ist zu bemerken, daß mit der vorgestellten Methodik automatisch gute Parameter gefunden werden können, wenn das interaktive Training akkurat erfolgt und keine widersprüchliche Information eingebracht wird. Solange also eine geeignete Kombination von Merkmalen, Selektions- und Klassifikationsalgorithmus existiert, kann die menschliche Segmentierung mit objektiven Mitteln mittels der vorgestellten Methodik automatisch nachgebildet werden.

Literaturverzeichnis

[1] Mona Sharma. Performance Evaluation of Image Segmentation and Texture Extraction Methods in Scene Analysis. Master's thesis, University of Exeter, 2001.

[2] D. Comaniciu, P. Meer, D. Foran. Image Guided Decision Support System for Pathology. *Machine Vision and Applications*, 11(4):213–224, 2000.

[3] Oliver Dehning. Wissensbasierte Inspektion industrieller Objekte. Technical report, Institut für Nachrichtentechnik und Informationsverarbeitung, Universität Hannover, 1995.

[4] K.-M. Lee, W.N. Street. Model-based Detection, Segmentation and Classification for Image Analysis using On-line Shape Learning. *Machine Vision and Applications*, 13:222–233, 2003.

[5] T. Matsuyama. Expert Systems for Image Processing: Knowledge-Based Composition of Image Analysis Processes. *Computer Vision, Graphics, and Image Processing*, 48:22–49, 1989.

[6] A.K. Jain, R.P.W. Duin, J. Mao. Statistical Pattern Recognition: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), 2000.

[7] B. Bhanu, J. Peng. Adaptive Integrated Image Segmentation and Object Recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 30(4):427–441, 2000.

[8] M. Beller, R. Stotzka, T. O. Müller. Application of an interactive feature-driven segmentation. In *Biomedizinische Technik*, pages 210–211, 2004.

[9] M. Hastenteufel, C. Cárdenas, Ch. Giess, G. Glombitza, P. Hassenpflug, H.-P. Meinzer. Evaluierung von interaktiven, texturanalytischen Segmentierungsverfahren. In *Bildverarbeitung für die Medizin 2004*, pages 232–236, 2001.

[10] L. Kuhnert. Texturbasierte Segmentierung. Technical report, Universität Karlsruhe, 2003.

[11] Pavel Paclík, Robert P.W. Duin, Geert M.P. van Kempen, and Reinhard Kohlus. Supervised segmentation of textures in backscatter images. In *Proceedings 16th International Conference on Pattern Recognition*, pages 490–493, 2002. http://citeseer.nj.nec.com/539598.html.

[12] S.K. Warfield, J. Dengler, J. Zaers et al. Automatic identification of gray matter structures from mri to improve the segmentation of white matter lesions. In *Journal of Image Guided Surgery*, pages 326–338, 1995.

[13] Bruce Wooley and George Brannon Smith. Region-growing techniques based on texture for provincing the ocean floor. In *ACM Southeast Regional Conference*, pages 99–103, 1998. http://citeseer.nj.nec.com/312814.html.

[14] P. Perner. An Architecture for a CBR image segmentation system. *Artificial Intelligence, Engineering Applications of Artificial Intelligence*, 12(6):749–759, 1999.

[15] Christian Küblbeck. *Automatische und optimierte Konfiguration von Texturanalysesystemen*. PhD thesis, Universität Erlangen, 1999.

[16] Aldo von Wangenheim. Cyclops - Expert Shell System for the Development of Applications in the Area of Medical Image Analysis. Technical report, Universidade Federal de Santa Catarina, 1999.

[17] Heiko Münkel. Technischer Arbeitsbericht für das DFG-Projekt LI 279/21. Technical report, Universität Hannover, 1999.

[18] Frank Baumbach. Automatische Adaption regionenorientierter Segmentierungsverfahren. Master's thesis, Universität Hannover, 1998.

[19] Th. Wirz. Entwicklung einer bildauswertungsorientierten Inferenzmaschine in CYCLOPS. Master's thesis, Universität Kaiserslautern, 1994.

[20] S.Y. Wan and W.E. Higgins. Symmetric region growing. *IEEE Transactions on Image Processing*, 12(9):1007–1015, 2003.

[21] P. Haberäcker. *Praxis der Digitalen Bildverarbeitung und Mustererkennung.* Hanser, 1995.

[22] Bernd Jähne. *Digitale Bildverarbeitung.* Springer, 1997.

[23] Al Bovik. *Handbook of Image and Video Processing.* Academic Press, 2000.

[24] Regina Pohle and Klaus D. Toennies. Segmentation of medical images using adaptive region growing. In *Proceedings of SPIE (Medical Imaging 2001)*, volume 4322, pages 1337–1346, 2001.

[25] Pavel Paclík and Robert P.W. Duin. Multi-spectral image segmentation algorithm combining spatial and spectral information. *Proceedings of SCIA 2001 conference*, pages 230–235, 2001.

[26] K.-H. Tan, N. Ahuja. Selecting Objects with Freehand Sketches. In *Proc. IEEE International Conference on Computer Vision*, 2001.

[27] Volker Metzler, Ralf Vandenhouten, Jörg Krone, and Reinhard Grebe. Wissensbasierte Bildsegmentierung mittels stochastischer Optimierung. In *Digitale Bildverarbeitung in der Medizin, Tagungsband 5.Freiburger Workshop*, 1997.

[28] J. Iivarinen, A. Visa. Fault Analysis of Running Paper Web. Technical report, Laboratory of Computer and Information Science, Helsinki University of Technology, 1998.

[29] B.v. Ginneken, A.F. Frangi, J.J. Staal, B.M.ter Haar Romeny, M.A. Viergever. Active Shape Model Segmentation with Optimal Features. *IEEE Transactions on Medical Imaging*, 21(8):924–933, 2002.

[30] D. Cremers, T. Kohlberger and C. Schnörr. Nonlinear Shape Statistics in Mumford-Shah Based Segmentation. *7th European Conference on Computer Vision*, 2002.

[31] B. Bhanu, S. Lee and J. Ming. Adaptive Image Segmentation Using a Genetic Algorithm. *IEEE Transactions on Systems, Man and Cybernetics*, 25(12):1543–1567, 1995.

[32] J. Facon H.A. Legal-Ayala. Image segmentation by learning approach. In *Proceedings of ICDAR 2003*, 2003.

[33] Wikipedia – The free Encyclodedia. http://www.wikipedia.org.

[34] Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition 2nd Edition.* Academic Press, 1990.

[35] Sholom M. Weiss, Casimir A. Kulikowski. *Computer Systems That Learn.* Morgan Kaufmann, 1991.

[36] Xiaofen Ren, Jitendra Malik. Learning a Classification Model for Segmentation, 2003.

[37] Thomas Lehmann, Walter Oberschelp, Erich Pelikan und Rudolf Repges. *Bildverarbeitung für die Medizin.* Springer, 1997.

[38] M. Kerschner. *Snakes für Aufgaben der digitalen Photogrammetrie und Topographie.* PhD thesis, Technische Universität Wien, 2003.

[39] S. König, J. Hesser. Live-wires on edges of presegmented 2d-data. In *Bildverarbeitung für die Medizin 2003*, pages 156–160, 2003.

[40] R. M. Haralick, K. Shanmugam, I. Dinstein. Textural Features for Image Classification. *IEEE Transactions on Systems, Man and Cybernetics, SMC-3*, 1973.

[41] Wolfgang Abmayr. *Einführung in die digitale Bildverarbeitung.* B. G. Teubner Stuttgart, 1994.

[42] Klaus Toennies. Vorlesungsskript: Grundlagen der Bildverarbeitung (11) - Merkmale. http://sgwww.cs.uni-magdeburg.de/bv/skript/bv1/vlbv12.pdf.

[43] Winfried Kurth. Vorlesungsskript: Bildanalyse und Bildverstehen. http://www-gs.informatik.tu-cottbus.de/w̃wwgs/bia2_v07.pdf.

[44] I. Guyon, A. Elisseeff. An Introduction to Variable and Feature Selection. *Machine Learning Research 3*, 3(3):1157–1182, 2003.

[45] University of Waikato. WEKA: Waikato Environment for Knowledge Analysis. http://www.cs.waikato.ac.nz/~ml/index.html.

[46] Tim O. Müller. *Interaktive Synthese medizinischer Diagnosesysteme.* PhD thesis, Universität Karlsruhe, 2005.

[47] D. Michie, D.J. Spiegelhalter, C.C. Taylor. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, 1994.

[48] Khoral Research Inc. Khoros. http://www.khoral.com.

[49] National Instruments. LabVIEW. http://www.ni.com/labview.

[50] T.O. Müller, R. Stotzka, M. Beller, N.V. Ruiter, V. Hartmann. Schneller Aufbau medizinischer Diagnosesysteme mit der Komponentensoftware ICE. In *Bildverarbeitung für die Medizin 2004*, pages 443–447, 3 2004.

[51] Ian H. Witten, Eibe Frank. *Data Mining.* Morgan Kaufmann Publishers, 2000.

[52] Lothar Sachs. *Angewandte Statistik.* Springer, 9th edition, 1999.

[53] N. A. Thacker, A. F. Clark, J. Barron, R. Beveridge, C. Clark, P. Courtney, W.R. Crum, V. Ramesh. Performance Characterisation in Computer Vision: A Guide to Best Practices.

[54] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[55] M. Heath, K.W. Bowyer, D. Kopans et al. Current status of the digital database for screening mammography. In *Digital Mammography*, pages 457–460. Kluwer Academic Publishers, 1998.

[56] P. Brodatz. Textures, A Photographic Album for Artists and Designers, 1966.

[57] M.B. Popli. Pictorial Essay: Mammographic Features of Breast Cancer. *Indian Journal of Radiology and Imaging*, 11(4):175–179, 2001.

[58] M. Heath, K.W. Bowyer, D. Kopans. Current status of the digital database for screening mammography. *Digital Mammography*, 1998.

[59] H. Kobatake, M. Murakami, H. Takeo et. al. Computerized detection of malignant tumors on digital mammograms. *IEEE Trans. Med. Imaging*, 18(5):369–378, 1999.

[60] M.A. Kupinski and M.L. Giger. Automated seeded lesion segmentation on digital mammograms. *IEEE Trans. Med. Imaging*, 17(4):510–517, 1998.

[61] H. Li, Y. Wang, K.J.R. Liu et. al. Computerized Radiographic Mass Detection - Part I and II. *IEEE Trans. Med. Imaging*, 20(4):289–313, 2001.

[62] Mikaël Rousson, Thomas Brox, Rachid Deriche. Active Unsupervised Texture Segmentation on a Diffusion Based Feature Space. Technical report, Institut National de Recherche en Informatique et Automatique, 2003.

[63] A. Vailaya and A.K. Jain. Incremental learning for bayesian classification of images. In *Proceedings of ICIP 99*, volume 2, pages 585–589, 1999. http://citeseer.nj.nec.com/tuceryan98texture.html.

[64] Stefan Nordbruch. Automatische Klassifikation von Objekten für die visionbasierte Roboterregelung. Master's thesis, Universität Bremen, 2000.

[65] J. Beutel, H. L. Kundel, R. L. van Metter. *Handbook of medical imaging.* SPIE, 2000.

[66] Thomas Melzer. Vorlesungsskript: Einführung in die Mustererkennung. http://www.prip.tuwien.ac.at/~melzer/lehre/efme/VO_teil1.pdf.

[67] Thorsten Hermes. Vorlesungsskript: Bildverarbeitung. http://www.informatik.uni-bremen.de/~Hermes_VL/lectures/ws0102/17.01.2002.folien.pdf.

[68] Evangelia Micheli Tzanakou. *Supervised and unsupervised pattern recognition*. CRC Press, 2000.

[69] M. Tuceryan and A.K. Jain. Texture analysis. In *The Handbook of Pattern Recognition and Computer Vision*, pages 207–248, 1998. http://citeseer.nj.nec.com/tuceryan98texture.html.

[70] Edward J.Delp, Mostafa Analoui. Texture Analysis and its Applications in Medical Imaging. Technical report, SPIE's Medical Imaging Symposium, 2000.

[71] M. Wirth. Texture analysis. Technical report, Universiy of Guelph, 2000.

[72] Mary M. Galloway. Textural analysis using grey level run lengths. *Computer graphics and image processing, Vol. 4*, 1975.

Anhang A

Merkmale

In diesem Kapitel finden sich die Berechnungsvorschriften für alle im Laufe dieser Arbeit verwendeten Merkmale. Zur Erleichterung sind die Merkmale in alphabetisch sortiert. Grundsätzlich werden für die Berechnung der Merkmale keine Einheiten verwendet; diese können jedoch durch die Kenntnis der Auflösung berücksichtigt werden. Wenn nicht anders angegeben ist die Berechnungsvorschrift für den zweidimensionalen Fall angegeben, die meist problemlos auf die dritte Dimension erweitert werden kann.
Die Bezeichnung der Merkmale erfolgt wie im Abkürzungsverzeichnis in Englischer Sprache. Sofern sinnvoll, wird den Bezeichnung eine deutsche Übersetzung vorangestellt. Innerhalb der Formeln werden zur besseren Lesbarkeit zusätzliche Symbole benutzt, die an geeigneter Stelle eingeführt werden.

A.1 Statistische Merkmale

Statistische Merkmale beschreiben die statistischen Eigenschaften der Grauwerte eines Objekts. Es gibt Merkmale, die Aussagen über die Grauwerte selbst machen und solche, die das Grauwerthistogramm beschreiben.
Das Histogramm ist ein Diagramm der Häufigkeit $H(g)$) des Auftretens eines Grauwertes g für alle im Objekt vorkommenden Grauwerte.

Number of Greyvalues - Anzahl der Grauwerte

Die Anzahl der Grauwerte (SNumGV) N ist die Zahl aller im Objekt vorkommenden unterschiedlichen Grauwerte [64].

Maximaler Grauwert

Der maximale Grauwert (SMax) G_{max} ist gebräuchlicherweise der hellste Grauwert des Objektes. Er repräsentiert die rechte Grenze des Histogramms [41, 64].

Minimaler Grauwert

Der minimale Grauwert (SMin) G_{min} ist der dunkelste Grauwert des Objektes. Er repräsentiert die linke Grenze des Histogramms [41, 64].

Median

Der Median (SMed) ist eine Kennziffer der mittleren Lage aller N Grauwerte des Objektes.

Grey Spread - Spannweite der Grauwerte

Die Spannweite (SGS) S entspricht der maximalen Differenz zwischen zwei Grauwerten [41, 64]:

$$S = G_{max} - G_{min} \tag{A.1}$$

Histogramm

Die folgenden Extraktoren beruhen auf der Auftrittswahrscheinlichkeit $P(g)$ eines Grauwertes g, welche aus dem Histogramm berechnet wird. Sie treffen größtenteils eine Aussage über die Intensitätsverläufe des Objektes. Das Subscript $_S$ zeigt, wenn nicht anders angegeben, die Zugehörigkeit zu histogrammbasierten Merkmalen an, da gleichnamige Merkmale möglicherweise auch für Texturen berechnet werden können.

Anisotropiekoeffizient

Aus der Entropie (s. Gl. A.5) wird der Anisotropiekoeffizient (SAnisoCoeff) C_{A_S} abgeleitet. Er ist ein Maß für die Symmetrie des Histogramms [64]:

$$C_{A_S} = \frac{-\sum_{g=1}^{S} P(g) \log_2 P(g)}{H_S} \tag{A.2}$$

mit

$$\sum_{g=1}^{S} P(g) \geq 0,5 \tag{A.3}$$

und H_S als Entropie des Objektes (s. Gl. A.5) und S als Spannweite des Histogramm (s. Gl. A.1). Symmetrische Histogramme besitzen einen C_{A_S} von 0,5. Jede Abweichung hiervon stellt eine Asymmetrie dar [64].

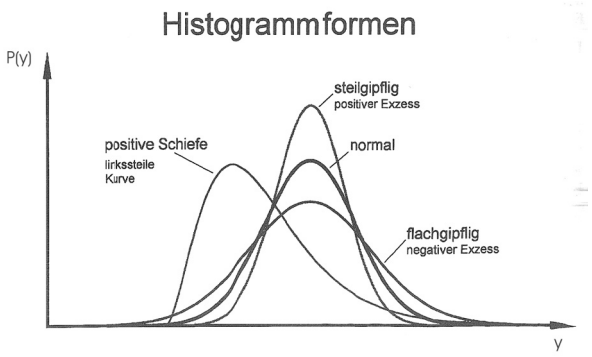


Abbildung A.1: *Histogrammformen: Gauß´sche Normalverteilung, positive Schiefe, positiver Exzeß (steilgipflig) und negativer Exzeß (flachgipflig) [41].*

Energie

Die Energie (SEner) E_S, in der Literatur auch Homogenität genannt, ist wie folgt definiert [43, 64, 65]:

$$E_S = \sum_{g=1}^{S} P(g)^2 \tag{A.4}$$

Entropie

Die Entropie (SEntr) H_S ist ein Maß für den mittleren Informationsgehalt des Objektes [64] und wird aus der relativen Wahrscheinlichkeit $P(g)$ der Graustufen des Histogramms abgeleitet [41, 43, 64, 65].

$$H_S = -\sum_{g=1}^{S} P(g) \log_2 P(g) \tag{A.5}$$

Kurtosis (Exzeß)

Die Kurtosis (SKurt, Exzeß) Ex_S zeigt an, wie weit eine Verteilung nach oben oder unten von der Gaußschen Normalform (s. Abb. A.1) abweicht [41]:

$$Ex_S = \frac{1}{\sigma_S^4} \sum_{g=1}^{S} (g - \mu_S)^4 P(g) \tag{A.6}$$

mit σ_S^2 als Varianz (s. Gl. A.13) und μ_S als Mittelwert (s. Gl. A.9) des Objektes.

Kurtosis Measure - Maßzahl des Exzesses

Die Maßzahl Exzeß (SKurtMeas) Ex_{M_S} ist ein Maß für den Exzeß [41, 64] in dimensionsloser Form. Die Subtraktion um den Wert drei wird durchgeführt, damit der Exzeß der Gauß'schen Normalverteilung Null ist.

$$Ex_{M_S} = \frac{Ex_S}{\sigma_S^2} - 3 \tag{A.7}$$

Mode Greyvalues - Häufigster Grauwert

Der häufigste Grauwert (SModeGV) G_H ist der Grauwert mit der höchsten Erhebung im Histogramm. Er trifft eine relative Aussage über die Farbe des Objektes [64].

$$G_H = max\{P(g_i)\} \quad i = (0, ..., S) \tag{A.8}$$

Mittelwert

Der Mittelwert μ_S des Objektes ist der mittlere aller N Werte des Histogramms. Er ist nach [41, 43, 66, 64, 65, 67] definert als

$$\mu_S = \sum_{g=1}^{S} g\, P(g) \tag{A.9}$$

Skewness - Schiefe

Die Skewness (SSkew, Schiefe) S_S berechnet den Grad der Asymmetrie einer Verteilung [41, 43, 64, 67]. Ist die Verteilung nach links geneigt, so ist die Schiefe positiv, ist sie nach rechts geneigt ist die Schiefe negativ (siehe Abbildung A.1) .

$$S_S = \frac{1}{\sigma_S^3} \sum_{g=0}^{S-1} (g - \mu_S)^3 P(g) \tag{A.10}$$

Skewness Measure - Maßzahl der Schiefe

Die dimensionslose Maßzahl der Schiefe (SSkewMeas) S_{M_S} für eine Gauß'sche Normalverteilung [64]:

$$S_{M_S} = \frac{S_S}{\sqrt{\sigma_S^2}^3} \tag{A.11}$$

Standardabweichung

Die Standardabweichung (SStdDev) σ_S ist ein Maß für die Streuung um den Mittelwert (s. Gl. A.9) [41]:

$$\sigma_S = \sqrt{\sigma_S^2} \tag{A.12}$$

Varianz

Die Varianz (SVar) σ_S^2 ist die Summe der quadratischen Abweichungen vom Mittelwert (s. Gl. A.9). Bei geringer Varianz sind die Grauwerte in einem kleinen Bereich um den Mittelwert verteilt [41, 43, 66, 64, 67].

$$\sigma_S^2 = \sum_{g=0}^{S-1} (g - \mu_S)^2 P(g) \tag{A.13}$$

Variationskoeffizient

Der Variationskoeffizient (SVarCoeff) VAR_S berechnet das Verhältnis aus der Varianz (s. Gl. A.13) und dem Mittelwert (s. Gl. A.9). Ist die Varianz klein, so ist auch der Variationskoeffizient klein. Dies bedeutet, daß sich die Grauwerte um einen kleinen Bereich um den Mittelwert befinden [64, 41].

$$\mathrm{VAR}_S = \frac{\sigma_S^2}{\mu_S} \tag{A.14}$$

A.2 Texturmerkmale

In diesem Kapitel werden die Berechnungsvorschriften der Texturmerkmale vorgestellt. Sie lassen sich in zwei Gruppen einteilen: Zum einen Merkmale, die aus der Cooccurrence-Matrix und zum anderen Merkmale, die aus der Lauflängen-Matrix extrahiert werden können, einteilen. Das Subscript $_T$ bedeutet, wenn nicht anders angegeben, die Zugehörigkeit zu texturbasierten Merkmalen, da gleichnamige Merkmale auch für Histogramme berechnet werden können.
Texturen sind aus sich wiederholenden kleinen Mustern aufgebaute Strukturen. Diese Muster sind wiederum aus sich wiederholenden Grauwerten in einer lokalen Nachbarschaft zusammengesetzt. Hierbei können die Grauwerte in einer solchen Nachbarschaft auch variieren [67].

Cooccurrence-Matrix

Die Cooccurrence-Matrix ist ein wichtiges Werkzeug für die Texturanalyse, indem sie die räumlichen Zusammenhänge des Objektes berücksichtigt. Sie wurde Im Jahr 1973 erstmals von Haralick [40] vorgestellt.
Die Zusammenhänge werden durch die Wahrscheinlichkeit $P(g, g')$ des Auftretens zweier Grauwerte g und g' mit dem räumlichen Abstand d betrachtet. Hierbei wird d als „Displacement" bezeichnet. Bei festem d gilt:

$$0 \leq P(g, g') \leq 1 \,\forall\, g, g' \qquad \text{und} \qquad \sum_{y=1}^{L_C} \sum_{z=1}^{L_C} P(y, z) = 1,$$

wobei L_C die Länge der Cooccurrence-Matrix ist, welche über die Anzahl G verschiedener Grauwerte bestimmt wird.

In einer Cooccurrence-Matrix $M_d(y,z)$ werden die Auftrittswahrscheinlichkeiten für ein festes Displacement zusammengefaßt.

$$M_d(y,z) = \begin{pmatrix} P(1,1) & P(2,1) & \cdots & P(G,1) \\ P(1,2) & P(2,2) & \cdots & P(G,2) \\ \vdots & \vdots & \ddots & \vdots \\ P(1,G) & P(2,G) & \cdots & P(G,G) \end{pmatrix}$$

Eine so erstellte Cooccurrence-Matrix ist nicht zwangsweise symmetrisch. Sie beinhaltet Eigenschaften der Textur und ermöglicht die Berechnung numerischer Merkmale. Sie ist nicht rotationsinvariant und hängt von der Wahl des Displacementvektors ab. Die Erweiterung des Displacementvektors um die Winkel 0°, 45°, 90°, 135° und Mittelung für jeden Eintrag in der Cooccurrence-Matrix ermöglichen dies jedoch.

Absolute Value - Absolutwert

Der Absolutwert (TAbsVal) A_T ist vorzeichenloses ein Maß aus der Cooccurrence-Matrix [43]:

$$A_T = \sum_{y=0}^{L_C-1} \sum_{z=0}^{L_C-1} |y-z|\, P(y,z) \tag{A.15}$$

mit L_C als Länge der Cooccurrence-Matrix und $P(y,z)$ als Auftrittswahrscheinlichkeit an der Stelle y,z in der Cooccurrence-Matrix.

Energie

Die Energie (TEner) E_T als Summe aller Quadrate der Grauwerte stellt ein Maß für die Homogenität dar [10, 40, 43, 64, 68, 69, 70]:

$$E_T = \sum_{y=1}^{L_C} \sum_{z=1}^{L_C} P(y,z)^2 \tag{A.16}$$

Entropie

Die Entropie (TEntr) H_T ist ein Maß für den mittleren Informationsgehalt des Objektes [71]. Die Informationsmenge läßt sich aus der relativen Auftrittswahr-

scheinlichkeit $P(y,z)$ der Cooccurrence-Matrix ableiten [43, 40, 68, 71, 69, 10, 70].

$$H_T = -\sum_{y=1}^{L_C}\sum_{z=1}^{L_C} P(y,z)\log_2 P(y,z) \tag{A.17}$$

Homogenität

Ein homogenes Objekt hat eine Cooccurrence-Matrix mit einer Kombination von hohen und niedrigen Wahrscheinlichkeiten [71]. Wenn die Spannweite klein ist, sind die Wahrscheinlichkeiten um die Hauptdiagonale konzentriert, während bei einem heterogenen Objekt die Wahrscheinlichkeiten gleichmäßig verteilt sind. Die Homogenität (THom) Hom_T ist durch

$$Hom_T = \sum_{y=1}^{L_C}\sum_{z=1}^{L_C} \frac{P(y,z)}{1+|y-z|} \tag{A.18}$$

definiert [10, 69, 71].

Inverse Difference Moment - Inverser–Differenz–Moment

Der inverse Differenz Moment (TInvDiffMom) IDM wird durch

$$IDM = \sum_{y=1}^{L_C}\sum_{z=1}^{L_C} \frac{1}{1+(y-z)^2} P(y,z) \tag{A.19}$$

berechnet [10, 40, 68, 70].

Kontrast

Der Kontrast (TCont) Kon_T hängt von hellen und dunklen Bildpunkten, sowie ihrer Entfernung voneinander ab [41]. Ist im Objekt eine große Anzahl von Variationen vorhanden, so ist der Kontrast hoch [40, 69, 70, 71].

$$Kon_T = \sum_{n=0}^{L_C-1} n^2 \left\{ \sum_{\substack{y=1 \\ |y-z|=n}}^{L_C} \sum_{z=1}^{L_C} P(y,z) \right\} \tag{A.20}$$

Mittelwert

Der Mittelwert (TMean) μ_T stellt das einfachste Texturmerkmal dar [64]. Verschiedene Texturen können denselben Mittelwert besitzen, da sie durch Variation der Tönungsfläche bestimmt werden. Dies macht es in diesem Fall nicht möglich die verschiedenen Texturen zu trennen [64, 68].

$$\mu_T = \sum_{y=1}^{L_C} \sum_{z=1}^{L_C} y\, z\, P(y,z) \tag{A.21}$$

Standardabweichung

Die Standardabweichung (TStdDev) σ_T kann als Maß für die Rauhigkeit der Textur betrachtet werden, da sie die Differenz der elementaren Merkmale bezogen auf die Extraktionsfläche beschreibt [64]. Sie wird durch

$$\sigma_T = \sqrt{\sigma_T^2} \tag{A.22}$$

berechnet [64, 68], wobei σ_T^2 die Varianz (s. Gl. A.23) der Cooccurrence-Matrix ist.

Varianz

Die Varianz (TVar) σ_T^2 ist die Summe der quadratischen Abweichungen vom Mittelwert (s. Gl. A.21) und ein Maß für die Abweichung von der Grundgesamtheit [10, 40, 64].

$$\sigma_T^2 = \sum_{y=1}^{L_C} \sum_{z=1}^{L_C} (y - \mu_T)^2 P(y,z) \tag{A.23}$$

Maximum Probability - Maximale Wahrscheinlichkeit

Die maximale Wahrscheinlichkeit (TMaxProb) P_{max} berechnet die maximale in der Cooccurrence-Matrix auftretende Wahrscheinlichkeit [70, 71].

$$P_{max} = max\{P(y_i, z_j)\} \quad i,j = (0, ..., L_C) \tag{A.24}$$

Grenzwertvektor

Es gibt zwei Grenzwertvektoren p_x und p_y. $p_x(i)$ ist der i-te Eintrag in p_x, welcher durch das Aufsummieren der Zeilen der Cooccurrence-Matrix entsteht. Genau so ist $p_y(i)$ der i-te Eintrag in p_y, welcher durch das Aufsummieren der Spalten der Cooccurrence-Matrix entsteht.
Mit den Grenzwertvektoren ist es mit Gl. A.9 und Gl. A.12 möglich, die Mittelwerte $\overline{p}_x$, $\overline{p}_y$ und die Standardabweichungen σ_x, σ_y von p_x, p_y zu berechnen [40].

Cluster Shade - Cluster Abstufung Die Cluster Abstufung (TClustShade) CS für die Cooccurrence-Matrix entspricht der Schiefe des Histogramms und ist nach [68] definiert als

$$C_A = \sum_{y=1}^{L_C} \sum_{z=1}^{L_C} ((y - \overline{p}_x) + (z - \overline{p}_y))^3 P(y, z) \tag{A.25}$$

Cluster Prominence - Cluster Erhebung Die Cluster Erhebung (TClustProm) C_E für die Cooccurrence-Matrix entspricht dem Exzeß des Histogramms und ist gemäß [68] definiert als:

$$C_E = \sum_{y=1}^{L_C} \sum_{z=1}^{L_C} ((y - \overline{p}_x) + (z - \overline{p}_y))^4 P(y, z) \tag{A.26}$$

Korrelation Die Korrelation (TCorr) ρ ist ein Maß für die Linearität eines Objektes. Sie ist groß, wenn das Objekt eine erhebliche Menge linearer Strukturen enthält [10, 40, 43, 68, 69, 71].

$$\rho = \frac{\sum_{y=1}^{L_C} \sum_{z=1}^{L_C} (yzP(y, z)) - (\overline{p}_x \overline{p}_y)}{\sigma_x \sigma_y} \tag{A.27}$$

Kovarianz Die Kovarianz (TCov) Kov ist die Summe der Abweichungen vom Mittelwert (s. Gl. A.21) und somit ein Maß für die Abweichung von der Grundgesamtheit [43].

$$Kov = \sum_{y=1}^{L_C} \sum_{z=1}^{L_C} (y - \overline{p}_x)(z - \overline{p}_y) P(y, z) \tag{A.28}$$

Summenvektor

Der Summenvektor p_{x+y} ist durch

$$p_{x+y}(k) = \sum_{\substack{y=1 \\ y+z=k}}^{L_C} \sum_{z=1}^{L_C} P(y,z) \qquad k = 2, 3, ..., 2L_C \tag{A.29}$$

definiert [40].

Sum Average - Mittlere Summe Die mittlere Summe (TSumAvg) S_{avg} wird aus dem Summenvektor berechnet und ist durch

$$S_{avg} = \sum_{y=2}^{2L_C} y p_{x+y}(y) \tag{A.30}$$

definiert [40, 68].

Summenentropie Die Summe Entropie (TSumEntr) S_{En} wird aus dem Summenvektor berechnet und ist durch

$$S_{En} = -\sum_{y=2}^{2L_C} p_{x+y}(y) \log\{p_{x+y}(y)\} \tag{A.31}$$

definiert [40, 68].

Summenvarianz Die Summe Varianz (TSumVar) S_{Va} wird aus dem Summenvektor berechnetund ist durch

$$S_{Va} = \sum_{y=2}^{2L_C} (y - S_{En})^2 p_{x+y}(y) \tag{A.32}$$

definiert [40, 68].

Differenzvektor

Der Differenzvektor p_{x-y} ist durch

$$p_{x-y}(k) = \sum_{\substack{y=1 \\ |y-z|=k}}^{L_C} \sum_{z=1}^{L_C} P(y,z) \qquad k = 0, 1, ..., L_C - 1 \tag{A.33}$$

definiert [40].

Differenz-Entropie Die Differenz Entropie (TDiffEntr) D_{Entr} wird aus dem Differenzvektor berechnet und ist durch

$$D_{Entr} = -\sum_{y=0}^{L_C-1} p_{x-y}(y) \log\{p_{x-y}(y)\} \tag{A.34}$$

definiert [40, 68].

Differenz-Varianz Die Differenz Varianz (TDiffVar) D_{σ^2} ist die Varianz (s. Gl. A.13) des Differenzvektors [40, 68].

$$D_{\sigma^2} = \sigma_S^2(p_{x-y}) \tag{A.35}$$

Information Measure Correlation 1 - Informationsgehalt Korrelation 1

Der Informationsgehalt Korrelation 1 (TIMCorr1) I_{Ko_1} wird mit Hilfe der Entropie der Cooccurrence-Matrix (s. Gl. A.17), den Entropien der Vektoren p_x, p_y (s. Gl. A.5) und einer gemischten Entropie der Cooccurrence-Matrix und den Vektoren p_x, p_y berechnet [40, 68]. Er ist durch

$$I_{Ko_1} = \frac{H_T - HXY1}{max\{E_S(p_x), E_S(p_y)\}} \tag{A.36}$$

definiert, mit

$$HXY1 = -\sum_{y=1}^{L_C}\sum_{z=1}^{L_C} P(y,z) \log\{p_x(y)p_y(z)\} \tag{A.37}$$

und $max\{E_S(p_x), E_S(p_y)\}$ als Maximum der Entropien der Vektoren p_x und p_y.

Information Measure Correlation 2 - Informationsgehalt Korrelation 2

Der Informationsgehalt Korrelation 2 (TIMCorr2) I_{Ko_2} wird mit Hilfe der Entropie der Cooccurrence-Matrix (s. Gl. A.17) und der Entropie der Vektoren p_x, p_y (s. Gl. A.5) berechnet [40], [68]. Er ist durch

$$I_{Ko_2} = \{1 - e^{-2(HXY2-H_T)}\}^{\frac{1}{2}} \tag{A.38}$$

definiert, mit

$$HXY2 = -\sum_{y=1}^{L_C}\sum_{z=1}^{L_C} p_x(y)p_y(z)log(p_x(y)p_y(z)) \tag{A.39}$$

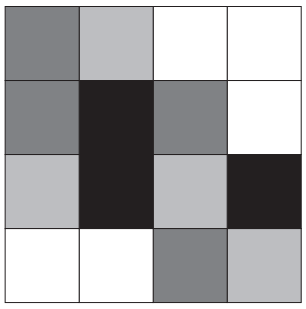
Maximaler Korrelationskoeffizient

Der maximale Korrelationskoeffizient (TMaxCorrCoeff) $C_{Korr_{max}}$ ist ein Maß, welches aus der Cooccurrence-Matrix und den Grenzwertvektoren berechnet werden kann. Er ist die Wurzel des zweitgrößten Eigenwertes der Matrix Q [40] mit

$$Q(y,z) = \sum_{k=1}^{L_C} \frac{P(y,k)P(z,k)}{p_x(y)p_y(k)} \tag{A.40}$$

Lauflängen-Matrix

In diesem Abschnitt werden die Berechnungsvorschriften für die Merkmale, welche aus der Lauflängen-Matrix extrahiert werden können vorgestellt. Die Lauflängen-Matrix stellt ein wichtiges Hilfsmittel für die Texturanalyse zur Verfügung. Zu ihrer Erstellung werden die Grauwerte mit der Häufigkeit des Auftretens ihrer Längen in einer Matrix eingetragen. Die Zeilen stehen für die Grauwerte (z. B. 0 - 255), die Spalten für die Länge des Laufes (1 - maximaler Durchmesser des Objektes). Ein Eintrag in der Matrix an der Stelle x, y mit dem Wert 2 besagt, daß der Grauwert x mit der Lauflänge y zweimal im Objekt vorkommt.
Bei der Erstellung der Matrix werden nur ganze Läufe betrachtet, Teilsequenzen eines Laufes werden nicht berücksichtigt, d. h. kommt ein Grauwert x mit der Lauflänge $y = 2$ (x zweimal nebeneinander) im Objekt vor, so wird in der Matrix an der Stelle x, $y = 2$ der Wert inkrementiert. Der Wert an der Stelle x, $y = 1$ bleibt unberührt.



(a)

1	2	3	3
1	0	1	3
2	0	2	0
3	3	1	2

(b)

Abbildung A.2: *Erstellung einer Lauflängenmatrix. (a) Grauwerte, (b) Lauflängen*

In Abbildung A.2 ist beispielhaft ein Grauwertbild dargestellt. In a sind die Grauwerte und in b die dazugehörigen Zahlenwerte zu sehen. Für einen Winkel von 0° (horizontal) für die Richtung der Lauflänge liefert die Berechnung folgende Matrix:

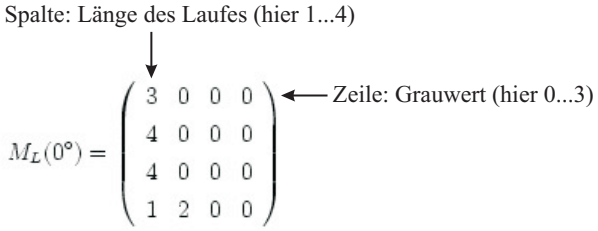


Abbildung A.3: Aus der Abbildung A.2 berechnete Lauflängenmatrix für einen Winkel von 0°(horizontal).

Diese Darstellung ist wie bei der Cooccurrence-Matrix nicht rotationsinvariant. Um dies zu ermöglichen müssen für die Erstellung der Lauflängen-Matrix die Winkel von 0°, 45°, 90° und 135° verwendet werden.

Short Run Emphasis - Betonung kurzer Läufe

Die Betonung kurzer Läufe (TRlSRE) dividiert im Zähler jeden Eintrag in der Lauflängen-Matrix durch das Quadrat ihre Länge. Der Nenner ist die Gesamtzahl der Läufe im Objekt und dient zur Normalisierung. Sie ist durch

$$TRlSRE = \frac{\sum_{y=1}^{L_L} \sum_{z=1}^{L_L} \frac{P_L(y,z)}{z^2}}{\sum_{y=1}^{L_L} \sum_{z=1}^{L_L} P_L(y,z)} \tag{A.41}$$

definiert [67, 71, 72], wobei L_L die Länge der Lauflängen-Matrix und $P_L(y,z)$ die Summe des Auftretens des Grauwertes y mit der Lauflänge z in der Lauflängen-Matrix ist.

Long Run Emphasis - Betonung langer Läufe

Die Betonung langer Läufe (TRlLRE) multipliziert im Zähler jeden Eintrag in der Lauflängen-Matrix mit dem Quadrat der Länge. Der Nenner dient auch hier

zur Normalisierung. Sie ist definiert [67, 71, 72] als

$$TRlLRE = \frac{\sum_{y=1}^{L_L}\sum_{z=1}^{L_L} z^2 P_L(y,z)}{\sum_{y=1}^{L_L}\sum_{z=1}^{L_L} P_L(y,z)} \tag{A.42}$$

Run Percentage - Prozentuale Anzahl der Läufe

Die prozentuale Anzahl der Läufe (TRlRunPerc) berechnet das Verhältnis der Läufe in der Lauflängen-Matrix und dem Volumen. Sie ist definiert [67, 71, 72] als

$$TRlRunPerc = \frac{\sum_{y=1}^{L_L}\sum_{z=1}^{L_L} P_L(y,z)}{V} \tag{A.43}$$

Greylevel Distribution - Ungleichmäßigkeit der Grauwerte

Die Ungleichmäßigkeit der Grauwerte (TRlGLDist) ist definiert [67, 71, 72] als

$$TRlGLDist = \frac{\sum_{y=1}^{L_L}\left\{\sum_{z=1}^{L_L} P_L(y,z)\right\}^2}{\sum_{y=1}^{L_L}\sum_{z=1}^{L_L} P_L(y,z)} \tag{A.44}$$

Run-length Distribution - Ungleichmäßigkeit der Lauflängen

Die Ungleichmäßigkeit der Lauflängen (TRlDist) ist definiert [67, 71, 72] als

$$TRlDist = \frac{\sum_{z=1}^{L_L}\left\{\sum_{y=1}^{L_L} P_L(y,z)\right\}^2}{\sum_{y=1}^{L_L}\sum_{z=1}^{L_L} P_L(y,z)} \tag{A.45}$$