Diploma Thesis, Mathematics and Computer Science Department, University of Mannheim, March 2000
by Stephan Kopf

Objekterkennung in Bildsequenzen unter Verwendung des krümmungsbasierten Skalenraums

Diplomarbeit
von

Stephan Richter
aus Langenfeld

vorgelegt am

Lehrstuhl für Praktische Informatik IV
Prof. Dr. W. Effelsberg

Fakultät für Mathematik und Informatik
Universität Mannheim

März 2000

Betreuer: Dipl.-Wirtsch.-Inf. Gerald Kühne

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Abb.	Abbildung
AVI	Audio Video Interleaved (Multimediaformat von Microsoft Windows)
Bsp.	Beispiel
CSS	Curvature Scale Space (krümmungsbasierter Skalenraum)
d. h.	das heißt
PGM	Portable Graymap Bildformat zum Speichern unkompremierter Grafiken
RGB-Bilder	Rot-Grün-Blau Bilder (Farbdarstellung in 24-Bit Farbtiefe)
Tab.	Tabelle
vgl.	vergleiche
YUV	Helligkeit und Farbkomponenten (Luminanz und Chrominanz)
z. B.	zum Beispiel
z. T.	zum Teil
z. Z.	zur Zeit

Kapitel 1

Einleitung

In den letzten Jahren hat die Entwicklung von Anwendungen im Bereich Multimedia enorme Fortschritte erzielt. Digitale Videos sind in vielen Geschäften erhältlich und werden in naher Zukunft die analogen Videokassetten ablösen. Fast jeder PC, der in den letzten Jahren auf den Markt kam, besitzt die Möglichkeit zum Abspielen und Editieren digitaler Videos. Diese bisher verfügbaren Grundfunktionalitäten für den Umgang mit Videos reichen vielen Anwendern nicht aus. Ein Textverarbeitungssystem bietet neben dem Laden, Speichern und Editieren von Texten die Möglichkeit zur Suche nach Begriffen. Übertragen auf digitale Videos würde das einer Suche nach Objekten oder Szenen innerhalb eines Films entsprechen. Falls der digitale Film keine zusätzliche Spur mit Metadaten enthält, ist diese Suche mit den heutigen Systemen nicht möglich.

Die folgende Arbeit soll in eingeschränktem Maß die Fähigkeit eines Menschen im Bereich der Objekterkennung in Bildsequenzen nachbilden. Das Objekterkennungsverfahren berücksichtigt ausschließlich die Kontur eines Objekts. Nach einem Segmentierungsschritt wird die Kontur des abgebildeten Objekts durch ihre konkav und konvex gekrümmten Bereiche beschrieben. Eine Transformation des Objekts in eine Abbildung im krümmungsbasierten Skalenraum ermöglicht in effizienter Weise die Bestimmung der am stärksten gekrümmten Bereiche der Kontur.

Die Arbeit gliedert sich wie folgt: Im zweiten Kapitel werden Merkmale der menschlichen Wahrnehmung dargestellt. Diese Charakteristika ermöglichen die Auswahl geeigneter Objekte für die Datenbank. Der Begriff der Krümmung wird in Kapitel drei eingeführt. Zusätzlich werden grundlegende Filter zur Manipulation von Bildern beschrieben. Die Merkmale von Abbildungen im krümmungsbasierten Skalenraum sind wesentlicher Bestandteil dieser Arbeit und folgen in Kapitel vier. Diese Abbildungen ermöglichen die Bestimmung der stärksten konkaven Bereiche einer Kontur.

Das fünfte Kapitel beschreibt den implementierten Objekterkennungsalgorithmus. Neben der Segmentierung des abgebildeten Objekts und der Berechnung der Abbildung im krümmungsbasierten Skalenraum wird ein Algorithmus vorgestellt, der zwei Objekte miteinander vergleicht. Die Ergebnisse, die mit diesem Verfahren zur Objekterkennung erzielt werden kön-

nen, werden anhand von Einzelbildern und Bildsequenzen im sechsten Kapitel vorgestellt. Dabei wird der Algorithmus sowohl für Bildsequenzen mit starren Objekten, wie z. B. Autos, als auch für Objekte, die ihre Form während der Bewegung deutlich verändern, getestet. Im abschließenden Kapitel wird das Verfahren kritisch beurteilt und Verbesserungsmöglichkeiten werden aufgezeigt.

Kapitel 2

Visuelle Wahrnehmung dreidimensionaler Objekte durch den Menschen

Das langfristige Ziel der rechnergestützten Bildanalyse ist eine qualitativ hochwertige Objekterkennung, die mit der Schnelligkeit und Präzision des Menschen vergleichbar ist. Der Mensch kann in vielfältigen Umgebungen und unter sehr variablen Bedingungen Objekte wiedererkennen. Trotz Änderungen der Helligkeit oder der Ausrichtung dreidimensionaler Objekte im Raum gelingt es einem Beobachter, eine passende Zuordnung eines Objekts zu einem Begriff in kürzester Zeit zu finden.

In diesem Abschnitt sollen die wesentlichen visuellen Eigenschaften, die ein Mensch zur Orientierung in seiner Umwelt benötigt, dargestellt werden. In der Forschung existieren zwei vorherrschende Theorien[1], wie dreidimensionale Objekte durch den Menschen wahrgenommen und im Gehirn gespeichert werden. Der Unterschied beider Theorien basiert auf der Bedeutung des Blickwinkels. Die für diese Arbeit relevante Theorie geht davon aus, dass für einen Menschen zur Erkennung eines bekannten Objekts bessere und schlechtere Ansichten existieren [5, 8, 32]. Sie legt die Annahme zugrunde, dass die Beobachtungsperspektive die Objekterkennung ganz wesentlich beeinflusst. Bestimmte Blickwinkel eines Objekts erleichtern dessen Erkennung. Daher wird vermutet, dass jedes Objekt in verschiedenen präferierten Ansichten als zweidimensionales Bild im Gehirn gespeichert wird. Ergänzt werden diese rein zweidimensionalen Abbildungen durch Tiefeninformationen. Dieser Ansatz soll Grundlage der folgenden Arbeit sein, da für die Objekterkennung zweidimensionale Abbildungen verwendet werden können und eine dreidimensionale Darstellung eines Objekts nicht benötigt wird.

Viele weitere Faktoren beeinflussen bei einem Menschen die Fähigkeit, Objekte schnell und zuverlässig zu erkennen. Die wichtigsten Faktoren werden nachfolgend erläutert und rechnerge-

[1] Die Theorien werden als *viewpoint invariant representation* und als *viewpoint dependent representation* bezeichnet.

stützte Ansätze zu deren Erkennung aufgezeigt. Voraussetzung für die weitere Untersuchung ist eine Reduktion der abgebildeten Objekte auf ihre Kontur. Die Arbeit liefert für ein sehr eingegrenztes Problemfeld eine Lösungsmöglichkeit zur Objekterkennung in Bildsequenzen.

2.1 Erkennung von Objekten

Der wichtigste Sinn, den ein Mensch zur Orientierung in seiner Umwelt benötigt, ist sein visueller Wahrnehmungssinn. Objekte und Strukturen müssen ständig analysiert und schnell erkannt werden.

Die Objekte, die ein Mensch wahrnimmt, werden als Helligkeitspunkte mit Hilfe der Netzhaut wahrgenommen. Die Zuordnung von Kanten, Oberflächenstrukturen und Konturen zu Objekten ist eine bemerkenswerte Leistung, die das Gehirn in kürzester Zeit vollbringen muss. Daher ist es auch nicht verwunderlich, wenn mehr als die Hälfte der Hirnrinde des Menschen für die Verarbeitung der visuellen Wahrnehmung zuständig ist [32].

Was bedeutet nun für einen Menschen das Erkennen dreidimensionaler Objekte? Im folgenden soll unter *Erkennen* eine Zuordnung eines speziellen Objekts zu einer Bezeichnung verstanden werden. Jede Bezeichnung eines Objekts ist üblicherweise in eine *Hierarchie* eingebettet. So lassen sich für die Abbildung einer speziellen Katze die Begriffe Vierfüßler, Säugetier, Katze oder Siamkatze verwenden. Bei einer dem Betrachter bekannten Katze kann alternativ der individuelle Name des Tieres angegeben werden [10]. In der Objekterkennung wurden *drei hierarchische Stufen* für die Wahl der Bezeichnung eingeführt [5]. Wenn ein Mensch ein Bild betrachtet, erkennt er zunächst die *Objektklasse*[2]. Die Objektklasse bezeichnet den Begriff, der einem Betrachter bei kurzem Blick auf das Objekt zuerst in den Sinn kommt. Hund, Katze oder Vogel sind typische Bezeichnungen für Objektklassen. Einige Klassen weisen jedoch Inkonsistenzen auf. Bei der Abbildung eines Pinguins fällt einem Betrachter üblicherweise direkt diese Bezeichnung ein und nicht die sonst übliche Bezeichnung *Vogel*. Ganz wesentlich für die Zuordnung zu einer Objektklasse ist die Kontur des betrachteten Objekts [33]. Da sich Pinguine in ihrer Kontur deutlich von anderen Vögeln unterscheiden, erscheint die Einführung einer eigenständigen Objektklasse *Pinguin* nachvollziehbar.

Abhängig von der gewählten Ansicht eines Objekts spielen beim Menschen Schattierungen, Oberflächentexturen und Lichtverhältnisse erst in späteren Erkennungsschritten eine Rolle [19]. In der Theorie wird angenommen, dass ein Mensch in einem ersten Wahrnehmungsschritt immer die Objektklasse zu ermitteln versucht, um anschließend mit Hilfe dieser grundlegenden Information das Objekt genauer zu spezifizieren.

In einem zweiten Schritt, der üblicherweise bei Menschen einen längeren Erkennungsprozess erfordert, wird eine *detaillierte Bezeichnung für ein Objekt*[3] gefunden. Als Beispiel aus

[2] Die Hierarchiestufe der Objektklasse wurde erstmals 1984 unter der Bezeichnung *basic-level* oder *entry-level* eingeführt [32, 33].

[3] Diese Hierarchiestufe bei der Objekterkennung wird als *subordinate-level* bezeichnet [32].

der Tierwelt kann ein spezieller Vogel, z. B. ein Spatz oder eine Amsel genannt werden. Genaue Bezeichnungen eines Objekts lassen sich nur dann ermitteln, wenn die Form des Objekts exakt wiedergegeben wird und wenn der Betrachter die Oberflächenbeschaffenheit und Farbe erkennen kann. Es reicht für einen Menschen häufig nicht mehr aus, lediglich die Kontur dieses Objekts zu betrachten. Um sicher ein spezielles Objekt zuordnen zu können, werden zusätzlich Oberflächeninformationen in Form von Farbe und Schattierungen benötigt [19, 10].

Die Umgebung, in der das gesuchte Objekt gesehen wird, kann die Objekterkennung erleichtern und beschleunigen. So wird ein Brotlaib deutlich leichter in der Umgebung einer Küche identifiziert als in einer für dieses Objekt ungewöhnlichen Umgebung. Auch zusätzliche Objekte, die neben dem zu identifizierenden Objekt abgebildet werden, erleichtern einem Menschen die Objekterkennung. Dem Betrachter wird als zusätzliche Information eine Orientierung und eine Größenvorstellung des unbekannten Objekts vermittelt. Mit Hilfe eines virtuellen Raumes wurde ein signifikanter Zusammenhang zwischen erfolgreicher und schneller Objekterkennung und einer dem Betrachter bekannten Umgebung nachgewiesen [8]. Bei Hintergrundinformationen und mehreren im Raum angeordneten Objekten gelingt eine Objekterkennung signifikant besser, d. h. die Fehlerrate und die benötigte Zeitdauer bis zur Erkennung des Objekts sinkt. Die Objekterkennung ist somit kontextsensitiv.

In einem dritten Erkennungsschritt kann eine *individuelle Objektbezeichnung* gefunden werden [32]. Ein Betrachter identifiziert ein spezielles, individuell bekanntes Objekt. Die Identifikation eines speziellen Objekts ist nur dann möglich, wenn das Objekt dem Menschen schon längere Zeit bekannt ist. Als zusätzliche Information müssen verschiedene Perspektiven des Objekts verfügbar sein. Ein Beispiel für ein spezielles Objekt kann ein Kanarienvogel sein, der sich nur minimal von anderen Kanarienvögeln unterscheidet, dessen Unterschiede dem Besitzer nach genauerer Betrachtung jedoch auffallen.

Menschen verwenden noch weitere Informationen zur Erkennung von Objekten wie z. B. die Fähigkeit der räumlichen Sicht. Diese führt jedoch nur zu einer geringfügig *besseren Objekterkennung*[4]. So liegt die Fehlerrate und die Zeitdauer bis zur Identifikation von Objekten in Stereobildern nur unwesentlich niedriger [5].

Besonders wichtig sind eine konsistente Umgebung und Hintergrundinformationen des Bildes, die dem Betrachter den Erkennungsprozess deutlich erleichtern. Ein grauer Hintergrund erschwert die Orientierung und dadurch die Wiedererkennung von zuvor gezeigten Objekten. Ein natürlicher und detailierter Hintergrund unterstützt die Erkennung [8]. Weitere wichtige Informationen kann ein Beobachter aus der Art und Weise entnehmen, wie sich die Helligkeit und die Farbe der Oberfläche oder die Form des Objekts bei Bewegung ändert [35]. Bei starren Objekten können durch die Bewegung bisher verdeckte Konturen sichtbar werden und bestehende Konturen verdeckt werden. Bei Lebewesen tritt zusätzlich noch eine deutliche Deformation der

[4]Unter einer *guten Objekterkennung* soll eine schnelle und möglichst fehlerfreie Erkennung verstanden werden, d. h. die Objekterkennung ist gut, wenn ein Mensch eine kurze Antwortzeit zur korrekten Identifikation des Objekts benötigt.

Kontur auf. Die Arme, Beine und der Kopf eines Menschen oder Tieres ändern ihre Position bei der Fortbewegung.

2.2 Repräsentation dreidimensionaler Objekte im Gehirn des Menschen

Ein Mensch erkennt Objekte, indem zunächst gleichhelle zusammenliegende Lichtpunkte auf der Netzhaut zu kleinen Strukturen zusammengefasst werden. Gerade oder nur schwach gekrümmte Linien werden dabei besonders leicht vom Menschen als zusammenhängende Struktur erkannt. Diese kleinen Strukturen werden mit ihren Nachbarstrukturen kombiniert und ergeben zunächst Flächen, später komplexe dreidimensionale Gebilde [32].

Es gibt zwei wesentliche Ansätze [5, 33], die beschreiben, wie ein Mensch Abbildungen von Objekten im Gehirn vermutlich speichert. Ausgehend von sich widersprechenden Hypothesen werden Annahmen bezüglich der Bedeutung des Beobachtungsstandpunkts eines Menschen getroffen. Der erste Ansatz nimmt an, dass der Mensch ein Objekt als Ganzes speichert. Sämtliche Informationen werden im Gehirn in Form eines dreidimensionalen Modells hinterlegt.

Wenn ein Mensch Objekte aus einer unbekannten Perspektive betrachtet und längere Zeit benötigt, um diese zu erkennen, so wird davon ausgegangen, dass relevante Konturen durch das Objekt selbst verdeckt werden. Je nach Drehung eines Objekts um die drei Koordinatenachsen wird mehr oder weniger Fläche für den Beobachter sichtbar. Allgemein ist jedoch bei der dreidimensionalen Speicherung des Objekts der Blickwinkel von untergeordneter Bedeutung.

Der zweite Ansatz geht von der Annahme aus, dass der Mensch von einem Objekt verschiedene zweidimensionale Ansichten gespeichert hat. Unbekannte Perspektiven eines bekannten Objekts erkennt ein Mensch, indem er die im Gehirn gespeicherten Ansichten bekannter Objekte so lange kombiniert, bis eine Kombination mit ausreichender Ähnlichkeit gefunden wurde. Wie dieser Kombinations- und Normalisierungsprozess im Menschen abläuft, ist aktuelles Forschungsgebiet und zur heutigen Zeit noch nicht abschließend geklärt [32, 8, 5]. Der *Beobachtungspunkt* des Betrachters ist im zweiten Ansatz von ganz zentraler Bedeutung.

Ein erster Hinweis für die Bedeutung dieser zweiten Theorie basiert auf Beobachtungen [3, 5], dass Objekte in den Perspektiven, in denen ein Mensch ein Objekt üblicherweise betrachtet, besonders leicht und zuverlässig wiedererkannt werden (vgl. auch Abb. 2.1 auf der nächsten Seite). In ungewohnten Perspektiven, bereitet die Erkennung größere Probleme und der Erkennungsprozess nimmt wesentlich mehr Zeit in Anspruch. Obwohl nicht sicher geklärt ist, wie die Erkennung eines Objekts in unbekannter Perspektive abläuft, wird vermutet, dass ein Mensch aus mehreren bekannten Ansichten das unbekannte Objekt kombiniert[5]. Je weiter eine Ansicht eines Objekts von einer bekannten Ansicht entfernt ist, desto länger dauert der Erkennungsprozess [8]. Die *Entfernung* zweier Ansichten kann in diesem Zusammenhang durch die benötigte

[5] Diese Kombinationsfähigkeit des Menschen wird als *Interpolation Across Views* bezeichnet [33].

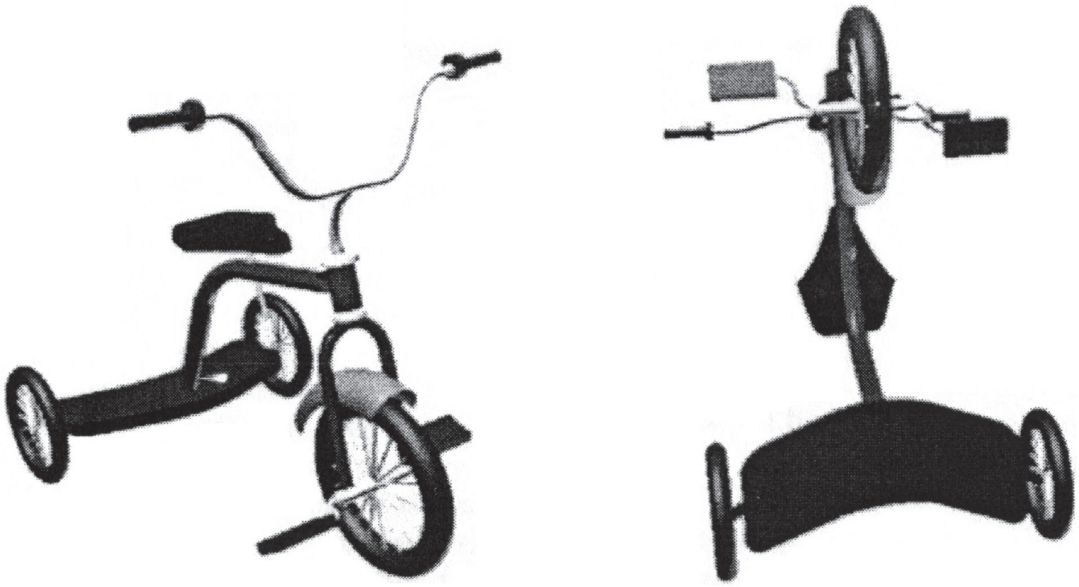

Abbildung 2.1: Schwierigkeiten beim Erkennen eines Dreirads. Die linke Abbildung wird von einem Menschen leicht erkannt, die Abbildung rechts stellt für einen Menschen eine ungewohnte Ansicht dar. Die Grafik wurde aus [5] entnommen.

Drehung des Objekts gemessen werden. Eine maximale Entfernung tritt bei einem Winkel von 90 Grad auf, da Rückansichten des Objekts wieder deutlich leichter erkannt werden können. Ein Beobachter, der mit unbekannten Ansichten eines Objekts konfrontiert wird, erkennt dieses Objekt zunächst schlecht oder falsch. Nach einer gewissen Lernzeit wird diesem Menschen die neue zweidimensionale Projektion des Objekts vertraut, und die Erkennungsrate nähert sich der Rate bei der üblichen Perspektive an.

Der Erkennungsprozess beim Menschen ist vermutlich eine Kombination dieser beiden, sich auf den ersten Blick widersprechenden Theorien. Neben der Speicherung spezieller und präferierter zweidimensionaler Ansichten eines Objekts werden zusätzliche dreidimensionale Informationen für den Erkennungsprozess benötigt, so beispielsweise auch die Schattierung eines Objekts. Ausschließlich anhand der Kontur lässt sich nicht bestimmen, ob die Oberfläche eines Objekts glatt oder gebogen ist, ob Vertiefungen in der Objektmitte auftreten oder wie die Position einzelner Teile des Objekts zueinander stehen. Dreidimensionale Informationen lassen sich aus den Farb- oder Helligkeitsverläufen rekonstruieren.

Für die folgende Arbeit erscheint besonders der zweite Ansatz erfolgversprechend. Die Objekterkennung in dieser Arbeit wird auf Basis einer Datenbank realisiert, die zuerst eingerichtet wird. Als Datenbasis dienen zweidimensionale Ansichten eines dreidimensionalen Objekts. Diese zweidimensionalen Projektionen sollen eine Objektklasse möglichst repräsentativ wiedergeben. Eine Analyse von Bildfolgen bietet die zusätzliche Möglichkeit, ein unbekanntes Objekt

aus unterschiedlichen Perspektiven zu betrachten. Gerade durch eine Drehung um die Koordinatenachsen sowie eine geringe Deformation des Objekts wird die Möglichkeit für eine Erkennung erhöht.

Eine noch zu klärende Frage besteht in der Auswahl dieser zweidimensionalen Referenzbilder für die Objekte der Datenbank. Als Grundlage sollen Ergebnisse aus dem Bereich der Wahrnehmungspsychologie herangezogen werden. In den Untersuchungen wurden zweidimensionale Ansichten - sogenannte *kanonische Sichten* - von dreidimensionalen Objekten ermittelt, die für den Menschen besonders schnell erkennbar sind. Diese kanonischen Sichten werden im folgenden Abschnitt erläutert.

2.3 Kanonische Sichten

Anfang der achtziger Jahre wurde der Begriff der *kanonischen Sichten* eingeführt[6]. Dieser Begriff bezeichnet ein oder mehrere besonders relevante zweidimensionale Projektionen eines bekannten dreidimensionalen Objekts. In verschiedenen Untersuchungen [3, 5] wurde deutlich, dass abhängig von der Aufgabenstellung unterschiedliche Ansichten eine größere Bedeutung gewinnen. In einem Experiment sollen sich Versuchsteilnehmer ein genanntes Objekt vorstellen, das mit Hilfe einer detaillierten Beschreibung, eines dreidimensionalen Modells oder eines virtuellen Raumes erfasst wird. Die Versuchsteilnehmer wählen fast immer eine *Ansicht im Profil*. Tiere oder Fahrzeuge werden nahezu immer von der Seite gezeigt (90 Grad Drehung zur Frontalansicht). Bei einer anderen Aufgabenstellung sollten Objekte durch die Versuchsteilnehmer um alle drei Koordinatenachsen gedreht werden, so dass eine möglichst gute Ansicht für einen Katalog entstehen würde. Die Ergebnisse dieses zweiten Experiments sind in Abbildung 2.2 auf der nächsten Seite abgebildet. In dieser zweiten Versuchsanordnung wurde deutlich, dass Ansichten ausgewählt werden, bei denen möglichst viel Oberfläche des Objekts sichtbar und die Kontur gut zu erkennen ist. Diese Projektionen sind sehr *fehlertolerant*[7]. Die präferierte Perspektive bekannter Objekte war eine Ansicht von schräg vorne (30 - 45 Grad), die zudem noch leicht erhöht war (10 - 15 Grad). Bei Flugzeugen und Hubschraubern wurden als einzige Ausnahme eine Abbildung von schräg unten gleichermaßen bevorzugt. Die beiden letztgenannten Objekte werden von Menschen üblicherweise in dieser speziellen Perspektive wahrgenommen. Jede Abweichung von diesen kanonischen Sichten (Profilansicht oder Projektion von schräg vorne), erhöht bis zu einem Winkel von 90 Grad die Fehlerrate bei der Objekterkennung [5].

Im wesentlichen zeichnen vier Merkmale [3] kanonische Sichten aus: Bevorzugt werden *vertraute Perspektiven,* d. h. Perspektiven in denen üblicherweise Objekte wahrgenommen werden. Ebenso spielt die *Funktionalität,* also die konkrete Verwendung eines Objekts, eine wesentli-

[6] Palmer, Rosch und Chase haben 1981 den Begriff *canonical views* geprägt [29]. Versuche mit Fotografien von Objekten aus unterschiedlichen Perspektiven ergaben, dass alle Versuchsteilnehmer bestimmte Perspektiven unabhängig vom gewählten Objekt, bevorzugen.

[7] Eine fehlertolerante Abbildung verändert ihre Kontur bei geringer Drehung des Objekts nur minimal.

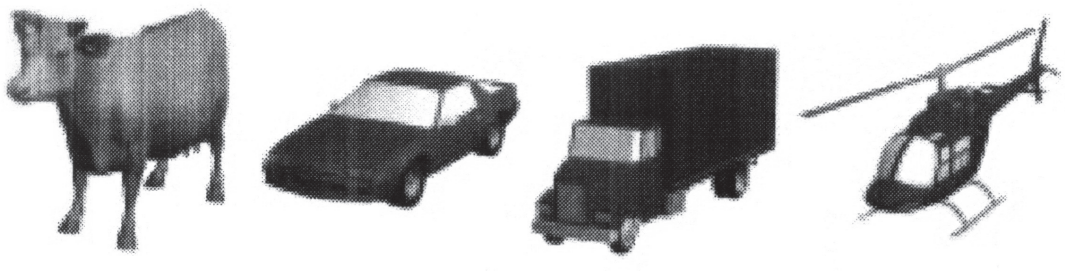

Abbildung 2.2: Beispiele für kanonische Sichten. Eine leicht erhöhte Perspektive von schräg vorne bildet für einen Menschen ein Objekt besonders gut ab. Die Grafik wurde leicht modifiziert aus [3] entnommen.

Abbildung 2.3: Typische und weniger typische Konturen eines Objekts. Verschiedene Ansichten eines Objekts werden anhand ihrer gefüllten Konturen abgebildet. Links ist eine kanonische Sicht in Form einer leicht erhöhten Profilansicht zu sehen. Nach rechts nimmt mit zunehmender Entfernung von der kanonischen Sicht die Möglichkeit der Wiedererkennung für einen Menschen ab. Die Abbildung wurde aus [10] entnommen.

che Rolle. Bei Gebrauchsgegenständen (Bsp.: Hammer, Teekanne) wird die Ansicht bevorzugt, in der ein Objekt üblicherweise benutzt wird. So haben Links- und Rechtshänder häufig unterschiedliche bevorzugte Ansichten. *Ästhetische Merkmale*, wie z. B. Symmetrieeffekte, beeinflussen zusätzlich die kanonische Sicht eines Objekts. Das wichtigste Merkmal liegt jedoch in der Möglichkeit, das Objekt leicht zu erkennen. Die *Qualität der Wiedererkennung* lässt sich anhand einer Vielzahl von Merkmalen, wie die Anzahl und Größe der verdeckten Strukturen [29] oder die *Stabilität der Projektion*[8], charakterisieren (vgl. Abb. 2.3). Alternativ kann die kanonische Sicht eines Objekts als diejenige Projektion eines Objekts definiert werden, die die größte Ähnlichkeit mit beliebigen anderen Projektionen dieses Objekts aufweist [10].

Aufgrund dieses Ergebnisses werden als Referenzbilder für die Datenbank im wesentlichen Profilansichten und Abbildungen mit leicht erhöhter Perspektive von schräg vorne verwendet.

[8]Eine Projektion ist dann stabil, wenn sie ähnlich zu leicht gedrehten Projektionen desselben Objekts ist [10]. Eine stabile Projektionen tritt dann auf, wenn die entsprechende Abbildung fehlertolerant ist.

Abbildung 2.4: Beispiele für die Komplexität der Segmentierung und Objekterkennung. Ein Mensch erkennt, dass es sich bei dieser Abbildung (entnommen aus [32]) um die drei Objekte *Ventilator*, *Pflanze* und *Stift* handelt, obwohl sich die Objekte gegenseitig verdecken, sich die Helligkeit und die Perspektive in den einzelnen Bildern ändert und die Objekte unterschiedlich angeordnet werden.

Bei der Analyse der Bildsequenzen der Testdaten ist zu untersuchen, inwieweit kanonische Sichten in Bildfolgen verstärkt auftreten und wie gut sie im Vergleich zu nicht-kanonischen Sichten erkannt werden können.

2.4 Besondere Fähigkeiten der visuellen Wahrnehmung

Der Mensch besitzt die Fähigkeit, sehr komplexe Situationen zu erfassen (vgl. Abb. 2.4). Hierzu gehört die Erkennung von mehreren sich teilweise gegenseitig verdeckenden Objekten, die Erkennung von Objekten aus ungewohnten Perspektiven und sich ändernden Lichtverhältnissen. Auch Schatten, Texturen und Spiegelungen auf der Oberfläche [19], sowie die Kombination dieser Faktoren beeinflussen die Objekterkennung. Ein Mensch kann Objekte in komplexen Bildern ohne diese Faktoren nicht immer identifizieren, wohingegen in der rechnergestützten Bilderkennung die Einbeziehung aller Faktoren das Problem zu komplex werden lässt. Da die menschliche Wahrnehmung sehr effizient funktioniert, soll sie der rechnergestützten Bilderkennung als Vorlage dienen, wobei die Anzahl der zu analysierenden Faktoren zugunsten der Komplexität deutlich eingeschränkt wird.

Unter der Annahme eines fixen Beobachters, können Objekte in allen drei Dimensionen gedreht werden. Einem Menschen fällt es deutlich leichter, eine Rotation eines bekannten Objekts um die z-Achse zu erkennen als eine Rotation um die x- oder y-Achse, selbst wenn zu dieser Rotation zusätzlich leichte Deformationen des Objekts auftreten [5]. Da die Form der Kontur bei dieser Drehung, abgesehen von minimalen Änderungen durch unterschiedliche Lichtverhältnisse, erhalten bleibt, kann eine Rotation um die z-Achse auch in der rechnergestützten Bilderkennung problemlos nachvollzogen werden (vgl. Abb. 2.5 auf der nächsten Seite). Im Rahmen dieser Arbeit hat eine Rotation um die z-Achse keinen Einfluss auf die Qualität oder Geschwin-

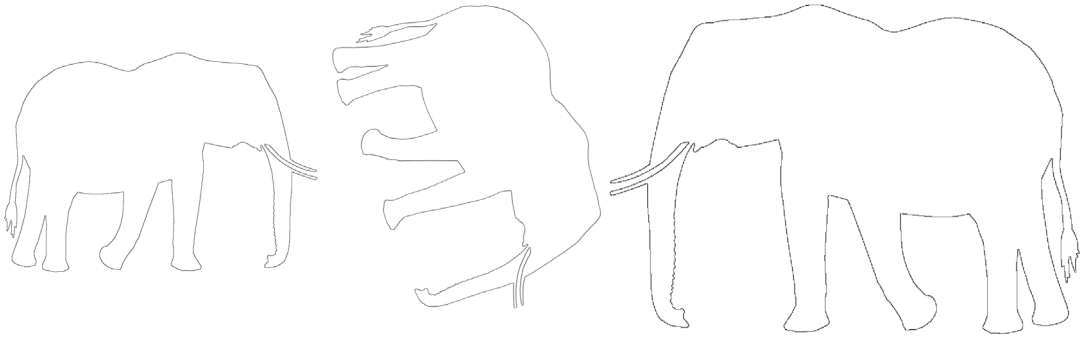

Abbildung 2.5: Einfache Transformation der Kontur eines Objekts. Trotz Rotation, Spiegelung und Zoom, kann der Elefant in allen drei Abbildungen leicht erkannt werden. Links ist die Originalkontur abgebildet, daneben eine Drehung im Uhrzeigersinn um die z-Achse von 70 Grad. Rechts ist die Kontur an der x-Achse gespiegelt und zusätzlich um 50 Prozent näher gezoomt worden. Alle Abbildungen kann ein Beobachter sicher identifizieren, auch wenn ein Betrachter für das Erkennen des mittleren Bildes die meiste Zeit benötigt.

digkeit der rechnergesteuerten Objekterkennung, ebenso wie eine Spiegelung des Objekts an der x- oder y-Achse.

Das Gehirn des Menschen kann bei der Objekterkennung sehr unterschiedliche Aufgaben erfüllen, von der Zuordnung zu einer Objektklasse bis hin zur Identifikation eines konkreten Objekts. Die rechnergestützte Bildverarbeitung bietet lediglich in speziellen Teilgebieten zufriedenstellende Lösungen an. Hochspezialisierte Lösungsansätze, wie beispielsweise die Gesichtserkennung, können üblicherweise nicht in anderen Bereichen der elektronischen Bildanalyse eingesetzt werden.

Die Objekterkennung eines Menschen muss in der rechnergesteuerten Bildanalyse durch eine Vielzahl sequenzieller Schritte nachgebildet werden. Neben der eigentlichen *Identifikation der Objekte* ist der *Segmentierungsprozess* von entscheidender Bedeutung. Beide Problembereiche können beim Menschen nicht voneinander getrennt betrachtet werden. Bei mehreren sich gegenseitig teilweise verdeckenden Objekten, wird schon während der Segmentierung eine Objekterkennung durchgeführt. Sobald ein Objekt erkannt wurde, wird die Segmentierung gegebenenfalls angepasst und die übrigen Objekte werden im folgenden leichter erkannt [8]. In vorhandenen Computeralgorithmen werden beide Problembereiche größtenteils getrennt betrachtet und sequenziell gelöst.

Der Problembereich der *Objektsegmentierung* wird in dieser Arbeit ausgeklammert. Es wird von der Annahme ausgegangen, dass die zu analysierenden Bildsequenzen nur *leicht segmentierbare Objekte* beinhalten. Ein Objekt zählt in dieser Arbeit als leicht segmentierbar, wenn die Kontur des Objekts anhand von Farbdifferenzen ermittelt werden kann. Das gesuchte Objekt

muss sich deutlich vom Hintergrund abheben. In vielen Bildern werden Objekte durch andere Objekte verdeckt. Während ein Mensch häufig die Vorstellung hat, wie die verdeckten Teile des Objekts aussehen, ist eine korrekte Segmentierung teilweise verdeckter Objekte nicht möglich. Aus dem möglicherweise fehlerhaften Segmentierungsschritt kann ein stark deformiertes Objekt resultieren.

Einfache Transformationen des Objekts, wie die oben beschriebene Rotation um die z-Achse, die Spiegelung an der x- oder y-Achse, die Verschiebung oder der Zoom, bereiten für die spätere Objekterkennung keine zusätzlichen Schwierigkeiten. Nach dem Segmentierungsprozess wird die Kontur in einer speziellen Weise, die invariant gegenüber den obigen Transformationen ist, parametrisiert.

Das Ziel dieser Arbeit soll eine zuverlässige und stabile Objekterkennung sein. Die Fehlerquote bei der Erkennung von verrauschten oder verzerrten Objekten steigt bei vielen Bildanalysealgorithmen im Vergleich zur menschlichen Wahrnehmung überproportional stark an. Inwieweit die Ergebnisse dieser Arbeit den Ansprüchen eines stabilen und robusten Verfahrens zur Bildanalyse gerecht werden können, wird im Laufe der Arbeit kritisch hinterfragt werden.

2.5 Reduktion der Bilddaten auf die Kontur

Unabhängig, ob beim Menschen von einer dreidimensionalen Speicherung eines Objekts oder von mehreren zweidimensionalen Abbildungen, ergänzt durch Tiefeninformationen, ausgegangen wird, handelt es sich bei diesen Daten um eine Fülle von Informationen, die durch heutige Rechnersysteme nicht in akzeptabler Zeit erfassbar und verarbeitbar sind.

Das in dieser Arbeit vorgestellte Verfahren verwendet als Datenbasis Farb- oder Graustufenbilder, die nach dem Segmentierungsschritt in Binärbilder[9] umgewandelt werden. Nach dieser deutlichen Datenreduktion bleibt lediglich die gefüllte Kontur des Objekts erhalten. Alle zusätzlichen Informationen gehen verloren, so beispielsweise die Oberflächenbeschaffenheit, Textur und auch die Orientierung[10] des Objekts in seiner Umgebung. Die Begriffe *Oben* oder *Unten* verlieren somit jede Bedeutung. Diese Unabhängigkeit von der Umgebung bietet natürlich auch Vorteile. So bewirkt eine Drehung eines Objekts keine Änderung der Ergebnisse im Bildanalyseprozess, wohingegen Bildrotationen den Erkennungsprozess beim Menschen geringfügig erschweren (vgl. Abb. 2.5 auf der vorherigen Seite).

Die Komplexität des Objekts wird in einem Farb- oder Graustufenbild bewusst reduziert. Viele Objekte kann ein Mensch jedoch problemlos anhand ihrer Konturen identifizieren. Ein nicht zu vernachlässigendes Problem ergibt sich aus der Tatsache, dass auch ein Mensch nicht jedes Objekt ausschließlich anhand seiner Kontur identifizieren kann. In Abbildung 2.3 auf Seite 9 kann das ganz rechts abgebildete Objekt praktisch nicht erkannt werden. In einer Untersu-

[9] Die einzelnen Bildpunkte in Binärbildern können die Zustände Null oder Eins annehmen, in Graustufenbildern sind üblicherweise 256 unterschiedliche Werte möglich.

[10] Die Orientierung im Raum ist für den Menschen zur Erkennung von Objekten von essenzieller Bedeutung [8].

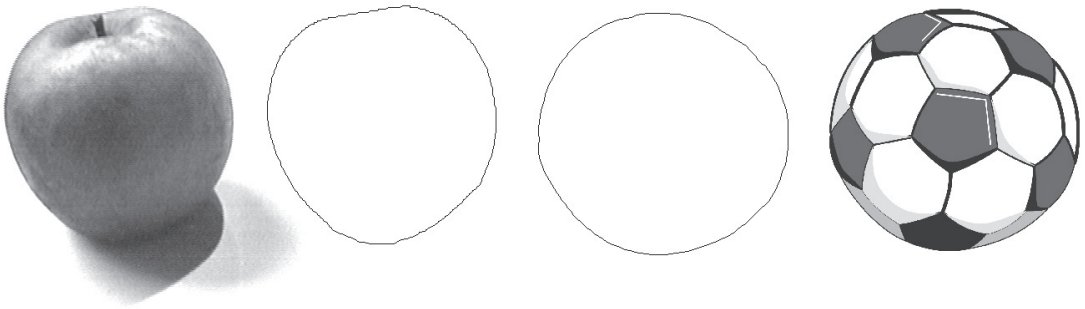

Abbildung 2.6: Beispiel für die Reduktion von Graustufenbildern auf ihre Kontur. Die Kontur des Apfels und Fußballs wurde so ermittelt, wie sie im Segmentierungsprozess in dieser Arbeit erfolgt. Die Kontur bietet bei dieser Ansicht nicht mehr ausreichend Information, um eine Objekterkennung zu ermöglichen [35].

chung[11] wurde gezeigt, dass alleine durch die Schattierung der Oberfläche eine deutlich bessere Objekterkennung für den Menschen möglich wird. Ein Mensch erkennt kugelförmige Objekte maßgeblich anhand der Oberflächenstruktur (vgl. Abb. 2.6). Die Objektgröße spielt bei Abbildungen eine untergeordnete Rolle, da ein absoluter Maßstab nicht immer erkennbar ist. Die Struktur der Oberfläche ist von entscheidender Bedeutung und vermittelt häufig eine eindeutige Assoziation mit einem Objekt. Voraussichtlich wird auch die rechnergesteuerte Bildanalyse schnell an ihre Grenzen stoßen, wenn sie nur Informationen, die aus der Kontur ableitbar sind, berücksichtigt.

Besonders in Bildsequenzen schwächen sich obige Erkennungsprobleme sowohl beim Menschen [35] als auch bei der rechnergestützten Analyse ab. Indem mehrere Bilder eines Objekts in unterschiedlichen Ansichten betrachtet werden, steigt die Möglichkeit der Wiedererkennung für einen Menschen deutlich an. Auch die Reihenfolge der gezeigten Bilder des Objekts spielt hierbei eine entscheidende Rolle. Die Erkennung von Objekten wird leichter, wenn sich sowohl die Perspektive als auch die Form des Objekts im Zeitablauf ändert.

Ein nicht zu unterschätzender Vorteil des Menschen liegt in seiner Fähigkeit, auf vorhandenes Wissen zurückzugreifen und zu lernen. Bei jeder Betrachtung eines Objekts aus einer unbekannten Perspektive steigt sein Wissen über dieses Objekt, gegebenenfalls auch über diese neuartige Perspektive. Die Vorteile der Analyse von Bildsequenzen im Rechner liegen in der Möglichkeit, viele Daten präzise zu speichern und einfache Änderungen des Bildes, wie Rotations- oder Zoomeffekte, nicht explizit berücksichtigen zu müssen. Einfache Merkmale zweier Objekte

[11] In [19] wird die Bedeutung von Schattierungen für die Objekterkennung verdeutlicht. Kanonische Sichten eines Objekts werden mit Schattierung signifikant schneller erkannt, als eine reine Darstellung der Kontur. Nur reine Profilansichten, beispielsweise ein Tier oder Fahrzeug von der Seite, ermöglichen einem Menschen einen gleich schnellen Erkennungsprozess.

können im Rechner sehr schnell miteinander verglichen werden.

Im folgenden Kapitel werden zunächst die grundlegenden Verfahren zur Bildaufbereitung erläutert. Diese Verfahren werden in Kapitel 5 angewendet, um unter besonderer Berücksichtigung der kanonischen Sichten eine Objekterkennung in Bildsequenzen durchzuführen.

Kapitel 3

Grundlagen der Bildverarbeitung

In dem folgenden Kapitel werden die grundlegenden Begriffe und Verfahren zur Bildverarbeitung vorgestellt, die im Rahmen dieser Arbeit benötigt werden. Um Objekte erkennen zu können, muss in einem ersten Segmentierungsschritt deren Kontur ermittelt werden. Mit Hilfe der Krümmung einer Kontur, also der Angabe, ob ein Kurvenstück konkav oder konvex gekrümmt ist, wird ein praktikabler Vergleich zweier Konturen überhaupt erst möglich. Aus diesem Grund beschäftigt sich ein umfangreicher Teil dieses Kapitels mit dem Begriff der Krümmung.

Abschließend werden zwei wichtige Filterklassen vorgestellt. Der Gaußfilter dient zur Glättung einer Wertemenge oder eines Bildes und wird speziell zur Glättung der Kontur verwendet. Die zweite wichtige Gruppe sind die morphologischen Filter. Zu dieser Gruppe zählt der *Closing-Filter*, der als Beispiel für die generelle Vorgehensweise konturvereinfachender Verfahren dient. Auf den Zusammenhang der morphologischen und krümmungsbasierten Verfahren wird in Kapitel 4 eingegangen.

3.1 Segmentierung

Um Objekte in der rechnergestützten Bildverarbeitung erkennen zu können, müssen zunächst die Grenzen eines Objekts ermittelt werden. Die Grenze zwischen zwei Objekten bzw. einem Objekt und dem Hintergrund des Bildes, bestimmt die Form des Objekts. Ein Bildverarbeitungssystem kann bei mehreren Objekten nur dann erkennen, welches ausgewählt werden soll, wenn entweder weitere Informationen über das gesuchte Objekt zur Verfügung stehen, oder wenn in dem zu untersuchenden Bild nur ein dominierendes Objekt existiert.

Die beiden Schritte *Segmentierung* und *Objekterkennung* sind rechnergesteuert schwer zu bewältigen, da für den Segmentierungsschritt Informationen über die Form des gesuchten Objekts benötigt werden, und vor der Inhaltsanalyse ein Segmentierungsschritt erfolgen muß. Um die Komplexität der Segmentierung zu reduzieren, wird die Annahme getroffen, dass in den zu analysierenden Bildern bzw. Bildsequenzen genau ein *leicht segmentierbares Objekt* abgebildet wird. Leicht segmentierbar bedeutet in diesem Zusammenhang, dass sich die Farbe des Objekts

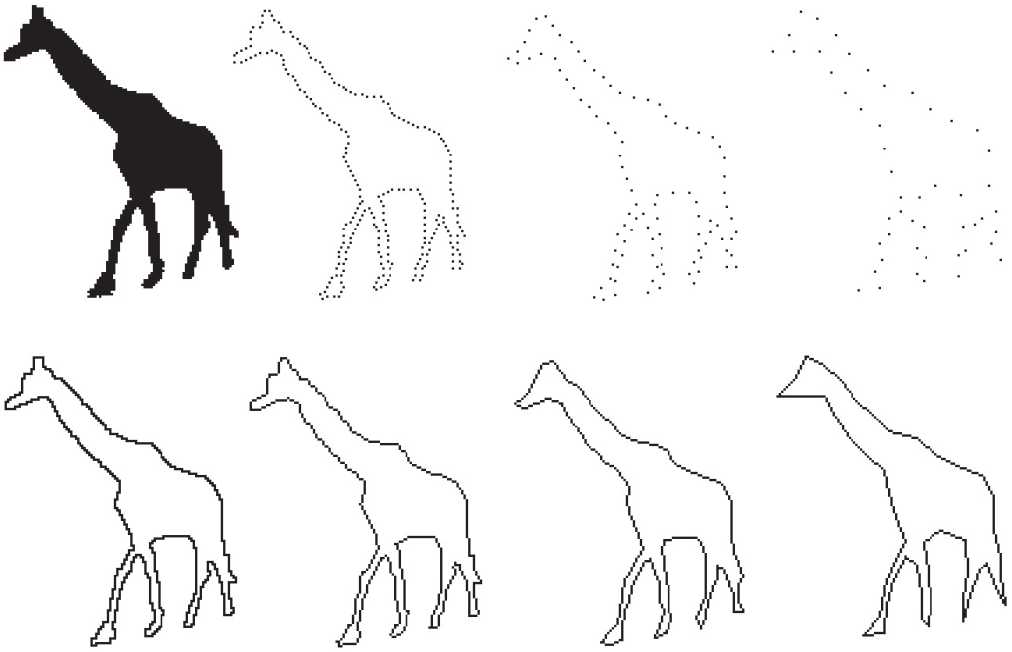

Abbildung 3.1: Bedeutung der Anzahl der Konturpixel. Links ist das Originalbild einer Giraffe mit der zugehörigen Kontur abgebildet. Daneben wird in der oberen Zeile die Kontur durch 250, 100 und 50 Wertepaare wiedergegeben. Durch das Verbinden benachbarter Wertepaare durch Geraden ergeben sich in der unteren Zeile geschlossene Konturen. Bei 50 Wertepaaren werden deutliche Differenzen zur Originalkontur sichtbar, bei 250 Wertepaaren sind diese kaum noch wahrnehmbar.

deutlich von der Hintergrundfarbe unterscheidet.

Die Größe eines Objekts spielt dann eine Rolle, wenn mehrere mögliche Objekte in einem ersten Segmentierungsschritt gefunden werden. In diesem Fall wird das Objekt ausgewählt, das die längste Kontur aufweist. Alle anderen Objekte werden aus dem Bild entfernt.

Für die weiteren Analyseschritte wird nur die Kontur des Objekts verwendet. Im nächsten Schritt folgt daher eine *Parametrisierung der Kontur*, die – im Vergleich zur Darstellung als Bild – lediglich die Punkte der Kontur als Wertepaare speichert. Diese Wertepaare sind, ausgehend von einem beliebigen Punkt der Kontur, so sortiert, dass im Uhrzeigersinn nebeneinanderliegende Konturpixel in einer aufsteigenden Liste gespeichert werden.

Da die Abbildungen eines Objekts, einmal aus der Nähe und einmal aus der Ferne betrachtet, durch unterschiedlich viele Wertepaare repräsentiert werden, müssen die Konturen *normalisiert* werden. Nach der Normalisierung wird jedes Objekt durch eine gleiche Anzahl von Konturpixeln dargestellt. Details bezüglich des Segmentierungsschrittes und zur Ermittlung der parametrisierten Kontur folgen in Kapitel 4 und 5.

3.2 Komplexität von Konturen

Bestimmte Merkmale eines Bildes, wie z. B. die Farbe lassen sich üblicherweise durch wenige Parameter exakt spezifizieren. Eine Kontur ist dagegen nicht mit wenigen Daten charakterisierbar. Neben der Möglichkeit, alle Koordinaten als Pixelpaare zu speichern, gibt es verschiedene Näherungsverfahren zur Beschreibung einer Kontur. Beispielsweise können mit Hilfe abschnittsweise definierter Polynome [21] Teilstücke der Kontur beschrieben werden.

Nach der Erfassung der Kontur beginnt die eigentliche Aufgabe der rechnergestützten Bildverarbeitung. Vergleiche mit anderen Konturen sollen anhand der ermittelten Konturdaten durchgeführt werden. Außerdem sollen Unterschiede und Ähnlichkeiten zweier Konturen bewertet werden. Diese Wertdifferenz beschreibt, ob mehr oder weniger stark differierende Konturen vorliegen, die dann auf unterschiedliche oder ähnliche Objekte hinweisen.

Nachfolgend wird ausgeführt, wie eine Kontur durch ihre Krümmung charakterisiert werden kann. Jedes Teilstück einer Kontur lässt sich als Gerade oder als konkav oder konvex gekrümmtes Kurvenstück beschreiben. Die Stärke der Krümmung kann in jedem einzelnen Punkt der Kontur berechnet werden. Besonders lange und stark in eine Richtung gekrümmte Bereiche sollen wesentliche Informationen über die Kontur des Objekts liefern.

Der folgende Abschnitt erläutert, wie Krümmungen in Bildern berechnet werden können.

3.3 Krümmung von Konturen

Die *Krümmung* in einem Punkt einer Funktion gibt an, wie stark die Tangente in diesem Punkt von Tangenten, die durch Punkte in der näheren Umgebung gehen, abweichen. Eine starke Krümmung liegt dann vor, wenn der Winkel zwischen diesen Tangenten groß ist. Bei einer Geraden stimmen alle Tangenten einer Funktion mit dieser überein, d.h. eine Gerade hat eine Krümmung von Null. In Abbildung 3.2 auf der nächsten Seite ist eine Funktion mit mehreren Tangenten abgebildet, die die Herleitung der Krümmung verdeutlichen.

Die Richtung der Abweichung der Tangente von der Funktion bestimmt, ob diese konvex oder konkav gekrümmt ist, bzw. ob eine Links- oder Rechtskrümmung vorliegt. Bei der linken Tangente in Abbildung 3.2 auf der nächsten Seite liegt eine Rechtskrümmung, bei der zweiten Tangente eine starke Linkskrümmung vor.

Bei geschlossenen Konturen handelt es sich um zweidimensionale Funktionen. Existiert unter allen Tangenten, die an die Kontur gelegt werden können, mindestens eine, die mit der Kontur mehr als einen Punkt gemeinsam hat, so liegt eine *nicht konvexe Kontur* vor [6]. Ein Ball oder ein Ei besitzt üblicherweise eine konvexe Kontur. In Abbildung 3.3 auf der nächsten Seite ist eine konvexe und eine nicht konvexe Kontur abgebildet.

Neben der gesammten Kontur können auch einzelne Bereiche konkav oder konvex sein. Wenn die Orientierung der Kontur im Uhrzeigersinn festgelegt wird, dann entsprechen Rechtskrümmungen konvexen Bereichen und Linkskrümmungen konkaven. In jeder Kontur eines rea-

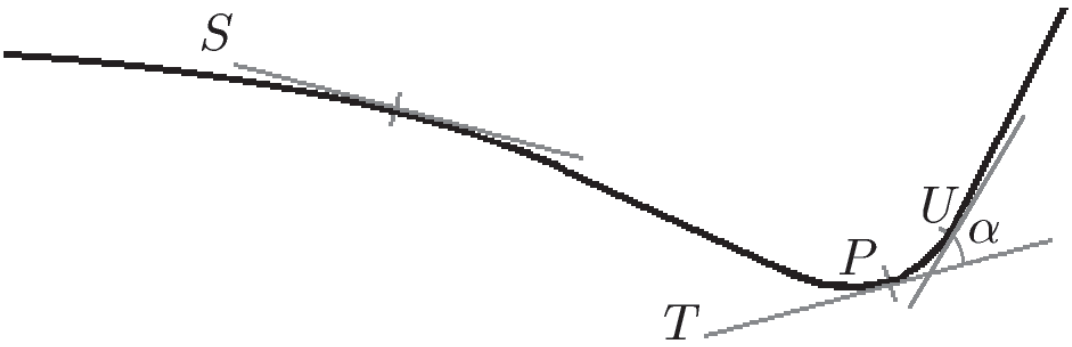

Abbildung 3.2: Darstellung einer Funktion mit zwei Tangenten *S* und *T*. An der linken Tangente hat die Funktion eine geringere Krümmung als an der rechten Tangente. Auch die Vorzeichen der Krümmungen unterscheiden sich. An einem Beispiel soll die Ermittlung der betragsmäßigen Werte der Krümmung verdeutlicht werden. Ausgehend von einem Punkt *P* wird in einer lokalen Umgebung ein Punkt *U* gewählt. Der Winkel α, der durch die beiden Tangenten in den Punkten *P* und *U* eingeschlossen wird, ermöglicht eine Berechnung des betragsmäßigen Wertes der Krümmung [12, 39].

Abbildung 3.3: Konvexe und teilweise konkave Konturen. Bei der linken Abbildung handelt es sich um eine konvexe Kontur, da keine Tangente mit der Kontur in mehr als einem Punkt übereinstimmt. Rechts ist eine nicht konvexe Kontur abgebildet.

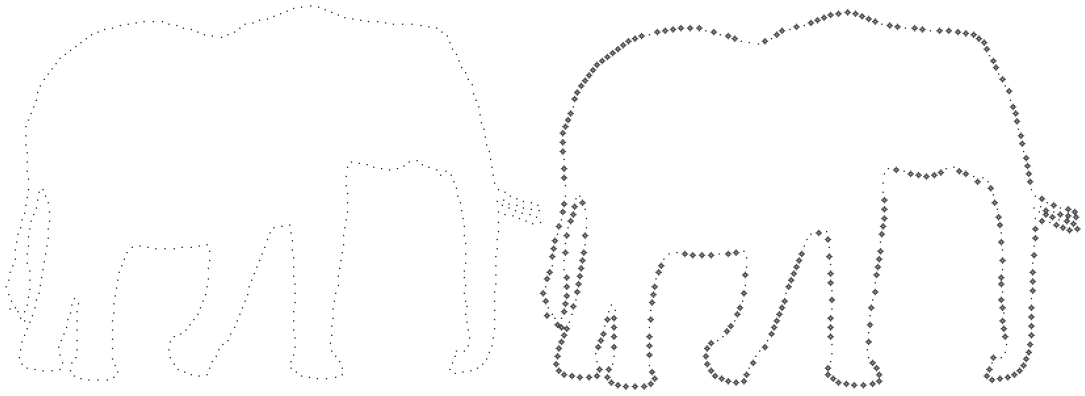

Abbildung 3.4: Konkave und konvexe Bereiche einer Kontur. Im linken Bild wird die Kontur eines Elefanten durch 400 Punkte abgebildet. Die hervorgehobenen Punkte sind konvexe Bereiche und entsprechen lokalen Rechtskrümmungen.

len Objekts müssen konvexe Bereiche auftreten, da sonst keine geschlossene Form des Objekts vorliegen kann. Konkave und konvexe Bereiche einer Kontur werden in Abbildung 3.4 an einer parametrisierten Kontur dargestellt, wobei die konvexen Bereiche hervorgehoben wurden.

Anschaulicher lässt sich die Krümmung über den Radius eines Kreises definieren (vgl. Abb. 3.5 auf der nächsten Seite). In einem gesuchten Punkt wird ein Kreis so an die Funktion gelegt, dass dieser Kreis in der Umgebung mit den Funktionswerten übereinstimmt. Aus der Größe des Radius r lässt sich anhand der Gleichung 3.1 der betragsmäßige Wert der Krümmung in diesem Punkt berechnen [18, 34].

$$|\kappa| = \frac{1}{r} \tag{3.1}$$

Das Vorzeichen der Krümmung hängt davon ab, ob der Mittelpunkt des Kreises unterhalb oder oberhalb der Funktion liegt. Abbildung 3.5 auf der nächsten Seite verdeutlicht den Zusammenhang zwischen der Krümmung und dem Radius eines an die Funktion gelegten Kreises. Diese zweite Definition ist intuitiv leichter verständlich, für die praktische Berechnung der Krümmung jedoch weniger geeignet.

Um in Grauwertbildern Krümmungen berechnen zu können, werden Isophoten, d. h. Lienien gleicher Grauwerte betrachtet. Die Berechnung der Krümmung erfolgt für die einzelnen Isophoten und entspricht der Berechnung der Krümmung in Binärbildern.

3.4 Berechnung der Krümmung

Die Krümmung einer Funktion lässt sich mit Hilfe ihrer ersten und zweiten Ableitungen in x- und y-Richtung berechnen.

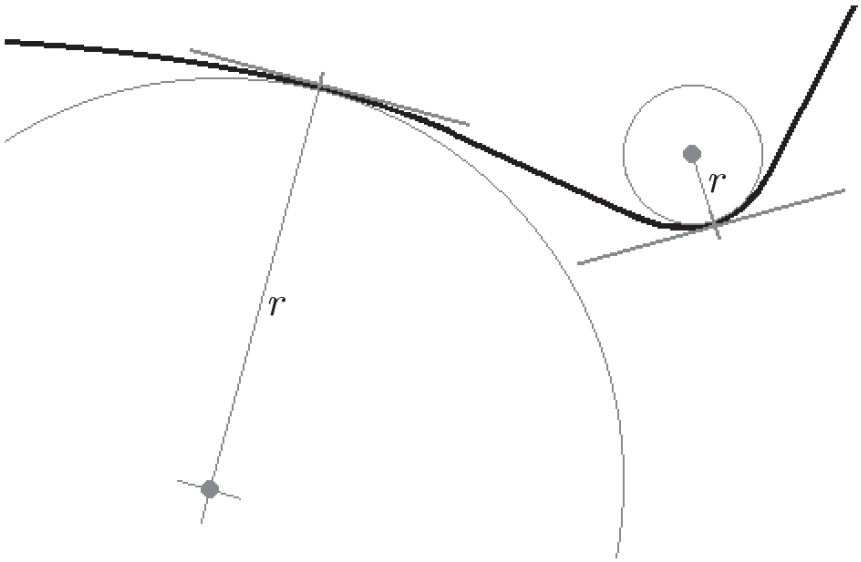

Abbildung 3.5: Zusammenhang zwischen dem Radius *r* eines Kreises, der eine Funktion in möglichst vielen benachbarten Punkten berührt, und der Stärke der Krümmung der Funktion (vgl. Formel 3.1 auf der vorherigen Seite). Die Position des Kreises bestimmt, ob eine Links- oder Rechtskrümmung vorliegt.

Abbildung 3.6: Beispiel für die grobe Schätzung dreier Tangenten mit Hilfe der Sekante durch zwei benachbarte Punkte. Abhängig von den Abständen der Pixel untereinander treten unterschiedlich starke Fehler bei der Bestimmung der Tangente auf. Die Kontur des abgebildeten Fischs wird nur durch 150 Punkte bestimmt. Die unterschiedlich großen Abstände der Punkte voneinander können den Fehler bei der Bestimmung der Tangente zusätzlich erhöhen.

Die *ersten Ableitungen* einer Funktion werden mit $\dot{x}$ und $\dot{y}$ bezeichnet. Stetige Funktionen stehen in der rechnergestützten Bildverarbeitung üblicherweise nicht zur Verfügung, da Bilder aus einer festen Anzahl diskreter Pixel bestehen. Die Steigung in einem Punkt muss im diskreten Fall durch Näherungsverfahren ermittelt werden. Sie lässt sich durch eine Sekante, die durch zwei benachbarte Punkte bestimmt wird, annähern. In Abbildung 3.6 auf der vorherigen Seite wird am Beispiel einer diskretisierten Kontur die Näherung der Tangente mit Hilfe der Sekante durch zwei benachbarte Punkte veranschaulicht. Es handelt sich in dieser Abbildung jedoch nicht um die Ableitungen $\dot{x}$ oder $\dot{y}$. Die Berechnung der Ableitungen in die x- und y-Richtung erfolgt in ähnlicher Weise.

In diskreten Bildern besitzt jeder Bildpunkt $P_{i,j}$ einen festen Wert. In Graustufenbildern liegt dieser im Bereich $[0,255]$. Die Parameter i und j bestimmen im Bild die aktuelle Spalte und Zeile des Bildpunktes. In einem Punkt $P_{i,j}$ berechnen sich die ersten Ableitungen für diskrete Bilder wie folgt.

$$\begin{aligned} \dot{x}_{i,j} &= \frac{P_{i+1,j} - P_{i-1,j}}{2 \cdot hx} \\ \dot{y}_{i,j} &= \frac{P_{i,j+1} - P_{i,j-1}}{2 \cdot hy} \end{aligned} \tag{3.2}$$

Unter hx und hy wird der Abstand vom zentralen Punkt zu einem Nachbarn in x- bzw. y-Richtung verstanden. Die Abstände aller benachbarter Pixel im Bild sollen gleich groß sein, so dass zwischen den Punkten $P_{i+1,j}$ und $P_{i-1,j}$ ein Abstand von $2 \cdot hx$ besteht.

Es ist wünschenswert zur Berechnung der zweiten Ableitungen nur die Werte der direkten Nachbarpunkte zu verwenden. Daher wird für die Herleitung der zweiten Ableitung in x-Richtung die erste Ableitung in den Punkten $P_{i-\frac{1}{2},j}$ und $P_{i+\frac{1}{2},j}$ in der Umgebung von jeweils einem halben Pixel berechnet.

Für die Ableitung in x-Richtung ergeben sich in den Punkten $P_{i-\frac{1}{2},j}$ und $P_{i+\frac{1}{2},j}$ in der Umgebung von $\frac{1}{2}$ die Gleichungen:

$$\begin{aligned} \dot{x}_{i-\frac{1}{2},j} &= \frac{P_{i,j} - P_{i-1,j}}{2 \cdot 0.5 \cdot hx} = \frac{P_{i,j} - P_{i-1,j}}{hx} \\ \dot{x}_{i+\frac{1}{2},j} &= \frac{P_{i+1,j} - P_{i,j}}{hx} \end{aligned} \tag{3.3}$$

Aus diesen neuen ersten Ableitungen (Formel 3.3) lässt sich die *zweite Ableitung* $\ddot{x}$ in x-Richtung im Punkt $P_{i,j}$ für eine Umgebung von $\frac{1}{2}$ berechnen. Die Berechnung von $\ddot{y}$ erfolgt analog.

$$\begin{aligned}\ddot{x}_{i,j} &= \frac{\dot{x}_{i+\frac{1}{2},j} - \dot{x}_{i-\frac{1}{2},j}}{hx} \\ &= \frac{\frac{P_{i+1,j}-P_{i,j}}{hx} - \frac{P_{i,j}-P_{i-1,j}}{hx}}{hx} \\ &= \frac{P_{i+1,j} - 2\cdot P_{i,j} + P_{i-1,j}}{hx \cdot hx} \end{aligned} \tag{3.4}$$

Bei zweidimensionalen Funktionen ist die Ableitung in xy-Richtung $\dot{xy}$ ebenfalls relevant. Eine Standardapproximation kann wie folgt berechnet werden [37].

$$\dot{xy}_{i,j} = \frac{(P_{i+1,j-1} - P_{i-1,j-1}) + (P_{i-1,j} + 1 - P_{i+1,j+1})}{4 \cdot hx \cdot hy} \tag{3.5}$$

Eine leichter zu erfassende Schreibweise für Gleichung 3.5 ist in Form der Matrix M, bei der das mittlere Element den Punkt $P_{i,j}$ repräsentiert.

$$M = \frac{1}{4 \cdot hx \cdot hy} \begin{pmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{pmatrix} \tag{3.6}$$

Gleichung 3.6 liefert in konkreten Berechnungen keine stabilen Ergebnisse. Eine zur Berechnung besser geeignete Approximation ist in Gleichung 3.7 gegeben[1].

$$M = \frac{1}{2 \cdot hx \cdot hy} \begin{cases} \begin{pmatrix} 0 & -1 & 1 \\ -1 & 2 & -1 \\ 1 & -1 & 0 \end{pmatrix} & \text{für alle } \dot{x} \cdot \dot{y} < 0; \\ \begin{pmatrix} -1 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -1 \end{pmatrix} & \text{sonst} \end{cases} \tag{3.7}$$

Nachdem alle benötigten Ableitungen durch Interpolation ermittelt werden können, lässt sich die *Krümmung*[2] $\kappa_{i,j}$ im Punkt $P_{i,j}$ berechnen. Da die Krümmung immer in einem Punkt $P_{i,j}$ berechnet wird, soll im folgenden die Schreibweise für die Ableitungen $\dot{x}_{i,j}$ zu $\dot{x}$ vereinfacht werden.

[1] In [37] werden die Gleichungen 3.6 und 3.7 vorgestellt.

[2] Zum Begriff der Krümmung vergleiche auch [36, 37]

$$\kappa = \frac{\ddot{x} \cdot \dot{y}^2 - 2 \cdot \dot{x} \cdot \dot{y} \cdot \dot{x}\dot{y} + \ddot{y} \cdot \dot{x}^2}{\dot{x}^2 + \dot{y}^2} \tag{3.8}$$

Mit dieser Formel [36] lässt sich die Krümmung für jeden Punkt einer zweidimensionalen Funktion berechnen.

3.5 Bildverarbeitung durch Filter

Ein *Filter* im Bereich der Bildverarbeitung ist ein algorithmisches Verfahren, das ein Bild in einer speziellen Weise manipuliert, um gewünschte Effekte zu erhalten. Filter können ein Bild glätten und dabei Rauschen entfernen. Andere Filter heben bestimmte Merkmale, wie spezielle Formen oder Farben, in einem Bild hervor. Typische Filter dienen der Glättung der Bilddaten, der Kantenverstärkung oder manipulieren die Form von Objekten.

Zwei Arten von Filtern sind in dieser Arbeit von besonderer Bedeutung, Gaußfilter und konturverändernde Filter. Zu dieser zweiten Kategorie gehören morphologische Filter, die am Ende dieses Kapitels vorgestellt werden.

3.5.1 Gaußfilter

Ein *Gaußfilter* ist ein spezieller Tiefpassfilter [15], der eine Glättung eines Bildes bewirkt und Rauschen entfernt.

Eine Maske wird über jedes Pixel gelegt, so dass sich der neue Wert aus der Summe der Maskenwerte multipliziert mit den entsprechenden Werten der Pixel im Originalbild berechnet. In Abbildung 3.7 auf der nächsten Seite wird der Gaußfilter mit einer Maskenbreite von 25 angewendet.

Es gibt zwei Methoden, um die Gaußmaske zu erstellen und mit Werten zu füllen. Die Breite der Maske kann durch Angabe einer ungeraden ganzen Zahl spezifiziert werden. Die Elemente werden anschließend mit einer Binomialverteilung gefüllt. Die zweite Methode erwartet die Angabe der Variablen σ und berechnet die Maskenwerte mit einer spezifizierten Genauigkeit anhand der Gaußverteilung.

Ein Beispiel soll die Berechnung eines Gaußfilters über die Maskenbreite verdeutlichen. Zur Vereinfachung erfolgt eine Glättung in eindimensionaler Richtung. Die Maske M mit einer Breite von 3 wird zunächst mit der Binomialverteilung gefüllt.

$$\begin{aligned} M &= (m_i) \\ &= \begin{pmatrix} 1 & 2 & 1 \end{pmatrix} \text{ mit } i = 1 \ldots 3 \end{aligned} \tag{3.9}$$

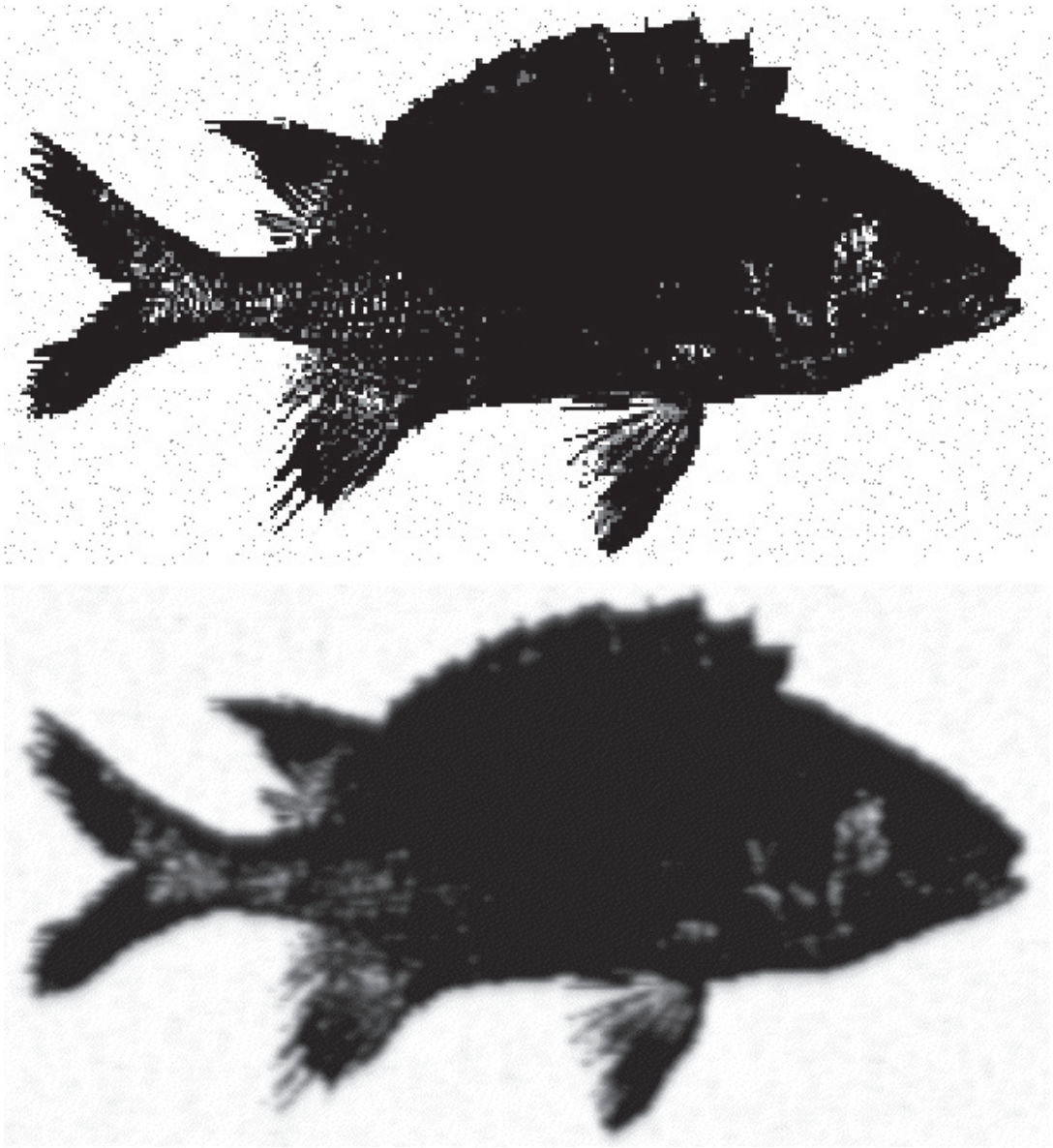

Abbildung 3.7: Anwendung eines Gaußfilters auf ein verrauschtes Bild. Oben ist das leicht verrauschte Bild eines Fisches abgebildet. Unten wird das Bild nach einer Gaußglättungen mit der Maskengröße von 25 Pixeln gezeigt. Das Rauschen im Bild verschwindet. Die Kontur wird jedoch unschärfer.

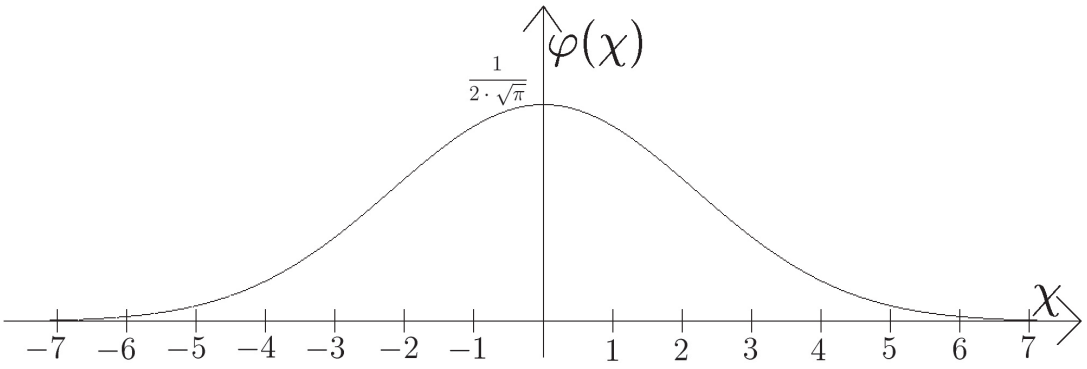

Abbildung 3.8: Gaußsche Glockenkurve im Bereich von $-7 \leq \chi \leq +7$ bei einer Standardabweichung von $\sigma = 2$. Die Maskenwerte des Gaußfilters werden anhand dieser Verteilung berechnet und anschließend normiert.

Da sich die durchschnittliche Helligkeit des Originalbildes nach Anwendung eines Gaußfilters nicht ändern soll, wird die Maske normiert, so dass die Summe der Maskenelemente Eins ergibt. Als Normierungsfaktor ergibt sich $1/\sum m_i = \frac{1}{4}$. Die normierte Gaußmaske hat dann die Werte:

$$\begin{aligned} M_{normiert} &= \frac{1}{\sum m_i}(m_i) \\ &= \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 \end{pmatrix} \\ &= \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{pmatrix} \end{aligned} \tag{3.10}$$

Zur Veranschaulichung sollen die Daten $B = (b_j) = (5,1,4,3,6,\ldots)$ mit $j = 1\ldots$ gegeben sein. Der neue Wert an der Position $\bar{b}_3$ berechnet sich wie folgt:

$$\begin{aligned} \bar{b}_2 &= m_1 \cdot b_2 + m_2 \cdot b_3 + m_3 \cdot b_4 \\ &= \frac{1}{4} \cdot 1 + \frac{1}{2} \cdot 4 + \frac{1}{4} \cdot 3 \\ &= 3 \end{aligned} \tag{3.11}$$

Neben einer Approximation mit Hilfe einer Binomialverteilung können die Werte der Maske direkt mit Hilfe der Gaußverteilung (vgl. Abbildung 3.8) berechnet werden.

Diese Verteilung berechnet sich nach der Formel 3.12. Für die konkrete Berechnung der Maske wird lediglich die Standardabweichnung σ und ein Parameter für die Genauigkeit benötigt. Die Standardabweichung gibt an, wie dicht die Verteilung ist, d.h. wie weit die Werte um den Erwartungswert μ schwanken. Der Erwartungswert kann zur Berechnung der Maske durch Null ersetzt werden.

$$\varphi(\chi) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{\frac{(\chi-\mu)^2}{2\cdot\sigma^2}} \tag{3.12}$$

Bei der Berechnung mit Hilfe der Binomialverteilung wird als Parameter lediglich die Maskenbreite benötigt. Das zusätzliche Maß für die Genauigkeit entfällt. In dieser Arbeit wird der Gaußfilter über die Maskenbreite spezifiziert.

Ein Vorteil dieses Filters ist die einfache Anwendungsmöglichkeit in einer beliebigen Anzahl von Dimensionen. Eindimensionale Felder und zweidimensionale Bilder, aber auch dreidimensionale Strukturen wie Filmsequenzen lassen sich durch die Anwendung eines Gaußfilters glätten. Ein Gaußfilter ist *separierbar* [37, 36], so dass die Berechnung für jede Dimension getrennt erfolgen kann und deutlich weniger Rechenzeit notwendig ist. Der Nachteil des Gaußfilters besteht darin, dass das Bild unschärfer wird, und die Objektgrenzen verschwimmen. Zur Segmentierung ist dieser Filter daher nicht geeignet.

Wie in Kapitel 4 gezeigt wird, dient der Gaußfilter zur Glättung der Krümmungen von Konturen. Angewendet auf eine parametrisierte Kontur eines Objekts, verändert ein Gaußfilter die Form dieser Kontur in ähnlicher Weise wie ein morphologischer Filter.

3.5.2 Morphologische Filter

Morphologische Filter basieren darauf, dass die Nachbarschaft eines Punktes betrachtet wird, und der hellste bzw. dunkelste Wert dieser Umgebung die neue Pixelfarbe des aktuellen Punktes bestimmt. Von *Dilatation* spricht man, wenn der hellste Wert der Nachbarschaft gesucht wird, von *Erosion* bei der Suche nach dem dunkelsten Wert.

Die durchschnittliche Helligkeit bleibt bei diesen morphologischen Verfahren nicht erhalten. Bei Dilatation wird das Bild bei zunehmender Anzahl an Iterationen immer heller, bei Erosionen immer dunkler (vgl. Abb. 3.9 auf der nächsten Seite). Nach endlicher Zeit nimmt bei wiederholter Anwendung der Dilatation das Bild die hellste auftretende Pixelfarbe an, bei der Erosion die Dunkelste. Jegliche Bildinformation in Bezug auf die Form des abgebildeten Objekts ist dann verloren gegangen. Auch die Kontur eines abgebildeten Objekts ändert sich deutlich. Ein besonderer Nachteil dieser beiden morphologischen Verfahren liegt in ihrer Rauschempfindlichkeit. Die Auswirkung von nur einem Prozent Rauschanteil im Bild (vgl. 3.10 auf Seite 28) macht die Problematik dieses Filters deutlich.

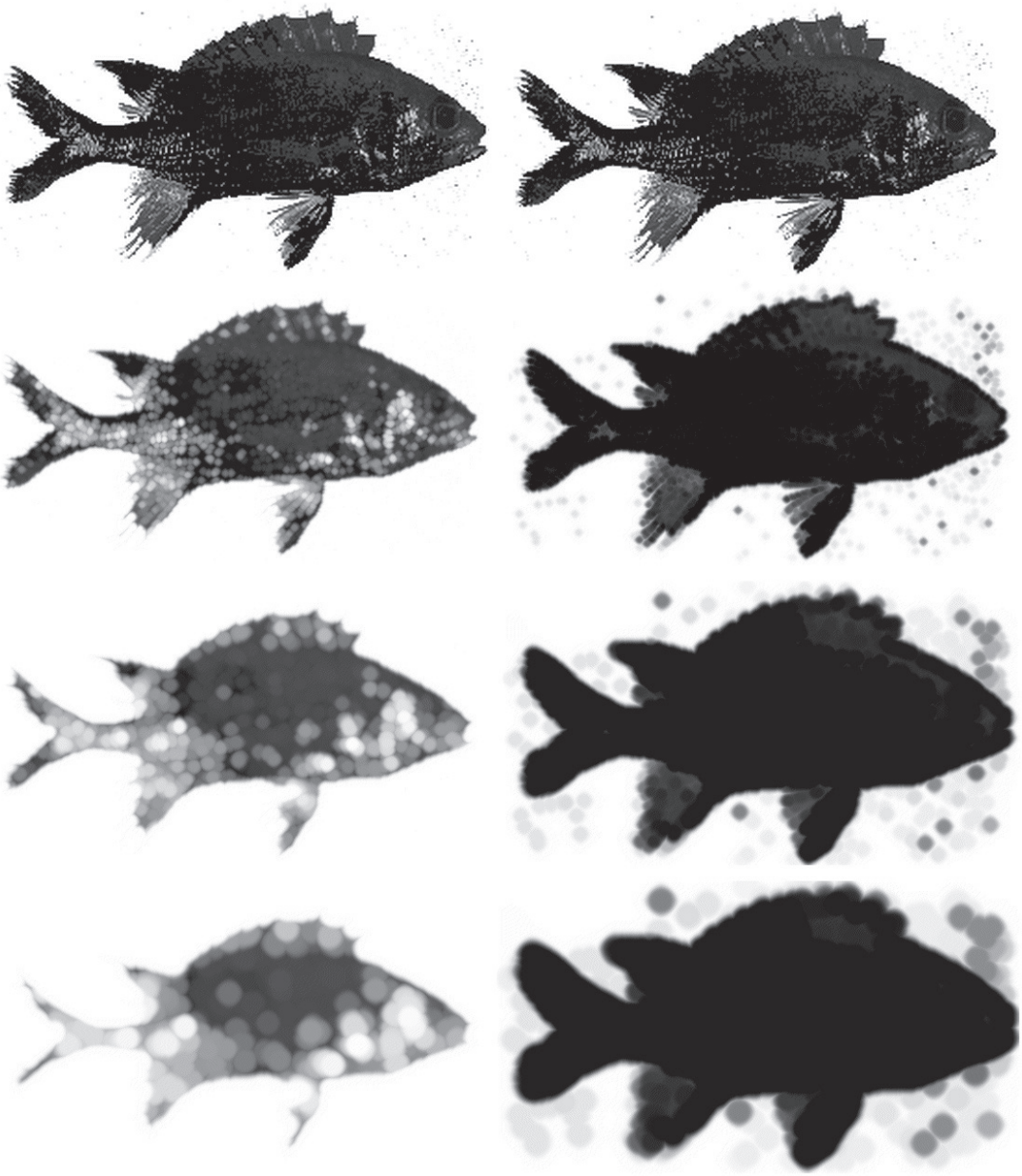

Abbildung 3.9: Beispiel für die morphologischen Verfahren Dilatation und Erosion. Bei allen Iterationen wurde eine Schrittweite von 0.4 gewählt. In der *linken Spalte* wird von von oben nach unten eine Dilatation mit 0, 8, 20 und 35 Iterationen durchgeführt. Die *rechte Spalte* verdeutlicht die Erosion bei gleicher Anzahl an Iterationen. Die oberste Zeile gibt jeweils das Originalbild (entnommen aus [26]) wieder. Als Region für diesen Filter wurde ein Kreis gewählt. Neue kreisförmige Strukturen können in allen Abbildungen deutlich erkannt werden.

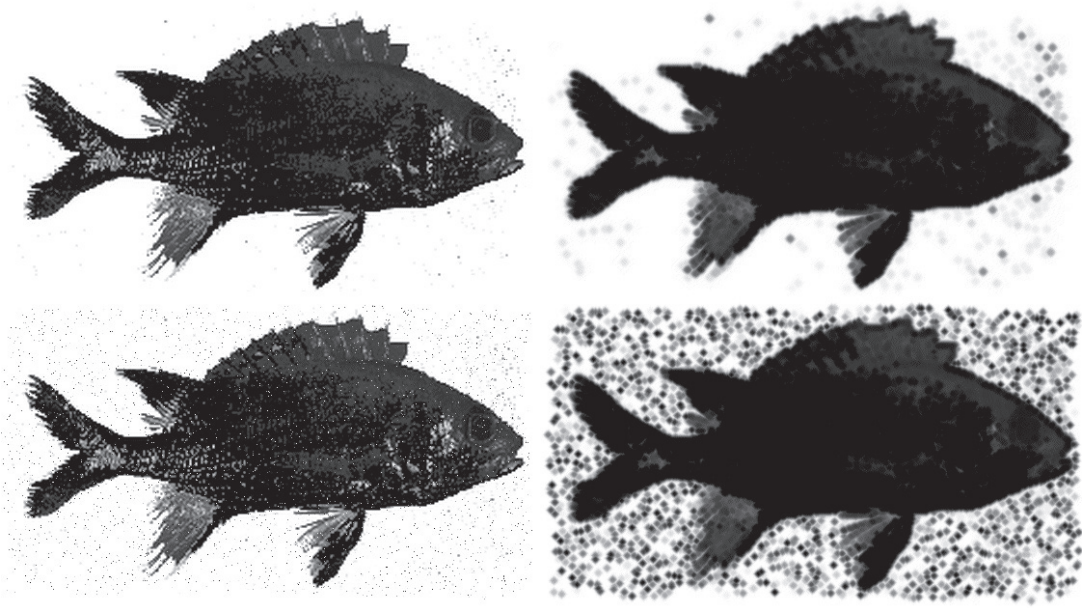

Abbildung 3.10: Erosion verrauschter Bilder. Bei den Iterationen wurde eine Schrittweite von 0.4 gewählt. Links oben ist das Originalbild, darunter das entsprechende Bild mit 1 Prozent zufällig verrauschter Pixel abgebildet. Rechts daneben wurde eine Dilatation mit 8 Iterationsstufen angewendet. Die Rauschempfindlichkeit dieses Filters ist rechts unten deutlich erkennbar.

Die gewählte Nachbarschaft, wie beispielsweise eine kreis- oder ellipsenförmige Umgebung um einen zentralen Punkt, hat Einfluss auf die Änderung der Form der Objekte im Bild. Bei einem Kreis, bilden sich im Bild ebenfalls kreisförmige Strukturen heraus, bei einer Ellipse, werden diese Strukturen verstärkt. In Abbildungen 3.9 auf Seite 27 und 3.10 auf der vorherigen Seite dient jeweils ein Kreis als Umgebung.

Die Kombination von Dilatation und Erosion ergeben wirkungsvolle Filter, der die Kanten eines Objekts verstärken, weniger rauschempfindlich sind und das abgebildete Objekt verformen. Aus der Verformung im Zeitablauf lassen sich Rückschlüsse auf die ursprüngliche Form des Objekts ziehen. Diese Informationen dienen später der Objekterkennung. Unter dem morphologischen Verfahren *Opening* versteht man die Anwendung von Dilatation und anschließender Erosion, unter *Closing* die beiden morphologischen Filter in umgekehrter Reihenfolge. Bei iterativer Anwendung dieser Filter wird das Objekt bei binären Bildern runder und die Größe verringert sich (vgl. Abb. 3.11 auf der nächsten Seite). Alle konkaven Bereiche verschwinden im Zeitablauf, so dass nach ausreichend langer Iteration ein konvexes Objekt entsteht.

Diese morphologische Filter weisen eine große Ähnlichkeit mit den krümmungsbasierten Verfahren zur Manipulation der Kontur auf. Diese Verfahren und der mathematische Zusammenhang mit den morphologischen Verfahren wird Thema des nächsten Kapitels sein.

Abbildung 3.11: Das morphologische Verfahren *Closing*. Es wird auf einem Binärbild mit einer Schrittweite von 0.4 ausgeführt. Von links oben nach rechts unten beträgt die Anzahl der Iterationen: 0, 5, 20, 35, 50, 80, 120, 150 und 180. Anschließend wurde das durch den Filter entstandende Graustufenbild in ein Binärbild umgewandelt. Closing verändert und vereinfacht die Form eines Objekts. Der Kopf wird *runder* und schrumpft, bis nur noch konvexe Bereiche vorhanden sind.

Kapitel 4

Krümmungsbasierter Skalenraum

In diesem Kapitel wird erläutert, wie eine Objekterkennung mit Hilfe von Konturen realisiert werden kann. Die effiziente Speicherung der wesentlichen Merkmale einer Kontur wird mit Hilfe einer Abbildung im krümmungsbasierten Skalenraum realisiert. Die Eigenschaften dieser Abbildungen sind Schwerpunkt dieses Kapitels.

4.1 Charakteristische Daten zur Beschreibung einer Kontur

Um einen Objektvergleich effizient durchzuführen, müssen relevante Faktoren von Objekten bestimmt werden, die mit wenigen Daten gespeichert werden können. Wenige Faktoren sollten ausreichen, da ein unbekanntes Bild mit allen vorhandenen Bildern der Datenbank verglichen werden muss. Die Berechnung der Faktoren sollte nur wenig Zeit in Anspruch nehmen. Wird anstatt eines einzelnen Bildes eine Bildsequenz analysiert, so ergibt die Kombination mit den gespeicherten Bildern der Datenbank mehrere hundert Bildvergleiche.

Ein Vektor nimmt für jedes Bild die charakteristischen Daten des abgebildeten Objekts auf. Einfache Merkmale zur Beschreibung eines Objekts sind Textur, Farbe oder Größe in Form von Höhe und Breite, da wenige Zahlen ausreichen, um diese Informationen zu erfassen. In Bildsequenzen kann zusätzlich die Bewegungsrichtung eines Objekts oder die Änderungen der Größe im Zeitablauf berücksichtigt werden.

Schwieriger ist die Erfassung der Kontur (vgl. Kapitel 2.5 auf Seite 12). Damit die wesentlichen Merkmale einer Kontur durch wenige Zahlen erfasst werden können, ist eine Umwandlung in einen anderen Skalenraum, den krümmungsbasierten Skalenraum, notwendig. Bei einer *Abbildung im krümmungsbasierten Skalenraum* handelt es sich um eine mehrdimensionale Abbildung *geometrisch invarianter Faktoren* [22, 23]. Geometrisch invariante Faktoren sind Nullstellen und Extrema der Krümmungsfunktion, die im Zeitablauf betrachtet werden. In dieser Arbeit werden nur die Nullstellen berücksichtigt. Die Nullstellen der Krümmungsfunktion entsprechen den Wendepunkten einer Kontur, also den Übergängen zwischen konkaven und konvexen Bereichen.

Die Abbildung im krümmungsbasierten Skalenraum liefert für die spätere Bildanalyse die Werte, die für einen Vergleich zweier Objekte benötigt werden. Besonders lange und stark gekrümmte Bereiche einer Kontur werden hervorgehoben. Die Positionen dieser Bereiche können mit Hilfe weniger Wertepaare gespeichert werden und sind charakteristisch für die Kontur des Objekts.

Bevor erläutert wird, in welcher Form eine Abbildung im krümmungsbasierten Skalenraum erzeugt und für Vergleiche der Kontur verwendet werden kann, werden zunächst Ergebnisse aus früheren Arbeiten zur Objekterkennung vorgestellt, die auf der Krümmung von Objekten basieren.

4.2 Objekterkennungssysteme auf Basis von Krümmungen

Erste umfangreiche Implementierungen von Objekterkennungsalgorithmen, die Abbildungen im krümmungsbasierten Skalenraum verwenden, basieren auf Arbeiten aus dem Jahr 1995 [22]. Einfache Objekte in Profilansichten werden anhand ihrer Konturen in die beiden Kategorien konvexe und nicht konvexe Objekte eingeteilt. Anhand der Stärke der Krümmung werden Objekte miteinander verglichen. Die Position der am stärksten gekrümmten Bereiche lässt sich durch eine Abbildung des Objekts im krümmungsbasierten Skalenraum bestimmen. Der Algorithmus ermöglicht es, bekannte Objekte zu erkennen. Mit einer umfangreichen Implementierung des Verfahrens [24, 25] können ähnliche Tiere in einer Datenbank mit 450 Meerestieren gesucht werden. Die Bilder der Datenbank bestehen ausschließlich aus gut erkennbaren Profilansichten. Der Fisch in Abbildung 3.7 auf Seite 24 stammt aus dieser Datenbank. In weniger als einer Sekunde vergleicht der Algorithmus eine unbekannte Kontur mit 50 ausgewählten Meerestieren und findet das Tier mit der ähnlichsten Kontur. Wie erfolgreich der Algorithmus funktioniert und wann Konturen einander ähneln, wird nicht genauer spezifiziert.

1997 wurde ein leicht modifiziertes Verfahren [2, 26] vorgestellt, das Chrysanthemenblätter erkennt. In England sind fast 3000 verschiedene Arten von Chrysanthemen registriert, wobei jährlich bis zu 300 neue Arten hinzukommen. In der Beispielanwendung sind 300 Blätterarten anhand ihrer lokalen Extremwerte im krümmungsbasierten Skalenraum gespeichert. Mit einer Wahrscheinlichkeit von über 95 Prozent trifft der Algorithmus die korrekte Aussage, ob die Abbildung eines Blattes zu einer bekannten Chrysanthemenart gehört oder nicht. Die 5 ähnlichsten Blätter werden ermittelt und ausgegeben. Letztendlich trifft ein Botaniker die Entscheidung, ob es sich tatsächlich um eine neuartige Züchtung handelt oder nicht.

Die Transformation dreidimensionaler Objekte in einen auf Krümmungen basierenden Skalenraum bereitet erhebliche Probleme, da in jedem Punkt auf der Oberfläche eines dreidimensionalen Objekts zwei Krümmungen gemessen werden können [23, 26]. In einem auf Torsion basierenden Skalenraum wird eine kleine Anzahl ausgewählter Objekte erfolgreich erkannt.

Der erste Ansatz, der nicht ausschließlich gut erkennbare Profilansichen in der Datenbank verwendet, stammt aus dem Jahr 1999 [1, 26]. Die Objekte wurden mit Hilfe affiner Transforma-

tionen verzerrt. Die Erkennung der verzerrten Profilansichten ist trotz größerer Deformationen möglich.

Andere auf Krümmungen basierende Ansätze wurden in [4] bei der Erkennung und Zählung von Bäumen implementiert. In Luftaufnahmen von Wäldern konnten Anzahl und Position von Bäumen erfolgreich identifiziert werden. Das Verfahren basiert auf dem in Abbildung 3.5 auf Seite 20 dargestellten Zusammenhang zwischen dem betragsmäßigen Wert der Krümmung und dem Radius eines Kreises. Der Mittelpunkt des Kreises bestimmt die Position im Bild, an der der nicht sichtbare Baumstamm liegen muss. Selbst in dichten Wäldern mit ineinander verwobenen Baumwipfeln ermöglicht das Verfahren eine gute Positionierung der Bäume. Weitere Arbeiten [20, 16] beschreiben Möglichkeiten zur Glättung von Objekten, bei denen die Stärke der Krümmung die Geschwindigkeit der Glättung bestimmt.

Allen Verfahren ist gemeinsam, dass sie überwiegend Profilansichten von Objekten verwenden. Diese speziellen kanonischen Sichten reichen jedoch nicht immer aus, um Objekte in Fotos oder Filmen zu erkennen. Als weitere Einschränkung analysieren die Verfahren Abbildungen *flacher Objekte*, wie Chrysanthemenblätter oder Fische, bei denen wenige Informationen durch räumliche Informationen auftreten. Wenn in der Datenbank nur Profilansichten gespeichert sind, können Objekte in natürlichen Perspektiven nicht erkannt werden. In dieser Arbeit werden unterschiedliche kanonische Sichten als Datenbasis für die Objekte verwendet. Die unbekannten Objekte werden in einer für einen Menschen natürlichen Perspektive betrachtet. In den getesteten Bildsequenzen ändern sich sowohl Größe und Lichtverhältnisse der Objekte, als auch der Blickwinkel auf diese. Nach dem Überblick über die schon implementierten Verfahren, wird die gemeinsame Grundlage – die Abbildung eines Objekts im krümmungsbasierten Skalenraum – erläutert.

4.3 Abbildungen im krümmungsbasierten Skalenraum

Ein Objekt kann in eine entsprechende Abbildung im krümmungsbasierten Skalenraum transformiert werden. Zwei Verfahren werden vorgestellt, die beide zu einer Vereinfachung der Funktion der Krümmungswerte führen. Die beiden Verfahren zur *Glättung der Krümmung* und zur *Glättung der Kontur* weisen viele Übereinstimmungen auf. Beide verwenden einen Gaußfilter, der im ersten Fall die Krümmungsfunktion und im zweiten Fall die Wertepaare der Konturpunkte glättet. Die Kontur wird runder und konkave Bereiche verschwinden mit zunehmender Iteration des Gaußfilters. Unterschiede beider Verfahren liegen im wesentlichen in der Rechenzeit. Mit zunehmender Iteration wird die Krümmungsfunktion glatter. Die Vereinfachung wird als zeitlicher Prozess interpretiert und führt zu dem Begriff des krümmungsbasierten Skalenraums.

4.3.1 Repräsentation der Kontur durch ihre Krümmung

Nachdem die äußere Kontur eines Objekts ermittelt wurde, wird in jedem einzelnen Punkt der Kontur die Krümmung berechnet. Die Berechnung verwendet als Eingabe Bilder, d. h. es werden

zweidimensionale Abbildungen verwendet. Bildsequenzen werden als Liste einzelner Bilder interpretiert, die getrennt voneinander analysiert werden. Für die weiteren Berechnungen wird die Kontur in parametrisierter Form geschrieben:

$$\Gamma(u) = (x(u), y(u)) \qquad u \in 0 \ldots n-1 \tag{4.1}$$

Die Variable u kann n verschiedene Werte annehmen, wobei n der Anzahl der Pixel der Kontur entspricht. Da die Breite und Höhe der Bilder nach oben beschränkt ist, wird die Länge einer Kontur durch eine eindeutige und feste Anzahl an Punkten bestimmt.

Ausgehend von einem beliebigen Startpunkt auf der Kontur, wird, bis zum erneuten erreichen des Startpunktes, jedes benachbarte Pixel im Uhrzeigersinn besucht. Dieser frei wählbare erste Punkt soll an der Position $\Gamma(0)$ liegen. Für die Ermittlung benachbarter Konturpunkte wird immer eine *Vierer-Nachbarschaft* betrachtet. Dabei handelt es sich um die vier benachbarten Punkte, die horizontal oder vertikal an den aktuellen Punkt angrenzen. Punkte auf einer Diagonalen werden nicht berücksichtigt. Durch die Verwendung der Vierer-Nachbarschaft sind die Abstände zweier benachbarter Konturpunkte immer identisch und werden mit einem Abstand der Größe eins festgelegt. Für jeden besuchten Punkt der Kontur wird der Parameter u um eins erhöht, so dass n der Länge der gesamten Kontur entspricht.

Die Kontur ist eine geschlossene Kurve, für die gilt:

$$\begin{aligned} \Gamma(0) &= \Gamma(n) && \text{und allgemein} \\ \Gamma(i) &= \Gamma(i+n) && \text{für } i = 0 \ldots n-1 \end{aligned} \tag{4.2}$$

Die parametrisierten Konturpixel $\Gamma(u)$ sind numeriert, d. h. sie werden in Form zweier Felder $X(u)$ und $Y(u)$ mit $u = 0 \ldots n-1$ gespeichert, wobei der Parameter u dem Index des Feldes entspricht. Für jeden Punkt u kann nach der Formel 3.8 auf Seite 23 die Krümmung $\kappa(u)$ berechnet werden. Als Grundlage für die Berechnung der Krümmung dient entweder das Binärbild mit der Kontur oder das ursprüngliche Graustufenbild. Zunächst wird ein Graustufenbild betrachtet, da sich durch die abgestuften Helligkeitswerte genauere Ergebnisse für die Berechnung der Krümmung ergeben.

Auch die Krümmungswerte werden in einem Feld gespeichert. Über den Index u ist ein direkter Zugriff auf die Felder $X(u)$, $Y(u)$ und $\kappa(u)$ möglich. Die Daten der drei Felder belegen im Vergleich zum ursprünglichen Graustufenbild im Durchschnitt weniger als 10 Prozent Speicherplatz. Die genaue Größe hängt von der Bildgröße, der Objektgröße und der Komplexität der Kontur ab.

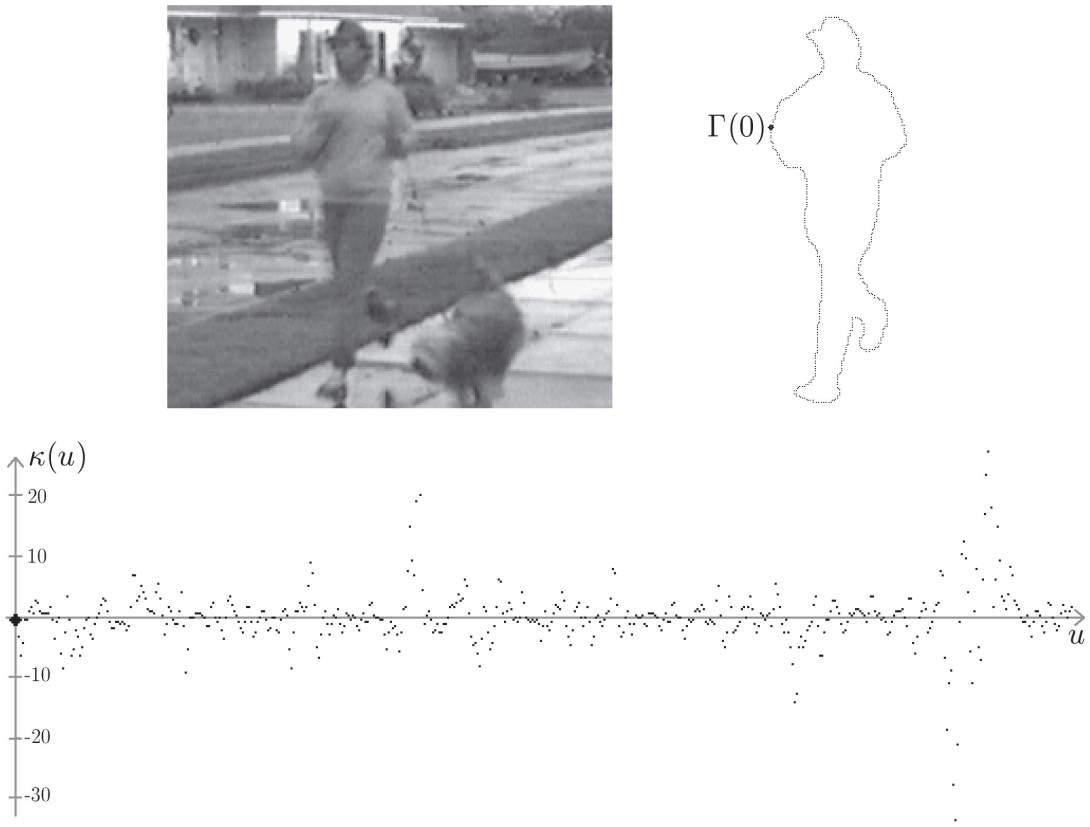

Abbildung 4.1: Abbildung einer Krümmungsfunktion. Darstellung eines Graustufenbildes, der entsprechenden Kontur und der Krümmungsfunktion der Kontur. Ausgehend vom Startpunkt $\Gamma(0)$ wird die Kontur im Uhrzeigersinn verfolgt. Die Kontur wird durch 500 Wertepaare abgebildet. Vorhandenes Rauschen und eine geringe Auflösung des Graustufenbildes führen zu den deutlichen Sprüngen in der Krümmungsfunktion.

In umgekehrter Richtung lassen sich aus den Werten einer Krümmungsfunktion Rückschlüsse auf die Form einer Kontur ziehen. Bei Binärdaten ist eine exakte Rekonstruktion der Kontur möglich, nur für die Rekonstruktion der Größe und der Orientierung der Kontur werden weitere Angaben benötigt. Die Krümmung gibt in jedem Punkt an, wo das nachfolgende Pixel angeordnet sein muss, wenn die Position des aktuellen und vorherigen Konturpixel bekannt ist. Zur Rekonstruktion der Kontur können zwei beliebige in Vierer-Nachbarschaft liegende Startpunkte des Bildes festgelegt werden. Anhand der Krümmungswerte wird ein dritter Punkt berechnet.

Informationen über die Orientierung des Bildes können nicht aus dem Krümmungsbild gewonnen werden. So muss möglicherweise eine Rotation des Bildes durchgeführt werden, um den Originalzustand des Bildes wieder herzustellen. Beträgt der Abstand zweier benachbarter Pixel eins, dann wird die Kontur in ihrer exakten Größe rekonstruiert.

In Abbildung 4.1 ist ein Graustufenbild, die zugehörige Kontur und die Krümmungsfunktion

abgebildet. Jeder Punkt der Kontur hat auf der x-Achse der Krümmungsfunktion einen Wert. Lokale konvexe Bereiche haben positive Krümmungswerte, konkave Bereiche negative Werte.

Der Parameter u wird üblicherweise auf Werte im Intervall $[0,1]$ normiert [23, 25]. Die Vereinheitlichung bietet für die zugrunde liegende mathematische Theorie Vorteile, für praktische Berechnungen ist sie nicht relevant. Damit ein Vergleich von Objekten möglich wird, muss jede Kontur auf eine gleiche Anzahl von Konturpunkten reduziert werden. Die Reduktion der Konturpunkte ist gleichbedeutend mit einer Skalierung der Bildgröße. In Abbildung 4.1 auf der vorherigen Seite wurden 500 Wertepaare zur Darstellung der Kontur verwendet. Geringe Fehler lassen sich durch diese Vereinfachung der Kontur nicht vermeiden (vgl. Abb. 3.1 auf Seite 16). In Kapitel 5 werden die Parameter des Algorithmus genauer spezifiziert und Auswirkungen durch Änderung der Anzahl der Konturpixel aufgezeigt.

4.3.2 Evolution der Krümmung

Trägt man die Werte der Krümmung abhängig vom Parameter u als Funktion ab (vgl. Abb. 4.1 auf der vorherigen Seite), so wird deutlich, dass durch die diskreten Werte des Bildes und durch die Approximation der Ableitungen Ungenauigkeiten bei den Krümmungswerten auftreten. Geringe Unebenheiten der Form der Kontur bewirken deutliche Abweichungen einzelner Werte. Größere konvexe Bereiche haben überwiegend positive Krümmungswerte, wobei durch den Abstand der Pixel untereinander und die diskreten Graustufenwerte auch vereinzelte negative Krümmungswerte auftreten. Es handelt sich um kleine konkave Bereiche mit einer Länge von wenigen Pixeln.

Um relevante konkave und konvexe Bereiche zu ermitteln und zufällige Abweichungen einzelner Werte zu vermeiden, wird die Krümmungsfunktion durch einen Gaußfilter geglättet. Ein eindimensionaler Gaußfilter glättet die Liste der Krümmungswerte $\kappa(u)$. Nach Anwendung eines Gaußfilters bleibt der Mittelwert aller Krümmungen erhalten und die Funktionswerte nähern sich diesem bei iterativer Anwendung an. In Abbildung 4.2 auf der nächsten Seite wird eine Krümmungsfunktion geglättet. Im Zeitablauf reduzieren sich die Anzahl der Nullstellen und der lokalen Extrema. Unter *Zeitablauf* soll die Anzahl der Iterationsstufen des Gaußfilters verstanden werden. Die Werte der Maxima sinken, die der Minima steigen. Um die Feinheiten der Struktur deutlich zu machen, wurde bei den letzten beiden Gaußglättungen dieser Abbildung ein feinerer Maßstab gewählt.

Die Glättung im Zeitablauf wird auch als *Evolution der Krümmung* bezeichnet [20, 25]. Mit zunehmender Zeit wird die Krümmungsfunktion glatter. Da der durchschnittliche Wert der Krümmung negativ ist, nähert sich die Krümmungsfunktion einer Geraden an, die unterhalb der x-Achse liegt. Konvexe Objekte weisen nur noch negative Krümmungswerte auf. Die Kontur wird mit der Glättung der Krümmungsfunktion runder. In dem Moment, in dem keine Nullstellen der Krümmungsfunktion mehr auftreten, verschwindet der letzte konkave Bereich der Kontur. Ein *stark konkaver Bereich* ist ein Abschnitt der Kontur, der besonders stark und lang in eine Richtung gekrümmt ist. Diese Abschnitte liefern die relevanten Informationen für die Ob-

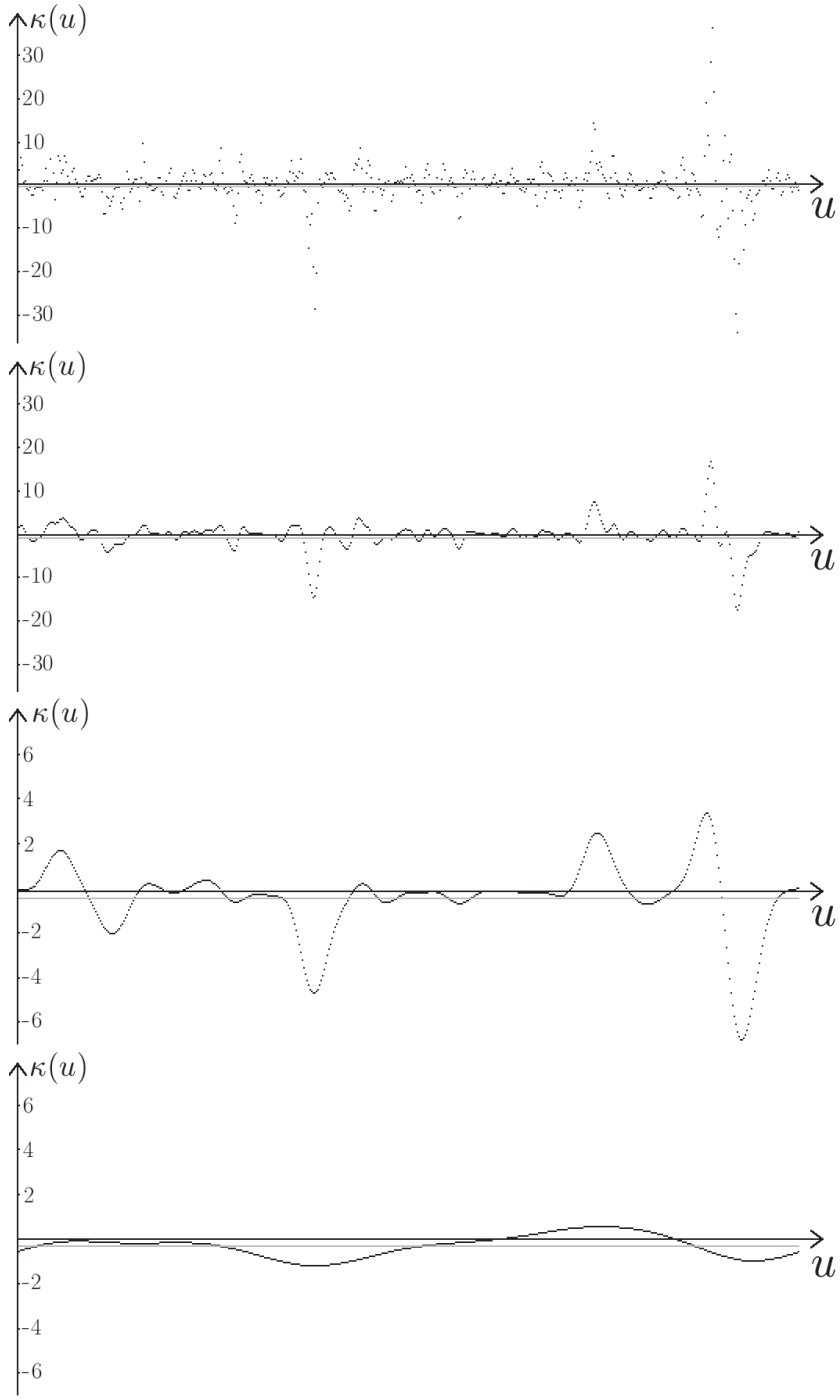

Abbildung 4.2: Krümmungsfunktion nach einer Gaußglättung. Bei einer Maskenbreite von 3 werden 0, 10, 150 und 2500 Iterationen durchgeführt. Die Annäherung der Werte in Richtung des Mittelwertes wird deutlich. In den unteren beiden Abbildungen wurde eine feinere Abbildung der y-Skala gewählt.

jekterkennung. Die Positionen der konkaven Bereiche können durch die Nullstellen der Krümmungsfunktion bestimmt werden. Der Zusammenhang zwischen geglätteten Krümmungen und entsprechenden Objekten wird in Abbildung 4.3 auf der nächsten Seite verdeutlicht.

Eine Gaußglättung der Krümmungsfunktion verformt eine Kontur in ähnlicher Weise wie das in Kapitel 3 vorgestellte morphologischen Verfahren *Closing*. Die Extrema der Krümmungsfunktion ändern ihre Werte schnell, während die Werte nahe dem Durchschnitt sich nur geringfügig ändern. Jede Kontur nimmt bei ausreichend langer Glättung die Form eines Kreises an, der in jedem Punkt eine gleich starke Krümmung besitzt [17]. Beim Closing-Verfahren ändern sich stark gekrümmte Bereiche der Kontur ebenfalls schneller, da dort innerhalb einer kleinen Umgebung deutliche Wertänderungen auftreten.

Zur Berechnung der Krümmung müssen Ableitungen approximiert werden. Diese Approximation liefert für Binärbilder ungenauere Ergebnisse als für Graustufenbilder. Daher ist das Verfahren zur Evolution der Krümmung für Graustufenbilder besser geeignet als für Binärbilder. Im nächsten Abschnitt wird ein anderes Verfahren vorgestellt, das eine Berechnung der Krümmung in Binärbildern ermöglicht.

4.3.3 Evolution der Kontur

Es existiert eine zweite Möglichkeit, um die Krümmungswerte zu glätten [1, 22, 24, 25]. Hierzu wird die parametrisierte Kontur $\Gamma(u)$ direkt geglättet. Ein eindimensionaler Gaußfilter vereinfacht die x- und y-Koordinaten der Kontur (X_u und Y_u). Die beiden Koordinaten werden getrennt voneinander geglättet, d. h. ein Gaußfilter glättet erst die Werte X_u und anschließend die Werte Y_u. Bei einer fest definierten Maskenbreite wird die Glättung so lange wiederholt, bis in der zugehörigen Krümmungsfunktion keine Nullstellen mehr auftreten. Die Kontur ist dann konvex.

In Abbildung 4.4 auf Seite 40 wird die Änderung der Kontur im Zeitablauf verdeutlicht. Die Ähnlichkeit zum morphologischen Verfahren Closing ist deutlich erkennbar (vgl. Abb. 3.11 auf Seite 30). Die Berechnung einer Iteration beansprucht beim Closing deutlich mehr Zeit als die Glättung mit einem eindimensionalen Gaußfilter. Während bei dem Closing-Verfahren in jedem Iterationsschritt jedes Pixel mindestens zweimal (Dilatation und Erosion) verändert wird und die zweidimensionale Umgebung des zu ändernden Pixel eingelesen werden muss, werden bei der Gaußglättung nur einmal die Konturpixel modifiziert. Die Anzahl der Konturpixel ist deutlich geringer als die Anzahl der gesamten Bildpunkte, so dass die Gaußglättung entlang der x- und der y-Koordinaten weniger Punkte berücksichtigen muss. Sie werden im Arbeitsspeicher in einem Feld abgelegt, so dass der Zugriff auf benachbarte Feldelemente durch den Cache sehr schnell erfolgen kann.

Zu jedem Zeitpunkt des Glättungsverfahrens entsteht genau eine Kontur des Objekts. Im Zeitablauf wird die Form des Objekts runder und kleiner, bis es zu einem Punkt zusammenschrumpft. Für die Glättung der Kontur ist eine exakte Berechnung der Krümmung nicht notwendig. Lediglich das Vorzeichen der Krümmung, das sich mit wenigen elementaren Rechenoperationen ermitteln lässt, ist für die spätere Bilderkennung von Bedeutung.

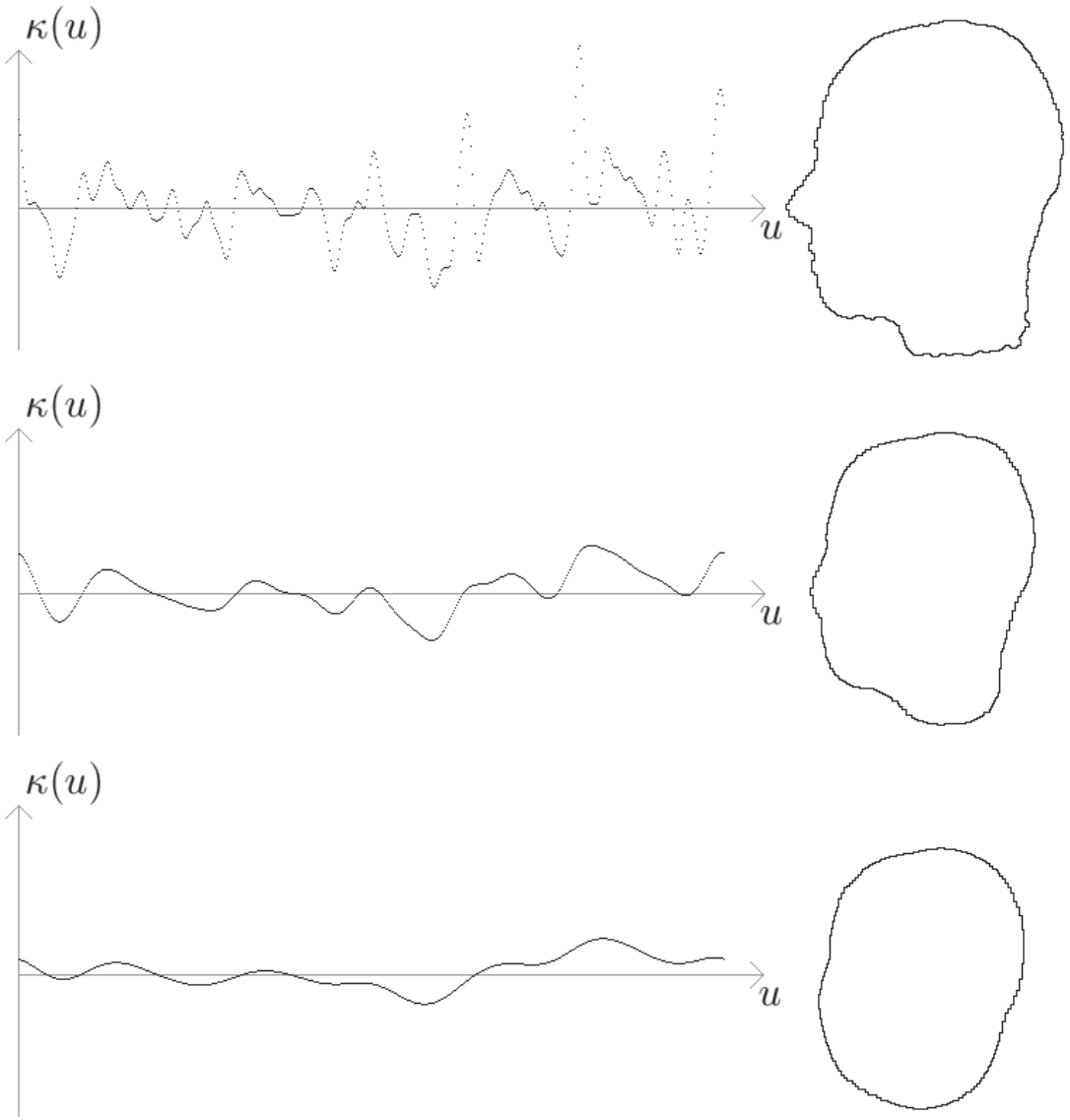

Abbildung 4.3: Konturänderungen bei der Evolution der Krümmung. Drei Konturen aus dem Beispiel zum Closing-Verfahren (vgl. Kapitel 3.5.2 auf Seite 26) werden mit den zugehörigen Krümmungsfunktionen abgebildet. Je glatter die Krümmungsfunktion wird, desto runder wird die Kontur des zugehörigen Objekts.

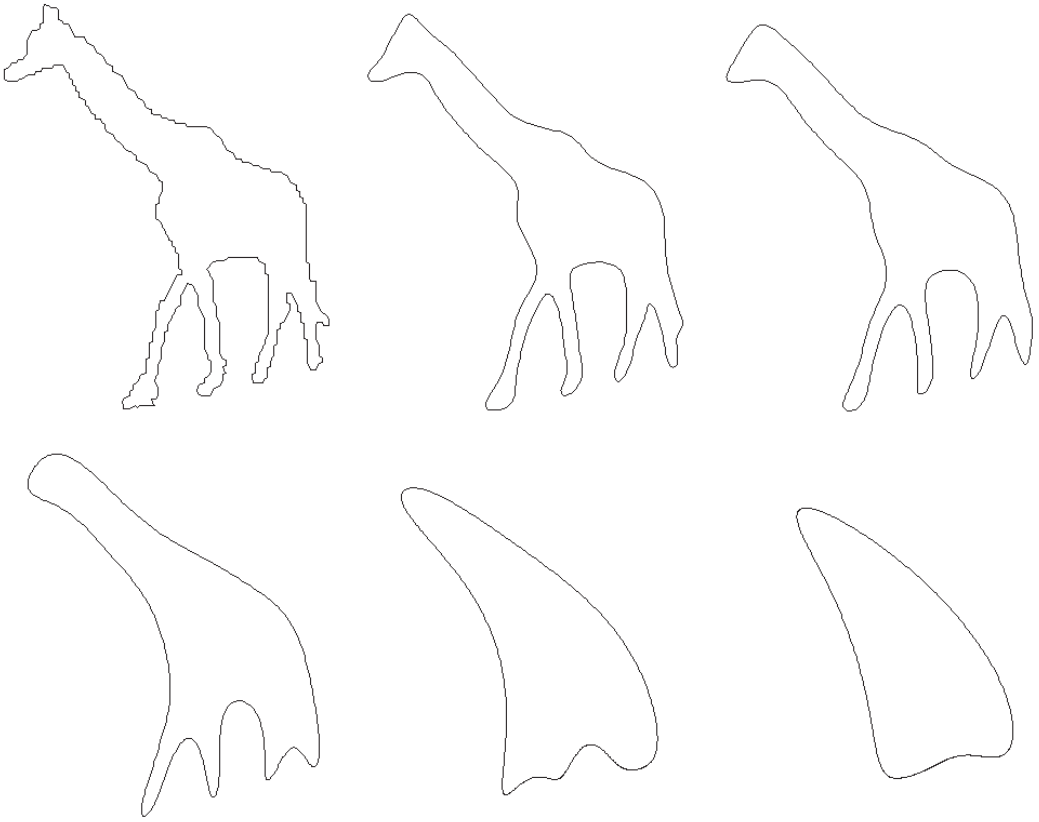

Abbildung 4.4: Evolution der Kontur. Die Abbildung zeigt 6 unterschiedlich stark geglättete Konturen. Die parametrisierte Kontur einer Giraffe wird entlang der x- und y-Koordinaten mit einer Gaußmaske der Breite 7 geglättet. Von links oben nach rechts unten werden 0, 6, 20, 80, 400 und 1000 Iterationen durchgeführt.

Bisher wurde noch nicht geklärt, wofür die Krümmungsfunktionen in den unterschiedlichen Zeitpunkten benötigt werden. Die Informationen, die aus der zeitlichen Folge der Krümmungsbilder ableitbar sind, werden im folgenden Abschnitt erläutert.

4.3.4 Geordnete Sequenzen von Krümmungsbildern

Beide vorgestellten Glättungsverfahren liefern für jeden Iterationsschritt, der als Zeitpunkt t gemessen wird, eine diskrete Funktion mit Krümmungswerten. Zu Beginn der Iteration, also bei kleinem t, treten deutliche Sprünge zwischen benachbarten Werten der Krümmungsfunktion auf.

Die Evolution der Krümmungsfunktion im Zeitablauf wird im folgenden auch als *geordnete Sequenz der Krümmungen* κ_t bezeichnet. Zu jeder Krümmungsfunktion gehört eine entsprechend vereinfachte Kontur $\Gamma_t(u)$.

Die Funktion der Krümmungswerte kann im jedem Zeitpunkt mit wenigen charakteristischen Merkmalen beschrieben werden. Wesentliche Punkte sind die lokalen Extrema und die Nullstellen. Nullstellen lassen sich besonders einfach berechnen, da sich das Vorzeichen zweier aufeinanderfolgender Krümmungswerte ändert und somit das Produkt dieser benachbarten Werte kleiner Null ist. Die Nullstellen zeigen einen Wechsel der Krümmung der Kontur an, also den Übergang eines konkaven Bereichs in einen konvexen oder umgekehrt.

Die Berechnung der Positionen der lokalen Extrema ist bei einer Sequenz geordneter Krümmungsbilder von untergeordneter Bedeutung, da sie sich im Zeitablauf nur geringfügig ändern. Die Evolution bewirkt, dass die längsten und am stärksten ausgeprägten konkaven Bereiche auch bei fortgeschrittener Iterationsstufe erhalten bleiben. Die kleinen konkaven Bereiche verschwinden zuerst. Jeweils zwei Nullstellen grenzen einen konkaven Bereich ein. Die beiden Nullstellen wandern während der Glättung zusammen. Wird die zeitliche Auflösung ausreichend fein gewählt, so treffen sich jeweils zwei Nullstellen in einem Punkt der Krümmungsfunktion. Die Punkte beschreiben die Position, an denen die Kontur maximal konkav gekrümmt ist. Zusätzlich zu der Stärke der Krümmung wird die Länge des konkaven Bereichs berücksichtigt.

Die Position der Nullstellen kann im Zeitablauf in einem Binärbild dargestellt werden. Die x-Achse beschreibt die entsprechende Position auf der Kontur und entspricht dem Parameter u. Auf der y-Achse wird die Zeit abgetragen, also die Anzahl der Iterationsschritte während der Evolution. Für jeden Iterationsschritt werden in der Abbildung die Nullstellen der Krümmungsfunktion, welche den Wendepunkten der Kontur entsprechen, durch einen Punkt markiert. Abbildung 4.5 auf der nächsten Seite zeigt die Abbildung einer Kontur im krümmungsbasierten Skalenraum. Besonders starke konkave Bereiche sind mit Ziffern gekennzeichnet, wobei der Bereich mit der Ziffer 1 der stärkste konkave Bereich ist. Er bleibt während der Evolution der Kontur am längsten erhalten.

Die *Abbildungen im krümmungsbasierten Skalenraum (CSS-Abbildungen)*[1] haben Eigen-

[1] Die englische Bezeichnung *curvature scale space image* wird in der Literatur mit CSS-image abgekürzt. Ent-

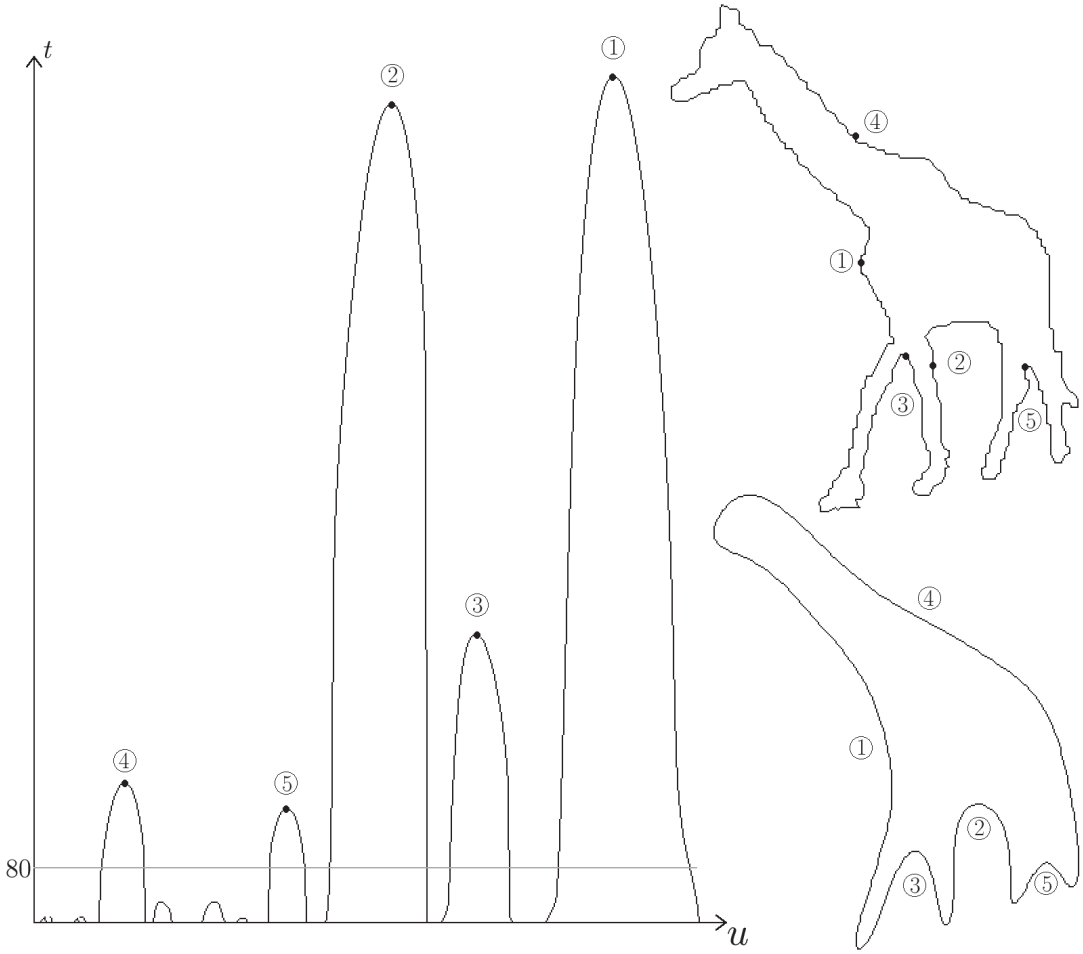

Abbildung 4.5: Abbildung im krümmungsbasierten Skalenraum. Rechts oben ist die ursprüngliche Kontur eines Objekts abgebildet, darunter die Kontur zum Zeitpunkt $t = 80$. Die entsprechende Abbildung im krümmungsbasierten Skalenraum ist links zu sehen. Die fünf stärksten konkaven Bereiche wurden in allen Abbildungen markiert.

schaften, die für eine Objekterkennung besonders vorteilhaft sind. Eine *Rotation* des ursprünglichen Objekts bewirkt lediglich eine Verschiebung der Krümmungsfunktion und somit der CSS-Abbildung entlang der x-Achse. Eine Änderung der Krümmungswerte oder Nullstellen tritt zu keinem Zeitpunkt auf. Ein anders gewählter Startpunkt der Kontur hat dieselbe Auswirkung wie eine Rotation. Da immer geschlossene Konturen vorliegen, entspricht eine Rotation eines Objekts einer Addition einer Konstanten zu dem Parameter u. Wie Abbildung 4.6 auf der nächsten Seite verdeutlicht, wirkt sich eine Drehung durch eine horizontale Verschiebung der Punkte der CSS-Abbildung und eine Spiegelung der Kontur durch eine Spiegelung der CSS-Abbildung aus.

Wird ein Objekt vergrößert, so erhöht sich die Anzahl der Konturpixel und damit die Anzahl der Punkte u auf der x-Achse im CSS-Bild. Die Krümmung im Objekt wird länger, so dass die Gaußglättung mehr Iterationsschritte benötigt, um die Kontur in einen konvexen Zustand zu überführen. Wie in Abbildung 4.7 auf Seite 45 deutlich wird, ist ein Vergleich zweier unterschiedlich breiter CSS-Bilder schwer möglich. Bei einer doppelten Anzahl an Konturpixel wenden für die Gaußglättung mehr als doppelt so viele Iterationen benötigt. Eine genaue Aussage über die Höhe der CSS-Abbildung kann nicht getroffen werden. Die Probleme, die durch unterschiedliche Größen der Objekte auftreten können, werden vermieden, indem jede Kontur durch eine feste Anzahl an Pixel repräsentiert wird. Nach dem Segmentierungsschritt wird die Kontur vereinfacht, so dass jedes Objekt durch eine feste Anzahl an Konturpaaren beschrieben wird.

Eine CSS-Abbildung ist sehr unempfindlich gegenüber Rauschen. In Abbildung 4.7 auf Seite 45 werden drei leicht differierende Konturen mit den zugehörigen CSS-Bildern dargestellt. Die CSS-Abbildungen zeigen nahe der x-Achse Unterschiede. Die Maxima der CSS-Abbildung, also die großen konkaven Bereiche der Kontur, weisen praktisch keine Unterschiede auf. Der Gaußfilter glättet das Rauschen zu einem frühen Zeitpunkt. Kleine Unterschiede der Kontur ändern die größeren gekrümmten Bereiche nicht. Aus der letzten Abbildung wird deutlich, welche Bereiche eines CSS-Bildes zur Beschreibung ähnlicher Konturen herangezogen werden können. In jeder CSS-Abbildung treten nur wenige lokale Hochpunkte auf. Der charakteristischer Vektor zur Beschreibung der Kontur eines Objekts wird die Position und den betragsmäßigen Wert der Hochpunkte speichern.

4.4 Eigenschaften des krümmungsbasierten Skalenraums

4.4.1 Maxima der CSS-Bilder

Informationen über die s größten lokalen Maxima der CSS-Abbildung sollen als Vektor gespeichert werden und wesentliche Merkmale der Kontur liefern. Die Positionen und Werte der lokalen Maxima werden in Form einer Liste mit Wertepaaren (vgl. Formel 4.3 auf Seite 46) gespeichert.

sprechend wird auch hier der Begriff *CSS-Abbildung* verwendet.

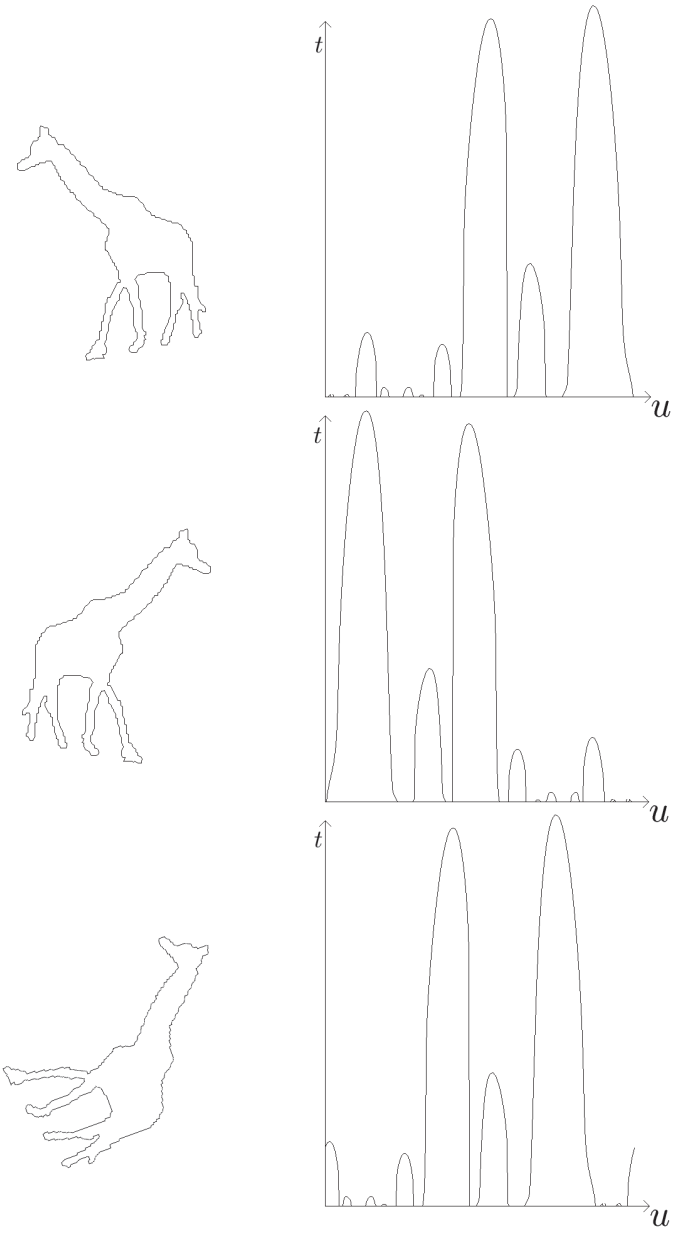

Abbildung 4.6: Auswirkung von Rotation und Spiegelung auf die Abbildung im krümmungsbasierten Skalenraum. Oben ist die Kontur im Originalzustand abgebildet, darunter die an der y-Achse gespiegelte Kontur. Der Startpunkt in der mittleren Abbildung ist unverändert geblieben. Die zugehörige Krümmungsfunktion entspricht der ersten Funktion, die an einer Parallelen zur y-Achse gespiegelt wurde. In der unteren Abbildung wurde das Objekt um 70 Grad gedreht und der Startpunkt verschoben. Bei der Krümmungsfunktion entspricht die Drehung einer Verschiebung entlang der x-Achse.

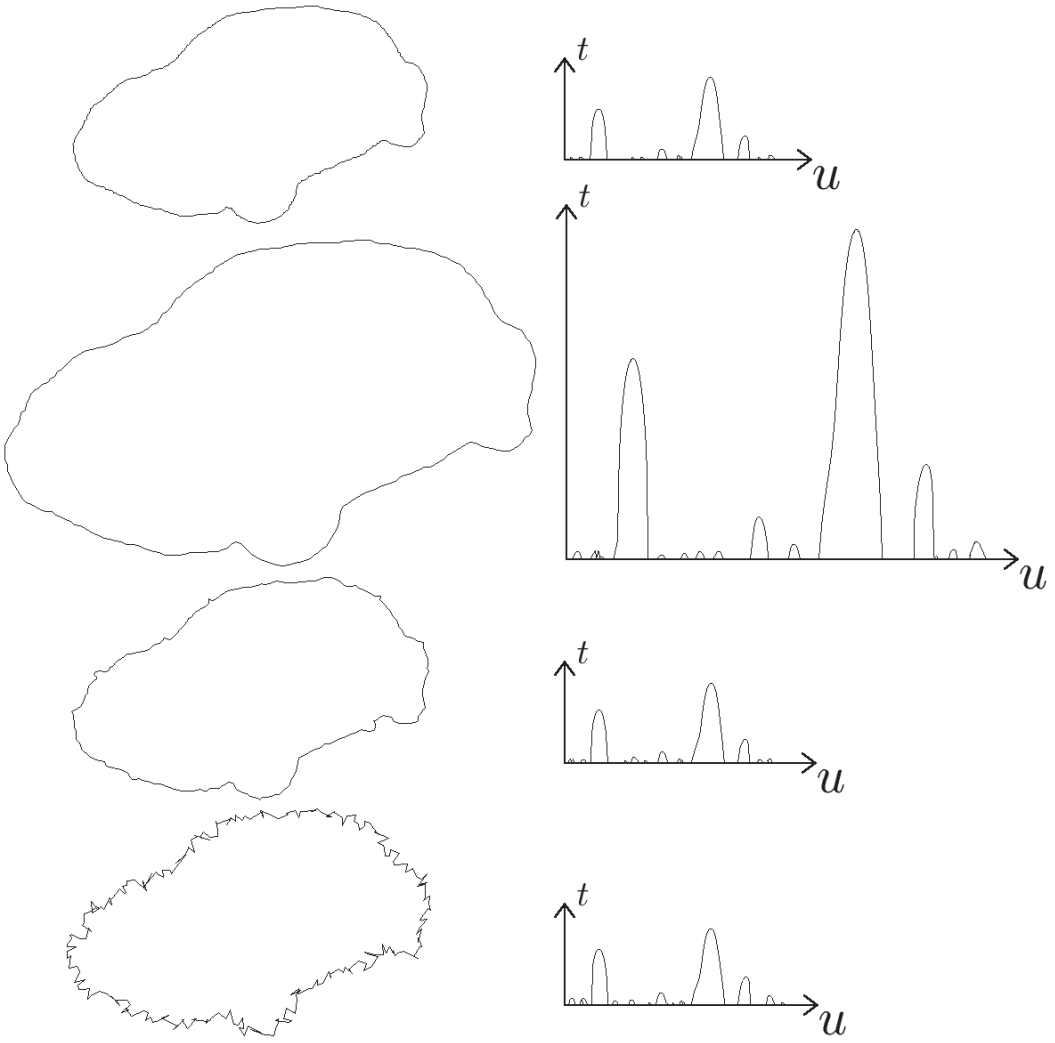

Abbildung 4.7: Auswirkung von Zoomeffekten und Rauschen auf die Abbildung im krümmungsbasierten Skalenraum. In den oberen beiden Bilder ist ein Auto in kanonischer Sicht abgebildet. In der dritten Kontur wurden bei dem zugrunde liegenden Graustufenbild kleine Änderungen durchgeführt. In der letzten Abbildung werden die Wertepaare der Kontur zufällig um bis zu sieben Pixel in die x- und y-Richtung verschoben. Alle CSS-Abbildungen weisen große Ähnlichkeiten auf. Ein Vergleich der CSS-Abbildungen mit dem vergrößerten Objekt bereitet jedoch Probleme, da der Faktor der Skalierung in x-Richtung nicht mit dem Faktor in y-Richtung übereinstimmt.

$$(u_1, \sigma_1), (u_2, \sigma_2), (u_3, \sigma_3), \ldots, (u_s, \sigma_s) \tag{4.3}$$

Die Wertepaare werden so sortiert, dass die größten Sigmawerte am Anfang der Liste stehen, und diese monoton abnehmen ($\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots$). Ein möglicher Vektor zur Beschreibung der Kontur kann folgendermaßen aussehen:

$$(22, 56), (115, 51), (11, 44), (76, 33), \ldots \tag{4.4}$$

Jeweils zwei Nullstellen der Krümmungsfunktion grenzen einen konkaven Bereich ein, der durch die Gaußglättung immer kleiner wird. Treffen während der Glättung beide Nullstellen in einem Punkt zusammen, so ergibt sich im Bereich der CSS-Maxima eine durchgängige Funktion. Durch die schnelle Gaußglättung treten jedoch häufig kleine Lücken in diesen Bereichen auf.

Vor der Ermittlung der lokalen Maxima wird das CSS-Bild analysiert und die Lücken, die in der Nähe der lokalen Maxima liegen, werden geschlossen. Die neuen Werte im Bereich der Lücken ergeben sich aus den Nachbarwerten. Bei mehreren möglichen Positionen eines lokalen Maximums wird die mittlere Position gewählt.

Der Vergleich, ob sich die Konturen zweier Objekte ähneln, basiert auf der Berechnung von Differenzen zwischen Wertepaaren von Vektoren, die aus den zugehörigen CSS-Bildern berechnet werden. Eine Differenz von fast Null weist auf eine große Ähnlichkeit der Konturen hin, bei steigender Differenz nehmen die Unterschiede der Konturen zu.

Der nächste Abschnitt verdeutlicht, dass keine exakten Krümmungswerte zur Berechnung eines CSS-Bildes benötigt werden. Ein vereinfachtes Verfahren wird vorgestellt, mit dem anhand dreier aufeinanderfolgender Wertepaare der Kontur das Vorzeichen der Krümmung ermittelt werden kann.

4.4.2 Approximation der Krümmung

Eine Approximation der Krümmung bietet sich nur in Kombination mit dem in Abschnitt 4.3.3 auf Seite 38 beschriebenen Verfahren zur Evolution der Kontur an, da bei der Evolution der Krümmung exakte Krümmungswerte benötigt werden.

Die Krümmung einer Kurve wird durch die ersten und zweiten Ableitungen bestimmt (vgl. Gl. 3.8 auf Seite 23). Bei einer nicht kontinuierlichen Kurve erfolgt die Berechnung der Ableitungen mit Hilfe von Näherungsverfahren. Für jedes Wertepaar der Kontur soll nur bestimmt

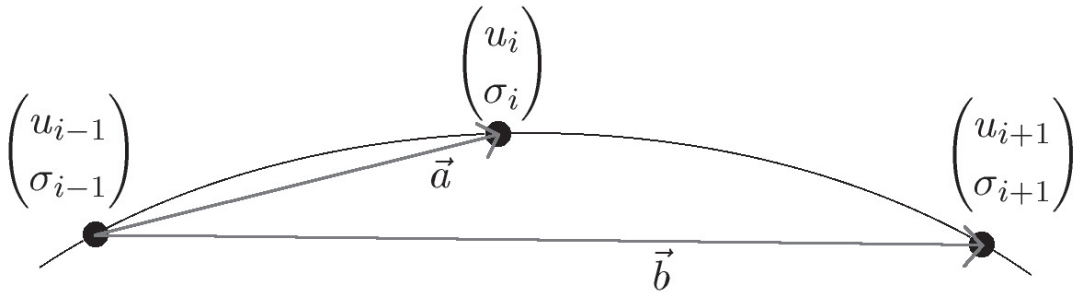

Abbildung 4.8: Approximation der Krümmung. Die beiden Vektoren $\vec{a}$ und $\vec{b}$ werden durch drei aufeinander folgende Punkte der Kontur $\Gamma(u)$ bestimmt.

werden, ob die Kurve konkav oder konvex ist, d. h. der Wert der Krümmung wird nicht benötigt. Die Position der Übergänge zwischen konkaven und konvexen Bereichen kann mit diesem Näherungsverfahren leicht bestimmt werden.

Zu jedem Punkt i wird die Sekante zwischen zwei weiteren Punkten *vor* und *nach* dem Punkt i, also $i-1$ und $i+1$ gebildet. Liegt der Punkt i *oberhalb* der Sekante, so ist die Kurve konvex, ansonsten konkav (vgl. Abb. 4.8).

Aus jeweils drei Wertepaaren $(u_{i-1},\sigma_{i-1}),(u_i,\sigma_i)$ und (u_{i+1},σ_{i+1}) mit $i \in 0\ldots u-1$ werden die Vektoren $\vec{a}$ und $\vec{b}$ berechnet.

$$\begin{aligned} \vec{a} &= \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} u_i \\ \sigma_i \end{pmatrix} - \begin{pmatrix} u_{i-1} \\ \sigma_{i-1} \end{pmatrix} \\ \vec{b} &= \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} u_{i+1} \\ \sigma_{i+1} \end{pmatrix} - \begin{pmatrix} u_{i-1} \\ \sigma_{i-1} \end{pmatrix} \end{aligned} \tag{4.5}$$

Ob der Punkt i unterhalb oder oberhalb der Sekante liegt, kann am Winkel α zwischen den beiden Vektoren $\vec{a}$ und $\vec{b}$ erkannt werden. Ist er negativ, so liegt der Punkt oberhalb, ansonsten unterhalb von $\vec{b}$. Die Kurve ist dann konvex bzw. konkav.

Für das Skalarprodukt der beiden Vektoren gilt:

$$\begin{aligned} |\vec{a}| \cdot |\vec{b}| \cdot \cos(\vec{a},\vec{b}) &= \vec{a} \cdot \vec{b} \\ &= a_1 \cdot b_1 + a_2 \cdot b_2 \end{aligned} \tag{4.6}$$

Der Kosinus des Winkels α zwischen den beiden Vektoren $\vec{a}$ und $\vec{b}$ wird in Gleichung 4.6 mit $\cos(\vec{a},\vec{b})$ bezeichnet. Am Kosinus des Winkels lässt sich jedoch nicht dessen Vorzeichen

ablesen, da er achsensymmetrisch zur x-Achse ist. Abhilfe schafft eine Transformation des Winkels um 90 Grad, die durch die Drehung einer der Vektoren erreicht werden kann.

Wird der zweite Vektor um 90 Grad gedreht, so gilt für den neuen Vektor $\vec{b}_\perp$:

$$\vec{b}_\perp = \begin{pmatrix} -b_2 \\ b_1 \end{pmatrix} \tag{4.7}$$

Für das Skalarprodukt mit dem transformierten Vektor gilt dann:

$$\begin{aligned} \vec{a} \cdot \vec{b}_\perp &= |\vec{a}| \cdot |\vec{b}_\perp| \cdot \cos(\vec{a}, \vec{b}_\perp) \\ &= -a_1 \cdot b_2 + a_2 \cdot b_1 \end{aligned} \tag{4.8}$$

Weiter gilt:

$$\begin{aligned} \vec{a} \cdot \vec{b}_\perp &= |\vec{a}| \cdot |\vec{b}_\perp| \cdot \cos(\vec{a}, \vec{b} - 90 \text{ Grad}) \\ &= |\vec{a}| \cdot |\vec{b}_\perp| \cdot \sin(\vec{a}, \vec{b}) \end{aligned} \tag{4.9}$$

Auf diese Weise erhält man den Sinus des Winkels, der bei 0 Grad sein Vorzeichen wechselt. Es reicht aus, das Vorzeichen des Ausdrucks zu betrachten. Ist es negativ, so ist auch der Winkel negativ. Alle weiteren Anteile dienen lediglich der Skalierung, die sowieso positiv und in diesem Zusammenhang nicht von Interesse ist. Der Punkt i liegt dann über der Sekante und die Kurve ist in diesem Punkt konvex.

Der Aufwand des Verfahrens ist äußerst gering und es ist sehr stabil. Ist eine Sekante sehr kurz, liegen die Punkte also sehr nahe zusammen, so kann sich die ermittelte Krümmung sehr oft ändern. Nach wenigen Gaußglättungen der Kontur verschwinden die Unregelmäßigkeiten. Das vereinfachte Verfahren zur Bestimmung der Übergänge konkaver und konvexer Bereiche der Kontur wird in dieser Arbeit verwendet.

4.4.3 Kritische Betrachtung

Die Transformation eines Objekts in den krümmungsbasierten Skalenraum bietet verschiedene Vorteile [23]. Jede Kontur wird in eine spezielle CSS-Abbildung umgewandelt, d. h. gleiche Konturen ergeben gleiche CSS-Abbildungen. Eine Größenänderung der Kontur hat keine Auswirkung und eine Rotationen verschiebt lediglich die CSS-Abbildung entlang der x-Achse. Umgekehrt gilt, dass unterschiedliche Konturen auch unterschiedliche CSS-Abbildungen besitzen.

Abhängig von der Anzahl der Konturpixel und der Größe der Gaußmaske können die Unterschiede jedoch minimal ausfallen.

Wesentlich für die Objekterkennung ist, dass es sich bei der Umwandlung in den krümmungsbasierten Skalenraum um ein stabiles Verfahren handelt. Kleine Unterschiede in der Kontur bewirken nur kleine Unterschiede im CSS-Bild. Inwieweit deutlich unterschiedliche Konturen zu ähnlichen CSS-Abbildungen führen, konnte bisher nicht sicher geklärt werden. In den verwendeten Ausgangsdaten (vgl. Kapitel 6) tritt diesbezüglich kein Problem auf.

Das Verfahren ist unempfindlich gegenüber Rauschen und geringeren Verzerrungen des Bildes. Für eine Implementierung ist es relevant, dass die Berechnung eines CSS-Bildes schnell erfolgt und wenige Daten zur Speicherung der relevanten Informationen benötigt werden.

Das Verfahren kann nicht für konvexe Objekte angewendet werden. Trotz deutlicher Unterschiede der Konturen, ist ein Ball nicht von einem Ei zu unterscheiden. Da in nahezu allen Objekten konkave Bereiche auftreten, stellt dies keine große Einschränkung dar. Bei Bildern mit ähnlichen Konturen, wie das Bild des Apfels und des Balls in Kapitel 2 (vgl Abb. 2.6 auf Seite 13), sind die Unterschiede der CSS-Bilder minimal. Hier kann ein Mensch die Objekte anhand der Konturen kaum unterscheiden. Da in diesem Fall auch die menschliche Wahrnehmung versagt, bestätigt dies eher die Qualität des Verfahrens.

Kapitel 5

Implementierung

Vor der eigentlichen Objekterkennung wird erläutert, wie die Kommunikation eines Menschen mit einem Bildverarbeitungssystem ablaufen kann. Der Algorithmus zur Erkennung von Objekten mit Hilfe von CSS-Abbildungen kann in drei Schritte untergliedert werden. Bei der *Segmentierung des Objekts* wird aus einem Graustufenbild die normalisierte parametrisierte Kontur des abgebildeten Objekts ermittelt. Es wird die Annahme getroffen, dass nur ein leicht segmentierbares Objekt abgebildet ist. In einem zweiten Schritt wird durch *Glättung der Konturdaten* eine Abbildung im krümmungsbasierten Skalenraum erzeugt, wobei die CSS-Maxima in eine Datenbank eingefügt werden. Ein *Vergleich zweier Objekte* ist in einem weiteren Schritt möglich. Die Implementierung dieser Schritte wird in den folgenden Abschnitten erläutert.

5.1 Kommunikation zwischen Mensch und Maschine

Suchalgorithmen, die Bilder automatisch in Kategorien einordnen oder spezielle Bilder suchen können, sind wegen ihrer Komplexität schwer zu implementieren. Es sind verschiedene Möglichkeiten denkbar, wie die Suche nach Bildern ablaufen kann. Die Eingabe von Begriffen entspricht dem klassischen Ansatz der Suche in Datenbanken, bei dem die Daten größtenteils manuell erfasst werden müssen. Die Datensätze enthalten mindestens die digitalen Bildinformationen und die Bezeichnung des abgebildeten Objekts. Eine Erweiterung der Datenbank könnte die Objekte in ein hierarchisches System einbetten, so dass die Suche nach ähnlichen Objekten möglich wird.

Alternativ kann die Anfrage an die Datenbank über die Eingabe eines Bildes realisiert werden. Dabei soll in jedem Bild genau ein Objekt dargestellt sein. Das Bild kann in Form einer Datei, einer gezeichneten Skizze oder durch Auswahl bereitgestellter Beispielbilder gewählt werden. Die Datenbank soll ähnliche Objekte liefern. Beispiele für implementierte Multimedia-Datenbanksysteme sind CueVideo, QBIC und Surfimage [9, 30, 31].

Der zweite Ansatz ist deutlich schwieriger realisierbar, da wesentliche Merkmale eines Objekts ermittelt werden müssen. Während im ersten Szenario für die Suche lediglich eine text-

basierte Anfrage verarbeitet werden muss, ist es für den zweiten Ansatz notwendig, wichtige Charakteristika im Eingabebild zu ermitteln.

Die Aufgabe der *Suche ähnlicher Objekte* wird von Menschen unterschiedlich gelöst, da der Begriff der Ähnlichkeit im Zusammenhang mit Objekten nicht klar definiert ist. Ein Mensch kann, wenn er eine Vorstellung über eine komplexe Kontur hat, diese nur schwer mit Worten wiedergeben. Die Beschreibung ist häufig sehr abstrakt und wenig anschaulich. Damit ein Bildverarbeitungssystem ähnliche Objekte suchen kann, müssen relevante Informationen über die Form und Farbe des Objekts zur Verfügung gestellt werden. Eine praktikable Benutzerschnittstelle zur Eingabe der Daten der Kontur fällt schwer. Tabellen oder Listen kommen nicht in Frage, da eine erst grobe, dann feiner werdende Strukturierung für Konturen nicht möglich ist.

In dieser Arbeit erfolgt die Eingabe und Ausgabe mit Hilfe von Bilddateien, in denen jeweils ein Objekt dargestellt ist. Auch Bildsequenzen können eingegeben werden. Sie werden als geordnete Liste von Einzelbildern interpretiert. Ziel dieser Arbeit ist es, Objekte, die eine ähnliche Kontur mit dem eingelesenen Objekt aufweisen, in der Datenbank zu suchen und auszugeben.

5.2 Segmentierung

Bei der Segmentierung werden folgende Rechenschritte für jedes Objekt durchgeführt:

- Laden eines Graustufenbildes aus einer Datei
- Umwandlung des Bildes in ein Binärbild
- Erzeugung einer Kontur für das gesuchte Objekt
- Parametrisierung und Normalisierung der gefundenen Kontur
- Abspeichern der Kontur in einer Datei

5.2.1 Organisation der Dateien

Beim Programmstart werden aus einer Konfigurationsdatei Informationen bezüglich der neu zu segmentierenden Bilder eingelesen. In der Konfigurationsdatei wird festgelegt, wie viele und welche Bilder segmentiert werden sollen, ebenso in welchen Verzeichnissen die Quellen- und Zieldateien gesucht bzw. abgelegt werden. Der Inhalt einer möglichen Konfigurationsdatei ist in Tabelle 5.1 auf der nächsten Seite dargestellt. Die Datenbank wird zum Zeitpunkt der Segmentierung nicht benötigt, da erst in einem späteren Schritt die Werte der CSS-Maxima berechnet und gespeichert werden.

Bei einem Programmablauf können mehrere Bilder segmentiert werden. Die Verarbeitung der Bilder erfolgt sequenziell, d. h. eine Parallelisierung der Prozesse findet nicht statt. Der

Merkmal	**Wert**
Anzahl der einzulesenden Bilder	3
Pfad für Eingabedateien	data/
Pfad für Ausgabedateien	result/
Anzahl der Quellenbilder	1
Bezeichnung 1	Apfel
Bezeichnung 2	...

Tabelle 5.1: Beispiel einer Konfigurationsdatei für die Segmentierung.

Nr.	**Bezeichnung**	**Eingabe/ Ausgabe**	**Bildtyp**	**Beschreibung**
1	Apfel	Eingabe	Graustufen	Originalbild
2	ApfelK	Eingabe (opt.)	Graustufen	gefiltertes Bild
3	Apfel-Kontur	Ausgabe	Binärbild	ursprüngliche Kontur
4	Apfel-KonturPixel	Ausgabe	Binärbild	Kontur in Pixelform
5	Apfel-KonturBild	Ausgabe	Binärbild	Konturpixel werden durch Linien verbunden
6	Apfel-Bild	Ausgabe	Graustufen	Originalbild
7	Apfel-Vektor	Ausgabe	Fließkomma	Vektordaten der Kontur

Tabelle 5.2: Dateistruktur bei der Segmentierung.

Wert *Anzahl der einzulesenden Bilder* gibt an, wie viele Bilder segmentiert werden sollen. Für jedes Bild muss eine *Bezeichnung* angegeben sein, die dem Dateinamen des zu segmentierenden Bildes entspricht.

Bei schwer segmentierbaren Objekten ermittelt das verwendete Segmentierungsverfahren die Kontur eines Objekts möglicherweise fehlerhaft. Zusätzliche Filter können das Ergebnis verbessern. Sollen exakte Krümmungswerte der Kontur berechnet werden (Evolution der Krümmung), so wird neben dem *gefilterten Bild* zusätzlich das *Originalbild* benötigt. Der Wert *Anzahl der Quellenbilder* bestimmt, ob in jedem Segmentierungsschritt ein oder zwei Bilder eingelesen werden.

Die Ausgabe des Segmentierungsprogramms besteht aus fünf Bilddateien, bei denen überflüssige Ränder entfernt worden sind. Tabelle 5.2 gibt an, welche Eingabe- und Ausgabedateien im Segmentierungsschritt verwendet werden. Die Eingabedateien werden nach der Konfigurationsdatei im Verzeichnis *data* gesucht, die Ausgabedateien im Verzeichnis *result* gespeichert. Die Dateien mit den Nummern 3-5 entsprechen den verschiedenen Konturen in Abbildung 3.1 auf Seite 16.

Die wichtigste Ausgabedatei, die im Segmentierungsschritt erzeugt wird, ist die Vektordatei,

da sie die Basis für weitere Berechnungen bildet. Sie speichert die x-und y-Koordinaten der Konturpunkte in Form eines Feldes. Optional können die Krümmungswerte in jedem Konturpunkt mit in die Datei geschrieben werden.

Alle Dateien werden in einem dem PGM-Format[1] ähnlichen Format gelesen und gespeichert. Das hier gewählte Dateiformat ist wie folgt aufgebaut: Nach den Bildinformationen folgen die eigentlichen Bilddaten in binärer Form. Der Header der Datei wird zeilenweise in ASCII-Zeichen gespeichert und kann durch Textverarbeitungsprogramme gelesen und geändert werden. Nach einer beliebigen Anzahl an optionalen Informationszeilen folgt die Größenangabe des Bildes in x- und y-Richtung. Anschließend wird die Anzahl der möglichen Farb- bzw. Graustufenwerte gespeichert. In dieser Arbeit wurden für alle Bilder 256 Graustufen verwendet, die durch Integer-Werte codiert werden. Lediglich bei dem Konturvektor handelt es sich um eine Feld mit Fließkommazahlen.

5.2.2 Bestimmung der Kontur eines Objekts

Für die Segmentierung wird angenommen, dass sich Objekt und Bildhintergrund in ihrer Helligkeit deutlich unterscheiden. Üblicherweise wird ein dunkles Objekt auf hellem Hintergrund abgebildet, der umgekehrte Fall ist jedoch auch möglich. Wenn ein Objekt an den Rand eines Bildes grenzt, dann können nicht mehr alle Nachbarpixel des Objekts betrachtet werden und es wird möglicherweise falsch segmentiert. Um Probleme mit Bildrändern zu vermeiden, wird um das eingelesene Graustufenbild ein weißer Rand mit einer Breite von drei gelegt.

Der Helligkeitswert H^*, der bei der Umwandlung in ein Binärbild bestimmt, wann ein Pixel Schwarz oder Weiß gesetzt werden soll, berechnet sich nach der Gleichung 5.1. Dunklere Pixelwerte als H^* werden zu Schwarz, hellere Werte zu Weiß. $\bar{H}$ bezeichnet die durchschnittliche Helligkeitswert des Bildes, H_W die Helligkeit der Farbe Weiß.

$$H^* = \frac{1}{2} \cdot (\bar{H} + H_W) \tag{5.1}$$

Mit diesem Verfahren werden deutlich mehr Flächen schwarz gefärbt als gewünscht (vgl. Abb. 5.1 auf Seite 56, links oben). Kleine Objekte und Rauschpixel werden in einem nächsten Schritt entfernt. Auch Pixel, die in ihrer Vierer-Nachbarschaft nur ein ähnlich helles Nachbarpixel besitzen, werden gelöscht. Das Verfahren reduziert das Rauschen im Bild, so dass nur noch größere schwarze oder weiße Flächen auftreten. Das Ergebnis dieses zweiten Schritts kann in Abbildung 5.1 auf Seite 56 (rechts oben) betrachtet werden.

Überall dort wo schwarze und weiße Flächen aneinander grenzen, werden Konturen ermittelt (vgl. Abb. 5.1 auf Seite 56, mitte links). Diese werden fortlaufend numeriert und jede Kon-

[1] Portable Graymap

tur erhält ihren eigenen Helligkeitswert (vgl. Abb. 5.1 auf der nächsten Seite, mitte rechts). Die Kontur, die die meisten Pixel enthält, wird ausgewählt, alle anderen Konturen werden gelöscht.

Da Unebenheiten in den Randbereichen der Kontur auftreten, kann sie noch nicht in eine parametrisierte Form überführt werden (vgl. Abb. 5.1 auf der nächsten Seite, unten links). Alle Pixel, die außerhalb der Kontur liegen, werden durch einen Füllalgorithmus markiert. Ausgehend von einem weißen Pixel am Rand des Bildes werden alle weißen Bereiche, die innerhalb einer Vierer-Nachbarschaft erreichbar sind, markiert. Die nicht markierten Pixel werden durch schwarze Pixel ersetzt. Das gesuchte Objekt besitzt jetzt eine zusammenhängende Fläche schwarzer Pixel, in der keine weißen Flächen mehr auftreten. Die Randpixel dieses schwarzen Objekts werden in einem letzten Schritt bestimmt (vgl. Abb. 5.1 auf der nächsten Seite, unten rechts).

Um Speicherplatz zu sparen, wird das Bild anschließend verkleinert, bis ein Rand von der Breite eines Pixel um das Objekt liegt. Die Breite der abgeschnittenen Ränder werden gespeichert, damit die Position des Objekts im ursprünglichen Graustufenbild wiedergefunden werden kann. Die wichtigsten Schritte der Segmentierung werden in Abbildung 5.1 auf der nächsten Seite zusammengefasst dargestellt.

5.2.3 Parametrisierung der Kontur

Nach der Segmentierung zeichnen sich Randpixel eines Objekts dadurch aus, dass in der Vierer-Nachbarschaft jedes Pixel genau zwei weitere Randpixel liegen. Ein beliebiges Randpixel wird ausgewählt und als Startpixel festgelegt. Anschließend wird die Kontur im Uhrzeigersinn durchlaufen, und jedes gefundene Randpixel wird mit seiner x- und y-Koordinate in einem Feld gespeichert.

Da alle Objekte durch eine einheitliche Anzahl an Konturpunkten beschrieben werden sollen, muss das Feld mit den Wertepaaren der Kontur normalisiert werden. Der Abstand zweier benachbarter Konturpunkte beträgt wegen ihrer Vierer-Nachbarschaft immer Eins. Der Faktor F legt fest (vgl. Gl. 5.2), wie groß der Abstand zwischen zwei neuen Konturpunkten sein muss, damit jede Kontur durch eine gleiche Anzahl von Punkten beschrieben wird.

$$F = \frac{\text{tatsächliche Anzahl der Konturpunkte}}{\text{gewünschte Anzahl der Konturpunkte}} \tag{5.2}$$

Wird eine Normalisierung mit einem Faktor F größer eins durchgeführt, dann ist der Abstand der neu ausgewählten Konturpunkte unterschiedlich groß. Strukturen, die vor der Normalisierung zwischen zwei Punkten existiert haben, können nicht mehr erkannt werden. Üblicherweise liegen wenige Pixel zwischen den ausgewählten Konturpunkten, so dass nur sehr feine Bereiche durch die Normalisierung geglättet werden.

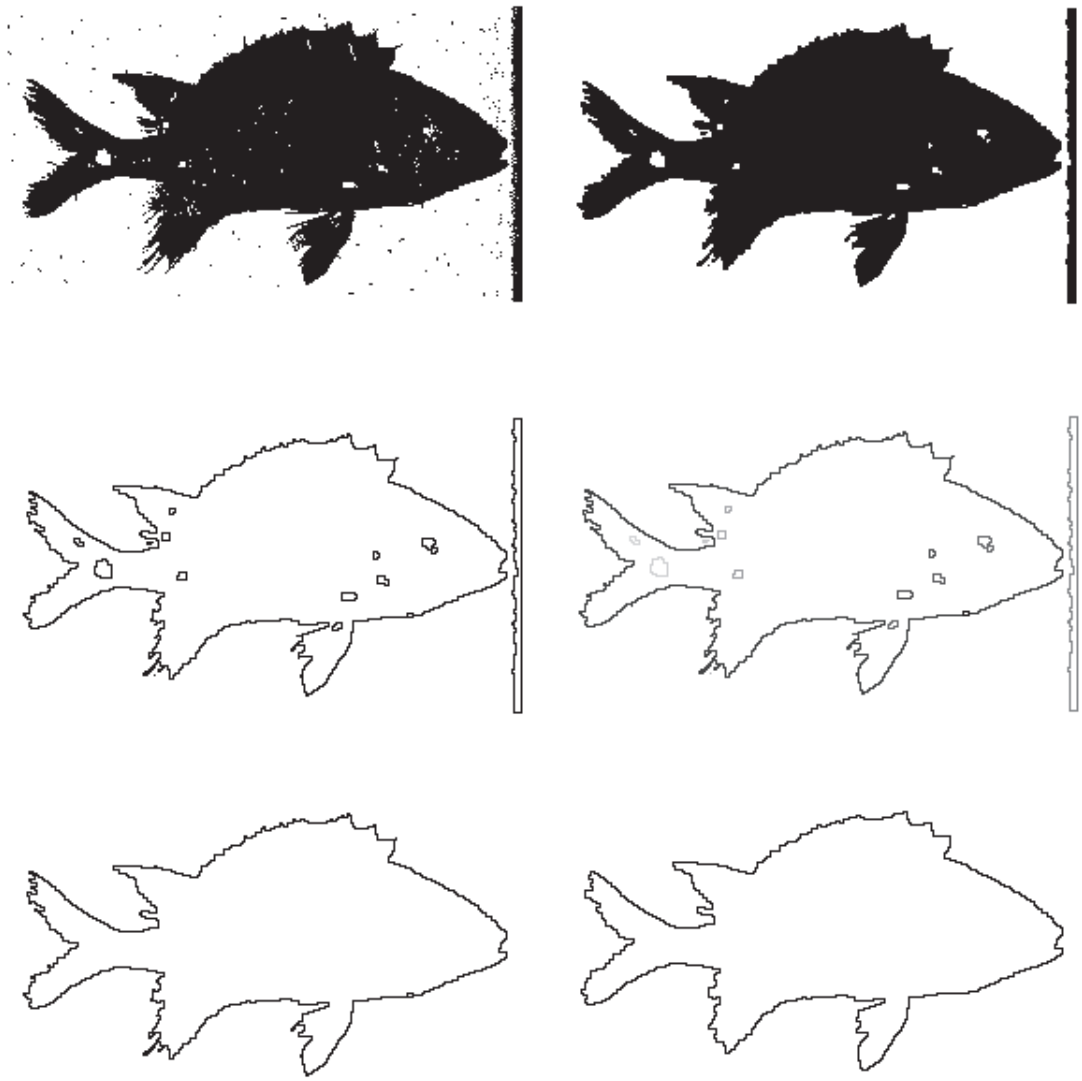

Abbildung 5.1: Einzelne Phasen der Segmentierung. In der oberen Zeile wird nach der Umwandlung in ein Binärbild Rauschen entfernt. In der mittlere Zeile (links) werden alle möglichen Konturen ausgewählt. Die einzelnen Konturen werden rechts durchgängig numeriert und durch unterschiedliche Graustufenwerte dargestellt. In der unteren Zeile wird die größte Kontur ausgewählt und geringe Unebenheiten werden entfernt.

5.2.4 Speicherung der Daten

Die ermittelten Daten werden nach der Parametrisierung und Normalisierung der Kontur in Form von Bilddateien und einer Vektorliste gespeichert. Alle neuen Bilder unterscheiden sich in ihrer Bildgröße vom ursprünglichen Bild, wobei die Größe von der Höhe und Breite des selektierten Objekts abhängt.

Die Vektorliste besitzt für jedes Bild immer dieselbe Größe. Eine Größe von knapp 4 KByte ergibt sich bei einer Wahl von 250 Wertepaaren zur Darstellung der Kontur.

5.2.5 Wahl der Parameter

Beim Vergleich zweier Objekte werden ähnliche Konturen durch niedrige Zahlen, unterschiedliche Konturen durch hohe Zahlen bewertet. Der Algorithmus liefert *gute Ergebnisse*, falls der Wert bei sehr ähnlichen Objekten minimiert und gleichzeitig bei unterschiedlichen Objekten groß ist.

Die Anzahl der Wertepaare zur Beschreibung der Kontur beeinflusst die Rechenzeit und die Genauigkeit des Verfahrens. Je mehr Wertepaare ausgewählt werden, desto länger benötigt insbesondere die Gaußglättung bei der Berechnung der CSS-Abbildung, desto genauer wird aber auch die Kontur beschrieben.

Für 200 oder weniger Wertepaare zur Beschreibung der Kontur ergeben sich schlechtere Ergebnisse beim Objektvergleich, da größere Bereiche der Kontur verloren gehen und die Ungenauigkeit zunimmt. Eine Anzahl von 250 Wertepaaren stellt einen guten Kompromiss zwischen Genauigkeit und Rechenzeit dar und liefert sehr gute Ergebnisse für die Objekterkennung.

5.2.6 Merkmale der Segmentierung

Getestet wurden alle Programme unter Windows 98 auf einem Intel-PC mit Celeron 466 MHz Prozessor und GNU C++-Compiler in der Version 2.81. Als zweiter Testrechner diente eine SUN Ultra-SPARC 2 mit 300 MHz und der Version 2.95.2 des GNU C++-Compilers. Der Intel-PC ist in allen Berechnungen durchgehend 30 Prozent schneller. Im folgenden werden nur die Rechenzeiten des Algorithmus auf der SUN angegeben. Eine detaillierte Beschreibung der Rechenzeiten ist im Anhang aufgelistet.

Die Auflösung der Bilder, die zwischen 150 und 800 Bildpunkten in x- und y-Richtung beträgt, ist ausschlaggebend für die Rechenzeit des Segmentierungsalgorithmus, der für die 40 Beispielobjekte der Datenbank etwa fünf Minuten Rechenzeit benötigt. Die vielen Lese- und Schreiboperationen beeinflussen maßgeblich die Programmlaufzeit. Für die gewählten 40 Testbilder werden über 80 Dateien von der Festplatte geladen und 200 Dateien gespeichert.

Ein wesentlicher Kritikpunkt des Segmentierungsverfahrens ist seine Fehleranfälligkeit. Die Graustufenwerte der Bilder ermöglichen keine Unterscheidung Objekt und zugehörigem Schatten. Bessere Verfahren zur Segmentierung könnten sich durch die Verwendung von Farbbildern

Merkmal	**Wert**
Anzahl der einzulesenden Bilder	1
Pfad für Eingabedateien	data/
Pfad für Ausgabedateien	result/
Bezeichnung 1	Elefant
Bezeichnung 2	...

Tabelle 5.3: Beispiel einer Konfigurationsdatei für die Berechnung der Maxima der CSS-Abbildung.

oder der Betrachtung von Bildsequenzen ergeben. Durch die Analyse der Bewegung eines Objekts kann eine deutlich bessere Segmentierung erfolgen [27, 14]. Weitere kantenverstärkende Verfahren [36], wie beispielsweise Perona-Malik-Filter, könnten vor dem Segmentierungsschritt angewendet werden. Es dürfen nur Verfahren eingesetzt werden, die die Kontur nicht modifizieren.

5.3 Maxima der CSS-Abbildung

Ähnlich wie bei der Segmentierung, wird der Programmablauf zur Berechnung der CSS-Abbildung über eine Konfigurationsdatei gesteuert. Der Aufbau dieser Datei ist beispielhaft in Tabelle 5.3 dargestellt.

Es können mit einem Programmablauf beliebig viele Konturdaten eingelesen und deren CSS-Maxima in der Datenbank gespeichert werden. Der Ablauf zur Berechnung der Maxima setzt sich aus folgenden fünf Schritten zusammen:

- Laden des Konturvektors eines Bildes
- Berechnung der CSS-Abbildung
- Schließen der Lücken in der CSS-Abbildung
- Ermittlung der Maxima der CSS-Abbildung
- Speichern der Wertepaare der CSS-Maxima in der Datenbank

5.3.1 Berechnung der CSS-Abbildung

Um aus den Konturdaten eine entsprechende Abbildung im krümmungsbasierten Skalenraum zu berechnen, wird das Verfahren zur *Evolution der Kontur* eingesetzt. Die in dem Konturvektor gespeicherten Wertepaare werden entlang der x- und der y-Koordinate geglättet. Zur Glättung wird ein Gaußfilter mit einer Maskenbreite von drei verwendet. Auch das Verfahren zur *Evolution der*

Krümmungsfunktion wurde getestet. Der Nachteil dieses zweiten Verfahrens liegt in der Notwendigkeit, die Krümmungswerte anhand eines Graustufenbildes zu berechnen. Dieses Graustufenbild wird durch den Segmentierungsprozess zur Verfügung gestellt. Obwohl die Glättung der Krümmungsfunktion geringfügig schneller als die Glättung der Konturdaten abläuft, kompensiert die zusätzliche Dateieingabe den gewonnenen Zeitvorteil. Bezüglich der Qualität der Objekterkennung konnten keine Unterschiede zwischen beiden Verfahren festgestellt werden. Die CSS-Abbildungen unterscheiden sich durch die unterschiedlich schnelle Glättung beider Verfahren. Deshalb dürfen die verschiedenen Verfahren nicht miteinander kombiniert werden.

Die Breite des Feldes, in dem die CSS-Maxima gespeichert werden, entspricht der Anzahl der Wertepaare im Konturvektor und ist mit 250 Werten festgelegt. Für jedes Konturpixel wird die letzte Nullstelle der Krümmungsfunktion gespeichert, die während der Gaußglättung aufgetreten ist. Die Kontur wird so lange geglättet, bis in der Krümmungsfunktion keine Nullstellen mehr auftreten. Bei keinem der im nächsten Kapitel beschriebenen 40 Testbilder werden mehr als 1400 Iterationen benötigt. In jedem Iterationsschritt erfolgt nach einer einmaligen Gaußglättung der Konturdaten die Berechnung des Vorzeichens der Krümmungsfunktion.

Die Gaußglättung wird für die x- und die y-Koordinaten getrennt durchgeführt. Der approximierte Wert der Krümmungsfunktion W_i wird in jedem Punkt i eines Punktes wird durch Einsetzen der Konturpunkte in die Gleichung 5.3 bestimmt (vgl. auch Gleichung 4.5 auf Seite 47 und 4.8 auf Seite 48).

$$W_i = (X_{i+1} - X_i) * (Y_{i+2} - Y_i) + (Y_{i+1} - Y_i) * (X_i - X_{i+2}) \tag{5.3}$$

Eine Nullstelle im Punkt i der Krümmungsfunktion liegt genau dann vor, wenn für zwei benachbarte Krümmungswerte gilt: $W_i \cdot W_{i+1} < 0$. Im Falle einer Nullstelle wird an die Position i im CSS-Feld die Zahl des aktuellen Iterationsschrittes eingetragen.

5.3.2 Reduktion der Lücken in der CSS-Abbildung

Wie in Kapitel 4.4.1 auf Seite 43 erläutert wurde, können Lücken im Bereich der Maxima der CSS-Abbildungen durch die Glättung des Gaußfilters auftreten. Nach der Berechnung der CSS-Abbildung werden diese Lücken geschlossen. Wenn der Wert des CSS-Feldes an der Position i nahe bei Null liegt und die beiden vorherigen Werte des Feldes monoton steigen, wird angenommen, dass eine Lücke an der Stelle i vorhanden ist. Sie wird geschlossen, indem die Umgebung rechts von der Position i betrachtet wird und die dort auftretenden Werte die Werte der Lücke ersetzen.

Objektbezeichnung	CSS-Maxima
Giraffe	862 805 283 144 115 23 19 7 6 5
Fisch	436 322 258 121 98 84 57 7 5 4
Ball	9 9 7 4 3 3 3 3 2 2

Tabelle 5.4: Beispiel für die Werte der CSS-Maxima dreier ausgewählter Objekte.

5.3.3 Ermittlung der CSS-Maxima

In einem abschließenden Schritt werden aus dem eindimensionalen CSS-Feld die Maxima abgelesen. In Tabelle 5.4 werden für drei in dieser Arbeit schon vorgestellte Objekte die jeweils 10 größten Maxima aufgelistet. Auch runde Objekte, wie beispielsweise der Ball in Abbildung 2.6 auf Seite 13, besitzen ganz kleine konkave Bereiche. Die CSS-Maxima des Balls in Tabelle 5.4 bestätigen dieses.

Es wird deutlich, dass sich die Werte der letzten Maxima kaum noch voneinander unterscheiden, da sie sehr kleine konkave Bereiche beschreiben. Bei einer Anzahl von 20 Maxima würden bei vielen Objekten die letzten Werte Null annehmen. Für die meisten verwendeten Objekte reichen die ersten 8 Werte zur Beschreibung der wesentlichen konkaven Bereiche aus. Um bei unbekannten Objekten keine relevanten konkaven Bereiche auszulassen, wurde der Wert für die Anzahl der zu speichernden Maxima auf 10 festgelegt.

Die CSS-Maxima werden absteigend sortiert gespeichert. Nach der Ermittlung eines Maximum wird der monoton steigende Bereich vor und der monoton fallende Bereich nach dem ausgewählten Maximum auf Null gesetzt. Alle Punkte bis zu dem nächsten lokalen Minimum erhalten dabei den Wert Null. Weisen mehrere benachbarte Punkte gleichhohe Werte auf, so wird der mittlere Punkt ausgewählt. Beim Schließen von Lücken tritt dieser Fall regelmäßig auf. Neben dem Wert des Maximums wird auch dessen Position gespeichert. Die ganzzahligen Wertepaare werden anschließend mit dem Namen des Objekts in der Datenbank gespeichert.

5.3.4 Laufzeitverhalten

Um die CSS-Maxima der 40 Beispielobjekte aus dem nächsten Kapitel zu ermitteln und zu speichern, werden insgesamt 45 Sekunden benötigt. Eine Verbesserung der Laufzeit könnte dadurch erreicht werden, dass die Daten der Konturen und der CSS-Maxima im Arbeitsspeicher gehalten werden. Die Lese- und Schreibzugriffe auf die Festplatte und die Übertragung der Dateien über das Netz beanspruchen einen erheblichen Teil der Rechenzeit.

Nach wenigen Glättungen mit dem Gaußfilter reduzieren sich die konkaven Bereiche auf zwei oder drei. Für die weitere Glättung könnte eine größere Filtermaske verwendet werden, um weniger häufig die Nullstellen der Krümmungsfunktion berechnen zu müssen. Dies würde zu zusätzlichen Problemen bei einem späteren Vergleich zweier CSS-Felder führen. Auch nimmt bei einer größeren Filtermaske die Breite der Lücken in den CSS-Abbildungen zu. Bei dynami-

Objekt 1	Objekt 2	Differenz
Fisch	Fisch	0
Fisch	gedrehter Fisch	28
Fisch	Mensch	192
Mensch	Fisch	192
Mensch	Dreirad	810

Tabelle 5.5: Beispiel für die bewertete Differenz zweier Objekte.

scher Anpassung der Maskengröße des Gaußfilters reduziert sich die Rechenzeit nur minimal. Sie bewirkt aber eine deutliche Zunahme der Komplexität und findet daher in diesem Programm keine Anwendung.

5.4 Vergleich zweier Objekte

Die Aussage, ob zwei Objekte eine ähnliche Kontur besitzen, wird durch einen Vergleich der CSS-Maxima getroffen. Die Bewertung der Unterschiede zweier Konturen erfolgt in Form einer positiven ganzen Zahl. Je niedriger dieser Wert ist, desto mehr ähneln sich die beiden Objekte. Ein Beispiel für die Differenz zwischen identischen und gedrehten Objekte ist in Tabelle 5.5 abgebildet.

Die für einen Vergleich zweier CSS-Felder benötigten Informationen stammen aus einer Konfigurationsdatei (vgl. Tabelle 5.6 auf der nächsten Seite). Die in der Datenbank gespeicherten CSS-Maxima werden im folgenden als *Daten des Modells* bezeichnet. Die *Daten der zu analysierenden Bilder* enthalten die unbekannten Objekte, die mit den Daten des Modells verglichen werden sollen. Sie können entweder als Kontur oder als CSS-Maxima eingelesen werden, wobei die zweite Alternative die Rechenzeit erheblich reduziert. In Tabelle 5.6 auf der nächsten Seite wird unter der Bezeichnung *Art der Quellendatei* der Wert Null für die Eingabe eines Vektorbildes und der Wert Eins zur Eingabe der CSS-Maxima verwendet. Damit aus einer Vektorliste die entsprechende CSS-Abbildung nicht mehrfach berechnet werden muss, werden die CSS-Maxima unter dem Pfad der Ausgabedateien gespeichert.

5.4.1 Differenz zweier Wertepaare

Bevor zwei Felder mit CSS-Maxima verglichen werden können, muss festgelegt werden, wie die Differenz D zweier Wertepaare (u_1, σ_1) und (u_2, σ_2) gemessen wird. Die Abstände zwischen den Punkten u_1 und u_2 bzw. σ_1 und σ_2 können nach Gleichung 5.4 auf der nächsten Seite berechnet werden. Ist die Breite der Krümmungsfunktion auf 250 Punkte festgelegt, so besteht zwischen den Punkten u_1 und u_2 ein Abstand von maximal 125 Punkten, da die Funktion der Krümmungswerte periodisch ist (vgl. Kapitel 4.3.1 auf Seite 33).

Merkmal	**Wert**
Anzahl der zu analysierenden CSS-Felder	25
Pfad für Eingabedateien	data/
Pfad für Ausgabedateien	result/
Art der Quellendatei	1
Bezeichnung 1	Auto057
Bezeichnung 2	...

Tabelle 5.6: Beispiel einer Konfigurationsdatei zum Vergleich von Objekten.

$$\begin{aligned} d_u &= min(|u_1 - u_2|, 250 - |u_1 - u_2|) \\ d_\sigma &= |\sigma_1 - \sigma_2| \end{aligned} \tag{5.4}$$

Die Differenz zweier Wertepaare wird wie der Abstand von Punkten im zweidimensionalen Koordinatensystem berechnet:

$$D = \sqrt{d_u \cdot d_u + d_\sigma \cdot d_\sigma} \tag{5.5}$$

Unterschiede in Bezug auf die Stärke des konkav gekrümmten Bereichs werden genauso stark gewertet wie Unterschiede der Position der Krümmung, d. h. d_u und d_σ werden gleich stark gewertet. In Abbildung 5.2 auf der nächsten Seite werden vier CSS-Abbildungen gezeigt, von denen jeweils zwei in einem Koordinatensystem abgebildet werden. Die Differenzen D dieser Wertepaare sind im oberen und unteren Koordinatensystem durch einen grauen Balken markiert worden und besitzen den selben Wert.

5.4.2 Normalisierung des ersten Wertepaares

Die ersten Wertepaare der beiden CSS-Listen enthalten jeweils den Wert und die Position des stärksten gekrümmten Bereichs der Kontur. Damit sich die Differenzen zweier CSS-Listen bei gedrehten Objekten nicht unterscheiden, muss die Position des am stärksten gekrümmten Bereichs im Modell mit der entsprechenden Position des unbekannten Objekts übereinstimmen. Die CSS-Abbildung des Modells wird entlang der x-Achse verschoben, bis die Positionen der beiden stärksten Krümmungen übereinstimmen. Diese Positionen werden im Modell wird mit $u_0(M)$ und im unbekannten Bild mit $u_0(B)$ bezeichnet. Abbildung 5.3 auf Seite 64 zeigt zwei CSS-Abbildungen, bei denen die untere entlang der x-Achse verschoben ist.

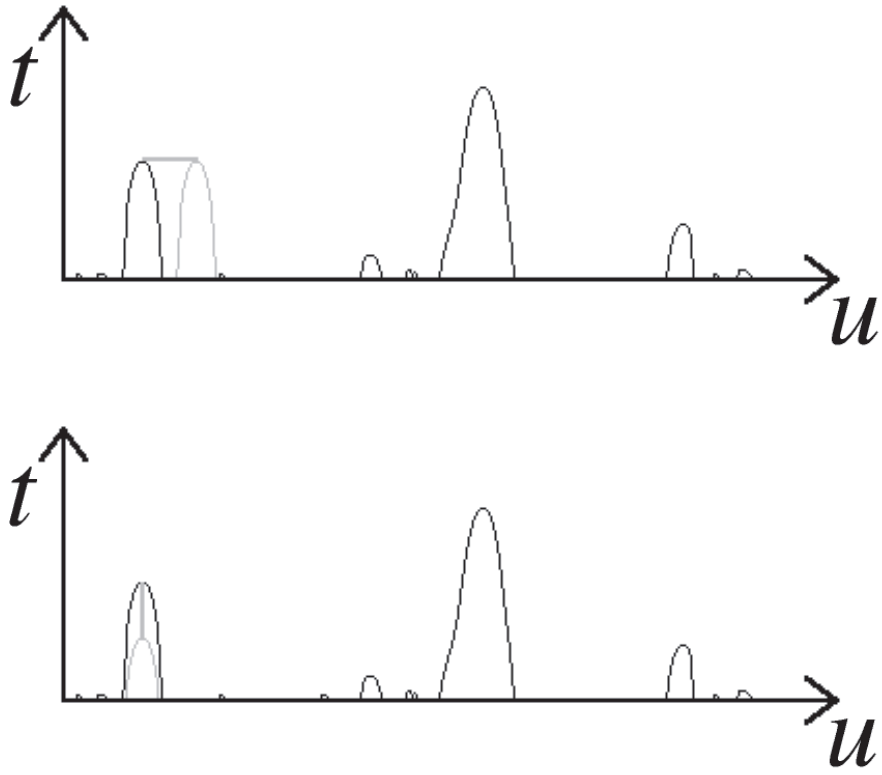

Abbildung 5.2: Berechnung der Differenz zweier CSS-Maxima. In jeder der beiden Grafiken werden zwei unterschiedliche CSS-Abbildungen in grau und schwarz dargestellt. Sie unterscheiden sich jeweils in genau einem konkaven Bereich. In der oberen Abbildung ist die Position des zweitstärksten konkaven Bereichs verschoben, in der unteren Abbildung ist dieser konkave Bereich weniger stark ausgeprägt. Die Differenzen der CSS-Maxima sind durch einen Balken markiert worden und in beiden Fällen identisch.

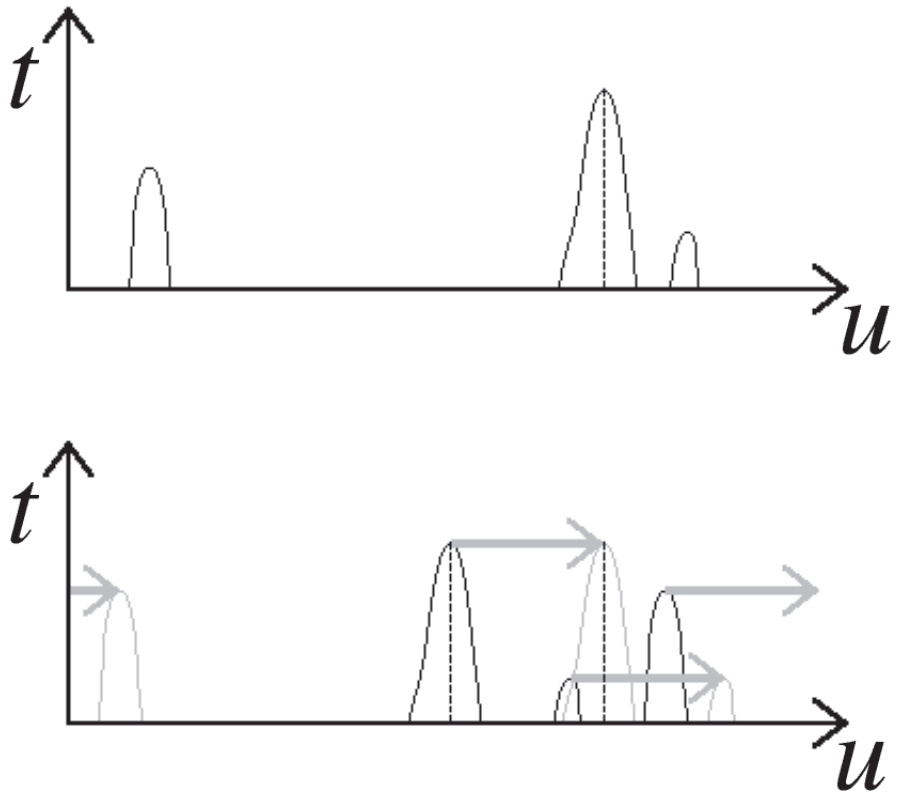

Abbildung 5.3: Normalisierung der CSS-Abbildung des Modells. Die obere Abbildung zeigt die CSS-Abbildung des unbekannten Objekts. Die ursprüngliche CSS-Abbildung des Modells ist in der unteren Abbildung schwarz, die normalisierte CSS-Abbildung ist grau eingezeichnet worden. Der Wert der Verschiebung V wird durch die Differenz von $u_0(M)$ und $u_0(B)$ bestimmt.

Die Berechnung des Wertes V, um den die Positionswerte der CSS-Liste des Modells verschoben werden, erfolgt mit der Gleichung 5.6). Die Positionen der CSS-Maxima des Modell werden mit $u_i(M)$, die des unbekannten Bildes mit $u_i(B)$ bezeichnet.

$$\begin{aligned} V &= u_0(M) - u_0(B) \\ u_i(M_{neu}) &= \Big(u_i(M) + V\Big) \quad MOD \quad 250 \end{aligned} \tag{5.6}$$

Nachdem die Positionen der größten CSS-Maxima beider CSS-Listen übereinstimmen, kann die Differenz zweier CSS-Listen berechnet werden.

5.4.3 Differenz zweier CSS-Listen

Die Differenz zweier CSS-Listen berechnet sich aus der Summe der Differenz von zehn Wertepaaren. Zu jedem Wertepaar der ersten CSS-Liste wird aus der zweiten Liste das Paar gesucht, das unter den bisher nicht berücksichtigten Wertepaaren die geringste Differenz aufweist. Die Reihenfolge wird durch die Größe der Sigmawerte der ersten CSS-Liste bestimmt. Beginnend mit dem größten Sigma werden die Wertepaare der Liste in absteigender Reihenfolge ausgewählt. Tabelle 5.7 auf der nächsten Seite zeigt die Berechnung der Differenz zweier CSS-Listen mit drei Wertepaaren. Die Wertepaare, die für die genaue Berechnung zugrunde gelegt werden, sind in Gleichung 5.7 aufgeführt.

$$\begin{aligned} \Gamma_{normalisiertes\ Modell} &= (139,178)(234,98)(107,69) \\ \Gamma_{unbekanntes\ Objekt} &= (139,186)(121,141)(11,54) \end{aligned} \tag{5.7}$$

Bei dem Verfahren handelt es sich um eine Heuristik, die eine möglichst geringe Differenz zweier CSS-Listen ermittelt. Die geringste Differenz zweier CSS-Listen wird nicht sicher gefunden, da dazu alle Kombinationen der Wertepaare des Modells mit dem unbekannten Objekt berechnet werden müssen.

Wenn die Modelldaten mit den zu analysierenden Daten vertauscht werden, treten gegebenenfalls unterschiedliche CSS-Differenzen auf.

Um das Problem zu umgehen, werden beide mögliche Differenzen berechnet und das Minimum wird ausgewählt.

Die Erkennung gespiegelter Objekte erfordert weitere Verfahrensschritte. Um gespiegelte Objekte zu erkennen, müssen die Positionen u_i der Wertepaare gespiegelt werden. Für die Wertepaare (u_i, σ_i) ergibt sich ein neuer Wert von $(250 - u_i, \sigma_i)$. Die Spiegelung von 4 CSS-Maxima

aktuelles Wertepaar des normalisierten Modells	**bisher nicht berücksichtigtes Wertepaar des unbekannten Objekts**	d_u	d_σ	**Differenz der Wertepaare**
(139, 178)	**(139, 186)**	**0**	**8**	**8**
	(121, 141)	18	37	41
	(11, 54)	122	124	173
(234, 98)	(121, 141)	113	43	120
	(11, 54)	**27**	**44**	**51**
(107, 69)	**(121, 141)**	**14**	**72**	**73**
Gesamte Differenz der beiden CSS-Listen				**132**

Tabelle 5.7: Rechenbeispiel zur Ermittlung der Differenz zweier Objekte. Die Wertepaare des Modells und des unbekannten Objekts stammen aus Gleichung 5.7 auf der vorherigen Seite.

mit der üblichen Breite der Krümmungsfunktion von 250 Punkten wird im folgenden Beispiel gezeigt:

$$\begin{aligned} \Gamma_{normal} &= (139,178)(204,106)(107,69)(63,7) \\ \Gamma_{gespiegelt} &= (111,178)(46,106)(143,69)(187,7) \end{aligned} \tag{5.8}$$

Durch den Tausch und die Spiegelung der beiden CSS-Listen muss die Berechnung in vier Variationen durchgeführt werden. Der minimale Wert dieser vier Berechnungen bestimmt die Differenz der beiden CSS-Datensätze. Trotz dieser vier Berechnungen handelt es sich um eine sehr schnelle Heuristik, die besonders die stark gekrümmten Bereiche berücksichtigt. Die weniger stark gekrümmten Bereiche werden später miteinander kombiniert.

5.4.4 Ausgabe der Ergebnisse

Für jeden Datensatz der Datenbank wird die Differenz zum unbekannten Objekt ermittelt. Ausgegeben wird eine Tabelle mit der Bezeichnung des Datensatzes und der entsprechenden Differenz. Die Objekte werden aufsteigend sortiert, d. h. sehr ähnliche Konturen werden an erster Stelle ausgegeben. Eine mögliche Ausgabe einer kleinen Datenbank mit 5 Elementen ist in Tabelle 5.8 auf der nächsten Seite in der mittleren Spalte abgebildet. Bei dem zu analysierenden Objekt handelt es sich um das Bild eines Menschen.

Bei einem Vergleich mit Bildsequenzen muss die Ausgabe erweitert werden. Neben den Differenzen zu jedem einzelnen Bild, wird ein Durchschnitt über alle Differenzen zu einem

Bezeichnung	**Differenz zu einem ausgewählten Einzelbild**	**Differenz über Bildsequenzen**
Mensch	140	206
Fisch	174	224
Katze	237	277
Auto	212	338
Apfel	588	321

Tabelle 5.8: Beispiel für die Bildschirmausgabe nach dem Vergleich einzelner Bilder und Bildsequenzen. In dem Einzelbild und der Bildsequenz werden Menschen abgebildet.

Verfahren	**Rechenzeit auf der SUN**	**kritischer Parameter für die Rechenzeit**
Segmentierung	5 Minuten	Bildgröße
Berechnung der CSS-Maxima	45 Sekunden	Iterationen der Gaußglättung (größter konkaver Bereich der Kontur)
Vergleich mit 40 Bildern der Datenbank	2 Sekunden	Anzahl der CSS-Maxima

Tabelle 5.9: Rechenzeit der Verfahren.
50 Objekte wurden mit jedem Verfahren getestet. Es handelt sich um Bildsequenzen aus Videos, die in ihrer Auflösung auf 400 Pixel beschränkt sind. Die Größe der Bilder beeinflußt insbesondere die Dauer der Segmentierung.

Element der Datenbank gebildet. Die Werte in der rechten Spalte der Tabelle 5.8 geben die durchschnittliche Differenz über alle Bilder der Bildsequenz wieder.

5.5 Merkmale des implementierten Algorithmus

Der Vergleich der einzelnen Objekte kann sehr schnell durchgeführt werden. In weniger als einer Sekunde wird eine Bildsequenz aus 25 Bildern mit den 40 Elementen der Datenbank verglichen. Hierzu müssen $25 * 40$ Differenzen zwischen CSS-Listen berechnet werden, wobei jede Differenz in vier Variationen berechnet wird. Die Berechnung ist nur dann so schnell möglich, wenn die CSS-Maxima der zu analysierenden Bilder im Vorfeld berechnet wurden.

Die Rechenzeit der einzelnen Schritte ist in einer Übersicht in Tabelle 5.9 dargestellt. Zusätzlich wird angegeben, welche Parameter die Rechenzeit am stärksten beeinflusst. Anhand der Zahlen wird deutlich, dass das Segmentierungsverfahren am meisten Zeit benötigt. Eine erste Verbesserung des Algorithmus fordert den Einsatz eines schnelleren und stabileren Verfahrens zur Segmentierung.

Kapitel 6

Experimentelle Ergebnisse

In diesem Kapitel werden die Ergebnisse vorgestellt, die bei Anwendung des Algorithmus zur Objekterkennung auf Bildsequenzen erzielt werden. Technische Details und die Darstellung der verwendeten Bildsequenzen können dem Anhang entnommen werden. Im nächsten Abschnitt wird die Datenbank mit den 40 unterschiedlichen Testobjekten vorgestellt. Nach der Analyse von Einzelbildern werden Bildsequenzen analysiert. Anschließend wird untersucht, inwieweit eine Objekterkennung möglich ist.

6.1 Einrichtung einer Testdatenbank

Die Konturen der 40 Objekte der Testdatenbank wurden so ausgewählt, dass sie für die menschliche Wahrnehmung besonders leicht erkennbar sind. Die überwiegende Anzahl der Bilder zeigt Lebewesen oder Fahrzeuge in kanonischer Sicht. Die Bilder stammen aus Fotos, Zeichnungen oder wurden aus Filmsequenzen ausgeschnitten. Die Konturen der Datenbank und der zu analysierenden Bildsequenzen stammen aus unterschiedlichen Quellen, d. h. kein Objekt der Datenbank wurde aus einer der zu analysierenden Bildsequenzen entnommen. Existieren mehrere ähnliche Objekte, so unterscheiden sich die Perspektiven, aus denen sie betrachtet werden. Ein Mensch wird beispielsweise im Profil, in einer Frontalansicht oder von schräg vorne gezeigt.

Die meisten gewählten Objekte verändern ihre Form innerhalb ihrer Bildsequenz. Bei beweglichen Objekten tritt eine Verformung auf. Sie ist aber auch bei starren Objekten durch Änderung der Perspektive möglich. Bewegen sich Tiere oder Menschen, so verformt sich durch die unterschiedlichen Positionen der Arme und Beine die Kontur. Zu starren Objekten zählen Fahrzeuge. In Bildsequenzen nähern sich Autos oft der Kamera, fahren an ihr vorbei und verschwinden anschließend in der Entfernung. Außerdem gibt es viele Filmszenen, in denen Autos um Straßenecken biegen. In Abbildung 6.1 auf der nächsten Seite sind Beispiele für drei Konturen der Datenbank abgebildet. Die Größe der Objekte hängt von dem eingelesenen Bild ab.

Bei der Segmentierung der Objekte traten sehr selten Probleme auf, da ausschließlich leicht segmentierbare Objekte ausgewählt wurden. In jedem Bild ist nur ein einziges Objekt abgebil-

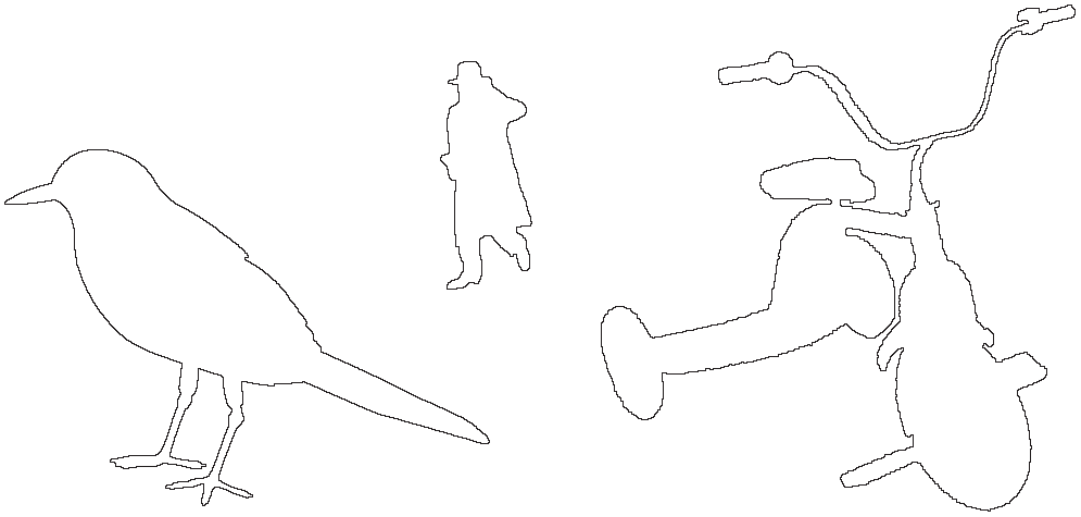

Abbildung 6.1: Ausgewählte Objekte der Datenbank. Bei den Konturen handelt es sich um eine Amsel, einen Mann mit Hut und Mantel und ein Dreirad.

det. Die maximale Größe der Bilder beträgt in x- und y-Richtung 800 Bildpunkte. Sie sind damit deutlich größer als die zu analysierenden Bildsequenzen, die sich maximal über 400 Bildpunkte horizontal oder vertikal ausdehnen. Die Segmentierung der 40 Objekte benötigt ungefähr 6 Minuten Rechenzeit (vgl. A.1 auf Seite IX). In einem zweiten Schritt werden die Maxima der Abbildung des krümmungsbasierten Skalenraums berechnet. Die Berechnung benötigt für die 40 Objekte weniger als eine Minute.

6.2 Untersuchung von Einzelbildern

Um einen ersten Eindruck von der Qualität der Objekterkennung zu gewinnen, werden zunächst einzelne Bilder mit den Objekten der Datenbank verglichen. Bei einem Vergleich zweier Objekte mit identischen Konturen tritt die erwartete Differenz von Null auf. Es existieren keine Unterschiede zwischen den Konturen und auch die Parametrisierung der Konturdaten unterscheidet sich nicht.

In einem zweiten Beispiel werden die Auswirkungen von Rotationen getestet. Der Fisch in Abbildung 6.2 auf der nächsten Seite wurde um 70 Grad gedreht. Rechts in dieser Abbildung wurde die Kontur des Fischs wieder zurückgedreht. Durch die Drehung mit Hilfe eines Grafikprogramms wurden die Konturen geringfügig verändert. Zusätzlich verschiebt sich der Startpunkt der Kontur, so dass unterschiedliche Pixel die Form des Objekts beschreiben. Ein Vergleich des gedrehten Fischs mit den Objekten der Datenbank ergibt eine Differenz von 71. Ein Vergleich zwischen unterschiedlichen Objekten zeigt, dass die Differenz sehr ähnlicher Konturen einen Wert von 100 nicht übersteigt. Konturen mit Differenzen über 250 besitzen deutliche

Abbildung 6.2: Objekterkennung bei Rotation. Links ist der ursprüngliche Fisch und in der Mitte der um 70 Grad gedrehte Fisch abgebildet. Rechts ist die mittlere Kontur wieder in den Originalzustand zurückgedreht worden. Bei jeder Drehung des Graustufenbildes ändert sich die Kontur geringfügig. Unterschiede sind beispielsweise an der Flosse links unten erkennbar.

Unterschiede. In Tabelle 6.1 auf der nächsten Seite werden beispielhaft einige Differenzen aufgelistet. Für einen besseren Vergleich der Unterschiede sind in Tabelle 6.2 auf der nächsten Seite zusätzlich die CSS-Maxima der Objekte abgebildet.

Auch wenn die Form des Objekts im stärkeren Maße modifiziert wird, kann weiterhin das ursprüngliche Objekt erkannt werden. In Abbildung 6.3 auf Seite 73 wird die Größe des Fisches verdoppelt, die vergrößerte Kontur gespiegelt und anschließend verzerrt. In den Tabellen 6.1 und 6.2 auf der nächsten Seite sind auch die Ergebnisse für den stark verformten Fisch abgebildet. Erfolgt die Verzerrung gleichmäßig über das ganze Bild, dann können die Objekte gut erkannt werden. Werden nur einzelne Bereiche des Bildes verzerrt, sinkt die Quote, mit der das modifizierte Objekt wiedererkannt wird.

Für einzelne Bilder liefert der Algorithmus gute Ergebnisse. Es ist jedoch anzumerken, dass bei der Auswahl der Objekte für die Datenbank nur leicht erkennbare also sehr charakteristische Konturen verwendet werden. Alle Objekte unterscheiden sich deutlich voneinander. Inwieweit ähnliche Ergebnisse in Bildsequenzen erzielt werden können, bei denen sich im Zeitablauf die Perspektive und die Form eines Objekts ändert, wird im folgenden Abschnitt beschrieben.

6.3 Untersuchung von Bildsequenzen

In diesem Abschnitt werden 6 kurze Bildsequenzen analysiert, die von verschiedenen Videos mit einer Auflösung von 400 x 400 Bildpunkten im AVI-Bildformat[1] aufgezeichnet wurden. In den Bildsequenzen sind starre Objekte und solche, die bei Bewegung ihre Form ändern, abgebildet. Filmszenen mit Autos, einem Lieferwagen oder einem Hubschrauber gehören zur ersten

[1] *Audio Video Interleaved* ist ein Multimediaformat von Microsoft Windows.

Rangfolge	unbekanntes Objekt	Objekt der Datenbank	Differenz
1	Fisch	Fisch	0
2		weiterer Fisch	146
3		Mensch	166
1	gedrehter Fisch	Fisch	71
2		Mensch	146
3		Eichhörnchen	192
1	zurück gedrehter Fisch	Fisch	49
2		LKW	134
3		Mensch	136
1	verzerrter Fisch	Fisch	69
2		Mensch3	148
3		Mensch5	153

Tabelle 6.1: Beispiel für die berechnete Differenz zweier Objekte.
Für jedes unbekannte Objekt werden die ähnlichsten drei Objekte der Datenbank ausgegeben. Gedrehte oder verzerrte Fische sind dem ursprünglichen Objekt immer noch sehr ähnlich.

Objektbezeichnung	CSS-Maxima	Differenz der CSS-Maxima zum Fisch
Fisch	436 322 258 121 98 84 57 7 5 4	
gedrehter Fisch	373 352 256 128 92 80 52 9 7 5	63 30 2 7 6 4 5 2 2 1
zurück gedrehter Fisch	385 328 256 120 91 78 60 7 7 5	51 6 2 11 7 6 3 0 2 1
verzerrter Fisch	444 297 251 109 82 81 74 10 9 5	8 25 7 12 16 3 17 3 4 1

Tabelle 6.2: Beispiel für die Unterschiede der CSS-Maxima bei modifizierten Objekten.
Bei dem gedrehten Fisch liegen die wesentlichen Unterschiede in den am stärksten konkav gekrümmten Bereichen. Beim verzerrten Fisch treten Unterschiede in allen Bereichen der Kontur auf.

Abbildung 6.3: Beispiel für Objekterkennung bei stärker modifizierter Konturen. Durch Größenänderung, Spiegelung und Verzerrung wird das Graustufenbild des linken Fischs in die Kontur des rechten Bildes umgewandelt.

Kategorie, die Darstellung eines Joggers und eines rennenden Hundes zur zweiten. Aus jeder Filmszene werden 8-25 Bilder ausgewählt, wobei der zeitliche Abstand der einzelnen Bilder bis zu eine Sekunde beträgt. Die Konturen aufeinanderfolgender Bilder unterscheiden sich hinsichtlich ihrer Größe oder Perspektive. Im Anhang sind die Konturen aller Objekte der Datenbank und der Videosequenzen abgebildet.

6.3.1 Bildsequenzen mit je einem PKW

In zwei verschiedenen Bildsequenzen werden die Möglichkeiten zur Erkennung von PKW's getestet. Die Bildsequenzen werden mit *PKW 1* und *PKW 2* bezeichnet und sind vollständig im Anhang in den Abbildungen C.1 bis C.4 dargestellt. Die erste Sequenz zeigt ein Auto, das auf der Straße wendet. Die 19 ausgewählten Bilder der ersten Bildfolge erlauben eine Ansicht des Autos von fast jeder Seite. Unter den fünf ähnlichsten Objekten tauchen die vier Autos der Datenbank auf. Das Ergebnis lässt für diese Bildsequenz die sichere Aussage zu, dass es sich bei dem zu analysierenden Objekt um ein Auto handelt (vgl. Tab. 6.3 auf der nächsten Seite).

Die Segmentierung bereitet wie in allen folgenden Bildsequenzen erhebliche Probleme. Das Auto unterscheidet sich nicht ausreichend vom Bildhintergrund. Erschwerend kommt hinzu, dass weitere Objekte wie Häuser oder Bäume in den Filmsequenzen abgebildet werden. Daher ist in dieser Filmszene eine Bildaufbereitung mit Hilfe eines Grafikprogramms vor der Segmentierung zwingend notwendig. Die aufbereiteten Bilddaten können durch die relativ geringe Auflösung in 28 Sekunden segmentiert werden. Die Laufzeiten aller Testsequenzen sind in Tabelle A.1 im Anhang aufgelistet.

In der zweiten Bildsequenz fährt ein PKW parallel zum Betrachter auf einer Straße. Auch in dieser Bildsequenz kann mit hoher Wahrscheinlichkeit die Aussage getroffen werden, dass es sich bei dem abgebildeten Objekt um ein Auto handelt (vgl. Tab. 6.4 auf Seite 75). Eine

Rangfolge	Objekt	Differenz
1	Auto 2	152
2	Auto 3	153
3	Auto 4	159
4	Haus	159
5	Auto 1	175
6	Fisch	178
7	Kopf	178
8	LKW	185

Tabelle 6.3: Analyse der der Bildsequenz *PKW 1*.
Die acht Objekte der Datenbank werden aufgelistet, die bei einem Vergleich mit der Bildsequenz die größte Ähnlichkeit besitzen.

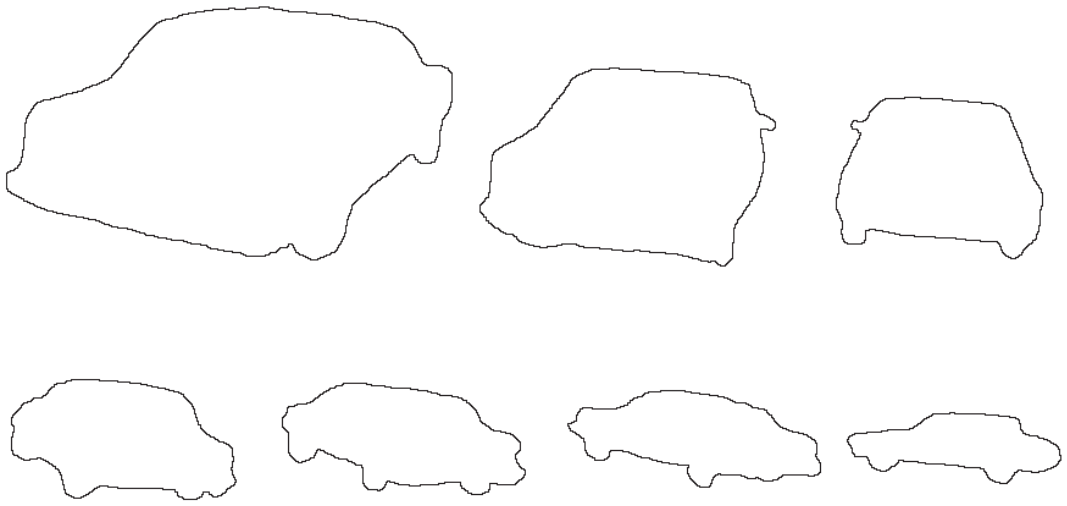

Abbildung 6.4: Ausgewählte Autos der Bildsequenz *PKW 1*. In einer Bildfolge, die aus 19 Bildern besteht, wendet ein PKW auf der Straße. Sieben ausgewählte Bilder werden beispielhaft abgebildet.

Rangfolge	Objekt	Differenz
1	Auto 3	142
2	Auto 2	145
3	Auto 4	146
4	Fisch	160
5	LKW	165
6	Haus	168

Tabelle 6.4: Analyse der Bildsequenz *PKW 2*.
Die sechs Objekte der Datenbank werden aufgezählt, die bei einem Vergleich mit der Bildsequenz die größte Ähnlichkeit besitzen.

Übersicht über die beiden Bildsequenzen ist in Abbildung 6.4 auf der vorherigen Seite und in Abbildung 6.5 auf der nächsten Seite dargestellt.

6.3.2 Bildsequenz mit einem Lieferwagen

Die dritte Bildsequenz besteht aus 19 Bildern, die mit den Objekten der Datenbank verglichen werden. Ein Lieferwagen fährt von links in das Bild. In den ersten ausgewählten Bildern ist der Lieferwagen nicht vollständig sichtbar. Er entfernt sich von der Kamera und wird in den letzten Bildern deutlich kleiner. Einen Überblick über die Filmszene liefert Abbildung 6.6 auf der nächsten Seite.

In der Datenbank sind vier Autos und ein LKW gespeichert. Die beiden Objekte, die der Bildsequenz am meisten ähneln, sind zwei dieser Autos. An den Position 5 und 7 folgen die anderen beiden Autos und an Position 10 schließlich der LKW (vgl. Tab. 6.5 auf Seite 77). Die vier Autos der Datenbank werden in Abbildung 6.7 auf Seite 77 dargestellt. Die unteren beiden Autos (Auto 3 und Auto 4) haben große Ähnlichkeit mit dem Lieferwagen, die oberen beiden Abbildungen deutlich geringere Ähnlichkeit, wobei die Frontalansicht (rechts oben) dem Lieferwagen am wenigsten ähnelt. Das Foto des Autos in der linken oberen Ecke ist in Höhe der Fahrertür aufgenommen worden. Eine kanonische Sicht würde das Auto aus größerer Höhe zeigen, mindestens aus der Höhe, in der ein Mensch ein Auto betrachtet. Die kanonische Sicht rechts unten stammt aus einem verrauschten Bild. Trotz der sehr unregelmäßigen Kontur aufgrund des Rauschens besteht bei diesem Objekt die größte Ähnlichkeit mit dem Lieferwagen der Bildsequenz.

Alle ähnlich erkannten Objekte haben eine sehr kompakte Form. Besonders die ersten Bilder der Sequenz besitzen durch die abgeschnittene Kante des Lieferwagens große Ähnlichkeit mit einem Haus. Daher treten die beiden Häuser der Datenbank an den Positionen 3 und 6 der ähnlichen Objekte auf. Der Lieferwagen besitzt sehr schwach ausgeprägte konkave Bereiche. Aus

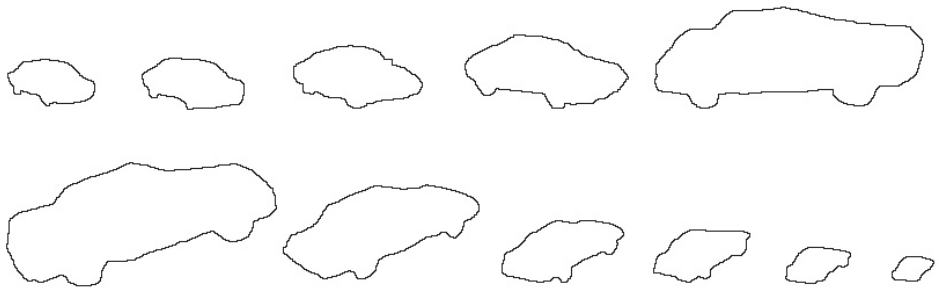

Abbildung 6.5: Ausgewählte Autos der Bildsequenz *PKW 2*. Die Bildfolge besteht aus 22 Bildern, von denen 11 abgebildet sind. Ein PKW fährt auf einer Straße, die das Bild von links nach rechts durchquert.

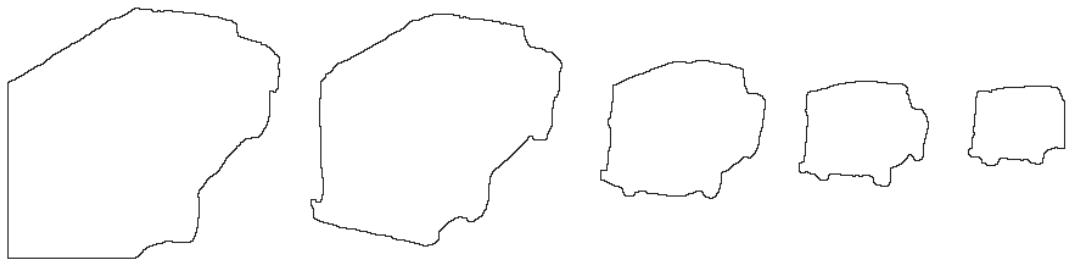

Abbildung 6.6: Ausgewählte Konturen der Bildsequenz *Lieferwagen*. Fünf ausgewählte Bilder dieser Bildsequenz werden beispielhaft abgebildet. Am Anfang dieser Bildsequenz fährt der Lieferwagen in das Bild, so dass dessen linker unterer Rand abgeschnitten wird.

Rangfolge	Objekt	Differenz
1	Auto 4	104
2	Auto 3	114
3	Haus 2	123
4	Kopf	125
5	Auto 1	143
6	Haus 1	144
7	Auto 2	157
8	Apfel	165
9	Ball	174
10	LKW	181

Tabelle 6.5: Analyse der Bildsequenz *Lieferwagen*.
Die zehn Objekte der Datenbank werden aufgelistet, die bei einem Vergleich mit der Bildsequenz die größte Ähnlichkeit besitzen. Der Lieferwagen der Bildsequenz besitzt nur sehr schwach ausgeprägte konkave Bereiche. Daher haben die zehn ähnlichsten Objekte eine überwiegend runde Kontur.

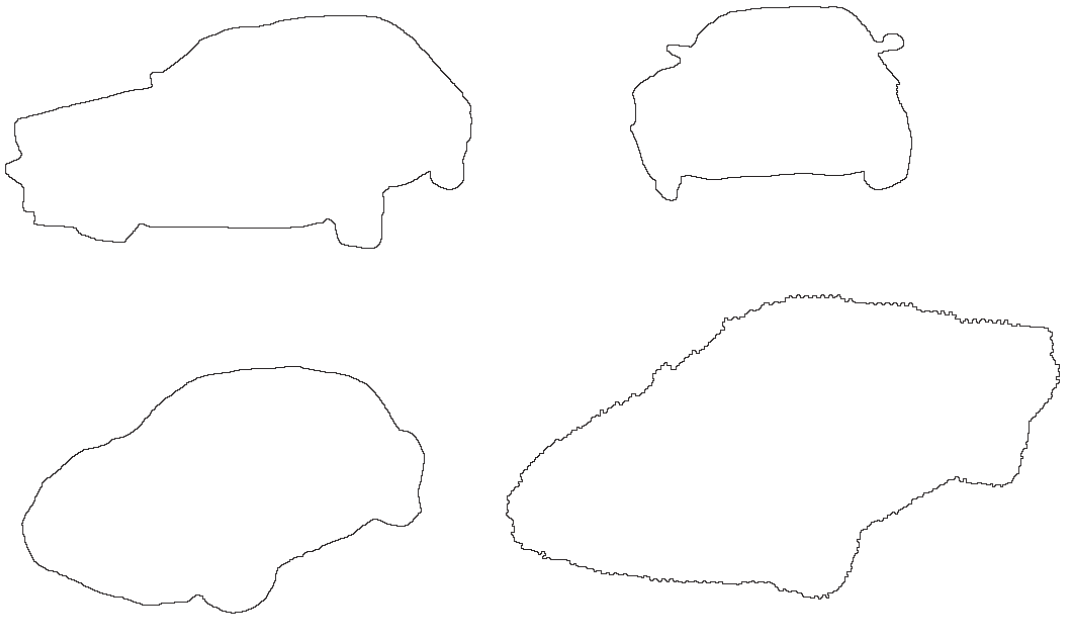

Abbildung 6.7: Beispiel für die Darstellung von Autos in der Datenbank. Von den vier abgebildeten Konturen der Autos werden die unteren beiden aus einer leicht erhöhten Perspektive gezeigt.

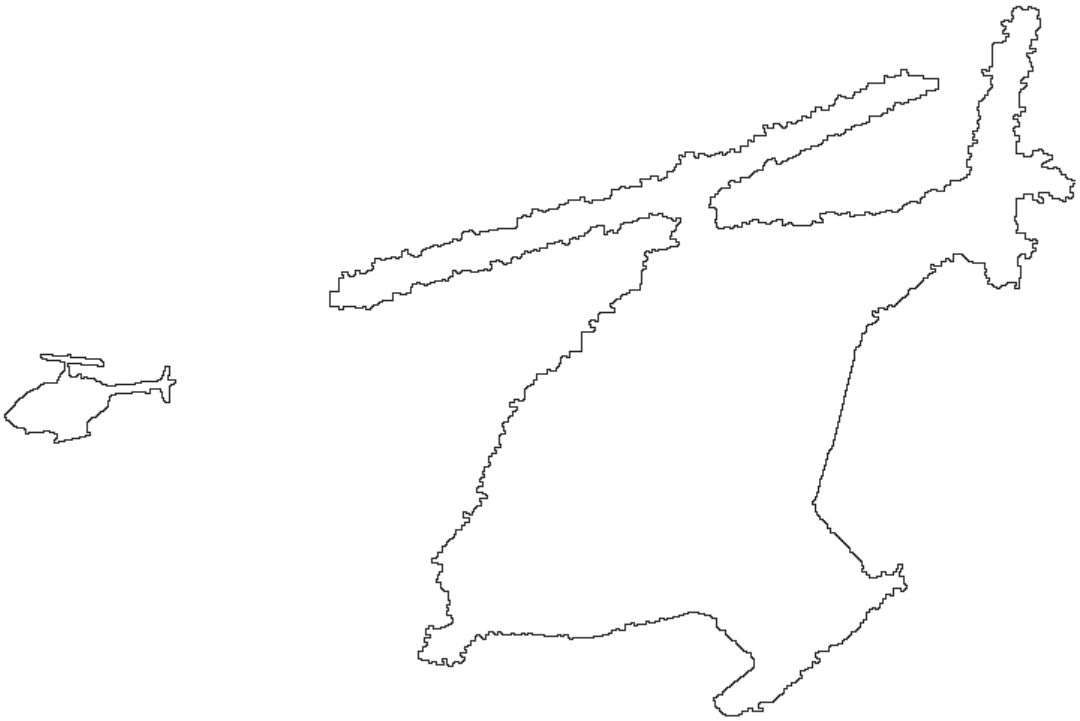

Abbildung 6.8: Beispiel für die Darstellung von Hubschraubern in der Datenbank. Die beiden Hubschrauber der Datenbank unterscheiden sich deutlich hinsichtlich ihrer Größe. Der große Hubschrauber stammt aus einem stark verrauschten Bild.

diesem Grund treten Konturen wie Kopf, Apfel und Ball unter den zehn ähnlichsten Objekten auf.

Die Erkennung des Lieferwagens verläuft insoweit erfolgreich, dass mit größter Wahrscheinlichkeit ein Auto erkannt wird. Die Ähnlichkeit zu einem PKW ist größer als zu einem LKW, da der LKW der Datenbank eine sehr eckige Kontur besitzt. Kanonische Sichten werden in diesem Beispiel vom Algorithmus leichter erkannt als Bilder aus anderen Perspektiven. Die Aussage, ob in den gewählten Bildsequenzen ein Objekt abgebildet wird, das in seiner Kontur große Ähnlichkeit mit der Kontur eines Autos aufweist, konnte in diesem Beispiel bestätigt werden.

6.3.3 Bildsequenz mit einem Hubschrauber

In einer vierten Testsequenz ist ein Hubschrauber abgebildet, der sich aus großer Entfernung der Kamera nähert. Bei den letzten der 14 Bilder ist der Hubschrauber nicht mehr vollständig sichtbar, und die hinteren Rotorblätter sind abgeschnitten. In Abbildung 6.8 sind die beiden Hubschrauber der Datenbank abgebildet. Der große Hubschrauber stammt aus einem stark verrauschten Bild, der kleine wurde einer unverrauschten Bildsequenz entnommen.

Die Segmentierung der Bildsequenz bereitet auch in diesem Experiment die größten Proble-

Rangfolge	Objekt	Differenz
1	Hubschrauber (groß)	184
2	Känguru	197
3	Elefant	199
4	Dreirad	210
5	Hirsch	213
6	Hubschrauber (klein)	217
7	Vogel	231

Tabelle 6.6: Analyse der Bildsequenz *Hubschrauber*.
Die sieben Objekte der Datenbank werden aufgelistet, die bei einem Vergleich mit der Bildsequenz die größte Ähnlichkeit besitzen.

me. Die Kontur eines Hubschraubers ändert sich deutlich, wenn beispielsweise der Rotor nicht vollständig erkannt wird.

Die Ergebnisse sind weniger eindeutig als in den vorherigen Bildsequenzen. Als ähnlichstes Objekt wird der große Hubschrauber gefunden. Die Unterschiede zwischen allen ähnlichen Objekten sind jedoch sehr gering. Auffällig ist, dass der große Hubschrauber deutlich besser als der kleine Hubschrauber erkannt wird (vgl. Tab. 6.6). Eine Erklärung könnte in der geringen Größe des Hubschraubers liegen. Wird die Abbildung eines Objekts zu klein gewählt, dann verschwinden entscheidende Details der Kontur. Wird eine Kontur durch deutlich weniger als 250 Randpixel bestimmt, so treten diese Ungenauigkeiten auf. Rauschen hat auch in diesem Beispiel sehr geringe Auswirkung auf die Qualität der Objekterkennung. Ebenfalls auffällig ist, wie gering die Differenzen zwischen den als ähnlich erkannten Objekten sind. Der Unterschied zwischen dem großen Hubschrauber und dem Känguru ist nicht so groß, wie man vielleicht vermutet hätte.

6.3.4 Bildsequenz mit einem Jogger

In den letzten beiden Filmszenen werden bewegliche Objekte abgebildet. Zunächst wird ein Jogger betrachtet, der sich von rechts der Kamera nähert und das Bild am linken Rand verlässt. Bei den 13 ausgewählten Bildern ist er zunächst in größerer Entfernung zu sehen, so dass die Kontur der ersten Bilder sehr ungenau ist. Mit zunehmender Größe des Joggers im Bild wird auch dessen Kontur feiner.

Für diese Versuchsreihe wurden sechs Abbildungen eines Menschen in die Datenbank integriert. Unter den sechs ähnlichsten Objekten werden fünf Menschen gefunden. Lediglich ein Fisch (vgl. Abb. 6.9 auf der nächsten Seite) wird fälschlicherweise als viert ähnlichstes Objekt erkannt. Die acht Objekte der Datenbank, die dieser Bildsequenz am meisten ähneln, werden in Tabelle 6.7 auf Seite 81 mit ihren Differenzen aufgelistet.

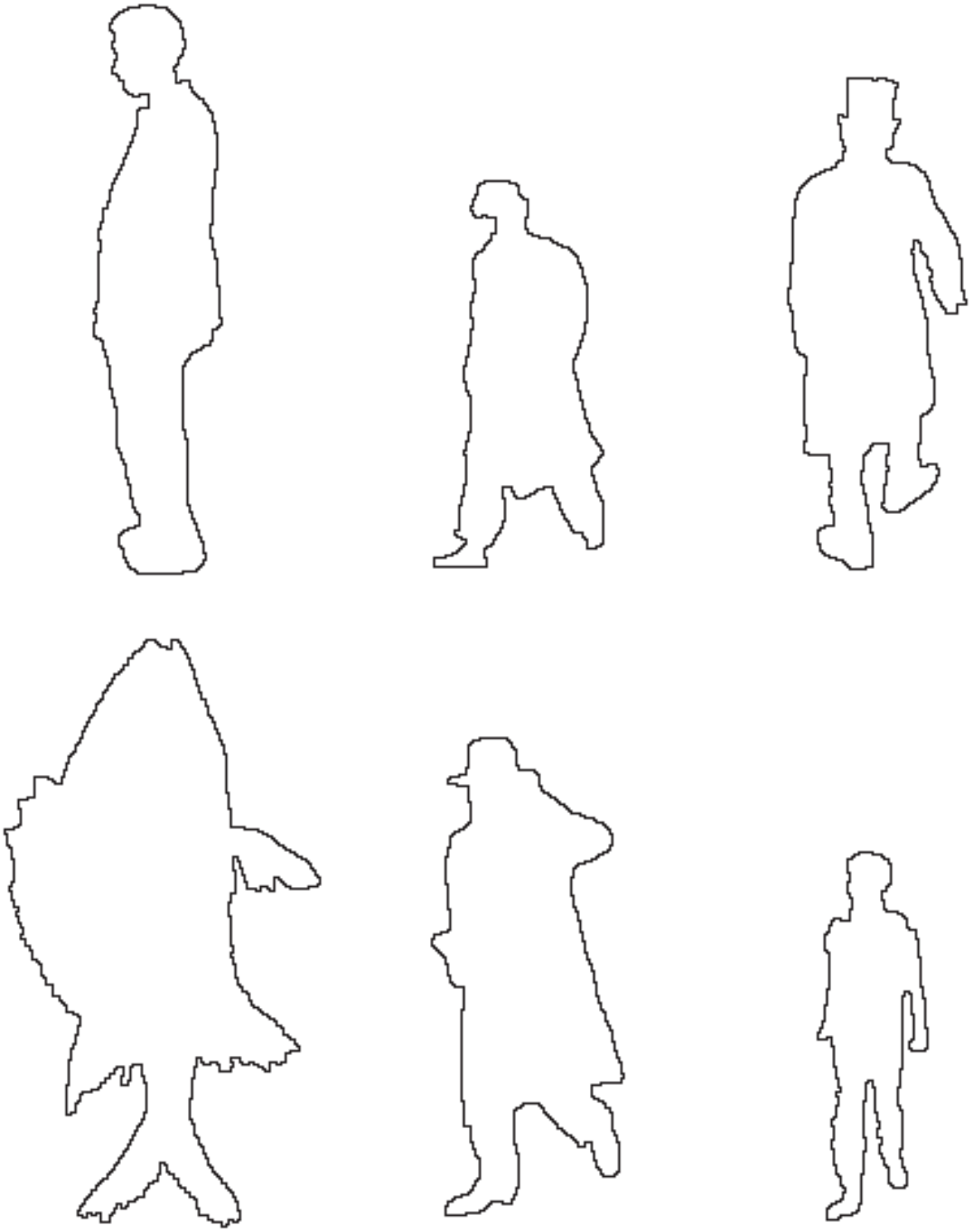

Abbildung 6.9: Ergebnisse der Bildsequenz *Jogger*. Es werden die sechs Konturen abgebildet, die die größte Ähnlichkeit mit dem Jogger aus der Bildsequenz aufweisen. Links oben ist das ähnlichste Objekt zu sehen, das Objekt rechts unten zeigt die größte Differenz unter diesen Objekten.

Rangfolge	Objekt	Differenz
1	Mensch 3	206
2	Mensch 4	206
3	Mensch 2	209
4	Fisch	212
5	Mensch 5	214
6	Mensch 1	215
7	Eichhörnchen	218
8	Kamel	224

Tabelle 6.7: Analyse der Bildsequenz *Jogger*.
Die acht Objekte der Datenbank werden aufgelistet, die bei einem Vergleich mit der Bildsequenz die größte Ähnlichkeit besitzen. Da die geringste Differenz mehr als 200 beträgt, unterscheiden sich die Konturen der Bildsequenz deutlich von den Konturen der Datenbank.

Die Differenzen der ähnlichsten Objekte der Datenbank zur Bildsequenz *Jogger* unterscheiden sich nur gering. So liegen die Differenzen der acht ähnlichsten Objekte weniger als 10 Prozent auseinander. Trotz dieser geringen Unterschiede der Differenzen kann mit sehr hoher Wahrscheinlichkeit die richtige Aussage über das abgebildete Objekt getroffen werden, da unter den 6 ähnlichsten Objekten fünf Menschen gefunden werden.

Wird statt der Bildsequenz nur ein einzelnes Bild betrachtet, so ist eine korrekte Erkennung nicht mehr möglich. Summiert über die 13 Einzelbilder werden nur vier Menschen an erster Position erkannt, was bei sechs möglichen Kandidaten der Datenbank äußerst gering ist. In Tabelle 6.8 auf der nächsten Seite ist die Verteilung der Menschen unter den ähnlichsten Objekten aufgelistet. Die meisten Menschen werden im vorderen Drittel gefunden. Die Ergebnisse der gesamten Bildsequenz unterscheiden sich somit deutlich von einer Betrachtung von Einzelbildern, da sich die Kontur eines Menschen durch Bewegung sehr stark verändern kann.

Auch wenn der Jogger aus der Bildsequenz für sich alleine betrachtet selten die größte Ähnlichkeit mit der Kontur eines Menschen der Datenbank aufweist, so ermöglicht ein Vergleich über die ganze Bildsequenz eine sichere Erkennung.

6.3.5 Bildsequenz mit einem Hund

Für diese Untersuchung wird erneut die vorherige Bildsequenz mit dem Jogger verwendet. Diesmal wird der Hund betrachtet, der neben dem Jogger rennt. Ein Mensch kann ihn ausschließlich anhand seiner Kontur nicht erkennen, da der Hund sehr klein ist und über keine spezifische Kontur verfügt. In einer Sequenz von Graustufenbildern kann ein Mensch diesen Hund jedoch erkennen (vgl. Abb. 6.10 auf Seite 83).

Als Ergebnis lässt sich festhalten, dass der Hund durch den implementierten Objekterken-

Rangfolge	Anzahl aller erkannten Menschen
1	4
2	9
5	23
10	38
15	49

Tabelle 6.8: Verteilung der korrekt erkannten Menschen in der Bildsequenz *Jogger*.
In der Datenbank existieren sechs Menschen. Die Bildsequenz *Jogger* besteht aus 13 zu analysierenden Bildern. Die Tabelle beschreibt, wie häufig ein Mensch bis zur angegebenen Rangfolge in allen 13 Bildern auftritt. Z. B. erkennt der Algorithmus unter den beiden ähnlichsten Objekten, d. h. Objekten mit der Rangfolge 2, neunmal einen Menschen.

Rangfolge	Objekt	Differenz
1	Eichhörnchen	178
2	Mensch 5	184
3	Mensch 4	185
4	Mensch 3	186
5	Fisch	189
6	Kamel	191
7	Kuh	197

Tabelle 6.9: Analyse der Bildsequenz *Hund*.
Unter den sieben ähnlichsten Objekten tritt kein Hund auf. Eine Objekterkennung ist bei dieser Bildsequenz nicht möglich.

nungsalgorithmus nicht erkannt wird (vgl. Tab. 6.9). Bei einem Vergleich weisen die CSS-Maxima der drei Hunde der Datenbank sehr große Differenzen zum Hund der Bildsequenz auf. Der Hund der Bildsequenz ist sehr klein und dessen Kontur verfügt über wenig Details. Die Hunde der Datenbank werden in Profilansichten gezeigt, so dass beispielsweise die Beine deutlich sichtbar sind. Die Hunde der Datenbank ähneln daher dem abgebildeten Hund der Bildsequenz nicht.

6.4 Beurteilung der Ergebnisse

Die Erkennung von Objekten in Bildsequenzen verläuft überaus erfolgreich. Deutliche Probleme bereitet das Verfahren zur Segmentierung, das jedoch nicht Schwerpunkt der Arbeit ist. Das Verfahren ist für leicht segmentierbare Objekte anwendbar, die in Bildsequenzen aus Videos

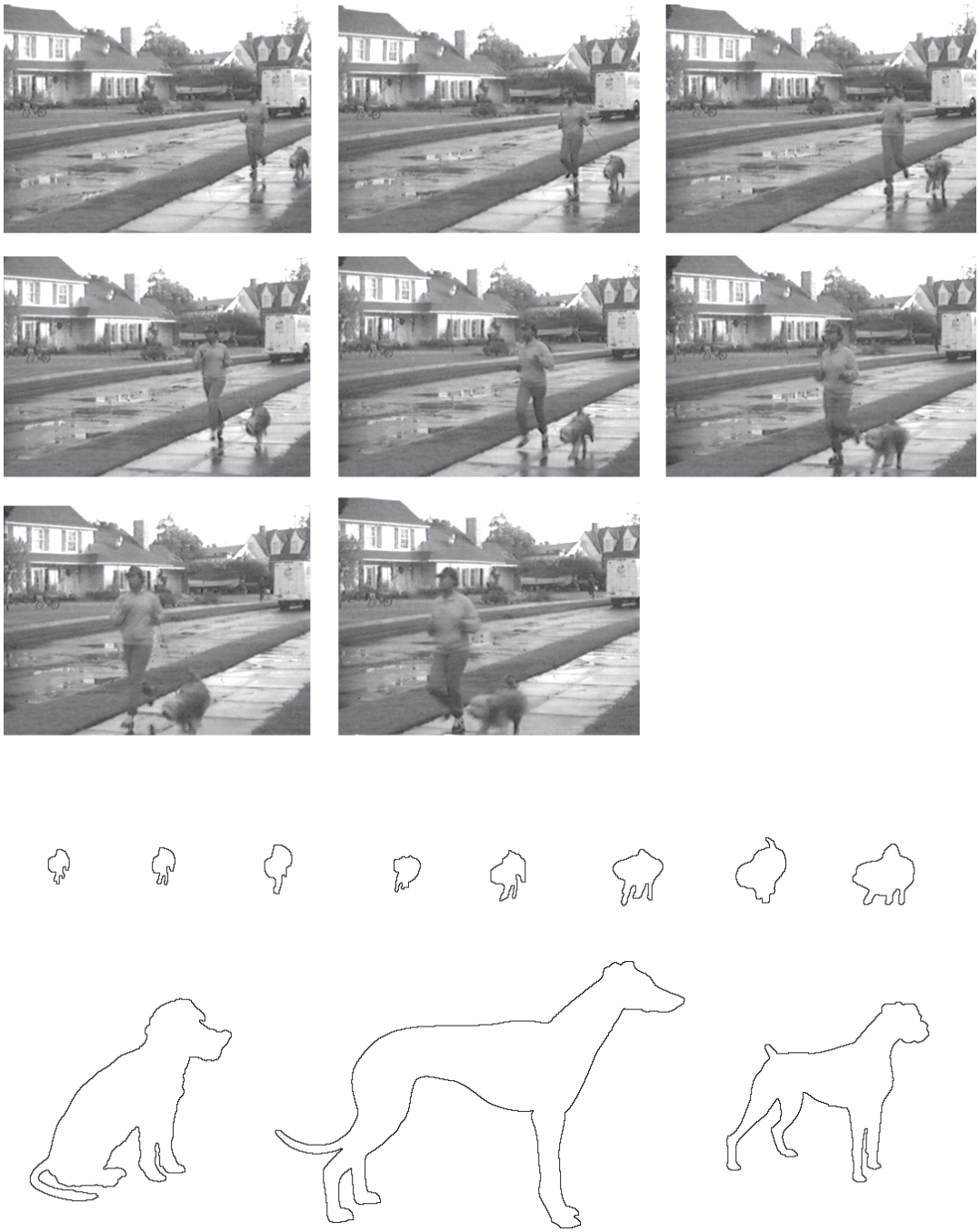

Abbildung 6.10: Ergebnisse der Bildsequenz *Hund*. In der Bildsequenz ist neben dem Jogger ein Hund zu sehen, dessen Kontur unter der Bildsequenz abgebildet ist. Die unterste Zeile zeigt die drei gespeicherten Hundekonturen der Datenbank. Sie können erwartungsgemäß durch den implementierten Objekterkennungsalgorithmus nicht erkannt werden.

üblicherweise nicht auftreten. Daher muss vor jedem Segmentierungsschritt das einzelne Bild mit einem Bildverarbeitungsprogramm vorbereitet werden. Ohne eine bessere Segmentierung kann das Verfahren für eine automatische Inhaltsanalyse von Bildsequenzen nicht eingesetzt werden.

Der zweite wesentliche Nachteil beruht auf den langen Laufzeiten der Algorithmen, insbesondere der Segmentierung. Um die 40 Objekte der Datenbank zu segmentieren, wird eine Rechenzeit von mehreren Minuten benötigt. Eine erste Verbesserung des Algorithmus sollte zunächst das Problem der Segmentierung lösen.

Die Berechnung der Abbildungen im krümmungsbasierten Skalenraum und die Ermittlung der CSS-Maxima benötigt bei längeren Bildsequenzen ebenfalls zuviel Zeit. Es kann ungefähr ein Bild pro Sekunde umgerechnet werden. Durch eine Vergrößerung der Gaußmaske lässt sich die Rechenzeit auf Kosten der Genauigkeit reduzieren.

Beim dritten Schritt, dem Vergleich eines unbekannten Objekts mit den 40 Objekten der Datenbank, können 20 unbekannte CSS-Datensätze pro Sekunde analysiert werden. Im Vergleich zu den anderen beiden Algorithmen ist diese Zeit akzeptabel. Eine deutliche Beschleunigung des Verfahrens ist möglich, wenn eine Vorauswahl für ähnliche Objekte getroffen wird. Beispielsweise ist es sinnvoll zwei Objekte nur dann miteinander zu vergleichen, wenn die Differenz der größten CSS-Maxima beider Objekte einen genauer zu spezifizierenden Schwellenwert nicht überschreitet.

Abgesehen von der Anzahl und den Namen der zu analysierenden Bilder ändern sich keine Parametern des Algorithmus. Die Objekterkennung funktioniert trotz unterschiedlicher Perspektiven stabil. Deutliche Fehler treten erst in der letzten Bildsequenz auf, bei der besonders kleine und unscharfe Objekte analysiert werden und keine in ihrer Kontur ähnlichen Objekte in der Datenbank existieren. Die Bildsequenz *Hund* macht deutlich, dass die ausgewählten Objekte der Datenbank die Qualität der Objekterkennung ganz wesentlich beeinflussen.

Kapitel 7

Zusammenfassung und Ausblick

Das Verfahren zur Objekterkennung mit Hilfe von Abbildungen im krümmungsbasierten Skalenraum ist schnell, sehr robust und liefert besonders gute Ergebnisse bei der Analyse von Bildsequenzen. Geringe Unterschiede zwischen Konturen, die durch Rauschen oder Änderungen der Perspektive auftreten können, reduzieren die Qualität der Objekterkennung nur unwesentlich. So ändert sich in allen untersuchten Bildsequenzen die Größe und Perspektive des abgebildeten Objekts. Die unterschiedlichen Perspektiven verbessern die Objekterkennung im Vergleich zur Analyse einzelner Bilder.

Die Segmentierung, die nicht Schwerpunkt dieser Arbeit ist, bereitet die meisten Probleme. Die Annahme, dass in den Bildern und Bildsequenzen nur leicht segmentierbare Objekte auftreten, ist nicht realistisch. Üblicherweise unterscheidet sich die Helligkeit der abgebildeten Objekte nur geringfügig vom Bildhintergrund. Eine automatische Segmentierung ist daher ohne Verwendung zusätzlicher Filter nicht möglich. Die lange Laufzeit des Segmentierungsalgorithmus macht eine Analyse kompletter Filme z. Z. schwer vorstellbar, Bildsequenzen können jedoch in akzeptabler Zeit analysiert werden.

Die ausgewählten Objekte der Datenbank beeinflussen ganz entscheidend die Qualität der Objekterkennung. Ihre Konturen unterscheiden sich deutlich voneinander, so dass sowohl ein Mensch als auch der implementierte Algorithmus Unterschiede zwischen den vorhandenen Objekten leicht erkennen kann. Werden deutlich mehr Objekte in die Datenbank integriert, so treten ähnlichere Konturen auf und die Objekterkennung wird weniger eindeutig. Inwieweit neben den kanonischen Sichten andere Blickwinkel für die rechnergesteuerte Bildanalyse relevant sein könnten, wurde im Rahmen dieser Arbeit nicht behandelt.

Die Qualität der Segmentierung ließe sich verbessern, indem die beiden Schritte Segmentierung und Objekterkennung miteinander kombiniert werden. Ein Objekt könnte in einem ersten Schritt grob segmentiert werden. Ähnliche Objekte der Datenbank lassen sich durch die robuste Objekterkennung schon anhand dieser groben Struktur ermitteln. Die möglichen Objekte der Datenbank werden anschließend verwendet, um die Kontur des Objekts genauer zu spezifizieren. Diese schrittweise Verfeinerung könnte schließlich zu einem deutlich besser segmentierten

Objekt führen.

Die Erkennung von Objekten erfolgt auf der Ebene von Objektklassen. Für einen Menschen enthält die Kontur nicht immer ausreichend Informationen, um unterschiedliche Objekte einer Objektklasse korrekt zu erfassen. Das Objekterkennungsverfahren könnte erweitert werden, indem zusätzlich Oberflächenstrukturen und Farben berücksichtigt werden. Statt Objektklassen können Objekte detaillierter erkannt werden.

Abschließend lässt sich festhalten, dass eine Objekterkennung in Bildsequenzen für das eingegrenzte Problemfeld dieser Arbeit möglich ist. Eine allgemeine und flexible Bilderkennung, die an die Fähigkeiten eines Menschen heranreicht und auch mehrere sich gegenseitig verdeckende Objekte erkennen kann, ist in der heutigen Zeit jedoch noch nicht realisierbar.

Anhang A

Laufzeitverhalten

Bezeichnung	**Anzahl der analysierten Bilder**	**Segmentierung der Objekte [in Sekunden]**	**Ermittlung der CSS-Maxima [in Sekunden]**	**Vergleich mit der Datenbank [in Sekunden]**
Datenbank	40	330	37	
PKW 1	19	28	13	1
PKW 2	22	16	17	1
Lieferwagen	19	27	18	1
Hubschrauber	14	15	18	0.5
Jogger	13	39	12	0.5
Hund	8	3	8	0.5

Tabelle A.1: Benötigte Rechenzeiten beim Erstellen der Datenbank und der Analyse der Bildsequenzen auf der SUN.

Bezeichnung	**Anzahl der analysierten Bilder**	**Segmentierung der Objekte [in Sekunden]**	**Ermittlung der CSS-Maxima [in Sekunden]**	**Vergleich mit der Datenbank [in Sekunden]**
Datenbank	40	220	25	
PKW 1	19	20	9	1
PKW 2	22	11	12	1
Lieferwagen	19	18	12	1
Hubschrauber	14	11	13	0.5
Jogger	13	26	9	0.5
Hund	8	2	6	0.5

Tabelle A.2: Benötigte Rechenzeiten beim Erstellen der Datenbank und der Analyse der Bildsequenzen auf dem Intel-PC.

Anhang B

Objekte der Datenbank

Abbildung B.1: Objekte der Datenbank. Von links-oben nach rechts-unten sind folgende Objekte abgebildet: Apfel, vier Autos, Bär, Ball, Delfin und Dreirad.

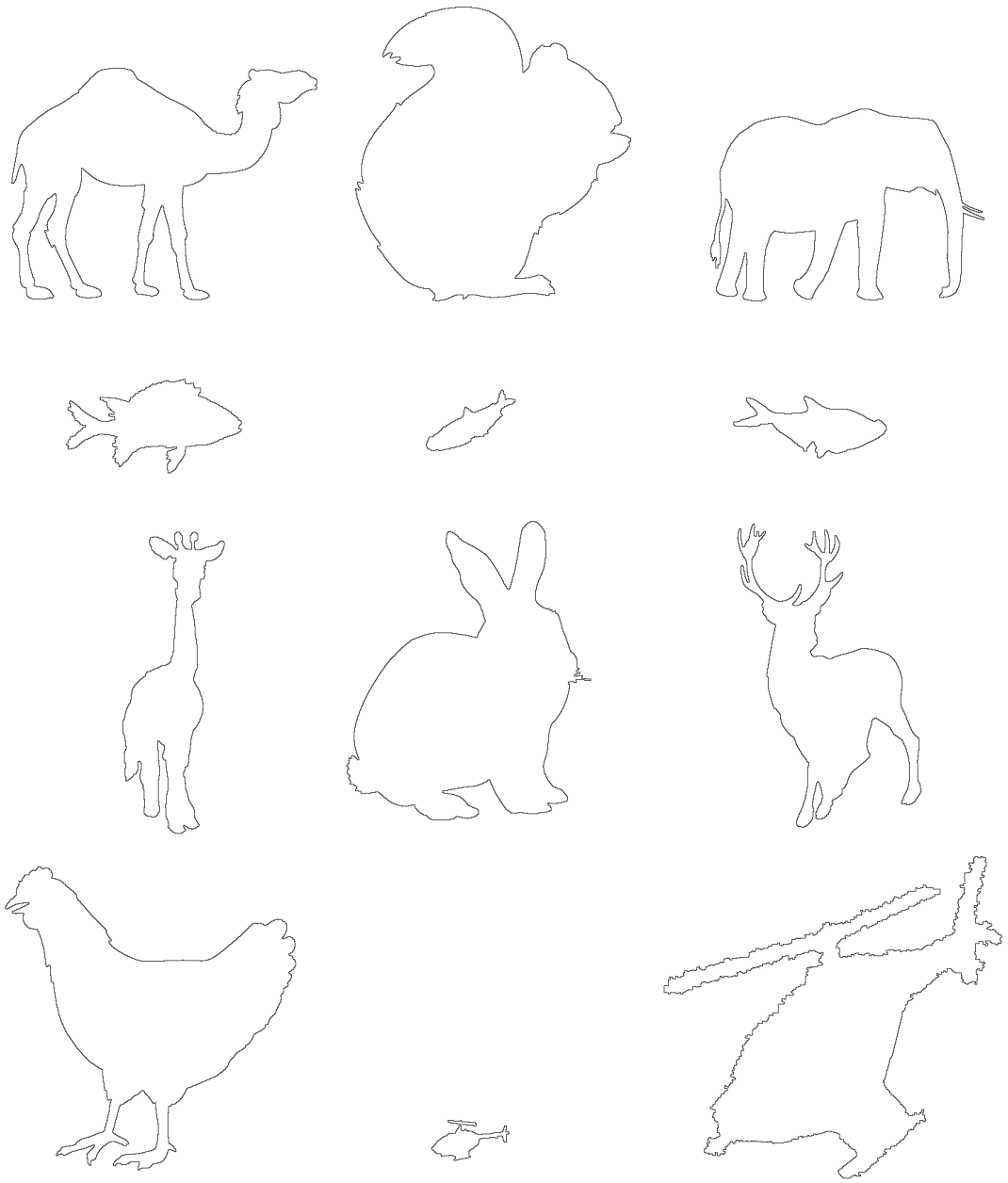

Abbildung B.2: Objekte der Datenbank. Von links-oben nach rechts-unten sind folgende Objekte abgebildet: Dromedar, Eichhörnchen, Elefant, drei Fische, Giraffe, Hase, Hirsch, Henne und zwei Hubschrauber.

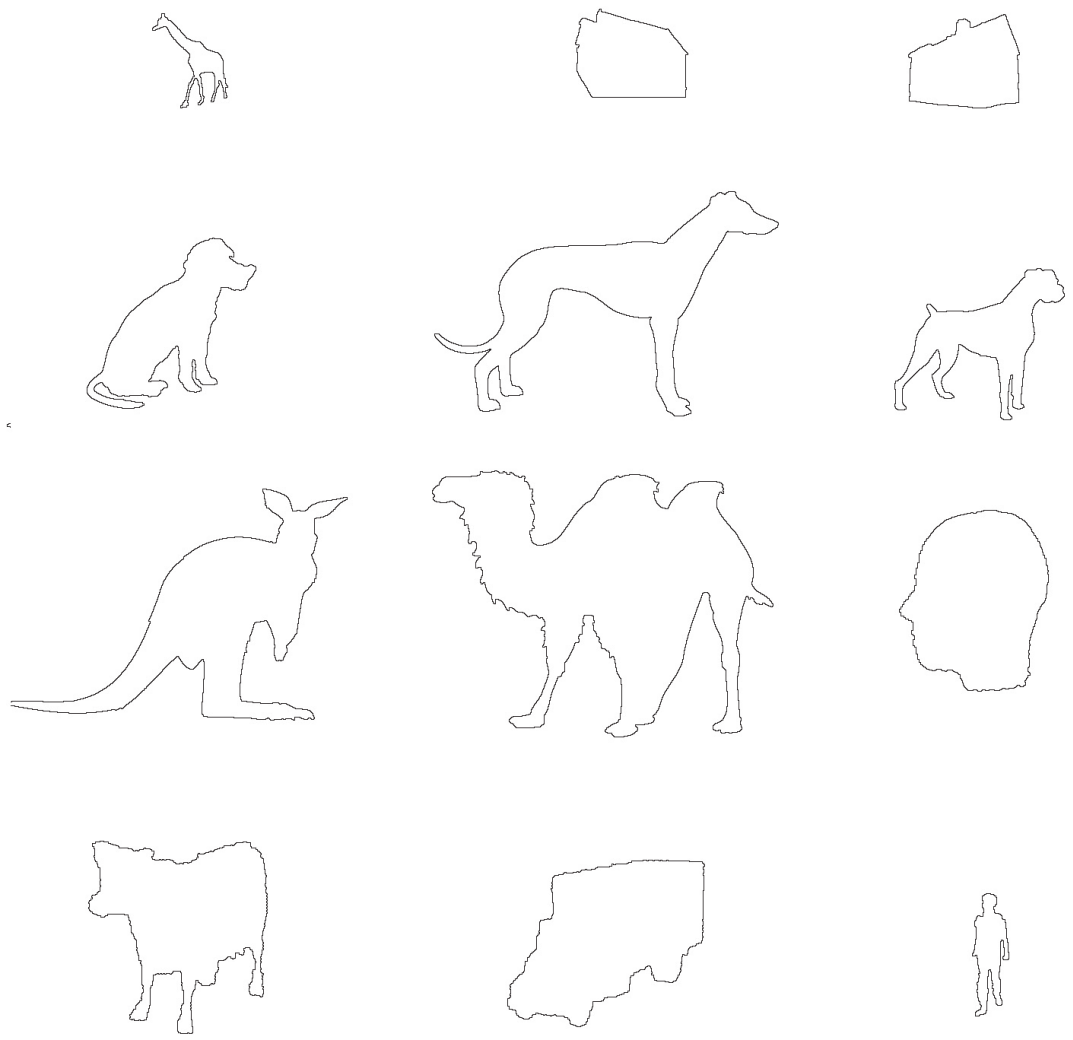

Abbildung B.3: Objekte der Datenbank. Von links-oben nach rechts-unten sind folgende Objekte abgebildet: Giraffe, zwei Häuser, drei Hunde, Känguru, Kamel, Kopf, Kuh, LKW und ein Mensch.

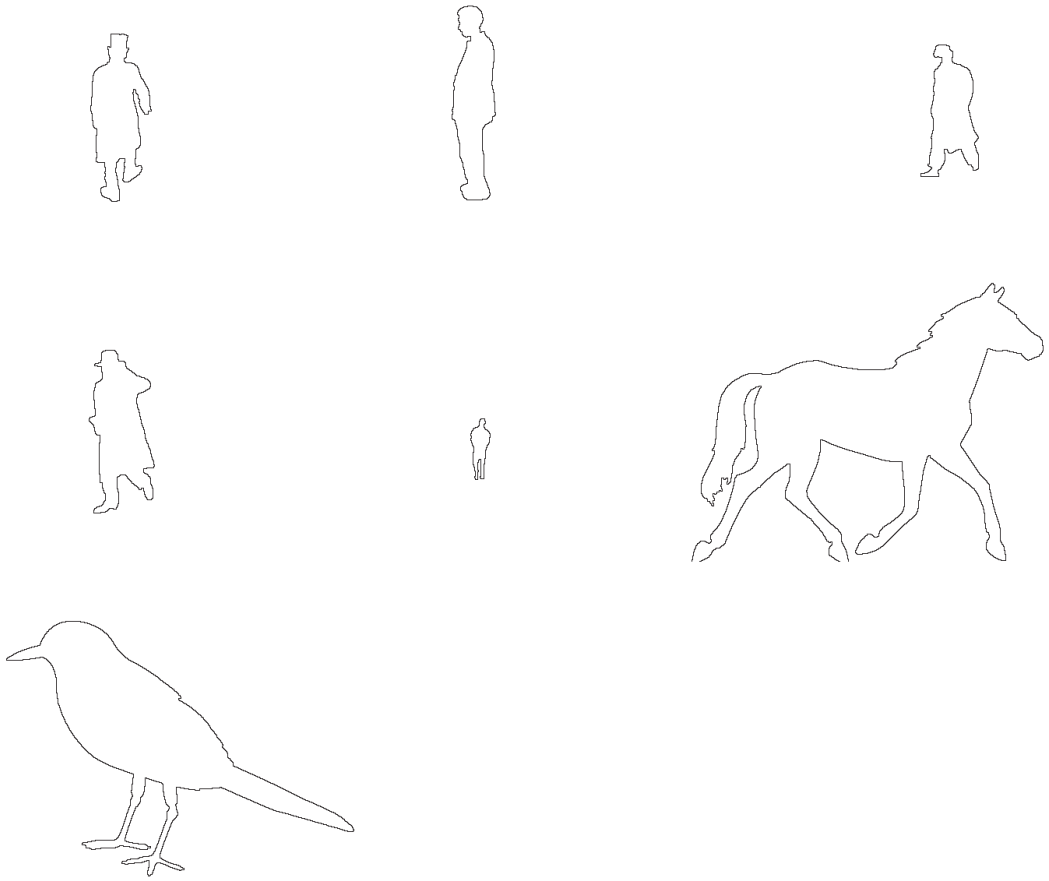

Abbildung B.4: Objekte der Datenbank. Von links-oben nach rechts-unten sind folgende Objekte abgebildet: fünf Menschen, Pferd und Vogel.

Anhang C

Konturen der Bildsequenzen

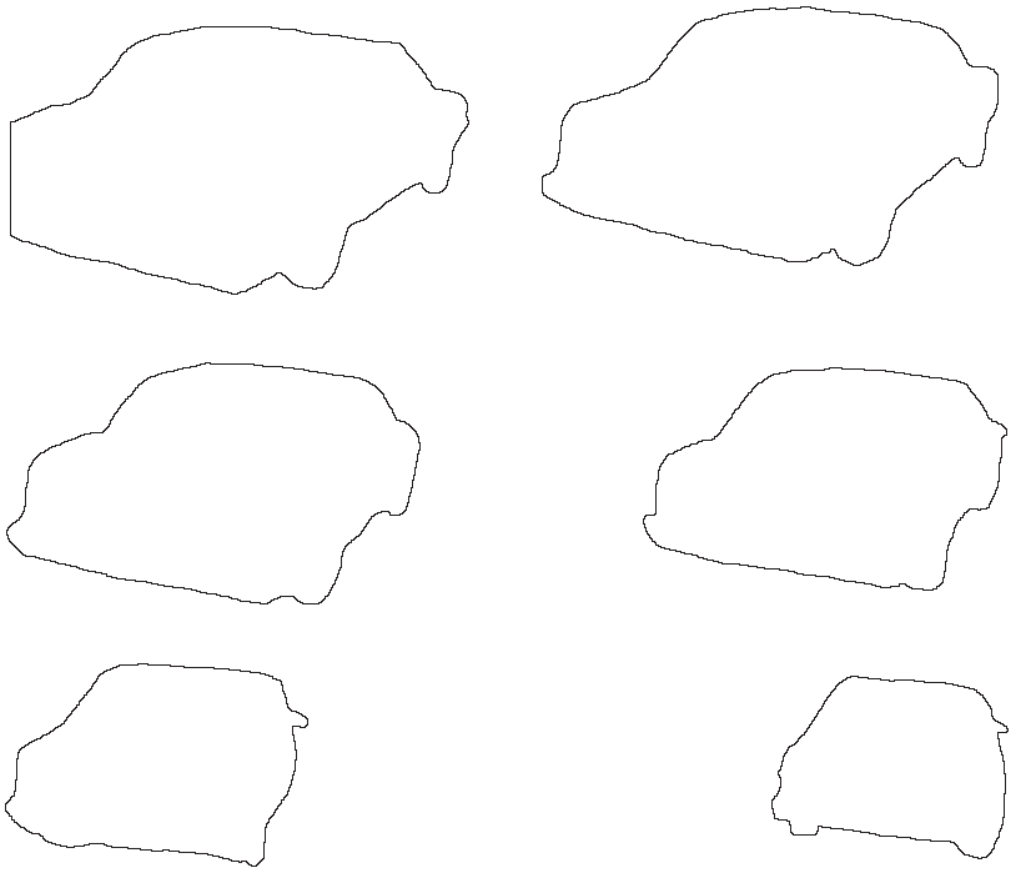

Abbildung C.1: Konturen der Bildsequenz *PKW 1*.

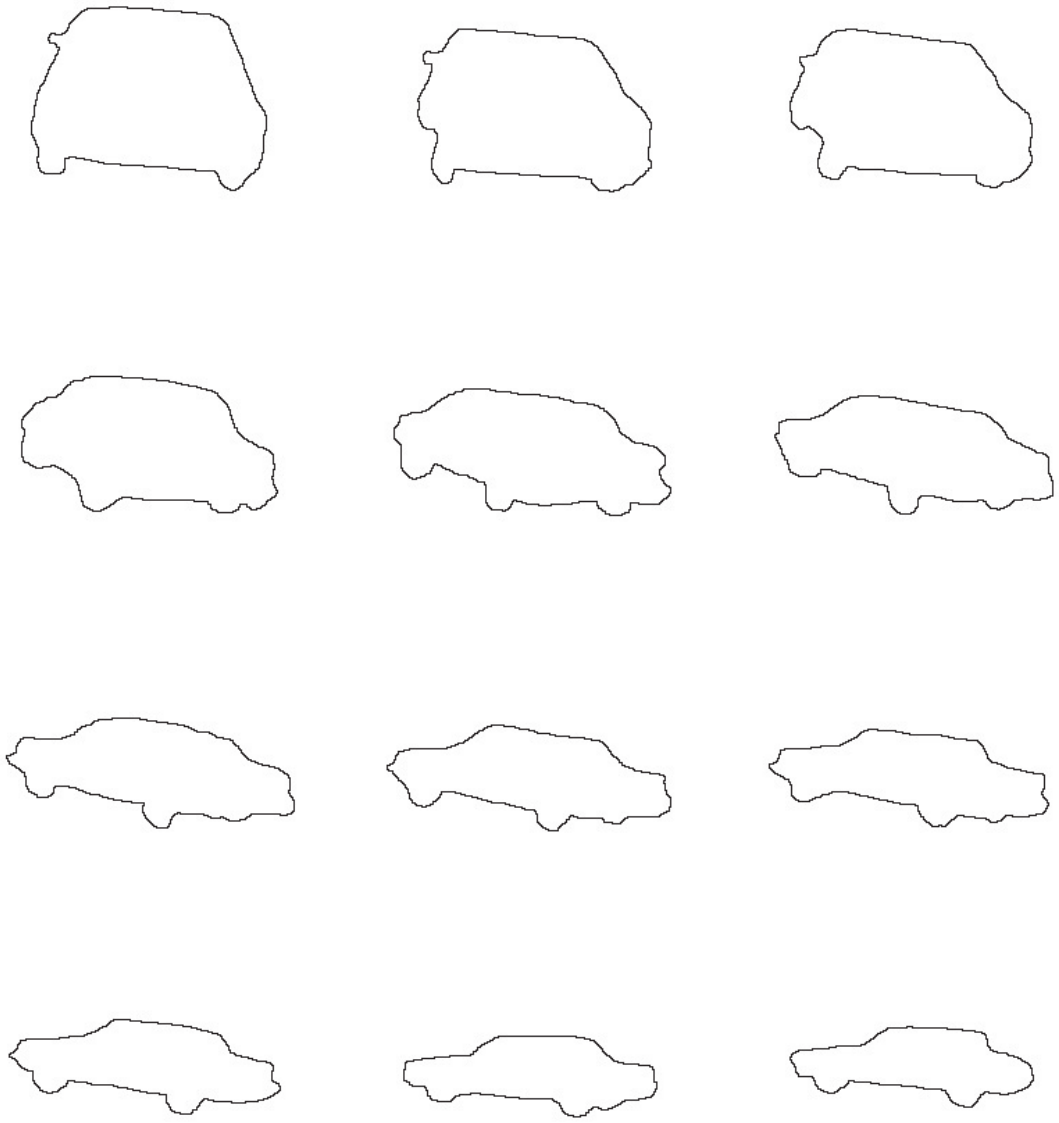

Abbildung C.2: Konturen der Bildsequenz *PKW 1* (Fortsetzung).

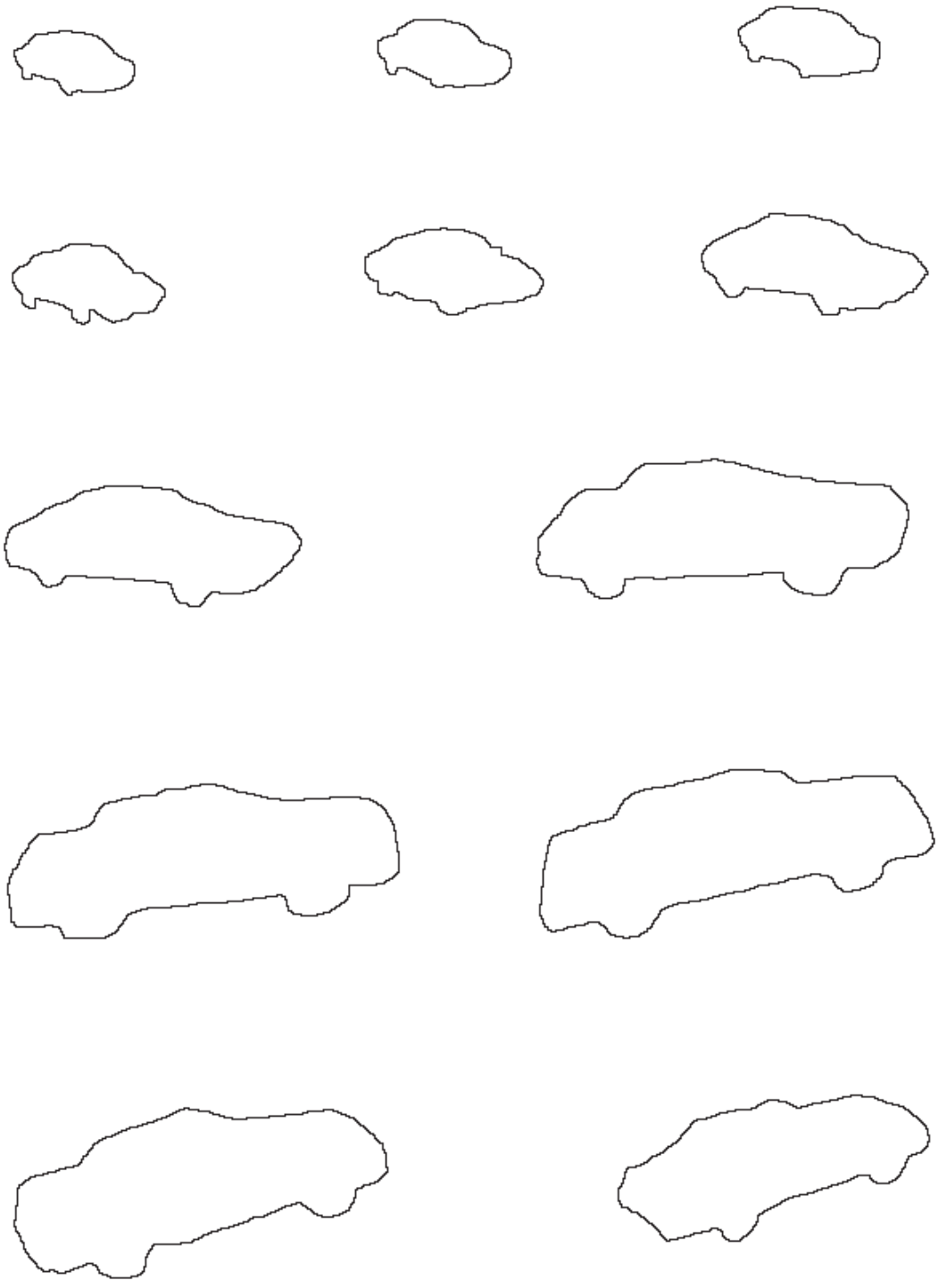

Abbildung C.3: Konturen der Bildsequenz *PKW 2*.

Abbildung C.4: Konturen der Bildsequenz *PKW 2* (Fortsetzung).

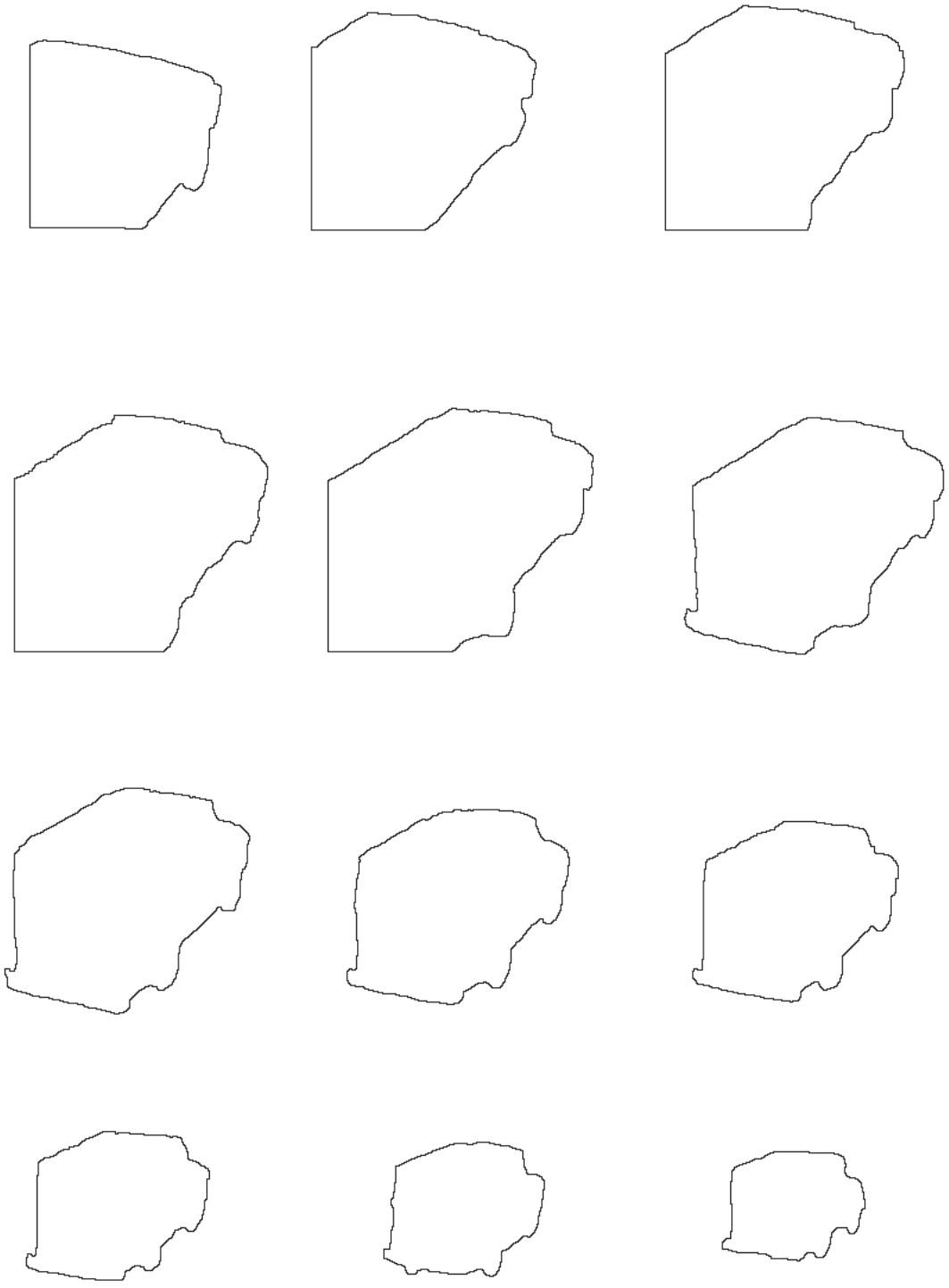

Abbildung C.5: Konturen der Bildsequenz *Lieferwagen*.

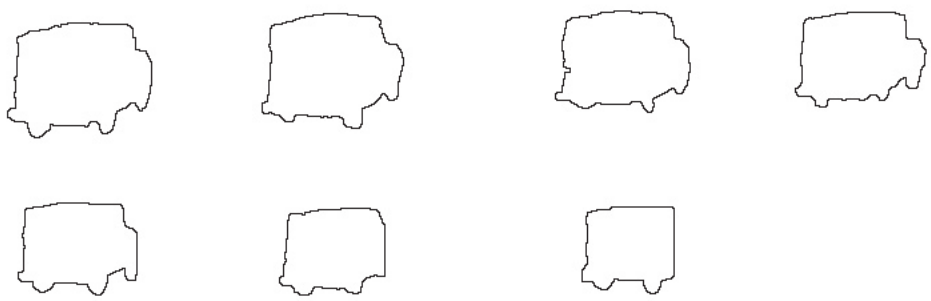

Abbildung C.6: Konturen der Bildsequenz *Lieferwagen* (Fortsetzung).

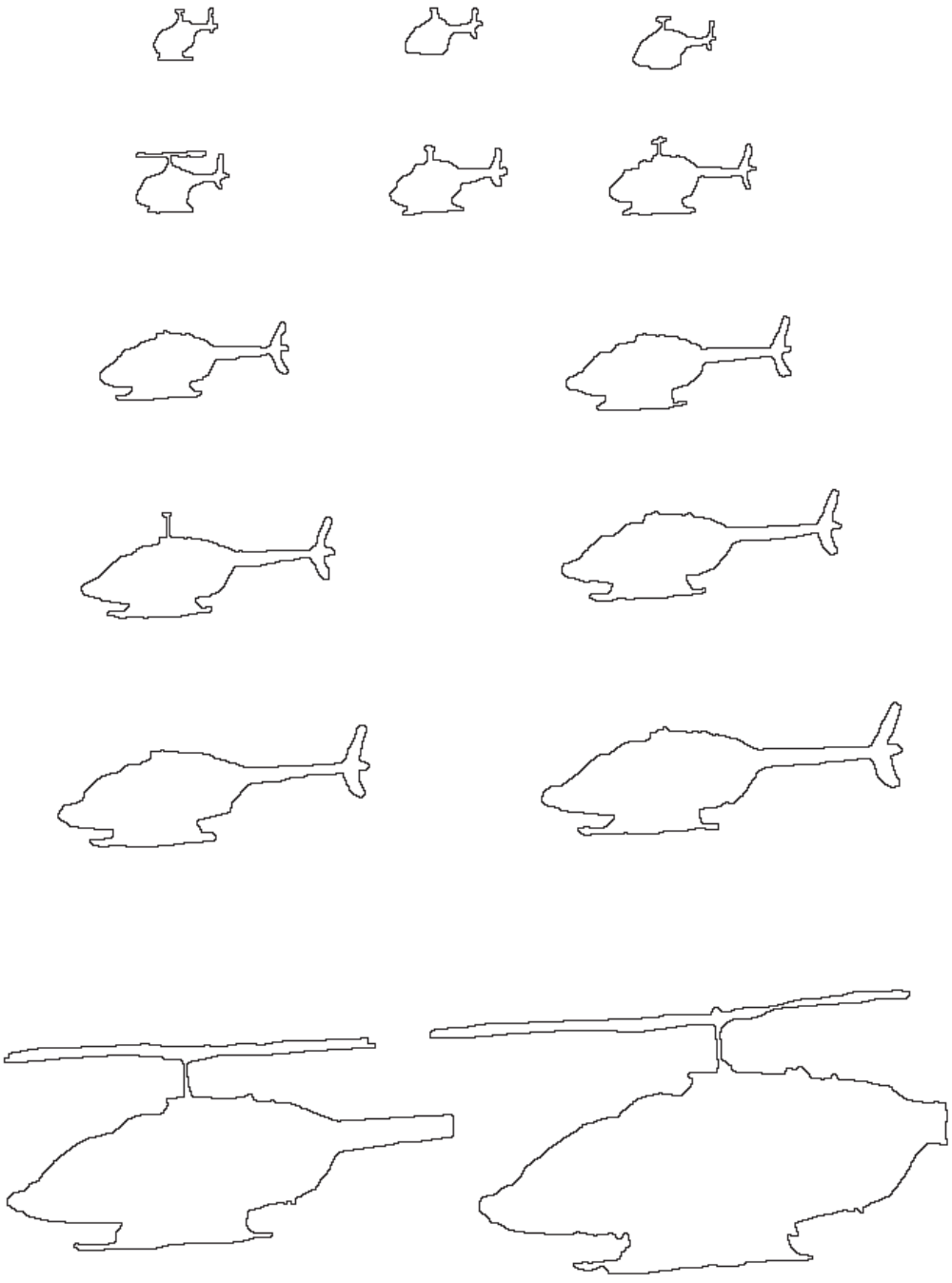

Abbildung C.7: Konturen der Bildsequenz *Hubschrauber*.

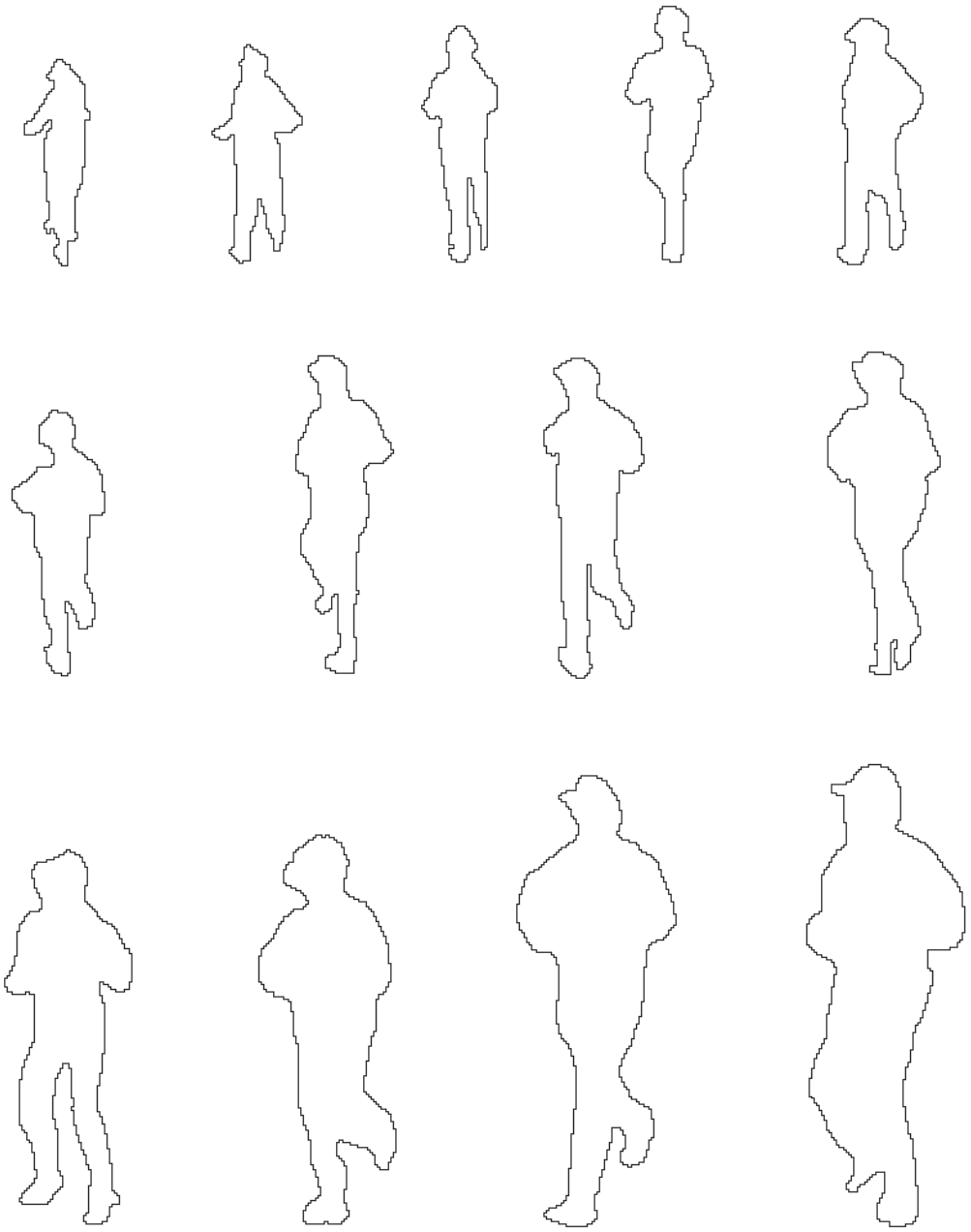

Abbildung C.8: Konturen der Bildsequenz *Jogger*.

Literaturverzeichnis

[1] S. Abbasi, F. Mokhtarian: *Shape Similarity Retrieval under Affine Transform: Application to Multi-View Object Representation and Recognition.* Proc. International Conference on Computer Vision, pp. 450-455, Corfu, Greece, 1999.

[2] S. Abbasi, F. Mokhtarian, J. Kittler: *Reliable Classification of Chrysanthemum Leaves through Curvature Scale Space.* Proc. International Conference on Scale-Space Theory in Computer Vision, pp. 284-295, Utrecht, The Netherlands, 1997.

[3] V. Blanz, M. J. Tarr, H. H. Bülthoff, T. Vetter: *What object attributes determine canonical views?* Technical Report No. 42, Max-Planck-Institut for Biological Cybernetics, Tübingen, Germany, 1996.

[4] T. Brandtberg, F. Walter: *Automated delineation of individual tree crowns in high spatial resolution aerial images by multiple-scale analysis.* Machine Vision and Appications, number 11, pp. 64-73, 1998.

[5] H. H. Bülthoff, S. Y. Edelman, M. J. Tarr: *How are three-dimensional objects represented in the brain?* CogSci Memo No. 5, Max-Planck-Institut for Biological Cybernetics, Tübingen, Germany, 1994.

[6] M. Carmo: *Differentialgeometrie von Kurven und Flächen.* Friedr. Vieweg & Sohn, Braunschweig, 1983.

[7] C. Christou, H. Bülthoff: *Differences between Active Explorers and Passive Observers in Virtual Scene Recognition.* Technical Report No. 62, Max-Planck-Institut for Biological Cybernetics, Tübingen, Germany, 1998.

[8] C. G. Christou, B. S. Tjan, H. H. Bülthoff: *Viewpoint Information Provided by a Familiar Environment Facilitates Object Identification.* Technival Report No. 68, Max-Planck-Institut for Biological Cybernetics, Tübingen, Germany, 1999.

[9] CueVideo: *Computer Science Research at Almaden - CueVideo.* Visual Media Management Group, http://www.almaden.ibm.com/cs/cuevideo2.html, IBM's Almaden Research Center, 2000.

[10] F. Cutzu, M. J. Tarr: *The representation of three-dimensional object similarity in human vision.* SPIE Proceedings from Electronic Imaging: Human Vision and Electronic Imaging II, 3016, San Jose, CA: SPIE, 460-471, 1997.

[11] K. Eriksson, D. Estep, P. Hansbo, C. Johnson: *Computational Differential Equations.* Cambridge University Press, Cambridge, New York, 1996.

[12] R. V. Gamkrelidze (Ed.): *Geometry I.* In: Encyclopaedia of Mathematical Sciences, Volume 28, Springer Verlag, Berlin, Heidelberg, 1980.

[13] R. C. Gonzalez, R. E. Woods: *Digital Image Processing.* Addison-Wesley Publishing Company, Reading, Menlo Park, New York, 1993.

[14] J. Hummel: *Objektsegmentierung und Bewegungsbestimmung in Ort-Zeit-Bildern.* Diplomarbeit, Universität Mannheim, Mannheim, 1996.

[15] B. Jähne: *Digitale Bildverarbeitung.* Springer-Verlag, Berlin, Heidelberg, 1997.

[16] B. Kimia, K. Siddiqi: *Geometric Heat Equation and Nonlinear Diffusion of Shapes and Images.* Computer Vision and Image Understanding, vol. 64, number 3, pp. 305-322, 1996.

[17] K. Kishimoto: *Characterizing Digital Convexity and Straightness in Terms of "Length" and "Total Absolute Curvature".* Computer Vision and Image Understanding, vol. 63, number 2, pp. 326-333, 1996.

[18] M. M. Lipschutz: *Differentialgeometrie.* McGraw-Hill Inc., Düsseldorf, New York, 1980.

[19] J. C. Liter, B. S. Tjan, H. H. Bülthoff, N. Köhnen: *Viewpoint Effects in Naming Silhouette and Shaded Images of Familiar Objects.* Technival Report No. 54, Max-Planck-Institut for Biological Cybernetics, Tübingen, Germany, 1997.

[20] R. Malladi, J. Sethian: *Image Processing: Flows under Min/Max Curvature and Mean Curvature.* Graphical Models and Image Processing, vol. 58, no. 2, pp. 127-141, 1996.

[21] A. Mayer: *Snakes: Aktive Konturen in der Bildsegmentierung.* Technical Report 59, Deutsches Krebsforschungszentrum Heidelberg, Heidelberg, Deutschland, 1994.

[22] F. Mokhtarian: *Silhouette-Based Isolated Object Recognition through Curvature Scale Space.* IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 17, no. 5, pp. 539-544, 1995.

[23] F. Mokhtarian: *A Theory of Multi-Scale, Torsion-Based Shape Representation for Space Curves.* Computer Vision and Image Understanding, vol. 68, number 1, pp. 1-17, 1997.

[24] F. Mokhtarian, S. Abbasi and J. Kittler: *Efficient and Robust Retrieval by Shape Content through Curvature Scale Space.* Proc. International Workshop on Image DataBases and MultiMedia Search, pp. 35-42, Amsterdam, The Netherlands, 1996.

[25] F. Mokhtarian, S. Abbasi, J. Kittler: *Robust and Efficient Shape Indexing through Curvature Scale Space.* Proc. British Machine Vision Conference, pp. 53-62, Edinburgh, UK, 1996.

[26] F. Mokhtarian: *CVSSP Home Page* Centre for Vision, Speech, and Signal Processing, http://www.ee.surrey.ac.uk/Research/VSSP/, University of Surrey, 2000.

[27] I. Muehl: *Objektsegmentierung unter Verwendung von Bewegungsvektoren.* Diplomarbeit, Universität Mannheim, Mannheim, 1995.

[28] T. Okatani, K. Deguchi: *Computation of the Sign of the Gaussian Curvature of a Surface from Multiple Unknown Illumination Images without Knowledge of the Reflectance Property.* Computer Vision and Image Understanding, vol. 76, number 2, pp. 125-134, 1999.

[29] S. Palmer, E. Rosch, P. Chase: *Canonical perspective and the perception of objects.* In J. Long, A. Baddeley (ed.), Attention and Performance IX, Hillsdale, NJ: Lawrence Erlbaum, 135-151, 1981.

[30] QBIC: *QBIC Home Page.* http://wwwqbic.almaden.ibm.com/, IBM's Query By Image Content, 2000.

[31] Surfimage: *Surfimage Web Demo.* http://www-rocq.inria.fr/cgi-bin/imedia/surfimage.cgi, 2000.

[32] M. J. Tarr: *Visual pattern recognition.* In A. Kazdin (ed.), Encyclopedia of Psychology, Washington, DC: American Psychological Association, in press.

[33] M. J. Tarr, H. H. Bülthoff: *Image-Based object recognition in man, monkey, and machine.* Cognition, Special issue on Image-based Recognition in Man, Monkey, and Machine, 67, 1-20, 1998.

[34] I. Vaisman: *A first course in differential geometry.* Marcel Dekker Inc., New York, Basel, 1984.

[35] G. Wallis: *Presentation order affects human object recognition learning.* Technical Report No. 36, Max-Planck-Institut for Biological Cybernetics, Tübingen, Germany, 1996.

[36] J. Weickert: *Anisotropic Diffusion in Image Processing.* B. G. Teubner, Stuttgart, 1998.

[37] J. Weickert: *Anwendung partieller Differentialgleichungen in der Bildverarbeitung.* Begleitende Unterlagen zur Vorlesung, Universität Mannheim, Sommersemester 1999.

[38] J. Weickert: *Coherence-enhancing diffusion of colour images.* Image and Vision Computing, Vol. 17, 201-212, 1999.

[39] T. J. Willmore: *An introduction to Differential Geometry.* Oxford University Press, Oxford, 1959.