Institut für Logik, Komplexität
und Deduktionssysteme (ILKD)
Universität Karlsruhe (TH)
Fakultät für Informatik

FARBBASIERTE SEGMENTIERUNG VON KÖRPERREGIONEN

Studienarbeit
von

Andy King

Februar 2002

Referent: Prof. Dr. A. Waibel
Betreuer: Dipl.-Inform. Rainer Stiefelhagen

Ich versichere hiermit, die vorliegende Studienarbeit selbstständig und ohne fremde Hilfe angefertigt zu haben. Die verwendeten Hilfsmittel und Quellen sind im Literatur- und Quellenverzeichnis vollständig aufgeführt.

Karlsruhe, 26. Februar 2002

Andy King

Inhaltsverzeichnis

Kapitel 1

Einleitung

Sowohl Menschen (wo und wer sie sind) als auch ihre Aktivitäten beziehungsweise ihr Verhalten durch Bildauswertung zu erkennen, das ist eines der wichtigsten Anwendungsgebiete im Maschinensehen.
Durch die verbesserte Mensch-Maschine-Interaktion, also wenn die Maschine intelligent, menschenähnlich und flexibel mit Menschen Daten austauschen und sie wahrnehmen kann, entstehen vielversprechende Anwendungen. Dazu zählen unter anderem virtuelle Realität, „kluge" Überwachungssysteme, intelligente Räume, verbesserte Benutzerschnittstellen, Bewegungsanalyse und modellbasierte Bildkodierung/-kompression (Tab. 1.1).

Ziel der vorliegenden Arbeit ist die Initialisierung von sogenannten Blobmodellen durch farbbasierte Segmentierung. Blobmodelle können zur Identifizierung einzelner Körperregionen beziehungsweise Körperteile, aber auch zur Unterscheidung verschiedener Personen benutzt werden.
So wurde beispielsweise an den „Interactive Systems Labs" (ISL) der Universität Karlsruhe und der Carnegie Mellon University die Farbmodellierung einzelner Teilnehmer einer Besprechung (neben anderen multimodalen Eingaben wie Sprache, Geräuschrichtung und Gesichtsfarbe) dazu benutzt, um zwischen Sprechern zu unterscheiden und eine Besprechung sinnvoll automatisch auszuwerten. In [14] wird dieser intelligente Besprechungsraum (sogenannter „Multimodal Meeting Room") genau beschrieben.

Aufbauend auf die bestehende Vordergrund-Hintergrund-Segmentierung, die die Basis der Personenverfolgung bildet und relevante Vordergrundobjekte liefert, wird in dieser Studienarbeit nun ein Farbmodell des dominierenden Vordergrundobjekts (der Person) gelernt.
Die Modelladaption erfolgt dabei durch den EM-Algorithmus.

Das folgende Kapitel beschäftigt sich mit den Grundlagen der Personenverfolgung mittels Videobildern, um allen Lesern einen breiten Hintergrund zu geben. Dies ist mehr oder weniger eine Zusammenfassung des Überblicks von Gavrila [9] .
Im dritten Kapitel wird auf die Blobmodellierung näher eingegangen. Dabei wird ein bestehendes System („Pfinder") betrachtet und die einzelnen Schritte zur Personenverfolgung erläutert.
Im vierten Kapitel werden zunächst allgemeine Segmentierungsansätze gezeigt. Dann folgt der Kern dieser Arbeit: dabei werden die Gauß-Mixturen zur Modellierung der Farbverteilung, der EM-Algorithmus, der eigentliche Clustering-Algorithmus und die Besonderheiten bei Mixturen näher beschrieben.
Das Kapitel ‚Experimente und deren Bewertung' veranschaulicht das bisher entwickelte System und zeigt anhand von Beispielbildern, wie gut die hier implementierte Farbsegmentierung funktioniert.
Das letzte Kapitel gibt eine Zusammenfassung der Arbeit und verweist auf Erweiterungsmöglichkeiten, die in der Zukunft erstrebenswert wären.

Anhang A beschreibt allgemeine statistische Modellierungsansätze.
Anhang B gibt einen Überblick über verschiedene Farbräume und zeigt wie aus dem Standardfarbraum RGB andere hier verwendete Farbräume durch Transformationen gewonnen werden können.
Im Anhang C sind begleitende Internetseiten zusammengefaßt und der Anhang D gibt Hinweise zu C++.
Die Implementierung der Algorithmen erfolgte unter Microsoft Visual C++ 6.0, eingebettet in ein größeres Softwareprojekt mit dem Namen „Person ID".

Allgemeines Anwendungsgebiet	Spezielle Anwendungen
virtuelle Realität (VR)	interaktive virtuelle Welten (z.B. intelligente Räume)
	(Video-)Spiele
	virtuelle Studios
	Animationen (z.B. synthetische Schauspieler)
	drahtlose VR-Schnittstellen
	Telekonferenz (z.B. Film, Werbung)
"kluge" Überwachungssysteme	Zugangskontrollsysteme
	Parkplatz-/Parkhausüberwachung
	Sicherung von Supermärkten/Verkaufsgeschäften
	Kontrolle von Geldautomaten
	Verkehrsüberwachung
verbesserte Benutzerschnittstellen	"menschenfreundliche" Interaktion
	automatische Übersetzung von Zeichensprache
	bequeme Steuerung mittels Gestik (im Gegensatz zu Tastatur, Joystick, Maus)
	Kommunikationsmöglichkeiten in sehr lauten Umgebungen (z.B. Flughäfen, Fabriken)
Bewegungsanalyse	inhaltsbasierte Gliederung/Erfassung von (Sport-)Videos ->Videodatenbanken
	angepaßtes Training (z.B. für Golf, Tennis)
	Choreographie (Tanz, Ballett)
	orthopädische Studien von Patienten
modellbasierte Bildkodierung	Videokompression bei sehr niedrigen Bit-Raten (MPEG-4)
	Steuerung eines sog. Video-Avatars durch high-level Beschreibung des Benutzers (Informationen über Kopf, Hände und Füße)

Tabelle 1.1: Überblick über die Anwendungsgebiete (aus [9])

Kapitel 2

Grundlagen der Personenverfolgung mittels Videobildern

Im vorliegenden Kapitel soll dem Leser zuerst ein Überblick über das Gebiet der Personenverfolgung mittels Videobildern gegeben werden. Dann werden verschiedene Methodiken vorgestellt und gezeigt, wie die Aktionserkennung funktioniert. Schließlich wird noch eine Bewertung der möglichen Ansätze gegeben.
Die Ausführungen entstammen hauptsächlich aus [9]: Gavrila spricht dabei allgemein vom „Looking at People"-Bereich innerhalb des Maschinensehens und versteht darunter die Auswertung von Bildern zur Erkennung von Menschen und deren Aktivitäten.

2.1 Überblick

Man unterscheidet beim „Looking at People" zwischen:
- Ganzkörperverfolgung
- Erkennung von Handgestik beziehungsweise Gesten als Eingabemöglichkeit
- Gesichtsauswertung: Schätzung der Kopfstellung, Gesichtserkennung, Gesichtsausdrücke und Lippenlesen

Eine weitere Unterscheidung ergibt sich durch die Variabilität bei den Sensoren:
- Sensorverfahren (sichtbares Licht, Infrarot, ...)
- Sensormultiplizität (monokular versus stereo), das heißt wie viele Kameras werden verwendet, um beispielsweise Tiefe im Raum zu erfassen
- Sensorplazierung (zentral oder verteilt)
- Sensormobilität (stationär versus bewegbar)

Die Hauptunterscheidung der Ansätze trifft man aber durch die Methodik und die Einteilung erfolgt hier im Großen und Ganzen nach der Dimension (Abb. 2.1).

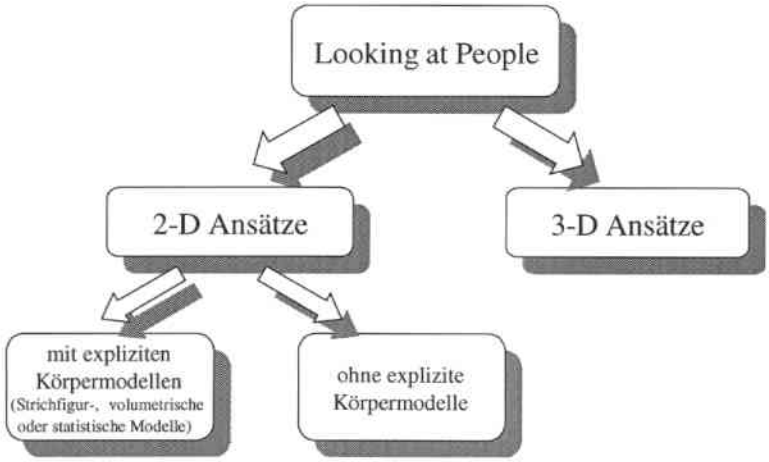

Abbildung 2.1: Einteilung der verschiedenen Ansätze

2.1.1 2-D Ansätze ohne explizite Körpermodelle

Hier wird menschliche Bewegung nur mit einfachen „low-level" 2-D Merkmalen (sogenannten „Features") beschrieben und die Auswertung erfolgt dann statistisch oder heuristisch. Oft werden morphologische Operationen angewendet, um Rauschen zu entfernen beziehungsweise zu unterdrücken.

Beispielsweise wurde in einem System zur Unterscheidung zwischen der Vorder- und Hinteransicht von Fußgängern (nach einer Größennormalisierung) ein sogenanntes Gitter auf die relevante Region gelegt: in jeder „Kachel" dieses Gitters wird anschließend ein einfaches Feature berechnet und dann werden noch alle Features zu einem einzigen Vektor zur festen Zeit t kombiniert. Durch Anwendung eines Ableitungsoperators erhält man so Wavelet-Koeffizienten, die als „low-level" Intensitäts-Features dienen. In einem Training wird dann noch bestimmt, welche Untermenge der Koeffizienten eigentlich sinnvoll ist und eine sogenannte „Support Vector Machine" (SVM) dient letztlich als Klassifikator zwischen Vorder- und Hinteransicht ([9]).

Bewegungsbasierte Segmentierungs- und Verfolgungstechniken wurden ebenfalls in diesem Gebiet des Maschinensehens in der Vergangenheit angewandt. So wurde zum Beispiel die durchschnittliche 2-D Bildgeschwindigkeit von Fußgängern berechnet: zuerst erhält man durch Korrelationstechniken über fortlaufende „Frames" (also zusammengehörige Videobilder) ein Bewegungsfeld (das anschließend noch räumlich und zeitlich geglättet wird) , danach erfolgt eine Quantisierung des Feldes, um anhand von Bewegungsrichtungen einzelne Regionen iterativ zu gruppieren ([9]).

Die Segmentierung von Bildern bei 2-D Ansätzen ohne explizite Modelle funktioniert oft nach folgendem Schema: Pixel werden anhand von farblichen (R,G,B) und räumlichen (x,y) Informationen durch gängige Clustertechniken gruppiert, wobei die zusätzliche räumliche Dimension das Clustern stabiler beziehungsweise erfolgreicher macht. Diese Pixelgruppen werden dann von einem Bild zum nächsten angepasst, was relativ einfach realisierbar ist.

2.1.2 2-D Ansätze mit expliziten Körpermodellen

Bei modell- und sichtbasierten Ansätzen wird a priori Wissen über menschliche Körper und deren Aussehen benutzt. Um mit Selbstverdeckung umgehen zu können, verwenden viele Systeme auch a priori Wissen über die Art der Bewegung (zum Beispiel nur bestimmte Posen und Haltungen) oder den Blickwinkel, unter dem das Ganze beobachtet wird.

Typische Modelle sind Strichmännchen, die mit Farbbändern oder sogenannten „Blobs" (siehe auch 3.1) ausgefüllt sind. Die Unterscheidung der Systeme erfolgt dabei nach Kanten/Farbbänder-, „Blob"- oder Punktmodellen. Letztere benutzen sogenannte Punkt-Features (zum Beispiel Position oder Geschwindigkeit) und haben bei mehr-Kamera-Systemen den Vorteil, dass die Features relativ leicht unter den verschiedenen Kameraperspektiven in Einklang gebracht werden können.

Statistische Körpermodelle sind sehr beliebt, um die Konturen von Händen oder ganzen Personen zu erkennen und zu verfolgen. Häufig kommen die sogenannten „active shape models" zum Einsatz: Zuerst werden in einer Trainingsphase typische Beispielformen durch bekannte räumliche Punkt-Features beschrieben. Dann wird mittels einer sogenannten „principal component analysis" (PCA) auf diesen Features die Parameterzahl der Beispielformen reduziert (man erhält dadurch die Eigenvektoren). Neben der Effizienz wird dadurch auch eine Generalisierung der Modelle erzeugt. Dies hat den großen Vorteil, dass man auch mit Deformationen umgehen kann, wobei die Konsistenz zur Trainingsmenge vorhanden sein muss. Nachteile dieser Methode können sein, dass die Features immer präsent sein müssen (keine Verdeckung), dass eine sehr gute initiale Schätzung für die Konvergenz der Methode notwendig ist und schließlich dass Zustände auftreten können, die in der Realität gar nicht vorkommen.

Bekanntester Vertreter der 2-D Ansätze mit expliziten Körpermodellen ist der „Pfinder" (person finder) vom MIT Media Lab [13]. Hier werden einzelne Körperregionen im Bild statistisch durch eine räumliche (x,y) und farbliche (Y,U,V) Gauß-Verteilung der zugehörigen Pixel beschrieben. Die Regionen entsprechen dabei typischerweise den Händen, dem Kopf, den Füßen, dem Hemd und der Hose. Zusätzlich wird noch ein statistisches Modell der Hintergrundregion mittels einer Gauß-Verteilung der Pixelfarbwerte erzeugt. Zuerst werden Pixel der Vordergrundregion als Abweichung von diesem Hintergrundmodell identifiziert. Anschließend werden dann die Körperregionen „erzeugt", wobei dieser Vorgang durch eine 2-D Kontur Auswertung unterstützt wird, um verschiedene Körperteile mittels Heuristiken zu finden.

Die Verfolgung beinhaltet dann eine Schleife mit folgenden 3 Teilaufgaben:

1. Vorhersage, wo sich die Person im neuen Bild befinden wird
2. Bestimmung der Wahrscheinlichkeit, dass ein Pixel zu einer bestimmten Körperregion oder zum Hintergrundmodell gehört und die entsprechende Zuweisung
3. Anpassung der Parameter der statistischen Modelle

Auf den „Pfinder" wird später noch genauer eingegangen (siehe 3.2), da dies das Referenzsystem für die hier am Institut entwickelte People Tracking Software ist.

2.1.3 3-D Ansätze

Allgemein ist es ziemlich schwierig aus 2-D Bildern die 3-D Bewegung abzuleiten beziehungsweise „wiederherzustellen". Doch bei der 3-D Personenverfolgung kann man sich das a priori Wissen über die Kinematik und die Proportionen des menschlichen Körpers zunutze machen. Außerdem unterstützen 3-D Formmodelle die Verfolgung, in dem bestimmte Ereignisse wie (Selbst-)Verdeckung oder (Selbst-)Kollision kompensiert beziehungsweise vorhergesagt werden.

Ein allgemeines „Framework" für die meisten modellbasierten Verfolgungssysteme (sowohl 2-D als auch 3-D) besteht aus vier Hauptkomponenten: Vorhersage, Synthese, Bildauswertung und Zustandsschätzung (Abb. 2.2).

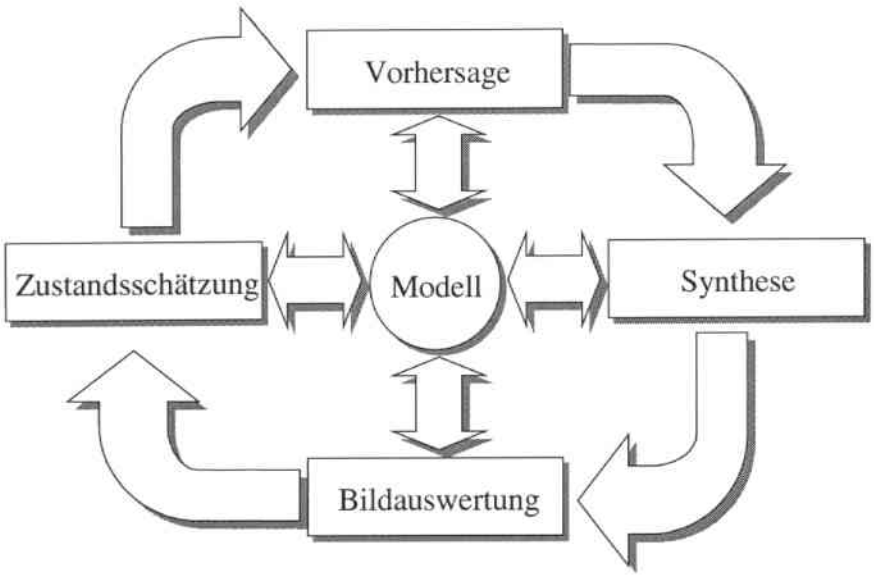

Abbildung 2.2: Hauptkomponenten eines modellbasierten Verfolgungssystems

Die Vorhersage-Komponente berücksichtigt die bisherigen Zustände (bis zur Zeit t) und macht eine Vorhersage für die Zeit t+1. Es gilt als wesentlich stabiler, die Vorhersage in einem höheren „Level" (also im Zustandsraum) als in einem niedrigen „Level" (im Bildraum) durchzuführen, da man dadurch leichter semantisches Wissen in den Verfolgungsprozess mit einbeziehen kann.
Die Synthese-Komponente übersetzt die Vorhersage von der Zustandsebene in die Mess- also Bildebene, wodurch dann die Bildauswertungs-Komponente ihren Fokus auf eine Untermenge von Regionen beziehungsweise Eigenschaften (Features) legen kann.
Schließlich berechnet die Zustandsschätzungs-Komponente mit Hilfe des segmentierten Bildes den neuen Zustand.

Bei erfolgreicher Implementierung der 3-D Verfolgung hat man den Vorteil, 3-D Verbindungswinkel als Features für anschließende Aktionszuweisungen zu verwenden. Diese Verbindungswinkel sind blickwinkelunabhängig und repräsentieren direkt die Körperhaltung. Außerdem sind sie im Vergleich zu 3-D Verbindungskoordinaten robuster im Bezug auf Variationen in der Körpergröße.

3-D Körpermodelle des Menschen bestehen allgemein aus zwei Komponenten:

- eine Skelettstruktur (das „Strichmännchen"): meistens einfache Segmente mit Verbindungs-/Eulerwinkeln, die verschiedene Freiheitsgrade besitzen
- eine Repräsentation für das das Skelett umgebende Fleisch: bei realistischer grafischer Modellierung bedient man sich einer flächenbasierten Repräsentation (zum Beispiel Polygone), doch beim Maschinensehen reicht eine volumentrische Darstellung (zum Beispiel Zylinder), da dadurch weniger Parameter auftreten

Es gibt dann einen „trade-off" zwischen der Genauigkeit und der Anzahl der Parameter des Modells. Ein frühes Beispiel ist der sogenannte „Bubbleman" (1980), bei dem Körperteile aus überlappenden Kugeln bestehen.
Eine ziemlich genaue Modellierung der Körperteile erhält man durch sogenannte „Superquadriken" (dies ist eine Verallgemeinerung von Ellipsoiden mit zusätzlichen „Rechteckigkeits"-Parametern entlang jeder Achse): dadurch sind so unterschiedliche Formen wie Zylinder, Kugeln, Ellipsoide und Hyperrechtecke darstellbar. Zusätzlich kann durch globale oder auch lokale Deformation der Superquadriken eine große Flexibilität der Modelle erreicht werden.

2.1.4 Aktionserkennung

Hat man sich schließlich für einen Ansatz entschieden, wird nun die Aktionserkennung interessant. Hierbei handelt es sich um ein Klassifikationsproblem, das zeitveränderliche Feature-Daten benutzt, die man aus dem früheren Segmentierungsschritt mittels der vorhin beschriebenen Techniken erhält.
Die Erkennung besteht nun darin, eine unbekannte Testsequenz mit einer „Bibliothek" von sogenannten „gelabelten" Sequenzen (also prototypischen Aktionen) zu vergleichen, die man durch Trainingsbeispiele erhält. Dabei sollten kleine räumliche und zeitliche Variationen innerhalb ähnlicher Klassen von Bewegungsmustern erlaubt sein.

Es gibt nun mehrere Methoden:

- raümlich-zeitliche „Templates" (nach vorhergehender Normalisierung), um einen generischen Aktivitätszyklus zu kennzeichnen: ein Zyklus wird dabei in eine feste Anzahl von Subintervallen zerlegt, für welche Bewegungs-Features berechnet werden. Das „Testtemplate" wird dann mit dem „Referenztemplate" bei allen möglichen zeitlichen Kombinationen mittels des „nearest centroid"-Algorithmus verglichen.

- „(Continuous) Dynamic Time Warping" (DTW) ist eine bekannte Technik des Dynamischen Programmierens, um Test- und Referenzmuster zu vergleichen, falls die zeitliche Einteilung nicht perfekt ausgerichtet ist, aber die zeitliche Ordnung stimmt. Wegen der gedanklichen Einfachheit und der Robustheit war DTW schon in den frühen Tagen der Spracherkennung sehr verbreitet.

- „Hidden Markov Modelle" (HMMs): Das sind nichtdeterministische Zustandsautomaten mit Übergangs- und Ausgabewahrscheinlichkeiten. Dabei wird sowohl ein Training, um die Anzahl der Zustände zu bestimmen und die Wahrscheinlichkeiten zu optimieren, als auch eine Klassifizierung benötigt. Für jede Bewegungsklasse wird ein eigenes HMM erzeugt.
  Die Möglichkeit von Trainingsdaten zu lernen, das fundierte mathematische „Framework" und die Fähigkeit, kontinuierliche (also nichtsegmentierte) Datenströme zu verarbeiten, machen HMMs im Vergleich zu DTWs „attraktiver". Aufgrund dieser Vorteile sind HMMs auch in der Spracherkennung weitverbreitet.

- „Neuronale Netze" (NN), die bei der Aktionserkennung seit kurzem im Einsatz sind.

Manchmal ist aber nicht die tatsächliche (zeitveränderliche) Bewegung von Interesse, sondern das (statische) Ergebnis (zum Beispiel mit dem Finger auf etwas zeigen). Dabei können beispielsweise regelbasierte Systeme zum Einsatz kommen, wobei hier die Kontext-Information von besonderer Bedeutung ist.
Es gibt auch Forschungsarbeiten, die in der Aktionserkennung mehr als ein Klassifikationsproblem sehen: es wird dabei versucht, die Bewegung in der Szene mit Hilfe von natürlicher Sprache („Bewegungsverben") innerhalb eines logischbasierten „Frameworks" zu beschreiben.

2.2 Zusammenfassung

Bei allen „Looking at People"-Ansätzen ist eine sogenannte nichtintrusive Vorgehensweise durch Kameraaufnahmen vorzuziehen. Intrusive Methoden (vor allem bei 3-D Ansätzen beliebt) verwenden zum Beispiel Marker am Körper oder allgemein ein aktives Sensorverfahren, was natürlich nicht immer wünschenswert beziehungsweise durchführbar ist.

Ob man eher einen 2-D oder 3-D Ansatz wählen sollte, hängt vor allem von der **Anwendung** ab.

Ein 2-D Ansatz ist für Anwendungen geeignet, bei denen eine genaue Erkennung der Körperhaltung nicht erforderlich beziehungsweise bei geringer Bildauflösung erst gar nicht möglich ist (zum Beispiel Verfolgung von Fußgängern). Es ist ebenfalls die einfachste und beste Lösung für Anwendungen mit einer einzigen Person, beschränkter Bewegung und einem einzigen Blickwinkel (zum Beispiel Erkennung von unterscheidbaren Handgesten, wobei die Kamera frontal auf die Hand gerichtet ist).

Ein essentielles Entwurfskriterium für 2-D Systeme ist, ob von vornherein explizite Modelle verwendet werden oder ob ein Lernansatz gewählt wird. Bei Systemen ohne explizite Körpermodelle ist es besonders wichtig, die Vordergrundregion initial genau zu bestimmen und zu segmentieren (zum Beispiel durch Hintergrundsubtraktion, Farbverteilung, Hindernisdetektion oder auch unabhängige Bewegungserkennung), weil eine fehlerhafte Verfolgung ansonsten nicht so schnell erkennbar ist.
Außerdem ist die richtige Normalisierung der Features, die man aus dieser Vordergrundregion erhält, unter Berücksichtigung sowohl der räumlichen als auch der zeitlichen Dimension ein Problem.

Ein 3-D Ansatz ist bei „indoor"-Anwendungen vorzuziehen, wo man sich eine hohe Unterscheidungsrate zwischen verschiedenen unbeschränkten und komplexen (vielen) menschlichen Bewegungen wünscht (zum Beispiel soziale Interaktionen wie Händeschütteln oder Tanzen). Beim 3-D erhält man eine präzisere und kompaktere Repräsentation des physikalischen Raumes, was eine bessere Vorhersage und auch das Umgehen mit Verdeckung/Kollision erlaubt. Außerdem wird es oft für virtuelle-Realitäts-Anwendungen benötigt.

Es sollte aber auch erwähnt werden, dass die sichtbasierte 3-D Verfolgung noch nicht ganz praxistauglich ist: es gibt relativ wenige Beispiele für die 3-D Gestenerkennung auf realen Daten und die meisten von ihnen verlangen sowohl Vereinfachungen (zum Beispiel beschränkte Bewegung, Segmentierung) als auch Einschränkungen (zum Beispiel Verarbeitungsgeschwindigkeit). Außerdem ist die Robustheit dieser Systeme ein großes Thema. Hier überzeugt vor allem der mehr-Kamera-Ansatz: Körperhaltungen und Bewegungen, die von einer Sicht aus mehrdeutig erscheinen (zum Beispiel durch Verdeckung oder Tiefe hervorgerufen), können durch eine andere Sicht „aufgelöst" werden.

Es gibt auf jeden Fall noch eine Vielzahl von Herausforderungen, die gelöst werden müssen, bevor die sichtgestützte 3-D Verfolgung wirklich praktisch angewendet werden kann. Dazu zählen:

- Die Modelgewinnungsfrage
  Das 3-D Modell ist a priori nicht komplett festgelegt, sondern es gibt zusätzlich zu den „Haltungsparametern" noch verschiedene „Formparameter", die mittels der Bilder geschätzt werden müssen (zum Beispiel durch eine separate Initialisierung mit bekannten Haltungen oder Bewegungen).

- Die Verdeckungsfrage
  (Selbst-)Verdeckung ist in den meisten Systemen noch ein weitgehend ungelöstes Problem und es gibt weder Kriterien, wann man die Verfolgung von Körperteilen anhalten und wiederbeginnen sollte noch gibt es eine klare Definition für Haltungsmehrdeutigkeiten.

- Die Modellierungsfrage
  Menschliche Modelle für die Sichtverfolgung sind zwar in Bezug auf Form und Gelenkverbindungen ausreichend parametrisiert, aber nur wenige beachten Beschränkungen wie Gelenkwinkelbegrenzungen und Kollision und noch weniger nehmen Rücksicht auf dynamische Eigenschaften wie Gleichgewicht. Im Gegensatz zu grafischen Anwendungen werden Farb- und Texturhinweise kaum genutzt. Außerdem können bestehende Systeme nicht mit locker sitzender beziehungsweise weiter Kleidung umgehen und Objekte, mit denen der Mensch interagiert (wie beispielsweise ein Buch oder ein Tisch) werden bisher nicht modelliert.

- Benutze Grundregeln
  Ein quantitativer Vergleich zwischen geschätzter und tatsächlicher Haltung ist sehr wichtig, um die Systeme zu bewerten und miteinander zu vergleichen. Damit die Simulationen realistisch sind, sollten sie Modellierungs-, Kalibrierungs- und Segmentierungsfehler beinhalten. Grundregeln erhält man am besten, in dem man Marker oder aktive Sensoren bei den realen Daten verwendet.
  3-D Daten kann man aber auch durch passive Sensorverfahren (zum Beispiel Triangulation) erhalten, die nicht auf Marker beruhen.

Sowohl für 2-D als auch 3-D Ansätze bleibt die Frage „Verfolgung versus Initialisierung" offen. Eine einfache Initialisierungsmethode, die jederzeit gestartet werden kann (vor allem, wenn die Verfolgung komplett schief läuft), ist verantwortlich dafür, wie robust das System in der realen Welt funktioniert.

Eine gewünschte Erweiterung ist die Detektion und Verfolgung mehrerer Leute in einer Szene. Einfache Techniken wie Hintergrundsubtraktion sind dann nicht mehr einsetzbar und es werden bessere beziehungsweise stärkere Modelle benötigt, um mit Verdeckungen und dem Entsprechungsproblem „Features – Körperteile" umgehen zu können.

Bei der Aktionserkennung ist besonders die Frage interessant, ob eine Menge von generischen Aktionen definiert und dann bei einer Vielzahl von Anwendungen verwendet werden kann. Dies könnten dann einzelne Aktionen (wie gehen, hüpfen oder sitzen) und Interaktionen mit Objekten (wie greifen, werfen oder drücken) oder anderen Leuten (wie Hände schütteln, küssen oder schlagen) sein. Außerdem bleibt die Frage offen, ob man passende Features bestimmen kann und welche Zuordnungsmethoden, die unabhängig von der Anwendung sein sollten, am besten funktionieren.

Die Klassifizierung der unterschiedlichen Aktionen erleichtert dann auch die Einführung einer symbolischen Komponente auf der obersten Ebene der Bildverarbeitung, um über die Szene „nachzudenken". Dafür bieten sich logischbasierte Ansätze wie Fuzzy Logic und temporale Logik an. Dabei ist nicht nur die Verbindung von der sensorischen zur symbolischen Ebene, sondern auch die umgekehrte Richtung wichtig: beispielsweise kann man dadurch zwischen einer feinskalierten (->Körperteile) und einer grobskalierten (->ganzer Körper) Verfolgung je nach Kontext wechseln.

Zusammenfassend kann man sagen, dass im „Looking at People"-Bereich zwar schon viele Fortschritte gemacht wurden, aber einige Fragen wie Bildsegmentierung, Gebrauch von Modellen, Verfolgung versus Initialisierung, mehrere Leute, Verdeckung und Rechenaufwand zum jetzigen Zeitpunkt immer noch offen sind.
Die Verknüpfung verschiedener Sensoren (Bereich, Infrarot, Geräusch,...) wird auf jeden Fall zu besseren Systemen führen.
Insgesamt entwickelt sich der „Looking at People"-Bereich laut Gavrila zum „Understanding People"-Bereich.

Kapitel 3

Blobmodellierung von Personen und Körperregionen

In diesem Kapitel soll der bereits im Grundlagenkapitel erwähnte bekannteste Vertreter der 2-D Ansätze mit expliziten Körpermodellen vorgestellt werden: das sogenannte „**Pfinder**" System vom MIT Media Lab [13]. Dabei wird die Person durch „Blobs" modelliert. Nun folgt kurz eine Erläuterung dieses Blob-Begriffs.

3.1 Der Blob-Begriff

Schon die sogenannten Gestaltpsychologen Anfang des letzten Jahrhunderts sahen es als natürlich an, atomare Teile einer Szene aufgrund der Nachbarschaft und der Erscheinung zu einem Klecks (im Englischen „blob" genannt) zu gruppieren beziehungsweise zusammenzufassen.
Heutzutage versucht man mit Hilfe der visuellen Kohärenz Pixel zu gruppieren und Bilder zu segmentieren. Aber oftmals benutzt man statt der Regionen selbst nur die Grenzen bzw. die Umrisse/Konturen dieser Region, was sich im Falle komplexer Szenen (also mit Leuten oder natürlichen Objekten) als unzuverlässig und schwer durchführbar erwiesen hat.
Pentland und Kauth haben 1977 in ihrer Veröffentlichung „Blob: An unsupervised clustering approach to spatial preprocessing of mss imagery." [11] die allgemein verwendete Blob-Darstellung entwickelt und eingeführt. Sie diente damals zur sehr kompakten und strukturell bedeutungsvollen Beschreibung von multispektralen Satelliten (MSS) Bildern. Dem sogenannten „Minimum Description Length"(MDL)-Prinzip folgend, werden hier Pixel aufgrund ihrer farblichen (beziehungsweise spektralen oder texturellen) und räumlichen Ähnlichkeit zu kohärenten Regionen (den Blobs) geclustert.

Pfinder verwendet genau diese Blob-Darstellung. Im folgenden Abschnitt wird das System genauer betrachtet und auch bewertet, da es als Referenzsystem für die hier am Institut entwickelte Software zur Personenverfolgung angesehen werden kann.

3.2 Pfinder als Referenzsystem

Pfinder (person finder) ist ein Echtzeitsystem zur Personenverfolgung und zur Verhaltensinterpretation, wobei sich in einer beliebig komplexen Szene nur eine Person befinden darf und die Kamera stationär sein muss (mit spezieller Hardware ist sogar die Echtzeitverfolgung bei Kamerarotation und Zoomen möglich). Es benutzt ein Multiklassen statistisches Modell aus Farbe und Form, um eine 2-D Repräsentation von Kopf, Händen und des ganzen Körpers bei unterschiedlichsten Blickwinkeln zu erhalten (Abb. 3.1).
Das System verwendet für die Erkennung und Verfolgung einen sogenannten „Maximum A Posteriori (Probability)"(MAP)-Ansatz und benutzt a priori Wissen über Menschen (also das erwähnte explizite Körpermodell), um das ganze zu beschleunigen und sich von Fehlern wieder zu „erholen". Der zentrale Algorithmus kann aber auch für die Verfolgung von Fahrzeugen oder Tieren angewendet werden.
Pfinder hat sich als sehr zuverlässig erwiesen und wird bei vielen Anwendungen wie drahtlosen Schnittstellen, Videodatenbanken und Kodierung bei geringer Bandbreite erfolgreich eingesetzt.

Abbildung 3.1: (links) Ausgangsvideobild, (Mitte) Segmentierung, (rechts) eine 2-D Repräsentation der Blobstatistiken

Im folgenden wird auf die Personen- und Szenenmodellierung, auf die Auswertungsschleife zur Verfolgung und auf die Initialisierung (Personendetektion und Aufbau des Multiblobmodells) näher eingegangen.

- Personenmodellierung

  2-D Punktcluster beziehungsweise 2-D Regionen haben (räumliche) Mittelwerte μ und Kovarianz-Matrizen K. Die räumliche Blobstatistik wird mit Hilfe eines (d-dimensionalen) Gauß-Modells wiedergegeben (siehe Anhang A.1):

$$\Pr(O) = \frac{e^{-\frac{1}{2}(O-\mu)^T \bullet (O-\mu)}}{\sqrt{(2\pi)^d |K|}}$$

  Zusätzlich gibt es noch eine „support map" $s_k(x,y)$ für jedes einzelne Pixel, die die tatsächliche Zugehörigkeit zu einem Blob k zeigt:

$$s_k(x,y) = \begin{cases} 1, (x,y) \in k \\ 0, sonst \end{cases}$$

Jeder Blob besitzt eine räumliche (x,y) und spektrale (Y,U,V) Komponente (hier wird also der YUV-Farbraum verwendet, siehe Anhang B). Die räumlichen und farblichen Verteilungen sind dabei unabhängig, das heißt die Kovarianzmatrix K_k ist blockdiagonal. Man könnte zusätzlich noch Bewegungs- und Texturmessungen zur Beschreibung der Blobs verwenden, aber das ist momentan zu rechenintensiv.

Jeder Blob kann auch eine detaillierte Darstellung seiner Form und Erscheinung haben, was als Unterschiede von der zugrundeliegenden Blobstatistik modellierbar ist. Die kompakte Repräsentation von Personen ist vor allem für Anwendungen mit geringer Bandbreite nützlich.

Die Statistik von jedem Blob wird rekursiv erneuert, das heißt Informationen aus dem neuesten Bild werden mit dem Wissen aus der aktuellen Klassenstatistik und der sogenannten „Priors" (also der Vorgaben) kombiniert (siehe Auswertungsschleife).

- Szenenmodellierung

Es wird angenommen, daß die Szene sich aus einer relativ statischen Umgebung (zum Beispiel einem Büro) und einer einzigen sich bewegenden Person zusammensetzt. Daher werden verschiedene Modelle für die Szene und die Person verwendet.

Die Szene (Klasse „Null") wird als Texturoberfläche modelliert, das heißt jedes Pixel gehört zu einer Gauß-Verteilung mit voller Kovarianzmatrix und einem Farbmittelwert. Eine der Hauptaufgaben von Pfinder ist es festzustellen, welche Pixel der Szene vom Menschen verdeckt werden und welche sichtbar sind.

In jedem „Frame" (also jedem neuen Bild) wird die Statistik der sichtbaren Pixel mit Hilfe eines einfachen Filters rekursiv erneuert:

$$\mu_t = \alpha(x,y,Y,U,V) + (1-\alpha)\mu_{t-1}$$

Dadurch werden Beleuchtungsänderungen und sogar Objektbewegungen (beispielsweise das Verschieben eines Buches, das natürlich immer noch zum Texturmodell gehört) kompensiert.

- Auswertungsschleife

Zur eigentlichen Personenverfolgung benötigt man nun im Prinzip vier Schritte:

1. Sage mit Hilfe der aktuellen Modelle vorher, wo sich die Person im neuen Bild befinden wird: Hier wird die räumliche Verteilung von jedem Blob mit Hilfe einfacher Newtonscher Dynamik-Gesetze (Position, Geschwindigkeit) und einem sogenannten „Kalman-Filter" geschätzt.

2. Berechne für jedes Pixel und für jedes Blobmodell sowie dem Szenenmodell die Wahrscheinlichkeit, dass das Pixel zu dem jeweiligen Modell gehört: hier kommt die sogenannte „Mahalanobis-Distanz" als Distanzmaß zum Einsatz (für eine genaue Erklärung dieses Distanzmaßes siehe Anhang A.1).
   Dabei machen einem vor allem Schatteneffekte zu schaffen. Deshalb macht man hier den folgenden Ansatz. Falls ein Pixel viel heller ist (also eine größere Y-Komponente besitzt) als von der Klassenstatistik vorhergesagt, dann braucht man sich um Schatteneffekte keine Gedanken machen. Nur wenn das Pixel dunkler als erwartet ist, dann wird die Chrominanz Information in Bezug auf die Helligkeit normalisiert (U*=U/Y, V*=V/Y).
   Aus Effizienzgründen werden die Farbmodelle sowohl im Standard YUV- als auch im normalisierten U*V*-Farbraum berechnet.

3. Bestimme mit Hilfe der berechneten Wahrscheinlichkeiten beziehungsweise Distanzen eine „support map" (also eine Zuweisung zu bestimmten Vordergrund-Blobs oder der Szene). Die Klassifikation erfolgt nach dem MAP (Maximum A Posteriori)-Prinzip, das heißt das Pixel wird der Klasse zugeordnet, bei der es die „beste" (also größte) Zugehörigkeitswahrscheinlichkeit beziehungsweise den kleinsten Distanzwert hat.
   Außerdem werden noch räumliche „Priors" und Zusammengehörigkeitsbedingungen (mit Hilfe von sogenanntem iterativen morphologischen Wachstum (Abb. 3.2)) zur Verbesserung des Ergebnisses verwendet.
   Zuerst wird durch eine Gauß-Mixtur, die alle Blobklassen enthält, eine Vordergrundregion (also die gesamte Person) morphologisch aufgebaut. Danach bilden sich dann die einzelnen Blobs ebenfalls mit morphologischem Wachstum unter der Bedingung, dass sich alles innerhalb der Vordergrundregion abspielt. Schließlich werden noch „2-D Markov Priors" angewendet, die die Wahrscheinlichkeiten für die Szenenklasse „glätten", da sonst die Blobs aufgrund von falschen Zuordnungen einzelner Pixel löchrig erscheinen.

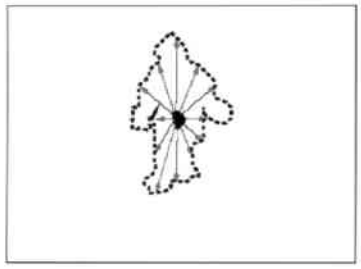

Abbildung 3.2: morphologisches Wachstum

4. Erneuere die Statistiken aller Blobmodelle und des Szenentexturmodells (durch Vergleich der alten und neuen Parameter kann man auch die dynamischen Modelle der Blobs erneuern): Der geschätzte neue Mittelwert μ_k ergibt sich direkt aus dem Erwartungswert E(x,y,Y,U,V) bzw. E(x,y,U*,V*) aller Pixel, die nun zu dieser Klasse k gehören. Die geschätzte Kovarianzmatrix K_k berechnet sich dann folgendermaßen:
   $K_k = E[((x,y,Y,U,V)-\mu_k)(x,y,Y,U,V)-\mu_k)^T]$ bzw. iterativ:
   $K_k = E[(x,y,Y,U,V)(x,y,Y,U,V)^T]-\mu_k\mu_k^T$

Um Fehlern in der Klassifikation und Instabilitäten im Modell vorzubeugen, sollte man Vorwissen für die spezielle Anwendung ausnutzen: beispielsweise ist die normalisierte Hautfarbe überraschenderweise bei unterschiedlicher Hautpigmentierung und Bräunung konstant. Verteilungen wie beispielsweise die der Hemdfarbe ändern sich zwar nicht, aber es gibt nur schwache „Priors" und über andere Sachen wie die Position einer Hand kann man von vornherein fast gar nichts sagen.

- Initialisierung von Pfinder

Pfinder startet mit einer Videosequenz, die keine Person enthält, um die Szene zu lernen. Typischerweise ist diese Sequenz relativ lang (1 Sekunde oder sogar mehr), um eine gute Schätzung der Farb-Kovarianz jedes Pixels zu erhalten.

- Personendetektion

Nachdem die Szene modelliert wurde, achtet Pfinder auf große Abweichungen von diesem Modell. Neue Pixelwerte werden im Farbraum mit Hilfe der Mahalanobis-Distanz mit der Klasse an der entsprechenden Stelle des bekannten Szenenmodells verglichen.
Falls eine genügend große Region von Veränderungen gefunden wurde (um Kamerarauschen auszuschließen), beginnt Pfinder ein Blobmodell der Person aufzubauen.

- Aufbau des Multiblobmodells der Person

Um die Blobmodelle zu initialisieren, verwendet Pfinder eine 2-D Kontur- bzw. Silhouettenanalyse (Abb. 3.3), die versucht, Kopf, Hände und Füße zu erkennen (dazu eignet sich am besten die sogenannte „star fish" Konfiguration). Diese Erkenntnis über die Orte der Körperteile wird auch als „Prior"-Wahrscheinlichkeit für die Bloberzeugung und –verfolgung benutzt (zum Beispiel wenn die Positionen von Gesicht und Händen im Bild bekannt sind, dann werden hautfarbene Blobs an diesen Stellen bevorzugt).

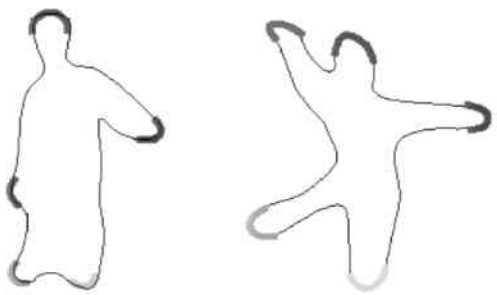

Abbildung 3.3: Features der Konturanalyse

Natürlich macht man sich die Farbverteilung der Person auch zu Nutze: für jede verschieden farbige Region wird ein Blob erzeugt. Typischerweise erhält man so Blobs für die Hände, den Kopf, die Füße, das Hemd und die Hose.

Falls eine Hand oder ein Fuß irgendwann verdeckt ist, dann wird der dazugehörige Blob vom Personenmodell einfach entfernt und wenn die Hand beziehungsweise der Fuß später wieder erscheint, dann wird entweder durch den Konturprozess (der Normalfall) oder den Farbtrennungsprozess ein neuer Blob erzeugt. Dies macht Pfinder gegenüber Verdeckungs- und Schatteneffekten sehr robust.

Die Konturanalyse kann die Features (wie Kopf, Hände, Füße) zwar in einem „Frame" finden, aber die Ergebnisse sind oftmals ziemlich verrauscht. Im Gegensatz dazu liefert der „Farbtracker" genaue Ergebnisse, aber er hängt sehr stark von der Stabilität der zugrundeliegenden Modelle und auch vom Vorhandensein der Features (zum Beispiel keine Verdeckung) ab. Die Aufgabe des Pfinders ist es nun, diese zwei „Tracker" mittels Heuristiken und „Prior"-Wahrscheinlichkeiten sinnvoll zu vereinigen, damit das System robust funktioniert.

Zu erwähnen ist, dass Pfinder mehrere Annahmen trifft, damit das ganze überschaubar bleibt. Pfinder erwartet zum Beispiel, dass die Szene bedeutend weniger Dynamik aufweist als die Vordergrundregion (also die Person), wobei kleine, allmähliche Änderungen in der Szene selbst oder der Beleuchtung kompensiert werden können.
Außerdem wird a priori angenommen, dass sich nur eine Person im Raum befindet. Wenn nun mehrere Personen präsent sind, ist das vor allem für die Gestenerkennung ein großes Problem.

Kapitel 4

Farbbasierte Segmentierung von Körperregionen

Um die Blobs beziehungsweise relevanten Regionen, das heißt die Person selbst, aber auch Körperteile im Bild zu finden, benutzt man eine sogenannte Segmentierung von Bildregionen. Dieser Schritt ist essentiell für ein System zur Personenverfolgung.

4.1 Segmentierungsmöglichkeiten

Allgemein gibt es mehrere Segmentierungsmöglichkeiten. Dazu zählen:

- Änderungsdetektion
- Kontur-/Silhouettenanalyse (siehe Pfinder)
- **farbbasierte Segmentierung**
- texturbasierte Verfahren
- ...

Hat man eine Bildfolge (hier die Originalvideosequenz), so bietet sich für den ersten Schritt eines Systems zur Personenverfolgung die Änderungsdetektion an, um die Person von der statischen Szene zu segmentieren.

Aufbauend auf diese Vordergrund-Hintergrund-Segmentierung, die bereits in einer früheren Studienarbeit ([8]) implementiert wurde (siehe auch 5.1), wollen wir in dieser Arbeit wie beim Pfinder den Farbansatz auf einen „Frame" (also ein einzelnes Videobild) anwenden. Es wird in diesem Schritt versucht, die Vordergrundobjekte beziehungsweise die Person direkt zu modellieren, das heißt es soll ein Farbmodell für verschiedene Körperregionen (wie Kopf-, Oberkörper-, Unterkörperbereich) geschätzt werden beziehungsweise es soll ein Blobmodell der Person durch die farbbasierte Segmentierung initialisiert werden.

Der Farbansatz hat gegenüber dem reinen Kontur-/Silhouettenansatz den großen Vorteil, dass er flächenbasiert ist, also bei verrauschtem „Input" viel robuster funktioniert und bessere Ergebnisse liefert.

Naiv könnte man annehmen, daß es eigentlich kein Problem darstellen sollte, die Farbregionen in einem Bild zu erfassen. Die einzelnen Pixelfarbwerte werden zum Beispiel fünf verschiedenen Farben (Hautfarbe, Haarfarbe, Hemdfarbe, Hosenfarbe und Schuhfarbe) zugeordnet: dadurch hat man eine Segmentierung der Person in verschiedene Farbbereiche.

Natürlich geht das nicht so einfach, denn es gibt mehrere Probleme:

- Kamerarauschen
- Verwaschene, zerknitterte Kleidung, wodurch die Farbe nicht mehr homogen erscheint
- Kleidung besitzt oft eine Textur
- Schatten-/Beleuchtungseffekte
- ...

Um mit diesen Problemen umgehen zu können, entscheidet man sich für eine statistische Modellierung, die sehr robust funktioniert.

Bevor auf den hier implementierten Farbansatz eingegangen wird, wird noch kurz gezeigt, wie andere die farbbasierte Methode zur Segmentierung benutzen.

Häufig kommt eine nichtparametrische Modellierung zum Einsatz (siehe Anhang A.2). So wird in [5] der sogenannte „mean shift"-Algorithmus verwendet und die Segmentierungsergebnisse sind sehr gut.
Die bekannteste Methode bei dieser Modellierungsgruppe ist aber der Histogramm-Ansatz. Der im Einleitungskapitel erwähnte intelligente Besprechungsraum des ISL ([14]) versucht anhand von charakteristischen Farbhistogrammen, einzelne Sprecher zu unterscheiden.
Ebenfalls mit Histogrammen arbeitet der sogenannte „head tracker" der Stanford University ([3]). Dabei kommt zusätzlich noch ein Intensitätsgradientenverfahren zum Einsatz, um eine Ellipse möglichst robust und schnell an die Kopfform anzupassen, und um den Kopfbewegungen zu folgen.

Die semiparametrische Modellierung ist ebenfalls sehr beliebt (siehe Anhang A.3). So verwendet Pfinder einen Mixtur-Ansatz, um das Multiblobmodell der Person aufzubauen.
Mit Mixturen von Gauß-Verteilungen wird auch in [1] gearbeitet. Dabei kommt für die Gruppierung der Pixel zu relevanten Regionen der EM-Algorithmus zum Einsatz. Das System verwendet außerdem noch Textur (siehe auch [2]), um eine sogenannte „Blobwelt" von Bildern einer Datenbank zu erhalten und neue, intuitivere Abfragemöglichkeiten zu generieren.

Genau diese Kombination aus Mixtur und EM-Algorithmus wollen wir in dieser Studienarbeit implementieren.

4.2 Segmentierung mit dem EM-Algorithmus

Um den Farbansatz wie in [1] anzuwenden, braucht man verschiedene Schritte, die aufeinander aufbauen. Die grobe Gliederung sieht so aus (in den nachfolgenden Abschnitten wird dann detailliert darauf eingegangen):

- Zuerst einmal sollten die Modelle entwickelt werden.
  In dieser Studienarbeit wird wie gesagt der Mixtur-Ansatz verwendet.

- Jetzt sollten die Modelle trainiert, also an die Daten angepasst werden.
  Für den Mixtur-Ansatz bietet sich der sogenannte „Expectation Maximization"(EM)-Trainingsalgorithmus an, da die Daten in diesem Fall nicht „gelabelt" sind (das heißt man kennt a priori ihre Klasse nicht) und weil man nie genau weiß, welche Farben auftreten werden.

- Schließlich sollten die einzelnen Pixel noch klassifiziert, also geclustert werden, was relativ leicht zu implementieren ist.

4.2.1 Modellierung der Farbverteilung mit Gauß-Mixturen

Wir verwenden eine d-dimensionale Gauß-Mixtur der folgenden Art (siehe Anhang A.3):

$$p(x) = \sum_{j=1}^{M} p(x|j)P(j), \qquad p(x|j) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_j|}} e^{-\frac{(x-\mu_j)^T \bullet (x-\mu_j)}{2\Sigma_j}}$$

Die Mixturverteilung soll dabei alle Farben der Vordergrundregion (Person) modellieren und die einzelnen Komponetenverteilungen spiegeln jeweils eine Farbklasse wider.
Wir machen die Vereinfachung (aus Gründen der Rechenkapazität), dass die Kovarianzmatrix Diagonalgestalt besitzt beziehungsweise ein skalares Vielfaches der entsprechenden Einheitsmatrix ist. Dabei wird also angenommen, dass die Varianzen statistisch unabhängig sind, was bei Farbanteilen durchaus sinnvoll ist.

Die Kovarianzmatrix beispielsweise für den RGB-Farbraum sieht dann folgendermaßen aus:

$$\Sigma_j = \begin{pmatrix} \sigma_{Rj}^2 & 0 & 0 \\ 0 & \sigma_{Gj}^2 & 0 \\ 0 & 0 & \sigma_{Bj}^2 \end{pmatrix} \quad \text{bzw.} \quad \Sigma_j = \sigma_j^2 I = \begin{pmatrix} \sigma_j^2 & 0 & 0 \\ 0 & \sigma_j^2 & 0 \\ 0 & 0 & \sigma_j^2 \end{pmatrix}$$

Zu schätzen sind also die Gewichts- beziehungsweise Mixingparameter $P(j)$, die einzelnen Mittel- beziehungsweise Erwartungswerte μ_j und die Kovarianzmatrizen Σ_j der multivariaten Gauß-Verteilungen.
Für die Adaption verwendet man den bekannten „Expectation Maximization"(EM)-Algorithmus, auf den nun im folgenden Abschnitt genauer eingegangen wird.

4.2.2 Der „Expectation Maximization"(EM)-Algorithmus

Verwendet man Standardtechniken für die Maximum Likelihood Schätzung der Mixturparameter (zum Beispiel Gradientenabstieg), so erhält man ein nichtlineares Optimierungsproblem, das in der Praxis nicht wünschenswert beziehungsweise nicht lösbar ist. Deshalb verwendet man einen iterativen Suchalgorithmus, der einem garantiert, dass die Likelihood bei jeder Iteration steigt bis man schließlich ein (lokales) Maximum erhält. Es handelt sich also um eine direkte Methode, um die Parameter zu berechnen beziehungsweise der Likelihood entsprechend anzupassen.
Dies ist genau die Vorgehensweise des „Expectation Maximization"(EM)-Algorithmus ([4], [6], [12]).
Die iterative Berechnung der „neuen" Parameter sieht dann folgendermaßen aus
(N bezeichnet dabei die Anzahl der Pixel der zu untersuchenden Vordergrundregion und x^n steht für den Farbvektor eines speziellen Pixels):

- $$\mu_j^{neu} = \frac{\sum_n P^{alt}(j|x^n)x^n}{\sum_n P^{alt}(j|x^n)}$$
- $$(\sigma_j^{neu})^2 = \frac{1}{d}\frac{\sum_n P^{alt}(j|x^n)\left\|x^n - \mu_j^{neu}\right\|^2}{\sum_n P^{alt}(j|x^n)}$$
- $$P(j)^{neu} = \frac{1}{N}\sum_n P^{alt}(j|x^n)$$

Wie man an den Formeln sieht, sollte zuerst der neue Mittelwert und dann die neue Varianz berechnet werden.
Eine zentrale Rolle spielt die a posteriori Wahrscheinlichkeit $P(j|x)$, die sich mit Hilfe der Formel von Bayes berechnen läßt:

$$P(j|x) = \frac{p(x|j)P(j)}{p(x)}$$

Der aktuelle Farbmittelwert der j-ten Komponentenverteilung ist also der Mittelwert aller untersuchten Farbvektoren, wobei dann jeweils über die a posteriori Wahrscheinlichkeit, dass der Farbvektor genau zu dieser Komponente beziehungsweise Farbklasse gehört, gewichtet wird.
Ähnlich verhält es sich mit der Berechnung der neuen Varianz.
Der Gewichtsparameter der j-ten Farbklasse ergibt sich über die a posteriori Wahrscheinlichkeiten für diese Komponente, wobei dann über alle Pixel gemittelt wird.

4.2.3 Zuordnungs-/Clustering-Algorithmus

Hat man die „richtigen" Parameter für die jeweiligen Komponentenverteilungen beziehungsweise Farbklassen bestimmt, ist es nun möglich, die einzelnen Pixel anhand der zugehörigen Farbvektoren zu Farbregionen zu gruppieren beziehungsweise zu clustern. Dies sollte die gewünschte Segmentierung in bestimmte Körperregionen liefern.

Die Frage ist nun, auf welche Art und Weise kann man die einzelnen Pixel sinnvoll und auch zuverlässig einer bestimmten Mixturkomponente beziehungsweise Farbklasse zuordnen.

Hier kommt wie schon beim Pfinder erwähnt das „Maximum A Posteriori"(MAP)-Prinzip zum Einsatz, das heißt das Pixel wird der Klasse zugeordnet, bei der es die „beste" (also größte) Zugehörigkeitswahrscheinlichkeit hat. Dies bedeutet man geht $P(j|x)$ für alle Klassen durch, wobei es natürlich genügt, den Term $P(x|j)P(j)$ (siehe Formel von Bayes) auf einen Maximalwert zu untersuchen.

Wir berechnen hier aber nicht direkt die Wahrscheinlichkeiten $P(x|j)$, sondern aus Effizienz- und Genauigkeitsgründen den Logarithmus davon:

$$\ln P(x|j) = -\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j) - \frac{1}{2}\ln|\Sigma_j| - \frac{d}{2}\ln(2\pi)$$

Die konstanten Terme braucht man dabei nicht berücksichtigen, deshalb genügt es, den folgenden Ausdruck:

$$-(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j) - \ln|\Sigma_j|$$

mit dem Gewichtsparameter $P(j)$ zu multiplizieren.

4.2.4 Besonderheiten bei Mixturen

Bei der Modellierung mit Gauß-Mixturen und der Adaption durch den EM-Algorithmus gibt es noch Besonderheiten, die man beachten sollte.

So ist die Anzahl der Normalverteilung, also die Variable M, ein offener Parameter. Bei unseren Experimenten wie auch bei anderen (siehe [1]) hat sich herausgestellt, daß 4 bis 6 Komponentenverteilungen ausreichend sind (wobei diese Verteilungen dann grob gesagt der Haut-, Haar-, Hemd-, Hosen- und Schuhfarbe entsprechen).

Natürlich könnte man auch einen „bottom-up"-Ansatz verfolgen: man nimmt sehr viele Gauß-Verteilungen und die Auswahl der relevanten Verteilungen erfolgt über die Mixturgewichte. Dadurch kristallisiert sich dann die optimale Anzahl M heraus und man bekommt eine sehr genaue Modellierung. Dies hat aber auch den Nachteil, dass möglicherweise zu viele Klassen eingeführt werden, was die Regionenerkennung erschwert und außerdem ist dieses Verfahren sehr rechenintensiv.

Um den EM-Algorithmus zu initialisieren, braucht man Startwerte für die Mittelwerte, Varianzen und Mixturgewichte. Eine häufige und auch hier implementierte Vorgehensweise setzt die Mittelwerte einfach zufällig auf irgendwelche Datenpunkte, also auf irgendwelche Farbvektoren der Vordergrundpixel. Der Varianzwert einer Komponente wird dann mit dem Abstand vom zugehörigen Farbmittelwert zum nächstgelegenen Komponentenzentrum initialisiert. Als Distanzmaß verwendet man den Euklidischen Abstand im jeweiligen Farbraum. Die Mixturgewichte werden zu Beginn des Trainings alle auf den Wert 1/M gesetzt.

Um das ganze etwas „deterministischer" und effizienter zu machen, wurde in dieser Studienarbeit auch ein bekannter Clustering-Algorithmus, der sogenannte „k-Means"-Algorithmus (hier k=M!) ([4], [7], [12]) für die initiale Wahl der Mittelwerte implementiert.

Dieser Algorithmus gliedert sich grob in folgende Teilschritte:

1. Wähle zufällig k Mittelwerte.
2. Weise jeden Datenpunkt x dem nächsten Mittelwert zu und
3. Berechne die Mittelwerte neu. Wiederhole 2. bis sich die Gruppierung nicht mehr ändert.

Schließlich ist die Bestimmung der Trainingszyklen des EM-Algorithmus ein offenes Problem. Am sichersten wäre ein Vergleich der Likelihood $L = \prod_{n=1}^{N} p(x^n)$ beziehungsweise der (Log)Likelihood $\ln L = \sum_{n=1}^{N} \ln p(x^n)$ aufeinanderfolgender Iterationen, was aber sehr rechenintensiv ist und das Verfahren somit sehr verlangsamt (man beachte, dass bei einer Bildauflösung von 640x480 bis zu 307200 Vordergrundpixel vorliegen können!).
Bei unseren Experimenten hat sich herausgestellt, daß 5 bis 10 Iterationen für gute Segmentierungen vollkommen ausreichend sind.

Kapitel 5

Experimente und deren Bewertung

5.1 Bisher entwickeltes System (Vordergrund-Hintergrund-Segmentierung)

Den Grundstein für das bisher entwickelte System legte Dirk Focken in seiner Studienarbeit „Adaptive Hintergrundmodelle zur Personenverfolgung“ [8].
Mit Hilfe von adaptiven Hintergrundmodellen auf Pixelebene wird hier die Silhouette von Personen beziehungsweise von bewegten Objekten aus einer Bildfolge extrahiert (Vordergrundsegmentierung) und mit Zusatzinformationen wie dem Schwerpunkt und der sogenannten „Bounding Box“ (das ist ein Rechteck, das die einzelnen Objekte gegeneinander abtrennt) versehen.

Abbildung 5.1 veranschaulicht das bisher entwickelte System.

Abbildung 5.1: Personenverfolgung mittels Vordergrund-Hintergrund-Segmentierung

Wie man sieht, kann die Segmentierung mehrere Vordergrundobjekte liefern, die zum Teil völlig irrelevant für den Verfolgungsprozess sind (zum Beispiel bedingt durch Spiegelungs- und Schatteneffekte (siehe auch Abb. 5.2)). Um die eigentliche Person im Bild zu finden, kann man die folgende Heuristik verwenden: Man betrachtet das Vordergrundobjekt, dessen „Bounding Box“ die größte Fläche besitzt und damit das Bild sozusagen „dominiert“ (in Abb. 5.1 ist das Objekt Nummer 0).

5.2 Benutzeroberfläche der Farbsegmentierung

In dieser Studienarbeit wurden alle wichtigen Farbräume implementiert und ihr Nutzen für die farbbasierte Segmentierung getestet.
Dabei macht man die Unterscheidung zwischen „normalen" Farbräumen und perzeptuellen Farbräumen (für eine detaillierte Erklärung der einzelnen Farbräume siehe Anhang B).

Die erste Gruppe beruht auf kartesischen Koordinaten und beinhaltet folgende Farbräume:

- **RGB** (red, green, blue)
- **NRGB**: wie RGB, aber Normalisierung des Wertebereiches zwischen 0 und 1
- **NRG**: helligkeitsnormalisierter 2-dimensionaler Farbraum, der sich vom RGB-Raum ableitet
- **YUV**: besitzt eine Helligkeitskomponente Y (Luminanz) und 2 Farbkomponenten U und V (Chrominanz)
- **U*V***: helligkeitsnormalisierter 2-dimensionaler Farbraum, der sich aus dem YUV-Farbraum ergibt (U*=U/Y, V*=V/Y)
- **YIQ**: ähnlich wie der YUV-Farbraum aufgebaut (1 achromatischer und 2 chromatische Kanäle), nur andere Farbkomponenten
- **TS** (tint, saturation): 2-dimensionaler Farbraum, der auf den NRG-Raum aufsetzt

Die zweite Gruppe (perzeptuelle Farbräume) basiert auf zylindrischen Koordinaten:

- **HSV** (hue, saturation, value)
- **HLS** (hue, lightness, saturation)

Abbildung 5.2 veranschaulicht die Benutzeroberfläche der farbbasierten Segmentierung.

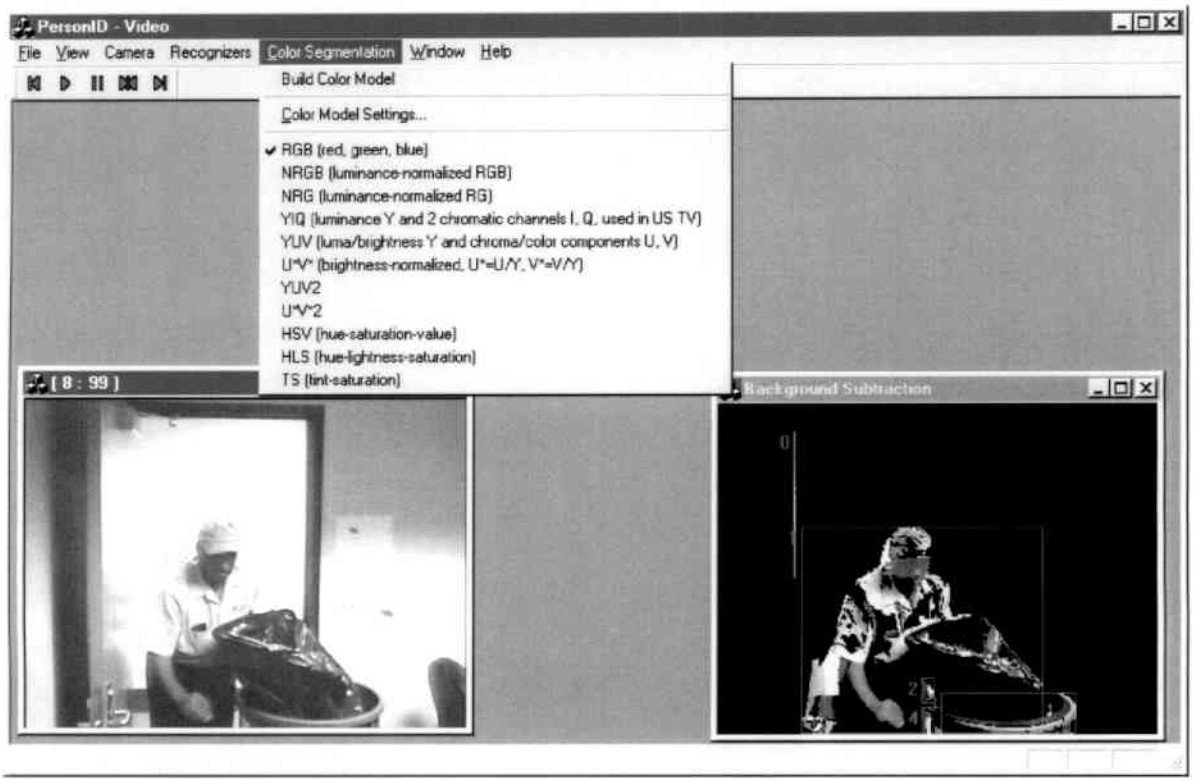

Abbildung 5.2: Personenverfolgung und Auswahl des Farbraumes

Über den Menüpunkt „Color Model Settings...“ gelangt man zu einem Farbmodell-Dialog (Abb. 5.3), der die Besonderheiten bei Mixturen und beim EM-Algorithmus (siehe 4.2.4) berücksichtigt.

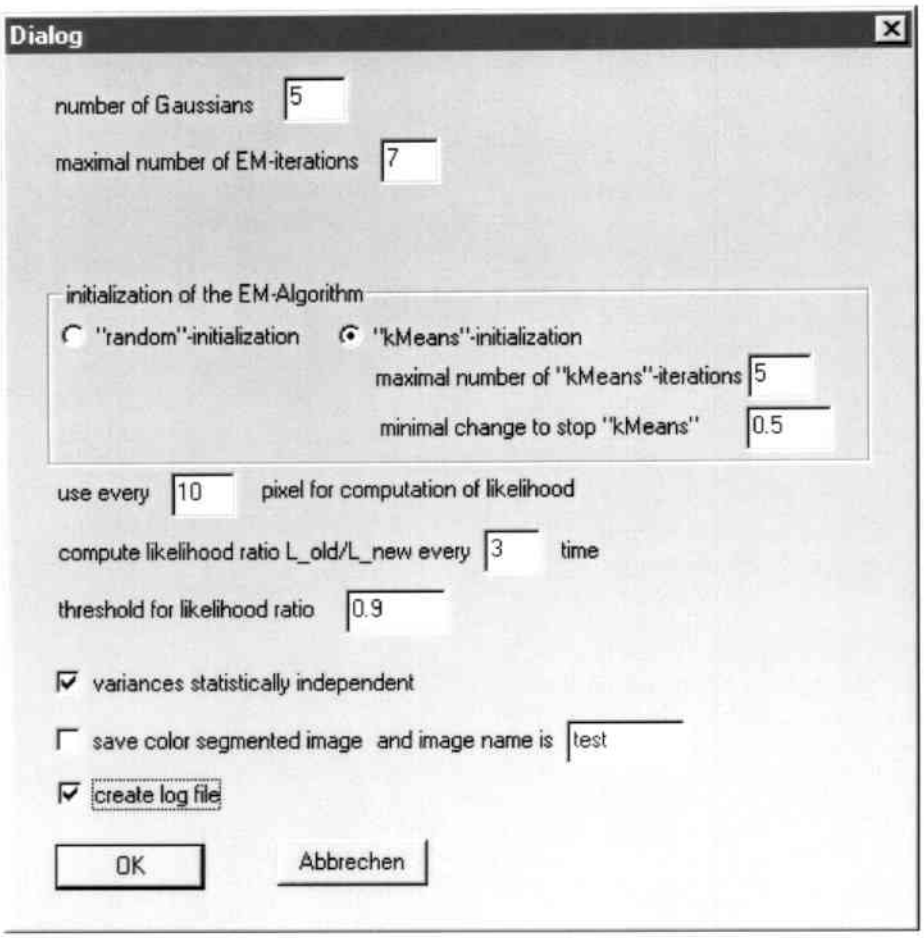

Abbildung 5.3: Farbmodell-Dialog

5.3 Beispielhaft segmentierte Personenbilder

Um die implementierte Segmentierung bewerten zu können, wurden einige Videosequenzen mit verschiedenen Personen aufgenommen. Abbildung 5.4 zeigt eine gewünschte Bildsegmentierung im bestehenden System (die einzelnen Körperregionen sind dabei jeweils mit dem geschätzten Farbmittelwert der zugehörigen Farbklasse eingefärbt).

Abbildung 5.4: (links oben) Ausgangsvideobild, (rechts oben) Vordergrund-Hintergrund-Segmentierung, (unten) farbbasierte Segmentierung von Körperregionen

Die nachfolgenden Abbildungen zeigen jeweils das Originalbild und das Ergebnis der farbbasierten Segmentierung in verschiedenen Farbräumen.
Die Beschriftung der Ergebnissegmentierungen ist dabei folgendermaßen zu lesen: Namen des Farbraumes_Anzahl der Gauß-Verteilungen_Anzahl der EM-Iterationen_Art der Initialisierung (r=random/zufällig, k=k-Means mit Angabe der Trainingszyklen)_Aufbau der Kovarianzmatrizen (i=(statistically/statistisch) independent/unabhängig beziehungsweise diagonal, is=skalares Vielfaches der entsprechenden Einheitsmatrix).

Es ist wichtig zu bemerken, dass die Ergebnisse natürlich von der vorausgehenden Vordergrund-Hintergrund-Segmentierung abhängen und deshalb etwas „zerklüftet" wirken.

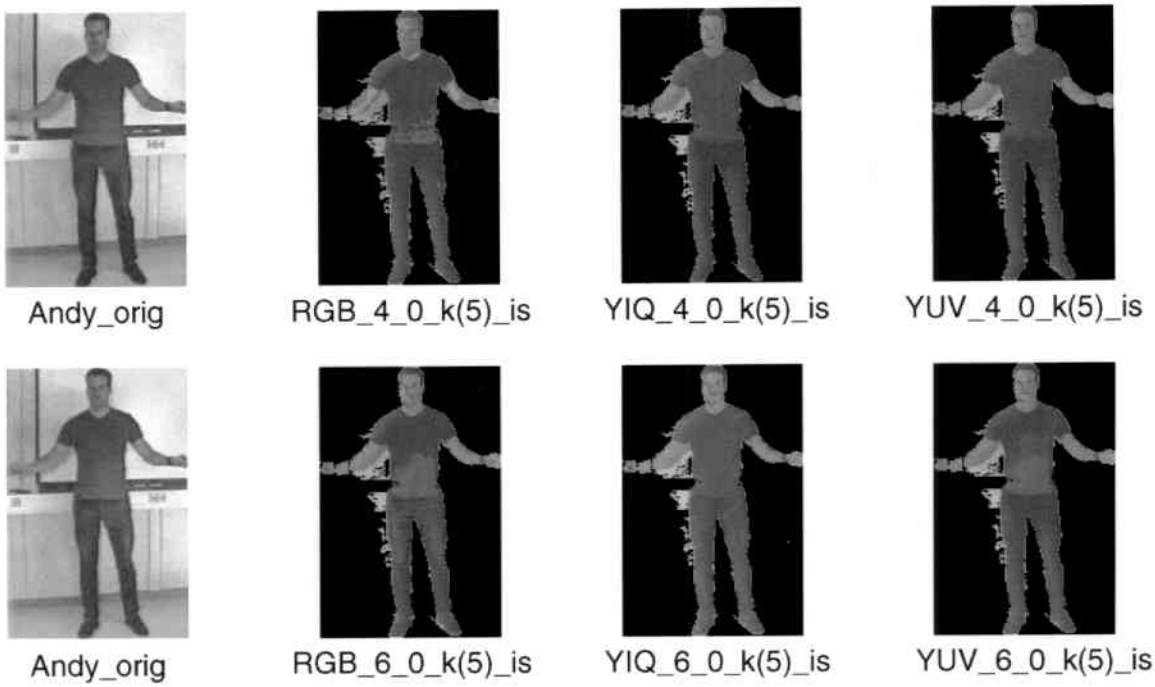

Abbildung 5.5: Beispielsegmentierungen I

Abbildung 5.5 zeigt die Ergebnisse, wenn man nur den k-Means-Algorithmus für die Adaption der Mittelwerte verwendet. Die Segmentierungen sind erstaunlich gut. Auffallend ist, dass der RGB-Farbraum sehr auf Licht- und Schatteneffekte reagiert.
Die Segmentierung funktioniert bei 4 vorgegebenen Farbklassen besser als bei 6 Komponentenverteilungen.

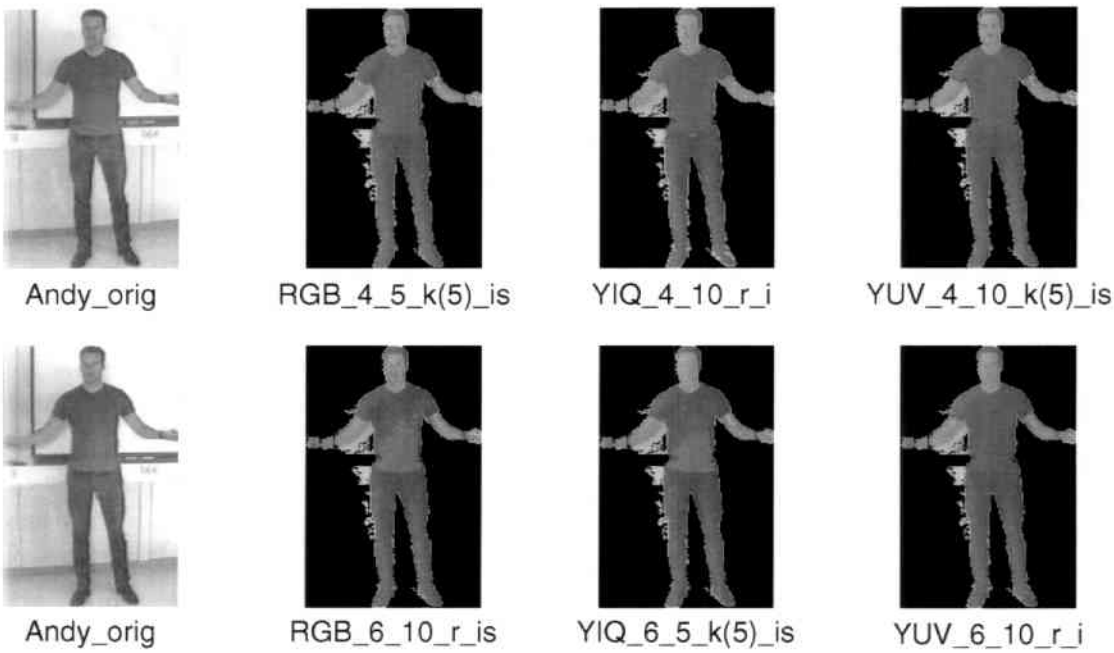

Abbildung 5.6: Beispielsegmentierungen II

In Abbildung 5.6 sind nun die Ergebnisse mit Hilfe des EM-Algorithmus zu sehen. Auffallend ist wieder, dass der RGB-Farbraum schlechter funktioniert wie der YIQ- beziehungsweise der YUV-Farbraum, da diese die Helligkeit (Y) separat kodieren.
Die Anzahl der vorgegebenen Farbklassen ist nicht mehr so bedeutungsvoll wie beim „reinen" k-Means-Algorithmus, da die EM-Adaption zusätzlich die Mixingparameter also die Gewichtung der Farbklassen berücksichtigt.

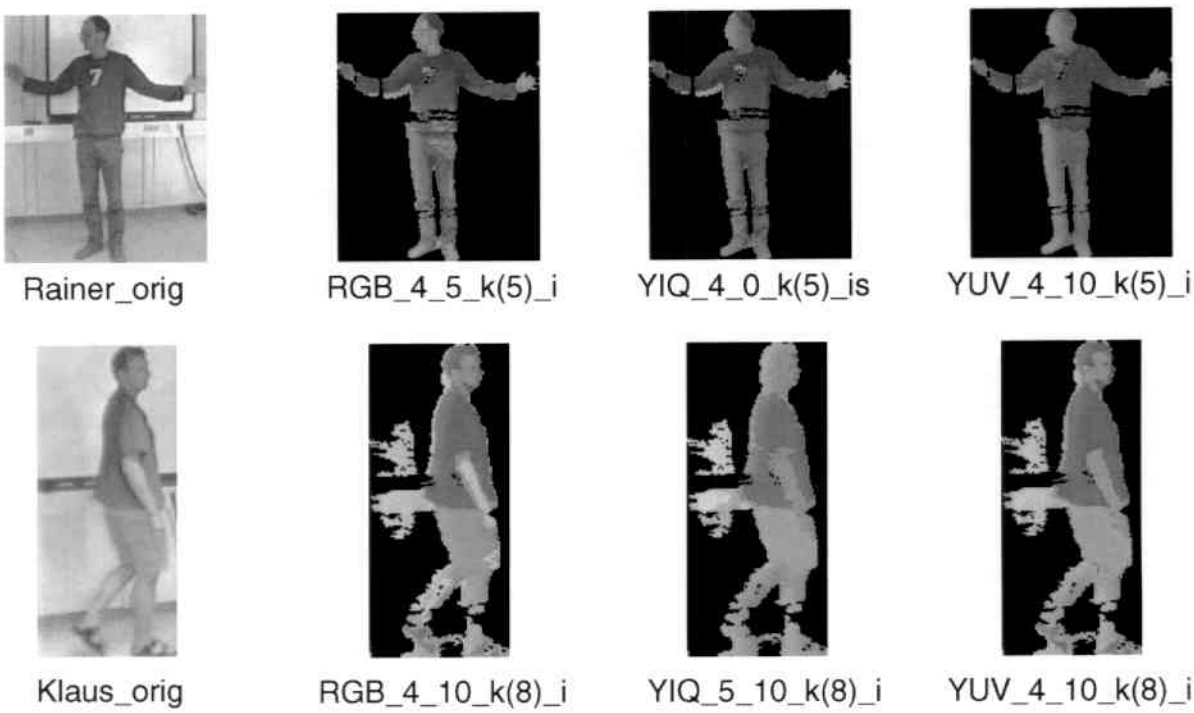

Abbildung 5.7: Beispielsegmentierungen III

In Abbildung 5.7 sind verschiedene Einstellungen und die entsprechenden Ergebnisse zu sehen. Hier funktioniert der YUV-Farbraum für die Segmentierung in einzelne Regionen sehr zuverlässig und ist damit die beste Wahl.
Interessant bei der Regionenerkennung im Bild oben rechts ist die Modellierung der hellen Schuhe: dort treten aufgrund der vier vorgegebenen Farbklassen hautfarbene Blobs auf.

Insgesamt hat sich bei den Experimenten herausgestellt, dass die Segmentierung in 3-dimensionalen Farbräumen robuster und besser funktioniert als in den 2-D Räumen NRG, U*V* und TS.
Die besten Ergebnisse über viele Segmentierungen gesehen lieferte der YUV-Farbraum.

Eigentlich nur für die Initialisierung des EM-Algorithmus vorgesehen, liefert der k-Means-Algorithmus alleine erstaunlich gute Ergebnisse. Trotzdem macht das EM-Training auf jeden Fall Sinn, da die Parameter (Mittelwerte, Varianzen und Mixturgewichte) besser an die Eingabedaten angepasst werden und somit die Segmentierung robuster funktioniert.

Die Anzahl der vorgegebenen Farbklassen hängt natürlich vom jeweiligen Eingabebild ab. Doch hat sich bei den Experimenten gezeigt, dass 4 bis 6 Komponentenverteilungen für eine sinnvolle Erkennung beziehungsweise Modellierung von Körperregionen normalerweise vollkommen ausreichend sind (siehe auch 4.2.4).

Kapitel 6

Zusammenfassung und Ausblick

In dieser Studienarbeit wurde ein farbbasierter Ansatz zur Segmentierung von Bildregionen implementiert. Das Verfahren kann dazu benutzt werden, Farbmodelle von Personen zu schätzen oder sogenannte Blobmodelle von Personen zu initialisieren. Diese können dann zur Verfolgung einzelner Körperregionen in Videosequenzen benutzt werden.

Bei dem implementierten Verfahren wird die Farbverteilung des Bildes einer Person durch eine Mixtur von Gauß-Verteilungen modelliert. Die Adaption der Parameter erfolgt dabei mit dem EM-Algorithmus. Eine Segmentierung des Bildes erhält man dann durch Zuordnung der Pixel zu den einzelnen Komponenten der Gauß-Mixtur.

Es wurden verschiedene Experimente zur Bildsegmentierung von Videosequenzen sich bewegender Personen durchgeführt. Dabei wurde zuerst mit Hilfe eines bereits implementierten Verfahrens zur Unterscheidung zwischen Vorder- und Hintergrundpixeln die Person vom Hintergrund segmentiert. Es wurde dann untersucht, welche Farbräume sich gut für die farbbasierte Segmentierung eignen. Dabei stellte sich heraus, dass auf den aufgenommenen Testbildern der YUV-Farbraum durchweg die besten Segmentierungsergebnisse lieferte.

Es konnte auch festgestellt werden, dass durch die Parameteradaption durch den EM-Algorithmus im Vergleich mit einem einfacheren Training durch den k-Means-Algorithmus, subjektiv bessere und robustere Segmentierungsergebnisse erzielt wurden.

In dieser Arbeit wird nur Farbinformation zur Segmentierung benutzt und die Ergebnisse sind erstaunlich gut. Es wäre sicher auch interessant, zusätzlich andere Merkmale wie beispielsweise die Position (x,y) der Pixel oder auch Textur für die Bildmodellierung und Segmentierung zu benutzen. Dies könnte mit dem implementierten Verfahren ohne weiteres untersucht werden, da dafür nur die Merkmalsvektoren um entsprechend neue Features erweitert werden müssen.

Da die hier verwendete farbbasierte Segmentierung keine Ortsinformation berücksichtigt, und jedes Pixel unabhängig von seinen Nachbarpixeln klassifiziert wird, könnten noch sogenannte morphologische Operatoren (um Löcher in den Regionen zu schließen) dazu beitragen, die Segmentierungsergebnisse zu verbessern beziehungsweise zu glätten.

Allgemein wäre der Übergang Körperregionen - Körperteile wünschenswert. Im Endeffekt interessiert man sich hauptsächlich für die Position des Kopfes, des Rumpfes, der Hände und der Arme. Um die gefundenen Bildregionen den Körperregionen einer Person zuzuordnen, könnte beispielsweise die Größe und die Lage der einzelnen Regionen benutzt werden. Außerdem kann man sich bestimmte „Priors“ wie Hautfarbe für die Körpersegmente zunutze machen.
Natürlich unterstützen dabei sowohl eine separate Konturanalyse (wie beim „Pfinder“-System) als auch explizite Körpermodelle die Identifizierung von Körperteilen.

Anhang A

Allgemeine Modellierungsansätze

In diesem Anhangskapitel soll Lesern, die mit statistischer Modellierung nicht so vertraut sind, ein Überblick über allgemeine Modellierungsansätze gegeben werden, damit die Begriffe und mathematischen Formeln, die in dieser Ausarbeitung gelegentlich verwendet werden, verständlich werden.
Die Ausführungen entstammen hauptsächlich aus [4].

Die Aufgabe der statistischen Modellierung besteht darin, die Dichte(funktion) p(x) einer Wahrscheinlichkeitsverteilung mit Hilfe einer endlichen Anzahl von Datenpunkten x^n (n=1,...,N), die zu dieser Dichtefunktion gehören, zu bestimmen. Dabei sollte noch erwähnt werden, dass die genaue Modellierung der Wahrscheinlichkeitsdichte bei höher dimensionalen Datenräumen (zum Beispiel ab d=10) extrem schwierig ist.

Im Prinzip gibt es 3 alternative Techniken, um die Dichte zu schätzen:

- **Parametrische Methoden**: Hier wird eine ganz spezielle Dichtefunktion angenommen beziehungsweise man geht von einer bekannten Verteilungsform aus. Dabei gibt es Parameter, die durch das Anpassen des Modells an die Datenmenge optimiert werden. Der Nachteil besteht darin, dass durch diese ganz spezielle parametrische Funktionsform möglicherweise nicht die „wahre" Dichte wiedergegeben werden kann.
- **Nichtparametrische Schätzung**: Im Gegensatz zur ersten Methode wird hier keine bestimmte Funktionsform angenommen. Stattdessen bestimmt sich die Dichtefunktion alleine durch die Daten selbst. Dies hat den Nachteil, dass die Modelle sehr schnell „unhandlich" werden (das heißt die Anzahl der Parameter im Modell nimmt mit der Größe der Datenmenge zu).
- **Semiparametrische Schätzung**: Hier wird versucht, die Vorteile der ersten zwei Techniken zu kombinieren. Einerseits wird eine sehr allgemeine Klasse von Funktionsformen erlaubt (in der die Anzahl der adaptiven Parameter auf systematische Weise erhöht werden kann, um flexiblere Modelle zu erhalten) und andererseits kann die Gesamtzahl von Modellparametern unabhängig von der Größe der Datenmenge variiert werden. Bekanntester Vertreter ist die Mixturverteilung.

A.1 Parametrische Methoden (Gauß-/Normalverteilung)

Das einfachste und auch das am meisten genutzte parametrische Modell ist die (allgemeine) Gauß- beziehungsweise Normalverteilung, die einige praktische analytische und statistische Eigenschaften hat. Wir beschränken uns auf diese Verteilung, um die Grundprinzipien der parametrischen Dichteschätzung zu erklären.

Für den eindimensionalen Fall sieht die Dichte der Normalverteilung folgendermaßen aus:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

wobei μ der Mittelwert (Erwartungswert) und σ^2 die Varianz (Streuung) ist (der Parameter σ alleine heißt Standardabweichung).
Der Mittelwert und die Varianz der eindimensionalen Normalverteilung lassen sich mit Hilfe der Erwartungswertfunktion $E[\cdot]$ beschreiben:

$$\mu = E[x] = \int_{-\infty}^{\infty} x p(x) dx, \qquad \sigma^2 = E\left[(x-\mu)^2\right] = \int_{-\infty}^{\infty} (x-\mu)^2 p(x) dx$$

Die allgemeine multivariate Dichte der Normalverteilung für d Dimensionen ist:

$$p(x) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} e^{-\frac{(x-\mu)^T \bullet (x-\mu)}{2\Sigma}}$$

wobei der Mittelwert μ nun ein d-dimensionaler Vektor, Σ eine d x d Kovarianzmatrix und $|\Sigma|$ die Determinante von Σ ist.
Die Parameter μ und Σ lassen sich wieder mit der Erwartungswertfunktion $E[\cdot]$ ausdrücken:

$$\mu = E[x], \qquad \Sigma = E\left[(x-\mu)(x-\mu)^T\right]$$

Σ ist eine symmetrische Matrix und hat also allgemein $d(d+1)/2$ unabhängige Komponenten.
Der Term, der im Exponent der allgemeinen multivariaten Gauß-Verteilung erscheint, wird als sogenannte „(quadrierte) Mahalanobis Distanz" vom Datenpunkt x zum Mittelwert μ bezeichnet:

$$\Delta^2 = (x-\mu)^T \Sigma^{-1} (x-\mu)$$

Datenpunkte mit gleichem Δ^2 bilden Hyperellipsoide (im zweidimensionalen Fall sind dies Ellipsen). Die Hauptachsen dieser Hyperellipsoide werden durch die Eigenvektoren u_i der Kovarianzmatrix Σ definiert (und die zugehörigen Eigenwerte λ_i geben die Varianzen entlang der entsprechenden Hauptachsenrichtungen an):

$$\Sigma u_i = \lambda_i u_i$$

Bei jeder nichtsingulären linearen Transformation des Koordinatensystems behält die Mahalanobis Distanz ihre quadratische Form bei und sie bleibt positiv definit (siehe [4]).

Häufig wird eine vereinfachte Form der Gauß-Verteilung verwendet, wobei dann die Kovarianzmatrix eine Diagonalgestalt hat und nur noch d Parameter besitzt (dies erhält man durch eine lineare Transformation in ein Koordinatensystem, das auf den Eigenvektoren von Σ basiert):

$$(\Sigma)_{ij} = \delta_{ij}\sigma_i^2, \quad \delta_{ij} = \begin{cases} 1, i = j \\ 0, sonst \end{cases}$$

$$p(x) = \frac{1}{\sqrt{(2\pi)^d \prod_{i=1}^{d} \sigma_i^2}} e^{-\frac{1}{2}\sum_{i=1}^{d} \frac{(x_i-\mu_i)^2}{\sigma_i^2}}, \quad p(x) = \prod_{i=1}^{d} p(x_i)$$

Man sagt, die Komponenten von x beziehungsweise die Variablen der Kovarianzmatrix sind statistisch/stochastisch unabhängig.

Eine weitere Vereinfachung erhält man, wenn man $\sigma_i = \sigma$ für alle i setzt (das heißt $\Sigma = \sigma^2 I$, wobei I die Einheitsmatrix ist), es gibt also nur noch einen einzigen Varianzparameter. Natürlich werden durch diese Vereinfachungen die Modelle nicht mehr so flexibel beziehungsweise es kann nicht mehr so gut generalisiert werden.

Um Werte für die „offenen" Parameter mit Hilfe der vorhandenen Datenmenge zu bestimmen, gibt es zwei (vom Ansatz her verschiedene) Standardverfahren beziehungsweise -techniken: zum einen Maximum-Likelihood-Schätzung und zum anderen (überwachtes) Lernen nach Bayes. Für eine genaue Erläuterung der Verfahren siehe [4].

A.2 Nichtparametrische Schätzung (Histogramme, Parzen-Fenster, k-nächster-Nachbar)

Bei den parametrischen Methoden wird davon ausgegangen, dass die zugrundeliegenden Wahrscheinlichkeitsdichten zumindest ihrer Form nach bekannt sind und zu ihrer Beschreibung „nur" eine Reihe von Parametern bestimmt werden müssen. Leider ist es aber häufig so, dass die in der Praxis auftretenden Dichtefunktionen nicht bekannten parametrisierten Verteilungsformen entsprechen.
Zu den bekanntesten nichtparametrischen Techniken, die die Form der zugrundeliegenden Verteilungen nicht als bekannt voraussetzen und die dafür die Daten selbst benutzen, zählen Histogramm-Methoden, Verfahren, die auf sogenannten Fensterfunktionen (im Englischen „kernel functions") basieren (Parzen-Fenster-Schätzung) und die k-nächster-Nachbar-Schätzung.
Die Grundidee all dieser Verfahren besteht in der Schätzung einer unbekannten Verteilungsfunktion. Dabei spielen die sogenannten „smoothing" Parameter, die bestimmen, wie glatt die geschätzte Dichtefunktion verläuft, stets eine wichtige Rolle.
Für eine detaillierte Erklärung des nichtparametrischen Ansatzes und der verschiedenen Ausprägungen siehe [4].

A.3 Semiparametrische Schätzung (Mixtur Modelle)

Sowohl der parametrische als auch der nichtparametrische Ansatz hat Vor- und Nachteile. Die parametrische Schätzung geht von einer ganz speziellen Form für die Dichtefunktion aus, die aber unter Umständen sehr von der „wahren" Wahrscheinlichkeitsdichte abweichen kann. Jedoch ist es bei parametrischen Modellen normalerweise möglich, die Dichtefunktion sehr schnell für neue Werte des Eingabevektors zu schätzen.
Dagegen erlauben nichtparametrische Methoden sehr allgemeine Funktionsformen für die Dichte. Sie haben aber den Nachteil, dass die Anzahl der Variablen im Modell direkt mit der Anzahl der Trainingsdatenpunkte wächst. Dies führt dann zu Modellen, die nur sehr langsam neue Eingabevektoren auswerten können.

Um die Vorteile beider Methoden zu vereinigen wünscht man sich Techniken, die nicht auf spezielle Funktionsformen beschränkt sind und bei denen die Größe des Modells nur mit der Komplexität des zu lösenden Problems und nicht mit der Datenmenge wächst. Dies führt dann zur Klasse der sogenannten semiparametrischen Modelle.

Den Preis, den man dafür zahlen muss, ist der viel größere Rechenaufwand für das Aufstellen der Modelle mit Hilfe der Datenmenge (das Training) im Vergleich zu den einfachen Prozeduren, die man für parametrische oder nichtparametrische Methoden braucht.
Bekanntester Vertreter der semiparametrischen Modelle ist das Mixturmodell beziehungsweise eine Mixturverteilung.
Das Modell für die Wahrscheinlichkeitsdichte ist eine Linearkombination von Basisfunktionen (sogenannten Komponentendichten) $p(x|j)$:

$$p(x) = \sum_{j=1}^{M} p(x|j)P(j)$$

wobei die Anzahl M der Basisfunktionen ein Parameter darstellt und viel kleiner als die Anzahl N der Datenpunkte ist. Die Koeffizienten $P(j)$ werden Mixturparameter (mixing parameter) oder auch Gewichte genannt und sie spiegeln die a priori Wahrscheinlichkeiten wider, dass der Datenpunkt x zur Komponente j der Mixtur „gehört". Die Wahrscheinlichkeit p(x) gibt also an, ob x überhaupt beobachtet wurde.
Folgende Bedingungen sollten erfüllt sein, damit p(x) wirklich eine Zufallsverteilung ist:

$$\sum_{j=1}^{M} P(j) = 1, \qquad 0 \le P(j) \le 1$$

$$\int p(x|j)dx = 1$$

Die Komponentenwahrscheinlichkeiten $p(x|j)$ kann man so auch als klassenbedingte Wahrscheinlichkeiten ansehen. Der Hauptunterschied zu einem „echten" Klassifikationsproblem liegt aber in der Art der Trainingsdaten. Man hat bei Mixturen keine Klassenlabels (also genaue Zuweisungen zu bestimmten Klassen) und das ist ein Beispiel für sogenannte unvollständige Daten.

Die a posteriori Wahrscheinlichkeit, dass ein beobachtetes x zu einer Komponente j „gehört" ist nach der Formel von Bayes:

$$P(j|x) = \frac{p(x|j)P(j)}{p(x)} \quad \text{und es gilt:} \quad \sum_{j=1}^{M} P(j|x) = 1$$

Häufig kommen Gauß-Dichtefunktionen als Basisfunktionen zum Einsatz und man spricht dann von einer Gauß-Mixtur beziehungsweise Mixtur von Gauß-Verteilungen.
Es ist wichtig anzumerken, dass es für die unterschiedlichsten Komponentendichtefunktionen möglich ist, jede kontinuierliche Dichte mit beliebiger Genauigkeit anzunähern, vorausgesetzt das Mixturmodell besitzt genügend viele Komponenten und die Parameter des Modells sind richtig gewählt.

Für das Training der Mixturen kommt wie bei den parametrischen Methoden die Maximum-Likelihood-Schätzung zum Einsatz. Die „normale" nichtlineare Optimierung ist jedoch nicht praxistauglich beziehungsweise zu rechenintensiv, stattdessen verwendet man häufig den sogenannten „Expectation Maximization"(EM)-Algorithmus. Für genaue Erläuterungen siehe [4].

Anhang B

Farbräume

B.1 Verschiedene Farbräume und deren Charakteristika

Um die verschiedenen Farben auszudrücken, wurden unterschiedliche Farbmodelle beziehungsweise Farbräume definiert. Dabei wird jede Farbe durch einen Punkt in einem mehrdimensionalen Raum eindeutig festgelegt. Bei den verschiedenen Farbmodellen haben diese Räume andere Achsen. Um von einer Darstellung in die andere zu kommen muss jeder Raumpunkt in einen neuen überführt werden. Die Transformation der hier verwendeten Farbräume aus dem Standardfarbraum RGB wird im Anhang B.2 beschrieben.

Kurz noch eine Erklärung der wichtigsten Begriffe:

- Farbton (englisch hue)
  Im physikalischen Sinne bezieht sich der Farbton auf die dominante Wellenlänge im Spektrum (400nm-700nm). Wenn eine Farbe als Spektralverteilung dargestellt wird, dann empfindet man den Farbton im Bereich des Extremwertes des Spektrums am stärksten.

- Helligkeit (Luminanz, englisch lightness/brightness/intensity)
  Bei gleichbleibendem Farbton kann eine Farbe dunkler oder heller erscheinen. Die Wahrnehmung der Helligkeit ist in den meisten Fällen unabhängig von der Wahrnehmung eines Farbtons, obwohl es Ausnahmen gibt (zum Beispiel erscheint Gelb immer etwas heller als Blau). Im physikalischen Sinne bezieht sich die Helligkeit auf die Menge des Lichtes, das im Auge eintrifft, also das Integral des Spektrums (das heißt bei zusätzlich eingeschalteter Lampe wirkt eine Farbe heller).

- Sättigung (Reinheit der Farbe, englisch saturation)
  Diese Eigenschaft gibt an, wie verwaschen eine Farbe aussieht. Bei hoher Sättigung sind die Farben pur, rein, kräftig. Bei niedriger Sättigung sind sie pastellfarben (zum Beispiel Kleidung nach häufigem Waschen). Physikalisch bezieht sich die Sättigung auf die Breite des Wellenlängenspektrums. Ist nur ein Impuls vorhanden, ist die Farbe gesättigt, ist eine Gleichverteilung vorhanden, ist die Farbe völlig ungesättigt (Graustufe).

Farbton und Sättigung beinhalten Farbinformation, Helligkeit nicht! Übrigens, ist die Tatsache, dass helle Regionen im Bild mehr streuen als vergleichsweise dunkle Regionen, ein bekanntes Phänomen.

Im folgenden werden die einzelnen hier verwendeten Farbräume genauer beschrieben.
Dabei macht man die Unterscheidung zwischen „normalen" Farbräumen und perzeptuellen Farbräumen. Die erste Gruppe beruht auf kartesischen Koordinaten und die zweite Gruppe basiert auf zylindrischen Koordinaten.

B.1.1 Auf kartesischen Koordinaten basierende Farbräume

B.1.1.1 RGB

Im RGB (red, green, blue)-Farbraum setzt sich jedes sichtbare Pixel aus drei Komponenten R(ot), G(rün) und B(lau) zusammen. Das Zahlentrippel kodiert also, wie groß der Anteil der Grundfarben ist.
Will man eine naturgetreue Farbwiedergabe am Computer erreichen, so muß jede dieser Komponenten mindestens 256 Ausprägungen haben.
Der Wertebereich der Farbanteile bewegt sich somit zwischen 0 und 255 und ist diskret.
Dabei entspricht (0,0,0) Schwarz, (255,255,255) Weiß und die Grauwerte befinden sich auf der Raumdiagonalen zwischen Ursprung und weißer Ecke (Abb. B.1).
Man benötigt damit genau ein Byte Speicherplatz pro Farbkomponente.
Ein einziges vollständiges Videobild erfordert eine Speichermenge von 768 Pixel x 576 Pixel x 3 Byte, also 1327104 Byte, das heißt ungefähr 1,2 MB pro Bild. Eine 2 Gigabyte Festplatte hätte somit (ohne Transformation in andere Farbräume (wie zum Beispiel YUV) oder Komprimierung) nur eine Videokapazität von knapp einer Minute.

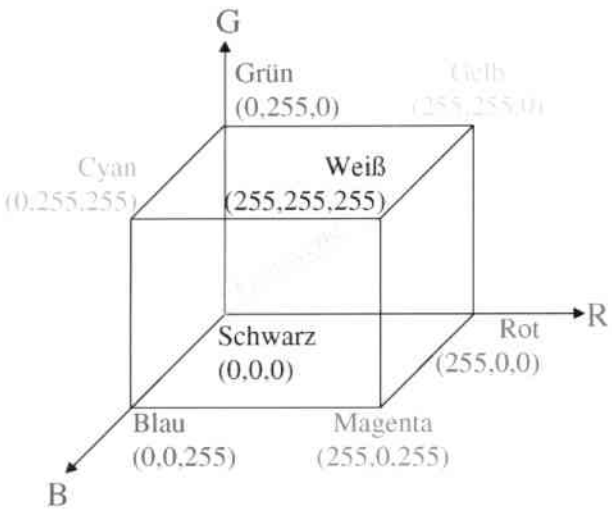

Abbildung B.1: RGB-Farbwürfel

Den RGB-Farbraum verwendet man für die additive Farbmischung (zum Beispiel Bildschirmdarstellung) und den sogenannten CMY(Cyan, Magenta, Gelb (Yellow))-Farbraum benutzt man für die subtraktive Farbmischung (zum Beispiel Druckerausgabe).
Der Unterschied zwischen dem RGB- und dem dazu „inversen" CMY-Farbraum ist der Ursprungspunkt: bei RGB ist er Schwarz und bei CMY ist er Weiß.

B.1.1.2 NRGB

Der NRGB-Farbraum ist im Prinzip genau gleich aufgebaut wie der RGB-Farbraum, nur dass die Farbanteile jeweils durch 256 geteilt werden. Damit erhält man eine Normalisierung (der Wertebereich liegt nun zwischen 0 und 1).
Die Komponenten des (normalisierten) CMY-Farbraumes erhält man dann folgendermaßen:
$(C, M, Y)^T = (1,1,1)^T - (R, G, B)^T$

B.1.1.3 NRG

Beim NRG-Farbraum macht man sich eine spezielle Normalisierung zu Nutze. Man teilt die einzelnen Farbanteile durch die Summe aller Farbanteile (das Trippel (R,G,B) repräsentiert nicht nur Farbe, sondern auch Helligkeit). Damit wird nun eine Farbkomponente redundant (hier die Blaukomponente), weil sie sich automatisch durch die Formel b=1-r-g ergibt. Dieser 2-dimensionaler Farbraum hat den großen Vorteil, daß das Farbmodell auf Beleuchtungseffekte nicht mehr so stark reagiert: es liegt eine Helligkeitsnormalisierung vor. Natürlich muss man bei der Rücktransformation in den Standard RGB-Farbraum vorsichtig sein: die Abbildung ist nämlich nicht eineindeutig.

B.1.1.4 YUV

Die in Europa gängige Fernsehnorm PAL (Phase Alternating Line) benutzt den YUV-Farbraum. Dieser Farbraum verwendet im Gegensatz zum RGB-Farbraum eine Helligkeitskomponente Y, das Schwarzweißsignal (Luminanz genannt) und die 2 Farbkomponenten U und V, das Farbsignal (Chrominanz genannt) (Abb. B.2, rechts).
Die Darstellung in diesem Farbraum hat einen Vorteil, der sich bei der Komprimierung und Übertragung von Videoinformation gewinnbringend einsetzen läßt. Das menschliche Auge ist gegenüber Helligkeitsinformation (Y) viel sensibler als gegenüber Farbinformation (U,V), das heißt der größte Teil der Information wird in der Luminanzkomponente des Bildes gespeichert. Dadurch fällt es nicht auf, wenn nur die halbe Farbinformation (4:2:2 Abtastung) oder sogar noch weniger (4:2:0 Abtastung) zur vollen Luminanzinformation übertragen wird. So lässt sich in einem ersten Schritt die Datenmenge einer Videoübertragung ohne sichtbare Qualitätsverluste um gut ein Drittel reduzieren.

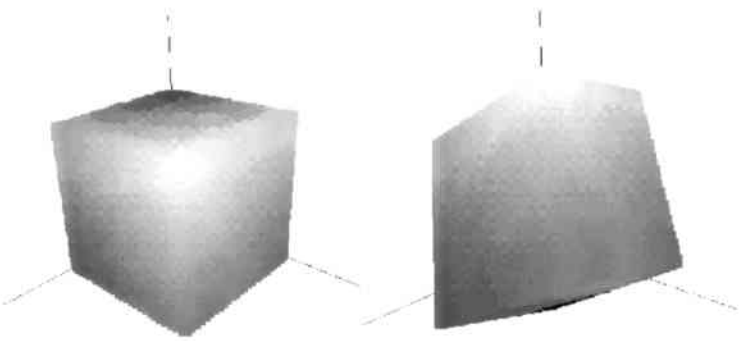

Abbildung B.2: (links) RGB-Farbwürfel im RGB-Farbraum (die vertikale Achse ist der Grünanteil), (rechts) Transformation des gleichen Würfels in den YUV-Raum (die vertikale Achse ist die Helligkeit Y)

B.1.1.5 U*V*

Wenn man in Abbildung B.2 (rechts) die Y-Koordinate durch Projektion in die UV-Ebene überflüssig macht, kann man Entscheidungen, die nur noch auf Farbe (Chromatik) beruhen, treffen.
Der U*V*-Farbraum ist genau so definiert, nämlich ein helligkeitsnormalisierter 2-dimensionaler Farbraum. Es gilt: U*=U/Y und V*=V/Y.

B.1.1.6 YIQ

In der amerikanische Fernsehnorm NTSC (National Television System Committee) wurde dieser Farbraum festgelegt: das US Fernsehen ist so ausgelegt, dass es die gleiche Empfangskodierung für S/W und Farbsehen verwenden kann (1 achromatischer Kanal und 2 chromatische Kanäle).
Der YIQ-Farbraum ist prinzipiell mit dem YUV-Farbraum identisch (er verwendet nur die 2 anderen Farbkomponenten I (Cyan-Orange Balance) und Q (Magenta-Grün Balance).

B.1.1.7 TS

Der TS (tint, saturation)-Farbraum setzt auf den helligkeitsnormalisierten NRG-Raum auf (siehe Anhang B.2). Die T-Komponente steht für den Farbton (Rot, Gelb, Grün, Blau, ...) und die S-Komponente gibt die Sättigung wieder, wobei die T- und S-Werte zwischen 0 und 1 normalisiert sind. Um eine gute Klassifizierung im Farbraum zu erhalten, werden beispielsweise in [14] Farbhistogramme im TS-Raum verwendet.

B.1.2 Auf zylindrischen Koordinaten basierende sog. perzeptuelle Farbräume

Solche psychovisuell gut angepassten Farbmodelle werden durch Farbton, Helligkeit und Sättigung beschrieben. Sie ermöglichen eine Farbauswahl, die von uns als natürlicher empfunden wird. Wie bereits angesprochen, enthalten Farbton und Sättigung im Gegensatz zur Helligkeit Farbinformation.
Um eine Farbpalette nach dieser Auswahl am Computer darstellen zu können, muss man ein dazu passendes geometrisches Modell finden. Perzeptuelle Farbräume unterscheiden sich durch die Wahl des Koordinatensystems von den vorher besprochenen: es werden zylindrische Koordinaten anstatt der einfachen kartesischen Koordinaten gewählt.
Verschiedene Farbräume werden von diesem intuitiven Farbmodellen abgeleitet.
Die bekanntesten sind HSV und HLS.

B.1.2.1 HSV

Der HSV (hue, saturation, value)- beziehungsweise HSB (hue, saturation, brightness)–Farbraum wird als Kegel (Abb. B.4) oder vereinfacht als sechskantige Pyramide (Abb. B.3) dargestellt, mit der Spitze bei Schwarz.
Die Skala für den Farbton H (hue) ist zyklisch: er wird durch den Winkel zwischen 0 und 360 Grad um die vertikale Achse beschrieben: per Konvention liegt Rot bei 0°, Gelb bei 60°, Grün bei 120°, Cyan bei 180°, Blau bei 240° und Magenta bei 300° (Komplementärfarben liegen also einander wie beim Farbkreis um 180 Grad gegenüber).
Die Sättigung S (saturation) ist der horizontale Anteil von der Mittelachse bis zum Randpunkt und nimmt Werte zwischen 0 und 1 an: 0 (Mittelachse) repräsentiert eine Graustufe, 1 (Rand) steht hingegen für eine voll gesättigte Farbe.
Die Grundhelligkeit V beziehungsweise B (value, brightness) ist der vertikale Anteil entlang der Mittelachse und läuft von Schwarz (V=0) bis zu den Farben mit maximaler Helligkeit, inklusive Weiß (V=1).
Man kann sich dieses Farbmodell im Prinzip so vorstellen. Zuerst kommt die Auswahl des Farbtons (Orange, Rot, Grün, ...). Dann folgt das Aufhellen oder Nachdunkeln (Weiß oder Schwarz) des Farbtons: Aufhellen durch Dazumischen von Weiß (verringere Sättigung) und Abdunkeln der Farbe durch Hinzumischen von Schwarz (verringere Helligkeit).

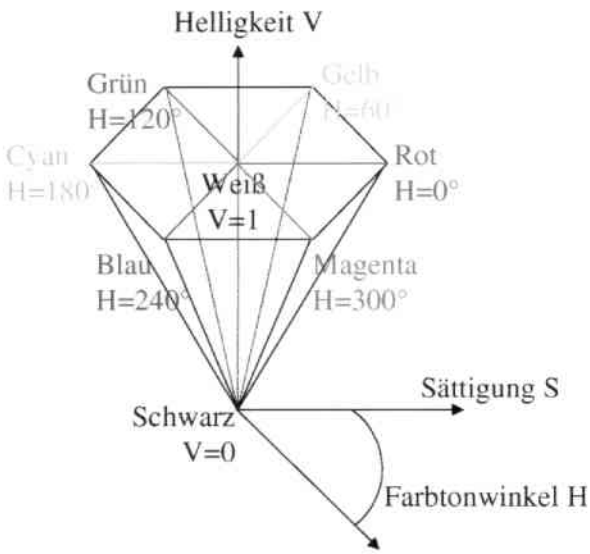

Abbildung B.3: HSV-Farbraum als Pyramide

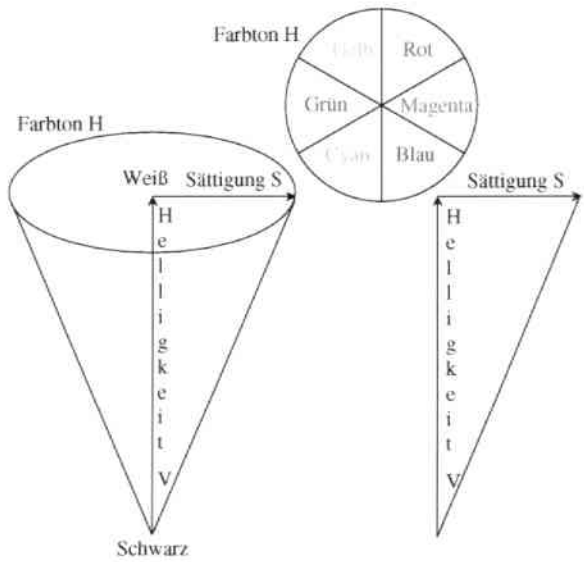

Abbildung B.4: HSV-Farbraum als Kegel (mit Querschnitten)

B.1.2.2 HLS

Der HLS (hue, lightness, saturation)- beziehungsweise HSI (hue, saturation, intensity) – Farbraum wird als Doppelkegel (Abb. B.5) oder vereinfacht als sechskantige Doppelpyramide dargestellt.
Man kann sich die Entstehung so vorstellen, indem man den Mittelpunkt der HSV Grundfläche „hochzieht", so dass sich sowohl V=0 als auch V=1 in einer Spitze (sich gegenüber) befinden. Das heißt Weiß und Schwarz sind in beiden Fällen nur ein Punkt.

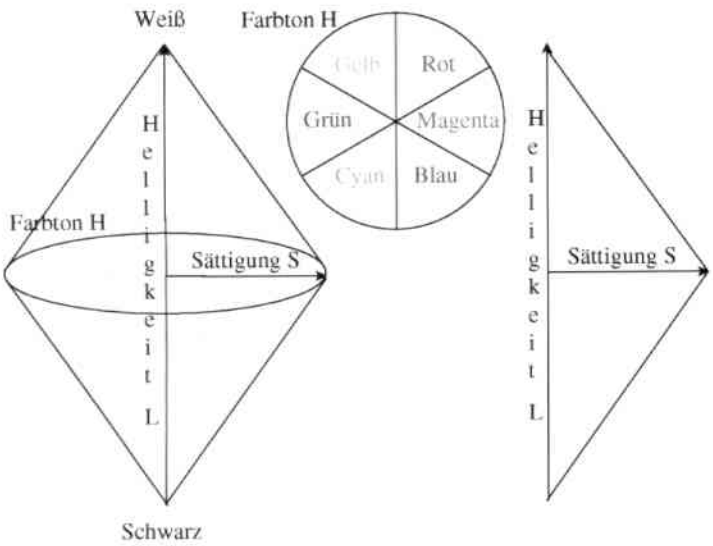

Abbildung B.5: HLS-Farbraum als Doppelkegel (mit Querschnitten)

B.2 Transformationsvorschriften der Farbräume

Die nachfolgenden Transformationen sind algorithmisch von oben nach unten zu lesen. Es ist zu beachten, dass die Transformationsmatrizen unterschiedlich definiert sein können, deshalb gibt es beim YUV- und analog beim U*V*-Farbraum zwei Varianten.

- RGB -> NRGB (rgb)

$$(r,g,b)^T = (R/256, G/256, B/256)^T$$

- RGB -> NRG (rg)

falls $(R+G+B)=0$:

$$(r,g)^T = (0,0)^T$$

$$(r,g)^T = (R/(R+G+B), G/(R+G+B))^T$$

- RGB -> YUV (1.Variante)

$$(Y,U,V)^T = \begin{pmatrix} 0.299 & 0.587 & 0.144 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{pmatrix} (R,G,B)^T$$

- RGB -> YUV (2.Variante)

$$(Y,U,V)^T = \begin{pmatrix} 0.2990 & 0.5870 & 0.1140 \\ -0.1687 & -0.3313 & 0.5000 \\ 0.5000 & -0.4187 & -0.0813 \end{pmatrix} (R,G,B)^T$$

- RGB -> U*V* (1.Variante)

$$(Y,U,V)^T = \begin{pmatrix} 0.299 & 0.587 & 0.144 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{pmatrix} (R,G,B)^T$$

falls $Y = 0$:

$$(U^*,V^*)^T = (0,0)^T$$

$$(U^*,V^*)^T = (U / Y, V / Y)^T$$

- RGB -> U*V* (2.Variante)

$$(Y,U,V)^T = \begin{pmatrix} 0.2990 & 0.5870 & 0.1140 \\ -0.1687 & -0.3313 & 0.5000 \\ 0.5000 & -0.4187 & -0.0813 \end{pmatrix} (R,G,B)^T$$

falls $Y = 0$:

$$(U^*,V^*)^T = (0,0)^T$$

$$(U^*,V^*)^T = (U / Y, V / Y)^T$$

- RGB -> YIQ

$$(Y,I,Q)^T = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.212 & -0.523 & 0.311 \end{pmatrix} (R,G,B)^T$$

- RGB -> TS

falls $(R+G+B)=0$:

$$(r,g)^T = (0,0)^T$$

$$(r,g)^T = \left(R/(R+G+B), G/(R+G+B)\right)^T$$

$$d = \sqrt{(r-1/3)^2 + (g-1/3)^2}$$

$$T = \begin{cases} 0, & r = 1/3, g \le 1/3 \\ 1/2, & r = 1/3, g > 1/3 \\ 3/4, & g = 1/3, r < 1/3 \\ 1/4, & g = 1/3, r > 1/3 \\ \frac{1}{2\pi}\arctan\left(\frac{g-1/3}{r-1/3}\right) + 1/4, & r > 1/3, g \neq 1/3 \\ \frac{1}{2\pi}\arctan\left(\frac{g-1/3}{r-1/3}\right) + 3/4, & r < 1/3, g \neq 1/3 \end{cases}$$

$$S = \begin{cases} 3d, & T = 1/4 \\ \frac{3d\left|\tan(T)-1\right|}{\sqrt{1+\tan^2(T)}}, & 0.1762 \le T < 0.5738 \\ 3d\cos(T-3/4), & 0.5738 \le T \le 0.875 \\ 3d\cos T, & sonst, \end{cases}$$

- RGB -> HSV

$$v = \max(R,G,B)$$
$$\Delta = v - \min(R,G,B)$$

$$s = \begin{cases} 0, & v = 0 \\ \frac{\Delta}{v}, & v \neq 0 \end{cases}$$

$$h = \begin{cases} 0, & s = 0 \\ \dfrac{G-B}{\Delta}, & v = R, \\ \dfrac{B-R}{\Delta} + 2, & v = G, \\ \dfrac{R-G}{\Delta} + 4, & v = B, \end{cases}$$

$h = h * 60$

Falls $h < 0$:
$h = h + 360$

Die kartesischen Koordinaten ergeben sich dann wie folgt aus den zylindrischen:
$H = sv\cos(2\pi h)$
$S = sv\sin(2\pi h)$
$V = v$

Anhang C

Verweise auf Internetseiten

Die folgenden Webseiten geben begleitende Informationen und veranschaulichen die in dieser Studienarbeit behandelten Themen. Natürlich kann in der schnelllebigen Internetwelt nicht gewährleistet werden, dass die Links dauerhaft gültig sind.

- „Pfinder"-Links [13]:
  http://pfinder.www.media.mit.edu/projects/pfinder/
  gibt einen Überblick über den Pfinder und enthält im Untermenü „SmartRooms" auch ein Demovideo („Interacting with virtual creatures").
  http://vismod.www.media.mit.edu/cgi-bin/tr_pagemaker?search=Pfinder
  ist sozusagen die Einstiegsseite: man erhält sowohl Links zur PS/HTML-Version des Pfinder TR-353 als auch unter „Abstract" PS/HTML-Versionen der ausführlicheren Thesis. Außerdem ist es möglich, hier eine „hardcopy" zu bestellen.
  http://vismod.www.media.mit.edu/tech-reports/TR-353/tr.html
  ist die HTML-Version des Pfinder Technical Reports No. 353 (siehe oben).
  http://vismod.www.media.mit.edu/vismod/demos/pfinder/ms/ms.html
  ist die HTML-Version der Pfinder M.S. Thesis von C.R. Wren (September 1996).

  http://www.tillmann-group.de/khm/kap2.htm
  ist das 2.Kapitel „Segmentierung und Tracking" der Diplomarbeit „Ein modellbasiertes Konzept zur regionenbasierten 3D-Bewegungsverfolgung" von Harald Tillmann (Informatik, Universität Dortmund).

- „Headtracker"-Links [3]:
  http://robotics.stanford.edu/~birch/headtracker/
  zeigt MPEG Videoclips und BMP Bildsequenzen zur Demonstration des „Headtrackers". Außerdem befinden sich hier der C/C++ Quellcode und Publikationen in PDF beziehungsweise PS.

- Links zum Farbsegmentierungsbeispiel von Comaniciu/Meer [5]:
  http://www.caip.rutgers.edu/riul/
  ist die Heimatseite des RIUL (Robust Image Understanding Laboratory).
  Unter „Research" gelangt man unter anderem auf Veröffentlichungen und Quellcode.
  http://www.caip.rutgers.edu/riul/research/robust.html
  erscheint beim Menüpunkt „Research: Robust Analysis" und enthält interessante Informationen.

- nützliche Links zur Normalverteilung:
  http://www.uni-konstanz.de/FuF/wiwi/heiler/os/vt-norm.html
  http://www.uni-konstanz.de/FuF/wiwi/heiler/os/test
  sind Applets, die die eindimensionale und bivariate Normalverteilung veranschaulichen.

- nützliche Links zum EM-Algorithmus:
  http://www.neurosci.aist.go.jp/~akaho/MixtureEM.html
  ist ein Applet, das zeigt, wie der EM-Algorithmus funktioniert.

  http://www-edlab.cs.umass.edu/cs689/em.html
  ist ein C-Programm, das 2 Mittelwerte durch den EM findet (EM for k-Means).

  http://www.cs.cmu.edu/~15781/web/notes/em.ppt
  http://www.cs.cmu.edu/~15781/web/notes/em.ps.gz
  Diese Links entstammen aus der „Machine Learning"-Vorlesung Herbst 2000 von Sebastian Thrun (Carnegie Mellon University) und die Foliensammlung hat den Titel „The EM Algorithm (or: More than you ever wanted to know about EM)".

  http://www.sgi.com/tech/mlc/
  MLC++ ist eine Bibliothek von C++ Klassen für überwachtes maschinelles Lernen.

- nützliche Links zu Farbräumen:
  http://www.wias-berlin.de/~telschow/dvg3/12Color.html
  zeigt unter anderem einen Algorithmus zur Konvertierung aus dem RGB- in den HSV-Farbraum (von Gerhard Telschow).

  http://www.uni-koblenz.de/fb4/publikationen/dissertationen/diss_Rehrmann.html
  Diese Dissertation aus dem Jahr 1994 von Volker Rehrmann (Informatik, Universität Koblenz-Landau) untersucht unter anderem Farbähnlichkeitsmaße in den Farbräumen RGB, YIQ, Lab, Luv und HSV.

  http://www.uni-bayreuth.de/lehre/dibito/vorlesung/node20.html
  ist das Kapitel „Ablage von Farbinformationen" (RGB-, CMY- und HSV-Farbraum) aus „Grundlagen der digitalen Bild- und Tonverarbeitung" (B.L. Winkler, Rechenzentrum Universität Bayreuth).

  http://www.uni-paderborn.de/fachbereich/AG/agdomik/extern/computergrafik/cg_skript/html/node133.html
  ist das Kapitel „Farbe" aus dem Skriptum zur Vorlesung „Computergraphik" 1994/1997 (Prof. Dr. Gitta Domik, Informatik, Uni-Gesamthochschule Paderborn) und enthält die Einzelthemen: psychophysikalische Grundlagen, Darstellung von Farben am Computer, RGB Farbmodell, C-M-Y Modell, YIQ System, YUV Farbraum und schließlich Perzeptuelle Farbräume (HSV, HLS).

Anhang D

Nützliches zu C++

Hier sind ein paar Angaben zu C++ - Literatur und - Links:

- Marshall P. Cline, Greg A. Lomow, and Mike Girou:
  C++ FAQ's (Frequently Asked Questions) (2nd edition)
  Addison-Wesley (Dezember 1998)
  ISBN 0-201-30983-1
  „gives practical answers to many questions that occur to programmers starting to use C++" (B.Stroustrup)

- Andrew Koenig and Barbara E. Moo:
  Accelerated C++: Practical Programming by Example (C++ In-Depth Series)
  (1st edition)
  Addison-Wesley (August 2000)
  ISBN 0-201-70353-X

- Stanley B. Lippman and Josee Lajoie:
  C++ Primer (3rd edition)
  Addison-Wesley (April 1998)
  ISBN 0-201-82470-1

- Bjarne Stroustrup:
  An Overview of the C++ Programming Language
  From „The Handbook of Object Technology" (Saba Zamir), CRC Press LLC (1999)
  ISBN 0-8493-3135-8

  http://www.research.att.com/~bs/
  ist die Heimatseite von Bjarne Stroustrup.

- Bjarne Stroustrup:
  The C++ Programming Language (3rd edition, hardcover)
  Addison-Wesley (1997)
  ISBN 0-201-8895-4

- **http://www.accu.org/**
  ACCU steht für „Association of C & C++ Users" und ist eine gemeinnützige Organisation für Leute, die sich mit C, C++ und Java beschäftigen. Dort gibt es Buchkritiken, nützliche Links und Termine für Veranstaltungen/Konferenzen.

- **http://anubis.dkuug.dk/jtc1/sc22/wg21/**
  WG21 ist die internationale Standardisierungsgruppe für die Programmiersprache C++ und ist für die ISO/IEC 14882 (aus dem Jahr 1998) und die ISO/IEC 1805 (C++ Perfomance) verantwortlich.

Literatur- und Quellenverzeichnis

[1] Serge Belongie, Chad Carson, Hayit Greenspan, and Jitendra Malik:
Color- and Texture-Based Image Segmentation Using EM and Its Application to Content-Based Image Retrieval
Computer Science Division, University of California at Berkeley, Berkeley, CA 94720-1776, USA
Proceedings of the IEEE ICCV (1998)

[2] Serge Belongie, Thomas Leung, Jitendra Malik and Jianbo Shi:
Contour and Texture Analysis for Image Segmentation
Computer Science Division, University of California at Berkeley, Berkeley, CA 94720-1776, USA
submitted to the International Journal of Computer Vision (December 1999)

[3] Stan Birchfield:
Elliptical Head Tracking Using Intensity Gradients and Color Histograms
Computer Science Department, Stanford University, Stanford, CA 94305, USA
Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, California, pp. 232-237 (June 1998)

[4] Christopher M. Bishop:
Neural Networks for Pattern Recognition
Oxford University Press (1995)
ISBN 0-19-853864-2

[5] Dorin Comaniciu and Peter Meer:
Robust Analysis of Feature Spaces: Color Image Segmentation
Department of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ 08855, USA
Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, June 1997, pp. 750-755

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin:
Maximum likelihood from incomplete data via the EM algorithm
Journal of the Royal Statistical Society B, vol. 39, pp. 1-38 (1977)

[7] Richard O. Duda and Peter E. Hart:
Pattern Classification and Scene Analysis
John Wiley & Sons (1973)
ISBN 0-471-22361-1

[8] Dirk Focken:
Adaptive Hintergrundmodelle zur Personenverfolgung
Studienarbeit am ILKD (Prof. A. Waibel), Universität Karlsruhe (Juli 2000)

[9] D. M. Gavrila:
The Visual Analysis of the Human Movement: A Survey
Image Understanding Systems, Daimler-Benz Research, 89081 Ulm, Germany and Computer Vision Laboratory, University of Maryland at College Park, MD 20742, USA
published in Computer Vision and Image Understanding, vol. 73, no. 1, pp. 82-98 (1999), Academic Press

[10] Ismail Haritaoglu, David Harwood, and Larry S. Davis:
W^{4}S: A Real-Time System for Detecting and Tracking People in 2½ D
Computer Vision Laboratory, University of Maryland at College Park, MD 20742, USA

[11] R.J. Kauth, A.P. Pentland, and G.S. Thomas:
Blob: An unsupervised clustering approach to spatial preprocessing of mss imagery.
In 11th Int'l Symposium on Remote Sensing of the Environment, Ann Arbor, MI, April 1977

[12] Tom M. Mitchell:
Machine Learning
McGraw-Hill International Editions, Computer Science Series (1997)
ISBN 0-07-115467-1

[13] Christopher R. Wren, Ali Azarbayejani, Trevor Darrell, and Alex „Sandy" Pentland:
Pfinder: Real-Time Tracking of the Human Body.
MIT Media Laboratory Perceptual Computing Section, Cambridge, MA 02139, USA
Technical Report No. 353
published in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 780-785 (July 1997)

[14] Jie Yang, Xiaojin Zhu, Ralph Gross, John Kominek, Yue Pan, Alex Waibel:
Multimodal People ID for a Multimedia Meeting Browser
Technical Report, Interactive Systems Laboratories, Carnegie Mellon University, Pittsburgh, PA 15213, USA (1999)

[15] Liang Zhao:
Recursive Context Reasoning for Human Detection and Part Identification
Ph.D. Thesis Proposal, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA (2000)