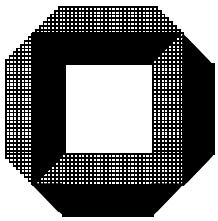

Universität Karlsruhe (TH)
Fakultät für Informatik
Institut für Komplexität und Logik
Prof. A. Waibel

Adaptive Hintergrundmodelle zur Personenverfolgung

Studienarbeit
von
Dirk Focken

Stand: 26. Juli 2000

Betreuer: Rainer Stiefelhagen

Inhaltsverzeichnis

Kapitel 1

Einleitung

Bisher war die Kommunikation zwischen Mensch und Maschine durch spezielle Eingabegeräte wie der Tastatur oder der Maus geprägt. Neue Ansätze für den Mensch-Maschine Dialog versuchen, ohne diese Eingabehilfsmittel auszukommen. Dies bedeutet, dass gesprochene Sprache oder Gestik bzw. die Körperhaltung eines Menschen als natürliche Eingabemittel im Mensch-Maschine Dialog verwendet werden.

Um die Gestik oder allgemeiner das Verhalten eines Menschen zu verstehen, kann in einem ersten Schritt ein Mustererkennungsprozess mittels einer Kamera versuchen, den Menschen im Bild zu finden und ihn zu verfolgen.
In weiteren Schritten kann dieses Wissen, um die augenblickliche Position eines Menschen, genutzt werden, um die Gestik und Körperhaltung des Menschen zu schätzen, um dann hieraus eine Botschaft des Menschen an die Maschine abzuleiten.

Auf diese Weise ist es möglich, einen intelligenten Raum zu schaffen, in dem ein Mensch auf natürliche Art mit Maschinen kommuniziert. Er kann rein durch seine Position im Raum bestimmen, wie die Lichtverhältnisse anzupassen sind oder per Gestik einen Anruf auf die Freisprechanlage umschalten. Beispiele für solche intelligenten Räume werden in [Maes et al., 1995] und [M.Torrance, 1995] beschrieben.

Die vorliegende Arbeit führt einen Teil des obigen ersten Schrittes 'Personenverfolgung' aus. Es wird die Silhouette von Personen bzw. von bewegten Objekten aus einer Bildfolge extrahiert und mit Zusatzinfomationen wie dem Schwerpunkt oder der 'Bounding Box' versehen. Diese Extraktion wird mit Hilfe von adaptiven Hintergrundmodellen durchgeführt.
Mit den Silhouetten bewegter Objekte kann ein Verfolgungsalgorithmus den obigen ersten Schritt vollends ausführen, nämlich die Personen im Bild zu verfolgen.

Allgemein ist das in dieser Arbeit behandelte Problem der Vordergrundsegmentierung[1], d.h. der Extraktion bewegter Objekte aus einer Bildfolge, eines der grund-

[1]Sich bewegende Objekte werden als Vordergrund angesehen.

legenden Probleme, die Bildverarbeitungssysteme lösen müssen, deren Anwendungsbereich nicht nur in interaktiven Mensch-Maschine Dialogsystemen wie z.B. bei intelligenten Räumen liegt, sondern auch die Anwendung finden im Bereich der Videoüberwächung bzw. bei Videokonferenzen.

Im folgenden Kapitel wird ein kurzer Überblick über Vordergrundsegmentierungsverfahren gegeben.
Im dritten Kapitel wird ausführlich die Thematik der 'Segmentierung mittels Hintergrundmodelle' behandelt und die darauf basierenden Algorithmen erläutert, die im Rahmen dieser Arbeit implementiert wurden.
Im vierten Kapitel werden die Probleme der beiden implementierten Algorithmen verdeutlicht. Am Ende des Kapitels soll ein kurzer qualitativer Vergleich dieser Algorithmen, einen Eindruck für ihre Leistungsfähigkeit geben.
Im fünften Kapitel wird das objektorientierte Design und die damit erstellte Klassenhierachie beschrieben, mit dessen Hilfe die Algorithmen realisiert wurden, und die auch als Rahmen für ähnliche Algorithmen dieser Art verwendet werden kann.
Das letzte Kapitel gibt eine Zusammenfassung der Arbeit und verweist auf Erweiterungsmöglichkeiten, die in der Zukunft erstrebenswert wären.
Der Anhang enthält einige Abschnitteüber in der Bildverarbeitung wohlbekannte Techniken zur Bildauswertung.
Die implementierten Algorithmen und der dabei geschaffene Rahmen (eine C++-Klassenhierarchie) wurden in der 'KANTS'-Bibliothek zusammengefasst. Die Dokumentation in Form von HTML-Seiten liegt der erstellten Klassenbibliotek 'KANTS' bei.

Kapitel 2

Vordergrundsegmentierung in Bildfolgen

Es gibt einige Ansätze, um Vordergrund von Hintergrund zu trennen. Der einfachste ist die Hintergrundsubtraktion. Es wird angenommen, dass das Hintergrundbild vollständig bekannt ist. Man muss es nur vom gerade beobachteten Bild 'abziehen', um den Vordergrund zu erhalten.

Ein anderer Ansatz führt eine Änderungsdetektion durch, um den Vordergrund zu extrahieren. Eine Änderungsdetektion kann aus einem einfachen Differenzenbild (auf Pixelebene) bestehen. Dabei wird eine Änderung eines Pixels detektiert, wenn die Differenz zum Pixel des vorigen Bildes einen Schwellwert überschreitet. Eine leistungsfähigere Änderungsdetektion beschreibt die Arbeit [Hsu et al., 1984], die nicht mehr auf einzelnen Pixeln sondern auf lokalen Umgebungen Änderung zum vorigen Bild schätzt. Dieser Ansatz nimmt an, dass sich Vordergrundobjekte stets bewegen. Folglich können diese Verfahren ein kurz stillstehendes Objekts nicht mehr als Vordergrund klassifizieren.

Ebenfalls auf Änderung allerdings zwischen mehreren Folgebildern basieren 'optische Fluss'-Verfahren (z.B. [Nagel, 1987]). Sie schätzen neben der reinen Änderung auch die Richtung der Änderung, d.h. wohin das sich bewegende Objekt im Bild 'fliesst'. Optischer Fluss ist ein sehr leistungsfähiger aber sehr Rechenzeit intensiver Ansatz. Für Echtzeitanwendungen muss daher auf spezielle Hardware zurückgegriffen werden.

Die zuerst angesprochene einfache Methode der Hintergrundsubstraktion kann noch um einiges verfeinert werden. Sicherlich wird nur in Ausnahmefällen das Hintergrundbild schon im voraus bekannt sein und für alle Zeit unverändert bleiben; vielmehr wird man anfangs eine 'leere' Szene als Hintergrundbild aufnehmen und dann dieses immer wieder adaptieren. Die ständige Anpassung des Hintergrundbildes ist nötig, um Änderungen der Lichtverhältnisse oder des Hintergrundes selbst Rechnung zu tragen. Ein solches Verfahren könnte sogar ohne ein Hintergrundbild als Initialisierung auskommen, indem das Hintergrundbild durch längere Beobachtung der Szene direkt geschätzt wird. Dann ist es besser nicht

von Adaption eines Hintergrundbildes sondern von Hintergrundmodellierung zu sprechen.

Hintergrundmodelle leisten zweierlei Dinge. Einerseits sind damit die Pixel im aktuellen Bild als Hintergrund- bzw. als Vordergrundpixel klassifizierbar und andererseits geben sie implizit an, auf welche Weise das Hintergrundbild an die Gegebenheiten anzupassen ist.

Die in dieser Arbeit implementierten Algorithmen basieren auf dem Ansatz der Hintergrundmodellierung. Zum einen war das Ziel der Arbeit einen Rahmen zu entwickeln, um Algorithmen, die Hintergrund modellieren, einfach und schnell realisieren zu können und zum anderen diesen Rahmen zu nutzen, um zwei Hintergrundmodelle zu implementieren. Diese beiden Modelle wurden genutzt, um Silhouetten von Personen in einer Echtzeitanwendung zu extrahieren.

Im folgenden Kapitel wird zuerst allgemein auf adaptive Hintergrundmodelle eingegangen, um dann zu den zwei implementierten Verfahren zu kommen.

Kapitel 3

Adaptive Hintergrundmodelle

Ein sehr einfaches Hintergrundmodell schätzt das Hintergrundbild, in dem die durchschnittlichen Farbwerte aller Pixel im Bild berechnet werden, wobei alle Bilder aus der nahen Vergangenheit zur Berechnung herangezogen werden. Diese Strategie verspricht nur dann, Erfolg zu haben, wenn sich alle Vordergrundobjekte ausreichend schnell bewegen. Bleibt ein Objekt länger stehen, so wird es langsam in den Hintergrund aufgenommen. Setzt das Objekt seine Bewegung fort, ist das Hintergrundbild für längere Zeit an dieser Stelle unbrauchbar bis die durchschnittlichen Farbwerte wieder im Bereich des eigentlichen Hintergrundes liegen.

Außerdem muss für die eigentliche Vordergrundsegmentierung ein Schwellwert im voraus festgelegt werden, der angibt mit welcher Abweichung vom erwarteten Hintergrundfarbwert ein beobachtetes Pixel als zum Vordergrund zugehörig erklärt wird. Dieser Schwellwert kann nicht während der Analyse einer Bildfolge sinnvoll geändert werden, da man keine Informationen darüber gesammelt hat, in welchem Ausmaß die Hintergrundfarbwerte eines Pixels schwanken.

Um solche Zusatzinformation zu erlangen, berechnet das Verfahren beschrieben in [Haritaoglu et al., 1998] zum einen den durchschnittlichen Hintergrundfarbwert eines Pixels und speichert zum anderen die minimale und maximale Abweichung vom Durchschnittswert. Damit kann der Schwellwert für jeden Pixel entsprechend angepaßt werden.

Der nächste Schritt besteht darin die Hintergrundfarbwerte eines Pixels statistisch zu modellieren.

3.1 Hintergrundfarbe als Gaußverteilung

Um den Hintergrundfarbwert eines Pixels in einer Bildfolge zu schätzen, wird in vielen Arbeiten die Annahme gemacht, dass dieser normal verteilt ist (z.B. [Wren et al., 1999], [Horprasert et al., 1999] und [Francois and Medioni, 1999]). Diese Arbeitshypothese bildet auch die Grundlage für das zuerst in dieser Studi-

enarbeit implementierte Verfahren zur Vordergrundsegmentierung, welches sich an die Publikation von [Yang et al., 1999] anlehnt.

In diesem Algorithmus wird der Hintergrundfarbwert eines Pixels in der Bildfolge als eine Gaußverteilung mit Dichtefunktion $p(\mathbf{x})$ modelliert. Daher reduziert sich das Problem der Schätzung der aktuellen Hintergrundfarbverteilung auf das Schätzen des Erwartungswertvektors μ (geschätzter Hintergrundfarbwert) und der Kovarianzmatrix Σ (geschätzte Schwankung des Farbwertes), die eine Gaußverteilung eindeutig charakterisieren:

$$p(\mathbf{x}) = \eta(\mathbf{x}, \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^t \Sigma^{-1}(\mathbf{x}-\mu)} \tag{3.1}$$

Da Farbräume auch mehrdimensional sein können (RGB, YUV oder HSV), wird hier gleich der mehrdimensionale Fall betrachtet; womit ein Farbwert eigentlich zu einem Farbvektor wird.
Außerdem ist zu beachten, dass die in diesem Verfahren verwendete Kovarianzmatrix, um den Algorithmus auch für Echtzeitanwendungen brauchbar zu machen, als diagonal angenommen wird. Zum Beispiel hätte Σ für den RGB-Farbraum die Form:

$$\Sigma = \begin{bmatrix} \sigma_R^2 & 0 & 0 \\ 0 & \sigma_G^2 & 0 \\ 0 & 0 & \sigma_B^2 \end{bmatrix}. \tag{3.2}$$

Für die Leser, die mit Gaußverteilungen wenig vertraut sind, bietet der Anhang A.1 eine kurze Darstellung der grundlegenden Eigenschaften und Merkmale dieser Verteilungsklasse.

Um den Algorithmus zu initialisieren, muss zuerst ein Hintergrundbild aufgenommen werden. Damit wird der Mittelwertvektor der Gaußdichtefunktion eines jeden Pixels initialisiert, indem der Mittelwert auf den Hintergrundfarbwert an derweiligen Stelle gesetzt wird und die Standardabweichungen der Farbkomponenten hohe Initialwerte erhalten. Das Verfahren normalisiert dazu alle Farbwerte auf das Intervall $[0, 1]$ und sieht hohe initiale Standardabweichungen als Werte im Bereich von $0.1 - 0.2$ an.

Auf diese Weise hat man schon ein erstes initiales Hintergrundmodell erstellt, mit dessen Hilfe nun Vordergrund- bzw. Hintergrundpixel klassifiziert werden können. Der Pixel an der Stelle (l,m) im aktuellen Bild wird als Vordergrundpixel klassifiziert, wenn nur eine Komponente x_i des aktuellen Farbwerts $\mathbf{x}$ mehr als $c * \sigma_i$ (wobei typischerweise $c = 2.5$ gewählt wird) von der zugehörigen Komponente μ_i des Mittelwerts μ abweicht; ansonsten gehört der Pixel (l,m) zum Hintergrund:

Vordergrundpixel	wenn $\exists x_i : \lvert x_i - \mu_i \rvert > 2.5\sigma_i$
Hintergrundpixel	sonst

Man sieht, wie der Schwellwert 2.5σ für die Abweichung vom Hintergrundfarbwert μ für jeden Pixel verschieden sein kann. So kann dem Phänomen Rechnung getragen werden, dass helle Regionen im Bild mehr streuen als vergleichsweise dunkle Regionen.

Als nächstes muss das Hintergrundmodell an die neu beobachteten Farbwerte des aktuellen Bildes adaptiert werden. Nur stellt sich die Frage, welche Pixel zu adaptieren sind.
Bei Pixeln, die klar zu Vordergrundobjekten gehören, will man das Hintergrundmodell nicht adaptieren, da ja kein Hintergrundpixel beobachtet wurde, und so eine Adaption das Hintergrundbild verfälschen würde. Die Entscheidung, welche Pixel wirklich zu Vordergrundobjekten gehören, kann sicher nicht dadurch beantwortet werden, dass man einfach alle als Vordergrundpixel klassifizierten Pixel als solche ansieht. Zum Beispiel können Rauscheinflüsse Pixel als Vordergrund klassifizieren, die eigentlich gerade Hintergrund zeigen und umgekehrt können so auch als Hintergrund angesehene Pixel Vordergrund sein. Wie man diese Frage besser beantwortet soll später behandelt werden.
Man nehme nun an, dass ein Satz von 'möglichst echten' Hintergrundpixeln vorgegeben ist.

Man adaptiert nur diese Hintergrund repräsentierenden Pixel. Der Mittelwert und die Diagonalkomponenten der Kovarianzmatrix, die zu einem Hintergrundpixel gehören, werden mit Hilfe einer Lernrate α und dem aktuellen Farbwert $\mathbf{x}$ wie folgt angepasst:

$$\mu = (1 - \alpha)\mu + \alpha(\mathbf{x} - \mu) \tag{3.3}$$

und

$$\Sigma = (1 - \alpha)\Sigma + \alpha(\mathbf{x} - \mu)(\mathbf{x} - \mu)^t \tag{3.4}$$

Diese Adaptionsformeln basieren zum einen auf einer Standardtechnik zum Schätzen der Gaußverteilungsparameter (Maximum-Likelihood Schätzung) und auf einem rekursiven Filter, um die Schätzung der Parameter für die Praxis praktikabel zu machen. Eine knappe Darstellung dieser Standardtechniken der Signalverarbeitung kann im Anhang (A.2 und A.3) nachgelesen werden, wo auch auf die entsprechende Literatur hingewiesen wird. Die Lernrate α liegt zwischen 0 und 1. Ist sie nahe bei 0 adaptiert sich der Hintergrund langsam an einen neuen Hintergrundwert oder anders ausgedrückt es werden Farbwerte des Pixels noch zur Schätzung der Parameter miteinbezogen, die stärker aus der Vergangenheit stammen. Liegt die Lernrate als Extrem bei 1 wird der aktuelle Farbwert des Hintergrundpixels als Mittelwert des Hintergrunds aufgefaßt. Im schon angesprochenen Abschnitt des Anhangs A.3 wird genauer auf die Auswirkungen der Wahl der Lernrate eingegangen.

Der obig beschriebene Algorithmus leistet zwei Dinge: Zum einen ermöglicht er aufgrund des gebildeten Hintergrundmodells die Klassifikation von Pixeln eines

Bildes in einer Bildfolge als Vorder- bzw. Hintergrundpixel. Zum anderen adaptiert er das geschätzte Hintergrundbild anhand neuer Bilder. Als freie Parameter des Algorithmuses treten der Faktor c bei der Klassifikation und die Lernrate α bei der Adaption des Hintergrundbildes auf.

Wie schon angesprochen reicht die bloße Klassifikation der Pixel als Vorder- und Hintergrundpixel nicht aus, um die Vordergrundobjekte zu extrahieren. Eine Nachverarbeitung, die Zusatzannahmen über die Vordergrundobjekte ausnutzt, muss versuchen Rauschphänomene zu unterdrücken, um einen 'möglichst echten' Satz an Hinter- bzw. Vordergrundpixeln zu finden.

3.1.1 Vom Vordergrundpixel zum Vordergrundobjekt

Zuerst kann angenommen werden, dass Vordergrundobjekte zusammenhängend sind und eine gewisse Mindestgröße aufweisen. So werden mit einem aus [Horn, 1986] wohlbekannten Verfahren die zusammenhängenden Komponenten, die aus Vordergrundpixeln bestehen, extrahiert (wobei das implementierte Verfahren den bekannten 4-er Zusammenhang benutzt). Die Pixel einer Komponenten, die eine Mindestgröße unterschreitet, werden fortan als Hintergrund betrachtet, weil man sie als Rauschphänomene ansieht. Die folgende Abbildung zeigt diesen Prozess.

Abbildung 3.1: Rauschunterdrückung mit Zusammenhangsoperator. Links sind alle zusammenhängenden Komponenten sichtbar. Rechts nur die Komponenten, die größer als 5 Pixel sind.

Anschließend führt die Annahme, dass Vordergrundobjekte in sich keine Löcher aufweisen (bei Personen ist dies sicher gerechtfertigt) dazu, Löcher mit Hilfe morphologischer Operatoren zu schließen. Wieder gibt der Anhang A.4 eine kurze Darstellung, wie morphologische Operatoren definiert sind. Die Auswirkungen dieser Operatoren zeigen die nachfolgenden Bilder (Abbildung 3.2). Bei dem implementierten Algorithmus wird das 'Closing' für das Schließen der Löcher in Vordergrundobjekten eingesetzt.

Um große Löcher zu schließen, kann es ratsam sein, einen speziellen morphologischen Operator mit einem besonderen Strukturelement zu definieren. Die Abbildung 3.3 zeigt ein Strukturelement und die damit erzielten Ergebnise. Das implementierte Verfahren kann alternativ auch diesen speziellen Operator für die Vordergrundpixelnachbearbeitung nutzen.

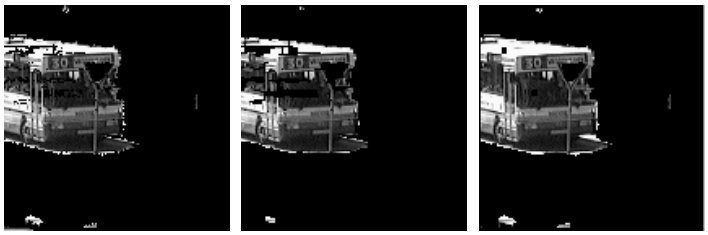

Abbildung 3.2: 'Opening' und 'Closing' mit einer 5×5 Maske. Links: das Originalbild. Mitte: Ergebnis des 'Opening'. Rechts: Ergebnis des 'Closing'.

Abbildung 3.3: Dilatation mit einem Strukturelement. Links: das Strukturelement. Mitte: das Originalbild. Rechts: Ergebnis der speziellen Dilatation

Nach diesen beiden Nachverarbeitungsschritten liegen die eigentlichen Vordergrundobjektregionen vor. Einige der ursprünglichen Vordergrundpixel wurden als Rauschphänomene dem Hintergrund zugeordnet, andererseits wurden durch das Schließen der Löcher einige Hintergrundpixel zum Vordergrund zugehörig erklärt. Die vorher angesprochene Adpation wird in dem implementierten Verfahren nur auf denjenigen Pixeln durchgeführt, die nach der eben besprochenen Nachbearbeitung als Hintergrund angesehen werden.

Das zweite in dieser Arbeit implementierte Verfahren verwendet mehrere Gaußverteilungen pro Pixel, um den Hinter- und den Vordergrund zu modellieren.

3.2 Mixtur von Gaußverteilungen als Hintergrundmodell

Einen allgemeineren Ansatz verfolgt C. Stauffer in [Stauffer and Grimson, 1998]. Hier werden Pixelfarbwerte für den Vorder- und Hintergrund durch mehrere Gaußverteilungen modelliert. Dieser Ansatz stützt sich auf die Theorie der 'Gaussian mixture models', die zum Beispiel in 'Neural Networks for pattern recognition' [Bishop, 1995] ausführlicher erläutert wird. Im folgenden wird nur auf die für das Verständnis des Algorithmuses nötigen Grundlagen eingegangen.

Unter einer 'Mixture of Gaussians' versteht man eine Zufallsverteilung mit Dichtefunktion $p(\mathbf{x})$, die aus einer Linearkombination von Gaußdichtefunktionen $p(\mathbf{x}|i)$

zusammengesetzt ist. Man schreibt:

$$p(\mathbf{x}) = \sum_{i=1}^{m} p(\mathbf{x}|i)P(i). \tag{3.5}$$

Die $P(i)$ werden als Mixturparameter (mixing parameter) oder auch als Gewichte bezeichnet (dann werden die $P(i)$ meist als ω_i geschrieben). Man muss fordern, dass diese $P(i)$ die folgenden Bedingungen erfüllen, damit $p(\mathbf{x})$ auch wirklich eine Zufallsverteilung darstellt:

$$\sum_{i=1}^{m} P(i) = 1 \tag{3.6}$$

$$0 \leq P(i) \leq 1 \tag{3.7}$$

Man kann die $P(i)$ als a priori Wahrscheinlichkeiten ansehen, dass ein $\mathbf{x}$ zu einer bestimmten Komponentenverteilung $p(\mathbf{x}|i)$ 'gehört'. Dann gibt die Darstellung 3.5 die Wahrscheinlichkeit $p(\mathbf{x})$ wieder, dass $\mathbf{x}$ überhaupt beobachtet wird. Die Komponentenverteilungen $p(\mathbf{x}|i)$ kann man so auch als klassenbedingte Verteilungen ansehen; die Komponentenverteilungen sind hier immer Gaußverteilungen. Die a posteriori Wahrscheinlichkeit, dass ein beobachtetes $\mathbf{x}$ zu einer Komponentenverteilung i gehört, ist nach Bayes:

$$P(i|\mathbf{x}) = \frac{p(\mathbf{x}|i)P(i)}{p(\mathbf{x})}. \tag{3.8}$$

Nach dieser kurzen Erklärung der Grundbegriffe für gaußische Mixturverteilungen ist zu klären, wie man hiermit Vorder- bzw. Hintergrundfarbwerte eines Pixels in einer Bildfolge modellieren kann.

In dem von Stauffer ([Stauffer and Grimson, 1998]) vorgestellten Verfahren modelliert eine Komponentenverteilung also eine Gaußverteilung jeweils eine Ballung von Farbwerten eines Pixels.
So könnte es in einer Straßenszene für ein Pixel z.B. drei Komponentenverteilungen geben, wobei eine die Farbballung, die vom grauen Aspahlt herrrüht, repräsentiert und die beiden übrigen Verteilungen ein rotes und schwarzes Auto darstellen, die vor kurzer Zeit durch das Sichtfeld dises Pixels gefahren sind.
Mit der Annahme, dass Hintergrund am häufigsten beobachtet wird und dass statischer Hintergrund eine niedrige Varianz aufweist, werden grob gesagt, die Komponentenverteilungen als Hintergrundverteilungen angesehen deren zugehörige Ballung sich auf viele Farbwerte und eine niedrige Varianz dieser Farbwerte stützt; im obigen Beispiel erfüllt diese Bedingung die Verteilung des grauen Asphaltes. Alle übrigen Komponentenverteilungen werden als Vordergrundverteilungen betrachtet; für das obige Beispiel wären dies die Verteilung der beiden Autos.
Liegt ein neu beobachteter Farbwert eines Pixels in der Nähe einer Hintergrundverteilung wird das Pixel als Hintergrundpixel klassifiziert sonst als Vordergrundpixel. Soweit zur Grundidee dieser Hintergrundmodellierungsmethode, um nun zur detaillierten Beschreibung des Verfahrens zu kommen.

3.2.1 Adaption bei mehreren Gaußverteilungen

Aufgrund der t zuletzt beobachteten Farbwerte $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t$ eines Pixels ist eine Mixtur von K Gaußverteilungen zu schätzen. Die Wahrscheinlichkeit, den aktuellen Farbwert $\mathbf{x}_t$ zu beobachten, ist nach diesem Modell:

$$P(\mathbf{x}_t) = \sum_{i=1}^{K} \omega_{i,t} \eta(\mathbf{x}_t, \mu_{i,t}, \Sigma_{i,t}) \tag{3.9}$$

wobei K die Anzahl der Verteilungen, $\omega_{i,t}$ das Gewicht (bzw. die a priori Wahrscheinlichkeit) für die $i$te Gaußverteilung zur Zeit t, $\mu_{i,t}$ der Erwartungswert der $i$ten Komponentenverteilung zur Zeit t, $\Sigma_{i,t}$ die Kovarianzmatrix zur $i$ten Gaußverteilung zur Zeit t und schließlich $\eta(\mathbf{x}_t, \mu_{i,t}, \Sigma_{i,t})$ eine Gaußverteilung wie in 3.1 repräsentiert.

Die Größe von K hängt bei diesem Algorithmus von der gewünschten Leistung bzw. von der verfügbaren Speicherkapazität ab. In der Publikation von Stauffer wird als typischer Bereich von K $3-5$ angegeben. Außerdem wird aus Gründen der gewünschten Echtzeitanwendbarkeit des Algorithmuses die Kovarianzmatrix Σ als diagonal angenommen, d.h. die einzelnen Komponenten eines Farbwertvektors werden als stochastisch unabhängig voneinander betrachtet, was sicher nicht die Realität wiederspiegelt:

$$\Sigma_{i,t} = \sigma_i^2 I \tag{3.10}$$

Ein neu beobachteter Farbwert $\mathbf{x}_{t+1}$ wird auf Zugehörigkeit zu einem der K Komponentenverteilungen geprüft. Dabei wird $\mathbf{x}_{t+1}$ als zugehörig zu einer Gaußverteilung angesehen, wenn $\mathbf{x}_{t+1}$ innerhalb von c Standardabweichungen liegt (wobei wieder typischer Weise c=2.5 gewählt wird).

Kann aber $\mathbf{x}_{t+1}$ keiner Komponentenverteilung zugeordnet werden, so wird die Verteilung mit dem niedrigsten Gewicht $\omega_{i,t}$ durch eine neue Gaußverteilung ersetzt, deren Mittelwert auf den aktuellen Farbwert $\mathbf{x}_{t+1}$ gesetzt wird und die ein anfangs niedriges Gewicht $\omega_{i,t+1}$ und eine hohe Standardabweichung $\sigma_{i,t+1}$ hat. Nun werden die Gewichte der Komponentenverteilungen wieder auf 1 normalisiert.

Im Fall, dass eine zugehörige Gaußverteilung gefunden wurde oder eine Gaußverteilung ersetzt wurde, werden die Gewichte wie folgt adaptiert:

$$\omega_{i,t+1} = \begin{cases} (1-\alpha)\omega_{i,t} + \alpha & \text{wenn } i \text{ ersetzte/zugehörige Verteilung} \\ (1-\alpha)\omega_{i,t} & \text{sonst} \end{cases} \tag{3.11}$$

Dies stellt einen rekursiven Filter dar, der mit der Wahl der Lernrate α ein mehr oder weniger breiten Ausschnitt der Vergangenheit in die Berechnung der Gewichte mit einbezieht. Näheres darüber ist im Anhang zu finden (A.3).

Wurde eine zugehörige $z$te Gaußfunktion gefunden, werden noch die Parameter dieser an den neuen Farbwert $\mathbf{x}_{t+1}$ angepaßt:

$$\mu_{z,t+1} = (1-\rho)\mu_{z,t} + \rho * \mathbf{x}_{t+1} \tag{3.12}$$

$$\sigma^2_{z,t+1} = (1-\rho)\sigma^2_{z,t} + \rho * (\mathbf{x}_{t+1} - \mu_{z,t+1})^t(\mathbf{x}_{t+1} - \mu_{z,t+1}) \tag{3.13}$$

wobei ρ die a posteriori Wahrscheinlichkeit darstellt, dass $\mathbf{x}_{t+1}$ zu der Gaußverteilung η_z gehört

$$\rho = \alpha\eta_z(z|\mathbf{x}_{t+1}) = \alpha\frac{p(\mathbf{x}|z)\omega_{z,t}}{p(\mathbf{x})}. \tag{3.14}$$

Ein Vorteile einer solchen Modellierung gegenüber dem vorigen Verfahren kann man sich direkt am Modell überlegen.
Bewegt sich ein Objekt in den Bildbereich und verweilt dort für lange Zeit (z.B. Parken eines Autos oder eine abgestellte Tasse), so ist der ursprüngliche Hintergrund nicht vergessen, sondern ist immer noch in einer Gaußverteilung gespeichert. Wird der Gegenstand wieder entfernt, kann die ursprüngliche Gaußverteilung des Hintergrunds schnell an Gewicht gewinnen und so das Hintergrundbild schneller als bei Ein-Gaußglockenmodellen wieder auf den aktuellen Stand gebracht werden.

Für die Adaption des Mixturmodells wurden nur zwei freie Parameter verwendet: c für die Zugehörigkeitsbestimmung und α als Lernrate bei der Adaption selbst.

3.2.2 Klassifikation in Hinter- und Vordergrundverteilungen

Der statische Hintergrund eines Bildes weist meist eine längere Sichtbarkeit und eine geringere Varianz als sich im Bild bewegende Objekte auf. Daher nimmt man an, dass der Hintergrund von den Verteilungen repräsentiert wird, die das höchste Gewicht $\omega_{i,t}$ und die geringste Varianz $\sigma_{i,t}$ aufweisen.

Um zu entscheiden welche Verteilungen den Hintergrund modellieren, werden daher die einzelnen Gaußverteilung absteigend gemäß ω/σ geordnet. Dieser Wert steigt mit zunehmenden Gewicht und mit sich verringender Standardabweichung. Die Verteilungen, die am wahrscheinlichsten Hintergrund repräsentieren, sind bei dieser Ordnung immer am Anfang zu finden, während die übrigen Verteilungen weiter hinten stehen. Eine neue Verteilung, die durch das Verdrängen einer alten entsteht, werden mit ihrem geringen Anfangsgewicht und hohen Varianz vorerst am Ende der Liste stehen. Dagegen werden Verteilungen, die schon lange sichtbar sind (hohes Gewicht) unter den ersten Verteilungen zu finden sein.

Die Hintergrundverteilungen stellen die ersten B Verteilungen in der eben erwähnten Liste dar, wobei

$$B = argmin_b \left(\sum_{i=1}^{b} \omega_i > T \right) \tag{3.15}$$

und T ein Schwellwert ist, der angibt welcher Gewichtsteil der Verteilungen zum Hintergrund zugehörig gesehen werden soll. Die übrigen Verteilungen stellen die Vordergrundverteilungen dar.

Wenn T klein gewählt wird (z.B. 0.01), so ist nur die erste Verteilung als Hintergrundverteilung zu betrachten. Ist T dagegen größer, können manchmal die ersten beiden Verteilungen als Hintergrund betrachtet werden. In Räumen ist der Hintergrund meist durch eine Gaußglocke hinreichend erklärt, während bei Straßenszenen Hintergründe aus zwei Gaußverteilungen Vorteile bieten können. Als einfachtes Beispiel denke man an ein blinkendes Warnlicht auf einer Baustelle.

Ein Pixel wird danach als Vordergrundpixel klassifiziert, wenn er nicht in der $c * \sigma$-Umgebung einer Hintergrundverteilung liegt. Liegt er aber innerhalb einer $c * \sigma$-Umgebung einer Hintergrundverteilung wird der Pixel als Hintergrundpixel klassifiziert.

Das Verfahren adaptiert in oben beschriebener Weise für jeden Pixel der Bildfolge seine Gaußverteilungen und entscheidet welche Gaußglocken als Hinter- bzw. Vordergrundverteilungen gelten. Damit kann man zu jedem Zeitpunkt Pixel als Vorder- oder Hintergrundpixel klassifizieren. Die freien Parameter dieses schon komplizierteren Verfahrens sind c um die Klassifikation durchzuführen, α um die Lernrate bei der Adaption festzulegen und T um die Anzahl der möglichen Hintergrundverteilungen festzulegen.
Allerdings ist trotzdem danach eine Nachbearbeitung wie bei den Ein-Gaußglockenmodellen nötig, um Rauscheinflüsse und Löcher in Objekten zu unterdrücken bzw. zu schließen. Schließlich erhält man danach die endgültigen Vordergrundregionen.

Es ist wichtig zu bemerken, dass der Mixturansatz trotzdem alle Pixel im Bild einem Adaptionsschritt unterzieht, da auch direkt der Vordergrund in diesem Hintergrundmodell modelliert wird.

Im nächsten Kapitel werden die allgemeinen Probleme angesprochen, die bei dieser Art der Hintergrundmodellierung und zwar bei beiden Verfahren auftreten. Außerdem wird ein zusätzlicher Nachbearbeitungsschritt vorgestellt, der Probleme bei der Silhouettenextraktion von Personen abmildert.

Kapitel 4

Probleme und Vergleich der implementierten Hintergrundmodelle

4.1 Probleme der implementierten Hintergrundmodelle

Die im vorigen Kapitel beschriebenen Hintergrundmodelle haben Probleme, bei schneller Veränderung der Lichtverhältnisse das Hintergrundbild korrekt zu schätzen bzw. Schatten als Hintergrund zu detektieren. Ausserdem werden Vordergrundobjekte, die einen geringen Kontrast zum durch sie verdeckten Hintergrund aufweisen, oft nicht als Vordergrund erkannt. Im folgenden werden diese Probleme anhand von Abbildungen kurz erläutert. Es ist anzumerken, dass nicht nur die beiden detailliert beschriebenen und implementierten Hintergrundmodelle Probleme dieser Art haben, sondern dass es sich bei den vorgestellten Problemen um allgemeine Schwachstellen des Hintergrundmodellansatzes handelt.

4.1.1 Probleme bei geringem Kontrast

In Abbildung 4.1 wird deutlich, dass der Segmentieralgorithmus Objekte, die einen zu niedrigen farblichen Kontrast zum Hintergrund aufweisen, als Hintergrund detektiert. Abhilfe könnte ein anderer Farbraum bieten, indem das betreffende Objekt einen höheren Kontrast zum Hintergrund aufweist. Nur wird es dann sicher andere Objekte geben, die unter dieser Konfiguration einen geringen Kontrast zum Hintergrund haben und daher das Problem wieder auftritt. Ohne Zusatzwissen über die Vordergrundobjekte kann man dieses Phänomen kaum in den Griff bekommen.

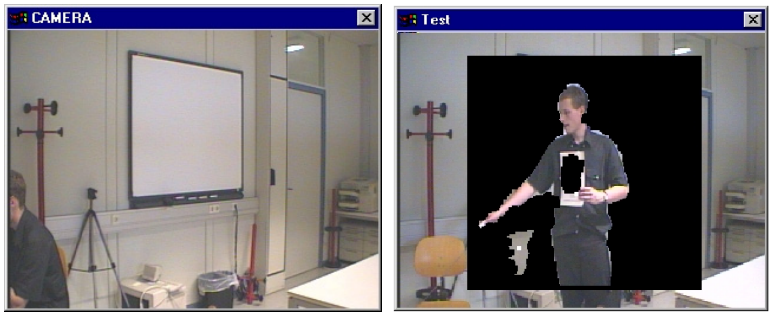

Abbildung 4.1: Links das Originalbild. Rechts die falsche Klassifikation eines weissen Blattes als Hintergrund, da das Papier einen zu niedrigen Kontrast zum weißen Smartboard hat.

4.1.2 Probleme bei Schatten

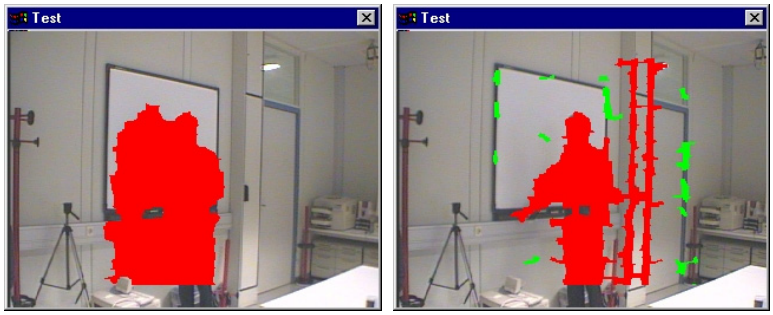

Abbildung 4.2: Links wird Schatten als Vordergrund erkannt (RGB-Farbraum). Rechts wird der Schatten als Hintergrund erkannt, aber der chromatische rg-Farbraum hat vermutlich einen geringen Signal-Rauschabstand, womit sich die zusätzlichen Artefakte erklären ließen.

In Abbildung 4.2 sieht man wie der Ein-Gaußglockenalgorithmus im RGB-Farbraum Schatten fälschlicherweise als Vordergrund erkennt. Um dieses Problem zu mildern, kann man chromatische Farbräume wie HSV oder rg verwenden. Im Unterschied zum RGB-Farbraum trennen diese die Farbinformation stärker von der Intensitätsinformation. Im HSV-Farbraum liegen die Farbinformationen in den Komponenten H und S, während die Intensität durch die V Komponente repräsentiert wird. Beim rg-Farbraum fehlt die Intensitätsinformation völlig. Die rg Farbwerte werden aus den RGB Farbwerten wie folgt berechnet:

$$r = \frac{R}{R+G+B}, g = \frac{G}{R+G+B} \tag{4.1}$$

Dies zeigt, dass die R und G Farbwerte mit $R+G+B$ normalisiert werden. Daher hängen sie nicht mehr von der Intensität (repräsentiert durch $R+G+B$) ab. Man sieht im rechten Bild von Abbildung 4.2, dass die Schatten im rg-Farbraum zwar deutlich schwächer hervortreten, nur scheinen durch das bloße Ausnutzung der Farbinformation mehr Artefakte beim Klassifikationsprozess zu entstehen.

Aus zeitlichen Gründen konnte in dieser Arbeit die Untersuchung der Ursachen dieser Störungen und die Umsetzung des HSV-Farbraums nicht durchgeführt werden.

4.1.3 Probleme bei der Personenverfolgung bzw. starken Lichtänderungen

Beim Ansatz der mehrere Gaußverteilungen pro Pixel verwendet, kann es bei Personen passieren, die sich im Bild für einen längeren Zeitraum nur wenig bewegen, dass sie als Hintergrund klassifiziert werden. Der Ein-Gaußglockenansatz hat dieses Problem in seiner bisher besprochenen Form nicht, da er als Vordergrund klassifizierte Pixel nicht adaptiert.

Als Abhilfe könnte man beim Mixturansatz ähnlich dem Ein-Gaußglockenansatz Objekte, die eine gewisse Mindestgröße aufweisen, einfach immer dem Vordergrund zu schreiben und die Adaption aller Punkte, die zu solch einem Objekt gehören verhindern.
Allerdings hinterlässt dann ein zum Beispiel verschobener Stuhl eine Vordergrundregion an seinem ursprünglichen Platz und bildet eine Vordergrundregion an seinem neuen Platz. Da diese Regionen nicht adaptiert werden, werden diese beiden Regionen nie in den Hintergrund aufgenommen.
In einem anderen Zusammenhang hat dieser Lösungsansatz noch gravierendere Auswirkungen: Änderen sich die Lichtverhältnisse stark, können sich die Farbwerte der Hintergründe stark verschieben. So entstehen grösse Vordergrundregionen im Bild. Mit obigen Ansatz würden diese Regionen nie adaptiert werden. Das Hintergrundbild könnte nie wieder auf seinen aktuellen Stand gebracht werden (gesetzt den Fall, dass die Lichtverhältnisse sich nicht wieder verändern).

Man muss anmerken, dass der Ein-Gaußglockenansatz ebenfalls dieses Problem hat, bei starken Lichtveränderungen nicht mehr adaptieren zu können. Demgegenüber kann er Personen, die stillstehen trotzdem immer als Vordergrund erkennen. Der Mixturansatz findet zwar auch bei starken Lichtveränderungen wieder zum eigentlichen Hintergrundbild, kann aber stillstehende Personen nach einer gewissen Zeit nicht mehr als Vordergrund klassifizieren.

Um zu erreichen, dass beide Verfahren zum einen nach starken Lichtveränderungen wieder zum aktuellen Hintergrundbild finden und zum anderen stillstehende Personen als Vordergrund zu erkennen, wurde nach der Idee aus [Yang et al., 1999] der 'Liveliness-Operator' implementiert.

Dieser erkennt durch das Auswerten der Bewegungen von Vordergrundregionen über mehrere Einzelbilder hinweg, ob eine Vordergrundregion als 'leblos' oder als 'lebendig' anzusehen ist. Eine 'leblose' Region (z.B. Stuhl) wird adaptiert, während 'lebendige' Regionen (Personen) unadaptiert im Vordergrund verbleiben.

Die detaillierte Beschreibung dieses Operators wird im nächsten Abschnitt gege-

ben.

4.2 'Liveliness-Operator'

Der 'Liveliness-Operator' soll Vordergrundregionen entweder als 'lebendig' oder 'leblos' klassifizieren. 'Leblose' Regionen werden adaptiert, da sie Dinge wie verschobene Gegenstände repräsentieren, die eigentlich Hintergrund sind. 'Lebendige' Regionen werden aber nicht adaptiert, da sie Personen also Vordergrund darstellen.

Die Grundidee um die Klassifikation durchzuführen besteht darin, dass 'leblose' Dinge stillstehen, während sich Menschen bewegen. Um die Bewegung einzelner Regionen im Bild zu schätzen, werden sie über mehrere Einzelbilder hinweg betrachtet.
Allerdings verändern sich diese Regionen mehr oder weniger stark von Bild zu Bild; daher stellt das Zuordnen einer Region im vorigen Bild zu ihrer etwas (oder gar stark) veränderten Variante im aktuellen Bild ein Problem dar. Um dieses Problem anzugehen, wird jeder aktuellen Vordergrundregion diejenige alte Vordergrundregion zugeordnet, die den nächst gelegenen Schwerpunkt aufweist und zusätzlich folgende drei Forderungen erfüllt: Der Schwerpunkt der Vordergrundregion darf sich von einem zum anderen Bild nicht zu stark unterscheiden (Schwellwert über Abstand zwischen den beiden Schwerpunkten), die Vordergrundregion muss eine gewisse Mindestgrösse aufweisen und die Regionsgrösse darf sich von alten aufs das neue Bild nur weniger als x (Schwellwert) Prozent ändern.

Die so über mehrere Bilder verfolgten Regionen werden, sobald sie das erste Mal erscheinen, mit einem Beweglichkeitswert $\hat{b}_{Region}$ versehen, der angeben soll, zu wieviel Prozent sich die Pixel einer Vordergrundregion über eine gewisse Zeitspanne (z.B. 10 Sekunden) veränderen. Um die Veränderung der Pixel auszurechnen wird die Anzahl der Pixel, die vom einen auf das andere Bild zu der Region hinzugekommen sind, zu der Anzahl der Pixel addiert, die nicht mehr zu der Region gezählt werden. Um die Prozentzahl b_{Region} zu erhalten, wird diese Summe durch die aktuelle Größe der Vordergrundregion geteilt.

Der aus den zwei letzten Bildern erhaltene Beweglichkeitswert b_{Region} wird genutzt, um die über eine längere Zeitspanne geschätzte Beweglichkeit $\hat{b}_{Region}$ mit Hilfe des Lernparameters α zu aktualisieren:

$$\hat{b}_{Region(aktuell)} = (1 - \alpha) * \hat{b}_{Region} + \alpha * b_{Region} \tag{4.2}$$

Dies stellt wieder einen rekursiven Filter dar. Wie schon im vorigen Kapitel angesprochen, wird im Anhang A.3 auf das Verhalten dieser Filterung eingegangen. Es sei wieder nur angemerkt, dass kleine Werte für den Lernparameter α wie 0.001 zur Berechnung von $\hat{b}_{Region}$ sehr viel tiefer in die Vergangenheit reichende Beweglichkeitswerte b_{Region} miteinbezieht als relativ grosse Werte für α wie 0.25.

Mit diesen geschätzten Beweglichkeitswerten $\hat{b}_{Region}$ für die Vordergrundregionen werden diese als leblose Regionen klassifiziert, wenn ihr $\hat{b}_{Region}$ unter einen Schwellwert wie 0.01 sinkt, während Objekte, deren $\hat{b}_{Region}$ über diesem Schwellwert liegt, als 'lebendig' klassifiziert werden.

Es ist wichtig, Vordergrundregionen, wenn sie zum ersten Mal im Bild erscheinen, die Gelegenheit zu geben, sich als lebendig zu beweisen ohne gleich am Anfang als 'leblos' klassifiziert zu werden. Deswegen versieht man sie beim ersten Auftreten mit einem relativ hohen initialen Beweglichkeitswert wie 0.2. So werden die Regionen wie angedeutet nicht gleich am Anfang als 'leblos' angesehen und zum Hintergrund adaptiert. Die 'lebendigen' Regionen bleiben im Vordergrund erhalten, da sie ja gerade nicht adaptiert werden.
Die Abbildung 4.3 zeigt den Nutzen des 'Liveliness-Operators'. Im linken Bild ist die Ausgangssituation zu sehen, im mittleren Bild ist der Stuhl rechts der erkannten Personenregion auf die linke Seite der Person verschoben worden. Man sieht wie das Verschieben des Stuhls eine Phantomvordergrundregion für den verschwundenen Stuhl und ein Teil[1] des nun links stehenden Stuhls eine Vordergrundregion gebildet hat. Im rechten Bild wird die Personenregion offenbar als lebendig angesehen und gehört zum Vordergrund, während die Phantomregion und die Stuhlregion schon als 'leblose' Bereiche in den Hintergrund adaptiert worden sind.

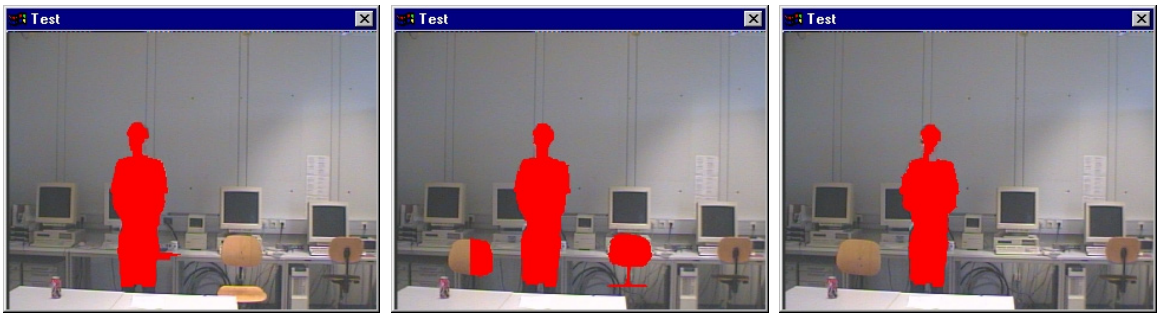

Abbildung 4.3: Effekt des Liveliness-Operators: Lebendige Objekte bleiben Vordergrund, während leblose Objekte langsam in den Hintergrund adaptiert werden. Links: Ausgangsbild. Mitte: Verschobener Stuhl. Rechts: Stuhlregionen sind Hintergrund, während die Person im Vordergrund bleibt.

4.3 Vergleich des Ein-Gaußglockenansatzes mit dem Mixturansatz

Der Vergleich der beiden Hintergrundmodelle wurde nur grob und daher qualitativ durchgeführt und soll nur einen Eindruck für die Leistungsfähigkeit des

[1]Der Algorithmus bearbeitete nicht das Gesamtbild, sondern nur einen in der Mitte zentrierten Ausschnitt, so dass der Stuhl nur teilweise als Vordergrund detektiert wurde.

jeweiligen Ansatzes geben.

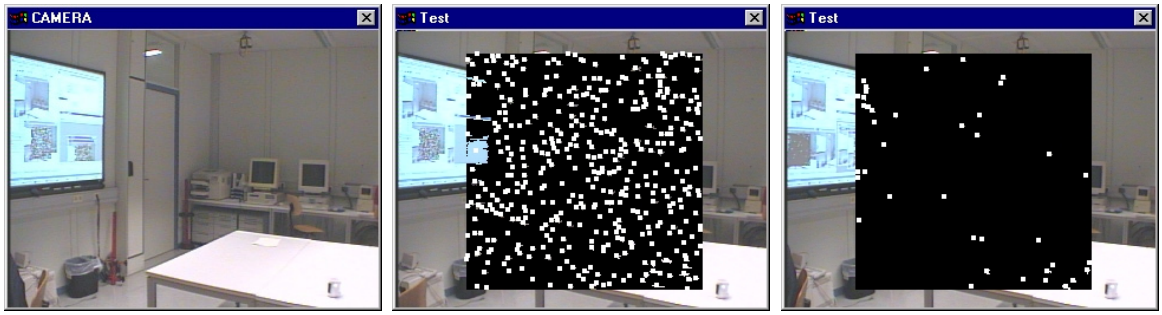

Abbildung 4.4: Der Ein-Gaußglockenansatz (Mitte) weist ein viel stärkeres Grundrauschen auf als der Mixturansatz (rechts) bei gleichem Lernparameter α für die Adaptionsgeschwindigkeit und einem eigentlich aus Hintergrund bestehendem Bild (links).

Anhand der realisierten Hintergrundmodelle konnte man feststellen, dass das Ein-Gaußmodell zwar schneller als der Mixturansatz war (ca. 5 FPS gegenüber ca. 3 FPS bei 100x100 Pixeln auf einem Intel Celeron 400MHZ) aber vor allem empfindlicher auf Verschattung und Beleuchtungsänderungen reagierte. Man musste die Parameter des Ein-Gaußglockenansatzes genauer einstellen, um ähnliche Ergebnisse zu erzielen wie der Mixturansatz.
Einen Eindruck für die höhere Anfälligkeit des Ein-Gaußglockenansatzes mag die Abbildung 4.4 geben. Im linken Bild sieht man das ursprüngliche Bild, das fast völlig aus Hintergrund besteht, im mittleren Bild ist das Ergebnis des Ein-Gaußglockenansatzes ohne Nachverarbeitung wiedergegeben und im rechten Bild wird das Klassifikationsergebnis des Mixturansatzes ebenfalls ohne Rauschen unterdrückende Nachbearbeitung gezeigt.

Wären die beiden Verfahren perfekt, würden keine Vordergrundregionen bzw. Vordergrundpixel detektiert werden. Die Bilder wären beide völlig schwarz. Da aber wegen 'Rauscheinflüssen' immer einige Vordergrundpixel detektiert werden, kommt es zum Grundrauschen. In den beiden Abbildungen wird deutlich, dass zum einen ein Grundrauschen bei beiden Ansätzen existiert und dass der Ein-Gaussglockenansatz bei gleichen Lernparametern ein sehr viel stärkes Grundrauschen besitzt als der Mixturansatz.

Kapitel 5

Softwaredesign und Implementationsdetails

In diesem Kapitel wird eine kurze Übersicht über das verwendete Softwaredesign gegeben, welches das Ziel hat, die beiden Hintergrundmodellierungsverfahren effizient zu implementieren und vor allem einen Rahmen zu schaffen, um weitere Verfahren ohne großen Aufwand realisieren zu können. Die genaue Dokumentation dieses Rahmens und auch der einzelnen Implementationsdetails ist in den der Klassenbibliothek beiliegenden HTML-Seiten nachzulesen. Diese Dokumentation wurde mit Hilfe des Dokumentationssystems 'doxygen' erzeugt. Der Quelltext des Projekts wurde mit dem Versionskontrollsystem 'cvs' verwaltet und die Algorithmen in C++ unter Linux und Windows NT implementiert und getestet.

5.1 Der objektorientierte Entwurf

Die Aufgabe, die beiden Hintergrundmodellierungsverfahren zu realisieren, wurde mit Hilfe des objektorientierten Softwareentwurfs gelöst. Dazu wurde zuerst die Interaktion zwischen einer Bildfolge und einem Hintergrundmodell aufbauenden Verfahren durch das 'Beobachter' Entwurfsmuster modelliert (vgl. [Gamma et al., 1995]).

Die Bildfolge wird durch das Subjektobjekt (KAVideoSeqSource) repräsentiert, dass auf irgendeine Weise die eigentlichen Bilddaten beschafft. Die Hintergrundmodell bildenden Verfahren werden als Beobachterobjekte modelliert. Die Beobachterobjekte (KAObserver) können sich mittels der 'attach' Methode bei dem Bildfolgenobjekt anmelden. Die 'detach' Methode ermöglicht die Abmeldung. Erlangt das Bildfolgenobjekt ein neues Bild, informiert es alle angemeldeten Beobachterobjekte mittels der 'update' Methode, dass ein neues Bild zur Verfügung steht und übergibt dabei eine Referenz auf dieses Bild (vgl. push Strategie in [Gamma et al., 1995]). Die Abbildung 5.1 zeigt das UML Klassendiagramm der eben beschriebenen Klassen.

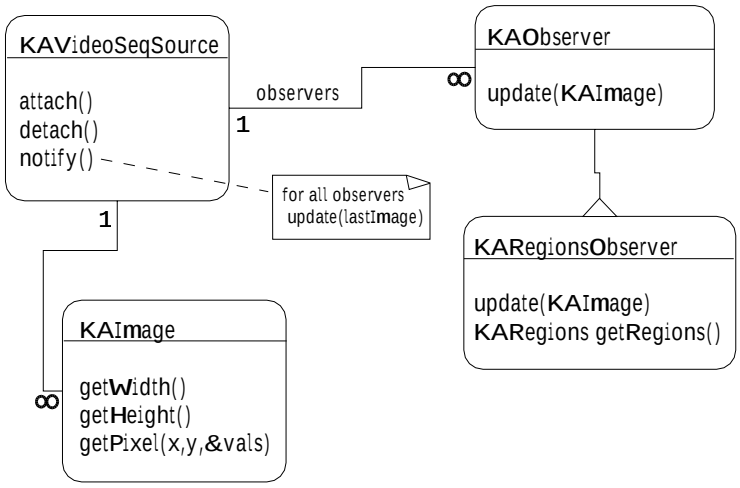

Abbildung 5.1: KAVideoSeqSource und KAObserver

Um von verschiedenen Bildformaten zu abstrahieren, werden Bilder durch die Klasse KAImage gekapselt. Diese Klasse ermöglicht den Beobachterobjekten über die Methode 'getPixel' den Farbwert eines Pixels an der Stelle (x, y) zu erfahren. Außerdem werden durch diese Klasse die allgemeinen Informationen Breite, Höhe des Bildes und der verwendete Farbraum bereitgestellt.

Die einzelnen Farbräume, die ein Bild nutzen kann, werden durch die Klasse KAColor abstrahiert. Die Methode 'getColorDimension' gibt an, wieviel Komponenten ein Farbvektor in diesem Farbraum aufweist. Die einzelnen Farbkomponenten sind alle auf ein Intervall von 0 bis 1 normiert. Dies dient dazu, dass einzelne Beobachterobjekte bei der Wahl initialer Varianzen für Farbwertverteilungen einen festen Wert wie 0.1 annehmen können.

In Kapitel 3 wurden die Nachverarbeitungsschritte wie das Finden von zusammenhängenden Komponenten und morphologischen Operatoren angesprochen, um die endgültigen Vordergrundobjekte zu erhalten. Diese Operationen sind in den Klassen KACCompOp und KAMorphoOp umgesetzt worden. Der in Kapitel 4 erläuterte 'Liveliness-Operator' findet sich in der Klasse KALivingOp wieder. Alle diese Operatoren arbeiten mit Binärbildern, die durch die Klasse boolImageT repräsentiert werden. In solch einem Binärbild gelten alle Pixel mit einem Wert $\neq 0$ als Vordergrund und Pixel mit Wert 0 als Hintergrund.

Die eigentlich gesuchten Vordergrundregionen werden durch die Klasse KARegions realisiert. Sie enthält nicht nur ein Binärbild, in dem die einzelnen Vordergrundregionen gespeichert sind[1], sondern auch die Zusatzinformationen, welche 'Bounding Box', welchen Schwerpunkt die Regionen aufweisen und außerdem welche 'Lebendigkeit' sie besitzen. Die letzte Information ist für den 'Liveliness-Operator' aus Kapitel 4 wichtig.

[1] Ein Pixel mit Wert 0 in diesem Bild gehört zum Hintergrund. Zur ersten Vordergrundregion gehören alle Pixel des Binärbilds mit einem Wert von 1, zur zweiten Region alle Pixel mit Wert 2 usw.

Um die Informationen über Vordergrundregionen einheitlich abfragen zu können, standardisiert die Klasse KARegionsObserver den Zugriff auf diese Informationen über die Methode 'getRegions'.

5.2 Das implementierte System

Zuerst wurden die Algorithmen unter Linux mit dem freien GNU C++ Compiler (g++) entwickelt und getestet. Dazu wurden zwei Bildfolgenklassen KAGrabSrc und KAFileSrc realisiert, die im ersten Fall die Bilddaten via 'video4linux API' direkt von einer Videograbberkarte liest (BTTV) und im zweiten Fall Bilddaten aus einer Bildfolgendatei (hauseigenes Format) einliest. Die beiden in Kapitel 3 beschriebenen Algorithmen sind in den Klassen KAJohnObserver (Ein-Gaußglocken Modell) und FAStaufferObserver (Mixturmodell) umgesetzt worden.

Um die erzielten Ergebnisse zu visualisieren, wurde die Simple Direct Media Layer[2] und die Magick++ Bibliothek[3] verwendet. Die Benutzung der Standard Template Library[4] vereinfachte die Implementierung vieler Teile des Systems deutlich.

Das Programm für die Demonstration der Algorithmen wurde unter Windows NT erstellt. Dazu wurde eine Bibliothek zur Bildverarbeitung, die am ILKD Waibel entwickelt wird, genutzt, um ein Bildfolgenobjekt zu erzeugen (KAWinGrabSrc), das per 'Video for Windows API' eine einfache Grabberkarte anspricht, um Bilddaten von einer Kamera zu gewinnen. Für die Visualisierung der Ergebnisse und die Benutzeroberfläche wurde neben der eben erwähnten Bibliothek noch direkt auf die 'Win32 API' zurückgegriffen. Die eigentlichen Algorithmen und Nachbearbeitungsoperatorenklassen ließen sich ohne Probleme unter Windows NT neu kompilieren. Dies zeigt, dass der oben erläuterte Entwurf weitgehendst Plattform unabhängig ist.

[2] SDL 1.1, www.devolution.com/ slouken/SDL
[3] ftp.simplesystems.org
[4] www.sgi.com/Technology/STL/

Kapitel 6

Zusammenfassung und Ausblick

In dieser Arbeit wurden zwei Verfahren vorgestellt, um Vordergrundregionen aus Bildfolgen zu extrahieren. Beide Algorithmen basieren auf Hintergrundmodellen auf Pixelebene. Die implementierten Algorithmen arbeiten in Echtzeit und können daher als Basisbausteine für Videoüberwachungssysteme oder Mensch-Maschine Dialogsysteme eingesetzt werden. Um die beiden Verfahren für die Personenverfolgung nutzbar zu machen, wurde ein spezieller Nachverarbeitungsschritt implementiert ('Liveliness-Operator'), der weitgehend verhindert, dass Personen, selbst wenn sie längere Zeit an einem Ort stehen bleiben, dem Hintergrund zugeordnet werden.

An einigen Beispielen und am Demonstrationsprogramm konnten die verschiedenen Probleme dieser beiden Ansätze wie abrupte Lichtänderung, Schatten und geringer Kontrast von Vordergrundobjekten beobachtet werden. Der Einsatz chromatischer Farbräume kann diese Probleme mindern. Der implementierte rg-Farbraum scheint aber nicht geeignet zu sein, da seine Benutzung zwar gerade die Detektion von Schatten als Vordergrund vermindert, aber zusätzliche Artefakte erzeugt. So wäre es in der Zukunft wünschenswert den HSV-Farbraum zu nutzen. Darüber hinaus könnte auch der Farbraum aus [Horprasert et al., 1999] realisiert werden, der speziell das Schattenproblem zu lösen sucht.

Sicher wäre es interessant weitere Hintergrundmodellierungsverfahren zu implementieren wie zum Beispiel das Verfahren in [Elgammal et al., 1999], das auf einer nicht parametrischen Schätzung der Farbwerte eines Pixels in einer Bildfolge beruht.
Eigene Tests und Vergleiche können erst Aufschluss über die Leistungsfähigkeit eines Verfahrens im jeweiligen Anwendungsfall geben. Daher wurden die Klassen, in der die beiden Verfahren implementiert wurden, mit Bedacht entworfen und mit dem Dokumentationssystem 'doxygen' dokumentiert, um solche Neuimplementation möglichst einfach zu machen, ohne schon gelöste Teilprobleme nochmals realisieren zu müssen.

Die Hintergrundmodellierungsverfahren sind nur ein erster Schritt, um zum Beispiel das Problem der Personenverfolgung zu lösen.

Der nächste Schritt wäre es, die hier implementierten Verfahren zu nutzen, um die Vordergrundobjekte direkt zu modellieren. Anhand der Silhouette einer Person, die über mehrere Einzelbilder verfolgt wird, könnte ein Farbmodell für den Kopf-, Oberkörper- und Unterkörperbereich geschätzt werden. Ein solches einfaches Personnenmodell ist zum Beispiel in [Wren et al., 1999] beschrieben. Wie in [Wren et al., 1999] gezeigt, können so Personen verfolgt und deren Verhalten interpretiert werden, um Maschinen bzw. virtuelle Agenten in einem intelligenten Raum zu steuern.

Anhang A

Adaptionstechniken für statistische Hintergrundmodelle

Wie in Kapitel 3 erläutert wurde, arbeiten einige Verfahren mit der Hypothese, dass die Hintergrundfarbwerte eines einzelnen Pixels über die Zeit betrachtet einer Gaußverteilung gehorchen.
Eine Gaußverteilung wird durch ihren Erwartungswertvektor μ und ihre Kovarianzmatrix Σ charakterisiert; es reicht daher diese beiden Parameter aufgrund der beobachteten Farbwerte eines Pixels (oder allgemeiner aufgrund der Werte einer Lernstichprobe) zu schätzen, um die eigentliche Verteilung zu erlangen.
Eine Methode dies zu tuen, ist die Maximum-Likelihood Schätzung. Dort wird angenommen, dass die zu schätzenden Parameter einen wahren festen Wert haben, aber unbekannt sind. Der beste Schätzwert der Parameter wird als derjenige angesehen, der die Wahrscheinlichkeit maximiert, die beobachteten Werte der Lernstichprobe zu erhalten.

In den nächsten beiden Abschnitten A.1 und A.2 werden zum einen grundlegende Eigenschaften der Gaußverteilung beschrieben und zum anderen die Maximum-Likelihood Schätzung erläutert, um aufgrund einer Lernstichprobe, d.h. z.B. beobachtete Farbwerte an einem Pixel, die Parameter einer Zufallsverteilung wie der Normalverteilung zu schätzen. Diese beiden Abschnitte basieren auf Kapitel 3 'Parameter Estimation and Supervised Learning' des Buches von Duda & Hart [Duda and Hart, 1973].

Im Abschnitt A.3 wird auf das Zeitverhalten eines in dieser Arbeit an vielen Stellen genutzen rekursiven Filters eingegangen.

Der Abschnitt A.4 erläutert morphologische Operatoren, die im Kapitel 3 beschriebenen Nachverarbeitungsschritt genutzt werden.

A.1 Gaußverteilung und ihre Eigenschaften

Um langsam zu beginnen, wird erst die eindimensionale Gaußverteilung betrachtet:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right] \tag{A.1}$$

deren Erwartungswert μ und Varianz σ^2 ist. Diese Verteilung ist offenbar durch diese beiden Parameter vollständig bestimmt. Die Werte einer Zufallsvariable, die normal verteilt ist liegen meist nahe am Erwartungswert, wobei 'nahe' von der Standardabweichung σ festgelegt wird; mehr als 68% der Zufallswerte liegen im Intervall $|x-\mu| \leq 2\sigma$, cirka 95% in 2.5σ Entfernung und bei 3σ schon mehr als 99% der Werte.

Die mehrdimensionale Gaußverteilung hat folgende Form:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu)^t\Sigma^{-1}(\mathbf{x}-\mu)\right] \tag{A.2}$$

dabei ist $\mathbf{x}$ ein Vektor der Dimension d, μ der d-dimensionale Erwartungswertvektor, Σ die $d \times d$ Kovarianzmatrix. Für diese Verteilung gilt

$$\mu = E[\mathbf{x}] \tag{A.3}$$

und

$$\Sigma = E[(\mathbf{x}-\mu)(\mathbf{x}-\mu)^t]. \tag{A.4}$$

Die Einträge auf der Diagonalen σ_{ii}^2 von Σ geben die Varianzen der Komponenten x_i von $\mathbf{x}$ an und Einträge auf der Nebendiagonalen σ_{ij}^2 die Kovarianzen zwischen x_i und x_j.

Sind die einzelnen Komponenten des Zufallsvektors $\mathbf{x}$ stochastisch unabhängig, dann sind die Kovarianzen $\sigma_{ij}^2 = 0$ und Σ wird zur einer Diagonalmatrix.

Man kann zeigen, dass eine lineare Transformation des Zufallsvektors $\mathbf{x}$ wie zum Beispiel eine Drehung des Koordinatensystems wieder Zufallsgrößen erzeugt, die normal verteilt sind. Etwas genauer, ist $\mathbf{x}$ normal verteilt mit Erwartungsvektor μ, Kovarianzmatrix Σ und ist A eine $d \times n$ Matrix, dann ist $\mathbf{y} = A^t\mathbf{x}$ ein normalverteilte Zufallsgröße mit Erwartungsvektor $A^t\mu$ und Kovarianzmatrix $A^t\Sigma A$.

Mit der Beobachtung, dass Kovarianzmatrizen Σ immer symmetrisch und reell sind, folgt, dass durch eine Drehung des Koordinatensystems Σ diagonalisiert werden kann. Diese Drehung kann durch eine $d \times d$ Matrix A beschrieben werden. Dann ist $\mathbf{y} = A^t\mathbf{x}$ wieder normal verteilt. Nur ist die Kovarianzmatrix $\Sigma_\mathbf{y} = A^t\Sigma_\mathbf{x}A$ für die Größe $\mathbf{y}$ diagonal, was nichts anderes bedeutet als dass die Komponenten des Zufallsvektors $\mathbf{y}$ stochastisch unabhängig voneinander sind.

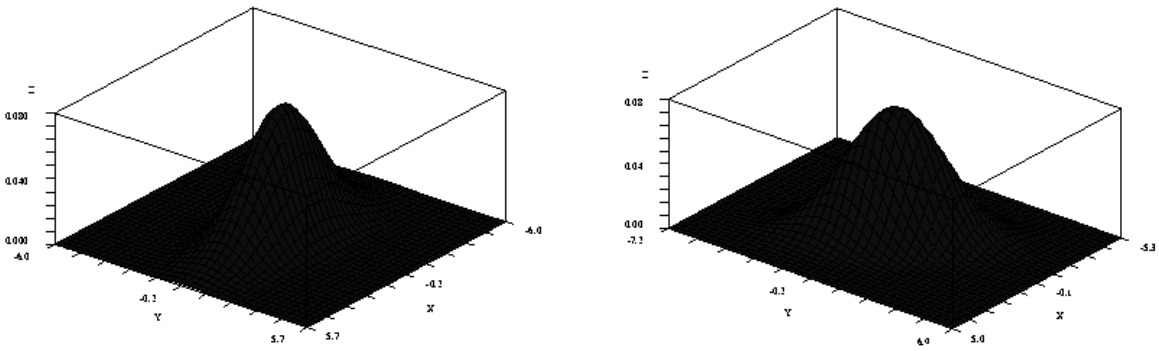

Abbildung A.1: Zwei dimensionale Gaußverteilungen

Es kann daher immer das Koordinatensystem so gedreht, dass die einzelnen Komponenten des Vektors $\mathbf{x}$ stochastisch unabhängig voneinander sind.

In der Abbildung A.1 ist ein grafisches Beispiel für das eben Gesagte zu sehen. Die Gaußverteilung links hat $\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ und $\Sigma_1 = \begin{pmatrix} 4 & 0 \\ 0 & 1 \end{pmatrix}$. Es wird deutlich, dass die x-Werte doppelt so stark wie die y-Werte streuen ($\sigma_x = 2$,$\sigma_y = 1$).

Die Verteilung rechts hat das gleiche μ aber ein anderes $\Sigma_2 = \begin{pmatrix} 1,75 & -1,299 \\ -1,299 & 3,25 \end{pmatrix}$. Dieses kann aber durch eine Drehung um $\pi/3$ in $\Sigma_1 = A^t \Sigma_2 A$ überführt werden, wobei $A = \begin{pmatrix} cos(\pi/3) & -sin(\pi/3) \\ sin(\pi/3) & cos(\pi/3) \end{pmatrix}$.

Die Abbildung A.1 zeigt auch, dass Punkte, die den gleichen euklidischen Abstand vom Erwartungswert haben, nicht die gleiche Wahrscheinlichkeitsdichte aufweisen müssen. Will man aber erreichen, dass Punkte, die den gleichen Abstand vom Erwartungswert aufweisen, auch die gleiche Wahrscheinlichkeitsdichte haben, so muss als Abstandsmaß die Mahalanobisdistanz r gewählt werden:

$$r^2 = (\mathbf{x} - \mu)^t \Sigma^{-1} (\mathbf{x} - \mu). \tag{A.5}$$

Man kann am Beispiel der linken Gaußverteilungen in der obigen Abbildung A.1 sich dieses Maß verdeutlichen, dass für die Mahalnobisdistanz die x-Werte halb so stark gewichtet werden, wie die y-Werte, da die x-Werte ein $\sigma_x = 2$, die y-Werte aber nur $\sigma_y = 1$ haben. Der Term Σ^{-1} in der Mahalanobisdistanz wird ja zu $\Sigma^{-1} = \begin{pmatrix} \frac{1}{2^2} & 0 \\ 0 & 1 \end{pmatrix}$, was die genannte ungleiche Gewichtung bewirkt.

Abschließend muss mit Hinblick auf den nächsten Abschnitt erwähnt werden, durch wieviele Parameter eine d-dimensionale Normalverteilung charakterisiert wird. Der Erwartungswertvektor hat d Parameter und die Kovarianzmatrix

hat wegen ihrer Symmetrie $d(d+1)/2$ Parameter, was zusammen für eine d-dimensionale Normalverteilung $d(d+3)/2$ unabhängige Parameter ergibt.

A.2 Maximum-Likelihood Schätzung

Die Hintergrundfarbwerte eines Pixels einer Bildfolge können als zufallsverteilt modelliert werden (ob dieses Modell sinnvoll ist, wird hier nicht besprochen). Mit anderen Worten diese Farbwerte können als unabhängige Menge von Stichprobenwerten angesehen werden, die nach einer bestimmten Zufallsverteilung $p(\mathbf{x})$ gezogen wurden. Diese Zufallsverteilung kann durch einige Parameter exakt bestimmt sein, die zu einem Parametervektor θ zusammengefasst werden können. Man schreibt dann auch $p(\mathbf{x}|\theta)$, um die Abhängigkeit deutlich zu machen.
Für diesen Fall reduziert sich das Problem auf das Schätzen des Parametervektors θ, um die angenommene Zufallsverteilung der Stichprobe möglichst adäquat wiederzugeben.

Ein Verfahren, um dies zu erreichen, ist die Maximum-Likelihood Schätzung. Angenommen die Stichprobe der einzelnen Hintergrundfarbwerte sei $\mathcal{S} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. Da die einzelnen $\mathbf{x}_i$ unabhängig voneinander gemäß $p(\mathbf{x}|\theta)$ 'gezogen' wurden, ist

$$p(\mathcal{S}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta). \tag{A.6}$$

Betrachtet als Funktion von θ, wird $p(\mathcal{S}|\theta)$ als Mutmaßlichkeit (likelihood) bezeichnet. Der Maximum-Likelihood Schätzer $\hat{\theta}$ ist der Wert von θ, der $p(\mathcal{S}|\theta)$ maximiert.

Für Hintergrundmodelle, die in dieser Arbeit besprochen werden, ist vor allem der Fall einer mehrdimensionalen Gaußverteilung von Nutzen. Wie man den Maximum-Likelihood Schätzer für den eindimensionalen Fall herleitet, kann man in [Duda and Hart, 1973] im Kapitel 3.2 nachlesen. Im Prinzip wird die Normalverteilung nach θ abgeleitet und der Maximum-Likelihood Schätzer als Nullstelle der Ableitung gefunden. Das Buch [Duda and Hart, 1973] gibt auch an, wo die Herleitung für den hier interessierenden mehrdimensionalen Fall zu finden ist [Anderson, 1958]. Der Maximum-Likelihood Schätzer für den Fall einer mehrdimensionalen Gaußverteilung lautet danach:

$$\hat{\mu} = \frac{1}{n} \sum_{k=1}^{n} \mathbf{x}_k \tag{A.7}$$

und

$$\hat{\Sigma} = \frac{1}{n} \sum_{k=1}^{n} (\mathbf{x}_k - \hat{\mu})(\mathbf{x}_k - \hat{\mu})^t. \tag{A.8}$$

Für die Anwendung bei Hintergrundmodellen ist es ratsam, diese Formeln so umzuschreiben, dass neue Stichprobenwerte verrechnet werden ohne alle bisherigen Werte nochmals betrachten zu müssen. In [Duda and Hart, 1973] findet man in der Aufgabe 9 des Kapitels 3:

$$\hat{\mu}_{n+1} = \hat{\mu}_n + \frac{1}{n+1}(\mathbf{x}_{n+1} - \hat{\mu}_n) \tag{A.9}$$

und

$$\hat{\Sigma}_{n+1} = \frac{n-1}{n}\hat{\Sigma}_n + \frac{1}{n+1}(\mathbf{x}_{n+1} - \hat{\mu}_n)(\mathbf{x}_{n+1} - \hat{\mu}_n)^t. \tag{A.10}$$

$\mathbf{x}_{n+1}$ ist der Stichprobenwert (ein neu beobachteter Farbwert eines Pixels), um den $\mathcal{S}$ ergänzt wird. So kann sich ohne großen Rechenaufwand die Gaußverteilung immer besser an die mit der Zeit wachsende Stichprobe anpassen.

Allerdings werden alle bisher beobachteten Werte in die Parameterschätzung miteinbezogen. Oft will man aber nur die Werte berücksichtigen, die höchstens eine gewisse Zeit zurückliegen. Man muss quasi ältere Stichprobenwerte 'vergessen'.

Exakt aber unpraktisch ist es, alle einzubeziehenden Stichprobenwerte in einer Schlange zu speichern und dann beim Einfügen eines neuen Wertes den ältesten zu entfernen, so kann man die Parameter genau schätzen; aber man verschwendet auch viel Speicherplatz.

Praktikabler aber von der oben geäußerten Idee abweichend ist es den Beitrag des neuen Stichprobenwertes mit fester 'Lernrate' in die Schätzung miteinzubeziehen, was in den Formeln

$$\hat{\mu}_{n+1} = (1 - \alpha_\mu)\hat{\mu}_n + \alpha_\mu(\mathbf{x}_{n+1} - \hat{\mu}_n) \tag{A.11}$$

und

$$\hat{\Sigma}_{n+1} = (1 - \alpha_\Sigma)\hat{\Sigma}_n + \alpha_\Sigma(\mathbf{x}_{n+1} - \hat{\mu}_n)(\mathbf{x}_{n+1} - \hat{\mu}_n)^t \tag{A.12}$$

resultiert.

A.3 Zeitverhalten der rekursiven Filterung

Um besser zu verstehen, wie der eben besprochene Ansatz, der auch auch zur Schätzung anderer Größen in dieser Arbeit verwendet wird, vom exakten Ausrechnen der Parameter über ein 'Zeitfenster' abweicht, kann man in Abbildung A.2 sehen, wie stark vorige Stichprobenwerte, die augenblickliche Parameterschätzung für verschiedene Lernraten beeinflussen. Dazu sind in der Abbildung auf der y-Achse die Gewichtungen der vorigen und des jetzigen Stichprobenwertes aufgetragen. Man sieht, dass bei niedriger Lernrate, die noch miteinbezogenen Werte viel weiter in der Vergangenheit zurückliegen als bei höheren Lernraten.

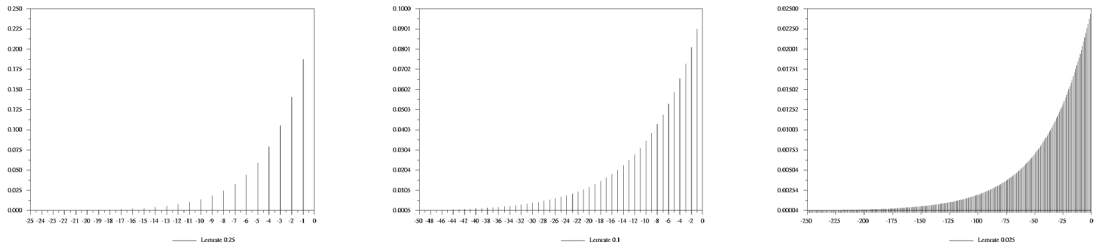

Abbildung A.2: Einfluss voriger Stichprobenwerte auf die Parameterschätzung

Man kann das 'Fenster' der noch miteinbezogenen Stichprobenwert wie folgt verstehen.
Addiert man die Gewichte der einzelnen Stichprobenwerte, die sich durch eine gewisse Lernrate α ergeben, vom aktuellen, dem 0ten, bis zu einem bestimmten Wert, dem $n$ten, auf, so sagt diese Summe $\mathcal{T}_n$ aus, zu welchem Anteil die Stichprobenwerte zwischen dem 0ten und dem nten die Schätzung insgesamt beeinflussen:

$$\mathcal{T}_n = \sum_{i=0}^{n} \alpha * (1-\alpha)^(i) = 1 - (1-\alpha)^{n+1} \tag{A.13}$$

Wenn der Anteil der Stichprobenwerte zwischen 0 und n bei 99.9% oder mehr liegt, kann man unter bestimmten Voraussetzungen davon ausgehen, dass die übrigen Stichprobenwerte jenseits von n, die Parameterschätzung vernachlässigbar beeinflussen. So kann man das 'Fenster' der noch miteinbezogenen Stichprobenwerte verstehen, als die Werte vom aktuellen bis zu diesem $n$ten Wert. Löst man $\mathcal{T}_n \geq 0.999$ nach n auf erhält man:

$$n \geq \frac{\log(1-0.999)}{\log(1-\alpha)} - 1. \tag{A.14}$$

Die Abbildung A.3 zeigt diese Beziehung grafisch; $n+1$ ist dort als die Breite des 99.9%-Fensters aufgetragen.

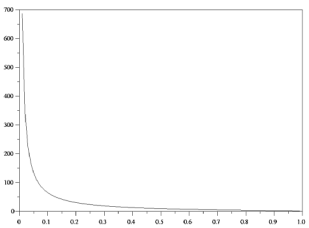

Abbildung A.3: Breite des 99.9%-'Fenster' in Abhängigkeit von der Lernrate

A.4 Morphologische Operatoren

Morphologische Operatoren kann man nutzen, um Löcher in Vordergrundobjekten zu schließen oder schärfere Umrandungen von Vordergrundobjekten (Silhoutten) zu erhalten.

Die beiden morphologischen Grundoperationen auf denen alle anderen morphologischen Operatoren aufbauen sind die Dilatation und Erosion. Bildlich gesprochen wird bei der Dilatation eine z.B. 3×3 Maske auf jedes Pixel eines Binärbildes gelegt (genauer der Mittelpunkt der Maske kommt auf dem Pixel zu liegen). Sei dieses Pixel an der Stelle (x, y). Liegt einer der 9 Maskenpunkte auf einem Pixel des Binärbildes, das gleich 1 ist, so wird das Pixel an (x, y) auch auf 1 gesetzt. Bei der Erosion hingegen bleibt das Pixel an der Stelle (x, y) nur dann auf 1 gesetzt, wenn jeder der 9 Maskenpunkte auf einem Binärbildpixel liegt, das gleich 1 ist. Sonst wird das Pixel an (x, y) zu 0. Klarer verständlich werden diese Operatoren in der Abbildung A.4.

Abbildung A.4: Dilatation und Erosion mit einer 3×3 Maske. Links: das Originalbild. Mitte: Ergebnis der Dilatation. Rechts: Ergebnis der Erosion.

Die Grundoperatoren können kombiniert werden. Das 'Opening' ist eine Erosion gefolgt von einer Dilatation und das 'Closing', das für das Schließen von Löchern in einer Figur interessant ist, besteht aus einer Dilatation gefolgt von einer Erosion. Eine Abbildung soll diese beiden Operatoren wieder verdeutlichen.

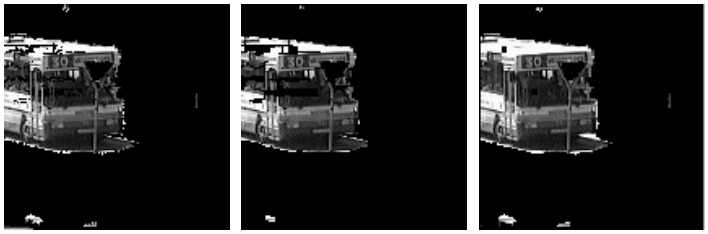

Abbildung A.5: Opening und Closing mit einer 5×5 Maske. Links: das Originalbild. Mitte: Ergebnis des Opening. Rechts: Ergebnis des Closing

Um große Löcher zu schließen, kann es ratsam sein, einen spezielle morphologische Operatoren zu definieren, die Dilatation und Erosion etwas anders durchführen. Die Maske dieses speziellen Operators besitzt Nullen und Einsen. Die Dilatationsoperation geht fast wie die Standarddilatation vor; aber der Pixel (x, y), auf dem die Maske liegt, wird nur dann auf 1 gesetzt, wenn einer der 1-Maskenpunkte,

auf einer 1 liegt. Bei der speziellen Erosion, bleibt der Pixel (x, y) nur dann auf 1 gesetzt, wenn alle 1-Maskenpixel auf einer 1 liegen. Die Abbildung A.6 zeigt ein Strukturelement, was eine andere Bezeichnung für die Maske des Operators ist, das große Löcher zu schließen vermag und seine Anwendung.

Abbildung A.6: Dilatation mit einem Strukturelement. Links: das Strukturelement. Mitte: das Originalbild. Rechts: Ergebnis der speziellen Dilatation

Literaturverzeichnis

[Anderson, 1958] Anderson, T. (1958). *An Introduction to Multivariate Statistical Analysis*, chapter 3. John Wiley, New York.

[Bishop, 1995] Bishop, C. M. (1995). *Neural networks for pattern recognition*, chapter 2. Oxford University Press.

[Duda and Hart, 1973] Duda, R. O. and Hart, P. E. (1973). *Pattern Classification and Scene Analysis*, chapter 3. John Wiley & Sons.

[Elgammal et al., 1999] Elgammal, A., Harwood, D., and Davis, L. (1999). Non-parametric model for background subtraction. In *ICCV 1999, Workshop 2 Frame-Rate*.

[Francois and Medioni, 1999] Francois, A. and Medioni, G. (1999). Adaptive color background modeling for real-time segmentation of video streams. In *Proc. of International on Imaging Science, System, and Technology, 1999, 227–232*.

[Gamma et al., 1995] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns, Elements of Reusable Object-Oriented Software*. Addison Wesley Publishing Company.

[Haritaoglu et al., 1998] Haritaoglu, I., Harwood, D., and Davis, L. (1998). W4: Who? when? where? what? a real time system for detecting and tracking people. *Face and Gesture Recognition*, pages 222–227.

[Horn, 1986] Horn, B. (1986). *Robot Vision*, pages 66–69, 299–333. The MIT Press.

[Horprasert et al., 1999] Horprasert, T., Harwood, D., and Davis, L. (1999). A statistical approach for real-time robust background substraction and shadow detection. In *ICCV 1999, Workshop 2 Frame-Rate*.

[Hsu et al., 1984] Hsu, Y., Nagel, H., and Rekers, G. (1984). New likelihood test methods for change detection in image sequences. *Computer Vision, Graphics, and Image Processing*, 26:73–106.

[Maes et al., 1995] Maes, P., Darrell, T., Blumberg, B., and Pentland, A. (1995). The alive system: full-body interaction with autonomous agents. In *Computer Animation'95 Proceedings, IEEE Press*, pages 11–18.

[M.Torrance, 1995] M.Torrance (1995). Advances in human-computer interaction: The intelligent room. In *Working Notes of the CHI 95 Research Symposium, May 6-7, Denver, Colorado.*

[Nagel, 1987] Nagel, H. (1987). the estimation of optical flow: relations between different approaches and some new results. *Artificial Intelligence*, 33:299–324.

[Stauffer and Grimson, 1998] Stauffer, C. and Grimson, W. (1998). Adaptive background mixture models for realtime tracking. In *Proc. of CVPR, 1998, 333–339.*

[Wren et al., 1999] Wren, C., Azarbayejani, A., Darrell, T., and Pentland, A. (1999). Pfinder: Real-time tracking of the human body. In *Photonics East, SPIE, volume 2615, 1995. Bellingham, WA.*

[Yang et al., 1999] Yang, J., Zhu, X., Gross, R., Kominek, J., Pan, Y., and Waibel, A. (1999). Multimodal people id for a multimedia meeting browser. Technical report, Interactive Systems Laboratories, Carnegie Mellon University, Pittsburgh, PA 15213, USA.