Visuelle Benutzermodellierung mit Tracking und Zeigegestenerkennung für einen humanoiden Roboter

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

der Fakultät für Informatik
der Universität Fridericiana zu Karlsruhe (TH)

genehmigte

Dissertation

von

Kai Nickel

aus Neustadt an der Weinstraße

Tag der mündlichen Prüfung:	**07.11.2008**
Erster Gutachter:	**Prof. Dr. A. Waibel**
Zweiter Gutachter:	**Prof. Dr. R. Dillmann**

Kurzzusammenfassung

Die vorliegende Arbeit befasst sich mit der visuellen Benutzermodellierung für humanoide Roboter. Dabei werden Verfahren entwickelt, um mit Mitteln der Bildverarbeitung eine komplette Perzeptionskette zu realisieren. Diese besteht aus dem räumlichen Lokalisieren des Benutzers, dem Tracking der Hände sowie der Kopfdrehung, und basierend darauf dem automatischen Erkennen menschlicher Zeigegesten.

Zur Personenlokalisierung wird ein Ansatz verfolgt, der durch Fusion mehrerer einfacher Merkmale sowohl schnell als auch robust arbeitet. Kernstück des Verfahrens ist ein Algorithmus, der die Methode der Demokratischen Integration [Triesch und Malsburg, 2001] zur dynamischen Merkmalsfusion auf partikelfilterbasiertes Tracking überträgt. Dabei geht er von einem allgemeineren Merkmalsbegriff aus, der neben den unterschiedlichen Merkmalstypen wie Farbe oder Bewegung auch die unterschiedlichen Bildregionen des Zielobjekts als gleichwertige Teilnehmer eines dynamischen Wettbewerbs betrachtet. So kann durch ein und dasselbe Verfahren sowohl der Ausfall einzelner Merkmalstypen als auch die Verdeckung einzelner Körperregionen bzw. Kameraansichten kompensiert werden.

Experimente mit einem Stereokamerakopf zeigen, dass sich durch Einsatz des Verfahrens die Anzahl der Trackingfehler um ca. 50% reduzieren lässt. Darüber hinaus wird die allgemeine Anwendbarkeit des Verfahrens demonstriert, indem es zusätzlich auch auf das Tracking in Mehrkameraumgebungen übertragen wird. Experimente mit den Daten des CLEAR'2007-Workshops zeigen mit einem Anstieg der MOTA-Werte von $88,7\%$ auf $94,0\%$ ebenfalls eine signifikante Verbesserung, die auf die dynamische Merkmalsfusion zurückzuführen ist.

Die Detektion von Zeigegesten erfolgt mit Hidden-Markov-Modellen, die mit den Hand-Trajektorien hunderter Beispiele trainiert werden. Darauf aufbauend wird untersucht, inwieweit die Beobachtung, dass Menschen beim Zeigen das Zielobjekt anblicken, zur Verbesserung der Gestenerkennung genutzt werden kann. In einem Experiment hierzu konnte die Anzahl der Fehlerkennungen durch Miteinbeziehung der Kopfdrehung um ca. 50% reduziert werden. So wird deutlich, dass Kopfdrehung tatsächlich ein wichtiges Merkmal zur automatischen Erkennung von Zeigegesten ist und dass dieser Sachverhalt auch technisch ausgenutzt werden kann. In abschließenden Experimenten mit einem multimodalen Dialogsystem nach [Holzapfel u. a., 2004] zeigt sich außerdem, dass durch die semantische Fusion von Sprache und Gestik deutlich mehr Benutzereingaben korrekt interpretiert werden können als mit einer der beiden Modalitäten allein.

Die hier entwickelten Verfahren zur Personenlokalisierung, zum Hand-Tracking und zur Zeigegestenerkennung wurden als Softwaremodule implementiert, die auf dem humanoiden Roboter ARMAR-III im Verbund mit anderen Perzeptionskomponenten in Echtzeit arbeiten. Dadurch wurde ARMAR-III als erster humanoider Roboter in die Lage versetzt, gleichzeitig und on-board mehrere Personen zu lokalisieren, ihre Gesichter zu erkennen und sprachgesteuert zu lernen und Zeigegesten in Verbindung mit sprachlichen Kommandos zu interpretieren.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Die steigende Anzahl und zunehmende Komplexität von Geräten und informationstechnischen Systemen, mit denen Menschen in ihrem Alltag konfrontiert werden, macht die Gestaltung der Mensch-Maschine-Schnittstelle zu einer immer größer werdenden Herausforderung. Der Mensch als Benutzer einer Maschine soll in die Lage versetzt werden, mit möglichst geringer Einarbeitungszeit die Funktionen der Maschine verstehen und nutzen zu können. Der Vorgang der Benutzung soll dabei eine Vielzahl von Voraussetzungen erfüllen: Er soll unter anderem der zu lösenden Aufgabe angemessen sein, den intellektuellen und körperlichen Fähigkeiten des Menschen gerecht werden, die aktuelle Situation berücksichtigen, Fehler und Ermüdung vermeiden – und nicht zuletzt dabei auch Spaß machen.

Neben konventionellen haptischen Eingabegeräten wie Tastatur, Maus, berührungsempfindlichen Bildschirmen, etc. sind inzwischen auch Schnittstellen entwickelt worden, die auf der Erkennung von Sprache basieren. Dieser neue Kommunikationskanal zwischen Mensch und Maschine erlaubt z. B. das berührungslose Benutzen von Geräten, das natürlichsprachliche Formulieren komplexer Befehle unter Umgehung hierarchischer Menüs oder das Benutzen von Kleinstgeräten ganz ohne Bildschirm oder Tastatur. Ein weiterer Kommunikationskanal, der vom Menschen intuitiv in der Kommunikation mit anderen Menschen verwendet wird, ist die visuelle Interpretation von Körpersprache (Mimik und Gestik). Durch die Verfügbarkeit von miniaturisierten und preiswerten Kameras sowie der zur Bildverarbeitung notwendigen Rechenleistung sind die Möglichkeiten gewachsen, auch die visuelle Wahrnehmung des Menschen zu einer Option bei der Gestaltung von Mensch-Maschine-Schnittstellen zu machen.

Die Forderung nach Berücksichtigung der von Menschen natürlicherweise verwendeten Kommunikationsformen betrifft in besonderer Form auch den Bereich der Robotik. Hier werden zurzeit immer komplexere mobile Roboter entwickelt, die nicht mehr nur auf umzäunte Produktionsbereiche beschränkt sind, sondern sich zusammen mit Menschen in gemeinsam genutzten Umgebungen bewegen sollen. In solchen Alltagsumgebungen gehört zur Mensch-Roboter-Kommunikation notwendigerweise auch die Fähigkeit, auf Sprache, Berührung und Gestik angemessen reagieren zu können.

Insbesondere gilt dies für so genannte humanoide Roboter, die dem Erscheinungsbild des menschlichen Körpers nachempfunden sind. Sie verfügen in der

Abbildung 1.1: Mensch-Roboter-Kommunikation per Sprache und Gestik in einer Haushaltsumgebung. Quelle: [SFB 588].

Regel über einen Torso mit Kopf, zwei Armen und zwei Beinen. Manche humanoide Roboter besitzen zudem 5-Finger-Greifer oder nachgebildete menschliche Gesichtsmerkmale wie Mund, Nase und Augenbrauen. Jedoch nicht alle diese Merkmale müssen vorhanden sein: So modellieren einige humanoide Roboter nur einen Teil des menschlichen Körpers, wie z. B. den Oberkörper, der dann auf einer fahrbaren Plattform montiert ist.

Die Motivation für die menschenähnliche Gestaltung von Robotern ist vielfältig: So verspricht die Orientierung an Ausmaßen und Freiheitsgraden des menschlichen Körpers eine gute Beweglichkeit des Roboters in Räumen, die für menschliche Nutzung eingerichtet wurden (Durchgänge, Türen, Treppen, Arbeitsflächen, etc.). Zudem ermöglicht sie es dem Roboter, für Menschen geschaffene Werkzeuge und Einrichtungen zu benutzen. Dies sind notwendige Voraussetzungen z. B. für den Einsatz in Kranken- und Pflegeeinrichtungen oder zur Unterstützung körperlich eingeschränkter Menschen im Haushalt. Neben ihrer Beweglichkeit kommt bei humanoiden Robotern ein zweiter zentraler Aspekt zum Tragen: ihre Fähigkeit zur Kommunikation mit Menschen. Die menschenähnliche Anmutung der Maschine weckt beim Benutzer Erwartungen bezüglich ihrer „Intelligenz“ bzw. ihrer Wahrnehmungs-, Lern- und Kommunikationsfähigkeiten, die es durch eine hoch entwickelte Mensch-Maschine-Schnittstelle zu erfüllen gilt.

Humanoide Roboter werden zum heutigen Zeitpunkt außer zu Werbezwecken nicht für produktive Tätigkeiten eingesetzt oder angeboten. Obwohl ihre Entwicklung vor allem in Japan auch unter kommerziellen Gesichtspunkten vorangetrieben wird – siehe hierzu bspw. [Dilba und Kölling, 2007] –, handelt es sich bei humanoiden Robotern noch um einen reinen Forschungsgegenstand. Ein solches Forschungsprojekt ist der im Jahr 2001 in Karlsruhe ins Leben gerufene Sonderforschungsbereich 588 „Humanoide Roboter - Lernende und kooperierende multimodale Roboter“ der Deutschen Forschungsgemeinschaft (DFG). Er

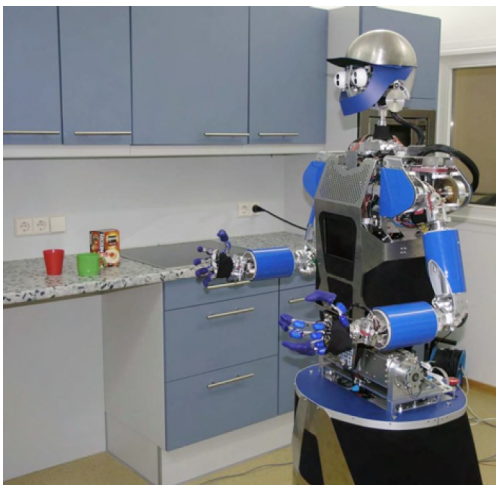


Abbildung 1.2: Der humanoide Roboter ARMAR-III in der Experimentierküche.

hat sich zum Ziel gesetzt, „Konzepte, Methoden und konkrete mechatronische Komponenten für einen humanoiden Roboter zu entwickeln, der seinen Arbeitsbereich mit dem Menschen teilt“ [SFB 588]. Mit der Eigenentwicklung ARMAR-III, siehe [Asfour u. a., 2006] und Abbildung 1.2, verfügt der SFB 588 über ein teilhumanoides mobiles Zweiarmsystem, das als Experimentierplattform für einen humanoiden Haushaltsroboter dient. Die im Rahmen dieser Arbeit entwickelten Softwarekomponenten sind Bestandteil der visuellen Perzeption von ARMAR-III und arbeiten dort im Verbund mit anderen Modulen aus den Bereichen Perzeption, Kognition und Handlungsausführung.

1.1 Problemstellung und Ziele

Eine grundlegende Voraussetzung zur Interaktion und Kommunikation ist das räumliche Lokalisieren und Verfolgen von Menschen, im Folgenden auch als *Tracking* bezeichnet. Dadurch kann der Roboter den Sensorkopf mit seinem begrenzten Sichtfeld auf den jeweiligen Benutzer ausrichten. Der Roboter signalisiert dem Benutzer dadurch zum einen seine Aufmerksamkeit, zum anderen ist das Nachführen der Kameras Voraussetzung für weitergehende visuelle Analysen über Körperhaltung oder Gestik des Benutzers. Menschen benutzen unter anderem Zeigegesten, um die Aufmerksamkeit ihres Gegenübers auf einen bestimmten Punkt zu lenken. Durch die automatische Erkennung von Zeigegesten eröffnet sich ein intuitiver Kommunikationskanal zum Roboter, mit dem der Aufmerksamkeitsfokus des Roboters vom Benutzer aktiv beeinflusst werden kann, um z. B. Richtungen oder Objekte im Raum zu referenzieren.

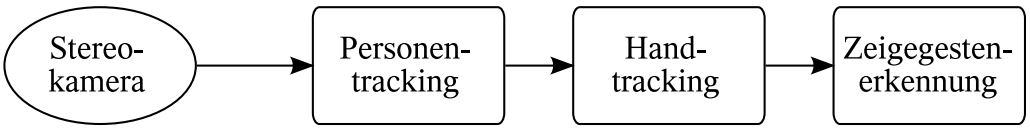


Abbildung 1.3: Übersicht der Komponenten zur visuellen Benutzermodellierung.

Im Rahmen dieser Arbeit soll ein Bildverarbeitungssystem zur Benutzermodellierung für einen humanoiden Roboter realisiert werden, das über die folgenden Fähigkeiten verfügt:

- Räumliches Lokalisieren eines oder mehrerer Benutzer sowohl im Nahbereich (ab $0,5m$ Entfernung zum Roboter) als auch im Fernbereich (bis max. $10m$)
- Räumliches Tracking beider Hände des Benutzers
- Automatische Erkennung von Zeigegesten und Bestimmung der räumlichen Zeigerichtung.

Die Verarbeitung der Bilder soll dabei in Echtzeit direkt an Bord des Roboters erfolgen, Einsatzdomäne des Systems ist eine Haushaltsumgebung mit mehreren Benutzern. Das System soll keine manuellen Eingriffe oder Initialisierung benötigen.

Beim Lokalisieren des Benutzers stellen sich insbesondere folgende Fragen, die beantwortet werden müssen:

- Innerhalb des Arbeitsbereichs $(0,5m - 10m)$ ändert sich das Abbild eines Menschen vom bildfüllenden Portrait bis hin zur niedrig aufgelösten Silhouette, die in der Szene eingebettet ist. Wie muss ein Trackingverfahren gestaltet sein, um mit dieser großen Varianz in der visuellen Erscheinung zurechtzukommen?
- Zur Modellierung von Menschen unter wechselnden Aufnahmebedingungen müssen unterschiedliche Merkmale herangezogen werden. Wie können die verschiedenen Merkmale dynamisch kombiniert werden, um den Ausfall einzelner Merkmale (z. B. bei teilweiser Verdeckung) zu kompensieren?
- Die beschränkte Rechenleistung an Bord verlangt vom Trackingverfahren, die Sensordatenauswertung auf bestimmte Bildbereiche zu beschränken, wodurch zusätzlich zum äußerlichen Aufmerksamkeitsfokus des Roboters (*overt attention*) auch ein innerlicher (*covert attention*) gegeben ist. Wie kann der Suchraum möglichst effizient eingeschränkt werden?

Bei der automatischen Erkennung von Zeigegesten stellen sich weitere Herausforderungen:

- Hände sind hochgradig bewegliche Objekte, ihr Abbild verändert sich schnell und drastisch. Wie können Hände in Echtzeit zuverlässig verfolgt werden?
- Die Ausführung von Zeigegesten variiert von Geste zu Geste und von Person zu Person. Mit welchen Modellen kann das typische Bewegungsmuster der Hand trotz dieser Varianz repräsentiert werden?
- Zeigegesten sind multimodale Phänomene, die Geste wird in der Regel von einer verbalen Äußerung begleitet und das Zielobjekt wird angeschaut. Wie können diese Beobachtungen zur Verbesserung der Zeigegestenerkennung genutzt werden?

1.2 Beiträge

Im Rahmen dieser Arbeit konnten insbesondere in drei Aspekten Fortschritte im Hinblick auf den Stand der Technik erzielt werden. Diese sind im Folgenden kurz zusammengefasst.

Das Personentracking folgt einem partikelfilterbasierten Ansatz. Zur dynamischen Merkmalsfusion wird dabei das Prinzip der Demokratischen Integration nach [Triesch und Malsburg, 2001] aufgegriffen. Um Demokratische Integration in einem Partikelfilter nutzbringend einsetzen zu können, wird ein neues Qualitätsmaß zur Merkmalsgewichtung entwickelt. Der so entstehende dynamische Wettbewerb zwischen den Merkmalen wird durch eine Verallgemeinerung des Merkmalsbegriffes auch auf Körperregionen und Kameraansichten ausgedehnt, so dass ein neuartiges dynamisches Trackingverfahren entsteht, das mit ein und demselben Mechanismus sowohl Ausfälle einzelner Merkmalstypen als auch die Nicht-Beobachtbarkeit einzelner Ziele bzw. Zielregionen überbrücken kann. Dieses universell einsetzbare Verfahren wurde in [Nickel und Stiefelhagen, 2007a] und [Nickel und Stiefelhagen, 2008] veröffentlicht und in unterschiedlichen Szenarien evaluiert: Experimente mit einem Stereokamerakopf zeigen, dass sich durch Einsatz des Verfahrens die Anzahl der Trackingfehler um ca. 50% reduzieren lässt. Darüber hinaus wird die allgemeine Anwendbarkeit des Verfahrens demonstriert, indem es zusätzlich auch auf das Tracking in Mehrkameraumgebungen übertragen wird. Experimente mit den Daten des CLEAR'2007-Workshop zeigen mit einem Anstieg der MOTA-Werte von $88,7\%$ auf $94,0\%$ ebenfalls eine signifikante Verbesserung, die auf die dynamische Merkmalsfusion zurückzuführen ist.

Zur automatischen Zeigegestenerkennung werden die Hände mit einer Kombination von Farb- und Disparitätsinformationen verfolgt und die Trajektorie der zeigenden Hand mit dedizierten Hidden-Markov-Modellen für drei verschiedene Gestenphasen klassifiziert. In der vorliegenden Arbeit werden dabei erstmals

zusätzliche Informationen in Form von visuellen Kopfdrehungsschätzungen zur Klassifikation von Zeigegesten genutzt. Es wird gezeigt, dass Kopfdrehung ein wichtiges Merkmal zur Erkennung von Zeigegesten ist, und dass die kombinierte Erkennung, basierend auf Handbewegung und Kopfdrehung, der einfachen Erkennung der Handbewegung überlegen ist. Experimente mit verschiedenen Probanden zeigen eine Reduktion der Fehlerkennungsrate von durschnittlich $26,4\%$ auf $12,9\%$ durch Berücksichtigung der Kopfdrehung. Diese Ergebnisse wurden unter anderem in [Nickel und Stiefelhagen, 2007b] veröffentlicht.

Die Verfahren zum Personen- und Handtracking sowie zur Zeigegestenerkennung wurden zusammen mit einem Verfahren zur Gesichtsidentifikation [Ekenel u. a., 2007] zu einem gemeinsamen Softwaremodul zur visuellen Benutzermodellierung verschmolzen, in den humanoiden Roboter ARMAR-III integriert, und dort mit dem Modul zur natürlichsprachlichen Dialogführung [Holzapfel, 2008] verknüpft. Dadurch wurde ARMAR-III als erster humanoider Roboter in die Lage versetzt, gleichzeitig und on-board Personen zu lokalisieren, ihre Gesichter zu erkennen und zu lernen, und Zeigegesten in Verbindung mit sprachlichen Kommandos zu interpretieren. Dieses Komplettsystem wurde u. a. in [Stiefelhagen u. a., 2007] veröffentlicht und in zahlreichen Demonstrationen (z. B. auf der CeBIT 2006) vorgeführt.

1.3 Aufbau der Arbeit

Die vorliegende Arbeit ist wie folgt aufgebaut:

Kapitel 2 beschreibt den Stand der Technik anhand verwandter Arbeiten in den Gebieten Personenlokalisierung und Gestenerkennung für mobile Roboter.

Kapitel 3 befasst sich mit dem Thema Personentracking. Hier werden zunächst Grundlagen und Standardverfahren zum Tracking beschrieben, aus denen im Anschluss ein Algorithmus zum Tracking mit dynamischer Merkmalsfusion entwickelt wird. Dieses neue Verfahren wird in zwei verschiedenen Szenarien evaluiert.

Kapitel 4 ist der Erkennung von Zeigegesten gewidmet. Dazu gehören das Tracking der Hände, die Modellierung und Klassifikation der Gesten, die Bestimmung der Zeigerichtung sowie die Verknüpfung von Zeigegesten mit Sprache und Kopfdrehung.

Kapitel 5 beschreibt die entwickelten Softwarekomponenten als Teil des Gesamtsystems des humanoiden Roboters ARMAR-III. Enge Berührungspunkte ergeben sich dabei zum natürlichsprachlichen Dialog sowie zur Gesichtsidentifikation.

Kapitel 6 fasst die Ergebnisse dieser Arbeit zusammen.

2 Verwandte Arbeiten

Für mobile Roboter, die mit Menschen interagieren – und sei es nur, um ihnen auszuweichen –, ist die visuelle Modellierung des Benutzers ein wichtiger Funktionsbestandteil. Beginnend bei grundlegenden Funktionen wie der Lokalisierung von Personen in der Umgebung wachsen die Ansprüche an die Benutzermodellierung mit den Fähigkeiten des Roboters. Insbesondere trifft dies auf teil- oder vollhumanoide Roboter zu, die durch ihr Äußeres beim Benutzer entsprechende Erwartungen wecken. Bekannte Vertreter dieser Klasse sind z. B. Asimo [Honda, 2008], HRP-2 [Kaneko u. a., 2004] oder der in Karlruhe entwickelte ARMAR-III [Asfour u. a., 2006], für dessen Funktionen die vorliegende Arbeit Beiträge liefert.

Existierende Arbeiten zur visuellen Benutzermodellierung lassen sich dabei grob in drei Gebiete einteilen, die sich aus den menschlichen Körperregionen ergeben, mit denen sie sich beschäftigen. Die erste Gruppe bilden die Arbeiten, die den Menschen als Ganzes betrachten, um daraus seinen räumlichen Standort abzuleiten (Lokalisierung bzw. Tracking). Eine zweite Gruppe von Arbeiten konzentriert sich auf den menschlichen Oberkörper. Hierzu gehören z. B. das Lokalisieren der Hände, aber auch das Tracking des Oberkörpers in Form eines artikularen Modells mit unterschiedlichem Detaillierungsgrad. Anwendung finden diese Informationen z. B. bei der Aktivitäten- oder Gestenerkennung. Eine dritte Gruppe analysiert schließlich das menschliche Gesicht. Dazu gehören u. a. Gesichtsidentifikation, Bestimmung des Aufmerksamkeitsfokus oder Erkennung von Mimik und Stimmungen.

Da sich die vorliegende Arbeit im Wesentlichen mit Personentracking und Zeigegestenerkennung befasst, werden im Folgenden verwandte Arbeiten aus diesen beiden Gebieten vorgestellt. Zum Bereich der Gestenerkennung soll hier auch die Beschaffung der dazu notwendigen Merkmale wie Handbewegung und Kopfdrehungsschätzung gehören.

2.1 Personenlokalisierung

Personentracking für Roboter

Zur Personenlokalisierung an Bord von Robotern können unterschiedliche Arten von Sensoren verwendet werden. Dazu zählen z. B. Laserscanner, Infrarotbewegungsmelder oder Radar, die nur begrenzte Informationen über die Umwelt liefern, dafür aber sehr schnell und robust funktionieren. Ein Beispiel dafür ist die Arbeit von [Schulz u. a., 2003], in der der mobile Roboter *Rhino* mit Laserscannern Personen in seiner Umgebung lokalisiert. Besondere Aufmerksamkeit wird dabei der Zuordnung der Messungen zu den einzelnen Personen zuteil, was dort mit einem *joint probability data association filter* (JPDAF) gelöst wird.

Auch durch Akustik ist eine Lokalisierung von Menschen möglich – zumindest dann, wenn sie sprechen oder andere Geräusche produzieren. In diesem Fall erfolgt die Schätzung in der Regel anhand von Laufzeitunterschieden des Schalls, der mit mehreren Mikrofonen am Körper des Roboters aufgenommen wird. So nutzen beispielsweise [Murase u. a., 2005] acht auf dem Torso des Roboters *SIG2* verteilte Mikrofone, um damit sich bewegende Sprecher zu lokalisieren. In ihrem Ansatz werden mit einem Beamforming-Algorithmus permanent alle Richtungen nach maximaler Schallintensität abgesucht. Die zeitliche Zuordnung zwischen den einzelnen Messungen und den Tracks erfolgt dann unter Verwendung mehrerer Kalmanfilter.

Die größte Menge an Informationen über Menschen in der Umgebung liefern Kameras. Dabei kann es sich um eine monokulare Kamera handeln oder aber – und dies ist bei humanoiden Roboter schon allein aufgrund der anthropomorphen Anmutung häufig der Fall – um zwei Kameras, die in einem in der Regel festen Abstand von einigen Zentimetern montiert sind. Mit einer solchen Stereokamera ist eine dreidimensionale Wahrnehmung der Szene möglich, was eine gute Schätzung für die Entfernung der Personen ermöglicht. Eine weitere Option stellen Panoramakameras dar, die dem Roboter eine 360°-Ansicht der Szene ermöglichen, dabei aber in ihrer Auflösung beschränkt sind und keine räumliche Information liefern.

In der Praxis sind durchaus auch Kombinationen der verschiedenen Sensorarten üblich. So stellen [Lang u. a., 2003] den mobilen Roboter *BIRON* vor, der mit einer aktiven Kamera (*pan-tilt*), zwei Mikrophonen und einem Laserscanner ausgestattet ist. Jeder dieser Sensoren lokalisiert den Benutzer auf seine Weise: Bei den Kamerabildern kommt Gesichtsdetektion nach [Viola und Jones, 2001] zum Einsatz, auf akustischer Seite wird der Zeitversatz des Schalls – und damit seine Richtung – per Kreuzkorrelation im Frequenzbereich gemessen, und in den Daten des Laserscanners werden Beinpaare detektiert. All diese Informationsquellen werden fusioniert, so dass *BIRON* als Ergebnis im Dialog mit Menschen immer den Sprecher fixieren und Personen folgen kann.

Ein früheres Beispiel für Sensorfusion zur Personenlokalisierung liefern [Nakadai u. a., 2001] mit dem humanoiden Kopf *SIG*. Hier werden Sprecher akustisch lokalisiert, während gleichzeitig per Farbmodell und *template matching* ihr Gesicht im Kamerabild lokalisiert wird. Die Steuerung des Roboterkopfes erfolgt dabei unter starker Priorisierung des akustischen Datenstroms.

[Miyashita u. a., 2004] lokalisieren Personen mit dem Teilhumanoiden *Robovie*, indem sie Messungen von einer gewöhnlichen Kamera, von einer Panoramakamera, von Ultraschallentfernungsmessern und von Infrarot-Bewegungsmeldern fusionieren. Das Tracking geschieht mit einem Partikelfilter nach [Isard und Blake, 1998a], wobei a-priori für die einzelnen Sensoren feste Verlässlichkeitsverteilungen angenommen werden.

In [Wilhelm u. a., 2004] wird ein zweistufiger Ansatz vorgestellt, mit dem der Serviceroboter *PERSES* seine Benutzer lokalisiert. Dabei wird zunächst mittels einer Panoramakamera und einer Reihe von Sonarsensoren grob nach Hinweisen in der Umgebung gesucht. Anschließend wird ein aktiver Kamerakopf in die entsprechende Richtung geschwenkt, um die Hypothese per Gesichtsdetektion zu überprüfen.

Personentracking mit Partikelfiltern

Die vorliegende Arbeit konzentriert sich auf das rein visuelle Tracking von Personen und bringt dabei als Trackingverfahren Partikelfilter zum Einsatz. Partikelfilter gehören zur Klasse der sequenziellen Monte-Carlo-Methoden (SMC-Methoden). Das sind stochastische Verfahren zur Zustandsschätzung eines dynamischen Prozesses, dessen Dynamik nur im statistischen Mittel bekannt ist, und der nur unvollständig beobachtet werden kann. Angewendet auf das Trackingproblem können sie z. B. eine kontinuierliche Bestimmung von Ort und Geschwindigkeit, basierend auf einer ungenauen Messung des Ortes, liefern. Spätestens seit der Formulierung des *Condensation*-Algorithmus durch [Isard und Blake, 1998a] erfreuen sich Partikelfilter in der Bildverarbeitung großer Beliebtheit.

So werden in [Isard und MacCormick, 2001] mehrere Personen, basierend auf Vordergrund-/Hintergrundmodellierung, im Bild einer monokularen Kamera verfolgt. Die Besonderheit der Arbeit liegt darin, dass sie einen Mechanismus enthält, bei dem der Partikelfilter sowohl die Position als auch die Anzahl der Personen in der Szene schätzt. Damit dies funktionieren kann, wird ein Multi-Blob-Beobachtungsmodell entworfen, das den Hypothesen trotz unterschiedlicher Objektanzahl vergleichbare Wahrscheinlichkeiten zuordnet.

Auch [Tao u. a., 1999] beschäftigen sich mit dem Tracking mehrerer Personen. Hier wird ein hierarchisches Samplingverfahren vorgeschlagen, bei der eine Ebe-

ne die Positionen der einzelnen Personen trackt, während die andere Ebene das Auftauchen und Verschwinden von Personen modelliert.

[Lanz, 2006] widmet sich dem Tracking von Personen in einer Mehrkameraumgebung, bei der gegenseitige Verdeckungen in jeder Perspektive die Regel sind. Er entwirft dazu das so genannte *Hybrid Joint-Seperable* (HJS) Modell, bei dem die Partikelmengen der einzelnen Personen unabhängig voneinander und daher effizient propagiert werden können, während gleichzeitig der Vergleich mit der Beobachtung unter Berücksichtigung des gemeinsamen Zustandsraumes stattfindet. Dadurch lassen sich Verdeckungen explizit behandeln, ohne an dem sonst für Partikelfilter typischen Problem der hohen Dimension des Zustandsraumes zu scheitern.

Personentracking mit mehreren Merkmalen

Ein wichtiges Thema beim Personentracking ist die geeignete Wahl der Merkmale. Häufig zum Einsatz kommen dabei folgende Merkmalstypen:

- Vordergrund-/Hintergrundmodelle, wie z. B. in [Stauffer und Grimson, 2000] oder [Isard und MacCormick, 2001]
- Disparitätenbilder aus der Stereobildverarbeitung, wie etwa von [Scharstein und Szeliski, 2002] oder [Veksler, 2003] erläutert, und z. B. von [Darrell u. a., 2001] angewandt
- Farbmodelle für Hautfarbe, wie z. B. [Yang u. a., 1997] zum Gesichtstracking, oder – in den letzten Jahren immer häufiger – für verschiedene Partitionierungen des Körpers, wie z. B. in [Satoh u. a., 2004], [Adam u. a., 2006], [Porikli, 2005], [Nummiaro u. a., 2003] oder [Lanz, 2006]
- Detektoren für Gesicht und Körper, wie z. B. Haar-feature-Klassifikatoren in [Viola und Jones, 2001; Lienhart und Maydt, 2002; Kruppa u. a., 2003], *Histogram-of-oriented-Gradients* (HoG) nach [Dalal und Triggs, 2005], oder kombinierte Detektionsverfahren wie in [Leibe u. a., 2005]
- Kanten bzw. Umrisse, modelliert z. B. durch Splines wie in [Branson und Belongie, 2005] oder [Isard und Blake, 1998b]

Häufig werden verschiedene Merkmale kombiniert, um dadurch eine deutlichere Trennung zwischen Person und Hintergrund zu erzielen. Ein Beispiel hierfür geben [Darrell u. a., 2000], die Gesichtsdetektion, Disparitätenbild und Hautfarbsegmentierung in einem bottom-up-Verfahren anwenden, um Personen zu lokalisieren. Weitere Beispiele sind [Wu und Huang, 2001], bei denen Farbe und eine Ellipsenform zum Kopftracking kombiniert werden, oder [Patil u. a., 2004] mit der Fusion von Differenzbild, Gesichtsdetektor und Farb-Blobs.

[Yang u. a., 2005] verwenden Farbmodelle, Gradientenhistogramme und Detektoren zum Tracking von Personen. Dabei werden die einzelnen Merkmale nacheinander angewendet, um zunächst mit schnellen und einfachen Merkmalen die Masse der Hypothesen ausschließen zu können. Die aufwändigeren Merkmale müssen dann nur zur Überprüfung der wirklich Erfolg versprechenden Kandidaten bemüht werden.

Dieser Gedanke einer Sortierung der Merkmale „von grob nach fein“ wurde auf sehr effiziente Art und Weise von [Pérez u. a., 2004] für die Merkmalsfusion speziell im Partikelfilter formuliert: Im so genannten *Layered Sampling* wird nach der Auswertung eines jeden Merkmals ein Resamplingschritt durchgeführt. Das bedeutet, dass für das jeweils nachfolgende Merkmal eine neue Partikelmenge gebildet wird, die sich genau um jene Bereiche im Zustandsraum ballt, die vom vorherigen Merkmal als vielversprechend erachtet wurden. Layered Sampling wird daher – wie später beschrieben – in einer erweiterten Form auch in der vorliegenden Arbeit eingesetzt.

Dynamische Merkmalsfusion beim Tracking

Die bisher geschilderten Beispiele für das Tracking mit mehreren Merkmalen folgen einem statischen Fusionsschema, d. h. der Einfluss der einzelnen Merkmale wird a priori festgelegt und ändert sich zur Laufzeit nicht; ein Beispiel hierfür ist [Miyashita u. a., 2004], bei dem die Fehlercharakteristik der einzelnen Sensoren/Merkmale fest modelliert wird. Tatsächlich sind Merkmale aber je nach Situation einmal besser und einmal schlechter in der Lage, die Person zu lokalisieren. Es liegt daher nahe, den Einfluss von Merkmalen situationsabhängig zu gestalten.

[Triesch und Malsburg, 2001] stellen hierzu das Verfahren der Demokratischen Integration (*democratic integration*, DI) vor, bei dem die Merkmale in Form einer gewichteten Summe integriert werden.[1] Die Gewichte werden dabei ständig der aktuellen Situation angepasst. Dies geschieht, indem jedes einzelne Merkmal mit dem Ergebnis verglichen wird, das durch die Fusion aller Merkmale entsteht. So können Merkmale, die von der Mehrheit abweichen, in ihrem Einfluss gedämpft werden. In der ursprünglichen Formulierung der DI werden *saliency maps* verschiedener Merkmale gebildet, und es wird per erschöpfender Maximumsuche die Hypothese gefunden. Diese Verfahrensweise findet auch direkte Anwendung in [Kim u. a., 2004], bei der ein anthropomorpher Roboterkopf einen Menschen fixiert.

Von [Spengler und Schiele, 2003] wird vorgeschlagen, DI in einen Partikelfilter zu integrieren, um dessen dem einfachen Trackingverfahren aus [Triesch und

[1] Eine Begründung für die Zulässigkeit und für die praktischen Vorteile der Summenregel zur Merkmalsfusion liefert [Kittler u. a., 1998].

Malsburg, 2001] überlegene Fähigkeiten, wie z. B. die inhärente Multihypothesensuche, in Verbindung mit dynamischer Merkmalsfusion nutzen zu können. In obiger Veröffentlichung bleibt es allerdings bei einem Vorschlag, da die tatsächliche Integration von Partikelfilter und DI nicht gezeigt wird: So werden zum einen in der Beispielimplementierung nur zwei Merkmale (Bewegung und Hautfarbe) verwendet, was es schwierig macht, von einem Mehrheitsentscheid zu sprechen, und zum anderen werden die Fusionsgewichte konstant gehalten, weil kein passendes Qualitätsmaß zur Verfügung steht.

Erst mit [Shen u. a., 2003] wird ein solches Qualitätsmaß zur Einbindung von DI in partikelfilterbasiertes Tracking eingeführt, mit dem die Fusionsgewichte dynamisch angepasst werden können. Vorgeführt wird das Verfahren am Beispiel des Gesichtstrackings mit den Merkmalen Farbe und Umriss.

In der vorliegenden Arbeit wird in Kapitel 3 der DI^2-Algorithmus vorgestellt, der vorwiegend auf die zuletzt genannten Publikationen Bezug nimmt.

2.2 Gestenerkennung

Zur automatischen Erkennung von Gesten für die Mensch-Maschine-Interaktion liefern [Pavlovic u. a., 1997] einen Überblick. Sie kommen dabei unter anderem zu einer Taxonomie von Handgesten für die Mensch-Maschine-Interaktion, die in Abbildung 2.1 wiedergegeben ist. Demnach lassen sich die „absichtlichen“ Handbewegungen in zwei Klassen unterteilen: Dies sind zum einen die manipulativen Gesten, die in der Robotik ein wichtiges Thema z. B. für die Erkennung von Griffen sind und in [Cutkosky, 1989] noch weiter differenziert werden, und zum anderen die kommunikativen Gesten. Diese wiederum zerfallen in die zwei Gruppen *Symbols*, die eine linguistische Rolle spielen sowie *Acts*, bei denen die Interpretation allein von der Beobachtung der Bewegung abhängt. Die in der vorliegenden Arbeit betrachteten Zeigegesten fallen in die zweite Klasse und bilden dort die Untergruppe der deiktischen Gesten.

Gesten werden in der Kommunikation von Mensch zu Mensch häufig in Verbindung mit Sprache eingesetzt. Dieses wurde z. B. von [McNeill, 1992] untersucht, der dabei zu einer Einteilung in vier Klassen kommt, die später in Abbildung 4.1 dargestellt ist. Auch hier stellen die deiktischen Gesten eine eigene von vier Klassen dar, die abgegrenzt wird von ikonischen, metaphorischen und den so genannten „Beat“-Gesten, die eher unwillkürlich ausgeführt werden. Aufbauend darauf formuliert [Cassell, 1998] Gedanken und erste Ansätze zur automatischen Integration von Gesten in einen Dialog zwischen Mensch und Maschine, die in diesem Fall durch einen grafischen Avatar repräsentiert wird.

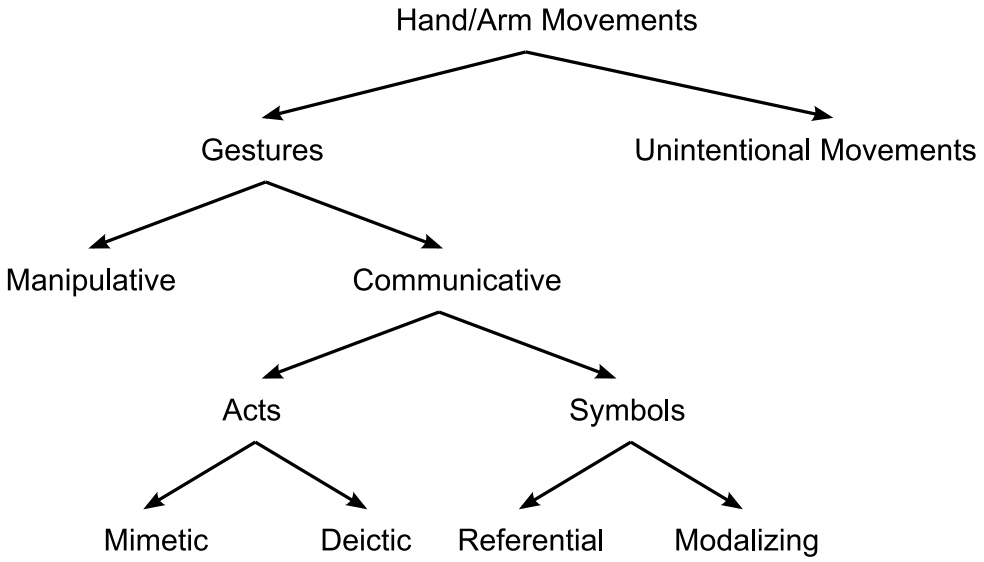


Abbildung 2.1: Taxonomie von Handgesten zur Mensch-Maschine-Interaktion nach [Pavlovic u. a., 1997]

Einen Überblick über automatische Erkennung von Gesten mit Mitteln der Bildverarbeitung liefern [Wu und Huang, 1999]. Sie unterscheiden dabei u. a. zwischen zwei Arten von Gesten bzw. ihrer Erkennung:

- Statische Gesten, die durch eine bestimmte Körperhaltung (Pose) eindeutig repräsentiert sind, und die somit durch Betrachtung eines Einzelbildes erkannt werden können
- Dynamische Gesten, bei denen zumindest ein Teil der Information auch im zeitlichen Ablauf liegt

Nicht immer ist die Zuordnung eindeutig: So betrachten etliche Arbeiten, wie im Folgenden beschrieben, Zeigegesten als statisch, während andere Arbeiten wiederum gerade ihre dynamischen Eigenschaften hervorheben.

Dynamische Gesten lassen sich nach [Kendon, 1986] anhand von Beobachtungen in die drei Bewegungsphasen *preparation*, *stroke* und *retraction* unterteilen. In Kapitel 4 der vorliegenden Arbeit wird bei der Erkennung von Zeigegesten eine ganz ähnliche Segmentierung vorgenommen und technisch nutzbar gemacht.

Automatische Zeigegestenerkennung

Die Erkennung von Zeigegesten selbst wurde in der Vergangenheit in der Regel nicht mit dynamischen Verfahren durchgeführt, die den Verlauf der Bewegung analysieren. So stellen [Kahn u. a., 1996] einen frühen Zeigegestenerkenner für die Mensch-Roboter-Interaktion vor, der eine Zeigegeste dann detektiert, wenn

der Arm für eine bestimmte Zeit seitlich von der Silhouette der Person abgesetzt erscheint. Es handelt sich dabei um ein rein zweidimensional arbeitendes Verfahren, bei dem auch die Auswahl des Zielobjektes durch einen Strahl im Bildbereich vorgenommen wird. Als Merkmale kommen dabei Grauwertbild, Kantenbild, Bewegung, Disparität und Hautfarbsegmentierung zum Einsatz.

Vergleichbar hiermit ist auch [Kortenkamp u. a., 1996], deren Ansatz sechs verschiedene Gesten, darunter eine Zeigegeste und eine „Halt!"-Geste, anhand der charakteristischen Winkel von Arm- und Schultergelenken statisch klassifiziert. Mit dem dort verwendeten 3D-Trackingsystem können erstmals auch 3D-Richtungsschätzungen vorgenommen werden. In [Littmann u. a., 1996] wird mithilfe Künstlicher Neuronaler Netze (KNN) die zeigende Hand über einem Tisch lokalisiert, und die dreidimensionale Handposition wird auf ein Gitter möglicher Ziele auf der Arbeitsfläche abgebildet.

[Waldherr u. a., 2000] beschreiben ein komplexes Bildverarbeitungssystem zur zweidimensionalen Gestenerkennung für einen mobilen Roboter, bei dem eine Stopp-Geste, eine Folgegeste sowie eine hohe und eine niedrige Zeigegeste untersucht werden. Hiervon wird allerdings nur die Folgegeste dynamisch modelliert, die anderen Gesten werden als statische Pose erkannt.

Ein Zeigegestenerkenner, der vollständig in 3D arbeitet, wird von [Jojic u. a., 2000] vorgestellt. Hier wird der Körper des Menschen als 3D-Punktwolke betrachtet, die allein aus dem Disparitätenbild gewonnen wird. Mit dem EM-Algorithmus wird eine Gaußmischverteilung mit zwei Komponenten auf die Punktwolke angepasst. Unterscheiden sich die durch die Kovarianzmatrizen gegebenen Richtungen beider Komponenten stark voneinander, dann wird davon ausgegangen, dass es sich bei einer der beiden um einen ausgestreckten Arm handelt. Die Zeigerichtungsschätzung erfolgt dabei kontinuierlich anhand der „Spitze" des Arms. Anwendungsbeispiel für dieses System ist das kontinuierliche Bewegen eines Cursors auf einer Projektionsfläche.

In eine ganz ähnliche Richtung geht die Arbeit von [Demirdjian und Darrell, 2002], bei der der Mensch ebenfalls als 3D-Punktwolke betrachtet wird, in die mit dem bekannten *iterative-closest-point* Algorithmus (ICP) ein Zylindermodell des Oberkörpers eingepasst wird. Auch hier geht es u. a. um Cursorsteuerung, eine explizite Detektion von Zeigegesten findet daher nicht statt.

[Starner u. a., 1998] liefern mit der bekannten Arbeit zur automatischen Erkennung von Gebärdensprache (*American Sign Language*) eines der ersten Systeme, in denen Hidden-Markov-Modelle (HMMs) zur Klassifikation von Handgesten verwendet werden. Ein zweiter früher Vertreter dieser Richtung ist [Becker, 1997], bei dem verschiedene Bewegungen aus dem Repertoire des T'ai Chi klassifiziert werden.

Eine Erweiterung von HMMs zur Gestenerkennung schlagen [Wilson und Bobick, 1998] vor. Sie weisen auf die Tatsache hin, dass manche Gesten von ei-

nem Parameter bestimmt werden, der ihre Ausführung stark beeinflusst. Bei der Zeigegeste ist dies die Zeigerichtung: Je nach Ort des Ziels wird die Geste unterschiedlich ausgeführt. Um dies explizit zu modellieren, wandeln sie die bekannten Algorithmen zur Klassifikation und zum Training von HMMs so ab, dass der bestimmende Parameter Teil des Modells selbst wird. So kann bei der Dekodierung der Geste durch das HMM automatisch der Richtungsparameter mit geschätzt werden.

Merkmale zur Zeigegestenerkennung

Für die Erkennung von Zeigegesten werden in der Literatur unterschiedliche Merkmale verwendet. So arbeiten [Kahn u. a., 1996] oder [Waldherr u. a., 2000] bildbasiert auf Merkmalskarten, in denen die Region um den zeigenden Arm betrachtet wird. Dem gegenüber stehen die reinen 3D-Verfahren wie [Jojic u. a., 2000] oder [Demirdjian und Darrell, 2002], die ausschließlich mit Punktwolken aus der Stereobildverarbeitung arbeiten, und sich dort auf die Armrichtung konzentrieren.

Bei der Erkennung anderer Arten von Handgesten hingegen ist z. B. aus [Starner u. a., 1998] oder [Becker, 1997] bekannt, dass Trajektorien der Hände geeignete Merkmale für HMMs darstellen. Da in der vorliegenden Arbeit Zeigegesten dynamisch modelliert und dadurch auch präzise zeitlich lokalisiert werden sollen, wird auch hier ein HMM-basiertes Verfahren eingesetzt, das sich auf Handtracking stützt. Siehe hierzu auch [Campbell u. a., 1996], die unterschiedliche Darstellungen von Handpositionen für die Gestenerkennung systematisch vergleichen.

In der Literatur wohlbekannt ist ein Zusammenhang zwischen dem Aufmerksamkeitsfokus eines Menschen und seiner Blickrichtung, siehe hierzu bspw. [Yarbus, 1967; Barber und Legge, 1976] oder [Glenstrup und Engell-Nielsen, 1995]. Daher ist es auch bei Zeigegesten naheliegend, dass Menschen tendenziell in die Richtung blicken, in die sie zeigen. In der vorliegenden Arbeit wird daher versucht, Zeigegesten durch Betrachtung sowohl der Handbewegung als auch der Kopfdrehung zu erkennen. Zur visuellen Schätzung von Kopfdrehung existieren robuste Verfahren, wie z. B. [Stiefelhagen u. a., 2000] oder [Ba und Odobez, 2004], auf denen hier aufgebaut werden kann.

Für die Verknüpfung von Kopfdrehungsschätzung und Handtracking zur automatischen Zeigegestenerkennung konnte in der Literatur bislang kein Beispiel gefunden werden. Erst mit [Gross u. a., 2006] – und damit nach dem Erscheinen eigener Vorarbeiten – wird ein kombiniertes System zum Einsatz auf einem mobilen Roboter vorgestellt, das die statische Körperhaltung beim Zeigen betrachtet. Dabei werden mit einer Kaskade von KNNs bildbasiert sowohl der

Oberkörper als auch der Kopf klassifiziert und dadurch auf die Zeigerichtung geschlossen.

3 Personentracking mit dynamischer Merkmalsfusion

Damit ein humanoider Roboter im Dialog mit dem Menschen die von seinem Erscheinungsbild geweckten Erwartungen erfüllen kann, muss er eine Reihe von Verhaltensweisen und Fähigkeiten nachbilden, die in der Mensch-Mensch-Kommunikation selbstverständlich sind. Eine dieser Fähigkeiten ist, den Kommunikationspartner durch Bewegung des Kamerakopfes ständig im Blickfeld zu behalten. Zum einen signalisiert dies dem Kommunikationspartner die Aufmerksamkeit des Roboters, zum anderen stellt es eine notwendige Voraussetzung für weitergehende Bildanalysen dar, wie z. B. Aktivitätenanalyse, Identifikation oder Gestenerkennung. Um den Kamerakopf nachführen zu können, muss der Mensch in Echtzeit im Kamerabild lokalisiert werden. Ergebnis dieses Vorgangs ist die räumliche Position des Menschen, die außer zur Nachführung auch noch für eine Reihe von anderen Zwecken erforderlich ist, wie z. B. zur sicheren Navigation/Bahnplanung, Objektübergabe, Situationserkennung, usw.

In diesem Kapitel wird ein Verfahren zum räumlichen Lokalisieren einer oder mehrerer Personen mithilfe eines Stereokamerasystems vorgestellt. Beim Entwurf des Lokalisierungsverfahrens wird auf die spezifischen Bedingungen beim Einsatz auf einem humanoiden Roboter eingegangen. Eine besondere Herausforderung liegt hierbei in Auswahl und Verknüpfung der Merkmale zur Lokalisierung des Menschen, wie folgende Aufstellung deutlich macht:

- Ein humanoider Roboter bewegt sich frei in verschiedenen Umgebungen. Das bedeutet, dass die Lichtverhältnisse unkontrollierbar sind und z. B. mit Schattenwurf oder Gegenlicht gerechnet werden muss. Ebenso kann sich der Bildhintergrund jederzeit ändern, und zwar sowohl durch Dynamik innerhalb der Szene, als auch durch Eigenbewegung des Roboters. Merkmale, die in einer Situation gut funktionieren, z. B. bei statischem Hintergrund, versagen zwangsläufig in anderen.
- Der Kamerakopf eines humanoiden Roboters befindet sich aus Sicht der Bildverarbeitung „mitten in der Szene“, d. h. die Bilder werden in der Regel nicht aus der günstigen Perspektive aufgenommen, die beispielsweise ein außerhalb der Szene leicht erhöht stehender Beobachter hätte. Infolgedessen variiert das Abbild des Menschen im Arbeitsbereich (ca. $0,5m - 10m$ Entfernung vom Roboter) sehr stark, wie in Abbildung 3.1

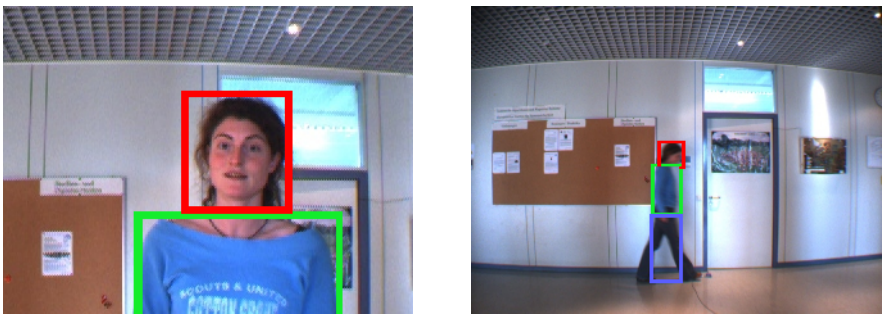


Abbildung 3.1: Das Abbild eines Menschen, aufgenommen aus der Egoperspektive des Roboters, ändert sich innerhalb des Arbeitsbereiches $(0,5m - 10m)$ stark. Die drei Körperregionen Kopf, Torso und Beine sind farblich hervorgehoben.

dargestellt ist. Bestimmte Merkmale zum Auffinden des Menschen in Nahaufnahmen sind in entfernten Aufnahmen nicht mehr anwendbar; dennoch soll nach Möglichkeit nicht komplett auf sie verzichtet werden.

- Die Rechenleistung an Bord des Roboters ist aus Gründen von Raum- und Energieeinsparung eng begrenzt und muss zudem mit anderen, parallel laufenden Funktionen des Roboters geteilt werden. Die Personenlokalisierung muss sich deshalb dynamisch auf bestimmte Bildbereiche beschränken. Auch können nur Merkmale verwendet werden, die mit den gegebenen Mitteln effizient in Echtzeit berechnet werden können.

Der Schwerpunkt des vorgestellten Verfahrens liegt daher auf der Frage, wie verschiedene, einfache Merkmale dynamisch kombiniert werden können, so dass der Ausfall einzelner Merkmale spontan durch andere Merkmale überbrückt werden kann. Dabei wird im Folgenden der Merkmalsbegriff dahingehend erweitert, dass damit nicht nur Merkmalstypen wie Farbe oder Bewegung gemeint sind, sondern auch unterschiedliche Regionen des Zielobjekts bzw. unterschiedliche Kameraansichten. Es zeigt sich, dass die Fähigkeiten des hier entworfenen Algorithmus nicht nur auf das eigentliche Szenario – das Tracking von Personen mit einem beweglichen Stereokamerakopf – anwendbar sind, sondern darüber hinaus auch auf andere Situationen übertragen werden können.

Im folgenden Abschnitt werden zunächst die Grundlagen des visuellen Trackings mithilfe von Partikelfiltern erläutert. Anschließend wird das Verfahren zur dynamischen Merkmalsfusion vorgestellt, das den Kernpunkt des neuen Algorithmus darstellt. Im dritten Abschnitt werden verschiedene einfache Merkmale zur Modellierung von Menschen gezeigt und mit dem dynamischen Fusionsalgorithmus zu einem kompletten Personentracker verknüpft. Zuletzt wird das Verfahren im vierten Abschnitt sowohl im Roboterszenario als auch in Mehrkameraumgebungen evaluiert.

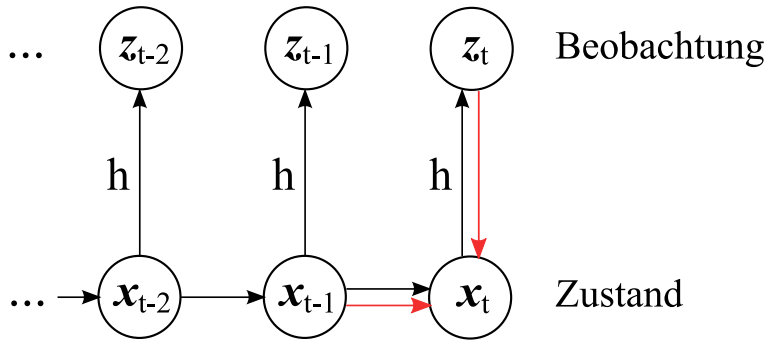


Abbildung 3.2: Tracking als kontinuierliche Schätzung des Zustands x_t anhand der Beobachtung z_t. Unter Verwendung der Markov-Annahme müssen jeweils nur die aktuelle Beobachtung und der vergangene Zustand betrachtet werden (rot dargestellt).

3.1 Grundlagen

Der Begriff *Tracking* steht in der Bildverarbeitung für das fortlaufende Lokalisieren eines Zielobjekts in Videosequenzen. Die Lokalisierung liefert dabei je nach Art des Trackings eine Reihe von Informationen über das Zielobjekt bzw. über seine aktuellen Eigenschaften. Dazu zählen z. B. Bildkoordinaten, Weltkoordinaten, Geschwindigkeit, Umriss, Lage, usw. Visuelles Tracking stellt somit eine Form der Zustandsschätzung dar, bei der der Zustand eines in der Regel bekannten Objekts anhand gestörter Messwerte (hier: Kamerabilder) zu ermitteln ist. Im Gegensatz zu einer einmaligen Zustandsschätzung auf einem Einzelbild – wie z. B. bei einer Detektion – ist beim Tracking eine Videosequenz gegeben, deren Einzelbilder in kurzem zeitlichen Abstand aufgenommen werden. Die Zustandsänderung des Zielobjekts zwischen zwei Bildern fällt in der Regel gering aus, so dass das Ergebnis der Schätzung aus dem vergangenen Bild eine wichtige Information zur Bearbeitung des aktuellen Bildes ist.

In Abbildung 3.2 wird dieser Sachverhalt dargestellt: Geschätzt werden soll der Objektzustand $\mathbf{x}_t$ zum Zeitpunkt t. Direkt sichtbar ist allerdings nicht der Zustand, sondern die Beobachtung $\mathbf{z}_t$, d. h. das Kamerabild. Diese wird über eine Abbildungsfunktion h vom Objektzustand beeinflusst. Aufgabe des Trackings ist es nun, $\mathbf{x}_t$ anhand der aktuellen Beobachtung $\mathbf{z}_t$ und der Historie $\mathbf{x}_{1..t-1}$ bzw. $\mathbf{z}_{1..t-1}$ zu schätzen. Dabei wird in der Regel die vereinfachende Markov-Annahme getroffen, nach der die Information über die Historie im vergangenen Zustand $\mathbf{x}_{t-1}$ subsumiert ist. So kann die Schätzung von $\mathbf{x}_t$ allein anhand von $\mathbf{z}_t$ und $\mathbf{x}_{t-1}$ durchgeführt werden, was den Aufwand drastisch verringert.

Betrachtet man die Zustandsschätzung als einen probabilistischen Prozess, dann gelangt man zu der zweistufigen Darstellung aus Abbildung 3.3. Im Vorhersa-

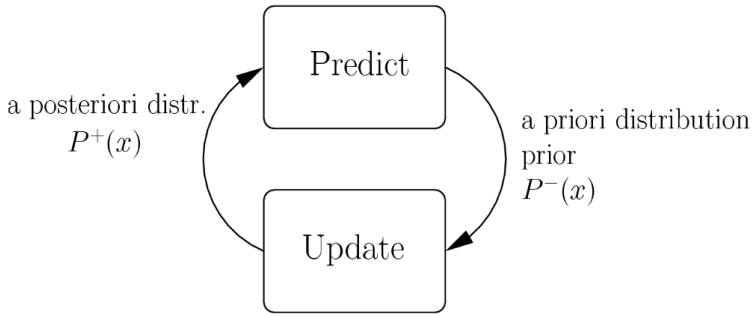


Abbildung 3.3: Zweistufige Darstellung des probabilistischen Trackings.

geschritt (*predict*) wird dabei die a-priori Wahrscheinlichkeitsverteilung $P^-(\mathbf{x})$ für den aktuellen Zustand $\mathbf{x}$ gebildet. $P^-(\mathbf{x})$ wird allein aus dem vergangenen Zustand und aus Vorwissen um die Systemdynamik, also z. B. um die mögliche Änderungsgeschwindigkeit des Zustands, gebildet. Erst im Aktualisierungsschritt (*update*) kommt als zweite Informationsquelle die aktuelle Beobachtung ins Spiel. Mit ihrer Hilfe wird aus der a-priori die a-posteriori Wahrscheinlichkeitsverteilung $P^+(\mathbf{x})$. Sie beinhaltet sämtliches Wissen zum aktuellen Zeitpunkt; ihr Erwartungswert kann als momentane Schätzung ausgegeben werden.

Im folgenden Unterabschnitt wird mit dem *Condensation*-Algorithmus nach [Isard und Blake, 1998a] ein bekanntes Verfahren aus der Klasse der Partikelfilter vorgestellt. Der Algorithmus ist in der Lage, den zuvor beschriebenen probabilistischen Prozess numerisch zu lösen. Er tut dies, ohne für die oben genannten Wahrscheinlichkeitsverteilungen analytische Formulierungen vorauszusetzen, die in der Praxis kaum zur Verfügung stehen.

3.1.1 Der *Condensation*-Algorithmus

Partikelfilter gehören zur Klasse der sequenziellen Monte-Carlo-Methoden (SMC-Methoden). Dies sind stochastische Verfahren zur Zustandsschätzung eines dynamischen Prozesses, dessen Dynamik nur im statistischen Mittel bekannt ist, und der nur unvollständig beobachtet werden kann. Angewendet auf das Trackingproblem können sie z. B. eine kontinuierliche Bestimmung von Ort und Geschwindigkeit basierend auf einer ungenauen Messung des Ortes liefern.

Eine in der Bildverarbeitung häufig anzutreffende Variante ist der so genannte *Condensation*-Filter nach [Isard und Blake, 1998a], der auch in dieser Arbeit verwendet werden soll: Es bezeichne $\mathcal{Z}_t = \{\mathbf{z}_1, \ldots, \mathbf{z}_t\}$ die Sequenz der Beobachtungen bis zum Zeitpunkt t und $\mathcal{X}_t = \{\mathbf{x}_1, \ldots, \mathbf{x}_t\}$ die Sequenz der dazugehörigen Systemzustände. Für die einzelnen Beobachtungen gelte die Annahme,

dass sie stochastisch unabhängig sind. Weiterhin gelte die Markov-Annahme, nach der der aktuelle Zustand nur von seinem Vorgänger, nicht aber von der gesamten Historie abhängt:

$$p(\mathbf{x}_t|\mathcal{X}_t) = p(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{3.1}$$

Die Entwicklung der Wahrscheinlichkeitsdichte für den Zustand erfolgt dann laut [Isard und Blake, 1998a] nach folgender Regel:

$$p(\mathbf{x}_t|\mathcal{Z}_t) = k_t \; p(\mathbf{z}_t|\mathbf{x}_t) \; p(\mathbf{x}_t|\mathcal{Z}_{t-1}), \tag{3.2}$$

wobei

$$p(\mathbf{x}_t|\mathcal{Z}_{t-1}) = \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_t|\mathbf{x}_{t-1}) \; p(\mathbf{x}_{t-1}|\mathcal{Z}_{t-1}) \tag{3.3}$$

und k_t eine von $\mathbf{x}_t$ unabhängige Normalisierungskonstante ist.

Die Entwicklungsregel nach 3.2 lässt sich als zeitabhängiges Pendant zur Regel von Bayes auffassen, bei dem $p(\mathbf{x}_t|\mathcal{Z}_{t-1})$ – als Äquivalent zur Bayes'schen a-priori Verteilung $p(\mathbf{x})$ – eine Prädiktion, basierend auf der a-posteriori Verteilung $p(\mathbf{x}_{t-1}|\mathcal{Z}_{t-1})$ aus dem vorherigen Zeitschritt darstellt.

Da $p(\mathbf{z}_t|\mathbf{x}_t)$ im Allgemeinen so komplex ist, dass 3.2 nicht in geschlossener Form vorliegt, verwendet der *Condensation*-Algorithmus als Näherungsverfahren das so genannte *factored sampling*. Dabei wird aus dem Zustandsraum eine Menge von Partikeln (*samples*) und zugehörigen Gewichten gebildet, $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)}),\ n = 1, \ldots, N\}$, die die Wahrscheinlichkeitsdichte $p(\mathbf{x}_t|\mathcal{Z}_t)$ repräsentiert, wobei die Genauigkeit der Annäherung mit dem Wert von N steigt.

Um die Partikelmenge zu einem Zeitpunkt t zu bestimmen, wird zunächst die a-priori Verteilung $p(\mathbf{x}_t|\mathcal{Z}_{t-1})$ betrachtet. So werden aus den N alten Partikeln $\mathbf{s}_{t-1}^{(n)}$, also der alten a-posteriori Verteilung, gemäß ihrer Gewichte $\pi_{t-1}^{(n)}$ zufällig N neue Partikel $\mathbf{s}_t^{(n)}$ gezogen. Diese müssen gemäß Gleichung 3.3 propagiert werden und bilden dann die Näherung für $p(\mathbf{x}_t|\mathcal{Z}_{t-1})$. Kommt nun die aktuelle Beobachtung $\mathbf{z}_t$ hinzu, können die neuen Gewichte als $\pi_t^{(n)} \propto p(\mathbf{z}_t|\mathbf{x}_t = \mathbf{s}_t^{(n)})$ berechnet werden.

Zusammengefasst müssen im *Condensation*-Algorithmus also zu jedem Zeitpunkt t zwei Schritte ausgeführt werden:

1. Die Vorhersage: Aus der alten Partikelmenge werden unter Berücksichtigung ihrer Gewichte zufällig N neue Partikel gezogen. Diese neuen Partikel werden mithilfe des *dynamischen Modells* $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ propagiert.
2. Die Messung: Die Gewichte $\pi_t^{(n)}$ der neuen Partikel werden mithilfe des Beobachtungsmodells $p(\mathbf{z}_t|\mathbf{x}_t = \mathbf{s}_t^{(n)})$ festgelegt.

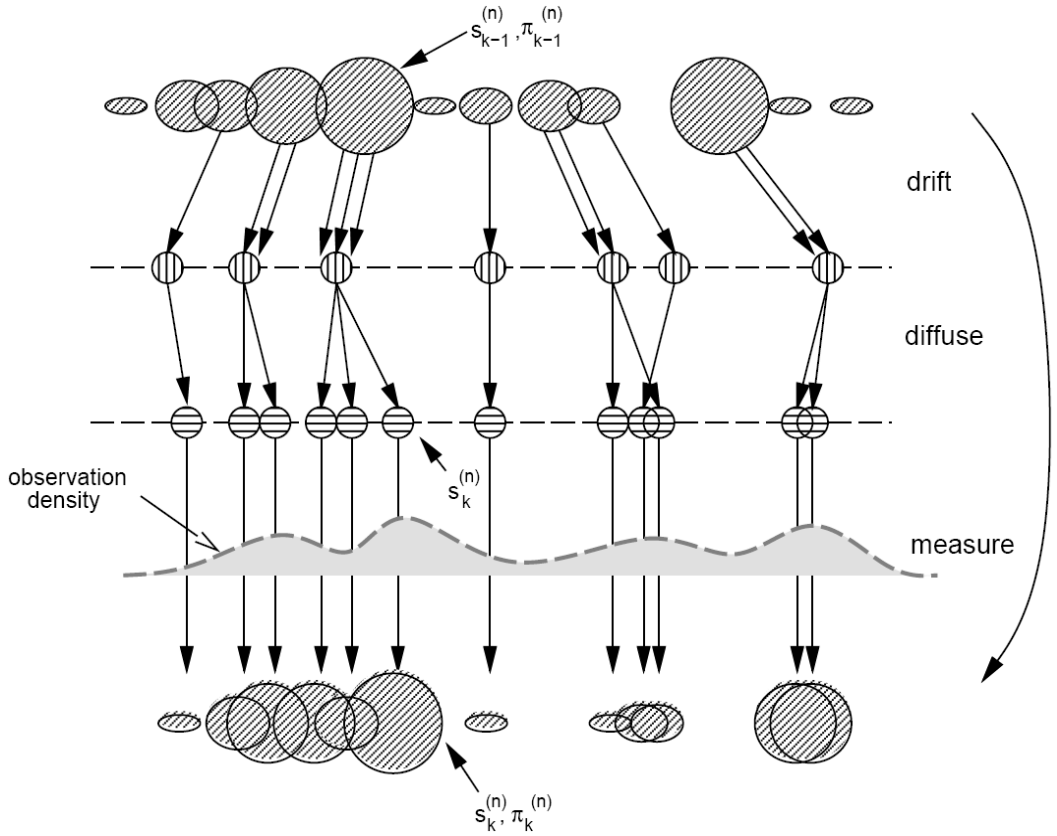


Abbildung 3.4: Graphische Darstellung des *Condensation*-Algorithmus, Quelle [Isard und Blake, 1998a]: Aus der Partikelmenge zum Zeitpunkt $t-1$ werden neue Partikel gezogen und mithilfe des dynamischen Modells propagiert. Das dynamische Modell besteht dabei in der Regel aus einem deterministischen Anteil (*drift*) und einem stochastischen Anteil (*diffusion*), der die Unsicherheit der Vorhersage repräsentiert. Mithilfe der Messung (*measure*) werden dann die Gewichte der neuen Partikel bestimmt.

Abbildung 3.4 stellt den Ablauf des Condensation-Algorithmus innerhalb eines Zeitschritts grafisch dar.

Als Hypothese $\hat{\mathbf{x}}_t$ über den aktuellen Zustand des Systems kann letztendlich der Erwartungswert der Partikelmenge gebildet werden. Wenn die Gewichte normiert sind als $\sum_n \pi_t^{(n)} = 1$, hat dieser die Form

$$\hat{\mathbf{x}}_t = E(\mathbf{x_t}|\mathbf{z_t}) = \sum_{n=1..N} \pi_t^{(n)} s_t^{(n)}. \tag{3.4}$$

3.1.2 Layered Sampling

Layered Sampling, eingeführt von [Pérez u. a., 2004], kann unter bestimmten Voraussetzungen die Effizienz eines Partikelfilters verbessern, wenn mehrere Merkmale zum Beobachtungsmodell beitragen. Hier soll zunächst kurz die allgemeine Funktionsweise des Layered Sampling beschrieben werden, konkret eingesetzt wird es später in Abschnitt 3.2.4 bei der dynamischen Merkmalskombination im Personentracking.

Unter der vereinfachenden Annahme, dass die Beobachtung $\mathbf{z}$ aus M unabhängigen Quellen stammt, lässt sich die Beobachtungswahrscheinlichkeit zerlegen zu:

$$p(\mathbf{z}|\mathbf{x}) = \prod_{m=1..M} p(\mathbf{z}^m|\mathbf{x}) \tag{3.5}$$

Nach [Pérez u. a., 2004] kann in diesem Fall die Zustandsübergangswahrscheinlichkeit ebenfalls in M aufeinander folgende Zwischenstufen zerlegt werden, so dass

$$p(\mathbf{x}|\mathbf{x}') = \int p_M(\mathbf{x}|\mathbf{x}^{M-1}) \cdots p_1(\mathbf{x}^1|\mathbf{x}') d\mathbf{x}^1 \cdots d\mathbf{x}^{M-1} \tag{3.6}$$

wobei $\mathbf{x}^1 \cdots \mathbf{x}^{M-1}$ Hilfswerte für die Zwischenzustände sind[1]. Wird ein Gauß'sches Zustandsübergangsmodell zugrunde gelegt, bedeutet dies eine Aufspaltung in M aufeinanderfolgende Stufen mit jeweils entsprechend geringerer Varianz. Weiterhin wird die vereinfachende Annahme gemacht, dass die m-te Messung $p(\mathbf{z}^m|\mathbf{x})$ nach Anwendung des m-ten Übergangsschrittes $p_m(\mathbf{x}^m|\mathbf{x}^{m-1})$ erfolgt. Dies führt zu einer stufenweisen Sampling-Strategie, bei der in der m-ten Stufe neue Samples aus einer Monte-Carlo-Schätzung der Verteilung $p_m(\mathbf{x}^m|\mathbf{x}^{m-1})\pi^{m-1}$ mit $\pi^m \propto p(\mathbf{z}^m|\mathbf{x}^m)$ gezogen werden.

Wie von [Pérez u. a., 2004] gezeigt, ist Layered Sampling dann von Vorteil, wenn für die Qualität der Beobachtungsquellen, d. h. der Merkmale, eine Ordnung von grob nach fein existiert. In diesem Fall führt Layered Sampling zu einer im Vergleich zur einstufigen Multiplikation nach Gleichung 3.5 effektiveren Suche im

[1] Auf die entsprechende Formel zur Zerlegung der *proposal distribution* wird an dieser Stelle verzichtet, da im *Condensation*-Filter die *proposal distribution* identisch mit dem Zustandsübergangsmodell ist.

Zustandsraum, bei der jede Stufe das Ergebnis der vorherigen Stufe verfeinert. Grund hierfür ist, dass ohne Layered Sampling alle Merkmale auf der ursprünglichen a-priori Partikelmenge ausgewertet würden, die das Ergebnis der Vorhersage aus dem vergangenen Zeitschritt ist. Beim Layered Sampling hingegen wird nur das „gröbste“ bzw. das am wenigsten spezifische Merkmal auf der a-priori Partikelmenge ausgewertet, danach erfolgt bereits das erste Resampling. Das zweite Merkmal in der Reihe wird anschließend auf dieser neuen Partikelmenge ausgewertet, bei deren Bildung über das erste Merkmal bereits Information aus der Beobachtung eingeflossen ist. So werden keine Berechnungen mehr an Punkte im Zustandsraum verschwendet, die bereits von einem der vorhergehenden Merkmale abgewiesen wurden.

3.1.3 Demokratische Integration

In der Bayes'schen Formulierung des Trackings, die in dieser Arbeit verwendet wird, haben Merkmale die Funktion, den Zusammenhang $p(\mathbf{z}|\mathbf{x})$ zwischen dem Zustandsvektor $\mathbf{x}$ und der Beobachtung $\mathbf{z}$ herzustellen. Stehen verschiedene Merkmale $\{f_c(\mathbf{z}), c = 1..C\}$ zur Verfügung, stellt sich die Frage nach ihrer Kombination. In der bekannten Arbeit von [Kittler u. a., 1998] werden verschiedene Arten der Kombination (Produkt, Summe, Mehrheitsentscheid etc.) auf ihre Eigenschaften hin untersucht. Die – Unabhängigkeit der Merkmale vorausgesetzt – unter probabilistischen Gesichtspunkten korrekte Multiplikation

$$p(f_1(\mathbf{z}), \cdots, f_C(\mathbf{z})|\mathbf{x}) = \prod_{c=1..C} p(f_c(\mathbf{z})|\mathbf{x}) \tag{3.7}$$

besitzt die unerwünschte Eigenschaft, dass der Ausfall eines einzelnen Merkmals das gesamte Produkt tilgt. Robustes Tracking sollte aber im Gegensatz dazu den Ausfall einzelner Merkmale verkraften können.

In [Kittler u. a., 1998] wird gezeigt, dass die so genannte Summenregel

$$p(f_1(\mathbf{z}), \cdots, f_C(\mathbf{z})|\mathbf{x}) = \sum_{c=1..C} p(f_c(\mathbf{z})|\mathbf{x}) \tag{3.8}$$

eine gute Näherung für 3.7 ist, wenn sich die a-posteriori Wahrscheinlichkeit des Systemzustandes nur geringfügig von seiner a-priori Wahrscheinlichkeit unterscheidet, also

$$p(\mathbf{x}|f_c(\mathbf{z})) = p(\mathbf{x})(1 + \delta) \tag{3.9}$$

mit $\delta << 1$. Beim partikelfilterbasierten Tracking ist genau dies der Fall: die a-priori Wahrscheinlichkeit für das aktuelle Bild wird aus der a-posteriori Wahrscheinlichkeit des zurückliegenden Bildes gebildet (siehe Gleichung 3.3). Dabei ist die Bewegung der Ziele – also die Veränderung des Systemzustandes – zwischen zwei Bildern gering, was durch eine ausreichend hohe Framerate garantiert werden kann. Die Summenregel besitzt nun die gewünschte Eigenschaft, dass

der Ausfall eines einzelnen Merkmals den Wert der Summe zwar vermindern, nicht aber annullieren kann. Das gemeinsame Beobachtungsmodell wird damit fehlertoleranter und stabiler.

In der obigen Formulierung der Summenregel ist der Einfluss der Merkmale auf das Ergebnis noch identisch und statisch. Möchte man aber den Einfluss der einzelnen Merkmale in Gleichung 3.9 kontrollieren, bietet sich die Einführung von Gewichten r_c an, so dass

$$p(f_1(\mathbf{z}), \cdots, f_C(\mathbf{z})|\mathbf{x}) = \sum_{c=1..C} r_c \; p(f_c(\mathbf{z})|\mathbf{x}) \tag{3.10}$$

mit $\sum r_c = 1$. Die Gewichte r_c stellen ein Maß für die Verlässlichkeit des jeweiligen Merkmals dar und werden in diesem Zusammenhang *reliabilities* genannt. Die Verlässlichkeit eines Merkmals ist situationsabhängig und lässt sich im Allgemeinen nicht a-priori festlegen.

Demokratische Integration nach [Triesch und Malsburg, 2001] ist ein Verfahren zur dynamischen Anpassung der *reliabilities* der einzelnen Merkmale. Danach wird der Wert von r_c nach der Übereinstimmung des jeweiligen Merkmals mit dem tatsächlichen Systemzustand bestimmt. Da dieser nicht bekannt ist, wird als Näherung der mit allen Merkmalen gemeinsam geschätzte Zustand betrachtet. Die Intuition hinter dem Verfahren ist die Erfahrung, dass die gemeinsame Schätzung stabiler ist als die Einzelschätzungen der jeweiligen Merkmale. Gibt es unter den Merkmalen Ausreißer, können diese identifiziert und in ihrem Einfluss gedämpft werden. Das Verfahren kann als selbstorganisierend bezeichnet werden, da die Entscheidung über die Gewichtung allein auf den ohnehin vorhandenen Informationen basiert, und keine neuen, außenstehenden Wissens- oder Entscheidungsquellen benötigt werden[2].

Zur Berechnung von r_c wird als Hilfsvariable das so genannte Qualitätsmaß q_c eingeführt. Werte für q_c nahe 1 repräsentieren dabei eine hohe Übereinstimmung des Merkmals mit dem gemeinsamen Ergebnis, Werte nahe 0 stehen für geringe Übereinstimmung. Die *reliabilities* werden nach jedem Zeitschritt mittels eines *leaky integrator* aus den normalisierten Qualitäten berechnet, so dass

$$\tau \dot{r}_c = \frac{q_c}{\sum_c q_c} - r_c, \tag{3.11}$$

wobei der Parameter τ die Geschwindigkeit der Anpassung an q_c beeinflusst.

Der Arbeit von [Triesch und Malsburg, 2001] liegt der Gedanke zugrunde, dass jedes Merkmal eine *saliency map* im Bildraum erzeugt, in der für jeden Bildpunkt die Unterstützung durch das jeweilige Merkmal eingetragen ist. Die ein-

[2] Gäbe es außerhalb der verwendeten Merkmale weitere externe Wissensquellen bzw. Heuristiken, dann würde sich die Frage stellen, wieso diese der Bestimmung der Gewichtung vorbehalten sein sollen, anstatt ihrerseits als reguläres Merkmal zum Tracking beizutragen.

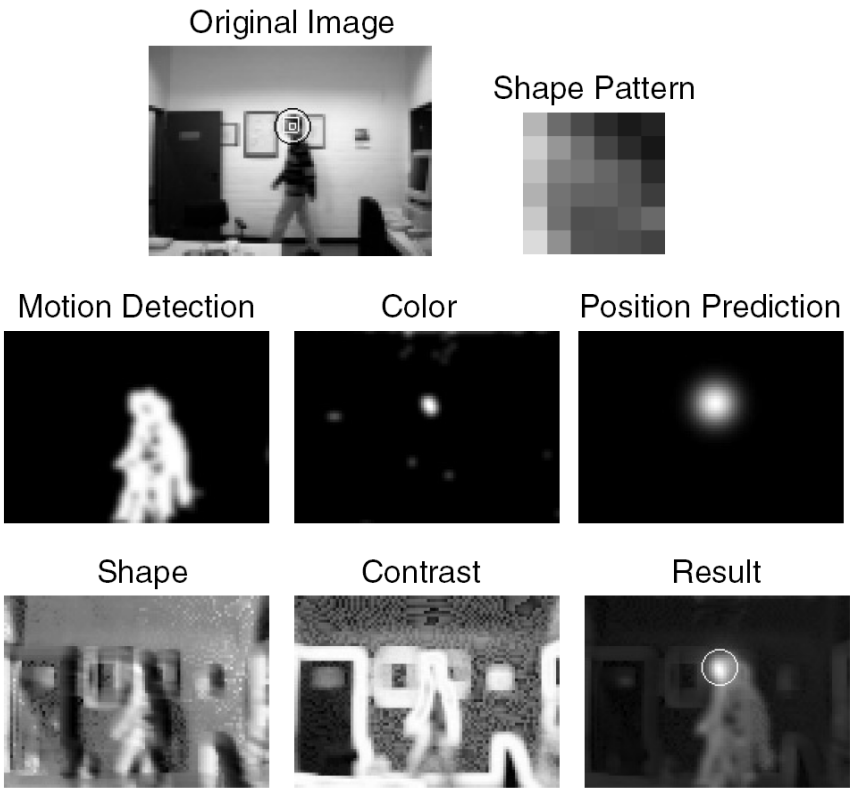


Abbildung 3.5: Demokratische Integration von *saliency maps* verschiedener Merkmale, Quelle [Triesch und Malsburg, 2001].

zelnen *saliency maps* werden dann als gewichtete Summe zu einer vereinten *saliency map* kombiniert, wobei die *reliabilities* die Mixturgewichte der jeweiligen Merkmale darstellen (siehe Abbildung 3.5).

Zur eigentlichen Berechnung der Qualität q_c werden von [Triesch und Malsburg, 2001] folgende Maße vergleichend vorgeschlagen. Dabei stehe $A_c(\mathbf{x})$ für den Durchschnittswert der *saliency map* von Merkmal c in einer Umgebung um die Koordinaten $\mathbf{x}$, und $\hat{\mathbf{x}}$ stehe für die gemeinsame Hypothese, also für die Koordinaten, in deren Umgebung der Wert der vereinten *saliency map* maximal ist.

- Uniform: $q_c = \frac{1}{C}$, dieses Maß stellt die Vergleichsgrundlage dar.
- Direkt: $q_c = A_c(\hat{\mathbf{x}})$
- Normalisiert: $q_c = \dfrac{A_c(\hat{\mathbf{x}})}{\sum_{\mathbf{x}} A_c(\mathbf{x})}$
- Distanz zum Durchschnitt: $q_c = A_c(\hat{\mathbf{x}}) - \bar{A}_c$ wobei $\bar{A}_c$ der Durchschnittswert der *saliency map* ist, und q_c nach unten durch den Wert 0 beschränkt wird.

- Korrelation: Punktweise Korrelation von A_c mit der vereinten *saliency map*. Hierbei kommt der gemeinsamen Hypothese $\hat{\mathbf{x}}$ keine Bedeutung zu.

Die Berechnung der gemeinsamen Hypothese $\hat{\mathbf{x}}$, also das eigentliche Tracking, erfolgt in der Formulierung von [Triesch und Malsburg, 2001] in Form einer Suche über die gesamte *saliency map*. Im folgenden Abschnitt wird der DI^2-Algorithmus entwickelt, in dem unter anderem das Konzept der DI so verändert wird, dass die Suche effizient mithilfe eines Partikelfilters durchgeführt und somit z. B. auch auf räumliches Tracking ausgedehnt werden kann.

3.2 Der DI^2-Algorithmus zur Merkmalsfusion

Im vorangegangenen Abschnitt wurden die Grundlagen des Trackings mit Partikelfiltern sowie die dynamische Merkmalsfusion nach dem Prinzip der Demokratischen Integration (DI) beschrieben. In ihrer ursprünglichen Formulierung ist die DI allerdings nicht im Partikelfilter einsetzbar, da die vorgeschlagenen Qualitätsmaße nicht mit einer Partikelmenge, also mit mehreren Hypothesen, funktionieren.

Um beide Verfahren dennoch kombinieren zu können, wird im ersten Unterabschnitt ein neues Qualitätsmaß für die DI von Merkmalen vorgeschlagen, das die gesamte Partikelmenge berücksichtigt, und so die Verbindung möglich macht. Anschließend wird kurz auf die Notwendigkeit einer adaptiven Skalierung der Merkmalswerte vor ihrer Kombination eingegangen. Im dritten Unterabschnitt wird der Wettbewerb zwischen den einzelnen Merkmalstypen ausgedehnt auf einen Wettbewerb, der zusätzlich zu den Merkmalstypen auch die unterschiedlichen Regionen des Zielobjekts umfasst. Dies führt zum so genannten DI^2-Algorithmus, der im Anschluss formuliert wird. Der Name soll ausdrücken, dass hier die Demokratische Integration (DI) auf ein zweidimensionales Feld von Merkmalen ausgedehnt wird, bei dem die eine Dimension für den Merkmalstyp und die andere Dimension für die Zielregion/Kameraansicht steht.

3.2.1 Ein Qualitätsmaß für Merkmale im Partikelfilter

In der ursprünglichen Arbeit von [Triesch und Malsburg, 2001] erfolgt das Tracking zu jedem Zeitschritt als erschöpfende Maximumsuche über einer *saliency map* im Bildraum, die aus den Beiträgen dedizierter *saliency maps* für jedes der Merkmale zusammengesetzt ist. Das Qualitätsmaß eines Merkmals bemisst sich dann entweder nach den Werten der jeweiligen *saliency map* in der Region, in der die gemeinsame Endhypothese liegt, oder alternativ aus dem Vergleich einer Hypothese, die nur auf einer *saliency map* beruht, mit der gemeinsamen Hypothese.

In [Shen u. a., 2003] wurde versucht, die zweite Variante auf partikelfilterbasiertes Tracking zu übertragen: Basierend auf der aktuellen Partikelmenge $\mathbf{s}^{(n)}$ und einer merkmalsabhängigen Menge von Gewichten $\pi_c^{(n)} \propto p(f_c(\mathbf{z})|\mathbf{x} = \mathbf{s}^{(n)})$ wird eine merkmalsabhängige Hypothese $\hat{\mathbf{x}}_c$ gemäß Gleichung 3.4 gebildet. Diese wird mit der gemeinsamen Hypothese $\hat{\mathbf{x}}$ verglichen. Die Distanz wird mithilfe einer Sigmoidfunktion normalisiert und dient dann als Qualitätsmaß:

$$q_c = \frac{tanh(-a|\hat{\mathbf{x}}_c - \hat{\mathbf{x}}| + b) + 1}{2} \tag{3.12}$$

mit manuell festgelegten Konstanten a und b. Sinkt die Distanz zwischen $\hat{\mathbf{x}}_c$ und $\hat{\mathbf{x}}$, dann steigt der Wert für q_c.

Obwohl diese Formulierung einleuchtend erscheint, gibt es trotzdem ein Schwierigkeit, wenn man sie auf das Trackingproblem anwendet: Angenommen sei der häufig auftretende Fall, dass sich das Ziel für eine Weile nicht bewegt hat, so dass – bedingt durch das Resampling – die Partikel symmetrisch um den Systemzustand herum verteilt liegen. Sei zudem angenommen, dass ein Merkmal in der aktuellen Situation wenig oder keine Unterstützung im Bild findet, und daher der Partikelmenge eine uniforme Wahrscheinlichkeitsverteilung zuweist. Als Konsequenz werden in diesem Fall sowohl die merkmalsabhängige Hypothese $\hat{\mathbf{x}}_c$ als auch die gemeinsame Hypothese $\hat{\mathbf{x}}$ dicht am Mittelwert der Partikelverteilung liegen. Dies führt nach 3.12 zu einem hohen Wert von q_c, obwohl das entsprechende Merkmal in Wirklichkeit überhaupt nicht in der Lage ist, das Ziel zu lokalisieren.

Um dieses Problem zu beseitigen, bedarf es eines neuen Qualitätsmaßes, das über [Triesch und Malsburg, 2001] und [Shen u. a., 2003] hinausgeht. Das Maß soll zwei Kriterien genügen: es soll quantifizieren, wie stark sich die merkmalsabhängige Wahrscheinlichkeitsmasse um die gemeinsame Hypothese $\hat{\mathbf{x}}$ herum ballt, und es soll quantifizieren, wie stark die Unterstützung des Merkmals am Ort der gemeinsamen Hypothese ist.

Das inverse quadratische Mittel $(\sum_n \pi_c^{(n)}|\mathbf{s}^{(n)} - \hat{\mathbf{x}}|_2^2)^{-1}$ der Partikelmenge erfüllt das erste Kriterium. Sein Wert ist allerdings nicht nur von den merkmalsabhängigen Gewichten $\pi_c^{(n)}$ abhängig, sondern unerwünschterweise auch von der merkmalsunabhängigen Streuung der Partikelmenge. Diese Abhängigkeit kann eliminiert werden, indem man das Merkmal zu einem hypothetischen Merkmal in Beziehung setzt, das allen Partikeln uniforme Gewichte $\frac{1}{N}$ zuweist. Siehe hierzu das Beispiel in Abbildung 3.6.

Das zweite Kriterium, die Frage nach dem Absolutwert der Unterstützung am Zielort, wird erfüllt, indem die nicht-normalisierte Ausgabe $p(f_c(\mathbf{z})|\hat{\mathbf{x}})$ des Merkmals an der Stelle der gemeinsamen Hypothese betrachtet wird. Dies ist wichtig, da ein Merkmal, das zwar in Richtung auf das Ziel konvergiert, dabei aber nur sehr geringe absolute Werte liefert, keine gute Unterstützung für das Tracking ist, und somit auch keine hohe Qualität zugewiesen bekommen sollte.

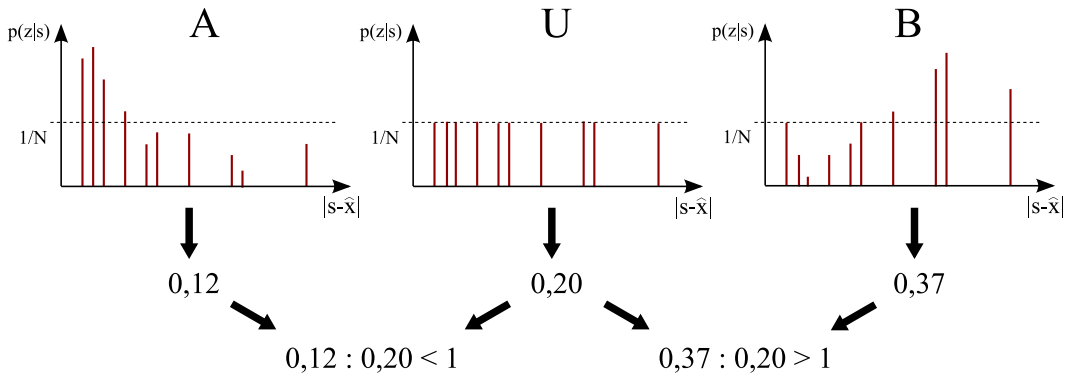


Abbildung 3.6: Vergleich zwischen einem guten (A) und einem schlechten (B) Merkmal sowie einem hypothetischen (U) Merkmal, das eine uniforme Ausgabe liefert. Die Merkmale sind dargestellt in Form der normalisierten Gewichte, die sie den Partikeln zuweisen. Die Partikel sind sortiert in Bezug auf ihre Entfernung zur gemeinsamen Hypothese $\hat{\mathbf{x}}$ auf einem Strahl aufgetragen. Für jedes Merkmal wird das quadratische Mittel berechnet. Ist es kleiner als das von (C), handelt es sich um ein gutes Merkmal, ist es größer, handelt es sich um ein schlechtes Merkmal.

Zusammengenommen führt dies zu der folgenden Formulierung eines universellen Qualitätsmaßes für Merkmale im Kontext partikelfilterbasierten Trackings:

$$q_c = \left(\frac{\sum_{n=1..N} \frac{1}{N} |\mathbf{s}^{(n)} - \hat{\mathbf{x}}|_2^2}{\sum_{n=1..N} \pi_c^{(n)} |\mathbf{s}^{(n)} - \hat{\mathbf{x}}|_2^2} \right)^{\lambda} p(f_c(\mathbf{z})|\hat{\mathbf{x}}) \tag{3.13}$$

Mit dem Exponenten $\lambda > 0$ kann die Volatilität des Qualitätsmaßes eingestellt werden.

3.2.2 Adaptive Skalierung

Bei der Kombination verschiedener Merkmale gilt es zu berücksichtigen, dass die Wertebereiche der einzelnen Merkmale je nach ihrer Implementierung unterschiedliche Schwankungsbreiten aufweisen. So könnte ein Merkmal für Bildregionen, die seinem Modell entsprechen, z. B. den zehnfachen Wert dessen liefern, was es für Bildregionen tut, die seinem Modell nicht entsprechen. Bei einem anderen Merkmal hingegen mag diese Schwankungsbreite nur das Fünffache betragen – und das, obwohl beide Merkmale gleichermaßen spezifisch und damit nützlich sind.

In der Formulierung des *Condensation*-Algorithmus werden die direkten Ausgaben der Merkmale für die Partikelmenge so normalisiert, dass ihre Summe 1

beträgt, was von der Form her einer Wahrscheinlichkeitsverteilung entspricht. Das Problem der unterschiedlichen Volatilität der Merkmale wird so allerdings nicht gelöst.

Bei DI^2 kommt der Volatilität der Merkmalswerte eine doppelte Bedeutung zu, da sie nicht nur der Bestimmung des kombinierten Ergebnisses dienen, sondern zusätzlich auch, wie im vorangegangenen Abschnitt beschrieben, zur Bestimmung der Merkmalsqualität benötigt werden. Die Wertebereiche sämtlicher Merkmale werden daher vor ihrer Verwendung skaliert; entscheidend für die Skalierung ist dabei die Schwankungsbreite, die das jeweilige Merkmal auf der Beobachtung zeigt. Dazu werden Mittelwert μ_c und Standardabweichung σ_c des Merkmals c auf einer größeren Zahl zufällig gewählter Bildausschnitte bestimmt. Damit wird das Merkmal dann wie folgt skaliert:

$$p(f_c(\mathbf{z})|\mathbf{x}) \leftarrow max\left(\frac{S_c(\mathbf{z},\mathbf{x}) - \mu_c}{\sigma_c}, 0\right) \tag{3.14}$$

Dabei bezeichne $S_c(\mathbf{z},\mathbf{x})$ die tatsächliche Bewertungsfunktion von Merkmal c. Die Statistiken μ_c und σ_c werden dabei langsam, aber stetig der aktuellen Beobachtung angepasst, da sich das Verhalten eines Merkmals in Abhängigkeit von der Situation ändern kann.

3.2.3 Verallgemeinerung des Merkmalsbegriffes

Für eine erfolgreiche Kombination sollten sich die verwendeten Merkmale möglichst orthogonal zueinander verhalten, d. h. dass verschiedene Merkmale in jeweils unterschiedlichen Situationen funktionieren bzw. versagen. Eine Möglichkeit, dies zu erreichen, ist es, wie bisher auch vorausgesetzt, unterschiedliche Transformationen $f_c(\mathbf{z})$ der Kamerabilder $\mathbf{z}$ zu verwenden:

$$p(\mathbf{z}|\mathbf{x}) = \sum_{c=1..C} p(f_c(\mathbf{z})|\mathbf{x}) \tag{3.15}$$

Darunter fallen Merkmalstypen wie Bewegung (Hintergrundsubtraktion, optischer Fluss, etc.), Farbe (Histogrammrückprojektion, Bhattacharyyadistanz, etc.) oder Form (Konturvergleich, Haar-Feature-Detektoren, etc.) Versagt einer dieser Merkmalstypen, kann dies in der Regel durch die anderen Merkmalstypen kompensiert werden.

Eine zweite Möglichkeit, orthogonale Merkmale zu erzeugen, ist es, unterschiedliche Projektionen $g_c(\mathbf{x})$ des Zustandsvektors $\mathbf{x}$ in den Bildbereich zu erzeugen:

$$p(\mathbf{z}|\mathbf{x}) = \sum_{c=1..C} p(\mathbf{z}|g_c(\mathbf{x})) \tag{3.16}$$

Sinnvoll ist dies in Situationen, in denen Merkmale, die sich auf bestimmte Teilregionen des Zielobjekts beziehen, weiterhin anwendbar sind, während andere

Merkmale, die sich auf andere Teilregionen beziehen, beispielsweise durch Verdeckung ausfallen. In diesem Fall handelt es sich bei den unterschiedlichen Merkmalen um unterschiedliche Partitionen des Zielobjekts bzw. um unterschiedliche Tracking-Kernel. Stehen mehrere Kameraansichten zur Verfügung, dann beinhaltet $g_c(\mathbf{x})$ auch die Auswahl der Kamera, in deren Bild $\mathbf{x}$ zur Auswertung projiziert wird. Das ist wichtig, da unterschiedliche Kameraansichten aufgrund von Perspektive oder von Verdeckung unterschiedlich gut geeignet sein können, um das Zielobjekt zu unterstützen.

In der vorliegenden Arbeit werden die Vorteile beider Strategien kombiniert: Merkmale im verallgemeinerten Sinne sind nun eine Kombination $m_c = \{f_c, g_c\}$ eines bestimmten Merkmalstyps $f_c(\mathbf{z})$ mit einer bestimmten Projektion des Zustandsvektors $g_c(\mathbf{x})$:

$$p(\mathbf{z}|\mathbf{x}) = \sum_{c=1..C} p(f_c(\mathbf{z})|g_c(\mathbf{x})), \tag{3.17}$$

Alle Merkmale aus der verallgemeinerten Menge $\{m_c\}$ treten bei der Merkmalsfusion nach Abschnitt 3.1.3 gleichberechtigt gegeneinander an. Dadurch können die selbstorganisierenden Fähigkeiten der Demokratischen Integration automatisch sowohl die Merkmalstypen als auch die Zielregionen/Kameraansichten auswählen, die in der aktuellen Situation am geeignetsten sind.

3.2.4 Formulierung des DI²-Algorithmus

Merkmalsfusion mittels Demokratischer Integration (DI), wie in Abschnitt 3.1.3 beschrieben, ist geeignet für Merkmale, die optional für die Existenz des Zielobjekts sind, d. h. das Zielobjekt kann das jeweilige Merkmal aufweisen, muss es aber nicht. Zusätzlich kann es jedoch auch notwendige Merkmale geben, die unbedingt erfüllt sein müssen, und deren Einfluss daher durch die dynamische Merkmalsfusion nicht eliminiert werden darf. Diese notwendigen Merkmale müssen aus dem dynamischen Wettbewerb herausgenommen werden und könnten z. B. multiplikativ dem Beobachtungsmodell aus Gleichung 3.10 hinzugefügt werden.

Wie allerdings in Abschnitt 3.1.2 ausgeführt, gibt es für Partikelfilter mit dem Layered Sampling eine bessere Alternative als die Multiplikation: So können die notwendigen Merkmale in einer eigenen Samplingstufe vor den optionalen Merkmalen ausgewertet werden. Durch das jeweils zwischen zwei Stufen stattfindende Resampling wird die a-priori Verteilung für die folgende Stufe auf jene Bereiche im Zustandsraum reduziert, die die vorherige Stufe akzeptiert hat. Die Ausnutzung der begrenzten Anzahl von Partikeln – und somit die Effizienz des Trackings – ist bei dieser gestuften Vorgehensweise höher, da in der folgenden Stufe keine Partikel mehr an Regionen verschwendet werden, die in der vorherigen Stufe als aussichtslos erkannt wurden; so stehen mehr Partikel für vielversprechende Regionen zur Verfügung.

Dies bedeutet, dass sich im Allgemeinen die aus C Merkmalen bestehende Merkmalsmenge in K disjunkte Teilmengen zerlegen lässt, die jeweils notwendig aufeinander aufbauen:

$$\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_K = \{1, ..., C\} \tag{3.18}$$

Innerhalb der einzelnen Merkmalsmengen kann es im allgemeinen Fall auch wieder Gelegenheit zu dynamischem Wettbewerb geben, wenn nur mindestens eines der Merkmale einer Teilmenge zwingend erfüllt sein muss. Sollte ein einzelnes Merkmal tatsächlich unabdingbar sein, kann es eine eigene Teilmenge mit $|\mathcal{C}_i| = 1$ bilden. Ein konkretes Beispiel für solche Merkmalsmengen findet sich später in Abschnitt 3.4.1.

Abbildung 3.7 zeigt den daraus entstehenden Algorithmus zur dynamischen Merkmalsfusion im Überblick, der im Folgenden als DI^2-Algorithmus bezeichnet wird. Der Name soll ausdrücken, dass in ihm die Demokratische Integration (DI) auf ein zweidimensionales Feld von Merkmalen ausgedehnt wird, bei dem die eine Dimension für den Merkmalstyp und die andere Dimension für die Zielregionen/Kameraansichten steht. Im Gegensatz zum konventionellen DI-Algorithmus aus [Triesch und Malsburg, 2001] besitzt der DI^2-Algorithmus dadurch eine Reihe von Vorteilen:

- Explizites Multi-Hypothesen-Tracking durch Integration in den Partikelfilter (Abschnitt 3.2.1)
- Automatische Auswahl des besten Merkmalstyps und zugleich der besten Zielregion bzw. Kameraansicht (Abschnitt 3.2.3)
- Effizientere Auswertung der Merkmale durch mehrstufiges Sampling. (Insbesondere ist keine erschöpfende Suche über das gesamte Bild mehr erforderlich.)

3.3 Merkmale zum Personentracking

Der humanoide Roboter Armar-III verfügt über einen Stereokamerakopf und bordeigene PCs mit eng begrenzter Rechenleistung. Als Konsequenz müssen einfache Merkmale verwendet werden, die mit sehr geringem Rechenaufwand ausgewertet werden können. Da die einzelnen Merkmale unter diesen Umständen für sich genommen nicht sehr diskriminativ sein können, kommt dem Fusionsalgorithmus eine besonders große Bedeutung zu.

Die in den folgenden Abschnitten vorgestellten Merkmalstypen basieren auf Bewegungsbildern, Farbhistogrammen, Haar-feature-Detektoren und Stereokorrelation. Wie gezeigt werden wird, lassen sie sich alle effizient und skalierungsunabhängig mit konstantem Aufwand berechnen. Abbildung 3.8 zeigt die unterschiedlichen Merkmalstypen anhand eines Ausschnittes aus einer Testsequenz.

Initialisierung:

- $\mathbf{s}_t^{0,(1..n)}, \pi_t^{0,(1..n)} = \mathbf{s}_{t-1}^{(1..n)}, \pi_{t-1}^{(1..n)}$

Wiederhole für $k = 1..K$**:**

- Resampling von $\mathbf{s}_t^{k-1,(1..n)}$ gemäß $\pi_t^{k-1,(1..n)}$
- Propagiere mit partiellem Zustandsübergangsmodell (nach 3.6)
  $\mathbf{s}_t^{k,(1..n)} \longleftarrow p_k(\mathbf{s}_t^{k,(1..n)} | \mathbf{s}_t^{k-1,(1..n)})$
- Auswertung der k-ten Merkmalsmenge:
  $\pi_t^{k,(1..n)} \propto \sum_{c \in \mathcal{C}_k} r_c p_c(\mathbf{z} | \mathbf{s}_t^{k,(1..n)})$

Abschluss:

- $\mathbf{s}_t^{(1..n)}, \pi_t^{(1..n)} = \mathbf{s}_t^{K,(1..n)}, \pi_t^{K,(1..n)}$
- Berechnung der Hypothese $\hat{\mathbf{s}}_t = \sum_i \pi_t^{(i)} \mathbf{s}_t^{(i)}$
- Aktualisierung der reliabilities für $k = 1..K$ (nach 3.11 und 3.13)
  $r_{c \in \mathcal{C}_k} \longleftarrow \hat{\mathbf{s}}_t, \mathbf{s}_t^{k,(1..n)}, \pi_t^{k,(1..n)}$

Abbildung 3.7: Der DI²-Algorithmus.

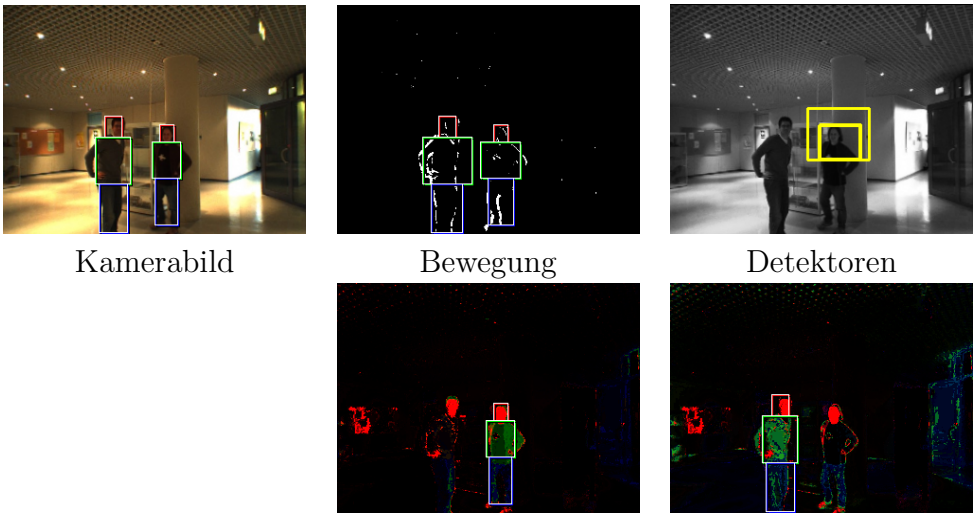


Abbildung 3.8: Übersicht der verwendeten Merkmalstypen. In dieser Darstellung sind die Rückprojektionen der Farbmodelle für Kopf, Torso und Beine in den rot/grün/blau-Kanälen des Visualisierungsbildes zusammengefasst. Die Berechnung der Stereokorrelation erfolgt punktweise für jede Hypothese und ist daher nicht als Bild darstellbar.

Wie in Abschnitt 3.2.3 ausgeführt, werden im Verbund mit den unterschiedlichen Merkmalstypen auch unterschiedliche Regionen des Zielobjektes als gleichberechtigte Merkmale betrachtet: Einige Merkmale konzentrieren sich dabei auf die menschliche Kopfregion, andere auf den Torso oder auf die Beinregion. Diese Regionen werden durch ein Quadermodell des menschlichen Körpers festgelegt, das in Abbildung 3.9 dargestellt ist. Aufpunkt des Modells ist der Kopfmittelpunkt. Die Ausmaße der drei Quader orientieren sich an den Abmessungen eines durchschnittlichen Menschen, die Körperhöhe – gegeben durch den Kopfmittelpunkt – ist dabei ein freier Parameter.

Durch Kombination der verschiedenen Merkmalstypen mit den drei Körperregionen ergeben sich 13 verschiedene Merkmale, die in den kommenden Abschnitten näher beschrieben werden. Dabei bezeichne $\mathcal{A}(\mathbf{x})$ im Folgenden eine der Bildregionen aus Abbildung 3.9, $|\mathcal{A}(\mathbf{x})|$ ihre Größe, und $\sum_{\mathcal{A}(\mathbf{x})} f_c(\mathbf{z})$ die Summe der Pixelwerte des Merkmalstyps f_c innerhalb $\mathcal{A}(\mathbf{x})$. Da alle Regionen durch rechtwinklige bounding boxes approximiert werden, können die Pixelsummen innerhalb der Regionen effizient mit konstantem Aufwand durch das Integralbild, siehe [Viola und Jones, 2001], berechnet werden.

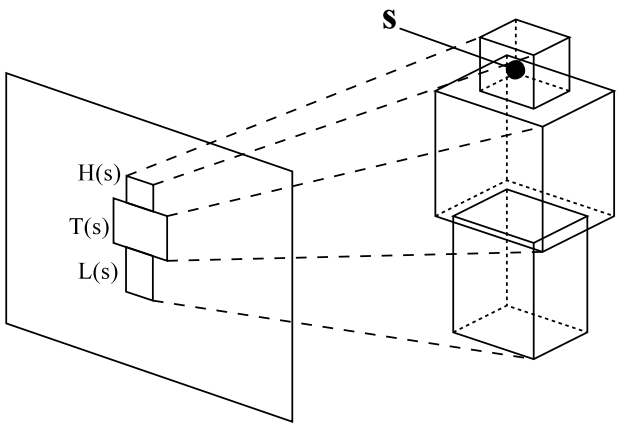


Abbildung 3.9: Quadermodell des menschlichen Körpers: der Zustandsvektor $\mathbf{x}$ wird in den Bildraum transformiert als Projektion entweder des Kopfes, des Torsos oder der Beine. Das Abbild des Quaders wird durch eine rechtwinklige bounding box approximiert.

3.3.1 Bewegung

Aufgrund der Eigenbewegung der Roboterkamera ist der Aufbau eines längerfristigen Hintergrundmodells nicht möglich. Als Alternative dazu bietet sich ein sehr kurzzeitiges Hintergrundmodell an, das zumindest in Phasen der Bewegungslosigkeit des Roboterkopfes bewegte Objekte vom Hintergrund trennen kann. Ein Extremfall eines solchen Kurzzeitmodells ist das Differenzbild, das aus der absoluten Differenz zwischen dem aktuellen und dem vergangenen Intensitätsbild entsteht, und mit einem Schwellwert binarisiert wird. Es setzt eine statische Kamera lediglich für die kurze Zeit zwischen zwei Bildern voraus.

Für ein bewegtes Objekt sind hohe Werte des Differenzbildes $\mathcal{M}(\mathbf{z})$ in der entsprechenden Bildregion zu erwarten. Die Beobachtungswahrscheinlichkeit für das Bewegungsmerkmal wird daher festgelegt als

$$p(\mathcal{M}(\mathbf{z})|\mathcal{A}(\mathbf{x})) \propto \frac{\sum_{\mathcal{A}(\mathbf{x})} \mathcal{M}(\mathbf{z})}{|\mathcal{A}(\mathbf{x})|} \cdot \frac{\sum_{\mathcal{A}(\mathbf{x})} \mathcal{M}(\mathbf{z})}{\sum \mathcal{M}(\mathbf{z})} \tag{3.19}$$

Der linke Faktor strebt dabei nach Maximierung von Bewegung innerhalb der durch $\mathcal{A}(\mathbf{x})$ gegebenen Bildregion, während der rechte Faktor danach strebt, möglichst viel Bewegung im gesamten Bild abzudecken. Der rechte Faktor verhindert somit, dass das Bewegungsmerkmal winzige Teilregionen voller Bewegung bevorzugt.

Beim Personentracking werden insgesamt drei Bewegungsmerkmale eingesetzt, im Folgenden als M-H, M-T und M-L bezeichnet, die sich auf die Kopf-, die Torso- und die Beinregion konzentrieren, wie es in Abbildung 3.9 dargestellt ist.

Die Bewegungsmerkmale können nur dann sinnvoll eingesetzt werden, wenn sich die Kamera zwischen zwei Bildern nicht bewegt. Wenn große Bildteile, bedingt durch die Eigenbewegung des Roboters, den Schwellwert überschreiten, sind die Bewegungsmerkmale nicht mehr aussagekräftig im Bezug auf das zu verfolgende Ziel. Zwar verfügt der Roboter über Sensoren, die seine Position und Lage messen, allerdings sind diese nicht genau und schnell genug, um z. B. Schwingungen des Kamerakopfes zu erfassen, die sich in den Bildern deutlich abzeichnen. Dank der dynamischen Merkmalsfusion aus Abschnitt 3.2.4 ist die Verwendung der Sensoren aber auch gar nicht notwendig: Der DI^2-Algorithmus ist in der Lage zu erkennen, wann die Bewegungsmerkmale keinen sinnvollen Beitrag mehr zum Tracking liefern, und kann ihren Einfluss auf das Ergebnis in diesen Fällen selbstständig herunterregeln. Das Gleiche passiert auch im umgekehrten Fall, wenn sich das Ziel nicht bewegt und somit im Differenzbild nicht erkennbar ist.

3.3.2 Farbe

Farbmodelle sind zum einen in der Lage, Personen vom Hintergrund zu segmentieren, und können zum anderen – als einziges der verwendeten Merkmale – verschieden gekleidete Personen voneinander unterscheiden. Farbmodelle sind translations- und rotationsinvariant, was zunächst einmal wünschenswerte Eigenschaften für das Tracking sind. Durch ihre Bekleidung weisen Menschen allerdings oft eine typische lokale Farbverteilung (z. B. schwarze Jacke über blauer Hose) auf, die durch ein globales Farbmodell pro Person nicht abgebildet werden kann. Daher werden im Folgenden drei dedizierte Farbmerkmale für die drei Körperregionen Kopf, Torso und Beine verwendet und mit C-H, C-T und C-L bezeichnet. Die Translations- und Rotationsunabhängigkeit der Farbmodelle wird also innerhalb der jeweiligen Regionen ausgenutzt, ihre räumliche Anordnung zueinander ist allerdings durch die menschliche Anatomie festgelegt.

Jedes der drei Farbmerkmale beinhaltet ein Histogramm mit 16^3 Töpfen, das eine Verteilung im RGB-Farbraum modelliert. Die Farbmerkmale passen ihre Modelle ständig mit dem in Abschnitt 3.4.6 beschriebenen Verfahren an die aktuelle Zielregion an. Dabei wird immer auch ein zweites Histogramm erstellt, das so genannte Hintergrundhistogramm, das die Farbverteilung des gesamten Kamerabildes repräsentiert[3]. Das neu gebildete Histogramm der Zielregion wird durch das Hintergrundhistogramm dividiert, wodurch das Quotientenhistogramm entsteht, welches das eigentliche Modell des Farbmerkmals darstellt.

[3]Dabei wird angenommen, dass die Vordergrundregion im Vergleich zur Gesamtgröße des Bildes klein ist, so dass das Histogramm des gesamten Bildes näherungsweise mit einem Histogramm des reinen Hintergrundes übereinstimmt.

Die *support map* $\mathcal{C}(\mathbf{z})$ für das Farbmerkmal entsteht dann durch Histogramm-Rückprojektion auf das Kamerabild. Die Beobachtungswahrscheinlichkeit für das Farbmerkmal ist dann analog zu 3.19 gegeben durch

$$p(\mathcal{C}(\mathbf{z})|\mathcal{A}(\mathbf{x})) = \frac{\sum_{\mathcal{A}(\mathbf{x})} \mathcal{C}(\mathbf{z})}{|\mathcal{A}(\mathbf{x})|} \cdot \frac{\sum_{\mathcal{A}(\mathbf{x})} \mathcal{C}(\mathbf{z})}{\sum \mathcal{C}(\mathbf{z})} \tag{3.20}$$

3.3.3 Detektoren

Haar-feature-Klassifikatoren nach [Viola und Jones, 2001] erzielen bei zur Laufzeit geringem Rechenaufwand gute Ergebnisse z. B. bei der Detektion von Gesichtern. Die Komplexität der verwendeten Merkmale ist skalierungsunabhängig, da diese effizient mithilfe des Integralbildes berechnet werden. Bei der üblichen Anwendung von Haar-feature-Kaskaden wird ein Suchfenster wiederholt in verschiedenen Größen über das Bild geschoben. Der Fensterinhalt wird bei jedem Schritt klassifiziert und dicht nebeneinander liegende Detektionen zu einer einzigen kombiniert. Eine auf diese Weise durchgeführte Suche in einem $W \times W$ großen Bild mit einem $F \times F$ großen Detektor, der n mal mit einem Skalierungsfaktor s vergrößert wird, benötigt die folgende Anzahl an Klassifikationen:

$$\#Klassifikationen = \sum_{i=0}^{n-1} (W - F \cdot s^i)^2 \tag{3.21}$$

Bei einer Bildregion von beispielsweise 100×100 Pixel und einer Gesichtsgröße zwischen 20 und 42 Pixel Kantenlänge (wie bei $n = 8, s = 1,1$) verursacht dies einen Aufwand von 44368 einzelnen Klassifikationen.

Im hier vorgeschlagenen Verfahren ist es allerdings nicht notwendig, das Bild flächendeckend zu durchsuchen. Da die propagierte Verteilung der Partikel aus dem zurückliegenden Zeitschritt als a-priori Verteilung für den aktuellen Zeitschritt verfügbar ist (siehe Gleichung 3.3), sind die wahrscheinlichsten Suchregionen implizit durch die Partikelmenge gegeben: Für jedes Partikel wird eine kopfgroße Region $\mathcal{A}(\mathbf{x})$ ins Bild projiziert und mit einer einzigen Klassifikation überprüft. Die Anzahl der Klassifikationen ist daher tatsächlich nur noch so hoch wie die Anzahl N der Partikel (typische Werte für N sind $50 - 300$.)

Die Klassifikatoren aus [Viola und Jones, 2001] sind in Stufen organisiert, die eine nach der anderen durchlaufen werden müssen, um am Ende zu einem positiven Klassifikationsergebnis zu gelangen. Das Verhältnis

$$\Gamma(\mathcal{A}(\mathbf{x})) = (\tfrac{\#\text{ durchlaufene Stufen}}{\#\text{ Stufen gesamt}})^{\omega} \tag{3.22}$$

kann daher als Konfidenzmaß der Detektion aufgefasst werden, wobei der Exponent ω den Abfall des Konfidenzwertes durch jede nicht passierte Stufe quantifiziert.

Um die Wahrscheinlichkeitsverteilung des Detektormerkmales zu glätten, werden die Klassifikationsergebnisse benachbarter Partikel miteinbezogen. Die Beobachtungswahrscheinlichkeit des Detektionsmerkmals an der Stelle $\mathbf{x}$ ergibt sich aus der stärksten Überlappung zwischen ihrer eigenen Bildregion $\mathcal{A}(\mathbf{x})$ und allen anderen positiv klassifizierten Regionen $\mathcal{A}'$ aus der Partikelmenge $s^{(n)}$.

$$\mathcal{A}' \in \{\mathcal{A}(\mathbf{x} = s^{(n)}) \mid \mathcal{A}(\mathbf{x} = s^{(n)}) \text{ positiv}\} \tag{3.23}$$

Die Beobachtungswahrscheinlichkeit des Detektormerkmals ist daher formuliert als

$$p(\mathcal{D}(\mathbf{z})|\mathcal{A}(\mathbf{x})) = \max_{\mathcal{A}'}\Gamma(\mathcal{A}') \cdot \Delta(\mathcal{A}', \mathcal{A}(\mathbf{x})), \tag{3.24}$$

wobei $\Delta(\cdot, \cdot)$ eine Distanzmetrik, basierend auf dem Überlappungsgrad zweier Bildregionen, darstellt.

Insgesamt werden zum Personentracking vier Detektormerkmale eingesetzt: eines für frontale Gesichter (D-F), je eines für linke (D-L) und rechte (D-R) Gesichtsprofile und eines für Oberkörper (D-U).[4]

3.3.4 Stereokorrelation

Das klassische Merkmal der Stereobildverarbeitung ist die Disparitätenkarte (*dense disparity map*), wie z. B. in [Scharstein und Szeliski, 2002] beschrieben. Sie entsteht durch flächendeckende Suche nach lokaler Korrelation entlang der Epipolarlinien der beiden Stereokamerabilder und wird üblicherweise noch durch nachträgliche Filterschritte verfeinert. Die Berechnung der Disparitätenkarte ist durch die flächendeckende Suche komplex und wäre im Vergleich zu den anderen in dieser Arbeit verwendeten Merkmalen deutlich zeitaufwändiger, sofern die Berechnung nicht durch Spezialhardware unterstützt wird. Ein zweites, fundamentaleres Problem als die Geschwindigkeit ist jedoch die Wahl der Größe des Korrelationsfensters: Durch ein zu großes Fenster gehen Details verloren, während durch ein zu kleines Fenster ein verrauschtes Ergebnis entsteht.

Im hier gewählten Ansatz werden beide Probleme umgangen: zum einen kann – ähnlich wie bei den Detektoren in Abschnitt 3.3.3 – auf die flächendeckende Suche nach Korrelation verzichtet werden, da nur eine lokale Suche in den Regionen nötig ist, die durch die a-priori Partikelverteilung gegeben sind. Zum anderen sind auch die Größen der Korrelationsfenster implizit durch die Größen der Zielregionen der jeweiligen Partikel gegeben. Anstatt wie üblich das gesamte Bild mit einer festen Fenstergröße zu korrelieren, wird hier also jedes Partikel individuell mit der für seine Zielregion optimalen Fenstergröße korreliert.

[4] Die Implementierung und das Training der Detektoren geschieht basierend auf [Lienhart und Maydt, 2002; Kruppa u. a., 2003], bereitgestellt von der OpenCV-Bibliothek.

Sei $\hat{\Theta}(\mathbf{x})$ die für den Zustand $\mathbf{x}$ aufgrund der Distanz zur Kamera theoretisch zu erwartende Disparität und $\Theta(\mathcal{A}(\mathbf{x}))$ die für die Region $\mathcal{A}(\mathbf{x})$ tatsächlich ermittelte Disparität. Dann ist die Beobachtungswahrscheinlichkeit für das Stereokorrelationsmerkmal gegeben durch

$$p(\mathcal{S}(\mathbf{z})|\mathcal{A}(\mathbf{x})) = \left(1 + |\Theta(\mathcal{A}(\mathbf{x})) - \hat{\Theta}(\mathbf{x})|^{\kappa}\right)^{-1}, \tag{3.25}$$

wobei κ ein Parameter ist, mit dem die Volatilität des Merkmals eingestellt werden kann.

Die Komplexität der lokalen Suche nach Korrelation $\Theta(\mathcal{A}(\mathbf{x}))$ ist skalierungsinvariant, da sie effizient mithilfe von Integralbildern erfolgen kann, wie dies auch schon für konventionelle Disparitätenkarten von [Veksler, 2003] vorgeschlagen wurde.

Insgesamt werden drei Stereokorrelationsmerkmale eingesetzt: S-H für die Kopfregion, S-T für den Torso und S-L für die Beinregion.

3.4 Personentracking mit dem DI^2-Algorithmus

Nachdem in den vorangegangenen Abschnitten der DI^2-Algorithmus zur dynamischen Merkmalsfusion entwickelt sowie eine Gruppe von Merkmalen zum Personentracking beschrieben wurden, sollen diese nun zusammengefügt werden, um damit eine oder mehrere Personen zu tracken.

Für jede Zielperson wird dabei ein dedizierter Partikelfilter instanziiert. In den folgenden Abschnitten werden Zustandsraum, dynamisches Modell und Beobachtungsmodell dieser einzelnen Tracker erläutert. Anschließend wird auf das Problem der Kollisionsvermeidung bzw. der gegenseitigen Verdeckung eingegangen, das bei der Implementierung mit einzelnen Trackern kritisch ist. Zuletzt werden der Mechanismus zur automatischen Initialisierung und Terminierung von Tracks sowie die automatische Aktualisierung der Merkmalsmodelle beschrieben.

3.4.1 Zustandsraum und dynamisches Modell

Der Zustandsraum des Partikelfilters besteht aus dem Ort und der Geschwindigkeit der Person, bezogen auf den Kopfmittelpunkt:

$$\mathbf{x} = \begin{pmatrix} x & y & z & v_x & v_y & v_z \end{pmatrix}^T \tag{3.26}$$

Das dynamische Modell $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, auch bekannt als Zustandsübergangsmodell oder Systemdynamik, muss beim Partikelfilter – ebenso wie das Beobachtungsmodell – nicht analytisch formuliert, sondern nur punktweise ausgewertet werden. Um einen Zustandsvektor zu propagieren, wird er zunächst mit einer stochastischen Komponente ν additiv verrauscht und anschließend mit einer deterministischen Zustandsübergangsmatrix $\mathbf{A}$ multipliziert:

$$\mathbf{x}' = \mathbf{A} \cdot (\mathbf{x} + \nu) \tag{3.27}$$

Das Systemrauschen ν entsteht durch komponentenweises Ziehen von Werten aus einer Normalverteilung $\eta \leftarrow \mathcal{N}(\mu = 0, \sigma = \hat{a})$, deren Standardabweichung $\hat{a}$ so gewählt ist, dass sie der typischen Beschleunigung menschlicher Bewegungen entspricht:

$$\nu = \begin{pmatrix} 0 & 0 & 0 & \eta & \eta & \eta \end{pmatrix}^T \tag{3.28}$$

Die Zustandsübergangsmatrix $\mathbf{A}$ modelliert eine gleichmäßig beschleunigte Bewegung für die Zeit Δt zwischen dem vorangegangenen und dem aktuellen Frame:

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.29}$$

Das dynamische Modell kann und sollte dazu benutzt werden, zusätzliche Einschränkungen (*constraints*) im Zustandsraum durchzusetzen[5]. Dazu gehören:

- Einhaltung der Grenzen eines festgelegten Trackingbereiches, z. B. die Abmessungen des Raums, in dem der Tracker eingesetzt wird. Partikel dürfen den Trackingbereich nicht verlassen, Stimuli außerhalb des Bereiches werden effektiv ignoriert.
- Beschränkung der y-Koordinate auf die durchschnittliche Körperhöhe von Menschen
- Beschränkung der Geschwindigkeit v_{xyz} auf Werte, die für Menschen typisch sind

[5]Denkbar wäre auch, die Einschränkungen erst im Beobachtungsmodell durchzusetzen und Partikel, die ihnen zuwiderlaufen, durch eine niedrige Bewertung zu eliminieren. Die Einschränkungen gleich im dynamischen Modell durchzusetzen, ist jedoch effektiver, da so die begrenzte Anzahl Partikel besser genutzt wird.

3.4.2 Beobachtungsmodell für einen Stereokamerakopf

Das Beobachtungsmodell $p(\mathbf{z}|\mathbf{x})$ erfüllt im Partikelfilter den Zweck, einen Zustand $\mathbf{x}$ anhand der Beobachtung $\mathbf{z}$ zu bewerten. Wie in Abschnitt 3.1.3 ausgeführt, ist das Beobachtungsmodell als gewichtete Summe von einzelnen Merkmalen formuliert:

$$p(\mathbf{z}|\mathbf{x}) = \sum_{c=1..C} r_c \; p(f_c(\mathbf{z})|\mathbf{x}) \tag{3.30}$$

Unter den in Abschnitt 3.3 definierten Merkmalen nehmen die Stereokorrelationsmerkmale eine Sonderstellung ein: kann für eine Hypothese keine unterstützende Korrelation im Bild gefunden werden, dann befindet sich am entsprechenden Ort mit hoher Sicherheit kein texturiertes Objekt und somit auch keine Zielperson. Bei den Stereokorrelationsmerkmalen handelt es sich demzufolge um *notwendige* Merkmale $\mathcal{C}_N \subset \{1, ..., C\}$. Um sich diesen Umstand zunutze zu machen, werden die Stereokorrelationsmerkmale in der ersten Stufe des Layered Sampling (vgl. Abschnitt 3.1.2) eingesetzt, d. h. alle Partikel müssen zunächst die Stereokorrelationsmerkmale passieren.

Alle anderen Merkmale werden analog dazu als *optionale* Merkmale $\mathcal{C}_H \subset \{1, ..., C\}$ bezeichnet, da ihre Unterstützung zwar zielführend ist, aber umgekehrt ihr Ausbleiben nicht zwangsläufig zum Verwerfen einer Hypothese führen muss. Sie sind in der zweiten Stufe des Layered Sampling versammelt und bewerten dort eine Partikelmenge, die bereits durch die erste Stufe vorgefiltert und durch Resampling optimiert wurde.

Das Vertrauen in die Stereokorrelationsmerkmale gilt aber nur für die Summe der drei; jedes einzelne davon (Kopf, Torso, Beine) kann durchaus z. B. durch Verdeckung oder durch Verlassen des Sichtfeldes bei Annäherung an die Kamera ausfallen. Im gewählten Ansatz werden daher die drei Stereokorrelationsmerkmale der ersten Stufe im Layered Sampling ihrerseits noch einmal mittels DI^2 dynamisch gewichtet. Eine Hypothese kann infolge dessen nur dann in die zweite Stufe zu den optionalen Merkmalen gelangen, wenn sie von mindestens einem Stereokorrelationsmerkmal unterstützt wird.

Abbildung 3.10 stellt den Ablauf zur Bewertung der Partikelmenge im Überblick dar.

3.4.3 Beobachtungsmodell für Mehrkameraumgebungen

Mehrkameraumgebungen im Sinne dieser Arbeit bestehen aus zwei oder mehr monokularen Kameras, deren Sichtfelder sich teilweise überlappen. Im Gegensatz zum Tracking mit dem Stereokamerakopf eines Roboters sind die Kameras

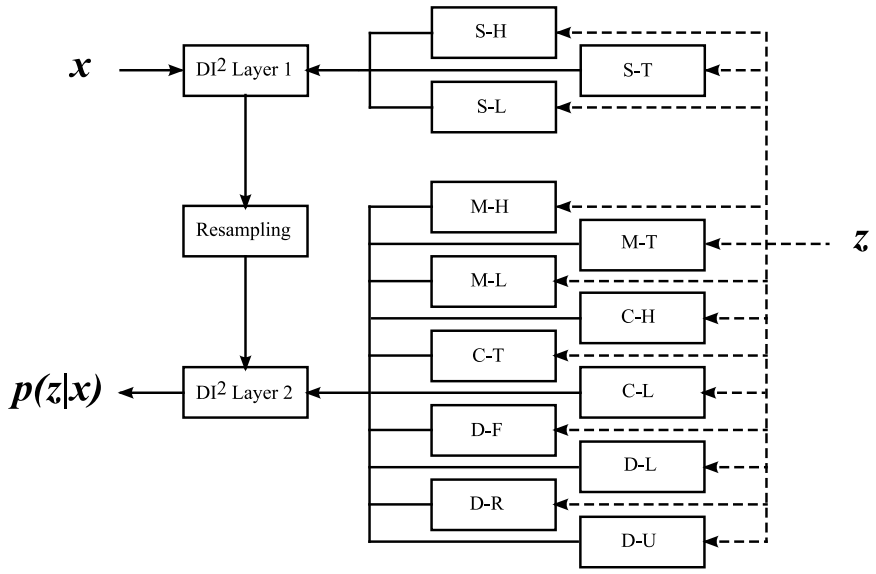


Abbildung 3.10: Das Beobachtungsmodell wird durch Layered Sampling in zwei Stufen unterteilt: In der ersten Stufe werden die Stereokorrelationsmerkmale mittels DI^2 dynamisch kombiniert, dann erfolgt ein Resampling. Die neue Partikelmenge wird in der zweiten Stufe durch die restlichen Merkmale bewertet, deren Ausgaben ebenfalls per DI^2 kombiniert werden.

hier fest angebracht und überblicken die Szene typischerweise aus einem leicht erhöhten Blickwinkel. Das Bewegungsmerkmal ist in diesem Szenario aussagekräftiger, da sich die Kameras selbst nicht bewegen. Das Stereokorrelationsmerkmal ist in Ermangelung einer Stereokamera nicht einsetzbar – dafür kann aber durch Kombination der Kameras im Überlappungsbereich indirekt räumliche Information gewonnen werden.

Bei der Formulierung eines Beobachtungsmodells für Mehrkameraumgebungen wird deutlich, dass zusätzlich zum Wettbewerb der Merkmalstypen und Körperregionen als dritte Dimension auch noch ein Wettbewerb zwischen den Kameras hinzukommt: Zur Bewertung einer Hypothese sind nämlich aufgrund von Verdeckungen, des begrenzten Blickwinkels oder der Entfernung nicht alle Kameras gleichermaßen geeignet.

Es liegt daher nahe, DI^2 auch auf Kameraebene einzusetzen, und jede Kamera als ein eigenes Merkmal zu betrachten, das seinerseits aus den bekannten Merkmalen Farbe, Bewegung und Detektoren zusammengesetzt ist. So soll die Fähigkeit von DI^2 ausgenutzt werden, über automatische Selektion der besten Merkmale implizit auch eine Selektion der geeignetsten Kameraansichten vorzunehmen.

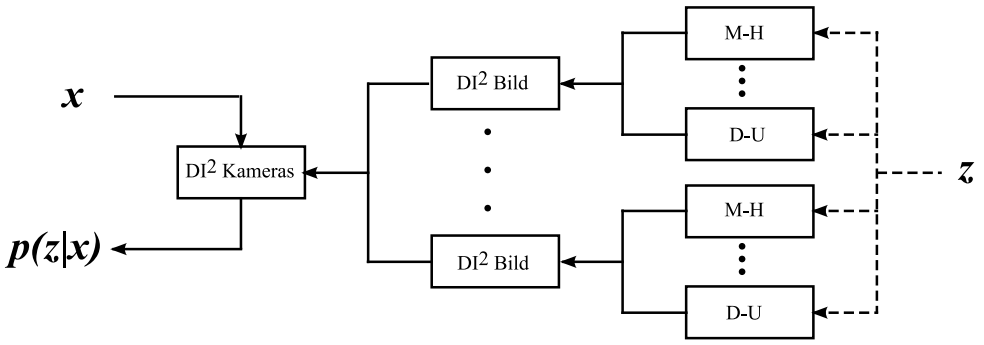


Abbildung 3.11: In Mehrkameraumgebungen werden zunächst die Merkmale in den einzelnen Kameraansichten jeweils per DI^2 kombiniert. Auf einer zweiten Ebene werden die Kameras selbst als Merkmale betrachtet und ihrerseits mittels DI^2 kombiniert. Das Ergebnis ist ein Beobachtungsmodell, das selbstständig die Gewichtung von Merkmalstypen, Körperregionen und Kameraansichten vornehmen kann.

Abbildung 3.11 skizziert das Beobachtungsmodell für Mehrkameraumgebungen. Die dynamische Merkmalsfusion wird hierbei hierarchisch auf Bildebene und auf Kameraebene eingesetzt.

3.4.4 Kollisionsvermeidung

Beim Tracking mehrerer Ziele tritt das Problem auf, dass sich zwei oder mehr Tracks auf ein und dasselbe Ziel konzentrieren. Das kann zur Folge haben, dass sich beide Tracks auf annähernd denselben Koordinaten befinden – und auch dort bleiben. Dieses Problem würde bei einer Formulierung mit einem gemeinsamen Zustandsraum für alle Ziele nicht entstehen, denn hier könnten solche Situationen explizit auf der so genannten Konfigurationsebene behandelt werden. Ein gemeinsamer Zustandsraum hat jedoch eine hohe Anzahl Dimensionen, die bei einem partikelfilterbasierten Trackingsystem zu einer exponentiell ansteigenden Komplexität führen, was für ein Echtzeitsystem nicht mehr tragbar wäre.

Daher wird hier zur Kollisionsvermeidung eine andere Strategie verfolgt, die auf gegenseitigem Ausschluss von Tracks im Bildbereich basiert. Dazu wird für jeden Track T eine Verdeckungskarte erstellt, in die die Zielregionen all jener Tracks eingezeichnet werden, die sich näher an der Kamera befinden, und damit T möglicherweise verdecken könnten (siehe Abbildung 3.12).

Bei der Bewertung eines Partikels von Track T durch das Beobachtungsmodell wird von allen Merkmalen die Verdeckungskarte im Zielbereich von T ausgewertet. Überschreitet die Verdeckung der Zielregion einen gewissen Schwellenwert,

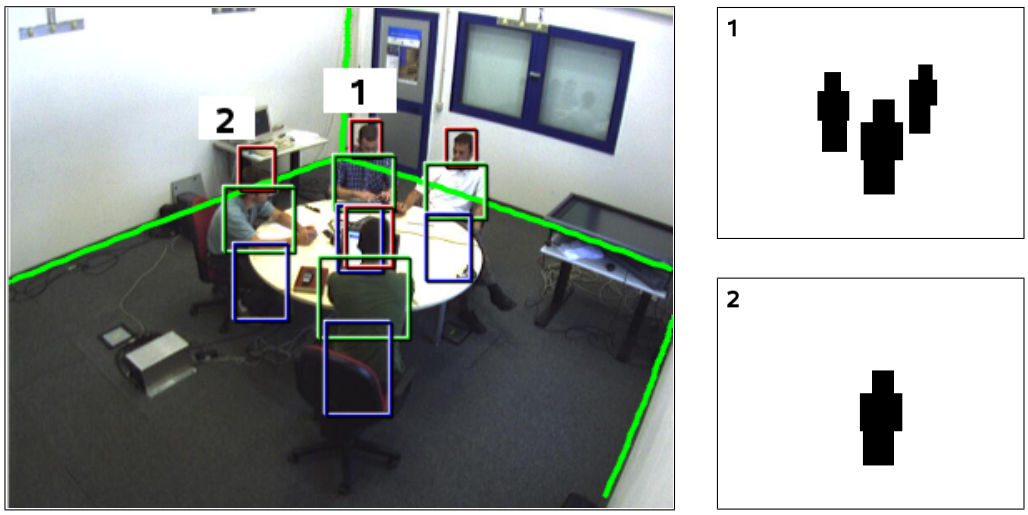


Abbildung 3.12: In der Verdeckungskarte eines Tracks sind die Zielregionen all jener Tracks ausmaskiert, die sich näher an der Kamera befinden.

wird dem Partikel ein Wert nahe 0 zugewiesen. So wird effektiv verhindert, dass Tracks im Hintergrund ihre Unterstützung aus Bereichen ziehen, die bereits von Tracks im Vordergrund belegt sind. Insbesondere die Situation, dass sich mehrere Tracks an einer Position überlappen, kann daher nicht mehr auftreten.

Die Auswertung der Verdeckungskarte kann – wie bei den Merkmalen selbst – effizient mithilfe des Integralbilds erfolgen. Werden im Beobachtungsmodell mehrere Samplingstufen verwendet, dann genügt es, die Verdeckung auf der ersten Ebene zu behandeln. Im Falle des Trackings mit der Stereokamera geschieht dies also auf Ebene der Stereokorrelationsmerkmale.

3.4.5 Automatische Initialisierung und Terminierung

Die Frage nach dem Start eines neuen Tracks bzw. der Beendigung eines laufenden Tracks, der sein Ziel verloren hat, ist wichtig und kann je nach Situation sogar schwieriger sein als das eigentliche Trackingproblem. Um die Existenzwahrscheinlichkeit eines Zielobjektes quantifizieren zu können, wird ein Gütemaß für die Hypothese eines Tracks benötigt. Man bekommt es, indem man die Ausgaben der einzelnen Merkmale an der Stelle der Hypothese $\hat{\mathbf{x}}$ betrachtet. Um alle Merkmale entsprechend ihres tatsächlichen Beitrags zum Tracking zu berücksichtigen, werden sie mit ihren aktuellen DI^2-Gewichten kombiniert:

Für eine Stereokamera: Hier müssen die Merkmale aus der ersten Stufe mit den Merkmalen aus der zweiten Stufe multipliziert werden, um den Vorgang des Layered Sampling abzubilden.

$$Q(\hat{\mathbf{x}}) = \left(\sum_{c \in C_N} r_c p_c(\mathbf{z}|\hat{\mathbf{x}}) \right) \cdot \left(\sum_{c \in C_H} r_c p_c(\mathbf{z}|\hat{\mathbf{x}}) \right) \tag{3.31}$$

Für eine Mehrkameraumgebung: Hier werden die Merkmale hierarchisch erst auf Bildebene C_H und dann auf Kameraebene C_K mit ihren Gewichten kombiniert.

$$Q(\hat{\mathbf{x}}) = \sum_{k \in C_K} r_k \left(\sum_{c \in C_H} r_c p_c(\mathbf{z}|\hat{\mathbf{x}}) \right) \tag{3.32}$$

Tracker, deren Qualitätsmaß $Q(\hat{\mathbf{x}})$ für eine bestimmte Zeit Γ einen Schwellenwert Θ^- unterschreitet, werden beendet.

Um umgekehrt einen neuen Track starten zu können, sobald ein neues Zielobjekt ins Sichtfeld gerät, ist eine permanente Suche im Bild notwendig. Da eine flächendeckende Suche aufgrund ihres Aufwands nicht in Betracht kommt, wird die gleiche Strategie wie beim eigentlichen Tracking angewendet: Ein dedizierter Partikeltracker, der so genannte *Aufmerksamkeitstracker*, durchsucht ständig den Zustandsraum abseits der existierenden Tracks. Um Kollisionen bzw. Überlappung mit existierenden Tracks zu vermeiden, wird auch für den Aufmerksamkeitstracker die Verdeckungskarte aus dem vorangegangenen Abschnitt aufgestellt.

Da der Aufmerksamkeitstracker eine möglichst breite Abdeckung des Zustandsraums leisten soll, werden 50% seiner Partikel nicht regulär mit dem Zustandsübergangsmodell propagiert, sondern mit uniformer Wahrscheinlichkeit aus dem Zustandsraum gezogen. Die Merkmalsmodelle des Aufmerksamkeitstrackers sollen sich keinem Ziel anpassen und werden daher nicht aktualisiert.

Der Aufmerksamkeitstracker verfolgt somit Stimuli, die unterhalb der eigentlichen Trackingschwelle liegen. Nach jedem Frame wird seine Partikelverteilung mit dem k-Mittelwerte-Verfahren geclustert. Sobald einer der Clustermittelpunkte einen Schwellenwert Θ^+ überschreitet, wird eine neuer, regulärer Track an dieser Position gestartet. Seine Partikelmenge wird normalverteilt um den Clustermittelpunkt initialisiert.

Die beiden Schwellen Θ^+ zum Erzeugen neuer Tracks bzw. Θ^- zum Beenden aktiver Tracks werden als Hystereseintervall um eine allgemeine Trackingschwelle Θ gewählt:

$$\Theta^- = \Theta - \zeta \quad < \quad \Theta + \zeta = \Theta^+ \tag{3.33}$$

Der Hysteresebereich ζ wird dabei so hoch eingestellt, dass ein in schneller Folge abwechselndes Erzeugen und Terminieren von Tracks im Grenzbereich um Θ unterbunden wird.

3.4.6 Automatische Modellanpassung

Bestimmte Merkmale, wie hier das Farbmerkmal, besitzen interne Modelle, die zur Laufzeit an das Ziel angepasst werden müssen, um dessen konkrete Eigenheiten abzubilden. Eine Möglichkeit besteht darin, dies bei der Initialisierung eines Tracks zu tun, und die Modelle danach nicht mehr zu verändern. Dieses Vorgehen ist jedoch in zweierlei Hinsicht problematisch: Zum einen wird die Initialisierung dadurch zu einem besonders kritischen Vorgang; wird nämlich nicht exakt die Zielregion initialisiert, bleiben die Modelle bis zum Lebensende des Tracks fehlerhaft. Zum anderen kann ein statisches Modell leicht versagen, wenn sich z. B. die Beleuchtungsbedingungen ändern oder sich die Person umdreht. Eine fortlaufende Anpassung der Merkmalsmodelle an das aktuelle Abbild des Ziels ist daher wünschenswert.

In [Triesch und Malsburg, 2001] wird diese Anpassung als kontinuierlicher Prozess beschrieben, der die Modelle mit einer festen Zeitkonstante τ_c aktualisiert, so dass

$$\tau_c \dot{P}_c = \hat{P}_c - P_c, \tag{3.34}$$

wobei P_c das interne Modell des Merkmals c ist, und $\hat{P}_c$ einen neuen Modellprototypen darstellt, der für die Bildregion der gemeinsamen Hypothese $\hat{\mathbf{x}}$ neu aufgestellt wird.

Eine mögliche Komplikation bei der automatischen Modellanpassung liegt darin begründet, dass ein Merkmal nach der Aktualisierung seines Modells die aktuelle Beobachtung nicht zwangsläufig besser als zuvor unterstützt. Zwar führt die Aktualisierung immer zu einem mindestens ebenso hohen Wert von $p_c(f_c(\mathbf{z})|\hat{\mathbf{x}})$ für die Prototypenregion wie zuvor, allerdings kann der Wert auch für andere Regionen als $\hat{\mathbf{x}}$ stark ansteigen. Wenn dies der Fall ist, können sich – scheinbar paradoxerweise – die diskriminativen Fähigkeiten eines Merkmals durch seine Aktualisierung de facto verringern.

Mit dem Qualitätsmaß der DI^2 aus Abschnitt 3.2.1 lässt sich dieser Fall allerdings erkennen und die Degeneration eines Merkmals durch eine schlechte Aktualisierung verhindern. Dazu muss die Aktualisierung mit einem nachfolgenden Qualitätstest überprüft werden:

1. Berechne q_c mit dem alten Modell P_c nach Gleichung 3.13
2. Berechne q'_c mit dem neuen Modell $\hat{P}_c$
3. Führe die Aktualisierung (Gleichung 3.34) nur durch, wenn $q'_c > q_c$

Die Bestimmung von q'_c erfordert eine temporäre Neuberechnung von $\pi_c^{(n)}$. Es genügt für diesen Test, die Berechnung nur mit einem Bruchteil der eigentlichen Partikelanzahl N durchzuführen, um den Rechenaufwand zu verringern.

3.5 Evaluation

Um die Funktionsfähigkeit des Trackingverfahrens auszuwerten, wurden verschiedene Experimente in unterschiedlichen Umgebungen durchgeführt. Gemessen wurde dabei zum einen die absolute Qualität des Trackings, und zum anderen speziell die Verbesserung, die durch die dynamische Fusion verallgemeinerter Merkmale mithilfe des DI^2-Algorithmus (Abschnitt 3.2.4) erreicht wird.

Die erste Gruppe von Experimenten wurde mit einer Stereokamera durchgeführt und demonstriert die Anwendbarkeit des Algorithmus im anvisierten Roboterszenario. Gezeigt wird hier insbesondere die Fähigkeit des Algorithmus, Personen sowohl in der Ferne zu lokalisieren, wenn ihre Silhouette klein aber komplett sichtbar im Bild eingebettet erscheint, als auch im Nahbereich, wenn nur ein kleiner Teil des Körpers das Bild bestimmt.

Aufgrund seiner allgemeinen Formulierung ist der DI^2-Algorithmus nicht allein auf das Tracking mit einer Stereokamera aus der Egoperspektive eines Roboters beschränkt. Um die Vielseitigkeit des Algorithmus zu demonstrieren, wurde daher eine zweite Gruppe von Experimenten zum Mehrpersonentracking in einer Multikameraumgebung durchgeführt, bei der der Algorithmus zusätzlich zum dynamischen Wettbewerb zwischen Merkmalstypen und Körperregionen auch einen Wettbewerb zwischen verschiedenen Kameraansichten unterstützt. Ein zweiter Grund für die Durchführung von Experimenten in der Mehrkameraumgebung war, dass – im Gegensatz zum Roboterszenario – hier ein großer Korpus von international genutzten Videodaten zur Verfügung stand.

3.5.1 Experimente mit der Roboterkamera

Der Algorithmus zum Personentracking mit dynamischer Merkmalsfusion wurde auf 11 Videosequenzen getestet, von denen einige auch Kamerabewegung beinhalten. Die Kopfregion wurde in 3 der 15 Bilder pro Sekunde manuell markiert und dient als Vergleichswert. Insgesamt wurden so 2312 Bilder annotiert.

Die 3D-Hypothese des Trackers wurde als Quader mit den durchschnittlichen Abmessungen eines Kopfes ins Bild projiziert und dort mit der manuell annotierten Kopfregion verglichen. Liefert der Tracker keine Ausgabe für ein Bild, das eine manuelle Annotierung enthält, handelt es sich um einen Auslassungsfehler (*miss*). Gibt es umgekehrt keine manuelle Annotierung und der Tracker liefert trotzdem eine Hypothese, handelt es sich um eine Fehlerkennung (*false positive*). Liegen sowohl Annotierung als auch Hypothese vor, aber es gibt keine Überlappung zwischen den beiden Regionen, wird das Bild zugleich als Auslassungsfehler und als Fehlerkennung gewertet.

	misses	false pos.
Uniforme Gewichte (Baseline)	10.2%	8.1%
DI^2 nach Gleichung 3.13	**4.6%**	**4.6%**
Dynamische Gewichte nach [Shen u. a., 2003]	11.1%	8.8%

Tabelle 3.1: Trackingergebnisse auf den Evaluationsdaten.

Insgesamt legte der Algorithmus in den Experimenten eine solide Leistung an den Tag. Kritische Situationen, in denen Tracks – wenn auch selten – verloren gingen, waren Zeitspannen, in denen der Proband nahezu bewegungslos entweder in größerer Entfernung zur Kamera oder in einer abgewandten Körperhaltung stand. In diesen Situationen erhielten die Detektor- sowie die Bewegungsmerkmale keine Unterstützung, und der Tracker musste sich allein auf die automatisch initialisierten Farbmerkmale verlassen, die nicht immer signifikant waren. Ein weiteres Problem waren Phantomtracks, die durch nicht-menschliche Bewegung im Hintergrund oder durch Fehldetektionen ausgelöst wurden. Diese wurden in seltenen Fällen durch die Farbmerkmale am Leben erhalten, die sich an die fälschlicherweise erkannten Körperregionen angepasst hatten. Im Regelfall konnte dies jedoch durch den Kontrollmechanismus aus Abschnitt 3.4.6 verhindert werden.

In Tabelle 3.1 sind die Ergebnisse der Evaluation dargestellt. Der Algorithmus wurde zum einen mit einem Baseline-System verglichen, in dem die Gewichtung der Merkmale statisch war. Zum anderen wurde er mit einer Variante verglichen, bei der die Gewichtung der Merkmale dynamisch mit dem Qualitätsmaß aus [Shen u. a., 2003] erfolgte. Der in dieser Arbeit vorgestellte Algorithmus übertrifft die beiden Vergleichssysteme deutlich im Hinblick sowohl auf die Fehlerkennungen als auch auf die Auslassungsfehler. Abbildung 3.13 zeigt die Entwicklung der Merkmalsgewichte (reliabilities) im Verlauf eines Abschnitts aus einer der Testsequenzen.

Bei der Durchführung der Experimente wurden folgende Ergänzungen zum Algorithmus aus Abschnitt 3.4.1 vorgenommen:

- Das Farbmerkmal für die Kopfregion (C-H) konvergiert erwartungsgemäß zur menschlichen Hautfarbe; sein internes Modell wird daher von allen Trackern gemeinsam verwendet. Für das Torso-Detektormerkmal wurde eine veränderte Region verwendet, die Kopf und Oberkörper umfasst.
- Quader, deren Projektion zu mindestens 80% außerhalb des Bildes lag, wurden mit einem kleinen positiven Wert von $\epsilon = 0,001$ gewertet. Diese Bewertung mit $\epsilon > 0$ ist wichtig, da der Tracker anderenfalls Zielen, die den Sichtbereich verlassen, nicht folgen könnte.

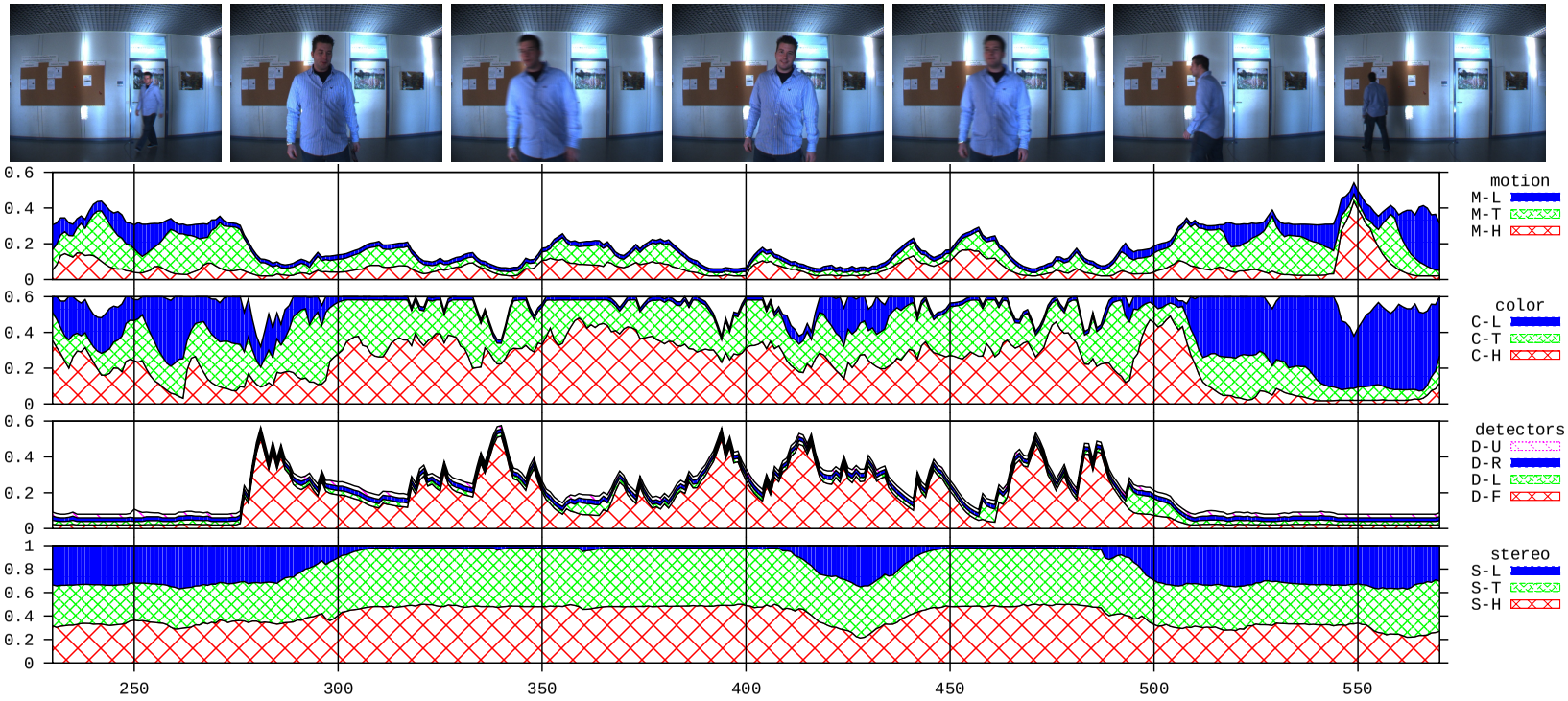


Abbildung 3.13: Entwicklung der Gewichte (reliabilities) im Verlauf einer Testsequenz. Die drei Stereomerkmale bilden die erste Stufe des Algorithmus, ihre Werte addieren sich zu 1. Die verbleibenden zehn Merkmale werden in der zweiten Stufe ausgewertet und summieren sich ebenfalls zu 1. Am Anfang der Sequenz nähert sich der Proband der Kamera. Während er geht (Bild 250), tragen die Bewegungsmerkmale für Beine und Torso (M-L,M-T) maßgeblich zum Tracking bei. Ab etwa Bild 300 verschwinden die Beine aus dem Sichtfeld der Kamera, was den automatischen Abfall der Gewichte aller Beinmerkmale (M-L, C-L, S-L) zur Folge hat. Während der Proband vor der Kamera steht (Bild 300-500), dominieren das Detektormerkmal für frontale Gesichter D-F sowie das Farbmerkmal für die Kopfregion C-H das Tracking. Der Einfluss von C-H fällt sofort wieder ab, wenn sich der Proband um Bild 520 umdreht und vor eine hölzerne Pinnwand geht, deren Farbe ähnlich der menschlichen Hautfarbe ist.

Anz. Partikel pro Tracker	$n = 150$
Schwellenwert Track / Timeout	$\Theta = 0.25$, $\Gamma = 2s$
Zeitkonstante DI Qualität	$\nu = 3$
Zeitkonstante DI reliability	$\tau = 4$
Zeitkonstante Farbaktualisierung	$\tau_c = 100$
Faktoren zur Merkmalseinstellung	$\lambda = 4, \kappa = 4, \omega = 10$

Tabelle 3.2: Wahl der Parameter bei der Durchführung der Experimente.

- Um die Dominanz eines einzelnen Merkmales zu verhindern, wurde der Wertebereich für die reliabilities beschränkt: $0.03 \leq r_c \leq 0.6$. Wie sich herausstellte, traten Situationen, in denen diese Schranken tatsächlich erreicht wurden, äußerst selten auf. Das steht im Einklang mit den Beobachtungen von [Triesch und Malsburg, 2001], die diesen seltenen Fall als „Despotismus“ bezeichnen.

Die Laufzeit des Algorithmus lag bei ca. 30 ms pro Bild bei einer leeren Szene und weiteren 10 ms für jedes verfolgte Ziel. Diese Werte beziehen sich auf eine Bildgröße von 320×240 Pixel und einen 2,4 GHz Pentium Prozessor. Die wichtigsten Parameter sind in Tabelle 3.2 aufgeführt.

3.5.2 Experimente in Mehrkameraumgebungen

Für das Roboterszenario wurde eigens – wie im vorangegangenen Abschnitt beschrieben – eine Menge von Testdaten gesammelt, annotiert und ausgewertet. Eine detaillierte quantitative Auswertung der Leistung des DI^2-Algorithmus erfordert allerdings eine deutlich größere Menge an annotierten Testdaten, die speziell für das Roboterszenario nicht zu Verfügung stand. Anders ist die Situation bei Mehrkameraumgebungen; hier steht eine große Menge von öffentlich verfügbaren Daten zur Verfügung. Dazu gehört z. B. der Korpus des CLEAR'07-Workshops [Stiefelhagen u. a., 2007], der im Folgenden Verwendung findet. Für den CLEAR'07-Korpus wurden in unterschiedlichen Räumen an verschiedenen Standorten Besprechungen aufgenommen und annotiert, bei denen mehrere Personen um einen Tisch sitzen bzw. sich frei im Raum bewegen. In den jeweiligen Besprechungsräumen sind mehrere monokulare Kameras fest an Decke und Wänden montiert, so dass ihr gemeinsames Sichtfeld den kompletten Raum abdeckt (*smart room*). Abbildung 3.14 zeigt einen Ausschnitt aus einer Sequenz des CLEAR'07-Korpus.

Typischerweise stehen aus den einzelnen Besprechungsräumen – wie abgebildet – Bilder von vier Schrägsichtkameras und einer Weitwinkel-Deckenkamera zur Verfügung. Die Aufnahmen wurden mit wechselnden Bildwiederholraten und Auflösungen gemacht, typische Werte liegen bei $15-25 fps$ sowie $640 \times 480 pixel$.

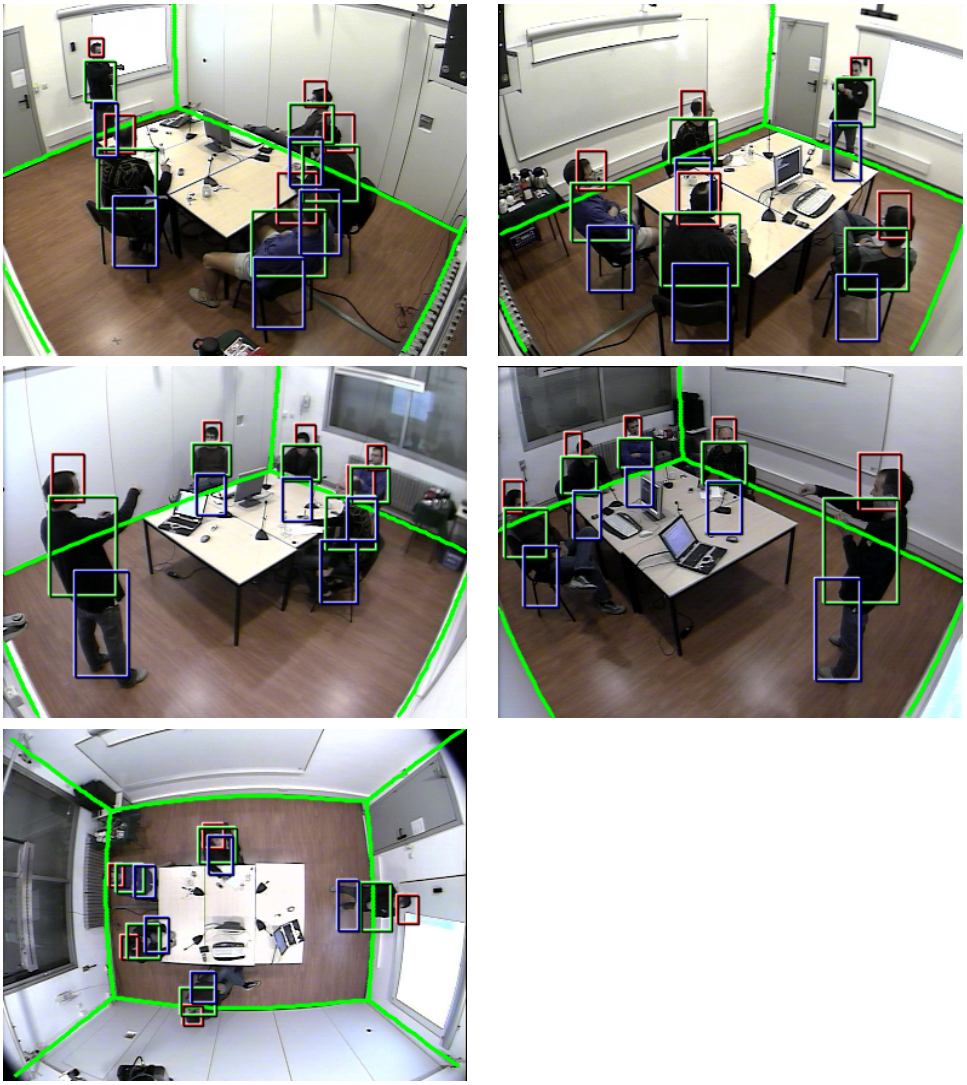


Abbildung 3.14: Beispielbilder aus dem CLEAR'07-Korpus. Tracks und Raumbegrenzungen sind dem Kamerabild überlagert.

Zusätzlich zu den Videosequenzen sind die intrinsischen und extrinsischen Kalibrierungen der Kameras gegeben sowie Aufnahmen der Besprechungsräume in verlassenem Zustand. Weiterhin existieren manuelle Annotationen unter anderem der Kopfmittelpunkte aller in den Bildern sichtbaren Personen – und zwar zum einen als Bildkoordinaten, und zum anderen als daraus triangulierte 3D-Koordinaten in einem globalen Raumkoordinatensystem.

In der Regel sitzen die Personen während der Aufnahme um einen Tisch, eine weitere Person steht vor einer Projektionsfläche und hält einen Vortrag. Gelegentlich stehen Personen auf, setzen sich, gehen umher, betreten oder verlassen den Raum. Für die folgenden Experimente wurden nur diejenigen Sequenzen herangezogen, in denen die Zahl der Personen im Raum konstant bleibt. Der Grund dafür ist, dass Fehler beim Erzeugen bzw. Löschen von Tracks so drastische Auswirkungen auf das Endergebnis hätten, dass die Effekte, die hier eigentlich gemessen werden sollen, überdeckt werden können. Auf diese Weise wurden aus dem CLEAR'07-Korpus 17 Segmente ausgewählt.[6]

Im Folgenden wird der DI^2-Algorithmus auf diesen Segmenten ausgewertet. Dadurch soll gezeigt werden, dass der dynamische Ansatz tatsächlich relevante Verbesserungen bringt, und dass er über das konkrete Roboterszenario hinaus auch für allgemeine Trackingaufgaben nutzbringend einsetzbar ist. Zur Auswertung wurden die *multi-object tracking accuracy/precision* Metriken aus [Bernardin u. a., 2006] verwendet:

MOTA: Die MOTA gibt den Prozentsatz korrekt getrackter Frames an. Dabei werden von zunächst 100% die Prozentsätze von Frames mit Auslassungen (*misses*), Hinzufügungen (*false positives*) sowie Verwechslungen (*mismatches*) von Personen abgezogen.[7] Zur Berechnung der MOTA ist die Angabe eines Maximalabstandes erforderlich, bis zu dem ein Track noch als korrekt zugeordnet gilt. Im CLEAR'07-Workshop wurde dieser Schwellenwert mit $0,5m$ festgelegt.

MOTP: Die MOTP gibt die Genauigkeit der Lokalisierung an, d. h. die durchschnittliche Distanz der Hypothese zur Annotation. Der Berechnung zugrunde gelegt werden alle korrekt zugeordneten Tracks, was dazu führt, dass der Wert der MOTP nach oben durch den Schwellenwert von hier $0,5m$ (und nach unten durch $0m$) begrenzt ist.

Die Experimente wurden unter vier unterschiedlichen Bedingungen durchgeführt. Ziel ist es, den Einfluss des DI^2-Algorithmus auf die Testergebnisse sichtbar zu machen:

[6] Die hier gezeigten Ergebnisse sind aufgrund des Verzichts auf automatische Initialisierung/Terminierung und der damit einhergehenden Sequenzauswahl nicht direkt mit den in [Stiefelhagen u. a., 2007] präsentierten Ergebnissen des CLEAR'07-Workshops vergleichbar. Bis auf diesen Unterschied sind Auswertungsmethodik und Metriken jedoch identisch.

[7] Der Wert der MOTA ist nach unten nicht begrenzt, d. h. er kann bei einer hohen Zahl von Fehlern auch Werte < 0 annehmen.

Uniform: Vergleichsgrundlage (Baseline) mit festen uniformen Gewichten (*reliabilities*), d. h. eine dynamische Anpassung findet nicht statt.

DI²: Vollständiger DI²-Algorithmus mit dynamischer Anpassung der Gewichte.

DI²-fix: Eingeschränkter DI²-Algorithmus, bei dem die Gewichtung der Körperteile uniform und fest ist, d. h. die „zweite Dimension“ des Wettbewerbs fehlt.

Shen: Direkter Vergleich zu Bedingung DI², allerdings mit dem Qualitätsmaß nach [Shen u. a., 2003] anstatt nach Gleichung 3.13.

In Tabelle 3.3 sind die Ergebnisse der Experimente detailliert aufgelistet und in Abbildung 3.15 grafisch zusammengefasst. Als wichtigstes Resultat zeigte sich, dass die MOTA des DI²-Algorithmus mit $94,0\%$ gegenüber der Baseline (Uniform) um $5,3$ Prozentpunkte höher ausfiel; dies bedeutet eine Reduktion der Trackingfehler um 47% durch Einsatz der dynamischen Merkmalskombination. Bei der MOTP zeigte sich ein ähnliches Bild: Hier konnte die Genauigkeit von vorher $113mm$ auf nun $65mm$ gesteigert werden, was einer Reduktion des Fehlers von 43% entspricht.

Wurde der dynamische Wettbewerb nur auf die Merkmalstypen beschränkt, wie in Bedingung DI²-fix der Fall, gingen bei der MOTA $4,2$ Prozentpunkte verloren, und auch die MOTP verschlechterte sich – wenn auch weniger stark – um $16mm$. Dadurch zeigte sich, dass beide Aspekte des DI²Algorithmus, d. h. sowohl die dynamische Gewichtung als auch die Ausdehnung des Wettbewerbs auf Zielregionen, Anteile am Erfolg haben.

Als letztes wurde das Qualitätsmaß nach [Shen u. a., 2003] getestet, das sich allerdings bei den vorliegenden Daten als kontraproduktiv erwiesen hat: Sowohl MOTA als auch MOTP blieben deutlich hinter der Baseline zurück. Grund für das schlechte Abschneiden ist vermutlich der in Abschnitt 3.2.1 beschriebene Umstand, der bei Multikameradaten besonders stark zum Tragen kommt: Merkmale aus Kameras, in denen das Ziel nicht sichtbar ist, werden mit einer uniformen Wahrscheinlichkeitsverteilung bewertet. Dies führt aber nach Gleichung 3.12 dazu, dass für das an sich nutzlose Merkmal trotzdem eine hohe *reliability* errechnet wird, wenn das Ziel für eine Weile stillsteht.

Segment	**Uniform**		**DI²**		**DI²-fix**		**Shen**	
	MOTA	MOTP	MOTA	MOTP	MOTA	MOTP	MOTA	MOTP
AIT-20061020B-A	94,3%	116*mm*	97,9%	65*mm*	97,7%	73*mm*	63,2%	224*mm*
AIT-20061020B-B	95,0%	155*mm*	87,9%	137*mm*	93,8%	110*mm*	20,4%	266*mm*
AIT-20061020C-A	95,2%	114*mm*	99,7%	51*mm*	99,7%	66*mm*	55,5%	221*mm*
AIT-20061020D-A	89,9%	104*mm*	80,0%	93*mm*	87,9%	79*mm*	69,2%	208*mm*
AIT-20061020-B	92,9%	74*mm*	99,5%	52*mm*	90,8%	55*mm*	61,6%	185*mm*
ITC-20060922A-B	97,4%	96*mm*	99,8%	35*mm*	99,8%	60*mm*	57,4%	171*mm*
ITC-20060922B-A	99,2%	135*mm*	100,0%	59*mm*	100,0%	64*mm*	63,1%	191*mm*
ITC-20060927-B	99,3%	83*mm*	100,0%	31*mm*	100,0%	47*mm*	71,4%	133*mm*
ITC-20060928-A	94,9%	125*mm*	100,0%	44*mm*	90,0%	79*mm*	44,3%	183*mm*
ITC-20060928-B	100,0%	97*mm*	100,0%	38*mm*	100,0%	62*mm*	86,9%	152*mm*
UKA-20060912-A	97,9%	127*mm*	100,0%	65*mm*	99,9%	78*mm*	32,0%	221*mm*
UKA-20060912-B	98,3%	105*mm*	100,0%	42*mm*	100,0%	78*mm*	34,8%	222*mm*
UKA-20061116-A	23,4%	234*mm*	66,9%	142*mm*	29,1%	169*mm*	−31,8%	235*mm*
UKA-20061120-B	39,7%	199*mm*	61,0%	100*mm*	49,2%	135*mm*	−2,7%	228*mm*
UPC-20060713-A	99,2%	75*mm*	99,7%	53*mm*	100,0%	67*mm*	93,8%	98*mm*
UPC-20060713-B	98,2%	92*mm*	98,6%	83*mm*	95,0%	108*mm*	90,6%	132*mm*
UPC-20060720-A	99,2%	74*mm*	100,0%	55*mm*	98,5%	67*mm*	75,9%	130*mm*
∅	88,7%	113*mm*	**94,0%**	**65mm**	89,8%	81*mm*	51,6%	177*mm*

Tabelle 3.3: Ergebnisse für den CLEAR'07-Korpus im Detail.

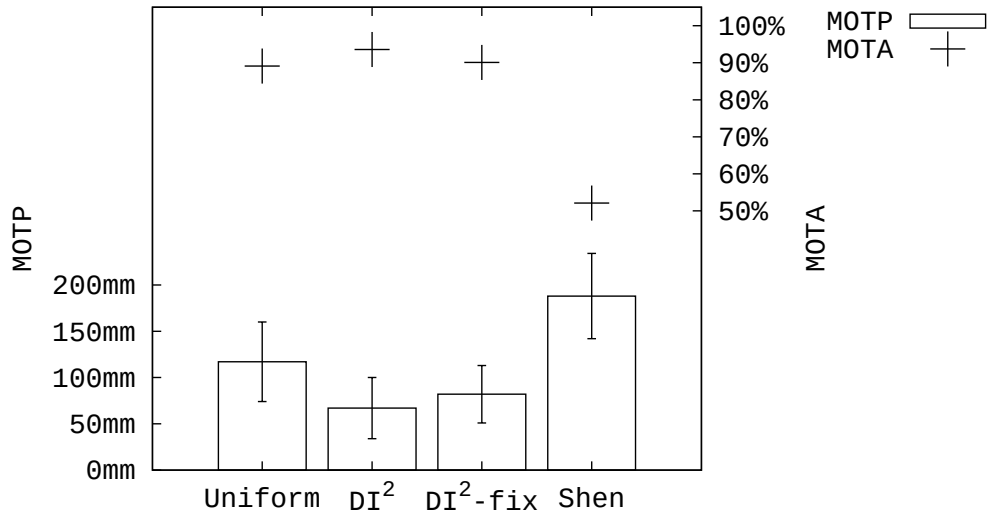


Abbildung 3.15: Ergebnisse für den CLEAR'07-Korpus (zusammengefasst).

4 Zeigegestenerkennung

Im vorangegangenen Kapitel wurde das Personentracking vorgestellt, das es dem Roboter ermöglicht, den Benutzer sowohl im Bild als auch räumlich zu lokalisieren, und seinen Kamerakopf auf ihn auszurichten. Damit sind die Voraussetzungen erfüllt, die für eine weitergehende visuelle Analyse des Menschen erforderlich sind, und die in diesem Kapitel bis zur automatischen Erkennung menschlicher Zeigegesten führen wird.

Zeigegesten werden intuitiv von Menschen verwendet, um die Aufmerksamkeit des Kommunikationspartners auf ein Objekt im Raum zu lenken, oder eine Richtung zu weisen. Gemäß der grundlegenden Untersuchung von Handgesten in [McNeill, 1992] werden sie in der Regel von sprachlichen Äußerungen begleitet. Dort bilden Zeigegesten die Klasse der deiktischen Gesten, die von ikonischen, metaphorischen, und den so genannten „Beat"-Gesten abgrenzt werden kann (siehe Abb. 4.1). Im Rahmen der vorliegenden Arbeit werden Zeigegesten definiert durch das typische Bewegungsmuster der zeigenden Hand. Weitergehende Merkmale wie z. B. die Stellung der Finger werden unter Rücksichtnahme auf die technische Realisierbarkeit hier nicht berücksichtigt.

Voraussetzung zur Erkennung von Zeigegesten ist zunächst einmal das räumliche Lokalisieren der Hände des Benutzers, das im ersten Abschnitt dieses Kapitels beschrieben wird. Sind die 3D-Trajektorien der Hände dann verfügbar, erfolgt die fortlaufende Klassifikation mit trainierten Gestenmodellen, wie im zweiten Abschnitt beschrieben. Unmittelbar nach Detektion einer Zeigegeste wird – wie im dritten Abschnitt ausgeführt – ihre räumliche Richtung geschätzt. Durch die Kopfdrehungsschätzung in Abschnitt vier tut sich zusätzlich zur Handbewegung eine weitere Informationsquelle auf, die zur Gestenerkennung genutzt wird. Und tatsächlich kann, wie die Auswertung in Abschnitt fünf zeigt, die Zeigegestenerkennung durch die Kopfdrehungsschätzung verbessert werden. Da es sich bei Zeigegesten um multimodale Phänomene handelt, die in der Regel im Verbund mit Sprache auftreten, wird in Abschnitt sechs untersucht, ob und wie sich die Erkennung bzw. die Verwendung von Zeigegesten mit dem Wissen um diesen Zusammenhang verbessern lässt. Abbildung 4.2 zeigt die Schritte zur automatischen Zeigegestenerkennung noch einmal im Überblick.

Ikonische Gesten: Bilden durch ihre Erscheinungsform Merkmale, Handlungen, Ereignisse etc. nach.

Metaphorische Gesten: Repräsentieren Konzepte, die keine physische Form haben. Anstelle der fehlenden Form tritt die Metapher. Beispiel: „Die Besprechung dauerte und dauerte“ in Verbindung mit einer kreisenden Bewegung der Hand.

Deiktische Gesten: Lokalisieren reale oder gedachte Gegenstände im Raum vor dem Sprecher.

„Beat“ -Gesten: Kleine Bewegungen wie mit einem Taktstock, deren Form sich nicht mit dem Inhalt der Sprache verändert. Beispiel: „Sie kam als erste, ich meine als zweite dran“ in Verbindung mit einem Auf-/Abwärtszucken der Hand.

Abbildung 4.1: Die vier Typen von Handgesten während der Rede nach McNeill [McNeill, 1992].

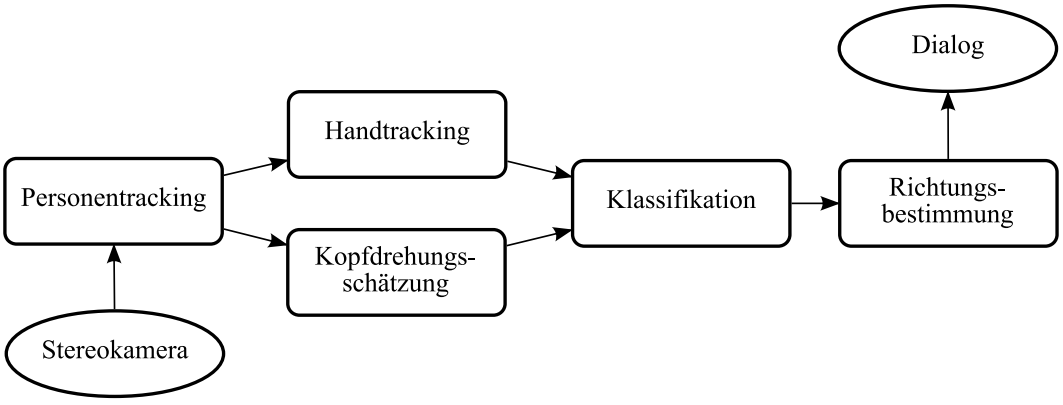


Abbildung 4.2: Verarbeitungsschritte zur automatischen Zeigegestenerkennung.

4.1 Tracking der Hände

Die Zeigegestenerkennung basiert auf einer Klassifikation der Handbewegung; dazu bedarf es einer robusten Lösung zum Tracking der Hände. Um spontane Benutzung zu ermöglichen, soll dabei auf das Tragen von Markern, Datenhandschuhen, o. ä. verzichtet und allein die Stereokamera des Roboters verwendet werden. Der Algorithmus muss dabei so schnell sein, dass er parallel zur Personenlokalisierung aus Kapitel 3 mit den begrenzten Bordmitteln des Roboters ausgeführt werden kann.

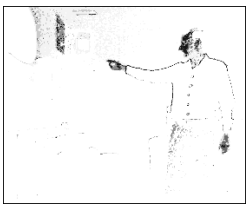
Da Hände eine äußerst variable Erscheinung im Kamerabild aufweisen, ist eine der wichtigsten Entscheidungen beim Lokalisieren von Händen die Wahl der Merkmale. Detektoren wie z. B. die von [Kölsch und Turk, 2004] eingesetzten Haar-Feature-Kaskaden, sind beschränkt auf bestimmte Schlüsselposen der Hand. Auch Kontur-Trackingverfahren, wie bspw. der Einsatz von Splines bei [Isard und Blake, 1998a], erfordern zum einen eine hohe Auflösung und zum anderen eine bestimmte Beobachtungsperspektive. Eine schnelle und zugleich rotationsinvariante Modellierung der Hand kann mithilfe von Farbmodellen erfolgen, wie sie bereits erfolgreich z. B. von [Pentland u. a., 1996; Starner u. a., 1998; Yang u. a., 1997] eingesetzt wurden. Ein inhärenter Nachteil von Farbmodellen ist die geringe Unterscheidungskraft zwischen dem modellierten Ziel und Objekten im Hintergrund, die eine ähnliche Farbe aufweisen.

Das in diesem Abschnitt beschriebene Verfahren kombiniert daher ein Hautfarbmodell für die Hände mit Informationen aus Disparitätenbildern, die aus der Stereobildverarbeitung (siehe hierzu z. B. [Scharstein und Szeliski, 2002]) gewonnen werden. Dadurch wird es möglich, die räumliche Struktur der Szene zu erfassen und Hintergrundobjekte auszuschließen, selbst wenn sie Hautfarbe aufweisen sollten. Die Personenlokalisierung aus Kapitel 3 liefert ergänzend – basierend auf Merkmalen des gesamten Körpers – eine Schätzung der Kopfposition, die vom Hand-Tracking zur weiteren Verkleinerung des Suchraumes genutzt wird.

4.1.1 Lokalisierung der Kandidaten

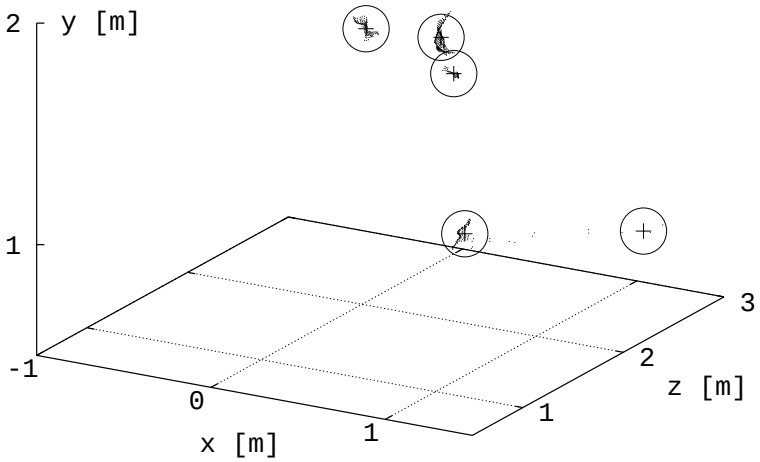
Da für Gesicht und Hände gleichermaßen Hautfarbe zu erwarten ist, wird zum Auffinden der Hände das bereits beim Personentracking eingesetzte und dort fortlaufend aktualisierte Histogramm für den menschlichen Kopfbereich wiederverwendet. Das in Abbildung 4.3.a gezeigte Hautfarbenbild entsteht durch Histogrammrückprojektion, wie bereits in Abschnitt 3.3.2 beschrieben.

Das Hautfarbbild wird mit einer Kombination morphologischer Operatoren gefiltert: Zunächst verbindet eine Dilatation benachbarte Pixel mit dem Ziel, zusammenhängende Regionen zu bilden. Dann werden durch zwei hintereinander



a. Hautfarbenbild

b. Disparitätenbild



c. 3D Hautfarb-Cluster

Abbildung 4.3: Merkmale für das Hand-Tracking: Im Hautfarbenbild repräsentieren dunkle Pixel hohe Wahrscheinlichkeit für Hautfarbe. Die Helligkeit der Pixel in der Darstellung des Disparitätenbildes korreliert mit ihrer Entfernung zur Kamera. Mit ihrer Hilfe werden Hautfarbpixel räumlich geclustert (dargestellt als Kreise).

ausgeführte Erosionsschritte alleinstehende Pixel eliminiert. Abschließend wird durch eine zweite Dilatation die ursprüngliche Größe der Regionen ungefähr wiederhergestellt. Alle Operationen verwenden dasselbe Strukturelement der Größe 3×3. Nach der morphologischen Filterung wird das Hautfarbbild mit einem festen Schwellenwert binarisiert und alle zusammenhängenden Regionen (*blobs*) per Komponentenanalyse (*connected component analysis*) extrahiert.

Für jede der Regionen wird nun mithilfe des Disparitätenbildes der 3D-Mittelpunkt aller zur Region gehörenden Pixel berechnet. Variieren die Pixel einer Region stark im Hinblick auf ihre Entfernung zur Kamera, dann werden sie mithilfe des k-Mittelwerte-Verfahrens aufgeteilt. Dadurch können hautfarbene Objekte auseinander gehalten werden, die sich zwar auf verschiedenen Entfernungsebenen befinden, deren Abbild jedoch zu einer Region verschmilzt. Jeder der so entstandenen Mittelwerte stellt nun einen Kandidaten für eine Hand oder ein Gesicht dar.

4.1.2 Bewertungsfunktion

Die Aufgabe des Hand-Trackings besteht darin, aus der gegebenen Liste der Kandidaten die wahrscheinlichste Hypothese $\mathbf{h}_t = (x_l, y_l, z_l, x_r, y_r, z_r)$ über die Position der linken und der rechten Hand zum Zeitpunkt t zu bilden. Mit jedem neuen Bild werden daher alle n möglichen Zuordnungen von Kandidaten zu

linker und rechter Hand in Form einer Liste von Zustandsvektoren $\{\mathbf{h}_t^{(1..n)}\}$ aufgestellt.

In einer Bayes'schen Formulierung des Problems unter Verwendung der Markov-Annahme berechnet sich $\mathbf{h}_t$ als

$$\mathbf{h}_t = argmax_{h \in \{\mathbf{h}_t^{(1..n)}\}}\ p(\mathbf{z}_t|\mathbf{h})\ p(\mathbf{h})\ p(\mathbf{h}|\mathbf{h}_{t-1}), \tag{4.1}$$

was eine punktweise Auswertung der drei im Folgenden beschriebenen Wahrscheinlichkeiten erforderlich macht.

Das Beobachtungsmodell

Die Wahrscheinlichkeit $p(\mathbf{z}_t|\mathbf{h})$ ist ein Maß für die Übereinstimmung zwischen Hypothese $\mathbf{h}$ und Beobachtung $\mathbf{z}_t$. Um es zu berechnen, werden für die beiden in $\mathbf{h}$ hinterlegten Handpositionen Quader ins Bild projiziert, die in ihrer Größe einer durchschnittlichen Hand entsprechen. Der Wert von $p(\mathbf{z}_t|\mathbf{h})$ berechnet sich dann proportional zum Durchschnitt der Pixelwerte des Hautfarbenbildes innerhalb der projizierten Rechtecke.

Das Anatomiemodell

$p(\mathbf{h})$ ist die a-priori Wahrscheinlichkeit der durch $\mathbf{h}$ ausgedrückten Körperhaltung. Die Körperhaltung ergibt sich aus den Handpositionen in $\mathbf{h}$ sowie der globalen Kopfposition. Der Wert von $p(\mathbf{h})$ ist hoch, wenn $\mathbf{h}$ eine häufig vorkommende Körperhaltung repräsentiert. Er beträgt 0, wenn $\mathbf{h}$ für eine Körperhaltung steht, die anatomische Beschränkungen verletzt, d. h. ein gewisser Maximalabstand einer Hand zum Kopf überschritten wird.

Um $p(\mathbf{h})$ zu berechnen, wurde ein Modell für die Aufenthaltswahrscheinlichkeiten der Hände relativ zum Kopf aufgestellt: Wie in Abbildung 4.4 dargestellt, sind die Handpositionen bei einem sich vor der Kamera bewegenden Menschen über längere Zeit betrachtet nicht gleichmäßig im Raum verteilt, sondern sie ballen sich in zwei lang gezogenen Zonen links und rechts des Torsos. Um diese Zonen zu modellieren, wurde eine dreidimensionale Gaußmischverteilung mithilfe des EM-Algorithmus auf Basis von Trainingsdaten eingelernt. Sie repräsentiert die Aufenthaltswahrscheinlichkeit der Hand relativ zum Kopf und ergibt – multiplikativ kombiniert für beide Hände aus $\mathbf{h}$ – den Wert für $p(\mathbf{h})$.

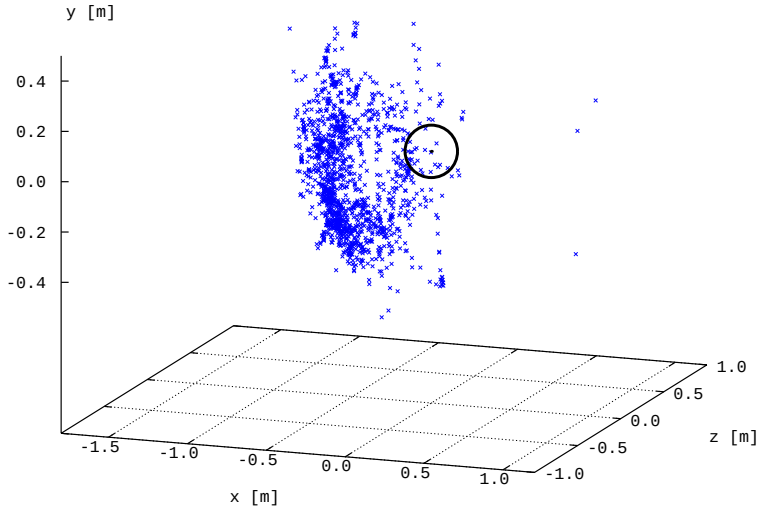


Abbildung 4.4: Koordinaten der rechten Hand relativ zum Kopf (Kreis) im Verlauf einer Trainingsaufnahme von 2min Länge.

Das Übergangsmodell

Der Term $p(\mathbf{h}|\mathbf{h}_{t-1})$ drückt die Wahrscheinlichkeit aus, mit der die Hypothese $\mathbf{h}$ der Nachfolger der Hypothese $\mathbf{h}_{t-1}$ zum zurückliegenden Zeitpunkt ist. Sei x_t die Position einer Hand zum Zeitpunkt t, dann beträgt die Distanz zwischen der auf Basis der zurückliegenden zwei Positionen gebildeten Vorhersage und der Hypothese

$$d = \|x_{t-1} + (x_{t-1} - x_{t-2}) - x_t\| \,. \tag{4.2}$$

Siehe hierzu auch Abbildung 4.5. Basierend auf d_l und d_r für die linke bzw. rechte Hand aus $\mathbf{h}$, wird dann die Übergangsfunktion wie folgt konstruiert:

$$p(\mathbf{h}|\mathbf{h}_{t-1}) \propto \max(1 - \frac{d_l}{d_{max}}, \epsilon) \cdot \max(1 - \frac{d_r}{d_{max}}, \epsilon) \tag{4.3}$$

Die Konstante d_{max} steht dabei für die größtmögliche natürliche Bewegung einer Hand in der Zeit zwischen zwei Bildern. Ist d_{max} überschritten, kommt der kleine Wert ϵ zum Tragen, der dafür sorgt, dass ein Übergang zwar unwahrscheinlich, aber doch immer möglich ist, so dass der Tracker prinzipiell statischen Fehlerzuständen entgehen kann.

4.1.3 Multi-Hypothesen-Tracking

Das genaue Verfolgen der im Vergleich zum gesamten Körper kleinen und sich schnell bewegenden Hände ist schwierig und nicht fehlerfrei. Aufgrund des identischen Beobachtungsmodells ist es nicht ohne weiteres möglich, zuverlässig zu

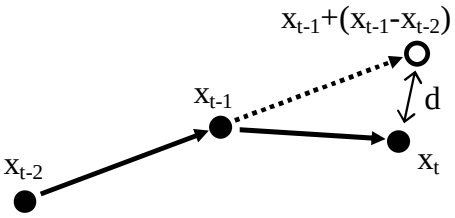


Abbildung 4.5: Die Übergangswahrscheinlichkeit basiert auf der Distanz d zwischen der zu erwartenden Position und der zu überprüfenden Position x_t.

entscheiden, welche der beiden getrackten Hände tatsächlich die linke bzw. die rechte Hand ist. Unter der Annahme, dass z. B. die rechte Hand in der Regel häufiger auf der rechten als auf der linken Körperseite zu beobachten ist, wäre es von Vorteil, wenn die Entscheidung über die Zuordnung der Hände verzögert bzw. zu einem späteren Zeitpunkt korrigiert werden könnte, anstatt dauerhaft an die falsche Zuordnung gebunden zu sein.

Durch Einführung des Multi-Hypothesen-Tracking wird diese Art von Korrekturmöglichkeit geschaffen: Nach jedem Bild wird anstatt einer einzigen Hypothese $\mathbf{h}_t$ eine n-besten Liste $\mathbf{h}_t^{1..m}$ von Hypothesen gebildet. Jede dieser Hypothesen ist mit ihrem Vorgänger aus dem letzten Zeitschritt assoziiert, wodurch eine Baumstruktur entsteht. Der Tracker sucht nun bei jedem Zeitschritt aufs Neue einen optimalen Pfad durch den Baum, auf dem er die akkumulierte Wahrscheinlichkeit des Beobachtungs-, Anatomie- und Übergangsmodelles maximiert.

Für die Offline-Bearbeitung aufgenommener Videosequenzen kann die im Baum gespeicherte Historie beliebig lang sein, sofern Speicherplatz und Rechenzeit es zulassen. Die Suche nach dem besten Pfad muss dann nur einmal am Ende der Sequenz erfolgen. In einem Live-System hingegen soll die Ausgabe der Hypothesen natürlich möglichst wenig verzögert werden, die Suche muss also nach jedem Bild erfolgen. Um den Aufwand im Rahmen zu halten, müssen hierbei die ältesten Teile des Baumes nach einer gewissen Zeit (z. B. $1s$) entfernt werden. Der Algorithmus besteht aus mehreren Schritten, die in Abbildung 4.6 zusammengefasst sind.

Die Einführung des Multi-Hypothesen-Tracking verbessert die Qualität des Hand-Trackings signifikant. Abbildung 4.7 zeigt die Verringerung des Trackingfehlers mit steigender Anzahl Hypothesen pro Frame n. Der Fall $n = 1$ entspricht dabei dem konventionellen Tracking. Als Trackingfehler werden hier solche Bilder gewertet, bei denen die vom Algorithmus geschätzte Handposition mehr als eine Handspanne (hier: $15cm$) von der manuell annotierten Handposition abweicht. Die Verwechslung von linker und rechter Hand zählt dadurch in der Regel als doppelter Fehler.

1. Stelle die Liste aller Hypothesen $s_t^{1..m}$ basierend auf den aktuellen Kandidatenregionen auf.
2. Lösche diejenigen Hypothesen $\mathbf{h}$, deren Werte für $p(\mathbf{z}_t|\mathbf{h})$ oder für $p(\mathbf{h})$ einen kleinen Schwellenwert ϵ unterschreiten.
3. Berechne für jede der m neuen Hypothesen den Wert $p(\mathbf{h}|\mathbf{h}_{t-1})$ im Bezug auf jede der n Hypothesen aus dem vorangegangenen Zeitschritt.
4. Bilde den gesamten Wert $p(\mathbf{z}_t|\mathbf{h})p(\mathbf{h})p(\mathbf{h}|\mathbf{h}_{t-1})$ für jede der $m \cdot n$ Kombinationen aus neuer und alter Hypothese.
5. Wähle die n besten Hypothesen aus der $m \cdot n$ großen Liste aus und füge jede davon als neues Kind ihres Elternknoten in den Baum ein.
6. Entferne Zweige des Baumes, die keinen Nachfolger in der aktuellen Liste der Hypothesen haben. Entferne Zweige des Baumes, die sich vom aktuell besten Zweig vor mehr als x Sekunden abgespalten haben.
7. Normalisiere die Werte der übrig gebliebenen aktuellen Hypothesen, so dass ihre Summe den Wert 1 ergibt.

Abbildung 4.6: Multi-Hypothesen-Algorithmus zum Handtracking.

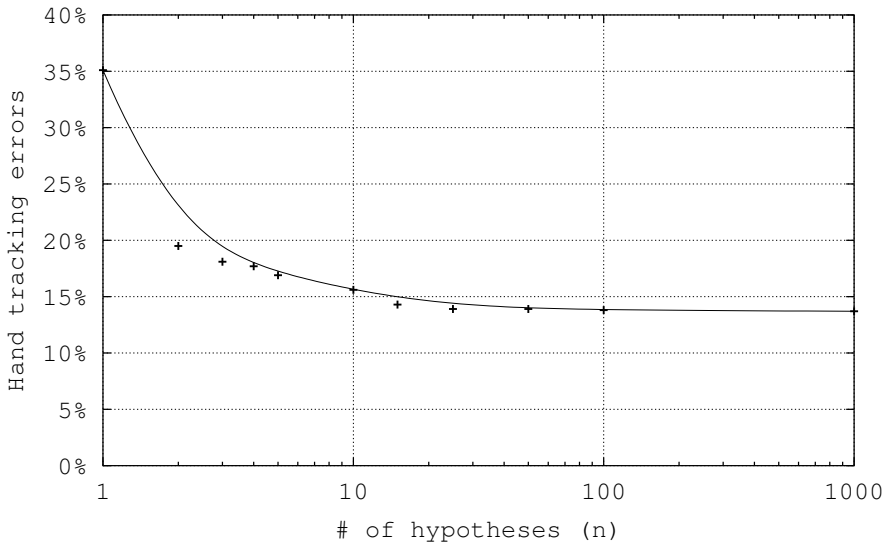


Abbildung 4.7: Prozentsatz der Bilder mit Fehlern beim Tracking der Hände im Bezug zur Anzahl der gespeicherten Hypothesen pro Frame n.

Trotz dieser Verbesserung ist das Tracking der Hände nicht völlig frei von Aussetzern und Verwechslungen. Gründe dafür sind insbesondere zeitweise Verdeckungen der Hand, Einflüsse von hautfarbenen Objekten insbesondere an der Kleidung sowie generell die hohe Geschwindigkeit der Handbewegung und die hohe Variabilität des Abbildes einer sich bewegenden Hand. Das im Folgenden beschriebene Verfahren zur Detektion von Zeigegesten muss daher in der Lage sein, mit partiell fehlerhaften Trajektorien tolerant umzugehen.

4.2 Detektion von Zeigegesten

Eine Zeigegeste durch Beobachtung der Handbewegung zu erkennen bedeutet, ein bekanntes Muster in all seiner Variation in einem störungsbehafteten Datenstrom zu finden. Dabei stellt sich insbesondere die Frage nach dem Modell, mit dem das Muster, also die Zeigegeste, modelliert werden kann, so dass die typische Vielfalt der möglichen Ausführungen berücksichtigt wird. Die beispielsweise in der Spracherkennung seit Jahren erfolgreich verwendeten kontinuierlichen Hidden-Markov-Modelle (HMMs) sind für dieses Einsatzgebiet prädestiniert: Das Muster wird durch eine Zustandsfolge repräsentiert, bei der jeder Zustand probabilistisch durch eine Gaußmischverteilung mit einem oder mehreren Mittelwerten und Kovarianzmatrizen gegeben ist. Zum Training von HMMs steht der bekannte Baum-Welch-Algorithmus zur Verfügung und zum Vergleich mit dem Anfragemuster der Viterbi-Algorithmus – siehe hierzu bspw. [Rabiner, 1989]. Im Folgenden wird beschrieben, wie Zeigegesten durch eine Gruppe von HMMs modelliert werden können, wie die eingehende Trajektorie der Hand fortlaufend damit klassifiziert wird, und welche Darstellung der Handkoordinaten zur Zeigegestenerkennung geeignet ist.

4.2.1 Phasenmodelle

Nach [Pavlovic u. a., 1997] können im Verlauf von Handgesten typischerweise drei Phasen unterschieden werden: die Vorbereitung (*preparation*), der Höhepunkt (*peak*) und der Ausklang (*retraction*). Um zur Detektion, d. h. zur zeitlichen Abgrenzung einer Zeigegeste gegenüber anderen Handbewegungen in der Lage zu sein, ist Wissen um die Charakteristika der jeweiligen Phasen notwendig. Dazu wurden 210 Zeigegesten von 15 unterschiedlichen Probanden aufgenommen und manuell annotiert. Betrachtet man die Aufnahmen im Hinblick auf die Phasen aus [Pavlovic u. a., 1997], können leicht die folgenden drei Teilabschnitte von Zeigegesten identifiziert werden:

- Vorbereitung: Die Hand bewegt sich aus einer beliebigen Startposition in Richtung des Zieles.

	μ	σ
Zeigegeste gesamt	$1.75s$	$0.48s$
Vorbereitung	$0.52s$	$0.17s$
Höhepunkt	$0.72s$	$0.42s$
Ausklang	$0.49s$	$0.16s$

Tabelle 4.1: Mittlere Dauer μ und Standardabweichung σ der einzelnen Phasen von Zeigegesten. Betrachtet wurden 210 Gesten von 15 verschiedenen Probanden.

- Höhepunkt: Die Hand verweilt bewegungslos am Punkt maximaler Auslenkung.
- Ausklang: Die Hand verlässt den Höhepunkt und kehrt in eine Ruhestellung zurück.

Die durchschnittliche Dauer der drei Phasen ist in Tabelle 4.1 dargestellt.

Da die drei Phasen stark unterschiedliche Charakteristika aufweisen, bietet es sich an, zur Modellierung der Zeigegeste nicht ein einziges HMM, sondern ein dediziertes HMM für jede der drei Phasen zu verwenden. Ein weiterer Grund für die Verwendung von drei Modellen ist der Wunsch nach einer möglichst präzisen Erkennung nicht nur der gesamten Geste, sondern insbesondere der Höhepunkt-Phase. Dieser Phase kommt später beim Ermitteln der Zeigerichtung in Abschnitt 4.3 eine besondere Bedeutung zu. Da die Höhepunkt-Phase die größte Varianz in Bezug auf ihre Dauer aufweist und zudem oft sehr kurz ist, würde sie in einem HMM für die komplette Zeigegeste nicht deutlich genug repräsentiert werden können.

Die Topologie der drei HMMs wurde experimentell bestimmt und ist in Abbildung 4.8 dargestellt. Bei dem gegebenen Umfang der Trainingsdaten (siehe Abschnitt 4.5) hat sich eine Anzahl von jeweils 3 Zuständen für die drei Phasenmodelle $M_{B,H,E}$ als optimal herausgestellt[1]. Die Ausgabewahrscheinlichkeiten der Zustände werden dabei als Mischverteilung zweier Gaußverteilungen formuliert.

Um einen Referenzwert für die Ausgaben der Phasenmodelle zu erhalten, wird ein weiteres Modell, das so genannte Null-Modell M_0, eingeführt. Es dient dazu, Zeigebewegungen von sonstigen Bewegungen abzugrenzen und repräsentiert all die Handbewegungen, die nicht einer Zeigegeste zugerechnet werden. Für M_0 hat sich eine ergodische Topologie als geeignet herausgestellt.

[1] Da die Zeigegeste bereits in drei Phasen unterteilt wurde, haben die drei Zustände innerhalb der einzelnen Phasenmodelle keine semantische Bedeutung mehr.

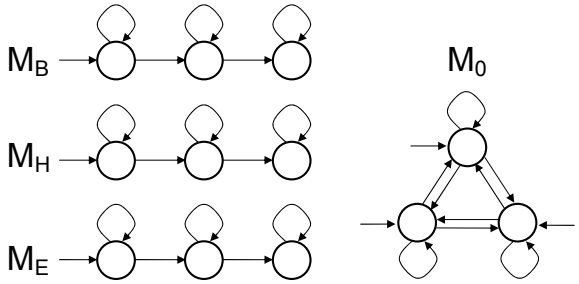


Abbildung 4.8: Um die einzelnen Phasen der Zeigegeste zu modellieren, werden HMMs mit 3 Zuständen eingesetzt (2 Gaußmischverteilungen pro Zustand). Ein zusätzliches ergodisches HMM repräsentiert sonstige Handbewegungen.

Alle Modelle werden mithilfe des EM-Algorithmus trainiert, der Aufbau des Merkmalsvektors ist in Abschnitt 4.2.3 beschrieben. Die Gestenphasen wurden für die Trainingsdaten manuell annotiert.

4.2.2 Segmentierung

Die Anwendung in der Mensch-Roboter-Interaktion erfordert eine fortlaufende Klassifikation (*run on*), d. h. eine Zeigegeste soll möglichst bald nach ihrer Ausführung vom Roboter erkannt werden, während der kontinuierliche Eingangsdatenstrom ununterbrochen weiterläuft. Demzufolge müssen die Daten nach jedem Bild klassifiziert werden – und es gibt nicht die Möglichkeit, eine falsche Entscheidung später zu korrigieren.

Die Längen der drei Zeigegestenphasen variieren stark, so dass ein Klassifikationszeitfenster fester Größe im Allgemeinen nicht genau die Geste abdecken würde, sondern entweder nur einen Teil von ihr oder aber noch zusätzliche Handbewegungen. In beiden Fällen würden die HMMs nicht gut zur Beobachtung passen. Daher wird ein Ansatz aus [Becker, 1997] verfolgt, bei dem nicht nur eine einzige, sondern eine Reihe von Sequenzen $s_{1..n}$ klassifiziert wird. Diese Sequenzen enden alle mit dem aktuellen Zeitschritt, reichen aber unterschiedlich weit in die Vergangenheit zurück. Die Längen der Sequenzen liegen im Bereich von $\mu \pm 2\sigma$ gemäß Tabelle 4.1 und decken somit die Längen aller im Training beobachteten Zeigegesten gut ab. Für jede Phase $p \in \{B, H, E\}$ wird nach der besten Subsequenz $\hat{s}_p$ gesucht, die exakt die gewünschte Gestenphase enthält. Die Suche nach der Sequenz mit der höchsten Beobachtungswahrscheinlichkeit $\hat{s}_p$ erfordert die Klassifikation aller Sequenzen. Wie von [Becker, 1997] bemerkt, kann dies effizient mit einem einzigen Durchlauf des Viterbi-Algorithmus über die längste Sequenz erledigt werden, die Wahrscheinlichkeiten für die kürzeren Sequenzen

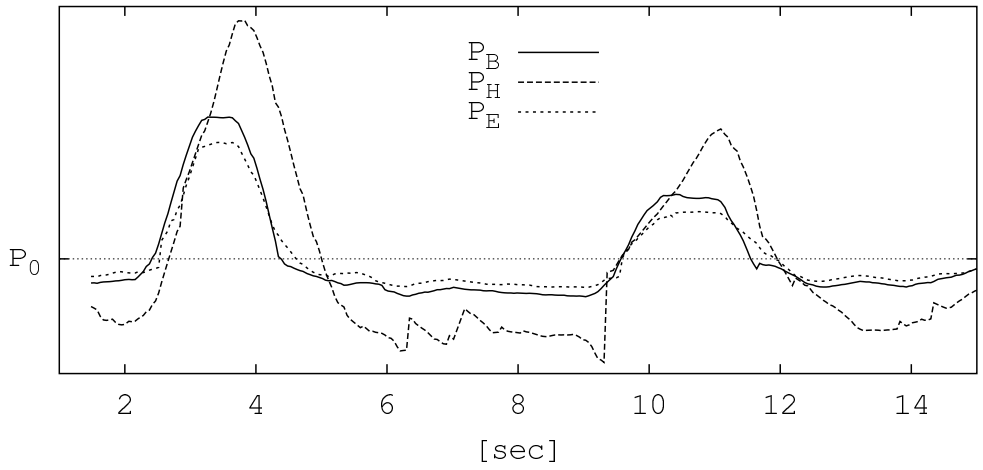


Abbildung 4.9: Log-Wahrscheinlichkeiten der Phasenmodelle in einer Sequenz mit zwei Zeigegesten.

sind Teilergebnisse dieser Berechnung. Da $P(\hat{s}_p|M_0)$ die Wahrscheinlichkeit ausdrückt, dass $\hat{s}_p$ *nicht* Teil einer Zeigegeste ist, werden damit die Ausgaben der Phasenmodelle normalisiert:

$$\begin{aligned} \hat{s}_p &= argmax\ logP(s_{1..n}|M_p) \\ P_p &= logP(\hat{s}_p|M_p) - logP(\hat{s}_p|M_0) \end{aligned} \tag{4.4}$$

Zur Detektion der Zeigegeste wird nach drei aufeinander folgenden Zeitintervallen gesucht, die hohe Ausgabewahrscheinlichkeiten P_B, P_H und P_E haben. Da P_H dazu neigt, die anderen beiden Modelle zu dominieren, wird die Klassifikationsregel wie folgt formuliert: Gesucht sind drei Zeitpunkte $t_B < t_H < t_E$, für die gilt

$$\begin{aligned} P_B(t_B), P_H(t_H), P_E(t_E) &> 0 \\ P_E(t_E) &> P_B(t_E) \\ P_B(t_B) &> P_E(t_B) \end{aligned} \tag{4.5}$$

Abbildung 4.9 zeigt an einem Beispiel den Verlauf der Ausgabewahrscheinlichkeiten der drei Phasenmodelle.

Sobald eine Zeigegeste detektiert wurde, wird innerhalb der Höhepunkt-Phase die Bestimmung der Zeigerichtung durchgeführt (siehe Abschnitt 4.3), und die Klassifikation wird für eine gewisse Totzeitspanne ausgesetzt, um eine mehrfache Erkennung derselben Geste zu verhindern.

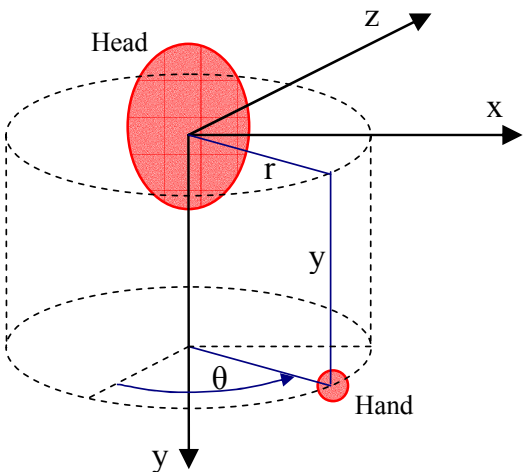


Abbildung 4.10: Die Handposition wird in ein zylindrisches Koordinatensystem im Kopfmittelpunkt transformiert.

4.2.3 Merkmale zur Gestenerkennung

Die Koordinaten der Hand werden vom Hand-Tracking in einem Weltkoordinatensystem ermittelt, d. h. sie sind abhängig von der Position des Menschen im Raum – siehe hierzu auch Abbildung 5.6 in Kapitel 5. Um den Merkmalsvektor invariant gegenüber der absoluten Position zu machen, werden die Handkoordinaten daher in ein System transformiert, das seinen Ursprung im Kopfmittelpunkt hat. Dabei wird angenommen, dass der Mensch beim Zeigen dem Roboter zugewandt ist. Da Zeigegesten sowohl mit der rechten als auch mit der linken Hand ausgeführt werden können, wird die Position der linken Hand an der Körperhauptachse gespiegelt. Beide Hände können so mit demselben Modell klassifiziert werden.

Von den verschiedenen Möglichkeiten zur Darstellung der Handposition – darunter kartesische, sphärische und zylindrische Koordinaten – haben sich die zylindrischen Koordinaten in einer Voruntersuchung als beste Wahl herausgestellt (siehe Abbildung 4.10). Um zu vermeiden, dass sich die Modelle den absoluten Positionen der Zeigeziele in den Trainingsdaten anpassen, werden die Geschwindigkeiten (Deltas) des sphärischen Winkels θ und der Höhe y anstelle der eigentlichen Werte verwendet. Der endgültige Merkmalsvektor zur Zeigegestenerkennung ist also gegeben als

$$(\Delta\theta, r, \Delta y)^T \tag{4.6}$$

Siehe hierzu auch [Campbell u. a., 1996], wo unterschiedliche Darstellungen von Handpositionen für die Gestenerkennung systematisch verglichen werden.

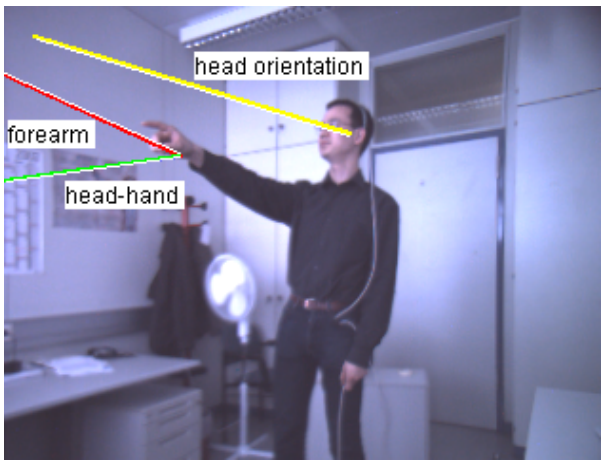


Abbildung 4.11: Unterschiedliche Ansätze zur Bestimmung der Zeigerichtung. (Die Linien wurden in 3D berechnet und zur Visualisierung ins Bild projiziert.)

4.3 Zeigerichtungsschätzung

Zum Bestimmen der Zeigerichtung wird von den Phasenmodellen der Gestenhöhepunkt bestimmt. Nur hier, während eines – im Allgemeinen kurzen – beinahe statischen Moments, lässt sich die angezeigte Richtung tatsächlich erkennen. Im Vergleich zu bisherigen Arbeiten wie z. B. [Jojic u. a., 2000], die die Handposition kontinuierlich im Verlauf der Geste zur Zeigerichtung erklären, ist dies eine Neuerung[2]. Eine kontinuierliche Schätzung kann – abhängig von der Anwendung – tatsächlich auch das gewünschte Verhalten sein, z. B. wenn es darum geht, mit der Hand den Cursor auf einem großen Bildschirm zu bewegen. Für die Erkennung dedizierter Zeigegesten in der Mensch-Roboter-Kommunikation jedoch ist es notwendig, die Information der gesamten Bewegung zu einer einzelnen Richtungsschätzung zu kondensieren.

Zum eigentlichen Schätzen der Zeigerichtung aus der Handposition wurden drei verschiedene Ansätze untersucht (siehe hierzu auch Abbildung 4.11):

Kopf-Hand-Linie: Die Kopf-Hand-Linie ist die Fortsetzung einer gedachten Verbindung zwischen Kopf und Hand. Zur Bestimmung der Linie werden dabei direkt die Kopf- und Handpositionen verwendet, wie sie vom Tracker geliefert werden.

Kopfdrehung: Da Menschen auf das Zeigeziel blicken, ist zu erwarten, dass auch der Kopf in Richtung des Ziels gedreht wird – was wiederum gemessen

[2]Eine Ausnahme hiervon stellt der Ansatz aus [Wilson und Bobick, 1998] dar, bei dem die gesamte Handbewegung genutzt wird, um den Parameter der Richtung bzw. Größe der Geste zu schätzen.

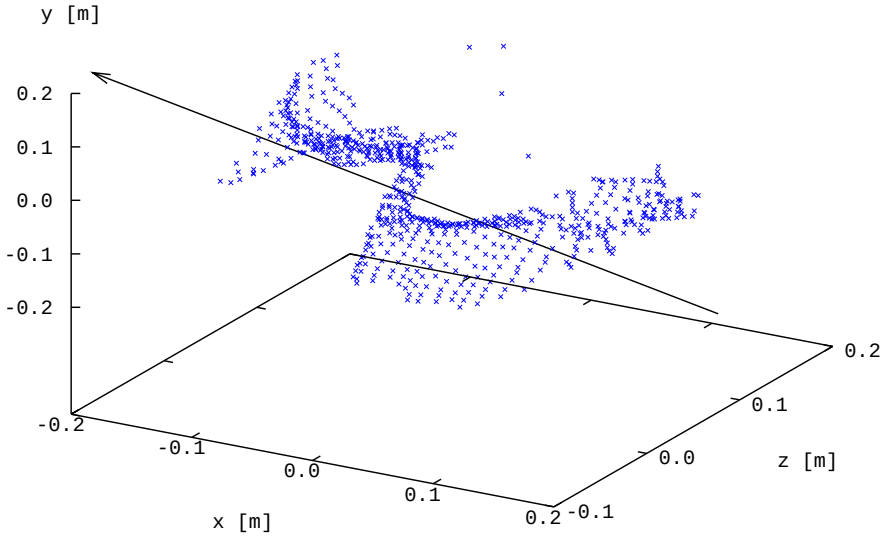


Abbildung 4.12: Eine Hauptkomponentenanalyse (PCA) der 3D-Punkte in der Umgebung der Hand ergibt die Orientierung des Unterarms (Pfeil).

und zur Bestimmung der Zeigerichtung genutzt werden kann. Die Kopfdrehung wird vergleichend mit zwei verschiedenen Verfahren bestimmt: zum einen mit einem am Kopf des Probanden befestigten Positions- und Lagesensor, und zum anderen rein visuell mit einem Verfahren, das im Anschluss in Abschnitt 4.4 beschrieben wird.

Unterarmorientierung: Zur Unterarmschätzung werden die 3D-Punkte innerhalb eines $20cm$ großen Radius um die Hand betrachtet, die aus dem Disparitätenbild gewonnen werden. Der Eigenvektor e_1 mit dem größten Eigenwert (erste Hauptkomponente) der Kovarianzmatrix der Punkte weist in Richtung der größten Varianz der Punktwolke. Da es sich beim Unterarm um ein längliches Objekt handelt, kann e_1 als Schätzwert für seine Lage interpretiert werden (siehe Abbildung 4.12). Das setzt voraus, dass sich außer dem Unterarm keine weiteren Objekte im fraglichen Radius um die Hand befinden, da diese die Form der Punktwolke beeinflussen würden. In Experimenten hat sich gezeigt, dass diese Voraussetzung gerade während des kritischen Gestenhöhepunkts in der Regel erfüllt ist, da sich die Hand dann ausgestreckt in einiger Entfernung vom Körper befindet. Zur Absicherung werden Unterarmschätzungen allerdings zurückgewiesen, wenn die erste Hauptkomponente nicht eindeutig dominiert, d. h. $e_1/e_2 < 1,5$.

Eine vergleichende Evaluation der drei Varianten zur Schätzung der Zeigerichtung findet sich später in Abschnitt 4.5.

4.4 Zeigegesten und Kopfdrehung

In den Videosequenzen, die für die vorliegende Arbeit aufgenommen wurden, konnte beobachtet werden, dass Menschen – zumindest in der ersten Hälfte einer Zeigegeste – dazu neigen, das Ziel anzuschauen. Dies mag daran liegen, dass sich die Probanden der Position des Zieles vor dem Zeigen noch einmal vergewissern wollen. Es kann aber auch sein, dass der Blick auf das Ziel eine kommunikative Funktion erfüllt, durch die die Wirkung der Geste verstärkt wird. Auch kann das Anblicken des Zieles unbewusster Ausdruck des gedanklichen Aufmerksamkeitsfokus des Menschen zum Zeitpunkt des Zeigens sein. Die intuitive Vermutung, dass Menschen im Allgemeinen an dem interessiert sind, was sie anschauen, wird durch die Literatur unterstützt [Yarbus, 1967; Barber und Legge, 1976; Glenstrup und Engell-Nielsen, 1995]. Neuere Studien [Maglio u. a., 2000; Brumitt und Cadiz, 2000] liefern zudem einen starken Anhaltspunkt dafür, dass Menschen natürlicherweise Objekte und Geräte, mit denen sie interagieren, auch anschauen.

Um zu untersuchen, ob das typische Blickmuster während einer Zeigegeste dazu genutzt werden kann, die Leistung der automatischen Zeigegestenerkennung zu verbessern, ist eine kontinuierliche Messung der Blickrichtung erforderlich. Existierende Verfahren zur Blickrichtungsmessung am menschlichen Auge erfordern den Einsatz körpergebundener Geräte oder spezieller Beleuchtung im Verbund mit hochauflösenden Bildern der Augen. Da dies bei der Kommunikation mit einem humanoiden Haushaltsroboter nicht zu erreichen ist, wird im Folgenden die Kopfdrehung als Schätzung für die eigentliche Blickrichtung verwendet.

4.4.1 Kopfdrehungsschätzung

Kopfdrehung kann im Gegensatz zur Blickrichtung nicht-invasiv mit den vorhandenen Mitteln aus den Bildern der robotereigenen Kamera geschätzt werden. Hierzu wird als Grundlage das Verfahren aus [Stiefelhagen u. a., 2000] verwendet, mit dem auch aus niedrig aufgelösten Bildern sehr schnell die horizontale und vertikale Kopfdrehung geschätzt werden kann. Die Methode arbeitet ansichtsbasiert (*appearance based*) mithilfe Künstlicher Neuronaler Netze (KNN): Zunächst wird der Kopf im Bild lokalisiert[3], in Grauwerte umgesetzt und auf eine feste Größe von 20×30 Pixel skaliert. Weitere Vorverarbeitungsschritte umfassen die Histogramm-Normalisierung des Grauwertbildes sowie das Berechnen von horizontalen und vertikalen Kantenbildern. Grauwert- und Kantenbilder werden dann zeilenweise zu einem Merkmalsvektor konkateniert und als Eingabe an ein Multi-Layer-Perzeptron mit einer versteckten Schicht angelegt. Die

[3] In [Stiefelhagen u. a., 2000] wird der Kopf als Farb-Blob im Bild einer Panoramakamera gesucht – in der vorliegenden Arbeit wird zu diesem Zweck das zuvor beschriebene Personentracking benutzt.

Ausgabeschicht besteht aus einem einzelnen Neuron, das den Drehungswinkel repräsentiert. Es existieren zwei separate Netze, eines für die horizontale (*pan*) und eines für die vertikale (*tilt*) Kopfdrehungsschätzung. In [Stiefelhagen u. a., 2000] werden auf einem Datensatz von aufgenommenen Besprechungen Ergebnisse von durchschnittlich 12° horizontalem bzw. 11° vertikalem Fehlerwinkel berichtet.

Eine der Hauptschwierigkeiten ansichtsbasierter Klassifikationsverfahren ist ihre große Abhängigkeit von den herrschenden Lichtverhältnissen. Unterschiedliche Lichtverhältnisse können das Abbild eines Kopfes in stärkerem Maße beeinflussen als dies der eigentliche Klassifikationsgegenstand, die Kopfdrehung, vermag. Insbesondere beim Einsatz auf einem mobilen Roboter sind die Lichtverhältnisse nicht mehr kontrollierbar und variieren sehr stark. Auf der anderen Seite erlauben weder die verfügbare Rechenleistung an Bord noch die Bildauflösung des Gesichtes den Einsatz von modellbasierten Verfahren, die sich auf Merkmalspunkte oder 3D-Modelle stützen.

Aus diesem Grund wird der Ansatz aus [Stiefelhagen u. a., 2000] hier weiterentwickelt, um ihn weniger anfällig gegenüber der Lichtsituation zu machen. Das neue Verfahren, gemeinsam mit [Seemann, 2003] bzw. [Seemann u. a., 2004] entwickelt, verwendet zusätzlich zum Grauwertbild auch das Disparitätenbild als Merkmal zur Klassifikation. Der Vorteil des Disparitätenbildes gegenüber dem Grauwertbild besteht darin, dass es deutlich weniger – unter für die Stereobildverarbeitung optimalen Gegebenheiten sogar gar nicht – von den herrschenden Beleuchtungsbedingungen abhängig ist. Ein zweiter Vorteil des Disparitätenbildes ist die dreidimensionale Rekonstruktion der Kopfoberfläche, die aber hier von untergeordneter Bedeutung ist, da die Tiefenauflösung aufgrund der niedrigen Bildauflösung des Kopfes gering ist.

Das Grauwertbild der Kopfregion wird auf eine feste Größe von 24×32 Pixel skaliert und histogrammnormalisiert. Gemeinsam mit dem skalierten Disparitätenbild der Kopfregion entsteht so ein Merkmalsvektor der Dimension 1536. Dieser Merkmalsvektor wird dann – analog zu [Stiefelhagen u. a., 2000] – von je einem KNN für den horizontalen bzw. den vertikalen Drehwinkel klassifiziert. Die hier verwendeten Netze haben eine Gesamtzahl von 1597 Neuronen, die in drei Schichten organisiert sind (Abbildung 4.14). Sie werden mit dem Standardverfahren *error back-propagation* personenunabhängig mit manuell annotierten Beispielbildern rotierter Köpfe trainiert. Die Referenzmessung der tatsächlichen Kopfdrehung erfolgt dabei mit einem am Kopf des Probanden befestigten Sensor, der seine Lage in 6 Freiheitsgraden anhand eines speziellen elektromagnetischen Feldes präzise bestimmen kann.

Um die mögliche Verbesserung von Kopfdrehungsschätzung durch Disparitätenbilder zu überprüfen, wurden mehrere Experimente durchgeführt. Dazu wurden Bildsequenzen von 10 verschiedenen Probanden aufgenommen, die zudem mit einem magnetischen Positions- und Lagesensor zur Messung der tatsächlichen

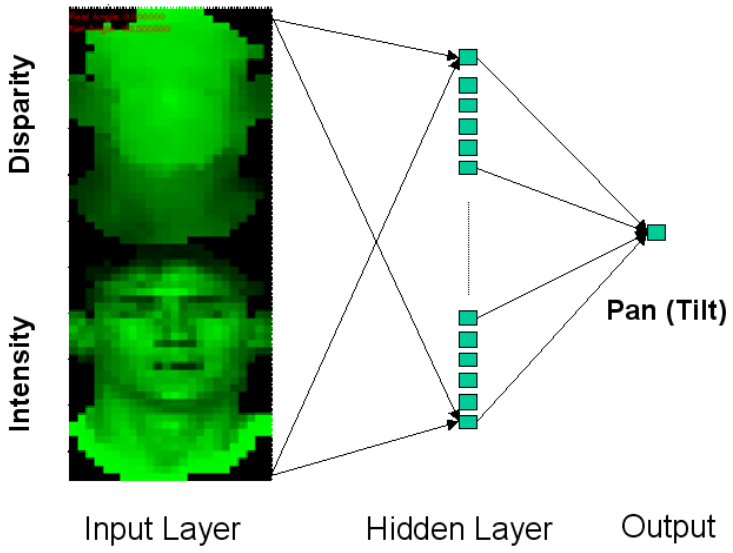


Abbildung 4.13: Zur Kopfdrehungsschätzung werden Grauwert- und Disparitätenbild auf eine feste Größe von jeweils 24×32 Pixeln skaliert und von je einem Künstlichen Neuronalen Netzwerk für den horizontalen bzw. den vertikalen Drehwinkel klassifiziert.

	horizontal	vertikal
Grauwert	$9,6°$	$8,8°$
Disparität	$11,0°$	$7,6°$
Grauwert+Disparität	$\mathbf{7,5}°$	$\mathbf{6,7}°$

Tabelle 4.2: Mittlerer Fehler der Kopfdrehungsschätzung mit Grauwertbildern, Disparitätenbildern und der Kombination aus beiden. Testbedingung: Nahaufnahmen.

Kopfdrehung ausgestattet waren (siehe Abbildung 4.13). Von jedem Probanden existieren Aufnahmen unter zwei verschiedenen Beleuchtungsbedingungen: eine mit frontalem Tageslicht und eine zweite mit seitlichem Kunstlicht.

Die Auswertung erfolgte so, dass die Netze reihum mit jeweils 9 der 10 Probanden trainiert und mit dem übrig gebliebenen Probanden getestet wurden. In Tabelle 4.2 sind die Ergebnisse des Experiments zusammengefasst: Die Kombination aus Grauwert- und Disparitätenbild übertrifft die beiden anderen Testbedingungen, die allein das Grauwert- bzw. Disparitätenbild verwenden.

Um speziell die geringere Empfindlichkeit gegenüber Beleuchtungsänderungen zu überprüfen, wurde ein zweites Experiment durchgeführt, bei dem die Netze mit den Aufnahmen einer Beleuchtungsbedingung trainiert und auf der anderen evaluiert wurden. Die Ergebnisse sind in Tabelle 4.3 aufgeführt. Hier zeigt sich, dass, wenn sich Trainings- und Testbedingungen drastisch unterscheiden, die Variante mit Disparitätenbildern allein den beiden anderen überlegen ist.

Abbildung 4.14: Ausschnitte aus dem ersten Datensatz zur Evaluation der Kopfdrehungsschätzung (Nahaufnahmen). Die beiden oberen Aufnahmen entstanden unter Tageslichtbedingungen, die unteren unter seitlichem Kunstlicht.

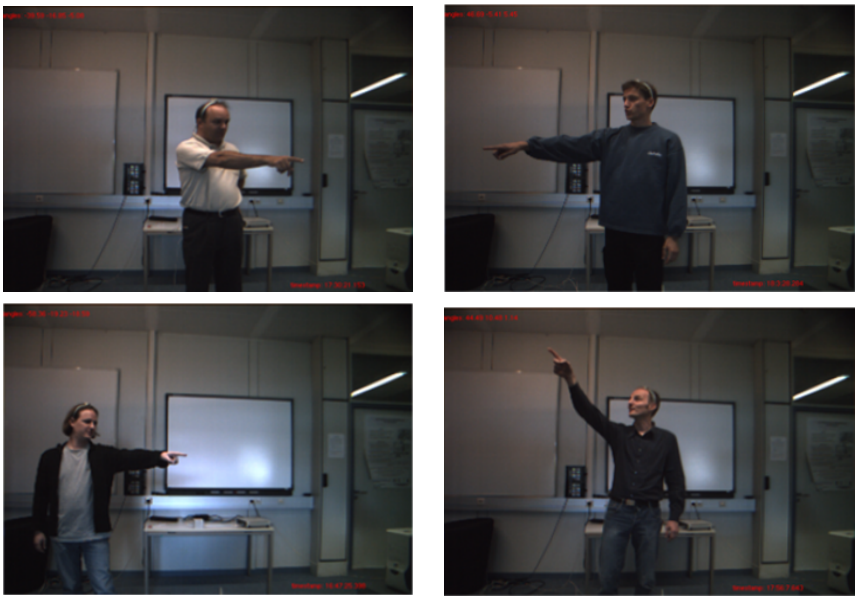


Abbildung 4.15: Zweiter Datensatz zur Evaluation der Kopfdrehungsschätzung: Mensch-Roboter-Szenario.

	horizontal
Grauwert	$13,9°$
Disparität	$\mathbf{9,6}°$
Grauwert+Disparität	$10,6°$

Tabelle 4.3: Mittlerer Fehler der Kopfdrehungsschätzung mit Grauwertbildern, Disparitätenbildern und der Kombination aus beiden. Testbedingung: Nahaufnahmen mit veränderter Beleuchtung.

	horizontal	vertikal
Grauwert	$15,5°$	$6,3°$
Disparität	$11,0°$	$5,7°$
Grauwert+Disparität	$\mathbf{9,7}°$	$\mathbf{5,6}°$

Tabelle 4.4: Mittlerer Fehler der Kopfdrehungsschätzung mit Grauwertbildern, Disparitätenbildern und der Kombination aus beiden. Testbedingung: Mensch-Roboter-Szenario.

In einem dritten Experiment wurde ein Szenario nachgestellt, das von den Bedingungen her der typischen Mensch-Roboter-Kommunikationssituation ähnelt: So war die Distanz zwischen Mensch und Kamera größer (ca. $2-3m$), und die Menschen konnten sich während der Aufnahme frei im Raum bewegen (siehe Abbildung 4.15). An diesem Experiment nahmen 6 Probanden teil. Tabelle 4.4 zeigt die Ergebnisse.

Zusammenfassend wird deutlich, dass in allen Experimenten der mittlere Fehler durch Hinzunahme des Disparitätenbildes signifikant sinkt. Besonders stark ist dies der Fall, wenn Trainings- und Testdaten unter unterschiedlichen Beleuchtungsbedingungen aufgenommen wurden. Die absoluten Werte der personenunabhängigen Schätzung von $9,7°$ horizontal und $5,6°$ vertikal im Mensch-Roboter-Szenario erscheinen gut genug, um als zusätzliches Merkmal in die Zeigegestenerkennung einfließen zu können. Abbildung 4.16 zeigt abschließend den Verlauf der Schätzung im Vergleich zum gemessenen Wert in einer Beispielsequenz von ca. $20s$ Dauer. Für detailliertere Angaben zur Methode und zum Versuchsaufbau siehe [Seemann u. a., 2004].

4.4.2 Kopfdrehung als Merkmal zur Zeigegestenerkennung

Entsprechend der eingangs beschriebenen Beobachtung, dass Menschen beim Zeigen auch das Referenzobjekt anblicken, gilt es nun eine Möglichkeit zu finden, die Erkennung von Zeigegesten, basierend auf Handbewegungen, mit den

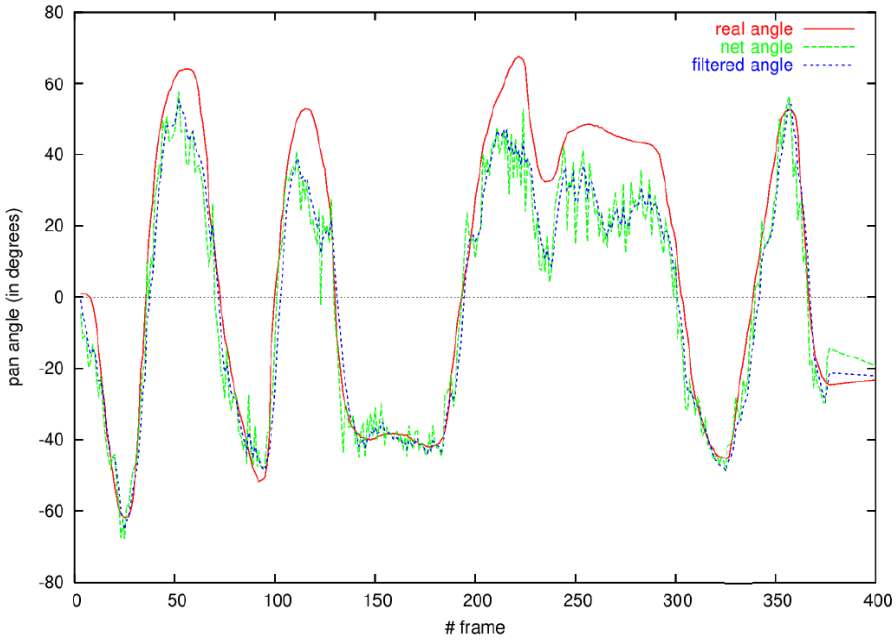


Abbildung 4.16: Geschätzte horizontale Kopfdrehung im Vergleich zum Sensormesswert. Zusätzlich eingezeichnet: der mit einem Kalmanfilter geglättete Schätzwert.

Werten aus der Kopfdrehungsschätzung zu kombinieren. Da nicht von vornherein bekannt ist, wie die Korrelation von Handbewegung und Kopfdrehung während einer Zeigegeste genau beschaffen ist, ist es nahe liegend, diese Beziehung automatisch, d. h. datengetrieben, durch die Hidden-Markov-Modelle lernen zu lassen.

Da Kopfdrehung und Handposition synchron in jedem Frame der Videosequenz gemessen werden, kann die Fusion bereits auf Merkmalsebene stattfinden. Zu den drei Handkoordinaten (siehe 4.6) werden dem Merkmalsvektor nun zwei weitere Dimensionen mit Kopfdrehungsinformation hinzugefügt. Dabei werden die Kopfdrehungswinkel aber nicht mit ihrem absoluten Wert in den Merkmalsvektor aufgenommen, da dieser – ähnlich wie schon bei den Handkoordinaten – abhängig von der Position des Referenzobjekts ist. Vielmehr ist die Differenz in der Auslenkung zwischen Kopf und Hand relevant: bezeichne θ_{Kopf} und ϕ_{Kopf} den Azimut- bzw. den Polarwinkel des Kopfes sowie θ_{Hand} und ϕ_{Hand} den Azimut- bzw. den Polarwinkel der Hand in einem Kopf-zentrierten sphärischen Koordinatensystem. Dann werden mit θ_{HR} und ϕ_{HR} die Winkeldifferenzen zwischen Kopfdrehung und Handposition berechnet:

$$\begin{aligned} \theta_{HR} &= |\theta_{Kopf} - \theta_{Hand}| \\ \phi_{HR} &= |\phi_{Kopf} - \phi_{Hand}| \end{aligned} \tag{4.7}$$

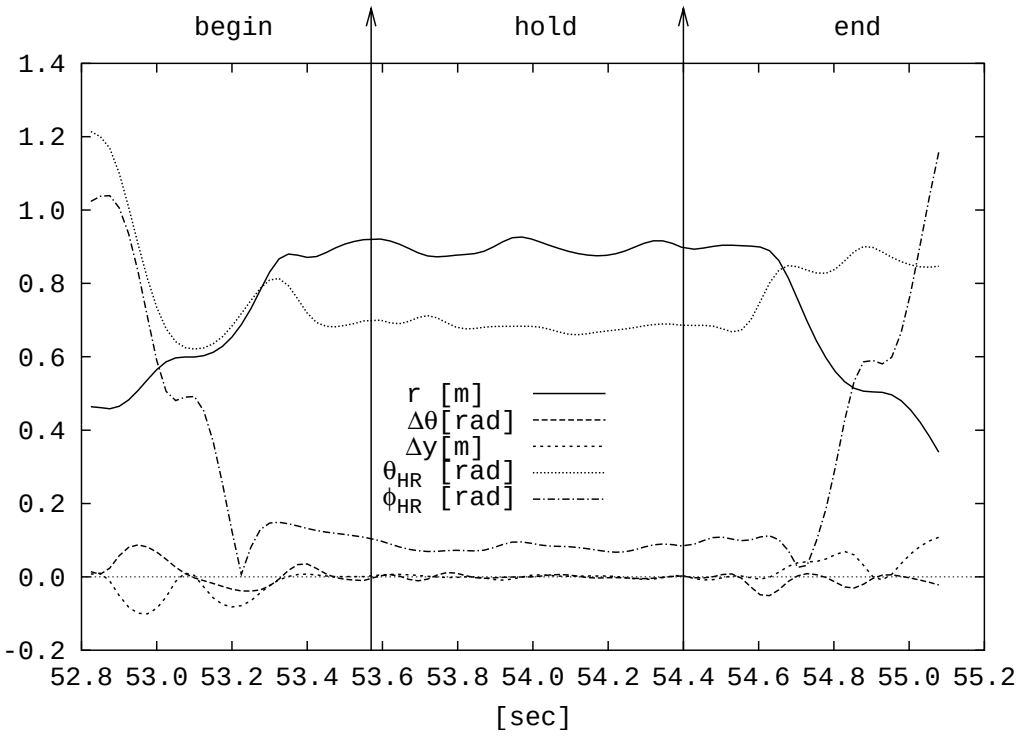


Abbildung 4.17: Entwicklung der Merkmale im Verlauf einer typischen Zeigegeste.

Der kombinierte Merkmalsvektor zur Zeigegestenerkennung ist somit gegeben als

$$(\Delta\theta, r, \Delta y, \theta_{HR}, \phi_{HR})^T \tag{4.8}$$

Abbildung 4.17 zeigt den Verlauf der 5 Komponenten des Merkmalsvektors im Verlauf einer typischen Zeigegeste. Es ist zu erkennen, dass die Werte der Kopf-Hand-Winkeldifferenz θ_{HR} bzw. ϕ_{HR} in der Vorbereitungsphase sinken und in der Ausklangphase wieder steigen. Zum Höhepunkt der Geste sind beide Werte niedrig, was bedeutet, dass Kopf und Hand in dieselbe Richtung orientiert sind.

4.5 Evaluation der Zeigegestenerkennung

Um die Qualität der automatischen Zeigegestenerkennung zu evaluieren, wurde ein Versuchsaufbau mit 8 unterschiedlichen Zielen errichtet (siehe Abbildung 4.18). Die Probanden wurden gebeten, sich im Sichtfeld der Kamera zu bewegen, und von Zeit zu Zeit auf eines der ausgezeichneten Ziele zu zeigen. Ihr Kommunikationspartner sollte dabei der Roboter, d. h. die Kamera sein. Insge-

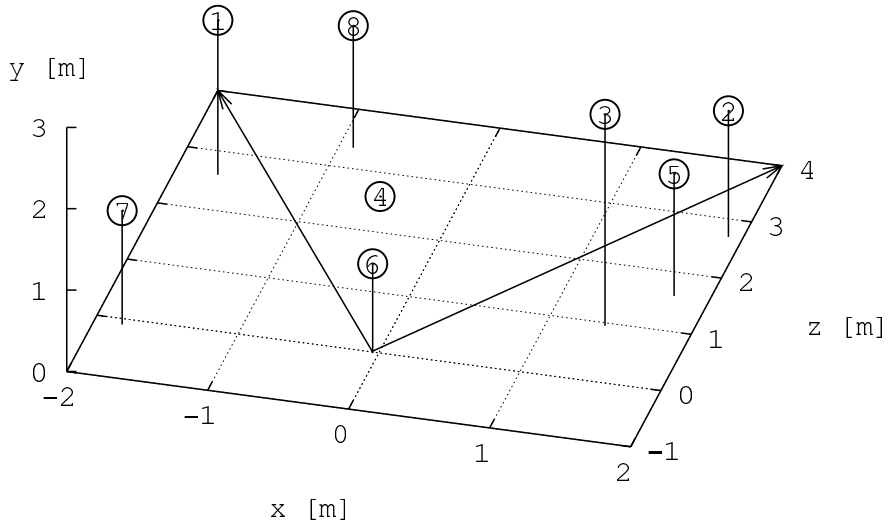


Abbildung 4.18: Markierte Ziele im Versuchsaufbau. Ziel Nr. 6 ist die Kamera selbst. Die Pfeile markieren das Sichtfeld der Kamera.

samt wurden auf diese Weise 129 Zeigegesten von 12 verschiedenen Probanden aufgenommen.

4.5.1 Genauigkeit der Zeigerichtungsschätzung

Zunächst soll die Genauigkeit der Zeigerichtungsschätzung unabhängig von möglichen Störeinflüssen aus der automatischen Gestendetektion gemessen werden. Daher wurden die für die Richtungsschätzung relevanten Höhepunkt-Phasen der Zeigegesten manuell annotiert. Als einzige Fehlerquelle bleibt also nur noch das Hand-Tracking: Aufgrund von Kalibrierungsungenauigkeiten und der begrenzten Auflösung der Stereobildverarbeitung kann sich eine kleine, aber unvermeidliche Abweichung zu den handvermessenen Koordinaten der Ziele ergeben.

In den Experimenten stellte sich heraus, dass die Kopf-Hand-Linie einen mittleren Fehlerwinkel von 25° erreichte. Damit war in 90% aller Fälle eine korrekte Identifikation des Zieles möglich. Die Unterarm-Linie funktionierte mit 39° mittlerem Fehler deutlich schlechter, was vermutlich an der ungenauen Messung des Unterarms liegt, die im Vergleich zu Kopf und Hand stärker schwankte. Zudem stellte sich heraus, dass die Probanden fast ausschließlich mit ausgestrecktem Arm zeigten, weshalb selbst bei einer genaueren Unterarmmessung kein Vorteil gegenüber der Kopf-Hand-Linie zu erwarten wäre. Zum Vergleich wurde auch die mit dem Sensor gemessene Kopfdrehung – stellvertretend für die Blickrichtung – zur Richtungsbestimmung verwendet. Da sie sich der Kopf-Hand-Linie gegenüber als nicht signifikant überlegen herausstellte, wurde dieser Ansatz nicht weiterverfolgt. In Tabelle 4.5 sind die Ergebnisse zusammengefasst.

	Kopf-Hand-Linie	Unterarm-linie	Kopfdrehung (Sensor)
a) Mittlerer Fehlerwinkel	25°	39°	22°
b) Ziele korrekt	90%	73%	75%
c) Verfügbarkeit	98%	78%	(100%)

Tabelle 4.5: Vergleich der drei Ansätze zur Schätzung der Zeigerichtung: a) mittlerer Fehlerwinkel zwischen extrahierter Zeigelinie (3D) und dem gemessenen Zielpunkt, b) Prozentsatz der Gesten, für die das korrekte Ziel (1 von 8) ermittelt werden konnte, und c) Verfügbarkeit der Messwerte während der Höhepunkt-Phase.

4.5.2 Erkennungsleistung

Zur Bestimmung der Zeigegesten-Erkennungsleistung wurde eine *leave-one-out* Strategie verwendet, die die vorhandenen Daten optimal nutzt: Evaluiert wurde jeweils mit den Daten eines Probanden, während die Daten der anderen Probanden zum Training verwendet wurden. Dies wurde für alle Probanden wiederholt, und die Ergebnisse wurden gemittelt. Die Erkennungsleistung wird mit zwei unterschiedlichen Maßen beschrieben:

- Die Erkennungsrate (*recall*) gibt den Anteil der korrekt detektierten Gesten in Bezug auf die Gesamtzahl aller ausgeführten Gesten an.
- Die Erkennungsgenauigkeit (*precision*) gibt den Anteil korrekt detektierter Gesten an, bezogen auf die Gesamtzahl aller detektierten Gesten (inklusive Fehlerkennungen).

Um herauszufinden, ob die Zeigegestenerkennung vom Merkmal Kopfdrehung profitieren kann, wurde die Evaluation mit drei unterschiedlichen Merkmalsvektoren durchgeführt:

1. nur Handposition
2. Handposition und Kopfdrehung vom Sensor
3. Handposition und visuell geschätzte Kopfdrehung

Zusätzlich zur Detektion der Gesten wurde bei den drei Versuchsbedingungen jeweils auch die Qualität der Zeigerichtungsschätzung auf den nun automatisch detektierten Gestenintervallen gemessen. Die Zeigerichtung wurde hierbei immer nach der Kopf-Hand-Linie ermittelt. Tabelle 4.6 fasst die Ergebnisse zusammen.

Das Baseline-System ohne Kopfdrehung erzielte ca. 80% Erkennungsrate bei einer Erkennungsgenauigkeit von 74%. Wenn Kopfdrehung dem Merkmalsvektor hinzugefügt wurde, stieg die Genauigkeit signifikant von 74% auf 87%, wobei

	Erkennungs-rate (recall)	Erkennungs-genauigkeit (precision)	Richtungs-fehler
nur Handposition	$79,8\%$	$73,6\%$	$19,4°$
mit Kopfdrehung (Sensor)	$78,3\%$	$86,3\%$	$16,8°$
mit Kopfdrehung (visuell)	$78,3\%$	$87,1\%$	$16,9°$

Tabelle 4.6: Erkennungsleistung der personenunabhängigen Zeigegestenerkennung mit und ohne Kopfdrehung.

die Erkennungsrate auf gleich hohem Stand blieb. D. h. die Anzahl der Fehlerkennungen (*false positives*) konnte durch Einsatz der Kopfdrehungsschätzung um 50% (relativ) gesenkt werden. Erfreulicherweise machte es dabei kaum einen Unterschied, ob die Kopfdrehungsinformation dabei vom präzisen Sensor oder aus der fehlerbehafteten visuellen Schätzung stammte. Bei der Zeigerichtungsschätzung konnte durch Einsatz der Kopfdrehungsinformation der mittlere Fehler von $19,4°$ auf $16,9°$ gesenkt werden. Da in beiden Fällen die Richtungsschätzung gleichermaßen mit der Kopf-Hand-Linie erfolgte, ist die Verbesserung allein auf die genauere zeitliche Segmentierung des Gestenhöhepunkts zurückzuführen.

4.6 Zeigegesten und Sprache

Bei der Interaktion mit einem humanoiden Roboter sind Zeigegesten allein kein hinreichendes Kommunikationmittel – sie entfalten ihre Wirkung erst in Verbindung mit Sprache. Dabei gibt es verschiedene Möglichkeiten, wie Zeigegesten in Verbindung mit Sprache auftreten:

Als Verstärkung einer Äußerung
„*Schalte den Fernseher ein!*“ + Zeigegeste in Richtung Fernseher.
In diesem Fall ist die Bedeutung der Geste redundant und kann allenfalls als zusätzliche Absicherung bei der Interpretation der Äußerung dienen.

Als Alternative zu einer Äußerung
„*Bring mir die Tasse!*“
„*Welche Tasse soll ich bringen?*“
Alternative 1: „*Die gelbe Tasse auf dem Tisch!*“
Alternative 2: Zeigegeste in Richtung einer bestimmten Tasse.

Als notwendige Ergänzung zu einer Äußerung
„*Stelle es genau dort hin!*“ + Zeigegeste an einen bestimmten Ort.
Die Position verbal zu beschreiben wäre hier nicht praktikabel.

1) nur Sprache	2) Sprache und Zeigegeste
M: *Please switch on the light!*	M: *Please switch on the light!*
R: *Which light?*	R: *Which light?*
M: *The big lamp.*	M: *This lamp!* + <Zeigegeste>
R: *Switching on the big lamp.*	R: *Switching on the big lamp.*

Tabelle 4.7: Beispieldialog zwischen Mensch (M) und Roboter (R).

In Zusammenarbeit mit [Holzapfel u. a., 2004] wurde ein Ansatz entwickelt, der Zeigegesten und Sprache semantisch fusioniert, um damit einen Roboter zu kommandieren. Dabei geht es darum, Zeigegesten, die Objekte im Raum referenzieren und die alternativ zur sprachlichen Objektbenennung verwendet werden, korrekt zu interpretieren und mit der Gesamtaussage in Verbindung zu bringen. Das System ermöglicht Dialoge zwischen Mensch und Roboter, wie sie beispielhaft in Tabelle 4.7 dargestellt sind. Im Folgenden sollen der Ansatz und die durchgeführten Experimente kurz dargestellt werden. Für eine weitergehende Beschreibung des Systems siehe [Holzapfel u. a., 2004].

4.6.1 Fusionsalgorithmus

Der hier verwendete Fusionsalgorithmus arbeitet regelbasiert auf Semantikebene. Dabei gibt es eine konzeptionelle Trennung zwischen einem anwendungsunabhängigen Parser und anwendungsabhängigen Fusionsregeln. Zur Repräsentierung der Semantik werden *Typed Feature Structures* (TFS) eingesetzt, siehe hierzu [Carpenter, 1992]. Der Parser nutzt *Constraints*, um zu bestimmen, welche Elemente zusammengefügt werden können. Anschließend wird die Ausgabe mithilfe von Konstruktionsregeln erzeugt. Das Zusammenfügen von Sprache und Gesten erfolgt hierbei nicht nur anhand von festen deiktischen Referenzwörtern wie „the“ , „this“ , „that“ , etc. – denn dies würde unter anderem ein fehlerfreies Funktionieren des Spracherkenners voraussetzen. Vielmehr läuft das Zusammenfügen informationsgesteuert ab, indem Objekttypen aus der Ontologie verglichen werden. Dabei werden Entscheidungen nicht hart gefällt, sondern im Sinne von n-besten Listen verwaltet.

Eingabeereignisse werden auf semantischer Ebene als Eingabetokens bezeichnet. Die Fusion findet statt, indem die passendste Regel auf eine Menge von Eingabetokens angewendet wird. Jede dieser Regeln definiert auf ihrer linken Seite, wie mehrere Tokens kombiniert werden. Auf ihrer rechten Seite stehen Vorbedingungen und Constraints, nach denen die Regel angewendet werden kann. Dazu gehören Angaben wie Anzahl, Modalität, Zeit und semantischer Kontext der Eingabetokens. Abbildung 4.19 zeigt als Beispiel eines Eingabetokens die TFS einer Zeigegeste. Die Auflösung einer Zeigegeste, d. h. ihre Zuordnung zu einem Objekt, geschieht dabei mit einer gewissen Konfidenz, die im Bild als *SCORE*

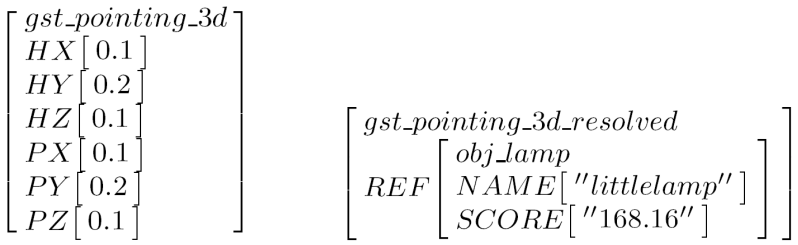


Abbildung 4.19: Eine Zeigegeste in ihrer semantischen Repräsentation als *Typed Feature Structure*. Angegeben sind in der Ursprungsform (links) Handposition und Zeigerichtung, anhand derer Objekte im Raum aufgelöst werden können (rechts).

bezeichnet ist. Der Zuordnungsalgorithmus erzeugt dabei nicht nur eine einzelne, sondern eine sortierte n-besten Liste von möglichen Auflösungen.

Zur Spracherkennung wird das Janus Recognition Toolkit [Finke u. a., 1997] mit dem Ibis-Decoder [Soltau u. a., 2001] verwendet. Dabei kommen im Spracherkenner und im Dialogmanager dieselben Kontextfreien Grammatiken (CFG) zum Einsatz, um den geparsten Baum mittels Konvertierungsregeln in TFS umzusetzen. Die Grammatik des Testsystems umfasste dabei 164 Nicht-Terminale und ca. 1000 Terminale, durch die ca. 232mio Eingabesätze generiert werden können.

Da bei der Fusion von Sprache und Gesten in einem Live-System der Ankunftszeitpunkt der Informationen nicht notwendigerweise linear ist – bspw. könnte eine Zeigegeste Z erst nach einem Sprechakt S weitergemeldet werden, obwohl tatsächlich Z vor S begonnen hat – erfolgt das Parsen immer auf einer Arbeitsmenge von Elementen, die für eine gewisse Zeit bereitgehalten werden, bis sie erfolgreich zusammengefügt werden können oder verfallen.

4.6.2 Experimente zur Fusion

Der multimodale Fusionsalgorithmus wurde in einem Mensch-Roboter-Interaktionsszenario getestet, bei dem der Benutzer den Roboter mittels Sprache und Zeigegesten instruieren konnte. Die Gesten konnten dazu benutzt werden, zwischen drei im Raum platzierten Lampen zu unterscheiden, deren Positionen – zusammen mit den Positionen von fünf weiteren Objekten – dem System bekannt waren. Am Experiment nahmen 7 Probanden teil, die in der Summe ca. 500 Eingaben produzierten, von denen 102 multimodaler Natur waren, d. h. Sprache in Verbindung mit Zeigegesten enthielten.

Zunächst wurde auf diesen Daten der zeitliche Zusammenhang zwischen Zeigegesten und Sprache untersucht. Dazu wurden die Gestenphasen manuell annotiert und mit der dazugehörigen sprachlichen Äußerung verglichen, die Ergebnisse

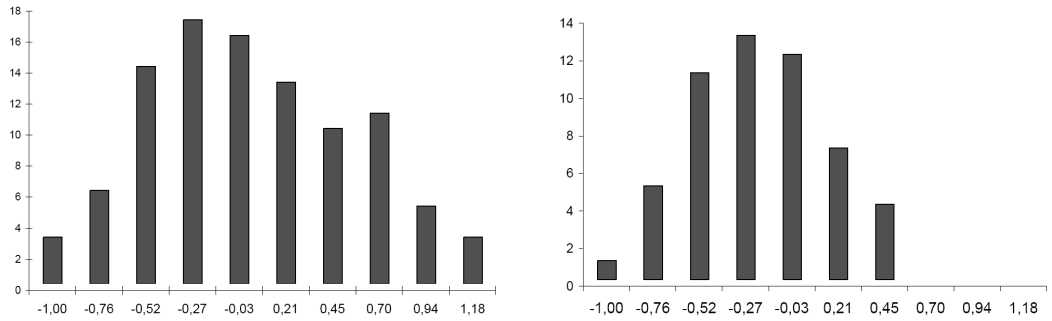


Abbildung 4.20: Zeitlicher Zusammenhang (in Sekunden) zwischen dem Anfang des Höhepunktes der Zeigegeste und dem Anfang des Sprachsignals (links) bzw. dem Anfang eines deiktischen Wortes (rechts). Ein Wert $< 0s$ bedeutet, dass der Höhepunkt der Geste vor der Sprache beginnt.

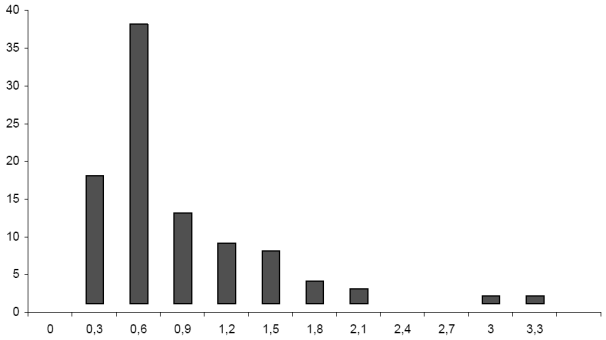


Abbildung 4.21: Dauer des Gestenhöhepunktes in den Testdaten (Angabe in Sekunden).

sind in Form einer Häufigkeitsverteilung in Abbildung 4.20 dargestellt. Es zeigt sich, dass die Mehrheit der Gestenhöhepunkte in einem Intervall von ca. $0,5s$ vor bis $0,7s$ nach Anfang des Sprachsignals beginnt, woraufhin der zeitliche Grenzwert zur Fusion im Folgenden überschlägig mit $\pm 1s$ abgeschätzt wurde. Zusätzlich wurde auch die zeitliche Korrelation des Gestenhöhepunkts mit dem Auftauchen eines deiktischen Wortes untersucht. Im Mittel begann der Gestenhöhepunkt $0,3s$ vor Beginn des deiktischen Wortes. Die Varianz war mit $0,14s$ sehr gering, so dass von einer starken Korrelation zwischen Gestenhöhepunkt und deiktischen Wörtern ausgegangen werden kann. Abschließend ist in Abbildung 4.21 noch die Verteilung der Dauer der Gestenhöhepunkte in den Testdaten dargestellt.

Als nächstes wurde die Qualität der Gestenerkennung überprüft. Von den 102 Zeigegesten konnten 89 korrekt erkannt werden, was einer Erkennungsrate (*recall*) von 87% entspricht. Die Auflösung der Gesten zu Zielobjekten erfolgte über einen Winkelvergleich zwischen geschätzter Zeigerichtung und tatsächli-

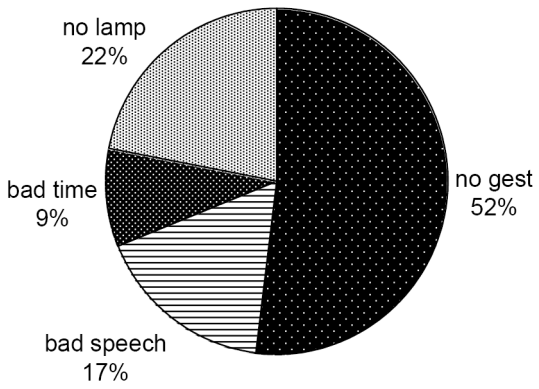


Abbildung 4.22: Verteilung der Fehlerklassen bei der multimodalen Fusion von Sprache und Zeigegesten. Zu weiteren Details siehe [Holzapfel u. a., 2004].

cher Richtung zum fraglichen Objekt. In 94% der Fälle befand sich nach der Auflösung das korrekte Objekt in der n-besten Liste, allerdings nur in 44% der Fälle auf dem ersten Platz. Dieses Ergebnis unterstreicht die Wichtigkeit der n-besten Liste, da so über die Ontologie in der Regel trotzdem noch der korrekte Zusammenhang hergestellt werden kann.

Dass dies auch gelingt, zeigt sich in der dritten Auswertung, bei der das Ergebnis der multimodalen Fusion betrachtet wurde. Von den insgesamt 102 multimodalen Benutzereingaben konnten 74% korrekt erkannt werden. Die Gründe für die verbleibenden 26% Fehler sind in Abbildung 4.22 aufgeführt: So scheiterte die Erkennung in 52% (relativ) daran, dass keine Geste erkannt wurde, und in weiteren 22% daran, dass die Auflösung des Zielobjekts nicht gelang. 17% der Fehler beruhten auf Fehlern des Spracherkenners und die restlichen 9% darauf, dass Sprache und Geste einen zu großen Zeitversatz hatten, um noch korrekt zugeordnet werden zu können.

5 Integration in den humanoiden Roboter ARMAR-III

Die in den Kapiteln 3 und 4 vorgestellten Verfahren – Personentracking, Hand-Tracking und Zeigegestenerkennung – werden nun in den humanoiden Roboter ARMAR-III integriert. Dort bilden sie gemeinsam ein Softwaremodul zur visuellen Benutzermodellierung, das im Folgenden als ARTHUR bezeichnet wird,[1] siehe hierzu auch [Nickel, 2008]. Dieses Softwaremodul wird wiederum mit anderen im SFB-588 existierenden Modulen verknüpft, die auf ARMAR-III zur Laufzeit für Objekterkennung, multimodalen Dialog, Sprecherlokalisierung, Handlungsausführung und Visualisierung sorgen.

Der humanoide Roboter ARMAR-III, siehe [Asfour u. a., 2006], verfügt über insgesamt 43 Freiheitsgrade: davon entfallen 7 auf Kopf und Halsgelenk, 3 auf das Hüftgelenk, 7 auf jeden der beiden Arme, 8 auf jede Hand und 3 auf die holonome Plattform. Die Lokalisierung der Plattform erfolgt durch Odometrie sowie durch Vergleich von Laserscannermessdaten mit einer Umgebungskarte. Der anthropomorphe Sensorkopf ist mit zwei unabhängig schwenkbaren Augen ausgestattet, in denen je zwei Kameras montiert sind, und zwar jeweils mit einem Weitwinkel- und einem Teleobjektiv. Die Kameras liefern Farbbilder mit einer Auflösung von 640×480 Punkten bei einer Framerate von bis zu $30 fps$. Die Übertragung des Sensormusters (Bayer-pattern) geschieht dabei über die IEEE-1394-Schnittstelle (FireWire), die Konvertierung nach RGB findet im PC statt.

Auf ARMAR-III ist für die Bildverarbeitung ein PC mit einem 1,6GHz Pentium M Prozessor[2] vorgesehen, den ARTHUR sich mit anderen Modulen teilt, die zur Laufzeit parallel ausgeführt werden. Da das Tracking von Personen und ihren Händen eine Framerate von mindestens $10 fps$ erfordert, darf also die Verarbeitung eines Stereobildpaares nicht länger als $100 ms$ dauern – eine effiziente Realisierung ist daher unerlässlich.

Um dieses Ziel zu erreichen, ist sowohl bei der Konzeption der Algorithmen als auch bei ihrer Implementierung eine Reihe von Maßnahmen erforderlich, die im folgenden Abschnitt dargestellt werden. Anschließend wird die Einbettung

[1] ARTHUR = Active Real-time Tracking for a Humanoid Robot

[2] Die Angaben beziehen sich auf den Stand im Jahr 2007.

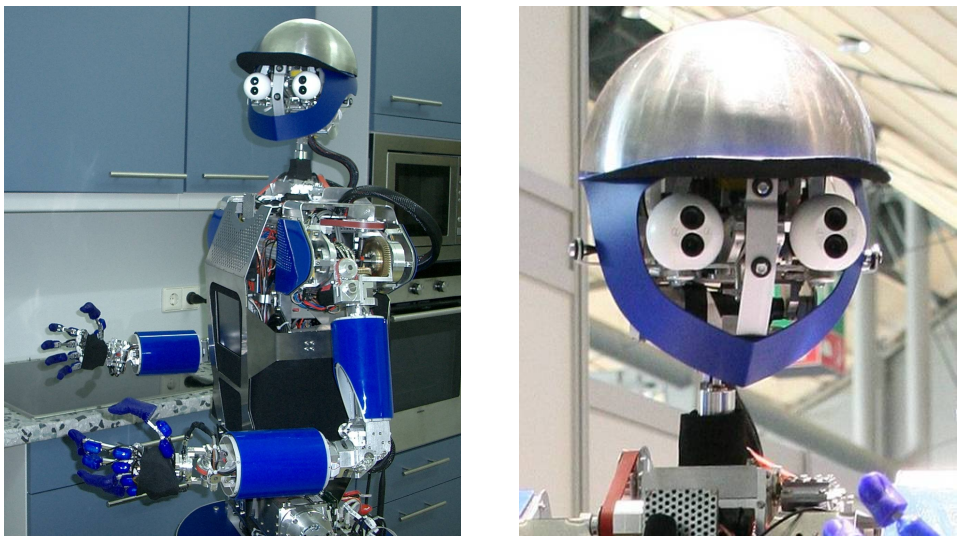


Abbildung 5.1: Der humanoide Roboter ARMAR-III.

von ARTHUR in das Gesamtsystem von ARMAR-III beschrieben. Im letzten Abschnitt werden einige Demonstrationsszenarien geschildert, in denen ARMAR-III durch Zusammenspiel aller Komponenten verschiedene Aufgaben bewältigt.

5.1 Maßnahmen zur Effizienzsteigerung

Implementierung in C++: Das Zielsystem auf ARMAR-III wird mit Linux betrieben, C++ wird dort von Entwicklungswerkzeugen bestens unterstützt und kann optimiert für die konkrete Prozessorarchitektur übersetzt werden. C++ stellt daher die erste Wahl dar. (Zur Laufzeit interpretierte Sprachen kommen unter den gegebenen Anforderungen nicht in Betracht.)

Nutzung von SSE-Befehlen: Als SSE (*Streaming SIMD Extensions*) wird eine Erweiterung des Befehlssatzes von x86-Prozessoren bezeichnet, durch die mit einer einzigen Anweisung mehrere elementare Operationen parallel ausgeführt werden können. Dazu existieren spezielle 128bit breite Register, in denen sich z. B. 16 einzelne Bytes in einem Rechenschritt addieren oder subtrahieren lassen. In ARTHUR wird SSE zur Berechnung der Stereokorrelation eingesetzt: Hier muss der Betrag der Differenz von zahlreichen Bildregionen berechnet werden, was durch Einsatz von SSE um den Faktor 16 beschleunigt wird.

Einfache Merkmale: Wie in Abschnitt 3.3 beschrieben, werden zur Personenlokalisierung – so wie später auch zum Handtracking – ausschließlich einfache Merkmale verwendet: Farbe in Form von Histogramm-Rückprojektion, Bewegung in Form von Hindergrundsubtraktion, Haar-feature-Detektoren

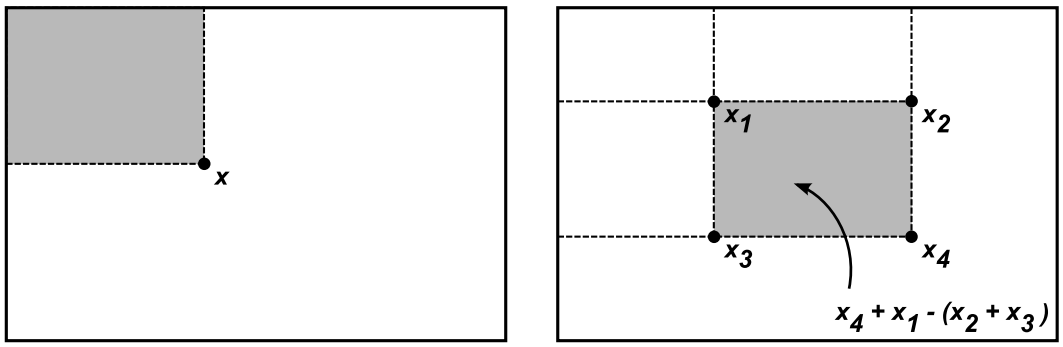


Abbildung 5.2: Im Integralbild nach [Viola und Jones, 2001] ist in jedem Punkt die Summe der Pixel eines Grauwertbildes oberhalb und links des entsprechenden Punktes eingetragen (Abbildung links); es kann mit einem einzigen Durchgang durch das Grauwertbild aufgestellt werden. Das Integralbild wird dazu genutzt, die Summe der Pixel eines beliebigen Rechtecks im Grauwertbild zu berechnen. Hierfür sind nur zwei Additionen und eine Subtraktion nötig. So berechnet sich die Summe des Rechtecks in der rechten Abbildung als $x_4 + x_1 - (x_2 + x_3)$.

nach [Viola und Jones, 2001] und Stereokorrelation. Einfach sind diese Merkmale in der Hinsicht, dass zu ihrer Berechnung nur ein einziger Durchgang durch das Bild bzw. die Bilder erforderlich ist. Der Aufwand wächst also lediglich linear mit der Bildgröße.[3] Dies gilt, wie in Abschnitt 3.3 dargestellt, auch für die Haar-feature-Detektoren, da diese im verwendeten Verfahren nur punktweise ausgewertet werden.

Einsatz von Integralbildern: Die Auswertung der Merkmale geschieht ausschließlich in rechteckigen Bildregionen, deren Summen sich durch Verwendung von Integralbildern nach [Viola und Jones, 2001] sehr schnell bilden lassen (siehe Abbildung 5.2). Der Aufwand zur Auswertung der Merkmale ist daher nicht abhängig von der Größe der Regionen, sondern stets konstant.

Aufgabenangemessene Skalierung: Nicht jedes Merkmal muss in voller Auflösung berechnet werden. So genügt es für die meisten Merkmale, das um den Faktor 2-4 skalierte Eingangsbild zu bearbeiten, um noch die gewünschte räumliche Auflösung zu erzielen.

Berücksichtigung räumlicher Randbedingungen: Wie in Abschnitt 3.4.1 beschrieben, werden zur Personenlokalisierung Partikelfilter in einem 3D-Zustandsraum eingesetzt und durch Projektionen der Partikel in den Bildraum ausgewertet. Dabei handelt es sich um ein top-down-Verfahren, das

[3] Auch für das Disparitätenbild kann dies durch dynamisches Programmieren erreicht werden, siehe z. B. [Veksler, 2003].

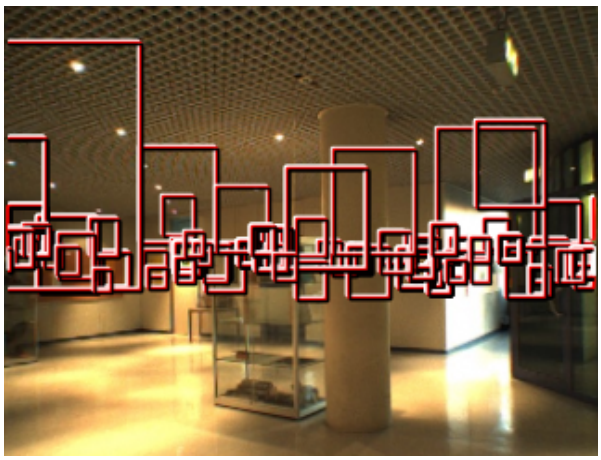


Abbildung 5.3: Reduktion des Suchraums durch räumliche Randbedingungen: Eingezeichnet sind 300 zufällig gewählte Suchregionen für den Kopf, bei deren Erzeugung die Größe des Raumes und die Höhe eines stehenden Menschen $(1,5m-2,0m)$ vorgegeben wurde. Erkennbar ist, dass große Bildbereiche von der Suche ausgeschlossen werden können.

es ermöglicht, schon bei Bildung der Hypothesen auf bekannte Randbedingungen Rücksicht zu nehmen. So werden von vornherein keine Hypothesen gebildet, die außerhalb des Raumes liegen, oder die zu hoch über der Bodenebene sind, als dass es sich dabei um Menschen handeln könnte (siehe Abbildung 5.3).

Suchraumoptimierung durch Prädiktion: Unter Berücksichtigung der Systemdynamik und durch Resampling stellt der *Condensation*-Algorithmus nach [Isard und Blake, 1998a] sicher, dass zu jedem Zeitschritt eine bezüglich der bekannten Annahmen optimale a-priori Partikelmenge gebildet wird. Es wird dadurch – ganz im Gegensatz zu einer erschöpfenden Suche im Bildraum – keine Rechenzeit damit vergeudet, Bereiche des Suchraums auszuwerten, die keinen ausreichenden Bezug zum letzten geschätzten Zustand haben.

Nutzung von Synergien: Zwischenergebnisse der Bildverarbeitung, wie z. B. die Disparitätenkarte, werden nur ein einziges Mal berechnet, auch wenn sie an den verschiedensten Stellen in unterschiedlichen Teilmodulen benötigt werden. Da die Menge solcher gemeinsam genutzter Informationen groß ist, liegt die Verwendung einer Blackboard-Architektur nahe.

Um weitere Synergien zu nutzen, wird zudem das Modul zur bildbasierten Gesichtsidentifikation von [Ekenel u. a., 2007] vollständig in ARTHUR aufgenommen. Durch eine enge Verknüpfung mit der Personenlokalisierung kann auf die erneute Suche nach dem Gesicht – ein notwendiger Schritt vor der Identifikation – verzichtet und so der Rechenaufwand deutlich reduziert werden.

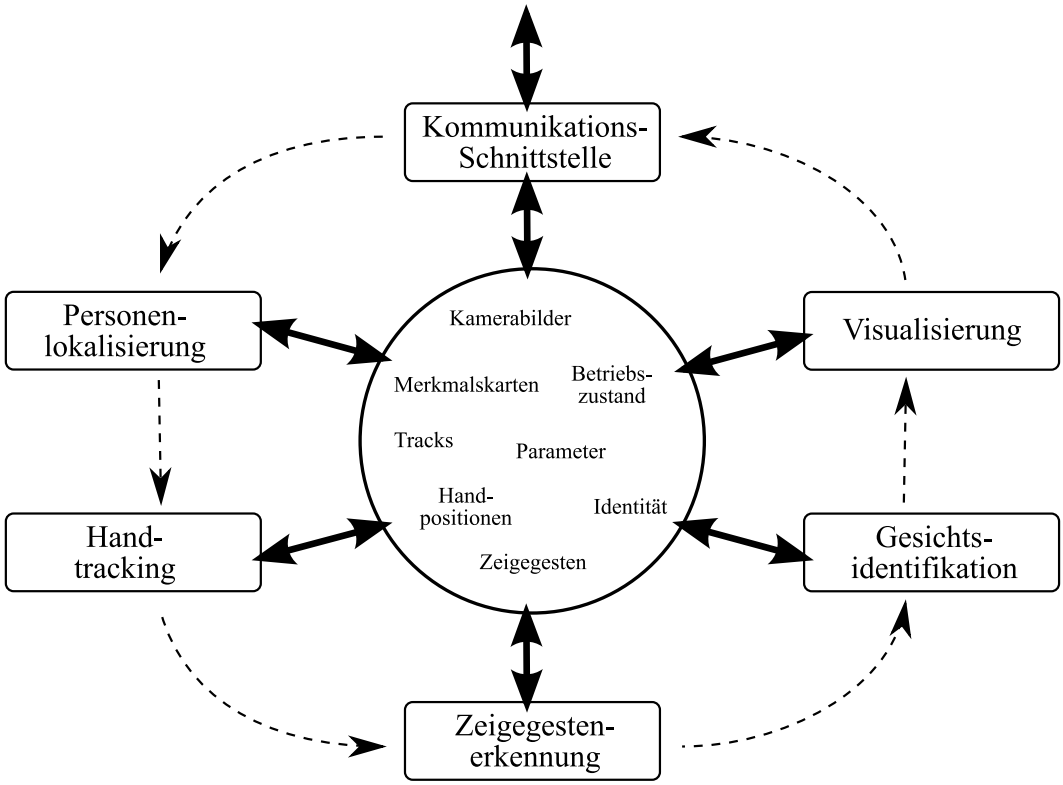


Abbildung 5.4: Aufbau des Softwaremoduls ARTHUR zur visuellen Benutzermodellierung. Die einzelnen Funktionsgruppen sind um ein zentrales Blackboard angeordnet, in dem gemeinsam genutzte Daten und Parameter abgelegt werden. Die gestrichelten Pfeile verdeutlichen die zeitliche Abfolge der Verarbeitungsschritte.

Die interne Struktur von ARTHUR ist in Abbildung 5.4 skizziert. Alle Teilmodule haben dabei Zugang zu einem zentralen Datenpool, dem so genannten *Blackboard*, in dem Zwischenergebnisse, Parameter und Betriebszustände abgelegt werden. Die Kommunikation nach außen geschieht über eine zentrale Schnittstelle, die u. a. an die verteilte Regelungsarchitektur MCA2 von ARMAR-III angeschlossen ist.

5.2 Zusammenspiel der Softwaremodule

Neben den in der vorliegenden Arbeit entwickelten Verfahren, die im Modul ARTHUR vereint sind, existieren auf ARMAR-III verschiedene andere Softwaremodule, die eng mit der visuellen Benutzermodellierung zusammenarbeiten. Dazu gehören:

Objekterkennung: Die Objekterkennung nach [Azad u. a., 2006] ist in der Lage, Objekte mittels 3D-Modellen zu lokalisieren und ihre Lage zu schätzen (Teller, Tasse, etc.). Zusätzlich kann sie gleich geformte Objekte anhand ihrer Textur unterscheiden (z. B. Apfelsaft- versus Orangensaftkarton).

Spracherkennung: Die Spracherkennung von ARMAR-III arbeitet sprecherunabhängig und erkennt dabei auch die Grenzen zwischen zwei Äußerungen (zur Segmentierung siehe [Kraft u. a., 2005]). Zum Einsatz kommen wahlweise die Kopfmikrofone des Roboters, siehe [Kraft und Wölfel, 2007], oder ein Nahbesprechungsmikrofon für den Benutzer, wenn die Umgebungsgeräusche hoch sind.

Sprecherlokalisierung: Die akustische Quellenlokalisierung nach [Bechler u. a., 2004] bestimmt die räumliche Position (in Form von Polar- und Azimutwinkel) eines Sprechers anhand der Laufzeitunterschiede des Schalls, der von den Mikrofonen am Roboterkopf aufgenommen wird. Die Sprecherlokalisierung wird zum Auffinden des Benutzers verwendet, wenn dieser vom visuellen Tracker nicht erfasst ist.

Dialogmanagement: Der multimodale Dialogmanager nach [Holzapfel, 2008] bildet semantische Repräsentationen der eingehenden Sprache und Gestik. Er ist in der Lage, z. B. bei fehlenden Informationen Rückfragen zu stellen oder dynamisch das Dialogziel zu wechseln. Wie in Abschnitt 4.6 beschrieben, werden hier Zeigegesten und sprachliche Äußerungen verknüpft und gemeinsam interpretiert. Zur Kommunikation mit dem Benutzer wird Sprachsynthese eingesetzt.

Ablaufsteuerung: Die zentrale Ablaufsteuerung regelt das zeitliche Zusammenspiel der Module während der Demonstrationsszenarien. Dazu gehört u. a. die Entscheidung darüber, ob der Roboterkopf den Sollwerten der visuellen bzw. akustischen Personenlokalisierung folgt, um den Benutzer zu fixieren, oder sich suchend nach einem bestimmten Objekt umsieht. Dazu ist die Ablaufsteuerung direkt in MCA2[4] eingebunden, mit deren Hilfe ARMAR-III gesteuert wird.

Visualisierung: Die dreidimensionale Visualisierung beinhaltet Position und Zustand des Roboters, die Aufenthaltsorte und Identitäten der Personen in seiner Umgebung sowie eine Darstellung der Szene und der in ihr enthaltenen Objekte (hier: Einrichtung der Experimentierküche).

Abbildung 5.5 gibt eine Übersicht über die beteiligten Module. Die Kommunikation untereinander geschieht auf verschiedenen Wegen: direkter Funktionsaufruf, MCA2, oder Punkt-zu-Punkt Netzwerkkommunikation. Dabei werden zwischen ARTHUR und seinen beiden direkten Kommunikationspartnern, der Ablaufsteuerung und dem Dialogmanager, folgende Nachrichten ausgetauscht:

[4] MCA2 (Modular Controller Architecture) ist eine netzwerktransparente verteilte Regelungsarchitektur für Roboter, siehe `http://www.mca2.org`.

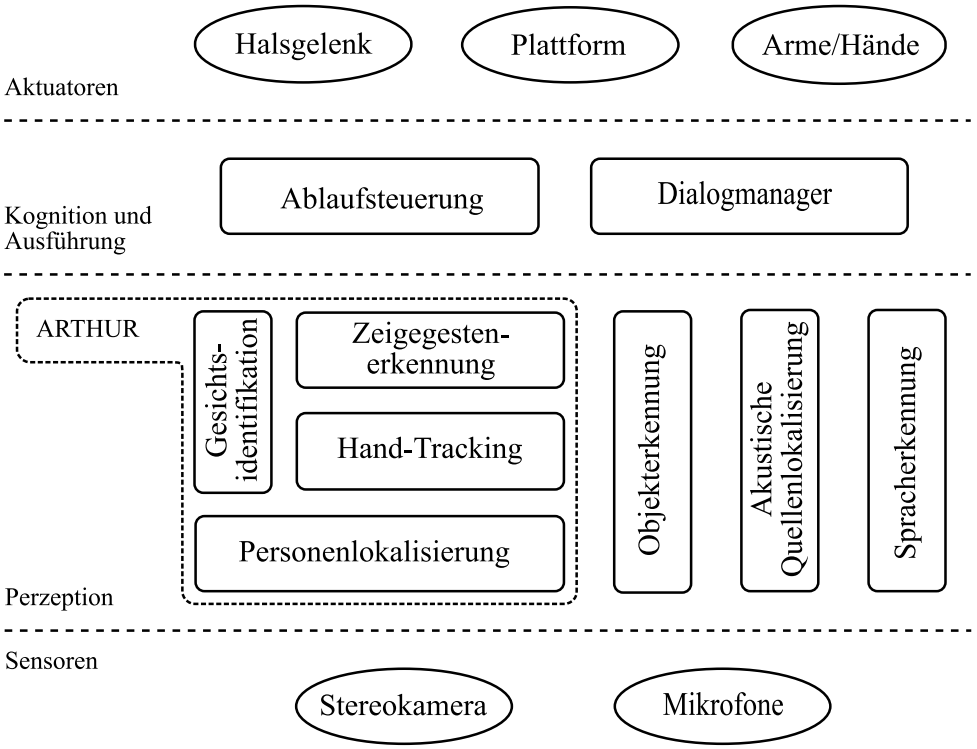


Abbildung 5.5: Softwaremodule zur Perzeption auf ARMAR-III. Die Skizze beschränkt sich auf Komponenten mit direktem Kontakt zur visuellen Benutzermodellierung, d. h. andere notwendige Funktionseinheiten wie z. B. Greifplanung sind hier nicht dargestellt.

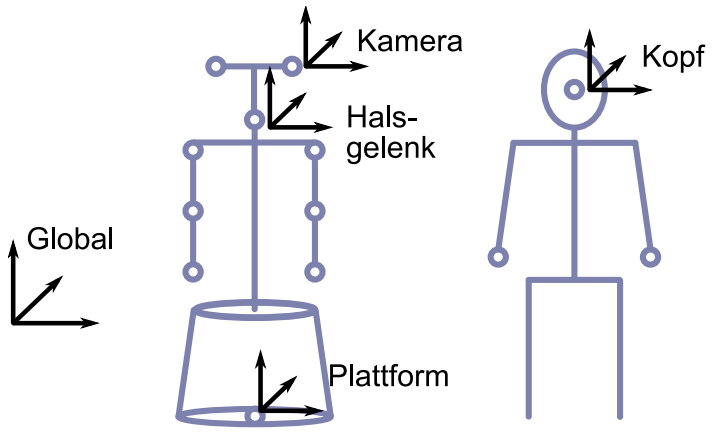


Abbildung 5.6: Überblick über die verschiedenen Koordinatensysteme.

- Ablaufsteuerung ⇆ ARTHUR:
  - → Linkes und rechtes Kamerabild
  - → Translationsvektor und Rotationsmatrix von Kamerakopf und Plattform
  - → Nachricht: ARTHUR an/aus
  - ← Positionen und Identitäten der Personen
- Dialogmanager ⇆ ARTHUR:
  - ← Positionen und Identitäten der Personen
  - ← Richtungsvektor erkannter Zeigegesten
  - → Nachricht: Start/Stop Gesichtstraining
  - ← Zustand des Gesichtstrainings

Notwendig für das Zusammenspiel der Module ist auch die Verwendung gemeinsamer Koordinatensysteme. Abbildung 5.6 skizziert die für die Funktion von ARTHUR wichtigsten Koordinatensysteme.

5.3 Fähigkeiten des Gesamtsystems

Die Fähigkeiten von ARMAR-III wurden in den vergangenen Jahren in zahlreichen Demonstrationen vorgeführt, wie z. B. zur CeBIT 2006, auf der ARMAR-III u. a. Besucher fixierte und begrüßte, Objekte lernte und erkannte, und natürlichsprachliche Dialoge mit Menschen führte. Im Folgenden sollen nun einzelne Fähigkeiten beschrieben werden, die durch Teamarbeit im Rahmen des SFB 588 unter Verwendung von ARTHUR realisiert werden konnten.

Personen fixieren

Mithilfe der Personenlokalisierung kann ARMAR-III seinen Sensorkopf auf den Dialogpartner ausrichten und automatisch nachführen. Dazu werden die von ARTHUR gelieferten Koordinaten des Benutzers mittels inverser Kinematik in Gelenkwinkelstellungen für die Freiheitsgrade des Halsgelenks umgerechnet und an die Halsgelenkregelung übergeben.

Der dadurch entstehende Effekt eines „aufmerksamen“ Roboters ist sowohl beim ersten Entdecken einer Person als auch während des weiteren Dialogs von entscheidender Bedeutung für die Wahrnehmung des Roboters durch den Menschen – und hat in Vorführungen stets aufs Neue für Faszination bei Besuchern gesorgt.

Personen folgen

Als Erweiterung der Fixierung kann ARMAR-III auch Personen folgen. Dabei wird die Benutzerposition zusätzlich auch der Plattformsteuerung als Sollwert vorgegeben, wobei darauf geachtet wird, dass ein Abstand von ca. $1m$ zum Benutzer stets eingehalten wird. Abgesichert wird die Fahrt von ARMAR-III permanent durch eine übergeordnete 360° Umgebungsüberwachung mithilfe dreier Laserscanner, die an der Plattform dicht über dem Boden angebracht sind.

Die Fähigkeit zu folgen erlaubt das berührungslose Führen des Roboters z. B. in Form einer Tour durch zuvor unbekannte Räume. Aktiviert wird der Folgemodus durch den Dialogmanager per Sprachkommando „follow me“.

Personen erkennen

Alle Gesichter, die die Personenlokalisierung entdeckt, werden von der Gesichtsidentifikation aus [Ekenel u. a., 2007] mit einer Menge von bekannten Gesichtern verglichen. Die Ergebnisse der Identifikation werden den entsprechenden Tracks zugeordnet und dort akkumuliert. So steigt die Konfidenz für die Identität einer Person mit der Zeit an, da immer mehr Klassifikationen von einzelnen Gesichtsansichten als Entscheidungsgrundlage zur Verfügung stehen.

Blickt ein Benutzer ARMAR-III beispielsweise für ca. $2sec$ frontal an, kann in der Regel bereits die Identität ermittelt und von ARTHUR weitergeleitet werden. Daraufhin begrüßt der Dialogmanager den Benutzer und spricht ihn fortan mit seinem Namen an.

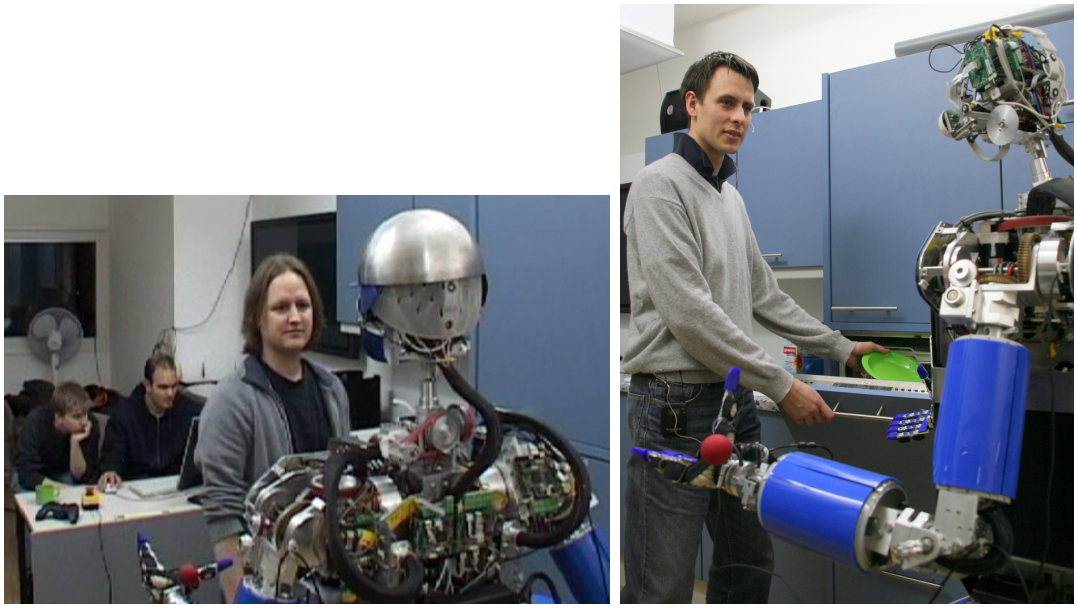


Abbildung 5.7: Interaktion mit ARMAR-III in der Experimentierküche.

Personen kennenlernen

Wird eine Person von der Gesichtsidentifikation als zuvor ungesehen klassifiziert, ist der Dialogmanager nach [Holzapfel und Waibel, 2008] dazu in der Lage, interaktiv einen neuen Namen zu lernen. Dabei wird der Benutzer zunächst gebeten, seinen Namen zu buchstabieren und diesen nach der Erkennung zu bestätigen. Bereits während dieses Dialogs wird von ARTHUR ein neues Modell mit den Gesichtsansichten des noch Unbekannten trainiert. Führt der Dialog zum Erfolg, teilt der Dialogmanager ARTHUR den Namen der Person mit, und das neue Modell wird gespeichert. In Zukunft kann ARMAR-III dann die Person wiedererkennen und mit ihrem Namen ansprechen.

Objekte bringen

Die Aufgabe, dem Benutzer ein Objekt zu bringen, besteht aus zahlreichen Teilschritten: zunächst teilt der Benutzer ARMAR-III in einem Dialog per Sprache oder Zeigegeste mit, welches Objekt gebracht werden soll:

„Please bring me the cup!“
„Which cup do you want me to bring?“
„This cup!“ + Zeigegeste in Richtung der Tasse.

Die Auflösung des Objekts erfolgt dabei durch Vergleich der Zeigerichtung mit den Einträgen in einer Datenbank, die die ungefähre Position aller Objekte im Raum enthält.

Sprache und Geste werden dabei wie in Abschnitt 4.6 beschrieben zu einer gemeinsamen semantischen Struktur fusioniert und können so direkt in eine Handlung umgesetzt werden. Anschließend fährt der Roboter die Position an, lokalisiert das Objekt präzise mithilfe der Objekterkennung und greift es mit seiner Hand. Dann wird mithilfe der Personenlokalisierung der Benutzer ausfindig gemacht und angefahren. Zuletzt streckt ARMAR-III seinen Arm in Richtung des Benutzers aus und lässt ihn das Objekt ergreifen.

6 Zusammenfassung

Die vorliegende Arbeit befasste sich mit der visuellen Benutzermodellierung für einen humanoiden Roboter. Dabei wurden Verfahren entwickelt, um mit Mitteln der Bildverarbeitung eine komplette Perzeptionskette zu realisieren. Diese besteht aus dem räumlichen Lokalisieren des Benutzers, dem Tracking der Hände sowie der Kopfdrehung, und basierend darauf der automatischen Detektion menschlicher Zeigegesten sowie deren Richtungsschätzung.

Zur Personenlokalisierung wurde ein Ansatz basierend auf Partikelfiltern eingesetzt, der durch Fusion mehrerer einfacher Merkmale sowohl schnell als auch robust arbeitet. Kernstück des Verfahrens ist der hier entwickelte DI^2-Algorithmus, der die Methode der Demokratischen Integration [Triesch und Malsburg, 2001] zur dynamischen Merkmalsfusion auf partikelfilterbasiertes Tracking überträgt. Dabei geht DI^2 von einem allgemeineren Merkmalsbegriff aus, der neben den unterschiedlichen Merkmalstypen wie Farbe oder Bewegung auch die unterschiedlichen Bildregionen des Zielobjekts als gleichwertige Teilnehmer eines dynamischen Wettbewerbs betrachtet. So kann durch ein und dasselbe Verfahren sowohl der Ausfall einzelner Merkmalstypen als auch die Verdeckung einzelner Körperregionen bzw. Kameraansichten kompensiert werden.

Bei Experimenten mit einem Stereokamerakopf zeigte sich, dass durch Einsatz von DI^2 der Prozentsatz fehlerhafter Frames (*misses* + *false positives*) von zusammen $18,3\%$ auf $9,2\%$ fast halbiert werden konnte. Der DI^2-Algorithmus ist ein universelles Verfahren zur Merkmalsfusion im Partikelfilter und kann in unterschiedlichen Situationen eingesetzt werden. Um diese Vielseitigkeit zu demonstrieren, wurden auch Experimente zum Tracking in einer Mehrkameraumgebung (*smart room*) durchgeführt. Hier ergab sich ein ähnliches Bild: Die Trackinggenauigkeit (MOTA) wurde durch DI^2 von $88,7\%$ auf $94,0\%$ gesteigert, während sich gleichzeitig die mittlere Ortsabweichung (MOTP) von $113mm$ auf $65mm$ reduzierte.

Für das 3D-Handtracking wurde ein probabilistisches Multihypothesen-Suchverfahren vorgestellt, das basierend auf Hautfarbsegmentierung und Tiefeninformation aus dem Disparitätenbild den wahrscheinlichsten Aufenthaltsort der Hände schätzt. Dabei wurde die Kopfposition aus der Personenlokalisierung als räumliche Einschränkung des Suchraums verwendet.

Zur Kopfdrehungsschätzung wurde ein ansichtsbasiertes Verfahren aus [Stiefelhagen u. a., 2000] weiterentwickelt mit dem Ziel, höhere Robustheit gegenüber

Beleuchtungsänderungen zu erreichen. Der Schlüssel hierzu lag in der zusätzlichen Verwendung des Disparitätenbildes, das sich gegenüber dem Grauwertbild durch seine inhärente Beleuchtungsinvarianz auszeichnet. Durch Experimente u. a. in einem Mensch-Roboter-Szenario konnte gezeigt werden, dass sich auf diese Weise der Schätzfehler von $15,5°$ auf $9,7°$ horizontal bzw. von $6,3°$ auf $5,6°$ vertikal reduziert.

Zur Zeigegestenerkennung wurde die Geste in drei Phasen zerlegt, jede davon repräsentiert durch ein dediziertes Hidden-Markov-Modell. Die Modelle wurden mit den Hand-Trajektorien hunderter Zeigegesten trainiert und zeigten sich in einem Experiment in der Lage, Zeigegesten mit einer Erkennungsrate (*recall*) von $79,8\%$ und einer Genauigkeit (*precision*) von $73,6\%$ zu detektieren. Anschließend wurde untersucht, inwieweit die Beobachtung, dass Menschen beim Zeigen das Zielobjekt anblicken, genutzt werden kann, um die Zeigegestenerkennung zu verbessern. Hierzu wurden die visuell geschätzten Kopfdrehungswinkel in den Merkmalsvektor aufgenommen und das Experiment wiederholt. Es zeigte sich, dass bei ungefähr gleich bleibender Erkennungsrate die Genauigkeit auf $87,1\%$ anstieg – dies entspricht einer Reduktion der *false positives* um ca. 50%. Auch bei der Zeigerichtungsschätzung ergab sich eine Verbesserung des Fehlerwinkels von $19,4°$ auf $16,9°$. Dadurch konnte nachgewiesen werden, dass Kopfdrehung tatsächlich ein wichtiges Merkmal zur automatischen Erkennung von Zeigegesten ist, und dass dieser Sachverhalt auch technisch ausgenutzt werden kann.

Der Verbindung zwischen Zeigegesten und Sprache wurde in Zusammenarbeit mit [Holzapfel u. a., 2004] nachgegangen. Dazu wurde ein Dialogsystem untersucht, das Zeigegesten und Sprache zeitbasiert fusioniert und auf gemeinsame *Typed Feature Structures* abbildet. In Experimenten zeigte sich ein Erfolg der multimodalen Fusion: das Dialogsystem konnte 74% der multimodalen Benutzereingaben korrekt erkennen, obwohl die Zeigegestenerkennung allein nur in 44% aller Fälle das korrekte Ziel als wahrscheinlichste Hypothese lieferte.

Die in der vorliegenden Arbeit entwickelten Verfahren zur Personenlokalisierung, Zeigegestenerkennung und zum Hand-Tracking wurden als Softwaremodule implementiert, die auf dem humanoiden Roboter Armar-III eingesetzt werden. Dort arbeiten sie in Echtzeit in engem Verbund mit anderen Komponenten zur akustischen Quellenlokalisierung, Gesichtserkennung, Objekterkennung, Handlungsausführung, Visualisierung und zum natürlichsprachlichen Dialog. Dadurch wurde Armar-III als erster humanoider Roboter in die Lage versetzt, gleichzeitig und on-board mehrere Personen zu lokalisieren, ihre Gesichter zu erkennen und sprachgesteuert zu lernen und Zeigegesten in Verbindung mit sprachlichen Kommandos zu interpretieren. Diese Fähigkeiten wurden seither in zahlreichen Demonstrationen vorgeführt.

Verzeichnis eigener Veröffentlichungen

[Gehrig u. a. 2005] Gehrig, T. ; Nickel, K. ; Ekenel, H.K. ; Klee, U. ; McDonough, J.: Kalman filters for audio-video source localization. In: *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Oct 2005, S. 118–121

[Holzapfel u. a. 2004] Holzapfel, H. ; Nickel, K. ; Stiefelhagen, R.: Implementation and Evaluation of a Constraint Based Multimodal Fusion System for Speech and 3D Pointing Gestures. In: *Proc. of the 6th Intl. Conf. on Multimodal Interfaces (ICMI)*. State College, PA, USA, October 2004, S. 175–182

[Nickel 2003] Nickel, K.: *Erkennung von Zeigegesten basierend auf 3D-Tracking von Kopf und Händen*, Universität Karlsruhe (TH), Diplomarbeit, März 2003

[Nickel 2008] Nickel, K.: *Arthur - Active Real-time Tracking for a Humanoid Robot. Benutzerhandbuch.* `http://isl.ira.uka.de/~nickel/arthur/`. 2008

[Nickel u. a. 2006a] Nickel, K. ; Ekenel, H.K. ; Voit, M. ; Stiefelhagen, R.: Audio-Visual Perception of Humans for a Humanoid Robot. In: *2nd Intl. Workshop on Human-Centered Robotic Systems*. Munich, Germany, October 2006

[Nickel u. a. 2006b] Nickel, K. ; Gehrig, T. ; Ekenel, H. K. ; McDonough, J. ; Stiefelhagen, R.: An Audio-Visual Particle Filter for Speaker Tracking on the CLEAR'06 Evaluation Dataset. In: *Proc. of the CLEAR'06 Evaluation and Workshop*, 2006, S. 69–80

[Nickel u. a. 2005] Nickel, K. ; Gehrig, T. ; Stiefelhagen, R. ; McDonough, J. W.: A joint particle filter for audio-visual speaker tracking. In: *7th Intl. Conf. on Multimodal Interfaces (ICMI)*. Trento, Italy, 2005, S. 61–68

[Nickel u. a. 2004] Nickel, K. ; Seemann, E. ; Stiefelhagen, R.: 3D-tracking of head and hands for pointing gesture recognition in a human-robot interaction scenario. In: *Intl. Conf. on Automatic Face and Gesture Recognition (FG)*, 2004, S. 565–570

[Nickel und Stiefelhagen 2003a] Nickel, K. ; Stiefelhagen, R.: Detection and Tracking of 3D-Pointing Gestures for Human-Robot-Interaction. In: *3rd*

IEEE-RAS Intl. Conf. on Humanoid Robots. Karlsruhe, Germany, October 2003

[Nickel und Stiefelhagen 2003b] Nickel, K. ; Stiefelhagen, R.: Pointing gesture recognition based on 3D-tracking of face, hands and head orientation. In: *5th Intl. Conf. on Multimodal Interfaces (ICMI).* New York : ACM Press, November 2003, S. 140–146

[Nickel und Stiefelhagen 2003c] Nickel, K. ; Stiefelhagen, R.: Real-Time Recognition of 3D-Pointing Gestures for Human-Machine-Interaction. In: *Proc of the 25th DAGM Symposium on Pattern Recognition.* Magdeburg, Germany, 2003, S. 557–565

[Nickel und Stiefelhagen 2004] Nickel, K. ; Stiefelhagen, R.: Real-Time Person Tracking and Pointing Gesture Recognition for Human-Robot Interaction. In: *Computer Vision in Human-Computer Interaction, ECCV Workshop on HCI.* Prague, Czech Republic, 2004, S. 28–38

[Nickel und Stiefelhagen 2007a] Nickel, K. ; Stiefelhagen, R.: Fast Audio-Visual Multi-Person Tracking for a Humanoid Stereo Camera Head. In: *IEEE-RAS Intl. Conf. on Humanoid Robots.* Pittsburgh, USA, 2007

[Nickel und Stiefelhagen 2007b] Nickel, K. ; Stiefelhagen, R.: Visual recognition of pointing gestures for human-robot interaction. In: *Image and Vision Computing* 25 (2007), Nr. 12, S. 1875–1884

[Nickel und Stiefelhagen 2008] Nickel, K. ; Stiefelhagen, R.: Dynamic Integration of Generalized Cues for Person Tracking. In: *European Conf. on Computer Vision (ECCV).* Marseille, Frankreich : Springer-Verlag, LNCS 5305, 2008, S. 514–526

[Seemann u. a. 2004] Seemann, E. ; Nickel, K. ; Stiefelhagen, R.: Head pose estimation using stereo vision for human-robot interaction. In: *Intl. Conf. on Automatic Face and Gesture Recognition (FG)*, 2004, S. 626–631

[Stiefelhagen u. a. 2006] Stiefelhagen, R. ; Bernardin, K. ; Ekenel, H.K. ; McDonough, J. ; Nickel, K. ; Voit, M. ; Woelfel, M.: Audio-Visual Perception of a Lecturer in a Smart Seminar Room. In: *Signal Processing - Special Issue on Multimodal Interfaces* 86 (2006), December, Nr. 12, S. 3518–3533

[Stiefelhagen u. a. 2007] Stiefelhagen, R. ; Ekenel, H. ; Fügen, C. ; Gieselmann, P. ; Holzapfel, H. ; Kraft, F. ; Nickel, K. ; Voit, M. ; Waibel, A.: Enabling Multimodal Human-Robot Interaction for the Karlsruhe Humanoid Robot. In: *IEEE Transactions on Robotics, Special Issue on Human-Robot Interaction* 23 (2007), October, Nr. 5, S. 840–851

[Stiefelhagen u. a. 2004a] Stiefelhagen, R. ; Fuegen, C. ; Gieselmann, P. ; Holzapfel, H. ; Nickel, K. ; Waibel, A.: Natürliche Mensch-Roboter Interaktion mittels Sprache, Blickrichtung und Gestik. In: *Robotik 2004, VDI-Bericht Nr. 1841.* Munich, Germany, June 2004

[Stiefelhagen u. a. 2004b] Stiefelhagen, R. ; Fügen, C. ; Gieselmann, P. ; Holzapfel, H. ; Nickel, K. ; Waibel, A.: Natural Human-Robot Interaction using Speech, Head pose and Gestures. In: *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. Sendai, Japan, September 2004, S. 2422 – 2427

[Voit u. a. 2005a] Voit, M. ; Nickel, K. ; Stiefelhagen, R.: Estimating the Lecturer's Head Pose in Seminar Scenarios - A Multi-view Approach. In: *Proc. Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms (MLMI), Edinburgh*, July 2005, S. 230–240

[Voit u. a. 2005b] Voit, M. ; Nickel, K. ; Stiefelhagen, R.: Multi-View Head Pose Estimation using Neural Networks. In: *Proc. of the 2nd Canadian conference on Computer and Robot Vision (CRV)*, 2005, S. 347–352

[Voit u. a. 2006a] Voit, M. ; Nickel, K. ; Stiefelhagen, R.: A Bayesian Approach for Multi-View Head Pose Estimation. In: *IEEE Intl. Conf. on Multisensor Fusion and Integration for Intelligent Systems*. Heidelberg, Germany, September 2006, S. 31–34

[Voit u. a. 2006b] Voit, M. ; Nickel, K. ; Stiefelhagen, R.: Neural Network-based Head Pose Estimation and Multi-view Fusion. In: *Proc. of the CLEAR'06 Evaluation and Workshop*, 2006, S. 291–298

[Voit u. a. 2007] Voit, M. ; Nickel, K. ; Stiefelhagen, R.: Head Pose Estimation in Single- and Multi-view Environments - Results on the CLEAR'07 Benchmarks. In: *Proc. of the 2nd International CLEAR Evaluation Workshop* Bd. Springer LNCS 4625. Baltimore, USA, May 2007, S. 307–316

[Wojek u. a. 2006] Wojek, C. ; Nickel, K. ; Stiefelhagen, R.: Activity Recognition and Room Level Tracking in an Office Environment. In: *IEEE Intl. Conf. on Multisensor Fusion and Integration for Intelligent Systems*. Heidelberg, Germany, September 2006, S. 25 – 30

[Wölfel u. a. 2005] Wölfel, M. ; Nickel, K. ; McDonough, J.: Microphone array driven speech recognition: influence of localization on the word error rate. In: *Proc. Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms (MLMI), Edinburgh*, July 2005, S. 320–331

[Ziegler u. a. 2006] Ziegler, J. ; Nickel, K. ; Stiefelhagen, R.: Tracking of the Articulated Upper Body on Multi-View Stereo Image Sequences. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. New York, USA, June 2006, S. 774–781

Literaturverzeichnis

[Adam u. a. 2006] Adam, A. ; Rivlin, E. ; Shimshoni, I.: Robust Fragments-based Tracking using the Integral Histogram. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006, S. 798–805

[Asfour u. a. 2006] Asfour, T. ; Regenstein, K. ; Azad, P. ; Schröder, J. ; Bierbaum, A. ; Vahrenkamp, N. ; Dillmann, R.: ARMAR-III: An Integrated Humanoid Platform for Sensory-Motor Control. In: *IEEE-RAS Intl. Conf. on Humanoid Robots*. Genoa, Italy, December 2006, S. 169–175

[Azad u. a. 2006] Azad, P. ; Asfour, T. ; Dillmann, R.: Combining Apperance-based and Model-based Methods for Real-Time Object Recognition and 6D Localization. In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. Beijing, China, 2006, S. 5339–5344

[Ba und Odobez 2004] Ba, S. O. ; Odobez, J.-M.: A probabilistic framework for joint head tracking and pose estimation. In: *Proc. of the 17th Intl. Conf. on Pattern Recognition (ICPR)* Bd. 4, 2004, S. 264–267

[Barber und Legge 1976] Barber, P. ; Legge, D.: *Perception and information, Chapter 4: Information Acquisition*. Methuen, London, 1976

[Bechler u. a. 2004] Bechler, D. ; Schlosser, M.S. ; Kroschel, K.: System for robust 3D speaker tracking using microphone array measurements. In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)* Bd. 3, 2004, S. 2117–2122

[Becker 1997] Becker, D. A.: Sensei: A Real-time Recognition, Feedback, and Training System for T'ai Chi Gestures / Massachusetts Institute of Technology, Media Lab. 1997. – Forschungsbericht

[Bernardin u. a. 2006] Bernardin, K. ; Elbs, A. ; Stiefelhagen, R.: Multiple Object Tracking Performance Metrics and Evaluation in a Smart Room Environment. In: *6th IEEE Intl. Workshop on Visual Surveillance (at ECCV)*. Graz, Austria, May 2006

[Branson und Belongie 2005] Branson, K. ; Belongie, S.: Tracking Multiple Mouse Contours (without Too Many Samples). In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* Bd. 1, 2005, S. 1039–1046

[Brumitt und Cadiz 2000] Brumitt, B. ; Cadiz, JJ: Let There Be Light: Comparing Interfaces for Homes of the Future / Microsoft Research MSR-TR-2000-92. September 2000. – Forschungsbericht

[Campbell u. a. 1996] Campbell, L. W. ; Becker, D. A. ; Azarbayejani, A. ; Bobick, A. F. ; Pentland, A.: Invariant features for 3-D gesture recognition. In: *Intl. Conf. on Automatic Face and Gesture Recognition (FG)*, 1996, S. 157–162

[Carpenter 1992] Carpenter, B.: *The Logic of Typed Feature Structures.* Cambridge University Press., 1992

[Cassell 1998] Cassell, J.: A Framework For Gesture Generation and Interpretation. In: *Computer Vision in Human-Machine Interaction*, Cambridge University Press, 1998, S. 191–215

[Cutkosky 1989] Cutkosky, M. R.: On Grasp Choice, Grasp Models, and the Design of Hands for Manufacturing Tasks. In: *IEEE Transactions on Robotics and Automation* 5 (1989), S. 269–279

[Dalal und Triggs 2005] Dalal, N. ; Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* Bd. 1, 2005, S. 886–893

[Darrell u. a. 2001] Darrell, T. ; Demirdjian, D. ; Checka, N. ; Felzenszwalb, P. F.: Plan-View Trajectory Estimation with Dense Stereo Background Models. In: *Intl. Conference on Computer Vision (ICCV)*, 2001, S. 628–635

[Darrell u. a. 2000] Darrell, T. J. ; Gordon, G. G. ; Harville, M. ; Woodfill, J. I.: Integrated Person Tracking Using Stereo, Color, and Pattern Detection. In: *Intl. Journal of Computer Vision* 37 (2000), Juni, Nr. 2, S. 175–185

[Demirdjian und Darrell 2002] Demirdjian, D. ; Darrell, T.: 3-D Articulated Pose Tracking for Untethered Deictic Reference. In: *Intl. Conf. on Multimodal Interfaces (ICMI)*, 2002, S. 267

[Dilba und Kölling 2007] Dilba, D. ; Kölling, M.: Toyota startet ins Roboter-Zeitalter. In: *Financial Times Deutschland* (2007), 12

[Ekenel u. a. 2007] Ekenel, H.K. ; Stallkamp, J. ; Gao, H. ; Fischer, M. ; Stiefelhagen, R.: Face Recognition for Smart Interactions. In: *IEEE Intl. Conf. on Multimedia and Expo*, July 2007, S. 1007–1010

[Finke u. a. 1997] Finke, M. ; Geutner, P. ; Hild, H. ; Kemp, T. ; Ries, K. ; Westphal., M.: The Karlsruhe-Verbmobil Speech Recognition Engine. In: *Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP).* Munich, Germany, 1997

[Glenstrup und Engell-Nielsen 1995] Glenstrup, A.J. ; Engell-Nielsen, T.: Eye controlled media: Present and future state / University of Copenhagen. 1995. – Forschungsbericht

[Gross u. a. 2006] Gross, H.-M. ; Richarz, J. ; Mueller, S. ; Scheidig, A. ; Martin, C.: Probabilistic Multi-modal People Tracker and Monocular

Pointing Pose Estimator for Visual Instruction of Mobile Robot Assistants. In: *Intl. Joint Conference on Neural Networks (IJCNN)*. Vancouver, Canada, 2006, S. 4209–4217

[Holzapfel 2008] Holzapfel, H.: A Dialogue Manager for Multimodal Human-Robot Interaction and Learning of a Humanoid Robot. In: *Industrial Robots Journal* 35 (2008), October, Nr. 6, S. 528–535

[Holzapfel und Waibel 2008] Holzapfel, H. ; Waibel, A.: Learning and Verification of Names with Multimodal User ID in Dialog. In: *Intl. Conf. on Cognitive Systems (CogSys)*. Karlsruhe, Germany, 2008

[Honda 2008] Honda: *ASIMO - The world's most advanced Humanoid Robot*. `http://asimo.honda.com`. 2008

[Isard und Blake 1998a] Isard, M. ; Blake, A.: Condensation–conditional density propagation for visual tracking. In: *International Journal of Computer Vision* 29 (1998), Nr. 1, S. 5–28

[Isard und Blake 1998b] Isard, M. ; Blake, A.: ICONDENSATION: Unifying Low-Level and High-Level Tracking in a Stochastic Framework. In: *European Conf. on Computer Vision (ECCV)*. Freiburg, Germany, 1998, S. 893–908

[Isard und MacCormick 2001] Isard, M. ; MacCormick, J.: BraMBLe: A Bayesian Multiple-Blob Tracker. In: *Intl. Conference on Computer Vision (ICCV)*, 2001, S. 34–41

[Jojic u. a. 2000] Jojic, N. ; Huang, T. S. ; Brumitt, B. ; Meyers, B. ; Harris, S.: Detection and Estimation of Pointing Gestures in Dense Disparity Maps. In: *Intl. Conf. on Automatic Face and Gesture Recognition (FG)*, 2000, S. 468–475

[Kahn u. a. 1996] Kahn, R. E. ; Swain, M. J. ; Prokopowicz, P. N. ; Firby, R. J.: Gesture recognition using the perseus architecture. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 1996, S. 734–741

[Kaneko u. a. 2004] Kaneko, K. ; Kanehiro, F. ; Kajita, S. ; Hirukawa, H. ; Kawasaki, T. ; Hirata, M. ; Akachi, K. ; Isozumi, T.: Humanoid Robot HRP-2. In: *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2004, S. 1083–1090

[Kendon 1986] Kendon, A. ; Nespolous, J. L. (Hrsg.) ; Perron, P. (Hrsg.) ; Lecours, A. R. (Hrsg.): *Current Issues in the Study of Gesture*. Laurence Erlbaum Associates, Hillsdale, London, 1986

[Kim u. a. 2004] Kim, H. ; Lau, B. ; Triesch, J.: Adaptive Object Tracking with an Anthropomorphic Robot Head. In: *Proc. of the 8th Intl. Conf. on the Simulation of Adaptive Behaviors (SAB'04)*, 13-17 July 2004

[Kittler u. a. 1998] Kittler, J. ; Hatef, M. ; Duin, R. P. W. ; Matas, J.: On Combining Classifiers. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence* 20 (1998), Nr. 3, S. 226–239

[Kölsch und Turk 2004] Kölsch, M. ; Turk, M.: Robust hand detection. In: *Intl. Conf. on Automatic Face and Gesture Recognition (FG)*, 2004, S. 614–619

[Kortenkamp u. a. 1996] Kortenkamp, D. ; Huber, E. ; Bonasso, R. P. ; Inc, Metrica: Recognizing and Interpreting Gestures on a Mobile Robot. In: *In Proceedings of AAAI-96*, AAAI Press/The MIT Press, 1996, S. 915–921

[Kraft u. a. 2005] Kraft, F. ; Malkin, R. ; Schaaf, T. ; Waibel, A.: Temporal ICA for Classification of Acoustic Events in a Kitchen Environment. In: *Proc. of Interspeech*. Lisboa, Portugal, 2005

[Kraft und Wölfel 2007] Kraft, F. ; Wölfel, M.: Humanoid robot noise suppression by particle filters for improved automatic speech recognition accuracy. In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. San Diego, USA, 2007, S. 1737–1742

[Kruppa u. a. 2003] Kruppa, H. ; Castrillon-Santana, M. ; Schiele, B.: Fast and robust face finding via local context. In: *IEEE Intl. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, October 2003

[Lang u. a. 2003] Lang, S. ; Kleinehagenbrock, M. ; Hohenner, S. ; Fritsch, J. ; Fink, G. A. ; Sagerer, G.: Providing the basis for human-robot-interaction: a multi-modal attention system for a mobile robot. In: *5th Intl. Conf. on Multimodal Interfaces (ICMI)*, 2003, S. 28–35

[Lanz 2006] Lanz, O.: Approximate Bayesian Multibody Tracking. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28 (2006), September, Nr. 9, S. 1436–1449

[Leibe u. a. 2005] Leibe, B. ; Seemann, E. ; Schiele, B.: Pedestrian Detection in Crowded Scenes. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* Bd. 1. Washington, DC, USA, 2005, S. 878–885

[Lienhart und Maydt 2002] Lienhart, R. ; Maydt, J.: An Extended Set of Haar-like Features for Rapid Object Detection. In: *Intl. Conf. on Image Processing (ICIP)* Bd. 1, September 2002, S. 900–903

[Littmann u. a. 1996] Littmann, E. ; Drees, A. ; Ritter, H.: Visual Gesture Recognition by a Modular Neural System. In: *Intl. Conf. on Artificial Neural Networks (ICANN)*. Bochum, 1996, S. 317–322

[Maglio u. a. 2000] Maglio, P.P. ; Matlock, T. ; Campbel, C.S. ; Zhai, S. ; ; Smith, B.A.: Gaze and speech in attentive user interfaces, 2000

[McNeill 1992] McNeill, D.: *Hand and Mind: What gestures reveal about thought*. The University of Chicago Press, Chicago IL., 1992

[Miyashita u. a. 2004] Miyashita, T. ; Shiomi, M. ; Ishiguro, H.: Multisensor-based human tracking behaviors with Markov chain Monte Carlo methods. In: *Proc. of the 4th IEEE/RAS Intl. Conf. on Humanoid Robots* Bd. 2, November 2004, S. 794–810

[Murase u. a. 2005] Murase, M. ; Yamamoto, S. ; Valin, J.-M. ; Nakadai, K. ; Yamada, K. ; Komatani, K. ; Ogata, T. ; Okuno, H. G.: Multiple Moving Speaker Tracking by Microphone Array on Mobile Robot. In: *Proc. of Interspeech.* Lisboa, Portugal, 2005

[Nakadai u. a. 2001] Nakadai, K. ; Hidai, K. ; Mizoguchi, H. ; Okuno, H. G. ; Kitano, H.: Real-Time Auditory and Visual Multiple-Object Tracking for Humanoids. In: *Proc. of the 17th Intl. Joint Conf. on Artificial Intelligence IJCAI.* Seattle, USA, 2001, S. 1425–1436

[Nummiaro u. a. 2003] Nummiaro, K. ; Koller-Meier, E. ; Roth, D. ; Gool, L. V.: Color-based object tracking in multi-camera environments. In: *25th German Pattern Recognition Symposium (DAGM)*, 2003, S. 591–599

[Patil u. a. 2004] Patil, R. ; Rybski, P. E. ; Kanade, T. ; Veloso, M. M.: People detection and tracking in high resolution panoramic video mosaic. In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2004, S. 1323–1328

[Pavlovic u. a. 1997] Pavlovic, V. ; Sharma, R. ; Huang, T. S.: Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19 (1997), Nr. 7, S. 677–695

[Pentland u. a. 1996] Pentland, A. P. ; Darrell, T. J. ; Azarbayejani, A. ; Wren, C. R.: Pfinder: Real-Time Tracking of the Human Body. In: *Intl. Conf. on Automatic Face and Gesture Recognition (FG)*, 1996, S. 51–56

[Pérez u. a. 2004] Pérez, P. ; Vermaak, J. ; Blake, A.: Data fusion for visual tracking with particles. In: *Proceedings of the IEEE* 92 (2004), March, Nr. 3, S. 495–513

[Porikli 2005] Porikli, F.: Integral Histogram: A Fast Way to Extract Histograms in Cartesian Spaces. In: *in Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2005, S. 829–836

[Rabiner 1989] Rabiner, L. R.: A tutorial on hidden Markov models and selected applications in speech recognition. In: *Proceedings of the IEEE* 77 (1989), S. 257–286

[Satoh u. a. 2004] Satoh, Y. ; Okatani, T. ; Deguchi, K.: A Color-based Probabilistic Tracking by Using Graphical Models. In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS).* Sendai, Japan, 2004

[Scharstein und Szeliski 2002] Scharstein, D. ; Szeliski, R.: A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. In:

International Journal of Computer Vision (IJCV) 47 (2002), April-June, Nr. 1/2/3, S. 7–42

[Schulz u. a. 2003] Schulz, D. ; Burgard, W. ; Fox, D. ; Cremens, A.: People tracking with mobile robots using sample-based joint probabilistic data association filters. In: *International Journal of Robotics Research (IJRR)* 22 (2003), Nr. 2, S. 99–116

[Seemann 2003] Seemann, E.: *Estimating Head Orientation with Stereo Vision*, Universität Karlsruhe (TH), Diplomarbeit, 2003

[SFB 588] SFB 588: *Humanoide Roboter - Lernende und kooperierende multimodale Roboter, Webseite.* `http://www.sfb588.uni-karlsruhe.de`

[Shen u. a. 2003] Shen, C. ; Hengel, A.v.d. ; Dick, A.: Probabilistic Multiple Cue Integration for Particle Filter Based Tracking. In: *Intl. Conf. on Digital Image Computing - Techniques and Applications*, 2003, S. 309–408

[Soltau u. a. 2001] Soltau, H. ; Metze, F. ; Fuegen, C. ; Waibel, A.: A one-pass decoder based on polymorphic linguistic context assignment. In: *Proc. of the Automatic Speech Recognition and Understanding Workshop (ASRU).* Trento, Italy, 2001

[Spengler und Schiele 2003] Spengler, M. ; Schiele, B.: Towards robust multi-cue integration for visual tracking. In: *Machine Vision and Applications* 14 (2003), S. 50–58

[Starner u. a. 1998] Starner, T. ; Weaver, J. ; Pentland, A.: Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20 (1998), Nr. 12, S. 1371–1375

[Stauffer und Grimson 2000] Stauffer, C. ; Grimson, W. E. L.: Learning Patterns of Activity Using Real-Time Tracking. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22 (2000), Nr. 8, S. 747–757

[Stiefelhagen u. a. 2007] Stiefelhagen, R. (Hrsg.) ; Bowers, R. (Hrsg.) ; Fiscus, J. (Hrsg.): *Multimodal Technologies for Perception of Humans, Joint Proc. of the 2nd Intl. Evaluation workshop on Classification of Events, Activities and Relationships, CLEAR 2007 and the Spring 2007 Rich Transcription Meeting Evaluation.* Bd. *4625.* Springer, 2007. (Lecture Notes in Computer Science)

[Stiefelhagen u. a. 2000] Stiefelhagen, R. ; Yang, J. ; Waibel, A.: Simultaneous Tracking of Head Poses in a Panoramic View. In: *Intl. Conf. on Pattern Recognition (ICPR)* Bd. 3, September 2000, S. 726–733

[Tao u. a. 1999] Tao, H. ; Sawhney, H. S. ; Kumar, R.: A Sampling Algorithm for Tracking Multiple Objects. In: *Workshop on Vision Algorithms*, 1999, S. 53–68

[Triesch und Malsburg 2001] Triesch, J. ; Malsburg, C.v.d.: Democratic Integration: Self-Organized Integration of Adaptive Cues. In: *Neural Computing* 13 (2001), Nr. 9, S. 2049–2074

[Veksler 2003] Veksler, O.: Fast variable window for stereo correspondence using integral images. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2003, S. 556–561

[Viola und Jones 2001] Viola, P. ; Jones, M.: Robust real-time object detection. In: *ICCV Workshop on Statistical and Computation Theories of Vision*, July 2001

[Waldherr u. a. 2000] Waldherr, S. ; Romero, R. ; Thrun, S.: A gesture based interface for human-robot interaction. In: *Autonomous Robots* 9 (2000), S. 151–173

[Wilhelm u. a. 2004] Wilhelm, T. ; Böhme, H.-J. ; Gross, H.-M.: A Multi-Modal System for Tracking and Analyzing Faces on a Mobile Robot. In: *Robotics and Autonomous Systems* 48 (2004), August, Nr. 1, S. 31–40

[Wilson und Bobick 1998] Wilson, A. D. ; Bobick, A. F.: Recognition and Interpretation of Parametric Gesture. In: *Intl. Conference on Computer Vision (ICCV)*, 1998, S. 329–336

[Wu und Huang 1999] Wu, Y. ; Huang, T. S.: Vision-Based Gesture Recognition: A Review. In: *Gesture-Based Communication in Human-Computer Interaction, International Gesture Workshop, GW*. Gif-sur-Yvette, France, 1999, S. 103–115

[Wu und Huang 2001] Wu, Y. ; Huang, T. S.: A Co-inference Approach to Robust Visual Tracking. In: *Intl. Conference on Computer Vision (ICCV)*, 2001, S. 26–33

[Yang u. a. 2005] Yang, C. J. ; Duraiswami, R. ; Davis, L. S.: Fast Multiple Object Tracking via a Hierarchical Particle Filter. In: *Intl. Conf. on Computer Vision*, 2005, S. 212–219

[Yang u. a. 1997] Yang, J. ; Lu, W. ; ; Waibel., A.: Skin-color modeling and adaption. 1997. – Forschungsbericht

[Yarbus 1967] Yarbus, A.L. ; Riggs, L. A. (Hrsg.): *Eye Movement and Vision*. Plenum Press, New York, 1967. – 171–196 S