Inkrementelles und interaktives Lernen von Handlungswissen für Haushaltsroboter

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

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

genehmigte

Dissertation

von

Michael Pardowitz

aus Bamberg

Tag der mündlichen Prüfung:	11.7.2007
Erster Gutachter:	Prof. Dr.-Ing. R. Dillmann
Zweiter Gutachter:	Prof. Dr.-Ing. A. Knoll

Danksagungen

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Technische Informatik an der Universität Fridericiana zu Karlsruhe (TH). Am Lehrstuhl „Industrielle Anwendungen der Informatik und Mikrosystemtechnik“ unter der Leitung von Prof. Dr.-Ing. R. Dillmann hatte ich die Möglichkeit, eingebunden in die deutsche, europäische und internationale Wissenschaftslandschaft zu den Themen künstlicher Intelligenz und maschinellem Lernen in der Robotik zu forschen.

Meinem Doktorvater Prof. Dr.-Ing. R. Dillmann möchte ich ganz herzlich für die tatkräftige Unterstützung sowohl durch die Bereitstellung eines großartigen Umfeldes als auch durch Hilfestellungen und ideelle Impulse danken. Sein stetes Interesse an dem Thema sowie die vielfältigen Vorschläge und Hinweise haben der Entwicklung dieser Arbeit steten Antrieb gegeben. Besonders möchte ich mich bei ihm für die zeitnahe und konstruktive Korrektur meiner Arbeit bedanken. Ebenso bin ich ihm Dank für die abwechslungsreichen Betätigungsfelder, das hohe in mich gesetzte Vertrauen und die nötigen Herausforderungen, die die Lehrstuhl- und Projektarbeit mit sich brachten, schuldig.

Ebenfalls sehr herzlich möchte ich mich bei Prof. Dr.-Ing. A. Knoll für die Übernahme des Korreferats bedanken, die sehr zügige Korrektur und die interessanten und beruhigenden Diskussionen vor der mündlichen Prüfung.

Ein besonderer Dank gilt natürlich allen Kolleginnen und Kollegen vom IAIM, die mich, auf die eine oder andere Art und Weise, bei der Erstellung dieser Arbeit unterstützt haben. Leider reicht hier der Platz nur, um einige zu erwähnen.

Ohne die Organisation, Information und nicht zuletzt Fürsorge unserer lieben Damen im Chefvorzimmer wäre diese Arbeit nicht möglich gewesen. Der Wert ihrer Hilfe bei der Bewältigung kleiner Probleme, mittlerer Unglücke und größerer Katastrophen ist unschätzbar. Ich danke insbesondere Isabelle Wappler für ihre nette und zutreffende Kritik an meinen Vorträgen, die aufmunternden Worte an der Kaffeemaschine und die Empathie bei leerem Süßigkeitenkorb kurz vor wichtigen Deadlines, Nela Redzovic für das

beste Gute-Nachricht-schlechte-Nachricht-Spiel als Abschluß der mühsamen Suche nach einem Prüfungstermin und Christine Brand („Wer hat eigentlich beim EURON-Treffen Cocktails bestellt?“) für die großzügige Berücksichtigung meiner ästhetischen Präferenzen hinsichtlich Stewardessenuniformen bei der Buchung von Flügen. Ach und – darf ich den Locher mitnehmen?

Ich danke Dr.-Ing. Peter Steinhaus und Dr.-Ing. Markus Ehrenmann für die Betreuung noch zu Studentenzeiten und die ersten Gelegenheiten, mich zu beweisen. Meinem Vorgänger und Mentor in den ersten Monaten, Dr.-Ing. Raoul Zöllner, möchte ich besonders für die richtungsweisenden Hilfestellungen zu Beginn meiner Arbeit, verbunden mit einem gehörigen Schuß Skeptik und Erduldung meines anfänglichen Elans, danken, für sein kreatives Chaos und für die lange Leine als mein Gruppenleiter. Schade, daß Du nicht länger geblieben bist!

Meine ehemaligen Bürokollegen, Dr.-Ing. Steffen Knoop und Stefan Vacek („Bis wann? Reicht vorgestern?“), haben immensen Anteil und Einfluß an bzw. auf die Entwicklung der in dieser Arbeit vorgestellten Inhalte gehabt. Ich danke Ihnen für all die Anregungen, den richtigen Schuß an Idealismus und Pragmatismus zum richtigen Zeitpunkt, die kollegiale Atmosphäre und die gemeinsame Zeit in 310. Ein Tip noch für die Zukunft: Vergeßt C++! Lernt Lisp! Selbiges gilt auch für meine jüngeren Kollegen Martin Lösch und Sven Schmidt-Rohr, die meine letzten Tage in 310 mit mir teilen und sich hoffentlich in Zukunft statt mir um die Zimmerpflanzen kümmern.

Bei meinen Studenten Bernhard Glaser, Sebastian Wiedemann, Max-Gerd Retzlaff, Matthias Rambow und Ulrike Balzert möchte ich mich für ihre Unterstützung und die gute Zusammenarbeit bedanken. Respekt! Viel Glück und Erfolg weiterhin!

Herzlich möchte ich allen Freunden für die zwischenmenschlichen Kontakte und die Erholungen vom Arbeiten danken, den Normannen, den Göttingern, und allen anderen, die mich in den letzten Jahren begleitet haben.

Meinen Eltern, Großeltern und Geschwistern gilt besonderer Dank, daß sie mir alles ermöglichten und zu jeder Zeit hinter mir standen, wenn ich ihre Unterstützung und ihren Zuspruch brauchte.

Und mir fehlen die Worte, Anne zu danken.

Karlsruhe, Juli 2007 *Michael Pardowitz*

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

Seit dem Ende der fünfziger Jahre des 20. Jahrhunderts werden im industriellen Umfeld in zunehmenden Maße Roboter zur Produktion und Manipulation von Waren und Gütern eingesetzt. Besonders in strukturierten Umgebungen werden Roboter mit großem Erfolg zur Erledigung einfacher, repetitiver Aufgaben herangezogen. Neben der Lösung der mechanischen, kinematischen und dynamischen Probleme hat sich die Programmierung der hochkomplexen Robotiksysteme als große Herausforderung erwiesen. Dies liegt zum einen an der sich zunehmend schwieriger gestaltenden Integration komplexer Sensorik und Aktorik zu einem Gesamtsystem. Es soll stabil, robust und verlässlich agieren und dabei nur geringstmöglichen Einschränkungen bezüglich der Flexibilität der zu erledigenden Aufgaben sowie deren effizienten Ausführung unterworfen sein. Andererseits zwingen die Veränderungen, der die Welt der industriellen Produktion in Zeiten der Globalisierung und zunehmenden Automatisierung unterworfen ist, zu immer kürzeren Produktionszyklen bei erhöhter Komplexität und Diversität der herzustellenden Produkte bzw. der zu verarbeitenden Waren und Gütern. Es sind immer kürzere Zeiträume für die Entwicklung von Steuersoftware für Roboter verfügbar. Zugleich wird ein stetig wachsendem Umfang an Funktionalität gefordert, gepaart mit der Notwendigkeit der Einhaltung hoher Standards in Bezug auf Sicherheit, Robustheit, Verfügbar- und Verlässlichkeit. Dieses Paradoxon führt dazu, dass Methoden zur schnellen und automatisierten Programmentwicklung immer stärker in den Mittelpunkt des Interesses rücken. Auf verschiedenen Abstraktions- und Komplexitätsebenen konnten erste Systeme zum Rapid-Prototyping und/oder der teilautomatisierten Programmierung im industriellen Umfeld implementiert und etabliert werden.

In den letzten Jahren hat sich der Fokus der Robotikforschung mehr und mehr von der industriellen Robotik weg hin zu den zukunftsweisenden Service- und humanoiden Robotern bewegt. Damit sind mobile Roboter ge-

meint, welche in unmittelbarer Nähe des Menschen wie z.B. in einer häuslichen Umgebung eingesetzt werden, um den Menschen bei der Erledigung alltäglicher Routineaufgaben zu unterstützen und zu assistieren. Aus diesem neuen Anwendungsfeld ergeben sich zusätzliche Anforderungen gegenüber herkömmlichen Robotern sowie neue Herausforderungen:

- Der Grad der Strukturiertheit in häuslichen Umgebungen ist deutlich geringer als im industriellen Umfeld. Gründe hierfür liegen in der Tatsache, dass im Wohnumfeld viel mehr Wert auf Ergonomie in Bezug auf den Menschen gelegt werden muss. Das ist in Produktionsbetrieben nicht der Fall, da hier der Fokus eher auf Funktionalität liegt. Eine starke Strukturierung wirkt jedoch vereinfachend auf die perzeptiven Fähigkeiten, die ein System aufweisen muss, um in seinem Umfeld bestehen und zielgerichtet agieren zu können. Daraus ergibt sich, dass höherer Aufwand bei der verwendeten Sensorik sowie der Verarbeitung der gewonnen Datenmengen erforderlich ist, um ein Robotersystem mit den Grundfunkionalitäten Navigation, Interaktion und Wahrnehmung der Umwelt auszustatten.

- Die beabsichtigte Kolaboration mit Menschen in *einem* gemeinsamen Arbeitsraum macht quantitativ aber auch qualitativ neue, komplexere Sicherheitsanforderungen erforderlich, denen ein Serviceroboter genügen muß. Diese können oft nur mit Hilfe von dedizierter Sensorik, hohem Verarbeitungsaufwand und ausgefeilten Systemarchitekturen erreicht werden.

- Serviceroboter im Haushaltsbereich werden für einen sehr weiten Anwendungsbereich konzipiert, da jeder Anwender unterschiedliche Erwartungen und Bedürfnisse hat und mit einer individuell gestalteten Wohnumgebung aufwartet. Letztere Unterschiede in der Umwelt des Robotersystems können nur zum Teil durch erhöhten Aufwand auf der Sensorik-Seite aufgefangen werden; die unterschiedlichen Anforderungen, Erwartungen und Aufgaben, die an den Roboter herangetragen werden, können nur durch einen sehr hohen Umfang an Funktionalität erfüllt werden. Es stellte sich heraus, dass dieser Umfang zu gross ist, um effizient implementiert zu werden und dennoch einen Haushalts-Serviceroboter zu erschwinglichen, d.h. auf den Kundenkreis des Anwenders zugeschnittenen Preisen anzubieten. Statt dessen erscheint es als gangbare Alternative, nach der Implementierung einer standardisierten Menge an Grundfunktionalitäten auf dem Roboter den Benutzer selbst die Programmierung der von ihm benötigten Funktionalität durchführen zu lassen. Dies hat neben der Reduzierung des vom

Anbieter zu implementierenden Funktionsumfangs zur Folge, daß die Programmierung zielorientierter, d.h. gut abgestimmt auf die genauen Bedürfnisse des Anwenders erfolgt. Weiterhin kann das Vorhalten der Funktionalität deutlich speichereffizienter erfolgen. Es wird nur ein bestimmter, die Bedürfnisse des Nutzers widerspiegelnder Umfang an Funktionalität im Speicher abgelegt, der dann aber auch verfügbar sein muss. Die Programmierung durch den Anwender stellt jedoch hohe Anforderungen an die Programmierschnittstelle. Im Gegensatz zu den Experten, die mit der Programmierung von Industrierobotern betraut sind, ist der Anwender von Service- und Haushaltsrobotern in der Regel nicht mit dem Problemfeld vertraut, in dem sich die Programmierung von Robotern bewegt. Kenntnisse über die Kinematik des Roboters, seine Dynamik und seine Sensorik sowie seine Perzeptionsfähigkeit können nicht vorausgesetzt werden; auch die Beherrschung der Programmierung von Rechnern im allgemeinen gehört nicht zum Kanon des Allgemeinwissens und sollte daher auch nicht als Ausgangsbasis dienen. Verfahren, wie das Teach-In von Verfahrpunkten, wie sie in der Industrierobotik eingesetzt werden, stellen zwar deutliche Vereinfachungen in der Roboterprogrammierung dar, sind aber immer noch für unerfahrene Programmierer sehr ermüdend, unintuitiv und langwierig. Stattdessen sind neue Möglichkeiten zur Programmierung von Robotern in Betracht zu ziehen.

Eine Herangehensweise an das letztere Problem stellt das *Programmieren durch Vormachen (PdV)* dar. Die Idee dahinter ist, den Benutzer die auszuführende Tätigkeit vormachen und sensorisch aufzeichnen zu lassen, und zwar derartig, daß sie anschliessend vom Roboter ausgeführt werden kann. Dabei kommt es darauf an, dass der Roboter die wesentlichen Eigenschaften einer Problemstellung sowie deren Lösung erfaßt, interpretiert und selbstständig lernt, diese unter ähnlichen Ausführungsbedingungen anzuwenden. Hierzu ist das Robotersystem mit der Fähigkeit zur Selbstorganisation und Intelligenz auszustatten. Im folgenden Abschnitt werden einige generelle Überlegungen zum erforderlichen und möglichen Grad an Intelligenz in technischen Systemen angestellt.

1.1 Intelligenz in Robotersystemen

Um darüber reden zu können, wie und inwieweit Robotersysteme mit Intelligenz ausgestattet werden können, ist es notwendig, den Begriff der Intelligenz näher zu beleuchten. Über die Frage, wie Intelligenz genau zu definieren ist, gibt es soviele Mutmaßungen wie intelligente Wesen. Selbst im Wissens- und

Forschungsgebiet der kognitiven Psychologie konnte man sich in den über hundert Jahren, seit dieses Gebiet existiert, nicht auf eine umfassende Definition einigen (siehe [Funke und Vaterrodt-Plünnecke 2004] für eine detailliertere Diskussion). Auch ohne eine feste Definition gibt es Maße für die Intelligenz eines Systems, die zwar weithin anerkannt, jedoch auch nicht völlig unumstritten sind [Guthke 1980]. Eine sehr wesentliche Kontroverse geht weiterhin von der Frage aus, ob Intelligenz erlernt oder angeboren ist [Riemann und Fleuret 2005]. Im Rahmen dieser Arbeit soll kein Versuch unternommen werden, Intelligenz zu definieren. Stattdessen wird von einigen Postulaten ausgegangen, die notwendige, aber nicht unbedingt hinreichende Bedingungen für Intelligenz darstellen.

Einen für diese Arbeit zentralen Aspekt von Intelligenz hat George Bernhard Shaw herausgestellt:

> *„Der Nachteil der Intelligenz besteht darin, daß man ständig gezwungen ist dazuzulernen.“*
>
> George Bernhard Shaw, 1856-1950

Man kann sicherlich geteilter Meinung darüber sein, ob die angesprochene Eigenschaft von Vorteil oder von Nachteil ist. Wichtig ist jedoch, daß Lernen und kontinuierliche Weiterentwicklung, neben anderem, ein wesentlicher Aspekt von Intelligenz ist.

Menschliches Lernen hat die Eigenschaft, daß es ein niemals abgeschlossener Prozeß ist und stets mit dem Auftreten neuer Erfahrungen rechnen muß. Diesen inkrementellen Prozeß hat zuerst der Lern- und Kognitionspsychologe Jean Piaget beim Menschen beobachtet und protokolliert. Er postulierte ständig fortlaufende Akkomodations- und Assimiliationsprozesse eines Individuums an seine Umwelt als Grundfunktionalitäten des menschlichen Lernens [Ginsburg und Opper 1969, Flavell 1963].

Ausgehend von diesen Feststellungen, stellt sich die Frage, ob Robotersysteme zu solchen kontinuierlichen, inkrementellen Lernprozessen fähig sind. Ein Überblick über die verfügbare Literatur und vorhandene Systeme (siehe Abschnitt 2) ergab, daß zwar eine Vielzahl von Lernalgorithmen und lernenden Systemen verfügbar ist, diese jedoch Lernen nur als zeitlich begrenzten, punktuellen und schnell beendeten Vorgang ansehen, auf den eine zeitlich deutlich länger ausgedehnte und vom Lernschritt streng abgenabelte Anwendungsphase folgt.

In dieser Arbeit soll ein Ansatz vorgestellt werden, welcher diese künstliche und für wirklich intelligente Systeme eher untypische Trennung aufhebt. Er soll die Verschränkung von Lern- und Ausführungsprozessen ermöglichen

sowie die inkrementellen Lernprozesse des Menschen nachbilden. Die Hoffnung ist, auf diese Weise Systeme realisieren zu können, welche intelligentes Verhalten wenigstens in einigen Aspekten realisieren. Die Probleme, die beim Entwurf und der Implementierung eines solches System auftreten, ist der folgende Abschnitt gewidmet.

1.2 Problemstellung

Es ist ein System zu definieren und zu implementieren, welches

- menschliche Demonstrationen von Handlungen und Aufgabenlösungen observiert, analysiert und in eine für den Roboter verständliche und ausführbare Weise aufbereitet,

- sein Wissen über unterschiedliche Aufgaben kontinuierlich mit steigendem Erfahrungsschatz erweitert, indem es autonom Schlüsse aus der vorhandenen Datengrundlage zieht und diese in das Gesamtwissen geeignet integriert.

Folgende Anforderungen und Randbedingungen ergeben sich daraus an ein System, das inkrementell und interaktiv aus mehreren Benutzerdemonstrationen lernt:

- Das System kann die Vorteile datengetriebender und wissensbasierter Verfahren miteinander kombinieren: Es ist möglich, aus einem einzigen Lernbeispiel mit einer bestimmten Verlässlichkeit eine semantisch korrekte Aufgabenbeschreibung zu lernen. Andererseits kann das Wissen über eine Aufgabenlösung verbessert, erweitert und verlässlicher gemacht werden, falls der Benutzer noch weitere Demonstrationen derselben Aufgabe vorführt.

- Weiterhin ist es möglich, durch das Zulassen einer mehrelementigen Trainingsdatenmenge für eine Task die Generalisierung von Aufgabenbeschreibungen zu optimieren. Bei einer einzelnen Benutzervorführung kann diese lediglich regelbasiert verallgemeinert werden, z.B. über Objektklassen. Sind jedoch zwei oder mehrere Vorführungen einer Task vorhanden, so kann festgestellt werden, welche Merkmale der Aufgabenlösung hohe Varianzen besitzen. Über solche Merkmale scheint eine weitere Generalisierung sinnvoll als über andere Eigenschaften, die eine deutlich geringere Varianz aufweisen. Es liegt nahe, dass eine solche

zielgerichtete Generalisierung zu besser angepassten Aufgabenbeschreibungen führt, die eine maximale Generalisierung entlang der irrelevanten Merkmale aufweisen, die relevanten Merkmale jedoch nicht übergeneralisiert.

- Die Möglichkeit, bereits aus einer einzigen Benutzervorführung ein ausführbares Programm zu generieren, schafft die Grundlage dazu, erst teilweise gelernte Aufgaben auszuführen und dann direkt vom Benutzer überprüfen zu lassen. So kann über natürliche Interaktionsmechanismen zu einem sehr frühen Zeitpunkt dem Benutzer eine Rückmeldung über die gelernten Merkmale der betrachteten Task gegeben werden und diese mit Kommentaren und Verbesserungen zu versehen. Alternativ kann auf dieser Basis der Benutzer entscheiden, eine vollständig neue Demonstration derselben Task durchzuführen.

- Durch die Hinzunahme zusätzlicher Vorführungen wird es dem System möglich, Handlungsalternativen zu erkennen, geeignet zu repräsentieren und einer Ausführungskomponente zur Verfügung zu stellen, so dass diese zur Ausführungszeit sich für eine Alternative entscheiden kann, die unterschiedlichen Optimialitätskriterien genügt, abhängig vom Ausführungskontext. Alternativen können unter anderem in der Reihenfolge der Aktionen, der bedingten Ausführung bestimmter Aktionen oder der Verwendung unterschiedlicher Verfahrtrajektorien zur Lösung derselben Aufgabe bestehen.

- Eine Identifikation gleicher oder ähnlicher Benutzervorführungen sollte durchgeführt werden können, um entscheiden zu können, ob eine bisher unbekannte Task vorgeführt wird oder ob die Vorführung genutzt werden kann, um das Wissen über eine bereits bekannte Task zu erweitern. Im letzten Fall muss auch die betreffende, bereits vorhandene Taskbeschreibung in einer Datenbasis von bisher gelernten Tasks gefunden und referenziert werden.

- Das System sollte in der Lage sein, auch auf einem spärlichen Bestand an Trainingsdaten Generalisierungen zu erstellen. Da dieser jedoch naturgemäss mit einer grossen Unsicherheit behaftet sein wird, muss die Möglichkeit bestehen, einmal gemachte Generalisierungen zu verwerfen und durch andere zu ersetzen, die von neuen Benutzerdemonstrationen nahegelegt werden. Dies ermöglicht es, die Vorteile sowohl wissensbasierter als auch datenbasierter Methoden zu nutzen.

- Zu einem frühen Zeitpunkt, bei dem nur eine oder sehr wenige Demonstrationen einer Task vorliegen, müssen unterschiedliche Hypothesen er-

stellt und parallel verfolgt werden, die sich mit zusätzlichen Vorführungen verhärten können oder verworfen werden müssen.

- Das Lernen sollte parallel zur Demonstration erfolgen und keine offline-Schritte benötigen, um eine zeitnahe Rückmeldung an den Benutzer und damit Interaktion überhaupt zu ermöglichen.

- Es sollte die Möglichkeit bestehen, Robotern zusätzliche Informationen z.B. durch akustische Kommunikationskanäle zur Verfügung zu stellen, die dieser verwenden kann, um das gelernte Aufgabenwissen zu verfeinern. In das System sollte also eine Spracherkennungs- und Verarbeitungskomponente sowie ein Dialogprozessor integriert sein.

1.3 Beitrag dieser Arbeit

Der Beitrag dieser Arbeit aus Sicht der Informatik zum Fortschritt des Arbeitsfeldes des Programmierens durch Vormachen besteht aus folgenden Kernpunkten:

- Einer systematischen Modellierung von Handlungswissen mittels grammatikalischer Beschreibungsverfahren, wie sie bereits seit Jahrzehnten in anderen Gebieten der Informatik, wie dem Übersetzerbau mit Erfolg eingesetzt werden. Mittels des Konzeptes von Detektorenketten wurden die Vorteile formalisierter Beschreibungsmöglichkeiten von hierarchische Strukturen für die Repräsentation und Klassifikation von Handlungen nutzbar gemacht. Die Handlungsabstraktion profitiert besonders wegen der Erweiterbarkeit, Modularität und Aussagekraft von der Beschreibung von Handlungswissen, wie sie in Kapitel 4 vorgeschlagen wird.

- Einer konsequenten Übertragung des kognitiven Entwicklungsmodells nach Piaget. Diese erlaubt es zukünftigen Haushaltsrobotern lebenslang autonom zu lernen. Im Detail wird es möglich, neues Handlungswissen in eine strukturierte Handlungsdatenbasis einzufügen und Relationen zwischen neu erlernetem Handlunswissen und bereits vorhandenem herzustellen, also einen Abgleich neuer Erfahrung mit Erinnertem zu leisten (siehe Kapitel 5).

- Methoden, wie Handlungswissen mittels Schlußfolgerungsverfahren erweitert und vervollständigt werden kann. Hierzu werden in Abschnitt 6 Verfahren vorgestellt, welche unterschiedliche Wissensinhalte normalisieren und fusionieren, mit dem Ziel, aus einzelnen Observationen nicht

wahrnehmbares Wissen zu gewinnen. Dies im Rahmen dieser Arbeit anhand der Erschließung von sequentieller Präzedenz von Aufgaben und Handlungen, sowie der Erkennung repetitiver Strukturen aus Benutzerdemonstrationen demonstriert.

1.4 Überblick über diese Arbeit

Im folgenden Kapitel 2 wird ein Überblick über verschiedene im Einsatz befindliche PdV-Systeme gegeben und deren jeweiligen Einschränkungen, Vor- und Nachteile erörtert. Hierbei wird auch detaillierter als in dieser Einführung auf die psychologischen Ansätze eingegangen, welche Lernprozesse beschreiben und erklären. In Kapitel 3 werden die Komponenten des im Rahmen dieser Arbeit erstellten Systems eingeführt sowie ihre Zusammenhänge erörtert. Die Kapitel 4 bis 6 stellen die wesentlichen Phasen des hier gewählten Ansatzes vor, namentlich die lexikalischen, syntaktischen, semantischen und schlussfolgernden Analyseschritte. Kapitel 7 beschreibt die mit dem erstellten System durchgeführten Experimente sowie deren Ergebnisse. Abschließend werden in Kapitel 8 die erzielten Ergebnisse zusammengefasst sowie denkbare zukünftige Forschungsrichtungen und Weiterentwicklungen aufgezeigt.

Kapitel 2

Lernen von Handlungswissen - ein Überblick

Ein Blick in aktuelle Konferenz- und Zeitschriftenbände der Robotik läßt erahnen, welch eine Vielfalt an Systemen zum Lernen von Handlungswissen heute existieren. Einen Überblick über die wichtigsten Arbeiten auf diesem Gebiet liefern [Schaal 1999, Dillmann et al. 1999, Billard und Siegwart 2003, Saunders et al. 2004]. Dabei wird bereits deutlich, daß Systeme zum Erlernen von Handlungswissen in erster Näherung von vier unterschiedlichen Faktoren charakterisiert werden, deren Ausprägungen die Einsatzfähigkeit und Mächtigkeit des Systems determinieren (siehe Abbildung 2.1):

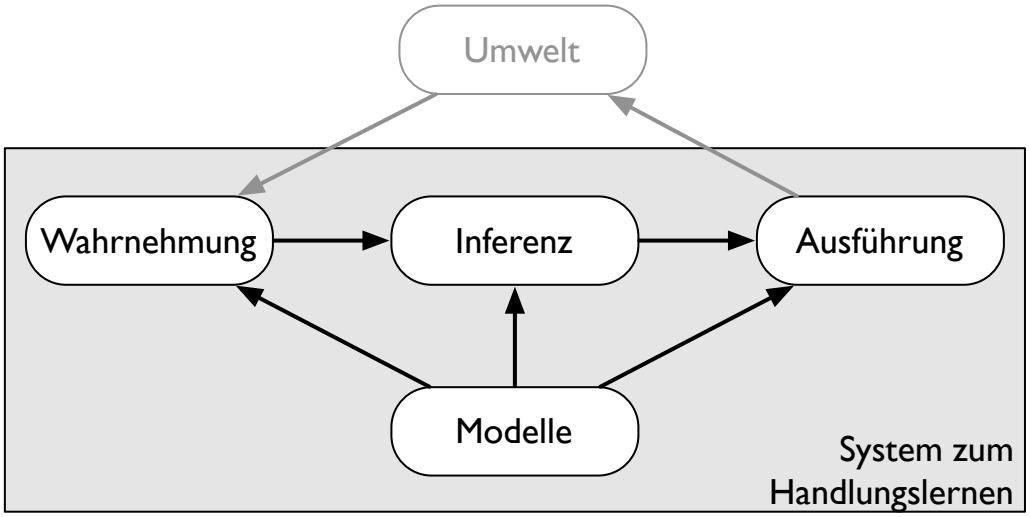


Abbildung 2.1: Schlüsselmodule eines Systems zum Lernen von Handlungswissen aus menschlichen Demonstrationen

- Die *Wahrnehmung* der Benutzerdemonstration ist der Eintrittspunkt für neue Erfahrungen aus der Umwelt in das System. Die Schnittstelle bilden unterschiedliche Sensorsysteme. Deren Daten müssen meist vorbearbeitet, gefiltert, ergänzt oder geglättet werden, bevor diese mittels

- *Inferenz* weiterverarbeitet werden. Die Inferenz ist die zentrale Komponente eines Systems zum Lernen von Handlungswissen. Hier werden die einströmenden Daten mittels Verfahren des Maschinellen Lernens in Handlungswissen, das vom Roboter genutzt werden kann, umgewandelt. Erwartungsgemäß findet sich hier auch eine besonders weite Varianz, die das gesamte Spektrum der Disziplin Maschinelles Lernen abdeckt (siehe [Mitchell 1997]). Das erlernte Handlungswissen wird anschließend durch

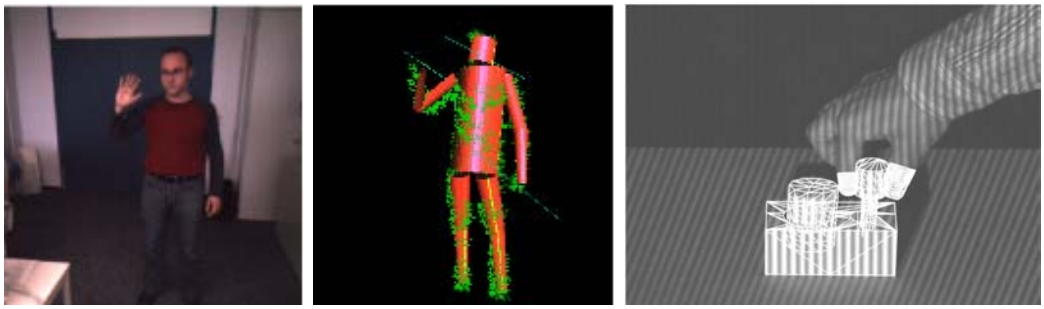
- *Ausführung in der Domäne* angewandt. Dies kann entweder mit einem realen Roboter direkt in der Domäne oder mittels Simulation in der Virtuellen Realität geschehen. Alle diese Phasen des Lernprozesses werden unterstützt und teilweise erst ermöglicht durch verschiedene

- *Modelle*. Diese können geometrische, relationale, hierarchische, semantische Repräsentationen der Umwelt, des Menschen und der Domäne umfassen. Sie dienen als Hintergrundwissen für alle anderen Prozesskomponenten.

In den folgenden vier Abschnitten werden die in der Literatur vorgefundenen Arbeiten zu lernenden Robotersystemen vor dem Hintergrund dieser Einteilung näher beleuchtet, analysiert und verglichen. Zuletzt folgt in Abschnitt 2.5 noch ein Überblick über den Standpunkt der Lern- und Entwicklungspsychologie unter dem Aspekt des Lernen von Handlungswissens.

2.1 Wahrnehmung

Die Wahrnehmung des Benutzers, seiner Handlungen und Absichten sowie seiner Umgebung stellt den Ausgangspunkt eines jeden lernenden Robotersystems dar. Dies geschieht mittels verschiedener sensorischer Beobachtungssysteme, die sich nicht nur hinsichtlich Genauigkeit, Verläßlichkeit und Aussagekraft unterscheiden. Ein weiterer wesentlicher Aspekt ist der Grad der Einschränkung und Behinderung des Menschen selber durch die Sensorik, also den Einfluß, den die Sensorik auf das Wahrzunehmende, den Menschen und seine Handlungen, ausübt. Generell gilt, daß eine höhere Genauigkeit und Aussagekraft mit größerer Einflußnahme auf den Menschen einhergeht. Das



(a) (b) (c)

Abbildung 2.2: Markerlose Erfassung des Menschen und seiner Umwelt. (a) Kamerabild. (b) ICP-basiertes Tracking menschlicher Bewegung. (c) Rekonstruktion von Objektlagen und Fingerposition unter Verwendung von Streifenprojektion. Quellen: (a, b) [Knoop et al. 2006] (c), [Jiar et al. 1996].

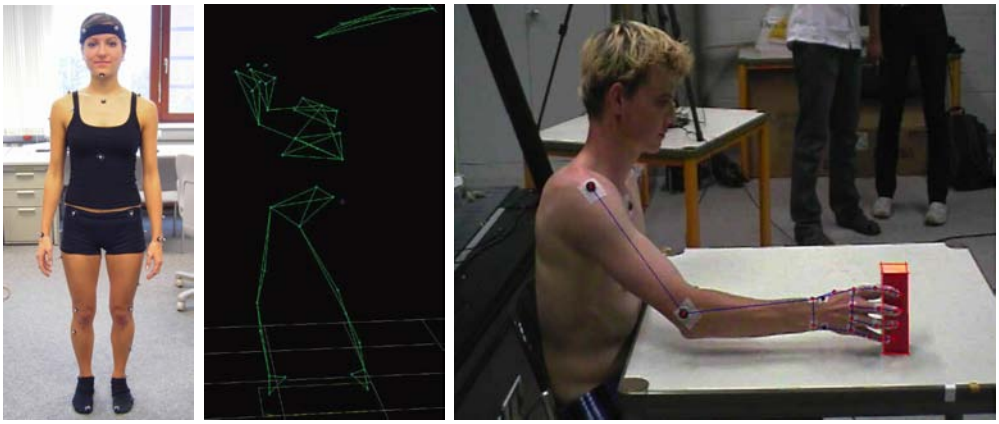
Einsatzspektrum reicht hierbei von der passiven und aktiven Beobachtung mittels visueller Sensorik über den Einsatz von sog. Exoskeletten, die direkt Bewegungen am Körper des Menschen abgreifen, bis hin zu der kompletten Ersetzung der Umgebung durch Simulation in der Virtuellen Realität. Diese Bandbreite soll in dem vorliegenden Abschnitt näher untersucht werden.

[Rittscher et al. 2002, Knoop et al. 2006] stellen verschiedene Verfahren zur markerlosen Beobachtung menschlicher Bewegung vor. Hierbei wird der Mensch lediglich durch die Beobachtung mittels optischer Sensorsysteme erfaßt. Aus den akquirierten Bildsequenzen wird eine Repräsentation der Bewegungen des Menschen generiert, die anschließend von Robotersystemen gelernt und umgesetzt werden können (siehe Abbildung 2.2 a und b) . Die Bewegungen können entweder mittels Kantendetektoren als Silhouetten detektiert werden [Rittscher et al. 2002], oder mit Methoden zur 3D-Rekonstruktion, z.B. mit Hilfe des Iterative-Closest-Point[1]-Verfahrens in ein aussagekräftigeres dreidimensionales approximierendes Modell des menschlichen Körpers verarbeitet werden [Knoop et al. 2006].

[Jiar et al. 1996, Kang und Ikeuchi 1997] benutzen einen aktiven Ansatz, um die Umwelt, in der der Mensch handelt, zu erfassen und in eine relationale Beschreibung zu überführen. Durch strukturiertes Licht, wie z.B. der Projektion eines Streifenmusters (siehe Abbildung 2.2 c) auf eine Szene, wird es möglich, dreidimensionale Informationen aus einem einzigen Kamerabild zu extrahieren. An diese dreidimensionalen Punktwolken wird nun mittels eines 3D-Modellabgleichs[2] die Lage aller Objekte und die Position der

[1] Abk.: ICP

[2] engl.: 3D template matching, Abk.: 3DTM



(a) (b) (c)

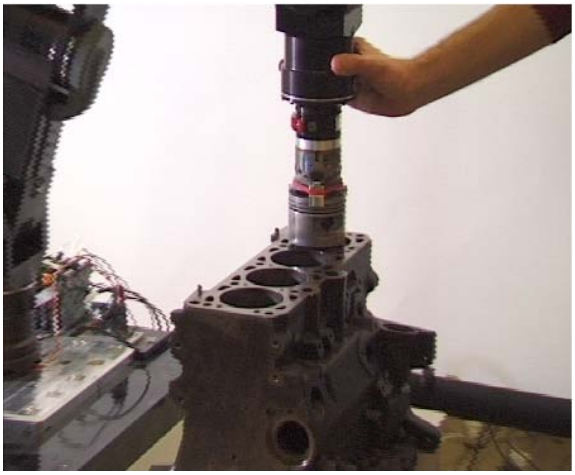
Abbildung 2.3: Markerbasierte Verfolgung menschlicher Bewegungen: (a) Anordnung der Marker. (b) Rekonstruktion des Drahtmodells. (c) Anwendung: Greifen von Gegenständen. Quellen: (a, c) [Wank et al. 2004], (b) [Vicon 2006].

Fingerspitzen des Menschen, der das Objekt manipuliert, ermittelt. Die auf diese Weise erhaltenen Informationen können mit Methoden des räumlichen Schließens[3] zu einem ersten räumlichen Eindruck weiter verarbeitet werden, um eine erste Interpretation der vorgeführten Handlung zu erreichen.

Genauere, aber auch im Betrieb deutlich aufwändigere Systeme zum Erfassen der Bewegung des Menschen und von Objektlagen bieten sogenannte Motion Capture Systeme, wie sie beispielsweise von der Firma Vicon vertrieben werden [Vicon 2006]. Hierbei werden eine Vielzahl von reflektierenden Markern auf dem Körper des Demonstrators befestigt (siehe Abbildung 2.3 a). Diese werden mit Hilfe von bis zu 12 aktiven Hochgeschwindigkeitskameras kontinuierlich und sicher verfolgt. Aus der Bewegung der einzelnen Markerpunkte läßt sich die beobachtete Bewegung auf ein biokinematisches Modell des Menschen abbilden (siehe Abbildung 2.3 b). Diese rekonstruierten Trajektorien der Extremitäten des Menschen werden in [Beth et al. 2003, Wank et al. 2004] benutzt, um menschliche Basisaktionen, wie das Ergreifen und Loslassen eines Gegenstandes, zu analysieren und zu erlernen (siehe Abbildung 2.3 c).

[Grunwald et al. 2001] beschreibt ein System, welches direkt die Gelenkwinkel eines Roboters erfaßt, indem auf dessen Gelenkwinkelenkoder zurückgegriffen wird. Hierbei wird der Roboter unter sogenannter Null-Kraft-

[3] engl.: spatial reasoning



(a)

(b)

Abbildung 2.4: Erfassung von Gelenkwinkeln. (a) Direkte Erfassung durch Gelenkwinkelenkoder am Roboter (b) Ganzkörper-Exoskelett zur Erfassung menschlicher Bewegungen. Quellen (a) [Grunwald et al. 2001] (b) [Meta Motion 2006].

Regelung[4] direkt durch haptische Interaktion mit dem Benutzer auf der gewünschten Bewegungsbahn geführt (siehe 2.4 a). Neben der Bewegung des Werkstücks werden dabei noch Kräfte- und Momente aufgezeichnet, die beim Fügevorgang auftreten. Dieses Verfahren hat den Vorteil, daß es einen hohen Grad an Intuitivität erlaubt und die Abbildung der gelernten Trajektorie auf den Roboter trivial ist, da der gesamte Lernprozeß im Konfigurationsraum des Roboters stattfindet. Nachteile bestehen jedoch in der Spezialisierung dieses Verfahrens auf industrielle Montagesysteme und in den hohen Sicherheitanforderungen, die eine enge Kooperation zwischen Mensch und Roboter erfordert. Dies hat zur Folge, daß solche Systeme erst in jüngerer Zeit für konkrete Anwendungen näher betrachtet werden [Nilsson et al. 2005].

Bei der Beobachtung der Handlungen des Menschen ist die direkte Erfassung der Gelenkwinkel problematisch, da keine Sensoren direkt an den Gelenken angebracht werden könne und Messungen nur über die Hautoberfläche durchgeführt werden können. Ein kommerziell erhältliches System ist das Gypsy-Motion-Cpaturing System [Meta Motion 2006], welches die Gelenkwinkel des Menschen durch an einem Ganzkörperanzug angebrachte Gelenkwinkelsensoren einliest (siehe 2.4 b). Obwohl diese Systeme einen hohen Genauigkeitsgrad aufweisen und in der Filmindustrie bereits mit großem

[4]engl. zero force control

(a) (b)

Abbildung 2.5: Direkte Erfassung der Fingerstellung (a) Datenhandschuh der Firma Immersion (b) Datenhandschuh mit haptischer Sensorik. Quellen (a) [Immersion], (b) [Zöllner et al. 2001].

Erfolg zur Bewegungserfassung und Animation künstlicher Charaktere eingesetzt werden, sind sie zur Instruktion von Robotersystemen nur bedingt geeignet, da das Anlegen des Anzugs große Sorgfalt und viel Zeit beansprucht sowie die Genauigkeit, wie sie für Industrieroboter gefordert ist, nicht erreicht werden kann.

Dieser Nachteil wird umgangen, wenn man sich nur auf die Erfassung der Körperteile konzentriert, die bei menschlicher Handlung üblicherweise die höchste Relevanz besitzen, nämlich der Hände. Zur Aufzeichnung der Hand- und Fingerstellung existieren kommerzielle Systeme [Immersion], die lediglich mittels Handschuh-ähnlicher Überzügen arbeiten (siehe Abbildung 2.5). Innerhalb dieser sind Dehnmeßstreifen fest vernäht, so daß sich die Flexion der Fingergelenke direkt auf sie überträgt. Mittels dieser Dehnmeßstreifen lassen sich die Gelenkstellungen der einzelnen Finger auf einfache Art und Weise elektronisch auslesen.

In [Zöllner et al. 2001] werden solche Datenhandschuhe mit Kraft-Meß-Sensoren an den Fingerspitzen bestückt. So können zusätzliche Erkenntnisse über die bei Manipulationshandlungen ausgeübten Kontaktrelationen und Kontaktkräfte zwischen Hand und Werkstück gewonnen und bei der Umsetzung auf einen Roboter genutzt werden.

[Aleotti et al. 2003] verzichtet unter Verwendung einer gleichartigen Sensorik komplett auf eine Demonstration im wirklichen Arbeitsraum des Menschen und verlagert sie in einen virtuellen Handlungsraum. Die damit deutlich reduzierte Komplexität von Objektmodellierung und Sensorik wird

erkauft durch Abstriche bezüglich der Intuitivität des Verfahrens und Ergonomie des Systems. Dennoch konnte in [Aleotti et al. 2003] der Nachweis erbracht werden, daß die gewonnenen Daten ausreichen, um damit Roboterprogramme zu erlernen.

2.2 Inferenz im Handlungslernen

Die zentrale Komponente im Handlungslernen ist sicherlich die Fähigkeit zur Inferenz. Erwartungsgemäß findet sich hier eine weite Bandbreite in den betrachteten Systemen vor.

Inferenz bezeichnet den Prozeß, bei dem logische Schlüsse und Verallgemeinerungen aus einer oder mehreren Thesen anhand von Daten und Inferenzregeln gezogen werden. Diese Inferenzregeln können sowohl induktive, deduktive, unüberwachte oder interaktive lernende Systeme und Prozesse unterstützen. Überblicke über die verwendeten Methoden, Paradigmen und Algorithmen finden sich unter [Sim et al. 2003, Mahadevan 1996].

- *Induktive* Methoden benötigen eine Reihe von Demonstrationen, um konkretes Handlungswissen zu gewinnen. Mit Hilfe von Inferenzmechanismen wird dabei vom Speziellen zum Allgemeinen gefolgert, d.h. die Lerndaten werden sukzessive zu allgemeingültigem Handlungswissen verarbeitet.

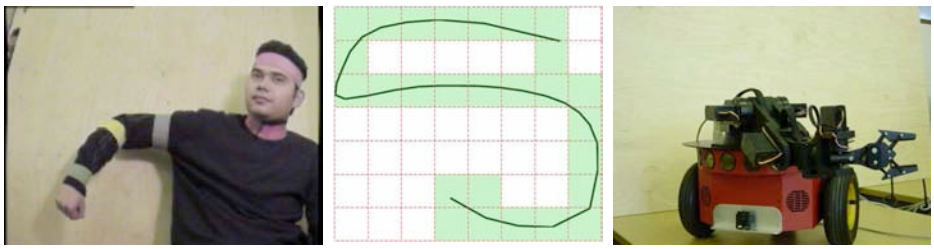
- *Deduktive* Methoden unterstützen dagegen den Übergang von allgemeingültigem Regel- und Domänenwissen zu spezialisiertem Handlungswissen. Bei vollständigem Hintergrundwissen ist es durchaus möglich, aus wenigen oder sogar unter Verzicht auf Lerndaten sinnvolles Wissen zu generieren. Sie gehen hierbei aus von allgemeinem Vorwissen und leiten daraus spezielles, anwendungsbezogenes Wissen ab.

- *Unüberwachte* Lernverfahren weisen den höchsten Grad an Autonomie bezüglich der Inferenz auf. Sie sind ausgezeichnet durch die geringste Einschränkung in Bezug auf die Inferenzregeln von allen Verfahren. Dadurch nimmt aber auch ihre Mächtigkeit und Aussagekraft gegenüber anderen Verfahren in gleichem Maße ab. Bei der Anwendung von unüberwachten Lernverfahren stehen statistische Ballungsverfahren im Vordergrund.

- Den geringsten Grad an Autonomie weisen *interaktive* Verfahren auf. Hier greift der Anwender direkt in den Inferenzprozeß ein, entweder direkt durch Vorgabe des Inferenzergebnisses oder durch Erläuterungen zu oder Bewertungen von Systemhypothesen.

In den folgenden Abschnitten wird ein detaillierteres Bild der unterschiedlichen Arten von Schlußfolgerungsverfahren in Systemen zum Handlungslernen von Roboter gegeben.

2.2.1 Induktive Inferenz



(a) (b) (c)

Abbildung 2.6: Einfache Induktive Algorithmen: k-näechster-Nachbar Methode (a) Demonstration (b) Merkmalsvektoren (c) Reproduktion. Quelle: (a-c) [Calderon 2005]

Ein einfaches Verfahren zur Wiedererkennung und Ausführung von Bewegungen präsentieren [Calderon und Hu 2003, Calderon 2005]. Hierbei wird eine zweidimensionale Bewegung des menschlichen Armes (siehe Abbildung 2.6 a) mittels eines lokalen k-nächste-Nachbarn[5]-Algorithmus mit einer Datenbank von beobachteten Bewegungsmustern verglichen und eine Klassifikation vorgenommen. Die extrahierten Merkmalsvektoren entstehen durch eine Rasterung in der Bewegungsebene sowie anschließender Binarisierung durch Bestimmung der durch die Handtrajektorie berührten Zellen des Rasters (siehe 2.6 b). Zuletzt wird ein Programm, welches in der Datenbank der erkannten Bewegungsklasse zugeordnet wurde, ausgeführt, welches den Roboterarm auf einer Bewegungsbahn führt, welche der demonstrierten so nahe wie möglich kommt(siehe Abbildung 2.6 c). Die Arbeit zeigt, daß bereits mit sehr einfachen Mitteln eine erfolgreiche Klassifikation von Handlungswissen vorgenommen werden kann. Die Nachteile dieses Verfahrens werden jedoch auch deutlich in der Beschränkung auf 2D-Anwendungen: Gerade das hier verwendete Verfahren ist sehr anfällig für den sogenannten „Fluch der Dimension“[6], der es verbietet, ein für höherdimensionale Räume zufriedenstellendes Ergebniss zu erhalten.

Die inverse Kinematik eines Roboterarms mit immerhin schon 6 Freiheitsgraden wird in [Aleotti et al. 2004] beschrieben. Über eine Trainingsdaten-

[5]endl.: k-nearest-neighbour
[6]engl.: curse of dimensionality

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

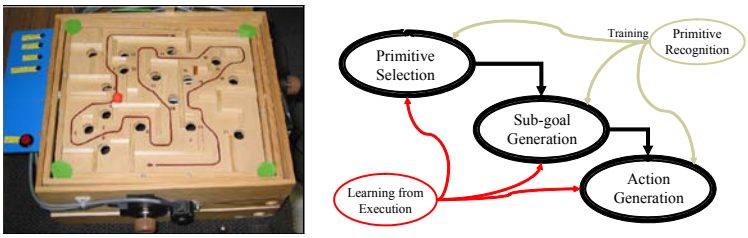
Abbildung 2.7: Komplexe Induktive Verfahren. (a-b) Reinforcement Learning (a) Demonstration (b) Reproduktion. (c-d) Markov-Ketten (c) Demonstration (d) Reproduktion. Quelle: (a-b) [Schaal et al. 2003a] (c-d) [Calinon und Billard 2006a]

menge, welche aus 243 einzelnen Punkten im Arbeitsraum des Roboterarms besteht, wird die inverse Kinematik folgendermaßen aufgebaut: Der Roboterarm steuert einen dieser Punkte im Gelenkwinkelraum an, die anschließend vom Benutzer angefahren wird. Dabei werden die kartesischen Koordinaten der Benutzerhand erfasst. So wird für jede der Trainingskoordinaten verfahren. Die inverse Kinematik kann anschließend durch mehrschichtige Neuronale Netze[7] für jeden einzelnen Gelenkwinkel gelernt werden. In einer Reproduktionsphase kann zuletzt jeder Punkt im kartesischen Raum in den Gelenkwinkelraum des Roboters abgebildet werden. Auch hier wurde mit Hilfe relativ einfacher Methoden die komplexe Aufgabe der inversen Kinematik erledigt, jedoch zu Kosten der Datenaufnahme, die sich aufgrund der großen Datenmenge mühselig und langwierig gestaltet.

Komplexere Verfahren, wie Reinforcement Learning, erlauben es nicht nur, einzelne Punkte im Arbeitsraum des Roboters zu betrachten, sondern diese über die zeitliche Dimension auszudehnen, d.h. gesamte Trajektorienverläufe zu erlernen. So wird in [Schaal et al. 2003b] von erfolgreichen Versuchen, beispielsweise die Schwungbewegungen für eine Tennis-Vorhand eines Lehrers durch einen Roboter reproduziert (siehe Abbildung 2.7 a und b). Hierbei ist besonders zu erwähnen, daß die Demonstration des Menschen nach der Übertragung auf den Roboter noch durch eigenständiges Trainieren verbessert wird.

[Calinon und Billard 2004, Calinon et al. 2005] und [Billard et al. 2004] präsentieren Systeme, welche es erlauben, noch komplexere und vor allem umfangreichere Bewegungen zu erlernen Am Beispiel der Reproduktion von Buchstaben (siehe Abbildung 2.7 c-d) werden mittels Hidden-Markov-Modellen die zeitlichen Abfolgen von verschiedenen

[7] engl.: Multi-layerd neural networks



(a) (b)

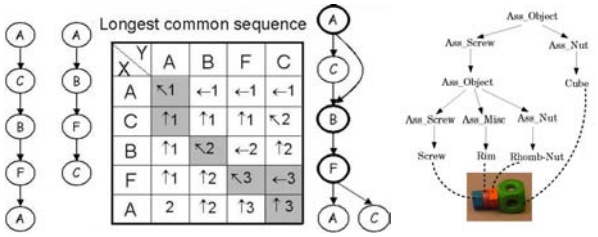
Abbildung 2.8: Deduktive Verfahren I. (a) Das Spiel „Marble Maze“ mit seitlich angebrachter Aktorik (b) Inferenzprozess Quellen: (a-b) [Bentivegna 2004]

Kontrollstrategien gelernt, welche es erlauben, mit mehreren Richtungs- und Zielwechseln umzugehen. Wiederum steigt dabei die Anzahl der benötigten Trainingsbeispiele mit der Komplexität der Aufgabe an, was dazu führt, daß auch für noch relativ einfache Aufgaben, wie das Zeichnen des Buchstabens „B“ 5-7 Demonstrationen notwendig sind. Weiterhin ist die Tatsache unbefriedigend, daß der Roboter wenig Möglichkeiten besitzt, dem Benutzer das gelernte Handlungswissen zu vermitteln, da sich seine Sicht auf das Handlungswissen gravierend von den menschlichen Denkstrukturen und Verbalisierungsfähigkeiten unterscheidet.

2.2.2 Deduktive Inferenz

Die Lücke in der expliziten Beschreibbarkeit des Handlungswissens, die bei den meisten induktiven Lernverfahren existiert (siehe Abschnitt 2.2.1), vermeiden die deduktiven Inferenzverfahren. Hierbei wird meist während der Lernprozesse auf eine explizite Menge an Hintergrundwissen oder eine Primitivenbiliothek zurückgegriffen, die von den konkreten Sensorsignalen abstrahiert und eine Zwischensprache bildet, die im allgemeinen gut auf menschliche Begrifflichkeiten abgebildet werden können.

Eine solche Zwischensprache bilden beispielsweise Primitiven, wie sie von [Bentivegna 2004] eingesetzt werden, um das Spiel „Marble Maze“ zu meistern. Hierbei geht es darum, eine Kugel auf einem vorgegebenen Pfad durch ein Labyrinth von Wänden an Löchern vorbei zu steuern, ohne die Kugel in ein Loch fallen zu lassen. Auf die Kugel können Kräfte ausgeübt werden, indem mittels zweier seitlich angebrachter Rollen die Ebene, in welcher die Kugel rollt, in zwei Dimensionen geneigt werden kann. Bei den verwendeten Primitiven wird unterschieden zwischen Rollen in eine Ecke, an einer Wand entlang, von einer Wand weg oder komplett frei. Primitiven sind hierbei Lö-



(a) (b)

Abbildung 2.9: Deduktive Verfahren II. (a) Auflösung mehrdeutigen Verhaltens. (b) Inferenz von Objekteigenschaften für zusammengesetzte Objekte. Quellen: (a) [Nicolescu und Mataric 2004], (b) [Bauckhage et al. 1998]

sungen für kleine Teile einer Gesamtaufgabe, die miteinander kombiniert werden können, um letztere zu lösen, in diesem Falle das erfolgreiche Erreichen des Zielpunktes mit der Kugel. Sie werden hauptsächlich eingesetzt, um die Komplexität des Lernproblems zu reduzieren. Primitiven enthalten ein sog. internes und ein externes Modell: das interne Modell beschreibt, wie die Primitive auf einem bestimmten Robotersystem auszuführen ist, d.h. es enthält Bewegungs- und Perzeptionsbefehle. Das externe Modell ist eine Beschreibung der Eigenschaften der Primitive. Es umfaßt Vor- und Nachbedingungen, Effekte, Ziele und Teilziele sowie Nebenwirkungen der Ausführung der Primitive und ist roboterunabhängig.

Auf der Ebene der Navigation ist die Erkennung der Primitiven ein deutlich schwierigeres Problem, da hier mehrere Primitiven gleichzeitig hypothetisch verfolgt werden können, es jedoch nicht klar ist, welche gerade die relevanteste ist. So können beispielsweise das Verfolgen einer Wand und das Anfahren eines bestimmten Objektes (zwei unterschiedliche Primitiven) bei entsprechender Umweltkonfiguration dieselben Bewegungen zur Folge haben, was das Problem der Erkennung von Primitiven nicht mehr eindeutig lösbar werden läßt (siehe [Nicolescu und Mataric 2001a, Nicolescu und Mataric 2001b]). Ebendort wird deshalb ein Verfahren vorgeschlagen, welches unterschiedliche Demonstrationen derselben Navigationsaufgabe in unterschiedlichen Umgebungen heranzieht, um die Mehrdeutigkeiten aufzulösen. Der verwendete Algorithmus basiert auf der Suche nach der längsten gemeinsamen Teilsequenz von Primitiven in unterschiedlichen Demonstrationen derselben Aufgabe sowie anschließendem Ausfiltern der nicht intendierten Verhalten nach dem Ausschlußprinzip (siehe Abbildung 2.9(a)).

Auf einer abstrakteren Ebene behandelt [Fritsch et al. 2000] die Auswirkungen von Fügeoperationen. Hierbei werden Operationen zum Zusammensetzen komplexer Objekte aus Einzelbausteinen betrachtet und anschlie-

ßend Rückschlüsse auf das neu erhaltene Objekt gezogen, die zuletzt dazu verwendet werden können, um automatisch perzeptive Operationen zur Erkennung des Objektes in einer Szene (siehe Abbildung 2.9 (b)) durchzuführen. Unter Einsatz von Methoden des Erklärungsbasierten Lernens wird aus einer Menge von Grundobjekten und der Art deren Verbindungen untereinander auf Eigenschaften des komplexen Objektes geschlossen, die ausgenutzt werden, um das Objekt bei weiteren Fügeoperationen durch visuelle Beobachtung identifizieren zu können. Wichtig ist hierbei, daß dies ein rekursiver Vorgang ist, d.h. die Produkte einer Operation können in weiteren Operationen als Ausgangsmaterialien dienen.

Einen hybriden Ansatz für Erklärungsbasiertes Programmieren komplexer Handlungen für Serviceroboter behandelt [Friedrich 1998]. Elementare Handlungsoperatoren werden dabei zu Handlungsketten zusammengefaßt, deren Eigenschaften durch prädikatenlogische Propagierung der vom Benutzer verursachten Änderungen und Effekte in einer Umgebung bestimmt werden. Eine sorgfältige Auswahl der Operationen, welche die Effekte einer Sequenz bewirken, erlauben es, irrelevante Operationen einer Gesamthandlung zu identifizieren und zu eliminieren.

2.2.3 Unüberwachte Inferenz

Zwei verschiedene Arten von unüberwachtem Lernen in Robotersystemen sind prinzipiell möglich: Einerseits das Lernen aus selbstständigem Durchführen von Handlungen mit dem Ziel, Operationssfolgen zu verbessern, um ein bestimmtes Optimalitätskriterium zu maximieren, auf der anderen Seite das autonome Erschließen von Strukturen und Konzepten.

Diskrete zeitdynamische Modelle senso-motorischer Fähigkeiten werden in [Atkeson und Schaal 1997] gelernt, indem das Robotersystem seine eigenen Aktionen und vor allem deren Auswirkungen in Bezug auf das Handlungsziel hin beobachtet. Durch eigenes Experimentieren verbessert es dadurch seine Fähigkeiten. Aus Erfolgen bzw. Mißerfolgen kann eine Belohnungs- und Bestrafungsfunktion abgeleitet werden, die erfolgreiche Aktionen fördert, schlechte hingegen sanktioniert. Daraus wird eine verbesserte Regelungsstrategie entwickelt, die es ermöglicht, komplexe senso-motorische Fähigkeiten aus eigenen Experimenten für das Robotersystem zur Verfügung zu stellen.

Einen Ansatz zum unüberwachten autonomen Lernen durch Interaktion mit der Umwelt präsentiert [Cohen 2000]. Hierbei werden einerseits Kategorien von Aktivitäten durch den Roboter erlernt, andererseits Abstraktionen über diesen Kategorien abgeleitet, die in sogenannten Konzepten münden. Basierend auf den Sensorwerten einer Roboterplattform werden mit steigen-

dem Umfang der Menge von verfügbaren Episoden Cluster identifiziert, die ähnliche Situationen umfassen. Innerhalb dieser Ballungen werden Prototypen identifiziert, die den Cluster repräsentieren. Aus diesen monolithischen Darstellungen des Wissens werden unter Hinzunahme der anderen Demonstrationen Konzepte formuliert, die das gesamte Handlungswissens eines Clusters enthälten. Als Ergebnis erhält man eine primitive Ontologie der Aktivitäten innerhalb des vorhandenen Erfahrungsschatzes.

2.2.4 Interaktive Inferenz

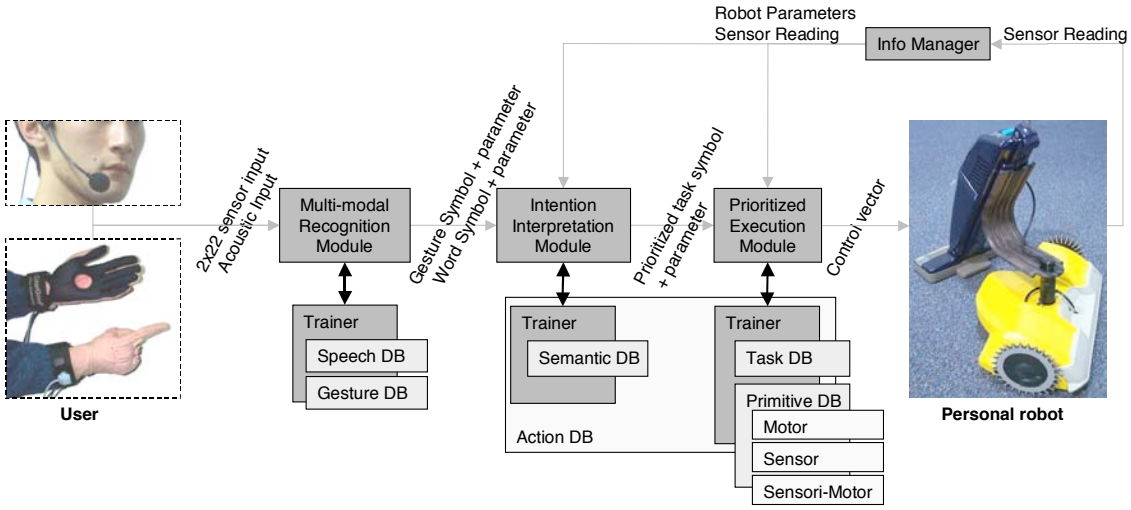
Menschliche Lernvorgänge basieren in erheblichem Umfang auf dem Gebrauch sozialer und kommunikativer Prozesse um den Wissenstransfer zu erleichtern und zu beschleunigen. In der jüngeren Zeit wird der Wissensakquisition durch Interaktion im Kontext von Programmieren durch Vormachen immer größere Bedeutung zugestanden.

[Scassellati et al. 2006] benutzt soziale Interaktionsformen zur Bestimmung des Aufmerksamkeitsfokus während des Wissenstransfers. Es wird die Tatsache ausgenutzt, daß die Objekte, denen eine höhere Relevanz zur Erreichung des Handlungszieles zukommt, häufiger im Blickfeld liegen, als weniger relevante. Vor diesem Hintergrund können mittels Blickrichtungsverfolgung durch ein Stereokamerasystem die im aktuellen Kontext wichtigen Objekte bestimmt werden. Dies ermöglicht eine Objektauswahl sowie weitere Auswertung der Benutzerintention und des Handlungsziels.

Positive bzw. negative Sanktionierung als Folge gelernten Verhaltens, d.h. eine Rückmeldung auf durchgeführte Handlungen, übertragen [Calinon und Billard 2006b] auf das Problem des Handlungslernens aus Demonstrationen. Zusätzlich zu den oben bereits erwähnten Möglichkeiten der Aufmerksamkeitsfokusierung durch Blickrichtungsverfolgung werden hier Zeigegesten sowie auch die gesamte Kopforientierung aufgenommen (siehe Abbildung 2.10(a)). Dies erlaubt es, direkt während der Ausführung einer gelernten Handlung dem System eine Bewertung seiner Aktionen mittels Kopfschütteln oder Nicken zu geben. Das System kann dadurch seine Hypothesen auf Übereinstimmung mit der Benutzerintention überprüfen und diese gegebenenfalls anpassen oder verwerfen. In einer Pilotstudie bestätigten zwanzig Versuchspersonen eine erhöhte Natürlichkeit, Benutzerfreundlichkeit und Einfachheit des Wissenstransfers verglichen mit einer interaktionslosen Methode.

Das in [Iba et al. 2003] beschriebene System verläßt sich zur Kommandierung und Programmierung eines Staubsaugerroboters einzig auf interaktive Mechanismen. Mittels sprachlicher und nonverbaler Kommunikation (neben Zeigegesten wird ein fest definiertes Gestenvokabular angewandt) wird der

(a)



(b)

Abbildung 2.10: Interaktive Inferenz. (a) Aufnahme von Kopf- und Handbewegungen zur Sanktionierung erwünschten Verhaltens. (b) Interaktive Progrmmierung mittles Gesten und Spracheingabe, Systementwurf. Quellen: (a) [Calinon und Billard 2006b] (b) [Iba et al. 2004].

Roboter in die gewünschten Areale geführt. Daraus kann bereits ein vorläufiges Roboterprogramm erstellt werden. Eine präemptive Ausführungsphase erlaubt es anschließend das Programm zu verfeinern, Teilaufgaben zu priorisieren und eventuell neue Programmteile hinzuzufügen.

2.3 Modellierung von Handlungswissen

Die Aufnahme, Speicherung, Weitergabe, Verarbeitung und Ausführung von Handlungswissen erfordert geeignete Repräsentationen seiner Inhalte. Dies erfordert adäquate Modelle, um die relevanten Details erfassen und weitergeben zu können. In unterschiedlicher Granularität und Komplexität werden zur Erfassung von Handlungswissen folgende Aspekte modelliert:

- Die *Umwelt*. Das Umweltmodell beschreibt die Umgebung, in welcher der Mensch bzw. der Roboter handelt und die vielfältigen Einflüße, welche die Umgebung auf die Handlungen hat.
- *Trajektorien*. Handlungen gehen einher mit der Durchführung von Änderungen in der Umwelt. Diese werden durch unterschiedlichst geartete Bewegungen des Menschen bzw. des Roboters initiiert. Deshalb sind spezielle Modelle zur Erfassung von Bewegungen nötig.
- Der *Mensch*. Die Handlungsmöglichkeiten des menschlichen Benutzers sowie seine internen Zustände sind neben der Erkennung zur Demonstrationszeit auch für die sichere und benutzerfreundliche Ausführung des Handlungswissens notwendig. Das Wissen hierüber wird in der Modellierung des menschlichen Benutzers zusammengefaßt.
- Die *Aufgabe*. Um ein intelligentes Systemverhalten wenigstens zu simulieren, ist es notwendig, Wissen über die Voraussetzungen, Kontexte, Ausführungsbedingungen, Ziele und Teilziele von Handlungen zu erlangen und vorzuhalten. Dies geschieht in der Aufgabenmodellierung.

Die oben genannten Modellarten sollen in den folgenden Unterabschnitten weiter detailliert und durch Beispiele aus der Literatur verdeutlicht werden.

2.3.1 Umweltmodelle

Modelle der Umwelt erlauben es, die vielfältigen Eigenschaften und Einflüsse von Objekten im Umfeld des Menschen bzw. des Roboters zu berücksichtigen. Es sind Objekt- und Umweltmodelle mit unterschiedlicher Granularität und Aussagekraft bezüglich geometrischer Beschaffenheit und der Semantik des Umfelds bekannt. Sie sind die Grundlage für die Bewertung, Interpretation und Deutung von Handlungen, die die Manipulation von Objekten beinhalten.

Geometrische Modelle von Objekten in Szenen können als Punkt-, Kanten- oder Flächenmodellen angefertigt werden. Sie beschreiben das Aussehen der Objekte und können im allgemeinen gut zu Simulationszwecken

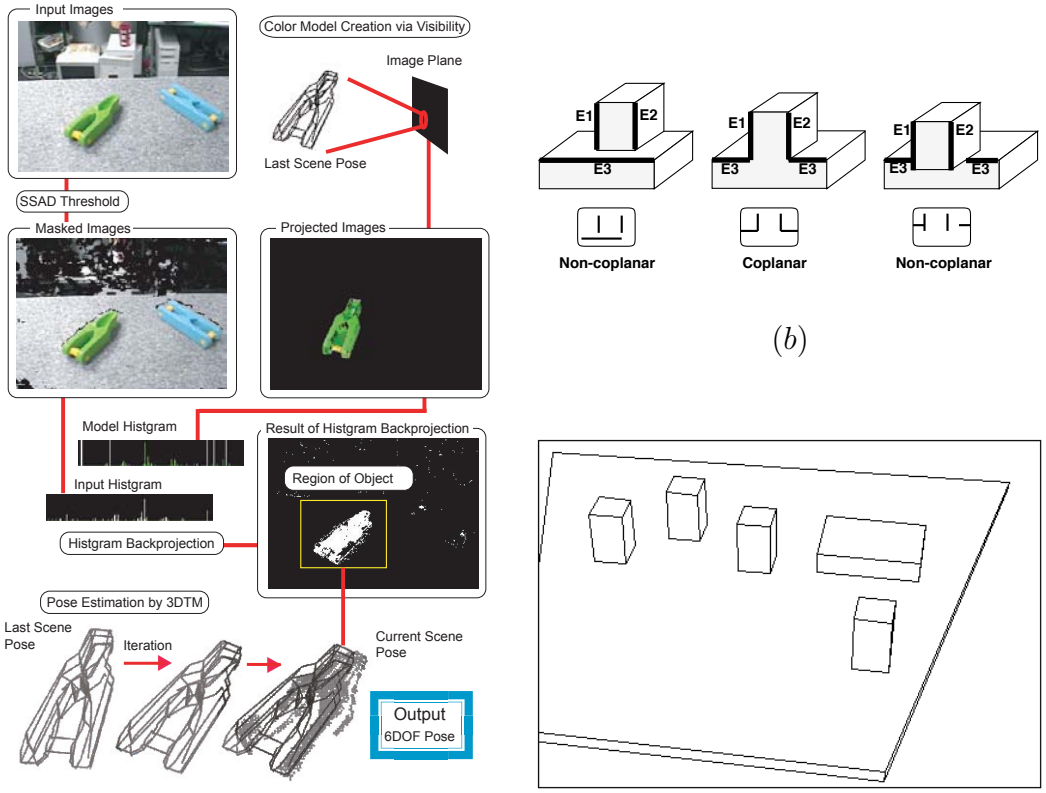


Abbildung 2.11: Objekt- und Umweltmodellierung. (a) Objektmodellierung zur Objektlagenerkennung aus Bildsequenzen (b) Modellierung von Objekt.Kontaktzuständen (c) Aus Bilddaten extrahierter Weltzustand einer Blockwelt. Quellen: (a) [Sato et al. 2002], (b-c) [Kuniyoshi et al. 1994].

genutzt werden, sowie als Grundlage zur sensorischen Erfassung der Objekte. [Sato et al. 2002] beschreibt ein Verfahren, das Kantenmodelle von Werkzeugobjekten auf einer Arbeitsfläche nutzt, um die Objekte zu erkennen sowie deren Position und Orientierung im Raum zu verfolgen (siehe Abbildung 2.11 (a)). Dies kann dazu dienen, die Handlungen, die mit den Objekten durchgeführt werden, zu erkennen und so die Benutzerhandlung zu interpretieren.

[Kuniyoshi et al. 1994] geht noch einen Schritt weiter: Aus Kantenmodellen für relative Beziehungen von Objekten (siehe Abbildung 2.11 (b)) wird eine relationale Beschreibung der Objektlagen zueinander entwickelt. Diese ist als einfache Blockwelt in Abbildung 2.11 (c) dargestellt. Aus diesen

Blockwelten können höherwertige semantische Beschreibungen des gesamten Umfeldes erstellt werden, die zu einer realitätsnäheren Interpretation der Benutzerhandlungen führen können.

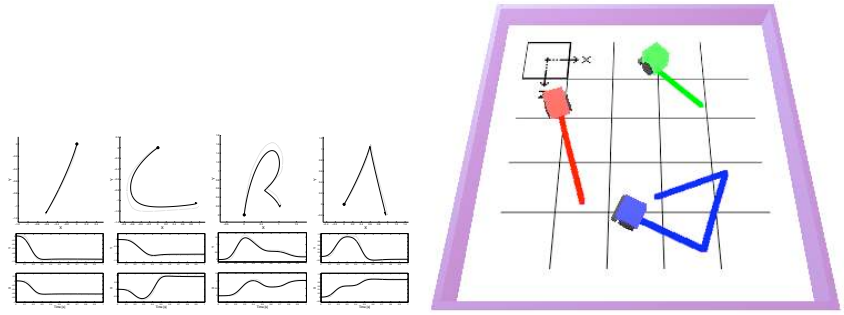
Semantische Objektmodelle gehen über die rein geometrische Modellierung von Objekten hinaus. Sie fügen den Objekten weitere Eigenschaften und Bedeutung zu. [Becher et al. 2006] beschreibt ein System zur autonomen und interaktiven Modellierung von Alltagsgegenständen im Haushaltsbereich. Objekte sind dabei durch verschiedene Merkmale beschrieben, wie beispielsweise „Objekt ist ein Flüssigkeitsbehälter“ oder „Objekt ist transportabel“. Anhand dieser Merkmale kann ein Objekt von Menschen einer Kategorie zugeordnet werden. Gleichzeitig erlauben es Objektmerkmale, dieses Objekt mit verschiedenen Aktionen zu assoziieren. Eine solche Sicht erlaubt einerseits einfache Kommunikation und Interaktion zwischen Mensch und Maschine, da beide in den selben Kategorien agieren, andererseits können Instruktionen des Benutzers relativ einfach auf die zugehörigen Objektkategorien abgebildet werden. Zusätzlich zu den Merkmalen hat eine konkrete Instanz eines Objektes noch Attribute und Attributwerte, die seiner Einzigartigkeit gerecht werden. Beispielsweise fallen hierunter die Position und Lage der Objektinstanz in der Welt oder der aktuelle Füllstand eines Flüssigkeitsbehälters. Hierbei handelt es sich um objektindividuelle, klassenunabhängige Eigenschaften, die aber unter Umständen für die semantische Bewertung von Handlungen, die diese Instanz eines Objektes manipulieren, von eminenter Bedeutung sind.

2.3.2 Trajektorienmodelle

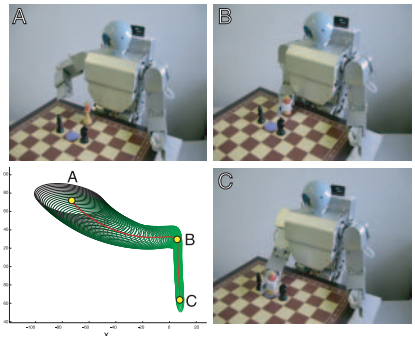
Der Repräsentation von Bewegungen durch Trajektorien kommt besondere Bedeutung zu, da es beim Programmieren durch Vormachen letztlich immer darum geht, Bewegungsdemonstrationen des Menschen auf ein Robotersystem zu übertragen.

[Ijspeert et al. 2002, Schaal et al. 2003a] beschreiben Bewegungen in kinematischen Koordinaten, z.B. Gelenkwinkeln eines Roboters oder kartesischen Koordinaten eines Endeffektors. Eine Steuerungsvorgabe bzw. ein Bewegungsplan wird als nichtlineare dynamische Attraktor-Differentialgleichung mittels lokal gewichteter Regression[8] gelernt, die autonom die gewünschte Trajektorie erzeugt. Abbildung 2.12 (a) zeigt die Aufteilung zweidimensionalier Trajektorien in Gelenkwinkelkoordinaten sowie deren dynamischen Bewegungsplan, welcher mittels Regression der nichtlinearen Differentialgleichungen erzeugt wurde. Zyklische, repetitive Bewe-

[8] engl.: locally weighted regression



(a) (b)



(c)

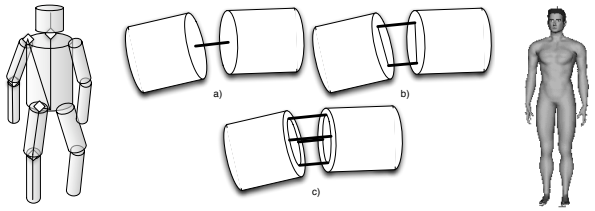
Abbildung 2.12: Trajektorienmodelle. (a) Bewegungsbeschreibung durch Gelenkwinkelkoordinaten. (b) Beschreibung von Schwarmverhalten durch Multi-Roboter-Trajektorien. (c) Trajektorienschlauch zur gezielten Beschränkung von Bewegungen. Quellen: (a) [Ijspeert et al. 2002], (b) [Alissandrakis et al. 2005] (c) [Calinon et al. 2006]

gungsbahnen werden in [D'Souza et al. 2001] mittels ähnlicher Verfahren behandelt und gelernt.

[Alissandrakis et al. 2005] verallgemeinert Trajektorien auf die Gesamtbewegungen eines Roboterschwarms. Es gelingt, bestimmte Schwarmverhalten durch Lernen von einem Modellverhalten zu erzeugen und auf mehreren Robotern simuliert zur Ausführung zu bringen (siehe Abbildung 2.12 (b)). Besonderer Augenmerk wird dabei auf die notwendige gegenseitige Perzeption und Synchronisation der Agenten, die einen Roboterschwarm formen, gelegt (siehe auch [Alissandrakis et al. 2003b]).

Ein wesentlicher Nachteil Trajektorien-basierter Verfahren ist sicherliche die mangelde Generalisierungsfähigkeit. Wünschenswert wäre es, nicht nur einzelne Bewegungen zu lernen, sondern komplette Trajektorienkorridore.

[Yeasin und Chaudhuri 2000] fasst unterschiedliche Trajektorien zu



(a) (b)

Abbildung 2.13: Modelle des Menschen (a) Kinematisches Modell des Menschen (b) Geometrisches Oberflächenmodell des Menschen in Animationsqualität. Quellen: (a) [Knoop et al. 2006] (b) [Humanoid Animation Working Group 2003].

sogenannten Trajektorienbündeln zusammen. Anschließend wird durch Mittelwertbildung eine Trajektorie erzeugt, welche innerhalb dieses Trajektorienbündels liegt. So entsteht ein Trajektorienkorridor. Die möglichen Abweichungen von dieser Solltrajektorien bleiben jedoch unquantifiziert, so daß zur Ausführungszeit eine Rest-Unsicherheit bleibt.

Eine andere Möglichkeit wird in [Calinon et al. 2006, Hersch und Billard 2006] beschrieben. Hier wird für jeden Trajektorienpunkt ein eliptischer Freiraum konstruiert, der den möglichen Trajektorienverlauf abdeckt. Dadurch wird es möglich, nicht nur den maximal möglichen Abstand der durchgeführten Trajektorie zur Solltrajektorie anzugeben, zusätzlich können bestimmte Dimensionen, in denen größere Varianz erlaubt ist, explizit von Dimensionen mit höherer Restrikition unterschieden werden. Abbildung 2.12 (c) zeigt dies deutlich am Beispiel einer Bewegungsaufgabe, die in einigen Teilen sehr beschränkt, in anderen sehr frei ist.

2.3.3 Mensch- und Benutzermodelle

Die Erfassung der Inention des Benutzers erfordert genaue Modellierung des Menschen unter besonderer Berücksichtigung seiner physischen Möglichkeiten und Beschränkungen, sowie der Interaktionsmöglichkeiten.

[Knoop et al. 2006] beschreibt die kinematischen Bewegungsmöglichkeiten des menschlichen Körpers durch seine Gelenkkonfigurationen (siehe Abbildung 2.13 (a)). Hierbei wird zwischen Kugel-, Knick- und elliptischen Gelenkformen unterschieden. Dieses Modell erlaubt es, mögliche Bewegungen zu bestimmen sowie diese in Bildfolgen wiederzuerkennen. Ein deutlich detaillierteres Modell des menschlichen Körpers wird in [Humanoid Animation Working Group 2003] standardisiert (siehe Ab-

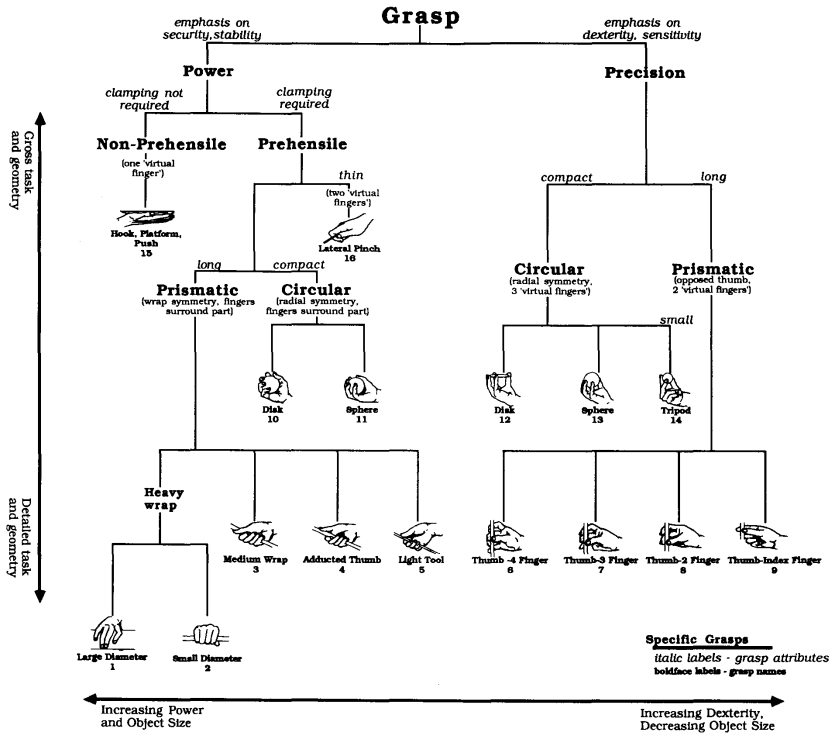


Abbildung 2.14: Funktionalitätsmodelle des Menschen am Beispiel von Greifmöglichkeiten der menschlichen Hand. Quelle: [Cutkosky 1989]

bildung 2.13 (b)). Dies erlaubt eine realitätsnahe Darstellung und ist insbesondere für den Animationsbereich ausgelegt.

Eine funktionale Modellierung menschlicher Griffarten wird in [Cutkosky 1989] vorgeschlagen (siehe Abbildung 2.14). Eine solche erlaubt es, den verwendeten Grifftyp, welchen ein Mensch zum Greifen eines Objektes gewählt hat, zu analysieren. Rückschlüsse auf die ausgeübten Kräfte und Momente sowie die Stabilität eines Griffes können benutzt werden, um Greifvorgänge bei Robotern zu parametrisieren und Handprothesen für den Menschen zu optimieren.

Eine Einteilung menschlicher Aktivitäten zu Observationszwecken wird in [Ehrenmann et al. 2003] vorgenommen. Hierbei wird primär zwischen performativen, kommentierenden und kommandierenden Aktionen unterschieden (siehe Abbildung 2.15). Je nachdem, welche Aktion vom Menschen durchgeführt wird, muß das Roboterverhalten eventuell angepaßt werden, um die Zielrichtung der Handlung verstehen zu können, sowie um die Benutzerakzeptanz zu maximieren.

Ein Modell für die Ermüdungsprozesse, die mit längeren Tätigkeiten ein-

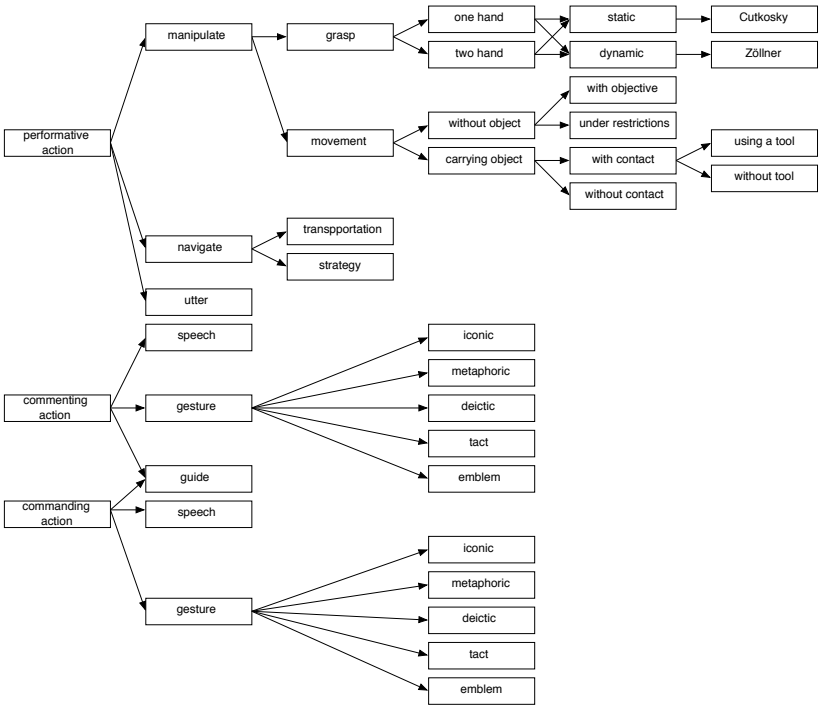


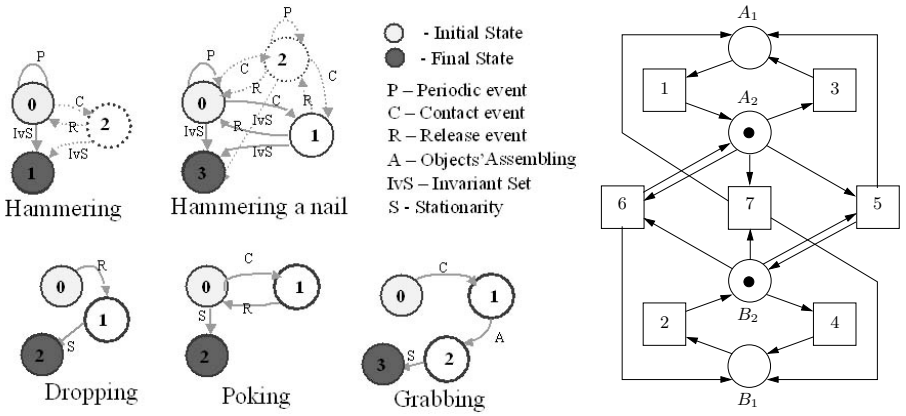
Abbildung 2.15: Aktivitätssmodelle des Menschen. Quelle: [Ehrenmann et al. 2003].

hergehen, wird in [Potkonjak et al. 2001] erörtert. Besonders wird hier darauf eingegangen, welche Auswirkungen solche Ermüdungserscheinungen auf die Ausführung der Aufgabe haben. Berücksichtigung dieser Faktoren verspricht eine höhere Klassifikationsleistung zum Observationszeitpunkt.

2.3.4 Aufgabenmodelle

Trajektorien, Umweltzustände und Modelle des Menschen reichen in der Regel noch nicht aus, um das Lernen von vollständigen Handhabungsoperationen ausreichend zu stützen. Zielgerichtete Handlungen des Menschen weisen meist sehr klare Strukturen auf. Diese können beispielsweise durch sequentielle Abläufe, zeitlich parallele Verhalten oder hierarchische Gruppierungen von Teilaufgaben erzeugt werden. Verschiedene Ansätze wurden erprobt, um diese strukturierenden Elemente aus Demonstrationen des Menschen sowie diese in eine rechneradäquate Darstellung zu überführen.

[Chen und McCarragher 2000, Onda et al. 1995] beschreiben unterschiedliche Systeme zum Erlernen feinmotorischer Handlungen aus Vorführungen. Dabei werden aus Demonstrationen einer Aufgabe zum Einführen



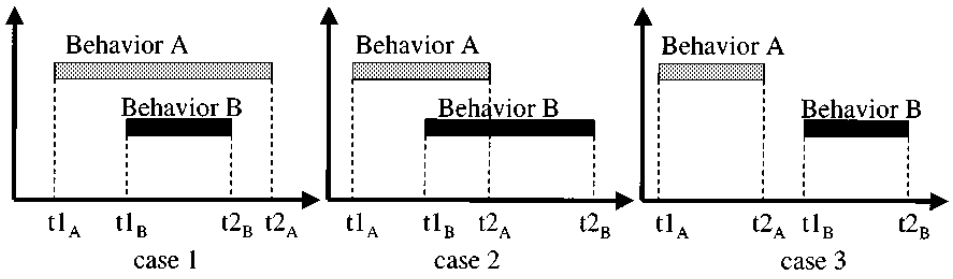
(a) (b)

Abbildung 2.16: Aufgabenmodelle I. (a) Endliche Automaten zur Repräsentation zeitlicher Abfolgen von Teilaufgaben bei Erledigung unterschiedlicher Tätigkeiten. (b) Petrinetz zur Koordination zweihändiger Operationen und Ereignisse. Quellen (a) [Arsenio 2004] (b) [Zöllner et al. 2004]

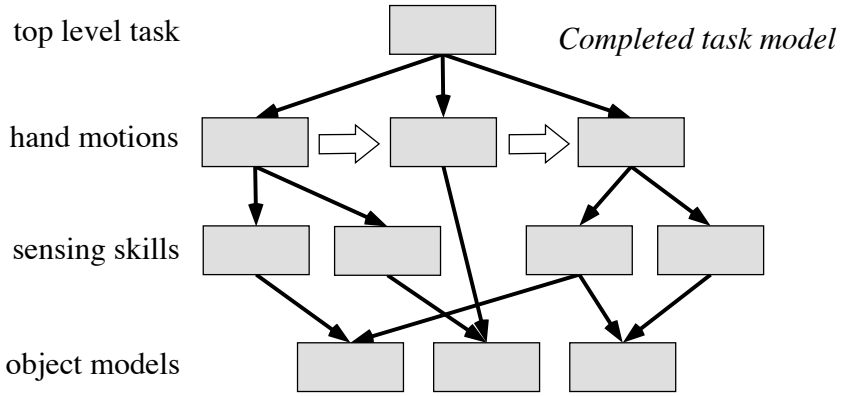
einer Spindel in eine Haltevorrichtung entscheidende Schlüsselpunkte einer Demonstration identifiziert. Deren konkrete Abfolge wird anschließend in einem endlichen Automaten repräsentiert. Endliche Automaten sind ein in der Informatik bereits früh entwickeltes Konzept und verfügen über einen hohen Grad an Formalisierung und eine gesicherte theoretische Basis, auf der sich sequentielle Repräsentationen zeitlich aufeinanderfolgender Ereignisse, wie z.B. von Kontaktzuständen bei Objektmanipulationen, darstellen lassen. [Arsenio 2004] benutzt eine ähnliche Repräsentation für endliche Zustandsautomaten, um verschiedene Aktionen (Winken mit einem Hammer in der Hand, Einhämmern eines Nagels, Fallenlassen eines Objekts, und Aufnehmen eines Objekts) voneinander zu unterscheiden (siehe Abbildung 2.16 (a))

Eine Repräsentation ähnlicher Mächtigkeit wie Endliche Zustandsautomaten stellen Petrinetze dar. [Zöllner et al. 2004] betont deren gute Eignung für die zeitliche Koordination von Aktionen innerhalb von Robotersystemen, da Petrinetze nicht nur rein sequentielle Abfolgen von Operationen sondern auch noch das asynchrone Auftreten unterschiedlicher Ereignisse berücksichtigen. Das erlaubt es, zweihändige Manipulationen, wie beispielsweise das Aufschrauben einer Kaffeedose, einfach zu modellieren (siehe Abbildung 2.16 (b)) und auf dem zweiarmigen Roboter Armar zur Ausführung zu bringen.

[Nicolescu und Mataric 2004] stellt ein System vor, welches die gleichzeitig aktiven Verhalten eines Roboters mit Navigationsfähigkeiten aus



(a)

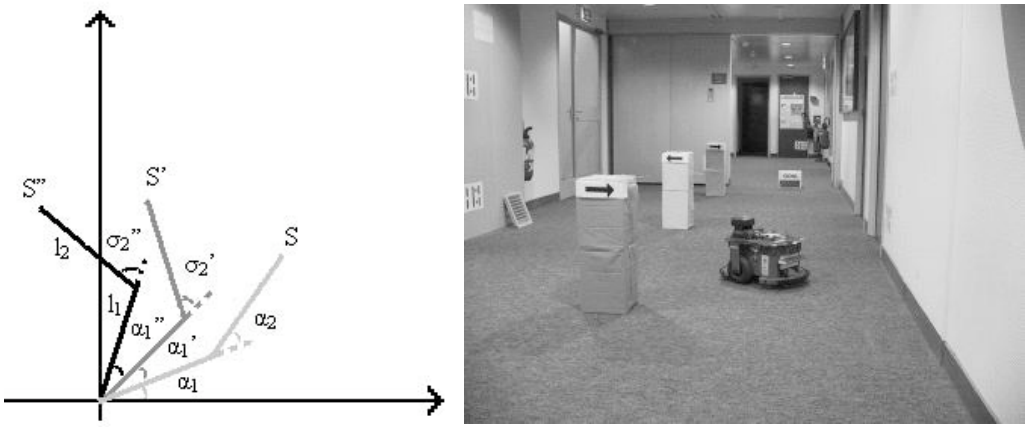


(b)

Abbildung 2.17: Aufgabenmodelle II. (a) Sequentielle und nebenläufige Verhalten. (b) Hierarchische Dekomposition von Aufgaben. Quellen: (a) [Nicolescu und Mataric 2001b] (b) [Miura et al. 2004]

Benutzervorführungen einer Navigationsaufgabe erlernt. Hierbei treten verschiedene Verhalten wie „Suchen eines orangen Objektes“ oder „Verfolgen der nächsten Wand“ auf. Diese Verhalten werden in den Handlungen des Benutzers detektiert und in ihrer zeitlichen Abfolge oder Nebenläufigkeit erfasst (siehe Abbildung 2.17 (a)). Eine verhältnismäßig informelle Beschreibung der Vorrang- und Nebenläufigkeitsbeziehungen wurde gewählt, um dem System zu großer Generalisierungsfähigkeit zu verhelfen. Diese wird benötigt, um aus unterschiedlichen Ausprägungen von Lösungen für dieselbe Navigationsaufgabe, die jedoch in unterschiedlichen Umgebungen durchgeführt wurden, eine allgemeine Repräsentation zu erstellen. Fehlerhaft detektierte Verhalten werden eliminiert.

Systemen, welche eine weitere Spanne von zu lernenden Aufgaben abdecken sollen, reichen solche Aufgabenrepräsentationen jedoch immer noch nicht aus. Die Schwierigkeit liegt oft in den unterschiedlichem Abstraktions-



(a) (b)

Abbildung 2.18: Einfache Roboter. (a) Stilisierte Roboter mit 2 Freiheitsgraden. (b) Pioneer 2DX Roboter zum Erlernen von Navigationsaufgaben. Quellen: (a) [Alissandrakis et al. 2003a] (b) [Hugues und Drogoul 2002].

ebenen, auf denen die Erfassung von Handlungswissen stattfindet. Abhilfe schaffen hier hierarchische Modelle der Aufgaben, wie sie beispielsweise von [Miura et al. 2004] vorgestellt werden. Hierbei wird eine komplexer Aufgabe (z.B. das Öffnen einer Tür) in einfachere Teilaufgaben (Bewegung des Greifers zum Türgriff, Schließen des Greifers, Bewegung des Greifers auf einer Kreisbahn) zerlegt. Teilaufgaben werden als „Makro“ bezeichnet, wenn sie in weitere Teilaufgaben einer niedrigeren Eben zerlegt werden können, während „Primitiven“ atomare Roboteroperationen bezeichnen. Ein solches hierarchisches Aufgabenmodell ist als Baumstruktur graphisch darstellbar (siehe Abbildung 2.17 (b)). Die Makros korrespondieren mit den inneren Knoten, die Primitiven mit den Blättern des Baumes.

2.4 Domänen für Handlungslernen

In den vorherigen Abschnitten sind bereits unterschiedliche Aufgabenstellungen für Systeme zum Handlungslernen skizziert worden. In diesem Abschnitt werden Domänen, d.h. Roboter, deren Aufgaben und Handlungsspielräume und ihr Umfeld detailliert vorgestellt.

Durch einfache Linien stilisierte, simulierte Roboter mit zwei rotatorischen Freiheitsgraden werden in [Alissandrakis et al. 2002, Alissandrakis et al. 2003a] benutzt, um zweidimensionale Bewegungsmuster kinematischer Ketten zu imitieren. Der Roboterstatus ist vollständig

durch die Angabe aller Gelenkstellungen spezifiziert, der Aktionsraum besteht aus deren Ableitungen (siehe Abbildung 2.18 (a)). Mit solch einem Robotermodell ist es möglich, kreisförmige, repetitive Bewegungsmuster von einem anderen Agenten zu imitieren. Aufgrund eines solchen sehr einfachen Aufgabenmodells konzentrieren sich die Arbeiten darauf, wie eine Demonstration eines Lehrers auf Roboter mit unterschiedlichen kinematischen Parametern, z.B. unterschiedlich langen Armelementen, abzubilden sind.

Navigationsaufgaben, wie das Durchfahren einer Tür oder eine Slalom-Fahrt durch einen Raum voller Hindernisse, können mit einem Experimentalsystem, wie es in [Hugues und Drogoul 2002] vorgestellt wird, gelernt werden. Auf eine Pioneer-2DX-Platform wurde ein Kamerasystem montiert. Das so aufgebaute System wird vom Benutzer mittels Fernsteuerung durch die Aufgabe geführt, während die Kamera eine Sequenz von Videodaten liefert. Automatische Detektion signifikanter Merkmale liefert eine kompaktere Darstellung der optischen Wahrnehmung des Systems. In Lernschritten kann eine Perzeptions-Aktions-Relation aufgestellt werden, die in einer Reproduktionsphase die demonstrierten Navigationsaufgaben meistert. Für die gestellten Anforderungen ist das System sicherlich ausreichend, es ist jedoch nicht in der Lage, auf dynamischere Umgebungen mit sich bewegenden Hindernissen, beispielsweise Benutzern oder anderen Robotern, umzugehen.

Eine ähnliche Verknüpfung zwischen Perzeption und Aktion wird in [Breazeal et al. 2005, Lockerd und Breazeal 2004] beschrieben. Ausgehend von der Hypothese, daß natürliche Kommunikationsmethoden und bedeutsame Interaktionen zwischen Menschen und Robotern deren Akzeptanz deutlich erhöhen, wird ein System zum Erlernen von Gesichts-Mimikri vorgestellt. Es wurde ein Roboter entworfen, welcher eine Vielzahl an aktuierbaren Freiheitsgraden im Gesichtsbereich aufweist, um eine reichhaltige Mimik zu ermöglichen (siehe Abbildung 2.19 (a-b)). Der Benutzer wird gebeten, eine Folge zufällig mit dem Roboter erzeugter Gesichtsausdrücke (sogenanntes Motor Babbling) nachzumachen. Dabei wird er von einem Kamerasystem beobachtet, das markante Merkmalspunkte im Gesicht (Position von Augenbrauen und Mundwinkel, Blickrichtung) verfolgt. Mittels Neuronaler Netze wird ein Zusammenhang zwischen den gemessenenen Merkmalspositionen im Gesicht des Benutzers und den Steuerkommandos für die Aktorik des Roboters gelernt. Dieser wird im Imitationsschritt umgekehrt, so dass der Roboter anschließend in der Lage ist, seine Ausdrucksweise an Mimik und Stimmungslage seines Gegenübers anzupassen. Dies soll es in zukünftigen Entwicklungsstufen erlauben, sozial intelligente Systeme zu schaffen, die als solche vom menschlichen Benutzer wahrgenommen und eher akzeptiert werden, als rein technische Systeme.

[Atkeson und Schaal 1997] beschreibt ein System, das feinmotorige

(a) (b) (c)

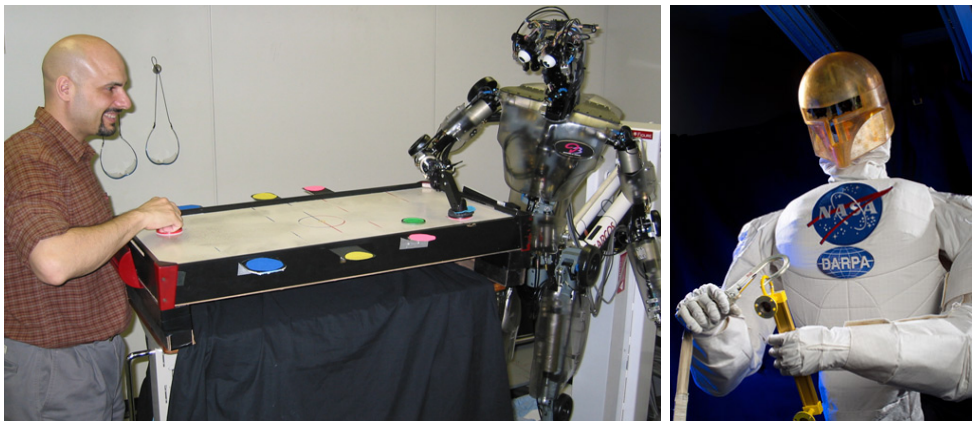
Abbildung 2.19: Roboter zum Erlernen von feinmotorischen Fähigkeiten. (a, b) Spielzeugroboter zum Erlernen sozialer Interaktionsmechanismen durch Mimik. (c) Roboterarm mit sieben Freiheitsgraden zum Erlernen des Pendelaufschwungs. Quellen: (a, b) [Breazeal et al. 2005], (c) [Atkeson und Schaal 1997]

Fähigkeiten, wie den Pendelaufschwung, erlernt. Hierbei hält die Roboterhand die Drehachse eines Pendels, um welche das Pendel wie um ein Gelenk frei drehbar gelagert ist. Initial hängt das Pendel gemäß der Schwerkraft nach unten. Das Ziel der Aufgabe besteht darin, die Hand so zu bewegen, daß das Pendel nach oben schwingt und in der invertierten Position balanciert wird. Diese Aufgabe ist auch für den Menschen eine Herausforderung, verlangt sie doch die Fähigkeit guter Hand-Auge-Koordination. Die Vorführung der Aufgabe durch einen Menschen wird von einem Stereokamerasystem beobachtet und auf einen hydraulischen Roboterarm mit sieben Freiheitsgraden (siehe Abbildung 2.19 (c)) übertragen. Mittels Techniken des Verstärkungslernens[9] wird eine Belohnungsfunktion aus der Beobachtung gelernt und anschließend durch weitere eigene Trainingsversuche verfeinert. Basierend auf dem gelernten Belohnungsmodell wird eine Regelungsstrategie erstellt, welche es erlaubt, mit Sensorungenauigkeiten, Totzeiten und komplexen dynamischen Wechselbeziehungen umzugehen.

[Bentivegna und Atkeson 2001, Bentivegna et al. 2002] stellen ein System vor, welches Spielzüge im AirHockey[10] von seinen Gegnern erlernt und anwenden kann (siehe Abbildung 2.20(a)). Die Bewegungen des Gegnerischen Paddles, und des Pucks werden von einem humanoiden Roboter mittels eines im Sensorkopf integrierten Stereokamerasystems verfolgt und analysiert. Die Vielzahl von Freiheitsgraden, die der humanoide Roboter aufweist, werden

[9] engl.: reinforcement learning

[10] Brettspiel, welches hohe Anforderungen an Reaktionsfähigkeit der Spieler stellt, bei dem ein Puck mit gezielten Schlägen in das gegnerische Tor befördert werden muß



(a) (b)

Abbildung 2.20: Roboter-Torsi. (a) Erkennung und Ausführung elementarer Spielzüge im Spiel AirHockey. Lernen von Werkzeughandhabungen mit dem Weltraumroboter Robonaut. Quellen: (a) [Bentivegna 2004] (b) [Bluethmann et al. 2004]

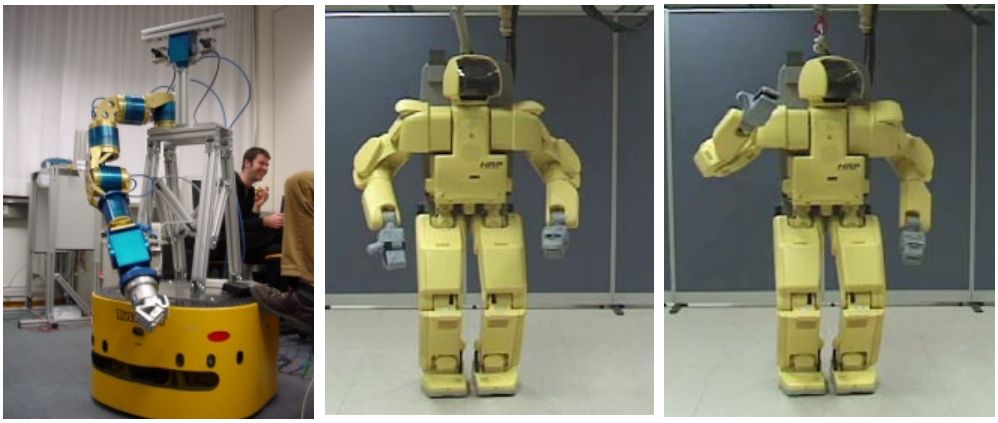
jedoch zugunsten der numerischen Beherrschbarkeit des Lernproblems eingeschränkt auf die Oberkörper-Aktuatoren, was aber immer noch 17 Gelenkwinkelfreiheitsgraden entspricht. Dennoch erzielt das System Leistungen, die durchaus im Bereich menschlicher Spieler liegen. Interessant ist hierbei, dass bereits mit einer einfachen inversen Kinematik, nämlich der Interpolation zwischen 6 verschiedenen, im kartesischen sowie im Gelenkwinkelraum genau bekannten Positionen, zufriedenstellende Resultate erzielt werden können.

Ein System zum Erlernen von Werkzeughandhabungen, das aus einem menschenähnlichen Torso mit zwei Armen und einem Sensorkopf besteht (siehe Abbildung 2.20 (b)), wird in [Bluethmann et al. 2004] vorgestellt. Das System, dessen Einsatzszenarien sowohl unterstützende Funktionen beim Auf- und Ausbau von Raumstationen in Zusammenarbeit mit menschlichen Astronauten, als auch autonome Exploration fremder Planeten vorsehen, ist in der Lage, sowohl von menschlichen Instruktionen aus der direkten Umgebung zu lernen, als auch von einem entfernten Teleoperateur. In beiden Fällen werden die relevanten senso-motorischen Merkmale einer Aufgabe für die Ausführung von beispielsweise dem Anziehen von Muttern und Schrauben gelernt. Betrachtet man die verwendete Methodik genauer, so fällt eine scharfe Trennung zwischen dem Lernen senso-motorischer Fähigkeiten und dem Lernen auf der Aufgaben-Ebene auf. Das Lernen selber erfolgt in vier verschiedenen Phasen: Zuerst werden die Sensordaten und Motorsteuerungskommandos abgetastet und als Zeitserien gespeichert. Mittels heuristischer

Bestimmung der Sensor-Motor-Koordinationspunkte wird der Datenstrom in Episoden zerteilt. Jeweils zueinandergehörige Episoden aus unterschiedlichen Demonstrationen werden anschließend normalisiert und gemittelt, die zuletzt in generalisierte Bewegungen zur Ausführung verfügbar gemacht werden.

Ein System zum Lernen von Handhabungungen im Haushalts- und Werkstattbereich stellt [Ehrenmann et al. 2002] vor. Zusätzlich zu dem vorherigen vorgestellten Roboter ist es zur Lokomotion mittels einer radgetriebenen Plattform in der Lage, so daß es seinem Benutzer zur Erledigung kooperativer Aufgaben folgen kann. Nach einer mit unterschiedlichen Sensoren erfolgenden Observationsphase werden die Demonstrationen segmentiert, indem gezielt nach Griffen sowie Umkehrpunkten in Trajektorien gesucht wird. Die Segmente werden anschließend semantisch analysiert und in eine hierarchische Abstraktion und Repräsentation des Handlungswissen überführt. Aus der semantischen Analyse kann zum Ausführungszeitpunkt bestimmt werden, ob das Handlungswissen in einem bestimmten Kontext anwendbar ist, zum gewünschten Ziel führt und welche Vorbedingungen zur Ausführbarkeit vorhanden sind bzw. erst noch geschaffen werden müssen. Die Abbildung von dem im menschlichen Handlungsraum repräsentierten Handlungswissen in den senso-motorischen Ausführungsraum des Roboters Albert (siehe Abbildung 2.21 (a)) erfolgt mittels eines regelbasierten Expertensystems. Vor der Ausführung kann das gelernte Handlungswissen simuliert und dem Benutzer visualisiert werden, um interaktiv Änderungen und Verbesserungen vornehmen zu können.

Einen humanoiden Roboter zum Erlernen von Tanzbewegungen präsentieren [Nakaoka et al. 2003] und [Shiratori et al. 2004]. Eine Tänzerin wird mittels eines Bewegungserfassungssystems bei Bewegungen eines japanischen Volkstanzes „Jongara-bushi“ beobachtet. Anschließend wird eine symbolische Repräsentation extrahiert, welche aus unterschiedlichen Primitiven aufgebaut ist. Diese Primitiven bestehen aus essentiellen Posen in der Bewegung der Arme sowie Schrittmustern der Beine. Die Bewegungen der Arme werden aus den aufgezeichneten generiert. Hierbei müssen jedoch insbesondere die dynamischen Beschränkungen des Roboters im Auge behalten werden. Dazu wird die demonstrierte Trajektorie auf den in der Regel langsameren Roboter abgebildet, indem die Dynamik innerhalb der erlaubten Bereich direkt abgebildet wird, ausserhalb jedoch einfach mit dem maximal möglichen Wert abgeschnitten wird. Das dadurch bedingte Hinterherhängen der Trajektorie wird dadurch vermieden, daß auf dieselbe Weise die Trajektorie rückwärts simuliert wird und anschließend im entstehenden Korridor zwischen den beiden Trajektorien gemittelt wird.



(a) (b) (c)

Abbildung 2.21: Service- und humanoide Roboter. (a) Serviceroboter Albert mit 6 Freiheitsgraden im Arm einer mobilen Plattform. (b,c) Humanoider HOAP2-Roboter beim Durchführen unterschiedlicher Tänze. Quellen: (a) [Ehrenmann et al. 2002] (b,c) [Nakaoka et al. 2003].

2.5 Psychologische Theorien zum Handlungslernen

In den vergangenen Abschnitten wurde eine Vielzahl technischer Systeme zum Erlernen von Handlungswissen vom Menschen und ihre Funktionsweisen vorgestellt. Dabei sind auch die Grenzen und Beschränkungen dieser Verfahren deutlich geworden. Um diese zu überwinden, erscheint es sinnvoll, die Fähigkeiten des Menschen auf dem Gebiet des Erlernens und Erschließens von Handlungswissen aus Demonstrationen und sozialen Interaktionen genauer zu betrachten und die dabei angewandten Methoden zu analysieren.

Die psychologischen Theorien zur Akquisition neuen Handlungswissens entstanden in einem klassischen Prozeß Hegelscher Dialektik (siehe [Hegel 1817]). Sie entstanden evolutiv aus dem Dreischritt zwischen These und Antithese hin zur Synthese, wobei jeder einzelne Schritt eine bestimmte wissenschaftliche Schule und damit verbundene Grundannahmen und Verfahren der Evaluation und experimentellen Validierung nach sich zog. Zum Verständnis jeder einzelnen Denkschule ist es notwendig, alle drei Standpunkte gründlich zu erörtern, um gegenseitige Abgrenzungen und Unterschiede überhaupt verstehen und würdigen zu können.

Die *dispositive* Hypothese, geht davon aus, daß Kognition und Handlungen entscheidend determiniert sind von Motivationskräften in Form von Impulsen, Bedürfnissen und Trieben, die allesamt unter der Bewußtseins-

schwelle angesiedelt sind. Die Beweggründe für Handlungen, sowie die den Handlungen zugrundeliegenden Modelle und Annahmen stammen allesamt aus Kräften und Prozessen innerhalb des Individuums selbst und ist abgekoppelt von äußeren Einflüssen. Kognitionsfähigkeit entwickelt sich aus ihren eigenen Grundlagen, und nicht aus externen Einflüssen [Spelke et al. 1992].

Die Kritik an der dispositiven Hypothese ist begrifflicher wie empirischer Art: Innere Determinanten wurden häufig aus dem Verhalten gefolgert, das sie angeblich bewirkten. So wird als Erklärung von Verhalten angesehen, was eigentlich lediglich Beschreibung von Verhalten war. So läßt sich beispielsweise aus Leistungsverhalten ein starkes Leistungsmotiv ableiten und ähnliche Zirkelschlüsse.

Weiterhin sind die Triebtheorien nicht ausreichend in der Lage, Voraussagen über die Zukunft zu treffen. Sie liefern saubere Deutungen vergangener Ereignisse, sind aber kaum zur Vorhersage von Verhalten geeignet [Bandura 1979]. Nun hängt die Stichhaltigkeit einer Theorie nicht so sehr davon ab, ein Geschehen rückschauend zu erklären, sondern in ihrer präzisen Angabe von Bedingungen, die die psychologischen Erscheinungen bestimmen, und wie genau sie die Mechanismen benennen kann, kraft derer sich die Einflußfaktoren auswirken. Auch in ihrer Operationalisierbarkeit schneiden solche Theorien eher schlecht ab, d.h. ihre Erkenntnisse lassen sich nicht in Verfahren und Prozeduren umsetzen, die ein bestimmtes Verhalten mit hoher Wahrscheinlichkeit zur Folge haben. Als man beispielsweise tatsächliche Verhaltensänderungen von Menschen, die ein Training basierend auf dispositiven Ansätzen absolviert hatten, mit den Veränderungen nicht trainierter Personen aus einer Kontrollgruppe verglich, ließen sich keinerlei statistisch signifikanten Unterschiede nachweisen [Bandura 1969].

Aufgrund dieser Tatsachen geriet die Kausalanalyse unbekannter innerer Determinanten für menschliches Verhalten in der weiteren Entwicklung der Verheltens- und Lerntheorie ins Hintertreffen. Seither hat man den externen Einflüssen und Stimulusbedingungen für menschliches Verhalten größeres Interesse gewidmet. Wiederholt konnten Forscher nachweisen, daß bestimmte Verhalten, die davor lediglich internen Motivationen zugeschrieben wurden, durch externe Einflüsse induziert, eliminiert und wiederhergestellt werden können. Dies gipfelte in der Formulierung der These von der *situationsbedingten* Verhaltensdetermination, wonach die Ursachen für bestimmte Handlungen nicht innerhalb des Individuums, sondern ausschließlich in der Umwelt zu suchen seien.

Als berühmteste Belege für die situationsbedingte Verhaltensbestimmung bzw. Reiz-Reaktionstheorie, gelten die von Pawlow an Hunden gemachten Untersuchungen (siehe [Pawlow 1972]). Iwan Pawlow entdeckte in den 20er Jahren bei Untersuchungen zu Speichelsekretion von Hunden den Mechanis-

mus des sog. klassischen Konditionierens. Der Organismus reagiert auf einen unkonditionierten Reiz (die Bereitstellung von Futter) reflexartig mit der Produktion von Speichel zur Auslösung des Verdauungsprozesses. Ein initial neutraler Reiz (ein Glockenton) löst für sich noch keine Speichelsekretion aus. Wird jedoch der neutrale Reiz gleichzeitig mit dem unkonditionierten Reiz gegeben und dies mehrfach wiederholt, so tritt bereits auf den neutralen Reiz die Reaktion auf, d.h. beim Läuten der Glocke reagiert der Hund mit erhöhter Speichelsekretion, sogar, wenn dies nicht mehr mit einer Gabe von Futter verbunden ist. Der anfangs neutrale Reiz ist zum konditionierten Reiz, die reflexartige Reaktion zum konditionierten Reflex geworden. Anschließende Untersuchungen zeigten, daß diese konditionierten Lernerfahrungen auch auf den Menschen übertragbar sind. Auch der umgekehrte Fall funktioniert: Es kann nicht nur durch Belohnung das Auftreten eines bestimmten Verhaltens bedingt werden, sondern durchaus auch durch Bestrafung die Veranlassung entstehen, ein bestimmtes Verhalten zu unterlassen.

[Bandura und Mischel 1965] beschreibt die Tatsache, daß Lernen nicht nur aus eigener Erfahrung bei der Durchführung eines bestimmten Verhaltens erfolgen kann, sondern auch durch sogenannte Rollenmodelle getrieben wird. Verhaltensmodifikation bzw. Verhaltenserwerb kann auch aus der Beobachtung der Konditionierung eines anderen Individuums, d.h. aus der Erfahrung, daß eine andere Person für bestimmtes Verhalten belohnt oder bestraft wird, resultieren. Dieser Prozeß des Lernen durch stellvertretende Belohnung/Bestrafung wird Beobachtungslernen oder Nachahmungslernen[11] genannt. Zwar existieren Schwierigkeiten, diesen Effekt im Humanversuch eindeutig nachzuweisen, weil vermieden werden muß, daß das angenommene Verhalten bereits einmal vorher gezeigt wurde. Die individuelle Lerngeschichte läßt sich jedoch bei Primaten, die in Zoos aufgewachsen sind, steuern. Beispielsweise konnte nachgewiesen werden, daß im Zoo aufgewachsene Rhesusaffen erst mit Angst auf eine Schlange reagierten, nachdem sie ihre Eltern bei einer Angstreaktion nach dem Reiz des Anblicks einer Schlange beobachten konnten. Wenn das geschah, erfolgte der Lernprozeß sehr rapide, es genügte bereits eine einzige Beobachtung, um das Verhalten dauerhaft zu erlernen. Hier wird auch die evolutionär absolute Notwendigkeit von Nachahmungslernen deutlich sowie ihre Überlegenheit bezüglich Konditionierungslernen aus eigenem Erleben heraus: In der hier gestellten Lernaufgabe ist eigenes Ausprobieren unmöglich, da eine einzige falsche Reaktion auf das Auftreten einer Schlange bereits den Tod und damit den abrupten Abbruch des Lernvorgangs zur Folge haben kann.

Diesen lerntheoretischen Ansätzen liegt gemeinsam zugrunde, daß letzten

[11] engl.: imitation learning

Endes nur die Reizbedingungen der Umwelt (ob direkt erfahren oder an einem Modell beobachtet) über das Lernen bzw. nicht-Lernen eines Verhaltens entscheiden. Nach diesem Lerngesetz wird es nach behavioristischer Auffassung möglich, durch klassischer Konditionierung oder Nachahmungslernen menschliches Verhalten durch Schaffung entsprechender Umweltbedingungen beliebig zu manipulieren.

Die Vorstellung, daß menschliches Verhalten extern gesteuert wird rief heftige Gegenreaktionen hervor. Kritik an der behavioristischen These wurde vor allem an der asymetrischen Auslegung empirischer Experimente laut: Der Experimentator kontrolliert die Umwelt des Lernenden. Daß dies tatsächlich so ist, läßt sich jedoch nur unter großen Schwierigkeiten begründen. Behavioristische Lernexperimente lassen sich auch gut umgekehrt deuten, nämlich daß eigentlich das Studienobjekt den Experimentator konditioniert: Immer wenn der Lerner das gewünschte Verhalten zeigt (das man durchaus auch als Reiz für den Experimentator deuten kann), teilt der Experimentator Belohnung aus (was auch als eine konditionierte Reaktion gesehen werden kann). Aus dieser Sichweise wurde die Richtung der Konditionierung umgedreht. Dies offenbart eine wesentliche Schwäche behavioristischer Forschung: Planvolles Handeln, das über die aktuelle Situation mit ihrer konkreten Reizkonfiguration hinausgeht, ist nur schwer zu erklären.

Die Beobachtung, daß Lebewesen in der Lage sind, aus der höchst umfangreichen Reizsituation diejenigen Reize zu selektieren, die in Bezug auf die aktuelle Situation relevant sind, und sie von der Masse an irrelevanten Situationsaspekten zu unterscheiden, führte letztlich zur Auflösung der Dispositionismus-/Situationismus-Kontroverse. [Baum 1973] stellte beispielsweise fest, daß Verhalten weniger durch unmittelbare Auswirkungen auf Aktionen, sondern durch ein integriertes Feedback gesteuert wird. Demnach betrachten Lebewesen Daten darüber, wie oft ihre Reaktionen im Lauf langer Zeiträume positiv verstärkt wurden. Entsprechend der Ergebnisse solcher Betrachtungen und der kumulativen Konsequenzen steuern sie dann ihr Verhalten.

Eine solche Integration von Ergebnissen über lange Zeiträume hinweg setzt kognitive Fähigkeiten voraus, die weder durch innere Kräfte noch von Umweltstimuli allein angestoßen werden. Die psychologische Lernfunktion wird hingegen durch Wechselwirkungen von inneren, persönlichen Determinanten mit externen Einflüssen erbracht. Sie erlaubt es, daß der Beobachter das vom Modell vorgeführte Verhalten in seiner Absicht erkennt, und nicht bei seiner Nachahmung nur zufällige Aspekte des Vorgeführten reproduziert unter Auslassung des Wesentlichen. Ein oft bemühtes Beispiel ist das eines Roboters, der lernen kann, die Bewegungen eines Schachspielers zu imitieren, der aber bei weitem nicht in der Lage ist, die Spielzüge zu reproduzieren.

[Piaget 1975] formulierte den menschliche Lernprozeß als kontinuierliche Adaption von Wissen. Diese Adaption des Wissens erfolgt angestoßen durch externe Einflüsse unter Rückgriff auf bereits internalisiertes Wissen. Demnach ist die gesamte menschliche Entwicklung von den ersten Anfängen bis zur Adoleszenz und darüber hinaus ein lebenslanger Lernprozeß. Hieraus resultiert, daß Lernen ein iterativer Prozeß ist, der stets auf Ergebnisse früherer Lernschritte zurückgreift und diese überprüft, verändert, optimiert oder verwirft. Dieser ist unterteilt in drei nicht notwendigerweise streng sequentielle Unterprozesse: Aufmerksamkeitsprozesse, Behaltensprozesse und Reproduktionsprozesse.

Aufmerksamkeitsprozesse entscheiden darüber, was aus der Fülle der Umwelteinflüsse selektiert und wahrgenommen wird. Diese stellen sicher, daß auch die wichtigen Merkmale des zu Modellierenden exakt aufgenommen werden. Die Selektion dieser Merkmale basiert auf bereits vorhandenem Wissen, wie es beispielsweise aus Eigenschaften des Beobachters, Eigenarten der Tätigkeit selbst oder aus vorangegangener oder gleichzeitiger menschlicher Interaktion gewonnen werden konnte. Für gute Lernergebnisse ist es insbesondere nötig, daß das Modell die Aufmerksamkeit des Beobachters über einen ausreichend langen Zeitraum zu fassen vermag.

Wenn ein Mensch sich nicht an die Beobachtung eines bestimmten modellierten Verhaltens erinnern kann, wird kein Einfluß von diesem ausgehen. Deshalb ist ein weiterer wesentlicher am Lernen nach Piaget beteiligter Prozeß das Speichern der Handlungen, die irgendwann einmal modelliert wurden. Dieser *Speicherungsprozess* stellt sicher, daß die Reaktionsmuster und das Verhalten des Modells in einer ausreichend von der konkreten Situation abstrahierten, symbolischen Art und Weise im Gedächtnis verankert sind. Gerade die symbolische Natur der Wissensinhalte ist hier von herausragender Bedeutung, da nur aufgrund hochentwickelter Symbolisierungsfähigkeit Menschen in der Lage sind, große Teile ihres Verhaltens aus Beobachtung zu lernen. Höhere kognitive Prozesse finden eher auf einer verbalen als auf einer direkt sinneseindrücklichen, bspw. der visuellen Ebene statt. Dies wird zum Beispiel deutlich an der Aufgabe, Einzelheiten des Weges, den ein Modell eingeschlagen hat, zu erlernen. Eine sprachliche Kodierung in eine Folge von Rechts- und Linkswendungen läßt sich besser aneignen, leichter behalten, effektiver generalisieren und später genauer reproduzieren, als wenn man sich direkt auf die visuelle Vorstellung des Weges verläßt. Beobachtungslernen wird durch symbolische Codierungen deutlich erleichtert, wenn nicht sogar erst ermöglicht, weil sie große Informationsmengen in leicht zu speichernder Form liefern.

Reproduktionsprozesse betimmen zuletzt, wie die symbolischen Repräsentationen in angemessene Handlungen umzusetzen sind. Nur selten lassen sich

Lerninhalte beim ersten Versuch ohne Fehler in richtige Handlungen umsetzen. Angemessene Nachbildungen lassen sich gewöhnlich nur durch Korrekturen an den ersten Versuchen erreichen. Die Diskrepanz zwischen symbolischer Repräsentation und der Beobachtung des Ergebnisses ist ein erster Anhaltspunkt für die Korrektur. Ein Problem stellt hierbei die Tatsache dar, daß beim Erlernen einiger Fähigkeiten, wie zum Beispiel des Schwimmens, die Reaktionen nicht richtig beobachtet werden können und der Organismus sich daher auf vage kinästhetische Hinweise oder Rückmeldungen eines Beobachters verlassen muss. Es treten Schwierigkeiten auf, Handlungen zu steuern, die man selbst nur teilweise beobachten kann.

Der wesentliche Beitrag von Piaget's Lerntheorie besteht darin, daß Lernen zwar auf externe Reize und Hinweise angewiesen ist, ohne die eine Entwicklung nicht möglich ist, andererseits interne Prozesse und Gedächtnisinhalte, die aus vorherigen Lernprozessen resultieren können, einen mindestens ebenbürtigen Einfluß auf erfolgreiches Lernen haben. Lernen findet also nie im leeren Raum statt, sondern baut auf bereits gelerntes Wissen auf.

2.6 Zusammenfassung

Die letzten Abschnitte gaben einen Überblick über vergangene und aktuelle Arbeiten zum Lernen von Handlungswissen in technischen Systemen und beim Menschen. So atemberaubend die Lernergebnisse in technischen Systemen auch sein mögen, so wenig können sie doch mit dem Lernvermögen von menschlichen Organismen mithalten. Deren Überlegenheit manifestiert sich im lebenslangen Lernen, in der massiven Verwendung von Vorwissen zum interpretieren, bewerten und verallgemeinern neuer Erfahrungen, der abstrakten und höchst flexiblen symbolischen Repräsentation von Wissensinhalten wie -struktur und der kontinuierlichen Adaption und Optimierung vorhandenen Wissens.

Der Überblick über verschiedene lernende Robotersystem läßt es dringend notwendig erscheinen, Methoden zur abstrakten Modellierung hierarchischen Handlungswissen, welches die subsymbolischen Ebenen hinter sich läßt, Methoden zum lebenslangen Lernen, welche einen gesamten Erfahrungsschatz des Robotersystems aufbauen, sowie Methoden zur inkrementellen Verbesserung und Vervollständigung des Handlungswissens zu formulieren, analysieren und implementieren. Sollen technische Robotersysteme ihren begrenzten Lernstrukturen entwachsen und ihre heutigen Grenzen hinter sich lassen, so ist eine Übernahme menschlicher Lernstrategien wie sie von der Verhaltens- und Entwicklungspsychologie erforscht und beschrieben werden ein erfolgversprechender Weg in eine Zukunft voller Chancen und Möglichkeiten.

Kapitel 3

Programmieren durch Vormachen als Kommunikationsprozess

Der vorangegangene Abschnitt zur Lernpsychologie hat gezeigt, daß menschliches Lernen großen Nutzen aus Demonstrationen zum Fähigkeitserwerb zieht bzw. teilweise erst durch diese ermöglicht wird. Ferner wurden erste Implementierungen von technischen lernenden Systemen vorgestellt sowie ihre jeweiligen Eignungen und Limitationen detailliert beschrieben. Sie alle profitieren in verschiedener Art und Weise vom Vorhandensein von Demonstrationen. Diese verwenden sie in unterschiedlichster Form zur Replikation von Verhalten oder Handlungen. Weiterhin ist all diesen Verfahren gemein, daß sie die Möglichkeiten, die sich einem Menschen bieten, der dieselben Handlungen observiert, genausowenig ausschöpfen wie sie dessen Flexibilität in Bezug auf die Bandbreite der lernbaren Aufgaben auch nur bruchstückweise erreichen.

So wird gerade an der Betrachtung des aktuellen Standes der Entwicklung im vorangegangenen Kapitel der weitere Forschungsbedarf deutlich. Im folgenden wird ein Ansatz vorgestellt, der sowohl die Hardwareunabhängigkeit auf sensorischer wie auf aktorischer Seite, als auch den Wegfall unnötig einengender Beschränkungen bei der Klasse der zu lernenden Aufgaben zum Ziel hat. Der Ansatz integriert dabei verschiedene Verfahren und Algorithmen der in Abschnitt 2 vorgestellten Systeme sowie die ebenda erläuterten Erkenntnisse der Lern- und Entwicklungspyschologie.

Das folgenden Kapitel beginnt mit der Präzisierung der Anforderungen und Randbedingungen, denen das beschriebene System genügen muß. Anschließend wird der Prozeß des Programmierens durch Vormachen detailliert vorgestellt. Im darauf folgenden Abschnitt werden einige grundlegenden Be-

griffe definiert sowie die gewählte Repräsentation von Handlungswissen als Makro-Operatoren vorgestellt. Zuletzt wird ein Überblick über die gewählte Systemarchitektur und die beteiligten Teilkomponenten gegeben.

3.1 Anforderungen und Randbedingungen

Eine akkurate Analyse der Aufgabe der Übertragung menschlichen Handlungswissens auf ein Robotersystem ergibt folgenden Anforderungskatalog:

- Die Demonstration der Handlungen sollte so *natürlich* wie möglich erfolgen. Dazu ist es erforderlich, den Menschen nur so weit wie unbedingt notwendig in seiner Handlungs- und Bewegungsfreiheit einzuschränken, wie es die adäquate sensorische Erfassung der Handlung erfordert. Aufwendige Präparations- und Kalibrationsphasen sind hierbei nach Möglichkeit zu umgehen oder auf ein Minimum zu reduzieren.

- Die Verarbeitung und Interpretation der beobachteten Demonstrationen soll weitestgehend *automatisiert* erfolgen. Dies trägt der Tatsache Rechnung, daß ein hohes Maß an Interaktion mit dem System während des Lernprozesses die Benutzerfreundlichkeit mindern würde. Eine Folgerung der Automation ist, daß die sensorisch erfassten Daten die für das Verständnis und die Nachahmung einer Aufgabenlösung relevanten Informationen enthalten.

- Dennoch soll das Handlungswissen in einer dem Benutzer verständlichen Form *erklär- und darstellbar* sein, um Benutzerinteraktionen auf dessen Wunsch zu ermöglichen und erleichtern. Diese Forderung ergibt sich aus der Tatsache der endlichen Intelligenz computerisierter Maschinen, die eine obere Komplexitätsschranke für die zu verarbeitenden Problemstellungen darstellt. Sollte diese einmal überschritten werden, so soll das System in der Lage sein, die getroffenen Hypothesen zu kommunizieren, zu visualisieren und auf Eingriff des Menschen hin zu adaptieren.

- Das Lernende System sollte *sensorunabhängig* funktionieren, d.h. keine dedizierte Sensorik benötigen, sondern nur abstrakte, logische Sensoren erfordern. Diese Forderung trägt dem Umstand Rechnung, daß auf unterschiedlichen Robotern und in unterschiedlichen Demonstrationsumgebungen durchaus verschiedenartige und unterschiedlich akkurate Sensorik vorhanden sein kann. Das Lernproblem, dem sich das System ausgesetzt sieht, stellt sich jedoch stets auf die gleiche Weise dar und ist jeweils mit den gerade zur Verfügung stehenden Ressourcen zu lösen.

- Benutzerfreundlichkeit erfordert ein reaktives System, welches in *Echtzeit* lernen kann. Diese Fähigkeit dient dazu, dem Benutzer stets Rückmeldung über seinen internen Systemzustand sowie den erreichten Lernfortschritt zu geben. Lange Reaktions-, Warte- und Lernzeiten mindern den Nutzen von Robotern in Alltagssituationen.

- Das System soll in der Lage sein, bei der Erfassung der Benutzerintention *kontextabhängig* zu reagieren. Je nach aktuellem Handlungskontext sind geeignete Aktionen auszuwählen, die aber unterschiedliche Bedeutungen in Bezug auf das Handlungsziel aufweisen. Ein Beispiel hierfür ist das Ergreifen eines Stiftes. Während der Handlung „Aufräumen“ kommt dieser Aktion eine völlig andere Semantik in Bezug auf die Vor- und Nachbedingungen der Aktion sowie des weiteren Vorgehens zu als im Kontext der Aufgabe “Schreiben“. Diese Forderung bedingt das Vorhandensein präziser und meist umfangreicher Kontext- und Semantikmodelle.

- Bereits nach der ersten Demonstration einer Handlung sollte ein Lernfortschritt sichtbar sein. Deshalb ist es sinnvoll, das Lernen *inkrementell* zu gestalten. Gerade bei Aufgaben, deren Ziele und relevante Merkmale nicht vollständig aus einer einzelnen Demonstration zu erschließen sind, ist es von großer Bedeutung, daß der Benutzer den Lernfortschritt des Systems stets verläßlich einschätzen kann. Dies sollte den Benutzer nur mit einer minimalen notwendigen Anzahl an Wiederholungen belasten. Das System soll also in der Lage sein, bereits aus wenig vorhandener Information ein initiales Modell der zu bearbeitenden Aufgabe zu erstellen und dieses in einem sukzessiven Prozeß der Fortentwicklung zu optimieren und der Intention des Benutzers anzunähern.

- Aus der letzten Anforderung resultiert das Bedürfnis nach einer *skalierbaren* und *wiederverwendbaren* Repräsentation des Handlungswissens in einer Handlungsdatenbasis, die es erlaubt, das Wissen zu speichern, abzurufen, zu generalisieren und für andere Teilkomponenten eines Robotersystems zu deren Gebrauch vorzuhalten. Dabei sollte das Handlungswissen weitgehend unabhängig von der vorliegenden Roboterarchitektur, -kinematik und -dynamik sein. Dies ist begründet in den in der Regel auftretenden Versions- und Technologieweiterentwicklungen bei der Aktorik und Roboterhardware. Ein Neu-Einlernen der Aufgaben und des Handlungswissens sollte dabei vermieden werden, damit die Akzeptanz eines solchen Systems nicht gemindert wird.

- Um einen Nutzen aus dem Lernprozeß ziehen zu können, sollte das

gelernte Handlungswissen auf eine Vielzahl von Robotersystemen *abbildbar* sein. Diese Forderung impliziert eine Festlegung auf eine bestimmte Bandbreite an Roboterhardware. So sollten die Zielsysteme über visuelle und haptische Sensorik verfügen, die einigermaßen den menschlichen perzeptiven Fähigkeiten ähnlich sind, sowie in Bezug auf Dynamik, Nutzlast und Steifigkeit die selben physikalisch Größenordnungen einnehmen. Die Transformation des Wissens auf ein spezielles Robotersystem bedarf unterschiedlicher Optimierungen, Adaptionen und Einschränkungen, welche mittels geeigneter Regelwerke und Erfahrungswerte in die Wissensbasis integriert werden können.

3.2 Der PdV-Prozess

Die Abbildung einer demonstrierten Handlung auf ein ausführbares Roboterprogramm erfolgt mittels eines komplexen und verzahnten Prozesses. Er kann als eine Transformationsfunkion

$$P(K) = T(\overrightarrow{s}(t), \overrightarrow{w}(t), \mathbf{M}(t)) \tag{3.1}$$

angesehen werden. Diese Funktion bildet die über die Zeit t integrierten Sensormeßwerte $\overrightarrow{s}(t)$, den Weltzustand $\overrightarrow{w}(t)$ sowie den jeweils aktuellen Erfahrungshorizont, expliziert durch die Menge des Modellwissen $\mathbf{M}(t)$ auf das Roboterprogramm P ab, welches Abhängigkeiten bzw. notwendige Vorbedingungen aufweist, die im Kontext K zusammengefaßt sind.

Diese Transformationsfunktion kann auch als empfängerseitige Deutung einer Kommunikationssituation interpretiert und damit nach [Goos und Waite 1984] als eine in verschiedene, sukzessiv aufeinander aufbauende Phasen gegliederte verkettete Verarbeitung der Eingabedaten aufgefaßt werden. Dieser wird durchgeführt als eine Sequenz von Transformationen bzw. Übersetzungen $(\mathbf{S}, \mathbf{L}_1), (\mathbf{L}_1, \mathbf{L}_2), \ldots, (\mathbf{L}_k, \mathbf{P})$, wobei S die Menge aller möglichen Sensoreingaben, P die Zielprogrammiersprache des Robotersystems und $\mathbf{L}_1, \ldots \mathbf{L}_k$ Zwischensprachen darstellen. Es gilt also $\overrightarrow{s}(t) \in \mathbf{S}$ und $P(K) \in \mathbf{P}$. Drei große Phasen innerhalb dieser Transformationskette können unterschieden werden:

- *Analyse*: Bestimmung der Primitiven einer Demonstration, ihrer Struktur und ihrer Bedeutung zur Extraktion des relevanten Handlungswissens

- *Generalisierung*: Abstraktion der Demonstration über ihren eigenen Horizont hinaus unter Rückgriff auf anderes Handlungswissen

- *Synthese*: Erzeugung eines Programmes in der auf einem konkreten Roboter lauffähigen Zielsprache

Diese Einteilung ist sinnvoll, separiert sie doch die Quellsprache (Sensoreingaben) in der Analysephase von der Zielsprache (Roboterprogrammiersprache) in der Synthesephase. Der mittlere Teil, die Generalisierungssphase, operiert im Raum einer Zwischensprache, die das Handlungswissen kodiert. Diese Zwischenphase wird im Rahmen dieser Arbeit durch die sogenannten Makro-Operatoren realisiert (siehe Abschnitt 3.3).

Von diesen drei Phasen ist die Analysephase die am meisten formalisierbare Aufgabe. Sie ist generell untergliedert in drei unterschiedliche Teilphasen: die *lexikalische*, die *strukturelle* und die *semantische* Analyse. Lexikalische Analyse behandelt die elementaren Primitiven der Handlung, syntaktische Analyse kümmert sich um die statischen Struktur einer Demonstration und die semantische Analyse erschließt das Ziel und die Intention hinter einer Gesamthandlung.

Ein wichtiger Nebeneffekt der Strukturierung des gesamten PdV-Prozesses als Prozesspipeline ist die Partitionierung der gesamten Modell- und Hintergrundwissensdatenbank $\mathbf{M}$ in paarweise disjunkte Teilmodelle, welche jeweils eindeutig einer bestimmten Phase bzw. Transformationsfunktion im Übersetzungsprozess zugeordnet sind. So ist beispielsweise das lexikalische Modell $\mathbf{M}_{lex}$ nur für die Transformation T_{lex} von Bedeutung, die den Sensoreingabestrom in die Zwischensprache des Primitivenalphabets abbildet. Weiterhin kann durch diese Phasenunterscheidung eine höhere Modularisierung erreicht werden, was gerade in Hinblick auf die Implementier- und Realisierbarkeit eines so komplexen Softwaresystems von hoher Bedeutung ist.

Neben der bereits benannten Transformationsfunktion T_{lex} existieren noch die Funktionen für die syntaktische (T_{syn}) und semantische (T_{sem}) Analyse sowie der Programmsynthese (T_{synth}) mit ihren jeweils dazugehörigen Modellen $\mathbf{M}_{syn}, \mathbf{M}_{sem}$ und $\mathbf{M}_{synth}$. Die Generalisierungsstransformation T_{gen} operiert auf der Handlungsdatenbank $\mathbf{M}_H$, welche alle gelernten Modelle enthält. Sie leistet die inkrementelle Abstraktion mittels Veränderung des jeweils betroffenen Verhaltens.

Abschließend läßt sich nun der in Gleichung 3.1 formulierte PdV-Prozess konkretisieren durch die Spezifikation der Transformationsfunktion als Verkettung der einzelnen im vorangegangenen definierten Übersetzungsfunktionen

$$T = T_{lex} \circ T_{syn} \circ T_{sem} \circ T_{gen} \circ T_{synth} \tag{3.2}$$

3.3 Makro-Operatoren als Repräsentation für Handlungswissen

Aus den in Abschnitt 3.1 formulierten Anforderungen an das System zum Programmieren durch Vormachen lassen sich unterschiedliche Forderungen an die Repräsentation des Handlungswissens ableiten. Diese umfassen vor allem seine Skalierbarkeit, Wiederverwendbarkeit, Sensorunabhängigkeit, Erklär- und Visualisierbarkeit, Abbildbarkeit und Erweiterbarkeit.

Handlungsaufgaben werden hierbei hierarchisch funktional aufgegliedert modelliert. Verglichen mit den verschiedenen Handlungsrepräsentationen wie sie beispielsweise in Abschnitt 2 vorgestellt wurden, kommen bei dieser Repräsentationsart ihre Vorteile besonders in den Bereichen der Skalier- und Erweiterbarkeit, Erklär- und Visualisierbarkeit zum Zuge. So beinhaltet der hierarchische Aspekt die Zerlegung jeder Aufgabe in eine endliche Zahl kleinerer Teilaufgaben. Dies führt zu einer benutzeradäquaten Repräsentation und Visualisierung, da man sich auf jeder Ebene nur auf einen geringeren Teil von Unterebenen konzentrieren muß. Der funktionale Aspekt besagt, daß die Zerlegung der Aufgabe in Teilaufgaben anhand der Teilziele der unterschiedlichen Aufgabenteile erfolgt. Dies ist insbesondere für die Erkennung des Handlungsziels von Bedeutung. Nur so kann der Abgleich zwischen der beobachteten Handlung und der Absicht des Benutzers nachvollzogen werden. Die hierarchisch gegliederte Handlungsrepräsentation unterstützt den Prozeß der Absichtserkennung durch Einbettung einer einzelnen Teilaufgabe in eine Struktur größerer Zusammenhänge, in einen Kontext. Sie erlaubt so die Parametrisierung und Steuerung der Algorithmen während der Beobachtung und erlaubt die Erkennung und Korrektur von Fehlern und Unstimmigkeiten während der Wissensakquisition. Besonders in diesen Fehlerfällen können solche Ausnahmesituationen zu Rückmeldungen und Rückfragen an den Benutzer genutzt werden, was es erlaubt, die Benutzerintention mit höherer Genauigkeit zu akquirieren.

Die nach diesen Prinzipien erfolgte Modellierung von Aufgaben ermöglicht deren symbolische Beschreibung als eine Folge von Teilaufgaben. Alle einer Teilaufgabe zugeordneten Symbole erhalten somit eine semantische Bedeutung und ermöglichen so die Bildung einer impliziten Begriffshierarchie. Eine hierarchisch-funktionale Repräsentation erleichtert darüber hinaus die Umsetzbarkeit auf ein Roboterprogramm, da sie den strukturellen Aspekt der „Divide-et-Impera“-Herangehensweise zur manuellen Erstellung von Programmen widerspiegelt.

Wie in Abbildung 3.1 dargestellt, wird eine Demonstration einer Aufgabe sukzessive in immer kleinere Teileinheiten zerlegt, bis die Ebene der Primiti-

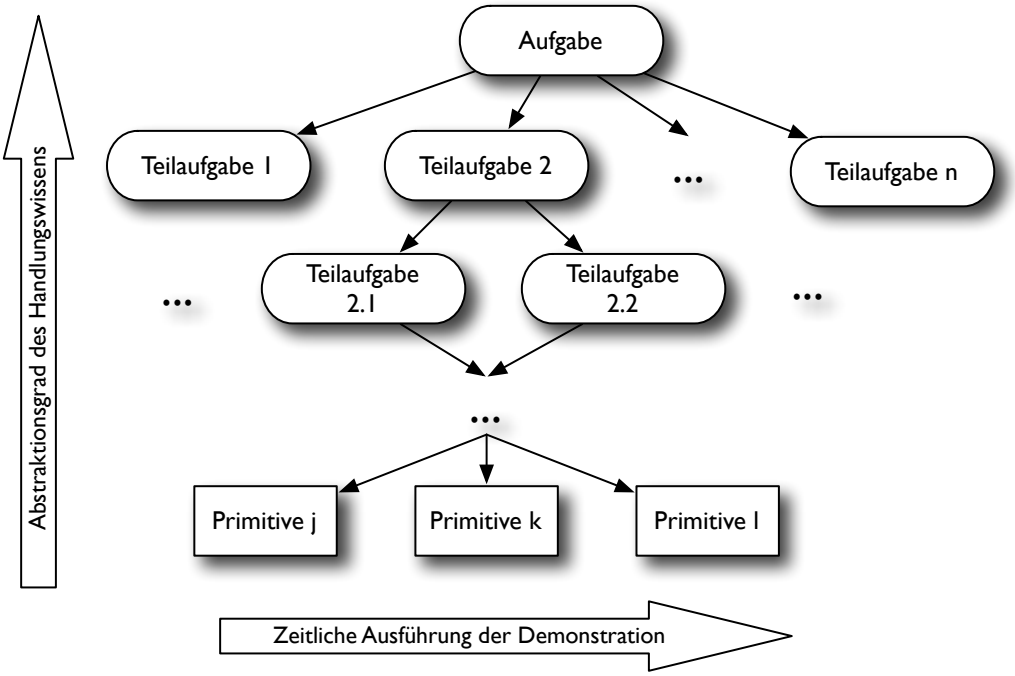


Abbildung 3.1: Hierarchisch-funktionale Aufgabenmodellierung

ven erreicht wird. Von jeder Ebene wird eine niedrigere, weniger abstrahierte Ebene beherrscht. Dabei stellt jede Ebene einer nächsthöheren Dienstleistungen zur Verfügung, die zu stärker abstrahierten Handlungen zusammengesetzt werden können.

Die unterste Ebene bilden die Primitiven. Bei der Auswahl dieser Primitiven ist besondere Sorgfalt notwendig, ist sie doch durch zwei unterschiedliche Anforderungen eingeschränkt. Zum einen soll jede einzelne Primitive auf einem Roboter ausführbar und doch gleichzeitig roboterunabhängig sein. Sie muß also eine ausreichend generelle, von konkreten Roboterkonfigurationen unabhängige, dennoch aber elementare Fähigkeit sein, daß sie auf jeder zur Lösung der Aufgabe in Frage kommenden Roboterkonfiguration verfügbar ist. Zum anderen muß sie mit vertretbaren sensorischen, algorithmischen und rechentechnischen Aufwand aus einer Demonstration extrahierbar sein. Ein Mensch muß also die einzelnen Primitiven im Sinne einer Vorführung oder prototypischen Demonstration ausführen können und die perzeptive Hardware muß diese beobachten können. Außerdem muß die Menge der Primitiven vollständig sein im Sinne der zu bewältigenden Aufgabenklassen und weiterhin noch orthogonal, d.h. Mehrfachbelegungen oder Replikationen anderer Primitiven mit nur marginalen Modifikationen sind zu vermeiden.

Angelehnt an [Zöllner 2005] und [Friedrich 1998] wird die Menge

der Primitiven definiert als eine Menge unterschiedlicher Bewegungs- und Griffprimitiven. Diese Wahl hat die Eigenschaften, daß alle Primitiven mit vertretbarem Aufwand sensorisch erfaßbar und auf Robotern bei geeigneter kinematischer Konfiguration ausführbar sind. Damit deckt sie die Menge aller Handlungen vollständig ab und ist zudem orthogonal.

Die Primitiven sind implementiert als sogenannte *Elementar-Operatoren (EO)* und man findet sie nur an den Blättern des aus der hierarchisch-funktionalen Repräsentation der Aufgabenlösung resultierenden Handlungsbaumes. Ihre formalisierte Definition sieht folgendermaßen aus:

Definition 1 (Elementaroperator) *Ein Elementaroperator ist die symbolische Darstellung der kleinsten Handlungselemente. Er weist im allgemeinen eine sehr enge Sensor-Aktor-Kopplung auf und muß daher spezialisiert auf die konkrete Roboterkonfiguration, auf der er ausführbar sein soll, implementiert sein. Die Effekte und Nachbedingungen von Elementaroperatoren bei gegebenen Umweltzustand sowie deren Erkennung aus menschlichen Demonstrationen sind jedoch roboterinvariant und hängen nur von der verwendeten perzeptiven Sensorik ab. Ein Elementaroperator besteht aus dem Tupel* $(N, \mathbf{P})$ *mit folgender Bedeutung:*

- N = *Name des Elementaroperators.*
- $\mathbf{P}$ = *Menge der Parameter des Elementaroperators.*
  Die Parameterliste kann beliebigen Umfang habe und Objekte, Positionen, Sensormeßwerte wie Bewegungsbahnen oder Fingerwinkel sowie beliebige andere Informationen enthalten.

Ein typisches Beispiel für Elementaroperatoren sind Bewegungen. Robotersysteme weisen in der Regel die Struktur einer kinematischen Kette auf, die eine ganze Reihe von Bewegungen zulassen, beispielsweise lineare Bewegungen zwischen zwei Punkten, Zirkular-, Polynombahnen oder das Abfahren einer Spline-Kurve. Wie diese jedoch genau umgesetzt werden, ist von Roboter zu Roboter unterschiedlich und vor allem auch deutlich verschieden von dem Arbeitsraum des Menschen. Deshalb sind elementare Bewegungsoperatoren prinzipiell vom Menschen auf einen Roboter übertragbar, erfordern aber einen erheblichen Aufwand zur Umplanung oder Anpassung der konkreten Trajektorie. Die konkrete Abbildung wird von einer roboterabhängigen Implementierung des Operators umgesetzt.

Die Menge der Bewegungsoperatoren abstrahiert die Eigenschaften des Zielrobotersystems und schafft die Grundlage für die Aggregierung weiterer, komplexerer Handlungsen. Sie erlaubt, daß auch beliebig komplex zusammengesetzte Handlungen im Prinzip auf jedem Roboter mit den genannten

Einschränkungen ausführbar sind. Eine notwendige Voraussetzung für ein erfolgreiches Lernen einer bestimmten Aufgabe ist daher, daß sich die Aufgabe auch unter Nutzung des Befehlssatzes der Elementaroperatoren lösen läßt.

Die Operatoren der darüberliegenden Schichten werden als Elemente einer Operatormenge definiert, die an die STRIPS-Repräsentation (siehe [Fikes et al. 1972]) angelehnt ist und wie folgt repräsentiert ist:

Definition 2 (Makro-Operator) *Ein Makro-Operator ist die symbolische Darstellung von nicht-elementaren Handlungselementen, welche Teilaufgaben oder die gesamte Aufgabe beschreiben. Ein Makro-Operator wird durch das Tupel* $(N, \mathbf{P}, \mathbf{O}, \mathbf{S}, \mathbf{P}r, \mathbf{V}, \mathbf{E})$ *beschrieben, welches folgendermaßen definiert ist:*

- N = *Name des Makro-Operators.*
- $\mathbf{P}$ = *Menge der Parameter des Makro-Operators. Änlich wie bei den Elementaroperatoren kann diese Liste beliebig lang sein und jede erdenkliche Art von Parametern enthalten, die für die Teilaufgabe von Relevanz sind. Dies können beispielsweise Wiederholungs- oder Abbruchbedingungen für untergeordnete Instanzen sein.*
- $\mathbf{O}$ = *die Menge aller Objekte die während des Geltungsbereiches des Makro-Operators von Bedeutung sind, beispielsweise manipulierte Objekte oder referenzierte, d.h. Objekte, die zur Lagebestimmung und Navigation anderer Objekte verwendet werden.*
- $\mathbf{S}$ = *die Menge aller Nachfolger, das heißt der Makro- oder Elementaroperatoren, die eine Teilaufgabe des Makro-Operators erledigen.*
- $\mathbf{P}r$ = *die Menge aller Vorrangbeziehungen zwischen den Nachfolgern (für eine genaue Definition der Vorrangbeziehungen siehe Abschnitt 6.2). Die Vorrangbeziehungen regeln die Abfolge der Lösung von Teilaufgaben.*
- $\mathbf{C}$ = *die Menge aller Abbruch- oder Stopbedingungen für den Makro-Operator. Die Ausführung eines Makro-Operators wird sofort unterbrochen, sobald eine der Bedingungen in* $\mathbf{C}$ *erfüllt ist.*
- $\mathbf{V}$ = *die Menge der Vorbedingungen, die alle erfüllt sein müssen, bevor die Ausführung des Makro-Operators starten kann. Die Ausführung eines Makro-Operators in einem Weltzustand, der diese Vorbedingungen nicht erfüllt, ist nicht definiert.*

- **E** = *die Menge aller Nachbedingungen oder Effekte eines Makro-Operators. Die Menge aller Effekte ist die Änderung des Weltzustandes, die auftritt, wenn der Makro-Operator in einem Weltzustand, der* **V** *erfüllt, korrekt ausgeführt wird.*

Alle Operatoren oberhalb der Ebene der Primitiven sind als Makro-Operatoren repräsentiert und stellen so eine allgemeine hierarchische funktionale Repräsentation von Handlungen im Manipulationsbereich dar. Diese Aktionen können in den unterschiedlichsten Domänen implementiert und angewendet werden, wie zum Beispiel bei Servicerobotern in Haushaltsumgebungen, die als Haushaltshilfen eingesetzt werden.

3.4 Hintergrund- und Umweltwissen im PdV

Das dieser Arbeit zugrundeliegende System zur Extraktion von Elementar-Operatoren aus Sequenzen von Weltzuständen erfordert eine vollständige Modellierung der Umwelt. Dabei wird angenommen, daß alle Veränderungen in der Umwelt auf Handlungen des Benutzers zurückgehen (Annahme einer geschlossenen Welt[1]). Die vorgeschlagenen symbolischen Verfahren nutzen ausschließlich die zeitliche Veränderung des Status der Umwelt, wie er in dem Umweltmodell repräsentiert wird.

Die Modellierung der Umwelt und aller in ihr enthaltenen Objekte wie Individuen erfolgt auf vier unterschiedlichen Ebenen: Der subsymbolisch-geometrischen sowie den semantisch höherwertigen relationalen, physikalischen und funktionalen Ebenen.

- *Geometrisches Modell:* Das geometrische Modell beschreibt die geometrische Struktur des Objekte, in der Regel durch Oberflächendefinitionen wie z.B. Dreiecksnetze. Lageinformationen von Objekten werden in Form homogener Transformationen zwischen dem lokalen Objekt- und dem globalen Basiskoordinatensystem angegeben.

- *Relationales Modell:* Das relationale Modell beschreibt aus geometrisch relativen Lagen der Objekte abgeleitete Attribute, die einen gesamten Umweltzustand beschreiben. Diese liegen in symbolischer Form vor und wurden so bestimmt, daß sie in etwa eine verbale menschliche Beschreibung der Szene wiederspiegeln.

- *Physikalisches Modell:* Das physikalische Objektmodell beschreibt die objektspezifischen Eigenschaften wie Gewicht, Schwerpunkt, Symme-

[1] engl.: Closed-World-Assumption

trieachsen, Kräfte, Momente oder Ablageflächen durch Paare von Attributen und zugeordneten Werten.

- *Funktionales Modell:* Das funktionale Modell beschreibt die über das physikalische Modell hinausgehenden Eigenschaften von Objekten, besonders in Hinblick auf die Verwendbarkeit von Objekten.

Die geometrische Visualisierung der Objekte und der Umwelt erfolgt in einer auf OpenInventor basierenden Simulationsumgebung *KaVis*, wie sie in [Schaude 1996] detailliert beschrieben ist. Die Vorteile dieser Komponente liegen in der Importierbarkeit von Daten, wie sie herkömmliche CAD-Programme erzeugen. Weiterhin ermöglicht die Simulationsumgebung die Visualisierung und Positionierung von Objekten, Erkennung von Kollisionen und Berührungen sowie die Darstellung zurückgelegter Trajektorien.

Das relationale Modell wertet auf die geometrisch korrekte Visualisierung aufbauend sogenannte „Semantische Variablen“ aus, die einer natürlichsprachlichen Beschreibung der relativen Objektlagen entspricht. Beispielsweise entspricht die Relation $Auf(Tasse, Untertasse)$ einer Umweltkonfiguration, in welcher sich eine Tasse auf einer Untertasse befindet. Derartige Aussagesätze bzw. die entsprechenden Relationen ermöglichen eine abstrakte Beschreibung von Umweltzuständen in Vor- und Nachbedingungen von Makro-Operatoren. Der Grund für die Wahl dieser Repräsentation als Ausgangspunkt für Lernalgorithmen liegt in der generellen, symbolischen Art und Weise der Darstellung von Lagebeziehungen. Sie abstrahiert die von Varianzen in der Positionierung von Objekten, wie sie beispielsweise durch sensorisches Rauschen entstehen können. Des weiteren gleicht sie die Informationsreduktion aus, die durch den Übergang von geometrisch akkurater Lagebeziehungen hin zu symbolischer Beschreibung auftritt. Zuletzt befreit sie die Lernalgorithmen von unnötiger Datenlast und sorgt für leichtere Erklär- und Explizierbarkeit, wie sie eine Darstellung bietet, die an menschliche natürlichsprachliche Beschreibungsformen angelehnt ist. Die räumlichen Relationen werden paarweise für je zwei Objekte definiert:

Definition 3 (Binäre Objektrelation) *Eine binäre Objektrelation R beschreibt für je zwei Objekte o_1, o_2 der Welt* $\mathbf{W}$ *abstrakt eine eindeutige räumliche Beziehung zwischen den beiden Objekten. Dabei gilt:*

$$R(o_1, o_2) = \begin{cases} true, & \text{wenn } o_1 \text{ in Relation } R \text{ zu } o_2 \text{ steht,} \\ false, & \text{sonst.} \end{cases}$$

Semantisch unterschiedliche Klassen binärer Relation wurden implementiert. Diese umfassen Richtungs- (nördlich, links von, höher als, ...),

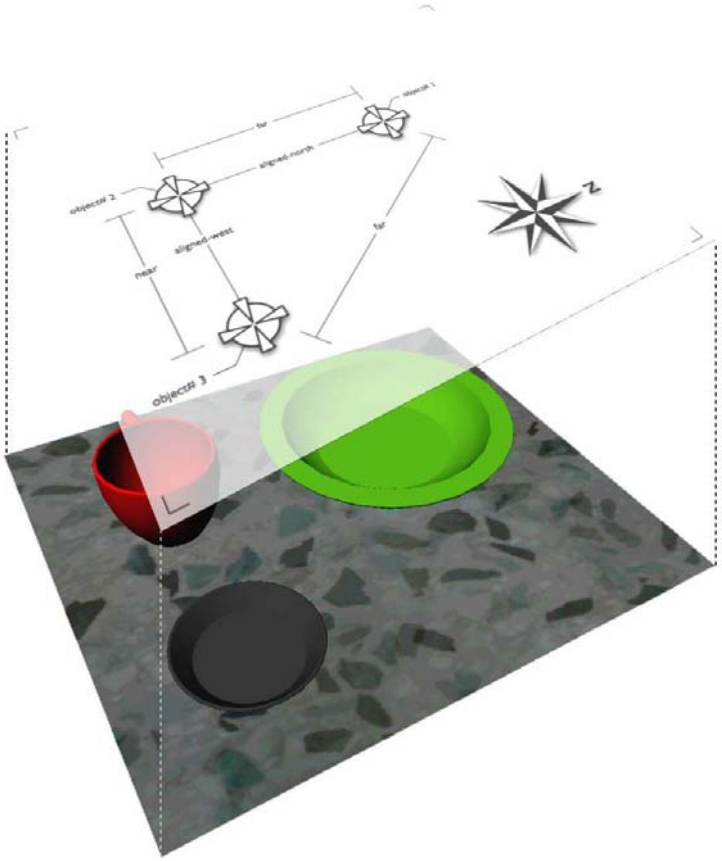


Abbildung 3.2: Semantische Beschreibung eines Weltzustandes durch geometrische binäre Relationen

Näherungs- (nahe bei, weit entfernt von, ...), Inklusions- (partiell enthalten in, umfaßt, ...), Stabiltäts- (trägt, bedeckt, ...), Kontakt- (berührt, auf, unter, ...) und Ausrichtungsrelationen (ausgerichtet an, ...). Eine vollständige Liste der implementierten Relationen findet sich in [Zöllner 2005]. Wichtig hierbei ist, daß die Menge der Relationen durch Bereitstellung einer Evaluierungsfunktion beliebig erweiterbar ist und somit jede denkbare Lagebeziehung zwischen zwei Objekten relational beschrieben werden kann.

Die dem System bekannten Objekte sind weiterhin gemäß eines Eigenschaftsmodells mit semantischer Zusatzinformation versehen: Eine Taxonomie für Objekte, die im Bereich der Haushalts- und Küchendomäne auftreten können, wurde erstellt (siehe [Becher et al. 2003]). Diese unterteilt die Menge aller Objekte zuerst in bewegliche und unbewegliche Objekte. Abhängig von Größe und Gewicht eines Objekts in Relation zu den menschlichen Abmessungen und Fähigkeiten können somit Greif- und Loslasshypothesen

auf Plausibilität überprüft und bewertet werden. In weiteren Verfeinerungen wird eine hierarchische Klassifizierung der Objekte vorgenommen. Eine solche Klassenbildung bedingt jede Art von Verallgemeinerungen über Objektklassen hinweg. So ist es möglich, von der Demonstration mit einer bestimmten Instanz einer Tasse auf Aufgaben unter Benutzung eines Trinkgefäßes zu generalisieren. Eine weitere Möglichkeit des physikalischen Modells versucht Objekte und ihre inneren Zustände zu beschreiben, beispielsweise Öffnungswinkel von Türen oder Zustände wie „heiß/kalt“ eines Ofens. Diese Informationen können für die Intentionserkennung von Bedeutung sein.

Die höchste Ebene der semantischen Objektmodellierung befaßt sich mit der funktionalen Beschreibung. Objekte können in Handlungen verschiedene, semantisch unterschiedliche Rollen einnehmen. Diese unterscheiden sich signifikant in den Teilzielen, die ein bestimmter Handlungsteil verfolgt. So kann ein Glas innerhalb einer Demonstration sowohl als befülltes als auch als befüllendes Objekt auftreten, wenn es beispielsweise als Messbecher verwendet wird. Die Informationen über mögliche Rollen können ebenfalls zum Ausschluß nicht plausibler Systemhypothesen genutzt werden.

Sowohl physikalische als auch funktionale Eigenschaften der Objekte sind durch sogenannte unäre Relationen über dem Objektraum definiert:

Definition 4 (Unäre Objektrelation) *Eine unäre Objektrelation R beschreibt für ein Objekt o der Welt $\mathbf{W}$ das Vorhandensein einer bestimmten Eigenschaft, bzw die Annahme eines bestimmten Wertes. Dabei gilt:*

$$R(o) = \begin{cases} true, & \text{wenn } o \text{ die Eigenschaft } R \text{ besitzt ,} \\ x, & \text{falls diese noch mit einem Attributwert } x \text{ verknüpft ist,} \\ false, & \text{sonst.} \end{cases}$$

Der Gesamtzustand der Welt, wie er sich dem System darstellt, wird als die Konjunktion einer Menge von (binären und unären) Relationen repräsentiert, die in einer Situation gültig sind. Er umfaßt somit eine komplette Beschreibung der in der Welt existierenden Objekte sowie deren Beziehungen untereinander:

Definition 5 (Zustand) *Der Zustand $\mathbf{S}^t_{\mathbf{W}}$, in dem sich eine Welt $\mathbf{W}$ zum Zeitpunkt t befindet ist definiert als*

$$\mathbf{S}^t_{\mathbf{W}} = \{(\bigcap_{R\in\mathbf{R},o_j,o_k\in\mathbf{O}} R(o_j,o_k))\bigcap(\bigcap_{R\in\mathbf{R}_1,o_j\in\mathbf{O}} R_l(o_j))|R \neq false\}, \qquad (3.3)$$

wobei $\mathbf{O}$ die Menge aller Objekte und $\mathbf{R}_1, \mathbf{R}_2$ die Menge aller unären bzw. binären Relationen bezeichnet.

Mit dieser Formalisierung des Weltzustandes läßt sich die Demonstration einer Handlung als Folge von Zuständen bzw. als Trajektorie im Zustandsraum beschreiben:

Definition 6 (Demonstration) *Eine Demonstration D mit der Dauer n in einer Welt $\mathbf{W}$ ist eine geordnete Liste*

$$D = (\mathbf{S}^0_{\mathbf{W}}, \mathbf{S}^1_{\mathbf{W}}, \ldots, \mathbf{S}^n_{\mathbf{W}}). \tag{3.4}$$

Die Demonstration überführt dabei die Welt aus dem Startzustand $\mathbf{S}^0_{\mathbf{W}}$ in den Endzustand $\mathbf{S}^n_{\mathbf{W}}$, unter Einnahme der Zwischenzustände $\mathbf{S}^i_{\mathbf{W}}, \forall 1 < i < n$.

Kehrt man diese eher phänomenologische Beschreibung von Aufgaben um und betrachtet die Handlungen, die zu solch einer Verkettung von Zuständen führen, so ergibt sich die Definition einer Operatorsequenz:

Definition 7 (Operatorsequenz) *Eine Operatorsequenz $O = (O_1, O_2, \ldots, O_n)$ ist eine Folge von Operatoren (dies können sowohl Elementar- als auch Makro-Operatoren sein, siehe die entsprechenden Definitionen 1 und 2), welche die Transformation einer Welt von einem Start- in einen Zielzustand erbringen, unter Einnahme aller Zwischenziele. Formal gilt*

$$\mathbf{E}(O_i) \subseteq \mathbf{V}(O_{i+1}), \forall i, 1 \leq i < n \tag{3.5}$$

Vor dem Hintergrund des Programmierens durch Vormachen ist man jedoch weniger an beliebigen Operatorsequenzen interessiert, sondern an Operatorsequenzen, welche besonders gut zu einer Demonstration passen. Diese Übereinstimmungsrelation ist noch näher zu formalisieren. Von besonderem Interesse ist für eine Demonstration eine Operatorsequenz, die genau dieselbe Folge von Zwischenzuständen aufweist, wie die observierte Handlung des Menschen. Eine solche Operatorsequenz nennt man Erklärung.

Definition 8 (Erklärung) *Eine Operatorsequenz $O = (O_1, O_2, \ldots, O_n)$ ist eine Erklärung einer Demonstration $D = (\mathbf{S}^0_{\mathbf{W}}, \mathbf{S}^1_{\mathbf{W}}, \ldots, \mathbf{S}^n_{\mathbf{W}})$ genau dann, wenn die Vorbedingungen des ersten Operators dem initialen Zustand entsprechen, die Nachbedingungen dem letzten Zustand und dazwischen alle Nachbedingungen eines Operators in den Vorbedingungen des darauf folgenden Operators enthalten sind, also wenn*

1. $\mathbf{V}(O_1) \subseteq \mathbf{S}^0_{\mathbf{W}}$ (3.6)
2. $\mathbf{N}(O_i) \subseteq \mathbf{V}(O_{i+1}) \subseteq \mathbf{S}^i_{\mathbf{W}}, \forall i, 1 \leq i < n$ (3.7)
3. $\mathbf{N}(O_n) \subseteq \mathbf{S}^n_{\mathbf{W}}$ (3.8)

Eine solche Erklärung ist deshalb interessant, da sie direkt eine Operatorsequenz liefert, welche dieselben Effekte auf die Umwelt hat, wie die vom Benutzer vorgeführte Demonstration. Programmieren durch Vormachen besteht somit im Finden einer Erklärung zu einer gegebenen Statusfolge bzw. Demonstration. Leider ist die Suche nach einer solchen Erklärung im allgemeinen ein NP-vollständiges Problem (siehe [Garey und Johnson 1978]), was eine maschinelle Auswertung erschwert, wenn nicht unmöglich macht. In den verbliebenen Abschnitten wird jedoch eine Methode vorgestellt, welche mittels heuristischen Hintergrundwissens, modellbasierter Verfahren und Techniken des Maschinellen Lernens im Falle der vorliegenden Problemklassen eine ausreichend gute Schätzung für die Erklärung einer Demonstration gibt.

3.5 Eine Architektur zum inkrementellen Handlungslernen

Die PdV-Prozeßkette, die einzelnen Teilkomponenten und die Datenflüsse des entwickelten Systems zum inkrementellen und interaktiven Programmieren von Robotern durch Vormachen sind in Abbildung 3.3 dargestellt. Die allgemeine Gliederung eines PdV-Systems, wie sie in Kapitel 2 vorgestellt wurde (vgl. Abbildung 2.1) findet sich ebenso darin wieder, wie die oben beschriebenen unterschiedlichen Analysephasen sowie psychologische Erkenntnisse zum Ablauf am Menschen beobachteter Lernprozesse. Die Algorithmen, Modelle und Methoden werden im Einzelnen in den kommenden Kapiteln dieser Arbeit behandelt. An dieser Stelle soll lediglich die Struktur der Umsetzung des vorgeschlagenen Konzeptes erläutert werden. Desweiteren werden die Gesamtzusammenhänge innerhalb dieser Softwarearchitektur dargestellt.

In der Komponente Sensorik findet der initiale Akquisition- bzw. Observationsteil des PdV-Systemes statt. In den im gegebenen Kontext durchgeführten Experimenten wurden Datenhandschuhe, magnetfeldbasierte Positions- und Orientierungssensoren, Stereokamerasysteme sowie ein aktives optisches System zur hochgenauen Vermessung von menschlichen Bewegungen eingesetzt. Eine detaillierte Beschreibung der verwendeten Sensorsysteme findet sich in Abschnitt 3.6. Durch den modularen Aufbau der sensorverarbeitenden Prozessschritte ist gewährleistet, daß die Integration von und Datenfusion mit anderen Sensorsystemen mit vertretbarem Implementierungsaufwand möglich ist.

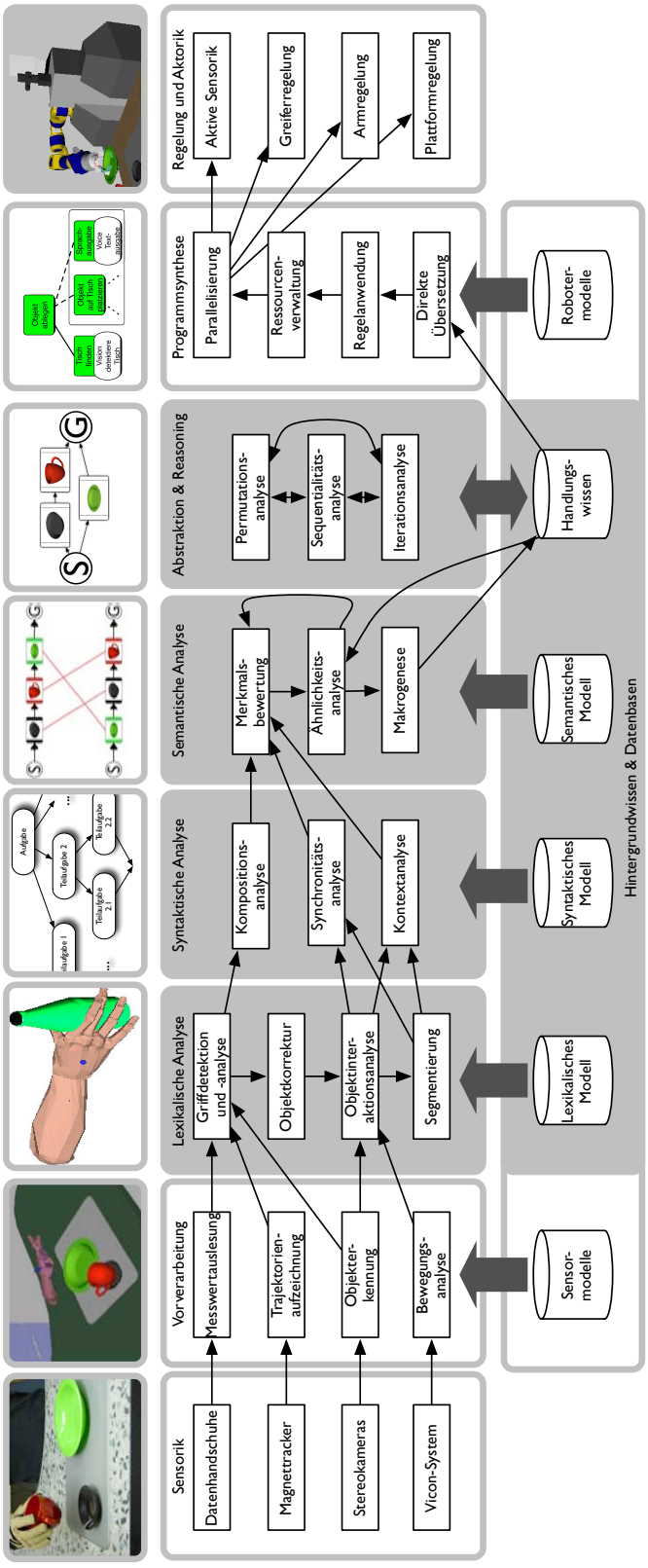


Abbildung 3.3: Systemkomponenten und ihre Zusammenhänge. Datenflüsse zwischen den Komponenten sind als gerichtete Pfeile dargestellt. Die im Rahmen dieser Arbeit erstellten oder verbesserten Systemkomponenten sind grau hinterlegt.

Bevor die gewonnenen Daten einer eingehenderen Verarbeitung und Interpretation zugänglich sind, müssen sie vorverarbeitet werden. Je nach verwendetem Sensorsystem sind dabei Verarbeitungsschritte wie die Normierung und Kalibrierung sämtlicher Sensoren, Elimination des Meßrauschens, Kompression, Segmentierung relevanter Teildatenmengen, oder noch komplexere vorverarbeitende Schritte nötig. Diese werden kurz in Abschnitt 3.6 zusammen mit der dazugehörigen Sensortechnik erläutert.

In der darauf aufbauenden lexikalischen Analysephase werden die grundlegenden verwendeten Symbole dekodiert. Dies bedeutet, daß aus dem vorverarbeiteten Sensordatenstrom die bedeutungstragenden Atome (die oben definierten Elementaroperatoren) gefunden und geeignet instantiiert werden. Hier findet die Griffdetektion und Klassifikation statt. Basierend auf heuristischem und regelbasiertem Hintergrundwissen wird eine Korrektur der Objektlagen und -orientierungen vorgenommen, die nicht in den semantisch niedrigeren Ebenen implementierbar ist. Darauf aufbauend wird die Interaktion von Objekten, wie sie beispielsweise bei Einschenkvorgängen oder Werkzeugbenutzung auftreten, analysiert und vorläufig interpretiert. Wenn diese Daten gesammelt sind, kann eine vollständige Segmentierung der Demonstration vorgenommen werden. Nach Abschluß dieses Verarbeitungsschrittes sind die Elementaroperatoren sowie ihre Anfangs- und Endpunkte vollständig bestimmt und können zur weiteren Interpretation zu größeren Gesamtzusammenhängen zusammengestellt werden. Sprachliche Kommentierungen der Demonstration, die der Benutzer gegebenenfalls zur Verfügung gestellt hat, werden in dieser Phase ebenfalls interpretiert und in die entsprechenden Systemhypothesen umgesetzt.

Im darauf folgenden syntaktischen Analyseprozeß wird über den segmentierten Elementaroperatoren der hierarchisch-funktional gegliederte Makro-Operatorbaum erzeugt. Er nimmt die Abstraktion und Interpretation der einzelnen Demonstration vor und überprüft die Hypothesen, welche das System generiert hat, auf Kohärenz. Inkonsistenzen werden durch Anpassung oder Verwerfen der Hypothesen in einen konsistenten Zustand überführt. Neben der syntaktisch korrekten Erzeugung eines Makro-Baumes wird hier eine Analyse der wechselseitigen Abhängigkeit paralleler Handlungsteile vorgenommen. Dies ist insbesondere relevant bei gleichzeitiger Handlung des Menschen mit beiden Händen: Je nach Art der durchgeführten Handlung sind die beiden Einzelhandlungen voneinander abhängig oder unabhängig. Zuletzt findet eine Analyse des in der Demonstration gezeigten Kontextes statt, d.h. eine Analyse der nötigen Vorbedingungen sowie der Nachbedingungen, denen die Ausführung der einzelnen Teile sowie die Gesamtdemonstration unterliegt.

Das somit gewonnene semantische Handlungswissen weist eine große Men-

ge an Merkmalen auf. Diese sind in den Vor- und Nachbedingungen, den Greif- und Loslaßoperatoren, den akquirierten Trajektorien oder beispielsweise in der Reihenfolge der Ausführung enthalten. All diese Merkmale haben unterschiedliche Relevanz für die Erreichung des Gesamtzieles, die zwischen unterschiedlichen Handlungsklassen signifikant variieren können. Abhängig von der Relevanz können unter Rückgriff auf das gesamte Handlungswissen, welches in Form einer Handlungsdatenbasis zur Verfügung gestellt wird, Ähnlichkeiten mit der aktuellen Demonstration gefunden und zur Interpretation genutzt werden. Dieser Prozess ist iterativ und wird mit der Makrogenese abgeschlossen, welche das neue Handlungswissen erzeugt sowie in die Handlungsdatenbasis integriert und somit das Handlungswissen für weitere Abstraktions- und Generalisierungsschritte zur Verfügung stellt.

Ein einzelner Makro-Operatorbaum repräsentiert das Wissen, das aus einer einzelnen Demonstration ableitbar ist. Er stellt die einzelne Erfahrung dar, die einer bestimmten Situation entspringt. Die Handlungsdatenbank steht für das gesamte Erfahrungswissen, die das System während seiner Laufzeit gewonnen hat, also den gesamten Erfahrungsschatz, der dem System zu einem konkreten Zeitpunkt zur Verfügung steht. Diese ist jedoch niemals als vollständig oder abgeschlossen aufzufassen, sondern steter Veränderung unterworfen. Sie besteht nicht nur aus der Ansammlung immer neuer Erfahrung, sondern auch in der Adaption, Generalisierung und Verfeinerung des Handlungswissens.

Die Veränderung des Handlungswissens erfolgt durch verschiedene Prozesse von Abstraktion und der Erschließung neuen Handlungswissens[2], die als Hintergrundprozesse die Fortschreibung und Konsistenzüberprüfung des Handlungswissens zu Zeiten niedriger Systemlast (beispielsweise parallel zum Aufladen der Stromversorgung während der Nacht) oder direkt der Demonstration folgend angestoßen werden können. Hier ist ebenfalls ein hoher Modularisierungsgrad wichtig, so daß neue Reasoning-Algorithmen einfach und mit minimalen Aufwand implementierbar und integrierbar sind. Im Rahmen dieser Arbeit wurden Verfahren zur Analyse von Operatorpermutationen, von wiederholt auftretenden Teilaufgaben und von sequentiellen Umordnungsmöglichkeiten bei der Ausführung von Handlungswissen implementiert und getestet.

Das bisher generierte Handlungswissen ist roboterunabhängig und repräsentiert die abstrakten Operationen, die zur Lösung von Aufgaben nötig sind. Soll das in der Handlungsdatenbank vorgehaltene Wissen auf einem konkreten Roboter zur Anwendung und Ausführung kommen, so ist ein Abbildungsprozess vonnöten, welcher das abstrahierte Handlungswissen auf die konkre-

[2]engl.: Reasoning

te Roboterhardware mit ihrer Sensorik und Aktorik abbildet. Dieser Prozess beginnt mit der direkten Übersetzung, welche den Makro-Operatorbaum auf einen Syntaxbaum einer Roboterprogrammiersprache abbildet. Mittels der iterativen Anwendung von Termersetzungsregeln innerhalb des Baumes muß anschließend roboterspezifisches Wissen in den Programmbaum eingefügt werden, wie beispielsweise Code zur Initialisierung bestimmter Komponenten oder zum Auslesen von Robotersensorik, die nicht im abstrahierten Handlungswissen vorhanden sind. Ressourcenbeschränkungen, die für einen bestimmten Roboter gelten, müssen anschließend aufgelöst werden. So ist eventuell ein Umplanungsschritt erforderlich, wenn auf einem Roboter mit einem Arm Handlungswissen ausgeführt werden soll, das das Vorhandensein von zwei Greifern voraussetzt. Zuletzt müssen bei der Parallelisierung von Handlungswissen verschiedene Aspekte betrachtet werden, die hier von Bedeutung sind, beispielsweise die Anzahl der gleichzeitig kontrollierbaren Freiheitsgrade eines Systems oder rechnerische Kapazitäten, die es ratsam erscheinen lassen, auf einen allzu großen Parallelisierungsgrad zu verzichten. Das solcherart durchgeführte Verfahren ist prinzipiell in der Lage, Programme in unterschiedlichen Roboterprogrammiersprachen zu erzeugen, bis hin zu direkter Generierung von Assembler-Code. Experimentell wurde lediglich die Exportierung in das Programmiersystem der sog. Flexiblen Programme (siehe [S. Knoop 2006]) evaluiert. Aufgrund der Turing-Mächtigkeit der Term- und Baumersetzungssysteme ([Goos und Waite 1984]) ist jedoch jede andere Programmrepräsentation als gleichwertig zu betrachten und prinzipiell eine mögliche Zielsprache. Das so erzeugte Roboterprogramm kann anschließend verwendete werden, um Roboter zu steuern und im häuslichen Umfeld als flexible Helfer einzusetzten. Im Rahmen dieser Arbeit wurden primär die mittleren (in Abbildung 3.3 grau hinterlegten) Prozesskomponenten implementiert und untersucht. Im Bereich der Sensorik sowie der Sensordatenvorverarbeitung, der Programmsynthese, Regelung und Aktorik wurde auf vorhandene Arbeiten zurückgegriffen. Da die verwendeten Sensorsysteme die notwendige Grundlage für das erhaltene Handlungswissen darstellen, werden sie im Rest dieses Kapitels näher vorgestellt. Der Vorstellung der zentralen Komponenten zum Lernen und Abstrahieren von Handlungswissen sind die restlichen Kapitel dieser Arbeit gewidmet.

3.6 Verwendete Sensorik

Die vorgestellten und implementierten Methoden und Algorithmen sind sensortransparent, d.h. unabhängig von einer konkreten Sensorik. Für das Verständnis der später beschriebenen Verfahren sowie deren Ergebnisse ist jedoch

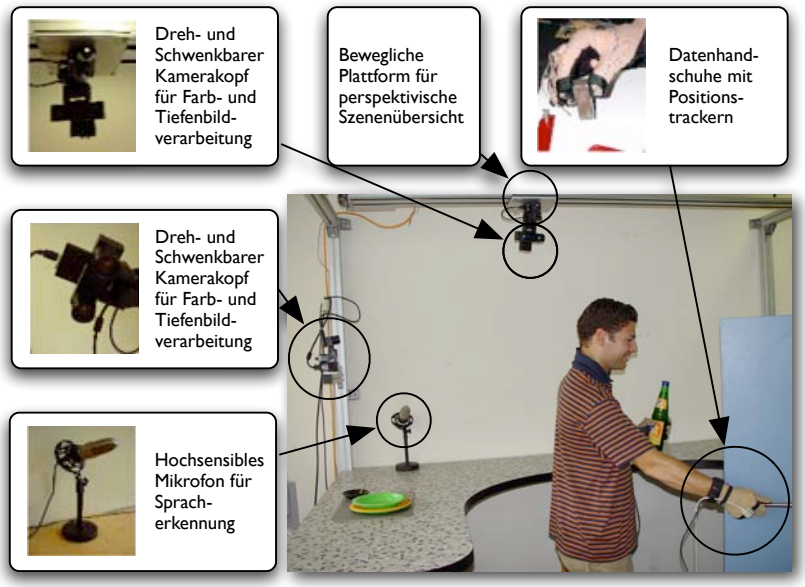


Abbildung 3.4: Sensorumgebung im Trainingscenter.

ein detailliertes Wissen über die genutzte Sensorik hilfreich. Deshalb sollen im Folgenden die verwendeten Systeme und Vorverarbeitungsmethoden kurz beschrieben werden, um deren Stärken und Schwächen abschätzen zu können. Das Sensorsystem wurde in einem sogenannten *Trainingscenter* zusammengefaßt, in dem die meisten Daten für die durchgeführten Experimente entstanden. Dieses Demonstrationsszenario ist in Abbildung 3.4 dargestellt.

3.6.1 Datenhandschuhe

Zur Erfassung der Fingerbewegungen und Griffe der menschlichen Hand wurde ein System aus mit einem Handschuh vernähten Dehnmeßtreifen verwendet (Firma Immersion [Immersion], siehe Abbildung 3.5 (a-b)). Es weist 22 unabhängige Freiheitsgrade, die jeweils mit einer Genauigkeit von ca. 2 Grad aufgenommen werden können. Die maximal mögliche Datenrate beträgt 150 Messwerte pro Sekunde, die mittels serieller Ausleseeinheiten über eine RS232-Schnittstelle in das Rechnersystem eingelesen werden. Das physikalische Meßprinzip basiert auf der Widerstandsänderung bimetallischer Streifen, die unter der Ausübung von Stauch- und Dehnverformungen auftreten. Durch geschickte Anordnung der Dehnmeßstreifen auf der Oberfläche des Datenhandschuhs können Rückschlüsse auf die Stellung der einzelnen Fingergelenke getroffen werden.

Aufgrund der elastischen Struktur des verwendeten Handschuhmaterials

Abbildung 3.5: Verwendeter Datenhandschuh. (a) Physikalischer Datenhandschuh mit Dehnmeßstreifen (b) Geometrisches Modell der Benutzerhand. (c) Abbildung von Widerstandswerten auf Gelenkwinkel.

und der von Mensch zu Mensch unterschiedlichen Anatomie ist es jedoch nicht möglich, jedem Widerstandsmeßwert global einen Gelenkwinkel zuzuordnen. Deshalb ist das System vor jeder Benutzung auf den jeweiligen Benutzer zu kalibrieren, um die Abbildungsfunktion von Widerständen in Gelenkwinkel zu bestimmen.

Diese Kalibrierung erfolgt, indem der Benutzer eine definierte Folge von Posen mit der Hand einnimmt. Daraus werden die minimalen und maximalen Widerstandswerte für jedes einzelne Gelenk extrahiert. Aufgrund der linearen Charakteristik der Dehnmeßstreifen, der orthogonalen Unabhängigkeit und der anatomischen Richtwerte für die maximale Gelenkstellungen kann so eine benutzerindividuelle lineare Übertragungsfunktion konstruiert werden, die eine Berechnung der Gelenkwinkel aus den Widerstandsmeßwerten für jedes einzelne Gelenk erlaubt (siehe Abbildung 3.5 (c)).

Mit dieser Kalibriermethode ist es möglich, die Fingerstellung bis auf etwa ein Grad Abweichung genau zu bestimmen. Das ist ausreichend, um die Position der Fingerspitzen, sowie die Gesamtpose der Hand zu erkennen. Diese Daten werden den nachfolgenden Verarbeitungsschritten, wie dem Erkennen von Griffen, zur Verfügung gestellt.

3.6.2 Magnetische Positionssensoren

Die Positionen der Benutzerhände im Raum werden durch ein magnetfeldbasiertes Tracking-System (Ascension Technology) aufgezeichnet. Dabei erzeugen drei rechtwinklig zueinander angeordneten Spulen in einem Magnetfeldemitter ein pulsierendes und mit einer dreidimensionalen Charakteristik versehendes Magnetfeld. Dieses wird durch drei ebenso angeordneten Spulen im Sensor auf dem Handrücken des Benutzers registriert. Aus den induzierten Signalen lassen sich die Raumlagen der Sensoren mit sechs Freiheitsgraden bestimmen. Die Meßreichweite liegt bei zwei Metern im Umkreis des Emit-

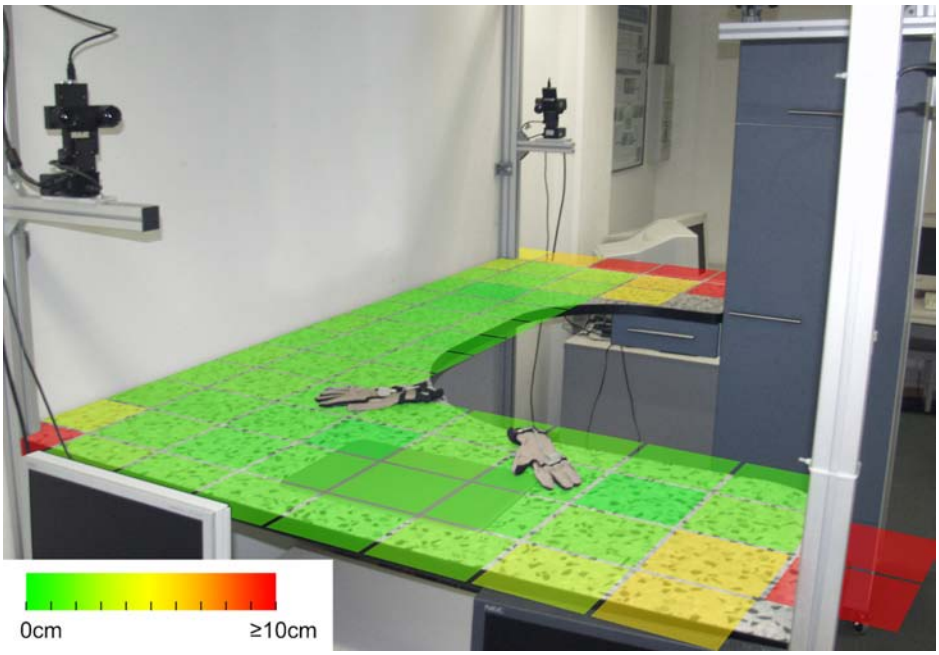


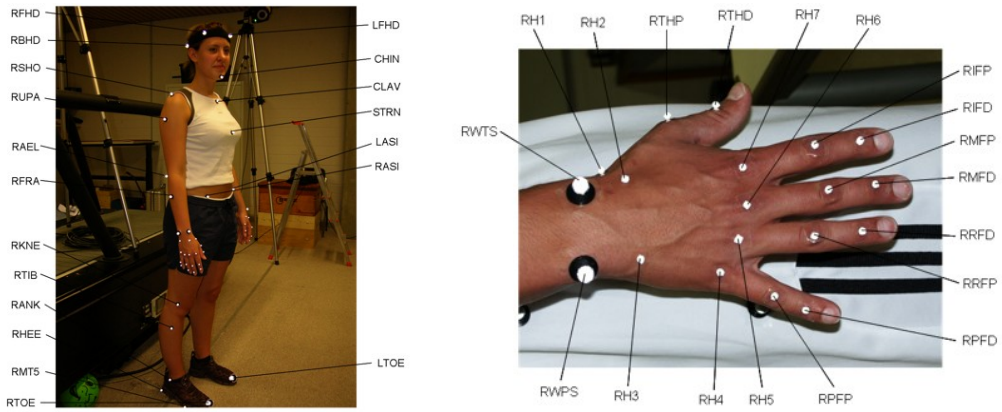
Abbildung 3.6: Kalibriergenauigkeit der Positionssensoren

ters, mit einer Genauigkeit von zwei Zentimetern und einer Auflösung von einem Millimeter. Die Datenrate beträgt 144 Meßwerte pro Sekunde, der Datenaustausch erfolgt über binäres Datenformat unter Nutzung der RS-232 Schnittstelle.

Durch das physikalische Meßprinzip bedingt, zeigte es sich, daß das Magnetfeld und damit die Positioniergenauigkeit sehr unter metallischen Objekten im Einzugsbereich des Trackingsystems leidet. Gerade Monitore und Computer senden zusätzlich noch starke Störsignale aus, die aus den Rohdaten herausgefiltert werden müssen. Deshalb müssen die gewonnenen Daten ebenfalls auf das jeweilige Umfeld hin kalibriert werden.

Zur Kalibrierung wurde angenommen, daß die exakten Positionsdaten durch Meßstörungen mit einem Rotations- und Translationsanteil überlagert wurden. Diese Annahme beruht auf der Beobachtung, daß sich Nichtlinearitäten nur in der unmittelbaren Umgebung von Objekten finden, welche Störungen im Magnetfeld verursachen. Deshalb läßt sich das Kalibrierungsproblem auf die Schätzung einer geeignete Rotation und Translation reduzieren.

Dazu wurden an unterschiedlichen, genau ausgemessenen Positionen im Trainingscenter Markierungen angebracht, an denen die aufgezeichneten Sensorwerte des Magnetfeldtrackers ausgewertet wurden. Anschließend wurde durch die Methode der kleinsten Fehlerquadrate angelehnt an [Horn 1987] die optimale Transformation geschätzt, welche die bekannten Positionen in



(a) (b)

Abbildung 3.7: Typischer Satz von Marken zur Beobachtung des gesamten Körpers und der Hand eines Probanden (SFB 588). (a) Markerset für den gesamten Körper. (b) Details von der Hand

die gemessenen Werte überführt. Durch Invertierung der Sensormatrix kann die Abbildungsfunktion der Meßwerte in den sechsdimensionalen Positions- und Orientierungsraum gefunden werden.

Eine abschließende Evaluierung ergab eine Abweichung von weniger als zwei Zentimetern in den wesentlichen Teilen der Arbeitsfläche, die nur an den Rändern und in der Nähe der Metallprofile überschritten wurde (siehe Abbildung 3.6).

3.6.3 Vicon Motion Capture System

Die vorgestellte Sensorik erlaubt eine kontinuierliche Bewegungserfassung des Menschen und von Objekten auf eine einfache, kostengünstige Art. Sie hat jedoch auch nicht zu verschweigende Nachteile, gerade in Bezug auf die Genauigkeit, bei der Kompromisse eingegangen werden mußten.

Sollen Bewegungsfähigkeiten für einen Roboter hochgenau und präzise erfaßt werden, so ist es sinnvoll, auf speziellere Hardware zurückzugreifen. So lassen sich unterschiedliche Handlungen des Menschen hochgenau mit einem Bewegungserfassungssystem aufgezeichnen, indem man mehrere schnelle und hochgenaue Kameras in Verbindung mit reflektierenden Marken am Körper des Menschen verwendet, wie sie auch bei der Bewegungsanalyse und -korrektur von Leistungssportlern zum Einsatz kommen.

Dazu werden auf dem gesamten Körper an genau definierten Punkte reflektierende Marker aufgeklebt. Diese werden von mehreren Kameras an verschiedenden Positionen aus aufgenommen und durch aktive Triangulation in

dreidimensionale Koordinaten umgerechnet. Aus den Positionen der Markerpunkte lassen sich Rückschlüsse auf die Position der einzelnen Körperteile ziehen. Die Marken müssen an geeigneten Punkten am Körper angebracht sein und sich außerdem stets im Blickfeld von mindestens zwei verschiedenen Kameras befinden.

Zur Platzierung der Marken wurde das im Rahmen des Sonderforschungsbereichs 588 „Humanoide Roboter“ definierte und umgesetzte Markerset verwendet (siehe Abbildung 3.7). Mit dieser Markenanordnung wurden verschiedene Probanden bei Handlungen in einer Küchenumgebung beobachtet. Die beobachteten Handlungssequenzen wurden vorverarbeitet und in einer Bewegungsdatenbank abgelegt. Diese Bewegungsdatenbank umfaßt Greif- und Loslassbewegungen von Küchenobjekten sowie die Bedienung von Küchengeräten wie einem Kühlschrank.

3.7 Zusammenfassung

In diesem Kapitel wurde ein Ansatz zu einem PdV-System erläutert und insbesondere auf dessen strukturellen Aspekte eingegangen. Er wurde entwickelt aus den eingangs formulierten und spezifizierten Anforderungen und Randbedingungen und vereinigt viele der im vergangenen Kapitel gegenübergestellten Konzepte des maschinellen Lernens und psychologischer Grundlagen. Der Ansatz faßt das Programmieren durch Vormachen als einen phasenartig ablaufenden Prozess des Dekodierens und Interpretierens auf, wie er bei menschlichen Kommunikationsvorgängen beobachtet wird. Dabei wird über die lexikalische, syntaktische und semantische Analyse einer Demonstration eine Generalisierungs- und Abstraktionsphase erreicht, aus der sich der Fundus der von einem Robotersystem gelernten Handlungen aufbaut.

Diese Handlungsdatenbank wird geformt von Roboterprogrammbäumen, welche als sogenannte Makro-Operatoren hierarchisch-funkional repräsentiert sind. Sie umfassen neben dem reinen Handlungswissen, wie eine bestimmte Aufgabe gelöst werden muß, auch noch Meta-Wissen. Das Meta-Wissen umfasst die Umwelt- und Vorbedingungen, sowie durch das Durchführen der Handlung erreichbaren Effekte. Diese sind durch relationale Beziehungen zwischen den Objekten repräsentiert, die bei einer Handlung benutzt oder referenziert werden.

Die Umsetzung einer demonstrierten Sequenz von Operatoren in einen erklärenden Programmbaum wurde überblicksartig durch Vorstellung der Systemarchitektur gezeigt. Grundfunktionalitäten jeder einzelnen Komponente sowie deren Zusammenspiel wurden erläutert. Da für das Verständnis der Funktionsweisen der einzelnen Systemkomponenten die Kenntnis der ange-

wandten Sensorsysteme von Bedeutung ist, wurden in einem kurzen Überblick diese vorgestellt.

Die nun folgenden Kapitel sollen die theoretischen Grundlagen und praktischen Implementierungsaspekte für jede einzelne Analysephase sowie deren Teilkomponenten bieten. Dabei werden die beiden ersten Analysephasen, die lexikalische und die syntaktische Phase, zusammen behandelt. Sie weisen einerseits einen sehr starken Formalisierungsgrad auf und können deshalb sehr kompakt dargestellt werden, andererseits sind die inhaltlichen Konzepte sehr stark miteinander verwoben.

Kapitel 4

Lexikalische und Syntaktische Analyse von Benutzerdemonstrationen

Die ersten Schritte des in Abschnitt 3.5 vorgestellten Analyse- und Interpretationsprozesses zum Programmieren durch Vormachen sind die lexikalische und die syntaktische Analyse. Die erstgenannte entspricht der Extraktion der bedeutungstragenden Elemente einer Benutzervorführung, die zweite der Erschließung der dahinterstehenden Struktur.

In diesem Kapitel werden diese beiden Phasen detaillierter vorgestellt und analysiert. Eine wesentliche Grundlage hierbei ist die Definition und Erkennung relevanter Ereignisse in der Benutzerdemonsntration, welche abstrakt in Abschnitt 4.1 behandelt wird. In den darauffolgenden Abschnitten wird auf konkrete Ereignisse, wie Griffe und spezielle Objektinteraktionen eingegangen. Abschnitt 4.5 beschreibt, wie die einzelnen elementaren Operationen voneinander abgegrenzt werden können. In den Abschnitten 4.7 und 4.8 werden die Grundlagen beschrieben, auf denen die Grammatik-getriebene Methode zur Bestimmung von Syntaxbäumen für Demonstrationen aufbaut. Die abschließenden Abschnitte 4.9 und 4.10 beschreiben dann, wie die Syntaxbäume verwendet werden können, um die wesentlichen Merkmale und Synchronitätsbedingungen einer Aufgabe aus einer Demonstration abzuleiten.

4.1 Detektion von Ereignissen

Am Anfang jedes Kommunikationsprozesses, als den eine PdV-Situation aufgefasst werden kann (siehe Abschnitt 3), steht empfängerseitig ein bislang

unstrukturierter kontinuierlicher Fluß an subsymbolischer Informationen, die direkt oder indirekt von den Sensordaten herrühren. Der erste Schritt hin zu der Erkennung von Semantik besteht darin, die verwendeten grundlegenden Handlungen zu erkennen und zu klassifizieren. Die Menge dieses grundlegenden Handlungsvokabulars wurde in Definition 1 beschrieben und formalisiert. Sie enthält alle elementaren Handlungen wie Griff- und Loslasshandlungen sowie spezielle Objektinteraktionshandlungen wie Einschenkvorgänge, Öffnungs- und Schließvorgänge beispielsweise für einen Kühlschrank oder Werkzeugbenutzungen. Diese Elementaroperatoren sind modelliert zusammen mit ihren hinreichenden minimalen Vorbedingungen sowie ihren Effekten, die für die Erschließung des gesamten Handlungszieles der Benutzerdemonstration von Bedeutung sind. So verändert beispielsweise ein Einschenkvorgang den Füllstand der beiden beteiligten Objekte oder ein Öffnungsvorgang den internen Zustand „Öffnungswinkel“ des Kühlschranks.

Um von dem konkreten Vorgang der Verarbeitung und Erkennung semantischer Einheiten zu abstrahieren, wird das Konzept des Detektors eingeführt.

Definition 9 (Detektor) *Ein Detektor überführt eine Menge kontinuierlicher oder diskreter Sensormeßwerte* $\mathbf{S}$ *des gleichen Zeitpunktes in eine symbolische Repräsentation, welche der Änderung des Weltzustandes entsprechen. Die Ausgabe eines Detektors kann zusätzlich noch von einem internen Zustand* $\mathbf{Z}_D$ *abhängen, der sich in Abhängigkeit der Eingabedaten ändert. Ein Detektor ist vollständig definiert durch ein Tupel von drei Funktionen: einer Selektionsfunktion*

$$s_D : \mathbf{S} \rightarrow Pot(\mathbf{S}),$$

die eine Teilmenge der Sensordatenmenge auswählt, dem Körper des Detektors, der eigentlichen Detektionsfunktion

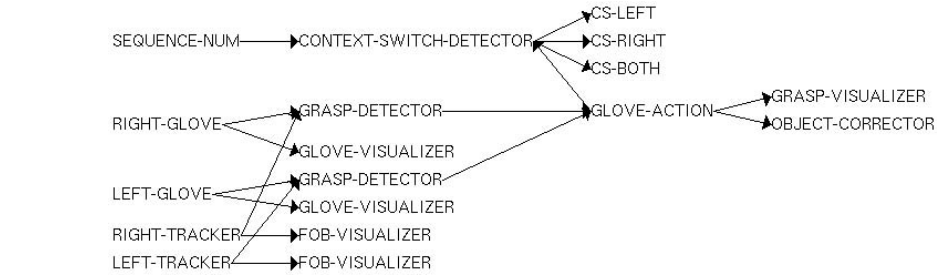
$$f_D : Pot(\mathbf{S}) \times \mathbf{Z}_D \rightarrow \mathbf{O} \cup \emptyset,$$

die die eigentliche Klassifikationsarbeit leistet und von der selektierten Teildatenmenge sowie dem internen Zustand des Detektors abhängt, sowie der Zustandsübergangsfunktion

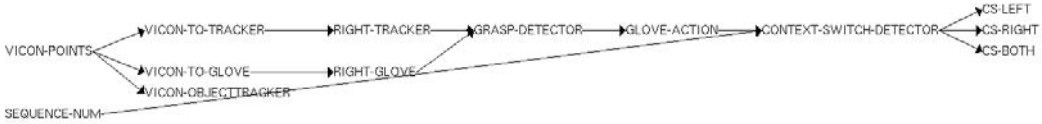
$$z_D(t+1) = c_D(s_D(\mathbf{S}), z_D(t)), c_D : Pot(\mathbf{S}) \times \mathbf{Z}_D \rightarrow \mathbf{Z}_D,$$

welche den Zustand in Abhängigkeit von den Sensorwerten verändert.

Ein Detektor ist durch die Angabe seiner Funktionen vollständig bestimmt. Darüber, wie seine Funktionen repräsentiert sind, ist im Interesse der Allgemeinheit des Detektoransatzes keine Aussage gemacht. Sie können



(a)



(b)

Abbildung 4.1: Detektorenketten. (a) Detektorenkette mit den Verarbeitungsschritten für die Sensorik des Trainingszentrums (siehe Abbildung 3.4. (b) Detektorenkette für das Vicon-Tracking-System (siehe Abbildung 3.7).

beispielsweise über die Klassifikationsfunktion eines neuronalen Netzes instanziiert sein, als regelbasierte Klassifikatoren, aber auch durch die direkte Angabe von maschinenlesbaren Codes, der die gewünschte Detektionsfunktion realisiert.

Die Ergebnisse jedes Detektors können als Eingangsparameter anderer, semantisch höherwertiger Detektoren dienen, die wiederum andere Ereignisse detektieren. Auf solche Art können komplexe Detektorketten generiert werden, welche auch komplizierten Detektionsaufgaben gewachsen sind. Dennoch ist die damit verbundene Funktionalität modular und erweiterbar, zur Laufzeit des Systems rekonfigurierbar und wiederverwendbar. So kann die Kenntnis einer neuen Elementaroperation durch Einfügen eines neuen Detektors bewältigt werden.

Zwei unterschiedliche Detektorenketten zeigt Abbildung 4.1. Abbildung 4.1 (a) zeigt sämtliche zur Detektion und Visualisierung verwendeten Detektoren, die zur Verarbeitung der Sensorik aus dem Trainingscenter, wie es in Abschnitt 3.6 vorgestellt wurde. Abbildung 4.1 (b) beinhaltet die Sensorenkette, welche zur Verarbeitung der Daten des Vicon-System aufgenommen wurden (siehe Abschnitt 3.6.3).

4.2 Griffdetektion und -analyse

Die grundlegenden Ereignisse, welche eine Handlung im Haushaltsbereich determinieren, sind das Ergreifen, Bewegen, Ablegen und Loslassen von Objekten. Der Erkennung von Griff- bzw. Loslassoperationen kommt deshalb in einem System zum Programmieren durch Vormachen hohe Bedeutung zu. Eine präzise und robuste Erkennung von Griffhandlungen ist der Ausgangspunkt, auf dem andere, semantisch höherwertige Systemkomponenten aufbauen.

Zur Griffbestimmung stehen dem System Informationen bezüglich der Position der Hand, der Gelenkstellungen der Finger sowie der Positionen der in der Umwelt durch das Stereokamerasystem erkannten Objekte zur Verfügung. Um verwertbare Aussagen über den Griffzeitpunkt sowie die verwendete Griffart treffen zu können, muß eine genaue Erkennung der Griffphase geleistet werden. Nur dann können verläßliche Informationen über die Griffart erhalten werden, die wesentlich für die Nutzbarkeit zum Ausführungszeitpunkt sind.

Unter Bezug auf die Arbeiten von [Friedrich 1998] und [Hauck et al. 1998] wurde ein regelbasierter, auf den Schwellwerten abgeleiteter Merkmale arbeitender Ansatz implementiert, welcher folgende Eingabedaten verwendet:

- Die von den Datenhandschuhen erfaßten Gelenkwinkel. Die Summe dieser Gelenkwinkel führt zu einem Schätzwert über den Grad der Geschloßenheit der Hand. Überschreitet dieser Wert einen bestimmten, objektabängigen Schwellwert, so erfüllt dies ein wesentliches Kriterium für die Erkennung des Beginns eines Griffes. Sinkt die Summe der Gelenkwinkel wieder deutlich darunter, so kann dies auf einen Loslaßpunkt hindeuten (Gleichung 4.1).

- Während des Griff- bzw. Loslaßzeitpunktes befindet sich die Geschwindigkeit in einem Minimalzustand. Idealerweise findet sogar ein Nulldurchgang statt (Gleichung 4.2).

- Ein weiteres wichtiges Indiz für das Vorliegen eines Griffes ist die Nähe des Hand zu einem Objekt. Griffe kann man auch daran erkennen, daß während der Greifphase der Abstand der Hand zu dem gegriffenen Objekt gering ist. Steigt der Abstand wieder, so ist der Griff gelöst worden, und eine Loslaßoperation hat stattgefunden. Weiterhin läßt sich das gegriffene Objekt durch Minimierung der Abstandsfunktion zu allen in der Umwelt vorhandenen Objekten einfach und zugleich verläßlich bestimmen (Gleichung 4.3).

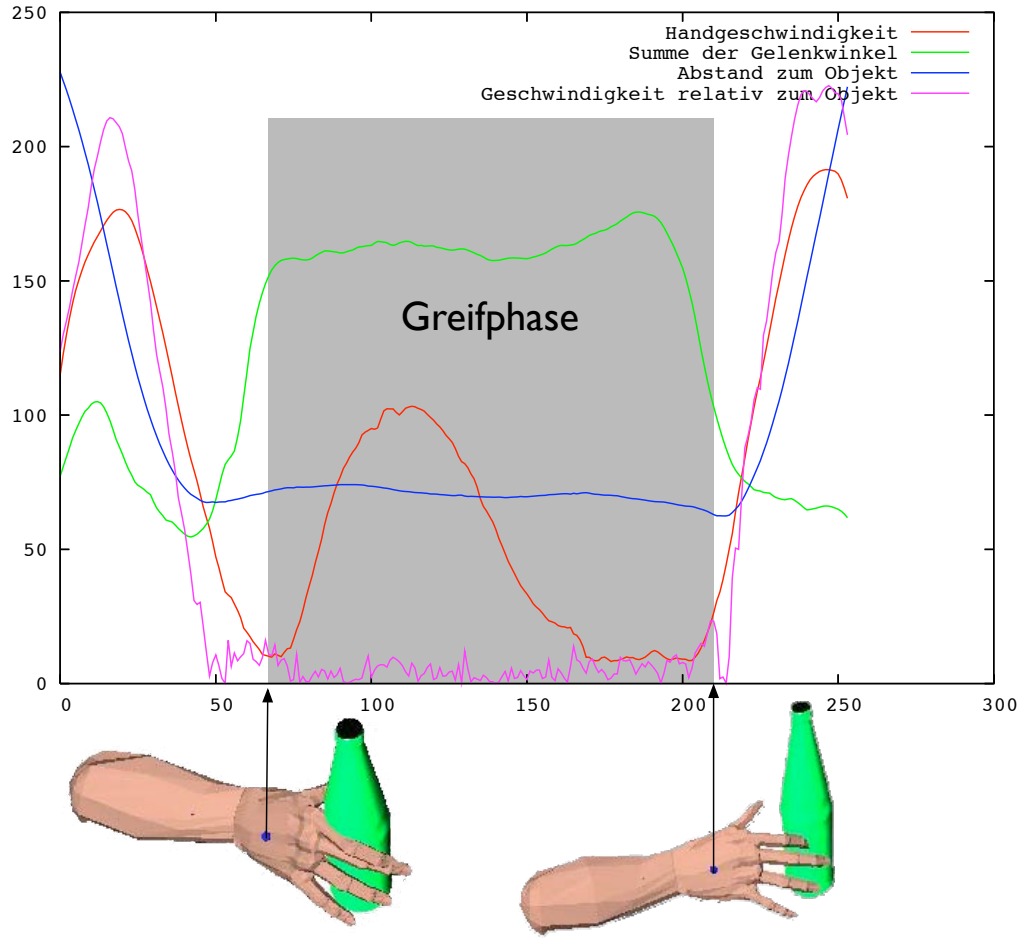


Abbildung 4.2: Regelbasierte Erkennung von Griff- und Loslaßzeitpunkten. Die Bestimmung der Greifphase findet anhand der dargestellten Parameter statt. Während der grau hinterlegten Phase ist das Objekt gegriffen.

- Ein letztes Merkmal, um den Loslaßzeitpunkt noch genauer bestimmen zu können, ist die erste Ableitung des gerade beschriebenen Merkmals nach der Zeit. Die Geschwindigkeit der Hand relativ zum gegriffenen Objekt liefert eine noch schärfere Schätzung des Objektablagezeitpunktes (Gleichung 4.4).

Der Zusammenhang zwischen den erwähnten Merkmalen und den Greif- und Öffnungszeitpunkten einer Benutzerdemonstration ist in Abbildung 4.2 dargestellt. Man erkennt, daß eine Auswertung der vier vorgestellten Merkmale mittels regel- und schwellwertbasierter Verfahren durchaus in der Lage ist, die Griffzeitpunkte zu erkennun und erklärungsbasiert zu detektieren.

Zusammenfassend läßt sich der Zusammenhang zwischen den Eingangs-

größen und dem gewünschten Prädikat folgendermaßen darstellen:

$$Griff(t) = \sum_{i=1}^{22} \alpha_i(t) > \theta_\alpha \tag{4.1}$$

$$\wedge \quad |\overrightarrow{v_{abs}}(t)| < \theta_v \tag{4.2}$$

$$\wedge \quad d_{obj}(t) < \theta_d \tag{4.3}$$

$$\wedge \quad |\overrightarrow{v_{obj}}(t)| < \theta_o \tag{4.4}$$

Mittels einfacher, regelbasierter Abfragen ist es so möglich, die Griff- und Loslaßzeitpunkte in einer Benutzerdemonstration zu erkennen. Somit wird es möglich, die grundlegenden Manipulationsoperationen zu erkennen und zeitlich zu bestimmen.

Für die erfolgreiche Umsetzung der Griff- und Loslaßsequenzen auf einen Roboter ist jedoch noch mehr Information nötig, als lediglich der Zeitpunkt eines Griffes. Eminente Bedeutung kommt auch der Bestimmung der Griffart zu. Der Mensch hat viele verschiedene Möglichkeiten, Gegenstände zu ergreifen. Diese sind in [Cutkosky 1989] näher untersucht und zusammengefaßt worden. So ist es beispielsweise von großem Interesse, ob der Benutzer einen Gegenstand mit dem Daumen parallel zu den anderen Fingern ergreift oder entgegengesetzt, oder ob er einen Griff verwendet, der die Ausübung großer Kräfte erlaubt, im Gegensatz zu Kräften, die eher zu präzisen Feinmanipulationen gedacht sind.

Die im Rahmen dieser Arbeit verwendete Griffhierarchie ist in Abbildung 4.3 dargestellt. Sie untergliedert die Menge aller anwendbaren Griffe zuerst in Kraft- und Präzisionsgriffe. Erstere dienen für schwere Objekte, sowie für schnelle Verfahrtrajektorien, die ein hohes Maß an Stabilität des Objektes in der Hand erfordern, letztere eher für die geschickte Manipulation leichter Objekte, bei der es auf hohe Genauigkeit der Bewegung des Objektes ankommt. Auf den nachgeordneten Ebenen wird nach Art und Form der gegriffenen Objekte (rund oder prismatisch) unterschieden.

Diese Grifftaxonomie eignet sich zur Beschreibung von im Haushaltsbereich typischerweise auftretenden Handlungen, wie [Friedrich 1998, Ehrenmann 1998] zeigen konnten. Aufgrund der guten Erkennungsleistung von über 80% auf repräsentativen Testdaten wurde der Ansatz von [Friedrich 1998] zur Klassifikation der einzelnen Griffe gewählt.

Diesem Ansatz folgend wurden dreischichtige „feed-forward“ Neuronale-Netzwerke gewählt. Für jede Hierarchieebene in der Grifftaxonomie, welche exakt einer Entscheidung über die Art bzw. Subklasse des verwendeten Griffes entspricht, wurde ein solches neuronales Netz trainiert. Die Entscheidungsklassen für jedes Neuronale Netz sind in Abbildung 4.3 rot umrandet.

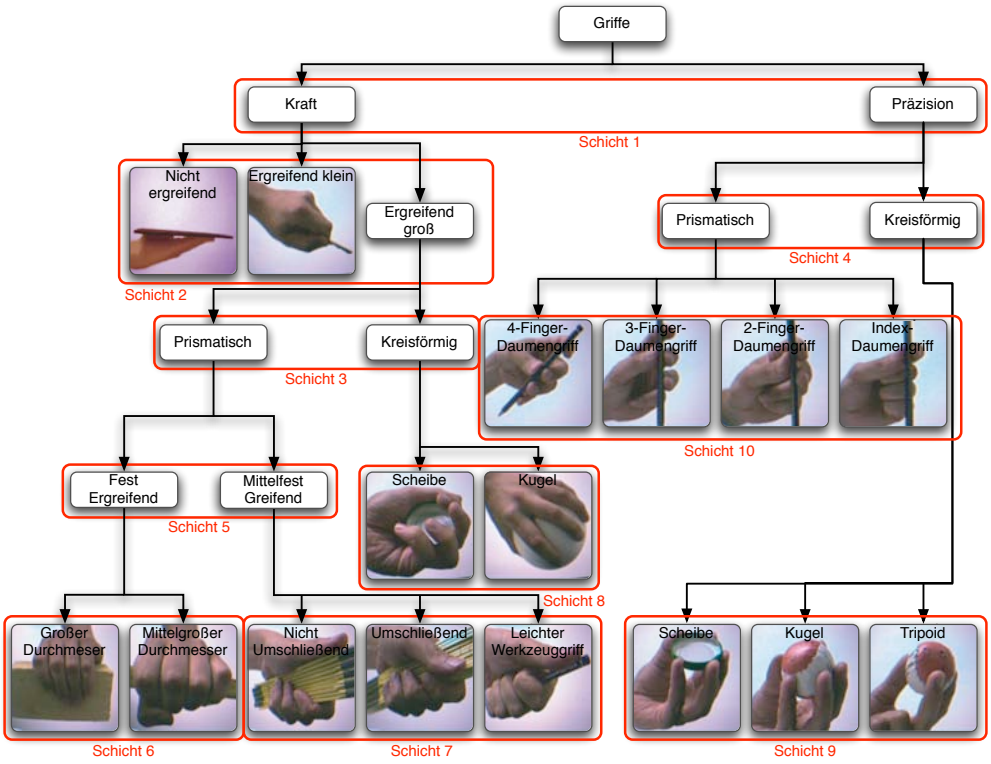


Abbildung 4.3: Grifftaxonomie nach [Cutkosky 1989]. Die Klassifikation erfolgt mittels hierarchischer Neuronaler Netze, deren Entscheidungsbereiche rot umrandet sind.

Ebenenweise wird die Entscheidung, ähnlich wie beim Entscheidungsbaumverfahren von jedem Netz in ein nachgeordnetes Neuronales Netz propagiert, bis ein Blatt des Baumes erreicht ist und damit die Klassfikationsentscheidung getroffen ist. Die mit dem erreichten Blatt verknüpfte Griffklasse wird anschließend als Klassifikationsergebnis zurückgegeben (siehe [Friedrich 1998, Friedrich et al. 1999]).

Diese Vergehensweise hat den Vorteil, daß in bei jeder Entscheidung lediglich zwei bis vier Klassen diskriminiert werden müssen. Daraus resultiert ein stabileres Verhalten der neuronalen Netze zum Lernzeitpunkt. Weiterhin müssen für Netze in weiter unten liegenden Schichten nur Teile der gesamten Datenmenge zum Training verwendet werden, was weiterhin für einen schnelleren Lernvorgang sorgt.

Als Datengrundlage für das Trainieren der Neuronalen Netze stand eine Trainingsmenge von 1280 Griffbeispielen zur Verfügung. Jede Griffklasse wurde durch 80 Exemplare repräsentiert, die von 10 verschiedenen Personen aufgenommen wurden, um eine benutzerunabhängige Griffklassifikation

zu erhalten. Die Trainingsdaten für jedes einzelne Netz bestand aus allen Griffbeispielen, die sich in der Taxonomie unterhalb dieses speziellen Neuronalen Netzes befanden. Alle anderen Griffbeispiele wurden beim Training nicht berücksichtigt.

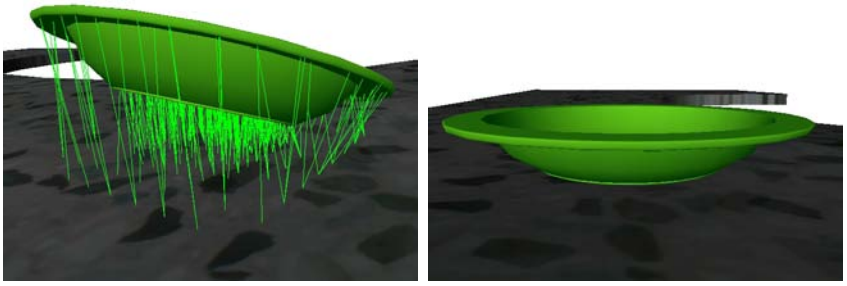
4.3 Objektlagenkorrektur

Neben dem korrekten Greifzeitpunkt sowie dem verwendeten Griff bei einer Manipulationshandlung ist ein weiterer, subsymbolischer Aspekt von zentraler Bedeutung zur vollständigen Erfassung der Benutzerhandlung: Die korrekte Position der Objektaufnahme- bzw. -ablagepunkte.

Deren korrekte Bestimmung wird im Idealfall von der verwendeten Sensorik (Stereokamera-basierte Objektverfolgung, bzw. Vicon-Tracking-System) durchgeführt. Leider stellt man in der Praxis fest, daß einerseits die Auflösungsgenauigkeit der Sensorsysteme nicht ausreichend hoch ist, um relational scharfe Aussagen wie „Ausgerichtet an“ zu stützen. Andererseits ist die Sensorik vom Sichtfeldbereich der Kameras abhängig, was in Haushaltsumgebungen durchaus eine substantielle Einschränkung darstellt. Dort ist in der Regel mit Okklusionen zu rechnen. Diese Verdeckungen der relevanten Objekte können beispielsweise bei Manipulationen im Kühlschrank auftreten, wenn der Benutzer, die Kühlschranktür und die Seitenwände des Kühlschrankes den gesamten Inhalt verdecken. Gerade in diesem Szenario, wenn nun ein Objekt wie eine Flasche gegriffen oder im Kühlschrank abgelegt werden soll, ist eine korrekte Objektposition von großer Relevanz.

Dieses Beispiel legt nahe, daß auch eine von der visuellen Sensorik unabhängige Objektlagenschätzung von hoher Bedeutung ist. Wesentliche Bedeutung kommt hierbei den anderen Sensormodalitäten (den Datenhandschuhen und Magnetfeldtrackern) zu, die nicht mit Okklusionsproblemen behaftet sind. Nimmt man nun an, daß solange der Griff gültig ist, sich die Position des Objektes relativ zur Hand nicht ändert (siehe Abbildung 4.2), so kann man die Position des Objektes durch Propagierung der Handposition fortschreiben. Sobald der Griff gelöst wird, ist das Objekt an seiner Endposition angelangt. Diese sogenannte statische Griffhypothese reicht aus, um die Position des gegriffenen ungefähr zu bestimmen. Durch sie ist es möglich, das Nichtvorhandensein visueller Informationen aufgrund von Okklusionen auf das oben als erstes angesprochene Problem der Sensorungenauigkeit zu reduzieren.

Als nächstes ist das Problem der iterativen Korrektur der Objektposition, ausgehend von einem initialen Schätzwert, zu lösen. Dazu betrachtet man mögliche Ablagepositionen für Objekte. Beispiele hierfür sind das Ab-



(a) (b)

(c) (d)

Abbildung 4.4: Objektpositionskorrektur mittels eines Iterative-Closest-Point-Verfahrens. (a,c) Vor der Korrektur. (b,d) Nach Anwendung des Korrekturschrittes. Die modellierten elastischen Kräfte, die die Objekte zu den Kontaktflächen ziehen sind als grüne Linien dargestellt.

legen eines Tellers auf einer ebenen Tischfläche, das Stellen einer Tasse auf eine Untertasse oder Zuschrauben einer Flasche mit einem Deckel. In allen drei Fällen ist die Ablageposition dadurch charakterisiert, daß Objektflächen (die Tellerunterseite und die Tischplatte oder der Tassenboden und die Oberseite der Untertasse) miteinander in Berührung sind. Diese Bedingung wird durch die Minimierung des Punkt-zu-Punkt-Abstandes zwischen den beiden involvierten Objekten erreicht.

Dem entspricht die gewählte Vorgehensweise: Mittels des Iterative-Closest-Point-Algorithmus (siehe [Horn 1987, Besl und McKay 1992]) werden Punkte auf dem manipulierten Objekt ausgewählt, die sich in enger Nachbarschaft zu Objektpunkten in der Umgebung befinden. Deren Abstände werden anschließend iterativ minimiert, bis die Summe der Distanzen minimal wird.

Die Ergebnisse dieses Verfahrens sind in Abbildung 4.4 dargestellt. Die Nächsten-Punkte-Paare sind in bunt eingezeichnet. Man erkennt deutlich die

gewonnene Positionsgenaugkeit und Aussagekraft der Objektpositionen in Termen der in Abbildung 3.2 dargestellten Beschreibungssprache für den Weltzustand.

4.4 Objektspezifische Aktionen

In den vergangenen Abschnitten wurde beschrieben, wie Greif- und Loslaßvorgänge aus Benutzerdemonstrationen für Handlungen im Haushaltsbereich erkannt und hinsichtlich verschiedener Aspekte interpretiert werden können. Menschliches Handeln weist jedoch einen weit größeren Umfang auf, als er durch das Aufnehmen und Ablegen von Objekten abgedeckt ist. Beispiele dafür sind das Einschenken von Flüssigkeiten aus einer Flasche, das hineindrehen einer Schraube oder das Öffnen des Kühlschrankes oder einer Schublade. All diese Aktionen zeichnen sich durch folgende zwei Merkmale aus:

1. Der Mensch führt mit einem oder mehreren gegriffenen Objekten bestimmte Bewegungen bzw. Handlungen aus. Das kann zum einen das Erreichen einer nahezu waagerechten Position einer Flasche relativ zu einem Gegenstand, welcher als Flüssigkeitsbehälter fungiert, oder die Bewegung auf einer Kreisbahn mit gegriffener Türklinke sein, aber auch die sensomotorisch komplexe Drehbewegung beim Öffnen eines Schraubverschlusses oder dem hineindrehen einer Schraube sein.

2. Solche Handlungen haben Änderungen interner Zustände von Objekten zur Folge. Beispielsweise verändert eine das Einschenken einer Flüssigkeit den Zustand eines Glases von „leer“ auf „voll“, das Drücken eines Knopfes an einem Küchengerät dessen Betriebsmodus von „aus“ auf „an“ oder das Aufschrauben eines Objektes wie einer Flasche deren Zustand von „geschloßen“ in „offen“.

Diese nur knapp skizzierte Vielfalt an objektspezifischen Operationen läßt sich nicht durch eine geschlossene, kompakte Formulierung entsprechend der Bestimmung von Greif- und Ablegeprimitiven gemäß den Gleichungen (4.1) bis (4.4) abdecken. Im Rahmen dieser Arbeit wurde auf eine umfangreiche Objektdatenbank zurückgegriffen, wie sie beispielsweise in [Becher et al. 2003] beschrieben worden ist. Diese erlaubt es, Merkmale und Funktionalität beliebiger Objekte zu modellieren.

Es wird daher vorgeschlagen, mit den objektspezifischen Daten phänomenologische und semantische Deskriptoren zu verknüpfen, welche genau die beiden unter 1. und 2. genannten Merkmale von den Objektaktionen repräsentieren und in einer geeigneten Form zum Demonstrationszeitpunkt zur

Verfügung stellen. Das phänomenologische Teilmodell besteht in Merkmalsabfragen, welche zum Demonstrationszeitpunkt ausgewertet werden müssen. Ist nämlich ein bestimmtes Objekt gegriffen, so können jederzeit beispielweise mit diesem Objekt verknüpfte spezifische Aktionen auftreten. Im Beispiel des Einschenkens aus einer Flasche werden, sobald eine Flasche gegriffen ist, folgende Merkmale evaluiert:

1. Öffnungszustand der Flasche: Wurde der Deckel entfernt?
2. Objektposition: Befindet sich die Flasche in einer waagerechten Position mit ihrer Öffnung über einem anderen Objekt (passives Objekt)?
3. Passives Objekt: Hat dieses Objekt die Eigenschaft, Flüssigkeitsbehälter zu sein und ist es bereit, Flüssigkeit aufzunehmen (geöffnet etc.)?

Werden diese Bedingungen positiv beantwortet so liegt nach dieser phänomenologischen Modellierung eine Einschenkoperation vor. Sie verändert den Füllzustand der zwei beteiligten Objekte, der Flasche und des Flüssigkeitsbehälters, welcher als passives Objekt an dieser Aktion beteiligt ist. Dies äußert sich in einer semantischen Modellierung der objektspezifischen Aktion „Einschenken aus einer Flasche“, welche mit der Flasche F und dem passiven Objekt O durchgeführt wurde in einem Kontext, bzw. der Vorbedingung von

$$Filled(F) = true$$

(modelliert durch eine unäre Relation gemäß Definition 4), sowie einem Effekt von

$$Filled(O) = true.$$

Das bedeutet, daß die Operation zur Voraussetzung hat, daß die Flasche nicht leer ist, sowie daß sie den Füllzustand der Tasse ändert. Da es mit der verwendeten Sensorik (siehe Abschnitt 3.6) nicht möglich ist, Füllstände von Objekten genau zu bestimmen, wurde auf eine genaue Modellierung von Füllstandsveränderungen, bzw. umgeschütteter Mengen verzichtet.

Für andere objektspezifischen Aktionen, wie sie oben beispielhaft genannt wurden, können sinngemäß phänomenologische und semantische Modelle definiert und implementiert werden. Dies wurde für die oben genannten Aktionen durchgeführt, das Konzept der phänomenologischen und semantischen Modellierung von Objektaktionen in umfangreichen Datenbanken ist jedoch allgemein.

Sind nun die Griffpunkte und Aktionszeitpunkte einer Demonstration bestimmt, so muß als nächstes die zeitliche Ausdehnung der einzelnen Aktionskontexte bestimmt werden. Dies findet in der Segmentierungsphase statt.

4.5 Segmentierung

Für die Erfassung der Bedeutung einer Benutzerdemonstration sind lediglich die Griff- und Loslaßzeitpunkte, sowie die Objekt-Aktionspunkte von Bedeutung. Zur Bestimmung der semantischen Information einer Benutzerhandlung sind also lediglich die bereits gefundenen Greifaktionen sowie die Interaktionen zwischen den Objekten relevant, wie sie in den vorigen Abschnitten behandelt worden sind.

Soll jedoch auch noch die Pragmatik einer Demonstration erfaßt werden, so ist mehr Information nötig. Die *Pragmatik* einer Benutzerdemonstration umfaßt im Gegensatz zur Semantik nicht eine funktionale Beschreibung der Handlung, sondern eine phänomenologische Repräsentation der Demonstration. Sie behandelt also nicht die Fragestellung, *was* während der Benutzerhandlung geschehen ist, sondern *wie* der Benutzer die Veränderungen in der Umwelt bewirkt hat. Diese Informationen sind von wesentlicher Bedeutung für die Umsetzung der menschlichen Aktionen auf den Manipulator. Sie umfassen im wesentlichen Trajektorien und Gelenkwinkelverläufe. So ist es beispielsweise für einen Greifvorgang von zentraler Bedeutung, nicht nur wann ein Griff geschloßen wurde, sondern auch wie die Anfahrtstrajektorie verlaufen ist, wie sich in dieser Phase der Öffnungs- bzw. Schließungszustand der Hand verändert hat, und wie nach Vollendung des Griffes das Objekt bewegt wurde.

Basisaktionen wie Griffe oder Objektinteraktionen stehen also nicht für sich, sondern sind eingebettet in Kontexte. Die einzelne Griffoperation steht beispielsweise im Gesamtkontext „Greifbewegung für ein Objekt“ , die Anfahrts- und Abfahrtstrajektorien umfaßt. Eine komplexe Fragestellung ist, wann der Mensch in seiner Bewegung zwischen den Kontexten wechselt, d.h. wann die Bewegung von der Abfahrtsbewegung des Objektaufnahmepunktes in die Anfahrtstrajektorie für den Objektablagepunkt übergeht.

Um solche Kontexte zu den einzelnen erkennbaren Basisaktionen zu bestimmen, wurden heuristische Verfahren, basierend auf den Trajektorienzügen selber, eingesetzt. Diese nutzen die Beobachtung aus, daß im allgemeinen das Ende einer Teiltrajektorie, bzw. der Anfang der nächsten Trajektorie erreicht ist, wenn sich die aktuelle Handposition in äquidistantem Abstand sowohl zur Objektgreifposition, als auch zur Objektablageposition befindet.

Diese Bedingung wird erfüllt durch den Trajektorienpunkt $\overrightarrow{p}_{cs}$, der den absoluten Betrag der Differenz der Abstände zu den Trajektorienpunkten zu den zeitlich begrenzenden Zeitpunkten t_{start} und t_{end} minimiert. Die Start- und Endzeitpunkte t_{start} und t_{end} werden durch die umgebenden Objektaufnahme- bzw. -ablagepunkte oder die begrenzenden Objektinteraktionspunkte bestimmt. Formal ausgedrückt ist

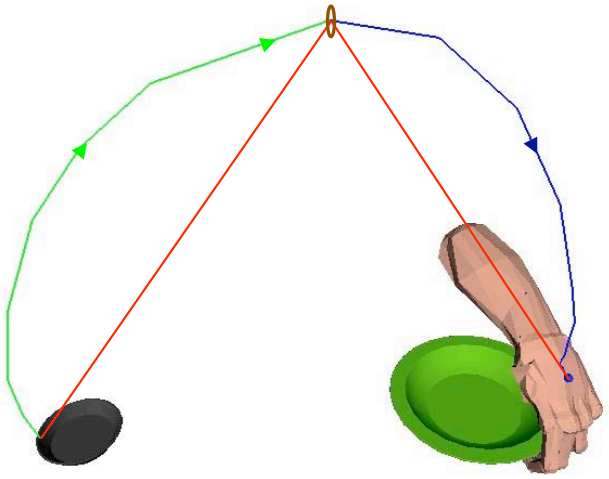


Abbildung 4.5: Heuristische Bestimmung von Kontextwechselpunkten nach Gleichung 4.5. Der Kontextwechselpunkt ist an der Position, wo die grüne Abfahrtstrajektorie von der Untertasse in die blaue Anfahrtstrajektorie für den Teller übergeht. Die beiden Distanzen zwischen dem Kontextwechselpunkt und dem Start- und Zielpunkt der Trajektorie (siehe Gleichung 4.5) sind rot dargestellt.

$$\overrightarrow{p}_{cs} = \min_{start \leq i \leq end} |(|\overrightarrow{p}(t) - \overrightarrow{p}(t_{start})| - |\overrightarrow{p}(t) - \overrightarrow{p}(t_{end})|)|. \tag{4.5}$$

Dieser Zusammenhang ist in Abbildung 4.5 verdeutlicht.

4.6 Sprachliche Kommentierungen

Bei der Weitergabe von Handlungs- und Ausführungswissen zwischen Menschen spielen während der Vorführung gegebene Erklärungen und Kommentare eine wesentliche Hilfestellung zum Verständnis der Intention und Zielsetzung der Handlung, falls jene Zielsetzung nicht oder nur schwer aus der Demonstration selbst geschlossen werden kann (siehe [Pardowitz et al. 2006d]). Der Demonstrator hat die Möglichkeit, während der Vorführung seine Intention durch Kommentare und Erläuterungen zu unterstreichen. Der Lernende bekommt dadurch Hilfestellungen, seine Aufmerksamkeit auf die relevanten Eigenschaften der Demonstration zu fokussieren.

Ein solches Verhalten sollte auch ein lernendes Robotersystem aufweisen. Es sollte in der Lage sein, seine Hypothesen und Annahmen, die es über die Aufgabe macht, zu verbessern, sobald ihm sprachliche Kommentare zur Vorführung zur Verfügung gestellt werden. Das Ergebnis sollte ein beschleunigter Lernprozess sein, der die Kommentare nutzt, um Hypothesen schneller zu verwerfen, zu korrigieren oder zu bestätigen und somit sicheres Handlungswissen zu haben.

Zur Umsetzung von Kommentaren in Bewertungen oder Modifikationen von Hypothesen benötigt das System zwei wesentliche Arten von Hintergrundwissen, die im folgenden kurz behandelt werden sollen: Zum einen eine syntaktisch-linguistische Ontologie der Aufgabendomäne, sowie einem Hypothesenmodell.

Die syntaktisch-liguistische Ontologie der Aufgabendomäne besteht aus der grammatikalischen Modellierung der Kommentare sowie ihrer Bedeutung für den menschlichen Benutzer. Dies ist notwendig, um dem System überhaupt ein Erkennen sowie eine Interpretation der Kommentare zu ermöglichen. Es bestimmt, was als Kommentar anzusehen ist, sowie wie die Semantik der Kommentare abzuleiten ist.

Das Hypothesenmodell beschreibt das Aufgabenmodell des Systems. Es bestimmt, wie Kommentare einer gewissen Semantik umzusetzen sind auf die Hypothesen, die das System aus Benutzerdemonstrationen generiert, sowie deren Modifikationen, die sich aus den Kommentaren ergeben. Hierbei sind unterschiedliche Arten von Kommentaren auf unterschiedliche Teile der Hypothesen abzubilden.

Die in dieser Arbeit verwendete syntaktisch-linguistische Ontologie deckt aus der Fülle aller folgende Kommentarmöglichkeiten vor allem Angaben zur Relevanz der erreichten oder zu erreichenden Teilziele ab. Diese beschreiben Aufnahme- und Ablagepositionen von Objekten durch Lokalitätsspezifikationen, wie sie bspw. durch Worte wie „neben“, „über“, „unter“ oder „rechts von ...“ indiziert werden. Der menschliche Benutzer bekommt somit die Möglichkeit, Kommentare wie „Ich stelle die Tasse auf den Tisch neben den Teller“ dem System zur Demonstrationszeit zur Verfügung zu stellen. Daraus kann das System Informationen über das gegriffene Objekt, Referenzobjekte und deren relative Position zueinander ziehen.

Nicht zufällig ähnelt diese Hypothesenstruktur sehr stark der relationalen Modellierung von Weltzuständen und Handlungszielen aus Abschnitt 3.4 (siehe Definition 3). Diese wurde entsprechend den beobachteten Handlungsmustern implementiert, welche der Mensch selbst anwendet. Somit ist eine sinnvolle Zuordnung von Kommentaren zu den Systemhypothesen, also relationalen Beziehungen zwischen Objekten, gegeben.

Zur Realisierung der Spracherkennung wurde das Dia-

logsystem von [Pommer et al. 2006, Holzapfel et al. 2006, Holzapfel und Waibel 2006] angewandt. Hierbei handelt es sich um ein Spracherkennungssystem, welches aus dem Dialogkontext Erwartungen formuliert und diese ausnutzt, um die Spracherkennungsrate zu verbessern. Abhängig vom sprachlichen Hintergrund des Benutzers und der konkreten Kommunikationssituation werden grammatikalische Regeln im Spracherkenner mit unterschiedlichen Gewichten versehen, welche es erlauben, die erfasste Information kontextabhängig optimiert zu erfassen.

Mit der Erkennung und Auswertung der sprachlichen Kommentare, welche eine Vorführung einer Manipulationsaufgabe begleiten, ist die lexikalische Analysephase abgeschlossen. Die Demonstration wurde vom kontinuierlichen und hochdimensionalen Sensordatenstrom abstrahiert und liegt jetzt als Folge von diskreten Ereignissen vor. Damit ist die Handlungsfolge, welche der Mensch gewählt hat, entschlüsselt. Die resultierenden Symbole sind in den folgenden Analyseschritten zu einer hierarchisch-funktionalen Darstellung des Handlungswissens, welche größere Gesamtzusammenhänge berücksichtigt und die Intention des Menschen hinter seinen Handlungen versucht zu verstehen.

4.7 Syntaktische Analyse von Benutzerdemonstrationen

Nach Beendigung der mit der lexikalischen Phase verbundenen Analyseschritte liegt eine erste formale Repräsentation einer Benutzerdemonstration vor, die in aufeinanderfolgende Segmente zerteilt ist, welche die niedrigsten Handlungseinheiten repräsentieren, wie zum Beispiel Greifbewegungen oder Einschenkoperationen. Zum kompletten Verständnis der gesamten Benutzerdemonstration, und nicht nur ihrer Einzelteile, ist eine hierarchische Abstraktion dieser Handlungseinheiten, zu höherwertigen Teilaufgaben sinnvoll. So können Greif-, Einschenk- und Loslaßoperationen beispielsweise zusammengefaßt werden zum Bedienen einer Person oder dem Decken eines Tisches.

Die syntaktische Analyse extrahiert aus der Benutzerdemonstration nun genau das Wissen um die hierarchischen Abstraktionsmöglichkeiten einer Benutzerdemonstrationen. Sie baut eine baumähnliche Erklärungsstruktur für das observierte Benutzerverhalten auf, unter besonderer Berücksichtigung der in der Demonstration vorhandenen Synchronitätsbedingungen, und konstruiert durch Kontextanalyse ein zusammenhängendes Bild der Benutzerintention hinter der Manipulationshandlung. Zur Modellunterstützung wird für diese Analysephase auf sogenannte „Attribut-Grammatiken“ zurückgegrif-

fen. Diese sollen im folgenden kurz vorgestellt werden.

Attribut-Grammatiken sind eine Erweiterung regulärer Grammatiken. Hier wird folgende, an [Goos und Waite 1984] angelehnte Definition regulärer Grammatiken angewandt:

Definition 10 (Kontextfreie Grammatik) *Eine kontextfreie Grammatik G ist definiert als ein Quadrupel $(\mathbf{T}, \mathbf{N}, \mathbf{P}, Z)$ mit*

- *der Terminalmenge $\mathbf{T}$,*
- *der Nichtterminalmenge $\mathbf{N}$,*
- *der Produktionsmenge $\mathbf{P}$*
- *sowie dem grammatikalischen Startsymbol (auch Satzsymbol genannt) $Z \in \mathbf{N}$,*

genau dann, wenn $(\mathbf{T} \cup \mathbf{N}, \mathbf{P})$ ein generelles Termersetzungssystem ist, dessen Produktionen die Eigenschaft haben, daß der zu ersetzende Teil auf der linken Seite der Produktion exakt einem Nichtterminal entspricht, bzw. daß

$$\forall p \in \mathbf{P} \exists A \in \mathbf{N}, \chi \in (\mathbf{T} \cup \mathbf{N})^* : p = A \rightarrow \chi. \quad \textit{(Kontextfreiheit)} \tag{4.6}$$

Grammatiken spezifizieren einen Prozeß, um Sätze einer bestimmten Sprache zu erzeugen und erlauben es daher, eine endliche Beschreibung für eine potentiell unendliche Sprache anzugeben. Reguläre Grammatiken sind deshalb von Interesse, weil automatisierte Verfahren existieren, welche einen Satz der durch eine Grammatik G beschriebenen Sprache in einen dazu passenden Strukturbaum überführen.

Definition 11 (Strukturbaum) *Gegeben sei ein geordneter Baum mit der Wurzel $k_0 \in \mathbf{T} \cup \mathbf{N}$ und deren unmittelbaren Nachfolgern $k_1, \ldots, k_n (n > 0)$. Dann ist k_0 die Wurzel eines* Strukturbaumes *gemäß der Grammatik $G = (\mathbf{T}, \mathbf{N}, \mathbf{P}, Z)$, genau dann, wenn*

1. *$k_0 = Z$*
2. *$Z \rightarrow k_1 \ldots k_n \in \mathbf{P}$*
3. *falls $k_i \in \mathbf{T}$, dann ist k_i ein Blatt des Baumes*
4. *falls $k_i \in \mathbf{N}$, dann ist k_i Wurzel eines Strukturbaumes gemäß der Grammatik $(\mathbf{T}, \mathbf{N}, \mathbf{P}, k_i)$*

Der Strukturbaum einer Demonstration entspricht nun genau der Morphologie eines Makro-Operator-Baumes, wie er in Abbildung 3.1 dargestellt ist. Das Ziel lautet nun, eine endliche Menge von Regeln zu spezifizieren, die der formalen Sprache der potentiell unendlichen Menge aller Benutzerdemonstrationen entspricht. Dies wird in Abschnitt 4.8 erreicht. Zuvor muß jedoch noch ein Überblick über das methodische Instrumentarium gegeben werden, welches dazu dient, die Strukturbäume zu einer gegebenen Grammatik aus Sätzen der von ihr beschriebenen Sprache zu rekonstruieren.

Gemäß [Goos und Waite 1984], S. 93, Theorem 5.15, existiert nun zu jeder regulären Grammatik G ein deterministischer endlicher Automat, welcher exakt dieselbe Sprache akzeptiert, wie G. Im Unterschied zu dieser arbeitet der jedoch nicht generierend (Top-Down), sondern akzeptierend bzw. konstruierend (Bottom-Up), das heißt, er bestimmt, ob eine Sequenz einem validen Satz der durch die Grammatik G beschriebenen Sprache entspricht, und liefert gegebenenfalls genau den Strukturbaum, welcher der morphologischen Struktur des Satzes entspricht. Bottom-Up-Automaten haben jedoch die Eigenschaft, daß sie im Allgemeinen deutlich umfangreicher als die dazugehörige reguläre Grammatik sind und das Regelwerk kaum mehr manuell aufzustellen ist. Stattdessen haben sie jedoch die wünschenswerte Eigenschaft, daß sie automatisch aus der grammatikalischen Beschreibung generierbar sind. Der dazu benötigte Algorithmus ist im Detail in [Goos und Waite 1984], Kapitel 7, beschrieben, und wird an dieser Stelle aus Platzgründen übergangen. Er gehört jedoch zu den elementaren Grundlagen des Übersetzerbaus und ist als solcher Element weiterführender Standardwerke des Compilerbaus sowie entsprechender Vorlesungen.

Ist einmal mit den oben erwähnten Methoden ein endlicher Bottom-Up-Automat konstruiert, so kann zu jeder Benutzerdemonstration der entsprechende Strukturbaum konnstruiert werden. Damit ist die Morphologie des Makro-Operator-Baums bestimmt. Weitergehende Information, wie sie von der Bestimmung der Vor- und Nachbedingungen der einzelnen Knoten im Strukturbaum auf allen Hierarchieebenen geliefert wird, kann nun als Berechnung von Attributen, die mit den einzelnen Knoten des Strukturbaumes verknüpft sind, formuliert werden. Dazu sind Attribut-Grammatiken die methodisch-theoretische Grundlage:

Definition 12 (Attribut-Grammatiken) *Eine Attribut-Grammatik* $AG = (\mathbf{N}, \mathbf{T}, \mathbf{P}, Z, \mathbf{A}, \mathbf{R})$ *ist eine kontextfreie Grammatik* $(\mathbf{N}, \mathbf{T}, \mathbf{P}, Z)$ *mit der Eigenschaft, daß zu jedem Symbol* $X \in \mathbf{N} \cup \mathbf{T}$ *aus der Menge der Symbole von AG eine Menge von Attributen* $\mathbf{A}(X)$ *definiert ist. Jedes dieser Attribute repräsentiert eine spezifische Eigenschaft des Symbols* X *und kann jedes Element einer vorspezifizierten Menge von Werten*

annehmen. $a \in \mathbf{A}(X)$ *wird im folgenden auch abgekürzt als* $X.a$*. Die Werte dieser Attribute werden berechnet gemäß einer Menge von Attributierungsregeln* $R(p) = \{X_i.a \leftarrow f(X_j.b, \ldots, X_k.c)\}$ *für die Produktionen* $p : X_0 \rightarrow X_1 \ldots X_n \in \mathbf{P}$ *der kontextfreien Grammatik.*

Mit Attribut-Grammatiken steht nun ein Werkzeug zur Spezifikation von Traversionssequenzen auf Strukturbäumen zur Verfügung, das eine kompakte und dennoch leicht verständliche Beschreibung der Kontextberechnungen zur Verfügung stellt. Die konkreten Instanziierungen der Attributmengen und Attribibutierungsregeln, die im Rahmen dieser Arbeit verwendet wurden, und zur Berechnung der Synchronitäts- und Kontextinformationen benutzt werden, sind in den Abschnitten 4.9 und 4.10 beschrieben.

4.8 Grammatikalische Modellierung der Makro-Operator-Morphologie

Abschnitt 4.7 beinhaltet die Grundlagen zur grammatikalischen Modellierung hierarchischer Baumstrukturen, welche als Grundlage für die Beschreibung von Makro-Operatoren dient. In diesem Abschnitt soll die konkret gewählte Grammatik bzw. Produktionenmenge motiviert und detailliert vorgestellt werden.

Bei den Benutzerdemonstrationen treten fünf verschiedene Zustände auf: Der Benutzer hat beide Hände frei, der Benutzer hat mit der rechten oder mit der linken Hand ein Objekt gegriffen, der Benutzer hat in beiden Händen ein Objekt gegriffen und diese Griffe in der Folge rechts-links erreicht, oder umgekehrt. Dies entspricht in der Grammatik der Nichtterminalmenge N = (<F>, <L>, <R>, <LR>, <RL>, <leftInteraction>, <rightInteraction>, <coordinatedInteraction>). Die Terminalmenge T besteht aus den elementaren Operationen „Griff mit links bzw. rechts“ (leftGraspEvent, rightGraspEvent), „Lösen des Griffes der linken bzw. rechten Hand“ (leftReleaseEvent, rightReleaseEvent), sowie den verschiedenen Objektinteraktionsevents, welche mit der linken, der rechten oder mit beiden Händen gleichzeitig ausgeführt werden können (leftInteractionEvent, rightInteractionEvent, coordinatedInteractionEvent). Um letztere zusammenzufassen, wurde noch das zusätzliche Interaktions-Nichtterminal $< I >$ eingeführt. Die Terminalmenge ist in direktem Zusammenhang zu den Detektoren, welche in Abschnitt 4.1, Definition 9, eingeführt wurden. Diese Ereignisse werden direkt von einem dazugehörigen Detektor aus den Benutzerdemonstrationen erkannt.

Die Produktionsregeln modellieren die Zustandsübergänge zwischen den einzelnen Griffzuständen. Sie erlauben so, eine korrekte Folge von Zuständen

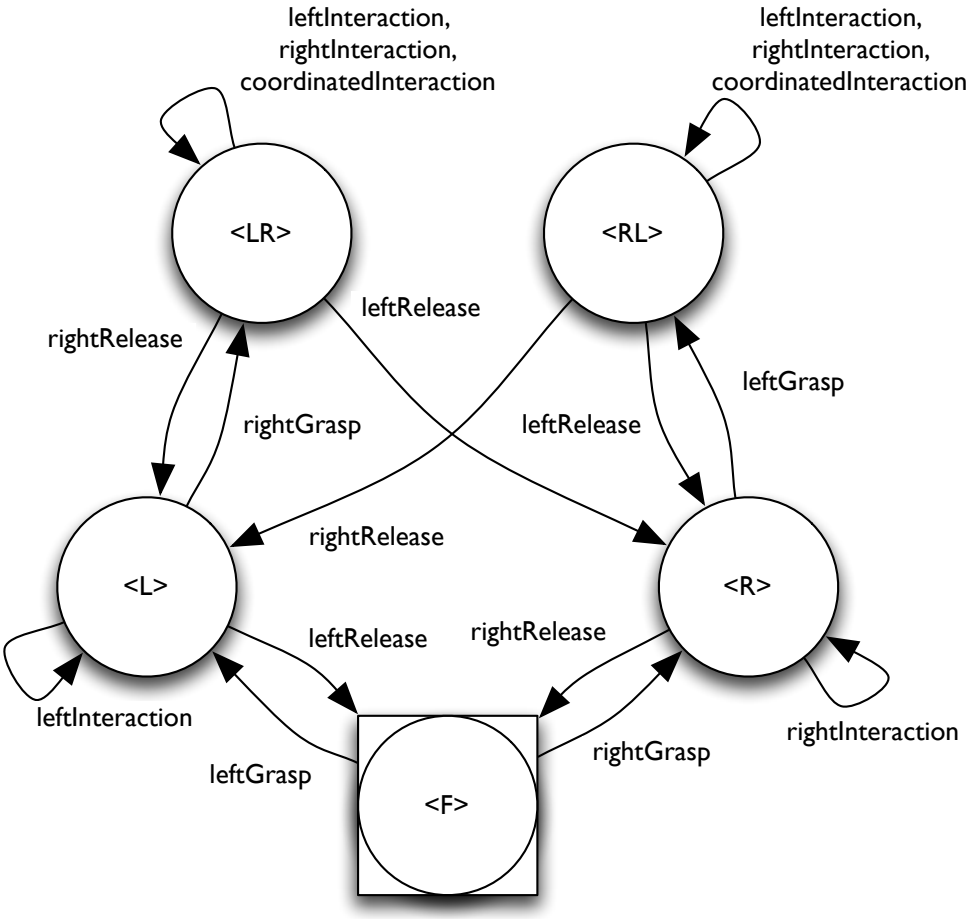


Abbildung 4.6: Graphische Darstellung der kontextfreien Grammatik zur Modellierung der strukturellen Morphologie von Makro-Operatoren.

zu erkennen, sowie den zugehörigen Makro-Operator strukturell und morphologisch zu determinieren. Damit steht die Struktur des Makro-Operators fest. Um die funktionale Beschreibung der Benutzerdemonstration zu erhalten, müssen verschiedene Informationen innerhalb des Strukturbaumes berechnet werden, zum Beispiel Synchronisationspunkte oder relationale Beschreibungen der Benutzerintention. Dies wird mittels der in Abschnitt 4.7 definierten Attributierten Grammatiken vorgenommen. Detaillierte Beschreibungen sind in beiden folgenden Abschnitten gegeben.

<F>	→	<rightGrasp> <R>
<R>	→	<rightRelease> <F>
<R>	→	<rightInteraction> <R>
<F>	→	<leftGrasp> <L>
<L>	→	<leftRelease> <F>
<L>	→	<leftInteraction> <L>
<L>	→	<rightGrasp> <LR>
<LR>	→	<righRelease> <L>
<LR>	→	<leftRelease> <R>
<LR>	→	<I> <LR>
<R>	→	<leftGrasp> <RL>
<RL>	→	<leftRelease> <R>
<RL>	→	<rightRelease> <L>
<RL>	→	<I> <RL>
<I>	→	<leftInteraction>
<I>	→	<rightInteraction>
<I>	→	<coordinatedInteraction>
<rightGrasp>	→	move* rightGraspEvent move*
<rightRelease>	→	move* rightReleaseEvent move*
<leftGrasp>	→	move* leftGraspEvent move*
<leftRelease>	→	move* leftReleaseEvent move*
<leftInteraction>	→	move* leftInteractionEvent move*
<rightInteraction>	→	move* rightInteractionEvent move*
<coordinatedInteraction>	→	<left> <right>
<left>	→	move* coordinatedInteractionEvent move*
<right>	→	move* coordinatedInteractionEvent move*

Tabelle 4.1: Produktionsregelmenge **P** der kontextfreien Grammatik zur Modellierung der strukturellen Morphlogie von Makro-Operatoren. Instanziierung gemäß der Definition 10.

4.9 Synchronitätsmodell

Zur Ausführung von Makro-Operatoren sind Informationen über die Synchronisierbarkeit von Aktionen von hoher Relevanz. Sie erlauben es Robotern, die über zwei Arme verfügen, Feinmanipulationen auszuführen, die über einen hohen Grad von Synchronisationsbedarf zwischen den beiden Manipulatoren verfügen.

Solcher Synchronisationsbedarf kann auftreten, wenn beispielsweise mit der einen Hand ein Kühlschrank geöffnet und geschloßen werden soll, und

mit der anderen Hand während des Öffnungszeitraumes ein Objekt im Kühlschrank gegriffen und herausholt werden soll. Dies ist ein durchaus praxisnahes Szenario, vgl. [Zöllner 2005]. Eine weitere Situation, wo feingranulare Synchronisation von Bedeutung ist, ist das zweihändige Einschenken, wobei das Objekt, in welches eingeschenkt wird, von einer Hand gehalten wird. In beiden Fällen ist ein Fehlschlag in der Synchronisation der beiden Teilhandlungen fatal: Im letzteren wird eventuell Flüssigkeit verschüttet, da das aufnehmende Gefäß nicht an der korrekten Position relativ zu dem Flüssigkeitsbehälter ist. Im ersten Beispiel kann bei Ausführung des Vorgangs zum Greifen und Herausholen des Objekts im Kühlschrank selbiger und/oder der Robotermanipulator beschädigt werden, falls die Kühlschranktüre noch oder schon wieder geschloßen ist.

Deshalb ist ein feinkörniger Abgleich zwischen den Händen bei zweihändig ausgeführten Bewegungen nötig. Dazu werden sogenannte Synchronisationspunkte eingeführt. Diese haben die Eigenschaft, daß an diesen Stellen die Ausführung der Handlung mit einer Hand eingefroren wird, bis beide Hände wieder synchron sind oder bis die andere Hand in der Wartestellung ist.

Als Synchronisationspunkte eignen sich die Endpunkte aller Elementaroperatoren, also aller Bewegungs-Operatoren sowie der genaue Zeitpunkt der detektierten Objektinteraktion. Will man die Synchronisationspunkte für die gesamte koordinierte Aktion bestimmen, so kann dazu die in Tabelle 4.2 aufgetragenen Attributierungsregeln verwenden. Der Datenfluß für ein beispielhaftes Interaktionsereignis ist in Abbildung 4.7 dargestellt.

Mit dem hier vorgestellten Verfahren ist es möglich, auch komplexere zweihändige Handhabungen zu synchronisieren, indem Attributierungsregeln ausgewertet werden.

4.10 Funktionale Kontextanalyse

Abschnitt 4.7 zeigte, wie die Morphologie der in Abschnitt 3.3 eingeführten hierarchisch-funktionalen Wissensrepräsentation aus einer Benutzerdemonstration abgeleitet werden kann. Der funktionale Aspekt, der für weitere Interpretationsschritte von hoher Relevanz ist, fehlt jedoch bisher. Er umfaßt die Berechnung der Effekte einer Handlung auf eine Szene sowie die Schätzung der Vorbedingungen, die gegeben sein müssen, um jene Handlung ausführen zu können.

Sowohl die Vorbedingungen der einzelnen Handlungen als auch deren Effekte und Auswirkungen werden in Termen über den in Abschnitt 3.4 beschriebenen geometrischen und objektbasierten Relationen beschrieben. Die Gesamtmenge an Relationen wird bei allen Greif- und Loslaßoperationen

rule <coordinatedAction> → <left> <right>
attribution
<coordinatedAction>.sync-points = (<left>.sync-points
<right>.sync-points)

rule <left> → <move-list> coordinatedInteractionEvent <move-list>
attribution
<left>.sync-points = (<move-list>[1].sync-points
coordinatedInteractionEvent.t_{end}
<move-list>[2].sync-points)

rule <right> → <move-list> coordinatedInteractionEvent <move-list>
attribution
<right>.sync-points = (<move-list>[1].sync-points
coordinatedInteractionEvent.t_{end}
<move-list>[2].sync-points)

rule <move-list> → <move> <move-list>
attribution
<move-list>[0].sync-points = (<move.t_{end} <move-list>[1].sync-points)

rule <move-list> → <move>
attribution
<move-list>.sync-points = <move.t_{end}

Tabelle 4.2: Attributierungsregeln zur Berechnung der Synchronisationspunkte bei koordinierten Bewegungen.

ausgewertet. Dies geschieht in der graphischen Simulationsumgebung KAVIS [Schaude 1996], auf die das hier beschriebene System aufbaut. So ist es möglich, mit Zeitstempeln versehene Schnappschüsse der Umweltbeschreibung zu generieren, aus der die relevanten Effekte und Vorbedingungen extrahiert werden können.

Gemäß Definition 6 ist eine Demonstration D in der Welt $\mathbf{W}$ als eine Folge von Zuständen definiert:

$$D = (\mathbf{S}^0_{\mathbf{W}}, \mathbf{S}^1_{\mathbf{W}}, \ldots, \mathbf{S}^n_{\mathbf{W}}).$$

Jedem dieser Zustandsübergänge $\mathbf{S}^i_{\mathbf{W}} \mapsto \mathbf{S}^{i+1}_{\mathbf{W}}$ kann jetzt ein Teilbaum k_i des Makrobaumes mit der Wurzel k_0 zugeordnet werden, welcher genau diesen

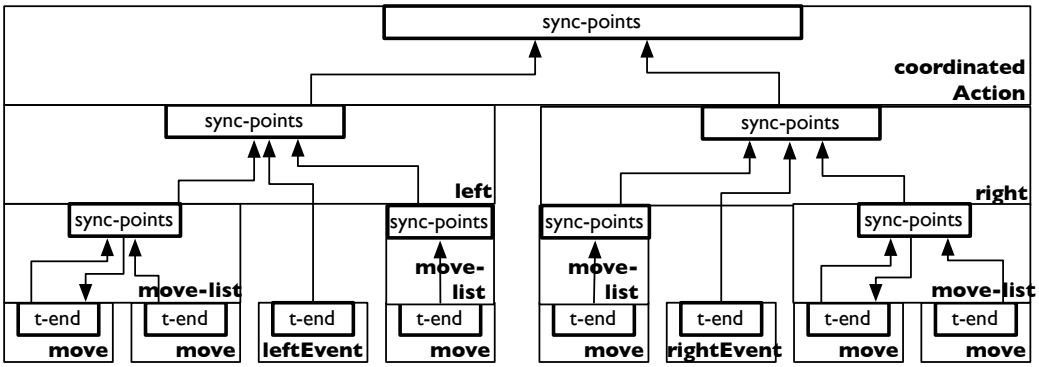


Abbildung 4.7: Datenfluß zur Berechnung der Synchronisationspunkte bei koordinierten zweihändigen Bewegungen. Auf der untersten Ebene sind die Elementaroperatoren aufgetragen, die höheren Schichten gehen bis zur Ebene des Nichtterminals <coordinatedInteraction>. Nichtterminale sind als dünn umrandete Rechtecke dargestellt, zu ihnen gehörige Attribute sind darin enthaltene fett umrandete Rechtecke. Datenflüße sind als gerichtete Pfeile dargestellt.

Zustandsübergang bewirkt. Jeder dieser Teilbäume ist also mit einem Attribut c versehen, so daß c genau den mit dem Operator k_i erzielten Veränderungen in der Szene übereinstimmt. Diese Veränderungen sind äquivalent mit dem Differenz zwischen dem Zustand nach Ausführung des Operators und dem Schnitt der Zustände vor und nach dem Operator. Es gilt also

$$k_i.c = \mathbf{S}_\mathbf{W}^{i+1} \setminus (\mathbf{S}_\mathbf{W}^{i+1} \cap \mathbf{S}_\mathbf{W}^{i}).$$

Durch Forwärtspropagierung (siehe Abbildung 4.8) dieser einzelnen Veränderungen lassen sich die einzelnen Veränderungen zu einem Gesamteffekt zusammenfassen. Dazu müssen Attributierungsregeln generiert werden, welche folgende Eigenschaften haben:

$$\begin{aligned} k_1.e &= k_1.c \\ k_{i+1}.e &= k_i \setminus \overline{k_i.c} \cup k_i.c \\ k_0.e &= = k_n.e \end{aligned}$$

Diese Regeln erlauben eine Forwärts-Berechnung des Gesamteffektes einer Demonstration. Um jedoch diese von Spontanaktionen zu befreien sowie um die Beiträge jeder einzelnen Teilhandlung zu den Gesamteffekten berücksichtigen zu können, muß nun eine Rückwärtsberechnung der gewünschten Effekte erfolgen. Dies Erfolgt über das Attribut der sogenannten erwünschten

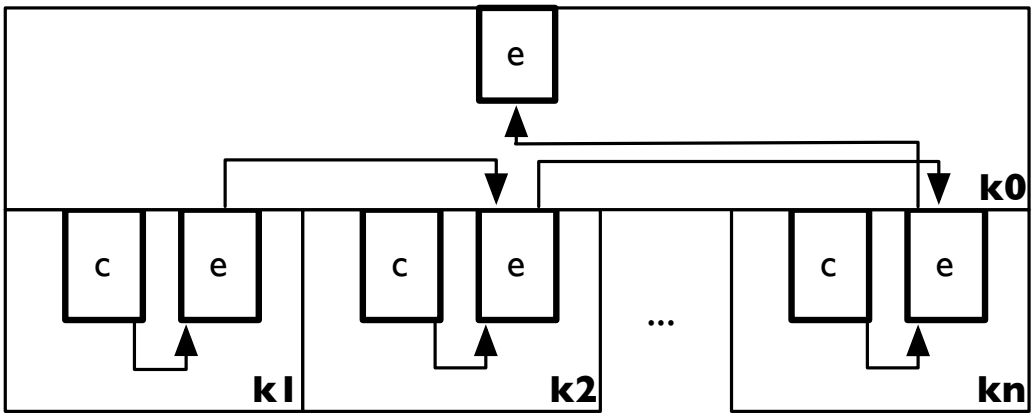


Abbildung 4.8: Forwärtspropagierung der Änderungen, die durch eine Operatorsequenz getätigt werden, zu einem Gesamteffekt. Änderungen sind dabei mit c (changes), Effekte mit e abgekürzt.

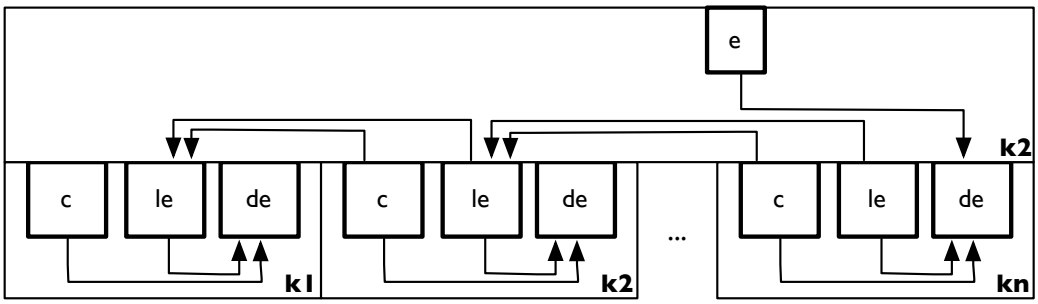


Abbildung 4.9: Rückwärtspropagierung der Effekte, um aus dem gesamten Effekt zu jedem Operator die jeweiligen Beiträge zu erhalten.

Effekte *de*, die mit Hilfe der Liste aller Effekte *le*, die noch offen, d.h. von bisher keinem Teilbaum erbracht worden sind, berechnet werden:

$$\begin{aligned} k_n.le &= k_0.e \\ k_i.le &= k_{i+1}.le \setminus k_{i+1}.c \\ k_i.de &= k_i.le \cap k_i.c \end{aligned} \tag{4.7}$$

Diese Berechnungen erlauben eine einfache Detektion von Spontanaktionen, die nichts zum Gesamteffekt beitragen, bzw. deren Effekte von darauffolgenden Aktionen eliminiert werden. Falls nämlich die erwünschten Effekte $k_i.de$ der leeren Menge $\emptyset$ entsprechen, so kann die Aktion weggelassen werden,

ohne daß die Demonstration eine andere Wirkung auf die Umwelt gehabt hätte. Dies führt zu einer Elimination unerwünschten Verhaltens und somit zu einem sparsamen Umgang mit den endlichen Systemressourcen Prozessorzeit und Arbeitsspeicher.

4.11 Zusammenfassung

Dieser Abschnitt präsentierte die ersten zwei Stufen des Interpretationsprozesses für menschliche Demonstrationen von Handlungen im Haushaltsbereich. Im ersten Schritt werden bestimmte Ereignisse durch sogenannte Detektorennetzwerke erkannt und in ihrer zeitlichen Ausdehnung bestimmt. Initiale Information zu Griff- und Loslassbewegungen des Menschen sowie zu Objektinteraktionen wird gesammelt, wie beispielsweise die Greifart, die verwendet wird, um ein Objekt zu ergreifen. Segmentierungsschritte erlauben es nun, die gesamte Demonstration in unterschiedliche Stücke mit einem jeweils eigenen, zusammenhängenden Kontext zu unterteilen. Sprachliche Kommentare, die während des Demonstrationsprozesses auftreten, werden den jeweiligen Demonstrationsteilen zugeordnet, damit die verbale Information für spätere Analyseschritte zur Verfügung steht und in geeigneter Form angewendet werden kann.

Die Bestimmung der morphologischen Struktur der Demonstration nahm den Rest des vorliegenden Kapitels ein. Hier wurde auf herausragend theoretisch fundierte Erkenntnisse aus dem Übersetzerbau zur Modellierung komplexer Eingabeströme, wie sie ein Sensorsystem zum Programmieren durch Vormachen darstellt, zurückgegriffen. Darauf aufbauend wurde ein morphologisch-grammatikalisches Modell des Handlungswissens erstellt, welchem sämtliche Eingabedaten genügen müssen. Mit fortgeschrittenen Parsingtechniken konnten diese Modelle verwendet wurden, um die strukturelle Analyse, die Etablierung von Synchronisierungspunkten und die Berechnung der Benutzerintentionen hinter der Demonstration zu erledigen.

Nach dem Durchlaufen der lexikalischen und der syntaktischen Analysephase steht ein vollkommen fertiger Makro-Operator zur Verfügung und harrt seiner Einordnung in den Gesamt-Erfahrungsschatz des Systems. Dieser erfolgt in der darauf aufbauenden semantischen Analysephase, in welcher eine Relevanzbewertung des Gelernten stattfindet und ein Ähnlichkeitsabgleich mit vorhandenem Handlungswissen durchgeführt wird.

Kapitel 5

Semantische Analyse

In den vorangegangenen drei Abschnitten wurde deutlich, dass gelerntes Aufgabenwissen einerseits angewandt werden muss, mit dem Ziel, zur Ausführung verfügbar zu sein, andererseits aber auch für eine Erweiterung, Veränderung oder Anpassung an eine Demonstration einer neuen oder die erneute Demonstration einer bereits gelernten Aufgabe zur Verfügung stehen muss. Die letzte Unterscheidung erfordert von den lernenden Komponenten eines PdV-Systems die nichttriviale Entscheidung, ob eine gerade beendete Demonstration die erneute Vorführung eines gelernten Tasks war oder ob eine bisher unbekannte Aufgabe ausgeführt wurde und ein neuer Task gelernt werden soll.

Es erscheint daher sinnvoll, eine Strukturierung des in Form von unterschiedlichen gelernten Aufgaben vorhandenen Handlungswissens vorzunehmen. Diese sollte folgende drei Funktionen unterstützen:

1. Die Bewertung von Ähnlichkeiten von Handlungswissen gemäß vorgegebener Metriken, das Herstellen von Bezügen zu anderem, bereits gelerntem Handlungswissen, und das Einbetten von Handlungswissen in ein Umfeld von ähnlichen oder gleichen Tasks (*Categorization*). Hierzu werden hierarchische Ähnlichkeitsmaße eingesetzt, wie sie in Abschnitt 5.2 vorgestellt werden.

2. Das Eingliedern von neuem Handlungswissen, entweder in Form eines neuen Tasks oder der Hinzunahme einer neuen Demonstration zu einer bereits bekannten Aufgabe, sowie die damit verbundenen Prozesse des Lernens und der Aktualisierung von Handlungswissen (*Learning*). Hierzu werden Verfahren betrachtet, wie sie in Abschnitt 5.3 dargestellt werden, sowie Reasoning-Methoden, wie sie das darauf folgende Kapitel vorstellt.

3. Dem Wiederauffinden gelernten Handlungswissens, entweder um es mit neuen Aufgabenvorführungen zu vergleichen und in Bezug zu setzen (siehe 1.), oder um es zur Ausführung in einem bestimmten Umweltkontext zu bringen (*Retrieval*).

Die Anforderungen, die sich aus diesen drei Aspekten ergeben, legen es nahe, eine hierarchische Klassenkategorisierung zur Strukturierung des gelernten Handlungswissens zu verwenden. Die zu erstellende hierarchische Aufgabentaxonomie besteht aus einem Kategorienbaum, dessen innere Knoten Klassen bzw. Unterklassen von Aufgaben repräsentieren, die Blätter jedoch konkrete Instanzen von Aufgaben, d.h. Benutzervorührungen, die von dem PdV-System gelernt wurden. Zur Einteilung einer Menge von Vorführungen in eine hierarchische Taxonomie können unterschiedliche Kriterien angewandt werden. Die einfachste Möglichkeit besteht sicher darin, den Benutzer jeder Demonstration einen Namen zuweisen zu lassen. Über die Äquivalenz der Namen lässt sich eine einfache, flache Hierarchie von Klassen realisieren. Beispielsweise können unterschiedliche Ausprägungen der Aufgabe „Tischdecken“, die als solche bezeichnet werden, zu einer Kategorie zusammengefasst werden und von anderen z.B. mit der Bezeichnung „Spülmaschine ausräumen“ unterschieden werden. Diese sehr einfache Kategorisierung ist jedoch aus verschiedenen Gründen suboptimal. Einerseits ist die Klassifikation seitens des Benutzers zu unscharf und andererseits wird durch eine einfache Benennung nur eine einschichtige Hierarchie induziert, wodurch ein wesentlicher Vorteil der hierarchischen Strukturierung verschenkt wird. Weiterhin ist es sinnvoll, Benutzerdemonstrationen zum Lernen zuzulassen, die nicht mit einem Namen markiert sind. Dies kann von Nutzen sein, wenn der Roboter den Menschen beim Durchführen von Aufgaben beobachtet, ohne vom Benutzer explizit in einen Lernmodus versetzt worden zu sein. Solchermaßen kann eine Menge von Handlungswissen auf eine sehr benutzerfreundliche Art und Weise akquiriert werden (*Unüberwachtes Lernen*).

Um eine (zumindest teilweise) unüberwachte Kategorisierung durchzuführen, werden Lernverfahren zur hierarchischen Konzepterstellung durchgeführt. Hierbei geht es darum, Ballungen[1] innerhalb einer Menge von Demonstrationen zu finden, d.h. für eine Menge von Demonstrationen $\mathbf{D}$ eine Partitionierung $\mathbf{D}_1, \mathbf{D}_2, ...\mathbf{D}_n$ zu finden, und jede der einzelnen Mengen wieder rekursiv weiter zu partitionieren, bis die entstehenden Mengen nur noch aus einzelnen Demonstrationen bestehen.

Da die Menge der Demonstrationen sich jedoch über die gesamte Einsatzzeit eines PdV-Systems stetig vergrössert, sie also nicht a-priori partitioniert werden kann, sollen inkrementelle Verfahren dazu verwendet werden, neu

[1] engl. cluster

hinzugefügte Aufgabenvorführungen in eine bereits bestehende Aufgabentaxonomie einzugliedern und diese mit Blick auf ein Optimalitätskriterium zu verändern. Als Optimalitätskriterium bieten sich Methoden zur Bewertung der *Kategorienützlichkeit*[2] an (siehe [Gennari et al. 1989]). Die Idee hinter der Kategorienützlichkeit besteht darin, die Entropie innerhalb einer Klasse zu minimieren. Eine Klasse ist umso nützlicher, je geringer die Entropie der Merkmale der in ihr enthaltenen Entitäten, in diesem Fall der Aufgaben, ist.

Im folgenden wird zuerst die kleinste Einheit semantischen Verständnisses vorgestellt: Das einzelne Merkmal und dessen Relevanz für eine bestimmte Aufgabenklasse. Der folgende Abschnitt 5.2 ist der Erkennung von Analogien bzw. Ähnlichkeiten zwischen unterschiedlichen Demonstrationen, welcher der Benutzer dem System gibt, gewidmet. Hier werden die zentralen Konzepte von Ähnlichkeit und von Distanzmaßen vorgestellt. Anschließend wird der prinzipielle Aufbau der Gedächtnisstruktur behandelt, bevor in Abschnitt 5.4 die algorithmische Grundlage zum Aufbau einer Handlungstaxonomie vorgestellt wird.

5.1 Merkmalsbewertung

Jede Demonstration einer Aufgabe und damit jeder mit den Methoden aus Abschnitt 4.10 erzeugte Makro-Operator besitzt verschiedene Merkmale, in denen die entscheidenden Eigenschaften der Aufgabe enthalten sind. Diese können umfassen:

- Die Anzahl und Art der verwendeten Objekte
- Die räumliche Anordnung der Objekte nach der Durchführung der Aufgabe
- Zustandsänderungen der manipulierten Objekte
- Die Art und Weise, wie Objekte manipuliert werden, z.B. Verfahrtrajektorien, verwendete Griffarten, eingenommene Körperhaltung/Pose während des Manipulationsvorgangs oder den Fokus der Aufmerksamkeit des Benutzers.

Je nach Art der gerade behandelten Aufgabe sind unterschiedliche dieser Merkmale relevant. Soll beispielsweise das Malen eines bestimmten Buchstabens das Lernziel sein, so ist der Fokus auf die durchgeführte Trajektorie zu legen, geht es hingegen darum den Tisch zu decken, so kommt es weniger

[2] engl. category utility

darauf an, mit welchen Trajektorien dies geschieht, sondern vielmehr darauf, welche Objektanordnung nach Ende der Demonstration gegeben ist. Jede einzelne Demonstration besteht nun aus intentionalen Aspekten und willkürlich erzeugten beziehungsweise irrelevanten Merkmalen. Aus einer einzigen Demonstration ist nicht zu erkennen, welche der erkannte Merkmale relevant sind und welche nicht. Stehen jedoch mehrere Demonstrationen derselben Aufgabe zur Verfügung, so ist davon auszugehen, daß sich die relevanten Merkmale wiederholen, da deren Wahrscheinlichkeit für eine identische Reproduktion sehr hoch sind. Mit ähnlicher Argumentation kann man erwarten, daß die irrelevanten Merkmale eine deutlich höhere Streuung aufweisen, das heißt, daß sich die Merkmale in den irrelevanten Merkmalen unterscheiden. Somit erscheint es möglich, daß unter Betrachtung mehrerer Demonstrationen derselben Aufgabe auf die relevanten Aspekte geschlossen werden kann und diese als „Kern“ der Aufgabe extrahiert werden können.

Eine wesentliche Kenngrösse zur Schätzung der Relevanz eines Merkmals ist also dessen Auftretenswahrscheinlichkeit in einer Menge von Demonstrationen der selben Aufgabenklasse $\mathbf{C}$ deren Definition folgendermaßen aussieht:

Definition 13 *Die Auftretenswahrscheinlichkeit für ein Merkmal m in einer Menge von Demonstrationen $\mathbf{C} = \{D_1, D_2, ..., D_n\}$ definiert als*

$$p(m, \mathbf{C}) = \frac{|\{D \in \mathbf{C} | m \in D\}|}{|\mathbf{C}|}$$

Es ist jetzt also eine Bewertungsfunktion zu entwerfen, welche einer Auftretenswahrscheinlichkeit innerhalb einer Aufgabenklasse ein Relevanzmaß bzw. ein Gewicht zuordnet, welches in Bezug zu dem Einfluß des Merkmals auf die Taskklasse steht, welcher die Demonstration angehört. Die hier gewählte Möglichkeit besteht darin, die Gewichte eines Merkmals gemäß seines Informationsgehaltes festzulegen.

Nach dem Shannon-Theorem (siehe [Duda et al. 1998]) ist der Informationsgehalt eines einzelnen Merkmals nach dessen Logarithmus bestimmt - also der Größe

$$-\log_2 p(m, \mathbf{C}).$$

Dieses Maß favorisiert jedoch Merkmale mit niedriger Auftrittswahrscheinlichkeit - also ist eine Umkehrung der Auftretenswahrscheinlicheit nötig, nach der die klassenbedingte Schätzung der Merkmalsrelevanz $w(m)$ folgendermaßen aussieht:

$$w(m) = -\log_2 \left(1 - p(m, \mathbf{C})\right). \tag{5.1}$$

Wenn eine ausreichend große Menge an Demonstrationen derselben Aufgabe zur Verfügung stehen, so ist davon auszugehen, daß dann eine ausreichend gute Schätzung der Merkmalsrelevanz durchgeführt werden kann (siehe [Billard et al. 2004]). In dem Fall, daß ein völlig neuer, dem System bisher unbekannter Task gelernt werden soll, ist jedoch diese Vorbedingung einer großen Trainingsdatenmenge nicht gegeben. Am Anfang dieses Lernprozesses sind statistische Kenndaten unverläßlich und nicht für belastbare Aussagen zu gebrauchen.

In diesem Lernstadium ist auf andere Kenndaten zur Bewertung der Merkmalsrelevanz zurückzugreifen. Als solche bietet sich die Verwendung von Hintergrundwissen an. Dieses Hintergrundwissen kann beispielsweise Grundannahmen über die allgemeine Merkmalsverteilung sein. Diese Form von Hintergrundwissen kann angewandt werden, wenn man postuliert, daß der Benutzer, wenn er die erste Demonstration einer neuen Aufgabe demonstriert, vor allem an den Merkmalen der Demonstration interessiert ist, welche sich von dem Hintergrundwissen unterscheiden. Somit erscheint es sinnvoll, die Relevanzbewertung als eine Diskriminationsfunktion auzufassen. Demnach ist die Relevanzbewertung anhand globaler Diskrimination als

$$w(m) = -log_2 p(m, \overline{\mathbf{C}})$$

vorzunehmen. Hierbei bezeichnet $\overline{\mathbf{C}}$ die Menge aller Demonstrationen (in Gegensatz zu den vorher verwendeten Einschränkung der Menge aller Demonstrationen, die nur zu dieser Klasse gehören). Dieses Maß favorisiert also Merkmale, die in der Gesamtmenge allen bekannten Handlungswissens selten vorkommen, was genau der Bevorzugung der Merkmale, die die neue Demonstration vom bisher bekannten unterscheiden, entspricht.

Wie kann jetzt jedoch ein allgemeingültiges Maß für die Merkmalsbewertung gefunden werden, welches das jeweilige Lernstadium berücksichtigt, also am Anfang bei einem neuen Task zur Merkmalsbewertung hauptsächlich auf Hintergrundwissen zurückgreift, bei zunehmenden aufgabenspezifischen Wissen in Form zusätzlicher Demonstrationen jedoch zunehmend auf dieses taskspezifische Wissen „umschaltet“?

Hierzu ist es nötig, ein Maß für die Vollständigkeit des verfügbaren Wissens zu entwickeln. Man nimmt an, daß das taskspezifische Wissen mit jeder Demonstration steigt und bei dem Vorhandensein von vier bis fünf Demonstrationen ausreichend Material vorhanden ist, um die relevanten Merkmale einer Aufgabe verläßlich bestimmen zu können. Diese Annahme wird gestützt von den Arbeiten in [Calinon et al. 2006]. Eine Übergangsfunktion zum aufgabenspezifischen Wissen sollte also von 0 angefangen bis nahezu 1 bei einem Umfang taskspezifischen Wissens von mehr als 6 Demonstrationen

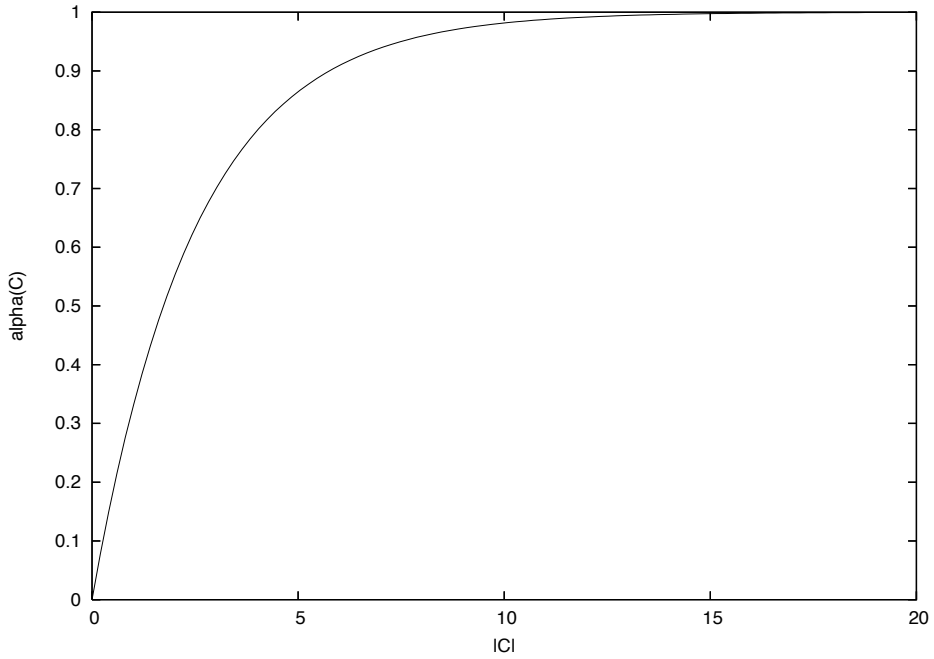


Abbildung 5.1: Übergangsfunktion $\alpha(\mathbf{C})$ vom Hintergrundwissen zum aufgabenspezifischen Wissen. α steigt mit dem Umfang von $\mathbf{C}$, bis sie sich an 1 annähert.

steigen. Gemäß [Pardowitz et al. 2006b, Pardowitz et al. 2006a] wurde folgende Übergangsfunktion gewählt:

$$\alpha(\mathbf{C}) = 1 - e^{-k \cdot |\mathbf{C}|}$$

Diese Funktion ist in Abbildung 5.1 dargestellt. Sie erlaubt es, wenn wenige bis gar keine Beispiele für eine bestimmte Aufgabe vorhanden sind, das aufgabenspezifische Handlungswissen mit 0 zu bewerten, während das unabhängige Hintergrundwissen mit dem Faktor $1 - \alpha = 1$ in die Relevanzbetrachtung eingeht. Auf der Gegenseite, wenn mehr als 5 Demonstrationen derselben Aufgabe vorhanden sind, wird das taskspezifische Wissen mit einem hohen Wert gewichtet, während das Hintergrundwissen nahezu vernachlässigt wird. k wurde im Rahmen dieser Arbeit zu $-\frac{\log_2 0.25}{5}$ gewählt. Es ist also möglich, mittels hoher Anzahlen von Wiederholungen, das Hintergrundwissen „lokal“ zu überschreiben.

Nun kann man die gesamte Relevanzbewertungsfunktion aufstellen, welche wie folgt definiert ist:

$$w(m) = -\left[\alpha(\mathbf{C}) \log_2 (1 - p(m|\mathbf{C})) + (1 - \alpha(\mathbf{C})) \log_2 p(m|\overline{\mathbf{C}})\right]$$

Diese Funktion ist aus zwei wesentlichen Teilen aufgebaut: der Auswertung des klassenspezifischen Wissens $-\log_2(1-p(m|\mathbf{C}))$, die diejenigen Merkmale favorisiert, welche häufig in Demonstrationen einer Aufgabe $\mathbf{C}$ auftreten, sowie der Auswertung des Hintergrundwissens $-\log_2 p(m|\overline{\mathbf{C}}$. Der letztere Term favorisiert genau diejenigen Merkmale, welche bisher selten im Hintergrundwissen auftreten, d.h. diejenigen, in denen sich die Demonstrationen von altem Wissen unterscheiden. Die beiden Terme sind normiert mit der Gewichtungsfunktion $\alpha(\mathbf{C})$ bzw. $(1-\alpha(\mathbf{C}))$ zwischen dem klassenspezifischen bzw. dem Hintergrundwissen.

Diese Bewertungsfunktion erfüllt alle gewünschten Eigenschaften: Sie gewichtet in Abhängigkeit der vorhandenen Demonstrationen entweder das Hintergrundwissen oder das aufgabenspezifische Wissen höher, wobei sie bei vielen vorhandenen Demonstrationen diejenigen bevorzugt, die eine hohe Ähnlichkeit mit den vorhandenen aufweisen, bei wenigen Demonstrationen diejenigen Merkmale hervorhebt, die sich vom bisherigen Hintergrundwissen unterscheiden (siehe [Pardowitz et al. 2007]).

Damit ist die wesentliche Grundlage geschaffen, einzelne Merkmale bei vorhandener Klassenzuordnung nach ihrer Relevanz zu bewerten. Im folgenden Kapitel wird untersucht, wie diese Relevanzbetrachtungen eingesetzt werden können, um ganze Aufgabendemonstrationen hinsichtliche ihrer Ähnlichkeit oder Unterschiedlichkeit zu bewerten.

5.2 Ähnlichkeitsanalyse und Distanzmaße

Die vorgeschlagenen Relevanzmaße sollen im folgenden dazu verwendet werden, die Ähnlichkeit zwischen Makro-Operatoren, also zwischen unterschiedlichen Bestandteilen des gesamten Handlungswissens, welches ein Robotersystem besitzt, zu bewerten. Dies ist notwendig, um Wissen unter Zuhilfenahme unterschiedlicher Demonstrationen zu generieren, also über Demonstrationsgrenzen hinweg Parallelitäten zu erkennen und anschließend auszunutzen. Dieses Ähnlichkeitsmaß sollte hierarchisch gegliedert sein, um sowohl zur Bewertung der Ähnlichkeit ganzer Demonstrationen, als auch zur Reidentifikation ähnlicher Demonstrationsteile anwendbar zu sein. Weiterhin sollte es die hierarchisch-funktionale Struktur des vorgeschlagenen Aufgabenmodells wiederspiegeln.

Die Bestimmung des Ähnlichkeitsmaßes erfolgt anhand folgender zwei Klassen von Merkmalen der Handlungen:

- Objektmerkmale: Diese beschreiben die Objekte, die in der Handlung verwendet, manipuliert oder referenziert werden, wie z.B. Tassen, Fla-

schen, Teller, etc. sowie deren Eigenschaften, Rollen oder interne Zustände.

- Inter-Objekt-Merkmale: Diese beinhalten die geometrischen Relationen, die als Vor- und Nachbedingungen vor oder nach der Ausführung von Teiloperationen erfüllt sind oder sein müssen. Diese Relationen bestehen aus linguistischen Variablen wie „Über“, „Unter“, „Rechts von...“ oder „Ausgerichtet an...“.

Die Ähnlichkeit zweier Merkmalsausprägungen m_1, m_2 wird durch ein Maß für die Ähnlichkeit auf Merkmalsebene bestimmt. Nimmt man an, daß die Merkmale auf das Einheitsinterval beschränkt sind, also $0 \leq m_1, m_2 \leq 1$, so bietet sich eine Ähnlichkeitsfunktion bzw. unscharfe Gleichheitsfunktion s an mit

$$s(m_1, m_2) = (1 - |m_1 - m_2|)^2.$$

Diese Funktion hat die Eigenschaft, daß gleiche Merkmalsausprägungen in einer Ähnlichkeit von 1 resultieren, wohingegen Merkmale mit maximal unterschiedlichen Ausprägungen zu einem Wert von 0 evaluiert werden. Alle Werte dazwischen werden auf das Einheitsinterval $[0; 1]$ abgebildet.

Eine naive Methode, die Ähnlichkeiten zweier Teiloperationen oder Aufgaben zu bewerten, besteht darin, die Ähnlichkeitsmaße aller einzelner Merkmale multiplikativ zu verknüpfen, also

$$sim_1(x, y) = \prod_{l=1}^{n} s(x_l, y_l)$$

für zwei Merkmalsvektoren x, y aus dem Merkmalsraum $\mathbf{R}^n$.

Es stellte sich jedoch heraus, dass bei dieser Wahl irrelevante Merkmale das Ähnlichkeitsmaß dominieren und zu einer zu starken Verrauschung und damit unbrauchbaren Ergebnissen führen. Weiterhin ist an diesem Maß zu bemängeln, daß es die mit viel Mühe geschätzten unterschiedlichen Relevanzen für jedes einzelne Merkmal nicht berücksichtigt.

Als Alternative bietet es sich an, die Merkmale gemäß ihrer Relevanzschätzung zu gewichten. Mit dem Relevanzmaß $w(m)$ für ein Merkmal m wird damit das Ähnlichkeitsmaß sim zu zwei Merkmalsvektoren x und y:

$$sim_w(x, y) = \prod_{l=1}^{n} (w_l^2 + s(x_l, y_l)^2 - w_l^2 s(x_l, y_l)^2), \tag{5.2}$$

(siehe [Pedrycz und Rocha 1993]). Dies entspricht der Multiplikation der Ergebnisse der Anwendung des unscharfen Lukasiewicz-Oder-Operators auf die Merkmale und ihrer dazugehörigen Gewichte, wie sie in Abschnitt 5.1 eingeführt wurden.

Aufbauend auf diesem Grundmaß für die Ähnlichkeit von Mengen von Merkmalen, läßt sich ein zusammengesetztes Maß für die Ähnlichkeiten von Handlungsdemonstrationen definieren, welches es erlaubt, ähnliche Aufgaben zu identifizieren und diese zu hierarchischen Repräsentationen des gesamten Ausführungswissens zusammenzufügen, sowie bereits bekannte Aufgaben und Teiloperationen in neuen Demonstrationen wiederzuerkennen.

Dies geschieht durch Mittelung über die Ähnlichkeit der Vor- und Nachbedingungen (s_{pre}, s_{post}), sowie der durchschnittlichen Ähnlichkeit der Nachfolger eines Makro-Operators (s_{succ}). Diese sind folgendermaßen aufgebaut:

$$\begin{aligned} s_{pre} &= sim_w(Precond(D_1), Precond(D_2)) \\ s_{post} &= sim_w(Postcond(D_1), Postcond(D_2)) \\ s_{succ} &= \frac{1}{|Successors(D_1)|} \sum_{A \in Successors(D_1), B \in Successors(D_2)} sim_w(A, B) \end{aligned} \tag{5.3}$$

wobei sim das gesamte resultierende Ähnlichkeitsmaß ist, welches definiert ist zu

$$sim(D_1, D_2) = \frac{1}{3}\left[s_{pre}(D_1, D_2) + s_{post}(D_1, D_2) + s_{succ}(D_1, D_2)\right]$$

Aus der Definition ersieht man leicht, daß dies ein rekursives normiertes Ähnlichkeitsmaß ist, d.h. es greift zur Ähnlichkeitsschätzung auf die hierarchisch-funktionale Baumstruktur der Repräsentation des Handlungswissens als Makro-Operator zurück und liefert für maximal disjunkte Eingaben einen Ähnlichkeitswert von 0, für identische Eingaben eine Ähnlichkeit von 1 und für alle anderen Eingaben einen Wert zwischen 0 und 1, welcher dem gewichteten Anteil der beiden Eingaben gemeinsamen Merkmale entspricht. Die Gewichtung findet anhand der Merkmalsrelevanz, die im vorherigen Abschnitt behandelt wurde, statt.

Wichtig hierbei ist, daß bei der Schätzung der Merkmalsrelevanz, die für das Ähnlichkeitsmaß von hoher Bedeutung ist, davon ausgegangen wurde, daß die Klassenzugehörigkeit bekannt ist, d.h. daß es dem System einfach ist, zu bestimmen, ob eine Handlung beispielsweise vom Typ „Tisch decken“ oder von der Art „Spülmaschine ausräumen“ ist. Das ist sicherlich nur in einem sehr eingeschränkten Operationsmodus eines Systems zum inkrementellen und autonomen Lernen von Handlungswissen der Fall, wenn nämlich der Mensch, beispielsweise durch verbale oder tastaturbasierte Interaktion diese Information zusätzlich zu jeder Demonstration zur Verfügung stellt. Die in

der Einleitung zu dieser Arbeit vorgestellte Vision eines autonom lernenden Systems, welches sich das Handlungswissen des Menschen, auch wenn dieser sich unbeobachtet fühlt, abschaut, läßt eine solche Annahme jedoch nicht zu. Ein solches System muß in der Lage sein, autonom Klassen oder prototypische Handlungen zu erkennen, Neuigkeiten[3] wahrzunehmen, und diese in sein Handlungswissen nahtlos zu integrieren.

Dazu ist es jedoch notwendig, die Gesamtstruktur des Handlungswissens, die Gedächtnisstruktur eines solchen Systems, genauer unter die Lupe zu nehmen. Dies geschieht im folgenden Abschnitt.

5.3 Einspeisung in die Gedächtnisstruktur

Die Gedächtnisstruktur eines inkrementell lernenden Systems für Haushaltsroboter sollte es ermöglichen, neues Wissen, das dem System verfügbar gemacht wird, auf einfache und dennoch effektive Art und Weise in seinen Erfahrungsschatz aufzunehmen und das vorhandene Wissen integriert zur Verfügung zu stellen. Dabei sollte es autonom Analogien und Ähnlichkeiten erkennen und im Sinne einer Fusion ausnutzen.

Menschen weisen diese Fähigkeit durchaus auf. Mit einem bestimmten Problem konfrontiert, greifen sie auf erinnerte änliche Situationen zurück, welche der Aktuellen ähnlich sind, und die ihnen eventuell helfen, das aktuell bestehende Problem zu lösen. Diese Problemlösung bedeutet mehr als eine einfache Kopie der Lösung aus früheren Situationen, stattdessen ist das Problemlösungswissen aus der Erinnerung an neue Anforderungen und Randbedingungen anzupassen. Ein Beispiel für diese Art der Problemlösung findet bei der Verwendung juristischer Präzedenzfälle statt, wo aus vergangenen Entscheidungen Fingerzeige zur Interpretation neuer Fälle abgeleitet werden können, um Entscheidungsprozesse abzukürzen und ökonomischer gestalten zu können.

Betrachtet man Menschen genauer beim Lösen von Problemen, so ist es sehr wahrscheinlich, daß sie eine Strategie anwenden, welche [Kolodner 1993] als „Fallbasiertes Schließen“[4] bezeichnet[Ross 1989]. Dieser Prozess zeichnet sich vor allem durch den Rückgriff auf drei unterschiedliche Komponenten aus:

1. *Referenzen* auf andere Fälle bzw. ähnliche Wissensinhalte sind vorteilhaft. Fälle werden für spätere Problemlösungen erinnert und in das bereits bekannte Wissen integriert.

[3] engl. novelties

[4] engl.: Case-Based Reasoning

2. *Reasoning/Schlußfolgern* kompensiert Differenzen zwischen vergangenen und der aktuellen Situation.

3. *Lernen* findet statt als Resultat von Reasoning. Es hält die Ergebnisse des Reasoning-Prozesses für zukünftige Handlungs- und Entschiedungssituationen vor.

Fallbasiertes Schließen ist jedoch nicht nur ein psychologisches Konzept beim Verständnis menschlicher Erinnerungs- und Entscheidungsprozesse, sondern auch eine Entwurfsmethodik für wissensbasierte Expertensysteme. Sie wird oft angewandt, da sie relativ einfach und natürlich ist, ohne die Implementierungsvielfalt einzuschränken. Besondere Eignung weist sie für Probleme auf, die nur schwer regelbasiert faßbar und abstrakt beschreibbar sind. Gerade für die Implementierung menschlicher Eigenschaften, die diese nur implizit formulieren können, sind sie mit Erfolg eingesetzt worden. In manchen Domänen ist es nur schwer möglich, Experten ihr gesamtes Wissen mitteilen zu lassen, das sie benutzen, um Probleme zu lösen. Hingegen ist es oft ein deutlich einfacherer Ansatz, ihren Erfahrungsschatz durch die Schilderung charakteristischer Situationen und ihrer angewandten Lösungsansätze zu erfassen [Goodman 1989, Simmoudis 1992].

Ein System zum Fallbasiertem Schließen lernt primär durch die Akkumulation neuer Erfahrungen in die Gedächtnisstruktur und durch die Verbindungen, die zwischen den einzelnen Wissensteilen geknüpft werden. Lernen ist somit ein Wesensmerkmal eines solchen Systems, das aus dessen normaler Funktionsweise resultiert. Auf der anderen Seite steigt seine Lernfähigkeit mit der Mächtigkeit seiner schlußfolgernden Komponenten. Die Gesamtmächtigkeit eines solchen Systems ist also sowohl von den zur Verfügung stehenden Daten abhängig, als auch von den systeminhärenten Möglichkeiten, diese Daten zu abstrahieren, zu verallgemeinern und zu verbessern. Es lernt nicht dadurch, daß es immer wieder mit derselben Erfahrung konfrontiert wird, es lernt aber auch nicht allein durch eine große Menge an Daten, sondern dadurch, den vorhandenen Erfahrungsschatz systematisch optimal auszunutzen.

Fallbasiertes Schließen ist ein zyklischer Prozess, welcher durch das Finden ähnlicher Situation oder Fälle[5], der Wiederverwertung und der Verfeinerung[6] der schlußfolgernden Komponente sowie das Einfügen[7] des neuen Wissens in die gesamte Erfahrungswelt charakterisiert ist. Im Rahmen dieser Untersuchung wurde sowohl das Wiederfinden ähnlicher Fälle als

[5] engl. Retrieve
[6] engl. Reuse bzw. Refine
[7] engl. Retain

auch die Einspeisung neuer Erfahrung („Case-Based“) durch eine Clustering-Funkionalität realisiert, welche im folgenden Abschnitt behandelt werden soll. Der Wiederverwendungs- und Verfeinerungsschritt („Reasoning“) ist in Kapitel 6 detaillierter dargestellt.

5.4 Unüberwachter Aufbau einer Handlungstaxonomie

In vorherigen Abschnitt wurde der generelle Aufbau eines Fallbasierten Systems vorgestellt, welcher es erlaubt, Handlungswissen in disjunkte Handlungsklassen zu partitionieren, diese inkrementell zu erweitern und neue Klassen zu integrieren. In diesem Abschnitt soll jetzt die konkrete Implementierung, wie sie im Rahmen dieser Arbeit vorgenommen wurde, detailliert beschrieben werden.

Gemäß den Vorüberlegungen aus Abschnitt 5.3, geht es nun darum, Struktur innerhalb einer Menge von Datenmustern $\mathbf{D} = (D_1, D_2, \ldots, D_N)$ zu finden. Diese Struktur soll weiterhin in einem lesbaren und sowohl für das System, als auch für den Benutzer einfach verständlichen Form repräsentiert werden. Generell wird eine Repräsentation durch eine ausreichend kleiner Menge an Prototypen als eine solche Form angesehen (siehe [Pedrycz 2005]). Diese Prototypen werden als typische Elemente der Datenmenge ausgewählt. Sie sollen in einer Art und Weise gewählt werden, daß sie zum einen die Daten im höchstmöglichen Maße repräsentieren, und zum anderen voneinander unterschiedlich, d.h. disjunkt sind. Diese beiden Anforderungen werden im folgenden in einer Zielfunktion formuliert, die anschließend verwendet wird, um den Clustering- bzw. Prototyp-Auswahl-Prozess zu steuern.

Ein wesentlicher Teil dieser Zielfunktion ist ein Performanzmaß, welches angibt, wie gut ein Datenmuster als Prototyp die Daten beschreibt. Dieses iterative Maß ist folgendermaßen durch vollständige Induktion definiert:

- Ein Prototyp für den ersten Cluster v_1 wird so ausgewählt, daß er die Summe der Ähnlichkeitsmaße zu allen Elementen der Datenmenge maximiert, d.h.

$$(v_1, w_1) = \arg \max_{v \in \mathbf{D}, w \in [0;1]^n} \sum_{k=1}^{N} sim_w(x_k, v) \tag{5.4}$$

  mit $sim_w(x, y)$ gemäß Gleichung 5.2. Dieser Prototyp und sein dazugehöriger Gewichtungssvektor kann durch direkte Suche im Datenbestand und anschließende Optimierung des Gewichtungssvektors gefunden werden.

- Seien L Prototypen $v_1, \ldots v_L$ und ihre dazugehörigen Gewichte $w_1, \ldots, w_L$ in vorherigen Schritten gefunden worden. Im darauf folgenden Schritt für das $L+1$-te Paar von Prototyp und dazugehörigem Gewichtsvektor wird dieselbe Performanzfunktion angewandt, jedoch mit der zusätzlichen Einschränkung, daß v_{L+1} die bisherigen Prototypen nicht „duplizieren“ sollte, also nicht zu nahe an ihnen liegt, also keine neuen Daten repräsentiert. Um genau das zu vermeiden, werden für jeden bisherigen Prototypen i Faktoren der Form

  $$1 - sim_{w_i}(v, v_i)$$

  eingeführt, die genau diese Bedingung der maximalen Entfernung von v_i bewirken. Zusammenfassend wird der $L+1$-Prototyp gefunden durch Maximierung von

  $$(v_{L+1}, w_{L+1}) = \arg \max_{v \in \mathbf{D}, w \in [0;1]^n} (1 - sim_1(v_L, v)) \ldots (1 - sim_1(v_1, v)) \cdot \sum_{k=1}^{N} sim_w(x_k, v) \tag{5.5}$$

  Diese Vorgehensweise berücksichtigt alle bisher ausgewählten Prototypen. Interessanterweise läßt sich beweisen [Pedrycz 2005], daß das Performanzmaß eine monoton fallende Funktion von der Anzahl der bereits berücksichtigten Prototypen ist, d.h. für steigende L fällt das Performanzmaß monoton. Somit kann man ein einfaches Abbruchkriterium für diesen Prozeß durch Angabe eines Schwellwertes θ_P formulieren, wonach der Auswahlprozeß nur solange fortgesetzt wird, solange der Performanzindex des Prototypes sich oberhalb dieser Schwelle bewegt.

Bisher wurde die Optimierung des Gewichtungsvektors nicht betrachtet. Sie stellt jedoch einen integralen Bestandteil des gesamten Clustering-Verfahrens dar. Dem soll im Rest diesen Abschnittes Rechnung getragen werden.

Gemäß Gleichung 5.5, kann die Suche nach einem Gewichtsvektor w_{L+1} für den Prototyp v_{L+1} beschrieben werden durch

$$(v_{L+1}, w_{L+1}) = \arg \max_{v \in \mathbf{D}, w \in [0;1]^n} G \cdot \sum_{k=1}^{N} sim_w(x_k, v),$$

wobei der Term G nicht von w_{L+1} abhängt, und somit für die Optimierung von w_{L+1} als konstant angenommen werden kann. Führt man als Nebenbedingung ein, daß sich die Gewichte w_{L+1} zu eins addieren sollen, also

$$\sum_{j=1}^{n} w_j = 1, \tag{5.6}$$

so ergibt sich ein lineares Optimierungsproblem mit Nebenbedingungen.

Die detaillierte Ableitung des Gewichtungsvektors kann nun durch Anwendung der Lagrange-schen Multiplikatoren angegangen werden. Zuerst wird die erweiterte Form des Performanz-Index erzeugt:

$$V = G \cdot \sum_{k=1}^{N} \left\{ \prod_{j=1}^{n} (w_j^2 + s(x_{kj}, v_j)^2 - w_j^2 s(x_{kj}, v_j)^2) \right\} - \lambda \left(\sum_{j=1}^{n} w_j - 1 \right)$$

Durch Nullsetzen der Ableitungen von V nach den einzelnen Elementen des Gewichtsvektors sowie des Lagrange-schen Multiplikators

$$\frac{dV}{dw_s} \text{ und } \frac{dV}{d\lambda}$$

ergibt sich

$$\frac{dV}{dw_s} = G \sum_{k=1}^{N} \frac{d}{dw_s} \left\{ A_{ks} \cdot (w_s^2 + s(x_{ks}, v_s)^2 - w_s^2 s(x_{ks}, v_s)^2) \right\} - \lambda = 0 \quad (5.7)$$

wobei

$$A_{ks} = \prod_{j=1, j \neq s}^{n} (w_j^2 + s(x_{kj}, v_j)^2 - w_j^2 s(x_{kj}, v_j)^2),$$

definiert ist.

Weitere Ableitung des inneren Terms liefert

$$\frac{d}{dw_s} \left\{ A_{ks} \cdot (w_s^2 + s(x_{ks}, v_s)^2 - w_s^2 s(x_{ks}, v_s)^2) \right\} = 2 A_{ks} w_s (1 - s(x_{ks}, v_s)^2).$$

Setzt man dies in Gleichung (5.7) ein, so erhält man

$$\frac{dV}{dw_s} = 2 G w_s \cdot \sum_{k=1}^{N} A_{ks} (1 - s(x_{ks}, v_s)^2) - \lambda = 0.$$

Weitere Umformungen ergeben daraus

$$w_s = \frac{\lambda}{2G \cdot \sum_{k=1}^{N} A_{ks} (1 - s(x_{ks}, v_s)^2)}. \quad (5.8)$$

Einsetzen in die Nebenbedingung (5.6) liefert

$$\frac{\lambda}{2} \sum_{j=1}^{n} \frac{1}{G \cdot \sum_{k=1}^{N} A_{kj} (1 - s(x_{kj}, v_j)^2)} = 1$$

oder

$$\frac{\lambda}{2} = \frac{1}{\sum_{j=1}^{n} \frac{1}{G \cdot \sum_{k=1}^{N} A_{kj}(1 - s(x_{kj}, v_j)^2)}}. \tag{5.9}$$

Setzt man nun den Wert für λ aus (5.9) in (5.8) ein, so ergibt sich

$$w_s = \frac{1}{\sum_{j=1}^{n} \frac{\sum_{k=1}^{N} A_{ks}(1 - s(x_{ks}, v_s)^2)}{\sum_{k=1}^{N} A_{kj}(1 - s(x_{kj}, v_j)^2)}}. \tag{5.10}$$

Mit diesem Ergebnis läßt sich der Algorithmus zum iterativen Finden von Prototypen für Taskklassen folgendermaßen zusammenfassen (siehe [Pardowitz et al. 2006c]): In einem ersten Schritt werden alle Demonstrationen als potentielle Prototypen ausgewertet. Dazu werden für jede Auswahl der optimale Gewichtsvektor gemäß Gleichung (5.10) berechnet. Anschließend wird das Paar ausgewählt, welches von allen N verfügbaren Optionen den maximalen Wert für $Q(L)$ liefert. Das dazugehörige Datum wird als neuer Prototyp etabliert, zusammen mit seinem optimalen Gewichtsvektor w_L.

Jeder einzelne Prototyp hat nun seinen eigenen Gewichtsvektor, der durchaus von den anderen Gewichtsvektoren verschieden sein kann. Die Interpretation für diese unterschiedliche Gewichtsvektoren lautet, daß diese die „lokalen“ Charakteristiken des Merkmalsraumes darstellen. Dieser Umstand ist schematisch dargestellt in Abbildung 5.2. Wie man hier sehen kann, sind unterschiedliche Merkmale der Aufgabenrepräsentation für unterschiedliche Aufgabenklassen von unterschiedlicher Relevanz, wie die Ausdehnung der Aufgabenklassen in den verschiedenen Dimensionen zeigt. Beispielsweise ist für die Aufgabenklasse, welche zu dem Cluster in der unteren linken Ecke gehört, die y-Komponente von höherer Relevanz bzw. von niedrigerer Spreizung als die Attributwerte für die x-Komponente. Dieser Umstand ist für den Cluster in der oberen rechten Ecke umgekehrt. Die Gewichtsvektoren w_i, w_j, welche mit diesen beiden Aufgabenklassen verbunden sind, reflektieren diese Tatsache durch unterschiedliche Werte für die einzelnen Komponenten. Der Merkmalsraum kann somit als anisotroper Raum aufgefaßt werden, welcher durch die Gewichtung mit den Merkmalsvektoren entzerrt bzw. normiert wird (siehe Abbildung 5.2 rechts).

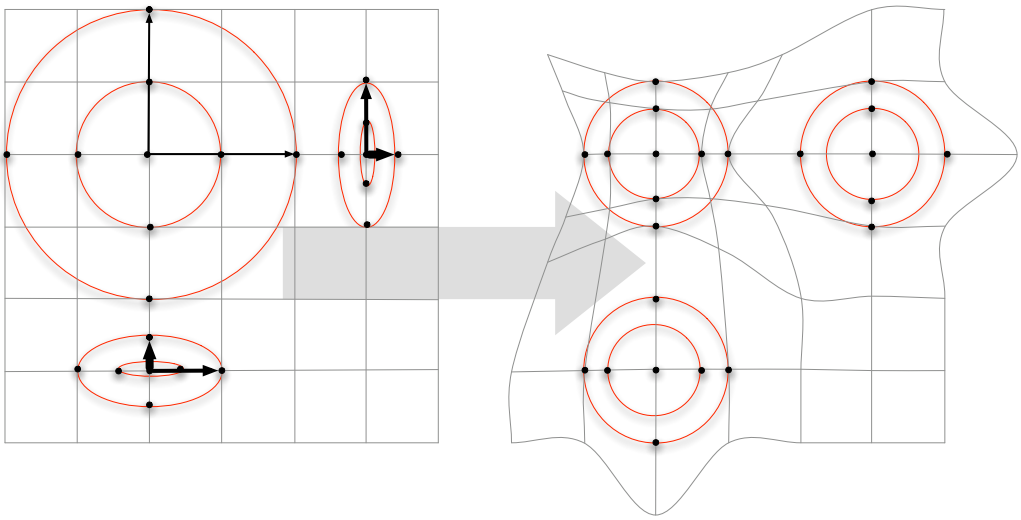


Abbildung 5.2: Anisotropie des Merkmalsraumes der Benutzerdemonstrationen, repräsentiert durch Gewichtsvektoren zu den Prototypen der Taskklassen.

5.5 Zusammenfassung

In diesem Kapitel wurden die mathematischen Grundlagen und Verfahren vorgestellt, welche zur semantischen Interpretation von Benutzerdemonstrationen eingesetzt werden. Semantik wird hierbei nicht als a-priori gegebene Größe aufgefaßt, sondern muß von dem System durch schrittweise Integration neuer Sensordaten in seinen gesamten Erfahrungsschatz aktiv und inkrementell geleistet werden.

Dazu ist es zuerst nötig, die Ähnlichkeit neuer Erfahrungen mit bisher schon in den Erfahrungsschatz integrierten Wissensteilen zu bewerten. Hierzu wurden Ähnlichkeitsmetriken definiert, welche unter Rückgriff auf Schätzungen der Relevanz einzelner Merkmale eine Bewertung und Vernetzung unterschiedlicher Wissensteile ermöglichen.

Der gesamte, fallbasierte Aufbau der Gedächtnisstrukturen des lernenden Systems ist vorgestellt worden, sowie der Algorithmus zur autonomen Bestimmung von Kategorien in der Wissensbasis über inkrementelle Auswahl von Prototypen und Optimierung deren lokaler Relevanzmerkmale.

In diesem Abschnitt wurde die Grundlage zur Verknüpfung von neuem mit bereits vorhandenem Wissen gelegt. Eine reine Etablierung von Referenzen zwischen den einzelnen Benutzerdemonstrationen reicht jedoch nicht aus, um einen den Aufwand rechtfertigenden Vorteil zu erlangen. Deshalb müssen die gefundenen Korrespondenzen im Sinne weitergehender Abstraktion und

Verallgemeinerung ausgenutzt werden. Die hierzu dienenden Verfahren zum automatisierten Schlussfolgern und Reasoning werden im folgenden Kapitel vorgestellt.

Kapitel 6

Inkrementelles Lernen und Reasoning

Das vergangene Kapitel 5 hat beschrieben, wie Benutzerdemonstrationen einer Handlung hinsichtlich ihrer Semantik analysiert und erfaßt werden können, wie Verknüpfungen zwischen der neuen Demonstration und dem bisher vorhandenem Erfahrungsschatz aufgebaut werden, sowie wie das neue Handlungswissen in die gesamte Handlungsdatenbank eingeordnet werden kann. Es ist somit dem System möglich, neue Eindrücke in einen Wissenskontext einzuordnen und Handlungsklassen autonom zu erkennen, also verschiedene Handlungsschwerpunkte oder disjunkte zu erledigende Aufgaben aus dem Erfahrungsschatz zu extrahieren.

Obwohl dieses Einordnen für sich genommen schon eine erstaunliche kognitive Leistung ist, wird der Nutzen für den Anwender noch verstärkt, wenn das neu erlernte Handlungswissen nicht nur durch einfache Einordnung in den Erfahrungsschatz abgehandelt wird. Vielmehr ist es von großer Bedeutung, zu dem Gesamtwissen, das zu einer Handlungsklasse gehört, durch Verschmelzen der Information, die mit jeder einzelnen Demonstration dieser Klasse verbunden ist, zusätzliches Handlungswissen hinzuzufügen. Dieses kann in abstrakteren, allgemeineren oder treffenderen Repräsentation für die Aufgabenklasse resultieren. Wichtig ist dabei der konstruktive Aspekt, wie er auch in psychologischen Untersuchungen bei Menschen festgestellt wurde (siehe [Bandura 1969]). Das bedeutet, daß sich das System seine Modelle für das Handlungswissen schrittweise selbst aufbaut und verändert.

Im Gegensatz zu den Verfahren aus dem vorherigen Kapitel ist bei den Verfahren, welche hier beschrieben werden sollen, die Klassenzugehörigkeit bekannt. Diese Tatsache bewirkt, daß die Verfahren zur Fusion von Intra-Klassen-Wissen meist mittels Methoden des *überwachten* Lernens und Schlußfolgerns erfolgen kann, welche sich von den unüberwachten Lernver-

fahren durch die unterschiedliche Aufgabenstellung unterscheiden.

Ein weiteres wesentliches Merkmal der Verfahren und Algorithmen in diesem Kapitel ist, daß sie allesamt zur Unterklasse der *inkrementellen* Lernverfahren zählen. Das bedeutet, daß sie nicht über sämtliche Lerndaten verfügen müssen, um mit dem Lernprozeß beginnen zu können, sondern iterativ ihr Wissen erweitern, ergänzen oder umstrukturieren, sobald neue Informationen verfügbar sind. Dies entspricht einer Auffassung, daß das erhaltene Wissen stets als vorläufig betrachtet werden muß und für weitere Adaptionen zur Verfügung stehen muß. Dies entspricht dem menschlichen Erfahrungs-basierten lebenslangen Lernen, welches einen sehr wesentlichen Beitrag zum Phänomen der Intelligenz ausmacht.

In diesem Kapitel sollen nun die verschiedenen Techniken, Methoden und Verfahren zum inkrementellen Lernen und Schlußfolgern bzw. Reflektieren über Handlungsaufgaben vorgestellt werden. Dabei werden zuerst Verfahren behandelt, die es erlauben Operatorpermutationen und Umordnungen zu erkennen (Abschnitt 6.1). Im Anschluß werden diese Permutationen genutzt, um die Sequentialitätsbedingungen, die für eine Aufgabe gelten, zu extrahieren (Abschnitt 6.2). Anschließend wird ein theoretisches Rahmenwerk zum parallelen Verfolgen multipler Hypothesen vorgestellt, welches auf die algebraische Verknüpfung atomarer Hypothesenräume basiert, die sogenannte Versionsraumalgebra (siehe Abschnitt 6.3). Abschließend wird dieses Rahmenwerk instanziiert mit dem Ziel, Schleifenkonstrukte in repetitiven Aufgabenstellungen zu Erlernen (siehe Abschnitt 6.4).

6.1 Operatorpermutationen

Zum Lernen von Umordnungsmöglichkeiten, die sich einem Robotersystem beim Ausführen einer Aufgabe ergeben, werden mit den Verfahren aus Abschnitt 6.2 Aufgabenpräzedenzgraphen aus verschiedenen Demonstrationen gelernt. Die verschiedenen Demonstrationen bestehen aus unterschiedlichen Sequenzen derselben Operationen, d.h. sie umfassen dieselben Teilhandlungen, diese sind jedoch in verschiedenen Reihenfolgen angeordnet. In diesem Abschnitt werden Methoden vorgestellt, um die Unterschiede in den verwendeten Sequenzen zu repräsentieren, sowie um diese Beschreibungen aus unterschiedlichen Demonstrationen zu extrahieren.

Dazu wird zuallererst der Begriff der Operatorpermutation definiert. Eine solche Operatorpermutation ist eine Funktion, welche eine bijektive Abbildung zwischen zwei Operatorlisten herstellt.

Definition 14 (Operatorpermutation) *Eine Operatorpermutation* $\mathbf{P}_{\mathbf{L}_1,\mathbf{L}_2} \subset \mathbf{L}_1 \times \mathbf{L}_2$ *zwischen zwei geordneten Listen identischer Länge von*

Operatoren $\mathbf{L}_1 = (o_1, o_2, \ldots, o_n), \mathbf{L}_2 = (o'_1, o'_2, \ldots, o'_n)$ *ist eine Menge von n Operatortupeln aus* $\mathbf{L}_1 \times \mathbf{L}_2$*, wobei jeder Operator in genau einem dieser Tupel vorkommt. Formal gilt also*

1. $\forall x \in \mathbf{L}_1 \exists y \in \mathbf{P}_{\mathbf{L}_1,\mathbf{L}_2} : y = (x, z), z \in \mathbf{L}_2$
2. $\forall x \in \mathbf{L}_2 \exists y \in \mathbf{P}_{\mathbf{L}_1,\mathbf{L}_2} : y = (z, x), z \in \mathbf{L}_1$
3. $\forall (x_1, y_1), (x_2, y_2) \in \mathbf{P}_{\mathbf{L}_1,\mathbf{L}_2} : x_1 = x_2 \Rightarrow y_1 = y_2 \wedge y_1 = y_2 \Rightarrow x_1 = x_2$

Die zugehöriger bijektive Permutationsabbildung $\mathbf{P}_{\mathbf{L}_1,\mathbf{L}_2}(o) : \mathbf{L}_1 \mapsto \mathbf{L}_2$ *und die dazugehörige Inverse* $\mathbf{P}^{-1}_{\mathbf{L}_1,\mathbf{L}_2}(o) : \mathbf{L}_2 \mapsto \mathbf{L}_1$ *können folgendermaßen konstruiert werden:*

$$\mathbf{P}_{\mathbf{L}_1,\mathbf{L}_2}(o) = \begin{cases} p, & \textit{falls } (o,p) \in \mathbf{P}_{\mathbf{L}_1,\mathbf{L}_2} \\ \text{Fehler}, & \textit{sonst.} \end{cases}$$

$$\mathbf{P}^{-1}_{\mathbf{L}_1,\mathbf{L}_2}(o) = \begin{cases} p, & \textit{falls } (p,o) \in \mathbf{P}_{\mathbf{L}_1,\mathbf{L}_2} \\ \text{Fehler}, & \textit{sonst.} \end{cases}$$

In dieser Arbeit werden hauptsächlich die Listen der Teilaufgaben von Makrooperatoren verwendet. Seien nun zwei Makro-Operatoren D_1, D_2 gemäß Definition 2 (siehe Seite 63) gegeben, mit ihren jeweils zugehörigen geordneten Listen von Nachfolgern $\mathbf{S}_1, \mathbf{S}_2$, so wird folgende verkürzende Schreibweise verwendet:

$$\mathbf{P}_{D_1,D_2} := \mathbf{P}_{\mathbf{S}_1,\mathbf{S}_2}$$

Für höherwertige Lernverfahren sind Permutationen interessant, die zusätzliche Eigenschaften aufweisen. Zum Lernen von Reihenfolgebedingungen ist es von Bedeutung, diejenige Permutation zu kennen, welche jeden Nachfolger in $\mathbf{S}_1$ dem äquivalenten Operator in $\mathbf{S}_2$ zuordnet, so daß also gilt:

$$(x, y) \in \mathbf{P}_{D_1,D_2} \Leftrightarrow x \equiv y$$

gemäß einem Äquivalezmaß $\equiv$.

Leider ist es nur schwer möglich, ein solches Äquivalenzmaß explizit zu definieren und anzuwenden. Dies liegt an dem stets vorhandenen Rauschen, welches sämtlichen von Sensordaten abgeleiteten Informationen anhaftet. Deshalb wurde in dieser Arbeit ein implizites Äquivalenzmaß verwendet, welches die Ähnlichkeiten zwischen den permutierten Operatoren maximiert. Diese Wahl ist motiviert durch die Tatsache, daß mit hoher Wahrscheinlichkeit zwei Operatoren äquivalent sind, wenn sie eine große Ähnlichkeit aufweisen. Formal definiert wird dies durch die Maximierung der Subaufgaben-Korrespondenz

$$sk(\mathbf{P}_{D_1,D_2}) = \sum_{(x,y) \in \mathbf{P}_{D_1,D_2}} sim_w(x, y) \tag{6.1}$$

unter Verwendung des Ähnlichkeitsmaßes gemäß Gleichung 5.2 (siehe Abschnitt 5.2) sowie des klassenspezifischen Gewichtungsvektors w.

Eine beispielhafte Verteilung von Ähnlichkeiten für zwei Operatorlisten (x_1, x_2, x_3) und (y_1, y_2, y_3) ist in Abbildung 6.1 dargestellt. Hier besteht eine hohe Ähnlichkeit zwischen den Operatoren x_1 und y_2, x_2 und y_3, x_3 und y_1 sowie x_3 und y_3. Für x_3 existieren also lokal gesehen zwei aussichtsreiche Kandidaten, nämlich y_1 und y_3 für die Zuordnung, unter dem Gesichtspunkt der globalen Optimierung der Permutation gemäß Gleichung 6.1 kommt jedoch nur y_1 in Frage, da eine Wahl von y_3 in einen insgesamt niedrigeren Wert für $sk(\mathbf{P}_{D_1,D_2})$ resultieren würde. Somit wäre die optimale Permutation

$$\mathbf{P}_{D_1,D_2} = \{(x_1, y_2), (x_2, y_3), (x_3, y_1)\}.$$

Nach der Spezifikation der optimale Permutation stellt sich die Frage, wie diese berechnet wird. Für Probleme des Umfanges des oben angegebenen Beispiels ist eine „Brute-Force"-Suche durch Ausprobieren sämtlicher möglicher Permutationen noch möglich. Es ist jedoch zu beachten, daß die Anzahl der möglichen Permutationen fakultativ von der Anzahl der zu permutierenden Elemente abhängt, d.h. es gibt $n!$ verschiedene Permutationen. Dieser Wert steigt exponentiell mit n an und ist für höhere n nicht mehr rechnerisch handhabbar. Es sind also effizientere Methoden zur Bestimmung der optimalen Permutation zu entwerfen und zu implementieren. Das oben dargestellte Problem des Findens einer optimalen Permutation ist ein Spezialfall eines komplizierten grundlegenden Problems der Graphenalgorithmen. Für einen gegebenen Graphen ist eine *Paarung* zu finden, nämlich eine Kantenmenge zu finden, in der kein Knoten mehr als einmal erscheint (siehe [Sedgewick 1992], Kapitel 34).

Das Problem besteht nun darin, einen effizienten Algorithmus zu entwerfen und zu implementieren, welcher eine Menge an Kanten im Graphen auswählt, wovon jede die Eigenschaft hat, daß die verbundenen Knoten in keinem anderen Paar vorkommen, und weiterhin, daß die Summe der Gewichte dieser Kantenmenge maximal ist. So wird jeder Knoten, der von einer der Kanten der Paarung berührt wird, mit dem anderen Knoten dieser Kante zu einem Paar vereinigt. Die Graphen, die hier betrachtet werden, haben weiterhin die Eigenschaft, daß sie *bipartit* sind, das heißt, daß alle Kanten zwischen zwei disjunkten Teilmengen von Knoten verlaufen. Die Knoten lassen sich also in zwei Teilmengen einteilen, und keine Kante verbindet zwei Knoten aus derselben Teilmenge.

Allgemein können Graphen durch Adjazenzmatrizen dargestellt werden. Hat der Graph $G = (\mathbf{N}, \mathbf{E})$ die Knoten- und Kantenmenge $\mathbf{N} = \{x_1, x_2, \ldots, x_n\} \cup \{y_1, y_2, \ldots, y_n\}$ bzw. $\mathbf{E}$ und ist die Kantengewichtung durch

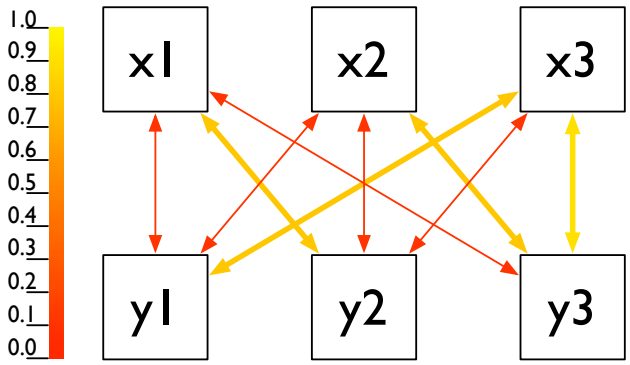


Abbildung 6.1: Bestimmung der optimalen Operatorpermutation unter Berücksichtigung des Similaritätsmaßes sim_w. Hohe Ähnlichkeiten korrespondieren mit gelben dick markierten Verbindungslinien, niedrige mit roten dünnen.

eine Funktion $w : \mathbf{E} \mapsto [0; 1]$ festgelegt, so ist die Adjazenzmatrix M gegeben durch

$$M_{i,j} = sim_w(x_i, y_j).$$

Eine solche Adjazenzmatrix für den in Abbildung 6.1 dargestellten Ähnlichkeitsgraphen ist

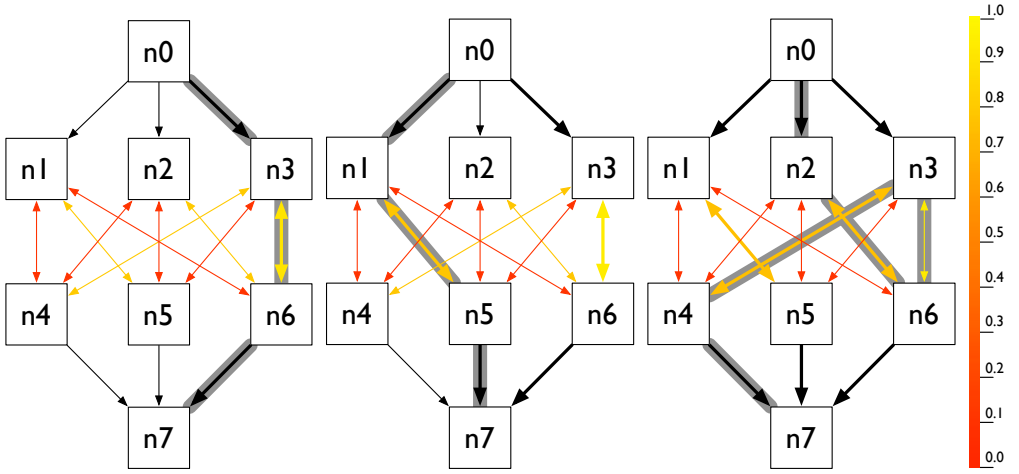
$$M_{i,j} = sim_w(x_i, y_j) = \begin{bmatrix} 0.1 & 0.8 & 0.1 \\ 0.1 & 0.1 & 0.8 \\ 0.8 & 0.1 & 0.9 \end{bmatrix}. \tag{6.2}$$

Unter Bezug auf [Sedgewick 1992] wird hier ein Ansatz verwendet, welcher zur Lösung des Matching-Problems in bipartiten Graphen dieses zurückführt auf das Problem des Flusses in einem Netzwerk. Dazu wird zunächst ein neuer Knoten, genannt Quelle, erzeugt. Von diesem werden Kanten zu allen Knoten der einen Teilmenge von $\mathbf{N}$ erzeugt. Alle Kanten von Knoten in dieser Menge führen zu Knoten in der anderen Teilmenge. Jene erhalten zusätzlich eine weitere Kante, welche zu einem ebenfalls neu erstellten Senken-Knoten führen. Zu einem bipartiten Graphen $(\mathbf{N}_1 \cup \mathbf{N}_2, \mathbf{E})$ mit der Adjazenzmatrix M läßt sich so der aktualisierte Flußgraph $(\mathbf{N}', \mathbf{E}')$ konstruieren, welcher durch

$$\mathbf{N}' = \mathbf{N}_1 \cup \mathbf{N}_2 \cup \{Q, S\} \text{ und } \mathbf{E}' = \mathbf{E} \cup \{(Q, n_i) | n_i \in \mathbf{N}_1\} \cup \{(n_i, S) | n_i \in \mathbf{N}_2\}$$

sowie die Adjazenzmatrix

$$M'_{i,j} = \begin{cases} 1.0 & \text{falls } i = 0 \wedge 0 < j \leq |\mathbf{N}_1|, \\ 1.0 & \text{falls } i = |\mathbf{N}_1| + |\mathbf{N}_2| + 1 \wedge |\mathbf{N}_1| < j \leq |\mathbf{N}_1| + |\mathbf{N}_2|, \\ M_{i-1,j-1-|\mathbf{N}_1|} & \text{falls } 0 < i \leq |\mathbf{N}_1| \wedge |\mathbf{N}_1| < j \leq |\mathbf{N}_1| + |\mathbf{N}_2|, \\ 0.0 & \text{sonst} \end{cases}$$



(a) (b) (c)

Abbildung 6.2: Berechnung der optimalen Operatorpermutation durch Maximierung des Flusses durch ein gewichtetes Netzwerk. Zwischenergebnisse des Algorithmus nach dem (a) ersten, (b) zweiten und (c) dritten Schritt. Man sieht, wie in Schritt 3 die anfängliche Fehlannahme der Äquivalenz von $n3$ und $n6$ korrigiert wird.

definiert wird. Der bipartite Graph aus Abbildung 6.1 wird somit umgeformt zu dem, welcher in Abbildung 6.2 zu sehen ist.

Der Algorithmus zur Bestimmung des maximalen Flusses (siehe [Floyd 1962, Warshall 1962]) in einem Graphen geht nun folgendermaßen vor: Er sucht schrittweise den Pfad von der Quelle bis zur Senke des Graphen, welcher die meisten ungenutzten Kapazitäten besitzt. Hierbei zählen rückwärtsgerichtete Kanten negativ, da sie der Umleitung des Flusses durch einen anderen Knoten entsprechen. Dies ist in Abbildung 6.2 dargestellt am Beispiel der Adjazenzmatrix aus Gleichung 6.2. Im ersten Schritt findet der Algorithmus den Pfad mit dem maximalen Fluss von der Quelle n_0 zur Senke n_7, nämlich den Pfad über n_3 und n_6 (grau markiert). Es wird also der Fluß durch diesen Pfad in der Größenordnung von 0.9 angenommen (dick schwarz). Der nächstbeste Pfad von der Quelle zur Senke geht über n_1 und n_5. Auch hier wird der Fluss angenommen. Nun ist von n_2 aus in einem einfachen Pfad nur noch n_4 mit dem Fluß 0.1 frei erreichbar, allerdings der „Umweg“ über n_6, n_3 und n_4 ermittelt ein deutlich besseres Ergebnis von 0.8. So wird die Verbindung von n_3 nach n_6 mit der Gewichtung von 0.9 zugunsten der lokal minimal schlechteren Kante von n_3 nach n_4 aufgegeben, was jedoch dazu führt, daß mit der Kante (n_2, n_6) noch eine gewichtige Kante aufgenommen werden kann. So ist der maximale Fluß von $2,4$ erreicht

worden, was der maximalen Permutation entspricht:

$$\mathbf{P}_{D_1,D_2} = \{(x_1, y_2), (x_2, y_3), (x_3, y_1)\}.$$

6.2 Lernen sequentieller Unabhängigkeiten

Betrachtet ein System eine einzige Vorführung einer Aufgabenlösung, so stellen sich ihm die zur Lösung der Aufgabe durchgeführten Aktionen als eine sequentielle Folge von Operationen dar. Die Reihenfolge, in denen die Operationen, aus denen sich die Demonstration zusammensetzt, ausgeführt werden und die Teilziele einer Vorführung erreicht werden, sind durch zwei Faktoren bestimmt:

- Durch die Eigenschaften der zu lösenden Aufgabe induzierte *sequentielle Abhängigkeiten*. Diese stellen zeitliche Präzedenzbeziehungen dar, die sich aus der spezifischen Aufgabe ergeben. So ist es beispielsweise beim Tischdecken notwendig, dass die Platzierung einer Untertasse erfolgt, bevor eine Tasse daraufgestellt wird[1].

- *Sequentialisierung* von temporal nicht abhängigen Teilaufgaben. Auch wenn die zu erledigende Aufgabe keine Reihenfolge für die Operationen vorgibt, muss sich der Benutzer zum Vorführungszeitpunkt entscheiden, in welcher Abfolge er sie demonstriert. So ist es (um beim Tischdecken zu bleiben) egal, ob zuerst ein Teller oder das danebenliegende Besteck platziert wird. Der Benutzer muss jedoch entscheiden, welche der beiden Aktionen er zuerst durchführt. Dadurch entsteht eine (willkürliche) Sequentialisierung zweier sequentiell nicht abhängiger Operationen.

Für ein lernendes System ist es jedoch nicht ohne weiteres möglich, die intentionalen von den willkürlichen Vorrangbeziehungen einer einzigen Benutzervorführung zu unterscheiden. Die einzige gesicherte Hypothese, die das System anstellen kann, ist, alle zeitlichen Abfolgebeziehungen ernst zu nehmen bzw. alle Reihenfolgebeziehungen für sequentielle Abhängigkeiten zu halten. Diese pessimistische Annahme sichert zu, daß keine relevanten Sequentialitätseigenschaften bei der Ausführung verletzt werden. Dies könnte nämlich zu Beschädigungen von Objekten oder sogar am Manipulator selbst führen, beispielsweise, wenn bei dem Holen eines Objektes aus dem Kühlschrank nicht beachtet wird, daß die Operation des Öffnens der Kühlschranktür zeitlich vor dem Greifvorgang erfolgen muß, da andernfalls versucht wird,

[1] Unter der durchaus realistischen Annahme, dass die Untertasse mit der darauf abgestellten Tasse von einem Robotersystem nicht mehr kontrolliert manipulierbar ist

durch die geschlossene Tür hindurch zu greifen. Dementsprechend sollte beim Vorhandensein nur einer einziger Demonstration die Reihenfolge zur Ausführung gewählt werden, welche alle zeitlichen Abhängigkeiten ernst nimmt, welche in der Demonstration beachtet wurden.

Führt der Benutzer jedoch dieselbe Aufgabe noch einmal vor, so kann es sein, dass er eine andere Sequentialisierung der zeitlich unabhängigen Operationen wählt. Wird diese andere Demonstration dem System zur Verfügung gestellt, so kann die Hypothese verfeinert werden, indem einige der bisher als sequentielle Abhängigkeiten angenommenen Vorrangbeziehungen gestrichen bzw. als sequentialisiert erkannt werden.

Die sequentiellen Abhängigkeitsbeziehungen werden in einem sogenannten Präzedenzgraphen repräsentiert, der folgendermassen definiert ist:

Definition 15 (Präzedenzgraph) *Ein Präzedenzgraph ist ein gerichteter Graph* $G = (\mathbf{N}, \mathbf{E})$ *mit der Menge der in den Benutzervorführung auftretenden Operationen* $\mathbf{N}$ *und der Kantenmenge* $\mathbf{E}$*. Jeder Kante* $(o_1, o_2) \in E$ *entspricht eine sequentielle Abhängigkeit zwischen den Operationen* o_1 *und* o_2*, d.h.* o_1 *muss vor* o_2 *ausgeführt werden, bzw. mit der Ausführung von* o_2 *kann erst begonnen werden, wenn* o_1 *komplett und fehlerfrei ausgeführt wurde. Dies wird im folgenden auch durch die intuitive Schreibweise* $o_1 \rightarrow_G o_2$ *ausgedrückt.*

Wie oben beschrieben, erfordert eine korrekte Ausführung des Handlungswissens, welches in einem Präzedenzgraphen kodiert ist, daß jeder einzelnen Präzedenzbeziehungen Rechnung getragen wird. Dies wird im folgenden mit der Aussage bezeichnet, daß eine bestimmte Sequenz (beispielsweise die geplante Ausführungsfolge) mit dem Taskpräzedenzgraphen *konsistent* ist. Formal ausgedrückt gilt für eine mit dem Taskpräzedenzgraph $G = (\mathbf{N}, \mathbf{E})$ konsistente Sequenz $D = (o_{i_1}, o_{i_2}, ..., o_{i_n})$, daß für jede Präzedenzrelation $o_j \rightarrow_G o_k$ die Operationen $o_j = o_{i_l}$ und $o_k = o_{i_m}$ in der richtigen Reihenfolge in D auftreten, d.h. $l < m$. Die Tatsache, daß eine bestimmte Sequenz D mit dem Präzedenzgraphen P konsistent ist, wird im folgenden auch mit $P \mapsto D$ abgekürzt.

Für eine Menge von Vorführungen desselben Tasks $\mathbf{D} = \{D_1, D_2, ..., D_n\}$ ist nun ein Präzedenzgraph zu finden, mit dem alle Demonstrationen konsistent sind, d.h. eine Menge von Präzedenzbeziehungen, die von keiner der Demonstrationen verletzt wird. Dieser sollte die erhaltenen Demonstrationen nicht über-generalisieren (also zuwenige sequentielle Abhängigkeiten annehmen), jedoch die Präzedenzrelationen so generell wie möglich erfassen, d.h. sämtliche Unabhängigkeiten, die sich aus der Demonstrationsmenge ergeben, berücksichtigen.

Weiterhin soll das Verfahren, welches den Taskpräzedenzgraphen zu einer gegebenen Menge an Demonstrationen berechnet, wie alle Verfahren in diesem Abschnitt, dem Anspruch der Inkrementalität genügen, d.h. es sollte in der Lage sein, den gelernten Taskpräzedenzgraphen unter Betrachtung einer neuen Demonstration anzupassen und zu generalisieren, ohne daß die gesamte Menge der bisher schon erhaltenen Demonstrationen neu betrachtet werden muß. Dies entspricht einem induktiven Ansatz, wonach ein initialer Präzedenzgraph für die erste Demonstration ihrer Art, d.h. in einer neuen Taskklasse erstellt wird, der inkrementell mit jedem neuen Trainingsbeispiel verändert wird.

Für den ersten Schritt, also den Fall, daß lediglich eine einzige Demonstration vorliegt, ist die Berechnung des Taskpräzedenzgraphen relativ einfach. Wie oben motiviert, soll der Taskpräzedenzgraph gewählt werden, welcher keinerlei Freiheiten bezüglich der Umordnung von Operationen zuläßt, und somit die Erfüllung sämtlicher Sequentialitätsbeziehungen garantiert. Dies ist mit dem restriktivsten Präzedenzgraph $P^D = (\mathbf{N}, \mathbf{E}^D)$ zu einer Demonstration $D = (o_1, o_2, \ldots, o_n)$ gewährleistet, welcher aus der Knotenmenge, die sämtliche Operationen umfasst $\mathbf{N} = \{o_i \mid 1 \leq i \leq n\}$, und der Kantenmenge

$$\mathbf{E}^D = \{(o_i, o_j) \mid \forall i, j : i < j\} \tag{6.3}$$

besteht.

Mit jeder weiteren verfügbaren Demonstration soll das System diesen initialen Taskpräzedenzgraphen generalisieren. Im Extremfall, wenn überhaupt keine sequentiellen Abhängigkeitsbeziehungen zwischen den Operatoren einer Aufgabe bestehen, wenn also die Reihenfolge frei und unbeschränkt wählbar ist, wird der Taskpräzedenzgraph mit der leeren Relationenmenge am Ende dieses Prozesses stehen, also mit Resultat $(\mathbf{N}, \emptyset)$ zum Abschluß kommen.

Dieses Generalisieren findet statt im Raum sämtlicher Taskpräzedenzgraphen, mit denen die initiale Demonstration konsistent ist. Dieser Raum der mit den Trainingsdaten konsistenten Hypothesen ist unter Umständen (abhängig von der Anzahl der Basisoperationen) sehr groß. Deshalb ist es zur Anwendung von Lern- und Generalisierungsschritten nötig, ein Koordinatensystem bzw. eine Struktur in diesem Raum zu verankern. Dafür wird die Generalisierungsrelation definiert.

Definition 16 (Generalizität und Spezifizität von Präzedenzgraphen) *Ein Präzedenzgraph $G = (\mathbf{N}, \mathbf{E}_G)$ wird als* genereller *als ein Präzedenzgraph des selben Tasks mit den selben Operationen $S = (\mathbf{N}, \mathbf{E}_S)$ bezeichnet, genau dann, wenn $\mathbf{E}_G \subset \mathbf{E}_S$, d.h. wenn G nur einen Teil der Präzedenzbeziehungen von S enthält. Analog ist S* spezieller *als G, wenn $\mathbf{E}_S \supset .\mathbf{E}_G$. Die*

Abkürzungen $G \succeq_P S$ bzw. $S \preceq_P G$ stehen für die Aussagen „mindestens so generell wie“ beziehungsweise „mindestens so speziell wie“, welche durch

$$\mathbf{E}_G \subseteq \mathbf{E}_S \text{ beziehungsweise } \mathbf{E}_S \supseteq \mathbf{E}_G$$

definiert sind.

Generellere Taskpräzedenzgraphen erlauben dem Robotersystem mehr Umordnungsmöglichkeiten zum Ausführungszeitpunkt, speziellere schränken ihn deutlicher ein. Im Gegenzug können zu generelle Taskpräzedenzgraphen zu Ausführungsfehlern und damit verbunden zu kostspieligen Ausfällen oder Gefahrensituationen für das System oder sogar den menschlichen Benutzer führen. Speziellere Taskpräzedenzgraphen führen im schlimmsten Fall zu suboptimalen Ausführungen der Aufgabe führen. Aus an Sicherheitsaspekten orientierten Überlegungen sind somit speziellere Taskpräzedenzgraphen vorzuziehen, und diese sind konservativ zu verallgemeinern, d.h. nur wenn ausreichend Daten vorliegen, die eine weitere Verallgemeinerung stützen.

Im folgenden wird nun angenommen, daß das System nach der Eingabe von m Demonstrationen $\{D_1, \ldots, D_m\}$ den spezifischsten Taskpräzedenzgraphen $P_m = (\mathbf{N}, \mathbf{E}_m)$ gelernt hat und eine neue Demonstration derselben Aufgabe D_{m+1} observiert und diese der korrekten Aufgabenklasse zuordnen kann. In dieser Situation muß das System den gelernten Taskpräzedenzgraphen so anpassen, daß er in der neuen Demonstration eventuell vorhandenes, bisher unbekanntes Wissen über den Taskpräzedenzgraphen in selbigen integriert, daß heißt ihn gegebenenfalls verallgemeinert. Gemäß der Theorie des Versionsraumlernens [Mitchell 1997] kann dies durch sukzessive konservative Generalisierung der erreichten Hypothese geschehen, die genau so weit geht, daß die neue Hypothese gerade die speziellste Hypothese ist, die die vorherige umfasst und gerade noch die neue Demonstration. Angewandt auf die konkrete Repräsentation der Hypothese durch Taskpräzedenzgraphen, ist es notwendig, zu vermeiden, daß relevante Präzedenzrelationen aus der Kantenmenge des Vorranggraphen gestrichen werden. Deshalb wird die minimale Generalisierung des Taskpräzedenzgraphen P_m gewählt, die mit der die Vorführung konsistent ist. Formal kann man nun die beste Wahl für den neuen Präzedenzgraphen $P_{m+1} = (\mathbf{N}, \mathbf{E}_m)$ formulieren als

$$P_{m+1} \succeq_P P_m \wedge P_{m+1} \mapsto D_{m+1} \wedge (\nexists P' : P' \mapsto D_{m+1} \wedge P_{m+1} \succ P' \succ P_m). \quad (6.4)$$

Formuliert man Formel 6.4 in Mengenschreibweise um, so kann die neue Menge an Präzedenzrelationen $\mathbf{E}_{m+1}$ als eine Funktion der vorherigen gelernten Hypothese $P_m = (\mathbf{N}, \mathbf{E}_m)$ und der restriktivsten Hypothese $P^{D+1} = (\mathbf{N}, \mathbf{E}^{D+1)}$, welche gemäß Gleichung 6.3 berechnet werden kann:

$$\mathbf{E}_{m+1} = \mathbf{E}_m \cap \mathbf{E}^{D+1} \quad (6.5)$$

Die skizzierte Vorgehensweise läßt sich auch durch das Ausschluß- oder Kontradiktions-Eliminations-Verfahren beschreiben: Initial wird angenommen, daß alle Reihenfolgebeziehungen der ersten Demonstration relevant sind. Wird nun im folgenden die Erfahrung gemacht, daß bestimmte Reihenfolgebeziehungen vom Menschen bei der Demonstration einer Aufgabe nicht eingehalten werden, so entsteht ein Wiederspruch zwischen der Hypothese bzw. dem internen Modell des Systems. Dieser Wiederspruch wird nun durch die Adaption der Hypothese gemäß Gleichung (6.5) aufgelöst. Dies hat zum Resultat, daß die Hypothese konform mit der bisherigen Erfahrung ist. Wird die Hypothese durch erneute Demonstrationen in Frage gestellt, so beginnt der Lernprozeß von neuem. So ist ein inkrementell lernendes System entstanden, welches sein Modell des zu beherrschenden Handlungswissens kontinuierlich und mit offenem Ende während der gesamten Lebens- bzw. Operationszeit des Systems aktualisiert und verbessert.

Abschließend kann der Prozeß des inkrementellen Präzedenzgraphenlernens folgendermaßen zusammengefaßt werden, wie er ursprünglich von [Pardowitz et al. 2005, Zöllner et al. 2005] beschrieben und später von [Ekvall und Kragic 2006] verifiziert wurde:

1. Für die erste Demonstration D_1 einer neuen Taskklasse wird der Taskpräzedenzgraph mit $P_1 = P^{D_1}$ gemäß Gleichung 6.3 initialisiert.

2. Für jede weitere auftretende Demonstration D_{m+1} derselben Taskklasse wird ebenfalls der restriktivste Präzedenzgraph $P^{D_{m+1}}$ konstruiert. Anschließend wird mittels Maximierung der Subaufgaben-Korrespondenz $sk()$ die optimale Operatorpermutation zwischen der bisher erhaltenen Hypothese P_m und der neuen Demonstration D_{m+1} ermittelt. Zuletzt wird gemäß Gleichung 6.5 die speziellste Widerspruchsfreie Hypothese gebildet und zur Basis für weitere Lernschritte sowie Ausführungsoperationen genommen, bis sie durch weitere Erfahrungen einer erneuten Abänderung bedarf.

Visuell intuitiv läßt sich dieser Algorithmus des Erschließens von Taskpräzedenzgraphen durch die Elimination wiedersprüchlicher Kanten in der transitiven Hülle der Vereinigung von Reihenfolgebeziehunhen darstellen. Seien zwei unterschiedliche Taskpräzedenzgraphen $(\mathbf{N}_1, \mathbf{E}_1)$ und $(\mathbf{N}_2, \mathbf{E}_2)$ gegeben und existiere eine Permutation $P_{\mathbf{N}_1,\mathbf{N}_2}$, so bildet man mit der Vereinigung der transitiven Hülle den gemeinsamen Graphen $(\mathbf{N}_1, \mathbf{E}')$ mit

$$\mathbf{E}' = \mathbf{E}_1^+ \cup \left\{ \left(P^{-1}_{\mathbf{N}_1,\mathbf{N}_2}(i), P^{-1}_{\mathbf{N}_1,\mathbf{N}_2}(j) \right) | (i,j) \in \mathbf{E}_2^+ \right\}.$$

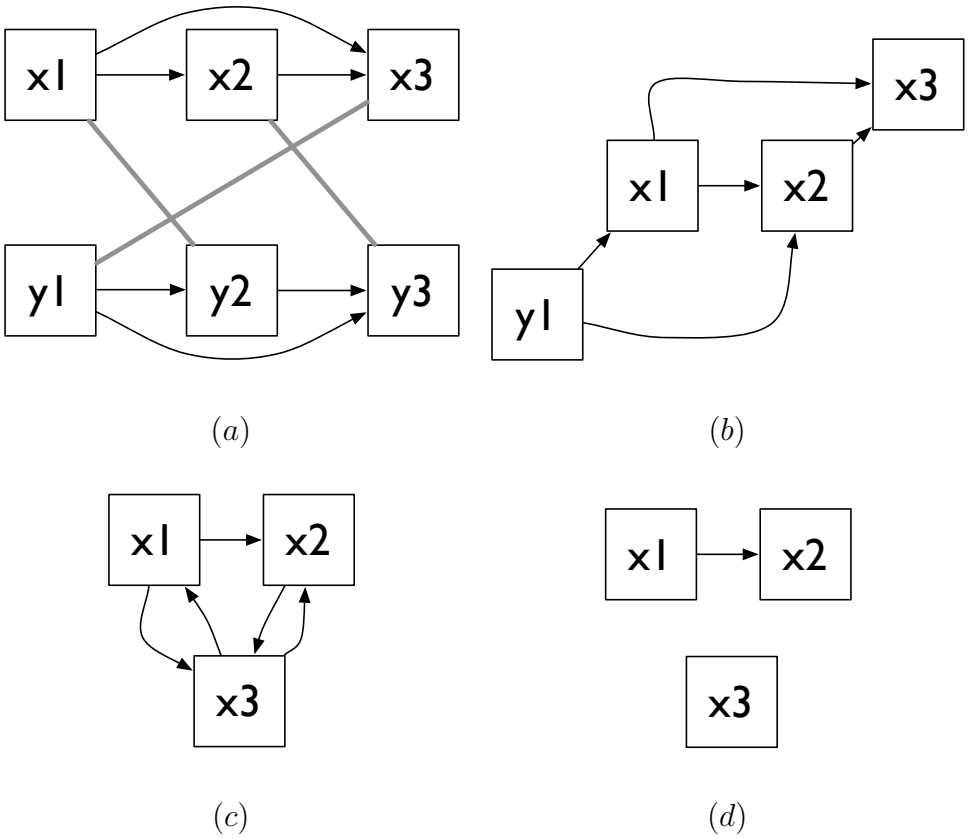


Abbildung 6.3: Visuell intuitive Darstellung des Algorithmus zum Erschliessen des Taskpräzedenzgraphen. (a) Ausgangslage mit zwei Sequenzen. Äquivalenzen in grau eingezeichnet. (b) Verschmelzung der Knoten x1 mit y2 bzw. x2 mit y3. (c) Verschmelzung des Knoten x3 mit y1. Auftritt von Widersprüchen in der Menge der Präzedenzrelationen. (d) Auflösung der Widersprüche.

Widersprüchliche Kanten treten auf, wenn man Kanten findet, die in zwei Richtungen existieren, d.h. wenn gilt

$$\exists (i,j) \in \mathbf{E}' : (j,i) \in \mathbf{E}'.$$

Die Bedeutung von einer solchen Aussage gemäß der Definition eines Task präzedenzgraphen (siehe Definition 15) wäre nämlich, daß i vor j ausgeführt werden muß, aber auch j vor i, was definitiv ein unerfüllbarer Taskpräzedenzgraph wäre. Dem steht aber entgegen, daß durchaus Demonstrationen der Aufgabe existieren, die einen zugrundeliegenden Präzedenzgraph erfüllen. Daher müssen solche Kanten gestrichen werden.

Eine solche Situation ist in Abbildung 6.3 dargestellt. In 6.3(a) sind die Sequenzen zweier Aufgabendemonstrationen abgebildet. Die Vorrangbeziehungen sind durch pfeilförmige Verbindungen dargestellt, wobei die transitiven Kanten durch geschwungene Form gekennzeichnet sind. Die optimale Permutation, welche schon aus Abbildung 6.1 bekannt ist, ist durch graue dicke Kanten markiert. Schrittweise werden nun die durch graue Kanten verbundenen Knoten miteinander verschmolzen. Das Ergebnis nach der Verschmelzung der Knoten x_1 mit y_2 bzw. x_2 mit y_3 ist in Abbildung 6.3(b) zu sehen. Bei der Verschmelzung von y_1 mit x_3 (siehe 6.3 (c)) treten dann die oben erwähnten Widersprüche in der Wissensrepräsentation auf. Es existieren plötzlich Kanten zwischen x_1 und x_3 bzw. x_2 und x_3 in entgegengesetzte Richtungen, was zu einem Taskpräzedenzgraphen führt, der keinerlei mit ihm konsistente Sequenz haben kann. Einzige Möglichkeit, das Handlungswissen wieder in einen konsistenten Zustand zu überführen, ist, die widersprüchlichen Kanten zu streichen. Das Endergebnis ist in Abbildung 6.3(d) dargestellt. Es existieren keinerlei Abhängigkeiten mehr zum oder vom Knoten x_3, während die in beiden Demonstrationen observierte Abhängigkeit zwischen x_1 und x_2 bestehen bleibt.

6.3 Versionsraumalgebren

Im Verfahren zum Lernen von sequentiellen Unabhängigkeiten aus dem vorangegangenen Abschnitt konnte auf eine relativ einfache Hypothesenrepräsentation zurückgriffen werden, nämlich das Netzwerk eines gerichteten Graphen. Dies erlaubt die Anwendung vergleichbar einfacher Lernmethoden. Sollen jedoch komplexere Programme gelernt werden, wie sie durch die Hinzunahme repetitiver Aufgaben auftreten, sind solche „flachen “ Repräsentationen nicht länger ausreichend. Hierfür erscheinen *Versionsraumalgebren* (siehe [Lau et al. 2004]) eine machbare Alternative. In diesem Abschnitt soll diese grundlegende Theorie vorgestellt werden, welcher der Erschließung von Wissen über repetitive Schleifenkonstrukte aus multiplen Demonstrationen zugrunde liegt.

Versionsraumalgebren basieren auf der Formulierung des Problems des Maschinellen Lernens eines Konzeptes als Suche im Raum der mit einer Trainingsdatenmenge konsistenten Hypothesen, wie sie ursprünglich von T. Mitchell vorgenommen wurde (siehe [Mitchell 1982]). Eine im Konzeptlernen eingesetzte Hypothese ist dabei eine binärwertige Funktion, welche komplexe Objekte auf einen Wahrheitswert abbildet. Im Programmieren durch Vormachen, wie es hier betrachtet wird, geht es im Gegensatz dazu um Funktionen, die komplexe Objekte in einen Raum komplexer Objekte abbildet.

Um Versionsraumalgebren formal genauer fassen und analysieren zu können, sind einige grundlegende Definitionen nötig.

Definition 17 (Hypothesen, Hypothesenräume und Bias) *Eine* Hypothese *ist definiert als eine Funktion, welche ein Element aus ihrem Definitionsbereicht* $\mathbf{I}$ *auf ein Element ihres Wertebereichs* $\mathbf{O}$ *abbildet. Ein* Hypothesenraum $\mathbf{H}$ *ist eine Menge von Funktionen mit derselben Signatur, d.h. alle Elemente dieses Hypothesenraums haben denselben Definitions- und Wertebereich* $\mathbf{I}$ *bzw.* $\mathbf{O}$*. Der* Bias *bestimmt, welche Teilmenge des Universums aller möglichen Funktionen mit der Signatur* $\mathbf{I} \mapsto \mathbf{O}$ *den Hypothesenraum konstituiert.*

Ein stärkerer Bias korrespondiert mit einem kleineren Hypothesenraum, einer geringeren Unsicherheit und damit mit einem schnelleren Lernergebnis, aber einer eingeschränkten Mächtigkeit des Lernverfahrens, da eventuell relevante mögliche Funktionen nicht mehr im Hypothesenraum liegen. Umgekehrt ermöglicht ein schwächerer Bias einen größeren Hypothesenraum und damit eine höhere Mächtigkeit, bei erhöhtem Aufwand an benötigten Trainingsdaten. Für eine genauere Diskussion, siehe [Mitchell 1997].

Definition 18 (Konsistenz) *Ein Trainingsbeispiel* (i, o) *mit* $i \in \mathbf{I}, o \in \mathbf{O}$ *und eine Hypothese* $h \in \mathbf{H}$ *sind miteinander konsistent, genau dann, wenn gilt* $h(i) = o$*. Dieser Fakt ist ausgedrückt durch das Konsistenzprädikat*

$$C(h, \mathbf{D}) \equiv \bigwedge_{(i,o)\in\mathbf{D}} h(i) = o.$$

Eine Menge an Trainingsdaten $D = \{D_1, D_2, \ldots, D_n\}$ *und eine Hypothese sind miteinander konsistent, genau dann, wenn jedes Trainingsdatum* D_i *und die Hypothese* h *miteinander konsistent sind für alle* i *mit* $1 \leq i \leq n$*.*

Definition 19 (Versionsraum) *Ein Versionsraum* $\mathbf{V}_{\mathbf{H},\mathbf{D}} \subset \mathbf{H}$ *ist die Menge aller Hypothesen* $h \in \mathbf{H}$*, welche mit der Trainingsdatenmenge* $\mathbf{D}$ *konsistent sind. Im folgenden werden die Indizes* $\mathbf{H}$ *und* $\mathbf{D}$ *weggelassen, wenn klar ist, welcher Hypothesenraum und welche Datenmenge gemeint ist.*

Wird die Trainingsdatenmenge vergrößert, im hier gegebenen Fall durch die Observation einer neuen Demonstration einer Aufgabe, so ändert sich u. U. der zugehörige Versionsraum und muß aktualisiert werden, um sicherzustellen, daß er nur noch die Hypothesen enthält die die Versionsraumeigenschaft erfüllen.

[Hirsh 1991] zeigte, daß für das von Mitchell propagierte Konzeptlernen die Versionsräume durch ihre generellsten und speziellsten Grenzen darstellbar sind[2], wenn nur eine „genereller-als“ Relation auf den verschiedenen Funktionen definiert ist. Dies erlaubt eine kompakte und dennoch aussagekräftige Repräsentation. Der Versionsraum besteht nämlich aus allen Hypothesen, welche genereller als die speziellste Grenze, aber nicht genereller als die generellste Grenze sind. Die ursprüngliche Implementation der Versionsräume machte von dieser Aussage zur effizienten Repräsentation der Versionsräume ausgiebigen Gebrauch. Es ist jedoch nicht immer möglich, die benötigte Generalisierungsrelation zu definieren, deshalb macht die hier verwendete Methode Gebrauch von sogenannter hierarchischer Dekomposition, welche eine andere Möglichkeit definiert, komplexe Versionsräume zu strukturieren.

Diese Methode beruht auf der Aufgabe des Konzeptes eines monolithischen Versionsraums und führt stattdessen die hierarchische Gliederung von Hypothesen- und Versionsräumen ein. Es treten nun zwei verschiedene Arten von Versionsräumen auf: Atomare Versionsräume gemäß der Definition von Mitchell und komponierte Versionsräume, welche durch Anwendung von Kompositionsoperatoren aus anderen (entweder komponierten oder atomaren) Versionsräumen entstanden sind. Diese Komposition bzw. Dekomposition ermöglicht eine effiziente Darstellung komplexer und umfangreicher Hypothesenräume. Komposition von Versionsräumen wird durch die Definition einer algebraischen Struktur auf der Menge der Versionsräume ermöglicht. Dabei wird ein komplexer, höherwertiger Versionsraum aus den Versionsräumen, welche einfachere Funktionen enthält, zusammengesetzt. Diese Kompositionsmethoden sind die Vereinigung[3] und die Konkatenation[4] von Versionsräumen, die jeweils wieder einen neuen Versionsraum ergeben.

Definition 20 (Vereinigung von Versionsräumen ($\cup$)) *Seien* $\mathbf{H}_1$ *und* $\mathbf{H}_2$ *zwei Hypothesenräume, welche beide Funktionen derselben Signatur* $\mathbf{I} \mapsto \mathbf{O}$ *enthalten und* $\mathbf{D}$ *eine Menge von Trainingsbeispielen. Die Versionsraumvereinigung von* $\mathbf{V}_{\mathbf{H}_1,\mathbf{D}}$ *und* $\mathbf{V}_{\mathbf{H}_2,\mathbf{D}}$ *ist definiert als*

$$\mathbf{V}_{\mathbf{H}_1,\mathbf{D}} \cup \mathbf{V}_{\mathbf{H}_2,\mathbf{D}} = \mathbf{V}_{\mathbf{H}_1 \cup \mathbf{H}_2,\mathbf{D}}$$

Die Vereinigungsoperation ist gemäß [Lau et al. 2003, Lau 2001] kommutativ und assoziativ. Daraus läßt sich für spätere Zwecke folgern, daß die Reihenfolge, in welcher die Trainingsdaten eintreffen irrelevant für das Lernergebnis sind.

[2] engl.: Boundary Set Representable (BSR).
[3] engl. Union
[4] engl. Join

Definition 21 (Konkatenation von Versionsräumen ($\triangleright\triangleleft$)) *Sei* $\mathbf{D}_1 = \{d_1^j\}_{j=1}^n$ *eine Menge von n Trainingsbeispielen von der Form* (i, o)*, wobei* $i \in \mathbf{I}, o \in \mathbf{O}$ *und ähnlich für* $\mathbf{D}_1 = \{d_2^j\}_{j=1}^n$*. Sei nun* $\mathbf{D}$ *die Folge von n Paaren* (d_1^j, d_2^j)*. Die Konkatenation der beiden dazugehörigen Versionsräume* $V_{\mathbf{H}_1,\mathbf{D}_1} \triangleright\triangleleft V_{\mathbf{H}_2,\mathbf{D}_2}$ *ist die Menge geordneter Paare, welche mit den Trainingsdaten konsistent sind, d.h.*

$$V_{\mathbf{H}_1,\mathbf{D}_1} \triangleright\triangleleft V_{\mathbf{H}_2,\mathbf{D}_2} \equiv \{(h_1, h_2) | h_1 \in V_{\mathbf{H}_1,\mathbf{D}_1}, h_2 \in V_{\mathbf{H}_2,\mathbf{D}_2}, C((h_1, h_2), \mathbf{D})\}.$$

Versionsraum-basierte Algorithmen weisen einen gemeinsamen Nachteil auf: Sie können nicht mit verrauschten Daten umgehen. Deshalb wird hier noch eine probabilistische Erweiterung der Versionsraumalgebra vorgeschlagen, welche genau diesen Nachteil zu umgehen versucht. Jede Hypothese im Versionsraum wird nun mit einer Wahrscheinlichkeit versehen. Dies erlaubt es nicht nur, Rauschen in den gemessenen Daten zu tolerieren, indem Hypothesen niemals aus dem Versionsraum eliminiert werden, sondern ihnen nur eine kleine Wahrscheinlichkeit ungleich Null zuzuweisen, so daß inkonsistente Hypothesen niemals „unwiderbringlich“ aus dem Versionsraum entfernt werden, sondern lediglich „hintenangestellt“ werden. Zusätzlich kann unter Umständen noch Domänenwissen eingeführt werden, welche Hypothesen mit höherer Wahrscheinlichkeit auftreten, als andere. Weiterhin kann bereits ein Lernergebnis einfach erzielt werden, bevor der Versionsraum gegen eine einzige Hypothese konvergiert, indem eine Gesamtwahrscheinlichkeit für die beste Hypothese ermittelt wird.

Zusätzlich zum Entwurf der Versionsräume müssen nun a-priori Wahrscheinlichkeiten Pr definiert werden, so daß für atomare Versionsräume $\mathbf{H}$

$$\sum_{h \in \mathbf{H}} Pr(h|\mathbf{H}) = 1$$

ist, sowie für Versionsräume $\mathbf{V}_i$ einer Versionsraumsvereinigung $\mathbf{W}$

$$\sum_{h \in \mathbf{H}} Pr(\mathbf{V}_i|\mathbf{H}) = 1.$$

Die Wahrscheinlichkeit P für eine Hypothese läßt sich nun für alle Fälle folgendermaßen definieren:

$$P(h|\mathbf{V}) = \begin{cases} Pr(h|\mathbf{V}), & \text{falls } \mathbf{V} \text{ atomar,} \\ \sum_{\forall i | h \in \mathbf{V}_i} w_i \cdot P(h|\mathbf{V}_i), & \text{falls } \mathbf{V} \text{ Versionsraumvereinigung von} \mathbf{V}_i, \\ k \times \prod_i P(h_i|\mathbf{V}_i), & \text{falls } \mathbf{V} \text{ Konkatenation von } \mathbf{V}_i. \end{cases}$$

Hierbei sind w_i die a-priori-Gewichtungen für die einzelnen Teilversionsräume und k eine normalisierende Konstante. Unter Benutzung dieser probabilistischen Versionsraumerweiterung kann jeder Hypothese in einem Versionsraum eine Wahrscheinlichkeit zugeordnet werden, so daß zu einer Menge

von Demonstrationen **D** eine nach Wahrscheinlichkeit geordnete Liste von Hypothesen erzeugt werden kann.

In diesem Abschnitt wurde das theoretische Rüstzeug entwickelt, welches es erlaubt, repetitive Aufgabenstellungen zu analysieren und Schleifenkonstrukte zu extrahieren. Dies ist das Thema des folgenden Abschnitts.

6.4 Lernen von Kontrollstrukturen für repetitive Aufgaben

Ein wichtiges Gebiet, in welchem PdV-Systeme eingesetzt werden, ist die Ausführung von repetitiven, eintönigen Aufgaben. Dem Benutzer sollen von PdV-Systemen in den unterschiedlichsten Domänen sich wiederholende und nur in Details unterschiedliche Aufgaben abgenommen werden, indem aus Beobachtungen des Verhaltens des Benutzers die zugrundeliegende Handlungsabsicht erkannt wird. Diese wird dann verwendet, um die Umsetzung automatisch und autonom vom lernenden System durchführen zu lassen.

Ein Beispiel für eine repetitive Aufgabe in der Haushaltsdomäne ist das Ausräumen einer Spülmaschine: Hier geht es darum, alle Objekte, die im Spülmaschinenkorb vorhanden sind, aufzunehmen und an einen bestimmten Ort zu stellen. Die repetitive Aufgabe ist es, jedes einzelne Objekt an den zugehörigen Platz aufzuräumen. Eine andere, im Rahmen dieser Arbeit betrachtete Aufgabe, besteht darin, für n Personen den Tisch zu decken, wobei jede Person dasselbe Gedeck erhalten soll, beispielsweise eine mit Tee zu befüllende Tasse auf einer Untertasse. Diese beiden Instanzen von repetitiven Aufgaben zeigen schon eine große Bandbreite an variablen und konstanten Determinanten einer Aufgabe: Im ersten Fall ist lediglich die Eigenschaft der Objekte, vor der Ausführung der Aufgabe im Spülmaschinenkorb zu sein, von Bedeutung, im anderen Fall ist die Objektklasse variabel und außerdem Determinante für die Endposition des Objektes.

Die Versionsraumhierarchie, die im Rahmen dieser Arbeit entwickelt und verwendet wurde, ist in Abbildung 6.4 dargestellt. Die folgende Diskussion zeigt zuerst auf, wie die invarianten und variablen Bestandteile für einzelne Aktionen gelernt werden können, bevor auf das Lernen ganzer repetitiver Programme eingegangen wird.

Abbildung 6.4(c) zeigt die hierarchische Definition des Versionsraums für eine Aktion. Jede Funktion in diesem Versionsraum bildet einen Weltzustand in einen neuen Weltzustand ab. Ein Trainingsbeispiel für diesen Versionsraum besteht aus einem Makrooperator der zweiten, also der Einzelaktionsebene. Dementsprechend besteht der Versionsraum für eine Aktion aus den wesent-

lichen Einzelaktionen, welche im implementierten System erkannt werden können: Dem Transport eines Objektes, dem Öffnen eines Objektes sowie von Einschenkoperationen.

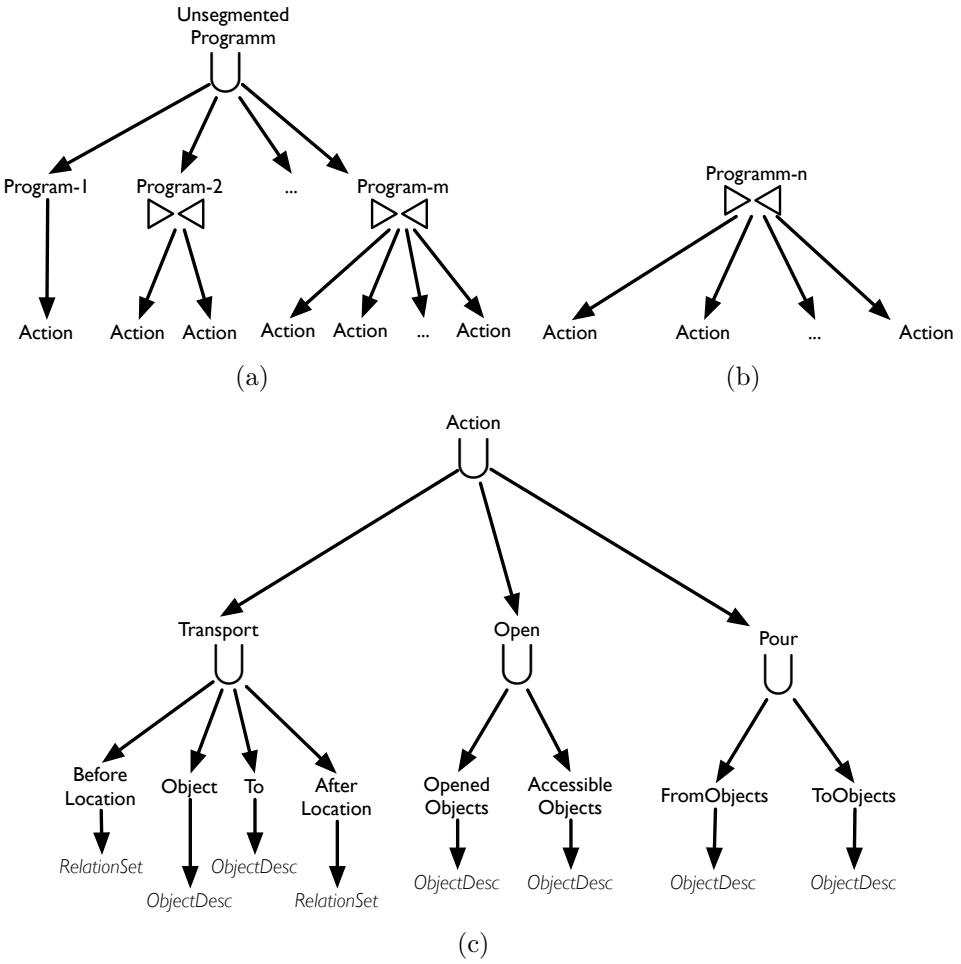


Abbildung 6.4: Versionsräume zum Erlernen repetitiver Aufgaben. (a) Versionsraum für ein unsegmentiertes Programm mit unbekannter Länge. (b) Versionsraum für ein Programm mit bekannter Länge. (c) Versionsraum für einzelne Aktionen. Atomare Versionsräume sind jeweils kursiv dargestellt, zusammengesetzte in normaler Schrift.

Transportoperationen können durch vier verschiedene Entitäten parametriert werden. Dazu gehören die Vorbedingungen vor Durchführung der Aktion, das bewegte Objekt, das Objekt, zu welchem die Operation hinführte (dem Relativobjekt), sowie die relationale Beschreibung des Weltzustandes nach Durchführung der Aktion. Der erste sowie der letzte dieser Parameter ist

jeweils ein atomarer Versionsraum, welcher eine verallgemeinerte relationale Beschreibung von Weltzuständen gemäß 3 darstellt. Die objektbezogenen Parameter sind durch atomare Versionsräume gegeben, welche Objektverallgemeinerungen betrachten.

Der atomare Versionsraum für Relationenmengen $V_{R\text{elationSet},\mathbf{D}}$ ist folgendermaßen definiert:

- Besteht die Demonstrationsdatenmenge $\mathbf{D}$ aus lediglich einem einzigen Element, ist also $|\mathbf{D}| = 1$, so wird der Relationen-Versionsraum mit der Vorbedingung Pre der Makrooperator-Aktion initialisiert, also

$$V_{R\text{elationSet},\mathbf{D}} = Pre.$$

- Ist $\mathbf{D} = \{D_1, D_2, \ldots, D_{n+1}\}$, so ist bereits in vergangenen Iterationsstufen ein Versionsraum $V_{R\text{elationSet},\mathbf{D}'}$ mit $\mathbf{D}' = \{D_1, D_2, \ldots, D_n\}$ konstruiert worden. Sei nun Pre wieder die Vorbedingung für die neu hinzugefügte Aktion, so konstruiert sich der neue Versionsraum aus

$$V_{R\text{elationSet},\mathbf{D}} = V_{R\text{elationSet},\mathbf{D}'} \bigcap Pre.$$

Der Versionsraum für Objektbeschreibungen $V_{O\text{bjDesc},\mathbf{D}}$ ist strukturiert durch die Generalitätsbeziehung $\succeq_O$, welche folgendermaßen definiert ist:

$$\mathbf{O} \succeq_O \{C(O)\} \succeq_O \{O\} \succeq_O \emptyset.$$

Hierbei bezeichnet für ein Objekt O $C(O)$ die zu diesem Objekt gehörige Objektklasse und $\mathbf{O}$ die Menge aller dem System bekannten Objekte. Ein einzelnes Objekt läßt sich also zu allen Objekten derselben Art (die dieselbe Objektklasse aufweisen) verallgemeinern und weiter zur Menge aller Objekte.

Der atomare Versionsraum für die Objektbeschreibung $V_{O\text{bjDesc},\mathbf{D}}$ wird nun folgendermaßen definiert:

- Besteht die Demonstrationsdatenmenge $\mathbf{D}$ aus lediglich einem einzigen Element ($|\mathbf{D}| = 1$), so wird der Versionsraum $V_{O\text{bjDesc},\mathbf{D}}$ mit der Menge initialisiert, welche genau das beschriebene Objekt O enthält, also

$$V_{O\text{bjDesc},\mathbf{D}} = \{O\}.$$

- Ist $\mathbf{D} = \{D_1, D_2, \ldots, D_{n+1}\}$, so wird (falls nötig) der Objekt-Versionsraum $V_{O\text{bjDesc},\mathbf{D}'}$ mit $\mathbf{D}' = \{D_1, D_2, \ldots, D_n\}$ gemäß der oben definierten Generalitätsbeziehung $\succeq_O$ so weit verallgemeinert, bis gilt:

$$V_{O\text{bjDesc},\mathbf{D}} \succeq_O V_{O\text{bjDesc},\mathbf{D}'} \wedge V_{O\text{bjDesc},\mathbf{D}} \succeq_O O.$$

Basierend auf diesen beiden atomare Versionsräumen lassen sich nun die komponierten Versionsräume zusammensetzen. Der Versionsraum für eine Transportoperation ist demnach (siehe Abbildung 6.4 (c)) als Vereinigung der Versionsräume von Relationenmengen für Vor- und Nachbedingungen für den Transport dargestellt, zusammen mit Objektdeskriptoren für das manipulierte und das relative Objekt. Im oben zitierten Beispiel des Ausräumens eines Spülmaschinenkorbs würde nach einigen Demonstrationen idealerweise der Versionsraum für die Transportoperation sich wie folgt darstellen: Der Teilversionsraum für die Vorbedingungen würde gegen die Bedingung konvergieren, daß sich das manipulierte Objekt innerhalb des Spülmaschinenkorbs befand. Das manipulierte Objekt ist nicht weiter bestimmt, genauso, wie das Relativobjekt, da die Aufgabenbeschreibung die Aufforderung enthält, alle Objekte aus dem Spülmaschinenkorb, ohne Ansehen der Objektklasse, zu entfernen. Da auch die Ablageposition nicht näher spezifiziert ist, sind beide Objekt-Versionsräume mit der Klasse aller Objekten, also **O** belegt. Die Belegung der Nachbedingungen hängt von der konkreten Ausführung der Demonstrationen ab, ist aber unter Umständen auch leer. Damit ist der Versionsraum für die gesamte Transportbewegung nur durch die Vorbedingung, daß sich das Objekt im Spülmaschinenkorb befinden muß, ausreichend beschrieben.

Im Fall des anderen Beispiels, des Tischdeckens, sei hier die Aktion des Transports einer Tasse beschrieben, so daß diese schlußendlich auf einer Untertasse steht. Hier wird der Versionsraum für die Vorbedingung unter Umständen kollabieren, falls die Tassen von unterschiedlichen Positionen aufgenommen werden. Stattdessen sind die anderen drei Versionsräume interessanter: Der Versionsraum für das manipulierte Objekt wird genau der Klasse von Tassen, der des Relativobjektes der Klasse der Untertassen entsprechen. Dies trägt der Tatsache Folge, daß diese Aktion mehrfach für zwar unterschiedliche Objektinstanzen ausgeführt wurde, diese Instanzen also nicht allgemein genug waren, um alle Demonstrationen abzudecken, jedoch die jeweiligen Objektklassen diese Eigenschaft aufweisen. Die Nachbedingung wird gegen die Relation konvergieren, daß die Tassen auf der jeweiligen Untertasse stehen. Damit entspricht dieser Versionsraum der Aufgabe, sämtliche Tassen auf verfügbare Untertassen zu stellen.

Die anderen Aktionen, welche erlernt und verallgemeinert werden können, sind Öffnungsoperationen (beispielsweise eines Kühlschranks) und Einschenkoperationen (siehe Abbildung 6.4 (c)). Von diesen sollen hier jetzt nur noch die Einschenkoperationen detaillierter behandelt werden, da der andere Fall nahezu analog abzuhandeln ist. Besteht nun die Aufgabe darin, aus einer Teekanne in sämtliche Tassen (die beispielsweise mit den oben beschriebenen Methoden auf die Untertassen gestellt wurden) Flüssigkeit einzuschänken,

so ist hierbei der konstante Parameter die spezifische Teekanne, der variable hingegen die Tassen. Der erste Versionsraum enthält also nur ein konkretes Objekt, der zweite die Klasse der Tassen.

Der Versionsraum für allgemeine Aktionen besteht nun aus der Vereinigung der drei Versionsräume für Transport-, Öffnungs- und Einschenkoperationen. Das System ist jedoch um weitere Aktionen erweiterbar, indem einfach der entsprechende Versionsraum designiert und implementiert wird.

Da nun beschrieben wurde, wie einzelne Aktionen gelernt und verallgemeinert werden können, bleibt noch übrig, die einzelnen Aktionen zu Programm- bzw. Schleifendurchläufen zusammenzusetzen. Zunächst wird dazu angenommen, daß die Anzahl der Aktionen in einem Schleifendurchlauf n bekannt ist. Ein Schleifendurchlauf besteht also aus n unterschiedlichen Aktionen, welche durch sequentielle Verkettung zu einem Gesamtprogramm kombiniert werden. Der entsprechende Versionsraum entsteht durch eine n-fache Verkettung von Versionsräumen für Aktionen, wie sie in Abbildung 6.4 (b) dargestellt ist.

Im oben vorgestellten Beispiels des Decken eines Tisches zum Tee ist $n = 2$. Mit jedem Schleifendurchlauf sind zwei Aktionen durchzuführen: Das Platzieren einer Tasse auf einer Untertasse, sowie das Befüllen derselben Tasse unter Zuhilfenahme der Teekanne. Enthält die Demonstration mehr als zwei Aktionen, so sind also mehr als ein Schleifendurchlauf vorhanden und die Ausführung des Schleifenkörpers startet erneut, was zur Generalisierung der beiden Versionsräume für die beiden Basis-Aktionen verwendet wird. Das Programm besteht also aus zwei Versionsräumen, und zwar entsprechen diese einzeln den beiden oben besprochenen Aktions-Versionsräumen für die Aufgabe des herstellen eines Tee-Gedecks.

Betrachtet man nun den allgemeinen Fall, daß die die Anzahl der im Schleifenrumpf vorhandenen Aktionen n unbekannt ist, so stellt sich folgende Situation: Beobachtet wurde eine Demonstration mit m Aktionen. Geht man davon aus, daß der Benutzer mindestens eine komplette Durchführung des Schleifenrumpfes veranlaßt hat, so kann man daraus schließen, daß die Schleifenlänge maximal m beträgt. Beispielsweise kann der Benutzer eine Demonstration mit einer Länge des Schleifenkörpers von 3 beginnen, dann die ersten beiden Schritte des zweiten Schleifendurchlaufes starten und dann entscheiden, daß das System ausreichend gesehen hat, um die gesamte Schleife zu lernen. Der erhaltene Makrooperator enthält also fünf elementare Aktionen, die Schleifenlänge jedoch kann irgendwo zwischen einer und fünf Aktionen liegen.

Für diese Situation wurde der Versionsraum des unsegmentierten Programms entworfen (siehe Abbildung 6.4 (a)). Die wesentliche Idee besteht darin, Hypothesen für Programme aller möglicher Längen von 1 bis m auf-

zustellen und kontinuierlich zu aktualisieren. Dementsprechend besteht der Versionsraum für ein unsegmentiertes Programm also aus der Vereinigung von m Programmen mit jeweils wachsender Programmlänge von 1 bis m, welche als Verkettung von Aktionen ausgeführt sind. Der erste Versionsraum wird aktualisiert mit allen m Aktionen, als ob jede Aktion ein Beispiel eines Schleifendurchlaufs mit einem Rumpf von genau einer Aktion wäre. Der zweite Versionsraum wird mit $m/2$ Aktionspaaren aktualisiert, und so weiter. Weiterhin werden im probabilistischen Versionsraumverfahren alle Elemente der Vereinigung so gewichtet, daß kurze Programme gegenüber längeren bevorzugt werden. Dies verhindert, daß das Verfahren als optimale Hypothese immer einen einzelnen Schleifendurchlauf mit einer Rumpflänge von m annimmt.

In der Praxis kollabieren viele dieser Versionsräume sehr schnell, da die Aktionen nicht zueinander passen, außer für Programme mit einer Länge von ganzzahligen Vielfachen von n. Verschachtelte Schleifen, in denen einzelne Aktionen wiederum aus Schleife mit möglicherweise mehreren Aktionen bestehen, werden von diesem Versionsraum-Design nicht abgedeckt und bleiben Ausgangspunkt für zukünftige Forschung.

6.5 Zusammenfassung

Dieses Kapitel hat gezeigt, wie die in Klassen unterschiedlicher Aufgabendemonstrationen vorhandene Information genutzt werden kann, um ganzheitliches und abstrahiertes Handlungswissen zu erhalten.

Es wurden Verfahren vorgestellt, welche es erlauben, Teile unterschiedlicher Demonstrationen desselben Tasks miteinander in Beziehung zu setzen. Diese basieren auf das im vergangenen Kapitel eingeführte Ähnlichkeitsmaß (siehe Abschnitt 5.2) und haben zum Ziel, eine Permutation von Teiloperationen zu berechnen, welche die Summe des mit jeder Paarungbildung verbundenen Ähnlichkeitswertes maximiert. Dazu wurde ein Verfahren vorgestellt, welche die Ahnlichkeitsmatrix der Operationen zweier Demonstrationen zu einem gerichteten Graphen erweitern, auf den das Verfahren zur Maximierung des Flusses in Netzwerken anwendbar ist. Der maximale Fluß entlang eines Schnitts entlang der Grenze zwischen den beiden beteiligten Demonstrationen definiert dann genau die optimale Permutation, welche Paare von Operationen, welche den unterschiedlichen Demonstrationen angehören, bildet.

Diese Paarungen entsprechen äquivalenten Operationen. Die relativen Positionsdifferenzen zwischen unterschiedlichen Demonstrationen können in Präzedenzgraphen umgesetzt werden. Die Art und Weise, wie dies geschieht,

wurde im inkrementellen Verfahren zum Erschließen von Präzedenzgraphen umgesetzt. Dieses Verfahren beruht auf dem Ausschluß von mit den Demonstrationen aufgrund der vorkommenden Positionen inkonsistenten Präzedenzbeziehungen in den transitiven Reihenfolgegraphen der Demonstrationen.

Anschließend wurde ein abstraktes Verfahren zum parallelen Verfolgen multipler Hypothesen mittels hierarchischer Versionsräume vorgestellt. Diese komplexen Räume der mit einer Menge von Demonstrationen konsistenten Hypothesen entsteht durch die algebraische Verknüpfung von atomaren oder komponierten Versionsräumen.

Diese algebraische Struktur wurde im abschließenden Abschnitt für das Problem des Lernens von repetitiven Aufgaben eingesetzt. Es konnte eine Modellierung von Schleifenstrukturen sowie ihrer Rümpfe gefunden und präsentiert werden. Besondere Aufmerksamkeit erforderte hierbei die Bestimmung der Schleifenrumpflänge. Dies wurde erreicht durch eine konsequente Verfolgung mehrerer Hypothesen für unterschiedliche Längen und Ausscheiden der unwahrscheinlichen Lauflängen.

Nachdem in den vergangenen Kapiteln die lexikalischen, syntaktischen, semantischen und verallgemeinernden Analysephasen zum Lernen von Handlungswissen theoretisch durchdrungen wurden, steht eine experimentelle Validierung der vorgestellten Konzepte und Verfahren noch aus. Dies wird der Inhalt des folgenden Kapitels sein.

Kapitel 7

Experimentelle Evaluation

In diesem Kapitel werden die einzelnen Analysephasen und die darin verwendeten Verfahren evaluiert. Alle im Rahmen dieser Arbeit erstellten Algorithmen und Verfahren werden einer quantitativen Auswertung unterzogen. Dabei wird in der Darstellung der Ergebnisse in Reihenfolge der Theoriekapitel dieser Arbeit vorgegangen.

7.1 Evaluation der lexikalischen und syntaktischen Analyse

Im Rahmen der lexikalischen Analyse wurden elementare Operationen, wie das Greifen und Loslassen eines Objektes, oder spezielle Aktionen wie das Öffnen eines Kühlschrankes detektiert. Die syntaktische Analysephase stellte die detektiertn Ereignisse in einen Kontext, der nach grammatikalisch spezifizierten Regeln aufgebaut ist. Hierbei ist insbesondere darauf einzugehen, wie das System mit den unterschiedlichen Sensormodalitäten umgeht. Deshalb wurden einerseits Experimente mit einem Vicon-Tracking-System ausgewertet. Zum anderen wurden Daten von magnetfeldbasierten Positionssensoren und Datenhandschuhen, sowie dem Stereokamerasystemen (siehe Abschnitt 3.6) ausgewertet.

7.1.1 Transport einer Flasche

Abbildung 7.1 zeigt Schlüsselpunkte aus dem Experiment, welches aus dem Transport einer Flasche von einer vorgegebenen Start- zur Zielposition bestand. Diese Daten wurden im Rahmen des Sonderforschungsbereiches 588 „Humanoide Roboter“ aufgenommen und sollen primär der Ausformung

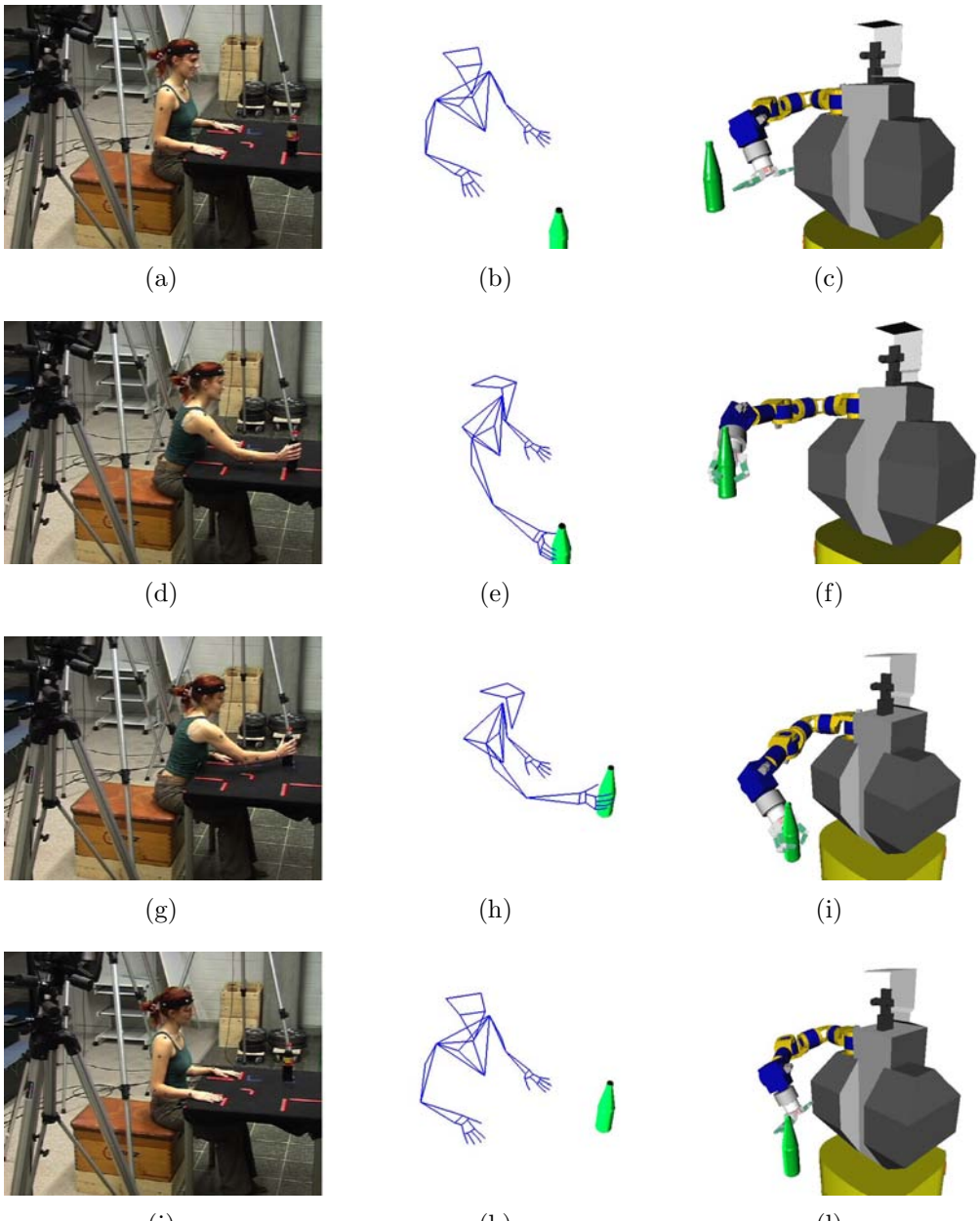


Abbildung 7.1: Experiment mit dem Ganzkörper-Tracking-System Vicon.

menschenähnlicher Bewegungsformen bei Manipulationen dienen. Es wurden insgesamt 448 Frames an Daten mit Hilfe eines Vicon-Motion-Tracking-Systems (siehe [Vicon 2006]) aufgenommen.

Im Initialzustand wurde die Versuchsperson angehalten, die Hände an dafür vorgesehene Markierungen auszurichten. Anschließend griff sie mit der

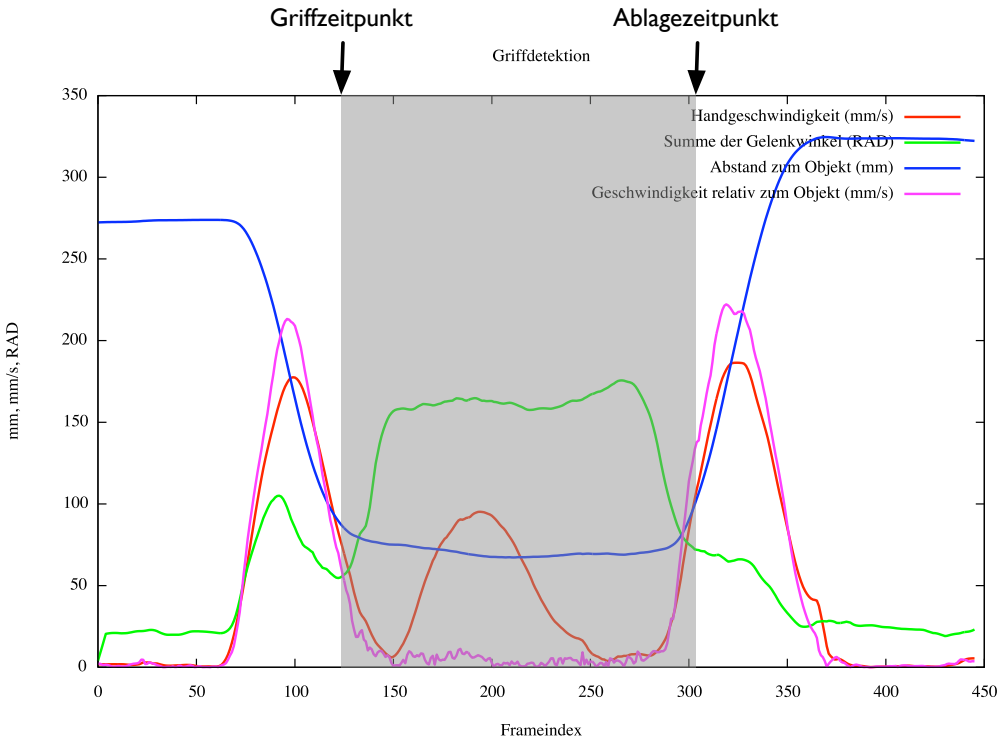


Abbildung 7.2: Lexikalische Analyse im Experiment mit den Daten des Bewegungserfassungssystems [Vicon 2006]

rechten Hand eine Flasche und überführte sie mit einer zielgerichteten Bewegung in die Endposition, woraufhin die beiden Arme wieder in die Positionen entlang den Markierungen ausgerichtet wurden.

Während der lexikalischen Analyse wurden mit Hilfe der in Abschnitt 4.2 präsentierten Verfahren zwei Ereignisse festgestellt: Das Greifen und das Loslassen der Flasche. Der Griffzeitpunkt wurde in Frame 133, der Ablagezeitpunkt in Frame 304 gefunden. Die Entwicklung der Sensormeßwerte bzw. deren abgeleitete Größen der Geschwindigkeit der Hand, der Summe aller Gelenkwinkel, des Abstandes der Hand zum gegriffenen Objekt und der Handgeschwindigkeit relativ zum Objekt, anhand derer die Griffdetektion gemäß Gleichung (4.1-4.4) durchgeführt wurde, ist in Abbildung 7.2 dargestellt. Der grau unterlegte Bereich entspricht genau der erkannten Greifphase.

Mit den Verfahren aus Abschnitt 4.7 erstellte das System anschließend die hierarchisch-funktionale Repräsentation des Handlungswissens. Das Ergebnis ist in Abbildung 7.3 visualisiert. Hierbei ist unter den Baumelementen für die elementaren Operatoren, den Bewegungs- und Griff- bzw. Ablageoperationen der jeweilige Operator aufgetragen. Die Bewegungsoperatoren sind durch die

abgefahrenen Trajektorien in rot dargestellt.

Man erkennt den korrekten Aufbau des Syntaxbaumes: Die gesamte Aufgabe besteht aus einer Transport-Operation, welche untergliedert ist in eine Pick- und eine Place-Phase. Jede von diesen beiden wiederum gliedert sich in eine Anfahrtstrajektorie, die Durchführung eines Griff- bzw. Loslassoperators und eine Abfahrtstrajektorie. Trajektorien sind untergliedert in einen oder mehrere lineare Bewegungsoperatoren. Die Linearisierung fand mit Hilfe des Verfahrens zum iterativen Anpassen der Endpunkte[1] statt.

7.1.2 Einschenken aus einer Flasche

Im vorangegangenen Experiment kamen die Sensordaten aus dem Motion-Capture- oder Bewegungserfassungssystems [Vicon 2006] zum Einsatz. Im Rest dieses Kapitels kommen vor allem Daten aus dem für Handlungen im Kontext von Arbeiten im Haushalt zugeschnittenen Demonstrations-Zentrum (siehe Abbildung 3.4) zum Einsatz, da bei ihnen eine reichhaltige Trainingsumgebung sowie eine große Zahl von Sensoren zur Handlungsbeobachtung für reproduzierbare Situationen realisiert wurde. Es sei angemerkt, daß das PdV-System mit beiden Sensormodalitäten arbeiten kann.

Der in diesem Abschnitt beschriebene Versuch hatte zum Ziel, komplexere Handlungsstränge, welche mittels raumgreifender Bewegungen ausgeführt wurden, zu erfassen und zu verarbeiten. Dazu hat die Versuchsperson einen Kühlschrank zu öffnen, daraus eine Flasche zu entnehmen, aus dieser Flasche in eine auf dem Tisch stehende Tasse einzuschenken und abschließend die Flasche auf dem Tisch abzustellen. Das System ist in diesem Experiment mit dem Vorhandensein unterschiedlicher Elementaren Operationen konfrontiert, speziell dem Öffnen des Kühlschranks, dem Eingießen aus einer Flasche, neben den schon bisher demonstrierten Fähigkeiten des Erkennens von Greif- und Loslaßvorgängen. Die aufgenommene Demonstration besteht insgesamt aus 188 Frames und dauerte etwa 9 Sekunden, was einem Durchsatz von 20 Frames pro Sekunde entspricht.

Die wesentlichen Episoden des Experiments sind in den Abbildungen 7.4 und 7.5 zu sehen. Die erste erkannte Aktion ist der Griff des Türgriffs des Kühlschranks in Frame 28 mit der rechten Hand. Selbige führt dann den Türgriff (und damit die verbundene Tür) auf einer zirkularen Bahn bis sie einen Öffnungswinkel von circa 90° erreicht. In Frame 47 ist diese Bewegung beendet und der Türgriff wird losgelassen. Anschließend bewegt sich die linke Hand zielorientiert in den Kühlschrank hinein, wo sie in Frame 71 die Flasche ergreift.

[1] engl. Iterative Endpoint Fit (IEPF).

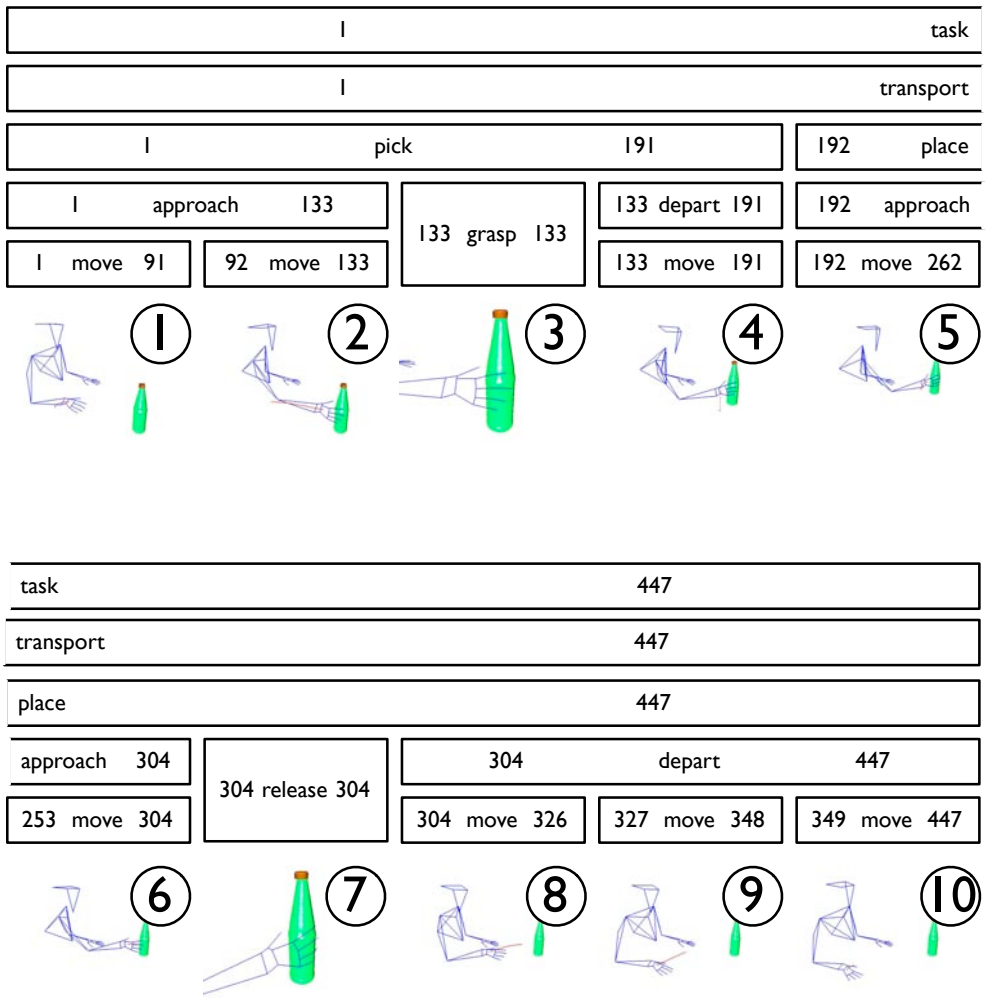
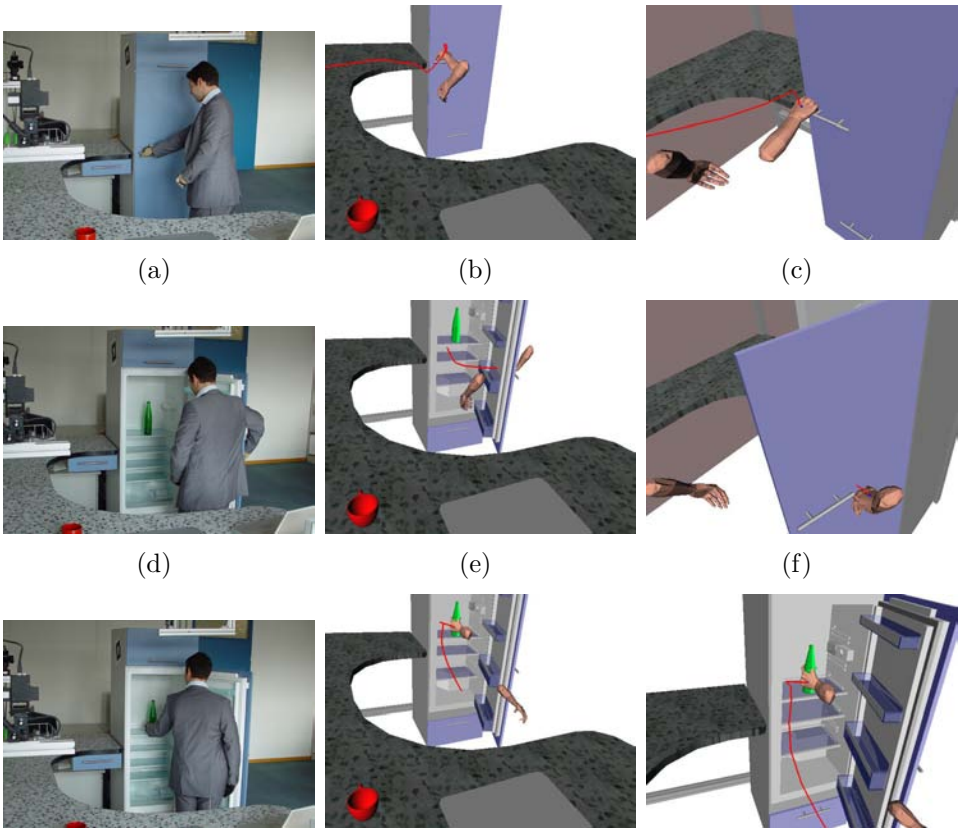


Abbildung 7.3: Syntaktische Analyse. Segmentierung der Demonstration in Teilsequenzen. (1, 2) Lineare Bewegungssegmente der Anfahtsbewegung zum Greifen. (3) Detail zum Griffzeitpunkt. (4) Ende der Abfahrtsbewegung. (5, 6) Lineare Bewegungssegmente der Anfahrtsbewegung zum Abstellen. (7) Detail des Ablagepunktes. (8-10) Lineare Bewegungssegmente der Abfahrtsbewegung und Rückkehr zum Ausgangspunkt.

Daraufhin transportiert die Versuchsperson die gegriffene Flasche aus dem Kühlschrank hinaus und ebenfalls zielorientiert zur Arbeitsfläche, auf der eine Tasse steht. Die Flasche wird in eine nahezu waagerechte Position gebracht und in Frame 127 wird eine Eingießoperation aus der gegriffenen Flasche in die Tasse detektiert. Zuletzt wird die Flasche am rechten Rand des silbernen Tischsets abgestellt, und zwar in Frame 168.



(a) (b) (c)

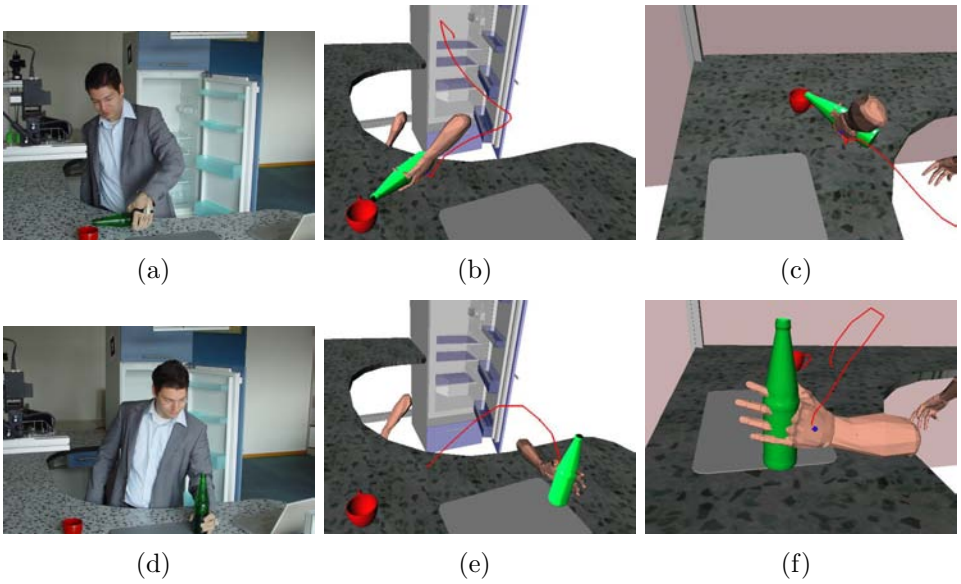
(d) (e) (f)

(g) (h) (i)

Abbildung 7.4: Schlüsselszenen des zweiten Experiments zum Erfassen komplexer zweihändiger Aufgaben (I). (a-c) Frame 28. (d-f) Frame 47. (g-i) Frame 71. (wird fortgesetzt)

Die Verfahren zur Segmentierung der Demonstration aus Abschnitt 4.5 werden zur Erkennung der Kontextwechselpunkte in den Frames 38, 58, 99 und 148 angewandt. Hierbei sind folgende Einteilungen gemacht worden: Der Kontextwechselpunkt in Frame 38 teilt die zirkulare Bewegung zum Öffnen des Kühlschranks in zwei Hälften. Der (beidhändige) Kontextwechselpunkt in Frame 58 markiert das Ende der gesamten Teilaufgabe zum Öffnen des Kühlschranks (mit der rechten Hand) bzw. den Beginn der Greifbewegung hin zur Flasche (mit der linken Hand). Die gesamte Einschüttoperation wird weiterhin an den Punkten 99 und 148 unterteilt, um dem Wechsel in der Benutzerintention vom Greifen der Flasche hin zur eigentlichen Einschüttoperation, bzw. von dieser hin zum Abstellen der Flasche Rechnung zu tragen.

Mit Methoden zum Erstellen der syntaktischen Struktur dieser Demonstration, wie sie in Abschnitt 4.8 basierend auf einer Handlungsgrammatik



(a) (b) (c)

(d) (e) (f)

Abbildung 7.5: (Fortsetzung) Schlüsselszenen des zweiten Experiments zum Erfassen komplexer zweihändiger Aufgaben (II). (a-c) Frame 127. (d-f) Frame 168. Die linke Spalte zeigt die Versuchsperson beim Durchführen der Aufgabe, die mittlere den Überblick über die sensorische Erfassung der Szene, in der rechten Spalte sieht man detaillierte Aufnahmen zur genaueren Bewertung des Geschehens.

beschrieben wurden, erstellt das System den in Abbildung 7.6 dargestellten Strukturbaum. Hierbei wurde (abweichend von Abbildung 7.3) die unterste Ebene der einzelnen Bewegungselementaroperationen aus Platzgründen nicht aufgeführt. Die Bedeutung der Kontextwechselpunkte kann man anhand der Höhe der Lücken in den Segmenten erkennen: Der Kontextwechselpunkt bei Frame 38 hat den Wechsel von der Pick- hin zur Place-Operation zur Folge (3. Stufe). Er trennt lediglich zwei einhändige Operationen voneinander. Der Kontextwechselpunkt in Frame 58 hingegen geht bis hoch zur 2. Stufe. Dieser trennt zwei semantisch höhere Operationen von gröberer Granularität voneinander, nämlich das Öffnen der Kühlschranktür vom Einschenkvorgang. Letzerer wird wiederum in die Greif-, Einschenk- und Abstellphase unterteilt.

Die Funktionale Beitragsanalyse (Abschnitt 4.10) stellt den letzten Teil der syntaktischen Analysephase dar und ebnet den Weg in die semantische Analyse. Sie analysiert die Auswirkungen, welche die Benutzerhandlung auf die Umwelt hat. Die Ergebnisse sind in Tabelle 7.1 dargestellt. Die Teilhandlung des Öffnens des Kühlschranks ändert den Zustand der Kühlschranktür. Dies ist durch die Relation open (kühlschrank-tuer) dargestellt. Der anschließende Teil des Einschenkens aus der Flasche in die Tasse verändert

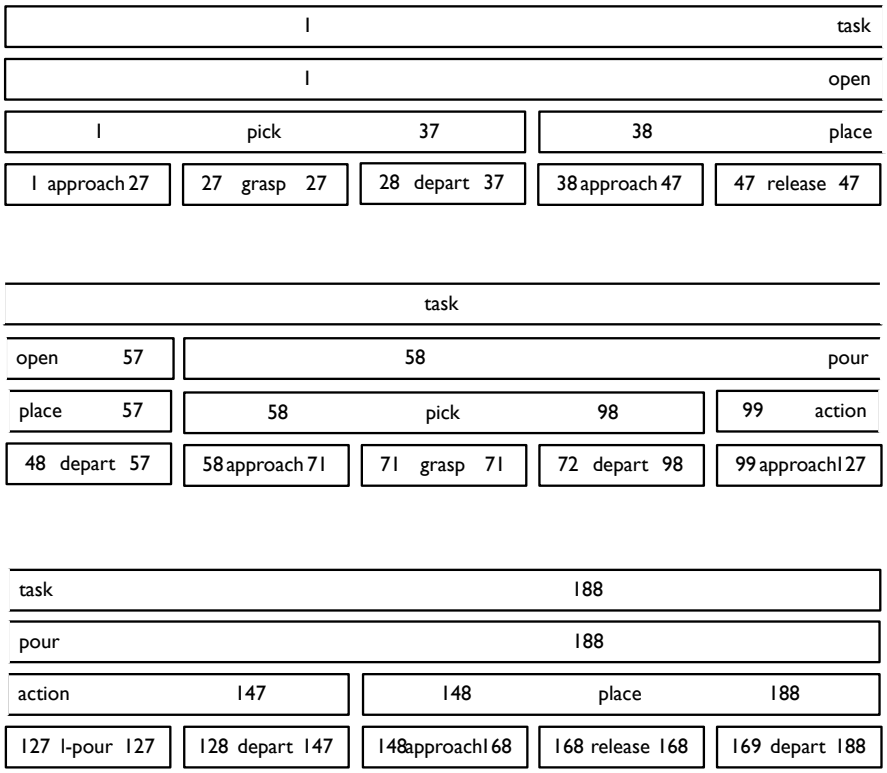


Abbildung 7.6: Syntaktische Struktur der Einschenken-Aufgabe

den Zustand der Tasse auf filled (kaffee-tasse fantabottle) als direktes Resultat der eigentlichen Fülloperation, sowie die geometrischen Relationen, welche zwischen der Flasche sowie den Objekten Tasse sowie dem silberne Tischset existieren. Hierbei ist festzustellen, daß die Flasche nun nicht mehr östlich, sondern westlich von der Tasse steht (2., 3., 9. und 10. Relation des Einschenken-Makros), sowie auf dem silbernen Tischset (7. Relation).

Der Gesamtbeitrag des kompletten Makro-Operators besteht nun, wie man sieht, genau aus der Vereinigung der Beiträge beider feingranulareren Makro-Operatoren. Dies ist unmittelbar einsichtig, da keinerlei Effekte der einzelnen Makro-Operatoren durch spätere Operationen wieder rückgängig gemacht wurden.

Weiterhin ist zu bemerken, daß die Menge an Relationen noch relativ umfangreich ist. Ein Benutzer würde erwarten, daß lediglich die Relation on (fantabottle set-silber) in der finalen Menge an Beiträgen enthalten ist, neben den Tatsachen der Änderungen des Füllstandes und dem Öffnen des Kühlschranks. Dieser Mißstand wird in der folgenden semantischen Analy-

Makro	Relation	Wahrheitswert
open 1 - 57	open (kuehlschrank-tuer)	true
pour 58 - 188	filled (kaffee-tasse fantabottle)	true
	east (fantabottle kaffee-tasse)	false
	west (fantabottle kaffee-tasse)	true
	higher (fantabottle set-silver)	false
	south (fantabottle set-silver)	false
	east (fantabottle set-silver)	false
	on (fantabottle set-silver)	true
	near (fantabottle set-silver)	true
	east (kaffee-tasse fantabottle)	true
	west (kaffee-tasse fantabottle)	false
	lower (set-silver fantabottle)	false
	north (set-silver fantabottle)	false
	west (set-silver fantabottle)	false
	near (set-silver fantabottle)	true
task 1 - 188	open (kuehlschrank-tuer)	true
	filled (kaffee-tasse fantabottle)	true
	east (fantabottle kaffee-tasse)	false
	west (fantabottle kaffee-tasse)	true
	higher (fantabottle set-silver)	false
	south (fantabottle set-silver)	false
	east (fantabottle set-silver)	false
	on (fantabottle set-silver)	true
	near (fantabottle set-silver)	true
	east (kaffee-tasse fantabottle)	true
	west (kaffee-tasse fantabottle)	false
	lower (set-silver fantabottle)	false
	north (set-silver fantabottle)	false
	west (set-silver fantabottle)	false
	near (set-silver fantabottle)	true

Tabelle 7.1: Beiträge der einzelnen Teilhandlungen und der Gesamthandlung

sephase angegangen, indem die Relationen mit einem Relevanzmaß versehen werden, indem über mehrere Demonstrationen derselben Aufgabe hinweg die Invarianzen und Variablen ermittelt werden.

7.2 Evaluation der semantischen Analyse

Der vorangegangene Abschnitt zeigt, wie einzelne Benutzerdemonstrationen sensorisch erfasst werden können, wie daraus elementare Operationen erkannt werden können sowie wie daraus ein hierarchisch-funktionales Programm erstellt werden kann. Um ein solches Programm allerdings interpretieren zu können muß das lernende System die einzelne Demonstration in seinen Erfahrungsschatz einordnen können. Hierzu wurden in Kapitel 5 Verfahren entwickelt, die im folgenden detailliert evaluiert werden sollen.

Das System wurde im hier beschriebenen Experiment mit 10 verschiedenen Benutzervorführungen getestet, welche aus drei verschiedenen Aufgaben bestanden:

- Zwei Vorführungen der in Abschnitt 7.1.2 bereits behandelten Aufgabe des Einschenkens aus einer Flasche.

- Vier unterschiedlichen Vorführungen der Aufgabe des Tischdeckens mit drei verschiedenen Objekten. Die Aufgabe bestand darin, ein Tischset zu decken unter Benutzung eines Tellers, einer Untertasse und einer Tasse. Die Tasse sollte dabei am Ende auf der Untertasse und rechts neben dem Teller positioniert werden. Die Vorführungen wurden mit unterschiedlicher Reihenfolge vorgenommen (näheres dazu in Abschnitt 7.3.1), jedoch in einer Art und Weise, daß am Ende immer dieselbe Objektkonfiguration erzeugt wurde.

- Ebenfalls vier verschiedene Vorführungen für eine komplexere Aufgabe zum Tischdecken mit sechs unterschiedlichen Objekten. Hierbei sollten zwei Teller übereinander, ein Tasse wieder auf einer Untertasse, sowie zwei Löffel auf eine bestimmte Art und Weise angeordnet werden. Nähere Beschreibungen finden sich in Abschnitt 7.3.

Die resultierende Demonstrationsmenge $\mathbf{D}$ hat also den Umfang $n = 10$ und die Makro-Operatoren sind in der Reihenfolge angeordnet, wie sie von der obigen Aufzählung induziert wird, d.h. D_1 und D_2 sind zwei Vorführungen des Einschenkvorgangs, D_3 bis D_6 und D_7 bis D_{10} enthalten die Demonstrationen zum Tischdecken mit drei bzw. sechs Objekten.

Mit den bereits in den vorherigen Abschnitten evaluierten Verfahren wurden aus den Benutzerdemonstrationen Makro-Operatoren erstellt. Auf diesen Makro-Operatoren wurde anschließend eine Ähnlichkeits- und Ballungsanalyse durchgeführt, wie sie in Abschnitt 5.4 beschrieben wurde. Dabei wurde auf das Ähnlichkeitsmaß aus Abschnitt 5.2 zurückgegriffen.

Zunächst wurde dafür die symmetrische Ähnlichkeitsmatrix R aufgestellt, welche in Tabelle 7.2 aufgetragen ist. Diese ist folgendermaßen zu interpretieren: jedes Element $r_{i,j}$ von R entspricht dem Ähnlichkeitsmaß $sim(D_i, D_j)$ gemäß der Definition 5.2 aus Abschnitt 5.2. Dementsprechend sind sämtliche Diagonalelemente $r_{i,i}$ gleich eins, da dies der Selbstähnlichkeit jeden einzelnen Makro-Operators entspricht. Die Nichdiagonalelemente nehmen Werte zwischen 0 und 1 ein, entsprechend der Ähnlichkeit der beiden verglichenen Makro-Operatoren. Das geschulte Auge erkennt hier bereits Bereiche mit hoher Ähnlichkeit, die genau den Aufgabenklassen des Einschenkens und des Tischdeckens mit drei bzw. sechs Objekten entsprechen. Diese Bereiche wurden der Deutlichkeit halber farbig unterlegt. Der rote Bereich entspricht der Aufgabe des Einschenkens, der grüne dem Tischdecken mit drei, der blaue dem Tischdecken mit sechs Objekten.

$$R = \begin{bmatrix}
1.00 & 0.99 & 0.47 & 0.41 & 0.44 & 0.50 & 0.14 & 0.16 & 0.15 & 0.18 \\
0.99 & 1.00 & 0.47 & 0.41 & 0.44 & 0.50 & 0.14 & 0.16 & 0.15 & 0.18 \\
0.47 & 0.47 & 1.00 & 0.84 & 0.66 & 0.72 & 0.55 & 0.52 & 0.55 & 0.57 \\
0.41 & 0.41 & 0.84 & 1.00 & 0.75 & 0.75 & 0.60 & 0.61 & 0.58 & 0.59 \\
0.44 & 0.44 & 0.66 & 0.75 & 1.00 & 0.78 & 0.53 & 0.57 & 0.52 & 0.57 \\
0.50 & 0.50 & 0.72 & 0.75 & 0.78 & 1.00 & 0.52 & 0.46 & 0.45 & 0.47 \\
0.14 & 0.14 & 0.55 & 0.60 & 0.53 & 0.52 & 1.00 & 0.79 & 0.83 & 0.82 \\
0.16 & 0.16 & 0.52 & 0.61 & 0.57 & 0.46 & 0.79 & 1.00 & 0.82 & 0.81 \\
0.15 & 0.15 & 0.55 & 0.58 & 0.52 & 0.45 & 0.83 & 0.82 & 1.00 & 0.86 \\
0.18 & 0.18 & 0.57 & 0.59 & 0.57 & 0.47 & 0.82 & 0.81 & 0.86 & 1.00
\end{bmatrix}$$

Tabelle 7.2: Ähnlichkeitsmatrix für 10 Aufgabendemonstrationen. $r_{i,j}$ entspricht dem Ähnlichkeitswert zweier Demonstrationen $sim(D_i, D_j)$.

Das System konnte erfolgreich die drei Aufgabenklassen identifizieren und die einzelnen Demonstrationen diesen Clustern zuordnen. Die Zugehörigkeitswerte zu den einzelnen Klassen sind in Tabelle 7.3 mit ihren numerischen Werten dargestellt und in Abbildung 7.7 visualisiert. Auffällig hierbei ist, daß das System bei den Demonstrationen D_3 bis D_6, also dem Tischdecken mit drei Objekten, eine gewisse Unsicherheit aufweist, die sich in Werten von bis zu 0.61 bei der Zugehörigkeit zur Klasse des Tischdeckens mit sechs Objekten äußert. In Graphik 7.7 ist dies anhand des nur relativ geringen Abstands der blauen Linie zur grünen sichtbar, die genau die beiden Tischdeckaufgaben mit sechs bzw. drei Objekten repräsentieren. Der Grund hierfür liegt darin, daß sich die beiden Aufgaben hinsichtlich ihrer Zielkonfiguration sehr stark ähneln. Inwiefern eine solche Unsicherheit vom System zu bewer-

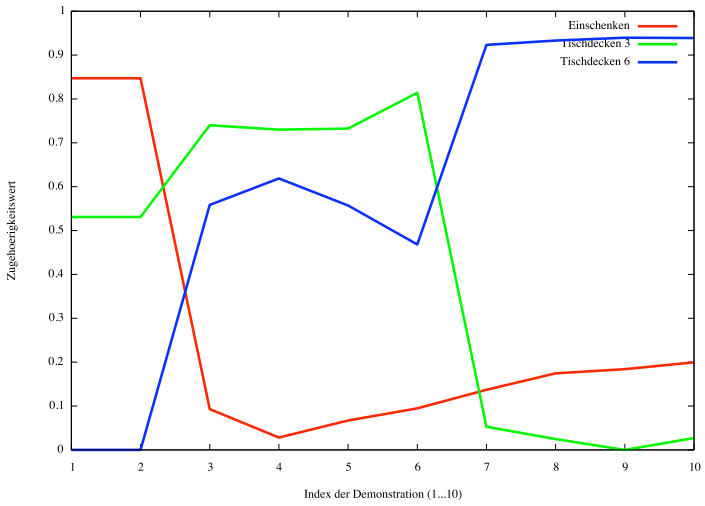


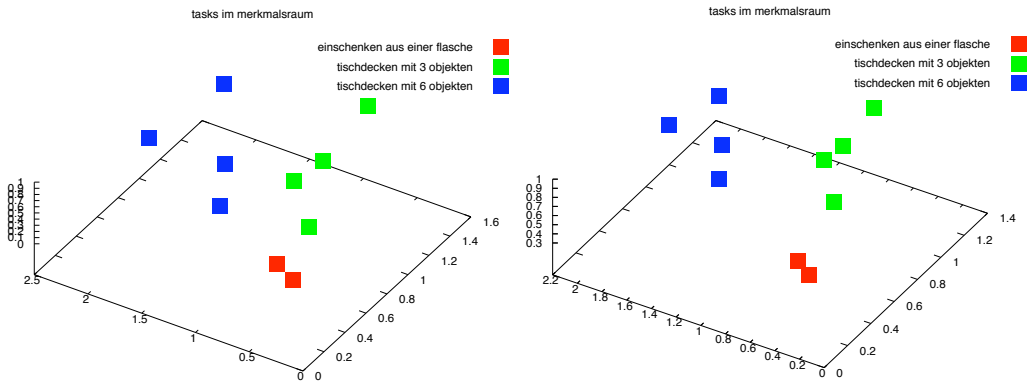
Abbildung 7.7: Visualisierung der Klassenzugehörigkeit. Für jede Demonstration D_i ist der Zugehörigkeitswert zu jedem der drei Cluster aufgetragen.

ten ist und wie es damit umgehen sollte, ist weiterer Forschungsbedarf. Im hier gewählten Ansatz ist jedoch nur der maximale Klassenzugehörgkeitswert für die Ballungsanalyse ausschlaggebend.

D	T6	T3	E
D_1	0.00	0.53	**0.84**
D_2	0.00	0.53	**0.84**
D_3	0.55	**0.74**	0.09
D_4	0.61	**0.73**	0.02
D_5	0.55	**0.73**	0.06
D_6	0.46	**0.81**	0.09
D_7	**0.92**	0.05	0.13
D_8	**0.93**	0.02	0.17
D_9	**0.93**	0.00	0.18
D_{10}	**0.93**	0.02	0.19

Tabelle 7.3: Klassenzugehörigkeit der einzelnen Demonstrationen zu den identifizierten Aufgabenklassen des Tischdeckens mit sechs (T6), bzw. drei (T3) Objekten und dem Einschenken (E).

Nachdem die Cluster bestimmt und die einzelnen Demonstrationen diesen zugeordnet sind, können die klassenspezifischen Ähnlichkeitsmaße bestimmt werden, die die Merkmalsverteilung innerhalb jeder einzelnen Klasse berücksichtigen. Dies führt dazu, daß die einzelnen Demonstrationen innerhalb der



(a) (b)

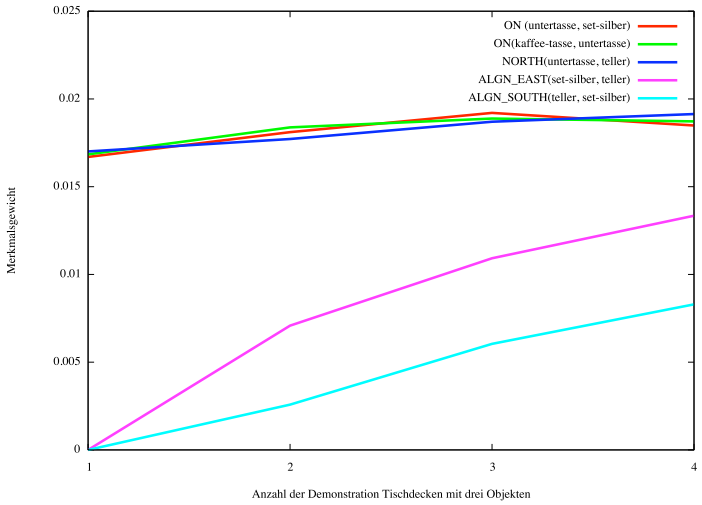
Abbildung 7.8: Veränderung der Ähnlichkeitswerte unter Berücksichtigung der klassenimmanenten Merkmalsverteilung und Relevanzschätzung der einzelnen Merkmale. (a) Tasks im Merkmalsraum ohne Berücksichtigung der Merkmalsverteilung. (b) Tasks im Merkmalsraum unter Berücksichtigung der klassenindividuellen Merkmalsverteilung. Man erkennt, daß die Cluster enger zusammenrücken.

Cluster näher zusammenrücken, d.h. der Umfang der Cluster reduziert wird.

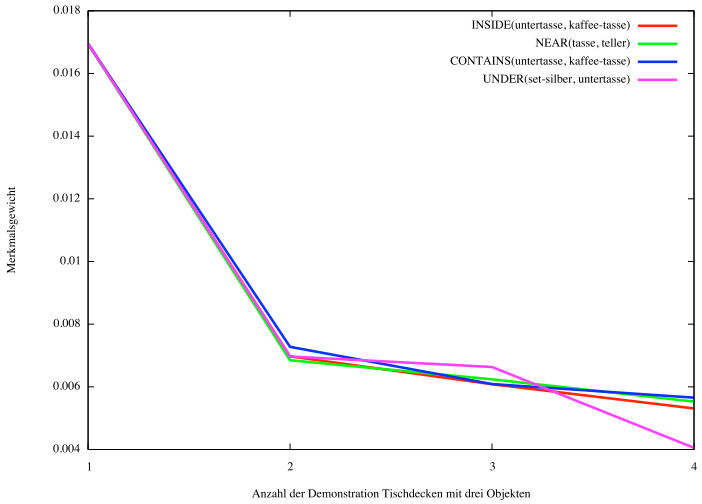
$$R_{T3} = \begin{pmatrix} 1.00 & 0.90 & 0.77 & 0.81 \\ 0.90 & 1.00 & 0.83 & 0.83 \\ 0.77 & 0.83 & 1.00 & 0.84 \\ 0.81 & 0.83 & 0.84 & 1.00 \end{pmatrix} \quad R_{T6} = \begin{pmatrix} 1.00 & 0.85 & 0.85 & 0.84 \\ 0.85 & 1.00 & 0.86 & 0.88 \\ 0.85 & 0.86 & 1.00 & 0.91 \\ 0.84 & 0.88 & 0.91 & 1.00 \end{pmatrix}$$

Tabelle 7.4: Veränderung der Ähnlichkeitswerte unter Berücksichtigung der klassenimmanenten Merkmalsverteilung und Relevanzschätzung der einzelnen Merkmale. Der Vergleich mit Tabelle 7.2 auf Seite 160 ergibt, daß die klassenspezifische Ähnlichkeit steigt.

Dieser Effekt ist in Abbildung 7.8 sichtbar. Abbildung 7.8 (a) stellt die aufgenommenen Tasks im Merkmalsraum dar, ohne daß die Merkmalsverteilung berücksichtigt worden wäre. Wurde die klassenspezifische Merkmalsfunktion $sim_w(x, y)$ verwendet, wie in Abbildung 7.8 (b), so steigen die Ähnlichkeitswerte innerhalb jeder einzelnen Klasse. Das führt dazu, daß die Demonstrationen, die einen Cluster ausmachen, näher zusammenrücken. Die numerischen Werte für die Ähnlichkeit sind in Tabelle 7.4 aufgetragen. Der Vergleich mit den grün bzw. blau markierten Bereichen in Tabelle 7.2 liefert



(a)



(b)

Abbildung 7.9: Relevanzbewertung für Merkmale der Aufgabe des Tischdeckens mit drei Objekten. (a) Als relevant erkannte Merkmale. (b) Irrelevante Merkmale.

eine deutlich erhöhte klasseninterne Ähnlichkeit.

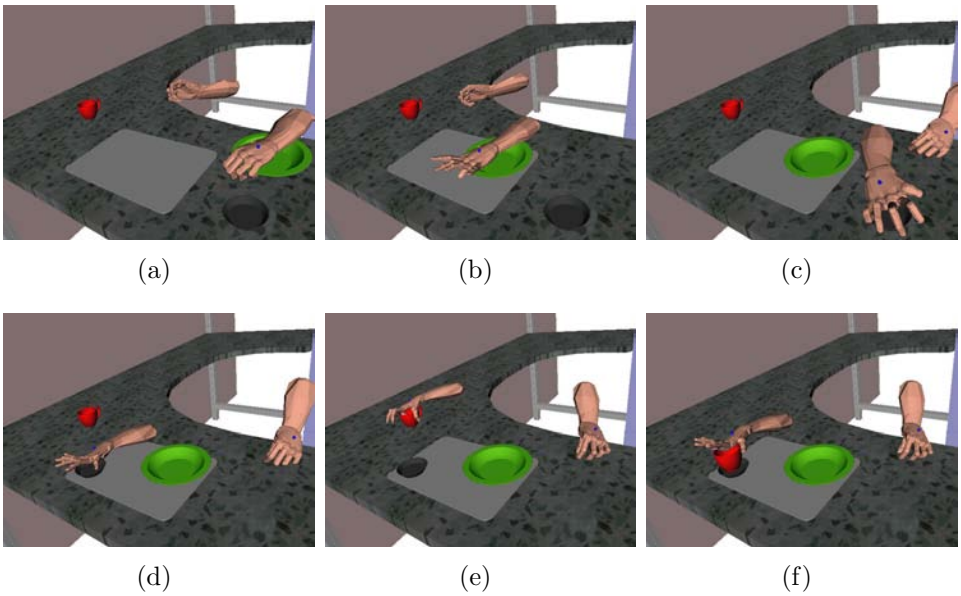
Stellt man sich die Frage, wie diese erhöhte Ähnlichkeit zustandekommt, so wird man in Abschnitt 5.1 fündig. Das Ähnlichkeitsmaß ist dementsprechend aus den Ähnlichkeiten für einzelne Merkmale zusammengesetzt. Diese sind jedoch entsprechend einem inkrementell mit jeder zusätzlichen Demonstration angepassten Gewicht versehen. Jedes einzelne Merkmal geht entspre-

chend seinem Gewicht in die Gesamtähnlichkeit ein. Merkmale mit einer hohen Auftretenswahrscheinlichkeit haben dabei größeren Einfluß auf das klassenspezifische Ähnlichkeitsmaß $sim_w(x, y)$ über den Gewichtungsvektor w.

Die Entwicklung der Gewichte für unterschiedliche Relationen findet man in Abbildung 7.9. Verschiedene, vom System als relevant erachtete Merkmale finden sich in Abbildung 7.9 (a). Darunter fallen die Beschreibungen folgender Relationen: Die Untertasse befindet sich auf dem silbernen Platzset (ON (untertasse, set-silber)), die Kaffeetasse befindet sich auf der Untertasse (ON (tasse, untertasse)), die Untertasse ist nördlich vom Teller (NORTH(untertasse, teller)). Weitere als fast genauso relevant erachtete Relationen sind die Tatsache, daß der Teller sowohl an der Süd- als auch an der Ostkante des silbernen Platzsets ausgerichtet wurden (ALGN_SOUTH(set-silber, teller), ALGN_EAST(teller, set-silber)). Diese Merkmale waren beide durch die erste Demonstration nicht erfüllt und wurden daher fälschlicherweise vom System mit einer niedrigen Relevanz bewertet. Diesen Fehler konnte das System aber unter Zuhilfenahme zusätzlicher Demonstrationen mit der Zeit beheben. Hier kommt eine der wesentlichen Stärken des inkrementellen Lernens zum Tragen, nämlich daß einmal gemachte Hypothesen (zu denen die Schätzung der Relevanz eines Merkmals gehört) durchaus verworfen oder verbessert werden können, sobald zusätzliche Informationen verfügbar werden.

Ein ähnliches Verhalten ist bei der Aussortierung irrelevanter Merkmale erkennbar (siehe Abbildung 7.9(b)). Es waren in der ersten Demonstration noch Kollisionen zwischen der Untertasse und der Kaffeetasse vorhanden. Dies führte zu einem Vorhandensein der Merkmale INSIDE(untertasse, tasse) sowie CONTAINS(untertasse, tasse), die aufgrund der Nichtverfügbarkeit weiterer Informationen mit ähnlich hoher Relevanz bewertet wurden, wie die anderen Merkmale. Kamen jedoch mit weiteren Demonstrationen zusätzliche Informationen hinzu, so konnte die Merkmalsgewichtung angepasst werden, so daß die Merkmale zu einem viel geringeren Teil in die Berechnung der Ähnlichkeitsfunktion eingehen.

In diesem Abschnitt wurde gezeigt, wie das System zu inkrementellen Lernen von Handlungswissen autonom und unüberwacht Klassen von Handlungswissen in seinem Erfahrungsschatz findet, sowie wie die Charakteristika innerhalb dieser Klassen dazu benutzt werden, den Merkmalsraum zu entzerren, so daß die durchschnittliche Ähnlichkeit innerhalb einer Klasse mit steigender Anzahl an Beispielen zunimmt. Klassen enthalten jedoch noch mehr Informationen, als sie lediglich durch Anpassung der Relevanzwerte erhalten werden können. Der kommende Abschnitt befasst sich deshalb mit der Erschließung neuen Handlungswissens innerhalb der Handlungsklassen mittels Reasoning-Techniken.



(a) (b) (c)

(d) (e) (f)

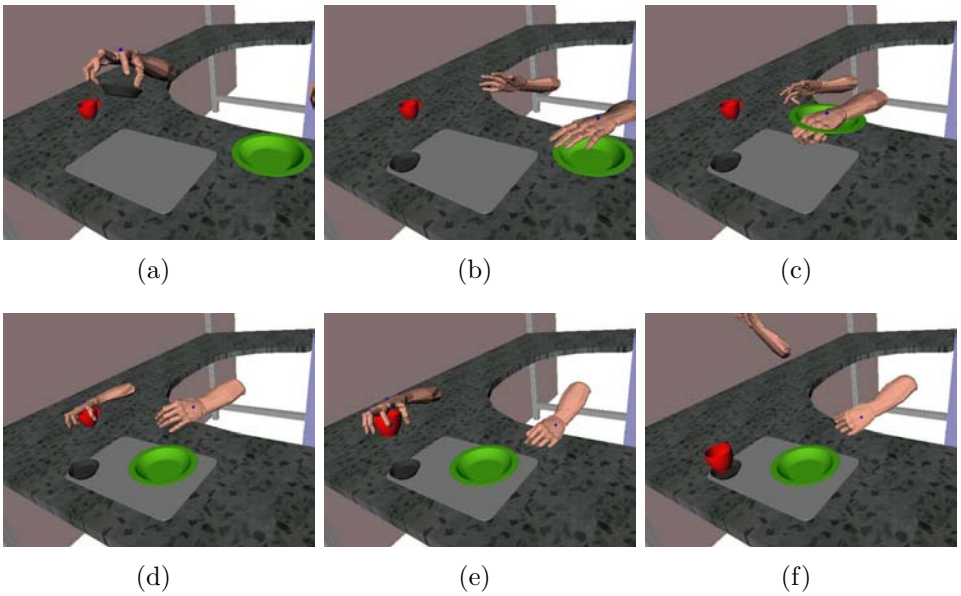
Abbildung 7.10: Tischdecken in der Reihenfolge Teller, Untertasse, Tasse. Die Schlüsselszenen der Sequenz befinden sich an den Indizes 24, 86, 127, 177, 209, 286.

7.3 Evaluation der Reasoning-Phase

Im vorangegangenen Abschnitt wurde vorgestellt, wie das im Rahmen dieser Arbeit erstellte System zum inkrementellen und interaktiven Lernen seinen Erfahrungsschatz autonom in unterschiedliche disjunkte Teilmengen von gelernten Aufgaben einteilt und wie klassenspezifische Relevanzschätzungen für die einzelnen Demonstrationen erstellt werden können. Das Ergebnis dieses Prozesses ist eine flache Taxonomie von Handlungen sowie Gewichtungsparameter für einzelne Merkmale. Im Folgenden wird gezeigt, wie die gewonnenen Klasseninformationen ausgenutzt werden können, um klassenspezifisches zusätzliches Handlungswissen aus dem so strukturierten Erfahrungsschatz zu gewinnen.

Zuerst ist es jedoch notwendig, die im vorangegangenen Abschnitt vorgestellten Daten genauer zu betrachten. Insbesondere die beiden Tischdeckaufgaben können auf sehr unterschiedliche Art und Weise demonstriert werden.

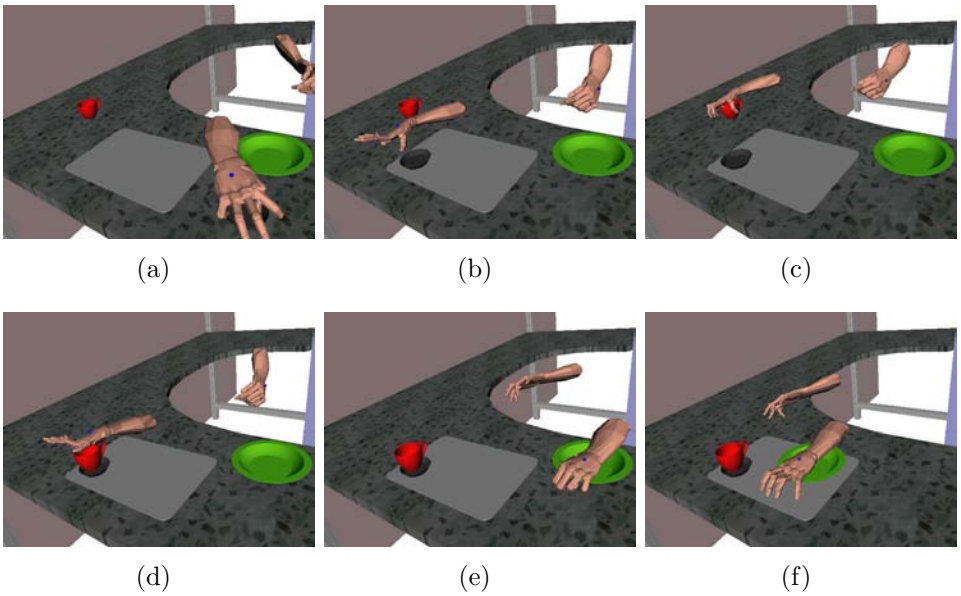
In der Klasse der Aufgabe zum Tischdecken mit drei Objekten wurde die erste Demonstration in der Folge gegeben, wie sie in Abbildung 7.10 dargestellt ist: Die Versuchsperson stellt zuerst den Teller an seine Zielposition, anschließend die Untertasse und zuletzt die Tasse. Die zweite der vier Demonstrationen erfolgte in derselben Reihenfolge. In der dritten Demons-



(a) (b) (c)

(d) (e) (f)

Abbildung 7.11: Tischdecken in der Reihenfolge Untertasse, Teller, Tasse. Die Schlüsselszenen der Sequenz befinden sich an den Indizes 41, 86, 111, 158, 176, 218.



(a) (b) (c)

(d) (e) (f)

Abbildung 7.12: Tischdecken in der Reihenfolge Untertasse, Tasse, Teller. Die Schlüsselszenen der Sequenz befinden sich an den Indizes 47, 95, 121 173, 202, 243.

tration kam etwas Varianz in dieses Muster: Die Untertasse wurde vor dem Teller platziert, abschließend gefolgt von der Tasse (siehe Abbildung 7.11) . In der letzten Demonstration dieser Taskklasse kam die Reihenfolge Untertasse, Tasse und Teller zum Einsatz (siehe Abbildung 7.12). Diese unterschiedlichen Ausführungssequenzen bedingen Unabhängigkeiten in den einzuhaltenden Reihenfolgebeziehungen. Um diese analysieren zu können, ist es zuerst nötig, die Operatorpermutationen zu bestimmen.

7.3.1 Bestimmung der Operatorpermutation

Das Verfahren zum Berechnen der optimalen Operatorpermutation wurde in Abschnitt 6.1 theoretisch fundiert und vorgestellt. Es geht aus von der Matrix der Suboperatoren-Korrespondenz $M_{i,j}$ für zwei Demonstrationen D_1, D_2 mit ihren nachgeordneten Operatoren $O_{1,i}, O_{2,j}$ für $0 < i, j \leq n$. Im Fall der Aufgabe des Tischdeckens mit drei Objekten ist $n = 3$. Die Korrespondenzmatrix M für die Korrespondenz zwischen der ersten und der dritten, bzw. der ersten und der vierten Demonstration der Aufgabe ist in Tabelle 7.5 aufgetragen. Im Fall für M_{D_1,D_3} erkennt man leicht die Vertauschung der ersten beiden Operationen, das große Diagonalelement für die dritte Operation entspricht keiner Reihenfolgeveränderung für die Manipulation der Tasse. Im Fall für M_{D_1,D_4} ist die erste Operation (Platzieren des Tellers) hinter die beiden anderen Operationen getauscht worden.

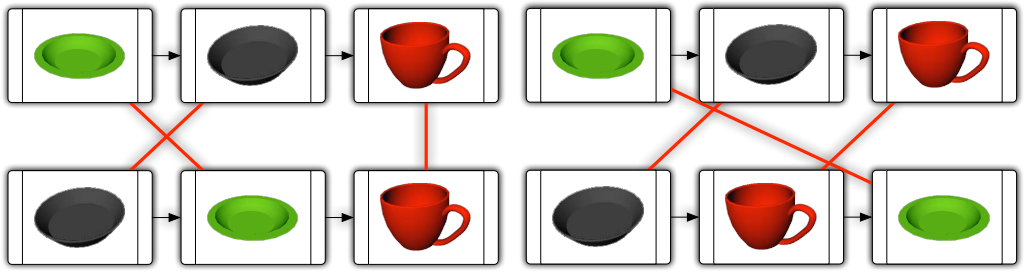
Abbildung 7.13 zeigt die aus den Suboperator-Korrespondenzmatrizen berechneten Operatorpermutationen. Es wird in beiden Fällen (für die dritte sowie die vierte Demonstration in Referenz zur ersten) der jeweils korrekte Teiloperator der Einen dem entsprechenden Operator der Referenzsequenz zugeordnet. Das System ist also in der Lage, die erste Operation von Demonstration 1 der zweiten Operation von Demonstration 3 und umgekehrt zuzuordnen, sowie zu erkennen, dass in beiden Demonstrationen die dritte Operation den Platz nicht gewechselt hat (Abbildung 7.13 (a)). Genauso ist in Demonstration 4 die letzte Operation der Referenzdemonstration vor die beiden anderen gezogen worden, was sich im Überkreuzen der roten Zuordnungen in Abbildung 7.13 äußert.

Für den komplexeren Fall des Tischdeckens mit sechs Objekten wurde die in Tabelle 7.6 dargestellte Suboperator-Korrespondenzmatrix berechnet. Das daraus resultierende Ergebnis in der Operatorpermutation ist in Abbildung 7.14 dargestellt.

Wenn die Operatorpermutationen bestimmt sind, kann das System die Permutationen nutzen, um sein Wissen über Vorrangbeziehungen zwischen den einzelnen Operationen zu vervollständigen. Das geschieht mit Verfahren, die im folgenden Abschnitt vorgestellt werden.

$$M_{D_1,D_3} = \begin{bmatrix} 0.451 & \mathbf{0.982} & 0.342 \\ \mathbf{1.000} & 0.434 & 0.427 \\ 0.353 & 0.251 & \mathbf{0.926} \end{bmatrix} \quad M_{D_1,D_4} = \begin{bmatrix} 0.451 & 0.365 & \mathbf{0.972} \\ \mathbf{0.999} & 0.450 & 0.455 \\ 0.353 & \mathbf{0.888} & 0.272 \end{bmatrix}$$

Tabelle 7.5: Suboperatoren-Korrespondenz für die Demonstrationen der Aufgabe des Tischdeckens mit drei Objekten. Die optimalen Zuordnungen von Operatoren sind in fetter Schrift markiert.M_{D_1,D_3} enthält die Ähnlichkeitsmatrix für die erste und die dritte Demonstration, M_{D_1,D_4} die für die erste und vierte Demonstration.



(a) (b)

Abbildung 7.13: Suboperator-Permutation für die Demonstrationen der Aufgabe des Tischdeckens mit drei Objekten. Die einzelnen Suboperatoren sind durch die Objekte repräsentiert, die mit ihnen manipuliert werden. (a) Operatorpermutation zwischen der ersten und der dritten Demonstration. (b) Operatorpermutation zwischen der ersten und der vierten Demonstration.

$$M_{D_1,D_3} = \begin{bmatrix} 0.740 & 0.709 & \mathbf{0.965} & 0.666 & 0.675 & 0.677 \\ 0.704 & 0.697 & 0.625 & 0.654 & \mathbf{0.958} & 0.641 \\ 0.712 & 0.706 & 0.633 & 0.687 & 0.647 & \mathbf{0.922} \\ 0.756 & \mathbf{0.897} & 0.701 & 0.657 & 0.666 & 0.719 \\ 0.756 & 0.701 & 0.653 & \mathbf{0.941} & 0.667 & 0.674 \\ \mathbf{0.951} & 0.769 & 0.717 & 0.718 & 0.727 & 0.737 \end{bmatrix}$$

Tabelle 7.6: Suboperatoren-Korrespondenz für die Demonstrationen der Aufgabe des Tischdeckens mit sechs Objekten. Die optimalen Zuordnungen von Operatoren sind in fetter Schrift markiert.

7.3.2 Evaluation des Lernens von Päzedenzgraphen

Basierend auf den Operatorpermutationen des vorangegangenen Abschnitts werden nun die Ergebnisse des Lernens von Taskpräzedenzgraphen aus Ab-

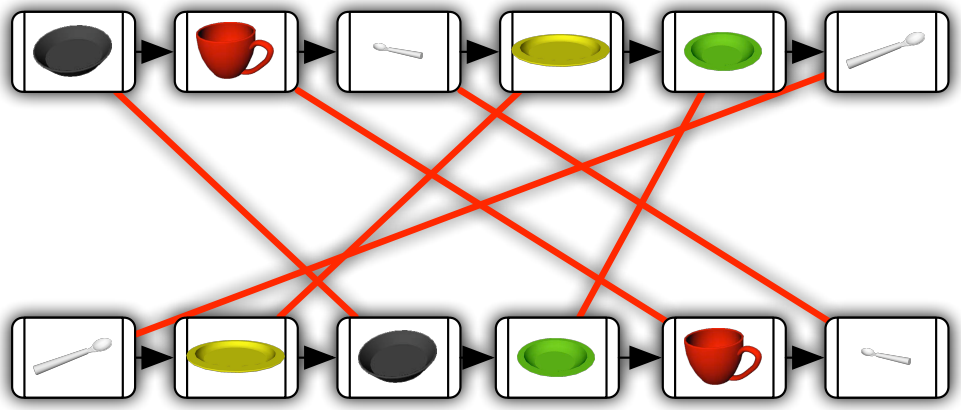


Abbildung 7.14: Suboperator-Permutation für die Demonstrationen der Aufgabe des Tischdeckens mit sechs Objekten. Die einzelnen Suboperatoren sind durch die Objekte repräsentiert, die mit ihnen manipuliert werden.

schnitt 6.2 vorgestellt.

Aus den einzelnen Demonstrationen D_i der Tischdeckaufgabe werden die restriktivsten Präzedenzgraphen P^{D_i} für jede einzelne Demonstration erstellt. Die darin enthaltenen Kantenmengen $\mathbf{E}^{D_i}$ sind in Tabelle 7.7 dargestellt. Der restriktivste Präzedenzgraph (mit der Kantenmenge $\mathbf{E}^{D_1}$) für die erste Demonstration enthält die Aussage, daß der Teller vor beiden anderen Objekten platziert werden muss, sowie die Untertasse vor der Tasse (drei Präzedenzrelationen). Entsprechende Aussagen gelten für die dritte Demonstration: Die Untertasse muss vor dem Teller und der Tasse platziert werden, der Teller nur vor der Tasse. Bei der Integration dieser Aussagen in das Gesamtwissen des Systems treten, wie in Abschnitt 6.2 geschildert, Inkonsistenzen auf. So widerspricht die Forderung der ersten Demonstration, daß der Teller vor der Untertasse manipuliert werden muss, der Aussage, dass die Untertasse vor dem Teller in der dritten Demonstration platziert wurde. Dieser Widerspruch wird durch Elimination beider Postulate erreicht. Die resultierende Menge an Vorrangbeziehungen enthält nur noch die Aussagen, daß sowohl die Untertasse wie der Teller vor der Tasse behandelt werden, ohne zwischen diesen beiden eine Vorrangbeziehung festzulegen. Der daraus resultierende Präzedenzgraph ist in Tabelle 7.8 graphisch dargestellt.

Sammelt das System noch zusätzlich die Erfahrung, daß der Teller auch als letztes Objekt platziert werden kann, so gehen in die Berechnung der Taskpräzedenzgraphen die Präzedenzbeziehungen Untertasse vor Tasse und Teller neben der Tatsache, daß die Tasse auch vor dem Teller platziert werden kann, ein. Das resultiert in der Streichung der Präzedenzbeziehung zwischen

i	$\mathbf{E}^{D_i}$	$\mathbf{E}_i$
1	$o_1 \rightarrow_G o_2$ $o_2 \rightarrow_G o_3$ $o_1 \rightarrow_G o_3$	$o_1 \rightarrow_G o_2$ $o_2 \rightarrow_G o_3$ $o_1 \rightarrow_G o_3$
3	$o_2 \rightarrow_G o_1$ $o_1 \rightarrow_G o_3$ $o_2 \rightarrow_G o_3$	$o_2 \rightarrow_G o_3$ $o_1 \rightarrow_G o_3$
4	$o_2 \rightarrow_G o_3$ $o_3 \rightarrow_G o_1$ $o_2 \rightarrow_G o_1$	$o_2 \rightarrow_G o_3$

Tabelle 7.7: Präzedenzrelationen im restriktivsten Präzedenzgraph für jede Demonstration sowie gelernter Gesamtpräzedenzgraph am Beispiel des Tischdeckens mit drei Objekten.

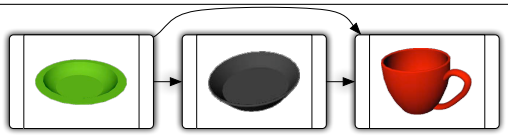
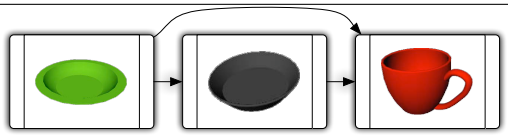
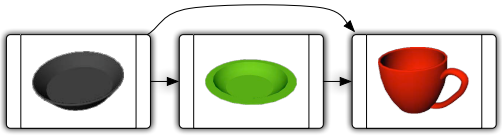
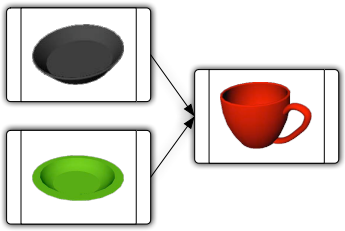
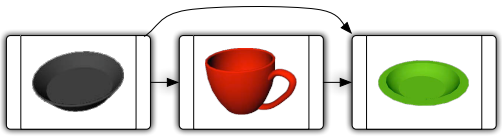
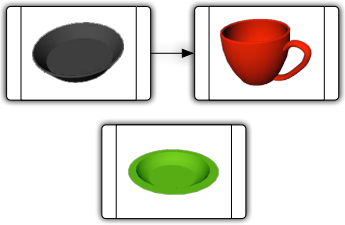
i	$\mathbf{E}^{D_i}$	$\mathbf{E}_i$
1		
3		
4		

Tabelle 7.8: Inkrementelles Lernen von Taskpräzedenzgraphen anhand des Beispiels des Tischdeckens mit drei Objekten

der Manipulation des Tellers und der Tasse, so daß nur noch eine Reihenfolgebeziehung zwischen der Platzierung der Untertasse und der Platzierung der Tasse in $\mathbf{E}_4$ übrig bleibt (siehe Tabelle 7.7 und Abbildung 7.8).

Anhand dieses Beispiels werden die Stärken und Charakteristika des in-

krementellen Lernens gut sichtbar: Das System startet mit einer recht konservativen Schätzung der Reihenfolgebeziehungen zwischen den einzelnen Teilaufgaben, nämlich mit der Annahme, daß alle demonstrierten Reihenfolgebeziehungen relevant und zur fehlerfreien Ausführung einzuhalten sind. Dennoch ist mit dieser Schätzung eine Ausführung der gelernten Aufgabe möglich. Das System ist also bereits nach der ersten Demonstration in der Lage, die Aufgabe erfolgreich zu meistern, verwendet jedoch unter Umständen eine suboptimale Reihenfolge.

Dies ändert sich auch nicht mit der zweiten Demonstration. Diese führte dem System noch keine zusätzlichen Informationen in Bezug auf die Reihenfolge der Operatoren zu. Mit der dritten Demonstration derselben Aufgabe wird das System jedoch mit einer Inkonsistenz innerhalb seiner Wissensbasis konfrontiert. Daraus resultiert nach dem inkrementellen Lernschritt die Freiheit, die Reihenfolge zwischen der Platzierung des Tellers und der Untertasse zu bestimmen - solange beide vor der Tasse ausgeführt werden. Das System gewinnt so einen Freiheitsgrad, der für Planungsschritte in der Ausführungsphase genutzt werden kann, um das gelernte Handlungswissen in optimierter und flexiblerer Art und Weise zur Ausführung zu bringen. Mit der letzten Demonstration kommt noch ein weiterer Lernfortschritt hinzu. Dieses Lernen ist demnach ein kontinuierlicher, nicht endender Prozeß.

Das Beispiel des Tischdeckens mit drei Objekten diente der einfachen und übersichtlichen Darstellung der Funktionsweise des inkrementellen Lernens von Präzedenzgraphen. Im folgenden soll gezeigt werden, daß das implementierte Verfahren skaliert, und zwar an dem Beispiel des Tischdeckens mit sechs Objekten: Einer Untertasse, auf die eine Tasse zu platzieren ist, in der ein Löffel steckt. Auf einen großen, flachen Teller ist ein Suppenteller zu stellen, sowie rechts daneben ein Suppenlöeffel. Weiterhin ist die Objektverteilung initial davon bestimmt, daß die Untertasse in dem Suppenteller ist, letzterer also nicht bewegt werden kann, bevor die Untertasse platziert wurde. Die Start- und Zielkonfiguration dieser Aufgabe ist in Abbildung 7.15

Die ersten beiden Demonstrationen sind in der Reihenfolge gegeben worden, wie sie in Abbildung 7.16 (a) dargestellt ist: Zuerst wird die Untertasse bewegt, dann die Tasse darauf gestellt sowie der Löffel in die Tasse gesteckt. Nach der Platzierung des flachen Tellers kann der mittlerweile freie Suppenteller darauf gestellt werden und zuletzt der Suppenlöffel rechts daneben. Die daraus resultierenden $\frac{n\cdot(n-1)}{2} = 15$ Vorrangbeziehungen des restriktivsten resultierenden Präzedenzgraphen sind in Tabelle 7.9 dargestellt.

Die dritte und vierte Demonstration dieser Aufgabe liefert ein sehr unterschiedliches Bild (siehe Abbildung 7.16 (b)): Nach dem Suppenlöffel kann der flache Teller ohne Probleme platziert werden. Um jedoch nun den Suppenteller darauf zu stellen, musste zuerst die Untertasse an ihren Platz gebracht

(a) (b)

Abbildung 7.15: Start- und Zielkonfiguration der Tischdeckaufgabe mit sechs Objekten.

werden. Danach kann wieder in gewohnter Weise die Tasse und zuletzt der Kaffeelöffel platziert werden.

Bereits aus diesen zwei unterschiedlichen Reihenfolgen kann das System eine Menge irrelevanter Reihenfolgebeziehungen eliminieren. Die letztendlich gefundenen Relationen sind in Tabelle 7.9 grau hinterlegt. Man sieht also, daß das System beim Vorliegen entsprechender Demonstration auch einen großen Anteil der generierten Präzedenzrelationen bei fortgeschrittenem Lernvorgang durchaus als irrelevant erkennen und verwerfen kann. Der resultierende Präzedenzgraph ist in Abbildung 7.16 (c) dargestellt. Er beinhaltet bereits eine große Menge an Umordnungsmöglichkeiten zum Ausführungszeitpunkt. Lediglich fünf verschiedene Reihenfolgebeziehungen sind einzuhalten: Die Platzierung der Untertasse muß vor der Tasse und dem kleinen Löffel erfolgen (letzteres ist eine transitive Kante), die der Tasse vor dem Löffel, die des flachen Tellers vor dem Suppenteller, und die der Untertasse vor dem Suppenteller. Die letzte Reihenfolgebeziehung rührt aus der komplexen Anfangslage her, in welcher die Untertasse die direkte Bewegung des Suppentellers behindert und eine Entfernung der Untertasse nötig macht.

Am Beispiel des Tischdeckens mit sechs Operationen konnte gezeigt werden, wie schnell der Lernfortschritt des System vonstatten gehen kann, wenn entsprechend gute Lernbeispiele vorliegen. Bereits aus zwei unterschiedlichen Reihenfolgen der Vorführung der Aufgabe läßt sich der Präzedenzgraph mit den meisten Freiheitsgraden für diese Aufgabe rekonstruieren.

In diesem Abschnitt wurde das Verfahren zum Lernen von Taskpräzedenzgraphen evaluiert, welches in Abschnitt 6.2 theoretisch fundiert wurde. Damit wurde ein Verfahren zum autonomen Erschließen von Handlungswis-

$\mathbf{E}^{D_1}$	$o_1 \rightarrow_G o_3$	$o_1 \rightarrow_G o_5$	$o_1 \rightarrow_G o_6$
	$o_1 \rightarrow_G o_4$	$o_2 \rightarrow_G o_4$	$o_2 \rightarrow_G o_6$
	$o_2 \rightarrow_G o_5$	$o_3 \rightarrow_G o_5$	$o_3 \rightarrow_G o_6$
	$o_4 \rightarrow_G o_6$	$o_5 \rightarrow_G o_6$	$o_4 \rightarrow_G o_5$
	$o_3 \rightarrow_G o_4$	$o_2 \rightarrow_G o_3$	$o_1 \rightarrow_G o_2$
$\mathbf{E}^{D_3}$	$o_6 \rightarrow_G o_1$	$o_6 \rightarrow_G o_2$	$o_6 \rightarrow_G o_3$
	$o_6 \rightarrow_G o_5$	$o_4 \rightarrow_G o_5$	$o_4 \rightarrow_G o_3$
	$o_4 \rightarrow_G o_2$	$o_1 \rightarrow_G o_2$	$o_1 \rightarrow_G o_3$
	$o_5 \rightarrow_G o_3$	$o_2 \rightarrow_G o_3$	$o_5 \rightarrow_G o_2$
	$o_1 \rightarrow_G o_5$	$o_4 \rightarrow_G o_1$	$o_6 \rightarrow_G o_4$
$\mathbf{E}_3$	$o_1 \rightarrow_G o_3$	$o_1 \rightarrow_G o_5$	$o_4 \rightarrow_G o_5$
	$o_2 \rightarrow_G o_3$	$o_1 \rightarrow_G o_2$	

Tabelle 7.9: Präzedenzrelationen im restriktivsten Präzedenzgraph für jede Demonstration sowie gelernter Gesamtpräzedenzgraph am Beispiel des Tischdeckens mit sechs Objekten.

sen in einer Handlungsdatenbasis evaluiert.

7.3.3 Evaluation des Lernens repetitiver Aufgaben

Zur Evaluation der Algorithmen und Verfahren zum Lernen repetitiver Aufgaben (siehe Abschnitt 6.4) wurden zwei Aufgaben verwendet, welche einen hohen Grad an Wiederholung aufweisen: Die eine Aufgabe bestand aus dem Ausräumen eines Spülmaschinenkorbs. Hierbei wurden vier Becher, welche im Spülmaschinenkorb lokalisiert waren, aus diesem hinausgenommen und auf den Tisch gestellt. Die einzelnen Schlüsselframes dieser Aufgabe sind in Abbildung 7.17 zu sehen. Die andere Aufgabe bestand darin, für zwei Personen ein aus zwei Objekten bestehendes Tischset herzurichten, und zwar jeweils einen Teller und einen Becher (siehe Abbildung 7.18).

Der Versionsraum, wie er nach der Demonstration der Aufgabe des Entleeren des Spülmaschinenkorbs aufgebaut ist, ist in Abbildung 7.19 teilweise dargestellt. Die Verschmelzung der Versionsräume aller Einzelaktionen zu einer einzigen hat eine Beschreibung ergeben, die die minimale Schnittmenge aller Aktionen darstellt. Der Versionsraum ist folgendermaßen zu interpretieren: Er enthält die Anweisung aus dem Spülmaschinenkorb (Before Location: *IN (obj, korb)*) alle Becher (Object: *class becher*) auf das silberne Tischset zu legen (To: *object set-silber*), und zwar so, daß hinterher die Bedingungen *ON (obj, set-silber)* und *COVERS (obj, set-silber)* gelten. Die Versionsräume für Programme größerer Länge sind in Abbildung 7.19 aus Platzgrün-

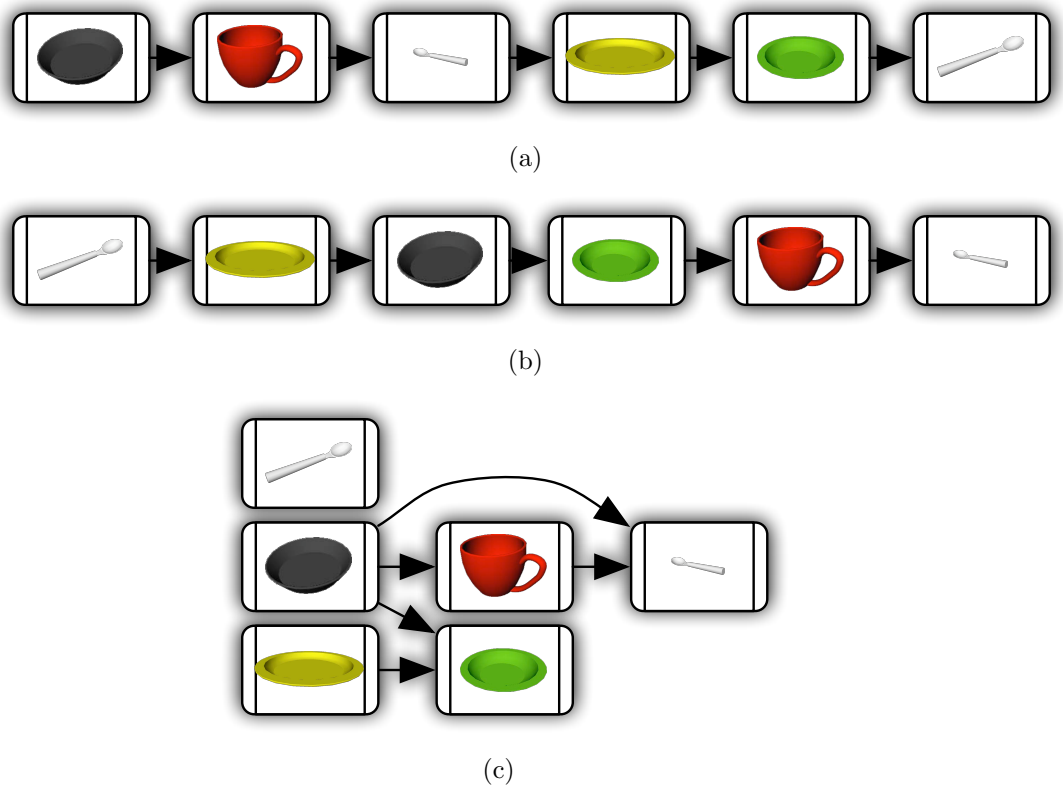
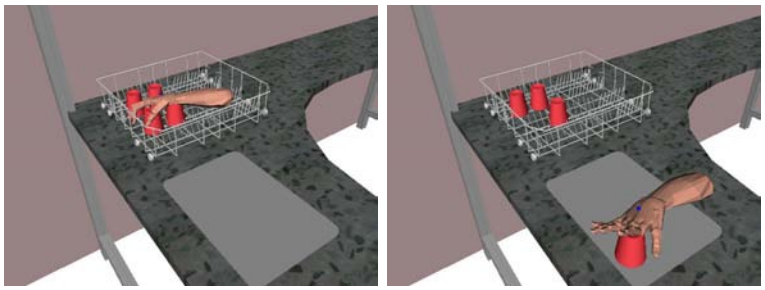


Abbildung 7.16: Präzedenzgraph für das Beispiel des Tischdeckens mit sechs Objekten.

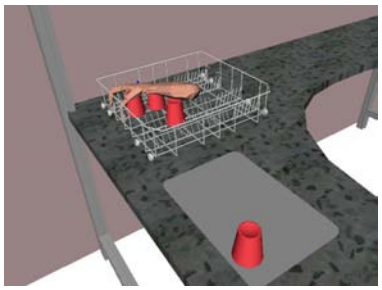
den ausgelassen worden, sind jedoch weniger kompakt und aussagekräftig. Die Modellierung der Versionsräume mit einer probabilistischen Bewertungsfunktion liefert als optimales Programm das mit der Länge 1. Die einzelnen Werte können in Abbildung 7.21 (a) und der mittleren Spalte von Tabelle 7.10 eingesehen werden.

Das komplexere Beispiel des Tischdeckens mit zwei Objekten für mehrere Personen liefert bei Annahme, daß es sich ein Programm der Länge eins handelt nur eine recht allgemeine Operation, die jede Art von Objekt (Object: *nil*) auf ein Objekt der Klasse Tischset ablegt. Die Hypothese, daß das Programm exakt zwei Operationen in der Schleife hat, liefert deutlich stärkere Aussagen, nämlich daß zuerst ein Teller (object: *class teller*) auf ein silbernes Tischset (relative: *class set-silber*) zu legen ist, und zwar so daß hinterher die Relationen *ON(obj, set-silber)*, *COVERS(obj, set-silber)* sowie *ALGN_EAST(obj, set-silber)* gelten. Letzteres sagt aus, daß der Teller am östlichen Rand des silbernen Sets ausgerichtet ist. Anschließend ist ein Becher (object: *class becher)* auf dasselbe silberne Tischset zu platzieren, und

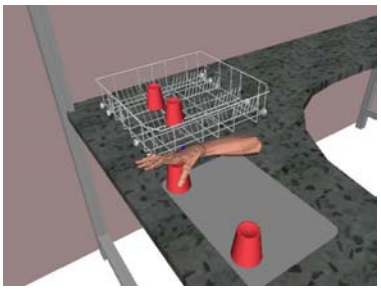


(a)

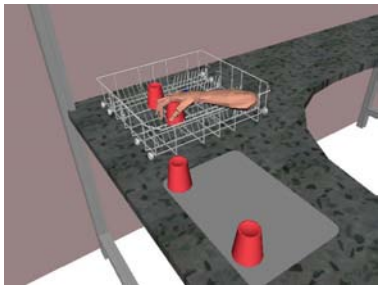
(b)



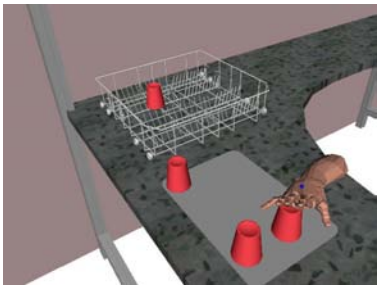
(c)



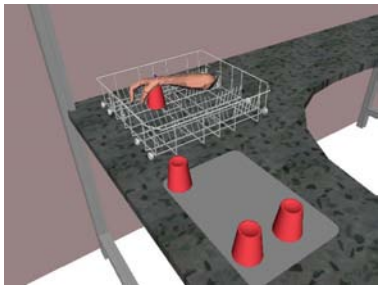
(d)



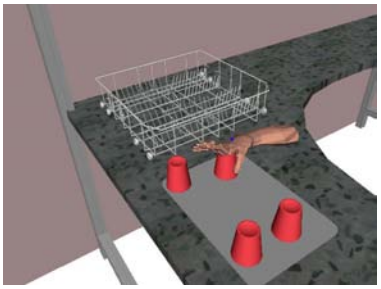
(e)



(f)

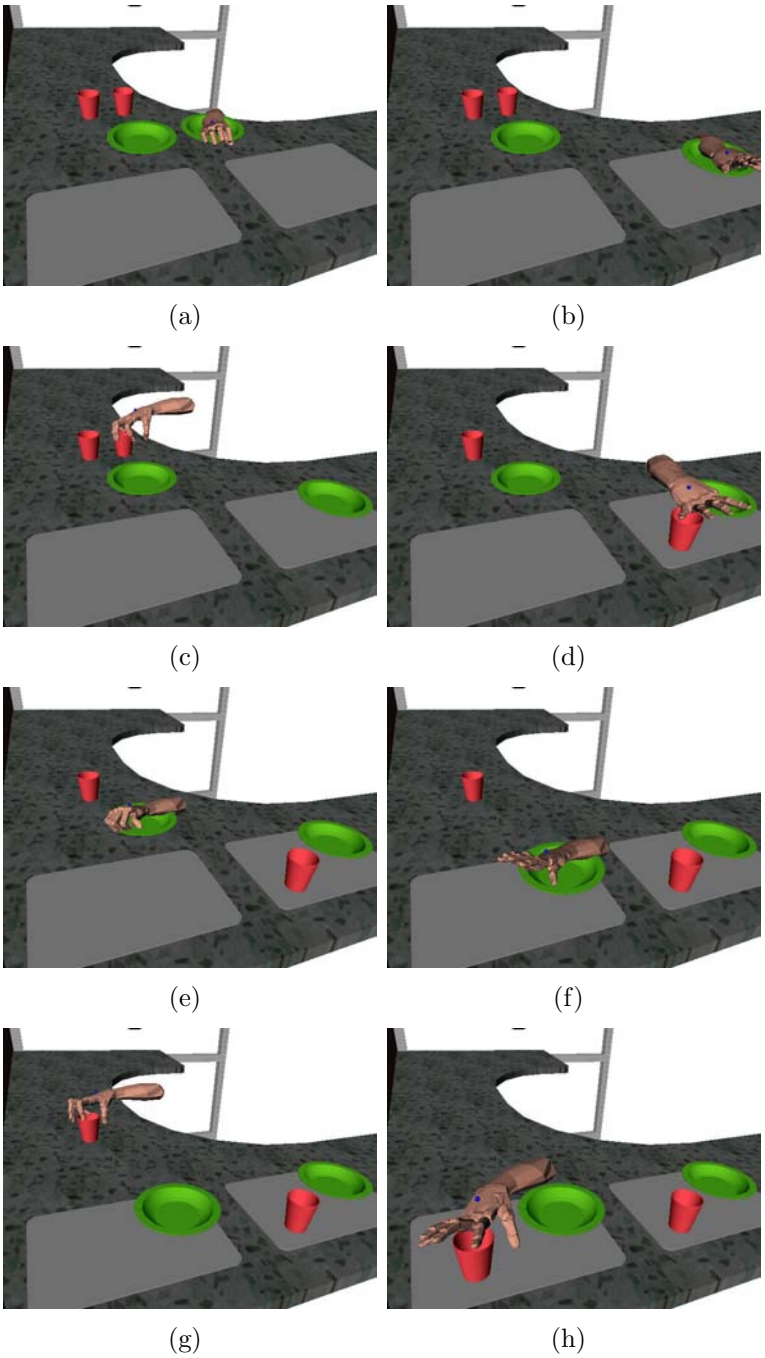


(g)



(h)

Abbildung 7.17: Lernen von repetitiven Aufgaben. Entladen des Spülmaschinenkorbs. (a-h) Die Schlüsselframes mit den Indizes 18, 55, 77, 107, 124, 152, 169, 194.



(a) (b) (c) (d) (e) (f) (g) (h)

Abbildung 7.18: Lernen von repetitiven Aufgaben. Tischdecken für mehrere Personen. (a-h) Die Schlüsselframes mit den Indizes 19, 55, 77, 108, 128, 156, 173, 202.

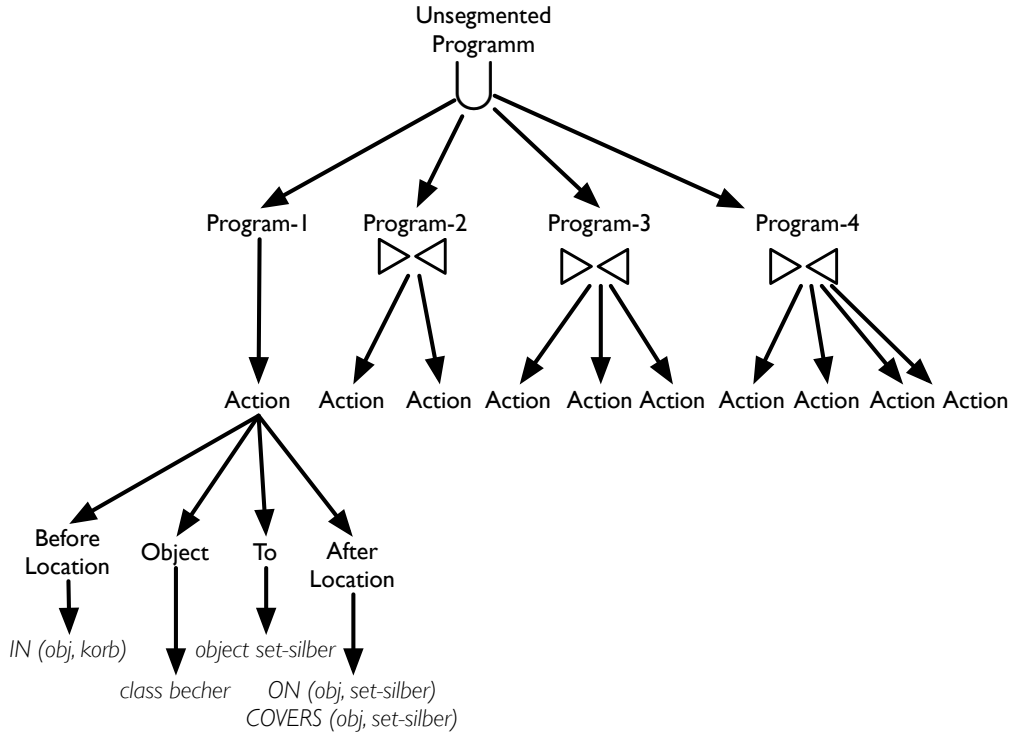


Abbildung 7.19: Versionsraum für die repetitive Aufgabe des Ausräumens des Spülmaschinenkorbs.

zwar so, daß zusätzlich noch *WEST(obj, teller)* gilt, der Becher also westlich des Tellers zu liegen kommt.

n	Korb ausräumen	Tischdecken für mehrere Personen
1	**0.86430174**	0.25929055
2	0.62188506	**0.75821686**
3	0.45893526	0.17608522
4	0.31304073	0.46570644

Tabelle 7.10: Bewertung für die Programmlänge. Das Programm mit der höchsten Bewertung wird ausgewählt (fett markiert).

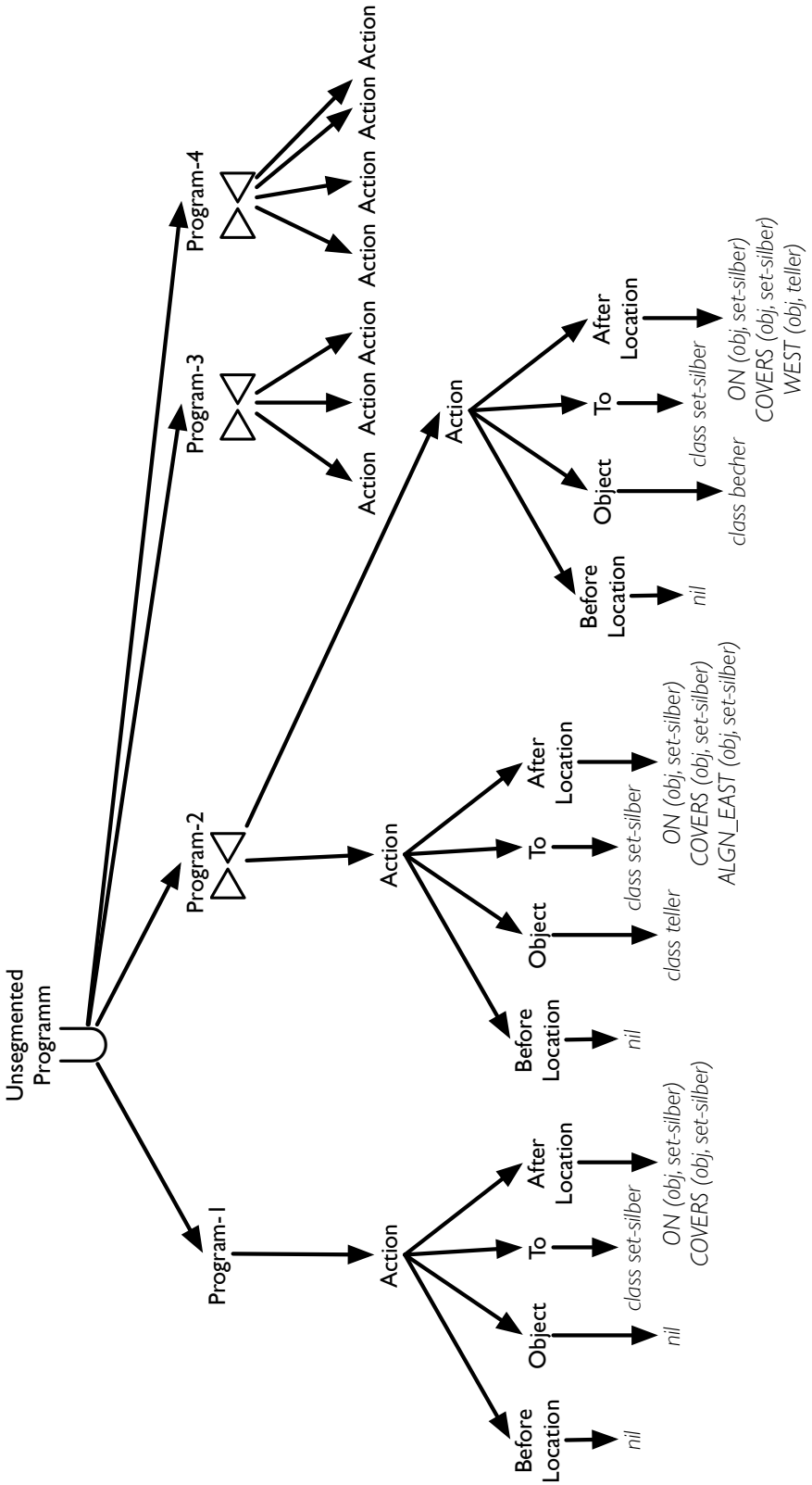


Abbildung 7.20: Versionsraum für die repetitive Aufgabe des Tischdeckens für mehrere Personen.

Die Betrachtung der Werte für das probabilistische Gütemaß für Versionsräume liefert eine deutliche Präferenz für Programme mit geradzahligen Längen (siehe Abbildung 7.21 (b) und Tabelle 7.10, rechte Spalte). Die Versionsräume für Programme der Länge eins und drei sind aufgrund der Versuche, zu viele Aktionen unterschiedlicher Art nahezu kollabiert und spielen für die Endauswahl keine Rolle mehr. Die Entscheidung zwischen den Hypothesen für die Längen zwei und vier gewinnt das Programm mit der Länge zwei aufgrund der Bevorzugung kurzer Programme.

Abschließend läßt sich feststellen, daß das System mit ausreichend hoher Sicherheit repetitive Aufgaben erkennen und den Schleifenkörper mit genügender Präzision bestimmen konnte.

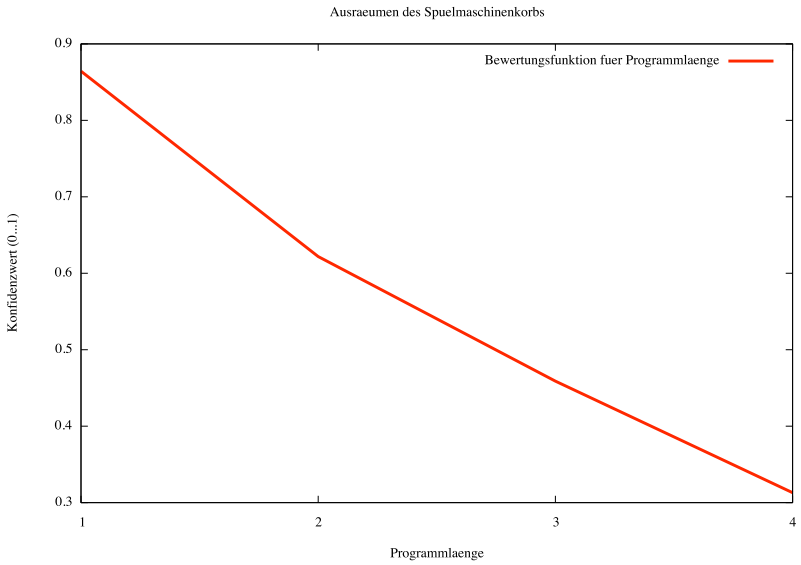
7.4 Zusammenfassung

Dieses Kapitel beschrieb die Experimente, welche zur Evaluation der im Rahmen dieser Arbeit entwickelten Verfahren und Algorithmen zum inkrementellen und interaktiven Lernen von Handlungswissen durchgeführt wurden.

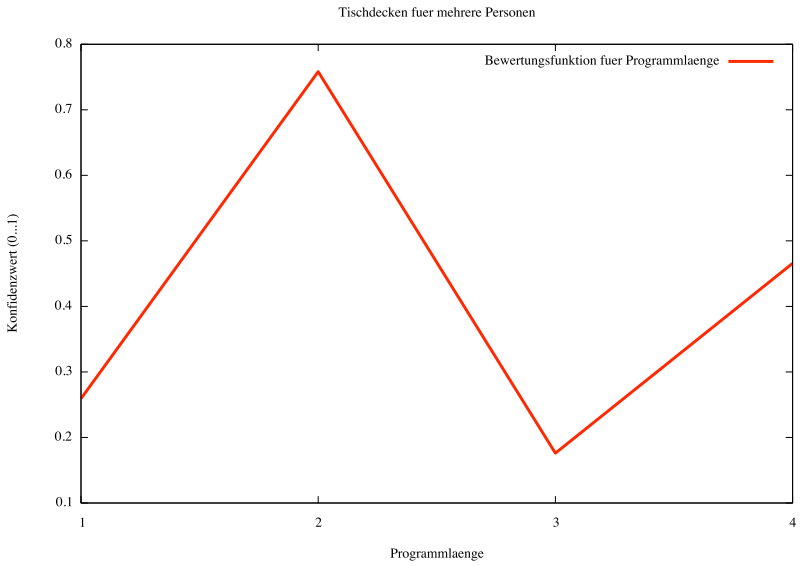
Es konnte gezeigt werden, daß die lexikalische und syntaktische Analysephase mit unterschiedlichen Sensormodalitäten, wie dem eigens für diese Arbeit geschaffenen Trainingszentrum und den Daten eines optischen Bewegungserfassungssystems umzugehen versteht. Aus beiden Arten von Sensordatenströmen ließen sich hierarchisch-funktionale Beschreibungen von Handlungswissen extrahieren und interpretieren, wie sie am Beispiel von Manipulations- und Einschenkoperationen verdeutlicht wurden.

Das System konnte selbstständig Klassen von Handlungswissen in einer größeren Menge von Demonstrationen erkennen und die gewonnene Klassifikation zur Bestimmung klassenvarianter Relevanzen von Merkmalen der Aufgaben nutzen. Hierbei konnte zwischen der Klasse des Befüllens von Trinkgefäßen und verschiedenen Klassen zum Tischdecken mit variablen Konfigurationen unterschieden werden.

In der letzen schlußfolgernden Phase konnten die Verfahren zum inkrementellen Lernen von Präzedenzgraphen anhand unterschiedlicher Beispiele verifiziert und getestet werden. Die Evaluation des Lernens repetitiver Programme erzielte interessante Ergebnisse, welche eine weitergehende Beschäftigung in der Zukunft nahelegen.



(a)



(b)

Abbildung 7.21: Bewertungsmaß für die Programmlänge. (a) Ausräumen des Spülmaschinenkorbs (b) Tischdecken mit zwei Objekten für mehrere Personen.

Kapitel 8

Schlussbetrachtungen

Die aktuelle Entwicklung der Servicerobotik führt dazu, daß Roboter in zunehmenden Maße die Fabrikhallen verlassen und in unstrukturierten Umgebungen mit zunehmend komplexen Aufgaben betraut werden. Der dadurch erforderliche hohe Programmieraufwand wird mehr und mehr auf den Anwender übertragen werden.

Dazu ist ein Programmiersystem nötig, welches auf natürliche Art und Weise mit dem menschlichen Anwender interagiert und dem Laien eine Roboterkommandierung ohne Programmierkenntnisse ermöglicht. Dazu ist ein solches Robotersystem mit Intelligenz und Lernfähigkeit auszustatten.

In dieser Arbeit wurde ein wesentlicher Beitrag geleistet, entwicklungspsychologische Erkenntnisse in ein maschinelles System zu integrieren, welches auf autonome und inkrementelle Art und Weise sein Handlungs- und Programmwissen erweitert und vom Benutzer abschaut.

Hierzu wurde ein mehrstufiger Verstehens- und Analyseprozess vorgestellt, welcher kommunikationstheoretisch fundiert als Übersetzungsprozess ausgelegt wurde. Er besteht zuerst aus der Phase der Dekodierung des elementaren Vokabulars, welches in einer Demonstration des Benutzers verschiedene Basishandlungen detektiert (lexikalische Phase). Darauf aufbauend wird die syntaktische Struktur gemäß einer definierten Handlungsgrammatik abgeleitet, welche es erlaubt, das Lernproblem auf unterschiedlichen Abstraktionsebenen zu betrachten. Diese syntaktische Analysephase erstellt eine baumartige Repräsentation der demonstrierten Handlung. Innerhalb dieses Strukturbaums wird die Demonstration hinsichlich iherer Bedeutung, d.h. ihrer Vorbedingungen und Beiträge, in der semantischen Analysephase bestimmt. Damit einher geht auch eine Bewertung der Relevanz unterschiedlicher möglicher Hypothesen für die eigentliche Intention des Benutzers und eine Einordnung des gelernten Wissens in den Referenzrahmen der bisherigen Erfahrung des Systems. Basierend auf dieser Einordnung und Relevanzbewertung kann

das System in einer abschließenden Phase schlußfolgernd sein Handlungswissen über einzelne Erfahrungen hinweg verallgemeinern und abstrahieren. Die Verfahren, die hierbei zum Einsatz kamen, zeichnen sich dadurch aus, daß sie die aufgestellten Hypothesen niemals als endgültigen Schlußpunkt des Lernens betrachten, sondern neue, erweiternde oder sogar gegensätzliche, in der Zukunft liegende Erfahrungen antizipieren.

In den einzelnen Phasen wurden jeweils unterschiedliche Methoden und Modelle vorgeschlagen und evaluiert, welche die erorderliche Interpretationsleistung erbringen: Im lexikalischen Analyseschritt wurde das Konzept der Detektoren-Netzwerke eingeführt, welche als Mustererkenner auf den zur Verfügung stehenden Eingabedaten arbeiten. Das Konzept wurde validiert anhand der Erkennung von Greifoperationen, Objektinteraktionen wie dem Umschenken von Flüssigkeiten oder der Änderung von Objektzuständen durch Manipulationshandlungen.

In der syntaktischen Analyse kamen grammatikalische Modellierungen zum Einsatz, um die Struktur von Handlungen und Handlungsteilen zu bestimmen. Diese sind einfach erweiterbar und können auf unterschiedlichste Anwendungsfelder, je nach dem gewünschten Einsatzgebiet zugeschnitten und instantiiert werden.

Die Repräsentation des gesamten Erfahrungsschatzes des Systems wurde mittels fallbasierter Methoden bewerkstelligt. Es wurde weiterhin ein Verfahren zum autonomen Erkennen von Prototypen in größeren Handlungsdatenbasen mittels Clustering-Methoden entwickelt, was zusätzlich Informationen zu den wesentlichen Eigenschaften und Merkmalen der einzelnen Konstituenten der Erfahrung liefert. Dies erlaubt es, Verbindungen zwischen einzelnen Wissensinhalten zu ziehen und diese für die spätere Nutzung zur Verfügung zu stellen.

8.1 Diskussion

Diese Arbeit entstand unter der Annahme holistischer und allgemeingültiger Lösungsverfahren. Dennoch mußten Abstriche bei diesem Prinzip in Kauf genommen werden, um ein abgeschlossenes und realisierbares System zu entwickeln. Diese gehen einher mit Einschränkungen, denen das System in seinem aktuellen Entwicklungsstand unterworfen ist. Diese sollen hier kurz identifiziert und benannt werden:

- Das System ist mit dem Anspruch der Sensorunabhängigkeit entworfen worden. Es kamen jedoch nur Sensoren zum Einsatz welche die vielfältigen Probleme visueller Sensorik umgingen. Gerade in Bezug

auf Okklusionen und Sichtbarkeitsbeschränkungen konnten so Vereinfachungen gemacht werden. Das Ziel eines autonomen Systems, welches nicht auf externe, absolute Sensorik angewiesen ist, bleibt jedoch unverändert bestehen.

- Durch die Auswahl der Basisaktionen, welche vom System erkannt werden können, stellt eine weitere Einschränkung des Systems dar. Werden Basishandlungen, wie beispielsweise Interaktionshandlungen vorgenommen, die im Basisvokabular nicht enthalten sind, so ist das System in Bezug auf diese blind. Existieren keine Detektoren für eine bestimmte Elementaraktion, so kann das System diese nicht erkennen und für eine Erweiterung seines Erfahrungsschatzes nutzen.

- Die Definition einer Handlungsgrammatik ist vom Anwendungsentwickler vorzunehmen. Das Design dieser syntaktischen Struktur, welcher eine Handlung genügen muß, bestimmt auch wie diese zu analysieren und hinsichtlich ihrer Semantik zu interpretieren ist. Sind die entsprechenden Strukturen nicht gegeben, so ist das System auch hier blind in Bezug auf darüber hinausgehenden Handlungen.

- Die Fähigkeit des Systems, explizierbares Handlungswissen zu erstellen verlangt eine Dialogkomponente, welche dem Benutzer dieses Wissen in einer natürlichsprachigen Art und Weise mitteilt. Im Rahmen dieser Arbeit wurden lediglich visuelle Rückkopplungen in Betracht gezogen. Ein letztendlich einsetzbares System würde mir hoher Wahrscheinlichkeit von multimodalen Benutzerschnittstellen profitieren.

- Im System sind momentan nur Verfahren zum Erschließen von Umordnungsmöglichkeiten und repetitiven Abläufen integriert. Es sind jedoch weitergehende Verfahren an dieser Stelle denkbar, von denen das System profitieren kann und wird. Diese umfassen räumlich-zeitliches Schlußfolgern[1] oder das Schlußfolgern mittels automatischer Planungs- und Problemlösungssysteme, basierend auf formalen Logiken erster und höherer Ordnung.

Die oben angeklungenen Einschränkungen und Limitationen des erstellten Systems bedingen vielfältige Weiterentwicklungen und Verbesserungen an verschiedenen Systemkomponenten. Diese können im Rahmen zukünftiger Forschungsarbeiten geleistet werden, wie sie im kommenden Abschnitt aufgezeigt werden sollen.

[1] engl. spatio-temporal reasoning

8.2 Ausblick

Das in dieser Arbeit skizzierte System wurde mit dem Anspruch und der Zielsetzung entworfen und implementiert, das Phänomen der Intelligenz und des lebenslangen Lernens für technische System nutzbar zu machen. Die Bescheidenheit gebietet jedoch, diese Arbeit nur als Meilenstein auf dem langen Weg zu diesem Ziel aufzufassen. So wirft auch das hier entwickelte System eine Vielzahl interessanter Fragestellungen und damit verbunden potentielle Forschungsthemen auf, die eine tiefergehende Betrachtung rechtfertigen:

- Eine Übertragung der hier vorgestellten Methoden auf die eingeschränktere Sensorik, mit denen ein Robotersystem ausgestattet ist, wird eine Vielzahl interessanter Forschungsfragestellungen aufwerfen, wie beispielsweise, wie mit beschränkter Verfügbarkeit an Systemressourcen (Speicherplatz, Rechenkapazität) umzugehen ist, wie die geringere Aussagekraft der von interner Sensorik gelieferten Eingabedaten zu kompensieren ist und wie mit nur unvollständig erfaßbaren Handlungen (Okklusionen, etc.) zu verfahren ist.

- Sind die Sensorikprobleme gelöst, so ist ein System zum inkrementellen, autonomen Lernen mit Fähigkeiten auszustatten, die es ihm erlauben, Unsicherheit expliziter zu repräsentieren und anzugeben. Hier können Methoden zum Unscharfen Modellieren und Schließen zum Einsatz kommen.

- Ist ein explizites Maß für die Unsicherheit von Wissensinhalten gefunden, so kann und wird ein als intelligent zu bezeichnendes System dies ausnutzen, um diese durch eigene Handlungen oder durch Nachfrage bei Experten zu minimieren. Proaktives Handeln ermöglicht es einem solchen System, selbstständig seinen Erfahrungshorizont zu erweitern, indem es jeweils versucht, um die nächste „Ecke“ zu schauen.

- Menschliche Verständnis- und Lernprozeße sind stets eingebunden in unterschiedliche situative und emotionale Kontexte. Das hier vorgestellte System hat diese bisher ignoriert, da hierfür keine ausgefeilte Sensorik existiert. Gerade in der jüngeren Zeit werden jedoch Herangehensweisen entwickelt, diese technisch-kognitiven Systemen nahezubringen. Ein Einbezug solcher Methoden und Algorithmen kann dazu dienen, inkrementell lernende Systeme mit noch höherer Benutzerfreundlichkeit und Interaktivität auszustatten, was zu einem verbesserten Anwenderbezug genutzt werden kann.

- Kooperatives Handeln wird die Operationsweise eines lernenden System in Zukunft stark bestimmen. Die Frage, wie kooperatives Agieren und Lernen qualitativ und quantitativ gestalltet werden sollte, konnte bisher noch nicht einmal ansatzweise befriedigend beantwortet werden. Dieses Gebiet wird sicherlich vielfache Weiterentwicklungsmöglichkeiten zukünftiger Forschungsarbeiten und -ansätze bieten.

- In dieser Arbeit wurde eine dedizierte Architektur zum inkrementellen und interaktiven Lernen von Handlungswissen vorgestellt und evaluiert. Diese ist jedoch nur als Ausgangs- oder Startpunkt zur Entwicklung einer lernenden, intelligenten Gesamtarchitektur technisch-kognitiver Systeme anzusehen.

Abschließend bleibt nur noch zu betonen, daß sich zur Fortentwicklung der erstellten Konzepte, Methoden und Algorithmen umfangreiche und spannende Bereiche eröffnen, die hier leider nur ansatzweise skizziert werden konnten.

Literaturverzeichnis

[Aleotti et al. 2003] Aleotti, Jacopo, S. Caselli und M. Reggiani (2003). *Toward Programming of Assembly Tasks by Demonstration in Virtual Environments*. Proceedings 2003 Intl. Workshop on Robot and Human Interactive Communication, Millbrae, CA, USA, Oct. 31 - Nov. 2, S. 309–314.

[Aleotti et al. 2004] Aleotti, Jacopo, A. Skoglund und T. Ducket (2004). *Position Teaching of a Robot Arm by Demonstration with a wearable input Device*. In: *Proc. of the Intl. Conf. on Intelligent Manipulation and Grasping, Genoa, Italy, July 1-2*, S. 443–448.

[Alissandrakis et al. 2002] Alissandrakis, Aris, C. Nehaniv und K. Dautenhahn (2002). *Do as I Do: Correspondences across Different Robotic Embodiments*. In: Polania, D., J. Kim und T. Martinetz, Hrsg.: *Proc. Fifth German Workshop on Artificial Life (GWAL5)*, S. 143–152.

[Alissandrakis et al. 2003a] Alissandrakis, Aris, C. Nehaniv und K. Dautenhahn (2003a). *Solving the Correspondence Problem Between Dissimilarly Embodied Robotic Arms Using the ALICE Imitation Mechanism*. In: *Proceedings of the second International Symposium on Imitation in Animals and Artifacts*, S. 79–92.

[Alissandrakis et al. 2003b] Alissandrakis, Aris, C. Nehaniv und K. Dautenhahn (2003b). *Synchrony and Perception in Robotic Imitation across Embodiments*. In: *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, S. 923–930.

[Alissandrakis et al. 2005] Alissandrakis, Aris, C. Nehaniv, K. Dautenhahn und J. Saunders (2005). *Achieving Corresponding Effects on Multiple Platforms: Imitating in Context Using Different Effect Metrics*. In: *Proceedings of the Third International Symposium on Imitation in Animals and Artifacts*, S. 10–19.

[Arsenio 2004] Arsenio, A. M. (2004). *Learning task sequences from scratch: applications to the control of tools and toys by a humanoid robot*. In: *Proceedings of the 2004 IEEE International Conference on Control Applications*, Bd. 1, S. 400–405.

[Atkeson und Schaal 1997] Atkeson, Christopher und S. Schaal (1997). *Robot Learning From Demonstration*. In: Jr., D. H. Fisher, Hrsg.: *Proc. 14th Intl. Conf. on Machine Learning (ICML)*, S. 12–20. Morgan Kaufmann.

[Bandura 1969] Bandura, Albert (1969). *Principles of Behavior Modification*. Holt, Rinehart & Winston, New York.

[Bandura 1979] Bandura, Albert (1979). *Sozial-kognitive Lerntheorie*. Klett-Cotta, Stuttgart.

[Bandura und Mischel 1965] Bandura, Albert und W. Mischel (1965). *Modification of self-imposed delay of reward through exposure to life and symbolic models*. Journal of Personality and Social Psychology, 2:698–705.

[Bauckhage et al. 1998] Bauckhage, C., F. Kummert und G. Sagerer (1998). *Modeling and recognition of assembled objects*. In: *Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society*, Bd. 4, S. 2051–2056.

[Baum 1973] Baum, W. (1973). *The Correlation-based Law of Effect*. Journal of the Experimental Analysis of Behavior, 20:137–153.

[Becher et al. 2003] Becher, R., P. Steinhaus und R. Dillmann (2003). *Interactive Object Modelling for a Humanoid Service Robot*. In: *Proceedings of the Conference on Humanoids*.

[Becher et al. 2006] Becher, R., P. Steinhaus, R. Zöllner und R. Dillmann (2006). *Design and Implementation of an Interactive Object Modelling System*. In: *Proc. Robotik/IRS 2006*.

[Bentivegna und Atkeson 2001] Bentivegna, D. C. und C. G. Atkeson (2001). *Learning from observation using primitives*. In: *Proceedings of the IEEE International Conference on Robotics and Automation*, Bd. 2, S. 1988–1993.

[Bentivegna et al. 2002] Bentivegna, D. C., A. Ude, C. G. Atkeson und G. Cheng (2002). *Humanoid robot learning and game playing using*

PC-based vision. In: *IEEE/RSJ International Conference on Intelligent Robots and System, 2002*, Bd. 3, S. 2449–2454.

[Bentivegna 2004] Bentivegna, Darrin C. (2004). *Learning from Observation Using Primitives*. Doktorarbeit, College of Computing, Georgia Institute of Technology.

[Besl und McKay 1992] Besl, Paul und N. McKay (1992). *A Method for Registration of 3-D Shapes*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 14(2):239–256.

[Beth et al. 2003] Beth, T., I. Boesnach, M. Haimerl, J. Moldenhauer, K. Bös und V. Wank (2003). *Characteristics in Human Motion – From Acquisition to Analysis*. In: *IEEE Intl. Conference on Humanoid Robots HUMANOIDS*, S. 56ff.

[Billard et al. 2004] Billard, Aude, Y. Epars, S. Calinon, S. Schaal und G. Cheng (2004). *Discovering optimal imitation strategies*. Robotics and Autonomous Systems, 47(2-3):69–77.

[Billard und Siegwart 2003] Billard, Aude und R. Siegwart, Hrsg. (2003). *Workshop on Robot Programming Through Demonstration*.

[Bluethmann et al. 2004] Bluethmann, W., R. Ambrose, M. Diftler, E. Huber, A. Fagg, M. Rosenstein, R. Platt, R. Grupen, C. Breazeal, A. Brooks, A. Lockerd, R. A. Peters, O. C. Jenkins, M. Mataric und M. Bugajska (2004). *Building an autonomous humanoid tool user*. In: *4th IEEE/RAS International Conference on Humanoid Robots*, Bd. 1, S. 402–421.

[Breazeal et al. 2005] Breazeal, Cynthia, D. Buchsbaum, J. Gray, D. Gatenby und B. Blumberg (2005). *Learning From and About Others: Towards Using Imitation to Bootstrap the Social Understanding of Others by Robots*. Artificial Life, 11(1):31–62.

[Calderon 2005] Calderon, Carlos Acosta (2005). *Robot Imitation from Human Body Movements*. In: *Proceedings of the Convention of the Society for the Study of Artificial Intelligence and Simulation of Behaviour*.

[Calderon und Hu 2003] Calderon, Carlos Antonio Acosta und H. Hu (2003). *Robotic Societies: Elements of Learning by Imitation*. In: *Proceedings of the IASTED International Conference Applied Informatics*.

[Calinon und Billard 2004] Calinon, S. und A. Billard (2004). *Stochastic Gesture Production and Recognition Model for a Humanoid Robot*. In: *IEEE/RSJ Intl Conference on Intelligent Robots and Systems (IROS)*.

[Calinon und Billard 2006a] Calinon, S. und A. Billard (2006a). *Learning of Gestures by Imitation in a Humanoid Robot*. Cambridge University Press, K. Dautenhahn and C.L. Nehaniv Aufl. in press.

[Calinon und Billard 2006b] Calinon, S. und A. Billard (2006b). *Teaching a Humanoid Robot to Recognize and Reproduce Social Cues*. In: *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*.

[Calinon et al. 2005] Calinon, S., F. Guenter und A. Billard (2005). *Goal-Directed Imitation in a Humanoid Robot*. In: *Proceedings of the IEEE Intl Conference on Robotics and Automation (ICRA)*, Barcelona, Spain.

[Calinon et al. 2006] Calinon, S., F. Guenter und A. Billard (2006). *On Learning the Statistical Representation of a Task and Generalizing it to Various Contexts*. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

[Chen und McCarragher 2000] Chen, Jason und B. J. McCarragher (2000). *Programming by Demonstration - Constructing Task Level Plans in a Hybrid Dynamic Fram*. In: *Proc. 2000 IEEE Intl. Conf. on Robotics & Automation, San Francisco, CA*.

[Cohen 2000] Cohen, Paul R. (2000). *Learning Concepts by Interaction*. Technischer Bericht, Computer Science Department, University of Massachusetts at Amherst. Technical Report 00-52.

[Cutkosky 1989] Cutkosky, Mark (1989). *On Grasp Choice, Grasp Models and the Design of Hands for Manufacturing Tasks*. IEEE Transactions on Robotics and Automation, 5(3):269–279.

[Dillmann et al. 1999] Dillmann, R., O. Rogalla, M. Ehrenmann, R. Zöllner und M. Bordegoni (1999). *Learning Robot Behaviour and skills based on human demonstration and advice: the machine learning paradigm*. In: *9th International Symposium of Robotics Research (ISSR '99), Snowbird, UT, USA*, S. 229–238.

[D'Souza et al. 2001] D'Souza, A., S. Vijayakumar und S. Schaal (2001). *Learning inverse kinematics*. In: *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Bd. 1, S. 298–303.

[Duda et al. 1998] Duda, Hart und Stork (1998). *Pattern Classification*. John Wiley & Sons, Inc.

[Ehrenmann et al. 2002] Ehrenmann, M., R. Zollner, O. Rogalla und R. Dillmann (2002). *Programming service tasks in household environments by human demonstration*. In: *Proc. 11th IEEE International Workshop on Robot and Human Interactive Communication*.

[Ehrenmann et al. 2003] Ehrenmann, M., R. Zöllner, O. Rogalla, S. Vacek und R. Dillmann (2003). *Observation in Programming by Demonstration: Training and Exection Environment*. In: *Humanoids 2003, Karlsruhe/Munich, Germany, October 2003*.

[Ehrenmann 1998] Ehrenmann, Markus (1998). *Objekterkennung in Kamerabildern in einem fusionierten Ansatz von Maschinensehen und Griffwinkelbetrachtung*. Diplomarbeit, Universität Karlsruhe, Fakultäta für Informatik.

[Ekvall und Kragic 2006] Ekvall, S. und D. Kragic (2006). *Learning Task Models from Multiple Human Demonstrations*. In: *In RO-MAN 06: The 15th IEEE International Symposium on Robot and Human Interactive Communication, University of Hertfordshire, UK*.

[Fikes et al. 1972] Fikes, R.E., P. Hart und N. Nilsson (1972). *Learning and executing generalized robot plans*. Artificial Intel ligence, 3(4).

[Flavell 1963] Flavell, J. (1963). *The developmental psychology of Jean Piaget*. Van Nostrand, New York.

[Floyd 1962] Floyd, R.W. (1962). *Algorithm 97: Shortest Path*. Communications of the ACM, 5(6):345.

[Friedrich et al. 1999] Friedrich, H., V. Grossmann, M. Ehrenmann, O. Rogalla, R.-D. Zöllner und R. Dillmann (1999). *Towards cognitive elementary operators: grasp classification using neural network classifiers*. In: *Proceedings of the IASTED International Conference on Intelligent Systems and Control*, S. 121–126.

[Friedrich 1998] Friedrich, Holger (1998). *Interaktive Programmierung von Manipulationssequenzen*. Doktorarbeit, Fakultät für Informatik der Universität Karlsruhe (TH).

[Fritsch et al. 2000] Fritsch, J., F. Lomker, M. Wienecke und G. Sagerer (2000). *Detecting assembly actions by scene observation*. In: *International Conference on Image Processing*, Bd. 1, S. 212–215.

[Funke und Vaterrodt-Plünnecke 2004] Funke, Joachim und B. Vaterrodt-Plünnecke (2004). *Was ist Intelligenz?*. Beck, München, 2. Auflage Aufl.

[Garey und Johnson 1978] Garey, Michael R. und D. S. Johnson (1978). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco.

[Gennari et al. 1989] Gennari, John, P. Langley und D. Fisher (1989). *Models of Incremental Concept Formation*. Artificial Intelligence, 40:11–61.

[Ginsburg und Opper 1969] Ginsburg, H. und S. Opper (1969). *Piaget's theory of intellectual development. An introduction.*. Prentice-Hall, New-Jersey.

[Goodman 1989] Goodman, M. (1989). *Case-Based Reasoning in Battle Planning*. In: *Proceedings of the Workshop on Case-Based Reasoning (DARPA), Pensacola Beach, FL.*

[Goos und Waite 1984] Goos, Gerhard und W. Waite (1984). *Compiler Construction*. Springer.

[Grunwald et al. 2001] Grunwald, G., G. Schreiber, A. Albu-Schffer und G. Hirzinger (2001). *Touch: The Direct Type of Human Interaction with a Redundand Service Robot*. In: *Proceedings of the IEEE Int. Workshop on Robot and Human Interactive Communication (RO-MAN).*

[Guthke 1980] Guthke, Jürgen (1980). *Ist Intelligenz meßbar? Eine Einführung in die Probleme der psychologischen Intelligenzforschung und Intelligenzdiagnostik*. Deutscher Verlag der Wissenschaften, Berlin, 2. Auflage Aufl.

[Hauck et al. 1998] Hauck, A., M. Sorg, G. Färber und T. Schenk (1998). *A biologically motivated model for the control of visually guided reach-to-grasp movements*. In: *Proceedings of the International Conference on Intel ligent Systems*, S. 295–300.

[Hegel 1817] Hegel, G.W.F. (1817). *Näherer Begriff und Einteilung der Logik*. Enzyklopädie der philosophischen Wissenschaften, S. 79–82.

[Hersch und Billard 2006] Hersch, Micha und A. G. Billard (2006). *A model for imitating human reaching movements*. In: *HRI '06:*

Proceeding of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction, S. 341–342, New York, NY, USA. ACM Press.

[Hirsh 1991] Hirsh, H. (1991). *Theoretical Underpinnings of Version Spaces*. In: *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, S. 665—670.

[Holzapfel et al. 2006] Holzapfel, H., T. Schaaf, H. Ekenel, C. Schaa und A. Waibel (2006). *A Robot learns to know people - First contacts of a Robot*. Springer.

[Holzapfel und Waibel 2006] Holzapfel, H. und A. Waibel (2006). *A Multilingual Expectations Model for Contextual Utterances in Mixed-Initiative Spoken Dialogue*. In: *International Conference on Speech and Language Processing (INTERSPEECH 2006), Pittsburgh PA, USA*.

[Horn 1987] Horn, Berthold (1987). *Closed-form solution of absolute orientation using unit quaternions*. Journal of the Optical Society of America, Part A, 4(4):629–642.

[Hugues und Drogoul 2002] Hugues, Louis und A. Drogoul (2002). *Synthesis of Robot's Behaviors from few Examples*. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[Humanoid Animation Working Group 2003] Humanoid Animation Working Group (2003). *Information technology — Computer graphics and image processing — Humanoid animation (H-Anim), Annex B*. ISO/IEC FCD 19774 - Humanoid Animation Specification.

[Iba et al. 2003] Iba, Soshi, C. Paredis und P. Khosla (2003). *Intention aware interactive multi-modal robot programming*. In: *Proceedings. 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Bd. 4, S. 3479 – 3484.

[Iba et al. 2004] Iba, Soshi, C. Paredis und P. Khosla (2004). *Interactive Multi-Modal Robot Programming*. In: *9th International Symposium on Experimental Robotics*.

[Ijspeert et al. 2002] Ijspeert, A. J., J. Nakanishi und S. Schaal (2002). *Movement imitation with nonlinear dynamical systems in humanoid robots*. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Bd. 2, S. 1398–1403.

[Immersion] Immersion. *CyberGlove II Wireless Glove*.

[Jiar et al. 1996] Jiar, Yunde, M. Wheeler und K. Ikeuchi (1996). *Hand Action Perception and Robot Instruction*. Technischer Bericht CMU-CS-96-116, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA.

[Kang und Ikeuchi 1997] Kang, S.B. und K. Ikeuchi (1997). *Toward automatic robot instruction from perception – Mapping human grasps to manipulator grasps*. IEEE Trans. on Robotics and Automation, 13(1):81–95.

[Knoop et al. 2006] Knoop, S., S. Vacek, K. Steinbach und R. Dillmann (2006). *Sensor Fusion for model-based 3D tracking*. In: *Proceedings of the International Conference o Multisensor Fusion and Integration for Intelligent Systems (MFI)*.

[Kolodner 1993] Kolodner, Janet L. (1993). *Case Based Reasoning*. Morgan Kaufmann Publishers, Inc.

[Kuniyoshi et al. 1994] Kuniyoshi, Yasuo, M. Inaba und H. Inoue (1994). *Learning by Watching: Extracting reusable Task Knowledge from Visual Observation of Human Performance*. IEEE Trans. on Robotics and Automation, 10(6):799–822.

[Lau 2001] Lau, Tessa (2001). *Programming by Demonstration: a Machine Learning Approach*. Doktorarbeit, Department of Computer Science and Engineering, University of Washington.

[Lau et al. 2004] Lau, Tessa, L. Bergman, V. Castelli und D. Oblinger (2004). *Programming shell scripts by demonstration*. In: *Workshop on Supervisory Control of Learning and Adaptive Systems, AAAI 2004, San Jose, CA*.

[Lau et al. 2003] Lau, Tessa, S. A. Wolfman, P. Domingos und D. S. Weld (2003). *Programming by Demonstration Using Version Space Algebra*. Machine Learning, 53(1-2):111–156.

[Lockerd und Breazeal 2004] Lockerd, A. und C. Breazeal (2004). *Tutelage and socially guided robot learning*. In: *Proceedings. 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Bd. 4, S. 3475–3480 vol.4.

[Mahadevan 1996] Mahadevan, Sridhar (1996). *Machine Learning for Robots: A Comparison of Different Paradigms*. In: *Proceedings Workshop*

Towards Real Autonomy, IEEE/RSJ International Conference on Intelligent Robots and Systems.

[Meta Motion 2006] Meta Motion, 268 Bush St. Nr. 1 San Francisco, CA 94104 (2006). *Gypsy Motion Capture System*. Brochure.

[Mitchell 1997] Mitchell, Tom (1997). *Machine Learning*. McGrah-Hill Companies, Inc.

[Mitchell 1982] Mitchell, Tom M. (1982). *Generalization as Search*. Artificial Intelligence, 18(2):203–226.

[Miura et al. 2004] Miura, Jun, Y. Yano, K. Iwase und Y. Shirai (2004). *Task Model-Based Interactive Teaching*. In: *Proc. IROS 2004 Workshop on Issues and Approaches to Task Level Control*.

[Nakaoka et al. 2003] Nakaoka, S., A. Nakazawa, K. Yokoi, H. Hirukawa und K. Ikeuchi (2003). *Generating whole body motions for a biped humanoid robot from captured human dances*. In: *Proceedings if the IEEE International Conference on Robotics and Automation*, Bd. 3, S. 3905–3910.

[Nicolescu und Mataric 2001a] Nicolescu, M. N. und M. J. Mataric (2001a). *Experience-based representation construction: learning from human and robot teachers*. In: *Proceedings of th IEEE/RSJ International Conference on Intelligent Robots and Systems*, Bd. 2, S. 740–745.

[Nicolescu und Mataric 2001b] Nicolescu, M. N. und M. J. Mataric (2001b). *Learning and interacting in human-robot domains*. Systems, Man and Cybernetics, Part A, IEEE Transactions on, 31(5):419–430.

[Nicolescu und Mataric 2004] Nicolescu, Monica und M. Mataric (2004). *Natural methods for robot task learning: Instructive Demonstrations, Generalization and Practice*. Proceedings of the second international joint conference on Autonomous agents and multiagent systems, S. 241–248.

[Nilsson et al. 2005] Nilsson, K., R. Johansson, A. Robertsson, R. Bischoff, T. Brogardh und M. Hägele (2005). *Productive robots and the SMErobot project*. In: *Third Swedish Workshop on Autonomous Robotics, Stockholm, September 1-2, 2005*.

[Onda et al. 1995] Onda, Hiromu, H. Hirukawa, F. Tomita, T. Suehiro und K. Takase (1995). *Assembly motion teaching system using*

position/force simulator - extracting a sequence of contact state transition. In: *Proceedings of the International Conference on Intelligent Robots and Systems-Volume 1*, S. 938–945, Washington, DC, USA. IEEE Computer Society.

[Pardowitz et al. 2005] Pardowitz, M., R. Zöllner und R. Dillmann (2005). *Learning Sequential Constratins of Tasks from User Demonstrations*. In: *IEEE-RAS Intl. Conf. on Humanoid Robots (HUMANOIDS), Tsukuba, Japan*.

[Pardowitz et al. 2006a] Pardowitz, M., R. Zöllner und R. Dillmann (2006a). *Incremental acquisition of task knowledge applying heuristic relevance estimation*. In: *Proceedings of th International Conference on Robotics and Automation (ICRA) 2006*, S. 3011–3016.

[Pardowitz et al. 2006b] Pardowitz, M., R. Zöllner und R. Dillmann (2006b). *Incremental Learning of Task Sequences with Information-Theoretic Metrics*. In: *Proceedings of the European Robotics Symposium (EUROS06)*.

[Pardowitz et al. 2006c] Pardowitz, M., R. Zöllner und R. Dillmann (2006c). *Unsupervised and Incremental Acquisition of and Reasoning on Holistic Task Knowledge for Household Robot Companions*. In: *Proceedings of the 2006 International Robotics Symposium (IROS 2006)*, S. 5060–5065.

[Pardowitz et al. 2006d] Pardowitz, M., R. Zöllner, S. Knoop und R. Dillmann (2006d). *Using Physical Demonstrations, Background Knowledge and Vocal Comments for Task Learning*. In: *Proceedings of the 2006 International Robotics Symposium (IROS 2006)*, S. 322–327.

[Pardowitz et al. 2007] Pardowitz, M., R. Zöllner, S. Knoop und R. Dillmann (2007). *Incremental Learning of Tasks from User Demonstrations, Past Experiences and Vocal Comments*. IEEE Transactions on Systems, Man and Cybernetics, Part B, 37(2):322–332.

[Pawlow 1972] Pawlow, Iwan Petrowitsch (1972). *Die bedingten Reflexe*. Kindler, München.

[Pedrycz und Rocha 1993] Pedrycz, W. und A. Rocha (1993). *Fuzzy-set based models of neurons and knowledge-based networks*. In: *IEEE Transactions on Fuzzy Systems*, Bd. 1(4), S. 254–266.

[Pedrycz 2005] Pedrycz, Witold (2005). *Knowledge-Based Clustering: From Data to Information Granules*. John Wiley & Sons, Inc.

[Piaget 1975] Piaget, Jean (1975). *Nachahmung, Spiel und Traum*. Klett, Stuttgart.

[Pommer et al. 2006] Pommer, T., H. Holzapfel und A. Waibel (2006). *Radid Simulation-Driven Reinforcement Learning of Multimodal Dialog Strategies in Human-Robot Interaction*. In: *International Conference on Speech and Language Processing (INTERSPEECH 2006), Pittsburgh PA, USA*.

[Potkonjak et al. 2001] Potkonjak, V., D. Kostic, S. Tzafestas, M. Popovic, M. Lazarevic und G. Djordevic (2001). *Human-like behavior of robot arms: general considerations and the handwriting task - mathematical description of human-like motion: distributed positioning and virtual fatigue*. In: *Robotics and CIM*, Bd. 17, S. 305–315.

[Riemann und Fleuret 2005] Riemann, R. und F. Fleuret (2005). *Genetik und Persönlichkeit*. In: Hennig, J. und P. Netter, Hrsg.: *Biopsychologische Grundlagen der Persönlichkeit*, S. 539–629. Spektrum, Heidelberg.

[Rittscher et al. 2002] Rittscher, J., A. Blake und S. Roberts (2002). *Towards the automatic analysis of complex human body motions*. Image and Vision Computing, 20(12):905–916.

[Ross 1989] Ross, B. H. (1989). *Some Psychological Results on Case-Based Reasoning*. In: *Proceedings of the Workshop on Case-Based Reasoning (DARPA), Pensacola Beach, FL*.

[S. Knoop 2006] S. Knoop, S. R. Schmidt-Rohr, R. Dillmann (2006). *A Flexible Task Knowledge Representation for Service Robots*. In: *The 9th International Conference on Intelligent Autonomous Systems (IAS-9)*, Kashiwa New Campus, The University of Tokyo, Tokyo, JAPAN.

[Sato et al. 2002] Sato, Yoshihiro, K. Bernardin, H. Kimura und K. Ikeuchi (2002). *Task analysis based on observing hands and ob jects by vision*. In: *Proceedings of the IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, Bd. 1208-1213.

[Saunders et al. 2004] Saunders, J., C. L. Nehaniv und K. Dautenhahn (2004). *An experimental comparison of imitation paradigms used*

in social robotics. In: *13th IEEE International Workshop on Robot and Human Interactive Communication*, S. 691–696.

[Scassellati et al. 2006] Scassellati, B., C. Crick, K. Gold, E. Kim, F. Shic und G. Sun (2006). *Social development*. Computational Intelligence Magazine, IEEE, 1(3):41–47.

[Schaal et al. 2003a] Schaal, S., A. Ijspeert und A. Billard (2003a). *Computational Approaches to Motor Learning by Imitation*. Phil. Trans. Royal Soc. London, (358):537–547.

[Schaal et al. 2003b] Schaal, S., J. Peters, J. Nakanishi und A. Ijspeert (2003b). *Control, Planning, Learning, and Imitation with Dynamic Movement Primitives*. In: *Workshop on Bilateral Paradigms on Humans and Humanoids, IEEE International Conference on Intelligent Robots and Systems*.

[Schaal 1999] Schaal, Stefan (1999). *Is imitation learning the route to humanoid robots?*. Trends in Cognitive Sciences, 3:233–242.

[Schaude 1996] Schaude, Horst (1996). *Dokumentation zu KaVis*. Technischer Bericht, Institut für Prozessrechentechnik, Automation und Robotik, Universität Karlsruhe (TH).

[Sedgewick 1992] Sedgewick, Robert (1992). *Algorithmen in C++*. Addison-Wesley, 9 Aufl.

[Shiratori et al. 2004] Shiratori, Takaaki, A. Nakazawa und K. Ikeuchi (2004). *Detecting Dance Motion Structure through Music Analysis*. Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, S. 857–862.

[Sim et al. 2003] Sim, Siang Kok, K. W. Ong und G. Seet (2003). *A Foundation for Robot Learning*. In: *The Fourth International Conference on Control and Automation*, S. 649–653.

[Simmoudis 1992] Simmoudis, E. (1992). *Using Case-Based Retrieval for Customer Technical Support*. IEEE Expert, 7(5):7–13.

[Spelke et al. 1992] Spelke, Elisabeth, K. Breinlinger, J. Macomber und K. Jacobson (1992). *Origins of knowledge*. Psychological Review, 99(4):605–632.

[Vicon 2006] Vicon (2006). *Vicon Bodybuilder - Flexible Kinematic and Kinetic Modelling Tool*. Brochure.

[Wank et al. 2004] Wank, V., A. Fischer, K. Bös, I. Boesnach, J. Moldenhauer und T. Beth (2004). *Similarities and Varieties in Human Motion Trajectories of Predefined Grasping and Disposing Movements*. In: *IEEE Intl. Conference on Humanoid Robots (HUMAOIDS)*, S. 311–321.

[Warshall 1962] Warshall, S. (1962). *A Theorem on Boolean Matrices*. Journal of the ACM, 9(1):11–12.

[Yeasin und Chaudhuri 2000] Yeasin, M. und S. Chaudhuri (2000). *Towards automatic robot programming: Learning human skill from perceptual data*. IEEE Trans. on Systems Man & Cybernetics-B, 30(1):180–185.

[Zöllner et al. 2004] Zöllner, R., T. Asfour und R. Dillmann (2004). *Programming by Demonstration: Dual-Arm Manipulation Tasks for Humanoid Robots*. In: *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*.

[Zöllner et al. 2001] Zöllner, R., O. Rogalla, R. Dillmann und J. Zöllner (2001). *Dynamic Grasp Recognition Within The Framework Of Programming By Demonstration*. In: *10th IEEE International Workshop on Robot and Human Interactive Communication (Roman)*.

[Zöllner 2005] Zöllner, Raoul-Daniel (2005). *Erlernen zweihändiger feinmotorischer Handhabungen*. Doktorarbeit, Fakultät für Informatik der Universität Karlsruhe (TH).

[Zöllner et al. 2005] Zöllner, Raoul-Daniel, M. Pardowitz, S. Knoop und R. Dillmann (2005). *Towards Cognitive Robots: Building Hierarchical Task Representations of Manipulations from Human Demonstration*. In: *Proc. of the Intl. Conf. on Robotics and Automation (ICRA), Barcelona, Spain*.