Repräsentation operationaler Begriffe zum Lernen aus Roboter-Sensordaten

Stefan Sklorz

betreut von
Prof. Dr. Katharina Morik
und
Dr. Joachim Hertzberg

31. Januar 1995

Diplomarbeit am Fachbereich Informatik an der
Universität Dortmund

Zusammenfassung

Die vorliegende Arbeit ist im Rahmen des Projektes B-Learn II (ESPRIT P7274) angefertigt worden, dessen leitende Idee die Integration maschineller Lernverfahren in reale Robotersysteme im Hinblick auf die Entwicklung zukünftiger Robotergenerationen ist. Ein Ziel innerhalb des Projektes ist es, eine Verbindung zwischen den Handlungen eines mobilen Roboters und den mit ihnen einhergehenden Wahrnehmungen herzustellen, um auf diesem Wege eine Begriffsoperationalisierung zu erzielen.

Während mit Beginn der vorliegenden Arbeit bereits ein erster, erfolgversprechender Ansatz zum Lernen von an Handlungen orientierten Wahrnehmungsmerkmalen entwickelt wurde, besteht der Beitrag dieser Arbeit in der Gewinnung von Handlungsmerkmalen aus Roboter-Sensordaten, ihrer Zusammenführung mit dem bestehenden Konzept handlungsorientierter Wahrnehmungsmerkmale und der Vorgabe einer Begriffshierarchie, an deren Spitze *operationale Begriffe* stehen.

Operationale Begriffe ermöglichen einerseits eine aufgabenbezogene Roboterprogrammierung und können andererseits zur Erklärung von durchgeführten Roboteraktionen verwendet werden. Sie erlauben eine Handlungsplanung ebenso wie eine Kontrolle der Handlungsausführung.

Die in dieser Arbeit vorgeschlagene Repräsentation operationaler Begriffe ermöglicht ferner eine Formulierung von Lernaufgaben zu ihrer Gewinnung. Exemplarisch wird in der vorliegenden Arbeit gezeigt, daß diese Lernaufgaben für logikbasierte, induktive Lernverfahren handhabbar sind, und daß die vorgeschlagene Repräsentation somit zum Lernen operationaler Begriffe aus Roboter-Sensordaten geeignet ist.

Danksagung

Diese Arbeit fertigte ich während meiner Tätigkeit als studentische Hilfskraft am Lehrstuhl für Künstliche Intelligenz des Fachbereichs Informatik an. Der Lehrstuhlinhaberin *Katharina Morik* möchte ich aufrichtig danken für die praktische Ausbildung, die ich an ihrem Lehrstuhl erhalten habe und für den Einblick in die Bearbeitung von Forschungsthemen, der mir an ihrem Lehrstuhl großzügig gewährt wurde. Darüberhinaus gab Sie mir stets wichtige thematische Anregungen, über die ich sehr dankbar bin. Mein aufrichtiger Dank gilt ebenfalls *Joachim Hertzberg*, der kurzfristig bereit war, meine Diplomarbeit zu begutachten und mir nützliche Hinweise für das Schreiben der Arbeit gab.

Ganz besonders danken möchte ich *Volker Klingspor*, der mich während meiner Arbeit vor allem unterstützt hat. Seine fortwährende Diskussionsbereitschaft, seine vielfältigen Anregungen für meine Arbeit und der außerordentliche Einsatz, mit dem er mir unser gemeinsames Arbeitsthema nahebrachte, trugen zu einer freundschaftlichen Arbeitsatmosphäre bei, die ich sehr vermissen werde.

Mein besonders herzlicher Dank gilt *Monika Burg*, die mir trotz ihrer enormen Arbeitsbelastung stets als Ratgeberin zur Seite stand. Zuletzt, aber nicht weniger herzlich, bedanke ich mich bei meinen Eltern, *Elfie* und *Joachim Sklorz*, die mir mein Studium ermöglichten und mir Ihre Unterstützung in jeglicher Hinsicht niemals verwehrten.

Inhaltsverzeichnis

Kapitel 1

Motivation

In den letzten zehn Jahren hat die Anzahl der weltweit in der Industrie eingesetzten Robotersysteme enorm zugenommen. So hat sich beispielsweise die Anzahl der Ende 1982 weltweit im industriellen Einsatz befindlichen Systeme von 55 000 [W.R.S.D., 1983] bis Ende 1988 auf 200 000 [B.R.A., 1989] beinahe vervierfacht. Die weitaus meisten dieser Systeme werden heute in der Produktion eingesetzt.

Ihr Einsatz wurde zunächst durch die Vorstellung motiviert, eine Erhöhung der Produktivität bei gleichzeitiger Senkung der Kosten zu erzielen. Veränderte Marktanforderungen führten inzwischen jedoch zu gewandelten Zielvorstellungen im Zusammenhang mit dem Einsatz von Industrierobotern. So wird die Wirtschaftlichkeit in der industriellen Produktion heute nicht mehr allein durch die Produktivität, sondern in zunehmendem Maße auch durch die Flexibilität der Produktion bestimmt. Eine sich ständig vergrößernde Produktvielfalt und eine bedarfsorientierte Produktion mit geringeren Stückzahlen bedingt, daß von den eingesetzten Maschinen immer kürzere Einrichtungs- und Umrüstungszeiten abverlangt werden müssen, um noch wirtschaftlich produzieren zu können.

Die heutzutage im Einsatz befindlichen Roboter der sogenannten ersten und zweiten Generation [Levi, 1988] (S.12) können die wachsenden Anforderungen nur noch bedingt erfüllen. Bei ihnen handelt es sich um starre Systeme, die neben einer aufgabenspezifisch präparierten Umgebung eine aufwendige, roboterspezifische Programmierung erfordern. Zukünftige Robotersysteme sollten sich hingegen durch ein erhöhtes Maß an Flexibilität, Anpassungsfähigkeit und Autonomie auszeichnen. Insbesondere die Forderung nach Autonomie setzt Eigenschaften voraus, von deren Besitz die heutigen Robotersysteme noch weit entfernt sind. [Birk und Kelly, 1981] zählen zu diesen Eigenschaften:

- die Fähigkeit zur Kommunikation mit der Umwelt,
- die Fähigkeit, für vorgegebene Aufträge selbständig Handlungspläne zu generieren,

- die Fähigkeit zur automatischen Ausführung und Überwachung von Handlungsplänen,
- die Fähigkeit zum Verständnis der Umgebung durch den Gebrauch von Sensoren und Modellen und
- die Fähigkeit auf unvorhergesehene Situationen reagieren zu können.

Mit der Fähigkeit zur selbständigen Generierung, Ausführung und Überwachung von Handlungsplänen geht die Möglichkeit einer aufgabenbezogenen Programmierung mit Hilfe einer Kommandosprache einher, die sich auf abstrakte Anweisungen — *gehe durch die Tür* oder *schraube Werkstück A an Werkstück B* — beschränken kann. Anweisungen dieser Art können dann vom System selbst in hierzu notwendige Teilaufgaben und schließlich in durchzuführende Handlungen mit dabei wahrzunehmenden Merkmalen zerlegt werden. Die oben angesprochene aufwendige roboterspezifische Programmierung für eine aufgabenspezifisch präparierte Umgebung entfällt damit. Für die Ausführung und die Überwachung von Handlungen ist dabei von essentieller Bedeutung, daß Wahrnehmungen mit Handlungen in Verbindung gesetzt werden [Brady, 1985], [Klingspor und Morik, 1995].

Die Entwicklung von derart leistungsfähigen und auch praktisch einsetzbaren Robotersystemen der sogenanten dritten Generation steht noch an ihrem Anfang und befindet sich zur Zeit in der Phase der Grundlagenerforschung. [Levi, 1988] (S.20ff) macht deutlich, daß Techniken aus dem Bereich der künstlichen Intelligenz eine Schlüsselfunktion für ihre Realisierung einnehmen könnten. In diesem Forschungsgebiet sind bereits grundlegende Konzepte zum Erwerb, zur Darstellung und zur Manipulation von Wissen sowie zur Bearbeitung von Planungsaufgaben erarbeitet worden. Gelingt es mit Hilfe dieser Konzepte eine Brücke zwischen Handlungen und Wahrnehmungen zu errichten, so wird der Weg hin zu einsatzfähigen Robotern der dritten Generation weitaus weniger beschwerlich sein.

Im Mittelpunkt vielfältiger Forschungsaktivitäten mit dieser Ausrichtung stehen gegenwärtig mobile Robotersysteme. [de Velde, 1991] sieht in ihnen ein archetypisches Beispiel für autonome Systeme. Ausgestattet mit Sensoren können sie sich in einer natürlichen Umgebung aus eigener Kraft umherbewegen und mit der Umgebung interagieren, um ein gestelltes Ziel zu erreichen. Autonomie muß sich hier also nicht allein auf die zur Fortbewegung notwendigen Ressourcen beschränken, sondern kann darüber hinaus auch naheliegend auf die eigenverantwortliche Planung, Durchführung und Überwachung von Handlungsplänen im Sinne von [Birk und Kelly, 1981] ausgedehnt werden. Die Interaktion mit der Umwelt kann neben der Möglichkeit des Umganges mit unvorhersehbaren Ereignissen auch die oben angesprochene Möglichkeit einer aufgabenbezogenen Programmierung mit einschließen. Mobile Robotersysteme stellen demnach einen geeigneten Forschungsgegenstand zur Beantwortung grundlegender Fragestellungen im Hinblick auf zukünftige Robotergenerationen dar.

1.1 Problemstellung

Die vorliegende Arbeit ist im Rahmen des Projektes B-Learn II (Behavioural Learning II, ESPRIT P7274) angefertigt worden. Die leitende Idee des Dortmunder Arbeitsanteiles innerhalb des Projektes ist die Überprüfung der Tauglichkeit einer Integration logikbasierter induktiver Lernverfahren in reale Robotersysteme im Hinblick auf die Entwicklung zukünftiger Robotergenerationen.

Mit dieser Zielsetzung ist ein zweistufiges Potential verbunden. Das im vorangegangenen Abschnitt bereits angesprochene Potential für die Robotik liegt in der Aussicht, eine einfachere und flexiblere Handhabung von Robotersystemen zu erreichen. Das Potential für das maschinelle Lernen liegt in der Möglichkeit, eine Verbindung zwischen Handlungen eines Roboters einerseits und den damit einhergehenden Wahrnehmungen andererseits im Sinne von [Brady, 1985] zu schaffen. Wird diese Verbindung durch den Aufbau einer integrierenden Repräsentation in einer Weise gestaltet, die es erlaubt, Handlungen in Abhängigkeit von Wahrnehmungen zu bestimmen, auszuführen und während ihrer Durchführung zu kontrollieren, so könnte auf diesem Wege eine Begriffsoperationalisierung erzielt werden. Mit Hilfe *operationaler Begriffe* sollte es dann möglich sein, ein Robotersystem mit den nach [Birk und Kelly, 1981] notwendigen Eigenschaften für Autonomie auszustatten und insbesondere eine aufgabenbezogene Programmierung zuzulassen. Die für maschinelle Lernverfahren notwendige Repräsentation von Beispielen sollte im Zuge einer Verbindung von Handlung und Wahrnehmung nicht mehr vorab definiert werden müssen, sondern direkt aus den Sensordaten des Roboters während der Durchführung von Handlungen gelernt werden können. Als Basis für die Untersuchungen dient der in Karlsruhe entwickelte mobile Roboter PRIAMOS [Dillmann *et al.*, 1993].

Ausgehend von diesen Überlegungen ist es ein erklärtes Ziel des Projektes, aus den während einer Fahrt aufgenommenen Sensordaten von PRIAMOS, unter Berücksichtigung der dabei durchgeführten Handlungen, abstrakte Wahrnehmungs- und Handlungsmerkmale mit Hilfe induktiver Lernverfahren zu gewinnen und über den Aufbau einer hierarchischen Begriffsstruktur miteinander zu verbinden. Mit Beginn der vorliegenden Arbeit lag hierzu bereits ein erster, erfolgversprechender Ansatz zum Lernen einer Hierarchie von an Handlungen orientierten Wahrnehmungsmerkmalen vor [Morik und Rieger, 1993]. Ihre Zusammenführung mit geeigneten Handlungsmerkmalen zu operationalen Begriffe stand jedoch noch aus.

Vor diesem Hintergrund läßt sich die generelle Problemstellung der vorliegenden Arbeit wie folgt formulieren:

Wie können die aus den Sensordaten eines mobilen Roboters gelernten Wahrnehmungsmerkmale mit seinen während der Aufnahme der Sensorinformationen durchgeführten Handlungen in einer Weise in Verbindung gebracht werden, die es ermöglicht, operationale Begriffe zu lernen?

Zur Bearbeitung dieser Problemstellung müssen u.a. folgende Fragestellungen untersucht werden:

- Wie kann ein Konzept für den Einsatz operationaler Begriffe in mobilen Robotersystemen gestaltet werden?
- Welchen Anforderungen müssen operationale Begriffe hierzu genügen?
- Was sind geeignete Handlungsmerkmale?
- Wie können Handlungsmerkmale gewonnen werden?
- Wie können Handlungsmerkmale mit Wahrnehmungsmerkmalen in Verbindung gebracht werden?
- Wie können aus dieser Verbindung operationale Begriffe aufgebaut werden?
- Ist die Lern- und Anwendbarkeit operationaler Begriffe gewährleistet?
- Wie kann die Verwendung operationaler Begriffe zur Handlungsplanung gestaltet werden?

1.2 Gang der Untersuchung

Zur Erarbeitung der vorgenannten Problemstellung ist im Rahmen der vorliegenden Arbeit folgende Vorgehensweise gewählt worden:

Im **Kapitel 2** *Rahmenbedingungen* werden die Rahmenbedingungen erörtert, in die sich die weiteren Teile der vorliegenden Arbeit einfügen. Hierzu gehören vor allem die von [Morik und Rieger, 1993] vorgestellten Ergebnisse zur Repräsentation von Wahrnehmungsmerkmalen und die von der Universität Karlsruhe gelieferten Sensordaten von PRIAMOS [Dillmann *et al.*, 1993] bzw. die Daten eines PRIAMOS-Simulationssystems [Janakiraman und Wallner, 1992]. Neben einer einheitlichen, zusammenfassenden Darstellung dieser Arbeiten enthält dieses Kapitel aber auch eigene Arbeitsergebnisse. Hierzu zählt ein bei der Bildung von Wahrnehmungsmerkmalen eingesetzter Algorithmus zur Festlegung von Sensorklassen sowie eine Erörterung von grundlegenden Betrachtungen zur Realisierung einer eigenen Robotersimulation, auf denen aufbauend im Anhang C.2 eine algorithmische Beschreibung eines einfachen Simulationssystems skizziert wird.

Im **Kapitel 3** *Aufbau einer Begriffshierarchie* wird schließlich unter Berücksichtigung der im Kapitel 2 vorgestellten Arbeiten ein Ansatz vorgeschlagen, wie eine Verbindung zwischen Handlungen eines mobilen Roboters und seinen mit den Handlungen einhergehenden Wahrnehmungen auf dem Wege zu einer Begriffsoperationalisierung hergestellt werden kann. Im einzelnen enthält dieses Kapitels:

- eine Definition von Basis-Handlungsmerkmalen,
- die Beschreibung eines Verfahren zur inkrementellen Berechnung von Basis-Handlungsmerkmalen aus den von PRIAMOS zur Verfügung gestellten Daten,
- einen Vorschlag für eine Repräsentation, die Wahrnehmungs- und Handlungsmerkmale zusammenführt,
- einen Vorschlag für eine Beschreibung operationaler Begriffe mit ihrer Hilfe,
- eine Diskussion der Anforderungen, die von einer Repräsentation operationaler Begriffe im Hinblick auf die oben skizzierte Problemstellung zu erfüllen sind, sowie
- die Bereitstellung eines Konzeptes für den Einsatz operationaler Begriffe in mobilen Robotersystemen.

Im **Kapitel 4** *Unterstützung der Beispielgenerierung* steht die zum Lernen operationaler Begriffe notwendige Beispielgenerierung im Vordergrund. Die mit diesem Kapitel verfolgte Zielsetzung ist es dabei jedoch nicht, einzelne Algorithmen zur Beispielgenerierung vorzuschlagen, sondern eine grafischen Benutzeroberfläche vorzustellen, die zur Unterstützung der Beispielgenerierung einsetzbar ist. Im einzelnen wird in diesem Kapitel erörtert:

- warum eine Unterstützung der Beispielgenerierung notwendig ist,
- wie diese Unterstützung gestaltet werden kann,
- welche funktionalen Aspekte von der bislang realisierten Oberfläche abgedeckt werden, und
- welche Weiterentwicklungen der Oberfläche denkbar sind.

Im **Kapitel 5** *Lernen der Begriffshierarchie* wird gezeigt, daß die gewählte Repräsentation operationaler Begriffe eine Formulierung von Lernaufgaben erlaubt. Anhand einiger Lernexperimente wird exemplarisch vorgeführt, daß diese Lernaufgaben für logikbasierte induktive Lernverfahren handhabbar sind, und daß die vorgeschlagene Repräsentation operationaler Begriffe somit zum Lernen aus Roboter-Sensordaten geeignet ist. Im einzelnen enthält dieses Kapitel:

- eine Charakterisierung des eingesetzten Lernverfahrens, sowie seine Einordnung in den Rahmen der induktiven logischen Programmierung,
- eine Erörterung der für die durchgeführten Lernexperimente ausgewählten Roboterfahrten,
- eine Beschreibung des den Lernexperimenten zugrundegelegten Experimentdesigns sowie

- eine ausführliche Diskussion der erzielten Lernresultate.

Im **Kapitel 6** *Perspektiven für die Handlungsplanung* werden erste Ideen hinsichtlich der Verwendbarkeit operationaler Begriffe zur Handlungsplanung erörtert.

Im **Kapitel 7** *Diskussion* wird anhand einer kurzen Darstellung zweier grundlegender und entgegengesetzter Positionen bei der Entwicklung mobiler Systeme gezeigt, daß die an ihnen häufig geäußerte Kritik auf den Einsatz operationaler Begriffe in mobilen Robotersystemen nicht zutrifft. Diese Diskussion zeigt zugleich die Vorzüge, die mit dem Einsatz operationaler Begriffe verbunden sind.

Im **Kapitel 8** *Zusammenfassung und Ausblick* wird die vorliegende Arbeit mit einer kurzen Zusammenfassung der erzielten Ergebnisse sowie der zukünftigen Arbeitsthemen abgeschlossen werden.

Kapitel 2

Rahmenbedingungen

Mit diesem Kapitel wird das Ziel verfolgt, die Rahmenbedingungen vorzustellen, in die sich die weiteren Teile der vorliegenden Arbeit einfügen werden. Hierzu ist es in zwei Hauptteile gegliedert. Thema des ersten Teiles (Abschnitt 2.1) sind die der Arbeit zugrundeliegenden Sensordaten des Karlsruher Roboters PRIAMOS. Hierzu werden im Abschnitt 2.1.1 einige relevante technische Daten und die Eigenschaften von PRIAMOS vorgestellt, während Abschnitt 2.1.2 die durch das Karlsruher PRIAMOS-Simulationssystem zur Verfügung gestellten Daten erörtert . Im Abschnitt 2.1.3 hingegen werden grundlegende Betrachtungen zur Realisierung einer eigenen Robotersimulation beschrieben, die die Basis der im Anhang C.2 skizzierten algorithmischen Beschreibung eines einfachen Simulationssystems bilden. Thema des zweiten Teiles (Abschnitt 2.2) ist die Überführung der im Abschnitt 2.1 beschriebenen Sensordaten in eine symbolische Repräsentation von Wahrnehmungsmerkmalen. Hierzu wird im Abschnitt 2.2.2 die Repräsentation von Basis-Wahrnehmungsmerkmalen vorgestellt, auf der aufbauend im Abschnitt 2.2.3 die Bildung abstrakterer Wahrnehmungsmerkmale erfolgt. Abschnitt 2.2.1 erläutert die für die Bildung von Wahrnehmungsmerkmalen notwendige Festlegung von Sensorklassen, über die ein zeitgleiches oder zeitversetztes Auftreten von Meßwertmustern einzelner Sensoren charakterisiert werden kann. Das Kapitel wird schließlich mit einer kurzen Zusammenfassung im Abschnitt 2.3 abgeschlossen.

2.1 Sensordaten eines mobilen Roboters

Im Rahmen des Projektes B-Learn II arbeitet die Dortmunder Arbeitsgruppe mit Sensormeßwerten, die durch den Roboter PRIAMOS der Universität Karlsruhe [Dillmann *et al.*, 1993] bzw. durch ein PRIAMOS-Simulationssystem [Janakiraman und Wallner, 1992] erzeugt werden. Während im Abschnitt 2.1.2 die Form der aus Karlsruhe zur Verfügung gestellten Daten näher betrachtet wird, dient Abschnitt 2.1.1 der Beschreibung von PRIAMOS (siehe auch Abbildung im Anhang A) und seiner für den Aufbau operationaler Begriffe relevanten Eigenschaften sowie der Angabe einiger technischer

Daten. Im Abschnitt 2.1.3 werden grundlegende Betrachtungen zur Realisierung einer eigenen Robotersimulation erörtert.

2.1.1 Roboter Priamos

Priamos ist ein mobiles System, welches mit vier separat angesteuerten Mecanum-Rädern ausgestattet ist, die dem System drei Freiheitsgrade der Bewegung ermöglichen. Im einzelnen sind dies eine Bewegung in longitudinaler und transversaler Richtung sowie eine Drehung beliebiger Richtung um das Zentrum des Vehikels. Vorwärtsbewegung und Drehung können dabei simultan erfolgen. Priamos ist demnach in der Lage, sich in beliebiger Richtung bei beliebiger Raumorientierung fortzubewegen, was nahelegt, dem Roboter perse kein festes *vorne* (und damit *hinten*, *rechts* und *links*) zuzuordnen, sondern *vorne* stets in Bewegungsrichtung anzunehmen. Im Abschnitt 3.2 wird auf diese Überlegung näher eingegangen und gezeigt, daß sie sich in der Repräsentation der Basis-Handlungsmerkmale sinnvoll berücksichtigen läßt. Bei einem Gewicht von 300 kg, einer Länge von 650 mm und einer Höhe von 750 mm ist Priamos in der Lage, sich mit einer maximalen Geschwindigkeit von 3 km/h fortzubewegen.

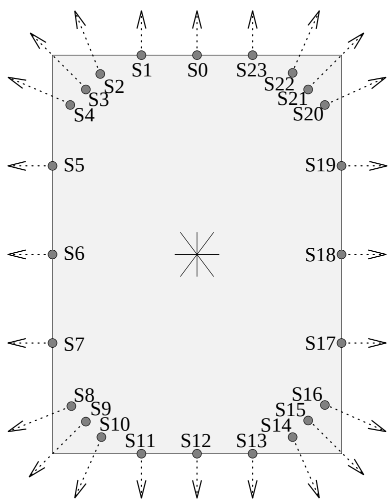


Abbildung 2.1: Anordnung der Ultraschallsensoren von Priamos

Priamos ist mit 24 Ultraschallsensoren ausgestattet, die in Dreier-Formation an den Seiten sowie an den Ecken jeweils auf gleicher Höhe montiert sind. Das in Karslruhe entwickelte zugehörige Sensorkontroll- und Signalverarbeitungssystem ist in der Lage, die Signale aller 24 Sensoren bis zu fünfmal in der Sekunde abzusenden, wiederaufzunehmen und zu verarbeiten. Abbildung 2.1 zeigt schematisch die Anordnung der Sensoren, die Richtungen, in der die Ultraschallkeulen gesendet werden, sowie die den Sensoren zur Identifizierung zugeordneten Bezeichner.

2.1.2 Sensordaten der Karlsruher PRIAMOS-Simulation

An der Universität Karlsruhe ist, insbesondere zum Zwecke des Studiums des Verhaltens der verwendeten Ultraschallsensoren unter verschiedenen Umweltbedingungen, ein Simulationssystem für PRIAMOS entwickelt worden [Janakiraman und Wallner, 1992]. Mit Hilfe dieses Systems lassen sich beliebige Fahrten von PRIAMOS durch frei definierbare virtuelle Umgebungen (z.B. Büroräume, Lagerhallen etc.) derart simulieren, daß die in der Simulation berechneten Signalwerte der Sensoren denen bei einer realen Fahrt aufgenommenen Meßwerten nahe kommen. Das System ist hierzu unter anderem in der Lage :

- die Schallbündelung der eingesetzten Ultraschallsensoren,
- die Reflexionseigenschaften der Umgebung und
- evtl. auftretende Mehrfachreflexionen

zu simulieren. So besitzt die ausgesendete Schallkeule gängiger Ultraschallsensoren üblicherweise einen Öffnungswinkel von 5°–40° [Knieriemen, 1991] (S.27). Ein zu kleiner Öffnungswinkel führt zur Beschallung eines nur kleinen Ausschnittes der Umgebung, so daß der Sensor genau auf mögliche anzupeilende Objekte ausgerichtet sein muß, um noch eine Reflexion des ausgesendeten Signals zu empfangen. Ein zu großer Öffnungswinkel hingegen macht unter Umständen eine Positionsbestimmung des Reflexionsortes unmöglich, nämlich dann, wenn mehrere gleichentfernte Reflexionsorte von der Schallkeule erfaßt werden. Wird ein Objekt von der Schallkeule getroffen, so entscheiden oftmals seine Reflexionseigenschaften darüber, ob das reflektierte Signal vom Sensor zur Entfernungs- bzw. Positionsbestimmung wiederaufgenommen werden kann. Dies ist zum Beispiel der Fall, wenn das angepeilte Objekt eine Raumorientierung besitzt, die nicht genau senkrecht zur Abstrahlrichtung des Sensors liegt. Die in der Praxis oftmals auftretenden Mehrfachreflexionen liegen vor, wenn die von den Sensoren ausgestrahlten Schallsignale über mehrere Objekte reflektiert zu den Sensoren zurückgelangen. Die Konsequenz ist, daß anhand der Entfernungsmessungen keinerlei verwertbare Positionsinformationen über mögliche Objekte zu ermitteln sind.

Zur Durchführung eines Simulationslaufes wird einerseits die Beschreibung der gewünschten Umgebung und andererseits die Beschreibung der von PRIAMOS durchzuführenden Fahrt in dieser Umgebung benötigt. Die Beschreibung der Umgebung setzt sich aus den Beschreibungen einer beliebigen Anzahl von Objekten zusammen. Jedes Objekt ist dabei durch eine Menge flächendefinierende Polygonzüge charakterisiert, die wiederum durch die Angabe der kartesischen Koordinaten ihrer Eckpunkte dargestellt sind. Jeder Fläche ist weiterhin ein Winkel, der *Reflexionswinkel*, zugeordnet, der die maximal zulässige Abweichung von einer im Idealfall senkrecht zur Abstrahlrichtung der Schallkeule verlaufenden Raumorientierung der betrachteten

Fläche angibt. Für Abweichungen unterhalb der maximal zulässigen Abweichung wird angenommen, daß sie eine Reflexion des Schallsignals zurück zum Sensor ermöglichen. Zur Beschreibung der zu simulierenden Fahrt des Roboters sind Positionen innerhalb der simulierten Umgebung anzugeben, an denen das Sensorkontroll- und Signalverarbeitungssystem von PRIAMOS eine Ultraschallmeßung durchführt. Zu jeder dieser Roboterpositionen, die durch Angabe ihrer kartesischen Koordinaten festgelegt werden, gehört die Festlegung der Orientierung, die der Roboter an dieser Position im Raum einnimmt. Die *Roboterorientierung* (Definition 2.1) ist hierzu definiert als Winkel zwischen der x-Achse des zugrundegelegten Koordinatensystems und der Sensor S_0 und die Roboterposition verbindenden Geraden.[1]

Die Reihenfolge der angegebenen Roboterpositionen wird als in zeitlicher Abfolge vom Roboter in der Simulation angefahrener Positionen interpretiert, so daß jeder Roboterposition ein *diskreter Zeitpunkt der Simulation* zugeordnet ist. Zu jedem Zeitpunkt der Simulation findet eine Sensormessung der 24 eingesetzten Sensoren statt. Bei einer vorher festgelegten Anzahl von Sensormessungen pro Sekunde läßt sich somit die Geschwindigkeit, mit der sich PRIAMOS in der Simulation bewegt, über verschieden groß gewählte Abstände zeitlich aufeinanderfolgender Roboterpositionen variieren. Drehungen des Roboters an einer festen Position werden hingegen über die Zuordnung mehrerer Roboterorientierungen zu dieser Position beschrieben. Analog zu den Bewegungsgeschwindigkeiten können die Drehgeschwindigkeiten dann wieder über die gewählte Differenz zwischen den zugehörigen Roboterorientierungen festgelegt werden.

Definition 2.1 (Roboterorientierung RAlpha) *Die Roboterorientierung zu einem bestimmten Zeitpunkt t der Simulation ist definiert als Winkel zwischen der x-Achse des der Simulation zugrundeliegenden Koordinatensystems und der die Sensorposition von Sensor S_0 zum Zeitpunkt t und die Roboterposition zum Zeitpunkt t verbindenden Geraden.*

Definition 2.2 (Sensororientierung SAlpha) *Die Sensororientierung eines Sensors des Roboters zu einem bestimmten Zeitpunkt t der Simulation ist definiert als Winkel zwischen der x-Achse des der Simulation zugrundeliegenden Koordinatensystems und der die Sensorposition zum Zeitpunkt t und die Roboterposition zum Zeitpunkt t verbindenden Geraden plus einem für jeden Sensor definierten positiven oder negativen Zahlenwert.*

Die Sensororientierung charakterisiert die Abstrahlrichtung des betrachteten Sensors. Insbesondere entspricht die Sensororientierung von Sensor S_0 der Roboterorientierung, da der für Sensor S_0 festgelegte zu addierende Wert Null ist.

Abbildung 2.2 zeigt ein Beispiel für die Definition einer Umgebung mit zugehöriger Roboterfahrt. Die Umgebung besteht aus zwei Objekten, einem

[1] Die dritte Dimension kann dabei außer Acht gelassen werden, da sich PRIAMOS mit seinen Mecanum-Rädern nur auf ebenem, glatten Untergrund bewegen kann.

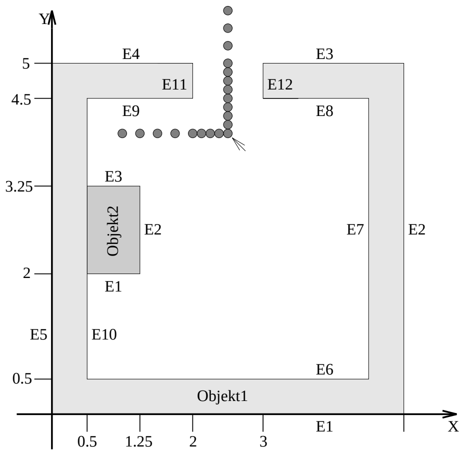


Abbildung 2.2: Umgebungsbeispiel mit zugehöriger Roboterfahrt

Raum und einem Schrank. Sie wird der Einfachheit halber senkrecht von oben betrachtet, so daß die objektbildenden Flächen als Kanten dargestellt sind. Jedem Objekt und jeder Kante ist dabei ein eindeutiger Bezeichner zur Identifizierung zugeordnet. An der markierten Roboterposition findet eine Drehung des Roboters statt, die durch mehrere dieser Position zugeordnete Roboterorientierungen charakterisiert ist. Die Tabellen B.1 und B.2 im Anhang B enthalten eine Beschreibung dieser Szenerie im oben angegebenen Format.

Die Durchführung eines Simulationsexperimentes mit Hilfe der Karlsruher PRIAMOS-Simulation (und in gleicher Weise das real eingesetzte Robotersystem) liefert nun für jede Roboterfahrt eine enorme Menge an Daten. Diese Daten bilden die Basis zum Aufbau und zum Lernen operationaler Begriffe aus Robotersensordaten. Um mit ihnen im Kontext dieser Arbeit arbeiten zu können, werden sie in Form logischer Fakten repräsentiert.

Definition 2.3 (m/13-Fakten) *Ein logischer Fakt der Form :*

$$m(Trc, T, S, Dist, SAlpha, SX, SY, SZ, PX, PY, PZ, O_i, E_i),$$

wird als zu einer Roboterfahrt korrespondierender m/13-Fakt bezeichnet, genau dann, wenn der mit S bezeichnete Sensor des Roboters, zum mit T bezeichneten Zeitpunkt der Fahrt, eine durch SAlpha beschriebene Sensororientierung besitzt und den durch die Koordinaten PX, PY, PZ beschriebenen Punkt der Umgebung mit Abstand Dist von der durch die Koordinaten SX, SY, SZ beschriebenen Sensorposition zum Zeitpunkt T anmißt, und weiterhin der angemessene Punkt ein Punkt der Fläche E_i des Objektes O_i ist. Trc ist dabei eine beliebige, der Identifizierung der Roboterfahrt dienende Zeichenfolge.[2]

[2] Die Bezeichnung *Trc* ist dabei eine mnemotechnische Abkürzung vom engl. *trace*.

Definition 2.4 (rp/5-Fakten) *Ein logischer Fakt der Form :*

$$rp(Trc, T, RX, RY, RAlpha),$$

wird als zu einer Roboterfahrt korrespondierender rp/5-Fakt bezeichnet, genau dann, wenn der Roboter sich zum mit T bezeichneten Zeitpunkt der Simulation an dem durch die Koordinaten RX, RY beschriebenen Punkt der Umgebung befindet und eine durch RAlpha beschriebene Roboterorientierung besitzt. Trc ist dabei eine beliebige, der Identifizierung der Roboterfahrt dienende Zeichenfolge.

2.1.3 Ideen für eine eigene Roboter-Simulation

Ausgangspunkt der im folgenden Abschnitt ausgeführten Überlegungen ist die ursprüngliche Idee der Dortmunder Arbeitsgruppe in BLearn II, eine Robotersimulation vor Ort zur Verfügung zu stellen, die einerseits einfach genug ist, um sie in relativ kurzer Implementierungszeit realisieren bzw. erweitern zu können, und andererseits die Möglichkeit bietet, mit ihr kurzfristig Simulationsdaten erzeugen zu können, die als Basisdaten für eingeschränkte Experimente nutzbar sind. Motiviert wird diese Idee durch die doch verhältnismäßig lange Zeitspanne, die zwischen der Anforderung von Simulationsdaten einer PRIAMOS-Fahrt und der Verfügbarkeit der Daten in Dortmund liegt. Vor diesem Hintergrund ist klar, daß in diesem Abschnitt keineswegs das Ziel verfolgt wird, Grundlagen eines der Karlsruher PRIAMOS-Simulation ebenbürtigen Systems zu legen, sondern vielmehr grundlegende Ideen und notwendige Berechnungen für die Entwicklung einer stark vereinfachenden Simulationsumgebung bereitzustellen. So vereinfacht die im Anhang C.2 vorgestellte algorithmische Beschreibung einer Robotersimulation[3] gegenüber einem auf Realitätsnähe ausgelegten System in vielerlei Hinsicht. Hierzu zählt zum Beispiel:

- Die Simulation spielt sich in einem zweidimensionalen Raum ab.
- Die Sensoren werden als ideale Entfernungsmesser betrachtet, die bis zu einer vorgegebenen Distanz immer exakte Daten liefern.
- Beschleunigungs- und Verzögerungseffekte bei Roboterbewegungen werden nicht berücksichtigt.

Die im folgenden aufgestellten Sätze und Definitionen bilden die Basis für dieses Simulationssystem. Entsprechende Querverweise sind in die algorithmische Beschreibung im Anhang C.2 aufgenommen. Um eine bessere Lesbarkeit zu gewährleisten, sind Beweise aus dem laufenden Text genommen worden. Falls für die Beweise nicht ausdrücklich Literaturhinweise angegeben

[3] Zum Bedauern des Autors ist diese Robotersimulation nie implementiert worden, da mit dem unerwartet gemachten Angebot das Karlsruher PRIAMOS-Simulationssystem in Dortmund zu installieren kein Bedarf mehr danach bestand. Bis heute ist es jedoch bei einem bloßen Angebot geblieben.

sind, gehen sie ausnahmslos auf den Autor zurück und können im Anhang C.1 nachgelesen werden.[4]

Ausgangspunkt der nachfolgenden Betrachtungen ist die Vorstellung, daß das zu simulierende Szenario zum einen durch eine Menge $\mathcal{G}$ von Punkte-Paaren und zum anderen durch eine Folge $\mathcal{R}$ von Punkte-Paaren gegeben ist. Die Punkte-Paare der Menge $\mathcal{G}$ beschreiben dabei Strecken, die ihrerseits die zweidimensionale Umgebung des zu simulierenden Szenarios charakterisieren. $\mathcal{R}$ definiert hingegen einen Streckenzug durch diese Umgebung, der den durch den Roboter zurückgelegten Weg repräsentiert. Alle Teilstrecken des Streckenzuges sind gerichtet, wobei eine Orientierung vom ersten genannten zum zweiten genannten Punkt eines Punkte-Paares angenommen wird. Auf eine Darstellung von Lösungsmöglichkeiten des Sichtbarkeitsproblems (siehe hierzu z.B. [Guibas und Yao, 1980]) ist verzichtet worden, da der Aufwand von Algorithmen zur Erkennung von verdeckten Strecken erst bei einer Menge $\mathcal{G}$ lohnend ist, die in ihrer Größe jenseits der geplanten Simulationen liegt. Zur Notation ist zu bemerken, daß Tupel des $\mathbb{R}^2$ durch Kleinbuchstaben sowohl mit überschriebenem Kreis als auch überschriebenem Pfeil gekennzeichnet sind. Hierdurch wird zum Ausdruck gebracht, daß im ersten Fall über die Punkte der Ebene und im zweiten Fall über die diese Punkte beschreibenden Vektoren geredet wird.

Für $\vec{u}, \vec{v}, \mathring{u} \in \mathbb{R}^n$ und $\lambda \in \mathbb{R}$ beschreibt die Form $\vec{u} = \mathring{u} + \lambda\vec{v}$ eine Gerade durch $\mathring{u}$ parallel zu $\vec{v}$. Von einem festen Punkt $\mathring{u}$ werden Vielfache von $\vec{v}$ als freie Vektoren abgetragen, so daß alle somit beschriebenen Punkte auf einer Geraden liegen. Umgekehrt, wird jeder Punkt der Geraden von $\mathring{u}$ durch ein $\lambda \in \mathbb{R}$ erreicht. Für $\lambda \in [0, .., 1] \subseteq \mathbb{R}$ wird die Strecke zwischen $\mathring{u}$ und $(\mathring{u} + \vec{v})$ charakterisiert. Die Form $\vec{u} = \mathring{u} + \lambda\vec{v}$ wird i.allg. als Parameterdarstellung einer Geraden (bzw. im weiteren auch einer Strecke) bezeichnet.

Satz 2.1 *Sei r eine Strecke des $\mathbb{R}^2$ mit der Parameterdarstellung $r = \mathring{v} + \lambda(\mathring{w} - \mathring{v})$ mit $0 \leq \lambda \leq 1$ und $\mathring{p}$ ein beliebiger durch ein festes λ definierter Punkt auf r. Eine zu r orthogonale Gerade s durch den Punkt $\mathring{p}$ wird durch die Parameterdarstellung $s = \mathring{p} + \mu(-x_2, x_1)$ mit $\mu \in \mathbb{R}^2$ beschrieben, wenn $(-x_2, x_1)$ durch $(x_1, x_2) := (\mathring{w} - \mathring{v})$ bestimmt ist.*

Beweis: Anhang C.1.

Hilfssatz 2.1 *Seien g und s Geraden des $\mathbb{R}^2$ mit den Parameterdarstellungen $g = \mathring{u} + \lambda(\mathring{v} - \mathring{u})$ und $s = \mathring{w} + \mu(\mathring{x} - \mathring{w})$ mit $\lambda, \mu \in \mathbb{R}$. Die beiden Geraden enthalten einen gemeinsamen Punkt $\mathring{s}$, gdw. $(\mathring{v} - \mathring{u})$ und $(\mathring{x} - \mathring{w})$ linear unabhängig sind.*

[4]Mit den hier verwendeten Sätzen werden zum Teil sehr spezielle Aussagen getroffen, so daß sie in der gängigen Literatur (etwa zur grafischen Datenverarbeitung) nicht zu finden sein werden.

Beweis: Anhang C.1.

Bemerkung 2.1 *Zwei Vektoren $\vec{a}, \vec{b} \in \mathbb{R}^2$ sind linear abhängig, genau dann wenn $a_2 b_1 = b_2 a_1$ gilt.*

Satz 2.2 *Seien g und s die in Hilfssatz 2.1 beschriebenen Geraden. Wenn g und s einen gemeinsamen Punkt $\mathring{s}$ enthalten, dann läßt sich $\mathring{s}$ durch Einsetzen der Lösung des linearen Gleichungssystems*

$$\begin{aligned} u_1 + \lambda(v_1 - u_1) &= w_1 + \mu(x_1 - w_1) \\ u_2 + \lambda(v_2 - u_2) &= w_2 + \mu(x_2 - w_2) \end{aligned}$$

für λ in g oder für μ in s berechnen.

Beweis: Anhang C.1.

Im folgenden wird die Lösung des linearen Gleichungssystems aus Satz 2.2 berechnet. Sei für eine einfachere Schreibweise $\vec{\alpha} := (\mathring{v} - \mathring{u})$ und $\vec{\beta} := (\mathring{x} - \mathring{w})$. Dann gilt,

$$\begin{aligned} u_1 + \alpha_1\lambda &= w_1 + \beta_1\mu \\ u_2 + \alpha_2\lambda &= w_2 + \beta_2\mu. \end{aligned} \tag{2.1}$$

Mit $a := (u_1 - w_1)$ und $b := (u_2 - w_2)$ ist (3.2) eine äquivalente Umformung von (3.1).

$$\begin{aligned} a + \alpha_1\lambda &= \beta_1\mu \\ b + \alpha_2\lambda &= \beta_2\mu \end{aligned} \tag{2.2}$$

$$\begin{aligned} \alpha_1\lambda - \beta_1\mu &= -a \\ \alpha_2\lambda - \beta_2\mu &= -b \end{aligned} \tag{2.3}$$

Durch die Multiplikation der ersten Gleichung aus (3.3) mit β_2 und der zweiten mit $(-\beta_1)$, ergibt sich das äquivalente Gleichungssystem (3.4).

$$\begin{aligned} \alpha_1\beta_2\lambda - \beta_1\beta_2\mu &= -a\beta_2 \\ -\alpha_2\beta_1\lambda + \beta_1\beta_2\mu &= b\beta_1 \end{aligned} \tag{2.4}$$

Die Addition der ersten Gleichung zur zweiten in (3.4), und eine sich anschließende Äquivalenzumformung der neuen zweiten Gleichung, führt zur Stufenform (3.5).

$$\begin{aligned} \alpha_1\beta_2\lambda - \beta_1\beta_2\mu &= -a\beta_2 \\ (\alpha_1\beta_2 - \alpha_2\beta_1)\lambda &= b\beta_1 - a\beta_2 \end{aligned} \tag{2.5}$$

Damit ergeben sich für λ und $\mathring{s}$ die Beziehungen

$$\lambda = \frac{b\beta_1 - a\beta_2}{\alpha_1\beta_2 - \alpha_2\beta_1} \quad und \quad \mathring{s} = \mathring{u} + \frac{b\beta_1 - a\beta_2}{\alpha_1\beta_2 - \alpha_2\beta_1}(\mathring{v} - \mathring{u}).$$

Bemerkung 2.2 *Wenn g eine Strecke ist (also $0 \leq \lambda \leq 1$), dann ist $\overset{\circ}{s}$ nur dann gemeinsamer Punkt von g und s, wenn $0 \leq \frac{b\beta_1 - a\beta_2}{\alpha_1\beta_2 - \alpha_2\beta_1} \leq 1$ gilt.*

Hilfssatz 2.2 *Seien g und s sich schneidende Geraden des $\mathbb{R}^2$ mit den Parameterdarstellungen $g = \overset{\circ}{u} + \lambda(\overset{\circ}{v} - \overset{\circ}{u})$ und $s = \overset{\circ}{w} + \mu(\overset{\circ}{x} - \overset{\circ}{w})$ mit $\lambda, \mu \in \mathbb{R}$ und sei $\overset{\circ}{s} \in \mathbb{R}$ der Schnittpunkt. Dann gilt die Gleichung:*

$$\overset{\circ}{s} = \overset{\circ}{u} + \frac{(u_2 - w_2)(x_1 - w_1) - (u_1 - w_1)(x_2 - w_2)}{(v_1 - u_1)(x_2 - w_2) - (v_2 - u_2)(x_1 - w_1)}(\overset{\circ}{v} - \overset{\circ}{u}).$$

Beweis Der Beweis ist durch Satz 2.2 und die sich anschließende Lösung des linearen Gleichungssystems geführt. □

Satz 2.3 *Sei r eine Gerade des $\mathbb{R}^2$ mit der Parameterdarstellung $r = \overset{\circ}{u} + \lambda(\overset{\circ}{v} - \overset{\circ}{u})$ mit $\lambda \in \mathbb{R}$ und $\overset{\circ}{p}$ ein Punkt auf r. Die Koordinaten der Punkte $\overset{\circ}{q}$ und $\overset{\circ}{q'}$ auf r mit Abstand $l \in \mathbb{R}_+$ von $\overset{\circ}{p}$ entsprechen den Lösungen des Gleichungssystems*

$$\begin{aligned} (x - p_1)^2 + (y - p_2)^2 - l^2 &= 0 \\ (x - u_1)(v_2 - u_2) - (y - u_2)(v_1 - u_1) &= 0. \end{aligned}$$

Beweis: Anhang C.1.

Bemerkung 2.3 *Eine einfachere Lösung der in Satz 2.3 beschriebenen Problemstellung liefert im folgenden Satz 2.5.*

Definition 2.5 (Blickrichtung) *Die Blickrichtung einer in einer Parameterdarstellung gegebenen Geraden (Strecke) sei durch die Orientierung des Richtungsvektors der Parameterdarstellung festgelegt.*

Der folgende Satz 2.4 entspricht Satz 14.1 in [Pavlidis, 1982] mit dem Unterschied, daß er dort für homogene Koordinaten notiert ist.

Satz 2.4 *Sei r eine Gerade des $\mathbb{R}^2$ mit der Parameterdarstellung $r = \overset{\circ}{u} + \lambda(\overset{\circ}{v} - \overset{\circ}{u})$ mit $\lambda \in \mathbb{R}$ und $\overset{\circ}{q}$ ein beliebiger Punkt, der nicht auf r liegt. Der Punkt $\overset{\circ}{q}$ liegt in Blickrichtung rechts von r, genau dann wenn*

$$q_1(u_2 - v_2) + q_2(v_1 - u_1) + (u_1v_2 - u_2v_1) < 0$$

gilt. Ansonsten liegt $\overset{\circ}{q}$ links von r.

Beweis Die Gültigkeit des Satzes ergibt sich aus der Gültigkeit von Satz 14.1 in [Pavlidis, 1982]. □

Satz 2.5 *Sei r eine Gerade des $\mathbb{R}^2$ mit der Parameterdarstellung $r = \mathring{u} + \lambda(\mathring{v} - \mathring{u})$ mit $\lambda \in \mathbb{R}$ und $\mathring{p}$ ein Punkt auf r. Der Punkt $\mathring{q}$, der in Blickrichtung mit Abstand $a \in \mathbb{R}_+$ von $\mathring{p}$ auf r liegt, ist durch*

$$\mathring{q} = \mathring{p} + \frac{a}{\|\mathring{v} - \mathring{u}\|} * (\mathring{v} - \mathring{u})$$

bestimmt.

Beweis: Anhang C.1.

Mit Hilfe der wenigen bisher gezeigten Aussagen ist es bereits möglich, ein minimales Robotersimulationssystem gemäß der eingangs dieses Abschnittes erörterten Motivation zu realisieren. Wie eine algorithmische Beschreibung eines solchen Systems aussehen könnte, zeigt Anhang C.2.

2.2 Wahrnehmungsmerkmale

Während einer Fahrt des Roboters nehmen seine Sensoren über die Zeit Meßwerte auf, die die durchfahrene Umgebung in einer der Sensorik eigenen Sichtweise charakterisieren. Um von den Sensormeßwerten zurück auf die Beschaffenheit der Umgebung schließen zu können, ist es notwendig, diese Sichtweise in einer geeigneten Form zu interpretieren. Eine Möglichkeit hierzu besteht darin, in den Meßwertfolgen der Sensoren typische Muster oder Merkmale einer bestimmten Umgebungskonstellation ausfindig zu machen, deren zeitgleiches oder auch zeitversetztes Auftreten während der Durchführung einer bestimmten Handlung einen Rückschluß auf die Umgebung zuläßt. Während in den Abschnitten 2.2.2 und 2.2.3 der Aufbau und die Repräsentation dieser Merkmale gemäß [Morik und Rieger, 1993] beschrieben wird, geht es im Abschnitt 2.2.1 um die Bildung von Sensorklassen, über die ein zeitgleiches oder zeitversetztes Auftreten von Meßwertmustern einzelner Sensoren ausgedrückt werden kann.

2.2.1 Bildung von Sensorklassen

Die im Abschnitt 2.1.1 beschriebene und in Abbildung 2.1 dargestellte Anordnung der 24 Ultraschallsensoren von PRIAMOS legt eine Einteilung der Sensoren in 8 grundsätzlich zu unterscheidende Sensorgruppen nahe. Jeweils eine Gruppe wird dabei durch die an den 4 Seiten des Roboters gelegenen Ultraschallsensoren gebildet, und jeweils eine weitere durch die an den 4 Ecken von PRIAMOS montierten Sensoren. Bei einer am Sensor S_{23} der Abbildung 2.1 beginnenden und links herum fortgeführten Numerierung der Sensorgruppen gehören demnach zur Gruppe G_1 die Sensoren S_{23}, S_0, S_1, zur Gruppe G_2 die Sensoren S_2, S_3, S_4 und entsprechend für die anderen Sensorgruppen: G_3: S_5, S_6, S_7; G_4: S_8, S_9, S_{10}; G_5: S_{11}, S_{12}, S_{13}; G_6: S_{14}, S_{15}, S_{16};

G_7: S_{17}, S_{18}, S_{19}; G_8: S_{20}, S_{21}, S_{22}. Bei einer Einschränkung der Vorwärtsbewegungsmöglichkeiten von PRIAMOS auf eine Bewegung in Richtung einer seiner 4 Seiten bei untersagter, gleichzeitiger Drehung, und der in Abschnitt 2.1.1 bereits erwähnten Annahme, daß *vorne* stets in Bewegungsrichtung liegt (auf beide Punkte geht Abschnitt 3.2 noch näher ein), sind diesen Sensorgruppen in Abhängigkeit von der momentanen Bewegungs- und Roboterorientierung (Definition 2.1) die Sensorklassen `vorne`, `rechts`, `links`, `hinten`, `vorne_links`, `vorne_rechts`, `hinten_links` und `hinten_rechts` zugeordnet. Die Bewegungsorientierung ist dabei definiert als Winkel zwischen der x-Achse des zugrundeliegenden Koordinatensystems und der zwei oder mehrere zeitlich aufeinanderfolgende Roboterpositionen verbindenden Geraden (siehe Definition 2.6).

Definition 2.6 (Bewegungsorientierung BAlpha) *Die der Bewegung des Roboters von einer Position A der Umgebung zum Zeitpunkt t_i der Roboterfahrt zu einer Position B der Simulationsumgebung zum Zeitpunkt t_j der Fahrt mit $A \neq B$ zugeordnete Bewegungsorientierung BAlpha ist definiert als Winkel zwischen der x-Achse des der Roboterfahrt (Simulation) zugrundeliegenden Koordinatensystems und der A und B verbindenden Geraden, falls alle in zeitlicher Abfolge der Fahrt zwischen A und B angefahrenen Positionen ebenfalls auf G liegen, und alle die gleiche und als einzige zugeordnete Roboterorientierung wie A zum Zeitpunkt t_i und B zum Zeitpunkt t_j besitzen.*

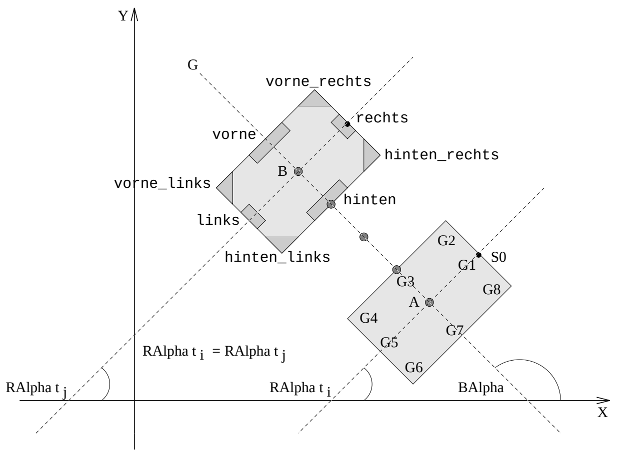


Abbildung 2.3: Bildung von Sensorklassen an einem Beispiel

Abbildung 2.3 verdeutlicht das bisher Erläuterte an einem Beispiel. Für die in diesem Beispiel vorgegebene Bewegungs- und Roboterorientierung wird der Sensorgruppe G_3 die Sensorklasse `vorne` und der Sensorgruppe G_7 die Sensorklasse `hinten` zugeordnet. Entsprechend gilt für die anderen Sensorgruppen: G_4: `links_vorne`, G_5: `links`, G_6: `links_hinten`, G_8: `rechts_hinten`, G1: `rechts` und G_2: `rechts_vorne`.

Die Bestimmung der einer Sensorklasse zugehörigen Sensorgruppe kann bei vorgegebener Bewegungs- und Roboterorientierung nach Algorithmus 2.1 erfolgen.

Eingabe: *RAlpha*, *BAlpha*, Sensorklasse *SK*
Ausgabe: Sensorgruppe *SG*

- Berechnet in Abhängigkeit von *RAlpha* und *BAlpha* die zu einer Sensorklasse gehörende Sensorgruppe.
- Die Sensorgruppen G_1, G_2, ..., G_8 seien wie im oben stehenden Text definiert.

```
  Diff := RAlpha - BAlpha;
  CASE Diff OF
     0          : CASE(*) SK OF
                    vorne: SG := G1; hinten: SG := G5;
                    rechts: SG := G7; links: SG := G3;
                    vorne_rechts: SG := G8; vorne_links: SG := G2;
                    links_hinten: SG := G4; rechts_hinten: SG := G6
                  END(*CASE*)
     (90 ∨ -270) : CASE SK OF
                    vorne: SG := G7; hinten: SG := G3;
                    rechts: SG := G5; links: SG := G1;
                    vorne_rechts: SG := G6; vorne_links: SG := G8;
                    links_hinten: SG := G2; rechts_hinten: SG := G4
                  END(*CASE*)
     (-90 ∨ 270) : CASE SK OF
                    vorne: SG := G3; hinten: SG := G7;
                    rechts: SG := G1; links: SG := G5;
                    vorne_rechts: SG := G2; vorne_links: SG := G4;
                    links_hinten: SG := G6; rechts_hinten: SG := G8
                  END(*CASE*)
     (180 ∨ -180) : CASE SK OF
                    vorne: SG := G5; hinten: SG := G1;
                    rechts: SG := G3; links: SG := G7;
                    vorne_rechts: SG := G4; vorne_links: SG := G6;
                    links_hinten: SG := G8; rechts_hinten: SG := G2
                  END(*CASE*)
  END(*CASE*)
END(*Algorithmus*)
```

Algorithmus 2.1: Bestimmung der Sensoren einer Sensorklasse

Durch eine kleine Abwandlung von Algorithmus 2.1 läßt sich umgekehrt aber ebenso einfach für einen gegebenen Sensor die Sensorklasse ermitteln, die der den Sensor enthaltenen Sensorgruppe in Abhängigkeit von *RAlpha* und *BAlpha* zugeordnet ist. Hierzu sind die inneren CASE-Anweisungen durch eine Ermittlung der Gruppenzugehörigkeit des gegebenen Sensors *S* zu ersetzen, über die dann die Auswahl der zugeordneten Sensorklasse *SK* erfolgt. So treten beispielsweise an Stelle des mit (*) gekennzeichneten CASE-Konstruktes die Anweisungen:

$S \in G_1 \Rightarrow SK :=$ `vorne`; $S \in G_2 \Rightarrow SK :=$ `vorne_links`;
$S \in G_3 \Rightarrow SK :=$ `links`; $S \in G_4 \Rightarrow SK :=$ `links_hinten`;
$S \in G_5 \Rightarrow SK :=$ `hinten`; $S \in G_6 \Rightarrow SK :=$ `links_rechts`;
$S \in G_7 \Rightarrow SK :=$ `rechts`; $S \in G_8 \Rightarrow SK :=$ `vorne_rechts`

Die den Sensorgruppen G_1, G_2, ...,G_8, in Abhängigkeit von der vom Roboter durchgeführten Bewegung zugeordneten Sensorklassen, geben Auskunft über die relative Lage eines möglicherweise anzupeilenden Objektes der Umgebung und abstrahieren dabei weitgehend von der tatsächlichen Lage der Sensoren am Roboter. Ihre Festlegung steht insofern im Einklang mit der Forderung, dem Roboter kein festes *vorne* (und damit *rechts*, *links*, *hinten*) zuzuordnen. In den beiden folgenden Abschnitten wird deutlich werden, daß nicht nur die in diesem Abschnitt zu Klassen zusammengefaßten Sensoren gemeinsame oder im zeitlichen Zusammenhang gemeinsam zu interpretierende Signalmuster beschreiben können, sondern daß es durchaus auch sinnvoll sein kann, aus den vorgegebenen 8 Sensorklassen weitere Oberklassen zu bilden. Beispiel solcher Klassen sind etwa die beiden Sensorklassen, die G_1 und G_5 bzw. G_3 und G_7 enthalten und je nach Bewegungs- und Roboterorientierung die Bezeichnungen `rechts_und_links` sowie `vorne_und_hinten` erhalten.

2.2.2 Repräsentation von Basis-Wahrnehmungsmerkmalen

Nachdem im Abschnitt 2.1 die durch den Roboter PRIAMOS zur Verfügung gestellten Sensordaten erörtert wurden, wird im vorliegenden Abschnitt erläutert, wie diese Daten zur Überführung in eine symbolische Repräsentation von Basis-Wahrnehmungsmerkmalen interpretiert werden können [Morik und Rieger, 1993]. Basis-Wahrnehmungsmerkmale bilden die Grundlage zur Beschreibung der im Abschnitt 2.2.3 vorgestellten Wahrnehmungsmerkmale nach [Morik und Rieger, 1993]. Diese wiederum dienen der Charakterisierung von Merkmalen von Objekten, von Objekten der Umgebung und letztlich der Umgebung selbst, durch die sich der Roboter während einer Fahrt bewegt. Die Konstruktion der Basis-Wahrnehmungsmerkmale ist daher von enormer Bedeutung für den Aufbau der im Kapitel 3 beschriebenen Begriffe, da sie ihre Bedeutungsverankerung in der wirklichen Welt verkörpern ([Wrobel, 1991], [Harnad, 1990]). [Wrobel, 1991] stellt heraus, daß die Konstruktion von begriffscharakterisierenden Merkmalen nicht losgelöst vom Aufbau und dem Einsatz der durch sie beschriebenen Begriffe betrachtet werden kann. Dies gilt natürlich erst recht, wenn die beschriebenen Begriffe zum maschinellen Lernen weiterer Begriffe (Kapitel 5) und zum Aufbau einer Begriffshierarchie (Kapitel 3) herangezogen werden. Erst durch eine Verzahnung des Vorganges der Konstruktion der Basismerkmale und dem Lernen abstrakterer Begriffe der vorgegebenen Begriffshierarchie ist eine Überprüfung der Tauglichkeit der Basismerkmale hinsichtlich ihrer Ausdrucksstärke im Hinblick auf die Lernbarkeit der Begriffe möglich.

Im Rahmen des Projektes BLearn II ist dies dadurch gewährleistet, daß im Falle nicht ausreichender Lernbarkeit der Begriffe bei Verwendung einer bestimmten Menge von Basis-Wahrnehmungsmerkmalen ein anderer Satz von Basis-Wahrnehmungsmerkmalen ermittelt wird. Hierzu arbeitet der in [Klingspor *et al.*, 1995] vorgestellte Algorithmus zur Konstruktion von Basis-Wahrnehmungsmerkmalen in Abhängigkeit von einer Menge verschiedener Parameter. Über eine Bewertungsfunktion zur Überprüfung der Qualität der gelernten Begriffe wird dann u.U. ein verwendeter Parametersatz modifiziert und der Lernprozeß wiederholt. Dabei ist es das Ziel, die Suche nach einem geeigneten Parametersatz über die Formulierung einer zu lösenden Lernaufgabe zu realisieren. Im Rahmen der vorliegenden Arbeit kann dieser Gedankengang jedoch nicht weitergehend vertieft werden. Für eine Darstellung des zugehörigen Berechnungsalgorithmus sei auf [Klingspor *et al.*, 1995] verwiesen. Eine Ausarbeitung der Thematik bietet [Wessel, 1995]. Die im folgenden angestellten Betrachtungen basieren im wesentlichen auf der Arbeit von [Morik und Rieger, 1993]. Eine umfassende und weitergehende Darstellung findet sich darüberhinaus in [Klingspor *et al.*, 1995].

Grundlage für die Bildung der Basis-Wahrnehmungsmerkmale sind die im Abschnitt 2.1.2 beschriebenen m/13-Fakten (Definition 2.3) zur Beschreibung der Sensordaten von PRIAMOS. Repräsentiert werden sie nach [Klingspor *et al.*, 1995] durch logische Fakten der Form:

$$\texttt{BW\_Merkmal}(\mathit{Trc},\ \mathit{SAlpha},\ S,\ T_1,\ T_2,\ \mathit{Grad}).$$

Hierbei steht die Prädikatsvariable `BW_Merkmal` für mögliche Prädikatsnamen verschiedener Basis-Wahrnehmungsmerkmale (s.w.u), während *Trc*, *SAlpha*, S sowie T_1 und T_2 die schon aus der Definition 2.3 bekannten Bezeichner sind. Im einzelnen wird ihnen die Bedeutung zugewiesen, daß die durch T_1 und T_2 bezeichneten Zeitpunkte, der durch *Trc* identifizierten Roboterfahrt, ein Intervall diskreter Zeitpunkte der Fahrt beschreiben[5], für welches gilt, daß der mit S bezeichnete Sensor des Roboters zu jedem im Intervall liegenden Zeitpunkt die Sensororientierung *SAlpha* (Definition 2.2) besitzt. Mit *Grad* wird der dem durch T_1 und T_2 beschriebenen Zeitintervall zugeordnete Gradient bezeichnet. In einem einfachen Fall entspricht dieser Wert dem Gradienten von Sensor S zwischen T_1 und T_2 (Definition 2.7). Seine genaue Festlegung ist aber vom verwendeten Parametersatz des in [Klingspor *et al.*, 1995] beschriebenen Algorithmus zur Berechnung der Basis-Wahrnehmungsmerkmale abhängig. Eine alternative Festlegung des Wertes ist beispielsweise der Mittelwert aller Gradienten von S zwischen unmittelbar aufeinanderfolgenden Zeitpunkten des vorgegebenen Intervalls.

Definition 2.7 (Gradient eines Sensors) *Sei mit S ein Sensor des Roboters bezeichnet, und seien weiterhin T_1 und T_2 zwei diskrete Zeitpunkte einer Roboterfahrt mit $T_1 < T_2$. Mit D sei ferner die vom Roboter zwischen T_1 und T_2 zurückgelegte Distanz bezeichnet. Wenn Sensor S zum Zeitpunkt T_1 den Meßwert $Dist_1$ und zum Zeitpunkt T_2 den Meßwert $Dist_2$ ermittelt, so ist der Gradient Grad von S zwischen T_1 und T_2 durch:*

[5] Es gelte stets $T_1 < T_2$.

$$Grad := \frac{Dist_2 - Dist_1}{D}$$

definiert, falls die Sensororientierung SAlpha von S zu jedem Zeitpunkt der Fahrt zwischen T_1 und T_2 unverändert und $D \neq 0$ ist.

Anhand des einem Basis-Wahrnehmungsmerkmal zugeordneten Gradienten lassen sich Aussagen über die Charakteristik des zeitlichen Verlaufes der Sensormeßwerte des zugehörigen Sensors ableiten. Diese Charakteristik wiederum spiegelt sich im Prädikatsnamen des zugehörigen Basis-Wahrnehmungsmerkmals wider. So sind beispielsweise `aufsteigend`, `absteigend`, `konstant`, `ansteigender_Sprung`, `absteigender_Sprung`, `keine_Messung`, `keine_Bewegung`, `direkt_daraufzu` und `direkt_davon_weg` mögliche Werte für `BW_Merkmal`. Die Bedeutung einiger dieser Bezeichnungen wird im folgenden in dem in Abbildung 2.4 dargestellten Beispiel aus [Klingspor und Morik, 1995] erläutert. Die Abbildung zeigt die über einen bestimmten Zeitraum aufgenommenen Meßwerte von Sensor S_1.

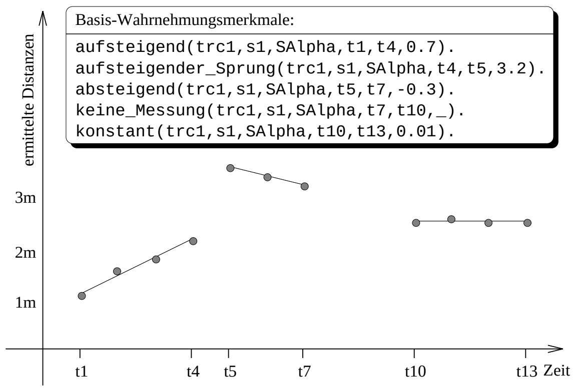


Abbildung 2.4: Sensormessungen und Basis-Wahrnehmungsmerkmale

Nach Abbildung 2.4 lassen sich fünf unterschiedliche Phasen der im zeitlichen Verlauf dargestellten Sensormessungen unterscheiden, die wiederum durch fünf verschiedene Basis-Wahrnehmungsmerkmale repräsentiert werden können. In der zwischen t_1 und t_4 liegenden ersten Phase können die Meßwerte durch eine linear ansteigende Funktion approximiert werden. Die Gradienten von Sensor S_1 zwischen jeweils zwei aufeinanderfolgenden Zeitpunkten in dem durch T_1 und T_4 beschriebenen Intervall liegen im Mittel bei 0.7.[6] Repräsentiert wird dieser Sachverhalt durch das Basis-Wahrnehmungsmerkmal `aufsteigend(trc1,s1,SAlpha,t1,t4,0.7)`. Der zwischen

[6]Um die im folgenden angegebenen Werte der Gradienten nachrechnen zu können, ist die Angabe der vom Roboter zwischen den einzelnen Zeitpunkten zurückgelegten Distanzen erforderlich. Da dies jedoch nur unwesentlich zum Verständnis des Beispiels beiträgt, wird darauf ebenso wie auf die Angabe der Sensororientierungen zur Vereinfachung verzichtet.

Zeitpunkt t_4 und Zeitpunkt t_5 liegende Gradient ist größer als 1. Da die Differenzen zwischen zwei Meßwerten für den Fall, daß dieselbe Kante angepeilt wurde, nicht größer sein kann, als die durch den Roboter im zugehörigen Zeitraum zurückgelegte Strecke, wird diese Situation als `aufsteigender_Sprung` bezeichnet [Klingspor und Morik, 1995]. Bei dieser Überlegung wird natürlich vorausgesetzt, daß sich alle Objekte im Raum bewegungslos an immer derselben Position befinden. Der mittlere Gradient zwischen t_5 und t_7 liegt bei -0.3. Die im zugehörigen Intervall liegenden Sensormeßwerte lassen sich bei negativem Gradienten durch eine linear absteigende Funktion approximieren, so daß dieser Sachverhalt durch das Basis-Wahrnehmungsmerkmal `absteigend` repräsentiert wird. Zwischen Zeitpunkt t_7 und t_{10} konnten keine Meßwerte durch Sensor S_1 ermittelt werden. Das zugehörige Zeitintervall wird daher durch das Basis-Wahrnehmungsmerkmal `keine_Messung` beschrieben. Die Gradienten zwischen aufeinanderfolgenden Zeitpunkten im durch t_{10} und t_{13} eingeschlossenem Zeitintervall liegen alle nahe beim Wert 0. Das zugehörige Basis-Wahrnehmungsmerkmal wird entsprechend als `konstant` bezeichnet. Die Wahrnehmungsmerkmale `direkt_daraufzu` und `direkt_davon_weg` können als Spezialfälle von `absteigend` und `aufsteigend` betrachtet werden, bei denen der zugeordnete Gradient nahe beim Wert 1 bzw. nahe beim Wert 0 liegt. Aus Definition 2.7 folgt für diese Fälle unmittelbar, daß sich der Roboter im zugehörigen Zeitabschnitt der Fahrt mit einer Bewegungsorientierung (Definition 2.6) auf ein Objekt zubewegt bzw. von einem Objekt wegbewegt, die gleich der Sensororientierung (Definition 2.2) von Sensor S_1 ist.

Abschließend bleibt festzuhalten, daß mit Hilfe von Basis-Wahrnehmungsmerkmalen die Sensormessungen einzelner Sensoren in ihrem zeitlichen Verlauf durch lineare Funktionen approximiert werden können. Hierzu ist eine Aufteilung des Zeitraumes einer Roboterfahrt in disjunkte Zeitabschnitte erforderlich, in denen eine solche Approximation erfolgen kann. Der in [Klingspor *et al.*, 1995] beschriebene inkrementelle Algorithmus zur Berechnung der Basis-Wahrnehmungsmerkmale muß demnach Mittel bereithalten, die

- die Anfangs- und Endzeitpunkte geeigneter Zeitabschnitte festzulegen vermögen, und

trotz einer möglicherweise vorhandenen „Verrauschtheit“ der Sensormessungen in der Lage sind

- zur Approximation geeignete Funktionen auszuwählen.

Die Grundlage hierzu bilden die Gradienten der Sensoren und eine Menge von Parametern [Klingspor *et al.*, 1995], über deren Festlegung der Approximationsprozeß gesteuert wird.

2.2.3 Repräsentation abstrakter Wahrnehmungsmerkmale

Im vorangegangenen Abschnitt 2.2.1 wurde einerseits erörtert, wie aus den Sensormessungen einzelner Sensoren Signalmuster erzeugt werden können, die den zeitlichen Verlauf der Meßwerte in disjunkten Zeitabschnitten einer Roboterfahrt charakterisieren, und andererseits, wie diese Signalmuster in Form von Basis-Wahrnehmungsmerkmalen repräsentiert werden können. Einzelne Basis-Wahrnehmungsmerkmale eröffnen jedoch für sich alleine betrachtet zunächst noch nicht die Möglichkeit, weitergehende Aussagen über die Umgebung zu machen. Hierzu zählen beispielsweise Aussagen über Objekte, an denen sich der Roboter während der Aufnahme der Sensormessungen vorbeibewegt, vor denen er stehenbleibt, oder sich dreht. Erst die Betrachtung des Auftretens mehrerer Basis-Wahrnehmungsmerkmale im zeitlichen Zusammenhang kann hierüber einen Aufschluß bringen, wenn dabei zudem die Roboterbewegung relativ zu den angepeilten Objekten Berücksichtigung findet. Im vorliegenden Abschnitt wird skizziert, wie mit Hilfe der Basis-Wahrnehmungsmerkmale über mehrere Abstraktionsstufen hinweg Umgebungsmerkmale charakterisiert werden können, die letztlich der Beschreibung handlungsorientierter Wahrnehmungsmerkmale dienen.

Die im folgenden angestellten Betrachtungen beschreiben die Ergebnisse von [Morik und Rieger, 1993]. Es kann nicht das Ziel dieses Abschnittes sein, diese Ausführungen erschöpfend wiederzugeben und zu diskutieren, sondern vielmehr einen Eindruck über den Aufbau und die Repräsentation handlungsorientierter Wahrnehmungsmerkmale zu vermitteln, der zum besseren Verständnis der im Abschnitt 3.3 beschriebenen wahrnehmungsintegrierenden Handlungsmerkmale beiträgt. Handlungsorientierte Wahrnehmungsmerkmale sind ein integraler Bestandteil der Beschreibung wahrnehmungsintegrierender Handlungsmerkmale. Dabei erfolgt eine Unterscheidung in zeitpunkt- und zeitintervallabhängige Wahrnehmungsmerkmale. Während [Morik und Rieger, 1993] den Aufbau und die Repräsentation zeitintervallabhängiger Wahrnehmungsmerkmale erörtern, ist die Entwicklung zeitpunktabhängiger Wahrnehmungsmerkmale z.Z. noch nicht abgeschlossen, so daß ihr Aufbau an dieser Stelle nicht näher betrachtet werden kann. Für eine umfassende Behandlung der in diesem Abschnitt dargestellten Thematik sei ansonsten auf [Morik und Rieger, 1993] und [Klingspor *et al.*, 1995] verwiesen.

In einem ersten Abstraktionsschritt werden die Basis-Wahrnehmungsmerkmale zur Bildung von Sensor-Merkmalen herangezogen. Sensor-Merkmale charakterisieren Kantenkonstellationen von gleichen oder verschiedenen Objekten in Abhängigkeit von der Bewegungsorientierung (Definition 2.6), mit der sich der Roboter relativ zur Raumorientierung der angepeilten Kanten an ihnen vorbeibewegt. Zur Illustration dieses Sachverhaltes diene Beispiel 2.1, für welches noch einmal auf Abbildung 2.2 des Abschnittes 2.1.2 zurückgegriffen wird.

Beispiel 2.1 *Unter der Annahme, daß sich der Roboter mit einer Bewegungsorientierung BAlpha = 90° und einer Roboterorientierung (Definition 2.1) RAlpha = 0° entlang der Kante* E_2 *des Objektes* O_2 *in Richtung auf den oberen Bildrand zubewegt, könnte der Sensor* S_{12} *der Sensorgruppe* G_5 *die durch* E_2 *und die Kante* E_{10} *des Objektes* O_1 *beschriebene Kantenkon-*

stellation durch seine Sensormessungen erfassen. Die Sensormessungen von Sensor S_{12} wiederum sind in ihrem zeitlichen Verlauf durch die im vorausgegangenen Abschnitt erläuterten Basis-Wahrnehmungsmerkmale annäherungsweise in abstrakter Form beschrieben. Für den betrachteten Zeitraum ist hier die Folge `konstant`, `absteigender_Sprung`, `konstant` *zu erwarten. Diese drei in zeitlicher Folge nacheinander auftretenden Basis-Wahrnehmungsmerkmale von Sensor S_{12} charakterisieren demnach die Kantenkonstellation E_2/E_{10}, für den Fall, daß sich der Roboter parallel an ihr vorbeibewegt.*

Für eine Bewegung diagonal zu den betrachteten Kanten würde dieselbe Kantenkonstellation natürlich durch eine andere Folge von Basis-Wahrnehmungsmerkmalen beschrieben werden. Eine schematische Darstellung möglicher Kantenkonstellationen, die ihnen zugeordneten Sensor-Merkmale, sowie die relativ zu ihnen verlaufenden Roboterbewegungsrichtungen zeigt Abbildung 2.5.

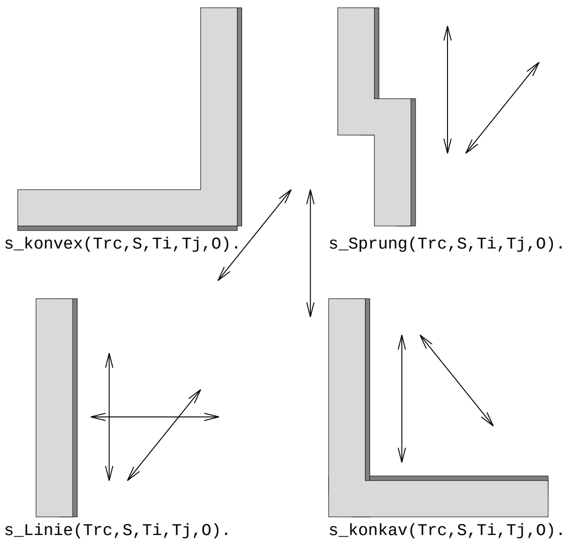


Abbildung 2.5: Sensor-Merkmale und zugehörige Kantenkonstellationen

Nach Beispiel 2.1 sollte klargeworden sein, daß Sensor-Merkmale durch Folgen von Basis-Wahrnehmungsmerkmalen repräsentiert werden können. Als Repräsentationsformalismus verwenden [Morik und Rieger, 1993] Hornklauseln, d.h. eine eingeschränkte Form der Prädikatenlogik erster Stufe. Eine Hornklausel (im weiteren wird einfacher nur von Regeln gesprochen) zur Beschreibung des im Beispiel 2.1 skizzierten Sensor-Merkmals ist bespielsweise[7]:

[7] Gemäß der in PROLOG üblichen Schreibkonvention, werden im folgenden Konstanten mit kleinem und Variablen mit großem Anfangsbuchstaben beginnend notiert.

`konstant`$(Trc, SAlpha, S, T_1, T_2, Grad_1)$ &
`aufsteigender_Sprung`$(Trc, SAlpha, S, T_2, T_3, Grad_2)$ &
`konstant`$(Trc, SAlpha, S, T_3, T_4, Grad_3)$
$\longrightarrow$ `s_Sprung`$(Trc, S, T_1, T_4, parallel)$.

Allgemein werden Sensor-Merkmale nach [Klingspor *et al.*, 1995] durch eine Gruppe von Prädikaten der Form:

`S_Merkmal`$(Trc,\ S,\ T_1,\ T_2,\ O)$.

dargestellt. Hierbei steht die Prädikatsvariable `S_Merkmal` für mögliche Prädikatsnamen verschiedener Sensor-Merkmale, also zum Beispiel `s_Sprung`, `s_konkav`, `s_konvex` und `s_Linie` aus Abbildung 2.5, während Trc, S, T_1 und T_2 die schon aus Definition 2.3 bekannten Bezeichner sind. Die Variable O steht für die in Frage kommenden Roboterbewegungsrichtungen relativ zu der durch `S_Merkmal` beschriebenen Kantenkonstellation. Mögliche Werte für O sind `parallel`, `diagonal`, `gerade_daraufzu` und `gerade_davon_weg`.

Aus den Sensor-Merkmalen einzelner Sensoren werden in einem weiteren Abstraktionschritt Sensorgruppen-Merkmale gebildet. Ebenso wie die Sensor-Merkmale charakterisieren auch sie bestimmte Kantenkonstellationen von Objekten der durchfahrenen Umgebung, mit dem Unterschied, daß sie dabei von den Bezeichnungen einzelner Sensoren abstrahieren, indem sie sich auf Sensorgruppen, respektive Sensorklassen, beziehen. Zur Erläuterung wieder auf Beispiel 2.1 zurückkommend, ist zu erwarten, daß über die Sensoren S_{11} und S_{13}, die wie Sensor S_{12} der Sensorgruppe G_5 angehören, beim Passieren der Kantenkonstellation E_2/E_{10} dieselbe Sequenz von Basis-Wahrnehmungsmerkmalen und damit dasselbe Sensor-Merkmal ermittelt werden kann, wie auch schon für Sensor S_{12}. Nach Algorithmus 2.1 wird der Sensorgruppe G_5 für die beschriebene Fahrt die Sensorklasse `links` zugeordnet. Die mit der Bildung von Sensorgruppen-Merkmalen verfolgte Idee ist, die drei über die Sensoren S_{11}, S_{12}, S_{13} ermittelten `s_Sprung`-Merkmale zu einem gemeinsamen Merkmal `sg_Sprung` zusammenzufassen, welches sich dann für den betrachteten Zeitraum, losgelöst von den konkreten Bezeichnungen der beteiligten Sensoren, lediglich auf die Sensorklasse `links` bezieht. Um dabei zusätzlich noch eine gewisse Robustheit gegenüber Meßfehlern einzelner Sensoren zu erzielen, wird die Bildung eines Sensorgruppen-Merkmals auch dann zugelassen, wenn nur zwei der zu erwarteten drei Sensor-Merkmale über eine Sensorgruppe ermittelt werden können. Eine das Sensorgruppen-Merkmal `sg_Sprung` beschreibende Regel ist demnach beispielsweise:

s_Sprung$(Trc,S_i,T_{i,1},T_{i,2},parallel)$ &
s_Sprung$(Trc,S_j,T_{j,1},T_{j,2},parallel)$ &
sKlasse$(Trc,S_i,T_{i,A},T_{i,E},links, WSeite)$ &
sKlasse$(Trc,S_j,T_{j,A},T_{j,E},links, WSeite)$ &
$(T_{j,1} - T_{i,1}) \leq 2$ &
$T_{i,A} \leq T_{i,1}$ & $T_{i,2} \leq T_{i,E}$ &
$T_{j,A} \leq T_{j,1}$ & $T_{j,2} \leq T_{j,E}$
$\longrightarrow$ sg_Sprung$(Trc,links,T_{i,1},T_{j,2}, WSeite,parallel)$.

Hierbei stellt das Prädikat sKlasse zusammen mit den aufgeführten Zeitpunktabhängigkeiten sicher, daß sowohl Sensor S_i als auch S_j während der Wahrnehmung des Merkmals s_Sprung zur Sensorklasse links gehören. Das Argument $WSeite$ gibt dabei zusätzlich an, ob der betrachtete Sensor an einer schmalen (schmale_Seite) oder an einer breiten Seite (breite_Seite) des Roboters montiert ist. Die Bedingung $(T_{j,1} - T_{i,1}) \leq 2$ an die in den s_Sprung-Prädikaten verwendeten Zeitpunkte beschreibt weiterhin, daß das ermittelte Sensor-Merkmal zeitlich etwas früher durch Sensor S_i beginnend wahrgenommen wird als durch Sensor S_j.

Allgemein werden Sensorgruppen-Merkmale nach [Klingspor *et al.*, 1995] durch eine Gruppe von Prädikaten der Form:

$$\texttt{SG\_Merkmal}(Trc,\ SKlasse,\ T_1,\ T_2,\ WSeite,\ O).$$

dargestellt. Hierbei steht die Prädikatsvariable SG_Merkmal für mögliche Prädikatsnamen verschiedener Sensorgruppen-Merkmale, also in Anlehnung an die Bezeichnungen der zugehörigen Sensor-Merkmale zum Beispiel sg_-Sprung, sg_konkav, sg_konvex und sg_Linie, während *Trc*, T_1 und T_2 die schon aus Definition 2.3 bekannten Bezeichner sind. Die Variable O steht wieder für die bei der Beschreibung der Sensor-Merkmale erörterte Roboterbewegungsrichtung relativ zu der durch das Sensorgruppen-Merkmal charakterisierten Kantenkonstellation. *SKlasse* hingegen gibt die nach Algorithmus 2.1 ermittelte Sensorklasse der dem Sensorgruppen-Merkmal zugrundeliegenden Sensorgruppe an oder bei einer entsprechenden Erweiterung von Algorithmus 2.1 auch Oberklassen wie beispielsweise rechts_und_links sowie vorne_und_hinten.

Mit den Sensor- und Sensorgruppen-Merkmalen werden Kantenkonstellationen von Objekten der durchfahrenen Umgebung anhand der Basis-Wahrnehmungsmerkmale charakterisiert. Um nun auch Aussagen über die Art der Objekte, an denen sich der Roboter während der Aufnahme der den Basis-Wahrnehmungsmerkmalen zugrundeliegenden Sensormessungen vorbeibewegt, gewinnen zu können, ist es erforderlich, das für bestimmte Objekte typische zeitgleiche und/oder zeitversetzte Auftreten von Sensorgruppen-Merkmalen zu ergründen. Läßt sich für ein bestimmtes Objekt in Abhängigkeit von der Art der Durchführung einer bestimmten Handlung (im einfachsten Fall eine Vorwärtsbewegung *parallel* zur angepeilten Kantenkonstellation) ein diskriminierendes Muster von Sensorgruppen-Wahrnehmungen festlegen, so werden diese auf einer höheren Abstraktionsstufe zu handlungsorientierten Wahrnehmungsmerkmalen zusammengefaßt. Handlungsorientierte Wahrnehmungsmerkmale beschreiben demnach komplexe Merkmale von Objekten unter Berücksichtigung der zu ihnen relativ verlaufenden Roboterbewegungsrichtung, die eine eindeutige Identifizierung der Objekte im Rahmen der Sensorik des Roboters zulassen.

Zur Veranschaulichung dieser Überlegungen an einem Beispiel diene die in Abbildung 2.6 schematisch dargestellte Roboterfahrt durch eine Türöffnung, wobei an dieser Stelle anzumerken ist, daß die im Abschnitt 2.1.1 beschriebene Sensorik des Roboters nicht ohne weiteres in der Lage ist, diese Fahrt anders wahrzunehmen, als eine Fahrt zwischen zwei Schränken, Regalen oder ähnlichen Objekten hindurch. Da dies jedoch die im vorliegenden Abschnitt erörterten Ideen im Grundsatz nicht in Frage stellt (die Sensorik könnte ja prinzipiell durch eine beliebig bessere (z.B. ein Laser-System) ausgetauscht werden), sei im weiteren davon abgesehen.

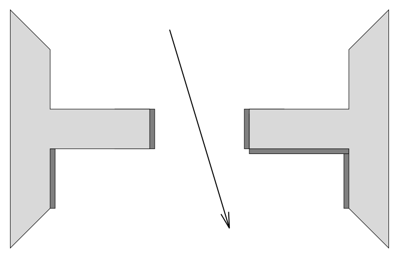


Abbildung 2.6: Roboterfahrt durch eine Tür

Bei der in Abbildung 2.6 skizzierten Fahrt werden die den Sensorklassen `links` und `rechts` zugeordneten Sensorgruppen unabhängig voneinander jeweils das Sensorgruppen-Merkmal `sg_Sprung` registrieren. Das von der Sensorklasse `rechts` wahrgenommene `sg_Sprung`-Merkmal wird dabei zeitlich etwas früher erkannt, als das von der Sensorklasse `links` wahrgenommene, da die Bewegungsrichtung des Roboters schräg zum Türrahmen verläuft. Werden die beiden Sensorgruppen-Merkmale unter Berücksichtigung ihres zeitlich versetzten Auftretens zu einer gemeinsamen Merkmalsbeschreibung zusammengezogen, so charakterisiert dieses zusammengesetzte, abstraktere Wahrnehmungsmerkmal die Tür aus Sicht der eingesetzten Sensorik für eine diagonal zum Türrahmen verlaufende Fahrt. Für diesen Fall erfolgt die Beschreibung des zusammengesetzten, abstrakteren Merkmals in Abhängigkeit von der durchgeführten Handlung. Merkmale dieser Art werden daher im weiteren auch als handlungsorientierte Wahrnehmungsmerkmale bezeichnet. Dem handlungsorientierten Wahrnehmungsmerkmal des erörterten Beispiels wird dabei der Name `durch_eine_Tür` zugewiesen. Eine mögliche Regel zu seiner Beschreibung ist beispielsweise:

`sg_Sprung`*(Trc,*`links`*,T_1,T_2, WSeite,diagonal)* &
`sg_Sprung`*(Trc,*`rechts`*,T_3,T_4, WSeite,diagonal)* &
$T_1 < T_3$ & $T_2 < T_4$ &
$T_A \leq T_1$ & $T_4 \leq T_E$
$\longrightarrow$ `durch_eine_Tür`*(Trc,T_A,T_E,*`rechts_und_links`*, WSeite, diagonal).*

Das handlungsorientierte Wahrnehmungsmerkmal `durch_eine_Tür` wird zu der eingangs des Abschnittes erwähnten Klasse der zeitintervallabhängigen Wahrnehmungsmerkmale gezählt, da sich die Wahrnehmung auf ein Zeitintervall bezieht, in dem der Roboter eine Bewegung durchführt. Im Gegensatz hierzu beziehen sich die z.Z. noch nicht entwickelten zeitpunktabhängigen Wahrnehmungsmerkmale auf Wahrnehmungen zu einem bestimmten Zeitpunkt. Da sie für den Aufbau der im Kapitel 3 beschriebenen operationalen Begriffe von ebenso großer Bedeutung wie die zeitintervallabhängigen, handlungsorientierten Wahrnehmungsmerkmale sind, werden sie gegen Ende des vorliegenden Abschnittes ebenfalls in ihrer allgemeinen Form beschrieben. Allgemein werden handlungsorientierte Wahrnehmungsmerkmale nach [Klingspor *et al.*, 1995] durch eine Gruppe von Prädikaten der Form:

`ZI_Wahrnehmung`(*Trc*, T_1, T_2, *WRichtung*, *WSeite*, *O*).

dargestellt. Hierbei steht die Prädikatsvariable `ZI_Wahrnehmung` für mögliche Prädikatsnamen zeitintervallabhängiger, handlungsorientierter Wahrnehmungsmerkmale. Neben dem vorgestellten Wahrnehmungsmerkmal `durch_eine_Tür` gehören hierzu zum Beispiel weiterhin die Prädikate `entlang_einer_Wand` und `entlang_einer_Tür`, die die für eine Fahrt vorbei an einer Wand bzw. einer Tür typischen Wahrnehmungen charakterisieren. Während die Variablen *Trc*, T_1, T_2 und *O* die schon weiter oben erläuterten Bezeichner sind, gibt *WRichtung* die Richtung an, aus der im durch T_1 und T_2 beschriebenen Zeitintervall etwas wahrgenommen wird. Diese Wahrnehmungsrichtung entspricht dabei der mit *SKlasse* bezeichneten Sensorklasse der Sensorgruppen-Merkmale. Die Sensorklassen werden demnach auf der Ebene handlungsorientierter Wahrnehmungsmerkmale als Richtungsangaben interpretiert, was erst durch ihre im Abschnitt 2.2.1 beschriebene Festlegung, in Abhängigkeit von der Bewegungs- und Roboterorientierung, ermöglicht wird, die unabhängig von einer tatsächlichen Lage der Signalmuster empfangenen Sensoren am Roboter erfolgt.

Abschließend sei noch die im weiteren Verlauf dieser Arbeit vorausgesetzte Form der zeitpunktabhängigen Wahrnehmungsmerkmale angegeben. Nach [Klingspor *et al.*, 1995] gehören sie zu einer Gruppe von Prädikaten der Form:

`ZP_Wahrnehmung`(*Trc*, *T*, *WRichtung*, *WSeite*).

Hierbei sind die Variablen *Trc*, *T*, *WRichtung* und *WSeite* die schon aus der Darstellung der zeitintervallabhängigen Wahrnehmungsmerkmale bekannten

Bezeichner. Die Prädikatsvariable `ZP_Wahrnehmung` steht dabei für mögliche Prädikatsnamen zeitpunktabhängiger Wahrnehmungsmerkmale. Hierzu gehören beispielsweise unter anderem `vor_einer_Tür`, `parallel_in_einer_Ecke` und `vor_einer_Wand`, die eine Erkennung der ihrem Namen zu entnehmenden Situationen zu einem bestimmten durch T bezeichneten Zeitpunkt ermöglichen.

2.3 Zusammenfassung

Im vorstehenden Kapitel wurden die Rahmenbedingungen vorgestellt, in die sich die weiteren Teile dieser Arbeit einfügen werden. Hierzu zählen einerseits die im Abschnitt 2.2 erörterten Ergebnisse von [Morik und Rieger, 1993] zur Repräsentation handlungsorientierter Wahrnehmungsmerkmale und andererseits die von der Universität Karlsruhe gelieferten Sensordaten von PRIAMOS [Dillmann *et al.*, 1993] bzw. die Daten eines PRIAMOS-Simulationssystems [Janakiraman und Wallner, 1992]. Auf beide Punkte wurde im Abschnitt 2.1 eingegangen. Neben einer einheitlichen, zusammenfassenden Darstellung dieser Arbeiten enthält das Kapitel aber auch eigene Arbeitsanteile. Hierzu gehört der im Abschnitt 2.2.1 vorgestellte Algorithmus zur Zuordnung von Sensorklassen zu Sensorgruppen, und die im Abschnitt 2.1.3 erörterten grundlegenden Betrachtungen zur Realisierung einer eigenen Robotersimulation, auf denen aufbauend im Anhang C.2 eine algorithmische Beschreibung eines einfachen Simulationssystems skizziert wird.

Kapitel 3

Aufbau einer Begriffshierarchie

Im vorangegangenen Kapitel werden vornehmlich die Rahmenbedingungen abgesteckt, in die die vorliegende Arbeit eingebettet ist. Hierzu zählen vor allem die im Abschnitt 2.2 vorgestellten Ergebnisse von [Morik und Rieger, 1993] zur Repräsentation handlungsorientierter Wahrnehmungsmerkmale, sowie die von der Universität Karlsruhe gelieferten Sensordaten von PRIAMOS [Dillmann *et al.*, 1993]. Im vorliegenden Kapitel wird unter Berücksichtigung dieser Arbeiten ein Vorschlag unterbreitet, wie die im Kapitel 1 formulierte Zielsetzung von BLearn II erreicht werden kann, wonach eine Verbindung zwischen Handlungen eines mobilen Roboters einerseits und seinen mit den Handlungen einhergehenden Wahrnehmungen andererseits hergestellt werden muß, um auf diese Weise zu einer Begriffsoperationalisierung zu gelangen. Im einzelnen besteht der Beitrag dieses Kapitels in der Festlegung von Basis-Handlungsmerkmalen, ihrer Berechnung aus den von PRIAMOS zur Verfügung gestellten Daten, der Entwicklung wahrnehmungsintegrierender Handlungsmerkmale, einer Beschreibung operationaler Begriffe mit ihrer Hilfe, sowie einer Diskussion von Anforderungen, die von einer Repräsentation operationaler Begriffe zu erfüllen sind.

Zunächst auf den letzten Punkt bezugnehmend wird im Abschnitt 3.1 insbesondere ein Modell zur Interpretation von Richtungsangaben erörtert, und ein Konzept zur Gestaltung der Bildung und des Einsatzes operationaler Begriffe vorgestellt. Im Abschnitt 3.2 wird auf diesen Überlegungen aufbauend eine Repräsentation von Basis-Handlungsmerkmalen vorgeschlagen, die sowohl eine inkrementelle Berechnung von Basis-Handlungsmerkmalen während einer kontrollierten Roboterfahrt in der sog. Lernphase des Roboters, als auch ihre Übersetzung in elementare Roboteroperationen erlaubt, so daß Basis-Handlungsmerkmale in der sog. Arbeitsphase des Roboters als an den Roboter abzusetzende Handlungsanweisungen verwendet werden können. Die Idee, eine Unterscheidung zwischen Lern- und Arbeitsphase des Roboters einzuführen, wird ebenfalls im Abschnitt 3.1 motiviert. Im Abschnitt 3.2 wird mit der dort vorgeschlagenen Repräsentation wahrnehmungsintegrierender Handlungsmerkmale gezeigt, daß sich Basis-Handlungsmerkmale mit den im Abschnitt 2.2 beschriebenen handlungsorientierten Wahrnehmungsmerkmalen auf dem Wege zu einer Begriffsoperationalisierung in einer sinnvollen Weise verbinden lassen. Auf welche Weise wahrnehmungsintegrierende Handlungsmerkmale dann zu operationalen Begriffen zusammengefügt werden können, wird im Abschnitt 3.4 erörtert. In diesem Abschnitt wird die vorgeschlagene Repräsentation operationaler Begriffe dargestellt. Das vorliegende Kapitel wird schließlich mit einer kurzen Zusammenfassung im Abschnitt 3.5 abgeschlossen.

3.1 Anforderungen an die Repräsentation

Eines der vordringlichsten Ziele innerhalb des Projektes BLearn II ist eine Verbindung zwischen Handlung einerseits und der mit ihr einhergehenden Wahrnehmung andererseits auf dem Wege zu einer Begriffsoperationalisierung zu schaffen. Eine interessante Diskussion der Bedeutung von Handlungen und Wahrnehmungen bei der Begriffsbildung findet sich in [Wrobel, 1991]. Für eine Übertragung dieses Leitgedankens auf den im Abschnitt 2.1.1 vorgestellten Roboter bedeutet dies nicht nur die Notwendigkeit der Zusammenführung von aus Sensordaten gewonnenen Wahrnehmungsmerkmalen und den während einer Wahrnehmung durchgeführten Handlungen des Roboters, sondern darüberhinaus die Gestaltung dieser Verbindung in einer Weise, die es gestattet, Handlungen in Abhängigkeit von Wahrnehmungen zu bestimmen, auszuführen und während ihrer Durchführung zu kontrollieren, also kurz eine Begriffsoperationalisierung zu erzielen. Die im Abschnitt 2.2 vorgestellten handlungsorientierten Wahrnehmungsmerkmale beschreiben zwar bereits die aus den Sensordaten des Roboters gewonnenen Wahrnehmungen auf allen zu ihrem Aufbau notwendigen Abstraktionsebenen auch in Abhängigkeit von der Durchführung bestimmter Handlungen, eine Rückführung von Wahrnehmungsmerkmalen auf die zu ihrer Wahrnehmung erforderlichen Handlungen und deren Ausführung ist jedoch mit ihrer Hilfe nicht möglich, da ihre Repräsentation die hierzu notwendigen Handlungsanweisungen nicht beinhaltet. Eine Operationalisierung im obigen Sinne liegt an dieser Stelle demnach noch nicht vor.

Mit jeder Handlungsausführung ist weiterhin ein Ziel verknüpft. Dieses Ziel kann auf der einen Seite die Handlungsausführung an sich beinhalten, während auf der anderen Seite die Handlungsausführung als Teil einer Kette aufeinanderfolgender Handlungen dazu dienen kann, dem eigentlichen Ziel ein Stück weit näherzukommen. Aus dem letzteren Fall ergibt sich die Forderung, eine Möglichkeit zur Handlungsplanung mit Hilfe der zu bildenden operationalen Begriffe bereitzustellen. Eine Handlungsplanung unter Zuhilfenahme operationaler Begriffe setzt dabei zweierlei voraus. Zum einen muß für jeden operationalen Begriff eindeutig sein, in welcher konkreten Situation er zur Erreichung welches Zieles einsetzbar ist, und zum anderen muß sein Einsatz zu einer Situation führen, von der aus die Anwendung weiterer operationaler Begriffe denkbar ist. Auf diese Weise soll es einem Planer zumindestens möglich sein, ausgehend von einem operationalen Begriff, dessen Ausführung dem zu erreichenden Ziel entspricht, rückwärts verkettend einer Folge operationaler Begriffe zu bestimmen, deren Ausführung aus der konkret vorhandenen Situation heraus zum Ziel führen kann. Die den einzelnen operationalen Begriffen zugeordneten Ziele, und damit letztlich die operationalen Begriffe selbst, sollen dabei so abstrakt gehalten sein, daß sie leicht von Menschen zu verstehen und vorzugeben sind und somit als Sprachelemente einer Kommandosprache höherer Ebene fungieren können.

Die Vorgabe operationaler Begriffe bezieht sich dabei einzig und allein auf ihren Einsatz als Kommandosprache, während der zu ihrer Bildung notwendige Prozeß, von menschlicher Vorgabe losgelöst, weitgehend automatisch über den Einsatz maschinellen Lernens erfolgen soll. Dies setzt die Wahl einer Repräsentation voraus, deren Ausdruckskraft einerseits genügt, die abstrakten Begriffe im Verlauf von Sensormessungen eindeutig widerzuspiegeln, und deren Aufbau andererseits von derart einfacher Struktur ist, daß die mit ihr verbundenen, zu stellenden Lernaufgaben durch maschinelle Lernverfahren handhabbar bleiben. Diese zugleich mit stärkste Anforderung an operationale Begriffe unterscheidet sie im Hinblick auf ihre Verwendung als Planungsoperatoren deutlich von den klassischen Ansätzen beim Planen, da die Begriffe durch die Art und Weise ihres Bildungsprozesses eine klare Zuweisung von Bedeutung in der Umwelt erfahren. Eine interessante Diskussion der Bedeutung der Symbolverankerung bei Begriffsbildungsprozessen liefert [Harnad, 1990].

Die vorangegangenen Überlegungen werden im folgenden zu einem Anforderungskatalog zusammengefaßt, der als Leitlinie insbesondere für die weiteren Abschnitte des vorliegenden Kapitels zur Beschreibung der Repräsentation operationaler Begriffe fungiert. Auf das Lernen operationaler Begriffe indessen wird im Kapitel 5 eingegangen, während Kapitel 6 ausblickend einige Aspekte des Einsatzes operationaler Begriffe als Planungsoperatoren beleuchtet.

1. Es sind Basis-Handlungsmerkmale festzulegen, die sich auf der einen Seite leicht in elementare Roboteroperationen übersetzen lassen und auf der anderen Seite inkrementell aus den vom Sensorkontroll- und Signalverarbeitungssystem während der Durchführung bestimmter Handlungen zur Verfügung gestellten Daten ermittelt werden können.

2. Die gewählten Basis-Handlungsmerkmale sind in einer Weise mit den im Abschnitt 2.2 beschriebenen handlungsorientierten Wahrnehmungsmerkmalen zu verbinden, die es ermöglicht, Handlungen des Roboters unter der Kontrolle von während der Handlungsausführung ermittelten Wahrnehmungsmerkmale durchzuführen.

Für ein besseres Verständnis der Notwendigkeit dieser beiden Forderungen ist es bereits an dieser Stelle angezeigt, eine Trennung zwischen Lern- und

Arbeitsphase des Roboters einzuführen. Während in der *Lernphase* dem Roboter bestimmte Handlungen in Abhängigkeit von bestimmten Wahrnehmungen vorgeführt werden (z.B. *joy stick controlled*) und aus dieser Vorführung dann operationale Begriffe im Rahmen der in diesem Kapitel vorgestellten Begriffshierarchie gelernt werden, erfolgt in der *Arbeitsphase* der Einsatz der gelernten Begriffe zur Erreichung vorgegebener Ziele. In der Lernphase müssen daher Basis-Handlungsmerkmale aus den vom Sensorkontroll- und Signalverarbeitungssystem zur Verfügung gestellten Daten zum Lernen der Begriffshierarchie ermittelt werden können. Dieselben Basis-Handlungsmerkmale dienen dann in der Arbeitsphase als an den Roboter abzusetzende Handlungsanweisungen, auf die in Abhängigkeit von aktuell registrierten Wahrnehmungen über die gelernte Begriffshierarchie geschlossen wird. Zur Ausführung der Handlungsanweisungen ist daher eine Übersetzung in robotereigene Elementaroperationen erforderlich, wobei die Kontrolle der Handlungsanweisungen — ganz allg., wann soll mit einer Handlung begonnen werden; was soll während ihrer Ausführung wahrgenommen werden; wann soll eine Handlung beendet sein? — wiederum über Wahrnehmungsmerkmale erfolgt.

3. Auf einer höheren Abstraktionsstufe der Verbindung von Handlungs- und Wahrnehmungsmerkmalen ist zu gewährleisten, daß abhängig von der Registrierung bestimmter Wahrnehmungen eine Verbindung zu den für diese Wahrnehmungen typischen Handlungen hergestellt werden kann.

4. Für jeden der zu bildenden operationalen Begriffe muß zum einen klar sein, in welcher Situation er eingesetzt werden kann und zum anderen, zu welcher konkreten Situation ein erfolgreicher Einsatz führt.

5. Im Sinne eines Einsatzes als Planungsoperatoren muß eine Möglichkeit vorhanden sein, operationale Begriffe über Situationen, die ihren Einsatz erlauben, und solche, zu der ihr erfolreicher Einsatz führt, zu verbinden, um auf diese Weise, ausgehend von einem Zielbegriff, rückwärts verkettend eine Folge operationaler Begriffe bestimmen zu können, deren Ausführung aus der konkret vorhandenen Situation heraus zum Ziel führen kann, wobei dem Ziel die aus der Anwendung des Zielbegriffes resultierende Situation entspricht.

6. Die durch Anwendung einzelner operationaler Begriffe erreichbaren Situationen sollen sich in der Namensgebung der Begriffe widerspiegeln und dabei von derart grundsätzlicher Natur sein, daß die operationalen Begriffe als Sprachelemente einer Kommandosprache höherer Ebene fungieren können.

7. Im Rahmen der vorgegebenen Repräsentation sollen operationale Begriffe durch maschinelle Verfahren gelernt werden können.

Die vorletzte Forderung schließt insbesondere mit ein, daß Richtungsangaben in einer für Menschen gewohnten Weise repräsentiert und auch interpretiert werden. Zur Veranschaulichung dieser Überlegung diene als Beispiel eine Situation, in der sich der Roboter in einem Raum befindet und den Auftrag erhält, sich aus dem Raum durch die Tür zu bewegen, um dann nach rechts dem Gang folgend vor der nächsten Tür stehen zu bleiben.

Um dem auftragerteilenden Menschen die Formulierung dieses Auftrages zu erleichtern (im Idealfall sollte die angegebene Formulierung ausreichend sein), muß die Richtungsangabe `rechts` in diesem Auftrag durch den Roboter losgelöst von seinem internen Bezugssystem für Richtungsangaben interpretiert werden können. Auf diese Weise wird es dem Roboter möglich, die Richtungsangabe `rechts` unabhängig von der Bewegungsrichtung (*vorwärts, seitwärts, rückwärts*) zu verstehen, die er gemäß seinem eigenen Bezugssystem beim Durchschreiten der Tür einschlägt. Der auftragerteilende Mensch muß sich in diesem Fall dann keine Gedanken mehr darüber machen, in welcher Richtung für den Roboter *vorne* liegt, und wie er den Auftrag formulieren muß, damit die von ihm gewünschte Bewegungsrichtung auch tatsächlich eingeschlagen wird (etwa: wenn du die Tür mit einer Bewegung in Richtung `vorne` durchschreitest, dann bewege dich anschließend nach `rechts`, und wenn Du die Tür mit einer Bewegung in Richtung `hinten` durchschreitest, dann bewege dich nach `links`, etc.).

Die sich aus diesen Überlegungen ergebene Idee ist, Richtungsangaben in Aufträgen stets *extrinsisch* zu interpretieren. Eine Unterscheidung zwischen *intrinsischen* und *extrinsischen* Richtungsangaben erfolgt in vielen Arbeiten zum räumlichen Schließen mit kognitionsorientierter Ausrichtung. Während intrinsische Richtungsangaben im Sinne einer „natürlichen“ Gerichtetheit verwendet werden (*„vorne ist immer da, wo die Nase ist“*), werden extrinsische Richtungsangaben unter Bezug auf die aktuelle Bewegungsrichtung interpretiert (*„vorne liegt immer in Bewegungsrichtung“*). [Habel *et al.*, 1993] bieten eine umfassende Einführung in diese Thematik, die aus Platzgründen hier nicht weitergehend vertieft werden soll. Zur Realisierung dieser Idee wird in der vorliegenden Arbeit zwischen absoluten (intrinsisch interpretierten) und relativen (extrinsisch interpretierten) Richtungsangaben unterschieden.

Während sich demnach die absoluten Richtungsangaben stets auf das robotereigene Bezugssystem beziehen, werden die relativen Richtungsangaben in Abhängigkeit von der aktuellen Bewegungsrichtung festgelegt, indem stets die aktuelle Bewegungsrichtung als in Richtung relativ `vorne` betrachtet wird. Bewegt sich der Roboter demnach auftragsgemäß durch die Tür, so ist dies in diesem Modell eine Bewegung in Richtung relativ `vorne`, ganz gleich mit welcher Seite des Roboters zuerst der Türrahmen passiert wird. Wird weiterhin die Richtungsangabe `rechts` des Auftrages als relative Richtungsangabe aufgefaßt, so entspricht sie der im obigen Auftrag gemeinten Vorstellung von `rechts`. Zur Ausführung des Auftrages bedarf es dann noch einer Umrechnungsmöglichkeit von relativen in absolute Richtungsangaben, bzw. auch in umgekehrter Weise für die Lernphase. Wie diese Umrechnung durchgeführt wird, und wie das vorgeschlagene Modell zur Richtungsangabe im einzelnen realisiert wird, behandelt Abschnitt 3.2.

3.2 Basis-Handlungsmerkmale

Im vergangenen Abschnitt wird unter Punkt 1) des von der Repräsentation operationaler Begriffe zu erfüllenden Anforderungskataloges herausgestellt, daß Basis-Handlungsmerkmale in der Lernphase aus den vom Roboter zur Verfügung gestellten Daten inkrementell ermittelt werden sollen und daß sie weiterhin in der Arbeitsphase als an den Roboter abzusetzende Handlungsanweisungen verwendbar sein müssen.

Vor dem Hintergrund des zweiten Gesichtspunktes wird im Abschnitt 3.2.1 erläutert werden, welche Basis-Handlungsmerkmale zum Aufbau der im vorliegenden Kapitel vorgestellten Begriffshierarchie herangezogen werden, und wie sie repräsentiert werden können. Auf den ersten Aspekt bezugnehmend wird dann im Abschnitt 3.2.2 dargestellt werden, wie eine inkrementelle Berechnung der vorgeschlagenen Basis-Handlungsmerkmale aus den vom Sensorkontroll- und Signalverarbeitungssystem des Roboters zur Verfügung gestellten Daten in der Lernphase erfolgen kann.

3.2.1 Repräsentation von Basis-Handlungsmerkmalen

Handlungsanweisungen an den Roboter werden sich zwangsläufig auf die Handlungsausführungsmöglichkeiten des verwendeten Robotertypes beschränken müssen. Nach den Ausführungen im Abschnitt 2.1.1 ist der im Projekt BLearn II eingesetzte Roboter PRIAMOS in der Lage, beliebige Bewegungen in longitudinaler und transversaler Richtung sowie Drehungen beliebiger Richtungen um sein Zentrum durchzuführen, wobei zudem Dreh- und Vorwärtsbewegungen simultan erfolgen können (PRIAMOS IST DEMNACH *holonom* IN DER EBENE). Eine derartige Vielfältigkeit in den Bewegungsmöglichkeiten ist sicherlich von den meisten mobilen Systemen nicht zu erwarten, was im Hinblick auf die Übertragbarkeit des verfolgten Ansatzes auf andere Systeme als Argument für eine Einschränkung der Bewegungsfreiheit von PRIAMOS zu bewerten ist. Als ein stärkeres Argument für eine Einschränkung der Bewegungsabläufe ist jedoch die von PRIAMOS verwendete Sensorik ins Feld zu führen. Bei voller Ausschöpfung der möglichen Freiheitsgrade bei der Durchführung von Handlungen ist insbesondere in Anbetracht der im Abschnitt 2.1.2 angedeuteten Problematik beim Einsatz von Ultraschallsensoren zu erwarten, daß die Erkennung von typischen auf bestimmte Umgebungskonstellationen rückführbaren Mustern in den Meßwertfolgen der Sensoren, nur mit einem unverhältnismäßig großem Aufwand zu realisieren ist, der die prinzipielle Praktikabilität des verfolgten Ansatzes in keinster Weise deutlicher hervorheben kann, als es derselbe Ansatz unter vereinfachten Bedingungen ohnehin schon vermag. Aus diesem Blickwinkel ist die bereits im Abschnitt 2.2.1 bei der Bildung von Sensorklassen eingeführte Einschränkung der Vorwärtsbewegungsmöglichkeiten von PRIAMOS auf eine Bewegung in Richtung auf einer seiner 4 Seiten bei untersagter, gleichzeitiger Drehung keineswegs als inhaltliche, sondern als rein technische Vereinfachung zu betrachten. Abbildung 3.1 zeigt schematisch die somit verbleibenden Handlungsausführungsmöglichkeiten.

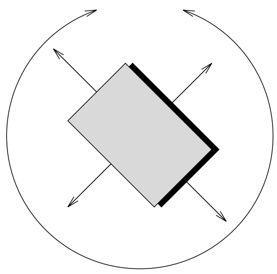


Abbildung 3.1: Erlaubte Bewegungsmöglichkeiten des Roboters

Bei Einschränkung der Bewegungsfreiheit von PRIAMOS in der beschriebenen Weise verbleiben als Handlungsanweisungen die Aufforderungen an den Roboter, sich in einer bestimmten Richtung vorwärtszubewegen, sich in einer bestimmten Richtung zu drehen, oder stehen zu bleiben. Zur Durchführung der Vorwärtsbewegung bedarf es einer Richtungsangabe im robotereigenen Bezugssystem. Hingegen benötigt das die Handlungsanweisung beschreibende Basis-Handlungsmerkmal nach den unter Punkt 6) ausgeführten Überlegungen des vorangegangenen Abschnittes zum Aufbau der Begriffshierarchie ebenso die Angabe der relativen Bewegungsrichtung. Die Unterscheidung zwischen Lern- und Arbeitsphase muß demnach in der Repräsentation des Vorwärtsbewegungen beschreibenden Basis-Handlungsmerkmals berücksichtigt werden. Erreicht wird dies, indem das entsprechende Basis-Handlungsmerkmal auf zwei Ebenen repräsentiert wird, die sich nur dadurch unterscheiden, daß in einem Fall die Richtungsangaben relativ und im anderen Fall absolut angegeben werden, d.h., für den ersten Fall extrinsisch und für den zweiten Fall intrinsisch interpretiert werden. Diese Unterscheidung ist für Drehbewegungen nicht erforderlich, da die Festlegung von Drehrichtungen unabhängig vom aktuellen `vorne` des Roboters erfolgen kann. Basis-Handlungsmerkmale werden nach diesen Betrachtungen durch Prädikate der Form:

`bleibe_stehen`(Trc, T_1, T_2),
`bewege_dich`$(Trc, T_1, T_2, GK, \mathit{RelRichtung})$,
`a_bewege_dich`$(Trc, T_1, T_2, GK, \mathit{AbsRichtung})$,
`drehe_dich`$(Trc, T_1, T_2, RGK, D\mathit{Richtung})$

dargestellt. Hierbei sind *Trc*, T_1 und T_2 die bereits bekannten Bezeichner zur Identifizierung der betrachteten Roboterfahrt sowie zur Kennzeichnung der Start- und Endzeitpunkte der ausgedrückten Handlungsanweisungen. *RelRichtung* bzw. *AbsRichtung* stehen für die relativen bzw. absoluten Bewegungsrichtungsangaben, d.h. für `links`, `rechts`, `vorne` sowie `hinten`. Zur Definition eines robotereigenen Bezugssystems sei die

Seite, an der Sensor S_0 montiert ist (siehe Abbildung 2.1), stets als absolut `vorne` liegend bezeichnet. Zur Unterscheidung der Werte von *RelRichtung* werden weiterhin die Werte für *AbsRichtung* mit dem Präfix 'a_' notiert, also z.B. `a_vorne`. Die Variablen GK und RGK stehen für Geschwindigkeitsklassen, die sich auf Bewegungs- bzw. Rotationsgeschwindigkeiten bei Vorwärtsbewegungen und Drehungen beziehen. Mögliche Werte für GK sind `langsame_Bewegung`, `mittlere_Bewegung` sowie `schnelle_Bewegung`, während entsprechende Werte für RGK mit `langsame_Drehung`, `mittlere_Drehung` und `schnelle_Drehung` bezeichnet werden. Die Festlegung dieser Werte erfolgt in der Lernphase über die Zuordnung:

$$\begin{array}{lll}
0 < v_B \leq 0.1 & \Rightarrow GK & := \texttt{langsame_Bewegung}, \\
0.1 < v_B \leq 0.5 & \Rightarrow GK & := \texttt{mittlere_Bewegung}, \\
0.5 < v_B \leq 1 & \Rightarrow GK & := \texttt{schnelle_Bewegung}, \\
0 < v_D \leq 2 & \Rightarrow RGK & := \texttt{langsame_Drehung}, \\
2 < v_D \leq 10 & \Rightarrow RGK & := \texttt{mittlere_Drehung}, \\
10 < v_D \leq 90 & \Rightarrow RGK & := \texttt{schnelle_Drehung},
\end{array}$$

wobei v_B die Bewegungsgeschwindigkeit in Meter/Sekunde und v_D die Rotationsgeschwindigkeit in Grad/Sekunde angibt. Die Berechnung von v_B und v_D ist einfach möglich, da die Anzahl der vom Sensorkontroll- und Signalverarbeitungssystem durchgeführten Sensormessungen pro Sekunde bekannt ist, und somit auf die zwischen Meßzeitpunkt T_1 und T_2 vergangene Zeit in Sekunden geschlossen werden kann. In der Arbeitsphase können die Geschwindigkeitsklassen im sichersten Fall durch die jeweils kleinste Geschwindigkeit interpretiert werden, die ihnen noch zugeordnet ist.

Im Abschnitt 3.1 wird bereits motiviert, warum eine Unterscheidung zwischen absoluten und relativen Richtungsangaben sinnvoll ist. Im folgenden wird das dort angesprochene Modell zur Richtungsangabe weiter ausgeführt. Die Festlegung der absoluten Richtungsangabe *AbsRichtung* in einem Basis-Handlungsmerkmal `a_bewege_dich/5` ist klar, da sie stets relativ zur Lage von Sensor S_0 erfolgt. Die Festlegung der im Abschnitt 2.2.1 beschriebenen Sensorklassen und damit auch die Festlegung der Richtungsangaben der im Abschnitt 2.2.3 vorgestellten handlungsorientierten Wahrnehmungsmerkmale erfolgt hingegen gemäß der Vorstellung, daß die aktuelle Bewegungsrichtung als in Richtung relativ `vorne` betrachtet wird. Hierbei ist es allerdings noch nicht notwendig gewesen, in den Roboterfahrten mehr als eine Bewegung in einer Richtung zu betrachten. Für Roboterfahrten, in denen Vorwärtsbewegungen und auch Rotationen gleichermaßen vorhanden sind, reicht diese intuitiv einfache Vorstellung alleine nicht mehr aus, da

- noch festzulegen ist, was nach Durchführung einer Rotation oder längerem Stehen auf einer Stelle als relativ `vorne` zu betrachten ist, und
- diese Festlegung zur Erfüllung von Punkt 2) im Abschnitt 3.1 — nach der die Basis-Handlungsmerkmale in einer Weise mit den hand-

lungsorientierten Wahrnehmungsmerkmalen zu verbinden sind, die es ermöglicht, Handlungsanweisungen in der Arbeitsphase unter der Kontrolle von Wahrnehmungsmerkmalen durchzuführen — im Einklang mit den für die handlungsorientierten Wahrnehmungsmerkmale festgelegten Wahrnehmungsrichtungsangaben stehen muß, sowie weiterhin

- sichergestellt sein muß, daß die in den zu lernenden operationalen Begriffen verwendeten Richtungsangaben in der Arbeitsphase auf absolute Richtungsangaben in den Basis-Handlungsmerkmalen zurückgeführt werden können.

Zur Illustration der Erweiterung des Modells '`vorne` *liegt immer in Bewegungsrichtung*' diene als Beispiel eine Situation, in der der Roboter den Auftrag erhält, sich solange vorwärtszubewegen bis er eine Wand wahrnimmt, stehen zu bleiben, sich dann um 90° vor der Wand nach `links` zu drehen, um dann seinen Weg nach `links` entlang der Wand fortzusetzen. Da die Wahrnehmungsrichtungsangaben der handlungsorientierten Wahrnehmungsmerkmale nach dem Modell '`vorne` *liegt immer in Bewegungsrichtung*' festgelegt werden, wird der Roboter unabhängig von der Bewegungsrichtung, die er zu Beginn seines Auftrages einschlägt, die Wand mit Sensoren der Sensorklasse `vorne` oder anders ausgedrückt in Richtung `vorne` wahrnehmen. Es ist naheliegend, diese Wahrnehmungsrichtung für den Zeitraum, in dem der Roboter bewegungslos vor der Wand stehen bleibt, beizubehalten, so daß das aktuelle `vorne` bei Ausführung der Handlungsanweisung `bleibe_stehen` erhalten bleibt. Demzufolge entspricht auch die im Auftrag gemeinte Vorstellung von `links` dem aktuellen `links` des Roboters, und es wird zur Durchführung der Rotation keine Umrechnung der Richtungsangabe des Auftrages erforderlich. Für die Durchführung der Drehbewegung sind zwei mögliche Vorgehensweisen zur Festlegung, was während und nach der Rotation als `vorne` angenommen wird, denkbar:

(A) Das aktuelle `vorne` bleibt während und nach der Rotation erhalten.

(B) Während und nach der Rotation wird die Drehrichtung zum neuen aktuellen `vorne`, also die ursprüngliche Richtung `links` bzw. `rechts`.

Sei zunächst Variante (B) angenommen. Mit dem Moment des Beginns der Drehbewegung ändert sich das aktuelle `vorne`, und zwar wird nun die ursprüngliche Richtung `links` als in Richtung `vorne` liegend betrachtet. Nach Durchführung der 90° Drehung liegt das aktuelle `vorne` in Richtung des ursprünglichen `hinten`, was für eine 90° Drehung nicht gerade intuitiv anmutet. Um gemäß seinem Auftrag seine Fahrt fortzusetzen, ist eine Bewegung in Richtung `rechts` notwendig, wobei die Wand in Richtung `hinten` wahrzunehmen ist — es sei daran erinnert, daß sich der Roboter vor der Rotation auf die Wand zubewegte! Im Moment der Vorwärtsbewegung wird das aktuelle `rechts` zum neuen `vorne`, und die Wand, die zuvor noch in Richtung `hinten` wahrgenommen wurde, ist nun `rechts` zu erwarten.

Sei im folgenden Variante (A) angenommen. Bei der Drehbewegung bleibt das aktuelle `vorne` erhalten. Nach Durchführung der 90° Drehung liegt das ursprüngliche `hinten` nun in Richtung `links`. Um gemäß seinem Auftrag seine Fahrt fortzusetzen, ist eine Bewegung in Richtung `vorne` notwendig, wobei die Wand, auf die sich der Roboter vor der Rotation zubewegte, in Richtung `rechts` wahrgenommen wird. Im Moment der Vorwärtsbewegung bleibt das aktuelle `vorne` erhalten, und die Wand wird weiterhin in Richtung `rechts` wahrgenommen.

Die Realisierung beider Varianten erfordert Richtungsumrechnungen in Abhängigkeit von der Durchführung bestimmter Handlungen. So muß beispielsweise die Bewegungsrichtungsangabe `links` des Auftrages für Variante (A) nach Durchführung der Linksrotation als `vorne` und für Variante (B) als `rechts` interpretiert werden. Wenn Handlungen nach Punkt 2) im Abschnitt 3.1 unter der Kontrolle von Wahrnehmungen durchgeführt werden sollen, gilt entsprechend das Gleiche auch für die Wahrnehmungsrichtungsangaben. Beispielsweise muß sich der Roboter zur Durchführung der 90° Rotation solange drehen, bis die Wand, die anfänglich `vorne` wahrgenommen wird, im Falle von Variante (A) durch die Sensorklasse `rechts` und für Variante (B) durch die Sensorklasse `hinten` registriert wird. Bei Wahl von Variante (B) muß demnach der im Abschnitt 2.2.1 vorgestellte Algorithmus zur Bestimmung der Sensoren einer Sensorklasse in geeigneter Form erweitert werden, während dies für Variante (A) nicht erforderlich ist, da in diesem Fall die Festlegung der Sensorklassen von der Rotation unberührt bleibt. Da zudem Variante (B) durchaus weniger intuitiv nachvollziehbar ist, wird Variante (A) zur Erweiterung des Modells '`vorne` *liegt immer in Bewegungsrichtung*' herangezogen.

Die notwendigen Umrechnungen von Richtungsangaben werden mit Hilfe von Hintergrundwissen dargestellt. Einzelheiten hierzu werden im Zusammenhang mit der im Abschnitt 3.4 vorgestellten Repräsentation operationaler Begriffe erörtert werden. In der Arbeitsphase wird es auch bei Verwendung von Variante (B) notwendig sein, die relativen Bewegungsrichtungsangaben — z.B. `vorne`, nach Durchführung der Rotation im Beispielauftrag — zur Ausführung von Handlungen in absolute Richtungsangaben umzurechnen, währenddessen dieser Vorgang in der Lernphase in umgekehrter Weise erfolgen muß. Wie dies problemlos möglich ist, wird im Abschnitt 3.2.2 erörtert werden. Nach den vorangegangenen Ausführungen ergibt sich folgendes Modell zur Festlegung der relativen Richtungsangaben:

- Die Richtung relativ `vorne` liegt für Vorwärtsbewegungen des Roboters stets in Bewegungsrichtung.

- Die Richtung relativ `vorne` ändert sich nicht, wenn der Roboter stehen bleibt.

- Die Seite des Roboters, die in Richtung relativ `vorne` liegt, zeigt auch nach Durchführung einer beliebigen Rotation die Richtung relativ `vorne` an.

Zum Abschluß dieses Abschnittes wird mit Abbildung 3.2 die Unterscheidung von absoluten und relativen Bewegungsrichtungen und ihre intrinsische bzw. extrinsische Interpretation nocheinmal an einem Beispiel illustriert. Der gestrichelte Pfeil gibt dabei die Richtung an, aus der das Vehikel ursprünglich gekommen ist. Weiterhin zeigt der schwarze Kreis am schematisch dargestellten Roboter die Lage von Sensor S_0 und damit stets die Seite des Roboters, die in Richtung absolut `vorne` liegt.

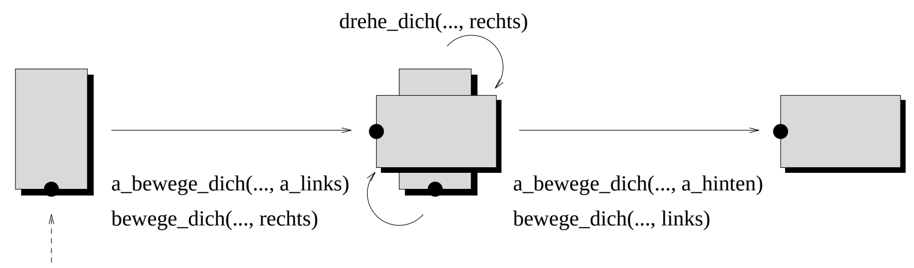


Abbildung 3.2: Beispiel für absolute und relative Bewegungsrichtungen

3.2.2 Berechnung von Basis-Handlungsmerkmalen

Nach der Vorstellung der Repräsentation von Basis-Handlungsmerkmalen im vergangenen Abschnitt, wird im vorliegenden Abschnitt im Sinne von Punkt 1) des Abschnittes 3.1 erörtert, wie Basis-Handlungsmerkmale inkrementell aus den vom Sensorkontroll- und Signalverarbeitungssystem des Roboters zur Verfügung gestellten Daten in der Lernphase ermittelt werden. Die Basis dieser Betrachtungen bilden die im Abschnitt 2.1.2 eingeführten rp/5-Fakten (Definition 2.4), wobei im weiteren vorausgesetzt wird, daß diese Fakten für jeden Meßzeitpunkt einer Fahrt verfügbar sind.

Im folgenden wird unter Punkt (A) das Basisverfahren zur Berechnung der Handlungsmerkmale (Algorithmus 3.1) erläutert. Die Aufgabe dieses Verfahrens ist die Bestimmung des Start- und Endzeitpunktes eines Basis-Handlungsmerkmals aus den in zeitlicher Folge sukzessive verarbeiteten rp/5-Fakten. Diese Zeitpunkte werden zusammen mit weiteren Informationen, wozu insbesondere der Typ (*keine Bewegung, Bewegung, Drehung*) des erkannten Basis-Handlungsmerkmals gehört, an den unter Punkt (B) vorgestellten Algorithmus zur Berechnung der Argumente des erkannten Merkmals weitergegeben (Algorithmus 3.2), wobei die Argumentberechnung in Abhängigkeit vom Merkmalstyp erfolgt. Zur Ermittlung der absoluten und relativen Bewegungsrichtung bei Vorwärtsbewegungen sowie zur Drehrichtung bei Rotationen werden von diesem Algorithmus weitere Verfahren herangezogen. Im einzelnen werden unter Punkt (C) die Ideen zur Bestimmung der absoluten Bewegungsrichtung (Algorithmus 3.3) erörtert, während die Ausführungen unter Punkt (D) der Erläuterung der Berechnung der relativen Bewegungsrichtung (Algorithmus 3.4) dienen. Abschließend wird unter Punkt (E) ein Verfahren zur Bestimmung der Drehrichtung und der dabei zurückgelegten Distanz in Gradeinheiten vorgestellt (Algorithmus 3.5).

(A) Basisverfahren zur Berechnung von Handlungsmerkmalen

Das Ziel des Verfahrens ist es, die Start- und Endzeitpunkte von Basis-Handlungsmerkmalen aus den während einer kontrollierten Roboterfahrt in der Lernphase in zeitlicher Folge eingelesenen rp/5-Fakten zu ermitteln. Jeder rp/5-Fakt beschreibt neben der Roboterposition auch die Roboterorientierung *RAlpha* (Definition 2.1) zum betrachteten Zeitpunkt. Über Veränderungen der Roboterpositionen und -orientierungen im zeitlichen Verlauf der Fahrt lassen sich Rückschlüsse auf die durchgeführten Basis-Handlungen ableiten. Bleibt der Roboter beispielsweise während seiner Fahrt eine zeitlang stehen, so werden alle zu diesem Zeitabschnitt korrespondierenden rp/5-Fakten die gleiche Roboterposition und -orientierung beschreiben. Hingegen werden Vorwärtsbewegungen des Roboters bei unterbleibender simultaner Rotation durch konstante Werte für *RAlpha* und sich von Zeitpunkt zu Zeitpunkt verändernde Positionsangaben charakterisiert. Umgekehrt verhält es sich bei Rotationen. Für die Dauer einer Drehbewegung bleibt die Position des Roboters konstant, während sich seine Orientierung verändert.

Eingabe: rp/5-Fakten
Ausgabe: Basis-Handlungsmerkmale

- Die zu einer Roboterfahrt korrespondierenden rp/5-Fakten (Definition 2.4) werden aufsteigend nach ihren zugehörigen Zeitpunkten geordnet nacheinander eingelesen. Der Algorithmus ermittelt aus den sukzessive eingelesenen rp/5-Fakten inkrementell die zur Roboterfahrt korrespondierenden Basis-Handlungsmerkmale. Die Ausgabe der Basis-Handlungsmerkmale erfolgt nacheinander in der Reihenfolge ihrer Berechnung.

BEGIN

(1) • Lies den ersten Fakt $rp(Trc_{Start}, T_{Start}, RX_{Start}, RY_{Start}, RAlpha_{Start})$ ein.
• Falls kein Fakt eingelesen werden konnte, beende das Verfahren.
(2) • Lese den nächsten Fakt $rp(Trc_i, T_i, RX_i, RY_i, RAlpha_i)$ ein.
• Falls kein weiterer Fakt eingelesen werden konnte, beende das Verfahren.
• $Trc_{Ende} := Trc_i; T_{Ende} := T_i; RX_{Ende} := RX_i; RY_{Ende} := RY_i; RAlpha_{Ende} := RAlpha_i$
• $Trc_n := Trc_i; T_n := T_i; RX_n := RX_i; RY_n := RY_i; RAlpha_n := RAlpha_i$
(3) CONDITION
(a) $(RAlpha_{Start} = RAlpha_i \quad AND \quad (RX_{Start}, RY_{Start}) = (RX_i, RY_i))$
THEN • Status:=*keine Bewegung*;
• DO
∘ Lese den nächsten Fakt $rp(Trc_n, T_n, RX_n, RY_n, RAlpha_n)$ ein.
∘ Falls kein Fakt eingelesen werden konnte, beende die WHILE-Schleife.
∘ IF $(RAlpha_{Start} = RAlpha_n \quad AND \quad (RX_{Start}, RY_{Start}) = (RX_n, RY_n))$
THEN $Trc_{Ende} := Trc_n; T_{Ende} := T_n; RX_{Ende} := RX_n;$
$RY_{Ende} := RY_n; RAlpha_{Ende} := RAlpha_n$
WHILE $(RAlpha_{Start} = RAlpha_n \quad AND \quad (RX_{Start}, RY_{Start}) = (RX_n, RY_n))$

(b) $(RAlpha_{Start} = RAlpha_i \quad AND \quad (RX_{Start}, RY_{Start}) \neq (RX_i, RY_i))$
THEN • Status:=*Bewegung*;
• DO
... (* analog zu DO-WHILE in (3a) *)
WHILE $(RAlpha_{Start} = RAlpha_n \quad AND \quad (RX_{Start}, RY_{Start}) \neq (RX_n, RY_n))$

(c) $(RAlpha_{Start} \neq RAlpha_i \quad AND \quad (RX_{Start}, RY_{Start}) = (RX_i, RY_i))$
THEN • Status:=*Drehbewegung*;
• DO
... (* analog zu DO-WHILE in (3a) *)
WHILE $(RAlpha_{Start} \neq RAlpha_n \quad AND \quad (RX_{Start}, RY_{Start}) = (RX_n, RY_n))$

(d) $(RAlpha_{Start} \neq RAlpha_i \quad AND \quad (RX_{Start}, RY_{Start}) \neq (RX_i, RY_i))$
THEN • Status:=*nicht erlaubt*;
• DO
... (* analog zu DO-WHILE in (3a) *)
WHILE $(RAlpha_{Start} \neq RAlpha_n \quad AND \quad (RX_{Start}, RY_{Start}) \neq (RX_n, RY_n))$
END;(*CONDITION*)
(4) • Berechne das Basis-Handlungsmerkmal gemäß **Algorithmus 3.2**.
• Eingabe für Algorithmus 3.2 sind: Status,$T_i, RAlpha_i, Trc_{Start}, T_{Start}, RX_{Start}, RY_{Start}, RAlpha_{Start}, T_{Ende}, RX_{Ende}, RY_{Ende}$ und $RAlpha_{Ende}$.
(5) **Falls $T_n = T_{Ende}$ gilt, beende das Verfahren !**
(6) • $Trc_{Start} := Trc_{Ende}; T_{Start} := T_{Ende}; RX_{Start} := RX_{Ende};$
$RY_{Start} := RY_{Ende}; RAlpha_{Start} := RAlpha_{Ende}$
• $Trc_i := Trc_n; T_i := T_n; RX_i := RX_n; RY_i := RY_n; RAlpha_i := RAlpha_n$
• $Trc_{Ende} := Trc_i; T_{Ende} := T_i; RX_{Ende} := RX_i; RY_{Ende} := RY_i; RAlpha_{Ende} := RAlpha_i$
• Die Variablen $Trc_n, T_n, RX_n, RY_n, RAlpha_n$ behalten ihre Werte bei.
(7) **Fahre mit Schritt(3) des Algorithmus fort.**
END;(*Algorithmus*)

Algorithmus 3.1: Basisverfahren zur Berechnung von Handlungsmerkmalen

Diese Überlegungen bilden die Basis für Algorithmus 3.1. Die Schritte 1) und 2) im Algorithmus werden nur einmal zu Beginn der Roboterfahrt ausgeführt und lesen die zu den ersten beiden Zeitpunkten der Fahrt gehörenden rp/5-Fakten ein. In diesen Schritten werden den Variablen mit den Indizes *Start, i, n* und *Ende* die Werte der durch diese Fakten beschriebenen Zeitpunkte, Roboterpositionen sowie Roboterorientierungen zugewiesen. Der Einfachheit halber werden diese Variablen im folgenden auch nur durch ihre Indizes benannt werden. Im einzelnen erhalten die *Start*-Variablen die Werte des zum ersten Zeitpunkt und die *i-, n-* sowie *Ende-* Variablen die Werte des zum zweiten Zeitpunkt der Fahrt korrespondierenden rp/5-Faktes. Schritt 1) und 2) können als Initialisierungsphase von Algorithmus 3.1 betrachtet werden. Abbildung 3.3 zeigt die somit beschriebenen Variablenbindungen noch einmal schematisch im Teil A).

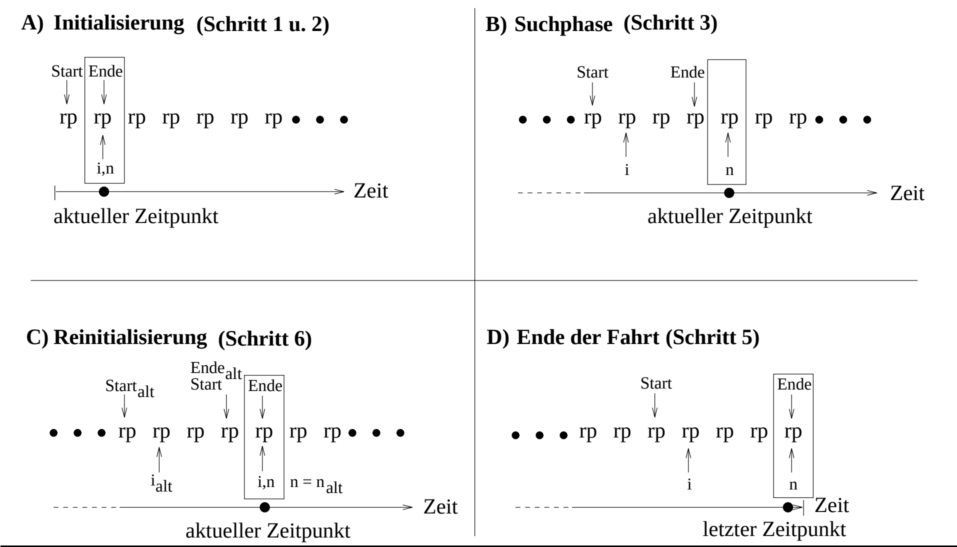


Abbildung 3.3: Illustration der Variablenbindungen im Algorithmus 4.1

Im dritten Schritt von Algorithmus 3.1 werden zunächst die Werte der *Start-* und *i*-Variablen im Hinblick auf die oben beschriebenen Bedingungen zur Bestimmung der zwischen den ersten beiden Zeitpunkten ausgeführten Basis-Handlung miteinander verglichen. Ist die Roboterorientierung und die Roboterposition zu den beiden betrachteten Zeitpunkten identisch, so wird der Variablen *Status* der Wert *keine Bewegung* zugewiesen. In diesem Fall liegen die betrachteten Zeitpunkte am Anfang eines Zeitabschnittes der Fahrt, in dem der Roboter keine Bewegung durchführt, also das Basis-Handlungsmerkmal `bleibe_stehen/3` ausführt. Analog gilt für die Schritte 3a), 3c) und 3d), daß zwischen Zeitpunkt T_{Start} und T_i eine Vorwärtsbewegung, eine Drehung und eine Vorwärtsbewegung mit simultaner Drehung, also eine nicht erlaubte Bewegung durchgeführt wird. Trifft eine dieser Bedingungen zu, so werden solange weitere rp/5-Fakten eingelesen, bis sie schließlich scheitert. Dabei werden stets den n-Variablen die Werte des aktuell eingelesenen rp/5-Faktes zugewiesen. Die alten Variablenwerte werden dabei überschrieben. Für den Fall, daß die betrachtete Bedingung für die Werte der *Start*- und n-Variablen noch Gültigkeit besitzt, werden auch die *Ende*-Variablen auf die aktuell eingelesenen Werte gesetzt. Nach Ablauf dieser als Suchphase zu charakterisierenden Arbeitsphase des Algorithmus hat sich die in Abbildung 3.3 B) schematisch dargestellte Situation eingestellt.

Mit T_{Start} und T_{Ende} sind nach der Suchphase die Anfangs- und Endzeitpunkte eines Basis-Handlungsmerkmals ermittelt, dessen Typ durch die Variable *Status* gekennzeichnet ist. Für den Fall, daß zwischen T_{Start} und T_{Ende} mindestens ein weiterer Zeitpunkt der Roboterfahrt bekannt ist, werden die Werte des zum zweiten Zeitpunkt korrespondierenden rp/5-Faktes des betrachteten Zeitabschnittes durch die i-Variablen gespeichert. Ansonsten entsprechen die Werte der i-Variablen den Werten der *Ende*-Variablen. Diese Unterscheidung wird für die durch Algorithmus 3.5 beschriebene Ermittlung der Drehrichtung einer Rotation noch von Bedeutung sein.

Die ermittelten Start- und Endzeitpunkte des erkannten Basis-Handlungsmerkmals vom Typ *Status* dienen zusammen mit den zugehörigen Roboterpositionen und -orientierungen, sowie den Werten der i-Variablen als Eingabe für Algorithmus 3.2 zur Ermittlung der Argumente des Handlungsmerkmals. Der Aufruf von Algorithmus 3.2 erfolgt im Schritt 4) von Algorithmus 3.1. Sollte der Fall aufgetreten sein, daß die Werte der *Ende*-Variablen gleich den Werten der n-Variablen sind (siehe Abbildung 3.3 D)), so ist die WHILE-Schleife einer der Schritte 3a), 3b), 3c) oder 3d) vorzeitig abgebrochen worden, da kein weiterer rp/5-Fakt eingelesen werden konnte, so daß Schritt 5) des Algorithmus das Verfahren in dieser Situation korrekt beendet. Sollte dieser Fall nicht eingetreten sein, so erfolgt im Schritt 6) die in Abbildung 3.3 C) dargestellte Reinitialisierung der *Start-*, *i-* und *Ende*-Variablen. Die *Start*-Variablen erhalten dabei die Werte der *Ende*-Variablen, während den *i*- und *Ende*-Variablen die Werte der n-Variablen zugewiesen werden, die ihre alten Werte beibehalten. Nach Durchführung von Schritt 6) liegt die gleiche Situation wie zu Beginn nach der Durchführung von Schritt 1) und 2) vor, so daß das Verfahren im Schritt 7) des Algorithmus mit Schritt 3) wie beschrieben fortfahren kann.

(B) Verfahren zur Bestimmung der Argumente

Nachdem unter Punkt (A) erläutert wird, wie Algorithmus 3.1 die zu den Start- und Endzeitpunkten eines Basis-Handlungsmerkmals korrespondierenden rp/5-Fakten bestimmt, wird im folgenden beschrieben, wie Algorithmus 3.2 aus diesen Fakten die zum Basis-Handlungsmerkmal gehörenden Argumente berechnet. Eingabe für Algorithmus 3.2 sind die Werte der Variablen *Status* sowie der *i, Start-* und *Ende*-Variablen im Schritt 4) von Algorithmus 3.1. Abhängig vom Wert der Variablen *Status* sind vier grundsätzliche Vorgehensweisen zu unterscheiden. Für die Fälle, daß die Variable *Status* die Werte *keine Bewegung* oder *nicht erlaubt* besitzt, sind keine weiteren Argumente zu berechnen, so daß hier direkt die Ausgabe des Basis-Handlungsmerkmals `bleibe_stehen`($Trc_{Start}, T_{Start}, T_{Ende}$) bzw. der Information über eine nicht erlaubte Aktion in Form des Faktes `nicht_erlaubte_Aktion`($Trc_{Start}, T_{Start}, T_{Ende}$) erfolgen kann.

Eingabe: Status,T_i, $RAlpha_i$, Trc_{Start}, T_{Start}, RX_{Start}, RY_{Start}, $RAlpha_{Start}$, T_{Ende}, RX_{Ende}, RY_{Ende} und $RAlpha_{Ende}$
Ausgabe: Basis-Handlungsmerkmal

- Während Algorithmus 3.1 den Start- und Endzeitpunkt (T_{Start} und T_{Ende}) eines Basis-Handlungsmerkmals sowie seinen Typ (*Status*) ermittelt hat, berechnet der vorliegende Algorithmus die zum Merkmal gehörenden Argumente und gibt das dann vollständige Basis-Handlungsmerkmals aus.

BEGIN

CASE Status OF

keine Bewegung :
- Gib das Basis-Handlungsmerkmal `bleibe_stehen`($Trc_{Start}, T_{Start}, T_{Ende}$) aus.

nicht erlaubt :
- Gebe `nicht_erlaubte_Aktion`($Trc_{Start}, T_{Start}, T_{Ende}$) aus.

Bewegung :
- Berechne die zwischen Position (RX_{Start}, RY_{Start}) und (RX_{Ende}, RY_{Ende}) gefahrene Geschwindigkeit G.
- Ordne G eine Geschwindigkeitsklasse GK zu (siehe Abschnitt 3.2.1).
- Berechne für die Bewegung von (RX_{Start}, RY_{Start}) nach (RX_{Ende}, RY_{Ende}) die Bewegungsorientierung *BAlpha* (Definition 2.6).
- Berechne **gemäß Algorithmus 3.3** die absolute Bewegungsrichtung *AbsRichtung* der Fahrt von (RX_{Start}, RY_{Start}) nach (RX_{Ende}, RY_{Ende}) aus $RAlpha_{Start}$ und *BAlpha*.
- Falls die letzte absolute Bewegungsrichtung *LAbsRichtung* noch unbestimmt ist, setze die relative Bewegungsrichtung *RelRichtung* gleich *AbsRichtung* ohne den Präfix 'a_'. Ansonsten berechne *RelRichtung* **gemäß Algorithmus 3.4** aus *AbsRichtung* und *LAbsRichtung*.
- Speichere *AbsRichtung* als *LAbsRichtung*.
- Gebe das Basis-Handlungsmerkmal `bewege_dich`($Trc_{Start}, T_{Start}, T_{Ende}$, *GK*, *RelRichtung*) aus.

Drehung :
- Falls $T_i = T_{Ende}$ gilt, berechne die Drehrichtung *DRichtung* der zwischen T_{Start} und T_{Ende} erfolgten Drehung sowie die dabei zurückgelegte Distanz in Grad *DiffGrad* **gemäß Algorithmus 3.5** aus $RAlpha_{Start}$ und $RAlpha_{Ende}$. Ansonsten beziehe **gemäß Algorithmus 3.5** auch $RAlpha_i$ zur Berechnung mit ein.
- Berechne die Rotationsgeschwindigkeit *RG* der zwischen T_{Start} und T_{Ende} erfolgten Drehung aus *DiffGrad*, T_{Start} und T_{Ende}.
- Ordne *RG* eine Rotationsgeschwindigkeitsklasse *RGK* zu (siehe Abschnitt 3.2.1).
- Gebe das Basis-Handlungsmerkmal `drehe_dich`($Trc_{Start}, T_{Start}, T_{Ende}$, *RGK*, *DRichtung*) aus.

END;(*CASE*)
END;(*Algorithmus*)

Algorithmus 3.2: Berechnung eines Basis-Handlungsmerkmals

Sei im weiteren zunächst der Fall betrachtet, daß die Variable *Status* den Wert *Bewegung* besitzt, was gleichbedeutend damit ist, daß zwischen Zeitpunkt T_{Start} und T_{Ende} der Roboterfahrt eine Vorwärtsbewegung stattfindet. Zur Ausgabe des Basis-Handlungsmerkmals `bewege_dich`/5 müssen für diesen Fall noch als Argumente die Geschwindigkeitsklasse und die relative Bewegungsrichtung bestimmt werden. Da die Position des Roboters in der Umgebung sowohl für den Zeitpunkt T_{Start} als auch für den Zeitpunkt T_{Ende} bekannt ist, kann die im betrachteten Zeitabschnitt der Roboterfahrt zurückgelegte Distanz in Metern leicht ermittelt werden. Weiterhin ist bekannt, wieviele Sensormessungen pro Sekunde vom Sensorkontroll- und Signalverarbeitungssystem des Roboters durchgeführt werden, so daß über die Anzahl der zwischen T_{Start} und T_{Ende} liegenden Meßzeitpunkte auf die zwischen T_{Start} und T_{Ende} vergangene Zeit in Sekunden geschlossen werden kann, womit auch die gefahrene Geschwindigkeit G im betrachteten Zeitraum bestimmt ist. Die Zuordnung der Geschwindigkeit G zu einer Geschwindigkeitsklasse GK kann dann beispielsweise wie im Abschnitt 3.2.1 beschrieben erfolgen.

In einem nächsten Schritt wird die bei der Vorwärtsbewegung eingehaltene Bewegungsorientierung *BAlpha* (Definition 2.6) berechnet. *BAlpha* und die Roboterorientierung $RAlpha_{Start}$ (Definition 2.1) im betrachteten Zeitraum dienen als Eingabe für Algorithmus 3.3, der die absolute Bewegungsrichtung *AbsRichtung* bestimmt. Im Algorithmus 3.2 wird dabei jeweils die letzte bestimmte absolute Bewegungsrichtung in der Variablen *LAbsRichtung* gespeichert, da im Algorithmus 3.4 die Berechnung der relativen Bewegungsrichtungsangabe in Abhängigkeit von der letzten absoluten und der aktuellen absoluten Bewegungsrichtung erfolgt.

Vor dem Hintergrund, daß bei Verwendung des im Abschnitt 3.2.1 beschriebenen Modells zur Bestimmung der relativen Richtungsangaben die geforderte Umrechnung der relativen in absolute Bewegungsrichtungen in der Arbeitsphase nur dann eindeutig beschrieben werden kann, wenn zu Beginn verschiedener Roboterfahrten die Relation zwischen relativen und absoluten Richtungsangaben stets gleich festgelegt wird, wird *RelAbsRichtung* für die erste Vorwärtsbewegung einer Fahrt mit der dabei eingeschlagenen absoluten Bewegungsrichtung gleichgesetzt. Durch diese Idee ist es möglich, eine Relation zwischen den aktuellen absoluten, den letzten absoluten und den relativen Bewegungsrichtungen anzugeben, nach der sowohl die relative Bewegungsrichtung durch die Angabe der absoluten und der letzten absoluten Bewegungsrichtung, als auch die absolute Bewegungsrichtung durch die Angabe der relativen und der letzten absoluten Bewegungsrichtung eindeutig bestimmt ist. Im Algorithmus 3.4 wird diese Überlegung zur Berechnung von *RelAbsRichtung* ausgenutzt werden. Durch eine kleine Abwandlung von Algorithmus 3.4 kann der für die Arbeitsphase benötigte Algorithmus zur Bestimmung der absoluten Richtungsangabe einer auszuführenden Vorwärtsbewegung gewonnen werden. Unter Punkt D) wird auf diese Betrachtungen noch einmal Bezug genommen werden.

Nachdem mit Hilfe von Algorithmus 3.4 die relative Bewegungsrichtung der zwischen T_{Start} und T_{Ende} liegenden Vorwärtsbewegung aus *AbsRichtung* und *LAbsRichtung* ermittelt ist, kann die Ausgabe des Basis-Handlungsmerkmals `bewege_dich`($Trc_{Start}, T_{Start}, T_{Ende}, GK, RelRichtung$) erfolgen. Das zugehörige Basis-Handlungsmerkmal `a_bewege_dich`(Trc_{Start},-

T_{Start}, T_{Ende},*GK*,*AbsRichtung*) wird hingegen nicht mit ausgegeben, da es für die Lernphase des Roboters keine weitere Rolle spielt.

Sei nun abschließend der Fall angenommen, daß die Variable *Status* den Wert *Drehung* besitzt. Für diesen Fall findet zwischen den Zeitpunkten T_{Start} und T_{Ende} der Roboterfahrt eine Rotation statt. Zur Ausgabe des zugehörigen Basis-Handlungsmerkmals `drehe_dich/5` wird somit die Berechnung der Drehrichtung *DRichtung* und der Rotationsgeschwindigkeitsklasse *RGK* notwendig. *DRichtung* wird dabei mit Hilfe von Algorithmus 3.5 ermittelt, der zudem noch die bei der Rotation zurückgelegte Distanz in Gradeinheiten *DiffGrad* liefert. Eingabe für Algorithmus 3.5 ist die Roboterorientierung zum Zeitpunkt T_{Start} und zum Zeitpunkt T_{Ende}, sowie ggf. die Roboterorientierung $RAlpha_i$ zum Zeitpunkt T_i, falls $T_i \neq T_{Ende}$ ist. Einzelheiten dieser Berechnungen werden unter Punkt E) erörtert werden. Die Ermittlung der Rotationsgeschwindigkeit *RG* in Grad/Sekunde kann analog zur Bestimmung der Bewegungsgeschwindigkeit über die Angabe der Sensormessungen pro Sekunde und die Anzahl der zwischen T_{Start} und T_{Ende} liegenden Meßzeitpunkte erfolgen. Der Rotationsgeschwindigkeit *RG* wird dann beispielsweise, wie im Abschnitt 3.2.1 beschrieben, eine Rotationsgeschwindigkeitsklasse *RGK* zugeordnet. Mit der Festlegung von *RGK* kann schließlich das Basis-Handlungsmerkmal `drehe_dich`(Trc_{Start}, T_{Start}, T_{Ende},*RGK*,*DRichtung*) ausgegeben werden.

(C) Berechnung der absoluten Bewegungsrichtung

Mit der Berechnung der Bewegungsorientierung *BAlpha* (Definition 2.6) durch Algorithmus 3.2 kann die absolute Bewegungsrichtung *AbsRichtung* der durch Algorithmus 3.1 erkannten Vorwärtsbewegung im betrachteten Zeitabschnitt der Roboterfahrt mit Hilfe der für diesen Zeitabschnitt bekannten Roboterorientierung $RAlpha_{Start}$ (Definition 2.1) erfolgen. Die Notwendigkeit der Unterscheidung zwischen absoluten und relativen Bewegungsrichtungsangaben wird bereits im Abschnitt 3.1 erörtert, während im Abschnitt 3.2.1 zur Festlegung eines robotereigenen Bezugssystems die Roboterseite, an der Sensor S_0 montiert ist, als stets in Richtung absolut `vorne` liegend definiert wird. Da weiterhin im Abschnitt 2.1 die Roboterorientierung über die Sensororientierung (Definition 2.2) von Sensor S_0 festgelegt wird, ist es zur Bestimmung der absoluten Bewegungsrichtung zunächst einmal naheliegend, die absolute Bewegungsrichtung über einen Vergleich der Roboterorientierung und der Bewegungsorientierung zu ermitteln. Entspricht beispielsweise für den betrachteten Zeitabschnitt die Bewegungsorientierung der Roboterorientierung, so ist dies gleichbedeutend damit, daß die Vorwärtsbewegung in Richtung der Roboterseite mit montiertem Sensor S_0 erfolgt, also in Richtung `a_vorne`. Analog gilt für eine Vorwärtsbewegung in Richtung `a_hinten`, daß die Bewegungsorientierung gerade der um 180° verschobenen Roboterorientierung entspricht, während eine Bewegung in Richtung `a_links` bzw. in Richtung `a_rechts` durch eine gegenüber der Bewegungsorientierung um 90° bzw. 270° nach links verschobene Roboterorientierung gekennzeichnet ist. Es gelten demnach die folgenden Beziehungen:

$$
\begin{aligned}
AbsRichtung &= \texttt{a\_vorne} &\Leftrightarrow BAlpha &= RAlpha_{Start}\\
AbsRichtung &= \texttt{a\_hinten} &\Leftrightarrow BAlpha &= (RAlpha_{Start} + 180) mod\, 360\\
AbsRichtung &= \texttt{a\_links} &\Leftrightarrow BAlpha &= (RAlpha_{Start} + 90)\ mod\, 360\\
AbsRichtung &= \texttt{a\_rechts} &\Leftrightarrow BAlpha &= (RAlpha_{Start} + 270) mod\, 360
\end{aligned}
$$

Diese Betrachtungen bilden den Kern von Algorithmus 3.3 zur Berechnung der absoluten Bewegungsrichtung. Im angegebenen Verfahren findet zudem aber ein bei realen Roboterfahrten oftmals auftretender Umstand Berücksichtigung, der bei direkter Verwendung der angegebenen Beziehungen eine Bewegungsrichtungsberechnung unmöglich macht. So kann es in der Praxis passieren, daß der Roboter aufgrund von Bodenunebenheiten bei einer Vorwärtsbewegung trotz Einhaltung einer bestimmten Bewegungsrichtung vom Kurs abdriftet. Dies hat zur Folge, daß die berechnete Bewegungsorientierung nicht mehr exakt den obigen Gleichungen entspricht, da daß Kontrollsystem des Roboters interessanter Weise in der Lage ist, Abweichungen von der Ideallinie für die Positionsbestimmung zu berücksichtigen.

Eingabe: $BAlpha$ und $RAlpha_{Start}$
Ausgabe: $AbsRichtung$

- Der Algorithmus berechnet für eine Bewegung des Roboters in Abhängigkeit von der Roboterorientierung $RAlpha_{Start}$ und der Bewegungsorientierung $BAlpha$ die absolute Bewegungsrichtung $AbsRichtung$, in die sich der Roboter in dem im Algorithmus 3.1 ermittelten Zeitraum vorwärtsbewegt.
- Mit Δ sei die maximal zulässige Abweichung des Wertes für $BAlpha$ vom Idealwert bezeichnet, für die noch eine Zuordnung einer absoluten Bewegungsrichtung erlaubt sein soll. Idealwerte für $BAlpha$ sind bei einer Bewegung in Richtung: `a_vorne`:$RAlpha$; `a_links`:$RAlpha + 90°$; `a_hinten`:$RAlpha + 180°$; `a_rechts`:$RAlpha + 270°$. Für $\Delta = 45°$ erfolgt immer eine Richtungszuordnung.

BEGIN

CONDITION
(a) $(2 * \Delta \geq (BAlpha - (RAlpha_{Start} - \Delta) + 360) mod 360)$
THEN • $AbsRichtung$:= `a_vorne`
(b) $(2 * \Delta \geq (BAlpha - (RAlpha_{Start} + 90 - \Delta) + 360) mod 360)$
THEN • $AbsRichtung$:= `a_links`
(c) $(2 * \Delta \geq (BAlpha - (RAlpha_{Start} + 180 - \Delta) + 360) mod 360)$
THEN • $AbsRichtung$:= `a_hinten`
(d) $(2 * \Delta \geq (BAlpha - (RAlpha_{Start} + 270 - \Delta) + 360) mod 360)$
THEN • $AbsRichtung$:= `a_rechts`
END;(*CONDITION*)
END;(*Algorithmus*)

Algorithmus 3.3: Berechnung der absoluten Bewegungsrichtung

Im Algorithmus 3.3 werden zur Lösung dieses Problems Abweichungen der berechneten *BAlpha*-Werte von den nach oben stehenden Gleichungen zu erwartenden Idealwerten zugelassen, die noch in einem über einen Parameter festlegbaren Toleranzbereich liegen. Abbildung 3.4 veranschaulicht diese Idee. Mit Δ sei dabei die maximal zulässige Abweichung des berechneten Wertes für *BAlpha* vom Idealwert bezeichnet, für die noch eine Zuordnung einer absoluten Bewegungsrichtung erfolgen soll. Der maximale Wert für Δ ist 45°. In diesem Fall wird, wie in Abbildung 3.4 dargestellt, jeglicher Vorwärtsbewegung eine eindeutige absolute Bewegungsrichtungsangabe zugewiesen.

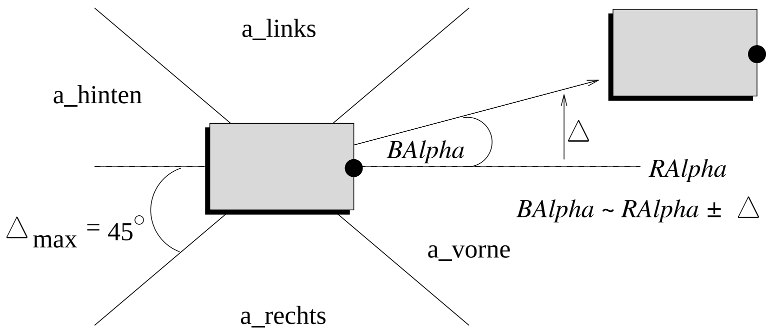


Abbildung 3.4: Idee zur Lösung des Abdriftproblems

(D) Berechnung der relativen Bewegungsrichtung

Nachdem im Abschnitt 3.2.1 erörterten Modell zur Festlegung der relativen Bewegungsrichtungsangaben liegt relativ `vorne` für Vorwärtsbewegungen des Roboters stets in Bewegungsrichtung, während die nach dieser Vorstellung betrachtete vordere Seite des Roboters für Rotationen und ein bewegungsloses Verharren an einer Position während und nach der Durchführung dieser Handlungen weiterhin als Bezugssystem zur Bestimmung der relativen Richtungsangaben dient. Da im Rahmen dieses Modells zu Beginn einer Roboterfahrt — also noch bevor der Roboter eine erste Vorwärtsbewegung durchführt — die Richtung relativ `vorne` und damit auch `hinten`, `rechts`, `links`, `links_vorne` etc. noch unbestimmt ist, obgleich schon in dieser Anfangsphase Wahrnehmungen registriert werden können, die im Sinne von Punkt 2) des Abschnittes 3.1 auf höheren Abstraktionsebenen der zu bildenden Begriffshierarchie Einfluß auf die Wahl der im folgenden möglichen Handlungen ausüben sollen, werden die relativen Richtungsangaben für diese Anfangsphase durch die absoluten Richtungsangaben definiert. Für den unter Punkt B) beschriebenen Algorithmus 3.2 zur Argumentberechnung der erkannten Basis-Handlungsmerkmale wird folglich für die erste Vorwärtsbewegung die relative Bewegungsrichtung gleich der absoluten Richtungsangabe gesetzt.

Das Verhältnis zwischen relativen und absoluten Richtungsangaben ist somit zu Beginn verschiedener Roboterfahrten stets gleich und verändert sich bei Verwendung des erörterten Modells zur Festlegung der relativen Bewegungsrichtungsangaben während der Fahrt in gesetzmäßiger Weise. Da die Anzahl der möglichen Bewegungsrichtungen begrenzt ist, möglich sind jeweils 4 relative und absolute Bewegungsrichtungen, ist auch die Anzahl der verschiedenen möglichen Beziehungen zwischen relativen und absoluten Richtungsangaben (`a_vorne` ≡ `links`, `a_links` ≡ `rechts`, `a_links` ≡ `links`, ...) begrenzt. Die Idee des im folgenden vorgestellten Algorithmus 3.4 zur Bestimmung der relativen Bewegungsrichtung ist nun, die für die betrachtete Situation zutreffende Beziehung zu erkennen, um auf diese Weise die relative Bewegungsrichtung zu berechnen. Die Erkennung der richtigen Beziehung ist aufgrund des verwendeten Bewegungsrichtungsmodells über die letzte absolute Bewegungsrichtung eindeutig, da relativ `vorne` für Rotationen und Stehen auf einer Stelle erhalten bleibt. Es läßt sich demnach eine dreistellige Relation zwischen den letzten absoluten, den betrachteten absoluten und den zu ermittelnden relativen Bewegungsrichtungen aufstellen, die sowohl zur Berechnung der relativen Bewegungsrichtungen durch die Angabe der absoluten und der letzten absoluten Bewegungsrichtungen, als auch der absoluten Bewegungsrichtungen durch die Angabe der relativen und der letzten absoluten Bewegungsrichtungen verwendet werden kann. Abbildung 3.5 zeigt diese Relation.

R(a_vorne, a_vorne, vorne)	R(a_rechts, a_vorne, links)
R(a_vorne, a_hinten, hinten)	R(a_rechts, a_hinten, rechts)
R(a_vorne, a_rechts, rechts)	R(a_rechts, a_rechts, vorne)
R(a_vorne, a_links, links)	R(a_rechts, a_links, hinten)
R(a_hinten, a_vorne, hinten)	R(a_links, a_vorne, rechts)
R(a_hinten, a_hinten, vorne)	R(a_links, a_hinten, links)
R(a_hinten, a_rechts, links)	R(a_links, a_rechts, hinten)
R(a_hinten, a_links, rechts)	R(a_links, a_links, vorne)

Abbildung 3.5: Relation zwischen Bewegungsrichtungen

Algorithmus 3.4 entspricht einer direkten Übersetzung der in Abbildung 3.5 dargestellten Relation R in eine `IF-THEN`-Struktur. Werden die *AbsRichtung* betreffenden Teile der `IF`-Bedingungen mit den `THEN`-Teilen vertauscht, und die `IF-THEN`-Anweisungen zur Erhaltung der `CONDITION`-Anweisung von Algorithmus 3.4 neu geordnet, so ergibt sich der für die Arbeitsphase benötigte Algorithmus zur Berechnung der absoluten Bewegungsrichtungen, die nach Abschnitt 3.2.1 zur Durchführung einer Vorwärtsbewegung erforderlich sind.

(E) Berechnung der Drehrichtung

Zum Abschluß dieses Abschnittes bleibt noch Algorithmus 3.5 vorzustellen, der die Drehrichtung *Drehrichtung* von Rotationen und die dabei zurückgelegte Distanz in Gradeinheiten *DiffGrad* für den im Algorithmus 3.1 ermittelten Zeitraum der Roboterfahrt berechnet. Als Eingabe für Algorithmus

Eingabe: $AbsRichtung$ und $LAbsRichtung$
Ausgabe: $RelRichtung$

- Eingabe ist die absolute Bewegungsrichtung $AbsRichtung$ für den im Algorithmus 3.1 ermittelten Zeitraum einer Roboterfahrt, sowie die vor diesem Zeitraum liegende letzte absolute Bewegungsrichtung $LAbsRichtung$. Der Algorithmus berechnet aus diesen Angaben die relative Bewegungsrichtung $RelRichtung$ des Roboters in der betrachteten Zeitspanne.

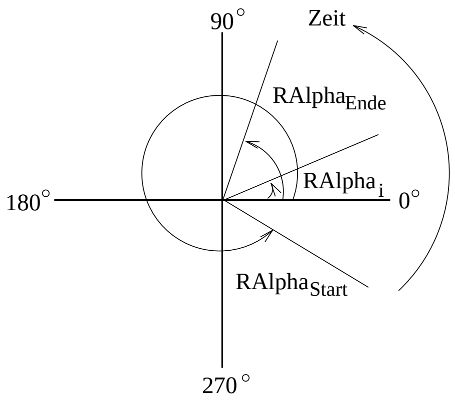
```
BEGIN
    CONDITION
        (a) ((LAbsRichtung = a_vorne AND AbsRichtung = a_vorne) OR
             (LAbsRichtung = a_hinten AND AbsRichtung = a_hinten) OR
             (LAbsRichtung = a_rechts AND AbsRichtung = a_rechts) OR
             (LAbsRichtung = a_links AND AbsRichtung = a_links))
            THEN • RelRichtung := vorne
        (b) ((LAbsRichtung = a_vorne AND AbsRichtung = a_hinten) OR
             (LAbsRichtung = a_hinten AND AbsRichtung = a_vorne) OR
             (LAbsRichtung = a_rechts AND AbsRichtung = a_links) OR
             (LAbsRichtung = a_links AND AbsRichtung = a_rechts))
            THEN • RelRichtung := hinten
        (c) ((LAbsRichtung = a_vorne AND AbsRichtung = a_rechts) OR
             (LAbsRichtung = a_hinten AND AbsRichtung = a_links) OR
             (LAbsRichtung = a_rechts AND AbsRichtung = a_hinten) OR
             (LAbsRichtung = a_links AND AbsRichtung = a_vorne))
            THEN • RelRichtung := rechts
        (d) ((LAbsRichtung = a_vorne AND AbsRichtung = a_links) OR
             (LAbsRichtung = a_hinten AND AbsRichtung = a_rechts) OR
             (LAbsRichtung = a_rechts AND AbsRichtung = a_vorne) OR
             (LAbsRichtung = a_links AND AbsRichtung = a_hinten))
            THEN • RelRichtung := links
    END;(*CONDITION*)
END;(*Algorithmus*)
```

Algorithmus 3.4: Berechnung der relativen Bewegungsrichtung

3.5 liefert Algorithmus 3.2 die Roboterorientierung (Definition 2.1) zu Beginn der Drehbewegung $RAlpha_{Start}$ und am Ende des betrachteten Zeitabschnittes $RAlpha_{Ende}$. Für den in der Regel zu erwartenden Fall, daß in dieser Zeitspanne noch mindestens eine weitere Sensormessung durch das Sensorkontroll- und Signalverarbeitungssystem durchgeführt werden konnte, wird zudem die Roboterorientierung $RAlpha_i$ zum Zeitpunkt $T_{Start+1}$ durch Algorithmus 3.5 verarbeitet. Abhängig davon, ob $RAlpha_i$ bekannt ist oder nicht, sind zur Berechnung der Drehrichtung zwei grundsätzliche Vorgehensweisen zu unterscheiden.

Bei Angabe von $RAlpha_i$ kann die Rotationsrichtung über einen Vergleich der drei Winkelwerte bestimmt werden, da von diesen Werten die zeitliche Abfolge ihrer Ermittlung bekannt ist. Gilt beispielsweise die in Abbildung 3.6 graphisch dargestellte Beziehung $RAlpha_i < RAlpha_{Ende} < RAlpha_{Start}$, so kann diese Relation nur durch eine Rotation in Richtung `links` hervorgerufen worden sein, da der für $RAlpha_i$ bestimmte Wert zeitlich vor dem Wert von $RAlpha_{Ende}$ und nach dem Wert von

$RAlpha_{Start}$ berechnet wurde. Durch scharfes Hinsehen folgt aus Abbildung 3.6 unmittelbar, daß der Wert von *DiffGrad* für dieses Beispiel gerade $360 - (RAlpha_{Start} - RAlpha_{Ende})$ ist. Da drei Werte gegeben sind, müssen von einem Verfahren zur Ermittlung der Drehrichtung in analoger Weise noch fünf weitere Fälle berücksichtigt werden. Zusammen mit dem betrachteten Beispiel werden sie durch die Schritte a), b), ..., f) im `THEN`-Teil der ersten `IF`-Anweisung von Algorithmus 3.5 wiedergegeben.



$RAlpha_i < RAlpha_{Ende} < RAlpha_{Start}$

=> RRichtung := links
DiffGrad := (360 - ($RAlpha_{Start}$ - $RAlpha_{Ende}$)

Abbildung 3.6: Beispiel für eine Linksrotation

Liegt $RAlpha_i$ zur Berechnung der Rotationsrichtung nicht vor — etwa weil der Roboter sich nur ein kleines Stück weit gedreht hat, oder aber weil er sich enorm schnell (es sei darn erinnert, daß bis zu 5 Sensormessungen pro Sekunde verarbeitet werden können!) drehte, — so ist die tatsächlich eingeschlagene Drehrichtung nur unter Zuhilfenahme von $RAlpha_{Start}$ und $RAlpha_{Ende}$ prinzipiell nicht bestimmbar. Für diesen Fall erfolgt die Berechnung der Rotationsrichtung nach Algorithmus 3.5 gemäß der Strategie, daß stets die Richtung mit kleinstem Wert für *DiffGrad* als Drehrichtung angenommen wird. Da in der Praxis enorm schnelle Drehbewegungen wohl eher ausgeschlossen sein werden, ist zu erwarten, daß bei dieser Vorgehensweise in der Regel auch die tatsächliche Rotationsrichtung ermittelt wird.

Zur Umsetzung dieser Idee wird im zweiten Teil von Algorithmus 3.5 zunächst überprüft, ob der Wert von $RAlpha_{Start}$ kleiner oder größer als der Wert von $RAlpha_{Ende}$ ist. Es sei zunächst der Fall $RAlpha_{Start} < RAlpha_{Ende}$ angenommen. Ist die Differenz *Diff* der beiden Winkelwerte kleiner als 180°, so deutet dies im Sinne der obigen Strategie auf eine `Links`-Rotation hin, da der Wert von $RAlpha_{Start}$ zeitlich früher als der Wert von $RAlpha_{Ende}$ berechnet wurde. Der Wert von *DiffGrad* entspricht für diesen Fall gerade der ermittelten Differenz. Ist die Differenz hingegen größer als

Eingabe: $RAlpha_{Start}, RAlpha_{Ende}$ und ggf. $RAlpha_i$
Ausgabe: $DRichtung$ und $DiffGrad$

- Der Algorithmus berechnet die Richtung $DRichtung$ in die sich der Roboter in dem im Algorithmus 3.1 ermittelten Zeitraum gedreht hat. Eingabe ist die Roboterorientierung am Anfang $RAlpha_{Start}$ und am Ende der Drehbewegung $RAlpha_{Ende}$ sowie ggf. auch die Roboterorientierung $RAlpha_i$ an einem Zwischenzeitpunkt. Neben der $DRichtung$ wird zudem die bei der Drehung zurückgelegte Distanz in Grad $DiffGrad$ bestimmt.

- Ist für eine Drehung keine Roboterorientierung zwischen $RAlpha_{Start}$ und $RAlpha_{Ende}$ gegeben, so ist $DRichtung$ nicht bestimmbar. Der Algorithmus folgt in diesem Fall der Strategie, stets die Drehrichtung mit dem kleineren Wert für $DiffGrad$ anzunehmen.

```
BEGIN

• IF 'RAlpha_i ist bekannt.'
  THEN
      CONDITION
      (a) (RAlpha_i < RAlpha_Ende < RAlpha_Start)
          THEN DRichtung := links; DiffGrad := 360 − (RAlpha_Start − RAlpha_Ende)
      (b) (RAlpha_Start < RAlpha_i < RAlpha_Ende)
          THEN DRichtung := links; DiffGrad := RAlpha_Ende − RAlpha_Start
      (c) (RAlpha_Start < RAlpha_Ende < RAlpha_i)
          THEN DRichtung := rechts; DiffGrad := 360 − (RAlpha_Ende − RAlpha_Start)
      (d) (RAlpha_i < RAlpha_Start < RAlpha_Ende)
          THEN DRichtung := rechts; DiffGrad := 360 − (RAlpha_Ende − RAlpha_Start)
      (e) (RAlpha_Ende < RAlpha_Start < RAlpha_i)
          THEN DRichtung := links; DiffGrad := 360 − (RAlpha_Start − RAlpha_Ende)
      (f) (RAlpha_Ende < RAlpha_i < RAlpha_Start)
          THEN DRichtung := rechts; DiffGrad := RAlpha_Start − RAlpha_Ende
      END;(*CONDITION*)
  END;(*IF*)
• IF 'RAlpha_i ist nicht bekannt.'
  THEN
      CONDITION
      (a) (RAlpha_Start < RAlpha_Ende)
          THEN • Diff := RAlpha_Ende − RAlpha_Start
               • IF Diff < 180
                 THEN DRichtung := links; DiffGrad := Diff
                 ELSE DRichtung := rechts; DiffGrad := 360-Diff
      (b) (RAlpha_Start > RAlpha_Ende)
          THEN • Diff := RAlpha_Start − RAlpha_Ende
               • IF Diff < 180
                 THEN DRichtung := rechts; DiffGrad := Diff
                 ELSE DRichtung := links; DiffGrad := 360-Diff
  END;(*IF*)
END;(*Algorithmus*)
```

Algorithmus 3.5: Berechnung der Drehrichtung

180°, so liegt nach Algorithmus 3.5 eine Rechtsrotation vor, und der Wert von $DiffGrad$ ist $360 - Diff$. Für den Fall $RAlpha_{Start} > RAlpha_{Ende}$ werden bei gleicher Vorgehensweise entgegengesetzte Richtungsangaben festgelegt.

3.3 Wahrnehmungsintegrierende Handlungsmerkmale

Im Abschnitt 3.2 wird die Repräsentation und die Berechnung von Basis-Handlungsmerkmalen erörtert, während im Abschnitt 2.2 handlungsorientierte Wahrnehmungsmerkmale vorgestellt werden. Im vorliegenden Abschnitt wird nun gezeigt, wie sich Basis-Handlungsmerkmale und handlungsorientierte Wahrnehmungsmerkmale auf dem Wege zu einer Begriffsoperationalisierung zusammenführen lassen.

Zu diesem Zweck werden im Abschnitt 3.3.1 wahrnehmungsintegrierende Handlungsmerkmale entwickelt, über die im Sinne von Punkt 3) des Abschnittes 3.1 eine Verbindung von Handlungs- und Wahrnehmungsmerkmalen hergestellt wird. Zum Lernen der im Abschnitt 3.3.1 vorgeschlagenen Repräsentation wahrnehmungsintegrierender Handlungsmerkmale ist jedoch in einem beschränkten Umfang Hintergrundwissen erforderlich. Dieses Hintergrundwissen wird im Abschnitt 3.3.2 erläutert.

3.3.1 Repräsentation wahrnehmungsintegrierender Handlungsmerkmale

Wahrnehmungsintegrierende Handlungsmerkmale beinhalten sowohl Basis-Handlungen, die in der im Abschnitt 3.1 erörterten Arbeitsphase des Roboters als an den Roboter abzusetzende Handlungsanweisungen dienen, als auch Wahrnehmungsmerkmale, die über die im Abschnitt 2.2 vorgestellte Begriffshierarchie aus den Sensormeßwerten während der Durchführung bestimmter Handlungen vorwärtsinferiert werden können. Die damit verfolgte Idee ist, Handlungen des Roboters gemäß Punkt 2) des Abschnittes 3.1 unter der Kontrolle von während der Handlungsausführung ermittelten Wahrnehmungsmerkmalen durchführen zu können. Kapitel 6 vorwegnehmend, wird ein wahrnehmungsintegrierendes Handlungsmerkmal in diesem Sinne abgeleitet, wenn bei Ausführung seines Handlungsanteiles durch den Roboter die zugehörige Wahrnehmung über eine Vorwärtsinferenz der bei der Ausführung aufgenommenen Sensormeßwerte abgeleitet werden kann.

Die in diesen Überlegungen ausgedrückte Vorstellung von dem, was wahrnehmungsintegrierende Handlungsmerkmale ausmacht, liegt schon im Bereich dessen, was nach Abschnitt 3.1 zu einer Begriffsoperationalisierung gezählt werden kann, mit dem Unterschied, daß eine Handlungsbestimmung in Abhängigkeit von Wahrnehmungen — etwa „wenn Du vor der Tür stehst, dann gehe auch durch" — in diesen Überlegungen noch nicht berücksichtigt wird. Im Abschnitt 3.4 wird jedoch gezeigt werden, daß wahrnehmungsintegrierende Handlungsmerkmale quasi bereits die „Bausteine" sind, aus denen dann in einfacher Weise operationale Begriffe zusammengesetzt werden können.

Nach diesen allgemeinen Betrachtungen, die damit zusammengefaßt werden können, daß wahrnehmungsintegrierende Handlungsmerkmale Handlungen in Abhängigkeit von Wahrnehmungen zu beschreiben helfen, werden im folgenden einige dieser Merkmale anhand von Beispielsituationen erörtert werden. Zur Veranschaulichung der folgenden Ausführungen diene Abbildung 3.7.

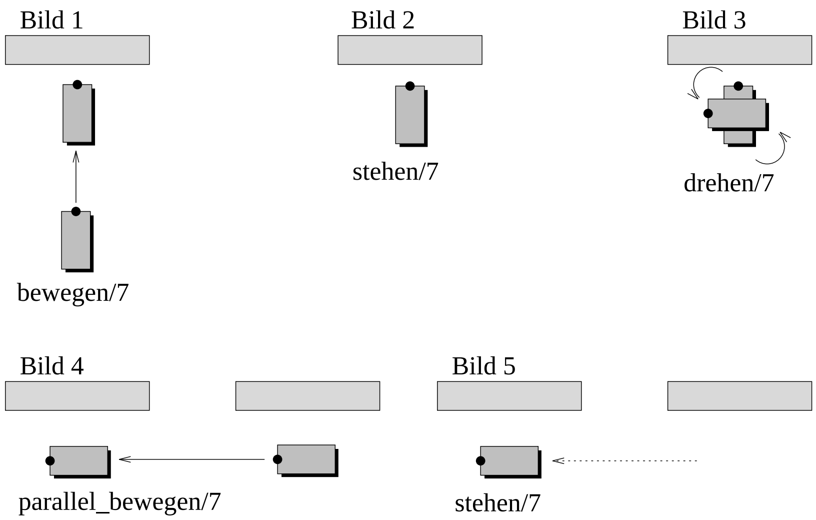


Abbildung 3.7: Beispiele wahrnehmungsintegrierender Handlungsmerkmale

Bild 1) in Abbildung 3.7 zeigt schematisch eine Situation, in der der Roboter sich zunächst ohne eine Wahrnehmung registrierend solange in eine Richtung vorwärtsbewegt, bis er schließlich auf eine Wand trifft. Diese Vorwärtsbewegung für einen bestimmten Zeitraum, dessen Ende dadurch bestimmt wird, daß eine Wand in Bewegungsrichtung wahrgenommen wird, kann durch das weiter unten stehende, näher charakterisierte 7-stellige Prädikat `bewegen/7` ausgedrückt werden. Das wahrnehmungsintegrierende Handlungsmerkmal `bewegen/7` bezieht sich dabei ausschließlich auf die Vorwärtsbewegung bis zum Zeitpunkt, an dem die Wand wahrgenommen wird, und schließt das sich anschließende Stehenbleiben vor der Wand nicht in seine Beschreibung mit ein. Zur Beschreibung dieser in Bild 2) dargestellten Situation dient das wahrnehmungsintegrierende Handlungsmerkmal `stehen/7`. Bild 3) in Abbildung 3.7 steht für eine Situation in der der Roboter sich solange dreht, bis er in einer bestimmten Richtung die Wahrnehmung `vor_einer_Wand` aus den Sensormeßwerten vorwärtsinferieren kann. Diese Situation kann durch das wahrnehmungsintegrierende Handlungsmerkmal `drehen/7` ausgedrückt werden. Das wahrnehmungsintegrierende Handlungsmerkmal `p_bewegen/7` beschreibt hingegen eine Vorwärtsbewegung, die parallel entlang eines während der Bewegung wahrgenommenen Objektes erfolgt. Bild 4) zeigt schematisch als ein Beispiel hierfür eine Vorwärtsbewegung für einen Zeitraum, in dem das handlungsorientierte Wahrnehmungsmerkmal `entlang_einer_Tür` wahrgenommen wird.

Im Bild 5) von Abbildung 3.7 ist, wie im Bild 2), eine Situation dargestellt, in der der Roboter bewegungslos vor einer Wand steht. Im Gegensatz zur Situation im Bild 2) wird hier jedoch die vorherige Wahrnehmung `entlang_einer_Tür` mit in die Beschreibung durch das wahrnehmungsintegrierende Handlungsmerkmal `stehen/7` aufgenommen. Das Merkmal `stehen/7` ist demnach mit einer Art Gedächtnis zur Speicherung der letzten Wahrnehmung ausgestattet, was im Hinblick auf die Ausführung von Aufträgen bzw. die Planung von Handlungsabläufen zur Auftragsdurchführung sinnvoll sein kann. So ist es beispielsweise zur Durchführung des Auftrages, sich durch eine Tür zu bewegen, sinnvoll, in Betracht zu ziehen, daß gerade die Wahrnehmung `entlang_einer_Tür` registriert wird.

Nach diesen beispielhaften Betrachtungen werden wahrnehmungsintegrierende Handlungsmerkmale als Prädikate der Form:

`stehen`$(Trc, T_1, T_2, W, WRichtung, WSeite, LW)$,
`drehen`$(Trc, T_1, T_2, RGK, DRichtung, W, WRichtung)$,
`bewegen`$(Trc, T_1, T_2, GK, RelRichtung, W, WRichtung)$,
`p_bewegen`$(Trc, T_1, T_2, GK, RelRichtung, W, WRichtung)$

---

näher charakterisiert. Hierbei stehen die Variablen *Trc*, T_1 und T_2 jeweils für den Bezeichner der Roboterfahrt, sowie für den Start- bzw. Endzeitpunkt des durch die Merkmale beschriebenen Zeitraumes. Die Variable *W* steht für die Namen möglicher handlungsorientierter Wahrnehmungsmerkmale, die gegen Ende oder während der durch die wahrnehmungsintegrierenden Handlungsmerkmale charakterisierten Handlungen im betrachteten Zeitraum registriert werden. Analog hierzu gibt *LW* im wahrnehmungsintegrierenden Handlungsmerkmal `stehen/7` den Namen der Wahrnehmung wieder, die unmittelbar vor dem Zeitpunkt T_1 registriert wird. Die Richtung, aus der die aktuelle Wahrnehmung empfangen wird, wird dabei jeweils durch *WRichtung* angezeigt. *RGK*, *GK*, *DRichtung* und *RelRichtung* sind die schon aus der Beschreibung der Basis-Handlungsmerkmale im Abschnitt 3.2.1 bekannten Bezeichner zur Angabe der Rotations- und Bewegungsgeschwindigkeitsklasse, sowie der Rotations- und relativen Bewegungsrichtung. Die Variable *WSeite* gibt zusätzlich zur Information über die Wahrnehmungsrichtung *WRichtung* an, ob die wahrnehmungsregistrierenden Sensoren an einer schmalen oder breiten Seite des Roboters montiert sind. Mögliche Werte sind daher `schmale_Seite` und `breite_Seite`. Diese Information kann für die Auswahl der als nächstes durchzuführenden Handlung in einer Situation, in der der Roboter beispielsweise vor einer Tür steht, von Bedeutung sein. Steht er etwa mit einer schmalen Seite vor der Tür, so kann als nächste Aktion die Tür durchfahren werden, während bei Wahrnehmung der Tür von Sensoren einer breiten Seite vor dem Durchfahren der Tür zunächst eine Rotation notwendig sein wird.

Regeln zur Beschreibung der wahrnehmungsintegrierenden Handlungsmerkmale setzen sich, wie bereits eingangs des Abschnittes erörtert, aus handlungsorientierten Wahrnehmungsmerkmalen und Basis-Handlungsmerkmalen zusammen. Der im Abschnitt 2.2.3 getroffenen Unterscheidung zwischen zeitpunkt- und zeitintervallorientierten Wahrnehmungsmerkmalen kommt auf der Ebene der wahrnehmungsintegrierenden Handlungsmerkmale ebenfalls eine Bedeutung zu. So unterscheiden sich die in Abbildung 3.7 skizzierten Handlungsmerkmale gerade auch dadurch, daß in einem Fall Handlungen bis zur Erfassung eines zeitpunktorientierten Wahrnehmungsmerkmals und im anderen Fall Handlungen während der Registrierung einer zeitintervallorientierten Wahrnehmung beschrieben werden. Um diese Unterscheidung auch in der Modellierung stärker hervorzuheben, werden zur Beschreibung der wahrnehmungsintegrierenden Handlungsmerkmale die im folgenden Abschnitt beschriebenen Prädikate `zp_Wahrnehmung/5` und `zi_Wahrnehmung/7` eingesetzt, die eine direkte Verbindung zu den im Abschnitt 2.2 erläuterten handlungsorientierten Wahrnehmungsmerkmalen herstellen. Mögliche Regeln zur Beschreibung der erörterten wahrnehmungsintegrierenden Handlungsmerkmale sind dann beispielsweise:

$\texttt{zp\_Wahrnehmung}(Trc, T_1, W1, WRichtung, WSeite)$ &
$\texttt{bleibe\_stehen}(Trc, T_1, T_2)$
$\longrightarrow \texttt{stehen}(Trc, T_1, T_2, W1, WRichtung, WSeite, W2)$

$\texttt{zi\_Wahrnehmung}(Trc, T_0, T_1, W, WRichtung, WSeite, O)$ &
$\texttt{zp\_Wahrnehmung}(Trc, T_1, W1, WRichtung1, WSeite1)$ &
$\texttt{bleibe\_stehen}(Trc, T_1, T_2)$
$\longrightarrow \texttt{stehen}(Trc, T_1, T_2, W1, WRichtung1, WSeite1, W)$

$\texttt{bewege\_dich}(Trc, T_1, T_2, GK, RelRichtung)$ &
$\texttt{zp\_Wahrnehmung}(Trc, T_2, W, WRichtung, WSeite)$ &
$\longrightarrow \texttt{bewegen}(Trc, T_1, T_2, GK, RelRichtung, W, WRichtung)$

$\texttt{drehe\_dich}(Trc, T_1, T_2, RGK, DRichtung)$ &
$\texttt{zp\_Wahrnehmung}(Trc, T_2, W, WRichtung, WSeite)$ &
$\longrightarrow \texttt{drehen}(Trc, T_1, T_2, RGK, DRichtung, W, WRichtung)$

$\texttt{bewege\_dich}(Trc, T_1, T_2, GK, RelRichtung)$ &
$\texttt{zi\_Wahrnehmung}(Trc, T_1, T_2, W, WRichtung, WSeite, parallel)$ &
$\longrightarrow \texttt{p\_bewegen}(Trc, T_1, T_2, GK, RelRichtung, W, WRichtung)$

Diese Regeln zur Beschreibung der vorgestellten wahrnehmungsintegrierenden Handlungsmerkmale stellen typische Beispiele solcher Regeln dar und geben im wesentlichen noch einmal die Erläuterungen zur Abbildung 3.7 formal wieder. Inwieweit diese Regeln in der Lernphase des Roboters erlernt werden können, wird im Kapitel 5 diskutiert werden.

3.3.2 Verwendetes Hintergrundwissen

Zum Lernen der im vorangegangenen Abschnitt beschriebenen wahrnehmungsintegrierenden Handlungsmerkmale ist es aufgrund der Einführung von `zp_Wahrnehmung/5` und `zi_Wahrnehmung/7` notwendig, eine Beschreibung dieser Prädikate als Hintergrundwissen, etwa in der Form von Regeln, zur Verfügung zu stellen. Beide Prädikate dienen lediglich als Verbindungsglied zwischen den im Abschnitt 2.2 beschriebenen handlungsorientierten Wahrnehmungsmerkmalen und den wahrnehmungsintegrierenden Handlungsmerkmalen, so daß sie über das Argument W direkt in handlungsorientierte Wahrnehmungsmerkmale übersetzt werden können. Zu diesem Zweck können einfache Regeln mit `zp_Wahrnehmung/5` und `zi_Wahrnehmung/7` als Konklusionen und handlungsorientierten Wahrnehmungsmerkmalen als Prämissen aufgestellt werden, wobei abhängig von der Prämisse einer Regel in ihrer Konklusion für das Argument W eine auf den Namen des handlungsorientierten Wahrnehmungsmerkmals referierende Konstante eingetragen wird. Einige Beispiele solcher Regeln sind:

`vor_einer_Tür`$(Trc, T_1, WRichtung, WSeite)$
$\longrightarrow$ `zp_Wahrnehmung`$(Trc, T_1,$`vor_einer_Tür`$, WRichtung, WSeite)$

`vor_einer_Wand`$(Trc, T_1, WRichtung, WSeite)$
$\longrightarrow$ `zp_Wahrnehmung`$(Trc, T_1,$`vor_einer_Wand`$, WRichtung, WSeite)$

`durch_eine_Tür`$(Trc, T_1, T_2, WRichtung, WSeite, O)$
$\longrightarrow$ `zi_Wahrnehmung`$(Trc, T_1, T_2,$`durch_eine_Tür`$, WRichtung, WSeite, O)$

Bereits im vorherigen Abschnitt wird angedeutet, daß die Verwendung von `zp_Wahrnehmung/5` und `zi_Wahrnehmung/7` keineswegs notwendig ist, sondern lediglich der Unterstützung einer klaren Modellierung dient. So könnten prinzipiell anstelle dieser Prädikate direkt die handlungsorientierten Wahrnehmungsmerkmale in die Prämissen der die wahrnehmungsintegrierenden Handlungsmerkmale beschreibenden Regeln eingetragen werden. Das dies nicht so gemacht wird, bietet neben dem erwähnten Modellierungsaspekt zudem die Möglichkeit, die Verbindung zwischen Wahrnehmungsmerkmalen und wahrnehmungsintegrierenden Handlungsmerkmalen einheitlich zu beschreiben. Für die im Kapitel 5 beschriebenen Versuche zum Lernen wahrnehmungsintegrierender Handlungsmerkmale bedeutet dies insbesondere, daß weniger Regelschemata verwendet werden müssen.

3.4 Operationale Begriffe

Während Abschnitt 3.3 die Repräsentation wahrnehmungsintegrierender Handlungsmerkmale erörtert, wird im vorliegenden Abschnitt gezeigt, wie sich mit ihrer Hilfe operationale Begriffe beschreiben lassen, die den im Abschnitt 3.1 aufgestellten Anforderungen an die Repräsentation genügen. Im

Abschnitt 3.4.1 wird hierzu eine mögliche Repräsentation operationaler Begriffe vorgestellt. Einige der in diesem Abschnitt vorgeschlagenen operationalen Begriffe benötigen jedoch Hintergrundwissen zur Umrechnung von Richtungsangaben. Im Abschnitt 3.4.2 wird zum einen gezeigt werden, wie das erforderliche Hintergrundwissen repräsentiert werden kann, und zum anderen erläutert, welche Bedeutung ihm bei der Beschreibung operationaler Begriffe zukommt.

3.4.1 Repräsentation operationaler Begriffe

Nach Abschnitt 3.1 liegt im Sinne dieser Arbeit eine Begriffsoperationalisierung vor, wenn Handlungen des Roboters mit während ihrer Ausführung aus den Sensordaten ermittelten Wahrnehmungen in einer Weise in Verbindung gesetzt werden, die es ermöglicht, Handlungen in Abhängigkeit von Wahrnehmungen zu bestimmen, auszuführen und während ihrer Durchführung zu kontrollieren. Diese Forderung wird zum Teil schon von der Repräsentation wahrnehmungsintegrierender Handlungsmerkmale erfüllt. Was dieser Repräsentation jedoch noch fehlt, ist eine Struktur, auf der aufbauend eine Handlungsbestimmung unter der Berücksichtigung von Wahrnehmungen erfolgen kann, so daß die Möglichkeit besteht, sie zu einer Handlungsplanung zur Erreichung vorgegebener Ziele einzusetzen.

So muß nach Punkt 4) im Abschnitt 3.1 für jeden der zu bildenden operationalen Begriffe zum einen klar sein, in welcher Situation er eingesetzt werden kann, und zum anderen zu welcher konkreten Situation ein erfolgreicher Einsatz führt. Im Sinne von Punkt 5) desselben Abschnittes sollte die zu findende Struktur im Hinblick auf die Realisierung einer Handlungsplanung gewährleisten, daß operationale Begriffe über Situationen, die ihren Einsatz erlauben, und solche zu der ihr erfolgreicher Einsatz führt, verbunden werden können. Über diese Verbindung sollte es dann möglich sein, ausgehend von einem Zielbegriff rückwärts verkettend eine Folge operationaler Begriffe zu bestimmen, deren Ausführung aus der konkret vorhandenen Situation heraus zum Ziel führen kann. Dem Ziel entspricht in dieser Vorstellung die Situation, in die der erfolgreiche Einsatz des letzten operationalen Begriffes der Folge mündet.

Werden die im Abschnitt 3.3 vorgestellten wahrnehmungsintegrierenden Handlungsmerkmale vor dem Hintergrund dieser Betrachtungen noch einmal näher beleuchtet, so lassen sich die durch sie repräsentierten Situationen einer Roboterfahrt durch zwei voneinander zu unterscheidende Gruppen charakterisieren. Auf der einen Seite werden Situationen beschrieben, in denen die Handlung des Roboters darin besteht, nichts zu tun, d.h. bewegungslos auf einer Stelle zu verharren, während über seine Sensoren eine bestimmte Wahrnehmung registriert wird, und auf der anderen Seite werden durch den Roboter bestimmte Aktionen in Abhängigkeit von Wahrnehmungen durchgeführt, die seine Lage relativ zur Umgebung verändern. Die letztere Gruppe wahrnehmungsintegrierender Handlungsmerkmale beschreibt demnach Situationen, die den Roboter der Erfüllung seines Auftrages ein Stück weit näherbringen, während es sich im erstgenannten Fall eher um Momentaufnahmen der Roboterfahrt handelt, die für sich alleine betrachtet Zeitspannen charakterisieren, in denen der Roboter seinem eigentlichen Ziel nicht näher kommt.

Die Idee zur Repräsentation operationaler Begriffe ist, wahrnehmungsintegrierende Handlungsmerkmale beider Gruppen in einer Weise zu Begriffen zusammenzufassen, die Situationen vor der Anwendung der Begriffe, während und nach ihrer Anwendung beschreiben. Nach den vorausgegangenen Überlegungen liegt es dabei nahe, dies in einer Form zu tun, in der sich an ein im obigen Sinne eine Momentaufnahme charakterisierendes wahrnehmungsintegrierendes Handlungsmerkmal, ein Handlungsmerkmal anschließt, das den Roboter seinem Ziel näherbringt, was wiederum durch ein eine Momentaufnahme beschreibendes wahrnehmungsintegrierendes Handlungsmerkmal festgehalten wird. Nach dieser Idee besitzen operationale Begriffe hinsichtlich ihres Aufbaus eine Struktur der Form:

`<stehen/7>` &
`<`*WIHM*`/7>` &
`<stehen/7>`
⟶ `<operationaler Begriff>`,

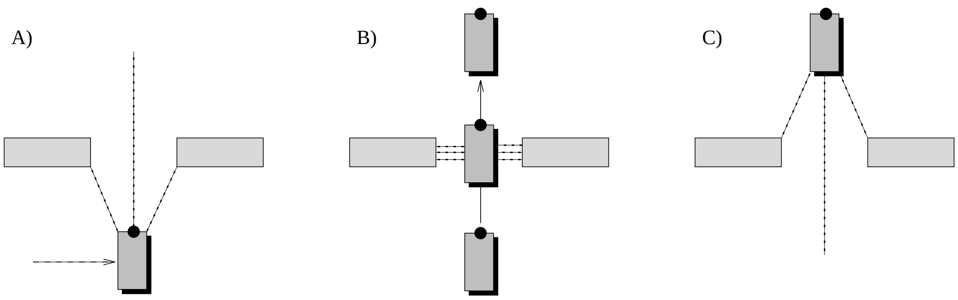
wobei *WIHM* für `bewegen/7`, `drehen/7`, `p_bewegen/7` oder `d_bewegen/7` („d_“ steht für diagonal) stehen kann, und ggf. noch Hintergrundwissen zur Festlegung von Richtungsangaben in die Beschreibung einzelner operationaler Begriffe aufgenommen werden muß.

Zur Anwendung eines operationalen Begriffes muß demnach der Roboter zunächst einmal für eine gewisse Zeitspanne bewegungslos an einer Position stehen bleiben, wobei die zur Anwendung des Begriffes erforderliche Wahrnehmung zu registrieren ist. Die dann folgende unter der Kontrolle einer weiteren Wahrnehmung stehende und durch *WIHM* beschriebene Aktion wird dann durch ein erneutes bewegungsloses Verharren bei gleichzeitiger Erfassung einer dritten Wahrnehmung abgelöst. Bei einer Verarbeitung von maximal 5 Sensormessungen aller Sensoren pro Sekunde durch das Sensorkontroll- und Signalverarbeitungssystem stellt das zweimalige Stehenbleiben bei der Anwendung eines operationalen Begriffes keine praxisrelevante Einschränkung für die Durchführung von Roboterfahrten in der Arbeitsphase dar.

Es bietet im Gegenteil den Vorteil, daß die eingangs geforderte Verknüpfung operationaler Begriffe zur Handlungsplanung jeweils über die in ihrer Beschreibung enthaltenen wahrnehmungsintegrierenden Handlungsmerkmale vom Typ `stehen/7` erfolgen kann. Unter diesem Blickwinkel erinnert die Struktur operationaler Begriffe entfernt an den Aufbau klassischer Planungsoperatoren nach dem STRIPS-Schema [Fikes und Nilsson, 1971]. Für operationale Begriffe kann dieses Schema durch *<<Anwendbarkeit>, <Handlung mit Wahrnehmung>, <Verifikation>>* charakterisiert werden. Im Kapitel

6 wird der Zusammenhang zwischen STRIPS-Operatoren und operationalen Begriffen nocheinmal aufgegriffen werden.

Nach diesen allgemeinen Betrachtungen wird im folgenden der operationale Begriff `bewege_dich_durch_die_Tür/4` beispielhaft näher betrachtet werden. Wie durch die Namensgebung bereits angedeutet wird, beschreibt er eine Fahrt des Roboters durch eine geöffnete Tür. Um eine Tür durchfahren zu können, muß sich der Roboter zunächst einmal in eine Position direkt vor eine Tür begeben. Ein Basis-Handlungsmerkmal `bleibe_stehen/3` in Verbindung mit einem handlungsorientierten Wahrnehmungsmerkmal `vor_einer_Tür/4` bildet demnach die Voraussetzung für die Anwendbarkeit von `bewege_dich_durch_die_Tür/4`. Diese zur Anwendung notwendige Situation wird durch ein entsprechendes wahrnehmungsintegrierendes Handlungsmerkmal `stehen/7` repräsentiert. Bild A) in Abbildung 3.8 veranschaulicht diese Situation an einem konkreten Beispiel.



zu A) : stehen(t88,72,75,vor_einer_Tür,links,schmale_Seite,vor_einer_Tür)
zu B) : p_bewegen(t88,75,95,langsame_Bewegung,links,durch_eine_Tür,rechts_und_links)
zu C) : stehen(t88,95,98,vor_einer_Tür,hinten,schmale_Seite,durch_eine_Tür)

Abbildung 3.8: Beispiel `bewege_dich_durch_die_Tür/4`

Während der sich anschließenden Fahrt durch die Tür (Bild B) in Abbildung 3.8) wird die Ausführung des Basis-Handlungsmerkmals `bewege_dich/5` durch das handlungsorientierte Wahrnehmungsmerkmal `durch_eine_Tür/6` kontrolliert, d.h., daß zur Durchführung der gewünschten Handlung ein entsprechendes wahrnehmungsintegrierendes Handlungsmerkmal `p_bewegen/7` bewiesen werden muß. Die Tür ist schließlich erfolgreich durchfahren worden, wenn wiederum das Wahrnehmungsmerkmal `vor_einer_Tür/4` registriert wird. Aufgrund des im Abschnitt 3.1 beschriebenen Modells „*vorne ist immer Bewegungsrichtung*" wird `vor_einer_Tür/4` bei erfolgreichen Anwendungen stets durch die Sensorklasse `hinten` wahrgenommen werden, so daß sie als Konstante in ein entsprechendes, die Verifikationssituation beschreibendes wahrnehmungsintegrierendes Handlungsmerkmal `stehen/7` eingetragen werden kann. Bild C) in Abbildung 3.8 illustriert diese Situation am Beispiel. Damit ergibt sich für die Beschreibung des operationalen Begriffes `bewege_dich_durch_die_Tür/4` die Regel:

`stehen`($Trc, T_1, T_2,$ `vor_einer_Tür`, $Richtung, schmale\_Seite, LW$) &
`p_bewegen`($Trc, T_2, T_3, GK, Richtung,$ `durch_eine_Tür`, $rechts\_und\_links$)&
`stehen`($Trc, T_3, T_4,$ `vor_einer_Tür`, $hinten, schmale\_Seite,$ `durch_eine_Tür`)
$\longrightarrow$ **bewege_dich_durch_die_Tür**(Trc, T_1, T_4, T_3),

---

Um eine Verkettung operationaler Begriffe auf einer Planungsebene über die Zeitpunkte erzielen zu können, wird der Zeitpunkt T_3 neben der Kennzeichnung der Roboterfahrt Trc sowie dem Start- bzw. Endzeitpunkt des operationalen Begriffes T_1 bzw. T_4 als zusätzliches Argument in den Begriff aufgenommen. Mit der verfolgten Idee, die operationalen Begriffe auf der Ebene der wahrnehmungsintegrierenden Handlungsmerkmale über die jeweiligen Verifikations- und Anwendbarkeitssituationen zu verbinden, entspricht T_3 gerade dem Startzeitpunkt des als nächstes anwendbaren Begriffes.

Der operationale Begriff `bewege_dich_durch_die_Tür/4` gehört zu den wenigen Begriffen, die gänzlich ohne zusätzliche Informationen über die Umrechnung von Richtungsangaben auskommen. Für die meisten anderen Begriffe müssen solche Informationen in irgendeiner Weise bereitgestellt werden, damit sie als wichtiger Bestandteil ihrer Beschreibung beim Lernen der Begriffe Berücksichtigung finden können. Im Abschnitt 3.4.2 werden die notwendigen Umrechnungsinformationen, die zum Lernen der Begriffe als Fakten repräsentiert, dem Lernverfahren in Form von Hintergrundwissen zur Verfügung gestellt werden, im einzelnen erörtert.

Im folgenden wird noch ein weiterer Begriff beispielhaft näher betrachtet werden, der zu der Gruppe operationaler Begriffe gehört, deren zu lernende Beschreibung Hintergrundwissen enthält. An dieser Stelle sei ausdrücklich daran erinnert, daß die in diesem Abschnitt angegebenen Beschreibungen operationaler Begriffe dem Roboter nicht vorgegeben werden sollen, sondern daß es das Ziel ist, sie in der Lernphase durch kontrollierte (z.B. *joy stick* geführte) Roboterfahrten automatisch zu erwerben. Insofern sind sie lediglich als mögliche Beschreibungen aufzufassen, die in der Lernphase genauso, oder aber nur in ähnlicher Form gelernt werden können.

Zur Illustration des nächsten Beispiels diene Abbildung 3.9. Bild A) dieser Abbildung zeigt eine Situation in der der Roboter vor einer Wand steht und zuvor bei einer Vorwärtsbewegung die Wahrnehmung `entlang_einer_Tür/6` registriert hat. Wie im voherigen Beispiel könnte sein Auftrag nun lauten, durch die Tür zu fahren, also den operationalen Begriff `bewege_dich_durch_die_Tür/4` auszuführen.

Noch einmal auf Punkt 6) des Abschnittes 3.1 zurückblickend, wird an diesem Beispiel deutlich, daß operationale Begriffe als Sprachelemente einer Kommandosprache höherer Ebene fungieren können. So könnte beispielsweise der Auftrag an den Roboter durch das Kommando „*bewege_dich_durch_die_Tür*“ ausgedrückt werden. Unter der Annahme eines

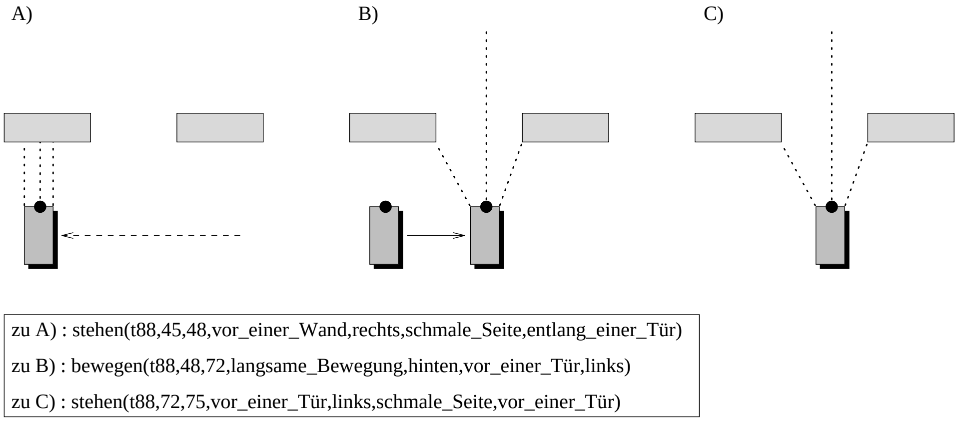


Abbildung 3.9: Beispiel `bewege_dich_vor_die_Tür/4`

vorhandenen Planungssystems[1] für operationale Begriffe könnte dieses System für den gegebenen Auftrag in der skizzierten Situation entscheiden, solange zurückzufahren bis auf der linken Seite die Wahrnehmung `vor_einer_Tür/4` registriert wird und anschließend den operationalen Begriff `bewege_dich_durch_die_Tür/4` auszuführen, um auf diese Weise die Zielsituation zu erreichen. Das Zurücksetzen vor die Tür wird dabei durch den operationalen Begriff `bewege_dich_vor_die_Tür/4` charakterisiert. Eine mögliche Regel zu seiner Beschreibung lautet nach dem vorgetragenen Beispiel etwa:

$\texttt{stehen}(Trc, T_1, T_2, \texttt{vor\_einer\_Wand}, WRichtung, WSeite,$
$\quad \texttt{entlang\_einer\_Tür}) \;\&$
$\texttt{hw\_gegenüberliegende\_Richtung}(WRichtung, WRichtung1) \;\&$
$\texttt{bewegen}(Trc, T_2, T_3, GK, hinten, \texttt{vor\_einer\_Tür}, WRichtung1) \;\&$
$\texttt{stehen}(Trc, T_3, T_4, \texttt{vor\_einer\_Tür}, WRichtung1, WSeite, \texttt{vor\_einer\_Tür})$
$\longrightarrow$ **bewege_dich_vor_die_Tür**(Trc, T_1, T_4, T_3),

In der durch das erste wahrnehmungsintegrierende Handlungsmerkmal `stehen/7` ausgedrückten Anwendbarkeitssituation des Begriffes ist als letzte Wahrnehmung der Name des handlungsorientierten Wahrnehmungsmerkmals `entlang_einer_Tür/6` als Konstante eingetragen, so daß der Begriff auch wirklich nur für die im Beispiel skizzierte Situation und Varianten anwendbar wird. Das Zurücksetzen vor die Tür wird durch das wahrnehmungsintegrierende Handlungsmerkmal `bewegen/7` ausgedrückt, wobei die relative Bewegungsrichtungsangabe aufgrund des im Abschnitt 3.1 geschilderten Modells zur Festlegung von Richtungsangaben für alle denkbaren Beispielsitua-

[1] Ein Planungssystem für operationale Begriffe ist im Rahmen des Projektes BLearn II bisher noch nicht entwickelt worden. Arbeiten hierzu sind Gegenstand aktueller Forschungen, so daß diesem Umstand in naher Zukunft Rechnung getragen wird.

tionen stets *hinten* ist. Für das entsprechende Argument in `bewegen/7` kann demnach ebenfalls eine Konstante eingetragen werden.

In dem in Abbildung 3.9 Bild B) dargestellten Beispiel wird die Tür bei der Vorwärtsbewegung in Richtung *hinten* auf der linken Seite wahrgenommen werden. Natürlich könnte dies auch die rechte Seite sein, wenn sich der Roboter beispielsweise zuvor im dargestellten Raum von links nach rechts an der Tür vorbeibewegt hat. In diesem Fall wäre im entsprechenden handlungsorientierten Wahrnehmungsmerkmal `entlang_einer_Tür/6` die Wahrnehmungsrichtung *links* eingetragen, die sich dann auch im wahrnehmungsintegrierenden Handlungsmerkmal `stehen/7` der Anwendbarkeitsbedingung des dargestellten Begriffes als Wahrnehmungsrichtungsangabe *WRichtung* für `vor_einer_Wand/4` wiederfinden würde. Die zu erwartende Wahrnehmungsrichtung *WRichtung1* in `bewegen/7` hängt also unmittelbar von der Wahrnehmungsrichtung *WRichtung* in `stehen/7` ab. Abhängig vom konkreten Wert für *WRichtung* bindet das Hintergrundwissen `hw_gegenüberliegende_Richtung/2` den Wert der Variablen *WRichtung1* stets an den richtigen Wert. Also beispielsweise für die in Abbildung 3.9 Teil B) dargestellte Situation an den Wert *links*.

Neben den bisher im Detail vorgestellten operationalen Begriffen `bewege_dich_durch_die_Tür/4` und `bewege_dich_vor_die_Tür/4` sind noch eine Reihe weiterer Begriffe denkbar, die in diesem Abschnitt nicht im Einzelnen erörtert werden können. Hierzu zählen beispielsweise:

`bewege_dich_näher_zur_Wand/4`,
`drehe_dich_vor_der_Wand/4`,
`bewege_dich_parallel_zur_Wand_in_die_Ecke/4`,
`drehe_dich_in_der_Ecke_zur_vorderen_Wand/4`,
`bewege_dich_entlang_der_Tür/4` und
`drehe_dich_vor_der_Tür/4`.

Die Bedeutung dieser Begriffe läßt sich im wesentlichen bereits aus ihren Namen erkennen. Zum Teil werden sie aber auch noch in den weiteren Abschnitten dieser Arbeit näher betrachtet werden. Im Anhang D werden mögliche Regeln zu ihrer Beschreibung aufgelistet. Inwieweit diese oder ähnliche Regeln in der Lernphase des Roboters gelernt werden können, wird Kapitel 5 zeigen. Im Kapitel 6 werden erste Ideen zur zukünftigen Anwendung operationaler Begriffe als Planungsoperatoren angedacht. Im folgenden Abschnitt wird jedoch zunächst das beim Lernen operationaler Begriffe zu berücksichtigende Hintergrundwissen erläutert.

3.4.2 Verwendetes Hintergrundwissen

Im vorangegangenen Abschnitt wurde bereits dargelegt, daß einige der zu lernenden operationalen Begriffe Hintergrundwissen zur Umrechnung von

Richtungsangaben in ihrer Beschreibung benötigen. Im vorliegenden Abschnitt wird nun erörtert, wie das erforderliche Hintergrundwissen repräsentiert wird, und warum es zur Beschreibung operationaler Begriffe überhaupt notwendig ist.

Zunächst auf den zweiten Aspekt bezugnehmend, ist zu allererst festzuhalten, daß Hintergrundwissen, wie es im folgenden erläutert wird, für die Phase des Lernens operationaler Begriffe keine entscheidende Rolle spielt. So könnte beispielsweise der im voherigen Abschnitt erläuterte Begriff `bewege_dich_vor_die_Tür/4` in der Lernphase des Roboters auch ohne Kenntnis der Relation `hw_gegenüberliegende_Richtung/2` durch maschinelle Lernverfahren gelernt werden. Im einfachsten Fall würde dann eine Begriffsbeschreibung erworben werden, die sich von der angegebenen nur dadurch unterscheidet, daß die Prämisse `hw_gegenüberliegende_Richtung/2` wegfällt. Doch wie ist es um die Anwendbarkeit dieses Begriffes in der Arbeitsphase bestellt?

Zur Beantwortung dieser Frage sei zur Verdeutlichung ein konkretes Beispiel herangezogen. Angenommen der Roboter befindet sich in einer Situation, die durch das wahrnehmungsintegrierende Handlungsmerkmal $\texttt{stehen}(t88, 45, T_2, \texttt{vor\_einer\_Wand}, rechts, schmale\_Seite, \texttt{entlang\_einer\_Tür})$ beschrieben wird, wobei die Variable T_2 erst im Moment einer Situationsänderung — etwa durch eine eingeleitete Vorwärtsbewegung oder eine Rotation — an einem bestimmten Zeitpunkt gebunden wird (zur Illustration diene abermals Abbildung 3.9). Sollte sein Auftrag nun lauten, sich durch die Tür zu bewegen, so könnte ein fiktives Planungssystem für operationale Begriffe einem fiktiven Modul zur Ausführungsüberwachung operationaler Begriffe die Anweisung erteilen, den operationalen Begriff `bewege_dich_vor_die_Tür/4` anzuwenden.

Sei zunächst der Fall angenommen, daß die Relation `hw_gegenüberliegende_Richtung/2` nicht in die Beschreibung des operationalen Begriffes aufgenommen wurde. In diesem Fall müßte das Modul zur Ausführungsüberwachung das wahrnehmungsintgrierende Handlungsmerkmal $\texttt{bewegen}(t88, T_2, GK, hinten, \texttt{vor\_einer\_Tür}, WRichtung1)$ als nächste auszuführende Anweisung betrachten. Mit Hilfe eines einfachen Meta-Interpreters für die in diesem Kapitel aufgebaute Begriffshierarchie könnte diese Anweisung auf das Basis-Handlungsmerkmal $\texttt{bewege\_dich}(t88, T_2, T_3, GK, hinten)$ und das handlungsorientierte Wahrnehmungsmerkmal $\texttt{vor\_einer\_Tür}(t88, T_3, WRichtung1, WSeite)$ zurückgeführt werden. Das Basis-Handlungsmerkmal `bewege_dich/5` wird als nächstes in ein entsprechendes Handlungsmerkmal `a_bewege_dich/5` übersetzt. Möglicherweise ist auch noch eine übergeordnete Strategie vorhanden, die für die vorliegende Situation die Geschwindigkeitsklasse GK auf den Wert `langsame_Bewegung` setzt, so daß `a_bewege_dich/5` in elementare Roboteroperationen übertragbar und damit ausführbar wird. Mit der Ausführung von `a_bewege_dich/5` wird die Variable T_2 an einen konkreten Wert gebunden. Mit dem Beginn der Vorwärtsbewegung muß nun für jeden Zeitpunkt der Fahrt der Versuch unternommen werden, das handlungsorientierte Wahrnehmungsmerkmal `vor_einer_Tür/4` vorwärts aus den Sensormeßwerten gemäß der im Abschnitt 2.2 vorgestellten Hierarchie zu inferieren, um auf diese Weise den Zeitpunkt T_3 zu binden. Da *WRichtung1* ungebunden ist, obgleich es in der beschriebenen Situation sehr wohl eindeutig ist, daß es sich hier um den Wert *links* handeln muß, könnte der Fall auftreten, daß zuvor eine andere Tür oder etwas Türähnliches aus einer falschen Richtung wahrgenommen wird. Für diesen Fall würde die Ausführung von `bewege_dich_vor_die_Tür/4` im Sinne der Auftragserfüllung scheitern.

Sei nun der Fall betrachtet, daß `hw_gegenüberliegende_Richtung/2` ein Teil der Beschreibung von `bewege_dich_vor_die_Tür/4` ist. In diesem Fall wird *WRichtung1* über den Wert *rechts* an den konkreten Wert *links* gebunden, so daß sich das geschilderte Problem nicht ergibt. Im Gegenteil bietet sich zudem der Vorteil, daß die Suche nach dem Wahrnehmungsmerkmal `vor_einer_Tür/4` auf eine einzelne Sensorklasse beschränkt werden kann.

Die Ausführungen am vorgetragenen Beispiel zeigen deutlich, daß es Anwendungssituationen gibt, in denen die Zusammenführung wahrnehmungsintegrierender Handlungsmerkmale in der im Abschnitt 3.4.1 erörterten Weise alleine nicht ausreicht, um im Sinne einer Anwendbarkeit günstige Beschreibungen operationaler Begriffe zu erhalten. Möglich wird dies erst, wenn zudem Informationen über situationsabhängig einsetzbare Beziehungen zwischen Richtungsangaben, wie sie etwa durch die Relation `hw_gegenüberliegende_Richtung/2` repräsentiert werden, in die Beschreibung operationaler Begriffe aufgenommen werden. Die im Kapitel 5 vorgetragenen Ergebnisse zum Lernen operationaler Begriffe zeigen, daß die Aufnahme dieser Informationen in die Beschreibung einzelner operationaler Begriffe dem verwendeten maschinellen Lernverfahren überlassen werden kann, wenn sie dem Verfahren für den Lernvorgang als Hintergrundwissen zur Verfügung gestellt werden. Hierzu werden die Relationen in Form von Grundfakten repräsentiert.

Im folgenden werden die wichtigsten Beziehungen zwischen Richtungsangaben beschrieben. Zur Illustration ihrer Verwendung in operationalen Begriffen werden zudem auch mögliche Beschreibungen weiterer operationaler Begriffe vorgestellt.

$\boxed{\texttt{hw\_WRichtung\_nach\_Rotation}(WRichtung, DRichtung, NWRichtung)}$

- `hw_WRichtung_nach_Rotation/3` gibt für die Durchführung einer 90°-Drehung mit Drehsinn *DRichtung* die Wahrnehmungsrichtung *NWRichtung* an, in der ein Objekt nach Durchführung der Rotation wahrgenommen wird, wenn es vor der Rotation in Richtung *WRichtung* registriert wurde.
- Mögliche Werte für
  - *WRichtung* sind: *links, rechts, vorne, hinten, rechts_vorne, rechts_hinten, links_vorne* und *links_hinten.*
  - *DRichtung* sind: *rechts* und *links.*

- *NWRichtung* sind: *siehe WRichtung.*

- Es exitieren 16 `hw_WRichtung_nach_Rotation/3` Fakten. Im einzelnen werden sie im Anhang D aufgeführt werden.

$\boxed{\texttt{hw\_neue\_Wahrnehmungsseite}(Seite, NSeite)}$

- `hw_neue_Wahrnehmungsseite/2` legt bei einem Wert `schmale_Seite` für *Seite* den Wert `breite_Seite` und bei einem Wert *breite_Seite* den Wert *schmale_Seite* für *NSeite* fest.
- Mögliche Fakten sind:
  `hw_neue_Wahrnehmungsseite`$(breite\_Seite, schmale\_Seite)$ und
  `hw_neue_Wahrnehmungsseite`$(schmale\_Seite, breite\_Seite)$.

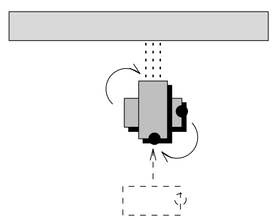


Abbildung 3.10: Beispiel `drehe_dich_vor_der_Wand/4`

Zur Veranschaulichung der Verwendung von `hw_WRichtung_nach_Rotation-/3` und `hw_neue_Wahrnehmungsseite/2` in operationalen Begriffen diene als Beispiel der operationale Begriff `drehe_dich_vor_der_Wand/4`. In Abbildung 3.10 wird hierzu eine Beispielsituation illustriert.

`stehen`$(Trc, T_1, T_2,$ `vor_einer_Wand`$, WRichtung, WSeite, LW)$ &
`hw_WRichtung_nach_Rotation`$(WRichtung, DRichtung, WRichtung1)$ &
`drehen`$(Trc, T_2, T_3, RGK, DRichtung,$ `vor_einer_Wand`$, WRichtung1)$ &
`hw_neue_Wahrnehmungsseite`$(WSeite, WSeite1)$ &
`stehen`$(Trc, T_3, T_4,$ `vor_einer_Wand`$, WRichtung1, WSeite1,$
`vor_einer_Wand`$)$
$\longrightarrow$ **drehe_dich_vor_der_Wand**(Trc, T_1, T_4, T_3),

Der operationale Begriff `drehe_dich_vor_der_Wand/4` beschreibt eine 90° Rotation vor einer Wand, die in Richtung *WRichtung* wahrgenommen wird. In der in Abbildung 3.10 dargestellten Situation liegt beispielsweise die Wand vor der Durchführung der Drehbewegung in Richtung *vorne*. Die Ausführung der Rotation wird durch das wahrnehmungsintegrierende Handlungsmerkmal `drehen/7` des operationalen Begriffes beschrieben, wobei die Rotationsbewegung mit Drehsinn *DRichtung* solange durchgeführt wird, bis die Wand, die vor Beginn der Drehbewegung in Richtung *vorne* registriert wurde, in Richtung *WRichtung1* wahrgenommen wird. Die Handlungsausführung erfolgt demnach in dem Sinne unter der Kontrolle einer Wahrnehmung, als daß die Wahrnehmung hier das Handlungsende bestimmt. Hierzu ist es jedoch erforderlich, daß die zu erwartende Wahrnehmungsrichtung *WRichtung1* der Wand bereits mit Beginn der Rotation bekannt ist, da unter Umständen die Drehung mit der Wahrnehmung eines — aus der Sicht der Sensoren wandähnlichen — Objektes aus einer anderen Richtung zu früh abgebrochen werden könnte. Die Wahrnehmungsrichtung *WRichtung1* ist jedoch mit der Festlegung der Rotationsrichtung *DRichtung* und der bekannten Wahrnehmungsrichtung *WRichtung* eindeutig bestimmbar. So ist der Wert für *WRichtung1* für die in Abbildung 3.10 dargestellt Drehung in Richtung *rechts*, abhängig von der alten Wahrnehmungsrichtung *vorne*, gerade *links*. Dieser Zusammenhang wird durch `hw_WRichtung_nach_Rotation/3` in die Beschreibung des operationalen Begriffes aufgenommen. Über einen entsprechenden Fakt `hw_WRichtung_nach_Rotation`$(vorne, rechts, links)$ kann die Variable *WRichtung1* vor Beginn der Rotation an den richtigen Wert gebunden werden.

Bei einer erfolgreichen Anwendung von `drehe_dich_vor_der_Wand/4` ist auch der Wert für *WSeite1* des die Verifikationssituation beschreibenden wahrnehmungsintegrierenden Handlungsmerkmals `stehen/7` eindeutig. Falls die Wand vor der Durchführung der 90° Rotation von Sensoren einer der beiden breiten Seiten des Roboters registriert wurde, so wird sie nach Ausführung der Rotation von Sensoren einer schmalen Seite erfaßt werden, bzw. analog auch in umgekehrter Weise. In der Beschreibung des operationalen Begriffes `drehe_dich_vor_der_Wand/4` wird dieser Zusammenhang durch `hw_neue_Wahrnehmungsseite/2` wiedergegeben.

`hw_WRichtung_nach_Bewegung`$(WRichtung, Richtung, NWRichtung)$

- `hw_WRichtung_nach_Bewegung/3` gibt für die Durchführung einer Vorwärtsbewegung in Richtung *Richtung* die Wahrnehmungsrichtung *NWRichtung* an, in der ein Objekt während der Durchführung der Vorwärtsbewegung wahrgenommen wird, wenn es vor der Bewegung in Richtung *WRichtung* registriert wurde.
- Mögliche Werte für
  - *WRichtung* sind: *links, rechts, vorne, hinten, rechts_vorne, rechts_hinten, links_vorne* und *links_hinten.*
  - *Richtung* sind: *links, rechts, hinten* und *vorne.*
  - *NWRichtung* sind: *links* und *rechts*
- Es exitieren 16 `hw_WRichtung_nach_Bewegung` Fakten. Im einzelnen werden sie im Anhang D aufgeführt werden.

`hw_WRichtung_einer_Ecke1`$(WRichtung, Richtung, NWRichtung)$

- `hw_WRichtung_einer_Ecke1/3` gibt für die Durchführung einer Vorwärtsbewegung in Richtung *Richtung*, die parallel zu einer Wand erfolgt, die vor Beginn der Bewegung in Richtung *WRichtung* wahrgenommen wurde, die Richtung *NWRichtung* an, in der eine mögliche Ecke registriert werden würde.

- Mögliche Werte für

  - *WRichtung* sind: *rechts, links, hinten* und *vorne.*
  - *Richtung* sind: *siehe WRichtung.*
  - *NWRichtung* sind: *links_vorne* und *rechts_vorne.*

- Es exitieren 8 `hw_WRichtung_einer_Ecke1/3` Fakten. Im einzelnen werden sie im Anhang D aufgeführt werden.

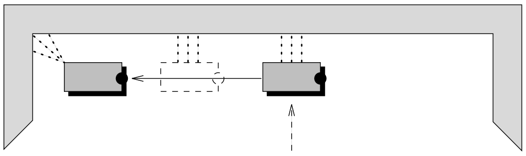


Abbildung 3.11: Beispiel `bewege_dich_parallel_zur_Wand_in_die_Ecke/4`

Zur Verdeutlichung der Verwendung von `hw_WRichtung_nach_Bewegung-/3` und `hw_WRichtung_einer_Ecke1/3` in operationalen Begriffen diene als Beispiel der operationale Begriff `bewege_dich_parallel_zur_Wand_in_die_-Ecke/4`. In Abbildung 3.11 wird eine Situation dargestellt, in der dieser Begriff Anwendung findet.

$\texttt{stehen}(Trc, T_1, T_2, \texttt{vor\_einer\_Wand}, WRichtung, WSeite, LW)$ &
$\texttt{hw\_WRichtung\_nach\_Bewegung}(WRichtung, Richtung, WRichtung1)$ &
$\texttt{p\_bewegen}(Trc, T_2, T_3, GK, Richtung, \texttt{entlang\_einer\_Wand}, WRichtung1)$&
$\texttt{hw\_WRichtung\_einer\_Ecke1}(Richtung1, Richtung, WRichtung2)$ &
$\texttt{stehen}(Trc, T_3, T_4, \texttt{parallel\_in\_einer\_Ecke}, WRichtung2, WSeite,$
$\quad \texttt{entlang\_einer\_Wand})$
$\longrightarrow$ **bewege_dich_parallel_zur_Wand_in_die_Ecke**(Trc, T_1, T_4, T_3),

Ausgangspunkt des Beispiels ist eine Situation, in der der Roboter zunächst auf einer Stelle steht und dabei die Wahrnehmung `vor_einer_Wand/4` in Richtung *vorne* registriert. Die Anwendung von `bewege_dich_parallel_-zur_Wand_in_die_Ecke/4` sieht nun die Durchführung des wahrnehmungsintegrierenden Handlungsmerkmals `p_bewegen/7` vor, die eine Angabe der Bewegungsrichtung *Richtung* und der Wahrnehmungsrichtung *WRichtung1* der Wand für die Kontrolle der Bewegung parallel zu ihr erfordert. Die

Durchführung des Auftrages sich in die Ecke zu bewegen, kann in der dargestellten Umgebung sowohl durch eine Bewegung in Richtung *links* als auch durch eine Bewegung in Richtung *rechts* erfüllt werden. Die Festlegung der tatsächlich einzuschlagenden Bewegungsrichtung könnte, falls sie durch den Auftrag nicht näher bestimmt ist, durch eine übergeordnete Strategie bestimmt werden, nach der unbekannte Räume vielleicht stets links herum den Wänden folgend erkundet werden. Abhängig von der Wahl der Bewegungsrichtung und der Richtung in der die Wand vor Beginn der Bewegung wahrgenommen wird, ist auch die neue Wahrnehmungsrichtung *WRichtung1* bestimmt. Für eine Bewegung in Richtung *links*, wie sie in Abbildung 3.11 dargestellt wird, ist dies beispielsweise die Wahrnehmungsrichtung *rechts*. Dieser Zusammenhang wird durch `hw_WRichtung_nach_Bewegung/3` ausgedrückt. Entsprechend gibt es einen Fakt `hw_WRichtung_nach_Bewegung`$(vorne, links, rechts)$.

Die Verifikationssituation von `bewege_dich_parallel_zur_Wand_in_die_Ecke/4` wird durch ein wahrnehmungsintegrierendes Handlungsmerkmal `stehen/7` beschrieben, für deren Ableitung der Roboter die Wahrnehmung `parallel_in_einer_Ecke/4` in Richtung *WRichtung2* registrieren muß. Die Festlegung von *WRichtung2* wird dabei durch die Wahl von *WRichtung1* und *Richtung* bestimmt. Für die dargestellte Bewegung in Richtung *links* und die Wahrnehmungsrichtung der Wand während dieser Bewegung in Richtung *rechts*, ist beispielsweise die Ecke in Richtung *rechts_vorne* zu erwarten, was durch `hw_WRichtung_einer_Ecke1/3` in der Beschreibung von `bewege_dich_parallel_zur_Wand_in_die_Ecke/4` ausgedrückt wird. Entsprechend hierzu gibt es einen Fakt `hw_WRichtung_einer_Ecke`$(rechts, links, rechts\_vorne)$.

$\boxed{\texttt{hw\_WRichtung\_einer\_Ecke2}(WRichtung, DRichtung, NWRichtung)}$

- `hw_WRichtung_einer_Ecke2/3` gibt für die Durchführung einer 90° Drehung mit Drehsinn *DRichtung* die Wahrnehmungsrichtung *NWRichtung* an, in der eine Ecke nach Durchführung der Rotation wahrgenommen wird, wenn sie vor der Rotation in Richtung *WRichtung* registriert wurde.
- Mögliche Werte für
  - *WRichtung* sind: *rechts_vorne, links_vorne, rechts_hinten* und *links_hinten*.
  - *DRichtung* sind: *links* und *rechts*.
  - *NWRichtung* sind: *siehe WRichtung*.
- Es exitieren 8 `hw_WRichtung_einer_Ecke2/3` Fakten. Im einzelnen werden sie im Anhang D aufgeführt werden.

$\boxed{\texttt{hw\_gegenüberliegende\_Richtung}(Richtung, NRichtung)}$

- `hw_gegenüberliegende_Richtung/2` gibt zur Richtung *Richtung* die entgegengesetzte Richtung *NRichtung* an.

- Mögliche Fakten sind:
  `hw_gegenüberliegende_Richtung`$(rechts, links)$,
  `hw_gegenüberliegende_Richtung`$(links, rechts)$,
  `hw_gegenüberliegende_Richtung`$(vorne, hinten)$ und
  `hw_gegenüberliegende_Richtung`$(hinten, vorne)$.

3.5 Zusammenfassung

Im vorstehenden Kapitel wurde im Abschnitt 3.4 eine Repräsentation operationaler Begriffe vorgestellt, die sich einerseits nahtlos in die im Abschnitt 2.2 beschriebene Begriffsstruktur handlungsorientierter Wahrnehmungsmerkmale einfügt und andererseits die Verwendung operationaler Begriffe als Planungsoperatoren in Aussicht stellt. Zu diesem Zweck wurde im Abschnitt 3.3 eine Repräsentation wahrnehmungsintegrierender Handlungsmerkmale vorgestellt, die eine Verbindung zwischen handlungsorientierten Wahrnehmungsmerkmalen und den im Abschnitt 3.2 vorgestellten Basis-Handlungsmerkmalen schaffen.

Während die Basis-Handlungsmerkmale aus den vom Roboter zur Verfügung gestellten Daten nach dem im Abschnitt 3.2.2 beschriebenen Verfahren inkrementell berechnet werden können, sollen Regeln zur Beschreibung von wahrnehmungsintegrierenden Handlungsmerkmalen und operationalen Begriffen ebenso wie für handlungsorientierte Wahrnehmungsmerkmale durch maschinelle Lernverfahren in der Lernphase des Roboters erworben werden. Die Trennung zwischen Lern- und Arbeitsphase des Roboters wurde im Abschnitt 3.1 erörtert.

Hiernach werden dem Roboter in der Lernphase bestimmte Handlungen in Abhängigkeit von bestimmten Wahrnehmungen vorgeführt (z.B. *joystick controlled*). Während einer solchermaßen kontrollierten Roboterfahrt erfolgt eine inkrementelle Berechnung von Basis-Handlungs- und Basis-Wahrnehmungsmerkmalen, aus denen dann im Rahmen der in diesem Kapitel beschriebenen Begriffshierarchie operationale Begriffe gelernt werden. In der Arbeitsphase erfolgt hingegen der Einsatz der gelernten Begriffe zur Erreichung bestimmter Zielsituationen. Basis-Handlungsmerkmale dienen hier als an den Roboter abzusetzende Handlungsanweisungen, auf die in Abhängigkeit von registrierten Wahrnehmungen über die gelernte Begriffshierarchie geschlossen wird.

Neben der Lern- und Arbeitsphase des Roboters wurde im Abschnitt 3.1 eine Anforderungsliste an operationale Begriffe vorgestellt, die als Leitlinie der darauffolgenden Abschnitte zur Entwicklung operationaler Begriffe diente. Abbildung 3.12 zeigt noch einmal schematisch die vorgestellte Begriffshierarchie zur Repräsentation operationaler Begriffe mit den zugehörigen Lernaufgaben im Überblick.

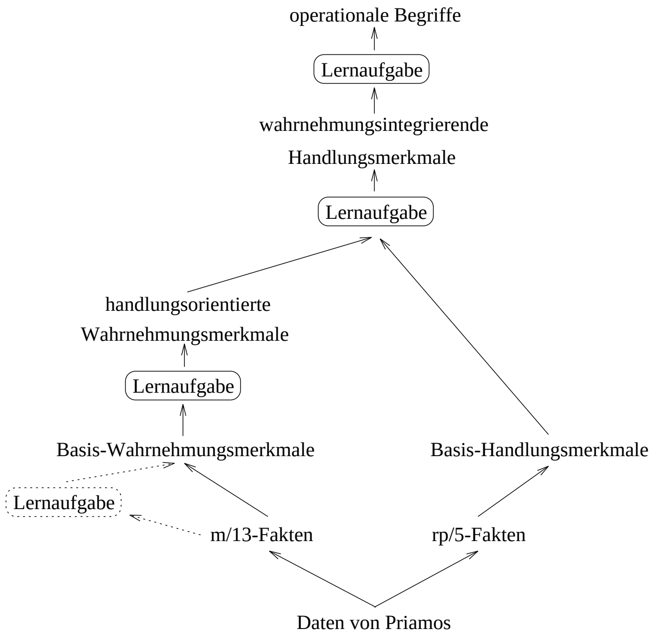


Abbildung 3.12: Begriffshierarchie zur Repräsentation operationaler Begriffe

Am Ende dieses Kapitels steht nun sowohl eine Repräsentation operationaler Begriffe als auch ein Konzept für ihren Einsatz bereit. Für eine praktische Anwendung operationaler Begriffe in der Arbeitsphase des Roboters bedarf es jedoch noch sowohl eines Moduls zur Ausführungsüberwachung operationaler Begriffe als auch eines für operationale Begriffe zugeschnittenen Planungssystems. Beide noch zu entwickelnden Systeme sind nicht Bestandteil der vorliegenden Arbeit. Welche Aufgaben von einer Ausführungsüberwachungskomponente übernommen werden müssen, wurde jedoch schon zum Teil an den Beispielen operationaler Begriffe in den Abschnitten 3.2.1 und 3.2.2 deutlich gemacht. Es wurde ebenfalls Wert darauf gelegt, klar darzulegen, daß eine Verkettung operationaler Begriffe über ihre Anwendbarkeits- und Verifikationssituationen möglich ist, so daß der Weg zur Entwicklung eines Planungssystems für operationale Begriffe offen ist.

Im Kapitel 6 wird der Gedanke operationale Begriffe als Planungsoperatoren einzusetzen noch einmal ausblickend aufgegriffen werden. Das Lernen operationaler Begriffe wird hingegen Gegenstand von Kapitel 5 sein. Im Kapitel 4 wird jedoch zuvor eine Oberfläche zur Unterstützung einer interaktiven Beispielgenerierung vorgestellt werden, die in Zukunft zu einer Plattform für sämtliche Anwendungskomponenten operationaler Begriffe ausgeweitet werden kann.

Kapitel 4

Oberfläche zur Unterstützung der Beispielgenerierung

Nachdem im vorangegangenen Kapitel eine Repräsentation operationaler Begriffe zum Lernen aus Roboter-Sensordaten vorgestellt wurde, steht im vorliegenden Kapitel die zum Lernen von Begriffen der vorgeschlagenen Begriffshierarchie notwendige Beispielgenerierung im Vordergrund. Das mit diesem Kapitel verfolgte Ziel ist es jedoch nicht, einzelne Algorithmen zur Beispielgenerierung vorzuschlagen, sondern die Entwicklung einer Oberfläche vorzustellen, die zur Unterstützung der Beispielgenerierung einsetzbar ist. Warum eine Unterstützung der Beispielgenerierung sinnvoll ist, und wie sie gestaltet werden kann, wird im einzelnen in Abschnitt 4.1 erörtert werden. Im Abschnitt 4.2 wird dann die Unterstützungsoberfläche konkret über die Vorstellung ihrer bisher realisierten wesentlichen Funktionen näher beschrieben werden. Denkbare Weiterentwicklungen werden das Thema von Abschnitt 4.3 sein. Das Kapitel wird schließlich durch eine kurze Zusammenfassung im Abschnitt 4.4 beendet werden.

Implementiert wurde die Oberfläche mit Hilfe der von Quintus Prolog (Realease 3.1.1) gelieferten Schnittstelle zum X-Window-System. Im einzelnen sind alle Komponenten, die eine Grafikausgabe beinhalten, in ProXL — der direkten Schnittstelle zwischen Prolog und Xlib — programmiert, während die sonstigen Programmteile in ProXT — der Prolog-Schnittstelle zum Motif-Toolkit und zu den X-Toolkit-Intrinsics — realisiert sind. Auf eine Darstellung von Einzelheiten der Implementierung wird in der vorliegenden Arbeit aus Platz- und Übersichtsgründen bewußt verzichtet. Eine gute Einführung in die Programmierung des Motif-Toolkits und des X-Window-Systems findet sich in [Gottheil *et al.*, 1992].[1]

[1] Im Gegensatz dazu, ist das von Quintus Prolog mitgelieferte Handbuch zur Beschreibung der X-Window Schnittstelle in keinster Weise zu empfehlen. Zum Bedauern des Autors ist es jedoch für die Programmierung in Quintus Prolog unverzichtbar, es als Nachschlagewerk zu Rate zu ziehen.

4.1 Motivation

Die Idee der im Abschnitt 3.1 eingeführten Lernphase des Roboters ist, dem Roboter bestimmte Handlungen in Abhängigkeit von bestimmten Wahrnehmungen während einer kontrollierten Fahrt durch eine unbekannte Umgebung vorzuführen. Durch den Einsatz maschineller Lernverfahren sollen dann aus dieser Trainingsfahrt Begriffe aus der im vorangegangenen Kapitel entwickelten Begriffshierarchie gelernt werden. Hierzu ist die Vorgabe bzw. die Bildung geeigneter Beispiele[2] erforderlich, die die Trainingsfahrt des Roboters in der Terminologie der aufgebauten Begriffshierarchie hinreichend genau charakterisieren.

Während Basis-Handlungs- und Basis-Wahrnehmungsmerkmale bereits während der Trainingsfahrt inkrementell aus den vom Roboter zur Verfügung gestellten Daten automatisch berechnet werden können[3], setzt die Bildung von Beispielen für handlungsorientierte Wahrnehmungsmerkmale, wahrnehmungsintegrierende Handlungsmerkmale und operationale Begriffe eine Klassifikation von in bestimmten Zeitabschnitten der Fahrt registrierten Objekten bzw. Kantenkonstellationen von Objekten der Umgebung voraus, die eine Automatisierung der Beispielgenerierung erschwert.

Unter diesen Aspekt fällt sicher auch die zum Lernen einiger Begriffe notwendige Fokussierung auf ganz bestimmte Zusammenhänge zwischen Handlungen und Wahrnehmungen mit Hilfe der Beispielvorgabe. Steht beispielsweise der Roboter während der Trainingsfahrt vor einer Tür und registriert neben der Tür auch die Wand rechts neben ihm, bevor er die Tür durchfährt, so kann der operationale Begriff `bewege_dich_durch_die_Tür/4` während dieser Trainingsfahrt nur dann gelernt werden, wenn die Beispielgenerierung den Zeitraum des Stehenbleibens nicht nur durch ein Beispiel für das handlungsorientierte Wahrnehmungsmerkmal `vor_einer_Wand/4`, sondern zumindestens auch noch durch ein Beispiel für das Wahrnehmungsmerkmal `vor_einer_Tür/4` erfaßt. Denn es ist ja gerade der Zusammenhang zwischen dieser Wahrnehmung und der sich anschließenden Handlung, der gelernt werden soll. Eine erfolgreiche Beispielgenerierung umfaßt demnach nicht nur die reine Erzeugung von Beispielmengen, sondern darüber hinaus auch eine gezielte Auswahl bestimmter Beispiele zur Steuerung des sich anschließenden Lernprozesses.

Während für Teile der Wahrnehmungsseite der Begriffshierarchie (zu diesen gehören neben den Basis-Wahrnehmungsmerkmalen, Sensor-Merkmale, Sensorgruppen-Merkmale und handlungsorientierte Wahrnehmungsmerk-

[2] Unter Beispielen werden im Rahmen dieser Arbeit stets Grundfakten verstanden. So ist beispielsweise *drehen(t203,171,180,mittlere_Drehung,links,vor_einer_Tür,vorne)* ein Beispiel für das wahrnehmungsintegrierende Handlungsmerkmal `drehen/7`.

[3] Wie bereits im Abschnitt 2.2.2 beschrieben, gilt für die Basis-Wahrnehmungsmerkmale zudem im besonderen, daß sie in Abhängigkeit von einem bestimmten Parametersatz berechnet werden. Im Falle nicht ausreichender Lernbarkeit der Wahrnehmungsmerkmale der Begriffshierarchie wird dieser Parametersatz mit Hilfe einer Bewertungsfunktion modifiziert. Anschließend wird der Prozeß zur Bildung der Basis-Merkmale mit dem dann veränderten Parametersatz wiederholt [Klingspor und Morik, 1995].

male) bereits erste Algorithmen zur automatischen Beispielgenerierung entwickelt sind[4], werden Beispiele für Begriffe der Handlungsseite (zu diesen gehören neben den Basis-Handlungsmerkmalen, wahrnehmungsintegrierende Handlungsmerkmale und operationale Begriffe) z.Z. noch *per Hand* für eine Trainingsfahrt vorgegeben. Ein Ziel für zukünftige Arbeiten ist daher sicherlich, eine weitgehend automatische Generierung von Beispielen für die gesamte Begriffshierarchie zu erreichen. In Anbetracht der bereits gemachten Erfahrungen auf der Wahrnehmungsseite und den oben angedeuteten Schwierigkeiten ist jedoch nicht zu erwarten, daß die Beispielgenerierung vollkommen losgelöst von menschlicher Kontrolle erfolgen können wird. Es müssen demnach Mittel zur Verfügung gestellt werden, auch in automatisch generierten Beispielmengen nachträgliche Korrekturen gezielt durchführen zu können.

Diese Überlegungen führten zur Entwicklung einer Oberfläche zur Unterstützung der Beispielgenerierung. Diese Unterstützung, die sich z.Z. noch lediglich auf die Generierung von Beispielen für Begriffe der Handlungsseite der Begriffshierarchie bezieht (mit Ausnahme der handlungsorientierten Wahrnehmungsmerkmale), umfaßt neben einer Visualisierung der Trainingsfahrt das Ein- und Auslesen von Beispieldateien, die Möglichkeit Beispielmengen zu edieren und unter der Kontrolle einer Komponente zur Überprüfung der syntaktischen Korrektheit einzelner Beispiele zu modifizieren, sowie die Einbindung des unter Abschnitt 3.2.2 beschriebenen Verfahrens zur Berechnung der Basis-Handlungsmerkmale aus den von PRIAMOS zur Verfügung gestellten Daten. Die Visualisierung der Trainingsfahrt bietet neben der Möglichkeit einer exakten Nachverfolgung der vom Roboter während der Fahrt durchgeführten Handlungen und der Wiedergabe einer maßstabsgetreuen Darstellung der durchfahrenen Umgebung die Anzeige der von den Sensoren zu jedem Meßzeitpunkt der Fahrt angepeilten Raumpunkte.

Die Anzeige der angepeilten Raumpunkte für jeden Meßzeitpunkt der Fahrt ist, insbesondere im Falle einer bekannten Umgebung, ein unentbehrliches Hilfsmittel zur Unterstützung der Beispielgenerierung für die Wahrnehmungsmerkmale aller Abstraktionsstufen, und zwar unabhängig davon, ob die Beispielgenerierung nun automatisch, oder über manuelle Eingaben erfolgt. Mit ihrer Hilfe kann überprüft werden, ob bestimmte Wahrnehmungen überhaupt von den Sensoren im Verlaufe der Trainingsfahrt erfaßt werden konnten. Die Unterstützungsoberfläche bietet also in dieser Hinsicht eine echte Inspizierbarkeit der Beispielmengen im Sinne von [Morik, 1989]. Findet nun ein Beispielgenerierungsalgorithmus für eine durchgeführte Trainingsfahrt kein Beispiel für die Wahrnehmung `vor_einer_Tür/4`, obwohl der Roboter während der Fahrt vor einer Tür stehen blieb, so muß dies nicht notwendigerweise eine Schwäche des eingesetzten Verfahrens sein, sondern es kann ebensogut daran liegen, daß die Sensorik die Tür, von der vom Roboter eingenommenen Raumposition aus, nicht richtig erfassen konnte. Sollte es hingegen ein Problem des eingesetzten Verfahrens sein, so kann

[4]Diese Algorithmen sind kein Bestandteil der vorliegenden Arbeit.

ein entsprechendes Beispiel über die Nutzung der Unterstützungsoberfläche nachträglich in die zugehörige Beispielmenge aufgenommen werden.

Die Komponente zur Überprüfung der syntaktischen Korrektheit von Beispielen stellt dabei stets sicher, daß manuell eingetragene, oder aber manuell veränderte, automatisch generierte Beispiele im Rahmen der im Abschnitt 2.2 bzw. im Kapitel 3 vorgestellten Repräsentationsprache bleiben. Abweichungen von der vorgegebenen Repräsentationsprache — hierzu zählen beispielsweise nicht bekannte Prädikatsnamen, fehlende Argumente, oder unbekannte Argumentbezeichnungen — führen zu einer Ablehnung der Aufnahme des betreffenden Beispiels in die zugehörige Beispielmenge, sowie zu einer Meldung, die über den möglicherweise vorhandenen Repräsentationsfehler Auskunft gibt.

Die Entwicklung der im folgenden Abschnitt 4.2 in ihren wesentlichen Funktionen vorgestellten Unterstützungsoberfläche ist zum jetzigen Zeitpunkt keineswegs als abgeschlossen zu betrachten. Sie stellt jedoch in ihrer jetzigen Form eine geeignete Plattform für einen weitergehenden zukünftigen Ausbau dar. Einige Ideen hierzu werden gegen Ende des vorliegenden Kapitels im Abschnitt 4.3 dargelegt werden.

4.2 Funktionalität

Wie bereits im vorangegangenen Abschnitt angekündigt, werden im vorliegenden Abschnitt die wesentlichen, bereits implementierten Funktionen der Unterstützungsoberfläche vorgestellt werden. Hierzu zählen:

- das Lesen und Schreiben von Beispieldateien,
- der Aufruf und die Ausführung von Programmen zur Beispiel- und Datenerzeugung,
- die grafische Darstellung einer Trainingsfahrt,
- die Möglichkeit, Beispiele nach ihren Typen sortiert zu edieren, zu verändern, zu löschen und einzugeben, sowie
- die Überprüfung von Beispieleingaben hinsichtlich ihrer Konsistenz mit der vorgegebenen Repräsentation.

Der vorliegende Abschnitt ist jedoch keineswegs als eine Art „Handbuch“ zur Benutzerführung zu verstehen, d.h. im besonderen, daß nicht alle Popup- und Pulldown-Menüs mit den ihnen zugehörigen Sub-Menüs und Menüpunkten vorgestellt werden. Das Ziel dieses Abschnittes ist vielmehr, mit Hilfe einer kurzen Darstellung, einen Eindruck über die Funktionalität der Unterstützungsoberfläche zu vermitteln. Hierzu werden zur Illustration gezielt einige konkrete Bedienungsmerkmale der Oberfläche erörtert werden.

Mit dem Aufruf des Programmes erscheint die in Abbildung 4.1 dargestellte Hauptbedienungskomponente der Unterstützungsoberfläche. Sie setzt sich

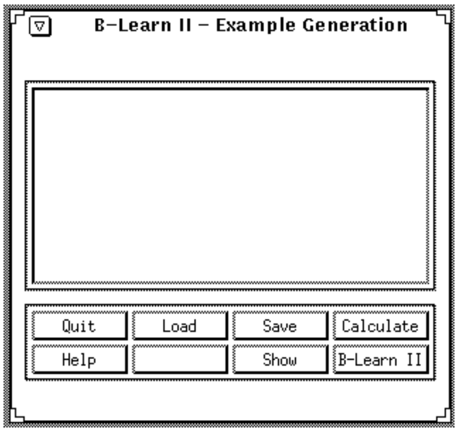


Abbildung 4.1: Hauptbedienungskomponente der Unterstützungsoberfläche

aus einer zweireihigen Menüleiste, hinter der sich mehrere Pulldown-Menüs verbergen[5], und einem Nachrichtenfenster zur Darstellung von Informationen des Systems an den Benutzer bzw. die Benutzerin zusammen.

Im Nachrichtenfenster werden zudem sämtliche Aktionen des Benutzers bzw. der Benutzerin — etwa das Löschen eines Beispiels einer Beispielmenge — mitprotokolliert, wobei der ins Fenster eingetragene Informationstext bei bereits voller Belegung des Fensters für neue Einträge nach oben aus dem sichtbaren Bereich weggeschoben wird und in einer temporären Protokolldatei gesichert wird. Mit Hilfe der Cursor-Tasten ist es dem Benutzer bzw. der Benutzerin jedoch möglich, die bereits aus dem Nachrichtenfenster verschwundenen Textteile zurückzuholen. In diesem Sinne fungiert das Nachrichtenfenster als ein über die Protokolldatei hin und her schiebbares Sichtfenster. Da sich zudem Textteile des Nachrichtenfensters mit Hilfe der Maustasten in den weiter unten beschriebenen Editor zur Beispielgenerierung kopieren lassen, eröffnet dieses besondere Merkmal der Oberfläche dem Benutzer bzw. der Benutzerin die Möglichkeit, irrtümlicherweise gelöschte Beispiele wieder in die zugehörige Beispielmenge einzutragen, ohne dabei das betreffende Beispiel über die Tastatur neu eingeben zu müssen.

Für jedes hinter der Menüleiste verborgene Pulldown-Menü enthält die Menüleiste einen Titel (*Quit, Load, Save, Calculate, ...*), der über die Funktion des Menüs Auskunft gibt. Durch Anklicken eines Titels mit der Maus klappt das zugehörige Menü herunter. Abbildung 4.2 zeigt zur Illustration das hinter dem Menüpunkt *Calculate* verborgene Pulldown-Menü mit den

[5]An dieser Stelle ist vielleicht als besonderes Merkmal der Implementierung erwähnenswert, daß diese Menüleiste mit den dahinter liegenden Pulldown-Menüs mit Hilfe eines eigens für die Oberfläche entwickelten Moduls zur Generierung von hierarchischen Pulldown-Menüs erzeugt wird. Da Pulldown-Menüs im wesentlichen immer nach dem gleichen Schema programmiert werden, ist die mit diesem Modul verfolgte Idee, den Aufbau verschiedener Menüs immer mit denselben Programmteilen zu realisieren, wobei individuelle Unterschiede in ihrer Gestaltung und Funktion in einer Spezifikationsdatei festgelegt werden. Der verwendete Programmcode wird damit insgesamt kürzer und übersichtlicher.

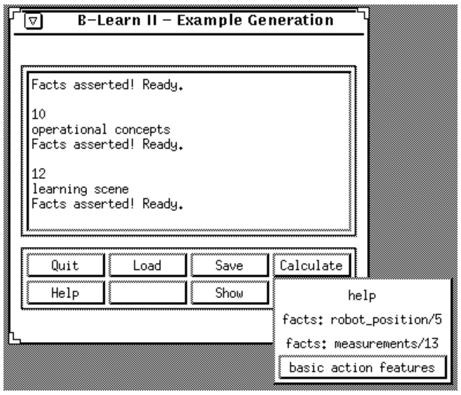


Abbildung 4.2: Pulldown-Menü hinter dem Menüpunkt *Calculate*

Menüpunkten *help, facts: robot_position/5, facts: measurements/13* und *basic action features*[6].

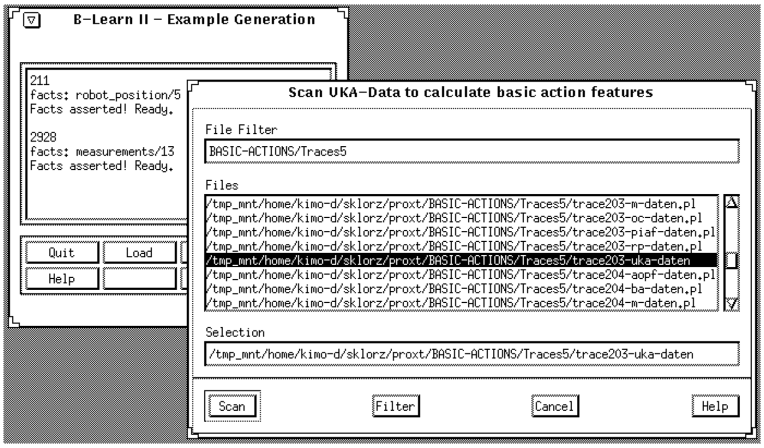


Abbildung 4.3: Beispiel für eine File-Selection-Box

Mit dem Anklicken einer dieser Menüpunkte erscheint eine File-Selection-Box (siehe hierzu Abbildung 4.3), über die der Benutzer bzw. die Benutzerin eine Datei auswählen kann, die beispielsweise für das in Abbildung 4.3 illustrierte Beispiel die vom Sensorkontroll- und Signalverarbeitungssystem des Roboters zur Verfügung gestellten Daten enthält. Mit der Aus-

[6] Die engl. Bezeichnungen *robot_position/5, measurement/13* bzw. *basic action features* stehen für die in der vorliegenden Arbeit bisher verwendeten Bezeichnungen rp/5-Fakten (Definition 2.4), m/13-Fakten (Definition 2.3) bzw. Basis-Handlungsmerkmale. Im weiteren Verlauf dieses Abschnittes werden sowohl deutsche als auch englische Bezeichner der verschiedenen Begriffe der vorgeschlagenen Begriffshierarchie verwendet. Im allg. sollte eine Übersetzung klar sein, so daß im einzelnen nicht mehr gesondert darauf hingewiesen wird.

wahl einer entsprechenden Datei werden aus diesen Daten die zu der betrachteten Trainingsfahrt gehörenden rp/5-Fakten, m/13-Fakten und Basis-Handlungsmerkmale berechnet, die von diesem Zeitpunkt an von der Unterstützungsoberfläche für einen Zugriff durch den Benutzer bzw. durch die Benutzerin verwaltet werden.

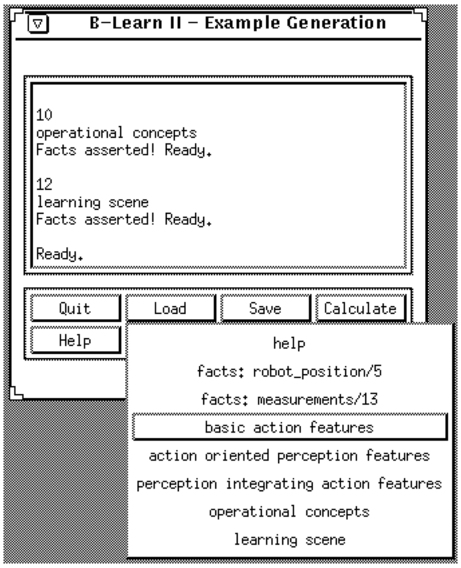


Abbildung 4.4: Pulldown-Menü hinter dem Menüpunkt *Load*

Dieser Zugriff kann zum Beispiel darin bestehen, sich die berechneten Daten anzeigen zu lassen, oder aber — wieder über eine File-Selection-Box — in Dateien zu schreiben. Die Menüpunkte *Show* und *Save* der Menüleiste verbergen hierzu entsprechende Pulldown-Menüs, deren zugehörige Menüpunkte auf die entsprechenden Datentypen (robot_position/5, measurements/13, basic action features) referieren. Über die Menüpunkte *Show* und *Save* können zudem auch eingetragene bzw. eingeladene Beispielmengen gesichert bzw. ediert werden. Das Einladen von Daten und Beispielen aus Dateien erfolgt dabei in analoger Weise über den Menüpunkt *Load*. Zur Illustration diene Abbildung 4.4.

Das Edieren von Beispielmengen wird über eine Selection-Box realisiert. Eine solche Box, wie sie Abbildung 4.5 zur Anzeige von wahrnehmungsintegrierenden Handlungsmerkmalen wiedergibt, besteht zum einen aus einem scrollbaren Fenster zur Darstellung einer Liste der bereits eingetragenen Beispiele und zum anderen aus einem Selektionsfeld, über das weitere Beispiele eingetragen, oder bereits eingetragene Beispiele modifiziert werden können. Die Veränderung bzw. das Löschen bereits eingetragener Beispiele erfolgt dabei über einen Mausklick auf das betreffende Beispiel im Listenfenster — worauf es automatisch ins Selektionsfeld kopiert wird — und die anschließende Betätigung des *Add-Buttons* — bei vorausgegangener Veränderung

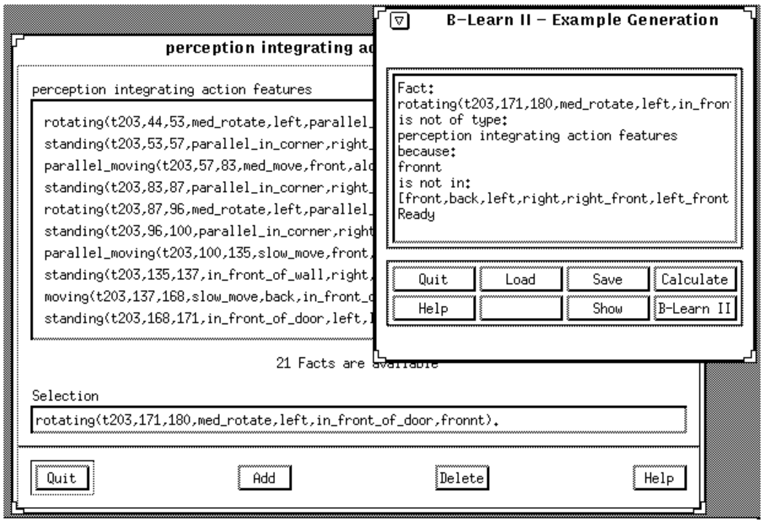


Abbildung 4.5: Selection-Box für eine Beispieleingabe

des Beispiels im Selektionsfeld — bzw. des *Delete-Buttons*. Neben der manuellen Eintragung eines Beispiels über die Tastatur ist auch das Kopieren von zuvor gelöschten Beispielen aus dem Nachrichtenfenster ins Selektionsfeld möglich. In jedem Fall wird aber mit der Betätigung des *Add-Buttons* vor einer Aufnahme in die betrachtete Beispielmenge überprüft, ob das im Selektionsfeld stehende Beispiel in der geforderten Repräsentation vorliegt[7].

Abbildung 4.5 dokumentiert den Versuch, das Beispiel `rotating`(*t203,171, 180,med_rotate,left,in_front_of_door,fronnt*) in die Beispielmenge wahrnehmungsintegrierender Handlungsmerkmale für die Trainingsfahrt *t203* aufzunehmen. Der Tipp-Fehler in der Wahrnehmungsrichtungsangabe „*fronnt*“ wird jedoch vom Modul zur syntaktischen Überprüfung von Beispieleingaben erkannt und verhindert somit korrekterweise die Aufnahme dieses Beispiels. Eine entsprechende Meldung im Nachrichtenfenster weist gleichzeitig den Benutzer bzw. die Benutzerin auf den gemachten Fehler hin (siehe hierzu das Nachrichtenfenster in Abbildung 4.5).

Über das Modul zur Überprüfung der Übereinstimmung einzutragender Beispiele mit der vorgegebenen Repräsentation kann natürlich nicht sichergestellt werden, daß diese Beispiele auch tatsächlich die durchgeführte Trainingsfahrt beschreiben. Ebensowenig möglich ist es, hierüber eine Fokussierung auf bestimmte Beispiele zur Steuerung des Lernprozesses abzuwickeln. Der Benutzer bzw. die Benutzerin kann jedoch bei der Bearbeitung dieser Problemstellungen vom Modul zur grafischen Darstellung der Trainingsfahrt

[7] Bzgl. der Implementierung ist vielleicht erwähnenswert, daß das die Überprüfung durchführende Modul sich hierbei einer Spezifikationsdatei bedient, in der die verschiedenen Beispieltypen über die Beschreibung ihrer möglichen Prädikatsnamen, ihrer Arität und Argumentsorten genau charakterisiert sind. Diese Spezifikationsdatei kann vom Benutzer bzw. von der Benutzerin auch in Unkenntnis der Implementierung des Überprüfungsmoduls in leichter Weise verändert oder um weitere Einträge ergänzt werden.

unterstützt werden.

In der Regel wird für Trainingsfahrten die durchfahrene Umgebung genau bekannt sein. Für diesen Fall ermöglicht die Unterstützungsoberfläche zur Beispielgenerierung eine nachträgliche zweidimensionale Visualisierung der Fahrt, die neben der maßstabsgetreuen Darstellung der durchfahrenen Umgebung eine bewegte Animation der vom Roboter durchgeführten Handlungen bietet. Darüberhinaus werden für jeden Meßzeitpunkt der Fahrt die von den Sensoren angepeilten Raumpunkte nachgezeichnet, so daß sich der Benutzer bzw. die Benutzerin anhand der Visualisierung einen Eindruck darüber verschaffen kann, welche Wahrnehmungen von den Sensoren während der Fahrt überhaupt registriert wurden.

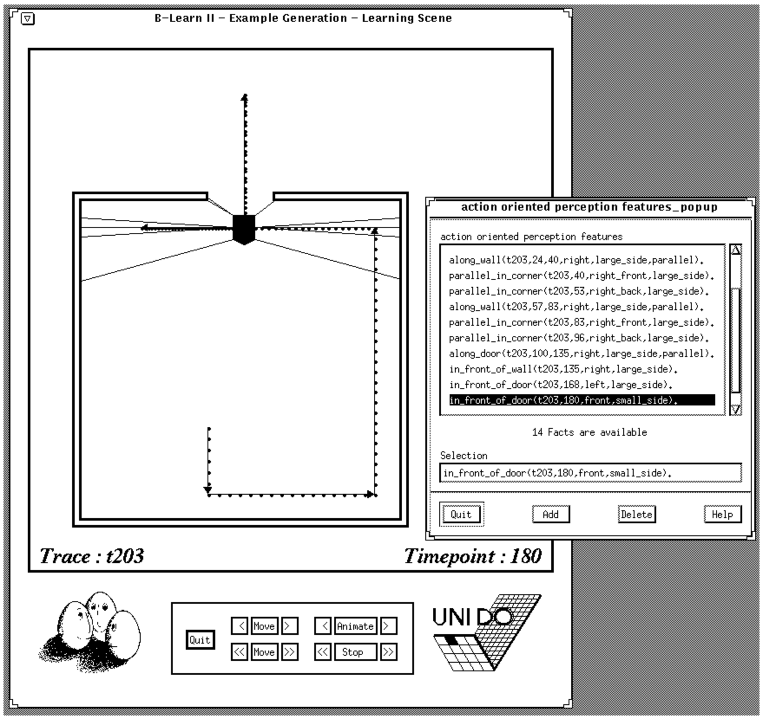


Abbildung 4.6: Visualisierung einer Trainingsfahrt

Zur Illustration des Visualisierungsmoduls diene Abbildung 4.6. Die grafische Darstellung zeigt, daß der Roboter zum Meßzeitpunkt *180*, der Trainingsfahrt *t203*, das Wahrnehmungsmerkmal `vor_einer_Tür/4` registriert. Damit ist klar, daß das eingetragene Beispiel `in_front_of_door`(*t203,180, front,small_side*) nicht im Widerspruch zu der durchgeführten Fahrt steht. Darüberhinaus wird durch die grafische Darstellung deutlich, daß der betrachtete Meßzeitpunkt der Fahrt ebenfalls durch die Wahrnehmungsbeispiele `in_front_of_wall`(*t203, 180,right,large_side*) und `in_front_of_wall` (*t203,180,left,large_side*) charakterisiert werden könnte. Ob diese Beispiele ebenfalls in die Beispielmenge handlungsorientierter Wahrnehmungsmerkmale aufgenommen werden oder nicht, könnte vom Benutzer bzw. der Benutzerin im Hinblick auf das durchzuführende Lernexperiment entschieden werden.

4.3 Perspektiven für die Weiterentwicklung

Die im Rahmen des Projektes BLearn II erarbeiteten und zum Teil auch schon in den vorangegangenen Kapiteln dieser Arbeit vorgestellten Ideen zum Aufbau und dem Einsatz operationaler Begriffe zur flexibleren Handhabung und Navigation mobiler Robotersysteme werden in der kommenden Projektphase die Basis für die Entwicklung eines prototypischen Steuerungssystems des Karlsruher Roboters PRIAMOS bilden. Im Hinblick auf diese Zielsetzung wird angestrebt, eine Benutzeroberfläche zu realisieren, über die sowohl die Lernphase als auch die Arbeitsphase des Roboters abgewickelt werden kann. Die im vorangegangenen Abschnitt vorgestellte Unterstützungsoberfläche zur Beispielgenerierung kann zu einem solchen System ausgebaut werden. Ohne auf eine nähere Systemarchitektur einzugehen, werden im vorliegenden Abschnitt einige der hierzu benötigten zusätzlichen Bedienungskomponenten aufgeführt.

Die vorgestellte Unterstützungsoberfläche zur Beispielgenerierung bietet zum Teil bereits Merkmale, die von einer Benutzeroberfläche zur Steuerung und Ablaufkontrolle der Lernphase des Roboters erwartet werden können. Hierzu zählt sicherlich neben der Visualisierungskomponente, über die eine nachträgliche Korrektur von Beispielmengen bzw. eine Fokussierung auf bestimmte Beispiele dieser Mengen erfolgen kann, auch das Modul zur Überprüfung der Übereinstimmung von Beispielen mit der vorgegebenen Repräsentation. Um die Unterstützungsoberfläche zu einer im obigen Sinne einsetzbaren Benutzeroberfläche auszubauen, sind jedoch noch eine Reihe weiterer Bedienungskomponenten zu realisieren. Hierzu zählt beispielsweise:

- die Erweiterung der internen Repräsentation der Unterstützungsoberfläche um die noch fehlenden Abstraktionsstufen der Wahrnehmungsseite,
- die Entwicklung bzw. die Einbindung von Algorithmen zur Beispielgenerierung,
- die Schaffung einer Möglichkeit, die vorgeschlagene Repräsentationsprache operationaler Begriffe über die Oberfläche zu variieren, bzw. zu ergänzen und
- die Einbindung der zum Lernen der Begriffshierarchie einsetzbaren Verfahren des maschinellen Lernens.

Zum Lernen der Begriffshierarchie sind bislang die in der Werkbank MOBAL [Morik *et al.*, 1993] eingebundenen Verfahren RDT [Kietz und Wrobel, 1992] und GRDT [Klingspor, 1994a] eingesetzt worden. Einige Resultate hierzu werden im Kapitel 5 noch vorgestellt werden. Diesen Verfahren gemeinsam ist, daß sie zur Durchführung des Lernprozesses Metawissen über die zu lernenden Regeln einsetzen, um auf diese Weise zu einer sinnvollen Einschränkung des Hypothesenraumes zu gelangen. Im Falle von RDT handelt es sich dabei um Schablonen (sog. *Regelschemata*) zur syntaktischen Beschreibung von zu lernenden Regeln, während für GRDT dieser Ansatz dahingehend erweitert wird, daß anstelle der Vorgabe einzelner Schemata eine Beschreibung von Mengen von Schablonen über Grammatiken vorgegebenen wird. Ohne auf weitere Einzelheiten dieser Verfahren einzugehen, sollte im Hinblick auf ihren möglichen Einsatz bereits an dieser Stelle festgehalten werden, daß unter den letzten der genannten Punkte der obigen Aufzählung sicherlich auch fallen wird,

- daß Regelschemata, Grammatiken und Hintergrundwissen zur Steuerung des Lernprozesses mit Hilfe der Oberfläche festgelegt werden können.

Für den Einsatz von Lernverfahren i.Allg. sollte es dem Benutzer bzw. der Benutzerin weiterhin möglich sein, mit Hilfe der Oberfläche:

- gelernte Regeln zu edieren und
- Konfigurationsdateien zur Durchführung von Lernexperimenten zu erstellen, zu speichern und auszuführen.

Um die Unterstützungsoberfläche zu einer Benutzeroberfläche auszubauen, die neben der Lernphase auch die Kontrolle der Arbeitsphase des Roboters durch einen Benutzer bzw. durch eine Benutzerin ermöglicht, wird es ferner erforderlich sein, eine Schnittstelle zu schaffen, über die operationale Begriffe als Sprachelemente einer Kommandosprache höherer Ebene an den Roboter abgesetzt werden können. Damit einhergeht die Realisierung und Einbindung:

- eines Planungssystems für operationale Begriffe und
- eines Kontrollsystems zur Ausführungsüberwachung.

Die Aufgabe des Planungssystems wird zum einen darin bestehen, ausgehend von einem Zielbegriff (einem *Kommando*) rückwärts verkettend eine Sequenz operationaler Begriffe (einen *Plan*) zu bestimmen, deren Ausführung zum Zielbegriff führen kann, und zum anderen diese Sequenz durch eine Folge von Basis-Handlungsmerkmalen und handlungsorientierten Wahrnehmungsmerkmalen zu interpretieren, die das Kontrollsystem zur Ausführungsüberwachung in die Lage versetzt, den Roboter zur Durchführung bestimmter Handlungen zu veranlassen, wobei gleichzeitig eine Überprüfung der dabei registrierten Wahrnehmungen durch das Kontrollsystem erfolgt. Bei einer Abweichung der registrierten von den erwarteten Wahrnehmungen muß das Kontrollsystem die Möglichkeit besitzen, eine neue Ausgangssituation zu erreichen, von der aus das Planungssystem einen neuen Plan zur Erreichung der Zielsituation generieren kann. Planungs- und Kontrollsystem werden nach diesen Überlegungen also eng miteinander kooperierende Subsysteme des Gesamtsystems zur Robotersteuerung mit Hilfe operationaler Begriffe sein. Erste weitergehende Ideen zum Einsatz operationaler Begriffe als Planungsoperatoren werden im Kapitel 6 erörtert werden.

Die Idee im Hinblick auf die Erweiterung der Unterstützungsoberfläche ist nun, den Benutzer bzw. die Benutzerin stets über den aktuellen Systemstatus bei der Planverfolgung zu informieren. Hierzu zählt beispielsweise:

- die Möglichkeit, den aktuell verfolgten Plan anzeigen zu können,
- die Möglichkeit, stets über die aktuell durchgeführten Handlungen und die dabei registrierten Wahrnehmungen Aufschluß geben zu können,
- die Möglichkeit, daß Scheitern eines verfolgten Planes über die erwarteten und nicht registrierten Wahrnehmungsmerkmale begründen zu können, sowie
- die Möglichkeit, die durchgeführte Fahrt mit dem Erreichen der Zielsituation nachträglich beschreiben zu können.

Alle in diesem Abschnitt aufgeführten zusätzlichen Bedienungskomponenten der Unterstützungsoberfläche zur Beispielgenerierung sind lediglich als Vorschläge im Hinblick auf ihren weiteren Ausbau zu einer Benutzeroberfläche zur Steuerung und Ablaufkontrolle der Lern- und Arbeitsphase des Roboters aufzufassen. Mit dem nun folgenden Abschnitt wird das vorliegende Kapitel durch eine kurze Zusammenfassung seiner wesentlichen Inhalte abgeschlossen werden.

4.4 Zusammenfassung

Im vorstehenden Kapitel wurde eine Oberfläche zur Unterstützung der zum Lernen operationaler Begriffe notwendigen Beispielgenerierung vorgestellt. Ohne auf konkrete Algorithmen zur Beispielgenerierung einzugehen, wurde hierzu im Abschnitt 4.1 herausgearbeitet, warum eine Unterstützung der Beispielgenerierung sinnvoll ist, und wie diese Unterstützung gestaltet werden kann.

Im einzelnen wurde u.a. darauf hingewiesen, daß eine Automatisierung der Beispielgenerierung aufgrund der hierzu notwendigen Klassifikation von registrierten Objekten problematisch ist und daher zumindestens einer menschlichen Kontrolle bedarf. Ferner wurde erörtert, daß über die Festlegung der Beispielmengen eine gezielte Steuerung des sich anschließenden Lernprozesses erfolgen kann. Die Steuerung des Lernprozesses über die Beispielvorgabe kann für einzelne Begriffe der vorgeschlagenen Begriffshierarchie insbesondere dann notwendig werden, wenn der Lernprozeß auf bestimmte Zusammenhänge zwischen Handlungen und Wahrnehmungen fokussiert werden soll.

Inwieweit diese Aspekte neben weiteren im Abschnitt 4.1 diskutierten Überlegungen von der entwickelten Oberfläche abgedeckt werden, wurde im Abschnitt 4.2 deutlich. In diesem Abschnitt wurde die Oberfläche durch die Beschreibung ihrer bisher realisierten Funktionen näher charakterisiert. Neben der Möglichkeit, Beispielmengen zu edieren und unter der Kontrolle einer Komponente zur Überprüfung der syntaktischen Korrektheit einzelner Beispiele zu modifizieren, gehört zu ihren wesentlichen Merkmalen ein Visualisierungsmodul, mit dessen Hilfe eine durchgeführte Trainingsfahrt grafisch dargestellt werden kann.

Im Hinblick auf die noch ausstehende Entwicklung eines prototypischen Steuerungssystems des Karlsruher Roboters PRIAMOS, wurde im Abschnitt 4.3 erörtert, welche zusätzlichen Bedienungskomponenten notwendig sind, um die bisher realisierte Unterstützungsoberfläche zur Beispielgenerierung zu einer Benutzeroberfläche zur Abwicklung der Lern- und Arbeitsphase des Roboters auszubauen.

Kapitel 5

Lernen der Begriffshierarchie

Im Kapitel 3 wurde eine Repräsentation operationaler Begriffe vorgestellt, die sich nahtlos in die mit Beginn der vorliegenden Arbeit bereits entwickelte und im Abschnitt 2.2 vorgestellte Repräsentation handlungsorientierter Wahrnehmungsmerkmale [Morik und Rieger, 1993] einfügt. Zu diesem Zweck wurde im Kapitel 3 eine Repräsentation wahrnehmungsintegrierender Handlungsmerkmale vorgestellt, mit der eine Verbindung zwischen handlungsorientierten Wahrnehmungsmerkmalen und Basis-Handlungsmerkmalen geschaffen wird, die letztlich dazu beiträgt, daß wahrnehmungsintegrierende Handlungsmerkmale als „Bausteine" zum Aufbau operationaler Begriffe Verwendung finden können. Während die Basis-Handlungsmerkmale mit Hilfe des im Kapitel 3 erörterten Verfahrens inkrementell aus den vom Roboter aufgenommenen Sensordaten berechnet werden, erlaubt die gewählte Repräsentation eine Formulierung von Lernaufgaben für wahrnehmungsintegrierende Handlungsmerkmale und operationale Begriffe. Im vorliegenden Kapitel wird nun exemplarisch gezeigt, daß diese Lernaufgaben für logikbasierte induktive Lernverfahren handhabbar sind, und daß die im Kapitel 3 vorgestellte Repräsentation somit zum Lernen operationaler Begriffe aus Roboter-Sensordaten geeignet ist.

Im Abschnitt 5.1 werden die im vorliegenden Kapitel untersuchten Lernaufgaben für wahrnehmungsintegrierende Handlungsmerkmale und operationale Begriffe im einzelnen formuliert. Daneben enthält dieser Abschnitt eine Beschreibung des den durchgeführten Lernversuchen zugrundeliegenden Experimentdesigns. Im Abschnitt 5.2 erfolgt eine kurze Charakterisierung der im Rahmen des Projektes BLearn II eingesetzten maschinellen Lernverfahren und ihre Einordnung in das Forschungsgebiet der induktiven logischen Programmierung. Die für die Lernexperimente verwendeten Beispieldaten sowie die zu ihrer Erzeugung mit Hilfe des Karlsruher PRIAMOS-Simulationssystems durchgeführten Trainingsfahrten werden im Abschnitt 5.3 näher betrachtet. Abschnitt 5.4 enthält schließlich eine Diskussion der Ergebnisse der durchgeführten Lernexperimente, während Abschnitt 5.5 das vorliegende Kapitel mit einer kurzen Zusammenfassung abschließt.

5.1 Lernaufgaben und Experimentdesign

Das Ziel des vorliegenden Abschnittes ist es zum einen, die sich mit der im Kapitel 3 vorgeschlagenen Repräsentationshierarchie operationaler Begriffe ergebenen Lernaufgaben für die Lernphase des Roboters genauer zu charakterisieren, und zum anderen über ihre Erörterung, das den im Abschnitt 5.4 beschriebenen Lernexperimenten zugrundegelegte Experimentdesign zu motivieren. Bei der Beschreibung der Lernaufgaben werden die im Kapitel 3 eingeführten Begriffe der Repräsentationshierarchie plakativ durch Aussagen aus diesem Kapitel unter Angabe entsprechender Abschnittsverweise umschrieben. Obgleich dies für die Formulierung der Lernaufgaben nicht notwendig ist, bietet diese Vorgehensweise die Möglichkeit einige Aspekte zu rekapitulieren, die die Entwicklung der erörterten Begriffshierarchie maßgeblich leiteten. Vor dem Hintergrund dieser Überlegung läßt sich die Lernaufgabe für wahrnehmungsintegrierende Handlungsmerkmale folgendermaßen beschreiben:

LA1: Lernen wahrnehmungsintegrierender Handlungsmerkmale

- **Gegeben sind:**
  - *Beispiele für Basis-Handlungsmerkmale*, die für eine Reihe verschiedener Trainingsfahrten des Roboters aus dem vom Sensorkontroll- und Signalverarbeitungssystem zur Verfügung gestellten Daten gemäß dem in Abschnitt 3.2.2 beschriebenen Verfahren berechnet werden.
  - *Beispiele für handlungsorientierte Wahrnehmungsmerkmale*, die gemäß Abschnitt 2.2.3 komplexe Merkmale von Objekten der Umgebung während der Trainingsfahrten unter Berücksichtigung der zu ihnen relativ verlaufenden Roboterbewegungsrichtungen beschreiben.
  - *Hintergrundwissen in Form von Regeln für* `zi_Wahrnehmung/7` *und* `zp_Wahrnehmung/5`, über die gemäß Abschnitt 3.3.2 eine Unterscheidung der gegebenen handlungsorientierten Wahrnehmungsmerkmale in zeitintervall- und zeitpunktorientierte Wahrnehmungsmerkmale getroffen wird.
  - *Beispiele für wahrnehmungsintegrierende Handlungsmerkmale*, die gemäß Abschnitt 3.3.1 bestimmte Handlungen des Roboters während der Trainingsfahrten in Abhängigkeit von registrierten Wahrnehmungen beschreiben.
  - *Eine Menge von Regelschemata*, über die nach [Kietz und Wrobel, 1992] für das hier zum Lernen eingesetzte Verfahren RDT eine

Einschränkung des Hypothesenraumes vorgenommen wird[1].

- **Zu lernen sind:**
  - *Regeln, die wahrnehmungsintegrierende Handlungsmerkmale in Abhängigkeit von handlungsorientierten Wahrnehmungsmerkmalen und Basis-Handlungsmerkmalen charakterisieren.*

In der Lernphase des Roboters werden auch die handlungsorientierten Wahrnehmungsmerkmale über die Lösung von Lernaufgaben für Basis-Wahrnehmungsmerkmale, Sensor-Merkmale und Sensorgruppen-Merkmale gelernt. Da diese Lernaufgaben nicht als Bestandteil der vorliegenden Arbeit zu zählen sind, werden entsprechende Lernexperimente im Rahmen der den hier beschriebenen Versuchen zugrundeliegenden Trainingsfahrten nicht durchgeführt. Die Vorgabe der Beispiele für handlungsorientierte Wahrnehmungsmerkmale erfolgt stattdessen unter Zuhilfenahme der im Kapitel 4 erörterten Unterstützungsoberfläche zur Beispielgenerierung, indem über eine Visualisierung der Trainingsfahrten abgeschätzt wird, welche Wahrnehmungsmerkmale in welchen Zeitabschnitten der Fahrten inferiert werden können, und entsprechende Beispiele dann manuell in die Beispielmenge eingetragen werden. Auf diese Weise ist es möglich, die im Kapitel 3 beschriebenen Teile der Begriffshierarchie zur Repräsentation operationaler Begriffe unabhängig von der im Abschnitt 2.2 vorgestellten Repräsentation ihrer Wahrnehmungsseite im Hinblick auf ihre Eignung zum Einsatz maschineller Lernverfahren in der Lernphase des Roboters zu überprüfen. Für eine Formulierung der Lernaufgaben der Wahrnehmungsseite der Begriffshierarchie und die Darstellung erster Lernversuche sei daher auf [Morik und Rieger, 1993] verwiesen. Weitergehende Lernexperimente auf der Wahrnehmungsseite und eine umfassendere Erörterung findet sich darüberhinaus in [Klingspor *et al.*, 1995].

Auch die Beispiele für die wahrnehmungsintegrierenden Handlungsmerkmale und die Beispiele für die operationalen Begriffe der betrachteten Trainingsfahrten, die zur Lösung der zweiten im folgenden beschriebenen Lernaufgabe notwendig sind, werden mit Hilfe der Unterstützungsoberfläche zur Beispielgenerierung vorgegeben, da z.Z. noch keine Algorithmen zur automatischen Beispielgenerierung zur Verfügung stehen. Die Lernaufgabe für operationale Begriffe kann in folgender Weise beschrieben werden:

LA2: Lernen operationaler Begriffe

- **Gegeben sind:**
  - *Beispiele für wahrnehmungsintegrierende Handlungsmerkmale*, die gemäß Abschnitt 3.3.1 bestimmte Handlungen des Roboters

[1] Regelschemata und ihre Bedeutung für den Lernprozeß werden im Abschnitt 5.2 noch näher erörtert werden.

während verschiedener Trainingsfahrten in Abhängigkeit von registrierten Wahrnehmungen beschreiben.

- *Hintergrundwissen zur Umrechnung von Richtungsangaben*, das nach Abschnitt 3.4.2 über die dort angegebenen Relationen durch eine Menge von Fakten repräsentiert wird.
- *Beispiele für operationale Begriffe*, die gemäß Abschnitt 3.4.1 jeweils eine Reihe von Handlungen während der Trainingsfahrten in Abhängigkeit von registrierten Wahrnehmungen zusammenfassend charakterisieren.
- *Eine Menge von Regelschemata*, über die nach [Kietz und Wrobel, 1992] für das hier zum Lernen eingesetzte Verfahren RDT eine Einschränkung des Hypothesenraumes vorgenommen wird.

- **Zu lernen sind:**
  - *Regeln, die operationale Begriffe gemäß der im Abschnitt 3.4.1 vorgeschlagenen Struktur durch jeweils drei wahrnehmungsintegrierende Handlungsmerkmale charakterisieren, so daß es mit ihrer Hilfe möglich ist, Handlungen in Abhängigkeit von Wahrnehmungen zu bestimmen, auszuführen und während ihrer Durchführung zu kontrollieren, sowie sie zu einer Handlungsplanung zur Erreichung vorgegebener Ziele einzusetzen.*

Die zur Lösung dieser Lernaufgabe herangezogene Menge der Beispiele für wahrnehmungsintegrierende Handlungsmerkmale setzt sich auf der einen Seite aus denselben Beispielen zusammen, die auch schon zur Lösung der erstgenannten Aufgabe verwendet werden und wird auf der anderen Seite durch Beispiele ergänzt, die mit Hilfe der gelernten Regeln zur Charakterisierung wahrnehmungsintegrierender Handlungsmerkmale zusätzlich abgeleitet werden können. Da die Beispielmenge der wahrnehmungsintegrierenden Handlungsmerkmale für die Lernaufgabe *LA2* somit von den Lernergebnissen der Lernaufgabe *LA1* weitgehend unabhängig ist — die Anzahl der nicht zusätzlich abgeleiteten Beispiele ist zum Lernen bereits ausreichend —, bedeutet dies insbesondere, daß beide Lernaufgaben in diesem Sinne unabhängig voneinander bearbeitet werden. Alternativ zu dieser Vorgehensweise ist es auch möglich, die für die Lernaufgabe *LA2* notwendige Menge der Beispiele wahrnehmungsintegrierender Handlungsmerkmale auf die Beispiele zu beschränken, die mit Hilfe der bei der Lösung von *LA1* gewonnenen Regeln ableitbar sind, also die Bearbeitung von *LA2* gänzlich in Abhängigkeit von den Lernergebnissen von *LA1* durchzuführen.

Diese Vorgehensweise besitzt gegenüber dem gewählten Experimentdesign den Vorteil, daß sie Aussagen über die Lernbarkeit der betrachteten Teile der Repräsentationshierarchie im Ganzen ermöglicht und den Nachteil, daß im Falle unzureichender Lernbarkeit von Regeln zur Charakterisierung wahrnehmungsintegrierender Handlungsmerkmale eine Beurteilung der Lernergebnisse von *LA2* keine Rückschlüsse zuläßt, die die Eignung der im Abschnitt 3.4 vorgeschlagenen Repräsentation operationaler Begriffe bzgl. der Einsetzbarkeit maschineller Lernverfahren näher charakterisieren.

Obgleich bei den in diesem Kapitel beschriebenen Experimenten beide Lernaufgaben im obigen Sinne unabhängig voneinander betrachtet werden, können im Abschnitt 5.4 auf einem Umweg dennoch Aussagen über die Gesamtlernbarkeit der im Kapitel 3 erörterten Repräsentationshierarchie getroffen werden. Hierzu werden aus der den Experimenten zur Lernaufgabe *LA2* zugrundeliegenden Beispielmenge alle Beispiele für wahrnehmungsintegrierende Handlungsmerkmale entfernt. An ihre Stelle treten einzig die mit der Lösung von *LA1* gelernten Regeln zur Charakterisierung wahrnehmungsintegrierender Handlungsmerkmale. Eine sich anschließende Überprüfung, wieviele der vorgegebenen Beispiele für operationale Begriffe von den gelernten Regeln zu ihrer Charakterisierung noch abgeleitet werden können, bringt Aufschluß darüber, welche Regeln auch dann noch lernbar sind, wenn die Lernaufgabe *LA2* in Abhängigkeit von den Ergebnissen zur Lösung von *LA1* bearbeitet wird. Weiterhin kann diese Überprüfung als Maß für die Güte der gelernten Regelmenge im Hinblick auf ihre Anwendbarkeit in der Arbeitsphase des Roboters gelten, da sie gleichsam Auskunft darüber gibt, welche operationalen Begriffe durch sie charakterisiert werden und somit in der Arbeitsphase anwendbar sind. Abbildung 5.1 zeigt das sich aus diesen Überlegungen ergebende und den in Abschnitt 5.4 erörterten Lernversuchen zugrundeliegende Experimentdesign.

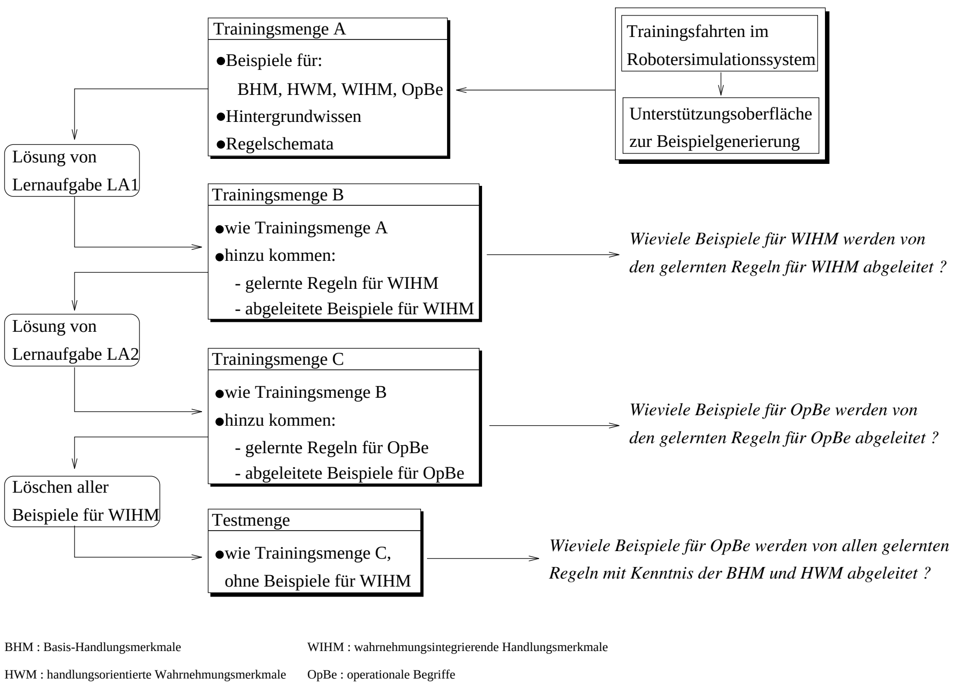


Abbildung 5.1: Experimentdesign

Die zu den Trainingsmengen B und C in Abbildung 5.1 eingetragenen und im Abschnitt 5.4 untersuchten Fragestellungen dienen der Festlegung der Qualität der beiden gelernten Regelmengen hinsichtlich ihrer Erklärungsfähigkeit der zugehörigen Beispielmengen. Für die Beurteilung dieser Fragestellungen werden beide Regelmengen unabhängig voneinander betrachtet, da die Trainingsmenge C noch die Beispiele für die wahrnehmungsintegrierenden Handlungsmerkmale enthält. Diese werden erst zur Erzeugung der in Abbildung 5.4 eingezeichneten Testmenge entfernt, anhand derer die weiter oben bereits motivierte Fragestellung untersucht werden kann[2]. Einzelheiten, die die Zusammensetzung der Trainingsmenge A betreffen, werden neben einer näheren Beschreibung der durchgeführten Traininsfahrten im Abschnitt 5.3 erörtert. Im Abschnitt 5.4 werden die auf dem in Abbildung 5.1 skizzierten Experimentdesign beruhenden Lernversuche mit den dabei erzielten Resultaten vorgestellt. Eine Charakterisierung des zum Lernen eingesetzten Verfahrens erfolgt im nun anschließenden Abschnitt.

5.2 Einordnung der eingesetzten Lernverfahren

Das mit dem vorliegenden Abschnitt verfolgte Ziel ist es, einige der Ideen der im Rahmen des Projektes BLearn II zur Lösung der im vorangegangenen Abschnitt beschriebenen Lernaufgaben eingesetzten Lernverfahren RDT [Kietz und Wrobel, 1992] und GRDT [Klingspor, 1994a] zu skizzieren. Auf diese Weise soll ein Bild über den Einsatz maschinellen Lernens innerhalb des Projektes und damit auch in dieser Arbeit vermittelt werden, das zum Verständnis der im vorliegenden Kapitel erörterten Lernexperimente notwendig ist und darüberhinaus eine vollständigere Sichtweise auf die mit dieser Arbeit verfolgte Zielsetzung erlaubt, nämlich eine Repräsentation operationaler Begriffe zum Lernen aus Roboter-Sensordaten zu entwickeln. Da die in diesem Kapitel dargestellten Lernexperimente mit Hilfe des Verfahrens RDT durchgeführt wurden und weiterhin GRDT als Weiterentwicklung von RDT prinzipiell auf den selben Ideen beruht wie RDT, wird dem Lernverfahren RDT in diesem Abschnitt besondere Beachtung geschenkt. Auf eine Erörterung von Einzelheiten des Verfahrens wird dabei jedoch verzichtet. Ebensowenig erfolgt in diesem Abschnitt eine Einführung in die Terminologie des logik-basierten induktiven Lernens. Eine Einführung in diese Thematik bietet [Morik, 1993].

Die maschinellen Lernverfahren RDT und GRDT sind hinsichtlich ihrer Arbeitsweise dem Bereich der induktiven logischen Programmierung (*ILP*) zuzuordnen, einem Forschungsgebiet, das nach [Muggleton, 1992] im Hinblick auf die dort eingesetzten Methoden und Zielsetzungen als „Schnittmenge" der Forschung im Bereich des maschinellen Lernens und der logischen Programmierung charakterisiert werden kann. Die in diesem Bereich angesiedel-

[2] Bedauerlich ist, daß es aufgrund des im Abschnitt 2.1.3 bereits angedeuteten Mangels an mittelfristig aus Karlsruhe zu bekommenden Simulationsdaten von PRIAMOS nicht möglich ist, diese Fragestellungen an einer anderen Beispielmenge als an der zum Lernen verwendeten zu untersuchen.

ten Lernverfahren lassen sich im Hinblick auf die mit ihnen verfolgte Zielsetzung in zwei grundsätzlich voneinander zu trennende Gruppen aufteilen. Auf der einen Seite handelt es sich um Verfahren, die aus einer Menge positiver und negativer Beispiele eines Begriffes und gegebenem Hintergrundwissen eine zugehörige Begriffsbeschreibung induzieren, während es auf der anderen Seite darum geht, in einer gegebenen Menge von Formeln Eigenschaften zu entdecken und zu beschreiben, die Abhängigkeiten zwischen den gegebenen Formeln charakterisieren. Diesen auf [Plotkin, 1971] und [Helft, 1989] zurückgehenden Sichtweisen der induktiven logischen Programmierung ist gemeinsam, daß sie als Repräsentationsformalismus die Prädikatenlogik erster Stufe zugrundelegen.

Es ist hier nicht der Platz, über eine Formalisierung beider Ansätze ihre Unterschiede herauszuarbeiten. Eine entsprechende Arbeit findet sich in [Lübbe, 1995]. Für eine sinnvolle Einordnung der in dieser Arbeit durchgeführten Lernexperimente in den Bereich der induktiven logischen Programmierung ist es jedoch angebracht, zumindestens auf zwei der wesentlichen Unterschiede hinzuweisen.

Während für die induzierten Begriffsbeschreibungen gefordert wird, daß aus ihnen, zusammen mit dem Hintergrundwissen, alle zugehörigen positiven Beispiele logisch gefolgert werden können, muß von den induzierten Formeln der zweiten Sichtweise nicht notwendiger Weise eine vollständige Überdeckung der entsprechenden Teilmengen der gegebenen Formelmenge erzielt werden. Weiterhin spielt hier die Unterscheidung zwischen Hintergrundwissen und negativen sowie positiven Beispielen keine Rolle, da sie gemeinsam als gegebene Formelmenge betrachtet werden. Diese zweite Sichtweise trifft auf alle im Rahmen des Projektes BLearn II durchzuführenden Lernexperimente zu, also insbesondere auch auf die in diesem Kapitel beschriebenen Versuche. RDT und GRDT werden demnach im Rahmen dieser Arbeit nicht als „Begriffslerner“ im Sinne von [Plotkin, 1971] eingesetzt[3], sondern als Lernverfahren zum Entdecken von Abhängigkeiten in gegebenen Formelmengen im Sinne der *ILP*-Sichtweise von [Helft, 1989].

Nach einer auf [Mitchell, 1982] zurückgehenden Idee werden *ILP*-Verfahren im allgemeinen als Suchprozeß in einem geordneten Raum aller in der Repräsentationssprache bildbaren Formeln (Hypothesen) zur Lösung der gestellten Lernaufgaben realisiert. Als Ordnungskriterium zwischen den Hypothesen fungiert dabei zumeist die logische Folgerung. Aufgrund ihrer Unentscheidbarkeit in der Prädikatenlogik erster Stufe, müssen sich die Verfahren jedoch bei einer syntaktischen Konstruktion der logischen Folgerung auf Ableitungsregeln beschränken, die zwar korrekt und entscheidbar aber nicht mehr vollständig sind. Zu diesen Ableitungsregeln zählt beispielsweise die θ-Subsumtion [Plotkin, 1971], auf die im weiteren Verlauf dieses Abschnittes noch einmal zurückgekommen wird. Die mit der Verwendung entsprechender Ableitungsoperatoren einhergehende Verkleinerung des Suchraumes

[3] [Lübbe, 1995] macht deutlich, daß RDT unter Einhaltung bestimmter Rahmenbedingungen auch zum Lernen von Begriffsbeschreibungen eingesetzt werden kann.

reicht jedoch im allgemeinen nicht aus, um die gestellten Lernaufgaben effizient lösen zu können. Der Repräsentationsformalismus für Beispiele, Hintergrundwissen und Hypothesen wird aus diesem Grunde für die verschiedenen *ILP*-Verfahren zusätzlich auf bestimmte Teilmengen der Prädikatenlogik reduziert, während parallel dazu angestrebt wird, den verkleinerten Hypothesenraum nicht einfach „blind", sondern mit Hilfe geeigneter Strategien zu durchsuchen, über die während des Suchprozesses möglichst große Teilräume für die Suche nach geeigneten Hypothesen ausgeschlossen werden können.

Das in die Werkbank MOBAL [Morik *et al.*, 1993] integrierte Lernverfahren RDT verwendet als Repräsentation beispielsweise eine erweiterte funktionsfreie Hornklausellogik, die auch negierte Literale zuläßt. Es ist hier nicht der Platz, MOBAL's Repräsentationsformalismus im einzelnen zu erörtern. Hierzu sei auf [Morik *et al.*, 1993] (S.27—60) und [Wrobel, 1987] verwiesen. Zum Verständnis der prinzipiellen Arbeitsweise von RDT ist es aber notwendig, den Begriff des Regelschemas näher zu charakterisieren.

Bei Regelschemata handelt es sich um Formeln mit Prädikatsvariablen anstelle von Prädikatssymbolen. Sie stellen demnach syntaktische Konstrukte (*Schablonen*) zur Beschreibung von Mengen von syntaktisch gleichen Formeln dar. Für das Lernverfahren RDT wird über die Vorgabe einer Menge von Regelschemata zu Beginn des Lernprozesses der zu durchsuchende Hypothesenraum beschrieben, d.h. insbesondere, daß nur solche Hypothesen gelernt werden können, die durch die gegebenen Regelschemata in ihrer syntaktischen Form beschrieben. Über eine Erweiterung der θ-Subsumtion von [Plotkin, 1970] auf Regelschemata [Morik *et al.*, 1993] (S.182) wird der Hypothesenraum gemäß der somit gewonnenen Generalisierungsbeziehung zwischen den Regelschemata geordnet. Das Lernverfahren GRDT unterscheidet sich im wesentlichen dahingehend von RDT, als daß hier zur Beschreibung der zum Lernen notwendigen Regelschemata kontextfreie Grammatiken eingesetzt werden, was zu einer kompakteren Darstellungsweise führt.

Neben der Beschreibung des Hypothesenraumes durch Regelschemata und seiner Strukturierung über die zwischen ihnen definierte Generalisierungsbeziehung werden verschiedene andere Verfahren eingesetzt, um den Suchraum für RDT weiter zu beschränken. Auf diese Verfahren soll hier aber nicht eingegangen werden.

Bei der Suche nach geeigneten Hypothesen versucht das Verfahren schließlich sukzessive die vorgegebenen Regelschemata mit Hilfe der durch Hintergrundwissen und Beispiele gegebenen Prädikatssymbole zu instanziieren, wobei stets gemäß der Generalisierungsbeziehung von generelleren zu spezielleren Regelschemata vorgegangen wird. Durch die Benutzervorgabe einer Liste von Prädikatssymbolen beschränkt RDT die Suche dabei auf Hypothesen, die diese Prädikatssymbole in ihrer Konklusion aufweisen. Die auf diese Weise ermittelten Hypothesen werden weiterhin dahingehend überprüft, ob sie dem sog. Akzeptanzkriterium genügen. Das Akzeptanzkriterium wird dabei vom System über die Benutzervorgabe einer Reihe von Parametern bestimmt, auf die hier nicht im einzelnen eingegangen werden kann. Eine vollständige Aufzählung der RDT-Parameter findet sich in [Sommer *et al.*, 1993]. Mit Hilfe der folgenden Definitionen aus [Morik *et al.*, 1993] (S.185) ist es jedoch möglich, die zur Durchführung der in diesem Kapitel beschriebenen Lernexperimente eingesetzten Akzeptanzkriterien verständlich zu beschreiben, ohne dabei näher auf die zu ihrer Bildung notwendigen Parametersätze eingehen zu müssen.

Sei $H = p(x_1, ..., X_m) \rightarrow q(x_1, ..., x_n)$ eine zu überprüfende Hypothese, wobei $m \geq n$ gilt, und $p(x_1, ..., X_m)$ für die Konjunktion aller Prämissen von H über den Variablen $x_1, ..., X_m$ steht. Zur Formalisierung des Akzeptanzkriteriums können u.a. folgende Definitionen aus [Morik *et al.*, 1993] (S.185) eingesetzt werden:

- $pos(H) := \{(c_1, ..., c_n) \mid p(c_1, ..., c_m) \;\&\; q(c_1, ..., c_n)\}$, die positiven Instanzen von H.
- $neg(H) := \{(c_1, ..., c_n) \mid p(c_1, ..., c_m) \;\&\; not(q(c_1, ..., c_n))\}$, die negativen Instanzen von H.
- $pred(H) := \{(c_1, ..., c_n) \mid p(c_1, ..., c_m) \;\&\; unknown(q(c_1, ..., c_n))\}$, die unbekannten Instanzen von H, d.h. Instanzen, die weder als falsch noch als wahr bekannt sind, also von H vorhergesagt werden.
- $total(H) := pos(H) \cup neg(H) \cup pred(H)$

Das konkrete Akzeptanzkriterium eines Lernexperimentes kann mit Hilfe dieser Definitionen durch einen Ausdruck charakterisiert werden, der sich neben den oben angegebenen Ausdrücken aus Konjunktionen, Disjunktionen und arithmetischen Operatoren zusammensetzt. Ein Beispiel ist: $pos \geq 50 \;\&\; \frac{neg}{pos} < 0.5$. Bei Verwendung des mit diesem Ausdruck beschriebenen Akzeptanzkriteriums zum Lernen, können beispielsweise nur Hypothesen gefunden werden, die mindestens 50 positive und mehr als zweimal soviele positive wie negative Beispiele ableiten.

5.3 Verwendete Beispieldaten

Nachdem im vorangegangenen Abschnitt das für die in diesem Kapitel beschriebenen Lernexperimente eingesetzte Lernverfahren RDT [Kietz und Wrobel, 1992] näher charakterisiert und in das Forschungsgebiet der induktiven logischen Programmierung eingeordnet wurde, und im Abschnitt 5.1 zum einen eine Erörterung des den Lernversuchen zugrundeliegenden Experimentdesigns und zum anderen eine Formulierung der zu bearbeitenden Lernaufgaben *LA1* und *LA2* stattgefunden hat, erfolgt im vorliegenden Abschnitt eine Skizzierung der vom Roboter durchgeführten Trainingsfahrten zur Erzeugung der für den Lernprozeß notwendigen Beispielmengen. Hierzu wird eine der Trainingsfahrten beispielhaft vorgestellt. Darüberhinaus wird die Zusammensetzung der den Lernversuchen zugrundeliegenden Trainingsmenge A (siehe hierzu Abschnitt 5.1, Abbildung 5.1) näher beschrieben.

Die Basis für die vorgestellten Lernexperimente zur Lösung von *LA1* und *LA2* bilden insgesamt fünf mit Hilfe des Karlsruher PRIAMOS-Simulationssystems durchgeführte Roboterfahrten. Allen Fahrten gemeinsam ist die durchfahrene Umgebung, ein leerer Raum mit einer Türöffnung. Der Roboter folgt weiterhin in allen Fahrten solange den Wänden des Raumes, bis er schließlich an der Tür vorbeifährt. Ein sich anschließendes Zurücksetzen vor die Tür, eine 90° Rotation und letztlich das Durchfahren der Tür zum Abschluß der Trainingsfahrt ist ferner in allen Fahrten enthalten. Die Fahrten unterscheiden sich hingegen im wesentlichen in drei Punkten:

- Entweder befindet sich der Roboter zu Beginn der Trainingsfahrt nahe genug an einer Wand, um ihr folgen zu können, oder er muß sich hierzu zunächst einer Wand nähern.
- Die Entfernung zwischen der Roboterposition zu Beginn der Fahrt und der Position der Tür im Raum ist für alle Trainingsfahrten verschieden. So reicht es inbesondere für einige Trainingsfahrten aus, einer Wand zu folgen, um die Tür zu finden, während der Roboter in anderen Trainingsfahrten hierzu zwei bzw. drei Wänden folgen muß.
- Für einige Trainingsfahrten sind Bodenunebenheiten simuliert, die zur Folge haben, daß der Roboter während der Fahrt trotz Einhaltung einer bestimmten Bewegungsrichtung leicht vom Kurs abdriftet (siehe hierzu Abbildung 3.4 im Abschnitt 3.2.2).

Auf der Ebene der operationalen Begriffe können diese Trainingsfahrten jeweils durch eine Auswahl der Begriffe :

```
bewege_dich_näher_zur_Wand/4
drehe_dich_vor_der_Wand/4
bewege_dich_parallel_zur_Wand_in_die_Ecke/4
drehe_dich_in_der_Ecke_zur_vorderen_Wand/4
bewege_dich_entlang_der_Tür/4
bewege_dich_vor_die_Tür/4
drehe_dich_vor_der_Tür/4
bewege_dich_durch_die_Tür/4
```

beschrieben werden. Das Ziel der durchgeführten Lernexperimente ist daher insbesondere, Regeln zur Charakterisierung dieser operationalen Begriffe aus den durchgeführten Trainingsfahrten zu lernen. Abbildung 5.2 zeigt eine der Trainingsfahrten in einer schematischen Darstellung. Diese Trainingsfahrt wird für den Lernvorgang durch folgende Beispiele operationaler Begriffe beschrieben:

1. `bewege_dich_parallel_zur_Wand_in_die_Ecke`$(t201, 1, 17, 13)$
2. `drehe_dich_in_der_Ecke_zur_vorderen_Wand`$(t201, 13, 30, 26)$
3. `bewege_dich_entlang_der_Tür`$(t201, 26, 55, 52)$
4. `bewege_dich_vor_die_Tür`$(t201, 52, 72, 69)$
5. `drehe_dich_vor_der_Tür`$(t201, 69, 83, 81)$
6. `bewege_dich_durch_die_Tür`$(t201, 81, 112, 109)$

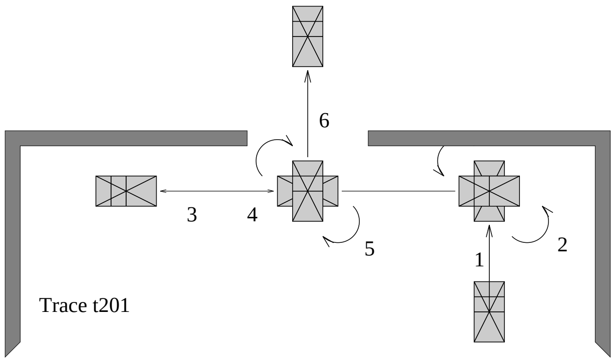


Abbildung 5.2: Beispiel für eine Trainingsfahrt

Wie bereits im Abschnitt 5.1 erläutert, werden diese Beispiele, ebenso wie die zugehörigen Beispiele für wahrnehmungsintegrierende Handlungsmerkmale und handlungsorientierte Wahrnehmungsmerkmale, mit Hilfe der im Kapitel 4 beschriebenen Oberfläche zur Unterstützung der Beispielgenerierung vorgegeben. Die zum Lernen notwendigen Beispiele für Basis-Handlungsmerkmale werden hingegen für jede Trainingsfahrt gemäß dem in Abschnitt 3.2.2 beschriebenen Verfahren berechnet. Aus wievielen Beispielen und wieviel Hintergrundwissen sich die den Lernversuchen zugrundeliegende Trainingsmenge A (siehe Abschnitt 5.1, Abbildung 5.1) zusammensetzt, zeigt Tabelle 5.1.

$\#BHM$	$\#HWM$	$\#WIHM$	$\#OpBe$
Anzahl der Beispiele			
65	49	65	30
$\#HW_1$	$\#HW_2$	$\#RS_1$	$\#RS_2$
Hintergrundwissen & Regelschemata			
6	54	6	9

Tabelle 5.1: Zusammensetzung von Trainingsmenge A

In dieser Tabelle steht $\#BHW$, $\#HWM$, $\#WIHM$ sowie $\#OpBe$ jeweils für die Beispiele der Basis-Handlungsmerkmale, der handlungsorientierten Wahrnehmungsmerkmale, der wahrnehmungsintegrierenden Handlungsmerkmale sowie der operationalen Begriffe. Die Anzahl der Regeln, die zur Beschreibung von `zi_Wahrnehmung/7` und `zp_Wahrnehmung/5` in Form von Hintergrundwissen zum Lernen von Regeln zur Charakterisierung wahrnehmungsintegrierender Handlungsmerkmale vorgegeben werden, ist in der mit HW_1 überschriebenen Spalte der zweiten Zeile der Tabelle wiedergegeben. Die Anzahl der Fakten hingegen, die beim Lernen operationaler Begriffe zur Beschreibung von Relationen zwischen Richtungsangaben benötigt werden, ist in der mit HW_2 bezeichneten Spalte der zweiten Zeile eingetragen. In den mit RS_1 und RS_2 gekennzeichneten Spalten der zweiten Zeile ist jeweils die Anzahl der Regelschemata aufgeführt, die zum Lernen wahrnehmungsintegrierender Handlungsmerkmale und operationaler Begriffe eingesetzt werden. Eine Erörterung der Ergebnisse dieser Lernversuche wird im nun folgenden Abschnitt erfolgen.

5.4 Lernergebnisse

Im vorliegenden Abschnitt werden die Lernergebnisse der dem in Abschnitt 5.1 beschriebenem Experimentdesign zugrundeliegenden Lernversuche zum Lernen der im Kapitel 3 erörterten Repräsentationshierarchie vorgestellt. Die für die Erzeugung der notwendigen Beispielmengen erforderlichen Trainingsfahrten sind im vorangegangenen Abschnitt näher charakterisiert worden, während das zum Lernen eingesetzte Verfahren RDT [Kietz und Wrobel, 1992] bereits im Abschnitt 5.2 vorgestellt wurde. Für jede der beiden im Abschnitt 5.1 formulierten Lernaufgaben *LA1* und *LA2* sind drei Lernexperimente durchgeführt worden, die sich in der Vorgabe des jeweils eingesetzten Akzeptanzkriteriums unterscheiden. Wie bereits im Abschnitt 5.2 erörtert, werden die vom Lernverfahren in Betracht gezogenen Hypothesen nur dann akzeptiert, wenn sie dem für das Lernexperiment vorgegebenem Akzeptanzkriterium genügen. Im weiteren Verlauf dieses Abschnittes werden die drei eingesetzten Kriterien mit *CC1*, *CC2* und *CC3* bezeichnet. Im einzelnen bedeutet:

CC1: $pos(H) \geq 1$, d.h., jede gelernte Regel zur Charakterisierung eines Begriffes muß mindestens eines der Eingabenbeispiele[4] des betrachteten Begriffes ableiten und darf beliebig viele Beispiele für den Begriff ableiten, die nicht Eingabebeispiele sind.

CC2: $pos(H) \geq 1$ & $pred(H) \leq 0$, d.h., jede gelernte Regel zur Charakterisierung eines Begriffes muß mindestens eines der Eingabebeispiele des betrachteten Begriffes ableiten und darf keine Beispiele für den Begriff ableiten, die nicht Eingabebeispiele sind.

CC3: $pos(H) \geq 3$ & $pos(H) \geq 0.5 * total(H)$, d.h., jede gelernte Regel zur Charakterisierung eines Begriffes muß mindestens drei Eingabebei-

[4] Im folgenden werden als Eingabebeispiele stets die Beispiele eines Begriffes bezeichnet, die explizit für den Lernvorgang vorgegeben werden.

spiele des betrachteten Begriffes ableiten, und mindestens 50 % aller von ihr abgeleiteten Beispiele sind Eingabebeispiele.

Die Wahl dieser Akzeptanzkriterien erfolgte unabhängig von den damit erzielten Lernergebnissen, d.h., daß sie nicht über eine Reihe verschiedener Lernläufe als die Kriterien bestimmt wurden, die die besten Resultate lieferten[5]. Ausschlaggebend für ihre Wahl ist hingegen die Idee gewesen, die Lernbarkeit von Begriffen der vorgestellten Repräsentation unter zwei extremen und zugleich gegensätzlichen Bedingungen — nämlich *CC1* und *CC2* — zu beleuchten. Das Akzeptanzkriterium *CC3* liegt hingegen, wenn davon abgesehen wird, daß hier $pos(H) \geq 3$ und nicht $pos(H) \geq 1$ gelten muß, zwischen den durch *CC1* und *CC2* ausgedrückten extremen Positionen. Da die Anzahl der Eingabebeispiele einiger Begriffe sehr gering ist, ist die geforderte Anzahl der mindestens von einer Regel ableitbaren Eingabebeispiele für die drei Versuchsreihen mit einem Beispiel bzw. drei Beispielen niedrig gewählt worden. Die weiter unten erläuterten Ergebnisse zeigen, daß die Wahl von $pos(H) \geq 3$ für *CC3* für einige Begriffe sogar noch zu hoch angesetzt wurde, da sie nur mit jeweils einem Beispiel in der Eingabemenge vertreten sind. Dieser Umstand erklärt auch, warum die Resultate für die Lernversuche mit *CC3* hinsichtlich ihrer Qualität nicht in etwa in der Mitte der Ergebnisse von *CC1* und *CC2* angesiedelt sind. Im Nachhinein muß die Festlegung von *CC3* in diesem Punkt daher als eher ungünstig bewertet werden.

Die Tabelle 5.2 zeigt die bei der Lösung von Lernaufgabe *LA1* erzielten Resultate. Die Regeln zur Charakterisierung der in Tabelle 5.2 namentlich aufgeführten wahrnehmungsintegrierenden Handlungsmerkmale sind von RDT für jedes der drei Lernexperimente zusammen *in weniger als einer Minute* gelernt worden[6], so daß auf die Angabe der Lernzeiten im einzelnen verzichtet wird.

Die mit Ergebniseinträgen versehenen Zeilen der Tabelle sind in fünf Blöcke zu jeweils drei Zeilen aufgeteilt. Die ersten vier Blöcke enthalten jeweils die Resultate der drei Lernläufe für das wahrnehmungsintegrierende Handlungsmerkmal mit dessen Namen sie überschrieben sind. Der mit *Summe* überschriebene Block summiert diese Resultate für jeden Lernlauf einzeln auf. Jede der Zeilen eines Blockes gibt die Ergebnisse eines Lernlaufes wieder, der durch die Angabe des verwendeten Akzeptanzkriteriums zu Beginn der Zeile bestimmt ist.

Die erste Spalte der Tabelle zeigt für jeden Lernlauf und für jedes wahrnehmungsintegrierende Handlungsmerkmal die Anzahl der zu seiner Charakterisierung gelernten Regeln. Die zweite Spalte gibt hingegen jeweils die An-

[5] Es ist durchaus möglich, aber nach Ansicht des Autors nicht legitim, über eine Reihe von Lernexperimenten, oder über die genaue Analyse eines Lernexperimentes, einen Parametersatz zu finden, der für die gegebene Beispielmenge zum bestmöglichen Lernergebnis führt. In der Regel liefern aber auf diese Weise abgestimmte Parametersätze auf anderen Datensätzen weitaus schlechtere Resultate, was den Sinn einer solchen Vorgehensweise in Frage stellt.

[6] Alle Lernläufe wurden auf einer SPARC-Station 10 mit 48 MB Arbeitsspeicher durchgeführt.

	#gelernte Regeln	#Beispiele insges.	#Eingabe-Beispiele	#abgeleitete Beispiele	%Abdeckung Eingabebeispiele
`stehen/7`					
CC1	3	54	35	54	100%
CC2	2	35	35	19	54.28%
CC3	2	35	35	19	54.28%
`p_bewegen/7`					
CC1	1	14	14	14	100%
CC2	1	14	14	14	100%
CC3	1	14	14	14	100%
`bewegen/7`					
CC1	1	20	6	20	100%
CC2	—	6	6	—	—
CC3	—	6	6	—	—
`drehen/7`					
CC1	1	10	10	10	100%
CC2	1	10	10	10	100%
CC3	1	10	10	10	100%
Summe					
CC1	**6**	**98**	**65**	**98**	**100%**
CC2	**4**	**65**	**65**	**43**	**66.15%**
CC3	**4**	**65**	**65**	**43**	**66.15%**

Tabelle 5.2: Qualität der gelernten Regeln für Handlungsmerkmale

zahl der nach Ablauf des Lernprozesses insgesamt vorhandenen Beispiele für einen Begriff an. Diese Anzahl entspricht jeweils der Summe der Anzahl der Eingabebeispiele und der Anzahl der durch die gelernten Regeln abgeleiteten Beispiele, die nicht Eingabebeispiele sind. In der dritten Spalte steht jeweils die Anzahl der Eingabebeispiele, während in der vierten Spalte jeweils die Anzahl der durch die gelernten Regeln ableitbaren Beispiele eingetragen ist. Die Frage, wieviele der Eingabebeispiele zur Menge der ableitbaren Beispiele gehören, wird für jedes wahrnehmungsintegrierende Handlungsmerkmal und für jedes Lernexperiment durch einen entsprechenden Eintrag in der fünften Spalte beantwortet. Ein Eintrag von 100 % bedeutet beispielsweise, daß die gelernten Regeln zur Charakterisierung eines Begriffes alle Eingabebeispiele des Begriffes ableiten.

Für die wahrnehmungsintegrierenden Handlungsmerkmale `p_bewegen/7` und `drehen/7` wurde in den drei Lernläufen jeweils eine Regel zu ihrer Charakterisierung gelernt. Die beiden gelernten Regeln in jedem Lernlauf entsprechen dabei genau den im Abschnitt 3.3.1 angegebenen Beispielregeln zur Beschreibung von `p_bewegen/7` und `drehen/7`. Neben einer jeweiligen Abdeckung der Eingabebeispiele von 100 % ist für sie auffällig, daß sie keine zusätzlichen Beispiele ableiten. Die beiden gelernten Regeln zur Beschreibung des wahrnehmungsintegrierenden Handlungsmerkmals `stehen/7` in den Lernläufen mit *CC2* und *CC3* sind für beide Lernläufe identisch und sind auch in dem Lernexperiment mit *CC1* gelernt worden. Nicht gelernt werden konnte mit *CC2* und *CC3* hingegen die im Lernlauf mit *CC1* gelernte Regel:

```
zp_Wahrnehmung(X,Y,Z,U,V) &
bleibe_stehen(X,Y,V1)
→ stehen(X,Y,V1,Z,U,V,Z).
```

Mit ihrer Hilfe können 35 Beispiele für `stehen/7` abgeleitet werden, von denen aber nur 16 Eingabebeispiele sind. Die von ihr demnach zusätzlich abgeleiteten 19 Beispiele haben zur Folge, daß sie nicht den Akzeptanzkriterien *CC2* und *CC3* genügen können. Erwähnenswert ist noch, daß neben der oben angegebenen Regel auch eine der beiden anderen gelernten Regeln im Abschnitt 3.3.1 bereits vorgestellt wurde. Bei der dritten gelernten Regel handelt es sich um einen Spezialfall. Sie entspricht in ihrer Form der oben angegebenen Regel mit dem Unterschied, daß bei ihr die Variable für den ersten Zeitpunkt durch die Konstante *1* und die Variable für die letzte Wahrnehmung durch die Konstante *unbekannt* gebunden ist. Diese Regel beschreibt demnach ein Stehenbleiben zu Beginn einer Trainingsfahrt, wobei eine letzte Wahrnehmung zum Zeitpunkt *1* noch nicht definiert ist.

Die im Lernexperiment mit Akzeptanzkriterium *CC1* gelernte Regel zur Charakterisierung des wahrnehmungsintegrierenden Handlungsmerkmals `bewegen/7` entspricht ebenfalls der im Abschnitt 3.3.1 vorgestellten Beispielregel. Da sie neben den sechs Eingabebeispielen 14 weitere Beispiele für `bewegen/7` ableitet, konnte diese Regel nach der Definition der Akzeptanzkriterien *CC2* und *CC3* in den zugehörigen Lernversuchen nicht gelernt werden.

Der Vergleich der gelernten Regelmengen zur Charakterisierung der wahrnehmungsintegrierenden Handlungsmerkmale für die drei untersuchten Lernexperimente zeigt, daß für den Lernlauf mit dem Akzeptanzkriterium *CC1* eine zusätzliche Regel für `stehen/7` sowie eine Regel für `bewegen/7` gelernt werden konnten, die beide das Akzeptanzkriterium *CC2* bzw. das Kriterium *CC3* nicht erfüllen. Der Grund hierfür ist, daß diese Regeln jeweils mehr Beispiele ableiten als Eingabebeispiele vorhanden sind. Es ist zu erwarten, daß dieses Problem bei einer größeren Menge von Eingabebeispielen nicht auftreten wird. Diese Vermutung wird dadurch gestärkt, daß die im folgenden erörterten Lernergebnisse zur Lösung von Lernaufgabe *LA2* deutlich machen, daß die zusätzlich abgeleiteten Beispiele für wahrnehmungsintegrierende Handlungsmerkmale insofern keinen nachteiligen Einfluß auf das Lernen von Regeln zur Charakterisierung operationaler Begriffe ausüben, als daß diese Regeln neben den Eingabebeispielen keine weiteren Beispiele für operationale Begriffe ableiten[7]. In diesem Umstand zeigt sich insbesondere, daß die zusätzlich abgeleiteten Beispiele für wahrnehmungsintegrierende

[7] Nach dem in Abschnitt 5.1 beschriebenem Experimentdesign, enthält die für die Lösung von Lernaufgabe *LA2* herangezogene Trainingsmenge B neben den Eingabebeispielen auch die mit den gelernten Regeln für wahrnehmungsintegrierende Handlungsmerkmale zusätzlich ableitbaren Beispiele.

Handlungsmerkmale durchaus sinnvolle Beschreibungen von während der Trainingsfahrten aufgetretenen Situationen darstellen. Eine größere Menge von Eingabebeispielen könnte demnach nicht nur mit Hilfe einer größeren Anzahl von Trainingsfahrten zur Verfügung gestellt werden, sondern auch dadurch, daß dieselben Situationen während einer Fahrt durch verschiedene wahrnehmungsintegrierende Handlungsmerkmale charakterisiert werden. So kann beispielsweise eine durch `bewegen/7` beschriebene Situation in vielen Fällen auch durch `p_bewegen/7` und umgekehrt dargestellt werden. Unter Berücksichtigung dieser Überlegungen ist nicht nur die für den Lernlauf mit *CC1* erzielte Gesamtabdeckung der Eingabebeispiele von 100 % als hervorragendes Ergebnis zu bewerten, sondern ebenso die jeweils mit *CC2* und *CC3* erzielte Abdeckung von 66.15 %.

Die Tabelle 5.3 zeigt nun die bei der Lösung von Lernaufgabe *LA2* erzielten Resultate. Die Regeln zur Charakterisierung der in Tabelle 5.3 namentlich aufgeführten operationalen Begriffe sind von RDT für jedes der drei durchgeführten Lernexperimente zusammen *in weniger als drei Minuten* gelernt worden, so daß hier, ebenso wie schon für die wahrnehmungsintegrierenden Handlungsmerkmale, auf eine Angabe der Lernzeiten im einzelnen verzichtet wird. Tabelle 5.3 entspricht in ihrem Aufbau Tabelle 5.2.

Bei den Lernexperimenten zum Lernen von Regeln zur Charakterisierung operationaler Begriffe ist der Einfluß der Größe der Beispielmenge geringer, als bei den wahrnehmungsintegrierenden Handlungsmerkmalen. So konnte mit *CC3* immerhin noch eine Gesamtabdeckung der Eingabebeispiele von 90 % erzielt werden. Eine Analyse der mit diesem Lernexperiment erzielten Resultate zeigt, daß die nicht 100 %-tige Abdeckung auf eine für das Akzeptanzkriterium *CC3* ungünstig zusammengesetzte Beispielmenge zurückzuführen ist.

So sind in den durchgeführten Trainingsfahrten die operationalen Begriffe `bewege_dich_näher_zur_Wand/4` und `drehe_dich_vor_der_Wand/4` nur einmal vertreten, d.h., daß die Menge der Eingabebeispiele für diese Begriffe nur jeweils ein Beispiel enthält. Regeln zur Charakterisierung dieser Begriffe konnten demnach bereits nach der Definition des vorgegebenen Akzeptanzkriteriums *CC3* nicht gelernt werden. Für den operationalen Begriff `bewege_dich_parallel_zur_Wand_in_die_Ecke/4` ist ebenfalls nur ein Beispiel vorhanden, bei dem der Roboter vor der Vorwärtsbewegung die Wahrnehmung `parallel_in_einer_Ecke/4` registriert. Dieses Beispiel kann nicht durch die mit *CC3* gelernte Regel:

```
stehen(X,Y,Z,vor_einer_Wand,U,V,V1) &
hw_WRichtung_nach_Bewegung(U,V2,V3) &
p_bewegen(X,Z,V4,V5,V2,entlang_einer_Wand,V3) &
hw_WRichtung_einer_Ecke1(V3,V2,V6) &
stehen(X,V4,V7,parallel_in_einer_Ecke,V6,V,entlang_einer_Wand)
→ bewege_dich_parallel_zur_Wand_in_die_Ecke(X,Y,V7,V4)
```

abgeleitet werden, da diese Regel eine Vorwärtsbewegung entlang einer in einer bestimmten Richtung wahrgenommenen Wand zu einer Ecke des Raumes

	#gelernte Regeln	#Beispiele insges.	#Eingabe-Beispiele	#abgeleitete Beispiele	%Abdeckung Eingabebeispiele
`bewege_dich_durch_die_Tür/4`					
CC1	4	5	5	5	100%
CC2	1	5	5	5	100%
CC3	1	5	5	5	100%
`drehe_dich_vor_der_Tür/4`					
CC1	2	5	5	5	100%
CC2	2	5	5	5	100%
CC3	2	5	5	5	100%
`bewege_dich_vor_die_Tür/4`					
CC1	3	5	5	5	100%
CC2	2	5	5	5	100%
CC3	2	5	5	5	100%
`bewege_dich_entlang_der_Tür/4`					
CC1	4	5	5	5	100%
CC2	2	5	5	5	100%
CC3	2	5	5	5	100%
`drehe_dich_in_der_Ecke_zur_vorderen_Wand/4`					
CC1	4	4	4	4	100%
CC2	4	4	4	4	100%
CC3	4	4	4	4	100%
`bewege_dich_parallel_zur_Wand_in_die_Ecke/4`					
CC1	3	4	4	4	100%
CC2	2	4	4	4	100%
CC3	1	4	4	3	75%
`drehe_dich_vor_der_Wand/4`					
CC1	4	1	1	1	100%
CC2	4	1	1	1	100%
CC3	—	1	1	—	—
`bewege_dich_näher_zur_Wand/4`					
CC1	2	1	1	1	100%
CC2	2	1	1	1	100%
CC3	—	1	1	—	—
Summe					
CC1	**26**	**30**	**30**	**30**	**100%**
CC2	**19**	**30**	**30**	**30**	**100%**
CC3	**12**	**30**	**30**	**27**	**90%**

Tabelle 5.3: Qualität der gelernten Regeln für operationale Begriffe

charakterisiert.

Ein Vergleich der gelernten Regelmengen zur Charakterisierung operationaler Begriffe für die drei untersuchten Lernexperimente zeigt, daß alle im Lernlauf mit dem Akzeptanzkriterium *CC3* gelernten Regeln auch mit *CC2* und alle mit *CC2* gelernten Regeln auch mit *CC1* gelernt werden konnten. Alle Regelmengen enthalten dabei Regeln, deren Entfernung aus der jeweiligen Regelmenge zu keiner Abnahme der Gesamtabdeckung der Eingabebeispiele führt. Die vergleichsweise große Anzahl gelernter Regeln mit `CC1` — 26 gegenüber 19 mit *CC2* — kann dadurch erklärt werden, daß die für das Lernexperiment mit *CC1* zugrundegelegte Beispielmenge gegenüber den Versuchen mit *CC2* und *CC3* zusätzliche Beispiele für wahrnehmungsintegrierende Handlungsmerkmale enthält. Wie bereits bei der Erörterung der Lernresultate zur Lernaufgabe *LA1* angesprochen, sind diese zusätzlichen Beispiele auf die mit *CC1* gegenüber dem Lernexperiment mit *CC3* zusätzlich gelernten Regeln zur Charakterisierung von `p_bewegen/7` und `stehen/7` bzw. auf die gegenüber *CC2* zusätzlich gelernte Regel zur Charakterisierung von `stehen/7` zurückzuführen. Da keine der gelernten Regeln Beispiele ableitet, die nicht Eingabebeispiele sind, sind die Resultate der Lernexperimente mit *CC1* und *CC2* optimale Ergebnisse für die gegebene Beispielmenge. Bei einer größeren Beispielmenge ist nach den vorgetragenen Betrachtungen sicher zu erwarten, daß auch mit *CC3* eine 100 %-tige Abdeckung der Eingabebeispiele erreicht werden kann.

Wie im Abschnitt 5.1 beschrieben, ist die für die Bearbeitung von Lernaufgabe *LA2* verwendete Beispielmenge der wahrnehmungsintegrierenden Handlungsmerkmale nach dem zugrundegelegten Experimentdesign weitgehend unabhängig von den Lernergebnissen für die Lernaufgabe *LA1*. Die bisher vorgestellten Resultate erlauben daher noch keine Aussage darüber, inwieweit die gelernten Regelmengen für die wahrnehmungsintegrierenden Handlungsmerkmale und operationalen Begriffe gemeinsam in der Arbeitsphase des Roboters anwendbar sind. Die im Abschnitt 5.1 hierzu gestellte Frage, wieviele der Eingabebeispiele für operationale Begriffe von *allen* gelernten Regeln unter Verwendung des Hintergrundwissens mit alleiniger Kenntnis der Eingabebeispiele für die Basis-Handlungsmerkmale und die handlungsorientierten Wahrnehmungsmerkmale abgeleitet werden können, kann mit Hilfe von Tabelle 5.4 beantwortet werden.

	#gelernte Regeln	#Beispiele insges.	#Eingabe-Beispiele	#abgeleitete Beispiele	%Abdeckung Eingabebeispiele
operationale Begriffe					
CC1	26	30	30	30	100%
CC2	19	30	30	4	13.33%
CC3	12	30	30	4	13.33%

Tabelle 5.4: Performanz der gelernten Regeln für operationale Begriffe

Die geringe Abdeckung der Eingabebeispiele für die operationalen Begriffe bei den Lernexperimenten mit den Akzeptanzkriterien *CC2* und *CC3* von nur 13.33 % ist einzig auf die oben angesprochenen, nicht gelernten Regeln für `stehen/7` und `bewegen/7` zurückzuführen. Da insbesondere das wahrnehmungsintegrierende Handlungsmerkmal `stehen/7` für die Charakterisierung aller operationalen Begriffe von grundlegender Bedeutung ist, muß für eine ausreichende Anwendbarkeit der gelernten Regelmengen in der Arbeitsphase des Roboters stets gewährleistet sein, daß sie möglichst viele Regeln zu seiner Beschreibung enthalten.

5.5 Zusammenfassung

Im vorstehenden Kapitel wurde für die im Kapitel 3 vorgestellte Repräsentationshierarchie gezeigt, daß die an der Spitze dieser Repräsentation stehenden operationalen Begriffe im Rahmen der im Abschnitt 3.1 erläuterten Lernphase des Roboters gelernt werden können. Hierzu erfolgte im Abschnitt 5.1 eine Formulierung von Lernaufgaben für wahrnehmungsintegrierende Handlungsmerkmale und operationale Begriffe. Eine Erörterung dieser Aufgaben führte zu einem Experimentdesign für eine Reihe von Lernläufen, mit deren Hilfe die Frage nach der Lernbarkeit operationaler Begriffe bejaht werden konnte.

Im einzelnen wurde im Abschnitt 5.2 das für die Durchführung der Lernläufe verwendete maschinelle Lernverfahren RDT [Kietz und Wrobel, 1992] näher charakterisiert. Auf eine Darlegung von Einzelheiten wurde dabei zugunsten einer allgemeinverständlich gehaltenen Skizzierung der prinzipiellen Arbeitsweise des Verfahrens und seiner Einordnung in das Forschungsgebiet der induktiven logischen Programmierung verzichtet. Das Thema von Abschnitt 5.3 waren die mit Hilfe des Karlsruher PRIAMOS-Simulationssystems durchgeführten Trainingsfahrten des Roboters. Sie bildeten die Basis zur Erzeugung der für den Lernprozeß notwendigen Beispielmenge, deren Bildung mit Hilfe der in Kapitel 4 vorgestellten Oberfläche zur Unterstützung der Beispielgenerierung erfolgte. Neben einer allgemeinen Beschreibung der durchgeführten Trainingsfahrten enthält Abschnitt 5.3 die exemplarische Vorstellung einer Trainingsfahrt und eine Erörterung der Zusammensetzung der Beispielmenge. Im Abschnitt 5.4 wurden schließlich die Ergebnisse der durchgeführten Lernexperimente diskutiert.

Über eine ausführliche Analyse dieser Ergebnisse konnte der Einfluß der Größe der Beispielmenge auf die erzielten Resultate im einzelnen nachvollzogen werden. Es zeigte sich beispielsweise, daß die unzureichende Anzahl an vorhandenen Beispielen für einige operationale Begriffe dafür verantwortlich war, daß mit Hilfe einer der gelernten Regelmengen zur Charakterisierung operationaler Begriffe lediglich eine 90 %-tige Abdeckung der Eingabebeispiele erreicht wurde. Für die beiden anderen durchgeführten Lernversuche lag die Abdeckung der zugehörigen Regelmengen jedoch bei 100 %. Eine entsprechende Auswertung der gelernten Regelmengen zur Charakterisierung wahrnehmungsintegrierender Handlungsmerkmale ergab Abdeckungen von 100 % und zweimal 66 % der Eingabebeispiele.

Zur Bewertung der Anwendbarkeit der gelernten Regelmengen in der Arbeitsphase des Roboters wurde ferner untersucht, wieviele der aus den Trainingsfahrten bekannten Beispiele für operationale Begriffe von allen gelernten Regeln mit Kenntnis der Basis-Handlungsmerkmale und der handlungsorientierten Wahrnehmungsmerkmale abgeleitet werden konnten. Die Ergebnisse dieser Untersuchung konnten mit 100 % und zweimal 13 % für die durchgeführten Lernexperimente beziffert werden, wobei die nur 13 %-tigen Abdeckungen wiederum auf die geringe Größe der Beispielmenge zurückgeführt werden konnten.

Die in diesem Kapitel vorgetragenen Ergebnisse zeigen demnach, daß die im Kapitel 3 vorgestellte Repräsentation operationaler Begriffe zum Lernen aus Roboter-Sensordaten geeignet ist, und daß der Anwendbarkeit operationaler Begriffe in der Arbeitsphase des Roboters im Prinzip nichts im Wege steht.

Kapitel 6

Perspektiven für die Handlungsplanung

Das Ziel des vorliegenden Kapitels ist es, einige Überlegungen hinsichtlich des noch zu entwickelnden Planungssystems für operationale Begriffe anzustellen. Im wesentlichen basieren die folgenden Ausführungen auf der Arbeit von [Klingspor und Sklorz, 1995]. Auf Ideen, die im besonderen auf den ersten Autor zurückgehen, wird im Text ausdrücklich hingewiesen.

Operatoren zur Handlungsplanung beschreiben im klassischen Sinne und vereinfachend dargestellt Situationen, in denen sie anwendbar sind und Situationen, in die ihre erfolgreiche Anwendung mündet. Über die in diesem Sinne Anwendbarkeits- und Nachfolgesituationen ist eine Verkettung der Operatoren zu einem Plan möglich, dessen Ausführung durch die Anwendung der den Plan bildenden Operatoren — in der Reihenfolge wie sie im Plan auftreten — zu einer gewünschten Zielsituation führt[1].

Im Planungssystem STRIPS [Fikes und Nilsson, 1971] werden Operatoren beispielsweise durch ein Tripel $< V, D, A >$ formalisiert [Biundo *et al.*, 1993] (S.793). Hierbei beschreibt V die Bedingungen, die gelten müssen, damit der betrachtete Operator angewendet werden darf. Weiterhin steht D für eine Liste von Prädikaten, die nach Anwendung des betrachteten Operators aus dem zugrundeliegenden Weltmodell gelöscht werden, während A die Prädikate enthält, die nach Ausführung des betrachteten Operators dem Weltmodell hinzugefügt werden. Die Anwendbarkeitssituation wird demnach durch V und die Nachfolgesituation durch D und A charakterisiert.

Wird die Struktur der STRIPS-Operatoren von einem abstrakten Standpunkt aus mit der im Abschnitt 3.4.1 angegebenen Struktur operationaler Begriffe verglichen, so korrespondiert unter diesem Blickwinkel die durch das erste wahrnehmungsintegrierende Handlungsmerkmal `stehen/7` charakterisierte Anwendbarkeitssituation eines operationalen Begriffes mit V, während seine durch das zweite wahrnehmungsintegrierende Handlungsmerkmal `stehen/7`

[1] Für eine umfassende Einführung in das Forschungsgebiet *Planen* sei auf [Biundo *et al.*, 1993] verwiesen.

beschriebene Verifikationssituation mit D und A korrespondiert. Die für einen operationalen Begriff durch ein weiteres wahrnehmungsintegrierendes Handlungsmerkmal beschriebene *Handlung mit zugehöhriger Wahrnehmung* findet bei dieser abstrakten Betrachtungsweise kein Pendant in der Struktur der STRIPS-Operatoren. Der Unterschied zwischen operationalen Begriffen und STRIPS-Operatoren, der sich aus diesen Betrachtungen ergibt, wird von [Klingspor, 1994b] in etwa in der folgenden Weise beschrieben.

STRIPS-Operatoren beschreiben die Änderungen des Weltzustandes, die sich bei der Ausführung der zugehörigen Handlungen einstellen. Sie beschreiben jedoch nicht die Handlungen selbst oder die Wahrnehmungen, die während der Ausführungen der Handlungen zu erwarten sind. Die Verwendung von Operatoren durch einen realen Roboter erfordert daher die Bereitstellung von zusätzlichen Informationen. Im Gegensatz dazu enthalten operationale Begriffe alle zur Handlungsplanung notwendigen Informationen über durchzuführende Handlungen und dabei zu registrierende Wahrnehmungen. Der Einsatz operationaler Begriffe für die Robotersteuerung ermöglicht daher im besonderen im Gegensatz zum Einsatz von Operatoren [Fikes *et al.*, 1972] eine Klassifikation von Handlungen und Wahrnehmungen. Ein Roboter, der bei einer Raumerkundung zum Beispiel durch Zufall durch eine Tür fährt, erkennt die Tür als solche [Klingspor, 1994b].

Die Aussicht operationale Begriffe über ihre Anwendbarkeits- und Verifikationssituationen zu Plänen zu verketten, ermöglicht die Entwicklung eines im Prinzip einfachen vorwärts- oder rückwärtsplanenden Systems. Beispielsweise könnte ein rückwärtsplanendes Verfahren grob dargestellt wie folgt arbeiten.

Ausgehend von einem operationalen Begriff, dessen Verifikationssituation der Zielsituation entspricht, werden weitere operationale Begriffe ermittelt, deren Verifikationssituationen der Anwendbarkeitssituation des Ausgangsbegriffs entsprechen. Diese operationalen Begriffe stellen alle möglichen Vorgänger im zu generierenden Plan dar, deren Ausführung mit anschließender Ausführung des Ausgangsbegriffes in der Zielsituation mündet. Werden diese Begriffe als neue Ausgangsbegriffe aufgefaßt, und wird das beschriebene Verfahren solange rekursiv wiederholt, bis ein Begriff ermittelt wird, dessen Anwendbarkeitssituation der aktuellen Situation entspricht, so ist eine Sequenz operationaler Begriffe gefunden, deren Ausführung die aktuelle Situation in die Zielsituation überführen kann, also ein *Plan*.

Der auf die beschriebene Weise ermittelte Plan ist natürlich nur ein möglicher Plan von vielen Plänen, die zur Zielsituation führen können. Ob gerade dieser oder ein anderer Plan zur Zielsituation führen kann, hängt von den Gegebenheiten der Umgebung ab, in der sich der Roboter bewegt. Weichen diese Gegebenheiten von den implizit im ermittelten Plan enthaltenen Annahmen über die Umgebung ab, so scheitert der Plan bei seiner Ausführung. Um diese Unwägbarkeiten besser handhaben zu können, sollte daher ein noch zu entwickelndes Planungssystem für operationale Begriffe mehr leisten können, als das beschriebene Verfahren. *Volker Klingspor* diskutiert hierzu zwei Ideen in [Klingspor und Sklorz, 1995], die im folgenden skizziert werden.

- Ermittelt ein Planungssystem für operationale Begriffe für die gegebene Problemstellung nicht nur einen Plan, sondern eine Menge von möglichen Plänen, so können diese Pläne durch einen gerichteten Graphen repräsentiert werden. Hierzu entsprechen die Knoten im Graphen operationalen Begriffen. Eine gerichtete Kante zwischen zwei Knoten beschreibt dann eine unmittelbare Vorgänger/Nachfolger-Relation zwischen den zugehörigen Begriffen in einem möglichen Plan. Wird ein operationaler Begriff nur durch einen Knoten im Graphen repräsentiert, so wird der Graph Zyklen enthalten, da mögliche Pläne Sequenzen gleicher operationaler Begriffe enthalten können, und ein möglicher Plan denselben operationalen Begriff mehrfach aufweisen kann. Abhängig von den während der Durchführung von Handlungen registrierten Wahrnehmungen kann an jedem Knoten entschieden werden, auf welchem gerichteten Weg der Graph durchlaufen wird, und damit welcher der repräsentierten möglichen Pläne verfolgt wird.

Dieser Ansatz macht die Planverfolgung und -ausführung etwas robuster gegen unvorhergesehene Gegebenheiten der Umgebung. Da jedoch immer nur eine kleine Menge von möglichen Plänen im Graphen repräsentiert werden kann, reicht diese Vorgehensweise alleine noch nicht aus, um Unwägbarkeiten besser handhaben zu können. *Volker Klingspor* macht daher in [Klingspor und Sklorz, 1995] noch einen zweiten Vorschlag.

- Hiernach könnte es das Ziel sein, immer nur den kürzesten möglichen Plan zu generieren, und diesen solange wie möglich durch die Ausführung der zugehörigen operationalen Begriffe zu verfolgen. Scheitert der Plan in einer bestimmten Situation, so wird ausgehend von dieser Situation ein neuer kürzester Plan ermittelt.

Dieser inkrementelle Ansatz für ein Planungssystem verleiht dem Robotersystem die Möglichkeit, auf alle unerwartet auftretenden Ereignisse reagieren zu können[2]. Diese Fähigkeit wird jedoch mit einem unakzeptablen Zeitaufwand erkauft. Im schlimmsten Fall müßte gar das Planungssytem nach jedem durchgeführtem Schritt eines möglichen Planes einen komplett neuen Plan berechnen.

Die derzeitigen Aktivitäten bei der Entwicklung eines Planungssystems für operationale Begriffe konzentrieren sich daher darauf, eine Verbindung zwischen den beiden skizzierten Ansätzen herzustellen. Mengen von möglichen Plänen könnten in der beschriebenen Weise durch Graphen repräsentiert und solange mit Hilfe der Graphrepräsentation verfolgt werden, bis eine Neuplanung erforderlich ist. Einmal berechnete Graphen könnten dabei für eine spätere Wiederverwendung gespeichert werden.

[2] Hierbei wird vorausgesetzt, daß für jede Situation mindestens ein operationaler Begriff zur Anwendung kommen kann.

Unter der Annahme, daß ein bestimmter Plan aus operationalen Begriffen verfolgt werden soll, stellt sich die Frage, auf welche Weise dies mit möglichst geringem Zeitaufwand während der Ausführung der zugehörigen Handlungen möglich ist. Der zunächst naheliegende Gedanke, den Plan während der Planverfolgung durch den Roboter schrittweise durch einen Interpreter für operationale Begriffe auswerten zu lassen, ist mit einem hohen Aufwand während der Ausführung der operationalen Begriffe verbunden. Im schlimmsten Fall müßte nach Ausführung eines Begriffes eine kurze Pause im Bewegungsablauf des Roboters eingelegt werden, um den nächsten operationalen Begriff im Plan auf ein Basis-Handlungsmerkmal und zugehörige handlungsorientierte Wahrnehmungsmerkmale zurückzuführen.

Eine andere Idee ist es, den Plan a priori zu interpretieren, also noch vor seiner Ausführung durch den Roboter. Ein Plan bestehend aus einer Folge operationaler Begriffe wird hierbei durch einen Interpreter auf Sequenzen von Handlungs-/Wahrnehmungs-Tupeln bzw. Tripeln zurückgeführt.

Jedes dieser Tupel bzw. Tripel enthält ein Basis-Handlungsmerkmal und ein bzw. zwei handlungsorientierte Wahrnehmungsmerkmale. Die Zeitpunkte in den Merkmalen sind dabei uninstantiiert. Läßt sich bei Ausführung eines Basis-Handlungsmerkmals durch den Roboter über eine Vorwärtsinferenz der bei der Handlungsausführung aufgenommenen Sensormeßwerte eine Instantiierung der demselben Tupel (Tripel) zugehörigen Wahrnehmungsmerkmale berechnen, so entspricht dies der Erfüllung eines wahrnehmungsintegrierenden Handlungsmerkmals eines operationalen Begriffes des Planes. Jeder operationale Begriff des Planes setzt sich nach der im Abschnitt 3.4.1 angegebenen Struktur aus drei wahrnehmungsintegrierenden Handlungsmerkmalen zusammen und läßt sich als abgrenzbarer Teilplan des gegebenen Planes auffassen. Die Erfüllung eines operationalen Begriffes durch die Erfüllung seiner wahrnehmungsintegrierenden Handlungsmerkmale verifiziert den Gesamtplan bis zu der Stelle, an der sich der operationale in der Folge operationaler Begriffe findet. Die Erfüllung sämtlicher operationaler Begriffe des Planes in der beschriebenen Weise entspricht der Erfüllung des Gesamtplanes bei gleichzeitiger Ausführung desselben.

Die in diesem Ansatz verborgene Schwierigkeit liegt in der Feststellung, daß es für jeden Plan aus operationalen Begriffen verschiedene Sequenzen von Handlungs-/Wahrnehmungs-Tupeln bzw. Tripeln (im folgenden kurz: *Planinstanzen*) gibt, deren Ausführung im obigen Sinne den Plan erfüllen. Hierfür gibt es zwei Gründe:

- Einzelne Begriffe der Repräsentationshierarchie können u.U. durch verschiedene Regeln beschrieben werden.
- Einzelne Regeln können im allg. mit verschiedenen Instantiierungen angewendet werden.

Ein Interpreter der beschriebenen Art ist prinzipiell über *Backtracking* in der Lage, sämtliche Planinstanzen eines gegebenen Planes zu berechnen[3]. Da es sich dabei um eine große Anzahl von Planinstanzen handeln kann, wird es bei einer Verfolgung dieser Idee zweckdienlich sein, diese Berechnung auf Teilpläne des Gesamtplanes zu beschränken. Die Auswahl einer Planinstanz zur Verfolgung des Teilplanes kann dann möglicherweise wieder als Planungsaufgabe aufgefaßt werden. Zur Illustration der vorgetragenen Idee wird dieses Kapitel mit einem Beispiel beendet.

Angenommen, der Roboter steht mit einer seiner schmalen Seiten vor einer Wand und bekommt den Auftrag sich in eine Ecke zu stellen. Eine möglicher Plan zur Erfüllung dieses Auftrages ist sich der Wand zu nähern, sich vor ihr zu drehen und sich parallel an ihr entlang zu bewegen, bis schließlich eine Ecke des Raumes erreicht wird. Mit Hilfe operationaler Begriffe läßt sich dieser Plan folgendermaßen beschreiben:

```
bewege_dich_näher_zur_Wand(...) &
drehe_dich_vor_der_Wand(...) &
bewege_dich_parallel_zur_Wand_in_die_Ecke(...)
→ plan(...)
```

Prinzipiell sind für diesen Plan vier mögliche Planinstanzen[4] denkbar, die auf die folgende Weise näher charakterisiert werden können:

1. *auf die Wand zu bewegen, links herumdrehen, rückwärts entlang der Wand bis zur Ecke bewegen*

2. *auf die Wand zu bewegen, links herumdrehen, vorwärts entlang der Wand bis zur Ecke bewegen*

3. *auf die Wand zu bewegen, rechts herumdrehen, rückwärts entlang der Wand bis zur Ecke bewegen*

4. *auf die Wand zu bewegen, rechts herumdrehen, vorwärts entlang der Wand bis zur Ecke bewegen*

Wie diese Planinstanzen in Form von Sequenzen aus Handlungs-/Wahrnehmungs-Tupeln bzw. Tripeln aussehen, zeigt Abbildung 6.1 beispielhaft für die zweite Planinstanz. Jeder operationale Begriff des Planes (*Begriff 1, Begriff 2, Begriff 3*) korrespondiert dabei mit drei Tupeln bzw. Tripeln. Die Darstellung zeigt, wie sich die jeweiligen Anwendbarkeits- und Verifikationssituationen der Begriffe überdecken. Die Verbindung zwischen

[3] Zu Experimentierzwecken ist ein entsprechendes Programm vom Autor bereits implementiert worden. Da es sich dabei jedoch bisher lediglich um eine unausgereifte Studie handelt, wird in dieser Arbeit auf eine nähere Darstellung verzichtet.

[4] Werden die möglichen Geschwindigkeiten für Drehungen und Vorwärtsbewegungen nicht als fest vorgegeben angenommen, so sind es sogar noch mehr. Für das Beispiel werden die Geschwindigkeiten der Einfachheit halber als gegeben angenommen.

Handlungen und Wahrnehmungen ist über die zu instantiierenden Zeitpunkte gewährleistet. Die richtige Instantiierung zur Ausführungszeit festzulegen, ist eine der Aufgaben der noch zu entwickelnden Ausführungsüberwachungskomponente für operationale Begriffe. Die bei der Vorwärtsbewegung auf die Wand zurückzulegende Distanz könnte mit Hilfe einer übergeordneten Strategie bestimmt werden, oder aber durch die Einführung eines neuen handlungsorientierten Wahrnehmungsmerkmals (z.B. `nah_vor_einer_Wand`) gesteuert werden.

Anfangszustand:	`bleibe_stehen`$(t201, 1, D)$ `vor_einer_Wand`$(t201, 1, vorne, schmale\_Seite)$
Begriff 1:	`bewege_dich`$(t201, D, C, langsame\_Bewegung, vorne)$ `vor_einer_Wand`$(t201, C, vorne, schmale\_Seite)$
	`bleibe_stehen`$(t201, C, B)$ `vor_einer_Wand`$(t201, C, vorne, schmale\_Seite)$
Begriff 2:	`drehe_dich`$(t201, B, H, langsame\_Drehung, links)$ `vor_einer_Wand`$(t201, H, rechts, breite\_Seite)$
	`bleibe_stehen`$(t201, H, G)$ `vor_einer_Wand`$(t201, H, rechts, breite\_Seite)$
Begriff 2:	`bewege_dich`$(t201, G, J, langsame\_Bewegung, vorne)$ `entlang_einer_Wand`$(t201, G, J, rechts, breite\_Seite, parallel)$ `parallel_in_einer_Ecke`$(t201, J, rechts\_vorne, breite\_Seite)$
	`bleibe_stehen`$(t201, J, A)$ `entlang_einer_Wand`$(t201, G, J, rechts, breite\_Seite, parallel)$ `parallel_in_einer_Ecke`$(t201, J, rechts\_vorne, breite\_Seite)$

Abbildung 6.1: Beispiel für eine Planinstanz

Kapitel 7

Diskussion

In der Literatur wird eine Fülle von Projekten beschrieben, deren Zielsetzung die Entwicklung von Robotersystemen der sog. dritten Generation ist. Eine tabellarische Übersicht von Projekten mit Schwerpunkt Navigation findet sich in [Hoppen, 1992] (S.45-47). [Kreuziger und Cord, 1992] liefern ferner einen umfassenden Bericht über den Einsatz von Lernverfahren in der Robotik. Ein kurzer, punktueller Vergleich operationaler Begriffe mit einigen ausgewählten Arbeiten findet sich darüberhinaus in [Klingspor *et al.*, 1995].

Das Ziel des vorliegenden Kapitels ist es nicht, den in dieser Arbeit vorgeschlagenen Ansatz zur Repräsentation operationaler Begriffe im Hinblick auf ihren Einsatz in mobilen Robotersystemen durch eine Gegenüberstellung mit vergleichbaren Arbeiten in das Forschungsgebiet der Robotik einzuordnen. Vielmehr geht es darum, anhand einer kurzen Darstellung zweier grundlegender und entgegengesetzter Positionen bei der Entwicklung mobiler Systeme zu zeigen, daß die an ihnen häufig geäußerte Kritik auf den Einsatz operationaler Begriffe in mobilen Robotersystemen nicht zutrifft. Dies zeigt zugleich die Vorzüge, die mit dem Einsatz operationaler Begriffe verbunden sind.

Die Entwicklung eines der ersten mobilen und weitgehend autonomen Robotersysteme mit der Bezeichnung SHAKEY begann bereits 1969 am *Stanford Research Institute* (siehe hierzu z.B. [Nilsson, 1984]). SHAKEY ist mit einer Videokamera und optischen Entfernungsmessern ausgestattet, die es ihm ermöglichen die durchfahrene Umgebung mit Hilfe einer Rasterkarte zu modellieren. Die Rasterkarte unterteilt die Umgebung in befahrbare und nicht befahrbare Regionen und wird während der Ausführung der Fahrten ergänzt. Die Erkennung von nicht befahrbaren Bereichen der Umgebung erfolgt dabei im Prinzip dadurch, daß SHAKEY während einer Fahrt an Hindernisse anstößt. Für die Wegplanung werden die in der Karte gespeicherten Umweltinformationen dahingehend ausgenutzt, daß an günstigen Positionen (z.B. entlang von Hindernissen) Landmarken gesetzt werden, die zu einem Wegenetz verbunden werden. Innerhalb eines aufgebauten Wegenetzes wird der kürzeste Weg zur jeweiligen Zielposition mit Hilfe des A^*-Suchalgorithmus

[Hart *et al.*, 1968] ermittelt.

Sobald ein Weg über die Landmarken ermittelt ist, nimmt SHAKEY seine Fahrt auf. Während der Fahrt erfolgt solange keine weitere Kontrolle, bis der Roboter auf ein unbekanntes Hindernis stößt! In diesem Fall wird nach dem beschriebenem Prinzip mit Hilfe der Kamera ein Weg gesucht, der um das Hindernis herumführt.

Die dem Gesamtsystem zugrundeliegende Kontrollstruktur entspricht einer nach funktionalen Aspekten aufgebauten Hierarchie verschiedener Systemkomponenten. Stark vereinfachend dargestellt, steht auf oberster Ebene eine Planungskomponente, die abstrakte Befehle (z.B. „gehe zum Raum 21“) in verschiedene Aufgaben unter Ausnutzung bekannter Landmarken (z.B. „fahre von Punkt A nach Punkt B“) zerlegt. Die nächst tiefere Komponente übersetzt diese Aufgaben in Fahrbefehle und verarbeitet dabei die vom Sensorsystem gelieferten Umweltinformationen. Obgleich der Beginn der Entwicklung von SHAKEY mehr als 25 Jahre zurückliegt, bildet diese in der Literatur als *horizontal, hierarchisch* charakterisierte Kontrollstruktur immer noch die Basiskonstruktionsweise für viele mobile Robotersysteme.

[Steels, 1993] kritisiert diesen traditionellen Ansatz zum Aufbau einer Kontrollstruktur dahingehend, daß Fahrbefehle losgelöst von während ihrer Ausführung registrierten Wahrnehmungen nicht immer genau den gewünschten Handlungen entsprechen. Wenn sich ein Roboter beispielsweise in einer bestimmten Richtung mit einer bestimmten zurückzulegenden Distanz bewegen soll, so können äußere Einflüsse geringeren Ausmaßes (z.B. Bodenunebenheiten, unregelmäßige Motoransteuerung, ...) dazu führen, daß sowohl die Richtung als auch die vorgegebene Distanz nicht exakt eingehalten werden.

Eine Kontrollstruktur für den Einsatz operationaler Begriffe ist zwar bisher noch nicht festgelegt worden, doch die in dieser Arbeit vorgestellte Repräsentation operationaler Begriffe verknüpft bereits Wahrnehmungen mit Handlungen auf allen erörterten Abstraktionsstufen mit Ausnahme der Ebene der Basis-Handlungsmerkmale. Im vorangegangenen Kapitel wurde deutlich, daß dies zur Folge hat, daß die Ausführung von Basis-Handlungsmerkmalen nach ihrer Übersetzung in elementare Roboteroperationen nicht unabhängig von bestimmten Wahrnehmungsmerkmalen erfolgt.

Die Ausführung des operationalen Begriffes `bewege_dich_durch_die_Tür` wird beispielsweise auf das Basis-Handlungsmerkmal `bewege_dich` zurückgeführt, und während der Ausführung von `bewege_dich` ist das handlungsorientierte Wahrnehmungsmerkmal `durch_eine_Tür` vorwärtszuinferieren. Die Ausführung von `bewege_dich_durch_die_Tür` gelingt schließlich, wenn in Richtung `hinten` die Wahrnehmung `vor_einer_Tür` registriert wird. Die Berücksichtigung von Angaben konkreter Distanzen und Positionen (Landmarken) ist daher beim Einsatz operationaler Begriffe nicht erforderlich. Gelingt es mit Hilfe einer noch zu entwickelnden Ausführungsüberwachungskomponente für operationale Begriffe dieses Potential im Rahmen einer geeigneten Kontrollstruktur auszunutzen, so trifft die von [Steels, 1993] geäußerte Kritik auf die Verwendung operationaler Begriffe zur Steuerung mobiler Robotersysteme nicht zu.

[Steels, 1993] kritisiert den skizzierten Aufbau einer Kontrollstruktur weiter dahingehend, daß die für die Planungsebene notwendige symbolische Repräsentation von Handlungen und Wahrnehmungen und ihre erforderliche Verbindung zu Roboteraktionen und Sensormessungen von den Designern des Systems vorgegeben werden. Die dem Folgerungsmechanismus des Planungssystems zugrundeliegenden Regeln können demnach lediglich die Behandlung von Situationen erlauben, die von den Systementwicklern als möglicherweise auftretende Situationen im vorhinein erkannt werden. Die Konsequenz ist, daß der Einsatz von Robotersystemen dieser Architektur auf einen eng umgrenzten Rahmen beschränkt bleiben muß.

Diese Kritik trifft auf den Einsatz operationaler Begriffe nur bedingt zu, da mit Ausnahme der Basis-Handlungsmerkmale alle zu ihrer Repräsentation notwendigen Abstraktionsebenen mit Hilfe maschineller Lernverfahren gelernt und in diesem Sinne nicht a priori im Detail festgelegt werden. Die dem Lernprozeß zugrundeliegende Repräsentation verhilft dem System darüberhinaus zu einer *Flexibilität* im Umgang mit ungewöhnlichen, nicht vorhergesehenen Situationen. So wird beispielsweise das Durchfahren einer Türöffnung auch dann als solche erkannt werden, wenn der Türrahmen nicht die zu erwartende Breite von 25-30cm aufweißt, sondern vielleicht von einer 1-2m breiten Mauer umschlossen wird.

Der Rahmen, in dem maschinelle Lernverfahren zum Lernen operationaler Begriffe eingesetzt werden, ist allerdings zugegebenermaßen durch die verwendete Repräsentationssprache begrenzt. Es stellt sich jedoch die Frage, ob dies wirklich zu gravierenden Nachteilen führt, da sie leicht und vor allem unabhängig von den bereits bestehenden Systemkomponenten erweitert werden kann. Die Lernphase des Roboters erspart den Entwicklern bei einer Einsatzortveränderung eine Restrukturierung des Gesamtsystems, und verhilft dem System insofern zu einer hohen *Adaptivität* im Hinblick auf wechselnde Arbeitsumgebungen.

Eine Erhöhung der Adaptivität steht auch bei Arbeiten im Vordergrund, die versuchen, die unterschiedlichen Verhaltensebenen von Lebewesen nachzubilden (siehe z.B. [Brooks, 1991] für eine kurze Übersicht).

Die Kernidee dieser Ansätze bildet ein von [Brooks, 1986] vorgestelltes Konzept für eine Kontrollstruktur, bei der die einzelnen Systemkomponenten nicht mehr Funktions-, sondern Verhaltensebenen darstellen, die hierarchisch geordnet sind (die sog. *vertikale, verhaltensorientierte Kontrollstruktur*). Mit jeder Verhaltensebene wird ein bestimmtes Ziel verfolgt (z.B. ziellos umherfahren, Zusammenstöße mit Objekten vermeiden, Objekte identifizieren, ...) und ein bestimmtes Verhalten des Systems realisiert. Alle Verhaltensebenen haben Zugriff auf die Sensorinformationen und können die Roboterfahrt direkt beeinflussen! Der Ausfall einer höheren Ebene führt daher nicht unweigerlich zum Ausfall des Gesamtsystems. Die sich in dieser Kontrollstruktur widerspiegelnde Philosophie entspricht der Forderung nach der

Untrennbarkeit von Handlungen und Wahrnehmungen, die auch bereits aus der Kritik von [Steels, 1993] herauszuhören ist.

Die bisher nach diesem Konzept entwickelten Systeme (z.B. [Mataric, 1992] bzw. [Mataric, 1994], [Firby, 1992], [Steels, 1991]) zeichnen sich durch eine hohe Adaptivität aus. Da für sie kein a priori Wissen über die Einsatzumgebungen erforderlich ist, können sie ausgezeichnet in unbekannten Umgebungen operieren. Ein rein verhaltensbasierter Ansatz birgt jedoch die Problematik einer nur unzureichenden Möglichkeit der Einflußnahme auf das Systemverhalten von außen. Die von [Firby, 1992] und [Steels, 1991] vorgeschlagenen Ansätze erlauben zwar in gewisser Hinsicht eine Manipulation des Gesamtverhaltens des Systems, von einer adäquaten *Mensch-Maschine-Kommunikation*, wie sie bereits für SHAKEY realisiert wurde, sind sie jedoch weit entfernt.

Im Gegensatz dazu können operationale Begriffe als Kommandosprache höherer Ebene fungieren. Sie bieten ebenfalls eine enge Verbindung von Handlungen und Wahrnehmungen. Diese Verbindung ist im wesentlichen auf allen Repräsentationsebenen vorhanden und daher im besonderen auch auf der Kommandoebene. Damit bieten operationale Begriffe die Möglichkeit, daß Systemverhalten durch das System selbst in einer für Menschen verständlichen Form zu erklären. Wenn beispielsweise der Befehl `bewege_dich_durch_die_Tür` vom System nicht befolgt werden kann, weil der Raum keine Türöffnung besitzt, so könnte eine mögliche Erklärung des Systems für die Nichtbefolgung der Anweisung in leichter Weise mit Hilfe der zur Auftragsverfolgung ausgeführten operationalen Begriffe generiert werden. Eine mögliche Erklärung könnte etwa lauten:

Der Befehl `bewege_dich_durch_die_Tür` *konnte nicht durchgeführt werden, weil der Befehl* `bewege_dich_entlang_der_Tür` *nicht durchgeführt werden konnte. Der Befehl* `bewege_dich_entlang_der_Tür` *konnte nicht durchgeführt werden, weil die Wahrnehmung* `vor_einer_Tür` *nicht registriert werden konnte. Es wurde stattdessen der Befehl* `bewege_dich_parallel_zur_Wand_in_die_Ecke` *viermal ausgeführt.*

Kapitel 8

Zusammenfassung und Ausblick

Die vorliegende Arbeit ist im Rahmen des Projektes B-Learn II (Behavioural Learning II, ESPRIT P 7274) angefertigt worden, dessen leitende Idee die Integration maschineller Lernverfahren in reale Robotersysteme ist. Mit dieser Zielsetzung des Projektes ist ein zweifaches Potential verbunden. Das Potential für die Robotik liegt in einer einfacheren und flexibleren Handhabung zukünftiger Robotersysteme. Dies umfaßt zum Beispiel die Portierbarkeit von Robotern in neue, bis dato unbekannte Arbeitsumgebungen, und damit die Möglichkeit, mit unerwartet auftretenden Ereignissen umzugehen. Es soll fernerhin möglich sein, die Einrichtungs- bzw. Umrüstungszeiten für Roboteranwendungen zu verkürzen und die Bedienung von Robotern menschengerechter zu gestalten. Aus Sicht der Robotik konzentrieren sich die Forschungsaktivitäten innerhalb des Projektes demnach auf zwei Arbeitsfelder: die Handhabung und die Navigation. Das Potential für das maschinelle Lernen liegt in der Aussicht, eine Verbindung zwischen Handlung einerseits und der mit ihr einhergehenden Wahrnehmung andererseits im Zuge einer intergrierenden Repräsentation herzustellen, um auf diesem Wege eine Begriffsoperationalisierung zu erzielen. Die für maschinelle Lernverfahren notwendige Repräsentation von Beispielen und Beobachtungen zum Lernen muß dann nicht mehr vorab definiert werden, sondern kann direkt aus den Sensordaten des Roboters während der Durchführung einer bestimmten Handlung gelernt werden. Ausgehend von diesen Überlegungen ist es ein Ziel innerhalb des Projektes, aus Sensordaten und zugehörigen Elementarhandlungen eines mobilen Roboters abstraktere handlungsorientierte Wahrnehmungs- und wahrnehmungsorientierte Handlungsmerkmale mit Hilfe logikbasierter induktiver Lernverfahren zu gewinnen und im Zuge des Aufbaus einer hierarchischen Begriffsstruktur miteinander zu verbinden.

Während mit Beginn der vorliegenden Arbeit bereits ein erster, erfolgversprechender Ansatz zum Lernen handlungsorientierter Wahrnehmungsmerkmale entwickelt wurde, besteht der Beitrag dieser Arbeit in der Gewinnung von Handlungsmerkmalen aus Roboter-Sensordaten, ihrer Zusammenführung mit der bereits bestehenden Begriffsstruktur handlungsorientierter Wahrnehmungsmerkmale und der Vorgabe einer Begriffshierarchie, an deren Spitze operationale Begriffe stehen.

Zu diesem Zweck wurden wahrnehmungsintegrierende Handlungsmerkmale entwickelt, die eine Verbindung zwischen handlungsorientierten Wahrnehmungsmerkmalen und den ebenfalls in dieser Arbeit vorgestellten Basis-Handlungsmerkmalen schaffen. Während die Basis-Handlungsmerkmale aus den vom Roboter aufgenommenen Sensordaten gemäß dem vorgeschlagenen Algorithmus inkrementell berechnet werden, erlaubt die vorgestellte Repräsentation der Begriffshierarchie eine Formulierung von Lernaufgaben für wahrnehmungsintergrierende Handlungsmerkmale und operationale Begriffe.

Exemplarisch wurde in der vorliegenden Arbeit gezeigt, daß diese Lernaufgaben für logikbasierte induktive Lernverfahren handhabbar sind, und daß die hier vorgeschlagene Repräsentation somit zum Lernen operationaler Begriffe aus Roboter-Sensordaten geeignet ist.

Operationale Begriffe integrieren Handlungs- und Wahrnehmungsmerkmale höherer Abstraktionsstufen. Das mit ihrer Verwendung zur Steuerung mobiler Robotersysteme verbundene Potential konnte in dieser Arbeit nur angedeutet werden. So wurde dargestellt, daß mit ihrer Hilfe Handlungsplanung für mobile Roboter in der Weise möglich ist, daß ausgehend von einem Zielbegriff rückwärts verkettend eine Sequenz operationaler Begriffe bestimmt werden kann, deren Ausführung zum Zielbegriff führt. Die Ausführung einer solchen Sequenz durch einen Roboter bedarf jedoch einer Ausführungsüberwachungskomponente, die ebenso wie ein Planungssystem für operationale Begriffe noch zu entwickeln ist. Zu den vordringlichsten zukünftigen Arbeiten zählt weiterhin eine Ergänzung der bisher zur Verfügung stehenden abstrakten Wahrnehmungsmerkmale. Für den realen Einsatz eines Robotersystems in einer unbekannten Umgebung, werden die bisher entwickelten Merkmale bei weitem nicht ausreichen. Erhebliche Anstrengungen werden weiterhin notwendig sein, um daß im Rahmen des Projektes bisher nur vereinzelnd angegangene Problem der Beispielgenerierung zufriedenstellend zu lösen. Die in der vorliegenden Arbeit vorgestellte Oberfläche zur Unterstützung der Beispielgenerierung wird erst dann zweckmäßig eingesetzt werden können, wenn Algorithmen zur Beispielerzeugung bereitstehen, deren Ergebnisse mit Hilfe der Oberfläche inspiziert werden können.

Anhang A

PRIAMOS

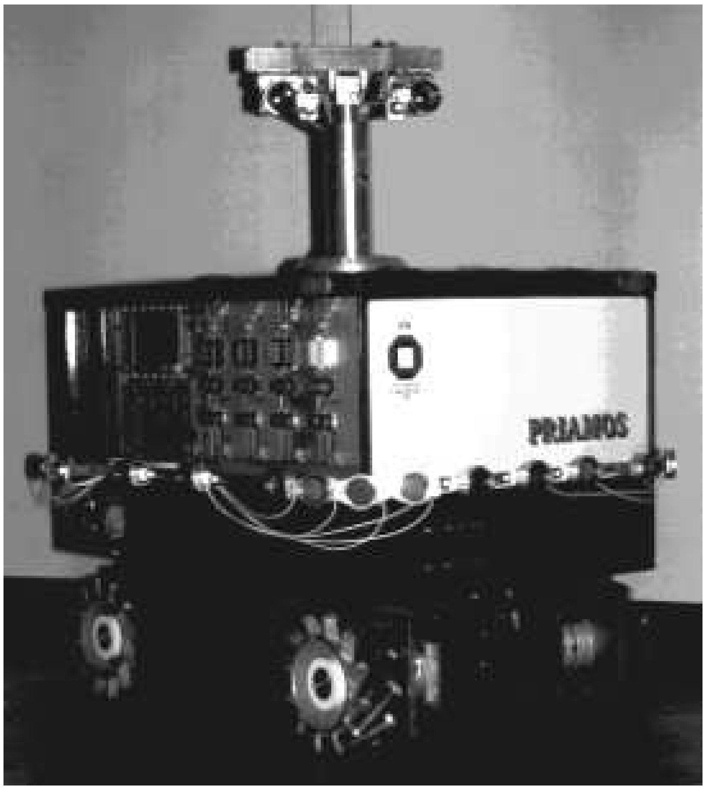


Abbildung A.1: Der Karlsruher Roboter PRIAMOS

Anhang B

Beispieldatenformate

```
2     \* Anzahl der Objekte
12    \* Anzahl der Polygonzüge
      \*  x, y, z
0.0 0.0 0.0   5.0 0.0 0.0  5.0 0.0 2.0   0.0 0.0 2.0
5.0 0.0 0.0   5.0 5.0 0.0  5.0 5.0 2.0   5.0 0.0 2.0
5.0 5.0 0.0   3.0 5.0 0.0  3.0 5.0 2.0   5.0 5.0 2.0
0.0 5.0 0.0   2.0 5.0 0.0  2.0 5.0 2.0   0.0 5.0 2.0
0.0 0.0 0.0   0.0 5.0 0.0  0.0 5.0 2.0   0.0 0.0 2.0
0.5 0.5 0.0   4.5 0.5 0.0  4.5 0.5 2.0   0.5 0.5 2.0
4.5 0.5 0.0   4.5 4.5 0.0  4.5 4.5 2.0   4.5 0.5 2.0
4.5 4.5 0.0   3.0 4.5 0.0  3.0 4.5 2.0   4.5 4.5 2.0
2.0 4.5 0.0   0.5 4.5 0.0  0.5 4.5 2.0   2.0 4.5 2.0
0.5 0.5 0.0   0.5 4.5 0.0  0.5 4.5 2.0   0.5 0.5 2.0
2.0 4.5 0.0   2.0 5.0 0.0  2.0 5.0 2.0   2.0 4.5 2.0
3.0 4.5 0.0   3.0 5.0 0.0  3.0 5.0 2.0   3.0 4.5 2.0
3     \* Anzahl der Polygonzüge
0.5   2.0   0.0    1.25  2.0   0.0 1.25  2.0   2.0    0.5   2.0   2.0
1.25  2.0   0.0    1.25  3.25  0.0 1.25  3.25  2.0    1.25  2.0   2.0
1.25  3.25  0.0    0.5   3.25  0.0 0.5   3.25  2.0    1.25  3.25 2.0
```

Tabelle B.1: Beispiel: Datenformat Simulationsumgebung

```
\*  x, y,α
1.0     4.0    180.0      2.5    4.0     100.0
1.25    4.0    180.0      2.5    4.0     90.0
1.5     4.0    180.0      2.5    4.0     90.0
1.75    4.0    180.0      2.5    4.125   90.0
2.0     4.0    180.0      2.5    4.25    90.0
2.125   4.0    180.0      2.5    4.375   90.0
2.25    4.0    180.0      2.5    4.5     90.0
2.375   4.0    180.0      2.5    4.625   90.0
2.5     4.0    180.0      2.5    4.75    90.0
2.5     4.0    180.0      2.5    4.875   90.0
2.5     4.0    160.0      2.5    5.0     90.0
2.5     4.0    140.0      2.5    5.25    90.0
2.5     4.0    120.0      2.5    5.5     90.0
2.5     4.0    110.0      2.5    5.75    90.0
```

Tabelle B.2: Beispiel: Datenformat Roboterfahrt

Anhang C

Einfache Robotersimulation

C.1 Mathematische Grundlagen

Satz 2.1: *Sei r eine Strecke des* $\mathbb{R}^2$ *mit der Parameterdarstellung* $r = \mathring{v} + \lambda(\mathring{w} - \mathring{v})$ *mit* $0 \leq \lambda \leq 1$ *und* $\mathring{p}$ *ein beliebiger durch ein festes* λ *definierter Punkt auf* r*. Eine zu* r *orthogonale Gerade* s *durch den Punkt* $\mathring{p}$ *wird durch die Parameterdarstellung* $s = \mathring{p} + \mu(-x_2, x_1)$ *mit* $\mu \in \mathbb{R}^2$ *beschrieben, wenn* $(-x_2, x_1)$ *durch* $(x_1, x_2) := (\mathring{w} - \mathring{v})$ *bestimmt ist.*

Beweis Sei also $\vec{x} = (x_1, x_2) := (\mathring{w} - \mathring{v})$, dann ist $\vec{y} = (y_1, y_2) := (-x_2, x_1)$ sicher orthogonal zu $\vec{x}$, da für das Skalarprodukt $< \vec{x}, \vec{y} >= x_1(-x_2) + x_2 x_1 = 0$ gilt. Da weiter $\vec{x}$ parallel zu r ist ($\vec{x}$ ist ja der Richtungsvektor der Parameterdarstellung von r) und $\vec{y}$ orthogonal zu $\vec{x}$ ist, läßt sich ein $\mathring{q}$ der gesuchten Gerade durch eine Parallelverschiebung von $\vec{y}$ in $\mathring{p}$ berechnen. So ist zum Beispiel durch $\mathring{q} := \vec{y} + \mathring{p} = (-x_2 + p_1, x_1 + p_2)$ ein entsprechender Punkt $\mathring{q}$ bestimmt. Mit der $\mathring{p}$ und $\mathring{q}$ enthaltenen Geraden in Parameterdarstellung $s = \mathring{p} + \mu(\mathring{q} - \mathring{p})$ *mit* $\mu \in \mathbb{R}^2$ ergibt sich durch Einsetzen die Behauptung. □

Hilfssatz 2.1: *Seien* g *und* s *Geraden des* $\mathbb{R}^2$ *mit den Parameterdarstellungen* $g = \mathring{u} + \lambda(\mathring{v} - \mathring{u})$ *und* $s = \mathring{w} + \mu(\mathring{x} - \mathring{w})$ *mit* $\lambda, \mu \in \mathbb{R}$*. Die beiden Geraden enthalten einen gemeinsamen Punkt* $\mathring{s}$*, gdw.* $(\mathring{v} - \mathring{u})$ *und* $(\mathring{x} - \mathring{w})$ *linear unabhängig sind.*

Beweis Da durch $(\mathring{v} - \mathring{u})$ und durch $(\mathring{x} - \mathring{w})$ die Richtungsvektoren der Parameterdarstellungen der Geraden gegeben sind, folgt aus der Definition der Parameterdarstellung, daß sie der jeweiligen Geraden parallel sind. Mit der Definition der linearen Unabhängigkeit folgt die Behauptung. □

Satz 2.2: *Seien* g *und* s *die in Hilfssatz 2.1 beschriebenen Geraden. Wenn* g *und* s *einen gemeinsamen Punkt* $\mathring{s}$ *enthalten, dann läßt sich* $\mathring{s}$ *durch Ein-*

setzen der Lösung des linearen Gleichungssystems

$$\begin{aligned} u_1 + \lambda(v_1 - u_1) &= w_1 + \mu(x_1 - w_1) \\ u_2 + \lambda(v_2 - u_2) &= w_2 + \mu(x_2 - w_2) \end{aligned}$$

für λ in g oder für μ in s berechnen.

Beweis Wenn $\mathring{s}$ in g und in s enthalten ist, dann gibt es durch die Definition der Parameterdarstellung ein $\lambda \in \mathbb{R}$ und ein $\mu \in \mathbb{R}$ mit $\mathring{s}=\mathring{u} + \lambda(\mathring{v} - \mathring{u})$ und $\mathring{s}=\mathring{w} + \mu(\mathring{x} - \mathring{w})$. Durch Gleichsetzung ergibt sich die Gleichung $\mathring{u} + \lambda(\mathring{v} - \mathring{u}) = \mathring{w} + \mu(\mathring{x} - \mathring{w})$ und somit, da sowohl die Addition als auch die Skalarmultiplikation im $\mathbb{R}^2$ komponentenweise definiert ist, das lineare Gleichungssystem der Behauptung. □

Satz 2.3: *Sei r eine Gerade des $\mathbb{R}^2$ mit der Parameterdarstellung $r = \mathring{u} + \lambda(\mathring{v} - \mathring{u})$ mit $\lambda \in \mathbb{R}$ und $\mathring{p}$ ein Punkt auf r. Die Koordinaten der Punkte $\mathring{q}$ und $\mathring{q}'$ auf r mit Abstand $l \in \mathbb{R}_+$ von $\mathring{p}$ entsprechen den Lösungen des Gleichungssystems*

$$\begin{aligned} (x - p_1)^2 + (y - p_2)^2 - l^2 &= 0 \\ (x - u_1)(v_2 - u_2) - (y - u_2)(v_1 - u_1) &= 0. \end{aligned}$$

Beweis Der Punkt $\mathring{p}$ sei der Mittelpunkt eines Kreises mit Radius l. Dann sind $\mathring{q}$ und $\mathring{q}'$ gerade die Schnittpunkte des Kreises mit der Geraden r. Nach Pythagoras folgt für die Koordinaten aller Punkte eines Kreises mit Radius l um den Ursprung die Beziehung $x^2 + y^2 = l^2$. Mit der Verschiebung des Mittelpunktes des Kreises vom Ursprung in $\mathring{p}$ ergibt sich $(x-p_1)^2+(y-p_2)^2-l^2 = 0$. Die Koordinaten aller Punkte auf der Geraden r sind durch ein bestimmtes $\lambda \in \mathbb{R}$ und $x = u_1+\lambda(v_1-u_1)$, $y = u_2+\lambda(v_2-u_2)$ bestimmt. Durch die jeweilige Auflösung nach λ und Gleichsetzung der beiden Gleichungen ergibt sich für die Koordinaten (x,y) aller Punkte auf r die allgemeinere Beziehung $(x - u_1)(v_2 - u_2) - (y - u_2)(v_1 - u_1) = 0$. Da $\mathring{q}$ und $\mathring{q}'$ nach Definition sowohl auf r als auch auf dem Kreis mit Radius l und Mittelpunkt $\mathring{p}$ liegen, genügen ihre Koordinaten dem Gleichungssystem der Behauptung. □

Satz 2.5: *Sei r eine Gerade des $\mathbb{R}^2$ mit der Parameterdarstellung $r = \mathring{u} + \lambda(\mathring{v} - \mathring{u})$ mit $\lambda \in \mathbb{R}$ und $\mathring{p}$ ein Punkt auf r. Der Punkt $\mathring{q}$, der in Blickrichtung mit Abstand $a \in \mathbb{R}_+$ von $\mathring{p}$ auf r liegt, ist durch*

$$\mathring{q}=\mathring{p} + \frac{a}{\|\mathring{v} - \mathring{u}\|} * (\mathring{v} - \mathring{u})$$

bestimmt.

Beweis Da $\overset{\circ}{p}$ und $\overset{\circ}{q}$ auf r liegen, gibt es ein $\lambda \in \mathbb{R}$, so daß $\overset{\circ}{q} = \overset{\circ}{p} + \lambda(\overset{\circ}{v} - \overset{\circ}{u})$ gilt. Da der Vektor $\frac{(\overset{\circ}{v} - \overset{\circ}{u})}{||\overset{\circ}{v} - \overset{\circ}{u}||}$ der Einheitsvektor zu $(\overset{\circ}{v} - \overset{\circ}{u})$ ist, hat der Vektor $\frac{a}{||\overset{\circ}{v} - \overset{\circ}{u}||} * (\overset{\circ}{v} - \overset{\circ}{u})$ die Länge a und die gleiche Orientierung wie $(\overset{\circ}{v} - \overset{\circ}{u})$. Demnach ist $\frac{a}{||\overset{\circ}{v} - \overset{\circ}{u}||}$ das gesuchte λ. □

C.2 Algorithmische Beschreibung

Die vorgestellte Simulation berechnet für eine beliebig vorgegebene Szenerie die von Sensoren eines Roboters ermittelten Distanzen zu Hindernissen entlang eines Weges. Die Szenerie besteht aus der zweidimensionalen Beschreibung eines Raumes, eines Roboters und eines vom Roboter zurückzulegenden Weges. Die Geschwindigkeit des Roboters und die Abtastfrequenz seiner Sensoren ist beliebig aber konstant über die zurückzulegende Strecke. Beschleunigungs- und Verzögerungseffekte bleiben demnach unberücksichtigt. Für die Sensoren gilt, daß sowohl ihre Anzahl als auch ihre Abtastrichtung relativ zur Bewegungsrichtung des Roboters frei wählbar sind. Sie werden als ideale Entfernungsmesser betrachtet, die bis zu einer vorgegebenen Distanz exakte Daten liefern. Die Szene, der Roboter sowie sein Weg sind aus Teilstrecken zusammengesetzt. Die im Algorithmus durchgeführten Berechnungen basieren auf den im Abschnitt 2.1.3 angestellten Überlegungen. Entsprechende Verweise sind in die algorithmische Beschreibung eingearbeitet.

Die zur Beschreibung der Szene notwendigen Daten werden in einem ARRAY G verwaltet. Hierbei handelt es sich um die Koordinaten der Anfangs- und Endpunkte der die Szene beschreibenden Teilstrecken. Auch die Fahrt des Roboters wird durch die Koordinaten der Anfangs- und Endpunkte der die Roboterfahrt beschreibenden Teilstrecken definiert. Sie werden in einem ARRAY R abgelegt. Der Speicherung der ermittelten Simulationsdaten dient hingegen ein ARRAY E. Dabei handelt es sich um Sensorinformationen zu diskreten Zeitpunkten sowie um die jeweilige Position des Roboters sowohl in absoluten Koordinaten, als auch durch Angabe der entsprechenden Teilstrecke des Weges.

Die den Roboter beschreibenden Daten werden in ein ARRAY S eingelesen. Dies sind die Anzahl der Sensoren, deren Abtastrichtung relativ zur Bewegungsrichtung des Roboters, sowie deren Lage am Roboter relativ zur Roboterposition. Die Abtastrichtung eines Sensors wird durch einen Winkel, den Abtastwinkel, festgelegt. Der Bestimmung seiner Position dient neben dem sogenannten Offset-Winkel eine Entfernungsangabe, der Offset-Abstand zur definierten Roboterposition. Abbildung C.1 zeigt eine somit mögliche Roboterdefinition. Die verschiedenen Offset-Winkel und -Abstände besitzen über die gesamte Simulation konstante Werte. Die Koordinaten, die die Lage der Sensoren relativ zur Position des Roboters beschreiben, sind hingegen abhängig von seiner Bewegungsrichtung. Gleiches gilt für die

Orientierung der Abtastrichtung der Sensoren. Die relativen Koordinaten und die Abtastrichtung der Sensoren lassen sich jedoch aus Abtastwinkeln, Offset-Winkeln und -Abständen berechnen, so daß ihre Angabe eine für die Simulation hinreichende Beschreibung des Roboters darstellt.

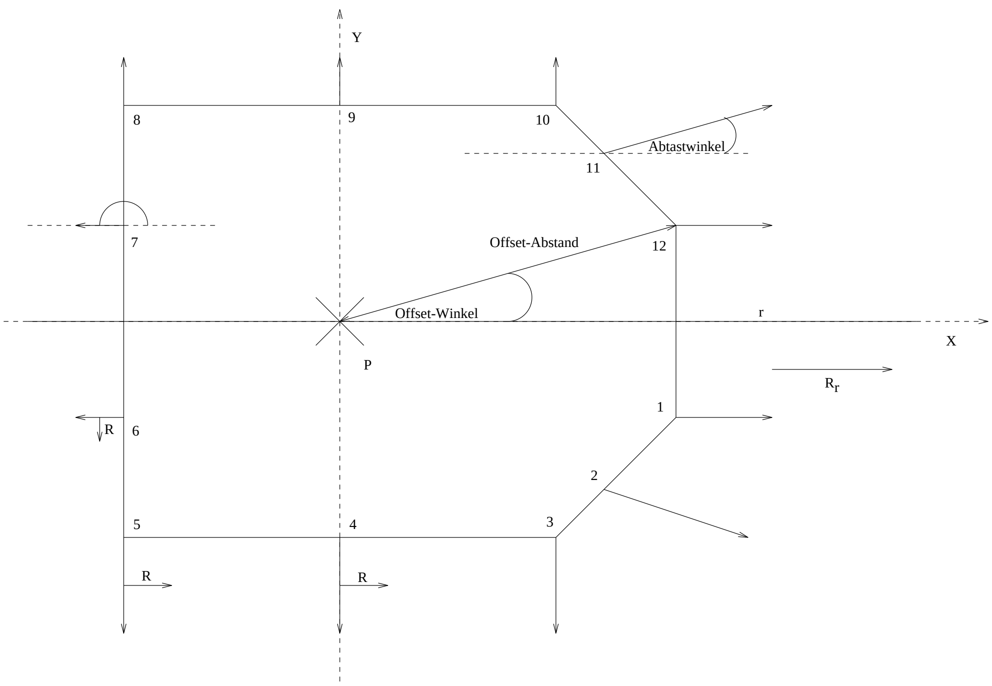


Abbildung C.1: Definition des Roboters

Neben diesen Daten werden die absoluten Positionskoordinaten aller Sensoren sowie die Position der von ihnen erkannten Punkte der Szenerie über die gesamte Simulationszeit in einem ARRAY A festgehalten. Am Ende der Simulation werden die Einträge von ARRAY A gemeinsam mit weiteren Simulationsdaten in einem gesonderten File gesichert. Dieses Datenfile könnte die Basis für eine eingeschränkte graphische Darstellung der Simulation bilden.

C.2.1 Verwendete Datenstrukturen

G = ARRAY[1..a-g],[1..4] OF REAL
G[i,j] : i-te Strecke
j=1 : x-Koordinate des Anfangspunktes der i-ten Strecke
j=2 : y-Koordinate des Anfangspunktes der i-ten Strecke
j=3 : x-Koordinate des Endpunktes der i-ten Strecke
j=4 : y-Koordinate des Endpunktes der i-ten Strecke

R = Array[1..a-r],[1..4] OF REAL
R[k,l] : k-te Strecke
l=1 : x-Koordinate des Anfangspunktes der k-ten Strecke
l=2 : y-Koordinate des Anfangspunktes der k-ten Strecke
l=3 : x-Koordinate des Endpunktes der k-ten Strecke
l=4 : y-Koordinate des Endpunktes der k-ten Strecke

E = Array[1..a-mp],[1..(a-s + 3)] OF REAL
E[m,n] : m-ter Meßzeitpunkt
n=1 : Meßwert von Sensor 1 zum Zeitpunkt m
n=2 : Meßwert von Sensor 2 zum Zeitpunkt m
n=3 : Meßwert von Sensor 3 zum Zeitpunkt m
...
n=a-s : Meßwert von Sensor a-s zum Zeitpunkt m
n=a-s + 1 : x-Koordinate der Roboterposition zum Zeitpunkt m
n=a-s + 2 : y-Koordinate der Roboterposition zum Zeitpunkt m
n=a-s + 3 : Nummer der Strecke auf der sich der Roboter zum Zeitpunkt m befindet

S = Array[1..a-s],[1..3] OF REAL
S[s, t] : Sensor s
t=1 : Offset-Abstand von Sensor s
t=2 : Offset-Winkel von Sensor s
t=3 : Abtastwinkel von Sensor s

A = Array[1..a-mp],[1..4a-s] OF REAL
A[u, v] : u-ter Meßzeitpunkt
v=1 : x-Koordinate der absoluten Position von Sensor 1 zum Zeitpunkt u
v=2 : y-Koordinate der absoluten Position von Sensor 1 zum Zeitpunkt u
v=3 : x-Koordinate des von Sensor 1 zum Zeitpunkt u erkannten Punktes
v=4 : y-Koordinate des von Sensor 1 zum Zeitpunkt u erkannten Punktes
v=5 : x-Koordinate der absoluten Position von Sensor 2 zum Zeitpunkt u
v=6 : y-Koordinate der absoluten Position von Sensor 2 zum Zeitpunkt u
v=7 : x-Koordinate des von Sensor 2 zum Zeitpunkt u erkannten Punktes
v=8 : y-Koordinate des von Sensor 2 zum Zeitpunkt u erkannten Punktes
...
v=4a-s - 3: x-Koordinate der absoluten Position von Sensor a-s zum Zeitpunkt u
v=4a-s - 2: y-Koordinate der absoluten Position von Sensor a-s zum Zeitpunkt u
v=4a-s - 1: x-Koordinate des von Sensor a-s zum Zeitpunkt u erkannten Punktes
v=4a-s : y-Koordinate des von Sensor a-s zum Zeitpunkt u erkannten Punktes

C.2.2 Algorithmen

Prozedurname: ROSI
Argumente: keine
BEGIN

1. INITIALISIERUNG;
2. SIMULATION;
3. SIMULATIONSDATEN_SICHERN;
4. GRAFIKDATEN_SICHERN;

END; (* Hauptschleife ROSI *)

Prozedurname: INITIALISIERUNG
Argument: Zeitpunkt

- Liest die Daten zur Beschreibung der Szene ein.
- Liest die Daten zur Beschreibung des Roboterpfades ein.
- Liest die Daten zur Beschreibung des Roboters ein und berechnet aus ihnen weitere für die Simulation benötigte Daten.

BEGIN

1. SZENEDATEN_EINLESEN;
   (* ARRAY G werden seine Werte zugewiesen; a-g:Anzahl der Strecken *)
2. PFADDATEN_EINLESEN;
   (* ARRAY R werden seine Werte zugewiesen; a-r:Anzahl der Strecken *)
3. ROBOTERDATEN_EINLESEN;
   (* ARRAY S werden seine Werte zugewiesen; a-s:Anzahl der Sensoren; die Roboter-Geschwindigkeit V , sowie die Abtastfrequenz F wird festgelegt *)
4. SIMULATIONSPARAMETER_BERECHNEN;
   (* W:Weglänge; GZ:benötigte Gesamtzeit; a-mp:Anzahl der Meßpositionen; a:Abstand zwischen zwei Meßpunkten *)
5. MESSPOSITIONEN_BERECHNEN;

END; (* INITIALISIERUNG *)

Prozedurname: SIMULATION
Argumente: keine

- Für jeden Meßzeitpunkt wird separat die Prozedur Simuliere aufgerufen, die für jeden Sensor die Simulationsdaten des entsprechenden Zeitpunktes berechnet.

BEGIN

FOR m:=1 TO a-mp DO

SIMULIERE(m);

END; (* FOR m *)

END; (* SIMULATION *)

Prozedurname: SIMULIERE
Argument: Zeitpunkt

- Berechnet für einen beliebigen Zeitpunkt der Simulation die Sensordaten des Roboters.

```
BEGIN

n:=1;
REPEAT

    Strecken-Nr := E[Zeitpunkt, (a-s + 3)];
    Roboter-Position := (E[Zeitpunkt, (a-s + 1)], E[Zeitpunkt, (a-s + 2)]);
    OFFSET_OR_ABTASTRICHTUNG (n, Strecken-Nr, OFFSET, Offset);
    OFFSET_OR_ABTASTRICHTUNG (n, Strecken-Nr, ABTASTRICHTUNG,
    Abtastrichtung);
    Sensor-Position := VEKTORADDITION(Roboter-Position, Offset);
    SENSORDATEN_BERECHNEN(Zeitpunkt, n, Sensor-Position, Abtastrichtung);
    n:=n+1;

UNTIL (n > a-s);

END; (* SIMULIERE *)
```

Prozedurname: MESSPOSITIONEN_BERECHNEN
Argumente: keine

- Berechnet die Positionen auf dem Pfad R, an denen der Roboter Sensor-Informationen erhält.

```
BEGIN
Zähler := 1;
Bezugspunkt := (R[Zähler,1], R[Zähler,2]);

FOR m:=1 TO a-mp DO

    NÄCHSTER_PUNKT(Bezugspunkt, a, Zähler, Meßposition);
    (* Meßposition ist der Punkt mit Abstand a in Blickrichtung (Definition 2.5)
       vom Bezugspunkt auf der Geraden Zähler. *)

    IF PUNKT_NICHT_AUF_STRECKE?(Meßposition, Zähler)
       THEN
            Zähler := Zähler + 1;
            Hilfs-Bezugspunkt := (R[Zähler, 1], R[Zähler,2]);
            (* Anfangspunkt der Folgestrecke *)
            Hilfs-a := PUNKTE_ABSTAND(Hilfs-Bezugspunkt, Meßposition);
            (* Länge des über die Strecke Zähler-1 herausragenden Stückes. *)
            NÄCHSTER_PUNKT(Hilfs-Bezugspunkt, Hilfs-a, Zähler, Meßposition);
            (* Umklapp-Strategie *)
    END;

    (E[m, (a-s + 1)], E[m, (a-s + 2)]) := Meßposition;
    (* die m-te Meßposition wird gesichert *)
    E[m, (a-s + 3)] := Zähler;
    (* die Nummer der Strecke auf der sich die m-te Meßposition befindet wird gesichert *)
    Bezugspunkt := Meßposition;
    (* die ermittelte m-te Meßposition wird neuer Bezugspunkt zur Berechnung der
    (m+1)-ten Meßposition*)

END; (* FOR m *)

END; (* MESSPOSITIONEN_BERECHNEN *)
```

Prozedurname: OFFSET_OR_ABTASTRICHTUNG
Argumente: Sensor-Nr, Strecken-Nr, Anfrage, Vektor

- Anfrage = OFFSET : Berechnet die Position von Sensor-Nr relativ zur Position des Roboters in Abhängigkeit von seiner Bewegungsrichtung, die durch den Richtungsvektor von Strecke-Nr vorgegeben ist.
- Anfrage = ABTASTRICHTUNG : Berechnet einen Vektor, dessen Orientierung die Abtastrichtung von Sensor-Nr angibt. Die Abtastrichtung ist ebenfalls von der Bewegungsrichtung des Roboters abhängig.

```
BEGIN

CASE Anfrage OF
    ABTASTRICHTUNG: Orientierungswinkel := ORIENTIERUNGSWINKEL_BERECHNEN(
                                           RICHTUNGSVEKTOR_BERECHNEN(
                                             (R[Strecken-Nr, 1], R[Strecken-Nr, 2]),
                                             (R[Strecken-Nr, 3], R[Strecken-Nr, 4])))
                    Abtastwinkel        := S[Sensor-Nr,3];
                    alpha               := Orientierungswinkel + Abtastwinkel;
                    IF alpha ≥ 360 THEN alpha := alpha - 360;
                    Hypothenuse         := 1;
    OFFSET:         Orientierungswinkel := ORIENTIERUNGSWINKEL_BERECHNEN(
                                           RICHTUNGSVEKTOR_BERECHNEN(
                                             (R[Strecken-Nr, 1], R[Strecken-Nr, 2]),
                                             (R[Strecken-Nr, 3], R[Strecken-Nr, 4])))
                    Offset-Winkel       := S[Sensor-Nr, 2];
                    alpha               := Orientierungswinkel + Offset-Winkel;
                    IF alpha ≥ 360 THEN alpha := alpha - 360;
                    Hypothenuse         := S[Sensor-Nr,1];
END;

Vektor[1] := cos(alpha) * Hypothenuse;
Vektor[2] := sin(alpha) * Hypothenuse;
END; (* OFFSET_OR_ABTASTRICHTUNG *)
```

Prozedurname: SENSORDATEN_BERECHNEN
Argumente: Meßpunkt-Nr, Sensor-Nr, Sensor-Position, Abtastrichtung

- Berechnet die Abstandsmeßung von Sensor-Nr, wenn sich der Roboter an der durch Meßpunkt-Nr gegebenen Position befindet.

```
BEGIN

A[Meßpunkt-Nr, 4 * Sensor-Nr - 3] := Sensor-Position[1];
A[Meßpunkt-Nr, 4 * Sensor-Nr - 2] := Sensor-Position[2];
A[Meßpunkt-Nr, 4 * Sensor-Nr - 1] := 99999;
A[Meßpunkt-Nr, 4 * Sensor-Nr] := 99999;
beta := Abtastrichtung;
kleinster-Abstand := 99999;

FOR i:=1 TO a-g DO

    alpha := RICHTUNGSVEKTOR_BERECHNEN((G[i, 1], G[i, 2]), (G[i, 3], G[i, 4]))
    IF LINEAR_ABHÄNGIG?(alpha, beta)
      THEN
          Schnitt := FALSE; (* kein Schnitt nach Hilfssatz 2.1 *)
          i:=i+1;
      ELSE
          b := G[i, 2] - Abtastrichtung[2];
          lambda := (b*beta[1]-a*beta[2]) / (alpha[1]beta[2]-alpha[2]beta[1]);
```

```
            IF NOT(0 ≤ lambda ≤ 1)
              THEN
                  Schnitt :=FALSE; (* kein Schnitt nach Bemerkung 2.2 *)
                  i:=i+1;
              ELSE
                  Schnitt := TRUE; (* Schnitt-Koordinaten nach Hilfssatz 2.2 *)
                  Schnitt-Punkt[1] := G[i,1] + lambda * alpha[1];
                  Schnitt-Punkt[2] := G[i,2] + lambda * alpha[2];
            END;
        END;

    IF Schnitt = TRUE
        THEN
            Abstand := PUNKTE_ABSTAND(Sensor-Position, Schnitt-Punkt);
            R       := ((-1)*Abtastrichtung[2], Abtastrichtung[1]);
                        (* Orthogonale Hilfsgerade nach Satz 2.1 *)
            rechts  := RECHTS?(Schnitt-Punkt, Sensor-Position, R);
            IF rechts AND Abstand < kleinster-Abstand
                   THEN
                       kleinster-Abstand := Abstand;
                       A[Meßpunkt-Nr, 4 * Sensor-Nr - 1] := Schnitt-Punkt[1];
                       A[Meßpunkt-Nr, 4 * Sensor-Nr]     := Schnitt-Punkt[2];
            END;
    END;

END; (* FOR i *)

E[Meßpunkt-Nr, Sensor-Nr] := kleinster-Abstand;

END; (* SENSORDATEN_BERECHNEN *)
```

Prozedurname: NÄCHSTER_PUNKT
Argumente: Bezugspunkt, Abstand, Strecken-Nr, *Vektor*

- Berechnet für eine gegebene Strecke und einen Punkt auf dieser Strecke einen Punkt auf der durch die Strecke definierten Geraden, der in Blickrichtung mit einem bestimmten Abstand auf der Geraden liegt.

```
BEGIN

Richtungsvektor := RICHTUNGSVEKTOR_BERECHNEN(
                                (R[Strecken-Nr, 1], R[Strecken-Nr, 2]),
                                (R[Strecken-Nr, 3], R[Strecken-Nr, 4])
                                                  );
Skalar        := Abstand / NORM(Richtungsvektor);
Hilfs-Vektor  := SKALARMULTIPLIKATION(Skalar, Richtungsvektor);

Vektor        := VEKTORADDITION(Bezugspunkt, Hilfs-Vektor);

END; (* NÄCHSTER_PUNKT *)
```

Prozedurname: PUNKT_NICHT_AUF_STRECKE?
Argumente: Punkt, Strecken-Nr

- Für einen gegebenen Punkt einer Geraden wird überprüft, ob er nicht auf einer definierten Teilstrecke der Geraden liegt.
- Der Punkt muß vom Anfangspunkt der definierten Teilstrecke der Geraden in Blickrichtung auf der Geraden liegen !

BEGIN

```
Anfangspunkt := (R[Strecken-Nr, 1], R[Strecken-Nr, 2]);
(* erster Punkt der definierten Teilstrecke *)
Streckenlänge := STRECKENLÄNGE_BERECHNEN( (R[Strecken-Nr, 1], R[Strecken-Nr, 2]),
                                           (R[Strecken-Nr, 3], R[Strecken-Nr, 4]));
IF PUNKTE_ABSTAND(Anfangspunkt, Punkt) ≤ Streckenlänge
   THEN
        RETURN FALSE;
   ELSE
        RETURN TRUE;
END;

END; (* PUNKT_NICHT_AUF_STRECKE? *)
```

Prozedurname: RECHTS?
Argumente: Punkt, Punkt-Gerade, Orientierung-Gerade

- Für einen beliebigen Punkt 'Punkt' wird überprüft, ob er in Blickrichtung rechts von einer gegebenen Gerade liegt.
- Der Punkt 'Punkt' ist kein Punkt der Geraden.

```
BEGIN

u1 := Punkt-Gerade[1]; u2 := Punkt-Gerade[2];
q1 := Punkt[1]; q2        := Punkt[2];

v1 := Orientierung-Gerade[1] + u1;
v2 := Orientierung-Gerade[2] + u2;

IF q1(u2 − v2) + q2(v1 − u1) + (u1v2 − u2v1) < 0
   THEN
        RETURN TRUE;
        (* Punkt liegt nach Satz 2.4 in Blickrichtung rechts von der gegebenen Geraden *)
   ELSE
        RETURN FALSE;
END;

END; (* RECHTS? *)
```

Prozedurname: RICHTUNGSVEKTOR_BERECHNEN
Argumente: Anfangspunkt, Endpunkt

- Berechnet den Richtungsvektor für eine durch zwei Punkte definierte Strecke mit Orientierung vom Anfangs- zum Endpunkt.

```
BEGIN

Hilfsvektor := SKALARMULTIPLIKATION((-1), Anfangspunkt);
Richtungsvektor := VEKTORADDITION(Endpunkt, Hilfsvektor);

RETURN Richtungsvektor;

END; (* RICHTUNGSVEKTOR_BERECHNEN *)
```

Prozedurname: ORIENTIERUNGSWINKEL_BERECHNEN
Argument: Vektor

- Berechnet den Winkel zwischen Vektor und x-Achse.
- Der Winkel liegt zwischen 0 und 360 Grad.

BEGIN

Hypothenuse := NORM(Vektor);
Ankathete := BETRAG(Vektor[1]);

IF Vektor[1] $\geq$ 0 AND Vektor[2] $\geq$ 0 THEN alpha := $cos^{-1}(\frac{Ankathete}{Hypothenuse})$; END;
IF Vektor[1] $<$ 0 AND Vektor[2] $\geq$ 0 THEN alpha := $180 - cos^{-1}(\frac{Ankathete}{Hypothenuse})$; END;
IF Vektor[1] $\leq$ 0 AND Vektor[2] $<$ 0 THEN alpha := $180 + cos^{-1}(\frac{Ankathete}{Hypothenuse})$; END;
IF Vektor[1] $>$ 0 AND Vektor[2] $<$ 0 THEN alpha := $360 - cos^{-1}(\frac{Ankathete}{Hypothenuse})$; END;

RETURN alpha;

END; (* ORIENTIERUNGSWINKEL_BERECHNEN *)

Prozedurname: STRECKENLÄNGE_BERECHNEN
Argumente: Anfangspunkt, Endpunkt

- Berechnet die Länge einer durch zwei Punkte definierten Strecke.

BEGIN

Streckenlänge := PUNKTE_ABSTAND(Anfangspunkt, Endpunkt);

RETURN Streckenlänge;

END; (* STRECKENLÄNGE_BERECHNEN *)

Anhang D

Beispiele

D.1 Operationale Begriffe

stehen(Trc,T_1,T_2,vor_einer_Tür,WRichtung,schmale_Seite,LW) &
p_bewegen(Trc,T_2,T_3,GK,WRichtung,durch_eine_Tür,WRichtung1) &
stehen(Trc,T_3,T_4,vor_einer_Tür,hinten,schmale_Seite,durch_eine_Tür)
$\rightarrow$ bewege_dich_durch_die_Tür(Trc,T_1,T_4,T_3).

stehen(Trc,T_1,T_2,vor_einer_Tür,WRichtung,breite_Seite,LW) &
hw_WRichtung_nach_Rotation(WRichtung,DRichtung,WRichtung1) &
drehen(Trc,T_2,T_3,RGK,DRichtung,vor_einer_Tür,WRichtung1) &
stehen(Trc,T_3,T_4,vor_einer_Tür,WRichtung1,schmale_Seite,vor_einer_Tür)
$\rightarrow$ drehe_dich_vor_der_Tür(Trc,T_1,T_4,T_3).

stehen(Trc,T_1,T_2,vor_einer_Wand,WRichtung,WSeite,entlang_einer_Tür) &
hw_gegenüberliegende_Richtung(WRichtung,WRichtung1) &
bewegen(Trc,T_2,T_3,GK,hinten,vor_einer_Tür,WRichtung1) &
stehen(Trc,T_3,T_4,vor_einer_Tür,WRichtung1,WSeite,vor_einer_Tür)
$\rightarrow$ bewege_dich_vor_die_Tür(Trc,T_1,T_4,T_3).

stehen(Trc,T_1,T_2,parallel_in_einer_Ecke,WRichtung,WSeite,LW) &
hw_WRichtung_nach_Bewegung(WRichtung,RelRichtung,WRichtung1) &
p_bewegen(Trc,T_2,T_3,GK,RelRichtung,entlang_einer_Tür,WRichtung1) &
stehen(Trc,T_3,T_4,vor_einer_Wand,WRichtung1,WSeite,entlang_einer_Tür)
$\rightarrow$ bewege_dich_entlang_der_Tür(Trc,T_1,T_4,T_3).

stehen(Trc,T_1,T_2,vor_einer_Wand,WRichtung,WSeite,LW) &
hw_WRichtung_nach_Bewegung(WRichtung,RelRichtung,WRichtung1) &
p_bewegen(Trc,T_2,T_3,GK,RelRichtung,entlang_einer_Tür,WRichtung1) &
stehen(Trc,T_3,T_4,vor_einer_Wand,WRichtung1,WSeite,entlang_einer_Tür)
$\rightarrow$ bewege_dich_entlang_der_Tür(Trc,T_1,T_4,T_3).

stehen(Trc,T_1,T_2,parallel_in_einer_Ecke,WRichtung,WSeite,LW) &
hw_WRichtung_einer_Ecke2(WRichtung,DRichtung,WRichtung1) &
drehen(Trc,T_2,T_3,RGK,DRichtung,parallel_in_einer_Ecke,WRichtung1) &
stehen(Trc,T_3,T_4,parallel_in_einer_Ecke,WRichtung1,WSeite,parallel_in_einer_Ecke)
$\rightarrow$ drehe_dich_in_der_Ecke_zur_vorderen_Wand(Trc,T_1,T_4,T_3).

stehen(Trc,T_1,T_2,vor_einer_Wand,WRichtung,WSeite,LW) &
hw_WRichtung_nach_Bewegung(WRichtung,RelRichtung,WRichtung1) &
p_bewegen(Trc,T_2,T_3,GK,RelRichtung,entlang_einer_Wand,WRichtung1) &
hw_Richtung_einer_Ecke1(WRichtung1,RelRichtung,WRichtung2) &
stehen(Trc,T_3,T_4,parallel_in_einer_Ecke,WRichtung2,WSeite,entlang_einer_Wand)
$\rightarrow$ bewege_dich_dich_parallel_zur_Wand_in_die_Ecke(Trc,T_1,T_4,T_3).

stehen(Trc,T_1,T_2,parallel_in_einer_Ecke,WRichtung,WSeite,LW) &
hw_WRichtung_nach_Bewegung(WRichtung,RelRichtung,WRichtung1) &
p_bewegen(Trc,T_2,T_3,GK,RelRichtung,entlang_einer_Wand,WRichtung1) &
hw_Richtung_einer_Ecke1(WRichtung1,RelRichtung,WRichtung2) &
stehen(Trc,T_3,T_4,parallel_in_einer_Ecke,WRichtung2,WSeite,entlang_einer_Wand)
$\rightarrow$ bewege_dich_dich_parallel_zur_Wand_in_die_Ecke(Trc,T_1,T_4,T_3).

stehen(Trc,T_1,T_2,vor_einer_Wand,WRichtung,WSeite,LW) &
hw_WRichtung_nach_Rotation(WRichtung,DRichtung,WRichtung1) &
drehen(Trc,T_2,T_3,RGK,DRichtung,vor_einer_Wand,WRichtung1) &
hw_neue_Wahrnehmungsseite(WSeite,WSeite1) &
stehen(Trc,T_3,T_4,vor_einer_Wand,WRichtung1,WSeite1,vor_einer_Wand)
$\rightarrow$ drehe_dich_vor_der_Wand(Trc,T_1,T_4,T_3).

stehen(Trc,T_1,T_2,vor_einer_Wand,WRichtung,WSeite,LW) &
bewegen(Trc,T_2,T_3,GK,WRichtung,vor_einer_Wand,vorne) &
stehen(Trc,T_3,T_4,vor_einer_Wand,vorne,WSeite,vor_einer_Wand)
$\rightarrow$ bewege_dich_näher_zur_Wand(Trc,T_1,T_4,T_3).

D.2 Hintergrundwissen

hw_WRichtung_nach_Rotation(links,links,vorne).
hw_WRichtung_nach_Rotation(links,rechts,hinten).
hw_WRichtung_nach_Rotation(rechts,links,hinten).
hw_WRichtung_nach_Rotation(rechts,rechts,vorne).
hw_WRichtung_nach_Rotation(vorne,links,rechts).
hw_WRichtung_nach_Rotation(vorne,rechts,links).
hw_WRichtung_nach_Rotation(hinten,links,links).
hw_WRichtung_nach_Rotation(hinten,rechts,rechts).
hw_WRichtung_nach_Rotation(rechts_vorne,links,rechts_hinten).

```
hw_WRichtung_nach_Rotation(rechts_vorne,rechts,links_vorne).
hw_WRichtung_nach_Rotation(rechts_hinten,links,links_hinten).
hw_WRichtung_nach_Rotation(rechts_hinten,rechts,rechts_vorne).
hw_WRichtung_nach_Rotation(links_vorne,links,rechts_vorne).
hw_WRichtung_nach_Rotation(links_vorne,rechts,rechts_hinten).
hw_WRichtung_nach_Rotation(links_hinten,links,links_vorne).
hw_WRichtung_nach_Rotation(links_hinten,rechts,rechts_hinten).

hw_WRichtung_nach_Bewegung(links,hinten,rechts).
hw_WRichtung_nach_Bewegung(links,vorne,links).
hw_WRichtung_nach_Bewegung(rechts,hinten,links).
hw_WRichtung_nach_Bewegung(rechts,vorne,rechts).
hw_WRichtung_nach_Bewegung(vorne,links,rechts).
hw_WRichtung_nach_Bewegung(vorne,rechts,links).
hw_WRichtung_nach_Bewegung(hinten,links,links).
hw_WRichtung_nach_Bewegung(hinten,rechts,rechts).
hw_WRichtung_nach_Bewegung(rechts_vorne,links,rechts).
hw_WRichtung_nach_Bewegung(rechts_vorne,hinten,links).
hw_WRichtung_nach_Bewegung(rechts_hinten,links,links).
hw_WRichtung_nach_Bewegung(rechts_hinten,vorne,rechts).
hw_WRichtung_nach_Bewegung(links_vorne,rechts,links).
hw_WRichtung_nach_Bewegung(links_vorne,hinten,rechts).
hw_WRichtung_nach_Bewegung(links_hinten,rechts,rechts).
hw_WRichtung_nach_Bewegung(links_hinten,vorne,links).

hw_Richtung_einer_Ecke1(rechts,vorne,rechts_vorne).
hw_Richtung_einer_Ecke1(rechts,hinten,links_vorne).
hw_Richtung_einer_Ecke1(links,vorne,links_vorne).
hw_Richtung_einer_Ecke1(links,hinten,rechts_vorne).
hw_Richtung_einer_Ecke1(hinten,rechts,rechts_vorne).
hw_Richtung_einer_Ecke1(hinten,links,links_vorne).
hw_Richtung_einer_Ecke1(vorne,rechts,links_vorne).
hw_Richtung_einer_Ecke1(vorne,links,rechts_vorne).

hw_WRichtung_einer_Ecke2(rechts_vorne,links,rechts_hinten).
hw_WRichtung_einer_Ecke2(rechts_vorne,rechts,links_vorne).
hw_WRichtung_einer_Ecke2(links_vorne,links,rechts_vorne).
hw_WRichtung_einer_Ecke2(links_vorne,rechts,links_hinten).
hw_WRichtung_einer_Ecke2(links_hinten,links,links_vorne).
hw_WRichtung_einer_Ecke2(links_hinten,rechts,rechts_hinten).
hw_WRichtung_einer_Ecke2(rechts_hinten,links,links_hinten).
hw_WRichtung_einer_Ecke2(rechts_hinten,rechts,rechts_vorne).
```

Literaturverzeichnis

[Birk und Kelly, 1981] J.R. Birk und R.B. Kelly. An Overview of the Basic Research Needed to Advance the State of Knowledge in Robotics. In *IEEE Transactions on Systems, Man and Cybernetics*, Band 11, S. 574—579, 1981.

[Biundo *et al.*, 1993] Susanne Biundo, Andreas Günter, Joachim Hertzberg, Josef Schneeberger und Wolfgang Tank. Planen und Konfigurieren. In Günther Görz (Hrsg.), *Einführung in die künstliche Intelligenz*, Band 1, Kapitel 7, S. 767—828. Addison-Wesley, Bonn, New York , Paris [u.a.], 1993.

[B.R.A., 1989] B.R.A. British Robot Assoziation: Annual Report of UK Investment in Robot Automation in Manufacturing Industry. *Robotics and Autonomous Systems*, 5(3):289—292, 1989.

[Brady, 1985] M. Brady. Artificial Intelligence and Robotics. *Artificial Intelligence*, 26:79—121, 1985.

[Brooks, 1986] Rodney A. Brooks. A Robust Layered Control System for a Mobile Robot. In *IEEE Journal of Robotics and Automation*, 1986.

[Brooks, 1991] Rodney A. Brooks. The Role of Learning in Autonomous Robots. In Manfred K. Warmuth und Leslie G. Valiant (Hrsg.), *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, San Mateo, California, August 1991. Morgan Kaufmann.

[de Velde, 1991] Walter Van de Velde. Toward Learning Robots. In Walter Van de Velde (Hrsg.), *Toward Learning Robots*, Special Issues of Robotics and Autonomous Systems, S. 1—6. MIT Press, London, Cambridge, 1991.

[Dillmann *et al.*, 1993] R. Dillmann, J. Kreuziger und F. Wallner. PRIAMOS — An Experimental Platform for Reflexive Navigation. In Groen, Hirose und Thorpe (Hrsg.), *IAS-3: Intelligent Autonomous Systems*, S. 174—183. IOS Press, 1993.

[Fikes *et al.*, 1972] R. Fikes, P. Hart und N. J. and Nilsson. Learning and Executing Generalized Robot Plans. *Artificial Intelligence Journal*, 3:251—288, 1972.

[Fikes und Nilsson, 1971] R.E. Fikes und N.J. Nilsson. STRIPS: A New Approach to Theorem Proving in Problem Solving. *Journal of Artificial Intelligence*, 2:189—208, 1971.

[Firby, 1992] James R. Firby. Building symbolic primitives with continuous control routines. In James Hendler (Hrsg.), *Proc. of the first Intern. Conf. on AI Planning Systems (AIPS-92)*, S. 62—69, San Mateo, CA, 1992. Morgan Kaufmann.

[Gottheil *et al.*, 1992] K. Gottheil, H.-J. Kaufmann, Th. Kern und R. Zhao. *X und Motif — Eine Einführung in die Programmierung des Motif-Toolkits und des X-Windows-Systems.* Springer-Verlag, Berlin, Heidelberg, New York [u.a.], 1992.

[Guibas und Yao, 1980] L. J. Guibas und F. F. Yao. On Translating a Set of Rectangles. In *Proc. Twelfth ACM Symp. on Theory of Computing*, S. 154—160, Los Angeles, Calif., 1980.

[Habel *et al.*, 1993] C. Habel, M. Herweg und S. Pribbenow. Wissen über Raum und Zeit. In Günther Görz (Hrsg.), *Einführung in die künstliche Intelligenz*, Band 1, Kapitel 1, S. 139—204. Addison-Wesley, Bonn, New York , Paris [u.a.], 1993.

[Harnad, 1990] Stevan Harnad. The Symbol Grounding Problem. *Physica D*, 42:335—346, 1990.

[Hart *et al.*, 1968] Peter E. Hart, Nils J. Nilsson und Bertram Raphael. A Formal Basis for Heuristic Determination of Minimum Cost Paths. In *IEEE Transaction on Systems Science and Cybernetics*, Band SSC-4, S. 100—107, 1968.

[Helft, 1989] N. Helft. Induction as nonmonotonic inference. In *Proceedings of the 1st International Conference on Representation and Reasoning*, 1989.

[Hoppen, 1992] Peter Hoppen. *Autonome Mobile Roboter — Echtzeitnavigation in bekannter und unbekannter Umgebung.* Nummer 87 in Reihe Informatik. BI Wissenschaftsverlag, Mannheim, Wien, Zürich [u.a.], 1992.

[Janakiraman und Wallner, 1992] P.A. Janakiraman und F. Wallner. Reliable Prediction of Ultrasonic Range Measurements for Robotic Applications. Zu beziehen an der Universität Karlsruhe, Institut für Prozeßrechentechnik und Robotik, Deutschland, Oktober 1992.

[Kietz und Wrobel, 1992] J.-U. Kietz und S. Wrobel. Controlling the complexity of learning in logic through syntactic and task-oriented models. In S. Muggleton (Hrsg.), *Inductive Logic Programming*, Kapitel 16, S. 335—360. Academic Press, London, 1992. Ebenfalls erhältlich als Arbeitspapier der GMD, Nr. 503, 1991.

[Klingspor *et al.*, 1995] Volker Klingspor, Katharina Morik und Anke Rieger. Learning Operational Concepts from Sensor Data of a Mobile Robot. *Machine Learning Journal*, 1995. Zur Begutachtung eingereicht im September 1994.

[Klingspor und Morik, 1995] Volker Klingspor und Katharina Morik. Towards Concept Formation Grounded on Perception and Action of a Mobile Robot. In *Proc. of the 4th Intern. Conference on Intelligent Autonomous Systems*, 1995. Im Druck.

[Klingspor und Sklorz, 1995] Volker Klingspor und Stefan Sklorz. Representing, Learning, and Executing Operational Concepts. Zur Begutachtung eingereicht: 3rd European Workshop on Learning Robots, 1995.

[Klingspor, 1994a] Volker Klingspor. GRDT: Enhancing Model-Based Learning for Its Application in Robot Navigation. 518, Forschungsberichte des Fachbereichs Informatik der Universität Dortmund, Deutschland, Januar 1994.

[Klingspor, 1994b] Volker Klingspor. Repräsentation operationaler Begriffe. In Ralph Bergmann, Jürgen Paulokat, Anna-Maria Schoeller und Holger Wache (Hrsg.), *Beiträge zum 8. Workshop Planen und Konfigurieren*, S. 88—97, Kaiserslautern, April 1994.

[Knieriemen, 1991] Thomas Knieriemen. *Autonome Mobile Roboter: Sensordateninterpretation und Weltmodellierung zur Navigation in unbekannter Umgebung.* Nummer 80 in Reihe Informatik. BI Wissenschaftsverlag, 1991.

[Kreuziger und Cord, 1992] Jürgen Kreuziger und Stephan Cord. Anwendungen symbolischer Lernverfahren in der Robotik. Interner Bericht 23, Universität Karlsruhe, Fakultät für Informatik, Deutschland, 1992.

[Levi, 1988] Paul Levi. *Planen für autonome Montageroboter.* Nummer 191 in Informatik-Fachberichte; Subreihe künstliche Intelligenz. Springer-Verlag, Berlin, Heidelberg [u.a.], 1988.

[Lübbe, 1995] Marcus Lübbe. Datengesteuertes Lernen von syntaktischen Einschränkungen des Hypothesenraumes für modellbasiertes Lernen. Diplomarbeit, Fachbereich Informatik, Universität Dortmund, Deutschland, 1995.

[Mataric, 1992] Maja J. Mataric. Integration of Representation Into Goal-Driven Behaviour-Based Robots. In *IEEE Transaction on Robotics and Automation*, Band 8, S. 304—312, 1992.

[Mataric, 1994] Maja J. Mataric. Interaction and Intelligent Behaviour. Doktorarbeit, MIT, Cambridge, Massachusetts, 1994.

[Mitchell, 1982] Tom M. Mitchell. Generalization as Search. *Artificial Intelligence Journal*, 18(2):S.203—226, 1982.

[Morik *et al.*, 1993] K. Morik, S. Wrobel, J.-U. Kietz und W. Emde. *Knowledge Acquisition and Machine Learning — Theory, Methods, and Applications.* Academic Press, London, 1993.

[Morik und Rieger, 1993] Katharina Morik und Anke Rieger. Learning action-oriented perceptual features for robot navigation. In *Workshop notes: Learning Robots of the ECML-93*, 1993. Ebenfalls erhältlich als Research Report 3, Univ. Dortmund, Informatik VIII, D-44221 Dortmund, Deutschland.

[Morik, 1989] Katharina Morik. Sloppy Modeling. In Katharina Morik (Hrsg.), *Knowledge Representation and Organization in Machine Learning*, S. 107—134. Springer Verlag, Berlin, New York, 1989.

[Morik, 1993] Katharina Morik. Maschinelles Lernen. In Günther Görz (Hrsg.), *Einführung in die künstliche Intelligenz*, Band 1, Kapitel 3, S. 247—303. Addison-Wesley, Bonn, New York , Paris [u.a.], 1993.

[Muggleton, 1992] Stephen Muggleton. Inductive Logic Programming. In Stephen Muggleton (Hrsg.), *Inductive Logic Programming*, The A.P.I.C. Series, No 38, Kapitel 1, S. 1—29. Academic Press, London [u.a.], 1992.

[Nilsson, 1984] N. J. Nilsson. Shakey the Robot. Technical Note 323, Artificial Intelligence Center, SRI International, Menlo Park, CA, 1984.

[Pavlidis, 1982] Theodosios Pavlidis. *Algorithms for Graphics and Image Processing.* Springer-Verlag, Berlin, Heidelberg, 1982.

[Plotkin, 1970] G. D. Plotkin. A note on Inductive Generalization. In B. Meltzer und D. Michie (Hrsg.), *Machine Intelligence*, Kapitel 8, S. 153—163. American Elsevier, 1970.

[Plotkin, 1971] G. D. Plotkin. A further note on inductive generalization. In B. Meltzler und D. Michie (Hrsg.), *Machine Intelligence*, Band 6, Kapitel 8, S. 101—124. American Elsevier, 1971.

[Sommer *et al.*, 1993] Edgar Sommer, Werner Emde, Jörg-Uwe Kietz, Katharina Morik und Stefan Wrobel. MOBAL 2.2 User Guide. Arbeitspapiere der GMD, GMD Bonn, Deutschland, September 1993.

[Steels, 1991] L. Steels. Emergent frame recognition and its use in artificial creatures. In *Proc. of the 12th IJCAI*, S. 1219—1224, Sidney, Australia, 1991.

[Steels, 1993] Luc Steels. Building Agents out of Autonomous Behavior Systems. In L. Steels und R. Brooks (Hrsg.), *The 'artificial life' route to 'artificial intelligence' – Building situated embodied agents.* Lawrence Erlbaum, New Haven, 1993.

[Wessel, 1995] Stephanie Wessel. Lernen qualitativer Merkmale aus numerischen Robotersensordaten. Diplomarbeit, Universität Dortmund, Fachbereich Informatik, 44221 Dortmund, Deutschland, 1995.

[Wrobel, 1987] Stefan Wrobel. Higher-Order Concepts in a Tractable Knowledge Representation. In K. Morik (Hrsg.), *GWAI-87 11th German Workshop on Artificial Intelligence*, S. 129—138, Berlin, New York, Tokyo, Oktober 1987. Springer Verlag.

[Wrobel, 1991] Stefan Wrobel. Die Umweltverankerung von Konzeptbildungsprozessen. *KI – Künstliche Intelligenz*, 5(1):22—26, 1991.

[W.R.S.D., 1983] W.R.S.D. Worldwide Robotics Survey and Directory. , Robot Institute of America, Dearborne, MI, 1983.

Abbildungsverzeichnis

Tabellenverzeichnis