SpIRo
Sprachbasierte Instruktion kraftbasierter Roboterbewegungen

Von der Universität Bayreuth
zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte Abhandlung

von
Kim Wölfel
aus Datteln

1. Gutachter:	Prof. Dr. Dominik Henrich
2. Gutachter:	Prof. Dr. Diedrich Wolter

Tag der Einreichung:	08.01.2021
Tag des Kolloquiums:	26.05.2021

Danksagung

An dieser Stelle möchte ich allen danken, die mich während und besonders am Ende meiner Promotion unterstützt haben. Allen voran möchte ich dabei meinem Doktorvater Prof. Dr. Dominik Henrich dafür danken, dass er mir ermöglicht hat, an seinem Lehrstuhl zu promovieren, und mich währenddessen mit hilfreichen Vorschlägen unterstützt hat.

Außerdem möchte ich meinen Kollegen am Lehrstuhl und den Studenten danken, welche stets zu Diskussionen bereit waren, egal ob fachlicher oder anderer Natur. Im Speziellen Dorian Rohner, Eric Orendt, Edgar Schmidt, Josua Bloeß, Michael Gradmann und Tobias Werner, da mich diese nicht nur während der Arbeit, sondern auch in der Freizeit begleitet haben.

Letztlich möchte ich meiner Familie und meinen Freunden danken, welche mich auf dem Weg begleitet, unterstützt und motiviert haben. Besonders möchte ich dabei meiner Frau Nadine und meiner Mutter danken, welche in jeder Situation für mich da waren.

Zusammenfassung

Eine intuitive Robotersteuerung zu entwickeln, ist seit langem Gegenstand der Forschung. Ansätze basierend auf graphischen Oberflächen, Sprachsteuerung, Gestenerkennung oder einer Kombination dieser Ansätze wurden für verschiedene Anwendungsfälle getestet. Eine rudimentäre Sprachsteuerung von mobilen Robotern ist mittlerweile sogar schon im Haushalt möglich. Neben dem häuslichen Gebrauch sollen Roboterarme jedoch in Zukunft auch in kleinen und mittleren Unternehmen eingesetzt werden und dort im besten Fall von Nichtexperten im Bereich der Robotik bedienbar sein.

Ein Grund dafür, dass eine Sprachsteuerung von Roboterarmen noch nicht weit verbreitet ist, stellt, neben den hohen Kosten der Roboterarme, die Schwierigkeit der Abbildung von natürlichsprachlichen Instruktionen in für das Robotersystem verständliche Instruktionen dar. Gerade bei kraftbasierten Bewegungen ist eine Vereinfachung der Instruktionen notwendig, da man von einem Nutzer nicht erwarten kann, dass alle für die Bewegung notwendigen Parameter explizit spezifiziert werden. Um dem zu entgehen, werden in dieser Arbeit eine Reihe an Verfahren vorgestellt, welche eine intuitive sprachbasierte Instruktion von kraftbasierten Roboterbewegungen erlaubt.

In dieser Arbeit wird zunächst ein Ansatz erweitert, welcher eine Abbildung von Instruktionen auf Roboterbewegung basierend auf physikalischen Gesetzen und Eigenschaften erlaubt, indem Kombinationen dieser Bewegungen ermöglicht werden. Um den Nutzern die Last abzunehmen numerische Kraftwerte angeben zu müssen, wird ein Modell vorgestellt, welches eine Abbildung von unscharfen Parametern auf scharfe Kraftparameter erlaubt. Die Notwendigkeit eines solchen Modells wurde dabei mit Hilfe einer speziellen Nutzerstudie dargelegt, welche ebenfalls Bestandteil dieser Ausarbeitung ist. Da Menschen dazu neigen, offensichtliche Parameter in Instruktionen wegzulassen, wird ein System vorgestellt, welches eine Validierung solcher Instruktionen mittels sogenannter Affordanzen ermöglicht. Da selbst syntaktisch und semantisch korrekte Instruktionen zu fehlerhaften Ausführungen führen können, weil sich beispielsweise der Nutzer nicht vollkommen über die Fähigkeiten des Robterarms im Klaren ist, wird zudem ein Ansatz vorgestellt, welcher die Bewegungen zunächst in einer physikbasierten Simulation ausführt und auf eventuelle Abweichungen überprüft.

Das Gesamtsystem wird zuletzt mit Hilfe eines Prototyps evaluiert, welcher die oben genannten Ansätze umsetzt. Abschließend erfolgt eine Zusammenfassung der Arbeit, sowie ein Ausblick auf mögliche Erweiterungen.

Abstract

A longterm goal in robotics research is the generation of an intuitive robot control. Approaches based on graphical user interfaces, speech control, gesture control or a combination of these have been evaluated for a variety of applications. Meanwhile, a basic speech control of mobile robots is already possible in households. Besides the domestic use, it is also planned to introduce robots in small and medium-sized enterprises, where even non-experts should be able to operate them.

One of the reasons for not commonly using speech control of robot arms is, besides the high costs, the challenge of transforming natural language user input into instructions that are comprehensible for robot systems. Especially in the case of force-based motions a simplification of instructions is necessary, since it should not be required of a user to state all essential motion parameters. To avoid this, several approaches are presented in this work, which allow an intuitive speech-based instruction of force-based robot motions.

Initially, an approach is extended which allows the transformation of instructions into robot motions based on the laws of physics as well as physical parameters by facilitating a combination of these motions. To relieve users from specifying crisp force parameters, a model is introduced, which allows a mapping of uncertain parameters to crisp force parameters. The necessity of such a model is shown by a specific user study in this work. Since people tend to omit blatant parameters in instructions, a system for validating such instructions by means of so called affordances is introduced. Even syntactically and semantically correct instructions might lead to invalid executions, because users might lack profound knowledge of e.g. the robot capabilities. Thus, an approach is introduced to execute motions in a physics-based simulation and to check the results for possible deviations.

Last but not least the overall system is being evaluated by means of a prototype, which implements the afore-mentioned approaches. Eventually this work is summed up and an overview of possible future work is presented.

Inhaltsverzeichnis

Abkürzungsverzeichnis

$\mathcal{PPE}$	Menge der Prinzipiellen Physikalischen Effekte, mit Elementen PPE
$\mathcal{VPE}$	Menge der Verbalisierten Physikalischen Effekte, mit Elementen VPE
$\mathcal{EVE}$	Menge der Elementaren Verbalisierten Effekte, mit Elementen EVE
$\mathcal{KVE}$	Menge der Kombinierten Verbalisierten Effekte, mit Elementen KVE
$\mathcal{A}$	Menge der Affordanzen, mit Elementen A
$\mathcal{HM}$	Hybride Bewegung (Hybrid Motions), mit Elementen HM
$\mathcal{MP}$	Manipulationsprimitiv, mit Elementen MP
$\mathcal{MPN}$	Manipulationsprimitivnetz, mit Elementen MPN
FFM	Fuzzy Force Model
POI	Parameter of Interest
WoZ	Wizard of Oz
KMU	Kleine und mittlere Unternehmen
LBR	Leichtbauroboter
MRK	Mensch-Roboter-Kollaboration
TCP	Tool Center Point
TTS	Sprachsynthese (Text To Speech)

Kapitel 1

Einleitung

Dieses Kapitel gibt zunächst einen kurzen Überblick über aktuelle Konzepte zur Instruktion von Roboterarmen und motiviert die Notwendigkeit eines Ansatzes, welcher die Möglichkeiten einer sprachbasierten Instruktion erweitert (Kapitel 1.1). Abhängig davon werden Fragestellungen definiert, welche in dieser Arbeit bearbeitet und beantwortet werden (Kapitel 1.2) und Rahmenbedingungen festgelegt, für welche dieses System konzipiert wurde (Kapitel 1.3). Abschließend beinhaltet dieses Kapitel eine Darstellung über die Kapitel dieser Arbeit (Kapitel 1.4).

1.1 Motivation

Roboterarme werden seit einiger Zeit vermehrt in der Industrie eingesetzt, um die Produktivität zu erhöhen und Arbeiter zu entlasten. Die Hauptaufgabe bestand dabei bisher in der Massenfertigung von Werkstücken. Ein relativ neues Einsatzgebiet für Roboter sind hingegen kleine und mittlere Unternehmen (KMU), welche sich meist auf die Anfertigung kleiner Stückzahlen oder auch von Sonderanfertigungen konzentrieren. Da jedoch auch in KMU der Einsatz von Roboterarmen eine Effizienzsteigerung verspricht, gibt es immer mehr Bestrebungen, mögliche Lösungen umzusetzen (siehe Abbildung 1.1). Ein Beispiel dafür ist die *KMU-NetC*[1] des Bundesministeriums für Forschung und Bildung. Neben Industrierobotern (beispielsweise zum Fräsen in Schreinereien), kommen auch Leichtbauroboter (LBR) in Frage. Diese können zwar nicht so schwer heben wie Industrieroboter, erlauben dafür jedoch eine Zusammenarbeit mit dem Arbeiter als sogenannte CoBots. Neben dem Einsatz für repetitive oder ergonomisch belastende Arbeiten können diese zudem als flexible Unterstützung (z.B. zum Fixieren eines Bauteils) genutzt werden. Außerdem profitiert man von der hohen Genauigkeit der Roboter und erspart Arbeitern eine möglicherweise gesundheitsschädliche Umgebung (z.B. Staub).

Eine große Herausforderung, die es dabei zu bewältigen gilt, ist, dass die Interaktion mit den LBR möglichst einfach ist, damit neben der Anschaffung des LBR nicht auch noch kostspielige und zeitintensive Fortbildungen der Arbeiter anfallen. In den letzten Jahrzehnten wurde eine Vielzahl an Konzepten zur intuitiven Interaktion von Robotern und Menschen entwickelt, welche

[1] https://www.bmbf.de/de/automatisierung-fuer-kmu-kollege-roboter-an-der-hobelbank-7675.html

Abb. 1.1: Anwendungsfälle von CoBots in KMU: Einsatz zur Palettierung [Khalid17] (links) und Vermessung von Bauteilen [Hessler] (rechts).

auf visuelle, auditive, haptische Kanäle oder einer Kombination dieser (Multimodale Steuerung) beruhen. Aktuellere Konzepte bauen zudem auf VR- und AR-Brillen auf, um den Nutzern Informationen bereitzustellen, die sichtbare Komponenten um Expertenwissen erweitern oder auch nicht sichtbare Komponenten einblenden (z.B. den Motor innerhalb einer Karosserie).

Ähnlich wie bei der zwischenmenschlichen Interaktion existiert auch bei der Mensch-Roboter-Kooperation (MRK) ein hohes Potential für Missverständnisse, da Nichtexperten, aber auch Experten, in manchen Situationen die Fähigkeiten und das Verhalten eines Roboterarms falsch einschätzen. Dabei kann es sich um einfache Daten wie maximale Lasten und Geschwindigkeiten oder aber auch komplexere Daten, wie beispielsweise die geplante Trajektorie handeln. Eine weitere Herausforderung ist demnach, fehlerhafte Instruktionen zu erkennen, zu analysieren und entsprechend zu reagieren. Die Reaktion kann dabei entweder daraus bestehen, dass fehlende oder fehlerhafte Parameter aus Kontextwissen erschlossen werden, oder, dass zusammen mit dem Nutzer ein Ausbessern der Instruktion durchgeführt wird.

Die Probleme, die es zu lösen gilt, sind also zum einen die Abbildung von symbolischer Information, in Form von menschlichen Kommunikationsformen, auf subsymbolische Information, welche für einen Roboter verständlich sind. Zum anderen die Rückrichtung, um einem Nutzer bestmöglich auftretende Probleme oder Schwierigkeiten bei der Verarbeitung von Instruktionen zu erläutern. Die beste Lösung seitens Kommunikationskanal scheint dabei eine multimodale Interaktion zu sein, da manche Kommunikationskanäle für bestimmte Objektparameter für den Menschen intuitiver sind und für das System das Auftreten von Mehrdeutigkeiten verringert. Um solch eine multimodale Interaktion zu gestalten, sollten die einzelnen Kanäle erschöpfend untersucht werden, damit die bestmöglichen Kombinationen erzeugt werden können. Da eine Untersuchung aller Kanäle den Umfang einer einzelnen Arbeit überschreiten würde, beschäftigt sich diese Arbeit mit dem akustischen Kanal, bzw. dem verbalen Dialog.

1.2 Problemstellung

Obwohl eine Vielzahl an Ansätzen zur Sprachsteuerung von Robotern existiert, ist das Gebiet der Instruktionen von kraftbasierten Roboterbewegungen noch nicht sehr dicht besiedelt. Dies liegt vermutlich daran, dass für solche Bewegungen eine höhere Anzahl an Parametern übergeben oder aus dem Kontext generiert werden muss. Das Ziel dieser Arbeit ist die Definition eines Ansatzes zur verbalen Mensch-Roboter-Interaktion, welcher sowohl eine flexible Instruktion von kraftbasierten Bewegungen erlaubt, als auch eine robuste Instruktion und Ausführung von Roboterbewegungen ermöglicht. Als flexibel gilt ein System hier, wenn es mit einem umfangreichen Wortschatz umgehen kann und auch unvollständige Instruktionen über Kontextwissen in Roboterbewegungen transformieren kann. Als robust ein System hier bezeichnet, wenn eingehende Instruktionen sowohl auf sprachlicher Ebene, als auch auf Bewegungsebene validiert werden und fehlerhafte Ausführung gegebenenfalls zusammen mit dem Nutzer ausgebessert werden. Formal lässt sich dieses Problem über folgenden Abbildungen g und ρ darstellen:

$$g : \mathcal{K} \times \mathcal{I} \rightarrow \mathcal{M} \times \mathcal{F}. \tag{1.1}$$

Diese Abbildung transformiert gegebenes Kontextwissen bzw. Domänenwissen $\mathcal{K}$ zusammen mit einer übergebenen Instruktion $\mathcal{I}$ entweder in eine Roboterbewegung $\mathcal{M}$, falls eine eindeutige Abbildung möglich ist, in eine Rückmeldung $\mathcal{F}$, falls die Abbildung nicht eindeutig auf eine Bewegung abgebildet werden kann, oder auf eine Kombination aus beiden, falls neben der reinen Ausführung zusätzliche Informationen an den Nutzer zurückgegeben werden müssen. Sei nun $\rho(g)$ eine Bewertungsfunktion, welche die Zufriedenheit von Nutzern mit der Abbildung von Instruktionen auf Bewegungen und Rückmeldungen beschreibt, so kann man die generelle Herausforderung einer Sprachsteuerung als folgendes Optimierungsproblem darstellen:

$$\arg\max_{g} \rho(g). \tag{1.2}$$

Eine Erweiterung der Abbildung g spiegelt sich dadurch in einer Verbesserung von ρ wieder. Die Abbildung g kann dabei mit Bezug auf eine Vielzahl an Eigenschaften erweitert werden. In der Arbeit liegt der Fokus auf der Flexibilität und der Robustheit. Um diese beiden Eigenschaften zu verbessern werden im folgenden eine Reihe an wissenschaftlichen Fragestellungen formuliert, welche in den nachfolgenden Kapiteln beantwortet werden.

Ein Ansatz zur Definition solcher Bewegungen steht in dieser Arbeit in Form von sogenannten *Verbalisierten Physikalischen Effekten* (VPE) [Spangenberg17] zur Verfügung, welche im nächsten Kapitel näher beschrieben werden. Der Hintergedanke bei VPE ist die Verknüpfung einer Roboterbewegung mit genau einem physikalischen Gesetz, was auf der einen Seite eine für Menschen verständliche Grundlage darstellt, auf der anderen Seite lediglich eine Definition von elementaren Bewegungen zulässt. Um diesen intuitiven Ansatz auch für weitere Bewegungen nutzen zu können, ist die erste in dieser Arbeit bearbeiteten Fragestellungen:

F1 In wie weit kann die Menge der durch VPE abbildbaren kraftbasierten Bewegungen über deren Kombination erweitert werden?

Um Nutzern die Übergabe numerischer Werte zu ersparen, werden diese durch Platzhalter ersetzt, vordefiniert oder mit Hilfe der sogenannten Fuzzy-Logik definiert. Die Validität für den Einsatz letzterer wurde für positionsgeregelte Bewegungen ausgiebig untersucht. Zu kraftbasierten Bewegungen konnte jedoch keine Veröffentlichung im Rahmen der Recherche zu dieser Arbeit erfasst werden. Da zum einen nicht klar ist, ob und wie Nutzer solche unscharfen Parameter formulieren, und zum anderen, welchen Bewegungen sie entsprechen, ergeben sich also die folgenden wissenschaftlichen Fragestellungen:

F2 In wie weit und in welcher Form nutzen Anwender unscharf formulierte Parameter als Synonym für numerische Kraftwerte?

F3 In wie weit können unscharf formulierte Kraftparameter auf kraftbasierte Roboterbewegungen abgebildet werden?

Nicht nur vage bzw. unscharf formulierte, sondern auch mehrdeutige Instruktionen sind Bestandteil aktueller Forschung. Gerade bei referenzieller Mehrdeutigkeit werden oft Präferenzen zur Auflösung der Mehrdeutigkeit verwendet. Eine Nebenbedingung dieser Arbeit ist jedoch, dass Präferenzen der Nutzer nicht zur Verfügung stehen. Eine Alternative dazu bieten sogenannte Objekt-Affordanzen, welche die Fähigkeiten von Objekten für bestimmte Manipulationen darstellen (z.B. erreichbar, hebbar). Um den Nutzen der Affordanzen für eine Mehrdeutigkeitsauflösung zu erfassen, wurde folgende wissenschaftliche Fragestellung formuliert:

F4 In wie weit können Affordanzen zur Auflösung von Mehrdeutigkeiten in Instruktionen und zur Validierung von Instruktionen genutzt werden?

Da bei kraftbasierten Bewegungen je nach Anwendungsfall die entstehende Roboterbewegung selbst von Experten nicht immer vorhergesagt werden kann, ist eine vorhergehende Simulation einer Roboterbewegung eine Möglichkeit mögliche Schäden zu vermeiden. Der Einsatz von Simulationen wird schon zur Prüfung von Erreichbarkeiten von Objekten oder potentiellen Kollisionen eingesetzt. Meist werden dadurch optimale Bewegungen erzeugt oder Ausführungen gefährlicher Bewegungen vermieden. Der explizite Einsatz einer Simulation während der Instruktion zur Überprüfung verhältnismäßig kurzer Bewegungen inklusive einer Dialogkomponente zur Interaktion mit Nutzern im Problemfall wurde bisher noch nicht tiefergehend untersucht. Daher gilt es die letzte wissenschaftliche Fragestellung zu lösen:

F5 In wie weit kann eine instruierte kraftbasierte Bewegung online simuliert, analysiert und per Interaktion mit einem Nutzer validiert werden?

Die vorliegende Arbeit stellt Ansätze vor, welche sich mit den Fragestellungen F1 - F5 beschäftigen. Die Kombination dieser Ansätze ergibt ein Gesamtsystem, welches die Instruktion von Roboterarmen einfacher und sicherer macht und somit einen wissenschaftlichen Zugewinn bedeutet.

1.3 Abgrenzung

Diese Arbeit befasst sich mit der Transformation und Validierung von verbalen Instruktionen für kraftbasierte Roboterbewegungen. Da eine allgemeingültige Lösung dieser Aufgabe den Rahmen einer einzelnen Arbeit überschreiten würde, erfolgt nun eine Spezifikation der in dieser Arbeit umgesetzten Interaktion basierend auf in [Onnasch16] eingeführten Taxonomie (siehe Abbildung 1.2) zur Einordnung der Arbeit in die MRK, basierend auf die berücksichtigten Fehlertypen bezogen auf die Mensch-Roboter-Kommunikation [Honig18] und der untersuchten Validierungsebenen nach [Marge19].

Zunächst erfolgt die Einbettung des in dieser Arbeit vorgestellten Systems nach [Onnasch16] bezogen auf für die MRK relevante Eigenschaften der Interaktionsklassifikation, der Roboterklassifikation und der Teamklassifikation. Der Mensch kann in diesem Ansatz die Interaktionsrollen Operateur, Kooperateur und Kollaborateur einnehmen. Dabei kann der Nutzer das System sowohl als Operateur instruieren, als Kooperateur zusammen mit dem System eine Instruktion validieren oder als Kollaborateur gemeinsam mit dem Roboter an einem Werkstück arbeiten. Bei der Kollaboration nimmt der Nutzer hauptsächlich die Aufgabe des Fixierens wahr. Da ein gemeinsames Ziel verfolgt wird, ist die vorherrschende Interaktionsform ebenfalls als kooperativ einzustufen.

Die roboterbezogenen Kriterien beschreiben die Aufgabe, das Einsatzgebiet, die Morphologie und den Autonomiegrad des Roboters. Die Aufgaben des Roboters sind der Informationsaustausch, um Nutzer über auftretende Komplikationen zu informieren, und die Manipulation, in diesem Fall die Be- und Verarbeitung Bauteilen bzw. deren Oberflächen. Das Einsatzgebiet sind KMU, welche in [Onnasch16] nicht explizit berücksichtigt wurden. Die Morphologie ist funktional in Form eines 7-DoF Roboterarmes gegeben. Der Autonomiegrad wird in den

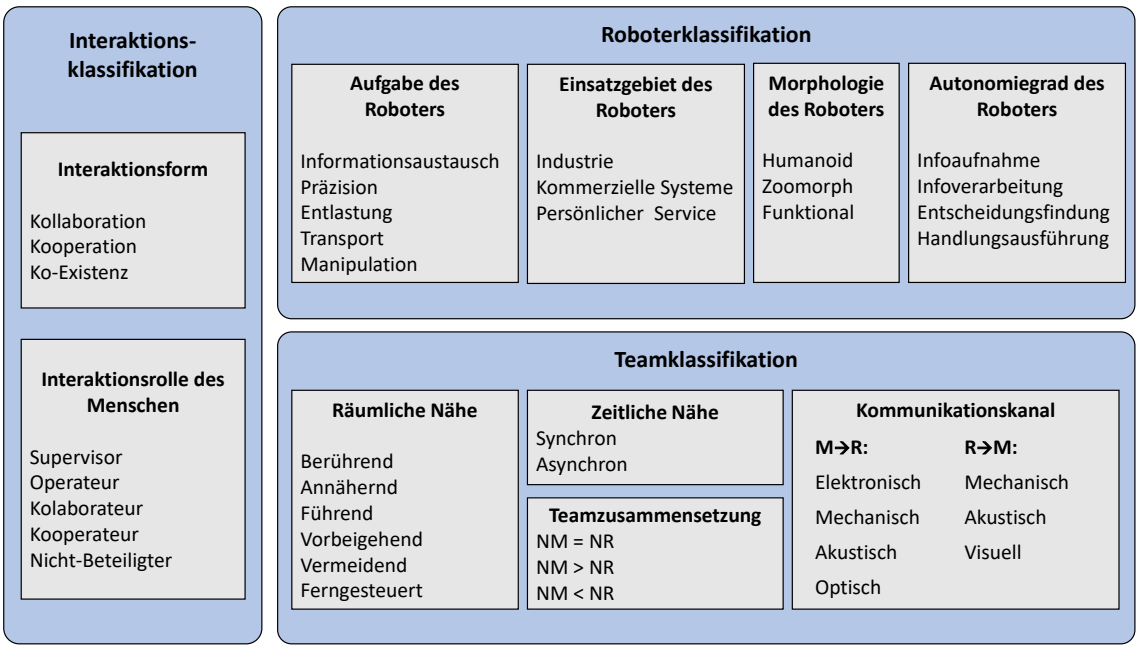


Abb. 1.2: Schematische Darstellung der in [Onnasch16] definierten Interaktions-Taxonomie der MRK.

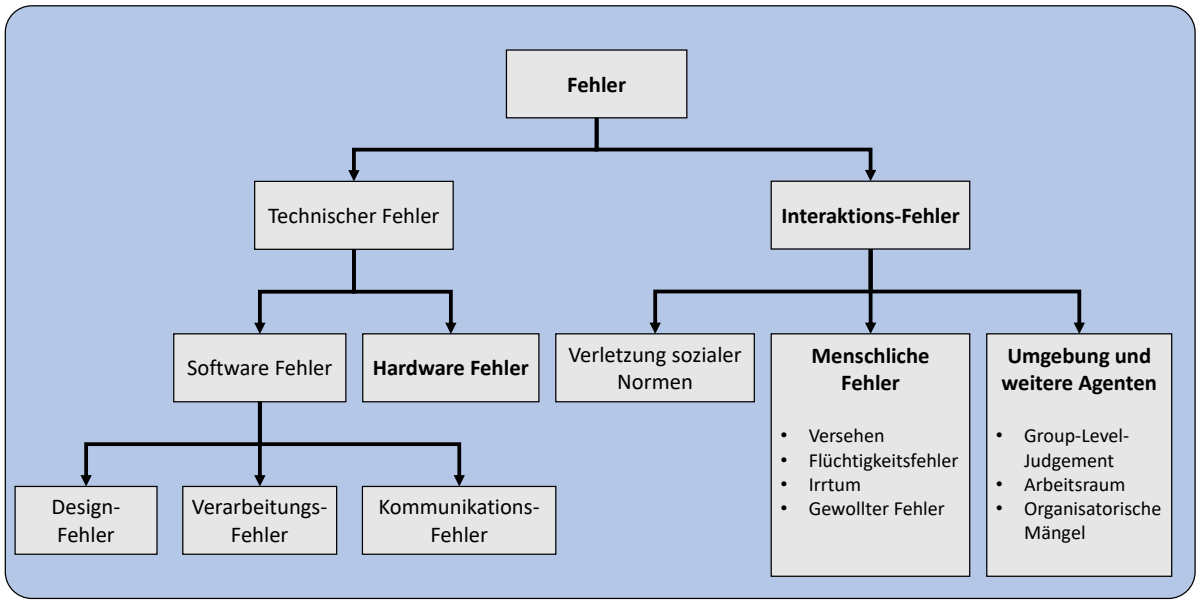


Abb. 1.3: Fehler-Taxonomie bezogen auf die Mensch-Roboter-Kommunikation nach [Honig18].

Bereichen Informationsaufnahme, Informationsverarbeitung, Entscheidungsfindung und Handlungsausführung von gering bis hoch eingestuft und stellt dar, in wie fern eine Intervention von Nutzern eine Rolle spielt. Die Informationsaufnahme erfolgt hier vollkommen autonom, die Verarbeitung und Entscheidung ist weder hoch, noch niedrig, da diese Aktionen im Problemfall zusammen mit den Nutzern durchgeführt werden. Die Ausführung ist hingegen wieder komplett autonom.

Die teambezogenen Kriterien spezifizieren den Kommunikationskanal, die räumliche Nähe, die zeitliche Nähe und die Teamzusammensetzung. Der Kommunikationskanal von Mensch zu Roboter findet in dieser Arbeit akustisch statt. Der Roboter kommuniziert hauptsächlich akustisch mit dem Menschen, auch wenn eine Komponente zur visuellen Kommunikation vorhanden, jedoch noch nicht ausgiebig getestet wurde. Zeitlich gesehen erfolgt die Interaktion asynchron, da instruierte Bewegungen während der Ausführung nicht angepasst werden können. Eine Ausnahme stellt dabei ein Abbruch der aktuell ausgeführten Bewegung dar. In den besprochenen Anwendungen entspricht die Anzahl der Nutzer der der Roboter. Die Anzahl der Nutzer kann allerdings auch erhöht werden. Die räumliche Zusammensetzung wird als vorbeigehend und ferngesteuert angenommen, da Nutzer das System sprachlich instruieren und unter Umständen einen benachbarten Arbeitsraum mit dem Roboter haben.

In [Honig18] wird eine Taxonomie zur Klassifikation von Fehlern bei der Mensch-Roboter-Kommunikation eingeführt, welche die Fehlerarten grundlegend in technische und Interaktionsfehler trennt. In dieser Arbeit werden Ansätze vorgestellt, welche sich mit einem Teil dieser Fehler auseinandersetzen. Auf der Seite der technischen Fehler werden Hardwarefehler in dem Sinne betrachtet, dass der zugrundeliegende Roboterarm bei der Validierung von Instruktionen berücksichtigt wird. Also beispielsweise, ob ein Umsetzen von einer Bewegung abhängig von der Steuerung oder auch der Kinematik möglich ist. Softwarefehler werden nicht näher betrachtet,

da der Schwerpunkt auf der Interaktion des Nutzers mit dem System liegt.

Auf der Seite der Interaktionsfehler liegt der Fokus speziell auf menschlichen Fehlern und Fehlern, welche durch die Umgebung und weitere Agenten entstehen. Verletzungen sozialer Normen, wie beispielsweise eine ungewöhnliche Formulierung der Rückmeldung werden hier nicht gezielt untersucht. Menschliche Fehler werden zudem in generelle Fehler, bewusste falsche Aktionen, Flüchtigkeitsfehler (Slips) in Form von Verwechslungen, Irrtümern, entstanden durch Unwissen oder zu wenig Aufmerksamkeit, und gewollte Fehler eingeteilt. Zwecks der Umgebung und weiteren Agenten liegt der Schwerpunkt auf sich ändernden Umgebungen und der eventuellen Unfähigkeit von Nutzern, Situationen oder Fähigkeiten des Systems einzuschätzen. Also in wie fern eine generelle Instruktion bezogen auf den aktuellen Anwendungsfall valide ist.

Eine Validierung bei einer verbalen MRK ist notwendig, da nicht davon ausgegangen werden kann, dass Nutzer den vollen Umfang der Fähigkeiten des System, also der Sprachverarbeitung und des Roboterarms, kennen. Welche Fehler dabei generell auftreten können, wird in [Marge19] über die sogenannten *Levels of Understanding* verdeutlicht (siehe Tabelle 1.1). Die Kanalebene beschreibt dabei das Erkennen von dem Zeitpunkt, an dem eine Instruktion vollkommen übergeben wurde (dem Endpunkt). Ein Endpunktfehler tritt dementsprechend dann auf, wenn der Roboter den Menschen unpassend unterbricht oder zu lange nach der Instruktion noch auf weiteren Input wartet. Die Aufgabe auf der Signalebene besteht daraus, die erfassten Signale korrekt auf Wörter abzubilden. Werden Wörter nicht oder falsch abgebildet, tritt hier ein Erkennungsfehler auf. Ein Fehler seitens des Nutzers stellt hier demnach eine Instruktion dar, welche nicht mit der zugrundeliegenden Grammatik aufgelöst werden kann. Die Aufgabe der Konversationsebene ist die Verarbeitung von Dialoghandlung in Form von Kommandos, Bestätigungen und Anfragen seitens der Nutzer. Kommunizieren Nutzer Dialoghandlungen außerhalb dieser vier Typen entsteht der Fehler: Außerhalb der Domäne. Auch hier geht der Fehler vom Nutzer aus. Die oberste Ebene evaluiert die bis dahin grammatikalisch und semantisch korrekt übergebene Instruktion basierend auf dem geltenden Kontext, also beispielsweise den aktuellen Arbeitsraum.

Nach [Clark96] sollte diese Ebene deshalb als gemeinsames Projekt angesehen werden, da der Roboter eine geeignete Bewegung planen soll und der Nutzer auf darauf achten soll, dass die Bewegung auch ohne Probleme ausgeführt wird. Der Fokus liegt in dieser Arbeit auf eben dieser Ebene und enthält zudem eine Erweiterung, indem die Bewegung vor der realen Ausführung in einer Simulation ausgeführt und evaluiert wird. Dadurch wird den Nutzern ein Teil der kogni-

Tab. 1.1: Verständnis-Ebenen im Mensch-Roboter-Dialog nach [Marge19].

Verständnis-Ebene	**Aufgabe**	**Parameter**	**Fehlertyp**
Projekt	Bewegungsplanung	Projekthandlung	Mehrdeutigkeit
Konversation	Interpretation	Dialoghandlung	Außerhalb der Domäne
Intention	Syntaxanalyse	Syntax	Grammatikfehler
Signal	Spracherkennung	Wortschatz	Erkennungsfehler
Kanal	Endpunkterkennung	Akustisches Signal	Endpunkt-Fehler

tiven Last abgenommen wird.

1.4 Kapitel-Übersicht

Um eine Grundlage für die Kapitel zu schaffen, welche das im Rahmen dieser Arbeit erstellte Gesamtsystem umsetzen, werden im nächsten Kapitel zugrunde liegende Konzepte und der dazugehörige Stand der Forschung vorgestellt. Dabei werden Ansätze zur Definition und Kombination von kraftbasierten Bewegungen eingeführt, ein Konzept zur Abbildung von unscharf formulierten Parametern auf numerische Werte präsentiert, ein Überblick zur Auflösung von Mehrdeutigkeiten gegeben und verschiedene Ansätze zur Validierung von Roboterbewegungen miteinander verglichen.

In Kapitel 3 wird das Grundkonzept näher beschrieben, auf welchem diese Arbeit aufbaut. Dazu zählt die genutzte Definition von kraftbasierten Bewegungen und die Erzeugung dieser aufgrund von sprachlichen Instruktionen und teilweise vordefiniertem Kontextwissen. Außerdem wird die hier verwendete Anpassung der Verarbeitung von Instruktionen erörtert.

Kapitel 4 befasst sich mit der Fragestellung F1. Es wird zunächst die Notwendigkeit einer Erweiterung des Grundkonzepts auf werkzeugabhängige Bewegungen und Kombinationen dieser motiviert. Der eingeführte Ansatz wird zudem im Rahmen einer Nutzerevluation untersucht und die Ergebnisse der Evaluation werden zusammengefasst und diskutiert.

Der Inhalt von Kapitel 5 beschäftigt sich mit den Fragestellungen F2 und F3 und gliedert sich somit in zwei Teile. Zur Beantwortung von F2 wird im ersten Teil ein Konzept eingeführt, mit welchem eine Erfassung der Art und Häufigkeit von kraftbasierten Instruktionsparametern ermöglicht wird. Im zweiten Teil wird ein Ansatz vorgestellt, durch den unscharfe Kraftparameter auf numerische Kraftwerte abgebildet werden können. Dies entspricht einer Antwort auf die Fragestellung F3. Die Nützlichkeit dieses Ansatzes wird über eine Prototyp-Anwendung gezeigt. Abschließend werden die Ergebnisse der beiden Teile in einer Zusammenfassung untersucht.

Kapitel 6 beschreibt einen Ansatz zur Beantwortung von Fragestellung F4. Zur Validierung bzw. zur Mehrdeutigkeitsauflösung von Instruktionen wird das Kontextwissen um eine eigene Affordanzdefinition erweitert. Nach einer detaillierten Erläuterung dieses Ansatzes wird dessen Nützlichkeit anhand einer Nutzerevaluation überprüft. Die gewonnenen Erkenntnisse werden abschließend dargelegt.

In Kapitel 7 wird letztlich ein Ansatz eingeführt, welcher sich mit der Fragestellung F5 beschäftigt. Der wissensbasierte Ansatz, der auf einer Physiksimulation aufbaut, wird zuerst erläutert und dann anhand einer Prototypevaluierung näher untersucht. In einer abschließenden Zusammenfassung setzt sich die Arbeit mit den Resultaten auseinander und es werden Vorschläge für eine Erweiterung des Systems für zukünftige Arbeiten vorgestellt.

In Kapitel 8 wird zunächst ein Prototyp vorgestellt. Dieser Prototyp stellt eine Implementierung der eingeführten Ansätze dar und ermöglicht dadurch eine Evaluierung des Gesamtsystems. Diese Evaluierung erfolgt in Form einer abschließenden Nutzerstudie, in der sowohl die Zufriedenheit der Nutzer mit dem Gesamtsystem gemessen wird, als auch Präferenzen der Nutzer

hinsichtlich der Rückmeldung des System erfasst werden.

Das letzte Kapitel 9 beinhaltet zum einen eine allgemeine Zusammenfassung dieser Arbeit und zum anderen einen Ausblick, welcher mögliche zukünftige Themen basierend auf dieser Arbeit zusammenfasst.

Kapitel 2

Grundlagen und Stand der Forschung

Die vorliegende Arbeit fordert ein Vorwissen in mehreren Disziplinen aus dem Bereich der Interpretation und Validierung von sprachlichen Instruktionen. Um ein Verständnis der in den jeweiligen Kapiteln vorgestellten Ansätze zu ermöglichen, werden deshalb im Folgenden die notwendigen Grundlagen eingeführt und der dazugehörige Stand der Forschung vorgestellt.

Zu Beginn wird der grundlegende Ablauf der sprachlichen Instruktion von Robotersystemen dargelegt, da dieser die Basis der vorliegenden Arbeit darstellt. Dabei werden verschiedene Spezialisierungen bezogen auf die Anwendungsbereiche und die Art der Instruktion vorgestellt. In den darauf folgenden Kapiteln werden diejenigen Komponenten des Ablaufs näher betrachtet, welche Bestandteil dieser Arbeit sind.

Der erste Teil beschäftigt sich mit der Abbildung von Instruktionen auf Roboterbewegungen. Zunächst wird das Fundament für Kapitel 3 und 4 gelegt, indem Definitionen von kraftbasierten Bewegungsprimitiven eingeführt werden. Im Fokus liegen dabei die sogenannten *Manipulationsprimitive*, da diese in dem Konzept verwendet werden, auf dem diese Arbeit aufbaut. Neben dem grundlegenden Ansatz werden außerdem Varianten und Konzepte zur Kombination von Manipulationsprimitiven vorgestellt. Danach werden die Konzepte des Wizard-of-Oz Experiments und die Grundlagen der Fuzzy-Logik näher erklärt, da diese in Kapitel 5 verwendet werden. Neben der generellen Einführung findet in beiden Fällen eine Übersicht über den Stand der Forschung statt. Bei den Wizard-of-Oz Experimenten erfolgt zudem eine Einführung von Kriterien, welche eine Klassifikation von dieser Art von Experimenten ermöglichen.

Der zweite Teil beschäftigt sich mit der Validierung bzw. der Entscheidungsfindung im Rahmen der Instruktion. Dafür wird zum einen das Konzept der Affordanzen eingeführt, da dies in Kapitel 6 neben der Validierung von Instruktionen auch zur Auflösung von Mehrdeutigkeiten genutzt wird. Außerdem erfolgt eine Übersicht zu Ansätzen der simulationsbasierten Validierung, welche in Kapitel 7 genutzt wird, um die Resultate von instruierten Bewegungen abschätzen und bewerten zu können.

In einer abschließenden Zusammenfassung werden die vorhandenen Lücken in den vorgestellten Bereichen identifiziert, mit denen sich die in dieser Arbeit eingeführten Ansätze beschäftigen.

2.1 Sprachbasierte Mensch-Roboter-Kommunikation

Um die Interaktion mit Robotern für Experten zu erleichtern und auch Nichtexperten zu ermöglichen, wird schon seit langem an Methoden geforscht, welche solch eine Interaktion auf verschiedenen Kommunikationskanälen realisieren. Der sprachliche Kommunikationskanal hat dabei den Vorteil, dass Menschen aus dem Alltag gewohnt sind, sprachlich zu kommunizieren, und dass Nutzern dadurch die Möglichkeit gegeben wird, andere Aufgaben parallel zu bearbeiten. Ein umfassendes Survey im Bereich der Sprachsteuerung von Robotern ist in [Liu19] dargestellt. Diejenigen Ansätze, welche den kompletten Ablauf einer Instruktion beschreiben, entsprechen dabei meist dem in Abbildung 2.1 dargestellten Aufbau, welcher an den in [Tellex20] präsentierten Aufbau angelehnt ist.

Demnach werden sprachliche Instruktionen über eine Spracheingabe-Komponente in Text umgewandelt und dann im Rahmen einer Sprachverarbeitung in eine semantische Repräsentation überführt. Die semantische Repräsentation kann dabei in Form eines Phrasen-Struktur-Baums oder auch einer Einbettung in eine Ontologie umgesetzt sein. Im nächsten Schritt erfolgt dann das sogenannte *Grounding* [Harnad90], welches einer Abbildung von Bestandteilen der semantischen Repräsentation auf dem System bekannte Strukturen entspricht. Dies umfasst Objekte im Arbeitsraum oder auch hinterlegte Bewegungsabläufe. Zusammen mit Informationen aus der Weltrepräsentation, welche unter Umständen mittels Perzeptionskomponenten, wie Kameras oder Kraftsensoren, aktualisiert wird, erfolgt die Erzeugung einer *Grounded Representation*. Diese Repräsentation stellt also eine Abbildung der Instruktion auf dem System bekannte Strukturen dar. Basierend auf dieser Repräsentation kann in der Entscheidungsfindung eine sprachliche Aktion, eine physikalische Aktion oder eine Kombination aus beiden erzeugt werden, welche dementsprechend an den Roboter und den Nutzer weitergeleitet werden kann.

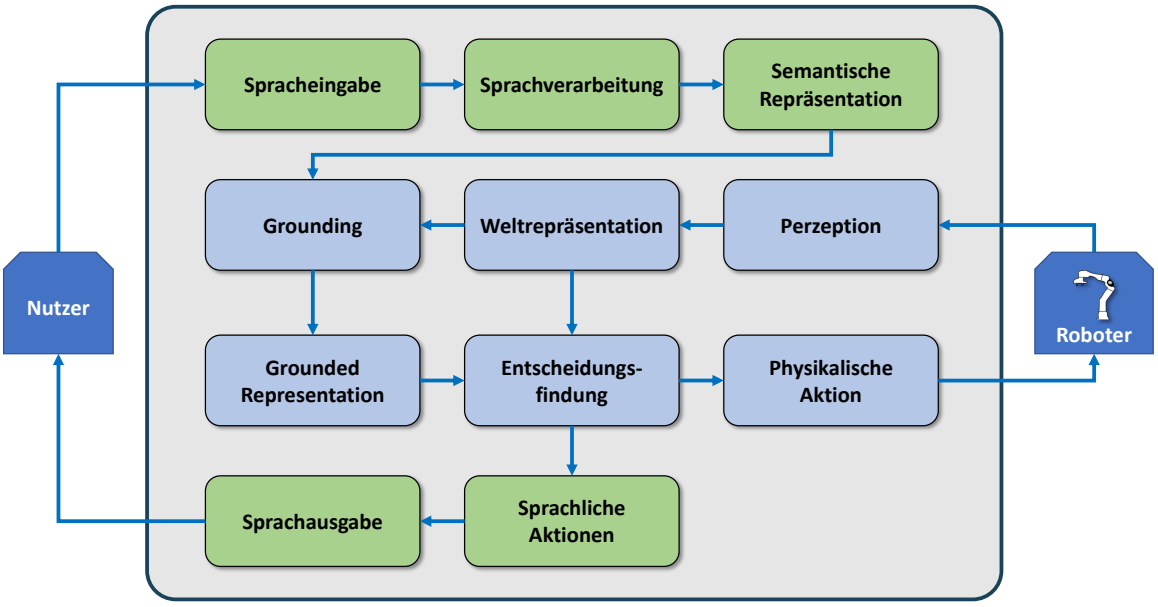


Abb. 2.1: Grundlegender Aufbau eines Systems zur sprachbasierten Kommunikation eines Roboters nach [Tellex20], bestehend aus sprachbezogenen Komponenten (grün) und systembezogenen Komponenten (blau).

Generell stellt jede der in Abbildung 2.1 dargestellten Komponenten einen eigenen Forschungsbereich dar. Die folgende Übersicht beschränkt sich jedoch ausschließlich auf den Teil des Aufbaus, welcher für diese Arbeit relevant ist. Dazu gehören die Struktur der Spracheingabe, die Transformation von Instruktionen auf Roboterbewegungen inklusive der jeweiligen Anwendungsgebiete und die Formen der Rückmeldungen des Systems an den Nutzer.

Der Aufbau von Instruktionen kann generell in Form von drei Ausprägungen auftreten: Strukturiert, semi-struktieriert und unstrukturiert [Aggarwal12]. Ansätze basierend auf strukturierten Anweisungen ([Knoll97], [Bugmann05]) gehen davon aus, dass ein festes Schema besteht, mit Hilfe dessen gültige Instruktionen erkannt und weiterverarbeitet werden können. Die Implementierung eines solchen Systems ist zwar verhältnismäßig leicht, schränkt den Nutzer dabei jedoch in der Bedienung des Systems stark ein. Dieses Problem wird über den Einsatz von semi-strukturierten Anweisungen ([Tellex13], [Spangenberg17], [Stenmark17]) entschärft. Dort wird kein festes Schema, sondern eine Struktur definiert, nach der Anweisungen aufgebaut sein müssen. Unstrukturierte Anweisung wären letztliche jene, welche keinerlei Einschränkung bezogen auf ihren Aufbau besitzen. Unter den im Rahmen der Recherche für diese Arbeit erfassten Arbeiten konnte kein Ansatz erfasst werden, welcher in diese Kategorie fällt.

Bezogen auf die abbildbaren Bewegungen kann man Ansätze grob in jene einteilen, welche eine Instruktion von Grobbewegungen erlauben, und jene, die Feinbewegungen erlauben. Grobbewegungen bezeichnen dabei Bewegungen, welche ohne Rückmeldung eines Sensors ausführbar sind. Ansätze aus diesem Bereich sind oft im Anwendungsgebiet der mobilen Robotik angesiedelt ([Kollar14], [Matuszek13]) oder setzen Pick-and-Place Aufgaben um ([Misra16], [Boteanu16]). Feinbewegungen stellen hingegen Bewegungen dar, welche zudem Sensorrückmeldungen während der Ausführung berücksichtigen. Anwendungen gibt es dabei sowohl im Haushaltsbereich ([Tenorth10], [Bollini13]), als auch bei Montage-Aufgaben ([Knoll97], [Stenmark15]). In den meisten Fällen erfolgt die Zuweisung bzw. Parametrierung von vordefinierten Bewegungen, ohne einen weiteren Zusammenhang zu liefern. Der in [Spangenberg17] erarbeitete Ansatz schließt diese Lücke, indem physikalische Gesetze bei der Parametrierung genutzt werden. Da hier auf diesem Ansatz aufgebaut wird, erfolgt eine nähere Beschreibung des Konzepts in Kapitel 3.

In manchen Fällen ist eine Rückmeldung an den Nutzer notwendig, da beispielsweise ungültige Eingaben oder Mehrdeutigkeiten auftreten. Übersichten über Ansätze, welche sich mit solchen Fällen beschäftigen, sind in [Honig18] und [Marge19] gegeben. Neben der Erkennung von Mehrdeutigkeiten spielt auch die Art der Auflösung von Mehrdeutigkeiten eine große Rolle bei der Intuitivität von Systemen. Daher ist sie schon seit geraumer Zeit Bestandteil der Forschung ([Liu10], [Lemaignan17], [Marge19], [Williams19]). Die meisten Ansätze versuchen dabei Mehrdeutigkeiten über die Nutzung von Rückfragen aufzulösen. Die Rückfragen bestehen in diesen Fällen aus Ja/Nein Fragen [Deits13], [Hemachandra14], einer Auflistung aller Möglichkeiten [Marge15], oder generischen W-Fragen [Tellex13], mit denen der mehrdeutige Parameter erfragt wird.

2.2 Definition und Kombination von kraftbasierten Roboterbewegungen

Um Nutzern eine Definition von Roboterbewegungen zu erleichtern, ist diese seit längerem nicht mehr nur im Gelenkraum oder Arbeitsraum, sondern auch im Aufgabenraum möglich [Zieliński95]. Im Aufgabenraum werden dabei Bewegungen nicht wie im Gelenkraum, über eine Konfiguration $\Theta \in \mathbb{R}^n$ mit n = Anzahl der Gelenke, oder wie im Arbeitsraum, über eine Endeffektorpose $T \in \mathbb{R}^{4\times4}$ bezogen auf ein globales Koordinatensystem, definiert. Stattdessen definiert man sie im Bezug auf ein lokales Koordinatensystem, dem sogenannten *Taskframe* $\mathcal{TF}$ [Mason81]. Dies ermöglicht eine flexiblere Definition von Bewegungen, da sie beispielsweise abhängig von der Lage eines Objekts im Raum definiert werden können. So lässt sich beispielsweise das Öffnen einer Tür relativ leicht beschreiben, indem der Taskframe $\mathcal{TF}_{\text{Tür}}$ so in den Angeln der Tür positioniert wird (siehe Abbildung 2.2, links), dass ein Öffnen als Drehung um eine der Achsen dargestellt werden kann.

Erweitert man diese Repräsentation um die Möglichkeit für jede Raumrichtung eine eigene Regelstrategie anzugeben, erhält man letztlich den *Task-Frame-Formalismus* [Bruyninckx96]. Darüber ist eine Angabe von Bewegungen darstellbar, welche sowohl positionsgeregelt als auch kraftgeregelt sein können. Eine formale Definition, welche diesen Ansatz umsetzt, ist in Form von Hybriden Bewegungen gegeben, welche in [Finkemeyer10] folgendermaßen definiert werden:

$$\mathcal{HM} = (\mathcal{TF}, \mathcal{D})$$
$$\mathcal{D} := \{(v_i, t_i) | i \in 1, \ldots, 6\}$$

Hier entspricht $\mathcal{D}$ einer sogenannten *Adaptiven Selektionsmatrix*, welche für jeden Freiheitsgrad eine Kombination der geforderten Regelstrategie t_i und den jeweiligen Sollwerten $v_i \in \mathbb{R}$ enthält. Somit lässt sich das Schrauben einer Schraube umsetzen, indem ein Taskframe $\mathcal{TF}_{\text{Schraube}}$ wie in Abbildung 2.2, Mitte, definiert wird, und eine Kraftregelung entlang der Z-Achse, sowie eine Drehung um die Z-Achse angegeben wird.

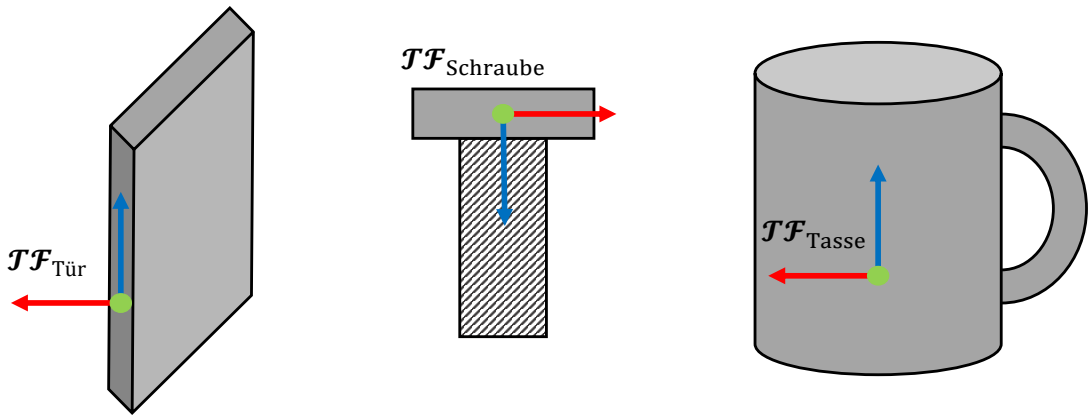


Abb. 2.2: Beispiele für ein Taskframes zur Beschreibung einer Drehung einer Tür (links), Manipulation einer Schraube (mittig) und einer erlaubten Drehung einer Tasse (rechts).

Um zudem ausgerüstete Werkzeuge nutzen zu können und die Ausführung der Bewegungen robuster zu machen, wurde dieses Konzept zu den sogenannten Manipulationsprimitiven $\mathcal{MP}$ [Finkemeyer04] erweitert:

$$\begin{aligned}\mathcal{MP} &= (\mathcal{HM}, \tau, \lambda).\\ \tau &:= \{\tau_i | \tau_i = \{werkzeugname, werkzeugkommando\}\}\\ \lambda &:= S \rightarrow \{true, false\}\end{aligned}$$

Dabei werden über τ Werkzeugkommandos, wie das Öffnen des Greifers, definiert und über λ Abbruchkriterien eingeführt. Diese Kriterien enthalten Bedingungen S, welche Grenzwerte für die jeweiligen Einträge der Freiheitsgrade festlegen. So lässt sich im Vergleich zu einer Definition im Arbeitsraum leicht definieren, dass eine Tasse nur entlang der lokalen Z-Achse gedreht werden darf, indem man den Taskframe entsprechend zu der Tasse modelliert (siehe Abbildung 2.2, rechts). Dadurch können Fehlerfälle, wie zum Beispiel das versehentliche Umkippen der Tasse, modelliert werden, welche zu einem Abbruch der Bewegung führen.

Da gerade für komplexere Bewegungen eine Kombination solcher MP notwendig ist, wurde außerdem das Konzept der Manipulationsprimitivnetze ($\mathcal{MPN}$) [Thomas09] eingeführt. Diese enthalten neben einem Start- und einem Stopp-MP, die Menge aller enthaltenen MP, sowie eine Menge K, welche Verknüpfungen zwischen den jeweiligen MP beinhaltet. Formal lässt sich ein MPN folgendermaßen darstellen:

$$\mathcal{MPN} := (MP, MP_{start}, MP_{stop}, K). \tag{2.1}$$

Ein Framework, welches eine Kombination von kraftbasierten Bewegungen ähnlich zu dem in dieser Arbeit eingeführten Ansatz erlaubt, ist in [Thomas13] über die domänensspezifische Sprache *LightWorks* eingeführt worden. Darin können sogenannte *Elementare Aktionen* (EA), welche Manipulationsprimitiven entsprechen, sequentiell zu komplexeren Bewegungen bzw. *Skills* verknüpft werden. Die EA können dabei sowohl Roboterbewegungen, als auch Werkzeugoperationen beschreiben. Ein Unterschied zu MP ist, dass für EA zunächst Startkonditionen gelten müssen, damit die Ausführung dieser Bewegung beginnt. Dies kann beispielsweise der Abschluss einer anderen Bewegung sein. Das Beenden einer Bewegung erfolgt über weitere Konditionen, welche bei einem Schrauben zum Beispiel über ein maximales Moment definiert werden können. Eine parallele Verknüpfung von EA ist nicht explizit möglich, da jede EA separat von der Steuerung verarbeitet wird. Solch eine Kombination muss daher explizit in einem einzelnen EA modelliert werden.

In [Weidauer14] wurde ein Ansatz eingeführt, welcher eine Verknüpfung von Bewegungsprimitiven über Petri-Netze in Form von Stellen-Transitions-Netzen $\mathcal{PN}$ [Desel96] ermöglicht. Jede Stelle entspricht in diesem Netz einem sogenannten *Manipulation Task* ($\mathcal{MT}$), welcher wie folgt definiert ist:

$$\mathcal{MT} := \{\alpha, \{\mathcal{HM}|\mathcal{PN}|\emptyset\}, \rho, \tau, \omega\} \tag{2.2}$$

Er entspricht also entweder einer Hybriden Bewegung $\mathcal{HM}$, einem Stellen-Transitions-Netz $\mathcal{PN}$ oder einem Warten $\emptyset$. Außerdem enthalten MT Startkonditionen α, eine Menge an Werkzeugkommandos τ und eine Menge an Abbruchbedingungen ω. Die hybriden Bewegungen beinhalten hier zudem das Gerät, welches die entsprechende Bewegungen ausführen soll. Dies erlaubt nicht nur eine sequentielle und damit zyklische Verknüpfung einzelner Komponenten für einen Roboterarm, sondern auch eine parallele Ausführung von Bewegung bei Mehrroboter-Systemen oder auch einer Mensch-Roboter-Kooperation.

Dieser Ansatz wurde in [Pek16] um die Synchronisationsprimitive *Barrier* und *Range* erweitert. Durch die *Barrier* wird dabei ein Setzen von willkürlichen Synchronsationspunkten ermöglicht und damit Synchronisationen unabhängig von Start- und Endpunkten einzelner Tasks erlaubt. Bei der *Range*-Synchronisation werden zwei Bewegungen so umgesetzt, dass sie den identischen zeitlichen Endpunkt besitzen. Um diese Komponenten umsetzen zu können, wurde die MT-Definition so angepasst, dass die Stellen-Transitions-Netze durch einen Zustandsautomaten ersetzt wurden. Die Gerätedefinition wurde von den Hybriden Bewegungen gelöst und direkt in den MTs definiert. Innerhalb von MTs wurde das Wechseln von Zuständen zudem über sogenannte *Switches* umgesetzt.

Neben Ansätzen zur Kombinationen von Bewegungsprimitiven auf Basis von Manipulationsprimitiven bzw. deren Varianten existieren noch eine Vielzahl weiterer Ansätze, wie die *Semantic Event Chains* [Aein13] oder auch *Behaviour Trees* [Guerin15]. Da diese nicht im Fokus dieser Arbeit liegen, werden sie nicht näher erläutert.

2.3 Wizard-of-Oz Studien

Der erste Abschnitt dieses Kapitels enthält eine kurze Übersicht über das Grundkonzept des Wizard-of-Oz (WoZ) Experiments. In der zweiten Hälfte wird ein Kriterienkatalog aus der Literatur übernommen, welcher eine Klassifikation von WoZ Experimenten ermöglicht.

Der grundlegende Gedanke bei WoZ Experimenten ist, dass ein Nutzer mit einem System interagiert, von dem er denkt, dass es autonom handelt. In Wirklichkeit sitzt jedoch ein sogenannter Wizard im Hintergrund und steuert entweder einen Teil des Systems oder sogar das komplette System. Dieses Konzept erfreut sich einer großen Beliebtheit, da so bereits in der frühen Entwicklungsphase Systeme evaluiert werden können, ohne zunächst einen Prototyp entwickeln zu müssen. Eine umfassende Übersicht von insgesamt 54 Veröffentlichungen, welche sich mit WoZ Experimenten und deren Erweiterungen befassen, ist in [Riek12] dargestellt.

Eine Eigenschaft, nach der Veröffentlichungen in dieser Übersicht klassifiziert werden, ist der *Wizard Control Type*, welcher beschreibt, welche Komponente des Systems bezüglich verbaler Interaktion, non-verbaler Interaktion, Navigation, Manipulation, Sensorik oder Mapping des Systems vom Wizard beeinflusst wird. In den letzten Jahren lag der Schwerpunkt dabei bei Ansätzen, welche sich mit der verbalen Komponente auseinandergesetzt haben. Zudem wurden nicht-verbale Komponenten, wie die Mimik oder Gestik, näher betrachtet und auch Navigationsaufgaben häufig von einem Wizard durchgeführt, was beispielsweise über einen Controller leicht umzusetzen ist. Zu den weniger betrachteten Komponenten zählen hingegen die Manipulation

[Ralph08], die Sensorik [Yamaoka09], und das Mapping [Fischer11].

Neben dem Wizard Control Type wurden in [Riek12] außerdem die folgenden vier Kriterien für eine Klassifikation von WoZ Experimenten eingeführt: Die Kelley Kriterien [Kelley84], die Fraser und Gilbert Kriterien [Fraser91], die Green Kriterien [Green04] und die Steinfeld Kriterien [Steinfeld09].

Das Kelley Kriterium ($C1$) stellt dar, in welchem Ausmaß der Wizard das zu evaluierende System kontrolliert. In iterativen Anwendungen von WoZ Experimenten für ein System kann mit diesem Kriterium erfasst werden, in wie weit Komponenten noch simuliert werden und in wie weit sie bereits umgesetzt wurden. Die Fraser and Gilbert Kriterien ($C2$) wurden eingeführt, um Vorbedingungen zu spezifizieren, welche eine Erfassung der Nützlichkeit einer WoZ Studie ermöglichen. Diese Vorbedingungen bestehen daraus, dass es möglich sein muss das zukünftige System unter menschlichen Einschränkungen zu simulieren, das Verhalten des zukünftigen Systems beschreiben zu können, und dass eine überzeugende Simulation des Systems möglich ist. Die Green Kriterien ($C3$) dienen zur Verbesserung des Experiments durch die Spezifikation von *Nutzerinstruktionen*, welche Nutzern vorgeben, wie sie mit dem System interagieren können; *Verhaltenshypothesen* der Nutzer, welche die Erwartungen des Versuchsleitenden an das Nutzerverhalten darstellen; und *Roboterverhalten*, welche definieren, wie sich der Roboter bei gegebenen Instruktionen verhalten soll. Die Steinfeld Kriterien ($C4$) ermöglichen letztlich eine Klassifikation von WoZ Experimenten basierend auf der mensch- oder roboterzentrierten Auslegung. Die verschiedenen Typen sind dabei Wizard-with-Oz (Nutzung realer Technologie in einem simulierten oder kontrollierten Umfeld), Wizard-and-Oz (reale Technologie in einem realen Umfeld), Oz-of-Wizard (der Nutzer wird simuliert) und Oz-with-Wizard (das Verhalten des Systems wird gemessen und nicht das Verhalten der Nutzer).

Das WoZ Experiment, welches dem Ansatz in dieser Arbeit am nächsten kommt, da der Control Type ebenfalls der Manipulation entspricht, wurde in [Ralph08] durchgeführt. Die Intention hinter dieser Studie war, Informationen über die Instruktion von greiferbezogenen Aufgaben zu erlangen. Dafür sollten Nutzer einen Roboterarm inklusive Greifer dazu instruieren eine Reihe an Objekten zu greifen. Der Wizard hat den Roboterarm dabei über eine graphische Nutzeroberfläche bewegt und somit das komplette System gesteuert. Die in C2 enthaltenen Punkte waren komplett erfüllt und nach C3 wurden dem Nutzer eine Reihe an Instruktionen bzw. Strukturen vorgegeben, welche genutzt werden konnten. Im Bezug auf C4 entspricht dies einem Wizard-and-Oz Experiment, da sowohl reale Technologie, als auch ein reales Umfeld verwendet wurde.

2.4 Fuzzy-Logik

Eine Herausforderung bei der Abbildung von symbolischen Informationen auf subsymbolische Informationen ist eine unscharfe Angabe von Parametern. Beispiele dafür sind Angaben wie: *ein wenig nach links* oder *fahre schnell*. Um solche Formulierungen interpretieren zu können, hat sich die Methode der *Fuzzy-Logik* [Bothe95], [Zadeh65] mittlerweile in einer Reihe an Anwendungsfällen bewährt. Der Grundgedanke hierbei ist, dass man relevante numerische Parameter

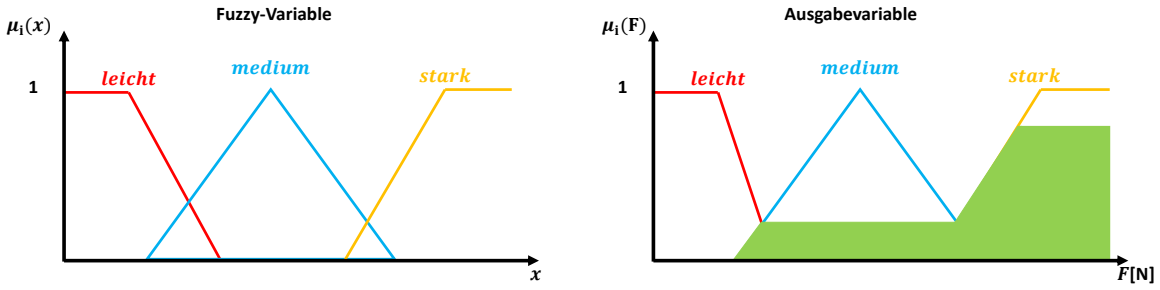
zunächst in eine unscharfe Darstellungsform überführt (*Fuzzifizierung*), diese dann über sogenannte *Fuzzy-Regeln* evaluiert (*Logisches Schließen* bzw. *Inferenz*) und letztlich den Ausgabeparameter aus einer unscharfen Darstellung in einen numerischen Wert umwandelt (*Defuzzifizierung*).

Grundlegend entspricht die Fuzzy-Logik laut [Schmidt11] einer unscharfen Erweiterung der scharfen Prädikatenlogik in der Form, dass Prädikate nicht mehr entweder wahr oder falsch sein können, sondern über Zugehörigkeitsgrade im Intervall [0,1] verfügen. Systemrelevante Eigenschaften, wie beispielsweise die Objekthärte oder die Temperatur, werden über sogenannte *Fuzzy-Variablen* V_l modelliert. Jede dieser Variablen enthält eine Menge an *Fuzzy-Termen*, welche unterschiedliche Ausprägungen, wie leicht oder stark, der damit beschriebenen Eigenschaft darstellen. Für jeden Fuzzy-Term wird zudem eine Zugehörigkeitsfunktion $\mu : \mathbb{R} \rightarrow [0, 1]$ angegeben, welche den Wertebereich eines scharfen Eingabewertes auf das Intervall [0,1] abbildet, und damit angibt, wann ein Fuzzy-Term mit welcher Ausprägung auftritt (Abbildung 2.3a). Die Zugehörigkeitsfunktionen können dabei beispielsweise als Dreiecks- oder Trapezoid-Funktion modelliert werden und sich auch überschneiden.

Im Rahmen der Fuzzyfizierung werden die Zugehörigkeitsfunktionen basierend auf dem jeweiligen scharfen Eingangswert ausgewertet und pro Fuzzy-Variable und Eingangswert in einem Zugehörigkeitsvektor $Z \in [0, 1]^n$ abgelegt. Würde für den in Abbildung 2.3a dargestellten Fall ein scharfer Eingabewert fuzzyfiziert werden, welcher bei 0,3 liegt, so würde der entsprechende Zugehörigkeitsvektor folgendermaßen aussehen:

$$Z(0, 3) = (0, 5, 0, 5, 0).$$

Die Hauptaufgabe der Inferenz ist die Beschränkung und Verknüpfung der Fuzzy-Terme der Ausgangswerte zu einer einzelnen Zugehörigkeitsfunktion, welche im Rahmen der Defuzzyfizierung für die Berechnung eines scharfen Ausgabewertes benötigt wird. Diese Beschränkung geschieht auf Basis der entsprechenden Zugehörigkeitsvektoren und vordefinierten Fuzzy-Regeln, welche das für den jeweiligen Anwendungsfall nötige Expertenwissen darstellen. Sie stellen somit Relationen zwischen eingehenden und ausgehenden Fuzzy-Variablen dar, indem ausgehende



(a) Dreieckszugehörigkeitsfunktionen einer Fuzzy-Variable.

(b) Beschränkung und disjunkte Verknüpfung der Fuzzy-Terme einer Fuzzy-Variable.

Abb. 2.3: Beispiele einer Fuzzy-Variable und deren Fuzzy-Termen (a) und einer disjunkten Verknüpfung von beschränkten Fuzzy-Termen, visualisiert durch die grüne Fläche (b).

Fuzzy-Terme aus UND- und ODER-Verknüpfungen von eingehenden Fuzzy-Termen gefolgert werden. Ein Beispiel für eine solche Regel ist folgende:

$$\text{FALLS Eingabe1 = leicht UND Eingabe1 = stark DANN Ausgabe = stark,} \tag{2.3}$$

wobei Eingabe1, Eingabe2 und Ausgabe $\in V_l$ linguistische Variablen sind, welche die unscharfen Werte der Ein- und Ausgabeparameter beschreiben. Ein Vorteil dieser Regeln ist dabei, dass sie selbst für Nichtexperten verständlich sind, da sie natürlichsprachlich definiert werden. Die Aktivierung der jeweiligen Fuzzy-Terme der Ausgangsgröße geschieht wie folgt: Ist lediglich ein Eingabewert vorhanden, wird die Zugehörigkeit dieses Werts genutzt, um die Zugehörigkeitsfunktion des Ausgangsterms zu beschränken. Sind jedoch wie in Gleichung 2.3 mehrere Eingabewerte vorhanden, so muss daraus ein Wert generiert werden. Laut [Schmidt11] werden in diesem Fall je nach Verknüpfung die in Tabelle 2.1 dargestellten Umrechnungen durchgeführt. Wenn alle Regeln ausgewertet wurden, findet zuletzt eine disjunktive Verknüpfung der jeweils beschränkten Terme zu einer Zugehörigkeitsfunktion μ_{res} statt (siehe Abbildung 2.3b), welche in der Defuzzyfizierung zur Berechnung eines scharfen Wertes genutzt wird.

Der letzte Schritt, die Defuzzyfizierung, bestimmt aus μ_{res} einen scharfen Wert. Hier gibt es entweder die Möglichkeit, den resultierenden Wert anhand von Extremwerten von μ_{res} zu bestimmen (Max-Methode, Links-Max-Methode, Rechts-Max-Methode, Mean-Max-Methode) oder den Wert über den Schwerpunkt der Fläche unter μ_{res} zu bestimmen (Schwerpunkt-Methode und Alpha-Schwerpunkt-Methode). Bei der Max-Methode wird derjenige y-Wert gewählt, welcher mit dem maximalen Funktionswert korrespondiert. Für den also gilt: $y := max\{\mu_{\text{res}}(y)|y \in Y\}$. Ist dieser Wert nicht eindeutig, entsteht dadurch also eine Menge an Kandidaten $Y_{\max}$, so bleiben die Alternativen der Links-Max-Methode, durch welche der kleinste Werte über $y := min(Y_{\max})$ bestimmt wird; der Rechts-Max-Methode, bei der der größte Werte mittels $y := max(Y_{\max})$ berechnet wird; und die Mittelwert-Max-Methode, welche den Mittelwert der Werte über $y := \sum_{i \in 1,\ldots,m} \frac{y_i}{m}$ berechnet. Im Gegensatz dazu bestimmen die Schwerpunkt-Methoden den Schwerpunkt der Fläche unter μ_{res} zur Bestimmung des scharfen Wertes. Die Alpha-Schwerpunkt-Methode erlaubt die Angabe einer unteren Schranke $\alpha \in [0, 1]$ von μ_{res} zur Rauschunterdrückung. Welche dieser Methoden man wählt, hängt vom Anwendungsfall ab.

Eine Anwendung der Fuzzy-Logik findet in vielen Fällen in der Regelungstechnik statt, da sich damit Zusammenhänge von Systemparametern aufgrund der natürlichsprachlichen Formulierung von Fuzzy-Regeln verständlich darstellen lassen. Doch auch im Bereich der sprachbasierten Instruktion wurden mittlerweile eine Reihe an Anwendungen für die Abbildung von

Tab. 2.1: Möglichkeiten zur Kombination von Aktivierungsgraden im Rahmen der Inferenz für die Werte m und n.

Name	**t-Norm AND**	**s-Norm OR**
Minimum, Maximum	min(n,m)	max(n, m)
Algebraisches Produkt, Summe	m · n	n + m - n · m
Beschränkte Differenz, Summe	max(0, n + m - 1)	min(1, n + m)

unscharfen Parametern auf Basis der Fuzzy-Logik entwickelt. Ein umfassender Überblick dazu wurde in [Muthugala18] veröffentlicht, wobei nicht nur die bereits umgesetzten Parametertypen dargestellt wurden, sondern auch auf jene hingewiesen wurde, welche bisher noch nicht näher betrachtet wurden. Dazu gehört beispielsweise eine während der Ausführung wirkende Kraft (Abbildung 2.4).

Bereits untersuchte Parameter werden nun näher erläutert. Bezogen auf Objekte im Arbeitsraum wurden größenbezogene Parameter, wie "groß", "lang" oder "klein", umgesetzt [Jayawardena06], [Muthugala14]. Außerdem wurden zur Beschreibung der Umwelt Distanzangaben näher untersucht [Schiffer12], [Jayasekara09], [Muthugala16]. Dies hat zu Ansätzen geführt, welche eine Abbildung von relationalen Distanzangaben, wie "nah" oder "fern", auf scharfe Werte umgesetzt haben. Bezogen auf den Roboter wurden zudem bereits Ansätze zur Abbildung geschwindigkeitsbezogener Parameter eingeführt, welche Parameter wie "schneller" bzw. "langsamer" in Navigationsanwendungen [Pulasinghe04], [Lin98] umsetzen. Neben richtungsbezogenen Parametern, wie "etwas nach links", [Skubic04] wurden auch Gelenkkonfigurationsparameter bei der Instruktion eines Roboterarms [Jayawardena07], [AG10] untersucht.

Die Interpretation der unscharfen Informationen in solchen Instruktionen basiert meistens entweder auf dem Nutzen für unterschiedliche Anwendungsfälle (zum Beispiel der Position von Objekten im Arbeitsraum) oder auf der Anpassung von bereits übergebenen Instruktionen, auch bekannt als *Coach-Player-System*. In solchen Fällen werden lediglich kurze Instruktionen wie "etwas mehr" oder "weiter" übergeben, um ein gewünschtes Resultat zu erzielen. Der Vorteil von ersterem ist, dass es schneller zur Ausführung durch den Roboter kommt, wobei jedoch möglicherweise keine für den aktuellen Nutzer perfekten Ergebnisse geliefert werden. Bei letzterem verhält es sich genau entgegengesetzt.

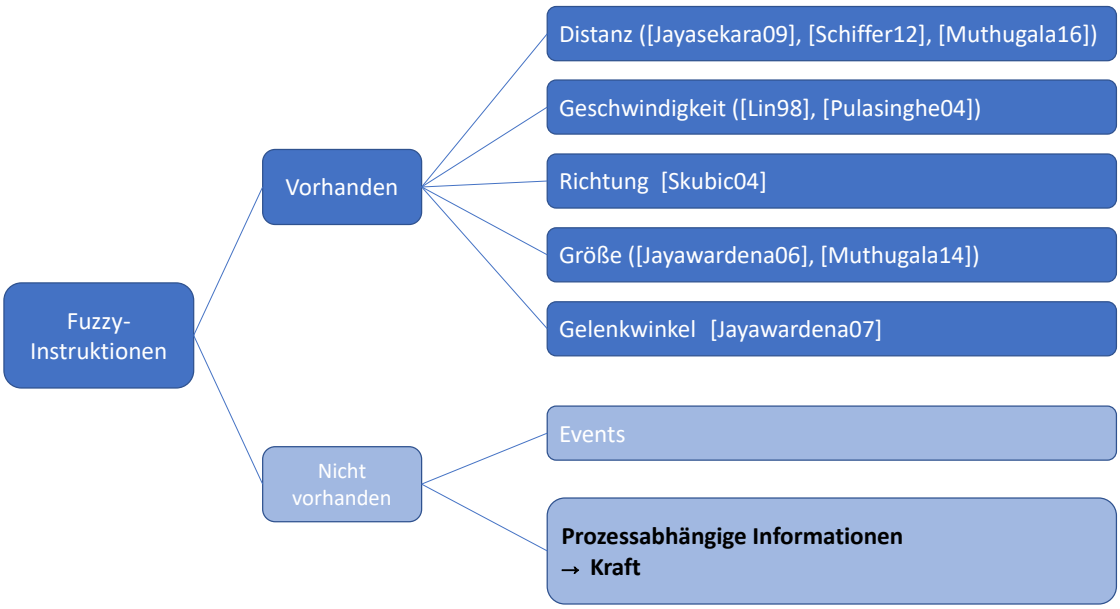


Abb. 2.4: Übersicht über die bisher publizierten Ansätze zur Abbildung von unscharfen Parametern basierend auf der Fuzzy-Logik nach [Muthugala18].

Für kraftbasierte Parameter wurde bisher laut [Muthugala18] noch keiner dieser Ansätze umgesetzt, daher wird hier ein Ansatz vorgestellt, welcher eine Lösung des Problems darstellt.

2.5 Affordanzen

Ein Problem bei der Instruktion von Robotersystemen durch Nichtexperten ist, dass ein Bewusstsein über Roboterfähigkeiten und deren Grenzen nicht vorausgesetzt werden kann. Dies führt unter Umständen zu fehlerhaften Instruktionen, welche wiederum ungewollte Roboterbewegungen zur Folge haben, durch die das Vertrauen in das Robotersystem sinkt. Um dies zu vermeiden, werden Mechanismen benötigt, welche Nutzer während der Instruktion unterstützen. Beispiele dafür sind die Validierung übergebener Parameter auf ihre Ausführbarkeit für einen Weltzustand oder auch Komponenten zur Erfassung und Auflösung von Mehrdeutigkeiten.

Ein Konzept was sich, gerade im Bezug auf Fähigkeiten, für eine Validierung anbietet, sind die in [Gibson77] eingeführten *Affordanzen* $\mathcal{A}$. Dieses Konzept beschreibt, was mit einem Objekt gemacht werden kann bzw. was nicht mit einem Objekt gemacht werden kann [Min17]. Formal wird das Konzept dabei meist als Zusammenhang eines Objekts, einer Handlung mit diesem Objekt und einem daraus resultierenden Effekt definiert, wie beispielsweise in [Montesano07]. In [Cruz16] wurde dieses Konzept, auf *Konzeptualisierte Affordanzen* erweitert, indem zusätzlich der Zustand des Roboters berücksichtigt wird. Dadurch ist es möglich, in einem System zu hinterlegen, welche Interaktionen mit Objekten in seiner Umgebung möglich sind.

Im Bereich der Robotik wurde dieses Konzept bereits für eine Vielzahl an Anwendungen umgesetzt, was auch zu einer Erstellung von einigen Übersichten geführt hat. So wurde der Einsatz von Affordanzen im Bereich der Psychologie, den Neuro-Wissenschaften und der Robotik miteinander verglichen [Jamone16], Anwendungen für die Mensch-Roboter-Interaktion wurden zusammengefasst [Moratz08], [Ardón20] und Erkenntnisse im Bereich der entwicklungsbasierten Robotik wurden diskutiert [Min16]. Beispiele für bisher umgesetzte Affordanzen sind dabei die *Greifbarkeit* [Song15], [Detry11], die *Befahrbarkeit* ([Carvalho16], [Kostavelis12], [Ugur11]), und die *Einsetzbarkeit* eines Werkzeugs [Abelha16], [Zhu15].

Die so gelernten oder im System hinterlegten Affordanzen können dabei auch bei der Interpretation von sprachbasierten Instruktionen verwendet werden, da übergebene Parameter auf ihre Ausführbarkeit untersucht werden können. Eine weitere Anwendung stellt die Auflösung von Mehrdeutigkeiten dar, welche schon in ein paar Fällen näher untersucht wurde ([Heikkilä12], [Min17], [Chen20]).

Der in [Heikkilä12] eingeführte Ansatz erlaubt eine Angabe von unvollständigen oder auch mehrdeutigen Instruktionen, für welche basierend auf in einer Datenbank hinterlegten Informationen und einer Prädiktion Rückfragen gestellt bzw. Mehrdeutigkeiten aufgelöst werden. Die Datenbank enthält dabei Objekt-Aktion-Paare und die Prädiktion basiert auf den vorhergehenden Instruktionen. Der Anwendungsfall besteht in diesem Fall aus einer Telekommunikation zwischen einem Astronauten und einem mobilen Roboter.

In [Chen20] wurde ein Neurales Netz mittels einem großen Text-Corpus angelernt, so dass Mehr-

deutigkeiten in einer Instruktion über das Wissen der generellen Verwendung der Satzteile aufgelöst werden können. Die Affordanz wird dabei also hauptsächlich dadurch ausgedrückt, wie häufig ein Wort in Verbindung mit anderen Wörtern auftaucht. Steht ein großer Text-Corpus zur Verfügung, können mit diesem Ansatz vielversprechende Ergebnisse erzielt werden.

2.6 Simulationsbasierte Validierung

Neben der Evaluation von Vorbedingungen um die Validität einer Bewegung abzuschätzen, wurde eine große Anzahl an Ansätzen veröffentlicht, welche eine Bewegung entweder symbolisch oder subsymbolisch validieren und auf eine Erfüllbarkeit bzw. das Auftreten von Abweichungen untersuchen. Die Ansätze lassen sich grob in drei Kategorien unterteilen: Rein symbolische Ansätze [Briggs15], [Sattar14], teilweise simulierende Systeme [Mösenlechner13], [Kresse17] und Systeme, welche die komplette Bewegung simulieren [Rockel15], [Kunze17]. Der Grund für die Reduktion der Simulation auf Teilschritte oder sogar den Verzicht auf eine Simulation ergibt sich dabei aus dem hohen Rechenaufwand, welcher allerdings auch gleichzeitig detailliertere Informationen liefert.

Die erste Kategorie enthält Ansätze, welche rein symbolisch arbeiten und auf eine physikbasierte Simulation verzichten. Beispiele dafür sind Ansätze basierend auf der Prädikatenlogik [Briggs15] oder Ansätze, welche eine Risiko-Einschätzung basierend auf Sensorwerten durchführen [Sattar14]. Bei ersteren werden die Bewegungen in Form von logischen Prädikaten dargestellt und miteinander verknüpft. Basierend auf einer Wissensdatenbank werden dann die jeweiligen Regeln ausgewertet und somit entschieden, ob eine Ausführung möglich ist. Letztere schätzen zum Beispiel bei Navigationsaufgaben die Wahrscheinlichkeit für eine erfolgreiche Bewegung darüber ein, dass sowohl der Zustand des System, als auch der Umgebungszustand berücksichtigt wird.

Teilweise simulierende Ansätze, auch als *Projektion* oder *Imagination* bezeichnet, nutzen eine physikbasierte Simulation an Stellen, an denen ein symbolisches Schließen nicht mehr ausreicht, und nehmen den höheren Rechenaufwand für genauere Ergebnisse in Kauf. Häufige Anwendungsfälle dafür sind die Berechnung von Kollisionen mit Objekten im Arbeitsraum oder auch Stabilitätsprüfungen von erstellen Konstruktionen [Mösenlechner13].

Neben vielen Ansätzen, welche eine komplette Simulation vor dem Einsatz des Systems nutzen, um ein System zu parametrieren [Kresse17] oder Erfahrungswerte mit Nutzern zu erhalten [Misra16], existieren bereits Ansätze, welche die Simulation während der Ausführung durchführen. Die Ansätze von [Rockel15] und [Kunze17] sind dabei unserem am ähnlichsten und werden deshalb näher betrachtet.

In [Rockel15] wurde die *Funktionale Imagination* eingeführt, welche die Ausführung einer Bewegung im Rahmen einer Festkörpersimulation überprüft und deren Erfolg feststellt. Eingehende Instruktionen werden zunächst über einen Planer in einen Bewegungsablauf umgewandelt und simuliert. Tritt bei einer Teilbewegung eine Fehlausführung auf, so wird diese Simulation verworfen und ein erneutes Planen findet statt, um eine bessere Parametrierung zu erhalten.

Neben der Simulation der kompletten Bewegung und dem Speichern von Objektparametern

wird in [Kunze17] die Simulation dadurch erweitert, dass eine zusätzliche Definition von Objekten als graphenähnliche Strukturen ermöglicht, wodurch Deformation und Zerbrechen dieser simuliert werden kann. Zudem werden Unsicherheiten im Bezug auf Objekttransformationen berücksichtigt, welche genutzt werden können, um beispielsweise Sensorrauschen zu simulieren.

2.7 Zusammenfassung

In diesem Kapitel wurden die Grundlagen für die folgenden Kapitel eingeführt und der Stand der Forschung zu den jeweiligen Themen wurde präsentiert.

Wizard-of-Oz Experimente wurden bereits für einige Anwendungen im Bereich der mobilen Robotik und der Steuerung von Roboterarmen umgesetzt. Dabei wurden viele Eingabegeräte, wie graphische Nutzeroberflächen oder Controller, genutzt. Ein Steuerung des Nutzer-Roboters mit einem baugleichen Roboter wurde jedoch noch nicht untersucht. Daher wird ein solcher Ansatz in Kapitel 5.1 eingeführt.

Der Einsatz der Fuzzy-Logik hat bisher ebenfalls zu einer Reihe an Veröffentlichungen in der Robotik geführt. So wurden Ansätze zur Spezifikation von Robotergeschwindigkeiten und von Objekteigenschaften vorgestellt. Ein Ansatz zur Abbildung von Kraftparametern konnte im Stand der Forschung jedoch nicht identifiziert werden. Da dies die Flexibilität der Instruktion verbessern würde, wird in Kapitel 5.2 ein Konzept eingeführt, welches sich diesem Problem widmet.

Affordanzen haben sich als nützliches Konzept in der Robotik bewiesen und wurden auch bereits für eine Mehrdeutigkeitsauflösung eingesetzt. Dabei wurde in den bisherigen Ansätzen ein gewisses Maß an Vorwissen vorausgesetzt, welches bei der Bearbeitung neuartiger Projekte nicht gegeben sein muss. Daher wird in Kapitel 6 ein Ansatz eingeführt, welcher ähnlich zu dem Ansatz aus [Heikkilä12] eine wissensbasierte Mehrdeutigkeitsauflösung durchführt, bei der jedoch nicht nur die Objekte, sondern auch der verwendete Roboterarm bei der Auswertung der Affordanzen berücksichtigt wird. Außerdem wird in diesem Kapitel das Verhalten von Nutzern bei der Auflösung von Mehrdeutigkeiten untersucht.

Auch im Bereich der simulationsbasierten Validierung wurden bereits einige Ansätze vorgestellt, welche Probleme von Roboterbewegungen vor der Ausführung erkennen und bewerten. Dabei lag der Fokus bisher jedoch weder gezielt auf der Bewertung von kraftbasierten Bewegungen, noch auf der Betrachtung von Resultaten, die von der Norm abweichen, als valide Ergebnisse. Daher beschäftigt sich der in Kapitel 7 vorgestellte Ansatz mit dieser Lücke.

Kapitel 3

Grundkonzept

Bevor die einzelnen im Rahmen dieser Arbeit entworfenen Komponenten näher beschrieben werden, erfolgt in diesem Kapitel eine Erläuterung des in [Spangenberg17] eingeführten Konzepts, auf dem diese Arbeit aufbaut. Grundlegend wurde mit diesem Konzept ein wissensbasierter Ansatz eingeführt, mit dem Verben auf kraftbasierte Roboterbewegungen abgebildet werden können. Dafür wurden insgesamt drei Komponenten umgesetzt, welche für die Extraktion der Information aus sprachlichen Instruktionen, das Hinterlegen von Umgebungsinformation in Form von Kontextwissen und die Erstellung von Roboterbewegungen aus einer Kombination der Instruktionen und dem Kontextwissen erlauben.

Modelliert man die Instruktion von Robotersystemen als dreischichtige Systemarchitektur (siehe Abbildung 3.1), bestehend aus einer Nutzerschicht, einer Transformationsschicht und einer Kontrollschicht, so ist der Hauptteil des Ansatzes in der Transformationsschicht anzusiedeln, welche

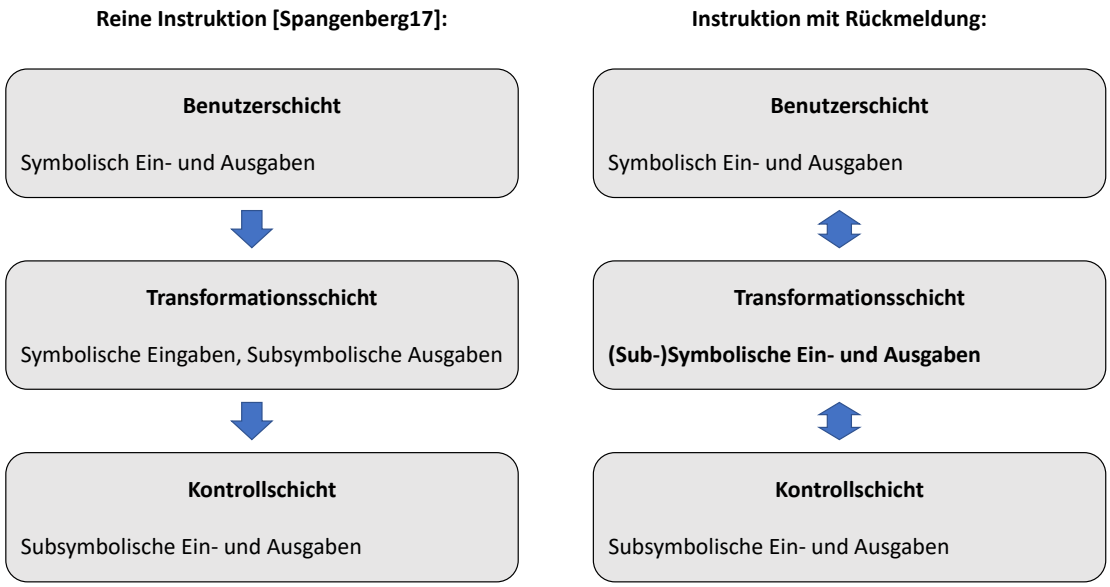


Abb. 3.1: Visualisierung des in [Spangenberg17] vorgestellten Ansatzes (links) und des in dieser Arbeit eingeführten Ansatzes (rechts) in Form eines dreischichtigen Systemarchitektur.

aus der symbolischen Information in Form von Nutzerinstruktionen aus der Nutzerschicht subsymbolische Informationen in Form von Reglerparametern für das Robotersystem generiert. Dies erlöst den Nutzer davon, alle Bewegungsparameter spezifizieren zu müssen, und erlaubt dadurch selbst Nichtexperten, Roboter sprachbasiert zu instruieren.

Im Folgenden werden diese drei Komponenten näher beschrieben, wobei für die Extraktion der Informationen aus Instruktionen außerdem eine Erweiterung eingeführt wird, welche Informationen aus Instruktionen zunächst in eine ontologiebasierte Wissensrepräsentation überführt, bevor eine Bewegung daraus erzeugt wird.

3.1 Bewegungserzeugung

Grundlage der in dieser Arbeit verwendeten Bewegungsprimitive sind die in [Spangenberg17] eingeführten *Verbalisierten Physikalischen Effekte* (VPE), als intuitives Framework bewiesen, um kraftbasierte Bewegungen zu definieren. Der Hintergedanke bei VPE ist, dass bei der Parametrierung der Bewegung physikalische Gesetze genutzt werden, welche eine gemeinsame Wissensbasis für Mensch und Roboter darstellen, da beide diesen gehorchen müssen. Grundlegend sind VPE wie folgt definiert:

$$\mathcal{VPE} = \{(V, E) | V \in \mathcal{V}, E \in \mathcal{PPE}\}. \tag{3.1}$$

Dabei stellt $\mathcal{V}$ die Menge aller über VPE abbildbaren Verben dar und $\mathcal{PPE}$ die Menge an sogenannten *Prinzipiellen Physikalischen Effekten*. Ein Verb wird also einer Bewegung zugeordnet. Die Menge der $\mathcal{PPE}$ ist wiederum folgendermaßen definiert:

$$\mathcal{PPE} = \{(T, I, O, MPN) | T \in \mathcal{PO}, I \in \mathcal{PQ}^{\alpha}, O \in \mathcal{PQ}^{\beta}\} \tag{3.2}$$

Hier stellt $\mathcal{PO} = absorb, change, transform$ die Menge an physikalischen Operationen dar, welche Bewegungen definieren, bei denen physikalische Größen absorbiert werden (Berührung eines Objekt), verändert werden (Hochheben eines Objekts) oder transformiert werden (Schieben eines Objekts). Die Parameter I und $O \in \mathcal{PQ}$ stellen die Eingangs- und Ausgangsparameter der vorherrschenden physikalischen Gesetze dar und MPN beschreibt die ausgeführte Bewegung in Form eines Manipulationsprimitivnetzes (siehe Kapitel 2).

Das Konzept der VPE erlaubt also die Abbildung von Verben auf kraftbasierte Roboterbewegungen, welche auf Basis von einer Objektdatenbank und physikalischen Gesetzen möglich ist. Die Intuitivität dieses Ansatzes wurde zudem im Rahmen einer Nutzerstudie erfasst, in der Nutzer dazu aufgefordert wurden, eine Reihe an mit diesem Ansatz darstellbaren Bewegungen mit dem Roboterarm auszuführen. Diese Ausführungen wurden danach mit Ausführungen durch das System verglichen, wobei eine hohe Ähnlichkeit beobachtet wurde. Die in dieser Studie betrachtete Bewegungsmenge wird im Folgenden als Menge der elementaren Bewegungen $\mathcal{M}_{\text{VPE}}$ bezeichnet und besteht aus den folgenden Bewegungen: *Bewegen, Drehen, Drücken, Heben, Senken, Kippen, Berühren, Schieben* und *Ziehen*.

3.2 Physikalisches Wörterbuch

Eine Kontextdatenbank, welche sowohl Informationen über Objekte im Arbeitsraum, als auch über auftretende Prozesse enthält, wurde in [Spangenberg17] in Form eines sogenannten *Physikalischen Wörterbuchs* umgesetzt. In diesem Wörterbuch werden zum einen statische Informationen bereitgestellt, welche aus einer Objekt- sowie einer Prozessdatenbank geladen werden. Zum anderen können Einträge des Wörterbuchs dynamisch über externe Komponenten wie Sensoren parametriert werden.

In den statischen Informationen sind dabei sowohl Informationen über Objektparameter, Prozessparameter oder auch Spezifikationen von Satzbestandteilen zu finden. Zu den Objektparametern gehören beispielsweise die Farbe, das Gewicht oder auch die Geometrie. Prozessparameter setzen sich aus Informationen über zugrundeliegende physikalische Gesetze und darin enthaltene grundlegende oder abgeleitete physikalische Größen, sowie deren Einheiten und Werte, zusammen. Informationen zu Satzbestandteilen sind letztlich in Form von Arten von Phrasen, wie Nominalphrasen oder Adjektiv-Phrasen, und deren Funktionen vorhanden.

In dieser Arbeit wird auf diesem Ansatz aufgebaut, in dem ebenfalls das Vorhandensein eines physikalischen Wörterbuchs angenommen wird, welches Informationen zu dem verfügbaren Kontext enthält. Im Folgenden wird diese Informationsquelle als Kontextdatenbank (Kontext-DB) bezeichnet.

3.3 Parametrierung von Roboteraktionen

In [Spangenberg17] erfolgt die Parametrierung der Roboterbewegungen im Grunde in zwei Schritten. Zuerst werden symbolische Nutzerinstruktionen in einen Phrasenstrukturbaum überführt. Mit Hilfe der im physikalischen Wörterbuch hinterlegten Information wird anschließend eine Roboterbewegung erzeugt, sofern dies möglich ist. Ist eine Abbildung einer Instruktion aufgrund nicht bekannter Parameter, wie einem fehlenden Verb oder einem nicht ausreichend spezifizierten Objekt, nicht möglich, so wird keine Bewegung erzeugt.

In dieser Arbeit wird als Zwischenrepräsentation kein Phrasenstrukturbaum, sondern eine auf einer Ontologie aufbauende Struktur verwendet, da dies eine Umsetzung von Dialogen erleichtert. Diese Struktur entspricht dabei einer Umsetzung des Prinzips der *Verb-Frames* [Chen20]. Dabei wird einem Verb $v \in \mathcal{V}$ eine Menge an möglichen bzw. notwendigen Parametern zugeordnet. Die hier genutzte Struktur realisiert das Verb-Frame-Prinzip mittels sogenannten *Spiro-Kommandos* $s \in \mathcal{S}$, welche wie folgt definiert sind:

$$s = \{(v, w, t, g) | v \in \mathcal{V}, w \in \mathcal{W}, t \in \mathcal{T}, g \in \mathcal{G}\} \tag{3.3}$$

Die Spiro-Kommandos setzen sich dabei aus einem Verb v, einem zu manipulierenden Werkstück w, einem Werkzeug t und Zusatzspezifikationen g zusammen. Die Menge $\mathcal{V}$ besteht dabei aus den Verben, welche auf kraftbasierte Roboterbewegungen abgebildet werden können. Die Menge $\mathcal{W}$ enthält Objektspezifikationen von Objekten im Arbeitsraum. Diese setzt sich zusammen

aus einem Arbeitsraumobjekttyp, z.B. Würfel, Schraube oder Holzbrett, und Adjektiven, wie Objektfarbe, -größe, oder auch -gewicht, welche eine Spezifikation eines Objektes ermöglichen. Für die Menge der Werkzeuge gilt: $\mathcal{T} \subset \mathcal{W}$. Die Menge $\mathcal{G}$ stellt mögliche Zielspezifikationen oder auch gewünschte Nachbedingungen dar. Entweder in Form einer numerischen Angabe für Distanzen oder Winkelangaben, oder Angaben, welche relativ zu einem weiteren Objekt in der Szene übergeben werden. Aus diesen Informationen wird zusammen mit anwendungsspezifischem Zusatzwissen eine Roboterbewegung erzeugt.

Eine Instruktion wie beispielsweise:

"Stelle den Klotz mit dem Greifer auf den Schwamm!"

entspricht dem Spiro-Kommando:

$s = \{$Verb: Stellen, Werkstück: Klotz, Werkzeug: Greifer, Spezifikation: auf, Schwamm$\}$.

Eine sprachliche Instruktion wird damit also auf die für eine Bewegung relevanten Informationen reduziert. Somit wird zudem kein zusätzlicher Schritt für eine semantische Erweiterung benötigt.

3.4 Übersicht

Der in [Spangenberg17] eingeführte Ansatz hat sich rein mit der Abbildung von Instruktionen auf Bewegungen befasst und daher lediglich jeweils eine unidirektionale Verbindung von der Nutzerschicht in Richtung der Transformationsschicht und der Transformationsschicht in Richtung der Kontrollschicht umgesetzt. Solange valide, eindeutige Instruktionen übergeben werden, ist eine Erzeugung von Bewegungen auch möglich. In dieser Arbeit wird der Fall betrachtet, dass auch mehrdeutige oder nicht valide Instruktionen gegeben sein können, aber dennoch versucht werden soll, eine valide Bewegung zu erzeugen. Anstatt fehlende oder abweichende Parameter über eine Optimierung zu schätzen, wird hier über Rückfragen an den Nutzer gearbeitet. Dies hat hauptsächlich den Grund, dass hier davon ausgegangen wird, dass für eine Optimierung nicht genügend Erfahrungswerte vorhanden sind.

Die Rückmeldung des Systems führt dabei im Hinblick auf die dreischichtige Systemarchitektur zu einer Erweiterung der unidirektionalen zu einer bidirektionalen Verbindung (siehe Abbildung 3.1), da in Fehlerfällen die jeweiligen Informationen an die höhere Schicht weitergereicht werden, bis sie dem Nutzer in der Nutzerschicht symbolisch präsentiert werden und damit die Flexibilität und Robustheit des Systems erweitert wird. Die in dieser Arbeit vorgestellten Ansätze sind dabei wie folgt den jeweiligen Schichten zuzuordnen. Die Erweiterung des zugrundeliegenden Bewegungsmodells in Kapitel 4 wird der Kontrollschicht zugeordnet. Kapitel 5.1 untersucht die Verwendung von Parametern während einer Instruktion und gehört deshalb zur Nutzerschicht. Der in Kapitel 5.2 eingeführte Ansatz erlaubt eine Abbildung von unscharfen Parametern auf

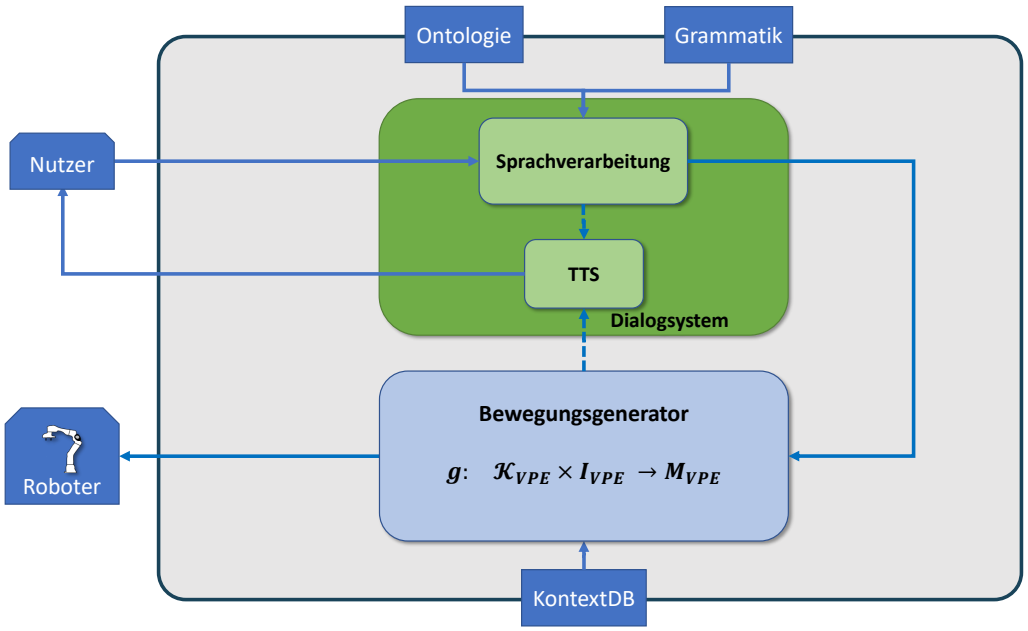


Abb. 3.2: Ausgangslage des Gesamtsystems.

numerische Werte und gehört dadurch zur Transformationsschicht. Kapitel 6 ermöglicht eine Anpassung der Instruktionen und arbeitet dabei sowohl auf der Nutzer, als auch der Transformationsschicht. Weiterhin beinhaltet Kapitel 7 einen Ansatz zur simulationsbasierten Validierung von Instruktionen und befindet damit sowohl auf der Transformationsschicht als auch auf der Kontrollschicht. Daraus ergibt sich das in Abbildung 3.2 dargestellte System, welche die Ausgangslage dieser Arbeit darstellt.

Kapitel 4

Kombination elementarer kraftbasierter Roboterbewegungen

Eine große Herausforderung bei der Erstellung einer intuitiven Sprachsteuerung ist die Definition einer geeigneten Schnittstelle. Verlangt man die Spezifikation aller notwendigen Parameter für eine Kraftbewegung, ist das sowohl für Experten als auch Nichtexperten sehr aufwendig bis unmöglich. Daher wird ein Verfahren benötigt, welches dem Nutzer eine möglichst natürliche Instruktion erlaubt. Für elementare kraftbasierte Bewegungen haben sich dabei die in Kapitel 3 eingeführten VPE bewährt.

Eine Nebenbedingung dieses Ansatzes ist, dass pro Effekt jeweils ein physikalisches Gesetz enthalten ist. Die Menge $\mathcal{V}_{VPE} \in \mathcal{V}$ der somit abbildbaren Verben beschränkt sich dadurch auf elementare Verben, also solche, die eine elementare Bewegung (siehe Kapitel 3) ausdrücken. Eine Kombination mehrerer VPE ist bisher jedoch weder auf sprachlicher Ebene durch Kommandos wie: “Drücke den Schwamm auf die Oberfläche und bewege den Schwamm nach rechts!”, noch auf Bewegungsebene durch Kommandos wie: “Wische mit dem Schwamm nach rechts!” umgesetzt worden. Da eine sprachliche Kombination von parallelen Bewegungen unter Umständen langwierig ist und zu Missverständnissen führen kann, wird in diesem Kapitel ein Ansatz vorgestellt, welcher eine Kombination auf Bewegungsebene erlaubt, und damit auch implizit eine sprachliche Kombination umsetzt. Dieses Kapitel beschäftigt sich demnach mit der Frage:

F1 In wie weit kann die Menge der durch VPE abbildbaren kraftbasierten Bewegungen über deren Kombination erweitert werden?

Dafür wird zunächst anhand von Beispielverben genauer dargelegt, warum eine Erweiterung von $\mathcal{V}_{VPE}$ notwendig ist. Hierauf aufbauend wird eine Erweiterung der VPE auf eine explizite Angabe eines Werkzeugs vorgestellt. Darauf aufbauend wird ein Konzept eingeführt, mit Hilfe dessen eine Kombination von VPE möglich ist. Somit kann eine Erweiterung der abbildbaren Verben und Bewegungen erfolgen. Mit einer Nutzerevaluation wird die Intuitivität des neuen Modells geprüft und in einer abschließenden Zusammenfassung werden die erzielten Erkenntnisse diskutiert. Der dargestellte Ansatz zur Erweiterung von VPE ist in [Wölfel18a] veröffentlicht.

4.1 Ansatz

Im Folgenden wird der in dieser Arbeit verfolgte Ansatz vorgestellt, mittels dem eine Kombination von kraftbasierten Roboterbewegungen ermöglicht wird. Nach einer Motivation der Notwendigkeit solch einer Kombination, wird zunächst gezeigt, wie die bisherige Bewegungsrepräsentation um eine Werkzeugabhängigkeit erweitert wird. Anschließend wird die theoretische und praktische Kombination von Bewegungen erläutert.

4.1.1 Werkzeugabhängigkeit und Kombination

Bei der sprachlichen Kommunikation lässt sich beobachten, dass Menschen auf verschiedene Weisen unvollständige Instruktionen verwenden bzw. Instruktionen minimal paraphrasieren. So werden häufig Ellipsen genutzt, wenn sich Teile einer Instruktion mit der vorhergehenden Instruktion decken. Ein Beispiel dafür sind die Instruktionen "Schiebe den roten Klotz nach vorne!" - "Und jetzt [schiebe] den blauen [Klotz]!". Das Verb *Schieben* und der Werkstücktyp *Klotz* sollen also für die zweite Instruktion aus dem Kurzzeitgedächtnis übernommen werden.

Neben Ellipsen tritt eine solche Verkürzung auch bei einer Instruktion von Verben in Zusammenhang mit einem Werkzeug auf. Die Beispiele aus Tabelle 4.1 zeigen, welchen Einfluss Werkzeuge auf Instruktionen haben. So entsteht aus der Instruktion "Drücke mit einem Stempel!" die verkürzte Instruktion "Stempel!". Dies kann man sich beispielsweise für die Erzeugung von Bewegungen zu Nutze machen, da Werkzeuge nicht zusätzlich angegeben werden müssen, sondern aus dem Verb abgeleitet werden können. Um diese Abhängigkeit bei einer Abbildung von Instruktionen auf Bewegungen berücksichtigen zu können, bedarf es einer expliziten Werkzeugdefinition. Da die in dieser Arbeit genutzten VPE eine solche Definition nicht vorsehen, muss eine entsprechende Erweiterung erfolgen.

Ein ähnlicher Effekt tritt auf, wenn zwei oder mehrere Bewegungen zu einer Bewegung zusammengefasst werden (siehe Tabelle 4.2). Dabei werden sowohl parallele, also gleichzeitige, als auch sequentielle Verbindungen benötigt. So ergibt sich aus einem gleichzeitigen *Drücken* und *Bewegen* beispielsweise ein *Schneiden* und aus einem *Drücken* und *Drehen* ein *Schrauben*. Neben der Art der Verknüpfung existiert auch hier eine Werkzeugabhängigkeit bei der Wahl des entsprechenden Verbs. Dies fällt insbesondere bei einem gleichzeitigen Drücken und Bewegen in eine Richtung auf, da selbst für diese vergleichsweise simple Kombination bereits fünf unterschiedliche Verben je nach Werkzeugabhängigkeit entstehen.

In der Definition der VPE ist enthalten, dass pro Effekt ein physikalisches Gesetz gilt (siehe

Tab. 4.1: Verb-Werkzeug Instruktion und dazu gehörige alternative Instruktionen.

Verb-Werkzeug Instruktion	**Alternative Instruktion**
Drücke mit dem Stempel auf das Papier	Stempel das Papier
Drücke mit dem Messer in den Apfel	Steche in den Apfel
Drücke mit dem Prägewerkzeug in die Knete	Präge die Knete

Tab. 4.2: Verben, Werkzeuge, vermutete Bewegungsprimitive und deren sequentielle bzw. parallele Kombinationen.

Verb	**Werkzeug(e)**	**Bewegungsprimitive**	**Kombination**
Schneiden	Messer	Drücken, Bewegen	Parallel
Kratzen	Nagel, Messer	Drücken, Bewegen	Parallel
Malen	Stift, Pinsel	Drücken, Bewegen	Parallel
Bügeln	Bügeleisen	Drücken, Bewegen	Parallel
Einfügen	Greifer	Drücken, Bewegen	Parallel
Schütten	Flasche, Kanne	Heben, Drehen, Senken	Sequentiell
Schöpfen	Kelle	Senken, Drehen, Heben	Sequentiell
Schrauben	Schraubenzieher	Drücken, Drehen	Hybrid
Bohren	Bohrer	Drücken, Drehen	Hybrid

Kapitel 3). Dies kann bei den vorgestellten Kombinationen nicht gewährleistet werden, da für jede elementare Bewegung bereits ein physikalisches Gesetz benötigt wird. Dies hat zur Folge, dass das Modell der VPE erweitert werden muss, damit die Menge der abbildbaren Verben $\mathcal{V}_{VPE}$ vergrößert werden kann. Diese Erweiterung erfolgt nun zunächst über die Erweiterung des VPE-Konzepts auf eine explizite Werkzeugabhängigkeit und zudem durch die Einführung eines Ansatzes zur Kombinationen einzelner VPE.

4.1.2 Elementare Verbalisierte Effekte

Das in Kapitel 3 eingeführte Konzept der VPE hat sich als intuitiver Ansatz zur Abbildung von kraftbasierten Roboterbewegungen auf Verben bewährt. In einem VPE werden dabei Informationen über das benutzte Werkzeug implizit innerhalb der Manipulationsprimitive gespeichert und nicht als Parameter übergeben. Soll eine Bewegung mit einem speziellen Werkzeug ausgeführt werden, muss dieses also zunächst ausgerüstet werden. Eine Instruktion einer Bewegung mit einem gegriffenen Werkzeug bzw. dem Greifer würde demnach beispielsweise folgendermaßen ablaufen:

Hebe das Werkzeug!

Drücke in das Werkstück!

Stelle das Werkzeug ab!

Dabei kann jedoch für das Heben, Drücken und Abstellen eine unterschiedliche Greifpose für das Werkzeug benötigt werden. In diesem Fall würde mit dem aktuellen System je nach hinterlegten Greifposen ein ungewolltes Resultat entstehen. Eine mögliche Lösung, welche eine erfolgreiche Abarbeitung dieser Instruktionen ermöglicht, wäre, vor jeder Teilbewegung die aktuelle Greifpose mit einer geforderten Greifpose zu vergleichen und eine Umgreifroutine aufzurufen, falls ein

Unterschied besteht. Dies würde jedoch auch zu einer längeren Ausführungszeit der Bewegung führen. Eine Möglichkeit auch dieses Problem zu umgehen ist die Einführung von komplexeren Instruktionen, welche ein Aufgreifen und Ablegen von Werkzeugen kapseln. Dadurch könnten die oben genannten Instruktionen auf die einzelne Instruktion

Drücke mit dem Werkzeug in das Werkstück!

reduziert werden. Für die Abbildung der Instruktion auf eine Roboterbewegung bedeutet dies, dass zunächst überprüft wird, ob das genannte Werkzeug bereits gegriffen ist und ob es bezogen auf das Verb korrekt gegriffen ist. Gelten beide Bedingungen, so kann die Bewegung erzeugt werden. Ist das Werkzeug bereits gegriffen, aber die Greifpose nicht korrekt, so muss die Greifpose angepasst werden. Ist das Werkzeug noch nicht gegriffen, so kann es initial mit der korrekten Greifpose aufgegriffen werden. Im Bezug auf den Bewegungsablauf müssen demnach die Annäherungs- und die Endphase erweitert werden, welche im Rahmen der VPE bereits implizit modelliert werden. Die Annäherungsphase wird dabei um die Aufnahme des Werkzeugs mit einer Greifpose erweitert, welche entweder vordefiniert ist oder zur Laufzeit bestimmt wird. Gleiches gilt für die Abschlussphase. Hier wird eine Bewegung hinzugefügt, welche die Ablage des Werkzeugs umsetzt (siehe Abbildung 4.1).

Um VPE in diesem Sinne anpassen zu können, bedarf es also einer Menge aus möglichen Werkzeug-Greifpose-Kombinationen $\mathcal{T}_G = \{(t, c) | t \in \mathcal{T}, c \in \mathcal{R}^{4\times4}\}$, welche mit einem VPE verbunden werden können. Die Erweiterung der VPE um diese Werkzeug-Greifpose-Kombination wird im folgenden als *Elementare Verbalisierte Effekte* $\mathcal{EVE}$ bezeichnet:

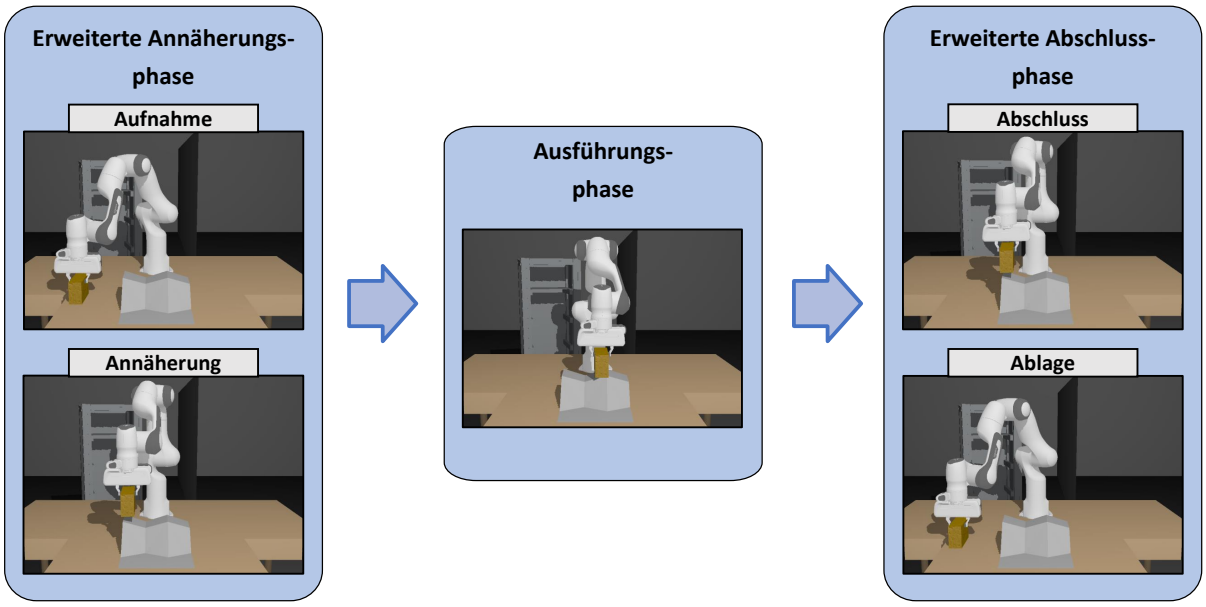


Abb. 4.1: Erweiterung der Bewegungsphasen von VPE um eine Aufnahmephase eines Werkzeugs vor und einer Ablagephase nach der Bewegung.

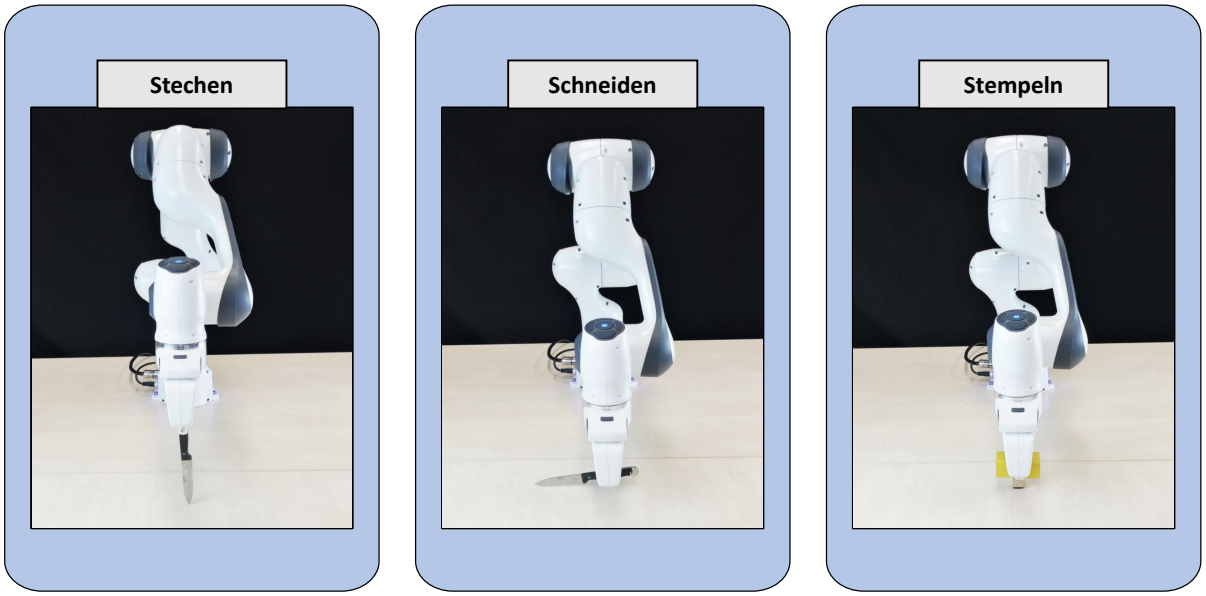


Abb. 4.2: Kombinierte Bewegung aus Drücken und Bewegen mit einem Messer (Stechen, Schneiden) und einem Stempel (Stempeln).

$$\mathcal{EVE} = \{(T, P) | T \in \mathcal{T}_G, P \in \mathcal{VPE}\}. \tag{4.1}$$

So wird ermöglicht, dass Werkzeuge mit einer bestimmten Greifpose für eine bestimmte Bewegung definiert werden können. Außerdem ist somit die Verwendung eines zugrunde liegen Verbs und dessen Bewegung für eine Auswahl von unterschiedlichen Werkzeugen möglich.

Betrachtet man beispielsweise ein Drücken, so ergeben sich mit einem Messer, je nachdem wie es gegriffen wurde, unterschiedliche Beschreibungen bzw. Verben für die resultierende Bewegung (siehe Abbildung 4.2). Zeigt die Spitze des Messers in Bewegungsrichtung des Roboterarms bzw. auf den Kontaktpunkt, so wird es als *Stechen* interpretiert. Ist das Messer im Greifer jedoch um 90 Grad gedreht, so wird es als *schneiden* interpretiert. Die zugrunde liegende Bewegung wäre dabei die gleiche und nur die Greifpose wäre unterschiedlich.

Angelehnt an die Notation der $\mathcal{PPE}$ werden die $\mathcal{EVE}$ auch in absorbierende, transformierende und verändernde Effekte unterteilt. Absorbierende Effekte stellen dabei kraftüberwachte Bewegungen dar, also Bewegungen welche solange ausgeführt werden, bis ein Grenzwert in den Kräften bzw. Momenten festgestellt wird, wie zum Beispiel für das Verb bzw. die Bewegung *Berühren*. Transformierende Effekte stellen kraftgeführte Bewegungen dar, also Bewegungen bei denen gewisse Kräfte oder Momente während der Ausführung aufrecht erhalten bleiben sollen, wie beispielsweise beim *Schieben*. Die verändernden Effekte stellen schließlich rein positionsgeregelte Bewegungen dar.

4.1.3 Kombinierte Verbalisierte Effekte

Über $\mathcal{EVE}$-Elemente können elementare Bewegungen über deren Verben mit einem Werkzeug verknüpft werden. Um längliche Instruktionen wie "Drücke und drehe mit dem Greifer!" oder "Drücke und ziehe mit dem Schwamm!" zu umgehen, bedarf es jedoch einer Möglichkeit, die Verb-Werkzeug-Kombinationen zu einem Verb zusammenzufassen. Außerdem sollen elementare Bewegungen parallel und sequentiell zu neuen Bewegungen kombiniert werden können.

Ein Ansatz, welcher sich für eine strukturierte sequentielle Verknüpfung von Bewegungsprimitiven bewährt hat, sind endliche Automaten (siehe Kapitel 2), bei denen jeder Zustand einer Bewegung entspricht. Da die generelle Definition von Automaten eine parallele Abarbeitung von zwei Zuständen nicht erlaubt, wird sie im Folgenden auf das Modell der sogenannten Manipulationsknoten $\mathcal{N}$ erweitert:

$$\mathcal{N} = \{E^{\alpha} | E \in \mathcal{EVE}, \alpha \in \mathbb{N}_0^+\}. \tag{4.2}$$

Ein Manipulationsknoten kann einen oder mehrere Elemente E aus der Menge $\mathcal{EVE}$ enthalten und erlaubt dadurch die Umsetzung von parallelen Kombinationen. Die Anzahl der enthaltenen Elemente ist über den Parameter α definiert. Gilt $\alpha > 1$, so bedeutet dies, dass mehr als eine Bewegung durch den Roboterarm gleichzeitig ausgeführt wird. Außerdem können über diesen Wert Fehlerzustände modelliert werden, indem $\alpha = 0$ gesetzt wird.

Aus der Menge der Manipulationsknoten lässt sich dann ein Zustandsautomat definieren, welcher die folgenden Einträge enthält: Einen Startzustand $N_0 \in \mathcal{N}$, eine Menge an Zielzuständen $N_F \subset \mathcal{N}$ und Übergänge $C = (N_i, N_j), N_i, N_j \in \mathcal{N}$. Die Übergänge werden dabei nicht durch eine Eingabe von außen gesteuert, sondern über die Stopp-Kriterien der in den Knoten enthaltenen Manipulationsprimitive ausgelöst. Zudem gibt es die Möglichkeit, dass in einem Knoten ein Fehlverhalten über Stopp-Kriterien erkannt wird. Solche Fehlverhalten können beispielsweise durch ein *Hängen bleiben* an einem Objekt im Arbeitsraum entstehen, was über ein zeitliches Kriterium abgefangen werden kann.

Fügt man diesem Zustandsautomaten ein Verb und eine explizite Werkzeugdefinition hinzu, erhält man ein Modell, welches seine sequentielle und parallele Verknüpfung von werkzeugabhängigen Bewegungen erlaubt, die Menge der *Kombinierten Verbalisierten Effekte* ($\mathcal{KVE}$), welche folgendermaßen definiert sind:

$$\mathcal{KVE} = \{\ (V, T, N, C, N_0, N_F) \mid V \in \mathcal{V},\ T \in \mathcal{T}_G,\ C \in \mathcal{N} \times \mathcal{N},\ N_0, N_F, N \subset \mathcal{N}\ \}. \tag{4.3}$$

Dabei entspricht $V \in \mathcal{V}_{KVE}$ abbildbaren Verben, welche einzelne werkzeugabhängige Bewegungen oder Kombinationen derer beschreibt. Über T wird erneut ein Werkzeug inklusive einer Greifpose spezifiziert und $N_{KVE} \subset \mathcal{N}$ entspricht einer Untermenge der Menge an *Manipulationsknoten* $\mathcal{N}$, welche die durch $\mathcal{V}_{KVE}$ beschriebenen Bewegungen darstellt. Die explizite Angabe des Werkzeugs und der damit verbundenen Greifpose ist auch bei einem Element KVE $\in \mathcal{KVE}$ notwendig, da es wie ein Element aus $\mathcal{EVE}$ mit unterschiedlichen Werkzeugen ausgeführt werden

kann.

Ein Spezialfall stellt die Ausführung mit dem Greifer dar. Dort ist keine Angabe einer Greifpose notwendig, da hier angenommen wird, dass der Greifer fix am Roboterarm angebracht ist. Dadurch kann sich die Transformation vom Roboterarm zum Greifer nicht verändern und wird somit eindeutig modelliert.

Beispiele für KVE sind in Abbildung 4.3 in Form von Schneiden mit einem Messer und Schrauben mit dem Greifer zu sehen. Aus der Instruktion:

Schneide den Kuchen mit dem Messer!

würde sich folgendes KVE ergeben:

$$\begin{aligned} \mathrm{KVE}_{\mathrm{Schneiden}} &= (\mathit{Schneiden}, (\mathrm{Messer}, c_{\mathit{Messer}}), \{N_0\}, C, \{N_0\}, \{N_0\}), \\ N_0 &= \{E^2\} = \{E_{t,0}, E_{a,0}\}, \\ C &= \{\}. \end{aligned}$$

Die Pose des Messers und des Kuchens ist dem System dabei bekannt und c_{Messer} beschreibt die in Abbildung 4.3, links dargestellte Greifpose. Das EVE $E_{t,0} = \{(\mathrm{Messer}, VPE_{\mathrm{Bewegen}})\}$ wird dabei als Transform-EVE mit kartesischen absoluten Belegungen der ersten drei Freiheitsgrade umgesetzt. Das EVE $E_{a,0} = \{(\mathrm{Messer}, \mathrm{VPE}_{\mathrm{Drücken}})\}$ wurde als absorb umgesetzt, bei dem lediglich der dritte Freiheitsgrad kraftbasiert belegt ist. Das zugrundeliegende Manipulationsprimitivnetz besteht damit neben den erweiterten Annäherungs- und Abschlussphasen aus einem Manipulationsprimitiv, bei dem die ersten zwei Freiheitsgrade kartesisch und der dritte Freiheitsgrad kraftbasiert definiert sind, was einer parallelen Ausführung eines Drücken MP und eines Bewegen MP in einem identischen Taskframe entspricht. Grundlegend ist das Schneiden

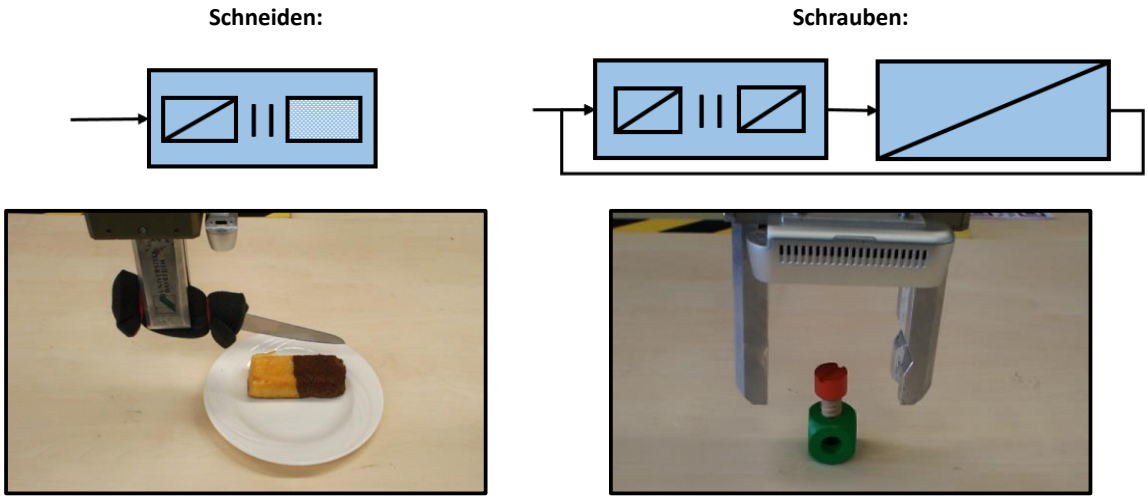


Abb. 4.3: Beispiele für ein rein paralleles KVE (*Schneiden*) und eine hybrides KVE (*Schrauben*).

hier als Bewegung entlang einer Achse des Objektes umgesetzt, da es um die Bewegung an sich geht und nicht darum, welche Aufteilung Nutzer bei einem Schneiden erwarten.

Ein Beispiel für ein sequentielles KVE ist ein Schrauben mit dem Greifer. In diesem Fall wird nicht nur die Schraubbewegung ausgeführt, sondern es wird eine zusätzliche Drehbewegung in entgegengesetzte Richtung nötig, wenn die Gelenkwinkelgrenzen des Roboterarms erreicht wurden, ohne dass das gewünschte Drehmoment anliegt. Der Roboterarm muss sich hier zunächst zurückdrehen, bevor die Schraubbewegung weiter ausgeführt werden kann. Für die Instruktion:

Schraube die Schraube mit dem Greifer rein!

würde also folgendes KVE initialisiert werden:

$$\begin{aligned}\mathrm{KVE}_{\mathrm{Schrauben}} &= (\textit{Schrauben}, (\mathrm{Greifer}, \{\}), \{N_0, N_1\}, C, \{N_0\}, \{N_1\}),\\ N_0 &= \{E^2\} = \{E_{t,0}, E_{a,0}\}, N_1 = \{E_{t,1}\}\\ C &= \{(N_0, N_1), (N_1, N_0)\}.\end{aligned}$$

Die Pose der Schraube ist dem System bekannt und der Würfel, in den die Schraube eingeführt wird, ist am Tisch fixiert. Das EVE $E_{t,0} = \{(\mathrm{Greifer}, \mathrm{VPE}_{\mathrm{Drehen}})\}$ und das EVE $E_{t,1} = \{(\mathrm{Greifer}, \mathrm{VPE}_{\mathrm{Bewege}})\}$ sind dabei in Form eines Transform-EVE modelliert und das EVE $E_{a,0} = \{(\mathrm{Greifer}, \mathrm{VPE}_{\mathrm{Drücken}})\}$ ist als Absorb-EVE umgesetzt. Falls ein Schraubenzieher zur Verfügung gestanden wäre, wären die entsprechenden Bewegungen mit dem Schraubenzieher initialisiert worden.

4.1.4 Kombination von Bewegungen

Um aus den modellierten KVE auch ausführbare Roboterbewegungen erstellen zu können, werden zum einen Komponenten benötigt, welche die sequentielle Kombinationen ermöglichen, und zum anderen Komponenten, welche eine parallele Kombinationen zu einem einzelnen Manipulationsprimitivnetz (MPN) zusammenführen können. Es wird nun zunächst näher beschrieben, wie im Rahmen dieser Arbeit eine sequentielle Kombination umgesetzt wurde. Danach werden die Nebenbedingungen für eine parallele Kombination spezifiziert und ein Ansatz für die Umsetzung erläutert.

Bei der sequentiellen Kombination von Manipulationsknoten wird das resultierende MPN M_{res} folgendermaßen erzeugt: Zu Beginn wird M_{res} mit dem MPN des Startknoten N_0 initialisiert. Danach werden die durch die Übergänge C definierten MPN der Nachfolgeknoten zu M_{res} hinzugefügt, wobei die Indizes der jeweiligen Manipulationsprimitve (MP) basierend auf den bis zu diesem Zeitpunkt hinzugefügten MPN angepasst werden.

Eine allgemeingültige parallele Kombination von MPN stellt eine Herausforderung dar, da die

einzelnen Bestandteile auf MPN-Ebene synchronisiert werden müssen. Dies bedeutet generell eine Synchronisation der jeweiligen Manipulationsprimitive und somit der Taskframes, der adaptiven Selektionsmatrizen und der Abbruchbedingungen. Da ein allgemeingültiges Umsetzen eines solchen Ansatzes nicht Teil dieser Arbeit ist, werden im Folgenden Nebenbedingungen eingeführt, welche die Art der möglichen Kombinationen bestimmen.

Pro Knoten wird vorausgesetzt, dass die Anzahl der MP gleich ist, also eine Kombination der MPN eine komponentenweise Kombination der MP bedeutet. Ist die Anzahl der MP pro Knoten unterschiedlich, wird ein zusätzlicher Synchronisationsansatz benötigt. Um beispielsweise eine Bewegung umzusetzen, bei der ein *Drücken* erst ab einem bestimmten Zeitpunkt t_1 erfolgen soll, muss diese Bewegung dementsprechend in Teilbewegungen mit und ohne vorhandenes *Drücken* unterteilt werden (siehe Abbildung 4.4). Ein temporales Beeinflussen der Startzeit ist momentan nicht umgesetzt, könnte aber über den in [Pek16] vorgestellten Ansatz in zukünftigen Anwendungen eingebaut werden.

Außerdem wird festgelegt, dass die Taskframes von zwei zu kombinierenden MP identisch sind. Durch diese Bedingung lassen sich die Komponenten der jeweiligen Selektionsmatrizen eindeutig im Sinne der Raumrichtungen miteinander verbinden. Um diese Kombination umzusetzen, werden die Einträge der Selektionsmatrizen miteinander verglichen und ausgewählt. Im Falle von zwei kartesischen Einträgen wird der betragsmäßig kleinere Wert übernommen. Im Falle einer Mischung aus einem kartesischen und einem kraftbasierten Eintrag in einem Freiheitsgrad wird letzterer bevorzugt. Die Bevorzugung der Kraftkomponenten dient dem Zweck, dass eine sichere Ausführung angestrebt wird. Die Abbruchkriterien werden in diesem Fall auf die gleiche Weise ausgewählt.

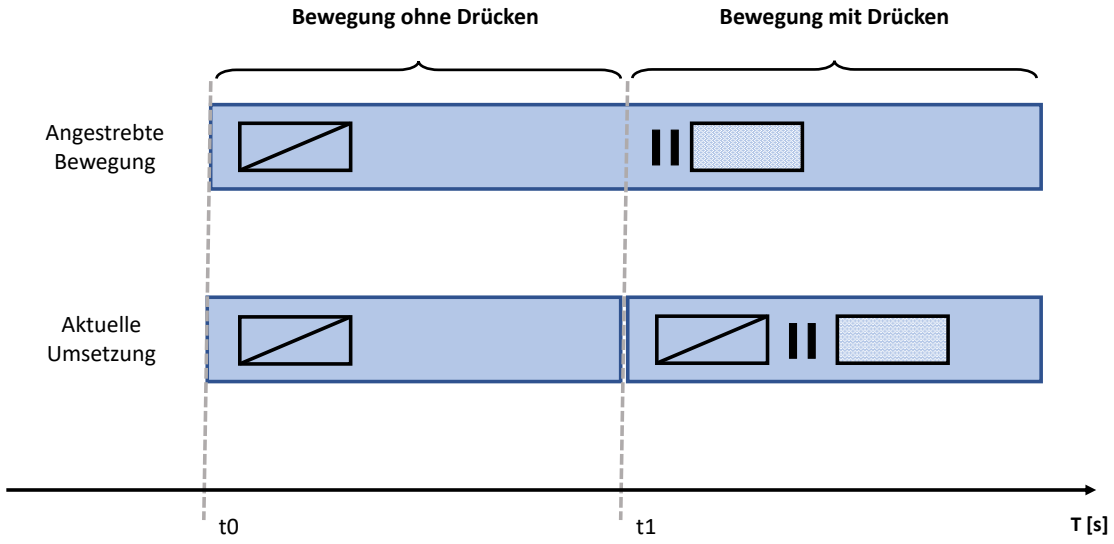


Abb. 4.4: Aufteilung einer Bewegung, bei der ein Drücken erst ab einem Zeitpunkt t_1 stattfindet in zwei KVE.

4.2 Nutzerevaluation

Ein Grund dafür, dass VPE als Grundlage für KVE gewählt wurden, ist deren nachgewiesene Intuitivität [Spangenberg17]. Deshalb stellt sich im Rahmen einer Evaluation des Ansatzes vor allem die Frage, ob durch die Erweiterung des Ansatzes der VPE, also bei KVE, die Intuitivität erhalten bleibt. Hierfür wurde eine Nutzerstudie durchgeführt, bei der überprüft wurde, ob Nutzer kraftbasierte Bewegungen intuitiv so benennen und strukturieren würden, wie es in dem hier vorgestellten Ansatz geschieht. Im ersten Teil wurden den Nutzern gleiche Bewegungskombinationen mit unterschiedlichen Werkzeugen präsentiert, um diese von den Nutzern benennen zu lassen. Im zweiten Teil wurden die Nutzer dazu aufgefordert, angegebene Verben über eine Kombination aus vorgegebenen Primitivbewegungen zu modellieren.

Insgesamt haben $n = 18$ Nutzer an der Studie teilgenommen, wobei 67 % jünger als 30 Jahre, 11 % zwischen 30 und 40 und 22 % älter als 40 Jahre alt waren. Der Anteil männlicher Teilnehmer betrug 78 % und der Anteil weiblicher Teilnehmer 22 %. Die Teilnehmer bestanden dabei sowohl aus Experten als auch Nichtexperten im Bereich Robotik.

Im ersten Experiment wurde den Nutzern die in Abbildung 4.5 dargestellten Roboterbewegungen präsentiert. Eine animierte Version der Abbildung ist in diesem Video[1] dargestellt. Nach der Betrachtung der jeweiligen Bewegung hatten die Teilnehmer die Aufgabe alle Verben zu notieren, welche ihrer Meinung nach die gezeigte Bewegung beschreiben (einschließlich Synonymen).

Um das Resultat dieses Experiments zu evaluieren, wurde eine Einheitlichkeit E_1 berechnet (siehe Abbildung 4.6, blaue Balken), indem die Anzahl des am häufigsten verwendeten Verbs $a_{1,\max}$ durch die Anzahl der Teilnehmer geteilt wurde, also:

$$E_i = \frac{a_{i,\max}}{n} \tag{4.4}$$

[1] `http://www.ai3.uni-bayreuth.de/resypub/files/wlfel2018a.Userstudy.video.php`

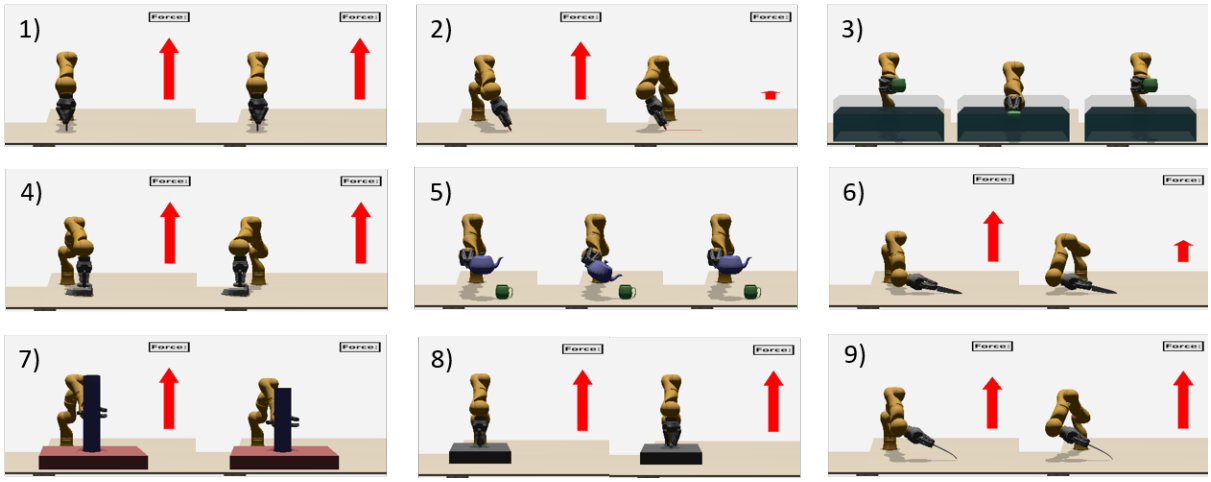


Abb. 4.5: Visualisierung der Roboterbewegungen für die Verben: 1) *Bohren*, 2) *Malen*, 3) *Schöpfen*, 4) *Bügeln*, 5) *Schütten*, 6) *Schneiden*, 7) *Einfügen*, 8) *Schrauben* und 9) *Kratzen*. Der rote Pfeil symbolisiert die Kraft, welche durch die Umgebung (ohne Berücksichtigung der Gravitation) auf den Greifer wirkt.

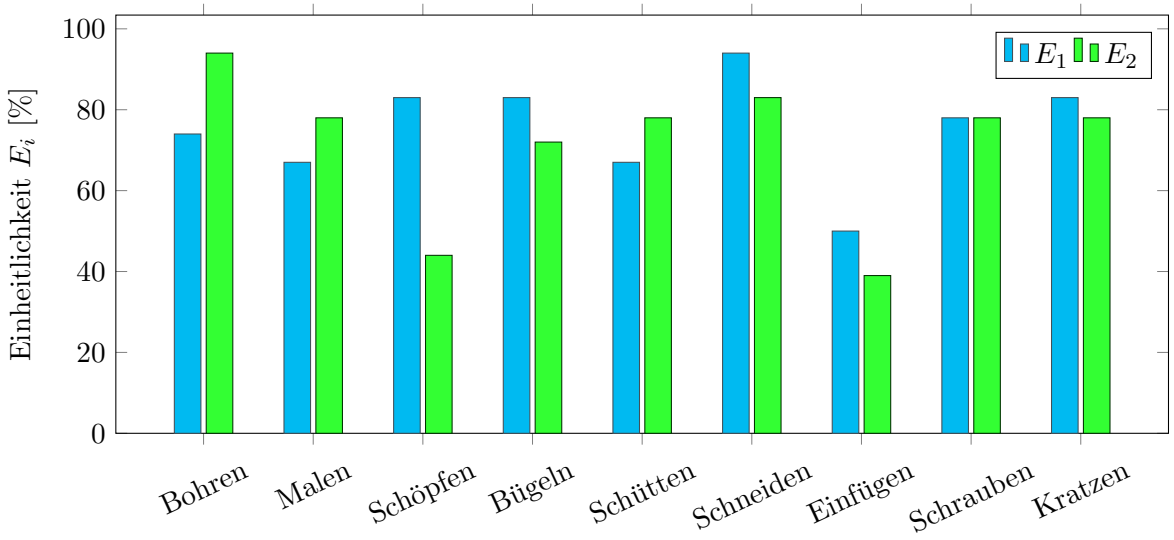


Abb. 4.6: Einheitlichkeit der Antworten im ersten Teil (E_1) und im zweiten Teil (E_2) der Nutzerevaluation.

Für den Großteil der Bewegungen ergibt sich dabei eine hohe Übereinstimmung der notierten Verben, was zu Einheitlichkeiten von über 70 % führte. Die Einheitlichkeiten von 67 % für *Malen* und *Schütten* entstehen vermutlich von der Mehrdeutigkeit dieser Bewegungen. Die Bewegung für *Malen* wurde von manchen Teilnehmer beispielsweise als *Markieren* oder *Zeichnen* interpretiert. Dies könnte daran liegen, dass der Kontext nicht weiter bestimmt wurde und das Werkzeug abstrakt in Form eines Stifts in einheitlicher Farbe dargestellt wurde. Beim *Einfügen* hat schließlich die Angabe von Teilen der Gesamtbewegung (*Schieben, Ziehen*) oder Synonymen wie *Verbinden* zu einer Einheitlichkeit von lediglich 50 % geführt.

Im zweiten Experiment wurden den Teilnehmern neun durch KVE abbildbare Verben zusammen mit den folgenden Bewegungsprimitiven bereitgestellt: *Drehen, Schieben, Heben, Senken* und *Drücken*. Die Teilnehmer sollten dann aufzeichnen, wie sie aus den Primitiven die Verben kombinieren würden. Als Kombinationsmöglichkeiten wurde das Zeichnen eines Pfeils zwischen

Tab. 4.3: Häufigste von den Nutzern in der zweiten Aufgabe der Evaluation gewählte Bewegungskombinationen um Verben darzustellen.

Verb	**Gewählte Kombination**
Bohren	Drücken und Drehen
Malen	Drücken und Ziehen
Schöpfen	Senken, Drehen, Drehen, Heben
Bügeln	Drücken und Ziehen
Schütten	Drehen
Schneiden	Drücken und Ziehen
Einfügen	Senken, Ziehen und Drücken
Schrauben	Drücken und Drehen
Kratzen	Drücken und Ziehen

zwei Verben (sequentielle Kombination), das Zeichnen von Bewegungen übereinander (parallele Kombination) oder eine Mischung aus den beiden (hybride Kombination) erlaubt. Den Teilnehmern wurde zudem freigestellt, dass sie zusätzliche Anmerkungen notieren dürfen.

Die am häufigsten gewählten Kombination zu den jeweiligen Verben sind in Abbildung 4.3 dargestellt. Die Interpretation des zweiten Experiments erfolgte erneut über die Definition einer Einheitlichkeit E_2 der Antworten, welche durch ein Teilen der am häufigsten genutzten Kombination $a_{2,\max}$ geteilt durch die Anzahl der Nutzer berechnet wird. Die geringste Einheitlichkeit gab es beim *Schöpfen* (44 %) und *Einfügen* (39 %), was vermutlich daran liegt, dass diese Bewegungen je nach Anwendungsfall unterschiedlich aufgebaut seien können. Die Hauptunterschiede waren die Abfolge von *Drehen*, *Heben* und *Senken*. Die anderen Kombinationen erreichten Einheitlichkeiten von über 70 %.

Die Resultate der beiden Experimente zeigen, dass das eingeführte Modell der KVE eine Abbildung von Verben auf Kombinationen von kraftbasierten Bewegungen so entspricht, wie Nutzer es intuitiv ebenfalls darstellen würden. Abgesehen von den Mehrdeutigkeiten bei *Malen* und *Einfügen* waren sich die Teilnehmer bei der Benennung der Bewegungen einig und wählten auch abhängig vom Werkzeug unterschiedliche Verben für die gleiche Kombination. Beachtet man, dass nicht nur Experten, sondern auch Nichtexperten an der Studie teilgenommen haben, liegt der Schluss nahe, dass die Intuitivität bei der Erweiterung von VPE auf KVE erhalten geblieben ist.

4.3 Zusammenfassung

In diesem Kapitel ist ein neuartiger Ansatz beschrieben, um die Liste abbildbarer Instruktionen $\mathcal{V}_{VPE}$ und kraftbasierter Roboterbewegungen $\mathcal{M}_{VPE}$ aus [Spangenberg17] zu den Mengen $\mathcal{V}_{KPE}$ und $\mathcal{M}_{KVE}$ zu erweitern. Dies ist ein weiterer Schritt in Richtung der Vision eines optimalen Robotersystems, da gilt $|\mathcal{V}_{KVE}| > |\mathcal{V}_{VPE}|$ und $|\mathcal{M}_{KVE}| > |\mathcal{M}_{VPE}|$. Durch die Erweiterung der VPE auf EVE und letztendlich auf KVE ist eine Definition von kraftbasierten Bewegungen möglich. Bewegungs-Primitive können nun sequentiell, parallel oder hybrid kombiniert werden und eine explizite Zuweisung eines Werkzeugs wurde ermöglicht. Die Intuitivität dieses Ansatzes wurde im Rahmen einer Nutzerstudie evaluiert und eine Beibehaltung der Intuitivität wurde festgestellt. Im Gesamtkonzept spiegelt sich dieser Ansatz in Form des erweiterten Bewegungs-Generators wieder (Siehe Abbildung 4.7).

Die Antwort auf die eingangs gestellte Frage ist demnach, dass bei einem statischen Einarm-System basierend auf VPE eine Erweiterung auf Primitivkombinationen unter Nebenbedingungen möglich ist. Diese Erweiterung in Form von KVE besteht dabei aus sequentiellen und parallelen Kombinationen, die auf neue Verben abgebildet werden können. Bei der parallelen Kombination ist dabei die aktuelle Nebenbedingung zu berücksichtigen, dass bei den beteiligten Bewegungsprimitiven der zugrunde liegende Taskframe übereinstimmen muss. Verglichen mit anderen Ansätzen zur Kombination von Feinbewegungen ist hierbei der Zugewinn, dass zum einen VPE als Bewegungsprimitive verwendet werden und zum anderen neben einer sequentiellen auch eine parallele Kombination durch die Einführung von Manipulationsknoten ermöglicht

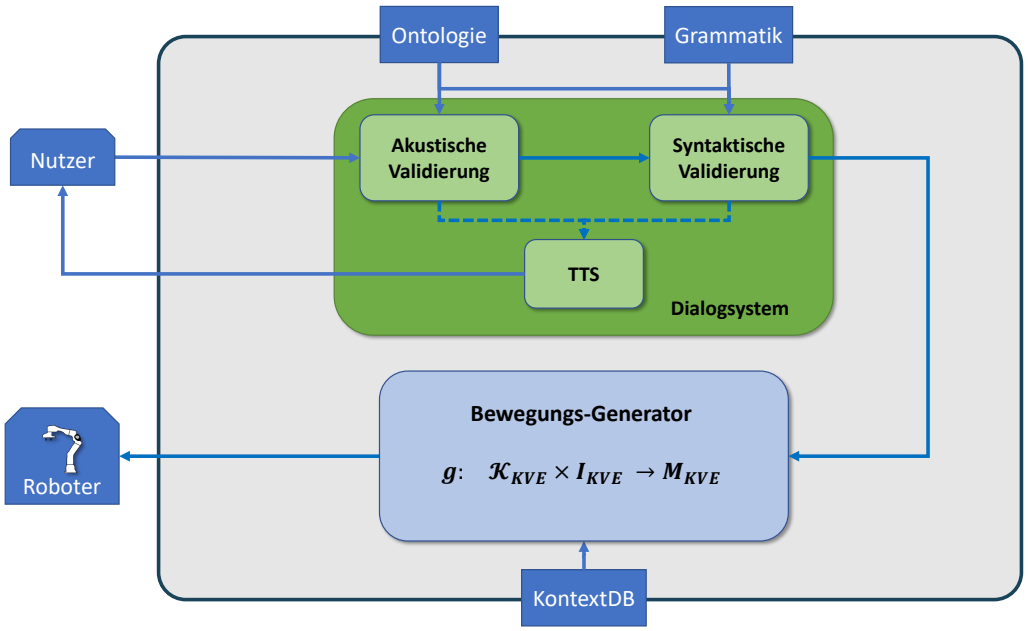


Abb. 4.7: Übersicht über das Gesamtkonzept nach der Erweiterung der VPE auf KVE.

wird.

Mögliche Erweiterungen dieses Ansatzes sind auf unterschiedlichen Ebenen möglich. In dieser Form sind KVE auf die Manipulation einzelner Werkstücke und nicht etwa einer Menge an Werkstücken ausgelegt. Daher sollte eine Erweiterung auf Objektmengen in einer zukünftigen Arbeit näher betrachtet werden. Außerdem wäre eine Verknüpfung von mehreren Instruktionen hilfreich, falls diese sich nicht über ein alternatives Verb formulieren lassen. Ein Beispiel dafür wäre “Stelle die roten Klötze einen Zentimeter nach rechts und stelle die weißen Klötze auf die roten Klötze!”.

Neben diesen Erweiterungen wäre zudem eine Überprüfung von bereits erfüllten Teilbewegungen vor einer Ausführung nützlich. Instruiert ein Nutzer beispielsweise “Hebe den Schwamm!” und darauf folgend “Wische mit dem Schwamm über den Tisch!”, würde derzeitig die Aufnahme des Schwamms wiederholt werden, obwohl sie bereits durchgeführt wurde.

Kapitel 5

Interpretation unscharf formulierter kraftbasierter Roboterinstruktionen

Ein Grund für die Notwendigkeit einer intuitiven Schnittstelle zur Instruktion von kraftbasierten Roboterbewegungen ist, dass Menschen es nicht gewohnt sind Kraftwerte explizit anzugeben. In zwischenmenschlichen Unterhaltungen werden beispielsweise selten Aussagen wie: "Wische *mit 2 Newton* über den Tisch!" statt "Wische *leicht* über den Tisch!" genutzt.

Solch ein Verhalten wurde in anderen Anwendungsgebieten, wie der Kommunikation von Distanzen oder Geschwindigkeiten, schon untersucht und mit der in Kapitel 2 beschriebenen Fuzzy-Logik umgesetzt. Das Konzept des Wizard-of-Oz Experiments stellt dabei eine Möglichkeit dar, Informationen über die Verwendung von unscharfen Parametern zu erfassen. Für kraftbezogene Parameter wurde bisher jedoch noch kein Konzept vorgestellt, welches speziell auf diese Art von Roboterbewegungen ausgerichtet ist. Daher beschäftigt sich der erste Teil dieses Kapitels mit der wissenschaftlichen Fragestellung:

F2 In wie weit und in welcher Form nutzen Anwender unscharf formulierte Parameter als Synonym für numerische Kraftwerte?

Nachdem ein Konzept für eine Wizard-of-Oz Studie zur Erfassung solcher Parameter vorgestellt und evaluiert wird, erfolgt die Einführung eines Ansatzes, welcher eine Antwort auf folgende Frage darstellt:

F3 In wie weit können unscharf formulierte Kraftparameter auf kraftbasierte Roboterbewegungen abgebildet werden?

Die Erzeugung eines Ansatzes, welcher eine solche Abbildung ermöglicht, würde die Menge der abbildbaren Instruktionen und Bewegungen erweitern und damit die Flexibilität des Systems erhöhen. Im Folgenden findet zunächst die Einführung und Evaluation eines neuartigen Wizard-of-Oz Konzepts (veröffentlicht in [Wölfel20c]) statt. Danach wird ein Ansatz zur Abbildung unscharfer Kraftparameter eingeführt (veröffentlicht in [Wölfel19b]) und die Erkenntnisse werden in einer Zusammenfassung diskutiert.

5.1 Wizard of Botz

In diesen Kapitel wird ein Konzept für eine Wizard-of-Oz Studie vorgestellt, welche im Rahmen dieser Arbeit zur Erfassung der Verwendung von Kraftparametern in gesprochenen Instruktionen erstellt wurde: Wizard-of-Botz. Dabei wird zunächst das grundlegende Konzept erläutert, welches danach anhand einer Nutzerevaluation zum einen auf seine Nützlichkeit evaluiert wird und zum anderen genutzt wird, um die Informationen über die Verwendung von Kraftparametern zu erhalten. Abschließend werden die Ergebnisse diskutiert und Vorteile des Systems dargelegt.

5.1.1 Konzept

Im Rahmen der hier vorgestellten Studie soll herausgefunden werden, ob und in welcher Weise Nutzer unscharfe Kraftparameter während Instruktionen verwenden. Die Sprache der Nutzer sollte dabei nicht eingeschränkt werden, damit möglichst natürliche Ergebnisse erzielt werden. Eine Durchführung mit Hilfe eines autonomen Prototyps ist dabei nicht umsetzbar, da Formulierungen von Kraftparametern erst durch die Studie erhalten werden. Um die Studie dennoch mit einem realen Roboterarm durchführen zu können, um eine möglichst realistische Situation zu erzeugen, hat sich der Ansatz der Wizard-of-Oz Studie (siehe Kapitel 2) angeboten.

Bei der hier eingeführten Studie entspricht der Wizard dem Kontrolltyp *Manipulation* [Riek12], was bedeutet, dass er ausschließlich die von den Nutzern instruierten Bewegungen ausführt und dabei weder verbale noch nicht-verbale Rückmeldungen an den Nutzer weiterleitet. Der grundlegende Aufbau der Studie verfolgt die üblichen Prinzipien einer Wizard-of-Oz Studie: Der Nutzer und der Wizard befinden sich an separaten Arbeitsplätzen und der Nutzer weiß nicht, dass der Wizard das System des Nutzers steuert (siehe Abbildung 5.1). Der Hauptbestandteil des Aufbaus sind Roboterarme R1 und R2, für den Wizard und den Nutzer, welche über eine sogenannte Gravitationskompensation verfügen, was bedeutet, dass der Wizard sie bewegen kann, indem er sie per Hand führt. Im Idealfall sind R1 und R2 außerdem mit dem selben Endeffektor ausgestattet, da sich der Wizard so allein auf die Ausführung der Bewegung konzentrieren kann.

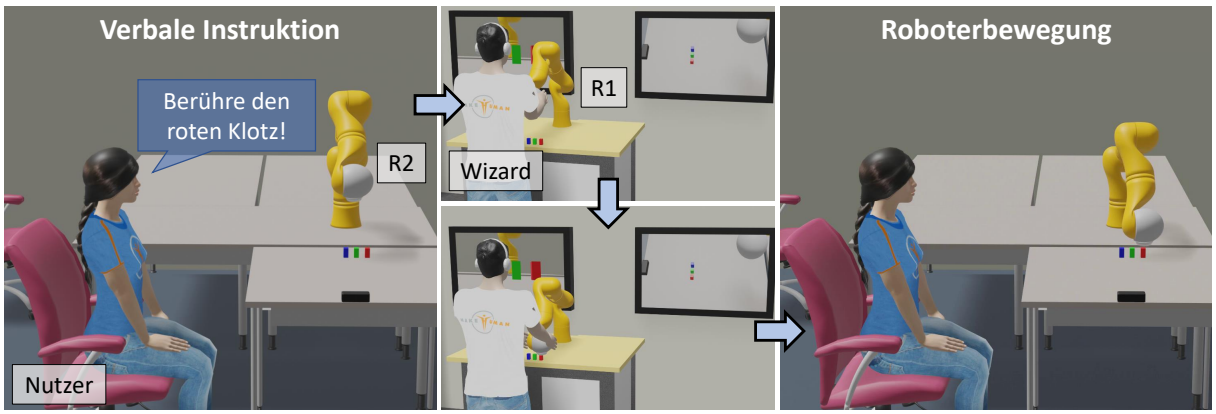


Abb. 5.1: Versuchsablauf: Die Instruktion des Nutzers wird an den Wizard weitergeleitet; dieser führt sie mit R1 aus. Die Bewegung wird dann auf den Nutzerroboter R2 übertragen.

Um dem Wizard einen Einblick in den Arbeitsbereich des Nutzers zu ermöglichen, sind eine Reihe an Webcams zu montieren, welche den Nutzer-Arbeitsbereich sofern möglich vollständig abdecken. Dadurch wird dem Wizard die Möglichkeit gegeben auf unerwartete Reaktionen reagieren zu können, welche lediglich im Nutzer-Arbeitsraum auftreten. Dazu zählen Situationen wie das unbeabsichtigte Betreten des Arbeitsraums durch den Nutzer, oder auch Kollisionen aufgrund eines leicht abweichenden Versuchsaufbaus. Weiterhin sollte zumindest ein Mikrofon im Arbeitsbereich des Nutzers vorhanden sein, sodass der Wizard nicht nur eine visuelle, sondern auch eine akustische Rückmeldung erhält, bzw. die Instruktionen des Nutzers auch bei einem lauteren Geräuschpegel gut wahrnehmen kann. Im Idealfall erhält der Wizard also eine Rückmeldung über den haptischen, visuellen und auditiven Kanal.

Der Wizard steuert R1 während der Studie fern, indem er R2 führt. Dies wird darüber ermöglicht, dass die Gelenkstellungen $\Theta_2 \in \mathbb{R}^7$ von R2 sowie Greifer-Kommandos über ein lokales Netzwerk an ein Programm übermittelt werden, welches für die Steuerung von R1 zuständig ist. Dieses Programm gibt Θ_2 an die Steuerung von R1 als Zielkonfiguration $\Theta_1 \in \mathbb{R}^7$ weiter. Dies führt dazu, dass R1 die Bewegungen von R2 mit einer kurzen Verzögerung imitiert und ein Greifen, sowie loslassen von Objekten im Arbeitsraum möglich ist. R1 wird rein positions-geregelt bewegt und ist mit einer leichten Nachgiebigkeit ausgestattet. Das bedeutet, dass der Roboterarm im Falle von Kollisionen nicht bremst oder versucht auf der geplanten Bahn weiterzufahren, sondern dass er zwar versucht der Bahn zu folgen, dabei jedoch ausgelenkt werden kann. Im Falle einer ungewollten Kollision wird so das Ausmaß von Schäden reduziert.

Die in Abbildung 5.2 dargestellte räumliche Trennung ermöglicht dem Wizard R2 so zu bewegen, dass es für Nutzer die Illusion erzeugt, dass das System die Instruktionen der Nutzer autonom in Roboterbewegungen umwandelt. Außerdem befindet sich ein Versuchsleiter in der Nähe des

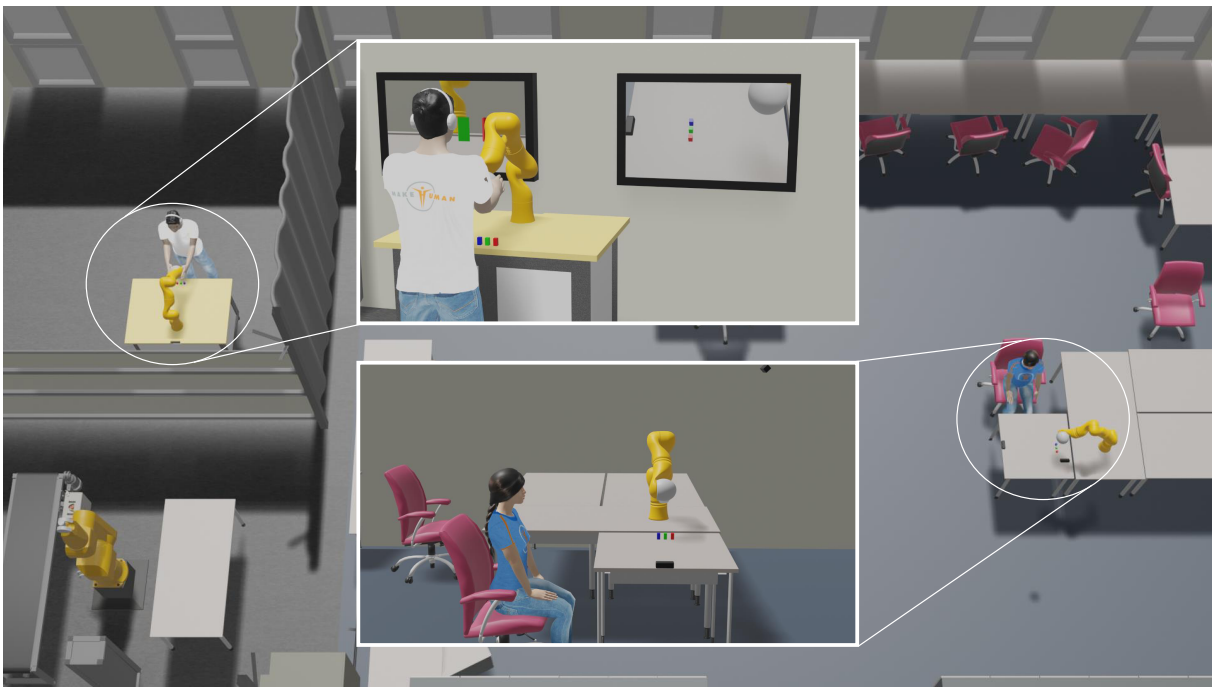


Abb. 5.2: Versuchsaufbau: Der Wizard mit zwei Monitoren, einem Roboterarm und einem Headset, getrennt vom Nutzer mit einem Roboterarm und zwei Webcams.

Tab. 5.1: Werkstücke in der Startsituation und geforderte Zielzustände für die Aufgaben T1 bis T4 aus der Wizard-of-Botz Studie.

Aufgabe	**Werkstück(e)**	**Zielzustand**
T1	3 ungeprägte Knetmassen	3 unterschiedlich dicke Prägungen
T2	Leeres Blatt Papier	3 unterschiedlich dicke Striche auf dem Papier
T3	3 separate Klötze	Ein aus Bauklötzen konstruierter Torbogen
T4	3 Gläser an Startposition	3 Gläser an Zielposition

Nutzers, um eventuelle Rückfragen zu beantworten.

5.1.2 Nutzerevaluation

Um zum einen die Nützlichkeit der Studie zu erfassen und zum anderen Informationen über unscharfe Parameter zu sammeln, wurde das vorgestellte Wizard-of-Botz Konzept in Form eines Prototypen umgesetzt und evaluiert. Der grundlegende Aufbau des Prototypen ist folgender: Der Nutzer und der Wizard befinden sich an unterschiedlichen Stellen im selben Raum und sind dabei durch einen Vorhang voneinander getrennt. So ist gesichert, dass kein Sichtkontakt vom Nutzer zum Wizard besteht (siehe Abbildung 5.2). Die verwendeten Roboterarme sind in diesem Prototyp ein KUKA LBR IV und ein KUKA LBR IV+ [Bischoff10], welche sich nicht relevant voneinander unterscheiden. Aus Mangel an identischen Greifern, wurden zwei unterschiedliche Greifer verwendet: ein Schunk Zweifinger-Greifer am Roboter R1, und einen Robotiq Dreifinger-Greifer am Roboter R2.

Da der Schwerpunkt der Studie auf der Kraftrückmeldung liegt, welche über die Roboterarme umgesetzt wird, wurde der Unterschied nicht als hinderlich angesehen, da Abweichungen in der Greifergeometrie über die visuelle Rückmeldung kompensiert werden können. Für die visuelle Rückmeldung ist der Aufbau mit zwei Webcams ausgestattet, welche sowohl über dem Nutzer-Arbeitsbereich, als auch daneben montiert sind, so dass, unter anderem wegen den unterschiedlichen Greifern, Kontakte früh genug erkannt werden. Zur akustischen Rückmeldung für den Wizard wurden die in den Webcams enthaltenen Mikrofone genutzt.

Die Nutzer wurden gebeten, den Roboterarm so zu instruieren, dass er die vier in Abbildung 5.3 dargestellten Aufgaben durchführt. Die wesentliche Einschränkung war dabei, dass die Nutzer ausschließlich ihre Stimme zur Instruktion verwenden durften. Die Aufgaben setzten sich dabei zusammen aus: Prägen von Knete (T1), Malen mit einem Pinsel (T2), Stapeln von Bauklötzen (T3), und dem Schieben von unterschiedlich gefüllten Gläsern zu jeweiligen Zielpositionen (T4). Die Aufgaben sind dabei in zwei Phasen aufgeteilt: Einer Einführungsphase und einer Instruktionsphase. Während der Einführungsphase werden den Nutzern die jeweiligen Werkstücke in der Startsituation und Zielzustände für die aktuelle Aufgabe gezeigt (siehe Tabelle 5.1). Während T1 ist dabei ein Prägewerkzeug eingespannt, während T2 ein Pinsel zusammen mit einem mit Farbe gefüllten Behälter gegeben und bei T3 und T4 steht der Zweifinger-Greifer zur Verfügung. Das erlaubte Vokabular wird nicht eingeschränkt und die Nutzer wurden explizit darauf hingewiesen, dass sie das System natürlich-sprachlich instruieren können.

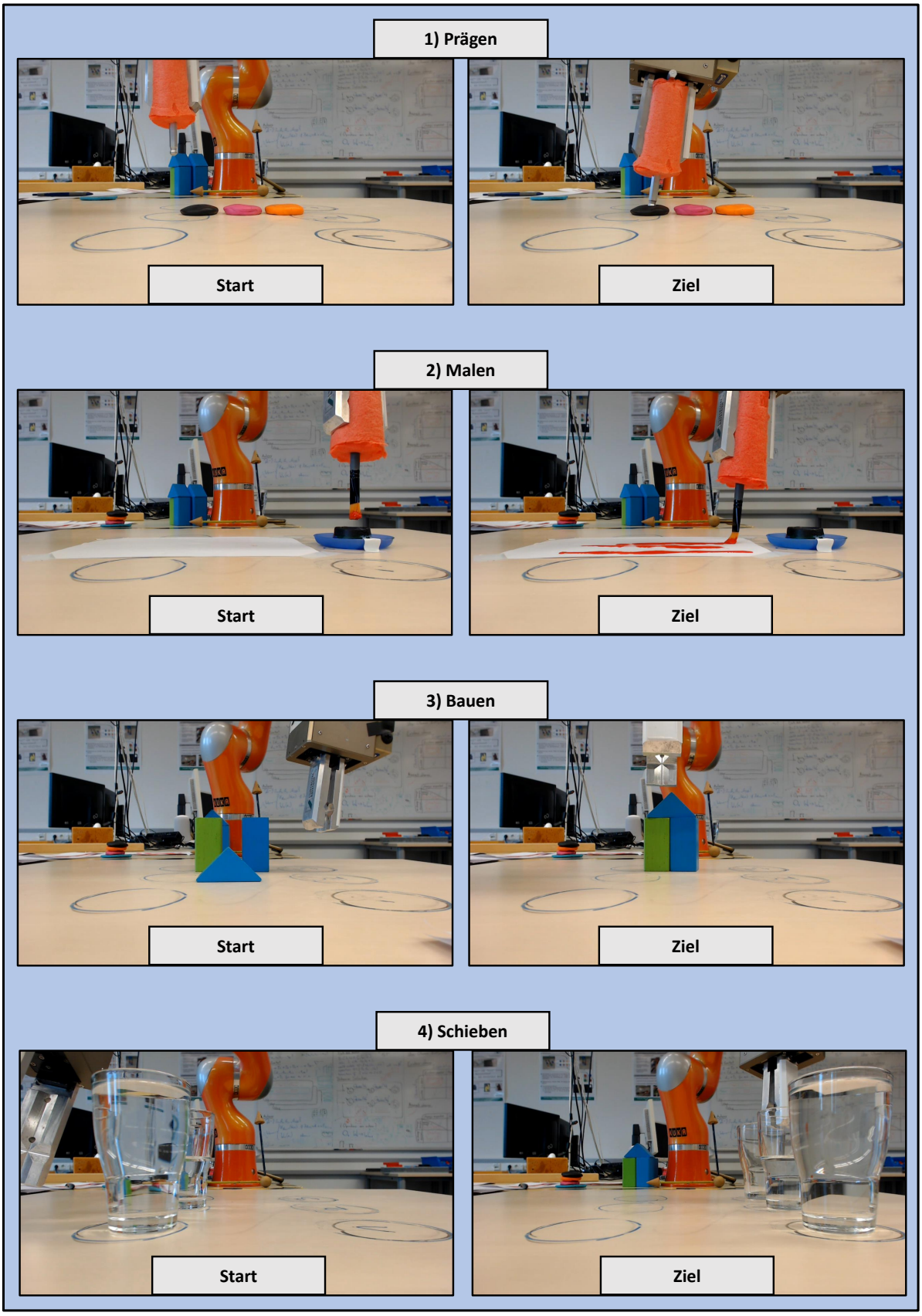


Abb. 5.3: Start und Zielzustände der bearbeiteten Aufgaben eines Studienteilnehmers: 1) Prägen von Knete, 2) Malen mit einem Pinsel, 3) Erstellen eines Torbogens und 4) Schieben von Gläsern.

Nachdem die Aufgaben bearbeitet wurden, werden die Nutzer zudem darum gebeten, die folgenden Aussagen auf einer Skala von -3 (stimme gar nicht zu) bis 3 (stimme stark zu) zu bewerten und bei Bedarf weitere Anmerkungen über das System zu machen:

Q1 Es war leicht den Roboter zu programmieren

Q2 Der Roboter hat sich so verhalten wie ich es vermutet habe

Q3 Mir gefiel die Möglichkeit, den Roboter korrigieren zu können

Q4 Eine verbale Rückmeldung wäre praktisch gewesen

Q5 Die Bewegungen des Roboters waren nicht komisch

Q6 Ich könnte mir vorstellen einen Roboter in Zukunft so zu instruieren

Q7 Eine sprachbasierte Steuerung reicht vollkommen aus

Die Aufgaben T1 - T4 wurden generell so konzipiert, dass sie sowohl über rein positions-geregelte Bewegungen ausgeführt werden könnten, als auch über kraftbasierte Bewegungen. Ziel der Studie ist eine Antwort auf die Frage zu finden, ob und, falls ja, wie Nutzer unscharfe Kraftparameter verwenden. In T1 wäre also beispielsweise entweder ein “Fahre 3 Zentimeter in die Knete” oder ein “Drücke leicht in Knete” möglich.

5.1.3 Ergebnis

Insgesamt haben $n = 15$ Nutzer an der Evaluation teilgenommen, wobei 6 männlich und 9 weiblich waren und das Durchschnittsalter 28 Jahre betrug. Die Verteilung der Angaben hinsichtlich ihrer Erfahrung im Bereich Robotik, Programmierung und Sprachsteuerung ist in Abbildung 5.4 dargestellt. Im Bereich von 1 (Nicht-Experte) bis 7 (Experte) lag die durchschnittliche Erfahrung E im Programmieren bei $E_{\text{Prog}} = 3,3$, im Bezug auf Robotik bei $E_{\text{Prog}} = 2,5$, und im Bereich Sprachsteuerung bei $E_{\text{Sprache}} = 2,4$. Während 27% der Teilnehmer noch nie ein Gerät per Sprache gesteuert haben, hat der Großteil der verbleibenden Teilnehmer zumindest ihr Smartphone per Sprache instruiert.

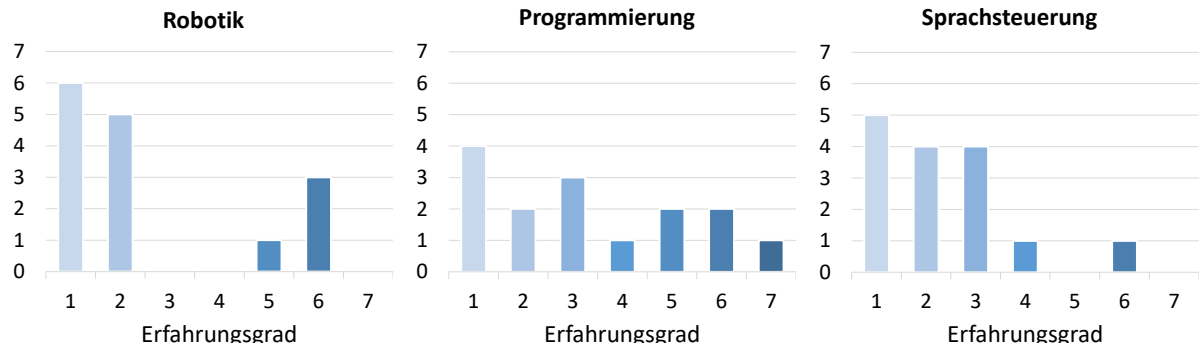


Abb. 5.4: Eigene Einschätzung der Wizard-of-Botz Teilnehmer zu den jeweiligen Gebieten auf einer Skala von 1 (Nicht-Experte) bis 7 (Experte).

Im Folgenden werden die gewählten Instruktionen der Nutzer während der Bearbeitung der vier Aufgaben näher untersucht. Während der Instruktionsphase wurden größtenteils Beobachtungen gemacht, welche sich mit dem in [Ralph08] durchgeführten Experiment decken, bei welchem ebenfalls eine Wizard of Oz Studie zur Objektmanipulation durchgeführt wurde. Der Unterschied zu dieser Studie ist dabei, dass der Wizard den Roboterarm in [Ralph08] über eine graphische Nutzeroberfläche gesteuert hat. Dies ermöglicht zwar eine bessere Wiederholbarkeit und eine höhere Genauigkeit, schränkt den Wizard jedoch in seiner Flexibilität ein.

Die wesentlichen Beobachtungen sind, dass, bis auf ein paar Ausnahmen, alle Teilnehmer dazu tendieren, Objekte im Arbeitsraum als räumliche Referenzen zu verwenden, also Instruktionen wie "Fahre über die orangene Knete!", anstatt reine Richtungsvorgaben, wie "Fahre zehn Zentimeter nach links!" zu nutzen. Bei Teilnehmern fand ebenfalls eine Vermenschlichung des Roboters in der Form statt, dass sich die Teilnehmer bei dem Roboter bedankt haben, wenn etwas erfolgreich ausgeführt wurde, oder sie den Roboter gerügt haben, falls er sich nicht so bewegt hat, wie es gewünscht war.

Ein Teil der Nutzer (20 %) hat die Aufgaben komplett über elementare Befehle gelöst, was in diesen Fällen zu einer längeren Bearbeitungszeit geführt hat, da jeder Teilschritt instruiert wurde. Die restlichen 80 % haben komplexere natürlich-sprachliche Instruktionen verwendet und dabei insgesamt in 80 % der Fälle generell unscharfe Parameter genutzt und in 73,3 % unscharfe Parameter bezogen auf die Kraft verwendet. Die unscharfen Parameter werden im Folgenden anhand der Formulierung relativ zum Verb oder dem Resultat klassifiziert. Beispiele für genannte Parameter sind in Tabelle 5.2 dargestellt. Von diesen 73,3 % haben dabei 82 % der Nutzer Verbabhängige Parameter wie beispielsweise *leicht, mittelleicht, sanft, vorsichtig, wie eine Feder, mit wenig Druck* oder *mit viel Druck* verwendet und 64 % Resultat-abhängige Parameter durch Formulierungen wie *mittlere Stanztiefe, dicken Strich, dünnen Strich, tiefen Abdruck, schwachen Abdruck, mittelstarken Abdruck, dicken Streifen, leichten Strich, dünnen Strich, ganz leichten dünnen Strich, breiten Strich, schmalen Strich* und *mittleren Strich* verwendet. Dieses Ergebnis zeigt, dass eine Interpretation von unscharfen Parametern für ein flexibles Robotersystem notwendig ist.

Die Auswertung der Fragebögen (siehe Abbildung 5.5) hat ergeben, dass die Nutzer es generell leicht fanden, den Roboterarm zu programmieren. Dies zeigt, dass eine natürlich-sprachliche Instruktion präferiert wird, da der Großteil der Nutzer den Roboterarm natürlich-sprachlich instruiert hat. Außerdem fanden sie, dass sich der Roboterarm so verhalten hat, wie sie es sich

Tab. 5.2: Von den Nutzern während der Wizard of Botz Studie genannte unscharfe Verbabhängige bzw. Resultat-abhängige Kraftparameter.

Aufgabe	**Unscharfe Angaben**
T1	drücke leicht/mittelleicht, mittlere Stanztiefe, tiefen/schwachen Abdruck
T2	sachte absetzen, dicken/dünnen Strich
T3	senke sanft, leicht absetzen, verschiebe leicht/vorsichtig
T4	fahre/schiebe langsam

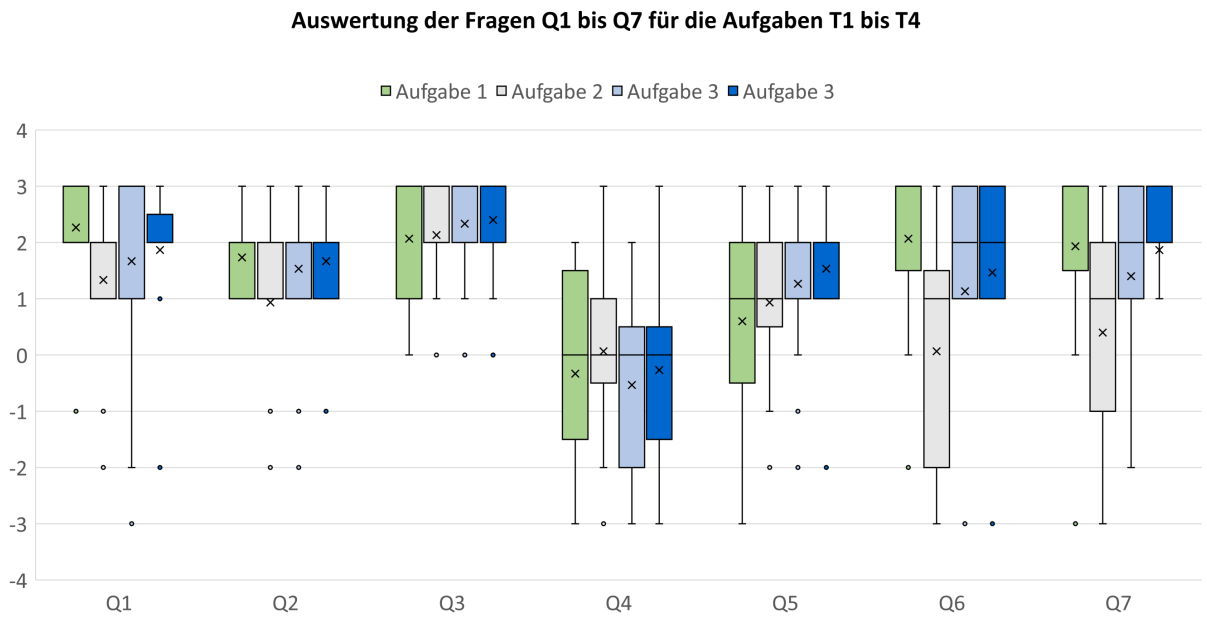


Abb. 5.5: Antworten der Nutzer auf die spezifischen Fragen Q1 - Q7 nach der Bearbeitung jeder Aufgabe.

vorgestellt haben. Die Möglichkeit, den Roboter korrigieren zu können, hatte weder eine durchaus gute, noch eine überaus schlechte Wirkung auf die Nutzer. Insgesamt wurde die Bewegung des Roboters als nicht komisch eingestuft, was zeigt, dass mit dem aktuellen Stand des Systems glaubwürdig Roboterbewegungen simuliert werden können. Außerdem konnte sich die Mehrheit der Nutzer vorstellen, einen Roboter in Zukunft so zu instruieren, und war der Meinung, dass eine sprachbasierte Steuerung vollkommen ausreicht.

Eine weitere Beobachtung war, dass sowohl Nichtexperten als auch Experten in Robotik nicht realisiert haben, dass der Roboterarm durch einen Wizard bewegt wurde. Dies bekräftigt unsere Vermutung, dass es mit unserem Ansatz möglich ist, kraftbasierte Bewegungen realistisch zu simulieren. Dabei entsteht die Illusion eines bereits vollkommen implementierten Systems. Dadurch, dass der Wizard-Roboterarm durch Handführen bewegt wird, ist es zudem möglich einen Nichtexperten als Wizard einzusetzen.

5.1.4 Stärken und Grenzen des Systems

Dadurch, dass der Wizard nicht ein Modell des eigentlichen Roboterarms bewegt, sondern eine identische Version des Roboters bedient, werden eine Reihe an Versuchskonfigurationen ermöglicht. So ist es durch die einfache Instruktion des Roboterarms durch den Nutzer möglich, ein Wizard-of-Oz Experiment durchzuführen, bei dem nicht nur der eigentliche Nutzer als Versuchsperson fungiert, sondern auch der Wizard. Damit könnte man im Rahmen einer Studie beispielsweise herausfinden, wie Nutzer gewisse Bewegungen instruieren, und gleichzeitig Informationen darüber generieren, welche Roboterbewegung Nutzer sich unter der gegebenen Instruktion vorstellen würden.

Dadurch, dass der Arbeitsbereich inklusive Roboterarm auf beiden Seiten identisch ist, ergibt sich zudem die Möglichkeit, Kommunikationskanäle des Wizards einzuschränken oder auch zu erweitern, um damit Systeme zu simulieren, die nur über eine eingeschränkte Sensorik verfügen. Beispiele für eine Einschränkung wären ein Verbinden der Augen oder eine Verwendung von Noise-Canceling Kopfhörern, um den visuellen bzw. akustischen Kommunikationskanal zu sperren und dem Wizard damit alleine eine haptische Rückmeldung zu überlassen. Beispiele für eine Erweiterung sind visuelle oder akustische Rückmeldungen an den Wizard, um Distanzen und Kräfte besser einschätzen zu können. So könnte eine Einblendung der aktuellen *Tool Center Point*-Pose (TCP-Pose) im Raum oder eine Visualisierung der am TCP anliegenden Kraft auf einem Monitor oder auch einer VR-Brille eine hilfreiche Erweiterung sein.

Momentane Grenzen des Ansatzes liegen in der Genauigkeit der erzeugten Bewegungen. Vergleicht man das aktuelle System mit dem System in [Ralph08], so würden die per GUI erzeugten Bewegungen eine höhere Wiederholgenauigkeit bzw. genauere Bahnen generieren. Eine Lösung dieser Einschränkung wäre, dem Wizard zum einen ein Speichern von Zwischenposition zu ermöglichen, welche während der Ausführung angefahren werden können, und zum anderen ein Sperren von Raumachsen umzusetzen, so dass der Wizard im Stande ist, auch gerade Bahnen abfahren zu können. Damit könnten Instruktionen wie “Fahre zurück!” oder “Wiederhole die Bewegung!” robuster umgesetzt werden, wodurch die Illusion besser aufrecht erhalten bleibt.

Doch selbst mit den aktuellen Einschränkungen haben die Resultate dieser Nutzerevaluation gezeigt, dass es sowohl für Nicht-Experten als auch für Experten im Bereich der Robotik die Roboterbewegung nicht unnatürlich gewirkt haben, obwohl die Bewegungen weder eine hohe generelle Genauigkeit, noch eine hohe Wiederholgenaugkeit hatten.

5.2 Fuzzy Force Model

Die aus der vorgehenden Nutzerstudie identifizierten Informationen motivieren die Erstellung eines Ansatzes zur Transformation von unscharf formulierten Kraftparametern auf für das Robotersystem verständliche numerische Werte. Wie in Kapitel 2 beschrieben, ist die Fuzzy Logik ein beliebtes Werkzeug, um symbolische Informationen in subsymbolische Informationen umzuwandeln. Da es für die Abbildung von Kraftparametern bisher noch keinen auf der Fuzzy-Logik basierenden Ansatz gibt, wurde im Rahmen dieser Arbeit erfasst, in wie fern die Fuzzy-Logik auf dieses Gebiet anwendbar ist. Im Folgenden wird der *Fuzzy Force Model* (FFM) Ansatz eingeführt, welcher generiert wurde, um diese Aufgabe zu erledigen. Nach der Spezifikation des Ansatzes wird dessen Anwendbarkeit im Rahmen einer Prototyp-Evaluation dargelegt.

5.2.1 Ansatz

Die vorhergehende Studie hat gezeigt, dass die Bedeutung unscharfer Parameter in sprachlichen Instruktionen von einer Reihe an Nebenbedingungen abhängt. So spielen das Resultat, das Objektmaterial oder auch das ausgerüstete Werkzeug eine Rolle. Der im Folgenden eingeführte Ansatz erlaubt daher eine Interpretation von unscharfen Parametern abhängig von

dem zu manipulierenden Werkstück, dem ausgewählten Werkzeug und übergebenen Kraftspezifikationen. Die Struktur des Ansatzes ist dabei in Abbildung 5.6 visualisiert. Basierend auf der Fuzzy-Logik ermöglicht dieser Ansatz eine Transformation von unscharfen Instruktionen auf kraftbasierte Roboterbewegungen, sofern relevante Objekteigenschaften von Werkstücken, Werkzeugen und dem genutzten Robotersystem, zum Beispiel Materialparameter und deren Zugehörigkeitsfunktionen, bereitgestellt werden können. Dazu werden nicht nur explizit übergebene Parameter berücksichtigt, sondern auch implizit über den Kontext gegebene Eigenschaften.

Im Bezug auf Spiro-Kommandos wird dafür die Verbeigenschaft erweitert, sodass nicht mehr nur ein Verbtyp, sondern auch einen Kraftaufwand zur Verfügung steht. Es ergibt sich also eine erweiterte Menge der abbildbaren Instruktionen $\mathcal{I}_{KVE,Fuzzy} = \mathcal{I}_{KVE} \times \mathcal{E}$, wobei $\mathcal{E} = \{leicht, medium, stark, \dots\}$ eine Teilmenge der Menge an unscharfen Formulierungen darstellt, welche direkt mit einem Verb verknüpft werden. Umschreibungen dieser Parameter durch Resultate oder Anpassungen, wie "leichter" oder "stärker" werden im Rahmen dieser Arbeit nicht betrachtet.

Der grundlegende Aufbau des Modells wird nun anhand einer Beispielinstruktion dargestellt. Danach erfolgt eine nähere Erläuterung der einzelnen Bestandteile. Die Beispielinstruktion ist dabei folgende:

"Wische leicht mit dem Schwamm über den Tisch!".

Daraus ergibt sich das Spiro-Kommando:

$$s = \{((Wischen, leicht), Tisch, Schwamm, \{\})\}. \tag{5.1}$$

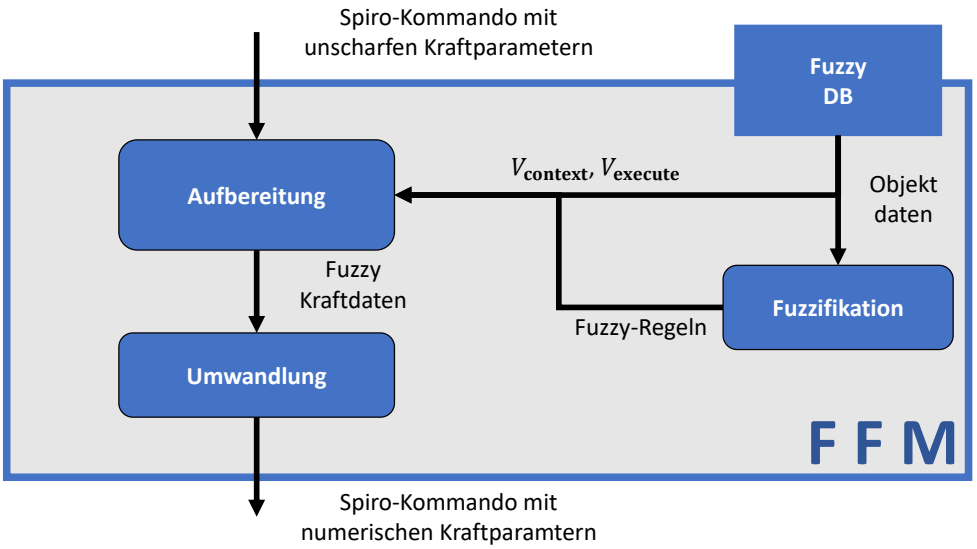


Abb. 5.6: Übersicht über den Fuzzy Force Model Ansatz, welcher unscharfe Parameter aus einer Instruktion in numerische Werte umwandelt und an einen Bewegungsgenerator weiterleitet.

Im Aufbereitungsschritt werden basierend auf den Einträgen des Spiro-Kommandos die notwendigen Fuzzy-Regeln und entsprechenden linguistischen Variablen aus der externen Fuzzy Datenbank geladen, welche einen Teil der Objektdatenbank darstellt. In diesem Fall würden also die Regeln für das Verb *Wischen* und die linguistischen Variablen für Werkzeugparameter ("mit dem Schwamm") sowie für die Werkstückparameter ("den Tisch") geladen werden. Die Objektdaten, im Beispiel die Härte des Schwamms, werden dann in einem Zwischenschritt in eine unscharfe Repräsentation (*weich*) überführt. Zudem erfolgt in der Aufbereitung das logische Schließen, bei dem aus den erfassten Daten unscharfe Kraftdaten für die Ausführung generiert werden. Im Falle des gegebenen Beispiels entspricht das einer Begrenzung der entsprechenden Zugehörigkeitsfunktion der Kraftausgabe für die Härte des Schwamms und der übergebenen Ausprägung *leicht*. Im Umwandlungsschritt erfolgt letztlich die Defuzzifizierung und damit die Transformation der unscharfen Kraftausgabe in einen numerischen Wert (z.B. 2 N). Dieser Wert wird dann für die Bewegungserzeugung aus Kapitel 4 genutzt.

Kontextdatenbank und Fuzzifikation

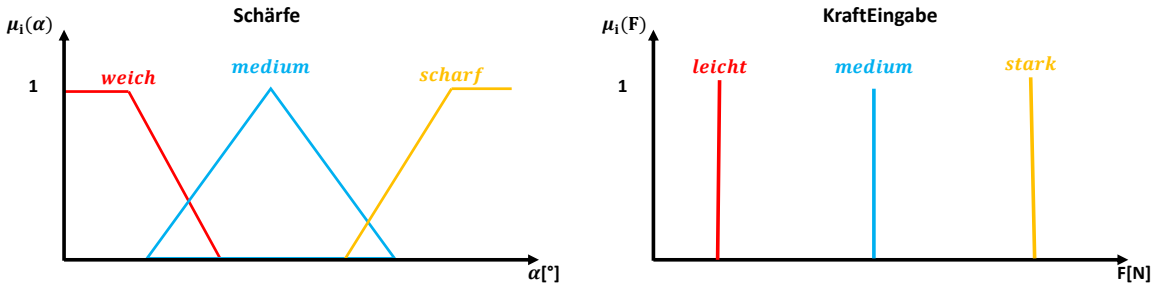
Um die Fuzzy-Logik nutzen zu können ist eine Erweiterung des verfügbaren Kontextwissens auf Objekteigenschaften und deren Bedeutung notwendig. Objekteigenschaften werden dabei in Form von sogenannten linguistischen Variablen V_l dargestellt (siehe Kapitel 2). Im Rahmen dieser Arbeit werden die folgenden drei Untermengen von linguistischen Variablen genutzt:

- Spezifizierte Kraftparameter $V_{\mathrm{F,I}}$, welche einen geforderten Kraftaufwand beschreiben, der entweder explizit vom Nutzer übergeben wird oder als *medium* angenommen wird, sofern nichts angegeben wird
- Linguistischen Variablen für Kontextparameter V_{Kontext}, welche beispielsweise die Härte eines Werkstücks, oder die Geometrie des Werkzeugs beschreiben
- Ausführungsparameter $V_{\mathrm{F,O}}$, welche ebenfalls nicht explizit instruiert werden, sondern über ein logisches Schließen erfasst werden

Die dafür notwendigen Informationen werden in einer Kontextdatenbank hinterlegt, welche Daten über vorhandene Materialien, Werkzeuge und Roboterparameter zusammen mit bewegungsabhängigen Fuzzy-Regeln enthalten, welche für den Reasoning-Schritt in der Aufbereitung notwendig sind.

Obwohl $V_{\mathrm{F,I}}$ bereits als unscharfer Parameter übergeben wird, wird es als sogenannte Singleton-Funktion modelliert (siehe Abbildung 5.7b). Dies entspricht zwar auf der einen Seite einer eindeutigen Abbildung, erlaubt auf der anderen Seite allerdings die Verwendung dieses Parameters in den Fuzzy-Regeln und damit eine Fallunterscheidung. Die Zugehörigkeitsfunktionen für Elemente aus V_{Kontext} und $V_{\mathrm{F,0}}$ werden mit geeigneten Funktionen modelliert, welche für erstere die beschriebenen Objekteigenschaften passend beschreiben und für letztere die Fähigkeiten des verwendeten Roboterarms (Payload, Kinematik) abdecken.

Dabei können die Zugehörigkeitsfunktionen Dreiecks-, Trapezoid- oder weitere Funktionen sein (siehe Abbildung 5.7a). Da für die Auswertung der Fuzzy-Regeln unscharfe Objektparameter



(a) Dreiecks-Zugehörigkeitsfunktion für die linguistische Variable Schärfe.

(b) Singleton-Zugehörigkeitsfunktion für die linguistische Variable *Kraft*.

Abb. 5.7: Beispiele für Zugehörigkeitsfunktionen in Form der linguistischen Variablen Schärfe und Kraft.

benötigt werden, erfolgt zudem eine Fuzzifikation der Objekteigenschaften basierend auf deren Zugehörigkeitsfunktionen. So erhält das System für die Härte-Auswertung des Schwamms beispielsweise *weich* als Resultat, welche für weiteren Schritte notwendig ist.

Aufbereitung

Die Hauptaufgabe der Aufbereitung ist die Umwandlung der relevanten numerischen Objekteigenschaften, die Evaluation der entsprechenden Fuzzy-Regeln und die Erfassung der daraus resultieren Kraftdaten. Generell entspricht der verfolgte Ansatz dabei einem Multi-Input Single-Output (MISO) Prinzip, da aus der Kombination einer Reihe an Eingangsparametern ein einzelner Ausgangsparameter bestimmt wird. Der Kraftaufwand des Roboters $V_{F,O}$ wird dabei für die möglichen Kombinationen von Fuzzy-Regeln definiert, so dass für alle möglichen Kombinationen ein Kraftwert bestimmt werden kann. Ein Beispiel für *Schneiden* ist dabei in den folgenden Regeln gegeben:

$$\text{FALLS KraftEingabe} = \text{leicht UND ObjektHärte} = \text{weich UND WerkzeugSchärfe} = \text{stumpf DANN KraftAusgabe} = \text{medium} \tag{5.2}$$

$$\text{FALLS KraftEingabe} = \text{leicht UND ObjektHärte} = \text{weich UND WerkzeugSchärfe} = \text{medium DANN KraftAusgabe} = \text{leicht} \tag{5.3}$$

$$\text{FALLS KraftEingabe} = \text{leicht UND ObjektHärte} = \text{weich UND WerkzeugSchärfe} = \text{scharf DANN KraftAusgabe} = \text{leicht} \tag{5.4}$$

$$\text{FALLS KraftEingabe} = \text{leicht UND ObjektHärte} = \text{medium UND WerkzeugSchärfe} = \text{stumpf DANN KraftAusgabe} = \text{medium} \tag{5.5}$$

$$\text{FALLS KraftEingabe} = \text{leicht UND ObjektHärte} = \text{medium UND WerkzeugSchärfe} = \text{medium DANN KraftAusgabe} = \text{medium} \tag{5.6}$$

FALLS KraftEingabe = leicht UND ObjektHärte = medium UND
WerkzeugSchärfe = scharf DANN KraftAusgabe = leicht (5.7)

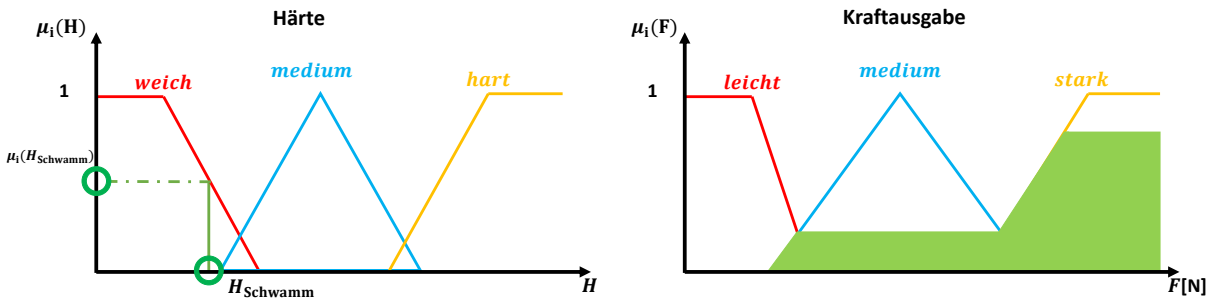
FALLS KraftEingabe = leicht UND ObjektHärte = hart UND
WerkzeugSchärfe = stumpf DANN KraftAusgabe = hart (5.8)

FALLS KraftEingabe = leicht UND ObjektHärte = hart UND
WerkzeugSchärfe = medium DANN KraftAusgabe = hart (5.9)

FALLS KraftEingabe = leicht UND ObjektHärte = hart UND
WerkzeugSchärfe = scharf DANN KraftAusgabe = medium (5.10)

In den Regeln 5.2 bis 5.10 ist die KraftEingabe direkt aus der Instruktion extrahiert worden bzw. standardmäßig auf medium gesetzt worden. ObjektHärte, WerkzeugSchärfe und KraftAusgabe werden jedoch zur Laufzeit aus hinterlegten Objektinformationen und den entsprechenden Zugehörigkeitsfunktionen berechnet (siehe Abbildung 5.8a). Generell sind diese Regeln und Zugehörigkeitsfunktionen nicht allgemeingültig, sondern hängen von der jeweils vorherrschenden Domäne ab. So kann es beispielsweise sein, dass ein Messer in einer Küchendomäne als scharf gilt, in der Medizin jedoch eher die Anforderung medium erfüllen würde. Ein weiterer Diskussionspunkt bei den Regeln ist die Frage, welche Parameter letztendlich berücksichtigt werden. Die hier gewählte Lösung dieses Problems ist, dass nur die Parameter berücksichtigt werden, die auch im Rahmen dieser Arbeit zur Verfügung stehen. Grundsätzlich sollten diese Regeln für weitere Anwendungen von entsprechenden Experten definiert werden, um eine Berücksichtigung aller relevanten Parameter zu garantieren.

Ist eine der Regeln erfüllt und wurde dabei neben der KraftEingabe nur eine weitere Zugehörigkeitsfunktion ausgewertet, so wird dessen Aktivierungsgrad für die Berechnung von KraftAusgabe genutzt. Liegen dahingegen mehrere Zugehörigkeitsfunktionen vor, so wird der resultierende Aktivierungsgrad der jeweiligen KraftAusgabe basierend auf der in Kapitel 2 eingeführten Minimum-Funktion auf den minimalen Aktivierungsgrad gesetzt. Dadurch erfüllt die



(a) Fuzzyfizierung des Härtewertes für einen Schwamm.

(b) Beschränkung und Verknüpfung der Ausgabemenge.

Abb. 5.8: Beispiele für die Berechnung eines Aktivierungsgrades $\mu(H)$ für den Härtewert eines Schwamms (links) und für die Beschränkung und disjunkte Verknüpfung (grüne Fläche) der Ausgabemenge der KraftAusgabe (rechts).

resultierende Kraft den gewünschten Effekt, befindet sich dabei jedoch im unteren Grenzbereich. Damit wird im Idealfall die notwendige Ausführungskraft und damit die Belastung des Roboterarms möglichst gering gehalten. Für die erfüllten Regeln werden abschließend die damit zusammenhängenden Ausgabeterme von KraftAusgabe beschränkt und letztlich miteinander disjunkt verknüpft (siehe Abbildung 5.8, b). Die dadurch entstandene unscharfe Ausgabemenge wird dann an die Umwandlungsebene weitergeleitet, in der daraus ein numerischer Wert generiert wird.

Umwandlung

Die Umwandlung hat hauptsächlich die Aufgabe der Defuzzifizierung der unscharfen Ausgabemenge in Form der KraftAusgabe und der Rückgabe des dadurch numerischen Spiro-Kommandos. Für die Defuzzifizierung ist eine Vielzahl an Methoden möglich, welche sich im Rechenaufwand und zu erwartenden Resultaten unterscheiden. Um im Rahmen dieser Arbeit eine geeignete Methode zu erfassen, wurden daher eine Reihe an Bedingungen definiert, welche idealerweise erfüllt sein sollten:

R1 Keine maximalen Kraftwerte während der Ausführung

R2 Vermeidung von Sensorrauschen

R3 Berücksichtigung eines möglichst breiten Spektrums

Die betrachteten Defuzzifizierungsmethoden sind dabei die in Kapitel 2 vorgestellten Methoden: Max-Methode, Links-Max-Methode, Rechts-Max-Methode, Mittelwert-Max-Methode, Schwerpunkt-Methode und Alpha-Schwerpunkt-Methode. Die Evaluierung der Methoden hat dabei das in Tabelle 5.3 dargelegten Resultat geliefert.

Die Max-Methode und die Rechts-Max-Methode haben dabei bezogen auf R1 eine schlechte Bewertung erhalten, da bei diesen Methoden höhere Resultate bevorzugt werden, was im Vergleich zu den anderen Methoden eher dazu führen kann, dass höhere Kraftwerte erzeugt werden. Bezogen auf R2 kann bei dieser Auswahl lediglich die Alpha-Schwerpunkt-Methode positiv bewertet werden, da diese im Gegensatz zu den anderen eine Rauschunterdrückung integriert hat. Die Nebenbedingung R3 wird letztlich am besten von der Schwerpunkt und der Alpha-Schwerpunkt-

Tab. 5.3: Bewertung der jeweiligen Defuzzifizierungsmethoden basierend auf die Bewertungsmaße R1 bis R3 und die Rechenzeit.

Methode	**Max**	**Links-Max**	**Rechts-Max**	**Mittelwert-Max**	**Schwerpunkt**	**Alpha-Schwerpunkt**
R1	-	+	-	+	+	+
R2	-	-	-	-	-	+
R3	-	-	-	-	+	+
Rechenzeit	-	+	-	+	-	-

Methode umgesetzt, da dort die komplette Fläche unter der disjunkten Verknüpfung der Ausgabemenge berücksichtigt wird und nicht nur einzelne Punkte.

Bezogen auf die Rechenzeit schneiden die beiden Centroid Methoden am schlechtesten ab, da dort der Schwerpunkt der Fläche unter der beschränkten Ausgabemenge berechnet wird und nicht wie bei den anderen Ansätzen lediglich einmal über die Werte der Ausgabemenge iteriert werden muss.

Insgesamt ergibt sich daraus, dass die Modifizierte Centroid-Methode die vielversprechendste Methode bezogen auf die Nebenbedingungen R1 - R3 darstellt. Bezogen auf die Rechenzeit schneidet dabei zwar die Modifizierte Centroid Methode am schlechtesten ab, was jedoch für die Anwendung keine große Rolle spielt, da die Rechenzeit für die Anwendungsfälle vernachlässigbar klein bleibt.

5.2.2 Prototypische Evaluierung

Der vorgestellte Ansatz verfolgt das Ziel, eine Vielzahl an kraftbasierten Bewegungen unscharf verbal instruieren zu können. Um die Funktionsweise und die Validität des Ansatzes zu zeigen, wurde ein Prototyp umgesetzt, welcher für eine Reihe an Verben und variierendem Kraftaufwand Bewegungen generieren kann. Als Beispielanwendung wird damit im Folgenden eine Präge-Bewegung umgesetzt. Die genutzten Materialien sind dabei Knete und Seife, für welche Informationen über deren Härte vorhanden sind. Die betrachtete Nutzerinstruktion ist folgende: “Präge das Objekt leicht!”. Wobei als Objekt einmal die Seife und einmal die Knete genannt wurde.

Die Sprachverarbeitungskomponente wandelt die Instruktion zunächst in ein Spiro-Kommando um, extrahiert das Verb *Präge*, das Werkstück *Objekt*, den Kraftaufwand *leicht* und erschließt sich das Werkzeug *Präge-Werkzeug* aus dem Kontext. Dieses unscharfe Spiro-Kommando wird mittels dem Fuzzy Force Model in ein numerisches Spiro-Kommando umgewandelt.

Der erste Schritt im FFM ist dabei die Beschaffung der notwendigen Fuzzy-Informationen, also den Fuzzy Regeln für die Bewegung *Prägen* und die entsprechenden Zugehörigkeitsfunktionen zusammen mit den folgenden Objektinformationen für die linguistischen Variablen aus der Kontextdatenbank: ObjektHärte und KraftAusgabe. Die Zugehörigkeitsfunktionen für ObjektHärte wurden dabei über die Shore Härteskala [DIN03] modelliert. Die Zugehörigkeitsfunktionen für die KraftAusgabe sind so modelliert, dass sie den Kraftumfang des ausgerüsteten Roboterarms abbilden. Die maximale Kraft soll also ohne potentielle Schäden ausgeübt werden können. In dem hier gewählten Prototyp ist diese Kraft auf 100 N begrenzt. Um das Beispiel übersichtlich zu gestalten wurden die folgenden Regeln dabei für die Seife und die Knete definiert:

$$\begin{aligned}\text{FALLS KraftEingabe} = \text{leicht UND ObjektHärte} = \text{weich UND}\\ \text{WerkzeugHärte} = \text{hart DANN KraftAusgabe} = \text{weich}\end{aligned} \tag{5.11}$$

$$\begin{aligned}\text{FALLS KraftEingabe} = \text{leicht UND ObjektHärte} = \text{medium UND} \\ \text{WerkzeugHärte} = \text{hart DANN KraftAusgabe} = \text{medium}\end{aligned} \tag{5.12}$$

Die Zugehörigkeitsfunktion für die Ausgangskraft ergibt in diesem Fall μ_{soft} bzw. μ_{medium} und ist begrenzt durch die Härtewerte der jeweiligen Materialien. Wird die modifizierte Centroid-Methode angewandt, so ergeben sich die folgenden Kraftwerte für die beiden Objekte: $F_{\text{Knete}} = 10$ N und $F_{\text{Seife}} = 80$ N. Die dadurch entstehenden Bewegungen erzeugen Ergebnisse, welche bei der Knete mit den finalen Zuständen aus der Wizard of Botz Nutzerstudie übereinstimmen. Für die Seife stand kein Vergleich zu Verfügung, da jedoch eine leichte Einprägung zu sehen war, wurde die Ausführung auch hier als erfolgreich angenommen.

5.3 Zusammenfassung

Der wissenschaftliche Zugewinn durch die in diesem Kapitel vorgestellten Ansätze ist zweigeteilt. Zum einen wurde ein Wizard-of-Oz Ansatz eingeführt, welcher unter anderem eine Simulation von kraftbasierten Bewegungen durch Experten und Nichtexperten ermöglicht. Zum anderen wurde ein Modell eingeführt, welches eine Übertragung von unscharf formulierten Instruktionen auf kraftbasierte Bewegungen erlaubt, sofern geeignetes Kontextwissen im Bezug auf Objekteigenschaften vorhanden ist.

Die Evaluierung des Wizard-of-Botz Ansatzes hat die Vermutung bestätigt, dass auch bei der Instruktion von Kraftparametern unscharfe Formulierungen verwendet werden, und zugleich eine Auswahl dieser Parameter geliefert. Dabei hat sich zudem gezeigt, dass sich Formulierungen dabei nicht unbedingt auf das Verb beziehen ("Male leicht!"), sondern auch auf das Resultat ("Einen dünnen Strich"). Neben den Informationen über die Kraftparameter wurde zudem erfasst, dass der Versuchsaufbau eine authentische Simulation von Roboterbewegungen erlaubt, da unter den Nutzern sowohl Nichtexperten als auch Experten im Bereich Robotik nicht bemerkt haben, dass der Roboterarm eigentlich von einem Menschen gesteuert wird. Abschließend wurden Stärken und Grenzen des Ansatzes diskutiert und Lösungsvorschläge für die Grenzen bereitgestellt.

Durch die Verwendung des FFM wurde die Flexibilität der Instruktion mittels Spiro-Kommandos erhöht, da nicht mehr lediglich das reine Verb zur Instruktion zur Verfügung steht, sondern auch noch eine Ausprägung des Verbs mit übergeben werden kann. Dies befreit Nutzer von der Angabe von numerischen Kraftangaben und erlaubt eine menschlichere Instruktion von Bewegungen. Das genutzte Konzept eines Verbs besteht dadurch nicht mehr lediglich aus einem Verbtyp, sondern auch aus einem Kraftaufwand.

Zusammenfassend lässt sich sagen, dass durch die in diesem Kapitel eingeführten Systeme die Menge der abbildbaren Instruktionen und die Menge der abbildbaren Bewegungen erweitert wurde. Durch die Erweiterung von $\mathcal{I}_{KVE}$ auf $\mathcal{I}_{KVE,Fuzzy} = \mathcal{I}_{KVE} \times \mathcal{E}$ und die Erweiterung von $\mathcal{M}_{VPE}$ auf $\mathcal{M}_{VPE,Fuzzy} = \mathcal{M}_{VPE} \times \mathcal{M}_{Fuzzy}$ wurde damit eine Erhöhung der Flexibilität

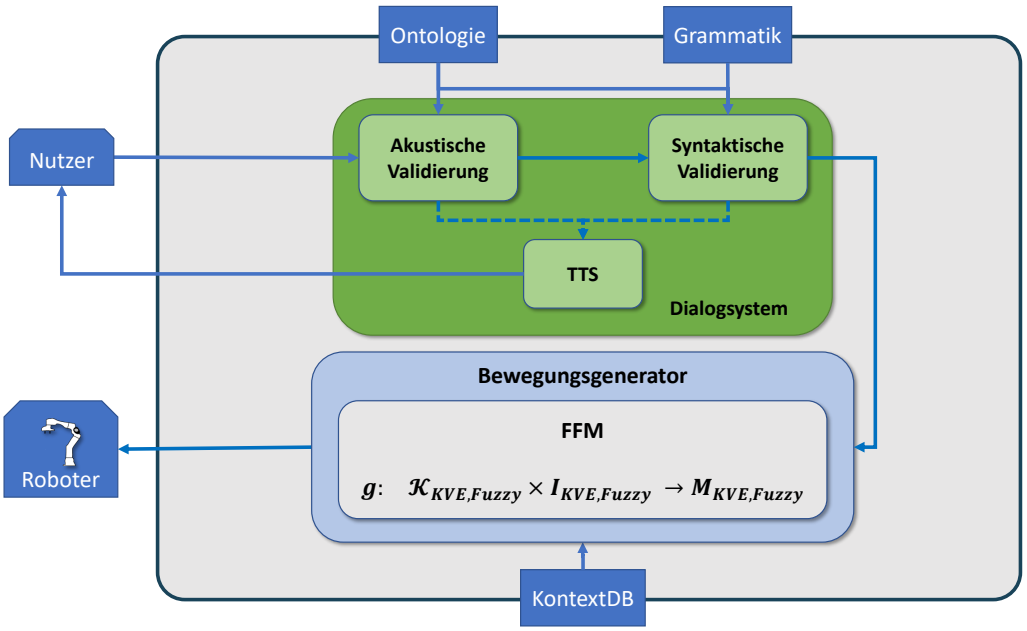


Abb. 5.9: Übersicht über das Gesamtkonzept nach der Erweiterung der KVE um unscharfe Instruktionen bzw. Bewegungen.

des Gesamtansatzes erreicht. Diese Erweiterung ist dabei jedoch nur mit einem umfangreicheren Kontextwissen möglich, da entsprechende Fuzzy-Regeln, Zugehörigkeitsfunktionen und Objektinformationen benötigt werden. Der dadurch erweiterte Zustand des in dieser Arbeit umgesetzten Gesamtkonzepts ist in Abbildung 5.9 dargestellt.

Eine weitere Betrachtung des Wizard-of-Botz Konzepts wäre in Form eines direkten Vergleichs mit einem GUI-basierten Ansatz wie in [Ralph08] interessant. Außerdem würde eine Untersuchung der Reduktion der dem Wizard zur Verfügung stehenden Kommunikationskanäle wertvolle Erkenntnisse liefern. Bezogen auf das FFM bestehen potentielle zukünftige Erweiterungen des Systems aus der Interpretation von resultatabhängigen Parametern und der Anpassung von bereits instruierten Bewegungen über einzelne Wörter wie *stärker* oder *mit mehr Kraft*.

Kapitel 6

Affordanzbasierte Validierung

Bis jetzt wird in dieser Arbeit davon ausgegangen, dass die vom Nutzer übergebenen Instruktionen valide sind. Valide bedeutet hier, dass eine Instruktion alle relevanten Parameter entweder enthält oder Parameter eindeutig aus dem Kontext erschlossen werden können. Außerdem bedeutet es, dass die daraus resultierende Bewegung vom System in der gegebenen Situation ausgeführt werden kann. In realen Anwendungen ist nicht gewährleistet, dass zu jeder Zeit valide Instruktionen vorliegen, da sich Nutzer entweder nicht über die Fähigkeiten des Systems und deren Grenzen bewusst sind, unabsichtlich fehlerhafte Instruktionen übergeben werden oder Missverständnisse auftreten. Deshalb bedarf es einer Validierung von Instruktionen, um mögliche Fehler abzufangen, weil diese Schäden an Objekten im Arbeitsraum bedeuten können.

Neben einer Validierung auf Kanal-, Signal- oder Intentionsebene (siehe Kapitel 1), spielt eine Validierung auf der Konversationebene eine große Rolle. Auf dieser Ebene wird überprüft, in wie fern Bestandteile einer Instruktion in einem Anwendungsfall valide bezüglich einer Metrik sind. Das Konzept der Affordanzen (siehe Kapitel 2) ist ein vielversprechender Kandidat, um eine solche Metrik zu definieren, da damit bewegungsbezogene Fähigkeiten von Instruktionsparametern modelliert werden können. Ein Nebeneffekt davon ist, dass solch eine Metrik zudem für eine Mehrdeutigkeitsauflösung genutzt werden kann, indem nicht valide Kandidaten ausgeschlossen werden können. Außerdem besteht die Möglichkeit valide Kandidaten bezogen auf diese Metrik zu bewerten. Dieses Kapitel befasst sich daher mit der Frage:

F4 In wie weit können Affordanzen zur Auflösung von Mehrdeutigkeiten in Instruktionen und zur Validierung von Instruktionen genutzt werden?

Das angestrebte Ziel ist also die Erweiterung der Abbildung g, so dass auch unvollständige und nicht valide Instruktionen, unter der Erweiterung des Kontextwissens auf Affordanzen, zu validen Roboterbewegungen oder klärenden Rückmeldungen führen. Dafür wird zunächst ein Ansatz vorgestellt, welcher eine affordanzbasierte Validierung ermöglicht und zur Auflösung von Mehrdeutigkeiten genutzt werden kann, und in [Wölfel20a] veröffentlicht ist. Die Nützlichkeit dieses Ansatzes wird anhand von Ergebnissen einer Nutzerstudie dargelegt. In einer Zusammenfassung werden die Erkenntnisse dieses Kapitels diskutiert und Einschränkungen erörtert.

6.1 Ansatz

Der vorgestellte Ansatz ermöglicht eine Validierung und Mehrdeutigkeitsauflösung von Informationen aus natürlich-sprachlichen Instruktionen. Er setzt sich dabei aus den folgenden vier Hauptbestandteilen zusammen: Der Kommunikation, der Identifikation, der Validierung und der Aufbereitung. Außerdem wird die Kontextdatenbank um Informationen zu objektabhängigen und verbabhängigen Affordanzen erweitert (siehe Abbildung 6.1).

Ausgangspunkt ist eine sprachliche Nutzerinstruktion, welche im Kommunikationsschritt in ein vollständiges oder unvollständiges Spiro-Kommando umgewandelt wird. Ein unvollständiges Kommando liegt dann vor, wenn ein, für eine Bewegung, relevanter Parameter entweder nicht vorhanden oder mehrdeutig ist. Diese Information dient als Eingabe für die Identifikation, welche für die Erfassung der instruierten Bewegung zuständig ist. Dabei wird entweder anhand des Verbs auf die für die Abbildbarkeit notwendigen Parameter geschlossen oder anhand der übergebenen Objekte ein wahrscheinliches Verb erschlossen. Diese Zuordnung geschieht dabei basierend auf den Affordanzen, welche für die jeweiligen Komponenten definiert werden. Im Rahmen der Validierung wird dann überprüft, ob die Affordanzen in der gegebenen Situation erfüllt sind, und eine darauf basierende Bewertung der Komponenten findet statt. Im Falle einer eindeutigen Instruktion wird die validierte Information an den in Kapitel 5 eingeführten Bewegungsgenerator übergeben, damit dieser daraus ein KVE erzeugt. Falls eine Mehrdeutigkeit oder auch eine in der gegebenen Situation inkorrekte Bewegung erkannt wurde, wird die zugrundeliegende Affordanz extrahiert und die Information an die Aufbereitung weitergeleitet. Dort wird eine entsprechende verbale Rückmeldung zur Auflösung der Mehrdeutigkeit bzw. zur Information über die Ungültigkeit der Instruktionen erzeugt und an den Nutzer weiter gegeben. Die einzelnen Komponenten werden nun näher erklärt.

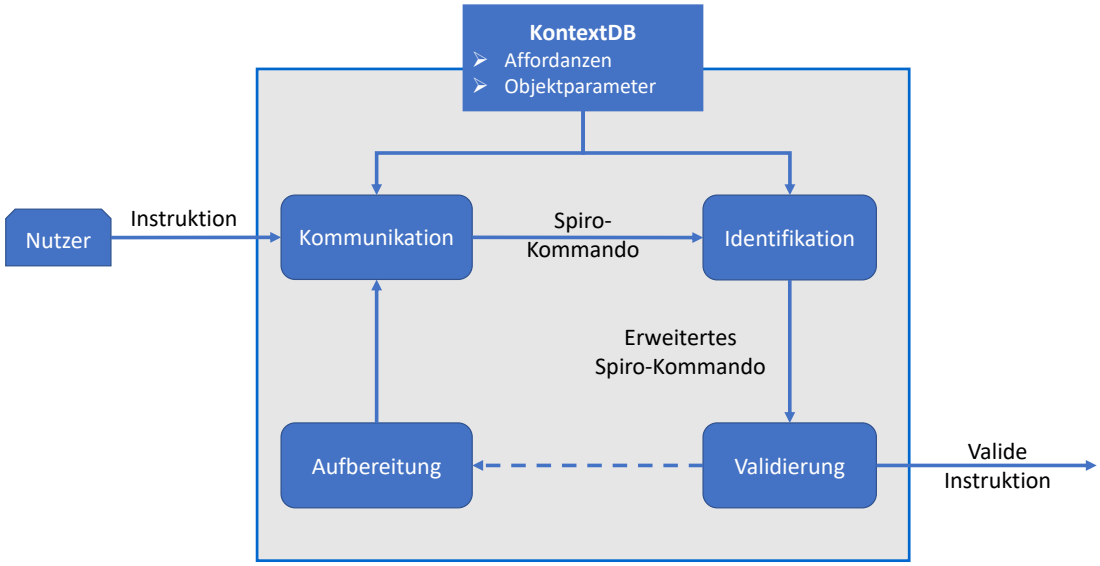


Abb. 6.1: Übersicht über den eingeführten Ansatz. Instruktionen werden zunächst in ein Spiro-Kommando umgewandelt, im Identifikationsschritt und Validierungsschritt auf Mehrdeutigkeit und Validität geprüft und, falls möglich, als valide Instruktion weitergeleitet. Alternativ wird der Nutzer über Probleme oder Mehrdeutigkeiten informiert.

6.1.1 Affordanz-Definition

Wie in Kapitel 2 beschrieben, existiert eine Vielzahl an Möglichkeiten, um Affordanzen für Objekte zu definieren und diese zu erfassen. In dieser Arbeit wird die Menge der Affordanzen $\mathcal{A}$ angelehnt an die Definition in [Cruz16] nicht nur als Tripel von Objekten, Aktionen und Effekten definiert, sondern auch der Zustand des Roboters berücksichtigt. Unter diesen Affordanzen gibt es zum einen jene, welche sich während einer Interaktion nicht ändern, im Folgenden statisch genannt. Zum anderen gibt es jene, welche sich aufgrund von äußeren Einflüssen ändern können und im Folgenden als dynamisch bezeichnet werden. Die Änderung wird dabei über einen Aktivierungsgrad $r \in [0, 1]$ dargestellt, wobei 0 einer Nichterfülltheit einer Affordanz entspricht und 1 einer Anwesenheit der Affordanz entspricht. Je höher r ist, desto stärker ist eine Affordanz also erfüllt. Formal werden diese erweiterten Affordanzen wie folgt definiert:

$$\mathcal{A}_r = \{(a, r) | a \in \mathcal{A}, r \in [0; 1]\}. \tag{6.1}$$

Beispiele für solche Affordanzen sind in Abbildung 6.2 dargestellt. Dabei gelten hier die *Greifbarkeit* und die *Hebbarkeit* als statische Affordanzen. Die Greifbarkeit wird hier darüber bestimmt, ob eine Greifpose für den genutzten Roboter in der KontextDB hinterlegt ist oder nicht. Die Hebbarkeit wird im Rahmen dieser Arbeit basierend auf der maximalen Hebelast des Roboterarms bestimmt. Da dieses System nicht darauf ausgelegt ist, dass sich der genutzte Roboterarm während der Bedienung ändert, kann die Affordanz also entweder 1 oder 0 sein. Die *Erreichbarkeit* entspricht dagegen einer dynamischen Affordanz, da sie darüber berechnet wird, ob die Greifpose für das Objekt in der aktuellen Pose vom Roboterarm angefahren werden kann. Kurz gesagt also, ob es eine Lösung der Inversen Kinematik für die entsprechende Objektpose gibt.

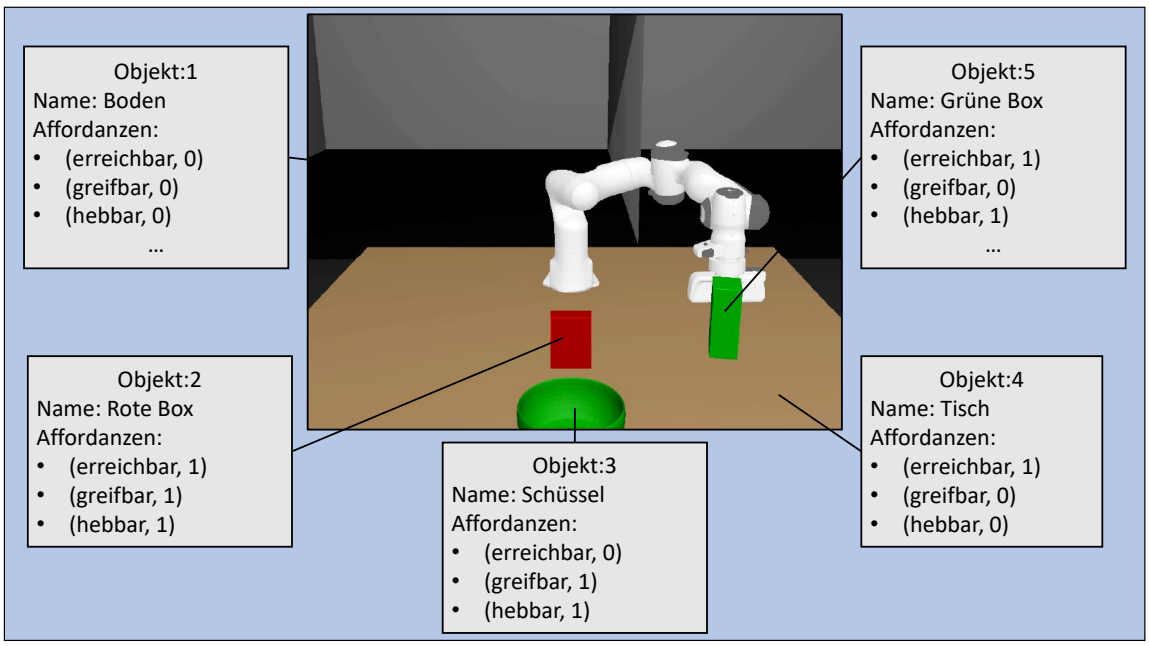


Abb. 6.2: Beispiele für statische und dynamische Affordanzen von Objekten in einem Arbeitsraum.

Generell ist die Einteilung in statische und dynamische Affordanzen stark von der Komplexität des Systems abhängig. Evaluiert man Affordanzen nur basierend auf einem Objekt, so kann eine Hebbarkeit als statische Affordanz modelliert werden. Auf diese Weise würden dann allerdings Situationen nicht abgedeckt werden, bei denen auf diesem Objekt weitere Objekte platziert werden, so dass eine Hebbarkeit durch externe Einflüsse beeinflusst wird. Zudem können Affordanzen zwar zum Startzeitpunkt erfüllt sein, sich innerhalb von einem dynamischen System jedoch während der Ausführung verändern. Ein Ansatz, der eine solche Dynamik berücksichtigt, wird im nächsten Kapitel näher beschrieben. Hier gilt es zunächst, Affordanzen in Form von Vorbedingungen zu nutzen, wodurch das Qualifizierungsproblem beachtet werden muss, da die Auswahl an Vorbedingungen nicht willkürlich getroffen werden sollte. Deshalb wird die Auswahl an Vorbedingungen in dieser Arbeit auf jene beschränkt, welche vom System ausgewertet werden können.

6.1.2 Kommunikation

Der Verarbeitungsschritt der Kommunikation hat generell sowohl die Aufgabe der Transformation von gesprochenen Instruktionen in Spiro-Kommandos, als auch der Transformation von erfassten Instruktionsfehlern und Mehrdeutigkeiten in natürlich-sprachliche Rückmeldungen.

Letztere ist die Erzeugung von verbalen Rückmeldungen an den Nutzer für die folgenden vier Fälle: Jeder Parameter der Instruktion wurde korrekt übergeben und die Instruktion konnte vollständig auf eine eindeutige Bewegung abgebildet werden (C1), eine Mehrdeutigkeit ist aufgetreten, das System konnte jedoch eine eindeutige Auflösung erstellen (C2), das System war nicht im Stande eine Auflösung der Mehrdeutigkeit zu generieren (C3), das System hat eine ungültige Instruktion identifiziert (C4) (siehe Abbildung 6.3).

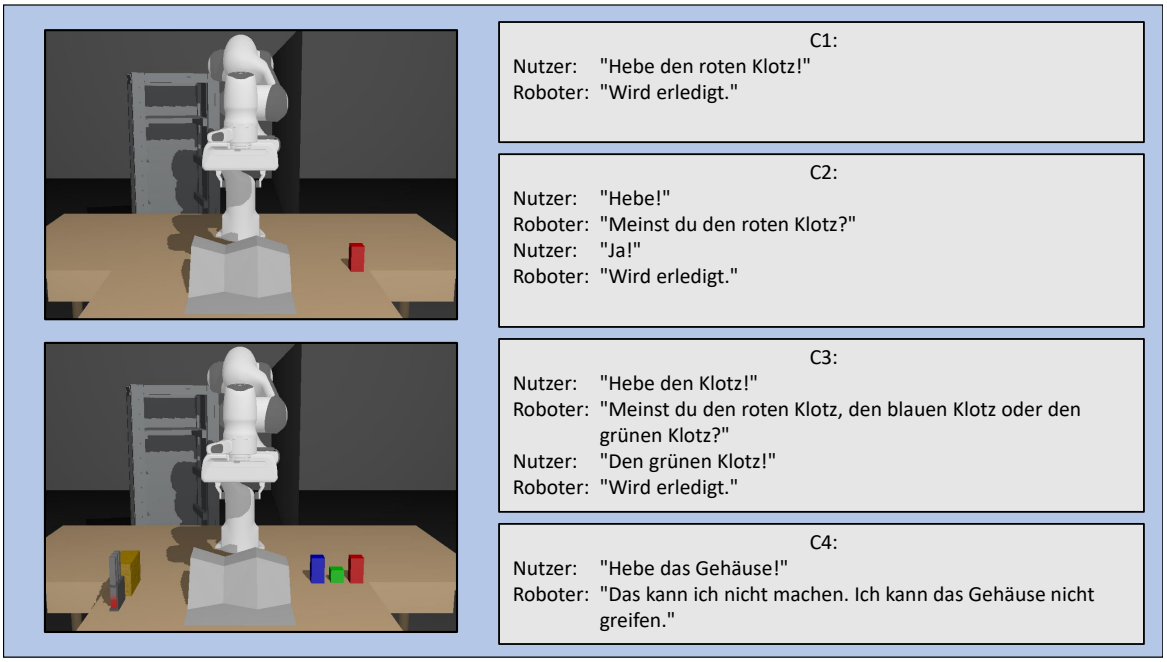


Abb. 6.3: Beispielfälle für eine eindeutige Instruktion (C1), eine aufgelöste Mehrdeutigkeit (C2), bestehende Mehrdeutigkeit (C3) und nicht valide Instruktion (C4).

Im Fall C1 wird eine zustimmende Antwort, wie “Wird erledigt!” an den Nutzer weitergeleitet. Im Fall C2 paraphrasiert das System die Instruktionen mit dem aufgelösten Parameter und gibt sie als Rückfrage an den Nutzer zurück. So wird dem Nutzer die Möglichkeit offen gehalten die Bewegung abzubrechen, indem er “Nein!” oder “Stop!” sagt. Im Fall C3 werden dem Nutzer die möglichen Belegungen des mehrdeutigen Parameters in Form einer Frage zurückgegeben, sodass der Nutzer explizit den Parameter spezifiziert. Diese Frage wird dabei basierend auf dem in [Williams19] eingeführten Design-Prinzipien erstellt. Dort wurde erfasst, dass im Falle einer Mehrdeutigkeit die möglichen Optionen lediglich dann komplett präsentiert werden, wenn es nicht mehr als drei Möglichkeiten sind. Sonst sollen Ja/Nein oder W-Fragen generiert werden. Im Fall C4 wird der Nutzer darüber informiert, dass die Instruktion in dem betrachteten Fall nicht auf eine valide Bewegung abgebildet werden kann und der problematische Parameter wird an den Nutzer übergeben. Die möglichen Rückmeldungen können dabei Roboter- oder Werkstück-abhängig formuliert werden. Eine Untersuchung der Präferenz zu einer dieser Abhängigkeit erfolgt in Kapitel 8.

Als Eigenschaft zur Mehrdeutigkeitsauflösung werden dabei die Parameter gewählt, welche eine Unterteilung erlauben, also beispielsweise die Objektfarbe, der Objekttyp oder die Größe. Sind alle natürlich-sprachlichen Parameter erschöpft, bleibt letztlich die Auflösung über die Objekt-position, da diese hier für Objekte als eindeutig definiert wird. Dies stellt auch eine Grenze der rein sprachlichen Kommunikation gegenüber einer visuellen Darstellung dar. Im visuellen Fall könnte man die mehrdeutigen Objekte semantisch erweitern, indem man ihnen Nummern zuweist und den Nutzer dann nach der entsprechenden Nummer fragt. Diese Möglichkeit steht in der sprachlichen Kommunikation nicht zur Verfügung.

6.1.3 Identifikation

Der Identifikationsschritt hat die Aufgabe, aus einem möglicherweise unvollständigen Spiro-Kommando und zugehörigem Kontextwissen das kommunizierte Verb und die dazu gehörigen Parameter zu identifizieren. Dafür sind vom System Bewegungsvorlagen gegeben, welche zu einem Verb jeweils die dazugehörigen notwendigen Parameter bzw. Valenzen angeben. Tritt der Fall auf, dass zu wenig Parameter oder mehrdeutige Parameter übergeben wurden, wird versucht das Spiro-Kommando mit Hilfe der in [Gorniak07] beschriebenen *Affordanz Filterung* zu befüllen, sofern dies notwendig ist. Diese Filterung ist hier über die folgenden beiden Schritte umgesetzt: Der Bewegungsidentifikation in Form der Extraktion eines Verbs (T1) und der Parameteridentifikation in Form von Affordanz-bezogenen Abhängigkeiten der restlichen Instruktionsparameter zu dem Verb (T2).

Der Schritt T1 dient der Erschließung des Verbs in der Instruktion, wobei generell zwei Fälle auftreten können: Das Verb wird explizit übergeben (ist in der Instruktion enthalten) oder das Verb wird implizit übergeben (ergibt sich aus dem Kontext). Der erste Fall ist trivial, da lediglich die Bewegungsvorlage zu dem entsprechenden Verb aus der Datenbank geladen werden muss und mit T2 fortgefahren wird. Die Verknüpfung von Verben und Bewegungen wird hier als eindeutig angenommen. Dadurch ist hier keine weitere Mehrdeutigkeit möglich. Eine Abbildung von Verben auf mehrere Bewegungen, wie zum Beispiel, dass bei *Geben* sowohl ein *Schieben*,

als auch ein *Platzieren* umgesetzt werden kann, sollte zukünftig jedoch auch näher betrachtet werden.

Im zweiten Fall ist das Verb nicht direkt vorhanden, was wiederum zwei Möglichkeiten bereitstellt: Der Nutzer gibt das Verb nicht an, weil es sich mit dem Verb aus einer vorangehenden Instruktion deckt oder er gibt es nicht an, weil es für ihn offensichtlich ist, welches Verb gemeint ist. Die erste Möglichkeit tritt auf, wenn repetitive Aufgaben instruiert werden. Ein Anwendungsfall wäre die Instruktionsfolge: “Schiebe den Klotz nach rechts!” - “Und jetzt [schiebe] den Zylinder!”. Dieses Konstrukt entspricht einer sprachlichen Ellipse und es sind eine Reihe an Ansätzen vorhanden, welche solche Instruktionsfolgen auflösen können indem sie ein Kurzzeitgedächtnis simulieren. Hier liegt der Schwerpunkt auf der zweiten Möglichkeit, für welche nun ein affordanzbasierter Ansatz zur Erfassung wahrscheinlicher Parameter eingeführt wird.

Der Grundansatz ist, über Zusatzinformationen der übergebenen Parameter auf potentielle Kandidaten für das Verb zu schließen (siehe Abbildung 6.4). Dabei können generell drei Fälle auftreten: Es wird ein eindeutiges Verb gefunden, es werden eine Reihe an möglichen Verben gefunden und es wird kein passendes Verb identifiziert. Im ersten Fall wird die aufgelöste Verbeigenschaft übernommen und mit T2 fortgefahren. Im zweiten Fall wird eine Rückfrage an den Nutzer generiert, in der der Nutzer gefragt wird, welche der möglichen Verben gemeint war. Im letzten Fall wird der Nutzer darüber informiert, dass mit diesem Objekt in der aktuellen Situation keine Manipulation durchgeführt werden kann.

Ein Beispiel dafür wäre die Instruktion: “Die große Schraube!”. Anstatt gar nicht auf die Instruktion zu antworten oder die offene Frage zu stellen: “Was soll ich damit machen?”, werden hier die validen Affordanzen des spezifizierten Werkstücks erfasst und es wird damit versucht die Mehrdeutigkeit des Verbs aufzulösen. Offene Rückfragen werden hier wenn möglich vermieden,

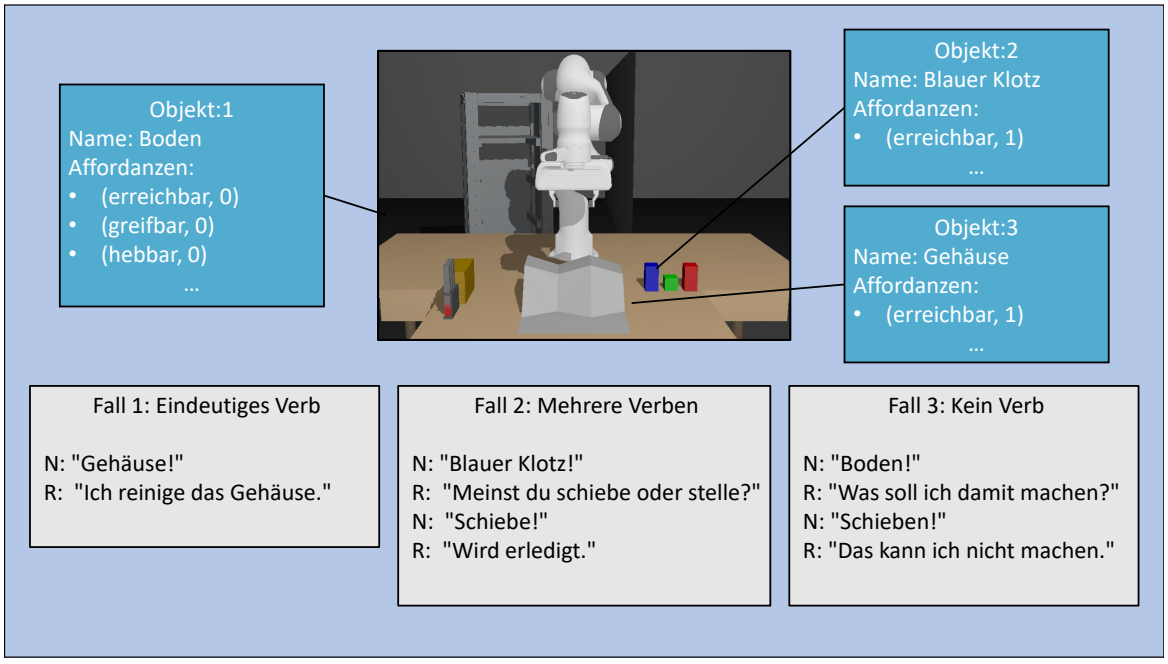


Abb. 6.4: Beispiele für die möglichen Identifikationsschritte für ein nicht spezifiziertes Verb über Objekt-Affordanzen.

da dort die Möglichkeit besteht, dass Wörter genutzt werden, welche dem System nicht bekannt sind und dadurch der Dialog verlängert wird, was zu unnötig langen Instruktionen und damit zu Kosten führen kann.

Wurde das Verb erfolgreich identifiziert, gilt es noch die verbleibenden Parameter, wie das Werkstück oder einen geeigneten Zielzustand, über die Bewegungsvorlage aus der Datenbank zu identifizieren. Im trivialen Fall sind auch hier alle Parameter in der Instruktion übergeben worden und ein erweitertes Tupel kann generiert und an die Validierung weitergegeben werden. Ist einer der Parameter mehrdeutig, da er nicht angegeben wurde oder nicht ausreichend spezifiziert wurde, muss eine Auflösung erfolgen. Um geeignete Kandidaten zu identifizieren, werden erneut Affordanzen betrachtet, nur das diesmal Werkstücke oder Werkzeuge und nicht Verben identifiziert werden müssen. Wird in der Menge der Objekte ein Kandidat erfasst, bei welchem gilt $a_i > 0$ wird dieser zu den Kandidaten hinzugefügt. Enthält diese Menge der Kandidaten mehr als ein Element, so wird der entsprechende Paramter des Spiro-Kommandos mit diesen Kandidaten befüllt und dem erweiterten Spiro-Kommando hinzugefügt. Dieses erweiterte Spiro-Kommando wird dann im Validierungsschritt weiterverarbeitet.

Ein Beispiel dafür ist die Instruktion: “Schiebe nach links!”. Eine Möglichkeit wäre den Nutzer zu fragen: “Welches Objekt?”. Dabei läuft man allerdings wieder Gefahr, dass der Nutzer das Objekt falsch benennt oder nicht weiß, wie er es nennen soll, und die Situation als unangenehm empfindet. Mit diesem Ansatz können die möglichen Objekte identifizieren und im Idealfall ein eindeutiges Objekt erfassen, da es zu diesem Zeitpunkt eventuell nur ein Objekt gibt, welches geschoben werden kann.

6.1.4 Validierung

Im Validierungsschritt werden die Instruktionsparameter basierend auf ihren dynamischen Affordanzen in der vorherrschenden Situation bewertet. Die Ausgabe der Validierung ist eine Einordnung der einzelnen Parameter des Spiro-Kommandos basierend auf den Kategorien: VALIDE, NICHT VALIDE, MEHRDEUTIG und AUFGELÖST. Tritt der Fall auf, dass in der Identifikation bereits ein Kommando als NICHT VALIDE kategorisiert wird, weil beispielsweise kein mögliches Verb gefunden wurde, so wird das Kommando gleich an die Interpretation weitergeleitet.

Bei der Kategorisierung werden nun nicht nur statische Affordanzen, sondern auch dynamische Affordanzen der Komponenten betrachtet, da während der Manipulation eines Objekts beispielsweise die Erreichbarkeit anderer Objekte in darauffolgenden Manipulationen verändert werden können. Ein Parameter wird als VALIDE markiert, falls er die vom Verb geforderten Affordanzen enthält und diese in der gegebenen Situation auch ausreichend erfüllt sind. Für das Verb *Heben* werden beispielsweise die Affordanzen *Erreichbar, Greifbar* und *Hebbar* benötigt. Ein Parameter wird wiederum als NICHT VALIDE markiert, falls er in der gegebenen Situation nicht über die erforderliche Affordanz verfügt. Dies kann beispielsweise gelten, wenn ein Objekt gegriffen werden soll, die definierte Greifpose jedoch in der aktuellen Objekt-Pose nicht kollisionsfrei erreichbar ist. Ein Parameter wird als MEHRDEUTIG markiert, falls die Kandidaten-Menge des Slots mehr als ein Element enthält und es keinen Kandidaten gibt, der die übrigen Kandidaten bezogen auf die Affordanzen überragt. Für den also gilt:

$$r_i >> r_j, i \neq j. \tag{6.2}$$

Solch ein Fall könnte auftreten, wenn es zwar mehrere Klötze im Arbeitsraum gibt, jedoch nur einer davon kollisionsfrei erreichbar ist. Ein Parameter wird schließlich als AUFGELÖST kategorisiert, falls eine Auflösung der Mehrdeutigkeit über die Ungleichung 6.2 möglich war. Die kategorisierten, erweiterten Spiro-Kommandos werden dann anschließend im Interpretationsschritt verwendet, um den Nutzer über die Reaktion des Systems auf die Instruktion zu informieren.

6.1.5 Interpretation

Die Aufgabe der Interpretation besteht abhängig vom Ergebnis der Validierung aus einem der folgenden Schritte:

- Weiterleiten des Spiro-Kommandos an den Bewegungsgenerator
- Ein Weiterleiten des Spiro-Kommandos sowie ein Weiterleiten einer Bestätigung an die Kommunikation
- Weiterleiten einer Mehrdeutigkeit bzw. eines Fehlers an die Kommunikation

Abhängig von der Bewertung der Komponenten des erweiterten Kommandos werden vier verschiedene Arten von Nachrichten unterschieden. Tritt der Fall auf, dass für alle Parameter der Instruktion die Markierung VALIDE gilt, wird eine Bestätigungsnachricht generiert und an den Dialogmanager weitergeleitet. Das Spiro-Kommando wird an den in Kapitel 5 beschriebene Bewegungsgenerator übergeben. Falls das Kommando neben Parametern mit der Kategorie VALIDE auch Parameter mit der Kategorie AUFGELÖST enthält, wird das Spiro-Kommando an den Bewegungsgenerator weitergeleitet, und eine Bestätigungsnachricht, welche die aufgelöste Instruktion paraphrasiert, wird an die Kommunikationskomponente gesendet. Damit wird nach einer Bestätigung der Auflösung gefragt und dadurch dem Nutzer die Möglichkeit offen gehalten, die Bewegung abzubrechen. Falls für einen Parameter der Fall NICHT VALIDE auftritt, wird eine Abbruchnachricht an die Kommunikationskomponente gesendet, welche die problematische Komponente des Kommandos enthält. Ein Beispiel hierfür ist die Rückmeldung: “Ich kann den Tisch nicht heben.” auf die Instruktion “Hebe den Tisch!”. Falls einer der Parameter die Markierung MEHRDEUTIG enthält, wird eine Klärungsnachricht erzeugt und an den Dialogmanager gesendet, welche die Kandidaten für eine mehrdeutige Komponente enthält.

Durch diese Schritte wird gewährleistet, dass keine fehlerhafte Instruktion an das System weitergeleitet wird und der Nutzer stets natürlichsprachlich darüber informiert wird, warum das System wie reagiert.

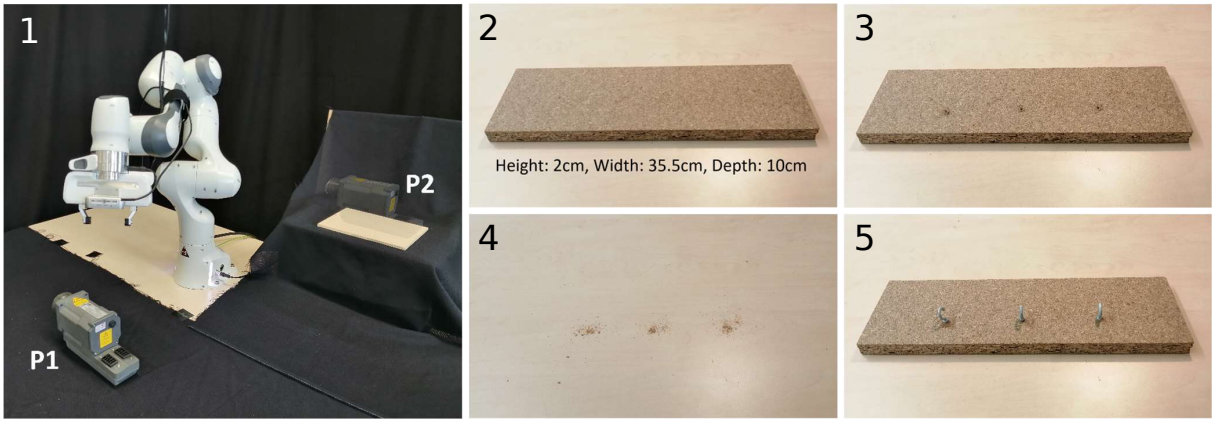


Abb. 6.5: 1: Anwendungsfall zur Instruktion des Roboterarms. 2 - 4: Zwischenschritte des zu bauenden Schlüsselbretts. 5: Finales Schlüsselbrett.

6.2 Nutzerevaluation

Die nachfolgend beschriebene Nutzerstudie dient zum einen dazu, Information darüber zu erhalten, wie Nutzer auf unvollständige und mehrdeutige Instruktionen reagieren, um das Verhalten mit der hier konzipierten Rückmeldung zu vergleichen. Zum anderen wird betrachtet, warum die Nutzer in dieser Art auf solche Instruktionen reagieren, um die hier konzipierte Mehrdeutigkeitsauflösung damit vergleichen zu können. Nach einem Überblick über den gewählten Aufbau (Kapitel 6.2.1), erfolgt eine Evaluierung und Diskussion der Ergebnisse (Kapitel 6.2.2).

6.2.1 Aufbau

Die Studie wurde rein digital durchgeführt mit Hilfe des Online Tools *Survey Monkey*[1]. Der generelle Ablauf bestand aus zwei Teilen.

Im ersten Teil der Studie wurde ein kurzes Anwendungsszenario gewählt, welches die Notwendigkeit der Validierung von selbst einfachen Instruktionen darlegt und somit die Nützlichkeit des hier vorgestellten Ansatzes unterstreicht. Den Nutzern wurde dafür die in Abbildung 6.5-1 dargestellte Arbeitsumgebung zusammen mit der Aufgabe präsentiert, den Roboter dazu zu bringen, einen Motor mit einem Gewicht von 4 kg von der Position P1 zur Position P2 zu befördern. Die Intention dieser Aufgabe lag darin, zu untersuchen, in wie fern sich Nutzer über die Fähigkeiten von Roboterarmen oder in diesem Fall Leichtbaurobotern bewusst sind. In diesem Fall also, ob der Leichtbauroboter überhaupt im Stande ist, diesen Motor zu heben.

Im zweiten Teil nehmen die Nutzer ohne es zu wissen die Rolle eines Leichtbauroboters bzw. eines Dialogsystems ein, welches den Roboter steuert. Zusammen mit einem weiteren Arbeiter haben sie die Aufgabe ein Schlüsselbrett herzustellen (siehe Abbildung 6.5, 2 - 5). Die Unteraufgaben für die Nutzer wurden dabei so gewählt, dass sie von einem Roboterarm entweder effizienter und genauer ausgeführt werden können (Zeichnen, Bohren) oder als mühsam angesehen werden (Arbeitsplatz reinigen). Für jede instruierte Teilaufgabe erhalten die Nutzer dabei ein Bild vom aktuellen Zustand des Schlüsselbretts, sowie die Instruktion des anderen Arbeiters und die zu

[1] https://www.surveymonkey.de/

diesem Zeitpunkt verfügbaren Werkstücke und Werkzeuge. Für jede dieser Situationen sollten die Nutzer dann eine der folgenden Reaktionsarten (RT1 - RT6) wählen, welche in ausgeschriebener Form zu Verfügung standen:

RT1 Eine verbale Aufzählung von drei möglichen Kandidaten und keine Ausführung

RT2 Eine verbale Aufzählung von zwei möglichen Kandidaten und keine Ausführung

RT3 Eine Rückfrage im Bezug auf mehrdeutige/fehlende Kandidaten und keine Ausführung

RT4 Eine verbale Rückmeldung in Form der aufgelösten Instruktion und eine Ausführung

RT5 Ein verbales Abnicken und eine Ausführung

RT6 Keine verbale Rückmeldung und keine Ausführung

In einer vorhergehenden Prototyp-Studie wurde eine offene Frage im Bezug auf die Reaktion gestellt. Da die Antworten auf diese Frage zu weit auseinander gingen wurde deshalb auf diese Alternative ausgewichen.

Insgesamt wurden den Nutzern die folgenden drei Situationen präsentiert:

1. Instruktion: “Zeichne das Muster M1 auf das Brett!”, Werkzeuge: Bleistift, Text-Marker und Permanent-Marker
2. Instruktion: “Stecke die Haken in die Löcher!”, Werkstücke: Haken mit unterschiedlichen Längen
3. Instruktion: “Säubere den Arbeitsbereich!”, Werkzeuge: Schwamm, Lappen und Handfeger

Zwischen jeder Situation wurden die Nutzer zudem darüber informiert, was der Mitarbeiter in der Zwischenzeit erledigt hat. Zwischen 1 und 2, dass der Mitarbeiter die entsprechenden Löcher gebohrt hat und zwischen 2 und 3, dass der Mitarbeiter das Schlüsselbrett zur weiteren Verarbeitung an eine andere Station gebracht hat.

6.2.2 Ergebnis

Insgesamt haben 15 Nutzer an dem ersten Teil der Studie teilgenommen. Sowohl Experten als auch Nichtexperten im Bereich der Robotik haben teilgenommen. Trotzdem ist keinem der Nutzer aufgefallen, dass der Motor die Maximallast des Roboterarms überschreiten würde, obwohl das Gewicht angegeben war. Jeder Nutzer hat den Roboterarm demnach dazu instruiert den Motor von P1 nach P2 zu bewegen. Dieses Ergebnis zeigt deutlich, dass eine Validierung selbst für einfache Aufgaben notwendig ist. Ohne eine solche Validierung hätte der Roboterarm im besten Fall keine Bewegung erzeugt, weil beispielsweise keine geeignete Greiferpose vorhanden ist. Im schlimmsten Fall hätte der Roboterarm versucht den Motor zu heben, was entweder zu einem Abrutschen des Greifers vom Motor oder einer Endlosschleife geendet hätte, da der Roboterarm nicht genügend Kraft aufbringen konnte. Beide Ergebnisse hätten jedenfalls eine

Tab. 6.1: Gewählte Reaktionstypen (RT1 bis RT6) der Nutzer im zweiten Teil der Studie für die Aufgaben: Markieren, Einfügen und Reinigen.

Aufgabe	**RT1**	**RT2**	**RT3**	**RT4**	**RT5**	**RT6**
Markieren	2	3	1	7	3	0
Einfügen	3	0	1	9	3	0
Reinigen	2	1	1	4	7	1

Unzufriedenheit des Nutzers zur Folge gehabt. Mit dem vorhergehend beschriebenen Ansatz ist solch eine Validierung möglich. In diesem Fall würde die Rückmeldung: "Ich kann den Motor nicht heben. Er ist zu schwer!" erzeugt werden. Das kann zwar auch zu einer Unzufriedenheit des Nutzers führen, vermeidet jedoch Schäden an Roboter oder Werkstück, welche vermutlich schwerer wiegen würden.

Am zweiten Teil der Studie haben insgesamt 16 Personen teilgenommen. Eine Übersicht der Ergebnisse ist in Tabelle 6.1 zu sehen. In der ersten Situation (Zeichnen des Musters) favorisierten 43% der Teilnehmer RT4 und dabei besonders die Benutzung des Bleistifts. Ein Teilnehmer schrieb sogar, dass er diese Rückmeldung gewählt hat, weil er damit seinem Mitarbeiter die Möglichkeit gibt zu intervenieren, falls ihm die Auflösung nicht zusagt. In der Einfüge-Aufgabe haben sich 56% der Nutzer für RT4 entschieden. Dabei haben diesmal sogar zwei Nutzer ihre Entscheidung damit begründet, dass damit dem Mitarbeiter die Möglichkeit gegeben wird die Bewegung abzubrechen, falls sie nicht so interpretiert wurde, wie er es sich gedacht hat. Die übrigen Nutzer haben zudem angegeben, dass ihre Entscheidung von den physikalischen Eigenschaften der Werkstücke bzw. Werkzeuge abhängig waren. In der letzten Situation (Reinigung des Arbeitsbereichs) hat sich die Mehrheit der Teilnehmer für den Rückmeldungstyp RT5 als Favorit entscheiden und RT4 am zweithäufigsten gewählt. Basierend auf den Meinungen der Teilnehmer war der Handfeger in dieser Situation am besten geeignet, oder zumindest genauso gut wie der Lappen. Unser Ansatz würde in dieser Situation ein ähnliches Ergebnis liefern.

Das Ergebnis der Nutzerstudie zeigt, dass der vorgestellte Ansatz zum einen notwendig ist, und zum anderen ein Verhalten simuliert, welches ein menschliches Verhalten gelungen imitiert. Eine Untersuchung der Formulierungsart der Rückmeldungen findet in Kapitel 8 statt.

6.3 Zusammenfassung

In diesem Kapitel wurde ein neuartiger Ansatz eingeführt, um Mehrdeutigkeiten basierend auf Affordanzen aufzulösen und damit auch Instruktionen validieren zu können. Bezogen auf die Problemstellung wird dadurch die Liste abbildbarer Instruktionen $\mathcal{I}_{KVE,FUZZY}$ in dem Sinne erweitert, dass auch Teilmengen der Instruktionen entweder auf Bewegungen, sprachliche Rückmeldungen oder einer Kombination aus beiden möglich sind. Damit eine solche Erweiterung überhaupt möglich ist, muss erneut die Menge an Kontextwissen $\mathcal{K}_{KVE}$ um die Affordanzen und Möglichkeiten diese zu bestimmen erweitert werden. Zusammenfassend kann man also sagen, dass sowohl die Flexibilität als auch die Robustheit des in dieser Arbeit beschriebenen

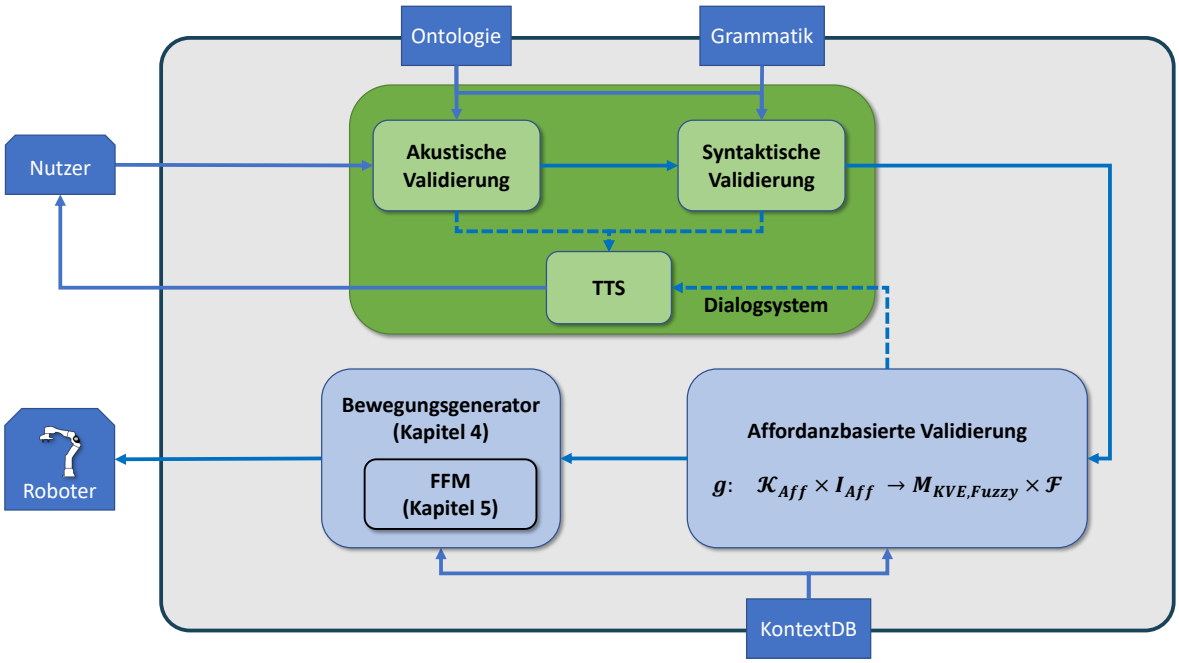


Abb. 6.6: Übersicht über das Gesamtkonzept nach der Einführung der affordanzbasierten Validierung von Instruktionen.

Gesamtkonzepts verbessert wurden. Der aktuelle Stand des Gesamtsystems ist in Abbildung 6.6 dargestellt.

Eine vielversprechende Erweiterung dieses Ansatzes ist die Umsetzung einer Komponente, welche Verben mit ähnlichen Affordanzen als Alternative vorschlägt, falls das aktuelle Verb nicht ausgeführt werden kann. Ein Beispiel dafür wäre, dass der Nutzer möchte, dass ein Objekt von Punkt A zu Punkt B gestellt wird. Das Objekt ist allerdings so schwer, dass der Roboterarm es nicht heben kann. In solchen Fällen könnte Beispielsweise eine Alternative wie “Ich könnte es aber schieben.” erstellt werden. Eine Umsetzung dieser Idee ist mit dem aktuellen Ansatz für alle Verben schon möglich. Eine gezielte Bereitstellung von ähnlichen Bewegungen würde jedoch ein effizienteres Arbeiten mit dem System ermöglichen.

Dieser Ansatz geht davon aus, dass Komponenten zur Verfügung stehen, welche dynamische Affordanzen zur Laufzeit auswerten können, und dass die mit einem Verb verknüpften Affordanzen eine erfolgreiche Bewegungserzeugung garantieren. Da selbst mit einem komplett validen Kommando das Resultat einer Bewegung von den Erwartungen der Nutzer abweichen kann, wird im nachfolgenden Kapitel ein Ansatz vorgestellt, welcher dieses Problem für kraftbasierte Bewegungen angeht.

Kapitel 7

Simulationsbasierte Validierung

Der letzte in dieser Arbeit vorgestellte Validierungsschritt besteht aus der Analyse bzw. Bewertung der instruierten Bewegung auf Basis einer physikbasierten Simulation und entspricht einer dynamischen Evaluierung von Affordanzen. Die Notwendigkeit solch einer Simulation ergibt sich aus dem fehlenden Bewusstsein seitens des Nutzers darüber, wie ein Roboterarm eine Bewegung ausführen wird (auf welcher Bahn er sich befindet) und wie die Umwelt auf diese Ausführung reagiert (ob ungewollte Kollision oder Instabilitäten auftreten).

Normalerweise werden physikbasierte Simulationen im Rahmen einer Optimierung von Bewegungsparametern genutzt, welche optimale Ablagepositionen oder auch Trajektorien zur Folge haben. Eine Eigenschaft solcher Ansätze ist jedoch, dass Ausnahmesituationen vermieden werden, was zu Problemen führen kann, wenn genau diese Situationen vom Nutzer gewollt sind. Solche Fälle können beispielsweise auftreten, wenn Objekte in der Umgebung durch eine Bewegung verschoben werden, es dem Nutzer aber nichts ausmacht oder der Nutzer gezielt einen nebenläufigen Effekt erzielen will, der im System nicht hinterlegt ist. Um solche Bewegungen trotzdem instruieren zu können, wird in diesem Kapitel ein Ansatz vorgestellt, welcher im Falle einer Abweichung von gewöhnlichen Resultaten dem Nutzer die Wahl überlässt, ob die Bewegung trotzdem ausgeführt werden soll, in dem eine Bestätigung durch den Nutzer angefordert wird. Die wissenschaftliche Fragestellung, die damit behandelt wird ist folgende:

F5 In wie weit kann eine instruierte kraftbasierte Bewegung online simuliert, analysiert und per Interaktion mit einem Nutzer validiert werden?

Durch solch eine simulationsbasierte Validierung sinkt das Risiko von Fehlausführungen, was eine Steigerung der Robustheit zur Folge hat. Dies spiegelt sich rein formal in einer Erweiterung der Rückgabemenge $\mathcal{F}$ wieder. Es wird nun zunächst ein Ansatz vorgestellt, welcher eine Simulation und Validierung von KVE erlaubt. Nach einer Evaluation eines Prototyps, welcher diesen Ansatz umsetzt, erfolgt eine Zusammenfassung und Diskussion der Ergebnisse. Der im Rahmen der simulationsbasierten Validierung umgesetzte Ansatz ist dabei in [Wölfel20b] veröffentlicht.

7.1 Ansatz

Der verfolgte Ansatz in dieser Arbeit orientiert sich in sofern an vorhergehenden Arbeiten, dass eine physikbasierte Simulation genutzt wird, um das dynamische Verhalten des Roboterarms und der Objekte im Arbeitsraum während der Ausführung abschätzen zu können. Die Abschätzung wird häufig genutzt, um entweder Bewegungsparameter zu optimieren, oder um Instruktionen generell zu validieren. Eine Optimierung kann dabei allerdings dazu führen, dass unnatürliche Resultate wegoptimiert werden, ohne dass der Nutzer darüber informiert wird. Dieser Ansatz löst diesen Grenzfall dadurch, dass bei unnatürlichen Resultaten eine Bestätigung des Nutzers gefordert wird, so dass eine Ausführung trotzdem möglich ist.

So könnte es beispielsweise sein, dass der Nutzer eine Menge an Objekten bearbeiten möchte und den Roboterarm nur instruiert, das mittlere Objekte zu ihm zu schieben, da dadurch die restlichen Objekte mit geschoben werden würden. Generell gilt es für das System allerdings als ungewollte Abweichung, wenn während des Schiebens eines Objekts andere Objekte im Raum ebenso bewegt werden. Wenn keine Informationen zu nachfolgenden Instruktionen oder Bewegungen bei der Planung einer Teilbewegung bekannt sind, kann das System im Normalfall keine ideale Lösung finden. Daher wird hier die Rückfrage als Lösung des Problems umgesetzt.

Außerdem liegt der Schwerpunkt hier auf der Bewertung von kraftbasierten Roboterbewegungen, was dazu führt, dass man für die Interpretation der generierten Bewegung eine ideale Ausführung nur schätzen kann, da die ausgeführten Bewegungen stark von der Umwelt abhängen (siehe Abbildung 7.1). Soll beispielsweise nur über den Tisch gewischt werden, entsteht eine lineare Bewegung, im Falle des Winkels zuerst eine schräge Bewegung nach oben, gefolgt von einer schrägen Bewegung nach unten. Bei einem Gehäuse erhöht sich die Komplexität der Bewegung

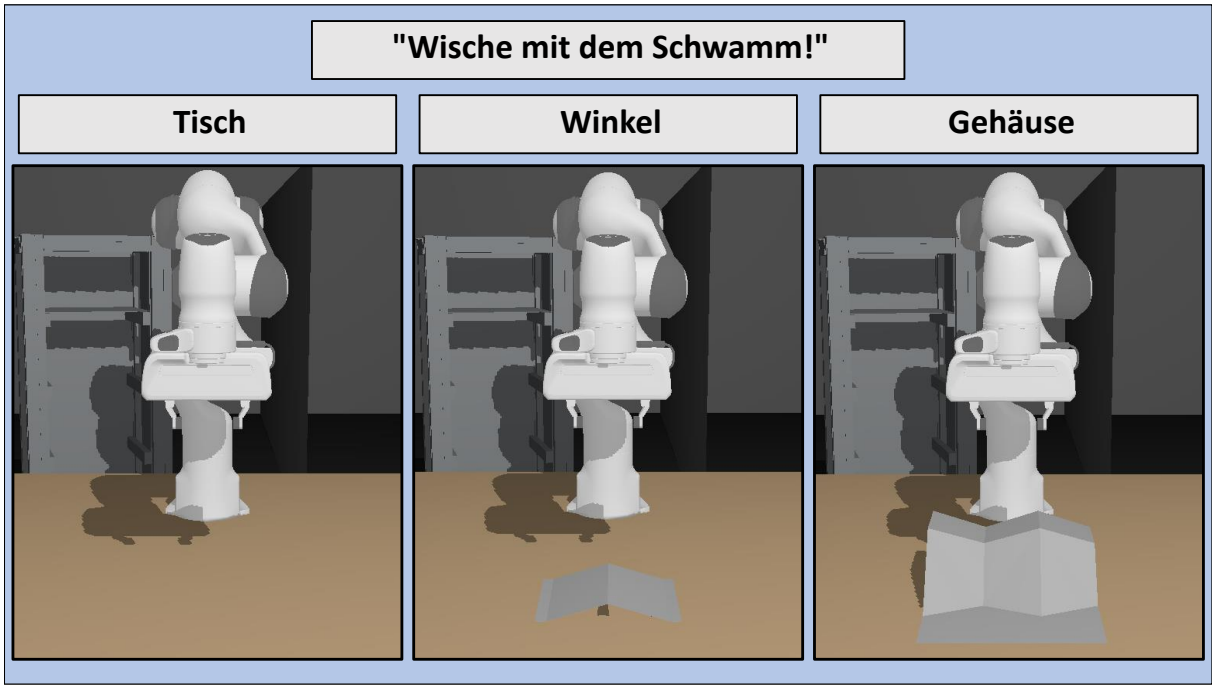


Abb. 7.1: Drei unterschiedliche Trajektorien für eine Bewegung bestehend aus Fahren nach rechts und Drücken nach unten.

erneut. Zugrunde liegt jedoch in jeder Situation die gleiche hybride Bewegung bestehend aus einem Drücken auf die Oberfläche und einer Bewegung von links nach rechts. Ein Abschätzen von möglichen Kollisionen mit anderen Objekten im Arbeitsraum wird dadurch noch aufwendiger als bei positions-geregelten Bewegungen, was unter Umständen zu Schäden am Roboter oder der Umgebung führen kann.

Eine Übersicht über den hier verfolgten Ansatz ist in Abbildung 7.2 zu sehen. Nachdem eine Instruktion wie in Kapitel 6 auf Eindeutigkeit geprüft und wie in Kapitel 5 beschrieben in ein KVE transformiert wurde, wird das KVE zunächst in einer Simulationskomponente ausgeführt. Dadurch können Abweichungen von einer erwarteten Ausführung erfasst werden ohne, dass ungewollte Schäden in der realen Szene entstehen. Dies geschieht darüber, dass pro Simulationsschritt der Weltzustand in einem Parameter-Protokoll notiert wird, welches an einen Interpreter weitergeleitet wird. Der Interpreter untersucht die Parameterwerte basierend auf Kontextwissen auf auffällige Abweichungen. Sind die Werte plausibel, wird das KVE an das Robotersystem zur Ausführung weitergegeben. Wird eine Auffälligkeit erkannt, so erfolgt die Weitergabe erst nach einer Bestätigung durch den Nutzer. Damit ist es auch möglich, Bewegungen ausführen zu lassen, welche nicht der Norm entsprechen. Dieser Fall tritt auf, wenn dem Nutzer Abweichungen in einer bestimmten Situation egal sind oder sogar gewollt sind.

Die einzelnen Schritte des Ansatzes werden nun im Detail erklärt. Dazu erfolgt zunächst die Definition der Objekteigenschaften, welche während der Simulation bestimmt werden und im Parameter-Protokoll erfasst werden können. Danach wird ein Konzept zur Definition von Erwartungswerten der Objekte und des Roboterarms für kraftbasierte Bewegungen eingeführt und die darauf basierende Validierung näher beschrieben. Abschließend erfolgt eine Erläuterung der Weiterverarbeitung der validierten Resultate.

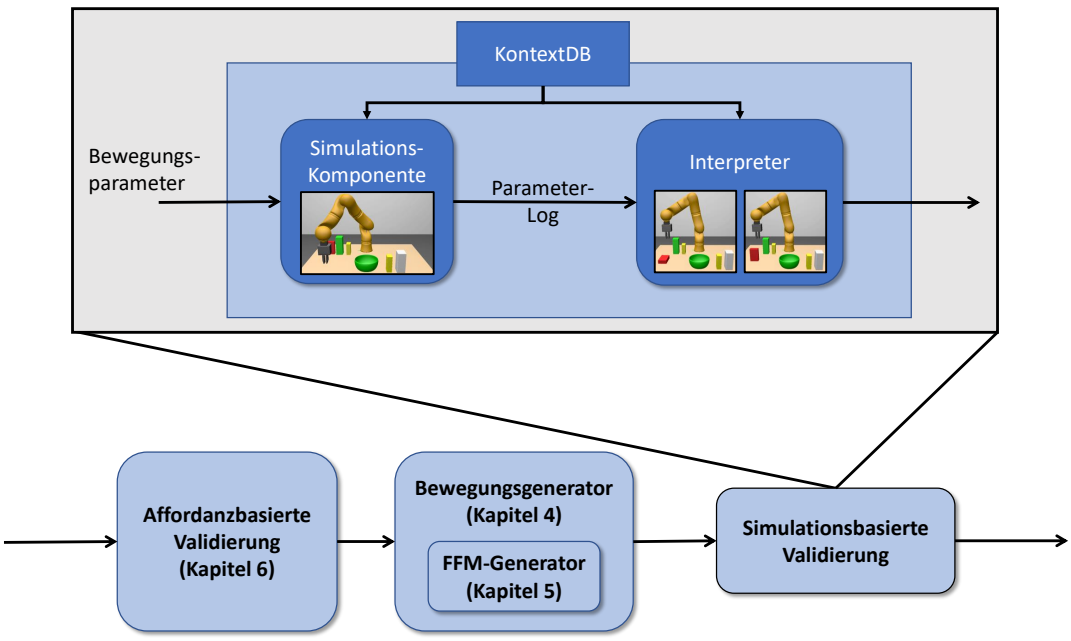


Abb. 7.2: Übersicht über die Komponenten der Anwendung. Über die Spiro Komponente werden Instruktionen in Roboterbewegungen überführt und mit Hilfe von Kontextwissen validiert.

7.1.1 Objektbeschreibung

Da bei kraftbasierten Bewegungen die Trajektorie für eine Bewegung nicht wie bei einer positionsgeregelten Bewegung über eine Menge aus Konfigurationen oder Posen beschrieben werden kann, werden gewünschte Resultate hier über Objektzustände definiert (siehe Abbildung 7.3). Der hier verfolgte Ansatz trennt dafür die im Arbeitsraum vorhandenen Objekte pro Bewegung in drei Gruppen: Werkstück, Roboterarm und Restobjekte.

Die erste Gruppe besteht im Rahmen dieser Arbeit lediglich aus einem Element, da hier Instruktionen von Bewegungen betrachtet werden, welche lediglich ein Objekt verändern. In der zweiten Gruppe ist der Robotergreifer bzw. das ausgerüstete Werkzeug enthalten. In die Gruppe der Restobjekte fallen die übrigen Objekte im Arbeitsraum. Die letzte Gruppe enthält eine weitere Unterscheidung in statische und dynamische Restobjekte. Statische Restobjekte sind dabei jene Objekte, welche sich sicher während der kompletten Bewegung nicht verändern werden. Meistens sind dies der Tisch oder der Boden, allerdings können dies auch fest montierte Bauteile sein. Dynamische Restobjekte sind dementsprechend Bauteile im Raum, welche ihren Zustand durch Kollisionen mit dem Werkstück, Greifer oder durch Nebeneffekte verändern können. Diese Unterteilung erfolgt dabei, damit während der Simulation unnötige Berechnung für statische Objekte ausgelassen werden können, die allerdings während der Simulation berücksichtigt werden müssen.

Der Zustand eines Objekts s ist dabei folgendermaßen definiert:

$$s = \{x, q, v, \omega, f, \tau\}, \tag{7.1}$$

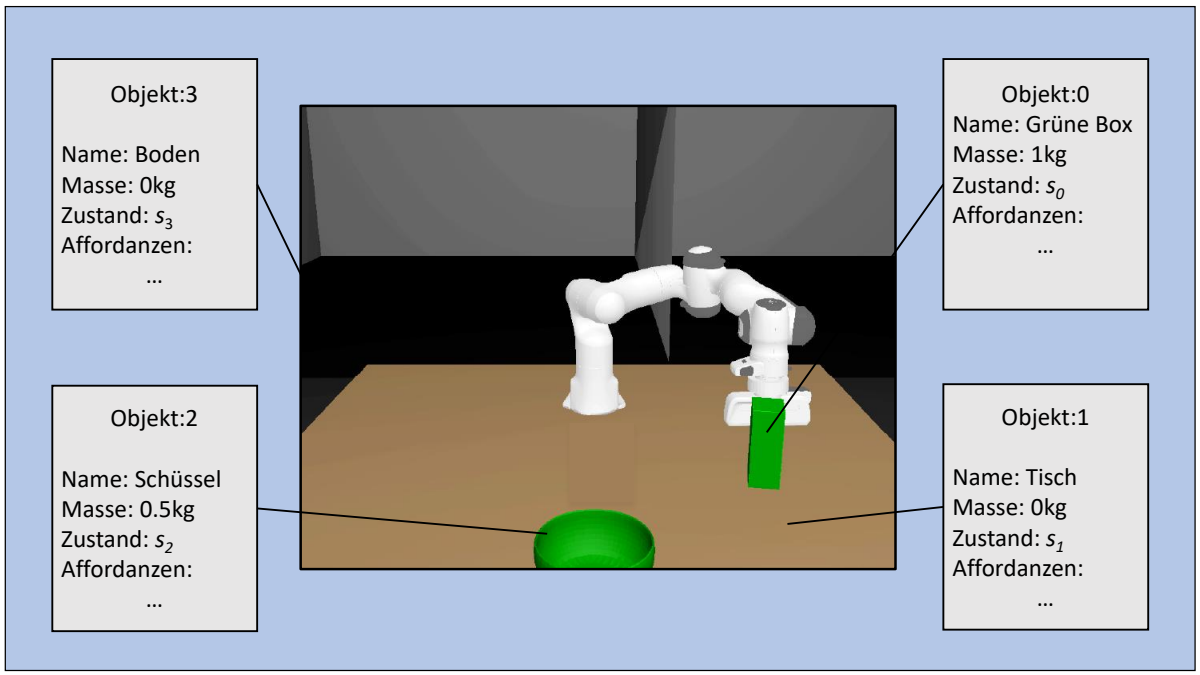


Abb. 7.3: Beispiel für eine digitale Repräsentation des Arbeitsraums. Werkstück: Grüne Box; Roboterarm: Greifer; dynamisches Restobjekt: Schüssel; Statische Restobjekte: Boden und Tisch.

wobei $x \in \mathbb{R}^3$ die Position des Objekts bezogen auf ein globales Koordinatensystem in der Roboterbasis beschreibt, $q \in \mathbb{H}$ die Orientierung im Raum darstellt, $v \in \mathbb{R}^3$ die lineare Geschwindigkeit, $\omega \in \mathbb{R}^3$ die Winkelgeschwindigkeit, $f \in \mathbb{R}^3$ Kräfte, welche von außen auf das Objekt wirken und $\tau \in \mathbb{R}^3$ Momente, die auf das Objekt wirken. Für jeden Zeitpunkt $t \in \mathbb{R}_0^+$ werden diese Objektzustände dabei in einem Weltzustand $w(t)$ zusammengeführt, welcher folgendermaßen definiert ist:

$$w(t) = \{s_0, \ldots, s_{n-1}, t\}, \tag{7.2}$$

wobei $n \in \mathbb{N}^+$ der Anzahl an Objekten im Arbeitsbereich entspricht. Neben der Gravitation, wird in dieser Arbeit davon ausgegangen, dass lediglich der Roboterarm für auftretende Kräfte und Momente im Arbeitsraum verantwortlich ist. Eine zusätzliche Interaktion durch den Nutzer oder simulierte Arbeiter werden also nicht berücksichtigt.

Neben der Bewertung einer Bewegung kann dieser Ansatz außerdem dazu verwendet werden, um die Sensorik eines Systems zu bewerten, indem die Erfolgsrate von Bewegungen für Rekonstruktionsgenauigkeiten simuliert wird. Ein Anwendungsfall dafür ist in Kapitel 7.2 dargestellt.

7.1.2 Parameters of Interest

Eine Bewertung einer kraftbasierten Bewegung ist, wie eingangs beschrieben, über eine Referenztrajektorie nicht immer möglich. Daher werden Effekte von Bewegungen hier nicht über Gelenkkonfigurationen, sondern über die eingeführten Objektbeschreibungen bzw. deren Änderung über die Zeit beschrieben und bewertet. Dies stellt eine Verknüpfung aus bestehenden Ansätzen dar, welche zum einen Bewegungen über Nebeneffekte im Raum validiert haben, und zum anderen den Zustand des Werkstücks während der Ausführung näher betrachtet haben.

Eine allgemeingültige Definition von validen Resultaten für eine Bewegung ist dabei nicht möglich, da diese stark von der Domäne und der subjektiven Einschätzung der Nutzer abhängen. Es muss also zunächst eine Untermenge der zu berücksichtigenden Parameter definiert werden. Zudem muss eine Möglichkeit gegeben werden, Resultate anwendungsspezifisch definieren zu können. Hier besteht diese Untermenge aus den in den Objektzuständen enthaltenen dynamischen Parametern, da diese von der Simulation bereitgestellt werden können. Eigenschaften wie eine Verformung und das Volumen werden demnach nicht berücksichtigt, da dieser Ansatz zunächst auf Starrkörper ausgelegt ist. Eine Erweiterung auf deformierbare Objekte wäre allerdings interessant für zukünftige Projekte. Neben den Objektbeschreibungen spielen für eine Bewegung die Typen des Werkstücks, Werkzeugs und der Rest-Objekte eine wichtige Rolle. Beim Schieben eines Balls ist es normalerweise egal, ob er sich dreht oder nicht, beim Schieben eines Klotzes spielt die Orientierung meistens jedoch eine wichtige Rolle, da der Nutzer vermutlich nicht möchte, dass er während der Bewegung umkippt. Je nach Werkzeug können unterschiedlich große Kräfte auftreten und eine erlaubte Veränderung von Restobjekten ist auch von deren Typ abhängig. So macht es dem Nutzer vermutlich wenig aus, wenn ein großes Paket etwas verschoben wird. Ein Kippen von einem gefüllten Wasserglas ist in der Regel jedoch vom Nutzer unerwünscht.

Um einen Anfang in der Berücksichtigung von Objekteigenschaften zur Validierung von Bewegungen zu machen, wird hier vorerst davon ausgegangen, dass für jedes Rest-Objekt die gleichen Grenzwerte erlaubt sind. Für die Spezifikation der Abweichungen werden im Folgenden sogenannte *Parameter of Interest Maps* (POI-Maps) eingeführt. Zu jeder Bewegung wird darin festgehalten, welche Parameterabweichungen für ein Werkstück, den Greifer und die dynamischen Restobjekte im Arbeitsraum während der Bewegung erlaubt sind. Für die Parametrierung der POI-Map wird in dieser Arbeit angenommen, dass diese wissensbasiert mit Bezug auf Objektparametern wie dem Typ oder dem Material arbeitet. Eine weitere Möglichkeit zur Generierung der POI-Maps wäre ein lernender Ansatz. Die Umsetzung solch eines Ansatz war jedoch nicht Schwerpunkt dieser Arbeit. Ein Beispiel für eine POI-Map ist in Abbildung 7.4 dargestellt.

Die Dreiteilung in Greifer, Werkstück und Restobjekte im Arbeitsraum ermöglicht die folgende Berücksichtigung von dynamischen Parametern. Dadurch, dass der Greifer separat betrachtet wird, kann erfasst werden, ob die geforderten Kräfte einen festgelegten Grenzwert einhalten, und ob der Greifer bzw. der Roboterarm aufgrund einer Kollision mit einem Hindernis bewegungsunfähig wird. Eine Überschreitung der Kraft würde im realen Fall zu einer Unterbrechung der Bewegung führen und eine Bewegungsunfähigkeit hätte ebenfalls eine Verzögerung der Bearbeitung des Bauteils zur Folge. In beiden Fällen würde dies zu zusätzlichen Kosten führen, was durch den vorgestellten Ansatz vermieden werden kann.

Das Werkstück wird getrennt betrachtet, da Grenzwerte für dieses Objekt im Normalfall von den Anforderungen an die Restobjekte abweichen. Außerdem wird darüber die Spezifikation bei Abweichungen vereinfacht. Für die Restobjekte können ebenfalls Grenzwerte für die dynamischen Parameter angegeben werden. Alternativ kann auch spezifiziert werden, dass entweder einzelne oder alle Parameter nicht berücksichtigt werden.

```
<POI>
        <Verb Typ="Schieben">
        <Werkstueck>
                <PositionThreshold x=".01" y=".01" z=".01" />
                <RotationThreshold x=".01" y=".01" z=".01" />
                < KraftThreshold x="Obj.Max.x" y=" Obj.Max.y " z=" Obj.Max.z " />
        <Werkstueck/>
        <Werkzeug>
                < PositionThreshold x=".01" y=".01" z=".01" />
                <RotationThreshold x=".0001" y=".0001" z=".0001" />
                < KraftThreshold x="Rob.Max.x" y=" Rob.Max.y " z=" Rob.Max.z " />
        <Werkzeug/>
        <RestObjekte>
                <PositionThreshold x=".001" y=".001" z=".001" … />
                <LinearGeschwindigkeitThreshold x=".001" y=".001" z=".001" … />
                <RotationThreshold x=".0001" y=".0001" z=".0001" />
                <WinkelGeschwindigkeitThreshold x=".001" y=".001" z=".001" … />
                <KraftThreshold x=".001" y=".001" z=".001" … />
        <RestObjekte/>
<POI/>
```

Abb. 7.4: Beispiel-POI Map für das Verb *Schieben*.

7.1.3 Simulation

Um die Objekt-Zustände während der Ausführung eines KVE zu erhalten, wird eine Festkörper-Simulation genutzt. Zu Beginn der Instruktion wird dafür eine virtuelle Kopie des Arbeitsraums über eine XML-basierte Konfigurationsdatei geladen. Diese kann dabei entweder manuell erstellt oder mit Hilfe entsprechender Rekonstruktionsalgorithmen und geeigneter Sensorik befüllt worden sein. Diese Konfiguration stellt den Start-Weltzustand $w(0)$ dar. Neben den Objektzuständen s_i sind zudem Informationen über das Material, dasd Gewicht und die Geometrie der einzelnen Objekte vorhanden, so dass die entsprechenden Festkörper generiert und der Simulation hinzugefügt werden können.

Für jede eingehende Instruktion erhält die Simulation die entsprechenden KVE, erstellt ein Parameter-Protokoll $\mathcal{L}_P$, aktualisiert die Regelung des simulierten Roboterarms und befüllt das Parameter-Protokoll nach jedem Simulationsschritt mit der Aktualisierungsrate $\delta t \in \mathbb{R}^+$. Die Alternative, Objektzustände nur zu sichern, falls eine ausreichend große Abweichung zu den entsprechenden POI-Paramtern auftritt, würde zwar unter Umständen zu einer Reduktion der Daten führen, erhöht allerdings die Komplexität der Weiterverarbeitung der Daten und kann je nach Grenzwert zu einem Informationsverlust führen.

Die Regelung des simulierten Roboters ist mit einer Momentregelung umgesetzt, um die Simulation möglichst nah an der realen Ausführung zu orientieren. Für Manipulationsprimitive wird zunächst überprüft, ob eine rein positionsgeregelte oder kraftbasierte Bewegung ausgeführt werden soll. Im ersten Fall werden geforderte Gelenkstellungen über eine Interpolation im Konfigurationsraum berechnet und als Zielkonfiguration zwischengespeichert. Im zweiten Fall wird eine tiefpassgefilterte Regelung umgesetzt, welche zunächst basierend auf den aktuellen Kraftwerten eine kartesische Zielposition berechnet und diese dann mittels einem Algorithmus zur Berechnung der Inversen Kinematik des Roboterarms in eine Zielkonfiguration umwandelt. Am Ende jedes Aktualisierungsschrittes wird dann überprüft, ob die Stopkriterien dieses Manipulationsprimitivs erfüllt sind. Die entsprechende Zielkonfiguration wird dann über einen *Inverse Dynamik*-Ansatz in jedem Simulationsschritt berechnet. Diese Konfiguration wird dann in entsprechende Gelenkmomente für den Roboterarm überführt, wobei zudem Kräfte und Momente am Roboterarm berücksichtigt werden.

An zwei Stellen kann es dabei zu einem Abbruch der Bewegung kommen, bevor das Manipulationsprimitiv komplett ausgeführt wurde. Zum einen kann es bei der Berechnung im Rahmen der kraftbasierten Bewegung dazu kommen, dass die Inverse Kinematik nicht lösbar ist. Zum anderen kann es passieren, dass der Roboterarm so mit einem Restobjekt kollidiert, dass er nicht mehr weiter fahren kann. In diesen Fällen wird die entsprechende Information über den Abbruch an den Nutzer weitergeleitet.

Kann eine exakte Rekonstruktion nicht garantiert werden, oder soll die Regelung für einen bestimmten Rekonstruktionsfehler evaluiert werden, kann das System ebenfalls genutzt werden. In diesem Fall wird die Simulation in mehreren Durchläufen mit variierenden Greiferposen ausgeführt, um die Rekonstruktionsfehler zu simulieren. Für jede Simulation wird dann der Weltzustand $w(t)$ in einem eigenen Parameter-Protokoll $\mathcal{L}_P$ gesichert.

7.1.4 Interpreter

Das Parameter-Protokoll $\mathcal{L}_P$ wird letztendlich an einen Interpreter weitergereicht, welcher dessen Einträge auf Basis der zugehörigen POI-Map auf Auffälligkeiten untersucht und unter Umständen eine Rückfrage an den Nutzer generiert. Für jeden Eintrag in $\mathcal{L}_P$ wird zunächst betrachtet, ob er als signifikant markiert ist. Ist dies der Fall und angegebene Parameter unter- bzw. überschreiten die Grenzwerte, so wird dieser Simulationsdurchlauf als auffällig gekennzeichnet und der betroffene Parameter wird gemeinsam mit dem Zeitstempel und der Objekt-ID in ein Resultat-Protokoll $\mathcal{L}_R$ eingetragen. Wurde keine Unauffälligkeit erkannt, so wird der KVE an das Robotersystem weitergeleitet.

Ist eine Sensorungenauigkeit mit angegeben, so berechnet man für eine Bewegung die Erfolgsquote $p_s \in [0, 1]$ als Verhältnis der erfolgreich ausgeführten Bewegungen zu der Anzahl der insgesamt ausgeführten Durchläufe. Im Falle $p_s = 1$ wurde kein Risiko identifiziert und der KVE wird an das Robotersystem weitergeleitet. Falls $p_{\min} < p_s < 1,\ p_{\min} \in [0, 1[$, werden die entsprechenden Abweichungen im Resultat-Protokoll gesichert und als potentiell auffällig markiert. Falls $p_s < p_{\min}$ werden die entsprechenden Abweichungen ebenfalls im Resultat-Protokoll gesichert und als sehr auffällig markiert.

Gilt nach der Interpretation $|\mathcal{L}_R| > 0$ so wird basierend auf den Abweichungen eine Rückmeldung an den Nutzer gesendet, welche zum einen die Unauffälligkeit beschreibt und den Nutzer zum anderen fragt, ob die Bewegung trotzdem ausgeführt werden soll. Gilt zudem $|\mathcal{L}_R| > 3$ so muss die Menge an Rückmeldungen reduziert werden, damit nicht wie in Kapitel 6 beschrieben, eine als störend empfundene Rückmeldung erzeugt wird. Daher muss die Information gegebenenfalls reduziert werden. Eine Reduktion kann dabei auf einige Arten erfolgen. Hier wurde die in Tabelle 7.1 dargestellte Reduktion umgesetzt.

Der triviale Fall ist gegeben, wenn nur eine Abweichung bei einem Objekt auftritt. In diesem Fall wird diese Abweichung verbalisiert. Eine Abweichung der Position wird dabei als Verschieben verbalisiert. Eine Abweichung entlang der Z-Achse kann unter Umständen als Herunterwerfen oder Herunterrutschen verbalisiert werden. Da die dafür notwendigen Informationen im Rahmen dieser Arbeit nicht vorliegen, wird auf ein Verschieben ausgewichen. Als Verdrehen wird eine Abweichung der Rotation um die Z-Achse beschrieben, während eine Abweichung um die X- oder Y-Achse als Umkippen verbalisiert wird.

Tritt eine Abweichung für eine Reihe an Objekten auf und ist die Anzahl der Objekte kleiner 3, so erfolgt zunächst eine Aufzählung der Objekte und danach die verbalisierte Abweichung. Ist die Anzahl größer 3, so wird lediglich kommuniziert, dass die Abweichung für eine Reihe

Tab. 7.1: Bestimmung des Inhalts der Rückmeldung im Falle von Abweichungen bei der Validierung von Instruktionen.

# Objekte	**Eine Abweichung**	**Mehrere Abweichungen**
1	Abweichung	Identifikation präferierter Abweichung
> 1	Sammlung der Objekte	Sammlung der Objekte und präferierte Abweichung

an Objekten gilt. Treten mehrere Abweichungen für ein Objekt auf, so wird eine präferierte Abweichung ausgewählt. Hier gilt beispielsweise, dass ein Umkippen die höchste Präferenz hat und somit lediglich diese Abweichung kommuniziert. In den Fall, dass für mehrere Objekte jeweils mehrere Abweichungen auftreten, erfolgt zunächst für jedes Objekt die Erfassung einer eindeutigen Abweichung und danach die Zusammenfassung der Objektnamen.

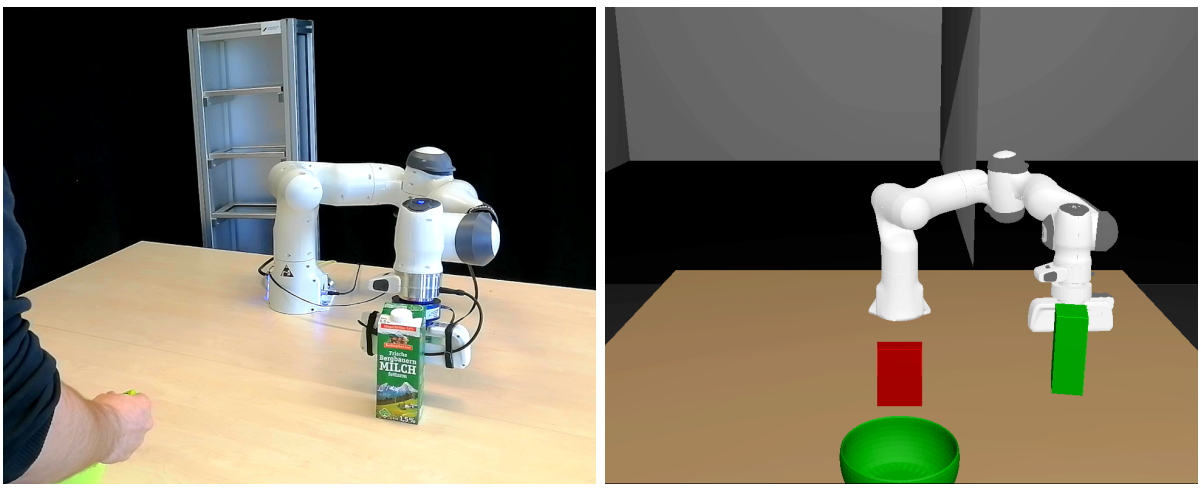
7.2 Prototyp-Evaluierung

Die Nützlichkeit dieses Ansatzes wird in folgendem Beispiel verdeutlicht: Dem Schieben von Objekten mit variierenden Rekonstruktionsgenauigkeiten (siehe Abbildung 7.5).

Das Szenario besteht darin, dass ein Nutzer plant einen Kuchen zu backen und gerade dabei ist die Zutaten zu verrühren. Im Fall 1 benötigt der Nutzer noch Kakaopulver, also instruiert er das System: “Kannst du mir den Kakao herschieben?”. Im zweiten Fall tritt das gleiche Ereignis mit einer Milchpackung auf. Im System ist hinterlegt, dass *Herschieben* ein Schieben-KVE in Richtung des Nutzers, also in positiver Y-Richtung, bedeutet. Dabei soll der Kontaktpunkt zwischen Greifer und Werkstück unter dem Schwerpunkt des Werkstücks liegen. Die restlichen KVE Parameter werden aus der Datenbank extrahiert und das KVE wird an die Simulation weitergereicht. Die Simulation wird $n = 9$ mal mit unterschiedlichen Offsets ausgeführt und eine minimale Erfolgsrate von $p_{min} = 80\%$ wird festgelegt.

Der genutzte Roboterarm ist ein *Franka Emika Panda*[1]. Die POI-Map für *Herschieben* wird aus der KontextDB geladen. Sie besteht aus einer Translation in Y-Richtung, begrenzt von einer Kraft, welche die Beschädigung des Objekts verhindert. Die erlaubte Abweichung der Orientierung der beiden Packungen wird auf einen Winkelbereich beschränkt. Dadurch kann ein Umkippen der Packungen erkannt werden. Die Restobjekte sind als nicht signifikant markiert, da sie sich außerhalb des betroffenen Bereichs befinden. Die Sensorgenauigkeit ist mit den Ab-

[1] https://www.franka.de/de/



(a) Reales Robotersystem (b) Simuliertes System.

Abb. 7.5: Betrachtetes Anwendungsszenario: Schieben von Objekten.

Tab. 7.2: Erfolgsrate R_{milk} und R_{cocoa} für die Rekonstruktionsgenauigkeiten ϵ.

ϵ [mm]	1	2	3	4	5	6	7	8	9	10
R_{milk} [%]	100	88	88	88	88	88	88	88	77	77
R_{cocoa} [%]	55	55	55	55	55	55	55	55	55	55

weichungen $\epsilon \in [1\ mm, \ldots, 10\ mm]$ in Y- und Z-Richtung modelliert worden. Das Szenario wurde auf einem System mit den folgenden Eigenschaften umgesetzt: Intel Core i7-6700HQ Prozessor und einer NVIDIA GeForce GTX 960M Grafik-Karte.

Um eine Aussage über die Echtzeitfähigkeit des Ansatzes zu gewinnen wurde die Zeit pro Simulationsschritt d_{step} und die gesamte Simulationsdauer gemessen. Insgesamt dauerte ein Simulationschritt, welcher den Zeitraum 1 ms simuliert 0.086 ms und eine komplette Simulation dauerte ca. 909 ms für eine Bewegung, welche 13.39 s in der realen Welt benötigt. Somit wird in diesem Fall eine Beschleunigung mit einem Faktor 14 erreicht, was für kurze Bewegungsabläufe als echtzeitfähig angesehen werden kann.

Die Erfolgsraten der beiden Fälle sind in Tabelle 7.2 dargestellt. Im Milch-Fall führt die Simulation für eine Genauigkeit von 1 mm zu einer erfolgreichen Ausführung in allen Durchläufen. Die Erfolgsrate sinkt um 12 % für 2 - 8 mm und fällt schließlich auf 77 % für 9 und 10 mm. In diesem Fall würde der Nutzer vom System informiert werden, dass die Milchpackung beim Verschieben vermutlich umfällt, falls eine Genauigkeit von 1 mm nicht erreicht werden kann. Im Kakao-Fall wird lediglich eine Erfolgsrate von 55 % für alle Genauigkeiten erreicht. Hier würde der Nutzer demnach benachrichtigt werden, dass die Kakaopackung sehr wahrscheinlich umfallen wird. Das liegt daran, dass der Roboterarm nicht weit genug unter den Schwerpunkt gelangt und somit ein Drehmoment an der Kakaopackung anliegt, welches zum Umfallen führt. Wenn es den Nutzer nicht stört, dass die Packung umfällt, weil es wichtiger ist, dass die Packung lediglich näher zum Nutzer geschoben wird, kann er die Bewegung trotzdem ausführen lassen und erspart sich die Packung selber zu holen.

Ein Vergleich mit der realen Ausführung ist außerdem in Abbildung 7.6 dargestellt. Die realen Resultate sind ähnlich zu den simulierten Bewegungen, was die Schlussfolgerung nahe legt, dass

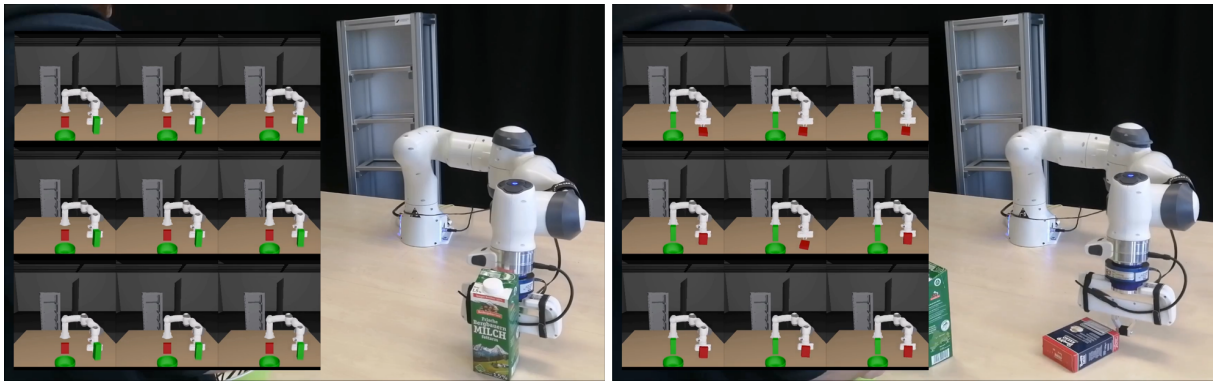


Abb. 7.6: Betrachtetes Anwendungsszenario: Schieben von Objekten für den Fall Milch (links) und Kakaopulver (rechts).

eine Simulation der Bewegungen valide ist.

7.3 Zusammenfassung

Der in diesem Kapitel eingeführte Ansatz erlaubt eine Simulation von kraftbasierten Roboterbewegungen, die Aufzeichnung von auftretenden Objektzuständen und die Interpretation der Ergebnisse. Die Interpretation erfolgt dabei auf Kontextwissen, welches über POI-Maps basierend auf einem Verb definiert wird. Dies entlastet den Nutzer in dem Sinne, dass die Bewegungsverläufe des Roboters nicht bekannt sein müssen, und selbst Instruktionen übergeben werden können, welche zu Kollisionen führen. Tritt eine Kollision oder eine andere Abweichung auf, welche von der Norm abweicht, wird eine Bestätigungsanfrage an den Nutzer zusammen mit Informationen zu der Abweichung generiert, so dass der Nutzer selbst entscheiden kann, ob eine Bewegung trotzdem ausgeführt werden soll.

Mit der damit einhergehenden Erweiterung der Menge an Rückmeldung $\mathcal{F}$ auf $\mathcal{F}_{Spiro}$ wird dadurch das Gesamtsystem (siehe Abbildung 7.7) robuster. Dies hat zur Folge, dass die Wahrscheinlichkeit sinkt, dass Nutzer durch Fehlverhalten das Vertrauen in den Roboter verlieren und stellt damit einen weiteren Schritt Richtung einer optimalen Roboterinstruktion dar. Der verbleibenden Schritte sind eine Evaluation des Gesamtsystem, sowie eine Untersuchung der Präferenz von Nutzern hinsichtlich der Formulierung der Rückmeldungen (roboterspezifisch, werkstückspezifisch oder hybrid), welche im folgenden Kapitel mittels eines Prototyps durchgeführt wird.

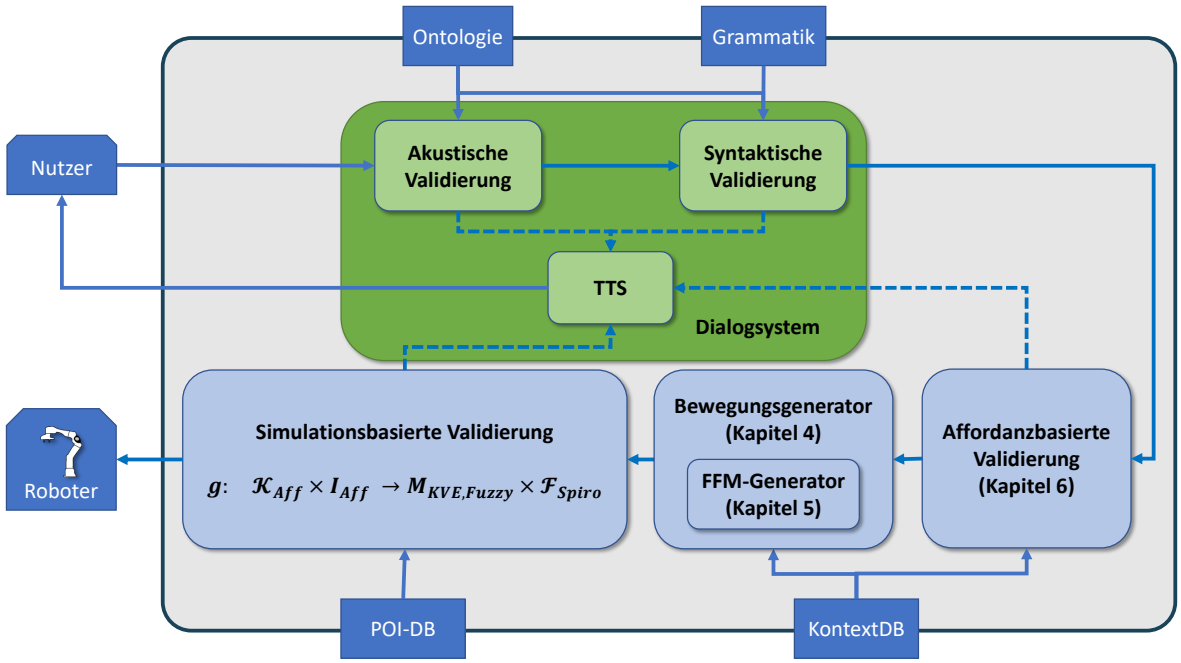


Abb. 7.7: Übersicht über das Gesamtkonzept nach der Einführung der simulationsbasierten Validierung von Instruktionen.

Kapitel 8

Evaluierung

In den vorhergehenden Kapiteln wurde eine Reihe an Fähigkeiten eingeführt und evaluiert, welche zum einen die Flexibilität und zum anderen die Robustheit der sprachbasierten Instruktion von Roboterbewegungen ermöglichen. In diesem Kapitel werden die Fähigkeiten in Form eines prototypischen Gesamtsystems zusammengefügt, um im Rahmen einer weiteren Nutzerstudie zu erfassen, in wie weit der in dieser Arbeit beschriebene Ansatz eine flexible und robuste Instruktion von kraftbasierten Roboterbewegungen erlaubt.

Dazu werden zunächst die Software- und Hardware-Komponenten des umgesetzten Prototyps näher beschrieben. Grundlegend geht es dabei um die Hardware-, sowie deren Ansteuerung, der Implementierung des Dialogsystems, welches die subsymbolischen Informationen der Nutzer erfasst, verarbeitet und an die Hauptanwendung weiterleitet, und die Hauptanwendung, welche die symbolische Information validiert und je nach Ergebnis an das Dialogsystem zurückgibt oder in Form von subsymbolischer Information an die Hardware-Ansteuerung weiterleitet. Zudem wird geklärt, wie die Weltrepräsentation umgesetzt wurde, und eine graphische Nutzerschnittstelle vorgestellt.

Die Nutzerstudie dient zur Evaluierung des Gesamtsystems in vier Teilschritten. Zunächst erfolgt eine Erfassung, in wie fern die vom System erzeugten Bewegungen den Erwartungen der Nutzer entsprechen. Danach wird die Zufriedenheit der Nutzer mit dem System über eine Bearbeitung von zwei Anwendungsfällen erfasst. Im dritten Teilschritt wird evaluiert, welche Form der Rückmeldung den Vorstellungen der Nutzer entspricht und im letzten Teilschritt wird erfasst, in wie weit eine zusätzliche visuelle Rückmeldung von Nutzern gewollt ist.

Abschließend werden die Ergebnisse der Nutzerstudie evaluiert und im Rahmen einer Zusammenfassung diskutiert.

8.1 Prototyp

In diesem Kapitel werden die einzelnen Komponenten des implementierten Prototyps näher beschrieben, welche sich aus der Hardware(-Ansteuerung), dem Dialogsystem und der Haupt-

Abb. 8.1: Die drei Hauptmodule des entwickelten Prototyps. Das Hardware-Modul ist für die Ansteuerung des Roboterarms zuständig, das Dialogsystem für die Interaktion mit dem Nutzer und die Hauptanwendung SpIRo für die Validierung und Transformation von Instruktionen in KVEs.

anwendung zusammensetzen (siehe Abbildung 8.1). Neben einer Beschreibung des verwendeten Roboterarms, wird in Kapitel 8.1.1 ein Überblick über die Steuerung des Roboters gegeben. Die Bestandteile des Dialogsystems werden in Kapitel 8.1.2 näher erklärt und die Implementierung der Hauptanwendung, welche sich aus der Weltrepräsentation, der Simulation der Bewegungen, der Validierung der Instruktionen und der graphischen Nutzerschnittstelle zusammensetzt, wird in Kapitel 8.1.3 vorgestellt.

8.1.1 Hardware

Die grundlegend verwendete Hardwarekomponente besteht aus einem Franka Emika Panda Roboterarm, welcher über eine interne Kraft-Moment-Sensorik verfügt, die eine Messung der am Roboterarm anliegenden Kräfte und Momente ermöglicht. Zudem ist eine Erweiterung des Systems um eine JR3-Kraftmessdose möglich, falls eine höhere Genauigkeit von Kraftmessungen erforderlich ist. Dabei muss jedoch berücksichtigt werden, dass dadurch auch die Bewegung des Roboterarms eingeschränkt wird, da die Kraftmessdose zwischen dem letzten Segment des Roboterarms und dem montierten Greifer befindet. Der Roboterarm ist dabei stationär und mit einem Backengreifer ausgerüstet, welcher ebenfalls eine Kraftrückmeldung zur Verfügung stellt.

Die Hardware-Ansteuerung ist in Form eines Transmission-Protocol-Servers (TCP-Servers) auf C++-Basis umgesetzt. Eingehende KVEs werden dabei mit unterschiedlichen Reglern, abhängig von den Bewegungstypen umgesetzt. Besteht ein Manipulationsprimitiv aus einer rein Positionsgeregelten Bewegung, so wird eine Freiraum-Bewegung angenommen und der Roboterarm im Gelenkraum mit einer hohen Geschwindigkeit verfahren. Besteht das Primitiv aus einem Fahren auf Kontakt, so wird eine von Franka Emika Panda bereitgestellte Funktion genutzt, um diese Bewegung umzusetzen. Ist letztlich eine kraftgeführte Bewegung definiert worden, wird ein für diese Arbeit in C++ implementierter kartesischer Regler verwendet, um diese Bewegung

auszuführen.

8.1.2 Dialogsystem

Das Dialogsystem wurde auf Basis der ontologiebasierten Dialogsystem-Plattform *Semvox*[1] [Pfalzgraf08] implementiert, da es eine flexible Generierung und Anpassung von Dialogsystemen erlaubt. Für die Spracheingabe und Sprachausgabe wurde *Nuance*[2] gewählt, da es schon erfolgreich in marktführenden Systemen integriert wurde. Um eine Reihe von Instruktionen abbilden zu können, wurde eine Ontologie definiert, welche Instruktionen basierend auf Teilkomponenten, wie Werkstücktypen, Verbtypen, dem geforderten Kraftaufwand und weiteren Spezifikationen erlaubt. Auf Basis dieser Eigenschaften wurde ein Dialogsystem definiert, welches Instruktionen erkennt, einzelne Bestandteile validiert, auf Mehrdeutigkeiten prüft, versucht, diese aufzulösen, und auf externe Events sprachbasiert reagiert. Zudem wurde eine TCP-Kommunikation implementiert, welche sowohl die Umwandlungen von Instruktionen in Spiro-Kommandos und deren Weiterleitung erlaubt, als auch eine Umwandlung von eingehenden Nachrichten auf Events umsetzt.

8.1.3 Hauptanwendung

Die Hauptanwendung ist auf C++-Basis implementiert und setzt eine Reihe an Aufgaben um: Die Kommunikation zwischen dem Dialogsystem und der Hardware-Ansteuerung, die Abbildung von unscharfen Parametern auf numerische Kraftwerte, die Erzeugung von KVE aus Spiro-Kommandos, die affordanzbasierte Validierung und die simulationsbasierte Validierung. Zudem enthält sie eine Komponente zur Aktualisierung des Weltzustandes und eine Visualisierung des Dialogs und des Weltzustandes in Form einer Nutzerschnittstelle.

Für die Erzeugung von KVEs wurde ein Interface implementiert, welches abhängig von Instruktionsparametern nach dem Factory-Pattern KVE Schablonen befüllt. Neben den Objektparametern eines Werkstücks werden dabei zur Laufzeit auch Kraftparameter über einen eigens implementierten Fuzzy-Reasoner erschlossen. Die benötigten Informationen über die Werkstücke und Werkzeuge erhält das System über eine XML-basierte Konfigurationsdatei, welche Objekteigenschaften wie die Masse, die Geometrie, das Material und Affordanz-Ausprägungen enthält.

Die simulationsbasierte Validierung wurde mittels der *Bullet Physics Engine*[3] implementiert. Objekte im Arbeitsraum können dabei momentan entweder als Quader oder Dreiecksmodelle geladen und simuliert werden. Über sogenannte Callback-Funktionen erhält man von der Simulation pro Zeitschritt dynamische Parameter, sowie Kräfte zwischen Kontaktklassen, welche für die einzelnen Objekte wiederum in der Konfigurationsdatei angegeben werden. Der Roboterarm ist über einen sogenannten *multi-body* implementiert und wird wie das reale Äquivalent über KVEs bzw. die daraus entstehenden Gelenkmomente angesteuert. So ist eine Simulation von kraftbasierten Bewegungen möglich und Objektparameter können in Simulationszeitschritten

[1] https://www.semvox.de/en/technologies/odp-s3-the-basis-for-interactive-systems/
[2] https://www.semvox.de/en/technologies/nuance-speech-recognition-tts/
[3] https://github.com/bulletphysics/bullet3

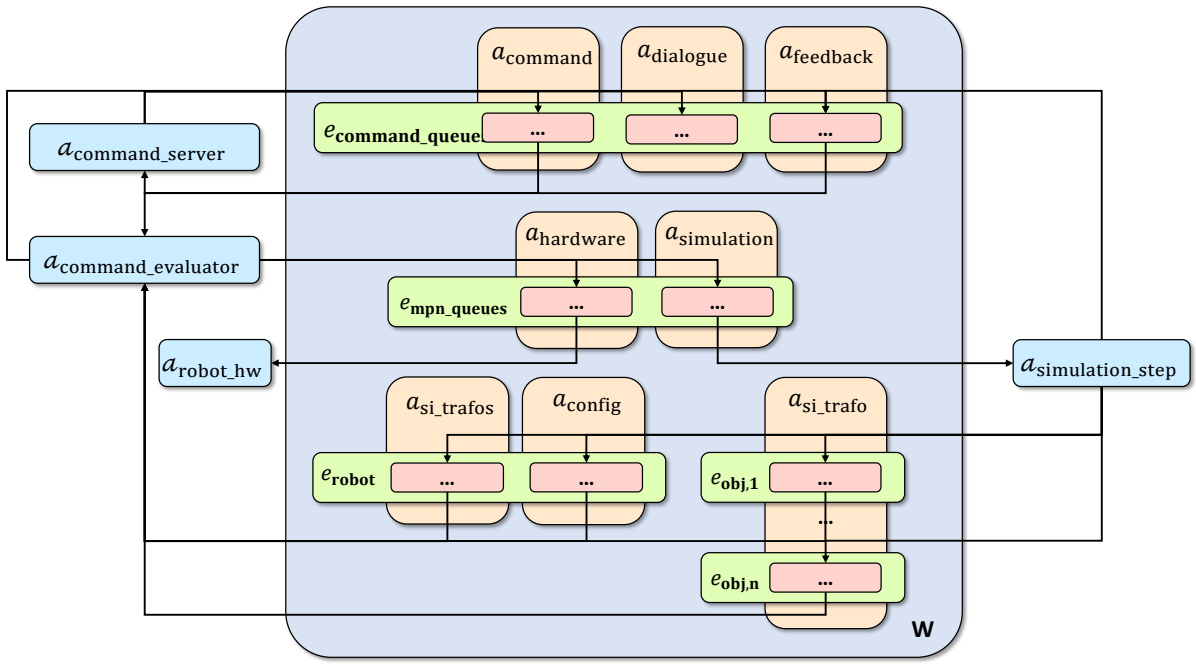


Abb. 8.2: Weltrepräsentation basierend auf dem ENACT-Framework.

von 1ms protokolliert werden.

Das Weltmodell ist über das Entity-Actor Framework ENACT implementiert worden (siehe Abbildung 8.2). Grundlegend wurden vier Aktoren umgesetzt: CommandServer, CommandEvaluator, RobotHardware und SimulationStep. CommandServer ist für die Kommunikation mit dem Dialogsystem zuständig und leitet sowohl den Dialogverlauf an die GUI weiter, als auch Instruktionen an den CommandEvaluator und sendet Information, wie Bestätigungsanfragen oder Benachrichtigungen an das Dialogsystem. RobotHardware kommuniziert wiederum mit der Hardwarekomponente in der Form von KVE. Der CommandEvaluator ist zum einen für die affordanzbasierte Validierung zuständig, und zum anderen für die Transformation von KVE aus Spiro-Kommandos. Der Aktor SimulationStep kümmert sich um die simulationsbasierte Validierung, indem eingehende KVE simuliert und basierend auf POI-Maps validiert werden, und leitet die KVE je nach Resultat weiter an die Hardwaresteuerung oder erzeugt eine Rückmeldung.

Um zum einen eine Überprüfung der simulierten Bewegungen zu ermöglichen, und zum anderen eine Übersicht des bisherigen Dialogs zu generieren, wurde zudem eine graphische Nutzerschnittstelle implementiert (siehe Abbildung 8.3). Zu Beginn des Programms wird dem Nutzer die Möglichkeit gegeben zwischen unterschiedlichen Konfigurationen des Roboterarms zu wählen (mit/ohne Kraftmessdose, kleine/große Greiferbacken) und eine Arbeitsraumkonfigurationsdatei zu wählen, welche den realen Arbeitsraum repräsentiert. Hier wurde diese Datei manuell erstellt. Steht eine geeignete Objektrekonstruktion zur Verfügung, kann die Datei allerdings auch darüber erstellt werden.

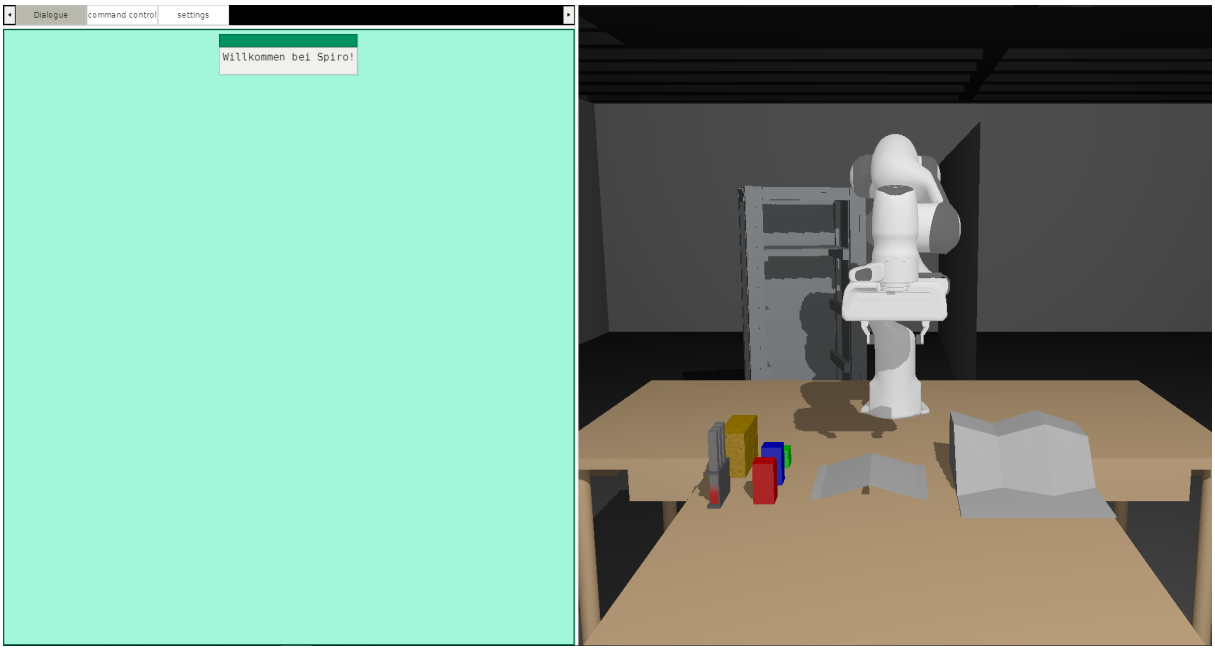


Abb. 8.3: Spiro GUI: Chatverlauf aufgeteilt in Roboter- und Nutzernachrichten (links) und Simulationsumgebung (rechts).

8.2 Nutzerevaluation

In diesem Kapitel wird der in dieser Arbeit vorgestellte Ansatz anhand des im vorherigen Kapitels beschriebenen Prototyps im Rahmen von vier Aufgaben evaluiert. Die erste Aufgabe dient dazu, die Erzeugung von KVEs und die Abbildung unscharfer Parameter auf Kraftwerte zu evaluieren. In der zweiten Aufgabe erhalten die Nutzer zwei Teilaufgaben, die sie mit dem System lösen sollen. Dies beweist die korrekte Funktionsweise des Systems und gibt einen Einblick in die Zufriedenheit der Nutzer mit dem System, da diese mittels einem Quesi-Bogen [Naumann10] erfasst wird. Aufgabe 3 dient dazu, herauszufinden, welche Art der Rückmeldung die Nutzer in Fehlerfällen bevorzugen und Aufgabe 4 sammelt Informationen darüber, in wie fern Nutzer eine zusätzliche visuelle Rückmeldung benötigen.

8.2.1 Set-Up

Insgesamt haben 16 Teilnehmer an der Nutzerevaluation teilgenommen. Das Durchschnittsalter betrug dabei 31 Jahre und die Teilnehmer setzten sich aus 2 weiblichen, 13 männlichen und einem diversen Teilnehmer zusammen, welche größtenteils einen Masterabschluss hatten. Auf einer Skala von 1 (wenig Erfahrung) von 7 (viel Erfahrung) ergab sich die in Abbildung 8.4 dargestellte Verteilung. Die per Sprachsteuerung bedienten Geräte setzen sich dabei hauptsächlich aus der Steuerung von Smartphones, Smartdevices und Autos zusammen.

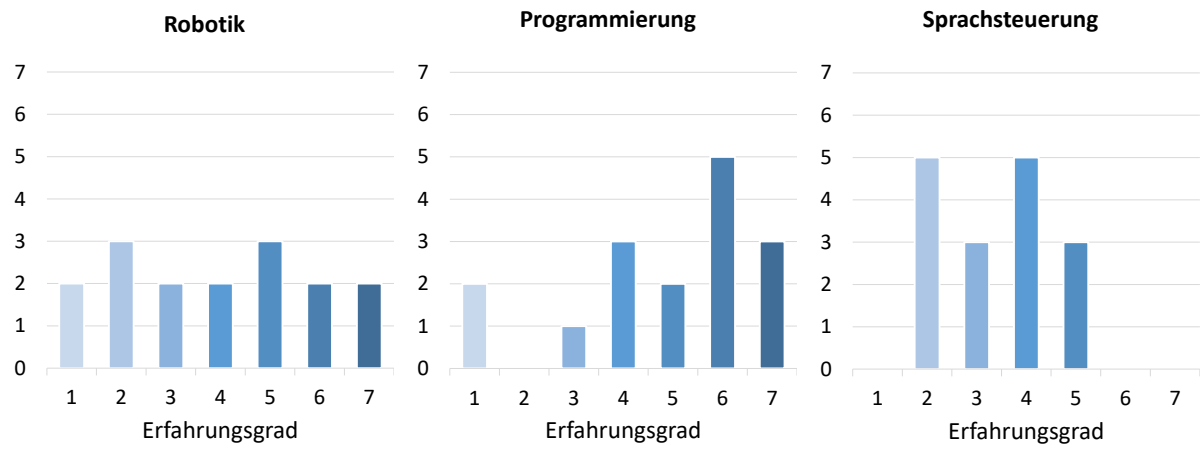


Abb. 8.4: Einschätzungen der Erfahrung der Nutzer in den Bereichen Programmieren, Robotik und Sprachsteuerung auf einer Skala von 1 sehr wenig bis 7 sehr viel.

8.2.2 Aufgabe 1: Benennung kraftbasierter Bewegungen

Die erste Aufgabe besteht darin, dass den Nutzern zehn Roboterbewegungen präsentiert werden und sie dazu aufgefordert sind, zum einen eine Instruktion zu nennen, welche ihrer Meinung nach zu dieser Bewegung führt, und zum anderen den Kraftaufwand in der Form *leicht, normal* und *stark* dazu anzugeben. Für die Instruktionen wurden ebenfalls Strukturen angegeben. Die Nutzer konnten aus folgenden Strukturen wählen:

- Verb und Spezifikator: mache leicht, mache stark
- Verb mit Spezifikator im Nebensatz: mache, und drücke dabei leicht auf
- Verb, das einer Bewegung mit entsprechender Kraft entspricht

Die gezeigten Bewegungen bestanden dabei aus den fünf folgenden unterschiedlichen Bewegungstypen: Zeichnen mit einem Bleistift, Radieren mit einem Radiergummi, Drücken mit einem Schraubenzieher, Wischen mit einem Schwamm und Streichen mit einem Pinsel, die jeweils zweimal mit unterschiedlichem Kraftaufwand durchgeführt wurden (wenig, normal und viel Kraftaufwand; siehe Abbildung 8.5). Es wurden dabei deshalb nur jeweils zwei Kraftaufwände gewählt, damit eine genauere Aussage darüber getroffen werden kann, in wie fern erzeugte Bewegungen den Erwartungen der Nutzer entsprechen.

Die Ergebnisse aus Aufgabe 1 sind in Tabelle 8.1 dargestellt. Bei 16 Teilnehmern gab es im Schnitt eine Übereinstimmung für einen Kraftaufwand von 75%. In 70 % der Bewegungen entsprechen die genannten Instruktionen auch eindeutig den erzeugten Bewegungen. Bei 10 % (Bewegung 2) waren sich die Nutzer uneinig, ob es normal oder viel Kraftaufwand war und bei den restlichen 20 % fiel es den Nutzern ebenfalls schwer, zu entscheiden, ob es sich um viel oder normalen Kraftaufwand gehandelt hat.

Grundlegend kann man also sagen, dass die mit diesem Prototypen umgesetzten Bewegungen im Rahmen dieser Nutzerevaluation den Erwartungen der Teilnehmer entsprechen. Die Unsicherheit

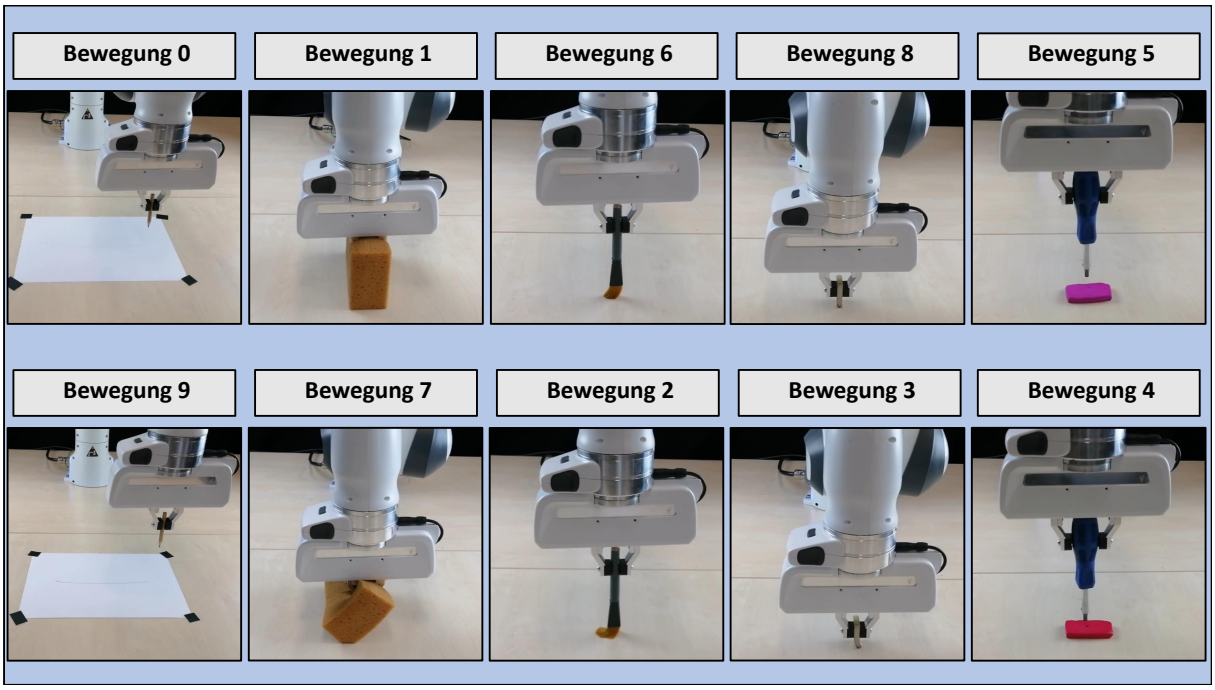


Abb. 8.5: Beispielbewegungen in der ersten Aufgabe, wobei die jeweils zueinander gehörenden Bewegungen übereinander dargestellt sind.

Tab. 8.1: Einschätzungen des Kraftaufwands in der ersten Aufgabe für die Bewegungen aus Abbildung 8.5

Bewegung	0	1	2	3	4	5	6	7	8	9
wenig	13	10	1	0	1	14	10	0	13	0
normal	3	6	7	11	13	2	6	2	3	15
viel	0	0	8	5	2	0	0	14	0	1
Intention	wenig	wenig	viel	normal	viel	wenig	wenig	viel	wenig	viel

bei Bewegung 2 lässt vermuten, dass Nutzer unterschiedliche Vorstellungen davon haben, ab wann eine Bewegung mit viel Kraftaufwand ausgeführt wird. Bei Bewegung 4 scheint es so, als würden Nutzer den Kraftaufwand bei solchen Bewegungen unterschätzen, da diese Bewegung schon mit der maximalen Kraft des Roboterarms (ca. 3N) ausgeführt wurde. Bei Bewegung 9 haben Nutzer während der Studie bereits gesagt, dass es bei einem Zeichnen mit einem Bleistift schwierig zu urteilen ist, ob normal oder mit viel Kraft aufgedrückt wurde.

8.2.3 Aufgabe 2: Interaktion mit dem Gesamtsystem

In der zweiten Aufgabe wurden die Nutzer dazu aufgefordert, direkt mit dem Prototypen zu interagieren. Der Grund für diese Aufgabe ist, zu erfassen, wie zufrieden die Nutzer mit dem System sind, und ob die in dieser Arbeit beschriebenen Fähigkeiten in Kombination eingesetzt werden können. Dafür wurde den Nutzern die in Abbildung 8.6 dargestellte Situation präsentiert und die folgenden zwei Teilaufgaben gestellt:

2.1 Das Bauteil Gehäuse wurde von einem Mitarbeiter falsch markiert. Es sollte nicht grün, sondern rot markiert sein. Bitte instruieren Sie das System so, dass die grüne Markierung entfernt und eine rote Markierung hinzugefügt wird

2.2 Es ist bald Feierabend und Sie sollten den Arbeitsplatz aufräumen lassen. Das Gehäuse kann stehen bleiben. Die drei bunten Klötze sollten allerdings in den blauen Bereich gestellt werden, damit ein weiterer mobiler Roboter sie in ein Regal räumen kann. Instruieren Sie das System so, dass sich die drei Klötze danach im Zielbereich befinden

Neben diesen Aufgaben wurde den Nutzern zudem mitgeteilt, dass sie das System fragen können, welche Verben, Werkstücke und Werkzeuge es bereits kennt. Außerdem wurden folgende Listen mit den dem System für diesen Anwendungsfall bekannten Wörtern zusätzlich unter der Aufgabenstellung angegeben, um Missverständnisse zu umgehen:

- **Verben:** Markieren, Reinigen, Wischen, Schieben und Stellen
- **Werkstücktypen:** Gehäuse und Klotz
- **Werkzeugtypen:** Stift und Schwamm

Diese Listen stellen dabei nur einen Bruchteil der bereits umgesetzten Verben dar. Diese Reduktion wurde bewusst durchgeführt, damit im Rahmen dieser Aufgabe der Schwerpunkt auf den Eigenschaften des im Prototypen umgesetzten Systems liegt. Dies entspricht der Abbildung von kraftbasierten Bewegungen, der Auflösung von Mehrdeutigkeiten über die affordanzbasierte Validierung und der simulationsbasierten Validierung. Es sollte also nicht untersucht werden, ob ein Nutzer möglichst ohne Vorwissen oder allein mit elementaren Bewegungen die gegebenen Aufgaben lösen kann.

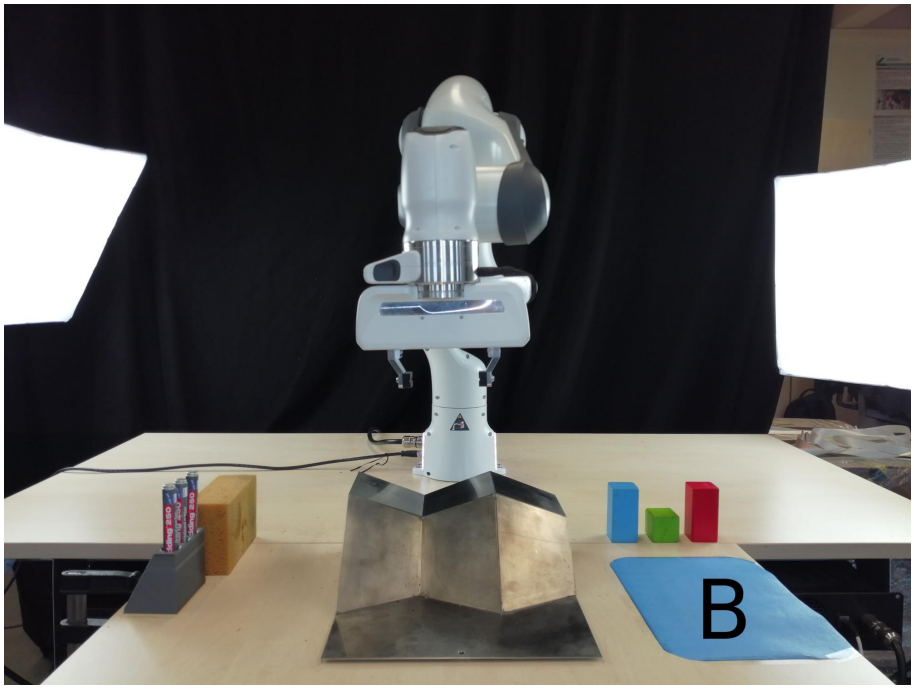


Abb. 8.6: Ausgangslage für die Lösung der beiden Teilaufgaben 2.1 und 2.2.

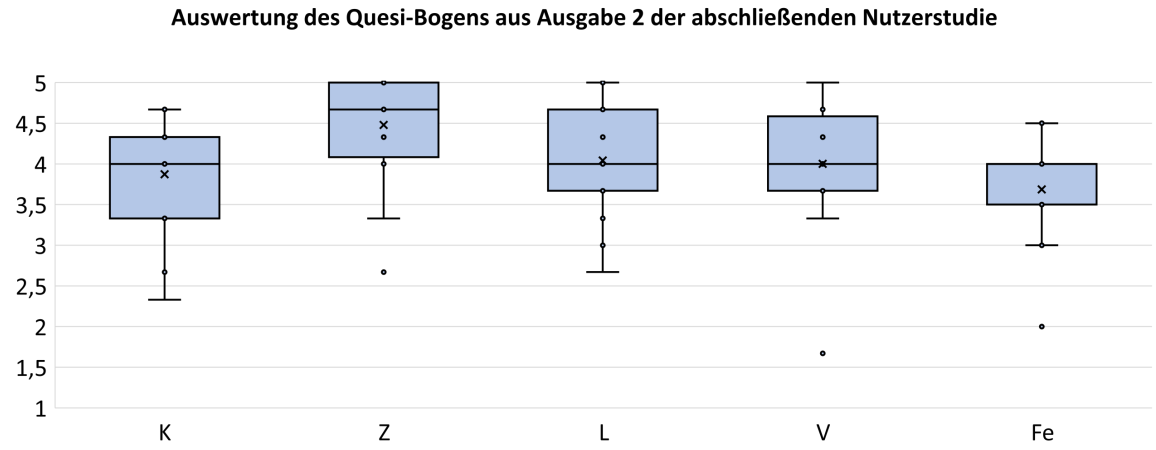


Abb. 8.7: Ergebnisse des Quesi-Fragebogens in den Bereichen: Kognitive Beanspruchung (K), Wahrgenommene Zielerreichung (Z), Wahrgenommener Lernaufwand (L), Vertrautheit bzw. Vorwissen (V) und wahrgenommene Fehlerrate (Fe).

Die Evaluation erfolgte dabei durch einen Quesi-Fragebogen zur Erfassung der Nutzerzufriedenheit mit dem System und einem SSEE-Fragebogen zur Erfassung des gefühlten Aufwands während der Bearbeitung der Aufgaben. Der Quesi-Fragebogen evaluiert dabei die kognitive Beanspruchung (K), die wahrgenommene Zielerreichung (Z), den wahrgenommenen Lernaufwand (L), die Vertrautheit bzw. das Vorwissen (V) und die wahrgenommene Fehlerrate (Fe). Insgesamt ist dabei eine Bewertung zwischen 1 (trifft gar nicht zu) und 5 (trifft vollkommen zu) möglich. Die Mittelwerte für die einzelnen Komponenten sind in Abbildung 8.7 dargestellt. Bei den SSEE-Fragebögen können Nutzer die Anstrengung während der Bearbeitung einer Aufgabe auf einer Skala von 0 bis 220 eintragen, wobei 0 für nicht anstrengend und 220 für über alle Maßen anstrengend steht.

Insgesamt wurde ein Quesi-Wert von 3,88 erreicht, was eine Zufriedenheit der Nutzer mit dem System repräsentiert. Der durch den SSEE-Fragebogen erfasste Aufwand für die Bearbeitung der Teilaufgaben 2.1 und 2.2 betrug 39,47 bzw. 53,15, was bedeutet, dass die Anstrengung im Mittel lediglich als *etwas anstrengend* empfunden wurde. Dies zeigt, dass die Nutzer selbst ohne umfassende Einführung intuitiv mit dem System arbeiten konnten.

Den Nutzern wurde außerdem die Möglichkeit gegeben, Anmerkungen für eine Verbesserung des Systems zu machen. Vorgeschlagene Erweiterung waren dabei: Eine Visualisierung des Zustands des Dialogsystem, also ob es gerade zuhört oder nicht. Dies könnte in Form einer Lampe oder einem Bereich in der Nutzeroberfläche umgesetzt werden. Zudem wurde vorgeschlagen, dass Instruktionen nicht erneut wiederholt werden sollten, wenn das System eine Alternative vorschlägt, sondern auch hier nur eine Bestätigung ausreichen sollte. Solch eine Fähigkeit kann problemlos im gleichen Stil wie die bisherigen Bestätigungen umgesetzt werden. Neben einer gewünschten verbalen Verknüpfung von Bewegungen wurde zudem eine Integration einer Objektrekonstruktion vorgeschlagen, welche vor jeder Simulation die simulierte Szene mit der Realität vergleicht und falls nötig anpasst, um Ungenauigkeiten in der Simulation auszubessern.

8.2.4 Aufgabe 3: Präferenz hinsichtlich der Rückmeldung

Die dritte Aufgabe dient dazu, Informationen darüber zu erhalten, in welcher Form und in welchem Detailgrad Nutzer auf problematische Ausführungen hingewiesen werden wollen. Die untersuchten Formen sind dabei: *abstrakt*, *roboterbezogen*, *werkstückbezogen* und *umgebungsbezogen*. Dazu wurden den Nutzern insgesamt fünf Anwendungsszenarien in Form eines Bildes zur Szenenbeschreibung, einer Instruktion des Nutzers an das System und einer Reihe an Antwortmöglichkeiten des Systems gegeben. Aus diesen Möglichkeiten sollten sie sich ihren Favoriten aussuchen und ihre Antwort begründen.

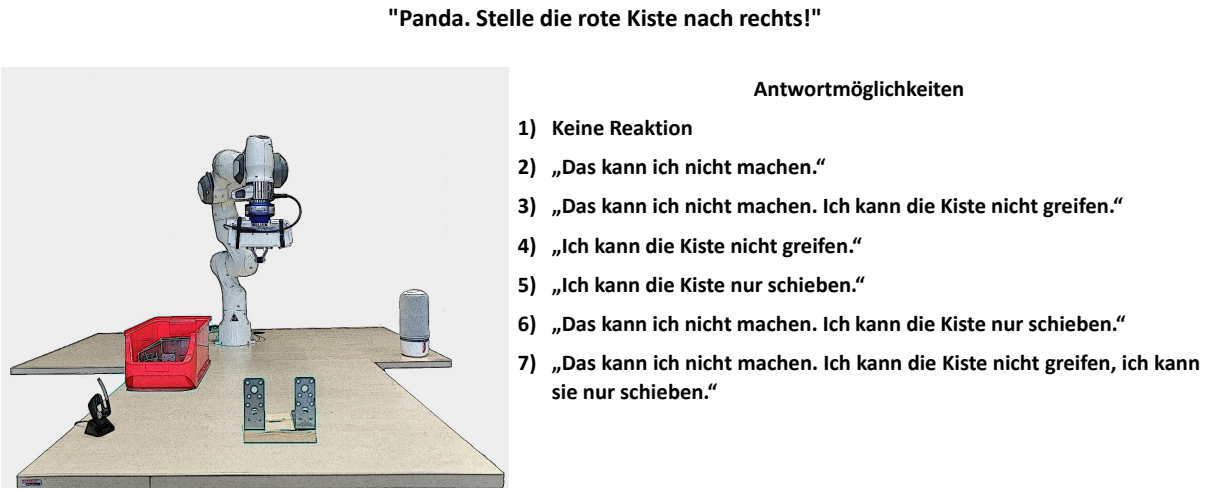


Abb. 8.8: Szenario 1. Beispiel für eine nicht erfüllte Affordanz: *hebbar*.

Im ersten Szenario (siehe Abbildung 8.8) ist der Anwendungsfall dargestellt, dass ein Werkstück fertig gestellt wurde und nun die rote Kiste weggeräumt werden soll. Die angegebene Instruktion ist folgende: “Panda. Stell die rote Kiste nach rechts!” Da die Kiste zu schwer ist, die Affordanz *hebbar* also nicht erfüllt ist, kann der Roboterarm sie nicht heben, sondern höchstens schieben. Daher werden die in Abbildung 8.8 angezeigten Rückmeldungen vorgeschlagen.

Hier lag bei den Nutzern der Schwerpunkt beim Rückgabetyp 5: “Ich kann die Kiste nur schieben” (siehe Abbildung 8.13). Als Begründung wurde in den meisten Fällen genannt, dass in dieser Rückmeldung bereits alle benötigten Informationen implizit enthalten sind, welche in den restlichen Antwortmöglichkeiten explizit ausgedrückt werden. Indem erklärt wurde, dass die Kiste nur geschoben werden kann, war klar, dass das geforderte Greifen der Kiste nicht ausgeführt werden kann. In den Fällen, in denen Rückgabetyp 7 gewählt wurde, wurde als Begründung genannt, dass es gerade zu Beginn der Arbeit mit solch einem System hilfreich sein könnte, wenn man in Fehlerfällen möglichst umfangreich über Probleme informiert wird.

Im zweiten Szenario (siehe Abbildung 8.9, links) ist als Beispiel gegeben, dass zusammen mit dem Roboterarm eine Reihe an Holzklötzen bearbeitet werden soll. Als nächstes soll der blaue Klotz bearbeitet werden, also wird die folgende Instruktion übergeben: “Panda. Gibt mir bitte den blauen Klotz!” Auch hier ist eine Affordanz, die Erreichbarkeit des blauen Klotzes, nicht erfüllt, da es ansonsten zu Kollisionen mit anderen Klötzen kommen würde. Es werden roboterzentrierte,

Abb. 8.9: Szenario 2. Beispiel für eine objektbedingt nicht erfüllte Affordanz: *erreichbar*.

objektzentrierte, abstrakte und hybride Rückmeldungen vorgeschlagen (siehe Abbildung 8.9).

Diesmal lag bei den Nutzern der Schwerpunkt beim Rückgabetyp 6: “Das kann ich nicht machen. Es stehen andere Klötze im Weg” (siehe Abbildung 8.13). Dies wurde damit begründet, dass diese Antwortmöglichkeit die Informationen, wie die Unerreichbarkeit des Werkstücks, Nutzern bereits implizit zur Verfügung stellt. In den Fällen, in denen Rückgabetyp 7 gewählt wurde, wurde erneut als Begründung genannt, dass es gerade zu Beginn der Arbeit mit solch einem System hilfreich sein könnte, wenn man in Fehlerfällen möglichst umfangreich über Probleme informiert wird.

Abb. 8.10: Szenario 3. Beispiel für eine roboterbedingt nicht erfüllte Affordanz: *erreichbar*.

Im dritten Szenario (siehe Abbildung 8.10, links) ist die Anwendung dargestellt, dass der Roboterarm dem Nutzer ein Werkstück zur Montage anreichen soll. Daher ist die Instruktion gegeben: “Panda. Bitte gib mir den letzten Winkel!”. Auch hier ist die Affordanz *erreichbar* nicht erfüllt. Diesmal ist der Grund dafür jedoch die eingeschränkte Reichweite des Roboterarms und nicht eine Kollision.

Die Resultate für dieses Szenario (siehe Abbildung 8.13) zeigen, dass hier die Rückgabetypen 4 (“Das ist zu weit weg.”) und 5 (“Mein Arm ist zu kurz.”) präferiert wurden. Aus den Erläuterungen zu diesen Antworten lässt sich zudem schließen, dass der Großteil der Nutzer beide Antwortmöglichkeiten gleichermaßen als valide ansehen. 3 Teilnehmer haben zudem geschrieben, dass Rückgabetyp 4 menschlicher und damit geeigneter wirkt.

Abb. 8.11: Szenario 4. Beispiel für eine werkstückbedingt nicht erfolgreiche Ausführung.

Im vierten Szenario (siehe Abbildung 8.11, links) soll der Nutzer die beiden Bauteile im Vordergrund miteinander verschrauben und benötigt dafür die Schraubenkiste. Daher erfolgt die Instruktion: “Panda. Kannst du mir die Schraubenkiste herschieben?” Dies ist ein Beispiel für die simulationsbasierte Validierung, da während des Schiebens in der Simulation erkannt wird, dass die Schraubenkiste dabei umfallen würde. Also wird vom System die in Abbildung 8.11, rechts gezeigten Rückmeldungen erzeugt.

Die eindeutige Präferenz der Nutzer in Szenario 4 (siehe Abbildung 8.13) liegt bei Rückgabetyp 6: “Ich würde dabei die Kiste wahrscheinlich umwerfen. Soll ich es trotzdem tun?” Die Nutzer nannten als Begründung, dass diese Antwort zum einen genug Informationen über das auftretende Problem bereitstellt, und dass die Übertragung der Verantwortung an den Nutzer eine gute Idee ist.

Das letzte Szenario (siehe Abbildung 8.12, links) entspricht dem vierten Szenario mit dem Unterschied, dass diesmal zwei weitere Kartons beim Schieben der Schraubenkiste in der Schieberichtung der Schraubenkiste liegen. Das verhindert zwar, dass die Schraubenkiste umfällt, führt jedoch dazu, dass die beiden Kartons verschoben werden. Auch dies wird über die simulationsbasierte Validierung erfasst. Der Unterschied zu Szenario 4 ist jedoch, dass die Abweichung diesmal auf der Seite der anderen Werkstücke liegt.

Für dieses Szenario 5 ist das Resultat ebenfalls eindeutig (siehe Abbildung 8.13). Diesmal präferieren die Nutzer jedoch eine andere Antwortmöglichkeit: Den Rückgabetyp 5: “Dabei würde ich mit der anderen Kiste zusammenstoßen. Soll ich es trotzdem tun?”. Als Begründung wurde genannt, dass dieser Rückmeldungstyp möglichst knapp alle notwendigen Informationen bereitstellt. Auch hier wurde als Grund für die Wahl zudem genannt, dass für Nutzer die

Abb. 8.12: Szenario 5. Beispiel für eine objektbedingt nicht erfolgreiche Ausführung.

Möglichkeit besteht zu entscheiden, ob eine Bewegung ausgeführt werden soll.

Zusammenfassend kann geschlussfolgert werden, dass Nutzer keine Präferenz haben, was den Bezug der Rückmeldung angeht. Das heißt, Rückgabetypen, die roboterbezogen oder objektbezogen sind, werden gleichermaßen als valide angenommen, solange dem Nutzer alle relevanten Informationen zumindest implizit zur Verfügung gestellt werden. Eine explizite Angabe aller Informationen wird als hilfreich angesehen, wenn der Nutzer noch nicht mit dem System vertraut ist. Sobald die Bewegungen des Systems besser eingeschätzt werden können, werden jedoch möglichst kurze Rückmeldungen präferiert. Eine Rückfrage des Systems, ob eine Bewegung im Falle einer Abweichung im Sinne von Kapitel 7 trotzdem ausgeführt werden soll, wurde in den meisten Fällen positiv angenommen.

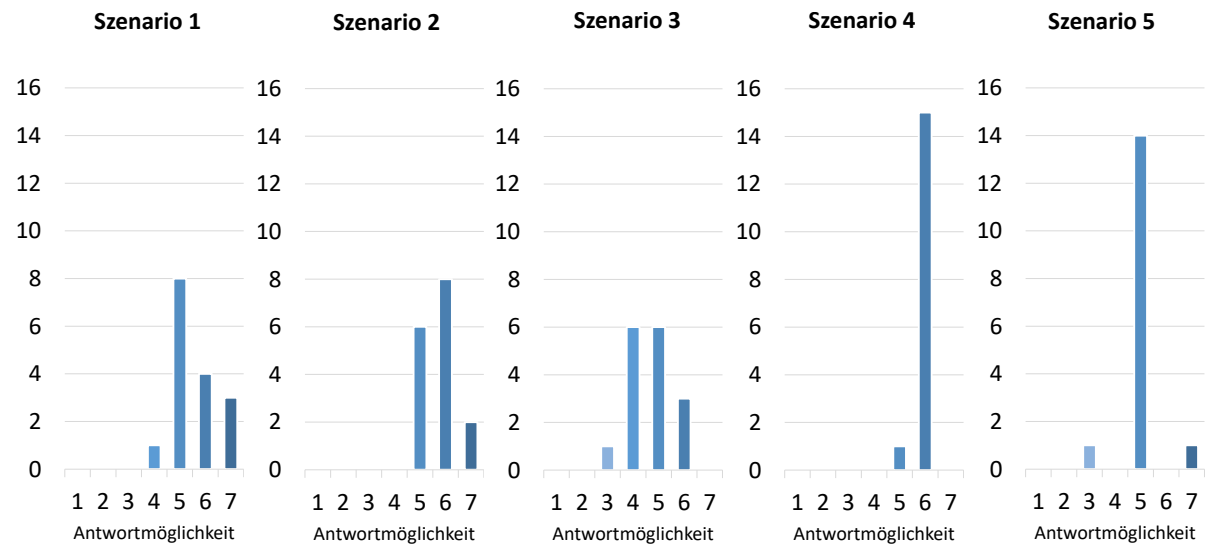


Abb. 8.13: Anzahl der gewählten Rückgabetypen in Szenario 1 bis Szenario 5 von Aufgabe 3 der abschließenden Nutzerstudie.

8.2.5 Aufgabe 4: Visuelle Rückmeldung

Die abschließende Aufgabe der Nutzerstudie soll Informationen darüber sammeln, in wie weit Nutzer neben der sprachlichen Rückmeldung ein zusätzliche visuelle Rückmeldung befürworten würden. Dazu wird den Nutzern die in Abbildung 8.14 dargestellte Nutzeroberfläche präsentiert, welche sowohl den Dialogverlauf, als auch den Arbeitsraum in digitaler Form darstellt. Die Meinung der Nutzer zu einer solchen visuellen Rückmeldung wird dann über folgende Fragen erfasst:

1. Finden Sie die Anzeige des Chatverlaufs sinnvoll oder nicht? Bitte begründen Sie Ihre Antwort.
2. Fänden Sie es gut oder schlecht, wenn Sie die Roboterbewegung in der Simulation angezeigt bekommen bevor der reale Roboterarm sie ausführt?
3. Falls Sie 2. Mit „Ja“ beantwortet haben: Wäre es Ihnen lieber, wenn die komplette Bewegung gezeigt wird oder würde Ihnen der Endzustand genügen?
4. Fänden Sie es besser wenn die Simulation in Echtzeit oder wenn die Simulation beschleunigt dargestellt wird?
5. Fänden Sie es gut oder schlecht, wenn im Fehlerfall in der Simulation angezeigt wird, wann und wo der Fehler auftritt?

Die Resultate für die erste und zweite Frage der vierten Aufgabe sind in Tabelle 8.2 dargestellt. Die Hälfte (52 %) der Nutzer fänden die Anzeige eines Chatverlaufs gut, 25 % sehen ihn nur

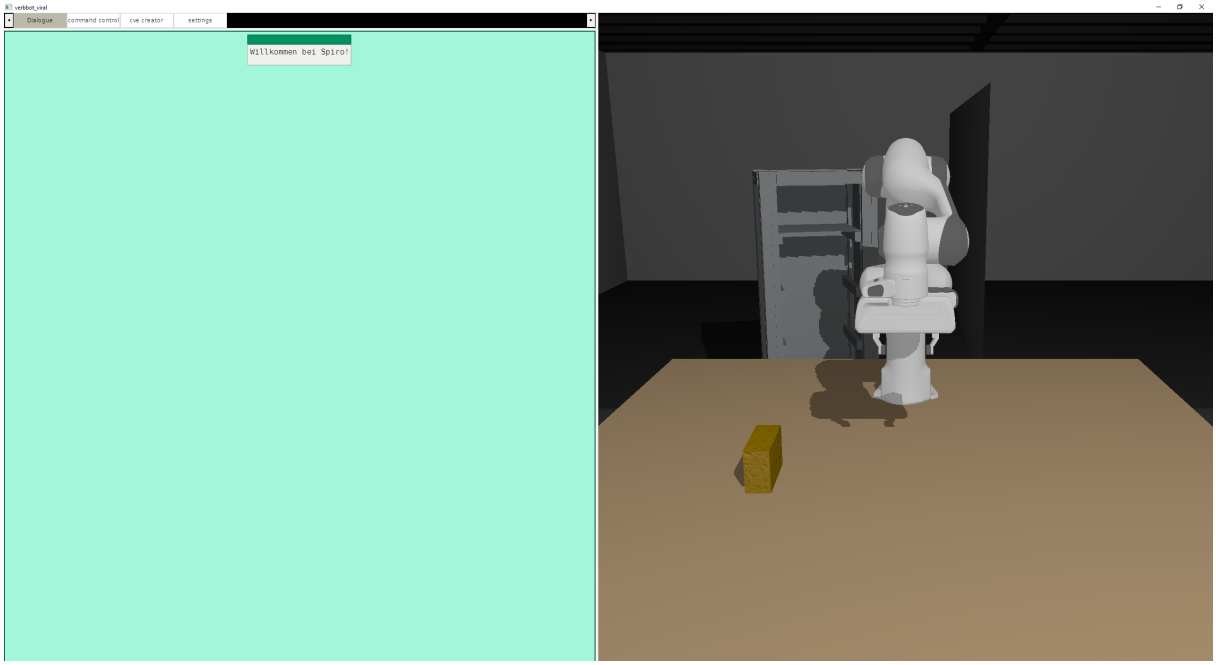


Abb. 8.14: Beispielbild der entwickelten graphischen Nutzeroberfläche für Aufgabe 4 der Nutzerevaluation. Links: Chatverlauf zwischen dem Nutzer und dem Dialogsystem. Rechts: Simulationsfenster mit einer digitalen Repräsentation des Arbeitsraums.

Tab. 8.2: Aufgabe 4: Fragen 1 und 2. Anzahl der Nennungen für *gut, schlecht* und *geteilt*, für den Fall, dass sich ein Nutzer nicht entscheiden konnte.

Frage	**gut**	**schlecht**	**geteilt**
1	8	4	4
2	6	3	7

als sinnvoll an, falls Missverständnisse auftreten können und 25 % würden komplett auf ihn verzichten. Bezogen auf die Notwendigkeit einer Visualisierung der Bewegung sind je 37,5 % dafür, dass sie hilfreich wäre und 43,75 % sind erneut geteilter Meinung, was von den Nutzern dadurch begründet wird, dass sie eine Visualisierung nur bei noch unbekannten Bewegungen oder im Fehlerfall präferieren würden. Falls eine Visualisierung gewollt ist, würden 75 % der Nutzer die komplette Bewegung vorziehen, 16,7 % nur einen Teilausschnitt und 8,3 % lediglich den Endzustand bevorzugen. Auf die vierte Frage wurde größtenteils (62,5 %) geantwortet, dass die Bewegung beschleunigt dargestellt werden sollte, 18,75 % würden nur irrelevante Teilabschnitte beschleunigen und 18,75 % bevorzugen eine Simulation in Echtzeit. Die letzte Frage wurde durchweg mit gut beantwortet, was zeigt, dass neben der sprachlichen Rückmeldung im Fehlerfall eine zusätzliche visuelle Rückmeldung das Problem verdeutlichen könnte. Dies bestätigt die Annahme aus Kapitel 1, dass für eine optimale MRK ein multimodaler Ansatz benötigt wird.

8.3 Zusammenfassung

In diesem Kapitel wurde ein Prototyp vorgestellt, welcher die im Rahmen dieser Arbeit eingeführten Ansätze zu einem Gesamtsystem kombiniert. Weiterhin wurde dieser Prototyp dazu genutzt, die Funktionsweise des Systems und die Zufriedenheit der Nutzer mit dem System in einer Nutzerstudie zu bestimmen.

Der Prototyp stellt dabei drei generell unabhängige Komponenten zur Verfügung: Das Dialogsystem, die Hardware-Ansteuerung und die Hauptanwendung. Dies erlaubt einen beliebigen Austausch der Komponenten, solange die Anforderungen an die Kommunikationsschnittstelle erfüllt sind. Neben der Sprachsteuerung ist zudem eine Steuerung und Simulation des Roboterarms über eine graphische Nutzeroberfläche umgesetzt worden, welche auch unabhängig vom Dialogsystem Testläufe von modellierten Bewegungen erlaubt.

Die Nutzerevaluation bestand aus vier Teilaufgaben. In der ersten Teilaufgabe wurden Nutzern Beispielbewegungen gezeigt, welche mit dem Prototyp generiert wurden, um zu erfassen, wie sehr diese Bewegungen den Erwartungen der Nutzer entsprechen. Mit anderen Worten wurde also Überprüft, ob die in Kapitel 4 eingeführte Kombination von VPE in Form von KVE valide Ergebnisse liefert. Die Ergebnisse lassen darauf schließen, dass das System größtenteils Bewegungen erzeugen kann, welche den Erwartungen entsprechen. In der zweiten Aufgabe wurde die Zufriedenheit der Nutzer mit dem System erfasst, und damit untersucht, ob durch die Kombination von den in Kapitel 4 - Kapitel 7 eingeführten Ansätzen eine intuitive Robotersteuerung realisierbar ist. Die Ergebnisse haben gezeigt, dass die Nutzer mit dem Gesamtsystem zufrieden

sind, was sich in einem Quesi-Wert von 3,88 auf einer Skala von 1 - 5 widerspiegelt. Teilaufgabe 3 und 4 der Nutzerevaluation galten der Erfassung von Präferenzen der Nutzer bei Rückmeldungen im auditiven und visuellen Bereich. In Aufgabe 3 wurden dabei die möglichen Rückgabetypen des Systems im Fehlerfall bereitgestellt und die Nutzer sollten jeweils begründet einen Favoriten wählen. Die Rückgabemöglichkeiten bezogen sich dabei auf verschiedene Bestandteile des Arbeitsraums, wie den Roboter oder das Werkstück, oder waren abstrakt formuliert. In der letzten Aufgabe ging es schließlich darum, den Bedarf eines bereits implementierten visuellen Feedbacks zu erfassen. Dabei kam heraus, dass zumindest im Fehlerfall eine visuelle Rückmeldung gewollt ist. Bei Teilbewegungen, welche keine Abweichungen enthalten, wurde eine beschleunigte Darstellung als Präferenz identifiziert. Insgesamt hat die Nutzerstudie gezeigt, dass das System eine intuitive MRK ermöglicht und vielversprechende Erweiterungen wurden erfasst.

Kapitel 9

Ausklang

In diesem abschließenden Kapitel werden die Ergebnisse und Erkenntnisse aus dem Hauptteil der vorliegenden Arbeit zusammengefasst und diskutiert, in wie fern die in Kapitel 1 formulierten wissenschaftlichen Fragen beantwortet werden konnten bzw. in wie weit eine Lösung der eingangs formulierten Problemstellung gefunden wurde. Zudem erfolgt ein Ausblick, welcher mögliche zukünftige Erweiterungen der in dieser Arbeit eingeführten Ansätze enthält.

9.1 Zusammenfassung

Das in Kapitel 1 spezifizierte Ziel dieser Arbeit ist die Verbesserung der Flexibilität, sowie der Robustheit einer sprachbasierten Mensch-Roboter-Kommunikation. Dabei sollte die Menge an abbildbaren Instruktionen und abbildbaren Bewegungen erweitert werden. Gleichzeitig wurden Möglichkeiten zur Validierung von Instruktionen und Bewegungen vor der Ausführung auf dem realen Roboter definiert. Damit sollte eine robuste Ausführung ermöglicht werden und damit das Vertrauen des Nutzers in das System gesichert werden. Da ein allgemeingültiges Erreichen dieses Ziels den Rahmen einer einzelnen Arbeit übersteigen würde, wurden eine Reihe an Fragestellung (F1 - F5) eingeführt, welche einen wichtigen Schritt in diese Richtung darstellen. Nachdem Grundlagen und der Stand der Forschung zu relevanten Gebieten eingeführt wurden, und das Grundkonzept, auf dem diese Arbeit aufbaut, erläutert wurde, wurden im Hauptteil der Arbeit Ansätze definiert, welche sich mit F1 - F5 beschäftigen.

In Kapitel 4 wurde dafür ein Ansatz vorgestellt, welcher sich damit auseinandersetzt, in wie weit sich kraftbasierte Bewegungen als eine Kombination von elementaren kraftbasierten Bewegungen darstellen lassen (F1). Der beschriebene Ansatz erlaubt in Form von *Kombinierten Verbalisierten Effekten* eine werkzeugabhängige parallele und sequentielle Kombination von kraftbasierten Bewegungen. Die zugrundeliegende Struktur basiert dabei auf der eines endlichen Automaten und erlaubt neben der Verknüpfung einzelner Bewegungen das Abfangen von Fehlern, sofern diese modellierbar sind. Zudem gilt momentan für die parallele Kombination eine Einschränkung der jeweiligen Task-Frame-Eigenschaften. Generell wurde dadurch jedoch eine Erweiterung der Instruktionsmenge, sowie der Bewegungsmenge ermöglicht.

Das darauf folgende Kapitel 5 beschäftigt sich zunächst damit, in wie weit und in welcher Form Anwender unscharf formulierte Parameter als Synonym für numerische Kraftwerte nutzen (F2). Eine Antwort auf F2 wurde dabei dadurch gefunden, dass eine neue Art des Wizard-of-Oz Experiments umgesetzt und zur Erfassung der Informationen genutzt wurde. Da die Nutzerevaluation unter anderem ergeben hat, dass Nutzer unscharf formulierte Parameter verwenden, wurde in diesem Kapitel zudem ein Ansatz eingeführt, welcher sich damit auseinandersetzt, in wie weit unscharf formulierte Kraftparameter auf kraftbasierte Roboterbewegungen abgebildet werden können (F3). Sofern geeignete Zugehörigkeitsfunktionen, Fuzzy-Regeln und die relevanten Objektinformationen im Kontextwissen enthalten sind, ermöglicht dieser Ansatz eine zufriedenstellende Abbildung von unscharfen Parametern auf numerische Werte. Die Anwendbarkeit dieses Ansatz wurde anhand einer Prototyp-Evaluation gezeigt und stellt eine Erweiterung der Instruktions- und Bewegungsmenge dar.

In der zweiten Hälfte des Hauptteils wurde zunächst ein Ansatz vorgestellt (siehe Kapitel 6), welcher sich damit auseinandersetzt, in wie weit Objekt-Affordanzen zur Auflösung von Mehrdeutigkeiten genutzt werden können (F4). Dafür wurde zunächst definiert, in welcher Weise Affordanzen im Rahmen dieser Arbeit genutzt werden. Danach wurde der zugrundeliegende Ansatz detailliert erklärt und im Rahmen einer Nutzerevaluation mit der Reaktion der Nutzern verglichen. Das Resultat dieses Kapitels war letztlich ein Ansatz, welcher zum einen eine affordanzbasierte Mehrdeutigkeitsauflösung umsetzt, und zum anderen eine Validierung von Instruktionen basierend auf Affordanzen erlaubt. Da somit sowohl unvollständige Instruktionen abbildbar sind, als auch Rückmeldungen in Fehlerfällen definiert wurden, entspricht das bezogen auf g einer Erweiterung der Instruktionsmenge und der Rückmeldungsmenge und damit einer Verbesserung der Flexibilität und der Robustheit des Gesamtsystems.

In Kapitel 7 wurde letztlich ein Ansatz eingeführt, welcher sich damit beschäftigt, in wie weit eine instruierte kraftbasierte Bewegung online simuliert, analysiert und per Interaktion mit einem Nutzer validiert werden kann (F5). Der erwähnte Ansatz bestimmte dabei auf Basis der in diesem Kapitel eingeführten *Parameters-Of-Interest-Maps* bewegungsabhängige Abweichungen von Objekteigenschaften während der Ausführung in einer Festkörper-Simulation. Falls Abweichungen identifiziert werden, verwirft der Roboter die Instruktion jedoch nicht, sondern präsentiert dem Nutzer das Problem und fragt ob er die Bewegung trotzdem ausführen soll. Damit wird selbst eine Ausführung von Bewegungen ermöglicht, welche nicht den gewöhnlichen Erwartungen entsprechen. Bezogen auf die Problemstellung wird mit Hilfe dieses Ansatzes, wie im vorhergehenden Kapitel, die Menge der Rückmeldungen erweitert und somit ein robusteres System erzeugt.

Die einzelnen Ansätze wurden außerdem in Form eines Prototyps umgesetzt und mittels einer abschließenden Nutzerevaluation bewertet (siehe Kapitel 8). Dabei wurden generell vier Erkenntnisse erzielt. Die erste Erkenntnis ist, dass das in Kapitel 4 eingeführte Konzept der *Kombinierten Verbalisierten Effekte* Bewegungen erzeugt, welche sich größtenteils mit den Erwartungen von Nutzern decken. Außerdem wurde gezeigt, dass Nutzer ohne langwierige Einführung zwei Manipulationsaufgaben zusammen mit dem Gesamtsystem lösen konnten und dabei mit der Interaktion zufrieden waren. Außerdem wurde angegeben, dass die Nutzer die Aufgaben mit geringem Aufwand lösen konnten. Zusätzlich wurde eine präferierte Art der Rückmeldung erfasst

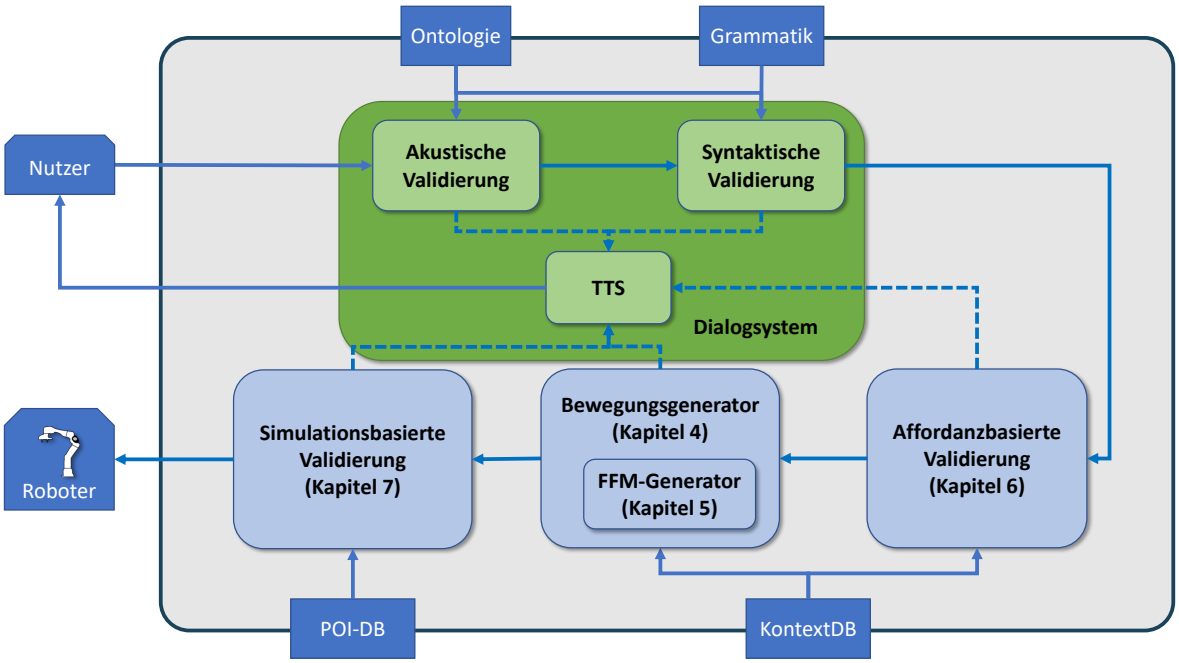


Abb. 9.1: Überblick über das in dieser Arbeit erstellte Gesamtsystem zur flexiblen und robusten sprachbasierten Instruktion von Roboterarmen.

und das Bedürfnis nach einer zusätzlichen visuellen Rückmeldung festgestellt.

Zusammenfassend kann man also feststellen, dass im Rahmen dieser Arbeit ein Gesamtsystem (siehe Abbildung 9.1) eingeführt wurde, welches Ansätze zur Erzeugung von kombinierten kraftbasierten Bewegungen ermöglicht und dabei, sofern relevantes Kontextwissen zu Verfügung steht, die Wahrscheinlichkeit einer robusten Ausführung von Roboterbewegungen erhöht.

9.2 Ausblick

Die vorgestellten Ansätze sind jeweils erste Vorstöße im Rahmen der jeweiligen Forschungsgebiete, welche in Zukunft weiter exploriert werden können. Im folgenden werden Ideen für mögliche Erweiterungen dieser Ansätze vorgestellt, welche eine Verbesserung der Flexibilität und Robustheit eines Systems zur sprachlichen Instruktionen bedeuten würden.

Im Bereich der kombinierten verbalisierten Effekte ist zum einen eine Erweiterung der abbildbaren Instruktionen und zum anderen eine Erhöhung der Mächtigkeit der Bewegungserzeugung notwendig, um die Flexibilität und Robustheit des Systems zu verbessern. Eine Erweiterung der Instruktionen auf Verben oder auch Satzkonstruktionen, welche repetitive Aufgaben symbolisieren, stellt beispielsweise eine hilfreiche Erweiterung dar. Dadurch können Verben wie *Stapeln* oder Satzkonstruktionen wie “Schiebe alle Klötze nach rechts!” abgebildet werden, wodurch eine effizientere Instruktion des Systems ermöglicht wird. Generell stellt dies eine Erweiterung der abbildbaren Instruktionen auf Werkstück-Objektmengen dar. Die Mächtigkeit der Bewegungserzeugung könnte darüber verbessert werden, dass der Taskframe der einzelnen Bewegungen über eine Heuristik bestimmt und nicht fest vorgegeben wird. Diese Heuristik könnte dabei den

Zustand des aktuellen Arbeitsraums oder auch vorhergehende Instruktionen berücksichtigen um einen optimalen Taskframe zu bestimmen.

Bezogen auf unscharfe Instruktion wäre eine Untersuchung von unscharf spezifizierten Resultaten ein interessantes Thema. Unscharf spezifizierte Resultate wurden in der in Kapitel 5 durchgeführten Studie in der Form "Drücke ein *tiefes* Loch in die Knete!" oder auch "Male einen *dicken* Strich!" erfasst. Hier ist also ein Ansatz gesucht, welcher die Formulierungen auf unscharfe Kraftparameter abbildet. Dieser könnte dann auch noch um eine Komponente erweitert werden, welche eine zusätzliche Anpassung von Bewegungen wie "noch tiefer!" ermöglicht, was Nutzern erlaubt, eine Bewegung noch während der Ausführung anzupassen.

Der in dieser Arbeit vorgestellte Ansatz zur affordanzbasierten Validierung (siehe Kapitel 6) könnte auf eine Komponente erweitert werden, welche im Falle einer nicht ausführbaren Instruktion, möglichst ähnliche Verben als Alternative vorschlägt. Als Basis dafür könnte dabei die in [Spangenberg17] eingeführte Kategorisierung von Bewegungen in *absorb, transform* und *change* Bewegungen genutzt werden. Dadurch könnte die Formulierung von Rückmeldungen verbessert und damit die Flexibilität des Systems erhöht werden.

Bei der simulationsbasierten Validierung wäre eine Erweiterung der Objektgeometrien von Starr-Körper auf deformierbare Objekte sinnvoll. Neben einer Simulation von Kleidungsstücken oder Flüssigkeiten über Feder-Masse-Systeme wäre zudem eine Anwendung der Finiten Elemente Analyse denkbar, so dass genauere Informationen über Verformungen von Werkstücken während einer Manipulation zur Verfügung stehen. Generell ist die Anwendung von FEM sehr zeitintensiv. Annäherungen können mittlerweile jedoch auch schon in der in Kapitel 8 erwähnten Bullet Physics Engine umgesetzt werden. Eine Erweiterung des Systems bezogen auf diesen Aspekt würde aber definitiv eine detailliertere Rückmeldung ermöglichen und damit die Flexibilität des Gesamtsystems verbessern. Außerdem wurde während der Evaluierung des Systems festgestellt, dass sich bei einer längeren Benutzung des Systems die Abweichung zwischen dem simulierten Arbeitsplatz und der realen Ist-Position vergrößert. Um diese Abweichung möglichst gering zu halten, wäre an dieser Stelle ein Ansatz interessant, welcher zwischen Teilbewegungen den Zustand des simulierten Arbeitsraums aktualisiert.

Abschließend kann man also feststellen, dass noch eine Reihe an Herausforderungen bewältigt werden muss, bis die Vision einer für den Menschen optimalen Sprachsteuerung von Roboterbewegungen erreicht ist. Trotzdem hat diese Arbeit einen relevanten Beitrag geleistet und hat weitere Ansatzpunkte zur Verbesserung der Mensch-Roboter-Kooperation geliefert.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[Abelha16] Paulo Abelha, Frank Guerin und Markus Schoeler. "A model-based approach to finding substitute tools in 3d vision data". In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, S. 2471–2478.

[Aein13] Mohamad Javad Aein u. a. "Toward a library of manipulation actions based on semantic object-action relations". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, S. 4555–4562.

[AG10] Buddhika AG, Keigo Watanabe, Kazuo Kiguchi und Kiyotaka Izumi. "Interpreting fuzzy linguistic information by acquiring robot's experience based on internal rehearsal". In: *Journal of System Design and Dynamics*. Bd. 4. 2. The Japan Society of Mechanical Engineers, 2010, S. 297–313.

[Aggarwal12] Charu C Aggarwal und ChengXiang Zhai. *Mining text data*. Springer Science & Business Media, 2012. ISBN: 978-1-4614-3222-7.

[Ardón20] Paola Ardón, Èric Pairet, Katrin S Lohan, Subramanian Ramamoorthy und Ronald Petrick. "Affordances in Robotic Tasks–A Survey". In: (2020).

[Bischoff10] Rainer Bischoff u. a. "The KUKA-DLR Lightweight Robot arm-a new reference platform for robotics research and manufacturing". In: *International Symposium on Robotics (ISR)*. VDE. 2010, S. 1–8.

[Bollini13] Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy und Daniela Rus. "Interpreting and executing recipes with a cooking robot". In: *Experimental Robotics*. Springer. 2013, S. 481–495.

[Boteanu16] Adrian Boteanu, Thomas Howard, Jacob Arkin und Hadas Kress-Gazit. "A model for verifiable grounding and execution of complex natural language instructions". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, S. 2649–2654.

[Bothe95] HH Bothe. *Fuzzy Logic: Einführung in Theorie und Praxis*. Berlin: Springer-Verlag, 1995. ISBN: 0-38756-166-8.

[Briggs15] Gordon Michael Briggs und Matthias Scheutz. ""Sorry, I Can't Do That": Developing Mechanisms to Appropriately Reject Directives in Human-Robot Interactions". In: *AAAI fall symposium series*. 2015.

[Bruyninckx96] Herman Bruyninckx und Joris De Schutter. “Specification of force-controlled actions in the”task frame formalisma synthesis”. In: *Transactions on Robotics and Automation*. Bd. 12. 4. IEEE, 1996, S. 581–589.

[Bugmann05] Guido Bugmann und J Norberto Pires. “Robot-by-voice: experiments on commanding an industrial robot using the human voice”. In: *Industrial Robot: An International Journal*. Emerald Group Publishing Limited, 2005.

[Carvalho16] Jônata Tyska Carvalho und Stefano Nolfi. “Behavioural plasticity in evolving robots”. In: *Theory in Biosciences*. Bd. 135. 4. Springer, 2016, S. 201–216.

[Chen20] Haonan Chen, Hao Tan, Alan Kuntz, Mohit Bansal und Ron Alterovitz. “Enabling robots to understand incomplete natural language instructions using commonsense reasoning”. In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, S. 1963–1969.

[Clark96] Herbert H. Clark. *Using language*. Cambridge university press, 1996. ISBN: 978-0-511-62053-9.

[Cruz16] Francisco Cruz, Sven Magg, Cornelius Weber und Stefan Wermter. “Training agents with interactive reinforcement learning and contextual affordances”. In: *Transactions on Cognitive and Developmental Systems*. Bd. 8. 4. IEEE, 2016, S. 271–284.

[Deits13] Robin Deits u. a. “Clarifying commands with information-theoretic human-robot dialog”. In: *Journal of Human-Robot Interaction*. Bd. 2. 2. Journal of Human-Robot Interaction Steering Committee, 2013, S. 58–79.

[Desel96] Jörg Desel und Wolfgang Reisig. “Place/transition Petri nets”. In: *Advanced Course on Petri Nets*. Springer. 1996, S. 122–173.

[Detry11] Renaud Detry u. a. “Learning grasp affordance densities”. In: *Paladyn, Journal of Behavioral Robotics*. Bd. 2. 1. Springer, 2011, S. 1–17.

[DIN03] DIN. “DIN EN ISO 868, Plastics and ebonite-Determination of indentation hardness by means of a durometer (Shore hardness)”. In: 2003.

[Finkemeyer04] Bernd Finkemeyer. “Robotersteuerungsarchitektur auf der Basis von Aktionsprimitiven”. Diss. Insitut fur Robotik, TU Braunschweig, 2004.

[Finkemeyer10] Bernd Finkemeyer, Torsten Kröger und Friedrich M Wahl. “The adaptive selection matrix—A key component for sensor-based control of robotic manipulators”. In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2010, S. 3855–3862.

[Fischer11] Kerstin Fischer. “Interpersonal variation in understanding robots as social actors”. In: *6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE. 2011, S. 53–60.

[Fraser91] Norman M Fraser und G Nigel Gilbert. “Simulating speech systems”. In: *Computer Speech & Language*. Bd. 5. 1. Elsevier, 1991, S. 81–99.

[Gibson77] James J Gibson. “The theory of affordances”. In: *Perceiving, Acting, and Knowing: Toward an Ecological Psychology*. 1977, S. 67–82.

[Gorniak07] Peter Gorniak und Deb Roy. "Situated language understanding as filtering perceived affordances". In: *Cognitive science*. Wiley Online Library, 2007.

[Green04] Anders Green, Helge Huttenrauch und K Severinson Eklundh. "Applying the Wizard-of-Oz framework to cooperative service discovery and configuration". In: *RO-MAN 2004. 13th IEEE International Workshop on Robot and Human Interactive Communication (IEEE Catalog No. 04TH8759)*. IEEE. 2004, S. 575–580.

[Guerin15] Kelleher R Guerin, Colin Lea, Chris Paxton und Gregory D Hager. "A framework for end-user instruction of a robot assistant for manufacturing". In: *international conference on robotics and automation (ICRA)*. IEEE. 2015, S. 6167–6174.

[Harnad90] Stevan Harnad. "The symbol grounding problem". In: *Physica D: Nonlinear Phenomena*. Bd. 42. 1-3. Elsevier, 1990, S. 335–346.

[Heikkilä12] Seppo S Heikkilä, Aarne Halme und André Schiele. "Affordance-based indirect task communication for astronaut-robot cooperation". In: *Journal of Field Robotics*. Bd. 29. 4. Wiley Online Library, 2012, S. 576–600.

[Hemachandra14] Sachithra Hemachandra, Matthew R Walter und Seth J Teller. "Information Theoretic Question Asking to Improve Spatial Semantic Representations". In: *AAAI Fall Symposium Series*. 2014.

[Hessler] Hessler. *Roboter UR 10: Digitalisierung im Handwerk, Kantenfräse*. URL: `https://www.c-hessler.de/roboter-ur-10/` (besucht am 07. 01. 2021).

[Honig18] Shanee Honig und Tal Oron-Gilad. "Understanding and resolving failures in human-robot interaction: Literature review and model development". In: *Frontiers in psychology*. Bd. 9. Frontiers, 2018, S. 861.

[Jamone16] Lorenzo Jamone u. a. "Affordances in psychology, neuroscience, and robotics: A survey". In: *Transactions on Cognitive and Developmental Systems*. Bd. 10. 1. IEEE, 2016, S. 4–25.

[Jayasekara09] Keigo Watanabe Jayasekara AG Buddhika P. und Kiyotaka Izumi. "Understanding user commands by evaluating fuzzy linguistic information based on visual attention". In: *Artificial Life and Robotics*. 2009.

[Jayawardena06] Keigo Watanabe Jayawardena Chandimal und Kiyotaka Izumi. "Learning of object identification by robots commanded by natural language". In: *International Conference on Intelligent Autonomous Systems (IAS)*. 2006.

[Jayawardena07] Keigo Watanabe Jayawardena Chandimal und Kiyotaka Izumi. "Posture control of robot manipulators with fuzzy voice commands using a fuzzy coach-player system". In: *Advanced Robotics*. Bd. 21. 3-4. Taylor & Francis, 2007, S. 293–328.

[Kelley84] John F Kelley. "An iterative design methodology for user-friendly natural language office information applications". In: *Transactions on Information Systems (TOIS)*. Bd. 2. 1. ACM New York, NY, USA, 1984, S. 26–41.

[Khalid17] Asma Khalid. *How A Robot Is Changing Furniture Making At A Factory In Fitchburg*. 2017. URL: https://www.wbur.org/bostonomix/2017/10/30/automation-factory-work (besucht am 07. 01. 2021).

[Knoll97] Alois Knoll, B Hildenbrandt und Jianwei Zhang. "Instructing cooperating assembly robots through situated dialogues in natural language". In: *Proceedings of International Conference on Robotics and Automation (ICRA)*. Bd. 1. IEEE. 1997, S. 888–894.

[Kollar14] Thomas Kollar, Stefanie Tellex, Deb Roy und Nicholas Roy. "Grounding verbs of motion in natural language commands to robots". In: *Experimental robotics*. Springer. 2014, S. 31–47.

[Kostavelis12] Ioannis Kostavelis, Lazaros Nalpantidis und Antonios Gasteratos. "Collision risk assessment for autonomous robots by offline traversability learning". In: *Robotics and Autonomous Systems*. Bd. 60. 11. Elsevier, 2012, S. 1367–1376.

[Kresse17] Ingo Kresse. "A semantic constraint-based Robot Motion Control for Generalizing Everyday Manipulation Actions". Diss. Technische Universität München, 2017.

[Kunze17] Lars Kunze und Michael Beetz. "Envisioning the qualitative effects of robot manipulation actions using simulation-based projections". In: *Artificial Intelligence*. Bd. 247. Elsevier, 2017, S. 352–380.

[Lemaignan17] Séverin Lemaignan, Mathieu Warnier, E Akin Sisbot, Aurélie Clodic und Rachid Alami. "Artificial cognition for social human–robot interaction: An implementation". In: *Artificial Intelligence*. Bd. 247. Elsevier, 2017, S. 45–69.

[Lin98] Chin-Teng Lin und Ming-Chih Kan. "Adaptive fuzzy command acquisition with reinforcement learning". In: *Transactions on Fuzzy Systems*. IEEE. 1998.

[Liu10] Changsong Liu, Jacob Walker und Joyce Y Chai. "Ambiguities in spatial language understanding in situated human robot dialogue". In: *AAAI Fall Symposium Series*. AAAI, 2010.

[Liu19] Rui Liu und Xiaoli Zhang. "A review of methodologies for natural-language-facilitated human-robot cooperation". In: *International Journal of Advanced Robotic Systems*. Bd. 16. 3. SAGE Publications Sage UK: London, England, 2019, S. 1729881419851402.

[Marge15] Matthew Marge und Alexander Rudnicky. "Miscommunication recovery in physically situated dialogue". In: *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. 2015, S. 22–31.

[Marge19] Matthew Marge und Alexander I Rudnicky. "Miscommunication detection and recovery in situated human–robot dialogue". In: *Transactions on Interactive Intelligent Systems (TiiS)*. Bd. 9. 1. ACM New York, NY, USA, 2019, S. 1–40.

[Mason81] Matthew T Mason. “Compliance and force control for computer controlled manipulators”. In: *Transactions on Systems, Man, and Cybernetics*. Bd. 11. 6. IEEE, 1981, S. 418–432.

[Matuszek13] Cynthia Matuszek, Evan Herbst, Luke Zettlemoyer und Dieter Fox. “Learning to parse natural language commands to a robot control system”. In: *Experimental robotics*. Springer. 2013, S. 403–415.

[Min16] Huaqing Min, Ronghua Luo, Jinhui Zhu, Sheng Bi u. a. “Affordance research in developmental robotics: A survey”. In: *Transactions on Cognitive and Developmental Systems*. Bd. 8. 4. IEEE, 2016, S. 237–255.

[Min17] Huaqing Min, Jinhui Zhu, Xinshi Xu, Pengshuai Yin, Guofei Zheng u. a. “Affordance learning and inference based on vision-speech association in human-robot interactions”. In: *International Conference on Robotics and Biomimetics (ROBIO)*. IEEE. 2017, S. 743–749.

[Misra16] Dipendra K Misra, Jaeyong Sung, Kevin Lee und Ashutosh Saxena. “Tell me dave: Context-sensitive grounding of natural language to manipulation instructions”. In: *The International Journal of Robotics Research*. Bd. 35. 1-3. SAGE Publications Sage UK: London, England, 2016, S. 281–300.

[Montesano07] Luis Montesano, Manuel Lopes, Alexandre Bernardino und Jose Santos-Victor. “Affordances, development and imitation”. In: *6th International Conference on Development and Learning*. IEEE. 2007, S. 270–275.

[Moratz08] Reinhard Moratz und Thora Tenbrink. “Affordance-based human-robot interaction”. In: *Towards Affordance-Based Robot Control*. Springer, 2008, S. 63–76.

[Mösenlechner13] Lorenz Mösenlechner und Michael Beetz. “Fast temporal projection using accurate physics-based geometric reasoning”. In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2013, S. 1821–1827.

[Muthugala14] M. A. V. J. Muthugala und A. G. B. P. Jayasekara. “Synthesizing fuzzy linguistic vocal responses by adapting perception of robot based on visual attention”. In: *International Conference on Information and Automation for Sustainability*. IEEE. 2014.

[Muthugala16] MA Viraj J Muthugala und AG Buddhika P Jayasekara. “Enhancing human-robot interaction by interpreting uncertain information in navigational commands based on experience and environment”. In: *International Conference on Robotics and Automation (ICRA)*. 2016.

[Muthugala18] MA Viraj J Muthugala und AG Buddhika P Jayasekara. “A Review of Service Robots Coping with Uncertain Information in Natural Language Instructions”. In: *IEEE Access*. 2018.

[Naumann10] Anja Naumann und Jörn Hurtienne. “Benchmarks for intuitive interaction with mobile devices”. In: *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*. 2010, S. 401–402.

[Onnasch16] Linda Onnasch, Xenia Maier und Thomas Jürgensohn. *Mensch-Roboter-Interaktion-Eine Taxonomie für alle Anwendungsfälle*. Bundesanstalt für Arbeitsschutz und Arbeitsmedizin Dortmund, 2016.

[Pek16] Christian Pek, Arne Muxfeldt und Daniel Kubus. “Simplifying synchronization in cooperative robot tasks-an enhancement of the manipulation primitive paradigm”. In: *21st International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE. 2016, S. 1–8.

[Pfalzgraf08] A Pfalzgraf, N Pfleger, J Schehl und J Steigner. “ODP-Ontology-based Dialogue Platform”. In: *Technical Report, SemVox GmbH. http://www. semvox. de/whitepapers*. 2008.

[Pulasinghe04] Koliya Pulasinghe, Keigo Watanabe, Kiyotaka Izumi und Kazuo Kiguchi. “Modular fuzzy-neuro controller driven by spoken language commands”. In: *Transactions on Systems, Man, and Cybernetics*. IEEE. 2004.

[Ralph08] Maria Ralph und Medhat A Moussa. “Toward a natural language interface for transferring grasping skills to robots”. In: *Transactions on Robotics*. Bd. 24. 2. IEEE, 2008, S. 468–475.

[Riek12] Laurel D Riek. “Wizard of oz studies in hri: a systematic review and new reporting guidelines”. In: *Journal of Human-Robot Interaction*. Bd. 1. 1. Journal of Human-Robot Interaction Steering Committee, 2012, S. 119–136.

[Rockel15] Sebastian Rockel u. a. “Integrating physics-based prediction with semantic plan execution monitoring”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, S. 2883–2888.

[Sattar14] Junaed Sattar und James J Little. “Ensuring safety in human-robot dialog—A cost-directed approach”. In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, S. 6660–6666.

[Schiffer12] Alexander Ferrein Schiffer Stefan und Gerhard Lakemeyer. “Reasoning with qualitative positional information for domestic domains in the situation calculus”. In: *Journal of Intelligent & Robotic Systems*. 2012.

[Schmidt11] Thorsten W Schmidt. “Temporale Fuzzy-Logik-Eine Vereinigung der Temporal-Logik und Fuzzy-Logik anhand von vorausschauenden Wartungs- und Überwachungssystemen”. Diss. Universität Bayreuth, 2011.

[Skubic04] Marjorie Skubic u. a. “Spatial language for human-robot dialogs”. In: *Transactions on Systems, Man, and Cybernetics*. Bd. 34. 2. IEEE. 2004, S. 154–167.

[Song15] Hyun Oh Song, Mario Fritz, Daniel Goehring und Trevor Darrell. “Learning to detect visual grasp affordance”. In: *Transactions on Automation Science and Engineering*. Bd. 13. 2. IEEE, 2015, S. 798–809.

[Spangenberg17] Michael Spangenberg. “Intuitive Roboterkommandierung basierend auf Verbalisierten Physikalischen Effekten”. Diss. Universität Bayreuth, 2017.

[Steinfeld09] Aaron Steinfeld, Odest Chadwicke Jenkins und Brian Scassellati. “The oz of wizard: simulating the human for interaction research”. In: *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*. 2009, S. 101–108.

[Stenmark15] Maj Stenmark. “Instructing industrial robots using high-level task descriptions”. Diss. Departement of Computer Science, Lund University, 2015.

[Stenmark17] Maj Stenmark, Mathias Haage, Elin Anna Topp und Jacek Malec. “Making Robotic Sense of Incomplete Human Instructions in High-Level Programming for Industrial Robotic Assembly.” In: *AAAI Workshops*. 2017.

[Tellex13] Stefanie Tellex u. a. “Toward information theoretic human-robot dialog”. In: *Robotics*. Bd. 32. MIT Press, 2013, S. 409–417.

[Tellex20] Stefanie Tellex, Nakul Gopalan, Hadas Kress-Gazit und Cynthia Matuszek. “Robots that use language”. In: *Annual Review of Control, Robotics, and Autonomous Systems*. Bd. 3. Annual Reviews, 2020, S. 25–55.

[Tenorth10] Moritz Tenorth, Daniel Nyga und Michael Beetz. “Understanding and executing instructions for everyday manipulation tasks from the world wide web”. In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2010, S. 1486–1491.

[Thomas09] Ulrike Thomas. *Automatisierte Programmierung von Robotern für Montageaufgaben*. Gesellschaft für Informatik, 2009.

[Thomas13] Ulrike Thomas, Gerd Hirzinger, Bernhard Rumpe, Christoph Schulze und Andreas Wortmann. “A new skill based robot programming language using uml/p statecharts”. In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2013, S. 461–466.

[Ugur11] Emre Ugur, Erhan Oztop und Erol Sahin. “Goal emulation and planning in perceptual space using learned affordances”. In: *Robotics and Autonomous Systems*. Bd. 59. 7-8. Elsevier, 2011, S. 580–595.

[Weidauer14] Ingo Weidauer, Daniel Kubus und Friedrich M Wahl. “A hierarchical extension of manipulation primitives and its integration into a robot control architecture”. In: *International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, S. 5401–5407.

[Williams19] Tom Williams, Fereshta Yazdani, Prasanth Suresh, Matthias Scheutz und Michael Beetz. “Dempster-shafer theoretic resolution of referential ambiguity”. In: *Autonomous Robots*. Bd. 43. 2. Springer, 2019, S. 389–414.

[Yamaoka09] Fumitaka Yamaoka, Takayuki Kanda, Hiroshi Ishiguro und Norihiro Hagita. “A model of proximity control for information-presenting robots”. In: *Transactions on Robotics*. Bd. 26. 1. IEEE, 2009, S. 187–195.

[Zadeh65] Lotfi A Zadeh. “Fuzzy sets”. In: *Information and Control*. Bd. 8. 3. 1965, S. 338–353.

[Zhu15] Yixin Zhu, Yibiao Zhao und Song Chun Zhu. “Understanding tools: Task-oriented object modeling, learning and recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2015, S. 2855–2864.

[Zieliński95] Cezary Zieliński. “Robot programming methods”. In: Oficyna Wydawnicza Politechniki Warszawskiej, 1995.

Eigene Publikationen

[Deese16] Kevin Deese, Daniel Goller, Felix Viebahn, Benjamin Wipfler und Kim Wölfel. *Automatisierte schichtweise Materialzuweisung für die FE-Analyse am Beispiel von biologischen Strukturen*. Bd. 18. 2016.

[Gradmann18] Michael Gradmann, Kim Wölfel und Dominik Henrich. "Kinesthetic Robot Force Response enhanced by the Laws of Physics". In: *27th International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE. 2018, S. 921–927.

[Wölfel18a] Kim Wölfel und Dominik Henrich. "Grounding verbs for tool-dependent, sensor-based robot tasks". In: *27th International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE. 2018, S. 378–383.

[Wölfel18b] Kim Wölfel, Tobias Werner und Dominik Henrich. "GroundSim: Animating Human Agents for Validated Workspace Monitoring". In: *Tagungsband des 3. Kongresses Montage Handhabung Industrieroboter*. Springer, 2018, S. 205–213.

[Wölfel19a] Kim Wölfel und Dominik Henrich. "Gesagt, (gefragt,) getan". In: *handling 06/2019*. 2019.

[Wölfel19b] Kim Wölfel und Dominik Henrich. "Grounding of uncertain force parameters in spoken robot commands". In: *International Conference on Robotics in Alpe-Adria Danube Region*. Springer. 2019, S. 194–201.

[Wölfel20a] Kim Wölfel und Dominik Henrich. "Affordance Based Disambiguation and Validation in Human-Robot Dialogue". In: *Annals of Scientific Society for Assembly, Handling and Industrial Robotics*. Springer Vieweg, Berlin, Heidelberg, 2020, S. 307–317.

[Wölfel20b] Kim Wölfel und Dominik Henrich. "Simulation-Based Validation of Robot Commands for Force-Based Robot Motions". In: *German Conference on Artificial Intelligence (Künstliche Intelligenz)*. Springer. 2020, S. 334–340.

[Wölfel20c] Kim Wölfel und Dominik Henrich. "Wizard of Botz: A Novel Approach to the Wizard of Oz Experiment". In: *International Conference on Robotics in Alpe-Adria Danube Region*. Springer. 2020, S. 557–564.

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die von mir angegebenen Quellen und Hilfsmittel verwendet habe. Weiterhin erkläre ich, dass ich die Hilfe von gewerblichen Promotionsberatern bzw. -vermittlern oder ähnlichen Dienstleistern weder bisher in Anspruch genommen habe, noch künftig in Anspruch nehmen werde. Zusätzlich erkläre ich hiermit, dass ich keinerlei frühere Promotionsversuche unternommen habe.

Bayreuth, den

Unterschrift