STUTTGARTER BEITRÄGE ZUR PRODUKTIONSFORSCHUNG

CHRISTIAN FRIEDRICH

Robotermanipulationsfähigkeiten zur Automatisierung von Instandhaltungsaufgaben

Universität Stuttgart

Fraunhofer IPA

STUTTGARTER BEITRÄGE ZUR PRODUKTIONSFORSCHUNG BAND 89

Christian Friedrich

Robotermanipulationsfähigkeiten zur Automatisierung von Instandhaltungsaufgaben

FRAUNHOFER VERLAG

Kontaktadresse:
Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA, Stuttgart
Nobelstraße 12, 70569 Stuttgart
Telefon 07 11/9 70-11 01
info@ipa.fraunhofer.de; www.ipa.fraunhofer.de

STUTTGARTER BEITRÄGE ZUR PRODUKTIONSFORSCHUNG

Titelbild: © Christian Friedrich

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über www.dnb.de abrufbar.
ISSN: 2195-2892
ISBN (Print): 978-3-8396-1484-6

D 93

Zugl.: Stuttgart, Univ., Diss., 2019

Druck: Mediendienstleistungen des Fraunhofer-Informationszentrum Raum und Bau IRB, Stuttgart
Für den Druck des Buches wurde chlor- und säurefreies Papier verwendet.

Fraunhofer-Informationszentrum Raum und Bau IRB
Postfach 80 04 69, 70504 Stuttgart
Nobelstraße 12, 70569 Stuttgart
Telefon 07 11 9 70-25 00
Telefax 07 11 9 70-25 08
E-Mail verlag@fraunhofer.de
URL http://verlag.fraunhofer.de

Robotermanipulationsfähigkeiten zur Automatisierung von Instandhaltungsaufgaben

Von der Graduate School of Excellence advanced Manufacturing Engineering
der Universität Stuttgart
zur Erlangung der Würde eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Abhandlung

Vorgelegt von

Christian Friedrich, M. Eng.
aus Gundelsheim am Neckar

Hauptberichter:	Univ.-Prof. Dr.-Ing. Dr. h. c. mult. Alexander Verl
Mitberichter:	Univ.-Prof. Dr.-Ing. Bernd Bertsche
Tag der mündlichen Prüfung:	15.04.2019

Institut für Steuerungstechnik der Werkzeugmaschinen
und Fertigungseinrichtungen
der Universität Stuttgart

2019

Vorwort

Die vorliegende Arbeit entstand während meiner Zeit als wissenschaftlicher Mitarbeiter am Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW) und der Graduate School of Excellence advanced Manufacturing Engineering (GSaME) der Universität Stuttgart. Der Deutschen Forschungsgemeinschaft (DFG) danke ich für die Finanzierung der Arbeit. Auch bei der Firma Schunk GmbH & Co. KG möchte ich mich für die Unterstützung durch Hardware bedanken.

Mein besonderer Dank gilt Herrn Prof. Dr.-Ing. Dr. h. c. mult. Alexander Verl für die wissenschaftliche Betreuung sowie die Freiheit bei der Gestaltung der Arbeit. Ebenso möchte ich mich bei Herrn Prof. Dr.-Ing. Bernd Bertsche für die Anmerkungen und Diskussionen während der Bearbeitung der Themenstellung und die Übernahme des Mitberichts bedanken.

Ein ganz besonderer Dank geht an Herrn Dr.-Ing. Akos Csiszar. Ohne die vielen Gespräche und Anregungen während der gesamten Zeit, wären einige Ideen definitiv nicht entstanden. Auch Herrn Dr.-Ing. Armin Lechler danke ich herzlich für seine Zeit und Impulse während der Arbeit. Ebenso möchte ich mich bei Euch nochmals für die Durchsicht und Korrekturen des Manuskripts bedanken. Desgleichen geht ein Dank an Frau Heide Kreuzburg für alle organisatorischen Themen rund um diese Schrift. Meinen ehemaligen Mitbewohnern Michael Voß, Christian Scheifele und Anja Elser danke ich für die hervorragende Büroatmosphäre und für die durchaus kurzweilige und schöne Zeit.

Auch gilt mein Dank allen Mitarbeitern am ISW, von der mechanischen und elektrischen Werkstatt, über die Verwaltung und allen wissenschaftlichen Mitarbeitern für die durchweg positive Gemeinschaft, die einem gerne an die Zeit ans ISW zurückdenken lässt. Des Weiteren geht mein Dank an alle Studenten, die mich bei praktischen Arbeiten unterstützt haben und ohne die viele Ideen wohl nie wären umgesetzt worden. Allen voran geht mein Dank an Herrn Ralf Gulde sowie an Christian von Arnim, Nils Bayer, Tobias Doser, Stefanie Herre, Michael Hertneck, Manuel Jung, Marcel Pelzer, Benedikt Rüther, Yuting Tang, Kai Wäller und Viktor Zielke.

Nicht zuletzt möchte ich mich bei meinen Eltern bedanken, ohne die es mir nicht möglich gewesen wäre diesen Weg zu gehen. Danke für die Unterstützung und die Freiheit meinen Weg zu finden! Auch möchte ich mich bei meinem Bruder bedanken, für die vielen fachlichen und nichtfachlichen Gespräche sowie die Durchsicht der Arbeit. Zugleich geht ein herzlicher Dank an meine Frau für den steten Rückhalt, die Geduld und den Freiraum in den gemeinsamen Jahren.

Gundelsheim, den 28.02.2018 Christian Friedrich

Kurzinhalt

In Produktionssystemen spielt die Anlagenverfügbarkeit und Produktqualität, vor allem im Hinblick auf ökonomische Unternehmensziele, eine entscheidende Rolle. Damit dies erreicht werden kann, unterliegen Produktionseinrichtungen regelmäßigen Instandhaltungsarbeiten. Während für Inspektionsaufgaben bereits Verfahren zur Verfügung stehen, welche die automatisierte Fehlerdetektion, -isolation und -identifikation erlauben, bestehen bisher keine Systeme, die eine automatische Wiederherstellung des Sollzustands ermöglichen. Aufgrund dessen untersucht diese Arbeit neuartige Manipulationsfähigkeiten, die es einem autonomen Robotersystem erlauben, Wartungs- und Instandsetzungsaufgaben zu automatisieren, hierdurch den menschlichen Akteur unterstützen und langfristig zu einer Attraktivitätssteigerung der Produktion in Hochlohnländern führen könnten.

Damit Robotersysteme derart komplexe Aufgaben unter realitätsnahen Bedingungen autonom lösen können, entwickelt diese Arbeit spezielle Fähigkeiten zur Planung, Steuerung und Regelung von Robotermanipulationen. Ein besonderes Hauptaugenmerk bei der Entwicklung dieser Methoden liegt dabei vor allem auf der zeiteffizienten Planung sowie der Möglichkeit zur Kompensation von Umweltunsicherheiten zwischen a priori und Sensordaten.

Für die Aufgabenplanung wird ein Verfahren entwickelt, welches auf Basis von CAD- und visuellen Sensordaten, die notwendigen Manipulationen in Form symbolischer Anweisungen generiert. Durch einen neuartigen stichprobenbasierten Ansatz wird eine zeiteffiziente Berechnung möglicher Demontageräume erlaubt. Damit eine zielgerichtete Akquirierung relevanter Sensordaten ermöglicht werden kann, wird ein Algorithmus vorgestellt, der mittels aufgabenabhängiger Metriken die Kombination von Kartenexploration und Objekterkennung in einer geeigneten Sensorpose zulässt. Zur Planung einer Bewegungsbahn, für die Ausführung der einzelnen Manipulationsaufgaben, werden, dem Stand der Technik gemäß, bekannte globale Bahnplanungsverfahren verwendet. Jedoch wird eine Vorverarbeitungsstrategie vorgeschlagen, die auf Grundlage einer adaptiven Schrittweitensteuerung eine deutliche Reduktion des Planungsraums zulässt, wodurch eine niedrigere Planungszeit bei höherer Erfolgsrate in der Lösungsfindung erzielt wird. Die aus der Planung generierte Beschreibung wird weitergehend in ein Anwenderprogramm umgesetzt. Hierzu wird ausgehend von einer allgemeinen Dekompositionsvorschrift die Generierung elementarer Roboterkontrollanweisungen erlaubt, welche aufgabenabhängig über propriozeptive oder exterozeptive Regler ausgeführt werden.

Die entwickelten Manipulationsfähigkeiten werden in eine Steuerungsarchitektur integriert und ganzheitlich an einem Demonstratorsystem, anhand praxisrelevanter Anwendungsfälle, experimentell validiert.

Short Summary

In current production systems, plant availability and product quality play an important role, especially concerning economic objectives. In order to be able to consider these factors, production systems are subject to regular maintenance. Whereas there are already methods for inspection tasks allowing automated fault-diagnosis, there are no methods available to solve the following service or repair task. For the possibility of automated maintenance and repair, strategies are investigated which permit a robot-based automation, thus support the human worker and, in the long run, enhance the attractiveness of production in high-wage countries.

For the ability of robotic systems to solve such complex tasks autonomously under realistic conditions, special skills for planning and control of robotic manipulations are developed. A particular emphasis in developing these methods is placed on time-efficient planning algorithms as well as to incorporate environmental uncertainties between existing a priori and sensor data.

For task planning a method is proposed, which generates, based on CAD and visual sensor data the required manipulations in the form of symbolic primitives. Through a novel sampling-based approach an extremely time-efficient computation of disassembly spaces is feasible. In order to enable a targeted acquisition of relevant sensor data, an algorithm is presented that allows by means of task-dependent metrics, the combination of map exploration and object recognition in a suitable sensor pose. For path planning, for the execution of the individual manipulation tasks, state-of-the-art algorithms are used. But, a pre-processing strategy is introduced, which allows a considerable reduction of the planning space based on an adaptive step size control, which achieves a significantly shorter planning period and a higher success rate in finding a possible solution. The description generated from the planning is further translated in an executable robot program. For this purpose, the decomposition into elementary robot control instructions based on a general decomposition rule is permitted, which are executed via proprioceptive or exteroceptive controllers in a task-oriented way.

The developed manipulation capabilities are integrated into a robot control architecture and are validated experimentally on a demonstrator system, using representative real-world applications.

„Trotzdem bin ich bei meinem Umweg über die Berge viel weiter gekommen, als wenn ich den flachen Pfaden gefolgt wäre.“

Reinhard Karl

Inhaltsverzeichnis

Abkürzungsverzeichnis

Abkürzungen

BoVW	Bag of Visual Words	**LOG**	Laplace of Gaussian
CAD	Computer Aided Design	**MEMS**	Mikro-Elektro-Mechanische-Systeme
CAx	Computer Aided x	**MLESAC**	Maximum Likelihood Estimation Sample Consensus
CANopen	Controller Area Network	**MST**	Minimum Spanning Tree
CCD	Charge-Coupled Device	**NBV**	Next-Best-View
CMOS	Complementary metal-oxide-semiconductor	**NN**	Nearest-Neighbor (Heuristics)
CMS	Condition Monitoring System	**NP**	Komplexitätsklasse nichtdeterministisch polynomiell
DBG/ NDBG	Directional Blocking Graph/ Non-Directional Blocking Graph	**OPC**	Open Platform Communications
DBSCAN	Density Based Spatial Clustering of Application with Noise	**PBVS**	Position-based visual servoing
DFG	Deutsche Forschungs-gemeinschaft	**PCL**	Point Cloud Library
DMS	Dehnungsmessstreifen	**PFH**	Point Feature Histogram
DOG	Difference of Gaussian	**PI**	Proportional Integral
DPM	Disassembly Precedence Matrix	**PnP**	Perspective-n-Point (Problem)
EKF	Extended Kalman Filter	**POSIT**	Pose from Orthography and Scaling with Iterations
FIR	Finite Impulse Response	**PRM**	Probabilistic Roadmaps
FTC	Fault Tolerant Control	**RANSAC**	Random Sample Consensus
GPU	Graphics Processing Unit	**RGBD**	Red Green Blue Depth
HOG	Histogram of Oriented Gradients	**ROC**	Receiver-Operating-Characteristic
IBVS	Image-based visual servoing	**ROS**	Robot Operating System
ICP	Iterative Closest Point	**ROI**	Region of Interest
KMR	Kraft-Momenten-Regelung	**RRT**	Rapidly-Exploring Random Tree

SFB	Sonderforschungsbereich	**SVM**	Support Vector Machine
SHOT	Signature of Histograms of OrienTations	**TCP**	Tool Center Point
SIFT	Scale Invariant Feature Transformation	**TSP**	Travelling Salesman Problem
SLAM	Synchronous Localisation and Mapping	**UKF**	Unscented Kalman Filter
SPA	Sense-Plan-Act	**urdf**	unified robot description file
SPS	Speicherprogrammierbare Steuerung	**VFH**	Vector Field Histogram
SVD	Singular Value Decomposition	**VJD**	Viola-Jones Detector

Wichtigsten Formelzeichen und Notationen

Skalare und Symbole

f_x, f_y	Brennweite
s_Δ	Adaptive Schrittweite
u_0, v_0	Optisches Zentrum
$\mathcal{FG}$	Featuregeometrie
$\mathcal{MP}$	Manipulationsprimitiv
A	Übertragungsfunktion Admittanz
C	Komponente/Bauteil
I	Visuelle Sensordaten
IG	Informationgain
$P(y\|x)$	Bedingte Wahrscheinlichkeit
S	Erfolgsrate Planung
Z	Übertragungsfunktion Impedanz
s	Laplace-Variable
t	Zeit
$\mathcal{AG}$	Baugruppenmodell
$\mathcal{GP}$	Geometrisches Primitiv
$\mathcal{PS}$	Produktionssystemmodell
$\mathcal{PVM}$	Probabilistische Voxelkarte
$\mathcal{sdof}$	Symbolische Freiheitsgradrelation
$\mathcal{ssr}$	Symbolisch, räumliche Relation
$\mathcal{v}$	Voxel
λ	Abbruchbedingung
ρ	Belegtheitsgrad
τ	Werkzeug und -kommando

Vektoren und Matrizen

$\underline{\tau}$	Antriebsvektor
$\underline{\not{f}}, \underline{\dot{\not{f}}}$	Featureposition, -geschwindigkeit
$\underline{f}$	Kraft- und Momentenvektor
$\underline{r}$	Kartesischer Punkt

$\underline{q}, \underline{\dot{q}}, \underline{\ddot{q}}$	Gelenkwinkel, -geschwindigkeit, -beschleunigung
$\underline{p}, \underline{\dot{p}}, \underline{\ddot{p}}$	Bahnpunkt/ Pose, -geschwindigkeit, -beschleunigung
$\underline{d}$	Demontagerichtung
$\underline{M}$	Massenmatrix
$\underline{K}$	Intrinsische Kameramatrix
$\underline{D}$	Dämpfungsmatrix
$\underline{C}$	Steifigkeitsmatrix
$\underline{f_{ssr}}$	Kontaktvektor einer symbolisch, räumlichen Relation
${}^{i}_{j}\underline{T}$	Homogene Transformationsmatrix
$\underline{J}$	Jakobi-Matrix

Räume, Gruppen und Mengen

$\mathcal{V}$	Voxelmenge
$\mathcal{U}\{-1, \dots, 1\}$	Gleichverteilung
$\mathcal{C}$	Konfigurationsraum
$SO(3)$	Spezielle orthogonale Gruppe
$SE(3)$	Spezielle euklidische Gruppe
$\mathbb{W}_d$	Demontageraum
$\mathcal{W}$	Arbeitsraum
$\mathcal{S}^3$	Einheitssphäre
$\mathcal{G}_{sdoff}$	Relationaler Graph
$\mathcal{G}_{ssr}$	Graph mit symbolisch, räumlichen Relationen/ Kontaktgraph

Operatoren und Funktionen

$d_E\left(\underline{x}, \underline{y}\right)$	Euklidischer Abstand
$\mathcal{ID}$	Inverse Dynamik
$\mathcal{IK}$	Inverse Kinematik
$\mathrm{bel}(X)$	Degree of Belief über X
$diag\left(\underline{A}\right)$	Diagonalmatrix
$mod(x, y)$	Modulo-Operator
$rank\left(\underline{A}\right)$	Rang einer Matrix
$\mathcal{DD}$	Direkte Dynamik
$\mathcal{DK}$	Direkte Kinematik

Indexe

$-$	A priori
$+$	A posteriori
$*$	Optimale Lösung
A	Zugänglichkeit
D	Demontage
V	Sichtbarkeit
p	Planung
pre	Vorverarbeitung

Abbildungsverzeichnis

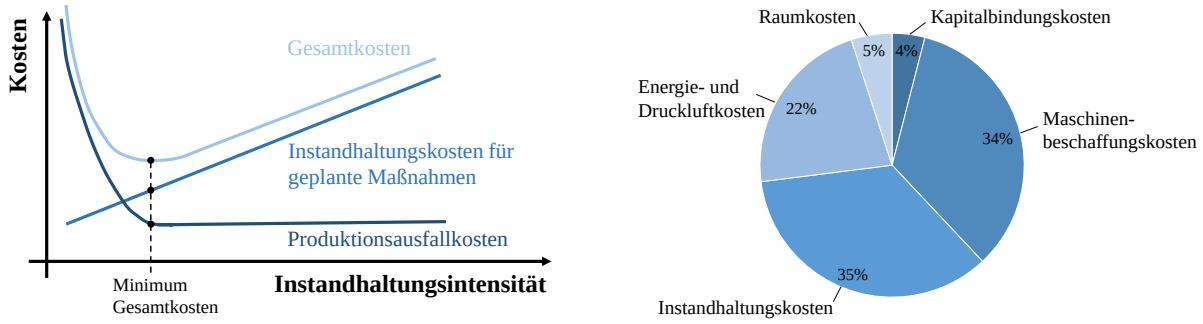
Tabellenverzeichnis

1 Einleitung

„Was immer du tun kannst oder wovon du
träumst, fang damit an. Mut hat Genie,
Kraft und Zauber in sich.“
Johann Wolfgang von Goethe

In heutigen Produktionssystemen spielt die Maschinen- und Anlagenverfügbarkeit eine immer wichtiger werdende Rolle und bildet ein wegweisendes Differenzierungsmerkmal im industriellen, globalen Wettbewerb. Ein wesentlicher Treiber zur Minimierung von Ausfallzeiten sowie zur Erhaltung von Prozess- und Produktqualität in Unternehmensstrukturen stellt dabei die Instandhaltung dar. Jedoch beeinflussen sich die Zuverlässigkeit, die Verfügbarkeit, die Instandhaltbarkeit und die Kosten einer Maschine oder Anlage gegenseitig und zum Investitionsentscheid müssen die kompletten Lebenszykluskosten betrachtet werden [1]. Dies erschwert die Organisation und Ausrichtung der Instandhaltungsaufwendungen zusätzlich, da diese dazu beitragen, die Gesamtkosten zu minimieren und über die Wirtschaftlichkeit von Unternehmen entscheiden, vergleiche Abbildung 1-1 (a). Bei den Instandhaltungskosten handelt es sich dabei um die einzige Kostenart auf die nach Inbetriebnahme einer Produktionsanlage noch Einfluss genommen werden kann [2]. Studien zeigen, dass die jährlichen Instandhaltungskosten circa 5% des Wiederbeschaffungswertes der Anschaffungskosten von Maschinen und Anlagen betragen [3]. Bei Werkzeugmaschinen entfallen auf die Instandhaltungskosten durchaus 15% [4] bis zu 35% [5] der Gesamtlebenszykluskosten, vergleiche hierzu Abbildung 1-1 (b).

Neben betriebs- und volkswirtschaftlichen Gesichtspunkten spielen auch soziale und technische Aspekte eine Rolle, wenn es um die wachsende Bedeutung der Instandhaltung in Produktions- und Unternehmensverbänden geht [3]. So werden heutzutage durch den punktuellen, technischen Einsatz automatischer Diagnosesysteme Fehler frühzeitig erkannt und somit die Ausfallzeiten reduziert und folglich eine bessere Ausnutzung des Abnutzungsvorrates gewährleistet. Durch die Anwendung zustandsorientierter Instandhaltung lassen sich Studien zufolge bis zu 21% der Produktionsausfälle sowie eine Reduktion der Instandhaltungskosten von 11% erreichen [6].



(a) (b)

Abbildung 1-1: Kostenverläufe in Abhängigkeit der Instandhaltungsintensität [7] (a); Lebenszykluskosten am Beispiel einer Werkzeugmaschine im Betrachtungszeitraum von zehn Jahren, aus [5] (b)

Dies zeigt, dass unter anderem die Automatisierung von Fehlerdiagnosemaßnahmen bereits einen erheblichen Anteil zur Effektivitätssteigerung der Instandhaltung beiträgt.

Hohe Stillstandszeiten, wie etwa in der automobilen Produktionstechnik, mit bis zu 14% [8], zeigen, dass die Instandhaltung hohe Potentiale bisher noch nicht ausreichend ausnutzt und ein erheblicher Optimierungsbedarf besteht. Bei Werkzeugmaschinen sind die Ausfallursachen mit bis zu 43% alleinig auf Verschleiß und Schmutz zurückzuführen [5]. Dies sind jedoch vermeidbare Ausfälle, da durch eine bedarfsgerechte und flexibel gesteuerte Durchführung von Reinigungs- und Instandhaltungsaufgaben eine deutliche Minimierung erzielbar wäre. Die Vermeidung von Stillstandszeiten wird von den jeweiligen Entscheidungsträgern zum größten Teil als sehr wichtig erachtet [6], da neben den direkten Kosten auch unmittelbare Folgewirkungen, wie Imageverluste, auftreten. Allerdings erfordert die Durchführung von Wartungs- und Instandsetzungsaufgaben den Einsatz von Instandhaltungspersonal, welche den größten Kostenverursacher im Instandhaltungsbereich darstellt [9], da Ausgaben für manuelle Arbeiten fast die Hälfte der gesamten Instandhaltungskosten betragen [10]. Dies ist für die Erhaltung des Produktionsstandorts Deutschland ein schritt- und zukunftsweisender Faktor, da Lohnkosten das maßgebliche Mittel der Wahl zur Standortfestlegung darstellen [11].

Damit auch weiterhin hochqualitative Produkte unter Beachtung des Kostendrucks in Hochlohnländern produzierbar sind, müssen zukünftig geeignete autonome Systeme zur Automatisierung von Instandhaltungsaufgaben erforscht werden. Eine bedarfsgerechte und flexibel gesteuerte Automatisierung von spezifischen Wartungs- und Instandsetzungsaufgaben könnte den Instandhalter entlasten, den Personalkosteneinsatz reduzieren und die Anlagenverfügbarkeit steigern. Schlussendlich gilt, dass die Instandhaltung ein wichtiger standortsichernder Faktor und Stellhebel für die Optimierung der Wertschöpfung und damit für die Zukunftsfähigkeit des Standorts Deutschland ist [3]. Deshalb müssen neue automatisierungstechnische Lösungswege gefunden werden, diese Herausforderungen anzunehmen, und in der Produktionstechnik zu etablieren.

1.1 Ausgangssituation, Problemstellung und Zielsetzung

Zur Unterstützung, sowie zur zustandsorientierten Instandhaltung werden automatisierte Inspektionsmethoden in Form von Fehlerdiagnosesystemen eingesetzt. Diese unterstützen den Instandhalter und können eine zielgerichtete Vorgehensweise von Instandhaltungsaufgaben ermöglichen. Jedoch bestehen für die Durchführung von Wartungs- oder Instandsetzungsaufgaben bisher keine Systeme, die eine Automatisierung in allgemeiner Form zulassen.

Eine Automatisierung dieser Aufgaben erfordert eine hohe Flexibilität und Autonomie des technischen Systems. Dabei muss dieses, ausgehend von einer vorausgegangenen Diagnose, eine aufgetragene Wartungs- oder Instandsetzungsaufgabe selbstständig ausführen können. Diese Arbeit untersucht wie sich Wartungs- und Instandsetzungsaufgaben automatisieren lassen und erforscht ein ganzheitliches Konzept mit den zugehörigen Methoden, um diese Problemstellung zu lösen. Dabei fokussiert sich der in dieser Arbeit gerichtete Blick auf die nachfolgend definierte Forschungsfrage:

„Wie kann auf Basis einer Fehlerdiagnose automatisiert eine Wartungs- oder Instandsetzungsoperation in einer Produktionsanlage durchgeführt werden?“.

Für die Beantwortung dieser Frage sollen Manipulationsfähigkeiten für Robotersysteme erforscht und entwickelt werden, die eine Automatisierung dieser Aufgaben erlauben. Dies erfordert die Bestimmung sowie die Ausführung eines Aufgabenplans (Handlungsplanung), welcher die Manipulation der einzelnen instandzuhaltenden Komponenten beschreibt. Die grundlegende Problemstellung besteht sowohl aufgrund den hohen Anforderungen an eine zeiteffiziente Bestimmung des Aufgabenplans, als auch an erforderliche Kompensationsmechanismen zur Behandlung von Umweltunsicherheiten. Die Ausführung der einzelnen Manipulationen muss reaktiv gestaltet sein, sodass auch bei nicht genau bekannten Umweltparametern eine zuverlässige Durchführung ermöglicht werden kann.

Zielsetzung dieser Arbeit ist die Entwicklung und Bereitstellung allgemeingültiger Methoden, welche es einem Robotersystem erlaubt, autonom Manipulationen zu planen und auszuführen, um eine vorgegebene Wartungs- oder Instandsetzungsaufgabe zu automatisieren. Ein Robotersystem soll dabei als autonom bezeichnet werden, wenn dieses die nachfolgende Definition nach [12] erfüllt:

„Autonomie ist die Fähigkeit eines Systems, in einer realen Umgebung, ohne externe Steuereingaben, selbständig, über längere Zeit, zu operieren.”

Dabei steht nicht das Ziel einer Vollautomatisierung im Vordergrund, sondern eine Lösung die das Instandhaltungspersonal unterstützt, indem manipulativ einfachere Aufgaben bedarfsgerecht automatisiert werden. Zur Erreichung dieser Ziele müssen flexible Planungs-, Steuerungs- und Regelungsstrategien erforscht werden, die Änderungen zur Ausführungszeit ermöglichen und Manipulationsaufgaben in teilweise a priori unbekannten Szenarien erlauben. Zur Koordination dieser Dienste soll eine modulare Steuerungsarchitektur bereitgestellt werden, welche unabhängig von der verwendeten Roboterplattform ist und sich einfach um neue Funktionalitäten erweitern lässt, sodass auch zukünftige Forschungsarbeiten einfach integrierbar sind.

1.2 Wissenschaftlicher Beitrag

Der wesentliche wissenschaftliche Beitrag besteht in der erstmalig allgemein analysierten, erforschten und umgesetzten Methodik an Manipulationsfähigkeiten, welches einem Robotersystem erlaubt, autonom Instandhaltungsaufgaben in produktionstechnischen Anlagen auszuführen. Die entwickelten Verfahren dieser Arbeit sind zu Teilen in wissenschaftlichen Beiträgen bereits vorab veröffentlich worden.

Zunächst werden hierzu die Potentiale und Möglichkeiten untersucht, welche bereits in der Vorarbeit [13] veröffentlicht wurden. Zur Bereitstellung und Koordination von Manipulationsfähigkeiten wurde eine geeignete Robotersteuerungsarchitektur entwickelt. Diese lässt die einzelnen Algorithmen zur Planung, Steuerung und Regelung, sowie zusätzlich nötige Sensorik und Roboterwerkzeuge in einfache Weise modular integrieren. Vorarbeiten zum Design der Steuerungsarchitektur sind in der Arbeit [14] dargestellt. Ähnliche Konzepte wurden im steuerungstechnischen Kontext für weitere Applikationsfelder in den Arbeiten [15], [16] eingesetzt.

Des Weiteren wurden neue Methoden zur Planung, Steuerung und Regelung von Robotermanipulationen entwickelt und experimentell validiert, die den Stand der Wissenschaft und Technik erweitern und auf die expliziten Anforderungen zur Automatisierung von Instandhaltungsaufgaben angepasst sind.

Für die Aufgabenplanung wird ein Algorithmus entwickelt, welcher besonders berechnungseffizient die Bestimmung vorhandener Demontageräume erlaubt. Auf Basis einer symbolischen Beschreibung lassen sich folgend Roboteraktionen planen. Die Verfahren sind in den Arbeiten [17], [18] veröffentlicht. Zur Kompensation von Umweltunsicherheiten werden binäre Bayes-Filter eingesetzt, die vorhandene CAD-Information mit der von der Objekterkennung bereitgestellten Information fusioniert. Das Objekterkennungssystem, welches insbesondere für den Anwendungsfall der roboterbasierten Instandhaltung angepasst ist, wird in der Vorarbeit [19] beschrieben. Zur Planung geeigneter Posen für die Bildverarbeitung wird ein neues aufgabenorientiertes Verfahren (Aufgabenexploration) in [20] vorgeschlagen. Dieses ermöglicht die dynamische Gewichtung der visuellen Teilaufgaben zur Kartenexploration und Objekterkennung in einer geeigneten Sensorpose. Dies erlaubt weitergehend eine geeignete Sequenzierung der geplanten Roboteraktionen auf Basis der Konzepte der Sichtbarkeit und Zugänglichkeit. Zur Verbesserung heutiger globaler Bahnplanungsverfahren wird eine Schrittweitensteuerung eingeführt, die einen erheblichen Vorteil hinsichtlich Berechnungszeit und Erfolgsrate in der Lösungsfindung mit sich bringt. Das Verfahren ist in [18], [21] veröffentlicht.

Die Umsetzung der Planungsergebnisse in ein ausführbares Anwenderprogramm erfolgt auf Basis einer geeigneten Dekompositionsvorschrift. Diese wird in der Arbeit [22] vorgestellt. Ebenso werden in dieser Experimente der Roboterregelung und der Aufgabenbeispiele vorgestellt.

1.3 Aufbau der Arbeit

Die vorliegende Arbeit gliedert sich in sieben Kapitel. Die einzelnen Kapitel behandeln dabei folgende Inhalte:

Kapitel 1. Beschreibt die Motivation Instandhaltungsaufgaben zu automatisieren und die hieraus resultierende Problemstellung, die damit verbunden ist.

Kapitel 2. Stellt den Stand der Forschung und Technik für die vorliegende Arbeit dar. Es werden einführend die wichtigsten Begrifflichkeiten und Zusammenhänge eingeführt. Weitergehend werden bestehende Automatisierungslösungen für die Instandhaltung sowie für aufgabenverwandte Domänen untersucht. Die Defizite heutiger Verfahren werden schlussendlich eindeutig identifiziert.

Kapitel 3. Die Analyse der Aufgaben und Anforderungen zeigt detailliert die zu lösenden Probleme auf, die für eine Automatisierung von Wartungs- und Instandsetzungsaufgaben notwendig sind.

Kapitel 4. Entwickelt ein Konzept zur zeiteffizienten Generierung von symbolischen Aufgabenplänen für Robotermanipulationen unter Umweltunsicherheiten. Die Planung der zugehörigen Bewegungsbahn erfolgt über ein neuartiges Vorverarbeitungsverfahren.

Kapitel 5. Es wird ein Verfahren vorgeschlagen, welches den symbolischen Aufgabenplan in ausführbare Steuerungselemente übersetzt. Diese werden dann zur Laufzeit durch eine schaltende Regelung ausgeführt.

Kapitel 6. Zeigt den experimentellen Aufbau mit zugehöriger Steuerungsarchitektur auf. Das Gesamtkonzept wird anhand praxisnaher Aufgaben gesamtheitlich validiert.

Kapitel 7. Fasst die Arbeit zusammen und gibt einen Ausblick auf mögliche zukünftige Forschungsarbeiten.

Abbildung 1-2 zeigt den Aufbau des neuen wissenschaftlichen Beitrags der Arbeit anhand des Informationsflusses. Ausgehend von einem Wartungs- oder Instandsetzungsauftrag auf Basis einer Fehlerdiagnose (angenommene Voraussetzung), bis zur automatisierten Wiederherstellung des Sollzustands, von der Planung (Kapitel 4) zur Steuerung und Regelung (Kapitel 5) bis hin zum experimentellen Teil, der Ausführung (Kapitel 6).

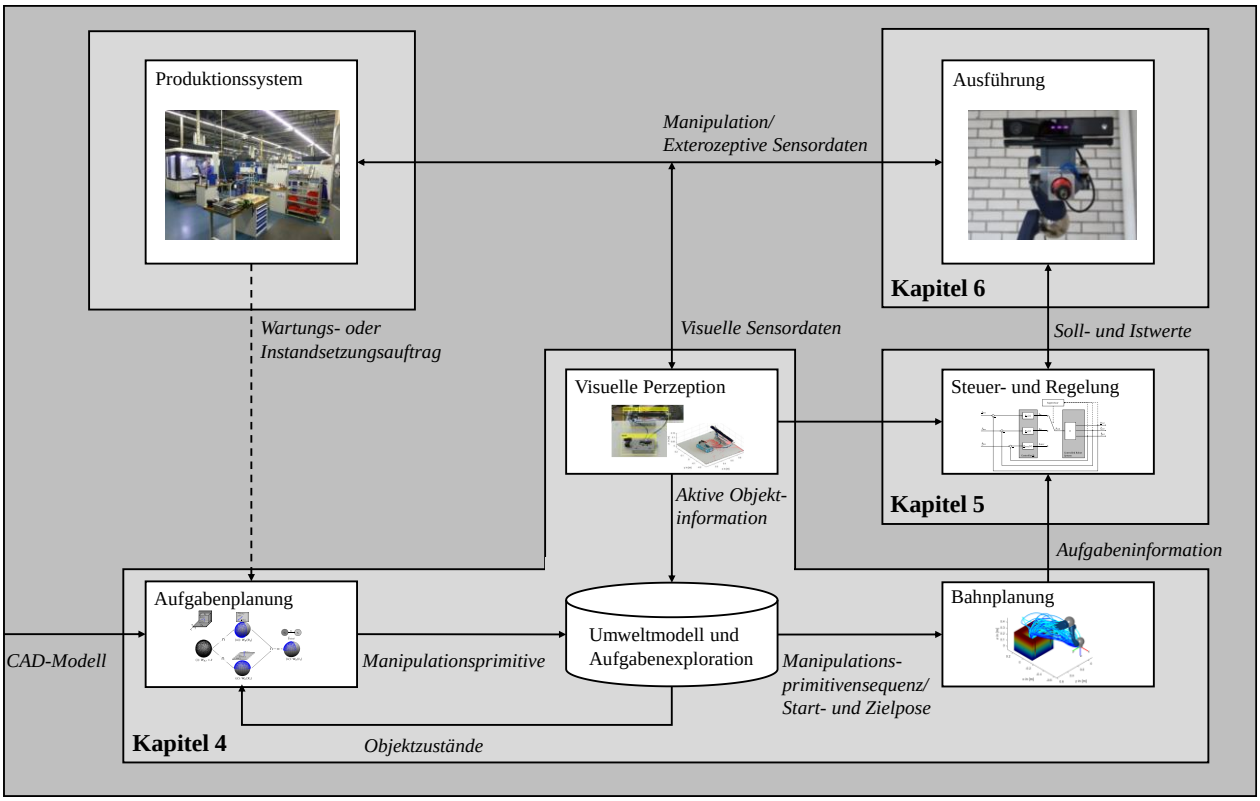


Abbildung 1-2: Aufbau und Forschungsbeitrag der Arbeit

2 Stand der Forschung und Technik

„Je reicher man an Urteilen ist, desto ärmer wird man an Vorurteilen.“
Henry Miller

Dieses Kapitel beschreibt den Stand der Forschung und Technik zur Automatisierung von Instandhaltungsaufgaben. Einführend werden in 2.1 sowohl die nötigen Begrifflichkeiten, als auch die Relevanz der Instandhaltung, besonders für die Produktionstechnik, dargestellt. Aufbauend wird in 2.2 die Automatisierung der Instandhaltung gemäß dem Ursache-Wirkungsprinzip eingeführt. Es werden Verfahren beschrieben, welche die automatisierte Fehlerdiagnose in Maschinen und Anlagen erlauben. Eine Bewertung der Verfahren zeigt, dass diese aus dem Blickwinkel der Forschung bereits weit fortgeschritten sind. Darauf aufbauend wird dargestellt, welche Methoden und Systeme bereits existieren, die erkannten Beeinträchtigungen automatisiert zu beheben. Abschließend werden in 2.3 Methoden der Autonomiebildung für technische Systeme betrachtet und hinsichtlich ihrer Übertragbarkeit auf die Problemstellung analysiert.

2.1 Instandhaltung in der Produktionstechnik

In heutigen Produktionssystemen spielt die Anlagenverfügbarkeit, vor allem im Hinblick auf ökonomische Ziele, eine entscheidende Rolle, wozu die Instandhaltung einen richtungsweisenden Beitrag leistet. Diese lässt sich gemäß DIN 31051 [23] in die folgenden Klassen gliedern:

- **Wartung:** Maßnahme zur Verzögerung des Abbaus des vorhandenen Abnutzungsvorrats.
- **Inspektion:** Maßnahmen zur Feststellung und Beurteilung des Istzustandes einer Einheit einschließlich der Bestimmung der Ursachen der Abnutzung und dem Ableiten der notwendigen Konsequenz für eine künftige Nutzung.
- **Instandsetzung:** Physische Maßnahme, die ausgeführt wird, um die Funktion einer fehlerhaften Einheit wiederherzustellen.
- **Verbesserung:** Kombination aller technischen und administrativen Maßnahmen sowie Maßnahmen des Managements zur Steigerung der Zuverlässigkeit und/oder Instandhaltbarkeit und/oder Sicherheit einer Einheit, ohne ihre ursprüngliche Funktion zu ändern.

Bei geeigneter Einhaltung und Auslegung der Mechanismen kann ein sicherheitskonformer Betrieb sowie eine minimale Stör- und Ausfallwahrscheinlichkeit erreicht werden. Um diese Ziele und die wachsenden Rahmenbedingungen bezüglich Produktqualität zu erreichen, ist die Instandhaltung einem stetigen Wandlungsprozess ausgesetzt. Zu Beginn der Entwicklung bis zum Jahre 1951, wurden Instandhaltungsmaßnahmen erst beim Ausfall eines Teilsystems oder der gesamten Anlage unternommen, bevor weitergehend erste präventive, bis hin zu zuverlässigkeitsorientierten Ansätzen, entstanden [24]. Die Wandlung kann gemäß Abbildung 2-1 aufgeteilt werden. Dieser Umstrukturierungsprozess korreliert mit der Erkenntnis, dass ein Wandel der Instandhaltung von einem reinen Kostenverursacher zu einem unternehmensübergreifenden Geschäftsprozess zu verzeichnen ist, der die Wertschöpfung aktiv unterstützt [25].

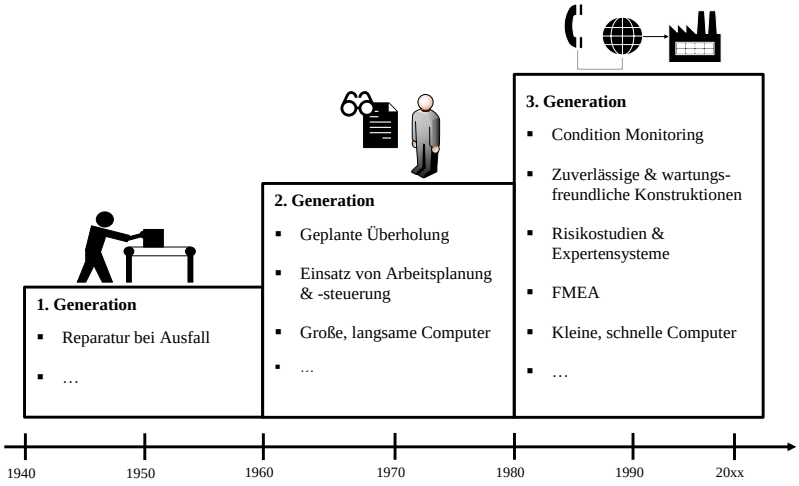


Abbildung 2-1: Wandel des Instandhaltungsprozess (Modifiziert nach [26], [27])

Das Spektrum an steigenden Anforderungen führt dazu, dass Aufgaben und Abläufe der Instandhaltung immer weiter optimiert werden, sodass Maschinen und Anlagen möglichst ausfallarm und effizient produzieren können. Resultierend aus dem technischen Fortschritt haben sich im Laufe der Zeit die grundlegenden Instandhaltungsstrategien von der reaktiven Instandhaltung (1. Generation) zunehmend mehr in die Richtung der präventiven Instandhaltung (2. und 3. Generation) verändert. Grundsätzlich lassen sich Instandhaltungsstrategien in die nachfolgenden Gruppen untergliedern, welche aus [28], [25] zusammengefasst sind.

- **Reaktive Instandhaltung:** Unter reaktiven Instandhaltungsmaßnahmen versteht man die Reparatur oder den Austausch einer Teilkomponente, sobald ein Fehler oder ein Ausfall aufgetreten ist. Der Vorteil, dass die maximale Lebenszeit der betroffenen Teilkomponente ausgenutzt wird, geht zu Kosten eines oftmals unvorhersehbaren Ausfalls der Gesamtanlage. Dies hat zur Folge, dass im Schadensfall die Instandsetzung unter großem Zeitdruck erfolgt und die erforderlichen Instandhaltungsressourcen wie Personal, Ersatzteile, Werkzeuge, Ausrüstungen und Hilfsmittel, oftmals nicht sofort zur Verfügung stehen oder für den Schadensfall extra vorgehalten werden müssen [25], [29].
- **Periodisch vorbeugende Instandhaltung:** Die einzelnen Teilkomponenten eines Systems werden nach einer definierten Zeit ausgetauscht. Vorteilig hierbei ist, dass der Ausfall des Gesamtsystems weitgehend minimiert wird, da bei geeigneter Wahl der Austauschzeit der einzelnen Teilkomponenten die Funktionsfähigkeit gewährleistet ist. Der Austausch kann während Maschinenstillstandzeiten erfolgen und hierdurch weitgehend geplant und optimiert werden. Nachteilig ist jedoch, dass geeignete Lebensdauerzeiten vorab ermittelt werden müssen.
- **Zustandsabhängige Instandhaltung:** Neben dem zeitlichen Einfluss werden für diese Methode die einzelnen Maschinenzustände als Eingriffskriterium verwendet. Die Zustandsüberwachung erfolgt mithilfe von Diagnosemethoden, welche entweder automatisiert oder durch menschliche Unterstützung erfolgen. Dies ermöglicht die Lebensdauer einzelner Komponenten geeignet auszunutzen und die weitgehend frühzeitige Detektion von Ausfällen. Der Nachteil besteht hauptsächlich in den technischen und finanziellen Aufwendungen, welche eingebracht werden müssen.

- **Vorrausschauende Instandhaltung:** Es wird eine Fehlerprädiktion verwendet, wodurch ein noch gezielterer Austausch stattfinden kann. Beispielsweise wird in [30] die Erstellung eines Verschleißmodells vorgeschlagen, auf dessen Datengrundlage sich unter Beachtung möglicher Fehlerursachen, die mittlere statistische Lebensdauer der Bauteile prädizieren lässt.

Eine optimale Instandhaltungsstrategie (Minimierung von Wartungs- und Ausfallzeiten) kann Elemente aus allen Strategien enthalten, muss aber an die Rahmenbedingungen der Unternehmensorganisation angepasst werden [6], [27]. Durch den Einsatz von automatisierungstechnischen Lösungen kann jedoch eine Zunahme an präventiven Instandhaltungsmaßnahmen bei verringertem Personaleinsatz erreicht werden, wodurch gleichzeitig eine Steigerung der Verfügbarkeit ermöglicht wird. Jedoch wird der Einsatz automatisierungstechnischer Lösungen auch durch die Instandhaltbarkeit [31] einer Maschine beeinflusst, welche unter anderem die Zugänglichkeit, Prüfbarkeit, Austauschbarkeit sowie die Standardisierung umfasst [32]. Diese Punkte werden maßgeblich durch die Konstruktion der Maschine bestimmt. Vor allem im Hinblick auf eine Automatisierung von Wartungs- und Instandsetzungsaufgaben stellen diese Punkte eine wesentliche Voraussetzung zur Etablierung dieser Lösungen dar.

2.2 Automatisierung von Instandhaltungsaufgaben

Die zunehmende Komplexität heutiger Produktionseinrichtungen erfordert vom menschlichen Werker immer mehr spezifisches Fachwissen zur Störungsvermeidung, -erkennung und -behebung. Im Zuge verschiedener Untersuchungen von Betriebsverhalten komplexer Produktionssysteme wurde diagnostiziert, dass deren Verfügbarkeit mit steigendem Komplexitätsgrad stark abfällt [33], [3]. Analysen haben ergeben, dass ein Großteil der Instandsetzungszeit auf die Eintrittszeit entfällt (Warte- und Wegzeiten bis Instandsetzungspersonal eintrifft) [34], [35]. Dies ist häufig auf unzureichende Hilfsmittel und festgelegte Abläufe zur Koordination des Instandhaltungspersonals, insbesondere bei dezentralen Instandhaltungsstrukturen, zurückzuführen [36]. Folglich werden zur Unterstützung des Werkers automatisierungstechnische Lösungen erforscht, entwickelt und untersucht, sodass die Anlagenverfügbarkeit verbessert werden kann. Weitergehend sollen die notwendigen Begrifflichkeiten eingeführt und bereits existierende Verfahren zur automatisierten Inspektion sowie zur Wartung und Instandsetzung, dargestellt werden.

Automatisierte Inspektion. Die automatisierte Inspektion eines Systems umfasst die Zustandserfassung (Messung oder Beobachtung) der relevanten Maschinen- oder Prozesszustände und die Ableitung von Anomalien gegenüber dem gewünschten Sollzustand (Fehlerdiagnose). Dabei lassen sich zur automatisierten Inspektion folgenden Instanzen nutzen, welche nach [37] aufgeführt sind:

- **Messmittel und Prüfverfahren:** Es wird eine mobile Inspektion mit externen Messmitteln und Prüfverfahren durchgeführt.
- **Prüfwerkstücke:** Die Information des Maschinenzustandes erfolgt durch messen und prüfen von Werkstücken, welche dem Prozess entnommen werden.
- **Zusätzliche Sensorik:** Es werden festinstallierte, zusätzliche Sensoren in der Anlage für die relevanten physikalischen Größen integriert.

- **Antriebsbasierte Diagnose:** Es werden antriebsbasiert, gemessene und bereits vorhandene physikalische Größen verwendet.

Die Zustandserfassung versteht sich als die Gewinnung aller nötigen Größen, welche zur nachfolgenden Fehlerdiagnose relevant sind. Bezüglich domänenabhängiger Messgrößen und Kennzahlen zur Zustandserfassung sei auf [38] verwiesen. Industriell wird die Fehlerdiagnose für gewöhnlich als Condition Monitoring System (CMS) bezeichnet. Zur automatisierten Diagnose von Fehlern hat sich mittlerweile eine beachtliche Anzahl an Methoden und Verfahren entwickelt. Diese gliedern sich nach [39] in die Schritte:

1) **Fehlerdetektion:** Dient der Erkennung von Abweichungen bestimmter Prozessgrößen vom nominalen Betriebszustand.
2) **Fehlerisolation:** Dient der Klassifikation des aufgetretenen Fehlers.
3) **Fehleridentifikation:** Die Größe und das Ausmaß des aufgetretenen Fehlers werden bestimmt.

Die Fehlerdetektion, als erster Schritt der Diagnose teilt sich nach [40], in die drei Gruppen:

- **Grenzwertüberwachung und Plausibilitätsprüfung:** Dies stellt die einfachste Möglichkeit zur Erkennung von Abweichungen gegenüber dem Nominalverhalten einer Anlage dar. Es werden dabei die Signalamplituden definierter Sensorstellen gemessen. Bei Überschreitung einer festgelegten Intervallgrenze des Sensorwertes wird dies als ein Abweichen vom Sollverhalten deklariert. Beispielhaft hierfür ist die Temperaturüberwachung in elektrischen Servoantrieben oder die Drucküberwachung in hydraulischen Anlagen. Ebenso werden oftmals Sensorsignalverläufe zeitlich aufgezeichnet um anhand der Änderungen schon frühzeitig Aufschluss über den zukünftigen Trend eines Maschinenzustands zu erhalten. Anwendung findet dieses Verfahren etwa in der Überwachung von Maschinenvibrationen oder bei Verschleißmessungen [41].
- **Signalmodellbasierte Methoden:** Diese Methoden bedienen sich häufig der Korrelations- oder Spektralanalyse. Hierbei müssen nicht zwingend zusätzliche Sensoren in die Maschine integriert werden. In [42], [43] werden Verfahren entwickelt, die antriebsnahe Signale in einer Werkzeugmaschine (Lage, Beschleunigung bzw. Strom) nutzen, um Wälzlagerschäden auf Basis einer Frequenzganganalyse zu detektieren.
- **Prozessmodellbasierte Methoden:** Modellbasierte Fehlerdetektionsverfahren stellen die leistungsfähigsten Verfahren zur Fehlerdiagnose dar [44]. Hierzu kommen oftmals Paritätsgleichungen oder Parameterschätzer sowie Zustandsbeobachter und bei nicht vernachlässigbarem Prozess- und/oder Sensorrauschen Kalman-Filter zum Einsatz. Die Symptome sind dann Änderungen von Parametern, Residuen zwischen Prozessmodell und Prozess-Ausgangsgrößen oder, Änderungen interner Zustandsgrößen [45]. Dabei eignen sich Paritätsgleichungen und Zustandsbeobachter besonders gut für additive Fehler unter Echtzeitanforderungen, während Parameterschätzer vor allem für multiplikative Fehler eingesetzt werden [46], [45]. Der Einsatz wurde bereits erfolgreich an spezifischen Baugruppen, wie etwa elektrischen und hydraulischen Antrieben und Ventilen sowie bei Werkzeugmaschinen eingesetzt, siehe hierzu etwa [41].

Der zweite und dritte Schritt im Diagnoseprozess, die Fehlerisolation und -identifikation, erfolgt für gewöhnlich auf der Grundlage von Klassifikations- oder Inferenzmethoden. Der breite Einsatz

von Methoden aus der künstlichen Intelligenz ist vor allem auf die verbesserte Leistungsfähigkeit gegenüber konventionellen Verfahren zurückzuführen, da die Möglichkeit zur Modellierung von Unsicherheiten durch Lern- und Fuzzyverfahren gegeben ist. Jedoch stellt es sich in der Praxis oft als schwierig heraus an geeignetes Expertenwissen und Trainingsdaten zu gelangen, welche für diese Methoden benötigt werden [47]. Es gibt hierzu zahlreiche Anwendungsbeispiele, wie die Verwendung von Self-Organizing Maps (Kohonennetze) zur Online-Diagnose in Automatisierungssystemen [48] oder der Rückschluss verschiedener Fehlerarten auf Basis von Getriebeschwingungsmessungen, welche mit Hidden Markov Modellen auch bei starkem Messrauschen zugeordnet werden können [49].

Zusammenfassend stehen für Diagnoseaufgaben bereits leistungsfähige Methoden zur Verfügung. Tabelle 2-1 fasst die einzelnen Ergebnisse nochmals zusammen. Unter der Kategorie „Mobil“ wird die extrinsische Datenerfassung mittels Sensoren verstanden.

Tabelle 2-1: Bewertung automatisierter Inspektionsmethoden

Eigenschaften		**Inspektionsmethode**			
		Grenzwert/ Trendüberwachung	Signalmodell	Prozessmodell	Mobil
Entwurf/ Integration	Keine zusätzliche Sensorik	○	◑	●	○
	Niedrige Komplexität	●	◑	○	◑
	Expertenwissen	○	◑	●	●
	Modularität	○	◑	◑	◑
	Invasiv	●	○	○	◑
Betriebsmerkmale	Gute Klassifikation	○	◑	●	◑
	Lange Entdeckungszeit	●	◑	○	◑
	Erkennen kleiner Änderungen	○	◑	●	●
	Echtzeitfähig	●	●	●	○
	Online	●	●	●	○
	Störgrößen-anfällig	○	◑	●	○
	Mehrere Signale	○	◑	●	○

○ Nein ◑ Neutral ● Ja

Um gute Diagnoseergebnisse zu erzielen, müssen die positiven Eigenschaften der einzelnen Methoden jedoch geeignet verknüpft und kombiniert werden [44]. Auch wurden die einzelnen Verfahren meistens auf Komponentenbasis entwickelt. Problematisch erscheint dies vor allem in komplexen Produktionssystemen, bei denen eine Vielzahl an Komponenten miteinander vernetzt ist und Wechselwirkungen betrachtet werden müssen. Industriell wird derzeit noch weitgehend auf fortgeschrittene Methoden der automatisierten Fehlerdiagnose verzichtet und einfache Grenzwert-

oder Trendanalysen durchgeführt, vergleiche hierzu beispielhaft [46]. Dies liegt vor allem an der zunehmenden Komplexität und Heterogenität der Systeme sowie das Unsicherheiten, Störungen und Rauschen, bereits im Entwurf berücksichtigt werden müssen, da diese sich negativ auf die Diagnoseergebnisse auswirken [50].

Automatisierte Wartung und Instandsetzung. Nach der Erkennung und Maßnahmenplanung zur Behebung eines aufgetretenen Fehlers oder für Wartungsaufgaben muss der nominale Zielzustand automatisiert rückgeführt werden. Hierzu existieren zwei differente Mechanismen, welche bei unterschiedlichem Fehlerausmaß eingesetzt werden können. Bei kleinen Fehlern kann durch Adaption oder Rekonfiguration der Steuerung/Regelung wieder das Nominalverhalten der Anlage erzwungen werden. Große Fehler, wie etwa der Ausfall systemfunktionsrelevanter Einheiten oder Wartungsaufgaben, erfordern jedoch ein zusätzliches externes System zur Herstellung des Sollzustandes. Diese Verfahren sollen folgend, bezüglich ihrer Eignung für die Produktionstechnik, diskutiert werden.

Adaption und Rekonfiguration. Oftmals besteht das Ziel, eine Maschine oder eine Anlage, bei Veränderung des dynamischen Systemverhaltens, bis zu einer planmäßigen Pause, unter der Anforderung von gleichbleibender Produktqualität, weiterbetreiben zu können. Hierfür bietet sich die Systematik der fehlertoleranten Steuerung/Regelung *(engl. „Fault-Tolerant Control“)* (FTC) an. Unter FTC versteht man die Adaption oder Rekonfiguration der Steuerung/Regelung, damit das geforderte dynamische Systemverhalten beibehalten werden kann. Man unterscheidet hierbei zwischen zwei grundlegenden Systemklassen:

- **Passive FTC-Systeme:** Es wird dasselbe Regel- und Steuergesetz für das nominelle und fehlerbehaftete System verwendet [51]. Passive FTC-Systeme eignen sich nur bei geringen Änderungen und es muss zwischen nomineller Systemdynamik und Fehlerdynamik in der Reglersynthese ein Kompromiss gefunden werden. Beispielhaft sind hierbei der Einsatz von robusten oder zuverlässigen Reglerstrategien [52].
- **Aktive FTC-Systeme:** Auf Grundlage einer Fehlerdiagnose wird ein Rekonfigurations- oder Adaptionsgesetz gefunden, welches das fehlerbehaftete System in das nominelle System rücktransformiert. Aktive FTC-Systeme bedienen sich den Methoden der adaptiven Regelungstechnik, siehe [53], [54], oder der Rekonfiguration mit virtuellen Sensoren oder Aktoren [55], [56], [57]. Virtuelle Sensoren und Aktoren können auch bei strukturellen Systemänderungen eingesetzt werden, in denen der adaptive Einsatz versagt. Die Grundidee besteht darin, dass bei einem Sensorausfall der gemessene Sensorwert durch einen reduzierten Beobachter ersetzt werden kann. Beim virtuellen Aktor wird davon ausgegangen, dass, bei Ausfall der physikalischen Stelleinheit, diese durch eine Linearkombination verbliebener Aktoren kompensiert werden kann. Das Funktionsprinzip des Verfahrens wird in [58], anhand eines verfahrenstechnischen Prozesses (Ventilausfall, Pumpenausfall), aufgezeigt.

Derartige Methoden sind im Allgemeinen für Produktionssysteme nicht geeignet, da der Entwurfsaufwand, als auch die Komplexität den Nutzen nicht aufwenden würden. Entsprechende Systeme rechtfertigen sich vor allem in sicherheitskritischen Infrastrukturen, in welchen eine hohe Fehlertoleranz vorhanden sein muss.

Externe Systeme. Bei Wartungs- oder Instandsetzungsarbeiten muss zwingend eine physische Interaktion mit der instandzuhaltenden Einheit stattfinden. Hierzu werden, aufgrund ihrer Flexibilität, bevorzugt mobile Robotersysteme eingesetzt, welche entweder teleoperierend, semi-autonom oder autonom operieren.

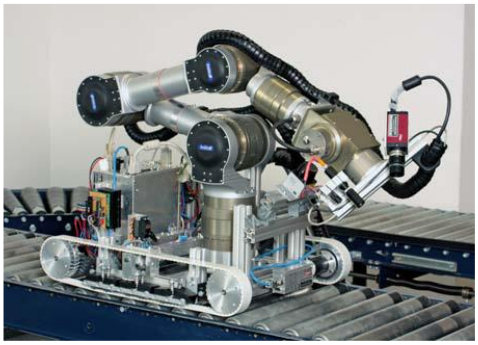
Bisher wurden diese Systeme häufig, in für Menschen unzugänglichen oder risikobehafteten Umgebungen, eingesetzt. Dies zeigen vor allem die zahlreichen Applikationsgebiete, wie etwa Energieübertragungsnetze [59], [60], [61], [62], [63] siehe Abbildung 2-2 (a), Abwassersysteme [64], [65], thermische Kessel [66], [67] und Brücken [68]. Diese Robotersysteme können dabei hauptsächlich Aufgaben wahrnehmen, welche von Reinigungs- und Inspektionsaufgaben bis hin zu einfachen Reparaturen reichen. Beispielhaft hierfür sind die Arbeiten zur Kanalrobotik, in der das Robotersystem pH-Werte, elektrische Leitfähigkeit, Sauerstoffgehalt und die Temperatur misst [64]. In der automatisierten Instandhaltung von Energieübertragungssystemen können die Stromleitungen mittels Wirbelstromanalyse auf Fehler untersucht [59] und bei Bedarf mit einer Spezialklammer repariert werden [61].

Prinzipiell sind diese Robotersysteme, sowohl konstruktiv als auch von ihren Fähigkeiten immer an einen speziellen Anwendungsfall angepasst. Haupteinsatzgebiete sind statische Umgebungen mit fest definierten Handlungs- und Aktionsmustern, welche nur eine begrenzte Autonomie erfordern. Es lassen sich damit nur spezielle Aufgaben lösen. Ein ganzheitlicher Ansatz zur Automatisierung von Instandhaltungsaufgaben wird nicht aufgezeigt. Des Weiteren existieren auch Robotersysteme, welche generalisierte Ansätze verfolgen und nachfolgend diskutiert werden sollen.

Zur Instandhaltung von Materialflusssystemen wird in den Arbeiten [69], [70], [71] ein mobiles Robotersystem entwickelt, welches primär Inspektionsaufgaben ausführt, siehe Abbildung 2-2 (b). Zur Navigation des mobilen Robotersystems wird das topologische Modell der Anlage verwendet, welches anhand von Planungs- und Konstruktionsunterlagen abgeleitet werden kann [69]. Hierdurch sind prinzipiell mögliche Navigationswege sowie zu wartende Strukturen bereits im Vorfeld bekannt. Zusätzlich wird eine sensorbasierte Fahrbahnerkennung verwendet. Die Erkennung eines spezifischen, instandzuhaltenden Objekts beruht auf dem Einsatz eines referenzmarkenbasierten Erkennungssystems. Das Robotersystem führt dabei vornehmlich Inspektionsaufgaben wie Schwingungsmessungen, Temperaturmessungen, oder optische Prüfungen aus. Das Steuerungsprogramm des Manipulators wird bei Verfügbarkeit von CAD-Daten mittels Offline-Programmierung oder per Teach-In generiert [70]. Dieses Vorgehen hat den gravierenden Nachteil, dass zusätzlicher manueller Aufwand im Einrichtebetrieb des Systems nötig ist. Auch wird keine Kollisionsvermeidung oder Ausweichstrategie bei Abweichung des festgelegten Steuerungsprogramms von der realen Umgebung betrachtet, was eine hohe Einschränkung darstellt.

Das mobile Robotersystem MIMROex [72], [73], [74] ist für Instandhaltungsaufgaben prozesstechnischer Anlagen ausgelegt. Es handelt sich hierbei um einen Knickarmroboter, welcher auf einer mobilen Plattform integriert ist, siehe Abbildung 2-2 (c). Das Robotersystem wird zum Ablesen von Sensorwerten eingesetzt. Weitere beispielhafte Aufgaben sind das Nachstellen von Stellrädern oder das Auswechseln von Feuer- und Gassensoren [72]. Der Roboter wird hierbei von einem Bediener teleoperiert, unterstützt den menschlichen Operator allerdings bei Navigations- und Manipulationsaufgaben, welche teilautonom durchgeführt werden können. Eine allgemeine Lösung zur Automatisierung von Wartungs- und Instandsetzungsaufgaben wird nicht entwickelt.

(a)



(b)

(c)

Abbildung 2-2: Robotersysteme für die automatisierte Instandhaltung: Expliner [60] (a); Roboter für Materialflussanlagen (b) [71]; Fraunhofer MIMROex (c) [73]; © IEEE 2008, 2010, 2009

Neben der Automatisierung von Instandhaltungsaufgaben wird auch die Unterstützung des Instandhalters durch kollaborative Roboterassistenten untersucht, wie das EU-Projekt „SecondHands- A robot assistant for industrial maintenance“ [75] zeigt. So wird in [76] eine Methode zur Arbeitsraumanalyse für einen geeigneten Mensch-Roboter-Interaktionsraum vorgeschlagen. In der Arbeit [77] wird ein Regler entwickelt, der eine Werkzeugübergabe zwischen Mensch und Roboter erlaubt. Diese Ansätze sind in Bezug auf das Ziel dieser Arbeit als komplementär zu verstehen und sollen deswegen nicht weiter betrachtet werden.

Tabelle 2-2 bewertet zusammenfassend die Fähigkeiten heutiger Instandhaltungsroboter. Es zeigt sich, dass diese vor allem im Bereich der Inspektion bereits automatisierte Teillösungen bereitstellen.

Tabelle 2-2: Bewertung von Robotersystemen für die automatisierte Instandhaltung

Eigenschaften/ Fähigkeiten		**Robotersystem für**			
		Kanalsysteme	Energiesysteme	Prozesssysteme (MIMROex)	Materialfluss-systeme
Automatisierte Inspektion		●	●	◑	◑
Automatisierte Wartung	Reinigen	●	○	○	●
	Ergänzen	-	○	○	○
	Konservieren	-	○	○	○
	Auswechseln	-	○	◑	○
	Nachstellen	○	●	◑	○
	Schmieren	-	-	○	○
Automatisierte Instandsetzung	Komponenten-austausch	○	○	○	○
	Reparatur von Funktionseinheiten	○	○	○	○

○ Nein ◑ Neutral ● Ja - Nicht notwendig

Für den spezifischen Einsatz sind diese Systeme für die Inspektion bereits gut geeignet, jedoch stellt sich die Automatisierung dieser Aufgaben in komplexeren Umgebungen, wie an den Beispielen des Materialflussinstandhaltungsroboters und MIMROex aufgeführt, als nur bedingt gelöst heraus. Für die Bereiche Wartung und Instandsetzung existieren bisher nur unzureichende bis keine Strategien, die eine universelle Automatisierung erlauben. Es existieren zwar Teillösungen, jedoch sind diese äußerst spezifisch und lassen sich nicht auf die Allgemeinheit an Wartungs- und Instandsetzungsprozessen übertragen.

Zusammenfassend gilt, dass bereits Verfahren zur Automatisierung von Instandhaltungsaufgaben untersucht sind. Jedoch werden vor allem Wartungs- und Instandsetzungsaufgaben fast durchweg durch Instandhaltungspersonal gelöst. Besonders eignen sich Robotersysteme als Mittel zur Automatisierung von Instandhaltungsaufgaben, da diese prinzipiell universelle Aufgaben übernehmen können. Jedoch fehlen grundlegende Methoden, um ausreichend Autonomie für das Robotersystem bereitzustellen, sodass umfassende Instandhaltungsaufgaben in Produktionssystemen automatisiert werden können. Die bestehenden Lösungen sind zu spezifisch um diese auf komplexere Aufgaben zu adaptieren. Dies gilt vor allem für Methoden zur Manipulationsplanung und Ausführung von Wartungs- und Instandsetzungsarbeiten durch Robotersysteme. Hierzu müssen neue Methoden entwickelt werden, die eine allgemeine und ganzheitliche Planung von Handlungssequenzen für Roboteraufgaben zulassen. Nur hierdurch lässt sich eine effektive Automatisierung von Instandhaltungsaufgaben in der Zukunft erreichen. Neben der Automatisierung von Wartungs- und Instandsetzungsaufgaben mittels Robotersystemen, werden prinzipiell ähnliche Fähigkeiten in den Bereichen Montage und Demontage benötigt, wenn auch die Rahmenbedingungen variieren. Diese Beiträge sollen weitergehend auf ihre Eignung für die automatisierte Instandhaltung untersucht und dargestellt werden.

Automatische Durchführung von Demontageaufgaben mit Robotersystemen. Erste theoretische Grundlagen, Demontageaufgaben automatisiert mithilfe von Robotersystemen auszuführen, wurde anhand der „Bauklötzchenwelt“ beispielsweise bereits in [78] erforscht. Anwendungsbezogene Lösungen werden vor allem für Elektronikprodukte entwickelt, da eine Automatisierung dieser Aufgabe, aufgrund der immer weiter zunehmenden Zahl von End-of-Life-Produkten benötigt wird [79] und die Ressourcenrückgewinnung sich sowohl ökologisch als auch ökonomisch als rentabel erweist. Praktische Aufgabenstellungen sind die Demontage von Fotoapparaten [80], Lithium-Ionen-Batterien [81], Batterien aus Fernbedienungen oder Taschenrechnern [82], Waschmaschinen [83] bis hin zu LCD-Bildschirmen [84], [85]. Dabei liegt der Fokus der Arbeiten auf der organisatorischen und mechanischen Gestaltung von Demontageanlagen, welche in Abhängigkeit des Automatisierungsgrad als manuelle, hybride oder vollautomatisierte Systeme designt werden. Auch wird die Entwicklung von Methoden für eine erweiterte Autonomie von Robotersystemen, die flexible Lösungen für unterschiedliche Produkte zur Verfügung stellen sollen, untersucht.

Ersteres wird unter anderem im Rahmen des Sonderforschungsbereichs DFG SFB 281-„Demontagefabriken zur Rückgewinnung von Ressourcen in Produkt- und Materialkreisläufen“ aufgegriffen. Zur Beherrschung der hohen Variantenvielfalt an Demontageoperationen wird in [86] ein Baukastensystem für flexible Demontagewerkzeuge vorgeschlagen und entwickelt. Je nach benötigter Operation wird das geeignete Werkzeug konfiguriert, wobei der Fokus auf dem Handhaben und Lösen von Verbindungen liegt. Um Demontageanlagen möglichst flexibel zu

gestalten wird in [87] eine Systematik in Form eines morphologischen Kastens entwickelt, welche gemäß den verfahrensbezogenen Anforderungen die Planung einer Demontageanlage unterstützt. Damit innerhalb von Demontagesystemen konsistente Informationen, welche zur Demontageplanung benötigt werden, verfügbar sind, schlägt [88] ein Demontageinformationssystem vor, welches Information über das Demontageobjekt, den Demontageprozess, benötigte Betriebsmittel und die Betriebsdatenerfassung enthält. Diese sind als Datenbanken integriert und werden zum übergreifenden Austausch über den OPC-Standard bereitgestellt. Der steuerungstechnische Aufbau der Demontageanlage wird in [83] aufgezeigt. Um eine möglichst hohe Flexibilität für verschiedene Anwendungsfälle zu erreichen, werden modulare SPS-Funktionsbausteine eingesetzt und an einer realen Anlage experimentell überprüft, siehe Abbildung 2-3 (a). Die automatische Erzeugung der einzelnen Demontagesequenzen und die algorithmischen Problemstellungen werden in den Arbeiten nicht betrachtet.

In den Arbeiten [89], [90], [91], [92], [93], [94] werden sowohl die Planung als auch die Ausführung von Demontageaufgaben detailliert untersucht, vergleiche auch Abbildung 2-3 (b). Die Konzepte werden zur Planung am Beispiel der Dekomposition von Fernbedienungen [91] oder zur Planung mit anschließender Ausführung am Beispiel der Zerlegung von PCs [93], [94] in der Praxis experimentell validiert. In [93] wird ein Industrierobotersystem zur Demontage verwendet. Die Planung erfolgt auf einem relationalem Produktmodell, welches in einer Datenbank hinterlegt ist. Das System ist mit einem stationären Stereokamerasystem ausgestattet, das zur Objekterkennung und -lokalisierung der Komponenten verwendet wird. Ist das Objekt erkannt, wird das zugehörige Produktdatenmodell geladen und hieraus die Roboterbahnen für die einzelnen Demontageoperationen generiert. Die Demontage erfolgt dabei semi-automatisch. In [94], [95] wird das Prinzip auf zwei kooperierende Industrieroboter übertragen, wodurch sowohl die Demontagezeit reduziert als auch Aufgaben welche nur durch Zwei-Arm-Manipulatoren möglich sind, ausgeführt werden können.

Die Arbeiten [84], [85], [96], [97] beschreiben vom Autonomiegrad eines der am weitesten entwickelten Robotersysteme, wobei dessen Funktionalität am Beispiel der (semi-)zerstörende Demontage von LCD-Bildschirmen aufgezeigt wird, siehe Abbildung 2-3 (c). Das formale Framework basiert auf dem Situationskalkül [98]. Zur Reduktion von Unsicherheit wird ein Bildverarbeitungssystem eingesetzt und diese Information mit einer Wissensdatenbasis kombiniert,

(a) (b) (c)

Abbildung 2-3: Robotersysteme für die automatisierte Demontage: Demontage von Waschmaschinen (a) [83] © IEEE 2007; Demontage von PC (b) [95] © Elsevier 2006; Demontage von LCD-Bildschirm (c) [96] © Esmerald Insight 2013

welche unter anderem die Demontagesequenz enthält. So können prinzipiell auch unbekannte Komponenten demontiert und anschließend in die Wissensdatenbank eingelernt werden. Mittels einer Inferenzmaschine werden die einzelnen Roboteraufgaben generiert, wobei aufgrund der Anforderungen, keine rein zerstörungsfreie Demontage durchgeführt werden muss, und somit keine reaktive Ausführung erfolgt. Die Bestimmung einer Demontagesequenz wird nicht betrachtet.

Automatische Durchführung von Montageaufgaben mit Robotersystemen. Ein frühzeitig entwickeltes Robotersystem stellt KAMRO (Karlsruher Autonome Mobile Roboter) [99], [100], [101] dar, welches über zwei Robotermanipulatoren verfügt und mit verschiedenen Sensoren (Kraft-Momenten, CCD und Ultraschall) ausgerüstet ist, vergleiche Abbildung 2-4 (a). Die Beschreibung der Roboteraufgabe erfolgt dabei auf unterschiedlicher Weise, wie etwa Montagevorranggraphen oder der Vorgabe von Elementaroperationen (Pick, Place, …). Die Sensoreingaben dienen der Planüberwachung und ermöglichen es KAMRO den Plan zu korrigieren. Die Evaluation erfolgt am Beispiel des Cranfield-Benchmark.

Im Rahmen des DFG SFB 360- „Situierte Künstliche Kommunikatoren“, wurde ein Robotersystem erforscht, welches autonom die Montage von sogenannten Baufix-Elementen (Holzspielzeugklötzchen) zu kompletten Modellen, wie einem Flugzeug, zulässt [102], [103], [104], siehe Abbildung 2-4 (b). Das System reagiert dabei auf menschliche Spracheingaben und setzt die Anweisungen, mittels einer Zwei-Arm-Kinematik, um. Zur Objekterkennung und Lageerfassung werden externe Multi-Kamera-Systeme, sowie in den Manipulator integrierte Kamerasensoren für die visuelle Regelung verwendet. Ebenso sind Kraft-Momenten-Sensoren montiert, sodass prinzipiell komplexe Montageaufgaben lösbar sind.

Eine weitere Möglichkeit Montageprozesse mit Robotersystemen möglichst flexibel zu automatisieren, stellt die Nutzung der Informationen aus CAD-Modellen dar. Durch die Entwicklung einer CAx-Kette für Montageaufgaben werden ein zeitlich reduzierter Einrichtebetrieb sowie eine höhere Automatisierung der gesamten Produktionskette erzielt. Eines der ersten bekannten Systeme, welches sowohl die automatische Planung als auch die Ausführung von Montagesequenzen durch ein Robotersystem erlaubt, stellt Archimedes 2 dar [105]. Archimedes 2 bestimmt aus CAD-Daten mögliche Montagefolgen und setzt diese in ein Roboterprogramm um. Allerdings lassen sich keine Lage- und Bauteiltoleranzen miteinbeziehen und das System kann nur monotone Montagesequenzen bearbeiten [106]. Ein kontinuierlich weiterentwickeltes System ist ^{High}Lap *(engl. „High Level Assembly Planning“)*. Das entwickelte System kann auf Basis von CAD- und Nutzerinformationen automatisch sogenannte Aktionsprimitive (elementare Roboteranweisungen) erzeugen, sodass Planung und Ausführung kombiniert wird [107], [108]. Diese werden sensorgestützt überwacht und geregelt, wodurch eine höhere Robustheit des Montageprozesses erzielt werden kann. Dieser Ansatz wird in [106], [109] erweitert, womit eine Prozesskette vom CAD bis zu einem automatisch generierten Roboterprogramm auf Basis von Aktionsprimitiven entsteht. Das System wird anhand verschiedener Anwendungen, wie der Montage einer Steckdose auf einer Hutschiene (Planung und Ausführung) siehe Abbildung 2-4 (c), Automobilscheinwerfer (Planung) oder einer Bohrmaschine (Planung) praktisch evaluiert. In [110] wird zusätzlich eine Greifplanung integriert. Eine rein CAD-basierte Planung ist allerdings für die Instandhaltung nicht zielführend, da davon ausgegangen werden muss, dass CAD und reale Umgebung nicht zwangsläufig übereinstimmen.

(a) (b) (c)

Abbildung 2-4: Robotersysteme für die automatisierte Montage: Zweiarmiger und mobiles Robotersystem KAMRO [101] (a); Montage von Baufix-Teilen (DFG SFB 360) [104] (b); Montage von Steckdose auf Hutschiene (^{High}Lap) [109] © Springer 2010 (c)

In den beiden vorherigen Absätzen wurde gezeigt, dass für aufgabenverwandte Probleme durchgehende Methoden sowie Robotersysteme existieren. Jedoch bestehen Hemmnisse für die Instandhaltung in Bezug auf Planungszeiten, Umweltunsicherheiten und Autonomie, worauf im Detail noch eingegangen wird. In nachfolgendem Absatz sollen deswegen die wesentlichen Ansätze zur Autonomiebildung, innerhalb der Domäne der Robotik, diskutiert werden.

2.3 Systemfunktionen und Methoden zur Autonomiebildung in der Robotik

Folgendes Unterkapitel beschreibt die wesentlichen Systemfunktionen und eingesetzten Methoden autonomer Robotersysteme. Dabei werden im Speziellen bestehende Methoden, bezüglich Planungs-, Steuerungs-, und Regelungsverfahren diskutiert, welche erst die allgemeine Generierung von Robotermanipulationen erlauben, wie diese für die Ausführung von Wartungs- und Instandsetzungsaufgaben benötigt werden. Eine Diskussion der bereits bestehenden Arbeiten ermöglicht dahingehend eine ganzheitliche Analyse zum aktuellen Handlungsbedarf. Neben einer Einführung in die Modellierung von Robotersystemen in 2.3.1, werden die wesentlichen Methoden, welche zur Autonomiebildung benötigt werden, betrachtet, und für ihren Einsatz in der automatisierten Instandhaltung analysiert. Abbildung 2-5 zeigt allgemein das Zusammenspiel der einzelnen Systemfunktionen, wie diese heutzutage ihren Einsatz in komplexen Robotersystemen finden. Die hinter den Systemfunktionen liegenden Methoden werden nachfolgend diskutiert. Die Grundlage zur Schaffung von Autonomie stellen dabei perzeptive und estimative Verfahren dar, welche in 2.3.2 eingeführt werden. In 2.3.3 werden gängige Steuerungsarchitekturen beschrieben, welche zur Verwaltung der einzelnen Funktionen in autonomen Systemen eingesetzt werden. Die in 2.3.2 gewonnenen Daten/Information ermöglicht die Generierung eines Welt- oder Umweltmodells und wird in 2.3.4 behandelt, welches die erfassten Daten/Information geeignet verwaltet und zur Manipulationsplanung für das Planungssystem, vergleiche 2.3.5, zur Verfügung stellt. Hierbei wird besonderen Wert auf bereits bestehende Planungsalgorithmen für die Demontage- und Montage, sowie auf Bahnplanungsalgorithmen gelegt. Abschließend erfolgt in 2.3.6 der Blick auf gängige Strukturen von propriozeptiven und exterozeptiven Regelungssystemen, wie diese für die reaktive Ausführung von Robotermanipulationen nötig sind.

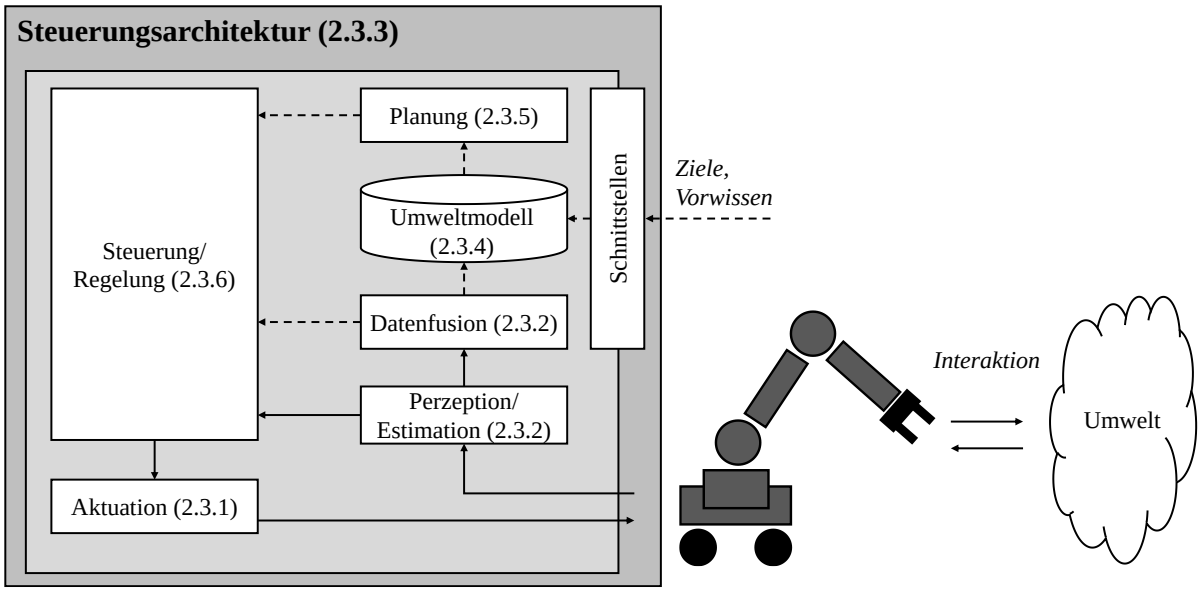


Abbildung 2-5: Zusammenspiel der Schlüsselfunktionen eines autonomen Robotersystems

2.3.1 Kinematische und dynamische Modellierung

Autonome, mobile Robotersysteme, wie sie ihren Einzug in industrielle Aufgaben zukünftig erhalten, bestehen für gewöhnlich aus einer mobilen Plattform sowie einem oder zwei Robotermanipulatoren. Aufgrund der hohen Effizienz bei Bewegungen in strukturierten Umgebungen, wie diese in Produktionseinrichtungen gegeben sind, ist die mobile Plattform mit einem Radfahrwerke (bspw. Standardräder, Castor-Räder, Mecanum-Räder oder Kugelräder) ausgestattet [111]. Als Robotermanipulator werden, aufgrund ihrer Leistungsdichte, Leichtbauvarianten eingesetzt, welche für gewöhnlich als serielle Kinematik ausgeführt sind. Als Aktoren werden bevorzugt elektrische Antriebe, in Form von Synchronmaschinen, verwendet.

Die mathematische Modellierung von kinematischen und dynamischen Zusammenhängen bilden die Grundlage zur Steuerung und Regelung von Robotermanipulationen. Deshalb werden weitergehend die wesentlichen Begrifflichkeiten zur kinematischen und dynamischen Modellierung eingeführt, wobei sich eine umfassendere Darstellung in entsprechender Grundlagenliteratur [112], [113], [114] findet.

Kinematische Modellierung. Um den Zusammenhang zwischen dem kartesischen Raum und dem Achsraum (auch Gelenkwinkelraum) zu beschreiben, verwendet man inverse oder direkte kinematische Transformationen. Die direkte Kinematik $\mathcal{DK}$ beschreibt dabei die Abbildung von gegebenen Gelenkkoordinaten $\underline{q}$ zu kartesischen Koordinaten $\underline{p} = (x \quad y \quad z \quad \alpha_x \quad \beta_y \quad \gamma_z)^T$, also

$$\mathcal{DK}: \underline{q} \rightarrow \underline{p}, mit\ \underline{p} \in \mathbb{R}^6;\ \underline{q} \in \mathbb{R}^n. \tag{2.1}$$

Die direkte Kinematik kann durch homogene Koordinaten der speziellen euklidischen Gruppe $SE(3)$ beschrieben werden, wobei $\underline{q}$ rotatorische Freiheitsgrade θ_i und/oder translatorische Freiheitsgrade d_j besitzen kann. Soll aus gegebener kartesischer Pose $\underline{p}$ die zugehörige Gelenkkonfiguration $\underline{q}$ bestimmt werden, spricht man von der inversen Kinematik $\mathcal{IK}$ (2.2).

$$\mathcal{IK}: \underline{p} \rightarrow \underline{q}. \tag{2.2}$$

Dynamische Modellierung. Wird die Roboterkinematik als Starrkörpersystem beschrieben, so kann die inverse Dynamik $\mathcal{ID}$, also die Abbildung von den Gelenkkoordinaten zu dem Antriebsmoment oder –kraft $\underline{\tau}$, durch (2.3) angegeben werden.

$$\mathcal{ID}: \underline{q}, \underline{\dot{q}}, \underline{\ddot{q}} \rightarrow \underline{\tau}, mit\ \underline{\tau} \in \mathbb{R}^n. \tag{2.3}$$

Die inverse Dynamik für eine gegebene Roboterkinematik kann beispielsweise durch das Lagrange, das rekursive Newton-Euler Verfahren oder das Prinzip nach D'Alembert bestimmt werden. Es sei jedoch angemerkt, dass die Berechnungskomplexität bei den Verfahren stark variiert, von $\mathcal{O}(n^4)$ bei Lagrange, über $\mathcal{O}(n^3)$ bei D'Alembert zu $\mathcal{O}(n)$ für das rekursive Newton-Euler, wobei n die Anzahl an Gelenken darstellt [115]. Wird hingegen der Vektor der Gelenkwinkel, sowie dessen Differentiale gesucht, spricht man von der direkten Dynamik $\mathcal{DD}$ (2.4).

$$\mathcal{DD}: \underline{\tau} \rightarrow \underline{q}, \underline{\dot{q}}, \underline{\ddot{q}}. \tag{2.4}$$

2.3.2 Perzeption, Estimation und Datenfusion

Zur Bestimmung von System- und Umweltzuständen werden in der autonomen Robotik verschiedene Sensoren, Beobachter und Schätzverfahren benötigt, welche sich in perzeptive und estimative Verfahren gliedern. Aufgrund der Komplexität, welche aus der realen Umwelt hervorgerufen wird, werden zur Handhabung von Mehrdeutigkeiten, sowie Sensor- und Prozessrauschen, verschiedene Techniken angewandt um eine genaue Repräsentation des Umweltzustands zu erhalten. Diese Verfahren stellen die Basis für autonomes Verhalten dar und bilden die Informationsgrundlage für die Aufgabenplanung, wie diese auch für die automatisierte Instandhaltung benötigt werden.

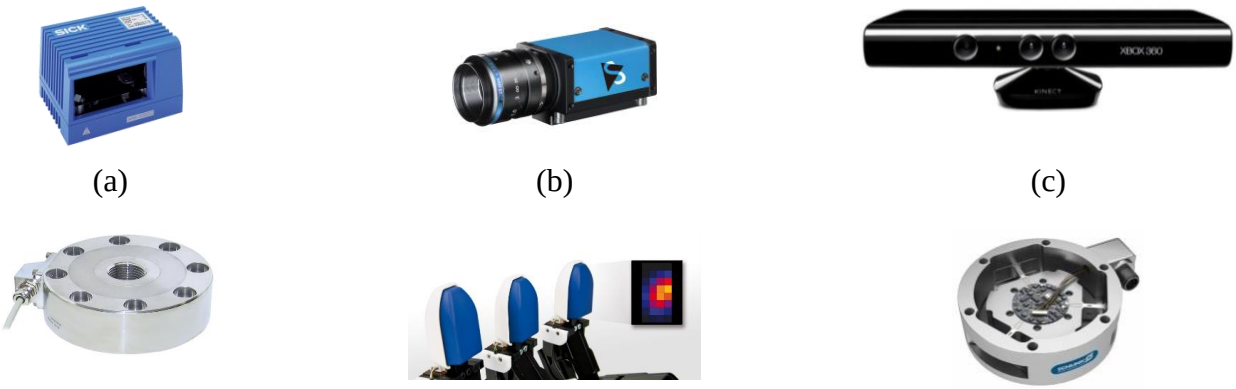
Perzeption. Die Perzeption in einem Robotersystem lässt sich weiter untergliedern in propriozeptive und exterozeptive Perzeption [114]. Propriozeptive Sensorik ermöglicht den Rückschluss auf interne Roboterzustände, wie Motorposition, -geschwindigkeit und -strom, beispielsweise durch einen Encoder, Tachogenerator oder Hall-Sensor. Exterozeptive Perzeption versteht sich als die Erfassung von Umweltzuständen, wie die Objekterkennung und -lokalisation auf Basis von Kameradaten oder der Rückschluss auf Kräfte und Momente, welche während einer Umweltinteraktion auf das Robotersystem wirken. Für eine umfassende Übersicht sei auf [116], [117] verwiesen. Weitergehend sollen die für diese Arbeit wichtigsten Sensorsysteme zur externen Perzeption näher dargestellt werden.

Kamera- und optische 3D-Sensorsysteme. Zur Bestimmung von semantischer und metrischer Umweltinformation, also der Klassifikation und Lokalisation von Objekten, werden für gewöhnlich optische Sensoren verwendet. Man unterscheidet hierbei zwischen Sensoren, welche für eine Messung eine 2D- oder eine (2.5D-) 3D-Abbildung ihrer Umwelt liefern. Klassische optische Entfernungsmesssysteme sind Lasersensoren oder -scanner, siehe Abbildung 2-6 (a). Diese liefern unabhängig vom physikalischen Messprinzip einen Entfernungswert oder eine Punktwolke für eine Messung. Auch kann Tiefeninformation durch RGB-Kameras (Farbkameras) gewonnen werden, Abbildung 2-6 (b). Dies erfolgt mittels Stereokamerasystem oder durch Beobachtung einer Szene aus zwei unterschiedlichen Perspektiven durch ein Monokamerasystem. Jedoch besteht in der Praxis oftmals das Problem aus beiden Szenenbeobachtungen übereinstimmende Bildmerkmale zu

finden [118]. Auch existieren 3D-Tiefenbildkameras (RGBD-Kameras (Red Green Blue Depth); Abbildung 2-6 (c)), mit einen RGB-Chip sowie einen integriertem Tiefenbildsensor. Das Messprinzip beruht dabei entweder auf dem Prinzip der Phasenverschiebung oder der aktiven Triangulation [119]. Während sich reine RGB-Kameras (CCD oder CMOS Technologie) mehr für die 2D-Objektklassifikation eignen, werden im Regelfall zur Objektlokalisation bessere Ergebnisse durch Lasersensoren erzielt. Mit der Kombination von Tiefen- und Farbinformation in RGBD-Kameras, wird ein integriertes Sensorsystem zur Verfügung gestellt, welches die Stärken beider Sensorsysteme vereint und eine robuste 3D-Lokalisierung und Klassifikation ermöglicht [120].

Kraft-Momenten-Sensoren. Für gewöhnlich kommen zur Bestimmung von Interaktionskräften des Robotersystems mit seiner Umwelt, taktile oder Kraft-Momenten-Sensoren zum Einsatz. Es existieren dabei Sensoren für ein- oder mehrdimensionale Messaufgaben. Taktile Sensoren, welche Information über den Kontaktpunkt oder die eingeleitete Kraft zur Verfügung stellen, werden etwa für die Regelung von Greifaufgaben eingesetzt [121]. Die einfachsten Sensoren stellen eine 1D-Messgröße zur Verfügung, wobei die Zug- oder Druckkraft für gewöhnlich auf Basis von Dehnungsmesstreifen, durch eine elektrische Widerstandsveränderung, erfasst wird (Abbildung 2-6 (d)). Taktile Sensormatrizen, vergleiche Abbildung 2-6 (e), erlauben zusätzlich die örtliche Auflösung des Kraftverlaufs. Als physikalisches Messprinzip werden kapazitive, piezoresitive, optische oder elektromechanische Effekte in Form von MEMS (Mikro-Elektro-Mechanische-Systeme) genutzt [122], [121]. Sollen feinfühlige Montageaufgaben mit dem Robotersystem durchgeführt werden, müssen die am Tool Center Point (TCP) einwirkenden Kräfte und Momente gemessen werden. Hierzu stehen industriell 6D-Kraft-Momenten-Sensoren (Abbildung 2-6 (f)) zur Verfügung, welche optisch oder mit DMS-Technik, die einwirkenden Kräfte und Momente auflösen sowie eine Kompensation der Eigenbeschleunigungen des Roboters erlauben.

Estimation. Stehen für bestimmte Systemzustände keine expliziten Sensoren zur Verfügung oder ist deren Integration konstruktiv schwierig oder zu kostspielig, besteht die Möglichkeit einen Beobachter für die Messaufgabe zu entwerfen. Ist das Beobachtbarkeitskriterium nach Kalman sichergestellt, so kann eine Rekonstruktion aus gemessenen Systemzuständen und dem Modell des Prozesses, beispielsweise durch einen Lueneberg-Beobachter erfolgen.



(a) (b) (c)

(d) (e) (f)

Abbildung 2-6: Gebräuchliche Sensoren zur exterozeptiven Perzeption: 2D-Laserscanner [123] (a); RGB-Kamera [124] (b); RGBD-Kamera [125] (c); Zug-Druck-Kraftsensor [126] (d); Taktile Sensormatrix [127] (e); 6D-Kraft-Momentensensor [128] (f)

Der Nachteil, dass sowohl Sensoren als auch der Prozess Unsicherheiten durch Rauschprozesse aufweisen, führt zur Erweiterung des Beobachters auf die Klasse der Zustandsschätzer. Bekannte Vertreter dieser Klasse sind etwa das Bayes-, das Partikel- oder das Kalman-Filter. Das Kalman-Filter [129] ist strukturell wie der Lueneberg-Beobachter aufgebaut, beachtet aber die Rauschgrößen von Sensor und Prozess und liefert unter Voraussetzung einer Gaußschen Normalverteilung des Rauschens, optimal geschätzte Zustandswerte. Die Erweiterung auf nichtlineare Systeme kann durch das Extended (EKF) oder das Unscented Kalman Filter (UKF) [130] erfolgen. Das EKF linearisiert dabei über eine Taylor-Entwicklung erster Ordnung den Prozess, während das UKF Sigma-Punkte (hier Mittelwert und Kovarianz) für jeden Zustand über eine Unscented-Transformation überführt [131], wodurch die Dichteverteilung der Nichtlinearität approximiert wird und hierdurch höhere Systemordnungen beachtet werden können.

Datenfusion. Aufgrund der Tatsache, dass zur Erfassung von Umweltzuständen kein universelles Messprinzip in Form eines Einzelsensors existiert [132], werden für komplexe Aufgaben Daten aus verschiedenen Sensoren fusioniert. Dies ermöglicht die Genauigkeit einer Hypothese zu maximieren und Mehrdeutigkeiten zu eliminieren. Die Fusion kann dabei auf drei verschiedenen Ebenen stattfinden. Die niedrigste Abstraktion auf der Sensordatenebene, über die Merkmalsebene bis zur symbolischen Ebene, welche zur Entscheidungsfindung dient [133]. Die Methoden beruhen dabei größtenteils auf statistischen Verfahren (bspw. Bayes, Dempster-Shafer Theorie), welche die Behandlung von Unsicherheiten erlauben. Diese lassen auch eine geeignete Fusion von a priori Information mit online Information zu, die aus Sensordaten extrahiert wurde. Anwendungen stellen etwa die Kombination von kraft- und bildbasierten Sensordaten für Konturfolgeaufgaben [134] oder die Mikromontage mit einem Robotermanipulator dar [135]. Einen allgemeinen Überblick der Thematik findet sich in [136], [132], [137].

2.3.3 Steuerungsarchitekturen

Die hohe Komplexität autonomer Robotersysteme erfordert eine Steuerungsarchitektur, welche die Koordination verschiedenster Dienste übernimmt, sodass die Aufgaben der Perzeption und Estimation, der Umweltmodellierung, der Planung und die Regelung von Aktionen geeignet ausgeführt werden können. Im Laufe der Erforschung autonomer Robotersysteme haben sich diverse Steuerungsparadigmen entwickelt, wobei die gebräuchlichsten folgend skizziert werden.

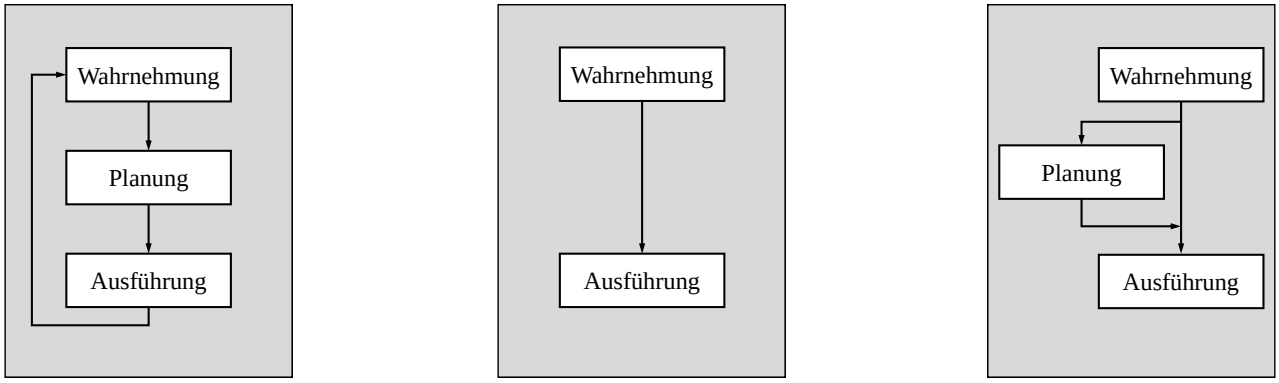
Deliberative Steuerungsarchitekturen. Die deliberative Struktur stellt das erste bekannte Architekturparadigma für ein autonomes Robotersystem dar und wurde für den am Stanford Research Institute entwickelten Roboter Shakey [138] eingesetzt. Es handelt sich dabei um eine hierarchische Anordnung der Komponenten Wahrnehmung-Planung-Ausführung *(engl. „Sense-Plan-Act“)*, weshalb oftmals auch der Begriff SPA-Architektur [139] verwendet wird. Die schematische Struktur ist in Abbildung 2-7 (a) abgebildet. Dabei werden zyklisch neue Sensorwerte erfasst und an die Planungskomponente weitergegeben. Im Planungsmodul werden die neuen Sensorwerte in eine geeignete symbolische Repräsentation abgebildet, welche die Umwelt beschreibt. Fortgeschrittene Ansätze ermöglichen ebenso die Aktualisierung des Umweltmodells bei Diskrepanz mit der Realität [140]. Auf Basis des Umweltmodells erfolgt dann die Planung, welche versucht, die vorgegebenen Ziele zu erreichen und diese in Form von Stellsignalen an die Aktoren weitergibt. Der geänderte Umweltzustand wird dann wieder erfasst und der Zyklus beginnt

erneut. Als vorteilhaft erweist sich die Tatsache, dass Ziele prinzipiell planbar sind. Problematisch ist jedoch, dass die Umwelt geeignet modelliert und darüber hinaus mit der Realität übereinstimmen muss. Kommt es zur Differenz von Modell zur Realität reagiert das Robotersystem nicht mehr geeignet und die geforderten Ziele werden nicht mehr erreicht. Auch kann nicht dynamisch auf Zustandsänderungen reagiert werden, da durch die sequentielle Planung kein reaktives Verhalten möglich ist. Bei komplexen Umweltmodellen führt das deliberative Paradigma ebenso zu einer hohen Zeit- und Speicherkomplexität.

Reaktive Steuerungsarchitekturen. Reaktive Ansätze verzichten auf den Einsatz der Planungskomponente. Hierdurch werden Sensoreingaben direkt auf die Aktoreinheit abgebildet, siehe Abbildung 2-7 (b). Berechnungen zwischen Sensoreingaben und Aktion sind erlaubt, solange das System „ausreichend schnell“ auf die erforderliche Situation reagiert [141]. Eines der bekanntesten Beispiele stellt die Subsumptions-Architektur [142] dar. In dieser sind verschiedene Verhaltensmuster in Form von Zustandsautomaten fest implementiert. Die Verhaltensmuster sind dabei nach Komplexitätsgrad angeordnet, sodass niederprioritäre Verhalten von höherwertigen beeinflusst werden können. Prinzipiell haben reaktive Architekturen den Vorteil, dass diese dynamisch auf Umwelteinflüsse reagieren können. Jedoch kann durch den Verzicht auf das Umweltmodell und die Planungskomponente keine langfristige Zielverfolgung stattfinden, wodurch sie sich nicht für alle Anwendungen eignet [143]. Ebenso lassen sich komplexe Vorgänge nicht geeignet abbilden. Die Optimalität des Lösungsweges ist dabei selten gegeben.

Hybride Steuerungsarchitekturen. Hybride Konzepte kombinieren die Ansätze der deliberativen und reaktiven Architektur, wodurch sowohl kurzfristige, dynamische als auch langfristige, globale Ziele erreichbar sind. Durch die Verknüpfung der Vorteile stellen sie die derzeit am häufigsten eingesetzte Architektur dar, wobei die strikte Zerlegung in diese drei Schichten nicht sonderlich streng behandelt wird [144], [145]. In Abbildung 2-7 (c) ist die schematische Struktur einer hybriden Architektur aufgeführt. Die Planungsebene kann dabei in den Regelkreis zwischen Wahrnehmung und Ausführung für zeitunkritische Aktionen eingreifen, während zeitkritische Aktionen (bspw. Kraftregelung) in Echtzeit auf Basis einer reaktiven Ebene gelöst werden. Speziell für die Ausführung von Manipulationsaufgaben existiert ein aktionsprimitivbasiertes Paradigma, das umfassend in der Monographie [146] und verschiedenen weiteren Veröffentlichungen [147], [148] diskutiert wird. Die Grundidee beruht dabei auf dem Compliance Framework nach [149]. Der Vorteil einer aktionsprimitivbasierten Steuerungsarchitektur liegt darin begründet, dass diese eine formale Methode darstellt, sensorgeregelte und -überwachte Robotermanipulationen eindeutig zu beschreiben und eine geeignete Kapselung zwischen Planungsebene und reaktiver Ausführungsebene abbildet. Diese ermöglicht die Steuerung und Regelung von Roboteraufgaben auf Grundlage elementarer Primitive, den sogenannten Aktionsprimitiven. Durch das Zusammensetzen von Manipulationsaufgaben aus einer konfigurierbaren Menge an Aktionsprimitiven wird eine hohe Wiederverwendbarkeit erzielt. Das Konzept wird in zahlreichen Arbeiten für unterschiedliche Manipulationsaufgaben [108], [106], [150] verwendet, was die praktische Relevanz und die Leistungsfähigkeit unterstreicht.

In den letzten Jahren hat sich als Architektur zur Umsetzung von Steuerungsarchitekturen im Bereich (Service-)Robotik, vor allem das Robot Operating System (ROS) [151] innerhalb der Forschung durchgesetzt.



(a) (b) (c)

Abbildung 2-7: Schematische Darstellung von Steuerungsarchitekturen autonomer Robotersysteme (Modifiziert nach [152]). Deliberative (a); reaktive (b); und hybride Architektur (c)

Es stellt bereits eine Vielzahl an Treibern für die unterschiedlichsten Hardwarekomponenten sowie verschiedene Algorithmen, beispielsweise für die Navigation mobiler Plattformen, zur Verfügung. Die Kommunikation in ROS nutzt Paradigmen der verteilten Systeme und bietet sowohl synchrone als auch asynchrone Kommunikation zwischen einzelnen Steuerungstasks an. Dies und die Möglichkeit der Verwendung verschiedener Programmiersprachen hat die Nutzenakzeptanz derart gesteigert, dass in der Forschung der Entwurf von Steuerungsarchitekturen oftmals auf Basis von ROS erfolgt.

Zusammenfassend gilt, dass unterschiedliche Steuerungsarchitekturen für autonome Robotersysteme bestehen. Derzeit existiert jedoch keine allgemein etablierte Architektur. Dies liegt vor allem an der Divergenz, hinsichtlich Einsatzgebiet und Autonomiegrad, welcher innerhalb der Robotik vorherrscht. Es lässt sich der Trend erkennen, dass für jede einzelne Anwendung die passende Struktur entwickelt wird [150], welche sich jedoch aus den drei Basisarchitekturen ableiten lässt, wenn auch spezielle, anwendungsabhängige Designentscheidungen getroffen werden.

2.3.4 Umweltdatenverarbeitung und Modellierung

Die Verarbeitung von Umweltdaten, die Bildung von Information und deren Interpretation, haben in der Robotik die Aufgabe eine interne Repräsentation der Umgebung aufzubauen, welche als Wissensgrundlage für das Robotersystem agiert. Allgemein beinhaltet das Umweltmodell alle Information in einem aufgabenzentrierten Abstraktionsniveau, welche a priori bekannt oder online mithilfe perzeptiver Maßnahmen gesammelt werden kann. Das Umweltmodell muss dabei so strukturiert und formalisiert sein, dass sich in der Planungsebene neue Aktionen aus diesem ableiten lassen. Folgend werden bekannte Methoden aufgezeigt, die eine Bildung eines Umweltmodells erlauben. Prinzipiell kann nach [153] grundsätzlich zwischen drei, in der Robotik gebräuchlichen Informationstypen, unterschieden werden:

- **Geometrische Information:** Betrachtet die Ausdehnung und Lage von Objekten.
- **Topologische Information:** Erweiterung um Beziehungen zwischen Objekten.
- **Semantische Information:** Zusätzliche Information über Art und Struktur der Objekte.

Die Gewinnung der einzelnen Informationstypen aus a priori und Sensordaten sollen weiterführend auf ihren Einsatz für die automatisierte Instandhaltung analysiert werden. Das Unterkapitel zeigt abschließend, die für die Robotik eingesetzten Methoden auf, die für verschiedene Aufgaben zur Umweltdaten- und Wissensverarbeitung sowie -modellierung eingesetzt werden.

Navigation. Die Navigation umfasst die Aufgaben Kartenerstellung, Lokalisierung und Bahnplanung, wobei letzteres ausführlich in 2.3.5 dargestellt ist. Die Aufgabe der synchronen Lokalisierung und Kartierung wird für gewöhnlich als SLAM *(engl. „Synchronous Localisation and Mapping“)* bezeichnet. Zur Erstellung einer Karte werden beständig einzelne Sensorbeobachtungen miteinander registriert. Als Repräsentationsmöglichkeiten der Roboterkarte werden geometrische, topologische [154] oder semantische [155], [156], [157] Konzepte eingesetzt. Während geometrische Karten sich sehr gut für die Bahnplanung eignen, besteht der Vorteil bei der Verwendung von topologischen Karten darin, dass diese sehr gut zur Planung eingesetzt werden können. Semantische Karten enthalten zusätzlich Informationen über kartierte Strukturen und Objekte zu bekannten Konzepten und Relationen sowie statisches Wissen über diese [158], womit ein erweiterter Autonomiegrad erst erlaubt wird. Geometrische Karten sind für den Einsatz von Bahnplaner unumgänglich, wobei deren Darstellung von einfachen 2D-Rasterkarten *(engl. Occupancy Grid Map)* bis hin zu Octomaps [159] reichen. Diese erlauben eine probabilistische Repräsentation, sowie aufgrund ihrer Datenstruktur, eine effiziente Speicherung dreidimensionaler Voxelkarten. Als Sensordaten zur Kartenerstellung werden vornehmlich optische Sensoren eingesetzt um das Navigationsproblem zu lösen. Realisiert werden Anwendungen auf Basis von RBG-Sensoren [160], [161] oder Laserscannern [162]. Erstere Ansätze werden oftmals als Visual SLAM oder Structure-from-Motion bezeichnet. Heutige Benchmark Systeme [163], [164] arbeiten mit RGBD-Sensordaten, da diese die Vorteile beider Ansätze kombinieren. Die Registrierung der einzelnen Beobachtungen aus den RGB-Daten stellt ein Perspective-n-Point (PnP) Problem dar, wobei die Transformationsschätzung mittels Random Sample Consensus (RANSAC) [165] und Acht-Punkte-Algorithmus [166] gelöst werden kann. Zur Registrierung von Punktwolken werden Iterative Closest Point (ICP) Algorithmen in verschiedenen Ausführungen verwendet [167], [168]. Gemeinsam haben fast alle Ansätze, dass diese mit probabilistischen Methoden, wie das Bayes-, Kalman-, oder Partikelfilter arbeiten um die geschätzte Genauigkeit der Posen zu verbessern. Auch werden graphbasierte Optimierungsmethoden zur Posenschätzung verwendet [169]. Für eine umfassende Übersicht hierzu sei auf das Standardwerk [170] verwiesen.

Objekterkennung und -lokalisierung. Ist die Umwelt des Roboters nicht vorab bekannt, müssen Objekte und deren Pose aus den zur Verfügung stehenden Sensordaten extrahiert werden. Als Sensoren wird für gewöhnlich auf die in 2.3.2 vorgestellten, optischen Sensoren zurückgegriffen. Zur optischen Erkennung und Lokalisation von Objekten bestehen grundsätzlich verschiedene Verfahren, welche sich nach [171] in die folgenden Kategorien gliedern lassen:

- **Bildkategorisierung:** Gesamtzuordnung der Information.
- **Objektlokalisation:** Erfassung einzelner Objekte in der Information. Aus der Annotation in Bildkoordinaten lässt sich dann über die intrinsische und extrinsische Transformation die metrische Lokalisation (PnP-Problem) durchführen.
- **Semantische Segmentierung:** Beschriftung jedes einzelnen Informationselements mit der zugehörigen Kategorie.

Da die Objekterkennung und -lokalisation ein eigenes Forschungsfeld darstellt, kann in dieser Arbeit nur eine kurze Übersicht über die wichtigsten Verfahren gegeben werden, die auch für die hier vorliegende Problemstellung von Bedeutung sind. Der gewöhnliche Ablauf einer Objekterkennungs- und Lokalisierungsaufgabe lässt sich für gewöhnlich in fünf Schritte, nach Abbildung 2-8, gliedern. Eine Segmentierung muss dabei nicht zwingend durchgeführt werden und deren Einsatz hängt stark von der durchzuführenden Klassifikationsaufgabe und des verwendeten Klassifikationsalgorithmus ab. Damit ein Objekt einer Klasse zugeordnet werden kann, müssen eindeutige Objekt- bzw. Bildmerkmale zur Repräsentation zur Verfügung stehen. Als Bildmerkmale können beispielsweise einfache geometrische Merkmale (Kanten, Kontur, Momente), Farbe und Grauwertverteilung (Histogramm), Textur oder Deskriptoren verwendet werden. Auch wird zwischen Merkmalen unterschieden, die lokal oder global die komplette Bildinformation beschreiben. Dabei eignen sich lokale Merkmale vor allem bei Überdeckungen und Variabilität der geometrischen Anordnung [172]. Der Objekterkennungsprozess kann dabei in Ansätze der erscheinungsbasierten *(engl. „appearance-based“)* oder modellbasierten *(engl. „model-based“)* Objekterkennung untergliedert werden. Erscheinungsbasierte Ansätze nutzen dabei für gewöhnlich Methoden der Mustererkennung, während modellbasierte Ansätze von einem vorab definierten geometrischem Modell der Szene ausgehen, welches mit der aktuellen Aufnahme gefittet wird [173]. Erscheinungsbasierte Ansätze erzielen meist sehr gute Resultate für Objekte, die eine hohe Textur aufweisen, während bei texturlosen Objekten meistens nicht genügend Bildmerkmale extrahiert werden können. Hierfür eignen sich modellbasierte Methoden. Jedoch haben diese wiederum den Nachteil, dass sie schlecht mit Verdeckungen zurechtkommen.

Erscheinungsbasierte Objekterkennung. In erscheinungsbasierten Ansätzen werden für RGB-Daten zumeist lokale, visuelle Merkmale mittels Keypointdetektoren erkannt und durch geeignete Deskriptoren (Merkmalsvektoren der einzelnen Keypoints) beschrieben. Abbildung 2-9 (a) zeigt beispielsweise die Anwendung eines SURF-Matching. Eines der bekanntesten Verfahren für RGB-Daten stellt SIFT [174], [175] *(engl. „Scale Invariant Feature Transformation“)* dar. Der große Vorteil ist, dass SIFT invariant gegenüber Skalierung und Rotation ist. Eine Weiterentwicklung wurde mit SURF [176] *(engl. „Speeded-Up Robust Features“)* eingeführt. SURF weist Geschwindigkeitsvorteile in seiner Berechnungsvorschrift gegenüber SIFT auf, da verschiedene Approximationen in dem Filterprinzip vorgenommen werden. Für ausgeprägte geometrische Konturen eignen sich vor allem HOG-Deskriptoren [177] *(engl. „Histograms of Oriented Gradients“)*, welche jedoch keine Skalierungs- und Rotationsinvarianz aufweisen. Prinzipiell existieren eine Vielzahl an Detektoren und Deskriptoren wie beispielsweise Haar- [178] oder BRISK-Features [179], was zeigt, dass bisher kein universell geeignetes Verfahren vorhanden ist.

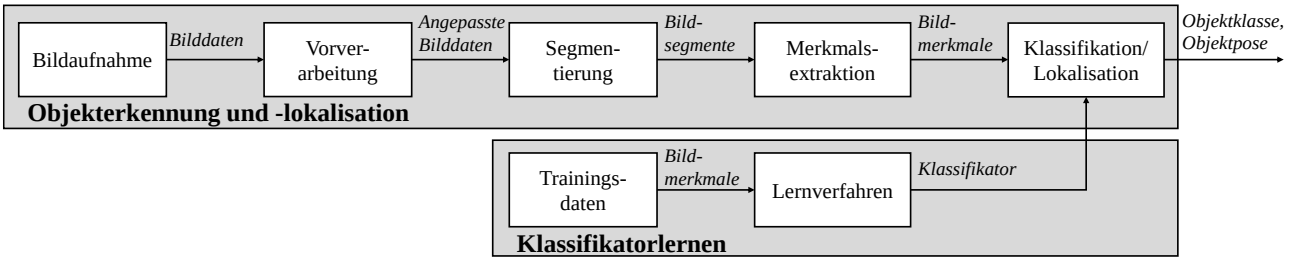


Abbildung 2-8: Vereinfachter Informationsfluss zur Objekterkennung und -lokalisation

Eine Arbeit die verschiedene Farbdeskriptoren vergleicht stellt [180] dar. Auch für Punktwolkendaten sind verschiedene Deskriptoren wie verschiedene Varianten von PFH [181] *(engl. „Point Feature Histogram“)* oder SHOT [182] *(engl. „Signature of Histograms of OrienTations“)* sowie deren zusätzlichen Kombination mit Farbdaten wie SHOTCOLOR [183] bekannt. Prinzipiell eignen sich Deskriptoren, die sowohl Form als auch Farbe vereinen, besser, siehe hierzu [184]. Ein verbreitetes Verfahren zur Objektkategorisierung ist der Bag of Visual Words (BoVW) Ansatz [185] (oft auch Bag of Features; oder Bag of Keypoints). Hierzu werden die Deskriptoren mittels k-means Clustering [186] gruppiert, wobei die k-Clusterzentren die visuellen Wörter darstellen. Das visuelle Vokabular bildet dann die Repräsentation des Bildes, welches als Histogramm beschrieben wird. Als Klassifikator wird für gewöhnlich eine Support Vector Machine (SVM) [171] eingesetzt. Der große Nachteil des Verfahrens ist, dass räumliche Informationen verworfen werden, da es sich um ein globales Verfahren handelt. Dies führt zu Erweiterungen wie etwa dem Spatial Pyramid Matching [187]. Ein Verfahren, das auf Basis mehrerer kaskadierter Klassifikatoren arbeitet und gleichzeitig eine Objektlokalisierung aufgrund seiner lokalen Bildbetrachtung erlaubt, stellt der Viola-Jones-Detektor [178] dar. Die Idee besteht darin, die ersten Stufen mit schwachen Klassifikatoren auszustatten, sodass eine schnelle Hintergrundentfernung durchführbar ist. Die folgenden Kaskaden verwenden stärkere Klassifikatoren, die nur noch eine reduzierte Anzahl an Anfragen bearbeiten müssen. Dies führt zu einer günstigen Zeitkomplexität des Verfahrens. Derzeitig nimmt aufgrund der gestiegenen Rechenleistung auch die Popularität von Convolutional Neural Networks (CNN) in der Bildverarbeitung zu, siehe hierzu etwa [188]. Diese erlauben eine sehr hohe Erkennungsrate, was vor allem auf die spezifische Generierung der visuellen Merkmale, durch die Faltung der einzelnen Eingabebilder, zurückzuführen ist.

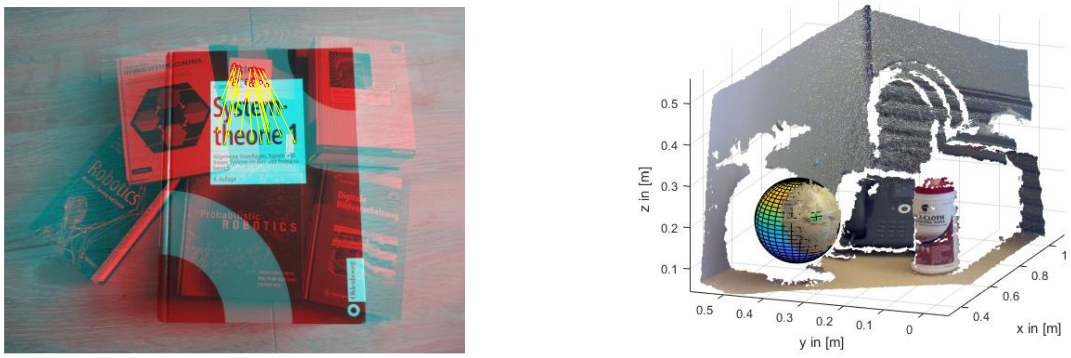
Modellbasierte Objekterkennung. Für die modellbasierte Objekterkennung werden etwa Templates für das Matching von Farbdaten eingesetzt und anhand des Korrelationsmaßes die Erkennungsaufgabe durchgeführt. Ein Verfahren, welches CAD-Modelle nutzt und auf Basis von monokularen Farbbildern arbeitet, wird in [189] präsentiert. Hierzu werden verschiedene virtuelle Ansichten generiert und mit dem Eingabebild auf Basis von Kanten abgeglichen. Auch für Punktwolkendaten sind Verfahren vorhanden, die ein Matching der eingehenden Sensordaten mit vorhandenen CAD-Modelle erlauben [190].

Weitergehend sollen die für die einzelnen Schritte der Objekterkennung, gemäß Abbildung 2-8, und die hierzu gebräuchlichen Verfahren vorgestellt werden.

Bildvorverarbeitung. Diese dient der Reduktion von Sensorrauschen durch lineare oder nichtlineare Filter, wie etwa Gauß- oder Medianfilter. Auch die Anpassung von Kontrast und Helligkeit sowie die Transformation in die jeweilig relevanten Farbräume, werden in diesem Schritt für Farbdaten durchgeführt. Punktwolkendaten werden oftmals einer Abstandsfilterung oder speziellen Rauschfiltern für Punktwolken, wie dem statistischen Ausreisfilter [191] oder dem Specklefilter [119], unterzogen. Zur Reduktion der Datenmenge werden beispielsweise stichprobenbasierte Downsampling-Verfahren oder Box-Grid-Filter eingesetzt.

Segmentierung. Versteht sich als Prozess, Bild- und Punktwolkeninformation, aufgrund von Merkmalen in zusammenhängende Bereiche, beispielsweise Objekte, zu gliedern. Man unterscheidet hierbei nach [192] zwischen folgenden Verfahren, wobei in der Praxis oftmals Kombinationen verwendet werden:

- **Pixelorientierte Segmentierung:** Es wird auf Basis der Pixel- oder Voxelinformation entschieden, ob dieser einem Objekt zugehörig ist. Vertreter sind Schwellwertverfahren, die als Merkmal die Farb- oder Grauwertverteilung nutzen. So kann auf Basis bi- oder multimodaler Verteilungen eine Segmentierung erfolgen (bspw. Otsu-Verfahren [193]). Problematisch ist jedoch, dass die Bestimmung geeigneter Schwellwerte nicht immer möglich ist und eine farbliche Homogenität der einzelnen Objekte vorhanden sein muss. Ein derzeit aktives Forschungsgebiet stellt auch die semantische Segmentierung mittels CNNs dar, die bisher bekannte Algorithmen deutlich übertrifft, für einen Überblick siehe [194].
- **Kantenbasierte Segmentierung:** Auf Basis der Kanteninformation werden einzelne Objektregionen bestimmt. Kantenbasierte Methoden nutzen in Farbbildern die Eigenschaft, dass hohe Ortsfrequenzen an Kantenregionen auftreten. Zur Detektion werden Ableitungsfilter wie das Laplace- oder Sobelfilter verwendet. Zur gleichzeitigen Reduktion der Rauschanteile ist vor allem der Einsatz von Bandpassfiltern wie dem DOG- *(engl. „Difference of Gaussian")* oder LOG-Filter *(engl. "Laplace of Gaussian")* von Vorteil. Als robustes Verfahren hat sich der Canny-Algorithmus [195] etabliert, welcher ein mehrstufiges Filterverfahren darstellt. Zur Glättung wird ein Gauß-Kernel angewendet und anschließend der Gradient über einen Sobel-Kernel bestimmt. Die Information über Orientierung und Betrag definiert dann die Kantenzugehörigkeit. In Punktwolken können Kanten beispielsweise über Scan-Line-Approximation bestimmt werden [196].
- **Regionenbasierte Segmentierung:** Beachtet nicht nur isoliert den einzelnen Informationsträger, sondern zusätzlich die Nachbarschaft, um eine Entscheidung zu treffen. Regionenbasierte Verfahren, wie etwa das Regionenwachstum *(engl. „Region Growing")*, wurden ursprünglich für die Segmentierung von Farbbildern entwickelt [197]. Auch lassen sich diese sehr gut für Punktwolkendaten einsetzen, wie in [157] gezeigt. Es handelt sich um einen iterativen Algorithmus, welcher in Abhängigkeit von Nachbarschaftskriterien die Region erweitert bis der Iterationsprozess keine Verbesserung mehr bringt. Für Punktwolkendaten sind auch clusterbasierte Verfahren wie DBSCAN (Density Based Spatial Clustering of Application with Noise) [198] und die Erweiterung auf ein generell räumliches Verfahren [199] einsetzbar. Die Grundidee besteht darin, dass ein Punkt einer Region zugehörig ist, wenn dieser eine definierte Anzahl an Nachbarschaftspunkten besitzt. Die Nachbarschaftsumgebung kann dabei über verschiedene Metriken wie die Euklidische- oder Manhattan-Distanz bestimmt werden. Vorteilhaft ist, dass die Anzahl der Cluster nicht a priori bekannt sein muss wie es bei k-means der Fall ist.
- **Modellbasierte Segmentierung:** Es werden nicht nur lokale Nachbarschaftsinformationen verwendet, sondern geometrische Modelle über die Objekte. Eines der bekanntesten modellbasierten Segmentierungsverfahren für Farbbilder stellt die Hough-Transformation dar. Diese dient der Erkennung verschiedener geometrischer Objekte auf der Basis von Binärbildern und nutzt dabei die Eigenschaft, dass Datenpunkte durch geometrische Parametergleichungen abgebildet werden. Für die modellbasierte Segmentierung von Punktwolkendaten werden diese auf geometrische Primitive wie Ebenen, Zylinder oder Sphären gefittet [200], wobei als Grundlage Varianten des RANSAC-Algorithmus wie der MLESAC-Algorithmus (Maximum Likelihood Estimation Sample Consensus) [201] eingesetzt werden, vergleiche Abbildung 2-9 (b), in der diese Anwendung aufgezeigt wird.



(a) (b)

Abbildung 2-9: Matching von SURF-Features aus zwei unterschiedlichen Szenen (a); Modellbasierte Segmentierung geometrischer Objekte (hier Sphäre) mittels MLESAC [202] (b)

Klassifikation. Die Aufgabe der Klassifikation ist es, einem Objekt eine Klasse aus einer Menge an möglichen Klassen anhand der extrahierten Merkmale zuzuordnen. Es wird unterschieden zwischen überwachten und unüberwachten Klassifikationsverfahren. Bei überwachten Verfahren stehen korrekte Ausgaben zum Lernen des Klassifikators zur Verfügung, während unüberwachte Verfahren Regelmäßigkeiten aufgrund von Eingabedaten lernen [203]. Ein bekanntes, überwachtes Lernverfahren stellt die SVM dar, welche sich durch ihre Generalisierungsfähigkeit und der geringen Rechenzeit der Lern- und Testschritte besonders auszeichnet [171]. Bei CNNs ist ein gängiges Klassifikationsverfahren die Nutzung der Softmax-Regression. Grundlagenliteratur über das Gebiet der Klassifikation stellen die Werke [186], [203] dar.

Objektlokalisation. Die Berechnung der Objektpose in Farbbildern kann, wie bereits im Abschnitt Navigation beschrieben, auf Basis von RANSAC und 8-Punkte-Algorithmus geschätzt werden. Ein weiteres, bekanntes monokulares Verfahren stellt der POSIT-Algorithmus [204] *(engl. „Pose from Orthography and Scaling with Iterations“)* dar. Zur Posenbestimmung mittels Punktwolkendaten wird der ICP-Algorithmus in verschiedenen Varianten eingesetzt, vergleiche [168]. Die Grundidee besteht darin, eine homogene Transformation für eine gegebene Punktwolke auf Basis eines Objektmodells zu schätzen, sodass ein definiertes Abstandsmaß minimiert wird.

Beispielsysteme. Leistungsfähige Bildverarbeitungssysteme, welche im Bereich der autonomen Robotik eingesetzt werden, sind vormals für Haushaltsumgebungen spezifiziert, wie beispielsweise die Arbeiten [205], [206], [207], [208] zeigen. Ein Framework, welches auf Basis von RGB-Daten arbeitet, stellt [205] dar. Dieses ist auf eine hohe Verarbeitungsgeschwindigkeit ausgelegt, wobei dies auf die Anpassung einzelner Algorithmen sowie architekturbasierter Maßnahmen rückgeführt werden kann, welche eine hohe Parallelisierung der einzelnen Operationen erlaubt. Zur Objektmodellierung werden SIFT und SURF-Feature extrahiert, die aus verschiedenen Objekt-ansichten gewonnen werden. Mittels Structure-from-Motion wird so ein räumliches 3D-Modell erstellt, welches für die spätere Objekterkennungs- und Lokalisierungsaufgabe verwendet werden kann. Zur Lösung des Korrespondenz- und Posenschätzungsproblems wird der Iterative Clustering Estimation (ICE) Algorithmus eingeführt. Der ICE bildet aus den einzelnen Feature Cluster, wodurch eine robustere Posenschätzung erzielt werden kann. Das System eignet sich aufgrund des erscheinungsbasierten Ansatzes nur für Objekte die ausreichend Textur aufweisen, da andernfalls nicht genügend Feature extrahiert werden können.

Das in der Arbeit [206] veröffentlichte Objekterkennungssystem arbeitet auf Basis von Tiefen- und RGB-Daten. Zur Objekterkennung verwendet das System unterschiedliche Features, wobei auf Basis von 3D-Merkmalen eine Oberflächensegmentierung und hieraus die Zuordnung der geometrischen Objektklasse erfolgen. Dadurch lässt sich die zentrale Objektregion *(engl. „region of interest")* (ROI) definieren, sodass aus dem zugehörigen RGB-Ausschnitt SIFT Features extrahiert werden, welche in einem Visual Bag of Words zur Klassifizierung verwendet werden. Falls kein Matching des Objekts mit der Objektdatenbank gefunden werden kann, wird das Objekt als neues Objekt in der Objektdatenbank registriert. Dieses Vorgehen eignet sich vor allem für Umgebungen, in denen eine Vielzahl an unterschiedlichen Objekten vorhanden ist.

Das in [208] vorgestellte System nutzt verschiedene Objekterkennungsstrategien in Abhängigkeit der aufkommenden Eingabedaten. So werden für texturierte Objekte erscheinungsbasierte Ansätze verwendet, während für texturlose Objekte modellbasierte Verfahren zum Einsatz kommen. Hierdurch lassen sich die Vorteile beider Ansätze kombinieren.

Neben den dargestellten Frameworks bestehen auch Open Source Bibliotheken wie OpenCV oder speziell für die Verarbeitung von Punktwolkendaten die Point Cloud Library (PCL) [209], [210], welche gängige Algorithmen zur Objekterkennung und Posenschätzung zur Verfügung stellen.

Next-Best-View. Die Kartierung oder Kartenexploration, sowie die Objekterkennung und -lokalisation sind in der Praxis sehr stark von der Beobachtungspose des Sensors abhängig. Aufgrund der hohen Relevanz wird die Planung einer optimalen Sensorpose, zur Lösung einer visuellen Aufgabe, bereits seit den 1980ern, beispielsweise in der Arbeit [211], für die Objektmodellierung diskutiert. Dieses unter Next-Best-View (NBV) bekannte Problem wurde bereits für zahlreiche weitere Anwendungen formuliert, beispielsweise zur Exploration [212] für die Objektrekonstruktion [213] oder die Objekterkennung [214]. Prinzipiell lassen sich alle Ansätze in modellbasierte und modellfreie NBV unterteilen. In [212] wird die Exploration des Konfigurationsraums mit einer Objektmodellierungsaufgabe kombiniert. Dabei wird den Aufgaben der Informationsgewinn *(engl. „Informationgain")* zugeordnet und statisch gewichtet, um einen neue Ansicht zu gewinnen, die beide Aufgaben gleichzeitig löst. Ein ähnliches Vorgehen wird in der Arbeit [215] vorgestellt, die die Objektrekonstruktion und Erkennung kombiniert. Nach der Bestimmung einer Menge an möglichen Sensorposen durch Boundary Search, wird die nächste Sensorpose aus einer gewichteten Kombination aus Entropiereduktion und Bewertung der Oberflächenqualität gewählt [216]. In [217] wird ein Explorationsansatz vorgestellt, der auf der Teilung von Arbeitsplatz für Mensch und Roboter ausgelegt ist. Das Aufmerksamkeitsverhalten des Roboters wird dabei durch einen NBV-Algorithmus gesteuert. Die Umgebung wird als Voxelgitter modelliert und es wird gezeigt, wie die Berechnung des NBV-Problems, durch die Verlagerung auf die GPU, zeitlich deutlich verbessert werden kann. Verschiedene Metriken für die Validierung eines NBV sind in der Arbeit [218] zusammengefasst und werden dort ausreichend diskutiert.

Wissensakquirierung, -repräsentation und -verarbeitung. In den beiden vorherigen Unterkapiteln wurden bekannte Verfahren dargestellt, die es erlauben aus Sensordaten relevante Umweltinformation zu gewinnen. Neben der reinen Nutzung von Sensorinformation wird oftmals auch domänenspezifisches Wissen a priori dem Robotersystem zur Verfügung gestellt oder kann gegebenenfalls aus CAD-Daten oder domänenrelevanten Wissensdatenbanken gewonnen werden. Jedoch muss dieses geeignet formalisiert, strukturiert und verwaltet werden, sodass eine weitere

Verarbeitung zur Schlussfolgerung nötiger Aktionen zur Lösung einer Aufgabe (Planung) ermöglicht wird. Nur so kann die nötige Flexibilität erzielt werden, die es erlaubt, auf Basis von explizitem Wissen durch ein Inferenzsystem implizites Wissen abzuleiten. Deshalb soll nachfolgend untersucht werden, welche Methoden zur Formalisierung von Wissen bestehen, welche aus der Arbeit [219] entnommen sind und hier kurz zusammengefasst werden. Abschließend soll auf bekannte Systeme der Wissensverarbeitung innerhalb der Robotik eingegangen werden.

Semantische Netze. Stellen eine graphische Repräsentation in Form von gerichteten Graphen dar. Dabei bilden die Knoten die Begriffe *(engl. „Concepts“)*, Methoden oder Objekte *(engl. „Entities“)* ab, während die Kanten die Beziehungen *(engl. „Relationships“)* repräsentieren. Die Kanten lassen sich dabei auch als zweistellige Prädikate darstellen [220]. Ein Vorteil dieser Repräsentation ist, dass Inferenz als Graphensuche gelöst werden kann. Demgegenüber stehen die Nachteile, einer nicht definierten Semantik und das nur binäre Relationen zwischen den Knoten möglich sind [144].

Frames. Das Frame Konzept geht auf die Arbeit [221] zurück. Frames stellen dabei einen Repräsentationstyp dar, welcher für ein Objekt die Attribute in sogenannten Slots beinhaltet. Ein Frame-System wird aus mehreren Frames gebildet, welches auch Vererbung zulässt. Frames eignen sich vor allem für die Repräsentation von Wissen in stark strukturierten Domänen, weisen aber die gleichen Probleme der fehlenden Semantik auf [219].

Logik. Die Grundlagen hierzu stammen, im Gegensatz zu den beiden vorherigen Konzepten, nicht aus den Kognitionswissenschaften, sondern aus der Mathematik. Vertreter sind die Aussagenlogik, die Prädikatenlogik oder die Beschreibungslogik, welche eine festgelegte Syntax und Semantik besitzen. Während die Aussagenlogik für praktische Problemstellungen zu einer oftmals komplizierten Darstellung führt, ermöglicht die Prädikatenlogik eine kompaktere Darstellung zur Formulierung von Aussagen über einen gesamten Grundbereich zu treffen. Sprachen zur Beschreibungslogik bauen auf der Prädikatenlogik erster Ordnung auf. Prinzipiell erscheint es heute, dass die Logik das für Wissen am besten geeignete analytische Werkzeug darstellt [222].

Anwendung in der Robotik. Ein bekanntes System stellt KnowRob dar, vergleiche die Arbeit [223]. KnowRob ist ein Framework zur Wissensverarbeitung, das vornehmlich für Haushaltsaufgaben entwickelt und evaluiert wurde. Das Framework arbeitet auf Basis von Logik, sodass sich Aufgaben durch Inferenz lösen lassen. Für die Implementierung wird Prolog verwendet. Die hierfür notwendige Wissensbasis wird aus verschiedenen Quellen extrahiert, wie etwa der Beobachtung menschlicher Akteure oder der Extraktion aus geeigneten Internetquellen.

2.3.5 Planungsverfahren

Die Planung dient dem Entwurf eines Handlungsvorganges, sodass ein gewünschter Zielzustand (möglichst optimal) erreicht werden kann. Ein Plan ist hierbei eine Darstellung, bei der eine oder mehrere Folgen von Aktionen entworfen werden, um von einem Anfangszustand, über eine Menge von Zwischenzuständen, zu einem Zielzustand zu gelangen [224]. Ein Plan sei allgemein gegeben durch ein Ziel, (sensorische) Umweltinformation über die Aufgabe und mögliche Systemaktionen. Durch die Planung wird es möglich, langfristige Ziele zu verfolgen sowie verfügbares Wissen zur Lösung einer Aufgabe sinnvoll einzusetzen. Bezüglich der Taxonomie des Planens bestehen nach [225] Unterschiede zwischen folgenden Klassen:

- **Lineares und nichtlineares Planen:** In linearen Plänen werden Aktionen in einer zeitlich determinierten Reihenfolge durchgeführt, während diese in nichtlinearen Plänen zeitlich nur partiell geordnet sind.
- **Monotones und nichtmonotones Planen:** In monotonen Plänen führt jede Aktion den Zustand in den Zielzustand über, während in nichtmonotonen Plänen eine Aktion den Zustand erst in Zwischenzustände überführt bis der Zielzustand erreicht wird.

Planen versteht sich als Grundlage der Systemautonomie und ist der Fachdomäne der künstlichen Intelligenz zuzuordnen. Für eine umfassende Übersicht sei auf [226], [144] verwiesen. Weitergehend werden die wesentlichen bekannten Methoden und Algorithmen dargestellt, welche zur Planung von Manipulationsaufgaben innerhalb der Robotik bekannt sind. Dabei werden die einzelnen Planungskomponenten, ausgehend von der obersten Planungsschicht, der Demontage- und Montageplanung über die Bewegungs- bis hin zur Greifplanung, analysiert, da diese für die gegebene Problemstellung von Bedeutung sind.

Demontage- und Montageplanung. Die Demontage- und Montageplanung befasst sich mit der Fragestellung, wie aus einer Baugruppe ein Bauteil oder eine Unterbaugruppe entfernt werden kann, oder wie sich aus einzelnen Bauteilen eine Baugruppe bilden lässt und welche Operationen und Bewegungen hierfür nötig und möglich sind. Auch entfallen hierunter Themen wie Ressourcenallokation, Arbeitszellengestaltung oder toleranzabhängige Fragen, wobei die Sequenzierung von (De-)Montageoperationen die wichtigste Aufgabe darstellt [227]. Die weiter aufgeführten Methoden werden unter anderem in den in 2.2 vorgestellten Gesamtsystemen eingesetzt und sollen folgend näher betrachtet und auf ihre Übertragbarkeit, hinsichtlich der automatisierten Instandhaltung, analysiert werden.

Damit automatisiert eine Folge an Montageschritten erzeugt werden kann, muss das Problem der geometrischen Separierbarkeit gelöst werden. Die meisten Ansätze arbeiten dabei auf der sogenannten „Assembly-by-Disassembly“-Strategie. Dies bedeutet, dass auch in der Montageplanung, die gesamte Baugruppe rückwärts in ihre Einzelteile zerlegt wird, weil dieses Vorgehen wesentlich effizienter ist, da sich weniger Kombinationsmöglichkeiten als bei einer Vorwärtssuche ergeben [228]. Damit ein geeigneter Plan erzeugt werden kann, spielt die gewählte Modellierung eine entscheidende Rolle, sowohl in der Generierung von (De-)Montagesequenzen, als auch in der Entwicklung eines intelligenten Planungsverfahrens [229] und soll weitergehend näher betrachtet werden. Darauf aufbauend werden Methoden und Verfahren zur Generierung von (De-)Montagesequenzen dargestellt. Umfassende Übersicht über die einzelnen Verfahren der (De-)Montageplanung bieten auch die Werke [230], [227], [231], [232].

Modellierung von Baugruppenstrukturen. Zur Modellierung von Verbindungen mechanischer Bauteile zu Baugruppen kommen für gewöhnlich Elemente aus der Graphentheorie zum Einsatz. Dabei weisen die einzelnen Modellierungsmethoden spezifische Vor- und Nachteile für die nachfolgende Generierung von Demontage- und Montagesequenzen auf, wobei nachfolgend die wichtigsten Modellierungsverfahren eingeführt werden.

Ein Verbindungs- oder Liaison-Graph ist ein ungerichteter Graph, in welchem die Knoten die Komponenten und die Kanten die Verbindungen repräsentieren [233]. Die Darstellung kann entweder als Adjazenzmatrix oder -liste erfolgen. Dabei existieren verschiedene Methoden den Raum des Graphs zu reduzieren, indem etwa die Verbindungskomponenten nicht als direkte

Komponenten betrachtet werden, sondern direkt die Kanten im Graph repräsentieren [233]. Eine ähnliche Methode wird in der Arbeit von [234] vorgestellt, in der mehrere Komponenten zu Unterbaugruppen zusammengefasst werden. Jedoch ist diese Methode, im Gegensatz zum klassischen Verbindungsgraph, schwer automatisiert zu bestimmen [235]. Ein weiterer Graph, welcher den Verbindungsgraph um die Information der bestehenden Freiheitsgrade zwischen den Komponenten erweitert, stellt der relationale Graph dar [108]. Durch sogenannte symbolisch, räumliche Relationen [236], [108], die die Kontaktinformation repräsentieren, werden die vorhandenen Freiheitsgrade bestimmt, wobei diese Information durch den Benutzer bereitgestellt werden muss. Die Kanten werden dann mittels symbolischer Relationen des Freiheitsgrads beschrieben. Oftmals werden noch weitere zusätzliche Informationen in der Baugruppen-modellierung hinzugefügt. So wird in [237] eine sinnvolle Gliederung in folgende Beschreibungs-elemente vorgeschlagen:

- **Geometrie:** Abbildung der geometrischen Zusammenhänge von Bauteilen, Baugruppen und des Produkts.
- **Position:** Abbildung der Position von Bauteilen und -gruppen und der Position von Kontakten (relativ/absolut).
- **Kontakt:** Abbildung des Kontaktzustands von zwei Oberflächen.
- **Verbindung:** Abbildung der Art der Verbindung eines Kontakts.
- **Zusammensetzung:** Abbildung des Zustands einer mittelbaren oder unmittelbaren Verbindung von zwei Bauteilen (Komposition).

Generierung von Montage- und Demontagesequenzen. Auf Grundlage der Beschreibung einer Baugruppe müssen weitergehend gültige Sequenzen gefunden werden. Dabei werden nach [108] folgende Bedingungen gefordert:

- **Geometrische Zugänglichkeit:** Keine Durchdringung der Bauteile während der Ausführung der Aufgabe.
- **Verbindbarkeit:** Es werden während einer Roboteraufgabe nur zwei Baugruppen gleichzeitig gefügt.
- **Baugruppenstabilität:** Eine Baugruppe ist stabil, wenn keine ungewollten Bewegungen der Bauteile während der Ausführung einer Aufgabe, bedingt durch Gravitation oder Fügekräfte, auftreten.

Die Bestimmung einer geeigneten Demontage- und Montagesequenz kann dabei beispielsweise nach folgenden Kriterien erfolgen [230]:

- **Minimaler Weg:** Kosten für die Bewegung zwischen den einzelnen Operationen, beispielsweise gemessen auf Zeitbasis.
- **Minimale Anzahl an Werkzeugwechseln:** Zeit die für das Wechseln eines Werkzeuges benötigt wird, wenn unterschiedliche Operationen ausgeführt werden müssen.
- **Minimale Zeit:** Gesamtzeit aus obigen und eventuell weiteren Faktoren.

Während, ausgehend von Einzelbauteilen in der Montage, Vorrangbedingungen beachtet werden müssen, sind es in der Demontage Blockierungsbedingungen. Dabei gilt, dass beide Prozesse umkehrbar sind, wenn alle Bauteile der Baugruppe Starrkörper darstellen und geometrische Einschränkungen getroffen sind [230].

Zur Modellierung von Demontage- und Montagesequenzen (DMS) werden für gewöhnlich gerichtete Graphen, Montagevorranggraphen, AND/OR-Graphen [238], Diamond Graphen [239] oder Petri-Netze [240], [241], verwendet. In [242] werden die unterschiedlichen Darstellungsformen, sowie deren Vor- und Nachteile, in Bezug auf Korrektheit und Vollständigkeit diskutiert. Abbildung 2-10 zeigt verschiedene Repräsentationsformen für DMS auf.

Allgemein fällt die Berechnung einer optimalen Demontagesequenz in die Klasse der NP-vollständigen Probleme [243], [244], wodurch eine hohe Anforderung an den Entwurf geeigneter Heuristiken für die Sequenzierung gestellt werden muss [106]. Die kombinatorische Komplexität der Zerlegung, in Abhängigkeit der Verbundenheit der Baugruppe, wird in [106] analysiert. So kann für eine starke Verbundenheit (jedes Bauteil ist mit allen anderen verbunden) mit n-Bauteilen, die maximale Anzahl an Zerlegungen Z_{max} mit Gleichung (2.5) angegeben werden.

$$Z_{max} = \frac{3^n + 1}{2} - 2^n. \tag{2.5}$$

Für schwach verbundene Baugruppen (jedes Bauteil hat maximal nur zwei Verbindungen) wird in der Arbeit die Zerlegungsanzahl mit (2.6) bestimmt.

$$Z_{min} = \frac{n(n+1)(n-1)}{6}. \tag{2.6}$$

Damit ist immer eine Zerlegungsanzahl $Z_{min} \leq Z \leq Z_{max}$ erforderlich. Zur Generierung einer optimalen Sequenz muss vorab das Problem der geometrischen Separierbarkeit gelöst werden. Während erste Planungsverfahren eine hohe Benutzerinteraktion in Form von Fragen erforderten [245], wurden automatisierte Verfahren auf Basis geometrischer Kontaktinformation entwickelt, um Bewegungsfreiheitsgrade zur Separation der einzelnen Komponenten zu bestimmen.

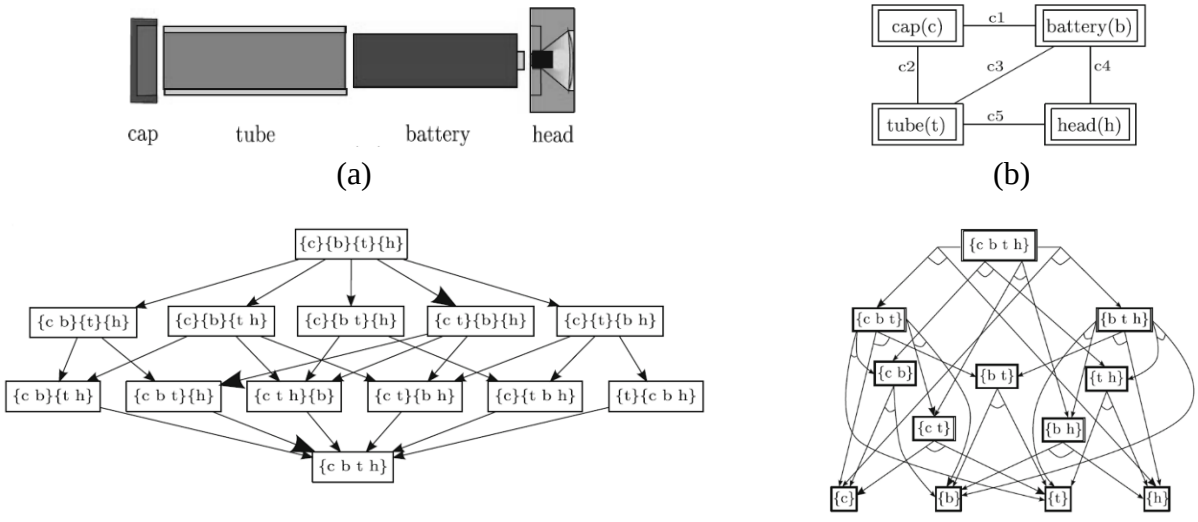


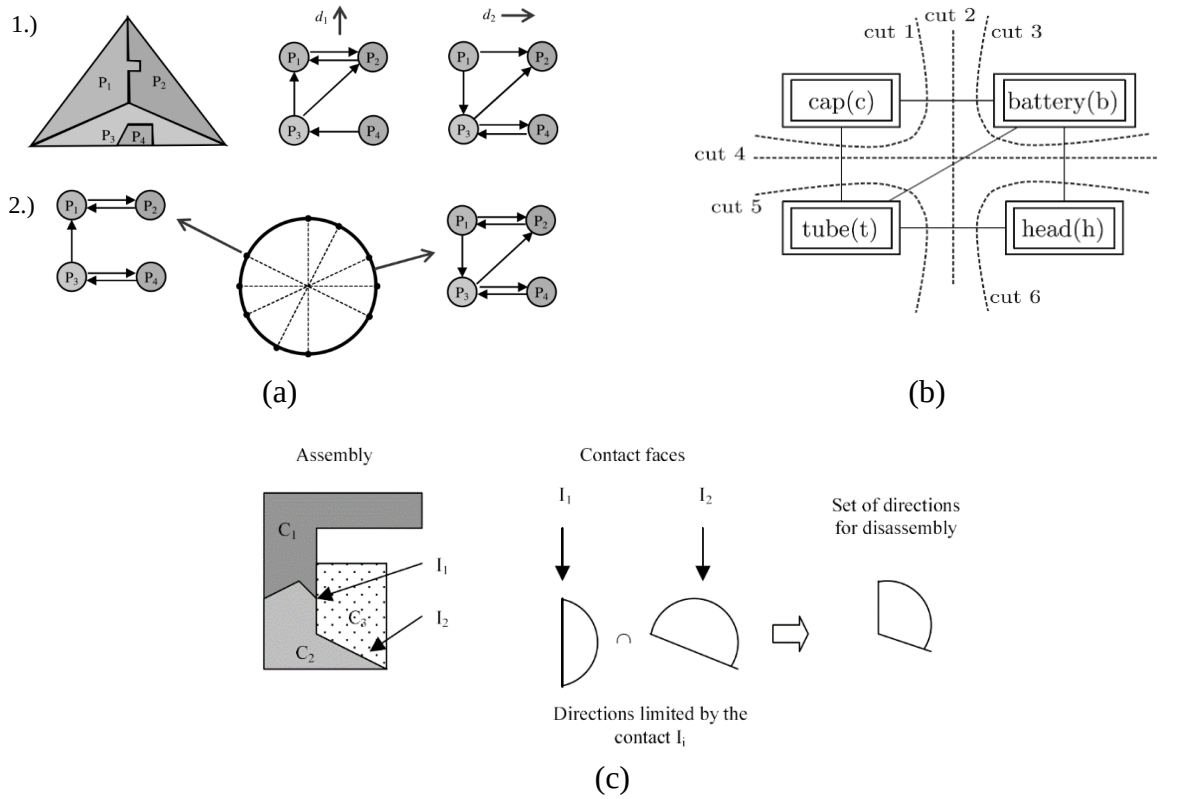
Abbildung 2-10: Beispiele unterschiedlicher Baugruppenrepräsentationsformen und Sequenzierungsverfahren (aus [227] © Springer 2013 modifiziert nach [242], [239], [246]): Demontagedarstellung einer Taschenlampe (a); Korrespondierender Verbindungsgraph (b); Sequenzierungsbeschreibung mittels gerichtetem Graph (c); Sequenzierungsbeschreibung mittels AND/OR-Graph (d)

Ein Beispiel stellt der in [247] eingeführte Directional Blocking Graph (DBG) und Non-Directional Blocking Graph (NDBG) dar. Dabei wird für jedes Bauteil eine translatorische Richtungsvorgabe getroffen und überprüft, ob sich diese in vorgegebener Richtung blockieren. Das Ergebnis wird in einem gerichteten Graph dem DBG abgespeichert, siehe Abbildung 2-11 (a). Der NDBG bildet eine Sphäre, mit diskret aufgeteilten Regionen, wobei jede Region durch einen DBG repräsentiert wird [248]. Ähnliche Verfahren stellen die Disassembly Precedence Matrix (DPM) [243], [240], [249] oder der Precedence Graph [250] dar. Beispiele zur Erzeugung der DPM werden im 2D-Raum verdeutlicht, wobei die freie Bewegungsrichtung immer nur in einer Koordinate des Baugruppensystems erfolgen kann. Eine Beschreibung bei unterschiedlicher Orientierung von Baugruppen- zu Komponentenkoordinatensystem wird nicht aufgezeigt. Auch können nicht alle möglichen Bewegungsfreiheitsgrade berechnet werden.

In [251] wird die sogenannte Cut-Set Methode eingeführt, welche die Bestimmung der Reihenfolge auf Basis des Verbindungsgraphs erlaubt. Die Methode arbeitet dabei nach der Assembly-by-Disassembly Strategie. Der Verbindungsgraph wird in alle möglichen Teilgraphen zerlegt und das Ergebnis in einem AND/OR-Graph, Abbildung 2-10 (d), gespeichert, vergleiche Abbildung 2-11 (b). Zur Entscheidung, welche Zerlegung möglich ist, wird ein relationales Baugruppenmodell verwendet, wobei dieses nicht automatisch generiert wird. Die Arbeiten [89], [92] nutzen CAD-Daten und erstellen aus dieser Information einen Kontaktgraph. Dabei werden die einzelnen Bauteile als Polyeder modelliert. Jeder Fläche wird ein Knoten zugewiesen und zwischen den einzelnen Flächen überprüft, ob ein Kontakt besteht. Ist dies der Fall, werden die Knoten über eine Kante verbunden. Um eine Menge an freien Bewegungen zu generieren, werden die einzelnen Kontakte als Sphäre beschrieben und mittels Durchschnitt zwischen einzelnen Kontaktpunkten die Menge aller freien Bewegungen erzeugt. Abbildung 2-11 (c) zeigt das Vorgehen. Die möglichen Bewegungen werden in einem diskreten Histogramm aufgeteilt und bewertet. Die Bewegung mit der geringsten Kollisionswahrscheinlichkeit wird weitergehend zur Separierung verwendet. Das Verfahren wird nur im zweidimensionalen Raum aufgezeigt. Eine praktikable Berechnungsvorschrift, zur Bestimmung der Schnittmenge, die den gesamten Demontageraum abbildet, wird nicht angegeben.

^{High}LAP [107], [108], [252], [253] verwendet als Eingabeinformation ebenso CAD-Daten sowie symbolisch, räumliche Relationen [236], [108], welche vom Benutzer angegeben werden. Die symbolisch, räumlichen Relationen $\mathcal{ssr}$ beschreiben die relative Lage der Bauteile zueinander und werden an den Features (hier Körpermerkmale) $\mathcal{F}$ jedes Bauteils definiert. Die in [236] eingeführten und in [108] erweiterten Features sind mit $\mathcal{F} \in \{Face, Shaft, Hole\}$ und die symbolisch, räumlichen Relationen zu $\mathcal{ssr} \in \{against, coplanar, aligned, \dots\}$ definiert. Dies ermöglicht den Aufbau eines relationalen Baugruppenmodells, welches gegenüber [254], sowohl geometrische als auch physikalische Randbedingungen berücksichtigt. Zur Berechnung des relationalen Graphs wird der sogenannte Extended Cycle Finder [255] verwendet, welcher auf der Idee von [256] aufbaut. Dabei werden Zyklen von Freiheitsgradrelationen $\mathcal{sdof} \in \{fix, lin, rot, agpp, fits\}$ analysiert und durch ein Regelwerk in eine neue Relation transformiert. Dies wird solange durchgeführt, bis jede Kante von Bauteil C_i nur noch eine $\mathcal{sdof}$-Relation zu C_j besitzt. Zur Zerlegung des relationalen Graphs wird die Cut-Set Methode [251], unter Berücksichtigung verschiedener Zwangsbedingungen, verwendet. Das Ergebnis wird weitergehend

in einem AND/OR-Graph gespeichert. Die Prüfung der geometrischen Zugänglichkeit erfolgt auf Basis von Departräumen nach [254], wobei in [106] gezeigt wird, dass das Verfahren nicht vollständig ist und Fehler entstehen können, da nicht alle Demontagerichtungen erkannt werden. Der generierte AND/OR-Graph wird anschließend nach verschiedenen Kriterien bewertet und daraus der Roboteraufgabenplan, in Form von Aktionsprimitiven, bestimmt. Ein Nachteil der direkten Zerlegung in Aktionsprimitive liegt nach [108] auch darin, dass nicht jede Roboteraufgabe vollautomatisch in Aktionsprimitive zerlegt werden kann, sondern nur in vorab spezifizierte Aufgaben. Die Arbeit verwendet keine allgemeingültigen Roboteraktionen, sondern zerlegt diese in spezifische Aufgaben, wie „Stift-In-Loch“, „Verschraubung“ oder „Objektablage“. Das System wird in den Arbeiten [257], [106], [109] erweitert. Durch einen effizienteren Algorithmus zur Lösung der geometrischen Separierbarkeit, werden die Konfigurationsraumhindernisse der einzelnen Bauteile ermittelt und durch stereographische Projektion auf einen umschreibenden Kreis geometrisch mögliche Fügerichtungen bestimmt [257], [106]. Auch werden probabilistische Bahnplaner untersucht, welche für Baugruppen verwendet werden, die keinen translatorischen Fügepfad besitzen. Dabei wird nicht das Zusammenspiel zwischen Roboter und Bauteilen betrachtet. Die Berechnung einer geeigneten Montagesequenz erfolgt ebenfalls auf Basis eines AND/OR-Graphs mittels Graphensuche. Die Berechnungszeiten für die Vorverarbeitung und Planung einer Montagesequenz beträgt für die Baugruppen durchschnittlich 2.3 Minuten/Bauteil.



(a) (b) (c)

Abbildung 2-11: Möglichkeiten zur Berechnung der geometrischen Separierbarkeit: 1.) Ein Bauteil mit seinen beiden DBG für die Bewegungsrichtungen d_1 und d_2. 2.) der NDBG des Bauteils aus [230] © Elsevier 2015 modifiziert nach [247]) (a); Darstellung der Cut-Set-Methode für das Taschenlampenbeispiel aus Abbildung 2-10 (aus [227] © Springer 2013 modifiziert nach [251]) (b); Berechnung der Separierbarkeit auf Basis von Kontaktinformationen (aus [89]) © SPIE 2001 (c)

Bewegungsplanung. Die Bewegungsplanung umfasst allgemein die Bahn- und die Trajektorienplanung. Die Bahnplanung dient der Bestimmung einer kollisionsfreien, geometrischen Bahn, von einem Start- zu einem Zielpunkt, wobei die Umgebung sowohl statisch als auch dynamisch sein kann. Die Trajektorienplanung hingegen berücksichtigt den zeitlichen Bezug, dient also der Planung der Bahngeschwindigkeit, der Bahnbeschleunigung und eventuell des Bahnrucks. Die Planung einer kollisionsfreien Bahn des TCP von Start- zu Zielpunkt kann nach verschiedenen ausgewählten Optimierungskriterien, wie beispielsweise einer weg- oder energieoptimalen Bahn, welche durch ein Kostenfunktional definiert wird, optimiert werden. Neben diesen Kriterien lassen sich weitere Einschränkungen, wie etwa der Arbeitsraum des Roboters, in die Aufgabe miteinbeziehen. Für gewöhnlich wird die Bahnplanungsaufgabe im Konfigurationsraum $\mathcal{C}$ durchgeführt, da dadurch die räumliche Ausdehnung des Roboters nicht weiter betrachtet werden muss. Man unterscheidet grundsätzlich zwischen lokalen, globalen und hybriden Verfahren zur Bahnplanung. Generell sind globale Verfahren sehr gut für die offline Planung und lokale Methoden sehr gut für die online Planung geeignet [258].

Folgender Abschnitt stellt zusammengefasst die wesentlichen Konzepte vor, wobei der Fokus auf Bahnplanungsverfahren gelegt wird, welche im Bereich von Manipulationsaufgaben in der Robotik eingesetzt werden. Die Bewegungsplanung weist in der Robotik bereits eine lange Tradition auf, weshalb für eine umfassende Übersicht auf die Werke [259], [260], [114], [113], [261] oder für eine erklärende Übersicht auf [262], [263] verwiesen sei.

Lokale Bahnplanungsverfahren. Lokale Bahnplanungsverfahren eignen sich vor allem für reaktive und echtzeitfähige Ansätze, in denen die Umgebung nicht bekannt ist oder sich während der Bewegungsausführung ändert. Somit können diese, vor allem in dynamischen Umgebungen, für die online Planung eingesetzt werden, da ihre Berechnungskomplexität im Allgemeinen gering ist. Jedoch besteht das grundsätzliche Problem, dass eine Optimalität der globalen Bahn nicht gegeben ist sowie die Konvergenz in einem lokalen Minimum möglich ist. Weitergehend sollen explizite lokale Verfahren aufgezeigt werden.

Eines der bekanntesten Verfahren zur lokalen Bahnplanung stellt die Potentialfeldmethode dar, welche in der Arbeit [264] erstmals dargestellt ist. Seitdem wurde das Verfahren kontinuierlich weiterentwickelt, um die Nachteile, wie die Konvergenz in lokalen Minimas, zu vermindern oder auf neue Anwendung wie die sichere Mensch-Roboter-Interaktion zu adaptieren [265], [258]. Die Grundidee besteht darin, dass der Roboter einem virtuellen Potentialfeld ausgesetzt ist, wobei Hindernisse eine abstoßende und das Ziel eine anziehende Wirkung auf den Roboter haben. Aus dem resultierenden Potentialfeld wird dann mittels Gradientenabstiegsverfahren die resultierende Kraft bestimmt, welche die Richtung und die Geschwindigkeit für den Roboter beschreibt, vergleiche hierzu Abbildung 2-12 (a).

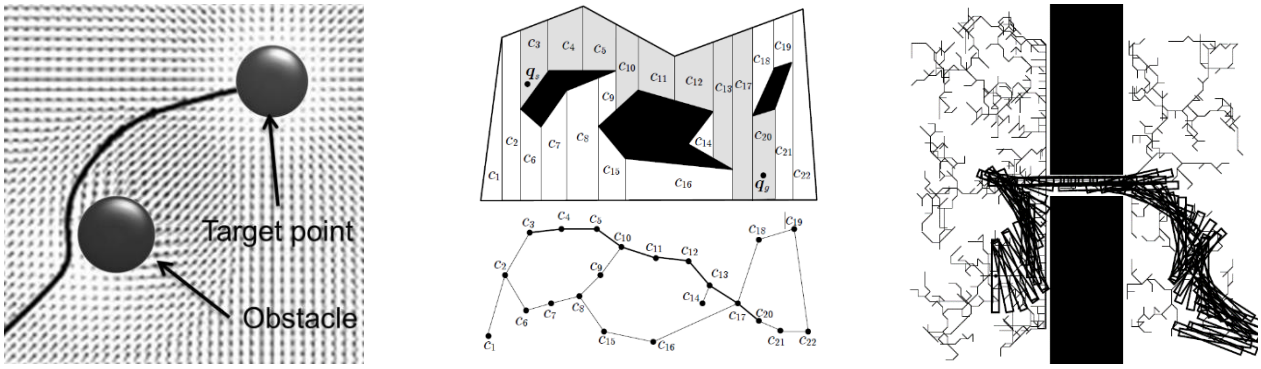
In [266] wird ein ähnliches Verfahren vorgestellt. Das Vector Field Histogram (VFH) geht von einem polaren Histogramm aus, welches die Häufigkeit der auftretenden Hindernisse (Hindernisdichte) bewertet, welche als zweidimensionale Belegtheitskarte erfasst wird. Die resultierende Bahn stellt dann diejenige mit der geringsten Dichte dar. Auf Basis von VFH wurden Erweiterungen wie VFH+ [267] und VFH* [268] entwickelt. Ein weiteres lokales Verfahren, das auf dynamischen Grundgleichungen basiert, stellt der Dynamic Window Ansatz [269], [270] dar. Dabei wird unter Beachtung der maximal möglichen Roboter- und Hindernisdynamik ein lokal

optimaler Geschwindigkeitssollwert bestimmt, sodass eine kollisionsfreie Bewegung im nächsten Geschwindigkeitsintervall, dem sogenannten Dynamic Window, möglich ist.

Globale Bahnplanungsverfahren. Globale Bahnplaner benötigen a priori das gesamte Umgebungsmodell, in welchem der Roboter die Aufgabe durchführen soll. Dies ermöglicht die komplette Berechnung der Bahn vorab, hat jedoch das Problem, dass diese nicht in einem dynamischen Umfeld eingesetzt werden können. Auch ist die Berechnungskomplexität der Verfahren in der Regel höher. Ein einfacher Vertreter dieser Klasse stellt etwa die Zellenzerlungsstrategie dar, siehe hierzu etwa [259], [114]. Der Raum wird hierbei zerteilt in freie Zellen (hier Trapeze) und Polygone, die ein Hindernis repräsentieren. Die freien Zellen lassen sich dann zu einem Graph verbinden, sodass die Planungsaufgabe ein Suchproblem darstellt, siehe Abbildung 2-12 (b). Weitere Verfahren verwenden Sichtbarkeitsgraphen oder Voronoi-Diagramme, welche das Umgebungsmodell für eine Graph-Suche geeignet aufbereiten oder heuristische Ansätze wie etwa genetische Algorithmen [271].

Für höherdimensionale Probleme werden oftmals stichprobenbasierte Verfahren wie Probabilistic Roadmaps (PRM) oder Rapidly-Exploring Random Tree (RRT) eingesetzt. Die RRT-Algorithmen [272], [273], [274] generieren dabei, ausgehend von einem Startpunkt (Wurzel), einen Baum, bis dieser den Zielpunkt erreicht, vergleiche Abbildung 2-12 (c). Die Knoten des Baumes werden dabei in jedem Expansionsschritt zufallsbedingt gewählt, wobei die Planungsaufgabe für gewöhnlich im Konfigurationsraum des Roboters durchgeführt wird. Es bestehen zahlreiche Erweiterungen des Elementaralgorithmus, welche verschiedene Probleme adressieren und auch online einsetzbar sind [275], [276]. Bei diesen stichprobenbasierten Algorithmen, welche auf dem Zufallsprinzip beruhen, ist sichergestellt, dass diese bei einer infiniten Anzahl an Stichproben eine Lösung finden, wenn diese physikalisch existiert (*engl.* „*probabilistically complete*“) [260].

Hybride Bahnplanungsverfahren. Hybride Bahnplanungsverfahren verknüpfen die Vorteile lokaler und globaler Planer. So können reaktive Strategien in die Planungsaufgabe integriert und lokale Minimas vermieden werden, wie dies beispielsweise in dem hybriden Framework nach [277] aufgezeigt wird. Die Grundidee besteht darin, eine globale Bahn zu planen und diese beim Aufkommen eines Hindernisses lokal zu verformen, wie dies bei den Elastic Bands [278], [277] oder der Kurvenflussmethode [279] zur Anwendung kommt.



(a) (b) (c)

Abbildung 2-12: Übersicht bekannter Bahnplanungsverfahren: Potentialfeldmethode [258] (a); Zellenzerlegung [114] © Springer 2009 (b); Rapidly Exploring Random Tree [272] (c)

Greifplanung. Zur Durchführung einer autonomen Manipulation gehört, neben der Berechnung einer kollisionsfreien Bahn der Roboterkinematik, ebenso die automatische Bestimmung eines geeigneten Objektgriffs. Die Planung einer geeigneten Greifkonfiguration variiert stark anhand Objekt- und Greifereigenschaften, wobei Griffstabilität und -qualität im Vordergrund stehen. Untergliedert werden Griffe prinzipiell gemäß der Taxonomie [280] in Kraft- und Präzisionsgriffe. Neben Ansätzen, optimale Griffe durch Vormachen zu bestimmen [281], basieren die im Bereich autonomer Robotersysteme eingesetzten Greifplaner für gewöhnlich auf modellbasierten Methoden. Die Stabilität eines Griffes wird dabei über ein Kontaktmodell bestimmt und bewertet. Zur Berechnung geeigneter Griffe stehen bereits leistungsstarke Simulationsumgebungen wie GraspIt! [282] zur Verfügung. Für praktische Anwendungen werden oftmals empirische Ansätze verwendet, da diese gegenüber analytischen häufig robuster arbeiten [283], wobei zur Kompensation von Planungsunsicherheiten reaktive Ansätze in die Greifplanung integriert werden [284]. Auch geht der Trend in die Kombination von Greif- und Bewegungsplanung, wie dies beispielsweise in [285] vorgeschlagen wird. Die meisten Verfahren sind jedoch zum Großteil für Mehrfingergreifer, wie [286], konzipiert. Untersuchungen haben allerdings ergeben, dass industriell ein Großteil aller Objekte bereits mit drei Grundgriffen abgedeckt werden kann, welche den Drei-Finger-Parallel-, den Drei-Finger-Zentrisch- und den Zwei-Finger-Griff umfassen [287]. Die Greifkonfiguration lässt sich für vordefinierte Objektklassen auch experimentell bestimmen. Ebenso konnte gezeigt werden, dass zufällig generierte Greifpunkte zu praktikablen Lösungen führen [288].

2.3.6 Regelungsmethoden

Für die in der Robotik verwendeten Regelungsmethoden, wird gemäß der Darstellung der Perzeption in 2.3.2, auch in propriozeptive und exterozeptive Verfahren unterteilt, vergleiche hierzu auch [289]. Die propriozeptive Regelung stellt hierbei die kinematische und die exterozeptive Regelung die verarbeitende Kontrolle über Umweltinteraktionen dar. Auch existieren Mischformen, sodass externe Sensoren eine direkte Beeinflussung der propriozeptiven Regelung zulassen. Dabei gilt, dass die einzelnen Regelungen auch anhand ihrer physikalischen Struktur verkoppelt sind, wodurch eine geeignete Abstimmung der einzelnen Teilregelungen auf das Gesamtsystem besonderer Bedeutung zukommt. Während die kinematische Regelung die Bahngenauigkeit sowie das Bahnfolgeverhalten sicherstellen, können nur durch den Einsatz exterozeptiver Verfahren, komplexe und a priori unbekannte Aufgaben mit direkter Umweltinteraktion gelöst werden. So können durch eine Kraft-Momenten-Regelung Bearbeitungskräfte definiert eingestellt und durch eine bildbasierte Regelung Bauteile mit einer vorab undefinierten Position oder Geschwindigkeit sicher manipuliert werden. Folgend sollen die einzelnen, bekannten Regelverfahren dargestellt werden.

Propriozeptive Regelung. Die propriozeptive oder interne Regelung dient der Kontrolle und der Anpassung der internen Zustände des Robotersystems. Als interne Zustände stehen für gewöhnlich die kinematischen Aktorgrößen (Position, Geschwindigkeit) sowie der Aktorstrom (oder inneres Aktormoment) zur Verfügung. Man unterscheidet prinzipiell zwischen kartesischer *(engl. „Operational Space Control/ Task Space Control“)* und achsbasierter Regelung *(engl. „Joint Space Control“)* [114]. Zur kartesischen Regelung verarbeitet der Regler direkt die kartesische Pose, wobei die Berechnung der Winkelgeschwindigkeiten der einzelnen Achsen, etwa über die

(pseudo)inverse Jacobi-Matrix des Manipulators erfolgen kann. Achsbasierte Regelungen verwenden als Regelgrößen die Achswerte des Manipulators, wobei die Achssollwerte steuerungsbasiert aus der kartesischen Pose über die inverse Kinematik bestimmt werden. Hierbei unterscheidet man zusätzlich zwischen dezentralen und zentralen Regelungskonzepten.

Dezentrale Strukturen stellen dabei die klassisch eingesetzte Variante in gängigen Bewegungssteuerungen dar [290]. Dabei wird jede Achse unabhängig voneinander in einer kaskadierten Regelung mit kinematischer Vorsteuerung und P-Positions-, PI-Geschwindigkeits- sowie PI-Stromregler betrieben, siehe beispielsweise [291]. Vorteilhaft an dieser Struktur sind vor allem die hohe Robustheit sowie die einfache Inbetriebnahme durch die Parametrierung der Reglerparameter, welche mit praktischen Verfahren gut möglich ist. Nachteilhaft ist jedoch, dass die einzelnen Achsen nicht voneinander entkoppelt werden und sich somit Bewegungskräfte und -momente als Störungen auf die separiert betrachteten Achsregler auswirken. Auch wird durch die kaskadierte Struktur ein Verlust der Bandbreite in Kauf genommen.

Zentrale Strukturen verwenden zur Regelung das Modell der inversen Dynamik. Diese dient dabei als inverse Vorsteuerung des Manipulators, da aus einer gegebenen Achssollposition direkt die nötigen Achssollmomente generiert werden können. Ein zweiter Freiheitsgrad mit Regler dient zur Kompensation von Modellunsicherheiten und Störungen. In der Robotik fallen diese Ansätze unter den Begriff „*Computed Torque Control“* und werden in der Regelungstechnik dem Gebiet der flachheitsbasierten Verfahren zugeordnet, siehe hierzu [292], [293]. Der Vorteil der Entkopplung wird jedoch durch den Nachteil erkauft, dass ein Modell des Roboters vorliegen und online mitgerechnet werden muss. Auch ist es nicht realistisch, dass in der Praxis alle Systemparameter des Manipulators vorliegen, was zusätzliche Schwierigkeiten verursacht und zu erweiterten adaptiven Ansätzen führt [294].

Exterozeptive Regelung. Die exterozeptive Regelung kann auch als Aufgaben- oder Prozessregelung aufgefasst werden. Sie dient folglich der Regelung von Interaktionen des Robotersystems mit seiner Umwelt. Hierdurch kann das Robotersystem in einer unbekannten Umgebung flexibel agieren, durch exterozeptive Sensorik Umweltzustände erfassen und durch den gezielten Einsatz von Regelstrategien geeignet reagieren. Folgend sollen die bekannten Regelungskonzepte für die Kraft-Momenten-Regelung sowie für die visuelle Regelung diskutiert werden. Abschließend sollen Arbeiten aufgezeigt werden, die die Regelungskonzepte in einem gemeinsamen Framework zur hybriden Regelung betrachten.

Kraft-Momenten-Regelung. Diese Regleransätze werden in der Robotik bereits seit den 1970er Jahren untersucht [295]. Die Anwendungsfälle sind dabei besonders vielfältig und reichen von Bearbeitungsaufgaben, wie dem Schleifen, über Montageaufgaben, der physikalischen Mensch-Roboter-Interaktion [296], [297] bis hin zur kooperierenden Werkstückhandhabung [298]. Man unterscheidet bei den Regelungsansätzen zwischen den Gruppen direkter und indirekter Kraft-Momenten-Regelung (KMR). Bei der indirekten KMR werden die Kräfte und Momente über die Bewegungsregelung eingestellt, während bei der direkten KMR direkt die Sollkraft geregelt wird [299]. Die Gewinnung der Istkräfte und -momente erfolgt für gewöhnlich durch einen am TCP angebrachten Kraft-Momenten-Sensor, vergleiche 2.3.2. Es existieren jedoch auch Ansätze, welche auf Basis der gemessenen Antriebsströme den Einsatz ohne kostspielige Sensorik ermöglichen [296]. Einen Überblick geben die Standardwerke [299], [114], [113].

Indirekte Kraft-Momenten-Regelung. Die indirekte KMR basiert auf der Vorgabe des Nachgiebigkeitsverhaltens des Robotersystems. Hierbei wird wiederum zwischen passiven und aktiven Systemen unterschieden. Passive Systeme basieren auf einer rein mechanisch wirkenden Nachgiebigkeit, welche beispielsweise auf Basis von Federelementen erzeugt wird. Dabei wird das mechanische Nachgiebigkeitssystem *(engl. „Remote Center of Compliance")*, vergleiche [300], am Endeffektor des Robotersystems angebracht. Vorteilhaft sind vor allem die einfache Realisierbarkeit, deren Kosteneffizienz sowie die hohe Systemdynamik. Jedoch ist der Ansatz nicht sonderlich effizient, da für jeden Anwendungsfall ein spezielles Nachgiebigkeitssystem eingesetzt werden muss [299]. Aktive Systeme stellen das Nachgiebigkeitsverhalten auf Basis der Systemdynamik der geregelten Roboterkinematik ein. Hierzu wird der Roboterkinematik das Systemverhalten eines Feder-Masse-Systems durch die geeignete Wahl der Reglerdynamik aufgezwungen. Zu diesen Regelverfahren gehören die Impedanz- und Admittanzregelung. Die Impedanzregelung stellt eine der am verbreitetsten eingesetzten KMR-Regler dar und wurde bereits in den frühen Arbeiten [301], [302], [303] untersucht. Ausgangslage stellt das Übertragungsverhalten des Systems als mechanische Impedanz (2.7) dar.

$$Z(s) = \frac{f_{e,i}(s)}{\dot{p}_i(s)}, mit\ s = \sigma + j \cdot \omega \tag{2.7}$$

Dabei beschreibt $f_{e,i}$ die resultierende Kraft auf Basis der Geschwindigkeit $\dot{p}_i$ (wird die Position verwendet spricht man auch von der verallgemeinerten Impedanz [296]) in Richtung der Koordinate i. Das Übertragungsverhalten für den kartesischen Raum wird nun durch die Wahl einer linear, zeitinvarianten Differentialgleichung 2. Ordnung gemäß (2.8) abgebildet.

$$\underline{f_e}(t) = \underline{M} \cdot \ddot{\underline{p}}(t) + \underline{D} \cdot \dot{\underline{p}}(t) + \underline{C} \cdot \underline{p}(t)\ mit\ \underline{M}, \underline{D}, \underline{D} \in \mathbb{R}^{6\times6}\ , \underline{f_e}, \underline{p}, \dot{\underline{p}}, \ddot{\underline{p}} \in \mathbb{R}^6\ . \tag{2.8}$$

Durch die Wahl der Parametermatrizen $\underline{M}$, $\underline{D}$ und $\underline{C}$ kann nun ein prinzipiell beliebiges Verhalten, sowie Sonderfälle wie die Steifigkeitsregelung für $\underline{M} = \underline{D} = 0;\ \underline{C} \neq 0$ oder die Dämpfungsregelung für $\underline{M} = \underline{C} = 0;\ \underline{D} \neq 0$ eingestellt werden [295]. In [114], [299] wird ein Impedanzregelgesetz eingeführt, welches mithilfe der inversen Dynamik $\mathcal{ID}$ die Stellmomente für die Antriebe generiert. Die kartesisch anliegenden Kräfte und Momente werden über die statische Abbildung durch die Jacobi-Matrix in den Achsraum transformiert. Durch die Substitution der klassischen Positionsregelung, durch den Impedanzregler, kann prinzipiell eine höhere Dynamik erreicht werden. Neben der Voraussetzung eines hinreichend genauen inversen Dynamikmodells muss die Impedanzregelung Störungen und Nichtlinearitäten der Positions- und Kraftregelaufgabe gleichzeitig kompensieren. Auch besteht in heutigen Robotersteuerungen nicht immer der Zugriff auf eine Strom- beziehungsweise Momentenschnittstelle.

Um eine separierte Betrachtung von Positions- und Kraftregelaufgabe zu erreichen, kann eine Admittanzregelung eingesetzt werden [299]. Diese kann als Sonderfall der indirekten KMR betrachtet werden. Die Admittanz stellt das inverse Übertragungsverhalten der Impedanz dar und kann folglich mit Gleichung (2.9) angegeben werden. Abbildung 2-13 zeigt die prinzipielle Struktur einer Admittanzregelung mit unterlagerter Lageregelung auf.

$$A(s) = \frac{\dot{p}_i(s)}{f_{e,i}(s)}. \tag{2.9}$$

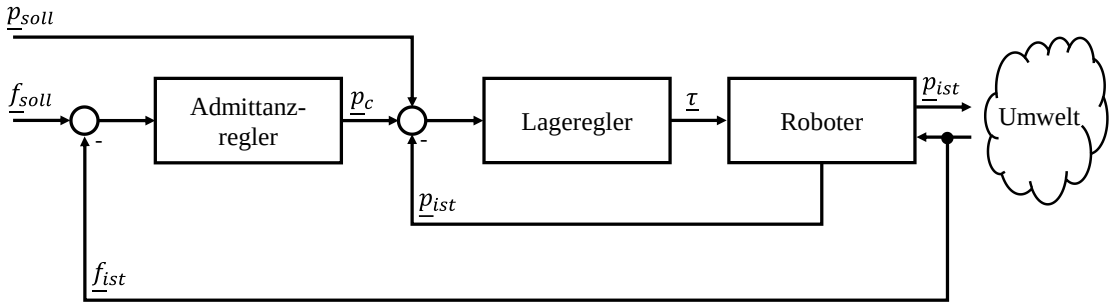


Abbildung 2-13: Struktur einer Admittanzregelung mit zusätzlichem Bahnfreiheitsgrad

Dabei gibt der Admittanzregler (hier) einen „nachgiebigen“ (*engl. „compliant“*) Positionseintrag $\underline{p}_c$ vor, welcher aus der Verschiebung der einwirkenden Kräfte und Momente nach (2.8) bestimmt wird. Durch die unterlagerte Lageregelung kann eine hohe Bahngenauigkeit erreicht werden, falls keine Kontaktkräfte auf das System einwirken.

Direkte Kraft-Momenten-Regelung. Reglerstrukturen, die die direkte KMR erlauben, lassen sich im Wesentlichen in die parallele und hybride KMR unterteilen. Eine der ersten bekannten Arbeiten zur parallelen KMR wird in [304] aufgeführt. Dabei haben sowohl Lageregelung, als auch KMR, gemeinsamen Eintrag an der Steuergröße des Roboters. In Abhängigkeit ob es sich um eine freie Bewegung oder einen Kontaktfall handelt, kann die Verstärkung des Steuereingriffs zwischen beiden Reglern variiert werden. Für die Auslegung der Regler können verschiedene Reglerdynamiken verwendet werden. Die hybride KMR basiert auf der Theorie von [149] und wurde erstmals umfassend in der Arbeit [305] diskutiert. Die Idee besteht darin, den Aufgabenraum in einen kraftgeregelten und einen positionsgeregelten, voneinander unabhängigen Unterraum zu teilen. Dabei wird über die Selektionsmatrizen $\underline{S}_{Lage}$ und $\underline{S}_{KMR}$ die Aktivität gesteuert, wobei immer $\underline{S}_{Lage} + \underline{S}_{KMR} = \underline{I}$, mit $\underline{I} = \mathrm{diag}(1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1)$ gilt.

Zusammenfassend gilt, dass verschiedene Reglerstrukturen für die KMR bereits ausführlich untersucht wurden. Prinzipiell haben Impedanz- und Admittanzregler den Vorteil, dass sie nicht zwischen zwei Zuständen (kein Kontakt und Kontakt) umschalten müssen. Jedoch erlaubt die Struktur keine gleichzeitige, präzise Positions- und Kraftregelung, für welche die direkten Ansätze besser geeignet sind [306]. Auf der anderen Seite muss für die hybride KMR eine explizite geometrische Beschreibung des Zielobjekts vorhanden sein, sodass der Regler in beschränkte und unbeschränkte Räume aufgeteilt werden kann.

Visuelle Regelung. Es werden bildgebende Sensoren verwendet (2D, 2.5D, 3D), aus welchen, nach der Datenaufbereitung die Steuerbefehle für die Kinematik über das visuelle Regelungsgesetz gebildet werden. Durch die Extraktion der Zielkoordinaten aus der Sensorinformation kann die Bahn für die Roboterkinematik prinzipiell gesteuert ausgeführt werden. Jedoch hat dies den Nachteil, dass die Positioniergenauigkeit direkt von der Güte der Sensorik und der Genauigkeit der kinematischen Kette des Roboters abhängt [307]. Eine Alternative stellt die Rückkopplung der abgeleiteten Steuergröße dar. Für diese Ansätze hat sich der Term „Visual Servoing“, bildbasierte Regelung, etabliert, welcher bereits 1979 in [308] eingeführt wurde. Die Gewinnung der Regelgröße ist allerdings aufwendig, da diese erst aus der Bildinformation extrahiert werden muss. Für gewöhnlich sind nach der Bildaufnahme die Schritte Vorverabeitung, Segmentierung und

Klassifikation nötig, wodurch leistungsfähige Algorithmen sowie eine hohe Rechengeschwindigkeit des Steuerungssystems benötigt werden. Nachteilig ist hierbei, dass durch die komplexe Bestimmung der Regelgröße relativ große Totzeiten auftreten, welche die Stabilität und die erreichbare Dynamik des Regelkreises maßgeblich beeinträchtigen [150]. Die Sensorik wird für gewöhnlich am TCP des Roboters integriert, wobei auch eine stationäre Anbringung des Sensors möglich ist [309]. Der prinzipielle Aufbau einer Visual Servoing Architektur lässt sich, gemäß [310] in vier grundlegende Strukturen einteilen. Man unterscheidet, ob die Ansteuerung des Roboters direkt oder über eine unterlagerte Lageregelung erfolgt und ob das Fehlersignal im Bild- oder im Arbeitsraum des Roboters bestimmt wird. Aufgrund der hohen Rechenzeit für die Bildverarbeitungsoperationen ist eine direkte Regelung der Kinematik (im Achsraum), wegen der ausgeprägten Nichtlinearitäten, äußerst komplex und erschwert sich durch die Singularitäten der kinematischen Kette, da diese nicht separiert betrachtet werden können [307]. Dies hat zur Folge, dass für gewöhnlich eine Regelungsarchitektur, mit einer unterlagerten Achs- beispielsweise Lageregelung, ausgeführt wird [311]. Weitergehend werden die relevanten Funktionsprinzipien und Arbeiten aufgezeigt.

Positionsbasierte visuelle Regelung (PBVS). Die Sollwertvorgabe für das PBVS Regelungsproblem entspricht einer metrischen Zielposition für den TCP des Roboters. Hierzu muss entweder Sensorik eingesetzt werden, welche eine direkte kartesische Positionsmessung des Ziels erlaubt, oder diese muss aus der zur Verfügung stehenden Bildinformation bestimmt werden. Zur Posenschätzung stehen zwar leistungsfähige Algorithmen zur Verfügung, jedoch sind diese äußerst berechnungsintensiv und die Genauigkeit hängt stark von der Kamerakalibrierung und dem Model der Objektgeometrie ab [312]. Das Problem ist stark mit den Structure-from-Motion Ansätzen verwandt [313]. Die Methoden zur Lösung des Problems lassen sich in analytische und Least-Squares Verfahren gliedern [307], wobei für eine allgemeine Übersicht auf [314], [315], [316] verwiesen sei.

Bildbasierte visuelle Regelung (IBVS). Die IBVS Methode verwendet zur Regelung des Roboters rein bildbasierte Merkmale, die aus der aufgenommenen Szene gewonnen werden. Die Anforderungen an die verwendeten Bildfeatures liegen dabei vor allem in der Skalierungsinvarianz und einer geringen Berechnungskomplexität, in Bezug auf den Bildverarbeitungsalgorithmus. Als Merkmale werden oftmals Bildfeatures (SIFT, SURF oder Harris-Operator, usw.) verwendet [312], [317]. Auch geometrische Marker wie Ecken, Kanten, Bildregionen [318], [319] oder künstliche Landmarken werden verwendet. Die Abbildung eines Merkmals $\underline{x}^w \in \mathbb{R}^3$ im Weltkoordinatensystem in die Bildebene des Sensors $\underline{f} \in \mathbb{R}^2$ lässt sich hierbei mit der Transformation, Gleichung (2.10) angeben.

$$\lambda \cdot \underline{\tilde{f}} = \underline{K} \cdot {}^{s}_{w}\underline{T} \cdot \underline{\tilde{x}}^w \, , mit \, \underline{K} = \begin{pmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{2.10}$$

Hierbei wird das erweiterte Bildmerkmal $\underline{\tilde{f}} = (u_i \quad v_i \quad 1)^\intercal$ und der erweiterte Weltpunkt mit $\underline{\tilde{x}}^w = (x_j \quad y_j \quad z_j \quad 1)^\intercal$ definiert, vergleiche Abbildung 2-14. Die extrinsischen Parameter sind hier durch die Transformation ${}^{s}_{w}\underline{T} \coloneqq (\underline{R}\ \underline{t})$, mit ${}^{s}_{w}\underline{T} \in \mathbb{R}^{3\times4}$ von Welt- nach Sensor- beziehungsweise Kamerakoordinatensystem gegeben.

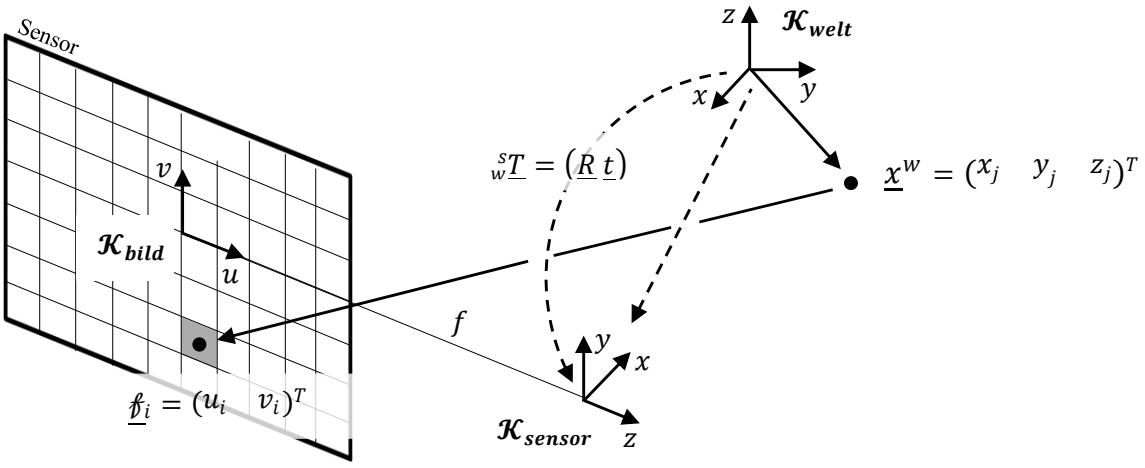


Abbildung 2-14: Abbildung eines Merkmals durch Kameramodell

Die intrinsischen Parameter $\underline{K}$ mit der Brennweite im Verhältnis zu den Pixeln f_x, f_y, dem optischen Zentrum u_0, v_0 und dem Scherungsparamter s lassen sich über Kamerakalibrierungsalgorithmen [320], [321] finden. Durch die Festlegung der optischen Achse in Richtung z-Achse des Kamerakoordinatensystems und der Wahl des Scherungsparameters mit $s = 0$, findet sich der Zusammenhang zwischen zweidimensionaler metrischer Koordinate $\underline{x}$ und Pixelkoordinate $\underline{\mathfrak{f}}$, des Merkmals gemäß Gleichung (2.11).

$$\underline{x} = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{x_j}{z_j} \\ \frac{y_j}{z_j} \end{pmatrix} = \begin{pmatrix} \frac{(u_i - u_0)}{f_x} \\ \frac{(v_i - v_0)}{f_y} \end{pmatrix}. \tag{2.11}$$

Über die perspektivische Projektion und der Änderungsgeschwindigkeit des Bildmerkmals, lässt sich nun ein Zusammenhang der Kamerageschwindigkeit $\underline{v}_c = \underline{\dot{x}}_c = (v_{c,x} \quad v_{c,y} \quad v_{c,z} \quad \omega_{c,x} \quad \omega_{c,y} \quad \omega_{c,z})^T$ über die Feature-Jakobi-Matrix (Interaction-Matrix) $\underline{J}_{\mathfrak{f}}$ herstellen [322], [312].

$$\underline{\dot{\mathfrak{f}}} = \underline{J}_{\mathfrak{f}} \cdot \underline{v}_c, mit\ \underline{J}_{\mathfrak{f}} = \begin{pmatrix} -\frac{1}{z_j} & 0 & \frac{u_i}{z_j} & u_i \cdot v_i & -(1 + {u_i}^2) & v_i \\ 0 & -\frac{1}{z_j} & \frac{v_i}{z_j} & 1 + {v_i}^2 & -u_i \cdot v_i & -u_i \end{pmatrix}. \tag{2.12}$$

In Abbildung 2-15 ist der schematische Aufbau dargestellt. Der Tiefenwert z_j des Weltpunktes $\underline{x}^w$ kann dabei über Schätzverfahren, Beobachteransätze [323] oder bei Einsatz eines 2.5D Sensors direkt gemessen werden. Zur Regelung von sechs Freiheitsgraden werden nun mindestens drei Features benötigt, wobei in der Praxis für gewöhnlich mehr Punkte, aufgrund von Messrauschen verwendet werden. Die Bestimmung der Geschwindigkeit erfolgt dann über die (Pseudo-)Inverse (Moore-Penrose) der Feature-Jakobi-Matrix.

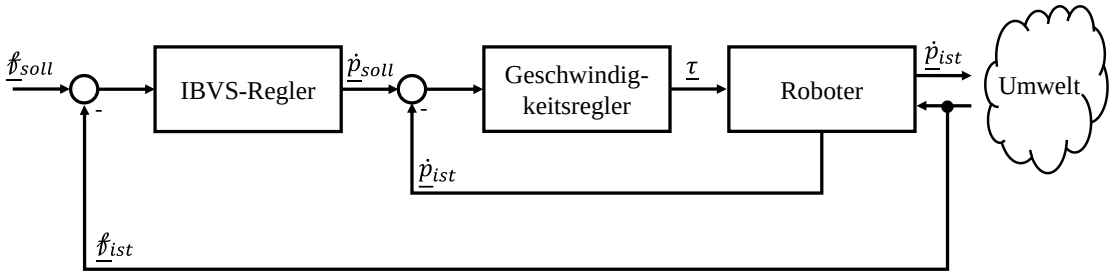


Abbildung 2-15: Bildbasierte visuelle Regelung mit unterlagerter Geschwindigkeitsregelung

Erweiterte Ansätze. In [324] wird eine Entkopplung der translatorischen und rotatorischen Bewegungskomponente vorgeschlagen, wodurch eine globale, asymptotische Stabilität erzielt werden kann. Eine weitere Strategie stellen die Arbeiten [325], [326] dar. Diese verwendet eine schaltende Regelung, welche nach ausgewählten Kriterien zwischen einem PBVS- und IBVS-Regler umschaltet, wodurch die Konvergenzgeschwindigkeit sowie die Stabilität verbessert werden kann. Dies liegt vor allem daran, dass sich PBVS besser für weiter entfernte Objekte eignet, während IBVS seine Stärken in der Feinpositionierung aufweist. Weitere Verfahren nutzen Tiefendaten von RGBD-Sensoren [327], [328]. Die Bildung der Regelgröße erfolgt direkt aus der Differenz von Soll- und Istbild. Somit können rechenaufwendige Schritte, wie die Posenschätzung und die Featureextraktion sowie das anschließende Featurematching vermieden werden. Nachteilig ist jedoch, dass das Verfahren sehr sensibel auf Sensorrauschen reagiert.

Hybride Regelung. Unter dem Begriff hybride Systeme/Regelung wird die Verknüpfung von zeitkontinuierlichen und ereignisdiskreten Systemen verstanden. Für komplexe, sensorgeführte Roboteraufgaben müssen zwangsläufig hybride Effekte beachtet werden, wenn zwischen einzelnen Teilreglern (beispielsweise bildbasierte Regelung; Kraft-Momenten-Regelung) zustandsabhängig, beziehungsweise autonom, umgeschaltet wird. Eine wichtige Unterklasse, welche zur Familie der hybriden Systeme gehört, stellen schaltende Systeme dar, siehe hierzu [329]. Eine prinzipielle Schwierigkeit besteht in der Stabilitätsanalyse hybrider Systeme. So ist bekannt, dass etwa durch ungeeignete Schaltvorgänge zwischen zwei linearen, stabilen Teilsystemen das Gesamtsystem instabil werden kann. Formal stehen für den Stabilitätsnachweis das Konzept der gemeinsamen Lyapunov Funktionen *(engl. „common Lyapunov functions")* zur Verfügung. Bei linearen Systemen kann durch „langsames" Umschalten zwischen den Teilsystemen Stabilität gewährleistet werden. Dies ist unter dem Konzept der Dwell-Time bekannt, vergleiche hierzu [329]. Eine gute Übersicht bietet auch [330]. Aus praktischer Sicht ist es dabei von besonderer Bedeutung, während dem Umschaltprozess, eine stoßfreie Steuergröße (Stetigkeit der Steuergröße) zu garantieren, sodass ungewünschte Systemfrequenzen oder Stellgrößenbeschränkungen vermieden werden. Zur Gewährleistung einer stoßfreien Umschaltung existieren verschiedene regelungstechnische Verfahren. Exemplarisch hierfür sei eine Reihe an Methoden genannt, welche unter dem Begriff Anti-Windup Bumpless Transfer fallen [331]. Die Grundidee besteht darin, dass die Anpassung der Steuergröße, durch einen rückgekoppelten Kompensator, erfolgt. Ähnliche Probleme, wie bei Umschaltvorgängen, treten im Gain-Scheduling auf. Zum Ablösen der einzelnen Teilregler wird deswegen eine Interpolation zwischen den Stellgrößenwerten der Teilregler vorgeschlagen [332]. Eine interessante Möglichkeit für den Entwurf von Umschaltvorgängen wird in der Arbeit [333], auf Basis eines flachheitsbasierten Ansatzes, vorgestellt.

In der Robotik werden zum Umschalten zwischen den Teilreglern verschiedene Konzepte vorgeschlagen. In [150] wird ein „sanftes“ Umschalten mit einem fuzzy-basierten Ansatz nach [334] eingeführt. Eine der relevantesten und innovativsten Arbeiten stellen [335], [336] dar. Es wird ein Verfahren zur Online-Trajektorien-Generierung (OTG) entwickelt, das ein Umschalten zwischen verschiedenen Reglern und Koordinatensystemen für beliebige Zeitpunkte und Systemzustände ermöglicht. Der Fokus der Arbeit liegt dabei auf der direkten Reaktion des Roboters auf unvorhersehbare Sensorevents, sodass eine Art „robotischer Reflex“ auf Umwelteingaben erlaubt wird. Der OTG erlaubt eine Synchronisation (Sollbeschleunigung und Sollgeschwindigkeit werden im Zielpunkt erreicht) der einzelnen Freiheitsgrade und eine glatte, zeitoptimale Trajektoriengenerierung innerhalb eines Steuerungszyklus. Wird beim Umschalten zwischen verschiedenen Teilreglern eine Instabilität erkannt, für welche verschiedenen Erkennungskriterien definiert sind, übernimmt der OTG die Planung einer Trajektorie, sodass ein sicherer Zustand erreicht werden kann [337]. Die Anwendung auf Aktionsprimitive wird in der Arbeit [338] diskutiert. Die Verfahren sind in der Reflexxes Motion Library [339] verfügbar, welche kommerziell erwerbbar ist.

2.4 Fazit und Handlungsbedarf

Der Stand der Forschung und Technik zeigt, dass bisher kein ganzheitliches Konzept existiert, welches ein Robotersystem befähigt, autonom Instandhaltungsaufgaben in produktionstechnischen Anlagen auszuführen. Davon betroffen sind vor allem die Bereiche Wartung und Instandsetzung. Für die automatisierte Inspektion stehen, wissenschaftlich betrachtet, bereits Möglichkeiten zur Verfügung, sodass diese Thematik nicht weiter behandelt werden soll. Auch für die Navigation der Plattform sind bereits ausreichend Lösungen erforscht, sodass die Mobilität des Robotersystems nicht Teil dieser Arbeit ist.

Das Kernelement dieser Arbeit bildet deswegen die Erforschung und Umsetzung von Manipulationsfähigkeiten zur Demontage und Montage unter Instandhaltungsbedingungen, da diese die wesentlichen Grundvoraussetzungen für fast alle Wartungs- und Instandsetzungsarbeiten darstellen. Die einzelnen Fähigkeiten müssen in ein Gesamtsystem integriert werden, wobei der Autonomiegrad mindestens dem der Montagerobotik entsprechen sollte, sodass ohne externe Hilfe die Automatisierung von Instandhaltungsaufgaben von der Planung bis zur Ausführung ermöglicht werden kann. Der implizite Handlungsbedarf für die eigene Arbeit setzt sich aus den nachfolgenden Punkten zusammen.

Planung von Wartungs- und Instandsetzungsaufgaben. Die Planung der einzelnen Manipulationen muss, sowohl auf Basis von Produkt- wie CAD-Daten, als auch mit Sensordaten, möglich sein. Nur so können Umweltunsicherheiten zwischen digitalem Modell und realer Umwelt kompensiert werden. Bisherige Systeme nutzen entweder nur Vorwissen oder Sensorinformationen oder entwickeln keine Möglichkeiten beide Informationsquellen geeignet zu fusionieren. Auch muss eine zeiteffiziente Planung der einzelnen Manipulationen möglich sein. Verfahren aus der Montagerobotik, mit dem Fokus in der Ermittlung einer nah-optimalen Sequenz, sind hierbei nicht sinnvoll einsetzbar. Ursächlich hierfür ist neben den hohen Planungszeiten auch die fehlende Verwendung von Sensorinformation für die Planung. Ein sinnvoller Einsatz und die Akzeptanz dieser Systeme ist nur möglich bei Planungszeiten von bis zu wenigen Sekunden und der

Möglichkeit vorab unbekannte Bauteile in die Planung miteinfließen zu lassen. Die Planung einer nah-optimalen Aufgabensequenz ist für die Instandhaltungsrobotik kein entscheidender Punkt, da der zeitliche Vorteil durch die bessere Sequenz oftmals erst ab einer hohen Anzahl an Wiederholungen derselben Aufgabe zum Tragen kommt. Auch wird in den Arbeiten zur Demontage- und Montageplanung keine Möglichkeit der kollisionsfreien Bahnplanung des Roboters, inklusive der Separationsschritte, aufgezeigt, welche für die erfolgreiche Ausführung der Aufgaben zwingend notwendig ist, da die kollisionsfreie Separation der einzelnen Bauteile nicht isoliert betrachtet werden kann. Ebenso sollte für die Bahnplanung ein Verfahren entwickelt werden, welches unabhängig von verwendetem (globalen) Bahnplanungsalgorithmus die Berechnungszeit reduziert. Bisher liegt der Fokus in der Forschung auf der Entwicklung neuer Bahnplanungsalgorithmen die immer eine spezifische Eigenschaft optimieren. Jedoch ist kein Verfahren bekannt, welches die Eigenschaften für eine bestimmte Klasse an Algorithmen insgesamt verbessert.

Steuerung und Regelung. Es bestehen bereits Lösungen, welche die flexible Nutzung verschiedenster Steuerungs- und Regelungsalgorithmen für die einzelnen Roboteraufgaben, in Form einer hybriden Steuerungsarchitektur, mittels Aktionsprimitiven, erlauben. Dies ist für die flexible Ausführung von Wartungs- und Instandsetzungsaufgaben auch zwingend und kann durch das Konzept der Aktionsprimitive gelöst werden. Jedoch besteht bisher kein Verfahren, welches rein symbolische Anweisungen aus der Planung in subsymbolische Elemente in Form von Aktionsprimitiven übersetzt, womit die Flexibilität über alle Systemebenen hoch gehalten werden kann. Für die Ausführung müssen geeignete exterozeptive Regelungsansätze, gemäß dem Stand der Technik, ausgewählt und in das Gesamtsystem integriert werden. Dabei sollte beim Entwurf besonders auf eine einfache Übertragbarkeit für industrielle Robotersteuerungen geachtet werden, sodass das Gesamtsystem auch für bestehende Robotersteuerungen Verwendung finden kann.

Tabelle 2-3 beschreibt nochmals zusammenfassend den Handlungsbedarf, der sich aus dem Stand der Forschung und Technik ableiten lässt.

Tabelle 2-3: Analyse des Handlungsbedarf

Domäne		**Methode/System**			
		Instandhaltungs-robotik	Montage-robotik	Demontage-robotik	**Eigene Arbeit**
Autonomiegrad		◔	◕	◑	◕
Planungsalgorithmen (Aufgabenplanung, Bahnplanung)	Geringe Berechnungskomplexität	○	◔	◑	●
	Optimalität	○	◕	○	◑
	Unsicherheits-kompensation	○	○	◑	●
Steuerung/Regelung	Kopplung zu Planung	◔	●	◑	●
	Flex. Steuerungs-programmgenerierung	○	◑	◔	◕

○ Nicht vorhanden ◔ Niedrig ◑ Mittel ◕ Hoch ● Sehr hoch

3 Aufgabenanalyse und Anforderungen

„Zu keiner einzigen Wahrheit ist man gelangt,
ohne dass man vorher vier oder vierzigmal
Unsinn gesagt hätte."
Fjodor Michailowitsch Dostojewskij

Folgendes Kapitel analysiert die Aufgaben und Anforderungen, um die in 1.1 dargelegte Forschungsfrage, auf Basis der bekannten Voraussetzungen und Defizite aus dem Stand der Forschung und Technik, in den Kapiteln 4, 5 und 6 geeignet zu beantworten. Das Ziel ist die Ausarbeitung der wesentlichen Aufgabenpunkte, sodass zukünftige Robotersysteme für die automatisierte Instandhaltung grundlegende Manipulationsaufgaben, im Kontext von Wartungs- und Instandsetzungsarbeiten, im produktionstechnischen Umfeld, autonom lösen können. Dabei soll der Fokus auf Demontage- und Montageoperationen liegen, wie diese unter den Rahmenbedingungen der Instandhaltung benötigt werden. Das Kapitel untergliedert sich in eine allgemeine Aufgabenanalyse, die die Hauptschwerpunkte der zu lösenden Aufgaben auf makroskopischer Ebene darstellt. Im zweiten Teil, der spezifischen Aufgabenanalyse, werden die relevanten Schwerpunkte im Detail für die gefundenen Hauptschwerpunkte analysiert.

3.1 Allgemeine Aufgabenanalyse

Damit ein Robotersystem eine Wartungs- oder Instandsetzungsaufgabe autonom lösen kann, müssen zentrale Punkte zur Autonomiebildung entwickelt werden. Die zentralen Schwerpunkte zur Erfüllung der Aufgabe lassen sich unter folgende Punkte zusammenfassen:

- **Aufgabenplanung:** Die Aufgabenplanung soll es dem Roboter ermöglichen, die einzelnen Teilaufgaben zur Lösung einer Aufgabe automatisch zu generieren. Die Aufgabenplanung soll auf Basis von vorhandener Information, als auch visueller Information arbeiten, sodass Umweltunsicherheiten kompensiert werden können.
- **Visuelle Perzeption:** Mit aus bildbasierten Sensoren gewonnener Information, soll dem Robotersystem Auskunft über aktuelle Umweltzustände gegeben werden. Die Information muss dabei so aufbereitet werden, dass sie in einer für die Planung geeigneten Repräsentation vorliegt.
- **Umweltmodell und Aufgabenexploration:** Für die hier gegebene Aufgabenstellung ist ein Umweltmodell für die zentrale Informationshaltung dringend erforderlich, denn nur so kann a priori und mittels visueller Perzeption gewonnene Information geeignet fusioniert und der Planung zur Verfügung gestellt werden. Auch muss es dem Robotersystem ermöglicht werden, in Abhängigkeit der aktuellen Umweltzustände, automatisch geeignete Sensorposen zu planen, sodass ausreichend Information über die zu lösende Aufgabe gesammelt werden kann.
- **Bahnplanung:** Neben der Bereitstellung eines Aufgabenplans muss für die Ausführung der Aufgabe die nötige Bahninformation vorhanden sein. Dies erfordert die Integration eines Bahnplaners.

- **Steuerung und Regelung:** Das Zusammenführen der Informationen aus der Aufgaben- und Bahnplanung für die Generierung eines Steuerungsprogramms (Anwenderprogramms) stellt einen grundlegenden Aspekt dieser Arbeit dar. Dies erlaubt die Verknüpfung der Planungsebene mit der Ausführung. Neben der Generierung des Anwenderprogramms müssen für die einzelnen Teilaufgaben verschiedene Regelungsfunktionen bereitgestellt werden, sodass eine reaktive Ausführung erlaubt wird.

Damit eine durchgängige Kette zur Automatisierung von Instandsetzungs- und Wartungsaufgaben bereitgestellt werden kann, muss eine Steuerungsarchitektur entwickelt werden, die die obigen Punkte geeignet kombiniert und abstrahiert. Dabei ist es von besonderer Bedeutung, dass die Architektur nicht nur eine sequentielle Abarbeitung erlaubt. Diese muss, wenn es die Gegebenheiten erfordern, eine Neuplanung der Aufgaben und eine Generierung des Anwenderprogramms zu jedem Zeitschritt ermöglichen. Die zentrale Architektur soll unabhängig von der verwendeten Roboterplattform sein, sodass diese auf verschiedene Systeme übertragbar ist. Aufgrund der stark algorithmischen Natur der einzelnen Aufgabenschwerpunkte, soll die Entwicklung nach Regeln der modellbasierten Softwareentwicklung erfolgen, da hierdurch Designentscheidungen stark erleichtert werden.

3.2 Spezifische Aufgabenanalyse

Die spezifische Analyse eruiert die nötigen Anforderungen, für die in der allgemeinen Analyse herausgestellten wesentlichen Schwerpunkte der Aufgabenplanung, der visuellen Perzeption, des Umweltmodell und der Aufgabenexploration, der Bahnplanung sowie der Steuerung und Regelung.

Aufgabenplanung. Die Planung von Wartungs- und Instandsetzungsaufgaben ist deutlich mit der Problemstellung in der automatisierten (De-)Montage verwandt. Jedoch lässt sich, wie bereits in 2.4 gezeigt, keine der bekannten Planungsverfahren sinnvoll auf die Domäne der automatisierten Instandhaltung übertragen.

Die Aufgabenplanung soll als Ausgangsinformation sowohl CAD-Baugruppenmodelle als auch entsprechend aufbereitete Information aus visueller Perzeption nutzen. Das Vorhandensein von CAD-Modellen ist für gewöhnlich gegeben, sodass dies keine wesentliche Einschränkung darstellt. Das in [108] vorgeschlagene Baugruppenmodell stellt eine geeignete Grundlage für die Aufgabenplanung dar. Durch die Nutzung des darin vorgeschlagenen relationalen Graphs soll ein Planungsalgorithmus entwickelt werden, welcher auf symbolischer Ebene arbeitet. Um eine Planung auf symbolischer Ebene zu erlauben, muss das Baugruppenmodell jedoch um semantische Information erweitert werden. Für die Generierung des relationalen Graphs muss ebenso ein vollständiger und korrekter Algorithmus zur Bestimmung des Demontageraums entwickelt werden. Die Aufgabenplanung muss zwingend folgende Punkte erfüllen:

- Möglichkeit zur Planung auf Basis von verschiedenen Informationsquellen.
- Allgemeingültiger Dekompositionsalgorithmus zur Bestimmung möglicher Demontage-richtungen.
- Planung auf Basis von symbolischen Operationen, welche die Manipulationsfähigkeit des Robotersystems abstrahieren. Dies erlaubt eine hohe Flexibilität in der Generierung eines Anwenderprogramms.
- Niedrige Berechnungskomplexität der gesamten Aufgabenplanung.

Visuelle Perzeption. Für die Objekterkennung existieren bereits ausreichend leistungsfähige Verfahren, die Verwendung für die gegebene Problemstellung finden können. Im Rahmen dieser Arbeit sollen deswegen keine neuen Algorithmen für die Objekterkennungsaufgabe entwickelt werden. Für den gegebenen Problemfall eignen sich vor allem erscheinungsbasierte Verfahren, da diese auch bei Abweichungen zwischen gelernter Klasse und realem Objekt eine hohe Performance aufweisen. In der Servicerobotik sind diese Verfahren jedoch zumeist für Haushaltsumgebungen eingesetzt und evaluiert worden. Die verwendeten Objektklassen weisen für gewöhnlich eine deutlich höhere farbliche Textur, im Vergleich zu industriellen Objekten, auf. Auch müssen innerhalb eines Objektes in der Regel keine weiteren Objekte erkannt werden, wie dies bei Baugruppen der Fall ist. Zur Lösung muss deswegen ein iteratives Erkennungsverfahren entwickelt werden, von der Baugruppe bis zu den einzelnen Bauteilen.

Für die metrische Kartierung der Umgebung stehen ebenso geeignete Methoden zur Verfügung. Speziell die Nutzung einer Octree-Datenstruktur für eine probabilistischen Voxelkarte, wie dies in [159] vorgeschlagen wird, zeichnet sich durch ihr speichereffizientes und skalierbares Repräsentationsformat aus. Im Rahmen dieser Arbeit muss für eine geeignete Perzeptionslösung deswegen Folgendes untersucht und entwickelt werden:

- Auswahl, Kombination und experimentelle Untersuchung von geeigneten erscheinungsbasierten Objekterkennungsverfahren für die semantische Annotation und Lokalisation.
- Entwicklung eines Konzepts zur Bestimmung geometrischer Primitive, dem Objekt zugehörig sowie von einfachen Kontaktbeziehungen zwischen den einzelnen Objekten.
- Zusammenführung der beiden obigen Verfahren für eine gemeinsame Szenenanalyse. Die Ausgabe der Szenenanalyse muss dabei in Form eines symbolisch, räumlichen Graphen erfolgen, sodass dieser für die Manipulationsplanung verwendbar ist.

Die Integration der einzelnen Konzepte in ein visuelles Perzeptionsframework schließt diesen Aufgabenpunkt ab.

Umweltmodell und Aufgabenexploration. Das Umweltmodell, als zentrale Informationsquelle für die Planung, soll eine geeignete Informationsstruktur bereitstellen. Dies umfasst die Verwaltung von semantischer, topologischer und metrischer Information. Für die Fusionierung der Information aus CAD und visueller Perzeption, soll ein probabilistischer Ansatz gewählt werden. So können einzelne Beobachtungen auf Basis ihres „Wahrheitsgehalt“ fusioniert werden.

Damit zielgerichtet visuelle Information für das Umweltmodell bereitgestellt werden kann, muss die Aufgabenexploration folgende Punkte erfüllen:

- Automatische Exploration der Aufgabe. Dies bedeutet, dass der Raum der Aufgabe ausreichend kartiert, als auch die einzelnen Teilobjekte erkannt werden. Die einzelnen visuellen Teilaufgaben sollen in einem gemeinsamen Optimierungsansatz kombinierbar sein, wodurch eine schnelle Aufgabenexploration sichergestellt wird.
- Dynamische Berücksichtigung der Umweltzustände, sodass die visuelle Teilaufgabe in der Aufgabenexploration bevorzugt wird, welche zum aktuellen Zeitpunkt geringer exploriert wurde.
- Zeiteffiziente Planung der einzelnen Sensorposen für die Aufgabenexploration.

Bahnplanung. Die automatische Generierung einer Bewegungsbahn muss durch den Roboter selbst ermöglicht werden. Geeignete Bahnplanungsalgorithmen sind, gemäß dem Stand der Forschung und Technik, vorhanden. Für die Anforderungen der automatisierten Instandhaltung eignen sich vor allem globale Bahnplaner, da von einer statischen Umgebung während der Ausführung ausgegangen werden kann. Ein aktuelles Defizit besteht jedoch in der direkten Abhängigkeit der Bahnplanungszeit mit dem verwendeten Planungsmodell und dessen strukturellen Umgebungseigenschaften. Die Größe des Planungsraums skaliert immer in Abhängigkeit der verwendeten Schrittweite des Planers und dessen lokalen Umgebungseigenschaften. Um dieses Defizit zu beseitigen, soll eine Vorverarbeitungsstrategie entwickelt werden, sodass die Dimension des Planungsraums deutlich reduziert wird. Die Vorverarbeitungsstrategie soll weitergehend folgende Eigenschaften aufweisen:

- Reduktion des Planungsraums, unabhängig von der verwendeten Bahnplanungsmethode (such- oder stichprobenbasiert).
- Keine negative Beeinflussung der Bahngüte, hinsichtlich Optimalität und Glattheit.
- Verbesserung der Erfolgsrate innerhalb eines vorgegebenen zeitlichen Planungshorizonts.

Des Weiteren soll die Bahnplanung folgenden Anforderungen genügen:

- Nutzung der von der visuellen Perzeption bereitgestellten, probabilistischen Voxelkarte.
- Möglichkeit zur modularen Integration weiterer Planungs- und Kollisionserkennungs-algorithmen sowie kinematischer Robotermodelle.
- Bereitstellung einer Schnittstelle, die eine Anbindung an diverse Robotersteuerungen über eine Positions-/ Geschwindigkeitsschnittstelle erlaubt.

Steuerung und Regelung. Die in der Aufgabenplanung generierten symbolischen Aktionen müssen, in eine für die Steuerung lesbare Sprache, übersetzt werden. Als Schnittstelle zwischen Planung und Ausführung eignet sich, wie bereits in 2.3.3 gezeigt, insbesondere das Konzept der Aktionsprimitive. Dieses soll aufgrund seiner Vorteile auch in dieser Arbeit verwendet werden. Jedoch soll eine automatische Generierung der Aktionsprimitiven, anhand des symbolischen Aufgabenplans, erlaubt werden. Hierzu müssen folgende Gegebenheiten erfüllt werden:

- Definition einer rekonfigurierbaren Menge an Aktionsprimitiven.
- Definition und Umsetzung einer Methode, welche auf Basis des symbolischen Aufgabenplans ein Anwenderprogramm mittels Aktionsprimitiven erzeugt. Das Verfahren soll auch die Möglichkeit bieten einen Alternativplan zur Laufzeit auszuführen.

Für die einzelnen nötigen Teilregler (Lageregelung, bildbasierte Regelung, Kraft-Momenten-Regelung), die zur Ausführung der verschiedenen Aufgaben nötig sind, existieren bereits ausgereifte und gut erforschte Regelungskonzepte. Diese können mittels der Aktionsprimitive in einer hybriden (hier: schaltenden) Regelungsarchitektur allokiert werden. Folgendes soll für eine erfolgreiche Ausführung in letztem Arbeitspunkt erfüllt werden:

- Einheitlicher Stellgrößeneingriff der Teilregler, wie dieser bei industriellen Steuerungen zur Verfügung steht (Position/ Geschwindigkeit).

Alle entwickelten Einzelverfahren sollen auf ihre Leistungsfähigkeit, durch anwendungsnahe Beispiele, experimentell validiert werden. Durch Validierung der Einzelverfahren, sowie die Integration in ein gemeinsames, roboterplattformunabhängiges Steuerungsframework, soll die

Gesamtfunktionalität des Systems ganzheitlich untersucht werden. Abbildung 3-1 verdeutlicht den Zusammenhang der zu lösenden Aufgabenpunkte zur Erfüllung der Anforderungen anhand des Informationsflusses. Dieser steht in Anlehnung an den Aufbau der Arbeit, vergleiche hierzu Abbildung 1-2.

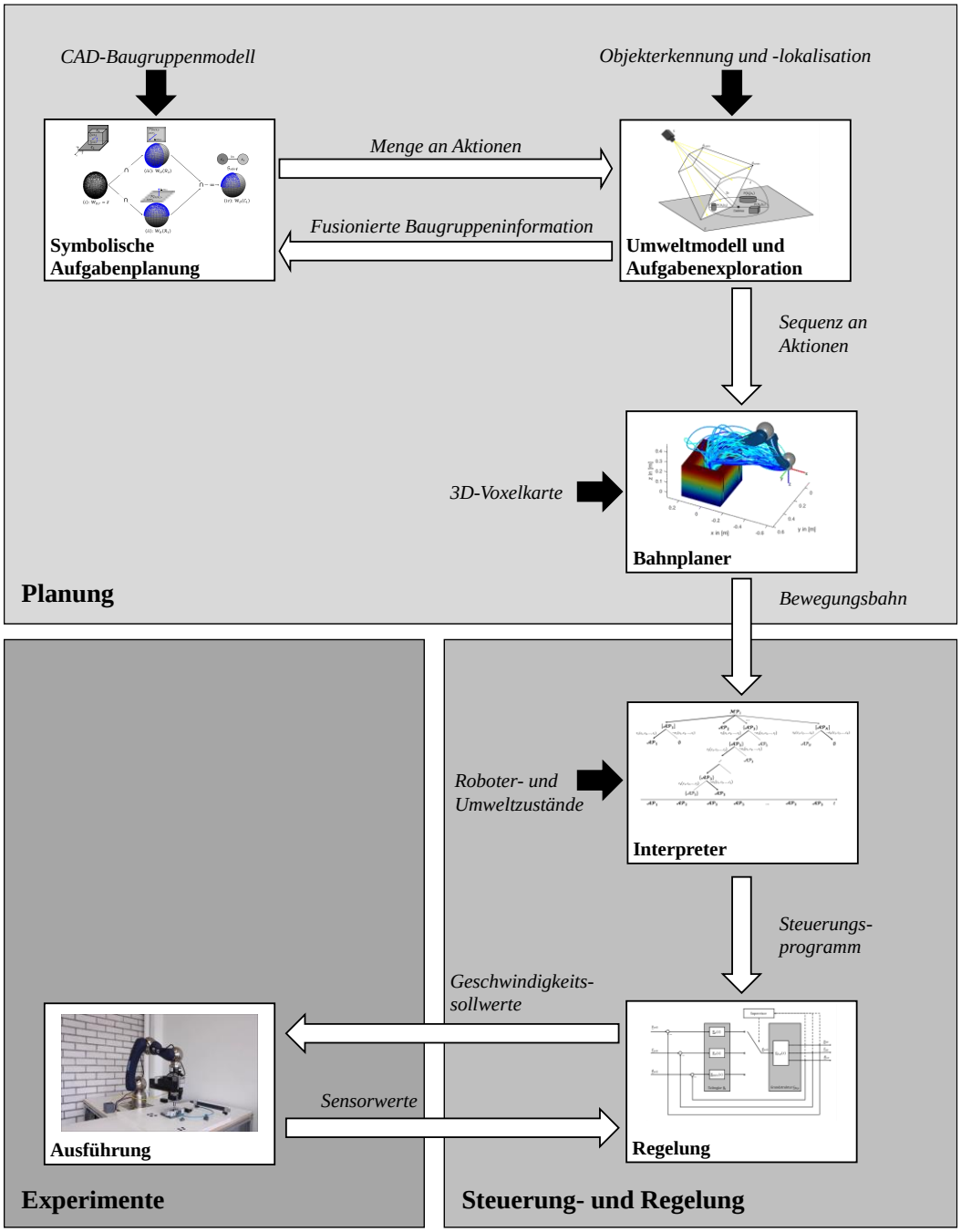


Abbildung 3-1: Informationsfluss des Gesamtkonzepts auf Basis der Aufgabenanalyse

4 Entwicklung von Methoden zur Planung von Robotermanipulationen

„Die Rechenautomaten haben etwas von den Zauberern im Märchen. Sie geben einem wohl, was man sich wünscht, doch sagen sie einem nicht, was man sich wünschen soll."
Norbert Wiener

Dieses Kapitel beschreibt neuartige Methoden, welche es einem Robotersystem erlaubt, einen Aufgabenplan für die Automatisierung von Instandhaltungsaufgaben zu erzeugen. Dabei liegt der Fokus auf der automatischen Generierung von Robotermanipulationen, mit dem Ziel, Wartungs- und Instandsetzungsaufgaben durch ein Robotersystem autonom ausführbar zu gestalten.

Zur Bereitstellung durchgängiger Umweltinformation wird ein Informationsmodell vorgeschlagen, dass topologische, semantische und metrische Informationen bereitstellt und eine Verknüpfung von a priori (CAD-Daten) und a posteriori (Visuelle Perzeption) Information gestattet. Dies erlaubt eine geeignete Abstraktionsmöglichkeit, welche die Grundlage für die Manipulationsplanung bildet. Für die Bestimmung der Demontageräume wird ein stichprobenbasiertes Verfahren vorgeschlagen, welches, gegenüber bekannten Verfahren aus dem Stand der Technik, äußerst effizient die vollständige Bestimmung relativer, translatorischer Demontageräume erlaubt. Auf Basis der Demontageräume wird der relationale Graph abgeleitet, welcher zur Planung der Handlungsanweisungen dient. Die Planung erfolgt mittels symbolischer Beschreibung der Aktionsfähigkeit, wodurch eine effiziente und flexible Berechnung des Manipulationsplans erfolgen kann.

Des Weiteren wird eine visuelle Perzeptionsstrategie vorgeschlagen, die eine Bestimmung einfacher Kontaktgraphen mit symbolisch, räumlichen Relationen ermöglicht. Somit kann visuelle Information über die Aufgabe mit in die Planung eingebracht werden. Die Fusion von a priori und Sensorinformation erfolgt durch einen probabilistischen Ansatz. Dies erlaubt die Verwaltung des Wahrheitsgehalts über die Umweltzustände auf einer einheitlichen und formalen Grundlage. Zur Planung geeigneter Sensorposen, für die Maximierung der erfassten visuellen Information, wird ein dynamischer Ansatz eingeführt, der die Verknüpfung visueller Aufgaben zulässt, womit eine geeignete, parallele Abarbeitung visueller Perzeptionsaufgaben erfolgen kann. Zusätzlich werden geeignete Sequenzierungsmechanismen des Manipulationsplans für die Ausführung untersucht.

Neben der Bestimmung der symbolischen Aktionen, ist die Bereitstellung von Bahninformation, zur Ausführung einer Aufgabe, zwingend. Zur Planung einer geeigneten Bahn, für die einzelnen Aktionen, werden aufgrund ihrer Leistungsfähigkeit dem Stand der Technik bekannte globale Bahnplanungsverfahren verwendet. Jedoch wird zur besseren Skalierung in Bezug auf Planungszeit und Bahnoptimalität ein Verfahren entwickelt, welches eine ausreichende Entkopplung mittels adaptiver Schrittweitensteuerung erlaubt.

Die entwickelten Verfahren werden in eigenen Frameworks (Manipulationsplanung, visuelle Perzeption und Bahnplanung) zur Verfügung gestellt, welche isoliert oder im Verbund zum

Gesamtsystem genutzt werden können. Abschließend wird die Struktur des gesamten Planungssystems diskutiert und Ausblicke auf zukünftige Forschungsarbeiten gegeben, die das System erweitern könnten.

4.1 Modellierung von Umweltinformation

Zur Manipulationsplanung werden sowohl a priori Informationen, aus CAD-Baugruppenmodellen, (Passive Information), als auch a posteriori Informationen aus visuellen Sensordaten (hier RGBD; Aktive Information), verwendet. Um passive und aktive Information geeignet zu fusionieren, zu verwalten und für die Planungsaufgabe bereitzustellen, wird in diesem Unterkapitel eine geeignete Modellierung der Umweltinformation vorgeschlagen. Hierzu wird das allgemeine Konzept der Objektinformationselemente eingeführt, welches sich als Schnittstelle von passiver und aktiver Information eignet und somit ein gemeinsames Informationsmodell abbildet.

Definition 1: Objektinformationselement. *Ein Objektinformationselement $\mathcal{I}_{\mathcal{O},i} = \langle \mathcal{I}_{\mathcal{S},i}, \mathcal{I}_{\mathcal{T},i}, \mathcal{I}_{\mathcal{M},i} \rangle$ beschreibt alle zu einem Objekt $\mathcal{O}_i$ zugehörige Information $\mathcal{I}_{\mathcal{O},i}$, welche sich in die folgenden drei Hauptgruppen gliedern lässt:*

- ***Semantisches Informationselement:*** *Das semantische Informationselement $\mathcal{I}_{\mathcal{S},i}$ beschreibt die semantische Information zu einem Objekt $\mathcal{O}_i$, wobei sich dieses aufteilt in $\mathcal{I}_{\mathcal{S},i} = \langle \mathcal{OS}_i, \{\mathcal{OA}_i\}, \{\mathcal{OG}_i\} \rangle$, mit:*
  - *$\mathcal{OS}_i$ entspricht der Objektsemantik, also dem eindeutigen semantischen Objektbezeichner.*
  - *$\{\mathcal{OA}_i\}$ entspricht der Menge an Objektattributen, welche sowohl objekt- als auch informationsspezifisch sein können.*
  - *$\{\mathcal{OG}_i\}$ entspricht der Menge an geometrischen Entitäten eines Objekts, welche als geometrische Primitive beschrieben werden.*
- ***Topologisches Informationselement:*** *Das topologische Informationselement $\mathcal{I}_{\mathcal{T},i}$ beschreibt die Verknüpfung eines Objekt $\mathcal{O}_i$ in Form eines Graphs $\mathcal{G}(\mathcal{O}, \mathcal{OA})$, wobei als Verknüpfungsrelationen verschiedene Objektattribute $\mathcal{OA}_i$ gelten können.*
- ***Metrisches Informationselement:*** *Das metrische Informationselement $\mathcal{I}_{\mathcal{M},i} = \langle {}^{m}_{n}\underline{T}_{\mathcal{O}_i}, {}^{m}_{n}\underline{T}_{\mathcal{OA}_i}, P_{\mathcal{OG}_i} \rangle$ beschreibt alle metrischen Beziehungen im euklidischen Raum. Hierzu zählen folgende Elemente:*
  - *${}^{m}_{n}\underline{T}_{\mathcal{O}_i}$ entspricht der homogenen Abbildung des Objekts $\mathcal{O}_i$ von Frame n nach Frame m, wobei ${}^{m}_{n}\underline{T}_{\mathcal{O}_i} \in SE(3)$.*
  - *${}^{m}_{n}\underline{T}_{\mathcal{OA}_i}$ entspricht der homogenen Abbildung des Objektattributs $\mathcal{OA}_i$ von Frame n nach Frame m, wobei ${}^{m}_{n}\underline{T}_{\mathcal{OA}_i} \in SE(3)$.*
  - *$P_{\mathcal{OG}_i}$ entspricht der parametrischen Beschreibung der geometrischen Primitive mit $P_{\mathcal{OG}_i} = \langle P_i, {}^{m}_{n}\underline{T}_{\mathcal{OG}_i} \rangle$ mit den Parametern P_i und der homogenen Abbildung ${}^{m}_{n}\underline{T}_{\mathcal{OG}_i} \in SE(3)$.*

Die gesamte Umweltinformation $\mathcal{I}_{\mathcal{U}}$ über N-Objekte lässt sich folglich durch die Vereinigung $\bigcup_i^N \mathcal{I}_{\mathcal{O},i} = \mathcal{I}_{\mathcal{U}}$ bestimmen, wobei die Vereinigungsvorschriften im topologischen Informationselement als Verknüpfungsrelationen eindeutig hinterlegt sind. Zudem besteht für ein Objektinformationselement sowie dessen Eigenschaften ein probabilistisches Modell $P\big(E \in \mathcal{I}_{\mathcal{O},i} | z\big)$, welches für eine Beobachtung z die Wahrscheinlichkeit P für die Eigenschaft $E \in \mathcal{I}_{\mathcal{O},i}$ beschreibt. Diese Beschreibung eignet sich, wie in 4.4.1 gezeigt wird, besonders für die Fusionierung von

passiver und aktiver Information. Weitergehend wird ein Modell für die Informationsmodellierung von Produktionssystem und Baugruppe vorgeschlagen, welches nach der Struktur von Definition 1 gegliedert ist.

Modellierung von Produktionssystem und Baugruppe. Für die Modellierung von Produktionssystem und Baugruppe, eignet sich insbesondere die relationale Beschreibung [108], [254], welche aufgrund ihrer positiven Eigenschaften auch in dieser Arbeit verwendet wird und auf die speziell gegebenen Rahmenbedingungen, wie der Einführung semantischer Eigenschaften und der Gliederung nach Definition 1, erweitert wird. Dabei werden folgende Definitionen für die Modellierung eingeführt. Die Spezifikation der Baugruppe orientiert sich dabei an [108].

Definition 2: Modell Produktionssystem. *Ein Produktionssystem $\mathcal{PS} = \langle \mathcal{AG}, \mathcal{R} \rangle$ sei eine Menge an Baugruppen $\mathcal{AG} \in \{AG_1, AG_2, \dots, AG_n\}$, welche über Verbindungsrelationen $\mathcal{R} \in \{r_1, r_2, \dots, r_m\}$ miteinander verknüpft sind. Die Verbindungsrelationen sind nach Definition 3 gegeben.*

Definition 3: Modell Baugruppe. *Das Baugruppenmodell $\mathcal{AG} = \langle \mathcal{C}, \mathcal{R} \rangle$ setzt sich zusammen aus einer Menge an Bauteilen $\mathcal{C} \in \{C_1, C_2, \dots, C_i\}$ und den Verbindungsrelationen $\mathcal{R} \in \{r_1, r_2, \dots, r_j\}$ zwischen den einzelnen Bauteilen. Dabei lassen sich die einzelnen Eigenschaften wieder semantischer, topologischer und metrischer Information, nach Definition 1 zuordnen.*

- ***Bauteil:*** *Ein Bauteil $C_i = \langle \mathcal{OS}_{C_i}, \mathcal{OA}_{C_i} \rangle$ setzt sich aus folgenden Informationselementen zusammen:*
  - *$\mathcal{OS}_{C_i} = \langle \mathcal{OS}_i, {}^{\mathcal{AG}_i}_{C_i}\underline{T} \rangle$ beschreibt eindeutig die Objektsemantik $\mathcal{OS}_i$ einer Komponente C_i, beispielsweise $\mathcal{OS}_{C_i} \in \{Schraube, Stift, Gehäuse, \dots\}$, sowie deren homogenen Abbildung ${}^{\mathcal{AG}_i}_{C_i}\underline{T}$ in zugehöriges Baugruppenkoordinatensystem $\mathcal{AG}_i$.*
  - *$\mathcal{OA}_{C_i}$ beschreibt die Objektattribute $\mathcal{OA}_{C_i} = \langle \mathcal{CF}, \mathcal{FG}_{ssr} \rangle$ mit:*
    - *Komponentenfeatures $\mathcal{CF} = \langle \{hole, thread, slot, \dots\}, \underline{o}^{\mathcal{CF}}, p_{\mathcal{CF}} \rangle$ umfasst alle der Komponente zugehörigen Features im Ursprung $\underline{o}^{\mathcal{CF}} \in \mathbb{R}^3$, beschrieben im Komponentenkoordinatensystem $\mathcal{CF}$, mit zugehöriger Featureparametrierung $p_{\mathcal{CF}}$.*
    - *Featuregeometrie $\mathcal{FG}_{ssr} = \langle \{plane, line, point, \dots\}, \underline{f}_{ssr} \rangle$ beschreibt die Verknüpfungsgeometrie an welcher eine symbolisch, räumliche Relation ssr mit Featurevektor $\underline{f}_{ssr} = (\underline{o}_{ssr} \ \ \underline{n}_{ssr})$ an Ursprung $\underline{o}_{ssr} \in \mathbb{R}^3$ mit Normalenvektor $\underline{n}_{ssr} \in \mathbb{R}^3$ definiert ist.*
- ***Verbindungsrelation:*** *Eine Verbindungsrelation $r_j = \langle C_i, C_j, ssr, r_N \rangle$ beschreibt den Zusammenhang zwischen einer Komponente C_i und C_j, welche durch die symbolisch, räumliche Relation* [236], [108], [106] *$ssr \in \{concentric, congruent, screwed, \dots\}$ miteinander verbunden sind (klassische Verknüpfungsrelationen im CAD). r_N beschreibt die Relationennummer (ID), welche bei mechanischen Verknüpfungen die Zugehörigkeit zu einer Verbindung beschreibt. Die geometrische Information ist in dem, der Verknüpfung zugehörigen, Featuregeometrieelement aus $\mathcal{FG}_{ssr}$ hinterlegt.*

Die einzelnen Daten zur Bildung des Informationsmodells lassen sich prinzipiell aus allen gängigen CAD-Systemen über eine API (Advanced Programming Interface) extrahieren. Ebenso lassen sich viele Eigenschaften aus visuellen Sensordaten bestimmen, wodurch redundante, fusionierbare Informationen aus passiver und aktiver Information vorliegen.

4.2 Symbolische Manipulationsplanung für Roboteraufgaben

Weitergehend wird eine Methode entwickelt, die, auf Basis des in vorherigem Unterkapitel eingeführten Baugruppenmodells, automatisch symbolische Roboteraktionen, sogenannte Manipulationsprimitive, erzeugt. Auf Basis der Manipulationsprimitive wird der Aufgabenplan zur Lösung einer Roboteraufgabe symbolisch beschrieben. Dabei ist das Verfahren in zwei Stufen unterteilt. Im ersten Schritt wird das relationale Baugruppenmodell aus den CAD-Daten generiert und mit einem stichprobenbasierten Verfahren, der relative Demontageraum für die einzelnen Bauteile, bestimmt. Auf Grundlage des Lösungsraums kann folglich der relationale Graph erzeugt werden, welcher im zweiten Schritt, der Aufgabenplanung, weiterverarbeitet werden kann. Diese arbeitet hauptsächlich auf Basis von Symbolen mittels einer definierten Regelgrammatik, wodurch eine zeiteffiziente Aufgabenplanung gewährleistet werden kann. Aspekte dieses Unterkapitel sind bereits in den Vorarbeiten [17], [18], [21] veröffentlicht worden.

4.2.1 Stichprobenbasiertes Verfahren zur Bestimmung von Demontageräumen

Die in diesem Unterkapitel entwickelte Methode basiert auf Basis der Kontaktinformationen, welche aus den Verbindungsrelationen im relationalen Baugruppenmodell abgeleitet werden können. Dieser Ansatz wird auch in anderen Arbeiten verfolgt, etwa [108], [92]. Jedoch sind die Verfahren unvollständig, führen zu Fehlern in der Lösung oder es wird kein Formalismus angegeben, welcher die Berechnung des Lösungsraums überhaupt erst erlaubt. Der hier entwickelte Ansatz ist allgemeingültig und formal so gestaltet, dass er auf Basis von geometrischer und semantischer Information arbeitet. Beginnend sollen zwei Definitionen eingeführt werden, welche im weiteren Verlauf noch benötigt werden.

Definition 4: Relativer Demontageraum $\mathbb{W}_{\boldsymbol{D,r}}^{(\boldsymbol{i})}$. *Der relative Demontageraum $\mathbb{W}_{D,r}^{(i)} \in SE(3) = \mathbb{R}^3 \times SO(3)$ beschreibt alle möglichen Bewegungsfreiheitsgrade $\underline{d}_{r,(i)} \in \mathbb{R}^6$, die ein Bauteil C_i in seiner aktuellen Pose, auf Basis der Verbindungsrelationen $\mathcal{R}$ definierten Kontakte, relativ zu allen anderen Bauteilen $C_j, \forall j \in \mathcal{AG}$ ausführen kann.*

Definition 5: Absoluter Demontageraum $\mathbb{W}_{\boldsymbol{D,a}}^{(\boldsymbol{i})}$. *Der absolute Demontageraum $\mathbb{W}_{D,a}^{(i)} \in SE(3) = \mathbb{R}^3 \times SO(3)$ beschreibt alle möglichen Bewegungsfreiheitsgrade $\underline{d}_{a,(i)} \in \mathbb{R}^6$, die ein Bauteil C_i in seiner aktuellen Pose, absolut zu allen anderen Bauteilen $C_j, \forall j \in \mathcal{AG}$ kollisionsfrei ausführen kann, bis es von der Baugruppe $\mathcal{AG}$ separiert ist, dementsprechend $C_i \cap \mathcal{AG} = \emptyset$.*

Somit gilt $\mathbb{W}_{D,r}^{(i)} \subseteq \mathbb{W}_{D,a}^{(i)}$. Praktisch bedeutet dies, dass ein Bauteil im relativen Demontageraum durchaus relativ zu anderen Bauteilen bewegt, aber nicht separiert werden kann. Der absolute Demontageraum beschreibt dagegen alle Bewegungsrichtungen, um ein Bauteil von seiner Baugruppe zu separieren. Abbildung 4-1 verdeutlicht diesen Zusammenhang anhand eines einfachen Beispiel. Das Bauteil C_1 besitzt einen relativen Demontageraum (a), jedoch kann es im aktuellen Zustand nicht separiert werden, da der absolute Demontageraum für C_1 nicht existiert (b). Prinzipiell ist die Bereitstellung von $\mathbb{W}_{D,a}^{(i)}$ für die Manipulationsplanung als geeigneter zu bewerten. Allerdings ist dies berechnungstechnisch äußerst ungünstig, da für jedes Bauteil eine Bewegungsanalyse mit Kollisionsprüfung ausgeführt werden muss und ein kombinatorisches Problem darstellt.

(a) (b)

Abbildung 4-1: Illustration von relativem und absolutem Demontageraum: Der relative Demontageraum von C_1 beinhaltet alle translatorischen Bewegungsrichtungen außer $z \in \mathbb{R}_-$ sowie keine Rotationen um die x- und y-Achse (a); Ein absoluter Demontageraum ist für C_1 jedoch nicht vorhanden, da zuvor C_3 von der Baugruppe $\mathcal{AG}$ separiert werden müsste (b)

Die Bestimmung von $\mathbb{W}_{D,r}^{(i)}$ dagegen ist deutlich einfacher, da hier die Kontaktinformationen genutzt werden können, um den relativen Demontageraum zu bestimmen. Dies ist jedoch mit dem Nachteil verbunden, dass sich der relative Demontageraum nur bedingt zur Separation einzelner Bauteile eignet. Allerdings wird sich in 4.4.3 zeigen, dass dieses Problem einfach umgangen werden kann, indem nur lokale Lösungsräume für die Sequenzierung der Demontagefolge berücksichtigt werden. Deswegen wird weitergehend eine Lösung für $\mathbb{W}_{D,r}^{(i)}$ entwickelt. Diese beruht auf der Diskretisierung des Raums, da wie nachfolgend gezeigt werden kann, hierdurch eine besonders günstige Berechnungskomplexität erzielt wird.

Sei der initiale, translatorische, relative Demontageraum $\mathbb{W}_D$ (zur besseren Lesbarkeit wird auf den Index r verzichtet) für ein kontaktloses Bauteil C_i gegeben zu einer dreidimensionalen Einheitssphäre $\mathcal{S}^3$ (alle Normalen auf $\mathcal{S}^3$ bilden die möglichen Demontagerichtungen), also

$$\mathbb{W}_D: \mathcal{S}^3 = \left\{ \underline{d} \in \mathbb{R}^3 \mid \left\| \underline{d} \right\|_2 = \| (x\ y\ z)^T \|_2 = 1 \right\}. \tag{4.1}$$

Für eine effiziente Berechenbarkeit soll $\mathcal{S}^3$ durch eine diskrete Punktmenge $\mathcal{S}_d^3$ approximiert werden, mit dem Ursprung der Normalen $\underline{d}$. Zur Gleichverteilung der Punkte auf $\mathcal{S}^3$ eignet sich vor allem das Diskretisierungsverfahren nach Marsaglia [340]. Wird $\omega_1, \omega_2 \in \mathcal{U}\{-1, \dots, 1\}$ aus der Gleichverteilung $\mathcal{U}$ gewählt, dann gilt $\forall \omega_1, \omega_2 \in \mathcal{S}^3$ falls diese die Eigenschaft (4.2) erfüllen.

$$\omega_1^2 + \omega_2^2 < 1. \tag{4.2}$$

Für die Zahlen ω_1, ω_2 kann der zugehörige kartesische Punkt $\underline{d} \in \mathcal{S}_d^3$, mit (4.3) bestimmt werden.

$$\underline{d}(\omega_1, \omega_2) = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 2\omega_1 \sqrt{1 - \omega_1^2 - \omega_2^2} \\ 2\omega_2 \sqrt{1 - \omega_1^2 + \omega_2^2} \\ 1 - 2(\omega_1^2 + \omega_2^2) \end{pmatrix} \tag{4.3}$$

Will man nun für C_i den zugehörigen Demontageraum $\mathbb{W}_D^{(i)}$ im freien translatorischen Raum $\mathbb{W}_D$ beschreiben, so müssen alle möglichen Bewegungsrichtungen aus den Kontaktinformationen

bestimmt werden. Dies lässt sich durch die Bildung der diskreten Teilmengen $\mathbb{W}_D^{(i)}(\mathcal{R}_k)$ für $k = 1,2,\dots,N$ erreichen, wobei N die Anzahl an Verknüpfungen $\mathcal{R} = \mathcal{ssr}_r\left(\mathcal{FG}_{ssr,m}, \mathcal{FG}_{ssr,n}\right)$ mit den symbolisch, räumliche Relation $\mathcal{ssr}_r$ sowie den zugehörigen Featuregeometrien $\mathcal{FG}_{ssr,m} \circ \mathcal{FG}_{ssr,n}$ zwischen Bauteil C_i und $C_r, \forall r \in \mathcal{AG}$ beschreibt. In Abbildung 4-2 (b)-(d) ist dies schematisch am Beispiel typischer Kontaktkonfigurationen aufgeführt. So können die Demontageräume für beispielsweise die Verknüpfung $congruent(plane, plane)$ durch eine Hemisphäre (b), $concentric(circle, circle)$ durch die Normalenvektoren der Kontakte (c), oder $screwed(cylinder, cylinder)$ durch eine leere Menge (d) beschrieben werden. Die Zuordnung der einzelnen, diskreten Teilmengen, ist aufgrund der Verknüpfungsbeschreibung $\mathcal{R}$, immer eindeutig definiert.

Nach Bildung der einzelnen Teilmengen, müssen diese noch in Richtung des zugehörigen Normalenvektors $\underline{n}_{ssr} = [e_x \quad e_y \quad e_z]^T$ transformiert werden. Die Beschreibung des Normalenvektors in Kugelkoordinaten in einer Einheitskugel, also $R = 1$, lässt sich durch Gleichung (4.4) angeben. Vergleiche auch die Darstellung in Abbildung 4-2 (a).

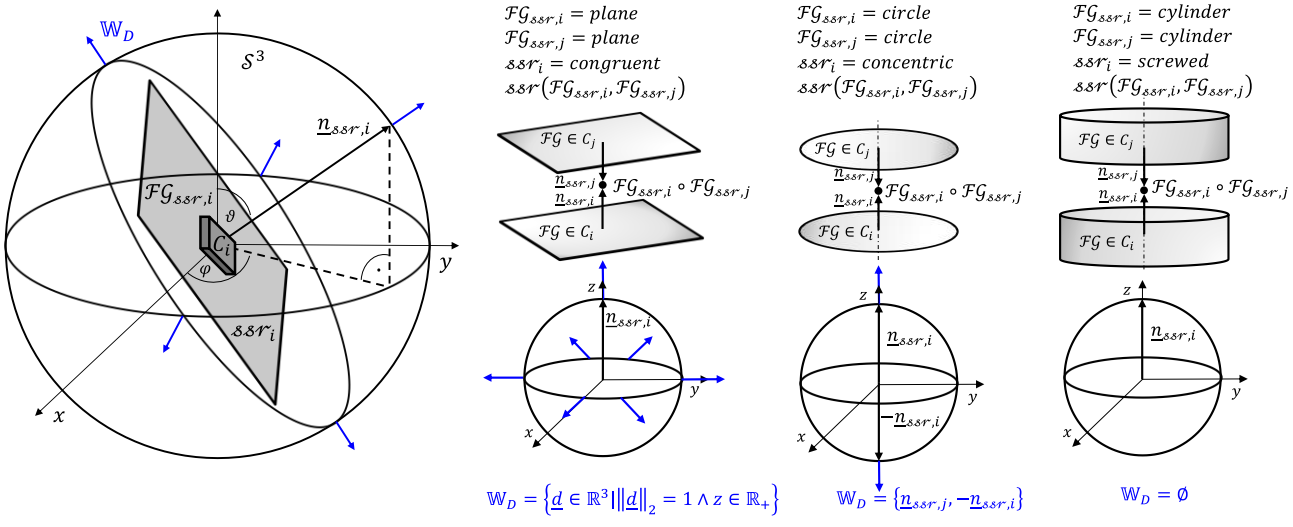
$$\begin{pmatrix} e_x \\ e_y \\ e_z \end{pmatrix} = \begin{pmatrix} sin(\vartheta) \cdot cos(\varphi) \\ sin(\vartheta) \cdot sin(\varphi) \\ cos(\vartheta) \end{pmatrix}, \tag{4.4}$$

Somit erfolgt die Bestimmung des Polarwinkels mit

$$\vartheta = arccos(e_z), \tag{4.5}$$

sowie des Azimutwinkels zu

$$\varphi = arccos\left(\frac{e_x}{sin(arccos(e_z))}\right). \tag{4.6}$$



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

Abbildung 4-2: Grundprinzip zur Erzeugung von Demontageräumen: Festlegung der Polarkoordinaten in Abhängigkeit des Normalenvektor (a); Schematische Darstellung der Lösungsräume auf Basis von Featuregeometrie und symbolisch, räumlicher Relation: deckungsgleiche Ebene (b); konzentrische Kreise (c); verschraubte Zylinder (d)

Folglich werden nun alle Punkte $\underline{d}^{(i)} \in \mathbb{W}_D^{(i)}$ in zugehörige Orientierung, gegeben durch $\underline{n}_{SSR}$, transformiert, wodurch sich Zusammenhang (4.7) ergibt.

$$\underline{d}^{(i)\#} = \overbrace{\begin{pmatrix} cos(\vartheta) & 0 & sin(\vartheta) \\ 0 & 1 & 0 \\ -sin(\vartheta) & 0 & cos(\vartheta) \end{pmatrix}}^{\underline{R}_y(\vartheta)} \cdot \overbrace{\begin{pmatrix} cos(\varphi) & -sin(\varphi) & 0 \\ sin(\varphi) & cos(\varphi) & 0 \\ 0 & 0 & 1 \end{pmatrix}}^{\underline{R}_z(\varphi)} \cdot \underline{d}^{(i)}. \tag{4.7}$$

Der resultierende Demontageraum für ein Bauteil C_i lässt sich dementsprechend für N-Kontaktrelationen für die korrespondierenden und transformierten Teilmengen $\mathbb{W}_D^{(i)}(\mathcal{R}_k)$ mit

$$\mathbb{W}_D^{(i)} := \left(\bigcap_{k=1}^{N} \mathbb{W}_D^{(i)}(\mathcal{R}_k) \right) \cap \mathbb{W}_D \tag{4.8}$$

angeben. Die Auswertung von Gleichung (4.8) gestaltet sich nun, aufgrund des diskreten Charakters, als denkbar einfach und effizient, da die Bestimmung der Schnittmenge auf Grundlage von Sortier- und Vergleichsoperationen gelöst werden kann. Hierzu werden die den Teilmengen zugehörigen Punkte $\{\underline{d}^{(i)\#}\}$ in Gebietsintervalle $[\underline{d}_s^{(i)\#} - \underline{\varepsilon} \quad \underline{d}_s^{(i)\#} + \underline{\varepsilon}]$ gerundet, sodass mehrere Punkte einem Gebietsintervall zugehörig sind. So kann beim Zusammenfallen von Gebietsintervallen zwischen $[\underline{d}_s^{(i)\#} - \underline{\varepsilon} \quad \underline{d}_s^{(i)\#} + \underline{\varepsilon}]$ und $[\underline{d}_t^{(i)\#} - \underline{\varepsilon} \quad \underline{d}_t^{(i)\#} + \underline{\varepsilon}]$ von einer Zusammengehörigkeit der Schnittmenge ausgegangen werden. In Abbildung 4-3 ist das Verfahren zur Bestimmung des relativen Demontageraums, auf Basis des hier vorgestellten stichprobenbasierten Verfahrens, dargestellt. Es werden die einzelnen Teilmengen bestimmt und mit dem initialen Demontageraum geschnitten, wodurch sich der finale Demontageraum ermitteln lässt. Der zugehörige Berechnungsvorgang zu diesem Beispiel ist in Abbildung 4-4 aufgeführt.

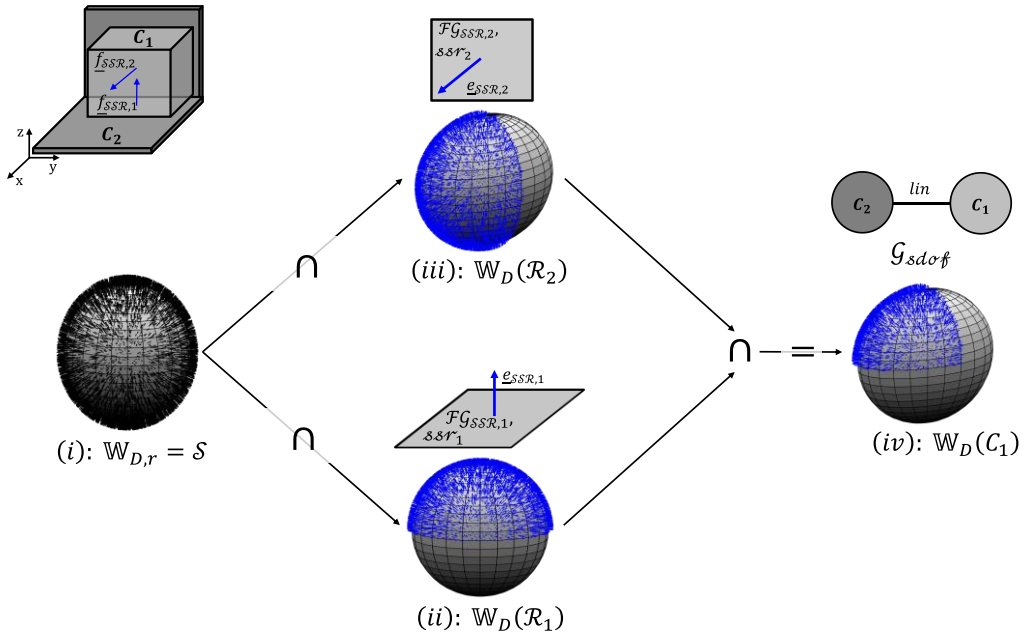


Abbildung 4-3: Stichprobenbasierte Bestimmung des relativen Demontageraums: Diskretisierung des freien translatorischen Raums (i); Diskretisierung der einzelnen Featuregeometrien (ii)-(iii); Berechneter Demontageraum für Bauteil C_1 (iv) und zugehöriger relationaler Graph

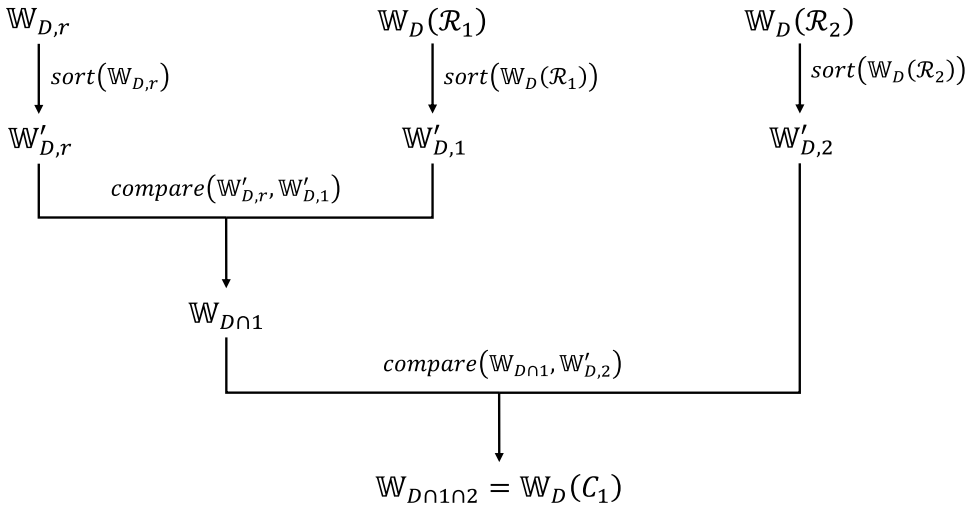


Abbildung 4-4: Berechnungsvorschrift über Sortier- und Vergleichsoperationen zu Abbildung 4-3

Die gerundeten Gebietsintervalle entsprechen pro Koordinate einer unsortierten $(n \times 2)$-Liste. Mittels Sortierung können diese in ihre zugehörige Diagonalform überführt werden. Die Schnittmenge kann dann über die Vergleichsoperation gebildet werden, wobei durch die Einteilung in Gebietsintervalle die Zuordnung der Schnittmengenbildung ermöglicht wird.

Die Berechnung des relativen translatorischen Demontageraums lässt sich somit auf eine einfache Sortierung, mit anschließender Vergleichsoperation, zurückführen. Dies ist vor allem für die Berechnungskomplexität des Verfahrens von Vorteil, da die Sortierung einer Liste der Länge n in $\mathcal{O}(n \cdot log(n))$ und einer Liste der Länge m in $\mathcal{O}(m \cdot log(m))$ liegt. Der anschließende Vergleich kann dann in $\mathcal{O}(n + m)$ gelöst werden, wodurch sich das Gesamtproblem für die Bestimmung des Demontageraums einer Komponente in $\mathcal{O}(n \cdot log(n))$ mit $n \geq m$ befindet. Eine ausreichende obere Grenze an Stichproben, zur Findung der translatorischen Freiheitsgrade, liegt bei $n \cong 260000$.

Die Bestimmung aller rotatorischen Demontagerichtungen ist auf diese Weise jedoch nicht durchführbar. Eine Teillösung kann auf Basis der Kontaktinformation durch die Bestimmung von

$$N = rank(\underline{n}_{ssr,1} \quad \underline{n}_{ssr,2} \quad \cdots \quad \underline{n}_{ssr,M}), N \subseteq M \tag{4.9}$$

erzielt werden. Gilt $N = 0$, existieren drei rotatorische Freiheitsgrade, da es sich um die Triviallösung, für ein kontaktfreies Bauteil handelt. Für $N = 1$ und $\underline{o}_{ssr,i} = \underline{o}_{ssr,j}$ für $\forall i, j = M - N$ existiert jeweils ein rotatorischer Freiheitsgrad. Eine genaue Darstellung ist auch in der Vorarbeit [17] dargestellt.

Algorithmus 4.1 beschreibt den gesamten Ablauf zur Bildung des relativen Demontageraums. Als Eingabe wird der Baugruppengraph oder Kontaktgraph mit den symbolisch, räumlichen Relationen $\mathcal{G}_{ssr}(C, ssr)$ erwartet. Nach Bestimmung des initialen, kontaktfreien Demontageraums $\mathbb{W}_D$ werden die einzelnen Teilmengen für die Kontaktrelationen berechnet und in jedem Schritt über die $intersect(\cdot)$-Funktion die resultierende Schnittmenge über Sortier- und Vergleichsoperation berechnet. Neben der Bestimmung des relativen Demontageraums für die einzelnen Bauteile, wird der relationale Baugruppengraph $\mathcal{G}_{sdof}(C, sdof)$, beschrieben als ungerichteter Graph mit der Zulassung von Schleifen, bestimmt. Dabei beschreibt $sdof$ den Freiheitsgrad in symbolischer Weise durch folgende Notation nach [108] (gekoppelte Freiheitsgrade werden nicht berücksichtigt):

- $\boldsymbol{fix}$: 0 translatorische und 0 rotatorische Freiheitsgrade.
- $\boldsymbol{lin}$: 1 translatorischen und 0 rotatorische Freiheitsgrade.
- $\boldsymbol{rot}$: 0 translatorische und 1 rotatorischen Freiheitsgrade.
- $\boldsymbol{fits}$: 1 translatorischen und 1 rotatorischen Freiheitsgrade.
- $\boldsymbol{agpp}$: 2 translatorische und 1 rotatorischen Freiheitsgrade.
- $\boldsymbol{free}$: 3 translatorische und 3 rotatorische Freiheitsgrade.

Die Zuordnung der einzelnen Symbole erfolgt auf Basis des bestimmten, relativen Demontageraums. Der relationale Graph wird weitergehend für die Aufgabenplanung benötigt.

Algorithmus 4-1: Stichprobenbasierte Freiheitsgradanalyse

Eingabe: Graph mit symbolisch räumlichen Relationen $\mathcal{G}_{ssr}(C, ssr)$; Relationales Baugruppenmodell $\mathcal{RM}$.

Ausgabe: Demontageräume $\langle \mathbb{W}_{D,C} \rangle$; Relationaler Graph $\mathcal{G}_{sdof}(C, sdof)$.

1: *// Berechne diskrete Sphäre*
2: $\mathcal{S} \leftarrow \text{marsagliaSphere}(N)$
3: $\mathbb{W}_{D,i} \leftarrow \mathcal{S}$;
4: **for** $i = 1{:}1{:}|C|$
5: **for** $j = 1{:}1{:}|\mathcal{FG}_{ssr}|$
6: $\mathbb{W}_{D,j} \leftarrow \text{getSubSpace}\left(\mathcal{FG}_{ssr,j}, \underline{f}_{ssr}\right)$;
7: **end for**
8: $\mathbb{W}_{D,i} \leftarrow \text{intersect}(\mathbb{W}_{D,j}, \mathbb{W}_{D,i})$;
9: $\langle \mathbb{W}_{D,C} \rangle \leftarrow \mathbb{W}_{D,i}$;
10: **end for**
11: $\mathcal{G}_{sdof} \leftarrow \text{getRelationalGraph}(\langle \mathbb{W}_{D,C} \rangle, \mathcal{G}_{ssr})$;
12: **return** $(\langle \mathbb{W}_{D,C} \rangle, \mathcal{G}_{sdof})$;

1: **function** $(\mathcal{S}) \leftarrow \text{marsagliaSphere}(N)$
2: $i = 1$;
3: **while** $(i \leq N)$
4: *// Bestimme zwei Zufallszahlen aus Normalverteilung*
5: $\omega_1 \leftarrow \mathcal{U}(-1, 1)$;
6: $\omega_2 \leftarrow \mathcal{U}(-1, 1)$;
7: **if** $(\omega_1^2 + \omega_2^2 < 1)$ **then**
8: $x(i) \leftarrow 2\omega_1\sqrt{1 - \omega_1^2 - \omega_2^2}$;
9: $y(i) \leftarrow 2\omega_2\sqrt{1 - \omega_1^2 + \omega_2^2}$;
10: $z(i) \leftarrow 1 - 2(\omega_1^2 + \omega_2^2)$;
11: $i = i + 1$;
12: **end if**
13: **end while**
14: $\mathcal{S} \leftarrow [x(:)\, y(:)\, z(:)]$;
15: **end function**

Da es sich in der Praxis meistens um translatorische Demontageoperationen handelt, vergleiche hierzu auch die Arbeit [108], soll eine vollständige Lösung für den rotatorischen Demontageraum nicht weiter untersucht werden. Durch die hier dargestellte Methode existiert ein effizientes und vollständiges Verfahren, zur Bestimmung des relativen Demontageraums, für alle translatorischen Demontagerichtungen. Dies ermöglicht die folgende symbolische Planung von Aktionen.

4.2.2 Aufgabenplanung auf Basis von Manipulationsprimitiven

Weitergehend wird ein Konzept vorgestellt, welches die automatische Planung von symbolischen Aktionen, auf Basis des relationalen Graphen, erlaubt. Durch die Überführung des Planungsproblems in eine rein symbolische Darstellung, kann eine höhere Abstraktionsebene erreicht werden, welche sich positiv sowohl für die Lösung des Planungsproblems als auch auf die Flexibilität der Ausführung, auswirkt. Dabei wird das Problem der Aufgabenplanung in Form einer mengenbasierten Repräsentationssprache beschrieben. Angelehnt an die Modelle deterministischer, endlicher Automaten, sei das Zustandsmodell für die Planungsdomäne definiert mit (4.10) mit der Menge an Zuständen $\mathcal{Z}$, der Menge an Ereignissen Σ, der Zustandsübergangsfunktion δ sowie einem Startzustand z_0 und der Menge an Zielzuständen $\mathcal{Z}_\Omega$.

$$\mathcal{P} \coloneqq \langle \mathcal{Z}, \Sigma, \delta, z_0, \mathcal{Z}_\Omega \rangle. \tag{4.10}$$

Dabei sei $\mathcal{Z}$ die Menge an Zuständen, welche durch (4.11) gegeben ist.

$$\mathcal{Z} \coloneqq \mathit{sdof}_j(C_i), \mathit{sdof} \in \{fix, lin, rot, fits, agpp, free\} \text{ und } \forall i \in \mathcal{G}_{\mathit{sdof}}, \tag{4.11}$$

Die Menge an Ereignissen $\{\sigma_1, \sigma_2, \dots, \sigma_N\} \in \Sigma$, beziehungsweise der möglichen Aktionen, wird durch den Aktionsraum (4.12), in Form von Manipulationsprimitiven $\{\mathcal{MP}\}$, definiert.

$$\Sigma \coloneqq \{\mathcal{MP}\} \in \{move, twist, pull, put\} \tag{4.12}$$

Dabei sei ein Manipulationsprimitiv ein symbolischer Operator, welcher die Aktionsfähigkeit eines Robotersystems in Bezug auf Manipulationsaufgaben beschreibt. Die Menge aller Manipulationsprimitive $\{\mathcal{MP}\}$ umfasst alle Aktionsfähigkeiten, die dem Roboter zur Verfügung stehen um eine Aufgabe zu lösen. Als elementare Primitive werden die Symbole nach Definition 6 eingeführt, wobei der wesentliche Unterschied in der späteren Weiterverarbeitung besteht, vergleiche 5.1. In Abbildung 4-5 sind die einzelnen Primitive exemplarisch dargestellt. Die einzelnen Symbole repräsentieren dabei Variablen, welche für die Ausführung noch mit expliziten Größen versehen werden müssen. Ein Manipulationsprimitiv $\mathcal{MP}$ besitzt neben der Zuordnung des Operators als Eingabeparameter zusätzlich die Festlegung eines Werkzeuges $\tau_l \in \{gripper, screwdriver, \dots\}$ und der Komponente C_j, welche durch $\mathcal{MP}_i$ direkt manipuliert werden soll.

Definition 6: Manipulationsprimitive.

move$(\cdot)$***:*** *Entspricht einer freien Bewegung von einer (symbolischen) Pose $\mathcal{p}_i$ zur (symbolische) Pose $\mathcal{p}_{i+1}$ in einem Referenzsystem $\mathcal{F}_R$ mit der Richtung $\mathcal{d}_i$.*

twist$(\cdot)$***:*** *Entspricht einer rotatorischen Bewegung von einer (symbolischen) Pose $\mathcal{p}_i$ zur (symbolische) Pose $\mathcal{p}_{i+1}$ in einem Referenzsystem $\mathcal{F}_R$, mit einer (symbolischen) Kraft/Moment $\mathcal{f}_i$ mit der Richtung $\mathcal{d}_i$.*

pull$(\cdot)$***:*** *Entspricht einer translatorischen Bewegung von einer (symbolischen) Pose $\mathcal{p}_i$ zur (symbolische) Pose $\mathcal{p}_{i+1}$ in einem Referenzsystem $\mathcal{F}_R$, mit einer (symbolischen) Kraft/Moment $\mathcal{f}_i$ mit der Richtung $\mathcal{d}_i$.*

put$(\cdot)$***:*** *Entspricht einer translatorischen Bewegung von einer (symbolischen) Pose $\mathcal{p}_i$ zur (symbolische) Pose $\mathcal{p}_{i+1}$ in einem Referenzsystem $\mathcal{F}_R$, mit einer (symbolischen) Kraft/Moment $\mathcal{f}_i$ mit der Richtung $\mathcal{d}_i$, wobei diese orthogonal zu einer Objektablage steht.*

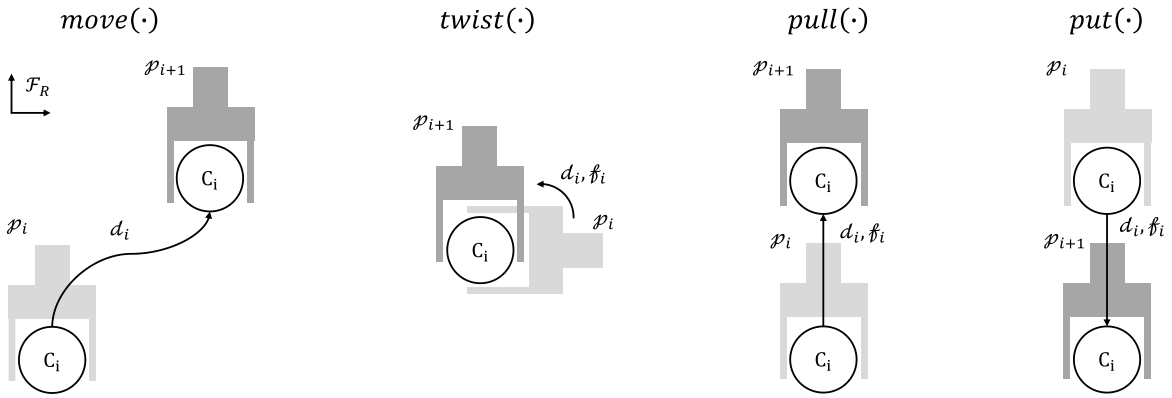


Abbildung 4-5: Übersicht der definierten Manipulationsprimitive als symbolische Operatoren

Der Operator für ein Manipulationsprimitv kann dann mit

$$\sigma_i \coloneqq \mathcal{MP}_i(C_j, \tau_l) \tag{4.13}$$

ausgedrückt werden. Folglich ordnet die Zustandsübergangsfunktion $\delta: \mathcal{Z} \times \Sigma \to \mathcal{Z}$ jedem Zustand $\mathcal{Z} \coloneqq \mathit{sdof}_k(C_j)$ auf Basis einer Aktion $\mathcal{MP}_i \in \Sigma$ eindeutig einen Nachfolgezustand $\mathcal{Z}' \coloneqq \mathit{sdof}_{k+1}(C_j), \forall j \in \mathcal{G}_{\mathit{sdof}}$ zu, also

$$\mathit{sdof}_{k+1}(C_j) \coloneqq \delta(\mathit{sdof}_k(C_j), \mathcal{MP}_i). \tag{4.14}$$

Für den Startzustand $z_\alpha = \mathit{sdof}_j(C_D)$, besteht im Falle des Austausches einer Komponente, hier C_D, das Ziel, diesen in einen definierten Endzustand $z_\Omega \in \mathcal{Z}_\Omega$, aus der Menge an Endzuständen $\mathcal{Z}_\Omega$ mit $z_\Omega = free(C_D)$ zu überführen. Praktisch gesprochen bedeutet dies, dass eine Komponente C_j in einem Freiheitsgradzustand sdof_k unter Anwendung eines Manipulationsprimitiv $\mathcal{MP}_i$ mit dem Werkzeug τ_l (sowie zugehörigem Werkzeugkommando) in einen neuen Freiheitsgradzustand sdof_{k+1} überführt wird. Die Auswahl des benötigten Manipulationsprimitiv erfolgt auf Basis der Regel (4.15).

$$\forall C, \forall \mathit{sdof}\ isObject(C) \wedge isRelation(\mathit{sdof}) \Rightarrow \mathcal{MP}_i. \tag{4.15}$$

Die Regel zum Ableiten des benötigten Werkzeuges kann allgemein durch

$$\forall C, \forall \mathcal{MP}\ isObject(C) \wedge isPrimitive(\mathcal{MP}) \Rightarrow \tau_l, \tag{4.16}$$

angegeben werden. Für die allgemeinen einstufigen Prädikate $isObject(\cdot)$, $isRelation(\cdot)$ und $isPrimitive(\cdot)$ sind explizite Prädikate in einer Regelbasis hinterlegt. So wird aus $isScrew(C)$ und $isRot(\mathit{sdof})$ das Manipulationsprimitiv $\mathcal{MP} = twist$ abgeleitet. Für $isScrew(C)$ und $isTwist(\mathcal{MP})$ wird geschlussfolgert, dass $\tau = screwdriver$ sein muss.

Der Planungsalgorithmus bestimmt initial den Teilgraph $\mathcal{G}^{(1)}_{\mathit{sdof}}$, wobei (4.17) gilt.

$$\mathcal{G}^{(1)}_{\mathit{sdof}} + C_D = \mathcal{G}_{\mathit{sdof}} \tag{4.17}$$

$\mathcal{G}^{(1)}_{\mathit{sdof}}$ enthält dabei die Menge aller Kanten $\{\mathit{sdof}\}^{(1)}$ von C_D. Die Manipulierbarkeit von C_D ist gegeben, wenn $\nexists\{\mathit{sdof}\}^{(1)} \in fix$. Ist dies nicht erfüllt, erfolgt die Bildung eines erweiterten

Teilgraphen $\mathcal{G}_{\mathcal{sdof}}^{(2)}$, der zusätzlich alle Komponenten $\{C\}^{(2)}$ enthält, die über $\{\mathcal{sdof}\}^{(2)}$ mit C_D verbunden sind. Für jede Komponente $C_k \in \{C\}^{(n)}$ werden die Verbindungskosten CC_k gemäß

$$CC_k = \sum_{i=0}^{deg_{\mathcal{G}_{\mathcal{sdof}}^{(n)}}(C_k)} CC_k(\mathcal{sdof}_i) \tag{4.18}$$

berechnet, wobei diese mit $CC(free) < CC(agpp) < CC(fits) < CC(rot) = CC(lin)$ definiert sind. Die einzelnen Komponenten werden dann gemäß ihrer Verbindungskosten manipuliert, wobei mit C_j begonnen wird, für $C_j = \min\left(CC_k\left(\{C\}^{(n)}\right)\right)$. Der Algorithmus terminiert, wenn die Bedingung $\mathcal{G}_{\mathcal{sdof}} - C_D = \emptyset$ erreicht ist. Existiert hier ebenso keine Lösung, muss der Teilgraph wiederum um eine weitere Knotenmenge erweitert werden.

Die Anwendung von (4.15), (4.16) zur Lösung der Zustandsübergangsfunktion (4.14) kann neben der direkten Beeinflussung von $\mathcal{sdof}_k(C_j) \rightarrow \mathcal{sdof}_{k+1}(C_j)$ auch eine Änderung im gesamten Graph, also $\{\mathcal{sdof}\}_k(\{C\}) \rightarrow \{\mathcal{sdof}\}_{k+1}(\{C\})$, bewirken. Die Auswirkung einer Manipulation auf weitere Komponenten kann, durch die im Baugruppenmodell festgelegte Relationennummer r_N, vergleiche 4.1, Definition 3, gelöst werden. Die Bestimmung der neuen Zustände erfolgt mittels einer Wissensbasis. So sind für elementare, mechanische Verbindungen (beispielsweise Schraub- oder Stiftverbindungen) die Zustandsübergänge eindeutig definiert. Zur eindeutigen Zuordnung des Nachfolgezustands werden zusätzlich die Objektattribute (beispielsweise $hole, thread, ...$) verwendet. Für Verbindungskonfigurationen, die nicht elementar beschreibbar sind, erfolgt eine neue Berechnung der Zustände durch das stichprobenbasierte Verfahren aus 4.2.1. In Algorithmus 4.2 ist das Planungsverfahren zusammengefasst.

Algorithmus 4-2: Symbolischer Demontagealgorithmus

Eingabe: Relationaler Graph $\mathcal{G}_{\mathcal{sdof}}(C, \mathcal{sdof})$; Demontierendes Bauteil C_D.

Ausgabe: Menge an Manipulationsprimitiven $\{\mathcal{MP}\}$.

goto C_D
// Lade alle Verbindungen j= 1,2,…,M in Reihenfolge der Verbindungskosten
$\{\mathcal{sdof}_j, C_j\} \leftarrow \text{sortCost}(\mathcal{G}_{\mathcal{sdof}} \cap C_D)$
while $\mathcal{G}_{\mathcal{sdof}} \cap C_D \neq \emptyset$ **do**
  if $(\mathcal{sdof}_j^{(D)} \neq fix)$ **then**
    // Symbolische Demontagelogik
    $\text{isObject}(C_i) \wedge \text{isRelation}(\mathcal{sdof}_j) \Rightarrow \mathcal{MP}_k$.
    $\text{isObject}(C_i) \wedge \text{isPrimitive}(\mathcal{MP}_k) \Rightarrow \tau_l$
    $\mathcal{sdof}_{j+1}(C_i) \coloneqq \delta(\mathcal{sdof}_j(C_i), \mathcal{MP}_k)$.
    // Update relationaler Graph
    ${}_{i+1}^{(i)}\mathcal{G}_{\mathcal{sdof}} \leftarrow {}_{i}^{(i-1)}\mathcal{G}_{\mathcal{sdof}}$
  else
    return (/)
  end if
  $j = j + 1$;
end while
return $(\{\mathcal{MP}\})$

Die Ausgabe des Manipulationsplaners ist die Menge aller Aktionen $\{\mathcal{MP}\}$, welche zur Lösung des Planungsproblems benötigt werden.

Durch die Planung, auf Basis der Topologie des relationalen Graphs, kann eine schnelle Bestimmung nötiger Aktionen erreicht werden. So kann das Planungsproblem in $\mathcal{O}(n)$ gelöst werden, da für eine Komponente C_i maximal m_i-Zustandsübergänge existieren können, also $n = m_i \cdot |C|$. Ein Nachteil des Verfahrens ist jedoch, dass eventuell mehr Manipulationen, als minimal nötig, generiert werden, da der Planer immer mit der Komponente beginnt, welche die geringsten Verbindungskosten aufweist. Diese Greedy-Eigenschaft führt dazu, dass die Optimalität der geplanten Menge an Manipulationsprimitiven nicht zwingend gegeben ist. Die Bestimmung alternativer Pläne, mit der Randbedingung einer minimalen Anzahl an Manipulationen, würde jedoch die lineare Komplexität des Planers aufheben, da alle möglichen Kombinationen bewertet werden müssten. Auch können durch den Planer nur lineare, monotone Pläne erzeugt werden, wodurch keine Komponenten zu Unterbaugruppen kombinierbar und gemeinsam manipulierbar sind. Dies ist vor allem dann nötig, wenn eine Komponente keinen freien Demontageraum besitzt, jedoch im Verbund zu einer Unterbaugruppe eine Demontage möglich ist.

4.2.3 Manipulationsplanungsframework

Die in 4.2.1 und 4.2.2 eingeführten Methoden und Algorithmen sind in ein Framework zur Manipulationsplanung integriert, vergleiche Abbildung 4-6. Dabei ist das Framework derart gestaltet, dass es universell an verschiedene CAD-Systeme gekoppelt werden kann.

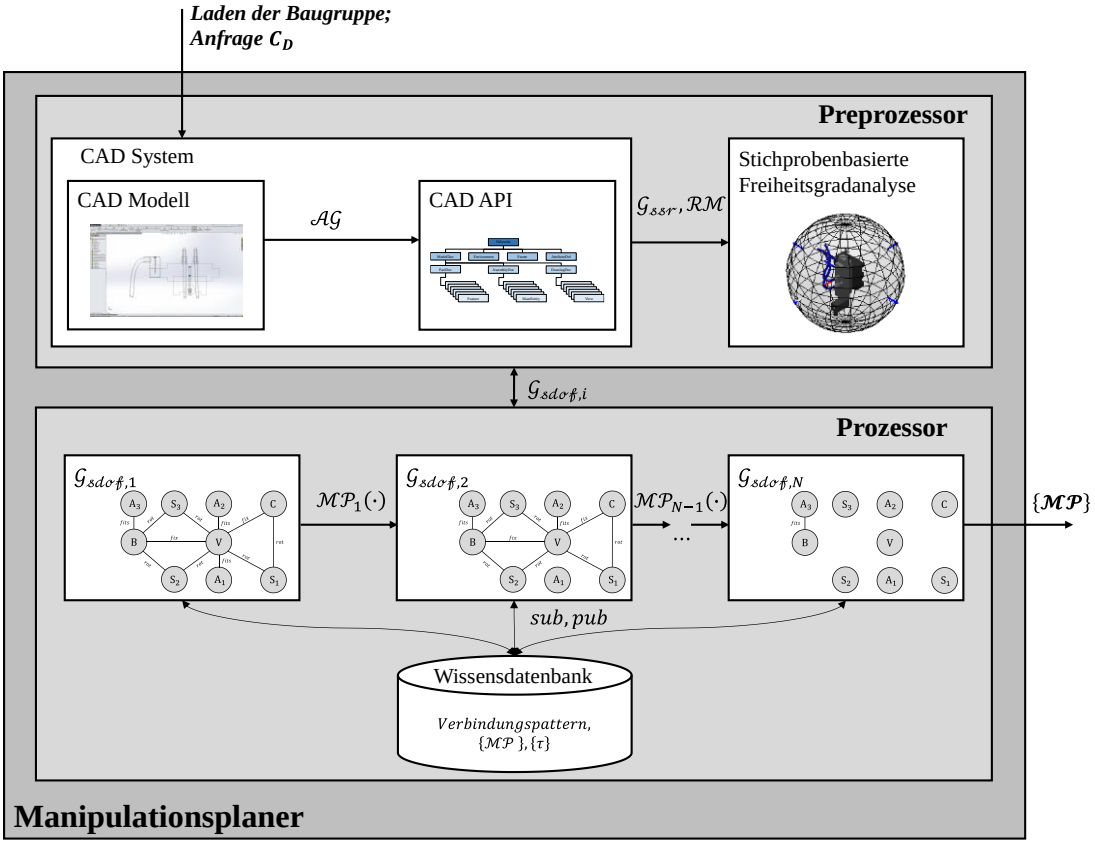


Abbildung 4-6: Architektur des Manipulationsplanungsframeworks

Die Integration, beziehungsweise Kopplung des CAD-Systems, an den Preprozessor des Manipulationsplaners, erfolgt über die systemspezifische API, welche von allen gängigen CAD-Systemen angeboten wird. Somit erfordert die Anbindung verschiedener CAD-Systeme nur eine proprietäre Schnittstelle sowie einen Wrapper, der die CAD-Informationen in das relationale Baugruppenmodell überführt. Das Laden der entsprechenden Baugruppe, sowie die Auswahl der zu demontierenden Komponente starten den Planungsvorgang. Die Menge an Manipulationsprimitiven, sowie die Demontageräume, werden weitergehend im Umweltmodell verwendet.

4.2.4 Validierung und Ergebnisse

Folgend sollen die Methoden zur Aufgabenplanung anhand akademischer und praxisrelevanter Baugruppen validiert werden. Zur besseren Verdeutlichung der entwickelten Algorithmen, werden die einzelnen Zwischenschritte im Preprozessor und Prozessor exemplarisch erläutert.

Funktionsweise Preprozessor. Die Funktionsweise des Preprozessors soll mit einem einfachen Beispiel verdeutlicht werden. Als Eingangsdaten wird das CAD-Baugruppenmodel, Abbildung 4-7 (a) verwendet. Darauf aufbauend wird die Information in die Struktur des relationalen Baugruppenmodells gegliedert und der Graph mit den symbolisch, räumlichen Relationen generiert, Abbildung 4-7 (b). Mittels des stichprobenbasierten Algorithmus 4-1 aus 4.2.1 erfolgt die Bestimmung der relativen Demontageräume für die einzelnen Bauteile, Abbildung 4-7 (d).

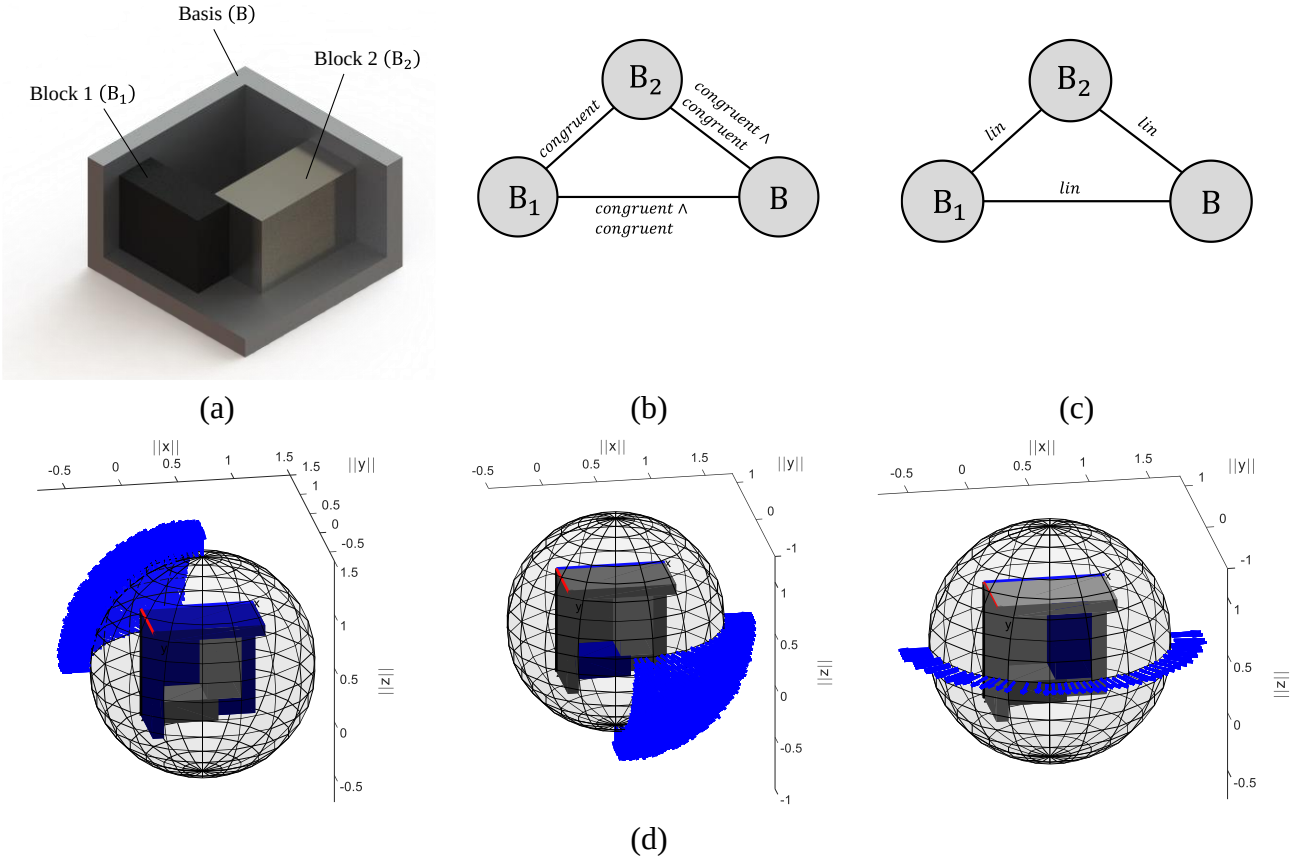
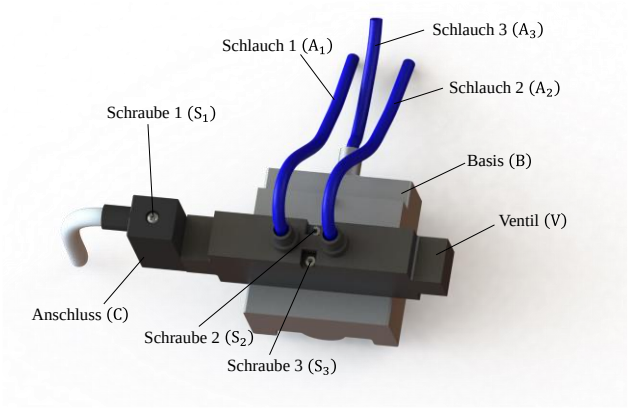


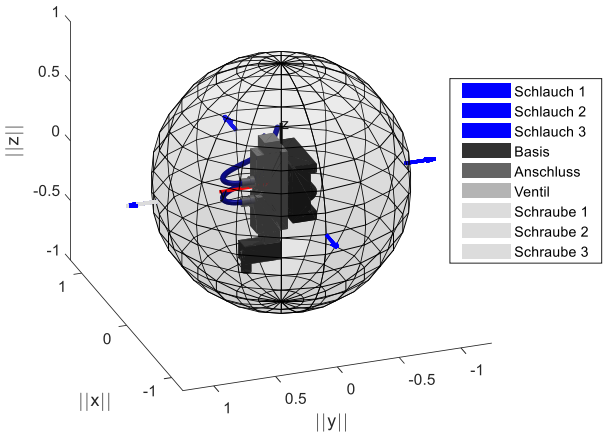
Abbildung 4-7: Arbeitsweise des Preprozessor anhand eines Blockbeispiels

Die Auswahl einer geeigneten Demontagerichtung aus dem Lösungsraum muss die Eigenschaft der Sichtbarkeit erfüllen, siehe 4.4.3. Aus den Demontageräumen lässt sich wiederum der relationale Graph ableiten, der für die symbolische Planung verwendet wird, siehe Abbildung 4-7 (c).

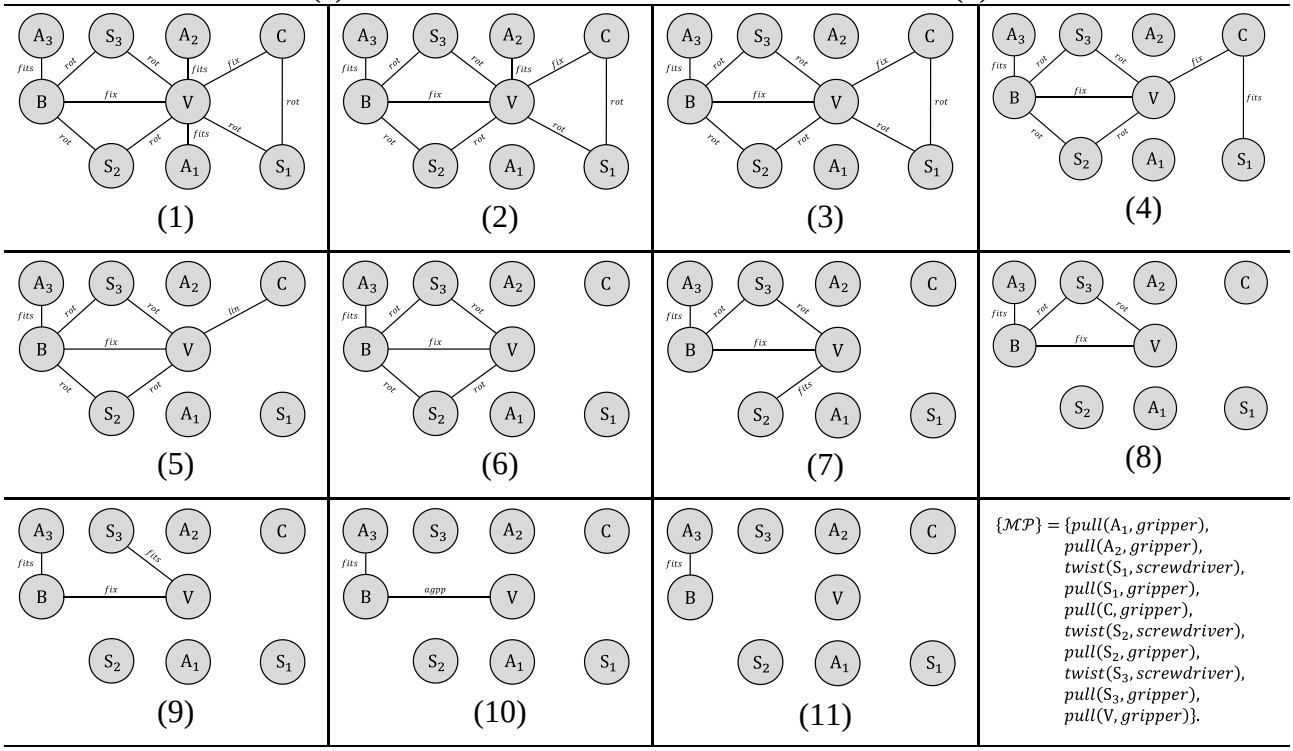
Funktionsweise Prozessor. Die Funktionsweise zur Planung der Manipulationsprimitivmenge soll am Beispiel einer Ventilbaugruppe verdeutlicht werden. Das Planungsziel stellt die Dekomposition des Ventils von der Baugruppe dar. In Abbildung 4-8 sind die einzelnen Zwischenschritte der Planung visualisiert. Der Prozessor beginnt mit der symbolischen Manipulation derjenigen Komponente mit den geringsten Verbindungskosten nach (4.18), hier Schlauch 1.



(a)



(b)



(c)

Abbildung 4-8: Arbeitsweise des Prozessors: CAD-Baugruppenmodell (a) und berechnete Demontageräume des Preprozessors (b); Erzeugte relationale Graphen mit zugehöriger Menge an Manipulationsprimitiven (c)

Bei gleichen Verbindungskosten zwischen verschiedenen Bauteilen (Schlauch 2) erhält das Bauteil Vorrang, welches als erstes in den Teilgraphen eingefügt wurde. Die abgeleiteten Werkzeuge können zusätzlich noch verschiedene Attribute besitzen (beispielsweise Zwei- oder Drei-Finger-Greifer). Prinzipiell können beliebige Werkzeuge in das Planungstool integriert werden.

Validierung Gesamtsystem und Ergebnisse. Der symbolische Manipulationsplaner wird weitergehend an verschiedenen Baugruppen validiert. Dabei sollen neben der einzelnen Lösungen der Pläne auch die Planungszeiten berücksichtig werden. Die Validierung des Planungssystems findet auf einem Standard-PC (Windows 10) mit Intel Core i5-6200U, 2.3GHz mit 8GB DDR3 RAM statt. Zur Validierung werden folgende Metriken verwendet:

- Durchschnittliche Zeit für die Vorverarbeitung t_{pre} in [s] pro Bauteil (Preprozessor).
- Durchschnittliche Zeit für die Planung t_p in [s] pro Dekomposition (Prozessor).
- Anzahl an benötigten Manipulationsprimitiven $|\{\mathcal{MP}\}|$ zur Planerfüllung.

Die Validierung erfolgt anhand sechs unterschiedlicher Baugruppen, gemäß Abbildung 4-9 (zeigt auch die Ergebnisse aus dem Preprozessor), die sich an die Arbeiten [108], [252], [227] orientieren.

Die Ergebnisse sind in Tabelle 4-1 zusammengefasst. $|C|$ stellt die Anzahl an Komponenten und $|sdof|$ die Anzahl an Kanten im relationalen Graph der Baugruppe dar. Die zu entfernende Komponente ist wieder mit C_D bezeichnet. Alle Experimente wurden zur Zeitmessung zehnmal wiederholt und sind als arithmetisches Mittel angegeben.

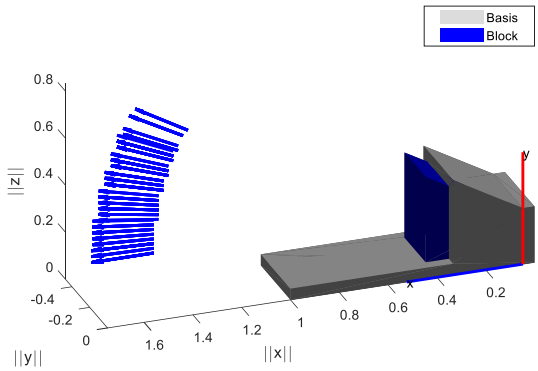
Tabelle 4-1: Ergebnisse der Manipulationsplanung

Baugruppe	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**
$\|C\| \in \mathbb{N}_+$	2	3	4	5	8	9
$\|ssr\| \in \mathbb{N}_+$	5	5	7	13	38	15
$\|sdof\| \in \mathbb{N}_+$	1	3	5	10	13	11
C_D	Block	Block 1	Platte	Platte 2	Bodenplatte 1	Ventil
t_{pre} *in* [s]	0.058	0.047	0.039	0.058	0.075	0.038
t_p *in* [s]	0.004	0.007	0.007	0.006	0.008	0.010
$\|\{\mathcal{MP}\}\| \in \mathbb{N}_+$	1	1	5	1	1	10

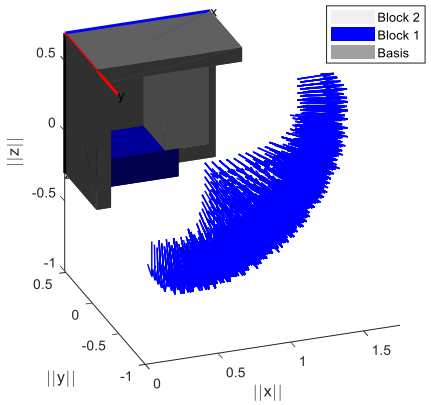
Es zeigt sich prinzipiell ein gleichbleibendes Verhalten der Vorverarbeitungs- und der Planungszeit über alle Baugruppen hinweg. Die Variation in der Zeit ist hauptsächlich auf die Betriebssystemumgebung zurückzuführen. Durch den effizienten, stichprobenbasierten Algorithmus, zur Berechnung der relativen Demontageräume, können in kurzer Zeit mögliche Demontagerichtungen vorgeschlagen und der relationale Graph erzeugt werden.

Durch den nachfolgenden Planungsvorgang mit linearer Komplexität, wird dem Robotersystem umgehend eine Menge an Manipulationen zur Verfügung gestellt, die zur Planerfüllung benötigt werden. Die Effizienz der hier vorgeschlagenen Mechanismen erlaubt erstmalig sinnvoll die Nutzung einer automatisierten Aufgabenplanung im Bereich der Instandhaltungsrobotik.

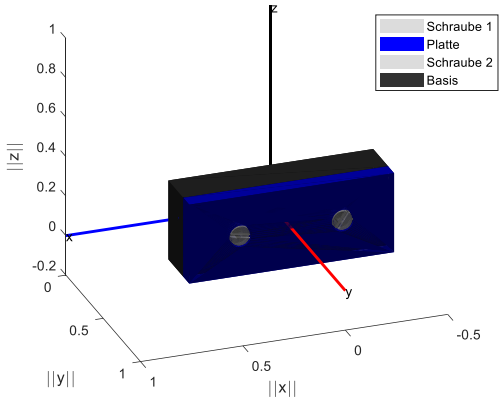
Nachdem gezeigt werden konnte, wie sich die effiziente Planung von Manipulationsaufgaben auf Basis von CAD-Daten gestalten lässt, soll im folgenden Unterkapitel untersucht werden, wie visuelle Information, ebenso für den hier vorgestellten Planer, verwendet werden kann. Erst die Möglichkeit zusätzliche aktive Information in die Planung miteinzubeziehen, erlaubt die Kompensation von Unsicherheiten zwischen passiver und aktiver Information, wodurch die praktische Nutzbarkeit für die automatisierte Instandhaltung erst gegeben ist.



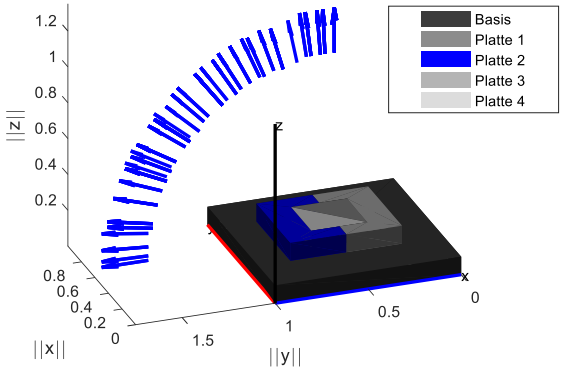
Baugruppe (1)



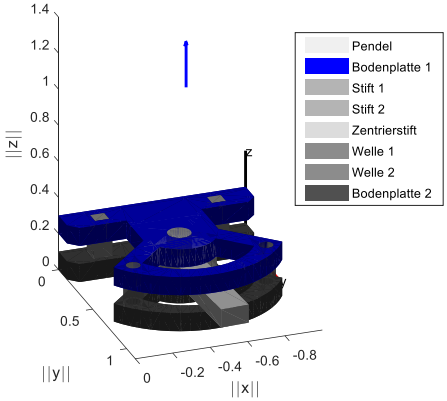
Baugruppe (2)



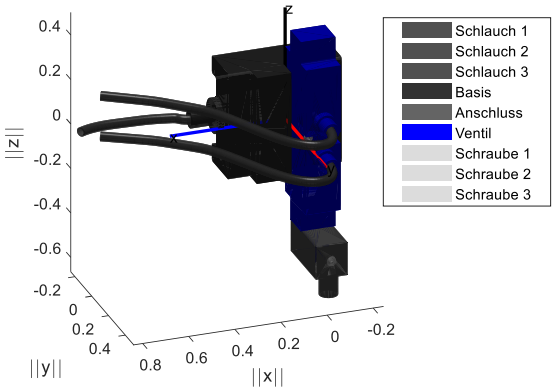
Baugruppe (3)



Baugruppe (4)



Baugruppe (5)



Baugruppe (6)

Abbildung 4-9: Baugruppen zur Validierung des Manipulationsplaners: Berechneter relativer Demontageraum zu aktuellem Baugruppenzustand aus dem Preprozessor ist jeweils in blau für C_D dargestellt

4.3 Visuelle Perzeption

Folgend wird ein visuelles Perzeptionssystem vorgestellt, welches den Anforderungen gemäß Kapitel 3 entspricht. Hierzu wird ein System entwickelt, dass sowohl die Objekterkennung und -lokalisation als auch die Szenenanalyse und Kartierung in einem Framework zur Verfügung stellt. Die erkannten und lokalisierten Objekte werden weitergehend in 4.4.1 mit den CAD-Daten fusioniert, wodurch ein konsistentes Umweltmodell für die Planung zur Verfügung steht. Auch wird ein Verfahren entwickelt, dass die Bildung eines Kontaktgraphs mit symbolisch, räumlichen Relationen erlaubt, wodurch der Kontaktgraph aus dem CAD um neue Objekte auf Basis der Umweltdaten erweitert werden kann. Hierdurch können auch Manipulationspläne für partiell unbekannte Baugruppen geplant werden. Durch die Registrierung der einzelnen Sensorposen in einer probabilistischen 3D-Voxelkarte steht gleichzeitig eine, für die Bahnplanung geeignete, metrische Umgebungsbeschreibung zur Verfügung.

Das vorliegende Unterkapitel zeigt weitergehend, die verwendeten Perzeptionsstrategien zur Objekterkennung und –lokalisation auf und geht auf die entwickelte Szenenanalyse ein. Darauf aufbauend wird die Kartierung und die verwendete Datenstruktur näher vorgestellt, da diese Eigenschaften, sowohl für die Entwicklung eines aufgabenorientierten Next-Best-View Algorithmus (4.4.2) als auch für eine neuartige Methode für die Bahnplanung (4.5), von tragender Bedeutung sind. Abschließend wird auf die Architektur des entwickelten, visuellen Frameworks eingegangen. Teile dieses Unterkapitel sind bereits in der Vorarbeit [19] veröffentlicht.

4.3.1 Visuelle Perzeptionsstrategien

In diesem Abschnitt werden visuelle Perzeptionsstrategien entwickelt, um die aktive Information, welche für die Manipulationsplanung benötigt wird, aus den Sensordaten zu bestimmen. Dazu gehören im Wesentlichen die Objektsemantik, die Objektpose sowie geometrische Objektentitäten, gemäß den Definitionen aus 4.1, die zur Bildung des symbolisch, räumlichen Kontaktgraphen beitragen.

Als visuelle Sensordaten werden zur Ableitung dieser Informationen sowohl Farb- (RGB) als auch Tiefendaten (D) verwendet. Dabei seien die Eingabedaten I^c, beschrieben im Kameraframe, gegeben durch

$$I^c = \langle I_{RGB}, I_D \rangle. \tag{4.19}$$

Nach der Vorverarbeitung werden diese in zwei unabhängigen Teilprozessen verwendet, sodass zum einen die Modellschätzung für die geometrische Primitive, zum anderen die Objekterkennung erfolgen kann. Auf Basis der geometrischen Primitive lassen sich Kontaktrelationen ableiten, die zur Bestimmung des symbolisch, räumlichen Kontaktgraphs aus den Sensordaten verwendet werden können. Mittels der Objekterkennung erfolgt die semantische Annotation der einzelnen Objekte, die zusätzlich für die Szenenanalyse verwendet werden kann. In Abbildung 4-10 sind die einzelnen Schritte der aktiven Informationsgewinnung anhand eines Beispiels aufgeführt. Die Vorverarbeitung stellt der Modellschätzung für die geometrische Primitive, sowie der Objekt-erkennung die zentrale Objektregion (ROI) mit zugehörigen Cluster bereit. Die Ausgaben beider Prozesse werden dann in der Szenenanalyse kombiniert, um einen Kontaktgraphen auf Basis aktiver Information abzuleiten. Die Verfahren werden folgend detailliert besprochen.

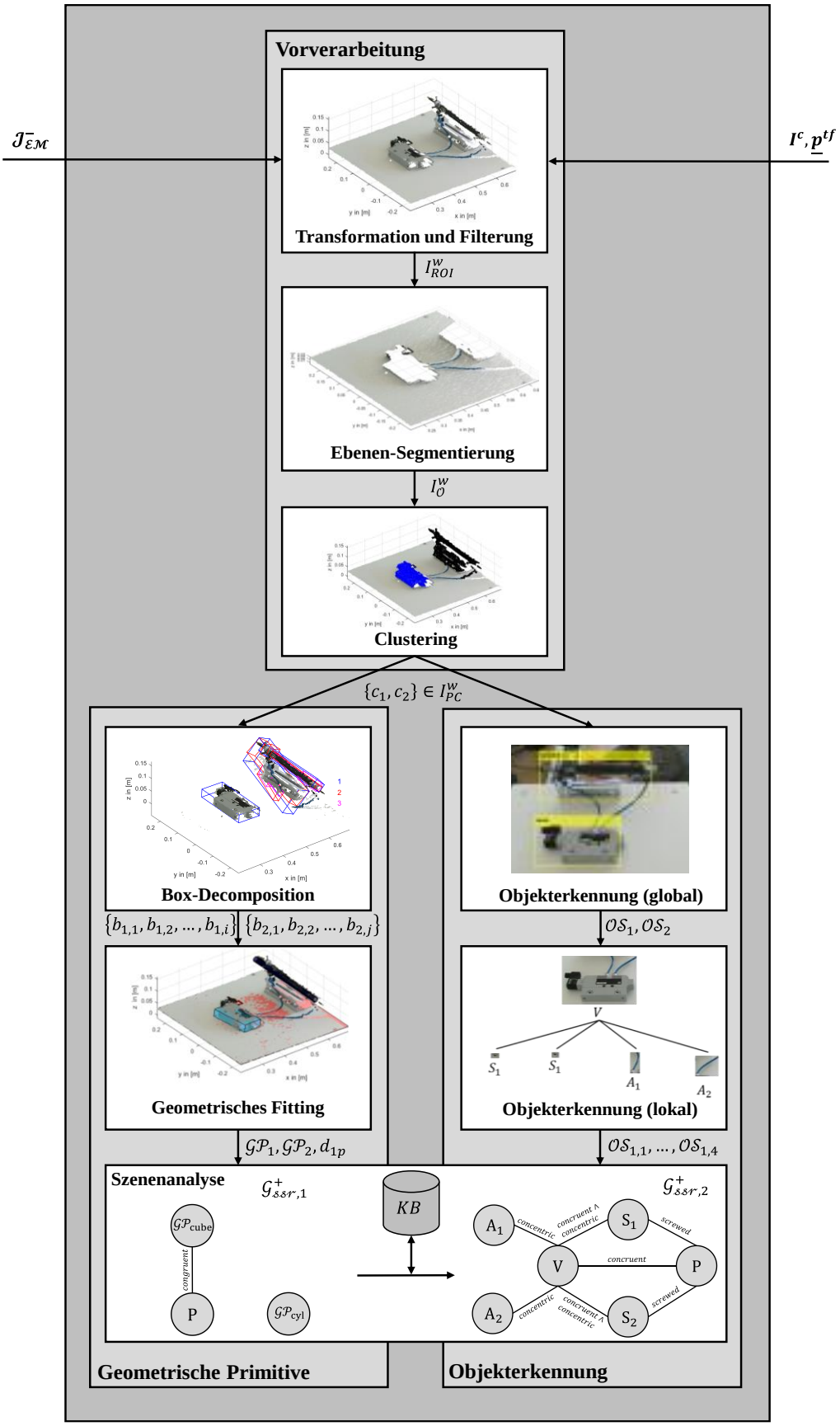


Abbildung 4-10: Perzeptionsschritte zur Bestimmung der aktiven Umweltinformation

Vorverarbeitung. Im Vorverarbeitungsschritt werden die Eingabedaten I^c im Kamerakoordinatensystem für die nachfolgenden Schritte aufbereitet. Hierzu werden alle Sensordaten $\underline{p}^c \in I^c$ im gemeinsamen Weltkoordinatensystem $\underline{p}^w \in I^w$ nach (4.20) registriert.

$$\underline{p}^w = {}^{w}_{tcp}\underline{T} \cdot {}^{tcp}_{c}\underline{T} \cdot \underline{p}^c, mit\ {}^{w}_{tcp}\underline{T}, {}^{tcp}_{c}\underline{T} \in SE(3). \tag{4.20}$$

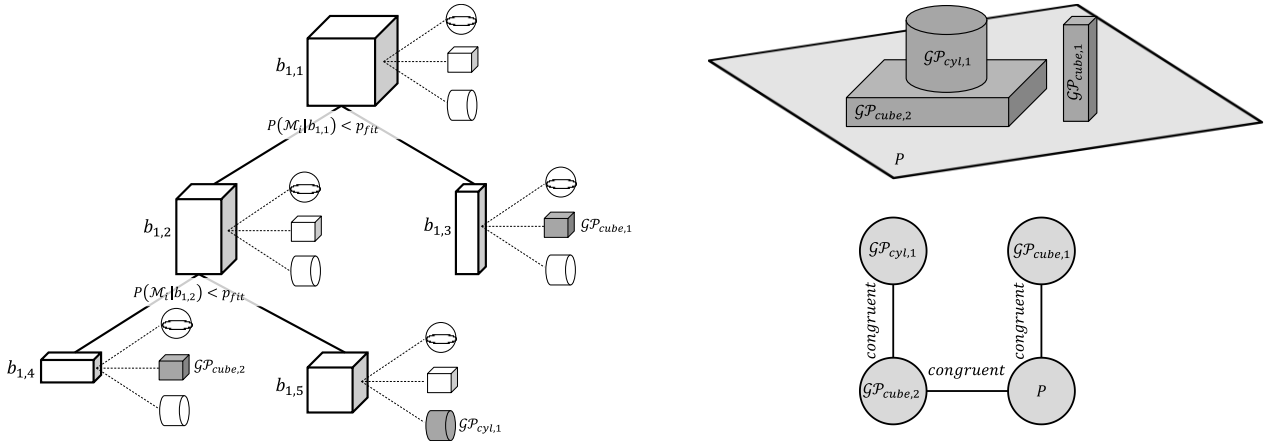
Weitergehend wird nur die ROI betrachtet, wobei sich diese aus der passiven Umweltinformation $\mathcal{J}^-_{\mathcal{EM}}$ ableiten lässt. Zur Rauschunterdrückung wird ein Medianfilter für I_{RGB} und ein Ausreißfilter nach [341], für I_D, eingesetzt, welches die Anzahl der nächsten Nachbarn beachtet. Das Ergebnis I^w_{ROI} wird im nachfolgenden Schritt für die Segmentierung der Ebene verwendet. Hierzu wird angenommen, dass alle Objekte auf einer Ebene montiert sind, wobei die Ebenenorientierung durch den Normalenvektor $\underline{n}$ gegeben ist und aus $\mathcal{J}^-_{\mathcal{EM}}$ bestimmt werden kann. Zur Plausibilitätsprüfung können zusätzlich die Normalenvektoren aus I_D bestimmt und verwendet werden. Die Ebenensegmentierung erfolgt über den MLESAC-Algorithmus [201].

Die möglichen Objektkandidaten I^w_O werden mittels DBSCAN [198] geclustert, sodass diese separiert betrachtet werden können. Zur Reduktion der Rechenzeit für das Clustering erfolgt vorhergehend die Anwendung eines Voxel-Grid-Filters auf I_D. Als Ergebnis werden dem Modul zur Modellschätzung und zur Objekterkennung die einzelnen Punktwolkencluster $\{c_1, c_2, \dots, c_N\}$ bereitgestellt.

Geometrische Primitive und Kontaktrelationen. Die Modellschätzung der geometrischen Primitive, auf Basis der Punktwolkencluster, erfolgt über das in [200] vorgeschlagene Verfahren, welches auf dem MLESAC-Algorithmus basiert. Für die geometrische Primitive werden als mögliche Modelle $\mathcal{GP} \in \{cube, cylinder, sphere\}$ festgelegt. Dies liegt nahe, da die meisten industriellen Komponenten aus Regelgeometrien abgeleitet werden können. Damit möglichst viele, auch vorab unbekannte Objekte, einem geometrischen Primitiv zugeordnet werden können, erfolgt eine Unterteilung der Objektcluster $\{c_1, c_2, \dots, c_N\}$ in unterteilende Bounding-Boxes $\{c_1 \rightarrow \{b_{1,1}, b_{1,2}, \dots, b_{1,i}\}, c_2 \rightarrow \{b_{2,1}, b_{2,2}, \dots, b_{2,j}\}, \dots, c_N \rightarrow \{b_{N,1}, b_{N,2}, \dots, b_{N,k}\}\}$.

Die Bestimmung dieser basiert auf dem Box-Decomposition Algorithmus [342]. Dieser bestimmt eine Bounding-Box mit minimalem Volumen, welche das Objektcluster komplett umfasst. Iterativ erfolgt eine weitere Unterteilung, auf Basis einer zweidimensionalen Projektion der Punkte, auf drei orthogonal zueinanderstehende Quaderseiten, welche die Fläche bei einer Aufteilung in zwei Rechtecke minimiert. Als Lösung im dreidimensionalen wird der Schnitt akzeptiert, welcher die Fläche am meisten minimiert und noch ein vorgegebenes Minimalvolumen aufweist. Für jede Bounding-Box $b_{k,l}$ erfolgt eine Modellschätzung $P(\mathcal{M}_i|b_{k,l}), \forall i \in \mathcal{GP}$. Wird ein vorgegebener Schwellwert der Modellanpassung p_{fit} überschritten, also $P(\mathcal{M}_i|b_{k,l}) > p_{fit}$, wird die Iteration zur Bildung des Box-Decomposition Baums für zugehörigen Knoten $b_{k,l}$ terminiert.

So lassen sich aus den eingehenden Punktwolkenclustern geometrische Modelle ableiten, die weitergehend für die Bestimmung von Kontaktinformationen Verwendung finden. In Abbildung 4-11 (a) ist dieses Verfahren an einem Beispiel exemplarisch dargestellt. Die initiale Bounding-Box $b_{1,1}$ stellt die Wurzel dar. Da für die Modellschätzung $P(\mathcal{M}_i|b_{1,1}) < p_{fit}$ gilt, erfolgt eine Unterteilung in $b_{1,2}$ und $b_{1,3}$. Für $b_{1,3}$ wird ein valides Modell geschätzt, während $b_{1,2}$ weiter unterteilt wird.



(a) (b)

Abbildung 4-11: Verfahren zur Bestimmung geometrischer Primitive und Kontaktinformationen

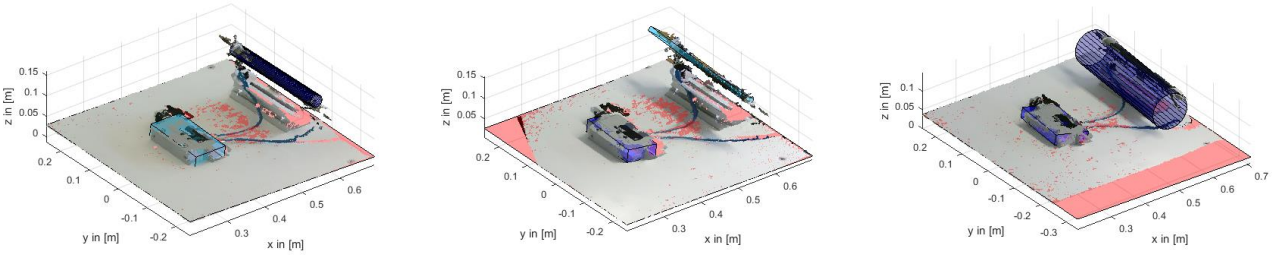
Für die neuen Blätter $b_{1,4}$ und $b_{1,5}$ können wiederum gültige Modelle geschätzt werden, sodass der Algorithmus auch für diese Knoten terminiert. Die Bestimmung möglicher Kontaktpunkte basiert auf dem euklidischen Abstandsmaß d_{ij} zwischen $\mathcal{GP}_i$ und $\mathcal{GP}_j$ gemäß Gleichung (4.21).

$$d_{ij} = \frac{\left\| d\left(\underline{E_i}, \underline{p_i}\right) - d\left(\underline{E_j}, \underline{p_i}\right) \right\|_2 + \left\| d\left(\underline{E_i}, \underline{p_j}\right) - d\left(\underline{E_j}, \underline{p_j}\right) \right\|_2}{2}. \tag{4.21}$$

Dabei stellen $\underline{E_i}$ und $\underline{E_j}$ die Kontaktebenen dar, welche zwischen zwei geometrischen Primitiven $\mathcal{GP}_i$ und $\mathcal{GP}_j$ korrespondieren und innerhalb einer Winkeltoleranz liegen. Da für Kugeln keine Kontaktebenen existieren, wird hier ein Kontaktpunkt verwendet. Für $d_{ij} < d_{tresh}$ wird ein Kontakt angenommen. Das Ableiten einer möglichen Kontaktrelation aus der Menge $\{\mathcal{ssr}\}$ kann dann, auf Basis der Kombination von $\mathcal{FG}_{\mathcal{ssr},i}$ und $\mathcal{FG}_{\mathcal{ssr},j}$, erfolgen. In Abbildung 4-11 (b) ist die Anordnung der geometrischen Primitive mit zugehörigem Graph $\mathcal{G}_{\mathcal{ssr}}$, für das zuvor eingeführte Beispiel dargestellt. Es sei angemerkt, dass sich in der Praxis hauptsächlich deckungsgleiche Beziehungen bestimmen lassen. Allein die Schlussfolgerung einer konzentrisch-Verknüpfung erweist sich, aufgrund von Okklusion, als schwierig. Auch lässt sich nicht eindeutig sicherstellen, dass es sich bei einer abgeleiteten Kontaktrelation um zwei unabhängige Komponenten handelt. Hierzu ist zwingend semantische Information notwendig, welche den geometrischen Primitiven ihre Semantik zuordnet. Dies wird im Laufe des Unterkapitels noch diskutiert.

Das entwickelte Verfahren soll, anhand eines Anwendungsfalls aus Pneumatikventil und -zylinder, untersucht werden. Hierzu werden drei unterschiedliche Kameraposen für die Aufnahme der Szene gewählt, siehe Abbildung 4-12. Zusätzlich sind die geschätzten geometrische Primitive visualisiert. Damit ein Modell als gültig anerkannt wird, soll $p_{fit} \geq 0.7$ gelten, da dies für eine Approximation für gewöhnlich ausreicht. Für die Validierung werden folgende Punkte beachtet:

- Anzahl der Punkte $|P| \in I_D$.
- Zeit für die Modellschätzung $t_{\mathcal{GP}}$ in [s].
- Tiefe des Box-Decomposition Baums d_{BD}.
- Anteil an gefitteten Modellpunkten (Schwellwert der Modellanpassung) mit $p_{fit} \in [0 \ldots 1]$.
- Güte der Modellschätzung $\mathcal{M}_{fit} \in [0 \ldots 1]$ über alle zugehörigen Modellparameter.



(a) (b) (c)

Abbildung 4-12: Ergebnisse der Modellschätzung dargestellt im Weltkoordinatensystem: Gefittete geometrische Primitive für Pose 1 (a); Pose 2 (b); Pose 3 (c)

In Tabelle 4-2 sind die Ergebnisse für die drei Ansichten aufgeführt. Die Güte des Verfahrens, zur Schätzung des Quaders für das Pneumatikventil, ist über alle Ansichten hinweg als gut zu bewerten. Für die Ansicht Pose 2, wird der Pneumatikzylinder fälschlicherweise als Quader erkannt. Auch wird in Pose 3 das Zylindermodell stark überschätzt. Dies ist vor allem auf das Sensorrauschen und die verfügbare Messgenauigkeit der verwendeten RGBD-Kamera zurückzuführen. Eine Gütesteigerung kann durch die Registrierung mehrerer Ansichten in einer Gesamtszene erreicht werden. Als Kontaktinformation lässt sich, anhand der geometrischen Primitive, nur die deckungsgleiche Beziehung zwischen Quader und Ebene ableiten, vergleiche Abbildung 4-11.

Tabelle 4-2: Ergebnisse der geometrischen Modellschätzung für unterschiedliche Objektansichten

Ansicht	***Pose 1***	***Pose 2***	***Pose 3***
$\|\boldsymbol{P}\| \in \mathbb{N}_+$	391187	415466	453747
$\boldsymbol{t_{\mathcal{GP}}}\ in\ [s]$	16.7	23.1	23.5
Bauteil C	$\mathcal{GP}_{cyl}$	$\mathcal{GP}_{cube}$	$\mathcal{GP}_{cyl}$
$\boldsymbol{d_{BD}} \in \mathbb{N}_+$	2	2	1
$\boldsymbol{p_{fit}} \in [0 \dots 1]$	0.74	0.86	0.84
$\boldsymbol{\mathcal{M}_{fit}} \in \mathbb{R}_+$	0.98	-	2.1
Bauteil V	$\mathcal{GP}_{cube}$	$\mathcal{GP}_{cube}$	$\mathcal{GP}_{cube}$
$\boldsymbol{d_{BD}} \in \mathbb{N}_+$	1	1	1
$\boldsymbol{p_{fit}} \in [0 \dots 1]$	0.81	0.83	0.72
$\boldsymbol{\mathcal{M}_{fit}} \in \mathbb{R}_+$	1.04	1.02	1.03

Durch die hier entwickelte Methode können Punktwolken automatisch in geometrische Primitive zerlegt und zugehörige, mögliche Kontaktrelationen bestimmt werden. Dies erlaubt es einem Robotersystem, in Kombination mit semantischer Objektinformation, einen möglichen Kontaktgraph zu bestimmen, welcher für die Manipulationsplanung verwendet werden kann. Die Bestimmung der zusätzlich notwendigen semantischen Information wird in nachfolgendem Absatz zur Objekterkennung behandelt.

Objekterkennung. Wie bereits im Stand der Technik 2.3.4 aufgezeigt, existieren bereits zahlreiche geeignete Methoden und Algorithmen für die Objekterkennung. In diesem Teil der Arbeit sollen die

entsprechenden Verfahren geeignet kombiniert werden, sodass eine robuste Objekterkennung, auf Basis erscheinungsbasierter Verfahren, ermöglicht werden kann. Bisher sind diese Verfahren innerhalb der Robotik zumeist für Haushaltsroboter untersucht worden, vergleiche beispielsweise [206], und es ist nicht bekannt wie gut eine Übertragbarkeit auf industrielle Umgebungen, aufgrund der oftmals geringen farblichen Textur, ermöglicht werden kann. Dies soll im Folgenden näher betrachtet werden.

RGBD-Objektdatensatz für industrielle Komponenten. Zum Lernen der Objektklassifikatoren wird ein Datensatz mit typischen industriellen Komponenten auf Basis von RGBD-Daten erstellt und verwendet. Der Objektdatensatz enthält derzeitig sechs unterschiedliche Komponenten und kann prinzipiell beliebig erweitert werden. Eine Skalierung auf größere Datensätze soll nicht beachtet werden. Ein Ausschnitt des Datensatz ist in Abbildung 4-13 dargestellt.

Globale Objekterkennung. Die globale Objekterkennung arbeitet mittels Bag of Visual Words-Ansatz [185], welcher für RGBD-Daten auf Grundlage des Objektdatensatz eingelernt wird. Die Ergebnisse für die Objekterkennung sind in Abbildung 4-14 aufgeführt. Für RGB-Daten wird SURF [176] sowohl als Feature Detektor als auch als Feature Deskriptor, mit einer 64 Merkmalsdimension, verwendet. Hierdurch wird eine skalierungsinvariante Erkennung, bei gleichzeitiger hoher Performance in der Featuredetektion, erreicht. Für die D-Daten erfolgt die Merkmalsbildung mittels HOG-3D Deskriptoren. Das Clustering der visuellen Wörter wird mittels k-means Algorithmus [186] und die nachfolgende Klassifikation durch eine Multi-Klassen Support Vector Machine [171] durchgeführt. Für die globale Objekterkennung werden die einzelnen, in der Vorverarbeitung segmentierten Cluster $c_i \in I_{PC}$, als Eingabe für die Objekterkennung verwendet.

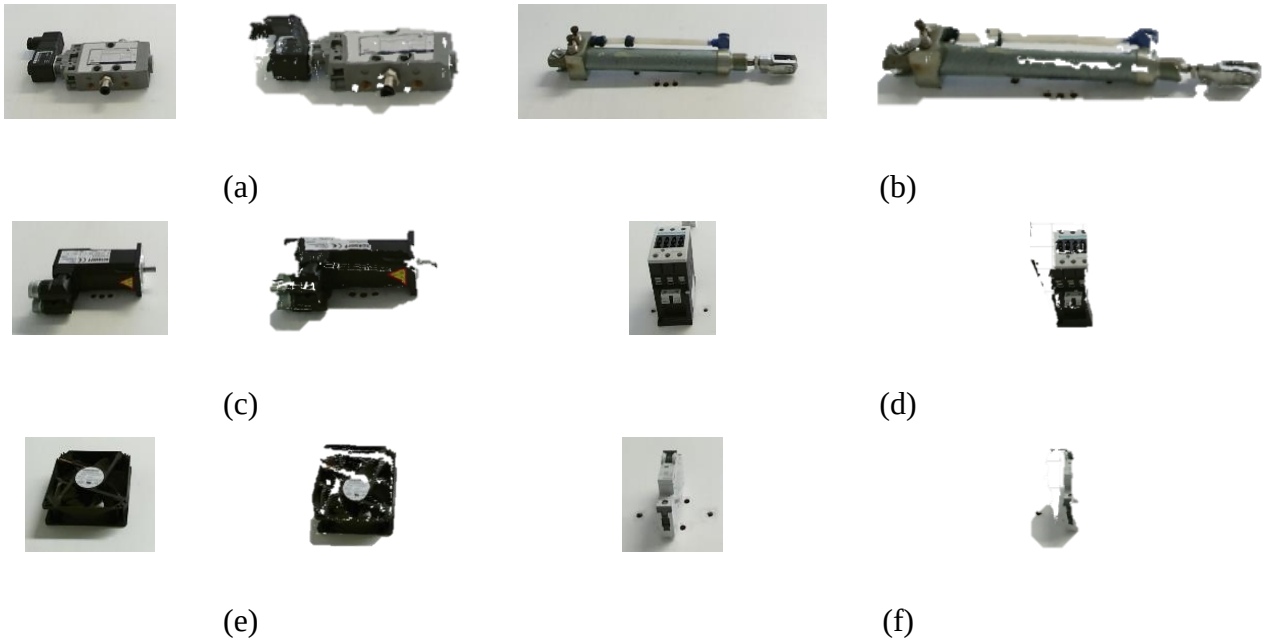


Abbildung 4-13: Ausschnitt aus dem RGBD-Objektdatensatz mit sechs unterschiedlichen, typischen Industriekomponenten: Pneumatikventil (3 Varianten/ 252 Aufnahmen) (a); Pneumatikzylinder (3 Varianten/ 252 Aufnahmen) (b); Servomotor (1 Varianten/ 84 Aufnahmen) (c); Schütz (4 Varianten/ 336 Aufnahmen) (d); Lüfter (4 Varianten/ 336 Aufnahmen) (e); Sicherung (3 Varianten/ 252 Aufnahmen) (f)

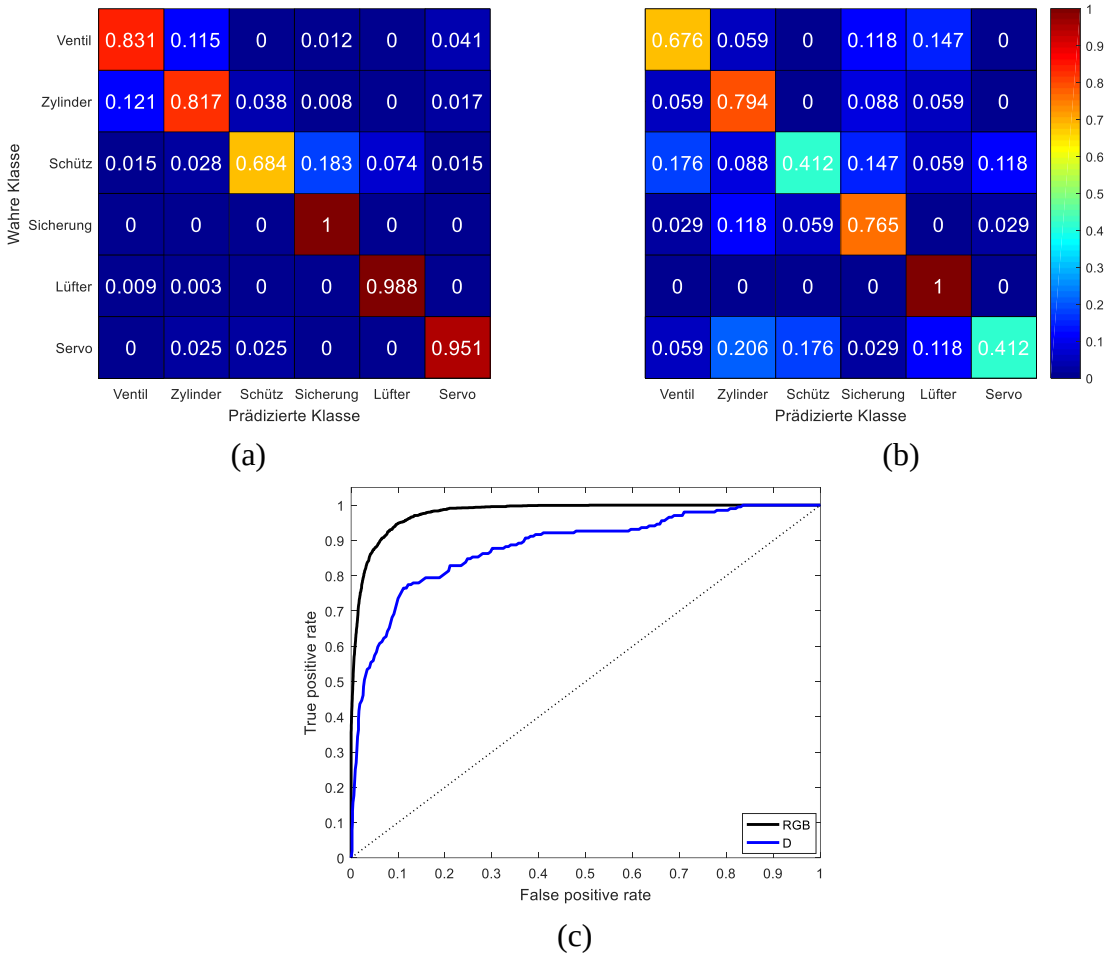


Abbildung 4-14: Ergebnisse globale Objekterkennung mit Bag of Visual Words: Konfusionsmatrix für RGB-Klassifikation (a); Konfusionsmatrix für D-Klassifikation (b); Receiver-Operating-Characteristic (ROC) für RGB- und D-Klassifikation (c)

Diese entsprechen oftmals einer Baugruppe und können nach erfolgreicher Annotation in der lokalen Objekterkennung weiterverarbeitet werden. Es zeigt sich, dass die Erkennung auf Basis von Farbdaten eine höhere Klassifikationsgüte aufweist. Die durchschnittliche Genauigkeit beträgt bei der RGB-Erkennung 88% und bei der D-Erkennung 68%. Die durchschnittliche Zeitdauer für die kombinierte Klassifikation beträgt circa 0.5 Sekunden. Zur Steigerung der Performance für die D-Klassifikation, sollte zukünftig der Einsatz von weiteren Tiefenfeaturedeskriptoren geprüft werden, vergleiche hierzu [184]. Grundsätzlich reichen die true-positive Raten jedoch für eine robuste Erkennung über die einzelnen Klassen hinweg, wenn die Klassifikationsergebnisse aus beiden Detektoren geeignet kombiniert werden.

Lokale Objekterkennung. Die lokale Objekterkennung arbeitet auf Basis der in der globalen Objekterkennung annotierten Objekte $\{\mathcal{OS}_1, \mathcal{OS}_2, \ldots, \mathcal{OS}_N\}$ und deren zugehörigen Farbdaten. Diese übernimmt die Aufgabe einzelne Elementarobjekte, wie Verbindungselemente (Schrauben, Stifte, usw.) oder Energieübertragungselemente (Kabel, Luftschlauch, usw.) zu erkennen, welche mit dem demontierenden Objekt verbunden sind. Aufgrund der praktischen Relevanz wird hier ein Verfahren entwickelt, welches sich besonders gut für die Detektion von Schrauben eignet.

Das hier entwickelte Verfahren zeigt eine deutlich höhere Erkennungsrate gegenüber dem Stand der Technik auf [96], benötigt hierfür jedoch eine geringfügig höhere Auflösung der Objekte. Als Basisverfahren wird der Viola-Jones Detektor (VJD) [178] eingesetzt. Als Feature Deskriptor wird HOG [177] verwendet. HOG hat zwar prinzipiell den Nachteil, dass es keine skalierungsinvariante Eigenschaften aufweist, jedoch können Schrauben, aufgrund ihrer rotationsymmetrischen Formeigenschaft sehr gut in diesem Deskriptor abgebildet werden. Der Einsatz von SURF oder Haar-Deskriptoren zeigt experimentell eine deutlich schlechtere Klassifikationsgüte für die vorliegende Objektklasse auf. Die Histogrammanalyse von HOG, mit unterschiedlichen Schrauben, unterstreicht diese Einsicht, vergleiche Abbildung 4-15. Es zeigt sich eine gleichmäßige und eindeutige Verteilung über unterschiedliche Aufnahmen hinweg. Für das Einlernen des VJD werden HOG-Deskriptoren über eine 8x8 Pixelzelle gebildet, Abbildung 4-15 (a). Dies erlaubt eine ausreichend genaue Merkmalsbeschreibung der Formeigenschaften, auch im Vergleich zu einer feineren Beschreibung über eine 4x4 Pixelzelle, Abbildung 4-15 (b), bei gleichzeitiger Reduktion der Deskriptordimension um circa 80%.

Der VJD ist in 12 Kaskaden gegliedert, mit einer true-positive Rate von 99,5% pro Kaskade. Für das Einlernen werden 400 positive und 180 negative Beispiele verwendet. Damit genügend Information für die Merkmalsbildung vorhanden ist, darf eine minimale Größe von 30x30 Pixeln nicht unterschritten werden. Zur Verringerung der false-positive Rate, bei gleichzeitiger Erhöhung der true-positive Rate, wird das Ergebnis des VJD zusätzlich durch ein BoVW validiert. Für den Deskriptor wird wiederum SURF verwendet, wobei die einzelnen Merkmale auf einem definierten Gitternetz gebildet werden. Es werden 86 positive und ebenso 86 negative (neue) Beispiele für den BoVW verwendet. Eine weitere Steigerung an Lerndaten konnte keine Verbesserung der durchschnittlichen Genauigkeit mehr erzielen. In Abbildung 4-16 sind die Ergebnisse für die Schraubenerkennung dargestellt. Die Konfusionsmatrix (a) zeigt die Klassifikationsgüte für den VJD. Das Erreichen einer true-positiv Rate von 95,6%, bei gleichzeitiger false-positive und false-negative Rate mit jeweils 4,3%, zeigt bereits gute Ergebnisse auf. Die Auswertung der true-negative Rate erfolgt nicht, da hierzu subjektiv Objektkandidaten festgelegt werden müssten.

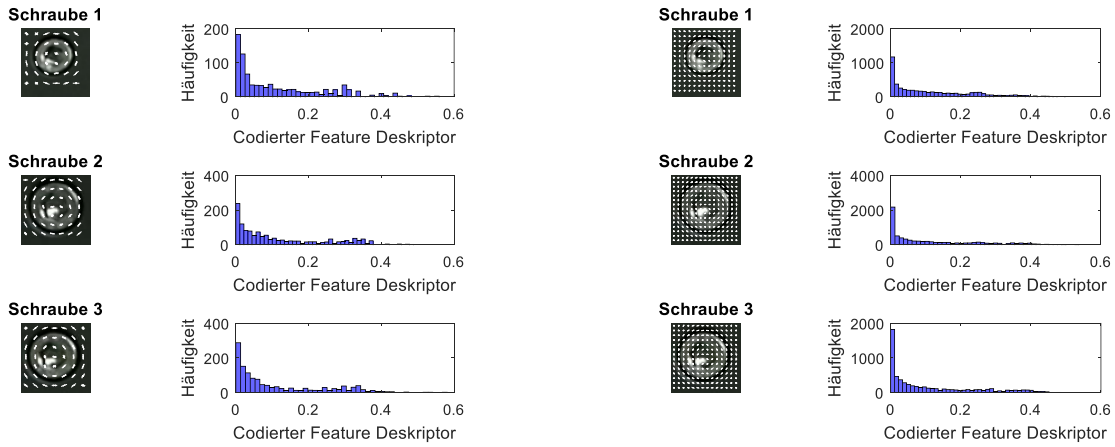
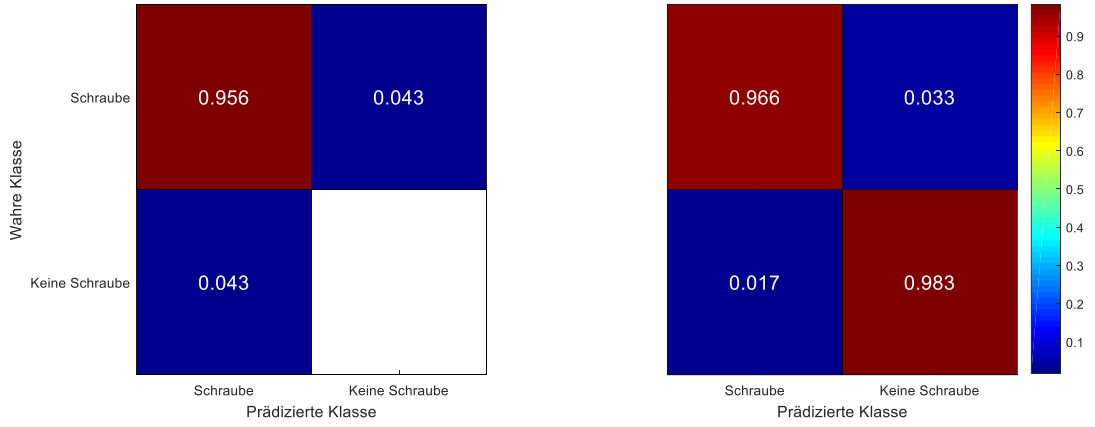


Abbildung 4-15: Untersuchung von HOG-Deskriptoren für die Merkmalsbeschreibung von Schrauben. 8x8 Pixelzelle (a); 4x4 Pixelzelle (b)



(a) (b)

Abbildung 4-16: Ergebnisse lokale Objekterkennung für Schrauben: Konfusionsmatrix für Viola-Jones Detektor (a); Konfusionsmatrix für Nachklassifikation mit Bag of Visual Words (b)

Die Konfusionsmatrix für den BoVW ist in (b) dargestellt. Es zeigt sich, dass 98,3% der vom VJD vorgeschlagenen false-positives richtig erkannt werden. Durch die Kombination beider Verfahren wird eine durchschnittliche Genauigkeit von 96,6% erreicht. Die durchschnittliche Zeitdauer pro Detektion beträgt 0.006 Sekunden.

Szenenanalyse. Die Bereitstellung der geometrischen und semantischen Information erlaubt eine gemeinsame Behandlung zur Szenenanalyse. Somit kann ein möglichst vollständiger Kontaktgraph $\mathcal{G}_{SSR}$, welcher für die Manipulationsplanung benötigt wird, abgeleitet werden. Folgend soll ein möglicher Ansatz diskutiert werden. Die von der Objekterkennung gelieferte Objektsemantik $\mathcal{OS}_i$ soll den einzelnen Primitiven $\mathcal{GP}_j$ zugeordnet werden, sodass die Information zur Bildung des Kontaktgraphs erleichtert werden kann. Hierzu wird die räumliche Ausdehnung E der Objekte beachtet. Prinzipiell lassen sich drei Fälle unterscheiden:

- Die räumliche Ausdehnung des Objekts $E(\mathcal{OS}_i)$ kann genau einem geometrischen Primitiv mit der Ausdehnung $E(\mathcal{GP}_j)$ zugeordnet werden. Es gilt $E(\mathcal{OS}_i) = E(\mathcal{GP}_j)$. $\mathcal{GP}_j$ kann eine eindeutige Semantik $\mathcal{OS}_i$ zugewiesen werden.
- Die räumliche Ausdehnung des Objekts $E(\mathcal{OS}_i)$ kann mehreren geometrischen Primitiven $E(\{\mathcal{GP}_j, \mathcal{GP}_k, \dots, \mathcal{GP}_L\})$ zugeordnet werden. Es handelt sich um ein zusammenhängendes Objekt für welches $E(\mathcal{OS}_i) = E(\mathcal{GP}_j \cup \mathcal{GP}_k \cup \dots \cup \mathcal{GP}_L)$ gilt.
- Die räumliche Ausdehnung des Objekts $E(\mathcal{OS}_i)$ kann keinem geometrischen Primitiv mit der Ausdehnung $E(\mathcal{GP}_j)$ direkt zugeordnet werden. Es lassen sich zwei Unterfälle unterscheiden:
  - $E(\mathcal{OS}_i) \subset E(\mathcal{GP}_j)$, wobei $\mathcal{GP}_j$ das geometrische Primitiv mit dem geringsten euklidischen Abstand d_{ij} ist.
  - $E(\mathcal{OS}_i) \cap E(\mathcal{GP}_j) = \emptyset$, es handelt sich um ein eigenständiges Objekt, für welches kein geometrisches Primitiv gefunden werden kann.

Die Zuordnung der geometrischen Primitiven zur Objektsemantik erleichtert die Szenenanalyse. Beispielsweise können für die Objekterkennung unbekannte Objekte, aufgrund ihres geometrischen Primitiv, in der Manipulationsplanung berücksichtigt werden. Durch das Zusammenfassen mehrerer

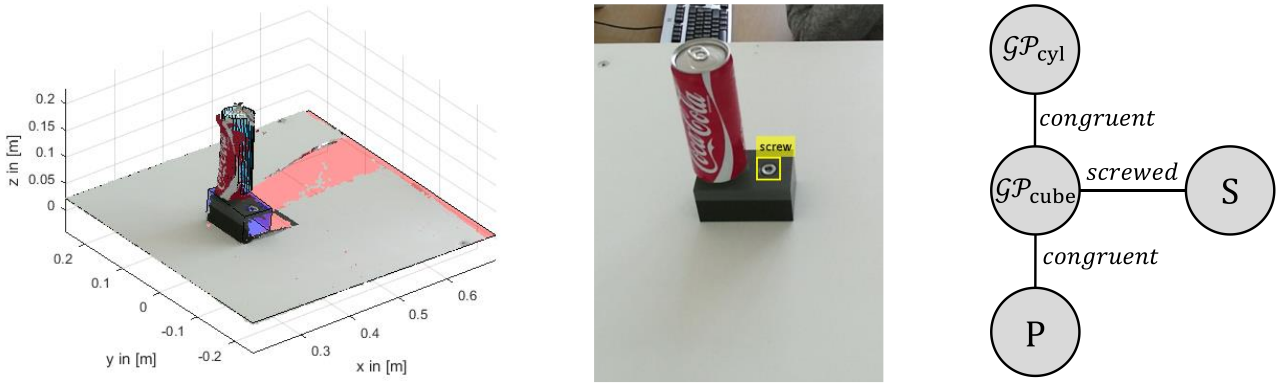
geometrischer Primitive durch die semantische Annotation, können falsch bestimmte Kontaktinformationen verworfen werden. Auch erlaubt die Kombination die Verknüpfung erkannter Objekte, für welche kein geometrisches Primitiv, aufgrund einer zu geringen Ausdehnung, existiert. Die Bestimmung der symbolisch, räumlichen Relationen ohne zwei geometrische Primitive erfolgt auf einer Regelbasis (4.22)-(4.23), welche vorab in einer Wissensdatenbank definiert wird.

$$\forall \mathcal{OS}, \forall \mathcal{GP}\ isObject(\mathcal{OS}) \wedge isPrimitive(\mathcal{GP}) \Rightarrow \mathcal{SSR}_k. \tag{4.22}$$

$$\forall \mathcal{OS}_i, \forall \mathcal{OS}_j\ isObject(\mathcal{OS}_i) \wedge isObject(\mathcal{OS}_j) \Rightarrow \mathcal{SSR}_l. \tag{4.23}$$

Die Kombination aus geometrischer und semantischer, aktiver Information soll, anhand eines Beispiels, untersucht werden. Die Szene besteht aus bekannter Plattenbaugruppe, vergleiche 4.2.4. Eine Schraube wird durch eine Getränkedose verdeckt. Die Klasse Getränkedose und Plattenbaugruppe existiert nicht für die Objekterkennung, sodass eine semantische Zuordnung nicht möglich ist. Es erfolgt eine valide Modellschätzung für die Plattenbaugruppe, mit der Zuordnung eines Quaders $\mathcal{GP}_{cube}$ und der Getränkedose als Zylinder $\mathcal{GP}_{cyl}$ (Abbildung 4-17, (a)). Die Objekterkennung liefert die erkannte Schraube S (Abbildung 4-17, (b)). Über die Abstandsmaße lässt sich dann der Graph mit den Kontaktrelationen angeben, siehe (Abbildung 4-17, (c)). Jedoch macht bereits dieses einfache Beispiel die Grenzen, in der Nutzung visueller Information für die Generierung des Kontaktgraphen, deutlich. Durch die fehlende, semantische Information über Zylinder und Quader, kann nur darauf geschlossen werden, dass es sich um zwei eigenständige Bauteile handelt. Zur rein visuell, sensorbasierten Bestimmung des Kontaktgraph müssten alle Objekte semantisch zugeordnet werden können. Auch dann könnten jedoch nur grundlegende Beziehungen aus den visuellen Daten geschlussfolgert werden.

Um komplexere Probleme zu lösen, könnten mögliche Bewegungsfreiheitsgrade, durch gezieltes Ausprobieren und durch Messung der Interaktionskräfte und –momente, erkannt werden. Dieser Weg soll hier allerdings nicht weiter untersucht werden, da in dieser Arbeit davon ausgegangen werden soll, dass nur kleine Änderungen zwischen passiver und aktiver Information vorhanden sind, für welche diese Methode äußerst geeignet ist.



(a) (b) (c)

Abbildung 4-17: Szenenanalyse durch die Kombination von geometrischer und semantischer Information. Geometrisches Fitting (a); Objekterkennung (b); Szenenanalyse und daraus resultierender Kontaktgraph (c)

4.3.2 Kartierung

Für die Repräsentation von Umgebungskarten bestehen bereits geeignete Methoden, auf welche im Rahmen dieser Arbeit, gemäß dem Stand der Technik, zurückgegriffen wird. Aufgrund der skalierbaren Genauigkeit, der probabilistischen Repräsentation sowie der speichereffizienten Datenstruktur, wird eine dreidimensionale, probabilistische Voxelkarte $\mathcal{PVM}$ nach [159] verwendet, welche auf Basis einer Octree-Datenstruktur arbeitet. Zur Kartierung werden die einzelnen Ansichten gemäß Gleichung (4.20) im gemeinsamen Weltkoordinatensystem registriert.

Dabei sei $\mathcal{PVM} = \bigcup_i^N \mathcal{v}_i$ für eine definierte Skalierung (Kantenlänge der Voxel), eine gegebene Karte mit den Voxeln $\mathcal{v}_i$. Dabei kann ein Voxel drei verschiedene Zustände aufweisen, mit

$$\mathcal{v}_i = \{unbekannt\ u\ , belegt\ o, frei\ f\}. \tag{4.24}$$

Dies ermöglicht die Darstellung der Karte $\mathcal{PVM} = \mathcal{V}_u \cup \mathcal{V}_o \cup \mathcal{V}_f$ in einen unbekannten Raum $\mathcal{V}_u$, einen belegten Raum $\mathcal{V}_o$ und einen freien Raum $\mathcal{V}_f$. Die Beschreibung des unbekannten Raums, kann folglich durch die Differenz mit dem explorierten Raum $\mathcal{V}_e = \mathcal{V}_o \cup \mathcal{V}_f$ mit

$$\mathcal{V}_u = \mathcal{PVM} \backslash \mathcal{V}_e \coloneqq \{\mathcal{v}_i | \mathcal{v}_i \in \mathcal{PVM} \wedge \mathcal{v}_i \notin \mathcal{V}_e\} \tag{4.25}$$

gebildet werden. Die Parametrisierung des Raumes erweist sich, insbesondere für die hier entwickelten Konzepte des aufgabenkombinierten Next-Best-View Algorithmus in 4.4.2 als auch für die adaptive Schrittweitensteuerung für die Bahnplanung in 4.5, als äußerst vorteilhaft.

4.3.3 Visuelles Perzeptionsframework

Die einzelnen Funktionalitäten sind in eine Softwarearchitektur, auf Basis des ROS Frameworks, integriert, sodass sich dieses modular in das Gesamtkonzept einfügen lässt. Abbildung 4-18 zeigt die Architektur mit zugehörigem Informationsfluss auf. Dabei ist das visuelle Framework in zwei Hauptkomponenten untergliedert.

Das Learning Layer dient der Integration neuer Objekte für die Objekterkennung. Vorab unbekannte Objekte müssen manuell ihrer semantischen Klasse zugewiesen werden. Die Generierung geeigneter Objektansichten in Form von Roboterposen, zum Lernen der Klassifikatoren, erfolgt automatisiert.

Das Perzeption Layer beinhaltet alle Funktionalitäten zur Bestimmung des aktiven Umweltmodells, welche in den vorherigen Abschnitten ausführlich besprochen wurden. Zusätzlich wird die Voxelkarte, die für die Bahnplanung bereitgestellt wird, aus der Registrierung der einzelnen Sensorposen erzeugt. Als Eingabedaten für das Perzeption Layer werden sowohl Farb- als auch darauf registrierte Tiefendaten, mit zugehöriger Roboterpose, verwendet.

Aufgrund der Architektur ist derzeit nur eine sequentielle Verarbeitung der eingehenden Sensordaten möglich. Zukünftig sollten die einzelnen unabhängigen Funktionen in unterschiedlichen Tasks abgearbeitet werden, sodass eine Verarbeitung der Sensordaten schneller möglich ist. Auch eine Erweiterung der Szenenanalyse, die bisher nur das Ableiten einfacher Relationen zulässt, würde die Autonomie des Systems weiter erhöhen.

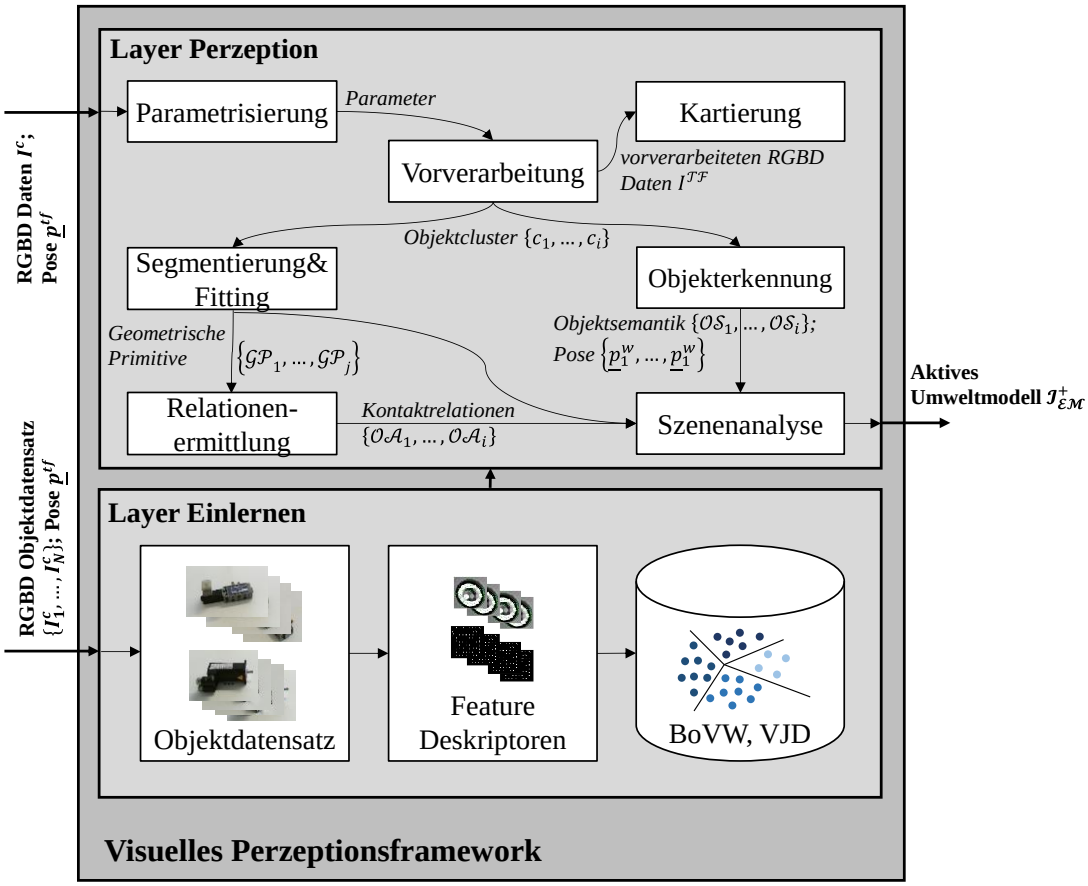


Abbildung 4-18: Architektur des visuellen Perzeptionsframeworks

4.4 Umweltmodell, Aufgabenexploration und Sequenzierung

In den vorherigen Kapiteln wurden Verfahren, zur Beschreibung und Gewinnung von passiver (CAD) und aktiver Information (visuelle Perzeption), entwickelt sowie eine Möglichkeit zur Planung von Roboteraufgaben, mittels Manipulationsprimitiven, geschaffen. Jedoch wurde nicht aufgezeigt, wie sich die beiden Welten von passiver und aktiver Information geeignet verbinden lassen, sodass eine möglichst vollständige und eindeutige Umweltrepräsentation für das Robotersystem vorliegt. Als zentrale Einheit verwaltet das Umweltmodell alle passive und aktive Information und koordiniert die Planungsvorgänge. Die Hauptaufgaben liegen folglich in den Bereichen:

- Fusion und Verwaltung von passiver und aktiver Information.
- Planung von Perzeptionsaufgaben zur zielgerichteten Aufgabenexploration.
- Sequenzierung der Menge an Manipulationsprimitiven.

Hierzu stellt dieses Unterkapitel verschiedene Strategien vor, welche alle auf dem in 4.1 eingeführten Umweltinformationsmodells beruhen. Dies ermöglicht es dem Robotersystem aktuelle Umweltzustände zu aktualisieren und somit selbstständig neue Aktionen, auf Grundlage der in 4.2.2 eingeführten Planungsmethode, zu bestimmen.

Das Unterkapitel beschreibt einführend die Informationsfusion auf Basis einer Bayes-Filterbank. Diese verwaltet den Belief-State des Robotersystems über die Beschaffenheit seiner Umwelt. Darauf aufbauend wird ein Next-Best-View Ansatz entwickelt, der eine parallele Ausführung

verschiedener Perzeptionsaufgaben, auf Basis dynamischer Metriken, erlaubt. Das Unterkapitel schließt mit der Sequenzierung der Menge an Manipulationsprimitiven auf Basis des vorhandenen und konsistenten Umweltmodells. Mittels der Konzepte der Sichtbarkeit und Zugänglichkeit, kann eine praxistaugliche Lösung geschaffen werden, welche die im relativen Demontageraum geplante Menge an Manipulationsprimitiven in eine Sequenz im absoluten Demontageraum überführt.

4.4.1 Probabilistische Fusion von passiver und aktiver Information

Die Fusion von passiver und aktiver Information erfolgt, aufgrund dem Aufbau des Umweltmodells, vergleiche Definition 1-3, auf symbolischer Ebene. Es wird ein probabilistischer Ansatz verfolgt, da dies die Behandlung von Unsicherheiten, auf einem allgemeingültigen Fundament, erlaubt. So können zum einen Unsicherheiten über das Vorhandensein eines passiven Informationselements, als auch Klassifikationsfehler durch die Objekterkennung, geeignet gehandhabt werden.

Beschreibt $P(\mathcal{O}_i|z_t)$ für eine Beobachtung z_t zum Zeitpunkt t die Wahrscheinlichkeit P für die Existenz eines Objekts $\mathcal{O}_i$, kann die Bestimmung des „*degree of belief*“ $\mathrm{bel}(\mathcal{O}_{i,t}) = P(\mathcal{O}_i|z_{1:t})$ einfach über ein binäres Bayes-Filter nach (4.26) erfolgen, also

$$\frac{P(\mathcal{O}_i|z_{1:t})}{1-P(\mathcal{O}_i|z_{1:t})} = \overbrace{\frac{P(\mathcal{O}_i|z_t)}{\left(1-P(\mathcal{O}_i|z_t)\right)\cdot\left(1-P(\mathcal{O}_i|z_{1:t-1})\right)}}^{\xi_i} \cdot P(\mathcal{O}_i|z_{1:t-1}). \tag{4.26}$$

Der initiale Schwellwert $\mathrm{bel}(\mathcal{O}_{i,0}) = \mathrm{bel}(\mathcal{O}_i^-)$ beschreibt das Vertrauen in zugehöriges, passives Informationselement $\mathcal{O}_i^-$. Das Vertrauen $P(\mathcal{O}_i|z_t) = S_{\mathcal{O}_i}$ in eine Sensorbeobachtung z_t wird im inversen Sensormodell $S_{\mathcal{O}_i}$ beschrieben. Die Zuordnung von passiver und aktiver Information erfolgt über die Gleichheit der Semantik $\mathcal{OS}_i^- \triangleq \mathcal{OS}_i^+$ (symbolische Ebene) innerhalb einer festgelegten Fehlerregion der Objektpose. Somit kann für jedes Objekt, gemäß Gleichung (4.27), ein eigener Filter initialisiert und verwendet werden.

$$\begin{pmatrix} \mathrm{bel}(\mathcal{O}_{0,t}) \\ \mathrm{bel}(\mathcal{O}_{1,t}) \\ \vdots \\ \mathrm{bel}(\mathcal{O}_{N,t}) \end{pmatrix} = \mathrm{diag}\left(\overbrace{\frac{\xi_0}{1+\xi_0\cdot\mathrm{bel}(\mathcal{O}_{0,t-1})}}^{\tilde{\xi}_0}, \tilde{\xi}_1, \dots, \tilde{\xi}_N \right) \cdot \begin{pmatrix} \mathrm{bel}(\mathcal{O}_{0,t-1}) \\ \mathrm{bel}(\mathcal{O}_{1,t-1}) \\ \vdots \\ \mathrm{bel}(\mathcal{O}_{N,t-1}) \end{pmatrix}. \tag{4.27}$$

Für die Implementierung wird auf die log-odds Variante des Filters zurückgegriffen [170]. Wird ein festgelegter Schwellwert $\mathrm{bel}(\mathcal{O}_{i,t}) \geq P_{trust,i}$ für einen Objektinformationszustand erreicht, wird dieser als „wahr“ angenommen und nicht mehr weiter im Filterupdate berücksichtigt.

Durch dieses einfache Konzept kann eine gemeinsame Basis, für die Fusion von passiver und aktiver Information, gefunden werden. Posteriori Objektinformation, welche keinem priori definierten Objekt zugeordnet werden kann, wird als neuer Zustand in den Filtergleichungen (4.27) berücksichtigt.

Prinzipiell stellt sich nun die Frage, wie das Robotersystem geeignet neue, aktive Information gewinnen kann, sodass möglichst zielgerichtet ein „wahres“ Umweltmodell für die Planung zur Verfügung steht. Hierzu soll ein Ansatz im nächsten Abschnitt beschrieben werden.

4.4.2 Aufgabenexploration mittels kombiniertem Next-Best-View Ansatz

Damit ein Robotersystem aufgabenorientiert neue, visuelle Information über seine Umwelt gewinnen kann, soll hier eine neue Methode diskutiert werden, die dieses Problem durch eine global, dynamische Gewichtung von Metriken der verschiedenen Aufgaben löst. Die Methode wurde bereits in der Vorarbeit [20] vorgestellt.

Seien T die Anzahl an visuellen Perzeptionsaufgaben $\mathcal{T}_{vp}$, welche der Roboter durch geeignete visuelle Perzeption lösen kann, dann existieren für die unterschiedlichen i-Aufgaben jeweils eine optimale Pose $\underline{p}_i^*$, welche das Informationgain $IG_{\mathcal{T}_{vp,i}}(\underline{p})$ für die spezifische Aufgabe $\mathcal{T}_{vp,i}$ maximiert, also

$$\mathcal{T}_{vp,i}\colon \underline{p}_i^* = \underset{\underline{p}\in\mathcal{W}_{sensor}}{\arg\max}\, IG_{\mathcal{T}_{vp,i}}(\underline{p}). \tag{4.28}$$

Dabei sei $\mathcal{W}_{sensor} \subseteq \mathcal{W}_{robot}$, der für den Sensor erreichbare, kollisionsfreie Raum, unter Berücksichtigung des verfügbaren Messbereichs. Da eine gemeinsame Evaluierungsmetrik für die Optimierung verschiedener Perzeptionsaufgaben nicht existiert, müssen diese anderweitig kombiniert werden. Damit eine „gute“ Sensorpose $\underline{p}_{com}$ zur Lösung aller Perzeptionsaufgaben gefunden werden kann, sollen die für die einzelnen Aufgaben optimalen Posen $\underline{p}_i^*$ dynamisch gewichtet werden. Gleichung (4.29) beschreibt den Zusammenhang der dynamischen Gewichtung.

$$\underline{p}_{com} = \frac{\sum_i^T \alpha_i \cdot \underline{p}_i^*}{\sum_i^T \alpha_i}, mit\ \alpha_i \in]0, \dots, 1]. \tag{4.29}$$

Dabei sei α_i ein dynamischer Gewichtungsfaktor, welcher durch die Metrik der jeweiligen Aufgabe bestimmt wird. Durch die linear, dynamische Gewichtung wird immer die Pose gewählt, welche global den Informationgain aller Aufgaben insgesamt verbessert. Der Zusammenhang erlaubt dabei die stärkere Berücksichtigung der Posen, welche die Aufgaben lösen, die bisher am wenigsten erfüllt sind. Folglich sollen für die fundamentalen Aufgaben der metrischen Exploration und der Objekterkennung geeignete Metriken entwickelt sowie das Gesamtkonzept des kombinierten aufgabenorientierten NBV diskutiert werden.

Next-Best-View für Raumexploration. Die Bestimmung einer, für die metrischen Exploration optimalen Pose $\underline{p}_{exp}^* \in \mathcal{W}_{sensor}$, kann aufgrund der gegebenen Kartenrepräsentation mittels Voxel, auf Basis der Mächtigkeit zwischen belegtem zu unbelegtem Raum definiert werden, also

$$M = \frac{|\mathcal{V}_o| + |\mathcal{V}_f|}{|\mathcal{V}_o| + |\mathcal{V}_f| + |\mathcal{V}_u|}, mit\ M \in [0, \dots, 1]. \tag{4.30}$$

Je kleiner die Mächtigkeit M, desto mehr Raum kann, bei optimaler Wahl der Pose, in einer Sensorbeobachtung exploriert werden.

Sei nun $f_{\mathcal{PVM}}(v_i)$ eine Funktion, welche für die Karte $\mathcal{PVM}$ die Belegtheitszustände der einzelnen Voxel zurückgibt, also

$$f_{\mathcal{PVM}}(v_i) = \begin{cases} v_i \in \mathcal{V}_u \\ v_i \in \mathcal{V}_o, \\ v_i \in \mathcal{V}_f \end{cases} \forall v_i \in \mathcal{PVM}. \tag{4.31}$$

Für eine Pose $\underline{p}$ lässt sich dann die Evaluierungsfunktion, welche den unbekannten Raum beschreibt durch auswerten von

$$f\left(\underline{p}\right) = \left|\sum_{i=1}^{K} f_{\mathcal{PVM}}(\mathcal{v}_i)\right| \circ \underline{p}, \forall \mathcal{v}_i \in \mathcal{V}_u \tag{4.32}$$

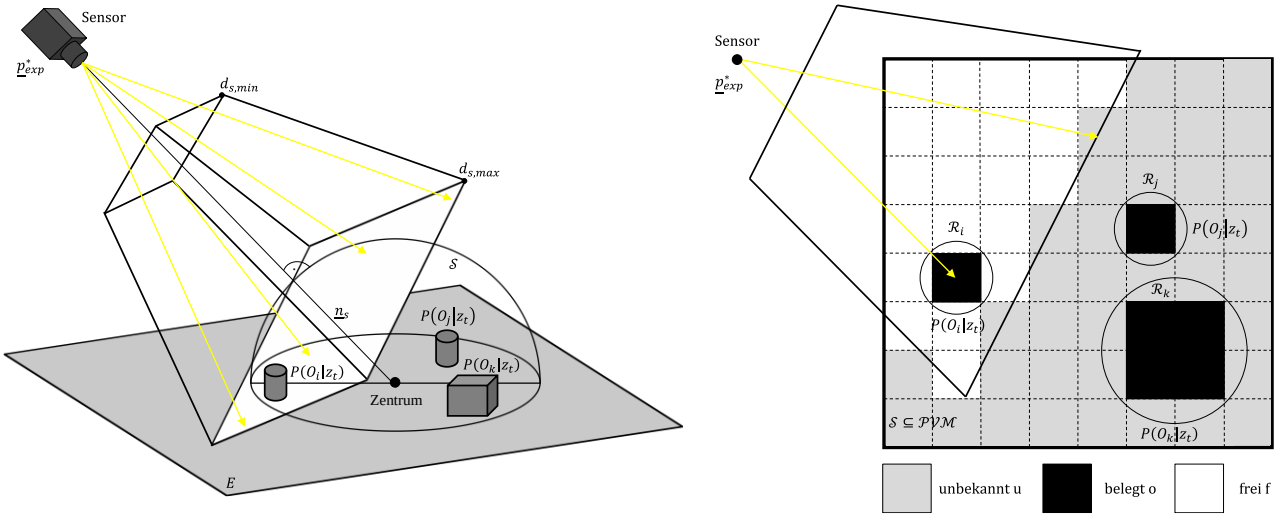
angegeben. Dabei sei K die Anzahl aller Voxel, welche im Sensorblickwinkel liegen. Das Sensormodell kann hier aufgrund seines Messprinzips durch einen Pyramidenstumpf beschrieben werden, vergleiche Abbildung 4-19.

Zur Auswertung der Evaluierungsfunktion (4.32) wird ein voxelbasierter Raytracing-Algorithmus verwendet [343]. Eine optimale Pose für die Exploration $\underline{p}^*_{exp}$ findet sich folglich aus der Maximierung der Evaluierungsfunktion (4.32) mit

$$\underline{p}^*_{exp} = \underset{\underline{p} \in \mathcal{W}_{sensor}}{\arg\max}\, f\left(\underline{p}\right). \tag{4.33}$$

Next-Best-View für Objekterkennung. Die Beschreibung einer Evaluierungsfunktion für die Objekterkennung kann auf verschiedenen Merkmalsebenen erfolgen. Aufgrund der Verwendung unterschiedlicher Segmentierungs- und Klassifizierungsverfahren für die Objekterkennung, eignet sich jedoch besonders ein generalisierter Ansatz, welcher auf der generellen Sichtbarkeit eines Objekts beruht. Dies bedeutet, eine Sensorpose $\underline{p}^*_{rec}$ ist für die Objekterkennung optimal, wenn die maximale Anzahl an Objekten, innerhalb des Sensorblickwinkels, im sichtbaren Voxelraum $\mathcal{V}_V$ liegen. Sei nun $\mathcal{R}_i$ die Region, welche durch ein Objekt O_i in der Karte $\mathcal{PVM}$ abgebildet wird und mit einer Wahrscheinlichkeit von $P(O_i|z_{1:t})$ existiert, dann ist $g_{\mathcal{PVM}}(\mathcal{v}_i)$ eine Funktion, welche prüft, ob ein Voxel $\mathcal{v}_j$ aus $\mathcal{R}_i$ im sichtbaren Raum $\mathcal{V}_V$ liegt, also

$$g_{\mathcal{PVM}}\left(\mathcal{v}_j\right) = \begin{cases} \mathcal{v}_j \in \mathcal{V}_V, wenn\ \mathcal{v}_j \in \mathcal{R}_i \wedge P_{min} \leq P(O_i|z_{1:t-1}) < P_{trust} \\ \mathcal{v}_j \notin \mathcal{V}_V, sonst \end{cases}, \forall \mathcal{v}_j \in \mathcal{R}_i. \tag{4.34}$$



(a) (b)

Abbildung 4-19: Aufgabenorientierter Next-Best-View: Schematischer Aufbau (a); Zuordnung der Objekte in Regionen exemplarisch im zweidimensionalen Raum (b)

In Abbildung 4-19 (b) ist dieser Zusammenhang schematisch im zweidimensionalen Raum dargestellt. Es werden hierzu alle Objekte mit $i = 1, \dots, N$ betrachtet. Die Wahrscheinlichkeiten dienen dazu eine geeignete Metrik für den aufgabenorientierten NBV zu beschreiben sowie für die initiale Pose die passive Information zu berücksichtigen, vergleiche 4.4.1. Äquivalent zur Evaluierung der Exploration erfolgt die Evaluierung der Pose nun durch die Anzahl der Voxel, welche zu einem Objekt gehören durch (4.35).

$$g\left(\underline{p}\right) = \left| \sum_{j=1}^{K} g_{\mathcal{PVM}}\left(\mathcal{v}_j\right) \right| \circ \underline{p}, \forall \mathcal{v}_j \in \mathcal{V}_V. \tag{4.35}$$

Die optimale Pose für die Objekterkennung $\underline{p}^*_{rec}$ maximiert dann die Anzahl an den Objekten zugehörigen Voxeln und ist mit

$$\underline{p}^*_{rec} = \underset{\underline{p} \in \mathcal{W}_{sensor}}{\arg\max}\, g\left(\underline{p}\right) \tag{4.36}$$

gegeben. Zusätzlich können weitere Nebenbedingungen in der Optimierung beachtet werden, wie etwa eine minimale oder maximale Distanz von Sensor zu Objekt.

Globale Aufgabenkombination. Damit nun eine Kombination der Aufgaben gemäß Gleichung (4.29) erfolgen kann, müssen die visuellen Teilaufgaben in einer normierten Metrik gewichtet werden. Für die Exploration lässt sich diese aus der Mächtigkeit M des explorierten Raums folgern und kann direkt durch die Anzahl der unbekannten $\mathcal{v}_u$, der belegten $\mathcal{v}_o$ und der freien Voxel $\mathcal{v}_f$ mit

$$\alpha_1 = 1 - M\left(\mathcal{v}_u, \mathcal{v}_o, \mathcal{v}_f\right), \alpha_1 \in]0, \dots, 1] \tag{4.37}$$

ausgedrückt werden. Die Metrik für die Objekterkennung kann durch die bedingte Wahrscheinlichkeit global über alle Objekte betrachtet werden.

$$\alpha_2 = 1 - \frac{\sum_{i=1}^{N} \mathrm{bel}\left(\mathcal{I}^{\mathcal{S}}_{\mathcal{O}_i,t}\right)}{\sum_{i=1}^{N} P_{trust,i}}, \alpha_2 \in]0, \dots, 1]. \tag{4.38}$$

Die dynamische Gesamtgewichtung für die Aufgabe der Exploration und der Objekterkennung, ist dann formal aus (4.29) ableitbar und mit (4.39) gegeben.

$$\underline{p}_{com} = \frac{1}{\alpha_1 + \alpha_2}\left(\alpha_1 \cdot \underline{p}^*_{exp} + \alpha_2 \cdot \underline{p}^*_{rec}\right) \tag{4.39}$$

Hierdurch können beide Aufgaben dynamisch, ihrer Relevanz entsprechend, berücksichtigt werden. Es ist prinzipiell leicht einsehbar, dass die linear Verknüpfung der zwei nichtlinearen Evaluierungsfunktionen (4.32) und (4.35) nicht die optimale Lösung in der Gesamtheit beschreibt. Jedoch lassen sich hierdurch unterschiedliche, visuelle Aufgaben auf einfache Weise in einer Sensorpose verknüpfen. Durch die dynamische Gewichtung wird immer die Aufgabe bevorzugt, über welche weniger Information existiert, was sich in der Praxis bei ungünstigen Nichtlinearitäten in der Anzahl der benötigten Sensorposen wiederspiegeln würde, im schlechtesten Fall (verharren in lokalem Minima) jedoch in einer unvollständigen Aufgabenexploration. Falls durch eine ungünstige Szene eine Aufgabe nicht oder nur teilweise erfüllt werden kann, besteht die

Möglichkeit eine Aufgabe zu bevorzugen und die dynamische Gesamtgewichtung nicht vorzunehmen. Damit eine Lösung für die Gleichungen (4.33) und (4.36) gefunden werden kann, wird in dieser Arbeit der Twiddle-Algorithmus (Algorithmus 4.3), vergleiche hierzu [344], aufgrund seiner recheneffizienten Struktur verwendet, der hier für den $\mathbb{R}^3$ erweitert wird. Als Eingabe erhält der Twiddle-Algorithmus die initiale Pose $\underline{p}$ sowie den Schrittweitenvektor $d\underline{p}$, welcher die Änderungsschrittweite für die Optimierung beschreibt.

Algorithmus 4-3: Twiddle-Algorithmus

Eingabe: Probabilistische Voxelkarte $\mathcal{PVM}$; Initiale Pose $\underline{p} \in \mathcal{W}_{sensor}$; Schrittweitenvektor der Pose $d\underline{p}$; Änderungsparamter γ; Abruchschranke ε.

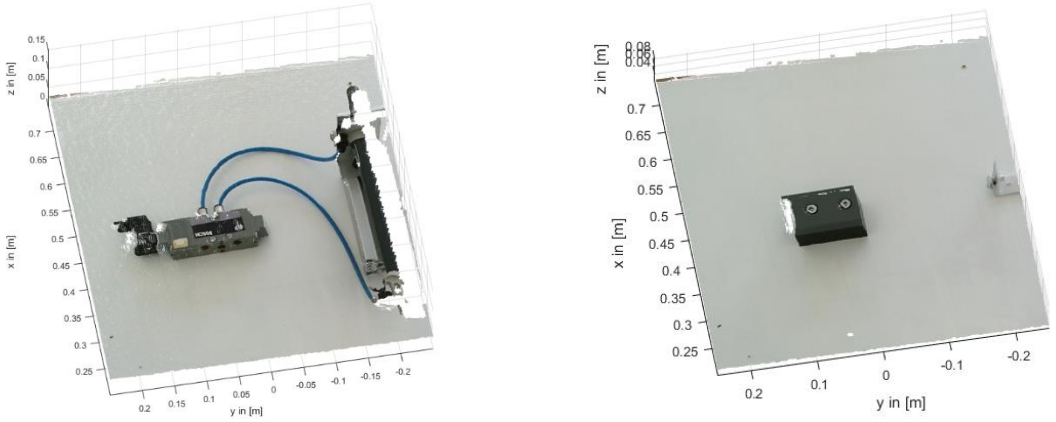
Ausgabe: Optimale Pose $\underline{p}^* \in \mathcal{W}_{\text{sensor}}$.

$bestBenefit \leftarrow \text{evalFunction}\left(\underline{p}(1{:}\,dim\left(\underline{p}\right), \mathcal{PVM}\right);$
while $(d\underline{p}^T \cdot d\underline{p} > \varepsilon)$
  for $i = 1{:}\,1{:}\,dim\left(\underline{p}\right)$
    $\underline{p}(i) = \underline{p}(i) + d\underline{p}(i);$
    $actBenefit \leftarrow \text{evalFunction}\left(\underline{p}(1{:}\,dim\left(\underline{p}\right), \mathcal{PVM}\right);$
    if $(actBenefit > bestBenefit)$ **then**
      $bestBenefit = actBenefit;$
      $d\underline{p}(i) = d\underline{p}(i) \cdot (1 + \gamma);$
    else
      $\underline{p}(i) = \underline{p}(i) - 2d\underline{p}(i);$
      $actBenefit \leftarrow \text{evalFunction}\left(\underline{p}(1{:}\,dim\left(\underline{p}\right), \mathcal{PVM}\right);$
      if $(actBenefit > bestBenefit)$ **then**
        $bestBenefit = actBenefit;$
        $d\underline{p}(i) = d\underline{p}(i) \cdot (1 + \gamma);$
      else
        $\underline{p}(i) = \underline{p}(i) + d\underline{p}(i);$
        $d\underline{p}(i) = d\underline{p}(i) \cdot (1 - \gamma);$
      end if
    end if
  end for
end while
return $(\underline{p}^*)$

Die Festlegung des Schrittweitenvektors kann experimentell erfolgen und wird im Abschnitt zur experimentellen Validierung betrachtet. Nachteil des Twiddle-Algorithmus ist, dass er zur Familie der Hill-Climbing-Algorithmen gehört, wodurch die Gefahr besteht, dass er in einem lokalen Maximum konvergieren kann. Es können jedoch ohne weiteres, andere Optimierungsverfahren zur Lösung integriert und eingesetzt werden. Damit das Problem der Konvergenz in einem lokalen Optimum verringert wird, werden die Startposen initial zufällig gewählt und dieser Vorgang wiederholt. Die Optimierung der Pose erfolgt über die translatorischen Größen, die Orientierung ist immer orthogonal zu einer Sphäre ausgerichtet, welche um den Baugruppenmittelpunkt aufgespannt wird, vergleiche hierzu Abbildung 4-19 (a).

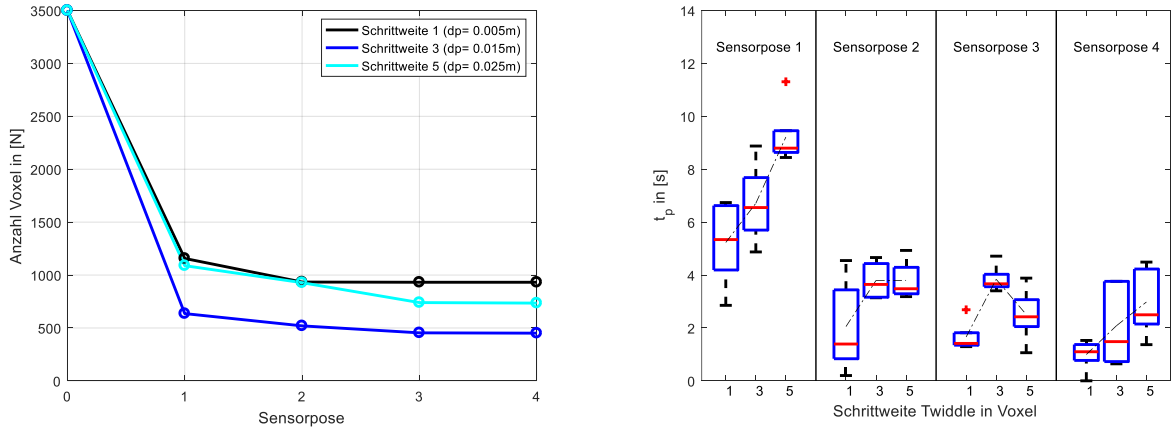
Experimentelle Validierung. Weitergehend soll der entwickelte Algorithmus des aufgabenorientierten NBV, anhand von anwendungsnahen Beispielen, untersucht werden. Hierzu werden zwei Szenen betrachtet, vergleiche Abbildung 4-20. Anhand Szene 1 (Abbildung 4-20 (a)) soll der Einfluss des Schrittweitenvektors des Twiddle-Algorithmus auf die Exploration analysiert werden. Szene 2 (Abbildung 4-20 (b)) dient der Untersuchung der Funktionsweise der Aufgabenexploration (Einzelaufgaben und kombinierter Ansatz). Das zugehörige experimentelle Setup ist in 6.1 beschrieben.

Evaluierung Schrittweitenvektor. Die Bestimmung einer geeigneten Schrittweite $\underline{dp} = (dx \quad dy \quad dz)^T$ für den Twiddle-Algorithmus erfolgt experimentell anhand Szene 1. Für größenmäßig stark variierende Baugruppen ist wiederum eine Anpassung erforderlich, da sonst keine gute Lösung erzielt werden kann. Die Auswertung erfolgt über die Lösung von Gleichung (4.33). Das Experiment wird mit drei unterschiedlichen Schrittweiten überprüft. Jede Optimierung pro Sensorpose wird fünfmal mit einer initial, zufällig gewählten Sensorpose wiederholt. Die beste Lösung wird als neue Sensorpose verwendet. Abbildung 4-21 (a) zeigt die Ergebnisse auf.



(a) (b)

Abbildung 4-20: Anwendungsbeispiele zur Validierung des aufgabenorientierten NBV: Szene 1 mit Ventilbaugruppe (a); Szene 2 mit Plattenbaugruppe (b)

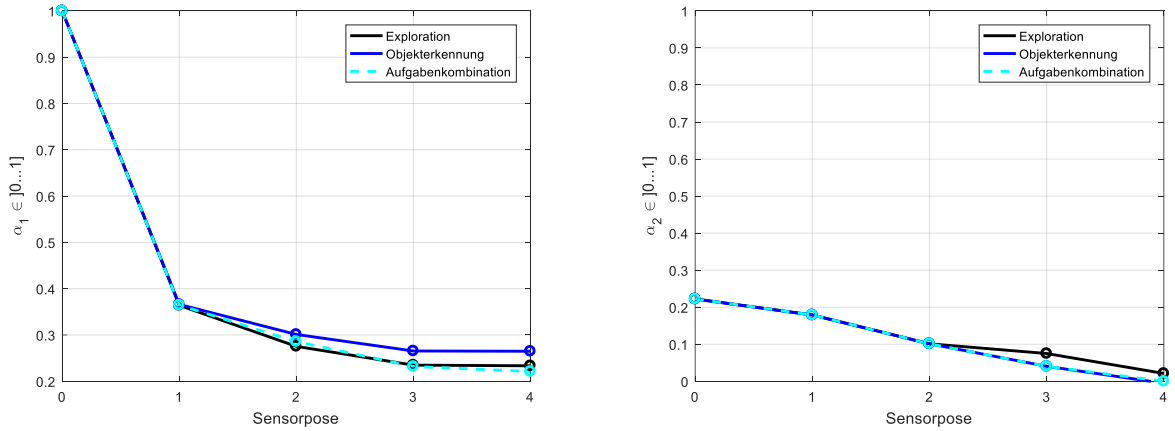


(a) (b)

Abbildung 4-21: Schrittweitenparametrierung des Twiddle-Algorithmus anhand Szene 1 (a); Durchschnittliche Zeit für jeweils 5 Optimierungen mit zufälligen Startwerten pro Sensorpose (b)

Es stellt sich heraus, dass die beste Lösung mit der Schrittweite 3 ($\underline{dp} = (0.015 \quad 0.015 \quad 0.015)^T$, in [m]) gefunden werden kann, da hier die Informationsgewinnung, gemessen anhand der Raumexploration, am Höchsten ist. Eine zu klein gewählte Schrittweite (Schrittweite 1) führt zwar zu einer schnellen Konvergenz, jedoch zu einer äußerst sub-optimalen Lösung, was sich an der geringen Informationsgewinnung zeigt. Dies liegt daran, dass die Lösung schnell in ein lokales Extremum fällt und dort aufgrund der geringen Schrittweite verharrt, da keine Verbesserung mehr auftritt, was zur Terminierung führt. Die hohe Konvergenzgeschwindigkeit macht sich auch anhand der niedrigen Planungszeit bemerkbar, vergleiche Abbildung 4-21 (b). Eine zu groß gewählte Schrittweite (Schrittweite 5) führt zu einem Springen zwischen geeigneten Lösungsposen, wodurch die Optimierung am meisten Zeit benötigt. Auch die gefundenen Lösungen sind schlechter als bei der Verwendung von Schrittweite 3. Das Verhalten hat sich bei weiteren, ähnlichen Experimenten bestätigt.

Evaluierung Einzelaufgaben und Aufgabenkombination. Folgend wird, anhand Szene 2, sowohl die Ausführung von Einzelaufgaben als auch die Aufgabenkombination untersucht. Als zu erkennende Objekte werden die beiden Schrauben der Baugruppe betrachtet. Dabei wird eine Schraube durch ein weiteres Objekt verdeckt, sodass diese in der initialen Sensorpose nicht erkannt werden kann. Es wird das Verhalten der beiden Metriken α_1 (Raumexploration) und α_2 (Objekterkennung) untersucht. Es wird zum einen das Verhalten bei Durchführung der Einzelaufgaben, also nur die Optimierung der Raumexploration (4.33) oder nur die Optimierung der Objekterkennung (4.36), zum anderen die dynamisch verknüpfte Optimierung mittels Aufgabenkombination (4.39) betrachtet. Das initiale Vertrauen in die passive Information, hier das Vorhandensein beider Schrauben, wird mit $\text{bel}(\mathcal{O}_i^-) = 0.7$, mit $i = 1{,}2$ angenommen (theoretisch müsste ein initiales Vertrauen von $\text{bel}(\mathcal{O}_i^-) = 0.5$ angenommen werden. Da aber eigentlich ein Faustwert von $\text{bel}(\mathcal{O}_i^-) = 0.9$ gelten sollte, wird einfach der Mittelwert aus beiden Annahmen verwendet. Dies ist jedoch für die Funktionsweise des Algorithmus unerheblich. Eine sinnvolle Annahme in das Vertrauen einer Klassifikation $P(\mathcal{O}_i|z_t)$ kann durch deren true-positive Rate erfolgen). Abbildung 4-22 zeigt die Ergebnisse für die Einzel- sowie die Aufgabenkombination.



(a) (b)

Abbildung 4-22: Verhalten der Metriken (4.37), (4.38) als Lösung für die Optimierungsprobleme unter Beachtung der Einzelaufgaben (Exploration (4.33), Objekterkennung (4.36)) und Aufgabenkombination (4.39): Explorationsmetrik α_1 (a); Objekterkennungsmetrik α_2 (b)

Erwartungsgemäß ergibt sich ein besseres Verhalten der Metrik, deren zugehörige Aufgabe optimiert wird. Die marginal bessere Lösung für die Raumexplorationsmetrik bei Verwendung der Aufgabenkombination gegenüber der rein explorativ geplanten Lösung, kann auf die lokalen Eigenschaften des Twiddle-Algorithmus zurückgeführt werden. Durch die Kombination beider Aufgaben, erkennt der Roboter die Objekte zuverlässig, während bei der reinen Raumexploration die Objekte zweimal weniger erkannt werden. In Abbildung 4-23 ist die Ausführung bei der Aufgabenkombination dargestellt. Durch die Kombination der Aufgaben erkennt der Roboter bereits für die erste geplante Sensorpose beide a priori vermuteten Objekte. Es zeigt sich, dass das Verfahren als sehr geeignet für die hier gegebene Problemstellung bewertet werden kann. So wird parallel eine metrische Voxelkarte, als auch die nötige Information zum Abgleich der passiven, durch die aktive Information, erzeugt. Somit steht ein vollständiges Umweltmodell zur Verfügung, welches zur Sequenzierung der Aufgaben benötigt wird. Ein Video hierzu findet sich unter [20].

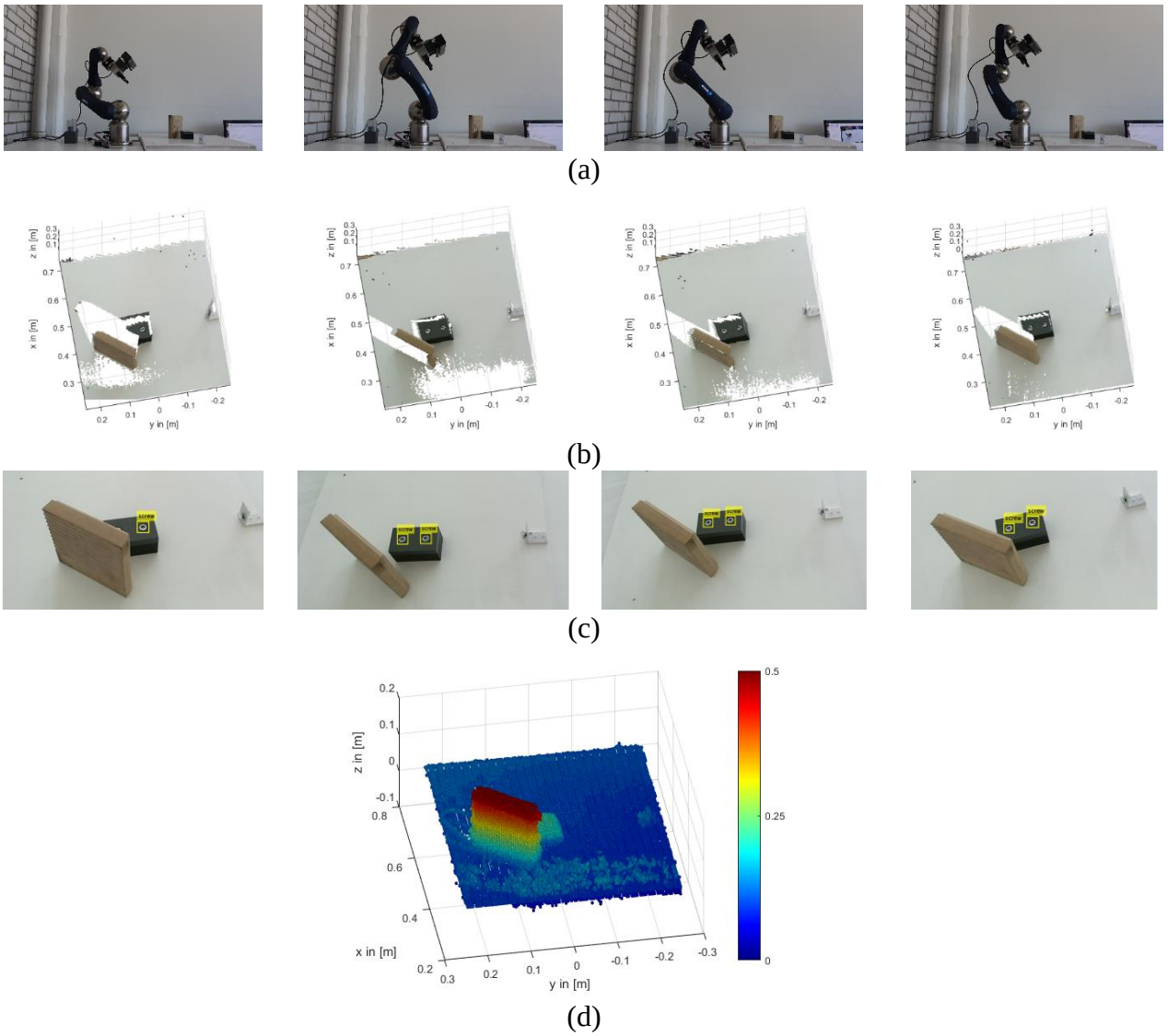


Abbildung 4-23: Ergebnisse der globalen Aufgabenkombination anhand Szene 1: Geplante Roboterposen auf Basis von Gleichung (4.39) (a); zugehörige Tiefeninformation (b); zugehörige RGB-Objekterkennungsergebnisse (c); Erzeugte Voxelkarte aus den vier Sensorposen (d)

4.4.3 Sequenzierung von Robotermanipulationen

Die in 4.2 vorgeschlagene Planungsmethode bestimmt die zur Lösung einer Aufgabe notwendige Menge an Manipulationsprimitiven $\{\mathcal{MP}\}$. Dabei besteht das Problem, dass diese Menge aufgrund der Betrachtung des Planungsproblems im relativen Demontageraum, nicht zwingend ausführbar ist. Dieses Unterkapitel entwickelt eine Methode, sodass die ungeordnete Menge $\{\mathcal{MP}\}$ in eine ausführbare und lokal optimale Aktionssequenz $\langle\mathcal{MP}\rangle$ überführt werden kann. Das Konzept ist bereits in den Vorarbeiten [18], [21] vorgestellt worden. Zu diesem Zweck sollen eingangs die Definitionen der Sichtbarkeit und der Zugänglichkeit eingeführt werden.

Definition 7: Sichtbarkeit. *Der sichtbare Raum $\mathcal{W}_V$ beschreibt den kartesischen Raum aller Bauteile C_i, welche zum aktuellen Demontagezustand sichtbar sind.*

Definition 8: Zugänglichkeit. *Der zugängliche Raum $\mathcal{W}_A$ beschreibt den kartesischen Raum aller Bauteile C_i, welche zum aktuellen Demontagezustand kollisionsfrei, unter Beachtung der Robotergeometrie, zugänglich beziehungsweise manipulierbar sind.*

Im Allgemeinen gilt, dass die den sichtbaren Bauteilen $C_i \in \mathcal{W}_V$ zugehörigen $\{\mathcal{MP}_V\}$ eine Untermenge aus $\{\mathcal{MP}\}$ bilden. Die Sichtbarkeit ist durch die Zustände von Gleichung (4.27) gegeben. Durch eine Kollisionsprüfung, für die Manipulation des Roboters mit allen sichtbaren Bauteilen C_i, lassen sich folglich alle kollisionsfrei, zugängliche Bauteile C_A und deren Primitive $\{\mathcal{MP}_A\}$ ermitteln. Dies führt auf folgenden Zusammenhang

$$\{\mathcal{MP}_A\} \subseteq \{\mathcal{MP}_V\} \subseteq \{\mathcal{MP}\}. \tag{4.40}$$

Die Menge $\{\mathcal{MP}_A\}$ zu den Bauteilen C_A entspricht die dem aktuellen Demontagezustand zugehörige Menge an ausführbaren Aktionen, welche sequenziert werden kann. Dabei wird auch die Abhängigkeit von Manipulationsprimitiven beachtet. Nach Ausführung aller Aktionen entsteht ein neuer sichtbarer Raum und das Verfahren kann wiederholt werden, bis die Erfüllung der Gesamtaufgabe gegeben ist, vergleiche hierzu Abbildung 4-24. Die Sequenzierung der einzelnen Manipulationsprimitiven kann als klassisches Optimierungsproblem mit (4.41) formuliert werden.

$$t^* = \underset{t_{path}, t_\tau \in D}{\arg\min} \langle\mathcal{MP}_{acc}\rangle \left(t_{path}, t_\tau\right), \tag{4.41}$$

Dabei stellen t_{path} die Wegkosten und t_τ die Werkzeugwechselkosten auf zeitlicher Basis dar. Die Bestimmung einer optimalen Zeit t^* ist ein symmetrisches Travelling Salesman Problem (TSP), wodurch bekanntermaßen $\left((n-1)!\right)/2$ Möglichkeiten für n-Manipulationsprimitive existieren. Prinzipiell ist TSP ausreichend untersucht und es existieren verschiedene Heuristiken und Näherungslösungen. Aufgrund der hier vorliegenden geringen Probleminstanz soll, zur Findung einer optimalen Lösung, der A*-Algorithmus mit verschiedenen Heuristiken genutzt werden. Hierzu muss die Terminierungsbedingung so umgeformt werden, dass nicht ein Zielzustand erreicht wird, sondern alle Zustände besucht werden. Es werden folgende Heuristiken verwendet:

- Zero Heuristik (Dijkstra),
- Nearest-Neighbor Heuristik (A*-NN),
- Minimum Spanning Tree Heuristik (A*-MST).

Eine kurze Beschreibung ist in 4.5.1 zu finden, da A* auch für die Bahnplanung verwendet wird.

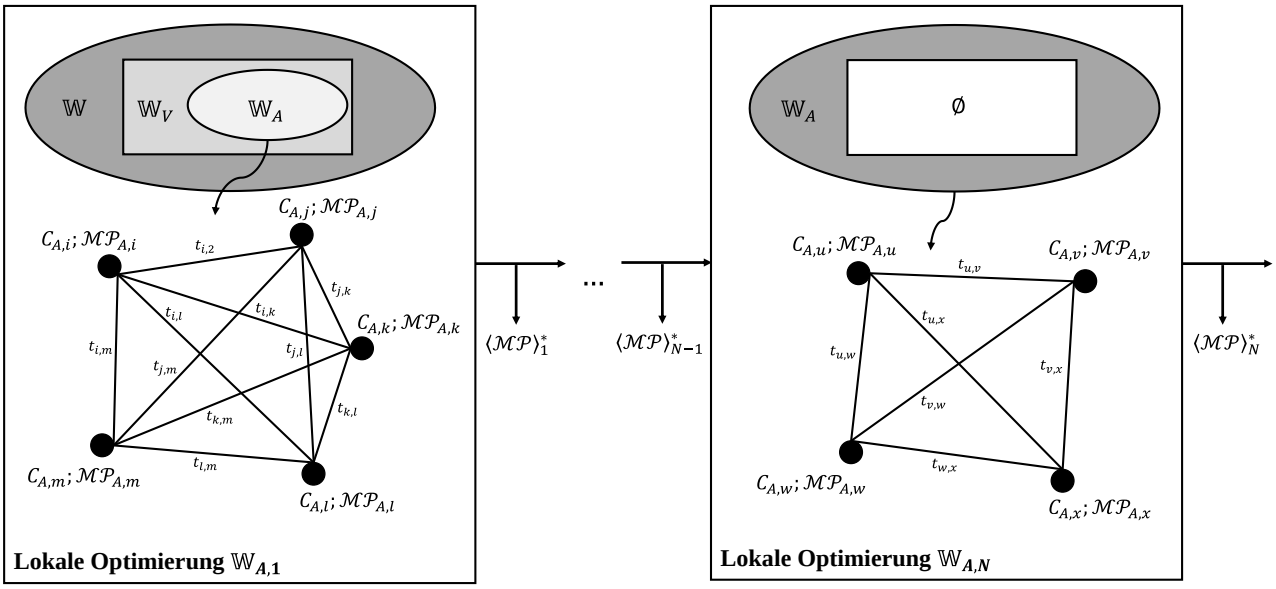
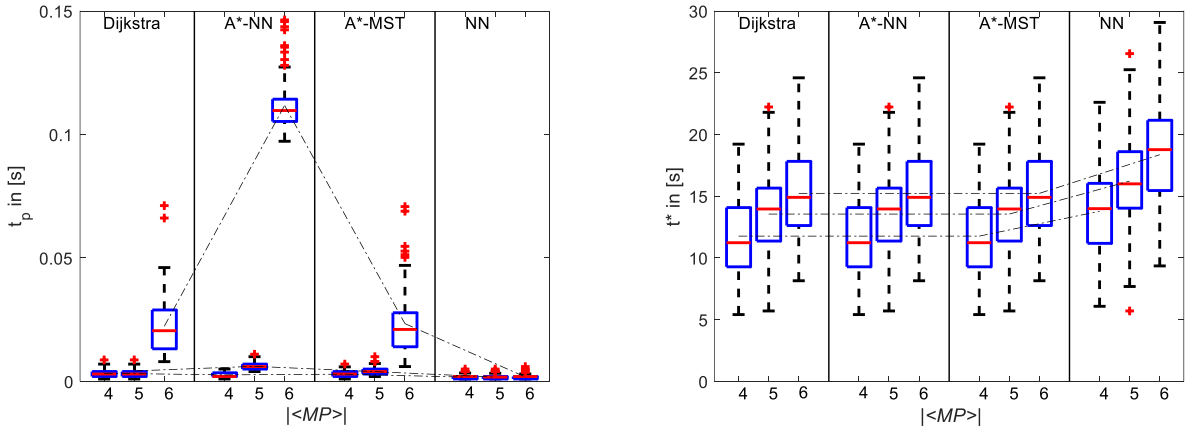


Abbildung 4-24: Darstellung der Sichtbarkeit und Zugänglichkeit mit lokaler Optimierung

Es ist leicht einsehbar, dass diese Lösung für größere und schlecht konditionierte Probleminstanzen nicht mehr praktikabel ist. Neben der Verwendung des A* mit unterschiedlichen Heuristiken wird der klassische Nearest-Neighbor Algorithmus (NN) verwendet, welcher zwar nicht zwingend eine optimale Lösung liefert, jedoch in $\mathcal{O}(n^2)$ mit $n = |\{\mathcal{MP}\}|$ terminiert.

Simulative Validierung. Die Sequenzierung wird zur simulativen Validierung für jeweils vier, fünf und sechs Manipulationsprimitive überprüft. Diese Anzahl entspricht auch den praktisch gewählten Beispielen in dieser Arbeit. Dabei wird jedes Experiment 100-mal, mit anfangs zufällig initialisierten Kostenmatrizen, durchgeführt, um unterschiedlich schwere Probleme zu simulieren. Abbildung 4-25 zeigt die Ergebnisse für die Planungszeit t_p (a) und die optimierte Ausführungszeit t^* (b) für gegebenes Setup auf. Es zeigen sich die erwarteten Effekte. Die durchschnittliche Verbesserung der sequenzierten Manipulationsprimitive, bei Verwendung einer optimalen Lösung, führt zu einer Zeiteinsparung von circa 21% gegenüber dem NN-Algorithmus.



(a) (b)

Abbildung 4-25: Ergebnisse der Sequenzierung der Manipulationsprimitive für 100 zufällig gewählte Probleminstanzen: Planungszeit (a); Ausführungszeit der Sequenzierung (b)

Dies wird natürlich durch den Nachteil der schlechten Skalierung, bezüglich der Planungszeit, erkauft. Im arithmetischen Schnitt ist die Planungszeit bei der Sequenzierung von sechs Manipulationsprimitiven, gegenüber dem NN-Algorithmus, um 12-mal höher für Dijkstra, 61-mal höher für A*-NN und 12-mal höher für A*-MST.

Bereits für die Sequenzierung von acht Primitiven ist der Vorteil der optimalen Lösung durch A*, aufgrund der hohen Planungszeit, gegenüber einer sub-optimalen Lösung durch den NN-Algorithmus, praktisch, also in Bezug auf die optimierte Ausführungszeit, nicht mehr gegeben. Somit lohnt es sich aus praktischer Sicht oftmals eine sub-optimale Lösung zu akzeptieren und dafür mit einer geringeren Planungszeit zu rechnen.

Es sei jedoch angemerkt, dass eine genaue zahlenmäßige Abschätzung nicht möglich ist, da hierzu mehr auf die Komplexität der einzelnen simulierten Probleme sowie auf die Implementierung der verwendeten Verfahren eingegangen werden müsste.

4.5 Bahnplanung für Robotermanipulationen

Neben der Bereitstellung der Sequenz an Manipulationsprimitiven $\langle \mathcal{MP} \rangle$, müssen noch Bahninformationen für die Verknüpfung und Ausführung der einzelnen Primitiven bestimmt werden. Zur Bestimmung einer kollisionsfreien Bahn $\underline{p} = \underline{f}(x, y, z, \alpha_x, \beta_y, \gamma_z)$ für einen stationären Robotermanipulator mit n-Drehgelenken gilt für den Konfigurationsraum $\mathcal{C}$ folgende Eigenschaft

$$\mathcal{C} \subseteq (\mathbb{R}^3 \times SO(3))^n, \tag{4.42}$$

wodurch eine praktikable Berechnung nicht gegeben ist [114]. Zur Lösung sind deshalb, wie bereits in 2.3.5 dargelegt, globale such- und stichprobenbasierte *(engl. „sampling-based“)* Bahnplanungsalgorithmen bekannt, welche eine implizite Berechnung des Konfigurationsraums erlauben. Jedoch besteht bei all diesen Bahnplanungsalgorithmen das Problem, dass die Planungszeit zur Findung einer kollisionsfreien Bahn direkt mit der Struktur der Umgebungskarte, welche zur Bahnplanung verwendet wird, skaliert. Es stellt sich somit immer die Frage, wie sich eine geeignete Schrittweite $|s|$ für den Planer finden lässt. Abbildung 4-26 zeigt schematisch die Problemstellung einer Bahnfindung zwischen Start- $\underline{r_s}$ und Zielpose $\underline{r_t}$ auf. Dieses Unterkapitel entwickelt eine Alternative, auf Basis einer adaptiven Schrittweitensteuerung, sodass die Exploration weitgehend an das Bahnplanungsproblem angepasst werden kann. Des Weiteren ergibt sich durch die direkte Kopplung der Manipulationsprimitive mit der Bahnplanung der Vorteil, dass die Kollisionsanalyse direkt für das auszuführende Primitiv bestimmt wird und so auch Interaktionen mit den Roboterwerkzeugen betrachtet werden können.

Eingehend sollen die aus dem Stand der Technik, vergleiche 2.3.5, bekannten, globalen Bahnplanungsalgorithmen näher eingeführt werden, die in dieser Arbeit zur Evaluierung der Schrittweitensteuerung verwendet werden, sodass die universelle Übertragbarkeit besser dargestellt werden kann. Darauf aufbauend wird die neue Methode zur Schrittweitensteuerung vorgestellt und von theoretischer Seite diskutiert. Die gesamten Algorithmen sind in ein Bahnplanungsframework integriert, welches roboterunabhängig Verwendung finden kann. Das Unterkapitel schließt mit der ausführlichen experimentellen Evaluierung der Methode. Teile dieses Verfahrens wurden bereits in den Vorarbeiten [18], [21] veröffentlicht.

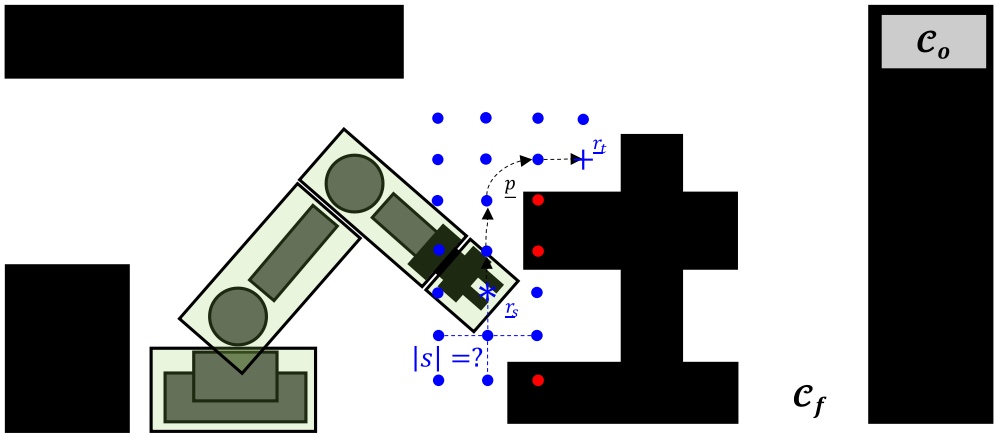


Abbildung 4-26: Schematische Darstellung der globalen Bahnplanung: Der Hindernisraum wird mit $\mathcal{C}_o$, der freie Raum mit $\mathcal{C}_f$ beschrieben; Die blauen Punkte stellen eine kollisionsfreie und die roten eine Exploration mit Kollision dar; Die Robotergeometrie wird mit Hüllkörpern approximiert

4.5.1 Verwendete Kollisionserkennung und Bahnplanungsalgorithmen

Wie bereits einführend erläutert, werden bekannte Kollisionserkennungs-, als auch globale Bahnplanungsalgorithmen, gemäß dem Stand der Technik, verwendet. Zunächst sollen die theoretisch bekannten Kollisionserkennungs- und Bahnplanungsalgorithmen, die im Rahmen dieser Arbeit genutzt und implementiert wurden, der Übersichtlichkeit und Vollständigkeit eingeführt werden. Für die Bahnplanung werden sowohl suchbasierte (Greedy, A*) als auch stichprobenbasierte Methoden (RRT, bi-RRT) eingesetzt.

Kollisionserkennung. Für die Kollisionserkennung findet ein zweistufiges Prüfungsverfahren Verwendung. In der ersten Stufe wird jedes Robotergelenk, durch eine umschließende Kugel, approximiert. Somit kann durch eine einfache Prüfung von Radien eine mögliche Kollision erkannt werden, die besonders zeiteffizient ist. Wird eine Kollision für ein Gelenk erkannt, wird dieses in der zweiten Stufe durch einen orientierten Quader *(engl. „Oriented Bounding Box“)* angenähert. Hierdurch wird eine gute Approximation der Geometrie der Roboterkinematik, bei gleichzeitiger Zeiteffizienz der Kollisionsprüfungsalgorithmen, erreicht.

Suchbasierte Planer. Suchbasierte Algorithmen haben für Bahnplanungsprobleme prinzipiell den Vorteil, dass diese einen konsistenten Pfad finden. Es wird also für ein gegebenes Problem immer derselbe Pfad bestimmt. Jedoch eignen sich suchbasierte Algorithmen nur begrenzt für das Bahnplanungsproblem, aufgrund des hochdimensionalen Planungsraums [345]. Trotzdem soll die Eignung suchbasierter Algorithmen, in Kombination mit der neuartigen adaptiven Schrittweitensteuerung bei klassischen Benchmarkproblemen untersucht werden. Folgend werden die wichtigsten Eigenschaften zweier Vertreter dieser Klasse aufgeführt.

A*.** Der A*-Algorithmus ist erstmalig in der Arbeit [346] beschrieben. Es handelt sich bei diesem Verfahren um einen suchbasierten und informierten Algorithmus, welcher vollständig (der Algorithmus terminiert in endlicher Zeit) und optimal (in Bezug auf die Kosten der Evaluierungsfunktion) ist, vergleiche beispielsweise auch [144]. Als Evaluierungsfunktion wird dabei Gleichung (4.43) verwendet. Die Variation von γ wird als weighted-A* bezeichnet.

$$f(n) = g(n) + \gamma \cdot h(n), mit\ \gamma \in [0, \dots, 1]. \tag{4.43}$$

$g(n)$ beschreibt die aktuellen Kosten vom Startzustand n_s zum Zustand n und $h(n)$ die von der Heuristik geschätzten Kosten vom Zustand n zum Zielzustand n_t. Für den klassischen A* ist $\gamma = 1$, wodurch die Heuristik mit der gleichen Gewichtung in das Optimierungsproblem miteingeht. Durch die Variation von γ wird der Algorithmus auch als gewichteter A* bezeichnet, wodurch die Exploration gezielt gesteuert werden kann, jedoch Eigenschaften hinsichtlich Optimalität verletzt werden. Dies bedeutet, dass für $\gamma < 1$, der Suchkorridor breiter wird, entgegengesetzt, also für $\gamma > 1$, wird der Suchkorridor schmaler, da Zustände, die näher am Ziel liegen, bevorzugt exploriert werden. Für den Sonderfall $\gamma = 0$ geht der A* in den Dijkstra-Algorithmus über, welcher zu den uninformierten Suchalgorithmen gehört, da er keine Abschätzung der Restkosten verwendet. Als Heuristik werden für die Bahnplanung gewöhnlich die Kosten auf Basis des euklidischen Abstandsmaß d_E, also zwischen Position $\underline{x}$ von Zustand n und Position $\underline{y}$ des Zielzustands mit

$$d_E\left(\underline{x}, \underline{y}\right) := \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2}\,, \tag{4.44}$$

definiert. Die Optimalität für den klassischen A* ist dabei, unter der Bedingung, dass $h(n)$ eine zulässig und monotone Heuristik ist, immer gegeben.

Greedy Best-First-Search. Der BFS-Algorithmus kann ebenfalls als Sonderfall des A*, für die Evaluierungsfunktion (4.45), angegeben werden.

$$f(n) = h(n). \tag{4.45}$$

Dies bedeutet, dass der Algorithmus von seinem aktuellen Zustand n_i immer den Nachfolgezustand n_{i+1} wählt, welcher aus Sicht von n_i optimal ist um den Zielzustand n_t zu erreichen. Dieses Verhalten gehört zur Klasse der Greedy-Algorithmen, welches nicht zwingend optimal ist, da nicht der gesamte Zustandsraum betrachtet wird. Auch sind Greedy-basierte Planer im Allgemeinen nicht vollständig. Vorteilhaft ist jedoch, dass sich aufgrund der rein heuristikgesteuerten Optimierung, eine geringere Anzahl an zu explorierenden Zuständen ergibt und somit häufig relativ schnell eine Lösung gefunden werden kann.

Stichprobenbasierte Planer. Gegenüber suchbasierten Planern eignen sich stichprobenbasierte Planer häufig besser für das hochdimensionale Bahnplanungsproblem. Aufgrund der stochastischen Natur der Verfahren ist jedoch keine Konsistenz der Bahn gegeben. Weitergehend sollen zwei grundlegende Vertreter aus der Klasse der stichprobenbasierten Planer eingeführt werden.

Rapidly-Exploring Random Tree. Der Rapidly-Exploring Random Tree (RRT) wurde erstmals in der Arbeit [272] vorgestellt. Dabei wird von einem Startzustand inkrementell eine Baumstruktur aufgebaut, bis der Zielzustand erreicht ist. Das Hinzufügen eines neuen Knotens erfolgt dabei über einen zufällig gewählten Punkt im Raum, welcher kollisionsfrei ist und nur eine definierte Schrittweite vom Mutterknoten entfernt ist. Der RRT ist probabilistisch vollständig, allerdings nicht asymptotisch optimal.

Bidirektionaler Rapidly-Random Tree. Der bidirektionale RRT (bi-RRT) stellt eine einfache Erweiterung des klassischen RRT dar. Dabei wird sowohl ein Baum vom Start- als auch vom Zielzustand aufgebaut. Wenn die Bäume zusammenwachsen kann ein Pfad vom Start- zum

Zielzustand bestimmt werden. Hierdurch kann die Planungszeit reduziert werden. Eine Erweiterung hiervon stellt der RRT-connect [274] dar.

Prinzipiell existieren, sowohl bei such- als auch bei stichprobenbasierten Planern [347], Erweiterungen und Heuristiken, welche bestimmte Eigenschaften bezüglich Effizienz, Speicherbedarf und Optimalität verbessern. Jedoch soll, zum Nachweis der Leistungsfähigkeit der hier entwickelten Methode und der besseren Nachvollziehbarkeit, auf die vorgestellten Basisplaner zurückgegriffen werden.

4.5.2 Schrittweitensteuerung für globale Bahnplanungsalgorithmen

Sei $\mathcal{A}^*$ ein Bahnplanungsalgorithmus, welcher für ein distanzkonstantes Explorationsgrid immer die optimale Bahn (optimal in Bezug auf den kürzesten Weg) liefert, dann skaliert die Planungszeit direkt in Abhängigkeit der verwendeten Explorationsschrittweite s von $\mathcal{A}^*$. Prinzipiell kann beobachtet werden, dass zwei Grenzfälle nach Gleichung (4.46)-(4.47) existieren.

$$\mathcal{A}^*(s_\downarrow) \rightarrow \underline{p}_\downarrow, t_{plan,\uparrow}, \tag{4.46}$$

$$\mathcal{A}^*(s_\uparrow) \rightarrow \underline{p}_\uparrow, t_{plan,\downarrow}. \tag{4.47}$$

Dies bedeutet, dass für eine kleine Explorationsschrittweite $s_\downarrow$ eine hohe Planungszeit $t_{plan,\uparrow}$, aber ein kürzerer Pfad entsteht. Wird die Schrittweite erhöht, also $s_\uparrow$ verwendet, resultiert dies in einem umgekehrten Ergebnis. Hieraus stellt sich die Frage, ob eine geeignete adaptive Schrittweite s_Δ existiert, für welche es möglich ist, einen (nahezu) optimalen Pfad $\underline{p}_\downarrow \approx \underline{p}^*$ in einer geringen Planungszeit $t_{plan,\downarrow}$ zu bestimmen. Unter einer geringen Planungszeit wird dabei verstanden, dass die Eigenschaft

$$\mathcal{A}^*(s_\Delta)\colon t_{plan,\downarrow} \cong \mathcal{A}^*(s_\uparrow)\colon t_{plan,\downarrow} \tag{4.48}$$

erfüllt ist. Zur Beantwortung dieser Frage wird weitergehend ein Verfahren vorgestellt, welches auf Basis der Information, über die Belegtheit der Umgebungskarte, eine Bestimmung einer geeigneten Schrittweite erlaubt. Sei die Umgebungskarte wieder gegeben durch die Definition der probabilistischen Voxelkarte $\mathcal{PVM}$, vergleiche 4.3.2. Aufgrund der Kartenkodierung in belegte v_o, in freie v_f und unbekannte Voxel v_u kann die globale Belegtheitsdichte mit

$$\rho_{global} = \frac{|\mathcal{V}_o|}{|\mathcal{V}_o| + |\mathcal{V}_f| + |\mathcal{V}_u|}, mit\ \rho_{global} \in [0 \dots 1]. \tag{4.49}$$

angegeben werden. Prinzipiell eignet sich nun für die Festlegung der Explorationsschrittweite eine kleine Schrittweite $s_{\Delta,\downarrow}$ für Umgebungskarten mit hoher Belegtheitsdichte $\rho_\uparrow$ und für niedrige Belegtheitsdichten $\rho_\downarrow$ eine große Schrittweite $s_{\Delta,\uparrow}$. Um nicht nur die globale Belegtheitseigenschaft zu betrachten, wird $\mathcal{PVM}$ iterativ in Unterräume $\mathcal{R}_i$ aufgeteilt. Eine Aufteilung erfolgt dabei in acht neue Unterräume, welche als Blätter in einem Octree $\mathcal{T}$ interpretiert werden können. Jedes Blatt wird dabei wieder geometrisch durch einen Quader $C_i = I_x \times I_y \times I_z = [x_{min}, x_{max}] \times [y_{min}, y_{max}] \times [z_{min}, z_{max}]$ repräsentiert, welcher in Abhängigkeit seiner lokalen Belegtheitsdichte ρ_i wiederum in neue Blätter $c_{i1}, c_{i2} \dots, c_{i8}$ unterteilt wird, wenn C_i die Forderung $\rho_i(\mathcal{R}_i) < \rho_{limit}$ nicht erfüllt.

Die Festlegung der Schrittweite $s_{\Delta,i}$ lässt sich dann am geeignetsten als Funktion der Baumtiefe $d_{\mathcal{T},i}$ des Unterraums $\mathcal{R}_i$ nach (4.50) angeben. Der Pseudocode in Algorithmus 4.4 beschreibt den Algorithmus zur Schrittweitensteuerung.

$$s_{\Delta,i} = g\left(d_{\mathcal{T},i}\right) = g\left(f(\rho_i)\right). \tag{4.50}$$

In Abbildung 4-27 ist die Methode, anhand eines zweidimensionalen Beispiels, dargestellt. Obige linke Abbildung (a) stellt die Ursprungskarte vor der Unterteilung dar. Weitergehend wird die Karte dreimal unterteilt, mit der ersten Unterteilung (b), der zweiten (c) und der dritten (d), welche in diesem Fall die unterste Voxelebene abbildet. Die Struktur, beziehungsweise die einzelnen Unterräume der Karte lassen sich dann als Baum beschreiben (e). Somit ergibt sich für die einzelnen Unterräume folgende Aufteilung mit $\mathcal{R}_i \in \left\{\{\mathcal{R}_{i,1}, \dots \mathcal{R}_{i,8}\}, \{\mathcal{R}_{i,1,1}, \dots \mathcal{R}_{i,1,8}\}, \dots, \{\mathcal{R}_{i,1,\dots,N,1}, \dots, \mathcal{R}_{i,1,\dots,N,8}\}\right\}, \dots, \mathcal{R}_N \in \left\{\{\mathcal{R}_{N,1}, \dots \mathcal{R}_{N,8}\}, \{\mathcal{R}_{N,1,1}, \dots, \mathcal{R}_{N,1,8}\}, \dots, \{\mathcal{R}_{N,1,\dots,M,1} \dots, \mathcal{R}_{N,1,\dots,M,8}\}\right\}$ sowie zugehöriger Explorationsschrittweitenvektor mit

$$\underline{s_\Delta} = \left[s_{\Delta,i,1}, \dots s_{\Delta,i,8}, s_{\Delta,i,1,1}, \dots s_{\Delta,i,1,8}, \dots, s_{\Delta,N,1,\dots,M,8}\right]. \tag{4.51}$$

Algorithmus 4-4: Adaptive Schrittweitensteuerung mit Unterraummethode

Eingabe: Probabilistische Voxelkarte $\mathcal{PVM}$.

Ausgabe: Probabilistische Voxelkarte mit Unterräumen $\mathcal{PVM}_{ASD}$;
Korrespondierende Schrittweite $\underline{s_\Delta}$ für Unterraum $\mathcal{R}_i$.

// Berechne globale Belegtheit
$\rho \leftarrow \frac{\sum_j o_j \in \mathcal{PVM}}{\sum_i v_i \in \mathcal{PVM}}$;
$\underline{s_\Delta} \leftarrow \mathrm{g}(\mathrm{d}_\mathcal{T})$;
$k \leftarrow 1$;
// Start Unterraumbildung
for $i = 1{:}1{:}8$
  while $(\rho > \rho_{limit})$ **do**
    if $(k > 1)$ **then**
      $\rho \leftarrow \frac{\sum_j o_j \in \mathcal{R}_i}{\sum_i v_i \in \mathcal{R}_i}$;
      $s \leftarrow \mathrm{g}(\mathrm{d}_\mathcal{T})$;
    end if
    $\{\mathcal{R}_{i,1}, \mathcal{R}_{i,2}, \dots, \mathcal{R}_{i,8}\} \leftarrow \mathrm{divideSpace}(\mathcal{R}_i)$;
    $\mathcal{PVM}_{ASD} \leftarrow \{\mathcal{R}_{i,1}, \mathcal{R}_{i,2}, \dots, \mathcal{R}_{i,8}\}$;
    $\underline{s_\Delta} \leftarrow s$;
    $k \leftarrow k + 1$;
  end while
end for
return $(\mathcal{PVM}_{ASD}, s_\Delta)$;

function $(\mathcal{R}_{i,1}, \mathcal{R}_{i,2} \dots, \mathcal{R}_{i,8}) \leftarrow \mathrm{divideSpace}(\mathcal{R}_i)$
  $\mathcal{R}_{i,1} \leftarrow [x_{min} + 0.5|I_x|, x_{max}] \times [y_{min}, y_{min} + 0.5|I_y|] \times [z_{min}, z_{min} + 0.5|I_z|]$
  $\mathcal{R}_{i,2} \leftarrow [x_{min}, x_{min} + 0.5|I_x|] \times [y_{min}, y_{min} + 0.5|I_y|] \times [z_{min}, z_{min} + 0.5|I_z|]$
  ⋮
  $\mathcal{R}_{i,8} \leftarrow [x_{min}, x_{min} + 0.5|I_x|] \times [y_{min} + 0.5|I_y|, y_{max}] \times [z_{min} + 0.5|I_z|, z_{max}]$
end function

Der Algorithmus ist, aufgrund der in der Praxis benötigten geringen Baumtiefe und den einfachen Berechnungsvorschriften, äußerst effizient und lässt sich leicht für vorhandene Voxelkarten umsetzen und als Vorverarbeitungsschritt in die Bahnplanung integrieren.

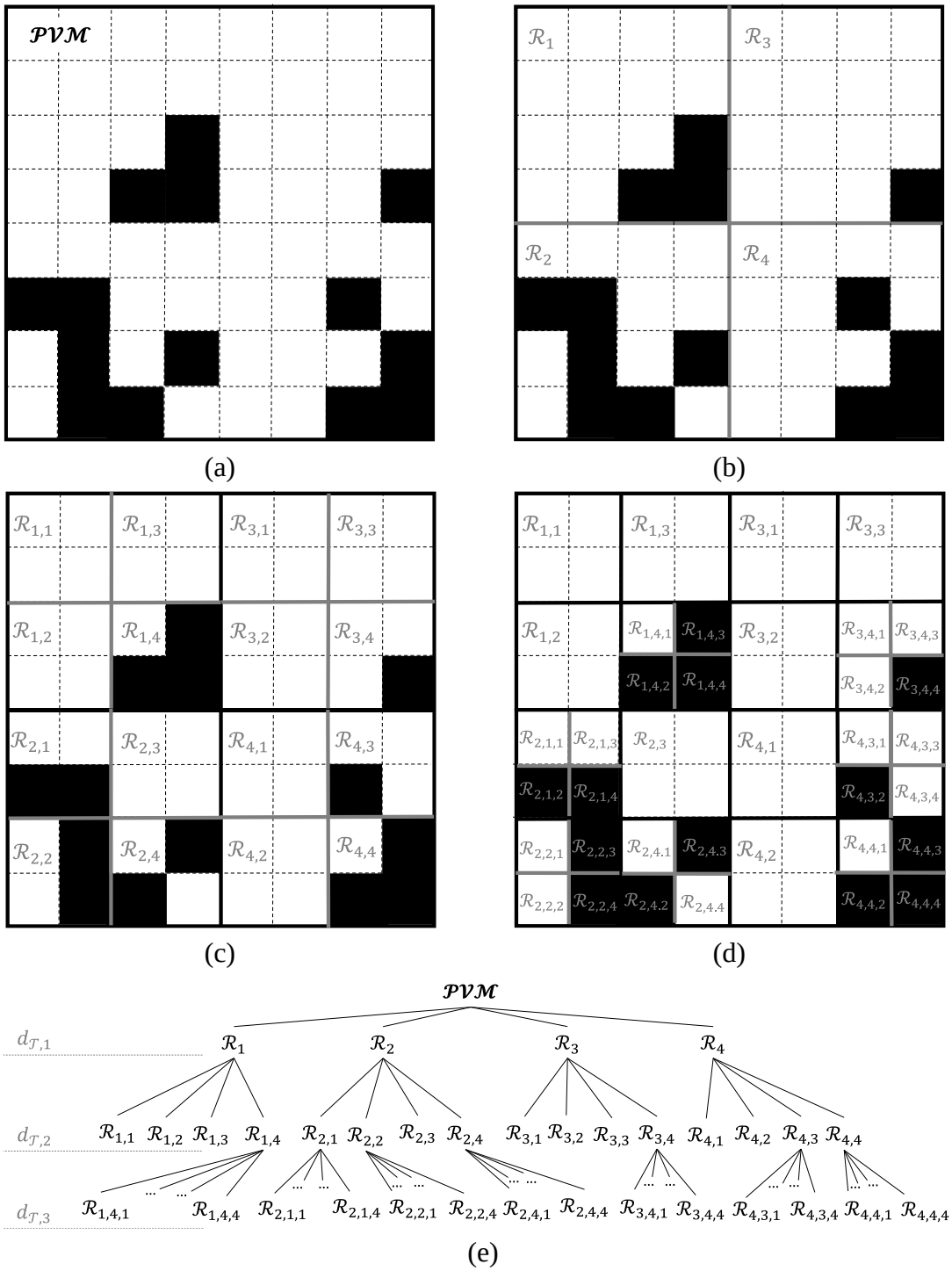


Abbildung 4-27: Adaptive Schrittweitensteuerung für globale Bahnplaner anhand zweidimensionalen Beispiel: Aus der Ursprungskarte (a) werden dabei in Abhängigkeit der Belegtheitsdichte neue Unterräume gebildet (b), (c), (d); Zugehöriger Baum beschreibt den Aufbau der Karte (e); Die Bildung des Graphen erfolgt nach iterative-depth-first

Optimalität und Fehlerabschätzung. Nachdem eine geeignete Berechnungsvorschrift für die Explorationsschrittweitensteuerung gefunden wurde, soll weitergehend untersucht werden, wie sich diese auf die Optimalität, im Sinne der kürzesten Bahn, auswirkt, vergleiche Abbildung 4-28. Dabei wird wieder angenommen, dass $\mathcal{A}^*$ eine optimale Lösung im Konfigurationsraum $\mathcal{C}$ liefert. Gleichzeitig gilt, dass die minimale Schrittweite auf die minimale Voxelgröße festgelegt wird, also $|s_{min}| = |v_{min}|$ gilt. Beschreibt nun $|d_i^*|$ die optimale Distanz von einem Zustand n_i zu einem Zustand n_{i+m}, in welchem $\mathcal{A}^*(s_{min})$ aufgrund einer Kollisionserkennung $n_{i+m+1} \in \mathcal{C}_o$ seine Explorationsrichtung ändern muss, dann kann für dasselbe Planungsproblem mit $\mathcal{A}^*(s)$ eine Abschätzung des maximalen Fehlers mit (4.52) erfolgen, wobei N die Anzahl an Orientierungswechsel darstellt.

$$e_{max} = \begin{cases} \sum_{i=1}^{N} \mathrm{mod}(|d_i^*| \cdot (|s|)^{-1}), für\ |d_i^*| > |s| \\ \sum_{i=1}^{N} \mathrm{mod}(|s| \cdot (|d_i^*|)^{-1}), sonst. \end{cases} \tag{4.52}$$

Dies ist leicht einsehbar, da für die Kombination $|d_i^*| \in (2m+1)$ und $|s| \in (2n+1)$ oder $|d_i^*| \in 2m$ und $|s| \in 2n$ für $\forall m, n \in \mathbb{N}_+$ immer ein Segmentfehler $e_j = 0$ entsteht. Für alle anderen Kombinationen wird $e_j = R$, da $|d_i^*| = k \cdot |s| + R$, mit $k \in \mathbb{N}_+$ gilt.

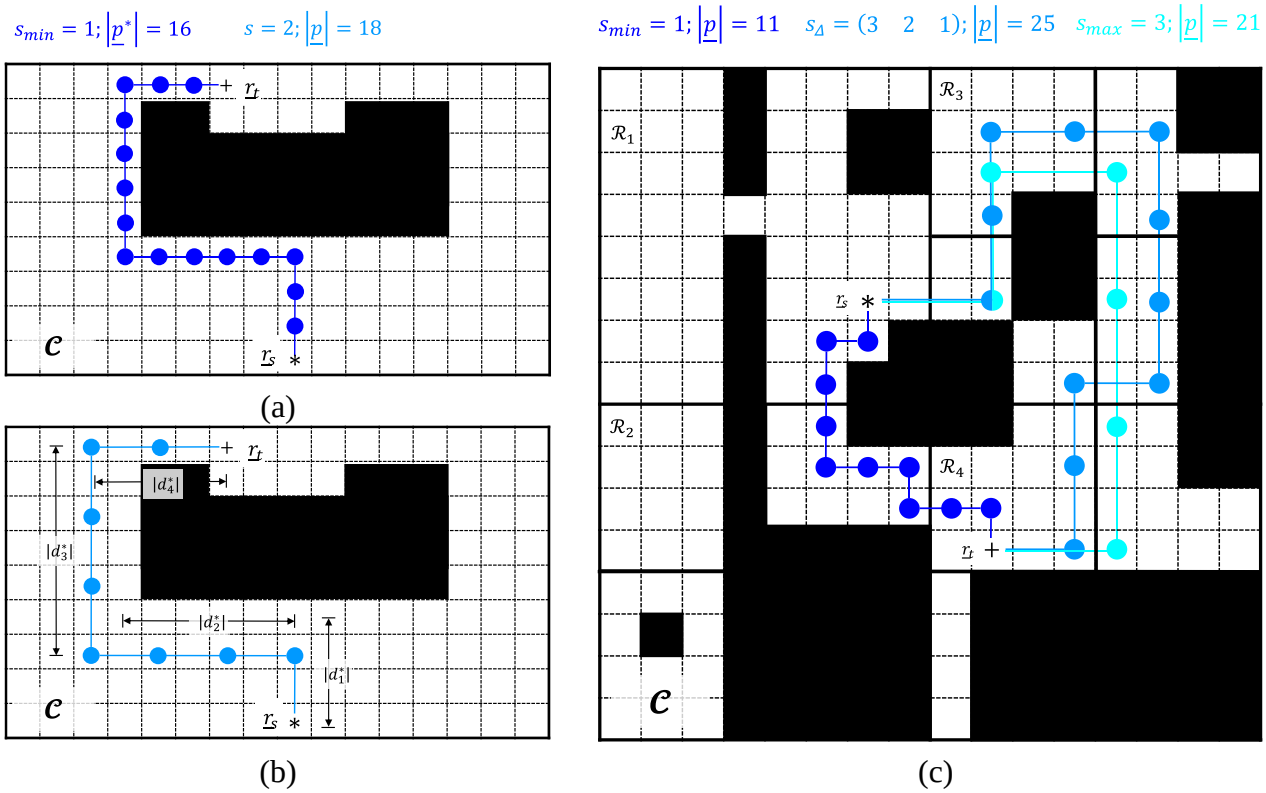


Abbildung 4-28: Fehlerabschätzung (a), (b); Schrittweitensteuerung (c): Durch Anwendung von s_{min} kann der kürzeste Pfad gefunden werden; Die Anwendung von s_Δ liefert für folgendes konstruiertes Beispiel ein schlechteres Ergebnis, als mit der maximalen Schrittweite s_{max}, da das Diskretisierungsgrid von s_{max} genau eine Schrittweite über die Hindernisgrenzen fällt

Die Fehlerabschätzung gilt nur für Fälle in denen $\mathcal{A}^*(s)$ in die gleiche Orientierungsrichtung wie $\mathcal{A}^*(s_{min})$ explorieren kann. In Abbildung 4-28 (a), (b) wird dies an einem einfachen Beispiel verdeutlicht. Für die minimale Schrittweite, welche genau dem Voxelgitter entspricht, kann der minimale Weg gefunden werden (a). Wird die Schrittweite verdoppelt, entsteht eine Abweichung von der optimalen Bahn (b). Dies kann dazu führen, dass für die adaptive Schrittweite in ungünstigen Kombinationen von $|d_i^*|$ zu s_Δ eine längere Bahn gegenüber der Planung mit s_{max} resultiert. Dies ist in Abbildung 4-28 (c) exemplarisch aufgezeigt.

4.5.3 Bahnnachbearbeitung

Da durch die verwendeten Planungsalgorithmen keine glatten Bahnen entstehen, werden diese vor Ausführung nachbearbeitet. Hierzu werden die Bahnstützpunkte $(\underline{r}_0, \underline{r}_1, \dots, \underline{r}_n)$, mit $\underline{r}_0 = \underline{r}_{start}$ und $\underline{r}_n = \underline{r}_{target}$ mittels Polynomen geeignet geglättet. Als mathematisches Werkzeug wird, aufgrund seiner Eigenschaften die Bernstein-Bezier-Raumkurve verwendet, welche zur Gruppe der approximativen Splines gehört. Die Vorteile sind dabei aufgrund folgender Eigenschaften gegeben, vergleiche auch [348]:

- Durch geeignete Intervallapproximation neigen die Polynome nicht zu Oszillationen.
- Der Start- und Zielpunkt wird durch das Stützpolygon interpoliert, also $\underline{p}(\underline{r}_0) = \underline{r}_s$ und $\underline{p}(\underline{r}_n) = \underline{r}_t$.
- Die Bahnzwischenpunkte $(\underline{r}_1, \underline{r}_2, \dots, \underline{r}_{n-1})$ bilden die konvexe Hülle der Raumkurve, beispielsweise des Stützpolygons.
- Für n-Stützpunkte gilt $\underline{p} \in \mathcal{C}^{n-1}(\mathbb{R})$.

Die Bernsteinschen Grundpolynome lassen sich gemäß Gleichung (4.53) angeben.

$$B_{i,n}(k) = \binom{n}{i} \cdot k^i \cdot (1-k)^{n-i}, mit\ k \in [0, k_{ipo}, \dots, 1]\ und\ i \in \mathbb{N}_+. \tag{4.53}$$

Das zugehörige Stützpolygon berechnet sich dann zu

$$\underline{p}(k) = \sum_{i=0}^{n} B_{i,n}(k) \cdot \underline{r}_i. \tag{4.54}$$

Durch die Wahl des Interpolationsparameters k_{ipo} lässt sich weitergehend die maximale Positionsänderung zwischen Interpolationswerten abbilden. Aufgrund der approximativen Eigenschaft sollten äußere Randpunkte einer zusätzlichen Kollisionsanalyse unterzogen werden.

4.5.4 Bahnplanungsframework

Zur Schaffung einer plattformunabhängigen Lösung, für die Nutzung der entwickelten Bahnplanungsfunktionalitäten, werden diese in ein gemeinsames Framework integriert. Dieses ist, vergleiche Abbildung 4-29, in vier Kernmodule gegliedert. Die Kopplung des Bahnplanungs-frameworks an industrielle Robotersteuerungen erfolgt dabei über eine Positionsschnittstelle, welche von allen gängigen Steuerungen angeboten wird. Die Kommunikationsanbindung, bezüglich Datenformat und Übertragungsart der Positionssollwerte aus dem Bahnplaner, muss an den herstellerspezifischen Steuerungsstandard adaptiert werden.

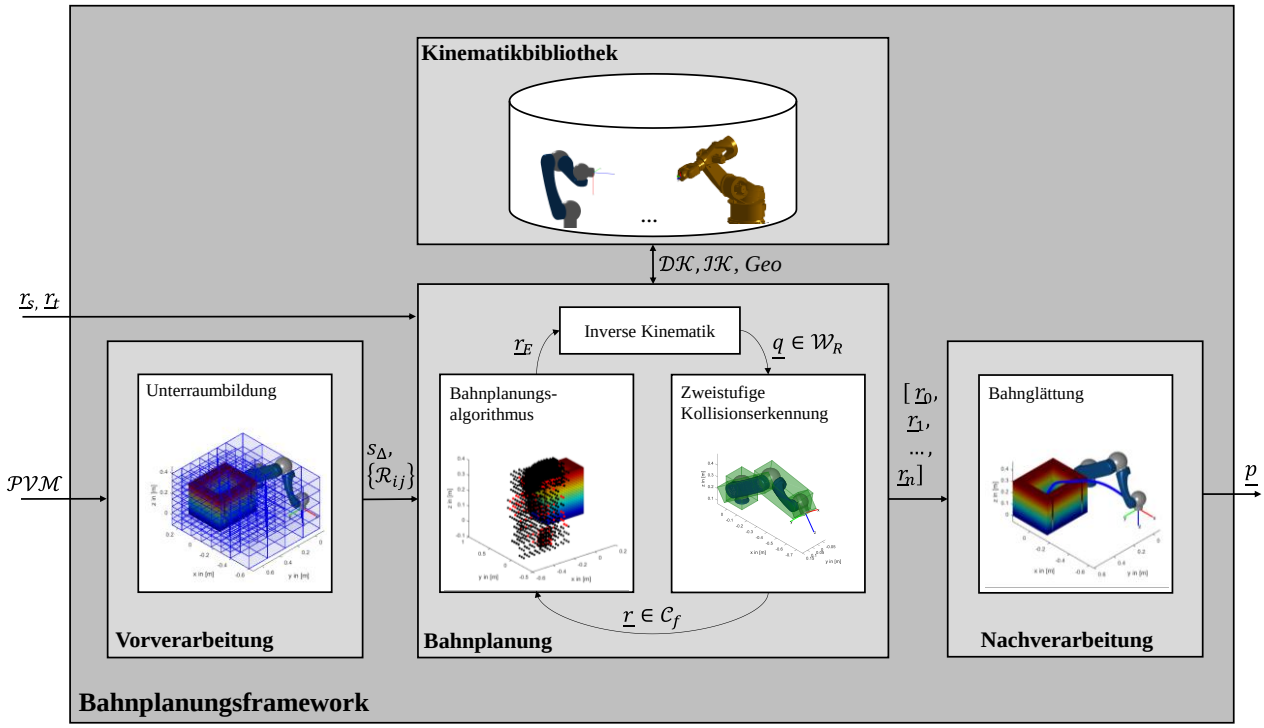


Abbildung 4-29: Architektur des Bahnplanungsframeworks

Durch die rein positionsbasierte Kopplung können weiterhin steuerungsinterne Funktionalitäten für die Trajektorienplanung verwendet werden.

Das Vorverarbeitungsmodul dient der automatischen Bestimmung der Unterräume, auf Basis der probabilistischen Voxelkarte und der dazugehörigen Schrittweite, dem jeweiligen Unterraum entsprechend.

Im Bahnplanungsmodul sind die vorgestellten such- und stichprobenbasierten Planungs-algorithmen sowie die Kollisionserkennung enthalten. Es lassen sich beliebig weitere globale Bahnplanungsalgorithmen oder andere Kollisionserkennungsalgorithmen integrieren. Nach einer Bahnplanungsanfrage (Start- und Zielpose, Parametrierung Planung und Auswahl Kinematik) lädt das Modul die zugehörige kinematische Beschreibung aus der Kinematikbibliothek. In dieser sind alle, für die Bahnplanung relevanten, roboterspezifischen Modelle hinterlegt (direkte und inverse Kinematik, Geometriemodell). Die Beschreibung kann dabei in dem urdf-Standard (unified robot description file) erfolgen.

Das Nachverarbeitungsmodul besitzt, neben der Funktion zur Spline-Approximation, verschiedene Filterfunktionen zur Bahnglättung. Spezifische Eigenschaften bezüglich Glattheit, maximaler Positionsänderung pro Interpolationstakt sowie Filterung definierter Frequenzspektren können somit benutzerspezifisch nachträglich parametriert werden.

4.5.5 Validierung und Ergebnisse

Abschließend soll das entwickelte Verfahren der adaptiven Schrittweitensteuerung, anhand vier verschiedener klassischer Benchmark-Szenarien, gegenüber dem Stand der Technik (ohne

Schrittweitensteuerung) evaluiert werden. Die verwendeten Benchmark-Szenarien und Auswertungsmetriken orientieren sich an den Arbeiten [349], [345].

Die Szenarien sind in Abbildung 4-30 und Abbildung 4-31 dargestellt. In Szenario 1 (Abbildung 4-30 (a), (c), (e)) muss der Roboter dabei einen Weg durch eine Durchgangspassage finden. Szenario 2 (Abbildung 4-30 (b), (d), (f)) stellt die Wegfindung anhand eines Haushaltsszenarios mit Tisch dar. Szenario 3 (Abbildung 4-31 (a), (c), (e)) beschreibt verschiedene Hindernisse im Raum, während Szenario 4 (Abbildung 4-31 (b), (d), (f)) eine klassische, industrielle Griff-in-die-Kiste Applikation zeigt.

Als Auswertungsmetriken werden folgenden Punkte verwendet:

- Zeit für die Bahnplanung t_p in [s],
- Zeit für die Kollisionserkennung $\mathrm{t_c}$ in [s],
- Länge der geplanten Bahn $\left|\underline{p}\right|$ in [m],
- Abweichung von optimaler Bahn $\left|\Delta\underline{p}^*\right| \in \{x \in \mathbb{R}_+ | x \geq 1\}$,
- Glattheit der geplanten Bahn κ in [rad],
- Anzahl an explorierter Knoten $|V| \in \mathbb{N}_+$,
- Erfolgsrate $S \in [0 \ldots 1]$ innerhalb eines Planungsintervalls von einer Minute.

Die verwendete Hardware entspricht dem Setup aus 4.2.4. Für die suchbasierten Planungsverfahren wird jedes Experiment pro Szenario fünfmal ausgeführt. Für die stichprobenbasierten Verfahren erfolgt eine Auswertung über 30 Testläufe, sodass eine ausreichend große Stichprobe gegeben ist.

Damit die Ergebnisse mit Schrittweitensteuerung, gegenüber einer konstanten Schrittweite, ganzheitlich evaluiert werden können, wird die adaptive Schrittweite mit den Intervallgrenzen $s_\Delta \in [s_{min}, \ldots, s_{max}]$ festgelegt. Für die maximale Baumtiefe der Unterraumbildung ergibt sich ein experimentell geeigneter Wert von $d_{\mathcal{T},max} = 3$. Eine weitere Unterteilung hat für die betrachteten Benchmarks aufgrund des Verhältnisses von Arbeitsraum und Hindernissobjekten, beziehungsweise der Voxelauflösung keine besseren Ergebnisse mehr erbracht. Die Berechnung der Schrittweite, Gleichung (4.50), erfolgt hier mit (4.55).

$$s_{\Delta,d_\mathcal{T}} = \frac{1}{2} \cdot s_{\Delta,d_\mathcal{T}-1}. \tag{4.55}$$

Abbildung 4-30 zeigt die Ergebnisse für suchbasierte Bahnplaner und Abbildung 4-31 für stichprobenbasierte Bahnplaner mit der Schrittweitensteuerung und den vier gewählten Benchmarkszenarien auf. Dabei sind von oben nach unten die einzelnen Ausgaben der Verarbeitungsschritte dargestellt. In (a) und (b) ist jeweils die Aufteilung der Unterräume mit obig beschriebener Parametrierung dargestellt. In (c) und (d) ist der explorierte Raum abgebildet. Die nachbearbeiteten Bahnen, welche vom Roboter ausgeführt werden, sind in (e) und (f) beschrieben.

Es zeigt sich deutlich, wie die Explorationsdichte in der Nähe der Hindernisse steigt, beziehungsweise im freien Raum abfällt, wodurch eine bedarfsgerechte, automatische Parametrierung der Schrittweite erzielt wird. Die genauen Effekte bezüglich der Auswertungsmetriken sollen folgend näher betrachtet werden.

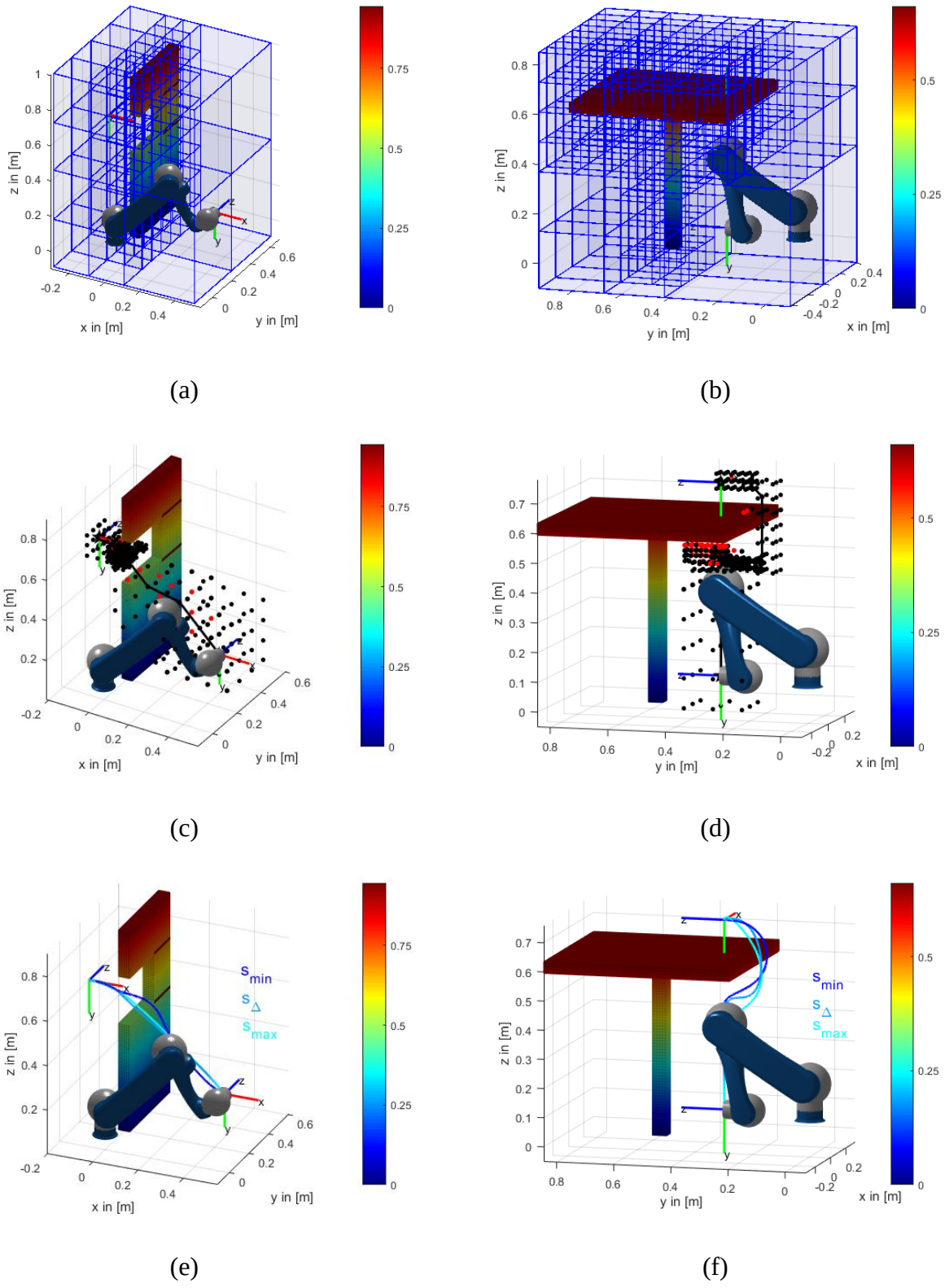
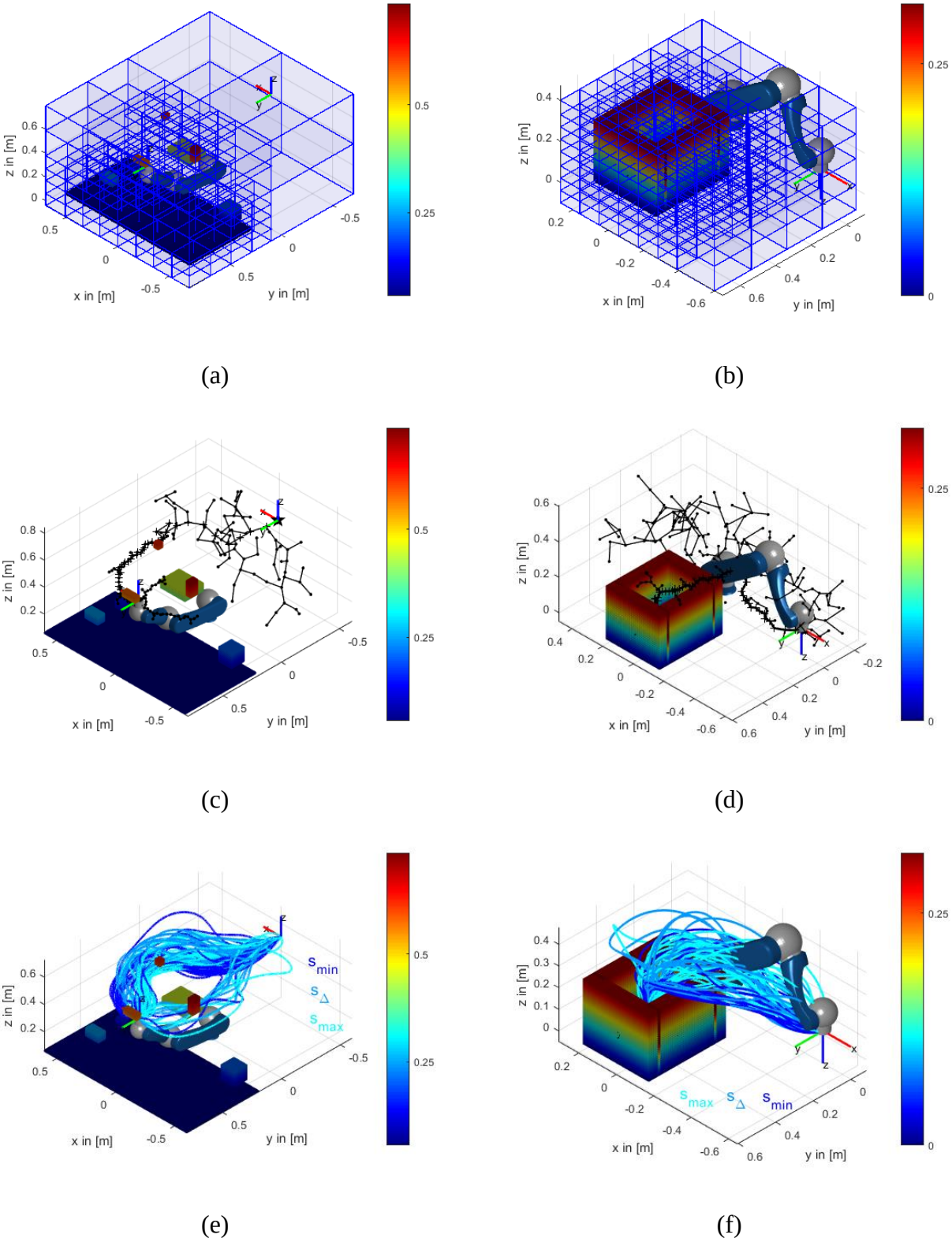


Abbildung 4-30: Planungsergebnisse mit suchbasierten Planern: Szenario 1- Durchgangspassage mit A*-Planer (a), (c), (e); Szenario 2- Tisch mit Greedy-Planer (b), (d), (f); Unterräume (a), (b); Explorierter Raum mit Schrittweitensteuerung (c), (d); Geplante Bahnen (e), (f)



(a) (b)

(c) (d)

(e) (f)

Abbildung 4-31: Planungsergebnisse mit stichprobenbasierten Planern: Szenario 3- Belegter Raum mit bi-RRT-Planer (a), (c), (e); Szenario 4- Schacht mit RRT-Planer (b), (d), (f); Unterräume (a), (b); Explorierter Raum mit Schrittweitensteuerung (c), (d); Geplante Bahnen (e), (f)

Zur rein visuellen Verdeutlichung der Methode ist in Abbildung 4-32 die Exploration ohne Schrittweitensteuerung, für die einzelnen Szenarien mit der Schrittweite s_{min}, dargestellt. Vergleicht man diese mit den Abbildung 4-30 (c), (d) und Abbildung 4-31 (c), (d), werden direkt die Vorteile einer Schrittweitensteuerung ersichtlich.

Im Fall der suchbasierten Planer für Szenario 1 und Szenario 2 in Abbildung 4-32 (a) und (b) ist die Anzahl an zu explorierenden Knoten, bis zur Lösungsfindung, deutlich vergrößert. Vor allem bei Verwendung des A*-Planer ist ohne Schrittweitensteuerung eine massiv höhere Planungszeit zu erwarten, vergleiche hierzu Abbildung 4-32 (a).

Für die stichprobenbasierten Planer, welche in Szenario 3 und Szenario 4 Verwendung finden (Abbildung 4-32 (c) und (d)), ist dieser Effekt ebenfalls, wenn auch nicht so stark, zu beobachten. Für den RRT-Planer in Abbildung 4-32 (d) wird aber besonders die Problematik einer fest parametrierten Schrittweite deutlich. Durch die zu klein gewählte Explorationsschrittweite, kommt es im freien Raum zu vielen Verzweigungen, wodurch die Zeit zur Lösungsfindung signifikant negativ beeinflusst wird. Dies hat ebenso Auswirkungen auf die Erfolgsrate, die hierdurch merklich sinkt.

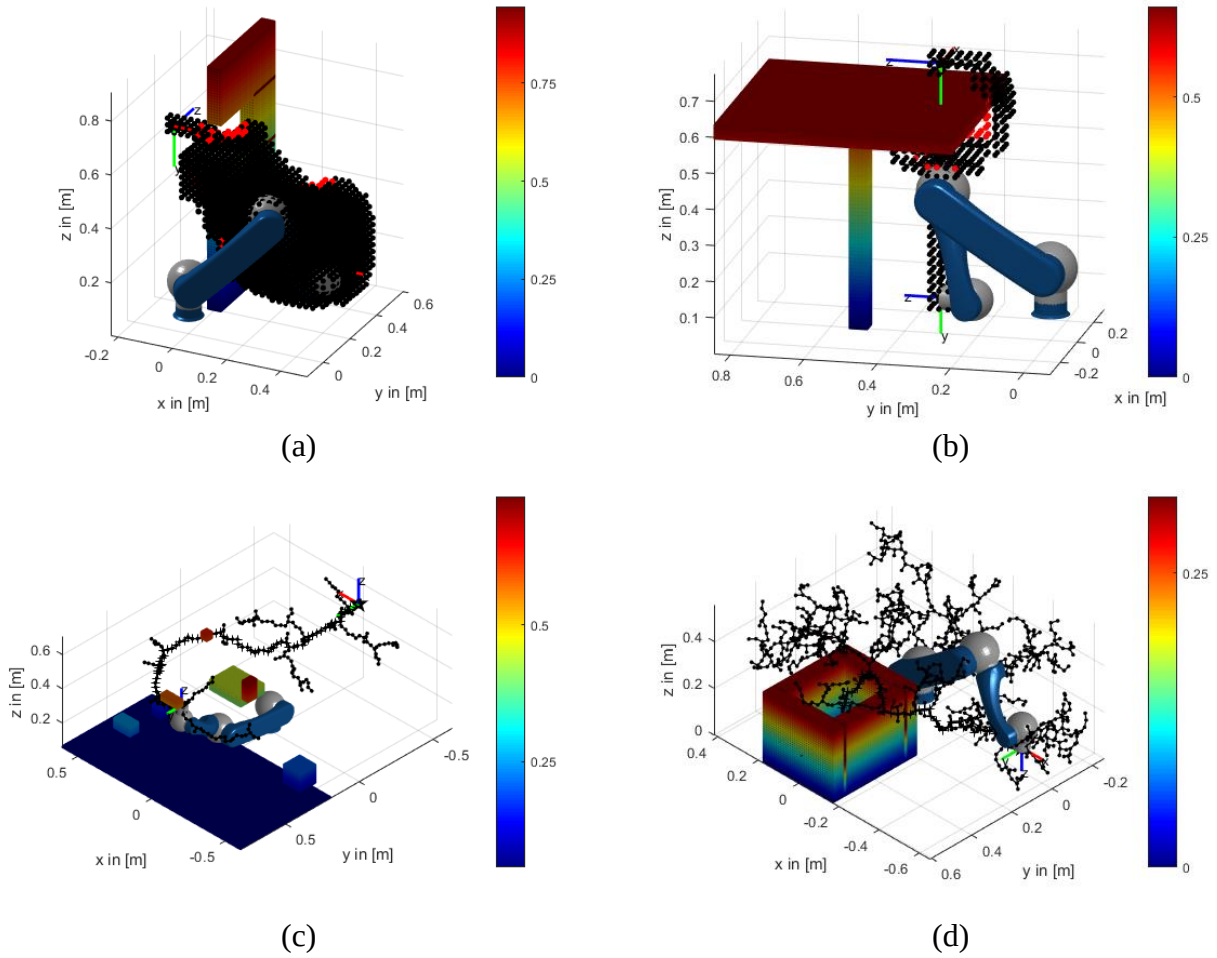


Abbildung 4-32: Explorierte Räume ohne Schrittweitensteuerung: Szenario 1- Durchgangspassage mit A*-Planer (a); Szenario 2- Tisch mit Greedy-Planer (b); Szenario 3- Belegter Raum mit bi-RRT-Planer; Szenario 4- Schacht mit RRT-Planer (d)

In den nachfolgenden Tabellen, Tabelle 4-3 bis Tabelle 4-6 sind für die vier Szenarien die Ergebnisse der Bahnplanungsmetriken als arithmetisches Mittel für alle Planer aufgeführt. Um einen besseren Einblick in die Ergebnisse bei stichprobenbasierten Verfahren zu bekommen, sind in Abbildung 4-33 bis Abbildung 4-36 zusätzlich die Metriken als Box-Plot dargestellt.

Bei den suchbasierten Planern zeigt die adaptive Schrittweite besonders ihre Vorteile auf, vor allem bei der Benutzung des A*-Planers. Durch die deutliche Reduktion an zu explorierenden Knoten, kann in der Planungszeit eine Einsparung um bis zu 70% erreicht werden, was sich weitergehend positiv auf die Zeit für die Kollisionsanalyse auswirkt. Die Abweichung von der optimalen Bahn ist zudem äußerst gering und führt in der Regel zu einer geringeren Bahnlänge, als bei der Verwendung der maximalen Schrittweite. In Szenario 1 ist dies nicht der Fall, was auf die Diskretisierungseffekte, vergleiche 4.5.2, zurückzuführen ist. Über die Glattheit der Bahn lässt sich keine direkte Aussage treffen, da diese zu stark vom Szenario abhängt. Die Verwendung des Greedy-Planers führt im Allgemeinen zu einer geringeren Explorationsrate, wodurch der zeitliche Vorteil nicht in extremen Maße zu tragen kommt. Auch hier wird jedoch im adaptiven Fall eine geringere Anzahl an zu explorierenden Knoten benötigt. Die Abweichung von der optimalen Bahn schwankt dabei stark in Abhängigkeit des Szenarios. Auch wird in Szenario 1 und Szenario 4 durch Greedy, eine bessere Bahn mit der maximalen Schrittweite gefunden, da eine kleine Schrittweite zu vielen Verzweigungen führen kann. Durch die Schrittweitensteuerung kann die Erfolgsrate zur Bahnfindung erhöht werden, da eine ungünstig gewählte, maximale Schrittweite zu keinem Ergebnis führen kann, vergleiche Szenario 3.

Auch bei der Verwendung von stichprobenbasierten Planern zeigt sich ein Vorteil, in der günstigeren Exploration bei Verwendung der Schrittweitensteuerung auf. In allen Fällen kann das arithmetische Mittel der explorierten Knoten reduziert werden, wodurch sich meistens auch die Planungszeit, in Bezug auf die Verwendung der minimalen Schrittweite, verringert. Prinzipiell ist die zeitliche Einsparung jedoch geringer als bei den suchbasierten Planern, da durch die statistische Natur eine oftmals schnellere Lösungsfindung erfolgt. Aufgrund der fehlenden Optimalität von RRT- und bi-RRT-Planern kann keine allgemeine Aussage über den Einfluss auf die Abweichung von der optimalen Bahn gegeben werden. Über alle Experimente hinweg sind die Ergebnisse mit minimaler, adaptiver und maximaler Schrittweite als ungefähr gleich zu bewerten. Auch hier kann allerdings, durch den Einsatz der adaptiven Schrittweite, die Erfolgsrate erhöht werden, wodurch der Planungsalgorithmus mit Schrittweitensteuerung im Allgemeinen als robuster, gegenüber der Struktur eines Szenarios zu bewerten ist. Dies kann vor allem in engen Passagen ausgenutzt werden, in denen RRT basierte Algorithmen prinzipiell ungeeignet sind.

Zusammenfassend zeigt sich, dass die adaptive Schrittweitensteuerung große Vorteile für die Bahnplanung hat. Durch die formal allgemein gehaltene Methode auf Basis von Voxelkarten, kann diese, unabhängig vom gewählten globalen Bahnplanungsalgorithmus, verwendet werden. Durch die Anpassung der Schrittweite an die lokalen Gegebenheiten der Unterräume, wird ein reduzierter Suchraum sowie eine erhöhte Erfolgsrate für die Bahnfindung erreicht. Auch ist über alle Experimente hinweg ein Trend in Richtung Verbesserung der Bahngüte zu erkennen. Der Einsatz des Verfahrens eignet sich somit hervorragend, wenn geringe Planungszeiten eine hohe Priorität spielen. Zukünftig sollten noch weitere Bahnplanungsalgorithmen in das Bahnplanungsframework eingepflegt und in Kombination mit der adaptiven Schrittweitensteuerung evaluiert werden.

Tabelle 4-3: Szenario 1: Arithmetisches Mittel der evaluierten Bahnplanungsmetriken

Szenario 1-Durchgang	***Greedy*** (s_{min})	***Greedy*** (s_{Δ})	***Greedy*** (s_{max})	***A**** (s_{min})	***A**** (s_{Δ})	***A**** (s_{max})	***RRT*** (s_{min})	***RRT*** (s_{Δ})	***RRT*** (s_{max})	***bi-RRT*** (s_{min})	***bi RRT*** (s_{Δ})	***bi-RRT*** (s_{max})
t_p *in* $[s]$	0.067	0.082	0.011	3.267	0.077	0.029	3.801	3.632	3.903	0.135	0.075	0.026
t_c *in* $[s]$	0.241	0.657	0.054	13.34	0.538	0.277	11.49	9.941	9.669	1.667	0.819	0.288
$\lvert\underline{p}\rvert$ *in* $[m]$	0.872	1.140	0.829	0.896	0.844	0.829	1.515	1.639	1.569	1.228	1.269	1.196
$\lvert\underline{\Delta p}^*\rvert \in \mathbb{R}_{+\setminus 1}$	1.209	1.582	1.150	1.243	1.171	1.150	2.101	2.273	2.176	1.703	1.761	1.658
κ *in* $[rad]$	0.347	2.009	0.468	0.299	0.253	0.468	0.839	1.049	1.452	0.943	1.142	1.440
$\lvert V\rvert \in \mathbb{N}_+$	336	330	106	3985	359	143	991	747	742	117	62	23
$S \in [0 \dots 1]$	1	1	1	1	1	1	1	1	1	1	1	1

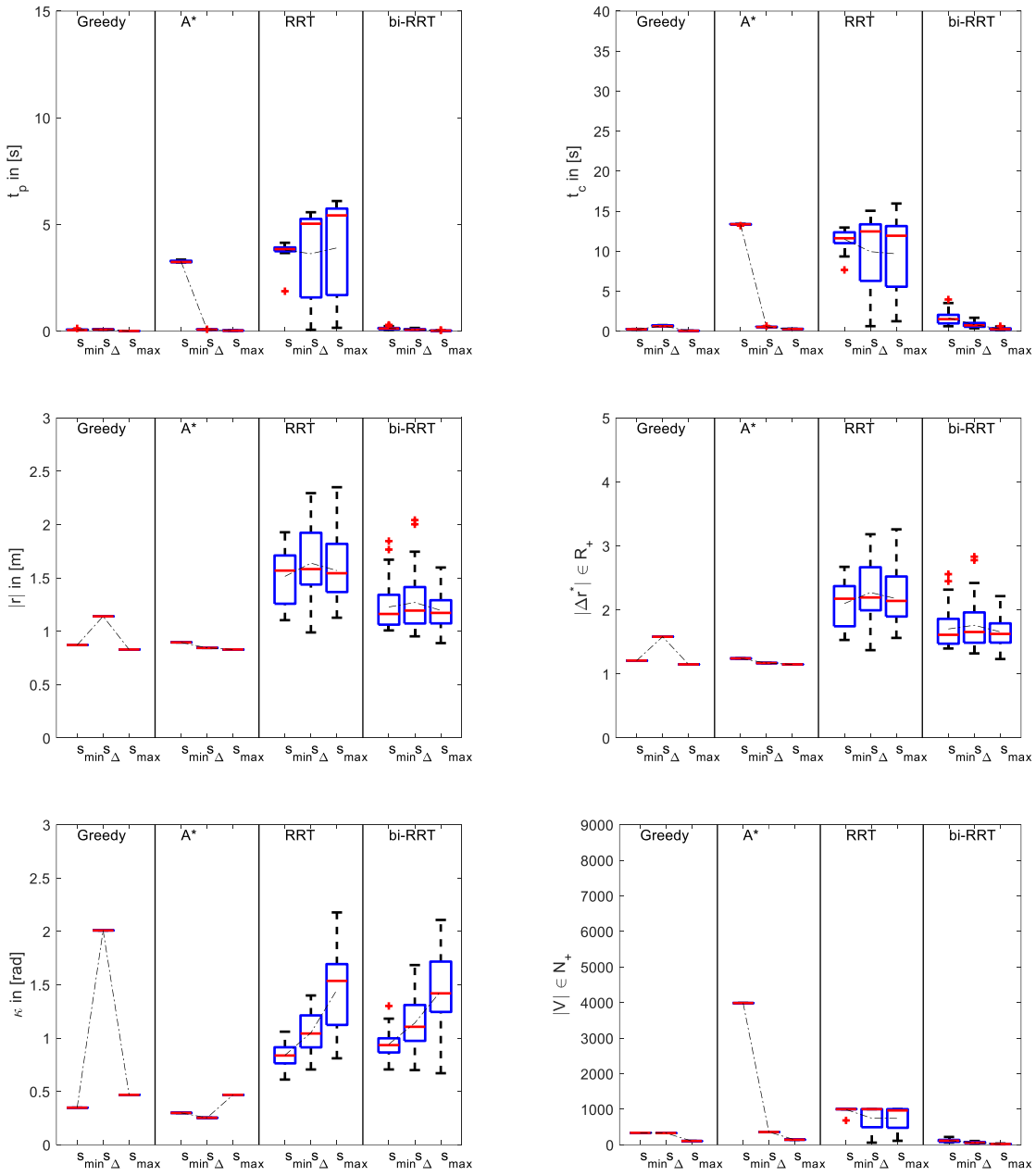


Abbildung 4-33: Szenario 1: Box-Plots für Bahnplanungsmetriken

Tabelle 4-4: Szenario 2: Arithmetisches Mittel der evaluierten Bahnplanungsmetriken

Szenario 2-Tisch	***Greedy*** (s_{min})	***Greedy*** (s_{Δ})	***Greedy*** (s_{max})	***A**** (s_{min})	***A**** (s_{Δ})	***A**** (s_{max})	***RRT*** (s_{min})	***RRT*** (s_{Δ})	***RRT*** (s_{max})	***bi-RRT*** (s_{min})	***bi RRT*** (s_{Δ})	***bi-RRT*** (s_{max})
t_p *in* $[s]$	0.098	0.089	0.017	3.853	0.932	0.137	3.361	3.718	3.905	0.164	0.147	0.038
t_c *in* $[s]$	0.534	0.587	0.072	10.32	4.421	0.512	5.304	4.988	4.044	1.338	1.214	0.321
$\lvert\underline{p}\rvert$ *in* $[m]$	1.121	1.066	1.257	0.847	0.862	1.057	1.396	1.435	1.528	1.208	1.294	1.254
$\lvert\Delta\underline{p}^*\rvert \in \mathbb{R}_{+\setminus 1}$	1.724	1.640	1.934	1.304	1.326	1.626	2.148	2.208	2.350	1.858	1.990	1.929
κ *in* $[rad]$	0.882	1.999	1.782	0.259	0.442	0.617	0.813	0.903	1.298	0.911	1.010	1.506
$\lvert V\rvert \in \mathbb{N}_+$	468	395	112	4800	1573	391	987	896	825	121	107	28
$S \in [0 \dots 1]$	1	1	1	1	1	1	0.87	0.97	1	1	1	1

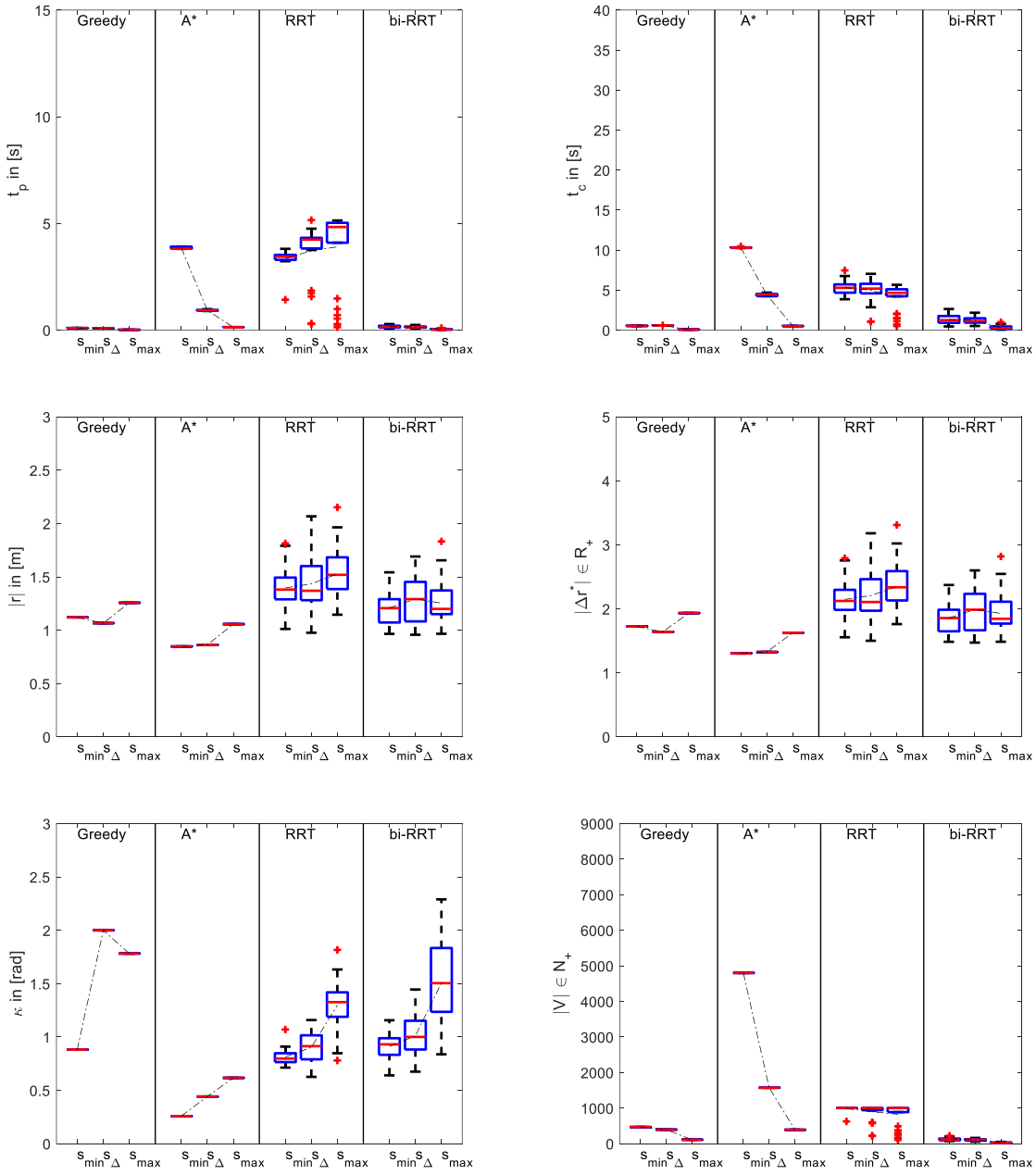


Abbildung 4-34: Szenario 2: Box-Plots für Bahnplanungsmetriken

Tabelle 4-5: Szenario 3: Arithmetisches Mittel der evaluierten Bahnplanungsmetriken

Szenario 3-Raum	***Greedy*** (s_{min})	***Greedy*** (s_{Δ})	***Greedy*** (s_{max})	***A**** (s_{min})	***A**** (s_{Δ})	***A**** (s_{max})	***RRT*** (s_{min})	***RRT*** (s_{Δ})	***RRT*** (s_{max})	***bi-RRT*** (s_{min})	***bi-RRT*** (s_{Δ})	***bi-RRT*** (s_{max})
t_p *in* $[s]$	0.374	0.365	-	7.929	2.188	0.141	5.291	6.111	7.354	0.384	0.170	0.075
t_c *in* $[s]$	4.067	4.089	-	27.86	18.69	0.695	8.552	8.026	6.556	4.209	2.302	0.868
$\lvert\underline{\boldsymbol{p}}\rvert$ *in* $[m]$	2.606	2.162	-	1.518	1.831	1.712	2.199	2.210	2.246	2.029	1.980	1.945
$\lvert\underline{\Delta \boldsymbol{p}}^*\rvert \in \mathbb{R}_{+\setminus 1}$	2.172	1.802	-	1.265	1.526	1.427	1.833	1.842	1.872	1.691	1.650	1.621
$\boldsymbol{\kappa}$ *in* $[rad]$	1.466	2.023	-	0.372	0.396	0.613	0.807	0.893	1.138	0.749	0.810	1.059
$\lvert \boldsymbol{V} \rvert \in \mathbb{N}_+$	1022	878	-	8221	3168	520	968	760	756	225	116	47
$\boldsymbol{S} \in [0 \dots 1]$	1	1	0	1	1	1	1	1	1	1	1	1

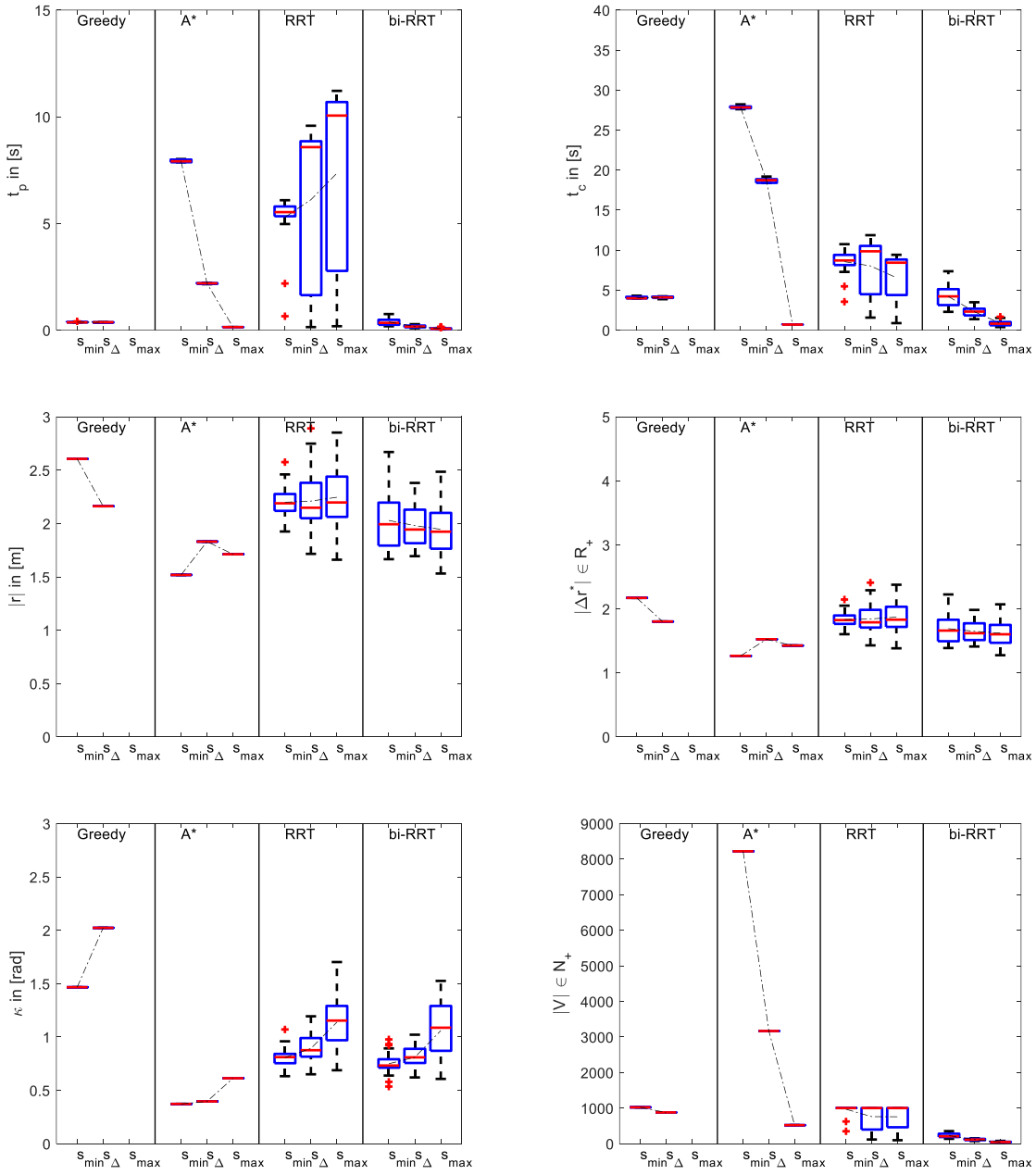


Abbildung 4-35: Szenario 3: Box-Plots für Bahnplanungsmetriken

Tabelle 4-6: Szenario 4: Arithmetisches Mittel der evaluierten Bahnplanungsmetriken

Szenario 4-Schacht	***Greedy*** (s_{min})	***Greedy*** (s_{Δ})	***Greedy*** (s_{max})	***A**** (s_{min})	***A**** (s_{Δ})	***A**** (s_{max})	***RRT*** (s_{min})	***RRT*** (s_{Δ})	***RRT*** (s_{max})	***bi-RRT*** (s_{min})	***bi-RRT*** (s_{Δ})	***bi-RRT*** (s_{max})
t_p *in* $[s]$	0.081	0.076	0.018	2.991	0.979	0.085	3.343	2.583	2.713	0.279	0.225	0.057
t_c *in* $[s]$	1.344	1.647	0.326	34.48	14.04	1.229	23.35	17.39	17.34	8.491	6.804	1.685
$\lvert \underline{p} \rvert$ *in* $[m]$	1.325	1.308	0.973	0.932	0.947	0.963	1.358	1.512	1.396	1.358	1.316	1.378
$\lvert \Delta \underline{p}^* \rvert \in \mathbb{R}_{+\setminus 1}$	1.965	1.939	1.443	1.381	1.404	1.428	2.013	2.242	2.070	2.014	1.951	2.043
κ *in* $[rad]$	1.048	1.182	0.452	0.420	0.373	0.559	0.846	0.904	1.166	0.893	0.905	1.371
$\lvert V \rvert \in \mathbb{N}_+$	471	422	109	3500	1744	193	965	727	675	144	113	29
$S \in [0 \dots 1]$	1	1	1	1	1	1	0.63	0.73	1	1	1	1

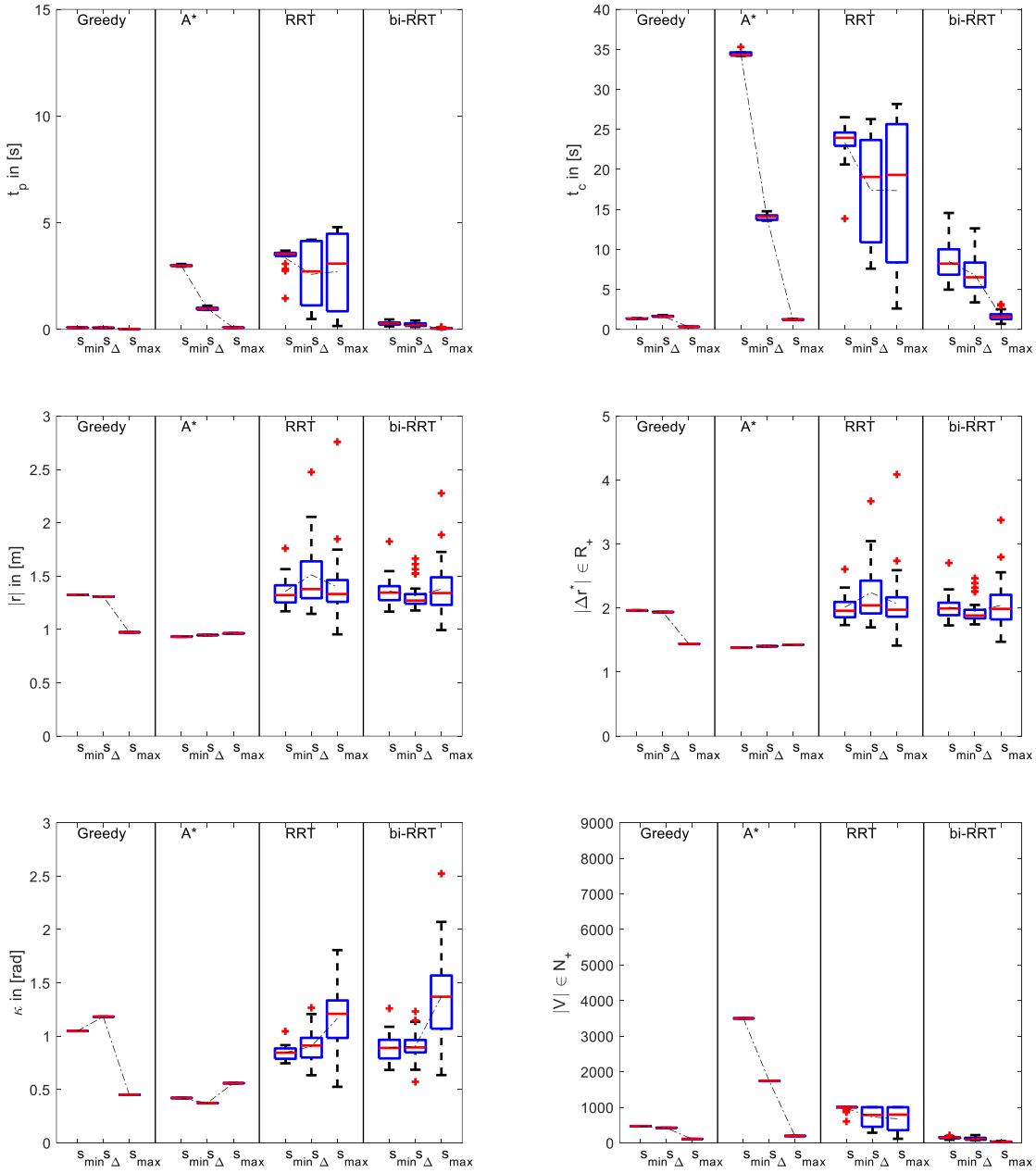


Abbildung 4-36: Szenario 4: Box-Plots für Bahnplanungsmetriken

4.6 Informationsfluss der Manipulationsplanung

In diesem Unterkapitel sollen abschließend Funktionszusammenhänge und Struktureigenschaften der gesamten Planungsfunktionalitäten, die in diesem Kapiteln entwickelt wurden, ganzheitlich anhand des Informationsflusses dargestellt werden.

Die Manipulationsplanung arbeitet zum einen, auf der aus dem CAD-Baugruppenmodell extrahiertem, relationalen Baugruppenmodell $\mathcal{RM}$ mit zugehörigem passivem Graph $\mathcal{G}_{ssr}^{-}$. Zum anderen, aus der extrahierten Information aus dem visuellen Perzeptionsprozess mit Szenenanalyse, welche die Generierung des aktiven Graph $\mathcal{G}_{ssr}^{+}$ mit räumlich symbolischen Relationen erlaubt. Auf Grundlage der Bayes-Filterbank, Gleichung (4.27), erfolgt die Fusionierung beider Kontaktgraphen auf symbolischer Ebene, unter Berücksichtigung der Komponentenzustände. Der gewonnene, fusionierte Graph $\mathcal{G}_{ssr}$ wird zur Bestimmung des relationalen Graphs $\mathcal{G}_{sdof}$ verwendet, der im symbolischen Prozessor folglich die Planung der Menge an Manipulationsprimitiven $\{\mathcal{MP}\}$ erlaubt. Die Sequenzierung führt schlussendlich zu einer lokal optimalen Reihenfolge $\langle\mathcal{MP}\rangle^*$ der einzelnen Manipulationsprimitive, über die Ansätze der Sichtbarkeit und Zugänglichkeit. Können nicht alle Komponenten der passiven Information, die zur Lösung der Aufgabe benötigt werden, der aktiven Information zugeordnet werden, muss eine weitere Aufgabenexploration, nach Ausführung der Teilmenge an Manipulationen, erfolgen. Abschließend bestimmt die Bahnplanung die einzelnen Teilbahnen $\langle\underline{p}\rangle$, den zugehörigen Manipulationsprimitiven entsprechend, mittels Wegpunkte der einzelnen Komponenten $\langle\underline{r}\rangle^*$, sowie der gewählten Demontagerichtung $\underline{d}^{(i)} \in \mathbb{W}_D^{(i)}$. In Abbildung 4-37 ist der Informationsfluss des Planungssystems dargestellt. Somit kann durch die in diesem Kapitel entwickelten Planungsverfahren alle relevante Information, die zur Generierung des Steuerprogramms notwendig ist, bereitgestellt werden.

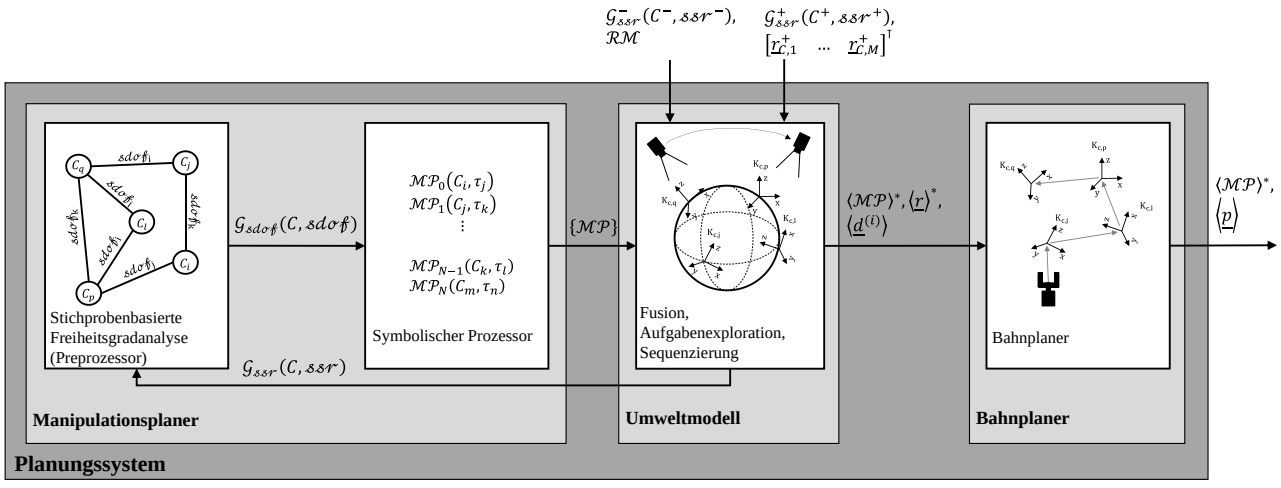


Abbildung 4-37: Informationsfluss der gesamten Planung zur Generierung des symbolischen Manipulationsplans mit zugehöriger Bahninformation

4.7 Zusammenfassung und Diskussion

Die in diesem Kapitel entwickelten Methoden, erlauben erstmalig die ganzheitliche, automatische Planung von Robotermanipulationen, wie diese unter den Rahmenbedingungen der automatisierten Wartung und Instandsetzung benötigt werden. Hierdurch wird es einem Robotersystem ermöglicht,

Manipulationspläne für die Wartung und Instandsetzung autonom, auf Basis von CAD- und RGBD-Daten, zu erzeugen. Zusammenfassend sollen die einzelnen Beiträge, bezüglich ihrer spezifischen Vor- und Nachteile kritisch diskutiert werden.

Manipulationsplanung. Der Manipulationsplaner ist derart gestaltet, dass er eine zeiteffiziente Berechnung der relativen Demontageräume gestattet. Durch die Rückkopplung von Sensorinformationen in den Planungsprozess, wird auch eine Planung unter Umweltunsicherheiten erlaubt. Aufgrund der Beschreibung der Planung, mittels symbolischer Aktionen, kann eine hohe Abstraktion erzielt werden, die vor allem für die Ausführung von Vorteil ist. Nachteilig an dem Planer ist, dass nur lokal zugängliche Bauteile, aufgrund der relativen Demontageräume in der Optimierung berücksichtigt werden, wodurch theoretisch schlechtere Lösungen möglich sind. Auch lassen sich nur lineare und sequentielle Pläne mit der Methode generieren. Für eine zukünftige Erweiterung sollten diese Nachteile behoben werden, sodass auch nichtlineare und nichtmonotone Pläne erzeugbar sind. So kann der Planer auch für eine Nutzung bei Zwei-Arm-Kinematiken effizienter eingesetzt werden. Auch sollten Stabilitätseigenschaften in der Demontagereihenfolge beachtet werden. Eine automatische Erkennung der Kontaktrelationen aus dem CAD mittels semantischer Annotation, würde den Autonomiegrad des Systems noch weiter erhöhen.

Visuelle Perzeption. Klassische Ansätze der erscheinungsbasierten Objekterkennung konnten erfolgreich, experimentell, am Beispiel verschiedener industrieller Komponenten, angewandt werden. Weiter wurde ein Ansatz vorgeschlagen, der die Ermittlung eines Kontaktgraphen mit symbolisch, räumlichen Relationen erlaubt, wie dieser für die Manipulationsplanung benötigt wird. Interessante Fragestellungen liegen vor allem in der Entwicklung einer verbesserten Szenen-analyse, sodass auch unter großen Umweltunsicherheiten eine erfolgreiche Planung möglich ist.

Umweltmodellierung, Aufgabenexploration und Sequenzierung. Die Nutzung von binären Bayes-Filtern gestattet eine geeignete Grundlage zur Informationsfusion. Durch die Aufgabenexploration, mittels dynamischer Gewichtung des Informationgain von Raumexploration und Objekterkennung wird in der Praxis für beide Aufgaben gleichzeitig eine gute Lösung erzeugt. Durch die Einführung der Sichtbarkeit und Zugänglichkeit lassen sich schlussendlich, lokal optimale Sequenzen an Manipulationsprimitiven bestimmen. Für die Aufgabenexploration sollten künftig auch Wegkosten in die Optimierung, zur Planung einer Sensorpose, einbezogen werden. So wäre der Ansatz nicht rein durch das Informationgain getrieben. Ein weiterer Vorteil könnte geschaffen werden, wenn die Sequenzierung eine Parallelisierung der Aktionszuordnung, vor allem in Hinblick auf einen Zwei-Arm-Kinematik, erlaubt.

Bahnplanung. Für die Bahnplanung ist eine effiziente Vorverarbeitungsstrategie, mittels adaptiver Schrittweitensteuerung, eingeführt worden. Experimentell konnte die Berechnungszeit bei den verwendeten Benchmarkproblemen um bis zu 70% (Planungsraum bis zu 90%) reduziert werden, ohne jedoch die Bahngüte zu reduzieren. Zukünftig sollten weitere Planer in das Framework integriert und mit der Vorverarbeitung evaluiert werden. Auch wäre es interessant, die Vorverarbeitung nicht nur anhand der globalen Belegtheit der Unterräume zu bestimmen sondern zusätzlich lokale Eigenschaften zu berücksichtigen. Dies würde eventuell den Vorteil bringen, dass die Grenzen der Unterregionen besser an die Belegtheitseigenschaften anpassbar wären und eine bessere Adaption der Schrittweite ermöglichen würde.

5 Steuerungsprogramm und Regelung

„Erfahrung ist nicht das, was einem zustößt.
Erfahrung ist das, was man aus dem macht,
was einem zustößt.“
Aldous Huxley

Im folgenden Kapitel wird ein Verfahren entwickelt, wie die aus der Planung gewonnene Information in ein ausführbares Anwenderprogramm übersetzt werden kann. Als Schnittstelle zwischen Planung und Ausführung wird der Formalismus der Aktionsprimitive verwendet. Zur automatischen Erzeugung eines Anwenderprogramms werden Dekompositionsvorschriften eingeführt, die regelbasiert, die jeweils benötigten Aktionsprimitive aus dem geplanten Manipulationsprimitiv erzeugen. Die Zerlegung erfolgt zur Laufzeit durch einen Interpreter, was eine hohe Flexibilität gewährleistet. Für die Ausführung der einzelnen Aufgaben werden neben einer Positionsregelung auch exterozeptive Regler, mit einer bildbasierten Regelung und einer Kraft-Momenten-Regelung, benötigt. Diese sind in den Interpreter integriert, sodass dieser prinzipiell für beliebige industrielle Robotersteuerungen Verwendung finden kann. Es wird ebenso auf die Integration der Regler und die unterlagerte Roboterregelung eingegangen. Abschließend erfolgen der experimentelle Entwurf und die Validierung der einzelnen Regler.

5.1 Anwenderprogrammgenerierung aus Manipulationsprimitiven

Weitergehend wird das Verfahren, zur automatischen Generierung des Anwenderprogramms, in Form von Aktionsprimitiven vorgestellt. Eingehend werden in 5.1.1 die wichtigsten Eigenschaften, des aus dem Stand der Technik bekannten Paradigmas, der Aktionsprimitive eingeführt. Darauf aufbauend werden in 5.1.2 die Dekompositionsvorschriften vorgestellt, die es erlauben symbolische Information in Form von Manipulationsprimitiven in Aktionsprimitive zu übersetzen. Hierzu müssen vorab eine Menge an konfigurierbaren Aktionsprimitiven festgelegt und implementiert werden. Die Konfiguration der Aktionsprimitiven erfolgt individuell durch die geplante Bahn-, Kraft-Momenten- oder Featurevorgabe, in Abhängigkeit des benötigten Reglers sowie den Werkzeugkommandos und dem jeweiligen Koordinatensystem. Die nötigen Entscheidungsregeln werden in 5.1.3 und das Verfahren zur Laufzeitinterpretation in 5.1.4 diskutiert. Somit wird eine durchgehende Prozesskette, von der symbolischen Planung über die subsymbolische Programmbeschreibung durch Aktionsprimitive bis zur numerischen Vorgabe von Sollwerten, erreicht. Teile dieses Unterkapitels sind bereits in der Vorarbeit [22] veröffentlicht.

5.1.1 Verwendetes Schnittstellenkonzept der Aktionsprimitive

Für die Kopplung von symbolischer Planung mit der Ausführungsebene, soll, aufgrund seiner bereits in 2.3.3 diskutierten Vorteile, das Konzept der Aktionsprimitive als subsymbolische Beschreibung des Anwenderprogramms, für die Steuerung der Robotermanipulationen verwendet werden. Die wesentlichen Eigenschaften der Aktionsprimitive sollen folglich eingeführt werden, die Ausführung ist den Arbeiten [146], [148] entnommen.

Ein Aktionsprimitiv sei ein elementarer Operator, welcher den kleinsten Schritt einer Roboteraufgabe darstellt und sowohl gesteuert als auch geregelt ausgeführt werden kann. Ein Aktionsprimitiv $\mathcal{AP}$ sei definiert als ein Tripel gemäß Gleichung (5.1).

$$\mathcal{AP} \coloneqq \langle \mathcal{HM}, \tau, \lambda \rangle. \tag{5.1}$$

Hierbei sei $\mathcal{HM}$ ein Hybrid Move, welcher die Roboterbewegung im festgelegten Koordinatensystem beschreibt. Dieser kann durch die Definition von Sensorfunktionen überwacht und durch die Aktivierung verschiedener Regler geregelt werden. Durch den Einsatz sensorgeführter Aktionen kann reaktiv mit der Umwelt interagiert werden. Dabei sei $\mathcal{HM}$ definiert durch das Tupel

$$\mathcal{HM} \coloneqq \langle TF, \mathcal{D} \rangle, \tag{5.2}$$

wobei TF das Task Frame (Koordinatensystem in welchem Roboterbewegung angegeben wird) spezifiziert und $\mathcal{D}$ die Menge der Sollwerte für die Bewegung zusammenfasst. Die Festlegung des TF erfolgt dabei in Bezug auf ein Reference Frame $RF \in \{tcp, base, world, joint \dots\}$, welches durch die Lage und Orientierung $\underline{r}^{RF} \coloneqq (x \quad y \quad z \quad \alpha_x \quad \beta_y \quad \gamma_z)^T$ für kartesisches Bezugsframe oder im Falle eines achsbasierten Bezugsframes durch $\underline{r}^{RF} \coloneqq (q_1 \quad \cdots \quad q_N)^T$ für einen Roboter mit N-Gelenken. Für jeden Sollwert $d_i \in \mathcal{D}$ müssen zusätzlich die vier Attribute $dof, level, d$ und $control$ angegeben werden. Mit dof wird der Freiheitsgrad von TF angegeben, auf welchen sich der Sollwert d_i bezieht, also für kartesische Werte mit $dof \in (x \quad y \quad z \quad \alpha_x \quad \beta_y \quad \gamma_z)^T$ und für Gelenkwinkelvorgaben mit $dof \in (q_1 \quad \cdots \quad q_N)^T$. Über das $level$-Attribut wird die Gültigkeit des Sollwerts festgelegt sowie Alternativen für einen Sollwert definiert, falls ein höher priorisierter Sollwert nicht erreicht werden kann. Das Attribut $d \in \mathbb{R}$ entspricht dem numerischen Wert der Sollgröße, während mit $control$ die verwendete Regelung mit zugehörigen Parametern angegeben wird.

Durch das Werkzeugkommando τ können definierte Werkzeuge, aus einer Menge an verfügbaren Werkzeugen $\{tool_1, tool_2, \dots, tool_i\}$, adressiert und durch eine gegebene Menge an Steuerkommandos $cmd_{tool_j} \in \left\{cmd_{tool_j,1}, cmd_{tool_j,2}, \dots, cmd_{tool_j,n}\right\}$ kommandiert werden.

$$\tau \coloneqq \left\{tool_j | tool_j = \{tool_1, tool_2, \dots, tool_i\}, \left\{cmd_{tool_j,1}, \dots, cmd_{tool_j,n}\right\}\right\}. \tag{5.3}$$

Die Beendigung eines Aktionsprimitiv wird durch die Abbruchbedingung λ bei Erreichen des *TRUE*-Zustands angegeben. Hierzu werden die für das Aktionsprimitiv eingesetzten Sensoreingaben S auf einen booleschen Wert abgebildet, folglich

$$\lambda: S_1 \times S_2 \times \dots \times S_i \rightarrow \{TRUE, FALSE\}. \tag{5.4}$$

Koordinatensysteme. Für die Beschreibung der einzelnen Bewegungen, werden gemäß Abbildung 5-1, die Koordinatensysteme $\{world, base, tcp, fts, rgbd, gripper, screwdriver, object\}$ eingeführt. Das $base$-Frame liegt ortsfest in Achse 1 des Roboters. Das $world$-Frame ist frei definierbar und wird als ortsfestes Bezugskoordinatensystem aufgabenabhängig verwendet. Die Sensorkoordinatensysteme des Kraft-Momenten (fts) und des RGBD-Sensors ($rgbd$) sind abhängig von der Roboterkonfiguration. Die Bestimmung der relativen Position und Orientierung des $rgbd$-Frame in Bezug auf das tcp-Frame erfolgt über die Hand-Auge-Kalibrierung [350].

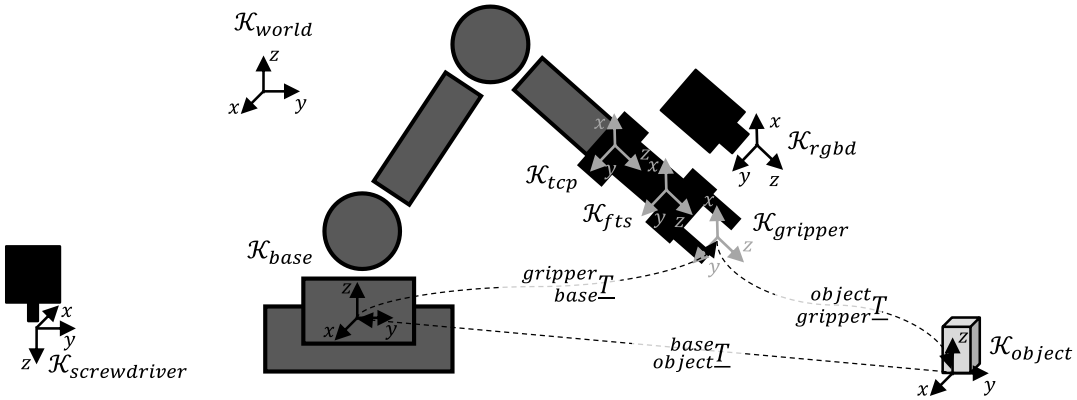


Abbildung 5-1: Verwendete Koordinatensysteme

Für die Werkzeuge wird das $screwdriver$-Frame und das $gripper$-Frame definiert. Auch wird für jedes Objekt ein eigenes Koordinatensystem festgelegt, welches zur Beschreibung der Objektmanipulation verwendet wird.

5.1.2 Dekomposition von Manipulationsprimitiven in eine Aktionsprimitivsequenz

Für die Erzeugung eines ausführbaren Anwenderprogramms, welches durch Aktionsprimitive beschrieben wird, muss das Manipulationsprimitiv, in Abhängigkeit des Kontextes von Werkzeug, der zu manipulierenden Komponente sowie dem nachfolgenden Manipulationsprimitiv zerlegt werden. Die allgemeinen Dekompositionsvorschriften erlauben eine flexible Generierung des Anwenderprogramms durch Aktionsprimitive. Es muss lediglich vorab eine Implementierung der parametrierbaren Aktionsprimitive erfolgen. Die Anzahl der einzelnen Aufrufe in der Aktionsprimitivsequenz erfolgt anhand von Entscheidungsregeln und wird in 5.1.3 diskutiert.

Verallgemeinert man die Dekomposition aller Manipulationsprimitive die prozessabhängig sind, also $\mathcal{MP}^C = \{\mathcal{MP}\}\backslash\{move, put\}$, lässt sich die allgemeine Sequenz (5.5) an Aktionsprimitiven bilden. Dabei bedeutet $[x]$ eine Ausführung von x ein- oder keinmal und $\{x\}$ eine mindestens ein- oder N-malige Ausführung des Aktionsprimitiv im Anwenderprogramm.

$$\mathcal{MP}^C(\forall\tau, \forall C) \rightarrow \; [getTool_i] \;\; [getObj_i] \;\; move \;\; \{processObj_1\} \;\; \{processObj_2\} \;\; \dots \;\; \{processObj_n\} \;\; move \;\; put \;\; [putTool_i]. \tag{5.5}$$

So können diese Manipulationsprimitive immer in eine Anrückbewegung, die eigentliche Prozessmanipulation und die abschließende Abrückbewegung, mit eventuell vorheriger Objektablage, zerlegt werden. Für die eigentlichen Prozessfunktionen $\{processObj_i\}$ wird ein rekursiver Aufruf zugelassen, sodass eine Unabhängigkeit, gegenüber kinematischen Einschränkungen der verwendeten Roboterkinematik, erreicht werden kann. Auf diese Eigenschaft soll später noch eingegangen werden. Optionale Aktionsprimitive wie $[getTool_i]$ oder $[getObj_i], [putTool_i]$ müssen in Abhängigkeit, der im nächsten Abschnitt definierten Entscheidungsregeln eingefügt werden.

Für die Dekomposition des $move$-Primitivs wird weiter eine Zerlegung nach (5.6) und für das put-Primitiv nach (5.7) definiert.

$$move(\forall\tau, \forall C) \rightarrow [roughPos] \quad [finePos]. \tag{5.6}$$

$$put(\forall\tau, \forall C) \rightarrow [putObj_i] \quad [move]. \tag{5.7}$$

Die einzelnen Aktionsprimitive entsprechen dabei den benötigten elementaren Funktionen, woraufhin folglich eingegangen wird. Das Aktionsprimitiv $roughPos$ übernimmt die Funktion der Grobpositionierung, welche rein positionsgeregelt für alle Koordinaten ausgeführt wird. Der Aufbau des Aktionsprimitiv ist in (5.8) beschrieben.

$$roughPos = \begin{cases} \mathcal{HM} = \begin{cases} \mathcal{TF} = / \\ \mathcal{D} = \begin{cases} control = (pos \quad pos \quad pos \quad pos \quad pos \quad pos) \\ d = (p_x \quad p_y \quad p_z \quad \alpha_x \quad \beta_y \quad \gamma_z) \end{cases} \end{cases} \\ \tau = \begin{cases} tool = / \\ cmd = / \end{cases} \\ \lambda = \underline{p}_g \end{cases}. \tag{5.8}$$

Der Term Grobpositionierung wird deswegen gewählt, da durch die Objekterkennung eine Positionsunsicherheit gegeben ist und die meisten Objektpositionen eine weitere Feinpositionierung, über ein bildbasiertes Regelungsverfahren, benötigen. Bei genau bekannter Objektposition, also Objektpositionen die nicht vom Bildverarbeitungssystem bereitgestellt werden, wie etwa Werkzeugpositionen oder Objektablagen, ist diese Bezeichnung irreführend. Das Aktionsprimitiv ist beendet, wenn die Zielpose $\underline{p}_g$ durch den Roboter erreicht wird.

Für die Feinpositionierung wird das Aktionsprimitiv $finePos$ (5.9) genutzt. Die Regelung erfolgt dabei für alle Koordinaten über einen bildbasierten visuellen Regler. Das Primitiv wird abgelöst, wenn das Zielfeature $\underline{\not{f}}_g$ des Objekts erreicht ist.

$$finePos = \begin{cases} \mathcal{HM} = \begin{cases} \mathcal{TF} = rgbd \\ \mathcal{D} = \begin{cases} control = (vsc \quad vsc \quad vsc \quad vsc \quad vsc \quad vsc) \\ d = (\not{f}_{1,u} \quad \not{f}_{1,v} \quad \not{f}_{2,u} \quad \not{f}_{2,v} \quad \not{f}_{3,u} \quad \not{f}_{3,v}) \end{cases} \end{cases} \\ \tau = \begin{cases} tool = / \\ cmd = / \end{cases} \\ \lambda = \underline{\not{f}}_g \end{cases}. \tag{5.9}$$

Für die Ausführung der prozessspezifischen Funktionen $\{processObj_1\}, \{processObj_2\}, \ldots, \{processObj_n\}$, können festgelegte Aktionsprimitivsequenzen definiert werden. Beispielsweise ist ein Schraubprozess einer Schraube definiert durch eine Aktionsprimitivsequenz, die sich aus der Kontaktherstellung des Schraubers mit der Schraube (5.10) sowie dem eigentlichen Schraubvorgang (5.11)-(5.12) zusammensetzt. Der Schraubvorgang erfolgt dabei immer in z-Koordinate im $screwdriver$-Frame. Das Werkzeugwechselaktionsprimitiv $getTool_i$ ist prinzipiell wie $roughPos$ aufgebaut, enthält jedoch zusätzlich das zugehörige $tool$ mit dem cmd-Befehl. Für das Ablegen von Objekten wird wiederum eine ähnliche Funktion wie für die Kontaktherstellung mit der Schraube verwendet. Die Regelung erfolgt jedoch, in Abhängigkeit des Objekts, Kraft-Momenten geregelt über die z-, α_x- und β_y-Koordinaten im $gripper$-Frame, sodass ein definiertes, kraftgeregeltes Ablegen des Objekts möglich ist.

$$processS_1 = \begin{cases} \mathcal{HM} = \begin{cases} \mathcal{TF} = screwdriver \\ \mathcal{D} = \begin{cases} control = (ftc \quad ftc \quad ftc \quad pos \quad pos \quad pos) \\ d = (f_x \quad f_y \quad f_z \quad / \quad / \quad /) \end{cases} \end{cases} \\ \tau = \begin{cases} tool = screwdriver \\ cmd = / \end{cases} \\ \lambda = f_{z,g} \end{cases}, \tag{5.10}$$

$$processS_2 = \begin{cases} \mathcal{HM} = \begin{cases} \mathcal{TF} = screwdriver \\ \mathcal{D} = \begin{cases} control = (pos \quad pos \quad ftc \quad pos \quad pos \quad pos) \\ d = (/ \quad / \quad f_z \quad / \quad / \quad /) \end{cases} \end{cases} \\ \tau = \begin{cases} tool = screwdriver \\ cmd = twistOn \end{cases} \\ \lambda = \theta_{SD} \end{cases}, \tag{5.11}$$

$$processS_3 = \begin{cases} \mathcal{HM} = \begin{cases} \mathcal{TF} = screwdriver \\ \mathcal{D} = \begin{cases} control = (pos \quad pos \quad pos \quad pos \quad pos \quad pos) \\ d = (/ \quad / \quad p_z \quad / \quad / \quad /) \end{cases} \end{cases} \\ \tau = \begin{cases} tool = screwdriver \\ cmd = twistOff \end{cases} \\ \lambda = p_{z,g} \end{cases}. \tag{5.12}$$

5.1.3 Entscheidungsregeln für Aktionsprimitivsequenzen

Zur Auflösung der einzelnen Funktionsaufrufe für das $\mathcal{AP}$ sollen weitergehend einfache Entscheidungsregeln $r(c_1, c_2, \dots, c_j) \rightarrow \{true, false\}$ mit den Bedingungen c_i eingeführt werden, die die allgemeinen Sequenzen (5.5)-(5.7) in die jeweilig benötigte Sequenz überführt. Für die Entscheidung, ob ein Aktionsprimitiv ein- oder keinmal in die Sequenz eingefügt wird, genügen folgende Regeln (5.13)-(5.18).

$$[getTool_i]: getTool_i \leftarrow (tool_i \neq tool_{i-1}) \vee (tool_i = \emptyset), \tag{5.13}$$

$$[putTool_i]: putTool_i \leftarrow (tool_{i+1} \neq tool_i) \vee (\mathcal{MP}_{i+1} = \emptyset), \tag{5.14}$$

$$[roughPos]: roughPos \leftarrow \left(\underline{p}_{g,i} \neq \underline{p}_{robot}\right), \tag{5.15}$$

$$[finePos]: finePos \leftarrow \left(\underline{p}_{g,i} = \underline{p}_{vp}\right), \tag{5.16}$$

$$\left[putObj_j\right]: putObj_j \leftarrow (C_{i+1} \neq C_i), \tag{5.17}$$

$$[getObj]: getObj \leftarrow (\mathcal{MP} \in assemblySequence). \tag{5.18}$$

Die Zulassung einer Rekursion wird für Prozessfunktionen erlaubt, sodass eine Implementierung des Aktionsprimitiv, unabhängig von der genutzten Roboterkinematik, ermöglicht wird. So wird ein Aktionsprimitiv wiederholt ausgeführt, wenn kinematische Randbedingungen während der Ausführung gemäß (5.19), nicht eingehalten werden können. Beispielsweise ist die Rekursion für eine Schraubbewegung eines Objekts nötig, wenn der auszuführende Drehwinkel außerhalb der Achsbeschränkung der Roboterkinematik liegt.

$$\{processObj_i\}: processObj_i \leftarrow \left(d_{act,max} \subset \lambda_i\right). \tag{5.19}$$

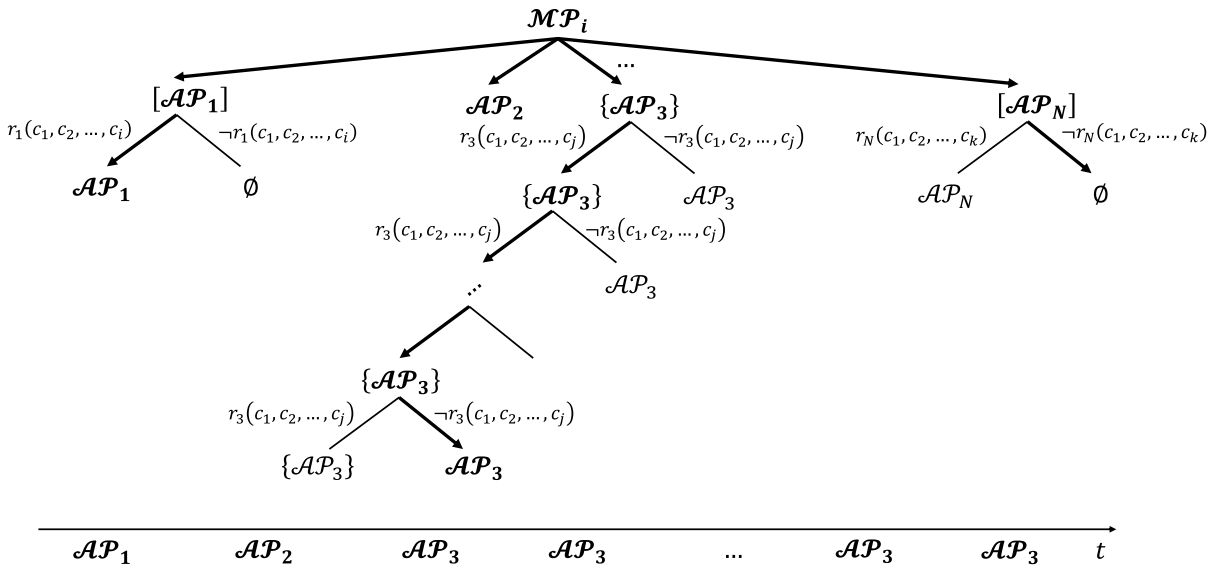


Abbildung 5-2: Ableitung von Aktionsprimitivsequenzen auf Basis von Entscheidungsregeln

Durch eine Rückwärtsbewegung und einen wiederholten Aufruf des Greifers mit der auszuführenden Bewegung, lassen sich auch folgende Aufgaben einfach und effizient lösen. Jedoch müssen diese Vorschriften vorab separiert betrachtet und bei der Implementierung berücksichtigt werden.

In Abbildung 5-2 ist die Generierung der Aktionsprimitive auf Basis der Entscheidungsregeln exemplarisch dargestellt. Nur bei Erfüllung der Bedingungen $r_1(c_1, c_2, \dots, c_i)$ für $[\mathcal{AP}_1]$ wird dieses ausgeführt. Für $\{\mathcal{AP}_3\}$ erfolgt nach Erfüllung $r_3(c_1, c_2, \dots, c_j)$ eine wiederholte Ausführung von $\mathcal{AP}_3$, bis $\neg r_3(c_1, c_2, \dots, c_j)$ gilt. Anschließend wird das nächste Aktionsprimitiv, welches aus der Sequenz (5.5)-(5.7) gegeben ist, weiterverarbeitet. Neben der Auflösung von Aktionsprimitiven aus einem Manipulationsprimitiv, können auch weitere Manipulationsprimitive, gemäß der Vorschrift (5.5)-(5.7), verwendet werden. Es sei angemerkt, dass die Entscheidungsregeln nach jedem Ausführungsschritt überprüft werden, da nicht alle Zustände, beispielsweise die Objektposition nach Ablage eines Objekts, vorhanden sind. So erfolgt eine sequentielle Generierung der auszuführenden Aktionsprimitive zur Laufzeit. Dies, sowie die Umsetzung des Konzepts, sollen im folgenden Unterkapitel diskutiert werden.

5.1.4 Laufzeitinterpretation

Die in den vorherigen Abschnitten besprochene Dekomposition und Prüfung der Entscheidungsregeln, zur Bestimmung der aufzurufenden Aktionsprimitive, erfolgt interpreterbasiert zur Laufzeit. Somit wird grundsätzlich eine flexible und fehlertolerante Ausführung der Manipulationsprimitive erlaubt, da bei Fehlschlag einer Aufgabe direkt ein alternativer Plan bestimmt und ausgeführt werden kann. Dies setzt jedoch die Bestimmung der relevanten Zustände (Roboter- und Objektzustände) sowie die Entwicklung eines neuen Planers voraus, was nicht weiter behandelt werden soll. Ein weiterer Vorteil ist darin zu sehen, dass vorab nicht für alle Transitionen ein Netz an Aktionsprimitiven erzeugt werden muss, sondern dies sequentiell, in Abhängigkeit der Zustände, geschieht. Der Aufbau ist in Abbildung 5-3 aufgeführt. Der Interpreter ist in einen Nicht-Echtzeit (NRT) und einen Echtzeit-Layer (RT) gegliedert.

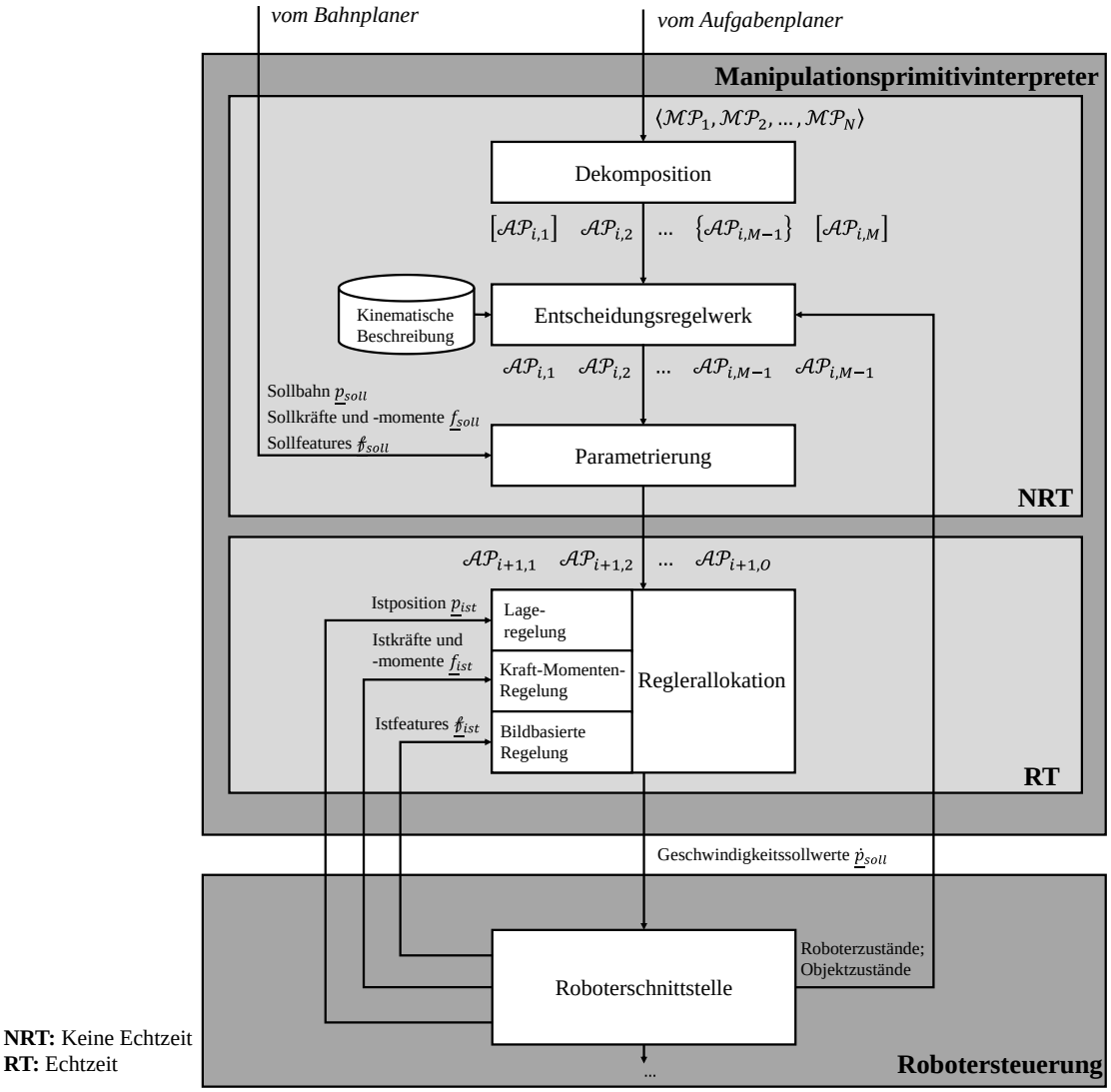


Abbildung 5-3: Architektur des Laufzeitinterpreters für Manipulationsprimitive

Im RT-Layer sind die verschiedenen Regler integriert, die als Ausgabewert eine kartesische Sollgeschwindigkeit an die Roboterschnittstelle weitergeben. Von dieser müssen auch die Roboterzustände (beispielsweise Greifer in Hand; Objekt gegriffen) und Objektzustände (beispielsweise Objektpose und zugehöriges Koordinatensystem) an das Entscheidungsregelwerk weitergegeben werden. Nach der Ausführung eines Aktionsprimitives werden die entsprechenden Zustände aktualisiert und für die Auswertung der Entscheidungsregeln verwendet.

Das Funktionsprinzip soll abschließend, anhand des Beispiels in Abbildung 5-4, näher beschrieben werden. Die Aufgabe besteht in der Demontage einer Schraube S aus einem Gewindeblock B. Die vom Aufgabenplaner erzeugte Manipulationsprimitivsequenz besteht genau aus einem Primitiv mit $twist(screwdriver, S)$. Auf Basis von (5.5) wird das Manipulationsprimitiv zerlegt. Da (5.13) erfüllt ist, muss initial die Beschaffung der Schraubeinheit durchgeführt werden. Aufgrund der Nichterfüllung von (5.18) erfolgt die Grobpositionierung und die Feinpositionierung, da sowohl (5.15) als auch (5.16) wahr ist. Der Schraubprozess erfolgt auf Basis der prozessspezifischen Aktionsprimitive (5.10)-(5.12).

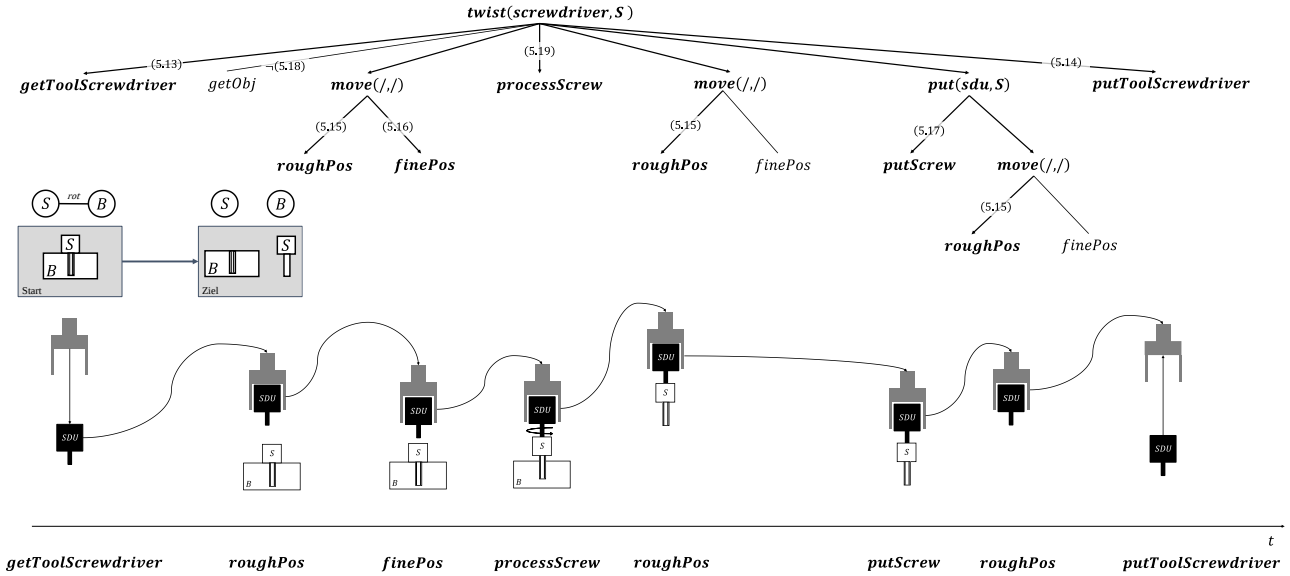


Abbildung 5-4: Beispiel der Laufzeitinterpretation für Manipulationsprimitive

Da der Ablageort fest definiert ist und dem Interpreter nicht vom Objekterkennungssystem bereitgestellt wird, muss für das nachfolgende $move$-Primitiv nur die Grobpositionierung ausgeführt werden. Das Manipulationsprimitiv wird durch die Ablage der Schraube sowie der Schraubeinheit beendet, da (5.17) und (5.14) erfüllt sind.

5.2 Analyse, experimenteller Entwurf und Validierung der Regelung

Es wird, gemäß dem Konzept der Aktionsprimitive, eine schaltende Regelung für die einzelnen Teilregler (Position-, Kraft-Momenten, Bildbasierte Regelung vergleiche 2.3.6) verwendet. Eingangs wird in 5.2.1 auf die verwendete kartesische Geschwindigkeitsschnittstelle eingegangen und deren Übertragungsverhalten untersucht. Der experimentelle Entwurf und die Validierung der exterozeptiven Regler schließen das Unterkapitel ab. Diese Untersuchungen sind mit in die Veröffentlichung [22] eingegangen.

5.2.1 Geschwindigkeitsschnittstelle und Analyse der propriozeptiven Regelung

Für die Kopplung des Interpreters, mit den einzelnen Teilregler an die Robotersteuerung, wird eine kartesische Geschwindigkeitsschnittstelle verwendet. Somit ist ein einheitlicher Steuereingriff der einzelnen Teilregler möglich, wodurch eine einfache Kopplung des vorgestellten Gesamtkonzepts an industrielle Robotersteuerungen gegeben ist. Unterlagert kann die von der Robotersteuerung bereitgestellte propriozeptive Regelung mit allen Steuerungsfunktionalitäten (kinematische Transformationen, Gütekriterien bei Singularitäten, Geschwindigkeitsplanung, usw.) standardmäßig verwendet werden. Es wird in dieser Arbeit auf die Roboterschnittstelle [351] zurückgegriffen. Der genaue Systemaufbau ist in 6.1 und 6.2 beschrieben. In Abbildung 5-5 ist die Kopplung der einzelnen Teilregler mit der unterlagerten Roboterregelung dargestellt. Im Interpreter sind gemäß Abbildung 5-3 die einzelnen Teilregler integriert. Für die visuelle Regelung wird ein IBVS-Regler $\underline{R}_{IBVS}$, für die Kraft-Momenten-Regelung ein Admittanzregler $\underline{R}_A$ und für die Position ein Proportionalregler $\underline{R}_p$, gemäß dem Prinzip der Aktionsprimitive in eine schaltende Regelungsstruktur eingebettet.

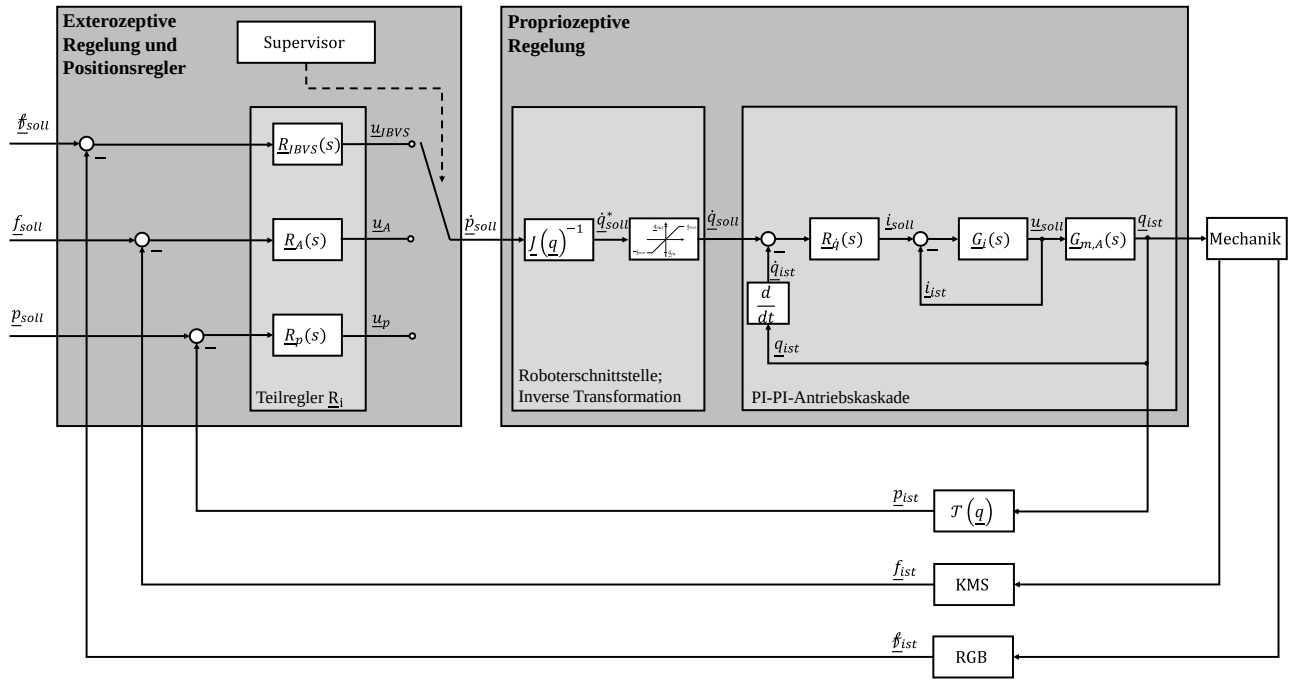


Abbildung 5-5: Blockschaltbild der Gesamtsystemregelung

Der Umschaltvorgang wird durch den Supervisor koordiniert, welcher die einzelnen Teilregler, in Abhängigkeit des $control$-Modus des Aktionsprimitives, aktiviert. Zur Vermeidung von Sollwertsprüngen beim Umschalten zwischen den Teilreglern, werden diese, über eine Interpolation mittels eines ruckreduzierten Polynom 5. Ordnung ineinander überführt. Die Interpolation ist in den einzelnen Reglern integriert. Die Achsgeschwindigkeiten werden dann über die kinematische Transformation bestimmt. Die propriozeptive Regelung besteht aus einer kaskadierten PI-Geschwindigkeits- und PI-Stromregelung mit antriebsseitiger Messung der Gelenkposition.

Des Weiteren soll das Übersetzungsverhalten der Geschwindigkeit und das dynamische Übertragungsverhalten der propriozeptiven Regelung, der verwendeten Roboterkinematik, untersucht werden. Dies erlaubt eine einfache Plausibilitätsprüfung, der im Interpreter erzeugten Stellgrößen, in Abhängigkeit der verwendeten Kinematik.

Analyse des Geschwindigkeitsübertragungsverhaltens. Die Analyse des statischen Geschwindigkeitsübertragungsverhaltens kann durch die Singulärwerte der Jacobi-Matrix $\underline{J}\left(\underline{q}\right)$ der direkten Kinematik $\underline{p} = \underline{f}_{\mathcal{DK}}\left(\underline{q}\right)$ erfolgen [352]. Dies wird in der Literatur auch häufig unter dem Begriff der Manipulierbarkeit gefasst. Für die verwendete serielle Sechsachs-Kinematik (6R) kann die Jacobi-Matrix allgemein mit (5.20) angegeben werden.

$$\underline{J}\left(\underline{q}\right) = \begin{pmatrix} \frac{\partial f_{\mathcal{DK},1}\left(\underline{q}\right)}{\partial q_1} & \frac{\partial f_{\mathcal{DK},1}\left(\underline{q}\right)}{\partial q_2} & \cdots & \frac{\partial f_{\mathcal{DK},1}\left(\underline{q}\right)}{\partial q_6} \\ \frac{\partial f_{\mathcal{DK},2}\left(\underline{q}\right)}{\partial q_1} & \frac{\partial f_{\mathcal{DK},2}\left(\underline{q}\right)}{\partial q_2} & \cdots & \frac{\partial f_{\mathcal{DK},2}\left(\underline{q}\right)}{\partial q_6} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{\mathcal{DK},6}\left(\underline{q}\right)}{\partial q_1} & \frac{\partial f_{\mathcal{DK},6}\left(\underline{q}\right)}{\partial q_2} & \cdots & \frac{\partial f_{\mathcal{DK},6}\left(\underline{q}\right)}{\partial q_6} \end{pmatrix}. \tag{5.20}$$

Wegen dem differentiellen Zusammenhang zwischen kartesischer Position zu Gelenkposition gilt

$$\underline{\dot{p}} = \underline{J}\left(\underline{q}\right) \cdot \underline{\dot{q}}. \tag{5.21}$$

Nun wird für die Jacobi-Matrix eine Singulärwertzerlegung (SVD) eingeführt, mit

$$\underline{J}\left(\underline{q}\right) = \underline{U}\left(\underline{q}\right) \cdot \underline{\Sigma}\left(\underline{q}\right) \cdot \underline{V}^T\left(\underline{q}\right). \tag{5.22}$$

Die Singulärwerte $\underline{\Sigma} = \mathrm{diag}(\sigma_1 \quad \sigma_2 \quad \dots \quad \sigma_6)$ beschreiben eine Streckung oder Stauchung von kartesischem zu Gelenkwinkelraum, während $\underline{U}$ und $\underline{V}$ unitäre Matrizen darstellen und für $\underline{U}, \underline{V} \in \mathbb{R}^{6\times6}$ eine Rotation oder Spiegelung repräsentieren. Prinzipiell ist nun von Interesse, wie hoch die maximale kartesische Geschwindigkeit $\underline{\dot{p}}$, die von den Teilreglern vorgegeben wird, in Abhängigkeit der Gelenkkonfiguration sein darf, sodass die maximal zur Verfügung stehende Antriebsgeschwindigkeit $\underline{\dot{q}}_{max,A} = (\dot{q}_{max,A} \quad \dot{q}_{max,A} \quad \dot{q}_{max,A} \quad \dot{q}_{max,A} \quad \dot{q}_{max,A} \quad \dot{q}_{max,A})^T$ nicht überschritten wird. Für eine konservative Abschätzung der maximal zulässigen kartesischen Geschwindigkeit und der Einfachheit wegen soll gelten, dass $\underline{U} = \underline{V} = \underline{I}$. Somit wird jeder Koordinate im kartesischen Raum ihre zugehörige Koordinate im Gelenkraum über die inverse Matrix der Singulärwerte zugeordnet. Es gilt dann

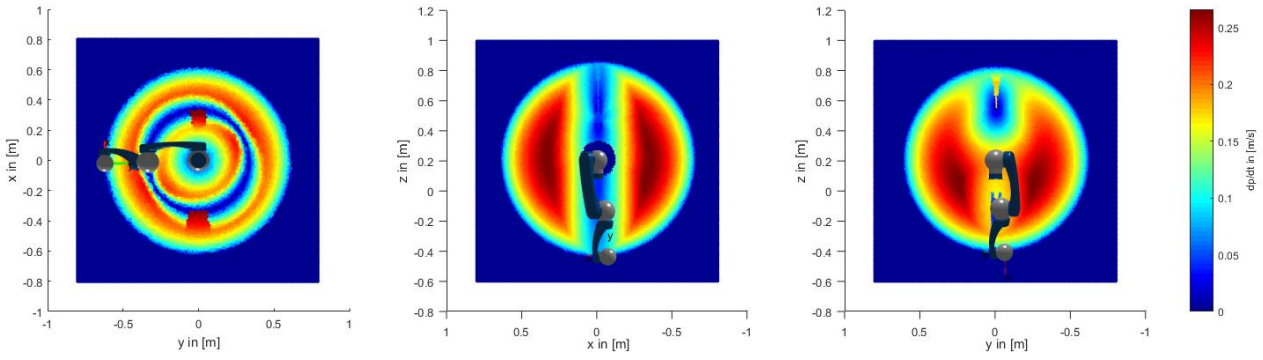
$$\underline{\dot{q}} = \left(\underline{U} \cdot \underline{\Sigma} \cdot \underline{V}^T\right)^{-1} \cdot \underline{\dot{p}} = \mathrm{diag}(\sigma_1^{-1} \quad \sigma_2^{-1} \quad \dots \quad \sigma_6^{-1}) \cdot \underline{\dot{p}}, \text{mit } \underline{U} = \underline{V} = \underline{I}. \tag{5.23}$$

Entscheidend für das Übertragungsverhalten ist nun der minimale Singulärwert $\sigma_{min} = \min\left(\underline{\Sigma}\right)$, da dann eine maximale Verstärkung zwischen kartesischem und Gelenkraum vorliegt, also

$$\dot{q}_{max} = \sigma_{min}^{-1}\left(\underline{q}\right) \cdot \left|\underline{\dot{p}}\right|. \tag{5.24}$$

Somit lässt sich die maximale kartesische Sollgeschwindigkeit mit (5.25) abschätzen. Abbildung 5-6 zeigt den örtlichen Verlauf der translatorischen, kartesischen Maximalgeschwindigkeiten für die verwendete Roboterkinematik in den drei Hauptebenen auf.

$$\left|\underline{\dot{p}}\right|_{max} \ll \dot{q}_{max,A} \cdot \sigma_{min}\left(\underline{q}\right). \tag{5.25}$$



(a) (b) (c)

Abbildung 5-6: Analyse der translatorischen kartesischen Maximalgeschwindigkeit mit der Abschätzungsmethode aus Gleichung (5.25); Verlauf der Maximalgeschwindigkeiten x-y-Ebene (a); x-z-Ebene (b); y-z-Ebene (c)

Mit örtlichem Abstand zu Singularitäten und nicht erreichbaren Konfigurationen kann ein relativ isotropes Verhalten im Arbeitsraum beobachtet werden. Die Abschätzung eignet sich besonders zur Plausibilitätsprüfung der kommandierten Sollwerte der einzelnen Teilregler, wodurch eine Überschreitung der maximal verfügbaren Antriebsgeschwindigkeiten vermieden werden kann.

Analyse der propriozeptiven Regelung. Neben dem statischen Geschwindigkeitsübertragungsverhalten soll das dynamische Übertragungsverhalten der propriozeptiven Regelung der Antriebe, bei geschlossenem Lageregler, untersucht werden. Dies rechtfertigt die Ausführungszeiten des Gesamtsystems in 6.3. Derzeit kann die Regelung nur mit einer maximalen Frequenz von 50Hz arbeiten, da dies die Begrenzung der verwendeten ROS Version darstellt. Für eine einfache Bestimmung relevanter Kenndaten wird ein lineares Systemverhalten angenommen. Das dieses nur lokale Gültigkeit besitzen kann, ist allein wegen dem Zusammenhang (5.21) einsehbar.

Für gewöhnlich kann eine lagegeregelte, kaskadierte Servoachse durch eine lineare Differentialgleichung 2. Ordnung, sowie einer eventuellen Totzeit beschrieben werden, vergleiche beispielsweise [291]. Jedoch hat eine Differentialgleichung 1. Ordnung mit Totzeit, gemäß (5.26), eine bessere Approximation des Streckenverhaltens ergeben, was sich, wie noch gezeigt wird, auf die dominante Pollage rückführen lässt, die für ein träges Streckenverhalten verantwortlich ist.

$$K \cdot p_{i,soll}(t - T_{tot}) = T \cdot \dot{p}_{i,ist}(t) + p_{i,ist}(t). \tag{5.26}$$

Zur Bestimmung der Parameter wird das System in unterschiedlichen Posen, gemäß Abbildung 5-7 (a)-(c), mit dem Eingangssignal (5.27) im Weltkoordinatensystem erregt.

$$p_{i,soll}(t) = A \cdot \sin(\omega(t) \cdot t)\,, mit\ A = 0.1\text{m}, \omega = [0.1, \dots, 70]\text{rad/s}. \tag{5.27}$$

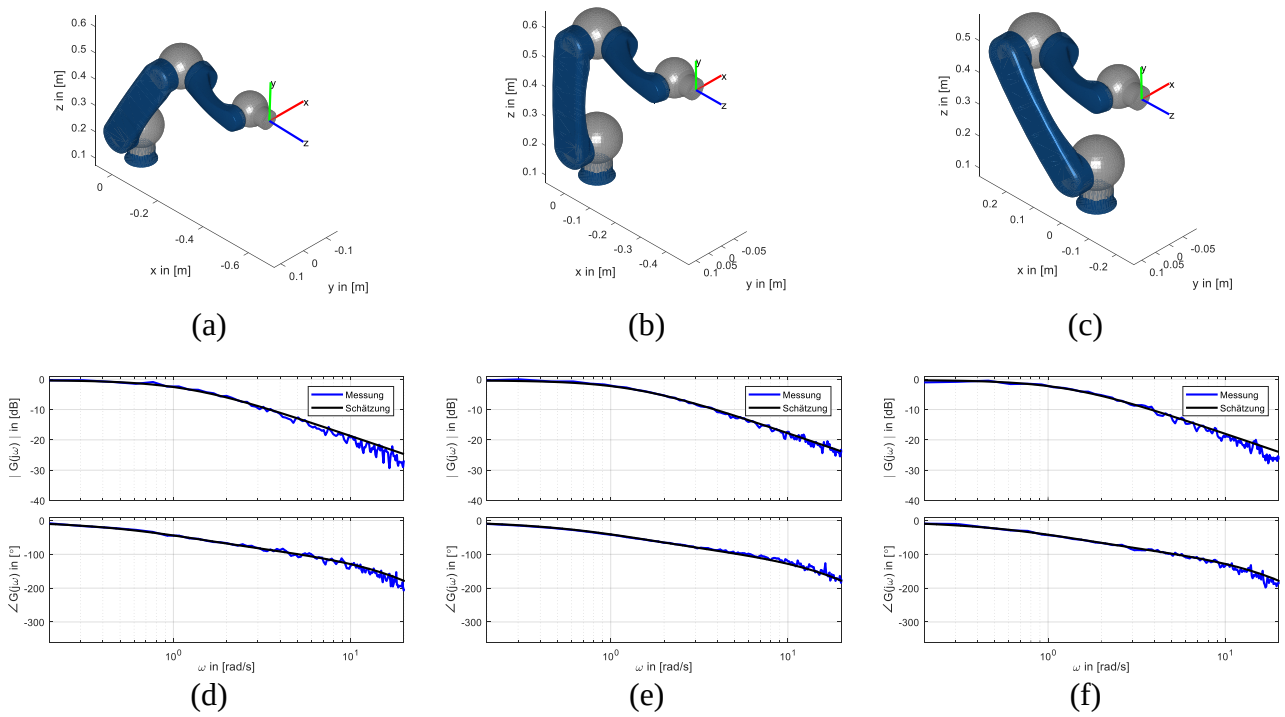


Abbildung 5-7: Dynamik der propriozeptiven Regelung. Identifikationsposen (a)-(c); Bode-Diagramm der Schätzung und Messung für die x- (d), y- (e) und z-Koordinate (f) in Pose 1 (a)

Die Identifikation wird separat für alle drei translatorischen Koordinaten, also $i = x, y, z$ ausgeführt. Die Bestimmung der Parameter T_{tot}, K und T erfolgt über einen Least-Squares Parameterschätzer [40]. Das System hat prinzipiell eine sehr große Totzeit mit $T_{tot} = 0.08\text{s}$, welche sich auf das verwendete Softwareframework und die niedrige Reglerfrequenz rückführen lässt. Die Dynamik des PT1-Gliedes ist ebenso äußerst gering. Es ergibt sich eine Zeitkonstante mit $T = 0.82\text{s}$ gemittelt über alle Identifikationsposen und Messungen ($K = 0.91\text{m}$). Die Varianz der Zeitkonstante liegt bei 0.007s, was zeigt, dass das Übertragungsverhalten für die unterschiedlichen Posen ziemlich identisch ist. Die maximale Bandbreite (-3dB) für die Messungen liegt bei 1.35rad/s. In Abbildung 5-7 (d)-(f) ist ein Vergleich der Modellschätzungen für Gleichung (5.26) mit den gemessenen Frequenzgängen für die drei translatorischen Koordinaten, exemplarisch für Pose 1, abgebildet. Es zeigt sich eine gute Übereinstimmung der Messung mit der Schätzung. Der normalisierte quadratische Fehler (NRMSE) beträgt über alle Schätzungen 90%.

5.2.2 Entwurf und Validierung der exterozeptiven Regler

Im folgenden Unterkapitel sollen die verwendeten exterozeptiven Regler und deren Parametrierung, hier Admittanzregler für die Kraft-Momenten-Regelung und IBVS-Regler für die bildbasierte Regelung, dargestellt werden. Als gemeinsamer Steuereingriff der exterozeptiven Regler wird, wie bereits in 5.2.1 erläutert, vergleiche auch Abbildung 5-5, eine kartesische Geschwindigkeitsschnittstelle eingesetzt. Folgend soll die Auslegung der einzelnen Teilregler diskutiert und dargestellt werden. Der Reglerentwurf erfolgt dabei experimentell nach empirischen Einstellregeln, da eine theoretische Auslegung nicht Fokus dieser Arbeit ist.

Admittanzregelung. Für die Kraft-Momenten-Regelung wird ein Admittanzregler verwendet. Aufgrund der einfachen Parametrierbarkeit und der guten Eignung in der Regelung bei hohen Kontaktsteifigkeiten und -dämpfungen, vergleiche [353], ist dieser Ansatz für die gegebene Problemstellung als besonders geeignet zu bewerten.

Aus dem Übertragungsverhalten der Admittanz, vergleiche 2.3.6 (Abschnitt Kraft-Momenten-Regelung), und der Wahl der Steuergröße als „nachgiebige“ Geschwindigkeit $\underline{\dot{p}_c}$, kann das Regelungsgesetz mit

$$\underline{\dot{p}_c}(t) = \underline{C}^{-1}\left(\underline{f_{soll}}(t) - \underline{f_{ist}}(t) - \underline{M}\,\underline{\ddot{p}_c}(t) - \underline{D}\,\underline{\dot{p}_c}(t)\right). \tag{5.28}$$

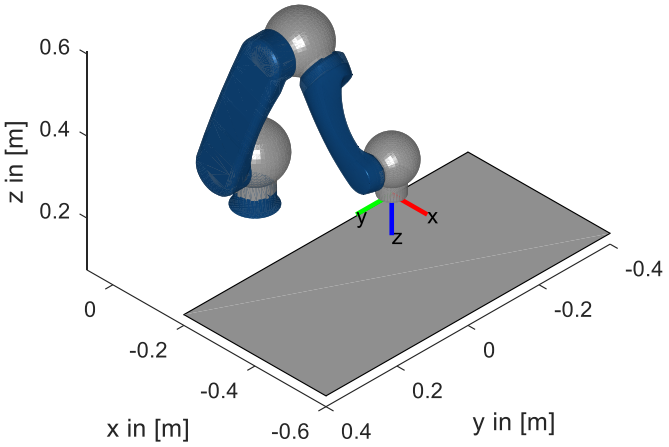
angegeben werden. Die Parametrierung des Reglers erfolgt dann durch die Festlegung der Elemente in den Diagonalmatrizen der Masse $\underline{M}$, der Dämpfung $\underline{D}$ sowie der Steifigkeit $\underline{C}$, mit den Einträgen $\text{diag}(m_1 \quad m_2 \quad \dots \quad m_6)$, $\text{diag}(d_1 \quad d_2 \quad \dots \quad d_6)$ und $\text{diag}(c_1 \quad c_2 \quad \dots \quad c_6)$.

Für die Validierung des Reglers wird eine Kontaktkraftherstellung mit einer harten Umgebung verwendet, da dies für (De-)Montageaufgaben eine der am häufigsten benötigten Funktionalitäten darstellt. Die Messung der Kräfte und Momente erfolgt über einen Kraft-Momenten-Sensor am TCP des Roboters. Für die genauen technischen Daten sei auf 6.1 verwiesen. Die Auswertung erfolgt exemplarisch für die z-Koordinate im Weltsystem, mit einer gewählten Roboterkonfiguration gemäß Abbildung 5-8 (a).

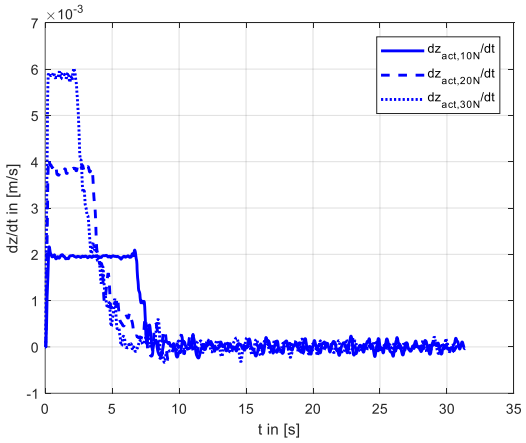
Die Reglerparametrierung erfolgt per klassischen Einstellregeln, da für einen analytischen Entwurf ein exaktes Kontaktmodell für Roboter und Umgebung vorhanden sein müsste. Die

Parameter werden mit $m_3 = 5\text{kg} \cdot \text{s}$, $d_3 = 250\text{kg}$ und $c_3 = 500\text{kg} \cdot \text{s}^{-1}$ gewählt. Die Sensorwerte werden alle 100Hz aktualisiert und wegen starkem Sensorrauschen zusätzlich mit einem FIR-Tiefpassfilter 5. Ordnung (Wahl der Filterkoeffizienten nach Moving Average) geglättet. In Abbildung 5-8 (b), (c) sind die Ergebnisse für drei unterschiedliche Kontaktkraft-herstellungsaufgaben mit $f_{z,soll,i} = \{10\text{N}, 30\text{N}, 40\text{N}\}$ dargestellt.

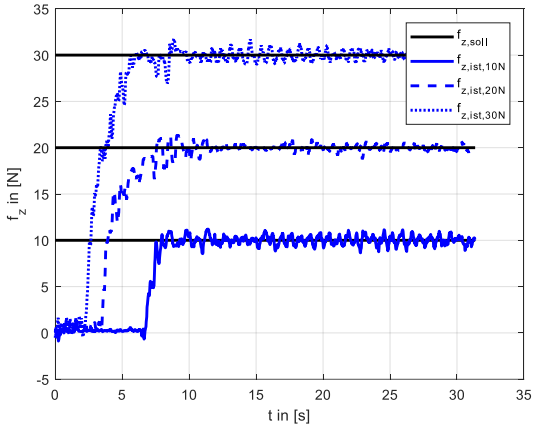
Es zeigt sich, dass eine stabile Ausführung der Aufgabe für gegebene Konfiguration gewährleistet werden kann. Jedoch kann aufgrund der begrenzten Sensordatenfrequenz mit Filterung und der Reglerfrequenz der unterlagerten PI-PI-Kaskadenregelung, nur eine geringe Dynamik im Admittanzregler vorgegeben werden. Die Einstellung der Sollkräfte beträgt circa 8s (Entfernung $\Delta z = 0.2\text{m}$ zu Objekt), was für den produktiven Einsatz des Systems deutlich zu gering ist.



(a)



(b)



(c)

Abbildung 5-8: Kontaktkraftherstellung mit Admittanzregler in z-Koordinate im Weltsystem für unterschiedliche Sollkräfte (a) für die gewählte Gelenkkonfiguration $\underline{q} = (0 \quad -0.366 \quad 1.756 \quad 0 \quad 1.019 \quad 0)^T\text{rad}$: Steuergröße (kartesische Geschwindigkeit) des Admittanzreglers (b); Soll- und Istkräfte für die Kontaktkraftherstellung (c)

IBVS-Regelung. Als visuelle Regelung wird ein klassischer IBVS-Regler eingesetzt, da dieser vor allem Vorteile in einer hohen Konvergenzgeschwindigkeit, bei geringen Regelabweichungen, mit sich bringt. Da prinzipiell von einer geringen Ungenauigkeit in der Posenbestimmung, durch die Objektlokalisation, ausgegangen werden kann (kleiner 0.005m), ist der IBVS-Regler sehr gut für die Feinpositionierung geeignet. Ein weiterer Vorteil besteht in der direkten Regelung in den (u, v)-Bildkoordinaten des optischen Sensors, wodurch eine rechenaufwändige Posenbestimmung des Zielobjekts in jedem Regeltakt entfällt. Dies führt zwangsläufig zu einer geringeren Totzeit in der Berechnung des Istwerts und damit einer höher erzielbareren Reglerdynamik. Die Herleitung des Reglers orientiert sich an die Darstellung [322].

Die Bestimmung der Sollgeschwindigkeiten im RGB-Kamerasystem $\underline{\dot{p}}^{rgb} \in \mathbb{R}^6$ erfolgt über die Inversion der Feature-Jakobi-Matrix $\underline{J_{ff}} \in \mathbb{R}^{2\times 6}$ in Abhängigkeit der Geschwindigkeit der Features im Bildraum $\underline{\dot{f}} \in \mathbb{R}^2$, vergleiche auch 2.3.6 (Abschnitt Bildbasierte visuelle Regelung), also

$$\underline{\dot{p}}^{rgb} = \underline{J_{ff}}^{-1} \cdot \underline{\dot{f}}. \tag{5.29}$$

Zur Inversion von $\underline{J_{ff}}$ und damit der vollständigen Regelung aller sechs kartesischen Freiheitsgrade müssen nun mindestens drei Features vorliegen. Im vorliegenden Fall werden immer vier Features betrachtet, also $\underline{f} = \left(\underline{f}_1 \quad \underline{f}_2 \quad \underline{f}_3 \quad \underline{f}_4\right)^T$, sodass eine zusätzliche Redundanz vorhanden ist, da auch bei Verlust eines Features noch $rank\left(\underline{J_{ff}}\right) = 3$ gilt. Somit erfolgt die Bildung von $\underline{J_{ff}}$ durch

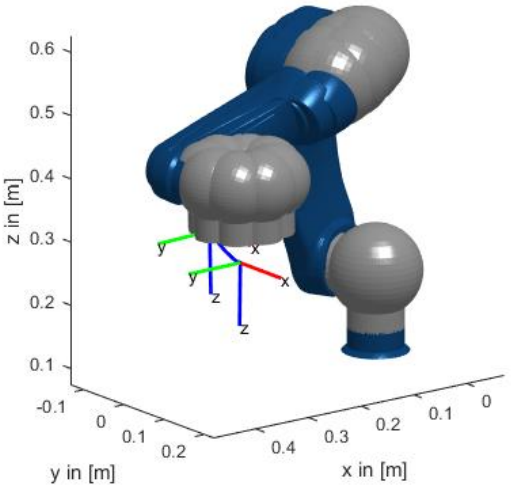
$$\underline{J_{ff}} = \left(\underline{J_{f,1}} \quad \underline{J_{f,2}} \quad \underline{J_{f,3}} \quad \underline{J_{f,4}}\right)^T. \tag{5.30}$$

Das Reglergesetz kann dann als klassischer P-Regler formuliert werden, mit

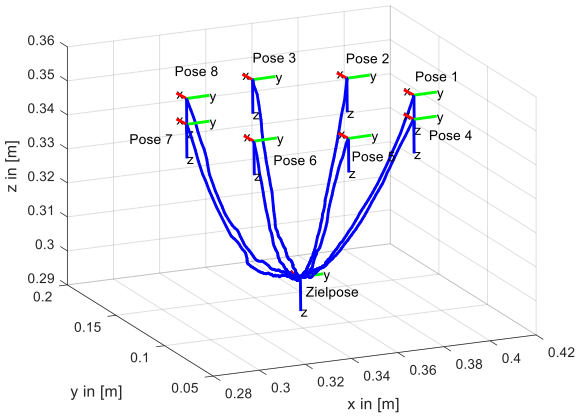
$$\underline{\dot{p}}^{rgb} = k_{ibvs} \cdot \left(\underline{J_{ff}}^T \underline{J_{ff}}\right)^{-1} \underline{J_{ff}}^T \cdot \left(\underline{\dot{f}}_{soll} - \underline{\dot{f}}_{ist}\right) = k_{ibvs} \cdot \underline{J_{ff}}^{+} \cdot \underline{e_f}, \tag{5.31}$$

wobei k_{ibvs} die Proportionalitätskonstante des Reglers darstellt. Die Parametrierung kann wiederum über einfache Einstellregeln erfolgen.

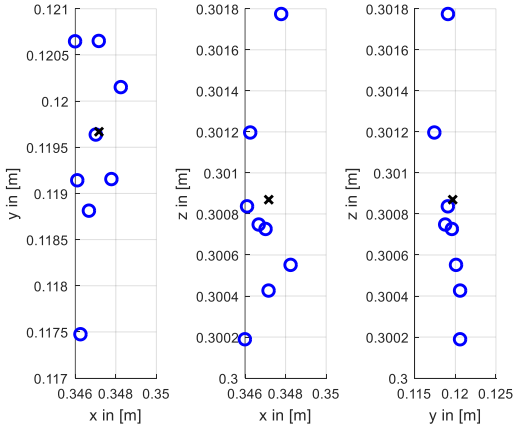
Für die Genauigkeitsuntersuchung des IBVS-Reglers zur Feinpositionierung werden acht kreisförmig angeordnete Startposen, mit einem Durchmesser von 0.05m in der x-y-Ebene des Weltsystems, gemäß Abbildung 5-9 (a), (b), gewählt. Der z-Abstand zur Zielpose beträgt ebenso 0.05m. Als Features werden Kreismarker, mit einem Durchmesser von 0.025m, verwendet. Die Reglerfrequenz wird maßgeblich durch die Featureextraktion begrenzt und liegt bei circa 20Hz. Der Regler ist mit $k_{ibvs} = 0.125$ parametriert. Das verwendete Hardwaresetup ist in 6.1 beschrieben. In Abbildung 5-9 sind die Ergebnisse für die obigen experimentellen Rahmenparameter dargestellt. Es zeigt sich, dass für alle Posen ein gutes Konvergenzverhalten (Abbildung 5-9 (b)), bei gleichzeitig akzeptabler Genauigkeit für die Feinpositionierung, erzielt werden kann. Der arithmetische Mittelwert der kartesischen Positioniergenauigkeit liegt bei 0.0012m sowie einer Positioniergenauigkeit in der Orientierung von 0.0036rad. Die Streuung der Positioniergenauigkeit ist in Abbildung 5-9 (c) abgebildet. Beispielhaft ist in Abbildung 5-9 (d) für Pose 4 der Verlauf der vier Features in der Bildebene des Sensors aufgeführt. Der zugehörige Regelfehler im Bildraum ist in Abbildung 5-9 (e) dargelegt.



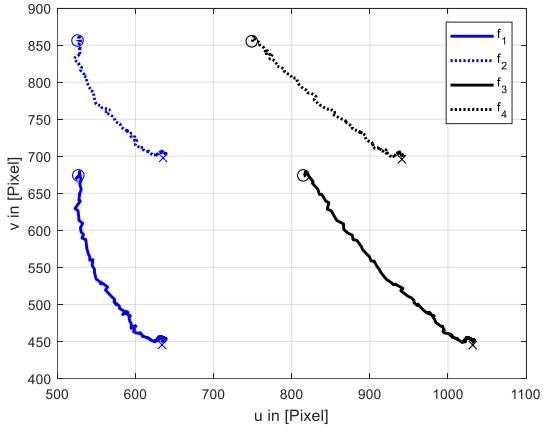
(a)



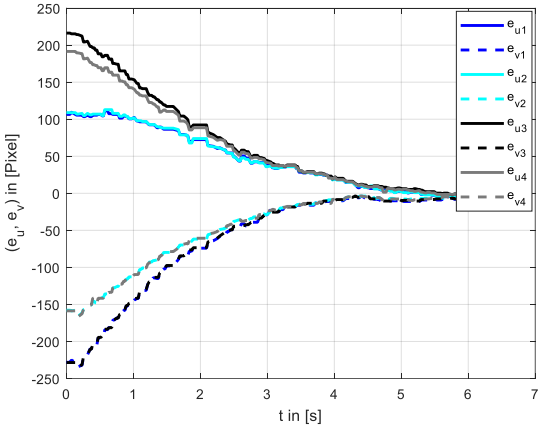
(b)



(c)



(d)



(e)

Abbildung 5-9: Ergebnisse des parametrierten IBVS-Reglers: Start- und Zielposen des Roboters (a); Kartesische Bahnen für acht unterschiedliche Startwerte (b); zugehörige Positioniergenauigkeit (c); Bahn der Features in Bildebene für Pose 4 (d); Zugehöriger Regelfehler der Features für Pose 4 (e)

5.3 Zusammenfassung und Diskussion

In diesem Kapitel wurde eine Methode entwickelt, die die automatische Generierung des Anwenderprogramms mittels der Planungsergebnisse aus Kapitel 4 erlaubt. Somit kann für Instandhaltungsaufgaben eine Automatisierung, von der Planung bis zur Ausführung, durch das Robotersystem erreicht werden. Folgend sollen die einzelnen Beiträge zusammenfassend diskutiert werden.

Anwenderprogrammgenerierung. Für die Umsetzung der Information aus der Planung in ein ausführbares Roboterprogramm wurde das bekannte Konzept der Aktionsprimitive verwendet. Zur automatischen Erzeugung werden die geplanten Manipulationsprimitive über allgemeingültige Dekompositionsvorschriften in eine Aktionsprimitivsequenz, mit optionalen und rekursiven Aktionsprimitiven, überführt. Anhand von Entscheidungsregeln wird dann zur Ausführungszeit anhand Roboter- und Objektzuständen entschieden, ob das jeweilige Aktionsprimitiv ausgeführt werden muss. Jedoch muss vorab eine Implementierung der benötigten Aktionsprimitive stattfinden. Die Konfiguration erfolgt dann über die Ausgabe der Planung. Die Schaffung eines interpreterbasierten Ausführungskonzepts ermöglicht prinzipiell eine flexible Umplanung zur Laufzeit. So kann bei Fehlschlag eines Aktionsprimitiv flexibel eine alternative Aktion ausgeführt werden. Zukünftig sollten auch Planungsfunktionalitäten entwickelt werden, die alternative Manipulationsprimitivsequenzen bestimmen und somit die Flexibilität der Ausführungsebene erst nutzbar machen.

Geschwindigkeitsschnittstelle. Zur Steuerung des Robotersystems erfolgt eine Sollwertvorgabe ausschließlich über kartesische Geschwindigkeiten (prinzipiell auch Positionen möglich). Somit ist eine einfache Anbindung an Robotersteuerungen, mit der Möglichkeit zur externen kartesischen Geschwindigkeitsvorgabe, möglich. Durch die konservative Abschätzung der maximal zulässigen Geschwindigkeit, kann eine einfache Plausibilitätsprüfung der Sollwerte im Interpreter stattfinden. Das Verfahren ist automatisiert und kann für beliebige Kinematiken, nach Eingabe der Denavit-Hartenberg-Parameter, eine Lösung finden.

Regelung. Für die Regelung der einzelnen Aktionen wurden, entsprechend dem Stand der Technik, bekannte, exterozeptive Regelgesetze ausgewählt und in eine schaltende Regelung, gemäß dem Prinzip der Aktionsprimitive, integriert. Es zeigt sich experimentell, dass die gewählten Ansätze für die Praxis ausreichend sind. Jedoch kann aufgrund der begrenzten Bandbreite der propriozeptiven Regelung nur eine geringe Dynamik in den exterozeptiven Teilreglern vorgegeben werden. Zur Steigerung der Dynamik des Gesamtsystems sollte die propriozeptive Regelung auf eine industrielle Steuerung integriert werden.

6 Versuchsumgebung, Experimente und Validierung des Gesamtkonzepts

„Je weniger man versucht hat, desto mehr hält man für möglich.“
Wolfgang Mocker

Folgendes Kapitel beschreibt einführend die aufgebaute Versuchsumgebung sowie die verwendeten Systemkomponenten der Roboterplattform. Weitergehend erfolgt die Darstellung der softwaretechnischen Umsetzung der Steuerungsarchitektur in ein ganzheitliches Systemdesign, welches die Integration der einzelnen Frameworks und Algorithmen aus Kapitel 4 und Kapitel 5 erlaubt. Anhand von typischen Demontage-, Montage-, Wartungs- und Instandsetzungsoperationen, erfolgt die experimentelle Validierung und Diskussion des Gesamtsystems. Abschließend werden die Leistungsfähigkeit und die technischen Grenzen für den Einsatz in der Praxis, der in dieser Arbeit entwickelten Methoden, reflektiert.

6.1 Roboterplattform und Hardwarekomponenten

Für die experimentelle Validierung der Methoden wurde eine Roboterplattform aufgebaut sowie zur Integration der einzelnen Planungs-, Steuer- und Regelungsverfahren eine Softwarearchitektur entwickelt, die folglich beschrieben werden soll. In Abbildung 6-1 ist die Roboterplattform mit den zugehörigen Sensoren und Werkzeugen dargestellt. Für die Roboterplattform wurden folgenden Komponenten verwendet.

Robotermanipulator. Als Kinematik wird der Powerball Lightweight Arm LWA 4P der Firma Schunk eingesetzt. Die Sechsachs-Kinematik ist speziell für mobile Anwendungen konzipiert, wobei die Leistungselektronik direkt in den einzelnen Modulen verbaut ist. Als Kommunikations-schnittstelle wird das CANopen (CiA DS402: IEC61800-7-201) Profil verwendet. Die elektrische Aktorik besteht aus bürstenlosen Servomotoren mit Permanentmagnet-Bremse und Pseudo-Absolutwertgebern, mit einer Wiederholgenauigkeit von ± 0.15mm, sowie einer maximalen Geschwindigkeit von $\pm 72°/s$ bei Nennlast. Der Arbeitsbereich beträgt $\pm 155.5°$ bei Achse 3, sowie $\pm 170°$ bei den restlichen Achsen. Alle technischen Angaben sind dem Datenblatt [354] entnommen.

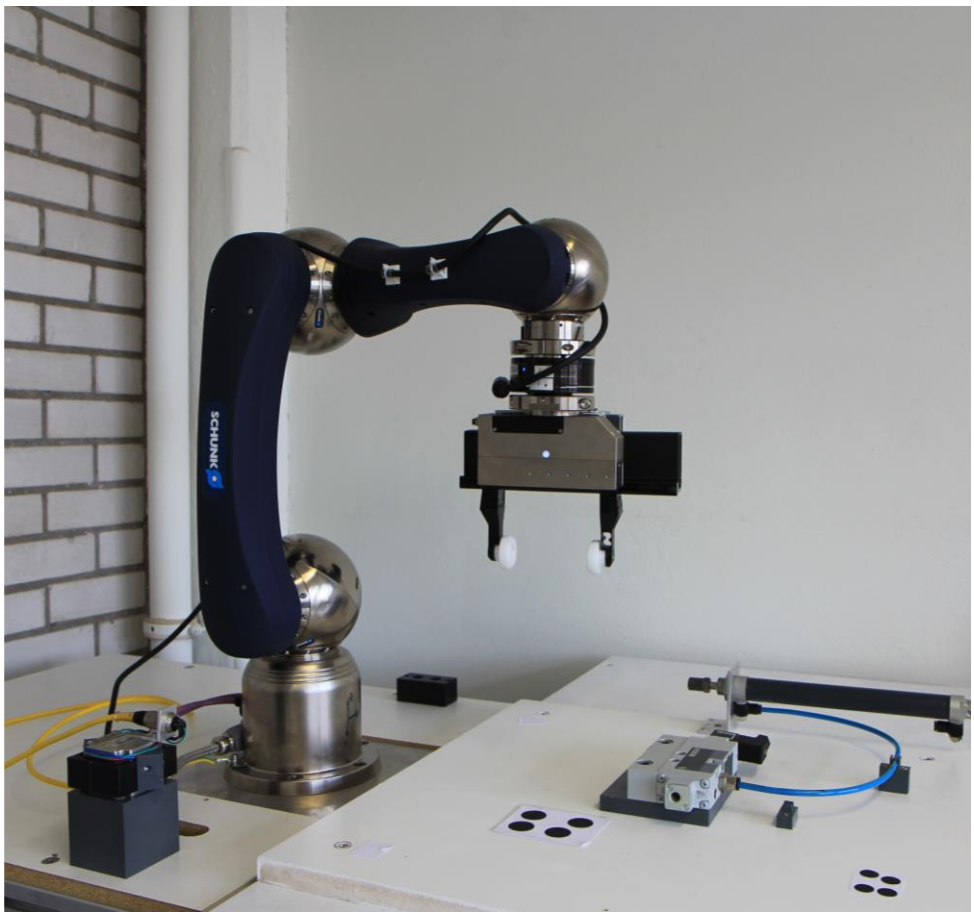
Sensoren und Werkzeuge. Als Sensoren stehen sowohl eine RGBD-Kamera in Hand-Auge Konfiguration als auch ein 6D Kraft-Momenten-Sensor zur Verfügung. Als Werkzeuge werden ein Zwei-Finger-Parallelgreifer und eine Schraubeinheit verwendet.

Kinect v2. Als RGBD-Kamera wird die Kinect v2 von Microsoft eingesetzt. Diese stellt einen RGB-Sensor, mit einer Auflösung von 1920×1080 Pixel, sowie einen Time-of-Fligth Tiefensensor, basierend auf Infrarot, mit einer Auflösung von 512×424 Pixel, zur Verfügung. Der Tiefensensor arbeitet im Bereich von 0.5m bis 4.5m. Die maximale Frame-Rate bei voller Auflösung beträgt 30Hz, die Datenübertragung erfolgt über USB 3.0. Eine genaue Evaluierung mit Genauigkeits-untersuchungen des Sensors ist in [355] zu finden.

FTM 75. Als Kraft-Momenten-Sensor wird das FTM 75 Modul der Firma Schunk verwendet. Dieses erlaubt das Messen von Kräften im Messbereich $\pm$300N und Momenten von $\pm$15Nm über alle sechs Raumrichtungen. Die minimale Messauflösung für die Kraftmessung beträgt dabei 0.3N; für die Momente 0.01Nm. Die Anbindung an die Steuerung erfolgt über das CANopen Protokoll im 100Hz Takt. Bezüglich weiterer technischer Angaben, siehe auch [356].

WSG 50. Der WSG 50 ist ein elektrischer 2-Finger-Parallelgreifer mit einer Greifkraft von bis zu 80N der Firma Schunk. Der Hubweg pro Greiffinger beträgt 55mm. Die Anbindung des Greifers erfolgt über CANopen an die Steuerung, vergleiche auch [357].

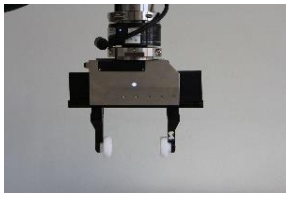
Schraubeinheit. Die verwendete Schraubeinheit besteht aus einem Schrittmotor, der über ein maximales Drehmoment von 0.9Nm verfügt. Die Drehzahl kann stufenlos im Bereich $\pm 20 s^{-1}$, über eine Pulsweitenmodulation durch eine Arduino gesteuerte Motortreiberkarte, vorgegeben werden.



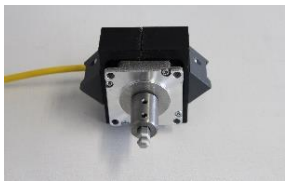
(a)

(b)

(c)



(d)



(e)

Abbildung 6-1: Demonstratoraufbau des Autonomen Instandhaltungsroboter (a): Verwendete Roboterkinematik Powerball Lightweight Arm LWA 4P der Firma Schunk; Als Sensoren wird eine Microsoft Kinect v2 (b) und der 6D-Kraft-Momenten-Sensor Schunk FTM 75 (c) eingesetzt; Als Werkzeuge wird das Greifsystem der Zwei-Finger-Parallelgreifer Schunk WSG 50 (d), sowie eine Schraubeinheit eingesetzt (e)

6.2 Umsetzung Steuerungsarchitektur

Das Design der Steuerungsarchitektur ist in drei Layern gegliedert, siehe Abbildung 6-2. Layer 1 bildet die eigentlichen Funktionalitäten der Steuerungsarchitektur ab, die in den beiden vorherigen Kapiteln dargestellt sind. Die Architektur wurde bereits in [14] veröffentlicht.

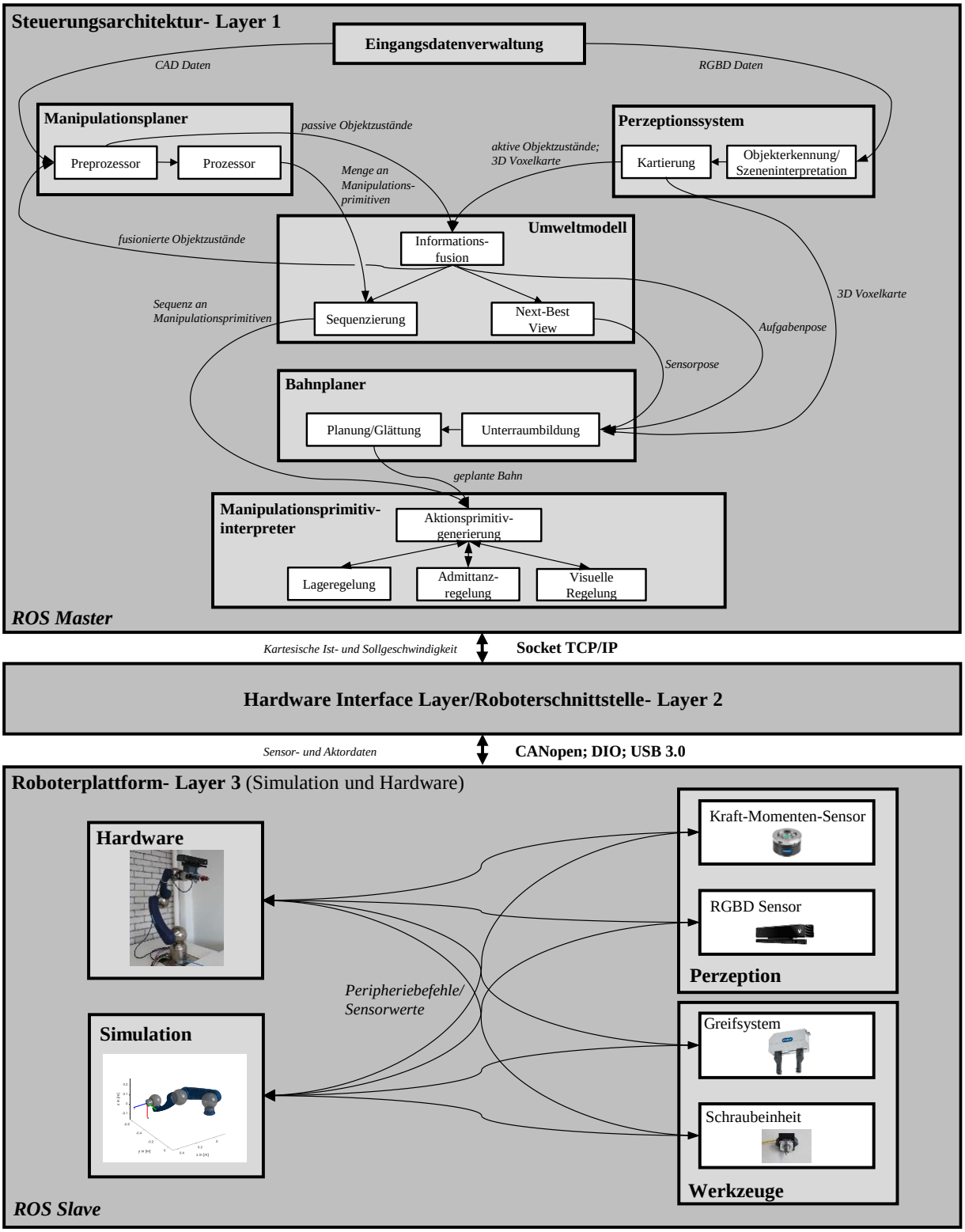


Abbildung 6-2: Struktur der Steuerungsarchitektur mit Hauptfunktionalitäten

Die Steuerungsarchitektur verfügt als Ausgabe über eine kartesische, sowie eine achsspezifische Schnittstelle für Sollgeschwindigkeiten und -positionen. Somit kann die Steuerungsarchitektur einfach und schnell auf unterschiedlichen Roboterplattformen Verwendung finden, indem extern Sollwerte vorgegeben werden. Als Softwareframework wird ROS [151] Indigo, aufgrund seiner Offenheit gegenüber vielfältigen Programmiersprachen, sowie der Vielzahl an bereits vorhandenen Funktionen und Treibern, verwendet. Die einzelnen Softwarefunktionalitäten sind in Matlab (Anbindung an ROS über die Robotics System Toolbox) und C/C++ realisiert. Die Abstraktion und Kommunikation mit der Zielhardware erfolgt auf Layer 2. Hierzu wird auf die CANopen Implementierung [351] zurückgegriffen, die für die Care-O-Bot Familie entwickelt ist und somit die Anbindung des LWA 4P sowie des FTM 75 und WSG 50 erlaubt.

In Layer 3 sind verschiedene hardwarespezifische Funktionen realisiert, die etwa eine Vorverarbeitung der Sensorwerte oder Ansteuerungsfunktionen für die Werkzeuge umsetzen. Als Steuerungsrechner wird ein Standard-PC mit Intel Core i7-4790K, 4000MHz mit 16GB DDR3 RAM auf einem Linux Ubuntu 14.04 LTS Betriebssystem verwendet. Für die CANopen Kommunikation zwischen Layer 2 und der Peripherie wird ein Peak System PCAN-USB-Adapter optoentkoppelt eingesetzt.

6.3 Experimentelle Validierung

In folgendem Abschnitt soll das Gesamtkonzept, anhand praxisnaher Beispiele, validiert werden. Zum einen werden zwei typische De- und Montageaufgaben mit einer „Stiftbaugruppe“ sowie einer „Plattenbaugruppe“ vorgestellt. Anschließend wird die Wartungsaufgabe „Kühlschmierstoff“ sowie eine Instandsetzungsaufgabe anhand einer „Ventilbaugruppe“ betrachtet. Die einzelnen Beispiele sind in Abbildung 6-3 aufgezeigt. Kern der Aufgabe ist jeweils die De- und Montage der jeweiligen Bauteile. In 6.3.1 wird die Ausführung sowie das zugehörige automatisch generierte Anwenderprogramm für diese vier Applikationen detailliert vorgestellt. 6.3.2 analysiert die Beispiele zusammenfassend. Teilergebnisse wurden schon in [22] vorgestellt.

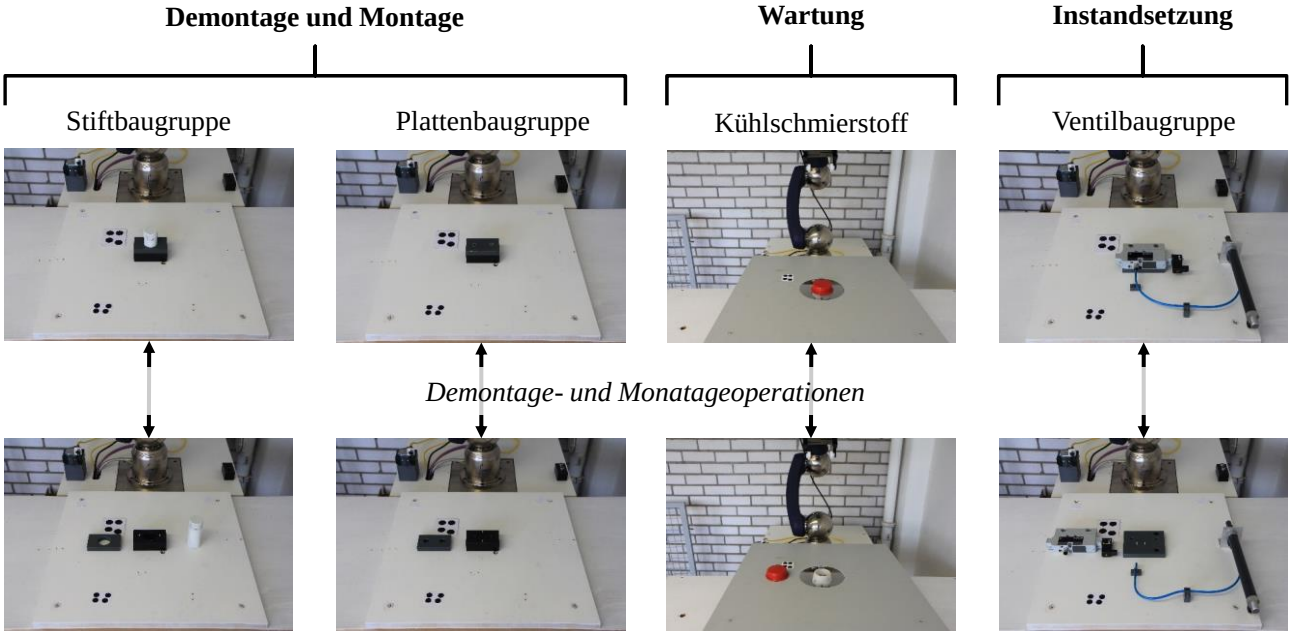


Abbildung 6-3: Beispielanwendungen zur experimentellen Validierung des Gesamtsystems

Für das Anwenderprogramm wird, aus Gründen der Übersichtlichkeit, immer nur der Demontagevorgang dargestellt. Die Positionssollwerte für die Erreichung der einzelnen Positionsziele $\underline{p}_{soll}$ werden direkt vom Bahnplaner generiert. Dies geschieht, unter Beachtung der vom Manipulationsplaner bereitgestellten Demontagerichtungen $\underline{d}$. Die Position für den Werkzeugwechsler ist fest vorgegeben. Die Sollwerte $\underline{f}_{soll}$, der Kräfte und Momente müssen vorab einmalig experimentell festgelegt werden, wobei die Richtungen wieder aus dem Planer kommen. Als Zielfeatures $\underline{\mathbf{f}}_{soll}$ für das Visual-Servoing werden derzeit zusätzliche, externe Marker verwendet. Videos der Experimente finden sich unter [358], [359], [360] und [361].

6.3.1 Beispielanwendungen

Folgend wird die Ausführung anhand vier typischer Anwendungen validiert. Unsicherheiten werden nicht mehr explizit betrachtet, da dies bereits in 4.3 und 4.4 erfolgreich validiert wurde.

Stiftbaugruppe. Die Stiftbaugruppe besteht aus den drei Bauteilen Stift, Platte und Grundplatte. Die Demontage umfasst das Entfernen des Stifts sowie der Platte. Die geplante Manipulationsprimitiv- mit der zugehörigen Aktionsprimitivsequenz, ist in Abbildung 6-4 dargestellt. Die Ausführung ist in Abbildung 6-5 aufgeführt. Nach der Stiftdemontage (Abbildung 6-5 (a)-(c)) erfolgt die Objektablage (Abbildung 6-5 (d), (e)).

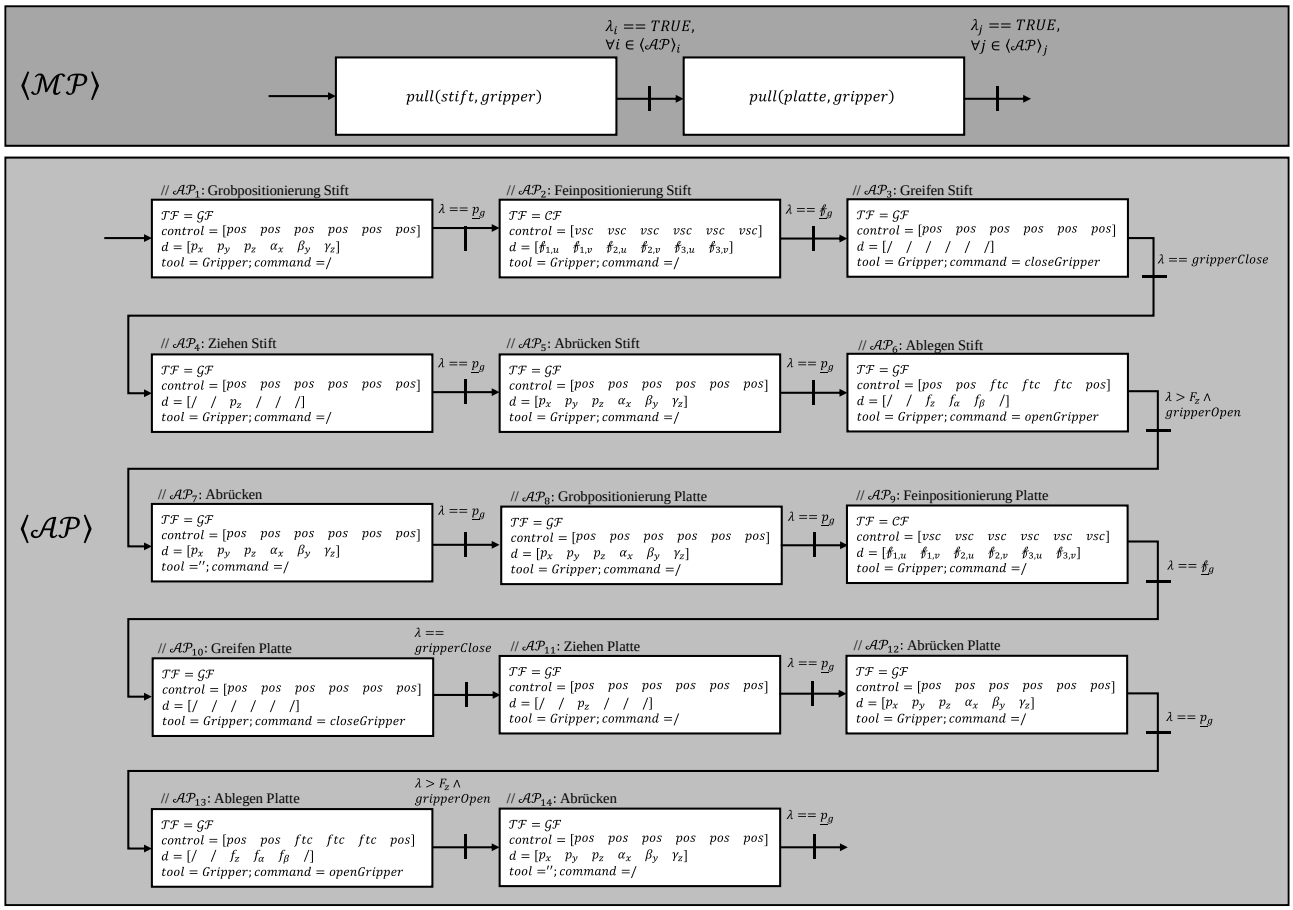
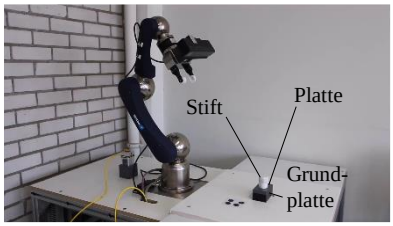
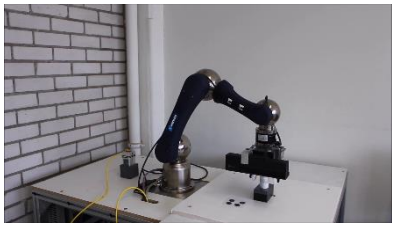


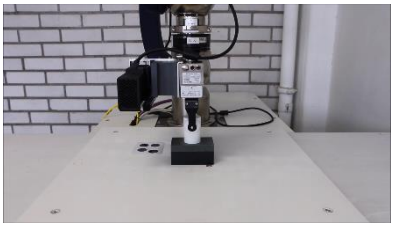
Abbildung 6-4: Stiftbaugruppe: Erzeugter Plan in Manipulationsprimitiven und zugehöriges Anwenderprogramm in Aktionsprimitiven (nur Demontage)



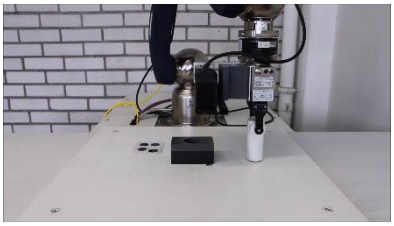
(a): *t= 0s* ($\mathcal{AP}_1$)



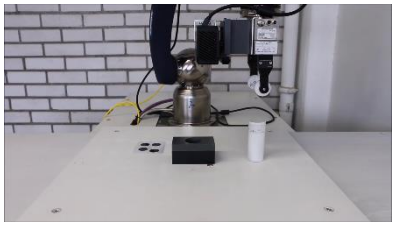
(b): *t= 24s* ($\mathcal{AP}_2$)



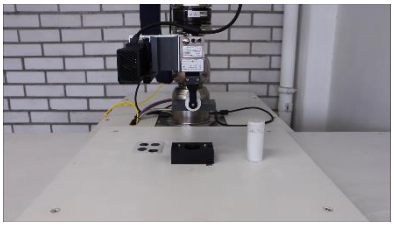
(c): *t= 28s* ($\mathcal{AP}_4$)



(d): *t= 38s* ($\mathcal{AP}_6$)

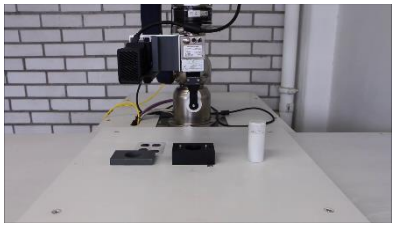


(e): *t= 46s* ($\mathcal{AP}_7$)



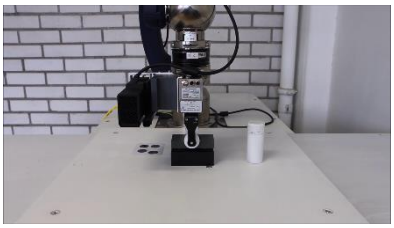
(f): *t= 68s* ($\mathcal{AP}_{12}$)

(g): *t= 82s* ($\mathcal{AP}_{13}$)

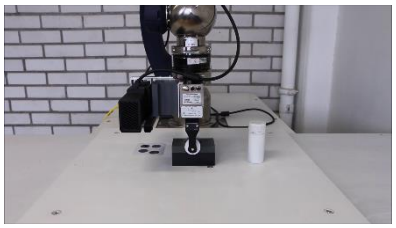


(h): *t= 92s* ($\mathcal{AP}_{14}$)

(i): *t= 104s*

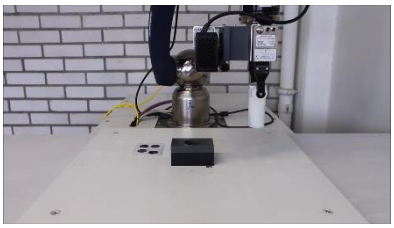


(j): *t= 115s*

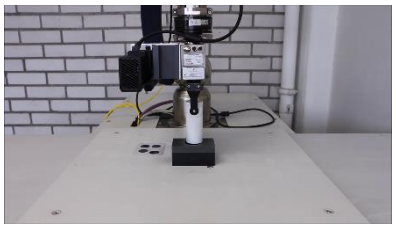


(k): *t= 121s*

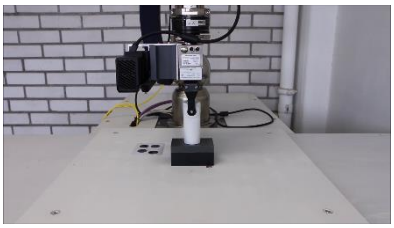
(l): *t= 131s*



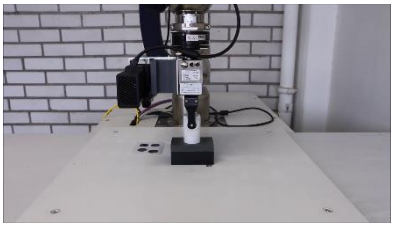
(m): *t= 138s*



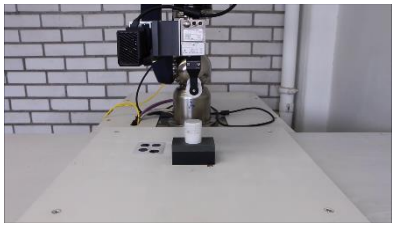
(n): *t= 153s*



(o): *t= 178s*



(p): *t=200 s*



(q): *t= 217s*

(r): *t= 259s*

Abbildung 6-5: Ausführung der (De-)Montageaufgabe „Stiftbaugruppe“

Hierzu wird der Admittanzregler verwendet, wobei nach Herstellung der Ablagekraft der Greifer geöffnet wird. Derselbe Vorgang wird für die Platte ausgeführt. Ab Abbildung 6-5 (h) erfolgt die Montage der beiden Bauteile. Zum Fügen beider Objekte wird wieder der Admittanzregler eingesetzt.

Plattenbaugruppe. Für die Plattenbaugruppe müssen sowohl zwei Schrauben als auch die Platte von der Grundplatte, demontiert werden. Der zugehörige steuerungstechnische Ablauf ist in Abbildung 6-6 und die Ausführung in Abbildung 6-7 beschrieben. Beginnend generiert der Interpreter die Aktion zur Aufnahme des Schraubmoduls, welches durch die Positionsregelung ausgeführt wird (Abbildung 6-7 (a), (b)). Nach der Grobpositionierung erfolgt die Feinpositionierung über den IBVS-Regler (Abbildung 6-7 (c)).

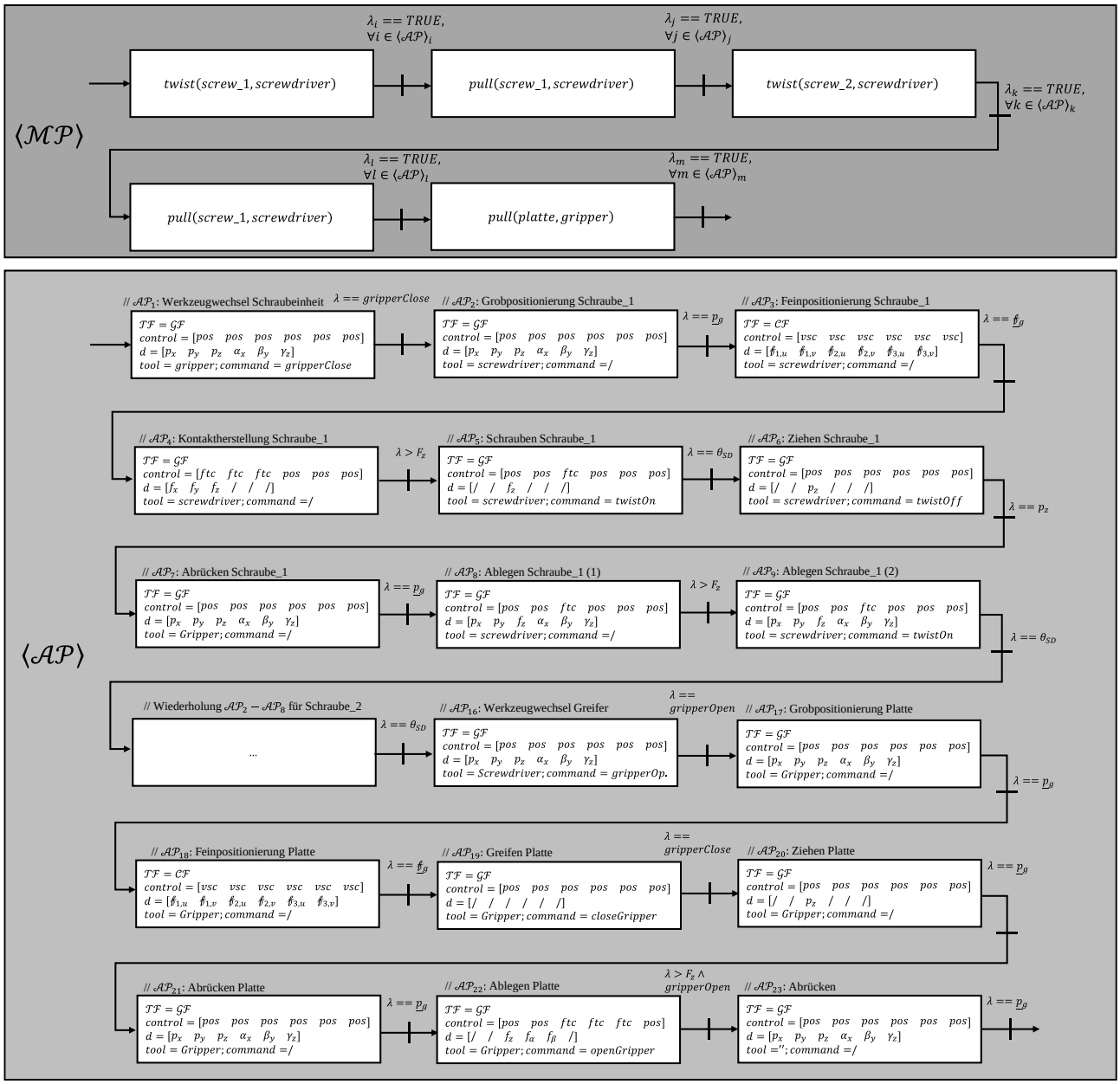
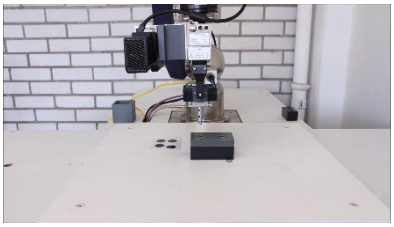


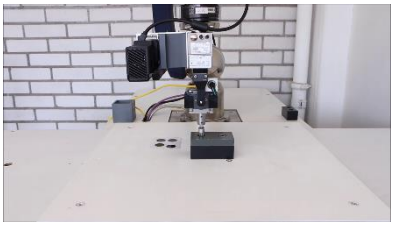
Abbildung 6-6: Plattenbaugruppe: Erzeugter Plan in Manipulationsprimitiven und zugehöriges Anwenderprogramm in Aktionsprimitiven (nur Demontage)

(a): *t= 0s* ($\mathcal{AP}_1$)

(b): *t= 26s* ($\mathcal{AP}_1$)



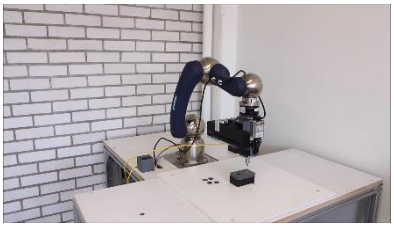
(c): *t= 42s* ($\mathcal{AP}_3$)



(d): *t= 64s* ($\mathcal{AP}_5$)

(e): *t= 72s* ($\mathcal{AP}_7$)

(f): *t= 87s* ($\mathcal{AP}_9$)



(g): *t= 110s* ($\mathcal{AP}_{11}$)

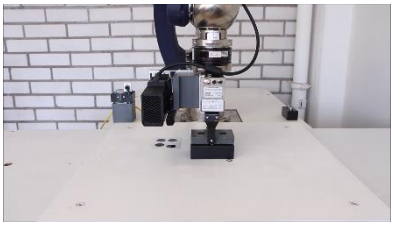
(h): *t= 140s* ($\mathcal{AP}_{15}$)

(i): *t= 194s* ($\mathcal{AP}_{17}$)

(j): *t= 232s* ($\mathcal{AP}_{22}$)

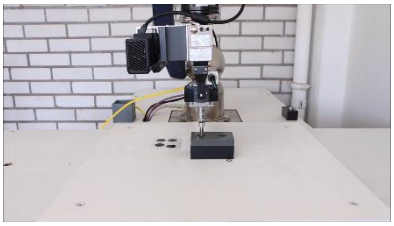
(k): *t= 260s* ($\mathcal{AP}_{23}$)

(l): *t= 273s*

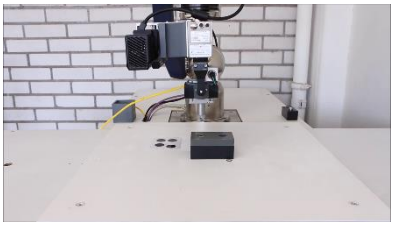


(m): *t= 286s*

(n): *t= 345s*



(o): *t= 356s*



(p): *t= 379s*

(q): *t= 420s*

(r): *t= 460s*

Abbildung 6-7: Ausführung der (De-)Montageaufgabe „Plattenbaugruppe“

Über den Admittanzregler wird die Einstellung einer vorab definierten Kontaktkraft erreicht und weitergehend der eigentliche Schraubprozess gestartet. Die Beendigung des Schraubprozess wird über eine Winkeltransition der Schraubeinheit ausgelöst. Die Schraube kann folglich abgezogen werden. Zur Ablage der Schraube ist eine definierte Halterung mit Gewinde angebracht. Für den Einschraubprozess wird wieder eine Kontaktkraft eingestellt und der Schraubvorgang erneut durch die Winkeltransition beendet. Der Vorgang ist in Abbildung 6-7 (d)-(f) aufgeführt. Für die zweite Schraube wiederholt sich der Ablauf. Die Abnahme der Platte wird nach Feinpositionierung durch den IBVS-Regler rein positionsbasiert ausgeführt (Abbildung 6-7 (i)-(k)). Ab Abbildung 6-7 (l) erfolgt wiederum der Montagevorgang.

Beispiel Wartungsaufgabe. Eine im produktionstechnischen Umfeld häufig zu tätigende Wartungsaufgabe, stellt das Nachfüllen von Kühlschmierstoff an Werkzeugmaschinen dar. Zur Lösung der Aufgabe muss der Roboter den verschraubten Kühlschmierstoffdeckel von dem Nachfüllstutzen entfernen. Der zugehörige Plan sowie das Anwenderprogramm, sind in Abbildung 6-8, und die Ausführung in Abbildung 6-9, dargestellt. Die der Grobpositionierung folgende Feinpositionierung definiert die Greifposition des Deckels (Abbildung 6-9 (a)-(c)). Es folgt die Drehung des Deckels mit translatorischer z-Überlagerung über eine reine Positionsregelung. Die Demontage des Deckels ist mit Erreichen der Zielposition, welche der Manipulationsplaner auf Basis des CAD-Modells bereitstellt, beendet (Abbildung 6-9 (d)-(h)). Es erfolgt das Abrücken und die Objektablage (Abbildung 6-9 (i)-(k)). Die Montage des Deckels ist ab Abbildung 6-9 (l) zu sehen.

Die eigentliche Servicefunktionalität, das Nachfüllen des Kühlschmierstoffs, wird nicht behandelt. Zur vollständigen, autonomen Lösung von Wartungsaufgaben, müssten, neben der Automatisierung des De- und des Montagevorgangs, zusätzlich Methoden entwickelt werden, die es erlauben Servicefunktionen allgemeingültig aus der zur Verfügung stehenden Information zu erzeugen.

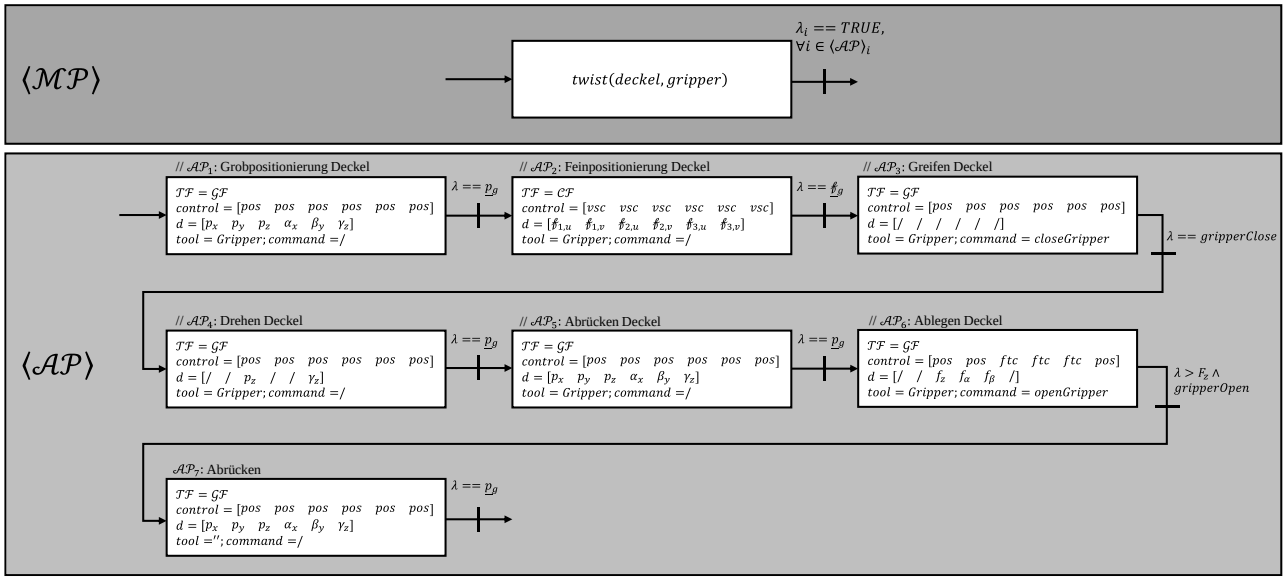
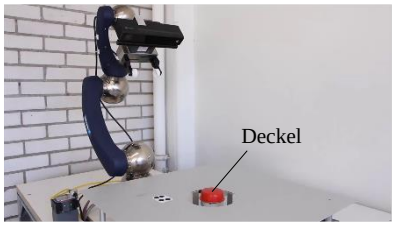


Abbildung 6-8: Kühlschmierstoff: Erzeugter Plan in Manipulationsprimitiven und zugehöriges Anwenderprogramm in Aktionsprimitiven (nur Demontage)



(a): *t= 0s* ($\mathcal{AP}_1$)

(b): *t= 18s* ($\mathcal{AP}_2$)

(c): *t= 28s* ($\mathcal{AP}_3$)

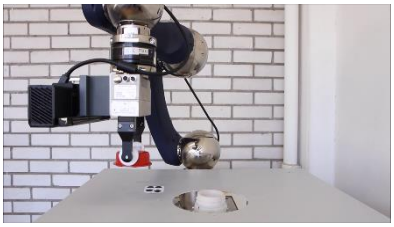
(d): *t= 40s* ($\mathcal{AP}_4$)

(e): *t= 69s* ($\mathcal{AP}_4$)

(f): *t= 105s* ($\mathcal{AP}_4$)

(g): *t= 210s* ($\mathcal{AP}_4$)

(h): *t= 230s* ($\mathcal{AP}_5$)

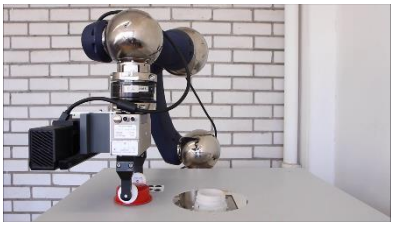


(i): *t= 236s* ($\mathcal{AP}_5$)

(j): *t= 240s* ($\mathcal{AP}_6$)

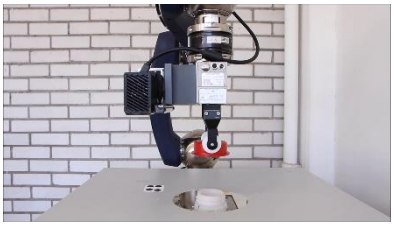
(k): *t= 256s* ($\mathcal{AP}_7$)

(l): *t= 261s*



(m): *t= 265s*

(n) *t= 267s*



(o): *t= 273s*

(p): *t= 277s*

(q): *t= 475s*

(r): *t= 504s*

Abbildung 6-9: Ausführung der Wartungsaufgabe „Kühlschmierstoff“

Beispiel Instandsetzungsaufgabe. Für das Anwendungsbeispiel der Instandsetzungsaufgabe muss ein Ventil demontiert werden. Hierzu müssen vorab die beiden Schrauben sowie der Luftschlauch entfernt werden, siehe Abbildung 6-10 und Abbildung 6-11. Die Ablage des Schlauches erfolgt rein positionsgeregelt (Abbildung 6-11 (k)). Die restlichen Vorgänge sind äquivalent zu den vorherigen Anwendungen.

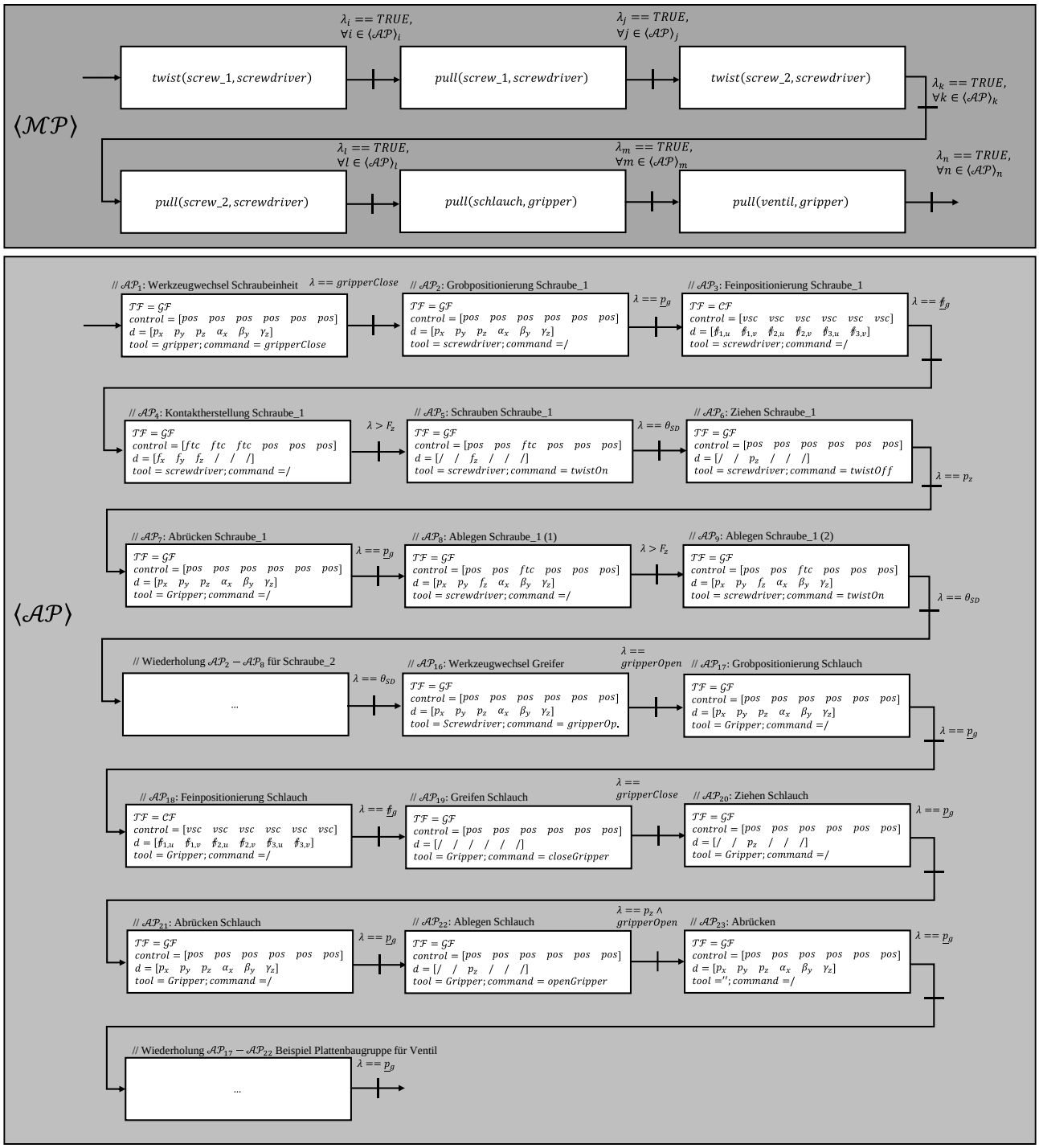
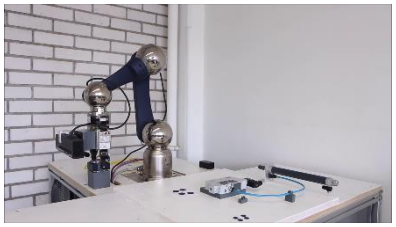


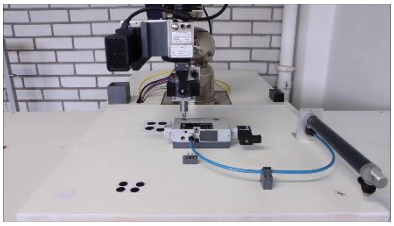
Abbildung 6-10: Ventilbaugruppe: Erzeugter Plan in Manipulationsprimitiven und zugehöriges Anwenderprogramm in Aktionsprimitiven (nur Demontage)

(a): *t= 0s* ($\mathcal{AP}_1$)

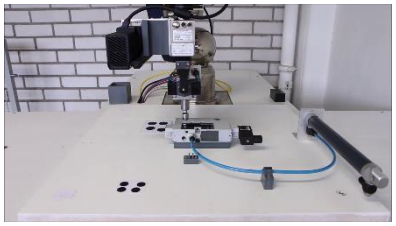


(b): *t= 30s* ($\mathcal{AP}_1$)

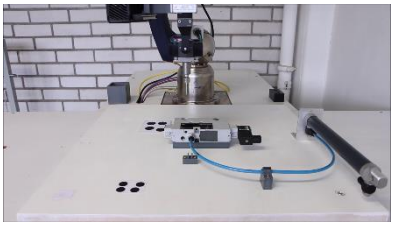
(c): *t= 40s* ($\mathcal{AP}_2$)



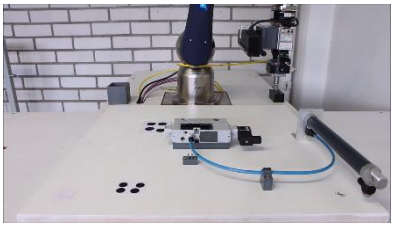
(d): *t= 50s* ($\mathcal{AP}_4$)



(e): *t= 62s* ($\mathcal{AP}_5$)

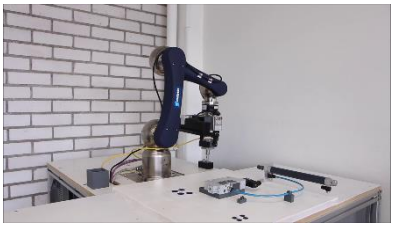


(f): *t= 70s* ($\mathcal{AP}_7$)



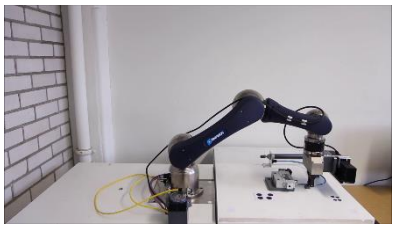
(g): *t= 87s* ($\mathcal{AP}_9$)

(h): *t= 126s* ($\mathcal{AP}_{13}$)



(i): *t= 160s* ($\mathcal{AP}_{15}$)

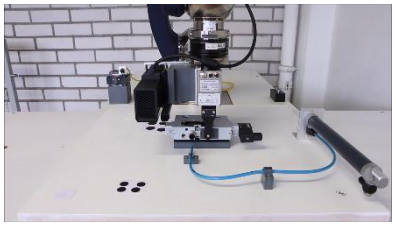
(j): *t= 212s* ($\mathcal{AP}_{18}$)



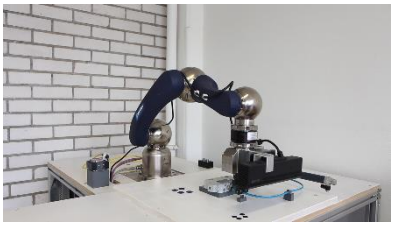
(k): *t= 230s* ($\mathcal{AP}_{22}$)

(l): *t= 267s* ($\mathcal{AP}_{27}$)

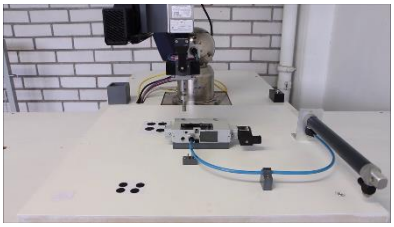
(m): *t= 298s* ($\mathcal{AP}_{29}$)



(n): *t= 319s*



(o): *t= 352s*



(p): *t= 422s*

(q): *t= 515s*

(r): *t= 545s*

Abbildung 6-11: Ausführung der Instandsetzungsaufgabe „Ventilbaugruppe“

6.3.2 Analyse der Aufgabenbeispiele

Die in 6.3.1 dargestellten Aufgabenbeispiele werden abschließend gesamtheitlich untersucht. Die Analyse erfolgt anhand folgender Messungen, wobei eine fünffache Wiederholung jeder Aufgabe durchgeführt wurde:

- Mittelwert $t_{-,exe}$ und Standardabweichung $t_{\sigma,exe}$ der Ausführungszeit der Aufgabe in [s].
- Zeit in [s] als Mittelwert $t_{-,path}$ und Standardabweichung $t_{\sigma,path}$, in welcher der Roboter in Positionsregelung betrieben wird.
- Zeit in [s] als Mittelwert $t_{-,vsc}$ und Standardabweichung $t_{\sigma,vsc}$, in welcher der Roboter in bildbasierter Regelung betrieben wird.
- Zeit in [s] als Mittelwert $t_{-,ftc}$ und Standardabweichung $t_{\sigma,ftc}$, in welcher der Roboter in Kraft-Momenten-Regelung betrieben wird.
- Zeit in [s] als Mittelwert $t_{-,n}$ und Standardabweichung $t_{\sigma,n}$ in welcher der Roboter sich in Nebenzeit befindet. Als Nebenzeiten werden Greifprozesse, Sensorkalibrierung und –trigger zusammengefasst, in welchen der Roboter keine Produktivbewegung ausführt.
- Anzahl der auszuführenden Manipulationsprimitive $|\mathcal{MP}| \in \mathbb{N}_+$.
- Erfolgsrate $S \in [0 \dots 1]$ über alle Experimente.

Die Analyse der einzelnen Zeiten soll vor allem zur Darstellung zukünftiger Optimierungspotentiale dienen sowie einen tieferen Einblick in die Dauer der Ausführung erlauben.

In Tabelle 6-1 sind die Ergebnisse für alle obigen Punkte ausgeführt, während Abbildung 6-12 explizit die einzelnen Zeitmessungen (Mittelwert (a); Standardabweichung (b)) darstellt. Es sei vorab angemerkt, dass die Gesamtdynamik des Systems sehr stark durch die niedrige Zykluszeit der Roboterregelung in ROS begrenzt ist, vergleiche auch 5.2.1. Durch Parameteranpassung (Regelungsparameter und kinematische Parameter) könnte eine geringfügige Verbesserung erreicht werden, was jedoch nicht das Ziel der vorliegenden Untersuchung sein soll. Vielmehr soll betrachtet werden, wie sich die einzelnen Zeiten auf die unterschiedlichen Regelverfahren aufteilen. Auch sollen die Ursachen bei einem Fehlschlag in der Ausführung diskutiert werden.

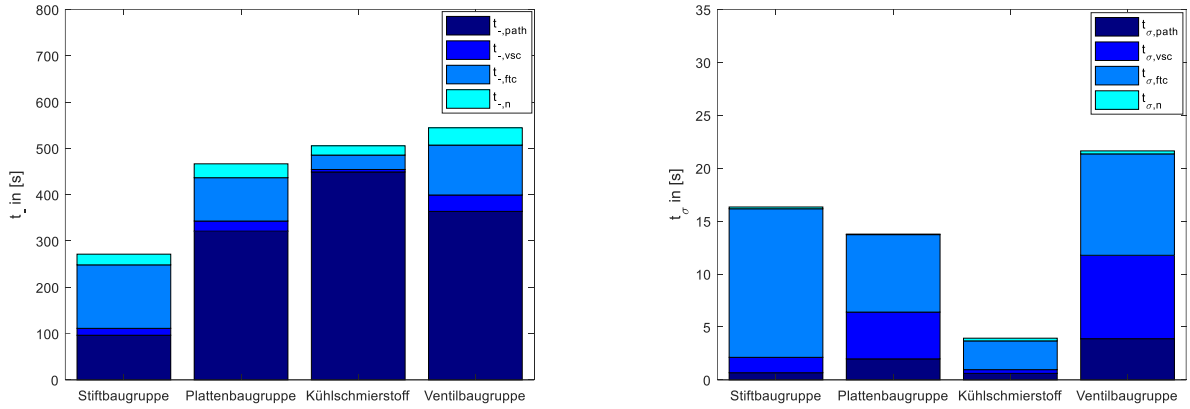
Es zeigt sich, dass, bis auf die Stiftbaugruppe, der Großteil der Gesamtheit auf rein bewegungsgeführte Manipulationen entfällt. Bei der Stiftbaugruppe wird für die Fügeapplikation des Stifts die Admittanzregelung verwendet. Aufgrund unterschiedlicher Startpositionen für die Fügeapplikation, ist auch hier die größte Standardabweichung zu beobachten. Die allgemein hohen Zeiten für Kraft-Momenten geregelte Manipulationen sind zum einen auf die niedrige Dynamik des Admittanzreglers, zum anderen auf die frühzeitige Verwendung des Reglers für die Funktion der Objektablage, zurückzuführen. Da ein Großteil der Manipulationen Kraft-Momenten geregelt ausgeführt wird, ergibt sich hier noch großes Optimierungspotential zur Reduktion der Ausführungszeit.

Beim Visual-Servoing ist die Dynamik, neben den bekannten Rahmenbedingungen, durch die zeitliche Dauer, in der Extraktion der Features, begrenzt. Die Standardabweichung für bildgeregelte Manipulationen kann auf unterschiedliche Lichtverhältnisse und daraus resultierende Abschattungen rückgeführt werden.

Die Erfolgsrate kann über alle Experimente als gut bezeichnet werden. Bis auf das Experiment mit dem Kühlschmierstoff schlägt das Experiment für alle anderen Baugruppen jeweils einmal fehl.

Tabelle 6-1: Ergebnisse der Aufgabenbeispiele

	Applikation 1-Stiftbaugruppe	***Applikation 2-Plattenbaugruppe***	***Applikation 3-Kühlschmierstoff***	***Applikation 4-Ventilbaugruppe***
$t_{-,exe}$ *in* [s]	271.456	466.574	505.517	544.385
$t_{-,path}$ *in* [s]	96.338	321.079	448.965	363.753
$t_{-,vsc}$ *in* [s]	14.876	21.926	5.369	35.473
$t_{-,ftc}$ *in* [s]	137.145	93.632	31.125	107.796
$t_{-,n}$ *in* [s]	23.097	29.937	20.057	37.362
$t_{\sigma,exe}$ *in* [s]	14.116	8.770	3.326	6.264
$t_{\sigma,path}$ *in* [s]	0.674	1.982	0.611	3.887
$t_{\sigma,vsc}$ *in* [s]	1.453	4.419	0.354	7.882
$t_{\sigma,ftc}$ *in* [s]	14.041	7.320	2.716	9.583
$t_{\sigma,n}$ *in* [s]	0.166	0.049	0.256	0.283
$\lvert\mathcal{MP}\rvert \in \mathbb{N}_+$	4	10	2	12
$S \in [0 \ldots 1]$	0.8	0.8	1	0.8



(a) (b)

Abbildung 6-12: Zeitmessungen der Anwendungsbeispiele für fünf Wiederholungen: Ausführungszeiten als Mittelwert (a); Standardabweichung der Ausführungszeiten (b)

Bei der Stiftbaugruppe wird, aufgrund des starken Sensorrauschens, einmal zu frühzeitig die Fügeposition des Stifts erkannt, wodurch dieser nicht richtig montiert werden kann. Bei der Platten- und der Ventilbaugruppe kann die Schraube aus der Ablageposition jeweils einmal nicht aufgenommen werden. Dies liegt vor allem an der mechanischen Gestaltung der Schraubeinheit, kann aber auch an dem Verlust der Kontaktkraft durch den Admittanzregler resultieren, die der geringen Reglerdynamik geschuldet ist.

6.4 Zusammenfassung und Diskussion

In diesem Kapitel ist der experimentelle Aufbau des Robotersystems sowie der entwickelten Steuerungsarchitektur dargestellt worden. Anhand von vier anwendungsnahen Beispielen, konnten die in Kapitel 4 und Kapitel 5 entwickelten Konzepte, gesamtheitlich experimentell validiert werden. Es zeigt sich, dass sich die Funktionalität der entwickelten Konzepte auch in den Laborexperimenten bestätigt. Damit konnte erstmalig einem Robotersystem die Fähigkeiten

gegeben werden, autonom Manipulationen zur Ausführung von Instandhaltungsaufgaben zu planen und auszuführen. Durch die geeignete Abstraktion, aufbauend auf Manipulationsprimitiven planungsseitig und Aktionsprimitiven als Bindeglied zur Ausführung, kann ein System ermöglicht werden, welches eine hohe Performance bei gleichzeitig hoher Autonomie gestattet.

Für zukünftige Arbeiten sollten, aus praktischer Sicht, alle Reglerfunktionalitäten auf ein Echtzeitbetriebssystem mit höherer Reglerfrequenz übertragen werden. So könnte die Dynamik des Gesamtsystems wesentlich gesteigert werden, da eine Ausnutzung der gesamten mechanischen Bandbreite erlaubt wird. Für den IBVS-Regler werden derzeitig externe Marker zur Bestimmung von Features verwendet. Für zukünftige Arbeiten sollte ein Verfahren entwickelt werden, welches es ermöglicht robust Features aus den zum Teil, aus Sicht der Bildverarbeitung betrachtet, komplizierten Objekten zu extrahieren. Hierdurch könnte vorab eine Merkmalsdatenbank für die einzelnen Objekte angelegt werden, die dem Robotersystem für die bildbasierte Regelung geeignete Features zur Verfügung stellt.

Neben der automatisierten (De-)Montage, welche sowohl für die Wartung als auch für die Instandsetzung fundamental ist, müssen künftig auch Methoden erforscht werden, die es erlauben, die eigentliche Servicefunktion, beispielsweise das Nachfüllen von Kühlschmierstoff oder das Reinigen einer Maschine, automatisch zu erzeugen. Ein weiterer großer und vor allem für die Praxis entscheidender Punkt, welcher in dieser Arbeit nicht betrachtet wurde, ist die Behandlung von Unsicherheiten in der Ausführung der Aufgaben. Wie reagiert der Roboter, wenn beispielsweise ein Objekt aufgrund von Umwelteinflüssen (Korrosion, mechanischer Verschleiß, …) nicht demontiert werden kann? Wie sieht ein Alternativplan aus? Wie kann allgemein das Fehlerhandling aussehen? Diese Fragestellungen sollen aber Teil zukünftiger Forschungsarbeiten sein, da das Ziel dieser Arbeit in der Findung einer allgemeinen, übergeordneten Lösungsstrategie zur Autonomiebildung liegt.

All diese Punkte sind für den Einzug autonomer Robotersysteme in den industriellen Alltag von höchster Relevanz, da neben einer hohen Funktionalität auch eine hohe Ausfallsicherheit und Robustheit des Systems gefordert werden muss.

7 Zusammenfassung und Ausblick

„Wir können nur eine kurze Distanz in die Zukunft blicken, aber dort können wir eine Menge sehen, was getan werden muss.“
Alan Turing

Diese Arbeit entwickelt neuartige Methoden und Algorithmen, welche einem Robotersystem die (Basis-)Fähigkeiten zur Automatisierung von Instandhaltungsaufgaben im produktionstechnischen Umfeld bereitstellen. Ausgehend von einer angefragten Aufgabe kann das Robotersystem die zugehörigen Demontage- und Montagevorgänge, welche einen Großteil der Aktionen in der Instandhaltung darstellen, unter realistischen Rahmenbedingungen autonom Planen und Ausführen. Somit kann diese Arbeit, die in 1.1 gestellte Forschungsfrage: *„Wie kann auf Basis einer Fehlerdiagnose automatisiert eine Wartungs- oder Instandsetzungsoperation in einer Produktionsanlage durchgeführt werden?“*, vollständig beantworten, wenn auch in Zukunft noch viele Probleme bis zum realen Praxiseinsatz zu lösen sind. Durch die Entwicklung besonders zeiteffizienter Planungsverfahren, die Umweltunsicherheiten kompensieren können sowie der Kopplung mit einer reaktiven Ausführung, ist ein erster Grundstein in die richtige Richtung gelegt.

Es bleibt abzuwarten, wie sich eine neue Generation von Robotersystemen für derart komplexe Aufgaben weiterentwickelt und einen Beitrag zur Attraktivitätssteigerung, für die Produktion in Hochlohnländern, leistet. Folgend sollen die geleisteten Beiträge dieser Arbeit sowie die aktuellen Grenzen und zukünftig möglichen Forschungsthemen zusammengefasst werden.

Geleistete Beiträge. In dieser Arbeit konnte erstmalig eine komplette Lösungskette zur Automatisierung von Instandhaltungsaufgaben durch Robotersysteme erforscht und experimentell validiert werden. Die wesentlichen Kernpunkte der entwickelten Methoden lassen sich wie folgt zusammenfassen.

- ***Aufgabenplanung:*** Es konnte eine Berechnungsvorschrift mit niedriger Berechnungskomplexität gefunden werden, die die Bestimmung vorhandener Demontageräume erlaubt. Durch die Abstraktion in ein symbolisches Planungsproblem kann die nachfolgende Findung nötiger Manipulationen ebenso effizient wie flexibel gelöst werden.
- ***Aufgabenexploration:*** Die entwickelte Methode erlaubt eine effiziente Akquirierung der relevanten Umweltinformationen durch die Planung geeigneter Sensorposen. Aufgrund der Möglichkeit zur Szenenanalyse lassen sich einfache Objektzusammenhänge erkennen und für die Aufgabenplanung bereitstellen, wodurch die Kompensation von Umweltunsicherheiten ermöglicht wird.
- ***Schrittweitensteuerung der Bahnplanung:*** Durch die Entwicklung einer Vorverarbeitungsmethode auf Basis der globalen Belegtheit der Umgebung, konnte eine Schrittweitensteuerung entwickelt werden, die unabhängig von dem verwendeten globalen Bahnplanungsalgorithmus, eine drastische Reduktion des Planungsraum erlaubt. Hierdurch

konnten bis zu 70% der Planungszeit eingespart (Planungsraum 90%) und die Erfolgsrate in der Lösungsfindung gesteigert werden, ohne dabei signifikant die Bahngüte zu beeinflussen.

- ***Laufzeitinterpretation der Manipulationsprimitive:*** Das vorgestellte Verfahren ermöglicht die Übersetzung der symbolischen Aktionen in elementare Roboteranweisungen. Somit kann der gesamte Manipulationsprozess, von der Planung bis zur Ausführung, autonom erfolgen. Durch den Einsatz eines Interpreters kann bedarfsgerecht auf Änderungen während der Planausführung eingegangen werden. Durch die online-Prüfung aktueller Objekt- und Roboterzustände müssen nicht für alle Transitionen die notwendigen Roboteranweisungen vorab bestimmt, sondern nur in Abhängigkeit der jeweiligen Entscheidungsregel eingefügt werden.
- ***Steuerungsarchitektur:*** Alle entwickelten Verfahren sind in eine unabhängige Steuerungsarchitektur integriert, welche für unterschiedliche Roboterplattformen Verwendung finden kann. Hierzu muss nur eine Parametrierung der Kommunikation an die kartesische Geschwindigkeitsschnittstelle des verwendeten Systems erfolgen.
- ***Experimentelle Validierung:*** Neben der Validierung der einzelnen Verfahren, konnte, anhand anwendungsnaher Beispiele, erfolgreich die Funktion des Gesamtsystems experimentell nachgewiesen werden.

Grenzen der Verfahren. Die entwickelten Methoden ermöglichen zwar eine generelle Lösung des vorgestellten Problems, jedoch sind auch einige Rahmenbedingungen zu beachten oder Grenzen vorhanden, die einen industriell sinnvollen Einsatz derzeit nicht vollständig gewährleisten. Dabei sollen zusammenfassend die wichtigsten Punkte genannt werden, die bereits ausführlich in den einzelnen Kapiteln diskutiert wurden.

- Für die Aufgabenplanung sind zwingend die notwendigen semantischen und geometrischen Informationen, gemäß des Baugruppenmodells, erforderlich. Des Weiteren kann die Planung fehlschlagen, falls zur Lösungsfindung die Bildung von Unterbaugruppen nötig ist. Ein weiteres Defizit besteht darin, dass mechanische Instabilitäten, die durch die Manipulationskräfte und -momente auftreten können, derzeit nicht berücksichtigt werden. Auch kann dem derzeit nicht entgegengewirkt werden, da nur ein Roboterarm zur Verfügung steht. Für viele praktische Aufgaben ist es jedoch notwendig, dass während der Manipulation ein weiterer Roboterarm das Bauteil oder die Baugruppe fixiert.
- Die entwickelte Szenenanalyse der visuellen Perzeption eignet sich bei a priori unbekannten Objekten derzeit nur bedingt zur Bestimmung eines Kontaktgraphs. Können die einzelnen Objekte semantisch nicht korrekt annotiert werden, kann durch die Bestimmung der geometrischen Primitive nur eine Vermutung über die Zuordnung der Objekte erfolgen.
- Derzeit bestehen erst die elementaren Grundfunktionen mit der automatisierten Generierung von Robotermanipulationen für die Demontage- und Montage. Für Wartungsaufgaben muss aber auch die eigentliche Wartungsaufgabe automatisch erzeugt werden, beispielsweise der Nachfüllprozess von Schmierstoffen oder das Reinigen von Komponenten. Hierzu ist eine Erweiterung des aktuellen Planers nötig.
- Eines der größten Hindernisse besteht gegenwärtig noch im Fehlschlag des Plans zur Ausführung. Zwar kann durch das interpreterbasierte Konzept direkt eine alternative Aktion ausgeführt werden, jedoch kann der aktuelle Planer derzeit keinen Alternativplan bereitstellt.

Zukünftige Forschungsthemen. Aus den Grenzen ergeben sich neue Zielstellungen, die in zukünftigen Forschungsarbeiten Beachtung finden könnten. Einige der zentralen Punkte und Fragestellungen lassen sich wie folgt formulieren.

- Zur Vermeidung von mechanischen Instabilitäten sollte ein Stabilitätstest, der Roboterinteraktionskräfte und Reibungs- sowie Zwangskräfte in der Baugruppe bilanziert, in die Planung integriert werden. Aus diesem könnten dann geeignete Interaktions- und Fixierpositionen für eine Zwei-Arm-Kinematik bestimmt werden. Durch die Verwendung einer Zwei-Arm-Kinematik ergeben sich auch neue Fragestellungen für die Planung, da es dann nicht mehr sinnvoll wäre nur lineare Pläne zu erzeugen.
- Neben der Verbesserung der Szenenanalyse sollten künftig nicht nur visuelle Sensordaten zur Bestimmung eines Kontaktgraphen bei a priori unbekannten Objekten verwendet werden, da dies auch nur eingeschränkt möglich ist. Zur Bestimmung vorhandener Freiheitsgrade kann das Robotersystem auch durch geschicktes Ausprobieren, und den zur Verfügung stehenden Kraft-Momenten-Sensordaten, eine geeignete Lösung finden. Für die Szenenanalyse würde der Einsatz von Instance Segmentation beispielsweise mit der in [362] vorgeschlagenen Architektur wahrscheinlich eine interessante und deutlich verbesserte Alternative darstellen.
- Die Bereitstellung von Alternativplänen bei Fehlschlag der Ausführung, muss zwingend erforscht werden. Erst hierdurch kann die Fehlertoleranz des Systems erhöht werden. Auch sollte das System aus Fehlschlägen lernen und geeignetere Sollwerte für die zur Verfügung stehenden Steuergrößen erzeugen. Besonders geeignet wären für diese Problemstellung datengetriebene Ansätze aus dem Bereich des maschinellen Lernens.

Die Lösung dieser Probleme ermöglicht eine weitere Verschiebung der Grenze zur praxistauglichen Nutzbarkeit und verspricht einen Einzug autonomer Robotersysteme in Produktionsstätten der Zukunft, wodurch dem Bedarf an steigender Automatisierung in Hochlohnländern Rechnung getragen werden kann.

Literaturverzeichnis

[1] B. Bertsche und G. Lechner, 2004, „Berechnung reparierbarer Systeme," in *Zuverlässigkeit im Fahrzeug-und Maschinenbau: Ermittlung von Bauteil-und System-Zuverlässigkeiten,* B. Bertsche und G. Lechner, Hrsg., Berlin: Springer, 3. Auflage, pp. 337-410, ISBN 978-3-540-20871-6.

[2] B. Leidinger, 2014, *Wertorientierte Instandhaltung- Kosten senken, Verfügbarkeit erhalten,* Wiesbaden: Springer Gabler, 1. Auflage, ISBN 978-3-658-04400-8.

[3] A. Kuhn, G. Schuh und B. Stahl, 2006, „*Nachhaltige Instandhaltung: Trends, Potentiale und Handlungsfelder Nachhaltiger Instandhaltung*," Frankfurt: VDMA Verlag GmbH, ISBN 3-8163-0522-9.

[4] F. Bünting, 2009, „Lebenszykluskostenbetrachtungen bei Investitionsgütern," in *Lebenszykluskosten optimieren: Paradigmenwechsel für Anbieter und Nutzer von Investitionsgütern,* S. Schweiger, Hrsg., Wiesbaden: Gabler, 1. Auflage, pp. 35-51, ISBN 978-3-8349-0989-3.

[5] E. Abele, M. Dervisopoulos und B. Kuhrke, 2009, „Bedeutung und Anwendung von Lebenszyklusanalysen bei Werkzeugmaschinen," in *Lebenszykluskosten optimieren: Paradigmenwechsel für Anbieter und Nutzer von Investitionsgütern*, S. Schweiger, Hrsg., Wiesbaden: Gabler, 1. Auflage, pp. 51-80, ISBN 978-3-8349-0989-3.

[6] G. Schuh, A. Kampker, B. Franzkoch und N. Wemhöner, 2005, „*Studie Intelligent Maintenance- Potentiale zustandsorientierter Instandhaltung*," Aachen: Eigendruck des Werkzeugmaschinenlabor, Rheinisch-Westfälische Technische Hochschule Aachen.

[7] K. Matyas, 2013, *Instandhaltungslogistik- Qualität und Produktivität steigern,* München: Carl Hanser, 5.Auflage, ISBN 978-3-446-43560-5.

[8] C. Eisele und E. Abele, 2012, „*Energieeffiziente Produktionsmaschinen durch Simulation in der Produktentwicklung*," Forschungsbericht des BMBF Verbundprojektes eSimPro, Institut für Produktionsmanagement, Technologie und Werkzeugmaschinen (PTW).

[9] W. Becker und F. Brinkmann, 2000, „*Kostenrechnung für die Instandhaltung- Ergebnisse einer empirischen Untersuchung*," Lehrstuhl für Betriebswirtschaftslehre, Universität Bamberg, ISBN 3–931810–19–4.

[10] R. B. Neurath, 1995, „*Die Bedeutung der Instandhaltung*," Bad Homburg: Eigendruck.

[11] S. Christoph, 2013, „*Industrielle Arbeitskosten im internationalen Vergleich*," iw- Institut der deutschen Wirtschaft, Bd. 40, Nr. 3, DOI 10.2373/1864-810X.13-03-06.

[12] G. A. Bekey, 2005, *Autonomous Robots: from biological inspiration to implementation and control,* Cambridge: MIT Press, 1. Auflage, ISBN 0-262-025788-7.

[13] C. Friedrich, A. Lechler und A. Verl, 2014, „Autonomous Systems for Maintenance Tasks – Requirements and Design of a Control Architecture," *International Conference on System-Integrated Intelligence: Challenges for Product and Production Engineering*, pp. 596-605, DOI 10.1016/j.protcy.2014.09.020.

[14] C. Friedrich, A. Lechler und A. Verl, 2016, „The Control Architecture RoViDiAsS- A Robotic Visual Disassembly and Assembly System," *IEEE International Conference on Industrial Technology*, pp. 113-118, DOI 10.1109/ICIT.2016.7474735.

[15] M. Birkhold, C. Friedrich und A. Lechler, 2015, „A model-based CNC Architecture for Process Programming," *The International Conference on Flexible Automation and Intelligent Manufacturing*, pp. 1-8, DOI 10.13140/RG.2.2.24695.47522.

[16] M. Birkhold, C. Friedrich und A. Lechler, 2015, „Automation of the Casting Process using a model-based NC Architecture," *IFAC-PapersOnLine*, Bd. 48, Nr. 17, pp. 195-200, DOI 10.1016/j.ifacol.2015.10.102.

[17] C. Friedrich, A. Lechler und A. Verl, 2016, „A Planning System for Generating Manipulation Sequences for the Automation of Maintenance Tasks," *IEEE International Conference on Automation Science and Engineering*, pp. 843-848, DOI 10.1109/COASE.2016.7743489.

[18] C. Friedrich, A. Csiszar, A. Lechler und A. Verl, 2018, „Efficient task and path planning for maintenance automation using a robot system," *IEEE Transactions on Automation Science and Engineering*, Bd. 15, Nr. 3, pp. 1205-1215, DOI 10.1109/TASE.2017.2759814.

[19] C. Friedrich, C. v. Arnim, A. Lechler und A. Verl, 2017, „Visual perception for robot based maintenance automation," *IEEE International Conference on Advanced Intelligent Mechatronics*, pp. 388-393, DOI 10.1109/AIM.2017.8014048.

[20] C. Friedrich, V. Ziehlke, M. Toussaint, A. Lechler und A. Verl, 2017, „Environment Modeling for Maintenance Automation- A Next-Best-View Approach for combining Space Exploration and Object Recognition Tasks," *IEEE International Conference on Automation Science and Engineering*, pp. 1445-1450, DOI 10.1109/COASE.2017.8256307.

[21] C. Friedrich, A. Csiszar, A. Lechler und A. Verl, 2017, „Fast robot task and path planning based on CAD and vision data," *IEEE International Conference on Advanced Intelligent Mechatronics*, pp. 1633-1638, DOI 10.1109/AIM.2017.8014252.

[22] C. Friedrich, R. Gulde, A. Lechler und A. Verl, 2019 „Maintenance automation: Concepts for manipulation planning, control and execution," *(will be submitted)*, pp. 1-9.

[23] DIN 31051, 2009, *Grundlagen der Instandhaltung*, Beuth Verlag GmbH.

[24] G. Pawellek, 2013, *Integrierte Instandhaltung und Ersatzteillogistik- Vorgehensweisen, Methoden, Tools*, Berlin: Springer Vieweg, 1. Auflage, ISBN 978-3-642-31382-0.

[25] M. Schenk, M. Endig, C. Freund, J. Götze, K. Hänsch, S. Kumetz und e. al., 2010, *Instandhaltung technischer Systeme- Methoden und Werkzeuge zur Gewährleistung eines sicheren und wirtschaftlichen Anlagenbetriebs*, Heidelberg: Springer, 1. Auflage, ISBN 978-3-642-03948-5.

[26] J. Moubray, 1997, *Reliability-centered maintenance*, New York: Industrial Press Inc., 2. Auflage, ISBN 978-0831130787.

[27] J. Reichel, G. Müller und J. Mandelartz, 2009, *Betriebliche Instandhaltung*, Dordrecht: Springer, 1. Auflage, ISBN 978-3-642-00501-5.

[28] K. Matyas, 2002, „*Ganzheitliche Optimierung durch individuelle Instandhaltungsstrategien*," Industrie Managment, Bd. 18, Nr. 2, pp. 13-16.

[29] M. Strunz, 2012, *Instandhaltung: Grundlagen- Strategien- Werkstätten*, Berlin: Springer Vieweg, 1. Auflage, ISBN 978-3-642-27389-6.

[30] B. v. Thiel, 2011, „Vorausschauende Instandhaltung durch intelligente Bauteile (SFB 653/ Titel-Nr. 9770)," *Weka- Instandhaltung online*.

[31] DIN 40041, 1990, *Zuverlässigkeit Begriffe*, Beuth Verlag GmbH.

[32] J. Wieser, 2009, *Intelligente Instandhaltung zur Verfügbarkeitssteigerung von Werkzeugmaschinen*, Dissertation, Fakultät für Maschinenbau, Universität Karlsruhe, Aachen: Shaker, ISBN 978-3-8322-7882-3.

[33] B. Bertsche und G. Lechner, 1999, *Zuverlässigkeit im Maschinenbau: Ermittlung von Bauteil- und System-Zuverlässigkeiten*, Berlin: Springer, 2. Auflage, ISBN 978-3-662-11004-1.

[34] C. Anders, 1998, *Adaptierbares Diagnosesystem bei Transferstraßen*, Dissertation, Fakultät Konstruktions- und Fertigungstechnik, Universität Stuttgart, Berlin: Springer, ISBN 978-3-540-65405-6.

[35] F.-U. Hess und T. Rahmann, 2013, „*Maintenance Efficiency Report*," Berlin: Eigendruck T.A. Cook.

[36] A. Seufzer, 2000, *Durchgängige Unterstützung von Instandsetzungsprozessen*, Dissertation, Fakultät Maschinenbau, Universität Hannover, Düsseldorf: VDI, ISBN 3-18-353902-0.

[37] M. Walther und A. Verl, 2007, „Antriebsnahe Maschinendiagnose," *Zuverlässigkeit und Diagnose in der Produktion*, Forschritt-Berichte VDI, Reihe 2, Nr. 663.

[38] DIN 2888, 1999, *Zustandsorientierte Instandhaltung*, Beuth Verlag GmbH.

[39] A. Schwung, A. Ortseifen und J. Adamy, 2010, „Anwendung zeitdiskreter rekurrenter Fuzzy-Systeme zur Fehlerdiagnose," *at-Automatisierungstechnik*, Bd. 58, Nr. 6, pp. 322-331, DOI 10.1524/auto.2010.0844.

[40] R. Isermann, 2008, *Mechatronische Systeme*, Berlin: Springer, 2. Auflage, ISBN 978-3-540-32336-5.

[41] R. Isermann, 2011, *Fault-Diagnosis Applications- Model-based Condition Monitoring: Actuators, Drives, Machinery, Plants, Sensors, and Fault-tolerant Systems*, Heidelberg: Springer, 1. Auflage, ISBN 978-3-642-12766-3.

[42] A. Dietmair, M. Walther und G. Pritschow, 2005, „Antriebsbasierte Maschinendiagnose," *wt Werkstattstechnik*, Bd. 95, Nr. 5, pp. 351-356.

[43] A. Verl, U. Heisel, M. Walther und D. Maier, 2009, „Sensorless automated condition monitoring for the control of the predictive maintenance of machine tools," *CIRP Annals - Manufacturing Technology*, Bd. 58, Nr. 1, pp. 375-378, DOI 10.1016/j.cirp.2009.03.039.

[44] E. Münz, 2005, *Identifikation und Diagnose hybrider dynamischer Systeme*, Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Universität Karlsruhe, Karlsruhe: Universitätsverlag, ISBN 3-86644-029-4.

[45] R. Isermann, 2010, „Modellbasierte Überwachung und Fehlerdiagnose von kontinuierlichen technischen Prozessen," *at-Automatisierungstechnik*, Bd. 58, Nr. 6, pp. 291-305, DOI 10.1524/auto.2010.0846.

[46] M. Münchhof, 2007, „Fehlerdiagnose für hydraulische Servo-Achsen," *at-Automatisierungstechnik*, Bd. 55, Nr. 2, pp. 96-103, DOI 10.1524/auto.2007.55.2.96.

[47] A. K. Jardine, D. Lin und D. Banjevic, 2006, „A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Mechanical Systems and Signal Processing*, Bd. 20, Nr. 7, pp. 1483-1510, DOI 10.1016/j.ymssp.2005.09.012.

[48] C. W. Frey, 2008, „Agentenbasierte Diagnose und Monitoring feldbusbasierter Automatisierungsanlagen mittels selbstorganisierender Karten und Fuzzy-Methoden," *atp-Automatisierungstechnische Praxis*, Bd. 49, Nr. 8, pp. 36-41.

[49] Q. Miao und V. Markis, 2007, „Condition Monitoring and classification of rotating machinery using wavelets and hidden Markov models," *Mechanical Systems and Signal Processing*, Bd. 21, Nr. 2, pp. 840-855, DOI 10.1016/j.ymssp.2006.01.009.

[50] P. Gerland, 2011, *Klassifikationsgestützte on-line Adaption eines robusten beobachterbasierten Fehlerdiagnoseansatzes für nichtlineare Systeme*, Dissertation, Fachbereich Maschinenbau, Universität Kassel, Kassel: University press, ISBN 978-3-89958-880-4.

[51] M. Blanke, M. Kinnaert, J. Lunze und M. Staroswiecki, 2016, *Diagnosis and Fault-Tolerant Control*, Heidelberg: Springer, 3. Auflage, ISBN 978-3-662-47942-1.

[52] S. Kanev, 2004, *Robust Fault-Tolerant Control*, PhD Thesis, University of Twente.

[53] H. Unbehauen, 2011, *Regelungstechnik 3- Identifikation, Adaption, Optimierung*, Wiesbaden: Vieweg Teubner, 7. Auflage, ISBN 978-3-8348-1419-7.

[54] P. Ioannou und J. Sun, 1996, *Robust adaptive control*, Mineola: Prentice-Hall, 1. Auflage, ISBN 978-0486498171.

[55] J. Lunze und T. Steffen, 2003, „Rekonfiguration linearer Systeme bei Aktor- und Sensorfehlern," *at-Automatisierungstechnik*, Bd. 51, Nr. 2, pp. 60-68, DOI 10.1524/auto.51.2.60.22802.

[56] J. Lunze und T. Steffen, 2006, „Control reconfiguration after actuator failures using disturbance decoupling methods," *IEEE Transactions on Automatic Control*, Bd. 51, Nr. 10, pp. 1590-1601, DOI 10.1109/TAC.2006.882938.

[57] J. Lunze und J. Richter, 2006, „Entwurfsmethode für den verallgemeinerten virtuellen Aktor zur Rekonfiguration von Regelkreisen," *at-Automatisierungstechnik*, Bd. 54, Nr. 7, pp. 353-361, DOI 10.1524/auto.2006.54.7.353.

[58] J. Richter, T. Schlage und J. Lunze, 2007, „Rekonfiguration eines thermofluiden Prozesses mittels virtuellem Aktor," *at-Automatisierungstechnik*, Bd. 55, Nr. 4, pp. 157-169, DOI 10.1524/auto.2007.55.4.157.

[59] J. Sawada, K. Kusumoto, T. Munakata, Y. Maikawa und Y. Ishikawa, 1991, „A mobile robot for inspection of power transmission lines," *IEEE Transactions on Power Delivery*, Bd. 6, Nr. 1, pp. 309-315, DOI 10.1109/61.103753.

[60] P. Debenest, M. Guarnieri, K. Takita, E. Fukushima, S. Hirose, K. Tamura, A. Kimura, H. Kubokawa, N. Iwama und F. Shiga, 2008, „Expliner- Robot for inspection of transmission lines," *IEEE International Conference on Robotics and Automation*, pp. 3978-3984, DOI 10.1109/ROBOT.2008.4543822.

[61] Y. Song, H. Wang, Y. Jiang und L. Ling, 2012, „AApe-D: a Novel Power transmission line maintenance robot für broken strand repair," *IEEE International Conference on Applied Robotics for the Power Industry*, pp. 108-113, DOI 10.1109/CARPI.2012.6473359.

[62] C. Garcia, R. Saltaren und R. Aracil, 2011, „Experiences in the development of a teleoperated parallel robot for aerial line maintenance," *Robotica*, Bd. 29, Nr. 6, pp. 873-881, DOI 10.1017/S0263574711000087.

[63] Y. Song, H. Wang und J. Zhang, 2014, „A Vision-Based Broken Strand Detection Method for a Power-Line Maintenance Robot," *IEEE Transactions on Power Delivery*, Bd. 29, Nr. 5, pp. 2154-2161, DOI 10.1109/TPWRD.2014.2328572.

[64] C. Birkenhofer, K.-U. Scholl, M. Zöllner und R. Dillmann, 2003, „MakroPLUS- Ein modulares Systemkonzept eines mehrsegmentigen, autonomen Kanalroboters," in *Informatik aktuell- Autonome mobile Systeme*, D. Rüdiger, W. Heinz und G. Tilo, Hrsg., Berlin: Springer, pp. 272-280, DOI 10.1007/978-3-642-18986-9_28.

[65] G. Besselmann, 2017, „*KAIRO – modularer schlangenartiger Inspektionsroboter*," FZI Forschungszentrum Informatik, [Online]. Available: http://www.fzi.de/forschung/projekt-details/kairo/. [Zugriff am 18 09 2017].

[66] X. Gao, Z. Jiang, J. Gao, D. Xu, Y. Wang und H. Pan, 2008, „Boiler maintenance robot with multi-operational schema," *IEEE International Conference on Mechatronics and Automation*, pp. 610-615, DOI 10.1109/ICMA.2008.4798826.

[67] X. Gao, D. Xu, Y. Wang, H. Pan und W. Shen, 2009, „Multifunctional robot to maintain boiler water-cooling tubes," *Robotica*, Bd. 27, Nr. 6, pp. 941-948, DOI 10.1017/S0263574709005360.

[68] G. Paul, S. Webb, D. Liu und G. Dissanayake, 2011, „Autonomous robot manipulator-based exploration and mapping system for bridge maintenance," *Robotics and Autonomous Systems*, Bd. 59, Nr. 7, pp. 543-554, DOI 10.1016/j.robot.2011.04.001.

[69] M. Bücker, 2011, *Entwicklung eines mobilen Roboters für die automatisierte Instandhaltung von Materialflusssystemen*, Dissertation, Fakultät Maschinenbau, Technische Universität Dortmund, Aachen: Shaker, ISBN 978-3-84440-0225-6.

[70] T. Brutscheck, 2012, *Handhabungssystem für die automatisierte Instandhaltung von Stetigfördersystemen*, Dissertation, Fakultät Maschinenbau, Technische Universität Dortmund, Aachen: Shaker, ISBN 978-3-8440-0843-2.

[71] B. Kuhlenkötter, M. Bücker und T. Brutscheck, 2010, „Deploying Mobile Maintenance Robots in Material Flow Systems Using Topological Maps and Local Calibration," Robotics (ISR), *2010 41st International Symposium on and 2010 6th German Conference on Robotics*, pp. 1-7.

[72] G. Arbeiter, A. Bubeck, J. Fischer und B. Graf, 2010, „Teleoperierte mobile Roboter warten Prozessanlagen- Niedrigere Betriebskosten und höhere Arbeitssicherheit," *atp Edition*, Bd. 52, Nr. 5, pp. 44-50.

[73] M. Bengel, K. Pfeiffer, B. Graf, A. Bubeck und A. Verl, 2009, „Mobile Robots for Offshore Inspection and Manipulation," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3317-3322, DOI 10.1109/IROS.2009.5353885.

[74] K. Pfeiffer, M. Bengel und A. Bubeck, 2011, „Offshore Robotics – Survey, Implementation, Outlook," *IEEE International Conference on Intelligent Robots and Systems*, pp. 241-246, DOI 10.1109/IROS.2011.6094661.

[75] o.V., 2015, „Second Hands- A Robot Assistant For Industrial Maintenance Task," [Online]. Available: https://secondhands.eu/. [Zugriff am 2017 09 06].

[76] N. Vahrenkamp, H. Arnst, M. Wächter, D. Schiebener, P. Sotiropoulos, M. Kowalik und T. Asfour, 2016, „Workspace Analysis for Planning Human-Robot Interaction Tasks," *IEEE-RAS 16th International Conference on Humanoid Robots*, pp. 1298-1303, DOI 10.1109/HUMANOIDS.2016.7803437.

[77] J. R. Medina, F. Duvallet, M. Karnam und A. Billard, 2016, „A Human-Inspired Controller for Fluid Human-Robot Handovers," *IEEE-RAS 16th International Conference on Humanoid Robots*, pp. 324-331, DOI 10.1109/HUMANOIDS.2016.7803296.

[78] B. Schmult, 1992, „Autonomous Robotic Disassembly in the Blocks World," *The International Journal of Robotics Research*, Bd. 11, Nr. 5, pp. 437-459, DOI 10.1177/027836499201100502.

[79] M. B.-V. Kuren, 2006, „Flexible robotic demanufacturing using real time tool path generation," *Robotics and Computer-Integrated Manufacturing*, Bd. 22, Nr. 1, pp. 17-24, DOI 10.1016/j.rcim.2005.01.002.

[80] M. Merdan, W. Lepuschitz, T. Meurer und M. Vincze, 2010, „Towards ontology-based automated disassembly systems," *IECON Annual Conference on IEEE Industrial Electronics Society*, pp. 1392-1397, DOI 10.1109/IECON.2010.5675479.

[81] M. Weyrich und Y. Wang, 2013, „Architecture design of a vision-based intelligent system for automated disassembly of E-waste with a case study of traction batteries," *IEEE Conference on Emerging Technologies & Factory Automation*, pp. 1-8, DOI 10.1109/ETFA.2013.6648043.

[82] P. Schumacher und M. Jouaneh, 2013, „A system for automated disassembly of snap-fit covers," *The International Journal of Advanced Manufacturing Technology*, Bd. 69, Nr. 9-12, pp. 2055-2069, DOI 10.1007/s00170-013-5174-8.

[83] H.-J. Kim, R. Harms und G. Seliger, 2007, „Automatic Control Sequence Generation for a Hybrid Disassembly System," *IEEE Transactions on Automation Science and Engineering*, Bd. 4, Nr. 2, pp. 194-205, DOI 10.1109/TASE.2006.880538.

[84] S. Vongbunyong, S. Kara und M. Pagnucco, 2013, „Application of cognitive robotics in disassembly of products," *CIRP Annals- Manufacturing Technology*, Bd. 62, Nr. 1, pp. 31-34, DOI 10.1016/j.cirp.2013.03.037.

[85] S. Vongbunyong, S. Kara und M. Pagnucco, 2015, „Learning and revision in cognitive robotics disassembly automation," *Robotics and Computer-Integrated Manufacturing*, Bd. 34, pp. 79-94, DOI 10.1016/j.rcim.2014.11.003.

[86] U. Rebafka, 2003, *Beitrag zur Entwicklung modularer Demontagewerkzeuge*, Dissertation, Fakultät Verkehrs- und Maschinensysteme, Technische Universität Berlin, Stuttgart: Fraunhofer Verlag, ISBN 978-3-8167-6381-9.

[87] J.-P. Härtwig, 2005, *Verfahren und Systeme zur Demontage komplexer Gebrauchsgüter*, Dissertation, Fakultät Verkehrs- und Maschinensysteme, Technische Universität Berlin, Stuttgart: Fraunhofer Verlag, ISBN 978-3-8167-6963-7.

[88] T. Keil, 2004, *Informationstechnische Integration hybrider Demontagesysteme*, Dissertation, Fakultät Verkehrs- und Maschinensysteme, Technische Universität Berlin, Stuttgart: Fraunhofer Verlag, ISBN ISBN 978-3-8167-6688-9.

[89] J. Pomares, F. Torres und T. Puente, 2001, „Disassembly movements for geometrical objects through heuristic methods," *International Symposium on Intelligent Systems and Advanced Manufacturing SPIE*, pp. 71-80, DOI 10.1117/12.455266.

[90] P. Mendez, T. Medina und J. Baeza, 2002, „Product disassembly scheduling using graph models," *Intelligent Systems and Advanced Manufacturing, International Society for Optics and Photonics*, pp. 63-70, DOI 10.1117/12.455265.

[91] F. Torres, S. Puente und R. Aracil, 2003, „Disassembly Planning Based on Precedence Relations among Assemblies," *The International Journal of Advanced Manufacturing Technology*, Bd. 21, Nr. 5, pp. 317-327, DOI 10.1007/s001700300037.

[92] J. Pomares, T. Puente, F. Torres, A. Candelas und P. Gil, 2004, „Virtual disassembly of products based on geometric models," *Computers in Industry*, Bd. 55, Nr. 1, pp. 1-14, DOI 10.1016/j.compind.2004.03.001.

[93] F. Torres, P. Gil, S. Puente, J. Pomares und R. Aracil, 2004, „Automatic PC disassembly for component recovery," *The International Journal of Advanced Manufacturing Technology*, Bd. 23, Nr. 1/2, pp. 39-46, DOI 10.1007/s00170-003-1590-5.

[94] F. Torres, S. Puente und C. Diaz, 2009, „Automatic cooperative disassembly robotic system: Task planner to distribute tasks among robots," *Control Engineering Practice*, Bd. 17, Nr. 1, pp. 112-121, DOI 10.1016/j.conengprac.2008.05.013.

[95] D. Baca, C. Soledad, S. T. P. Méndez und F. T. Medina, 2006, „Task planner for a cooperative disassembly robotic system," *IFAC Proceedings Volumes*, Bd. 39, Nr. 3, pp. 163-168, DOI 10.3182/20060517-3-FR-2903.00099.

[96] S. Vongbunyong, S. Kara und M. Pagnucco, 2013, „Basic behaviour control of the vision-based cognitive robotic disassembly automation," *Assembly Automation*, Bd. 33, Nr. 1, pp. 38-56, DOI 10.1108/01445151311294694.

[97] S. Vongbunyong und W. H. Chen, 2015, *Disassembly Automation- Automated Systems with Cognitive Abilities*, Cham: Springer, 1. Auflage, ISBN 978-3-319-15182-3.

[98] J. McCarty, 1963, „*Situations, Actions, and Causal Laws*," Stanford Artifical Intelligence Project, Stanford University.

[99] T. C. Lueth und U. Rembold, 1994, „Extensive Manipulation Capabilities and Reliable Behavior at Autonomous Robot Assembly," *IEEE International Conference on Robotics and Automation*, pp. 3495-3500, DOI 10.1109/ROBOT.1994.351033.

[100] T. Längle, T. Lueth, G. Herzog, E. Stopp und G. Kamstrup, 1995, „KANTRA - A natural language interface for intelligent robots," *Proceedings of the International Conference in Intelligent Autonomous Systems*, pp. 357-364, DOI 10.1109/ROMAN.1994.365947.

[101] T. Laengle, T. C. Lueth und U. Rembold, 1997, „A distributed Control Architecture for autonomous Robot Systems," *Series in Machine Perception and Artificial Intelligence*, Bd. 21, pp. 384-402, DOI 10.1142/9789812797773_0023.

[102] B. Hildebrandt, A. Knoll, C. Scheering und J. Zhang, 1999, „Ein situierter künstlicher Kommunikator für Konstruktionsaufgaben," *Kognitionswissenschaften*, Bd. 8, Nr. 3, pp. 137-142, DOI 10.1007/s001970050083.

[103] Y. O. v. Collani, 2001, *Repräsentation und Generalisierung von diskreten Ereignisabläufen in Abhängigkeit von Multisensormustern*, Dissertation, Technische Fakultät, Universität Bielefeld, urn:nbn:de:hbz:361-1657.

[104] M. C. Ferch, 2001, *Lernen von Montagestrategien in einer verteilten Multiroboterumgebung*, Dissertation, Technische Fakultät, Universität Bielefeld, urn:nbn:de:hbz:361-3722.

[105] S. G. Kaufman, R. Wilson, R. Jones, T. Calton und A. Ames, 1996, „The Archimedes 2 Mechanical Assembly Planning System," *IEEE International Conference on Robotics and Automation*, pp. 3361-3368, DOI 10.1109/ROBOT.1996.509225.

[106] U. Thomas, 2008, *Automatisierte Programmierung von Robotern für Montageaufgaben*, Dissertation, Carl-Friedrich-Gauß-Fakultät, Technische Universität Braunschweig, Aachen: Shaker, ISBN 978-3-8322-7101-5.

[107] F. Röhrdanz, H. Mosemann und F. Wahl, 1997, „Generating and evaluating stable assembly sequences," *Advanced Robotics*, Bd. 11, Nr. 2, pp. 97-126, DOI 10.1163/156855397X00272.

[108] H. Mosemann, 2000, *Beiträge zur Planung, Dekomposition und Ausführung von automatisch generierten Roboteraufgaben*, Dissertation, Carl-Friedrich-Gauß-Fakultät, Technische Universität Braunschweig, Aachen: Shaker, ISBN 3-8265-7710-8.

[109] U. Thomas und F. M. Wahl, 2010, „Assembly Planning and Task Planning- Two Prerequisites for Automated Robot Programming," in *Robotics Systems for Handling and Assembly*, B. Siciliano, O. Khatib und F. Groen, Hrsg., Berlin: Springer, pp. 333-354, DOI 10.1007/978-3-642-16785-0_19.

[110] U. Thomas, T. Stouraitis und M. A. Roa, 2015, „Flexible Assembly through Integrated Assembly Sequence Planning and Grasp Planning," *IEEE International Conference on Automation Science and Engineering*, pp. 586-592, DOI 10.1109/CoASE.2015.7294142.

[111] C. Connette, 2013, *Kinematische Modellierung und Regelung omnidirektionaler, nicht-holonomer Fahrwerke*, Dissertation, Fakultät für Konstruktions-, Produktions- und Fahrzeugtechnik, Universität Stuttgart, Stuttgart: Fraunhofer Verlag, ISBN 978-3-8396-0564-6.

[112] J. J. Craig, 2005, *Introduction to Robotics: Mechanics and Control*, Pearson Prentice Hall, 3. Auflage, ISBN 0-13-123629-6.

[113] M. W. Spong, S. Hutchinson und M. Vidyasagar, 2006, *Robot Modeling and Control*, John Wiley & Sons, Inc., 1. Auflage, ISBN 978-0471649908.

[114] B. Siciliano, L. Sciavicco, L. Villani und G. Oriolo, 2009, *Robotics- Modeling, Planning and Control*, London: Springer, 1. Auflage, ISBN 978-1-84628-641-4.

[115] K. Fu, R. Gonzalez und C. Lee, 1987, *Robotics: Control, Sensing, Vision, and Intelligence*, New York: McGraw-Hill Book Company, 1. Auflage, ISBN 978-0071004213.

[116] H. R. Everett, 1995, *Sensors for mobile robots- Theory and application*, Wellesley: A K Peters, Ltd., ISBN 978-1568810485.

[117] J. Borenstein, H. R. Everett und L. Feng, 1996, *Where am I? Sensors and Methods for mobile robot positioning*, Michigan: Eigendruck University of Michigan.

[118] R. B. Fisher und K. Konolige, 2008, „Range Sensors," in *Handbook of Robotics*, B. Siciliano und O. Khatib, Hrsg., Berlin: Springer, 1. Auflage, pp. 521-542, ISBN 978-3-540-30301-5.

[119] G. Arbeiter, 2013, *3-D-Umgebungserfassung für teil-autonome mobile Roboter*, Dissertation, Fakultät Konstruktions-, Produktions- und Fahrzeugtechnik, Universität Stuttgart, Stuttgart: Fraunhofer Verlag, ISBN 978-3-8396-0674-2.

[120] X. Ren, D. Fox und K. Konolige, 2013, „Change their Perception: RGB-D Cameras for 3-D Modeling and Recognition," *IEEE Robotics & Automation Magazine*, Bd. 20, Nr. 4, pp. 49-59, DOI 10.1109/MRA.2013.2253409.

[121] R. S. Dahiya, G. Metta und M. Valle, 2010, „Tactile Sensing—From Humans to Humanoids,“ *IEEE Transactions on Robotics*, Bd. 26, Nr. 1, pp. 1-20, DOI 10.1109/TRO.2009.2033627.

[122] M. R. Cutkosky, R. D. Howe und W. R. Provancher, 2008, „Force and Tactile Sensors,“ in *Handbook of Robotics*, B. Siciliano und O. Khatib, Hrsg., Berlin: Springer, 1. Auflage, pp. 455-476, ISBN 978-3-540-30301-5.

[123] o.V., 2017, „2D-LiDAR-Sensoren,“ SICK AG, [Online]. Available: https://www.sick.com/de/de/mess-und-detektionsloesungen/2d-laserscanner/lms4xx/lms400-1000/p/p109851. [Zugriff am 18 09 2017].

[124] o.V., 2017, „DFK 38UX267- USB 3.0 color industrial camera,“ Imaging Source, [Online]. Available: https://www.theimagingsource.com/products/industrial-cameras/usb-3.0-color/dfk38ux267/. [Zugriff am 18 09 2017].

[125] H. Gieselmann, 2010, „Heise,“ Microsoft Kinect, [Online]. Available: http://www.heise.de/newsticker/meldung/Microsoft-bietet-Kinect-Sensor-fuer-150-Dollar-an-1026954.html. [Zugriff am 18 09 2017].

[126] o.V., 2017, „Präzisions-Zug-Druck-Kraftsensor,“ MTS Messtechnik Schaffhausen GmbH, [Online]. Available: http://www.mts.ch/documents/8524_DE.pdf. [Zugriff am 18 09 2017].

[127] o.V., 2017, „Weiss Robotics GmbH & Co. KG,“ [Online]. Available: http://www.eu-robotics-sme.org/Weiss_files/weiss_robotics_wts.jpg. [Zugriff am 18 09 2017].

[128] Schunk, 2015, „Automation.at,“ Schunk, [Online]. Available: http://www.automation.at/detail/extrem-vielseitiger-kraft-momenten-sensor_23718. [Zugriff am 23 05 2015].

[129] R. E. Kalman, 1960, „A new approach to linear filtering and prediction problems,“ *Journal of Basic Engineering*, Bd. 82, Nr. 1, pp. 35-45.

[130] S. J. Julier, J. K. Uhlmann und H. F. Durrant-Whyte, 1995, „A new approach for filtering nonlinear systems,“ *IEEE American Control Conference*, pp. 1628-1632, DOI 10.1109/ACC.1995.529783.

[131] S. J. Julier und J. K. Uhlmann, 2004, „Unscented Filtering and Nonlinear Estimation,“ *Proceedings of the IEEE*, Bd. 92, Nr. 3, pp. 401-422, DOI 10.1109/JPROC.2003.823141.

[132] A. Knoll, 1994, „*Sensordatenfusion für Anwendungen in der Robotik*,“ KI Frühjahrsschule der GI.

[133] B. V. Dasarathy, 1997, „Sensor Fusion Potential Exploitation- Innovative Architectures and Illustrative Applications,“ *Proceedings of the IEEE*, Bd. 85, Nr. 1, pp. 24-38, DOI 10.1109/5.554206.

[134] J. Baeten und J. D. Schutter, 2002, „Hybrid Vision/Force Control at Corners in Planar Robotic-Contour Following,“ *IEEE/ASME Transactions on Mechatronics*, Bd. 7, Nr. 2, pp. 143-151, DOI 10.1109/TMECH.2002.1011251.

[135] Y. Zhou, B. J. Nelson und B. Vikramaditya, 2000, „Integrating Optical Force Sensing with Visual Servoing for Microassembly," *Journal of Intelligent and Robotic Systems*, Bd. 28, Nr. 3, pp. 259-276, DOI 10.1023/A:1008136711577.

[136] H. Durrant-Whyte und T. C. Henderson, 2008, „Multisensor Data Fusion," in *Handbook of Robotics*, B. Siciliano und O. Khatib, Hrsg., Berlin: Springer, 1. Auflage, pp. 585-610, ISBN 978-3-540-30301-5.

[137] B. Khaleghi, A. Khamis, F. O. Karray und S. N. Razavi, 2013, „Multisensor data fusion: A review of the state-of-the-art," *Information Fusion*, Bd. 14, Nr. 1, pp. 28-44, DOI 10.1016/j.inffus.2011.08.001.

[138] N. J. Nilsson, 1984, „*Shakey the robot*," Artifical Intelligence Center, Computer Science and Technology Division, SRI, Stanford.

[139] N. J. Nilsson, 1982, *Principles of Artifical Intelligence*, Berlin: Springer, 1. Auflage, ISBN 978-3-540-11340-9.

[140] R. A. Brooks, 1995, „Intelligence without reason," in *The artificial life route to artificial intelligence: Building embodied, situated agents*, L. Erlbaum Associates, pp. 569-595.

[141] R. Simmons, 1994, „Structured Control for Autonomous Robots," *IEEE Transactions on Robotics and Automation*, Bd. 10, Nr. 1, pp. 34-43, DOI 10.1109/70.285583.

[142] R. A. Brooks, 1986, „A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation*, Bd. 2, Nr. 1, pp. 14-23, March DOI 10.1109/JRA.1986.1087032.

[143] R. C. Arkin, 1998, *Behavior-based robotics*, Cambridge: MIT press, 3. Auflage, ISBN 978-0262011655.

[144] S. Russell und P. Norvig, 2012, *Künstliche Intelligenz- Ein moderner Ansatz*, München: Pearson Deutschland GmbH, 3. Auflage, ISBN 978-3868940985.

[145] C. Martens und A. Gräser, 2003, „Reaktive Kontrolle komplexer autonomer Systeme in einer agentenorientierten Mehrschichtenarchitektur," *at-Automatisierungstechnik*, Bd. 51, Nr. 3, pp. 101-112, DOI 10.1524/auto.51.3.101.20932.

[146] B. Finkemeyer, 2004, *Robotersteuerungsarchitektur auf Basis von Aktionsprimitiven*, Dissertation, Carl-Friedrich-Gauß-Fakultät, Technische Universität Braunschweig, Aachen: Shaker, ISBN 3-8322-2893-4.

[147] U. Thomas, B. Finkemeyer, T. Kröger und F. M. Wahl, 2003, „Error-Tolerant Execution of Complex Robot Tasks Based on Skill Primitives," *IEEE International Conference on Robotics and Automation*, pp. 3069-3075, DOI 10.1109/ROBOT.2003.1242062.

[148] B. Finkemeyer, T. Kröger und F. Wahl, 2005, „Aktionsprimitive: Ein universelles Roboter-Programmierparadigma," *at- Automatisierungstechnik*, Bd. 53, Nr. 4-5, pp. 189-196, DOI 10.1524/auto.53.4.189.62256.

[149] M. T. Mason, 1981, „Compliance and force control for computer controlled manipulators," *IEEE Transactions on Systems, Man and Cybernetics*, Bd. 11, Nr. 6, pp. 418-432, DOI 10.1109/TSMC.1981.4308708.

[150] G. Milighetti, 2010, *Multisensorielle diskret-kontinuierliche Überwachung und Regelung humanoider Roboter*, Dissertation, Fakultät für Informatik, Karlsruher Institut für Technologie, Karlsruhe: KIT Scientific Publishing, ISBN 978-3-86644-568-0.

[151] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler und A. Ng, 2009, „ROS: an open-source Robot Operating System," *ICRA workshop on open source software*, Bd. 3, Nr. 2, pp. 1-6.

[152] K. Regenstein, 2010, *Modulare, verteilte Hardware-Software-Architektur für humanoide Roboter*, Dissertation, Fakultät für Informatik, Karlsruher Institut für Technologie, KIT Scientific Publishing, ISBN 978-3-86644-527-7.

[153] R. Dillmann, 2014, „*Umweltmodellierung*," Vorlesung zur Robotik 3- Sensoren in der Robotik, Karlsruher Institut für Technologie.

[154] D. Kortenkamp und T. Weymouth, 1994, „Topological mapping for mobile robots using a combination of sonar and vision sensing," *Proceedings of the twelfth national Conference on Artificial Intelligence*, pp. 979-984, ISBN 0-262-61102-3.

[155] A. Nüchter und J. Hertzberg, 2008, „Towards semantic maps for mobile robots," *Robotics and Autonomous Systems*, Bd. 56, Nr. 11, pp. 915-926, DOI 10.1016/j.robot.2008.08.001.

[156] R. B. Rusu, Z. C. Marton, N. Blodow, A. Holzbach und M. Beetz, 2009, „Model-based and Learned Semantic Object Labeling in 3D Point Cloud Maps of Kitchen Environments," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3601-3608, DOI 10.1109/IROS.2009.5354759.

[157] R. B. Rusu, 2009, *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*, Dissertation, Fakultät für Informatik, Technische Universität München, http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20091006-800632-1-6.

[158] J. Hertzberg, K. Lingemann und A. Nüchter, 2012, *Mobile Roboter- Eine Einführung aus Sicht der Informatik*, Berlin: Springer Vieweg, 1. Auflage, ISBN 978-3-642-01726-1.

[159] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss und W. Burgard, 2013, „OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on octrees," *Autonomous Robots*, Bd. 34, Nr. 3, pp. 189-206, DOI 10.1007/s10514-012-9321-0.

[160] A. J. Davison, I. D. Reid, N. D. Molton und O. Stasse, 2007, „MonoSLAM: Real-Time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 29, Nr. 6, pp. 1052-1067, DOI 10.1109/TPAMI.2007.1049.

[161] W. Y. Jeong und K. M. Lee, 2006, „Visual SLAM with Line and Corner Features," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2570-2575, DOI 10.1109/IROS.2006.281708.

[162] A. Nüchter, K. Lingemann, J. Hertzberg und H. Surmann, 2005, „6D SLAM with Approximate Data Association," *IEEE Proceedings of the 12th International Conference on Advanced Robotics*, pp. 242-249, DOI 10.1109/ICAR.2005.1507419.

[163] P. Henry, M. Krainin, E. Herbst, X. Ren und D. Fox, 2012, „RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments," *The International Journal of Robotics Research*, Bd. 31, Nr. 5, pp. 647-663, DOI 10.1177/0278364911434148.

[164] F. Endres, J. Hess, J. Sturm, D. Cremers und W. Burgard, 2014, „3-D Mapping With an RGB-D Camera," *IEEE Transactions on Robotics*, Bd. 30, Nr. 1, pp. 177-187, DOI 10.1109/TRO.2013.2279412.

[165] M. Fischler und R. Bolles, 1981, „Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, Bd. 24, Nr. 6, pp. 381-395, DOI 10.1145/358669.358692.

[166] R. I. Hartley, 1997, „In Defense of the Eight-Point Algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 19, Nr. 6, pp. 580-593, DOI 10.1109/34.601246.

[167] P. J. Besl und N. D. McKay, 1992, „A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 14, Nr. 2, pp. 239-256, DOI 10.1109/34.121791.

[168] A. Segal, D. Haehnel und S. Thrun, 2009, „Generalized-ICP," *Robotics: Science and Systems*, Bd. 2, Nr. 4, pp. 435-442, DOI 10.15607/RSS.2009.V.021.

[169] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige und W. Burgard, 2011, „g²o: A General Framework for Graph Optimization," *IEEE International Conference on Robotics and Automation*, pp. 3607-3613, DOI 10.1109/ICRA.2011.5979949.

[170] S. Thrun, W. Burgard und D. Fox, 2006, *Probabilistic Robotics*, Cambridge: The MIT Press, 1. Auflage, ISBN 978-0262201629.

[171] H. Süße und E. Rodner, 2014, *Bildverarbeitung und Objekterkennung*, Wiesbaden: Springer Vieweg, 1. Auflage, ISBN 978-3-8348-2605-3.

[172] T. Deselaers, 2008, „Bildsuche, Objekterkennung und Diskriminative Modelle," in *Lecture Notes in Informatics*, D. Wagner, Hrsg., Gesellschaft für Informatik, pp. 51-60.

[173] P. Azad, 2008, *Visual Perception for Manipulation and Imitation in Humanoid Robots*, Dissertation, Fakultät für Informatik, Universität Karlsruhe, DOI 10.5445/IR/1000011294.

[174] D. G. Lowe, 1999, „Object Recognition from Local Scale-Invariant Features," *IEEE International Conference on Computer Vision*, pp. 1150-1157, DOI 10.1109/ICCV.1999.790410.

[175] D. G. Lowe, 2004, „Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, Bd. 60, Nr. 2, pp. 91-110, DOI 10.1023/B:VISI.0000029664.99615.94.

[176] H. Bay, A. Ess, T. Tuytelaars und L. V. Gool, 2008, „Speeded-Up Robust Features (SURF)," *Computer Vision and Image Understanding*, Bd. 110, Nr. 3, pp. 346-359, DOI 10.1016/j.cviu.2007.09.014.

[177] N. Dalal und B. Triggs, 2005, „Histograms of oriented gradients for human detection," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 886-893, DOI 10.1109/CVPR.2005.177.

[178] P. Viola und M. Jones, 2001, „Rapid Object Detection using a Boosted Cascade of Simple Features," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 511-518, DOI 10.1109/CVPR.2001.990517.

[179] S. Leutenegger, M. Chli und R. Y. Siegwart, 2011, „BRISK: Binary Robust Invariant Scalable Keypoints," *IEEE International Conference on Computer Vision*, pp. 2548-2555, DOI 10.1109/ICCV.2011.6126542.

[180] K. Mikolajczyk und C. Schmid, 2005, „A Performance Evaluation of Local Descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 27, Nr. 10, pp. 1615-1630, DOI 10.1109/TPAMI.2005.188.

[181] R. B. Rusu, N. Blodow und M. Beetz, 2009, „Fast point feature histograms (FPFH) for 3D registration," *IEEE International Conference on Robotics and Automation*, pp. 3212-3217, DOI 10.1109/ROBOT.2009.5152473.

[182] F. Tombari, S. Salti und L. D. Stefano, 2010, „Unique Signatures of Histograms for Local Surface Description," *European conference on computer vision*, pp. 356-369, DOI 10.1007/978-3-642-15558-1_26.

[183] F. Tombari, S. Salti und L. D. Stefano, 2011, „A combined Texture-shape Descriptor for enhanced 3D Feature Matching," *IEEE International Conference on Image Processing*, pp. 809-812, DOI 10.1109/ICIP.2011.6116679.

[184] L. A. Alexandre, 2012, „3D Descriptors for Object and Category Recognition: a Comparative evaluation," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1-6.

[185] G. Csurka, C. R. Dance, L. Fan, J. Willamowski und C. Bray, 2004, „Visual Categorization with Bags of Keypoints," *Workshop on statistical learning in computer vision*, pp. 1-22, DOI 10.1.1.72.604.

[186] C. M. Bishop, 2006, *Pattern Recognition and Machine Learning*, Singapore: Springer, 1. Auflage, ISBN 978-0387-31073-2.

[187] S. Lazebnik, C. Schmid und J. Ponce, 2006, „Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2169-2178, DOI 10.1109/CVPR.2006.68.

[188] A. Krizhevsky, I. Sutskever und G. E. Hinton, 2012, „Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, Bd. 25, Nr. 2, pp. 1097-1105, DOI 10.1145/3065386.

[189] M. Ulrich, C. Wiedemann und C. Steger, 2009, „CAD-based recognition of 3D objects in monocular images," *IEEE International Confernce on Robotics and Automation*, pp. 1191-1198, DOI 10.1109/ROBOT.2009.5152511.

[190] K. K. W. Lai, 2013, *Object Recognition and Semantic Scene Labeling for RGB-D Data*, PhD Thesis, Computer Science and Engineering, University of Washington.

[191] R. B. Rusu, N. Blodow, Z. Marton, A. Soos und M. Beetz, 2007, „Towards 3D Object Maps for Autonomous Household Robots," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3191-3198, DOI 10.1109/IROS.2007.4399309.

[192] B. Jähne, 2005, *Digitale Bildverarbeitung*, Berlin: Springer, 6. Auflage, ISBN 3-540-24999-0.

[193] N. Otsu, 1979, „A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, Bd. 9, Nr. 2, pp. 62-66, DOI 10.1109/TSMC.1979.4310076.

[194] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez und J. Garcia-Rodriguez, 2017, „*A review on deep learning techniques applied to semantic segmentation*," arXiv preprint arXiv:1704.06857, pp. 1-23.

[195] J. Canny, 1986, „A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 8, Nr. 6, pp. 679-698, DOI 10.1109/TPAMI.1986.4767851.

[196] X. Jiang und H. Bunke, 1999, „Edge Detection in Range Images Based on Scan Line Approximation," *Computer Vision and Image Understanding*, Bd. 73, Nr. 2, pp. 183-199, DOI 10.1006/cviu.1998.0715.

[197] R. Adams und L. Bischof, 1994, „Seeded region growing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 16, Nr. 6, pp. 641-647, DOI 10.1109/34.295913.

[198] M. Ester, H.-P. Kriegel, J. Sanders und X. Xu, 1996, „A Density-Based Algorithm for Discovering Clusters in Large Databases with Noise," *International Conference on Knowledge Discovery and Data Mining*, pp. 226-231.

[199] J. Sander, M. Ester, H.-P. Kriegel und X. Xu, 1998, „Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications," *Data mining and knowledge discovery*, Bd. 2, Nr. 2, pp. 169-194, DOI 10.1023/A:1009745219419.

[200] R. B. Rusu, N. Blodow, Z. C. Marton und M. Beetz, 2009, „Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1-6, DOI 10.1109/IROS.2009.5354683.

[201] P. H. Torr und A. Zisserman, 2000, „MLESAC: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, Bd. 78, Nr. 1, pp. 138-156, DOI 10.1006/cviu.1999.0832.

[202] Matlab, 2016, „Fit sphere to 3-D point cloud- pcfitsphere," Mathworks, [Online]. Available: http://de.mathworks.com/help/vision/ref/pcfitsphere.html. [Zugriff am 10 01 2016].

[203] E. Alpaydin, 2008, *Maschinelles Lernen*, München: Oldenbourg Wissenschaftsverlag, 1. Auflage, ISBN 978-3486581140.

[204] D. F. Dementhon und L. S. Davis, 1995, „Model-Based Object Pose in 25 Lines of Code," *International Journal of Computer Vision*, Bd. 15, Nr. 1, pp. 123-141, DOI 10.1007/BF01450852.

[205] A. Collet, M. Martinez und S. S. Srinivasa, 2011, „The MOPED framework: Object recognition and pose estimation for manipulation," *The International Journal of Robotics Research*, Bd. 30, Nr. 10, pp. 1284-1306, DOI 10.1177/0278364911401765.

[206] Z.-C. Marton, D. Pangercic, N. Blodow und M. Beetz, 2011, „Combined 2D–3D categorization and classification for multimodal perception systems," *The International Journal of Robotics Research*, Bd. 30, Nr. 11, pp. 1378-1402, DOI 10.1177/0278364911415897.

[207] G. Arbeiter, J. Fischer und A. Verl, 2010, „3D Perception and Modeling for Manipulation on Care-O-bot® 3," *IEEE International Conference on Robotics and Automation*, pp. 1-5.

[208] J. Fischer, 2015, *A user-oriented, comprehensive system for the 6 DoF recognition of arbitrary rigid household objects*, Dissertation, Fakultät Konstruktions- Produktions- und Fahrzeugtechnik, Universität Stuttgart, Stuttgart: Fraunhofer Verlag, ISBN 978-3-8396-0891-3.

[209] R. B. Rusu und S. Cousins, 2011, „3D is here: Point Cloud Library (PCL)," *IEEE International Conference on Robotics and Automation*, pp. 1-4, DOI 10.1109/ICRA.2011.5980567.

[210] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli und M. Vincze, 2012, „Point Cloud Library- Three-Dimensional Object Recognition and 6DoF Pose Estimation," *IEEE Robotics & Automation Magazine*, Bd. 19, Nr. 3, pp. 80-91, DOI 10.1109/MRA.2012.2206675.

[211] C. Connolly, 1985, „The Determination of next best views," *IEEE International Conference on Robotics and Automation*, pp. 432-435, DOI 10.1109/ROBOT.1985.1087372.

[212] M. Suppa, 2007, *Autonomous Robot Work Cell Exploration using Multisensory Eye-in-Hand Systems*, Dissertation, Fakultät für Maschinenbau, Gottfried Wilhelm Leibniz Universität Hannover.

[213] S. Wenhardt, B. Deutsch, E. Angelopoulou und H. Niemann, 2007, „Active visual object reconstruction using D-, E-, and T-optimal next best views," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1-7, DOI 10.1109/CVPR.2007.383363.

[214] F. Deinzer, C. Derichs und H. Niemann, 2006, „Integrated Viewpoint Fusion and Viewpoint Selection for Optimal Object Recognition," *Proceedings of the British Machine Vision Conference*, pp. 287-296, DOI 10.5244/C.20.30.

[215] S. Kriegel, M. Brucker, Z.-C. Marton, T. Bodenmüller und M. Suppa, 2013, „Combining object modeling and recognition for active scene exploration," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2384-2391, DOI 10.1109/IROS.2013.6696691.

[216] S. Kriegel, C. Rink, T. Bodenmüller, A. Narr, M. Suppa und G. Hirzinger, 2012, „Next-best-scan planning for autonomous 3D modeling," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2850-2856, DOI 10.1109/IROS.2012.6385624.

[217] R. Monica, J. Aleotti und S. Caselli, 2016, „A KinFu based approach for robot spatial attention and view planning," *Robotics and Autonomous Systems*, Bd. 75, pp. 627-640, DOI 10.1016/j.robot.2015.09.010.

[218] F. Bissmarck, M. Svensson und G. Tolt, 2015, „Efficient algorithms for Next Best View evaluation," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, p. 5876–5883, DOI 10.1109/IROS.2015.7354212.

[219] U. Reimer, 1991, *Einführung in die Wissensrepräsentation: netzartige und schema-basierte Repräsentationsformate*, Stuttgart: B. G. Teubner, 1. Auflage, ISBN 3-519-02241-9.

[220] J. Lunze, 2010, *Künstliche Intelligenz für Ingenieure*, München: Oldenbourg Wissenschaftsverlag, 2. Auflage, ISBN 978-486-70222-4.

[221] M. Minsky, 1974, „*A Framework for Representing Knowledge*," MIT-AI Laboratory Memo 306.

[222] B. Owsnicki-Klewe, K. v. Luck und B. Nebel, 2003, „Wissensrepräsentation und Logik-Eine Einführung," in *Handbuch der künstlichen Intelligenz*, G. Görz, C. Rollinger und J. Schneeberger, Hrsg., München: Oldenbourg Wissenschaftsverlag, 4. Auflage, pp. 153-197, ISBN 3-486-27212-8.

[223] M. Tenorth und M. Beetz, 2013, „KnowRob: A knowledge processing infrastructure for cognition-enabled robots," *The International Journal of Robotics Research*, Bd. 32, Nr. 5, pp. 566-590, DOI 10.1177/0278364913481635.

[224] P. Levi, 1988, *Planen für autonome Montageroboter*, Berlin: Springer, 1. Auflage, ISBN 978-3-642-74268-2.

[225] H.-J. Siegert und S. Bocionek, 1996, *Robotik: Programmierung intelligenter Roboter*, Berlin: Springer, 1. Auflage, ISBN 978-3-642-80067-2.

[226] M. Ghallab, D. Nau und P. Traverso, 2004, *Automated Planning- Theory and Practice*, San Francisco: Elsevier, 1. Auflage, ISBN 1-55860-856-7.

[227] P. Jiménez, 2013, „Survey on assembly sequencing: a combinatorial and geometrical," *Journal of Intelligent Manufacturing*, Bd. 24, Nr. 2, pp. 235-250, DOI 10.1007/s10845-011-0578-5.

[228] G. Görz, C.-R. Rollinger und J. Schneeberger, 2003, *Handbuch der künstlichen Intelligenz*, München: Oldenbourg Wissenschaftsverlag, 4. Auflage.

[229] Y. Tang, M. Zhou, E. Zussman und R. Caudill, 2000, „Disassembly Modeling, Planning, and Application: A Review," *IEEE International Conference on Robotics and Automation*, pp. 2197-2202, DOI 10.1109/ROBOT.2000.846354.

[230] S. Ghandi und E. Masehian, 2015, „Review and taxonomies of assembly and disassembly path planning problems and approaches," *Computer-Aided Design*, Bd. 67, pp. 58-86, DOI 10.1016/j.cad.2015.05.001.

[231] A. J. Lambert, 2003, „Disassembly sequencing: a survey," *International Journal of Production Research*, Bd. 41, Nr. 16, pp. 3721-3759, DOI 10.1080/0020754031000120078.

[232] J.-G. Kang und P. Xirouchakis, 2006, „Disassembly sequencing for maintenance: A survey," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, Bd. 220, Nr. 10, pp. 1697-1716, DOI 10.1243/09544054JEM596.

[233] A. J. D. Lambert und S. M. Gupta, 2005, *Disassembly Modeling for Assembly, Maintenance, Reuse and Recycling*, London: CRC Press, 1. Auflage, ISBN 1-57444-334-8.

[234] B. H. Krogh und A. C. Sanderson, 1985, „*Modeling and control of assembly tasks and systems*," Research Paper, Carnegie Mellon University.

[235] A. Delchambre, 1992, *Computer-aided Assembly Planning*, Dordrecht: Springer Science+Business Media Dordrecht, 1. Auflage, ISBN 978-94-010-5025-8.

[236] A. P. Ambler und R. J. Popplestone, 1975, „Inferring the Positions of Bodies from Specified Spatial Relationships," *Artificial Intelligence*, Bd. 6, Nr. 2, pp. 157-174, DOI 10.1016/0004-3702(75)90007-7.

[237] M. Wiesbeck, 2014, *Struktur zur Repräsentation von Montagesequenzen für die situationsorientierte Werkerführung*, Dissertation, Fakultät für Maschinenwesen, Technische Universität München, München: Herbert Utz Verlag, ISBN 978-3-8316-4369-1.

[238] L. S. H. De Mello und A. C. Sanderson, 1990, „AND/OR Graph Representation of Assembly Plans," *IEEE Transactions on Robotics and Automation*, Bd. 6, Nr. 2, pp. 188-199, DOI 10.1109/70.54734.

[239] T. L. D. Fazio und D. E. Whitney, 1987, „Simplified Generation of All Mechanical Assembly Sequences," *IEEE Journal of Robotics and Automation*, Bd. 3, Nr. 6, pp. 640-658, DOI 10.1109/JRA.1987.1087132.

[240] K. E. Moore, A. Gungor und S. M. Gupta, 1998, „Disassembly Process Planning Using Petri Nets," *IEEE Symposium on Electronics and the Environment*, pp. 88-93, DOI 10.1109/ISEE.1998.675037.

[241] K. E. Moore, A. Gungor und S. M. Gupta, 2001, „Petri net approach to disassembly process planning for products with complex AND/OR precedence relationships," *European Journal of Operational Research*, Bd. 135, Nr. 2, pp. 428-449, DOI 10.1016/S0377-2217(00)00321-0.

[242] L. S. H. De Mello und A. C. Sanderson, 1991, „Representations of mechanical assembly sequences," *IEEE Transactions on Robotics and Automation*, Bd. 7, Nr. 2, pp. 211-227, DOI 10.1109/70.75904.

[243] A. Gungor und S. M. Gupta, 1998, „*Disassembly sequence planning for complete disassembly in product recovery*,“ Department of Mechanical and Industrial, Northeastern University.

[244] L. E. Kavraki und M. N. Kolountzakis, 1995, „Partitioning a planar assembly into two connected parts is NP-complete,“ *Information Processing Letters*, Bd. 55, Nr. 3, pp. 159-165, DOI 10.1016/0020-0190(95)00083-O.

[245] A. Bourjault, 1988, „*Methology of Assembly Automation: A New Approach,“ in Robotics and Factories of the Future*, Springer, pp. 37-45, DOI 10.1007/978-3-642-73890-6_6.

[246] J. D. Wolter, 1991, „A Combinatorial Analysis of Enumerative Data Structures for Assembly Planning,“ *IEEE International Conference on Robotics and Automation*, pp. 611-618, DOI 10.1109/ROBOT.1991.131649.

[247] R. H. Wilson und J. C. Latombe, 1992, „Geometric Reasoning about Mechanical Assembly,“ *Artificial Intelligence*, Bd. 71, Nr. 2, pp. 371-396, DOI 10.1016/0004-3702(94)90048-5.

[248] B. Romney, C. Godard, M. Goldwasser und G. Ramkumar, 1995, „An Efficient System for Geometric Assembly Sequence Generation and Evaluation,“ *ASME Computers in Engineering*, pp. 699-712, DOI 10.1.1.131.8385.

[249] A. Gungor und S. M. Gupta, 1998, „Disassembly Sequence Planning for Products with Defective Parts in Product Recovery,“ *Computers and Industrial Engineering*, Bd. 35, Nr. 1-2, pp. 161-164, DOI 10.1016/S0360-8352(98)00047-3.

[250] J. M. Henrioud und A. Bourjault, 1992, „Computer Aided Assembly Process Planning,“ *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, Bd. 206, Nr. 1, pp. 61-66, DOI 10.1243/PIME_PROC_1992_206_056_02.

[251] L. S. H. De Mello und A. C. Sanderson, 1991, „A correct and complete algorithm for the generation of mechanical assembly sequences,“ *IEEE Transactions on Robotics and Automation*, Bd. 7, Nr. 2, pp. 228-240, DOI 10.1109/70.75905.

[252] H. Mosemann und F. M. Wahl, 2001, „Automatic Decomposition of Planned Assembly Sequences into Skill Primitives,“ *IEEE Transactions on Robotics and Automation*, Bd. 17, Nr. 5, pp. 709-718, DOI 10.1109/70.964670.

[253] F. Röhrdanz, H. Mosemann und F. M. Wahl, 1996, „HighLAP: A High Level System for Generating, Representing, and Evaluating Assembly Sequences,“ *IEEE International Joint Symposia on Intelligence and Systems*, pp. 134-141, DOI 10.1109/IJSIS.1996.565061.

[254] L. S. H. De Mello, 1989, *Task Sequence Planning for Robotic Assembly*, PhD Thesis, Carnegie Mellon University Pittsburgh, PA, USA.

[255] R. Gutsche, F. Röhrdanz und F. M. Wahl, 1995, „Assembly Planning Using Symbolic Spatial Relationships,“ in *Graphics and Robotics*, W. Straßer und F. M. Wahl, Hrsg., Springer, pp. 87-114, DOI 10.1007/978-3-642-79210-6_6.

[256] R. J. Popplestone, 1979, „Specifying Manipulations in Terms of Spatial Relationships," *International Seminar on Programming Methods and Language for Industrial Robots IRIA*, pp. 1-17.

[257] U. Thomas, M. Barrenscheen und F. M. Wahl, 2003, „Efficient assembly sequence planning using stereographical projections of c-space obstacles," *IEEE International Symposium on Assembly and Task Planning*, pp. 96-102, DOI 10.1109/ISATP.2003.1217194.

[258] A. Csiszar, 2017, *Online Path Planning for Industrial Robots with integrated Workspace Limits and Safety Criterion*, Dissertation, Fakultät Konstruktions-, Produktions- und Fahrzeugtechnik, Universität Stuttgart, http://dx.doi.org/10.18419/opus-9367.

[259] J.-C. Latombe, 1993, *Robot Motion Planning*, Kluwer Academic Publishers, 2. Auflage, ISBN 9780792392064.

[260] S. M. LaValle, 2006, *Planning Algorithms*, Cambridge University Press, 1. Auflage, ISBN 978-0521862059.

[261] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki und S. Thrun, 2005, *Principles of Robot Motion- Theory, Algorithms and Implementation*, MIT Press, 1. Auflage, ISBN 978-0262033275.

[262] S. M. LaValle, 2011, „Motion Planning- Part 1: The Essentials," *IEEE Robotics & Automation Magazine*, Bd. 18, Nr. 1, pp. 79-89 , DOI 10.1109/MRA.2011.940276.

[263] S. M. LaValle, 2011, „Motion Planning- Part 2: Wild Frontiers," *IEEE Robotics & Automation Magazine*, Bd. 18, Nr. 2, pp. 108-118, DOI 0.1109/MRA.2011.941635.

[264] O. Khatib, 1986, „Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *The International Journal of Robotics Research*, Bd. 5, Nr. 1, pp. 90-98, DOI 10.1109/ROBOT.1985.1087247.

[265] S. S. Ge und Y. J. Cui, 2000, „New potential functions for mobile robot path planning," *IEEE Transactions on Robotics and Automation*, Bd. 16, Nr. 5, pp. 615-620, DOI 10.1109/70.880813.

[266] J. Borenstein und Y. Koren, 1991, „The Vector Field Histogramm- Fast Obstacle Avoidance for Mobile Robots," *IEEE Transactions on Robotics and Automation*, Bd. 7, Nr. 3, pp. 278-288, DOI 10.1109/70.88137.

[267] I. Ulrich und J. Borenstein, 1998, „VFH+: Reliable obstacle avoidance for fast mobile robots," *IEEE International Conference on Robotics and Automation*, pp. 1572-1577, DOI 10.1109/ROBOT.1998.677362.

[268] U. Iwan und J. Borenstein, 2000, „VFH*: Local obstacle avoidance with look-ahead verification," *IEEE International Conference on Robotics and Automation*, pp. 2505-2511, DOI 10.1109/ROBOT.2000.846405.

[269] D. Fox, W. Burgard und S. Thrun, 1997, „The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, Bd. 4, Nr. 1, pp. 23-33, DOI 10.1109/100.580977.

[270] M. Seder und I. Petrovic, 2007, „Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," *IEEE International Conference on Robotics and Automation*, pp. 1986-1991, DOI 10.1109/ROBOT.2007.363613.

[271] M. Gao, J. Xu, J. Tian und H. Wu, 2008, „Path Planning for Mobile Robot Based on Chaos Genetic Algorithm," *IEEE Conference on Natural Computation*, pp. 409-413, DOI 10.1109/ICNC.2008.627.

[272] S. M. LaValle, 1998, „*Rapidly-Exploring Random Trees: A new tool for Path Planning*," Department of Computer Science, Iowa State University.

[273] S. M. LaValle und J. J. Kuffner, 2000, „*Rapidly-exploring random trees: Progress and prospects*," Iowa State University.

[274] J. J. Kuffner und S. M. LaValle, 2000, „RRT-connect: An efficient approach to single-query path planning," *IEEE International Conference on Robotics and Automation*, pp. 995-1001, DOI 10.1109/ROBOT.2000.844730.

[275] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli und S. Teller, 2011, „Anytime Motion Planning using the RRT*," *International Conference on Robotics and Automation*, pp. 1478-1483, DOI 10.1109/ICRA.2011.5980479.

[276] J. Bruce und M. Veloso, 2002, „Real-time randomized path planning for robot navigation," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2383-2388, DOI 10.1109/IRDS.2002.1041624.

[277] O. Brock und O. Khatib, 2002, „Elastic Strips: A Framework for Motion Generation in Human Environments," *The International Journal of Robotics Research*, Bd. 21, Nr. 12, pp. 1031-1052, DOI 10.1177/027836490202101200 2.

[278] S. Quinlan und O. Khatib, 1993, „Elastic Bands: Connecting Path Planning and Control," *IEEE International Conference on Robotics and Automation*, pp. 802-807, DOI 10.1109/ROBOT.1993.291936.

[279] M. Huptych, K. Groh und S. Röck, 2011, „Online path planning for industrial robots in varying environments using the curve shortening flow method," *Intelligent Robotics and Applications*, pp. 73-82, DOI 10.1007/978-3-642-25486-4_8.

[280] M. R. Cutkosky, 1989, „On Grasp Choice, Grasp Models, and the Design of Hands for Manufacturing Tasks," *IEEE Transactions on Robotics and Automation*, Bd. 5, Nr. 3, pp. 269-279, DOI 10.1109/70.34763.

[281] J. Romero, 2011, *From Human to Robot Grasping*, PhD Thesis, Computer Science and Communication, KTH Stockholm.

[282] A. T. Miller und P. K. Allen, 2004, „GraspIt! A versatile Simulator for Robotic Grasping," *IEEE Robotics & Automation Magazine*, Bd. 11, Nr. 4, pp. 110-122, DOI 10.1109/MRA.2004.1371616.

[283] J. Bohg, A. Morales, T. Asfour und D. Kragic, 2014, „Data-Driven Grasp Synthesis- A Survey," *IEEE Transactions on Robotics*, Bd. 30, Nr. 2, pp. 289-309, DOI 10.1109/TRO.2013.2289018.

[284] T. Haase, 2011, *Greifplanung und Greifskills für reaktives Greifen*, Dissertation, Fakultät für Informatik, Karlsruher Institut für Technologie, Karlsruhe: KIT Scientifc Publishing, ISBN 978-3-86644-740-0.

[285] N. Vahrenkamp, 2011, *Bewegungsplanung und sensorgestützte Ausführung für das Greifen auf humanoiden Robotern*, Dissertation, Fakultät für Informatik, Karlsruher Institut für Technologie, Karlsruhe: KIT Scientifc Publishing, ISBN 978-3-86644-664-9.

[286] W. T. Townsend, 2000, „The Barrett Hand grasper: programmably flexible part handling and assembly," *Industrial Robot: An International Journal*, Bd. 27, Nr. 3, pp. 181-188, DOI 10.1108/01439910010371597.

[287] A. Verl, D. Fritsch, T. Ledermann und H. Mütherich, 2007, „Greifen und Sehen als Schlüsseltechnologie für Roboter," *wt Werkstatttechnik*, Bd. 97, Nr. 9, pp. 694-699.

[288] C. Borst, M. Fischer und G. Hirzinger, 2003, „Grasping the Dice by Dicing the Grasp," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3692-3697, DOI 10.1109/IROS.2003.1249729.

[289] W. Weber, 2009, *Industrieroboter- Methoden der Steuerung und Regelung*, München: Carl Hanser, 2. Auflage, ISBN 978-3446410312.

[290] M. Weck und C. Brecher, 2006, *Werkzeugmaschinen 4- Automatisierung von Maschinen und Anlagen*, Berlin: Springer Vieweg, 6. Auflage, ISBN 978-3-540-22507-2.

[291] O. Zirn und S. Weikert, 2006, *Modellbildung und Simulation hochdynamischer Fertigungssysteme*, Berlin: Springer, 1. Auflage, ISBN-13 978-3-540-25817-9.

[292] J. Levine, 2009, *Analysis and Control of Nonlinear Systems- A Flatness-based Approach*, Dordrecht: Springer, 1. Auflage, ISBN 978-3-642-00838-2.

[293] R. Rothfuß, J. Rudolph und M. Zeitz, 1997, „Flachheit: Ein neuer Zugang zur Steuerung und Regelung nichtlinearer Systeme," *at-Automatisierungstechnik*, Bd. 45, Nr. 11, pp. 517-525, DOI 10.1524/auto.1997.45.11.517.

[294] W. Chung, L.-C. Fu und S.-H. Hsu, 2008, „Motion Control," in *Handbook of Robotics*, B. Siciliano und O. Khatib, Hrsg., Berlin: Springer, 1. Auflage, pp. 133-159, ISBN 978-3-540-30301-5.

[295] D. E. Whitney, 1987, „Historical Perspective and State of the Art in Robot Force Control," *International Journal of Robotics Research*, Bd. 6, Nr. 1, pp. 3-14, 10.1177/027836498700600101.

[296] A. Winkler, 2006, *Ein Beitrag zur kraftbasierten Mensch-Roboter-Interaktion*, Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität Chemnitz, urn:nbn:de:swb:ch1-200601808.

[297] S. Haddadin, 2011, *Towards Safe Robots: Approaching Asimov's 1st Law*, Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Rheinisch-Westfälische Technische Hochschule Aachen, Berlin: Springer, ISBN 978-3-662-51038-4.

[298] A. Spiller, 2014, *Unterstützung der Werkstückhandhabung kooperierender Industrieroboter durch Kraftregelung*, Dissertation, Fakultät Konstruktions-, Produktions- und Fahrzeugtechnik, Universität Stuttgart, Stuttgart: Fraunhofer Verlag, ISBN 978-3-8396-0711-4.

[299] L. Villani und J. D. Schutter, 2008, „Force Control," in *Handbook of Robotics*, B. Siciliano und O. Khatib, Hrsg., Berlin: Springer, pp. 161-185, ISBN 978-3-540-30301-5.

[300] J. D. Lane, 1980, „Evaluation of a remote center compliance device," *Assembly Automation*, Bd. 1, Nr. 1, pp. 36-46, DOI 10.1108/eb004135.

[301] N. Hogan, 1985, „Impedance Control: An Approach to Manipulation Part 1- Theory," *Journal of dynamic systems, measurement, and control*, Bd. 107, Nr. 1, pp. 1-7, DOI 10.1115/1.3140702.

[302] N. Hogan, 1985, „Impedance Control: An Approach to Manipulation Part 2- Implementation," *Journal of dynamic systems, measurement, and control*, Bd. 107, Nr. 1, pp. 8-16, DOI 10.1115/1.3140713.

[303] N. Hogan, 1985, „Impedance Control: An Approach to Manipulation Part 3- Applications," *Journal of dynamic systems, measurement, and control*, Bd. 107, Nr. 1, pp. 17-24, DOI 10.1115/1.3140701.

[304] S. Chiaverini und L. Sciavicco, 1993, „The Parallel Approach to Force/Position Control of Robotic Manipulators," *IEEE Transactions on Robotics and Automation*, Bd. 9, Nr. 4, pp. 361-373, DOI 10.1109/70.246048.

[305] M. H. Raibert und J. J. Craig, 1981, „Hybrid Position/Force Control of Manipulators," *Journal of Dynamic Systems, Measurement, and Control*, Bd. 103, Nr. 2, pp. 126-133, DOI 10.1115/1.3139652.

[306] T. Yoshikawa, 2000, „Force Control of Robot Manipulators," *IEEE International Conference on Robotics and Automation*, pp. 220-226, DOI 10.1109/ROBOT.2000.844062.

[307] S. Hutchinson, G. D. Hager und P. I. Corke, 1996, „A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, Bd. 12, Nr. 5, pp. 651-670, DOI 10.1109/70.538972.

[308] J. Hill und W. T. Park, 1979, „Real time control of a robot with a mobile camera," *Preceedings 9th ISIR*, pp. 233- 246.

[309] D. Kragic und H. I. Christensen, 2002, „*Survey on visual servoing for manipulation*," Centre for Autonomous Systems, Numerical Analysis and Computer Science, KTH Stockholm.

[310] A. C. Sanderson und L. E. Weiss, 1983, „Image-based visual servo control of robots," *26th Annual Technical Symposium. International Society for Optics and Photonics*, pp. 164- 169, DOI 10.1117/12.934098.

[311] I. Volosyak, 2005, *Untersuchung neuer Bildverarbeitungs- und Sensorkonzepte als Basis autonomer Automatisierungssysteme*, Dissertation, Fachbereich Physik und Elektrotechnik, Universität Bremen, Aachen: Shaker, ISBN 978-3832243241.

[312] P. Corke, Robotics, 2011, *Vision and control- fundamental algorithms in Matlab*, Berlin: Springer, ISBN 978-3-642-20143-1.

[313] T. S. Huang und A. N. Netravali, 1994, „Motion and structure from feature correspondences: A review," *Proceedings of the IEEE*, Bd. 82, Nr. 2, pp. 252- 268, DOI 10.1109/5.265351.

[314] K. Daniilidis und J.-O. Eklundh, 2008, „3-D vision and recognition," in *Handbook of Robotics*, B. Siciliano und O. Khatib, Hrsg., Berlin: Springer, 1. Auflage, pp. 543-562, ISBN 978-3-540-30301-5.

[315] D. Kragic, 2001, *Visual Servoing for Manipulation: Robustness and integration issues*, PhD Thesis, Royal Institute of Technology Numerical Analysis and Computer Science, University Stockholm.

[316] R. Hartley und A. Zissermann, 2003, *Multiple view geometry in computer vision*, Cambridge University Press, 2. Auflage, ISBN 978-0-521-54051-3.

[317] E. Malis, 2002, „Survey of vision-based control," *European naval ship design, captain computer IV forum, ENSIETA*, pp. 1-16.

[318] G. D. Hager und K. Toyama, 1996, „X Vision: Combining image warping and geometric constraints for fast visual tracking," *Computer Vision- ECCV 96, Lecture Notes in Computer Science*, Bd. 1065, pp. 507- 517, DOI 10.1007/3-540-61123-1_165.

[319] F. Chaumette, 2004, „Image Moments: A general and useful set of features for visual servoing," *IEEE Transactions on Robotics*, Bd. 20, Nr. 4, pp. 713-723, DOI 10.1109/TRO.2004.829463.

[320] R. Y. Tsai, 1987, „A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Journal of Robotics and Automation*, Bd. 3, Nr. 4, pp. 323- 344, DOI 10.1109/JRA.1987.1087109.

[321] Z. Zhang, 2000, „A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 22, Nr. 11, pp. 1330-1334, DOI 10.1109/34.888718.

[322] F. Chaumette und S. Hutchinson, 2008, „Visual servoing and visual tracking," in *Handbook of Robotics*, B. Siciliano und O. Khatib, Hrsg., Berlin: Springer, 1. Auflage, pp. 563-583, ISBN 978-3-540-30301-5.

[323] A. D. Luca, G. Oriolo und P. R. Giordano, 2007, „On-line estimation of feature depth for image-based visual servoing schemes," *IEEE International Conference on Robotics and Automation*, pp. 2823- 2828, DOI 10.1109/ROBOT.2007.363899.

[324] E. Malis, F. Chaumette und S. Boudet, 1999, „2-1/2-D Visual Servoing," *IEEE Transactions on Robotics and Automation*, Bd. 15, Nr. 2, pp. 238-250, DOI 10.1109/70.760345.

[325] N. R. Gans und S. A. Hutchinson, 2002, „A switching approach to visual servo control," *IEEE International Symposium on Intelligent Control*, pp. 770-776, DOI 10.1109/ISIC.2002.1157859.

[326] N. R. Gans und S. A. Hutchinson, 2007, „Stable visual servoing trough hybrid switched-system control," *IEEE Transactions on Robotics*, Bd. 23, Nr. 3, pp. 530-540, DOI 10.1109/TRO.2007.895067.

[327] C. Teulière und E. Marchand, 2012, „Direct 3D servoing using depth maps," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1741-1746, DOI 10.1109/IROS.2012.6385630.

[328] C. Teulière und E. Marchand, 2014, „A dense direct approach to visual servoing using depth maps," *IEEE Transactions on Robotics*, Bd. 30, Nr. 5, pp. 1242-1249, DOI 10.1109/TRO.2014.2325991.

[329] D. Liberzon, 2003, *Switching in Systems and Control*, New York: Springer Science+Business Media LLC, 1. Auflage, ISBN 978-1-4612-6574-0.

[330] J. Lunze und F. Lamnabhi-Lagarrigue, 2009, *Handbook of Hybrid Systems Control- Theory, Tools, Applications*, Cambridge: University Press, ISBN 978-0521765053.

[331] C. Edwards und I. Postlethwaite, 1998, „Anti-windup and Bumpless-transfer Schemes," *Automatica*, Bd. 34, Nr. 2, pp. 199-210, DOI 10.1016/S0005-1098(97)00165-9.

[332] J. Adamy, 2014, *Nichtlineare Systeme und Regelungen*, Berlin: Springer Vieweg, 2. Auflage, ISBN 978-3-642-45012-9.

[333] S. Osmic und A. Trächtler, 2008, „Flatness-based Online Controller Reconfiguration," *IEEE Annual Conference on Industrial Electronics*, pp. 204-209, DOI 10.1109/IECON.2008.4757953.

[334] H.-B. Kuntze, M. Sajidman und A. Jascubasch, 1995, „A Fuzzy-Logic Concept for Highly Fast and Accurate Position Control of Industrial Robots," *IEEE International Conference on Robotics and Automation*, pp. 1184-1190, DOI 10.1109/ROBOT.1995.525441.

[335] T. Kröger, 2010, *On-Line Trajectory Generation in Robotic Systems- Basic Concepts for Instantaneous Reactions to Unforeseen (Sensor) Events*, Berlin: Springer, 1. Auflage, ISBN 978-3-642-05174-6.

[336] T. Kröger und F. M. Wahl, 2010, „Online Trajectory Generation: Basic Concepts for Instantaneous Reactions to Unforeseen Events," *IEEE Transactions on Robotics*, Bd. 26, Nr. 1, pp. 94-111, DOI 10.1109/TRO.2009.2035744.

[337] T. Kröger und F. M. Wahl, 2010, „Stabilizing Hybrid Switched Motion Control Systems with an On-Line Trajectory Generator," *IEEE International Conference on Robotics and Automation*, pp. 4009-4015, DOI 10.1109/ROBOT.2010.5509428.

[338] T. Kröger und B. Finkemeyer, 2011, „Robot motion control during abrupt switchings between manipulation primitives," *Workshop on Mobile Manipulation at the IEEE International Conference on Robotics and Automation*.

[339] T. Kröger, 2011, „Opening the door to new sensor-based robot applications- The Reflexxes Motion Libraries," *IEEE International Conference on Robotics and Automation*, pp. 1-4, DOI 10.1109/ICRA.2011.5980578.

[340] G. Marsaglia, 1972, „Choosing a point from the surface of a sphere," *The Annals of Mathematical Statistics*, Bd. 43, Nr. 2, pp. 645-646, DOI 10.1214/aoms/1177692644.

[341] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha und M. Beetz, 2008, „Towards 3D Point cloud based object maps for household environments," *Robotics and Autonomous Systems*, Bd. 56, Nr. 11, pp. 927-941, DOI 10.1016/j.robot.2008.08.005.

[342] K. Huebner, S. Ruthotto und D. Kragic, 2008, „Minimum volume bounding box decomposition for shape approximation in robot grasping," *IEEE International Conference on Robotics and Automation*, pp. 1628-1633, DOI 10.1109/ROBOT.2008.4543434.

[343] J. Amantides und A. Woo, 1987, „A Fast Voxel Traversal Algorithm for Ray Tracing," *Eurographics*, Bd. 87, Nr. 3, pp. 3-10, DOI 10.2312/egtp.19871000.

[344] P. Norvig und S. Thrun, 2016, „Intro to Artifical Intelligence," Udacity, [Online]. Available: https://www.udacity.com/course/intro-to-artificial-intelligence--cs271. [Zugriff am 18 09 2016].

[345] B. Cohen, S. Chitta und M. Likhachev, 2013, „Single- and dual-arm motion planning with heuristic search," *The International Journal of Robotics Research*, Bd. 33, Nr. 2, pp. 305-320, DOI 10.1177/0278364913507983.

[346] P. E. Hart, N. J. Nilsson und B. Raphael, 1968, „A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, Bd. 4, Nr. 2, pp. 100-107, DOI 10.1109/TSSC.1968.300136.

[347] S. Karaman und E. Frazzoli, 2011, „Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, Bd. 30, Nr. 7, pp. 846-894, DOI 10.1177/0278364911406761.

[348] I. N. Bronstein und K. A. Semendjaew, 2008, *Taschenbuch der Mathematik*, Frankfurt am Main: Harri Deutsch, 7. Auflage, ISBN 978-3817120079.

[349] B. Cohen, I. A. Sucan und S. Chitta, 2012, „A Generic Infrastructure for Benchmarking Motion Planners," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 589-595, DOI 10.1109/IROS.2012.6386228.

[350] J. Bohren, 2015, „Hand-eye calibration integration using aruco_ros and VISP," 04 02 2015. [Online]. Available: https://github.com/jhu-lcsr/aruco_hand_eye. [Zugriff am 18 09 2017].

[351] o.V., 2017, „Github- CANopen," Fraunhofer IPA, [Online]. Available: https://github.com/ipa320. [Zugriff am 21 02 2017].

[352] T. Yoshikawa, 1985, „Manipulability of Robotic Mechanism," *The International Journal of Robotics Research*, Bd. 4, Nr. 2, pp. 3-9, DOI 10.1177/027836498500400201.

[353] A. Albu-Schäffer, 2002, *Regelung von Robotern mit elastischen Gelenken am Beispiel der DLR-Leichtbauarme*, Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, http://nbnresolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss2002042315464.

[354] o.V., 2016, „Technische Daten LWA4P,“ Schunk, [Online]. Available: http://mobile.schunkmicrosite.com/fileadmin/user_upload/broshures/SCHUNK_Technische_Daten_LWA4P.pdf. [Zugriff am 05 02 2016].

[355] P. Fankhauser, M. Bloesch, D. Rodriguez, R. Kaestner, M. Hutter und R. Siegwart, 2015, „Kinect v2 for Mobile Robot Navigation: Evaluation and Modeling,“ *IEEE International Conference on Advanced Robotics*, pp. 388-394, DOI 10.1109/ICAR.2015.7251485.

[356] o.V., 2017, „Schunk FTM 75,“ SCHUNK GmbH & Co. KG, [Online]. Available: http://de.schunk.com/de_de/greifsysteme/product/47021-ftm-75/. [Zugriff am 18 09 2017].

[357] o.V., 2017, „WSG-FWA 50-110,“ SCHUNK GmbH & Co. KG, [Online]. Available: http://de.schunk.com/de_de/greifsysteme/product/49968-31000905-wsg-fwa-50-110/. [Zugriff am 18 09 2017].

[358] C. Friedrich, 2017, „Anwendungsbeispiel Stiftbaugruppe,“ Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, [Online]. Available: https://www.youtube.com/watch?v=vRQx2LiBeSA&feature=youtu.be. [Zugriff am 21 11 2017].

[359] C. Friedrich, 2017, „Anwendungsbeispiel Plattenbaugruppe,“ Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, [Online]. Available: https://www.youtube.com/watch?v=DzoTZ-PwAso. [Zugriff am 21 11 20017].

[360] C. Friedrich, 2017, „Anwendungsbeispiel Kühlschmierstoff,“ Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, [Online]. Available: https://www.youtube.com/watch?v=-oJkdFYRuak. [Zugriff am 21 11 2017].

[361] C. Friedrich, 2017, „Anwendungsbeispiel Ventilbaugruppe,“ Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen, [Online]. Available: https://www.youtube.com/watch?v=YNTGjsosuss&feature=youtu.be. [Zugriff am 21 11 2017].

[362] K. He, G. Gkioxari, P. Dollar und R. Girshick, 2017, „Mask R-CNN,“ *IEEE International Conference on Computer Vision*, pp. 2980-2988, DOI 10.1109/ICCV.2017.322.

In Produktionssystemen spielt die Anlagenverfügbarkeit und Produktqualität eine entscheidende Rolle. Damit eine hohe Verfügbarkeit erreicht werden kann, unterliegen Produktionseinrichtungen regelmäßigen Instandhaltungsarbeiten. Während bereits automatisierte Inspektionsverfahren existieren, werden Wartungs- und Instandsetzungsaufgaben komplett manuell durchgeführt. Zur Steigerung der ökonomischen Effizienz heutiger Produktionseinrichtungen muss jedoch auch hier eine Teilautomatisierung erfolgen. Deswegen erforscht diese Arbeit erstmalig Planungs-, Steuerungs- und Regelungsverfahren für die Autonomiebildung von Robotersystemen mit dem Fokus auf Demontage- und Montageoperationen unter Instandhaltungsbedingungen. Die entwickelten Manipulationsfähigkeiten werden in eine Steuerungsarchitektur integriert und ganzheitlich anhand praxisrelevanter Anwendungsfälle validiert.

ISBN 978-3-8396-1484-6

9 783839 614846

FRAUNHOFER VERLAG