Entwicklung einer intuitiven Mensch-Maschine-Schnittstelle für die automatisierte Kleinserienmontage

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Sven Dose

aus Eutin

Tag der mündlichen Prüfung:	06.02.2014
Erster Gutachter:	Prof. Dr.-Ing. Rüdiger Dillmann
Zweiter Gutachter:	Prof. Dr.-Ing. Heinz Wörn

Selbstständigkeitserklärung

Hiermit versichere ich, diese Dissertation selbstständig, ohne unzulässige Hilfe Dritter und ohne Verwendung anderer als der kenntlich gemachten Hilfsmittel angefertigt zu haben. Alle aus anderen Arbeiten direkt oder indirekt übernommenen Inhalte sind vollständig unter Nennung ihrer Quelle angegeben.

Karlsruhe, Juli 2013

Kurzfassung

Der Trend zu kürzeren Produktlebenszyklen und einer höheren Variantenvielfalt erhöht die Zahl der Kleinserien in der industriellen Montage. Gleichzeitig verstärkt der hohe Anteil an Handarbeit den Kostendruck durch Niedriglohnländer. Ein Ansatz für die bisher meist zu aufwändige Automatisierung bei Kleinserien bieten neuartige, flexible Robotersysteme. Diese sind durch einen universellen Greifer und verschiedene Sensorsysteme zur Objektlokalisierung auch ohne Umbau für vielfältige Applikationen einsetzbar. Ihr Anwendungsgebiet umfasst Pick-and-Place-Aufgaben mit einer großen Bandbreite an Objekten und Varianten zur Bauteilbereitstellung. Eine Inbetriebnahme der Roboter mit herkömmlichen Methoden erfordert jedoch viel Zeit und Expertenwissen. Zudem wird das Erreichen eines robusten Dauerbetriebs durch das breite Anwendungsspektrum und die höhere Fehleranfälligkeit eines Universalsystems erschwert.

Ziel dieser Arbeit ist die Entwicklung einer intuitiven Mensch-Maschine-Schnittstelle, mit der auch unerfahrene Nutzer ein flexibles Robotersystem für Applikationen mit hoher Manipulationsgenauigkeit bei z.T. komplexer Bauteilzuführung programmieren können. Beim Einrichten der Objektlageerkennung wird die Auswahl einer geeigneten Sequenz von Sensoren und Algorithmen durch wenige Fragen mit hinterlegtem Entscheidungsbaum vereinfacht. Für die Parametrierung der Algorithmen werden verschiedene Zwischenergebnisse der Bildverarbeitungskette für eine intuitive Bewertung und Optimierung der Parametereinstellungen visuell aufbereitet. Die Programmierung von Roboterbahnen erfolgt über Jog-Steuerungen anhand speziell auf die Anforderungen zugeschnittener und intuitiv verständlicher Bewegungsrichtungen.

Ein wesentlicher Beitrag dieser Arbeit besteht in der autonomen, strategiebasierten Behandlung von wirtschaftlich unvermeidbaren Störungen im Betrieb mit dem Ziel, Ausfälle zu verhindern und die Prozessrobustheit von flexiblen Robotersystemen zu steigern. Über eine Datenbank mit Strategien kann der Bediener entsprechend den Anforderungen der Applikation verschiedene, geeignete Behandlungsmöglichkeiten für unterschiedliche Fehlerfälle bei der Objektmanipulation und -lokalisierung auswählen. Die zeitoptimale Aufrufreihenfolge der ausgewählten Strategien wird zur Laufzeit durch Schätzung des Fehlerfalls mittels eines Partially Observable Markov Decision Process (POMDP) geplant. Dieser Ansatz ermöglicht, die Kombination verschiedener Strategien zum Lokalisieren, Entfernen und Beschaffen von Objekten sowie zur Prüfung der Anwesenheit, um unterschiedliche Fehlerarten optimal zu behandeln. Zusätzlich wird die Behandlungsdauer reduziert. Die zur Optimierung erforderlichen statistischen Wahrscheinlichkeiten werden autonom als Erfahrungswissen aus vorhergehenden Behandlungen gelernt.

Die entwickelte Mensch-Maschine-Schnittstelle wird mit einem flexiblen Robotersystem an ver-

schiedenen Applikationen aus der Produktion der Robert Bosch GmbH evaluiert. Im Probandentest mit Einstellern aus der Fertigung wird gezeigt, dass eine erfolgreiche Programmierung komplexer Aufgaben kein Expertenwissen erfordert und die Inbetriebnahmedauer von ca. einer Woche auf wenige Stunden reduziert wird. Anhand von Dauerläufen bei fehleranfälligen Applikationen wird eine starke Steigerung der Prozessrobustheit durch die strategiebasierte Störungsbehandlung bei gleichzeitig geringem Inbetriebnahmeaufwand nachgewiesen. Der Einsatz eines POMDP sowie das Lernen des Erfahrungswissens ermöglichen die statistisch optimale Behandlung unterschiedlicher Fehlerfälle sowie eine Reduktion der Behandlungsdauer.

Abstract

The current trend of shorter product life cycle times and higher product variety increases the share of small batch sizes in assembly fabrication. At the same time the high amount of manual work raises the cost pressure by low-wage countries. Novel and flexible robot systems represent an approach for an economic automation that has been too extensive in most cases so far. They can be used for diverse applications without mechanical adaptation because of a universal gripper and various sensor systems for object localization. Their area of application covers pick-and-place tasks with a huge variety of objects and many variants of feeding these into the process. However conventional methods of commissioning require a lot of time and expert knowledge. Further, a robust and continuous operation is hard to achieve because of higher error-proneness of a universal system combined with the wide range of applications.

The aim of this work is to develop an intuitive human-machine interface that enables even inexperienced users to program a flexible robot system for assembly applications with high accuracy of manipulation and a complex feeding of components. For setting up the object detection, a few questions along a decision tree are used to simplify the selection of a proper sequence of localization algorithms and sensors. To reduce complexity of parameterizing the algorithms, different intermediate results of the image processing chain are processed and shown to the user. By intuitive evaluation of the visual results the parameter values can be optimized easily. The robot paths are programmed via teach-in using jog controls that offer directions of robot movements that are specially fitted regarding the requirements and can be used intuitively.

A fundamental contribution of this work is the autonomous and strategy based handling of errors that are economically inevitable during assembly operation. This avoids breakdowns and increases the robustness of flexible robot systems. By considering the requirements and limiting conditions of the application and using a data base with strategies, the user can properly assign several options to handle varying errors during object manipulation or localization. The ideal order of executing the selected strategies is planned at runtime by estimating the causal fault using a partially observable markov decision process (POMDP). This approach allows combining several strategies for localization, removing and obtaining objects as well as for checking their presence to handle different types of faults optimally. Further, the time for handling errors is reduced. The statistical probabilities used for optimization are learned autonomously from previous error handlings.

The human-machine interface is evaluated using a flexible robot system and different assembly applications at Robert Bosch GmbH. Tests including adjusters of the production show that complex assembly tasks can be programmed successfully without expert knowledge and the time needed can

be reduced from about one week to a few hours. Executing error-prone applications in continuous operation shows that the strategy-based error handling allows a significant improvement of process robustness while requiring only little commissioning effort. Using a POMDP combined with learning experience knowledge allows to handle varying fault types efficiently and to reduce the time needed.

Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit bei der Abteilung für Fertigungsautomatisierung (CR/APA) der Robert Bosch GmbH in Kooperation mit dem Institut für Anthropomatik vom Karlsruher Institut für Technologie (KIT).

Zuerst möchte ich mich besonders bei meinem Doktorvater, Prof. Dr.-Ing. Rüdiger Dillmann, für die Betreuung dieser Arbeit, das entgegengebrachte Interesse sowie seine Unterstützung in Form der wertvollen Rückmeldungen und Anregungen bedanken. Desweiteren danke ich Prof. Dr.-Ing. Heinz Wörn für die freundliche Übernahme des Koreferats, sowie Prof. Dr.-Ing. Jürgen Beyerer und Juniorprof. Dr.-Ing. Anne Koziolek für die angenehme und faire Prüfung. Zudem danke ich allen bereits genannten Professoren sowie Prof. Dr.-Ing. Johann M. Zöllner, Prof. Dr. Tanja Schultz, Prof. Dr. Walter F. Tichy und Prof. Dr.-Ing. habil. Björn Hein für die hilfreichen Kommentare und Anregungen zu meiner Arbeit im Rahmen der Professorenrunde.

Großer Dank gilt meinen beiden Betreuern bei Bosch, Dr. Peter Schlaich und Dr. Frank Röthling, für die wertvollen Diskussionen, ihr ehrliches und hilfreiches Feedback sowie die zu mancher Zeit nötigen aufmunternden Worte. Allen Kollegen aus der CR/APA danke ich ganz herzlich für ihre Hilfsbereitschaft sowie die stets lustige und schöne gemeinsame Zeit. Für die fachliche Unterstützung, die vielen Anregungen und die gute Zusammenarbeit danke ich im Speziellen dem ganzen APAS-Team und insbesondere Dieter Kunz, Thomas Witzig, Jan-Peter Eckhoff, Simon Jessen, Christian Diessner, Christoph Noack und Raffael Lorenz aus dem MMI-Teilprojekt. Weiterhin möchte ich Sabine Sayler für die vielen guten Ratschläge und Tipps danken, die sie mir aus ihrer Erfahrung als Doktorandin stets bereitwillig gegeben hat.

Bedanken möchte ich mich auch bei allen, die durch ihre Teilnahme an den Usability-Tests zur Evaluierung der Konzepte beigetragen haben.

Vielen Dank ebenfalls an Dorothee Schmidt, Christine Brand, Diana Kreidler und Isabelle Wappler für die allzeit freundliche und hilfsbereite Unterstützung in allen organisatorischen Fragen.

Schließlich möchte ich meinen Eltern von ganzem Herzen danken, die mich sowohl während meiner Ausbildung als auch bei dieser Arbeit stets unterstützt und ermutigt haben.

Mein tiefster Dank gebührt meiner Partnerin Kristin für die permanente Unterstützung, ihre Geduld und die vielen aufbauenden und motivierenden Worte während der gesamten Zeit. Ohne ihre Rückendeckung wäre diese Arbeit nicht möglich gewesen.

Karlsruhe, im August 2013 *Sven Dose*

Inhaltsverzeichnis

1. Einleitung

Die gezielte Erschaffung und Verwendung von komplexen Werkzeugen ist eine der menschlichen Eigenschaften, die eine Entwicklung der heutigen Kulturen ermöglichte. Schon lange besteht dabei der Traum von automatischen Werkzeugen, die eintönige und schwere Arbeit vollständig ohne menschliche Hilfe verrichten können. Bereits 2000 vor Christus schrieb Aristoteles in seinem Werk *Politik*:

„Wenn jedes Werkzeug auf Geheiß [...] das ihm zukommende Werk verrichten könnte [...], so bedürfte es weder für den Werkmeister den Gehilfen noch für die Herren der Sklaven."

Mit Einsetzen der Industrialisierung und der Erfindung wichtiger Schlüsseltechnologien, wie der Fließfertigung oder der Computer- und Robotertechnik, ist der Mensch diesem Ziel stetig nähergekommen. Heutzutage kann der Automatisierungsgrad in einigen Industriezweigen so weit gesteigert werden, dass vollautomatische Produktionsstraßen ganze Prozessketten ohne manuelle Unterstützung wirtschaftlich ausführen. Die Abbildung 1.1 zeigt beispielhaft eine Montagelinie, bei der verschiedenen Prozessschritte durch einen automatischen Werkstückumlauf verbunden sind. Neben der Freisetzung von manueller Arbeitskraft, die für anspruchsvollere Aufgaben, z. B. in der Planung, Administration oder dem Service eingesetzt werden kann, erzielt die Automatisierung häufig auch eine Qualitätssteigerung bei gleichzeitiger Kostenreduktion. Diese Faktoren erlauben auch in der globalisierten Welt eine nach wie vor konkurrenzfähige Fertigung in den Industrienationen im Vergleich mit Niedriglohnländern.

Abb. 1.1.: Vollautomatische Montagelinie der Firma *mta* [mta 2013]

Der Einsatz moderner Industrieroboter stellt einen wichtigen Faktor zum Erreichen hoher Automatisierungsgrade dar. Deutschland besaß 2011 mit einer Quote von 261 Robotern auf 10.000 Mitarbeiter in der industriellen Fertigung die dritthöchste Roboterdichte weltweit und der Verkauf von 19.500 neuen Industrierobotern in Deutschland bestätigte eine weiterhin hohe Nachfrage [IFR Statistical Department 2012]. Studien zeigen allerdings, dass sich die Automatisierung größtenteils auf die Fertigung von großen Stückzahlen beschränkt [Armbruster u. a. 2006], während in der Kleinserienmontage (KSM) zumeist noch in Handarbeit produziert wird. Wegen der geringen Losgrößen muss die Fertigung hier häufig umgestellt werden. Dies führt bei herkömmlichen Automatisierungslösungen zu hohen Kosten für mechanische Anpassungen und die nötige Umprogrammierung der Anlagen. Durch die aktuellen Entwicklungen zu einer größeren Variantenvielfalt von Produkten und kürzeren Lebenszyklen sinken die Stückzahlen und der Druck steigt insbesondere bei kleinen und mittleren Unternehmen (KMU). Laut einer Befragung von Unternehmern ist eine Verlagerung arbeitsintensiver Prozesse in Niedriglohnländer mit dem Ziel einer reinen Personalkosteneinsparung oft nicht wirtschaftlich, weshalb viele Firmen bereits wieder eine Rückverlagerung anstreben [Kinkel und Spomenka 2009]. Um die Konkurrenzfähigkeit der hiesigen Produktion zu stärken, sind flexible Automatisierungstechnologien erforderlich.

1.1. Problemstellung

Die industrielle Montage umfasst nach VDI Richtlinie 2860 das Verbinden von Einzelteilen zu einem komplexeren Produkt durch einen oder mehrere der Teilprozesse *Fügen* (Primärmontagefunktion), *Handhaben*, *Justieren* und *Kontrollieren* (Sekundärmontage). Um eine konstante und definierte Qualität zu erzielen, erfolgt das Fügen in Form von Urformen, Umformen, Schweißen, Löten oder Verschrauben auch bei der KSM häufig maschinell über einzelne Prozessstationen. Ähnliches gilt für das Justieren und Kontrollieren. Die mit flexiblen Robotersystemen zu automatisierenden Teilschritte der Montage beschränken sich in dieser Arbeit daher hauptsächlich auf das Handhaben der Bauteile in Form von Pick-and-Place-Abfolgen zum Be- und Entladen entsprechender Prozesse bzw. zum Palettieren von Bauteilen.

In der automatisierten Montage erfolgt die Werkstückhandhabung bei herkömmlichen Lösungen vielfach über Roboter. Deren Inbetriebnahme ist sehr zeitaufwändig und erfordert häufig Experten für Robotik, Programmierung und Bildverarbeitung. Gleichzeitig werden die Robotersysteme mechanisch für die konkrete Anwendung optimiert und alle Fehlermöglichkeiten innerhalb des Gesamtprozesses konstruktiv reduziert, um eine hohe Robustheit sowie eine geringe Taktzeit zu erreichen. Die Erkennung einer Störung löst in der Regel einen sofortigen Stillstand der Anlage aus und erfordert einen Bedienereingriff. Daher wird für einen robusten Betrieb häufig eine bis zu mehrere Wochen dauernde Optimierung durchgeführt. Durch die so entstehenden hohen Inbetriebnahme- und Anpassungskosten ist die Automatisierung nur bei großen Stückzahlen wirtschaftlich, da die Startinvestitionen auf viele produzierte Werkstücke umgelegt werden können (siehe Abbildungen

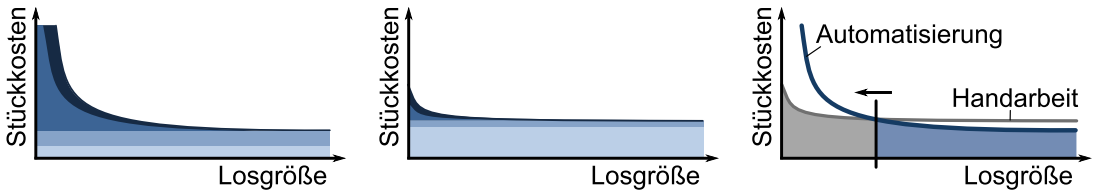


Abb. 1.2.: Zusammensetzung der Montagestückkosten abhängig von der Losgröße: a) für Automatisierung, b) für Handarbeit. c) Vergleich. Die Tabelle 1.1 zeigt die Kostenbestandteile.

Kostenfaktor	Kostenhöhe & Ursache bei: Automatisierung	Handarbeit
Betrieb	✚ Betriebsmittel	━ Lohnkosten
Anschaffung	━ Kauf + Installation	✚ Anpassungen bzgl. Arbeitsschutz und Ergonomie
Inbetriebnahme	━ Programmierung durch Experten	✚ Erklärung durch Einsteller
Störungsbehebung	━ Notstopp + Stillstand, aufwändiger Wiederanlauf	✚ Kompensation durch Handarbeiter

Legende: ✚: geringe Kosten; ━: hohe Kosten

Tab. 1.1.: Gegenüberstellung der Kostenfaktoren bei der manuellen und automatischen Montage

1.2a und 1.2c). Die Vorteile dieser Produktionsform liegen in den geringen Betriebskosten und den möglichen kurzen Taktzeiten, wie die Tabelle 1.1 zeigt.

In der Kleinserienmontage, wo die Produktion aufgrund der reduzierten Losgrößen häufig angepasst werden muss, erfolgt die Werkstückhandhabung wegen der hohen Flexibilität des Menschen zumeist in Handarbeit. Zur Umstellung der Fertigung genügt oft eine kurze verbale Einweisung in die Aufgabe, da der Mensch sich durch seine Erfahrung viele Sachverhalte intuitiv erschließen sowie auftretende Fehler direkt erkennen und flexibel beheben kann. Geringe mechanische Anpassungen der Arbeitsplätze sind oft nur hinsichtlich Arbeitsschutz und Arbeitsplatzergonomie nötig. Aufgrund der hohen Arbeitskosten wird die Handarbeit an Hochlohnstandorten jedoch nur bei geringen Stückzahlen eingesetzt, wenn eine Automatisierung teurer wäre (siehe Abbildungen 1.2b und 1.2c).

Durch den Trend zu kürzeren Produktlebenszyklen und einer höheren Variantenvielfalt nimmt der Anteil der Kleinserien in der Montage zu. Um auch unter diesen Bedingungen konkurrenzfähig zu bleiben, sind neue Konzepte zur Senkung der Startinvestitionen bei der Automatisierung nötig.

Einen möglichen Lösungsansatz bieten flexible Robotersysteme, die ohne aufwändige mechanische Anpassungen oder zusätzliche Hardwareanschaffungen für unterschiedliche Aufgaben verwendbar sind. Sie verfügen über einen Roboterarm an dessen Handflansch mehrere Kamerasysteme und ein universeller Greifer montiert sind, um ein großes Spektrum an Bauteilen ohne feste Vorpositionierung über verschiedene Algorithmen lokalisieren und handhaben zu können. Eine transportable Basis sowie eine integrierte Sicherheitstechnik erlauben es, diese Robotersysteme schnell

und ohne trennende Schutzeinrichtungen an wechselnden Arbeitsplätzen einzusetzen, wodurch der Installationsaufwand sinkt.

Die Inbetriebnahme dieser Systeme mit herkömmlichen Programmier- und Optimierungsmethoden ist sehr zeit- und kostenintensiv. Erforderliche Spezialisten sind in kleinen Firmen zudem häufig nicht beschäftigt. In dieser Arbeit soll daher eine Mensch-Maschine-Schnittstelle (MMS) entwickelt werden, mit der flexible Robotersysteme für typische industrielle Pick-and-Place-Aufgaben mit variablen Bauteilpositionen schnell und ohne Expertenwissen programmiert werden können. Insbesondere muss für einen wirtschaftlichen Einsatz eine hohe Robustheit des Automatikbetriebs erreichbar sein, da diese Robotersysteme durch die fehlende mechanischen Anpassungen an die Randbedingungen der auszuführenden Applikation fehleranfälliger als klassische Automatisierungslösungen sind.

1.2. Zielsetzung und Beitrag

Ziel der Arbeit ist die Entwicklung einer Mensch-Maschine-Schnittstelle, mit der flexible Robotersysteme mit geringem Aufwand für objektgeführte Greif- und Fügeaufgaben in der industriellen Kleinserienmontage programmiert werden können. Durch die Reduktion der Inbetriebnahmekomplexität soll das Einrichten komplexer Applikationen innerhalb weniger Stunden durch unerfahrene Bediener ohne Expertenwissen in Programmierung, Robotik und Bildverarbeitung ermöglicht werden. Gleichzeitig ist eine Störungsbehandlung zu entwickeln und zu integrieren, über die mit geringem Aufwand definierbar ist, wie das Robotersystem für einen robusten Dauerbetrieb auf typische, unvermeidbare Störungen im Betrieb autonom, unterbrechungsfrei und schnell reagieren kann.

Das Anwendungsspektrum für die Programmierung flexibler Robotersysteme umfasst das Be- und Entladen von Prozessstationen für Bearbeitungs- und Prüfaufgaben sowie das Palettieren am Fließbandende. Die Positionen von Bauteilen, Prozessnestern und Werkstückträgern können während der Ausführung der Aufgaben variieren. Für eine robuste Lageerfassung sind zum Teil verschiedene Kombinationen von Kamerasystemen und Erkennungsalgorithmen zu verketten. Die Auswahl dieser Sequenzen sowie die nötige Parametrierung der enthaltenen Algorithmen und Objektmodelle ist ebenso Bestandteil der Inbetriebnahme des Robotersystems wie das Definieren der zur Werkstückhandhabung erforderlichen Bewegungen von Roboter und Greifer. Die Programmierung des Systems soll werkstattbasiert direkt am System erfolgen, wobei a priori keine Modelldaten von Werkstücken und Umgebung zur Verfügung stehen. Für die Inbetriebnahme sind nur kostengünstige, roboterunabhängige und im Betrieb nach gängigen Bestimmungen als sicher geltende Komponenten einzusetzen.

Die Störungsbehandlung soll typische, im Betrieb auftretende und wirtschaftlich nicht vermeidbare Fehler und Störungen autonom und unterbrechungsfrei behandeln und so die Robustheit universeller Robotersysteme deutlich steigern. Eine flexible Berücksichtigung unterschiedlicher Anforderungen und Einschränkungen innerhalb der Applikationen ist zu ermöglichen. Als Sensoren

zur Störungserkennung umfassen typische flexible Robotersysteme nur kraftgeregelte Greifer mit Positionsrückmeldung sowie Lokalisierungsverfahren mit unspezifischen Fehlerrückmeldungen. Für eine Integration weiterer Sensoren ist die Wirtschaftlichkeit des Gesamtsystems zu berücksichtigen. Als Teil der Mensch-Maschine-Schnittstelle soll die Inbetriebnahme der Störungsbehandlung ebenfalls ohne Expertenwissen und mit wenig Aufwand erfolgen können. Im Störungsfall ist eine geringe Behandlungsdauer zu erreichen, um das System in Fertigungslinien mit kurzen Taktzeiten wirtschaftlich einsetzten zu können.

Der Beitrag dieser Arbeit liegt in der Entwicklung einer Prozesskette und der enthaltenen Komponenten zur schnellen und robusten Programmierung flexibler Robotersysteme für objektgeführte Handhabungsaufgaben. Neben der Inbetriebnahme ohne Expertenwissen liegt der Fokus vor allem auf einer Steigerung der Robustheit bei der Ausführung der erstellten Programmabläufe. Hierzu wird eine neuartige, strategiebasierte Störungsbehandlung eingesetzt, um auf unvermeidbare Fehler unterbrechungsfrei und flexibel entsprechend den Applikationsanforderungen zu reagieren. Der Bediener kann die zur Aufgabe passenden Strategien intuitiv zuweisen und die Ausführung über parameterarme Schnittstellen anpassen. Die Aufrufreihenfolge der ausgewählten Strategien wird zur Laufzeit anhand von autonom gelerntem Erfahrungswissen geplant. Im Vergleich zum Stand der Technik ermöglicht die Kombination von Erfahrungslernen und einem POMDP zur probabilistischen Zustandsschätzung und Ausführungsplanung eine optimale Behandlung unterschiedlicher Fehlerarten. Gleichzeitig sind nur sehr wenige Bedienereingaben erforderlich, um die Behandlung anhand von dessen Applikationswissens einfach und schnell starten und die benötigte Behandlungsdauer durch autonomes Lernen optimieren zu können.

Durch eine hohe Anzahl an Behandlungsmöglichkeiten und bedingt abhängige Erfolgswahrscheinlichkeiten bei ihrer Ausführung wäre die Lösungsberechnung für den POMDP sehr rechenaufwändig. Um eine regelmäßige Verbesserung des Erfahrungswissens berücksichtigen zu können, wurde eine Approximationsmethode durch Vorselektion geeigneter Aktion über Wahrscheinlichkeitsbäume entwickelt, die eine dynamische Berechnung zur Laufzeit zwischen aufeinander folgenden Applikationszyklen ermöglicht. Das entwickelte Konzept kombiniert die Stärken von Mensch und System und erlaubt so die wirtschaftliche Automatisierung selbst von störungsanfälligen Applikationen.

Die Kernthesen dieser Arbeit lassen sich wie folgt zusammenfassen:

- Intuitive Mensch-Maschine-Schnittstellen ermöglichen eine wirtschaftliche Programmierung flexibler Robotersysteme für industrielle, objektgeführte Handhabungsapplikationen mit geringen Losgrößen. Die Inbetriebnahme inklusive des Einrichtens komplexer Objektlageerkennungen ohne a priori Modelldaten kann ohne Expertenwissen innerhalb weniger Stunden erfolgen.

- Eine strategiebasierte Störungsbehandlung ermöglicht einen robusten Dauerbetrieb von flexiblen Robotern auch bei störungsanfälligen Applikationen. Die autonome, unterbrechungs-

freie und auf die Applikation zugeschnittene Verarbeitung von wirtschaftlich nicht vermeidbaren prozess- und konstruktionsbedingten Betriebsstörungen reduziert Anlagenstillstände und nötige Bedienereingriffe.

- Der Einsatz erfahrungsbasierten Lernens zur Schätzung des vorliegenden Fehlerfalls sowie zur Bewertung und optimierten Auswahl der statistisch geeignetsten Behandlungsstrategien ermöglicht eine Behandlung verschiedener Fehlerarten sowie eine deutliche Taktzeitreduktion im Störungsfall.

1.3. Aufbau der Arbeit

Die vorliegende Arbeit gliedert sich in 8 Kapitel. Der Aufbau ist im Folgenden dargestellt.

Kapitel 2 zeigt die Randbedingungen in der Kleinserienmontage. Die typischen Komponenten und Eigenschaften von flexiblen Robotersystemen werden erläutert und das Aufgaben- sowie das Bauteilspektrum analysiert. Potentiell auftretende Fehler und Störungen im Automatikbetrieb dieser Systeme werden über eine Fehlermöglichkeits- und Einflussanalyse untersucht. Ausgehend von den Ergebnissen werden die Anforderungen an die Mensch-Maschine-Schnittstelle und eine Störungsbehandlung abgeleitet.

Kapitel 3 stellt den aktuellen Stand von Technik und Forschung zur Inbetriebnahme flexibler Robotersysteme vor. Der Fokus liegt auf Ansätzen zur Erstellung von Roboterablaufplänen, dem Einrichten von Objektlageerkennungen sowie einer Steigerung der Prozessrobustheit.

Kapitel 4 beschreibt die entwickelte Mensch-Maschine-Schnittstelle und geht auf die Bedienerführung, das Einrichten der Objektlageerkennung sowie das Teachen der Roboterbewegungen ein.

Kapitel 5 erläutert die Funktion der strategiebasierten Störungsbehandlung sowie die Strategien, über die eine hohe Systemrobustheit erreicht wird. Basierend auf der Erfassung von Erfahrungswissen werden die Schätzung des Fehlerzustands sowie die statistisch optimale Strategieauswahl zur Laufzeit beschrieben. Weiterhin werden eine Reduktion des Rechenaufwands sowie die Initialisierung der statistischen Daten nach einem Applikationswechsel erläutert.

Kapitel 6 verdeutlicht die Systemstruktur der entwickelten Mensch-Maschine-Schnittstelle anhand des Aufbaus und der Schnittstellen zur Hardware. Dargestellt wird die Umsetzung der Fehlerbehandlung anhand der Steuerungsstruktur, des Strategieaufbaus sowie der Datenspeicherung.

Kapitel 7 beschreibt die Randbedingungen und Ergebnisse der zur Evaluierung dieser Arbeit durchgeführten Probandentests im Hinblick auf die Programmierdauer und die Akzeptanz durch die Nutzer. Anhand von Dauerläufen wird die Fehlerbehandlung bezüglich der erreichten Robustheit, des Inbetriebnahmeaufwands und der erforderlichen Behandlungsdauer bewertet.

Kapitel 8 fasst die vorliegende Arbeit zusammen und diskutiert mögliche Weiterentwicklungen.

2. Analyse der Randbedingungen für eine flexible Automatisierung der Kleinserienmontage

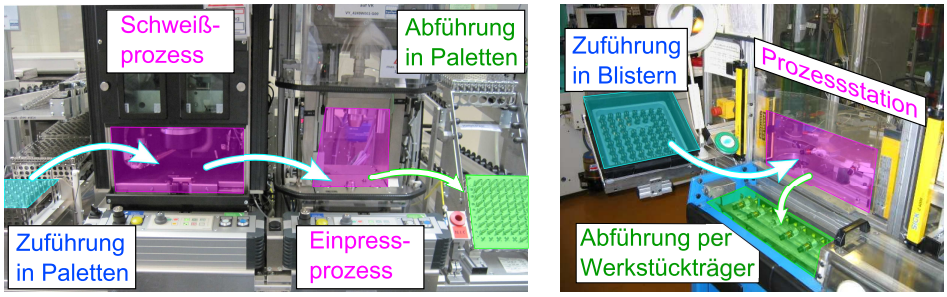
Im Rahmen dieser Arbeit wurden die Randbedingungen für die flexible Automatisierung der Kleinserienmontage analysiert. Der Abschnitt 2.1 zeigt hierzu das Aufgabenspektrum der manuellen Fertigung, das mittels eines Robotersystems abgedeckt und über die MMS programmiert werden soll. Erläutert wird ebenfalls das auftretende Spektrum an Bauteilen, deren Eigenschaften sowohl einen Einfluss auf die Lageerkennung und Handhabung der Objekte als auch auf die Robustheit des Automatikbetriebs haben. Der Abschnitt 2.2 stellt unterschiedliche Ausprägungen flexibler Robotersysteme vor und erläutert, welche Systemkonfigurationen zur Abdeckung des Aufgabenspektrums bei der weiteren Konzeption adressiert wird. Der Abschnitt 2.3 enthält die Ergebnisse einer Analyse der Betriebsstörungen, die bei einer flexiblen Automatisierung von industriellen Montageprozessen auftreten. Ausgehend von den Ursachen werden Möglichkeiten zur Steigerung der Robustheit abgeleitet. Zuletzt zeigt der Abschnitt 2.4 die aus den Randbedingungen folgenden Anforderungen an die zu entwickelnde MMS und die Störungsbehandlung.

2.1. Typische Applikationen in der Kleinserienmontage

Um die Anforderungen an eine MMS zur Programmierung flexibler Robotersysteme zu erfassen und Möglichkeiten zur Steigerung der Dauerlaufrobustheit zu identifizieren, wurden im Rahmen dieser Arbeit verschiedene manuelle Montageapplikationen bei der Robert Bosch GmbH hinsichtlich ihres Aufgaben- und Bauteilspektrums analysiert.

2.1.1. Aufgabenspektrum

Die industrielle Montage bezeichnet laut VDI Richtlinie 2860 die Teilprozesse Fügen, Handhaben, Justieren und Kontrollieren mit dem Ziel Einzelteile zu komplexeren Produkten zu verbinden. Das Aufgabenspektrum in der manuellen Montage hochwertiger Baugruppen beschränkt sich hiervon fast ausschließlich auf das Handhaben von Werkstücken. Fügeschritte wie Pressen, Schweißen, Löten, Verschrauben sowie Ur- und Umformen werden über maschinelle Prozessstationen durchgeführt, um eine konstant hohe Qualität zu erreichen. Gleiches gilt für das Justieren von Baugruppen. Die Abbildung 2.1a zeigt beispielhaft zwei Prozessstationen zum Verschweißen und Einpressen



(a) **(b)**

Abb. 2.1.: Be- und Entladen von Prozessstationen: a) Maschinelles Verbinden von Werkstücken über zwei Prozessstationen, b) Bauteilmontage und Prüfung über Prozessstation

von Werkstücken. Lediglich einfache Sichtprüfaufgaben zur Kontrolle auf grobe Beschädigungen und Vollständigkeit werden direkt vom Mitarbeiter durchgeführt. Toleranzmessungen und Funktionsprüfungen erfolgen zumeist über spezielle Prüfstationen.

Aktuelle Anwendungsszenarien für universelle Robotersysteme umfassen einen flexiblen Einsatz an Handarbeitsplätzen, um auf schwankende Auftragseingänge zu reagieren. Daher bestehen die zu automatisierenden Applikationen entsprechend der manuellen Montage hauptsächlich aus Pick-and-Place-Abfolgen. Die Aufgaben umfassen das Be- oder Entladen von Prozessstationen für Füge- oder Prüfschritte sowie das Palettieren von Bauteilen. Die Applikationen lassen sich in folgende Einzelschritte untergliedern:

- Vor- und Nachbereitung: Palette wechseln, Karton öffnen oder falten, Zwischenboden einlegen, Karton verschließen, Etikettieren, u. Ä.
- Montage:
  - Bauteilmanipulation: Greifen, Transfer, Umgreifen, Ablegen
  - Entscheidung treffen anhand: Prüfergebnis, Wiederholungsanzahl, Füllstand einer Werkstückaufnahme, u. Ä.
  - Anlagenkommunikation:
    - Prozess starten über: Taste, Hebel, Schieber, Anwesenheitssensor, u. Ä.
    - Prozessende erkennen anhand: fester Zeitdauer, Statusanzeige, Fertigteilauswurf, Rohteileinzug, Öffnung des Eingreifschutzes

Die Abbildung 2.1b zeigt ein typisches Arbeitsplatzlayout zum Be- und Entladen von Prozessstationen. Bei dieser Applikation werden die Bauteile über Blister in Kisten angeliefert und nach dem Prozess mittels Werkstückträgern abtransportiert. Erfolgt die Zu- oder Abführung der Komponenten innerhalb eines Prozesses automatisch, können Teilschritte entfallen.

Im Rahmen dieser Arbeit werden nur Montageaufgaben betrachtet, die einarmig und ohne Kooperation mit Personen ausführbar sind. Im Fokus stehen ebenfalls keine Handlungen, die komplexe Fertigkeiten erfordern, wie das gezielte Handhaben von biegeschlaffen Teilen, das Fügen von

Dichtringen unter Spannung oder das Anbringen von selbstklebenden Labels. Ausnahmen bestehen sofern sich Aktionen durch Hilfsmittel auf Pick-and-Place-Schritte reduzieren lassen. So kann ein Umgreifen von Werkstücken durch eine angepasste Aufnahme häufig in ein Ablegen und ein erneutes Greifen überführt werden.

Die zur Bauteilmanipulation verfügbare Taktzeit hängt bei den Applikationen stark vom enthaltenen Prozess sowie der Anzahl der durchzuführenden Teilhandlungen ab. In der Regel stehen für jede Aktion wenige Sekunden zur Verfügung. Vor diesem Hintergrund sind viele der komplexen Aktionen zur Teilebereitstellung für ein flexibles Robotersystem zu aufwändig. Sie könnten stattdessen z. B. von sogenannten *Milk Runnern* übernommen werden, welche die Arbeitsplätze zyklisch mit Rohteilen versorgen und die fertigen Produkte abtransportieren. Durch eine Bereitstellung vieler Einzelteile, z. B. mittels mehrerer Paletten, kann der autonome Arbeitszeitraum eines Robotersystems verlängert werden.

Die vielfältigen, zum Teil manuellen Handlungen für die Anlagenkommunikation lassen sich für eine effiziente Automatisierung mit einem Robotersystem häufig über ein Empfangen und Senden von digitalen Signalen abbilden.

Bei der Konzeption der MMS und der Optimierung der Robustheit sind zusätzlich verschiedene, teilweise auftretende Randbedingungen zu berücksichtigen:

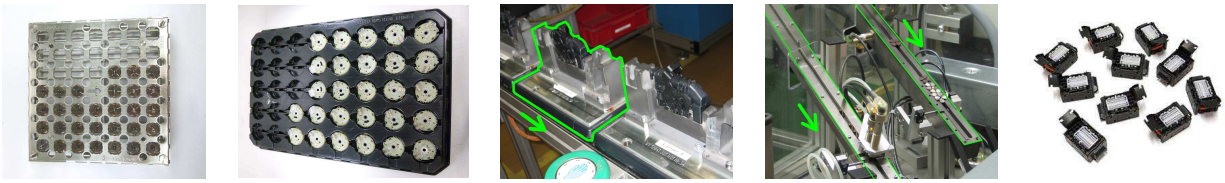
- Hoher Lärmpegel durch Produktionsumfeld
- Z. T. starkes Fremdlicht durch: Bearbeitungsprozesse, Arbeitsplatzbeleuchtung oder Sonnenlicht
- Bauteilverunreinigung durch: Späne, Öl, Schmier- und Schleifmittel

2.1.2. Bauteilspektrum und -zuführung

Vor dem Hintergrund einer flexiblen Automatisierung spielen die Beschaffenheit der Bauteile sowie die Art ihrer Zu- und Abführung in der Applikation eine wichtige Rolle für die korrekte Handhabung. Sie beeinflussen sowohl die Objektlokalisierung als auch die Bauteilmanipulation und sind bei der Auslegung der Mensch-Maschine-Schnittstelle sowie der Störungsbehandlung zu berücksichtigen. Für die Analyse der Eigenschaften wurde im Vorfeld der Arbeit ein charakteristischer Querschnitt von 52 Applikationen und den enthaltenen Bauteilen untersucht (vergleiche auch [Sayler 2011]). Die Abbildung 2.2 zeigt einen Querschnitt über das auftretende Bauteilspektrum. Eine automatische Zuführung der Teile erfolgt über Prozessnester einer Anlage, Bauteilrutschen oder Werkstückträger, die auf Förderbändern in einer Montagelinie umlaufen. Ebenso können die Bauteile manuell als Schüttgut, über Paletten oder in Transportkisten mit Blistereinsätzen bereitgestellt werden. Die Abbildungen 2.3a bis 2.3e zeigen verschiedene Beispiele.

In dieser Arbeit wird davon ausgegangen, dass sich die Bauteile, wie bei der Montage üblich, für die Handhabung in einem stationären Zustand befinden und nicht bewegen. Abhängig von den

Abb. 2.2.: Bauteilspektrum in der Kleinserienmontage



(a) **(b)** **(c)** **(d)** **(e)**

Abb. 2.3.: Verschiedene Varianten der Bauteilzu- und abführung. a) Palette, b) Blister, c) Werkstückträger mit Bauteil, d) Bauteilrutsche, e) Zuführung auf Pufferfläche

Objekteigenschaften und der Applikation kann die Lage eines Bauteils jedoch zwischen verschiedenen Bearbeitungszyklen variieren. Auch die Position von Paletten, Transportkisten und enthaltenen Blistereinsätzen kann im Betrieb schwanken, da diese zur Vermeidung aufwändiger mechanischer Anpassungen der Arbeitsplätze häufig nicht fixiert werden. Mögliche Ursachen für Lageveränderungen sind z. B. das manuelle Austauschen der Magazine nach vollständiger Bearbeitung aller Bauteile oder das Verschieben leichter Werkstückspeicher infolge ungenauer Bauteilabgriffe bzw. eines unsanften Absetzens schweren Bauteile. Das Ausmaß der Positions- und Orientierungsschwankungen von Werkstückspeichern ist von vielen Faktoren abhängig, kann aber häufig z. B. durch aufklebbare Anschlagswinkel oder Markierungen der Sollpositionen innerhalb einer Applikation reduziert werden. Die verbleibende Varianz bei der manuellen Positionierung ist stark von Handlichkeit und Gewicht der Bauteile und Magazine abhängig. Einen weiteren Einfluss auf die Genauigkeit haben alters- und nutzungsbedingte Verformungen der Paletten, Kisten und Blister.

Die Analyse der Bauteile und Applikationen ergab folgende Eigenschaften und Randbedingungen, die einen Einfluss auf eine mögliche Objektlokalisierung und Handhabung besitzen:

Bauteilbereitstellung	$\in$	{Palette / Blister (33%), Prozessnest (26%), Schüttgut (24%), Werkstückträger (9%), Freifläche (8%)}
Varianz Bauteilposition[1]	$\in$	{< 0,5mm (26%), 0,5 - 1mm (28%), 1 - 5mm (35%), > 5mm (11%)}
Varianz Bauteilorientierung[1]	$\in$	{< 1° (30%), 1 - 5° (29%), > 5° (41%)}
Art der Lagevarianz[1]	$\in$	{reine Rotation um Senkrechte (36%), feste Lage (28%), Rotation und Translation parallel zur Unterlage (18%), Orientierung schwankend - Position fest (14%), Position und Orientierung schwanken auf Unterlage (4%)}
Symmetrie der Ansicht	$\in$	{vollständig symmetrisch (31%), hoher Symmetrieanteil (28%), unsymmetrisch (41%)}
Oberflächenmusterung	$\in$	{homogener Farbverlauf (65%), texturiert (35%)}
Oberflächenprofil	$\in$	{überwiegend homogen (60%), stark unstetig (40%)}
Lichtstreuung der Oberfläche	$\in$	{reflektierend (52%), diffus streuend (48%)}
Höhe der Greiffläche	$\in$	{0,5 - 1mm (13%), 1 - 5mm (52%), > 5mm (35%)}
Greifoberfläche	$\in$	{glatt (52%), strukturiert (25%), rau (23%)}

Die Bereitstellung der Bauteile sowie die Art und Stärke der Lagevarianz beeinflussen, in welchem Umfang eine Lokalisierung erforderlich ist. Die Objektsymmetrie sowie die Musterung, das Profil und die Lichtstreuung der Oberfläche schränken hingegen die anwendbaren Erkennungsmöglichkeiten ein und wirken sich auf die Auftretenswahrscheinlichkeit potentieller Fehler und Störungen bei der Lokalisierung aus. Die Höhe und Beschaffenheit der Greiffläche beeinflussen zusammen mit der zuvor beschriebenen möglichen Bauteilverschmutzung auch die Robustheit der Handhabung.

2.2. Flexible Robotersysteme für die Kleinserienmontage

Dieser Abschnitt erläutert die Anforderungen an die Komponenten eines flexiblen Roboters und beschreibt ausgehend von bekannten Systemen die Eigenschaften der im Rahmen dieser Arbeit betrachteten Umsetzungsvarianten. Um bei einer Umstellung der Produktion schnell und ohne aufwändige Modifikationen am System oder der Applikation für ein großes Aufgabenspektrum einsetzbar zu sein, sind verschiedene Eigenschaften erforderlich (vgl. auch [Brecher u. a. 2006]):

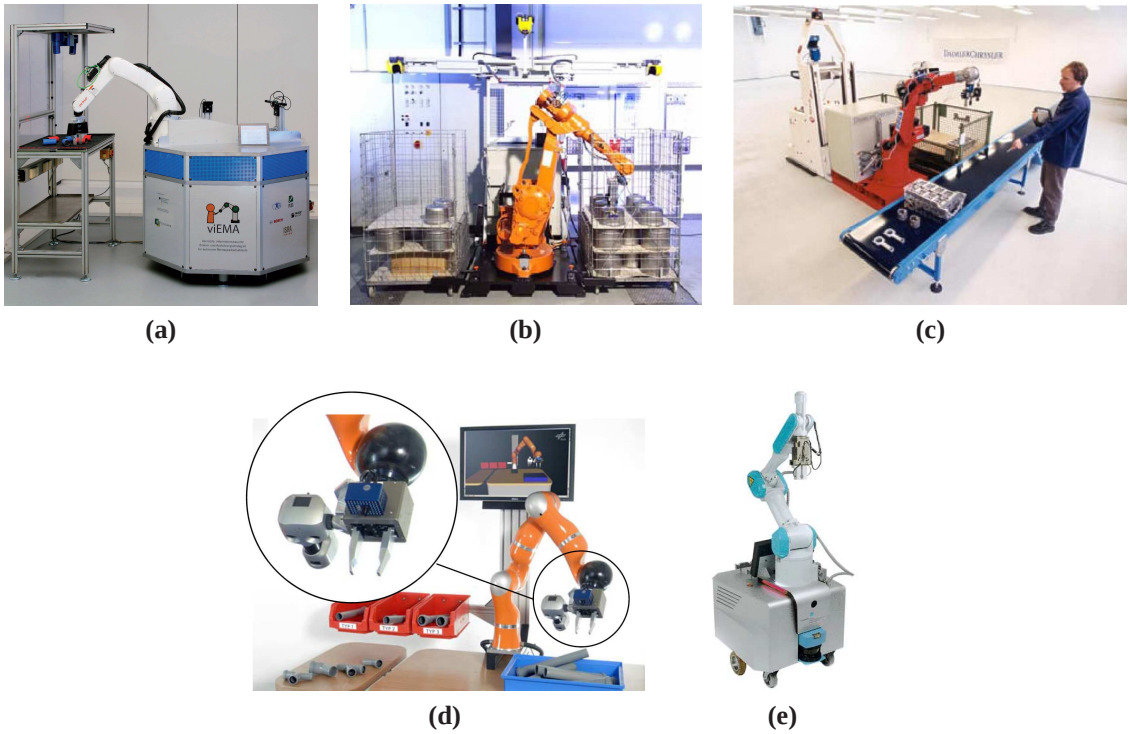
- Transportabilität für schnellen Wechsel des Arbeitsplatzes
- Kinematik mit ausreichendem Arbeitsraum und Freiheitsgraden für nötige Bewegungen
- Flexibler Greifer zur Handhabung verschiedener Teile ohne Umbau
- Sensorik zur Erfassung von variierenden Objektlagen

[1] ohne Zuführung als Schüttgut

- Schnittstelle für Kommunikation mit externen Anlagen
- Sicherheitstechnik für den gefahrlosen Einsatz neben dem Menschen

Die technische Umsetzung dieser Anforderungen variiert bei verschiedenen Systemen z. T. stark, wie Abbildung 2.4 an flexiblen Handhabungsrobotern aus der Forschung zeigt. Aus diesen und anderen Systemen ergeben sich vor allem folgende Umsetzungsvarianten für die genannten Aspekte:

Transport	$\in$	{fixierbare Rollen, aktiv angetriebene Basisplattform, Laschen für Hubwagen, Ösen für Kranhaken}
Kinematik	$\in$	{vertikaler Knickarm-Roboter, Scara-Roboter, Kombination modularer Einzelachselemente}
Greifer	$\in$	{2-/3-Backen-Greifer (u. a. 3-Finger-Sterngreifer), Saug-, Spezialgreifer}
Sensorik	$\in$	{3D-Formfit mittels Laserscanner, Time-of-Flight-Kamera oder Distanzmessung entlang Trajektorie, kamerabasierte kanten-, textur-, oberflächen- oder landmarkenbasierte Lokalisierung}
Kommunikation	$\in$	{Digitale I/O-Karte, Ethernet-Schnittstelle}
Sicherheit	$\in$	{Momentenüberwachung in den Achsgelenken, Umfeldüberwachung mittels Laserscanner oder Kameras, taktile Sensoren, kapazitive Nahbereichsüberwachung, Trittschutzmattten, mechanische Umzäunung, u. Ä.}



(a) **(b)** **(c)** **(d)** **(e)**

Abb. 2.4.: Unterschiedliche Ausprägungen von Handhabungsrobotern: a) Roboterzelle aus dem Projekt viEMA (siehe [viEMA 2013]), b) Robotersystem Porthos ([Matthias u. a. 2006]), c) DaimlerChrysler Manufacturing Assistant ([Helms 2003]), d) DLR Co-Worker ([Haddadin u. a. 2011]), e) Assistenzroboter rob@work ([Schraft u. a. 2004])

2.2.1. Betrachtete Varianten flexibler Robotersysteme

Von den vorgestellten Umsetzungsvarianten der einzelnen Komponenten schließen sich einige aufgrund der in Abschnitt 2.1.1 analysierten Randbedingungen bzgl. des Applikations- und Bauteilspektrums aus.

Als Kinematik werden Roboter mit mindestens 6 Freiheitsgraden betrachtet, die im Vergleich zu Scara-Robotern eine Verkippung des Greifers bzgl. aller Raumachsen ermöglichen. Wegen ihrer Robustheit bieten sich industrietaugliche vertikale Knickarm-Roboter an. Um das Konzept auf verschiedene Plattformen übertragen zu können, werden keine herstellerspezifischen Funktionen vorausgesetzt, wie z. B. Kraft-Null-Regelungen zum manuellen Führen der Kinematik. Erforderlich ist jedoch eine Möglichkeit zur Vermeidung von Kollisionen des Systems mit starren Hindernissen bzw. zum anschließenden Freifahren der Kinematik.

Der Greifer soll das vorgestellte Bauteilspektrum universell, ohne mechanische Anpassungen möglichst umfangreich handhaben können. Spezialgreifer scheiden daher ebenso aus wie Sauggreifer, da viele Werkstücke aufgrund von inhomogenen Oberflächenverläufen oder zu geringen Ausmaßen keine ausreichende Kontaktfläche bieten. Der Fokus liegt daher auf 2- und 3-Backen-Greifern, wie z. B. universellen 3-Finger-Sterngreifern (siehe [Sayler 2011]).

Die Sensorik des Systems muss vor allem das breite Spektrum an Objekten genau und wegen der geringen Taktzeit auch sehr schnell lokalisieren können. Um den Montageroboter mit geringem Wechselaufwand an verschiedenen Arbeitsplätzen einsetzen zu können, werden nur Sensoren betrachtet, die fest am System verbaut sind und nur einmalig zu kalibrieren sind. Wegen der kurzen Taktzeit fokussiert sich die Konzeption in dieser Arbeit auf kamerabasierte one-shot Lokalisierungsverfahren. Um das Bauteilspektrum abzudecken, können z. B. textur-, kontur-, landmarken- oder oberflächen-basierte Algorithmen verwendet oder sogar kombiniert werden. Die eingesetzte Sensorik muss entsprechend des Applikationsspektrums eine genaue Lokalisierung (< 1mm) innerhalb eines großen Suchbereichs (< 25 x 25cm) ermöglichen. Sollte ein einzelner Sensor nicht alle Anforderungen abdecken, können auch mehrere kombiniert werden.

Die konkrete Ausprägung der Sicherheitstechnik ist für die Konzeption der MMS sowie für die Robustheitssteigerung nicht relevant. Wegen des geringen Platzes in der Fertigung und der Anforderung an das System, innerhalb einer Montagelinie einsetzbar zu sein, wird im Folgenden davon ausgegangen, dass sich Menschen und Roboter im selben Arbeitsraum mit geringem Abstand bewegen können. Eine kollaborative Zusammenarbeit wird hingegen nicht betrachtet.

Die genaue technische Umsetzung der Aspekte Transportabilität und Kommunikation hat sowohl auf die MMS als auch auf den robusten Dauerlauf keine Auswirkungen. Beide Anforderungen werden bei der weiteren Betrachtung lediglich als erfüllt angesehen.

2.2.2. Typisches Bedienerprofil

Für die Konzeption einer intuitiv bedienbaren MMS ist es wichtig, die Fähigkeiten und Gewohnheiten der Zielgruppe des Systems zu kennen. Die Inbetriebnahme eines flexiblen Robotersystems

würde bei der Robert Bosch GmbH durch Einsteller erfolgen. Zu deren Aufgabengebiet gehört neben dem Vorbereiten und Rüsten von Maschinen auch die Organisation der Abläufe sowie das Analysieren und Beseitigen von auftretenden Fehlern im Fertigungsprozess. Typische Vertreter sind neben Systemmechaniker und Maschineneinrichter z. B. auch Prozessoptimierer oder Fertigungssystemgestalter. Laut Anforderungsprofil erfordern diese Stellen eine Ausbildung zum Industriemechaniker, Techniker o. Ä., so dass Kenntnisse in Programmierung, Robotik oder Bildverarbeitung nicht vorausgesetzt werden können. Dies deckt sich mit den Ergebnissen einer im Rahmen dieser Arbeit durchgeführten Befragung (Fragebogen siehe Anhang C.2), nach der jedoch grundlegende Erfahrungen im Umgang mit PCs und gängiger Software wie Windows oder Office vorhanden sind.

2.3. Klassifizierung von Störungen im Automatikbetrieb flexibler Robotersysteme

Für eine gezielte Optimierung sind als erstes die potentiellen Einflussfaktoren auf die Robustheit der automatischen Montage zu identifizieren. Da hierbei auftretenden Begriffe häufig uneinheitlich verwendet werden, erfolgen zunächst einige Definitionen für die vorliegende Arbeit. Anschließend werden die zur Fehleranalyse verwendete Methode sowie die Ergebnisse erläutert.

2.3.1. Begriffsdefinitionen bzgl. der Robustheit von Montageprozessen

In der englischsprachigen Literatur werden im Zusammenhang mit der Robustheit und der Fehleranfälligkeit häufig die Begriffe *Fault*, *Error* und *Failure* verwendet, um Ursache, Verlauf und Auswirkung eines Problems zu beschrieben. Da alle drei Begriffe oft missverständlich mit *Fehler* ins Deutsche übersetzt werden, erfolgt an dieser Stelle eine Definition für die vorliegende Arbeit in Anlehnung an [Loborg 1994].

Fehler (Fault)
Ein Fehler stellt die Ursache eines Problems dar und kann direkt oder indirekt zu einer Störung führen. Mögliche Fehler in einem Prozess sind z. B. defekte Systemkomponenten oder unerwartete Störeinflüsse von außen.

Störung (Error)
Eine Störung bezeichnet das Auftreten einer relevanten Abweichung zwischen dem spezifizierten und dem tatsächlichen Zustand infolge eines Fehlers. Bezogen auf die Steuerungstechnik ist eine Störung vor allem eine Abweichung zwischen dem realen Systemzustand und seiner internen Repräsentation. Die Abweichung kann dabei bereits erkannt oder latent sein. Wird eine Störung nicht korrigiert, zieht dies in der Regel einen Ausfall oder ein Versagen nach sich.

Ausfall / Versagen (Failure)
Der Ausfall oder das Versagen bezeichnet die Beeinträchtigung der Funktionstüchtigkeit einer Komponente infolge einer Störung, so dass die vorgesehene Aufgabe nicht erfüllt werden kann. Ist die Komponente Teil eines Systems, stellt der Ausfall gleichzeitig einen Fehler im Bezug auf das System dar.

2.3.2. Analyse auftretender Störungen und Ausfälle

Methode der Fehlermöglichkeits- und Einflussanalyse
Um potentielle Ausfälle des Systems zu identifizieren, wurde im Rahmen dieser Arbeit eine Fehlermöglichkeits- und Einflussanalyse (FMEA) durchgeführt. Dieses Verfahren wird bei der Entwicklung neuer Produkte als Teil des Qualitätsmanagements eingesetzt (siehe [DGQ 2012]). Durch eine strukturierte Analyse aller Funktionen einer Komponente sollen potentielle Störungen möglichst früh erkannt und über abzuleitende Maßnahmen bereits im Vorfeld verhindert werden. Abhängig vom Fortschritt des Entwicklungsprozesses werden unterschiedliche Aspekte analysiert, um Fehler im Zusammenspiel mit anderen Komponenten (System-FMEA), bei der Konstruktion (Konstruktions-FMEA) oder bei der Produktion (Prozess-FMEA) zu identifizieren.

Bei der Analyse werden für jede Funktion einer Komponente alle potentiellen Fehler und daraus folgenden Störungen untersucht. Zu bewerten sind jeweils die Auftretenswahrscheinlichkeit des Fehlers (A) sowie die Entdeckungswahrscheinlichkeit von Fehlern oder resultierenden Störungen bzw. die Erkennungsrate der Systemausfälle (E). Weiterhin wird die Bedeutung eines Ausfalls für den Prozess (B) eingestuft. Die Quantifizierung erfolgt mit Werten zwischen 0 und 10 aufsteigend anhand des Schweregrads über fest definierte Tabellen, die für eine objektive Bewertung klare Kriterien beinhalten. Anhand der Höhe der einzelnen Bewertung sowie am Produkt der drei Kennzahlen, der Risikoprioritätszahl (RPZ), kann die Relevanz eines Fehlers erkannt werden. Bei hohen Werten werden Maßnahmen zur Vermeidung oder Erkennung von Fehlern und Störungen bzw. zur Abschwächung der Auswirkung ergriffen. Anschließend wird die Relevanz erneut bewertet. Diese Schritte werden iterativ wiederholt, bis die Auswirkungen als vernachlässigbar angesehen werden oder eine weitere Reduktion des Risikos aufgrund des erforderlichen Aufwands unwirtschaftlich ist. In diesem Fall ist das verbleibende Restrisiko nachfolgend zu berücksichtigen.

Ergebnisse
Basierend auf den Ergebnissen der Analyse des Aufgabenspektrums und den Eigenschaften der betrachteten Robotersysteme (siehe Abschnitte 2.1.1 und 2.1.2) wurde eine FMEA durchgeführt. Bewertet wurden alle für den automatischen Dauerbetrieb relevanten Systemfunktionen der Roboter im Zusammenspiel mit den möglichen Ausprägungen der Montageprozesse. In mehreren Iterationen wurde zunächst analysiert, wie das Auftreten von Fehlern und Störungen über technische Maßnahmen soweit wie wirtschaftlich sinnvoll reduziert werden kann. Darüber hinaus werden in dieser Arbeit nur die Fehler berücksichtigt, die im regulären Betrieb auftreten, d.h. ohne techni-

schen Defekt einer Komponente. Bei folgenden Systemfunktionen verbleibt auch nach Optimierung ein relevantes Restrisiko:

- Objektlageerkennung
- Bauteilmanipulation
- Kommunikation mit externen Anlagen

Objektlageerkennung:

Die Tabelle 2.1 zeigt die bei der Objektlageerkennung auftretenden Fehler inklusive ihrer Herkunft. Unterschieden wird bei den Fehlerursachen hinsichtlich falscher Bedienereingaben bei der Inbetriebnahme des Robotersystems (I), manueller Eingriffe während des Betriebs (M) sowie schwankender Bedingungen im Prozess (P). Bei der manuellen Montage werden schwankende Randbedingungen durch den Menschen ausgeglichen, so dass vollständig fehlerfreie Prozesse nicht erforderlich sind. Für die Automatisierung können Möglichkeiten zur Reduktion von prozessbedingten Fehler- und Störungsrisiken nicht vorausgesetzt werden, da das Robotersystem auf den zu bedienenden Prozess keinen Einfluss hat. Ohne geeignete MMS sind auch Bedienerfehler bei der Inbetriebnahme nicht auszuschließen. Manuelle Eingriffe sind oft sogar nötig, um neue Werkstücke bereitzustellen, bearbeitete Teile abzutransportieren oder die Einhaltung von Fertigungstoleranzen

		Mögliche Störung		
Fehler	Fehlerherkunft	Bauteillage nicht erkannt	Bauteillage ungenau erkannt	Pseudo-Bauteillage erkannt
Kein Objekt zugeführt	P	x		
Objektposition außerhalb Toleranz	P, M	x		
Objektorientierung außerhalb Toleranz	P, M	x		
Objektmerkmale verdeckt durch dasselbe / andere Objekte	P	x		
Durch Prozessmittel verschmutzte Bauteiloberfläche	P	x		
Auftreten starker Lichteffekte auf Objekt	P	x		
Variierende Helligkeit im Objektbereich	P	x		
Gewählter Bildbereich kleiner als Lagetoleranz	I	x		
Gewählter Lokalisierungsalgorithmus ungeeignet	I	x		
Unrobuster Erkennungsalgorithmus für aktuelles Bauteil	I	x		
Schwellwert für Objekt-Matching zu gering	I		x	x
Uneindeutiges Merkmale bei teilweise symmetr. Bauteilen	I		x	

Tab. 2.1.: Bzgl. der Objektlageerkennung identifizierte Fehler mit relevantem Restrisiko, ihre Herkunft sowie resultierende Störungen. Fehlerherkunft P: prozessbedingt; M: manuelle Eingriffe in den Prozess; I: Bedienerfehler bei Inbetriebnahme

durch Entnahme und Kontrolle einzelner Werkstücke zu überprüfen. Hierdurch können die vom System erwarteten Füllstände und Objektpositionen abweichen.

Wie die Tabelle 2.1 zeigt, führen die meisten der vielen unterschiedlichen Fehler zu einem Misserfolg der Lokalisierung. Da ohne die berechnete Objektlage keine korrekte Bauteilmanipulation möglich ist, kommt der Betrieb zu einem Stillstand. Eine ungenaue Objektlokalisierung oder falsch erkannte Pseudo-Bauteillagen führen häufig zu noch schlimmeren Ausfällen, wie z. B. Kollisionen und Beschädigungen am Robotersystem, den Werkstücken oder der Produktionsanlage.

Bauteilmanipulation:

Auch bei der auf die Lokalisierung folgenden Bauteilmanipulation können diverse Fehler auftreten, wie die Tabelle 2.2 zeigt. Ein großer Anteil dieser Fehler betrifft Abweichungen zwischen der erwarteten und der tatsächlichen Objektlage, infolge derer es zu einem vollständigen Fehlgriff, einer Kollision mit dem Bauteil oder einer unbekannten Bauteillage im Greifer kommen kann. Weicht die erwartete Lage einer Bauteilaufnahme ab, kann ein ungenaues Ablegen folgen. Die übrigen Fehler führen häufig zum Verrutschen oder Verlieren des Werkstücks beim Transport oder zu einer Kollision von Roboter oder Greifer mit der Peripherie. Angegeben sind die jeweils zuerst eintretenden Fehlerfolgen. Die Abbildung 2.5 verdeutlicht, wie sich diese über verschiedene Störungen bis zu einem Ausfall des Systems oder einem anderen Schaden fortsetzen können. Die genaue Abfolge ist von den Randbedingungen der Applikation und dem auftretenden Fehler abhängig. So kann die aus einer unerwarteten Objektlage resultierende Kollision des Greifers mit einem Bauteil beispielsweise zu einem Not-Stopp führen, wenn die auftretenden Kontaktkräfte die sicherheitskritischen Grenzwerte überschreiten. Tritt lediglich eine Bauteilverschiebung auf, folgt daraus zumeist eine unbekannte Bauteillage im Greifer, die beim Transportieren oder Ablegen zu einer Kollision des Bauteils mit der Peripherie oder einem ungenauen Ablegen führen kann. Letztere kann sich z. B. in Störungen bei nachfolgenden Prozessstationen auswirken oder bei einem eventuellen erneuten

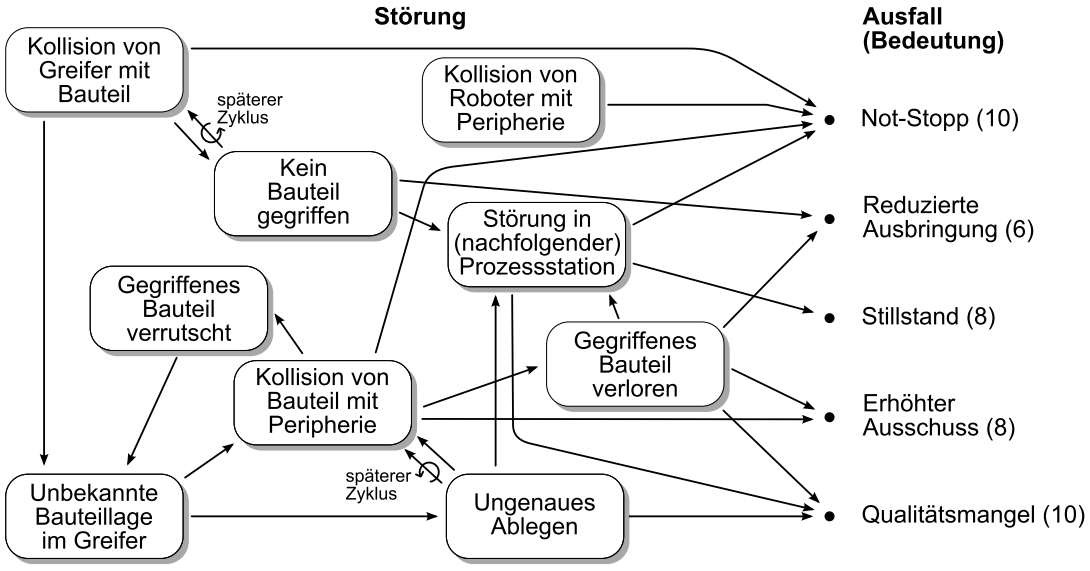


Abb. 2.5.: Fortpflanzungsmöglichkeiten aufgetretener Störungen sowie resultierende Ausfälle

		Mögliche Störung					
Fehler	Fehlerherkunft	Bauteil nicht gegriffen	Kollision des Greifers mit Bauteil	Bauteillage im Greifer unbekannt	Ungenaues Ablegen	Gegriffenes Bauteil verloren / verrutscht	Kollision von Roboter / Bauteil mit Umgebung
Fehlendes Bauteil bei ’blindem’ Abgriff	P, M	x					
Objekt nach Lokalisierung bewegt während Automatikbetrieb	P, M	x	x	x	x		
Abweichende Objektlage bei ’blinder’ Manipulation	P, M, I	x	x	x	x		
Objekt nach Lokalisierung beim Definieren der Manipulation bewegt	I	x	x	x	x		
Pseudo- / ungenau erkannte Objektlage	I	x	x	x	x		
Manipulation fehlerhaft definiert / für Zielposition nicht anwendbar	I	x	x	x	x		
Bauteil durch Betriebsmittel verunreinigt (anhaften / verrutschen)	P				x	x	
Gewählte Greifkraft zu gering	I					x	
Gewählte Greifposition ungeeignet	I					x	
Hindernis in Arbeitsraum eingebracht	M						x

Tab. 2.2.: Bzgl. der Bauteilmanipulation identifizierte Fehler mit relevantem Restrisiko, ihre Herkunft und resultierende Störungen. Fehlerherkunft P: prozessbedingt; M: manuelle Eingriffe in den Prozess; I: Bedienerfehler bei Inbetriebnahme

Greifen des Bauteils. Neben den direkt auftretenden Störungen kann es auch zu ungewollten Folgen in späteren Zyklen kommen, wenn beispielsweise ein ungenau abgelegtes Bauteil in einer Palette andere Nester blockiert. Die Abbildung 2.5 zeigt auch die für die FMEA festgelegte Bedeutung der Ausfälle. Eine der gewichtigsten Folgen ist der Not-Stopp, der im Unterschied zu einem Stillstand bei Detektion eines Fehlers zur Vermeidung von Schäden ausgelöst werden kann. Neben der Unterbrechung der Produktion folgt zwingend ein Bedienereingriff, um den Fehler zu finden, zu beheben und den Wiederanlauf des Systems zu quittieren. Dabei sind eventuelle Schäden an der Anlage oder den Werkstücken zu erkennen und zu beseitigen.

Qualitätsmängel stellen eine weitere sehr gewichtige Fehlerfolge dar. Ursachen sind z. B. fehlende Werkstücke oder Prozessschritte bei der Montage einer Baugruppe oder das kollisionsbedingte Verkratzen eines Bauteils infolge einer fehlerhaften Handhabung. Aufgrund der hohen Bedeutung von Qualitätsmängeln wird die korrekte Lage und Anwesenheit von Einzelteilen bei der Montage über Prozessstationen häufig mittels Sensoren überprüft. Ein Fehler führt in diesem Fall nur zu einem Stillstand und einem erhöhten Ausschuss. Eine reduzierte Ausbringung resultiert, wenn wegen einer misslungenen Bauteilhandhabung ein Bearbeitungszyklus ohne Werkstück auftritt.

Kommunikation mit externen Anlagen

Bei der Kommunikation mit externen Anlagen können neben technischen Problemen, die in dieser Arbeit nicht weiter betrachtet werden, lediglich eine Verwechslung der Output-Ports sowie eine fehlerhafte Signal-Parametrierung auftreten. Beides ist auf Bedienerfehler bei der Inbetriebnahme zurückzuführen, die beim Test der Applikation auffallen sollten. Neben dem Stillstand des Betriebs aufgrund eines ausbleibenden Handshakes, können daraus auch Qualitätsmängel folgen, wenn die Bearbeitung einer Komponente nicht ausgelöst wird. Verbleiben Bauteile wegen eines nicht gestarteten Prozesses in den Übergabestationen, sind auch Kollisionen möglich.

2.3.3. Abgeleitete Möglichkeiten zur Steigerung der Robustheit

Die analysierten Fehler sind für die Robustheit des Automatikbetriebs gleichsam relevant. Jede Störung kann zu einem schweren Ausfall führen, weil dessen Art und Bedeutung von den Randbedingungen der Applikation abhängen. Da die Ausfallwahrscheinlichkeit sowie die entsprechende Bedeutung über Anpassungen am Robotersystem aus wirtschaftlichen Gründen nicht weiter abzuschwächen sind, soll die Robustheit und Autonomie im Dauerbetrieb in dieser Arbeit durch Software-Lösungen erhöht werden.

Die durchgeführte Analyse ergab als Ursache für die auftretenden Fehler neben prozessbedingten Einflüssen und manuellen Eingriffen im Betrieb vor allem auch Bedienerfehler bei der Inbetriebnahme. Da die Programmierung vollständig über die zu entwickelnde MMS erfolgen soll, sind entsprechende Funktionen und Werkzeuge zu integrieren, damit der Bediener keine fehlerhaften Eingaben vornimmt bzw. diese erkennen und beheben kann. Dies betrifft sowohl die Auswahl und Parametrierung der Objektlageerkennung als auch das fehlerfreie Definieren der Manipulationsbewegungen. Hierbei müssen insbesondere Kollisionen vollständig vermieden werden, da ein Not-Stopp des Systems per Definition einen Bedienereingriff erfordert, der über die Software nicht autonom behandelt werden darf.

Manuelle Eingriffe hingegen sind auch durch eine geeignete MMS nicht zu verhindern. Die im System einzusetzende Sicherheitsfunktion soll es dem Bediener explizit ermöglichen, den Arbeitsraum auch während des autonomen Betriebs zu betreten, um Werkstückträger zu wechseln oder Bauteile auf Einhaltung von Toleranzen zu prüfen. Dabei verursachte Veränderungen an Objektlagen und Füllständen müssen im Betrieb fehlerfrei verarbeitet werden. Einzig das Verschieben eines Objekts zwischen der Lokalisierung und der Handhabung ist untersagt, da dies z. T. nicht erkannt und eine mögliche Kollision nicht verhindert werden könnte. Prozessbedingte Einflüsse, wie Fremdlicht oder Schmutz sowie das Auftreten von Störungen bei automatischen Bauteilzuführungen sind ebenfalls nicht vermeidbar, da das Einsatzkonzept der flexiblen Robotersysteme aufwändige Installationen und Anpassungen an den Arbeitsplätzen und Prozessen ausschließt.

Für einen robuster Betrieb sollen Fehler oder Störungen infolge manueller Eingriffe oder prozessbedingter Einflüsse erkannt und geeignet behandelt werden, noch bevor diese einen Ausfall verursachen. Die Abbildung 2.6 verdeutlicht den für diese Arbeit vorgesehenen zweigeteilten An-

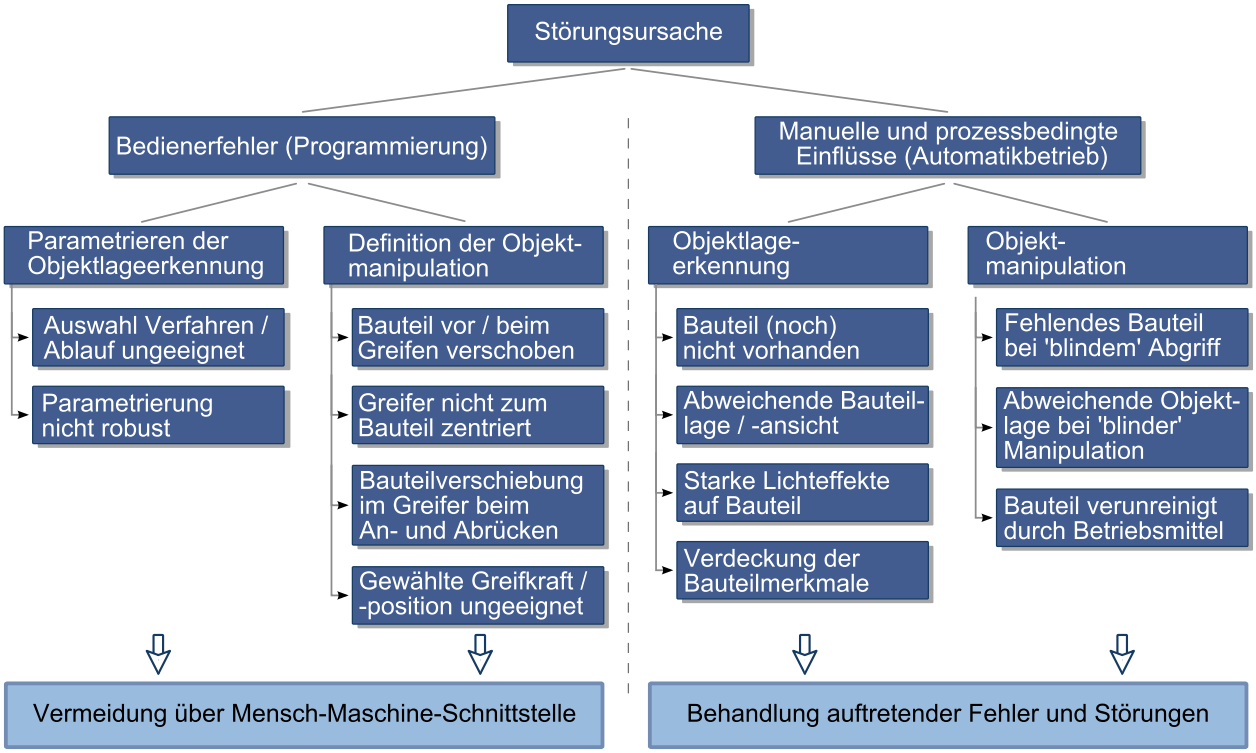


Abb. 2.6.: Analyse der Fehlerursachen mit relevantem Ausfallrisiko hinsichtlich Optionen zur Robustheitssteigerung

satz zur Unterbindung von Bedienerfehler bei der Inbetriebnahme und der Behandlung unvermeidbarer Störungen während des Betriebs.

2.4. Anforderungen an eine MMS für flexible Robotersysteme

Die Anforderungen an eine MMS, die eine breitere Nutzung flexibler Robotersysteme zur Automatisierung der Kleinserienmontage ermöglichen soll, gliedern sich in benötigte Eigenschaften zur Programmierung dieser Systeme sowie in die Anforderungen zur Steigerung ihrer Robustheit im autonomen Montagebetrieb.

2.4.1. Anforderungen zur intuitiven Programmierung von Montageabläufen

Für einen wirtschaftlichen Einsatz in der Fertigung, sollten sich Robotersysteme heutzutage im 2-Schicht-Betrieb innerhalb von circa einem Jahr amortisieren. Einfluss auf dieses Ziel haben neben den Anschaffungskosten vor allem die Verfügbarkeit und die Kosten für die Inbetriebnahme. Dies berücksichtigend ergeben sich die Anforderungen an die MMS basierend auf den Analysen der Robotersysteme sowie den Randbedingungen in der Kleinserienmontage wie folgt:

- **Intuitive Bedienung**

  Die Inbetriebnahme der Robotersysteme für neue Applikationen soll z. B. durch Einsteller erfolgen können. Die MMS muss daher intuitiv verständlich sein und darf kein Expertenwissen in Bildverarbeitung, Robotik oder Programmierung erfordern. Die vollständige und korrekte Programmierung der Systeme soll der Zielgruppe in einer maximal eintägigen Schulung vermittelbar sein.

- **Geringer Inbetriebnahmeaufwand**

  Für eine hohe Flexibilität soll die Inbetriebnahme eines Pick-and-Place-Zyklus innerhalb von circa einer Stunde durch einen Einsteller ermöglicht werden. Das Einrichten einer gängigen Applikation darf entsprechend nicht länger als einen halben Arbeitstag erfordern.

- **Abdeckung des Aufgabenspektrums**

  Für die Automatisierung in der Kleinserienmontage muss die MMS das Einrichten von Funktionen zum Greifen, Transportieren und Ablegen von Bauteilen mit hoher Genauigkeit (< 0.5 mm) ermöglichen. Um variable Objektpositionen erkennen zu können, sind für die Applikation geeignete Kombinationen von Kamerasystemen und Lokalisierungsalgorithmen auszuwählen, ggf. zu verketten und zu parametrieren. Für eine komplexe Interaktion mit Prozessstationen sind eine Kommunikation über digitale Ein- und Ausgänge sowie die Erstellung bedingter Programmabläufe erforderlich.

- **Fehlervermeidung**

  Um einen robusten Automatikbetrieb zu ermöglichen, muss die MMS geeignete Möglichkeiten zur Verfügung stellen, um die mittels der FMEA identifizierten Bedienerfehler bei der Inbetriebnahme vermeiden oder erkennen und beheben zu können.

- **Hardware**

  Das Einrichten der Robotersysteme soll direkt am System ohne separat erforderlichen PC-Arbeitsplatz erfolgen. Hierzu eingesetzte Zusatzhardware muss plattformunabhängig sein und einen nach gültigen Maßstäben sicheren Betrieb erlauben. Zusätzliche Kosten müssen durch einen entsprechend geringeren Inbetriebnahmeaufwand oder reduzierte Anforderungen kompensiert werden.

2.4.2. Anforderungen an eine autonome Behandlung auftretender Störungen

Für einen wirtschaftlichen Einsatz von flexiblen Robotersystemen müssen auch komplexe Applikationen automatisierbar sein, ohne dass es im autonomen Dauerbetrieb häufig zu Ausfällen kommt und ein Bedienereingriff erforderlich ist. Bei der bisherigen manuellen Fertigung wurden eventuelle

Störungen in der Bauteilzuführung oder im Prozess direkt durch den Menschen behoben, der anhand seiner Erfahrungen und Fähigkeiten zumeist auch flexibel auf veränderte Randbedingungen reagieren kann. Für die automatisierte Kleinserienmontage soll nicht nur die Optimierungsphase bei der Inbetriebnahme deutlich verkürzt werden. Die eingesetzten universellen Roboter sind im Bezug auf ihre Robustheit auch anfälliger als gezielt an die Aufgabe angepasste Spezialsysteme. Entsprechend muss die Robustheit für einen autonomen Betrieb zusätzlich geeignet erhöht werden.

Anhand dieser Ziele und den in Abschnitt 2.3 identifizierten Fehler- und Störungsmöglichkeiten ergeben sich die Anforderungen an die Optimierung der Robustheit wie folgt:

- **Robustheit des Automatikbetriebs**

  Um einen robusten Automatikbetrieb zu erreichen, sollen die im Rahmen der FMEA identifizierten und wirtschaftlich nicht ausreichend vermeidbaren Störungen bei der Objektlageerkennung und der Bauteilhandhabung automatisch erkannt werden. Sie müssen unterbrechungsfrei und entsprechend den Anforderungen der Applikation behandelbar sein, um Ausfälle und erforderliche Bedienereingriffe infolge von Störungen zu vermeiden.

- **Geringer Einrichtaufwand**

  Das Einrichten der Störungsbehandlung soll über die MMS des Roboters erfolgen und muss ebenfalls mit geringem Aufwand und ohne erforderliches Expertenwissen durchführbar sein. Für eine hohe Wirtschaftlichkeit ist ein geringes Verhältnis von Aufwand zu Nutzen nötig.

- **Kurze Behandlungsdauer**

  Eine schnelle Behandlung ist erforderlich, um für Applikationen mit kurzen Taktzeiten einsetzbar zu sein.

- **Vermeidung von Folgeschäden**

  Das Verhalten des Gesamtsystems muss nachvollziehbar und vom Einsteller kontrollier- und steuerbar sein, um Schäden an Werkstücken oder Anlagen durch die Störungsbehandlung zu vermeiden. Für den Prozess kritische Funktionen sind ggf. von einem Bediener freizugeben.

- **Hardware**

  Zusätzlich zu integrierende Hardware muss durch die erwarteten Einsparungen im Sinne der Rentabilität des Gesamtsystems gedeckt sein.

2.5. Zusammenfassung

Im Rahmen dieser Arbeit wurden 52 Montageapplikationen aus der Fertigung der Robert Bosch GmbH sowie die enthaltenen Bauteile hinsichtlich der Inbetriebnahmeanforderungen für eine robuste Automatisierung mit flexiblen Robotersystemen untersucht. Eine besondere Herausforderung

stellen die große Bandbreite an Bauteilen, die vielfältigen Varianten ihrer Zuführung zum Prozess sowie das geringe Spiel beim Fügen der Werkstücke dar. Die Schwierigkeiten zum Erreichen eines robusten Automatikbetriebs wurden ausgehend von den Anforderungen für typische Umsetzungsvarianten flexibler Robotersysteme mittels einer FMEA analysiert. Die Ergebnisse zeigen zwei Gruppen von Fehlerursachen, infolge derer es im Dauerbetrieb zu Ausfällen kommen kann. Zum einen treten Störungen durch Bedienerfehler bei der Inbetriebnahme auf. Diese Ursachen sind bei der Konzeption der MMS zu berücksichtigen und durch geeignete Mittel abzustellen. Dies betrifft insbesondere das Einrichten der Lokalisierung sowie das Definieren der Montagebewegungen. Die zweite Fehlergruppe umfasst Störungsursachen, die über die MMS nicht vermeidbar sind, da sie aus dem Prozess, manuellen Eingriffen oder Limitierungen des Robotersystems resultieren und aus wirtschaftlichen Gründen nicht weiter reduzierbar sind. Zur deutlichen Verbesserung der Robustheit, sind Ausfälle über eine geeignete Behandlung der ursächlichen Störungen zu verhindern.

3. Stand der Technik

Das nachfolgende Kapitel gibt einen Überblick über den aktuellen Stand der Technik zur vollständigen Inbetriebnahme von flexiblen Robotersystemen für die industrielle Fertigung. In Abschnitt 3.1 wird zunächst auf die unterschiedlichen in der Industrie und der Forschung angewendeten Methoden zur Programmierung der Roboterbewegungen und des Arbeitsablaufs eingegangen. Ist die Lage von Bauteilen bzw. deren Zielpositionen innerhalb eines Montageprozesses nicht fest zum Robotersystem, muss für die genaue Handhabung eine Objektlokalisierung eingerichtet werden. Da dieser Schritt unabhängig von der Programmiermethode ist, erfolgt in Abschnitt 3.2 eine separate Betrachtung einiger bekannter Ansätze aus Industrie und Forschung. In Abschnitt 3.3 werden verschiedene Methoden und Forschungsansätze zur Erhöhung der Prozessrobustheit bei auftretenden Fehlern und Störungen bzw. zur Reduktion der Ausfallzeiten vorgestellt.

Am Ende der Abschnitte erfolgt jeweils eine Bewertung der vorgestellten Methoden im Bezug auf den aktuellen Kontext und die in Kapitel 2 analysierten Anforderungen. Zusätzlich werden die in dieser Arbeit verfolgten Ansätze motiviert. Da die drei Themengebiete sehr umfangreich sind und eine detaillierte Beschreibung aller Ansätze den Rahmen dieser Arbeit übersteigt, geben die nachfolgenden Abschnitte lediglich einen Überblick über die für den Kontext relevanten Methoden.

3.1. Erstellen von Roboterablaufprogrammen und Bewegungsbahnen

In den letzten Jahren wurden für die Roboterprogrammierung viele, zum Teil auf spezielle Anwendungsgebiete zugeschnittene Ansätze entwickelt und gängige Verfahren erweitert, um die Komplexität und die Anforderungen an den Bediener zu reduzieren. Einen Überblick über die verschiedenen Verfahren und mögliche Klassifizierungen bieten [Lozano-Perez 1983], [Biggs und Macdonald 2003] sowie [Pan u. a. 2010]. An dieser Stelle werden zunächst die Unterschiede zwischen impliziten und expliziten sowie Online- und Offline-Programmierverfahren erläutert und deren klassische Vertreter vorgestellt. Die Erläuterungen zu den aktuell in Industrie und Forschung fokussierten Methoden erfolgt im Anschluss.

Online- vs. Offline-Programmierung

Mit Offline-Programmierung werden Verfahren bezeichnet, bei denen die Programmierung unabhängig vom Roboter, häufig z. B. am PC erfolgt. Typische Vertreter sind die textuelle sowie die grafische Programmierung. Letztere kann weiter in CAD- oder Icon-basierte Verfahren unterteilen

werden. Die Online-Programmierung findet hingegen direkt am oder mit dem Roboter statt. Klassische Methoden sind die Teach-In- oder die Playback-Programmierung. Die Tabelle 3.1 zeigt neben den typischen Vertretern die Vor- und Nachteile der Online- und Offline-Verfahren in Anlehnung an [Spur und Uhlmann 2005].

Als Vorteil der Offline-Verfahren kann die Programmierung bereits vor der Fertigstellung des Systems bzw. ohne Unterbrechung laufender Anlagen erfolgen. Der Bediener kann durch unterschiedliche Programmfunktionen unterstützt werden und vorhandene Daten berücksichtigen. Die Programme können deutlich komplexer sein und sind leichter zu ändern. Damit der Roboter in der realen Umgebung fehler- und kollisionsfrei läuft, müssen alle Positionen und Abmessungen, zumeist in Form von 3D-Modellen, sehr genau bekannt sein. Treten Abweichungen auf, sind bei der Inbetriebnahme aufwändige Korrekturen erforderlich.

Die Vorteile der Online-Verfahren liegen im deutlich geringeren Aufwand bei der Programmierung einfacherer Anwendungen. Die gewünschten Bewegungen werden in der Regel durch direktes Abfahren mit dem Roboter definiert. Die Erreichbarkeit und Kollisionsfreiheit ist damit sicher gestellt, ohne dass hochgenaue Modelle erforderlich sind. Das Ergebnis der Programmierung hängt jedoch stark von der Erfahrung des Bedieners ab, da dieser bei klassischen Methoden nur sehr eingeschränkt vom System unterstützt werden kann. Die erreichbare Genauigkeit entspricht dem Augenmaß des Anwenders.

	Online-Programmierung	Offline-Programmierung
Eigenschaften	+ Positionen, Erreichbarkeit und Kollisionen direkt überprüfbar + geringer Aufwand bei einfachen Applikationen - reale Robotersysteme und Anlagenumgebung erforderlich - eingeschränkte Unterstützung; Qualität z. T. von Fähigkeiten des Bedieners abhängig - Genauigkeit beschränkt durch Auge des Bedieners - mögliche Komplexität der Programme gering	+ Programmierung bereits in der Planungsphase möglich + Unterstützung durch intelligente, rechnerbasierte Hilfsmittel + Programmänderungen einfach durchführbar + Berücksichtigung vorhandener Daten möglich - genaue Modelle von Werkstücken und Roboterzelle nötig - hoher Startaufwand
Typische Vertreter	. Teach-In . Play-Back	. Textuelle Programmierung . Graphische Programmierung · Iconbasierte Programmierung · CAD- / Modell-basierte Progr.

Tab. 3.1.: Vergleich von Online- und Offline-Programmierung nach [Spur und Uhlmann 2005]. Bedeutung +: Vorteil, -: Nachteil

Explizite vs. implizite Programmierung
Bei der expliziten Programmierung, auch roboterbasiert genannt, wird definiert, wie eine Aufgabe durch den Roboter auszuführen ist. Der Bediener hat einen direkten oder mittelbaren Einfluss auf den Programmcode und gibt z. B. konkrete Bahnbewegungen und Sollwerte für die Aktoren vor. Im Gegensatz hierzu wird bei der impliziten oder aufgabenbasierten Programmierung angegeben, welches Ziel der Roboter erreichen soll. Die Ausführung wird durch das System auf Basis bereits bekannter Skills und Objekte gesteuert. Die Form der Programmierung ist für den Bediener einfacher und kann z. B. durch Sprachbefehle erfolgen und mit Gesten unterstützt werden. Die Skills können im Vorfeld z. B. durch eine explizite Programmierung erzeugt, durch Vormachen eingelernt oder in Form von Bewegungs- und Greifplanern bereitgestellt werden.

Vertreter klassischer Programmierverfahren
Zu den klassischen Online-Verfahren gehört die **Teach-In-Programmierung**, bei der die Roboterbewegungen durch eine Sequenz von Raumpunkten definiert wird, die bei der Programmausführung der Reihe nach abgefahren werden. Hierzu steuert der Bediener den Roboter im Vorfeld online an die einzelnen Posen und speichert diese gezielt ab. Beim *direkten Teach-In* wird die Kinematik z. B. mittels Kraft-Null-Regelung direkt am Werkzeug manuell geführt. Bei der Verwendung des robotereigenen Handbediengeräts oder ähnlicher Steuerungsmöglichkeiten spricht man vom *indirekten Teach-In*. Für die Ausführung im automatischen Betrieb können verschiedene Bewegungsmodi, wie linear, Punkt-zu-Punkt oder Interpolation, gewählt werden, mit denen die Roboterbahn zwischen den definierten Posen beeinflusst wird. Die Teach-In-Programmierung eignet sich z. B., wenn die genaue Roboterstellung nur an einzelnen Punkten relevant ist und dazwischen interpoliert werden kann.

Die **Playback-Programmierung** ähnelt stark dem Teach-In. Während der Bediener den Roboter entlang der zu wiederholenden Bahn steuert, werden die Raumpunkte in regelmäßigen zeitlichen Abständen automatisch gespeichert. Die ausgeführte Bewegung muss daher flüssig und permanent entlang der gewünschten Bahn verlaufen, weshalb die Kinematik oder ein lokalisierbares Ersatzwerkzeug zumeist direkt manuell geführt werden. Dieses Programmierverfahren eignet sich besonders für Anwendungen, bei denen der Verlauf ganzer Trajektorien genau vorzugeben ist, wie z. B. bei Lackierarbeiten.

Zu den Offline-Verfahren gehört die **textuelle Programmierung**, bei der das Roboterprogramm am PC über einen Texteditor in einer generischen Programmier- oder einer roboterspezifischen Steuersprache erstellt wird. [Biggs und Macdonald 2003] gibt einen Überblick über die jeweiligen Unterschiede. Diese Form der Programmierung ist zumeist sehr aufwendig und erfordert Erfahrung und Expertenwissen der Syntax. Abhängig vom Funktionsumfang der gewählten Sprache, ermöglicht die textuelle Programmierung oft eine hohe Komplexität der erstellten Roboterprogramme.

Bei der **CAD- oder Modell-basierten Programmierung** wird der Bediener beim Definieren der Bewegungsbahnen durch eine grafische, dreidimensionale Darstellung der Roboterzelle unterstützt.

Dies soll die Offline-Generierung genauer Roboterbewegungen vereinfachen und eine Übertragung des Ablaufs auf die reale Hardware ohne aufwändige Anpassungen ermöglichen. Hierzu ist der Arbeitsraum im Vorfeld zumeist aus CAD-Modellen des Roboters, der umgebenden Anlagen sowie der Objekte und Werkzeuge mit hoher Genauigkeit zu modellieren. Neben Expertenwissen erfordert dies häufig einen großen Zeitaufwand, der sich erst ab hohen Stückzahlen wirtschaftlich rentiert. Als Vorteil dieses Verfahrens sind die Roboterbewegungen bereits offline simulierbar, um die Erreichbarkeit aller Positionen sowie mögliche Kollisionen zu überprüfen. Desweiteren kann die Ausführungsdauer gemessen und das Zellenlayout sowie einzelne Bewegungen optimiert werden.

Die **iconbasierte Programmierung** zählt ebenfalls zu den grafischen Verfahren. Das Programm wird durch Verkettung einzelner Funktionsbausteine erzeugt und in Form von Graphen oder Ablaufdiagrammen dargestellt. Die Programmierung von Funktionssequenzen wird so auch für unerfahrene Bediener möglich. Für die Ausführung auf dem Roboter werden die einzelnen Operatoren mit den erforderlichen Daten wie z. B. den Roboterbewegungen parametriert, sofern keine vorhandenen Skills oder Planer eingesetzt werden.

Wegen einer intuitiven Bedienphilosophie auch bei komplexen Handlungen befindet sich das **Programmieren durch Vormachen** (PdV) im Fokus der Forschung. Bei dieser Methode wird der auf einen Roboter zu übertragende Arbeitsablauf zumeist mehrfach demonstriert und über Sensoren erfasst. Durch eine automatische Analyse der gesammelten Datenreihen sind semantische Teilschritte zu identifizieren, auf bereits bekannte Aktionen zu übertragen und verschiedenartige Zusammenhänge innerhalb des Prozesses zu erkennen. Als Ziel soll abstraktes, systemunabhängiges Wissen erlangt werden, um die konkrete Handlung auch bei ähnlichen Problemklassen anwenden zu können (siehe [Ehrenmann u. a. 2002] und [Dillmann 2004]). Eine wichtige Rolle kommt bei diesem Verfahren dem Bediener zu, der die Aufgabe vormacht. Die automatische Identifikation von temporalen, lokalen oder kausalen Zusammenhängen ist nur möglich, wenn die geltenden Beziehungen innerhalb aller Demonstration annähernd konstant bleiben und alle übrigen Randbedingungen mindestens ein Mal variieren. Weitere Unterscheidungsmöglichkeiten ergeben sich bei diesem Verfahren z. B. im Grad der Abstraktion. Während high-level Ansätze vollständige Aktionen oder sogenannter Skills einlernen, betrachten low-level Ansätze häufig Trajektorien oder die Parametrierung von Basisfunktionen (siehe [Soller und Henrich 2009]). Die zur Datenerfassung eingesetzten Sensoren können variieren und die Ausführung kann durch einen Menschen oder den Roboter erfolgen, der hierzu z. B. per Telemanipulation oder bei geeigneter Aktorik durch manuelles Führen der Kinematik gesteuert wird.

Hybride Programmierverfahren

Zusätzlich zu den klassischen, beschriebenen Methoden wurden viele sogenannte hybride Programmierverfahren durch Kombination von Online- und Offlinemethoden entwickelt, um einzelne Nachteile zu beheben. In der Praxis eingesetzte Vertreter sind z. B. die textuelle Programmierung mit Teach-In oder das sensorgestützte Teach-In. Bei erstem wird das Programm textuell am PC

erstellt, wobei die verwendeten Positionen durch Anfahren mit dem Roboter integriert werden können. Beim sensorgestütztes Teach-In wird der Bediener beim Führen des Roboters durch Sensoren unterstützt, um z. B. einen konstanten Abstand zum Werkstück zu halten oder dessen Oberflächenverlauf exakt zu folgen.

3.1.1. Roboterprogrammierung in der industriellen Praxis

In der industriellen Praxis werden Roboter zumeist mit klassischen Verfahren programmiert, wobei nach [Hahn 2007] die Teach-In-Programmierung am weitesten verbreitet ist. Die Steuerung der Systeme erfolgt dabei in der Regel über ein Handbediengerät (siehe Abbildung 3.1a), mit dem der Roboter relativ zu seiner Basis oder der Werkzeugorientierung bzw. alternativ auch achsweise verfahren werden kann. Spezielle Sensoren oder Kinematiken zum manuellen Führen des Roboterarms werden wegen der hohen Kosten in der Regel nur im Rahmen der Playback-Programmierung für bahnzentrierte Anwendungen eingesetzt.

Ebenfalls weit verbreitet ist die Kombination von Teach-In- und textueller Programmierung. Dieser Ansatz ermöglicht das Erstellen komplexer Abläufe und Funktionen über eine Programmiersprache. Gleichzeitig können alle Bewegungen online, ohne ein hochgenaues Arbeitsraummodell definiert werden. Die Abbildung 3.1b zeigt die Oberfläche des KUKA.OfficeLite, das auf einem PC oder dem Handbediengerät von Kuka (siehe Abbildung 3.1a) zur Programmierung verwendet werden kann. Da die Steuerung vieler Roboter auf einer herstellerspezifischen Syntax beruht, ist die Wahl der Programmiersprache oft durch das eingesetzte System vorgegeben. Gängige Sprachen sind z. B. Val3 von Stäubli, ABB Rapid von ABB, KRL von Kuka oder KAREL von Fanuc.

Bei der Fertigung sehr großen Stückzahlen kommt häufig auch die CAD-basierte Programmierung zum Einsatz, da die Umrüstzeiten durch Offline-Planung und Programmierung reduziert sowie taktzeitkritische Prozesse effizient optimiert werden können. Der erforderliche Aufwand für die Modellierung von Werkstücken und Arbeitsraum ist für KMU häufig zu hoch. Die bekanntesten Vertreter von herstellerspezifischer Programmiersoftware sind nach [Lüdemann-Ravit 2005] unter anderem RobotStudio von [ABB 2013b] (siehe Abbildung 3.2), KR SIM und KUKA Sim von

(a) (b)

Abb. 3.1.: Programmierung von Kuka Robotern: a) Bedienhandgerät (aus [Kazi u. a. 2005]), b) Programmieroberfläche KUKA.OfficeLite (aus [KUKA 2013a])

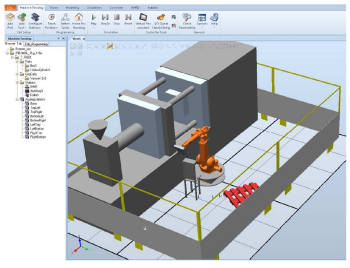


Abb. 3.2.: RobotStudio von ABB (aus [ABB 2013a])

[KUKA 2013b], Stäubli Robotics Suite von [Stäubli 2013] und Roboguide von [Fanuc 2013]. Die Marktführer bei roboterunabhängigen Software-Systemen seien das ehemalige eM-Workplace von Tecnomatix, das heute unter dem Namen Tecnomatix von [Siemens 2013] angeboten wird, sowie das System IGRIP (heutige Version: DELMIA V6) von [DELMIA 2013].

3.1.2. Forschungsansätze zur intuitiven Roboterprogrammierung

Aufgrund der häufigen Anwendung der Teach-In-Programmierung in der industriellen Praxis beschäftigen sich viele Forschungsansätze mit einer Optimierung dieser Methode. Im Vordergrund steht vielfach, das Definieren von Roboterbewegungen zu vereinfachen, da eine zielgenaue Positionierung mit den üblichen Handbediengeräten zeitaufwändig ist und einiger Erfahrung bedarf.

In [Meyer u. a. 2007] wird ein am Handflansch des Roboters angebrachter Kraft-Momenten-Sensor (KMS) verwendet, um den Roboterarm durch manuelles Führen zu steuern. Die vom Bediener aufgebrachten Kräfte und Momente werden über einen Griff auf den Sensor übertragen und in eine Ausgleichbewegung umgesetzt. Die dabei aufgezeichnete Trajektorie kann für Schweiß- oder Klebeprozesse als Kurve über eine grafische Bedienoberfläche (GUI) auf einem PDA nachbearbeitet werden. Zur Einbindung von Werkzeuginteraktionen kann die Ausgabe von Ansteuerungssignalen an einzelne Wegpunkte geknüpft werden. Zusätzlich sind Trajektorienabschnitten verschiedene Parameter zuweisbar, wie z. B. der Winkel zur Senkrechten.

Ein KMS wird in [Wang u. a. 2008] ebenfalls eingesetzt, um einen Roboterarm manuell zu führen und so für Entgratprozesse zunächst wenige Stützpositionen auf einer Werkstückoberfläche zu definieren. Anschließend dient der Sensor dazu, dem Oberflächenverlauf des Bauteils kraftgeführt zu folgen und die vollständige Schleifbahn zu erfassen.

In [Stopp u. a. 2002a] wird erläutert, wie ein Robotersystem für Pick-and-Place-Aufgaben über einen tragbaren Computer implizit programmiert werden kann, um Werkstücke aus verschiedenen Boxen auf ein Transportband zu legen. Der Bediener spezifiziert über den Computer zunächst Art, Anzahl und Reihenfolge der gewünschten Bauteile. Das System greift die Objekte nach einander und bewegt die eingebaute Kamera über ein Transportband. Der Bediener markiert die Zielablageposition auf dem Band mittels eines Laserpointers und der Roboter legt das Werkstück an der entsprechenden Stelle ab, nachdem eine kollisionsfreie Bewegung mittels Pfadplaners berechnet wurde. Damit die impliziten Befehle des Bedieners ausgeführt werden können, sind Umwelt, Objekte und Skills im Vorfeld einmalig zu modellieren und zu parametrieren. In [Stopp u. a. 2002b] wird hierzu ein zwischen Roboterarm und Greifer integrierter KMS verwendet, um zum einen die Bewegungen zum Greifen und Ablegen von Werkstücken zu definieren und das mobile System zum anderen durch die Werkhalle zu steuern. Verbotene Bereiche und relevante Maschinen können dabei über Gesten angezeigt werden.

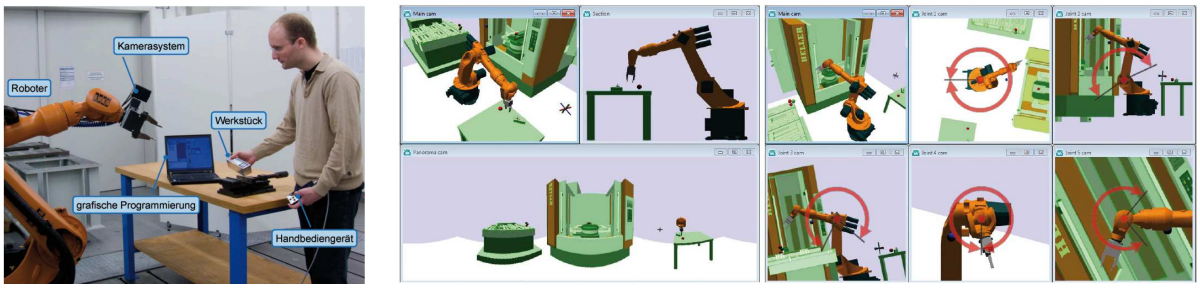
Zur Interaktion mit dem Serviceroboter CORA wird in [Iossifidis u. a. 2002] ebenfalls eine implizite Programmierung eingesetzt. Das einarmige System ist an einem Tisch befestigt und kann

Pick-and-Place-Aufgaben durchführen. Nachdem der Bediener mittels Spracherkennung ein zu übergebendes Objekt spezifiziert hat, führt der Roboter mittels Stereokamera die entsprechende Objektlageerkennung aus. Die Trajektorien für den Abgriff werden über einen Greif- und Bewegungsplaner berechnet. Während die Position der Tischplatte unveränderlich und bekannt ist, müssen alle Objekte für eine autonome Ausführung zuvor als Modell hinterlegt und benannt werden.

Eine andere Möglichkeit der impliziten Programmierung ist in [Matthias u. a. 2006] beschrieben. Im Projekt Porthos wird ein Roboter für Pick-and-Place-Aufgaben an industriellen Maschinen mit einem iconbasierten Ansatz programmiert. Der Bediener erstellt den Manipulationsablauf durch eine einfache Verkettung von Operatoren, wie Greifen oder Ablegen, und kann Ablaufschleifen oder eine Kommunikation mit externen Anlagen integrieren. Aufgrund der intuitiven Darstellung ist kein Expertenwissen erforderlich. Die verwendeten Objekte und Skills sowie das Zellenmodell sind jedoch im Vorfeld an jedem Arbeitsplatz einmalig zu definieren, wofür nach eigenen Angaben Robotik- und Anwendungswissen erforderlich sind. Zunächst werden alle Komponenten der Roboterzelle, wie Roboter, Greifer, Bauteile und Werkstückspeicher, als 3D-Modell aus einer Bibliothek ausgewählt. Ihre genaue Position im Raum wird über ein optisch lokalisiertes Tool bestimmt, mit dem mehrere vom System virtuell vorgegebene Punkte auf der realen Objektoberfläche vermessen werden. Die Trajektorien zum Be- und Entladen externer Anlagen werden durch Teach-In festgelegt, wobei nur eine Bewegung definiert werden muss, wenn die Bahnen übereinstimmen.

Für eine derartige iconbasierte Programmierung wird in [Bischoff u. a. 2002] ein spezieller Styleguide vorgestellt, dessen Kontext sich auf die Programmierung industrieller Roboterapplikationen bezieht. Er zeigt eine Auswahl intuitiv verständlicher Icons für viele Roboteraktionen und gibt Hinweise und Regel zum Erstellen von Bedieneroberflächen für Touchscreens.

Das in [Roßmann u. a. 2009] beschriebene Forschungsprojekt „ProDemo“ setzt ebenfalls eine iconbasierte Programmierung ein, um den Ablauf von industriellen Handhabungsaufgaben zu visualisieren und zusätzliche Aktionen integrieren zu können. Der Basisablauf wird zunächst per Teach-In erstellt. Der Bediener führt hierzu nicht den Roboter sondern das Werkstück auf der gewünschten Bahn durch den Prozess und speichert einzelne Positionen über ein Handbediengerät. Die Objektlage wird dabei direkt oder mittels angebrachter Landmarken über eine Stereokamera erfasst (siehe Abbildung 3.3a). Der Sensor ist am Roboter montiert und wird für eine gute Sichtbarkeit des Objekts automatisch nachgeführt. In der iconbasierten Visualisierung wird jeder gespeicherte Wegpunkt durch ein Symbol repräsentiert. Weitere Funktionen wie Schleifen oder die Kommunikation über eine Schnittstelle lassen sich in den Basisplan integrieren. Ist der Ablauf definiert, erstellt der Bediener in einer Simulationsumgebung ein 3D-Modell der Roboterzelle. Wie in [Matthias u. a. 2006] wählt er vorhandene Komponenten, wie den Roboter, den Greifer oder Maschinen aus einer Bibliothek aus und platziert diese über ein Zeigewerkzeug durch Lokalisierung weniger Oberflächenpunkte an den realen Objekten. Unbekannte Modelle können durch geometrische Formen oder konvexe Hüllen beschrieben werden, deren Position im Raum ebenfalls mit dem Tool bestimmt wird. Auf Basis des Modells kann die erstellte Roboterbewegung auf Kollisionen



(a) **(b)**

Abb. 3.3.: Methoden zur Definition der Robotertrajektorien aus dem Projekt ProDemo: a) Indirektes Definieren der Robotertrajektorien durch Führen des Bauteils (aus [Brecher u. a. 2009]), b) Steuerung des Roboters anhand verschiedener Ansichten in einer Simulationsumgebung (aus [Brecher u. a. 2010])

überprüft und ggf. optimiert werden. Die Bedienung des Systems erfolgt über eine GUI. Ein wichtiger Bestandteil ist dabei ein sogenanntes Tree-Menü, das eine Mischung aus Wizard, Menü und Toolbox darstellt, um möglichst viele Funktionen auf kleinem Raum anzuzeigen. Die Befehle sind hierarchische Strukturierung und in inhaltliche Gruppen gegliedert.

Eine Weiterentwicklung dieses Ansatzes wird in [Brecher u. a. 2010] vorgestellt, indem der Arbeitsraum des Roboters halbautomatisch modelliert wird. Ausgehend von einem zu Beginn dünnen und als hindernisfrei angenommenen Zylinder um das System, wird die Umgebung über einen am Roboter angebrachten Abstandssensor stückweise unter Anleitung des Bedieners erfasst. Konnten in einem Bereich keine Raumpunkte erkannt werden, gilt dieses Segment anschließend als befahrbar. Das System erweitert so den erlaubten Bereich, bis alle relevanten Elemente der Zelle modelliert sind. Eventuell auftretende Fehler werden durch den Bediener behoben. Als zweite Weiterentwicklung kann der Roboter über ein Bedienhandgerät gesteuert werden, dessen Lage im Raum erfasst und auf eine Roboterbewegung übertragen wird. Dies erlaubt eine Steuerung des Systems ohne den Arbeitsbereich zu betreten. Optional ist der Roboterarm auch über die Simulationsumgebung verfahrbar. Hierzu werden verschiedene virtuelle Kameraansichten und Schnittbilder der Roboterzelle angezeigt und Gelenkwinkelbeschränkungen eingeblendet (siehe Abbildung 3.3b). In diesem Modus können auch mögliche Arbeitsraumgrenzen eingetragen werden.

In [Doulgeri und Matiakis 2006] werden ebenfalls Darstellungen des Arbeitsraums eingesetzt, um einen Roboterarm mit Greifer für Pick-and-Place-Aktionen per Telemanipulation zu steuern. Zwei Kameras zeigen den realen Arbeitsraum frontal und seitlich im Überblick. Der Bediener kann Zielpositionen direkt eingeben, den Roboter stückweise bezüglich seiner X-, Y- und Z-Achse verfahren sowie einen Zielpunkt im Bild auswählen. Bei der direkten Positionsvorgabe im Bild wird der zugehörige Schnittpunkt zwischen dem Sichtstrahl der Kamera und der Tischplatte berechnet. Die Z-Koordinate bleibt konstant. Das gewählte Ziel kann vor dem Auslösen der Roboterbewegung über eine Darstellung im Kamerabild überprüft werden.

In [Ng u. a. 2008] wird beschrieben wie Pfade auf Bauteiloberflächen durch manuelle Auswahl innerhalb von Kamerabildern definierbar sind. Um dem Bediener eine intuitive Orientierung auf

einem per Laserscanner erstellten 3D-Modell des Werkstücks zu ermöglichen, werden Texturinformationen einer Kamera auf das per Monitor angezeigte Modell projiziert. Die Programmierung von Trajektorien entlang der Objektoberfläche erfolgt am PC durch Auswahl der gewünschten Pfadpunkte auf dem virtuellen Objekt.

Wie ein Roboterarm nur über Gesten und Sprachbefehle positioniert werden kann, wird in [Tatsuno u. a. 1996] erläutert. Die Bewegungsrichtung wird über Befehle wie „push“, „pull“ oder „lift“ definiert, während die Weite über Adverbien wie „slightly“ oder „moderately“ variiert wird. Zusätzlich kann die Steuerung durch Zeigegesten unterstützt werden. Ein Vergleich beider Kommandos ermöglicht eine Fehlerdetektion.

Ähnlich zum Einsatz von KMS wird in [Wosch und Feiten 2002] die Steuerung eines Roboterarms mittels einer taktilen Haut beschrieben, die um die Kinematik appliziert wird. Berührt der Bediener den Sensor an einer Stelle, wird je nach ausgewähltem Modus eine der Kraft gleichgerichtete oder entgegengesetzte Bewegung ausgeführt.

In [Neto u. a. 2009] wird beschrieben, wie die Programmierung eines Roboterarms per Teach-In durch eine Kombination von Spracherkennung und einem Remote Controller der Wii Konsole von Nintendo durchführbar ist. Dieses Bediengerät ist zur Erfassung der sechsdimensionalen Lage im Raum mit Beschleunigungssensoren und einer optischen Lokalisierung auf Basis von Infrarotlämpchen ausgestattet. Der TCP des Roboters wird durch Betätigung einer Taste am Controller und Vormachen einer Handbewegung so lange in die spezifizierte Richtung bewegt, wie die Taste gehalten wird (siehe Abbildung 3.4a). Gleichzeitig wird das Auftreten von Kollisionen des Werkzeugs über einen KMS überwacht, um den Bediener per Vibration des Controllers zu warnen oder die Bewegung bei Überschreitung einstellbarer Grenzwerte abzubrechen. Zusätzlich können Bewegungen in einzelne Raumrichtungen ganz unterbunden werden. Das Aktivieren von Werkzeugen am Roboter oder das Abspeichern von Armstellungen wird über Sprachbefehle gesteuert.

Eine ähnliche Methode zur Steuerung des Roboterarms mittels optisch lokalisierbaren Zeigewerkzeugs wird in [Hein u. a. 2008] beschrieben (siehe Abbildung 3.4b). Die von einem Bediener ausgeführten Armbewegungen werden mittels eines am Handbediengerät einstellbaren Faktors skalierbar auf den Roboter übertragen. Gleichzeitig wird ein CAD-Modell der Roboterzelle verwendet, um in Echtzeit den Abstand des Roboterarms von möglichen Hindernissen zu berechnen. Auf diese Weise kann der Bediener über ein Tonsignal vor Kollisionen gewarnt, die Robotergeschwindigkeit reduziert oder die Bewegung in kritische Richtungen blockiert werden. Über einen Planer können kollisionsfreie Bahnen zwischen zwei Punkten auch berechnet werden.

Der Einsatz eines haptischen Bediengeräts zur Steuerung eines Fräskopfes mit fünf Freiheitsgraden (DoF) wird in [Denkena u. a. 2006] beschrieben. Der Bediener bekommt das zu bearbeitende Werkstück in einer Simulationsumgebung anhand von CAD-Daten angezeigt und kann Maschinenbewegungen über ein Handbediengerät vorgeben. Dieses misst zur Steuerung die aufgebrachten Kräften und Momenten in allen sechs DoF und kann gleichzeitig mittels Kraftrückkopplung auf die Finger vor möglichen Kollisionen warnen.

(a) **(b)** **(c)**

Abb. 3.4.: Methoden zur Definition der Robotertrajektorien mittels Zeigewerkzeugen: a) Wii Remote Controller (aus [Neto u. a. 2009]), b) Optisch lokalisiertes Werkzeug (aus [Hein und Wörn 2009]), c) Projektion erstellter Schweißbahnen auf die Werkstückoberfläche und Bahnkontrolle mittels virtuellen Roboters (aus [Zäh und Vogl 2006])

In [Estable u. a. 2002] werden Roboterbewegungen per Teach-In über ein sogenanntes cyber-teach-tool definiert. Dieses Werkzeug besteht zum einen aus einem Handbediengerät, das mittels Magnetfeldsensoren lokalisierbar ist und mit dem der Bediener im Arbeitsraum des Roboters per Tastendruck Raumpunkte abspeichern kann. Zum anderen besteht das cyber-teach-tool aus einem ebenfalls lokalisierbaren Helm, an dem eine Cyber-Brille mit Durchsichtmodus sowie ein Mikrofon angebracht sind. Über die Brille können dem Bediener mittels Augmented Reality die gespeicherten Positionen sowie Zusatzinformationen dreidimensional im Arbeitsraum eingeblendet werden. Mittels Spracherkennung kann der Bediener den Bahnpunkten Zusatzfunktionen zuordnen.

Bei einem ähnlichen Ansatz wird in [Zäh u. a. 2004] der Einsatz von Augmented Reality zur Programmierung eines Schweißroboters vorgeschlagen. Einzelnen Stützpunkte der Robotertrajektorie werden ebenfalls über ein lokalisierbares Werkzeug im Arbeitsraum gespeichert und mittels einer Cyber-Brille visualisiert. Die zusätzliche Integration einer Simulationsumgebung mit Robotermodell ermöglicht neben der Kollisionsüberprüfung auch eine virtuelle Projektion des Roboters in den Arbeitsraum. Da der Tool-Center-Point dabei stets dem Zeigewerkzeug folgt, wird eine anschauliche Kontrolle der Roboterpose ermöglicht.

In [Zäh und Vogl 2006] wird eine Weiterentwicklung dieses Ansatzes erläutert, bei der Roboterbahnen über ein Zeigewerkzeug per Teach-In definiert und dabei direkt mittels Laserprojektion auf der Werkstückoberfläche visualisiert werden (siehe Abbildung 3.4c). Die angezeigten Positionen können mit dem Handbediengerät auch direkt auf dem Werkstück ausgewählt und z. B. verschoben werden. Zur Kontrolle der Roboterstellung an den Wegpunkten wird eine Darstellung der Kinematik direkt in einer Kameraansicht des Arbeitsraums überlagert und auf einem Monitor dargestellt. Auf eine Cyber-Brille kann verzichtet werden.

Für dieses Anwendungsszenario wird in [König u. a. 2012] ein akustisches Feedbacksignal vorgeschlagen, mit dem der Bediener direkt beim Teach-In mit einem Werkzeug vor möglichen Singularität, Arbeitsraumgrenzen und Gelenkwinkelbeschränkungen des Roboters gewarnt wird.

In [Bannat u. a. 2009] werden dem Bediener ebenfalls ohne Cyber-Brille verschiedene Informationen über Augmented Reality im Arbeitsbereich eingeblendet. Hierzu wird ein Tisch von oben mittels Kamera und Abstandssensor erfasst und Informationen sowie sogenannte Soft-Buttons von

einem Beamer auf die Arbeitsfläche projiziert. Diese Tasten können per Gestenerkennung ausgelöst werden. Eine weitere Bedienmöglichkeit ergibt sich durch den Einsatz eines Eye-Tracking-Systems, das eine längere Fixierung des Auges auf eine Taste erkennt und diese auslöst. Zusätzlich wird die Interaktion mit einem Roboter über die Sensoren gesteuert.

Um den Arbeitsraum von Schweißrobotern bei der Programmierung nicht betreten zu müssen, wird in [Hahn 2007] die Verwendung eines Bediengeräts vorgeschlagen, mit dem Positionen auf Werkstückoberflächen auch aus der Distanz erfasst werden können. Eingesetzt wird ein Laserdistanzmessgerät, das auf einem Stativ befestigt ist. Mit einem sichtbaren Lichtpunkt wird die gewünschte Position genau anvisiert, während die Koordinate aus einer Abstandsmessung sowie den mit integrierten Sensoren erfassten Horizontal- und Vertikalwinkeln berechnet wird. Über eine GUI kann der Bediener die Positionen überprüfen.

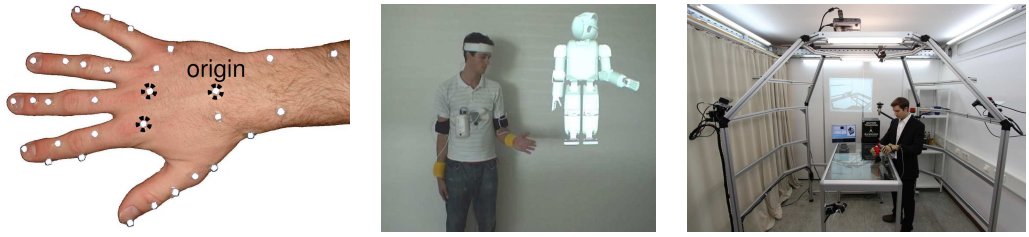
Verschiedene im Projekt „SMERobot“ entwickelte Ansätze zur Programmierung eines Roboters für Holzbearbeitungsaufgaben werden in [Som 2010] beschrieben. Das System besteht aus einem festen Werktisch sowie einem Industrieroboter, an dessen Handflansch verschiedene Werkzeuge montiert werden können. Das Programm zum Aussägen der Konturen einer Tischplatte wird durch Einscannen einer bemaßten Freihandskizze automatisch generiert. Nötige Bohrlöcher lassen sich anschließend über eine Eingabemaske mit intuitiv verständlicher Darstellung aller erforderlichen Werte parametriert. Mäanderförmige Bahnen zum Lackieren des Bretts können bereits anhand der bekannten Kontur generiert werden.

Viele Arbeiten zur Roboterprogrammierung haben sich auch mit der Weiterentwicklung im Bereich des PdV beschäftigt. Einen Überblick über verschiedene Ansätze zeigt [Billard u. a. 2007]. Unterschiede ergeben sich zum einen in den Methoden und Werkzeugen die zur Demonstration sowie zur Erfassung der Bewegungen eingesetzt werden. So erfolgt das Vormachen von Objektmanipulationen in [Calinon und Billard 2008] direkt über einen Roboterarm, der sich manuell führen lässt und über integrierte Sensoren die Trajektorien aufzeichnet. Durch die Verwendung derselben Kinematik für Demonstration und Ausführung entfällt die Transformation der gelernten Bewegungen auf das Zielsystem.

In [Shon u. a. 2005] wird die Bewegungen eines Menschen hingegen direkt über ein Motion-Capturing- System aufgezeichnet. Das erforderliche Mapping der Bewegungen auf den humanioden Roboter erfolgt mittels zweier Gaussian process regression models.

Ganz ähnlich wird auch in [Chang u. a. 2007] Motion-Capturing eingesetzt, um die Bewegung der menschlichen Hand bei der Bauteilmanipulation zu erfassen und aus den Daten auf eine von sechs verschiedenen Griffarten zu schließen. Für die Demonstration wird die Hand mit 31 Landmarken versehen (siehe Abbildung 3.5a), wobei auch mit fünf Markern bereits eine hohe Übereinstimmung der Klassifizierung erreicht wird.

In [Calinon und Billard 2007] wird die vorgemachte Objektmanipulation ebenfalls aus der Demonstration eines Menschen gelernt. Statt den Körper direkt oder über optische Landmarken zu erfassen, werden die Bewegungen über Sensoren am Torso, dem Hinterkopf, den Ober- und Un-



(a) **(b)** **(c)**

Abb. 3.5.: Methoden zur Erfassung menschlicher Bewegungen. a) durch Motion-Capturing mit Landmarken auf der Hand (aus [Chang u. a. 2007]), b) durch Magnetfeldsensoren an Kopf, Körper, Armen und Händen (aus [Calinon und Billard 2007]), c) Erfassung der Handpositionen, Fingerstellungen und Kontaktkräfte über Handschuhe bei gleichzeitiger Lokalisierung der manipulierten Objekte (aus [Jäkel 2013])

terarmen sowie den Händen aufgenommen (siehe Abbildung 3.5b). Die Sensoren messen die Beschleunigung, die Drehraten und das Erdmagnetfeld und erlauben durch Integration der Daten eine Bestimmung der jeweiligen Gelenkwinkeltrajektorien. Nachdem die demonstrierte Handlung erkannt wurde, wird diese durch einen humanoiden Roboter wiederholt. Der Bediener kann die Ausführung dabei kontrollieren und ggf. optimieren, indem er fehlerhafte Bewegungen durch direktes Führen der entsprechenden Roboterkomponente überstimmt.

Zur Erfassung der menschliche Handlungen erfolgt die Demonstration in [Jäkel 2013] innerhalb einer speziellen, mit verschiedenen Sensoren ausgestatteten Umgebung (siehe Abbildung 3.5c). Während die Objektlokalisierung über ein Stereokamerasystem erfolgt, trägt der Mensch bei der Demonstration Handschuhe, mit denen die Handposition per Magnetfeldmessung erfasst wird. Durch integrierte Zusatzsensoren sind gleichzeitig die Fingerstellungen sowie die Kontaktkräfte an deren Spitzen messbar. Vor der Übertragung auf einen humanioden Roboter, können die erlernten Skills in einer Simulation auf einem Monitor überprüft werden.

In [Aleotti u. a. 2004] werden die Handposition und die Fingerstellung ebenfalls mittels Handschuhen erfasst. Die einzulernenden Pick-and-Place-Aktionen werden jedoch nicht real durchgeführt. Die Bewegungen des Bedieners werden auf Handmodelle in eine Simulationsumgebung übertragen. Basierend auf einer dreidimensionalen Darstellung der Hände, der Umgebung sowie verschiedener Objekte wird die Manipulation nur simuliert. Unterstützt wird der Bediener über ein Vibrationsmodul in den Handschuhen, das anzeigt, wenn ein zu greifendes Objekt dicht an den Händen ist bzw. wenn sich ein abzulegendes Objekt dicht über dem Untergrund befindet.

In [Skoglund u. a. 2007] wird über einen Handschuh lediglich die Trajektorie eines Zeigefingers erfasst, um Pick-and-Place-Aktionen für einen Industrieroboter mit einem Vakuumgreifer zu programmieren. Der Manipulationsablauf wird zunächst über sogenannte Handlungsprimitive vorgegeben und anschließend vom Bediener einmalig vorgemacht, wobei dessen Fingerspitze den Greifersaugnapf repräsentiert (siehe Abbildung 3.7). Auf Basis der bekannten Aufgabe und einer Analyse der Endeffektorgeschwindigkeit kann die aufgezeichnete Bewegung automatisch segmentiert und den Handlungsprimitiven zugeordnet werden. Es wird dabei angenommen, dass Anrückbewe-

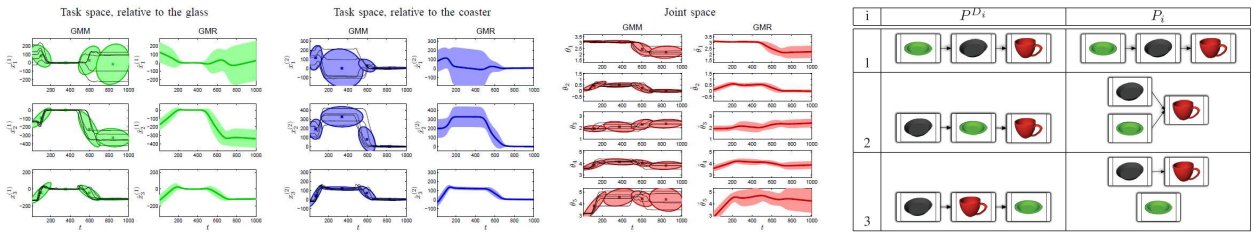
gungen des Greifers stets senkrecht sind und die Höhe der Tischplatte bekannt ist.

Neben der Segmentierung von Bewegungstrajektorien beschäftigen sich unterschiedliche Arbeiten auch mit der automatischen Erkennung der Prozessrandbedingungen. In [Calinon und Billard 2008] wird beschrieben, wie das jeweilige Bezugssystem verschiedener Trajektorien bei Pick-and-Place-Aktion mit beweglichen Objekten automatisch erkannt werden kann. Der Bediener demonstriert die Aufgabe hierzu mehrfach mit variierenden Objektpositionen. Anschließend werden mögliche Abhängigkeiten der Bewegungen über Gaussian Mixture Models im Gelenkwinkelraum des Roboters sowie in den Koordinatensystemen aller erkannten Objekte analysiert (siehe Abbildung 3.6a). Mittels einer pseudoinversen Jakobi-Matrix und der inversen Kinematik wird die Gesamtlösung mit den wahrscheinlichsten Abhängigkeiten berechnet.

In [Pardowitz u. a. 2005] werden sogenannte Task precedence Graphen erläutert, um aus mehrfachen Demonstrationen einer komplexen Handlung, wie einen Tisch zu decken, abstraktes Wissen über Abhängigkeiten und Freiheiten bezüglich der Reihenfolge der Einzelhandlungen zu lernen (siehe Abbildung 3.6b).

Eine Methode, um Handlungen aus einer sich wiederholenden Roboterbewegung effizient durch Vormachen zu definieren, wird in [Soller und Henrich 2009] beschrieben. Der Bediener entscheidet zunächst, dass eine Schleife folgt und gibt die Anzahl der Durchläufe an. Anschließend werden lediglich die ersten beiden sowie die letzte zyklische Teilbewegung demonstriert. Alle übrigen Iterationen werden berechnet.

In [Meeussen u. a. 2007] wird ein spezielles Werkzeug zum Vormachen von Pick-and-Place-Aktionen eingesetzt, um während des Ablegens komplexe Kontaktzustände zwischen dem transportierten Objekt und der Umgebung zu erfassen (siehe Abbildung 3.8). Das über ein Kamerasystem lokalisierte Werkzeug ist bei der Demonstration über einen Kraft-Momenten-Sensor (KMS) fest mit dem zu manipulierenden Gegenstand verbunden. Mittels Partikelfilter werden aus der aufgezeichneten Trajektorie des Werkzeugs, sowie den Kraft- und Momentverläufen die Kontaktzustände sowie zusätzliche geometrische Parameter bestimmt.
Einen ähnlichen Aufbau wird auch in [Okodi u. a. 2008] beschrieben, um die Parameter einer Peg-in-hole-Aktion automatisch durch mehrfaches Vormachen zu bestimmen. Über ein optisch getracktes Werkzeug wird das zu manipulierende Objekt bei der Demonstration geführt, während ein



(a) **(b)**

Abb. 3.6.: Lernen von Randbedingungen beim Programmieren durch Vormachen: a) Lernen der Bezugssysteme von Bewegungen (aus [Calinon und Billard 2008]), b) Lernen von Abhängigkeiten in der Reihenfolge bei Handlungsketten (aus [Pardowitz u. a. 2005])

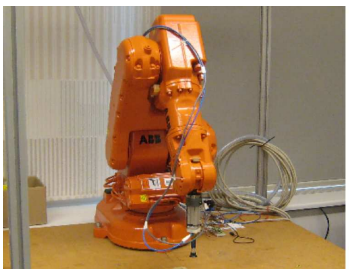


Abb. 3.7.: Erfassung der Fingerbewegung über einen lokalisierbaren Handschuh zur Übertragung der Trajektorien auf einen Roboter (aus [Skoglund u. a. 2007])

Abb. 3.8.: Werkzeug mit KMS zur Erfassung von Kontaktzuständen (aus [Meeussen u. a. 2007])

integrierter KMS die auf das Objekt wirkenden Kräfte und Momente erfasst. Die Segmentierung des Fügevorgangs erfolgt bei der Datenauswertung anhand der auftretenden Kontaktkräfte. Die Lage des Werkstückträgers sowie die An- und Abrückbewegung werden aus den Positionsdaten durch Interpolation gemittelt.

3.1.3. Bewertung der vorgestellten Programmiermethoden

Die industriellen Programmiermethoden eignen sich für die Kleinserienmontage nicht. Gängige Online-Verfahren verlangen viel Erfahrung für die fehlerfreie und robuste Roboterpositionierung sowie für die Integration von Sensorendaten und Kommunikationssignalen. Die Offline-Verfahren erfordern viel Expertenwissen und häufig auch zusätzliche Modelle von der Umgebung und den Objekten, deren Erstellung für den vorliegenden Anwendungsfall zu komplex und aufwändig ist.

Einige der Forschungsansätze verwenden eine implizite Programmierung, die sehr einfach und schnell erfolgen kann. Für die Ausführung müssen jedoch alle relevanten Objekte und Skills bekannt sein. Eine automatische Bahn- und Greifplanung ist wegen fehlender Modelle ähnlich wie eine Simulationsumgebung oder eine Kollisionskontrolle nicht einsetzbar. Die manuelle Erstellung der Umgebungsmodelle wird ebenso wie die halbautomatische Methode für den vorliegenden Anwendungsfall als zu aufwändig angesehen. Der Einsatz von optisch lokalisierten Zeigegeräten wird durch die engen und häufig nur einseitig einsehbaren Arbeitsplätze erschwert. Zusätzlich muss der Bediener darauf achten, die Sicht der Kameras auf das Werkzeug nicht zu verdecken. Magnetische Tracking-Systeme umgehen dieses Problem, sind für die industrielle Anwendung aber zu ungenau und ebenso wie andere zur Steuerung eingesetzte Sensoren sehr teuer. Da eine gleichzeitig einfach und schnell bedienbare sowie kostengünstige Möglichkeit zur Definition von Raumpunkten im industriellen Umfeld fehlt, wird das PdV für den vorliegenden Anwendungsfall wegen der erforderlichen mehrfachen Demonstration als zu zeitaufwändig angesehen. Zusätzlich ist eine ausreichend genaue Lokalisierung der Werkstücke wegen ihrer Struktur und geringen Größe mittels Übersichtskameras nicht sicher möglich. Der Einsatz verbaler Befehle mittels Spracherkennung erfordert eine exakte Kenntnis des unterstützen Vokabulars. Dies kann bei unerfahrenen Bedienern ebenso zu Problemen führen, wie der hohe Störgeräuschpegel im Fertigungsumfeld. Die iconbasierte Darstel-

Methoden zur Definition von Roboterposen und Trajektorien	Anforderungen: ausreichende Genauigkeit	Anwendbarkeit	intuitive Bedienung, geringer Aufwand	geringe zusätzliche Hardwarekosten
Industrielle Methoden zur Definition von Roboterposen	+	+	-	+
Definition von Roboterposen in der Forschung				
Online, mit Roboter				
Calinon und Billard (2008): Direktes, manuelles Führen der Kinematik	-	+	+	-
Meyer u.a. (2007), Wang u. a. (2008): Manuelles Führen über KMS	+	+	o	-
Stopp u. a. (2002): Manuelles Führen der Kinematik über 6D-Maus	+	+	o	o
Hein u. a. (2008), Neto und Pires (2009): Zeigen von Richtungsgesten mittels lokalisierter Handbediengeräte	+	+	o	o
Wösch und Feiten (2002): Steuerung des Arms über taktile Sensoren	-	-	o	-
Tatsuno u. a. (1996): Schrittweise Bewegungen über Sprachbefehle	o	o	o	+
Doulgeri und Matiakis (2006): Telemanipulation mit Übersichtskameras	-	o	+	o
Iossifidis u. a. (2002): Trajektorienberechnung über Bahn- und Greifplaner	+	-	+	+
Offline, ohne Roboter				
Shon u. a. (2004), Chang u. a. (2007): Motion Capturing am Menschen	-	-	o	o
Estable u. a. (2002), Jäkel (2013), Calinon und Billard (2007): Lokalisierung über Magnetfeld-Sensoren	-	+	o	-
Zäh u. a. (2004), König u. a. (2012): Positionsbestimmung durch optisch lokalisierte Werkzeuge	+	o	o	-
Meeussen u. a. (2008), Okodi u. a. (2008): Erfassung von Bewegungsprofilen über KMS an lokalisierten Werkzeugen	+	o	o	-
Ng u. a. (2009): Manuelle Selektierung von Oberflächenpunkten in texturiertem Umgebungsmodell	-	-	+	o
Denkena u. a. (2009): Positionsbestimmung über haptisches Eingabegerät in Simulationsumgebung	+	-	+	-
Hahn (2007): Erfassung von Oberflächenpunkten durch Lasermessgerät	o	-	o	-

Tab. 3.2.: Bewertung der vorgestellten Methoden zur Definition von Roboterposen und Trajektorien. Bewertung +: Anforderung voll erfüllt, o: Anforderung nur mit Einschränkungen erfüllt, -: Anforderung nicht erfüllt

lung des Arbeitsablaufs wird vor dem aktuellen Hintergrund als sehr vielversprechend angesehen und für die zu entwickelnde MMS eingesetzt. Die Tabelle 3.2 zeigt eine Bewertung der vorgestellten Ansätze zur Definition von Roboterposen und Trajektorien. Die beschriebenen Methoden zur intuitiven Programmierung komplexer Handlungsabläufe sind in der Tabelle 3.3 bewertet.

3.1.4. Lösungsansatz

Die in dieser Arbeit vorgestellte MMS basiert auf zwei intuitiv verständlichen Elementen der Bedienerführung. Der Programmablauf wird zunächst durch iconbasierte Programmierung aus anschaulichen Handlungsoperatoren erstellt und bietet eine verständliche Repräsentation der realen

Methoden zur intuitiven Programmierung komplexer Handlungsabläufe	Anforderungen: Anwendbarkeit	intuitive Bedienung, geringer Aufwand	geringe zusätzliche Hardwarekosten
Industriell eingesetzte Programmierverfahren	+	-	+
Roboterprogrammierverfahren in der Forschung			
Stopp u. a. (2002): Implizite Ablaufprogrammierung; Skillausführung über Planer auf Basis von Gesten und Modellen; Griffe z. T. über Teach-In mit 6D-Maus	-	+	+
Iossifidis u. a. (2002): Implizite Ablaufprogrammierung über Sprachbefehle; Skillausführung über Greif- und Bahnplanung	-	+	+
Estable u. a. (2002), Neto und Pires (2009): Teach-In mit Steuerung der Systemfunktionen über Sprachbefehle	o	-	+
Matthias u. a. (2005): Implizite, iconbasierte Programmierung; Einrichten der Skills durch Experten	+	-	+
Brecher u. a. (2009): Erweiterbarer Bewegungsablauf anhand Bauteiltracking; Zellmodellierung für Kollisionsprüfung	-	-	+
Calinon und Billard (2007), Pardowitz u. a. (2005), u. a.: PdV mit mehrfacher, vom Bediener variierter Demonstration	+	o	-
Bedienerführung und -unterstützung in der Forschung			
Zäh und Vogl (2006), Bannat u. a. (2009): AR über Projektion von Informationen oder Tasten in Arbeitsraum	-	+	-
Estable u. a. (2002), Zäh u. a. (2004): Visualisierung von Raumpunkten über AR mittels Cyber-Brille	+	+	-
Estable u. a. (2002), Neto und Pires (2009): Verwendung von Sprachbefehlen	o	-	+
König u. a. (2012): Akustische Warnung vor kritischen Roboterpositionen	+	+	+
Matthias u. a. (2005), Brecher u. a. (2009): Iconbasierte Darstellung des Programmablaufs	+	+	+
Brecher u. a. (2009): Tree-Menü zur Anzeige möglichst vieler Funktionen	+	o	+
Brecher u. a. (2010): Halb-automatische Modellierung der Roboterzelle	+	o	o

Tab. 3.3.: Bewertung der vorgestellten Methoden zur intuitiven Programmierung komplexer Handlungsabläufe. Bewertung +: Anforderung voll erfüllt, o: Anforderung nur mit Einschränkungen erfüllt, -: nicht erfüllt

Montageapplikation. Die hinzugefügten Aktionen werden vollständig wizardbasiert parametriert. Der Bediener folgt hierzu einer dynamisch erzeugten Schrittsequenz, um die relevanten Randbedingungen anzugeben, eine eventuelle Objektlageerkennung einzurichten (siehe Abschnitt 3.2.4) und die Roboterbewegungen durch Teach-In zu definieren. Um eine schnelle und fehlerfreie Bedienung ohne Expertenwissen zu ermöglichen, werden die anzugebenden Parameter über Visualisierungen verdeutlicht und Einstellungen über zugeschnittene Steuerelemente vereinfacht. Der Roboter wird über drei Jog-Controls gesteuert, deren Bewegungsrichtungen auf die unterschiedlichen Anforderungen der verschiedenen Bahnpunktarten zugeschnitten und verständlich visualisiert sind. Eine frei einstellbare Schrittweite ermöglicht sowohl schnelle als auch genaue Bewegungen. Positionen und relevante Eingaben können sofort getestet und ggf. einfach angepasst werden. Durch eine

intuitive Führung des Bedieners und Ausnutzung seines Applikationswissens wird eine schnelle Programmierung komplexer Applikationen auch ohne Expertenwissen ermöglicht. Außer einem Touchscreens für die GUI ist keine zusätzliche Hardware erforderlich.

3.2. Einrichten von Objektlageerkennungen

Eine wichtige Anforderung für die wirtschaftliche Automatisierung der Kleinserienmontage ist die Möglichkeit, durch eine Objektlokalisierung auch Werkstücke handhaben zu können, deren Position innerhalb gewisser Grenzen schwanken kann. Unabhängig von den Eigenschaften des Objekts und den Randbedingungen ergibt sich zumeist ein ähnlicher Ablauf der Lageerkennung. Zunächst werden im Vorfeld der Lokalisierung verschiedene Merkmale, die das Objekt in seinem relevanten Umfeld eindeutig beschreiben, sowie ein eigenes Koordinatensystem definiert. Zur Laufzeit werden entsprechende, in der Bildszene vorhandene Merkmale extrahiert und mit den gesuchten Eigenschaften abgeglichen. Bei ausreichender Übereinstimmung erfolgt die Lageberechnung des aktuellen Objektkoordinatensystems relativ zu den erkannten Merkmalen. Eine Übersicht über unterschiedliche für die Lokalisierung eingesetzte Objektmerkmale zeigt [Haug 2010].

Auch die Inbetriebnahme der Lokalisierungsverfahren sowie das Erstellen der für den Abgleich benötigten Objektmodelle folgen weitestgehend einem ähnlichen Muster, das in Abbildung 3.9 dargestellt ist. Abhängig von den Randbedingungen und dem verwendeten Verfahren können einzelne Schritte entfallen. Vor dem Hintergrund flexibler Robotersysteme sind speziell die Schritte 1 und 4 nur relevant, wenn das System über verschiedene Lokalisierungsverfahren bzw. eine aktive Beleuchtung verfügt. Die Anzahl der einzustellenden Parameter sowie die Auswahl der Merkmale (Schritte 5 - 9) sind sehr stark vom eingesetzten Verfahren abhängig. Durch eine Einschränkung auf spezielle Objekte und Randbedingungen, wie z. B. einfarbige und definierte Hintergründe, kann der Aufwand häufig stark reduziert und die Robustheit gesteigert werden.

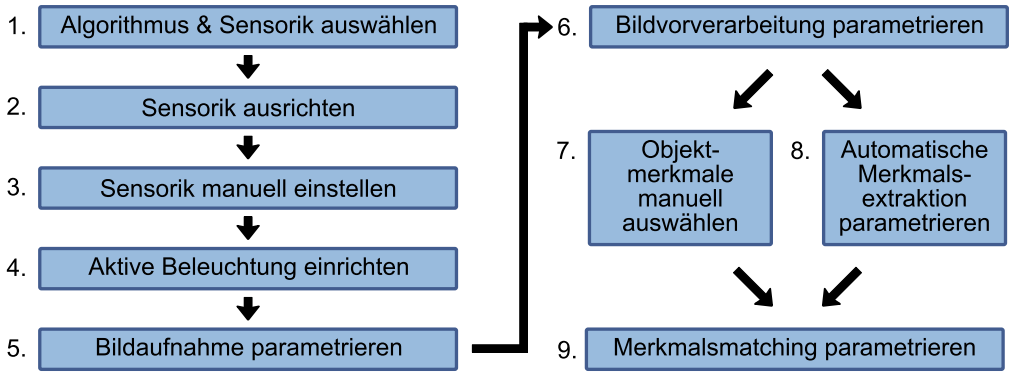


Abb. 3.9.: Typischer Ablauf der Erstellung und Parametrierung von Objektmodellen

3.2.1. Inbetriebnahme von Objektlageerkennungen in der industriellen Praxis

In der industriellen Praxis erfolgt das Einrichten einer Objektlageerkennung durch einen Experten. Die Auswahl eines passenden Lokalisierungsverfahren, einer geeigneten Sensorik sowie eventuell einer aktiven Beleuchtung basiert auf dessen Wissen und Erfahrung. Die eingesetzten Algorithmen sind in der Regel Bestandteil umfassender Bildverarbeitungsbibliotheken, mit denen verschiedene Verarbeitungsabläufe aus Basisoperatoren oder komplexen Funktionen über eine GUI zusammengesetzt werden. Beispiele für kommerzielle Bildverarbeitungsbibliotheken sind z. B. Halcon von MVTec, NeuroCheck vom gleichnamigen Hersteller, Coake von SAC, Vision Pro von Cognex oder Robot-Vision von Isra Vision. Da diese Programme universell für unterschiedliche industrielle Bildverarbeitungsaufgaben einsetzbar sind und sich z. T. auch auf Prüfaufgaben fokussieren, umfasst der Funktionsumfang viele Bereiche, wie z. B. Bildaufnahme, Vorverarbeitung, Segmentierung, Filterung, Transformation oder Klassifikation. Die Komplexität der Bedienung und die Mächtigkeit des Funktionsspektrums hängen von der eingesetzten Bibliothek ab. Die GUI von NeuroCheck erlaubt eine grafische Programmierung, bei der die Operatoren aus einer Liste ausgewählt, per Drag&Drop in einen seriellen Ablauf eingefügt und über Eingabemasken parametriert werden (siehe Abbildung 3.10). Die Befehle lassen sich einzeln ausführen und die Ergebnisse direkt in der GUI anzeigen.

In der GUI der Software Halcon werden Bildverarbeitungsabläufe hingegen textuell programmiert (siehe Abbildung 3.11). Das Anlegen von Variablen und eigenen Funktionen auf Basis der Operatoren erfordert mehr Expertenwissen, ermöglicht aber deutlich komplexere Algorithmen. Die Bibliothek umfasst auch ein Tool, um Templates für ein konturbasiertes Lokalisierungsverfahren durch Markierung relevanter Kanten einfach und schnell zu erstellen.

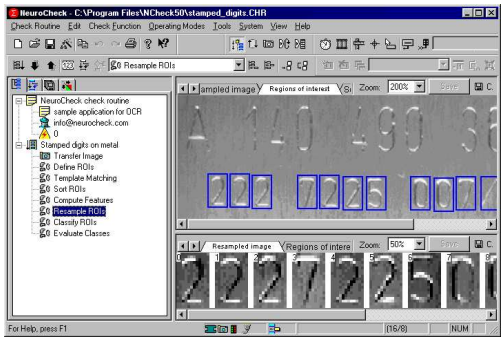


Abb. 3.10.: GUI von NeuroCheck. Verkettung von Funktionen aus einer Bibliothek (aus [DEEM Controls Inc. 2013])

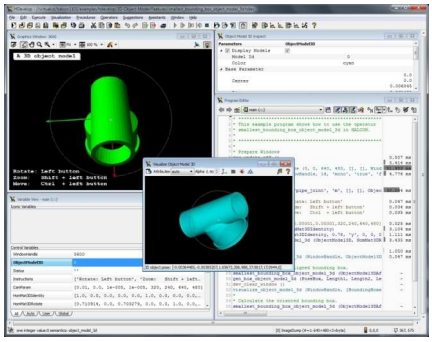


Abb. 3.11.: GUI von Halcon. Textuelle Programmierung komplexer Bildverarbeitungen über eine Funktionsbibliothek (aus [Opli 2013])

3.2.2. Forschungsansätze zum intuitiven Einrichten der Objektlokalisierung

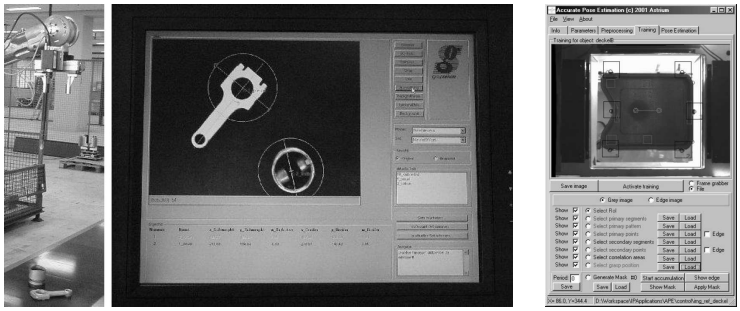
Die für industrielle Roboterapplikationen in der Forschung eingesetzten Verfahren zur intuitiven Inbetriebnahme einer Objektlageerkennung basieren zumeist auf kommerziellen Bildverarbeitungsbibliotheken und verwenden ähnliche Werkzeuge zur manuellen Merkmalssegmentierung.

In [Stopp u. a. 2001] wird eine kommerzielle Software der Graphikon GmbH zur Erstellung von Objektmodellen für eine kantenbasierte 2D-Lageerkennung eingesetzt (siehe Abbildung 3.12a). In Ansichten mit verschiedenen Objektdrehlagen markiert der Bediener jeweils die zur Lokalisierung relevanten Bauteilmerkmale mittels eines Laserpointers auf dem realen Werkstück oder über eine Bedienoberfläche am PC. Eine ähnliche, nicht im Detail beschriebene Methode wird auch in [Estable u. a. 2002] zur Erstellung von 2D-Objektmodellen eingesetzt. Bei beiden Ansätzen wird anhand von Bildern der GUI sowie der Fülle an enthaltenen Bedienelementen vermutet, dass die Bedienung Expertenwissen erfordert (siehe Abbildung 3.12b).

In [Matthias u. a. 2006] werden mit einer Laserlichtschnittkamera, Abstandssensoren und bildgebenden Sensoren gleich mehrere Verfahren zur Objektlokalisierung beschrieben. Die Methoden sind jedoch auf eine zylindrische Werkstückklasse zugeschnitten. Eine Inbetriebnahme oder Anpassung ohne Expertenwissen ist nicht vorgesehen. Lediglich das Raster von Werkstückspeichern kann über eine Assistenzfunktion auch von Einstellern vorgegeben werden.

Die in der Forschung für den häuslichen Bereich entwickelten Serviceroboter verfügen meist nur über ein einziges perzeptives Sensorsystem oder kombinieren standardmäßig alle vorhandenen Sensoren, so dass eine Auswahl von Algorithmen und Hardware nicht erforderlich ist. Sind neue Objekte online einlernbar, werden oft automatische oder semiautomatische Verfahren eingesetzt, bei denen der Bediener ggf. bei der Segmentierung unterstützt oder verschiedene semantische Eigenschaften parametriert.

In [Iossifidis u. a. 2002] erfolgt die Modellierung von Objekten, die vor dem Roboter CORA auf einem Tisch liegen, über einen automatischen Trainingsalgorithmus. Bei der Segmentierung der Gegenstände im Bild wird zunächst die Tischplatte anhand der bekannten Höhe und des großen



(a) **(b)**

Abb. 3.12.: Bedienoberflächen zum Einrichten einer Objektlageerkennung: a) aus [Stopp u. a. 2002a], b) aus [Estable u. a. 2002]

einheitlichen Farbbereichs identifiziert und anschließend nur noch die übrigen Bildregionen betrachtet. Zu den verwendeten Objekterkennungsmerkmalen zählt die Farbverteilung, die sich auf dem Objekt aus den Maxima des Histogramms über den HSI-Farbraum ergibt. Zusätzlich wird aus den rekonstruierten Tiefendaten des Gegenstands eine Ansicht aus der Vogelperspektive berechnet, die invariant bezüglich Betrachtungsrichtung und Abstand ist.

Zur halbautomatischen Erstellung von Objektmodell mit dem Roboter BIRON wird in [Möller u. a. 2005] der Einsatz von kombinierten Zeigegesten und Sprachbefehlen beschrieben. Zusätzlich wird ein iconischer Szenenspeicher eingesetzt, über den bisher aufgenommene Bilddaten der aktuellen Szene zur Verbesserung möglicher Modelltrainings eingesetzt werden können. Für die Segmentierung eines unbekannten Objekts schränkt der Bediener den möglichen Suchbereich des Objekts (ROI) per Zeigegeste ein und nennt per Sprachbefehl die grobe Farbe des Gegenstands. Um das Modell zusätzlich zur Farbinformation und der aktuellen Objektansicht zu erweitern, wird der Szenenspeicher nach aufgenommenen Objektansichten aus anderen Blickrichtungen durchsucht. Gefundene Bildausschnitte ermöglichen eine Erweiterung des Modells sowie eine robustere Erkennung aus verschiedenen Richtungen.

Zum Einlernen neuer Objekte wird in [Lömker 2004] ebenfalls ein automatischer Algorithmus zur Merkmalsextraktion beschrieben. Die Objektsegmentierung erfolgt halbautomatisch mittels eines Differenzbilds, für das der Bediener das relevante Objekt kurzzeitig aus dem Kamerasichtfeld entfernen muss. Um Fehler durch Helligkeitsschwankungen oder andere Unterschiede innerhalb der Kameraaufnahmen zu vermeiden, wird die Greifposition der Hand im Bild mittels Gestenerkennung geschätzt und berücksichtigt.

Ähnlich wird die Kombination eines automatischen Algorithmus zur Merkmalsextraktion mit einer halbautomatischen Objektsegmentierung in [Wersing u. a. 2006] beschrieben. Der Bediener initiiert ein Objekttraining über Sprachbefehle und hält das neue Objekt vor die Stereokamera des Systems (siehe Abbildung 3.13). Aus dem Tiefenbild wird zunächst eine ROI auf Basis des erwarteten Objektabstands berechnet und die haltende Hand mittels Hautfarbenerkennung aus der Objektregion entfernt. Die Objektmerkmale werden anschließend basierend auf der Farbe und räumliche Tiefe jedes Bildpunkts des ROI mittels eines adaptiven szenenabhängigen Filters extrahiert und in einer Eigenschaftsmap gespeichert.

In [Kasper u. a. 2012] wird ein mit verschiedenen Sensoren ausgestattetes Objektmodellierungssystem vorgestellt, mit dem neue Objektmodelle offline unter Einbeziehung eines Bedieners halbautomatisch erstellt werden. Über einen Laserscanner sowie eine hochauflösende Stereokamera können in Verbindung mit einem Drehteller die dreidimensionale Form sowie die Textur des Gegenstands erfasst werden (siehe Abbildung 3.14). In [Becher u. a. 2006] ist beschrieben wie semantische Informationen, wie die Objektklasse oder Eigenschaften und Funktion, mittels Spracheingabe spezifiziert werden können. In [Kasper u. a. 2007] wird zudem ein Datenhandschuh verwendet, um anhand der Greifkräfte und der Fingerstellung eines Menschen mögliche Objektgriffe und eine maximale Transportgeschwindigkeit intuitiv vorzugeben.

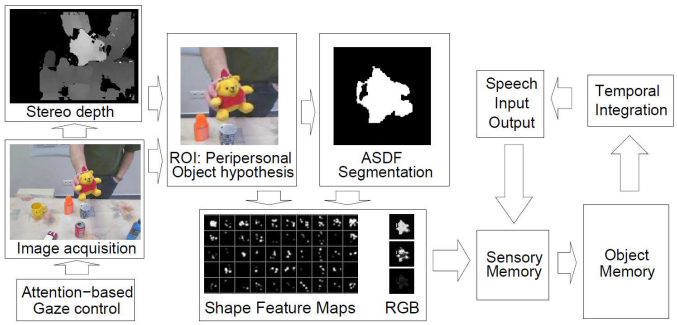


Abb. 3.13.: Halbautomatische Objektmodellierung durch manuelles Präsentieren und automatische Objektsegmentierung (aus [Wersing u. a. 2006])

Abb. 3.14.: Halbautomatische Objektmodellierung über ein Modellierungscenter (aus [Kasper u. a. 2012])

Zur Parametrierung von Objektmodellen über einen Dialog mit dem Roboter BIRON wird in [Lütkebohle u. a. 2009] ebenfalls der Einsatz von Spracherkennung vorgeschlagen. Das System kann selbstständig nach Informationen zu Objekten im Umfeld fragen, um ermöglicht unerfahrenen Nutzern ohne Kenntnis des vorhandenen Funktionsumfangs so eine Interaktion mit dem Roboter.

Ein Problem bei der Parametrierung und Klassifizierung von Objekten über offene Fragen besteht darin, dass der Nutzer die interne Ontologie des Roboters nicht kennt und durch unbewusste Verwendung von Synonymen keine Übereinstimmung erreicht. In [Holzäpfel 2006] werden nur für die Einordnung des Objekttyps offene Fragen verwendet. Für alle weiteren Eigenschaften wird der Nutzer in einem Dialog iterativ nach der Zugehörigkeit des Objekts zu bestehenden Kategorien befragt. Die Fragen ergeben sich anhand der Nutzerantworten aus der Baumstruktur der Ontologie.

3.2.3. Bewertung der vorgestellten Ansätze

Die in der Industrie eingesetzten Bibliotheken vereinfachen und beschleunigen durch eine jeweils zugehörige GUI das Erstellen von Bildverarbeitungsabläufen, so dass der Bediener zumeist keine umfangreichen Programmierkenntnisse benötigt. Das Aufbauen von Algorithmen aus einzelnen Operatoren und deren hohe Anzahl verhindern jedoch eine Bedienung ohne Expertenwissen in Bildverarbeitung. Ein enthaltenes Bedienwerkzeug zum manuellen Segmentieren von Objektmerkmalen für eine kantenbasierte Lageerkennung ist hingegen einfach bedienbar und wird in den Lösungsansatz übernommen (siehe Abschnitt 3.2.4).

Die in der Forschung für industrielle Anwendungen beschriebenen Ansätze basieren auf einem ähnlichen Werkzeug. Die übrigen Schritte der Parametrierung scheinen einen Experten zu erfordern. Die für die Lokalisierung von Gegenständen im häuslichen Bereich entwickelten automatischen Einlernmethoden sind für den Bediener sehr komfortabel. Eine robuste Anwendung der Algorithmen auf industrielle Applikationen mit zum Teil sehr kleinen, reflektierenden und homogenen Werkstücken auf komplexen Untergründen ist nicht möglich.

Für den vorliegenden Anwendungsfall fehlt ein Gesamtkonzept, mit dem der Bediener die Objektlageerkennung vollständig ohne Expertenwissen einrichten kann. Durch das große Bauteilspektrum und die hohen Anforderungen sind ggf. auch Sensoren mit unterschiedlichen Algorithmen zu kombinieren und zu verketten. Für diesen Schritt und die sich anschließende Parametrierung der Verfahren fehlen geeignete Methoden, die auch von unerfahrenen Einstellern bedienbar sind.

3.2.4. Lösungsansatz

Der in dieser Arbeit vorgestellte Ansatz zum Einrichten der Objektlageerkennung wird als Teil der MMS in das verwendete Wizardkonzept integriert (siehe Abschnitt 3.1.4). Die komplexe Auswahl einer geeigneten Sequenz von Kombinationen aus Kameras und Lokalisierungsverfahren wird über einen hinterlegten Entscheidungsbaum auf die Beantwortung weniger Fragen zur Applikation reduziert. Die so identifizierten, erforderlichen Lokalisierungsverfahren werden anschließend schrittweise eingerichtet. Hierzu werden die jeweiligen Parameter in inhaltliche Gruppen aufgeteilt. Als Ziel soll ihre Auswirkung auf den Lokalisierungsprozess anhand von jeweils speziell aufbereiteten Kamerabildern verständlich dargestellt werden. Der Bediener kann die Parameter auf diese Weise ohne Kenntnis ihrer Bedeutung optimieren, indem er ihre Werte variiert und anhand des visuellen Feedbacks intuitiv bewertet, ob das Ergebnis besser oder schlechter wird. Sind alle Parametergruppen eingestellt, kann die Lokalisierung sofort getestet werden.

3.3. Optimierung von Robustheit und Autonomie bei Montageprozessen

Eine Störung bezeichnet die Abweichung eines gemessenen oder erwarteten Zustands von der Realität aufgrund eines aufgetretenen Fehlers. Wird die Abweichung nicht behandelt, führt dies in der Regel zu einem Ausfall oder einem Versagen der betroffenen Komponente. Im Kontext dieser Arbeit geht es bei der Optimierung von Prozessrobustheit und -autonomie vor allem um Error-Recovery. Dies bedeutet, dass es trotz einer auftretender Störungen nicht zu einem Ausfall der Komponente bzw. des Gesamtsystems kommt, weil autonom ein Zustand erreicht wird, bei dem die interne Repräsentation wieder mit der Realität übereinstimmt. Da sich die Bezeichnung Error-Recovery abhängig von der Art der Behandlung mit *Störungskorrektur* oder *automatischer Wiederanlauf nach Störung* nur ungenau übersetzen lässt, wird der englische Begriff im Folgenden beibehalten. Nach [Loborg 1994] unterteilt sich Error-Recovery in drei wesentliche Aspekte: *Störungserkennung*, *Fehlerdiagnose* und *Recovery*.

Störungserkennung: Im ersten Schritt sind auftretende Abweichung zwischen Realität und erwartetem Zustand zu detektieren. Da über eine Störung indirekt auch auf einen Fehler geschlossen werden kann, wird häufig auch der Begriff Fehlererkennung verwendet. Zur Erkennung von

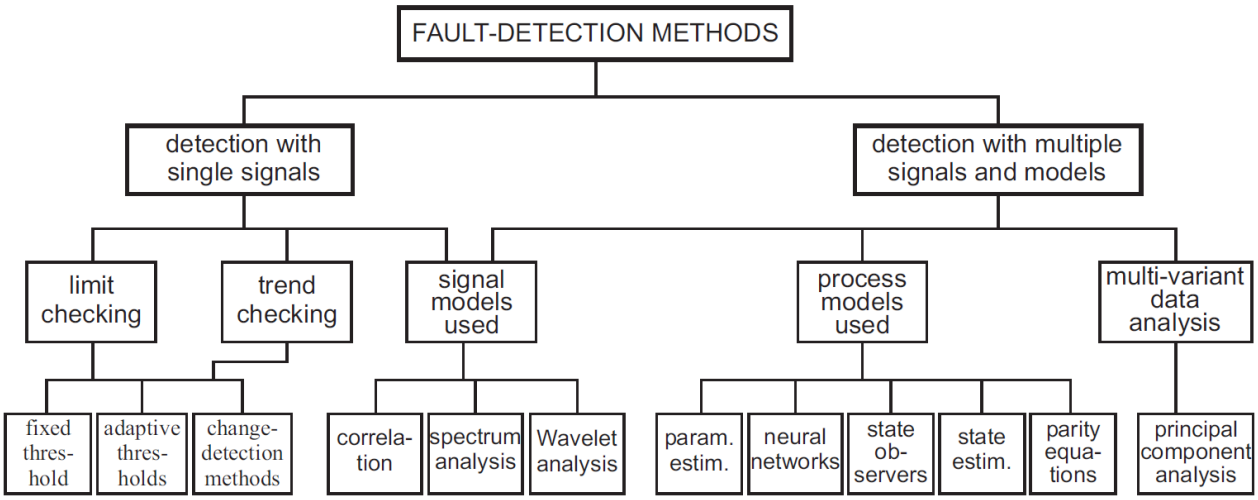


Abb. 3.15.: Fehlererkennungsmethoden (aus [Isermann 2006])

Wertabweichungen können gemessene Signale z. B. mit festen Grenzwerten, redundanten Komponenten oder internen System- bzw. Prozessmodellen abgeglichen werden. Einfache Methoden sind z. B. Schwellwert- oder Trendüberwachungen sowie Plausibilitätsprüfungen. Die Abbildung 3.15 stellt verschiedene Methoden zur Störungserkennung dar. Die Überprüfung kann zyklisch über sogenannte Monitor-Module erfolgen oder eventbasiert an spezielle Funktionen gekoppelt sein. In [Pettersson 2005] ist hierzu eine detaillierte Übersicht dargestellt.

Fehlerdiagnose: Um eine gezielte Behandlung zu ermöglichen und dabei keine zusätzlichen Fehler zu erzeugen, gilt es aus der Störung über den kausalen Zusammenhang auf den ursächlichen Fehler zu schließen. Dies kann in eindeutigen Fällen z. B. nach fest definierten Lookup-Tabellen oder über eine Fehlerbaum-Analyse erfolgen. Möglich sind auch statistische Bewertungen oder der Einsatz von maschinellen Lernverfahren, wenn eine manuelle Verknüpfung zwischen Störungsmerkmalen und Fehlern zu komplex wäre. Die Abbildung 3.16 zeigt verschiedene Verfahren.

Recovery: Dieser Punkt umfasst die nötigen Maßnahmen, um das System wieder in einen stö-

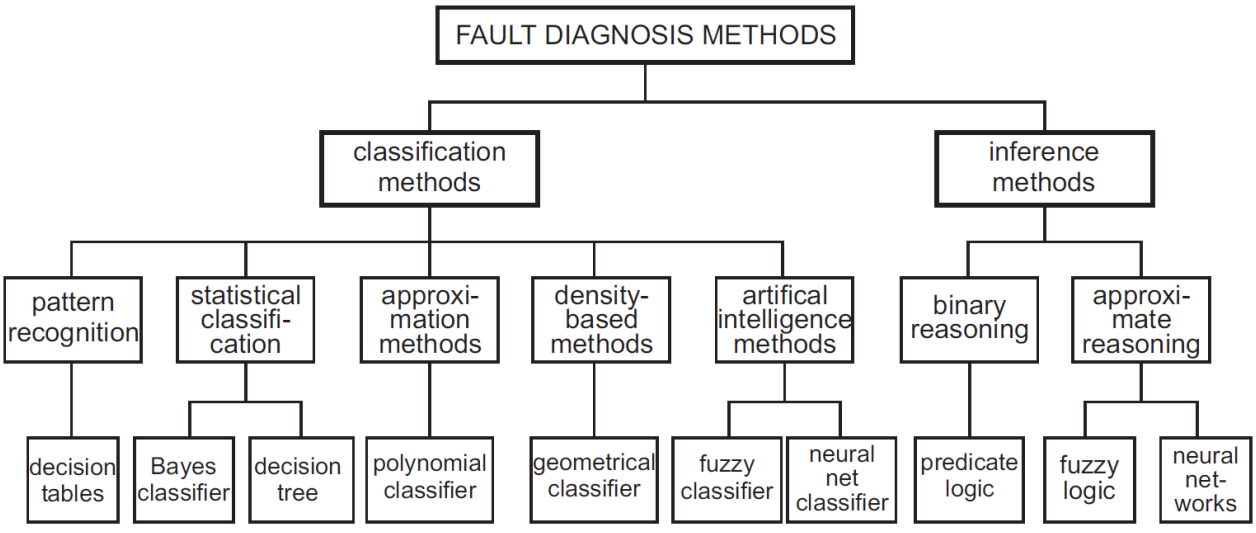


Abb. 3.16.: Fehlerdiagnosemethoden (aus [Isermann 2006])

rungsfreien Zustand zu überführen. Unterschieden werden backward- und foreward-recovery. Bei erstem wird das System auf einen bereits durchlaufenen, störungsfreien Zustand zurückgesetzt. Wurden zwischen dem Auftreten und der Erkennung der Störung z. B. externe Prozesse gestartet oder das Umfeld verändert, ist dies häufig nicht möglich. Foreward-recovery bedeutet, einen alternativen, störungsfreien Zustand zu finden und zu erreichen. Hierzu sind häufig zusätzliche Aktionen auszuführen, um Fehler zu beheben oder einen konsistenten Systemzustand herzustellen. Nach [Shim u. a. 2009] kann Error-Recovery in Fehler- und Störungsbehandlung unterteilt werden. Die Zuordnung ist davon abhängig, ob lediglich ein störungsfreier Systemzustand erreicht oder der Fehler auch erkannt und eliminiert bzw. isoliert wird.

Erreicht die Komponente, in der ein Fehler entdeckt wurde, eigenständig einen gültigen Systemzustand, ohne dass eine Störung von außen erkennbar ist, gilt sie als fehlertolerant. Wird die Störung hingegen von der Komponente nach außen gemeldet und von einem anderen Modul behoben, so vollführt das System als Ganzes den Prozess des Error-Recovery.

Error-Recovery-Methoden lassen sich ihrer Struktur nach unterschiedlich einordnen. Durch Verwendung von sogenannten programming language constructs, wie exception handler oder recovery points wird die Behandlung im Programmcode festgelegt. Ist das zur Diagnose oder Behandlung verwendete Wissen getrennt vom Quellcode spricht man von knowledge based systems. Bei graph based systems beruht der Programmablauf auf einer grafischen Darstellung, wie bei Petri-Netzen oder Automata. Einen Überblick bieten [Shim u. a. 2009], [Loborg 1994] und [Isermann 2006].

3.3.1. Optimierung der Prozessrobustheit in der industriellen Praxis

In der industriellen Montage liegt das Hauptaugenmerk im Bezug auf die Prozessrobustheit vor allem in der Vermeidung von Störungen, indem variable Einflüsse eliminiert werden. In der Regel sind hierzu aufwändige mechanische Vorrichtungen oder Anpassungen nötig, die sich nur bei großen Stückzahlen rentieren. Um variable Objektpositionen zu verhindern, werden beispielsweise spezielle Zuführsysteme mit Vorvereinzelung verwendet, die jedes Bauteil in einer fest definierten Lage bereitstellen. Zusätzlich werden die Prozesse von relevanten externen Einflüssen, wie Störlicht oder Erschütterungen, separiert. Manuelle Eingriffe in den Prozess sind nur durch wenige unterwiesene Mitarbeiter durchzuführen.

Um verbleibende für den Prozess oder die Qualität relevante Störungen zu detektieren, wird die Anlage mit geeigneten Sensoren ausgestattet, die z. B. die Anwesenheit und korrekte Lage zugeführter Bauteile prüfen. Wird eine Störung erkannt, stoppt die Anlage und in der Regel ist ein Bedienereingriff zur Behebung notwendig. Die Fehlerdiagnose erfolgt manuell auf Grundlage von Erfahrungswissen oder eines festen Diagnoseablaufs. Zur Unterstützung bieten einzelne Anlagen Diagnosehinweise, die auf Basis einer logischen Verknüpfung der Ein- und Ausgangssignale erstellt und über eine Mensch-Maschine-Schnittstelle angezeigt werden können. Die Störungsbeseitigung und der Wiederanlauf erfolgen manuell. Abhängig von der Komplexität der Anlage umfasst

das Bedienhandbuch hierzu vorgegebene Abläufe. Automatische Recovery-Prozeduren sind abgesehen von der Ausschleusung von Fehlerteilen nicht üblich.

3.3.2. Forschungsansätze zur Optimierung der Prozessrobustheit

In diesem Abschnitt werden verschiedene für den vorliegenden Anwendungsbereich relevante Forschungsansätze bezüglich Störungserkennung, Fehlerdetektion und Recovery beschrieben.

Zur Detektion verlorener oder fehlerhaft gegriffener Bauteile wird im Projekt Porthos, wie in [Matthias u. a. 2006] beschrieben, die sensorisch erfasste Greiferöffnungsweite ausgewertet. Die Störungsbehebung erfolgt manuell, wobei dem Bediener die vom System erwarteten Füllstände der Bauteilspeicher angezeigt werden, um einen konsistenten Zustand herstellen zu können. Bei der Objektlageerkennung wird eine Störung detektiert, sofern die erfasste Lageabweichung eine maximale Toleranz überschreitet. Die Lokalisierung wird dann an der ungenau erfassten Ist-Lage einmalig wiederholt. Besteht die Störung fort, wird der Ablauf unterbrochen und das Werkstück muss im erlaubten Suchbereich positioniert werden.

In [Iossifidis u. a. 2002] wird erläutert, wie eine fehlgeschlagenen Objektlageerkennung beim Robotersystems CORA durch direkte Unterstützung eines Bedieners behandelt wird. Dieser wird aufgefordert, mit der Hand auf das Objekt zu zeigen. Über die Erkennung der Zeigegeste wird der erwartete Objektbereich berechnet und die Lageerkennung mit reduziertem ROI wiederholt.

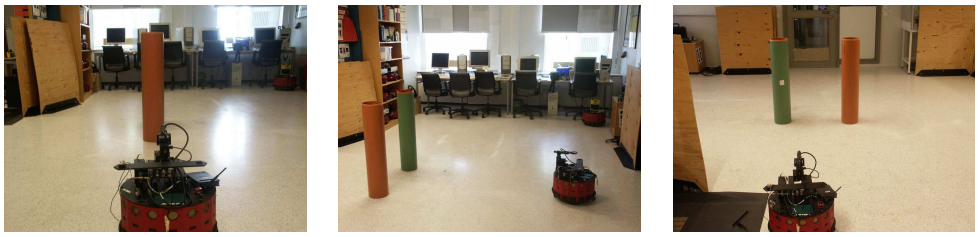
Bei einem ähnlichen Ansatz wird der Bediener in [Takizawa u. a. 2003] ebenfalls in die Fehlerdiagnose und Störungsbehebung eingebunden, wenn die Objektlageerkennung aufgrund des sogenannten Anchoring-Problems fehlschlägt. Bei diesem Problem ist eine eindeutige Lokalisierung des Objekts nicht möglich, weil entweder kein oder mehr als ein potentieller Kandidat gefunden werden. Das System zeigt dann den Suchbereich seiner Kamera über einen tragbaren Touchscreen an und der User kann auf die zum Zielobjekt gehörige Bildregion hinweisen, ohne selbst vor Ort zu sein. Andere Objekte, die den gesuchten Gegenstand eventuell verdecken, kann der Nutzer zu Unterstützung der Segmentierung identifizieren. Werden bei einer Suche mehrere Objekte gefunden, kann der Benutzer der Vorauswahl zustimmen, ein anderes der markierten Objekt auswählen, auf fehlende Ziele hinweisen oder falsche Ergebnisse anzeigen. Takizawa u. a. verwenden spezielle Bildverarbeitungsalgorithmen, die mit dem zusätzlichen Bedienerinput eine Optimierung der Objekterkennung ermöglichen, um ähnliche Störungen zukünftig zu vermeiden.

In [Broxvall u. a. 2004] wird ein weiterer Ansatz beschrieben, um Anchoring-Probleme bei der Objektlageerkennung durch einen mobilen Roboter autonom, mittels wissensbasierter Planung zu lösen. Die dreidimensionalen Zielobjekte unterscheiden sich bei gleicher Form zum Teil nur über die Anwesenheit oder das Fehlen verschiedener Markierungen, die nur aus bestimmten Richtungen sichtbar sind. In den betrachteten Szenarien sind die Objekte in Innenräumen verteilt und können von keiner Position ausreichend überblickt werden. Eine Störung der Objektlageerkennung tritt auf, wenn keine zur Beschreibung passenden Objekte lokalisierbar sind oder wenn sich bei mehre-

ren potentiellen Objektkandidaten nicht alle erforderlichen Merkmale direkt erkennen oder sicher ausschließen lassen. Zur Auflösung einer Störung kann das System die Position im Raum wechseln und die Lokalisierung wiederholen. So können Merkmale identifiziert werden, die zuvor nicht sichtbar waren. Bei Detektion mehrere Kandidaten werden die Objekte solange von verschiedenen Seiten betrachtet, bis alle nötigen Merkmale verifiziert bzw. falsche Zuordnungen eliminiert werden konnten. Bei diesem wissensbasierten Ansatz müssen neben den Zuordnungen der Merkmale zu den verschiedenen Objekten auch deren Geometrie sowie ein Umweltmodell bekannt sein, um geeignete Suchpositionen zu berechnen und die Anwesenheit erforderlichen Merkmale ausschließen zu können.

Dieser Ansatz wird in [Bouguerra 2008] noch erweitert, um den Erfolg von ausgeführten Aktionen anhand von semantischem Domänewissen zu validieren. Hierzu wird ein Partially Observable Markov Decision Process (POMDP) eingesetzt, der die statistische Übereinstimmung eines Objektkandidaten mit der Zielbeschreibung anhand von Wahrscheinlichkeiten berechnet. Ist die ermittelte Überdeckung der Objektmerkmale mit den Zielanforderungen nicht ausreichend sicher, lassen sich mit dem POMDP Zusatzhandlungen planen und ausführen, um die Störung durch einen Informationszugewinn aufzulösen. Bouguerra nutzt diesen Ansatz zum einen, um die Kategorisierung von Gegenständen nach dem Greifen zu überprüfen, wenn zuvor nicht alle Merkmale sichtbar waren. Zum anderen werden Navigationsstörungen beim Wechsel zwischen verschiedenen Innenräumen anhand typischer Einrichtungsgegenstände detektiert (siehe Abbildungen 3.17a bis 3.17c). Die zur Schätzung und Planung erforderlichen statistischen Daten sind manuell einzugeben.

In [Deiterding 2011] werden optimierte Suchbewegungen eingesetzt, um Störungen bei der Objektlageerkennung in industriellen Prozessen möglichst effizient zu beheben. Im betrachteten Szenario kann die Objektposition sehr stark variieren. Die zur Lokalisierung verwendete Kamera erfasst nur einen Bruchteil des möglichen Aufenthaltsbereichs, kann aber über einen Roboter bewegt werden. Schlägt die Lageerkennung fehl, wird angenommen, dass sich das Objekt nicht im Kamerablickfeld befindet. Um eine effiziente und schnelle Störungsbehebung zu erreichen, werden die sequentiell anzufahrenden Suchpositionen für die Kamera auf Basis einer Objektwahrscheinlichkeitskarte berechnet, die aus den Positionen vorheriger, erkannter Werkstücke automatisch generiert wird (siehe Abbildung 3.18). Bei der Optimierung können zusätzlich Kosten für das Bewegen



(a) **(b)** **(c)**

Abb. 3.17.: Autonome Behebung von Störungen bei der Objektlageerkennung durch Positionswechel in [Bouguerra 2008]: a) Position 1: ein Objekt verdeckt, b) Position 2: unterscheidendes Merkmal nicht sichtbar, c) Position 3: Anchoring-Problem gelöst

der Kamera berücksichtigt werden. Vorteile ergeben sich im Vergleich zu herkömmlichen Suchpfaden (z. B. Spirale) hauptsächlich bei einer sehr ungleichen, außerzentrischen Verteilung der Aufenthaltswahrscheinlichkeit. Weiterhin beschreibt Deiterding wie ein auftretender Drift in der Objektposition erkannt und durch frühzeitige Warnung des Bedieners, Berücksichtigung bei der Suche oder durch Eingriff in den Prozess behoben werden kann.

Für die industrielle Montage wird in [Chhatpar und Branicky 2005] ein effizienter Ansatz zur Behandlung von Störungen vorgestellt, die beim Fügen von Objekten in ungenau lokalisierte Aufnahmen auftreten. Hierzu wird im Vorfeld eine sogenannte insertion map erstellt, indem das Prozessnest mit dem Objekt in kleinen Abständen parallel zur Fügerichtung automatisch über einen Roboter abgetastet und die auftretenden Kräfte mit einem Kraft-Momenten-Sensor erfasst werden (siehe Abbildung 3.19). Tritt im Betrieb ein Störungsfall auf, wird dieser über den montierten KMS anhand unerwarteter Kräfte beim Fügen mit dem Roboter detektiert. Ein Algorithmus berechnet aus der insertion map den Objektversatz. Eventuelle Mehrdeutigkeiten werden über gezielte Testbewegungen mittels eines Kalman-Filters auf eine einzige Möglichkeit eingegrenzt. Ist die Lage vom Objekt zur Aufnahme bekannt, wird die angepasste Fügebewegung ausgeführt.

Das gezielte und zumeist kraftgeführte Fügen von Objekten in unterschiedliche Aufnahmen ist in der Robotikforschung ein viel betrachtetes Problem. Die dabei entwickelten Ansätze werden in der Regel bei jedem Ablegevorgang eingesetzt, lassen sich aber auch auf eine reine Anwendung im Fehlerfall übertragen. [Sayler 2011] zeigt eine Übersicht über gängige Methoden und stellt einen taktilen Ansatz vor.

In [Huang u. a. 2008] wird ein modellbasierter Ansatz verwendet, um Fehler beim industriellen Anschließen von Steckern zu erkennen und zu diagnostizieren. Während des Fügevorgangs werden mittels eines KMS die auf den Greifer wirkenden Kräfte sowie die Roboterposition erfasst und mit einem Referenzprofil verglichen. Treten in mindestens einer Messgröße zu große Abweichungen auf, wird der Fügevorgang gestoppt. Über ein mit ausgewählten Testdaten trainiertes Fuzzy-Pattern-Matching wird das stückweise lineare Kraftprofil in mehrere charakteristische Phasen aufgeteilt, um anschließend einem von vier bekannten Fehlerfällen zugeordnet zu werden. Das Verfahren ermöglicht neben der Integration von a priori Wissen eine kurze Rechenzeit bei guten Klassifikationsraten. Ausgenutzt werden bei der Zuordnung spezielle Eigenheiten der Stecker, wie

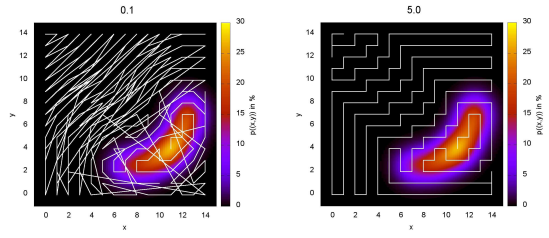


Abb. 3.18.: Optimierung der Suchabfolge über Objektwahrscheinlichkeitskarten. Links keine, rechts hohe Pfadkosten. (aus [Deiterding 2011])

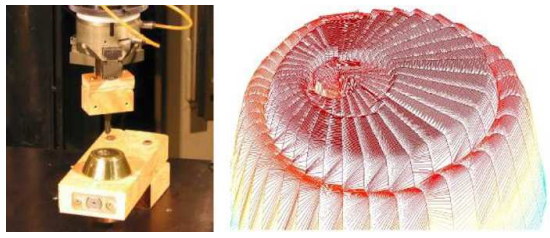


Abb. 3.19.: Iterative Lageschätzung und -korrektur über insertion maps (aus [Chhatpar und Branicky 2005])

z. B. ein einrastender Clipmechanismus, durch den im Kraftverlauf ein charakteristischer Anstieg entsteht (siehe Abbildung 3.20).

Um bei der automatisierten Montage mittels eines Roboters fehlerhaft gegriffene Bauteile schon vor dem Fügen zu detektieren, wird in [Di u. a. 2009] der Einsatz einer zusätzlichen Kamera vorgeschlagen. Diese ist hierzu an der Ablageposition montiert und dient zur Lokalisierung des Greifers, des Bauteils und der Werkstückaufnahme mittels eines Pattern-Matchings (siehe Abbildung 3.21). Tritt eine unerwartete translatorische Verschiebung der Komponenten zueinander auf, wird die Fügebewegung um den entsprechenden Versatz angepasst. Bei rotatorische Abweichungen ist ein Bedienereingriff nötig, da das Bauteil in diesem Fall nicht ausreichend im Greifer fixiert ist.

Neben der sensorischen Erfassung von unerwarteten Systemzuständen gibt es auch eine Reihe Arbeiten, die das Auftreten einer Störung anhand von Unregelmäßigkeiten im Programmablauf detektieren. In [Lee und Chuang 2009] werden kombinatorische Regeln verwendet, um Störungen in industriellen Fertigungsanlagen zu erkennen, deren Programmablauf auf einer Petri-Netz-Darstellung beruht. Sie unterscheiden zwischen Störungen in der Sensorik, der Aktorik sowie in der Sequenz der Fertigungsschritte. Um letztere zu detektieren, wird im Vorfeld für jeden Systemzustand die erwartete Markenanzahl im Petri-Netz über dessen P-Invariante berechnet und im Betrieb mit der tatsächlichen Menge verglichen. Die Erkennung von Störungen in der Sensorik und Aktorik erfolgt über manuell erstellte Logikfunktionen, bei denen die Sensordaten und Steuersignale auf sich widersprechende Belegungen überprüft werden. Wird eine Störung erkannt, muss diese von einem Bediener manuell behoben werden. Zur Unterstützung bei der Fehlerdiagnose erzeugen Lee und Chuang einen AND/OR-Fehler-Baum, mit dem die einer Störung vorausgehenden Systemzuständen und Sensordaten zurückverfolgt werden können.

Anhand der Überschreitung eines für jeden Prozessschritt festgelegten Zeitlimits werden Störungen bei Petri-Netz-basierten Montagesystemen in [Dotoli u. a. 2011] erkannt. Der hierzu eingesetzte Monitor basiert auf einem hybriden Petri-Netz erster Ordnung und überwacht neben der Gesamtzeit für einen Schritt auch die Dauer eventuell auftretender Unterbrechungen. Dies ermöglicht eine Alarmierung noch vor Ablaufen der für die Aufgabe erlaubten Maximaldauer. Für die

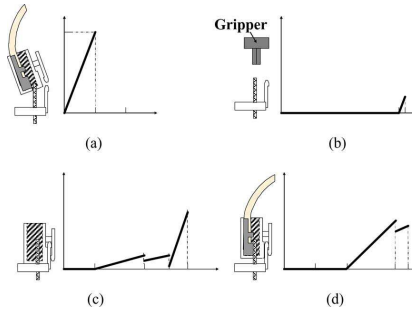


Abb. 3.20.: Diagnose der Störungsursache anhand des beim Fügen aufgetretenen Kraftprofils (aus [Huang u. a. 2008])

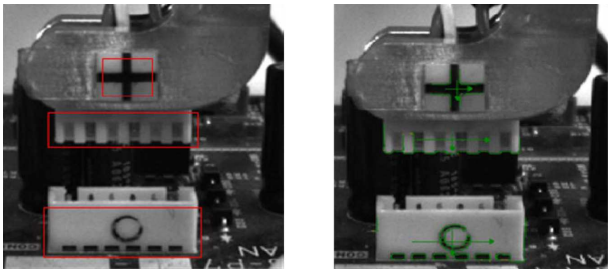


Abb. 3.21.: Erkennung auftretender Störungen vor dem Fügen und Bestimmung der Lageabweichungen mit einer Zusatzkamera (aus [Di u. a. 2009])

Anwendung des Monitors muss der Prozess eine annähernd konstante Ausführungsdauer besitzen. Die zulässigen Zeitabweichungen sind vom Bediener vorzugeben. Der Beginn eines Schritts, eine mögliche Unterbrechung sowie die Wiederaufnahme und die erfolgreiche Beendigung müssen detektierbar sein.

In [Loures u. a. 2006] wird ein Petri-Netz basiertes Monitor-Konzept vorgeschlagen, das neben der Fehlererkennung durch zusätzliche Informationen und statistische Daten auch zur Vorhersagen von sinnvollen Wartungen eingesetzt werden kann. Der Ansatz basiert auf einem Monitormodul mit einem Steuermodell, das die auszuführende Prozesssequenz umfasst. Zusätzlich beinhaltet das Modul ein internes Referenzmodell, das die möglichen Systemzustände beschreibt und den aktuellen Zustand schätzt. Stößt das Steuermodell Aktionen an, werden die entsprechenden Informationen auch an das Referenzmodell übertragen, das die zu erwartende Antwortzeit für das Feedback des Prozesses prognostiziert. Kommt die Antwort nicht wie erwartet oder nicht rechtzeitig, kann auf Fehler geschlossen werden. In diesem Fall wären der Fehler zu Identifizieren und eine Prognose über die Bedeutung sowie sofort einzuleitende Sicherheitsmaßnahmen zu erstellen. Kann das System die Störung nicht selbstständig beheben, wird ein Bediener für die Entscheidung über ein Interface kontaktiert. Über das Konzept hinausgehende Angaben zur Fehlerbeurteilung und Auswahl der Recovery-Schritte sind nicht näher erläutert.

Zur Vereinfachung des Wiederanlaufs einer Maschine nach einem fehlerbedingten Ausfall wird in [Andersson u. a. 2010] die automatische Generierung von sogenannten Restart-States vorgeschlagen. Der Ansatz beschränkt sich auf den Recovery-Vorgang für komplexe, Petri-Netz gesteuerten Anlagen, bei denen Fehler, z. B. in der Hardware, typischerweise manuell diagnostiziert und behoben werden. Es werden insbesondere die Fälle betrachtet, bei denen durch die Störung beschädigte Werkstücke aus dem Prozess entfernt werden müssen. Mit der vorgestellten Methode wird bereits vor Beginn der Produktion berechnet, welche Zustände der beteiligten Maschinen sich für einen Wiederanlauf eignen. Dabei wird berücksichtigt, dass wegen Beschädigung entfernte Werkstücke erneut hinzugefügt und einzelne Bearbeitungsschritte mit diesen Rohteilen eventuell zu wiederholen sind. Der Bediener kann bei einem manuellen Eingriff auf diese automatisch generierten Restart-States zurückgreifen und den Wiederanlauf so vereinfachen und beschleunigen.

Baydar und Saitou beschreiben in [Baydar und Saitou 2001] ebenfalls einen automatisch generierten Recovery-Ansatz zur Störungsbehebung in industriellen Montagezellen. Über eine Monte Carlo Simulation, basierend auf dem dreidimensional modellierten Aufbau der Zelle sowie den statistischen Fehlermodellen aller verwendeten Sensoren, Aktuatoren und Komponenten, werden potentielle Störungen offline simuliert und die entsprechenden Fehlerwahrscheinlichkeiten vorausberechnet. Basierend auf den Ergebnissen werden Recovery-Routinen mittels sogenannter genetischer Programmierung automatisch erstellt und können für die Ausführung auf den Steuerrechner übertragen werden.

3.3.3. Bewertung der vorgestellten Optimierungsansätze

Anhand der in Abschnitt 2.4.2 definierten Anforderungen werden die vorgestellten Verfahren zur Steigerung der Prozessrobustheit mit dem in der industriellen Montage üblichen Vorgehen verglichen und bewertet. Da die Anwendbarkeit der Störungsbehandlung bei einigen Verfahren von der Art der Fehlerdiagnose abhängig ist, werden beide Aspekte in Tabelle 3.5 gemeinsam betrachtet. Die Störungserkennung ist zumeist unabhängig. Die Bewertung erfolgt separat in Tabelle 3.4.

Störungserkennung

In der industriellen Montage werden Anlagen zur Störungserkennung an allen relevanten Stationen mit Sensoren ausgestattet. Um die korrekte Detektion unerwarteter Systemzustände zu ermöglichen, muss das Steuerprogramm um entsprechende Sensorabfragen erweitert werden. Bei kleinen Stückzahlen sind die Kosten für Anschaffung, Montage und Anschluss der erforderlichen Hardware zu hoch. Zusätzlich erfordert die Integration der Sensorsignale in die Anlagensteuerung Programmierkenntnisse, die im aktuellen Kontext nicht vorausgesetzt werden können.

	Anforderungen			
Verfahren zur Störungsdetektion	Störungsabdeckung	intuitive Bedienung, geringer Aufwand	Taktzeitneutralität	geringe zusätzliche Hardwarekosten
Störungsdetektion in der industriellen Montage	+	-	+	-
Störungsdetektion bei der Manipulation in der Forschung				
Matthias u. a. (2006): Überwachung der Greiferöffnungsweite bzgl. Schwellwert	o	+	+	+
Chhatpar und Branicky (2005): Detektion einer max. Fügekraft mittels KMS	o	+	-	-
Huang u. a. (2008): Vergleich der auftretenden Fügekräfte mit Sollprofil	+	+	-	-
Di u. a. (2009): Überprüfung der Bauteillage im Greifer mittels Zusatzkamera	o	o	o	o
Störungsdetektion bei d. Objektlokalisierung in d. Forschung				
Deiterding (2011), Iossifidis u. a. (2002): Überwachung der Ergebnisgüte anhand Schwellwert	+	+	+	+
Takizawa u. a. (2003), Broxvall u. a. (2004): Überprüfung von Kandidatenanzahl und Merkmalsübereinstimmung	+	+	+	+
Matthias u. a. (2007): Überwachung der Bauteilabweichung zur Soll-Lage	-	+	+	+
Bouguerra (2008): Abschätzung einer möglichen Störung mittels POMDP	+	-	+	+
Aufgabenunspezifische Störungsdetektion in der Forschung				
Lee und Chuang (2009): Abgleich von Soll- und Ist-Markenmenge in Petri-Netz; Plausibilitätstest für Signalein- und -ausgänge	-	+	+	+
Dotoli u. a. (2009) Überwachung der Ausführzeiten einzelner Tasks in Petri-Netz	-	+	+	+
Loures u. a. (2007): Vergleich von aktuellem Zustand mit internem, simuliertem Zustandsmodell	o	+	+	+

Tab. 3.4.: Bewertung der vorgestellten Ansätze zur Störungsdetektion: + Anforderung voll erfüllt, o Anforderung teilweise erfüllt, - Anforderung nicht erfüllt

Verfahren für Diagnose und Recovery	Anforderungen: Autonomie	Anwendbarkeit	Störungsabdeckung	intuitive Bedienung, geringer Aufwand	geringe zusätzliche Hardwarekosten
Diagnose und Recovery für die Montage in der Industrie	-	+	+	-	+
Diagnose und Recovery für die Manipulation in der Forschung					
Matthias u. a. (2006): Störungsbehebung durch Bediener; Unterstützung durch Anzeige der erwarteten Füllstände	-	+	+	+	+
Chhatpar und Branicky (2005): Ausgleich von Abweichungen über KMS und Insertion-Map	+	o	o	+	-
Huang u. a. (2008): Fehlerdiagnose durch Vergleich des Kraftprofils mit bekannten Fehlerfällen	o	+	-	+	-
Sayler (2011): Taktiles, modellfreies Fügen	+	o	o	+	-
Di u. a. (2009): Diagnose über Zusatzkamera, Behebung durch gezielte Kompensation des relativen Versatzes	+	+	o	o	o
Diagnose und Recovery für Objektlokalisierung in d. Forschung					
Iossifidis u. a. (2002): Keine Diagnose, Recovery durch Bedienerhinweise auf Zielregion	-	+	+	+	+
Takizawa u. a. (2003): Recovery durch Bedienerauswahl mittels Teleoperation; Optimierung der Lokalisierung	-	-	+	+	+
Matthias u. a. (2006): Behebung durch fest definierte, einmalige Wiederholung; danach Bedienereingriff	o	+	-	+	+
Broxvall u. a. (2004), Bouguerra (2008): Diagnose durch Bewertung über Wissensbasis; Recovery durch Zusatzaktionen mit Informationszugewinn	+	-	+	-	+
Deiterding (2011): Statistische Bewertung statt Diagnose; feste Behandlungsart mit Priorisierung	+	-	-	+	+
Unspezifische Diagnose und Recovery für Montageaufgaben					
Lee und Chuang (2009): Diagnose der Störungsursache über Fehlerbaum	-	-	+	-	+
Andersson u. a. (2011): Behebung durch Bediener. Vereinfachter Wiederanlauf durch Restart-States	-	+	+	+	+
Baydar und Saitou (2001): Fehler durch Simulation statistisch berechnet; Behandlung fest implementiert	+	-	-	-	+

Tab. 3.5.: Bewertung der vorgestellten Ansätze für Fehlerdiagnose und Recovery: + Anforderung voll erfüllt, o Anforderung teilweise erfüllt, - Anforderung nicht erfüllt

Die Forschungsansätze zur Detektion von Manipulationsstörungen verwenden bereits im System integrierte Sensoren für wechselnden Applikation. Die angewendeten Messverfahren führen häufig zu einer erhöhen Taktzeit in jedem Zyklus. Unter Berücksichtigung der geringen Fehlerrate ist dies ebenso wenig akzeptabel wie der hohe Preis einiger Sensoren. Die Überprüfung der Greiferöffnungsweite besitzt diese Nachteile nicht, vermag aber auch nicht alle Störungen sicher zu detektieren.

Die Objektlokalisierung gibt die Anzahl aller erkannten Objektlagen und den Grad der Übereinstimmung wieder, so dass Störungen direkt erkannt werden können.

Diagnose und Recovery
Die Fehlerdiagnose und Störungsbehebung erfolgt in der industriellen Fertigung in der Regel manuell. Da flexible Montageassistenten eine erhöhte Fehlerrate besitzen, würde eine Automatisierung mit diesem Ansatz wegen häufig erforderlichen Bedienereingriffen unwirtschaftlich. Ähnliches gilt für die Forschungsansätze, die den Bediener aktiv in den Error-Recovery-Prozess integrieren und so im Betrieb keine ausreichende Autonomie erreichen. Die Ansätze zur Störungsbehebung durch kraftgeregeltes Fügen kommen bei der Manipulation ohne manuelle Hilfe aus, erfordern jedoch teure Sensoren und einen stabilen Griff der Objekte. Der kamerabasierte Ansatz ist anwendbar, deckt jedoch nicht alle Störungen ab und erfordert zusätzlichen Inbetriebnahmeaufwand.

Bei der Objektlageerkennung ist eine gezielte Diagnose des Fehlers aufgrund der prozessbedingten Einflüsse sehr komplex. Wissensbasierte Verfahren, bei denen der Bediener viele oder komplizierte Daten manuell angeben muss, sind für eine schnelle Inbetriebnahme ohne Expertenwissen ungeeignet. Eine statistische Bewertung anhand einer automatisch erstellten Datenbasis vereinfacht die Inbetriebnahme und wird in dieser Arbeit weiterverfolgt (siehe Lösungsansatz). Der hierzu beschriebene Ansatz umfasst jedoch nur einen einzelnen Fehler und ist mit anderen Störungsursachen nicht vereinbar.

3.3.4. Lösungsansatz

Die Erkennung von Störungen bei der Bauteilmanipulation erfolgt in dieser Arbeit durch Abgleich der Greiferöffnungsweite. Die Methode erfordert keine aufwändige Sensorik oder Inbetriebnahme und ist taktzeitneutral. Da nicht alle fehlerhaft gegriffenen Objekte so erkennbar sind, werden ungenaue Lokalisierungsergebnisse zu Lasten der Erfolgsrate reduziert, indem der nötige Übereinstimmungsgrad zwischen gefundenen und gesuchten Werkstückmerkmalen erhöht wird. Störungen bei der Objektlokalisierung werden direkt anhand der Anzahl der gefundenen Bauteile detektiert.

Eine exakte Fehlerdiagnose ist ohne zusätzlichen Sensor bzw. ohne tiefe Eingriffe in die Lokalisierungsalgorithmen nicht möglich. Da eine falsche Systemreaktion zu unerlaubtem Verhalten führen kann, wird der Bediener in die Auswahl einer korrekten Behandlung eingebunden. Über eine Datenbank mit Behandlungsstrategien für alle Fehlerfälle kann er anhand seines Applikationswissens jeweils mehrere verschiedene Möglichkeiten angeben, mit denen das System auf eine Störung reagieren soll. Deren Ausführungsreihenfolge wird zur Laufzeit mittels eines partially observable markov decision process (POMDP) statistisch optimiert, indem der Fehler geschätzt und die geeignetsten Strategien ausgewählt werden. Die erforderlichen Wahrscheinlichkeiten werden als Erfahrungswissen aus vorangegangenen Fehlern automatisch ermittelt.

Dieser Ansatz ist flexibel und erfordert kein Expertenwissen. Die Behandlung kann bei unerwarteten Fehlern auch im Störungsfall um geeignete Strategien erweitert werden und wird so schrittweise robuster. Die erfahrungsbasierte Optimierung reduziert die Taktzeit und verhindert, dass ungeeig-

nete Behandlungen unnötig ausgeführt werden. Durch parameterarme Strategien ist der Inbetriebnahmeaufwand gering.

3.4. Zusammenfassung

Der Stand der Technik sowie aktuelle Forschungsansätze wurden hinsichtlich einer einfachen und schnellen Inbetriebnahme ohne Erfahrungswissen analysiert. Betrachtet wurden neben dem Erstellen von Ablaufplänen und dem Definieren von Roboterbewegungen auch die Inbetriebnahme komplexer Objektlokalisierungen sowie die Optimierung der Robustheit.

Für die industriell eingesetzten Methoden zum Definieren von Roboterablaufprogrammen sind häufig Kenntnisse über herstellerspezifische Programmiersprachen nötig. Ein fehlerfreies Positionieren des Roboterarms mittels des Bedienhandgeräts erfordert Erfahrung im Umgang mit den Koordinatensystemen. Die intuitivere, grafische Programmierung verwendet detaillierte 3D-Modelle, deren Erstellung bei kleinen Stückzahlen jedoch zu aufwändig ist. Die in der Forschung eingesetzten Methoden vereinfachen die Bedienung, erreichen aber häufig die nötige Positioniergenauigkeit nicht oder setzen teure Sensoren voraus. Daher wird in dieser Arbeit eine iconbasierte Programmierung zur Erstellung der Ablaufpläne mit einer wizardbasierten Parametrierung kombiniert, um den Bediener durch den vollständigen Prozess zu führen. Die Positionierung des Roboters kann ähnlich wie mit einem industriellen Bedienhandgerät sehr genau erfolgen, jedoch werden die Bewegungsrichtungen auf die Anforderungen zugeschnitten und intuitiv verständlich visualisiert.

Die in der industriellen Anwendung zum Einrichten der Objektlageerkennung eingesetzten Bildverarbeitungsbibliotheken besitzen für eine große Anwendbarkeit sehr vielfältige Verarbeitungsfunktionen. Durch die Vielzahl an Möglichkeiten erfordern aber sowohl die Auswahl einer geeigneten Hardware als auch die Parametrierung der Verfahren sehr viel Erfahrungswissen. Bei den Forschungsansätzen wird die Komplexität z. T. durch automatische Algorithmen zur Segmentierung von Objektregionen und relevanten Merkmalen reduziert. Da zumeist aber nur ein Sensorsystem unterstützt wird und der Bediener die automatischen Schritte nicht ausreichend beeinflussen kann, ist eine Übertragung der Ansätze auf die vorliegenden Objekte und Randbedingungen nicht möglich. In dieser Arbeit werden für eine breite Anwendbarkeit mehrere Sensoren und Algorithmen eingesetzt. Die Auswahl durch den Bediener wird über verschiedene Fragen entlang eines Entscheidungsbaums vereinfacht. Die Parametrierung erfolgt in kleinen Schritten anhand von intuitiv verständlichem Feedback auf Basis speziell aufbereiteter Kamerabilder.

Ein robuster Dauerbetrieb wird in der Industrie durch aufwändige Reduktion von variablen Einflüssen und eine zeitaufwändige Optimierungsphase erreicht. Für kleine Stückzahlen ist dieses Vorgehen nicht geeignet. Die Forschungsansätze behandeln jeweils nur einzelne Störungen. Komplexe Szenarien mit verschiedenen Fehlertypen und Behandlungsmöglichkeiten werden nicht betrachtet. In dieser Arbeit wird daher eine komplexe Störungsbehandlung für die Bauteilmanipulation und Objektlokalisierung entwickelt. Über eine Datenbank mit Strategien kann das System schnell

und ohne Expertenwissen an die jeweiligen Randbedingungen angepasst werden. Der Einsatz eines POMDP ermöglicht auf Basis von autonom gelerntem Erfahrungswissen eine Fehlerschätzung und Ausführungsplanung zur Laufzeit, um zeitoptimal verschiedene mögliche Fehlerursachen zu behandeln.

4. Entwicklung einer intuitiven MMS zur Programmierung flexibler Robotersysteme

Das vorliegende Kapitel beschreibt einen Ansatz für eine intuitive Mensch-Maschine-Schnittstelle (MMS) zur Programmierung von flexiblen Robotersystemen für industrielle Pick-and-Place-Aufgaben. Im Fokus steht insbesondere eine schnelle Inbetriebnahme ohne erforderliches Expertenwissen, um eine wirtschaftlichen Automatisierung auch bei kleinen Losgrößen zu ermöglichen.

In Abschnitt 4.1 werden zunächst Grundlagen erläutert, wie die im Kontext übliche Terminologie sowie Anforderungen und Richtlinien an die Gestaltung von MMS. Anschließend erfolgt in Abschnitt 4.2 eine Analyse der vom Bediener bei der Inbetriebnahme festzulegenden Parameter anhand einer Pick-and-Place-Aufgabe. Basierend auf den Ergebnissen sowie den geltenden Anforderungen und Gestaltungsrichtlinien wird eine Ablaufstruktur der MMS skizziert. Der Abschnitt 4.3 beschreibt das auf unerfahrene Nutzer ausgerichtete Konzept zur Bedienerführung, bevor in Abschnitt 4.4 eine intuitive Methode zur Inbetriebnahme komplexer Objektlageerkennungssequenzen vorgestellt wird. Die einfache und genaue Festlegung der Roboterbewegungen wird in Abschnitt 4.5 erläutert.

4.1. Grundlagen zur Gestaltung von MMS

Im Zusammenhang mit Mensch-Maschine-Schnittstellen treten verschiedene Begriffe auf, deren Bedeutung innerhalb der Literatur z. T. variiert. Es folgt daher zunächst eine Erläuterung zur Verwendung relevanter Termini in der vorliegenden Arbeit, bevor allgemeine Anforderungen an die Gestaltung von MMS erläutert werden.

4.1.1. Begriffsdefinitionen zu MMS

Die **Gebrauchstauglichkeit** (engl. *usability*) bezeichnet nach der Norm ISO 9241-11 (1999) „das Ausmaß, in dem ein Produkt durch bestimmte Benutzer in einem bestimmten Nutzungskontext genutzt werden kann, um bestimmte Ziele *effektiv*, *effizient* und *zufriedenstellend* zu erreichen."

An derselben Stelle wird **Effektivität** mit der Genauigkeit und Vollständigkeit beschrieben, mit der ein Benutzer ein bestimmtes Ziel erreicht. Bezogen auf den vorliegenden Kontext, entspricht dies der Qualität des erstellten Roboterprogramms und umfasst z. B. die Fehlerrate oder die erreichte Taktzeit im automatischen Betrieb.

Die **Effizienz** wird als der „im Verhältnis zur Genauigkeit und Vollständigkeit eingesetzte Aufwand“ beschrieben, mit dem der Benutzer ein bestimmtes Ziel erreicht. Im aktuellen Kontext entspricht der Aufwand der zur Programmierung einer Applikation benötigten Zeitdauer. Eventuell durch eine Fehlbedienung verursachte Schäden an Bauteilen oder Maschinen werden in dieser Arbeit nicht bei der Bewertung berücksichtigt, sind aber bestmöglich durch geeignete Maßnahmen zur vermeiden.

Die **Zufriedenheit** umfasst neben der positiven Akzeptanz des Nutzers gegenüber der MMS auch das Erreichen eines beeinträchtigungsfreien Ergebnisses.

Die Eigenschaft **intuitiv** wird in dieser Arbeit nach der in [Mohs u. a. 2007] entwickelten Definition wie folgt verwendet:

> Ein technisches System ist im Rahmen einer Aufgabenstellung in dem Maße intuitiv benutzbar, in dem der jeweilige Benutzer durch unbewusste Anwendung von Vorwissen effektiv interagieren kann.

Mit **Bediener** werden in dieser Arbeit die Einsteller (siehe Abschnitt 2.2.2) bezeichnet, die mit der MMS zielorientiert neue Montageabläufe erstellen oder vorhandene Programme modifizieren. Da die entsprechenden Bereiche der Schnittstelle eine Passwortauthentifizierung erfordern, ist der Bedienerkreis klar begrenzt.

4.1.2. Anforderungen zur Gestaltung einer MMS

Wie in Abschnitt 2.4.2 definiert, bestehen die Anforderungen an die MMS in dieser Arbeit zu großen Teilen darin, eine schnelle Programmierung von robusten und fehlerfreien Montageabläufen zu ermöglichen. Diese Ziele entsprechen der Definition von *Gebrauchstauglichkeit* bereits im Bezug auf die Effizienz und Effektivität. Da sich die Akzeptanz des Nutzers vermutlich ebenso auf beide Kriterien auswirkt, ist auch die Nutzerzufriedenheit eine direkte Zieleigenschaft. Die allgemeinen Anforderungen für eine hohe Gebrauchstauglichkeit gelten dementsprechend auch für die Gestaltung der MMS in dieser Arbeit. Die Norm ISO 9241-110 (2008) empfiehlt die sieben in Tabelle 4.1 beschriebenen Kriterien zur Identifikation der Gebrauchstauglichkeit von interaktiven Systemen und erläutert diese. Von besonderer Bedeutung im Bezug auf eine intuitive Bedienung durch unerfahrene Nutzer sind dabei die ersten vier Eigenschaften.

Für eine **aufgabenangemessene Bedienung** sind die Fähigkeiten der unerfahrenen Nutzer im Bedienkonzept zu berücksichtigen. Eine sinnvolle Unterstützung kann z. B. durch automatische Parametrierungen oder die Angabe von Standardwerten erreicht werden. Für eine gute Übersichtlichkeit sollten nur aktuell relevanten Informationen und Steuerelemente angezeigt und unnötige Dialogschritte vermieden werden.

Eigenschaft	Bedeutung
Aufgabenangemessenheit	System unterstützt Bediener für effektive und effiziente Aufgabenerledigung
Selbstbeschreibungsfähigkeit	Bediener versteht, was an welcher Stelle auf welche Weise ausgeführt werden kann
Erwartungskonformität	Abläufe, Funktionen und Systemrückmeldungen entsprechen den Erwartungen des Bedieners
Fehlertoleranz	Bediener kann fehlerhafte Eingaben ohne oder mit geringem Aufwand korrigieren
Lernförderlichkeit	Bediener wird unterstützt ein konzeptionelles Verständnis des Systems zu erlangen
Steuerbarkeit	Bediener kann Dialogabläufe selbst starten und Richtung sowie Geschwindigkeit beeinflussen
Individualisierbarkeit	Bedienschnittstelle ist an individuelle Fähigkeiten und Bedürfnisse anpassbar

Tab. 4.1.: Kriterien der Gebrauchstauglichkeit nach ISO 9241-110 (2008)

Zum Zwecke der **Selbstbeschreibungsfähigkeit** sind alle Dialoge und Elemente so zu gestalten, dass die Interaktion für den Bediener offensichtlich ist. Hierzu eignet sich der Einsatz von Konzepten und Steuerelementen, die auch unerfahrene Bediener bereits von anderen MMS kennen und daher intuitiv verstehen. Im Rahmen eines frühen Usability-Tests wurde ermittelt, dass die Zielgruppe beruflich sowie privat verschiedene Software-Programme verwendet und daher mit üblichen Konzepten vertraut ist.

Um **erwartungskonform** zu sein, d.h. dem intuitiven Verständnis des Bedieners sowie seinen Erfahrungen zu entsprechen, müssen gleiche oder ähnliche Dialoge und Steuerelemente ein konsistentes Systemverhalten zeigen.

Da unerfahrene Nutzer in der Regel mehr Fehler machen, ist eine hohe **Fehlertoleranz** anzustreben. Sind Fehleingaben nicht zu unterbinden, sollten diese einfach erkenn- und behebbar sein.

Wie an den beschriebenen Eigenschaften ersichtlich ist, trägt ein intuitives Bedienkonzept einen wesentlichen Teil zum Erreichen einer hohen Gebrauchstauglichkeit bei. Die Selbstbeschreibungsfähigkeit und die Erwartungskonformität basieren direkt auf dem intuitiven Verständnis und den Erfahrungen des Nutzers. Gleichzeitig unterstützt die Aufgabenangemessenheit diese beiden Aspekte und ist ein ebenfalls ein wichtiges Kriterium für eine intuitive GUI.

4.2. Strukturierung einer Pick-and-Place-Aufgabe

Zur geeigneten Auslegung einer intuitiven MMS sind zunächst die vom Bediener durchzuführenden Eingaben zu analysieren. Bei der Inbetriebnahme neuer Applikationen stellen die Pick-and-Place-Handlungen die komplexesten Aktionen dar. Um deren Parametrierung schnell und ohne Expertenwissen zu ermöglichen, wurde ein vollständiger Zyklus auf die vom System erforderten

Parameter untersucht. Die Abbildung 4.1 stellt den Ablauf in Teilhandlungen dar und zeigt die vom Bediener anzugebenden Parameter.
Der Zyklus beginnt mit einer Transferbewegung (oben links), um den Roboter ohne Kollision in die Nähe des zu greifenden Bauteils zu verfahren. Dieser Schritt ist nötig, wenn Hindernisse den direkten Anfahrweg versperren. Trajektorien, wie die Transferbewegung, werden durch Angabe einzelner Wegpunkte, der maximalen Bahngeschwindigkeit sowie einer Bewegungsart (z. B. linear oder Punkt-zu-Punkt) definiert. Sofern die Bauteillage nicht durch eine Vorrichtung fixiert ist, muss sie vor dem Abgriff erfasst werden. Abhängig vom Bauteil sowie der Art, wie es der Applikation zugeführt wird, können mehrere Lokalisierungsschritte nötig sein. Jede dieser Lageerkennungen erfordert eine Parametrierung sowie Informationen über die zu detektierenden Objektmerkmale. Die Ausführung erfolgt von einer Suchposition, die relativ zum Arbeitsplatz oder der zuvor erfassten Objektlage definiert ist. Bei einer Lageerkennung innerhalb einer Palette ist die Suchposition zusätzlich vom Raster der einzelnen Prozessnester abhängig. Für die anschließende Objektmanipulation wird ein Anrückpfad bis zur Greifposition relativ zur Bauteillage oder einem Palettennest definiert. Zur Vermeidung von Kollisionen ist die Öffnungsweite des Greifers vorzugeben. Das Zugreifen erfolgt mit angegebener Greifkraft und Richtung, bevor das Bauteil über einen definierten Abrückpfad vom Untergrund angehoben wird. Das Ablegen erfolgt entsprechend.

Eine Analyse der Parameter zeigt, dass sich bei deren Definition durch einen unerfahrenen Bediener unterschiedliche Schwierigkeiten ergeben, wie die Abbildung 4.1 verdeutlicht. Die vom Roboter anzufahrenden Positionen sind zumeist einfach nachzuvollziehen. Damit sich ein Bauteil beim Zugreifen nicht verschiebt, muss der Greifer z. B. genau zentriert und senkrecht zum Werkstück ausgerichtet werden. Eine ausreichend genaue Eingabe der entsprechenden Systemkoordinaten ist

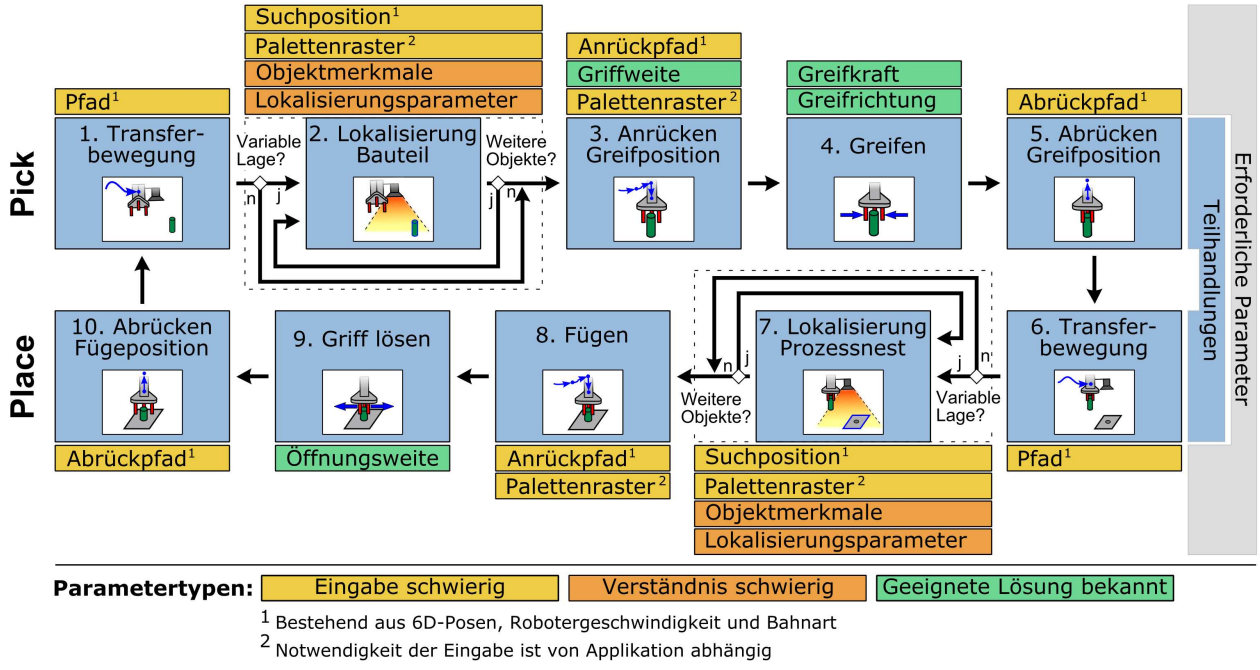


Abb. 4.1.: Aktionen einer typischen Pick-and-Place-Handlung sowie die erforderlichen Parameter

jedoch aufwändig und komplex. Bei der Objektlageerkennung kann das Einstellen der numerischen Parameter hingegen problemlos über gängige Steuerelemente erfolgen. Die Schwierigkeit liegt in der Identifikation geeigneter Werte. Wie die Abbildung 4.1 zeigt, sind für die Manipulation bereits einzelne Parameter verständlich und leicht einzustellen oder es sind geeignete Eingabemöglichkeiten im Stand der Technik beschrieben. Für die anderen beiden, komplexen Parameterkategorien sind in den Abschnitten 4.4 und 4.5 geeignete Methoden für eine schnelle und verständliche Bedienung ohne Erfahrung beschrieben. Eine geeignete Bedienerführung, die den Nutzer auf effiziente Weise durch den vollständigen Programmierprozess leitet und die Selbstbeschreibungsfähigkeit der gesamten MMS erhöht, wird in Abschnitt 4.3 vorgestellt.

4.3. Konzept zur intuitiven Bedienerführung

Die Bedienerführung trägt einen wichtigen Teil zum Erreichen einer effizienten Programmierung ohne Expertenwissen bei. Eine hohe Selbstbeschreibungsfähigkeit unterstützt den Nutzer, das Bedienkonzept der MMS schnell zu verstehen und zu erkennen, welche Angaben an welcher Stelle im Dialog erforderlich sind. Die Verwendung intuitiv verständlicher Komponenten reduziert den vom Bediener zur Orientierung und Informationserfassung benötigten Zeitaufwand und ermöglicht die Fokussierung auf die Parametrierung. Zusätzlich sollte eine aufgabenangemessene Bedienerführung eine fehlerfreie Programmerstellung unterstützen, indem sie den Bediener genau zu den relevanten Eingaben leitet und unnötige Passagen vermeidet.

Den Ausgangspunkt des in dieser Arbeit vorgestellten Bedienkonzepts bildet ein grafischer Ablaufplan, der aus verständlichen Handlungen, wie *Bauteil greifen* oder *Signal abwarten*, zusammengesetzten wird. Die Darstellung bietet eine intuitiv verständliche Repräsentation der bekannten Applikation und erlaubt so eine einfache Orientierung. Die eingefügten Aktionen, im Folgenden auch Arbeitsschritte genannt, werden abhängig von ihrer Komplexität über bis zu vier in einander übergehende Schritte parametriert. Die Abbildung 4.2 zeigt diese für die umfangreichsten Aktionen, *Bauteil greifen* und *Bauteil ablegen*. Die Reihenfolge der Parametrierung ergibt sich aus den Abhängigkeiten der Daten voneinander. Zunächst wählt der Bediener den benötigten Aktionstyp aus, der sich entsprechend des manuellen Montageablaufs ergibt. Durch den Arbeitsschritt sind die erforderlichen Parameter bekannt und der Bediener kann bei der Eingabe geeignet unterstützt werden. Bevor die Manipulation durch Teach-In festgelegt wird, ist die Lageerkennung einzurichten, um die Greiferbewegungen relativ zur jeweiligen Bauteillage zu speichern. Für einen konsistenten Ablauf werden Ablegeaktionen in derselben Reihenfolge definiert. In einem 4. Schritt sind die auszuführenden Behandlungsmöglichkeiten bei Störungen der Lokalisierung und der Manipulation definierbar. Dieser Schritt erfolgt zuletzt, da zum einen bzgl. der Lageerkennung auf zuvor eingestellte Parameter zurückgegriffen wird und das Einrichten der Störungsbehandlung zum anderen auch zur Laufzeit erfolgen kann.

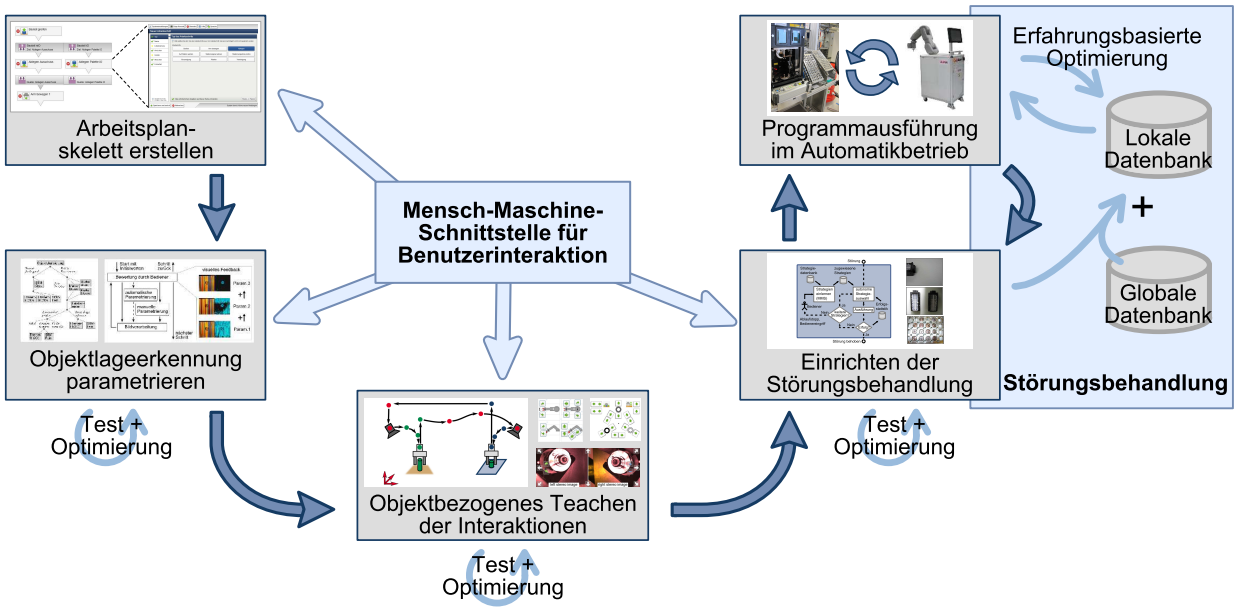


Abb. 4.2.: Ablauf zur Programmierung von komplexen Aktionen in vier Schritten

Der grafische Ablaufplan wird mittels iconbasierter Programmierung erstellt (siehe Abschnitt 4.3.1). Ausgehend von dieser Arbeitsplandarstellung erfolgt die Parametrierung aller enthaltenen Aktionen vollständig wizardbasiert, um den Bediener jederzeit führen und unterstützen zu können (siehe Abschnitt 4.3.2).

4.3.1. Ablaufplanerstellung durch iconbasierte Programmierung

Das zentrale Element der Benutzerführung bildet eine grafische Darstellung des Arbeitsablaufs, der mittels iconbasierter Programmierung aus einzelnen Aktionen aufgebaut wird. Alle zur Montage erforderlichen Teilhandlungen werden hierzu durch verständliche Arbeitsschritte abgebildet (siehe auch [Dose und Dillmann 2012a]). Die Tabelle 4.2 zeigt diese im Überblick. Die Handlungen sind großteils menschlichen Tätigkeiten nachempfunden, so dass ihre Funktion bereits aus dem Namen ersichtlich ist. Die letzten beiden Aktionen stellen nur Steuerungsfunktionen dar und werden nicht manuell eingefügt.

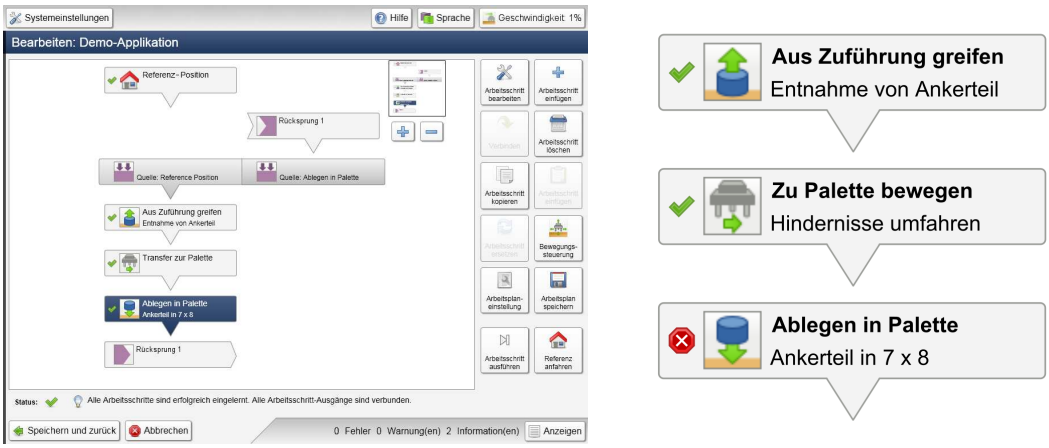
Die Abbildung 4.3a zeigt die Bedienoberfläche zur iconbasierten Programmierung. Im Feld auf der linken Seite wird der erstellte Ablauf größenskalierbar und mit Übersichtskarte angezeigt. Die Darstellung beginnt stets mit einer Aktion für die Referenzierung des Systems zum Arbeitsplatz. Diese ist zuerst zu parametrieren und auszuführen, damit im Anschluss alle Roboterposen relativ zu dieser Referenz oder anderen referenzierten Objekten gespeichert werden können. Dies ermöglicht bei einer Verschiebung des Robotersystems eine Neuberechnung alle veränderten Positionen relativ zum Arbeitsplatz. Die rechte Seite der Abbildung 4.3a zeigt die Funktionsleiste mit Schaltflächen, z. B. zum Hinzufügen, Bearbeiten, Löschen oder Kopieren von Aktionen. Eine Statusanzeige am unteren Rand informiert, ob der Arbeitsplan ausführbar ist oder ob noch unvollständige Arbeitsschritte oder offene Enden vorhanden sind.

Aktion	Beschreibung	Symbol
Referenzieren	Lokalisierung des Roboters am Arbeitsplatz	
Bauteil greifen	Bauteil aufnehmen, ggf. inklusive Lokalisierung	
Bauteil ablegen	Bauteil ablegen, ggf. inklusive Lokalisierung	
Arm bewegen	Pfad abfahren zur Kollisionsvermeidung	
Signal geben	Digitalen Ausgang setzen	
Signal abwarten	Warten bis digitaler Eingang gesetzt	
Warten	Definierte Totzeit abwarten	
Entscheiden	Bedingte Auswahl einer Folgeaktion	
Vereinigung	Aktionsstränge zusammenführen	
Rücksprung	Verbindung zwischen entfernten Aktionen	

Tab. 4.2.: Übersicht der zur Kleinserienmontage erforderlichen Arbeitsschritte

Die erstellten Arbeitsschritte werden jeweils durch ein Symbol repräsentiert, das neben einem frei wählbaren Namen und einer Beschreibung auch ein aktionsspezifisches Icon sowie den aktuellen Status des Arbeitsschritts anzeigt (siehe Abbildung 4.3b). Sind alle Parameter angegeben, besitzt die Aktion einen grünen Haken und ist über eine Schaltfläche ausführbar. Wenn alle Arbeitsschritte vollständig sind, ist der Arbeitsplan als Ganzes ausführbar.

Der grafische Aufbau des Programmablaufs aus verständlichen Arbeitsschritten ermöglicht eine intuitive Repräsentation der bekannten Arbeitsaufgabe und eine einfache Orientierung innerhalb des Arbeitsplans.



(a) **(b)**

Abb. 4.3.: Iconbasierte Programmierung des Programmablaufplans: a) Bedienoberfläche, b) Detailansicht der iconbasierten Verkettung mehrerer Arbeitsschritte

4.3.2. Wizardbasierte Aktionsparametrierung

Für die Parametrierung der Arbeitsschritte wird ein Wizard eingesetzt, so dass der Bediener einer vorgegebenen Sequenz an Schritten folgen kann. Dieses Konzept wird in vielen alltäglichen Programmen eingesetzt und sollte der Zielgruppe anhand ihrer Computerkenntnisse bekannt sein (siehe Abschnitt 2.2.2). Um den Bediener zusätzlich zu unterstützen und die Orientierung durch einen konsistenten Aufbau zu vereinfachen, wurde ein einheitliches Template für alle in der MMS eingesetzten Wizards erstellt, das in Abbildung 4.4 dargestellt ist. Die markierten Elemente und ihre Funktion sind wie folgt:

- Die **Titelzeilen** (1) zeigen neben der Wizardaufgabe und dem Namen der Aktion auch einen längeren Titel des aktuellen Schritts.
- Das **Menü** (2) zeigt alle Schritte der aktuellen Hierarchie. Die chronologische Auflistung dient zur Übersicht und Navigation. Der jeweilige Status signalisiert, ob Eingaben erforderlich sind, und vereinfacht so den Wiedereinstieg nach Unterbrechungen der Parametrierung.
- Der **Anweisungstext** (3) beschreibt die Aufgabe des Bedieners in diesem Schritt und weist auf Besonderheiten hin.
- Ein **Symbol** (4) zeigt den Inhalt des aktuellen Schritts zur schnellen Erfassung visuell an.
- Der **Funktionsbereich** (5) enthält in der Regel die zur Parametrierung erforderlichen Steuerelemente. In Abbildung 4.4 ist die Informationsseite gezeigt, die einen kurzen Überblick über die nachfolgenden Schritte und deren Symbole gibt.
- Über die **Steuertasten** (6) wird der nächste Schritt aufgerufen oder der Wizard verlassen.
- Die **Statusleiste** (7) zeigt wichtige Nachrichten wie Warnungen oder Störungen an.

Abb. 4.4.: Aufbau des einheitlichen Wizardtemplates

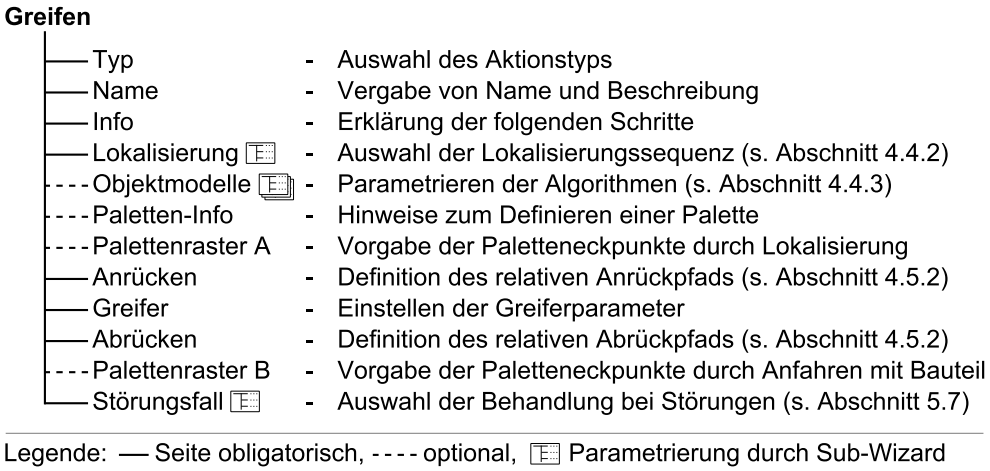


Abb. 4.5.: Sequenz der Wizardseiten für die Parametrierung einer Aktion zum Lokalisieren und Greifen von Bauteilen aus einer Palette

- Die **Steuerleiste** (8) erlaubt z. B. den Zugriff auf die Hilfe oder das Ändern der Sprache.

Zur Vereinfachung der Orientierung innerhalb des Wizards sind alle benötigten Parametrierschritte inhaltlich hierarchisch sortiert. Um den aktuellen Kontext übersichtlich darzustellen, zeigt das Menü (2) nur die aktuelle Ebene an. Tiefer liegende Bereiche sind über Sub-Wizards erreichbar. Die Abbildung 4.5 zeigt die möglichen Schritte der obersten Hierarchie zur Parametrierung einer Greifaktion. Die Seiten zur Parametrierung der Objektmodelle und Paletten sind optional. Sie werden dynamisch anhand der Benutzereingaben im Schritt *Lokalisierung* eingefügt, wenn die Aktion bewegliche Bauteile bzw. eine Palette umfasst. Von den beiden Seiten *Palettenraster A* und *B* wird jeweils nur eine angezeigt, abhängig von der Notwendigkeit die Bauteillage innerhalb der Paletten erkennen zu müssen oder blind greifen zu können. Die Parametrierung der Schritte *Lokalisierung*, *Objektmodelle* sowie *Störungsfälle* erfolgt aufgrund des hohen Umfangs über Sub-Wizards. Die in Abbildung 4.5 dargestellte Schrittsequenz gilt für die komplexe Greifaktion bzw. entsprechend für das Ablegen. Alle übrigen Aktionen umfassen weniger Schritte.

4.4. Methode zum intuitiven Einrichten einer Objektlageerkennung

Wie in Abschnitt 2.1.2 beschrieben, erfolgt die Bauteilzuführung zum Prozess in der Kleinserienmontage sehr vielfältig. Einerseits können Objekte in einem großen Bereich < 25cm x 25cm zugeführt werden, andererseits ist häufig eine hohe Genauigkeit von < 0,5 mm für eine fehlerfreie Manipulation erforderlich. Um beide Anforderungen abzudecken und gleichzeitig eine kurze Rechenzeit und den Einsatz günstiger Komponenten zu ermöglichen, sind z. T. mehrere Sensorsysteme mit verschiedenen Sichtbereichen und Genauigkeiten seriell zu kombinieren. Gleichzeitig sind verschiedene Lageerkennungsalgorithmen erforderlich, um das große Spektrum der Bauteile

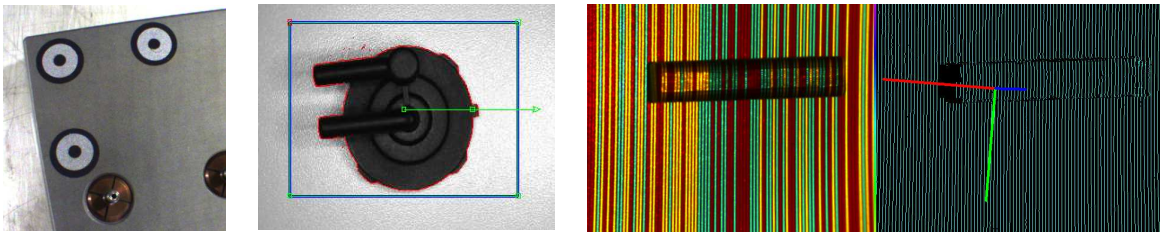
mit ihren vielfältigen Oberflächeneigenschaften abzudecken. Durch die Kombinations- und Verkettungsmöglichkeiten verschiedener Sensorsysteme und Algorithmen ist die Auswahl einer geeigneten Sequenz sowie die Parametrierung der enthaltenen Lokalisierungsschritte für den Benutzer sehr komplex. Für eine schnelle Inbetriebnahme der Objektlageerkennung ohne Expertenwissen in Bildverarbeitung ist der Prozess durch eine geeignete MMS entsprechend zu unterstützen und zu vereinfachen. Der Abschnitt 4.4.1 erläutert zunächst die im Konzept exemplarisch betrachteten Lokalisierungsverfahren, bevor die Auswahl einer geeigneten Lokalisierungssequenz sowie die Parametrierung der Algorithmen in den Abschnitten 4.4.2 und 4.4.3 beschrieben wird (siehe auch [Dose und Dillmann 2012b]).

4.4.1. Exemplarisch betrachtete Lokalisierungsverfahren

Das in dieser Arbeit entwickelte Konzept zur schnellen Inbetriebnahme ohne Expertenwissen wird exemplarisch an drei unterschiedliche Lageerkennungsverfahren in Kombination mit zwei verschiedenen Kamerasystemen erläutert. Die vorgestellte Methode lässt sich bei ausreichendem Zugriff auf den Quellcode der Lokalisierungsalgorithmen auch auf andere Verfahren übertragen. Die betrachtete Sensorik besteht zum einen aus einer Monokamera, die über ein Objektiv mit kurzer Brennweite ein großes Sichtfeld mit eingeschränkter Genauigkeit abdeckt. Die für eine störungsfreie Interaktion erforderliche Feinlageerkennung erfolgt anschließend über eine Stereokamera mit hoher Genauigkeit aber kleinem Blickfeld. Lässt die Applikation nur geringe Schwankungen der Objektlage zu, kann die Feinlageerkennung auch einzeln ausgeführt werden. Bei den eingesetzten Algorithmen werden mit einer landmarken-, einer kanten- und einer oberflächenbasierten Lageerkennung drei sehr unterschiedliche Verfahren exemplarisch betrachtet. Alle drei Methoden sind ansichtsbasiert und werden anhand einer einzigen Objektansicht parametriert und eingerichtet:

Die **3D-landmarkenbasierte Lokalisierung** verwendet in der Regel künstliche Merkmale, die sich stark von ihrer Umgebung abheben und robust zu detektieren sind. Im vorliegenden Fall dienen zur Wiedererkennung jeweils drei identische Marken, die ungefähr im rechten Winkel auf das zu lokalisierende Objekt aufgeklebt werden (siehe Abbildung 4.6a). Bei der Segmentierung der Marken im Kamerabild wird ein Kanten- mit einem Kreisdetektor kombiniert, um Kreise mit einem entsprechendem Hell-Dunkel-Wechsel und passendem Verhältnis der Umfänge zu detektieren. Für eine hohe Robustheit des Verfahrens auch bei starkem Umgebungslicht werden eine aktive Beleuchtung der Bildszene und reflektierende Marker eingesetzt. Bei der Bestimmung ihrer Lage im Raum werden die Markermittelpunkte über die kalibrierten Stereokameras trianguliert oder mittels eines Ellipsen-Fits und der Kalibrierung der Monokamera berechnet.

Das Verfahren eignet sich zur dreidimensionalen Lokalisierung von Objekten, die direkt oder über einen fixierten Index referenziert werden können, wie Arbeitsplätzen, Prozessnestern, Paletten oder Kisten.



(a) (b) (c)

Abb. 4.6.: Die exemplarisch betrachteten Lokalisierungsverfahren: a) landmarkenbasiert, b) konturbasiert, c) oberflächenbasiert mit projiziertem Muster zur Korrespondenzpunktzuordnung (links: linkes Stereokamerabild, rechts: farbkodiertes Abstandsbild)

Bei der **2D-kantenbasierten Lokalisierung** werden Objekte z. B. anhand charakteristischer Helligkeitskanten im Grauwertbild detektiert (siehe Abbildung 4.6b). Die zur Erkennung relevanten Kanten oder Texturen sind zuvor manuell auszuwählen. Verschiedene Bildverarbeitungsbibliotheken, wie z. B. Halcon der Firma MVTec [MVTec 2013], bieten anschauliche Bedienelemente zum Selektieren der Kanten. Im vorliegenden Fall wird ein 2D-Verfahren betrachtet, bei dem zur Lageberechnung vorausgesetzt wird, dass die Position und Orientierung der Objekte nur innerhalb einer zur Kamerablickrichtung senkrechten Ebene variieren.

Das Verfahren eignet sich damit für die zweidimensionale Lageerkennung von Bauteilen, Prozessnestern oder Paletten, deren Konturen oder Texturen charakteristische und kontrastreiche Kanten im Grauwertbild erzeugen und die z. B. ohne zu verkippen auf einem Tisch liegen. Die kantenbasierte Lokalisierung kann sowohl mit der Mono- als auch mit nur einem Sensor der Stereokamera kombiniert werden.

Die betrachtete **3D-oberflächenbasierte Lageerkennung** verwendet dreidimensionale Tiefendaten, in denen Objekte über ihre Kontur erkannt und anhand ebener Oberflächen lokalisiert werden. Für die dreidimensionale Rekonstruktion der Bildszene wird diese über einen Projektor mit einem farbkodierten Streifenmuster beleuchtet, um die Korrespondenzpunktzuordnung zu vereinfachen (siehe Abbildung 4.6c). Die Berechnung der Tiefenwerte erfolgt über eine Kombination von Triangulation und kalibriertem Projektor. Anhand der Kontur wird der Objektbereich zunächst in einem grauwertkodierten Abstandsbild segmentiert. Die Berechnung der Objektlage erfolgt über einen Ebenen-Fit auf planaren Bereichen des Objekts. Alternativ wären andere oberflächenbasierte Matching-Algorithmen einsetzbar, wie z. B. ein ICP-Verfahren.

Diese Methode eignet sich zur dreidimensionalen Lokalisierung von Bauteilen mit ebenen, nicht zu stark texturierten Oberflächen. Die Triangulation der Raumpunkte erfordert die Verwendung einer Stereokamera.

4.4.2. Bestimmung einer geeigneten Lokalisierungssequenz

Durch die verschiedenen Algorithmen und Sensorsysteme, die nicht nur untereinander kombiniert sondern auch seriell verkettet werden können, entsteht eine Vielzahl an Auswahlmöglichkeiten.

Beim Einrichten einer Objektlokalisierung ist daher zuerst das oder die geeignetsten Verfahren für eine robuste Lageerkennung zu wählen. Dabei muss die Lokalisierung für eine störungsfreie Automatisierung möglichst bei allen zulässigen Umgebungsbedingungen funktionieren und soll gleichzeitig so wenig Schritte wie möglich umfassen, um die Taktzeit und die Inbetriebnahmedauer nicht unnötig zu erhöhen. Da unerfahrene Bediener die Eigenheiten der Verfahren nicht kennen, ist die Entscheidung oft sehr komplex.

Um die Wahl einer geeigneten Sequenz von Lokalisierungsschritten zu vereinfachen, wurde im Rahmen dieser Arbeit analysiert, welche geeigneten Kombinationsmöglichkeiten bei den Applikationen in der Kleinserienmontage auftreten. Dabei sind verschiedene Grundsätze erkennbar, nach denen z. B. das Greifen und Ablegen von Bauteilen nur nach einer Feinlageerkennung erfolgen sollte. Weiterhin ist eine landmarkenbasierte Lokalisierung den anderen Verfahren wegen der höheren Robustheit nach Möglichkeit vorzuziehen. Eine Wiederholung derselben Kombination aus Algorithmus und Kamera innerhalb einer Sequenz bringt keinen Vorteil. Mit der Ausführung einer 3D-oberflächenbasierte Lageerkennung kann die Kamera zur Erkennung kleiner Merkmale durch eine 2D-kantenbasierte Lokalisierung relativ zu verkippten Werkstücken ausgerichtet werden. Die umgekehrte Reihenfolge bringt keinen Informationsgewinn.

Als Ergebnis der Analyse wurden 22 Kombinationen identifiziert, die in Abbildung 4.7 dargestellt sind. Jeder von oben nach unten verlaufende Pfad stellt eine sinnvolle Lokalisierungssequenz dar. Die *Anwesenheitsprüfung* umfasst im Gegensatz zur *Bauteillageerkennung* nur eine Objekterkennung und keine Lokalisierung. Die zu findenden Merkmale können vom Bauteil oder vom leeren Prozessnest stammen, so dass sowohl eine erfolgreiche als auch eine fehlgeschlagene Detektion die Anwesenheit bewerten können.

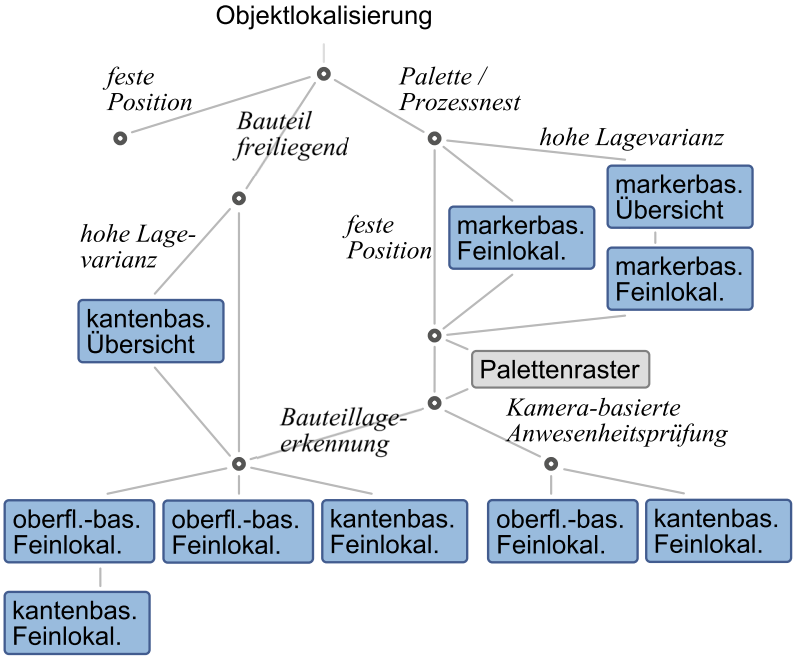


Abb. 4.7.: Geeignete Verknüpfungen der Lokalisierungsverfahren. Jeder Pfad stellt eine sinnvolle Kombination dar.

Bei der Konzeption der intuitiven MMS wurde der vom Bediener auswählbare Kombinationsumfang der Lageerkennungsverfahren auf die identifizierten Möglichkeiten reduziert, da nur in seltenen Spezialfällen andere Lokalisierungssequenzen sinnvoll sind. Basierend auf den sich ergebenden Auswahlmöglichkeiten wurde der in Abbildung 4.8 dargestellte Entscheidungsbaum entworfen und als Fragenkatalog in einer Wizardstruktur abgebildet. Die Auswahlmöglichkeiten werden jeweils in Form von Bildern und Texten verdeutlicht. Der Ablauf der einzelnen Seiten wird dynamisch anhand der vorherigen Auswahl erstellt. Nach Beantwortung von bis zu fünf verständlichen Fragen zur Applikation wird dem Bediener eine Liste mit den erforderlichen Lokalisierungsschritten angezeigt. Die abgefragten Randbedingungen der Applikation beschreiben z. B., ob die Zuführung der Werkstücke frei auf einer Fläche, einzeln geführt in einem Prozessnest oder in Gruppen mittels einer Palette erfolgt. Zusätzlich wird abgefragt, ob die Bauteile sowie potentielle Prozessnester und Paletten beweglich sind und wie groß der Bewegungsradius ist. Um dies zu beurteilen, kann die Zuführfläche anhand der angezeigten Kamerabilder mit deren Sichtfeldern abgeglichen werden. Die häufig letzte Frage betrifft die Entscheidung zwischen den zur Bauteillageerkennung eingesetzte kanten- oder oberflächenbasierten Verfahren. Ist sich der Bediener bei der Auswahl nicht sicher, kann er sich durch einen Zusatzwizard mit weiteren Fragen zum Werkstück einen Algorithmus vorschlagen lassen. Die hierzu abgefragten Randbedingungen betreffen die maximale Verkippung des Bauteils zu Senkrechten sowie verschiedene Eigenschaften, die eine Eignung für die kanten- bzw. die oberflächenbasierte Lokalisierung beschreiben. Insbesondere sind für eine erste Bewer-

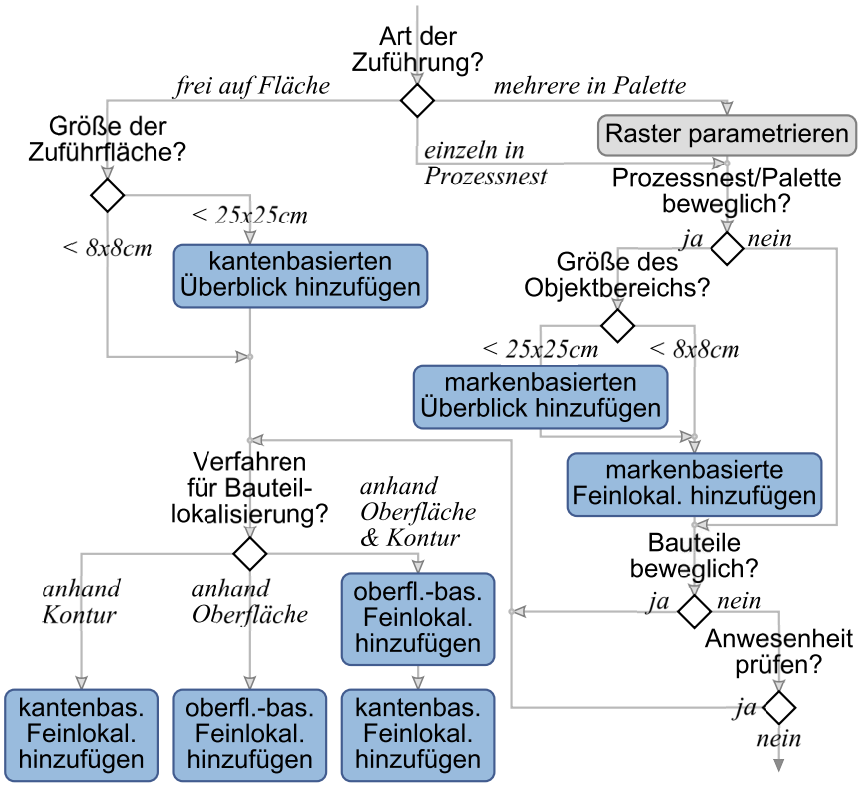


Abb. 4.8.: Anhand des Entscheidungsbaums werden die zu beantwortenden Fragen ausgewählt und die Liste mit erforderlichen Lokalisierungsschritten erstellt

tung der Verfahren auch Zusatzfunktionen integriert, mit denen die Bauteilmerkmale ohne gezielte Parametrierung im Kamerabild angezeigt werden können. Zuletzt wird anhand der Antworten ein Vorschlag generiert. Sind beide Verfahren möglich, wird zur häufig robusteren kantenbasierten Lokalisierung geraten. Neben dem Vorschlag werden dem Bediener auch Hinweise basierend auf seinen Antworten gegeben, worauf bei der späteren Auswahl der relevanten Objektmerkmale im Bezug auf eine mögliche Bauteilsymmetrie zu achten ist.

Sollten keiner der beiden Algorithmen die gewählten Bauteileigenschaften abdecken, wird der Bediener darauf hingewiesen, dass die Applikation nur mit einer Anpassung der Zuführung robust automatisierbar ist. Nach der Beantwortung aller Fragen des Hauptwizards, werden die vom Bediener zu parametrierenden Objektlageerkennungen aufgelistet.

4.4.3. Parametrierung von Objektlageerkennungen

Wenn die erforderlichen Lokalisierungsschritte bekannt sind, müssen die entsprechenden Verfahren eingerichtet werden. Hierzu sind die charakteristischen Objektmerkmale festzulegen und verschiedene Teilschritte zu konfigurieren, wie z. B. die Bildaufnahme, eine mögliche Vorverarbeitung oder die Extraktion und Zuordnung der Merkmale. Da der Bediener die Bedeutung der einzelnen Parameter in der Regel nicht genau kennt, ist eine Unterstützung durch die MMS erforderlich.
Um die Inbetriebnahme der Verfahren zu vereinfachen, wurde zunächst untersucht, wie die Auswirkung der einzelnen Parameter auf die Objektlageerkennung intuitiv verständlich dargestellt werden kann. Dabei wurden verschiedene visuelle Zwischenergebnisse der Bildverarbeitungskette identifiziert, die den Einfluss von jeweils kleinen Gruppen inhaltlich zusammenhängender Parameter verdeutlichen. Die Inbetriebnahme wurde entsprechend in mehrere Schritte unterteilt, wodurch sich die Bedienung direkt in das Wizardkonzept einfügt. Auf Basis aktueller Kameraaufnahmen des Objekts werden für die Visualisierungen verschiedene Zwischenergebnisse der Bildverarbeitungskette mit relevanten Informationen aufbereitet. Der Bediener kann die einzelnen Parameter über Schaltflächen variieren und anhand der Darstellungen intuitiv bewerten, ob sich das visuelle Ergebnis verbessert oder verschlechtert. Er kann die Parameter intuitiv optimieren, ohne ihre genaue Bedeutung innerhalb der Bildverarbeitungskette zu kennen. Die Abbildung 4.9 verdeutlicht diesen Prozess anhand eines Parameters zur Segmentierung eines Bauteils vom Untergrund. Der Wert wird so lange inkrementiert, bis im Abstandsbild nur noch das Objekt ohne Hintergrund erkennbar ist.

Für einige Schritte stehen zusätzlich automatische Parametrierfunktionen zur Verfügung, die bei den meisten Randbedingungen eine gute Einstellung einzelner Parameter oder ganzer Gruppen erreichen. Potentielle Fehleinstellungen werden vom Bediener erkannt und durch Wiederholung unter leicht veränderten Bedingungen oder durch manuelle Anpassung behoben. Als weitere Unterstützung sind die Parameter jeder Gruppe zusätzlich nach der Notwendigkeit einer Anpassung in die Kategorien *immer anzupassen*, *selten anzupassen* und *Expertenmodus* unterteilt, so dass der Bediener sich in der Regel auf die relevanten Parameter fokussieren kann. Wird in einem Schritt keine geeignete Einstellung erreicht, sind auch die seltenen Werte zu prüfen oder vorherige Schritte

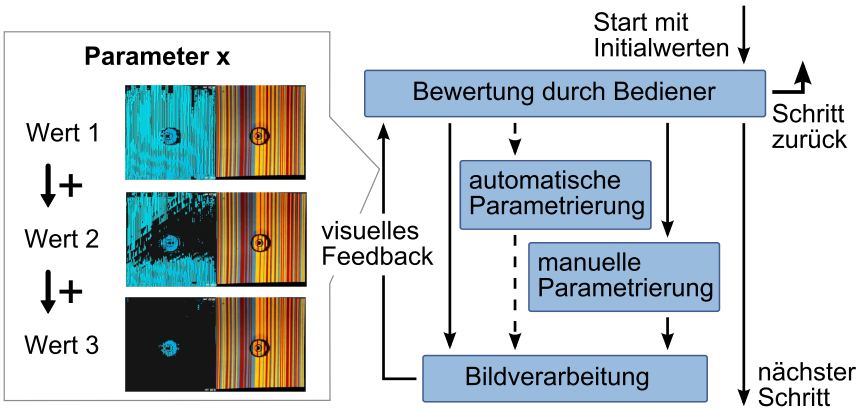


Abb. 4.9.: Optimierung unbekannter Parameter durch gezieltes Variieren und intuitives Bewerten anhand visuellen Feedbacks

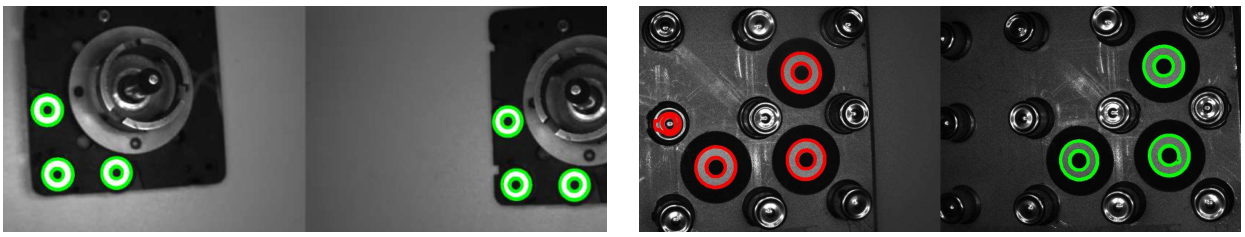
mit dem zusätzlichen Wissen über die aktuellen Schwierigkeiten zu wiederholen. Sind alle Schritte abgeschlossen, kann der Bediener die Robustheit der Lokalisierung bei variierenden Objektpositionen manuell testen. Optional kann die Lageerkennung automatisch mit mehreren, durch den Roboter veränderten Kamerapositionen überprüft werden. Wird die Lage von Objekt und Kamera zwischen mehreren Detektionen nicht verändert, zeichnet sich eine robuste Lokalisierung durch geringe, innerhalb der Toleranz liegende Roboterbewegungen beim Lageausgleich aus.

Die beschriebene Methode wird für alle drei Lokalisierungsverfahren angewendet und lässt sich auf andere Algorithmen übertragen, sofern intuitiv verständliche Darstellungen erstellbar und alle hierzu erforderlichen Parameter zugreifbar sind. Im Folgenden wird die Vorgehensweise für die exemplarisch betrachteten Verfahren erläutert.

3D-landmarkenbasierte Lageerkennung

Das Einrichten der landmarkenbasierten Lokalisierung erfordert nur wenige Schritte und funktioniert aufgrund der definierten Merkmale und ihrer reflektierenden Oberfläche sehr robust. Einzustellen sind die Bildaufnahme, die Beleuchtungsdauer sowie verschiedene Parameter, die bei der Landmarkensegmentierung zur Unterscheidung von anderen Kreisen im Bild dienen.

Nach dem Anbringen der Marken und dem Ausrichten der Kamera steht zum Einstellen der Bildaufnahme eine automatische Funktion zur Verfügung, die den Verstärkungsfaktor und die Belichtungszeit bei aktiver Beleuchtung in kleinen Schritten iteriert. Ausgewählt wird die Kombination, bei der die Grauwerte der hellen Flächen aller gefundenen Landmarken mit einem mittleren Zielwert übereinstimmen. Hierbei können Fehlsegmentierungen der Marken auftreten, da die Toleranzbereiche bei der Parametrierung sehr weit sind. Zur Bewertung und manuellen Einstellung werden die erkannten Merkmale in die Livebilder eingeblendet (siehe Abbildung 4.10). Bei einer korrekten Detektion wird die relative Lage der Landmarken zur Kamera im nächsten Schritt als Referenzposition gespeichert. Gleichzeitig werden die aktuellen Abmessungen und Längenverhältnisse der Marker sowie ihre Abstände zueinander erfasst, um die Markersegmentierung im Betrieb überprüfen zu können. Weitere Parameter, wie die Toleranzbereiche zur Merkmalssegmentierung,



(a) **(b)**

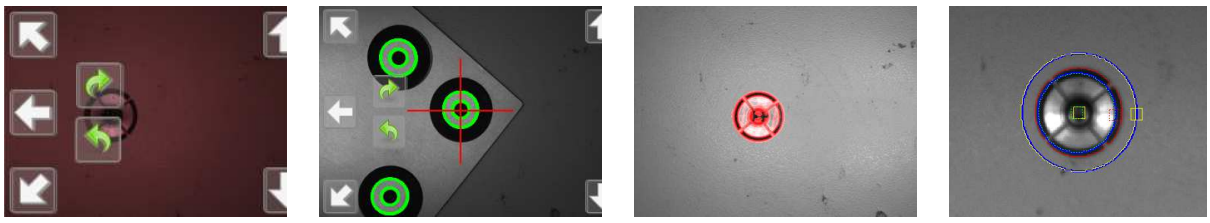
Abb. 4.10.: Darstellung der segmentierten Landmarken. a) Grüne Anzeige bei korrekter Markenanzahl, b) Rote Visualisierung bei abweichender Markenanzahl

sind nur anzupassen, wenn Tests mit verschiedenen Objektpositionen eine geringe Robustheit der Lokalisierung signalisieren.

2D-kantenbasierte Lageerkennung

Beim Einrichten der kantenbasierten Lokalisierung liegt eine wichtige Anforderung in der korrekten Positionierung der Stereokamera. Da dieses Verfahren nur zweidimensionale Lageabweichungen detektiert, muss nach der Zentrierung des Bildfelds eine der beiden Kameras mit der Blickachse senkrecht zur Bewegungsebene des Bauteils ausgerichtet werden. Ungenauigkeiten hierbei können später zu Störungen bei der Objektmanipulation führen. Zur Unterstützung beim Ausrichten des Sensors kann der Bediener eine Platte mit drei aufgeklebten Landmarken nutzen, um die Bewegungsebene des Objekts sensorisch zu erfassen. Die Platte wird hierzu auf den Untergrund gelegt und ihre Lage durch das von den Markierungen aufgespannte Koordinatensystem mit der Stereokamera detektiert, wie Abbildung 4.11b zeigt. Die hierzu nötige Lokalisierung erfolgt nur einmalig und erfordert daher keine robuste Parametrierung. Bei erkannter Lage des Koordinatensystems wird eine Kamera automatisch senkrecht zur Platte positioniert, wobei das Blickfeld und der Abstand nahezu unverändert bleiben. Die entsprechende Roboterpose wird anhand der Transformation zwischen Kamera und Tool-Center-Point (TCP) über die externe Kamerakalibrierung berechnet.

Nach dem Ausrichten des Sensors werden die Belichtungszeit, der Verstärkungsfaktor sowie die Beleuchtungsdauer so eingestellt, dass charakteristische Kanten im angezeigten Kamerabild deutlich erkennbar sind (siehe Abbildung 4.11c). Die Auswahl der Merkmale erfolgt über ein Tool der Bildverarbeitungsbibliothek Halcon der Firma MVTec durch Selektieren und Ausschließen einzel-



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

Abb. 4.11.: Parametrierung eines Objektmodells für die konturbasierte Lokalisierung. a) Ausrichten auf Bauteil, b) Kamera mittels Landmarkenlokalisierung senkrecht stellen, c) Bildaufnahme parametrieren, d) Objektmerkmale auswählen

ner Kantenbereiche mittels geometrischer Formen (siehe Abbildung 4.11d). Hierbei sind die im Rahmen der Auswahl der Lokalisierungsverfahren erzeugten Hinweise zur Wahl der Objektmerkmale zu beachten. Durch abschließende Tests mit variablen Bauteilpositionen kann die Lokalisierung überprüft werden.

3D-oberflächenbasierte Lageerkennung

Auch das Einrichten der oberflächenbasierten Lokalisierung wurde durch intuitiv verständliche Visualisierungen und Unterstützungsfunktionen vereinfacht. Nach dem Einstellen des Kameraabstands für ein scharfes Streifenmuster kann zunächst der zum Bauteil gehörende Bildbereich auf zwei Arten segmentiert werden. Liegt das Werkstück auf einem flachen Untergrund, kann dieser bei zuvor entferntem Bauteil dreidimensional erfasst und durch eine Ebene angenähert werden. Anschließend kann der Bediener die Ebene parallel verschieben, um den Hintergrund vom zurückgelegten Werkstück zu trennen. Es wird dabei angenommen, dass nur die vor der Ebene liegenden Raumpunkte zum Objekt gehören und nachfolgend auszuwerten sind. Der Prozess ist in Abbildung 4.12 veranschaulicht. Bei der zweiten Option wird auf ähnliche Weise nur ein relevanter Abschnitt des Objekts segmentiert. Die anhand ebener Flächen des Hintergrunds oder des Objekts erstellte Ebene definiert hierzu die Mitte eines Schlauchs, der die zu betrachtenden Raumpunkte einschließt. Die parallele Verschiebung sowie die Schlauchdicke werden manuell eingestellt. Nach der Segmentierung des Objektbereichs wird die Bildaufnahme so parametriert, dass die angezeigte Punktewolke das Objekt möglichst vollständig und fehlerfrei darstellt. Hierfür kann eine automatische Parametrierfunktion eingesetzt werden, die den Verstärkungsfaktor und die Belichtungszeit iteriert und die beste Kombination anhand der Menge der rekonstruierten Oberflächenpunkte ermittelt. Im Anschluss sind drei Schwellwerte zur Segmentierung der Farbstreifen im Hue-Saturation-Value-

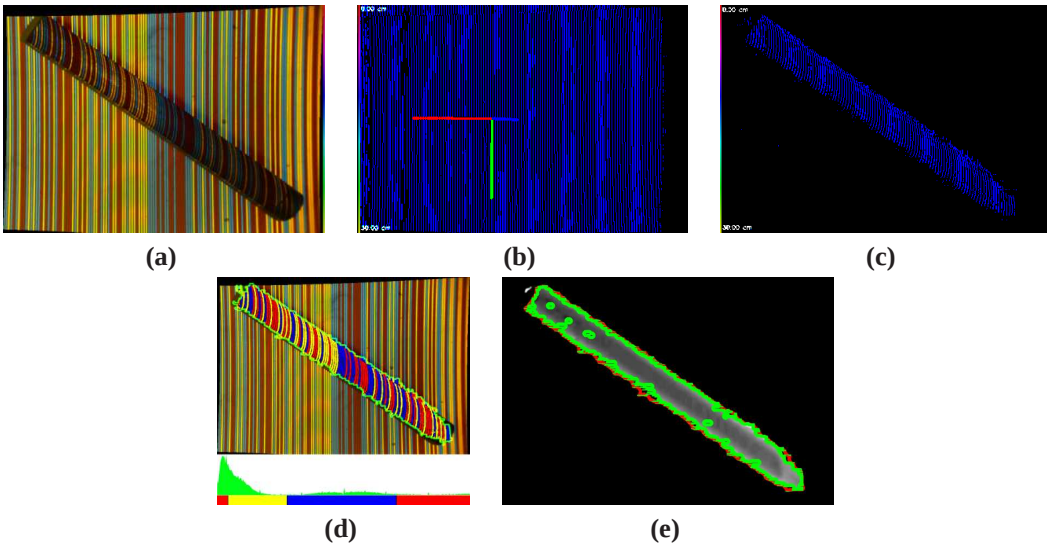


Abb. 4.12.: Parametrierung der oberflächenbasierten Lokalisierung. a) Einstellen des Kameraabstands für eine scharfe Streifenprojektion, b) Erfassen der Hintergrundebene ohne Bauteil, c) Segmentierung des Objektbereichs, d) Parametrierung der Farbstreifenzuordnung mittels Histogramm, e) Segmentierung der Bauteilkontur

Farbraum einzustellen. Hierzu wird neben dem Kamerabild mit dargestellter Farbzuordnung auch ein Histogramm über die auf dem Objekt verlaufenden Streifenanteile angezeigt. Die Schwellwerte sind vom Bediener in der Regel entsprechend der Minima einzustellen. Durch Approximation der Kurve können die Parameter auch automatisch berechnet werden. Zuletzt ist die relevante Kontur des Werkstücks halbautomatisch bei angezeigtem Ergebnis durch Dilatation und Erosion der Objektbereiche in der Tiefenkarte über einstellbare Schwellwerte festzulegen. Optional könnte auch das bei der kantenbasierten Lageerkennung beschriebene Segmentierungstool verwendet werden.

4.5. Methode zur effizienten Positionierung eines Roboterarms

Das Definieren von Roboterbewegungen macht einen großen Teil der Programmierung von Montageaufgaben aus. Als wichtigste Anforderung ist wegen der geringen Toleranzen bei den Fügeprozessen eine hohe Genauigkeit zu erreichen, gleichzeitig soll der Aufwand für eine schnelle Inbetriebnahme gering sein.

Wie in Abschnitt 3.1.3 dargestellt, ist eine Offline-Positionierung des Roboters in einer Simulationsumgebung wegen der fehlenden und daher aufwändig zu erstellenden Modelle im Rahmen der Kleinserienmontage nicht möglich. Bei der direkten Erfassung menschlicher Bewegungen oder beim Einsatz von lokalisierbaren Zeigewerkzeugen durch unerfahrene Nutzer kann keine für die Bauteilmanipulation ausreichende Genauigkeit vorausgesetzt werden. Die Definition der Bewegungsbahnen soll daher online, direkt mit dem Roboter im Zielarbeitsraum erfolgen. Dabei ist auf teure Hardwarekomponenten ebenso zu verzichten, wie auf die in der industriellen Praxis eingesetzten Bedienhandgeräte, bei denen die verwendet Koordinatensysteme für unerfahrene Nutzer nicht intuitiv verständlich sind (siehe auch [Dose und Dillmann 2012b]).

Um eine geeignete Methode zu entwickeln, die die Anforderungen an eine schnelle und genaue Positionierung erfüllt, werden die im Ablauf erforderlichen Roboterbewegungen in Abschnitt 4.5.1 zunächst auf ihre Eigenheiten analysiert. Die Abschnitte 4.5.2 und 4.5.3 beschreiben die entwickelten Steuerelemente zum Positionieren des Roboters sowie zum Speichern und Bearbeiten der Bewegungstrajektorien.

4.5.1. Analyse der erforderlichen Roboterpositionen

Die in einem für die flexible Montage erstellten Programm enthaltenen Roboterbewegungen können anhand des jeweils verfolgten Zwecks in drei unterschiedliche Kategorien eingeteilt werden. Die Abbildung 4.13 verdeutlicht diese und die Tabelle 4.3 zeigt die jeweiligen Eigenschaften. Die Bahnen bestehen zum einen aus relativ groben Transferbewegungen, mit denen der Roboterarm zur Umfahrung von Hindernissen von einem Interaktionsbereich zum nächsten gesteuert wird. Als Vorbereitung auf die Lageerkennung dient die anschließende Bewegung zum Ausrichten der Kamera

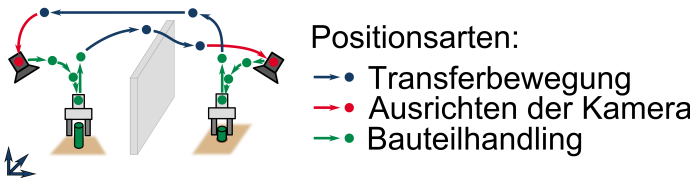


Abb. 4.13.: Positionsarten innerhalb eines Pick-and-Place-Ablaufs

Art der Position	Genauigkeit	Zusatzanforderung	Betrachtete Roboterkomponente
Transfer-bewegungen	< 2 cm	Hindernissen ausweichen	Roboterarm
Ausrichten der Kamera	< 3 mm	Objekt in Tiefenschärfe und Bildfeld zentriert	Kamera
Bauteilhandling	< 0,5 mm	berührungslos, keine Bauteilverschiebung	Greifer

Tab. 4.3.: Kategorien für Roboterbewegungen

relativ zum Objekt. Abhängig von dessen Größe und dem erfassten Bereich ist das Bildfeld mit einer mittleren Genauigkeit zu zentrieren und der Kameraabstand abhängig vom Tiefenschärfebereich einzustellen. Im Anschluss an die Lokalisierung erfolgt das Greifen oder Ablegen. Wegen des häufig geringen Spiels beim Fügen ist die Zentrierung des Greifers relativ zum Bauteil oder dem Ablageplatz mit hoher Genauigkeit durchzuführen. Eine Verschiebung des Bauteils durch Kollisionen muss unbedingt vermieden werden.

Für eine bedienerfreundliche MMS ist ein übersichtliches und leicht nachvollziehbares Konzept für die Definition von Roboterbewegungen einzusetzen. Die zu verwendende Bedienfunktion sollte daher für alle drei Positionsarten gleichermaßen geeignet sein.

4.5.2. Aufgabenangepasste Jog-Steuerung

In der industriellen Praxis wird häufig ein Handbediengerät verwendet, um den Roboterarm relativ zu den Achsen verschiedener Koordinatensysteme zu steuern. Durch die Wahl verschiedener Geschwindigkeiten erlaubt diese Methode sowohl eine schnelle als auch eine genaue Positionierung. Die vorgegebenen Bewegungsrichtungen sind von unerfahrenen Bedienern häufig jedoch nur schwer ersichtlich und haben keinen intuitiven Bezug zum Gesamtsystem oder dessen Einsatzzweck. Bei der Entwicklung einer MMS für ein flexibles Robotersystem ist die eingesetzte Hardware bekannt. Mit diesen Informationen sowie den Anforderungen an die Positionsarten aus Tabelle 4.3 kann das Manövrieren des Roboters deutlich vereinfacht werden. Die Bewegungsrichtungen können gezielt auf den Systemaufbau und die jeweilige Aufgabe angepasst werden, wodurch gleichzeitig eine intuitivere Zuordnung am System ermöglicht wird.

Das in dieser Arbeit entwickelte Konzept zum Ausrichten des Roboterarms basiert auf drei sogenannten Jog-Steuerungen, die jeweils an eine der Positionierungsarten angepasst sind. Zum Verfahren des Roboters wählt der Bediener zuerst eine Schrittweite und führt anschließend eine entsprechende Bewegung über verschiedene Tasten mit verständlich dargestellten Richtungen aus. Da die Einzelschritte mit Inkrementen aus 0.1mm bis 10cm bzw. 0.01° bis 10° frei einstellbar sind, können sowohl schnelle, weite als auch sehr feine Bewegungen durchgeführt werden. Gleichzeitig können Entfernungen direkt gesetzt werden, sofern diese bekannt oder schätzbar sind.

Die einzelnen an die Aufgaben angepassten Jog-Steuerungen werden nachfolgend beschrieben. Die vom Roboter anzufahrenden Zielpositionen sind in der Regel als Lage des Tool-Center-Points (TCP) in Basiskoordinaten anzugeben. Die in den Bedienelementen angezeigten Bewegungsrichtungen beziehen sich jedoch auf die in Tabelle 4.3 angezeigten Roboterkomponenten und müssen daher über ihr Bezugssystem in die neue TCP-Position in Basiskoordinaten überführt werden. Die Bezugssysteme des Greifers, der Kamera sowie der Roboterbasis sind in Abbildung 4.14 dargestellt. Die homogene Transformationsmatrix ${}^{A}_{B}T$ berechnen sich aus der Translation mit den Koordinaten *x*, *y* und *z* sowie der Rotation mit den Winkeln *Rx*, *Ry* und *Rz*, mit denen die Lage von *A* in Koordinaten von *B* beschrieben wird. Details zur Berechnung sind im Anhand unter B aufgeführt.

Die vom Roboter anzufahrende Zielposition ergibt sich über die aktuelle TCP-Position aus ${}^{Base}_{TCP}T$, der Transformation ${}^{TCP}_{Jog,orig}T$ zwischen dem TCP und dem Ursprung des Bezugskoordinatensystems der Jog-Steuerung sowie der Transformationsmatrix der gewählten Relativbewegung ${}^{Jog,orig}_{Relativ}T$. Die Matrix ${}^{TCP}_{Jog,orig}T$ entspricht abhängig von der verwendeten Jog-Steuerung ${}^{TCP}_{Grip}T$, ${}^{TCP}_{Cam}T$ oder ${}^{TCP}_{Base}T$ in Abbildung 4.14. Die zur Berechnung der homogenen Transformationsmatrizen erforderlichen Werte für *x*, *y*, *z*, α, β und γ ergeben sich aus der Lage der jeweiligen Koordinatensysteme zueinander und sind z. B. aus den Konstruktionsplänen des Robotersystems abzulesen. Die Translations- und Rotationswerte für die Matrix der Relativbewegung ${}^{Jog,orig}_{Relativ}T$ resultieren aus der Lage der gewählten Bewegungsrichtungen zum Bezugssystem sowie aus der gewählten Schrittweite. Die neue Zielposition des TCP in Basiskoordinaten errechnet sich schließlich über ${}^{Base}_{TCP,neu}T$ aus Formel 4.1.

$$ {}^{Base}_{TCP,neu}T = {}^{Base}_{TCP,alt}T \cdot {}^{TCP}_{Jog,orig}T \cdot {}^{Jog,orig}_{Relativ}T \cdot {}^{TCP}_{Jog,orig}T^{-}1 \tag{4.1} $$

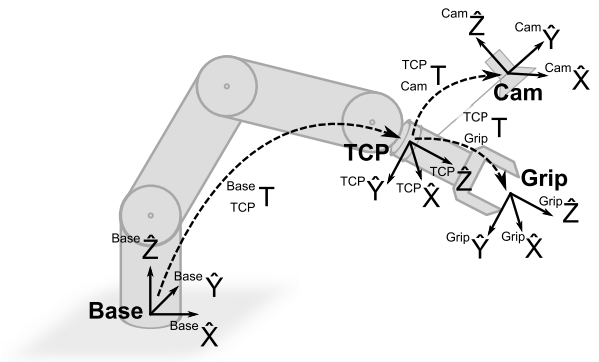


Abb. 4.14.: Lage des Greifer-, Kamera- und TCP-Koordinatensystems

Roboterbasierte Steuerung

Die roboterbasierte Steuerung dient hauptsächlich zur schnellen und kollisionsfreien Vorpositionierung der Kamera bzw. des Greifers, um anschließend die Feinausrichtung mit den nachfolgend beschriebenen Steuerungen durchzuführen. Abhängig von der Applikation müssen bei diesen groben Bewegungen zum Teil starke Orientierungsänderungen des TCP durchgeführt und weit entfernte Positionen angefahren werden. Das in Abbildung 4.15 dargestellte Bedienelement stellt hierzu verschiedene Bewegungsrichtungen zur Auswahl, die anhand von Bildern des Robotersystems veranschaulicht werden. Für eine einfache Zuordnung unabhängig von der Position des Bedieners zum System werden die Darstellungen auf Wunsch horizontal gespiegelt.

Zum Anfahren weit entfernter Ziele, die z. B. auf der anderen Seite des Robotersystems liegen, eignet sich besonders die Drehung um die Roboterbasis (1). Hierbei stehen alle übrigen Gelenke still, um in keine Gelenkwinkelbegrenzung zu fahren. Die weitere Annäherung kann entsprechend der intuitiv verständlichen Darstellungen radial (2) sowie entlang der Z-Achse des Roboters (3) bzw. des Greifers (4) erfolgen. Die Orientierung wird über verschiedene Drehmöglichkeiten (5-7) eingestellt. Ein Problem beim Verfahren des Roboters können die Gelengwinkelbeschränkungen darstellen, an denen der Roboter jeweils nicht weiter bewegt werden kann. Erreichbar sind diese Maximalauslenkungen hauptsächlich an der Handgelenkachse (7), weshalb dort der aktuelle Winkel angezeigt und als Warnung kurz vor der Begrenzung in gelb bzw. rot hervorgehoben wird. Die Visualisierung der Greiferstellung zur Senkrechten ermöglicht zudem das Ausrichten in einem bestimmten Winkel. Zum schnellen Anfahren häufiger Orientierungen kann der Greifer über drei zusätzliche Tasten auf kürzestem Weg auf einen 0°-, 45°- oder 90°-Winkel zur Senkrechten um das Greifzentrum gedreht werden (8). Um sehr weite, unerwartete Bewegungen zu verhindern, sind außerhalb des Bereichs um 45° jeweils nur die beiden an die aktuelle Lage angrenzenden Schaltflächen freigegeben.

Für die Berechnung der Zielposition des TCP wird bei den Richtungen 1, 2 und 4 in Formel 4.1 die Transformationsmatrix ${}^{TCP}_{Base}T$ für ${}^{TCP}_{Jog,orig}T$ verwendet. Bei den übrigen Richtungen wird eine

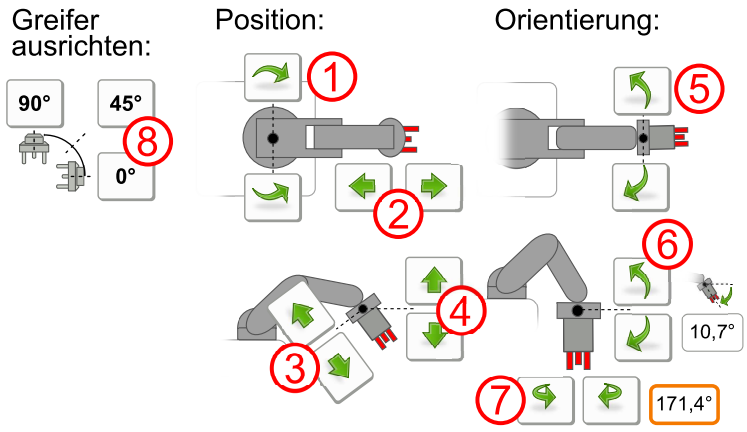


Abb. 4.15.: Roboterbasierte Jog-Steuerung zum groben Platzieren von Kamera oder Greifer

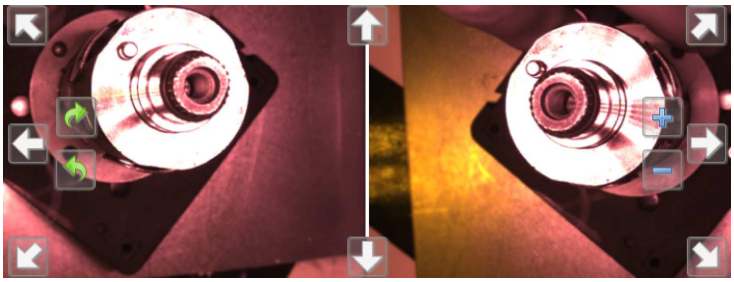


Abb. 4.16.: Kamerabasierte Jog-Steuerung zum Ausrichten des Bildbereichs und Erkennen von störenden Lichteffekten

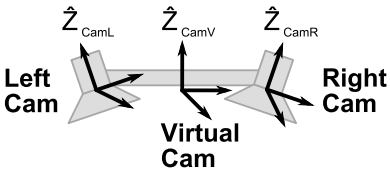


Abb. 4.17.: Lage der virtuellen Kamera im Stereosensor

4x4-Einheitsmatrix für Rotationen um den TCP bzw. ${}^{TCP}_{Grip}T$ für eine Drehung um das Greiferzentrum eingesetzt.

Kamerabasierte Steuerung

Die kamerabasierte Steuerung ist für das Zentrieren des Blickfelds auf den gewünschten Objektbereich ausgelegt. Der Bediener sieht das zyklisch aktualisierte Kamerabild und kann den Roboter über eingeblendete Richtungstasten am Bildrand so verfahren, dass sich die Kamera in die dargestellte Richtung bewegt. Die Abbildung 4.16 verdeutlicht das Prinzip. Zusätzlich kann die Kamera um die Blickachse gedreht und ihr Abstand zum Objekt verändert werden. Treten Lichteffekte auf dem Bauteil auf oder befindet sich dieses nicht innerhalb des Tiefenschärfebereichs, kann der Bediener dies anhand der Kamerabilder direkt erkennen und beheben. Um bei der Verwendung von Stereokameras beide Bilder gleichzeitig überprüfen zu können, werden diese parallel angezeigt. Die Bewegungen beziehen sich dann auf ein virtuelles Koordinatensystem, das sich in der Mitte zwischen den Kameras befindet (siehe Abbildung 4.17).

Die vom Roboter bei Betätigung einer Richtungstaste anzufahrende Zielposition wird ebenfalls über die Formel 4.1 berechnet. Die Transformationsmatrix ${}^{TCP}_{Jog,orig}T$ wird in diesem Fall durch ${}^{TCP}_{Cam}T$ ersetzt. Die relative Lage der Kamera zum TCP kann aus den extrinsischen Parametern der Kamerakalibrierung bestimmt werden. Die Transformationsmatrix ${}^{Jog,orig}_{Relativ}T$ beschreibt die gewünschte Relativbewegung der Kamera. Bei seitlichen Bewegungen entspricht ${}^{Cam,neu}_{Cam}T$ einer homogenen Translationsmatrix mit von null abweichenden Werten für x und bzw. oder y. Vorzeichen und Betrag hängen von der Schrittweite und der gewählten Bewegungsrichtung ab. Der Abstand der Kamera wird über eine Translationsmatrix mit einem entsprechenden Wert für z verändert und die Drehung um z über eine Rotationsmatrix ausgeführt.

Greiferbasierte Steuerung

Beim Einrichten der Werkstückhandhabung kommt es beim Teach-In auf eine sehr hohe Genauigkeit an. Ein störungsfreies Ablegen oder Fügen der Bauteile setzt eine definierte Lage im Greifer voraus. Hierfür muss der Roboter bei der Inbetriebnahme manuell so genau zur Greiffläche zentriert und parallel ausgerichtet werden, dass sich das Werkstück beim Zugreifen nicht außerhalb

der Toleranzen verschiebt. Beim Ablegen ist das Objekt entsprechend mittig in die Aufnahme zu fügen, damit keine Kollisionen, Verklemmungen oder Kratzer am Bauteil auftreten.

Die bei der greiferbasierten Steuerung angebotenen Bewegungsrichtungen sind speziell auf eine Zentrierung ausgelegt. Die Abbildung 4.18 zeigt das Bedienelement, das für einen 3-Finger-Greifer wegen dessen hoher Werkstückflexibilität umgesetzt wurde. Das Prinzip ist ähnlich aber auch auf eine andere Hardware übertragbar, wie die Abbildung 4.19 zeigt. Für eine einfache Zuordnung der Greiferfinger sind diese am System farbig zu markieren und im Bedienelement entsprechen zu symbolisieren. Die in Abbildung 4.18 angezeigten Richtungen ermöglichen eine seitliche Translation des Greifers, so dass sich jeweils ein Finger auf einer Geraden durch das Greifzentrum bewegt (1). Dies ermöglicht ein einfaches und schnelles Zentrieren durch das Angleichen der Abstände der Finger zum Bauteil bzw. eines gegriffenen Werkstücks zur Ablageposition. Als zusätzliche Bewegungsrichtung kann entlang von Geraden durch jeweils zwei Fingerspitzen verfahren werden (2), um z. B. die Greifposition entlang flach zum Greifer liegender Werkstücke anzupassen. Zur Feineinstellung der Orientierung kann von der Translation auf einen Rotationsmodus umgeschaltet werden, bei der das Greifzentrum ortsfest bleibt und der Greifer in die dargestellten Richtungen verkippt. Weiterhin kann der Greifer entlang der Z-Achse verfahren (3) und die Öffnungsweite ist veränderbar (4).

Die Berechnung der Transformationsmatrix ${}^{Jog,orig}_{Relativ}T$ für die relativen Bewegungen erfolgt bei den 2-Backen- und 3-Finger-Greifern mit radialer Fingerbewegung ähnlich wie bei der kamerabasierten Jog-Steuerung. Als Matrix ${}^{TCP}_{Jog,orig}T$ wird ${}^{TCP}_{Grip}T$ verwendet. Bei den in [Sayler 2011] beschriebenen 3-Finger-Stern-Greifern bewegen sich die Finger auf Kreisbahnen, so dass die relative Bewegung in Richtung eines Fingers in Abhängigkeit von der Greiferöffnungsweite berechnet werden muss.

Neben der Funktion zum direkten Drehen des Greifers auf einen Winkel von 0°, 45° oder 90° zur Roboterbasis (siehe (8) in Abbildung 4.15), gibt es in der greiferbasierten Steuerung eine halbautomatische Methode zum senkrechten Ausrichtung an beliebig orientierten Ebenen. Der Roboter kann auf diese Weise schnell und einfach positioniert werden, um lange Bauteile mit senkrechten

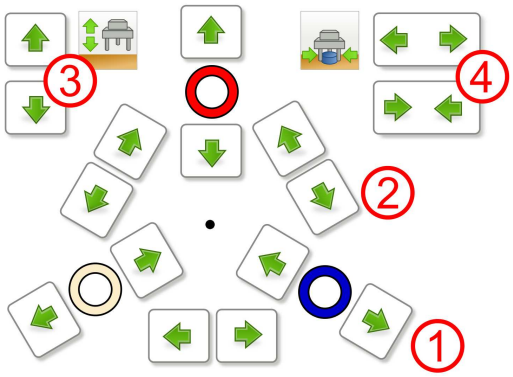


Abb. 4.18.: Jog-Steuerung zum genauen Ausrichten und Zentrieren eines 3-Finger-Greifers

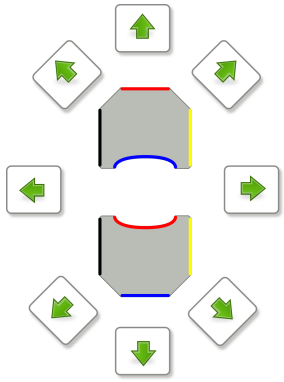


Abb. 4.19.: Alternative Jog-Steuerung für 2-Backen-Greifer

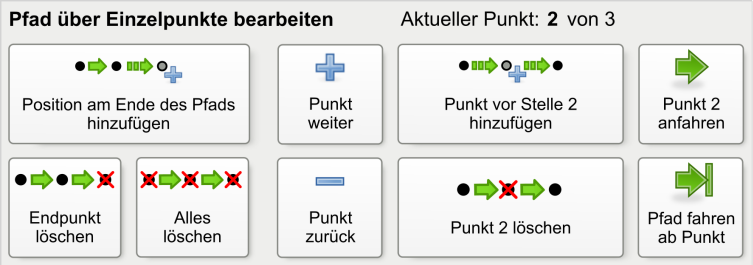


Abb. 4.20.: Bedienelement zum Vorgeben und Bearbeiten von Bahnbewegungen

Bewegungen in tiefe Prozessnester zu fügen oder kollisionsfrei aus diesen herauszuziehen. Das Ausrichten erfolgt ähnlich wie bei der in Abschnitt 4.4.3 beschrieben Parametrierung der kantenbasierten Lageerkennung. Nach einer einmaligen Lokalisierung einer mit Landmarken versehenen Platte, die man auf die relevante Ebene legt, wird deren Lage berechnet und der Roboter entsprechend positioniert.

4.5.3. Definieren und Bearbeiten von Robotertrajektorien

Die Bewegungen für den automatischen Montagebetrieb werden durch Teach-In mit dem Robotersystem in Form von einzelnen Punkten definiert, die durch serielles Abfahren einen Pfad ergeben. Verwendet wird neben den im vorigen Abschnitt erläuterten Jog-Steuerungen vor allem das in Abbildung 4.20 dargestellte Bedienelement, mit dem einzelne Positionen an beliebiger Stelle innerhalb des Pfads eingefügt oder gelöscht werden können. Um die Bewegung zu testen und ggf. direkt zu korrigieren, können erstellte Punkte einzeln oder als Pfad mit dem System angefahren werden. Die Robotergeschwindigkeit kann hierzu jederzeit reduziert werden.
Um dem Bediener die Inbetriebnahme zu erleichtern, werden an die Aufgabe angepasste Standardwerte für die Geschwindigkeit und die Interpolation der Bahnen vorgegeben, so dass Änderungen nur im Sonderfall sinnvoll sind. Um eine geringe Taktzeit zu erreichen, ist die maximale Robotergeschwindigkeit für Bahnbewegungen voreingestellt. Lediglich das Anfahren der letzten Position vor dem Bauteilabgriff sollte mit reduzierter Geschwindigkeit erfolgen, um Kollisionen durch ein Überschwingen des Roboterarms beim Abbremsen zu vermeiden. Für flüssige und schnelle Bewegungen werden alle Bahnen standardmäßig überschliffen abgefahren.

4.6. Zusammenfassung

In diesem Kapitel wurde eine intuitive MMS zur Programmierung von flexiblen Robotersystemen für industrielle Pick-and-Place-Applikationen mit variierenden Objektlagen vorgestellt. Wie die Strukturierung einer Pick-and-Place-Aufgabe zeigt, liegen die Hauptschwierigkeiten für eine schnelle und fehlerfreie Bedienung ohne Erfahrungswissen im genauen Definieren der Roboterpositionen relativ zu verschiedenen Objekten sowie in einer robusten, fehlerfreien Parametrierung der Objektlageerkennung.

Für die Roboterpositionierung wurden innerhalb des Programmablaufs drei Positionstypen mit unterschiedlichen Anforderungen identifiziert. Entsprechend dieser Anwendungsfälle wurden drei Steuerungsmöglichkeiten mit einheitlicher Bedienung umgesetzt. Diese ermöglichen sowohl ein schnelles Verfahren über große Distanzen als auch sehr genaue Positionierungen durch Auswahl einer Bewegungsweite und speziell angepassten Richtungen. Letztere sind für eine fehlerfreie Bedienung speziell auf die Anforderungen zugeschnitten und intuitiv verständlich visualisiert.

Das Einrichten der Objektlageerkennung wurde ebenfalls vereinfacht. Die Auswahl geeigneter Sequenzen von Algorithmen und Sensorsystemen wurde auf die Beantwortung weniger Fragen anhand eines hinterlegten Entscheidungsbaums reduziert. Für eine intuitive Parametrierung der gewählten Verfahren durch einen Bediener ohne entsprechendes Expertenwissen werden u. a. die Einflüsse der einzelnen Parameter über verständlich aufbereitete Zwischenergebnisse der Bildverarbeitungskette visualisiert.

Für eine effiziente Erstellung der Montageablaufpläne wird die in anderen Ansätzen bereits als positiv bewertete iconbasierte Programmierung mit einer wizardbasierten Parametrierung kombiniert, um dem Bediener unterbrechungsfrei durch die vollständige Inbetriebnahme zu leiten.

Das vorgestellte Konzept ist gleichzeitig für ein breites Applikationsspektrum anwendbar und vereinfacht die Bedienung im Vergleich zu industriellen Methoden deutlich. Der Ansatz erfordert keine aufwändige, zugeschnittene Hardware und ist plattformunabhängig, da sich die vorgestellten Konzepte auch auf andere Robotersysteme übertragen lassen. Für die Umsetzung der intuitiven Parametrierung der Lokalisierungsalgorithmen ist jedoch ein Zugriff auf den Quellcode erforderlich, um die Zwischenergebnisse als Nutzerfeedback aufzubereiten und anzuzeigen.

Der vorgestellte Ansatz ist anhand von Versuchen mit Probanden aus der Zielgruppe und realen Applikationen bzgl. der geforderten schnellen und robusten Programmierung ohne Erfahrungswissen zu evaluieren.

5. Entwicklung einer strategiebasierten, autonomen Störungsbehandlung

In diesem Kapitel wird ein Ansatz zur Steigerung der Robustheit von flexiblen Robotersystemen beschrieben. Wie in Abschnitt 2.3 erläutert, können Störungen infolge variabler Umgebungsbedingungen im betrachteten Kontext nicht vollständig vermieden werden. Das Ziel ist es stattdessen, daraus resultierende Ausfälle und Bedienereingriffe im Betrieb zu verhindern.

Eine automatische Erkennung des ursächlichen Fehlers bzw. das Ableiten einer passenden Gegenmaßnahme wäre speziell bei den identifizierten Störungen der Bildverarbeitung nur mit einem sehr hohen Aufwand zur Anpassung und Parametrierung der Algorithmen möglich. Als Ansatz werden daher Behandlungsstrategien zur autonomen Verarbeitung der auftretenden Störungen eingesetzt. Die Auswahl der jeweils zulässigen Strategien erfolgt durch einen Bediener, der die Anforderungen und Randbedingungen der Applikation kennt und berücksichtigen kann. Für eine schnelle Inbetriebnahme ohne Expertenwissen sind die Behandlungsstrategien anhand weniger Parameter an die Gegebenheiten anpassbar. Treten zur Laufzeit Störungen auf, werden die zugewiesenen Strategien bis zu einer erfolgreichen Behandlung autonom nacheinander ausgeführt. Die optimale Reihenfolge wird anhand statistischen Erfahrungswissens über einen Partially Observable Markov Decision Process (POMDP) berechnet, um eine geringe Behandlungsdauer zu erreichen.

Die Grundlagen zur Funktion und Anwendung eines POMDP werden zunächst in Abschnitt 5.1 erläutert, bevor das Gesamtkonzept der Störungsbehandlung in Abschnitt 5.2 im Detail vorgestellt wird. Der Abschnitt 5.3 beschreibt das für die Optimierung der Ausführungsdauer verwendete Modell des POMDP und erläutert die erforderlichen Parameter und Annahmen. Die zur Behandlung eingesetzten Strategien teilen sich in vier unterschiedliche Typen. Der Abschnitt 5.4 verdeutlicht ihre Funktion und die zur Inbetriebnahme einzustellenden Parameter. Für die Berechnung der optimalen Ausführungsreihenfolge der Strategien sind verschiedene statistische Daten erforderlich, die als Erfahrungswissen aus vorherigen Störungsfällen ermittelt werden, wie der Abschnitt 5.5 zeigt. Um stets die statistisch beste Behandlung zu ermöglichen, muss die anzuwendende Strategiesequenz innerhalb eines Bearbeitungszyklus aktualisierbar sein. In Abschnitt 5.6 sind hierzu verschiedene Maßnahmen zur Reduktion des Rechenaufwands beschrieben. Zuletzt gibt der Abschnitt 5.8 einen Überblick über die zum Einrichten der Störungsbehandlung verwendete MMS, die eine schnelle Inbetriebnahme ohne Erfahrungswissen ermöglicht.

5.1. Grundlagen zum Partially Observable Markov Decision Process

Der Partially Observable Markov Decision Process (POMDP) gehört zur Gruppe der zeitdiskreten, statistischen Markov Prozesse, mit denen sich Zustandsübergängen eines Systems unter Berücksichtigung von Unsicherheiten modellieren lassen. Als Voraussetzung für die Anwendbarkeit muss das nicht-deterministische Systemverhalten in Form von Wahrscheinlichkeiten bekannt sein. Zusätzlich muss die Markov-Annahme gelten, nach der zukünftige und vergangene Systemzustände voneinander unabhängig sind. Ein Folgezustand hängt somit nur von der ausgeführten Aktion sowie dem aktuellen Zustand ab. Eine Kenntnis über frühere Zustände und Aktionen bringt keine Vorteile. Ausgehend von den jeweiligen Randbedingungen und Eigenschaften lassen sich Markov Prozesse in verschiedene Kategorien einteilen. Wie Abbildung 5.1 zeigt, sind wichtige Kriterien eine vollständige Beobachtbarkeit der Systemzustände sowie die Beeinflussbarkeit der Zustandsübergänge durch eine aktive Auswahl der auszuführenden Aktionen. Da letztere Bedingung sowohl beim POMDP als auch beim Markov Decision Process (MDP) gegeben ist, eignen sich beide zur Planung von Aktionsfolgen mit statistisch optimalem Resultat. Die Optimierung der Lösungen erfolgt bezüglich der Kosten oder Belohnungen für die Ausführung von Aktionen oder das Erreichen von Zuständen. Als Unterschied zum reinen MDP wird ein POMDP für Systeme eingesetzt, bei denen der aktuelle Zustand z.B. durch Rauschen oder andere Sensorfehler nicht vollständig deterministisch erfassbar ist. Daher wird der aktuelle Systemzustand anhand von Beobachtungen sowie einem probabilistischen Modell geschätzt. Es ergibt sich eine Wahrscheinlichkeitsverteilung über alle möglichen Zustände, die bei jedem Schritt der Planung einer optimalen Handlungssequenz zu berücksichtigen ist. Die nachfolgende Beschreibung der Grundlagen von POMDP basieren auf [Kaelbling u. a. 1998], [Littman 1996], [Cassandra 1998] sowie [Thrun u. a. 2005]. Für weitere Details wird auf diese Quellen verwiesen.

Beschreibung eines POMDP

Ein POMDP kann durch das 6-Tupel <S,A,T,R,O,Z> beschrieben werden:

- $\mathcal{S} = \{s_0, s_1, \ldots, s_n\}$ stellt die Menge der möglichen Systemzustände dar
- $\mathcal{A} = \{a_0, a_1, \ldots, a_m\}$ definiert die Menge der auswählbaren Aktionen

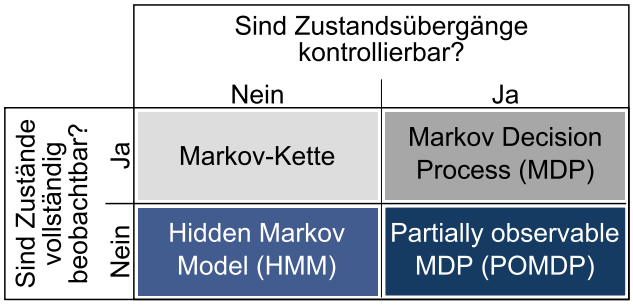


Abb. 5.1.: Einordnung verschiedener Markov-Prozesse

- $T : \mathcal{S} \times \mathcal{A} \rightarrow \prod(\mathcal{S})$ gibt als Transitionsfunktion die Wahrscheinlichkeitsverteilung der Folgezustände abhängig von der ausgeführten Handlung an. $T(s,a,s')$ stellt die Wahrscheinlichkeit dar, durch die Aktion a aus dem Zustand s in s' überzugehen
- $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ definiert als Reward-Funktion die Kosten $R(s,a)$ der Ausführung einer Aktion a im Zustand s. Abhängig vom Vorzeichen sind sowohl Strafen als auch Belohnungen möglich.
- $\mathcal{Z} = \{z_1, z_2, \ldots, z_l\}$ stellt die Menge der möglichen Beobachtungen dar
- $O : \mathcal{S} \times \mathcal{A} \rightarrow \prod(\mathcal{Z})$ zeigt als Beobachtungsfunktion die Auftretenswahrscheinlichkeit der Beobachtungen. $O(s_i, a_j, z_k) = p(z_k|s_i, a_j)$ gibt die Wahrscheinlichkeit zur Beobachtung von z bei Ausführung der Aktion a im Zustand s an

Ausgehend von einer Zustandsmenge $\mathcal{S}$ können verschiedene Aktionen aus $\mathcal{A}$ ausgeführt werden, die das System mit gewissen Wahrscheinlichkeiten (in T) in einen der möglichen Folgezustände überführen. Dieser ist nicht direkt erfassbar, sondern kann nur anhand von Beobachtungen (z_i) geschätzt werden, die mit unterschiedlichen Wahrscheinlichkeiten (in O) auftreten. Sowohl für die Ausführung einer Handlung als auch für das Erreichen eines Zustands können Kosten oder Belohnungen (in R) anfallen.

Belief State

Durch die eingeschränkte Beobachtbarkeit kann der aktuelle Systemzustand nicht sicher bestimmt werden. Er lässt sich nur aus einer Anfangshypothese und allen zurückliegenden Beobachtungen schätzen. Daher erfordert die Ermittlung des Folgezustands eine entsprechende Kenntnis der Vergangenheit. Dies macht die Berechnung mit zunehmendem Zeitschritt nicht nur stetig komplexer, es verletzt auch die Markov-Annahme. Aus diesem Grund wird der sogenannte Belief State b eingeführt, der die aktuelle Wahrscheinlichkeitsverteilung über alle Zustände angibt und damit alle vergangenen Beobachtungen beinhaltet. Da in der Folge Belief States anstatt der Systemzustände verwenden werden, ist auch die Markov-Annahme wieder erfüllt. Ein Folgebelief b' wird mittels eines Bayes-Filters nach [Kaelbling u. a. 1998] aus dem aktuellen Belief b, der Aktion a und der Beobachtung z über die Formel 5.1 berechnet.

$$\begin{aligned} b'(s') &= p(s'|z,a,b) \\ &= \alpha p(z|s',a,b)p(s'|a,b) \\ &= \alpha p(z|s',a)\sum_{j=1}^{|\mathcal{S}|} p(s'|a,b,s_j)p(s_j|a,b) \\ &= \alpha O(s',a,z)\sum_{j=1}^{|\mathcal{S}|} T(s_j,a,s')b(s_j) \end{aligned} \tag{5.1}$$

α stellt dabei den Normalisierungsfaktor dar. Als Ausgangslage der ersten Berechnung ist ein initialer Startbelief b_0 erforderlich.

Policy

Mittels eines POMDP soll für jeden Zeitschritt eine vom Agenten auszuführende Aktion berechnet werden, über die der Gesamtbenefit der Handlungen maximiert werden kann. Eine Vorschrift, die jedem Belief State eine Aktion zuweist, wird Policy genannt. Der Gesamtnutzen einer Policy p ergibt sich aus der erwarteten Summe der Kosten oder Belohnungen, die für jede ausgeführte Handlung bzw. jeden erreichten Zustand vergeben werden. Da die zukünftigen Belohnungswerte aufgrund der probabilistischen Zustandsübergänge und Beobachtungen nicht vorhersagbar sind, erfolgt die Maximierung anhand der statistisch zu erwartenden Belohnungen. Die Policy mit dem höchsten erwarteten Gesamtnutzen wird als optimal bezeichnet und kann z.B. mittels Value Iteration berechnet werden.

Value Iteration

Zur Veranschaulichung der Berechnung kann eine Policy als Baum dargestellt werden, der die Abfolge aller möglichen Aktionen entsprechend der Beobachtungen enthält (siehe Abbildung 5.2). Die Knoten symbolisieren jeweils einen Belief State mit der auszuführenden Aktion. Die Kanten repräsentieren die möglichen Beobachtungen, die zu einem Folge-Belief führen. Der oberste Knoten gibt die zur Policy p gehörende Startaktion $a(p)$ an. Zur Bestimmung des Gesamtnutzens einer Policy, müssen alle erwarteten zukünftigen Belohnungen berücksichtigt werden. Dabei ist die Anzahl der nachfolgend erlaubten Aktionsausführungen von Bedeutung, da der Nutzen einer Handlung abhängig von der verbleibenden Schrittmenge variieren kann. Wird eine Sequenz aus nur einer Handlung betrachtet, ergibt sich der Gesamtnutzen über die Formel 5.2 aus der Belohnung für die Ausführung dieser Aktion im entsprechenden Zustand.

$$V_p(s) = R(s, a(p)) \qquad \text{für } t = 1 \tag{5.2}$$

Umfasst die Handlungssequenz mehrere Aktionen, wird der Gesamtnutzen iterativ aus den direkten sowie allen zukünftig zu erwartenden Belohnungen nach Formel 5.3 bestimmt.

$$\begin{aligned} V_p(s) &= R(s, a(p)) + \gamma \cdot \text{Erwarteter Wert der Zukunft} \\ &= R(s, a(p)) + \gamma \sum_{s' \in S} T(s, a(p), s') \sum_{z_i \in \mathcal{Z}} O(s', a(p), z_i) V_{z_i(p)}(s') \end{aligned} \tag{5.3}$$

p stellt hier einen Policy-Baum mit t verbleibenden Schritten dar und $z_i(p)$ ist der aus der Beobachtung z_i folgende Policy-Unterbaum mit $t-1$ verbleibenden Schritten. Die Formel berücksichtigt demnach alle möglichen Folgezustände s' entsprechend der möglichen Beobachtungen z_i sowie der Übergangsfunktion T. Der zu erwartende zukünftige Nutzen geht rekursiv über das Ergebnis der jeweils zugehörigen Value Funktion aus dem Schritt $t-1$ ein. Der Faktor γ kann über eine Belegung zwischen 0 und 1 als sogenannter Discount-Wert dazu verwendet werden, in späteren Schritten erzielte Belohnungen anteilig geringer zu gewichten, um schnelle Lösung zu bevorzugen.

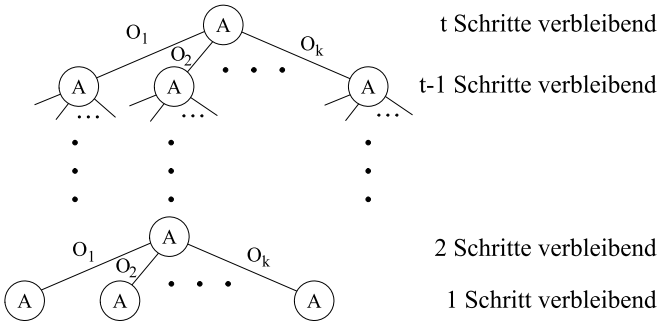


Abb. 5.2.: Ein t-Schritt Policy Baum nach [Kaelbling u. a. 1998]

Dies ist insbesondere bei Policies mit einer unendlichen Anzahl erlaubter Schritte nötig, um einen endlichen und somit vergleichbaren Gesamtnutzen zu erhalten.
Aufgrund der fehlenden Beobachtbarkeit der Systemzustände sind diese bei der Berechnung der Value-Funktion $V_p(s)$ jedoch nicht bekannt. Daher muss auf die Belief-Verteilung zurückgegriffen werden, mit der sich die Formel 5.3 für eine 1-Schritt-Policy in 5.4 überführen lässt.

$$V_p(b) = \sum_{s \in S} b(s) R(s, a(p)) \qquad \text{für } t = 1 \tag{5.4}$$

Der Nutzen einer 1-Schritt-Policy entspricht damit der Summe der Belohnungen für alle möglichen Zustände gewertet mit deren Wahrscheinlichkeit. Für Mehrschritt-Policies erweitert sich 5.3 entsprechend zu 5.5.

$$V_p(b) = \sum_{s \in S} b(s) V_p(s) \tag{5.5}$$

Damit gibt das Ergebnis der Value-Funktion jeweils den erwarteten Nutzen einer Policy p an, wenn die zugehörige Startaktion $a(p)$ ausgeführt wird. Um die optimale Policy V_π zu erhalten, sind nach Formel 5.6 alle Policies aus der endlichen Menge $\mathcal{P}$ zu ermitteln und für jeden Belief State jeweils diejenige auszuwählen, die den höchsten Nutzen erwarten lässt.

$$V_\pi(b) = \max_{p \in \mathcal{P}} V_p(b) \tag{5.6}$$

Der Verlauf des Nutzens einer Policy ist dabei linear über dem Belief State b (für nähere Informationen siehe [Thrun u. a. 2005]). Die Abbildung 5.3 zeigt den Verlauf des Nutzens von drei verschiedenen Policies für ein System mit zwei Zuständen und $t = 1$. Die Belief-Verteilung ist 1-dimensional, da $p(z_2) = 1 - p(z_1)$. Zusätzlich ist die optimale policy dargestellt, die sich jeweils aus den höchsten Teilstücken der einzelnen Policies zusammensetzt und stückweise linear sowie konvex ist (siehe [Thrun u. a. 2005]). Ebenfalls erkennbar ist, dass nicht alle Policies zum Optimum beitragen, da einzelne Verläufe über die gesamte Belief-Verteilung von anderen dominiert werden. Die entsprechenden Policies können zur Vereinfachung der nachfolgenden Rekursionen aus $\mathcal{P}$ entfernt werden. Da diese Policies bereits im aktuellen Zeitschritt nicht zur optimalen Lösung beitragen, werden sie auch keine Auswirkungen auf nachfolgende Zeitschritte haben, die auf

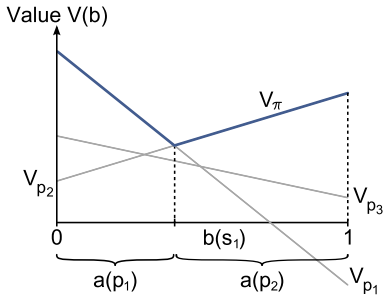


Abb. 5.3.: Value-Funktion für verschiedene beliefs

dem aktuellen aufbauen. Das Löschen der unbeteiligten Lösungen wird als Pruning bezeichnet.

5.2. Konzept einer effizienten, parametrierbaren Störungsbehandlung

Das Erreichen einer hohen Robustheit des Montagebetriebs ist eine wichtige Voraussetzung für eine wirtschaftliche Automatisierung der Kleinserienmontage. Wie in Abschnitt 2.3.3 analysiert, können Störungen bei der Objektlageerkennung und der Bauteilhandhabung im vorliegenden Anwendungsfall nicht vollständig ausgeschlossen werden. Zur Reduzierung von Maschinenstillständen und erzwungenen Bedienereingriffen müssen Störungen detektiert und mögliche resultierende Ausfälle verhindert werden. Im Folgenden wird das im Rahmen dieser Arbeit entwickelte Konzept zur automatischen Störungsbehandlung erläutert.

Detektion auftretender Störungen

Die Objektlageerkennung wird innerhalb des Montageablaufs aktiv aufgerufen. Auftretende Störungen können daher direkt anhand des Rückgabewerts der Funktion detektiert werden. Bei der Lokalisierung mittels kanten- und oberflächenbasierter Verfahren erfolgt ein Abgleich zwischen den bei der Inbetriebnahme festgelegten Referenzmerkmalen und den im Kamerabild segmentierten Merkmalen. Der Rückgabewert entspricht dem Grad der Übereinstimmung. Wird ein definierter Schwellwert überschritten, gilt das Objekt als erkannt, anderenfalls ist eine Störung aufgetreten. Befinden sich mehrere Bauteile im Bild, kann die Hypothese mit der höchsten Übereinstimmung ausgewertet werden. Bei der landmarkenbasierten Lokalisierung wird neben der Ähnlichkeit der Kreise, die anhand verschiedener Referenzgrößen bewertet werden, auch deren Anzahl überprüft. Werden zu wenig Landmarken detektiert oder können die drei relevanten aus mehreren Kandidaten nicht sicher identifiziert werden, wird ebenfalls eine Störung gemeldet.
Bei der Bauteilmanipulation kann die Störungsdetektion über eine sensorische Erfassung des Greiferöffnungswinkels beim kraftbasierten Griff erfolgen. Hierzu wird ein Referenzwert bei der Inbetriebnahme der Manipulation gespeichert, da diese vom Bediener durchgeführt und überwacht wird. Fehlen im Automatikbetrieb Bauteile beim Zugreifen oder werden während des Transports verloren, schließt der Greifer vollständig und die Störung wird erkannt. Verrutschen die Teile beim

Transport oder tritt eine unerwartete Bauteillage beim Greifen auf, führt dies zu einer unbekannten Greifposition. Diese wird als Störung erkannt, wenn die Greiferöffnungsweite dadurch vom Referenzwert abweicht. Da dies nicht immer zwingend eintritt, werden bei der Lokalisierung hohe Schwellwerte für die Merkmalszuordnung verwendet, um ungenau erkannte Bauteillagen zu verhindern. Dies führt bei der Lageerkennung zu häufigeren Störungen, die jedoch sicher erkannt und nach Möglichkeit behandelt werden. Um Störungen bei der Manipulation vor einer Kollision zu detektieren, erfolgt die Prüfung der Greiferöffnungsweite stets nach dem Zugreifen, nach dem Abrücken des Bauteils vom Untergrund sowie vor dem Anrücken zur Ablageposition und vor dem Öffnen des Greifers.

Sollte diese Störungsdetektion bei der Bauteilmanipulation nicht ausreichen, könnte eine zusätzliche Kamera an der Ablagestelle wie in [Di u. a. 2009] beschrieben eingesetzt werden, um abweichende Objektlagen zu detektieren. Dies erfordert neben der Hardware einen zusätzlichen Inbetriebnahmeaufwand und erhöht die Taktzeit auch in störungsfreien Zyklen. Dieser Ansatz wird im Rahmen dieser Arbeit nicht verfolgt.

Recovery - Überführung des Systems in einen störungsfreien Zustand

Wurden auftretende Störungen im Montageablauf detektiert, sind die Ausfälle zu verhindern, die ohne ein Eingreifen in der Regel folgen. Für die sichere Auswahl einer geeigneten Störungsbehandlung wäre die Kenntnis des konkreten Fehlers erforderlich. Dieser ist sowohl bei der Lokalisierung als auch bei der Bauteilhandhabung nur sehr komplex zu erfassen. Bei Störungen der Manipulation wären aufwändige Zusatzsensoren erforderlich, um die genaue Lageabweichung vom Bauteil im Greifer für einen gezielten Ausgleich zu erfassen oder z.B. durch eine taktile Regelung zu kompensieren. Bei Störungen der Objektlageerkennung ist eine automatische Diagnose der Ursache aufgrund der Vielzahl an Fehlermöglichkeiten nach heutigem Stand der Technik für den Anwendungsfall zu komplex. Da eine geeignete Behandlung stets auch applikationsabhängig ist und unpassende Maßnahmen zu Folgeschäden führen könnten, ist eine automatische Störungsbehandlung mit global definierten Maßnahmen nicht möglich.

Der in dieser Arbeit vorgestellte Ansatz zur unterbrechungsfreien Behandlung der Störungen bezieht daher den Bediener in die Auswahl geeigneter Maßnahmen mit ein. In einer globalen Datenbank werden hierzu verschiedene Behandlungsmöglichkeiten für alle identifizierten Fehlerfälle als Strategien abgelegt. Der Bediener kann geeignete und zulässige Maßnahmen anhand seines Applikationswissens für die autonome Verwendung im Montagebetrieb freigeben. Bei Störungen der Manipulation reicht zumeist eine Behandlungsmöglichkeit für verlorene bzw. fehlende Objekte sowie wenige Methoden in fester Reihenfolge zum Verarbeiten von unerwarteten Objektlagen. Bei der Lokalisierung können unterschiedliche Fehler auftreten, die oft nicht mit derselben Strategie zu beheben sind. Entsprechend können mehrere, sehr verschiedene Maßnahmen ausgewählt werden. Wie in Abbildung 5.4 dargestellt, verfügt jede Greif- und Ablege-Aktion über eine eigene lokale Datenbank an manuell zugewiesenen Strategien. Diese werden im Störungsfall nach einander bis zur erfolgreichen Behandlung ausgeführt. Ist keine geeignete Maßnahme vorhanden oder waren alle

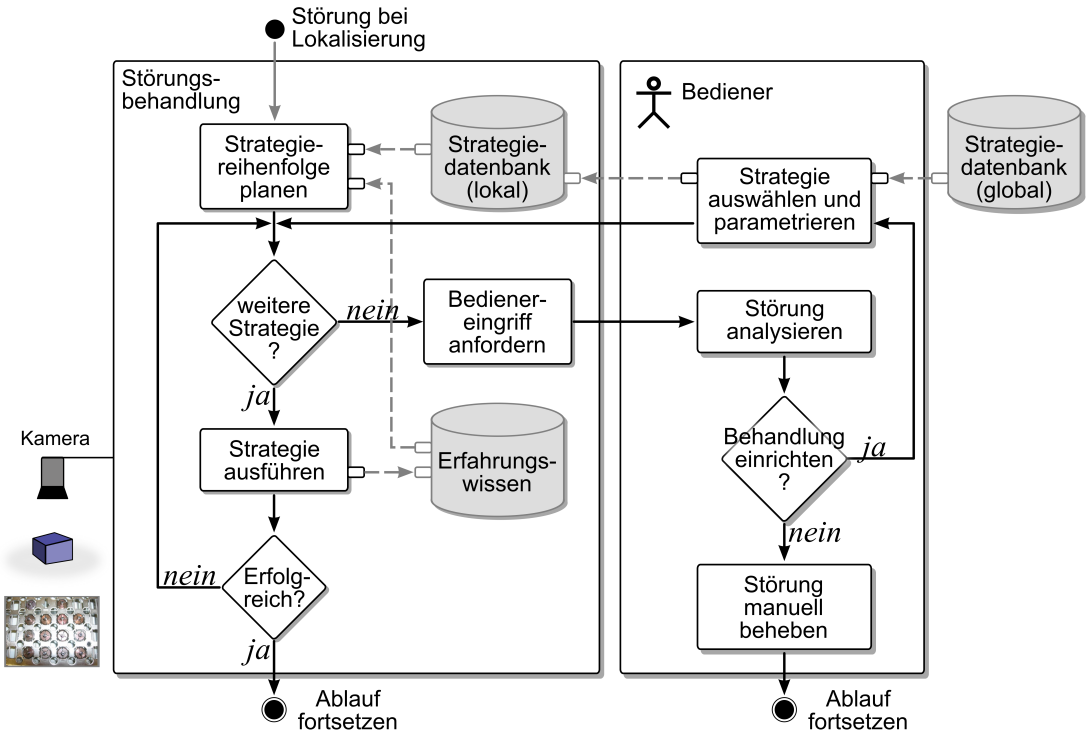


Abb. 5.4.: Überblick über das Gesamtkonzept zur strategiebasierten Störungsbehandlung

zugewiesenen Strategien erfolglos, wird die Ausführung des Arbeitsplans gestoppt und ein manueller Eingriff gefordert. Der Bediener entscheidet dann, ob er dem System eine passende Strategie vorgibt, um ähnliche Fehler zukünftig autonom zu behandeln. Zur Unterstützung der manuellen Fehlerdiagnose werden die Störungsmeldung sowie die Kamerabilder auf dem Monitor angezeigt. Zusätzlich geben die Stellung von Roboter und Greifer sowie die Lage des Bauteils Aufschluss über mögliche Ursachen. Hat der Bediener eine Strategie hinzugefügt, mit der die Störung behoben werden konnte, wird der Montagebetrieb fortgesetzt. Durch die zusätzlich erlaubten Maßnahmen wird die Robustheit der Ausführung iterativ erhöht.

Um häufige Störungen bereits von Beginn an behandeln zu können, lassen sich die zulässigen Strategien bereits bei der Inbetriebnahme zuweisen. Sie verfügen dabei über nur wenige, verständliche Parameter, um im Rahmen der Anforderungen an die MMS eine schnelle Anpassung an die Applikation zu ermöglichen und kein Expertenwissen zu erfordern.

Stehen bei der Lokalisierung im Störungsfall verschiedene Behandlungsstrategien zur Auswahl, wird die Reihenfolge ihrer Ausführung über einen partially observable markov decision process festgelegt. Hierzu wird der aktuelle Fehlerzustand anhand von Wahrscheinlichkeiten geschätzt und eine statistisch optimale Strategiereihenfolge geplant. Der Algorithmus ermöglicht so eine komplexe Behandlung verschiedener Fehlerfälle mittels unterschiedlicher Strategietypen und reduziert die zur Störungsbehebung erforderliche Taktzeit. Die benötigten statistischen Daten werden autonom als Erfahrungswissen aus den Resultaten vorhergehender Störungsbehandlungen gewonnen (siehe Abschnitt 5.5).

Dieser strategiebasierte Ansatz und die statistische Optimierung der Ausführungsreihenfolge ermöglichen eine einfach und schnell einzurichtende Störungsbehandlung und führen nur im Störungsfall zu einer geringen Taktzeiterhöhung. Zusätzlich hat der Bediener die volle Kontrolle über die ausführbaren Maßnahmen und kann die Zulässigkeit der Behandlung sicherstellen.

5.3. Modellierung der Störungsbehandlung als POMDP

Der Einsatz eines POMDP zur Steuerung einer Störungsbehandlung bei Montageaufgaben ist nach Kenntnis des Autors neuartig und wird daher zunächst motiviert, bevor das zu Grunde liegende Modell erläutert wird.

5.3.1. Motivation zum Einsatz eines POMDP

Wie in Abschnitt 2.3 erläutert, können bei der durch flexible Robotersysteme automatisierten Montage verschiedene Störungen bei der Bauteilmanipulation und der Objektlageerkennung auftreten. Durch die hohe Bandbreite an ursächlichen Fehlern ergibt sich bezogen auf die Lokalisierung eine Vielzahl an Störungszuständen, in denen sich das System befinden kann. Wie die Abbildung 5.5 verdeutlicht, unterteilen sich diese in *Lokalisierung permanent verhindert, Objekt fehlt* sowie verschiedene Varianten, wie die *Lokalisierung durch veränderliche Effekte verhindert* sein kann. Pro Zustand gibt es zumeist mindestens eine geeignete Strategie, die das System autonom wieder in den störungsfreien Betrieb überführt. Eine ideale Behandlungssteuerung würde abhängig vom ursächlichen Fehler direkt diese passende Strategie auswählen bzw. den Bediener verständigen, wenn der störungsfreie Zustand nicht autonom erreichbar ist.

In der realen Anwendung wird bei Auftreten einer Störung der Objektlokalisierung nur ein unspezifischer Misserfolg gemeldet. Wie bereits beschrieben, ist eine detailliertere Diagnose nach aktuellem Stand der Technik zu komplex. Die vom störungsfreien Betrieb abweichenden Zustände sind daher nicht vollständig beobachtbar, so dass sich die zur Behandlung geeigneten Aktionen nicht direkt ableiten lassen.

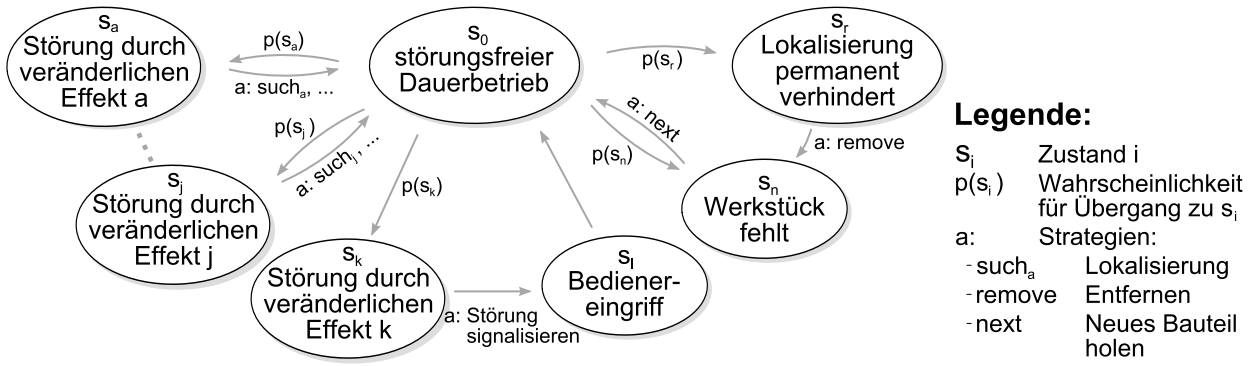


Abb. 5.5.: Nicht-beobachtbares Zustandsmodell des Systems im Bezug auf mögliche Störungen

Durch veränderliche Effekte verursachte Störungen könnten dennoch durch einen sequentiellen Aufruf aller Suchstrategien behoben werden, da ungeeignete Behandlungen keinen Einfluss auf das Bauteil haben und keine Folgeschäden verursachen. Abhängig von der Anzahl der eingesetzten Strategien und ihrer Wirksamkeit kann es bis zum Aufruf einer passenden Behandlung jedoch sehr lange dauern. Dieser Effekt wird noch verstärkt, wenn der Bediener wie im Konzept vorgesehen möglichst viele Strategien unabhängig von den bisherigen Störungsfällen prophylaktisch zuweist. Abhängig von den häufig sehr niedrigen Taktzeiten, könnte dies bereits den Einsatz flexibler Robotersystem in vielen Montagelinien verhindern.

Ist kein Objekt vorhanden, keine geeignete Strategie freigegeben oder eine Lokalisierung permanent verhindert, werden in jedem Fall alle zugewiesenen Behandlungsmöglichkeiten ausgeführt. Eine Unterscheidung dieser drei Fehlerfälle ist anhand der unspezifischen Rückmeldung über Lokalisierungsstrategien allein nicht möglich. Bei fehlendem Bauteil bestünde die geeignete Behandlung darin, mit dem nächsten fortzufahren. Wurde ein vorhandenes Objekt nur nicht lokalisiert, könnte diese Behandlung aber zu schweren Folgeschäden führen. Für eine Differenzierung beider Fälle sind daher Strategien zur Anwesenheitsprüfung erforderlich. Da diese Behandlungsklasse die Störung aber nicht beheben kann, sind zusätzliche auch Strategien zum Entfernen von Bauteilen bzw. zum Beschaffen neuer Werkstücke nötig.

Um die unterschiedlichen Störungen zulässig behandeln zu können und die Taktzeit im Störungsfall gleichzeitig nicht unnötig zu erhöhen, sind geeignete Entscheidungen bezüglich der Strategiereihenfolge und ihrer Auswahl erforderlich. Im Detail sind folgender Fragen zu beantworten:

- Welche Reihenfolge der Lokalisierungsstrategien ist optimal?
- Ist es wirtschaftlich sinnvoll alle zugewiesenen Lokalisierungsstrategien auszuführen?
- Wann und wie häufig ist die Objektanwesenheit für eine zuverlässige Aussage zu prüfen?
- Wann kann ein Objekt als fehlend bzw. als nicht lokalisierbar angenommen werden?

Eine von der Applikation abhängige, geeignete Beantwortung dieser Fragen durch einen unerfahrenen Bediener wird als zu komplex angesehen. Stattdessen soll die Strategieausführung anhand statistischer Wahrscheinlichkeiten gesteuert werden. Wegen der fehlenden Beobachtbarkeit der Systemzustände und der Möglichkeit, Strategien aktiv aufzurufen, eignet sich ein POMDP zur Lösung dieses Entscheidungsproblems.

5.3.2. Modellierung des Zustandsraums

Wie in Abschnitt 5.1 erläutert, kann ein POMDP durch das 6-Tupel $< S, A, T, R, O, Z >$ sowie den Start-Belief b_0, den Horizont H und den Discount-Faktor γ beschrieben werden. Entsprechend sind diese Bestandteile zu bestimmen. Die Wahrscheinlichkeiten für die Transitionsübergänge T und die Beobachtungen O können aufgrund der Komplexität nicht durch den Bediener vorgegeben

werden. Sie sollen als Erfahrungswissen im laufenden Betrieb gelernt werden. Da die Einteilung der Systemzustände einen großen Einfluss auf die Erlernbarkeit der statistischen Daten hat, ist diese Anforderung bei der Modellierung des POMDP zu berücksichtigen.

Eine strikte Trennung der Systemzustände nach auftretenden Fehlern, wie in Abbildung 5.5 dargestellt, ist innerhalb des betrachteten Anwendungsszenarios nicht möglich. Um die Wahrscheinlichkeit zu lernen, durch eine Lokalisierungsstrategie in den störungsfreien Zustand überzugehen, muss das Ausführungsergebnis dem Systemzustand jeweils zugeordnet werden. Wäre pro Störung stets nur eine Strategie erfolgreich, stünde der Ausgangszustand nach einer erfolgreichen Behandlung fest. In der Realität kann eine Behandlungsmethode jedoch mehrere Störungen beheben. Das Variieren der Kameraposition kann z.B. eine Verdeckung der relevanten Merkmale aufheben oder die Position einer zuvor störenden Reflexion auf dem Bauteil ändern. Das Erfahrungswissen wäre somit keinem Systemzustand zuzuordnen.

Definiert man im Gegensatz dazu einen Zustand als Menge alle Störungen, die mit einer bestimmten Strategie behoben werden, steht der jeweilige Ausgangszustand nach einer erfolgreichen Ausführung fest. Diese Aufteilung würde jedoch dazu führen, dass das System in mehr als einem Zustand gleichzeitig wäre, wenn verschiedene Behandlungsmöglichkeiten eine Störung lösen. Für eine eindeutige Zuordnung der Zustände müssten alle Strategien hinsichtlich der Erfolgs- und der Misserfolgsmöglichkeit durch Permutation kombiniert werden. Der Zustandsraum würde schnell sehr groß. Zusätzlich müssten für die Detektion der Zustände und die Zuordnung des Erfahrungswissens stets alle Strategien ausgeführt werden.

Eine geeignete Zustandsaufteilung ergibt sich hingegen, wenn man alle durch veränderliche Effekte verursachten Störungen aus Abbildung 5.5 zu einem Systemzustand zusammenfasst und daneben *Werkstück fehlt* und *Lokalisierung permanent verhindert* beibehält. Wird das Fehlen eines Objekts durch Anwesenheitsprüfungen festgestellt, ist die Störungsursache und damit der Systemzustand detektiert. Dasselbe gilt bei erfolgreicher Ausführung einer Lokalisierungsstrategie bzw. wenn alle zugewiesenen Behandlungsmethoden dieses Typs fehlschlagen. Entsprechend können die Resultate aller Strategieaufrufe nach einer abgeschlossenen Behandlung als Erfahrungswissen dem passenden Zustand zugeordnet und die relative Häufigkeit der Störungsursachen für den Startbelief b_0 erfasst werden.

Bei dieser Modellierung wäre es jedoch häufig ineffizient, bei einer permanent verhinderten Lokalisierung alle entsprechenden Strategien zur Zustandszuordnung ausführen zu müssen. Je mehr Behandlungsmöglichkeiten zugewiesen wurden, desto wahrscheinlicher ist es, dass mehrere Methoden nur geringe Erfolgschancen bieten. Wären bei permanenter Verhinderung zwangsläufig alle Strategien auszuführen, würde zuletzt eventuell viel Zeit für eine nur geringe zusätzliche Erfolgsaussicht verbraucht. Bei der Planung über einen POMDP mit dem vorliegenden Modell wird unabhängig von den Zuständen und der Anzahl der noch verbleibenden Strategien abgewogen, ob eine weitere Lokalisierung mehr Nutzen als das vorzeitige Entfernen des Objekts bringt. Die Trennung von *Lokalisierung permanent verhindert* und *Störung durch veränderliche Effekte* hat entsprechend

keinen Einfluss auf die Policy. Beide Zustände werden daher zu *vorhandenes Werkstück nicht lokalisiert* zusammengefasst. Die Abbildung 5.6 zeigt das resultierende Modell. Die Zuordnung und Berücksichtigung des Erfahrungswissens für die Lokalisierungsstrategien erfolgt nur bei erfolgreicher Lageerkennung. Gilt das Objekt als nicht lokalisierbar und wird entfernt, geht nur das Ergebnis dieser letzten Strategie in das Erfahrungswissen ein.

Das angepasste Modell des POMDP lässt sich über das 6-Tupel <S,A,T,R,O,Z> beschreiben:

- $\mathcal{S} = \{s_1, s_n\}$
- $\mathcal{A} = \{a_{such,1}, a_{such,2}, \ldots, a_{such,m}, a_{check}, a_{remove}, a_{next}\}$
- $\mathcal{T}$: Die Tabellen 5.1 bis 5.4 zeigen die Übergangswahrscheinlichkeiten getrennt nach Aktionen.
- $\mathcal{R}$: $R(a_{such,i}) = -t(a_{such,i})$; $R(a_{check}) = -t(a_{check})$; $R(a_{remove}|s_1) = -t(a_{remove}) - r_{remove,1}$; $R(a_{remove}|s_n) = -t(a_{remove}) - r_{remove,n}$; $R(a_{next}|s_1) = -r_{next,1}$; $R(a_{next}|s_n) = -r_{next,n}$ mit $t(a)$ als Ausführungsdauer der Strategie a und r_i als zusätzlicher Strafe
- $\mathcal{Z}$: $Z_{check} = \{z_{c+}, z_{c-}\}$; $Z_{such} = \{z_{s+};\ z_{s-}\}$; $Z_{next} = \{z_{n+}, z_{n-}\}$; $Z_{remove} = \{z_{r+}, z_{r-}\}$
- $\mathcal{O}$: Die Tabellen 5.5 bis 5.8 zeigen die Beobachtungswahrscheinlichkeiten getrennt nach Aktionen.

Innerhalb der Kontrolle des POMDP liegen nur die Zustände s_1 und s_n. Alle Strategietypen sind abhängig von der Applikation und den Zuweisungen des Bedieners als Aktionen nutzbar. Da die Planung der Aufrufreihenfolge mittels POMDP eine schnelle Störungsbehandlung bevorzugen soll, wird die benötigte Taktzeit als Bewertungsmaß eingesetzt. Jeder Strategieaufruf wird daher mit einem Wert in Höhe der erforderlichen Ausführdauer bestraft. Zusätzlich wird auch der unerwünschte Aufruf einer Holstrategie bei vorhandenem Objekt oder das generelle Entfernen eines Bauteils bestraft. Obwohl letzteres die einzig sinnvolle Behandlung darstellt, wenn die Lokalisierung eines vorhandenen Objekts unmöglich ist, kann eine Strafe sinnvoll sein. Werden Bauteile z. B. per

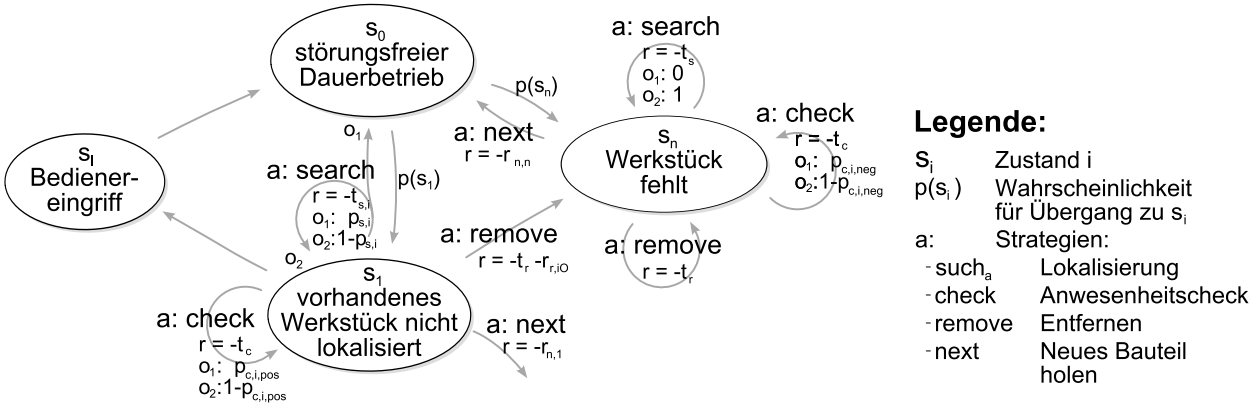


Abb. 5.6.: Reduziertes Zustandsmodell für die Zuordnung des Erfahrungswissens

Werkstückträger auf einem Fließband zugeführt, kann das Entfernen durch Senden eines digitalen Signals erfolgen. Da die Ausführungsdauer hierfür sehr gering ist, würden die zeitaufwändigeren Lokalisierungsstrategien nie aufgerufen. Abhängig von den Randbedingungen der Applikation kann das Systemverhalten über die zusätzlichen Kosten zwischen einer möglichst schnellen Behandlung und möglichst wenig separat abgelegten Werkstücken priorisiert werden.

Die Zustandsübergänge sind bei diesem Modell stark beschränkt. Wurde ein Objekt bisher nicht lokalisiert, ist ein störungsfreier Betrieb zum einen durch eine erfolgreiche Lageerkennung erreichbar. Zum anderen kann das Objekt entfernt werden. Dieser Zustand kann nur durch Holen eines neuen Bauteils verlassen werden. Die Tabellen 5.1 bis 5.4 zeigen die lernbaren Wahrscheinlichkeiten p_i der Transitionsübergänge. Zu beachten ist in Tabelle 5.4, dass beim Ausführen von a_{next} in Zustand s_1 ein Übergang zu s_0 stattfindet. Obwohl diese Aktion wegen möglicher resultierender Ausfälle nicht ausgeführt werden sollte, ist der Programmablauf anschließend fortführbar, da die aktuelle Störung als behoben gilt.

Die in den Tabellen 5.5 bis 5.8 dargestellten lernbaren Beobachtungswahrscheinlichkeiten ergeben sich aus der Erfolgsrate der Strategieausführung, wenn ein Objekt vorhanden ist. Bei der Anwesenheitsprüfung könnten auch fehlerhafte Pseudo-Erkennungen ohne Objekt möglich sein. Da es bei den Holstrategien keine Rückmeldung über die Anwesenheit eines Objekts gibt, ist die Beobachtung hiervon unabhängig.

Wie beschrieben, können die Transitions- und Beobachtungswahrscheinlichkeiten ebenso wie die mittleren Ausführungsdauern als Erfahrungswissen aus vorhergehenden Ergebnissen der Störungsbehandlung ermittelt werden. Das für den POMDP entworfene Zustandsmodell erfüllt somit die Anforderungen an eine autonome Erlernbarkeit der statistischen Daten.

		s'		
		s_0	s_1	s_n
s	s_1	0	1	0
	s_n	0	0	1

Tab. 5.1.: $T(s, a_{check}, s')$

		s'		
		s_0	s_1	s_n
s	s_1	$p_{s,i}$	$1-p_{s,i}$	0
	s_n	0	0	1

Tab. 5.2.: $T(s, a_{search}, s')$

		s'		
		s_0	s_1	s_n
s	s_1	0	$1-p_r$	p_r
	s_n	0	0	1

Tab. 5.3.: $T(s, a_{remove}, s')$

		s'		
		s_0	s_1	s_n
s	s_1	p_n	$1-p_n$	0
	s_n	p_n	0	$1-p_n$

Tab. 5.4.: $T(s, a_{next}, s')$

		z	
		z_{t+}	z_{t-}
s	s_1	$p_{c,i,pos}$	$1-p_{c,i,pos}$
	s_n	$p_{c,i,neg}$	$1-p_{c,i,neg}$

Tab. 5.5.: $O(s, a_{check}, z')$

		z	
		z_{s+}	z_{s-}
s_i	s_1	$p_{s,i}$	$1-p_{s,i}$
	s_n	0	1

Tab. 5.6.: $O(s, a_{search}, z')$

		z	
		z_{r+}	z_{r-}
s_i	s_1	p_r	$1-p_r$
	s_n	0	1

Tab. 5.7.: $O(s, a_{remove}, z')$

		z	
		z_{n+}	z_{n-}
s_i	s_1	p_n	$1-p_n$
	s_n	p_n	$1-p_n$

Tab. 5.8.: $O(s, a_{next}, z')$

Für die Berechnung der Policy ist ein endlicher Horizont anzunehmen, da die Störungsbehandlung möglichst schnell erfolgen soll und die Menge der Aktionen begrenzt ist. Da zufällige Störungen ausgeschlossen werden, ist jede Strategie nur einmalig auszuführen. Eine Ausnahme besteht bei Anwesenheitsprüfung mittels Zugreifens. Da das Bauteil bei der Ausführung ggf. bewegt wird, können infolge der veränderten Lage wieder alle Lokalisierungsstrategien erfolgreich sein. Bei Auftreten einer nicht autonom behebbaren Störung könnte das Zurücksetzen der Wahrscheinlichkeiten zu einer endlosen, wechselseitigen Ausführung von erfolgreichen Anwesenheitsprüfungen und fehlgeschlagenen Lageerkennungen führen. Da dies die Signalisierung eines erforderlichen Bedienereingriff verhindert, wird ein Discount-Faktor γ (mit $\gamma < 1$) zur schrittweisen Anpassung der Erfolgschancen eingeführt. Über $p(s_0|s_i, a_{such,j}, n_{check}) = \gamma^{n_{check}} \cdot p(s_0|s_i, a_{such,j}, 0)$ sinkt die Erfolgswahrscheinlichkeit einer Lokalisierungsstrategie abhängig von der Anzahl n_{check} der bereits ausgeführten Anwesenheitschecks. Ein genereller Discount nach jeder Aktion wird nicht eingesetzt, da der Nutzen späterer Lösungen durch die aufsummierten Zeitstrafen bereits reduziert wird.

Aus den Randbedingungen des Anwendungsgebiets ergeben sich verschiedene Eigenschaften für gültige Policies. Da der POMDP nur zur Behandlung von Störungen eingesetzt wird, ist die Planung nach einer erfolgreichen Suchstrategie durch den Übergang in den störungsfreien Betrieb s_0 beendet. Innerhalb der Policy ergeben sich bei dieser Beobachtung keine Folgeknoten. Ähnliches gilt auch nach der Beschaffung neuer Bauteile. Sind Strategien zur Entfernung bzw. zur Beschaffung von Bauteilen vorhanden, dürfen die verschiedenen Handlungsabfolgen einer Policy nicht auf Strategien zum Prüfen oder Entfernen sowie auf eine erfolglose Lokalisierung enden, damit Störungen stets behoben werden. Die Abbildung 5.7 verdeutlicht die Struktur entsprechender Policies. Um die Berechnung ungültiger Lösungen zu vermeiden, werden die Anforderungen zur Kombination der Aktionen bereits innerhalb der Value Iteration berücksichtigt.

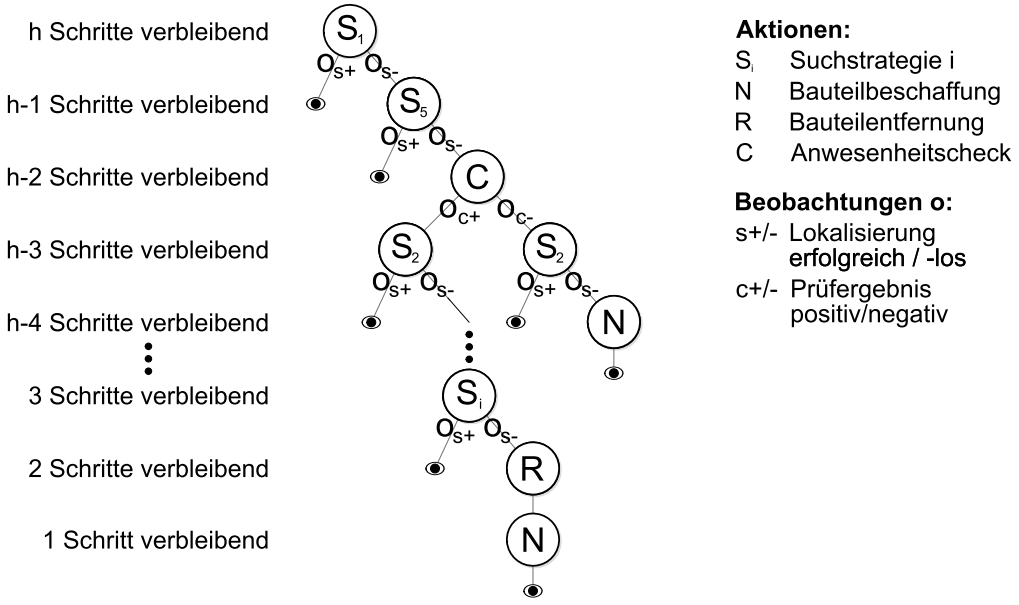


Abb. 5.7.: Typische Struktur einer Policy zur Behandlung von Lokalisierungsstörungen

5.4. Parameterarme Strategien zur Störungsbehandlung

Die autonome Behandlung der auftretenden Störungen erfolgt über Strategien, die zuvor vom Bediener ausgewählt und an die Applikation angepasst wurden. Wie in Abschnitt 2.3.3 analysiert, betreffen die im Automatikbetrieb auftretenden Störungen die Objektlageerkennung sowie die Bauteilmanipulation. Um für einen unterbrechungsfreien Betrieb alle relevanten Fehlerfälle abzudecken und gleichzeitig eine schnelle Störungsbehandlung zu ermöglichen, enthält die Datenbank Strategien aus den folgenden vier Klassen:

- Kamerabasierte Lokalisierungsstrategien
- Strategien zur Bauteilbeschaffung
- Strategien zur Bauteilentfernung
- Strategien zur Anwesenheitsprüfung

Die ersten drei Klassen dienen zur direkten Behebung von Störungen. Die Lokalisierungsstrategien wiederholen die Objektlageerkennung unter variierten Bedingungen, um vorhandener Objekte mit erkennbaren Merkmalen lokalisieren zu können. Fehlt ein Bauteil oder wurde dies aus dem Greifer verloren, können neue Werkstücke über geeignete Strategien beschafft werden. Hat ein Bauteil durch Verrutschen eine unbekannte Lage im Greifer, kann das System durch separates Ablegen des Werkstücke wieder in einen störungsfreien Zustand überführt werden. Eine ähnliche Behandlung erfolgt in Kombination mit einem vorherigen undefinierten Abgriff, wenn eine Werkstücklokalisierung z.B. wegen verdeckter Merkmale nicht möglich ist. Die Anwesenheitsprüfung hingegen kann Störungen nicht beheben und dient zur Beschleunigung der Behandlung, indem Informationen über den aktuellen Fehlerfall gewonnen werden.

Alle Strategien sind so implementiert, dass nur wenige Parameter einzustellen sind, um im Rahmen der übrigen MMS schnell und ohne Expertenwissen einsetzbar zu sein. Strategien, die innerhalb der jeweiligen Aktion nicht anwendbar sind, werden dem Bediener nicht angezeigt. Der geringe Inbetriebnahmeaufwand soll die Bereitschaft der Bediener erhöhen, bereits beim Einrichten der Applikation erste Strategien zuzuweisen, um verschiedene Störungen von Beginn an zu vermeiden.

5.4.1. Kamerabasierte Lokalisierungsstrategien

Die Lokalisierungsstrategien eignen sich zur Behandlung von Störungen, wenn das Objekt vorhanden und korrekt orientiert ist, die Lageerkennung aber wegen eines anderen Fehlers nicht erfolgreich war. Durch die Variation einzelner Randbedingungen, werden die Erfolgschancen vor einer erneuten Lokalisierung erhöht. Im vorliegenden Fall werden kamerabasierte Lokalisierungsverfahren betrachtet, bei denen die Sensoren am Roboter befestigt sind. Die Datenbank kann jedoch auch um Strategien erweitert werden, die auf anderen Randbedingungen beruhen.

Position variieren

Bei Ausführung dieser Strategie wird die Kamera vor der erneuten Lokalisierung parallel zur Bildebene um eine vom Bediener festzulegende Distanz verfahren. Durch das verschobene Bildfeld bzw. eine veränderte Lage der Kamera zum Objekt können Störungen durch Verdeckung, Lichteffekte oder eine unerwartet abweichende Bauteilposition behoben werden. Für eine schnelle Parametrierung ist nur die Schrittweite anzugeben, wodurch der Störungsbehandlung automatisch acht Einzelstrategien für die Bewegung entlang der X- und Y-Achsen der Kamera sowie der Diagonalen hinzugefügt werden (siehe Abbildung 5.8). Die Zielposition der Bewegung ergibt sich entsprechend der in Abschnitt 4.5.2 beschriebenen Translation bei der kamerabasierten Robotersteuerung.

Orientierung variieren

Diese Strategie verkippt die Kamera vor der Lokalisierung um einen festzulegenden Winkel. Der Drehpunkt liegt auf der Z-Achse der Kamera im mittleren Tiefenschärfebereich, so dass sich der Blickwinkel zum Objekt ändert, die Position im Bild aber nahezu konstant bleibt. Diese Variation kann Störungen infolge von Lichteffekten, Verdeckungen oder unerwartet starkem Verkippens des Objekts beheben. Wie bei *Position variieren* werden nach Auswahl dieses Strategietyps und Angabe des gewünschten Neigungswinkels automatisch acht Einzelstrategien hinzugefügt (siehe Abbildung 5.9). Eine Anwendung der Strategie ist bei der kantenbasierten Lageerkennung wegen der erforderlichen parallelen Kameraausrichtung nicht möglich.

Bildaufnahmeparameter variieren

Mit dieser Strategie werden die Belichtungszeit und der Verstärkungsfaktor vor einer erneuten Bildaufnahme variiert, um eine durch Schatten oder Reflexionen veränderte Helligkeit in relevanten Bildbereichen zu kompensieren. Der Bediener kann für beide Parameter jeweils einen nach oben und unten abweichenden Wert angeben. Abhängig von der Anzahl der angegebenen Werte werden der Störungsbehandlung durch die entstehenden Permutationsmöglichkeiten bis zu acht Einzelstrategien hinzugefügt.

Objektmodell variieren

Statt einzelne Parameter zu variieren, kann die Lokalisierung bei dieser Strategie mit einem vollständig anderen Objektmodell wiederholt werden. Die Anwendung eignet sich bei Auftreten star-

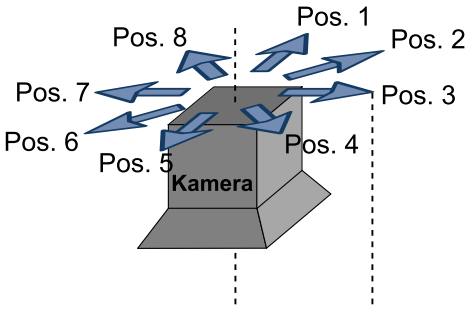


Abb. 5.8.: Mögliche Bewegungen der Stratgie *Variiere Position*

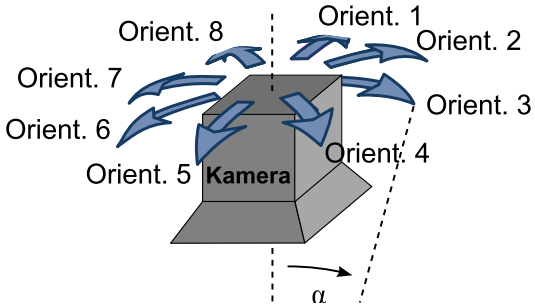


Abb. 5.9.: Mögliche Bewegungen der Stratgie *Variiere Orientierung*

ker Lichteffekte oder abweichender Objektansichten. Bei der Inbetriebnahme muss die Objektlage durch Variation der Kameraposition oder manuelle Abschattung einmalig mit dem ursprünglichen Modell erfasst werden, um die definierte Manipulationsbewegung auf das neue Modell übertragen zu können.

Unverändert wiederholen
Bei dieser Strategie wird die Lokalisierung ohne Variation der Randbedingungen wiederholt. Dies kann speziell nach einer Anwesenheitsprüfung helfen, bei der das Objekt bewegt wurde. Zusätzlich sind Verzögerungen bei der Bauteilzuführung ausgleichbar. Die Strategie benötigt keine Parameter.

Referenz-Suchposition verwenden
Bei der landmarkenbasierten Lageerkennung wird die Kamera unabhängig von der definierten Referenz-Suchposition jeweils anhand der zuletzt erkannten Objektlage ausgerichtet, da Paletten und Prozessnester zumeist länger an einem Ort liegen. Dies kann jedoch zu Störungen führen, wenn die Landmarken z.B. infolge eines manuellen Palettenwechsels stark von der vorherigen Lage abweichen. Bei Ausführung dieser Strategie erfolgt die Wiederholung der Lokalisierung von der Referenz-Suchposition. Die Anwendung ist auf die landmarkenbasierte Lokalisierung beschränkt. Eine Parametrierung ist nicht erforderlich.

Frei definierbarer Handlungsablauf
Zusätzlich zu den speziellen, parameterarmen Strategien kann der Bediener aus den Aktionen des Arbeitsplans frei definierbare Handlungsstränge erstellen, die vor einer Wiederholung der Lokalisierung ausgeführt werden. Um die Komplexität zu begrenzen und unübersichtliche Kaskaden zu vermeiden, können in dem Ablauf keine bedingten Verzweigungen und keine Störungsbehandlung integriert werden. Die Strategie eignet sich zum Beseitigen von Störungen durch gezielte Roboterbewegungen oder Manipulationen, indem Bauteile z.B. in eine definierte Lage gebracht werden. Wird das zu lokalisierende Objekt bei der Strategieausführung bewegt, ist dies vom Bediener bei der Parametrierung anzugeben. Das Einrichten der Aktionen erfolgt wie in Kapitel 4 beschrieben.

5.4.2. Strategien zur Bauteilbeschaffung

Wird das Fehlen eines Werkstücks im Ablauf detektiert, muss diese Störung zumeist aktiv behoben werden. Anderenfalls können unvollständig montierte Baugruppen oder eine Unterbrechung des Betriebs durch nicht ausgelöste Anwesenheitssensoren die Folge sein. Um neue Bauteile als Ersatz zu beschaffen, können diese zum einen von einer separaten Zuführung oder aus einem vorherigen Abschnitt der Applikation entnommen werden. Zum anderen kann der aktuelle Bearbeitungszyklus in einem Prozess auch abgebrochen und neu gestartet werden, um über eine reguläre Ausführung wieder in einen zulässigen Systemzustand zu gelangen.

Sprung innerhalb des Arbeitsplans
Mit dieser Strategie kann die aktuelle Ausführung abgebrochen und über eine vom Bediener anzugebende Transition mit einer beliebigen Aktion des Arbeitsplans fortgesetzt werden. Dies erlaubt eine gezielte Wiederholung bereits vorhandener Arbeitsschritte, mit denen das fehlende Bauteil zuvor gegriffen oder zum Prozess zugeführt wurde. Zusätzlich zum gewünschten Sprungziel kann der Bediener eine kollisionsfreie Bewegung zur Annäherung an die entsprechende Aktion definieren.

Flexibler Handlungsablauf mit Sprungmöglichkeit
Über frei definierbare Handlungsabläufe aus den Aktionen des Arbeitsplans kann z.B. auf die automatische Zuführung des nächsten Teils gewartet oder ein neues Bauteile aus hierfür bereitgestellten Werkstückspeichern entnommen werden. Über eine Sprungmöglichkeit kann die Ausführung anschließend ebenfalls mit jeder beliebigen Aktion des Arbeitsplans fortgesetzt werden. Dadurch eignet sich diese Strategie auch für Vor- oder Nachbereitende Handlungen vor einem Sprung, wie das Entfernen von anderen Werkstücken aus einer Maschine oder das Verstellen von Schiebern und Türen. Die Inbetriebnahme der Aktionen erfolgt wie in Kapitel 4 beschrieben. Bedingte Verzeigungen und Störungsbehandlungen sind nicht möglich.

Palettenposition überspringen
Wird das Fehlen eines Bauteils beim Abgriff aus einer Palette detektiert, kann der interne Positionszähler um einen Schritt inkrementiert werden, um das aktuelle Nest zu überspringen und das nächste Werkstück aufzunehmen. Eine Parametrierung dieser Strategie ist nicht nötig.

5.4.3. Strategien zur Bauteilentfernung

Mit dieser Strategieklasse lassen sich Werkstücke, die nicht entsprechend den Anforderungen verarbeitet werden können, zur Fortführung des Ablaufs aus dem Prozess entfernt. Bereits aufgenommene, aber im Greifer undefiniert verrutschte Bauteile lassen sich so separat abgelegen. Zusätzlich können Werkstücke nach erfolglosen Lokalisierungsversuchen mit dem Greifer oder zusätzlichen Werkzeugen entfernt werden.

Flexibler Handlungsablauf mit Sprungmöglichkeit
Über frei definierbare Aktionsketten können Werkstücken durch zusätzliche Bewegungen oder die Ansteuerung einer Maschine aus dem eigentlichen Prozess entfernt werden. Werkstücke mit nur wenig schwankender Lage können undefiniert gegriffen und separaten abgelegt werden. Bei größeren Toleranzen kann ein freiliegendes Bauteil mit Hilfswerkzeugen entfernt werden. In Abbildung 5.10 ist dargestellt, wie ein Bauteil mittels eines Rakels von einer Ablagefläche geschoben wird. Das Werkzeug ist hierfür an definierter Lage bereitgestellt und zentriert das Objekt beim Schieben über eine V-Form.

Verzweigungen und integrierte Störungsbehandlungen sind nicht verwendbar. Nach dem Entfernen des Bauteils folgt stets eine Strategie zur Bauteilbeschaffung. Diese ist separat zu erstellen, damit bei fehlenden Werkstücken eine getrennte Ausführung möglich ist.

(a) (b)

Abb. 5.10.: Bauteil mit Hilfe eines Rakels entfernen: a) Rakel greifen, b) Bauteil mit Rakel aus Ablagebereich entfernen

5.4.4. Strategien zur Anwesenheitsprüfung

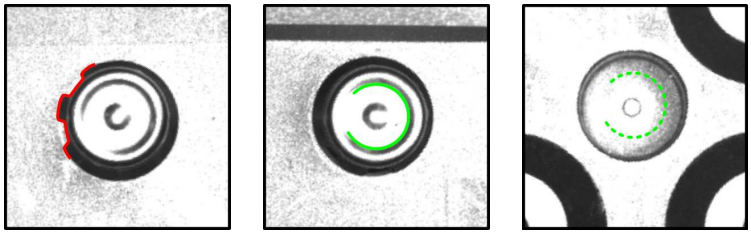
Die Anwesenheitsprüfungen dienen nur indirekt der Störungsbehandlung, da sie weder eine ausreichende Lokalisierung der Bauteile ermöglichen noch zur Lösung von Manipulationsproblemen beitragen. Die Informationen über die Präsenz von Werkstücken verbessern jedoch die Einschätzung der potentiellen Fehlerart und können so die Störungsbehandlung beschleunigen.

Alternative Objektmerkmale prüfen

Bei dieser Strategie wird die Anwesenheit des Objekts über andere Merkmale geprüft als bei der eigentlichen Lokalisierung. Im Gegensatz zur Suchstrategie *Objektmodell variieren* können z.B. auch symmetrische Merkmale verwendet werden, die keinen Aufschluss über die gesuchte Drehlage eines Werkstücks geben, wie in Abbildung 5.11 gezeigt. Die Ausführungsdauer dieser Strategie ist sehr kurz. Für die Verwendung ist eine einzelne Objekterkennung entsprechend der Beschreibung in Abschnitt 4.4.3 einzurichten.

Blinder Greiftest

Die Anwesenheit des Bauteils kann auch über den Greifer geprüft werden. Hierzu wird der Anrückpfad relativ zur aktuellen Suchposition abgefahren. Die Zielposition entspricht damit der gesuchten



(a) (b) (c)

Abb. 5.11.: Check der Bauteilanwesenheit über Lokalisierung mit alternativen Merkmalen: a) Bauteil mit originalem, fehleranfälligem Merkmal, b) alternatives, symmetrisches Merkmal ohne Erkennung der Drehlage, c) eine Pseudo-Lokalisierung ist mit dem Alternativmerkmal unwahrscheinlich

(a) **(b)** **(c)**

Abb. 5.12.: Greiftest nach vorheriger Bauteilzentrierung mit einem Hilfsmittel: a) und b) Zentrierbewegung hin und zurück, c) Greiftest an zentrierter Position

Greifposition ohne Ausgleich einer möglichen Lageabweichung. Schließen sich die Finger beim kraftgeregelten Greifen nicht vollständig, wird die Präsenz eines Bauteils vermutet. Bei der Ausführung der Prüfung tritt als Nebeneffekt oft eine Veränderung der Werkstücklage auf. Durch die Bewegung der Greiferfinger können Bauteile häufig zentriert und nachfolgende Lokalisierungsversuche begünstigt werden.

Da die Ausführung dieser Anwesenheitsprüfung auf dem bereits definierten Bauteilabgriff beruht, ist zur Anwendung keine zusätzliche Parametrierung erforderlich. Der Bediener kann aber den beim Anrücken verwendeten Greiferöffnungswinkel vergrößern, da eine schwankende Werkstücklage ohne Lokalisierung sonst zu einer Kollision führen könnte. Die Anwendung dieser Strategie ist auf das Greifen von Bauteilen beschränkt, deren Lage z.B. durch Prozessnester so weit begrenzt ist, dass ein blinder Testgriff kollisionsfrei ausgeführt werden kann.

Anwesenheitsprüfung mittels flexiblen Handlungsablaufs

Zur Ausführung komplexerer Überprüfungsbewegungen, z.B. mittels vorheriger aktiver Zentrierung eines Bauteils durch Hilfsmittel, lässt sich auch in dieser Strategieklasse ein frei definierbarer Handlungsablauf erstellen (siehe Abbildung 5.12). Das Verbot von Verzweigungen und Fehlerbehandlungen gilt wie zuvor beschrieben. Um ein Testergebnis zu erhalten, wird die vollständig störungsfreie Ausführung als positives Resultat gewertet. Fehlgeschlagene Lokalisierungen und fehlende Objekte werden als erfolglose Prüfung ausgelegt. Ein undefiniert gegriffenes Werkstück wird nicht als Störung gewertet, da ein Testgriff keine definierte Öffnungsweite besitzt. Das Einrichten der Aktionen erfolgt wie in Kapitel 4 beschrieben. Zusätzlich muss der Bediener angeben, ob die Bauteillage durch die Anwesenheitsprüfung beeinflusst wird, da dies die Erfolgswahrscheinlichkeiten bereits ausgeführter Suchstrategien zurücksetzen kann.

5.5. Lernen von statistischem Erfahrungswissen

Für die Berechnung einer optimalen Policy sind verschiedene statistische Daten erforderlich. Im vorliegenden Fall betrifft dies die Wahrscheinlichkeiten für das Auftreten der Zustandsübergänge und Beobachtungen, T und O. Zusätzlich werden die Höhe der Strafen in R und der Startbelief b_0

benötigt. Da ein Bediener die entsprechenden Wahrscheinlichkeiten nicht beurteilen kann, wird an dieser Stelle erläutert, wie die Daten als Erfahrungswissen aus vorangegangenen Störungen autonom gelernt werden können. Durch die geringen Stückzahlen in der KSM wird das Robotersystem häufig für neue Aufgaben umprogrammiert. Da zu Beginn einer neuen Applikation noch kein Erfahrungswissen vorliegt, werden in Abschnitt 5.7 Methoden beschrieben, wie mit dem vorgestellten Ansatz dennoch eine erste geeignete Strategiesequenz berechnet und die Initialisierung des Erfahrungswissens beschleunigt werden kann.

Die in R enthaltenen Strafen werden für das Aufrufen von Aktionen vergeben. Die Höhe ergibt sich für die jeweilige Strategie aus der Ausführungsdauer, die aus den gespeicherten Ergebnissen aller bisherigen Aufrufe gemittelt wird. Als Erfahrungswissen wird die Anzahl der Aufrufe einer Strategie sowie die Summe der Ausführzeiten für die Berechnung gespeichert. Ändert der Bediener die Parametrierung einer Strategie werden diese Werte zurückgesetzt. Eventuelle Zusatzkosten für die Aktionen zum Entfernen bzw. Holen eines Bauteils sind applikationsabhängig und werden vom Bediener ausgewählt (siehe Abschnitt 5.8).

Der Startbelief b_0 entspricht der Wahrscheinlichkeit, dass bei Eintreten einer Störung ein Bauteil vorhanden ist. Diese Angabe wird als Mittelwert aus der Anzahl der vorhandenen und fehlenden Objekte aus allen vorherigen Störungen gelernt. Die Anwesenheit eines Bauteils ist durch eine erfolgreiche Ausführung von Lokalisierung-, Anwesenheitsprüf- oder Entfernungsstrategien erkennbar. Die Vermutung über das Fehlen eines Objekts kann hingegen häufig nicht sicher bestätigt werden. Da das System keine automatische Rückmeldung über Fehlentscheidungen erhält, kann der Bediener diese ggf. über die MMS meldet, indem er z.B. bei einem Wechsel der Paletten die Anzahl übrig gebliebenen Bauteile eingibt.

Die Wahrscheinlichkeiten für Zustandsübergänge T sowie für das Auftreten einer Beobachtung O hängen zum Teil stark zusammen. Bei den Strategien zum Entfernen eines Werkstücks werden alle korrekten Ausführungen erkannt, so dass die Erfolgschance für einen Übergang zum fehlenden Objekt auch der Beobachtungswahrscheinlichkeit entspricht. Der Wert wird aus vorangegangenen Ausführungen gelernt, indem die Anzahl an jeweiligen Strategieaufrufen sowie der zugehörigen Erfolge gespeichert werden. Bei den Strategien zur Bauteilbeschaffung kann die Erfolgsrate nicht sicher beobachtet werden. Gleichzeitig wäre das Ergebnis auch nicht relevant, da ein fehlendes Objekt unabhängig von den Erfolgsaussichten nicht anders behandelbar ist. Beide Wahrscheinlichkeiten werden daher als sicher (Wahrscheinlichkeit $p = 1.0$) bewertet und nicht gelernt. Durch den Aufruf der Strategien zur Anwesenheitsprüfung kann kein Zustandsübergang erfolgen. Unabhängig von der Anwesenheit eines Objekts kann jedoch sowohl eine erfolgreiche als auch eine fehlgeschlagene Ausführung beobachtet werden. So kann ein blinder Prüfgriff ein vorhandenes Bauteil verfehlen oder eine Prüflokalisierung ein Pseudo-Objekt erkennen, das nicht vorhanden ist. Die Wahrscheinlichkeiten sind aus vorherigen Ausführungen lernbar, indem das Prüfergebnis nach Behandlung der Störung anhand der geschätzten Objektanwesenheit bewertet und eingeordnet wird. Das gesammelte Erfahrungswissen wird für jede zugewiesene Strategie zum Entfernen, Beschaf-

fen und Prüfen der Anwesenheit jeweils in einem Datensatz gehalten und gilt für alle Aufrufe und Berechnungen innerhalb dieser Aktion.

Auch bei den Lokalisierungsstrategien sind die statistischen Daten als Erfahrungswissen lernbar. Das Ergebnis jeder Ausführung wird nach der Behandlung anhand der zuletzt geschätzten Objektanwesenheit gewertet. Die Beobachtungs- und Übergangswahrscheinlichkeiten stimmen jeweils überein, da die Störungsbehandlung bei einer erfolgreichen Lokalisierung unabhängig von deren Korrektheit abgeschlossen wird. Das entsprechende Erfahrungswissen könnte für jede Lokalisierungsstrategie ebenfalls in einem eigenen Datenobjekt gespeichert und innerhalb der Aktion für alle Berechnungen verwendet werden. Es hat sich jedoch gezeigt, dass die Erfolgswahrscheinlichkeiten der Lokalisierungsstrategien nicht unabhängig voneinander sind. So unterscheiden sich die Erfolgschancen der übrigen Strategien vor und nach der Ausführung einer Behandlungsmethode zum Teil deutlich. Neben den vorhandenen Abhängigkeiten zwischen den verschiedenen Lokalisierungsstrategien hat vor allem die Ausführung von Manipulationen, die das Objekt bewegen, einen großen Einfluss auf die Erfolgrate der nachfolgenden Behandlungsmöglichkeiten. Die mehrfache Ausführung derselben Lageerkennungsvariante sollte bei Vernachlässigung zufälliger Effekte zudem keinen Erfolg bringen. Würde das Erfahrungswissen innerhalb der Aktion nur jeweils pro unterschiedliche Lageerkennungsstrategie erfasst, würde viel Potenzial zur Taktzeitoptimierung ungenutzt bleiben.

Um die bedingten Abhängigkeiten bei der Berechnung der optimalen Policy zu berücksichtigen, wird das statistische Erfahrungswissen für jede Kombination von zuvor ausgeführten Strategien einzeln erfasst. Entsprechend ergeben sich für jeden unterschiedlichen Belief State bei der Berechnung der optimalen Policy gesonderte Transition- und Beobachtungswahrscheinlichkeiten für die einzelnen Lokalisierungsstrategien.

5.6. Optimierung der Berechnungsdauer

Sobald im Betrieb eine Störung behandelt wurde, führen die zusätzlichen Informationen aus der Strategieausführung zu einer Veränderung des Erfahrungswissens. Dieser Effekt ist umso größer, je weniger Daten zur Verfügung stehen. Damit stets die statistisch beste Behandlung ermöglicht wird, ist die optimale Policy zur Laufzeit nach jedem Informationszuwachs erneut zu berechnen. Da zwei Störungen im Extremfall in aufeinander folgenden Zyklen der Applikation auftreten können, muss die Berechnung innerhalb der Zykluszeit von wenigen Sekunden erfolgen.

Die Berechnung der optimalen Policy mittels Value Iteration ist zumeist sehr rechenaufwändig. Die Menge der Belief States steigt abhängig von der Zahl an Aktionen durch deren Kombination sehr stark an. Die Tabelle 5.9 verdeutlicht dies für eine Behandlung mit einer Prüf- und n Suchstrategien sowie je einer Maßnahme zur Entfernung und Beschaffung eines Bauteils. Zusätzlich ist die Menge an Belief-states bei Einsatz von sechszehn Lokalisierungsvarianten angegeben, die z.B. durch „Position variieren“ und „Orientierung variieren“ anhand der jeweils acht Richtungen entstehen. Die Anzahl der unterschiedlichen Handlungssequenzen in jedem Zeitschritt ergibt sich

Zeitschritt h	Anzahl der Policy-Knoten
1	1
2	$1+1+16\cdot 1+1^2=19$
3	$1+1+16\cdot 19+19^2=667$
4	$1+1+16\cdot 667+667^2=455563$
...	
i	$1+1+n\cdot b_{i-1}+b_{i-1}^2$

Tab. 5.9.: Wachstum der Anzahl an Policy-Knoten abhängig vom Horizont

zum einen aus den beiden Strategien zum Beschaffen und Entfernen der Werkstücke. Weiterhin kommt eine quadratisch wachsende Menge an Möglichkeiten durch die beiden Beobachtungsergebnisse der Prüfstrategie hinzu. Diese ergibt sich durch Kombination aller Handlungsfolgen aus dem vorherigen Schritt, die zusätzlich auch mit allen Lokalisierungsstrategien kombiniert werden.

Zusätzlich kann die optimalen Policy wegen der bedingten Wahrscheinlichkeiten der Lokalisierungsstrategien nicht direkt wie in Abschnitt 5.1 beschrieben berechnet werden. Die Erfolgswahrscheinlichkeiten der Lageerkennungsvarianten hängen jeweils von den bei der Ausführung zeitlich vorhergehenden Behandlungsversuchen ab. Bei der Value Iteration wird die Sequenz von Aktionen bezogen auf deren Verwendung jedoch in umgekehrter Reihenfolge berechnet. Entsprechend stehen die zuvor ausgeführten Strategien noch nicht fest. Um diese Methode dennoch anwenden zu können, müssen bei der Berechnung alle möglichen Vorgängerkombinationen berücksichtigen werden. Die Abbildung 5.13 zeigt wie die für den Horizont 1 zuerst gewählte Strategie mit Index 1 abhängig von den anschließend kombinierten Strategien bis zu sieben unterschiedliche Erfolgswahrscheinlichkeiten erhalten kann. Es wird angenommen, dass eine unterschiedliche Reihenfolge derselben Vorgänger keinen Einfluss hat, da die Bildszene nicht verändert wird. Die Variantenzahl steigt in Abhängigkeit von der Menge zugewiesener Strategien sehr stark an. Zählt man zu den bedingten auch die eine unbedingte Wahrscheinlichkeit hinzu, so können sich bei n Strategien durch die Kombination der Vorgänger bis zu 2^{n-1} verschiedene Wahrscheinlichkeiten für den Erfolg einer Strategieausführung ergeben. Diese sind bei der Berechnung der optimalen Policy alle zu berücksichtigen.

Als zusätzlicher Effekt der bedingten Abhängigkeiten kann die Menge an Policies nach jedem Zeitschritt nicht so stark durch Pruning verringert werden. Für einen sinnvollen Vergleich mehrerer

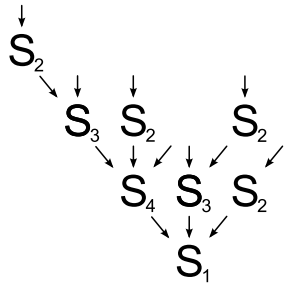


Abb. 5.13.: Möglichkeiten für bedingte Abhängigkeiten bei vier Strategien. Bedeutung S_i: Lokalisierungsstrategie mit Index i

Policies dürfen die Handlungssequenzen keinen unterschiedlichen Abhängigkeiten für die weitere Kombinierbarkeit mit den übrigen Lokalisierungsstrategien unterliegen. Durch die hohe Anzahl an Aktionen sind sehr viele unterschiedliche Zustände möglich, so dass nach dem Pruning zumeist sehr viele Lösungen übrig bleiben.

5.6.1. Vorselektion über einen Wahrscheinlichkeitsbaum

Mit der klassischen Value Iteration ist die optimale Policy bei Zuweisung mehrerer Strategien durch die resultierende Knotenmenge nicht innerhalb einer ausreichenden Zeit ermittelbar. Um den Rechenaufwand zu reduzieren, wird die Lösung des POMDP approximiert, indem die vielversprechendsten Lokalisierungsabfolgen vorausgewählt werden. Durch die verringerte Aktionszahl werden die zu überprüfenden Kombinationsmöglichkeiten deutlich reduziert.

Für die Vorauswahl werden die bedingten Erfolgswahrscheinlichkeiten der Lageerkennungsstrategien nicht in Matrizen für die einzelnen Policy-Knoten sondern in einem Wahrscheinlichkeitsbaum gespeichert (siehe Abbildung 5.14). Während die Kanten mit den bedingten Wahrscheinlichkeiten gewichtet sind, repräsentieren die Knoten die einzelnen Lokalisierungsstrategien und bilden entlang der Pfade die Kombinationsmöglichkeiten ab. Die Tiefe i gibt jeweils die Position innerhalb der Sequenz an. Da die wiederholte Ausführung einer Strategie zu keinem unterschiedlichen Ergebnis führen sollte, enthalten die Pfade keine Lokalisierungsvariante mehrfach. Zusätzlich werden Pfade mit übereinstimmender Liste an Vorgängern zusammengeführt (in Abbildung 5.14 blau hervorgehoben), da die Reihenfolge der vorherigen Knoten keinen Einfluss auf die Erfolgschance einer nachfolgenden Lageerkennung hat. Dies reduziert die Knotenmenge im Baum und ermöglicht eine einheitliche Erfassung des Erfahrungswissens bei denselben Kombinationen von Strategie und Vergangenheit.

Innerhalb des Wahrscheinlichkeitsbaums werden zur Berechnung der Policy nur die Knoten gesucht, die im Rahmen ihrer Sequenz die höchste Erfolgswahrscheinlichkeit innerhalb der benötigten Zeit bieten. Hierzu werden alle Pfade ausgehend vom Wurzelknoten betrachtet und an jedem Knoten die Erfolgswahrscheinlichkeit sowie die Ausführungsdauer der zum Knoten führenden Handlungssequenz berechnet (siehe Abbildung 5.15). Ist die ermittelte Erfolgsrate innerhalb der Dauer höher als bei den bisher untersuchten Knoten, wird die zum Kandidaten führende Sequenz

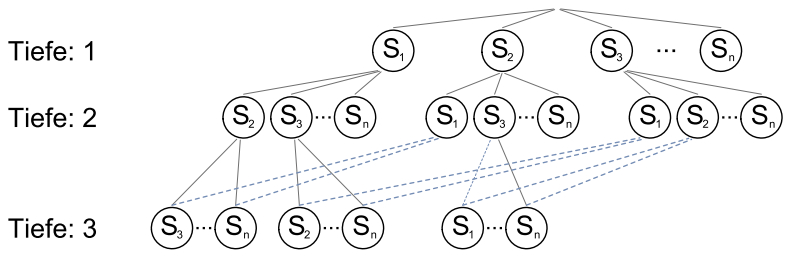


Abb. 5.14.: Speicherung des Erfahrungswissens der Lokalisierungsstrategien in Wahrscheinlichkeitsbäumen

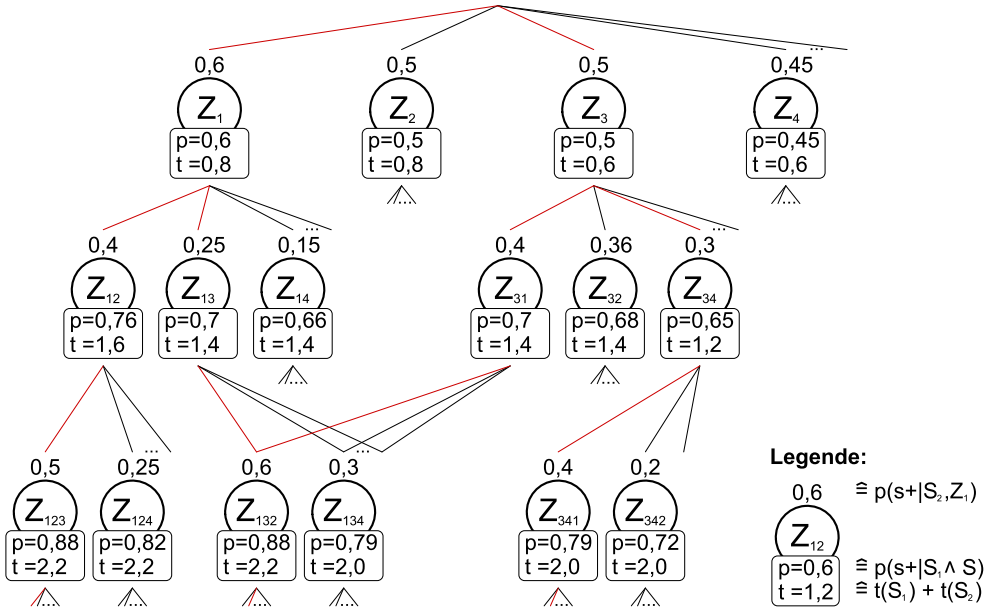


Abb. 5.15.: Vorauswahl der Lokalisierungsstrategien über einen Wahrscheinlichkeitsbaum. Die erfolgversprechendsten Sequenzen sind rot hervorgehoben

gespeichert. Vorherige Sequenzen, die dem neuen Kandidaten unterlegen sind, werden wieder aus der Liste entfernt. Ist zum Zeitpunkt der Berechnung bereits ein Knoten mit höherem Nutzen bekannt, werden vom aktuellen Knoten ausgehende Zweige nicht weiter verfolgt, sofern auch keiner der direkten Kindknoten eine maximale Erfolgschance bezogen auf die benötigte Zeit erreicht. Durch diese Abbruchbedingung wird nicht der vollständige Baum unterhalb jedes Knotens ausgewertet, wodurch viel Rechenaufwand gespart wird. Andererseits ist das Erreichen der global optimalen Lösung des POMDP nicht mehr sichergestellt, da eine hohe, auf die Kindknoten folgende Erfolgswahrscheinlichkeit nicht berücksichtigt würde. Es ist jedoch anzunehmen, dass entsprechende Konstellationen auch mit höherem Rechenaufwand im Baum nicht sicher erkennbar sind. Da die Elternknoten bei vorherigen Störungen wegen der geringeren Erfolgschance nicht ausgeführt würden, läge für die vielversprechenden nachfolgenden Kindknoten nur selten Erfahrungswissen vor. Eine mögliche Abweichung von der optimalen Policy ist daher mehr durch die eingeschränkte Erfassung des Erfahrungswissens als durch die Auswertung des Baums bedingt. Liegen für einen Knoten noch keine statistischen Daten vor, wird auf die globalen Wahrscheinlichkeiten zurückgegriffen (siehe Abschnitt 5.7). Zur Reduktion der Datenmengen werden die einzelnen Knoten erst in den Baum eingefügt, sobald das erste zugehörige Erfahrungswissen gewonnen wurde.

Als Ergebnis der Vorauswahl werden zumeist mehrere, unterschiedlich lange Strategiesequenzen identifiziert, die für eine bestimmte Ausführungszeit eine maximale Wahrscheinlichkeit für eine erfolgreiche Objektlokalisierung bieten (siehe Tabelle 5.10). Alle übrigen Kombinationsmöglichkeiten werden innerhalb der Value Iteration nicht berücksichtigt, wodurch der Rechenaufwand stark verringert wird.

$t(S_i)$	$p(s+Z_i)$	Z_i
0,6s	0,5	Z_3
0,8s	0,6	Z_1
1,2s	0,65	Z_{34}
1,4s	0,7	Z_{13}, Z_{31}
1,6s	0,8	Z_{12}
2,0s	0,82	Z_{342}
2,2s	0,88	Z_{123}, Z_{132}, Z_{312}

Tab. 5.10.: Liste der vielversprechendsten Strategiesequenz aus Abbildung 5.15

5.6.2. Trennung von Wahrscheinlichkeiten vor und nach der Anwesenheitsprüfung

Eine Besonderheit bei der Vorauswahl der Suchstrategien ergibt sich durch die Möglichkeit, dass die Ausführung einer Anwesenheitsprüfung zu einer Veränderung der Werkstücklage und somit zu einer Neubewertung der Erfolgswahrscheinlichkeiten führen kann. Bei einer variierten Bauteillage können auch bereits erfolglos aufgerufene Suchvarianten zu einer Lokalisierung führen. Zusätzlich kann z. B. ein undefinierter Testgriff die Erfolgsraten erhöhen, wenn starke Abweichungen des Werkstücks von einer Normallage durch die Fingerbewegung verringert werden. Da entsprechende Prüfstrategien jederzeit innerhalb der Behandlungsreihenfolge anwendbar sind, hängen an jedem Knoten im Baum zusätzlich zu den beschriebenen Abhängigkeiten nochmal die infolge der Prüfung veränderten Wahrscheinlichkeiten aller Strategien. Eine mögliche Abhängigkeit der Erfolgswahrscheinlichkeiten nach der Prüfung zu vorher ausgeführten Strategien konnte nicht nachgewiesen werden, weshalb die zusätzlichen Unter-Bäume alle gleich wären. Zur Reduktion der Knotenmengen werden die statistischen Daten bezüglich der Suchstrategien vor und nach einer Prüfung in separaten Bäumen erfasst, für die jeweils eine eigene Vorauswahl der vielversprechendsten Lokalisierungsstrategien erfolgt (siehe Abbildung 5.16). Eine Abhängigkeit der Erfolgschancen von der Anzahl vorheriger Anwesenheitsprüfungen konnte ebenfalls nicht nachgewiesen werden. Im zweiten Baum werden daher alle Strategieergebnisse unabhängig von der Häufigkeit möglicher Wiederholungen der Anwesenheitsprüfung gespeichert. Eine mit jeder erneuten Ausführung sinkende Erfolgschance wird wie in Abschnitt 5.3.2 beschrieben über einen Discount-Faktor berücksichtigt.

Um die aus der Vorauswahl in beiden Wahrscheinlichkeitsbäumen resultierenden *geprüften* und *ungeprüften* Strategiesequenzen korrekt in die Value Iteration zu integrieren, wurde diese um verschiedene Regeln erweitert. Diese betreffen vor allem den Algorithmus, durch den die gültigen Policies aus dem vorherigen Zeitschritt mit anderen Strategien kombiniert werden, und sorgen dafür, dass die Abhängigkeiten bei den Erfolgsraten geeignet berücksichtigt werden. Denn durch die rekursive Berechnungsweise werden die Behandlungsmaßnahmen, die bei der Planung hinter einer Strategie angefügt werden, zeitlich vor dieser ausgeführt. Die Regeln lauten inhaltlich wie folgt:

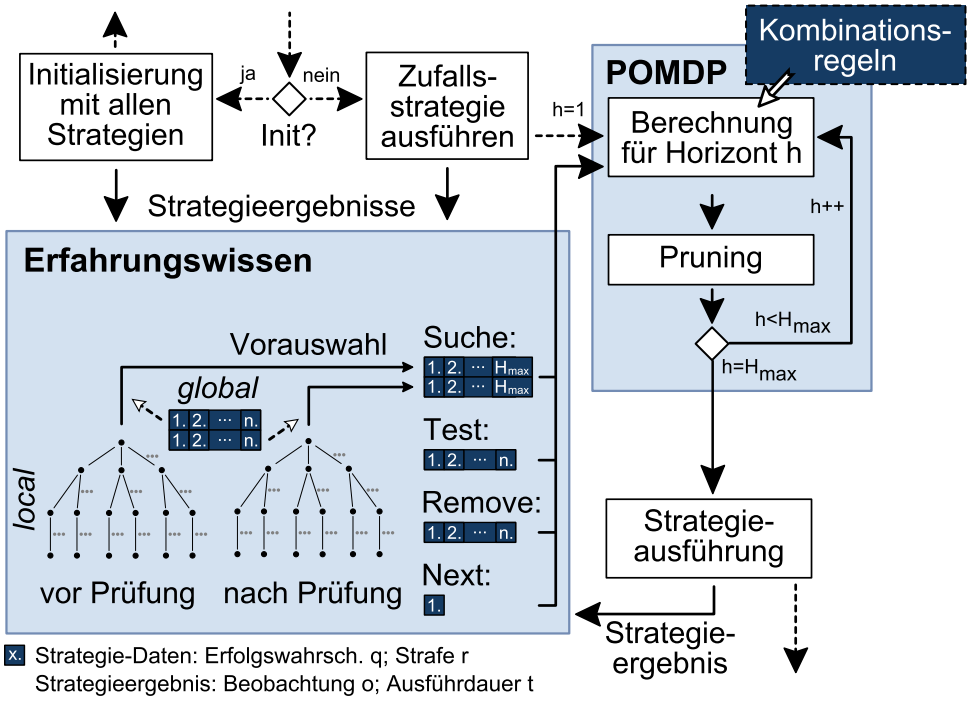


Abb. 5.16.: Zusammenspiel von Wahrscheinlichkeitsbäumen und Kombinationsregeln

- Auf eine Lokalisierungsstrategie mit Tiefe $i > 1$ muss die Strategie mit Tiefe $i - 1$ aus derselben Sequenz der Vorauswahl folgen.
- Auf eine Lokalisierungsstrategie mit Tiefe $i = 1$ aus dem geprüften Baum muss eine Prüfung folgen
- Eine Prüfung und eine geprüfte Lokalisierungsstrategie sind nur einsetzbar, wenn der vorhandene Zweig der Policy noch keine ungeprüften Strategien enthält
- Verwendete Sequenzen von Lageerkennungsstrategien müssen wegen der bedingten Wahrscheinlichkeiten vollständig integriert werden. Daher sind nur Lokalisierungsstrategien zu kombinieren deren Tiefe kleiner ist als der verbleibende Horizont
- Auf Strategien zum Prüfen, Entfernen oder Beschaffen eines Bauteils folgen alle geprüften und ungeprüften Suchstrategien aus den vorausgewählten Sequenzen unabhängig von deren Tiefe, sofern dies nicht durch die vorherige Regel ausgeschlossen wird

Durch Anwendung dieser Regeln innerhalb der Value Iteration werden nur zulässige Kombinationen erzeugt. Aus den Abhängigkeiten der Suchstrategien ergeben sich jedoch auch Einschränkungen bei der Anwendung des Prunings. Bei der enthaltenen Bewertung der verschiedenen berechneten Policies dürfen nur die Lösungen verglichen werden, die jeweils eine gültige Alternative zueinander darstellen und vom System direkt wählbar wären. Suchstrategien mit einer Tiefe $i > 1$ können nur miteinander verglichen werden, sofern ihre Tiefe übereinstimmt und sie aus derselben vorausgewählten Sequenz stammen. Anderenfalls bestehen Vorschriften über nachfolgend

durchzuführende Kombinationen, die den Nutzen der Policies beeinflussen und eine unabhängige Bewertung verhindern. Beim Pruning werden in dieser Arbeit nur die Policies verglichen, deren Startaktion einer ungeprüften Suche mit Tiefe 1 oder einer Strategie zum Anwesenheitscheck, Entfernen oder Beschaffen eines Bauteils entspricht.

5.7. Initialisierung des Erfahrungswissens

Der Einsatz eines POMDP ermöglicht eine statistisch optimale Behandlung der Störungen anhand von gelerntem Erfahrungswissen. Durch die geringen Losgrößen kommt es in der Kleinserienmontage jedoch häufig zur Inbetriebnahme neuer Applikationen. Das zur Optimierung der Strategieabfolge benötigte Erfahrungswissen liegt damit zu Beginn der neuen Fertigung noch nicht vor.

5.7.1. Verwendung globaler Wahrscheinlichkeiten

Damit dennoch ab der ersten Störung möglichst gute Policies berechenbar sind, werden zu Beginn globale Wahrscheinlichkeiten verwendet, solange noch applikationsspezifisches Erfahrungswissen fehlt. Die entsprechenden statistischen Werte ergeben sich durch Mittelung über die zugehörigen Daten aller auf dem Roboter gespeicherten Applikationen. Da sich die Erfolgswahrscheinlichkeiten zwischen den verschiedenen Lokalisierungsstrategien abhängig von der Montageaufgabe stark unterscheiden, wäre eine direkte Verwendung der Daten entsprechend der konkreten Behandlungsvarianten und des zugehörigen Belief State nicht sinnvoll. Der Mittelwert über die einzelnen Angaben würde statistisch zu einer aussagelosen Gleichverteilung führen. Die globalen Erfolgswahrscheinlichkeiten werden daher in Abhängigkeit von der Anzahl zuvor ausgeführter Lokalisierungsaufrufe ermittelt, wobei nur die Daten berücksichtigt werden, die auch als erfolgsversprechend in die Berechnung der Policy eingehen (siehe Abschnitt 5.6). Das globale Erfahrungswissen gibt damit an, welche Erfolgswahrscheinlich bei Aufruf der x-ten Lokalisierung bei einer optimierten Auswahl statistisch zu erwarten ist.

Auf Basis der globalen Daten kann bereits zu Beginn einer Applikation eine erste, statistisch geeignete Kombination der vier Behandlungsarten berechnet werden. Die Auswahl der Lokalisierungsstrategien erfolgt dabei zufällig bei Verwendung der gemittelten Erfolgsraten. Die globalen Daten werden nach und nach durch applikationsspezifisches Erfahrungswissen überstimmt.

5.7.2. Initialisierung durch Aufruf aller Strategien

Zur Bestimmung der optimalen Policy werden alle Strategien mit einander kombiniert und jeweils der zu erwartende Nutzen anhand ihrer Übergangs- und Beobachtungswahrscheinlichkeiten berechnet. Durch die Verwendung der abhängigen Wahrscheinlichkeiten benötigt jeder Knoten im Wahrscheinlichkeitsbaum eigene statistische Daten für die ausführbaren Lokalisierungsstrategien. Gleichzeitig wird pro behandelte Störung nur wenig Erfahrungswissen gewonnen. Von allen Kno-

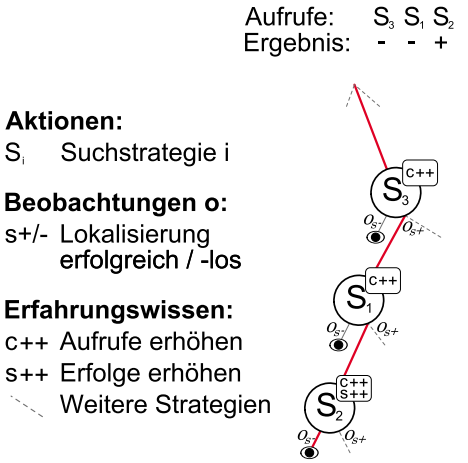


Abb. 5.17.: Erfahrungsgewinn bei einer normalen Ausführung der Policy bis zum ersten Erfolg

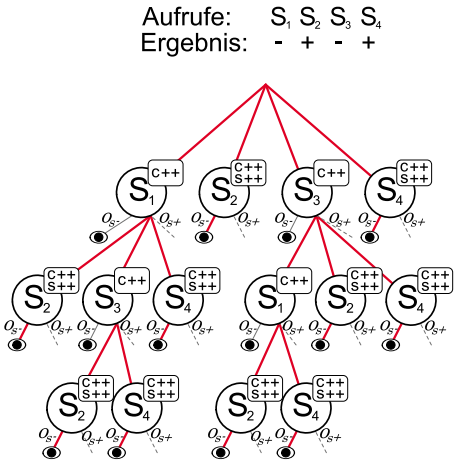


Abb. 5.18.: Erhöhter Erfahrungsgewinn durch ergebnisunabhängige Ausführung aller zugewiesenen Strategien

ten im Baum erhalten nur diejenigen entlang der optimalen Policy eine Information, die bis zur erfolgreichen Störungsbehandlung durchlaufen wurden. Die Abbildung 5.17 verdeutlicht dies an dem zur optimalen Policy gehörenden Ausschnitt des Wahrscheinlichkeitsbaums. Die Lokalisierungsstrategien S_3 und S_1 sind fehlgeschlagen, so dass nur der Aufrufzähler erhöht wird. S_2 war erfolgreich, weshalb auch der entsprechende Zähler inkrementiert wird. Durch diesen Modus, kann es abhängig von der zugewiesenen Strategiemenge lange dauern, bis für alle relevanten Knoten lokale Wahrscheinlichkeiten vorliegen.

Um schnell eine breitere Datenbasis zu erhalten, kann das Erfahrungswissen anhand der ersten Störungen initialisiert werden. Hierzu werden unabhängig vom jeweiligen Ausgang alle Strategien zur Lokalisierung und Anwesenheitsprüfung aufgerufen. Da die erzielten Ergebnisse voneinander unabhängig sind, können Daten für die Knoten entlang aller Zweige des Baums bis zur jeweils ersten erfolgreichen Lageerkennung abgeleitet werden, wie die Abbildung 5.18 zeigt. Die erhöhte Behandlungsdauer bei den zur Initialisierung verwendeten Störungen könnte nachfolgend anhand des zusätzlichen, genaueren Erfahrungswissens durch eine bessere Policy kompensiert werden. Über die Anzahl der durchzuführenden Initialisierungsläufe kann zwischen dem Informationszugewinn und der insgesamt hierfür benötigten Behandlungsdauer abgewogen werden. Statistisch sinkt die Verbesserung des Erfahrungswissens mit zunehmender Initialisierungsanzahl, so dass es eine optimale Anzahl geben sollte.

5.7.3. Korrektur von Pseudo-Maxima durch eine Zusatzstrategie

Ein weiteres Problem bei der automatischen Ermittlung der Beobachtungs- und Übergangswahrscheinlichkeiten stellt das Auftreten von Pseudo-Maxima dar. Durch die Berechnung der Aufrufreihenfolge anhand des statistischen Wissens werden nur die erfolgversprechendsten Strategien als Teil der optimalen Policy aufgerufen. Gleichzeitig führen die bedingten Wahrscheinlichkeiten dazu, dass nur die Knoten der optimalen Policy einen Informationsgewinn erhalten können. Schlagen

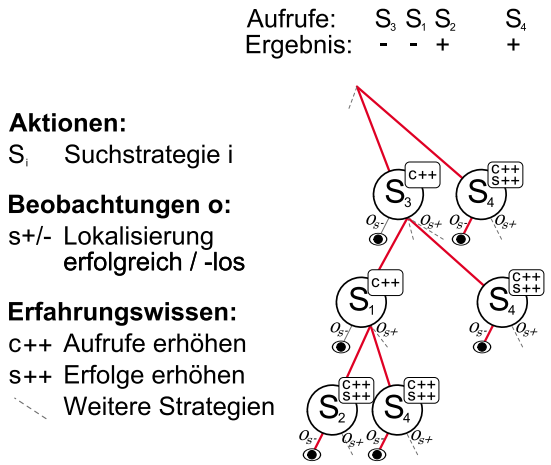


Abb. 5.19.: Zusätzlicher Erfahrungsgewinn durch Ausführung einer zufälligen Extrastrategie

Strategien mit eigentlich hohen Erfolgswahrscheinlichkeiten bei den ersten Ausführungen zufällig fehl, wird eine geringe Erfolgschance erfasst. Als Folge würde die Strategie nicht wieder aufgerufen, sofern Alternativbehandlungen erfolgversprechender scheinen. Der Fehler wird nicht entdeckt.

Um Abweichungen der Wahrscheinlichkeiten infolge einer geringen Datenmenge und somit resultierende Pseudo-Maxima zu korrigieren, kann bei jeder Störung eine zusätzliche, zufällig ausgewählte Suchstrategie ausgeführt werden. Da der Aufruf unabhängig von der berechneten Strategiereihenfolge ist, kann das Ergebnis an jedem erfolgreich ausgeführten Knoten aus der Policy als zusätzliche Information bezüglich der Wahrscheinlichkeiten einer entsprechenden Aktion genutzt werden. Wie die Abbildung 5.19 zeigt, kann eine einzelne unabhängige Strategieausführung mehrere Daten gleichzeitig verbessern. Für einen wirtschaftlichen Einsatz muss die Zusatzstrategie die statistischen Daten so weit optimieren, dass die extra aufgewendete Ausführungsdauer über eine dauerhaft geeignetere Policy kompensiert wird.

5.8. Parametrierung der Störungsbehandlung

Das Einrichten der Störungsbehandlung erfolgt über eine zusätzliche Unterstruktur im Parametrier-Wizard, die bei den Arbeitsschritten *Bauteil greifen* und *Bauteil ablegen* in die MMS integriert ist (siehe Abschnitt 4.3.2). Der Bediener kann bei Inbetriebnahme der Applikation bereits erste Strategien für vermutete Fehlerfälle hinzufügen. Für ein gutes Kosten-Nutzen-Verhältnis ist hierbei zwischen dem Parametrieraufwand und der Auftretenswahrscheinlichkeit der resultierenden Störung abzuwägen. Um unnötigen Aufwand zu vermeiden, kann der Nutzer geeignete Strategie auch hinzufügen, wenn eine Störung im Betrieb einen Ausfall verursacht und einen Bedienereingriff erfordert. Hierzu pausiert die Ausführung, sobald eine Störung im Betrieb mit den zugewiesenen Strategien nicht autonom behebbar ist, und der Bediener wird gerufen. Dieser bewertet die Störung und entscheidet, ob er diese manuell behebt oder eine passende Methoden zur Behandlung hinzufügen.

Zur Unterstützung der Strategieauswahl wird die vorliegende Störungsart gemeldet. Bei Lokalisierungsproblemen wird zudem das aktuelle Kamerabild angezeigt. Zusätzlich gibt die Objektlage vor der Kamera bzw. im Greifer je nach Störung Auskunft über mögliche Fehlerursachen. Die Behandlungsmöglichkeiten von verrutschten oder verlorenen Werkstücken sind zumeist direkt ersichtlich. Ist die Auswahl einer geeigneten Behandlung von Lokalisierungsproblemen nicht eindeutig, kann der Bediener eine Unterstützungsfunktion aufrufen. Diese grenzt die Auswahl ähnlich wie bei der Lokalisierung in Abschnitt 4.4.2 durch Beantwortung verschiedener Fragen ein. Die Abbildung 5.20 zeigt den hinterlegten Entscheidungsbaum. Am Ende dieses Prozesses bekommt der Bediener einen Hinweis über eine oder mehrere potentiell passende Strategien. Diese können dann geprüft werden, bevor der Bediener die geeignete Strategie auswählt. Nach dem Hinzufügen und erfolgreichen Ausführen der Strategie wird der Automatikbetrieb fortgesetzt.

Die Parametrierung der einzelnen Strategien erfolgt in einem von der Störungsart abhängigen Unterwizard jeweils über eine eigene Seite. Auf dieser werden zusätzlich auch Funktionen zum direkten Testen der Einstellungen angeboten. Die Abbildung 5.21 verdeutlicht dies an der Bedienoberfläche für die Strategie *Position variieren*. Die ausgewählte Schrittweite und die Funktion der

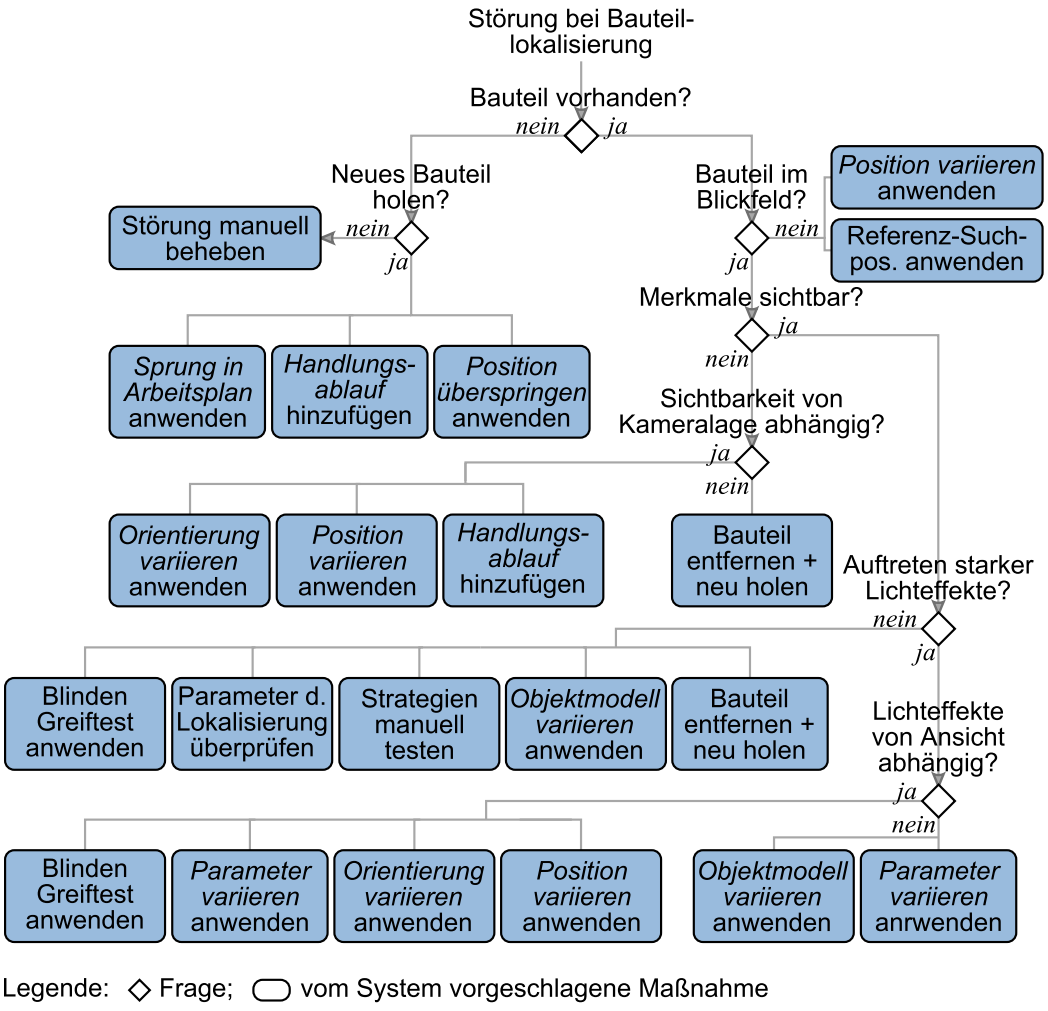


Abb. 5.20.: Entscheidungsbaum zur Unterstützung des Bedieners bei der Strategieauswahl

Abb. 5.21.: Wizard für Strategie *Position variieren*

Strategie können hier über entsprechenden Roboterbewegungen anhand der zyklisch aktualisierten Kamerabilder kontrolliert werden.

Als weitere Parametriermöglichkeit kann für jeden Arbeitsschritt einer von drei Modi gewählt werden, um die Berechnung der auszuführenden Strategiesequenz an die Randbedingungen der Applikation anzupassen:

- **Statistisch Zeitoptimal (standard):** Die Reihenfolge der Strategien wird, wie in Abschnitt 5.1 beschrieben, anhand der erfassten Wahrscheinlichkeiten und Ausführungszeiten berechnet. Die Strafen für die Entfernung eines Bauteils bzw. für das unerwünschte Beschaffen eines neuen Werkstücks basieren zunächst auf Standardwerten und können vom Bediener angepasst werden. Da alle anderen Kosten in Zeiteinheiten festgelegt sind, kann der Bediener den Zielwert z. B. danach bestimmen, wie viel Zeit zum manuellen Beheben einer Fehlentscheidung des Systems benötigt würde.

- **Minimale Bauteilausschleusung:** Die Reihenfolge der Strategien berechnet sich wie bei der vorherigen Option. Jedoch werden vor dem Entfernen oder Beschaffen eines Bauteils stets alle Lokalisierungsstrategien ausgeführt, um die höchste Entscheidungssicherheit zu erzielen.

- **Zeitlimit:** Bei dieser Option wählt der Bediener ein maximales Zeitlimit für die Störungsbehandlung. Die Policy wird wie zuvor berechnet, jedoch werden nur so viele Strategien kombiniert, dass eine maximale Ausführungsdauer der Sequenz nicht überschritten wird. Da die Policies für einen Horizont von $h = 1$ stets mit dem Entfernen oder Beschaffen eines Bauteils beginnen, wird die Behandlung innerhalb der Zeitvorgabe stets abgeschlossen.

Als weiteren Parameter kann der Bediener den Discount-Faktor γ anpassen, sofern eine Anwesenheitsprüfung zugewiesen ist, mit der die Objektlage verändert wird. Dieser Parameter gibt an, wie stark die Erfolgswahrscheinlichkeiten der Lokalisierungsstrategien nach mehrfacher Prüfung abnehmen. Bei einem hohen Wert unterhalb von 1 werden häufig Anwesenheitsprüfungen ausgeführt, da die Wahrscheinlichkeiten ohne starke Reduktion zurückgesetzt werden. Die Strategiese-

quenz besteht in der Folge abhängig vom zuvor erläuterten Modus nur aus wenigen verschiedenen, aber häufig angewendeten Lokalisierungsstrategien mit hohen Erfolgsraten. Bei einem geringeren Wert sinkt die Erfolgsrate nach der zweiten Prüfung stärker, wie in Abschnitt 5.3.2 beschrieben. Entsprechend werden zumeist mehr verschiedene Strategien ausgeführt, bevor geprüft wird.

Als Standardeinstellung hat sich ein Wert von 0.3 bewährt. Anpassungen sind zumeist nur sinnvoll, wenn das Objekt bei der Prüfung z.B. deutlich anders positioniert wird, so dass die Erfolgschancen konstanter bleiben.

Der Horizont für den POMDP ist nicht vom Bediener einzustellen. Sofern Strategien zum Prüfen, Entfernen und oder Beschaffen zugewiesen sind, wird mit einem Horizont in der Höhe der doppelten Anzahl an Lokalisierungsstrategien gestartet. Nachfolgend ergibt sich der neue Horizont aus der um 2 erhöhten Einstellung bei der vorherigen Störung. Auf diese Weise kann sich der Wert iterativ erhöhen. Durch einen Diskount-Faktor < 1 ist die optimale Strategiesequenz durch die sinkenden Erfolgsraten in jedem Fall in der Länge begrenzt. Für eine feste zeitliche Limitierung der Behandlungsdauer kann der Bediener den entsprechenden, oben erläuterten Modus wählen.

5.9. Zusammenfassung

In diesem Kapitel wurde eine neuartige Störungsbehandlung vorgestellt, mit der ein Bediener die Robustheit der erstellten Roboterprogramme bzgl. auftretender Störungen bei der Lokalisierung und Manipulation von Objekten optimieren kann. Die Störungsdetektion erfolgt durch Auswertung der Greiferöffnungsweite sowie der Rückmeldung der Objektlageerkennung. Da eine passende Behandlung applikationsabhängig und aus der Störungsmeldung nicht ableitbar ist, werden erlaubte Behandlungen durch den Bediener vorgeben. Hierzu enthält eine Datenbank effizient parametrierbare Strategien aus vier unterschiedlichen Klassen zum Lokalisieren, Prüfen der Anwesenheit, Entfernen sowie zum Beschaffen neuer Werkstücke. Für eine schnelle und korrekte Behandlung unterschiedlicher Störungsursachen wird die Strategiereihenfolge zur Laufzeit durch einen POMDP auf Basis einer Fehlerschätzung geplant. Die erforderlichen statistischen Daten werden als Erfahrungswissen aus der Behandlung vorheriger Störungen gelernt. Wegen der regelmäßigen Veränderung des Erfahrungswissens ist eine zyklische Aktualisierung der optimalen Strategiesequenz nötig. Durch die große Anzahl an möglichen Strategien und die Abhängigkeit der Erfolgschancen bei der Lokalisierung ist eine direkte Berechnung zu zeitaufwändig. Zur Reduktion des Rechenaufwands werden die bedingten Erfolgsraten der Lokalisierungsstrategien in einem Wahrscheinlichkeitsbaum gespeichert. Dies erlaubt eine Vorauswahl der effizientesten Strategieabfolgen, so dass bei der Lösung des POMDP weniger Kombinationen zu berücksichtigen sind.

Da bei einer Produktionsumstellung direkt nach der Programmierung des Systems noch kein Erfahrungswissen für die neue Applikation vorliegt, wird zunächst auf globale Wahrscheinlichkeiten zurückgegriffen. Zusätzlich wurden Methoden integriert, um den Informationszuwachs zu

beschleunigen und durch statistische Effekte auftretende Abweichungen im Erfahrungswissen zu beheben. Der Nutzen bzw. der optimale Einsatz dieser Funktionen ist im Folgenden zu evaluieren.

Die Inbetriebnahme der Störungsbehandlung kann als Teil des Gesamtkonzepts der MMS durch Beantwortung verschiedener Fragen zum Fehlerbild anhand eines hinterlegten Entscheidungsbaums unterstützt werden.

6. Umsetzung von MMS und Störungsbehandlung

Das folgende Kapitel beschreibt einen Systementwurf zur Umsetzung der vorgestellten Konzepte für die MMS und die Störungsbehandlung. Hierzu zeigt Abschnitt 6.1 zunächst eine Gesamtübersicht über das System und erläutert die erforderlichen Schnittstellen, die von den Hardwarekomponenten und -modulen bereitzustellen sind. Die Abschnitte 6.2 und 6.3 erläutern, wie die Behandlungsstrategien und die Behandlungssteuerung umgesetzt sind, um einen universellen Einsatz sowie eine Erweiterbarkeit für verschiedene Anwendungen zu ermöglichen. Die Datenstrukturen und das Zusammenspiel der einzelnen Datenbanken zur Speicherung des Erfahrungswissens wird in Abschnitt 6.4 beschrieben.

6.1. Gesamtübersicht

Das Gesamtsystem setzt sich aus den steuerbaren Hardwarekomponenten des Robotersystems und verschiedenen Software-Modulen zusammen, wie die Abbildung 6.1 verdeutlicht. Ein Roboter-Framework stellt dabei über ein Hardware-Steuermodul einen gebündelten Zugriff auf das Bildverarbeitungsmodul sowie auf die Greifer- und Robotersteuerung zur Verfügung. Die umgesetzte MMS besitzt zur Programmierung der Montageaufgaben neben dem Zugriff auf dieses Modul auch eine breite Schnittstelle zu verschiedenen Daten und Methoden des Roboter-Frameworks. Die Kommunikation mit dem Nutzer erfolgt über WPF-Bedienoberflächen, die über einen Touchscreen angezeigt und bedient werden. Die umgesetzte Störungsbehandlung wird ebenfalls über die MMS parametriert. Zur Ausführung der Strategien verfügt diese Komponente über einen Zugriff auf das Hardware-Steuerungsmodul sowie auf verschiedene Methoden des Frameworks. Der genaue Umfang der Schnittstellen wird im Folgenden erläutert.

6.1.1. Roboter-Steuerung

Die Roboter-Steuerung muss das Auslesen der aktuellen TCP-Position sowie das Anfahren von Raumpositionen mit verschiedenen Bewegungsmodi ermöglichen.

Auslesen der aktuellen Tool-Center-Point-Lage
Über diese Funktion kann die aktuelle Lage des TCP ausgelesen werden. Die Angabe kann sowohl in den sechs Achswinkeln als auch in Basiskoordinaten (X, Y, Z, Rx, Ry, Rz) erfolgen.

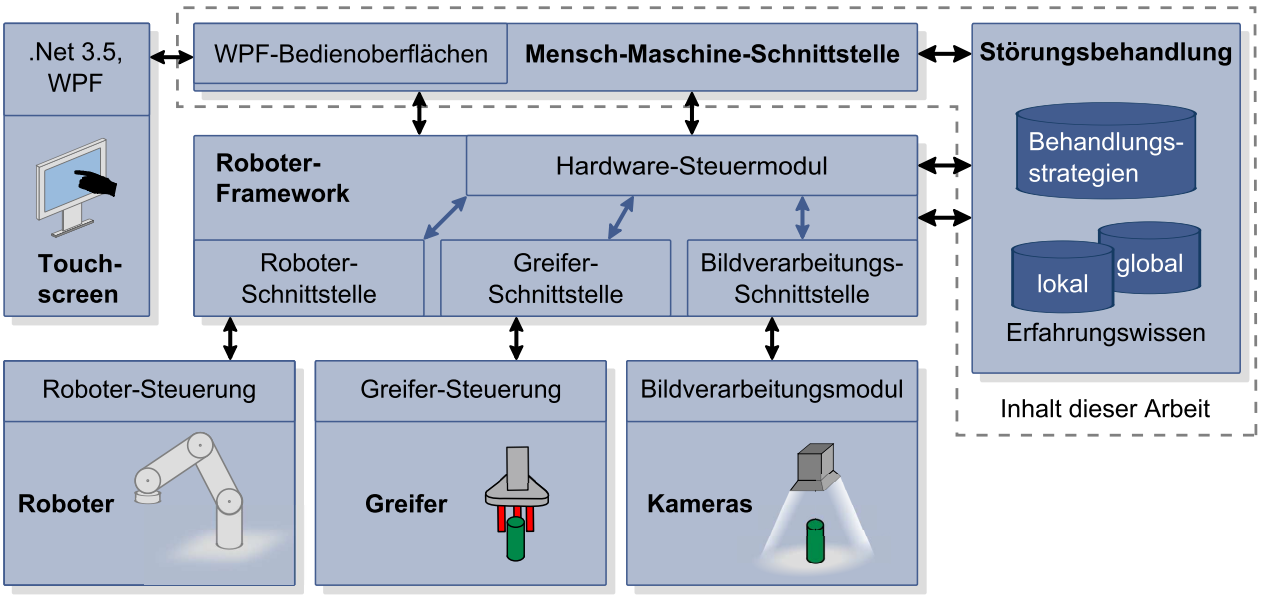


Abb. 6.1.: Überblick über die Komponenten des Gesamtsystems

Anfahren einer Solllage im Raum

Zur Positionierung des Roboterarms wird die Solllage des TCP anhand der Achswinkel oder der kartesischen Koordinaten vorgegeben. Die Anfahrbewegung ist über verschiedene Interpolationsmöglichkeiten variierbar. Ein *lineares Anfahren* führt zur direkten, kürzesten Bahn zwischen der Start- zur Zielposition. Im *Punkt-zu-Punkt-Modus* bewegt sich jede Achse gleichmäßig von ihrer aktuellen Position in die Solllage, wobei alle Achsrotationen gleichzeitig starten und enden. Zusätzlich können Bewegungen durch Überschleifen mehrerer Positionen einer Bahn flüssiger und schneller ausgeführt werden.

Setzen der maximalen Bahngeschwindigkeit

Zur manuellen Kontrolle der Roboterbewegung und zur Vermeidung von starkem Überschwingen des Arms beim Bremsen muss die maximale Bahngeschwindigkeit einstellbar sein.

6.1.2. Greifer-Steuerung

Über die Greifer-Schnittstelle muss der am Handflansch des Roboters montierte Greifer ansteuerbar sein. Zu den Funktionen gehören das Setzen und Auslesen der Öffnungsweite sowie ein kraftgeregelte Greifen.

Kraftgeregeltes Greifen

Für das kraftgeregelte Greifen ist die zur Ausführung aufrecht zu erhaltende Kraft anhand der Motorströme vorzugeben. Über die Greifrichtung sind sowohl Innen- als auch Außengriffe wählbar.

Definierte Fingeröffnungsweite einstellen

Anhand des Sollabstands der Finger vom Greifzentrum kann die Öffnungsweite mittels Positionsregelung für ein kollisionsfreies Annähern an ein Bauteil oder zum Freigeben von Werkstücken eingestellt werden.

Fingeröffnungsweite auslesen
Zur Speicherung der vom Bediener eingestellten Griffweite, zur Berechnung der Fingerpositionen oder zur Überprüfung eines korrekten Griffs muss die aktuelle Öffnungsweite in Form des Fingerabstands vom Greifzentrum auslesbar sein.

6.1.3. Bildverarbeitungsmodul

Das Bildverarbeitungsmodul muss die gesamte zur Objektlokalisierung erforderliche Funktionalität zur Verfügung stellen. Hierzu gehört neben der Ansteuerung von Kameras und einer aktiver Beleuchtung vor allem das Erstellen und Parametrieren von Objektmodellen sowie die Erkennung und Lageberechnung. Zusätzlich sind die beim Bearbeiten der Modelle vom Bediener zu bewertenden Zwischenergebnisse aus der Bildverarbeitungskette aufzubereiten und automatische Parametrierfunktionen zur Verfügung zu stellen.

Objektmodelle verwalten
Zum Integrieren oder Entfernen von Objektlokalisierungen innerhalb der Arbeitsschritte sowie zur Störungsbehandlung sind Methoden zum Erstellen, Löschen und Kopieren von der Modelle nötig.

Modellparameter setzen
Für eine robuste Lokalisierung muss der Bediener alle verschiedenen Parameter eines Objektmodells über entsprechende Methoden einstellen können. Die Anzahl und Art der erforderlichen Parameter hängen vom jeweils ausgewählten Lageerkennungsverfahren ab.

Modellparameter und Informationen auslesen
Zur Steuerung der Bedienoberflächen müssen die unterschiedlichen Grenz- und Initialwerte sowie die jeweils aktuellen Einstellungen aller Modellparameter auslesbar sein. Zusätzlich sind Informationen über geeignete Schrittweiten zum Einstellen der Sollwerte sowie ein kurzer Hilfetext für den Nutzer nötig.

Parametrierschritt aufrufen
Für die einzelnen in Abschnitt 4.4.3 beschriebenen Parametrierschritte sind unterschiedliche Methoden erforderlich, um ein geeignetes Feedback zur Bewertung der Parameter durch den Bediener aufzubereiten und evtl. eine automatische Justierung durchzuführen. Der Aufruf dieser Funktionen wird über die Art des Lokalisierungsverfahrens sowie über einen, den aktuellen Einstellschritt spezifizierenden Parameter gesteuert.

Bild anzeigen
Die Visualisierung der Bilddaten erfolgt über eine Callback-Funktion, an die der Inhalt von Steuerelementen mittels geeigneter Methoden geknüpft werden kann, um automatisch die aktuellsten Bilder in den Bedienoberflächen der MMS anzuzeigen. Die Funktion ist mit jedem neu erstellten Bild aufzurufen.

Lokalisierung ausführen
Eine der wichtigsten Methoden des Bildverarbeitungsmoduls ist das Ausführen einer Objektlageerkennung. Hierzu werden das zu lokalisierende Objektmodell sowie die aktuelle TCP-Lage des Roboterarms übergeben. Als Ergebnis wird die erfasste Objektlage bzw. eine Fehlermeldung im Falle eines Misserfolgs zurückgeliefert. Die bei erfolgreicher Ausführung berechnete Roboterkoordinate gibt dieselbe relative Lage des TCP zum Objekt an, die auch während der Erstellung des Modells vorlag.

6.1.4. Roboter-Framework

Das Roboter-Framework bildet das zentrale Element in der Softwarestruktur des Robotersystems. Neben dem Zugriff auf die Hardwarekomponenten werden für die MMS und die Störungsbehandlung alle erforderlichen Daten und Methoden zum Verwalten, Bearbeiten und Ausführen der Arbeitspläne sowie der enthaltenen Aktionen zur Verfügung gestellt.

Hardware-Steuerung übergeben
Das Framework muss über das Hardware-Steuerungsmodul einen gebündelten Zugriff auf die Schnittstellen aller Hardwarekomponenten bieten.

Arbeitspläne verwalten
Zur Visualisierung der vorhandenen Arbeitspläne für den Bediener kann eine vollständige Liste abgerufen werden. Darüber hinaus sind Methoden zum Erstellen und Löschen von Arbeitsplänen sowie zum Laden und Speichern der Inhalte vorhanden. Die Daten umfassen neben den enthaltenen Arbeitsschritten und den verknüpfenden Transitionen auch den Arbeitsplannamen sowie eine textuelle Beschreibung.

Arbeitspläne ausführen
Vollständig parametrierte Arbeitspläne werden über das Framework gestartet, gestoppt oder pausiert. Alle internen Variablen, wie z.B. Zähler von Palettenfüllständen, sind zurücksetzbar.

Arbeitsschritte verwalten
Das Framework bietet Methoden zum Erstellen und Verknüpfen von Arbeitsschritten innerhalb eines Arbeitsplans bzw. um diese zu löschen. Zusätzlich besteht Zugriff auf alle vom Bediener einzustellenden oder im Rahmen der Störungsbehandlung benötigten Parameter, Objektmodelle und Positionen der Aktionen.

Arbeitsschritte ausführen
Zum Test der Parametrierung und zur gezielten Störungsbehandlung müssen sowohl die Arbeitsschritte als Ganzes als auch die zum Ablauf gehörenden Teilfunktionen separat ausführbar sein.

6.1.5. Mensch-Maschine-Schnittstelle

Die Mensch-Maschine-Schnittstelle dient zur Programmierung und Steuerung des Robotersystems durch den Bediener. Die Implementierung erfolgte nach dem Model-View-Viewmodel-Pattern in .NET/C# und Microsoft Windows Presentation Foundation (WPF) aus dem .NET-Framework. Bei der Umsetzung unterteilt sich der Code aller Oberflächen bzw. der einzelnen Templates, aus denen diese aufgebaut sind, in zwei Bereiche. Die visuelle Darstellung wird als View über .xaml-Dateien beschrieben. Das in C# implementierte Viewmodel umfasst hingegen das Verhalten und die Funktion der Bedienoberflächen. Zusätzlich dient diese Schicht der Aufbereitung aller erforderlichen Daten aus dem Framework, das innerhalb des Patterns dem Datenmodell (Model) entspricht. Der Funktionsumfang der MMS ist in Kapitel 4 beschrieben.

6.1.6. Störungsbehandlung

Dieses Modul dient zur Parametrierung und Steuerung der Störungsbehandlung. Es enthält die Datenbanken mit den vom Bediener wählbaren Strategien sowie dem gelernten lokalen und globalen Erfahrungswissen, das zur Optimierung der Ausführungszeit eingesetzt wird. Das Modul ist in C# implementiert. Die Bedienoberflächen zur Parametrierung der Strategien sind in WPF umgesetzt.

Auswahl der zur Parametrierung erforderlichen Wizardseiten
Wie in Abschnitt 5.8 beschrieben, erfolgt die Parametrierung der Störungsbehandlung über einen Wizard innerhalb der MMS. Die Steuerung der anzuzeigenden Seiten erfolgt über das Modul zur Störungsbehandlung, um eine einfache Erweiterbarkeit ohne Änderungen am Code der MMS zu ermöglichen. Hierzu implementiert jede Strategie eine Methode, über die eine Anwendbarkeit für den aktuellen Fehlerfall geprüft und ggf. eine zugehörige Wizardseite, bestehend aus View und Viewmodel, zurückgegeben wird. Durch Iteration über alle Strategien werden die anzuzeigenden Bedienoberflächen aufgelistet (siehe Abschnitt 6.2).

Behandlungsstrategien verwalten
Das Modul zur Störungsbehandlung umfasst Methoden, um die verfügbaren Strategien über die jeweiligen Wizardseiten für eine Anwendung im Fehlerfall auszuwählen bzw. wieder zu deaktivieren. Für die Parametrierung durch den Bediener kann auf alle relevanten Daten einer Strategie zugegriffen werden.

Steuerung der Störungsbehandlung parametrieren
Über eine separate Wizardseite können die Steuerungsparameter der Störungsbehandlung eingestellt werden, wie z.B. die Priorität der Taktzeitoptimierung (siehe Abschnitt 5.8).

Störung behandeln
Diese Methode dient im Automatikbetrieb des Robotersystems zur Behandlung von detektierten Störungen. Die Funktion und die erforderlichen Eingangsparameter sind in Abschnitt 6.3 näher erläutert. Weitere Details zur Störungsbehandlung sind in Kapitel 5 ausgeführt.

6.2. Strategien zur Störungsbehandlung

Bei der Implementierung der zur Störungsbehandlung eingesetzten Strategien sind verschiedene Aspekte zu berücksichtigen. Zum einen steht im Zusammenhang mit der gesamten MMS ein schnelles Einrichten der Behandlung ohne Expertenwissen im Fokus, weshalb die Strategien gezielt mit nur wenigen, verständlichen Parametern zu entwerfen sind. Zum anderen ist eine flexible Einsetzbarkeit des Behandlungsmoduls für verschiedene Anwendungsfälle zu erreichen, um wie im vorliegenden Fall z.B. Störungen bei der Objektlageerkennung sowie der Bauteilmanipulation mit demselben Ansatz adressieren zu können. Dieser Aspekt umfasst insbesondere auch eine einfache Erweiterbarkeit um neue Strategien oder Behandlungsarten, da andere Einsatzmöglichkeiten in der Regel abweichende Anforderungen besitzen. Das Klassendiagramm in Abbildung 6.2 zeigt die wichtigsten Methoden und Parameter der Strategien.

Alle Strategien sind von einer Klasse abgeleitet, um einheitliche Schnittstellen für die Erweiterung der Datenbank mit neuen Behandlungsmethoden zu garantieren. Jeder Strategietyp verfügt zur Identifikation über eine feste *ID* vom Typ Guid. Da jede Behandlungsart in den Wahrscheinlichkeitsbäumen mehrfach mit unterschiedlichen Erfolgsraten enthalten ist, wird das Erfahrungswissen in den Knoten über eine gesonderte Klasse *BaumKnoten* gespeichert (siehe Abschnitt 6.3). Bei der Ausführung wird die jeweilige Strategie anhand der ID dem Baumknoten zugeordnet (siehe Abschnitt 6.4). Bei den meisten Strategietypen ist die Variantenanzahl fix und kann über global feste IDs adressiert werden. *VariierePosition* teilt sich z.B. fest in acht Unterstrategien für die verschie-

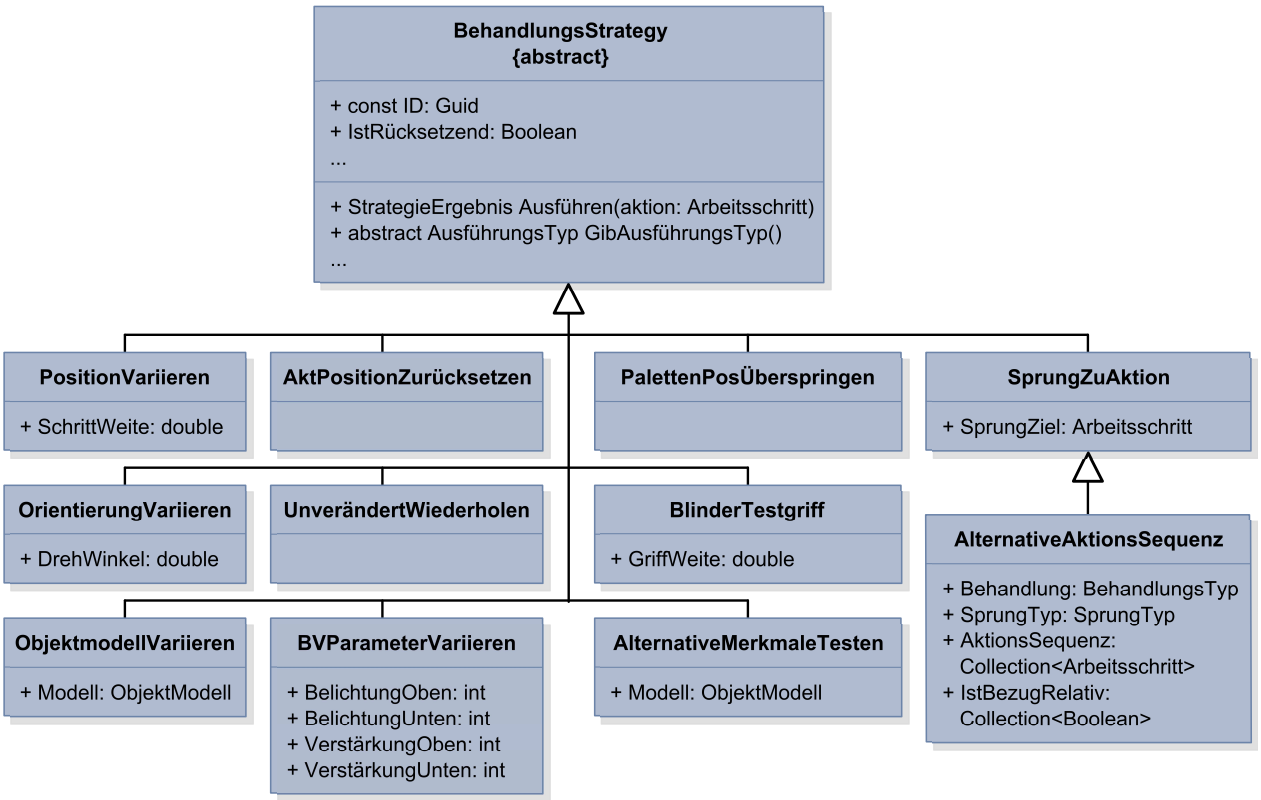


Abb. 6.2.: Klassendiagramm zu Aufbau und Vererbung der Behandlungsstrategien

denen Richtungen. Die *AlternativeAktionsSequenz* kann beliebig oft unterschiedlich eingerichtet werden, weshalb ein Teil der ID zufällig generierbar ist. Der Boolean *IstRücksetzend* gibt an, ob die Erfolgschancen der bereits ausgeführten Behandlungsmetoden nach dem Aufruf der aktuellen Strategie zurückzusetzten sind, weil das zu lokalisierende Objekt bewegt wurde.

Für die Störungsbehandlung verfügt jede Strategie über eine Methode *Ausführen()*, der der aktuelle Arbeitsschritt als Parameter übergeben wird. Dieser enthält einen Verweis auf das Hardware-Steuermodul und ermöglicht den Zugriff auf die Daten und Methoden der fehlgeschlagenen Aktion, um diese zu verwenden oder zu verändern. Das Rückgabeobjekt vom Typ *StrategieErgebnis* umfasst neben der Information zum Ausführungserfolg auch einen zur Original-Methode passenden, unspezifischen Rückgabeparameter vom Typ *object*, der über einen Wrapper in das Zielformat umgewandelt wird (siehe Abschnitt 6.3). Der Rückgabewert der abstrakten Methode *GibAusführungsTyp()* spezifiziert, ob die Strategie allein oder vor der Original-Methode ausgeführt wird.

Zusätzlich zu den beschriebenen Variablen verfügen einige der abgeleiteten Strategien über spezifische Daten, wie z. B. die Schrittweite zum Variieren der Kameraposition. Die Strategie *AlternativeAktionsSequenz* kann gleichermaßen für alle Behandlungsarten eingesetzt werden, indem der gewünschte Typ festgelegt wird. Die übrigen Daten umfassen die alternativ auszuführenden Arbeitsschritte und geben an, ob die enthaltenen Bewegungen relativ zu der aktuellen Roboterlage oder einer zu lokalisierenden Referenz ausgeführt werden. Über den Sprungtyp wird definiert, ob im Anschluss an den Aufruf ein Sprung ausgeführt, derselbe Arbeitsschritt wiederholt oder mit dem nachfolgenden fortgefahren wird.

Bedienoberflächen zur Parametrierung der Strategien

Neben den abgeleiteten Strategien sind zur Erweiterung der Datenbank entsprechende Klassen für die zur Auswahl und Parametrierung benötigten Bedienoberflächen hinzuzufügen. Im vorliegenden Fall umfasst dies entsprechend des MVVM-Patterns (Model View ViewModel) jeweils eine *ParamStrategieView* sowie ein zugehöriges *ParamStrategieViewModel*. Um die Auswahl durch den Benutzer zu vereinfachen, werden stets nur die anwendbaren Strategien angezeigt. Hierzu implementiert jedes ViewModel eine statische Methode *IstAnwendbar(aktion: Arbeitsschritt, param: objekt)*, die per Boolean die Anwendbarkeit entsprechend den Randbedingungen der Aktion zurückliefert. Auf diese Weise können fast alle Seiten mit dem in Listing 6.1 dargestellten Template in die Funktion zur Steuerung des Wizards eingebunden werden. Lediglich die Seiten für die AlternativeAktionsSequenz weichen ab, da für jede bereits vorhandene Strategie dieses Typs eine eigene Seite einzufügen ist. Die View wird nicht explizit erstellt, sondern dient zur Spezifikation des Seitenlayouts im Wizard.

```
1 if (paramStrategieViewModel.IstAnwendbar(zielAktion, objektModell))
2 {
3     wizard.pageCollection.Add(new ParamStrategieViewModel(...));
4 }
```

Listing 6.1: Template zur automatischen Steuerung des Wizardinhalts

6.3. Steuerung der Störungsbehandlung

Bei der Implementierung der Steuermethode für die Störungsbehandlung liegt der Fokus wie schon bei den Strategien auf einer flexiblen Wiederverwendbarkeit für unterschiedliche Anwendungsfälle. Zum einen kann die Störungsbehandlung eine vorhandene Originalmethode um die Verarbeitung von erkannten Störungen erweitern, wie bei der Objektlokalisierung. Zum anderen kann das Modul als zusätzliche Funktion in den Programmablauf integriert werden, um z. B. fehlerhafte Griffe zu detektieren und zu verarbeiten. Beide Szenarien werden durch alternative Ausführungen und verschiedene Modi abgebildet.

Ablauf der Störungsbehandlung

Im Falle der Erweiterung einer Funktion wird die Originalmethode beim Aufruf der Störungsbehandlung zusammen mit möglichen Eingangsparametern übergeben, wie in Listing 6.2 dargestellt. Die Methode ist hierzu wie weiter unten dargestellt entsprechend der Schnittstellen zu kapseln. Eine *PrüfeErgebnis()*-Methode kann zur Detektion von Störungen eingesetzt werden, wenn keine Originalmethode vorliegt oder diese kein Prüfergebnis zurückliefert.

Ist eine Originalmethode vorhanden, wird diese zu Beginn ausgeführt, wie das Listing 6.2 zeigt. Das Ergebnis wird ggf. über die Überprüfungsfunktion ermittelt (Zeilen 7). Im Falle einer Störungsdetektion wird unterschieden, ob eine erforderliche Menge an statistische Daten für eine gezielte Behandlung vorhanden ist oder ob ein Initialisierungslauf mit Ausführung aller Strategien durchzuführen ist (Zeile 11). Bei ausreichendem Erfahrungswissen wird die optimale Policy mit dem in Kapitel 5 beschriebenen Ansatz berechnet und die per ID referenzierten Strategien zyklisch ausgeführt. Hierbei wird über den Ausführungstyp unterschieden, ob die Behandlung alleinstehend oder in Kombination mit der originalen Methode aufgerufen wird (Zeilen 20 bis 28). Das Ergebnis sowie die Ausführungsdauer (Zeilen 18 und 29) werden zusammen mit dem zugehörigen Knoten der Policy temporär gespeichert, bis feststeht, ob eine erfolgreiche Behandlung möglich war, und eine entsprechende Einordnung des Erfahrungswissens erfolgen kann. Anhand der getätigten Beobachtung wird jeweils die Policy für den nächsten Zyklus ausgewählt (Zeile 33). Wurde die Störung durch eine Ausführung der vollständigen Policy nicht behoben, wird der Bediener alarmiert.

```
1  BehandlungsErgebnis Ausführen(Arbeitsschritt aktion, Func<object,
       StrategieErgebnis> origMethodeWrap, object inputParam, Func<bool>
       prüfeErgebnis)
2  {
3    // Initialisierung
4    ...
5
6    if (origMethodeWrap != null) stratErgebnis = origMethodeWrap(inputParam);
7    if(prüfeErgebnis != null) stratErgebnis.Erfolg = prüfeErgebnis();
8
9    if(!stratErgebnis.Erfolg)
10   {
```

```
11      if(!StatistikInitialisiert()) {InitLaufAusführen();}
12      else
13      {
14        aktPolicy = berechneOptimalePomdpPolicy(); //aktuelle Policy
15        while( aktPolicy != null )
16        {
17          aktStrategie = GibStrategieZuID(aktPolicy.Knoten.ID); //aktuelle
              Strategie
18          starteZeitmessung();
19
20          if(aktStrategie.GibAusführungsTyp() == AusführungsTyp.Prae)
21          {
22            aktStrategie.Ausführen(aktion);
23            stratErgebnis = originalMethode(inputParameter);
24          }
25          else if(aktStrategie.GibAusführungsTyp() == AusführungsTyp.Single)
26          {
27            stratErgebnis = aktStrategie.Ausführen(aktion);
28          }
29          ausführdauer = BeendeZeitmessung();
30          if(prüfeErgebnis!= null) stratErgebnis.Erfolg = prüfeErgebnis();
31
32          SpeicherErgebnisseTemporär(aktPolicy.Knoten, stratErgebnis,
              ausführdauer);
33          aktPolicy = WähleFolgePolicy(aktPolicy, stratErgebnis);
34        }
35        if(!IstStörungErfolgreichBehandelt()) ErwarteBedienerEingriff();
36        SpeichereErfahrungswissen(stratErgebnis);
37    }}
38    return behandlungsErgebnis;
39  }
```

Listing 6.2: Pseudocode der Methode zur Steuerung der Störungsbehandlung

Integration in bestehende Software

Bei der Integration des vorgestellten Ansatzes in eine bestehende Softwarestruktur sind zwei Anwendungsfälle zu unterscheiden. Sollen Störungen behandelt werden, die bisher nicht detektiert wurden, kann die Methode Ausführen(...) direkt aufgerufen werden. Die OriginalMethode entspricht hierbei *null* und *PrüfeErgebnis()* ist ggf. so zu kapseln, dass der Rückgabewert vom Typ Boolean ist. Bei nicht-binären, differenzierten Prüfresultaten ist zu analysieren, ob statt der statistischen eine gezieltere Behandlung möglich ist. Dient die Störungsbehandlung der Erweiterung einer bestehenden Methode, ist diese durch eine Kapselung der Ausführen-Funktion zu ersetzen. Das Listing 6.3 zeigt hierzu den originalen Methodenaufruf (Zeile 1) sowie die gekapselte Umsetzung (Zeile 3-9). Um innerhalb der Störungsbehandlung auf die zu erweiternde Funktion zugreifen zu können, ist diese hinsichtlich der geforderten Schnittstellen zu kapseln, wie Listing 6.4 zeigt.

```
1  public LokErgebnis LokalisiereObjekt(ObjektModell zielObjekt) {...}
2
3  public LokErgebnis LokalisiereObjektMitBehandlung(ObjektModell zielObjekt)
4  {
5    BehandlungsErgebnis ergebnis = behandlungsSteuerung.Ausführen(aktion,
6  LokalisiereObjektWrap, zielObjekt, null);
7
8    return ergebnis.outputObjekt as LokErgebnis;
9  }
```

Listing 6.3: Template einer gekapselten Lokalisierungsmethode mit Behandlung sowie der zu ersetzender Originalaufruf

```
1  public StrategieErgebnis LokalisiereObjektWrap(object inputParam)
2  {
3     ObjektModell modell = inputParam as ObjektModell;
4
5     LokErgebnis ergebnis = LokalisiereObjekt(modell);
6
7     return new StrategieErgebnis(ergebnis.Erfolg, ergebnis);
8  }
```

Listing 6.4: Kapselung der Orginialmethode hinsichtlich der definierten Schnittstelle

6.4. Statistisches Erfahrungswissen

Die erfassten Erfolgswahrscheinlichkeiten werden in Instanzen der Klassen *StrategieStatistik* bzw. *Baumknoten* gespeichert. Die erste enthält die Anzahl der bisherigen Aufrufe sowie der erzielten Erfolge und gibt die Erfolgsrate zurück, wie die Abbildung 6.3 anhand eines reduzierten Klassendiagramms zeigt. Nach einer Störungsbehandlung können beide Parameter abhängig vom Erfolg über die UpdateStatistik-Methode ggf. inkrementiert werden. Entfernt ein Bediener eine Strategie aus der Behandlung, wird das zugehörige StrategieStatistik-Objekt lediglich deaktiviert. Einmal erfasstes Erfahrungswissen steht somit nach einer Reaktivierung einer Strategie wieder zur Verfügung. Die Daten werden nur bei einer manuellen Änderung der Strategieparameter gelöscht. Die Zuordnung der jeweiligen Statistik zu den Strategien erfolgt wie bereits in Abschnitt 6.2 verdeutlicht über deren ID. Instanzen der StrategieStatistik-Klasse werden für Behandlungen vom Typ *Anwesenheitsprüfung*, *Entfernen* und *Nächstes Objekt* angewendet. Die Statistik für die Lokalisierungsstrategien wird in zwei Wahrscheinlichkeitsbäumen über Instanzen der Klasse BaumKnoten gehalten, die von *StrategieStatistik* abgeleitet ist, wie die Abbildung 6.3 verdeutlicht. Der erweiterte Variablen- und Methodensatz dient zur Erzeugung einer Baumstruktur, mit der die bedingten Abhängigkeiten der Erfolgsraten bei den Lokalisierungsstrategien abgebildet werden (siehe Abschnitt 5.6). Hierzu besitzt jeder Baumknoten eine Sammlung von Kindknoten, deren Statistik

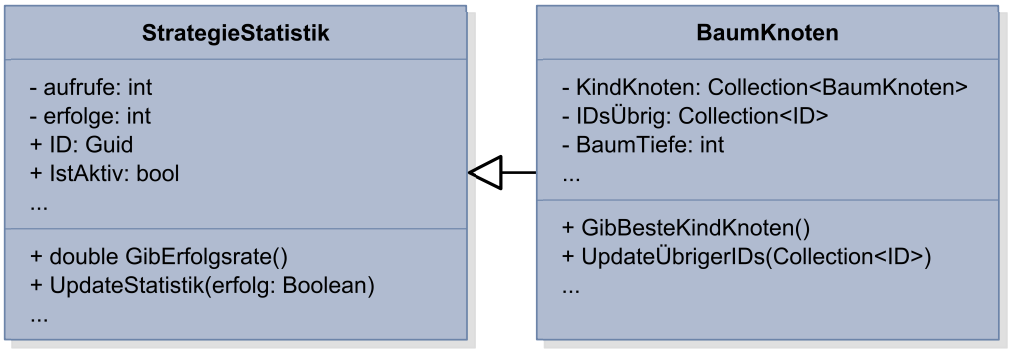


Abb. 6.3.: Klassen zur Speicherung des Erfahrungswissens

einen vorherigen Aufruf der zum Elternteil gehörenden Strategie voraussetzt. Da ein vollständig besetzter Wahrscheinlichkeitsbaum in der Regel sehr groß wäre, sich das Erfahrungswissen aber nur auf wenige Knoten verteilen würde, erfolgt die Initialisierung der Kindknoten erst bei Bedarf. Die Baumknoten verfügen hierzu über eine Liste der verbleibenden Strategien, die im Pfad ausgehend vom Wurzelknoten noch nicht enthalten sind, so dass eine mehrfache Verwendung derselben Behandlungsweise vermieden wird. Fügt der Bediener neue Strategien hinzu oder entfernt welche, werden die IDsÜbrig-Liste sowie die Kindknoten aktualisiert. Neu erstellten Baumknoten erhalten ihre initiale Erfolgsrate aus dem globalen Erfahrungswissen anhand der aktuellen Baumtiefe.

Die Struktur und Verteilung des gesamten gespeicherten Erfahrungswissens ist in Abbildung 6.4 dargestellt. Die lokale, zum aktuellen Arbeitsschritt gehörende Datenbank umfasst zum einen die beiden aus Baumknoten aufgebauten Wahrscheinlichkeitsbäume, in denen die Erfolgsraten der Lokalisierungsstrategien jeweils vor und nach einem Eingriff in die Szene enthalten sind. Zum anderen enthält die Datenbank jeweils ein Array für die StrategieStatistiken der zugewiesenen Strategien vom Typ *Anwesenheitsprüfung*, *Entfernen* sowie *Nächstes Objekt*. Bzgl. der letzten beiden wurde für die vorliegende Arbeit festgelegt, dass jeweils nur eine aktive Behandlungsmöglichkeit bei dem gegebenen Anwendungsfällen sinnvoll ist. Diese Einschränkung kann jedoch bei abweichenden Anforderungen aufgehoben werden. Die Ausführdauer der einzelnen Strategien wird in Form einer Tabelle gespeichert, in der zu jeder vorhandenen ID die Anzahl an Messwerten sowie deren akkumulierte Zeit enthalten ist.

Die Datenbank für das globale Erfahrungswissen hält alle Daten getrennt nach Arbeitsschritten, um die einzelnen Statistiken jeweils durch aktuellere Ergebnisse ersetzen zu können. Bei der Abfrage globaler Durchschnittswerte werden diese über alle gespeicherten Aktionen aus den zuvor gemittelten lokalen Werten berechnet, ohne dass die unterschiedliche Anzahl der jeweiligen Messwerte in den Arbeitsschritten das Ergebnis beeinflusst. Die im globalen Erfahrungswissen gespeicherten Datensätze der einzelnen Aktionen entsprechen den jeweiligen lokalen Daten, mit Ausnahme der Statistiken zu den Lokalisierungsstrategien. Da der Großteil der Knoten des Wahrscheinlichkeitsbaums nicht zu der auszuführenden besten Strategieabfolge beiträgt, werden nur die für den POMDP ausgewählten, erfolgversprechendsten Strategiesequenzen betrachtet. Innerhalb

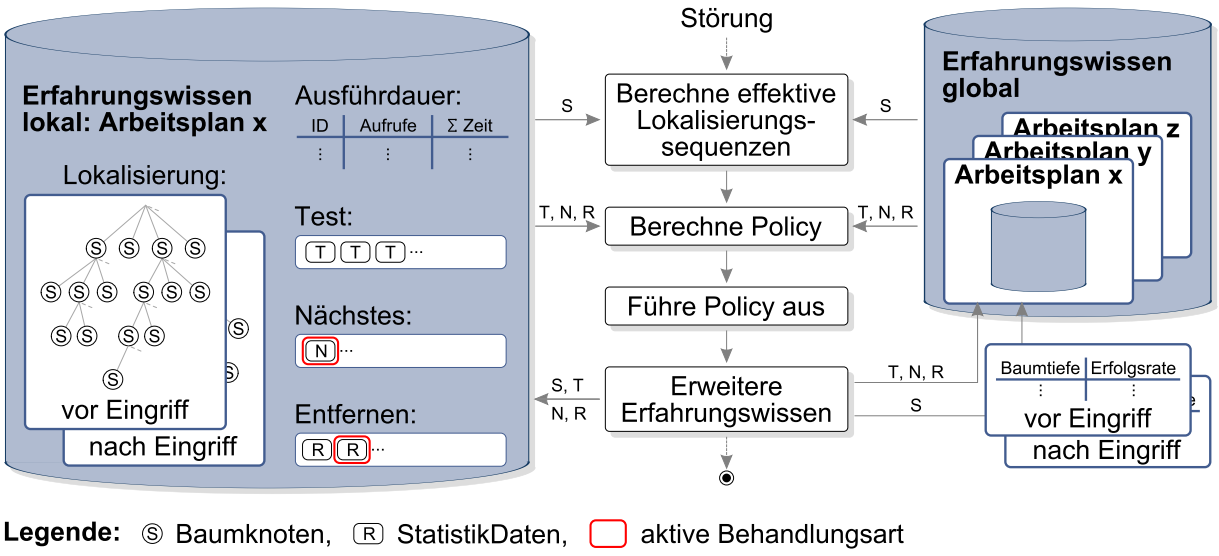


Abb. 6.4.: Verteilung des statistischen Erfahrungswissens

dieser Auswahl werden Mittelwerte abhängig von der jeweiligen Baumtiefe berechnet und gespeichert. Die Ergebnisse zeigen die durchschnittliche Erfolgsrate in der besten Lokalisierungssequenz.

Die Speicherung des Erfahrungswissens erfolgt über eine Serializer-Klasse des .NET-Frameworks, die alle Arrays, die Tabellen sowie die Wahrscheinlichkeitsbäume strukturiert in xml-Dateien überführt und entsprechend wieder laden kann.

6.5. Zusammenfassung

Die in dieser Arbeit vorgestellte MMS zur Erstellung und Optimierung von Programmabläufen für flexible Roboter erfordert verschiedene Systemkomponenten. Die Steuerung von Roboterarm, Greifer und Bildverarbeitung erfolgt über definierte Schnittstellen, die über ein Roboter-Framework bereitzustellen sind. Dieses hält auch die Arbeitspläne und Aktionen und verfügt über Funktionen zum Editieren der Parameter.

Die MMS sowie das Modul zur Störungsbehandlung sind in .NET/C# programmiert. Die Interaktion mit dem Bediener erfolgt über Bedienoberflächen, die mit der Windows-Presentation-Foundation umgesetzt sind. Die Steuerung der Störungsbehandlung kann als zusätzliche Funktion in den Programmablauf integriert werden oder eine vorhandene Methode ersetzen. In diesem Fall wird der Aufruf über Wrapper gekapselt, um die Übergabeparameter zu vereinheitlichen. Für eine einfache Erweiterbarkeit der Datenbank mit Behandlungsstrategien sind diese von einer gemeinsamen Basisklasse abgeleitet, die alle nötigen Schnittstellen vorgibt. Das Erfahrungswissen wird lokal sortiert nach dem Strategietyp gespeichert. Die bedingten Wahrscheinlichkeiten der Lokalisierungsvarianten werden in ein oder zwei Bäumen erfasst. Unterschieden wird, ob eine Lokalisierung vor oder nach einer das Objekt bewegenden Anwesenheitsprüfung ausgeführt wird. Zusätzlich zum lokalen Erfahrungswissen werden globale Daten gespeichert, die bei neuen Applikationen genutzt werden, solange noch keine ausreichenden statistischen Wahrscheinlichkeiten vorliegen.

7. Evaluierung

Die Evaluierung der in dieser Arbeit vorgestellten Verfahren und Methoden teilt sich in zwei inhaltliche Bereiche. In Abschnitt 7.2 wird die entwickelte MMS im Hinblick auf die Anforderungen nach einer Verkürzung der Inbetriebnahme von flexiblen Robotersystemen sowie einer vereinfachten Bedienbarkeit ohne Expertenwissen überprüft. In Abschnitt 7.3 folgt eine Bewertung der Störungsbehandlung im Hinblick auf die Ziele einer erhöhten Prozessrobustheit sowie einer kurzen Behandlungsdauer. Zur Evaluierung beider Aspekte wird dieselbe Versuchsplattform eingesetzt, die in Abschnitt 7.1 beschrieben wird.

7.1. Eingesetzte Versuchsplattform

Als Versuchsplattform für die Evaluierung der vorgestellten Konzepte wurde der Automatische Produktionsassistent APAS verwendet, der in Abbildung 7.1 dargestellt ist. Das flexible Robotersystem wurde bei der Firma Bosch speziell für die Kleinserienmontage entwickelt. Die Komponenten entsprechen den in Abschnitt 2.2 analysierten Anforderungen und werden nachfolgend beschrieben.

Roboterarm
Als Roboterarm wird ein Industrieroboter mit 6 Freiheitsgraden eingesetzt, der über eine Reichweite von 892mm bei einem kubischen Arbeitsraum verfügt Die maximale Bahngeschwindigkeit wurde auf 500mm/s begrenzt, um in Kombination mit einer applizieren Sicherheitsummantelung eine berührungslose Hindernisvermeidung zu erreichen. Die Schnittstelle zum Roboter bietet folgende Funktionen:

- Position auslesen: Die aktuelle TCP-Position kann in Achswinkeln, in kartesischen Roboterbasiskoordinaten oder direkt als homogene Transformationsmatrix ausgelesen werden.
- Positionen vorgeben: Die Sollposition des TCP kann in Achswinkeln oder kartesischen Basiskoordinaten vorgegeben werden. Das Anfahren kann per Punkt-zu-Punkt-Bewegung oder linear mit und ohne Überschleifen erfolgen.
- Geschwindigkeit setzen: Um die programmierten Bewegungen zu testen, kann die Bahngeschwindigkeit des Roboters manuell beschränkt werden.

Abb. 7.1.: Flexibles Robotersystem APAS

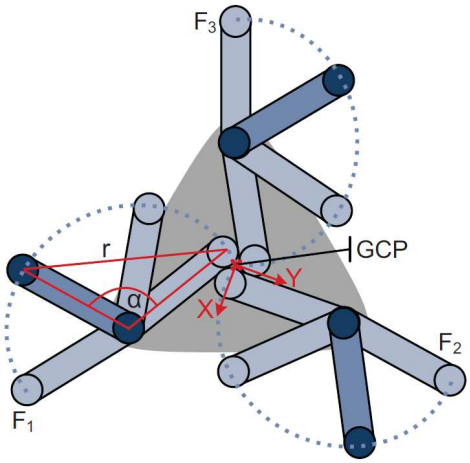


Abb. 7.2.: Greifbewegung beim 3-Finger-Sterngreifer (aus [Sayler 2011])

Greifer

Für die Bauteilmanipulation verfügt der APAS über einen 3-Finger-Sterngreifer, der dem in [Sayler 2011] beschriebenen System ohne Kraft-Momenten-Sensoren entspricht. Die Finger bewegen sich beim Öffnen und Schließen auf Kreisbahnen symmetrisch zum Greifzentrum (siehe Abbildung 7.2). Die maximale Öffnungsweite liegt bei 110 mm. Die Verwendung eines Innen- oder Außengriffs kann ebenso wie die über Motorströme geregelte Greifkraft vorgegeben werden. Die zugehörige Schnittstelle bietet folgenden Funktionsumfang:

- Öffnungsweite lesen: Zur Bestimmung der Greiferstellung ist die Öffnungsweite auslesbar
- Vorposition setzen: Der Greifer verfügt neben der Kraft- auch über eine Positionsregelung, der die Sollstellung der Greiferfinger vorgegeben werden kann.
- Kraftgeregeltes Greifen: Das Zugreifen und Festhalten von Objekten erfolgt über eine Kraftregelung. Hierzu sind neben der Greifrichtung die Soll-Motorströme vorzugeben

Kamerasysteme und Lokalisierungsalgorithmen

Zur Objektlageerkennung verfügt der APAS über ein Mono- und ein Stereokamerasystem. Beide sind am TCP des Roboters montiert und für die Hand-Auge-Koordination kalibriert. Der Tiefenschärfebereich liegt jeweils in einem Abstand zwischen 8 bis 20 cm zu den Objektiven. Die Kameraeigenschaften sind wie folgt:

Das Stereokamerasystem besteht aus zwei Kamerasensoren mit einer Auflösung von 768x576 Pixeln. Durch Objektive mit einer kurzen Brennweite umfasst der überlappende, scharfe Bildbereich abhängig von der Objektdistanz einen Durchmesser von ca. 8 bis 12 cm. Die Genauigkeit der

Lageerkennung ist bei den einzelnen Lokalisierungsverfahren jeweils ausreichend für eine fehlerfreie Bauteilmanipulation im Rahmen des beschriebenen Applikationsspektrums.

Die Monokamera verfügt bei einem Objektiv mit sehr kurzer Brennweite über einen scharfen Bildbereich von ca. 25 cm Durchmesser bei einer Auflösung von 1280x1024 Pixeln. Die Genauigkeit der Lageerkennung reicht für eine fehlerfreie Bauteilmanipulation unter den gegebenen Randbedingungen in der Regel nicht aus. Die Monokamera kann aber wegen des großen Blickfelds zur Vorpositionierung eingesetzt werden. Die erforderliche Genauigkeit wird anschließend durch eine Feinlageerkennung mit dem Stereosensor erreicht.

Die eingesetzten Objektlageerkennungsverfahren umfassen zur Abdeckung eines großen Bauteilspektrums jeweils einen Vertreter der in Abschnitt 4.4.1 beschriebenen 3D-landmarken-, 2D-kanten- und 3D-oberflächenbasierten Algorithmen. Da letzteres Verfahren eine Stereotriangulation zur Berechnung von Raumpunkten nutzt, ist es im Gegensatz zu den anderen Methoden nur mit dem Stereo- und nicht mit dem Mono-Sensor kombinierbar. Bei dem Verfahren wird zur Unterstützung der Korrespondenzpunktzuordnung ein farbcodiertes Streifenmuster auf den Bildbereich projiziert. Alle Verfahren arbeiten ansichtsbasiert im one-shot-Modus mit nur einer Objektaufnahme. Fertige 3D-Daten sind nicht erforderlich, da die Modellerstellung mit dem Robotersystem wie in Abschnitt 4.4.3 beschrieben erfolgt.

Die Schnittstelle zur Ansteuerung der Bildverarbeitungsfunktionen umfasst folgende Befehle:

- Lageerkennung ausführen: Bei erfolgreichem Aufruf gibt diese Funktion eine 6D-Roboterpose zum Ausgleich der Lagevarianz zurück, anderenfalls eine Störungsmeldung ohne Details zur Ursache.

- Objektmodell erstellen und parametrieren: Zum Einrichten der Objektmodelle sind für alle Verfahren verschiedene Funktionen zur automatischen Voreinstellung sowie zur manuellen Optimierung der Parameter vorhanden.

- Bild anzeigen: Über einen Callback können Bilder empfangen und angezeigt werden, die abhängig von der aufgerufenen Funktion automatisch erstellten werden. Das Spektrum reicht von den originalen Kamerabildern, über die Visualisierung der Objektlage im Bild bis zu aufbereiteten, intuitiv verständlichen Zwischenergebnissen der Bildverarbeitungskette.

Touchscreen zur Interaktion

Zur Anzeige der MMS und für die Interaktion mit dem Bediener ist ein 15” Touchscreen auf der Basisplattform dreh- und schwenkbar montiert. Die sichere Anwesenheit eines Menschen im Arbeitsraum des Roboters wird durch eine kapazitive Nahfeldüberwachung ermöglicht, die Hindernisse in der Nähe des Roboterarms erkennt und das System berührungslos stoppt.

7.2. Evaluierung der MMS zur intuitiven Programmierung flexibler Robotersysteme

Wie in Abschnitt 2.4.1 analysiert, bestehen die Hauptanforderungen an die MMS in einer Verringerung des Aufwands zur Inbetriebnahme sowie einer Reduktion von dessen Komplexität, um eine Programmierung und Parametrierung des Robotersystems auch ohne Expertenwissen zu ermöglichen. Zur Evaluierung der MMS wurden Usability-Tests mit mehreren Probanden durchgeführt. Abschnitt 7.2.1 beschreibt hierzu die Randbedingungen sowie die Aufgabenstellung für die Tests. Die anschließenden Abschnitte zeigen die Ergebnisse bezüglich der benötigten Inbetriebnahmedauer (Abschnitt 7.2.2) sowie der Gebrauchstauglichkeit und Akzeptanz durch die Testpersonen (Abschnitt 7.2.3).

7.2.1. Durchführung von Usability-Tests

Um die MMS hinsichtlich der benötigten Inbetriebnahmedauer, der Gebrauchstauglichkeit sowie des möglichen Verbesserungspotentials zu evaluieren, wurden im Rahmen dieser Arbeit Usability-Tests durchgeführt, bei denen acht Probanden jeweils eine Applikation nach realen Vorbild mit der in Kapitel 4 beschriebenen MMS in Betrieb genommen haben. Als Versuchsplattform diente der in Abschnitt 7.1 vorgestellte APAS.

Die Test-Applikationen

Für die Probandenversuche wurden zwei reale Applikationen ausgewählt, mit denen die unterschiedlichen Aspekte und Kombinationen von Aktionen und Objekterkennungsmethoden entsprechend der Analyse der Montageabläufe in Abschnitt 2.1.1 weitestgehend abgedeckt werden.

Die **Applikation I „Bremszylinder umsetzen“** entspricht einer reale Applikation, bei der Bremszylinder für den Weitertransport aus einer Kiste mit Blistereinsätzen auf einzelne Werkstückträger umgesetzt werden.

Der Ablauf der Montageaufgabe besteht bei der Automatisierung mit der Versuchsplattform aus drei Hauptaktionen (siehe Abbildung 7.3a). Zunächst muss die Position des Montageassistenten zum Arbeitsplatz durch Lokalisierung von angebrachten Landmarken referenziert werden (Aktion 1). Für den anschließenden Abgriff aus der Kiste, ist deren Position anhand von Landmarken mittels Übersichts- und Feinlageerkennung zu erfassen (Aktion 2). Innerhalb der Kiste befinden sich zwei Blister mit 4 x 6 Nestern, wobei nur einer der beiden Blister im Rahmen des Tests einzurichten war. Da die Lage der Bremszylinder innerhalb der labilen Kunststoffblister zur vertikalen Achse schwanken kann (siehe Abbildung 7.3f), ist eine 3D-Lageerkennung erforderlich, um die Bauteile fehlerfrei greifen und auf Werkstückträgern ablegen zu können (Aktion 3). Zwischen diesen Aktionen können Ausweichbewegungen zur Vermeidung von Kollisionen definiert werden. Da

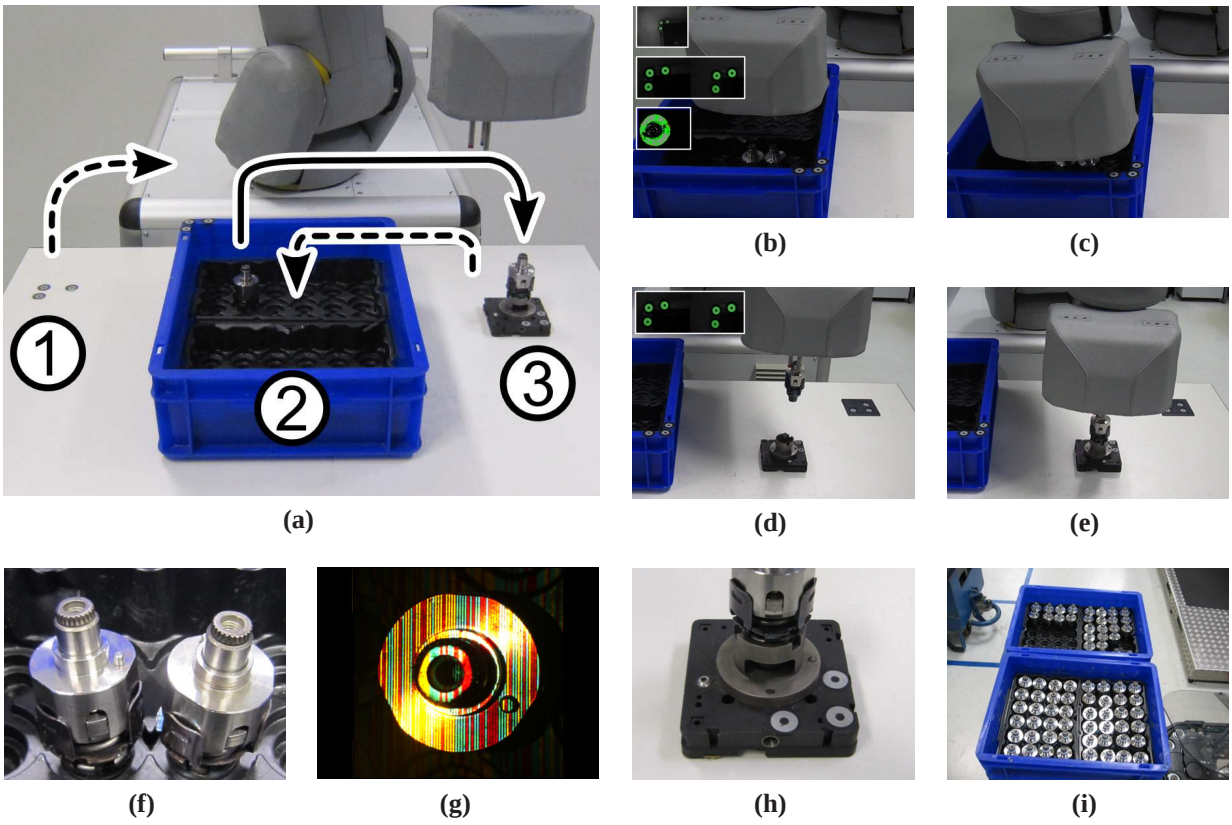


Abb. 7.3.: Applikation I „Bremszylinder umsetzen“: a) Übersicht, b) - e) Lokalisieren, Greifen und Ablegen der Bauteile, f) - i) Detailansichten von Bauteil und Werkstückträger

diese Aktionen abhängig von den jeweils gewählten An- und Abrückpfaden nicht von allen Probanden verwendet wurden, werden sie für den Ergebnisvergleich mit den jeweils nachfolgenden Hauptaktionen zusammengefasst.

Die Tabelle 7.1 zeigt Details zu den einzelnen Schritten. Die Komplexität beim Programmie-

Aktion		Teilschritte	Spiel / Lagevarianz	Griff / Ablage	besondere Schwierigkeit
1	Referenzierung am Arbeitsplatz	· Marker-Fit (fein)	-	-	-
2	Bremszylinder aus Kiste greifen	· Marker-Fit (grob) · Marker-Fit (fein) · Oberflächen-Fit (Bauteillage) · Blister-Muster · Bauteilabgriff	< 10 cm < 0,5 cm < 1 cm < 3 ° - < 1 mm	zentrierter 3-Finger-Griff 	· Parametrierung des Oberfl.-Fits für alle Nester · Kollisionsfreier Abgriff in Palette
3	Bauteil ablegen auf Werkstückträger	· Marker-Fit (fein) · Bauteilablage	< 1 cm < 2 mm	Hole on Peg 	-
4	Rücksprung	-	-	-	-

Tab. 7.1.: Details zu den Einzelschritten in Applikation I „Bremszylinder umsetzen“

ren dieser Applikation besteht vor allem im robusten Einrichten der Objektlokalisierungen, sowie dem fehlerfreien Bauteilhandling bei allen Blisternestern. Dabei ist auf Kollisionen mit den hohen Rändern der Kiste zu achten

Die **Applikation II „Ventilnadel palettieren“** ist ebenfalls an eine reale Applikation angelehnt. Bei dieser werden Ventilnadeln am Ende eines Schleifprozesses aus einer Bearbeitungsmaschine auf einen Ablagebereich befördert und sind für die Weiterverarbeitung in Waschrahmen abzustecken. Beim Einrichten der Applikation durch die Probanden ist zur Referenzierung des Montagesystems zuerst eine Lokalisierung des Arbeitsplatzes mittels Landmarken durchzuführen (Aktion 1 in Abbildung 7.4a). Die Bauteillage auf der Ablagefläche ist über ein kantenbasiertes Matching zu erfassen (Aktion 2). Nach dem Abgreifen ist die Nadel über eine ortsfeste Vorrichtung umzugreifen (Aktion 3) und in eine über Landmarken zu lokalisierende Palette abzustecken (Aktion 4).

Die Tabelle 7.2 zeigt Details zu den einzelnen Schritten. Die Komplexität beim Einrichten dieser Applikation besteht im robusten Parametrieren der Bauteillageerkennung sowie der Bauteilhandhabung mit ausreichender Genauigkeit. Die Palettennester weisen ein Spiel von 0,5 mm bzw. 1° zur Nadel auf. Bei größeren Abweichungen kann es wegen der beiden in Abbildung 7.4h gezeigten Führungslöcher in den Palettennestern zu Verklemmungen oder Beschädigungen kommen.

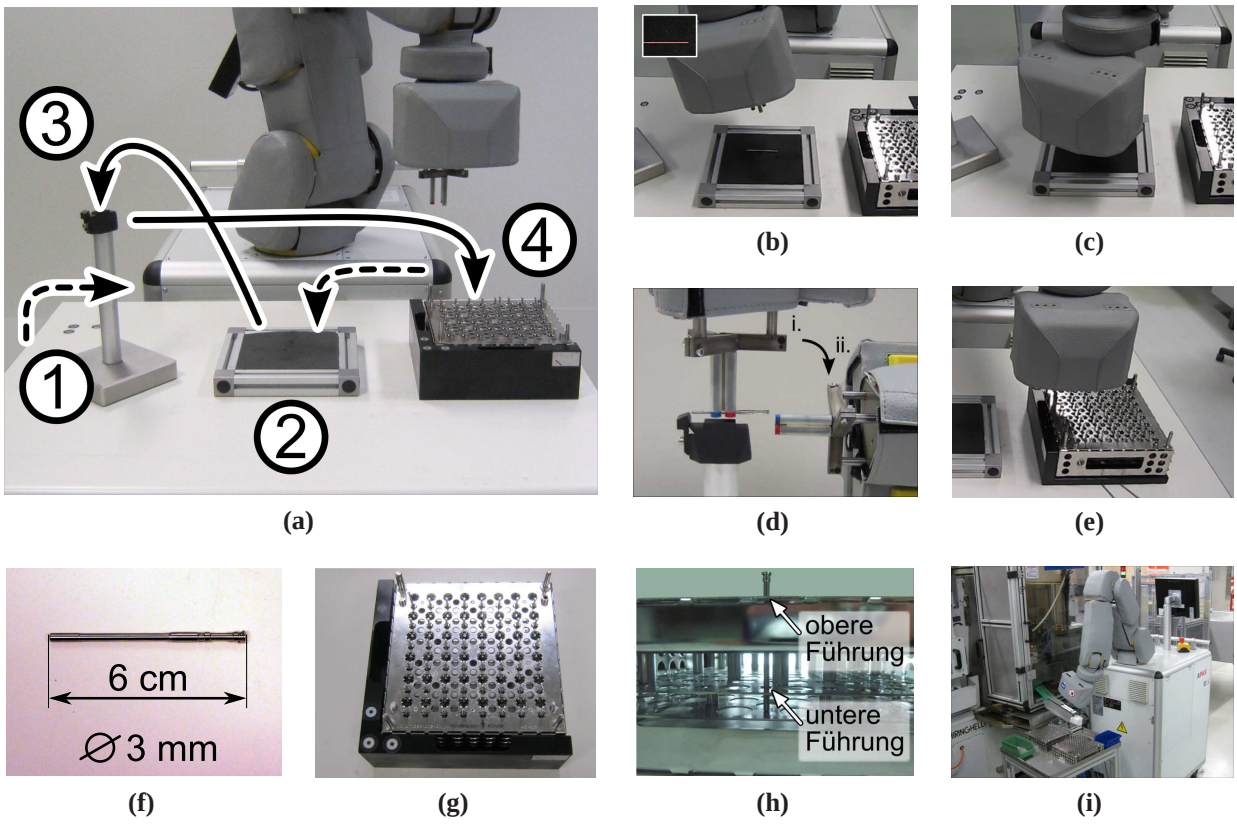
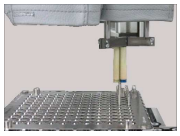
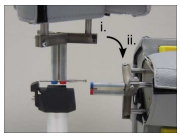


Abb. 7.4.: Applikation II „Ventilnadel palettieren“: a) Übersicht, b) - e) Lokalisieren, Greifen, Umgreifen und Abstecken des Bauteils, f) - h) Detailansichten von Werkstück und Palette, i) Applikation im Werk

Aktion		Teilschritte	Spiel / Lagevarianz	Handhabung	besondere Schwierigkeit
1	Referenzierung am Arbeitsplatz	· Marker-Fit (fein)	-	-	-
2	Ventilnadel von Fläche greifen	· Kanten-Fit (Bauteillage) · Bauteilabgriff	< 0,8 cm < 2 mm		· präzise Bauteillageerkennung · Abgriff ohne Verschiebung
3	Umgreifen über Vorrichtung	· Feste Position · Bauteilablage · Bauteilabgriff	 < 1 mm < 1 mm		· fehlerfreie Handhabung · kollisionsfreie Umorientierung
4	Ventilnadel in Palette abstecken	· Marker-Fit (fein) · Palettenmuster Bauteilablage	< 1,5 cm < 0,5 mm < 1 °		· Geringes Spiel d. Palettennester bzgl. Position und Orientierung
5	Rücksprung	-	-	-	-

Tab. 7.2.: Details zu den Einzelschritten in Applikation II „Ventilnadel palettieren“

Die Probanden

Für die Usability-Tests wurden Probanden aus zwei Personengruppen ausgewählt. In Gruppe I waren vier Einsteller aus zwei Werken der Robert Bosch GmbH. Sie waren im Anschluss für die eigenständige Inbetriebnahme und Betreuung zweier Prototypen des APAS in der Produktion zuständig und entsprechen genau der Zielgruppe (siehe Abschnitt 2.2.2). Die anderen vier Teilnehmer sind Mitarbeiter aus der Forschung und Vorausentwicklung. Sie hatten ebenfalls keine Vorkenntnisse oder Erfahrungen mit der MMS oder herkömmlicher Roboterprogrammierung. Um aus dem Usability-Test realistische Ergebnisse zu erhalten, bekamen die Probanden eine Einführung zum System. Die Einsteller erhielten eine eintägige Schulung zur MMS, wie sie bei der Anschaffung derartiger Systeme vorgesehen wäre. Sie umfasste Erklärungen in Präsentationsform, Demonstrationen sowie kleine, gemeinschaftliche Übungen am System. Im Usability-Test war das Umsetzen des Bremszylinders einzurichten. Gruppe II erhielt aus Zeitgründen lediglich Erläuterungen und Demonstrationen im Gesamtumfang von drei Stunden. Für den Versuch wurde die Applikation II „Ventilnadel palettieren“ verwendet.

Um möglichst realistische Ergebnisse zu erzielen, waren die Probanden angehalten die Applikation vollständig, selbstständig und gemäß den Anforderungen in der Produktion einzurichten. Um die Teilnehmer dabei zu einem umfassenden und ehrlichen Feedback zu ermuntern, wurde ihnen versichert, dass die erhobenen Daten und Ergebnisse später nicht auf ihre Person zurückzuführen seien und sich für sie kein Nachteil ergeben könne. Zusätzlich wurden die Probanden während der Durchführung weitestgehend von äußeren Störeinflüssen, wie z. B. der Beobachtung durch Kollegen, abgeschirmt.

Die Durchführung der Tests wurde zur Dokumentation evtl. auftretender Schwierigkeiten, Auf-

fälligkeiten oder dem geäußerten Probanden-Feedback von einem Versuchsleiter betreut. Hilfestellungen wurden nur gegeben, wenn der Versuch anderenfalls nicht fortgesetzt werden konnte. Für die Überprüfung, ob eigene Fehler erkannt und korrigiert werden können, wurden eventuelle fehlerhafte Eingaben oder Parametrierungen während der Durchführung nur erfasst und erst im Anschluss besprochen.

7.2.2. Evaluierung der Inbetriebnahmedauer

Um die Anforderungen hinsichtlich einer Verkürzung der Inbetriebnahmedauer zu evaluieren, wurden während der Durchführung der Usability-Tests alle Benutzereingaben automatisch inklusive eines Zeitstempels geloggt. Die Abbildung 7.5 zeigt die benötigte Dauer für die Hauptaktionen der Applikation I für alle 4 Probanden. Optionale „Arm bewegen“-Arbeitsschritte zur Vermeidung von Kollisionen sind zur besseren Vergleichbarkeit der Probandendaten mit der nachfolgenden Aktion zusammengefasst. Die Tabelle 7.3 gibt jeweils die mittlere sowie die minimal und maximal benötigte Dauer für die Aktionen an. Gezeigt ist die komplette, zur Parametrierung eines Schritts verwendete Zeit jeweils inklusive eventueller Tests und Überlegungspausen der Probanden. Da der Arbeitsplan ebenfalls getestet wurde, übersteigt die Gesamtdauer der Applikation die Summe der Einzelaktionen.

Die Tabelle 7.3 zeigt, dass die Inbetriebnahmedauer für die gesamte Applikation bei allen vier Testpersonen zwischen 1,5 Stunden und knapp 2 Stunden lag. Nach Einschätzungen eines Roboterexperten der Firma Bosch würde die Programmierung dieses Ablaufs mit herkömmlichen Me-

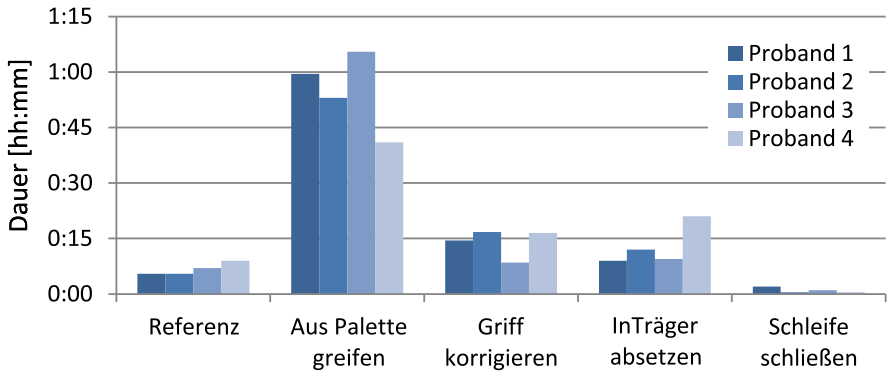


Abb. 7.5.: Erreichte Programmierdauer im Vergleich für Applikation I „Bremszylinder umsetzen“

Zeit: Aktion: [mm:ss]	Referenz	Aus Palette greifen	Griff korrigieren	In Träger absetzen	Schleife schließen	Gesamt
Mittel:	06:45	54:45	14:04	12:52	01:00	96:37
Min:	05:30	41:00	08:30	09:00	00:30	92:00
Max:	09:00	65:30	16:45	21:00	02:00	107:00
Standardabw.:	01:46	12:17	04:12	06:07	00:46	07:41

Tab. 7.3.: Übersicht über die Programmierdauer für Applikation I „Bremszylinder umsetzen“

thoden ca. 4-5 Arbeitstage dauern. Dies entspricht einer Aufwandsreduktion von über 90% bei unerfahrenen Bedienern ohne vorherige Kenntnisse der Roboterprogrammierung.

An Abbildung 7.5 und Tabelle 7.3 lässt sich außerdem erkennen, dass die Probanden jeweils ähnlich lange für die Programmierung der einzelnen Aktionen benötigten. Dies deutet darauf hin, dass die erforderliche Zeit im Wesentlichen personenunabhängig ist. Für verlässliche Aussagen sind mehr Probanden zu prüfen. Die schwankende Einrichtzeit zwischen den einzelnen Aktionen lässt sich durch deren unterschiedlichen Umfang erklären. So ist das „Greifen aus der Palette“ mit drei enthaltenen Objektlageerkennungen, dem Definieren des Palettenrasters und dem genauen Bauteilabgriff deutlich aufwändiger als die Referenzierung oder das Ablegen auf dem Werkstückträger.

Beim Einrichten des Bauteilabgriffs hat sich während der Versuchsdurchführung gezeigt, dass zwei wichtige Einschränkungen für das Erstellen eines funktionstüchtigen Ablaufs nicht aus den Informationen in der MMS ersichtlich waren. Dies betraf zum einen die Objektlageerkennung des Zylinders mittels Oberflächen-Fit. Zur Reduktion der rekonstruierten Punktewolke werden für die Lageerkennung nur Raumpunkte betrachtet, die innerhalb eines parallel zur Bauteiloberfläche verlaufenden Schlauchs liegen. Dessen Lage wird nur an einer Palettenposition anhand ebener Bauteiloberflächen definiert und auf die übrige Palette übertragen. Der entsprechende Bauteilbereich muss beim Erfassen daher möglichst waagerecht stehen, um an allen Palettenpositionen ungefähr dieselbe Höhe zu markieren. Diese Einschränkung wurde weder in der Einführung noch innerhalb der MMS thematisiert und wurde von keinem Probanden berücksichtigt. Ebenso war nicht ersichtlich, dass die drei Eckpositionen der Palette mit gleicher Orientierung des Greifers definiert werden müssen. Die Nichtbeachtung beider Einschränkungen führt zu unerwartetem Systemverhalten und verhindert einen funktionsfähigen Ablauf. Daher wurden die Probanden vom Versuchsleiter darüber aufgeklärt, sobald erkennbar war, dass sie das Problem ohne Hinweis nicht beheben können. Die Korrektur erfolgte wieder selbstständig. Die dazu benötigte Dauer ist in Abbildung 7.5 und Tabelle 7.3 separat dargestellt, da sich die entsprechende Zeit nach einer Anpassung der MMS einsparen lassen sollte.

Die Abbildung 7.6 sowie die Tabelle 7.4 zeigen die Ergebnisse für Applikation II. Man erkennt, dass die benötigte Parametrierdauer für die gesamte Applikation mit 2 bis 2,45 Stunden ebenfalls sehr gering ist. Die Programmierung mit herkömmlichen Methoden wurde von einem Roboterexperten der Firma Bosch mit ca. 5 Arbeitstagen geschätzt. Zusätzlich wurde eine sehr ähnliche Applikation mit diesem Bauteil und der Palette bereits von einem Mitarbeiter mit fortgeschrittenem Kenntnisstand direkt in C# programmiert. Hierzu wurde das Steuerungs-Framework des Systems mit allen spezifischen Methoden verwendet und der Roboter über das Handbediengerät verfahren. Diese Art der Inbetriebnahme erforderte drei vollständige Arbeitstage bis zur fehlerfreien Ausführung und damit ebenfalls deutlich länger als in den Usability-Tests.

Im Vergleich zu Applikation I benötigten die einzelnen Aktionen eine einheitlichere Programmierdauer, da ihr Umfang ausgeglichener ist. So enthält der Abgriff die Parametrierung der Objektlageerkennung mittels kantenbasiertem Fit, während das Umgreifen zwei Aktionen ohne Lage-

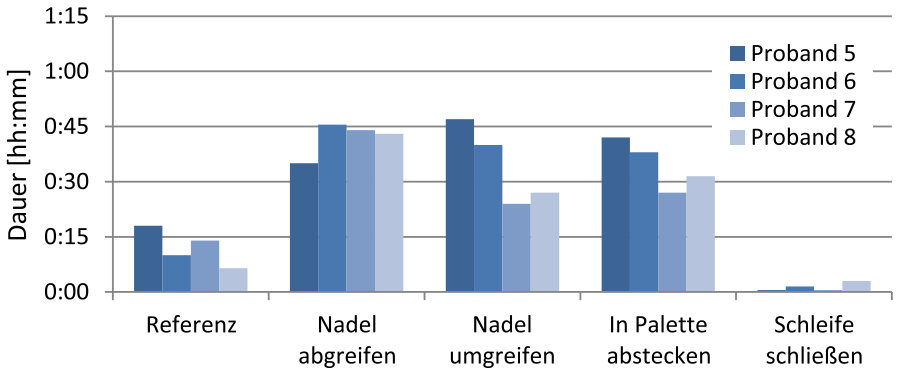


Abb. 7.6.: Erreichte Programmierdauer im Vergleich für Applikation II „Ventilnadel palettieren“

Zeit: [mm:ss] Aktion:	Referenz	Nadel abgreifen	Nadel umgreifen	In Palette abstecken	Schleife schließen	Gesamt
Mittel:	12:07	41:52	34:30	34:37	01:23	138:22
Min:	06:30	35:00	24:00	27:00	00:30	114:00
Max:	18:00	45:30	47:00	42:00	03:00	163:00
Standardabw.:	04:58	04:42	10:51	06:41	01:11	22:22

Tab. 7.4.: Übersicht über die Programmierdauer für Applikation II „Ventilnadel palettieren“

erkennung entspricht und beim Ablegen zusätzlich das Palettenmuster zu definieren ist. Gleichzeitig sind alle Manipulationen aufgrund der geringen Toleranzen sehr genau auszuführen. Die hohe Komplexität bei den Bewegungen könnte ein Grund sein, warum die Inbetriebnahmedauer insgesamt höher liegt als bei Applikation I.

Im Vergleich fällt ebenfalls auf, dass die Zeiten der Probanden bei Applikation II zueinander deutlich mehr variieren. Anhand der erfassten Daten lässt sich erkennen, dass Proband 5 beim Einrichten der kantenbasierten Objektlageerkennung deutlich schneller war als die übrigen Teilnehmer. Die Probanden 5 und 6 benötigten für die Positionierung des Roboterarms speziell beim Umgreifen und Palettieren länger. Ein möglicher Grund könnte neben persönlichen Unterschieden auch die kürzere Schulung ohne aktive Übungen sein. Die Probanden aus der zweiten Gruppe haben den Roboter während des Versuchs das erste Mal selbst gesteuert und gaben z. T. an, dass hierzu eine Übung im Vorfeld sinnvoll sei. Ein aussagekräftiger Vergleich dieser Probanden mit denen von Applikation I ist aufgrund der unterschiedlichen Aufgaben nicht möglich. Einzig die Referenz ist in beiden Applikationen enthalten und wird von der Gruppe mit längerer Schulung schneller parametriert. Statistisch relevante Aussagen hierzu sind bei dieser Stichprobengröße nicht sinnvoll. Bei weiteren Schulungen zu flexiblen Robotersystemen werden aktive Übungen empfohlen.

Um mögliches Potential für eine weitere Aufwandsreduktion beim Einrichten neuer Applikationen zu identifizieren, wurden die einzelnen Bestandteile der Inbetriebnahme detaillierter betrachtet. Die Analyse der zur Inbetriebnahme der Objektlageerkennung benötigten Zeiten verdeutlicht, dass speziell die Lokalisierung mittels Landmarken schnell, innerhalb von 2 bis 4 Minuten einzurichten ist, wie Tabelle 7.5 zeigt. Sowohl die oberflächen- als auch die kantenbasierte Erkennung bedürfen

Lokalisierungs-verfahren		Mittlere Einrichtdauer [min:sec] bei: Applikation I		Applikation II		∅ /1x
Marker	Grob	1x:	03:45	-		03:45
	Fein	3x:	01:56	2x:	03:23	02:31
Oberflächen-Fit		1x:	20:04	-		20:04
Korrektur			06:22			
Kanten-Fit		-		1x:	14:37	14:37
∑ Objektlage-erkennung			40:15		28:31	

Tab. 7.5.: Durchschnittliche Dauer zum Einrichten der Objektlageerkennungen

mit 15 bis 20 Minuten deutlich länger und bieten daher mehr Potential für größere Zeiteinsparungen. Da diese Verfahren jedoch deutlich aufwändiger sind, die Probanden die Parametrierung zum ersten bzw. zweiten Mal durchgeführt haben und Übungseffekte zu erwarten sind, werden die Ergebnisse als sehr vielversprechend eingestuft. Die korrekte Inbetriebnahme der Lokalisierungen durch Probanden ohne Expertenwissen wird als Erfolg bewertet.

Die Tabelle 7.6 stellt den Anteil der einzelnen Komponenten bei der Inbetriebnahme zueinander und abhängig von der insgesamt zur Programmierung benötigten Zeit dar. Angegeben ist neben der Dauer für das Einrichten der Objektlageerkennung auch die Zeit, die dem Positionieren des Roboters zuzuordnen ist. Die Angaben für Parametrierung, Test und Überlegungen entsprechen der verbleibenden Zeit. Man erkennt, dass trotz der kurzen Wege bei beiden Applikationen relativ viel Aufwand für das Einrichten der Roboterbewegungen notwendig war. Zu berücksichtigen ist, dass aufgrund der Toleranzen eine hohe Genauigkeit erforderlich war. Dennoch wird an dieser Stelle das größte Potential für weitere Zeiteinsparungen erwartet. Die vorgestellten Jog-Controls ermöglichen es auch unerfahrenen Benutzern, den Roboter wie gewünscht entsprechend der vorhandenen Anforderungen zu platzieren. Speziell das Verkippen des Greifers um die X- und Y-Achse des Roboters schien jedoch teilweise aufwändig und nicht vollständig intuitiv zu sein.

Bestandteil der Programmierung	Absoluter [min:sec] und relativer Anteil an der Gesamtdauer bei: Applikation I		Applikation II	
Objektlage-erkennung	40:15	41,7 %	28:29	20,6 %
Roboter-positionierung	40:37	42,0 %	68:30	49,5 %
Parametr., Tests, Überlegungen …	15:45	16,3 %	41:23	29,9 %
Gesamtdauer	96:37		138:22	

Tab. 7.6.: Absolute und relative Zeitanteile der einzelnen Programmierbestandteile

7.2.3. Evaluierung der Gebrauchstauglichkeit und Akzeptanz

Zur Reduktion des Inbetriebnahmeaufwands ist neben der erforderlichen Zeitdauer zunächst nur relevant, ob die Programmierung ohne Expertenwissen durchführbar ist, bzw. welche Probleme ein Nutzer nicht selbstständig lösen kann. Dennoch haben auch eine intuitive Bedienung sowie die Akzeptanz einen direkten Einfluss auf die Effizienz der Nutzer. Beide Eigenschaften sind stark subjektiv und erschweren eine allgemeingültige, unabhängige Bewertung. Zur Identifikation von Verbesserungspotential für künftige Weiterentwicklungen der MMS wurden im Rahmen der durchgeführten Usability-Tests die folgenden drei Methoden eingesetzt, um die Akzeptanz und Intuitivität der Bedienung zu bewerten:

- Feedback durch Probanden + Beobachtung durch Versuchsleiter
- Halbstrukturiertes Interview
- Fragebogen

Während der Inbetriebnahme wurden die Probanden ermutigt, über die Methode des lauten Denkens möglichst viel Feedback über die MMS und ihr Verständnis zu geben. Zusätzlich wurden sichtbare Auffälligkeiten, wie längeres Zögern der Testpersonen, fehlerhafte Parametrierungen sowie wiederholte oder unnötige Eingaben und Abläufe von einem Versuchsleiter erfasst. Die mehrfach auftretenden Beobachtungen und Äußerungen sind in die Kategorien „Generelles zur Bedienung“ sowie „Nötige Hinweise und Zusatzfunktionen“ eingeteilt. Die Tabelle 7.7 zeigt die wichtigsten Ergebnisse bezüglich der generellen Bedienung sowie die Häufigkeit des Auftretens innerhalb der Probandengruppe.

Ein sehr positives Ergebnis zur generellen Bedienung folgt aus der Beobachtung sowie den Aussagen, dass 7 von 8 Probanden die Hinweistexte bei erwartetem Systemverhalten nicht gelesen haben. Die Bedienung scheint dennoch hauptsächlich intuitiv erfolgt zu sein. Als Folge sollten wichtige Hinweise durch verständliche, auf einen Blick erfassbare Bilder oder Animationen dargestellt werden. Die intuitive Verwendung der Steuerelemente zeigte sich an anderer Stelle besonders

Beobachtung, Probanden-Feedback	Auftreten
- Bedienung erfolgt zumeist intuitiv, Texte werden nahezu nicht gelesen	7/8
- Optionales Anzeigen mehrfach verwendeter Steuerelemente verwirrt	8/8
- Unzureichendes Feedback: beim Speichern von Pfadpunkten, beim Verfahren kurzer Wege, über den Abschluss der autom. Parametrierung	6/8
- Jog-Controls: Prellen der Buttons führt zu ungewollten Bewegungen	5/8
- Jog-Controls: Drücken einer Taste mit Blick auf Greifer misslingt häufig	8/8
- Jog-Controls: Unterschiedliche Bewegungsweite durch ° und cm irritiert	4/8
- Probanden wissen, wie Fehlparametrierungen zu beheben sind	8/8
- Zusatzwizard zur Auswahl der Bauteilerkennung wurde verwendet	1/8
- „Zuordnung der farbigen Greiferfinger zu Jog-Control gut gelungen.“	2/8
- „Inbetriebnahme macht Spaß. Robotersteuerung etwas aufwändig.“	2/8

Tab. 7.7.: Beobachtungen und Feedback zur generellen Bedienung der MMS

stark, an der die Nutzer zur Platzierung der Landmarken an den zu lokalisierenden Paletten aufgefordert wurden. Um dabei die Position der Markierungen auf Wunsch mit den Kameras testen zu können, waren die Jog-Steuerungen zur Ausrichtung des Roboters angezeigt. Im Test war dieser Schritt nicht nötig, denn die Landmarken waren bereits angebracht. Da die Hinweistexte nicht gelesen wurden, verstanden alle Probanden die Anzeige der Steuerungen als Aufforderung, die Kamera auszurichten. Entsprechend waren sie irritiert, als dieser Schritt später ggf. „erneut“ durchzuführen war. Als Erkenntnis daraus sollten optionale Elemente, die mehrfach im Ablauf verwendet werden, erst auf Wunsch über eine Schaltfläche zugänglich gemacht werden. Weitere Aussagen und Beobachtungen zur Bedienung betrafen vor allem das Feedback des Systems sowie Details zu den Jog-Controls.

Die Tabelle 7.8 zeigt die wichtigsten Beobachtungen und Anmerkungen bezüglich sinnvoller Zusatzfunktionen und Hinweisen an den Bediener. Besonders relevant sind die ersten beiden Punkte, da es sich um Einschränkungen des Systems handelt, die aus der MMS nicht ersichtlich waren, wie in Abschnitt 7.2.2 erläutert. Die übrigen Auffälligkeiten betreffen vor allem die Parametrierung der Objektmodelle und wirkten sich weniger kritisch aus. Eine Berücksichtigung in Form von Hinweisen oder Anpassungen der Bedienelemente sollte die Inbetriebnahme aber vereinfachen und Zeit einsparen.

Die Ergebnisse stellen vor allem ein qualitatives Feedback dar, das durch die Zuordnung zu konkreten Bestandteilen der MMS direkt für Verbesserungen verwendet werden kann. Durch die schwer zu gewichtenden und z. T. auch entgegengesetzten Äußerungen und Beobachtungen bei den einzelnen Probanden ist eine Wertung und Akzeptanzbestimmung aus diesen Daten nur bedingt möglich. Aus diesem Grund wurden die Probanden nach der Inbetriebnahme um weiteres Feedback gebeten. In einem halbstrukturierten Interview wurde nach besonders positiven und negativen Aspekten der MMS sowie nach Verbesserungsmöglichkeiten gefragt. Da die Antworten ohne Vorgaben gegeben wurden, werden die Nennungen als besonders relevant bewertet, da sie in Erinnerung geblieben sind. Die Tabelle 7.9 zeigt alle mehrfach genannten Aspekte. Die gestellten Fragen sind im Anhang unter C.3 aufgelistet.

Hinweise und Zusatzfunktionen	Auftreten
- Fehlender Hinweis auf Limitierung: die Definition des ’Schlauchs’ beim oberflächenbasierten Fit gilt für alle Positionen in einer Palette	4/4
- Fehlender Hinweis auf Limitierung: alle Eckpositionen einer Palette sind mit derselben Greiferorientierung vorzugeben	4/4
- Hinweis auf geeigneten Kameraabstand zum Objekt fehlt beim Ausrichten	8/8
- Vorschlag: Parametrierfunktionen für Objektmodelle chronol. nummerieren	4/8
- Bedienerfehler: Objekt nach Lokalisierung manuell bewegt (wiederholen)	2/8
- Beobachtung: roboterbas. Jog-Steuerung für Feinpositionierung verwendet. Problem: Drehung um Achse 1 verändert Greiferlage gegenüber Z-Achse	4/8

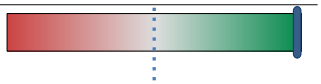
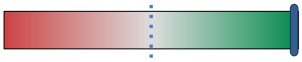
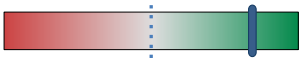
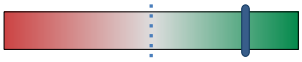
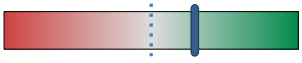
Tab. 7.8.: Beobachtungen zu Limitierungen und Verbesserungen der MMS

Nennung	Häufigkeit
Positive Nennungen	
- Wizardkonzept leitet Bediener gut	7x
- MMS ist selbsterklärend und übersichtlich	4x, 3x
- Status-Icons helfen bei Orientierung und Wiedereinstieg	2x
- Greifer mit farbigen Fingern einfach steuerbar	2x
Negative Nennungen	
- Bedienelement zur manuellen Modellparametrierung ist umständlich	4x
- Roboterbasierte Jog-Steuerung ist kompliziert oder erfordert Übung	2x
- Hinweis auf erforderlichen Kameraabstand fehlt	2x
Mögliche Verbesserungen	
- Tragbares Bediengerät für Robotersteuerung	4x
- Mehr und verständlichere Hinweisbilder	2x

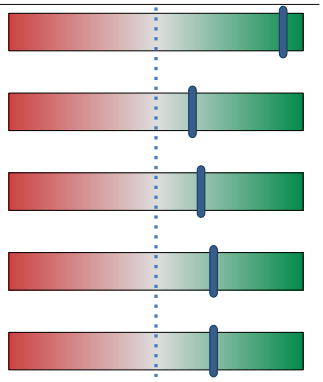
Tab. 7.9.: Nennungen der Probanden zu positiven und negativen Aspekten sowie möglichen Verbesserungen der MMS

Es ist zu erkennen, dass insgesamt mehr positive Aspekte genannt wurden. Dabei werden die MMS insgesamt sowie auch einzelne Elemente der Bedienerführung hervorgehoben. Optimierungsbedarf bzw. Mängel werden von den Probanden hauptsächlich im Bereich der roboterbasierten Steuerung und dem Bedienelement zu Parametrierung der Objektmodelle gesehen, das aus Platzgründen häufig gescrollt werden musste.

Nach dem Interview sollte zusätzlich die Zustimmung zu 15 Aussagen zur MMS bewertet werden. Die 5-stufige Skala war über die verbalen Marken „stimme nicht zu“, „stimme wenig zu“, „teils, teils“, „stimme eher zu“ und „stimme voll zu“ eingeteilt. Die genauen Fragen sind im Anhang unter C.1 aufgelistet. Es gab jeweils fünf Aussagen zur Bedienerführung, der Einrichtung der Objektlageerkennung sowie der Steuerung des Roboterarms. Die Tabellen 7.10 bis 7.12 zeigen die mittlere Bewertung (links: stimme nicht zu; rechts: stimme voll zu). Zu beachten ist, dass die Aussagen 2 und 5 zur Objektlageerkennung sowie Aussage 4 zur Robotersteuerung im Fragebogen invertiert

Aussage	Bewertung
Die Unterteilung der Applikation in einen Arbeitsplan mit einzelnen Arbeitsschritten finde ich verständlich	
Ich konnte mich innerhalb der Arbeitsplandarstellung gut orientieren und die Arbeitsschritte der Applikation zuordnen.	
Das Einrichten der Arbeitsschritte in Form von Wizards mit mehreren Schritten bzw. Seiten finde ich verständlich.	
Ich konnte die einzelnen Parametrierschritte im Zusammenhang des Arbeitsschritts einordnen.	
Ich wusste stets, wo Eingaben erforderlich sind und wie der Programmiervorgang fortzuführen war.	

Tab. 7.10.: Bewertung der Bedienerführung durch die Probanden. Bewertung rot/links: trifft gar nicht zu; grün/rechts: trifft voll zu

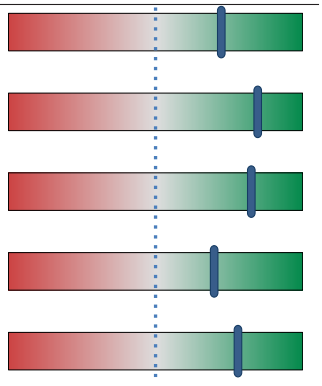
Aussage	Bewertung
Die Fragen zur Bauteilzuführung waren mit der vorhandenen Unterstützung gut zu beantworten.	
Ich würde die Abfolge der Erkennungsschritte zur Lageerkennung lieber nicht direkt auswählen.	
Beim Parametrieren der Objektmodelle fand ich es verständlich, was in den einzelnen Schritten zu tun ist.	
Die Ergebnisbilder, in denen die Auswirkung der einzelnen Parameter dargestellt wurde, fand ich verständlich.	
Das Einrichten der Objektmodell ist mir nicht schwer gefallen.	

Tab. 7.11.: Bewertung zur Positionierung des Roboters durch die Probanden. Bewertung rot/links: trifft gar nicht zu; grün/rechts: trifft voll zu

gestellt wurden, um ein konzentriertes Lesen der Texte zu bewirken. Für die nachträgliche Darstellung wurden die entsprechenden Aussagen und Ergebnisse negiert.

Die Bewertungen bestätigen die Ergebnisse des Interviews und der Beobachtungen. Die Bedienerführung wurde sehr positiv bewertet. Die geringere Zustimmung bei der letzten Frage in Tabelle 7.10 könnte hauptsächlich auf die genannten Defizite und Verbesserungsmöglichkeiten bei der Parametrierung der Objektlageerkennung zurückzuführen sein. Hierfür sprechen auch die Bewertungen zur Parametrierung der Objektmodelle beim Einrichten der Lageerkennung. Die Beantwortung der Fragen zur Bauteilzuführung scheint hingegen sehr verständlich. Von der direkten Auswahl der Schritte ist daher abzusehen, da sich in frühen Probandenversuchen gezeigt hat, dass es hierbei zu Fehlern und Verwechslungen kommt.

Bei der Bewertung zur Steuerung des Roboterarms ist zu erkennen, dass die roboterbasierte Jog-Steuerung etwas schwieriger zu bedienen ist und das Verfahren des Roboters insgesamt optimiert werden kann. Dies deckt sich mit den Ergebnissen aus dem Interview und den Beobachtungen.

Aussage	Bewertung
Ich konnte den Roboter über die „roboterbasierte“ Bewegungssteuerung gut über weite Strecken an die Zielposition verfahren.	
Ich konnte die Kamera über die „kamerabasierte“ Steuerung gut zum gewünschten Sichtbereich verfahren.	
Ich konnte den Greifer über die „greiferbasierte“ Bewegungssteuerung gut auf das Bauteil oder die Ablageposition ausrichten.	
Das Verfahren des Roboters war nicht kompliziert und erforderte nicht viel Überlegung.	
Das Einlernen ganzer Pfade über Einzelpunkte fand ich einfach.	

Tab. 7.12.: Bewertung zum Einrichten der Objektlageerkennung durch die Probanden. Bewertung rot/links: trifft gar nicht zu; grün/rechts: trifft voll zu

7.3. Evaluierung der strategiebasierten Störungsbehandlung

Zur Evaluierung der strategiebasierten Störungsbehandlung werden Anwendungsfälle definiert, die unterschiedliche Kombinationen der Fehler- und Störungsklassen in Verbindung mit den verschiedenen Objektlageerkennungsalgorithmen beinhalten. Eine erschöpfende Überprüfung aller Kombinationen ist aufgrund der hohen Vielfalt der Montageapplikationen im Rahmen dieser Arbeit nicht möglich. Die Evaluierung beschränkt sich daher auf die wichtigsten und häufigsten Fehler und Störungen. In der Beschreibung der Testszenarien wird auf zusätzliche Varianten zur Behandlung von ähnlichen Fehlerfällen hingewiesen. Die ausgewählten Applikationen stellen reale Produktionsschritte aus der manuellen Kleinserienmontage bei der Firma Bosch dar. Die Tabelle 7.13 zeigt die Abdeckung der unterschiedlichen Fehlerfälle. Die Buchstaben geben die jeweils verwendeten Lokalisierungsverfahren an (vergleiche Abschnitt 7.1).

Die Erfassung der statistischen Daten erfolgt durch Ausführung der Montageaufgaben mit dem in Abschnitt 7.1 beschriebenen Robotersystem APAS. Um eine ausreichende Anzahl an Störungen für statistische Aussagen zu erhalten, wurden die Applikationen I bis V so eingerichtet, dass ein regulärer Dauerbetrieb ohne manuelle Eingriffe möglich ist. Die verarbeiteten Werkstücke werden dazu durch das Robotersystem automatisch wieder an die Ausgangsposition zurückgesetzt. Um unter Laborbedingungen realistische Schwankungen der Bauteillage bei einer Rotation und Verkippung im Bezug auf die senkrechte Achse zu erhalten, werden die Werkstücke bei jedem Zyklus um wenige Grad um diese Achse gedreht und aus ca. 1 mm Höhe fallengelassen. Die Applikation VI wurde in einem Bosch-Werk in der regulären Fertigung ausgeführt.

	Merkmal:	Lage-erkennung			Fehlerursache für Störungen bei Lageerkennung						Störung der Manipulation	
	Applikation:	Landmarken	Oberfl.-Fit	Kanten-Fit	Reflexion	Verdeckung	Lage-abweichung	Pseudo-Merkmale	Kein Bauteil	Ansicht abweichend	Kein Bauteil	unbekannte Greiflage
I.	Bremszylinder umsetzen		x		O				O	O	x	
II.	Koppler lagerichtig zuführen	x		x		K		M	K	K	x	
III.	Zylinder lagerichtig zuführen			x	K				K	K	x	
IV.	Ventilplatte palettieren		x		O		O		O	O	x	x
V.	Ventilnadel palettieren	x		x		M	K		K	K	x	x
VI.	Datamatrix-Code prüfen			x			K		K	K	x	

Tab. 7.13.: Abdeckung der Fehlerszenarien durch ausgewählte Applikationen. Bedeutung M: landmarkenbasierte Lokalisierung, O: oberflächenbasierter Fit, K: kantenbasierter Fit

7.3.1. Applikation I: Bremszylinder umsetzen

In Applikation I werden Bremszylinder aus einer beweglichen Kiste auf Werkstückträger umgesetzt. Die Abbildung 7.7 verdeutlicht diesen Ablauf und die Tabelle 7.14 zeigt die Randbedingungen. Die zugeführten Bauteile stehen innerhalb der Kiste aufrecht in einem labilen Kunststoffblister und können durch ihr Eigengewicht bezüglich der senkrechten Achse verkippen. Für ein fehlerfreies Absetzen auf dem Dorn eines Werkstückträgers mit einem Spiel von unter 2 mm ist die Orientierung der Zylinder mittels 3D-Oberflächen-Fit zur erfassen. Dabei treten durch die schwankende Orientierung in ca. 3% der Fälle starke Lichteffekte auf dem Bauteil auf, so dass das projizierte Streifenmuster nicht vollständig erkannt wird und die Lokalisierung fehlschlägt (siehe Abbildung 7.8). Ursache können sowohl Reflexionen durch die aktive Beleuchtung oder das Umgebungslicht sein als auch eine zu geringe Intensität des projizierten Musters in den Kamerabildern infolge eines zu flachen Winkels vom Projektor zur Bauteiloberfläche. Die Lokalisierung der beweglichen Kiste sowie der Werkstückträger erfolgt robust über Landmarken. Die Handhabung des Bremszylinders mittels zentrischen 3-Finger-Griff erfolgt nach korrekter Lageerkennung fehlerfrei.

Aufgrund der Taktzeit und der Störungshäufigkeit bei der Lokalisierung der Bremszylinder (siehe Tabelle 7.15) kommt es aufgrund von Lichteffekten auf dem Bauteil im Schnitt alle 6 bis 7

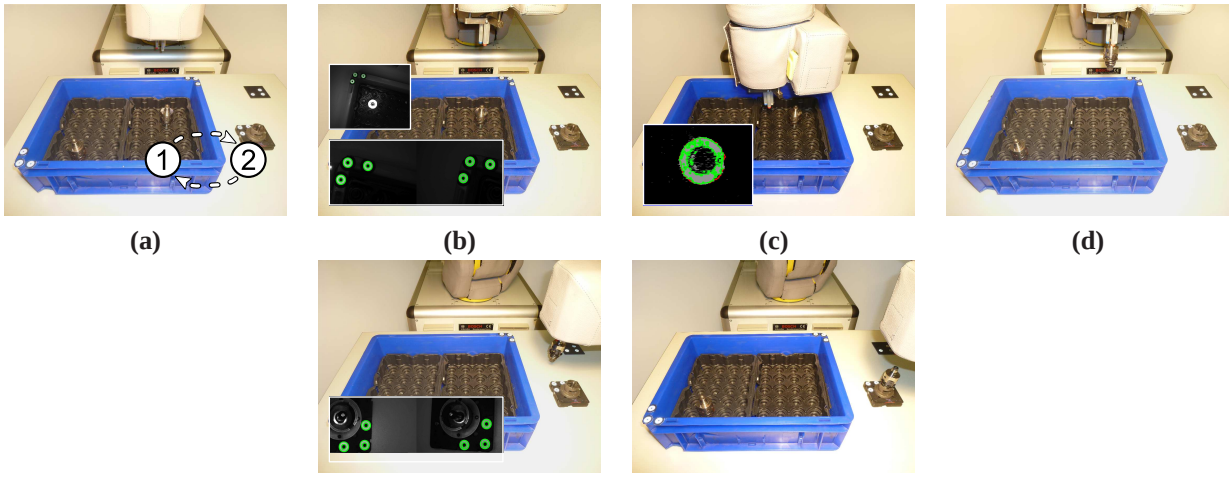


Abb. 7.7.: Ablauf der Applikation I „Bremszylinder umsetzen“: a) Überblick über die Einzelschritte. b) und c) Bauteil lokalisieren und greifen. e) und f) Auf Werkstückträger ablegen

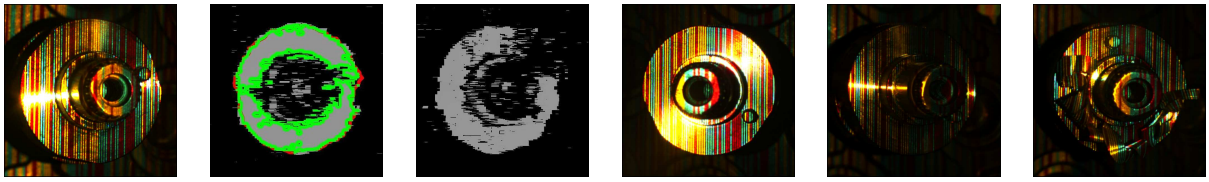


Abb. 7.8.: Detailansichten des Bremszylinders: a) Bauteil mit Streifenmuster, b) Erkannte Bauteiloberfläche, c) Lückenhafte Rekonstruktion, d) Reflexionen, e) Geringe Musterintensität, f) Span auf Bauteil

Objekt:	Bremszylinder	**Lagevarianz:**	-	Verkippung bzgl.
Zuführung:	labiler Blister			Z-Achse
Zielposition:	Werkstückträger (< 2mm)		-	Rot. um Z irrelevant
Zykluszeit:	12 s	**Lageerkennung:**		3D-Oberflächen-Fit

Tab. 7.14.: Details und Randbedingungen zur Applikation I „Bremszylinder umsetzen“

Störungsursache	Fehlerrate	MTBF	Strategie	Aufwand	Erfolgsquote
Lichteffekte auf Bauteil-oberfläche	∼3%*	6,6 min	- Position variieren - Orientierung variieren - Wiederholen - Testgriff	1 min 1 min 0,5 min 1 min	99,5%
Kein Bauteil	-		- Nächste Position	0,5 min	100%
Nicht erkennbar	-		- Entfernen	5 min	100%

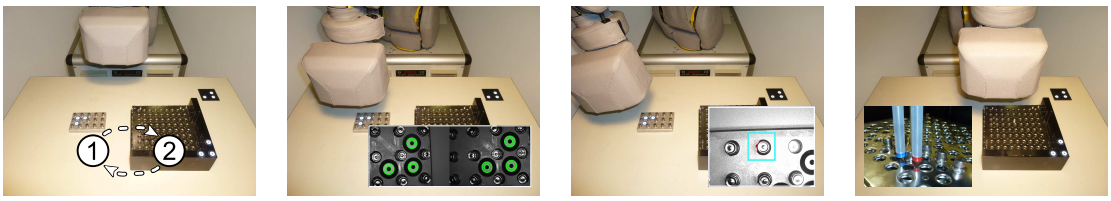
Tab. 7.15.: Details zu Störungsursachen und eingesetzten Strategien bei Applikation I. Legende: MTBF - mittlere Betriebsdauer zwischen Ausfällen; *: gemessen in 3753 Zyklen

Minuten zu einer Störung (mean time between failures - MTBF). Um die Robustheit des Ablaufs zu erhöhen, wurde die Lageerkennung um eine Störungsbehandlung erweitert. Für eine Lokalisierung trotz störender Lichteffekte in der Ausgangslage werden die Strategien zur Variation von Position und Orientierung eingesetzt. Da die grobe Bauteilposition durch den Blister vorgegeben ist, kann auch der Testgriff angewendet werden. Dieser reduziert im vorliegenden Fall häufig die Verkippung des Bauteils bezüglich der Senkrechten und erhöht die Erfolgswahrscheinlichkeit der Lokalisierungsstrategien. Das Wiederholen der Lageerkennung kann vor allem nach dem Test sinnvoll sein. Um ebenso fehlende oder nicht erkennbare Bauteile verarbeiten zu können, wurden zusätzlich die Strategien zum Entfernen des Objektes sowie zum Überspringen der Palettenposition verwendet. Die Tabelle 7.15 zeigt Details zur Störungsbehandlung.

Mit den aufgeführten Strategien wurde die Störungsrate bei der Lokalisierung um 99,5% reduziert, so dass der Ablauf im Schnitt nur noch alle 22 Stunden wegen Lichteffekten unterbrochen wird. Fehlende oder nicht erkennbare Bauteile können vollständig behandelt werden, solange letztere blind greifbar sind.

7.3.2. Applikation II: Koppler lagerichtig zuführen

In Applikation II sind sogenannte Koppler, die auf einer Palette zugeführt werden, lagerichtig in ein Prozessnest einzulegen. Die Abbildung 7.9 verdeutlicht den Ablauf. Die Position der Bauteile ist bis auf ein Spiel von 1 mm innerhalb der Palette definiert, jedoch ist die Drehlage um die senkrechte Achse zufällig. Wie in Abbildung 7.10 gezeigt, haben die Koppler an einem Außenring jeweils zwei Einbuchtungen, deren Position für den nachfolgenden Prozess relevant ist. Die Drehlage der Koppler muss daher vor der Handhabung mittels kantenbasiertem Fit erkannt werden. Wegen der geringen Merkmalsgröße ist für eine fehlerfreie Detektion eine sehr hohe Übereinstimmung mit der definierten Sollkontur erforderlich. Wie die Tabelle 7.17 zeigt trat bei 13,3% der Bauteile im



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

Abb. 7.9.: Ablauf der Applikation II „Koppler lagerichtig zuführen“: a) Überblick über die Einzelschritte, b) Lokalisierung der Zuführpalette, c) Bauteillage erkennen und Greifen, d) Bauteil lagerichtig in Palette absetzen

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

Abb. 7.10.: Detailansichten des Kopplers: a) Einbuchtungen an Bauteilkante, b) Korrekte Erkennung der Merkmale, c) - d) Erkennung durch Verdeckung und geringen Kontrast verhindert, e) Fehldetektion bei zu geringem Schwellwert bzgl. der Merkmalsübereinstimmung

getesteten Fall eine Störung bei der Lokalisierung auf. Eine Ursache hierfür ist die teilweise Verdeckung der Merkmale durch die hochstehende Zylinderform des Bauteils. Zusätzlich werden die Einbuchtungen abhängig von ihrer Position wegen fehlendem Kontrast zum Untergrund teilweise nicht erkannt. Liegt ein Bauteil sehr dicht am von den Merkmalen abgewandten Rand des Palettennests, erscheint unter den Einbuchtungen keine helle Palette sondern ein dunkles Loch, das für eine robuste Detektion zu wenig Kontrast zum Bauteil bietet (siehe Abbildung 7.10).

Zusätzlich kann es bei der Lokalisierung der Palette durch eine Lageänderung nach einem Palettenwechsel gelegentlich zu Störungen kommen, da die Kopplerober fläche unter bestimmten Beleuchtungsbedingungen den Landmarken ähnlich sieht.

Um einen robusten Automatikbetrieb zu erreichen wurde die Erkennung der Bauteildrehlage um eine Störungsbehandlung erweitert, bei der die Position der Kamera variiert werden kann, um die Verdeckung der Merkmale zu vermeiden. Zusätzlich wird ein Testgriff eingerichtet, der das Bauteil gleichzeitig um 15° verdreht absetzt und mehrfach angewendet werden kann. Durch die veränderte Position können die Kontrastprobleme behoben werden. Ein größerer Winkel würde die Ansicht stärker verändern, könnte aber relevante Lagen auslassen. Um fehlende oder nicht erkennbare Bauteile verarbeiten zu können, wurden zusätzlich die Strategien zum Entfernen des Objektes sowie zum Überspringen der Palettenposition verwendet. Die Tabellen 7.17 zeigt Details und Ergebnisse zur Störungsbehandlung. Zur Optimierung der Palettenlokalisierung wurden die Strategien zur Variation von Position und Orientierung erfolgreich eingesetzt, um die Beleuchtungsbedingungen zu verändern. Eine statistische Auswertung wurde nicht erstellt, da die Fehler manuell bedingt sind und für eine hohe Aussagekraft nicht geeignet simuliert werden können.

Objekt:	Koppler	**Lagevarianz:**	Rotation um Z
Zuführung:	Palette	**Lageerkennung:**	2D-Kanten-Fit
Zielposition:	Werkstückträger (< 1mm)		
Zykluszeit:	12 s		

Tab. 7.16.: Details und Randbedingungen zur Applikation II „Koppler lagerichtig zuführen“

Störungsursache	Fehlerrate	MTBF	Strategie	Aufwand	Erfolgsquote
- Verdeckung der Merkmale - geringer Kontrast	∼13,3%*	∼1,5 min	- Pos. variieren - Wiederholen - Testgriff inkl. Drehung um Z (15°)	1 min 0,5 min 4 min	100,0%
- Kein Bauteil	-		- Nächste Position	0,5 min	100%
- Nicht erkennbar	-		- Entfernen	6 min	100%

Tab. 7.17.: Details zu Störungsursachen und eingesetzten Strategien bei Applikation II. Legende: MTBF - mittlere Betriebsdauer zwischen Ausfällen; *: gemessen in 5027 Zyklen

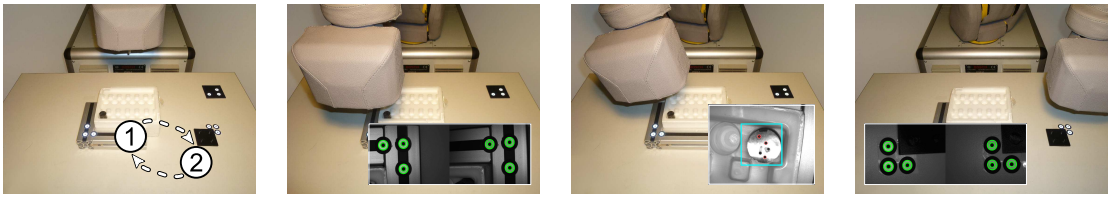
Mit den beschriebenen Maßnahmen konnten die Störungen bei der Lokalisierung der Koppler und der Palette vollständig vermieden werden. Fehlende oder nicht erkennbare Bauteile werden erfolgreich behandelt, da die eingeschränkte Objektposition einen sicheren blinden Abgriff ermöglicht.

7.3.3. Applikation III: Zylinder lagerichtig zuführen

In Applikation III sind Metallzylinder lagerichtig in ein Prozessnest abzusetzen. Die Bauteile werden über eine Palette mit sehr geringer Positionsvarianz jedoch mit zufälliger Drehlage bezüglich der senkrechten Achse zugeführt, wie die Abbildung 7.11 und die Tabelle 7.18 zeigen. Da die Komponenten auf der Unterseite eine außerzentrische Bohrung sowie eine Aussparung besitzen (siehe Abbildung 7.12) und auf einer entsprechenden Negativform abgesetzt werden, muss ihre Drehlage vor der Handhabung erfasst werden. Hierzu können Merkmale auf der Bauteiloberseite mittels kantenbasiertem Fit lokalisiert werden.

Aufgrund von Lichtreflexionen oder Spiegelungen auf der polierten Bauteiloberfläche werden die Merkmale bei ca. 30% der Lageerfassungen nicht ausreichend sicher erkannt, wie die Tabelle 7.19 zeigt. Die Detektion der Palettenposition sowie die Bauteilhandhabung erfolgt sehr robust.

Um die Störungen bei der Erfassung der Bauteildrehlage zu behandeln, werden Strategien zur Variation der Position sowie der verwendeten Belichtungszeit und des Verstärkungsfaktors bei der Bildaufnahme eingesetzt. Zusätzlich kann ein Testgriff durchgeführt werden, der die Drehlage gleichzeitig um 15° verändert und die Erfolgswahrscheinlichkeit einer wiederholten Lokalisierung unter Originalbedingungen erhöht. Um fehlende oder nicht erkennbare Bauteile verarbeiten zu können, wurden zusätzlich die Strategien zum Entfernen des Objektes sowie zum Überspringen der Palettenposition verwendet.



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

Abb. 7.11.: Ablauf der Applikation III „Zylinder lagerichtig zuführen“: a) Überblick über die Einzelschritte. b) Lokalisierung der Zuführpalette über Anschlagwinkel c) Bauteillage erkennen und Greifen. d) Bauteil lagerichtig auf Prozessnest absetzen

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

Abb. 7.12.: Detailansichten des Metallzylinders: a) Oberseite des Zylinders, b) Asymmetrische Unterseite, c) Korrekt erkannte Merkmale d) und e) Starke Spiegelungen und Reflexionen auf der Metalloberfläche

Mit den beschriebenen Maßnahmen konnten die Störungen bei der Lokalisierung des Metallzylinders vollständig vermieden werden. Fehlende oder nicht erkennbare Bauteile werden erfolgreich behandelt, da die eingeschränkte Objektposition einen sicheren blinden Abgriff ermöglicht.

Objekt:	Metallzylinder	**Lagevarianz:**	Rotation um Z
Zuführung:	Palette	**Lageerkennung:**	2D-Kanten-Fit
Zielposition:	Prozessnest (< 1mm)		
Zykluszeit:	10 s		

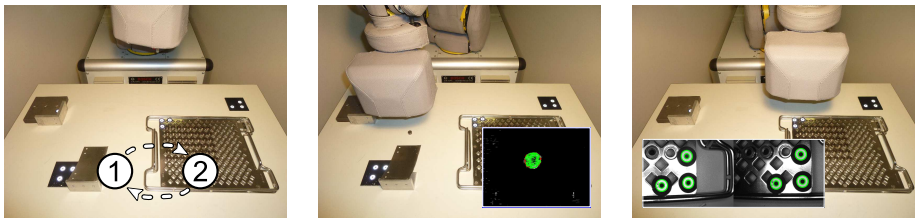
Tab. 7.18.: Details und Randbedingungen zur Applikation III „Zylinder lagerichtig zuführen“

Störungsursache	Fehlerrate	MTBF	Strategie	Aufwand	Erfolgsquote
- Reflexion / Spiegelungen auf Oberfläche um Merkmale	∼30,5%*	∼0,5 min	- Position variieren - Parameter variieren - Wiederholen - Testgriff inkl. Drehung um Z (15°)	1 min 3 min 0,5 min 4 min	100,0%
- Kein Bauteil	-		- Nächste Position	0,5 min	100%
- Nicht erkennbar	-		- Entfernen	6 min	100%

Tab. 7.19.: Details zu Störungsursachen und eingesetzten Strategien bei Applikation III. Legende: MTBF - mittlere Betriebsdauer zwischen Ausfällen; *: gemessen in 5480 Zyklen

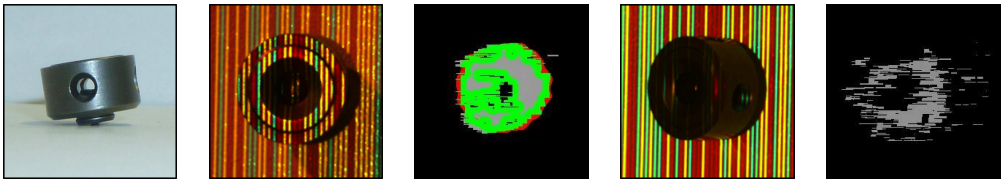
7.3.4. Applikation IV: Ventilplatte palettieren

In Applikation IV werden Ventilplatten über eine Rutsche aus einer Maschine ausgeschleust und sind zu palettieren. Die Abbildung 7.13 verdeutlicht den Ablauf. Die Bauteile haben auf der Unterseite einen Zapfen und können aufgrund der Zuführungsart in Position und Orientierung schwanken, wie die Abbildung 7.14 zeigt. Die Palette ist über Landmarken zu lokalisieren und bietet nur wenig Spiel beim Absetzen der Bauteile, wie die Tabelle 7.20 zeigt. Die Lage der Ventilplatten wird daher mittels 3D-Oberflächen-Fit ermittelt. Hierbei treten in ca. 19% der Fälle Störungen auf, wie die Tabelle 7.21 verdeutlicht. Ursächlich ist eine Kombination aus der kleinen Bauteiloberfläche zusammen mit auftretenden Lichteffekten und der nicht optimalen Eignung des Erkennungsalgorithmus für dieses Bauteilmaterial. Zusätzlich treten bei der Palettenlokalisierung Störungen auf, da die Oberseite der Ventilplatte unter bestimmten Randbedingungen als Landmarke erkannt wird. Um die Robustheit bei der Detektion der Bauteillage zu erhöhen werden Strategien zur Variation der



(a) **(b)** **(c)**

Abb. 7.13.: Ablauf der Applikation IV „Ventilplatte palettieren“: a) Ablauf der Applikation IV „Ventilplatte palettieren“: a) Überblick über die Einzelschritte. b) über eine Rutsche zugeführtes, frei liegendes Bauteil lokalisieren d) Palette lokalisieren und Bauteil fügen



(a) **(b)** **(c)** **(d)** **(e)**

Abb. 7.14.: Detailansichten des Kopplers: a) Seitenansicht des verkippenden Kopplers, b) zur Lokalisierung projiziertes Streifenmuster auf Bauteil, c) erfolgreiche Lageerkennung im 2,5D-Tiefenbild d) Darstellung der fehlenden Musterintensität bei flacherem Blickwinkel, e) fehlgeschlagene Lokalisierung durch schlechte 3D-Punkt-Rekonstruktion

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

Abb. 7.15.: Darstellung der Strategien „Anwesenheitsprüfung“ und „Entfernen“ für Applikation IV: a) Rakel greifen, b) Zentrierbewegung zur Positionierung des Bauteils, c) Anwesenheitstest per Griff, d) Bauteil mit Rakel entfernen

Objekt:	Ventilplatte	**Lagevarianz:**	Variation in Lage und Orientierung
Zuführung:	Rutsche		
Zielposition:	Palette (< 1mm)	**Lageerkennung:**	3D-Oberflächen-Fit
Zykluszeit:	14 s		

Tab. 7.20.: Details und Randbedingungen zur Applikation IV „Ventilplatte palettieren“

Störungsursache	Fehlerrate	MTBF	Strategie	Aufwand	Erfolgsquote
- Reflexion und / Streuung auf Bauteiloberfläche	∼19,0%*	∼1,25 min	- Position variieren - Orientierung variieren - Wiederholen - Testgriff mit vorheriger Zentrierung	1 min 1 min 0,5 min 16 min	92,2%
- Kein Bauteil	-		- Nächstes Teil	1 min	100%
- Nicht erkennbar	-		- Entfernen	5 min	100%

Tab. 7.21.: Details zu Störungsursachen und eingesetzten Strategien bei Applikation IV. Legende: MTBF - mittlere Betriebsdauer zwischen Ausfällen; *: gemessen in 4831 Zyklen

Position und Orientierung der Kamera eingesetzt sowie eine Wiederholung bei Originalbedingungen. Da dies für eine hohe Erfolgsquote nicht ausreicht, wird zusätzlich eine Anwesenheitsprüfung durch blindes Zugreifen integriert. Wegen der undefinierten Bauteilposition wird die Ventilplatte hierzu zunächst durch zwei entgegengesetzte Schiebebewegungen mit einem Metallwinkel an eine definierte Position geschoben. Eine ähnliche Schiebebewegung ermöglicht auch das Entfernen des Bauteils aus dem Suchbereich. Ist keine Ventilplatte vorhanden kann ein neues Teil angefordert werden. Zur Optimierung der Palettenlokalisierung werden Strategien zur Veränderung der Kameraposition und Orientierung eingesetzt. Wie bereits bei Applikation II beschrieben, treten Störungen bei der Lageerkennung der Palette nur auf, wenn diese manuell bewegt wurde. Da hier für aussagekräftige Daten kein ausreichend zufälliger und umfangreicher Prozess realisiert werden konnte, erfolgt keine statistische Auswertung der Behandlung.

Mit den beschriebenen Maßnahmen konnten Unterbrechungen des Programmablaufs durch Störungen bei der Lokalisierung der Ventilplatte sowie der Palette vollständig vermieden werden. Ebenso werden fehlende oder nicht erkennbare Bauteile erfolgreich behandelt. Durch die komplexen Strategien zur Prüfung der Anwesenheit und zur Entfernung des Bauteils kann die optimale Behandlung abhängig von der Applikation über der Taktzeit des Prozesses liegen. Werden die Bauteile z. B. automatisch zugeführt, kann es zu einem Teilestau auf der Ablagefläche kommen, der ggf. eine Greiferkollision verursacht. Im vorliegenden, simulierten Fall wurden die Bauteile durch das Robotersystem der Rutsche zugeführt, so dass Staus auszuschließen waren. In der Fertigung ist abhängig vom Prozess zu entscheiden, ob die Behandlung z. B. innerhalb eines festen Zeitlimits abgeschlossen sein muss (siehe Abschnitt 5.8).

7.3.5. Applikation V: Ventilnadel palettieren

Bei der Applikation V werden Ventilnadeln über eine Rutsche zugeführt und sind zu palettieren. Die kantenbasierte Lokalisierung der Bauteile erfolgt robust. Über eine Haltevorrichtung wird das Bauteil umgegriffen, da die Nadeln nur waagerecht aufgenommen und nur senkrecht in die Palette abgesteckt werden können. Die Abbildung 7.16 verdeutlicht den Ablauf. Durch den vorhergehenden Prozessschritt sind die Bauteile mit Flüssigkeit benetzt, wodurch es zu einer kurzen Anhaftung der Nadeln an den Greiferfingern kommen kann. Dies führt beim Ablegen in 0,2% der Fälle zum Herunterfallen von der Umgreifvorrichtung. Da der Ablauf ohne Störungsbehandlung normal fortgeführt wird, weist die befüllte Palette Lücken auf. Weitere Störungen können bei der Lokalisierung der Palette auftreten, wenn der Bediener diese so ablegt, dass ein Metalldorn auf der Palette die Landmarken verdeckt.

Zur Erhöhung der Ablaufrobustheit wurde das Aufnehmen der Nadel von der Umgreifvorrichtung um eine Störungsbehandlung erweitert. Schließen die Finger beim Zugreifen ohne Nadel vollständig, wird über eine Strategie ein neues Bauteil geholt. Zusätzlich kann die Position und Orientierung der Kamera bei der Lokalisierung der Palette variiert werden, um eine frei Sicht auf die Landmarken zu erreichen. Wie die Tabelle 7.23 zeigt, konnte die Behandlung 100% der Störungen lösen. Das Einrichten der Strategien benötigte ca. 3min. Eine statistische BEwertung der Markerlokalisierung wurde nicht durchgeführt. Da der Fehler nicht zufällig auftritt wäre das Ergebnis wenig aussagekräftig. Bezüglich der Bauteilmanipulation konnten alle Störungen behandelt werden, so dass die Palette stets vollständig befüllt war. Für die Bewertung der Wirtschaftlichkeit im industriellen Einsatz ist neben der Fehler- sowie der Behandlungsrate zu berücksichtigen, ob die fallengelassenen Werkstücke als Ausschuss gelten oder weiterverwendbar sind.

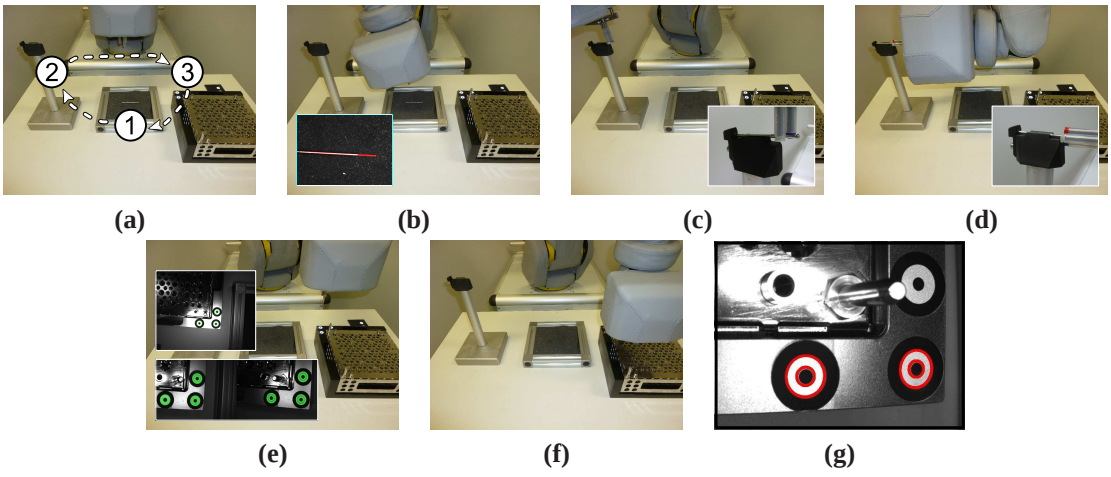


Abb. 7.16.: Ablauf der Applikation V „Ventilnadel palettieren“: a) Überblick über die Einzelschritte, b) Ventilnadel lokalisieren, c) & d) Nadel umgreifen, e) & f) Palette lokalisieren und Vantilnadel abstecken, g) Fehllokalisierung der Landmarken der Palette bei Verdeckung durch einen Dorn

Objekt:	Ventilnadel	**Lagevarianz:**	keine
Zuführung:	Prozessnest	**Lageerkennung:**	keine - Abgriff blind
Zielposition:	Palette (< 1mm)		
Zykluszeit:	16 s		

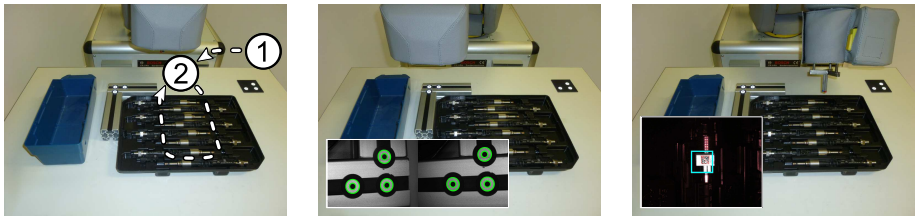
Tab. 7.22.: Details und Randbedingungen zur Applikation V „Ventilnadel palettieren“

Störungsursache	Fehlerrate	MTBF	Strategie	Aufwand	Erfolgsquote
- Kein Bauteil gegriffen, da Ablegen zuvor fehlerhaft	∼0,2%*	∼133 min	- Neues Bauteil holen	2 min	100,0%

Tab. 7.23.: Details zu Störungsursachen und eingesetzten Strategien bei Applikation V. Legende: MTBF - mittlere Betriebsdauer zwischen Ausfällen; *: gemessen in 5472 Zyklen

7.3.6. Applikation VI: Datamatrix-Code prüfen

In Applikation VI wird das flexible Robotersystem dazu eingesetzt, die Anwesenheit von Datamatrix-Codes auf gefertigten Injektoren zu überprüfen. Diese Merkmale werden in einem vorherigen Prozessschritt aufgebracht, um Messergebnisse zu speichern. Die Abbildung 7.17 zeigt den Ablauf. Bei der Applikation werden jeweils acht Werkstücke pro Palette zugeführt und sind zu prüfen. Die in der Tabelle 7.24 angegebene Zykluszeit ergibt sich aus der Prüfung und den häufigen Palettenwechseln. Verfügt ein Bauteil über keinen Code, wurde der Schritt nicht ordnungsgemäß durchlaufen und die Komponente ist auszusortieren. Das Bauteil wird dann über eine Remove-Strategie



(a) **(b)** **(c)**

Abb. 7.17.: Ablauf der Applikation VI „Datamatrix-Code prüfen“: a) Überblick über Einzelschritte, b) Palette über Anschlag lokalisieren, c) Anwesenheit Datamatrix-Codes prüfen

(a) **(b)** **(c)** **(d)** **(e)**

Abb. 7.18.: Detailansichten des Datamatrix-Codes: a) Erfolgreiche Erkennung des Merkmals, b) Bauteilansicht ohne Code, c) Seitenansicht eines verdrehten Bauteils, d) & e) Abweichung von Helligkeit und Kontrast durch Verdrehung des Bauteils

Objekt:	Injektor	**Lagevarianz:**	-	Rotation um
Zuführung:	Palette			Bauteilachse
Zykluszeit:	2,5 s	**Anwesenheitsprüfung:**	-	2D-Kanten-Fit

Tab. 7.24.: Details und Randbedingungen zur Applikation VI „Datamatrix-Code prüfen“

Störungsursache	Fehlerrate	MTBF	Strategie	Aufwand	Erfolgsquote
- Reflexion oder Abschattung auf Bauteil durch Verdrehung	∼6,3%*	∼40s	- Bauteil in erwartete Lage drehen	5 min	100,0%
- Fehlender Code auf Bauteil	0,017%	∼4h	- Werkstück entfernen	10 min	100,0%

Tab. 7.25.: Details zu Störungsursachen und eingesetzten Strategien bei Applikation VI. Legende: MTBF - mittlere Betriebsdauer zwischen Ausfällen; *: gemessen in 5424 Zyklen

automatisch gegriffen und separat abgelegt. Bei der Erkennung der Codes kann es zu Störungen kommen, wenn die Injektoren leicht um ihre Längsachse verdreht sind. Die polierte Fläche um den Code spiegelt dann sehr stark und verhindert eine Merkmalszuordnung, wie in Abbildung 7.18 verdeutlicht. Die Tabelle 7.25 zeigt Details zu den angewendeten Strategien.

Um die Aussortierung fehlerfreier Injektoren durch diese falsch-positive Erkennung zu vermeiden, wird eine Behandlungsstrategie angewendet, bei der der Greifer so am Bauteil vorbeifährt, dass dieses unabhängig von seiner Ausgangslage in die erwartete Lage gedreht wird. Die Erkennung wird daraufhin wiederholt. Mit dieser Behandlung können Injektoren mit fehlenden Codes aussortiert und Falsch-Detektionen vollständig verhindert werden. Diese Applikation zeigt zusätzlich, dass die Störungsbehandlung gleichzeitig auch für Prüfaufgaben geeignet ist, indem Fehlerteile direkt separat abgelegt werden.

7.4. Evaluierung der erfahrungsbasierten Strategieauswahl

Die Evaluierung der erfahrungsbasierten Strategieauswahl erfolgt repräsentativ an zwei Applikationen aus Abschnitt 7.3. Als Grundlage wird in Abschnitt 7.4.1 zunächst analysiert, welche Methode und Parametrierung zu einem optimalen Nutzen bzgl. der Initialisierung des Erfahrungswissens führen. Im Anschluss erfolgt die Bewertung des Optimierungsansatzes anhand von Dauerlaufversuchen in Abschnitt 7.4.2.

7.4.1. Evaluierung der Parameter zur Initilisierung des Erfahrungswissens

Der vorgestellte Ansatz für eine optimierte Strategieauswahl basiert auf der Verwendung von Erfahrungswissen. Dieses liegt jedoch direkt nach der Programmierung neuer Applikationen noch nicht vor. Zudem erfolgt der Informationszuwachs durch die bedingten Abhängigkeiten nur langsam. In diesem Abschnitt wird zunächst überprüft, welchen Einfluss die Parameter des Optimierungsalgorithmus auf den Informationszuwachs und somit auf die Behandlungsdauer im Störungsfall haben. Zu bewerten ist die ergebnisunabhängige Ausführung aller Behandlungsstrategien bei den ersten auftretenden Störungen sowie der Aufruf einer zufällig gewählten Zusatzstrategie bei jeder Störung (vgl. Abschnitt 5.5). Beide Konzepte sollen die Güte der statistischen Daten durch Zusatzinformationen erhöhen und zu einer besseren Schätzung des Fehlerzustands sowie zu einer gezielteren Behandlung führen.

Zur Evaluierung dieser Optionen wird der Verlauf der mittleren Behandlungsdauer für unterschiedliche Parametrierungen über viele Störungen analysiert. Um vergleichbare Randbedingungen garantieren zu können, werden die Ergebnisse der Strategieaufrufe auf Basis von realen Werten simuliert. Hierzu wurden im Vorfeld Dauerläufe an den Applikationen aus Abschnitt 7.3 mit einem realen Robotersystem durchgeführt. Im Störungsfall wurden dabei stets alle Strategien ausgeführt und deren Ergebnis sowie die Ausführdauer gespeichert. Auf Basis dieser Daten kann die Evaluierung bei unterschiedlichen Parametern vollständig vergleichbar erfolgen. Analysiert werden die Applikationen „Bremszylinder depalettieren“ und „Metallzylinder lagerichtig umsetzen“, da an diesen Aufgaben ein zufälliger Effekt verdeutlicht werden kann.

Initiale Ausführung aller Lokalisierungs- und Prüfstrategien

Wie in Abschnitt 5.5 beschrieben, dient die Ausführung aller Lokalisierungs- und Prüfstrategien bei den ersten Störungen zur schnellen Initialisierung der Wahrscheinlichkeitsbäume. Bei einer normalen Störungsbehandlung werden nur so lange Strategien ausgeführt, bis ein Erfolg eintritt. Entsprechend wenige Daten werden erfasst. Bei der ergebnisunabhängigen Ausführung aller Lokalisierungs- und Prüfstrategien werden deutlich mehr und gleichzeitig unabhängige Informationen gewonnen. Dieser Wissensvorteil sollte direkt nach der Initialisierung zu besseren Planungsergebnissen und somit zu einer reduzierten Behandlungsdauer führen. Da die Ausführung aller Strategien viel Zeit erfordert, ist zu analysieren, mit wie vielen Initialisierungsläufen eine insgesamt minimale Behandlungsdauer erzielt wird.

Zur Evaluierung dieser Frage wurde die Ausführung der Applikation „Bremszylinder depalettieren“ mit den aufgenommenen Daten simuliert und die Anzahl der Initialisierungsläufe jeweils von 0 bis 10 gesteigert. Die Versuche wurden mit jeder Parametrierung drei Mal ausgeführt und die Ergebnisse gemittelt, um den Einfluss einer zufälligen Strategieauswahl zu reduzieren. Diese erfolgt, wenn mehrere Behandlungsmöglichkeiten eine gleiche Erfolgswahrscheinlichkeit aufweisen. Für die gesamte Evaluierung gelten die Strafen $r_{next,1} = -100$, $r_{next,n} = -20$, $r_{remove,1} = -20$ und $r_{remove,n} = -20$. Da die Behandlungsdauer abhängig von den einzelnen Störungsfällen und der

gewählten Strategiereihenfolge stark schwankt, sind die direkten Messwerte nur schwer vergleichbar. Aus diesem Grund wird jeweils die mittlere Behandlungsdauer über alle zuvor aufgetretenen Störungen berechnet. Diese berechnet sich für die Störung mit dem Index i nach der Formel 7.1 aus der Dauer t_i der bisherigen Behandlungen.

$$f(i) = (\sum_{j=1}^{i} t_j)/i \tag{7.1}$$

Die Diagramme 7.19a bis 7.19f zeigen die Ergebnisse der Messungen ohne sowie mit 1, 5 und 10 vollständigen Initialisierungsdurchläufen für zwei Auswertungsvarianten. Bei den Abbildungen 7.19a, 7.19c und 7.19e ist die zur Initialisierung benötigte Zeit nicht berücksichtigt. Sie starten daher nicht bei Störung 1. Man erkennt, dass sich bei einem sowie bei fünf Initialisierungsläufen zunächst ein Vorteil bei der Behandlungszeit ergibt. Dieser egalisiert sich durch den kontinuierlichen Wissenszuwachs sowie durch zufällige Effekte jedoch innerhalb der ersten 100 Messungen. Bei der Durchführung von zehn Initialisierungsläufen ist bereits zu Beginn kein Vorteil mehr erkennbar. Die relative Verbesserung durch weitere Zusatzinformationen wird mit steigender Anzahl an Initialisierungen immer geringer. Gleichzeitig verbessert sich die Ausführung ohne Initialisierung im selben Zeitraum. Die Ergebnisse zeigen, dass die Reduktion der Behandlungsdauer durch die Initialisierung relativ gering ist und starken zufälligen Effekten unterliegt.

Die Abbildungen 7.19b, 7.19d und 7.19f verdeutlichen, dass die Initialisierung durch einen Aufruf aller Strategien in den meisten Fällen zu einer insgesamt erhöhten Behandlungsdauer führt. Die Diagramme zeigen die mittlere Behandlungsdauer ab der ersten Störung, so dass die Dauer für die Initialisierung berücksichtigt wird. Die Verläufe beginnen mit ca. 14 Sekunden, die für den Aufruf aller Strategien benötigt werden. Es ist erkennbar, dass der Zusatzaufwand für die Initialisierung nicht kompensiert wird und sich die absolute Behandlungsdauer mit steigender Anzahl durchgeführter Initialisierungen verschlechtert.

Die Behandlungsdauer zwischen den Messreihen gleicht sich auch auf lange Zeit nicht an, obwohl die Strategieergebnisse permanent in die Datenbasis einfließen. Die Ursache liegt darin, dass jeweils nur die Strategien mit den besten ermittelten Wahrscheinlichkeiten ausgeführt werden. Die gemessenen Erfolgsraten können jedoch durch zufällige Effekte nach unten abweichen. Schlägt eine Strategie während der Initialisierung zufällig überproportional häufig fehl, wird eine geringe Erfolgsquote erwartet. Als Folge wird die entsprechende Strategie bei vermeintlich besseren Alternativen nicht erneut aufgerufen und die Abweichung nicht erkannt. Die relativ geringe Anzahl an durchgeführten Initialisierungsläufen kann diesen Effekt nicht verhindern, wie die Tabelle 7.26 verdeutlicht. Dargestellt sind die mittlere Behandlungsdauer zwischen Störung 450 und 550 in Abhängigkeit von der Anzahl der Initialisierungen. Die Wertepaare sind zeitlich sortiert. Eine Abhängigkeit von der jeweiligen Anzahl an Initialisierung ist nicht erkennbar. Hierfür spricht auch, dass sich verschiedene Messungen mit gleicher Anzahl an Initialisierungen ebenfalls unterschei-

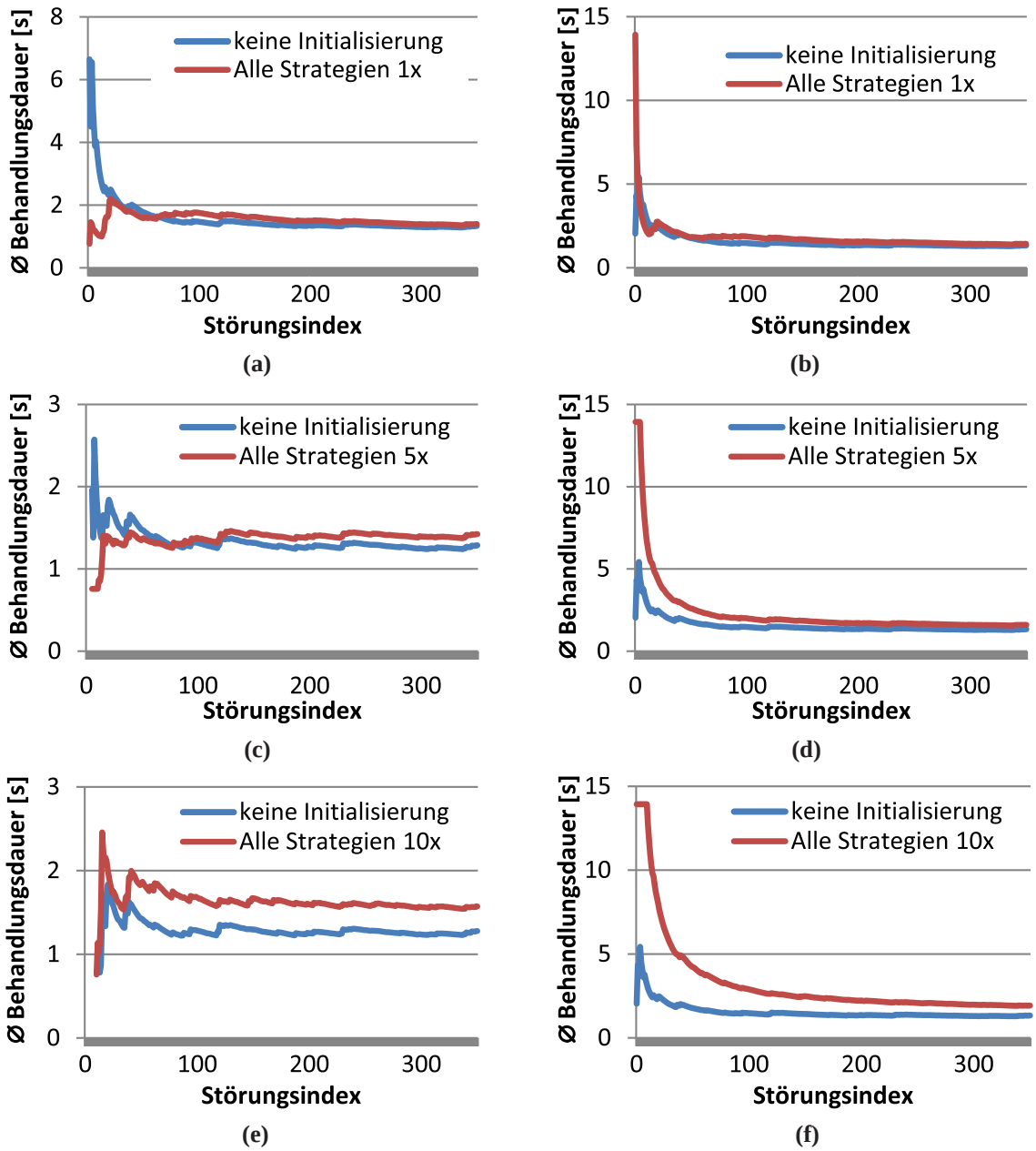


Abb. 7.19.: Mittlere Dauer der Störungsbehandlung bei 1, 5 und 10 Initialisierungsdurchläufen: a), c) und e): ohne Dauer der Initialisierung, b), d) und f) inklusive Initialisierungdauer

den. Die Abbildungen 7.19a bis 7.19f zeigen, dass der Einfluss der Pseudo-Maxima innerhalb der statistischen Daten den Vorteil der Initialisierung bereits nach wenigen Störungen dominiert.

Fazit: Die Initialisierung des Wahrscheinlichkeitsbaums durch mehrfache Ausführung aller Strategien führt in den meisten Fällen zu einer Erhöhung der absoluten Behandlungsdauer. Lediglich für eine einmalige Initialisierung ergab sich ein Vorteil. Dieser hängt aber maßgeblich von der Anzahl der zugewiesenen Strategien ab, da diese direkt in die erforderliche Initialisierungsdauer einfließt. Da es zum Konzept der intuitiv bedienbaren Störungsbehandlung gehört, dass Nutzer ohne Erfahrung schnell und ohne Nachteile für die Ausführdauer viele Strategien für die Behandlung hinzufügen können, ist mit einer größeren Anzahl an Strategien zu rechnen. Die Initialisierung

Initialisierungsanzahl	1	6	9	2	3	8	0	4	5	10	7
∅ Behandlungsdauer [s]	1,09	1,09	1,12	1,15	1,16	1,18	1,23	1,31	1,40	1,51	1,57

Tab. 7.26.: Mittlere Behandlungsdauer nach 450 bis 550 Störungen in Abhängigkeit von der Anzahl an Initialisierungen

zeigte bei den anderen getesteten Applikationen ebenfalls statistisch keine positiven Effekte. Auch bei der Verwendung von Prüfstrategien konnte kein signifikanter Vorteil festgestellt werden. Die Initialisierung durch ergebnisunabhängige Aufrufe aller Strategien wird im Folgenden nicht weiter betrachtet.

Optimierung des Erfahrungswissens durch Ausführung einer Zusatzstrategie

Wie gezeigt, erreicht die vorgestellte erfahrungsbasierte Störungsbehandlung nicht zwingend eine minimale Behandlungsdauer. Durch die Kombination von Erfahrungslernen und einer erfahrungsbasierten Ausführungsplanung kann die erfasste Erfolgswahrscheinlichkeit einer Strategie vom realen Wert nach unten abweichen, ohne dass dieser Fehler erkannt und behoben wird. Die zusätzliche Ausführung einer unabhängigen Strategie soll diesen Effekt beheben (sieheAbschnitt 5.7).

Zur Evaluierung dieser Methode wurde der Verlauf der Behandlungsdauer an verschiedenen Applikationen über viele Störungen jeweils mit und ohne Zusatzstrategie analysiert. Zur Reduktion von zufälligen Effekten sind nachfolgende Verläufe jeweils aus zehn Einzelmessungen gemittelt. Die Abbildung 7.20 zeigt die Ergebnisse für die Applikation „Bremszylinder umsetzen“. Verwendet wurden die Strategien „Position variieren“, „Orientierung variieren“ sowie „Wiederholen“. Dargestellt ist der Verlauf der mittleren Dauer über alle vorherigen Behandlungen in Abhängigkeit vom Störungsindex (vgl. Formel 7.1) sowohl mit als auch ohne Ausführung einer Zusatzstrategie. Störungen, die mit diesen Strategien ohne Anwesenheitsprüfung nicht behandelt werden konnten, sind nicht berücksichtigt. Da diese Fälle innerhalb der Messung nicht gleich verteilt sind, würden sie den zu analysierenden Verlauf verfälschen. Die Aussage der Messungen wird durch das Ignorieren dieser Störungen nicht beeinflusst.

Im Diagramm ist zu erkennen, dass die Behandlung mit zusätzlich ausgeführter Strategie deutlich länger dauert als ohne. Dieses Ergebnis ist im vorliegenden Fall auch zu erwarten, da die durchschnittliche Ausführdauer einer Zusatzstrategie hier ca. 0,78 s beträgt. Diese Zeit ist bei einer mittleren Behandlungsdauer von ca. 1,3 s nicht zu kompensieren. Der Verlauf „bei Aussetzen der Zusatzstrategie“ zeigt die Behandlungsdauer des Verlaufs „mit Zusatzstrategie“ abzüglich der mittleren Ausführdauer der Zusatzstrategie. Das Ergebnis entspricht der resultierenden Behandlungszeit, wenn ab dem entsprechenden Störungsindex keine Extrastrategie mehr ausgeführt würde. Die Differenz zwischen diesem Verlauf und dem „ohne Zusatzstrategie“ zeigt, dass eine Verbesserung der statistischen Datenbasis durch die Zusatzinformationen eintritt. Dieser Prozess ist circa ab der 300. Störung abgeschlossen, da der Abstand anschließend konstant ist. Ab diesem Zeitpunkt hat die Zusatzstrategie keinen positiven Effekt mehr. Jede weitere Ausführung würde nur unnötig Zeit kosten. Wie in Abbildung 7.20 erkennbar ist, tritt die leichte nachfolgende Verbesserung auch ohne zusätzliche Strategieausführungen ein. Die erreichbare Zeitreduktion liegt bei 9,9% oder ca. 0,13s.

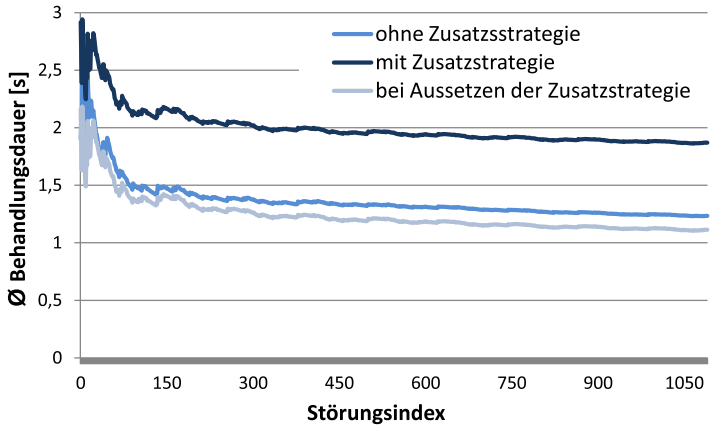


Abb. 7.20.: Verlauf der mittleren Behandlungsdauer bei Störungen in Applikation I abhängig von der Ausführung einer Zusatzstrategie

Da dies etwa einem Sechstel der mittleren Ausführdauer einer Zufallsstrategie entspricht, würde sich der bei den ersten 300 Störungen durchgeführte Zusatzaufwand ab der 1800. Störung rentieren und insgesamt zu einer Zeitreduktion führen. Bei einer Fehlerrate von ca. 3% und einer Zykluszeit von 12s für die Applikation I (siehe Abschnitt 7.3.1) ergibt sich ein Vorteil nach knapp 200 Stunden Betriebszeit. Da dies den Zeitraum einer Kleinserienmontage übersteigen kann, wäre bei sehr geringen Stückzahlen abzuwägen, ob die Ausführung von Zusatzstrategien sinnvoll erscheint.

Der Vorteil dieses Optimierungsansatzes ist jedoch stark von den Randbedingungen der Applikation abhängig. Die Abbildung 7.21 verdeutlicht dies an den entsprechenden Verläufen für die Applikation „Zylinder lagerichtig zuführen“ (siehe Abschnitt 7.3.3). Zur Störungsbehandlung wurden die Strategien „Position variieren“, „Parameter variieren“ sowie „Wiederholen“ zugewiesen. Die Optimierung der statistischen Daten ist nach ca. 170 Störungen abgeschlossen. Danach ist die Differenz zwischen „ohne Zusatzstrategie“ und „ohne Zusatz, verbessert“ annähernd konstant und fällt mit ca. 0,17s größer aus als bei der vorherigen Applikation. Dies entspricht einer Reduktion um 17,5%. Zusammen mit der geringeren Ausführdauer der Zufallsstrategie von ca. 0,52s trifft eine Kompensation des Zusatzaufwands bereits nach etwa der dreifachen Störungsmenge und somit nach 510 Behandlungen auf. Dies entspricht bei der Störungsrate von 30,5% und der Zykluszeit von 10 Sekunden (siehe Abschnitt 7.3.3) einer Betriebsdauer von ca. 4,6 Stunden. Unterhalb von dieser Zeitspanne wäre eine Automatisierung der Montage auch mit einem flexiblen Robotersystem wegen des Inbetriebnahmeaufwands nicht sinnvoll, so dass sich der Einsatz einer Zusatzstrategie in jedem Fall rentieren kann.

Die Ursache für die unterschiedlich starken Auswirkungen der Zusatzstrategie liegt in den Relationen zwischen den auftretenden lokalen und globalen Wahrscheinlichkeiten. Solange für eine Strategie keine statistischen Daten vorliegen, werden globale Erfolgswahrscheinlichkeiten verwendet, die über alle bisherigen Applikationen gemittelt werden. Für die erste auszuführende Strategie liegt dieser Wert bei der Applikation „Bremszylinder umsetzen“ bei ca. 61,5%. Diese Erfolgsrate wird zu Beginn für alle Strategien angenommen. In Abbildung 7.22 ist dieser Wert (rot) zusammen

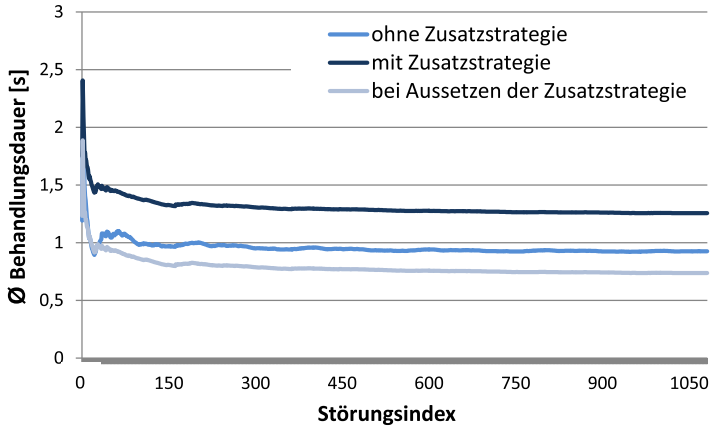


Abb. 7.21.: Verlauf der mittleren Behandlungsdauer bei Störungen in Applikation III abhängig von der Ausführung einer Zusatzstrategie

mit den lokalen Wahrscheinlichkeiten dargestellt, die bei dieser Applikation über alle Störungen gemessen wurden. Man erkennt, dass nur die vier besten Strategien höhere Raten aufweisen, sich zueinander aber nur wenig unterscheiden. Wird bei den ersten Störungen zufällig eine andere Strategie zuerst ausgewählt, fällt deren gemessene Erfolgschance statistisch nach wenigen Durchläufen unter die globale Rate. Bei der nächsten Störung wird daher eine andere Strategie ausgewählt, die noch nicht ausgeführt wurde und der noch die globale Erfolgswahrscheinlichkeit zugewiesen ist. Auf diese Weise werden die Strategien solange automatisch gewechselt, bis eine der vier besten gewählt ist und die lokale Erfolgschance über der globalen verbleibt. Bei dieser Applikation wird daher auch ohne Zusatzstrategie schnell eine sehr gute Lösung erreicht, so dass die zusätzliche Ausführung keine große Verbesserung bewirkt.

Bei Applikation „Zylinder lagerichtig zuführen“ beträgt die globale Erfolgswahrscheinlichkeit für die erste Strategie ca. 56,1%. Die Abbildung 7.23 zeigt zum Vergleich die lokalen Raten der verwendeten Behandlungsmöglichkeiten. Es liegen deutlich mehr Werte über der globalen Rate und diese unterscheiden sich mit einem maximalen Unterschied von ca. 26% zwischen 86,8% und 60,9% auch stärker. Wird z.B. die Strategie mit Index 22 zufällig zuerst ausgewählt und deren

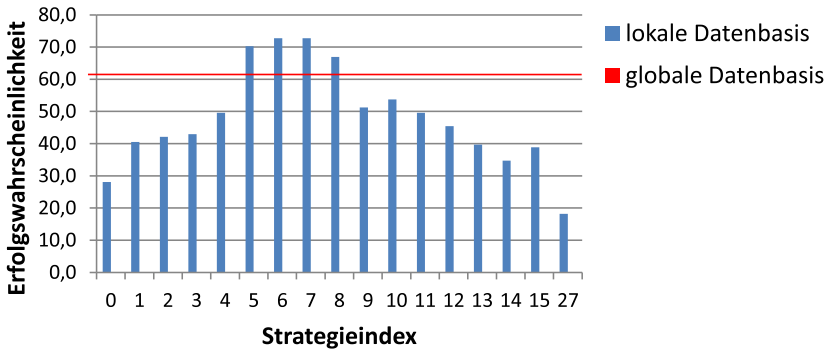


Abb. 7.22.: Lokale und globale Erfolgswahrscheinlichkeiten der zugewiesenen Strategien für Applikation I. Strategieindex 0-7: „Position variieren“, 8-15: „Orientierung variieren“, 27: "Wiederholung"

Erfolgsrate fiele durch mehrere frühe Erfolge nicht unter die globalen 56,1%, werden die besseren Erfolgswahrscheinlichkeiten anderer Strategien nicht erkannt. Das Potential für Verbesserungen ist hier deutlich höher und zeigt sich in der stärkeren Reduktion der mittleren Behandlungsdauer.

Der positive Effekt der verwendeten Zusatzstrategie kann auch anhand der unterschiedlichen Streuung verschiedener Einzelmessungen verdeutlicht werden. Das Diagramm 7.24a zeigt anhand der Verläufe der zehn Einzelmessungen ohne Zusatzstrategie für die Applikation „Zylinder lagerichtig zuführen“, dass die erreichten Behandlungszeiten zum Teil stark von den besten Ergebnissen abweichen. Das Diagramm 7.24b verdeutlicht, dass der Einsatz einer Zusatzstrategie Streuung der Werte stark reduziert. Dass die Einzelmessungen nicht alle exakt denselben Wert anstreben, liegt hauptsächlich an der zufälligen Auswahl der Extrastrategie und den daraus resultierenden statistischen Abweichungen. Angedeutet ist auch die mittlere Behandlungsdauer nach 1850 Störungen, die am unteren Rand der Streuung liegt, wenn die Zusatzstrategie nicht mehr ausgeführt würde.

Die Ergebnisse von beiden Applikationen zeigen, dass der Aufruf einer Zusatzstrategie zu einer Verbesserung der statistischen Daten führt. An den Abbildungen 7.20und 7.21 ist jedoch auch zu erkennen, dass sich die absolute Behandlungsdauer erst verringert, wenn die Extrastrategie ab einer geeigneten Störungsanzahl nicht mehr aufgerufen wird. Der entsprechende Zeitpunkt ist applikationsabhängig und muss daher aus dem Verlauf der Behandlungsdauer ermittelt werden. Hierzu ist die eintretende Sättigung der Optimierung bzw. eine nur noch sehr geringe Verbesserung zu detektieren. In den Abbildungen 7.20und 7.21 entspricht dies dem Index, ab dem der Verlauf der mittleren Behandlungsdauer über mehrere Störungen nicht mehr oder nur noch wenig und gleichförmig sinkt.

Zur groben Berechnung eines günstigen Zeitpunkt kann zunächst die mittlere Behandlungsdauer über alle zurückliegenden Störungen mit der Formel 7.1 berechnet werden. Anschließend wird für den aktuelle Index *i* die Steigung des Verlaufs über zehn Störungen mittels Formel 7.2 bestimmt und gespeichert.

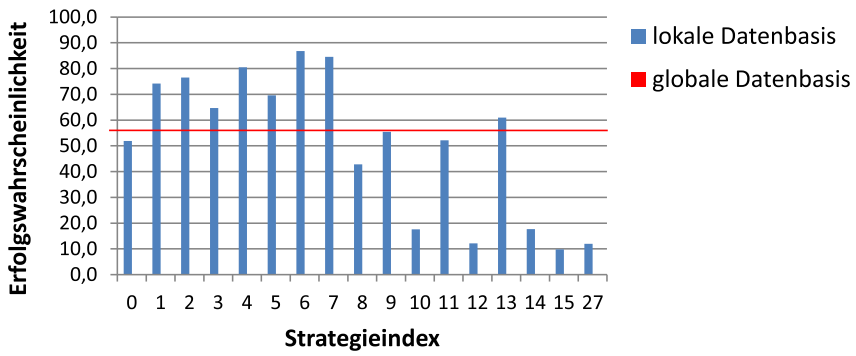


Abb. 7.23.: Lokale und globale Erfolgswahrscheinlichkeiten der zugewiesenen Strategien für Applikation III. Strategieindex 0-7: „Position variieren“, 8-15: „Orientierung variieren“, 27: "Wiederholung"

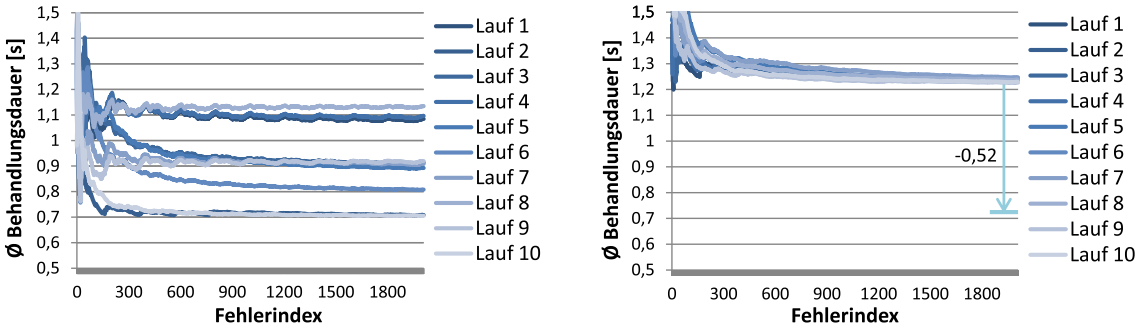


Abb. 7.24.: Entwicklung der mittleren Behandlungsdauer bei zehn Messungen: a) ohne Ausführung einer Zusatzstrategie, b) mit Ausführung einer Zusatzstrategie. Eingetragen ist in b) auch die erreichte Behandlungsdauer nach Aussetzen der Zusatzstrategie

$$g_i = (t_{i-10} - t_i)/10 \qquad \text{mit } t_j \text{ als Behandlungsdauer für Störung j} \tag{7.2}$$

Um zu erkennen, wann sich diese Steigerung nicht mehr stark verändert, wird wiederum die Steigerung über den Verlauf von g mittels der Formel 7.3 berechnet.

$$h_i = (g_{i-20} - g_i)/20 \tag{7.3}$$

Abschließend wird überprüft, ob die Beträge zwischen h_{i-30} und h_i alle jeweils unter $1 \cdot 10^{-4}$ liegen. In dem Fall würden keine weiteren Zusatzstrategien ausgeführt.

Diese Berechnung ermöglicht eine grobe Schätzung eines geeigneten Zeitpunkts zum Abschalten der Zusatzstrategie. Die enthaltenen Parameter sind empirisch ermittelt. Die betrachteten Intervallbreite für g_i ist so gewählt, dass Ausreißer zum einen weniger stark ins Gewicht fallen, zum anderen aber auch lokale Schwankungen noch erfassbar sind. Die steigende Intervallbreite bei der Berechnung von h_i und der Grenzwertüberprüfung ist wiederum nötig, damit die Ausführung einer Zusatzstrategie nicht bereits bei zufällig auftretenden, kurzen und waagerechten Verläufen ausgesetzt wird. Zur Konzeption eines detaillierteren Bewertungsalgorithmus sind weitere Untersuchungen zum Verlauf der Behandlungsdauer bei unterschiedlichen Konstellationen nötig. Mit dem beschriebenen Vorgehen wird die Ausführung einer Zusatzstrategie bei Applikation I nach 341 Störungen und bei Applikation III nach 102 Störungen gestoppt. Da die Optimierung der statistischen Daten zu diesen Zeitpunkten vollständig bzw. nahezu abgeschlossen ist (siehe Abbildungen 7.20und 7.21), wären die Ergebnisse ähnlich zu den Beispielrechnungen.

Fazit:

Der Einsatz einer Initialisierung bei den ersten Störungen bringt keinen oder nur einen sehr geringen absoluten Vorteil und wird nicht weiter betrachtet. Die Ausführung einer zufälligen Zusatzstrategie kann die Behandlungsdauer deutlich verringern, indem Ausreißer nach oben vermieden werden. Da die Zeitreduktion bei einer Störungsbehandlung in der Regel kleiner ist als die benötigte Ausführdauer einer Zusatzstrategie, tritt ein Vorteil jedoch erst ein, sobald die Extraausführung

nicht mehr aufgerufen wird. Das Abschalten ist ab einem bestimmten Zeitpunkt jedoch generell empfehlenswert, da die Verbesserung der statistischen Datenbasis in eine Sättigung läuft. Über einen sehr simplen Algorithmus konnten die Abschaltzeitpunkte bereits grob geschätzt werden. Um eine maximale absolute Zeitreduktion zu erreichen, bedarf es eines Regelalgorithmus, der den geeigneten Moment zum Umschalten deutlich genauer ermittelt. Mögliche Eingangsgrößen zur Detektion der Sättigung sind neben der Steigung im Verlauf der mittleren Behandlungsdauer ggf. auch die globalen und lokalen Erfolgswahrscheinlichkeiten der Strategien sowie die Anzahl ihrer bisherigen Aufrufe.

7.4.2. Evaluierung der erfahrungsbasierten Reduktion der Behandlungsdauer

Die erfahrungsbasierte Strategieauswahl kann die zur Behandlung auftretender Störungen benötigte Zeitdauer stark reduzieren. Die erreichbare Verbesserung hängt zum einen von der Güte der für die Strategien erfassten Erfolgswahrscheinlichkeiten ab. Zum anderen haben statistischen Abhängigkeiten innerhalb der Erfolgsraten sowie deren Relation zueinander einen großen Einfluss. Die Evaluierung der erfahrungsbasierten Strategieauswahl erfolgt anhand der Applikationen „Bremszylinder umsetzen“ (I) und „Zylinder lagerichtig zuführen“ (III), da diese wie in Abschnitt 7.4.1 gezeigt, sehr unterschiedliche Randbedingungen aufweisen. Um statistische Schwankungen zu reduzieren, wird die Behandlungsdauer jeweils über drei Einzelmessungen gemittelt. Die Störungsbehandlung wird zuvor auf Basis von Fehlerfällen simuliert, die mit dem Experimentiersystem an realen Applikationen in Dauerläufen aufgenommen wurden. Dies ermöglicht eine hohe Vergleichbarkeit der Ergebnisse innerhalb unterschiedlicher Applikationsszenarien.

Applikation I „Bremszylinder umsetzen“
Eine wichtige Voraussetzung für die sinnvolle Verwendung einer erfahrungsbasierten Optimierung der Behandlungssequenz ist das Auftreten einer ungleichmäßigen Verteilung der Erfolgswahrscheinlichkeiten. Das Diagramm 7.25 verdeutlicht dies für die zugewiesenen Strategien bei Applikation I. Eingesetzt werden die Lokalisierungsstrategien „Position variieren“ (Index 0 - 7), „Orientierung variieren“ (8 - 15) sowie „Lageerkennung unverändert wiederholen“ (27). Die Werte für $P(i)$ zeigen die über den vollständigen Datensatz berechneten Erfolgsraten, wenn die jeweiligen Strategien als erste Maßnahme der Behandlungssequenz ausgeführt werden. Die Daten zu $(P(i|6))$ verdeutlichen ein ungleichmäßiges Absinken der Wahrscheinlichkeiten, wenn die Behandlungsvarianten als zweites nach der zuvor besten Strategie 6 aufgerufen werden. Beide Diagramme weisen deutliche Schwankungen zwischen den Erfolgsraten der Strategien auf. Eine gezielte Planung der Reihenfolge kann daher deutliche Vorteile gegenüber einer zufälligen Auswahl ermöglichen.

Für die Bewertung der erfahrungsbasierten Strategieauswahl werden die benötigten Zeiten zur Behandlung der unterschiedlichen Fehlerarten analysiert und mit denen einer zufälligen Strategiereihenfolge verglichen. Die Tabelle 7.27 zeigt die Ergebnisse der Auswertung für die Applikation

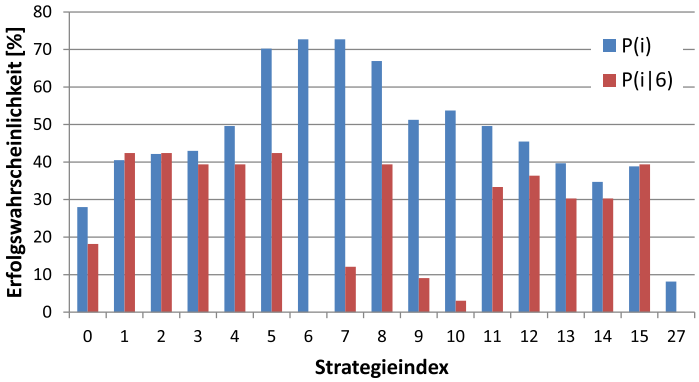


Abb. 7.25.: Darstellung der bedingten Wahrscheinlichkeiten

„Bremszylinder umsetzen“. Neben den genannten Lokalisierungsmöglichkeiten wurden auch die Strategien *Anwesenheitsprüfung durch blinden Griff*, *Bauteil entfernen* und *Nächste Palettenposition verwenden* eingesetzt. Da die Behandlungsdauer abhängig von den bei einer Applikation anwendbaren Strategietypen variiert, werden mit den genannten Strategien drei unterschiedliche Szenarien überprüft. Die erste Variante enthält nur Lokalisierungsstrategien. Bei der 2. Option kann zusätzlich die Objektanwesenheit geprüft werden, wohingegen das Bauteil im letzten Szenario auch entfernt und durch neue Werkstücke ersetzt werden kann.

Wie in Abschnitt 7.4.1 gezeigt, kann die Verwendung einer Zusatzstrategie zu einer deutlichen Reduktion der Behandlungsdauer führen. Die Auswertung umfasst daher jeweils die benötigte Behandlungsdauer ohne (O) sowie mit (E) Aufruf einer Extrastrategie. Außerdem ist die Behandlungsdauer angegeben, wenn beide Ansätze kombiniert werden (K) und die Zusatzstrategie nicht weiter ausgeführt wird, sobald der Informationszugewinn keine Verbesserung mehr hervorruft (vgl.

Behandlungs-szenario	Fehlerart	∅ Behandlungsdauer [s]				Zeitreduktion bzgl. zufälliger Wahl [%]			Behandlungen erfolglos [%]	
		R	O	E	K	O	E	K	R	O / E / K
1. Lokalisierung	Licheffekte auf Bauteil	3,43	2,53	3,17	2,41	26,0	7,4	29,6	9,8	9,8
2. Lokalisierung, Prüfung	Licheffekte auf Bauteil	3,40	1,75	2,20	1,74	48,4	35,7	48,7	0,5	0,0
3. Lokalisierung, Prüfung, Beschaffen, Entfernen	Licheffekte auf Bauteil	3,40	2,03	2,32	1,86	40,1	31,8	45,2	0,5	0,0
	kein Bauteil	a) 10 b) 24		7,30			a) 27 b) 69,6		0,0	0,0
	Lage nicht erkennbar	24		17,83			25,7		0,5	0,0

Tab. 7.27.: Überblick über die Ergebnisse der Dauerlaufversuche mit verschiedenen Fehler- und Strategieszenarien für Applikation I. Bedeutung R: zufällige Strategiereihenfolge, E: berechnet mittels POMDP mit Zusatzstrategie, O: ohne Zusatzstrategie, K: bei Abschalten der Zusatzstrategie nach Optimierung

Abschnitt 7.4.1). Wie die Abbildung 7.26 verdeutlicht, ist dieser Zeitpunkt nach circa 300 Störungen erreicht, da die Behandlungsdauer ohne Zusatzausführung anschließend im gleichen Maße sinkt. Die Tabelle 7.27 zeigt jeweils die mittlere Behandlungsdauer zwischen der 950. und 1050. Störung, da sich die Ergebnisse nachfolgend nahezu nicht mehr verbessern. Zur Bewertung der erfahrungsbasierten Planung ist die mittlere Behandlungsdauer bei zufälligen Strategiesequenzen (R) angegeben. Diese wurde als Mittelwert über 50 zufällig generierte Reihenfolgen gemessen. Zusätzlich gibt die Auswertung auch die prozentuale Zeitreduktion für die erfahrungsbasierte Planung der Strategiereihenfolge im Vergleich zur zufälligen Auswahl an. Weiterhin ist der Anteil an Störungen angegeben, der im jeweiligen Szenario nicht behandelt werden konnten.

Die Tabelle 7.27 zeigt, dass die erfahrungsbasierte Planung der Strategiereihenfolge zu einer deutlich kürzeren Behandlungsdauer führt als bei einer zufälligen Auswahl. Die erreichte Zeitreduktion liegt je nach Behandlungsszenario und abhängig vom Einsatz einer Zusatzstrategie zwischen 26,0% und 48,7%. Da die Behandlungsdauer ohne Extraausführung stets geringer ausfällt als mit, sollte die Zusatzstrategie im Rahmen der kombinierten Methode nur bis etwa zur 300. Störung eingesetzt werden. Auch mit der Extraausführung ist die Behandlungsdauer aber bereits geringer als bei einer zufälligen Auswahl.

Das Auftreten einer jeweils unterschiedlichen Behandlungsdauer bei Lichteffekten auf dem Bauteil folgt aus den verschiedenen Optionen innerhalb der untersuchten Szenarien. Werden nur Lokalisierungsstrategien eingesetzt, können 9,8% der Störungen nicht behandelt werden. Durch die erfolglose Ausführung aller Strategien steigt die benötigte Zeit und wirkt sich auf die durchschnittliche Behandlungsdauer aus. Der Einsatz einer Prüfstrategie, mit der gleichzeitig die Bauteillage verändert wird, reduziert die Anzahl unbehandelter Störungen deutlich. Die Prüfstrategie wird bei einer zufällig bestimmten Reihenfolge durchschnittlich etwas zu früh und damit insgesamt zu häufig ausgeführt. Es werden zwar alle Strategien behandelt aber die mittlere Dauer hierzu sinkt auf Grund der langen Ausführungszeit der Prüfung nahezu nicht. Die erfahrungsbasierte Auswahl ermöglicht hingegen durch die Berechnung eines optimalen Ausführungszeitpunktes und reduziert die Behandlungszeit deutlich. Im dritten Szenario können Bauteile mit einer angenommenen Feh-

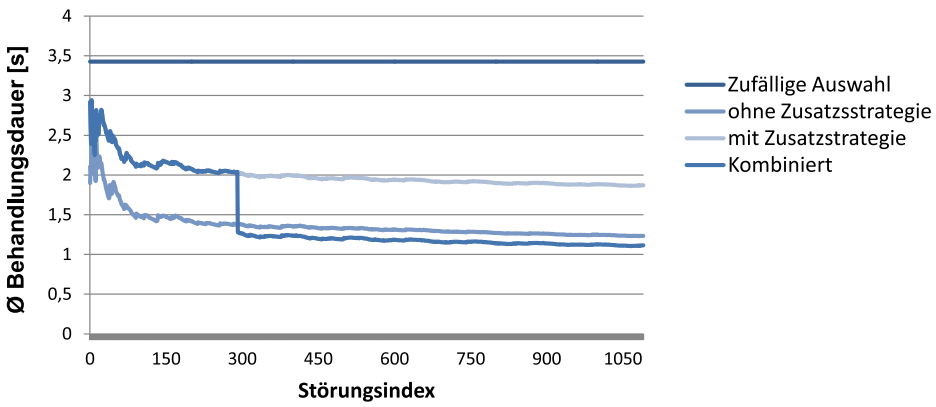


Abb. 7.26.: Vergleich der mittleren Behandlungsdauer bei verschiedenen Modi für Applikation I

lerquote von 3% auch fehlen oder nicht lokalisierbar sein. Die mittlere Behandlungsdauer bei Lichteffekten auf den Werkstücken steigt dabei an, da die Anwesenheitsprüfung bereits deutlich eher im Ablauf ausgeführt wird.

Bei diesem Szenario zeigt sich ein weiterer Vorteil der erfahrungsbasierten Strategieauswahl. Durch die Erfassung der Erfolgsrate der Prüfstrategie und die Schätzung des aktuellen Fehlers zur Laufzeit, kann automatisch bewertet werden, ob das Ergebnis einer Prüfung eindeutig ist bzw. wie häufig eine Ausführung statistisch sinnvoll ist. Bei einer zufälligen Strategieauswahl ergeben sich hier mehrere Konflikte. Für die Behandlung fehlender Bauteile ist es relevant, ob der Testgriff eindeutig ist. Ein vorhandenes Bauteil würde sicher detektiert und fehlende Objekte können sofort behandelt werden (Behandlungsdauer a)). Kann der Testgriff fehlschlagen, lässt sich ohne Erfahrungswissen nicht entscheiden, wie viele Wiederholungen zur Vermeidung von Fehlbehandlungen sinnvoll sind. Für eine höhere Sicherheit sollten alle Lokalisierungsstrategien mindestens noch ein zweites Mal ausgeführt werden (Behandlungsdauer b). Da ein einziger Testgriff bei Lichteffekten auf dem Bauteil in 0,5% der Fälle nicht ausreichte, war die Behandlung bei der zufälligen Auswahl nicht immer erfolgreich. Bei Verwendung einer Entfernen-Strategie führt dies zu einer unnötigen Entnahme eines Bauteils. Um dies zu verhindern, kann mehrfach geprüft werden. Hierbei steigt die Behandlungszeit im Falle eines nicht lokalisierbaren Bauteils jedoch stark an, da zwischen jeder Prüfung alle Lokalisierungsstrategien ausgeführt werden müssten. Entsprechend ergeben sich bei dieser Applikation für die Behandlung fehlender oder nicht erkennbarer Bauteile deutliche Vorteile beim Einsatz der erfahrungsbasierten Strategieauswahl. Die Reduktion der Behandlungsdauer liegt abhängig von den Randbedingungen und den gewählten Strategien zwischen 25,7% und 69,6%.

Die erfahrungsbasierte Strategieauswahl führte bei allen betrachteten Fehlerarten und Behandlungsszenarien gegenüber einer zufälligen Reihenfolge zu einer Reduktion der Behandlungsdauer um 20,1% bis 65,5%. Gegenüber den ohne Störungsbehandlung erforderlichen Bedienereingriffen ist der Vorteil noch deutlich größer. Im regulären Betrieb der Versuchsplattform in der Fertigung vergingen im Durchschnitt 4 min zwischen der Aufforderung zum Wechseln einer Palette und der Quittierung der Nachricht. Der Bedienereingriff war durch Verwendung zweier Paletten nicht zeitkritisch. Dafür war der jeweilige Zeitpunkt anhand des Palettenfüllstands abzuschätzen. Nimmt man die gemessenen 4 Minuten daher als Richtwert für die Dauer bis zum Bedienereingriff, ergibt sich sogar eine Zeitreduktion durch die Störungsbehandlung von 92,6% bis 99,2%.

Applikation III: „Zylinder lagerichtig zuführen“

Zur weiteren Evaluierung wurde das Verhalten der erfahrungsbasierten Strategieauswahl an der Applikation „Zylinder lagerichtig zuführen“ unter denselben Randbedingungen wie bereits zuvor überprüft. Eingesetzt wurden die Strategien *Position variieren*, *Bildaufnahmeparameter variieren*, *Lokalisierung unverändert Wiederholen*, *Bauteil entfernen*, *Nächste Palettenposition verwenden* sowie eine Anwesenheitsprüfung mit blindem Testgriff und einer aktiven Verdrehung des Bauteils um 20° zur Senkrechten. Wie die Tabelle 7.28 zeigt, konnte die Behandlungsdauer verglichen mit einer zufälligen Strategieauswahl um 16,5% bis 60% abhängig vom Behandlungsszenario reduziert

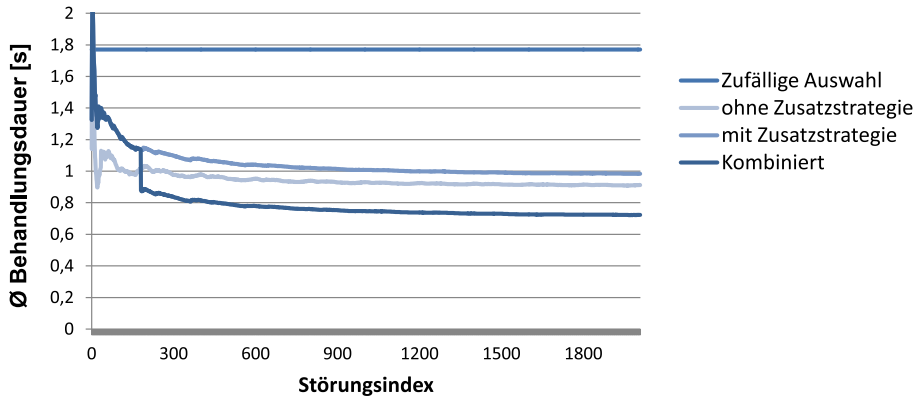


Abb. 7.27.: Vergleich der mittleren Behandlungsdauer bei verschiedenen Modi für Applikation III

werden. Durch die Verwendung einer Zusatzstrategie können die statistischen Daten klar optimiert werden. Wie in Abschnitt 7.4.1 gezeigt, ist die Konstellation der Erfolgswahrscheinlichkeiten bei dieser Applikation für eine schnelle Optimierung sehr günstig. Damit sich der Zusatzaufwand rentiert, sollten jedoch keine Extrastrategien mehr ausgeführt werden, sobald sich das Erfahrungswissen nicht mehr verbessert. Dieser Zeitpunkt liegt bei circa 170 Störungen, wie die Abbildung 7.27 an den Verläufen der mittleren Behandlungsdauer bei den unterschiedlichen Optionen verdeutlicht.

Bei dieser Applikation konnten alle Störungen, die durch Lichteffekte auf dem Bauteil verursacht werden, über die Lokalisierungsstrategien erfolgreich behandelt werden. Aus diesem Grund bewirkt der Einsatz einer Anwesenheitsprüfung zur Veränderung der Bauteillage keine weitere Zeitreduktion. Der leichte Vorteil bei der nachfolgenden Lokalisierung wird durch den Extraaufwand

Behandlungs-strategien	Fehlerart	∅ Behandlungsdauer [s]				Zeitreduktion bzgl. zufälliger Wahl [%]		
		R	O	E	K	O	E	K
1. Lokalisierung	Licheffekte auf Bauteil	1,13	0,90	1,22	0,70	20,1	-8,3	38,0
2. Lokalisierung, Prüfung	Licheffekte auf Bauteil	1,77	0,89	0,97	0,71	49,7	45,2	60,0
3. Lokalisierung, Prüfung, Beschaffen, Entfernen	Licheffekte auf Bauteil	1,77	0,98	1,06	0,78	44,6	40,1	56,1
	kein Bauteil	a) 9,35 b) 18,7		6,44			a) 31,1 b) 65,5	
	Lage nicht erkennbar	18,69		15,61			16,5	

Tab. 7.28.: Überblick über die Ergebnisse der Dauerlaufversuche mit verschiedenen Fehler- und Strategieszenarien für Applikation III. Bedeutung R: zufällige Strategiereihenfolge, E: berechnet mittels POMDP mit Zusatzstrategie, O: ohne Zusatzstrategie, K: bei Abschalten der Zusatzstrategie nach Optimierung

der Prüfung nahezu genau kompensiert. Bei der zufälligen Auswahl wird die Behandlungsdauer sogar erhöht, da die Anwesenheitsprüfung bei einigen ermittelten Reihenfolgen sehr früh erfolgt aber keinen deutlichen Vorteil bringt. Bei der erfahrungsbasierten Optimierung wird diese Strategie im ersten und zweiten Behandlungsszenario erst deutlich später ausgeführt. Im dritten Szenario können fehlende oder nicht lokalisierbare Bauteile mit einer Wahrscheinlichkeit von 3% auftreten. Die resultierenden Störungen konnten alle erfolgreich behandelt werden. Die durchschnittliche Behandlungsdauer lag zwischen 16,5% und 65,5% unter der einer zufälligen Strategieauswahl. Um die beiden zusätzlichen Fehlerfälle früh zu erkennen, wird die Anwesenheitsprüfung eher als zuvor ausgeführt, wodurch die Behandlungsdauer bei Lichteffekten leicht ansteigt.

Die erfahrungsbasierte Fehlerschätzung und Ausführungsplanung erreicht auch bei dieser Applikation mit einer Reduktion der Behandlungsdauer um 16,5% bis 65,5% gegenüber der zufälligen Strategiereihenfolge sehr gute Ergebnisse. Der so erzielbare Vorteil kann insbesondere bei Applikationen mit hoher Störungsrate oder sehr strengen Taktzeitbedingungen über die wirtschaftliche Automatisierbarkeit mit einem flexiblen Robotersystem entscheiden. In jedem Fall wird eine Steigerung der Produktivität erreicht. Geht man von einer mittleren Dauer von 4 Minuten zwischen der Ausgabe einer Störungsmeldung und der manuellen Behebung des Problems aus, ergibt sich im Störungsfall sogar eine Zeitreduktion von 93,5% bis 99,7% gegenüber den ohne Störungsbehandlung erforderlichen Bedienereingriffen.

7.4.3. Evaluierung der Berechnungsdauer der optimalen Policy

Durch den Einsatz von Erfahrungswissen zur Fehlerschätzung und Ausführungsplanung ist die optimale Policy zyklisch neu zu berechnen, wenn die statistische Datengrundlage bei einer behandelten Störung erweitert wurde. Im Extremfall können Störungen in zwei aufeinander folgenden Pick-and-Place-Zyklen auftreten. Dies entspricht in der Regel einem zeitlichen Mindestabstand von ungefähr 6 - 8 Sekunden. Um eine stets aktuelle optimale Policy zu ermöglichen, muss diese daher innerhalb weniger Sekunden berechenbar sein.

Die Abbildungen 7.28a und 7.28b zeigen den Verlauf der gemessenen Berechnungsdauer für die optimale Lösung am Beispiel der Applikationen I und III. Die Messungen wurden auf einem Notebook mit Intel Core i7-720QM mit 1.6 GHz ausgeführt. Bei Applikation I wurden die Strategien *Position variieren*, *Orientierung variieren*, *Lokalisierung unverändert Wiederholen*, *Anwesenheitsprüfung durch blinden Griff*, *Bauteil entfernen* und *Nächste Palettenposition verwenden* zugewiesen. Der Discount-Faktor γ lag, wie auch bei der Bewertung zuvor, bei 0,3 und es stellte sich eine maximale Sequenzlänge von 26 Strategien ein. Die gemessenen Berechnungszeiten schwanken und werden für eine übersichtlichere Darstellung in Abbildung 7.28a jeweils über die letzten 20 Messungen gemittelt. Die durchschnittliche Berechnung dauert 0,69s bei einer Standardabweichung von 0,19s. Entsprechend der 3-Sigma-Umgebung sind damit 99,7% der Kalkulationen innerhalb von 1,26s abgeschlossen. Die Dauer scheint weitestgehend unabhängig von der Knotenmenge, die

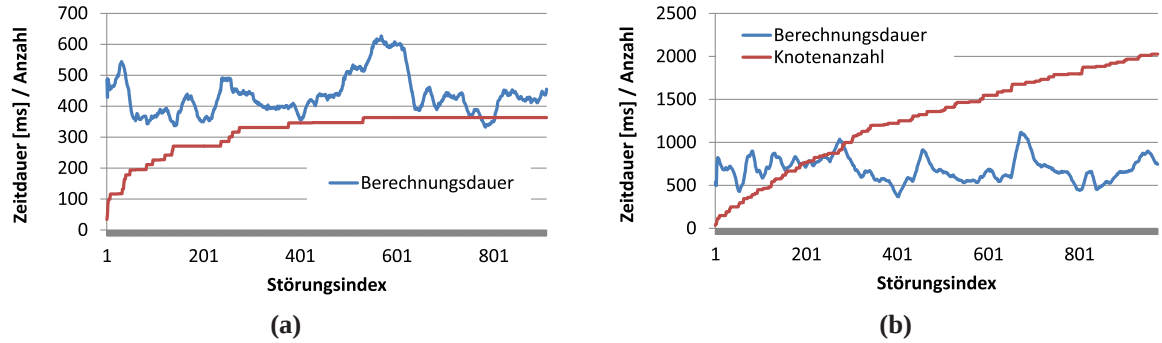


Abb. 7.28.: Messung der Dauer zur Berechnung der optimalen Policy (gemittelt jeweils über 20 Messungen): a) für Applikation I, b) für Applikation II

sich bei zunehmender Störungsanzahl einem Maximum annähert. Eine Berechnung innerhalb eines Pick-and-Place-Zyklus ist möglich.

Die Abbildung 7.28b zeigt die ebenfalls über 20 Werte gemittelten Berechnungszeiten für Applikation III. Die zugewiesenen Strategien umfassten *Position variieren*, *Bildaufnahmeparameter variieren*, *Lokalisierung unverändert Wiederholen*, *Bauteil entfernen*, *Nächste Palettenposition verwenden* sowie eine Anwesenheitsprüfung mit blindem Testgriff und einer aktiven Verdrehung des Bauteils um 20° zur Senkrechten. Der Discount-Faktor γ lag bei 0,1 und es stellte sich eine maximale Sequenzlänge von 32 Strategien ein. Auch bei dieser Applikation ist eine Berechnung der optimalen Policy mit einer mittleren Dauer von 0,43s zwischen zwei Pick-and-Place-Zyklen möglich. Bei einer Standardabweichung von 0,09s ist die Berechnung in 99,7% der Störungen entsprechend des 3-Sigma-Bereichs innerhalb von 0,71s abgeschlossen. Die Anforderungen an die maximale Berechnungsdauer werden damit deutlich erfüllt.

7.5. Zusammenfassung

Die Evaluierung der MMS sowie der Störungsbehandlung erfolgte mit einem flexiblen Robotersystem an verschiedenen Applikationen aus der Fertigung der Robert Bosch GmbH.

Die intuitive und schnelle Programmierung ohne Erfahrungswissen wurde in Usability-Tests mit acht Teilnehmern aus der Zielgruppe bestätigt, von denen die Hälfte Einsteller aus der Fertigung bei Bosch sind. Alle Probanden haben je eine von zwei komplexen Applikationen nach kurzer Einführung eigenständig und erfolgreich programmiert. Die erforderliche Zeit lag zwischen 1,5 und 3 Stunden und stellt damit eine Reduktion um ca. 90% gegenüber einer herkömmlichen Inbetriebnahme durch einen Experten auf Basis des vorhandenen Roboter-Frameworks dar. Die Bewertung der MMS durch die Teilnehmer war sehr positiv.

Die Optimierung der Dauerlaufrobustheit mittels einer strategiebasierten Störungsbehandlung wurde an sechs störungsanfälligen Applikationen gezeigt. Durch das vorgestellte Konzept konnten Ausfälle des Robotersystems durch Störungen bei der Manipulation oder der Objektlageerkennung

um 92% bis 100% reduziert werden. Die Inbetriebnahme dauerte jeweils nur wenige Minuten, wodurch das geringe Aufwand-Nutzen-Verhältnis bestätigt wurde.

Anhand von Dauerläufen mit zwei Applikationen und unterschiedlichen Störungskonstellationen konnte gezeigt werden, dass die erfahrungsbasierte Fehlerschätzung und Ausführungsplanung mit einem POMDP die Behandlungsdauer bei einem reinen Einsatz von Lokalisierungsstrategien um über 40% im Vergleich zu einer zufälligen Strategiereihenfolge reduziert. Gleichzeitig wird die Berücksichtigung verschiedener Fehlertypen ermöglicht. Die Behandlungsdauer bei fehlenden oder nicht lokalisierbaren Objekten konnte durch den Einsatz des gelernten Erfahrungswissens abhängig von den Randbedingungen um 16% bis 70% gegenüber der zufälligen Strategieauswahl verringert werden. Die mittlere Dauer zur Berechnung der optimalen Policy lag bei typischen Applikationen und einem handelsüblichen Notebook in 99,7% der Fälle unter 1,26 Sekunden und ermöglicht somit ein dynamisches Update im laufenden Betrieb. Die Störungsbehandlung erfüllt damit alle relevanten Anforderungen und kann das wirtschaftlich automatisierbare Applikationsspektrum von flexiblen Robotersystemen in der Kleinserienmontage deutlich erhöhen.

8. Schlussbetrachtung

Herkömmliche Automatisierungslösungen sind für die Kleinserienmontage nicht wirtschaftlich einsetzbar, da sie bei jedem Wechsel der Produktion aufwändig mechanisch angepasst sowie langwierig umprogrammiert und optimiert werden müssen. Einen möglichen Ansatz bieten flexible, für unterschiedliche Aufgaben verwendbare Robotersysteme. Die Programmierung ist mit herkömmlichen Methoden jedoch ebenfalls sehr aufwändig. Gleichzeitig sind die universellen Komponenten störungsanfälliger als Speziallösungen, wodurch ein robuster Dauerbetrieb oft nur mit aufwändiger Optimierung oder einer Anpassungen der Applikation erreichbar ist.

Das Ziel dieser Arbeit war die Entwicklung einer intuitiven Mensch-Maschine-Schnittstelle mit der flexible Robotersysteme schnell und ohne Expertenwissen für komplexe Pick-and-Place-Applikationen programmiert werden können. Dabei soll der Bediener die Robustheit der erstellten Programmabläufe optimieren können, um für einen wirtschaftlichen Einsatz der Systeme einen ausfallfreien Dauerbetrieb zu erreichen. Der Abschnitt 8.1 fast den entwickelten Lösungsansatz und die erzielten Ergebnisse dieser Arbeit zusammen. Im Abschnitt 8.2 werden vorhandene Limitierungen und mögliche Ansatzpunkte für eine Weiterentwicklung des Konzepts diskutiert.

8.1. Zusammenfassung

Für die Anforderungsanalyse wurden im Rahmen dieser Arbeit 52 typische Applikationen sowie die 45 enthaltenen Bauteile untersucht. Die mit einem flexiblen Robotersystem umsetzbaren Anwendungsfälle bestehen aus Pick-and-Place-Abläufen zum Be- und Entladen von Maschinen oder zum Palettieren und Verpacken von Werkstücken. Die Manipulation erfolgt dabei häufig mit nur sehr geringem Spiel zwischen Bauteilen und Aufnahme. Die Eigenschaften und Merkmale der Werkstücke unterscheiden sich ähnlich wie deren Zuführung zum Prozess stark. Für die anforderungskonforme Automatisierung der entsprechend vielfältigen Applikationen sind jeweils Komibinationen verschiedener Sensorsysteme mit unterschiedlichen Algorithmen robust auszuwählen und einzurichten.

Um die Robustheit des autonomen Montagebetriebs von flexiblen Robotern zu untersuchen, wurde eine Fehlermöglichkeits- und Einflussanalyse für typische Systemausprägungen durchgeführt. Als mögliche Quelle für Ausfälle oder andere Beeinträchtigungen des Dauerbetrieb wurden vor allem Störungen bei der Objektlageerkennung und Bauteilmanipulation identifiziert. Die ursächlichen Fehler lassen sich nach ihrer Herkunft in zwei Klassen aufteilen. Fehler bei der Inbetriebnahme und Programmierung der Applikation gehen auf den Bediener zurück und müssen über

geeignete Maßnahmen innerhalb der MMS verhindert werden. Die zweite Kategorie umfasst Fehler durch prozessbedingte Schwankungen der Randbedingungen, manuelle Eingriffe in den Prozess oder technische Defizite der Robotersysteme, die wirtschaftlich nicht weiter reduzierbar sind. Alle drei Störungsursachen sind nicht verhinderbar und müssen zur Vermeidung von Ausfällen den Anforderungen entsprechend in zulässige Systemzustände überführt werden.

Der Beitrag dieser Arbeit liegt in der Entwicklung einer Prozesskette und der enthaltenen Komponenten zur schnellen und robusten Programmierung flexibler Robotersysteme für objektgeführte Handhabungsaufgaben. Neben der Inbetriebnahme ohne Expertenwissen liegt der Fokus vor allem auf einer Steigerung der Robustheit bei der Ausführung der erstellten Programmabläufe. Hierzu werden unvermeidbare Fehler unterbrechungsfrei über verschiedene Strategien behandelt, die vom Bediener im Vorfeld entsprechend den Anforderungen der Applikation ausgewählt und eingerichtet werden. Die Ausführung wird zur Laufzeit anhand von autonom gelerntem Erfahrungswissen mittels eines Partially Observable Markov Decision Process (POMDP) geplant. Dies ermöglicht eine Behandlung verschiedener Fehlertypen bei gleichzeitig kürzerer Ausführdauer im Betrieb.

Das Einrichten der flexiblen Robotersysteme für Montageaufgaben erfolgt in drei Schritten. Der Bediener erstellt zunächst ein Arbeitsplanskelett des Montageablaufs mittels iconbasierter Programmierung. Die nachfolgende Parametrierung der enthaltenen Arbeitsschritte erfolgt für eine unterbrechungsfreie Bedienerführung vollständig wizardbasiert. Bei den komplexen Greif- und Ablege-Aktionen ist zuerst die Objektlokalisierung zum Ausgleich von Lagevarianzen einzurichten. Aufgrund der Vielfalt an Bauteilen und ihrer Zuführung zum Prozess besteht die Herausforderung hierbei in der Auswahl einer geeigneten Sequenz von häufig mehreren Lokalisierungsschritten sowie in der robusten Parametrierung der verwendeten Algorithmen. Ausgehend von einer Analyse aller gebräuchlichen Kombinationen anhand eines exemplarischen Sensorsystems wird die Auswahl der Lokalisierungssequenz über einen Entscheidungsbaum auf die Beantwortung weniger Fragen zur Applikation vereinfacht. Für die anschließende Parametrierung der Algorithmen werden dem Nutzer die Auswirkungen der Einstellungen anhand von intuitiv verständlichen Visualisierungen angezeigt, die aus Zwischenschritten der Bildverarbeitungskette aufbereitet werden. Die Darstellungen ermöglichen dem Nutzer eine Bewertung und Optimierung der Einstellungen, ohne die Bedeutung der Parameter zu kennen. Nach der Lokalisierung werden die Bewegungen zur Bauteilmanipulation durch Teach-In definiert, wobei die Herausforderung im Erreichen der erforderlichen Positioniergenauigkeit liegt. Ausgehend von drei identifizierten Klassen an Roboterposen wurden spezielle Steuerelemente entworfen, die dem Bediener anhand von verständlichen Darstellungen eine intuitive Zuordnung der Bewegungsrichtungen am System ermöglichen. Dies ermöglicht ein schnelles und ausreichend genaues Positionieren von Roboterarm, Kamerasensor sowie Greifer.

Zur Steigerung von Robustheit und Autonomie bei der Ausführung der so erstellten Arbeitspläne sind Ausfälle durch die identifizierten Störungen bei der Lageerkennung und Bauteilhandhabung zu verhindern. Für die Detektion von Fehlern und Störungen werden das Lokalisierungsergebnis und

die Greiferöffnungsweite bei gefasstem Bauteil überprüft. Eine geeignete Behandlung ist im Störungsfall aus den unspezifischen Systemrückmeldungen nicht ableitbar. Zusätzlich sind die erlaubten Maßnahmen zumeist applikationsabhängig. Um alle Randbedingungen und Anforderungen einzuhalten, erfolgt die Auswahl der anwendbaren Maßnahmen durch den Bediener anhand von dessen Applikationswissen. Tritt im Montagebetrieb eine Störung im Prozessablauf auf, kann das System autonom auf die so zugewiesenen Strategien zugreifen, um wieder einen störungsfreien Zustand zu erreichen. Da in einer Applikation häufig verschiedene Fehler auftreten, können mehrere Behandlungsvarianten ausgewählt werden. Für eine effiziente Inbetriebnahme im Rahmen der MMS wurde eine Datenbank mit Behandlungsstrategien für alle identifizierten Fehlerfälle innerhalb des Anwendungsszenarios umgesetzt. Die Strategien sind jeweils über nur wenige, verständlichen Parametern schnell und ohne Expertenwissen an die Randbedingungen der Applikation anpassbar. Um erwartete Störungen bereits vor dem ersten Ausfall zu behandeln, kann die Auswahl geeigneter Strategien auch im Rahmen der Programmierung des Roboters erfolgen. Ist bei einer Störung im Montagebetrieb noch keine geeignete Behandlungsmöglichkeit zugewiesen, stoppt das System und der Bediener kann eine passende Strategie mit Unterstützung durch die MMS auswählen. Der Dauerbetrieb wird durch die zusätzlichen Strategien immer robuster. Für eine wirtschaftliche Nutzung besteht die Herausforderung darin, die Reihenfolge der Strategieaufrufe so zu optimieren, dass die Taktzeit im Störungsfall durch die Behandlung nur wenig ansteigt. Eine Vorgabe der optimalen Abfolge ist für den Bediener zu komplex.

Zur Optimierung der Strategiereihenfolge im Störungsfall wird ein Partially Observable Markov Decision Process (POMDP) eingesetzt, um den ursächlichen Fehler zu schätzen und die Ablaufsequenz zu planen. Die erforderlichen statistischen Wahrscheinlichkeiten, wie die Erfolgschancen der Strategien, werden aus vorhergehenden Störungen jeweils als Erfahrungswissen gelernt. Das dem POMDP zugrunde liegende Zustandsmodell wurde so entworfen, dass die bei einer Behandlung erfassten Daten den jeweiligen Strategien als Erfahrungswissen zugeordnet werden können.

Eine Herausforderung bei diesem Ansatz besteht in dem hohen Rechenaufwand zur Ermittlung der optimalen Policy. Da sich das Erfahrungswissen durch den kontinuierlichen Informationszuwachs verändert, muss die Strategiereihenfolge zur Laufzeit häufig neu berechnet werden. Indem der Bediener möglichst viele Strategien zuweist, erhöht sich die Menge an Aktionen, die bei der Berechnung der optimalen Policy zu kombinieren sind. Zum anderen hat sich gezeigt, dass die Erfolgswahrscheinlichkeiten der Lokalisierungvarianten voneinander abhängen. Die Strategiereihenfolge wird bei der Value Iteration jedoch iterativ und antichronologisch zur Ausführung bestimmt, wodurch die im Störungsfall zeitlich vorher aufzurufenden Behandlungsmethoden und damit die zugehörige Abhängigkeit bei der Berechnung noch nicht feststehen. Um die Vorteile der bedingten Abhängigkeiten dennoch zu nutzen, sind alle Strategien mit allen möglichen Vorbedingungen untereinander zu kombinieren, wodurch der Rechenaufwand weiter steigt. Zusätzlich können sich die Erfolgswahrscheinlichkeiten verändern, wenn durch eine Anwesenheitsprüfung, wie z. B. durch blindes Zugreifen, die Objektlage verändert wird.

Zur Verringerung des Rechenaufwands werden die Erfolgschancen der Lokalisierungsstrategien in Wahrscheinlichkeitsbäumen gespeichert, in denen jeweils eine Vorauswahl der vielversprechendsten Lokalisierungsabfolgen bestimmt wird. Durch die so reduzierte Aktionsmenge sinkt die Anzahl der zu kombinierenden Aktionen innerhalb der Value Iteration deutlich. Gleichzeitig sind die zur Auswahl der bedingten Wahrscheinlichkeiten relevanten Vorgängerstrategien aus den chronologisch aufgebauten Sequenzen bekannt. Für die Umsetzung wurde der zur Berechnung der optimalen Policy verwendete Algorithmus um entsprechende Regeln zur Berücksichtigung der Randbedingungen erweitert.

Eine weitere Herausforderung besteht in den häufigen Umstellungen der Produktion innerhalb der Kleinserienmontage, wodurch zu Beginn der Ausführung noch kein Erfahrungswissen vorliegt. Um dennoch eine Strategiereihenfolge berechnen zu können, wird auf globales, über alle Applikationen gemitteltes Erfahrungswissen zurückgegriffen. Zusätzlich wurden Methoden integriert, mit denen der Informationszuwachs beschleunigt und durch statistische Effekte auftretende Fehler korrigiert werden können.

Die intuitive und schnelle Bedienung der MMS wurde durch Tests mit acht Probanden nachgewiesen, die entsprechend der Zielgruppe keine Erfahrungen in der Programmierung von Robotern besaßen und von denen vier als Einsteller in Werken der Robert Bosch GmbH arbeiten. Als Aufgabe war jeweils einer von zwei realen Pick-and-Place-Applikationen aus der Fertigung bei Bosch vollständig, inklusive der notwendigen Objektlageerkennungen zu programmieren. Als Versuchsplattform diente der bei Bosch entwickelte Automatische Produktionsassistent (APAS), ein flexibles Robotersystem mit universellem Greifer sowie unterschiedlichen Kamerasensoren und Lageerkennungsalgorithmen.

Die unerfahrenen Probanden konnten die jeweilige Aufgabe nach einer kurzen Einführung zur MMS alle eigenständig und vollständig programmieren. Die Parametrierung und die erreichte Genauigkeit ermöglichten eine Ausführung der Applikationen im Dauerbetrieb. Das Bedienkonzept, die Steuerung der Roboterbewegungen sowie das Einrichten der Objektlageerkennung wurden intuitiv verstanden und über einen Fragebogen von allen Teilnehmern sehr positiv bewertet. Das Robotersystem wurde von einigen der Probanden im Nachhinein bereits eigenständig für Applikationen in der Fertigung programmiert. Die innerhalb des Tests von den Teilnehmern zur Programmierung der Applikationen benötigte Zeit lag im Durchschnitt mit knapp über 1,5 bzw. 2,25 Stunden deutlich unter den Anforderungen. Nach Schätzung eines Roboterexperten von Bosch wurde die Dauer gegenüber einer herkömmlichen textuellen Programmierung durch einen Spezialisten, der das Roboter-Frameworks nutzt, um über 90% reduziert. Die intuitive Bedienbarkeit der MMS sowie die erzielbare Reduktion der Programmierdauer führten dazu, dass das in dieser Arbeit entwickelte Konzept als Teil des Robotersystems APAS bereits Bosch-intern in der Fertigung eingesetzt wird.

Die Optimierung der Dauerlaufrobustheit durch eine Störungsbehandlung konnte anhand von sechs typischen, störungsanfälligen Applikationen gezeigt werden. Durch den strategiebasierten Ansatz

konnte eine geeignete Behandlung für die auftretenden Fehler in allen Montageabläufen flexibel und innerhalb weniger Minuten eingerichtet werden. Diese Maßnahmen reduzierten die Ausfällen um 92,5% bis 100%. Diese Autonomiesteigerung im Dauerbetrieb ermöglicht bei einigen der Applikationen erst eine wirtschaftliche Automatisierung mit flexiblen Robotersystemen.

Der Einsatz einer erfahrungsbasierten Fehlerschätzung und Behandlungsplanung zur Optimierung der Taktzeit im Störungsfall wurde mittels Dauerlaufversuchen evaluiert. Anhand von zwei typischen Applikationen wurden verschiedene Strategiekombinationen und Parametrierungen im Dauerlauf mittels vergleichbarer, in realen Ausführungen erfassten Daten simuliert und gegenübergestellt. Die erfahrungsbasierte Fehlerschätzung und Ausführungsplanung mit einem POMDP reduzierte die Behandlungsdauer bei lageabhängigen Lokalisierungsstörungen je nach Szenario um 20% bis 60% im Vergleich zu einer zufälligen Strategieauswahl. Gleichzeitig wird die Berücksichtigung verschiedener Fehlertypen ermöglicht. Die Behandlungsdauer bei fehlenden oder nicht lokalisierbaren Objekten konnte durch den Einsatz des gelernten Erfahrungswissens abhängig von den Randbedingungen um 25% bis 70% gegenüber der zufälligen Strategieauswahl verringert werden. Die Störungsbehandlung erlaubte in den Dauerläufen eine Reduktion der störungsbedingten Ausfallzeiten der Maschine von über 92%. Grundlage für diese Bewertung ist eine durchschnittliche Dauer von ca. 4 min, die im Werk von der Anzeige einer Meldung bis zur manuellen Reaktion durch einen Bedienereingriff gemessen wurde. Die Berechnung der optimalen Strategieabfolgeerfolgte unter 1,26 Sekunden und ermöglicht somit ein dynamisches Update im laufenden Betrieb.

In einer Voruntersuchung wurden zwei Methoden zur Initialisierung und Verbesserung des Erfahrungswissens nach einer Produktionsumstellung analysiert. Eine ergebnisunabhängige Ausführung aller Strategien mit dem Ziel, möglichst viel Erfahrungswissen zu sammeln, um nachfolgend genauer planen zu können, ist demnach nicht sinnvoll. Die Zeit zur Ausführung aller Strategien wird durch das verbesserte Erfahrungswissen zumeist nicht kompensiert. Der Aufruf einer unabhängigen Zusatzstrategie kann hingegen durch statistische Effekte entstehende Abweichungen des Erfahrungswissen von der Wirklichkeit reduzieren. Um eine Verbesserung zu erreichen, ist die Zusatzstrategie nur auszuführen solange eine Optimierung erfolgt.

8.2. Ausblick

Die Ergebnisse der Evaluierung haben gezeigt, dass die in dieser Arbeit vorgestellte MMS eine schnelle Programmierung ohne Expertenwissen ermöglicht. Gleichzeitig kann der Bediener die Robustheit im autonomen Dauerbetrieb über das vorgestellte Konzept zur Störungsbehandlung deutlich erhöhen. Die Evaluierung hat aber gezeigt, dass bei beiden Aspekten noch Optimierungspotential vorhanden ist.

Mögliche Weiterentwicklungen bezüglich der intuitiven Programmierung flexibler Robotersysteme

Bei den Probandentests war erkennbar, dass beim Positionieren des Roboters noch Potential für eine Zeitreduktion und eine Komfortverbesserung besteht:

- *Verwendung eines tragbaren Eingabegeräts*

  Bei der Versuchsplattform erfolgte die Bedienung der MMS über einen Touchscreen, der fest, außerhalb des Arbeitsbereichs montiert ist. Bei der Positionierung mittels des greiferbasierten Steuerelements kommt es für die Bauteilhandhabung zumeist auf eine sehr hohe Genauigkeit an, die nur aus geringer Distanz überprüft werden kann. Die Bediener waren daher gezwungen ihren Standort häufig zwischen dem Greifer und der Bedienoberfläche zu wechseln. Die erforderliche Neuorientierung auf dem Eingabegerät reduziert den Komfort und die Geschwindigkeit. Die Verwendung eines tragbaren Touchscreens, der auch in der Nähe des Greifers bedienbar ist, wird angeraten.

- *Grobpositionierung des Roboterarms mittels Tablet-PC mit Lagesensorik*

  Die Testergebnisse zeigen, dass die Grobpositionierung des Roboterarms speziell bei starken Änderungen der Armkonfiguration bzw. der TCP-Orientierung länger dauerte und im Vergleich mit den anderen Steuerungselementen als komplizierter empfunden wurde. Entsprechend könnte eine Weiterentwicklung zur Positionierung des Roboterarms den Bedienkomfort steigern und die Programmierdauer weiter reduzieren. Da die Bedienelemente zur Positionierung von Kamera und Greifer sehr effizient und effektiv eingesetzt wurden sind diese beizubehalten. Bei neuen Methoden ist daher neben der generellen Eignung vor allem die Integrierbarkeit in das Gesamtkonzept zu prüfen. In Kombination mit dem Einsatz eines tragbaren Touchscreens wird vorgeschlagen, die Verwendung eines Tablet-PC zur Grobpositionierung des Roboters mittels integrierter Orientierungs- und Beschleunigungssensoren zu prüfen. Der TCP könnte sich beispielsweise parallel zur Oberfläche des Bildschirms ausrichten und den Bewegungen des Bediengeräts folgen.

- *Erweiterung auf Zweiarmigkeit*

  Bei der Analyse des typischen Applikationsspektrums zeigte sich, dass die Taktzeitvorgaben teilweise mit einer einhändigen Manipulation nicht erreichbar sind. Beim Bearbeiten von Bauteilen mit Prozessstationen kommt es auf ein schnelles Be- und Entladen an, da die Anlage in diesem Zeitraum nicht produktiv ist. Der Mensch nutzt den laufenden Prozess, um bereits ein neues Werkstück zu greifen. Der Teilewechsel erfolgt so nach Beendigung der Ausführung sehr schnell. Ein einarmiger Roboter muss zunächst das Fertigteil entnehmen und kann den Teilewechsel nicht während des Prozesses vorbereiten. Entsprechend könnte eine Weiterentwicklung der MMS für die Programmierung zweiarmiger Robotersysteme weiteres Automatisierungspotenzial in der KSM erschließen.

Mögliche Weiterentwicklungen bezüglich der erfahrungsbasierten Störungsbehandlung

Die Ergebnisse der Evaluierung haben gezeigt, dass das vorgestellte Konzept zur Störungsbehandlung die Robustheit im autonomen Dauerbetrieb deutlich erhöhen kann. Die erfahrungsbasierte Fehlerschätzung und Ausführungsplanung erlaubt dabei eine Behandlung unterschiedliche Fehlerarten bei optimierter Zeitdauern. Dennoch sind für eine industrielle Nutzung noch Weiterentwicklungen möglich:

- *Optimierung des Abschaltzeitpunkts für die unabhängige Zusatzstrategie*

  Die Ausführung einer unabhängigen Zusatzstrategie behebt mögliche, durch zufällige Effekte auftretende Abweichungen innerhalb der statistischen Daten. Durch die verbesserte Wissensbasis kann die Behandlungsdauer im Störungsfall reduziert werden. Da der Aufwand der zusätzlichen Strategie während der Optimierung nicht kompensiert wird, sollte diese nur zu Beginn einer Applikation ausgeführt werden, solange eine relevante Verbesserung eintritt. Da der Abschaltzeitpunkt im vorgestellten Ansatz exemplarisch und nur relativ grob ermittelt wird, kann eine Optimierung des Algorithmus den Zeitvorteil weiter erhöhen. Mögliche Eingangsgrößen sind neben der Steigung im Verlauf der mittleren Behandlungsdauer ggf. auch die globalen und lokalen Erfolgswahrscheinlichkeiten der Strategien sowie die Anzahl ihrer bisherigen Aufrufe.

- *Warnungen und Hinweise bei starken Veränderungen der Störungs- und Fehlerraten*

  Die Evaluierung hat gezeigt, dass der vorgestellte Ansatz Ausfälle im Betrieb autonom vermeiden kann, indem auftretende Störungen in zulässige Systemzustände überführt werden. Auf diese Weise erhält der Bediener jedoch keine Rückmeldung über die aktuellen Störungs- und Fehlerraten. Daher sollten automatische Überprüfungen der statistischen Daten in das Konzept integriert werden, um den Bediener ggf. über relevante plötzliche und schleichende Erhöhungen der Störungshäufigkeit zu informieren. In diesem Zusammenhang könnte auch geprüft werden, ob aus dem statistischen Erfahrungswissen Aussagen über eine geeignete Anpassung der normalen Ausführung ableitbar sind.

- *Detektion abweichender Bauteilgriffe durch Zusatzsensorik*

  Bei dem vorgestellten Ansatz werden Störungen bei der Bauteilmanipulation über die Greiferöffnungsweite detektiert. Abhängig vom Werkstück können Lageabweichungen auftreten, die keine Veränderung der Fingerabstände verursachen und nicht entdeckt werden. Ist eine Detektion dieser Störungen zwingend erforderlich, sind zusätzliche Sensoren nötig. Um die Taktzeit in störungsfreien Zyklen nicht zu erhöhen, sollte die Bauteillage während der Manipulation geprüft werden. Entsprechende Sensoren müssten jedoch in die Hardware integriert werden und wären nicht mehr systemunabhängig und ggf. auch sehr kostenintensiv.

Die Evaluierung der in dieser Arbeit entwickelten Konzepte zeigt, dass die Programmierung von flexiblen Robotersystemen für typische industrielle Pick-and-Place-Abläufe über eine intuitive MMS

auch ohne Expertenwissen innerhalb von 1,5 bis 3 Stunden ermöglicht wird. Der Einsatz einer strategiebasierten Störungsbehandlung erlaubt dem Bediener dabei die Robustheit und Autonomie des Dauerbetriebs zu optimieren. Diese Ansätze führen zusammen mit der erfahrungsbasierten Fehlerschätzung und Behandlungsplanung zu einer starken Steigerung der Wirtschaftlichkeit von flexiblen Robotersystemen und können so zu einer stärkeren Automatisierung der Kleinserienmontage beitragen.

A. Abkürzungen und Formelzeichen

Abkürzungen

A	Auftretenswahrscheinlichkeit eines Fehlers bei der FMEA
APAS	Automatischer Produktionsassistent
B	Bedeutung eines Ausfalls bei der FMEA
CAD	Rechnerunterstütztes Konstruieren (engl.: computer-aided design)
DoF	Freiheitsgrade (engl.: degree of freedom)
E	Entdeckungswahrscheinlichkeit eines Fehlers bei der FMEA
FMEA	Fehlermöglichkeits- und Einflussanalyse
GUI	Grafische Benutzeroberfläche (engl.: graphical user interface)
KMS	Kraft-Momenten-Sensor
KMU	Kleine und mittelständische Unternehmen
KSM	Kleinserienmontage
MDP	Markov Decision Process
MMS	Mensch-Maschine-Schnittstelle
MTBF	Mean time between failures
PdV	Programmieren durch Vormachen
POMDP	Partially Observable Markov Decision Process
ROI	Bereich des Interesses (engl.: region of interest)
RPZ	Risikoprioritätszahl bei der FMEA
TCP	Tool-Center-Point
WPF	Windows Presentation Foundation

Formelzeichen

$\mathcal{A}$	Menge der auswählbaren Aktionen des POMDP
a_i	Aktion mit Index i
$a(p)$	Startaktion der Policy p
b_i	Belief-Zustand mit Index i
Base	Index des Basiskoordinatensystems
Cam	Index des Kamerakoordinatensystems
Grip	Index des Greiferkoordinatensystems
O	Beobachtungsfunktion für POMDP
$p(s_i)$	Wahrscheinlichkeit für Übergang zum Zustand s_i
${}^{Base}p_i$	Raumpunkt i in Basiskoordinaten
R	Reward-Funktion für POMDP
$\mathcal{S}$	Menge der möglichen Systemzustände des POMDP

s_i	Systemzustand mit Index i
s, c, r, n	Indexe der Aktionen und Strategien: s - Lokalisieren, c - Anwesenheitsprüfung, r - Bauteil entfernen, n - Neues Bauteil beschaffen
${}^{B}_{A}T$	Homogene Transformationsmatrix zur Überführung aus A nach B
T	Transitionsfunktion für POMDP
TCP	Index des TCP-Koordinatensystems
V_p	Gesamtnutzen der Ausführung von Policy p
V_π	Gesamtnutzen der optimalen Policy
$\mathcal{Z}$	Menge der möglichen Beobachtungen im POMDP
z_i	Beobachtung mit Index i

B. Homogene Transformationsmatrizen

Die homogene Transformationsmatrix ${}^{Base}_{TCP}T$ bildet die Lage des TCP-Koordinatensystems in Basiskoordinaten des Roboters ab. Mittels Formel B.1 kann ein in TCP-Koordinaten bekannter Punkt ${}^{TCP}p$ in Basiskoordinaten umgerechnet werden, wobei sich $\tilde{p}$ nach der Formel B.2 aus den Koordinaten von p ergibt.

$$ {}^{Base}\tilde{p} = {}^{Base}_{TCP}T \cdot {}^{TCP}\tilde{p} \tag{B.1} $$

$$ \tilde{p} = \begin{pmatrix} p_x & p_y & p_z & 1 \end{pmatrix}^{\mathsf{T}} \tag{B.2} $$

Die Umkehrung erfolgt über die inverse Transformationsmatrix mit Formel B.3.

$$ {}^{TCP}\tilde{p} = {}^{Base}_{TCP}T^{-1} \cdot {}^{Base}\tilde{p} \tag{B.3} $$

Die homogene Transformationsmatrix ${}^{Base}_{TCP}T$ wird nach Formel B.4 aus der Verschiebung ${}^{Base}p_{TCP}$ sowie der Verdrehung ${}^{Base}_{TCP}R$ der Koordinatensysteme *TCP* und *Base* zueinander erstellt.

$$ {}^{Base}_{TCP}T = \begin{pmatrix} & {}^{Base}_{TCP}R & & {}^{Base}p_{TCP} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{B.4} $$

Der Translationsvektor ${}^{Base}p_{TCP}$ zum TCP ergibt sich aus den kartesischen Roboterkoordinaten ${}^{Base}x_{TCP}$, ${}^{Base}y_{TCP}$, ${}^{Base}z_{TCP}$, ${}^{Base}\alpha_{TCP}$, ${}^{Base}\beta_{TCP}$ und ${}^{Base}\gamma_{TCP}$ nach Formel B.5.

$$ {}^{Base}p_{TCP} = \begin{pmatrix} {}^{Base}x_{TCP} & {}^{Base}y_{TCP} & {}^{Base}z_{TCP} \end{pmatrix}^{\mathsf{T}} \tag{B.5} $$

Die Rotationsmatrix ${}^{Base}_{TCP}R$ wird aus einer Multiplikation von drei einzelnen Matrizen errechnet, die jeweils eine Drehung des Koordinatensystems um eine der drei Koordinatenachsen mit dem zugehörigen Winkel darstellen. Die Reihenfolge der Verkettung ist roboterspezifisch und hängt von der zur Berechnung der Orientierungswinkel verwendeten Drehsequenz ab. Für eine Euler-Drehung um konsekutive Koordinatenachsen ergibt sich ${}^{Base}_{TCP}R$ über Formel B.6

$$
{}^{Base}_{TCP}R(\alpha,\beta,\gamma) = \begin{pmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{pmatrix}. \tag{B.6}
$$

Für weitere Informationen siehe [Paul 1981].

C. Unterlagen zur Evaluierung der intuiviten MMS

C.1. Bewertungsbogen zur MMS

Fragebogen zum Programmieren von Applikationen mit der Bedienerschnittstelle des Automatischen Produktionsassistenten APAS

Im folgenden sind verschiedene Aussagen zu der Bedienerschnittstelle des APAS aufgelistet. Beurteilen Sie bitte basierend auf Ihren Erfahrungen mit dem APAS, wie stark diese Aussagen zutreffen. Hierfür steht Ihnen eine 5-stufige Skala zur Verfügung, wobei die Abstände jeweils gleich groß sind. Uns interessiert Ihre persönliche Meinung, d.h. es gibt keine "richtigen" oder "falschen" Antworten.

Frage:	Bewertungsskala				
	trifft zu	trifft eher zu	teils teils	trifft wenig zu	trifft nicht zu
Bedienerführung					
1. Die Unterteilung der Applikation in einen Arbeitsplan mit einzelnen Arbeitsschritten finde ich verständlich.	☐	☐	☐	☐	☐
2. Ich konnte mich innerhalb der Arbeitsplandarstellung gut orientieren und wusste welchen Teil der Applikation die Arbeitsschritte darstellen.	☐	☐	☐	☐	☐
3. Das Einlernen der Arbeitsschritte in Form von Wizards mit mehreren Schritten bzw. Seiten finde ich verständlich.	☐	☐	☐	☐	☐
4. Ich konnte die einzelnen Einlernschritte im Zusammenhang des Arbeitsschrittes einordnen.	☐	☐	☐	☐	☐
5. Ich wusste stets, wo Eingaben erforderlich sind und wie der Programmiervorgang fortzuführen war.	☐	☐	☐	☐	☐
Objekterkennung einlernen					
6. Die Fragen, wie das Bauteil in der Applikation zu- bzw. abgeführt wird, waren mit der vorhandenen Unterstützung gut zu beantworten.	☐	☐	☐	☐	☐
7. Ich kenne die Unterschiede und Vorzüge der Objekterkennungsverfahren und hätte die für den Arbeitsschritt benötigten Objektmodelle lieber direkt ausgewählt.	☐	☐	☐	☐	☐
8. Beim Einlernen der Objektmodelle fand ich es verständlich, was in den einzelnen Einlernschritten zu tun ist.	☐	☐	☐	☐	☐
9. Die Ergebnisbilder, in denen die Auswirkung der einzelnen Parameter dargestellt wurde, fand ich verständlich.	☐	☐	☐	☐	☐
10. Das Einlernen der Objektmodelle ist mir schwer gefallen.	☐	☐	☐	☐	☐
Bewegungen einlernen					
11. Ich konnte den Roboter über die „roboterbasierte“ Bewegungssteuerung gut über längere Strecken an die gewünschte Position verfahren.	☐	☐	☐	☐	☐
12. Ich konnte die Kamera über die „kamerabasierte“ Bewegungssteuerung gut zum gewünschten Sichtbereich verfahren.	☐	☐	☐	☐	☐
13. Ich konnte den Greifer über die „greiferbasierte“ Bewegungssteuerung gut auf das Bauteil oder die Ablageposition ausrichten.	☐	☐	☐	☐	☐
14. Das Verfahren des Roboters war kompliziert und erforderte viel Überlegung.	☐	☐	☐	☐	☐
15. Das Einlernen ganzer Pfade über Einzelpunkte fand ich einfach.	☐	☐	☐	☐	☐

C.2. Fragebogen zur Sozialstatistik

Geben Sie bitte weitere Angaben zu sich und Ihren bisherigen Erfahrungen an.

Alter: ☐ unter 25 ☐ 25 bis 35 ☐ 35 bis 45 ☐ 45 bis 55 ☐ über 55

Beschreiben Sie bitte kurz Ihre Aufgabe bei Bosch.

Welche Erfahrungen haben Sie im Umgang mit Bediengeräten, die per Touchscreen gesteuert werden (z.B. Handy, tragbarer PC, Ticketautomat, etc.)?

Welche Software nutzen Sie regelmäßig (Bitte mit Angabe ob beruflich und/oder privat)?

Welche Erfahrungen haben Sie im Umgang mit Robotern oder ähnlichen Handhabungsmaschinen (z.B. Bedienung, Programmierung, Planung, etc.)?

Welche Erfahrungen haben Sie im Bereich der Bildverarbeitung (Programmierung oder Einstellung von Erkennungsverfahren, Bearbeitung von Fotos, etc.)?

Besitzen Sie Kenntnisse in Programmiersprachen? Wenn ja, bitte Sprache (z.B. C++, Java, S7, etc.) und kurze Einschätzung der Kenntnisse (Grundkenntnisse, Fortgeschrittene Kenntnisse, Experte) angeben.

C.3. Interviewleitfaden

Nach der Programmierung der Montageapplikation wurden die Probanden in einem halbstrukturierten Interview zu ihren Erfahrungen mit der MMS befragt. Die Teilnehmer hatten so die Möglichkeit das entwickelte Konzept frei von Vorgaben bewerten zu können. Um die Ergebnisse besser vergleichen zu können, wurden die Fragen streng nach dem folgenden Leitfaden gestellt:

Leitfaden teilstrukturiertes Interview:

1. Gab es etwas an der MMS, das Ihnen besonders gut gefallen hat?
2. Gab es etwas an der MMS, das Ihnen gar nicht oder nicht so gut gefallen hat?
3. Was könnte man Ihrer Meinung nach noch verbessern? Haben Sie Vorschläge?
4. Bewerten Sie Ihren Gesamteindruck von der MMS zur Programmierung des APAS hinsichtlich der Bedienerfreundlichkeit auf einer Skala von 1 - „schlecht“ bis 10 - „sehr gut“. Sind Sie gut zurechtgekommen oder wussten Sie mehrfach nicht, wie sie weiter vorgehen sollen?
5. Trauen Sie sich zu, dieses System nach einer eintägigen Schulung selbst zu bedienen und zu programmieren?

D. Abbildungsverzeichnis

E. Tabellenverzeichnis

F. Literaturverzeichnis

[ABB 2013a] ABB Inc. (2013a). Machine tending software - easy to use tools for flexible & troublefree robotic machine tending.
URL: http://www05.abb.com. Letzter Zugriff: 08/2013.

[ABB 2013b] ABB Inc. (2013b). Robotstudio - Übersicht.
URL: http://www.abb.de/product/seitp327/b357cd0ca80f7bb0c12570c9003fffa9.aspx. Letzter Zugriff: 08/2013.

[Aleotti u. a. 2004] Aleotti, J., Caselli, S., und Reggiani, M. (2004). Leveraging on a virtual environment for robot programming by demonstration. *Robotics and Autonomous Systems*, 47(2):153–161.

[Andersson u. a. 2010] Andersson, K., Lennartson, B., und Fabian, M. (2010). Restarting manufacturing systems; restart states and restartability. *IEEE Transactions on Automation Science and Engineering*, 7(3):486–499.

[Armbruster u. a. 2006] Armbruster, H., Kirner, E., und Kinkel, S. (2006). *Neue Kundengruppen für Industrieroboter*. Mitteilungen aus der Produktionsinnovationserhebung 38. Fraunhofer ISI, Karlsruhe.

[Bannat u. a. 2009] Bannat, A., Gast, J., Rehrl, T., Rösel, W., Rigoll, G., und Wallhoff, F. (2009). A multimodal human-robot-interaction scenario: Working together with an industrial robot. In *Human-Computer Interaction. Novel Interaction Methods and Techniques*, pages 303–311. Springer.

[Baydar und Saitou 2001] Baydar, C. M. und Saitou, K. (2001). Off-line error prediction, diagnosis and recovery using virtual assembly systems. In *Proceedings of the IEEE International Conference on Robotics and Automation, 2001. (2001 ICRA)*, volume 1, pages 818–823. IEEE.

[Becher u. a. 2006] Becher, R., Steinhaus, P., Zollner, R., und Dillmann, R. (2006). Design and implementation of an interactive object modelling system. *VDI BERICHTE*, 1956:27.

[Biggs und Macdonald 2003] Biggs, G. und Macdonald, B. (2003). A survey of robot programming systems. In *Proceedings of the Australian Conference on Robotics and Automation, CSIRO*, page 27.

[Billard u. a. 2007] Billard, A., Calinon, S., Dillmann, R., und Schaal, S. (2007). Handbook of robotics chapter 59: Robot programming by demonstration.

[Bischoff u. a. 2002] Bischoff, R., Kazi, A., und Seyfarth, M. (2002). The morpha style guide for icon-based programming. In *Proceedings of the 11th IEEE International Workshop on Robot and Human Interactive Communication, 2002*, pages 482–487.

[Bouguerra 2008] Bouguerra, A. (2008). *Robust Execution of Robot Task-Plans: A Knowledge-based Approach*. Dissertation, Örebro University.

[Brecher u. a. 2009] Brecher, C., Göbel, M., Herfs, W., und Pohlmann, G. (2009). Roboterprogrammierung durch demonstration - ein ganzheitliches verfahren zur intuitiven erzeugung und optimierung von roboterprogrammen. pages 655–660.

[Brecher u. a. 2010] Brecher, C., Romann, J., Schlette, C., Herfs, W., Ruf, H., und Göbel, M. (2010). Intuitive roboterprogrammierung in der automatisierten montage: Ein hybrides verfahren zur programmierung durch direkte interaktion. In *wt Werkstattstechnik online, Jahrgang 100 (2010), Heft 9*, pages 681–686.

[Brecher u. a. 2006] Brecher, C., Schröter, B., Kürzel, A., Herchel, M., und Matthias, B. (2006). Portable robot systems for machine tending tasks. In *Proceedings of the 37th international symposium on robotics*, pages 15–17. München.

[Broxvall u. a. 2004] Broxvall, M., Coradeschi, S., Karlsson, L., und Saffiotti, A. (2004). Have another look: On failures and recovery planning in perceptual anchoring. In *Proceedings of the 4th International Cognitive Robotics Workshop (CogRob-2004)*.

[Calinon und Billard 2007] Calinon, S. und Billard, A. (2007). Active teaching in robot programming by demonstration. In *The 16th IEEE International Symposium on Robot and Human interactive Communication, 2007. (RO-MAN 2007)*, pages 702–707. IEEE.

[Calinon und Billard 2008] Calinon, S. und Billard, A. (2008). A probabilistic programming by demonstration framework handling constraints in joint space and task space. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. (IROS 2008)*, pages 367–372. IEEE.

[Cassandra 1998] Cassandra, A. R. (1998). *Exact And Approximate Algorithms for Partially Observable Markov Decision Processes*. Dissertation, Brown University.

[Chang u. a. 2007] Chang, L. Y., Pollard, N. S., Mitchell, T. M., und Xing, E. P. (2007). Feature selection for grasp recognition from optical markers. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. (IROS 2007)*, pages 2944–2950. IEEE.

[Chhatpar und Branicky 2005] Chhatpar, S. R. und Branicky, M. S. (2005). Localization for robotic assemblies using probing and particle filtering. In *Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 1379–1384. IEEE.

[DEEM Controls Inc. 2013] DEEM Controls Inc. (2013). Neurocheck - the concept. URL: http://www.deemencoders.com/VisionConcept.html. Letzter Zugriff: 08/2013.

[Deiterding 2011] Deiterding, J. (2011). *Simplified Integration of External Sensors in Industrial Robot Programs*. Dissertation, Universität Bayreuth.

[DELMIA 2013] DELMIA GmbH (2013). Delmia robotics offline programming. URL: http://www.3ds.com/products/delmia/products/robotics-programmers/robotics-offline-programming/. Letzter Zugriff: 08/2013.

[Denkena u. a. 2006] Denkena, B., Apitz, R., und Kowalski, P. (2006). Vr-unterstützung für die nc-programmierung fünfachsiger fräsprozesse. In *wt Werkstattstechnik online, Jahrgang 96 (2006), Heft 1/2*, pages 47–51.

[DGQ 2012] DGQ (2012). *FMEA - Fehlermöglichkeits- und Einflussanalyse*, volume Band 13-11. DGQ.

[Di u. a. 2009] Di, P., Huang, J., Chen, F., Sasaki, H., und Fukuda, T. (2009). Hybrid vision-force guided fault tolerant robotic assembly for electric connectors. In *International Symposium on Micro-NanoMechatronics and Human Science, 2009. MHS 2009*, pages 86–91. IEEE.

[Dillmann 2004] Dillmann, R. (2004). Teaching and learning of robot tasks via observation of human performance. *Robotics and Autonomous Systems*, 47(2):109–116.

[Dose und Dillmann 2012a] Dose, S. und Dillmann, R. (2012a). Eine intuitive mensch-maschine-schnittstelle für die automatisierte kleinserienmontage. In *Tagungsband 13. VDI Kongress Automation, Baden-Baden*, pages 271–274.

[Dose und Dillmann 2012b] Dose, S. und Dillmann, R. (2012b). Shop floor based programming of assembly assistants for industrial pick-and-place applications. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 4966–4971. IEEE.

[Dotoli u. a. 2011] Dotoli, M., Fanti, M. P., und Mangini, A. M. (2011). A fault monitor for automated manufacturing systems using a hybrid petri nets formalism. *Transactions of the Institute of Measurement and Control*, 33(1):149–167.

[Doulgeri und Matiakis 2006] Doulgeri, Z. und Matiakis, T. (2006). A web telerobotic system to teach industrial robot path planning and control. *IEEE Transactions on Education*, 49(2):263–270.

[Ehrenmann u. a. 2002] Ehrenmann, M., Rogalla, O., Zöllner, R., und Dillmann, R. (2002). Belehrung komplexer aufgaben: Programmierung durch vormachen in werkstätten und haushalten. *Robotik 2002, Ludwigsburg, Germany*.

[Estable u. a. 2002] Estable, S., Ahrns, I., Backhaus, H. G., El Zubi, O., und Münstermann, R. (2002). Intuitive teaching and surveillance for production assistants. In *Proc. of the 11th IEEE Int. Workshop on Robot and Human interactive Communication, ROMAN2002, Berlin, Germany*, pages 474–481.

[Fanuc 2013] Fanuc Robotics Deutschland GmbH (2013). Roboguide.
URL: http://www.fanucrobotics.de/. Letzter Zugriff: 08/2013.

[Haddadin u. a. 2011] Haddadin, S., Suppa, M., Fuchs, S., Bodenmüller, T., Albu-Schäffer, A., und Hirzinger, G. (2011). Towards the robotic co-worker. In *Robotics Research*, pages 261–282. Springer.

[Hahn 2007] Hahn, D. (2007). *Robotereinsatz in der werkstattorientierten Fertigung*. Dissertation, Universität Hannover.

[Haug 2010] Haug, F. (2010). *Ansichtsbasierte 6 DoF Objekterkennung mit lokalen kovarianten Regionen*. Dissertation, Ruprecht Karls Universität Heidelberg.

[Hein u. a. 2008] Hein, B., Hensel, M., und Wörn, H. (2008). Intuitive and model-based on-line programming of industrial robots: A modular on-line programming environment. In *Proceeding of the IEEE International Conference on Robotics and Automation, 2008. (ICRA 2008)*, pages 3952–3957.

[Hein und Wörn 2009] Hein, B. und Wörn, H. (2009). Intuitive and model-based on-line programming of industrial robots: New input devices. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 3064–3069. IEEE.

[Helms 2003] Helms, E. (2003). Assistierende, interaktive und sicher im industriellen umfeld agierende ortsflexible roboter. In *Tagungsband 2. Workshop für OTS-Systeme in der Robotik - Mensch und Roboter ohne trennende Schutzsysteme*, pages 87–102, Stuttgart.

[Holzäpfel 2006] Holzäpfel, M. (2006). Entwicklung eines teach-systems für industrieroboter auf der basis von orientierungssensoren. Diplomarbeit, Reutlingen University.

[Huang u. a. 2008] Huang, J., Fukuda, T., und Matsuno, T. (2008). Model-based intelligent fault detection and diagnosis for mating electric connectors in robotic wiring harness assembly systems. *IEEE/ASME Transactions on Mechatronics*, 13(1):86–94.

[IFR Statistical Department 2012] IFR Statistical Department (2012). Erfolgsgeschichte der industrieroboter setzt sich fort.

URL: http://www.worldrobotics.org/uploads/tx_zeifr/30_08_2012_PI_IFR_Industrieroboter.pdf. Letzter Zugriff: 08/2013.

[Iossifidis u. a. 2002] Iossifidis, I., Bruckhoff, C., Theis, C., Grote, C., Faubel, C., und Schoner, G. (2002). Cora: An anthropomorphic robot assistant for human environment. In *Proceedings of the 11th IEEE International Workshop on Robot and Human Interactive Communication, ROMAN2002, Berlin, Germany*, pages 392–398.

[Isermann 2006] Isermann, R. (2006). *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer.

[Jäkel 2013] Jäkel, R. (2013). *Learning of Generalized Manipulation Strategies in Service Robotics*. Dissertation, Karlsruher Institut für Technologie (KIT).

[Kaelbling u. a. 1998] Kaelbling, L. P., Littman, M., und Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134.

[Kasper u. a. 2007] Kasper, A., Becher, R., Steinhaus, P., und Dillmann, R. (2007). Developing and analyzing intuitive modes for interactive object modeling. In *Proceedings of the 9th international conference on Multimodal interfaces*, pages 74–81. ACM.

[Kasper u. a. 2012] Kasper, A., Xue, Z., und Dillmann, R. (2012). The kit object models database: An object model database for object recognition, localization and manipulation in service robotics. *The International Journal of Robotics Research*, 31(8):927–934.

[Kazi u. a. 2005] Kazi, A., Bunsendal, J., Haag, D., Baum, R., und Bischoff, R. (2005). Next generation teach pendants for industrial robots. In *Advances in Human-Robot Interaction*, pages 47–66. Springer.

[Kinkel und Spomenka 2009] Kinkel, S. und Spomenka, M. (2009). *Produktionsverlagerung und Rückverlagerung in Zeiten der Krise*. Modernisierung der Produktion, Mitteilungen aus der ISI-Erhebung 52. Fraunhofer ISI, Karlsruhe.

[König u. a. 2012] König, A., Kleinmann, K., und Weber, W. (2012). Verbesserung des einrichtprozesses von industrierobotern durch akustisches echtzeit-feedback. In *Automation 2012*, volume VDI Berichte 2171, pages 287–290. VDI Verlag GmbH.

[KUKA 2013a] Kuka Roboter GmbH (2013a). Kuka.officelite.
URL: http://www.kuka.be/main/products/kukasim/officeLite/d_officeLite.htm. Letzter Zugriff: 08/2013.

[KUKA 2013b] Kuka Roboter GmbH (2013b). Simulation (kuka.sim).
URL: http://www.kuka-robotics.com/germany/de/products/software/kuka_sim/. Letzter Zugriff: 08/2013.

[Lüdemann-Ravit 2005] Lüdemann-Ravit, B. (2005). *Ein System zur Automatisierung der Planung und Programmierung von industriellen Roboterapplikationen*. Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Universität Dortmund.

[Lee und Chuang 2009] Lee, J.-S. und Chuang, C.-C. (2009). Development of a petri net-based fault diagnostic system for industrial processes. In *35th Annual Conference of IEEE Industrial Electronics, 2009. IECON'09*, pages 4347–4352. IEEE.

[Littman 1996] Littman, M. L. (1996). *Algorithms for sequential decision making*. Dissertation, Brown University.

[Lömker 2004] Lömker, F. (2004). *Lernen von Objektbenennungen mit visuellen Prozessen*. Dissertation, Bielefeld University.

[Loborg 1994] Loborg, P. (1994). Error recovery in automation - an overview. In *AAAI Spring Symposium on Detecting and Resolving Errors in Manufacturing Systems. Stanford, USA*.

[Loures u. a. 2006] Loures, E. R., de Paula, M. A. B., und Santos, E. A. P. (2006). A control-monitoring-maintenance framework based on petri net with objects in flexible manufacturing system. In *Third International Conference on Production Research–Americas' Region*. Citeseer.

[Lozano-Perez 1983] Lozano-Perez, T. (1983). Robot programming. *Proceedings of the IEEE*, 71(7):821–841.

[Lütkebohle u. a. 2009] Lütkebohle, I., Peltason, J., Schillingmann, L., Wrede, B., Wachsmuth, S., Elbrechter, C., und Haschke, R. (2009). The curious robot-structuring interactive robot learning. In *IEEE International Conference on Robotics and Automation, 2009. ICRA'09*, pages 4156–4162. IEEE.

[Matthias u. a. 2006] Matthias, B., Prinz, S., Hoffmann, S., Ahlbehendt, N., Neumann, A., Matern, M., Brecher, C., Schröter, B., Görnemann, O., Kürzel, A., und Herchel, M. (2006). Abschlussbericht porthos - portable handhabungssysteme für den ortsflexiblen einsatz in der produktion. URL: http://www.porthos-roboter.de/de/veroeffentlichungen.html. Letzter Zugriff: 08/2013.

[Meeussen u. a. 2007] Meeussen, W., Rutgeerts, J., Gadeyne, K., Bruyninckx, H., und De Schutter, J. (2007). Contact-state segmentation using particle filters for programming by human demonstration in compliant-motion tasks. *IEEE Transactions on Robotics*, 23(2):218–231.

[Meyer u. a. 2007] Meyer, C., Hollmann, R., Parlitz, C., und Hägele, M. (2007). Programmieren durch Vormachen für Assistenzsysteme - Schweiß- und Klebebahnen intuitiv programmieren (Programming by Demonstration for Assistive Systems - Intuitive Programming of Welding and Gluing Trajectories). *it - Information Technology*, 49(4):238–.

[Möller u. a. 2005] Möller, B., Posch, S., Haasch, A., Fritsch, J., und Sagerer, G. (2005). Interactive object learning for robot companions using mosaic images. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.(IROS 2005)*, pages 2650–2655. IEEE.

[Mohs u. a. 2007] Mohs, C., Naumann, A., und Kindsmüller, M. C. (2007). Mensch-technik-interaktion: intuitiv, erwartungskonform oder vertraut? *MMI Interaktiv-User Experience: Vol. 1, No. 13*.

[mta 2013] mta - Unitechnologies SA (2013). *Vollautomatische Montagelinie*. URL: http://www.directindustry.com/prod/mta-unitechnologies-sa/assembly-lines-13765-298805.html. Letzter Zugriff: 08/2013.

[MVTec 2013] MVTec Software GmbH (2013). Halcon - the power of machine vision. URL: http://www.mvtec.com/halcon/. Letzter Zugriff: 08/2013.

[Neto u. a. 2009] Neto, P., Pires, J., und Moreira, A. (2009). High-level programming for industrial robotics: using gestures, speech and force control.

[Ng u. a. 2008] Ng, T. C., Wong, L. S., und Yang, G. (2008). Sensor fusion for intuitive robot programming. In *IEEE Conference on Robotics, Automation and Mechatronics, 2008*, pages 1038–1043.

[Okodi u. a. 2008] Okodi, S., Jiang, X., Konno, A., und Uchiyama, M. (2008). Human demonstration data for fast task teaching. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. (IROS 2008)*, pages 961–966. IEEE.

[Opli 2013] Opli (2013). Mvtec releases halcon 11.0.1. URL: http://www.opli.net/magazine/imaging/2012/mvtec_halcon_1_0_1.aspx. Letzter Zugriff: 08/2013.

[Pan u. a. 2010] Pan, Z., Polden, J., Larkin, N., Duin, S. V., und Norrish, J. (2010). Recent progress on programming methods for industrial robots. In *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*, pages 1–8.

[Pardowitz u. a. 2005] Pardowitz, M., Zollner, R., und Dillmann, R. (2005). Learning sequential constraints of tasks from user demonstrations. In *5th IEEE-RAS International Conference on Humanoid Robots, 2005*, pages 424–429. IEEE.

[Paul 1981] Paul, R. P. (1981). *Robot manipulators: mathematics, programming, and control*. Richard Paul.

[Pettersson 2005] Pettersson, O. (2005). Execution monitoring in robotics: A survey. *Robotics and Autonomous Systems*, 53(2):73–88.

[viEMA 2013] Projekt viEMA (2013). *Vernetzte, informationsbasierte Einlern- und Ausführungsstrategien für autonome Montagearbeitsabläufe*.
URL: http://www.viema.org/. Letzter Zugriff: 08/2013.

[Roßmann u. a. 2009] Roßmann, J., Ruf, H., und Schlette, C. (2009). Model-based programming 'by demonstration' - fast setup of robot systems (prodemo). In *T. Kröger and F. M. Wahl (Eds.): „Advances in Robotics Research - Theory, Implementation, Application" (German Workshop on Robotics 2009, June 9-10, TU Braunschweig)*, pages 159–168.

[Sayler 2011] Sayler, S. (2011). *Universelle Manipulationsstrategien für die industrielle Montage*. Dissertation, Karlsruher Institut für Technologie (KIT).

[Schraft u. a. 2004] Schraft, R. D., Helms, E., Hans, M., und Thiemermann, S. (2004). Man-machine-interaction and co-operation for mobile and assisting robots. In *Proceedings of Fourth International ICSC Symposium on Engineering of Intelligent Systems (EIS 2004), Madeira, Portugal*.

[Shim u. a. 2009] Shim, B., Baek, B., Kim, S., und Park, S. (2009). A robot fault-tolerance approach based on fault type. In *9th International Conference on Quality Software, 2009. QSIC'09*, pages 296–304. IEEE.

[Shon u. a. 2005] Shon, A. P., Grochow, K., und Rao, R. P. N. (2005). Robotic imitation from human motion capture using gaussian processes. In *5th IEEE-RAS International Conference on Humanoid Robots, 2005*, pages 129–134. IEEE.

[Siemens 2013] Siemens AG (2013). Tecnomatix.
URL: http://www.plm.automation.siemens.com/de_de/products/tecnomatix/index.shtml. Letzter Zugriff: 08/2013.

[Skoglund u. a. 2007] Skoglund, A., Iliev, B., Kadmiry, B., und Palm, R. (2007). Programming by demonstration of pick-and-place tasks for industrial manipulators using task primitives. In *International Symposium on Computational Intelligence in Robotics and Automation, 2007. (CIRA 2007)*, pages 368–373. IEEE.

[Soller und Henrich 2009] Soller, K. und Henrich, D. (2009). Intuitive robot programming of spatial control loops with linear movements. In Kröger, T. und Wahl, F. M., editors, *Advances in Robotics Research*, pages 147–158. Springer Berlin Heidelberg.

[Som 2010] Som, F. (2010). Intuitives bedieninterface für effiziente mensch-roboter-kooperation. In *at - Automatisierungstechnik, Vol. 58, No. 12*, pages 665–669.

[Spur und Uhlmann 2005] Spur, G. und Uhlmann, E. (2005). Industrieroboter. In Grote, K.-H. und Feldhusen, J., editors, *Dubbel*, pages T106–T112. Springer Berlin Heidelberg.

[Stopp u. a. 2002a] Stopp, A., Baldauf, T., Hantsche, R., Horstmann, S., Kristensen, S., Lohnert, F., Priem, C., und Ruscher, B. (2002a). The manufacturing assistant: safe, interactive teaching of operation sequences. In *Proceedings of the 11th IEEE International Workshop on Robot and Human Interactive Communication, ROMAN2002, Berlin, Germany*, pages 386–391.

[Stopp u. a. 2001] Stopp, A., Horstmann, S., Kristensen, S., und Lohnert, F. (2001). Towards interactive learning for manufacturing assistants. In *Proceedings of the 10th IEEE International Workshop on Robot and Human Interactive Communication, 2001*, pages 338–342. IEEE.

[Stopp u. a. 2002b] Stopp, A., Horstmann, S., Kristensen, S., und Lohnert, F. (2002b). Interactive learning of robot manufacturing assistants. In *Proc. of the 5th German Workshop on Artificial Life (GWAL02)*.

[Stäubli 2013] Stäubli International AG (2013). Stäubli robotics suite: Ihr pc-software-paket. URL: http://www.staubli.com/de/robotik/robotersoftware/staeubli-robotics-suite/. Letzter Zugriff: 08/2013.

[Takizawa u. a. 2003] Takizawa, M., Makihara, Y., Shimada, N., Miura, J., und Shirai, Y. (2003). A service robot with interactive vision-object recognition using dialog with user. In *Proc. of the 1st Int. Workshop on Language Understanding and Agents for Real World Interaction*, pages 16–23. Citeseer.

[Tatsuno u. a. 1996] Tatsuno, J., Kskubo, Y., Maisuysma, S., Eawabata, K., und Kobayashi, H. (1996). Human friendly teaching for industrial robots. In *Proceedings of the IEEE International Conference on Industrial Technology, 1996. (ICIT '96)*, pages 656–660.

[Thrun u. a. 2005] Thrun, S., Burgard, W., und Fox, D. (2005). *Probabilistic robotics*, volume 1. MIT press Cambridge.

[Wang u. a. 2008] Wang, J., Zhang, H., und Zhang, G. (2008). A force control assisted robot path generation system. In *Proc. of the 4th IEEE International Conference on Automation Science and Engineering, CASE 2008*, pages 528–533.

[Wersing u. a. 2006] Wersing, H., Kirstein, S., Götting, M., Brandl, H., Dunn, M., Mikhailova, I., Goerick, C., Steil, J., Ritter, H., und Körner, E. (2006). A biologically motivated system for unconstrained online learning of visual objects. In *Artificial Neural Networks–ICANN 2006*, pages 508–517. Springer.

[Wosch und Feiten 2002] Wosch, T. und Feiten, W. (2002). Reactive motion control for human-robot tactile interaction. In *Proceedings of the IEEE International Conference on Robotics and Automation, 2002 (ICRA '02)*, volume 4, pages 3807–3812.

[Zäh und Vogl 2006] Zäh, M. und Vogl, W. (2006). Interactive laser-projection for programming industrial robots. In *IEEE/ACM International Symposium on Mixed and Augmented Reality, 2006. ISMAR 2006*, pages 125–128.

[Zäh u. a. 2004] Zäh, M. F., Vogl, W., und Munzert, U. (2004). Beschleunigte programmierung von industrierobotern. In *wt Werkstattstechnik online, Jahrgang 94 (2004), Heft 9*, pages 438–441.