…

Inhalt

[1 Einleitung 1](#_Toc515451993)

[1.1 Ausgangssituation und Problemstellung 1](#_Toc515451994)

[1.2 Abgrenzung der Themenstellung 1](#_Toc515451995)

[1.3 Aufbau der Arbeit 1](#_Toc515451996)

[2 Grundlagen 2](#_Toc515451997)

[2.1 Robotik im Bereich teilautonomer technischer Assistenzsysteme 2](#_Toc515451998)

[2.2 Mustererkennung als Teilgebiet des maschinellen Sehens 5](#_Toc515451999)

[2.3 Visual Servoing 9](#_Toc515452000)

[3 Konzeption 11](#_Toc515452001)

[3.1 Manipulator mit Bildverarbeitungseinheit 11](#_Toc515452002)

[3.1.1 Anforderungen an das System 11](#_Toc515452003)

[3.1.2 Auswahl der Komponenten 14](#_Toc515452004)

[3.1.3 Aufbau des Systems 18](#_Toc515452005)

[3.2 Objekt- und Texterkennung 18](#_Toc515452006)

[3.2.1 Maschinelles Detektionsverfahren mit vorgegebenem Merkmalsraum 18](#_Toc515452007)

[3.2.2 Deep Learning Ansatz zur Detektion ohne Merkmalsraumvorgabe 19](#_Toc515452008)

[3.2.3 Auslesen der Tasterbeschriftung 19](#_Toc515452009)

[3.3 Objektverfolgung 19](#_Toc515452010)

[3.3.1 Echtzeit-Objektverfolgung in Videosequenzen 19](#_Toc515452011)

[3.3.2 Merkmalsabgleich und perspektivische Transformation zwischen Einzelbildern 20](#_Toc515452012)

[3.4 Entfernungsberechnung 20](#_Toc515452013)

[3.4.1 Berechnung durch Änderung des Winkels zum Ziel 20](#_Toc515452014)

[3.4.2 Berechnung durch Änderung der Bildgröße des Ziels 20](#_Toc515452015)

[3.5 Bewegungssteuerung 20](#_Toc515452016)

[3.5.1 Wahl des Koordinatensystems 20](#_Toc515452017)

[3.5.2 Positionierung des Effektors im Raum 20](#_Toc515452018)

[3.6 Integration der Teillösungen 20](#_Toc515452019)

[3.6.1 Benutzerschnittstelle 20](#_Toc515452020)

[3.6.2 Interaktion der Komponenten 20](#_Toc515452021)

[3.6.3 Ablauf des Erkennungs- und Betätigungsprozesses 21](#_Toc515452022)

[4 Prototypische Umsetzung 21](#_Toc515452023)

[4.1 Hardware 22](#_Toc515452024)

[4.1.1 Verwendete Komponenten 22](#_Toc515452025)

[4.1.2 Schnittstellen 22](#_Toc515452026)

[4.1.3 Aufbau des Prototypen und des Tastermodells 22](#_Toc515452027)

[4.2 Software 22](#_Toc515452028)

[4.2.1 Realisierung der Bildverarbeitungsfunktionen 22](#_Toc515452029)

[4.2.2 Firmware zur Bewegungssteuerung 22](#_Toc515452030)

[4.2.3 Schnittstellen 22](#_Toc515452031)

[4.2.4 Programmstruktur 22](#_Toc515452032)

[4.3 Ergebnisse der Prototypenentwicklung 22](#_Toc515452033)

[5 Fazit und Ausblick 22](#_Toc515452034)

[6 Literaturverzeichnis 24](#_Toc515452035)

[7 Abbildungsverzeichnis 28](#_Toc515452036)

[8 Anhang A: Materialliste 30](#_Toc515452037)

# Einleitung

Evtl. ohne unterkapitel

## Ausgangssituation und Problemstellung

Einsatzumgebung

z. B. ein an Rollstuhl befestigter Leichtbauroboter

## Abgrenzung der Themenstellung

…

Grenzen Stand der Technik

Ziele was wie womit

Was soll genau gemacht werden

Einsatzumgebung

Begrenzte Energieversorgung

Skizze Aufbau! (oder in 3)

Arbeitsschritte

Mensch-Maschinen-Interaktion hier nicht

Auch nicht Mobilität (autonome Stromversorgung bzw. so konzipiert, dass Stromverbrauch gering und später autonom möglich)

## Aufbau der Arbeit

# Grundlagen

In diesem Kapitel erfolgt eine kurze Einführung in die theoretischen Grundlagen der vorliegenden Arbeit, zudem wird die Thematik auf relevante Bereiche eingegrenzt. Der aktuelle Stand der Technik dieser wissenschaftlichen Themengebiete dient als Ausgangspunkt für die Konzeption im folgenden Kapitel.

Da sich die Arbeit mit der Konzeption einer teilautonomen Funktion für einen Assistenzroboter befasst, soll zunächst die technologische Ausgangsituation in diesem Bereich ermittelt werden. Anschließend erfolgt eine Einführung in die für die Umsetzung der Arbeit wichtigen Themengebiete Mustererkennung sowie Visual Servoing.

## Robotik im Bereich teilautonomer technischer Assistenzsysteme

Technische Assistenzsysteme sind allgemein Systeme, die dem Menschen bei der Bewältigung einer Aufgabe helfen bzw. eine bestimmte Tätigkeit erst ermöglichen. Eine einheitliche Definition des Begriffes fehlt bisher, stattdessen erfolgen Abgrenzungen je nach Themengebiet und Ziel. So definiert das Bundesministerium für Gesundheit in einer Studie technische Assistenzsysteme als Hilfsmittel zur Unterstützung pflegebedürftiger Menschen auf der Basis von Informations- und Kommunikationstechnologien, wozu auch mechatronische Systeme gezählt werden [1]. In dieser Arbeit sollen Systeme aus der Robotik betrachtet werden, welche hilfsbedürftige Menschen teilautonom bei der Bewältigung von Alltagsaufgaben unterstützen.

Roboter lassen sich nach Einsatzgebiet bspw. in Industrie-, Service-, Medizin‑, Erkundungs- oder Militärroboter unterteilen [2]. Serviceroboter werden zur Unterstützung des Menschen derzeit hauptsächlich im Haushalt eingesetzt, dringen aber zunehmend auch in den Bereich der Pflege vor. Nach DIN EN ISO 8373 wird auch bei gleicher Bauform und Funktionsweise zwischen Service- und Industrierobotern nach Einsatzgebiet unterschieden [3]. Im Folgenden sollen Roboter in Anlehnung an die VDI Richtlinie 2860 als frei programmierbare Bewegungsautomaten verstanden werden, die mit Hilfe eines Effektors Handhabungsaufgaben erfüllen können [4]. Bei teilautonomen Systemen sind i.d.R. Sensoren zur selbsttätigen Programmadaption vorhanden.

Ein Beispiel für einen in der Rehabilitation verwendeten Roboter ist der JACO Roboterarm der Firma Kinova Robotics. Dieser ist für eine Montage am Rollstuhl vorgesehen, um Menschen mit eingeschränkter Mobilität mehr Selbständigkeit im Alltag zu ermöglichen [5]. Gesteuert wird der JACO manuell per Joystick, Touchpad bzw. Kopf- oder Kinnsteuerung. Einfache Bewegungsabläufe können auch über eine USB-Schnittstelle programmiert werden. Es fehlt eine Sensorik für eine Adaption zur Laufzeit, daher sind teilautonome Funktionen mit dem JACO nicht zu realisieren.

Im aktuellen Forschungsprojekt SMART-Assist entwickelt das Institut für Robotik und Mechatronik des Deutschen Zentrums für Luft- und Raumfahrt (DLR) das Assistenzrobotiksystem EDAN (EMG-controlled daily assistant) [6]. Es besteht aus einem handelsüblichen elektrischen Rollstuhl, auf welchen ein DLR-Leichtbauroboterarm montiert ist. Schwerpunkte des Projekts sind eine auf Elektromyographie (EMG) basierende Steuerung, sichere Mensch-Maschine-Interaktion sowie adaptives Greifen. Der Sicherheitsaspekt wird durch den Einsatz von flexiblen, adaptiven Materialien (Soft Robotics Konzept) sowie eine Drehmomentregelung erreicht. Integrierte Teilautonomie soll Unterstützung bei Aufgaben des Alltagslebens bieten[[1]](#footnote-1). Veröffentlichungen zur Vorgehensweise oder Ergebnissen bei der Entwicklung von teilautonomen Funktionen existieren bislang nicht.

Ein ähnliches System ist der an der Universität Bremen entwickelte Assistenzroboter FRIEND (Functional robot arm with user-friendly interface for disabled people), welcher als Basis verschiedener Forschungsprojekte dient. Der Aufbau des Systems ist in Abbildung 1 zu sehen.



Abbildung 1: Assistenzsystem FRIEND der Universität Bremen [7]

Ein Schwerpunkt des ersten Forschungsprojekts ist das Ermöglichen eines teilautomatischen Greifvorgangs. Anstatt wie bspw. beim JACO jede Bewegung vorzugeben, wird hier der komplette Greifprozess mit zwei bis drei Nutzerkommandos initiiert und durchgeführt [7]. Dazu wird zunächst die Kamera am Effektor vom Nutzer in die Nähe des gewünschten Objekts manövriert, anschließend erfolgt mit Hilfe von Visual Servoing (vgl. Abschnitt 2.3) die teilautomatische Ansteuerung des Ziels. Als Einschränkung müssen die zu greifenden Objekte auf einer zum System gehörenden Ablage platziert werden (vgl. Smart Tray in Abbildung 1), welches über Sensoren Gewicht und Position des Objekts bestimmt. Im aktuellen Modellprojekt ReIntegraRob wird durch Verbesserung der Bildverarbeitung ein Arbeiten ohne Smart Tray ermöglicht. Ziel ist Reintegration einer tetraplegisch gelähmten Frau in ihren Beruf als Bibliothekarin [8]. Hierzu soll der Roboter Funktionen wie das Aufnehmen und Ablegen des Buches oder das Umblättern von Seiten autonom durchführen.

Es existieren zahlreiche weitere Industrie- und Forschungsprojekte, welche sich mit Reha- bzw. Assistenzrobotern für Menschen mit Behinderung befassen[[2]](#footnote-2). Bezüglich teilautonomer Funktionalität scheinen die hier vorgestellten Systeme EDAN und FRIEND am aktuellsten bzw. weitesten fortgeschritten. Forschungsbedarf ist allerdings weiterhin vorhanden, da bisher Unterstützung nur für wenige spezielle Aufgaben bzw. Tätigkeiten angeboten wird. Weiterhin ist die Verfügbarkeit der genannten Systeme problematisch, da sie entweder für den normalen Nutzer kaum erschwinglich[[3]](#footnote-3) sind oder ausschließlich in Forschungsprojekten zur Verfügung stehen.

## Mustererkennung als Teilgebiet des maschinellen Sehens

„Computer Vision, auch maschinelles Sehen genannt, umfasst verschiedene Methoden zur Erfassung, Verarbeitung, Analyse und Interpretation von Bildern.“ Mustererkennung in der Bildverarbeitung ist der Prozess zur Identifikation von Objekten. Teilschritte sind Bildaufnahme, Vorverarbeitung, Segmentierung, Merkmalsextraktion, [Klassifizierung](https://de.wikipedia.org/wiki/Klassifizierung) und Aussage bzw. Interpretation [9]. Heutige Standardmethoden zur Mustererkennung verfolgen i.d.R. einen statistischen Ansatz, bei dem Wahrscheinlichkeiten für die Zugehörigkeit eines Objektes zu verschiedenen Kategorien ermittelt werden. [10]

Die Bildvorverarbeitung hat zur Aufgabe, unerwünschte Signalbestandteile zu reduzieren (z. B. Rauschreduktion). Mit Hilfe von Segmentierung werden zusammenhängende bzw. zusammengehörende Regionen in Bildern identifiziert. Anhand von Merkmalen[[4]](#footnote-4) wie bspw. Kanten oder Eckpunkten wird in der Bildanalyse versucht, auf Bildinhalte zu schließen. Das Auffinden verschiedener Merkmale übernimmt die Merkmalsextraktion. Diese liefert für ein Bild einen Merkmalsvektor im Merkmalsraum. Die Klassifikation soll abschließend anhand des Merkmalsvektors die segmentierten Regionen oder Objekte in verschiedene Klassen einteilen [11].

In der visuellen Objekterkennung muss zwischen der Bildklassifikation und der Objektlokalisierung unterschieden werden. Die Bildklassifikation prüft, ob bzw. mit welcher Wahrscheinlichkeit in einem Bild mindestens ein Beispiel einer oder mehrerer Klassen existiert. Das ganze Bild wird der Klasse bzw. Kategorie mit der höchsten Wahrscheinlichkeit zugeordnet. Ziel der Objektlokalisierung (auch Objektdetektion genannt) ist das Auffinden von Objekten verschiedener Klassen inklusive der Bestimmung von Position und Ausdehnung. Für beide Aufgabenstellungen werden heutzutage häufig Verfahren des maschinellen Lernens (Machine Learning) oder des Deep Learnings eingesetzt. Während beim Machine Learning ein vorgegebener Algorithmus zur Extraktion fest vorgegebener Merkmale verwendet wird, erlernt das System beim Deep Learning nach und nach selbst, relevante Merkmale zu extrahieren [11].

Ein bekanntes Beispiel für ein Machine Learning Verfahren zur Objektdetektion ist die Methode nach Viola und Jones [12]. Als mögliche Merkmale für einen gegebenen Bildausschnitt werden hier die Differenzen von Grauwertsummen innerhalb verschiedener Rechtecktypen verwendet, sogenannte Haar-ähnliche Merkmale (vgl. Abbildung 2).



Abbildung 2: Vier Grundtypen zur Merkmalsberechnung nach Viola und Jones in verschiedener Skalierung und Position [12]

Da die Rechtecktypen in verschiedener Größe und Skalierung verwendet werden, ergibt sich eine hohe Anzahl an möglichen Merkmalen für die Klassifikation. Die Selektion der relevanten Merkmale wird anhand eines sogenannten Boosting-Verfahrens anhand von positiven und negativen Trainingsdaten erlernt. Ein weiterer Schritt zur Effizienzsteigerung ist die Verwendung einer Kaskade von Klassifikatoren, welche negative Bildausschnitte schnell verwirft [12]. Auf der Viola-Jones-Methode basieren zahlreiche weitere Verfahren, bspw. erreicht das Verfahren von Ahonen et al. durch die Verwendung von Local Binary Patterns als texturbasierte Merkmale eine schnellere Trainings- und Detektionszeit [13].

In einer für die Themenstellung interessanten Arbeit werden zunächst verschiedene Merkmale in einem ähnlichen Verfahren wie der Viola-Jones-Methode zur Detektion von unbekannten Fahrstuhltastern verwendet. In anschließenden Nachbearbeitungsschritten werden mögliche Falschdetektionen und nicht erkannte Taster weitgehend korrigiert. Diese Schritte basieren auf der Einschränkung auf solche Taster, die den ADA Richtlinien[[5]](#footnote-5) [14] entsprechen. Unter anderem muss sich dafür die Beschriftung immer - zumindest zusätzlich - links außerhalb des Tasters befinden. Auf der Bedientafel eines Fahrstuhls ergibt sich somit eine Art Gittermuster, welches für die Erkennung genutzt wird (vgl. Abbildung 3). Weiterhin wird im Systemaufbau der Arbeit ein Laser Scanner verwendet, mit welchem die zu erwartende Tastergröße (bekannt aus den ADA Richtlinien) geschätzt werden kann. Bei einem Testdatensatz mit 150 Bildern von Bedientafeln wird eine Erkennungsrate von 86,2 % erreicht, Angaben zur Ausführungsdauer fehlen allerdings [15].



Abbildung 3: Erkannte Taster (links) und zugehöriges ermitteltes Gitter (rechts) [15].

Seit dem ersten Sieg von Krizhesvky et al. beim ImageNet Wettbewerb sind Convolutional Neural Networks (CNNs) bei der Bildklassifikation ungeschlagen [16]. Bei CNNs handelt es sich um „faltende“ neuronale Netze, bei denen in jeder Faltungsschicht die Werte eines Filterkernels (Faltungsmatrix) selbständig durch Fehlerrückführung erlernt werden. Eine detaillierte Beschreibung kann bspw. in [17] gefunden werden. Auch der komplexere Bereich der Objektdetektion wird mittlerweile von verbesserten CNN-Architekturen dominiert. Ein erster Durchbruch gelang Girshick et al. mit sogenannten Region-based Convolutional Neural Networks (R-CNNs) [18]. Während vorherige Ansätze meist nach dem Sliding-Windows Prinzip[[6]](#footnote-6) vorgehen, erfolgt bei R-CNNs die Anwendung der neuronalen Netze zur Klassifikation nur auf einer wesentlich geringeren Anzahl von vorgefilterten Regionen. Dadurch kann die Trainings- und Detektionszeit erheblich verkürzt werden[[7]](#footnote-7). [19]

Nach dem ersten Erfolg des R-CNN-Ansatzes zur Objektlokalisierung versuchen Wissenschaftler die Laufzeiten oder die Performanz im Sinne von Erkennungs- und Fehlerraten zu verbessern. Die bislang erfolgreichsten Ansätze sind Faster R-CNNs, You Only Look Once (YOLO) und Single Shot Detector (SSD) mit MobileNet-Architektur[[8]](#footnote-8). Während YOLO bislang die schnellsten Ausführungszeiten vorweisen kann, erzielt SSD bei ebenfalls geringen Laufzeiten höhere Erkennungsraten [20].

Zur Laufzeit von Objekterkennungsverfahren in eingebetteten Systemen bzw. auf Low-Power-Hardware existieren nur wenige Veröffentlichungen. Eine aktuelle Arbeit vergleicht Laufzeiten und Erkennungsraten einiger der oben genannten sowie eines selbst entwickelten Deep Learning Verfahrens bei Ausführung auf einem Raspberry Pi 3 Einplatinencomputer. Bei allen getesteten Verfahren liegt die Bearbeitungszeit bei unter einem Bild pro Sekunde[[9]](#footnote-9). Höhere Erkennungsraten korrelieren mit schlechteren Laufzeiten [21]. Für echtzeitfähige Systeme zur Objektdetektion sind je nach Einsatzgebiet und Anforderungen noch weitere Verbesserungen notwendig.

## Visual Servoing

Aufgabe von Visual Servoing ist es, mit Hilfe von Bildverarbeitungsdaten die Bewegungen oder die gewünschte Position des Effektors eines Roboters zu steuern. Diese Daten können von einer am Manipulator befestigten Kamera stammen (eye-in-hand), wobei Bewegungen des Roboters auch Bewegungen der Kamera auslösen. Eine weitere Möglichkeit ist die Platzierung der Kamera außerhalb des Roboters, sodass dessen Bewegungen von einem festen Punkt aus beobachtet werden (eye-to-hand). Unterschieden wird weiterhin zwischen bildbasierter Regelung (Image Based Visual Servoing), bei welcher Bildinformationen zur direkten Steuerung der Gelenke des Roboters verwendet werden, und positionsbasierter Regelung (Position Based Visual Servoing), wo zunächst eine geometrische Interpretation der Bilddaten erfolgt und daraus ein Steuerbefehl im kartesischen Raum erzeugt wird [22]. Bei der bildbasierten Regelung wird die Regelabweichung aus der Lageabweichung eines vorher aufgenommenen Referenzbilds und der aktuellen Kameraaufnahme ermittelt. Für die positionsbasierte Regelung wird die Stellung der kinematischen Kette im Raum anhand von Bilddaten geschätzt und ein Positionsfehler als Regelabweichung verwendet [23].

Ein stabiler geschlossener Regelkreis zur Steuerung der Gelenke benötigt eine hohe Regelfrequenz. Daten aus Bildverarbeitungssystemen stehen allerdings häufig mit geringerer Frequenz (Bildrate) zur Verfügung, so dass die meisten Visual Servoing Systeme eine positionsbasierte Regelung anwenden [23].

Sowohl zur Positionsschätzung (positionsbasierte Regelung) als auch zur Bestimmung der Lageabweichung zum Referenzbild (bildbasierte Regelung) werden Merkmale (vgl. Abschnitt 2.2) aus der Bildverarbeitung verwendet. Die Transformation zwischen zwei Bildern kann anhand ausgewählter Merkmalspunkte berechnet werden. Werden die Merkmalspunkte markerbasiert ermittelt, führt dies zu einer schnellen und robusten Merkmalsextraktion. Dagegen ist die Verwendung von Merkmalen, die aus dem Zielobjekt generiert werden, zwar aufwendiger aber auch wesentlich flexibler einsetzbar. Zur Bestimmung der Merkmalspunkte liefert zunächst ein Merkmalsdetektor markante Merkmale im Bild, anschließend versieht sie ein Merkmalsdeskriptor mit zusätzlichen Informationen, wie bspw. Orientierung und Umgebung des Punkts. Brauchbare Merkmalspunkte sollten u. a. lokal, möglichst invariant gegenüber Skalierung, Translation und Rotation, wenig anfällig für Rauschen und Verwischen, gut unterscheidbar und effizient zu berechnen sein [24].

Ein bekanntes Verfahren zur Detektion und Beschreibung lokaler Merkmalen ist SIFT (Scale-Invariant Feature Transform), welches die oben genannten Kriterien weitgehend erfüllt. Eine effiziente Abwandlung von SIFT ist das SURF-Verfahren (Speeded Up Robust Features), welches ähnlich robuste und invariante Merkmale in kürzerer Berechnungszeit liefert [25].

Ein Vergleich zwischen verschiedenen Bildern desselben Objekts liefert zusammengehörige Merkmalspunkte (auch Feature Matching genannt). Sind die Bilder aus unterschiedlichen Perspektiven entstanden, kann anhand dieser Merkmalspunkte die Homographie[[10]](#footnote-10) und damit die Bewegung der Kamera (oder des Objektes) berechnet werden [24].

# Konzeption

In diesem Abschnitt wird auf der Grundlage der in Kapitel 2 beschriebenen technologischen Ausgangssituation ein Entwurf für die Funktion der teilautomatischen Tastererkennung und -betätigung erarbeitet. Da die Funktionalität an einem prototypischen Demonstrator gezeigt werden soll, sind zunächst Grundlagenentscheidungen bezüglich dessen Aufbau und verwendeter Komponenten zu treffen. Aus diesem Auswahlprozess können sich Restriktionen für die möglichen Lösungswege der Teilaufgaben ergeben.

Unsicherheiten, mögliche Fehler früh erwähnen

**Skizze Aufbau! (oder in 1)**

## Manipulator mit Bildverarbeitungseinheit

Aufgabe des Manipulators ist die physikalische Interaktion mit der Umgebung, hier also die Betätigung des gewünschten Fahrstuhltasters. Dazu muss er Steuerbefehle entgegen nehmen und eine gewünschte Position im Raum anfahren. Die Steuerbefehle stammen aus einer Bildverarbeitungseinheit, welche die anzusteuernde Position aus Kameradaten berechnet.

Da zahlreiche Möglichkeiten des Systemaufbaus bestehen, wie z. B. Anzahl und Art der Achsen, werden aus den funktionalen und nicht-funktionalen Anforderungen zunächst Kriterien für die Festlegung des Aufbaus und die Auswahl der verwendeten Komponenten abgeleitet.

### Anforderungen an das System

Die Anforderungen an das zu entwickelnde System ergeben sich aus den zu lösenden Teilaufgaben und den Bedingungen der Einsatzumgebung. Im Zielszenario soll die teilautonome Assistenzfunktion durch ein – z. B. in einen elektrischen Rollstuhl – eingebettetes System geleistet werden. Daher ist auch für den Demonstrator möglichst kompakte, energiesparende und günstige Hardware zu verwenden, welche dennoch leistungsstark genug für eine echtzeitnahe Ausführung der Programmlogik ist. Da bei dem zu entwickelnden Prototypen geringe Kosten eine hohe Priorität haben, wird die Antwortzeit in Form einer weichen Echtzeitanforderung auf zehn Sekunden festgelegt. D. h. das System muss in unter zehn Sekunden reagieren, nicht aber die vollständige Funktion ausführen.

Als Eingabe für die Bildverarbeitung in den Arbeitsschritten Objekt- und Texterkennung, Entfernungsberechnung sowie Objektverfolgung dienen Bilder im digitalen Format. Diese müssen in ausreichend hoher Auflösung, Bildqualität und Frequenz bereitgestellt werden. Die in Abschnitt 2.2 und 2.3 genannten Verfahren zur Mustererkennung nutzen Auflösungen, die alle modernen Digitalkameras und Kameramodule für Computer bieten. Da wie oben beschrieben Low-Power-Hardware für die Ausführung der Bildverarbeitung verwendet werden soll, wird die benötigte Eingangsbildrate durch die Rechenzeit der Objekt- und Texterkennung auf ein Bild pro Sekunde beschränkt (vgl. Kapitel 2.2).

Neben der Bildverarbeitung läuft die Programmlogik (die Gesamtablaufsteuerung) und die Verarbeitung von Benutzereingaben auf derselben Hardware (im Folgenden Hauptrechner genannt). Für letztere muss eine entsprechende Schnittstelle angeboten werden. Optional kann auch die Steuerung des Manipulators vom Hauptrechner übernommen werden. Soll aber bspw. Bildverarbeitung und optische Regelung parallel laufen, reicht die Leistungsfähigkeit unter Umständen nicht aus. Wird für die Aktorsteuerung separate Hardware verwendet, muss eine Schnittstelle zur Kommunikation mit dem Hauptrechner berücksichtigt werden.

Aufgabe des Manipulators ist die Positionierung des Effektors innerhalb des Arbeitsbereichs im dreidimensionalen Raum, welche durch einen externen Steuerbefehl angestoßen wird. Da für die Entwicklung des Demonstrators die Grundfunktionalität wesentlich ist, spielen Reichweite und Arbeitsbereich nur eine untergeordnete Rolle. Zur Umsetzung des Positionierbefehls nimmt die Steuerung des Manipulators Koordinaten[[11]](#footnote-11) entgegen und berechnet daraus Steuerbefehle für die einzelnen Motoren der Achsen. Diese müssen ausreichend Drehmoment für die Bewegung der Glieder sowie für die Tasterbetätigung besitzen, wobei letzteres kaum ins Gewicht fällt.

Da der Effektor keine Arbeitsvorgänge am Zielort ausführt und keine Werkstücke transportiert, sind weder Werkzeug noch Greifer erforderlich. Zur Begrenzung der ausgeübten Kraft ist zu überprüfen, ob bzw. wie viel Druck der Effektor auf einen Gegenstand ausübt. Es wird angenommen, dass nur solche Taster betätigt werden, die in einem kegelförmigen Bereich vor dem Manipulator liegen. Dieser Bereich wird durch das Sichtfeld der verwendeten Kamera bestimmt. Daher muss die Orientierung des Effektors im Raum nicht beliebig sein, lediglich sollten die Winkel zur Horizontalen und Vertikalen bei Betätigung des Tasters nicht zu groß werden[[12]](#footnote-12).

Weiterhin wird angenommen, dass sich keine Hindernisse im Arbeitsraum bzw. zwischen der Ausgangsposition des Effektors und dem anzusteuernden Taster befinden. Dadurch kann eine zeit- und ressourcenintensive Bahnplanung vermieden werden.

Aus den genannten Anforderungen kann abgeleitet werden, dass der Manipulator mindestens drei Achsen bzw. Freiheitsgrade benötigt, um die Positionier- bzw. Betätigungsaufgabe im Arbeitsraum zu erfüllen. Mehr Achsen können z. B. zu einer besseren Orientierung des Effektors im Raum oder zu einem größeren Arbeitsbereich beitragen, sind aber nicht zwingend erforderlich. Da in der Zieleinsatzumgebung wenig Platz zur Verfügung steht (bspw. ein an einem Rollstuhl befestigter Leichtbauroboterarm), ist maximal eine Translationsachse zu verwenden.

Die Energieversorgung des Systems ist für den Demonstrator nicht zwingend mobil einzuplanen. Nach Möglichkeit ist aber durch Verwendung energiesparender Komponenten und einer entsprechend dimensionierten Energiequelle eine spätere mobile Einsetzbarkeit zu erleichtern.

Da es sich bei dem zu entwickelnden System um einen Funktionsprototypen handelt, stehen Kriterien wie Zuverlässigkeit, Aussehen, Wartbarkeit, Sicherheitsanforderungen oder die Betriebs- und Umweltbedingungen nicht im Vordergrund. Daher werden diesbezüglich keine Anforderungen gestellt. Wichtig hingegen sind eine einfache Bedienbarkeit des Systems, die Portierbarkeit der Funktionalität auf andere Systeme, aufgrund des kleinen Zeitrahmens der Arbeit eine einfache Implementierbarkeit und geringe Kosten. Durch die Verwendung von offener Hard- und Software sowie die Verwendung von weit verbreiteten Komponenten können die Portierbarkeit erhöht und die Kosten gesenkt werden. Durch eine übersichtlich und einfach gehaltene Benutzerschnittstelle ist die verständliche Bedienung zu ermöglichen.

### Auswahl der Komponenten

Mit Blick auf die im vorherigen Kapitel beschriebenen Anforderungen und Kriterien erfolgt in diesem Abschnitt die Auswahl der für den Demonstrator zu verwendenden Komponenten. Da diese untereinander kompatibel sein müssen, wird zunächst die Hardware des Hauptrechners bestimmt und davon ausgehend die übrigen Bestandteile des Systems. Eine Materialliste der verwendeten Komponenten ist in Anhang A zu finden.

#### Hauptrechner

In [26] ist eine aktuelle Übersicht über gängige Einplatinencomputer mit ihren jeweiligen Vor- und Nachteilen wie bspw. verfügbare Anschlüsse, Leistungsfähigkeit und Preis zu finden. Nur aufgrund von Leistung und Preis würde die Wahl nicht auf den weit verbreiteten Raspberry Pi 3 fallen, allerdings sind noch andere Kriterien zu beachten. Um den Entwicklungsaufwand möglichst zu reduzieren ist eine gute Verfügbarkeit von Bibliotheken für Bildverarbeitungsaufgaben und Kommunikation über die vorhandenen Schnittstellen erforderlich. Die Größe der Anwendergemeinde bestimmt darüber, wie schnell und ob überhaupt Hilfestellungen erhalten werden können. Weiterhin ist eine Plattform, die seit Jahren weiterentwickelt wird, i.d.R. zuverlässiger als eine Neuentwicklung. Diese Punkte sprechen in Summe für den Raspberry Pi 3 als Hauptrechner für das vorliegende Projekt.

#### Sensoren

Für die Bildverarbeitung wird eine Kamera benötigt, an die wie oben erwähnt keine besonderen Ansprüche bestehen. Die erste Version des Raspberry Pi Kameramoduls ist günstig, klein, leicht, bietet eine Auflösung von maximal 2592 mal 1944 Pixeln und einen Videomodus mit 30 Bildern pro Sekunde bei FullHD-Auflösung [27]. Vor allem aufgrund der Kompaktheit ist dieses Modul für den zu entwickelnden Prototypen interessant. Es existiert eine neuere Version des Moduls mit besseren Leistungsdaten, welche allerdings nicht benötigt werden. Zudem ist der Preis höher, sodass hier die erste Version des Moduls ausgewählt wird.

Für die Abstandsberechnung des Effektors zum Ziel kann ein Abstandsensor, wie z. B. ein Time-of-Flight-Sensor, zum Einsatz kommen. Mit dieser dedizierten Hardware ist eine genaue und schnelle Bestimmung des Abstands zum Ziel möglich. Da aber bei dem zu entwickelnden Prototyp zunächst die Kosten im Vordergrund stehen und schon eine Kamera vorhanden ist, soll die Entfernungsberechnung mit dieser durchgeführt werden.

Zur Vermeidung von Schäden ist am Effektor ein Sensor zu verwenden, welcher die Überschreitung eines Grenzwerts der Krafteinwirkung erkennt. Für den Prototyp soll der Prozess der Tastererkennung und -betätigung zunächst anhand von Fotos von Fahrstuhlknöpfen simuliert werden. Daher ist ein einfacher Endlagenschalter mit einer kleinen Feder ausreichend.

#### Manipulator

In der Zieleinsatzumgebung kommt im Allgemeinen ein Roboterarm zum Einsatz, daher soll nach Möglichkeit auch für den Prototyp ein solcher Aktor verwendet werden. Aus Zeitgründen ist eine Neukonstruktion in der vorliegenden Arbeit nicht machbar. Zur Auswahl stehen die Anschaffung eines fertigen Manipulators oder eines Bausatzes sowie der Aufbau nach frei zugänglichen Bauplänen. Sowohl für fertige Roboter als auch für Bausätze gilt, dass sie entweder zu teuer sind oder ein geschlossenes System bilden, dass nur manuell steuerbar ist. Daher erfolgt ein Nachbau einer kinematischen Struktur, welcher zudem die Flexibilität bei der Wahl der Motoren, der Steuerungshardware und damit auch der Umsetzung der Steuerungssoftware erhöht.

Walter [28] ist ein Open Source Projekt für einen Gelenkarmroboter mit sechs Rotationsachsen. Baupläne für die Steuerungshardware werden ebenso zur Verfügung gestellt wie die Steuerungssoftware, welche bspw. eine Bahnplanung mit Interpolation mittels Bézier-Kurven bietet. Der Preis der Komponenten und der Aufwand zum Aufbau sprengen allerdings den Rahmen dieser Arbeit.

Die Auswahl an günstig herzustellenden Robotern mit frei verfügbaren Bauplänen ist beschränkt, insbesondere wenn sie von mehr als nur einem Anwender erprobt bzw. getestet sind. Umsetzbare Vorlagen beinhalten kleinere kinematische Strukturen, die Steuerung geht häufig nicht über die manuelle Bewegung der einzelnen Achsen hinaus. Eine Berechnung der inversen Kinematik zur Positionierung des Effektors erfolgt bspw. in einem Projekt mit einem 3‑Achs-Roboter mit zwei gleich langen Armen. Umfangreichere Berechnungen sowie eine Bahnplanung bietet von den hier betrachteten Projekten nur der oben genannte Roboter Walter, diese sind aber für den Demonstrator nicht zwingend erforderlich.

Geeignet für die Anwendung in der vorliegenden Arbeit scheint besonders das Modell eines ABB IRB460 im Maßstab 1:7, welches eine umfassende Anleitung bietet und von zahlreichen Nutzern (momentan über 80) erfolgreich nachgebaut wurde [29][[13]](#footnote-13). Zudem ist der Roboterarm durch die Verwendung 3D-gedruckter Teile preiswert und einfach aufzubauen. Drei Achsen bzw. Freiheitsgerade beschränken zwar den Arbeitsbereich, dafür ist die inverse Kinematik über den Kosinussatz bestimmbar. Somit muss kein iteratives Verfahren o.ä. implementiert werden. Zudem stellen weniger Gelenke auch weniger Fehlerquellen in der Positionierung dar. Der Effektor wird durch die Verwendung einer verknüpften Kinematik immer in der Horizontalen gehalten. Da es sich um keinen professionellen Industrieroboter handelt, muss mit einer geringen Präzision, Wiederholgenauigkeit und Robustheit gerechnet werden.

#### Steuerung des Manipulators

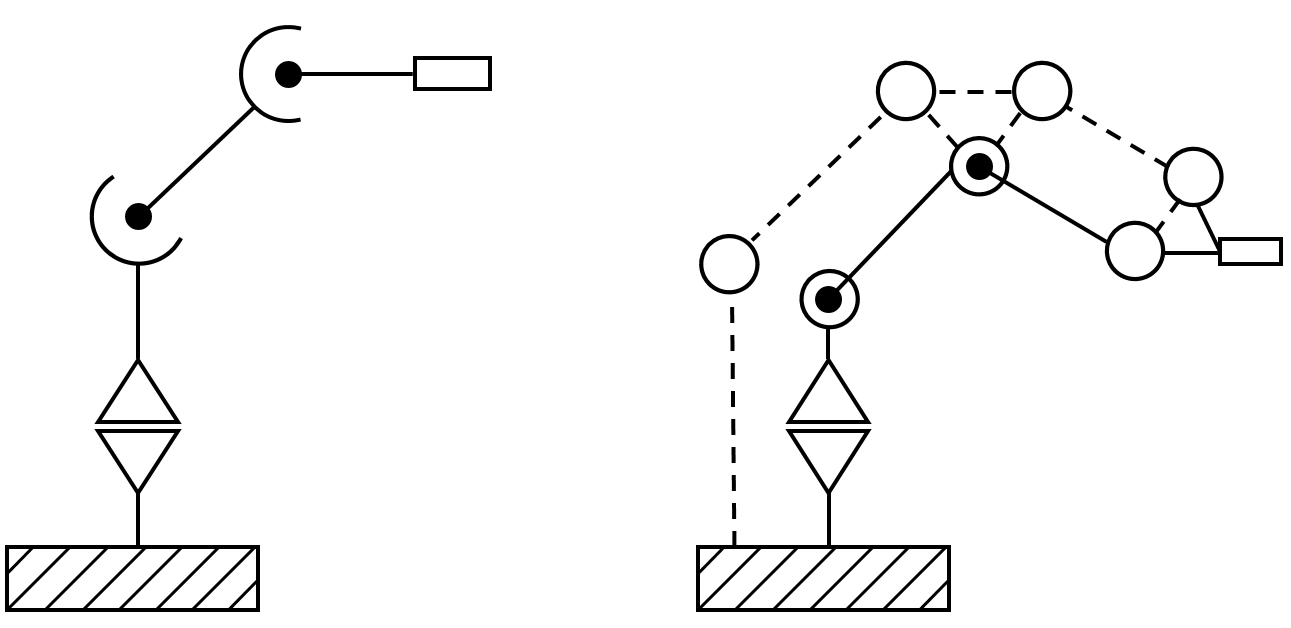
Durch den Einsatz dedizierter Hardware für die Steuerung des Manipulators können Ressourcenprobleme vermieden werden. Die Trennung der Software für die Robotersteuerung von der restlichen Programmlogik erhöht die Flexibilität, so ist eine andere Programmiersprache einsetzbar, der Code übersichtlicher und die Fehlersuche einfacher.

Die Verwendung eines verbreiteten Mikrocontrollers wie dem Arduino Uno bietet eine große Auswahl an Softwarebibliotheken und Programmierbeispielen. Das hierfür erhältliche Erweiterungsmodul CNC-Shield in Verbindung mit Schrittmotortreibern des Typs A4988 vereinfacht die Steuerung der Motoren. Grundsätzlich sind auch andere Mikrocontroller oder Treiber einsetzbar, der günstige Preis und die Eignung für die Steuerungsaufgabe führen hier zu einer willkürlichen Entscheidung für die bewährte Kombination aus Mikrocontroller, Erweiterungsmodul und Motortreibern. Die Kommunikation mit dem Hauptrechner kann über die serielle Schnittstelle unter Verwendung eines USB-Kabels erfolgen. Der zusätzliche Energiebedarf ist gering und kann für den Prototypen vernachlässigt werden.

### Aufbau des Systems

Translations- und Rotationsachsen (Skizze)

Kinematischer Aufbau



Eye-In-Hand Entscheidung!

Warum Eye-In-Hand? Kein Laserscanner o.ä., Ermittlung der Entfernung nur über Kamera möglich -> aus Veränderung durch Bewegung

Grenzwert für Winkel kann bei eye-in-hand systembedingt nicht überschritten werden wegen FOV

Komponenten, Controller, Treiber (hier?) etc.

Kamera, Effektor (mit Sensor/Endlagenschalter), Manipulator (mit Motoren), Controller (mit Treibern), Computer für Programmlogik (evtl. getrennt von Controller Vor-/ Nachteile) getrennte Funktionsbereiche (Steuerung Aktor, Programmlogik)

Aufbauskizze mit Kommunikationspfaden (Informationspfade, Daten)?

Komponentendiagramm oder so ähnlich

Unsicherheiten

Mögliche Fehlerquellen!!!

## Objekt- und Texterkennung

Zwei Verfahren -> Test Geschwindigkeit auf vorhandener Hardware

Erst Taster detektieren, dann Text auslesen

### Maschinelles Detektionsverfahren mit vorgegebenem Merkmalsraum

### Deep Learning Ansatz zur Detektion ohne Merkmalsraumvorgabe

… (Convolutional Neural Networks)

Transfer Learning! (relativ wenig Daten verfügbar (Anzahl Bilder bzw. ROIs) und schneller trainiert)

### Auslesen der Tasterbeschriftung

Textdetektion (nicht recognition) als Herausforderung

Recognition im Prinzip Klassifikation (jedes Zeichen ein Bild)

MNIST, char74k etc.

Vorgefertigte Bibliotheken

Alle Ansätze

## Objektverfolgung

Ziel / Unterschied bewegter Hintergrund

Hier Teil des Visual Servoing

Nur Translation interessant?

Kamera nicht „gedämpft“! (aber Kamera bewegt)

Laufzeit / mögliche Frequenz untersuchen

### Echtzeit-Objektverfolgung in Videosequenzen

Wäre einfacher (?) bzw. vorzuziehen (warum?)

Normalerweise unbewegter Hintergrund -> zu verfolgende Objekte bewegen sich durch das Bild (Bsp. Fußgänger oder Autos)

Hier aber bewegter Hintergrund, da Kamera bewegt

Bei höherer Frequenz Steuerbefehle schneller anpassbar

### Merkmalsabgleich und perspektivische Transformation zwischen Einzelbildern

## Entfernungsberechnung

2D Bild -> 3D Informationen

### Berechnung durch Änderung des Winkels zum Ziel

### Berechnung durch Änderung der Bildgröße des Ziels

## Bewegungssteuerung

Hier Teil des Visual Servoing

Erst Mechanik, Übersetzung Zahnräder etc.

IK

Dann Software

### Wahl des Koordinatensystems

Bezugspunkt?

Bezugssystem! (s. Robotik.pdf)

kartesisch

### Positionierung des Effektors im Raum

Inverse Kinematik…

Bewegungssteuerung -> PTP (?)

## Integration der Teillösungen

### Benutzerschnittstelle

### Interaktion der Komponenten

Kommunikation

Befehlssatz (oder in 4) (dann im Anhang)

### Ablauf des Erkennungs- und Betätigungsprozesses

# Prototypische Umsetzung

Implementierung und Auswertung der Teillösungen

Nur ein Modell!

Verfahren zur Tasterdetektion

Verwendete Bibliothek(en)

Wie womit implementiert

Auswertung / Gegenüberstellung der Verfahren

Zielverfolgung bei Kamerabewegungen

Firmware zur Bewegungssteuerung des Manipulators

Ansteuerung der Motoren

Kommunikation und Befehlssatz

Erkennung einer Tasterbetätigung

Endlagenschalter

Optional „nachgiebiger Finger“, der erst ab bestimmter Krafteinwirkung auslöst

Implementierung

Firmware Manipulator

Tests etc.

Test des Gesamtsystems

## Hardware

### Verwendete Komponenten

Endlagenschalter

### Schnittstellen

### Aufbau des Prototypen und des Tastermodells

## Software

### Realisierung der Bildverarbeitungsfunktionen

### Firmware zur Bewegungssteuerung

Mit Befehlssatz (oder in nächsten Abschnitt)

Erkennung Tasterdetektion

### Schnittstellen

Benutzer und Systemintern (nach außen und innen)

### Programmstruktur

## Ergebnisse der Prototypenentwicklung

# Fazit und Ausblick

# Literaturverzeichnis

|  |  |
| --- | --- |
| [1] | Bundesministerium für Gesundheit, „Abschlussbericht zur Studie: Unterstützung Pflegebedürftiger durch technische Assistenzsysteme,“ VDI/VDE Innovation + Technik GmbH, IEGUS – Institut für Europäische Gesundheits- und Sozialwirtschaft GmbH, Berlin, 2013. |
| [2] | T. Christaller, M. Decker, J. Gilsbach und G. Hirzinger, Robotik - Perspektiven für menschliches Handeln in der zukünftigen Gesellschaft, Berlin Heidelberg: Springer-Verlag , 2001. |
| [3] | DIN Deutsches Institut für Normung e.V., *Roboter und Robotikgeräte - Wörterbuch (ISO/DIS 8373:2010); Entwurf November 2010,* Berlin: Beuth Verlag GmbH, 2010. |
| [4] | VDI, „VDI Richtlinie 2860: Montage- und Handhabungstechnik; Handhabungsfunktionen, Handhabungseinrichtungen; Begriffe, Definitionen, Symbole,“ VDI, Düsseldorf, 1990. |
| [5] | Maheu et al., „Evaluation of the JACO robotic arm: Clinico-economic study for powered wheelchair users with upper-extremity disabilities,“ in *IEEE international conference on rehabilitation robotics (ICORR)*, Zürich, 2011. |
| [6] | Vogel et al., „An assistive decision-and-control architecture for force-sensitive hand–arm systems driven by human–machine interfaces,“ *The International Journal of Robotics Research (IJRR) , vol. 34, no. 6,* pp. 763-780, Mai 2015. |
| [7] | C. Martens, O. Prenzel und A. Graese, „The Rehabilitation Robots FRIEND-I & II: Daily Life Independency through Semi-Autonomous Task-Execution,“ in *Rehabilitation Robotics, Sashi S Kommu*, IntechOpen, 2007. |
| [8] | Universität Bremen - Institute of Automation, „Assistenzroboter FRIEND,“ 16 Januar 2014. [Online]. Available: http://www.iat.uni-bremen.de/sixcms/detail.php?id=555. [Zugriff am 18 Mai 2018]. |
| [9] | L. Priese, Computer Vision - Einführung in die Verarbeitung und Analyse digitaler Bilder, Berlin Heidelberg: Springer Vieweg, 2015. |
| [10] | M. Hassaballah, A. Abdelmgeid und A. Hammam, „Image Features Detection, Description and Matching,“ in *Image Feature Detectors and Descriptors: Foundations and Applications - Studies in Computational Intelligence (Book 630)*, Switzerland, Springer International Publishing, 2016, pp. 11-45. |
| [11] | H. Süße und E. Rodner, Bildverarbeitung und Objekterkennung - Computer Vision in Industrie und Medizin, Wiesbaden: Springer Vieweg, 2014. |
| [12] | P. Viola und M. Jones, „Robust Real-time Object Detection,“ in *SECOND INTERNATIONAL WORKSHOP ON STATISTICAL AND COMPUTATIONAL THEORIES OF VISION – MODELING, LEARNING, COMPUTING, AND SAMPLING*, Vancouver, 2001. |
| [13] | A. T., H. A. und P. M., „Face Recognition with Local Binary Patterns,“ in *Computer Vision - ECCV 2004. ECCV 2004. Lecture Notes in Computer Science, vol 3021*, Berlin, Heidelberg, Springer-Verlag, 2004. |
| [14] | American with Disability Act, „ADA Compliance Directory - ELEVATORS,“ 2018. [Online]. Available: http://www.ada-compliance.com/ada-compliance/ada-elevators.html. [Zugriff am 12 05 2018]. |
| [15] | E. Klingbeil, B. Carpenter, O. Russakovsky und A. Y. Ng, „Autonomous operation of novel elevators for robot navigation,“ in *2010 IEEE International Conference on Robotics and Automation*, 2010. |
| [16] | A. Krizhevsky, I. Sutskever und G. E. Hinton, „ImageNet Classification with Deep Convolutional Neural Networks,“ in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou und K. Q. Weinberger, Hrsg., Curran Associates, Inc., 2012, pp. 1097-1105. |
| [17] | Y. Lecun, L. Bottou, Y. Bengio und P. Haffner, „Gradient-based learning applied to document recognition,“ in *Proceedings of the IEEE*, 1998. |
| [18] | R. B. Girshick, J. Donahue, T. Darrell und J. Malik, „Rich feature hierarchies for accurate object detection and semantic segmentation,“ *CoRR,* Bd. abs/1311.2524, 2013. |
| [19] | A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto und H. Adam, „MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,“ *CoRR,* Bd. abs/1704.04861, 2017. |
| [20] | J. Hui, „Object detection: speed and accuracy comparison (Faster R-CNN, R-FCN, SSD, FPN, RetinaNet and YOLOv3),“ 27 März 2018. [Online]. Available: https://medium.com/@jonathan\_hui/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359. [Zugriff am 20 04 2018]. |
| [21] | Y. Zhang, H. Peng und P. Hu, „CS341 Final Report: Towards Real-time Detection and Camera Triggering,“ in *Project in Mining Massive Data Sets*, 2017. |
| [22] | F. Chaumette und S. Hutchinson, „Visual servo control, Part I: Basic approaches,“ *IEEE Robotics and Automation Magazine,* Bd. 13, pp. 82-90, 2006. |
| [23] | G. Palmieri, M. Palpacelli, M. Battistelli und M. Callegari, „A Comparison between Position-Based and Image-Based Dynamic Visual Servoings in the Control of a Translating Parallel Manipulator,“ *Journal of Robotics,* Bd. 2012, pp. 1-11, 2012. |
| [24] | R. Pieters, *Visual Servo Control - Vorlesung,* Eindhoven: Eindhoven University of Technology, 2012. |
| [25] | E. Karami, S. Prasad und M. S. Shehata, „Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images,“ *CoRR,* Bd. abs/1710.02726, 2017. |
| [26] | 1&1 Internet SE, „Vier aktuelle Raspberry-Pi-Alternativen – Einplatinenrechner im Vergleich,“ 2018 Januar 25. [Online]. Available: https://hosting.1und1.de/digitalguide/server/knowhow/raspberry-pi-alternativen-einplatinenrechner-im-check/. [Zugriff am 29 Januar 2018]. |
| [27] | J. Hughes, „Raspberry Pi Camera Module,“ 3 März 2017. [Online]. Available: https://www.raspberrypi.org/documentation/hardware/camera/README.md. [Zugriff am 6 Januar 2018]. |
| [28] | J. Alt, „Walter,“ 2 März 2017. [Online]. Available: https://walter.readthedocs.io/en/latest/. [Zugriff am 20 2 2018]. |
| [29] | J. Le, „Robot Arm MK2 Plus,“ 6 September 2017. [Online]. Available: https://www.thingiverse.com/thing:2520572. [Zugriff am 5 Januar 2018]. |
| [30] | Assistive Innovations, „iARM,“ Assistive Innovations, [Online]. Available: https://www.assistive-innovations.com/de/roboterarme/iarm-de. [Zugriff am 18 Mai 2018]. |

# Abbildungsverzeichnis

[Abbildung 1: Assistenzsystem FRIEND der Universität Bremen [7] 4](#_Toc515452038)

[Abbildung 2: Vier Grundtypen zur Merkmalsberechnung nach Viola und Jones in verschiedener Skalierung und Position [12] 6](#_Toc515452039)

[Abbildung 3: Erkannte Taster (a) und zugehöriges ermitteltes Gitter (b) [15]. 7](#_Toc515452040)

# Anhang A: Materialliste

1. Als Beispiel in Schaubildern oder Videos dient die Hilfe beim Trinken. [↑](#footnote-ref-1)
2. Ein Beispiel für ein verfügbares System ist der iARM der Firma Assistive Innovations [19], welcher ähnliche Funktionen wie der JACO bietet. [↑](#footnote-ref-2)
3. Die Preie von JACO und iARM liegen im Bereich von mehreren Zehntausend Euro. [↑](#footnote-ref-3)
4. Ein Merkmal ist eine numerische oder qualitative Information, die aus Bildpunkten einer Region (lokal) oder des gesamten Bildes (global) berechnet wird. Merkmale können bspw. farb‑, form- oder texturbasiert sein, aber auch durch abstrakte Operationen entstehen [10]. [↑](#footnote-ref-4)
5. In Deutschland entspricht nur ein Teil der Fahrstuhltaster diesen Richtlinien. [↑](#footnote-ref-5)
6. Beim Sliding-Window-Prinzip wird ein Fenster vordefinierter Größe Pixel für Pixel über das gesamte Bild verschoben und an jeder Position eine Klassifikation vorgenommen [11]. [↑](#footnote-ref-6)
7. Resultat ist eine Reduktion der Detektionszeit auf ca. 50 Sekunden auf einer High-End-CPU des Jahres 2013 [16]. [↑](#footnote-ref-7)
8. Insbesondere hat auch die Architektur der verwendeten tiefen neuronalen Netze, wie z. B. Anzahl und Typ der Schichten, einen Einfluss auf Erkennungsrate und Ausführungszeiten. MobileNets eignen sich für mobile Geräte und erzielen trotz kompakter Größe dennoch hohe Erkennungsraten [17]. [↑](#footnote-ref-8)
9. Die Detektionsrate liegt im Bereich von 0,1 fps bis 0,7 fps (frames per second) [↑](#footnote-ref-9)
10. Zu geometrischen Transformationen und homogenen Koordinaten siehe bspw. [11]. [↑](#footnote-ref-10)
11. Die Form der Koordinaten (bspw. kartesisch) und das Bezugssystem sind noch zu definieren. [↑](#footnote-ref-11)
12. Da an dieser Stelle keine genauen Aussagen getroffen werden können, wird der Grenzwert willkürlich auf 45° festgelegt. [↑](#footnote-ref-12)
13. Von dem hier beschriebenen Modell existieren zwei Varianten, von denen eine Schrittmotoren anstelle von Servomotoren verwendet. Da das verwendete Schrittmotormodell ein höheres Drehmoment und eine höhere Positioniergenauigkeit gegenüber dem eingesetzten Servomotor bietet, wird hier die erste Variante gewählt. [↑](#footnote-ref-13)