Roboter-Roboter Interaktion zur autonomen Objektübergabe

Master-Abschlussarbeit,

eingereicht am Fachbereich Technik
der FH Bielefeld

von Robin Rasch,
geboren am 05.02.1991 in Minden

13. Dezember 2015

Mit meiner Unterschrift bestätige ich, dass ich die Arbeit selbständig und nur mit den zugelassenen Hilfsmitteln erstellt habe.

Minden, den .....

Zusammenfassung

Diese Arbeit befasst sich mit der Entwicklung und Implementierung eines autonomen Roboter Systems. Dieses System bildet das Szenario einer Objektübergabe zwischen zwei selbstständigen Manipulatoren ab. Bei diesen beiden Manipulatoren handelt es sich um den Typ-gleichen Roboterarm des YouBot der Firma Kuka. Einer der Arme ist dabei stationär auf einem Tisch fixiert, der andere auf der mobilen YouBot Plattform.

Nach einer kurzen Einführung über das Nutzen von Robotern im Haushalt, sowie im intelligenten Gebäude, werden zunächst die Grundlagen und folgend ein aktueller Stand der Technik zusammengefasst. Anschließend wird sich diese Arbeit mit den zentralen Aspekten der Entwicklung und Implementierung befassen. Bei der Entwicklung wurden verschiedene Thematiken und Probleme der Robotik und Computer Vision berücksichtigt und bearbeitet. So wird in den folgenden Kapiteln auf das Setup der Roboter eingegangen, sowie den Themen der inversen Kinematik, der Segmentierung und Objekterkennung. Da das System in einem intelligenten Gebäude zum Einsatz kommen soll, ist die Thematik der Vernetzung einzelner Subsysteme und deren Zusammenspiel ebenfalls ein Bestandteil dieser Arbeit. Die Implementierung befasst sich mit der Algorithmik der einzelnen Probleme. Dabei wird Code exemplarisch vorgestellt, die vollständige Implementierung befindet sich im Anhang. Den Abschluss der Arbeit bilden eine Beurteilung und ein Fazit der umgesetzten Lösung, auch wird ein Ausblick über mögliche zukünftige Weiterentwicklungen gegeben.

Abstract

This paper considers the development and implementation of an autonomous robotic system. This system describes a scenario from a handover between two independent manipulators. These two manipulators are equally robot arms from the Kuka YouBot. One is stationary fixed at top of a table, the other one is on a mobile YouBot base.

After a short introduction about the harness of service robots at smart houses, this work outlines the principles and the state of the art. Following this paper attends to the key aspects of the development and the implementation. The development observes different topics and set of problems for robotic and computer vision. There will be a variety of subjects like roboter setup, inverse kinematic, segmentation and object recognition in the following chapters. Based on the operation at smart houses, networking and distributed systems are a topic in this paper, too. The implementation sink in algorithmic and solutions for single problems. There will be snippets of programm code, the complete code is at the appendix. A valutation and a conclusion of the implemented solutions, and also a prospect of possible future projects, form the ending of this work.

Inhaltsverzeichnis

1 Einleitung **1**

2 Grundlagen **5**
2.1 Nomenklatur 5
2.2 ROS: Robotic Operating System 6
2.2.1 Package und Nodes 6
2.2.2 Master und Parameter Server 7
2.2.3 Topics und Messages 8
2.2.4 Services 9
2.2.5 Action-Lib 9
2.3 Kinematik 10
2.3.1 Mechanismus 10
2.3.2 Kinematik 11
2.3.3 Forward Kinematics 11
2.3.4 Inverse Kinematik 12
2.4 Aufbau 13
2.4.1 Aufbau Teststand 13
2.4.2 YouBot 13
2.4.3 Sensoren 17
2.4.4 Netzwerk 17

3 Stand der Technik **18**
3.1 Physically Embedded Intelligent Systems - PEIS 18
3.2 Multirobots System - Komposition und Konfiguration 18
3.3 Handover - Übergabe zwischen Roboter-Maschine und Roboter-Roboter 18

4 Zusammenfassung und Ausblick **19**

5 Danksagung **20**

6 Literaturverzeichnis **21**

A Erster Anhang **22**

B Noch ein Anhang **22**

Abbildungsverzeichnis

1 Roboter in Seniorenheimen 2
2 Beispiel Szenario 1 3
3 Erweiterung Szenario 1 3
4 Master Service 7
5 Netzwerk Entlastung durch Peer-to-Peer Verbindungen 8
6 Topic Beispiel 8
7 YouBot Arm Kinematik 14
8 Arbeitsraum YouBot. Bilderquelle:Florek-Jasinska [2015] 15
9 YouBot Base 16
10 Mobile YouBot Plattform mit Sensorplatte. Bilderquelle:Kuka [2015] 17

Tabellenverzeichnis

1 Nomenklatur 5
2 YouBot Arm Joints 14

1 Einleitung

Der Demographische Wandel in Deutschland stellt die Gesellschaft und die Politik vor ein großes Problem. Nicht nur der fehlende Nachwuchs, der ein Arbeitnehmerloch hinterlässt, sondern auch eine immer älter werdende Gesellschaft sorgen für viele Fragezeichen. Neben kommunalen Problemen, wie teure Infrastruktur, ist eins der größten Anliegen die Altersarmut. Ein sinkendes Rentenniveau und steigende Kosten werden es in Zukunft unmöglich machen für eine gute Pflegeversorgung zu bezahlen. [Zandonella, 2013] Neben den hohen Kosten in der Pflege sind auch andere Faktoren die zu einer nicht akzeptablen Situation führen. So ist der Pflegeberuf zu unattraktiv für viele junge Menschen, wodurch auch hier ein großes Loch an Arbeitnehmern♀wahrzunehmen ist. Körperlich anstrengende Arbeit, die in kurzer Zeit ausgeführt werden muss, führen zu vielen physischen und psychischen Erkrankungen der Pfleger♀.[B. Badura, 2005]

Einen alternativen Weg bringt die Technik. Einfache zu bedienende Systeme können den Alltag vereinfachen und so auch älteren Menschen ein selbstständiges Leben ermöglichen. Alltägliche Aufgaben müssten nicht mehr von Pflegern♀übernommen werden, sondern könnten durch Maschinen erledigt werden. Schon heute können einzelne Kleinsysteme Haustätigkeiten wie Staubsaugen und Rasenmähen übernehmen. Ein Vorreiter auf diesem Gebiet ist das Roboterland Japan. Dort überlegte Kobayashi Hisato, Professor für Maschinenbau und Robotik an der Hosei-Universität in Tokio, sich schon 1999 Konzepte für Senioren-Service Roboter.[Wagner, 2009] Nach seinen Vorstellungen sollten dabei Roboter nicht autonom arbeiten, sondern von Familienangehörigen ferngesteuert und überwacht.[Kobayashi, 1999] Neben diesen Konzepten kann man aber auch schon angewandte Robotik in Japans Seniorenpolitik finden. Zwei Musterbeispiele zeigen dabei die unterschiedlichen Anwendungsszenarien für Roboter im privaten Umfeld. *Sinére Kōrien* der Firma Matsushita ist ein digitales Seniorenheim. Neben einer Smarthouse Anbindung gehört auch der Roboterteddy Kō-chan zur Ausstattung der einzelnen Zimmer. Dieser dient als Unterhaltungsroboter und Kommunikationsgerät mit den Pflegern♀. So kann mit den Kameraaugen im Teddy eine Aufnahme vom Raum gemacht werden und im Notfall den Pflegern♀eine Alarmmeldung geschickt werden. Weitere Sensoren, zu Beispiel Gewichtssensoren unter den Betten, geben Informationen über die Abwesenheiten von Patienten♀.[Wagner, 2009] Ein weiterer Anwendungsbereich für Roboter ist die *robotto serapī* (Robotertherapie). Dabei beschäftigen sich die Senioren♀mit tierähnlichen Robotern, wie Hunden, Seeroben oder Katzen. Im Zentrum der Therapie steht die Interaktion zwischen Patient♀und Roboter. Die Robotertherapie soll die Patienten♀aktivieren und deren Tagesabläufe abwechslungsreich gestalten. Außerdem steigert es die Kommunikation zwischen zwei Patienten♀, die am selben Roboter arbeiten. [Wagner, 2009]

Nicht nur in Japan, sondern auch in Deutschland wird sich mit dem Thema *Care(rob)bots* befasst. So finden sich unter dem Stichpunkt *Mensch-Maschine-Entgrenzungen* Studien zu der Thematik. Andere Untersuchungen befassen sich mit der Gegenseite, der Akzeptanz der Senioren♀für Roboter. So ergab eine Befragung der VDE-Studie "Mein Freund der Roboter", dass eine Mehrheit (56%) der Senioren♀Robotern im Haushalt offen gegenüber stehen und diese einem Pflege-/Altersheim vorziehen würden. Neben den bekannten Staubsauger- und Rasenmährobotern, sind es auch zukünftige Anwendung, wie ein roboterisierter Rollstuhl, die hohe

(a) Roboterteddy Kō-chan Quelle: [Panasonic, 2005]

(b) Roboterkatze zur Therapie Quelle: [Wagner, 2009]

Abbildung 1: Roboter in Seniorenheimen

Akzeptanzwerte erreichen. Die Studie zeigte aber auch, dass Senioren♀zunächst Robotern skeptisch gegenüberstehen. Spontan lehnten 40 Prozent der Senioren♀Roboter in ihrem privaten Umfeld ab, 60 Prozent empfanden Robotik sogar als unheimlich. jedoch zeigte sich, dass der Wunsch nach einer selbstständigen Lebensführung ein starker Faktor für die Akzeptanz ist. Dadurch ergibt sich eine Beliebtheit für Serviceroboter. So sind Roboter, die abgrenzbare Tätigkeiten im Haushalt selbständig erledigen, sehr beliebt. Wichtige Kriterien für die Akzeptanz waren zudem die intuitive Bedienbarkeit, die Robustheit und die Flexibilität gegenüber unterschiedlicher Handicaps. Auch menschliche Faktoren wie Geduld, Verständnis, Höflichkeit und Achtung der Intimsphäre waren den Anwendern♀wichtig.[Meyer, 2011]

Neben den sozialen Forschungen gibt es in Deutschland auch ingenieurwissenschaftliche Ergebnisse auf diesem Gebiet. So beschäftigt sich das Cluster of Excellence Cognitive Interaction Technology (CITEC) in Bielefeld, zusammen mit der Stiftung Bethel, in der Forschung auf dem Gebiet der Unterstützung für Demenzkranke. Die dort entwickelten Assistenzsysteme helfen den Patienten unter anderem beim Zähneputzen und geben ihnen so ein Stück Selbstständigkeit zurück.

Viele Forschungen und Entwicklungen beschäftigen sich momentan mit der Mensch-Maschine/Roboter Beziehung. Dabei geraten die Roboter-Roboter-Interaktionen in den Hintergrund. Da aber heutige Roboter keine Allround-Lösungen bieten, sondern meist hochgradig spezialisiert sind und nur eine Tätigkeit ausführen können(zum Beispiel Saugroboter, Fensterwischroboter und weitere), ist das Zusammenspiel und die Komposition der Roboter wichtig. Das Szenario in Abbildung 2 zeigt, wie solch eine Komposition aussehen könnte.

Dabei führt jeder Roboter selbstständig und unabhängig von den anderen seine Arbeit aus. Die Kommunikation geht immer von der Zentralen Steuereinheit, dem *Master*, aus. Zwischen den einzelnen Robotern findet kein großer Informationsaustausch statt. Die Änderung an dem Szenario in Abbildung 3 ändert jedoch die Art der Zusammenarbeit und der Kommunikation. Nun ist es nötig, dass die Roboter notwendige Informationen, wie die Übergabepose oder eine Synchronisierung der Gripper, austauschen.

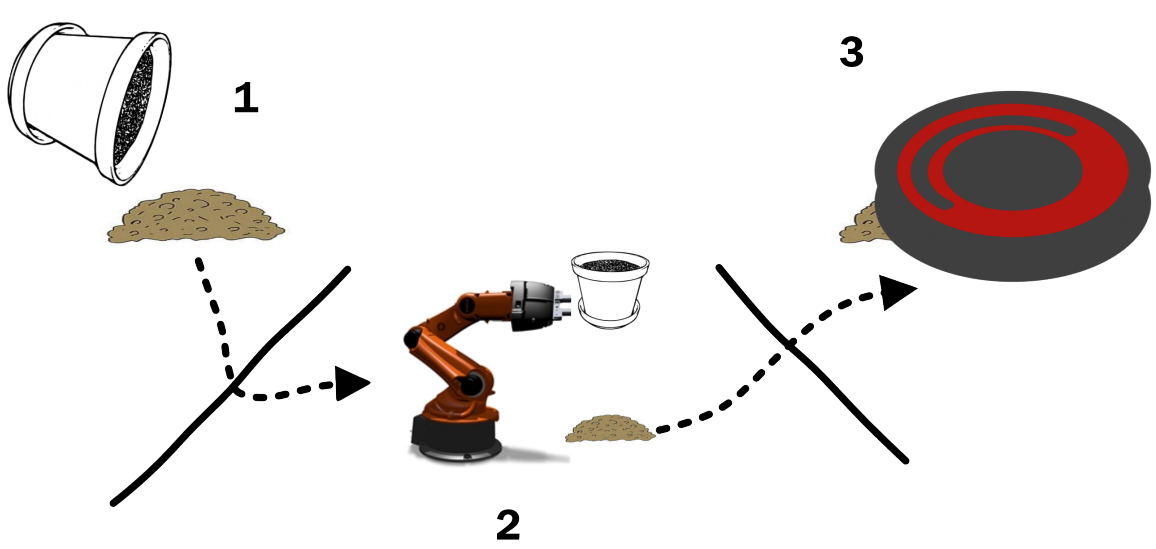


Abbildung 2: Mensch A stößt einen Blumentopf um (1). Ein Kamerasystem dediziert das der Topf umgefallen ist und Erde auf dem Boden liegt. Ein Roboter mit Arm wird gerufen, der zunächst die Vase aufhebt und an ihren Platz zurückstellt (2). Anschließend wird ein Saugroboter aktiviert, der die Erde weg saugt (3). Bei schwereren Verschmutzungen wird noch ein Wischroboter bestellt.

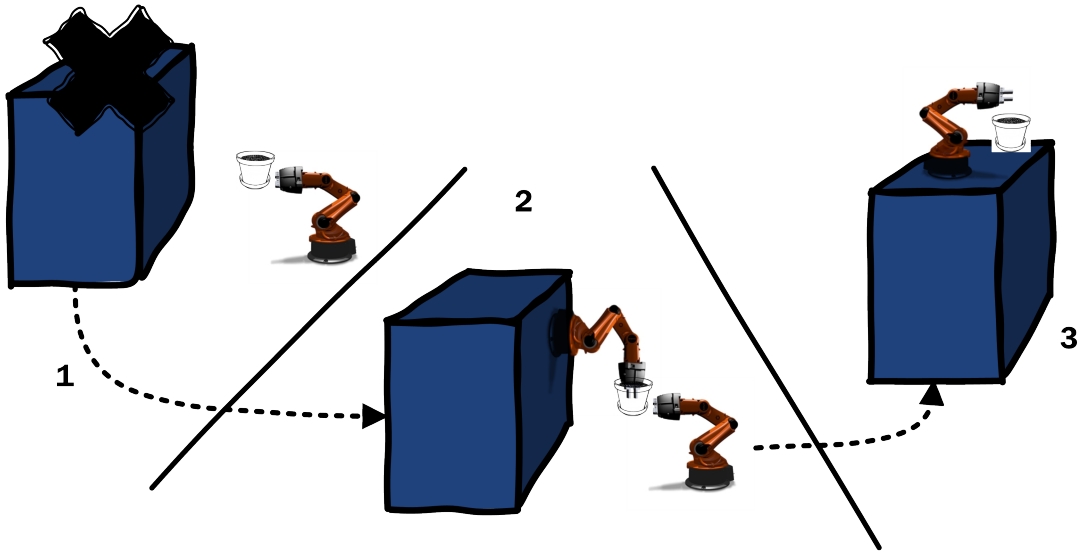


Abbildung 3: Beim Zurückstellen der Vase stellt der Roboter fest, dass er die gewünschte Position nicht erreichen kann(1), weil sein Arm zu kurz ist. Da auf der Anrichte noch ein weiterer Arm steht nimmt er die Kommunikation mit diesem Manipulator auf. Sie kommunizieren einen Übergabepunkt und führen ein Übergabemanöver durch. Arm 2 stellt nun die Vase an die gewünschte Position.

Der zentrale Aspekt dieser Arbeit befasst sich mit der Abstraktion von Szenario 2. Roboter 1 übergibt ein Objekt an Roboter 2. Dabei werden verschiedene Thematiken aufgegriffen, untersucht und erläutert. Das Folgende Kapitel befasst sich mit den Grundlagen dieser Arbeit, es werden einzelne Begriffe aus der Robotik und der Middleware **R**obot **O**perating **S**ystem erläutert, sowie der genutzte Laboraufbau dargestellt. Dabei wird die verwendete Hardware aufgelistet und im Detail betrachtet. Kapitel 3 zeigt schon bestehende Arbeiten im Bereich Robotik, die mit dieser Arbeit in Zusammenhang bestehen. So wird unter anderem der Begriff PEIS (Physically Embedded Intelligent Systems) erläutert und die Verbindung zur Thesis gezeigt. Des Weiteren werden Arbeiten für Multirobot Systeme und Handover-Methoden erklärt. Im darauf folgenden Kapitel

2 Grundlagen

Dieses Kapitel befasst sich mit den Grundlagen für die folgenden Kapitel. Es soll dem geneigten Leser Informationen zum Verständnis der Arbeit geben. Die anschließenden Unterkapitel beschreiben zunächst die verwendeten Mathematischen Ausdrücke und Bezeichner um die Lesbarkeit der späteren Formeln zu erleichtern. Nachfolgend wird die Middleware ROS genauer erläutert und unterschiedliche Funktionen sowie Einschränkungen aufgezeigt, die Auswirkungen auf die entwickelte Softwarearchitektur hatten. Darauf folgt eine Definition des Begriffs der Kinematik, welcher im späteren Verlauf der Arbeit für die Entwicklung der inversen Kinematik benötigt wird. Das abschließende Unterkapitel beschreibt die verwendete Hardware und den Laboraufbau im Detail. Weitere Informationen bezüglich der Robotik und grundlegender Mathematischer Begriffe sind im Buch "Robotics, Vision and Control"von Peter Corke zu entnehmen.

2.1 Nomenklatur

Die folgende Tabelle erläutert die genutzten mathematischen Symbole und orientiert sich an dem Buch "Robitcs, Vision and Control"Corke [2011]. Von links nach rechts sind das Symbol, die Einheit und die Beschreibung gegeben. Einige Symbole werden mehrfach verwendet und haben je nach Kontext eine andere Bedeutung.

Symbol	Einheit	Beschreibung
T	s	Messintervall
T		homogene Transformation, $T \in SE(2)$ oder $SE(3)$
${}^{A}T_{B}$		homogene Transformation zur Darstellung von Koordinatensystem B betrachtet von Koordinatensystem A. Koordinatensystem 0 bezeichnet das Weltkoordinatensystem und muss nicht explizit angegeben werden ${}^{0}T_{B} = T_{B}()$. Die inverse kann auf zwei Arten betrachtet werden: $({}^{A}T_{B})^{-1} = {}^{B}T_{A}$
θ	rad	Winkel im Bogenmaß. Wenn nichts weiteres gegeben ist sind Winkel immer im **Bogenmaß** zu sehen.
$\boldsymbol{\theta}$	rad	Vektor von Winkel im Bogenmaß.
$\theta_r\theta_p\theta_y$	rad	Roll-Pitch-Gear Winkel, beziehungsweise Rollen
α	°	Winkel in Grad. Wenn nichts weiteres gegeben ist sind Winkel immer im **Bogenmaß** zu sehen.
v	m s^{-1}	Geschwindigkeit.
$\boldsymbol{v}$	m s^{-1}	Geschwindigkeitsvektor. Meist $\in \mathbb{R}^3$.
ω	rad s^{-1}	Drehwinkel-Geschwindigkeit.
$\boldsymbol{\omega}$	rad s^{-1}	Drehwinkel-Geschwindigkeitsvektor. Meist $\in \mathbb{R}^3$.
$\boldsymbol{v}$		Geschwindigkeit-Drehung (engl. Twist, Screw). $\boldsymbol{v} \in \mathbb{R}^6$, $\boldsymbol{v} = (v_x, v_y, v_z, \omega_x, \omega_y, \omega_z)$
X,Y,Z		Kartesische Koordinaten.
ξ		Abstrakte Darstellung einer 3-dimensionalen Kartesischen Pose $\xi \in \mathbb{R}^6$
${}^{A}\xi_{B}$		Abstrakte Darstellung einer **relativen** 3-dimensionalen Kartesischen Pose, Pose B betrachtet von Pose A
$\oplus$		Hintereinanderausführung (Addition) zweier Posen
$\ominus$		Unärer Operator zur Invertierung von Posen

Tabelle 1: Nomenklatur

2.2 ROS: Robotic Operating System

Das Robotic Operating System, kurz ROS, ist ein Middleware-Framework zur Steuerung und Verwaltung von Personal-Robotern. Es verwendet eine Paket und Node Struktur um einzelne Aktoren und Sensoren einzelner Robotersystem zu betreiben. Das ROS dient dabei als Abstraktionsebene für die Entwickler und versucht die Anwendungsebene von der Hardwareebene zu trennen. Das Framework wird am Robotikinstitut Willow Garage in Zusammenarbeit mit der Open Source Robotics Foundation entwickelt, stammt aber ursprünglich vom Stanford Artificial Intelligence Laboratory. Dort wurde 2007 im Rahmen des Stanford-AI-Robot-Projektes die Entwicklung aufgenommen. ROS ermöglicht während der Entwicklung und des Betriebs ein Cross-Plattform System, da es kompatible Versionen für Windows (experimentell), Linux und Mac OS X gibt. Außerdem steht ein Multi(programmier)sprachen System zur Verfügung. So ist es möglich eigene Nodes mit C++, Python oder Java zu entwickeln.Quigley et al. [2009] Neben den Standardkonzepten und -paketen gibt es eine Community, die eigene Pakete auf der ROS-Homepage zur Verfügung stellt. Die folgenden Abschnitte erläutern die ROS-Kernkonzepte, welche für die Entwicklung eingesetzt wurden.

2.2.1 Package und Nodes

Packages dienen der strukturellen Verwaltung aller Dateien im ROS-System. Nodes, Launch-Files, Services, Actions und Messages sind immer an ihr Paket gebunden und können nur unter deren Namen erreicht werden. Für die Verlinkung und die Erreichbarkeit gebauter Pakete ist das ROS-interne Buildsystem *Catkin* zuständig. Es benötigt ebenfalls den **eindeutigen** Package-Namen für den Build und Verwaltungsprozess. Neben den ROS bezogenen Daten können in einem Paket aber auch ROS unabhängige APIs angelegt und verwaltet werden. So werden unter anderem einzelne Pakete für Treiber genutzt. Ein ROS Paket folgt dem "Goldilocks"Prinzip: Genug Funktionalität um nützlich zu sein, aber nicht zu viel, damit die Verwendbarkeit nicht zu komplex wird. Der Aufbau eines ROS Pakets findet sich in Anhang

Neben der strukturellen Verteilung gibt es auch die Funktionale Trennung. Dies wird durch einzelne *Nodes* erreicht. Jeder Node hat einen bestimmten Aufgabenbereich. Erst das Zusammenspiel mehrerer Nodes ergibt ein Robotersystem. So ist ein Node zum Beispiel für die Berechnung der Inversen Kinematik zuständig, ein anderer für die Lokalisierung des Roboters im Raum und ein dritter für das User-Interface. Diese Verteilung der Funktionalitäten hat mehrere Vorteile. Da jeder Node in einem eigenen Prozess gestartet wird ist das komplette System robuster. Jeder Node muss so entwickelt werden, dass er eigenständig weiterlaufen kann, auch wenn ein benötigter Node durch einen Fehler abgestürzt ist. Jeder Node wird mit einem System weit einmaligen Namen aufgerufen unter dem er vom System erreicht und angezeigt werden kann. Nodes kommunizieren untereinander mit Hilfe von Topics (siehe 2.2.3).Soll ein ROS Node gestartet werden kann dieser mit Hilfe des ROS-Befehls aufgerufen werden:

```
rosrun <paket-name> <nodetype-name>
```

Der Paketname und der Nodetype sind erfordliche Felder. Darüber hinaus können Parameter

angegeben werden die für den Node notwendig sind. Eine weitere Möglichkeit einen und mehrere Nodes zu starten ist ein Launchfile (siehe Anhang). In diesem können Nodes, Parameter und Argumente aufgelistet werden. Zu beachten ist jedoch, dass die Nodes in einer zufälligen Reihenfolge gestartet werden. Ein Launch-File kann mit folgendem Befehl gestartet werden:

```
roslaunch <paket-name> <launchfile-name>
```

Damit ein Node gestartet werden kann muss vorher ein ROS-Master aktiv sein, mit dem sich der Node verbinden kann. Ist der Node verbunden kann er auf die Topics, Services, Actions und den Parameter Server des Master zugreifen.

2.2.2 Master und Parameter Server

Der ROS Master ist ein zentraler Dienst für ganze System. Er beinhaltet einen Namen- und Registrierungs-Service an dem sich andere Nodes, auch Services, anmelden können. Der Master wird benötigt, damit die Nodes sich gegenseitig im System verbinden können. Der Nachrichten Transport geht nicht direkt über den Master er verbindet nur die beiden Nodes, welche anschließend via einer Peer-to-Peer Verbindung kommunizieren.

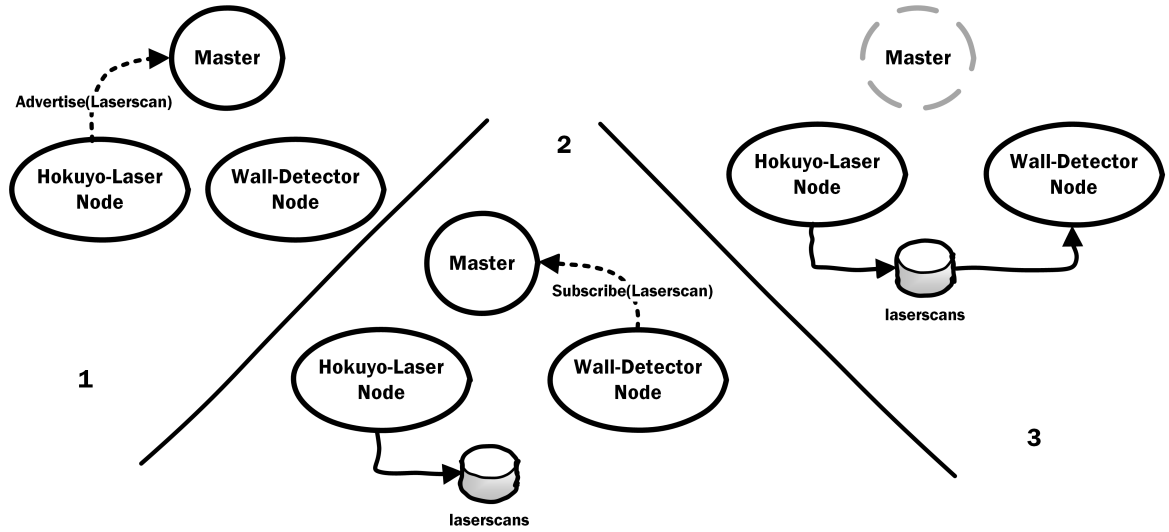


Abbildung 4: Master Service: (1) Der Hokuyo Laser Node meldet sich beim Master an, das er Laserscans zur Verfügung stellt. (2) Der Laser Node stellt seine Daten zur Verfügung und der Wall Detector meldet sich beim Master, dass er Laserscans benötigt. Der Master stellt die Verbingunsinformationen bereit und zieht sich dann zurück (graue Markierung). (3) Die Verbindung zwischen Laser Node und Detector Node ist hergestellt.

Dieses Verfahren reduziert die Netzwerklast stark im Vergleich zu einem Broadcasting-Verfahren. Tests zeigten, dass ein Kamera-Node (openni) ein Netzwerk mit seinen 3D-Punktwolken vollständig auslastet. Dabei wurde die Punktwolke auf der physikalischen Maschine eins erzeugt und auf eine zweite Maschine via WLAN übertragen. Die Delay-Zeit der Wolken war dabei sehr hoch > 1 Sekunde und wurde, je länger die Übertragung dauerte höher. Werden die Daten jedoch auf der Maschine vor verarbeitet und nur noch die wichtigen Daten übertragen, sowie die Übertragungsrate reduziert, ist die Übertragung der Daten unkritisch. Nun gehen nur

noch die Verbindungsdaten zwischen Nodes und Master über das Netzwerk und die reduzierte Datenmenge.

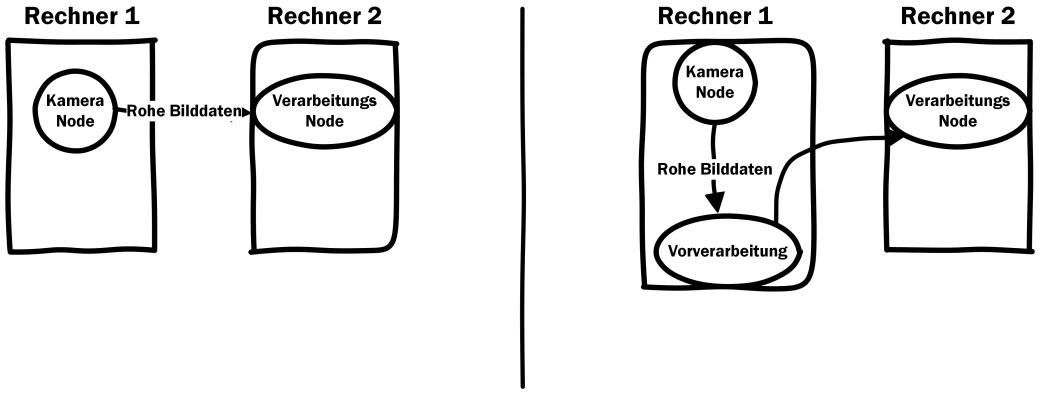


Abbildung 5: Netzwerk Entlastung durch Peer-to-Peer Verbindungen. Links: Ohne Datenvorverarbeitung, Rechts: Mit Datenvorverarbeitung

Neben dem Namen-Service verwaltet der ROS Master auch den Parameter Server. Dieser besteht aus einem Dictonary, einer großen Key-Value-Map, welches global erreicht werden können. Nodes haben so die Möglichkeit vordefinierte Daten ab zugreifen oder selber welche zu schreiben. So können Sensoren oder Aktoren kalibriert und konfiguriert werden. Der Zugriff auf diese Daten hat keine hohe Performance. Er ist für einen statischen Zugriff gedacht und kann mit einer Baum-Struktur geordnet werden, so lassen sich zusammenhängende Daten mit einer Abfrage am Server abrufen.

2.2.3 Topics und Messages

ROS bietet für die Kommunikation zwischen Nodes ein *Topic*-System. Jeder Topic entspricht dabei einem benannten Bus, auf welchen anonyme *Publisher* und *Subscriber* zugreifen können. Die Daten können dabei als TCP-Pakete (TCPROS) oder als UDP-Pakete(UDPROS) gesendet werden. Ein Topic besitzt einen eindeutigen Namen im System und einen eindeutigen Typen, der anhand des übersendeten Message-Typen definiert wird. Jede Topic kann nur einen Message-Typen versenden. Dieser Type ist nicht im Master bekannt, jedoch können sich Subscriber nur mit dem entsprechenden Typen anbinden. Die Anzahl der Subscriber und Publisher für einen Topic ist nur durch die Systemleistung beschränkt.

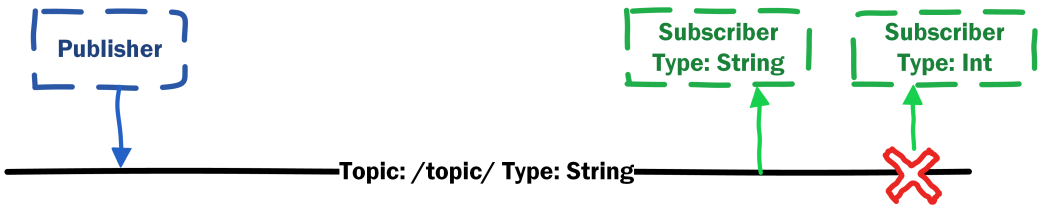


Abbildung 6: Beispiel für eine Topic mit dem Namen /topic und dem Type String. Ein Publisher schreibt Daten in den Bus, ein Subscriber liest diese aus. Die Verbindung eines zweiten Subscribers schlägt auf Grund des falschen Types fehl.

Publisher veröffentlichen ihre Daten in den Bus und haben nur einen schreibenden Zugriff. Sie

bekommen kein Feedback an wen die Daten gesendet worden sind oder ob die Daten überhaupt empfangen worden sind. Der erste Publisher, der sich auf einem Topic verbindet gibt den Typen vor.

Subscriber lesen die Daten vom Bus und geben sie für die Weiterverarbeitung weiter. Sie bekommen normalerweise keine Informationen über den Absender der Daten. Subscriber können sich nur bei einem Topic registrieren, wenn der Message-Type stimmt.

Subscriber und Publisher haben jeweils eigene Cache-Systeme die Nachrichten vor- oder zurückhalten können. Ein Topic ist ein einfachgerichteter Streaming-Dienst ist. Da in einem Roboter-System auch Remote Procedure Calls benötigt werden, um zum Beispiel Antworten auf Anfragen zu erhalten oder synchronisierte Aufgaben zu erledigen, wurde das Service-Paket als weiteres Kernkonzept angelegt.

2.2.4 Services

Services ermöglichen es einen Remote Procedure Call Node übergreifend durchzuführen. Dazu muss Node A seinen Service am Master registriert haben und einen Service Server gestartet haben. Node B kann nun mit einem Service Client auf den registrierten Service zugreifen und eine Anfragen starten. Während die Anfrage verarbeitet wird, kann Node B nun je nach Anforderung blockieren oder weiter arbeiten. Node A verarbeitet nun die Anfrage und antwortet Node B mit dem Ergebnis. Der Service selbst, sowie die Parameter und die Rückgabewerte, sind für jeden Service eindeutig und werden vorher via zwei Messages, eine für die Request und eine für die Response, festgelegt. Analog zu den Topics werden Services am Master mit einem eindeutigen Namen angemeldet unter dem sie erreichbar sind. Dies ermöglicht auch einen einfachen Austausch von verschiedenen Implementierungen.

2.2.5 Action-Lib

Ein aktiver Service schickt erst eine Rückmeldung an den Aufrufer, wenn er seine Aufgabe erledigt hat. Um auch einen Zwischenstand schicken zu können wurde die Actionlib eingeführt. Dieses Paket ist eigentlich keine Entwicklung der ROS-Entwickler sondern ein Community-Package. Inzwischen wurde es aber von der OSR-Foundation übernommen und als eins der Kernkonzepte eingestuft.

Analog zu den Services besitzt auch die Action-Lib einen Server- und einen Client Teil. Der Client fordert vom Server eine Aktion an, dabei übergibt er ein *Goal*. Dieses beinhaltet die benötigten Parameter für den Server. Der Server arbeitet die Anfrage ab, dabei kann er auch *Feedback*, Zwischenstände, zurückschicken. Ist die Anfrage abgearbeitet schickt der Server ein *Result* an den Client zurück. Wie auch beim Service ist die Art der Aktion eindeutig. Die Anzahl der Parameter bei Goal, Feedback und Result sind fest definiert und können zur Laufzeit nicht geändert werden.

2.3 Kinematik

Dieses Unterkapitel befasst sich mit dem Begriff der Kinematik und den dazu gehörigen Begriffen des Mechanismus und der inversen Kinematik. Außerdem werden einige mathematische Lösungswege für verschiedene Probleme innerhalb der genannten Themen genannt.

2.3.1 Mechanismus

Ein Mechanismus ist der Grundbegriff von beweglichen Körper. Dabei wird zwischen *Gliedern* (engl. Link, $L =$ Set of Links) und *Gelenken* (engl. Joint, $J =$ Set of Joints) unterschieden. Glieder sind die starren Körper eines Mechanismus und sind durch Gelenke miteinander verbunden. Ein Glied ist nicht auf ein Gelenk beschränkt, sondern kann mehrere haben ($N_{joint} \in \mathbb{N}_0$). Auch die Verbindung zwischen zwei Gliedern kann durch mehrere Gelenke realisiert sein. Ein Gelenk wiederum ist auf genau zwei Glieder beschränkt ($N_{link} = 0$) und ermöglicht so eine eingeschränkte relative Bewegung zueinander. Gelenke werden nicht als eigenständige physikalische Körper gesehen. Gelenkhälften, zum Beispiel die Schenkel eines Kippscharniers, werden als Bestandteil des damit verbundenen Glied betrachtet.

Glieder können mit zwei Parametern definiert werden, der Länge a und der Verdrehung α:

$$l_i \in L$$

$$a_i = \text{Länge von } l_i \tag{1}$$

$$\alpha_i = \text{Verdrehung von } l_i \tag{2}$$

Gelenke werden ebenfalls mit 2 Parametern definiert. Der Gelenk-Offset beschreibt den Abstand zwischen den zwei Gliedern eines Gelenks auf der Gelenkachse. Der Gelenkwinkel beschreibt die Rotation zwischen den beiden verbundenen Glieder im Bezug zur Gelenkachse. Corke [2011]

$$j_i \in J$$

$$d_i = \text{Gelenk-Offset von } j_i \tag{3}$$

$$\theta_i = \text{Gelenkwinkel von } j_i \tag{4}$$

Im Anschluss werden nur noch die englischen Begriffe Link und Joint benutzt.

2.3.2 Kinematik

Die *Kinematik* eines Mechanismus beschreibt die Bewegungsmöglichkeiten der Links relativ zueinander. Es wird dabei abstrahiert, ob ein Joint motorisch angetrieben oder passiv bewegt wird. Die Kinematik betrachtet bei der Bewegung auftretende *Geschwindigkeiten* und *Beschleunigungen*. Auftretende Kräfte werden nicht in der Kinematik, sondern in der *Dynamik* untersucht. Zentrale Aspekte dabei sind die Trägheits- und Schwerkraft.

Die *Kinematische Kette* beschreibt den Aufbau eines speziellen Mechanismus. Dabei ist die Anzahl der Joints pro Link auf maximal 2 beschränkt ($N_{joint} <= 2$). Besitzt jeder Link genau zwei Joints gilt die Kette als geschlossen, ansonsten als offen.

$$\text{Kinematische Kette} = \begin{cases} geschlossen & \forall l \in L | l_{joints} = 2 \\ offen & \text{für Rest} \end{cases} \tag{5}$$

Für diese Arbeit ist nur die offene Kette interessant, da die Manipulatoren der meisten Roboter an beiden Enden (Base und End-Effektor) frei sind. Dadurch ergibt sich für $N_{link} = N_{joint} + 1$. Typischerweise wird der erste Joint mit dem Index 1 versehen und der erste Link mit dem Index 0. Link 0 ist die *Base* des Manipulators und Link N beinhaltet den *End-Effektor*. Durch die Bezeichnung ergibt sich, dass Joint j die Links j-1 und j mit einander verbindet und den Link j bewegt. Corke [2011]

Die Transformation eines Link-Koordinatensystems j-1 zu Link j wird doch elementare Rotation und Translation beschrieben:

$$^{j-1}A_j(\theta_j, d_j, a_j, \alpha_j) = T_{Rz}(\theta_j)T_z(d_j)T_x(a_j)T_{Rx}(\alpha_j) \tag{6}$$

Durch die Vereinigung der einzelnen Matrizen ergibt sich:

$$^{j-1}A_j(\theta_j, d_j, a_j, \alpha_j) = \begin{pmatrix} cos\theta_j & -sin\theta_j cos\alpha_j & sin\theta_j sin\alpha_j & a_j cos\theta_j \\ sin\theta_j & cos\theta_j cos\alpha_j & -cos\theta_j sin\alpha_j & a_j sin\theta_j \\ 0 & sin\alpha_j & cos\alpha_j & d_j \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{7}$$

2.3.3 Forward Kinematics

Die Vorwärts-Kinematik (engl. Forward Kinematic) beschreibt die "vorwärts"Rechnung der Kinematik. Aus den gegebenen Joint-Informationen wird die Pose des End-Effectors berechnet. Hier reichen die Daten für den Gelenkwinkel θ_j bei Drehgelenken und über den Winkel-Offset d_j bei Schiebe-/ Schubgelenken, da die restlichen Werte für den Mechanismus als konstant angesehen werden können. Die Forward Kinematic wird häufig als Funktion K angegeben Corke [2011]

$$\xi_E = K(\mathbf{q}) \tag{8}$$

q entspricht dabei der aktuellen Joint-Konfiguration und ξ_E der Pose des End-Effectors. Durch die Kombination der Transformation aus Gleichung 6 ergibt sich für einen Manipulator mit N-JointsCorke [2011]

$$\xi_E \sim {}^0T_E = {}^0A_1\,{}^1A_2\,\ldots\,{}^{N-1}A_N \tag{9}$$

2.3.4 Inverse Kinematik

Die Inverse Kinematik berechnet die Inverse der Forward-Kinematic. Sie bestimmt die möglichen Joint-Konfigurationen $\mathbf{q}$ für eine Zielpose. Im Gegensatz zur Forward-Kinematic ist die Inverse Kinematik nicht eindeutig. Eine einfache Veranschaulichung kann man am menschlichen Körper sehen. Fixiert man das Handgelenk an einer Position kann man den Ellbogen in verschiedene Positionen bewegen ohne die Position der Schulter zu ändern. Für die Inverse Kinematik ergibt sich folgende FunktionCorke [2011]

$$\mathbf{q} = K^{-1}(\xi) \tag{10}$$

Aus Gleichung 8 und 10 ergibt sich auf Grund der Eindeutigkeit:

$$\xi = K(K^{-1}(\xi))$$

Aber **nicht**

$$\mathbf{q} = K^{-1}(K(\mathbf{q}))$$

Um eine Inverse Kinematik zu lösen gibt es drei Methoden: geometrisch (analytisch), numerisch und heuristisch.Steidl [2011]

Die geometrische Lösung nutzt die einfache geometrische Abstraktion des Mechanismus und versucht diesen mit trigonometrischen Funktionen abzubilden. Damit auch hier mehrere Lösungen bestimmt werden können, werden verschiedene Konfigurationen für bestimmte Abstraktionsmodelle geschaffen und alle Konfigurationen berechnet. Am Beispiel des Menschen wären mögliche Konfigurationen: Ellbogen oben oder Ellbogen unten. Bei der Verwendung der trigonometrischen Funktionen muss beachtet werden, dass die Umkehrfunktionen in bestimmten Bereichen nicht eindeutig, ungenau oder gar nicht definiert sind. Empfohlen wird dafür die Verwendung von atan2, der in den meisten Programmiersprachen vorhanden ist. Diese Methodik funktioniert nur für einfache Mechanismen bei denen möglichst alle Joints die selben Achsen haben.

Die numerische Lösung versucht durch Iterationen eine Näherung an die gewünschte End-Pose zu erreichen. Das ganze fällt in die Thematik der Optimierung. Für die numerische Lösung gibt es unterschiedliche Algorithmen. Dazu gehören Jacobian Transpose, Pseudo Inverse und Damped Least Square. Die numerischen Lösungen sind im Vergleich zur analytischen langsamer und ungenauer, jedoch lassen sich mit ihr größere und komplexere Manipulatoren berechnen.Steidl [2011]

Die heuristischen Lösungen nutzen abstrakte Bezüge und Vorhersagen für Veränderungen der End-Pose im Bezug zur Joint-Konfiguration um mögliche Konfigurationen zu bestimmen. Eine anschließende Güteberechnung (zum Beispiel euklidische Distanz) gibt an, ob eine Konfiguration verworfen oder angenommen wird. Wie für die numerische Lösung gibt es auch bei der heuristischen Lösung mehrere Algorithmen. Diese sind unter anderem Cyclic Coordinate Descent und Lagrange-Multiplier.Steidl [2011]

2.4 Aufbau

Dieses Kapitel befasst sich mit dem Aufbau der Versuchsanlage und der verwendeten Hardware. Dabei werden die Aspekte des räumlichen Aufbau genauso betrachtet wie die Netzwerkinfrastruktur und die genutzten Roboter und Sensoren.

2.4.1 Aufbau Teststand

2.4.2 YouBot

In dieser Arbeit werden zwei Roboter genutzt. Bei beiden handelt es sich um YouBots der Firma Kuka. Diese wurden 2010 erstmals auf der Automatica in München vorgestellt. Kuka ist ein deutscher Hersteller von Industrierobotern mit Sitz in Augsburg.Kuka hat aber auch Erfahrung im Bau von experimentellen Robotern wie etwas die Arme für Justin, einem Service-Roboter.

Die beiden YouBots unterscheiden sich beim Aufbau. YouBot 1 besteht aus einem Manipulator mit Gripper, YouBot 2 besteht aus Manipulator mit Gripper und mobiler Plattform mit omnidirektionalen Rollen. Zur besseren Unterscheidung und zur Lesbarkeit dieser Arbeit haben beide Roboter einen Namen bekommen. YouBot 1 entspricht dabei **Dummy**, YouBot 2 ist **Rose**. Im Weiteren Verlauf der Arbeit werden nur noch diese Namen genutzt um eine Verwechslung zu vermeiden.

Die Roboterarme von Dummy und Rose sind eine offene kinematische Kette mit fünf Joints, die von der Basis aufsteigend numeriert sind. An Joint 5 sind die Gripper montiert, diese bestehen aus zwei individuellen Fingern, die linear auf einer Achse bewegt werden können.

Joint 1 und Joint 5 sind zwei Drehgelenke um die Z-Achse, die bei ausgestreckter Haltung (Candle-Pose) parallel liegen. Joint 2, 3 und 4 sind Kippgelenke um die X-Achse, die immer parallel liegen. Durch diese Konstellation ergeben sich für den kompletten Arm fünf Freiheitsgrade (5-DOF). Durch die Einschränkung der Achsen auf die X- und Z-Achsen der einzelnen Joints ist ein seitlicher Griff bei $x_{base} = 0$ oder $y_{base} = 0$ nicht möglich. Die Höhe in Candle-Pose beträgt inklusiv Gripper 655mm. Die einzelnen Abstände lassen sich der Grafik 7 entnehmen.

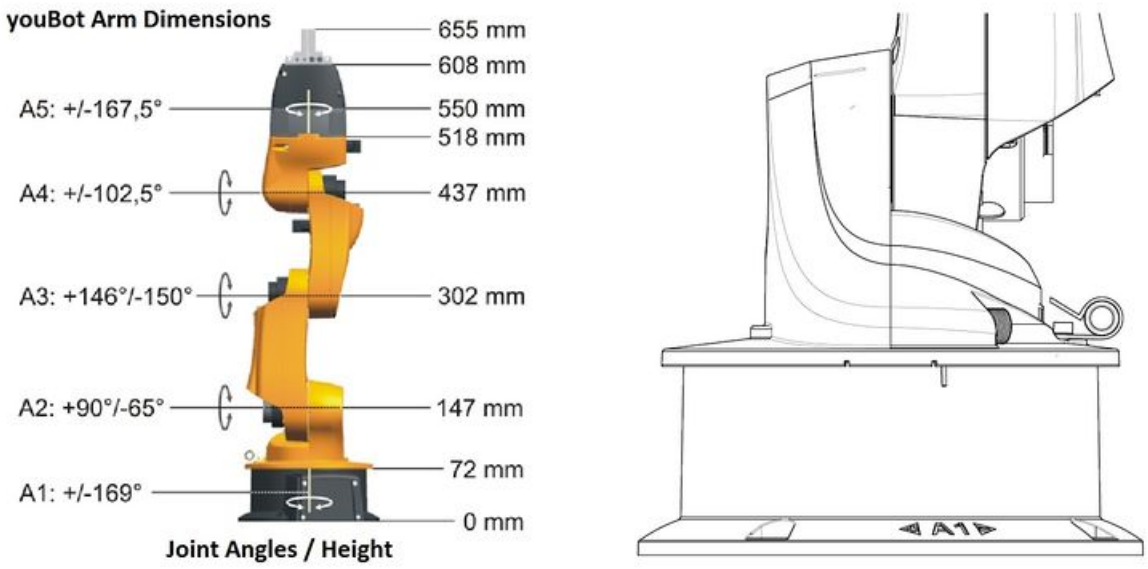


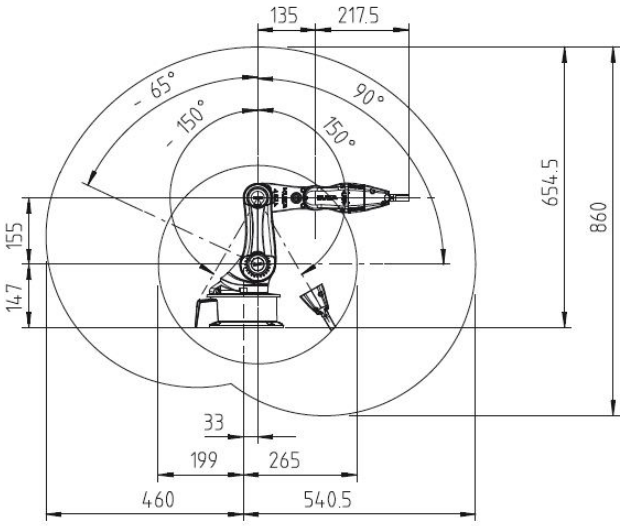
Abbildung 7: YouBot Arm Kinematik. Links: Detailierter Mechanismus mit Joint und Link Angaben. Rechts: Vergrößerte Darstellung der Basis. Bildquelle: Florek-Jasinska [2015]

Tabelle 2 stellt noch einmal den Drehbereich aller Joints dar. Dabei sind neben Bezeichner auch die minimalen und maximalen Winkel, sowie die Winkelgeschwindigkeiten. Eine Besonderheit stellt Joint 3 dar, durch den Aufbau bedingt wurde die Achse innerhalb der Treiber gespiegelt, sodass die Winkel um den 0° Punkt gespiegelt wurden. In der Tabelle und den folgenden Zeichnungen ist diese Spiegelung nicht beachtet.

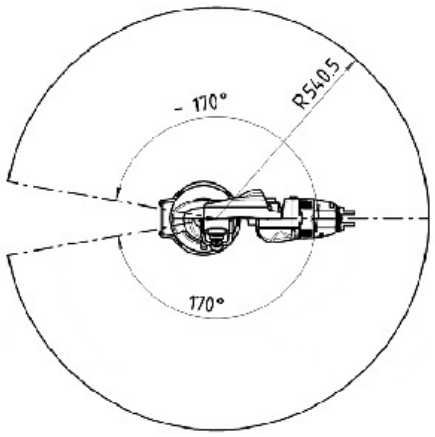
Bezeichnung	Max./Min. Winkel	Winkelgeschwindigkeit (rad / s)
Joint 1	$\pm$ 2.94	$\frac{\pi}{2}$
Joint 2	$+\frac{\pi}{2}$ / -1.13	$\frac{\pi}{2}$
Joint 3	+2.54 /-2.63	$\frac{\pi}{2}$
Joint 4	$\pm$1.78	$\frac{\pi}{2}$
Joint 5	$\pm$2.91	$\frac{\pi}{2}$

Tabelle 2: YouBot Arm Joints. Quelle: Florek-Jasinska [2015]

Der Arbeitsraum des YouBot Arms beschränkt sich durch die Joints. Die Grafiken 8(a) und 8(b) stellen den Arbeitsbereich eingeschränkt dar. Die Darstellung 8(a) zeigt mit Hilfe von Konturen die einzelnen Bahnen unter Beschränkung der Joints, sowie der End-Effektor Ausrichtung. Die äußerste Kontur gibt die maximale Reichweite für den End-Effektor an. Die Z-Achse steht dabei orthogonal zur Tangente an der Konturposition. Abbildung 8(b) gibt eine Draufsicht auf den Arbeitsraum. Durch die Beschränkung von Joint 1 befindet sich ein toter Winkel im "Rücken"der Arms. Dieser kann durch einen Überschlag des ganzen Arm, insbesondere Joint 2 und 3, und einer 180 °Drehung von Joint 1 dennoch erreicht werden. Dies wird durch den linken Bereich in Abbildung 8(a) klar.



(a) YouBot Arm Arbeitsraum. Reichweite der einzelnen Gelenk und einzelne Abmessung der Links und Link-Offsets. Einheitslose Angaben sind in mm.



(b) Draufsicht YouBot Arm Arbeitsraum.

Abbildung 8: Arbeitsraum YouBot. Bilderquelle:Florek-Jasinska [2015]

Rose besitzt neben dem Arm noch eine mobile Plattform. Diese ist 590 mm lang, 380 mm breit und 140 mm hoch. Die Plattform hat vier Räder mit einem Durchmesser von 47.5 mm. Jedes Rad lässt sich getrennt ansteuern. Bei den Rädern handelt es sich um Mecanum-Räder, die eine Steuerung in alle Richtungen ermöglicht. Dazu sitzen auf jeder Felge sechs drehbare tonnenförmige Rollen die im Winkel von 45° zur Achse des Rades angebracht sind. Damit Omni-Direktionale Bewegungen möglich sind werden die Räder abwechselnd zum Nachbar auf +45° oder -45° angeordnet. Die minimale Geschwindigkeit der Plattform beträgt $0.01\frac{m}{s}$, die maximale $0.8\frac{m}{s}$. Abbildung 9 zeigt den Aufbau und Abmessungen der Plattform.

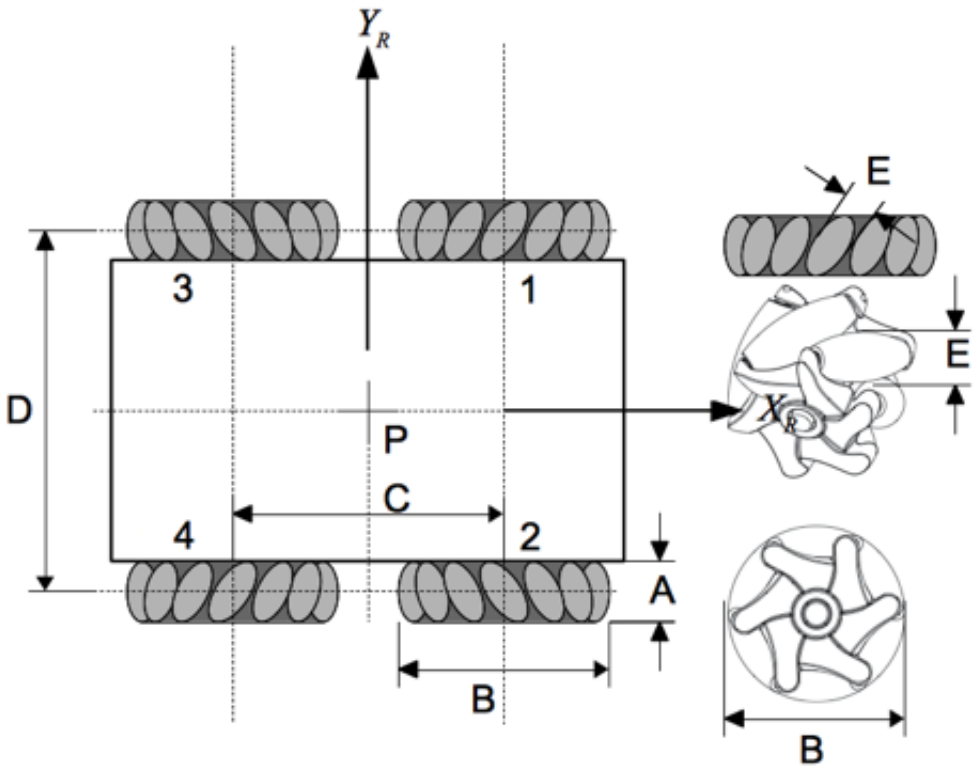
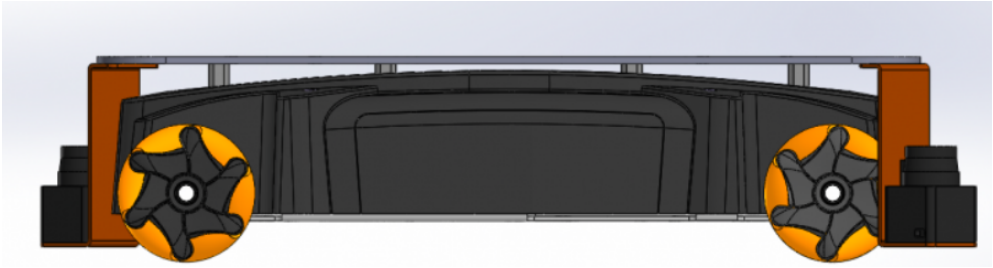
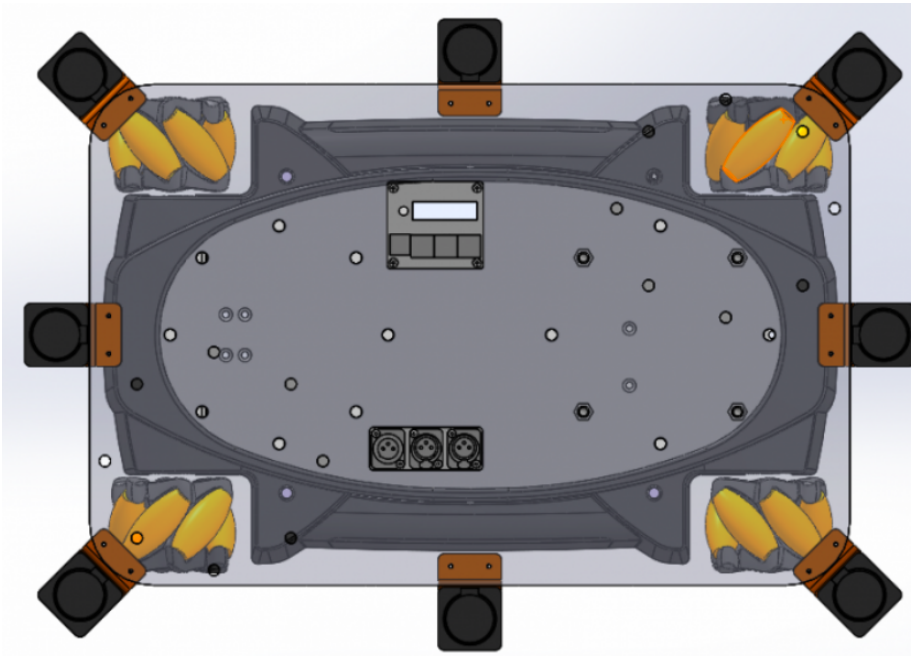


Abbildung 9: YouBot Base. A = 74.87 mm, B = 100 mm, C = 471 mm, D = 300.46 mm E = 28 mm. Die Bodenfreiheit der Plattform beträgt 20 mm. Bildquelle: Florek-Jasinska [2015]

Die Plattform von Rose ist modifiziert und besitzt eine Aufsatzplatte. Diese dient zur Montage zusätzlicher Sensoren und zur Schutz der Räder bei Kollisionen. Diese ist 600 mm lang und 396 mm breit. Durch die Abstandsbolzen und die Dicke der Platte (5 mm) erhöht sich die Plattform auf 150 mm (siehe Abbildung 10(a)).



(a) Seitenansicht mobile Plattform mit Sensorplatte



(b) Draufsicht mobile Plattform mit Sensorplatte

Abbildung 10: Mobile YouBot Plattform mit Sensorplatte. Bilderquelle:Kuka [2015]

Für die Rechenleistung der beiden Roboter sorgen neben den verbauten, eingebetteten Boards zwei Computer mit Linux Betriebssystemen. Der für Rose zuständige PC ist ein Mini-ITX und in der mobilen Plattform eingebaut. Dieser läuft mit einem Intel Atom D510 Dual Core mit 1.66 GHz als CPU und 2 GB DDR2 RAM. Als Festplattensystem ist eine 32 GB SSD verbaut. Der Rechner stellt einige IO-Schnittstellen zur Verfügung: sechs USB 2.0, ein VGA und zwei LAN Anschlüsse sind nutzbar. Dabei ist ein LAN Anschluss für den Ethercat-Anschluss der Plattform belegt. Diese wiederum bringt nochmal zwei LAN Anschlüsse mit. Der Rechner für Dummy ist ein BLALBLA

Weitere Details zu den Armen, der Plattform oder den Rechnern befindet sich im Anhang.

2.4.3 Sensoren

Für die Erkennung von Objekten und der Positionierung im Raum sind mehrere Sensoren nötig. In dieser Arbeit wird dabei auf das Kamerasystem Asus XTion Pro Live und auf einen Laserscanner Hokuyo URG-04LX-UG01 zurückgegriffen.

Die XTion Pro ist ein Tiefenkamerasystem von Asus. Es besteht aus einer RGB-Kamera, einer Tiefen-Kamera und zwei Mikrophonen. Die Kamera wurde von Prime Sense entwickelt und ist eine umgelabelte Microsoft Kinect. Die XTion ist kleiner und leichter als die Kinect und damit besser für die Robotik geeignet. Angeschlossen wird die XTion mit einem USB Kabel und betrieben mit dem ROS Paket OpenNI2. Das Sichtfeld der Kamera beträgt Horizontal 58°, 45°Vertikal und 70°Diagonal. Die nutzbare Distanz der Tiefenkamera beträgt zwischen 800 mm und 3500 mm. Die Auflösung der Tiefenkamera beträgt bei 30 Frames pro Sekunde 640x480 und bei 60 FPS 320x240. Die Auflösung des RGB-Bildes 1280x1024.

2.4.4 Netzwerk

3 Stand der Technik

3.1 Physically Embedded Intelligent Systems - PEIS

Andere beschäftigen sich grad mit ...

3.2 Multirobots System - Komposition und Konfiguration

Andere beschäftigen sich grad mit ...

3.3 Handover - Übergabe zwischen Roboter-Maschine und Roboter-Roboter

Andere beschäftigen sich grad mit ...

4 Zusammenfassung und Ausblick

Hier eine Zusammenfassung und der Ausblick

5 Danksagung

Ich bin allen Leuten furchtbar dankbar, vor allem den Jungs bei Google.

6 Literaturverzeichnis

Literatur

weitere B. Badura. *Fehlzeiten-Report 2004 Gesundheitsmanagement in Krankenhäusern und Pflegeeinrichtungen*. Badura, Prof. Dr. Bernhard Schellschmidt, Dr. Henner Vetter, Christian, 2005. ISBN 978-3-540-27051-5.

Peter Corke. *Robotics, Vision and Control - Fundamental Algorithms in MATLAB*. Springer Science & Business Media, Berlin Heidelberg, 1st ed. 2011 edition, 2011. ISBN 978-3-642-20143-1.

Monika Florek-Jasinska. Youbot detailed specifications, dec 2015. URL `http://www.youbot-store.com/wiki/index.php/YouBot_Detailed_Specifications`.

Hisato Kobayashi. *Technolife Series. Robotto wa tomodachi da! [Technolife-Serie. Der Roboter ist ein Freund!]*. Ohmsha, Tokyo, 1999.

Kuka. Mounting and sensor plate, dec 2015. URL `http://www.youbot-store.com/accessories/youbot-extensions/mounting-and-sensor-plate`.

Dr. Sibylle Meyer. *Mein Freund der Roboter*. BMBF/VDE Innovationspartnerschaft AAL, oct 2011. ISBN-10: 3800733420.

Panasonic, may 2005. URL `http://panasonic.com.jp/museum/smilennium/kaden/robot/38/01.html`.

Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5, 2009.

Daniela Steidl, 2011. URL `https://wwwcg.in.tum.de/fileadmin/user_upload/Lehrstuehle/Lehrstuhl_XV/Teaching/SS11/Proseminar-PIXAR/Daniela_Steidl_talk.pdf`.

Cosima Wagner. „Tele-Altenpflege“ und „Robotertherapie “: Leben mit Robotern als Vision und Realität für die alternde Gesellschaft Japans, 2009.

Bruno Zandonella. Bevölkerungsentwicklung und Renten. oct 2013.

A Erster Anhang

blah blah blah

B Noch ein Anhang

spannende Sache, oder?