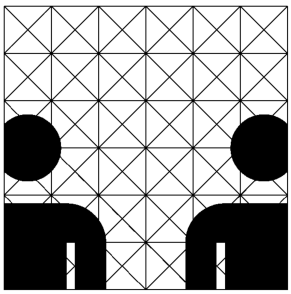


Universität Hamburg
Department Informatik

Technische Aspekte Multimodaler Systeme (TAMS)
Diplomarbeit

Praktische Realisierung einer haptischen Telerobotik-Steuerung für eine interaktive Nutzung

Mai 2009

Jan Bruder JanBruder@web.de

betreut von
Prof. Dr. Jianwei Zhang
Dr. Peer Stelldinger

Abstract

**Implementation of a
Haptic Telerobot Control System
for an Interactive Use**

This diploma thesis documents the implemantation of a new telemanipualation architecture. Different topics of research are covered, like UDP/IP data transmission over a wireless local area network and the use of haptic devices in C++ and Java. A method for a permanent solving of the inverse kinematics with applicable intermediary results is described. The system is integrated into a given service robot environment using a six degree of freedom manipulator. Aspects of transparency and stability are analysed with experimental results. Software design issues of this distributed system with client and server structure are discussed.

Zusammenfassung

**Praktische Realisierung einer
haptischen Telerobotik-Steuerung
für eine interaktive Nutzung**

In dieser Diplomarbeit ist die Implementation einer neuen Telemanipulations-Architektur dokumentiert. Verschiedene aktuelle Forschungsthemen der Telerobotik werden behandelt, wie die UDP/IP-Datenübertragung über Wireless LAN und die Nutzung von haptischen Geräten in C++ und Java. Ein Verfahren zur permanenten Lösung der inversen Kinematik mit verwendbaren Zwischenergebnissen ist beschrieben. Das System wird in eine bestehende Serviceroboter-Umgebung, unter der Verwendung eines Manipulatorarms mit sechs Freiheitsgraden, integriert. Aspekte der Transparenz und Stabilität werden dabei anhand von praktischen Experimenten untersucht. Softwaredesignfragen dieses verteilten Systems mit Client/Server-Struktur werden diskutiert.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Die Robotik ist für Computer-Programmierer ein sehr spannendes Arbeitsfeld. In diesem Bereich können Programme kreiert werden, die die reale Welt manipulieren, ohne dass dabei Menschen agieren müssen. Diese faszinierende Eigenschaft lässt die Robotik auf den ersten Blick sehr interessant erscheinen. Auf den zweiten Blick muss man jedoch feststellen, dass die praktische Umsetzung selbstständig oder intelligent agierender Roboter noch nicht sehr weit fortgeschritten ist. Selbst einfach erscheinende interaktive Aufgaben bedeuten einen großen Entwicklungsaufwand. Und oft ist es erheblich einfacher, die von einem Roboter geforderten Tätigkeiten von einem Menschen ausführen zu lassen. In einigen Bereichen hat sich die Benutzung von Robotern jedoch trotzdem durchgesetzt. So sind Roboter heutzutage in Fertigungsanlagen der Industrie und in Gefahrenbereichen, wie der Nuklear-Industrie und der Tiefsee-Forschung, nicht mehr weg zudenken. In genau diesen Bereichen beschränkt sich die Robotertechnik jedoch auf ein sehr geringes Maß an Autonomie. Heutige Roboter werden meistens für feste Bewegungsabläufe programmiert. Nur in Außnahmefällen werden Roboter gebraucht, die wirklich intelligent mit ihrer veränderlichen Umwelt interagieren müssen. Doch gerade in diesen Fällen kann kein völlig autonom handelnder Roboter eingesetzt werden, da diese Technik noch nicht weit genug entwickelt ist. Meistens würde ein nicht Gelingen der zu erfüllenden Aufgabe hohe Kosten produzieren oder einen unerwünscht großen Schaden anrichten.

Abbildung 1.1: Ein früher Telemanipulator in der Nuklear-Forschung.

Ein hierfür oft akzeptabler Kompromiss zwischen Robotik und menschlicher Arbeitskraft ist die Verwendung von Telerobotik. Dabei wird in Bereichen, die für Menschen zu gefährlich oder unzugänglich sind, ein nicht autonomer Roboter eingesetzt. Die

Aktionen dieses Teleroboters werden vollständig von einem menschlichen Benutzer gesteuert. Unterschiedliche Systeme dieser Art sind bereits erfolgreich eingesetzt worden. Es gibt zum Beispiel Telemanipulator-Arme in Laboratorien und ferngesteuerte Fahrzeuge mit Roboterarmen für die Erkundung der Tiefsee, so genannte Remote Operated Vehicle (ROVs). Von Vorteil ist bei diesen Systemen, dass ein Mensch jederzeit die Kontrolle über das System hat. Bei unvorhergesehenen Begebenheiten kann er sinnvoll entscheiden, was zu tun ist. Nachteilig ist, das mehrere Faktoren die Kontrolle eines solchen Systems sehr schwierig machen. Die ersten Telerobotiksysteme hat es zwar schon in den 1950er Jahren gegeben, doch viele der benötigten Technologie-Bereiche sind erst seit den 1990er Jahren ausgereift genug gewesen. Fragen nach der Qualität von Steuerungssystemen, der Datenübertragung und den Ein- und Ausgabeschnittstellen zu dem Benutzer sind bis heute Forschungsschwerpunkte der Robotik.

1.1 Motivation

Greift ein Mensch einen Gegenstand, der auf einem Tisch direkt vor ihm liegt, dann benutzt er dabei seinen visuellen Sinn und den Tastsinn. Eine derartige Aufgabe kann er jedoch meistens auch mit nur einem der beiden Sinne bewältigen. Mit verbundenen Augen, wie auch mit sehr harten Handschuhen, die die Berührungsempfindung einschränken, ist die komplette Aufgabe immer noch durchführbar. In diesen beiden eingeschränkten Fällen ist das Greifen jedoch nicht so schnell und präzise, wie ein Greifen mit kombinierter visueller und haptischer Unterstützung. Ohne die visuellen Reize dauert die anfängliche grobe Lokalisierung eines Gegenstandes länger. Nach der Lokalisierung ist der Tastsinn wichtiger, um die Position des Gegenstandes genauer bestimmen zu können. Offensichtlich spielen also beide Sinne eine wichtige Rolle bei dieser alltäglichen Benutzung von Arm und Hand.

In der Robotik wird diese Feststellung relevant, wenn ein Roboterarm Gegenstände greifen soll. Die einfachste Robotik-Umgebung, in der man dieser Aufgabe ständig begegnet, ist ein Telemanipulations-System. Dabei handelt es sich allgemein um ein Werkzeug, mit dem ein Mensch Gegenstände physisch manipulieren kann, ohne sie mit den eigenen Händen zu berühren. Häufig sind solche Systeme in Laboren anzutreffen, in denen mit Materialien gearbeitet wird, die nicht mit den eigenen Händen bearbeitet werden können. Dazu zählen zum Beispiel giftige oder radioaktive Stoffe, deren Nähe schädlich für Menschen ist. Eine ausreichend sicherer Entfernung oder Abschirmung ist daher nötig. Auch schwer erreichbare Arbeitsplätze, wie die Tiefsee oder die Oberfläche anderer Planeten, sind sinnvolle Einsatzgebiete für Telemanipulatoren.

Ein Telemanipulator sollte dabei idealer Weise dem Menschen nachempfunden sein, um die Benutzung so intuitiv wie möglich zu gestalten. Denn eine komplizierte Steuerung erhöht das Risiko von Bedienungsfehlern, die in risikobehafteten Arbeitsfeldern sehr problematisch sind. Ein typischer Telemanipulator in Form eines Roboterarmes sollte also im günstigsten Fall auch durch den Arm eines Menschen gesteuert werden und dessen Bewegungen so gut wie möglich imitieren können. Der Mensch kann den Roboterarm und sein Arbeitsumfeld dabei meistens über einen Videobildschirm oder direkt durch ein Sichtfenster beobachten.

Ein solches System ermöglicht dem menschlichen Benutzer ein interaktives Arbeiten mit der Umgebung des Telemanipulators. Aus den vorangegangenen Überlegungen erscheint es sinnvoll, den Tastsinn bei der interaktiven Steuerung eines Roboterarmes mit einzubeziehen. Dieses folgt auch einer allgemeineren Überlegung, dass die qualitative Benutzbarkeit des Telemanipulators erhöht wird, je mehr unterstützende Blickwinkel und zusätzliche Sinnesreize dem Benutzer geboten werden, die er intuitiv nutzen kann. Dieses kann vermutlich bei dem Benutzer das Erfahren der eigenen Telepräsenz positiv unterstützen. Und darüber hinaus kann es wahrscheinlich eine genauere Steuerung ermöglichen. Mit Einbeziehung des Tastsinnes ist hierbei gemeint, taktile und kinästhetische Informationen an den menschlichen Benutzer zu übermitteln. Dieses ist mit der Verwendung geeigneter Ein- und Ausgabegeräte, so genannte haptische Geräte, in unterschiedlich starkem Ausmaß möglich.

Im Arbeitsbereich TAMS des Fachbereiches Informatik der Universität Hamburg sind ein mobiler Service-Roboter und ein haptisches Eingabegerät in Benutzung. Auf die Bedeutung und die Details dieser vorhandenen Komponenten wird in den Kapiteln 2 und 4 eingegangen. Hier soll zunächst nur darauf hingewiesen werden, dass bisher kein System zur Verfügung steht, um diese beide Komponenten interaktiv miteinander zu benutzen. Der Service-Roboter vereint viele Komponenten, die ihn für diverse denkbare Forschungs- und Lehre-Projekte im Bereich der Robotik attraktiv macht. Er ist mobil, hat verschiedene Sensor-Systeme, besitzt zwei Arme mit Greifern und kann durch sein Linux-Betriebssystem vielseitig programmiert werden. Die Steuerung von Bewegungsabläufen seiner Arme kann jedoch bisher nicht interaktiv in Echtzeit vorgenommen werden. Für jede Bewegung müssen die Parameter vor der Ausführung geplant und festgelegt werden. Verschiedene Programme und Programmier-Bibliotheken ermöglichen es, durch Interpolation zwischen einzelnen Punkten im Raum einen zusammengesetzten Pfad entlangzufahren. Diese Gegebenheit erschwert eine allgemeine Nutzung des Service-Roboters für diverse Projekte und Untersuchungen. Es sind viele Anwendungen denkbar, bei denen eine einfache manuelle Echtzeit-Steuerung durch einen menschlichen Benutzer nützlich wäre. Es könnte Beispielsweise des maschinelle Lernen von festen Bewegungsabläufen untersucht werden. Ein weiteres

Beispiel wäre die Programmierung neuer Greifbewegungen für die Roboterhand. Deren praktische Anwendung könnte in Alltagssituationen ausprobiert werden.

Das ebenfalls im selben Arbeitsbereich vorhandene haptische Eingabegerät könnte dabei zum Einsatz kommen. Damit können Punkte und Orientierungen im drei dimensionalen Raum angesteuert, beziehungsweise eingegeben werden. Darüber hinaus handelt es sich um ein sogenanntes Force-Feedback-Device (Kraftausgabegerät). So kann eine entsprechende Software dem Benutzer das Auftreten von haptischen Reizen simulieren.

1.2 Fragestellung

Diese Arbeit soll sich der Frage widmen, ob es möglich ist, ein dynamisches Telemanipulations-System zu entwickeln, dass in diese vorhandene Robotik-Umgebung mit der vorhandenen Hard- und Software integriert werden kann. Es soll geklärt werden, welche aktuellen Forschungsergebnisse dabei hilfreich sein können.

1.3 Ziel dieser Arbeit

Das Ziel dieser Arbeit ist die Entwicklung eines speziellen Telemanipulations-Systems zur Integration in ein bestehendes Roboter-System. Dabei soll eine Software-Lösung entstehen, die mit der vorhandenen Hardware benutzt werden kann. Das System soll dabei explizit die Anforderungen:

- Nutzung eines vorgegebenen Roboterarmes in seiner aktuellen Konfiguration,
- Nutzung eines vorgegebenen Eingabegerätes,
- Ermöglichung von interaktiv bedienbaren Telemanipulationen und
- intuitive Bedienbarkeit

erfüllen. Die genauere Spezifikation der vorgegebenen Geräte wird in Kapitel 4 gegeben. Dort ist auch eine Konkretisierung der Ziele dieser Arbeit enthalten. An der aktuellen Stelle im Text kann diese noch nicht erfolgen, da bisher noch nicht die dafür nötigen Grundlagen erörtert worden sind.

Zusätzlich zu den explizit geforderten Anforderungen ergeben sich bei näherer Beschäftigung mit dem Thema die impliziten Anforderungen der:

- Behandlung von Sicherheitsaspekten und der
- Stabilität und Transparenz des Systems.

Der vorliegende Text dokumentiert die Entwicklung dieses Telemanipulations-Systems und diskutiert den aktuellen Stand der Forschung in relevanten Fachgebieten. Auf die Herkunft entsprechender Quellen aus Veröffentlichungen wird durch Literaturangaben hingewiesen. Im Rahmen der theoretischen Diskussion soll diese Arbeit der Fragestellung aus Abschnitt 1.2 nachgehen.

Fast alle Veröffentlichungen zu dem Thema Telemanipulation sind in englischer Sprache verfasst. Darum soll in dieser Arbeit eine deutschsprachige Zusammenfassung der Grundlagen, Inhalte und Methoden dieses Bereiches gegeben werden. Der Leser kann sich damit einen Überblick über das Thema verschaffen.

Durch den vorliegenden Text soll der Leser in die Lage versetzt werden, die Struktur und die Funktion des entwickelten Systems zu verstehen. Dadurch soll eine Weiterentwicklung des Systems durch den Leser ermöglicht werden. Zu diesem Zweck ist auch die Dokumentation und universitätsinterne Veröffentlichung des entwickelten Programmcodes geplant.

1.4 Andere Telerobotik-Arbeiten (Related Work)

In diesem Abschnitt der Einleitung soll kurz auf die historischen und aktuellen Entwicklungen der Telemanipulations-Forschung eingegangen werden. Hierbei werden die wichtigen Themen und Aspekte nur so kurz gestreift, dass der Leser einen Überblick präsentiert bekommt. Viele Aussagen sind dabei mit Quellenangaben versehen, um dem interessierten Leser eine tiefer gehende Literaturrecherche zu ermöglichen. Eine ausführlichere Diskussion der Grundlagen und Themen, die für diese Arbeit relevant sind, folgt erst in Kapitel 2.

Telemanipulation könnte als der Grundstein der praktischen Robotik bezeichnet werden [Anderson und Spong 1989]. Mitte der 1940er Jahre hat Ray Goertz in den USA begonnen mechanische Telemanipulatoren zu entwickeln, um radioaktive Materialien handhaben zu können [Yashiro und Ohnishi 2008]. Auf Grund der festen mechanischen Kopplung bei diesen Geräten sind deren Berührungen von Objekten für den Benutzer spürbar. Auftretende Kontaktkräfte werden durch die Steifigkeit der mechanischen Systemkomponenten übertragen. In den 1950er Jahren ist der erste elektrisch gesteuerte Telemanipulator mit künstlich erzeugter Kraftrückkopplung entwickelt worden. In den 1960er Jahren hat es erste Untersuchungen über die Probleme solcher Telemanipulations-Systeme bei größeren Distanzen gegeben. Lange Laufzeiten der Steuerungsignale neigen zur Destabilisierung von rückgekoppelten Systemen. Zu dieser Zeit sind Bandbreitenbegrenzungen der Signale eine Lösung für

dieses Problem gewesen [Anderson und Spong 1989, Vertut u. a. 1981]. Dieses resultiert jedoch in eine Veringerung der maximalen Geschwindigkeit des Roboterarmes. Blake Hannaford hat gezeigt, dass auch der Benutzer, durch eine sehr starre Führung des Kontrollgerätes, stabilisierend auf das gesamte System wirken kann [Hannaford 1989b]. Daraus folgt, dass nicht nur das Telemanipulations-System allein, sondern auch dessen Benutzer bei der Modellierung des Gesamtsystems einbezogen werden sollte. Hannaford hat dafür eine geschätzte mechanische Impedanz von Menschen benutzt, wobei er in diesen Wert auch die dynamischen Aspekte der Biomechanik und Neurologie einbezogen hat [Hannaford 1989b]. In einem weiteren Ansatz dazu wird eine Bewegen-und-Warten-Strategie benutzt, um mit sehr großen Zeitverzögerungen von mehreren Sekunden, oder länger, arbeiten zu können. Hierbei wird ein Befehl gesendet und erst nach dessen bestätigter Ausführung der nächste Befehl geplant, was sich bei der geduldsamen Fernsteuerung von Erkundungsrobotern auf anderen Planeten als sinnvoll erweist [Gat u. a. 1994]. Erst 1989 stellten Robert J. Anderson und Mark W. Spong Lösungsstrategien vor, um diesem Problemen entgegenzuwirken [Anderson und Spong 1989]. Dafür haben sie mit der Passivitäts-Theorie, analog zu systemtheoretischen Ansätzen, die geforderte Stabilität mathematisch definiert. Mit Hilfe der Streutheorie ist die Entwicklung einer Kontrollstruktur gelungen, mit der Telemanipulatoren mit beliebig großer Zeitverzögerung stabilisiert werden können [Niemeyer und Slotine 1991]. Nach [Yashiro und Ohnishi 2008] und [Ferre u. a. 2007, S. 1–7] entspricht der Ansatz der Streutheorie dabei dem Konzept der Wellendigitalfilter von Alfred Fettweis aus den 1970er Jahren [Bilbao 2004]. Für diese Herangehensweise werden funktional equivalente elektrische Schaltkreise benutzt, um die dynamischen Eigenschaften der Kontrollstruktur eines Telemanipulators zu beschreiben und zu analysieren. Die meisten seit dem entwickelten und veröffentlichten Architekturen zur Telemanipulations-Kontrolle benutzen diese Methode. Diese Systeme können überwiegend mit der generalisierten Vier-Kanal-Architektur [Lawrence 1993] beschrieben werden, auch wenn weniger als vier Kommunikations-Kanäle benutzt werden.

Die fortschreitende Entwicklung der Computernetzwerk-Technologie hat die Idee der Telemanipulation über das Internet inspiriert [Goldberg und Siegwart 2002]. Diverse Veröffentlichungen, wie zum Beispiel [Yokokohji u. a. 2002, Berestesky u. a. 2004], beschäftigen sich mit den dabei aufkommenden Problemen von variierender Zeitverzögerung und Datenverlust bei der Kommunikation.

Bei einem Ansatz für den Umgang mit Signallaufzeitproblemen wird ein dynamisches Modell des Roboters und seiner Umgebung benutzt. Mit diesem Modell kann voraussagend berechnet werden, welche visuellen und haptischen Ausgaben dem Benutzer geboten würden, wenn die Datenübertragung schneller wäre. In [Buzan und

Sheridan 1989] wird ein Konzept vorgestellt, mit dem auf diese Weise der aktuelle Roboterzustand und Kontaktkräfte ausgeben werden können, bevor diese in der realen Welt existieren. Diese Methode ähnelt der Offline-Programmierung [Craig 2005] von Robotern, da Telemanipulations-Aufgaben mit dem Modell auch ohne einen realen Roboter geplant und sogar geübt werden können. Problematisch ist hierbei die geeignete Modellierung einer möglicherweise stark veränderlichen Umwelt [Mitra u. a. 2007]. Bei derartigen Systemen ist es wichtig, dass die Roboter autonome Funktionalitäten besitzen. Sie sollten zum Beispiel eigenständig ungewollte Kollisionen mit Hindernissen vermeiden, denn diese können nicht immer anhand des Modells vorhergesagt werden. Nach [Gat u. a. 1994] eignet es sich hierbei auch, Kontrollmethoden zu benutzen, die Befehle in Form von kompletten Aufgabenbeschreibungen übertragen. Dabei wird keine direkte Bewegungskontrolle benutzt, wodurch der Benutzer einen weniger starken Einfluss auf das Ergebnis hat. Diese Kontrollmethoden liefern bei sehr großen Signallaufzeiten im Bereich von Sekunden bis Minuten bessere Ergebnisse, sie sind jedoch nicht Inhalt der hier vorliegenden Arbeit.

Ein interessantes Anwendungsgebiet der aktuellen Telemanipualtions-Forschung ist zum Beispiel die Telechirurgie in der Medizin [Schlag und Graschew 1999, Ghodoussi u. a. 2002]. Darüber hinaus ist die Telemanipulationsforschung zur Zeit sehr eng mit den Forschungsgebieten haptische Geräte und virtuelle Realität verknüpft, wie aus den inhaltlichen Schwerpunkten von Veröffentlichungen wie [Ferre 2008, Ferre u. a. 2007, Adelstein u. a. 2002] deutlich wird.

1.5 Gliederung dieser Arbeit

In Kapitel 2 werden die theoretischen und technischen Grundlagen vorgestellt, die für das Verständnis dieser Arbeit notwendig sind. Dabei werden anfangs missverständliche Ausdrucksweisen aus der Telerobotik diskutiert und für den Gebrauch im Rahmen dieser Arbeit klar definiert. Anschließend werden, in dieser Arbeit benutzte, Technologien und mathematische Verfahren vorgestellt.

Darauf folgt in Kapitel 3 eine detailliertere Beschreibung der Grundlagen, die speziell für moderne Telemanipulations-Systeme relevant sind.

Kapitel 4 beschreibt die für diese Arbeit vorhandenen Voraussetzungen für die praktische Realisierung eines Telemanipulations-Systems. Dafür wird die Hardware und Software vorgestellt, auf der das entwickelte System implementiert wird.

Auf eine konkrete Implementation eines Telemanipulations-Systems wird in Kapitel 5 eingegangen. Dabei wird die Funktionsweise der einzelnen Komponenten und die des gesamten Systems ausführlich erläutert.

Abschließend wird in den Kapiteln 6 und 7 über den Erfolg dieser Arbeit reflektiert. Es werden auch Möglichkeiten diskutiert, wie die Ergebnisse dieser Arbeit weiter verwendet werden können. Zudem werden bereits geplante Erweiterungen des entwickelten Telemanipulations-Systems erörtert.

1.6 Begriffe und Notationen

Der hier vorliegende Text beschäftigt sich mit Themen des Fachgebiets Informatik, speziell der Robotik, der Computer-Programmierung und der Mathematik. In diesen Bereichen gibt es derart viele Fachbegriffe, dass es unmöglich wäre, alle hier relevanten zu erläutern. Es sei hier nur darauf hingewiesen, dass dem Leser grundlegende Kenntnisse dieser Fachgebiete unterstellt werden. Fachbegriffe, die in dieser Arbeit nicht erklärt werden, sind entweder fester Bestandteil der zugehörigen Standardliteratur oder im alltäglichen Sprachgebrauch etabliert. Dieses betrifft zum Beispiel viele Begriffe, die aus der englischen Sprache stammend zum Fachvokabular der Informatik gehören, wie „Computer“, „Joystick“, „Client“, „Server“, „Compiler“ oder „Multi-Threading“. Die direkte deutsche Übersetzung dieser Begriffe wird meistens nicht eindeutig mit der fachbezogenen Bedeutung in Verbindung gebracht, darum erweist sich ihre Benutzung als nicht praktisch. In solchen Fällen wird nachfolgend also eher der englische Begriff verwendet. Wohingegen auch viele deutsche Ausdrücke, wie „Maus“, „native Programm-Bibliothek“ und „das Programm läuft in dem Fenster“, eine Übersetzung darstellen, die von Fachkundigen verstanden wird.

In dieser Arbeit werden typische Symbole aus der Mathematik und der Physik verwendet. Die Bedeutung dieser kann mathematischen Standardwerken oder Formelsammlungen, wie zum Beispiel [Bartsch 2004], entnommen werden. Gewisse mathematische Grundkenntnisse des Lesers in den Bereichen analytische Geometrie, lineare Algebra und Differenzialrechnug werden jedoch vorausgesetzt. Die in dieser Arbeit verwendeten mathematischen Symbole sind gesondert in Anhang A aufgeführt.

Abkürzungen, die in diesem Text verwendet werden, sind immer an der Stelle ihres ersten Vorkommens erklärt.

2 Technische Grundlagen

In diesem Kapitel werden zunächst einige Grundlagen erläutert, die für diese Arbeit erforderlich sind. Unterteilt in Abschnitte sollen dabei die benötigten Fachbegriffe, Gegenstände und Methoden einzelner Fachbereiche erklärt werden. Die Bereiche Robotertechnik und haptische Technik sind dabei gewiss die Hauptbereiche. Zusätzlich wird auf gewisse Grundlagen aus der Mathematik, wie Geometrie und spezielle Koordinaten-Transformationen, die in der Robotik häufig Verwendung finden, eingegangen. Auch wenige Grundbegriffe aus der Computer-Programmierung werden kurz aufgegriffen. Anschließend wird im nächsten Kapitel auf die vorhandene Robotik-Umgebung eingegangen, die für diese Arbeit zur Verfügung steht. Dort geht es hauptsächlich um die Hardware des TAMS-Labors, die bei der praktischen Umsetzung dieser Arbeit benutzt werden. Doch auch die Software-Umgebungen werden vorgestellt.

2.1 Roboter und Manipulatoren

In der Robotik werden Begriffe verwendet, die sehr irreführend sein können. Selbst die Benutzung des Begriffs Roboter führt häufig zu Verwirrung, da seine Bedeutung sehr frei ist und sich im Kontext stark verändern kann. Die Bedeutung des Wortes Roboter ist seit seiner Entstehung im Jahre 1921 nicht die gleiche geblieben [Stahl und Robat 2008]. Zu dieser Zeit waren damit künstlich geschaffene menschliche Arbeitssklaven gemeint. Später sind damit Maschinen bezeichnet worden, die Menschen in ihrem Aussehen oder ihrer Funktion nachgeahmt haben. Häufig wird ihnen eine programmierte, künstliche Intelligenz zugesprochen. Diese Bedeutung kommt wohl noch am ehesten dem nahe, was der Großteil der Menschen, die das Wort Roboter kennen, damit verbindet.

In dem Fachvokabular der Robotik, die schnell zur wissenschaftlichen Disziplin aufgestiegen ist, ist dieser Begriff jedoch in einer viel allgemeineren Form gebräuchlich. So werden seit den 1970er Jahren in industriellen Produktionsanlagen Maschinen eingesetzt, die selbstständig Aufgaben erfüllen, die vorher von Menschen ausgeführt worden sind. Diese sehen nicht aus wie Menschen. Oft handelt es sich nur um eine Art

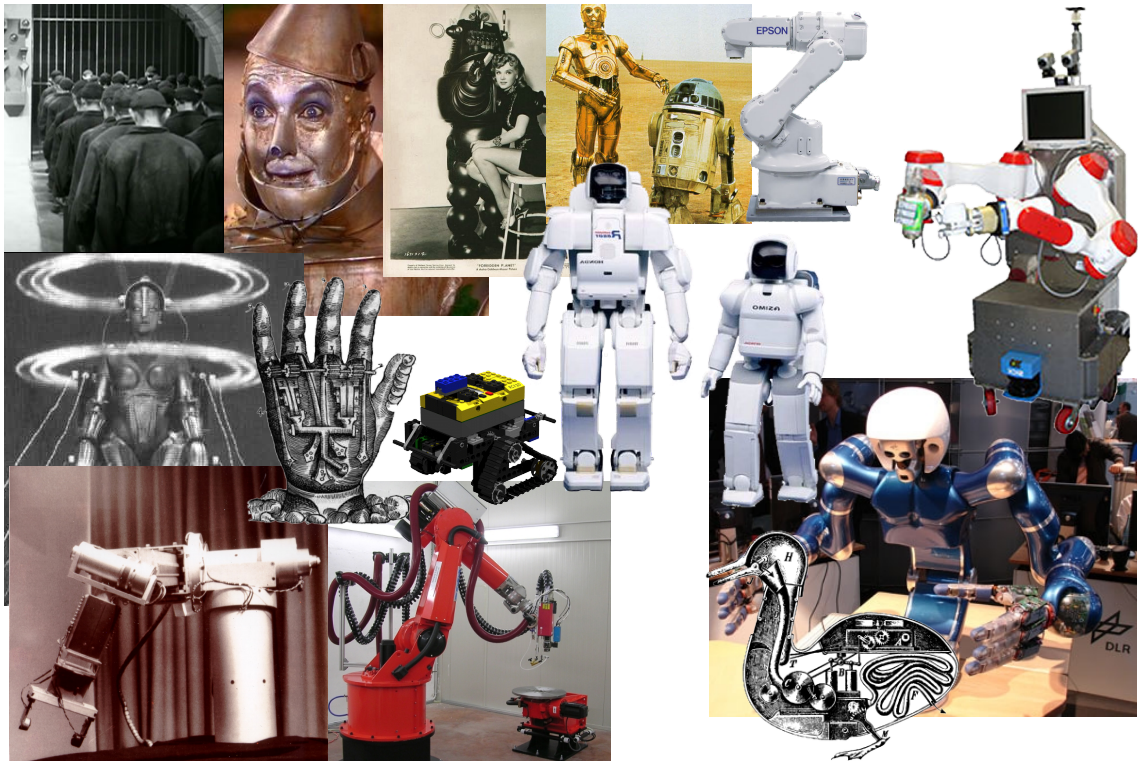


Abbildung 2.1: Kollage verschiedener historischer und aktueller Interpretationen des Roboterbegriffes.

Arm, der fest am Boden verankert ist. Auch ahmen sie nicht das komplette menschliche Verhalten nach. Sie wiederholen lediglich ein und den selben Bewegungsablauf um eine einzelne Aufgabe eines Menschen zu übernehmen. Trotzdem werden sie Industrieroboter, also Roboter, genannt. Der Hauptaspekt dieses Roboterbegriffes ist die Tatsache, das diese Maschinen umprogrammiert werden können, um neue Aufgaben zu übernehmen. Sie sind also allgemein einsetzbare Werkzeuge, die selbstständig arbeiten. Damit kommen sie der ursprünglich intendierten Bedeutung des künstlichen Arbeitssklaven wieder sehr nahe. Dabei wird einfach der Aspekt der Menschlichkeit ausgeblendet, dessen Bedeutung ohnehin noch schwerer zu definieren ist, wenn es um künstliche Menschen geht. Heutige offizielle Definitionen des Begriffes sind also so allgemein gehalten wie die Folgende aus [Lee und Suh 2006].

Definition nach Robot Institute of America. *Ein Roboter ist ein programmierbares Mehrzweck-Handhabungsgerät für das Bewegen von Material, Werkstücken, Werkzeugen oder Spezialgeräten. Der frei programmierbare Bewegungsablauf macht ihn für verschiedenste Aufgaben einsetzbar.*

Die Vieldeutigkeit dieses Begriffes erschwert den textlichen Umgang mit einem robo-

tiknahen Thema unnötig. Für die vorliegende Arbeit werden die verwendeten Begriffe daher etwas differenzierter definiert. Ein Roboter bezeichnet hier ein komplettes Ganzes, welches räumlich als eine Einheit betrachtet werden kann. Ein Roboter besteht meistens aus mehreren zusammengesetzten Teilen, wie Aktuatoren, mobilen Plattformen, Steuereinheiten und Computern. Diese stellen vermenschlicht ausgedrückt zum Beispiel Arme, Beine oder den Kopf dar. Ist nur ein einzelnes dieser Teile gemeint, dass mit einem Arm vergleichbar ist, wird es in dieser Arbeit als Manipulator bezeichnet. Dieser Begriff wird häufig für Industrieroboter verwendet, und betont deren Fähigkeit, die Umwelt zu manipulieren. An das Ende eines Manipulators wird meistens ein Werkzeug oder ein Sensor angebracht, dass als Endeffektor bezeichnet wird. Dabei kann es sich zum Beispiel um Greifer, Schweißgeräte oder Kameras handeln. Ein Greifer an einem Manipulator kann per Definition nun auch wieder als eigenständiger Roboter oder Manipulator bezeichnet werden. Darum wird er in dieser Arbeit zur Verdeutlichung immer als Greifer oder Endeffektor bezeichnet, während mit Manipulator nur der Roboterarm gemeint ist.

Abbildung 2.2: Manipulator mit Greifer als Endeffektor.

Da es in dieser Arbeit um Telemanipulation geht, werden Manipulatoren das Hauptthema sein. Daher werden nachfolgend die wichtigsten Grundbegriffe im Umgang mit Manipulatoren erläutert.

Nach [Craig 2005] hat jeder Manipulator eine Basiseinheit auf die einzelne Baugruppen aufgesetzt werden. Jede dieser Baugruppen ist beweglich an einem Gelenk angebracht. Üblich sind Rotations- oder Schub-Gelenke, die meistens durch Motoren und eine Steuerelektronik (Servos) in die gewünschte Stellung gebracht werden können. Die Gelenke werden auch als Achsen bezeichnet. Eine serielle Aneinanderreihung dieser beweglichen Teile fester Größe und Form bildet die kinematische Kette

eines Manipulators. Durch mathematische Verfahren kann die Raum-Position und -Orientierung aller Manipulatorelemente berechnet werden. Das wichtigste dieser Verfahren berechnet Position und Orientierung des Endeffektor-Arbeitspunktes, der als Tool Center Point, abgekürzt TCP, bezeichnet wird. Die Anzahl der Gelenke eines Manipulators wird allgemein als die Anzahl seiner Freiheitsgrade bezeichnet, da dies die Anzahl der Variablen ist, deren Werte frei wählbar sind. Mathematisch korrekt betrachtet, gibt die Anzahl der Freiheitsgrade die Anzahl der unabhängigen Positionsvariablen an. So bieten zum Beispiel zwei aufeinander folgende Schubgelenke, deren Achsen parallel liegen, nur einen Freiheitsgrad. Bei den meisten gebräuchlichen Manipulatoren sind die Gelenke jedoch so angeordnet, dass sie voneinander unabhängige Positionsvariablen realisieren. Allgemein gilt, je mehr Freiheitsgrade ein Manipulator hat, desto mehr Stellungen kann er mit seinem Tool Center Point erreichen. Für den Begriff Freiheitsgrad wird oft der englische Ausdruck Degrees of Freedom benutzt, der als DoF oder DOF abgekürzt wird.

Die kinematische Kette eines menschlichen Armes hat sieben Freiheitsgrade. Auf seine Basis folgt das Schultergelenk, dass als einzelnes Kugelgelenk schon drei Freiheitsgrade hat. Oberarm und Elle sind mit dem Ellenbogengelenk mit einem Freiheitsgrad verbunden. Die komplette Elle hat einen Freiheitsgrad, da sie um die eigene Längsachse verdreht werden kann, und das Handgelenk kann um zwei Achsen knicken und hat damit zwei Freiheitsgrade. Die meisten Manipulatoren sind an diesem Vorbild orientiert, um einen ähnlich großen Arbeitsbereich zu bieten. In der technischen Umsetzung von Manipulatoren sind die Gelenke und ihre Ansteuerung jedoch der aufwendigste Teil der Entwicklung und der Produktion. Daher werden in der Praxis häufig Manipulatoren mit weniger Freiheitsgraden eingesetzt.

2.2 Mathematische Grundlagen

Die Programmierung eines Manipulators erfordert viele allgemeine Berechnungsverfahren aus der liearen Algebra und der analytischen Geometrie. Zusätzlich sind auch robotikspezifische Standardverfahren gebräuchlich, die hier kurz erläutert werden sollen, da sie in dieser Arbeit benutzt werden. Im Folgenden werden dafür wichtige Begriffe und mathematische Verfahren erläutert, mit denen die Mechanik von Robotern beschrieben werden kann.

2.2.1 Koordinatensysteme

Die Bezeichnungen Position und Orientierung im Raum sind in den vorangegangenen Abschnitten schon oft verwendet worden. Mathematisch kann eine Raumposition, relativ zu dem Ursprung eines frei gewählten Koordinatensystems, durch einen 3×1-Vektor $\vec{p}$ angegeben werden. Eine Orientierung kann durch eine 3×3-Rotationsmatrix $\mathbf{R}$ beschrieben werden, die, relativ zu dem gewählten Koordinatensystem, die Basisvektoren des neuen, rotierten Koordinatensystems enthält.

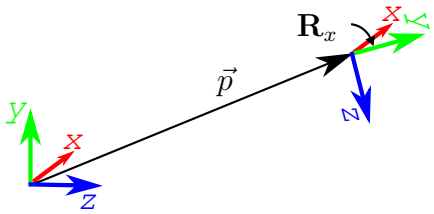


Abbildung 2.3: Veranschaulichte Koordinatentransformation aus einer Translation um $\vec{p}$ und einer Rotation $\mathbf{R}_x$.

Die in der Robotik üblichen kinematischen Berechnungen können als serielle Koordinatentransformationen betrachtet und implementiert werden. Zu diesem Zweck bietet sich die Verwendung von homogenen Koordinaten an.

$$\begin{pmatrix} \mathbf{R} & \vec{p} \\ \vec{0}^T & 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & p_1 \\ r_{21} & r_{22} & r_{23} & p_2 \\ r_{31} & r_{32} & r_{33} & p_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \left\{ \begin{array}{l} \text{Position und Orientierung,} \\ \text{homogene Koordinate und} \\ \text{Transformationsmatrix} \end{array} \right. \tag{2.1}$$

Damit können alle Transformationen durch Multiplikationen von 4×4-Matrix ausgeführt werden [Bartsch 2004]. Durch die Assoziativität der Matrixmultiplikation können die einzelnen Transformationen leicht zusammengefasst werden. So kann die Länge des ersten Manipulatorelementes als Translation $\mathbf{P}_1$ und das darauf folgende Gelenk als Rotation $\mathbf{R}_1$ modelliert werden. Diese beiden werden als Koordinatentransformation $\mathbf{T}_1 = \mathbf{P}_1 \cdot \mathbf{R}_1$ des kompletten Elementes zusammengefasst. Die Transformationsmatrix $\mathbf{T}_1$ enthält die Position und die Orientierung von dem Endpunt des Elementes relativ zu dem Basiskoordinatensystem. Es entsteht ein neues Koordinatensystem dessen Ursprung am Ende des Manipulatorelementes liegt und dessen Basisvektoren der Ausrichtung des Gelenkes entsprechen. Eine weitere Multiplikation mit der Transformationsmatrix $\mathbf{T}_2 = \mathbf{P}_2 \cdot \mathbf{R}_2$ überführt das daraus entstande Koordinatensystem an das Ende des nächsten Manipulatorelementes. Die komplette Berechnung der Lage des Tool Center Points eines n-achsigen Manipulators kann also mit der Formel $\mathbf{T}_{\text{TCP}} = \mathbf{T}_1 \cdot \mathbf{T}_2 \cdot \mathbf{T}_3 \cdots \mathbf{T}_n$ berechnet werden. Zu dem i-ten Element

des Manipulators gehört die Transformation $\mathbf{T}_i$. Für jede Transformation $\mathbf{T}_i$ müssen die relevanten oberen zwölf Matrixelemente bekannt sein, um die Kinematik ihres realen Manipulatorelementes zu beschreiben.

2.2.2 Denavit-Hartenberg-Transformation

Ein Standard im Umgang mit den Koordinatentransformationen einer kinematischen Kette ist seit 1955 die Denavit-Hartenberg-Transformation [Craig 2005, Thomas u. a. 2002, Shen u. a. 2007]. Sie erleichtert das Erstellen der homogenen Transformationsmatrizen einzelner Manipulatorelemente in einer strukturierten Weise. Es müssen dabei Regeln eingehalten werden, die festlegen, wie die Koordinatensysteme am Anfang und am Ende jedes Elementes ausgerichtet sind. Mit

$$\mathbf{T}_{\mathrm{DHT}}(\theta, d, a, \alpha) = \mathbf{R}_z(\theta) \cdot \mathbf{P}_z(d) \cdot \mathbf{P}_x(a) \cdot \mathbf{R}_x(\alpha) \tag{2.2}$$

kann die Denavit-Hartenberg-Transformation genau eines Manipulatorelementes aus den Transformations-Parametern (θ, d, a, α) berechnet werden. Damit kann jedes Element durch nur vier Parameter beschrieben werden. Der erste dieser Denavit-Hartenberg-Parameter $\theta \in \mathbb{R}$ beschreibt den Rotationswinkel des Gelenkes, dass nach Denavit-Hartenberg-Konvention am Anfang des Elementes liegen muss. Das rechtshändige Koordinatensystem muss so gewählt sein, dass die Rotation $\mathbf{R}_z(\theta)$ um die anfängliche z-Achse dreht. Darauf folgen eine Translation $\mathbf{P}_z(d)$ um die Länge $d \in \mathbb{R}$ in Richtung der Rotationsachse und eine Translation $\mathbf{P}_x(a)$ um $a \in \mathbb{R}$ entlang der x-Achse, die der Länge des Elementes entspricht. Zum Abschluss wird mit $\mathbf{R}_x(\alpha)$ um diese Achse mit dem Winkel $\alpha \in \mathbb{R}$ rotiert. Damit wird die z-Achse in die richtige Stellung für das Gelenk gebracht. In jedem Element sind drei dieser Parameter konstant und einer gibt die variable Gelenkstellung an. Die Gelenkstellung $j_i \in \mathbb{R}$ wird bei einem Rotationsgelenk also auf den θ-Wert des Elementes $i \in \{1,2,3,\ldots,n\}$ übertragen. Hierbei sei $n \in \mathbb{N}$ die Anzahl der Gelenke. Handelt es sich nicht um ein Rotationsgelenk können auch andere Parameter, wie d bei einem Schubgelenk, variabel sein. Im Folgenden werden zur Verkürzung der Darstellung nur kinematische Ketten mit Rotationsgelenken behandelt und es wird die Notation

$$\mathbf{T}_{\mathrm{DHT}}(j_i)_i = \mathbf{T}_{\mathrm{DHT}}(j_i,d_i,a_i,\alpha_i) \tag{2.3}$$

verwendet.

Durch die Multiplikation einzelner Denavit-Hartenberg-Transformationen kann wiederum die komplette Transformation $\mathbf{DHT}(\vec{j})$ einer kinematischen Kette berechnet werden, die durch gegebene Denavit-Hartenberg-Parameter beschrieben ist. Die Ge-

lenkwinkel j_i werden in dem Vektor $\vec{j}$ zusammengefasst, dessen Spaltenanzahl der Anzahl der Gelenke n entspricht.

$$\mathbf{DHT}(\vec{j}) = \mathbf{T}_{\mathrm{DHT}}(j_1)_i \mathbf{T}_{\mathrm{DHT}}(j_2)_i \mathbf{T}_{\mathrm{DHT}}(j_3)_i \cdots \mathbf{T}_{\mathrm{DHT}}(j_n)_i \tag{2.4}$$

Im Folgenden wird als Denavit-Hartenberg-Transformation immer die zusammengesetzte Transformation $\mathbf{DHT}(\vec{j})$ bezeichnet. Die Funktion $\mathbf{DHT}\colon \mathbb{R}^n \to \mathbb{R}^{4\times4}$ bildet die Vektorwerte des Gelenkwinkelraumes auf dreidimensionale Koordinatensysteme im kartesischen Raum ab.

Schwächen der Denavit-Hartenberg-Parameter

Abbildung 2.4 zeigt eine mögliche Wahl der Denavit-Hartenberg-Parameter für den Manipulator PA10-6C von Mitsubishi Heavy Industries. Einige Elemente des Manipulators sind in zwei Elemente unterteilt, von denen das zweite komplett konstante Parameter hat. Ohne diese Zusatzelemente wäre es nicht möglich die Basis der Folgeelemente korrekt auszurichten. Dieses ist eine Schwäche des Denavit-Hartenberg-Konzeptes. Eine Erweiterung der vier Parameter um einen fünften Rotations-Parameter um die y-Achse würde die Zusatzelemente unnötig machen. Ebenfalls können Kardan- und Kugelgelenke nur durch mehrere Denavit-Hartenberg-Transformationen beschrieben werden. Auch diese können nur durch das Hinzufügen weiterer Rotations-Parameter in jedem Element beschrieben werden. Für die Kompatibilität zu den, häufig in Industrie und Forschung verwendeten, Denavit-Hartenberg-Parametern wird in dieser Arbeit jedoch auf derartige Erweiterungen verzichtet.

Die Verwendung der homogenen Matrizen bei der Denavit-Hartenberg-Transformation führt zu vielen unnötigen Multiplikationen. Für eine effiziente Implementation können jedoch alle beteiligten Matrizen durch Ausmultiplizieren zusammengefast werden, so dass nur noch die nötigen Berechnungen enthalten sind. Sind alle verwendeten Matrizen homogene Matrizen, sollte die Implementation der Matrixmultiplikation selbstverständlich die unnötigen Multiplikationen mit 0 und 1 aussparen.

2.2.3 Inverse Kinematik

Für eine zusammenhängende Bewegung des Tool Center Points eines Manipulators wird der Begriff der Trajektorie benutzt [Craig 2005]. Dabei müssen Signalverläufe für die Stellungen $\vec{j}$ der einzelnen Gelenke berechnet werden. Der Signalverlauf aller Gelenke wird durch eine Funktion $\vec{j}(z)$ der Zeit $z \in \mathbb{R}$ notiert. Bei der Betrachtung praktischer Realisierungen wird diese Funktion diskretisiert und als Folge einzelner Werte

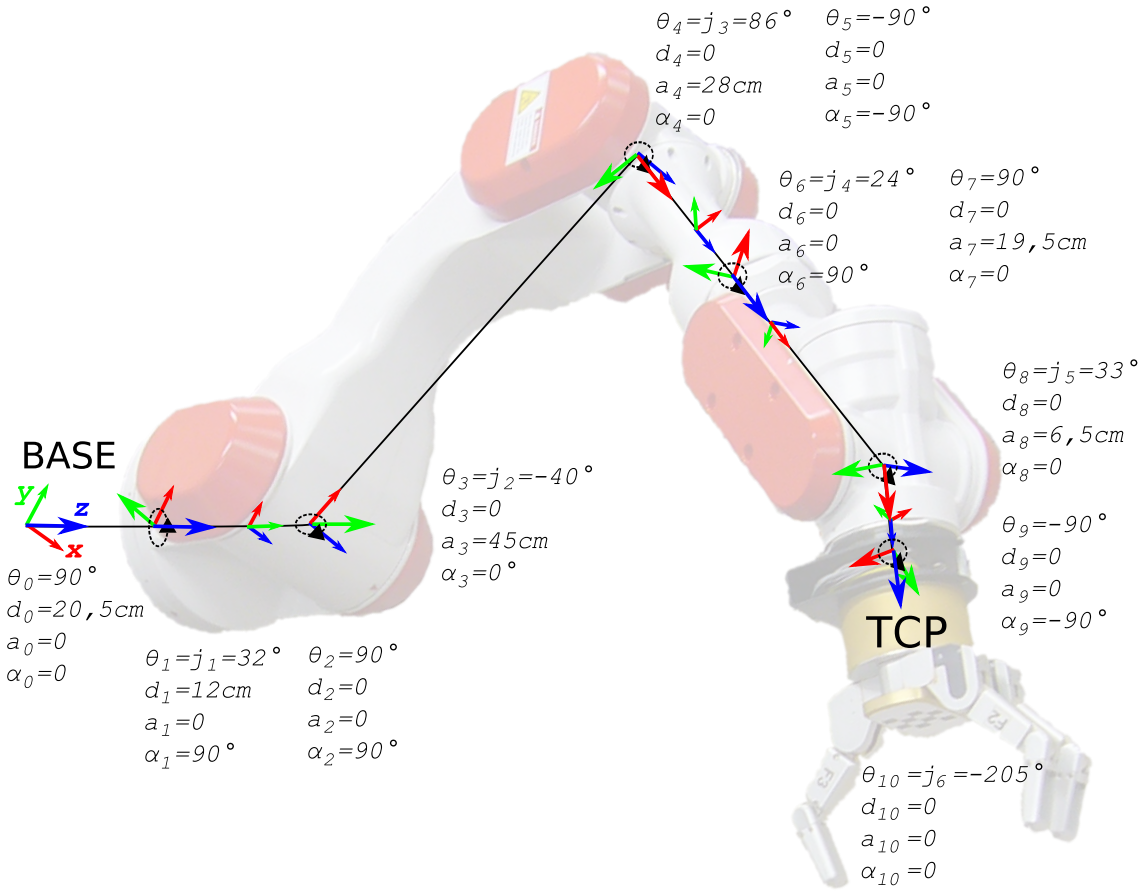


Abbildung 2.4: Mögliche Denavit-Hartenberg-Parameter für den Mitsubishi Heavy Industries PA10-6C. Die dazugehörigen Koordinaten-Transformationen sind veranschaulicht. Die aktuellen Gelenkstellungen sind durch $j_{1...6}$ gegeben.

$\vec{j}_t$, mit $t \in \mathbb{N}$ notiert. Dabei stellen die Werte von t einzelne äquidistante Zeitpunkte dar. Soll eine Trajektorie entlang einer geraden Linie in kartesischen Koordinaten der realen Welt verlaufen, so ergibt sich dafür meistens ein nicht linearer Singalverlauf $\vec{j}(z)$. Es wird eine Methode zur Berechnung der Funktion $\mathbf{IK}\colon \mathbb{R}^{4\times4} \rightarrow \mathbb{R}^n$ benötigt, um die kartesischen Raumkoordinaten auf passende Gelenkwinkelkoordinaten abzubilden.

Die Denavit-Hartenberg-Transformation ist eine Methode, mit der die direkte Kinematik eines Manipulators beschrieben und berechnet werden kann. Damit ist gemeint, dass bei gegebenen Gelenkstellungen $\vec{j}$ die Koordinaten-Transformation $\mathbf{DHT}(\vec{j})$ berechnet werden kann. Diese beschreibt die Position $\vec{p} \in \mathbb{R}^3$ und Orientierung $\mathbf{R} \in \mathbb{R}^{3\times3}$ des Tool Center Points relativ zur Koordinatenbasis. Es gibt dafür im-

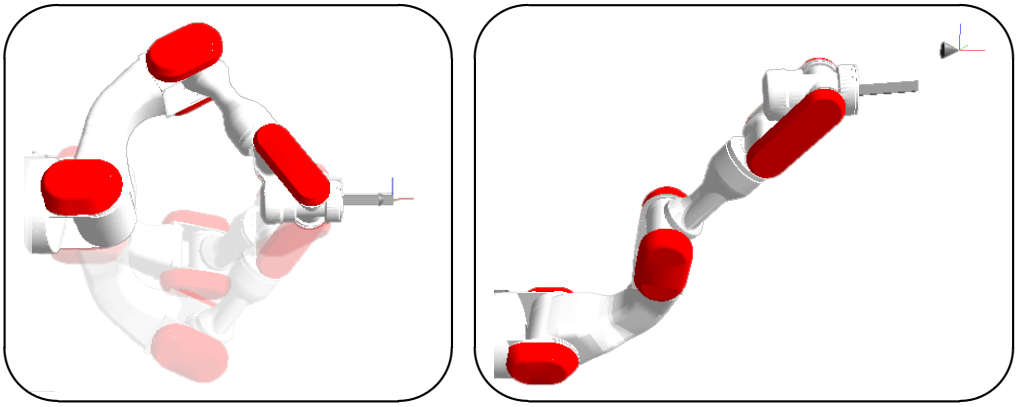
mer eine eindeutige Lösung.

$$\mathbf{DHT}(\vec{j}) = \mathbf{T}_{\text{Ziel}}(\mathbf{R}, \vec{p}) = \begin{pmatrix} \mathbf{R} & \vec{p} \\ \vec{0}^T & 1 \end{pmatrix} \tag{2.5}$$

Aufwendiger ist der umgekehrte Weg, bei dem Gelenkstellungen aus gegebenen Raumkoordinaten zu berechnen sind. Diese Berechnung wird als inverse Kinematik bezeichnet. Im Allgemeinen muss dafür das Gleichungssystem

$$\mathbf{0} = \mathbf{DHT}(\vec{j}) - \mathbf{T}_{\text{Ziel}}(\mathbf{R}, \vec{p}) \tag{2.6}$$

für bekannte $\vec{p}$ und $\mathbf{R}$ und unbekanntes $\vec{j}$ gelöst werden. Die Lösung ist dabei nicht unbedingt eindeutig oder vorhanden. Bei praktischen Anwendungen der Telemanipulation ist jedoch oft die Anforderung, immer Lösungen zu finden, an das Lösungsverfahren gestellt. Bei Mehrdeutigkeit muss genau eine sinnvolle Lösung gefunden werden. Falls keine Lösung existiert muss trotzdem eine Näherungslösung gefunden werden, die den Abstand zum geforderten Ziel minimiert. Abbildung 2.5a zeigt einen Manipulator, dessen Tool Center Point das Ziel durch unendlich viele verschiedene Gelenkstellungen erreichen kann. Er befindet sich in einer sogenannten Singularität. Es gibt dabei mehrere Möglichkeiten einen Zielpunkt zu erreichen. Abbildung 2.5b zeigt den selben Manipulator mit einem unerreichbaren Ziel. Die Lösung der inversen Kinematik existiert hierbei nicht.



(a) Mehrdeutigkeit (b) Unlösbarkeit

Abbildung 2.5: Veranschaulichung der Probleme von inversen Kinematiken.

Das Gleichungssystem (2.6) kann entweder mit algebraischen, mit geometrischen oder mit numerischen Lösungsverfahren gelöst werden [Craig 2005]. Die algebraischen Verfahren liefern exakte Lösungen und lassen genaue Aussagen über die Lösbarkeit und

die Menge der Lösungen zu. Die geometrischen Verfahren sind im Allgemeinen algebraische Verfahren, bei denen bekannte geometrische Eigenschaften der gegebenen Kinematik ausgenutzt werden. Die algebraischen Verfahren sind aus verschiedenen Gründen meistens nicht für Robotikanwendungen geeignet. Ungenauigkeiten bei der numerischen Speicherung von Zahlenwerten sorgen dafür, dass meistens keine Lösung gefunden werden kann. Aus diesem Grund werden in der Praxis und auch im nachfolgenden Text numerische Lösungsverfahren verwendet. Für weitere Informationen zu allgemeinen Lösungsverfahren der inversen Kinematik sei der Leser auf Veröffentlichungen zu diesem Thema, wie [Elias 2004, Park 2008, Wang und Chen 1991] oder [Craig 2005, S. 101–134], verweisen.

Numerische Verfahren verbessern die Lösung iterativ. Sie sind daher sehr gut dafür geeignet, Lösungsnäherungen zu finden, wenn keine Lösung existiert. Allgemein handelt es sich um ein Optimierungsproblem, bei dem der Positions- und Orientierungs-Fehler $e(\vec{j}) = dist\left(\mathbf{DHT}(\vec{j}) - \mathbf{T}_{\text{Ziel}}(\mathbf{R}, \vec{p})\right)$ minimiert werden soll. Die Funktion $dist\colon \mathbb{R}^{4\times4} \to \mathbb{R}$ ordnet dabei der Fehlermatrix ein reeles Maß für die Abweichung von Position und Orientierung zu. Dafür kann auch eine beliebige Metrik $d\colon \mathbb{R}^{4\times4} \times \mathbb{R}^{4\times4} \to \mathbb{R}$ mit $\mathbf{DHT}(\vec{j})$ und $\mathbf{T}_{\text{Ziel}}(\mathbf{R}, \vec{p})$ benutzt werden. Das Newtonverfahren [Bartsch 2004, S. 130] ist ein Beispiel für ein Verfahren zur Minimierung von e. Für den mehrdimensionalen Fall wird dabei die Jacobi-Matrix der partiellen Ableitungen von $e(\vec{j})$ benutzt, die ebenfalls numerisch approximiert werden kann. Die Jacobi-Matrix von $e\colon \mathbb{R}^n \to \mathbb{R}$ entspricht dem transponierten Gradienten von e. Dem Newtonverfahren ähnliche Verfahren haben im Allgemeinen gute Konvergenzeigenschaften, versagen jedoch bei Singularitäten oder schlecht gewählten Startwerten der Iteration. Gradienten-Abstiegsverfahren haben diese Probleme nicht [Wang und Chen 1991], darum wird in dieser Arbeit das Verfahren des steilsten Abstiegs nach [Luenberger 2003] zum Einsatz kommen. Es handelt sich hierbei um eine heuristische Suche im Gelenkwinkel-Lösungsraum. Dabei werden fortlaufend alle Werte der Gelenkwinkel j_i mit Hilfe des Gradienten der Fehlerfunktion $\vec{\nabla}e$ verbessert. Es wird die Iterationsvorschrift

$$\vec{j}_{t+1} = \vec{j}_t + c\,\vec{\nabla}e(\vec{j}_t) \tag{2.7}$$

benutzt. Um auch im Fall von $\vec{\nabla}e(\vec{j}_t) = \vec{0}$ weiter iterieren zu können, wird eine Folge von Pseudo-Zufallszahl $r_t \in [-\epsilon,\epsilon]$ mit $\epsilon \in \mathbb{R}$ addiert.

$$\vec{j}_{t+1} = \vec{j}_t + c\,\vec{\nabla}e(\vec{j}_t) + r_t \tag{2.8}$$

Der Wert $c \in \mathbb{R}$ beeinflusst die Abstiegsgeschwindigkeit. Die Werte c und ϵ können in Abhängigkeit von t oder $e(\vec{j}_t)$ verändert werden, um zum Beispiel die Konvergenz-

geschwindigkeit zu verbessern. Für $e(\vec{j_t}) = 0$ muss $\epsilon = 0$ gelten, damit die gefundene Lösung bestehen bleibt. Mit variablem ϵ kann zum Beispiel das Verfahren der simulierten Abkühlung erzeugt werden, mit dem die Suche nach dem globalen Minimum fortgeführt werden kann, wenn die normale Suche in einem lokalen Minimum stehenbleiben würde [Wegener 2005]. Die Funktion *dist* kann so gewählt werden, das die Positionsabweichung stärker auf das Ergebnis wirkt als die Abweichung der Orientierung. Die iterative Suche korrigiert in diesem Fall zuerst die Position und erst danach die Orientierung. Durch die Verwendung von Zufallszahlen, ist das Verfahren nicht deterministisch.

2.3 Haptische Eingabegeräte

Die haptische Wahrnehmung des Menschen umfasst sehr viele Teilbereiche, die in die zwei Bereiche Oberflächensensibilität und Tiefensensibilität aufgeteilt werden. Die Oberflächensensibilität erfasst Druckänderungen auf der Oberfläche, also der Haut beim Menschen. Sie wird als taktile Wahrnehmung bezeichnet. Unter kinästhetischer Wahrnehmung oder Tiefensensibilität versteht man die Wahrnehmung von Position, Lage, Bewegung und Kraftanstrengung einzelner Teile des eigenen Körpers. Zusätzlich fallen auch Temperatur- und Schmerzwahrnehmung unter haptische Wahrnehmung, jedoch wird auf diese hier nicht weiter eingegangen, da sie nichts mit dieser Arbeit zu tun haben.

Ein haptisches Ausgabegerät ist, analog zum Bildschirm für die visuelle Ausgabe, ein Gerät, das die haptische Wahrnehmung des Benutzers stimuliert. Diese Geräte können unterschiedliche Teile der Haptik adressieren. Bei Computerspiel-Steuerungen oder Mobiltelefonen werden häufig einfache taktile Vibrationsreize benutzt. Für Sehbehinderte gibt es Geräte, die die Blindenschrift Braille ausgeben und damit ebenfalls taktile Reize erzeugen. Besitzt ein haptisches Gerät einen Griff, auf den es Kräfte wirken lassen kann, so kann es kinästhetische Reize erzeugen. Der Benutzer der den Griff in der Hand hält, kann bei Bewegungen der Hand einen unterschiedlich starken Widerstand oder eine unterstützende Kraft spüren. Derartige Geräte gibt es in vielen verschiedenen Bauformen. Meistens handelt es sich um ein kombiniertes Eingabe- und Ausgabegerät, da die jeweils aktuelle Lage des Griffes über Sensoren erfasst und an den Computer geleitet wird. Es gibt Joysticks mit sogenanntem Force Feedback, die die, bei Joysticks übliche, zweidimensionale Eingabe durch eine zweidimensionale Kraftausgabe auf den selben Achsen ergänzen. Für die Bereiche virtuelle Realität und Robotik gibt es Geräte, die Eingaben und Ausgaben mit einer höheren Anzahl an Freiheitsgraden erlauben. Diese bedienen sich meistens paralleler oder serieller kinematischer Ketten, deren Gelenke Sensoren enthalten und zusätzlich mit

Servomotoren angesteuert werden können. Diese geben, genau wie der Joystick, nur einen Punkt im Raum aus. Der Benutzer fühlt die Position seiner Hand, und damit auch dieses Raumpunktes, dann mit Hilfe der kinästhetischen Wahrnehmung. Wird von dem Gerät eine Kraft ausgeübt, wird diese vom Benutzer auch taktil, als Druck auf der Haut wahrgenommen. Jedoch werden keine flächigen, taktil wahrnehmbaren Strukturen, wie bei der Braille-Schrift, ausgegeben. Eine räumliche Struktur kann nur ausgegeben werden, indem der Benutzer den Wahrnehmungspunkt bewegt und das Gerät dabei an unterschiedlichen Positionen die ausgegebene Kraft anpasst. Bei einer linearen Bewegung des Griffes entlang der x-Achse kann zum Beispiel abhängig von der aktuellen Position x die Kraft $F_y = |a \sin(x\frac{2\pi}{T})|$ in Richtung der y-Achse erzeugt werden. Je nach Wahl der Amplitude a und der Periodenlänge T erzeugt dies bei dem Benutzer unterschiedliche Wahrnehmungen. Mit kleineren Werten kann eine raue Oberfläche simuliert werden. Größere Werte erzeugen größere räumliche Konturen, wie eine Hügellandschaft. Die Berechnung dieser Kraftausgabe in Abhängigkeit von der Position wird hapitsches Rendern genannt.

Abbildung 2.6: Die haptischen Schreibtisch-Geräte Omega3 mit paralleler Kinematik (links) und PHANTOM Desktop mit serieller Kinematik (rechts).

Der Aufbau und die elektrotechnische Funktion dieser punktuellen haptischen Eingabegeräten entspricht allgemein dem von Manipulatoren. Darum werden die beiden Begriffe besonders in der Telerobotik-Literatur oft synonym verwendet. So liest man in dem Bereich der Telemanipulation häufig, dass ein Master-Manipulator einen Slave-Manipulator kontrolliert. Dies geht auch geschichtlich daraus hervor, dass bei den ersten Telemanipulatoren auf beiden Seiten des Systems Manipulatoren verwendet worden sind, die die gleiche Größe und Kinematik gehabt haben, siehe Abbildung 1.1 auf Seite 1. Die heute entwickelten Geräte sind überwiegend so klein, dass sie auf einem Schreibtisch platziert werden und zusammen mit anderen gewohnten

Ein- und Ausgabegeräten eines Computers genutzt werden können. Die Geräte aus Abbildung 2.6 gehören zu dieser Gruppe. Nachfolgend sind mit der Bezeichnung haptisches Gerät nur noch diese Schreibtischgeräte gemeint, falls dies nicht anders kenntlich gemacht ist.

Die haptische Wahrnehmung hat eine stärkere zeitliche Auflösung als die visuelle Wahrnehmung. Nach den Angaben der Hersteller von haptischen Geräten ist eine Ausgabe mit mindestens 1000 Hertz nötig, um bei einem menschlichen Benutzer den Eindruck von Kontakt mit festen Körpern zu simulieren [OpenHaptics 2004] [You und Sung 2008]. Die akustische Wahrnehmung arbeitet mit einer ähnlich hohen Erkennungsrate, da eine hohe Frequenz wie 4000 Hertz bereits nach etwa drei Wellendurchläufen erkannt wird. Digitale Audiosignale werden überwiegend Stereofon als zweidimensionales Signal mit einer Rate von mindestens 8000 Hertz ausgegeben. Bei der visuellen Wahrnehmung sind für flüssige Bewegungserkennung schon etwa 20 Hertz ausreichend, wobei die räumliche Auflösung wesentlich höher ist. Das dedeutet, dass eine größere Datenmenge verarbeitet werden muss. Bei der Programmierung von haptischen Geräten muss die hohe Ausgaberate beachtet werden und spezielle Programmiertechniken wie Multithreading und datenstromorientierte Programmierung in Echtzeit verwendet werden.

Tabelle 2.1: Häufig verwendete Formeln des haptischen Renderns.

Feder (Hooke'sches Gesetz)	$\vec{F}_f = k \cdot -\Delta\vec{x}$	Federkonstante k Positionsabweichung $\Delta\vec{x}$
Dämpfung	$\vec{F}_d = -d \cdot \vec{v}$	Dämfungskonstante d Geschwindigkeit $\vec{v} = \dot{\vec{x}} \left(= \frac{dx}{dt}\vec{x}\right)$
Reibungswiderstand (Coulomb'sche Reibung)	$\vec{F}_r = -c\lvert\vec{F}_N\rvert\frac{\vec{v}}{\lvert\vec{v}\rvert}$	Reibungskoeffizient c Normalkraft $\vec{F}_N$
Trägheit (Newton'sche Gesetze)	$\vec{F}_r = m \cdot \vec{a}$	Masse m Beschleunigung $\vec{a} = \dot{\vec{v}} = \ddot{\vec{x}}$

Für die Programmierung von haptischen Geräten gibt es Programmier-Schnittstellen der verschiedenen Hersteller, wie zum Beispiel das OpenHaptics Toolkit von SensAble Technologies [OpenHaptics 2004]. Es gibt auch einige Projekte, die Programmier-Bibliotheken entwickelt haben, die herstellerübergreifend benutzt werden können. Hierbei sind die quellcode-offenen Chai3D, JTouchToolkit [Archer 2007] und H3DAPI zu nennen.

Für das haptische Rendern werden Methoden der klassischen Mechanik aus der Physik verwendet. Da die Eigenschaften von physikalischen Körpern simuliert werden sollen, müssen deren statische und dynamische Eigenschaften unter Krafteinwirkung berechnet werden. Dabei kommen Formeln für die Beschreibung von Federn, Dämpfern, Reibungswiederstand und Massenträgheit zum Einsatz [OpenHaptics 2004]. Die am häufigsten dafür verwendeten Formel sind in Tabelle 2.1 aufgelistet. Die Berührung eines festen Körpers im Raum kann zum Beispiel mit dem Hooke'schen Gesetz simuliert werden. Mit einer Fallunterscheidung wird die Federkonstante k im Körperinneren auf einen sehr hohen Wert gesetzt und außerhalb des Körper ist $k = 0$. Die Positionsabweichung $\Delta\vec{x}$ entspricht hierbei dem Abstand zur Körperoberfläche.

2.4 Internet-Protokoll

Zur Übertragung von Daten zwischen Computern über ein Computernetzwerk werden Netzwerkprotokolle verwendet. Diese Protokolle definieren genau, auf welche Weise die zu übertragenden Nutzdaten als einzelne Datenpakete versendet werden. Das heutzutage am häufigsten benutze Netzwerkprotokoll ist das Internet-Protokoll, das mit IP abgekürzt wird. Die protokollbasierte Netzwerkkommunikation kann mit Schichtmodellen betrachtet werden, die das Zusammenwirken von mehreren auf einander aufbauenden Protokollen verdeutlichen. Das Internet-Protokoll benutzt ein Protokoll der darunter liegenden Netzzugangsschicht, wie zum Beispiel Ethernet oder WLAN. Auf der Schicht des Internet-Protokolls wird die Weiterleitung von Datenpaketen über mehrere Zwischenverbindungen geregelt, während die Protokollschichten darunter nur für die einzelnen Teilverbindungen der Weiterleitung zuständig sind. Auf das Internet-Protokoll bauen die Protokolle der Transportschicht auf. Wichtige Vertreter dieser Gruppe sind das Transmission Control Protocol (TCP/IP) [Postel 1981] und das User Datagram Protocol (UDP/IP) [Postel 1980]. Diese Protokolle regeln die Weiterleitung einzelner Datenpakete an die richtigen Anwendungsprogamme auf den verbundenen Computern. Oberhalb dieser Protokollschicht setzen die einzelnen Anwendungen ihre spezifischen Protokolle auf. Die Anwendungsschicht umfasst zum Beispiel das HTTP für Internetbrowser, FTP für Dateiübertragungen, DNS für Internet-Adressinformationen von Namensservern und NTP für Zeitsynchronisation.

Es sei erwähnt, dass es noch viele andere gebräuchliche Netzwerkprotokolle gibt als die hier vorgestellten. In dieser Arbeit wird jedoch nicht weiter auf diese eingegangen, da nachfolgend nur eine Verwendung der vorgestellten Standards vorgesehen ist. Weitergehende Informationen zum Thema Internet-Protokoll und Computernetzwerke können zum Beispiel in [Tanenbaum 2003, Stevens 2004] und den dort referenzierten Veröffentlichungen gefunden werden.

Tabelle 2.2: Schichtenmodell häufig verwendeter Netzwerkprotokolle. Protokolle, die für Telemanipulation genutzt werden, sind fett gedruckt.

Schicht	Beispiel-Protokolle			
Anwendungsschicht	HTTP, FTP, SSH, SMTP	DNS, DHCP, NTP, **TMP**		
Transportschicht	TCP	UDP	**ETP**	
Internetschicht	IP			**RTNP**
Netzzugangsschicht	Ethernet, WLAN			

Die Unterschiede zwischen TCP/IP und UDP/IP sind die Folgenden. TCP/IP ermöglicht eine verlässliche Verbindung zweier Anwendungen. Übertragungsfehler und -verluste werden dabei erkannt und korrigiert oder erneut gesendet. Das UDP/IP bietet diesen Vorteil nicht. Auf keiner Seite der Verbindung wird ein möglicher Verlust eines Datenpaketes erkannt. Der Vorteil des UDP/IP liegt jedoch in der wesentlich höheren Menge an Daten, die pro Zeiteinheit übertragen werden können. Eine allgemeine Eigenschaft der paketorientierten Datenübertragung ist die variierende Zeitverzögerung zwischen Sende- und Empfangszeitpunkt.

Für die Netzwerkkommunikation im Bereich der Telemanipulation gibt es bisher kein Standard-Protokoll auf der Anwendungsschicht. Meistens werden direktere Kommunikationsverbindungen benutzt, die nicht paketorientiert sind und höhere Datenraten haben. Wie jedoch schon in Abschnitt 1.4 beschrieben, hat es bereits erfolgreiche Ansätze gegeben, auch Internet-Protokoll-Netzwerke für Telemanipulations-Systeme zu verwenden. Allgemein wird dafür immer ein neues Protokoll entwickelt, dass für die schnelle Übertragung dann UDP/IP benutzt [Ferre 2008, S. 4]. Da diese vereinzelt verwendeten Protokolle keine weltweit bekannten Namen haben, werden sie innerhalb dieser Arbeit generisch als Telemanipulation Protocol (TMP) bezeichnet. Es sind auch schon Telemanipulations-Protokolle entwickelt worden, die direkt auf einer niedrigeren Protokollschicht aufsetzen, um den hohen Echtzeitanforderungen gerecht zu werden. Mit dem Efficient Transport Protocol (ETP/IP) wird in [Wirz u. a. 2008] eine alternative zu UDP/IP vorgestellt, die direkt auf IP aufsetzt. Durch ETP/IP wird ein neues Protokoll für die Transport- und Anwendungsschicht imple-

mentiert, dass durch Flusskontroll-Mechanismen leicht verbesserte Übertragungsdauern erreicht. Das Real-Time Network Protocol (RTNP/Ethernet) aus [Uchimura und Yakoh 2004] verarbeitet direkt die Pakete der Netzzugangschicht Ethernet. Es handelt sich dabei ebenfalls um eine Implementation des kompletten darüber liegenden Protokollstapels. Hierdurch kann ein noch höherer Geschwindigkeitsgewinn erreicht werden. Für diese Implementation wird jedoch ein spezielles Echtzeit-Betriebssystem, wie RT-Linux oder RTAI, verwendet [Uchimura und Yakoh 2004, S. 943].

Eine praktische Nutzung derartig spezieller Telemanipulations-Protokolle ist mit einem hohen Entwicklungsaufwand verbunden, da die Paketverarbeitung auf mehreren Protokollschichten neu implementiert werden muss. Die Verarbeitung der Standardprotokolle TCP/IP und UDP/IP ist hingegen bei vielen Betriebssystemen bereits integriert. Bei sogenannten Streaming-Protokollen handelt es sich um Netzwerkprotokolle für die effiziente Übertragung von Audio- oder Videodaten. Im Bereich dieser Protokolle gibt es Standards wie das Real Time Transport Protocol (RTP/IP) [Schulzrinne u. a. 2003] oder H.323 für Internet-Telefonie. Derartige Protokolle können ebenfalls das Internet-Protokoll verwenden, um sehr hohe Datenübertragungsraten zu erhalten. Dabei kommen Kontrollfluss-Mechanismen wie bei ETP/IP zum Einsatz. Darüber hinaus werden jedoch meistens große Datenmengen für die Übertragung stark komprimiert. Das angestrebte Ziel ist also nicht die, für Telemanipulations-Systeme wichtige, sehr schnelle Übertragung von sehr kleinen Datenmengen. Hierbei bringen Kompressionsverfahren keinen Vorteil, da sie das Versenden von Daten verzögern.

2.5 Zusammenfassung

In diesem Kapitel sind wichtige begriffliche, technische und theoretische Grundlagen für die vorliegende Arbeit und allgemein für Telemanipulations-Systeme erläutert worden. Die Begriffsdefinitionen von Roboter und Manipulator ist präzisiert worden. Der grundsätzliche Aufbau von Manipulatoren mit dazugehörigen mathematischen Konzepten ist vorgestellt worden. Diese umfassen homogene Koordinatentransformationen, die Denavit-Hartenberg-Transformation und die inverse Kinematik. Ebenso sind die haptische Wahrnehmung und die Möglichkeiten von haptischen Geräten vorgestellt worden. Eine kurze Einführung in das Protokollschichten-Modell von heutigen Computernetzwerken ist präsentiert worden. Nach der Einführung in diese allgemeinen Grundlagen kann nun eine Diskussion der Grundlagen folgen, die inhaltlich stärker mit dem Bereich der Telerobotik verknüpft sind.

3 Telemanipulations-Systeme

Die Hauptaufgabe dieser Arbeit ist die Entwicklung eines Telemanipulations-Systems. Viele dafür nötige Grundlagen sind bereits in dem vorangegangenen Kapitel beschrieben worden. Die bisher vorgestellten Grundlagen sind nicht nur für die Telemanipulation relevant. Sie finden ihre Anwendung auch in anderen Bereichen. In diesem Kapitel soll auf wichtige grundlegende Konzepte und Probleme der Telemanipulation in Forschung und Entwicklung eingegangen werden. Die Inhalte der folgenden Abschnitte geben jedoch nur die Grundlagen wieder, die noch nicht in anderen Abschnitten behandelt worden sind. Es werden hier also nicht alle Grundlagen von Telemanipulations-Systemen vorgestellt, wie der Titel dies vermuten lassen könnte.

In Anlehnung an die vorgestellte Unterscheidung der Begriffe Roboter und Manipulator aus Abschnitt 2.1 sollen in dieser Arbeit auch den Begriffen Telerobotik, Teleoperation und Telemanipulation unterschiedliche Bedeutungen zugewiesen werden. Es ist jedoch zu beachten, dass diese in der vorhandenen Literatur oft synonym verwendet werden. Innerhalb dieser Arbeit soll konsequent eine sinnvolle bedeutungstragende Unterscheidung dieser Begriffe eingehalten werden. Die Unterscheidung zwischen Telerobotik und Telemanipulation ist hierbei durch die Unterscheidung der Begriffe Roboter und Manipulator aus Abschnitt 2.1 geprägt. Telerobotik bezeichnet hier also die Steuerung eines Roboters aus der Entfernung, der möglicherweise mehrere Manipulatoren steuert. Telemanipulation und Teleoperation soll hingegen die Fernsteuerung eines Manipulators bezeichnen und damit die Manipulation der realen Welt betonen. Telerobotik bezeichnet zusätzlich auch ein wissenschaftliches Forschungsgebiet, zu dem auch das Gebiet der Telemanipulation gehört. Im Titel dieser Arbeit wird daher der Oberbegriff Telerobotik verwendet, um die Zuordnung zu themenverwandter Literatur zu ermöglichen. Durch diesem kurzen Begriffsklärungs-Exkurs soll nun erläutert werden, was genau mit Telemanipulation gemeint ist.

Telemanipulation ist ein Teilgebiet der Robotik. Bei einem Telemanipulations-System kontrolliert ein Mensch einen räumlich entfernten Manipulator. Diese Kontrolle dient einer gezielten Manipulation der realen Welt in der Umgebung des Manipulators. Menschen kommen dabei nicht mit dieser Umgebung in Berührung. Der Nutzen solcher Systeme ist, abgesehen von dem rein wissenschaftlichen Erkenntnisgewinn, die Möglichkeit Manipulationen in Bereichen durchzuführen, in denen Menschen nicht

selbst agieren können. Typische Vertreter solcher Bereiche sind in der Nuklearindustrie, in der Tiefsee und im Weltraum zu finden. Als Ursprung der Telemanipulation wird in vorhandener Literatur häufig die Forschung von Ray Goertz mit radioaktiven Materialien in den 1940er Jahren genannt [Ferre u. a. 2007, Hannaford 1989b, Furuta u. a. 1987, Yashiro und Ohnishi 2008]. Seit dem sind weltweit viele Telemanipulations-Systeme entwickelt worden und diesbezüglich viele Forschungsergebnisse veröffentlicht worden. Auch in anderen relevanten Forschungsgebieten, wie Digital-, Roboter-, Kommunikations-, Grafik- und Haptiktechnologie, sind seit dieser Zeit große Fortschritte gemacht worden. Diese Gebiete stellen Grundlagen für die Telemanipulation dar.

3.1 Bilaterale Telemanipulation

Ein Telemanipulations-System wird als bilateral bezeichnet, wenn die Steuerung des Eingabegerätes und des Manipulators bidirektional arbeitet. Der Manipulator verfolgt also nicht nur die Bewegung des Eingabegerätes, sondern die Bewegung des Eingabegerätes folgt wiederum auch der resultierenden Bewegung des Manipulators. Dadurch wird eine feste Kopplung zwischen Eingabe und Ausgabe erreicht, die gleichzeitig eine haptische Ausgabe erzeugt. Der Benutzer wird mit den kinästhetischen Informationen des Manipulators versorgt. Wenn vorhandene Drehmomentsensoren oder andere Kraftsensoren des Manipulators entsprechend einbezogen werden, können zusätzlich auch deren Werte am Eingabegerät ausgegeben werden. Auf diese Weise kann der Benutzer es auch spüren, wenn der Manipulator mit seiner Umgebung in Kontakt kommt oder Lasten bewegt. Bei der Wahl des Eingabegerätes ist prinzipiell zu beachten, welche Art der Ausgabe dem Benutzer geboten werden kann. Manipulatoren sind als Eingabegeräte geeignet. Bei der rückgekoppelten Ausgabe sind sie dazu geeignet die Position auszugeben. Es ist jedoch nicht jeder Manipulator auch dazu in der Lage gezielt Kräfte auszugeben. Haptische Geräte sind hingegen grundsätzlich nur dafür konzipiert Kräfte auszugeben. Eine Positionsausgabe muss also in eine entsprechende Ausgabe von Kräften überführt werden, die abhängig von der aktuellen Eingabeposition dieses Gerätes ist.

3.2 Transparenz und Telepräsenz

Die Bilateralität eines Telemanipulations-Systems soll es dem Benutzer erleichtern, geplante Manipulationsaufgaben durchzuführen [Handlykken und Turner 1980]. Die Übertragung von wirkenden Kräften und die kinestätische Kopplung zwischen dem

Manipulator und dem eigenen Arm soll unter idealen Bedingungen dafür sorgen, dass der Benutzer nicht mehr unterscheiden kann, ob er seinen Arm oder den künstlichen Arm benutzt. Den Manipulator zu bewegen soll sich wie die Bewegung des eigenen Armes anfühlen [Yokokohji und Yoshikawa 1990]. Ein derartiger Bewusstseinszustand, in dem ein Mensch das Gefühl hat, sein Bewusstsein seie mit einem fremden Körper verbunden, wird als Telepräsenz bezeichnet.

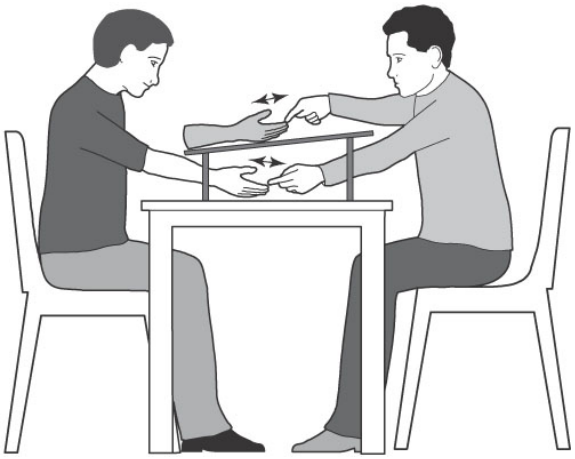


Abbildung 3.1: Skizzierter Aufbau des Gummihand-Experimentes. Die Grafik stammt aus [Ward 2008].

Schon durch das Vortäuschen einer Kombination aus wenigen Sinnesreizen kann das Gefühl einer solchen Telepräsenz erreicht werden. Ein sehr populäres Beispiel dafür ist die Gummihand-Illusion. Dabei wird eine Gummihand sichtbar vor den Augen der Versuchsperson platziert, wärend die echte Hand der Versuchsperson hinter einem Sichtschutz versteckt wird. Dann werden die Gummihand und die versteckte Hand mit identischen Bewegungen gestreichelt. Nach einigen Minuten beginnen die meisten Versuchspersonen die visuellen Reize mit den taktilen Reizen zu verknüpfen und bekommen das Gefühl, die Gummihand sei ihre eigene Hand [Ehrsson 2007]. Dieses Experiment zeigt die grundsätzlich vorhandene Möglichkeit von Telepräsenz-Empfindungen bei Menschen. Wird diese Erkenntnis auf die Telemanipulation übertragen, erscheint es möglich, das Telemanipulations-System für den Benutzer völlig transparent erscheinen zu lassen. Bei der Verwirklichung von Telemanipulations-Systemen ist ein hoher Grad an Transparenz ein wichtiges Ziel.

In [Yokokohji und Yoshikawa 1990] sind häufig verwendete Definitionen vorgestellt, die das Verhalten eines idealen Telemanipulations-Systems festlegen. Demnach müssen die Position und die Kraftausgabe des Eingabegerätes e und des Manipulators m zu jeder Zeit exakt gleich sein. Für die theoretische Analyse der Transparenz werden physikalische Größen, wie die Positionen $\vec{x}_e$ und $\vec{x}_m$, die Geschwindigkeiten $\vec{v}_e$

und $\vec{v}_m$ und die wirkende Kraft $\vec{F}_e$ und $\vec{F}_m$ von Eingabegerät e und Manipulator m, benutzt. Die Definition idealer Transparenz aus [Yokokohji und Yoshikawa 1990] fordert also $\vec{x}_e = \vec{x}_m \wedge \vec{F}_e = \vec{F}_m$. Eine andere Betrachtungsweise bezieht das dynamische Verhalten der mechanischen Komponenten des Systems ein. In der Praxis bieten pysikalische Körper einer auf sie wirkenden Kraft $\vec{F}$ einen unterschiedlich starken mechanischen Widerstand bei unterschiedlicher Frequenz f und Phasenlage φ von $\vec{F}$. Die mechanische Impedanz

$$\vec{Z}(f) = \frac{\vec{F}(f)}{\vec{v}(f)} e^{i(\varphi_F(f) - \varphi_v(f))} \tag{3.1}$$

ist bereits zur Modellierung einiger Telemanipulations-Architekturen verwendet worden, um den aus ihr resultierenden Effekten entgegen wirken zu können. Eine ideale Transparenz wird in einem solchen Modell durch die Forderung $\vec{Z}_e = \vec{Z}_m$ erzeugt [Ferre u. a. 2007, S. 166].

In der Praxis erweist es sich jedoch als unmöglich ein Telemanipulations-System mit dieser idealen Transparenz zu entwickeln. Die Zeitverzögerung bei der Datenübertragung und der Umsetzung von Bewegungen verringern die Transparenz. Viele Veröffentlichungen, wie [Ferre u. a. 2007, S. 191–209] und [Zhu und Salcudean 1995], beschäftigen sich mit der Analyse der Transparenz bei unterschiedlichen Systemarchitekturen. Nach diesen ist es zum Beispiel mit einer Vier-Kanal-Architektur mit gegenseitiger Impedanzkontrolle möglich ein hohes Maß an Transparenz zu gewährleisten. Dabei wird über vier Datenkanäle in beide Richtungen die Geschwindigkeit und die Kraft übertragen. Eine Zwei-Kanal-Architektur, bei der nur Positionsdaten übertragen werden, soll hingegen eine sehr geringe Transparenz bieten [Lawrence 1993]. Es gibt jedoch keine absoluten Bewertungskriterien für die Transparenz eines Telemanipulations-Systems. Relative Vergleiche und Vergleiche zur theoretischen idealen Transparenz werden stattdessen benutzt. Auch experimentelle Ergebnisse werden dafür analysiert. Doch wie zum Beispiel das Gummihand-Experiment zeigt, ist das Gehirn von unterschiedlichen Testpersonen unterschiedlich stark in der Lage, Unstimmigkeiten in den gebotenen Reizen zweckdienlich auszublenden. Einerseits setzt die Gummihand-Illusion nach unterschiedlich langer Zeit. Andererseits setzt sie bei einigen Menschen auch gar nicht ein.

3.3 Visuelle Unterstüzung

Der Gummihand-Effekt kann nur dadurch einsetzen, dass der Testperson stimmige taktile und visuelle Reize geboten werden. Bei Telerobotik-Systemen ist der mensch-

liche Benutzer meistens räumlich getrennt von der Umgebung, in der er mit dem Manipulator agiert. Dies bedeutet, dass er nicht die visuellen Reize geboten bekommt, die ohne die räumliche Trennung zur Verfügung ständen. Bei der Überbrückung von sehr kleinen räumlichen Abständen ist es teilweise möglich ein einfaches Sichtfenster zwischen dem Benutzerraum und dem Manipulatorraum einzurichten. Ist dies nicht möglich, werden Kamerasysteme und Bildschirme verwendet. Ohne die visuelle Unterstüzung wäre die Planung und Durchführung der meisten Telemanipulationen nicht möglich, da keine Übertragung von flächigen taktilen Reizen auf Hand und Fingern benutzt werden kann.

Eine weitere Möglichkeit der visuellen Unterstützung des Benutzers besteht in der Verwendung einer künstlichen Simulation des Manipulators. Bei der Simulation wird ein dynamisches Modell des Manipulators auf einem Bildschirm grafisch dargestellt. Für den Benutzer wird dadurch die Stellung des Manipulators deutlich. Die Simulation kann dabei an den aktuellen Zustand des Manipulators gekoppelt sein oder aber auch für sich allein nur als Simulation benutzt werden. Solche Verfahren werden als vorhersagende Anzeigen mit dem englischen Ausdruck Predictive-Display bezeichnet [Kosuge u. a. 2002]. Damit wird ihre besondere Bedeutung bei der Planung einer Telemanipulation hervorgehoben, bei der die reine Simulation genutzt wird, um vorherzusagen, wie sich die später ausgeführte reale Manipulation auswirkt. Zu diesem Zweck simulieren hoch entwickelte Systeme dieser Art nicht nur den Manipulator sondern auch dessen feste Umgebung und bewegliche Objekte in dieser Umgebung. Es wird eine virtuelle Realität erzeugt, die durch verschiedene Sensoren und den Manipulator mit der realen Welt verknüpft werden kann.

Ein Predictive-Display bringt auch bei der simultanen Benutzung des Manipulators den Vorteil, dass das Ergebnis einer geplanten Bewegung immer schon kurz vor deren Ausführung zu sehen ist. Übertragene Bilder einer Kamera sind garantiert immer erst kurz nach der Ausführung zu sehen. Der Benutzer kann also auf mögliche Fehleingaben schneller reagieren und die Sicherheit wird erhöht. Auch gewisse haptische Ausgaben können an das Predictive-Display gekoppelt werden und dadurch eine höhere Aktualität aufweisen als die realen Werte, die von dem Manipulator mit einer zeitlichen Verzögerung übertragen werden. Sind die aktuellen realen Werte des Manipulators übertragen worden, wird die Simulation sofort diesen Werten angeglichen. Die Simulation wirkt wieder direkt auf die haptische Ausgabe, also entsteht kein Nachteil gegenüber der Ausgabe ohne die Simulation. Für eine vorhergesagte kinästhetische haptische Ausgabe kann also die aktuelle Position im Modell benutzt werden.

3.4 Übertragung der Steuerungsdaten

Jedes Telemanipulations-System besitzt eine Komponente zur Übertragung von Daten zwischen der Bedienungseinheit (Master) und der Ausführungseinheit (Slave). Verschiedene Systeme unterscheiden sich dabei in der Übertragungsart und darin, welche Daten auf wie vielen Kanälen übertragen werden.

Für die Art der Kommunikation sind sehr viele Formen möglich. Die Daten können zum Beispiel per Kabel-, Glasfaser- oder Funkverbindung gesendet werden. Eine analoge Übertragung mit kontinuierlicher Kodierung auf eine physikalische Größe ist möglich und hat den Vorteil einer sehr hohen Übertragungs-Geschwindigkeit. Heutzutage ist die digitale Datenübertragung von sehr großer Bedeutung. Die Vorteile dieser Technik sind die rauschärmere Übertragung, durch Fehlerkorrektur, und Überbrückung größerer Distanzen. Zwischenstationen können das Signal durch Dekodierung und anschließende Rekodierung verlustfrei verstärken. Wobei immer auch zu bedenken ist, dass auch jede digitale Kommunikation letzten Endes trotzdem auf eine analoge Kodierung und Übertragung zurückgeführt werden muss. Dieses wird in entsprechender Hardware implementiert. Digitale Übertragung ist durch die aufwendigere Datenverarbeitung und die nötigen Umkodierungen jedoch langsamer als die rein analoge Übertragung.

Bei der digitalen Übertragung kann es sich um einen kontinuierlichen Datenstrom oder um eine diskrete Folge von einzelnen Datenpaketen handeln. Computernetzwerke benutzen paketorientierte Kommunikation. Bei Telemanipulations-Systemen, die ein Computernetzwerk benutzen, wird versucht einen kontinuierlichen Datenstrom zu immitieren, indem kleine Datenpakete sehr schnell nacheinander versendet werden. Die Verwendung dieser Methode ist bei der vorliegenden Arbeit vorgesehen.

Die Bezeichnung Kanal, wie in Übertragungskanal oder Datenkanal, wird in diesem Zusammenhang als Abstraktion einer physikalischen Datenleitung benutzt. Jeder Kanal soll dabei ausdrücken, dass die über ihn übertragenen Daten eine abweichende Bedeutung zu den Daten anderer Kanäle haben. Mehrere Datenkanäle müssen also getrennt von einander übertragen und verarbeitet werden. Bei jeder der zuvor vorgestellten Übertragungsformen kann jedoch durch eine geeignete Kodierung eine beliebige Anzahl an Datenkanälen über eine einzige physikalische Leitung transportiert werden.

Sehr charakteristisch für ein bilaterales Telemanipulations-System ist die Anzahl der Übertragungskanäle und deren Bedeutung. Hierdurch wird die grundlegende Architektur des Systems definiert. Die einfachste Architektur benutzt zwei Kanäle um die tatsächliche Manipulator-Position zum Master zu übertragen und die gewünschte

Manipulator-Position zurück zum Slave zu senden. Diese wird als Zwei-Kanal-Position-Position-Architektur bezeichnet [Ni und Wang 2002]. Da bei dieser Architektur keine Kraft- oder Drehmomentwerte übertragen werden, erscheint das System dem Benutzer nicht so transparent [Lawrence 1993]. Zu Transparenz siehe Abschnitt 3.2. Häufig wird die Zwei-Kanal-Kraft-Position-Architektur oder die Zwei-Kanal-Kraft-Geschwindigkeit-Architektur verwendet. Dabei werden die am Manipulator gemessenen Kräfte zur Benutzerseite übermittelt und die gewünschte Position oder Geschwindigkeit an die Manipulatorseite gesendet. Diese bietet ein höheres Maß an Transparenz, durch die direkt übermittelten Kraftwerte. Um die Eigenschaften verschiedener Architekturen besser vergleichen zu können, wird oft eine sehr allgemein formulierte Vier-Kanal-Architektur modelliert [Lawrence 1993]. Viele Architekturen können dabei durch einfache Anpassungen einiger Parameter modelliert werden [Ferre u. a. 2007, S. 170]. Bei der Vier-Kanal-Kraft-Geschwindigkeit-Architektur werden zum Beispiel Kräfte und Geschwindigkeiten in beide Richtungen übertragen. Wird hierbei der Geschwindigkeitskanal in Richtung zur Benutzerseite für eine dauerhafte Übertragung von Nullwerten konfiguriert, dann kann mit dem gleichen Modell eine Drei-Kanal-Architektur realisiert und analysiert werden.

3.4.1 Zeitverzögerung bei der Datenübertragung

Die verschiedenen Verbindungstypen haben sehr unterschiedliche Eigenschaften bezüglich der Übertragungsdauer $\mathrm{d}t = (t_e - t_s)$, wobei t_s und t_e die Sende- und Empfangszeitpunkte sind. Unterscheidungsmerkmale sind der Erwartungswert $E(\mathrm{d}t)$ und die Varianz $\mathrm{Var}(\mathrm{d}t)$ der Übertragungsdauer $\mathrm{d}t$ der Datenübertragungen einzelner Werte. Bei einer analogen Übertragung wird der Erwartungswert und die Varianz verschwindend gering ausfallen. Bei digitaler paketorientierter Übertragung im Netzwerk hingegen können beide Werte im Millisekunden-Bereich liegen oder bei ungünstigen Netzwerk-Protokollen oder -Strukturen sogar mehrere Sekunden groß sein.

3.5 Inverse Kinematik

Das in Abschnitt 2.2.3 erläuterte Problem der inversen Kinematik ist in der Telemanipulation relevant, wenn sich die kinematischen Ketten von Eingabegerät und Ausgabemanipulator unterscheiden. Sind die kinematischen Ketten identisch, kann die Datenübertragung und Synchronisation der Manipulatorstellung komplett im Gelenkwinkelraum gelöst werden. Bei heutigen Systemen ist dies jedoch selten der Fall, da häufig kleinere haptische Geräte zur Eingabe verwendet werden. Die Gelenkwin-

kelstellungen des Eingabegerätes müssen dann mit der direkten Kinematik in kartesische Koordinaten umgerechnet werden. Aus diesen Koordinaten müssen danach mit der inversen Kinematik des Manipulators mögliche Gelenkstellungen berechnet werden. Bei einem bilateralen System müssen die entsprechenden Berechnungen für den Rückweg in gleicher Weise durchgeführt werden. Jede dieser Berechnungen führt zu einer Zeitverzögerung der Datenübertragung. Bei der praktischen Implementierung ist zu berücksichtigen welche kinematischen Berechnungen an welcher Stelle im System sinnvoll sind. Meistens bieten die Programmier-Bibliotheken, mit denen die Hardware kontrolliert wird, effiziente Funktionen für die direkte und inverse Kinematik. Bei der Verwendung dieser Bibliotheken verliert das System selbst jedoch im allgemeinen die direkte Kontrolle über die Berechnung der inversen Kinematik. Die auftretenden Probleme, wie mehrfache oder nicht vorhandene Lösungen, können nicht unbedingt behandelt werden.

3.6 Regelschleife

Die übertragenen Daten werden auf der Manipulatorseite überwiegend in den Gelenkwinkelraum umgerechnet und an die Hardware-Steuerung weitergeleitet. Das Hardware-Kontrollsystem eines Manipulators ist meistens als geschlossene Regelschleife realisiert. Eine Trajektorie wird als Folge gewünschter Werte im Gelenkwinkelraum eingegeben. Dieses sind die Soll-werte. Daraus wird mathematisch die Folge der benötigten Drehmomente der einzelnen Gelenkmotoren berechnet. Die Regelschleife wird geschlossen, indem Sensorwerte der einzelnen Gelenke, die Ist-werte, wieder an das Kontrollsystem zurückgeleitet werden, um korrigierend auf die nachfolgend berechneten Ausgabewerte zu wirken [Craig 2005]. Dieser motorische Regelkreislauf wird als Servo bezeichnet. Überwiegend werden dabei PD- und PID-Regler verwendet. Sowohl für die Eingabewerte als auch für die Sensorwerte können verschiedene Kombinationen von Winkelpositionen, -geschwindigkeiten, -beschleunigungen oder Drehmomenten zum Einsatz kommen. Die komplette Regelschleife ist ein System aus digitalen und analogen elektrischen Teilen.

Diese Regelschleife ist überwiegend hierarchisch gegliedert und auf die Software und die Hardware verteilt. Dabei werden die Ausgabewerte jeder Schicht der Hierarchie zeitlich immer stärker aufgelöst, um letztendlich einen sehr glatten Bewegungsverlauf zu erhalten.

Nach den Ausführungen in Abschnitt 2.3 wird bei haptischen Geräten und Manipulatoren eine vergleichbare technische Umsetzung benutzt. Darum kann diese allgemeine Form der Regelschleife eines Manipulators auch für ein haptisches Gerät verwen-

det werden. Auch ein komplettes Telemanipulations-System kann auf einem höheren Abstraktionsgrad als eine Regelschleife betrachtet werden. Handelt es sich um ein bilaterales Telemanipulations-System, ergibt sich eine geschlossene Regelschleife.

Für theoretische Analysen von Telemanipulations-Systemen werden häufig analog aufgebaute elektrische Schaltungen modelliert. Blockdiagramme werden benutzt, um die Verbindungen einzelner Systemkomponenten darzustellen. Die physikalischen Größen Kraft und Geschwindigkeit werden dabei als Spannung und Strom uminterpretiert. Auf diese Weise können die vorhandenen Methoden der Elektrotechnik und Systemtheorie verwendet werden, um Aussagen über das resultierende Systemverhalten zu machen.

3.7 Stabilität

Die beiderseitige Kontrolle eines bilateralen Telemanipulators entspricht einem rückgekoppelten System. Die Systemtheorie definiert ein System nur dann als stabil, wenn es auf jede beschränkte Eingangsfunktion mit einer beschränkten Ausgangsfunktion reagiert [Girod u. a. 1997]. Bei der verwendeten Eingabemethode kann der Benutzer innerhalb der Zeit seines Lebens niemals eine unbeschränkte Eingabe erzeugen, darum sollten auch die resultierenden Steuerungswerte für den Manipulator beschränkt sein. Die Auswirkungen von unstabilen Ausgabewerten würden das System unkontrollierbar machen und die Sicherheit von Mensch und Maschine gefährden.

Die Unstabilität des Telemanipulations-Systems entsteht durch das dynamische Verhalten des Systems und einzelner Komponenten. Häufig gibt es bei rückgekoppelten Komponenten charakteristische Eigenresonanzfrequenzen, die das System destabilisieren. Nach der Passivitäts-Theorie für Telemanipulations-Systeme [Anderson und Spong 1989] darf keine der Komponenten des Systems die Energie erhöhen, die im System vorhanden ist. Ist dies der Fall, erhöht sich auch die Energie des gesamten Systems nicht und es ist stabil. Aufbauend auf diesem Konzept ist in [Niemeyer und Slotine 1991] eine Systemarchitektur für den Kommunikationsblock vorgestellt worden, die diesen energieerhaltend stabilisiert. Die zu übertragenden Steuerungsdaten, wie Positionen, Geschwindigkeiten und Kräfte, werden zu diesem Zweck durch eine lineare Transformation in sogenannte Wave-Variables transformiert. Dieses Vorgehen ähnelt der Übertragung der Daten im Spektralbereich. Diese Methode stabilisiert das komplette Telemanipulations-System bei beliebig großen konstanten Signallaufzeiten [Ferre u. a. 2007, S. 168–172]. Für variierende Signallaufzeiten, wie sie bei Computernetzwerken vorkommen, kann diese Methode jedoch nicht verwendet werden. Für dieses Problem gibt es noch keine allgemein verwendbare Lösung. Die hohe Komplexi-

tät des gesamten Systems macht eine sinnvolle, systematische Analyse, die rein theoretischer Natur ist, unmöglich. Die variierenden Zeitverzögerungen entstehen in der Praxis nicht nur bei der Netzwerkübertragung, sondern bei allen durchzuführenden Berechnungen, wie die der inversen Kinematik. Ein bilaterales Telemanipulations-System könnte als ein digitales IIR-Filter angesehen werden, bei dem eine hohe Zahl an variablen Parametern dafür sorgt, dass es keine konstante Impulsantwort aufweist. Es wäre vermutlich möglich, experimentell problematische Frequenzbereiche zu identifizieren. Durch geeignete digitale Filter, wie Bandpässe und Bandsperren, könnten diese dann gedämpft werden. Dabei müsste jedoch untersucht werden, welche Auswirkungen dies auf die Transparenzwirkung des Systems hat. Eine umfangreiche Untersuchung dieser Art würde jedoch den Rahmen dieser Arbeit übersteigen.

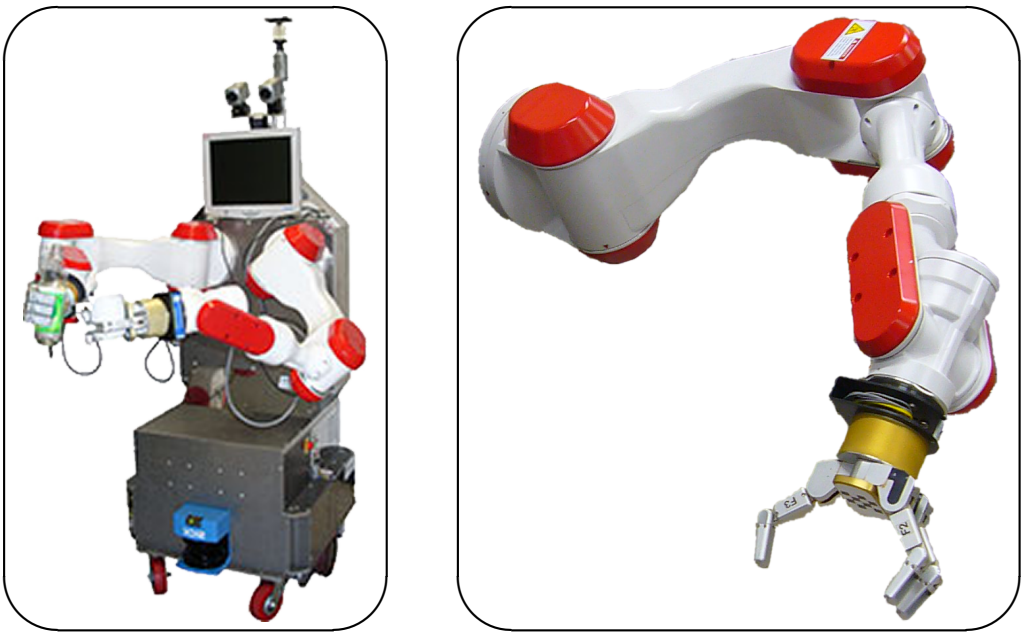
Aus [Hannaford 1989b] geht die wichtige allgemeine Erkenntnis hervor, dass jede Stabilisierung des Systems zu einem Verlust an Genauigkeit führt. Umgekehrt wirkt eine Steigerung der Genauigkeit eines bilateralen Telemanipulators wiederum destabilisierend auf das System. Bei der praktischen Realisierung muss also immer ein Kompromiss zwischen Stabilität und Genauigkeit gewählt werden. Die Genauigkeit des Systems korrespondiert dabei mit dem Begriff der Transparenz (vergleiche Abschnitt 3.2), da eine höhere Genauigkeit in der Umsetzung der eingegebenen Bewegungen zu einer höheren Transparenz des Systems führt. Der in [Hannaford 1989b] beschriebene nötige Kompromiss besteht also ebenso zwischen Stabilität und Transparenz eines Telemanipulations-Systems.

3.8 Zusammenfassung

In diesem Kaptiel sind die theoretischen technischen Grundlagen für die Umsetzung eines modernen Telemanipulations-Systems vorgestellt worden. Es gibt viele Modelle und Architekturen zur Realisierung und Analyse solcher Systeme. Die Grundkomponenten der Hardwaresteuerung, Netzwerkkommunikation und visuellen Unterstützung sind vorgestellt worden. Die beiden Hauptschwerpunkte aktueller Telemanipulations-Forschung sind Stabilität und Transparenz bei Datenübertragung mit variierender Zeitverzögerung.

4 Vorhandene Umgebung

In diesem Kapitel soll die vorhandene Umgebung beschrieben werden, die für die Umsetzung dieser Arbeit benutzt worden ist. Es handelt sich dabei um die, für diese Arbeit relevante, Hardware und die eingerichtete Software des Arbeitsbereiches Technische Aspekte Multimodaler Systeme, kurz TAMS, des Fachbereichs Informatik der Universität Hamburg.



(a) Service-Roboter TASER. (b) Manipulator PA10-6C mit Barrett Hand.

Abbildung 4.1: Service-Roboter Hardware des Arbeitsbereiches TAMS.

4.1 Mobiler Service-Roboter

Abbildung 4.1a zeigt den Service-Roboter TASER. Alle Hardware-Komponenten des TASERs sind auf der mobilen Plattform MP-L655 der Firma Neobotix aufgebaut. Zu

diesen Komponenten gehören unter anderem ein Industrie-Computer, zwei Manipulatoren mit Greifern, verschiedene Kamerasysteme und Laser-Abstandsensoren. Der benutzte Industrie-Computer mit einem Pentium IV Hauptprozessor mit 2,4 Gigahertz wird mit einem Linux-Betriebssystem betrieben. Über verschiedene Schnittstellen können alle Komponenten des Roboters angesteuert und programmiert werden.

4.1.1 Manipulator

Als Manipulatoren stehen dem TASER zwei PA10-6C, kurz PA10, der Firma Mitsubishi Heavy Industries zur Verfügung. Der PA10 hat sechs Freiheitsgrade und einen Aktionsradius von etwa einem Meter, ähnlich dem menschlichen Arm. Genaure Angaben zu den Ausmaßen und Gelenken des Manipulators sind durch die Denavit-Hartenberg-Parameter in Abbildung 2.4 gegeben. Der PA10 besitzt eine Traglast von zehn Kilogramm und kann mit Standard-Industrie-Computern betrieben werden. Auf dem Computer des TASER ist zu diesem Zweck die Programmier-Bibliothek Multi-RCCL von John Lloyd [Lloyd und Hayward 1995] eingerichtet. Sie bietet umfangreiche Funktionen, um mehrere Manipulatoren mit der Programmiersprache C++zu steuern. Abbildung 4.1b zeigt ein aktuelles Foto des linken Manipulators des TASERs und in Tabelle 4.1 sind dessen Grenzwerte für die Gelenkwinkel und deren Geschwindigkeiten ausgeführt.

Tabelle 4.1: Grenzwerte der Gelenke des PA10-6C. Die Grenzen sind mechanisch und durch die Software gegeben.

Gelenk Nummer	minimale und maximale Grenzen der Gelenkwinkel [°]	maximale Winkelgeschwindigkeiten [°/s]
1	-177 bis $+177$	$180/\pi \approx 57$
2	-64 bis $+124$	$180/\pi \approx 57$
3	-107 bis $+158$	$360/\pi \approx 115$
4	-255 bis $+255$	360
5	-165 bis $+165$	360
6	-255 bis $+255$	360

4.1.2 Greifer

Als Endeffektor ist am Ende jedes Manipulators die Drei-Finger-Hand der BH8-262 der Firma Barrett Technology angebracht. Diese Barrett-Hand hat vier Freiheitsgrade, jeweils einen für das Strecken und Schließen jedes Fingers und einen, mit dem die Orientierung zweier Finger verändert werden kann. Jeder der drei Finger hat zusätzlich einen integrierten Kraftsensor. Für die objektorientierte Programmierung der Barrett-Hand in C++ stehen Programm-Bibliotheken von Bernd Roessler und Tim Baier-Löwenstein zur Verfügung. Diese sind im Arbeitsbereich TAMS entwickelt worden. Details hierzu sind zum Beispiel [Baier u. a. 2006] zu entnehmen.

4.2 Computer-Netzwerk

Der Computer des TASERs ist durch eine drahtlose Verbindung in das Computer-Netzwerk des Arbeitsbereiches eingebunden. Dabei handelt es sich um eine 54Mbit WLAN-Verbindung und ein Netzwerk, dass die Internet-Protokoll-Familie unterstützt. Mehrere Arbeitsplatz-Computer sind ebenfalls an dieses Netzwerk angeschlossen. Über das Internet können auch Computer außerhalb des Arbeitsbereiches in das Netzwerk integriert werden.

4.3 Haptisches Eingabegerät

Abbildung 4.2: PHANTOM Desktop, ein haptisches Eingabegerät.

Als haptisches Gerät steht ein PHANTOM Desktop der Firma SensAble Technologies zur Verfügung. Abbildung 4.2 zeigt das Gerät in Benutzung. Damit können Raum-Positionen und Orientierungen mit sechs Freiheitsgraden eingegeben werden. Der Arbeitsbereich liegt dabei innerhalb eines Radius von sechs Zentimetern mit einer räumlichen Auflösung von durchschnittlich 433 Einheiten pro Millimeter. Die haptische Ausgabe von Kräften ist auf nur drei Freiheitsgrade beschränkt. Damit können Kräfte nur in Richtung der Raumachsen ausgegeben werden und keine Rotationskräfte auf die Orientierung, in Form von Drehmomenten. Zur Einbindung des Gerätes in eigene Programme kann die herstellereigene Programm-Bibliothek OpenHaptics verwendet werden.

4.4 Konkretisierung dieser Arbeit

In diesem Kapitel sind die Hardware-Komponenten vorgestellt worden, die in das Telemanipulations-System integriert werden sollen. Im Rahmen dieser Arbeit wird also ein manuelles Steuerungssystem für den Manipulator des beschriebenen Service-Roboters entwickelt. Die Steuerung soll eine intuitive und interaktive Nutzung ermöglichen, indem die Eingabe des beschriebenen haptischen Gerätes möglichst unmittelbar ausgeführt wird. Das vorhandene Computernetzwerk soll für die Übertragung der Steuerungsdaten genutzt werden. Die direkte Hardwaresteuerung über die externen Schnittstellen und die Datenbus-Systeme sind nicht Teil dieser Arbeit. Es werden die vorhandenen, bereits installierten Treiber und Programmier-Schnittstellen aller Geräte benutzt. Die Verwendung beider Manipulatoren des Roboters ist nicht vorgesehen. Das zu entwickelnde System wird nur die Steuerung eines Manipulators zur Zeit realisieren. Auch die Implementation unterschiedlicher Arten von Greifbewegungen übersteigt, bei der verwendeten Drei-Finger-Hand, den Rahmen dieser Arbeit. Verschiedene Greiftechniken werden also nicht behandelt und bei der praktischen Realisierung wird nur ein einfaches Öffnen und Schließen der Hand umgesetzt. Der Inhalt dieser Arbeit konzentriert sich auf die Bewegungssteuerung eines Manipulators, die eine ausreichende Menge an zu lösenden Problemen beinhaltet.

5 Software-Lösung

In diesem Kapitel wird das entwickelte Manipulator-Steuerungssystem vorgestellt. Begonnen wird mit der zugrunde liegenden Software-Architekur. Es werden auch die Entscheidungen für die verschiedenen, genutzten Betriebssysteme und Programmiersprachen erläutert. Darauf folgt eine Beschreibung der Architektur einzelner Komponenten. Hierbei wird auch die Funktionsweise dieser Komponenten und des gesamten Systems detailierter beschrieben.

5.1 Verteiltes System

Bei dem System handelt es sich um eine verteilte Anwendung. Ein Teil der Anwendung wird auf dem mobilen Roboter ausgeführt. Der zweite Teil läuft auf einem anderen Computer, an dem der Operator arbeitet. Beide Teile kommunizieren miteinander über das Computernetzwerk mit dem Internet-Protokoll. Es ist also im Folgenden zu klären, wie diese Verbindung am besten implementiert werden kann und welcher Teil der Anwendung welche Aufgaben zugeteilt bekommt.

Auf dem Roboter ist zur Zeit kein System zur Telemanipulation installiert. Damit ist hier gemeint, dass keine Funktionalität geboten wird, um den Manipulator von anderen Computern im Netzwerk aus, in Echtzeit zu steuern. Es ist möglich, über eine Secure Shell (SSH) Verbindung einzelne Manipulator-Steuerbefehle nacheinander auf dem Roboter auszuführen. Bei der angestrebten Echtzeit-Anwendung müssten diese Befehle jedoch in einer sehr hohen Rate von 50 bis 100 Hertz ausgeführt werden, um eine stetig wirkende Armbewegung zu erhalten [Craig 2005]. An den Rückkanal werden sogar noch höhere Anforderungen gestellt. Von dem Roboter muss die aktuelle Position des Manipulators und die aktuellen Werte der Kraftsensoren mit einer Rate von etwa 1000 Hertz an das haptische Gerät übertragen werden. Da die SSH-Verbindung das Transmission Control Protocol (TCP/IP) benutzt, welches zu langsam ist, können die nötigen Datenübertragungen nicht hierüber realisiert werden. Derartige Übertragungsraten lassen sich in dem vorhandenen Computer-Netzwerk mit dem User Datagramm Protocol (UDP/IP) realisieren. Jedoch ist die Höhe dieser

Rate nicht sichergestellt, wenn das Netzwerk gleichzeitig von anderen Anwendungen genutzt wird.

Es ist also notwendig eine Serveranwendung auf dem Roboter auszuführen, die erstens die UDP/IP-Übertragung der Daten und Befehle steuert. Zweitens muss dieser Telemanipulations-Server auch entstehende Probleme mit den variierenden Übertragungsdauern und -raten lösen. Die Benutzung dieser Struktur hat darüber hinaus den Vorteil, dass sich mehrere Clientanwendungen parallel mit dem Server verbinden können, ohne die Netzwerk-Auslastung zu erhöhen. Dafür muss der Server die Daten mit Multicast-UDP/IP-Paketen an mehrere Computer im Netzwerk gleichzeitig senden. So wäre es zum Beispiel möglich, eine Telemanipulation von einem Arbeitsplatz zu steuern und sie von anderen Arbeitsplätzen aus zu überwachen.

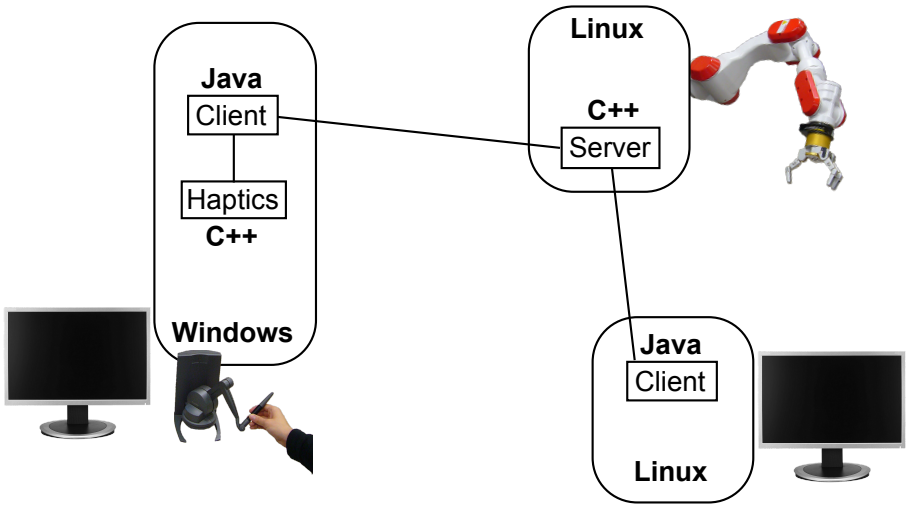


Abbildung 5.1: Grobe Struktur des geplanten Telemanipulations-Systems.

5.1.1 Programmiersprachen

Die beiden Teile der Anwendung werden auf verschiedenen Betriebssystemen ausgeführt. Auf dem Roboter wird ein Linux-System genutzt und der Operator-Arbeitsplatz mit dem haptischen Gerät wird mit Microsoft Windows XP betrieben. Bei den Programmier-Schnitstellen für den Manipulator und das Eingabegerät handelt es sich auf beiden Seiten um C++-Bibliotheken mit einbindbaren Headerfiles. Durch diese Begebenheit erscheint es offensichtlich sinnvoll, eine Serveranwendung in C++ für das Linux-System und eine Clientanwendung in C++ für Windows XP zu programmieren. Der Server stellt dabei eine Art Hintergrundprozess dar, der keine Bildschirmausgaben und keine direkte Benutzereingabe nutzt. Der Client hingegen soll eine

grafische Bedienoberfläche anbieten, um für den Benutzer vielseitige Ausgaben und Eingaben zu ermöglichen.

Zusätzlich soll es jedoch ermöglicht werden, das Telemanipulations-System auch von anderen Arbeitsplätzen aus zu überwachen, auf denen ein Linux-Betriebssystem benutzt wird. Diese Überlegung führt zu der Idee die grafische Bedienoberfläche plattformübergreifend in Java zu programmieren. Diese Java-Clientanwendung übernimmt allgemein die Kommunikation mit dem Server und bietet grafische Ausgaben und alternative Eingabemöglichkeiten. Wird sie auf dem Computer mit dem haptischen Gerät ausgeführt, nutzt sie eine Verbindung mit einer C++-Anwendung, die das haptische Gerät nutzbar macht. Durch diese Struktur ist das System sehr flexibel einsetzbar und es ist keine mehrfache Programmierung der Clientanwendung für verschiedene Betriebssystemen nötig. Abbildung 5.1 verdeutlicht diese geplante Struktur.

Alle Teile des Systems haben verschiedene Aufgaben parallel zu erledigen. Beispiele hierfür wären die softwareseitige Regelschleife des Manipulators und die parallel laufende Netzwerkkommunikation. Die nötigen Aufgaben werden auf verschiedene Programm-Threads aufgeteilt.

5.2 Telemanipulations-Server

Der Telemanipulations-Server ist eine Anwendung, die direkt auf dem Roboter ausgeführt werden soll. Dadurch kann die möglichst uneingeschränkte Kontrolle über die Roboter-Hardware gewärleistet werden. Auch können bestimmte Bedienfehler oder Notfallsituationen vom Server behandelt werden, auch wenn die Netzwerkverbindung ausgefallen ist. Dennoch soll der Server einen möglichst geringen Anteil an der zur Verfügung stehenden Rechenzeit des Roboter-Computers ausnutzen, um andere parallel zu ihm ausgeführte Prozesse nicht zu stören.

5.2.1 Aufgaben des Servers

Im Folgenden sollen die Aufgaben des Servers festgelegt werden. Dafür sind die folgenden Notationen nützlich. Der aktuelle Zustand der Hardware soll dabei mit $\vec{s}_t = (a_t, b_t)^{\mathrm{T}}$ bezeichnet werden. Dies ist der Vektor, der die Werte der Gelenkwinkel $a_t = (a_1,a_2,a_3,\ldots,a_m)_t^{\mathrm{T}}$ und die Werte der Kraftsensoren $b_t = (b_1,b_2,b_3,\ldots,b_n)_t^{\mathrm{T}}$ zum Zeitpunkt t enthält. Bei dem hier verwendeten Arm mit sechs Freiheitsgraden, einem Greifer mit vier Freiheitsgraden und drei Kraftsensoren in den Fingern, ist also $m = 10$ und $n = 3$. Der Vektor $\vec{s}_t$ hat dabei also 13 Elemente. Fordert ein Client

eine Änderung dieses Hardware-Zustandes, dann sendet er den Vektor mit den gewünschten Gelenkwinkel-Werten $\vec{c}_t$ als Steuerbefehl. Die Aufgaben des Servers sind in Abbildung 5.2 aufgeführt.

Der Server hat die Aufgaben:

- warten auf eingehende Clientverbindungen,
- bei Verbindung,
  - Verbindung überwachen,
  - $\vec{s}_t$ gewinnen,
  - $\vec{s}_t$ senden,
  - empfangene $\vec{c}_t$ überprüfen und
  - empfangene $\vec{c}_t$ an die Hardwaresteuerung weiterleiten.

Abbildung 5.2: Aufgaben des Servers

Die Überprüfung der empfangenen Befehle $\vec{c}_t$ soll sehr einfach implementiert sein, um die Roboter-Resourcen zu schonen aber dennoch einen gewissen Schutz vor Fehleingaben oder Fehlübertragungen zu gewährleisten. Die in UDP/IP verwendete Prüfsumme über jedes Datenpaket gewährleistet bereits in hohem Maße, dass nur korrekt übertragene Daten weiter verarbeitet werden. Jedes Paket soll zusätzlich mit einem Zeitstempel versehen werden. Dadurch können Pakate, die in falscher Reihenfolge empfangen worden sind, wieder geordnet werden. Zu alte Pakete sollten gar nicht mehr verarbeitet werden, da das komplette System gewissen Echtzeitanforderungen unterliegen soll. Das bedeutet, dass alle Befehle innerhalb eines festgelegten Zeitintervalles verarbeitet werden müssen. Zusätzlich wird überprüft, ob der Inhalt eines Datenpaketes in sich stimmig ist, damit nicht irgendwelche fremden, zufällig empfangenen UDP/IP-Pakete, als Steuerungsbefehle interpretiert werden. Zu diesem Zweck enthält jedes Paket einen Befehlstyp, eine Manipulatornummer und einen Längenwert, der mit der tatsächlichen Länge des Paketes verglichen werden kann.

5.2.2 Schnelle Datenübertragung

Die relevanten Daten, die über das Netzwerk übertragen werden müssen, sind die Vektoren $\vec{s}_t$ und $\vec{c}_t$. Hinzu kommen Daten zur Steuerung der Verbindung, die hier jedoch vernachlässigt werden können, da sie nicht so strengen zeitlichen Kriterien unterliegen wie $\vec{s}_t$. Der aktuelle Hardware-Zustand $\vec{s}_t$ muss kontinuierlich mit einer

sehr hohen Rate gesendet werden, um die Ausgaberate des haptischen Gerätes zu bedienen. Nach [You und Sung 2008, OpenHaptics 2004] sollte diese Rate im günstigsten Fall bei etwa 1000 Hertz liegen. Für derart hohe Übertragungsraten ist in einem Internet-Protokoll-Netzwerk das User Datagram Protocol (UDP/IP) [Postel 1980] vorgesehen. Bei ihm werden für die Datenübertragung nur die nötigsten Konzepte verwendet. Dadurch kann mit UDP/IP eine viel höhere Datenrate als mit anderen Protokollen, wie zum Beispiel TCP/IP, erreicht werden. Wenn sich mehrere Clients mit dem Server verbinden, dann müssen sie alle die gleichen Daten empfangen. Darum können diese mit einem UDP/IP Multicast versendet werden. So kann ein gesendetes Datenpaket von allen verbundenen Clients gelesen werden. Dies verringert die Netzwerklast und die benötigte Rechenzeit auf dem Roboter. Das Protokoll UDP/IP wird in anderen Systemen bereits erfolgreich für die Übertragung von Video und Audiodaten verwendet. Dabei kommen immer starke Kompressionsverfahren zum Einsatz. Auch für Anwendungen in der Telerobotik wird es bereits verwendet. In [Wirz u. a. 2008] wird ein neues Internet-Übertragungs-Protokoll namens Efficient Transport Protocol (ETP/IP) vorgestellt, dass speziell für haptische Anwendungen entwickelt worden ist. Dessen Hauptverbesserung gegenüber UDP/IP ist die automatische Anpassung der Senderate an die aktuelle Netzwerklast. Auch wenn dieser Ansatz sehr sinnvoll ist, wird es in dieser Arbeit nicht eingesetzt. Die in [Wirz u. a. 2008] experimentell gezeigten Vorteile erscheinen zu gering, um den verhältnismäßig großen Aufwand für die Implementation zu rechtfertigen. UDP/IP ist hingegen bei den verwendeten Betriebssystemen bereits implementiert. Bei dem Telemanpulations-System geht es darum, die $\vec{s}_t$ so schnell wie möglich an den Client zu übertragen, damit die dort empfangenen Daten so aktuell wie möglich sind. Aus diesem Grund erweist sich auch die in [You und Sung 2008] beschriebene Methode zur Kompression von haptischen Übertragungsdaten als unpraktisch, da sie den Rechenaufwand beim senden der Daten erhöht, was wiederum eine Verzögerung bedeutet.

Für das Telemanipulations-System wird ein sehr einfaches Übertragungs-Protokoll benutzt, dass auf UDP/IP aufsetzt und wie folgt strukturiert ist. Der Server sendet in möglichst kurzen Abständen den aktuellen Zustand der Hardware in einem $\vec{s}_t$-Datenpaket. Jedes dieser $\vec{s}_t$-Datenpakete beginnt mit zwei Byte-Werten, die den Typ des Paktetes identifizieren und die Nummer des betroffenen Manipulators angeben. Darauf folgen drei ganzzahlige Werte. Der erste enthält die Anzahl der nachfolgenden Nutzwerte. Im zweiten und dritten Wert steht der Sendezeitpunkt des Paketes in Sekunden und Mikrosekunden seit dem 1. Januar 1970 um 0:00 Uhr. Derartige Zeitangaben sind in der Programmierung üblich. Falls Pakete sich im Netzwerk überholen, hat der Empfänger also die Möglichkeit dieses festzustellen, damit sie nicht in der falschen Reihenfolge abgearbeitet werden. Darüber hinaus kann, mit einer vorheri-

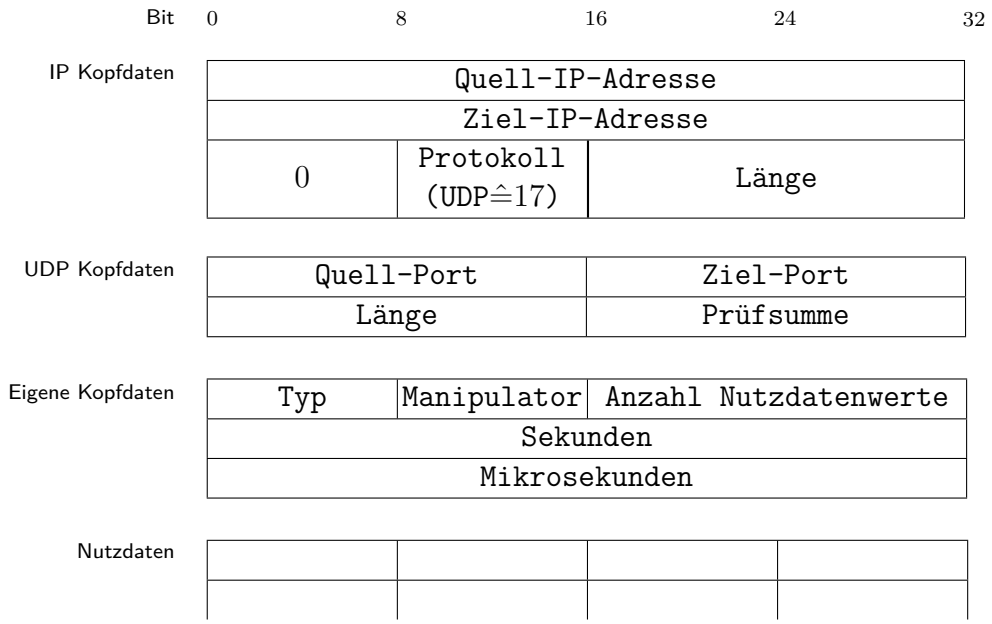


Abbildung 5.3: Datenpaket-Struktur für das Netzwerk-Protokoll des Telemanipulations-Systems.

gen Synchronisation der Uhren beider Computer, die Übertragungsgeschwindigkeit gemessen werden. Darauf folgen die Nutzdaten mit 4-Byte-Gleitkomma-Zahlen, in denen sich die Daten aus $\vec{s}_t$ befinden. Ein derartiges Datenpaket lässt sich von einem nativen Programm schnell erzeugen. Die entsprechenden Variablen können komplett in die zu sendende Datenstruktur kopiert werden und müssen weder beim Sender noch beim Empfänger umcodiert werden. Auf diese Weise ist es möglich, die $\vec{s}_t$ in den benötigten Intervallen zu übertragen. Die Anzahl der übertragenen $\vec{s}_t$ Vektoren pro Sekunde könnte hierbei erhöht werden, indem mehr als nur ein Vektor pro Paket gesendet wird. Doch in der Client-Anwendung wäre jeweils nur der zeitlich neueste Wert von Nutzen, da die Werte so aktuell wie möglich sein müssen. Dadurch kann also keine Verbesserung erreicht werden.

Die hier vorgestellte Struktur dieser Datenpakete ist trotz ihrer Einfachheit flexibel genug, um auch für andere benötigte Datenübertragungen des Systemes benutzt zu werden. Die Befehlsvektoren $\vec{c}_t$, mit denen die Client-Anwendung den Manipulator steuert, werden daher mit dem selben Protokoll übertragen. Das erste Byte ordnet jedem Datenpaket seinen Zweck zu. Abbildung 5.3 veranschaulicht die komplette Struktur der verwendeten Datenpakete.

5.2.3 Architektur des Servers

Nach diesen Vorüberlegungen kann eine Software-Struktur des Servers direkt aus der Liste seiner Aufgaben abgeleitet werden. In Abbildung 5.4 sind die Komponenten des Servers und ihre Beziehungen dargestellt. Das Programm startet mit dem zentralen Thread Server, der zunächst nötige Initialisierungen der Hardware durchführt. Danach erzeugt er vier neue Programm-Threads und kümmert sich selbst nur noch darum, periodisch die Daten der Hardware $\vec{s}_t$ zu lesen, und gegebenenfalls Benutzereingaben der Tastatur zu verarbeiten oder Logging-Ausgaben zu produzieren. Der erste neu erzeugte Thread, namens ConnectionWatcher, wartet auf eingehende Verbindungsanfragen von Clients auf einem festgelegten TCP/IP-Port. Für jede aufgebaute Verbindung wird ein neuer Connection-Thread erzeugt, der diese Verbindung aufrecht erhält und verwaltet. Der zweite erzeugte Thread, names CommandReceiver, verarbeitet eingehende UDP/IP-Pakete von Client-Anwendungen. Falls ein solches Paket einen Befehl zur Steuerung von Arm oder Hand enthält, wird er hier, wie in Abschnitt 5.2.1 beschrieben, auf seine Korrektheit und Gültigkeit überprüft. Jeder neue gültige Befehl überschreibt den Puffer des zuvor gespeicherten Befehls, auch wenn der alte Befehl noch nicht ausgeführt worden ist.

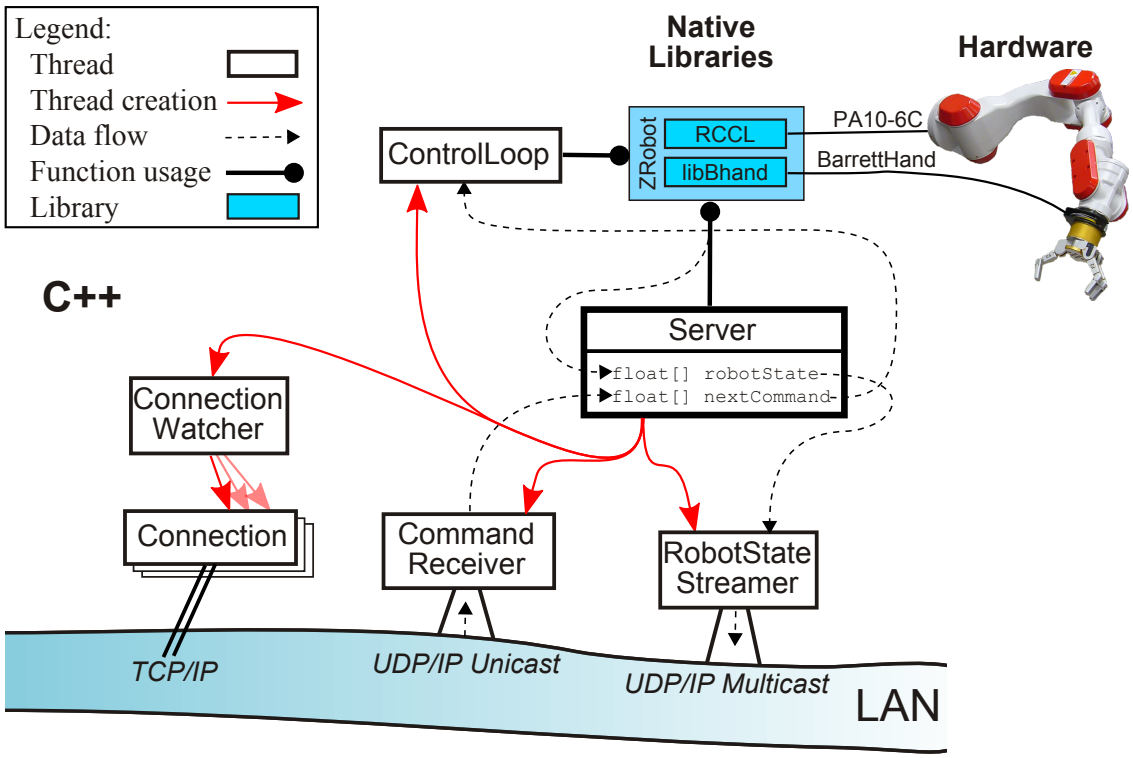


Abbildung 5.4: Vereinfachtes Schema des Telemanipulations-Servers.

Der dritte erzeugte Thread, der RobotStateStreamer, übernimmt die Aufgabe, die aktuellen Daten der Hardware $\vec{s}_t$ fortlaufend an alle Clients per UDP/IP-Multicast zu versenden. Durch den Multicast können die versandten Pakete prinzipiell von allen Programmen, die Zuriff auf das Netzwerk haben, empfangen werden. Der letzte neu erzeugte ControlLoop-Thread implementiert eine Regelschleife, die die aktuell auszuführenden Befehle an die Hardware weiterleitet.

Der Server ist in C++ programmiert und mit GNU-Compiler-Collection (GCC) für Linux kompiliert. Damit kann er als natives Programm ohne zusätzliche Laufzeitumgebungen ausgeführt werden. Zudem sind auf dem Roboter-Computer bereits C++-Programm-Bibliotheken, zur Kontrolle der Hardware von Arm und Hand, installiert. Diese sind so eingerichtet, dass sie in ein C++-Programm einkompiliert werden können.

5.2.4 Hardware-Steuerung

Die Hardware wird über die Programmier-Bibliotheken Multi-RCCL und libBhand gesteuert. Diese stellen Funktionen bereit, mit denen der Arm und die Hand des Roboters kontrolliert werden können. RCCL ist die Abkürzung für Robot Control C Library. Dabei handelt es sich um eine portable Sammlung von Funktionen und Datenstrukturen für die Programmierung von Manipulatoren unter Unix-Betriebssystemen [Lloyd und Hayward 1995, Hayward und Paul 1984]. Diese sind in den 1980er Jahren an verschiedenen Universitäten der USA entwickelt worden. Sie können in eigene C++-Programme eingebunden werden. Die Barrett-Hand wird mit der libBhand kontrolliert, die eine Erweiterung der Programmier-Bibliothek des Herstellers ist. Der entscheidende Punkt bei der Steuerung der Hardware ist an dieser Stelle jedoch nicht die Benutzung dieser speziellen Bibliotheken. Der Server kann auch auf andere Hardwareabstraktions-Bibliotheken aufgesetzt werden, da die genutzten Funktionen sehr fundamentale Grundfunktionen sind. Es muss für Hand und Arm lediglich möglich sein, eine bestimmte Gelenkwinkel-Position einzugeben, die dann von der Hardware umgesetzt wird. Falls Kraftsensoren vorhanden sind, müssen nur deren Werte ausgelesen werden können. Es ist nicht nötig, dass die Bibliotheken die inverse Kinematik berechnen oder Geschwindigkeiten, Beschleunigungen oder Kräfte in Kartesische Koordinaten umwandelt. Alle benötigten Funktionen arbeiten auf dem Gelenkwinkelraum. Die komplette aufwendige Übertragung des Kartesichen Raumes in den Gelenkwinkelraum wird in die Clients verlagert. Trajektorien, also Pfade, an denen der Arm entlang geführt werden soll, werden durch schnell aufeinander folgende Zwischenziele erzeugt. Auch diese werden direkt von den Clients berechnet.

Die Auslagerung dieser berechnungsintensiven Aufgaben in die Clients hat mehrere

Vorteile. Erstens wird der Computer des Roboters und das Netzwerk entlastet. Zweitens arbeitet der Client sowieso schon mit vielen verschiedenen Koordinatensystemen, durch die Kinematik des Eingabegerätes und die Visualisierung am Bildschirm. Drittens kann es dem Clientbenutzer direkter und schneller mitgeteilt werden, wenn eine Berechnung zu lang dauert. Außerdem spiegelt sich in dieser Struktur die angestrebte Passivität des Servers wieder. Dieser soll lediglich die Befehle ausführen. Der Client soll den aktiven Teil übernehmen. Er generiert die Befehle, die vom Benutzer gefordert werden. Ändert der Benutzer plötzlich die Richtung, so kann der Client daraus direkt neue Befehle generieren und diese absenden, ohne dass diese Umentscheidung durch die Netzwerklatenz verzögert verarbeitet wird.

Seine Hauptaufgabe ist es dafür zu sorgen, dass die geforderte Zielposition der Manipulator-Gelenke schnell und durch eine flüssige Bewegung erreicht wird. Dabei darf die Position, die Geschwindigkeit und die Beschleunigung jedes einzelnen Gelenkes die gegebenen Maximalwerte nicht überschreiten. Die geforderten Zielstellungen verändern sich dabei innerhalb sehr kurzer intervalle von wenigen Millisekunden. Das bedeutet, dass in einer Programmschleife mit sehr hoher Priorität die Ausgabe-Trajektorie ständig angepasst werden muss. Diese Programmschleife ist in dem ControlLoop-Thread implementiert. Es handelt sich dabei um eine sogenannte Callback-Funktion, die aus der Hardware-Bibliothek heraus jedesmal aufgerufen wird, wenn an die Hardware neue Werte für die Gelenkwinkel ausgegeben werden müssen. Die Gelenkwinkel werden dann von den Servo-Reglern auf die Gelenkmotoren übertragen. Um eine flüssige, glatte Bewegung zu erzeugen, werden die nachfolgenden Formeln für jedes Gelenk einzeln benutzt. Sei $\hat{x}_t$ die aktuell geforderte Gelenkstellung, $x_t, t \in \mathbb{N}$ die diskrete Folge der ausgegebenen Gelenkstellungen und v_t die Folge der ausgegebenen Winkel-Geschwindigkeiten. Und seien $v_{\max}$ und $a_{\max}$ die Maximalwerte für die Winkel-Geschwindigkeit und die Winkel-Beschleunigung. Alle auszugebenen Winkel x_t werden iterativ mit

$$x_{t+1} = x_t + v_t \tag{5.1}$$

$$\hat{v}_t = a_{\max}(\hat{x}_t - x_t) \tag{5.2}$$

und

$$v_{t+1} = \begin{cases} \hat{v}_t & \text{für} \quad |\hat{v}_t| < |v_t| \\ v_t + a_{\max} & \text{für} \quad v_{\max} > \hat{v}_t > v_t > 0 \\ v_t - a_{\max} & \text{für} \quad -v_{\max} < \hat{v}_t < v_t < 0 \\ v_t & \text{sonst} \end{cases} \tag{5.3}$$

berechnet. Das kontinuierliche Äquivalent zu der diskreten Ausgabefunktion wird durch die Wahl sehr kurzer Zeitintervalle zwischen den Ausgabewerten approximiert. In Abbildung 5.5 ist ein Beispiel-Bewegungsverlauf eines Manipulator-Gelenkes grafisch dargestellt, wie er von dieser Kontrollschleife berechnet werden würde. Zur Vereinfachung bleibt der Zielpunkt hierbei wärend der kompletten Berechnung konstant. In der oberen Grafik ist der sprungstellenfreie Verlauf der Geschwindigkeit und damit der ersten Ableitung der Ausgabefunktion verdeutlicht. Dieser Verlauf ist in Teilintervallen linear. Aus diesen Eigenschaften folgt die stetige Differenzierbarkeit der Ausgabefunktion, welche hier als ein geeignetes Glattheitskriterium für eine flüssige Bewegung ausreichen soll. Im Fall einer dynamischen Zielposition, kann ebenso die Stetigkeit der Eingabefunktion angenommen werden, da nur reale Bewegungen wiedergespiegelt werden. Eine Verkettung zweier stetiger Funktionen ergibt wieder eine stetige Funktion. Darum ist die Ausgabe auch bei einer praktischen, dynamischen Anwendung flüssig. Die Formel (5.3) ist darüber hinaus genau so gewählt, dass die gegebenen Grenzwerte der Geschwindigkeit und der Beschleunigung eingehalten werden.

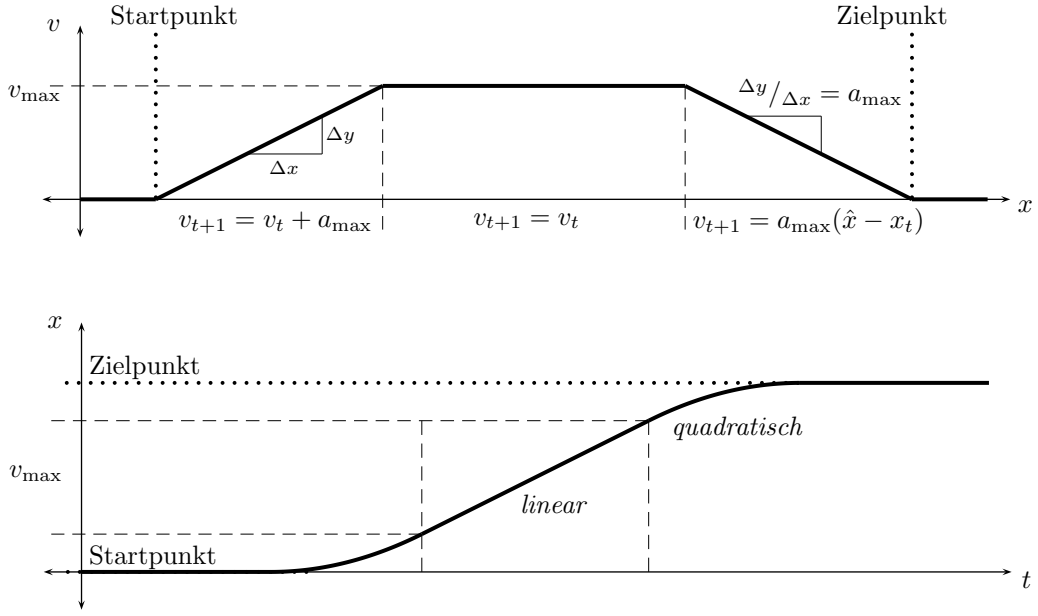


Abbildung 5.5: Darstellung der resultierenden Bewegung eines Gelenkes bei stationärer Zielstellung. Oben mit der Winkel-Geschwindigkeit v_t in Abhängigkeit von x_t und unten die Winkelstellung x_t über der Zeit t.

Bei der angestrebten Kopplung von haptischem Eingabegerät und dem Manipulator ist, wie schon für die andere Kommunikationsrichtung beschrieben, einer sehr zeitnahe Umsetzung der empfangenen Befehle wichtig. Die Client-Anwendung sendet

dem Telemanipulations-Server fortlaufend neue Gelenkwinkelwerte zu, die von der Regelschleife so schnell wie möglich auf den Manipulator übertragen werden müssen. Jedes neue Datenpaket beinhaltet den aktuellen Befehl für alle Gelenke. Ältere Pakete, die bis zu diesem Zeitpunkt noch nicht abgearbeitet worden sind verlieren ihre Gültigkeit. Aus diesem Grund speichert der entsprechende Netzwerk-Programm-Thread immer nur den jeweils aktuellen Befehl. Die Regelschleife benutzt ständig diesen Befehl als Zielposition für den vorgestellten Algorithmus. Dieser Befehl wird also in einer gemeinsam genutzten Variable gespeichert. Alle Gelenke werden dabei einzeln behandelt und benötigen jeweils nur einen einzelnen, einfachen Datentyp, wie eine Vier-Byte-Gleitkommazahl. Für dieses effiziente Verfahren wird also kein gegenseitiger Ausschluss-Mechanismus der konkurrierenden Prozesse benötigt.

5.3 Telemanipulations-Client

Der Telemanipulations-Client ist eine Anwendung, die auf einem beliebigen Computer im Netzwerk des Roboters ausgeführt werden kann. Dadurch soll möglichst vielen Benutzern Zugang zu dem Telemanipulations-Server ermöglicht werden. Nachfolgend werden die Aufgaben und die gewählte Software-Struktur des Client vorgestellt.

5.3.1 Aufgaben des Clients

Um eine geeignete Software-Struktur zu entwickeln, sollen die Aufgaben des Telemanipulations-Clients präzisiert werden. Es werden ähnliche Notationen wie in Abschnitt 5.2.1 benutzt, um mathematische Größen einfach darzustellen. Der Eingabewert $\mathbf{X}_t$ stellt die vom Benutzer gewünschte Zielposition und -Orientierung von Roboterarm und -Hand dar. Das virtuelle Robotermodell wird in der matrixförmigen Datenstruktur $\mathbf{V}_t$ gespeichert. Die Werte $\vec{s}_t$ und $\vec{c}_t$ werden, wie schon in 5.2.1, als aktueller Roboter-Hardware-Zustand und Steuerbefehl benutzt.

In Abbildung 5.6 sind die Aufgaben des Clients aufgeführt. Eine Software, die diese Aufgaben erfüllt, kann auf viel mehr Arten realisiert werden, als dieses beim Server der Fall ist. Nachfolgend werden die praktischen Umsetzungen der einzelnen Aufgaben als Teile der Software-Lösung beschrieben. Erst im Anschluss daran wird, wie schon für den Server, die Software-Struktur des kompletten Clients dargestellt.

Die Aufgaben des Clients sind:

- Benutzereingaben $\mathbf{X}_t$ gewinnen,
- $\mathbf{X}_t$ auf $\mathbf{V}_t$ übertragen (inverse Kinematik),
- Kraftausgabe an den Benutzer,
- 3D Darstellung des virtuellen Roboter-Modells $\mathbf{V}_t$ auf dem Bildschirm,
- bei Verbindung zum Server,
  - Verbindung überwachen,
  - $\vec{s_t}$ vom Server empfangen,
  - $\vec{s_t}$ auf $\mathbf{V}_t$ übertragen,
  - $\vec{c_t}$ generieren und senden,

Abbildung 5.6: Aufgaben des Clients

5.3.2 Virtuelles Modell

Das virtuelle Modell des Roboterarmes beschreibt dessen komplette kinematische Kette und soll darüber hinaus aktuelle Gelenkwinkel-Stellungen speichern. Da viele Berechnungen mit diesem Modell durchgeführt werden müssen, bietet es sich an, eine mathematische Beschreibung wie die Denavit-Hartenberg-Parameter zu benutzen, siehe Abschnitt 2.2.2. Die Verwendung der Denavit-Hartenberg-Transformationen ist ein weit verbreiteter Standard in der Industrierobotik. Mit ihnen können alle nötigen Koordinatentranformationen einer kinematischen Kette als einfache Matrixmultiplikationen beschrieben und berechnet werden. Darauf aufbauend wird für das Telemanipulations-System eine Datenstruktur benutzt, die zusätzlich zu den Denavit-Hartenberg-Parametern zu jedem kinematischen Glied eines Roboterarmes weitere nützliche Daten für den Client enthält. Diese Datenstruktur wird in dieser Arbeit als erweiterte Denavit-Hartenberg-Parameter bezeichnet. Zu jedem Gelenk werden der Gelenktyp und mögliche Minimal- und Maximalwerte gespeichert. Zusätzlich dazu wird ein Skalierungfaktor für die Gelenkwinkel angegeben, falls einzelne Winkel nicht im Bogenmaß interpretiert werden sollen, sondern zum Beispiel im Gradmaß oder im geodätischen Winkelmaß. Jedem Glied kann eine Datei zugewiesen werden, die die geometrische Form dieses Elementes enthält, um die dreidimensionale Darstellung des Modelles realistischer aussehen zu lassen. Für die Berechnungen mit dem virtuellen Robotermodell wird eine matrixförmige Datenstruktur $\mathbf{V}_t$ verwendet. Sie

enthält eine konstante, exakte Beschreibung der kinematischen Kette des Roboterarmes durch die hier beschriebenen erweiterten Denavit-Hartenberg-Parameter und die variablen Gelenkwinkelstellungen $\vec{j}_t$ des virtuellen Modells. Die Werte $\vec{j}_t$ sollen mit den Gelenkwinkelstellungen des realen Manipulators synchronisiert werden.

Eine Instanz dieser kompletten Datenstruktur wird in der Arbeit als virtuelles Modell bezeichnet. Sie korrespondiert direkt mit ihrer Bildschirmdarstellung in einer virtuellen Realität. In der Praxis werden zwei dieser virtuellen Modelle verwendet. Eines wird zur Verarbeitung der Benutzereingaben und deren Bildschirmdarstellung verwendet. Mit dem Zweiten werden die Werte des realen Manipulators verarbeitet.

5.3.3 Inverse Kinematik

Eine Hauptschwierigkeit des Telemanipulations-Clients ist das kontinuierliche Berechnen von Gelenkwinkeln aus der Eingabe des Benutzers. In den in Abbildung 5.6 genannten Aufgaben entspricht dies dem zweiten Punkt, $\mathbf{X}_t$ auf $\mathbf{V}_t$ zu übertragen. Dabei ist die Grundidee, dass der Benutzer fortlaufend Punkte im Raum angibt, zu denen sich die Hand des Roboters bewegen soll. Zusätzlich zu den Raumkoordinaten soll auch die gewünschte Orientierung eingegeben werden. Zusammengefasst wird diese Eingabe nachfolgend als Ziel bezeichnet und als $\mathbf{X}_t$ notiert. Dabei handelt es sich um eine 4×4-Matrix, die mit homogenen Koordinaten die Basis des Koordinatensystems des gewünschten Tool Center Points, relativ zur Koordinatenbasis des Manipulators, ausdrückt.

Das in Abschnitt 2.2.3 beschriebene iterative Gradientenabstiegsverfahren wird benutzt, um aus den eingehenden $\mathbf{X}_t$ fortlaufend passende Bewegungen der Gelenkwinkel zu generieren. Hierfür wird ein Algorithmus benutzt, der in der Lage ist, dies für beliebige virtuelle Manipulator-Modelle $\mathbf{V}_t$ zu berechnen. Es wird die Iterationsvorschrift aus Abschnitt 2.2.3

$$\vec{j}_{t+1} = \vec{j}_t + c\ \vec{\nabla} e(\vec{j}_t) + r_t \tag{2.8}$$

implementiert. Die Fehlerfunktion $e(\vec{j}_t) = dist\Big(\mathbf{DHT}(\vec{j}_t) - \mathbf{X}_t\Big)$ wird durch die Distanzfunktion

$$dist\,(\mathbf{A}) = \sqrt{\beta(\mathbf{B}_{1,1} + \mathbf{B}_{3,3}) + (1-\beta)\mathbf{B}_{4,4}} \tag{5.4}$$

$$\text{mit} \quad \mathbf{B} = \mathbf{A}^T\mathbf{A}, \quad \mathbf{A} \in \mathbb{R}^{4\times 4} \tag{5.5}$$

berechnet. Bei sehr kleiner Orientierungsabweichung ist der resultierende Wert hierbei ungefähr proportional zur euklidischen Norm der Positionsabweichung. Größere

Orientierungsabweichungen sorgen für einen größeren Wert der Distanzfunktion. Die Verwendung von nur zwei Rotationsabweichungswerten $\mathbf{B}_{1,1}$ und $\mathbf{B}_{3,3}$ reicht dafür aus. Der Faktor β gewichtet den Einfluss der Orientierungsabweichung. Diese Definition der Distanzfunktion zeigt mit einem kleinen Wert von $\beta \approx 0{,}1$ in der Praxis gute Ergebnisse. Für den diskreten Fall werden die einzelnen Einträge des Gradientenvektors $\vec{\nabla} e(\vec{j}_t)$ werden entsprechend einer angepassten Formulierung des Mittelwertsatzes approximiert. Demnach gilt, für alle in $(x - \mathrm{d}x, x + \mathrm{d}x)$ stetig differenzierbaren Funktionen f existiert mindestens ein $\xi \in \mathbb{R}$ mit $0 < \xi < \mathrm{d}x$, so dass gilt:

$$\frac{f(x - \mathrm{d}x) - f(x + \mathrm{d}x)}{2\mathrm{d}x} = f'(x \pm \xi) \tag{5.6}$$

Für $dx \in \mathbb{R}$ wird dabei ein sehr kleiner Wert $dx \ll \pi$ gewählt, um den Differenzenquotient für die Approximation von Ableitungen f' zu verwenden.

$$\vec{\nabla} e = \left(\frac{\partial e}{\partial x_1}, \frac{\partial e}{\partial x_2}, \frac{\partial e}{\partial x_3}, \ \ldots, \frac{\partial e}{\partial x_n} \right)^T \tag{5.7}$$

Wobei $n \in \mathbb{N}$ die Anzahl der Gelenke und wegen (5.6)

$$\frac{\partial e}{\partial x_i} = \frac{f(x_i - \mathrm{d}x) - f(x_i + \mathrm{d}x)}{2\mathrm{d}x} \tag{5.8}$$

mit $i \in \{1, 2, 3, \ \ldots, n\}$ ist.

Die hier vorgestellte Methode berechnet die inverse Kinematik fortlaufend. Sie ist in der Client-Anwendung implementiert und es werden nicht die serverseitigen Kinematikfunktionen genutzt. Hieraus entstehen die folgenden Vorteile:

- Die Planung von Telemanipulationen ist auch ohne Verbindung zum realen Roboter möglich.
- Das Verfahren liefert auch bei unerreichbaren Eingaben eine intuitiv sinnvoll erscheinende Lösung.
- Auch bei auftretenden Singularitäten wird eine sinvolle Lösung geboten.
- Der Benutzer kann die Lösung im laufenden Prozess anpassen.
- Auch Zwischenlösungen der iterativen Suche können von der Hardwaresteuerung des Manipulators bereits genutzt werden. Die resultierende Geschwindigkeit im Gelenkwinkelraum führt immer in die Zielrichtung.

Nachteilig ist hierbei das langsamere Konvergenzverhalten gegenüber anderen in Abbschnitt 2.2.3 angesprochenen Verfahren. Darüber hinaus ist die resultierende Trajektorie aller Zwischenergebnisse weder im kartesischen Raum noch im Gelenkwinkel-

raum linear. Daher wirkt sie bei der Suche nach weiter entfernten Zielen nicht immer zielgerichtet. Da die eingegebenen Ziele jedoch schnell aufeinander folgen, müssen in der Praxis nur sehr nahe liegende Ziele gesucht werden, bei denen dies nicht auffallen kann.

Die Tatsache, dass das Gradientenverfahren bei der schnellen Suche nach einer Lösung in lokale Minima geraten kann, in denen die Suche verharrt, stellt aus Benutzersicht einen Gewinn an Kontrolle und Tranzparenz dar. Bei der komplett manuellen Steuerung ist es angestrebt, das die Manipulatorkontrolle keine Befehle erzeugt, die der Benutzer nicht gewollt hat. Es wäre durch automatische Erkennung dieser Situationen und das parallele Ausführen mehrerer Suchen möglich, diese Minima automatisch zu verlassen. Dadurch würde immer das globale Minimum gefunden werden. In der praktischen Anwendung müssten dafür jedoch meistens große Schwenkbewegungen ausgeführt werden, die die Entfernung von Manipulator und Zielkoordinaten für kurze Zeit erhöhen. Die Art dieser Bewegung unterläge keinerlei Kontrolle durch den Benutzer. Sie wäre für unerfahrene Benutzer völlig unvorhersehbar und könnte in ungünstigen Fällen physikalische Beschädigungen herbeiführen. Bei der rein manuellen Steuerung ist es also von Vorteil, wenn dieses Problem nicht automatisch gelöst wird. Der Benutzer wird durch die visuelle und haptische Ausgabe in die Lage versetzt, lokale Minima zu erkennen. Durch den interaktiven Charakter der Anwendung, kann der Benutzer dann selbst kontrollieren, wie das Problem zu lösen ist. Er kann also durch das Ändern der Zielkoordinaten oder das geziehlte Ändern einzelner Gelenkwinkelstellungen eine neue, gefahrlose Trajektorie in Richtung zu dem eigentlich angestrebten Ziel erzeugen.

5.3.4 Kollisionsvermeidung

Die Vermeidung von Beschädigungen des Roboters und des Manipulators hat eine hohe Priorität. Die Ursachen von Beschädigungen können in zwei Kategorien unterteilt werden. Diese sind zum Einen Kollisionen mit Umgebungsobjekten, wie Tischen und Wänden, und zum Anderen die Kollision des Manipulators mit sich selbst oder mit dem Roboter. Nur durch ein detailgetreues, dynamisches Modell der kompletten Umgebung könnten Fälle der ersten Kategorie hinreichend vermieden werden. Dazu ist eine umfangreiche, kombinierte Nutzung vieler Sensordaten nötig, um dass Modell aktuell zu halten. Im Rahmen der vorliegenden Arbeit ist dieses nicht möglich. Fälle der zweiten Kategorie können jedoch behandelt werden, da die räumlichen Ausmaße und Bewegungen des Roboters bekannt sind. Darüber hinaus müssen der Roboter und der Manipulator auch für andere Aufgaben des Clients modelliert werden. Die zuvor beschriebene Suche nach der inversen Kinematik benutzt beispielsweise ein Modell

des Manipulators. Schon bei dieser Generierung der Trajektorien in der Client-Anwendung können diese Daten genutzt werden, um Beschädigungen zu vermeiden. Die Suche der inversen Kinematik soll also dahingehend geleitet werden, keine Ergebnisse zu liefern, die eine Selbstkollision zur Folge hätten. Für diese Kollisionvermeidung gibt es viele Implementationsmöglichkeiten.

Es wäre möglich die in Java 3D implementierte Kollisionsdetektion einzubeziehen, um die Wahrscheinlichkeit einer Selbstkollision exakt zu bestimmen. Hierfür müssten exakte geometrische Beschreibungsdaten des Roboters benutzt werden. Die Wartung und Aktualisierung dieser Daten würde jedoch einen großen manuellen Aufwand erzeugen, wenn Teile des Roboters verändert werden. Eine effizientere Berechnung über stark abstrahierte Eigenschaften der Manipulator-Geometrie ist sinnvoller. Der Rechenaufwand der einzelnen Iterationsschritte der inversen Kinematiksuche sollte gering gehalten werden. Es sollen also möglichst einfache Selbstkollisions-Bedingungen aufgestellt werden, die in jedem Iterationsschritt überprüft werden können. Bei einer erkannten Selbstkollision wird das Ergebnis dieses Iterationsschrittes verworfen und die Suche nach besseren Gelenkstellungen wird fortgesetzt.

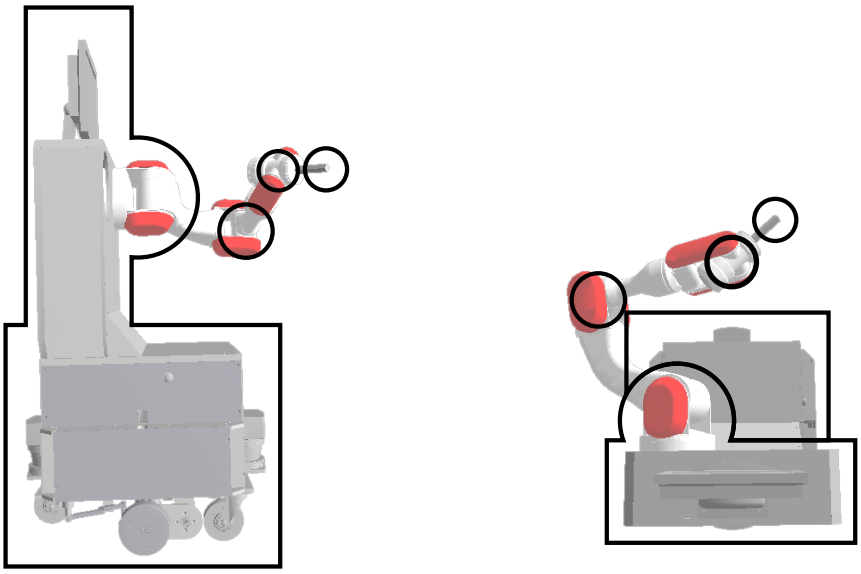


Abbildung 5.7: Vereinfachtes Roboter-Modell aus Quadern und Kugeln zur Erkennung von Selbstkollisionen.

Als erste Bedingung wird hierfür die minimale und maximale Auslenkung einzelner Gelenke beschränkt. Besonders im Bereich des Handgelenkes können dadurch Gelenkstellungen vermieden werden, in denen einzelne Teile der Hand mit dem Manipulator in Berührung kommen könnten. Für die weiteren Bedingungen werden mit Hilfe der

bekannten Kinematik die Raumpositionen einzelner Gelenke und des Tool Center Points berechnet. Für alle diese Koordinaten wird überprüft, ob sie mit einem stark vereinfachten Modell des Roboters in Berührung kommen. Dieses Modell besteht aus einfachen geometrischen Formen, wie Quadern, Ebenen und Kugeln. Diese sind so gewählt, dass alle Kollisionen mit dem Roboter, dem Manipulator oder dem Fußboden erkannt werden können. Ein zusätzlicher Sicherheitsradius wird dabei berücksichtigt. Der Arbeitsbereich des Manipulators sollte jedoch auch nicht zu sehr eingeschränkt werden. Abbildung 5.7 zeigt einen möglichen Aufbau für dieses Kollisionsmodell.

5.3.5 Netzwerk-Verbindung

Gemäß dem Übertragungsprotokoll aus Abschnitt 5.2.2 stellt der Client eine TCP/IP-Verbindung mit dem Telemanipulations-Server her. Der überwiegende Teil der Kommunikation erfolgt durch UDP/IP-Pakete. Der genaue Aufbau des Protokolls soll in diesem Abschnitt nicht erneut beschrieben werden, siehe dazu Abschnitt 5.2.2. Hier wird nur auf die Aspekte eingegangen, die für die Struktur des Clients relevant sind.

Besteht die TCP/IP-Verbindung mit dem Server, empfängt der Client UDP/IP-Datenpakete vom Server, die die aktuelle Position des Roboterarmes $\vec{s}_t$ enthalten. Diese müssen verarbeitet und zur Anpassung des virtuellen Modells benutzt werden. Die fortlaufenden Ergebnisse der inversen Kinematiksuche werden als Winkelbefehle $\vec{c}_t$ an den Server gesendet. Dieser übernimmt dann die entsprechende Steuerung des realen Manipulators.

5.3.6 Eingaben und Kraftausgaben

Das Telemanipulations-System soll eine Eingabemethode bereitstellen, die dem Benutzer ein intuitives Arbeiten erlaubt. Dafür muss ein Zielpunkt mit Orientierung $\mathbf{X}_t$ im dreidimensionalen Raum bewegt werden können. Es eignen sich also vorzugsweise Eingabemethoden mit sechs Freiheitsgraden. Zur Zeit sind die folgenden Techniken denkbar, die sich in zwei Hauptkategorien einordnen lassen.

Die erste Kategorie enthält alle Eingabegeräte, bei denen der Benutzer in der realen Welt ein Objekt bewegt, dessen Position und Orientierung automatisch bestimmt werden. Darunter fallen punktuelle haptische Geräte, eingabetaugliche Roboterarme und berührungslose Techniken, wie Multikamerasysteme oder gyroskopische Systeme. Die ersten beiden Techniken ermöglichen, zusätzlich zur Eingabe, die Ausgabe von Kräften, welches mit berührungslosen Techniken nicht möglich ist. All diese Techniken sind jedoch zur Zeit noch verhältnismäßig teuer und stehen daher nicht immer

zur Verfügung. Darum scheint es sinnvoll zu sein, zusätzlich zu dem geplanten haptischen Gerät, eine weitere Eingabemethode bereitzustellen, deren Vorteil einfach in der besseren Verfügbarkeit liegt.

Die zweite Kategorie enthält solche Eingabegeräte. Darunter fallen zum Beispiel die Computer-Tastatur, Computer-Maus oder ein Joystick. Eine derartige Eingabemethode kann nicht so intuitiv bedient werden. Die Bedienung zu beherrschen erfordert vermutlich eine längere Lernphase, da der Benutzer selbst lernen muss, in welcher Weise seine zweidimensionale Eingabe auf die dreidimensionale Ausgabe übertragen wird. Dieses ist überhaupt nur in Verbindung mit einer geeigneten visuellen Darstellung denkbar. Dadurch kann der Benutzer in seiner Lernphase überprüfen, welche Eingabe welche Auswirkung hat. Weitverbreitete Mäuse und Joysticks bieten überwiegend die Möglichkeit einer kontinuierlichen zweidimensionalen Eingabe. Eine dreidimensionale Eingabe ist mit ihnen jedoch schwierig zu realisieren. Die Einstellung der Orientierung macht es dann noch schwerer. Es gibt Joysticks oder die so genannte SpaceMouse, die über die dafür nötige Anzahl an Freiheitsgraden verfügen. Diese sind jedoch auf Grund ihres höheren Preises wiederrum nicht verbreitet genug. Bei der Benutzung der Tastatur können keine kontinuierlichen Bewegungsabläufe produziert werden. Jeder Tastendruck bewegt das Ziel schrittweise. Größere Schrittweiten ermöglichen ein schnelleres Arbeiten, jedoch sind kleinere Schrittweiten für genaues Arbeiten besser. Der Vorteil der Tastatur gegenüber Maus und Joystick ist jedoch, der große Umfang an benutzbaren Tasten. So kann das Ziel mit den Pfeiltasten und den Bildhoch- und Bildruntertasten dreidimensional verschoben werden. Durch das einfache zusätzliche pressen der Shift-Taste kann mit den selben Tasten die Orientierung rotiert werden. Diese Eingabemöglichkeit ist also zusätzlich implementiert.

Das in Abschnitt 4.3 beschriebene haptische Gerät wird für dieses System verwendet, da es die Anforderungen sehr gut erfüllt. Mit sechs Freiheitsgraden kann die Ziel-Position und -Orientierung in einer kontinuierlichen Bewegung eingegeben werden. Eine Einschränkung bietet es jedoch bei der Kraftausgabe, welche nur mit drei Freiheitsgraden ausgestattet ist. Das bedeutet, Kräfte können nur auf die Raumposition ausgegeben werden und nicht in Form von Drehmomenten auf die Orientierung. Größe und Form des Arbeitsbereiches unterscheiden sich zwischen dem haptischen Gerät und dem Manipulator. Die Koordinaten der Eingabe werden daher skaliert und verschoben, um den Hauptarbeitsbereich des haptischen Gerätes mit dem Hauptarbeitsbereich des Manipulators in Deckung zu bringen. Wie in Abbildung 5.8 veranschaulicht kann der Hauptarbeitsbereich als eine Kugel dargestellt werden, die den räumlichen Bereich enthält, in dem vermutlich die meisten Bewegungen ausgeführt werden sollen. Wie ebenfalls in der Abbildung zu erkennen ist, kann dieser Bereich sowohl von dem Eingabegerät als auch von dem Manipulator seitlich verlassen wer-

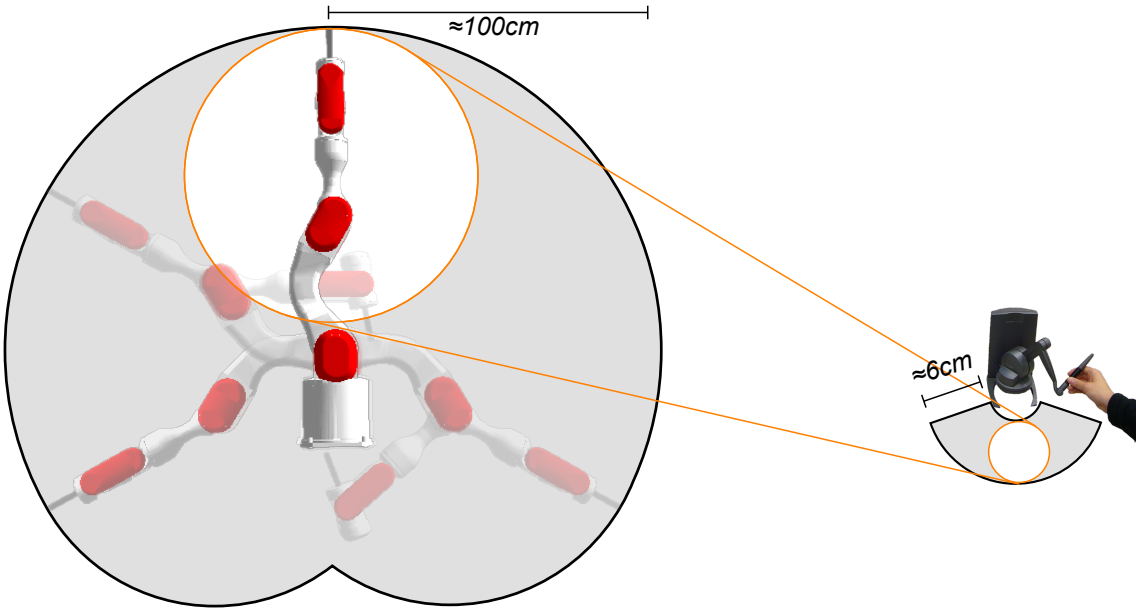


Abbildung 5.8: Draufsichten veranschaulichen die Größe der Arbeitsbereiche von Manipulator und haptischem Gerät (nicht Maßstabsgetreu).

den. Er stellt also keine Beschränkung des Arbeitsbereiches dar, sondern dient nur der Veranschaulichung der Skalierung der Eingabe. Die hohe räumliche Auflösung des verwendeten haptischen Gerätes von etwa 0,023 Millimertern erlaubt dabei auch die Steuerung von kleinen Bewegungen des Manipulators. Bei einem Skalierungsfaktor von 17 ist damit eine räumliche Auflösung der Manipulatorsteuerung von ungefähr 0,4 Millimetern möglich. Der Benutzer muss das Gerät jedoch mit sehr ruhigen und präzisen Handbewegungen steuern, denn auch alle Positionsabweichungen werden verstärkt ausgegeben. Aus diesem Grund kann der Skalierungsfaktor verändert werden, um die Handhabung bei rauschsensiblen Aufgaben zu erleichtern.

Die punktuell wirkende Kraft, die mit dem haptischen Gerät ausgegeben werden kann, sollte sinnvoll unterstützend genutzt werden. Unterstützend zur Eingabe der Position erscheint es sinnvoll, eine Kraft in die Richtung der aktuellen Position des Tool Center Pointes auszugeben. Der Benutzer spürt dadurch, an welcher Position sich der Endeffektor zur Zeit der Eingabe befindet. Um von diesem Punkt aus zu einem anderen zu gelangen, muss er eine Gegenkraft aufwenden. Durch dieses Prinzip ist das Telemanipulations-System bilateral. Der Benutzer erfährt eine Verbindung zwischen der kinästhetischen Wahrnehmung seines eigenen Armes und den kinästhetischen Informationen des Manipulators. Die zu erzeugende Kraft wird proportional

zur Positionsabweichung mit

$$\vec{F} = k \begin{pmatrix} \mathbf{TCP}_{1,4} - \mathbf{X}_{1,4} \\ \mathbf{TCP}_{2,4} - \mathbf{X}_{2,4} \\ \mathbf{TCP}_{3,4} - \mathbf{X}_{3,4} \end{pmatrix} \tag{5.9}$$

berechnet. **TCP** und **X** sind homogene Matrizen, deren Indizes ihre einzelnen Elemente auswählen. Mit der Federkonstanten k kann die Stärke der Kraft eingestellt werden. Kleinere Werte für k sorgen für eine ungenauere kinästhetische Ausgabe. Je größer der Wert gewählt ist, desto größer ist die Kraft, die der Benutzer aufbringen muss.

Zusätzlich könnten bestimmte Kraftsensorwerte des Manipulators zu dieser Kraftausgabe addiert werden. So könnte der Benutzer mit Hilfe von Drehmomentsensoren an den einzelnen Manipulatorgelenken die Kraft spüren, die bei Kontakt mit Umgebungsobjekten auftritt. Der verwendete Manipulator besitzt derartige Sensoren jedoch nicht. Durch Messung des Positionsfehlers zwischen der angesteuerten und der tatsächlich gemessenen Manipulatorposition kann theoretisch eine Schätzung der auftretenden Kontaktkräfte berechnet werden. Dieses ist bereits von [Ni und Wang 2002, Hashtrudi-Zaad und Salcudean 1996] angewendet worden. Der dafür nötige harte Umgebungskontakt ist bei der vorliegenden Arbeit jedoch zu jeder Zeit zu vermeiden gewesen, da die ungepolsterte Umgebung des verwendeten Roboters diesen beschädigen würde. Kraftsensoren besitzt der verwendete Roboter nur in den Fingern des Greifers am Ende des Manipulators. Die Einbindung der einzelnen Finger, durch Kinematik-Modellierung und Einarbeitung in neue Programmier-Schnittstellen, übersteigt jedoch den Rahmen dieser Arbeit und wird daher nicht realisiert.

5.3.7 Hardware-Steuerung

Die Hardware des Clients umfasst bei dessen Konzipierung haptische Eingabegeräte. Durch das Einbinden der Programm-Bibliothek OpenHaptics der Firma SensAble Technologies in das Programm, ist es möglich eine breite Produktpalette an haptischen Geräten für das Telemanipulations-System nutzbar zu machen [OpenHaptics 2004]. Es handelt sich jedoch um eine Bibliothek für die Programmierung mit C++.

Es gibt eine Quellcode offene Haptik-API für Java durch das JTouchToolkit von Ian John Archer, Software Engineer an der Birmingham City Universität [Archer 2007]. Dabei handelt es sich um die komplette Einbindung von OpenHaptics in Java. Es werden Java-Objekte bereitgestellt, die alle Funktionen und Datenstrukturen von OpenHaptics zur Verfügung stellen. Diese werden direkt über das Java Native Interface (JNI) mit den originalen OpenHaptics-Gegenstücken verbunden. Experimente

mit der Version 1.0 dieser Bibliothek haben jedoch teilweise Effizienzprobleme gezeigt. Das Hauptproblem liegt dabei in der Servoschleifen-Funktion, die bei der Haptikprogrammierung zu implementieren ist. Diese C++-Funktion wird durch OpenHaptics 1000 Mal pro Sekunde aufgerufen und muss jedes Mal die neu auszugebenen Kraftwerte berechnen. Bei diesem kurzen Zeitintervall muss die Funktion sehr effizient programmiert sein. Bei der Verwendung des JTouchToolkit ruft diese Funktion eine Java-Funktion auf, die dann die Kraftwerte berechnen muss. Bei dem Übergang zwischen C++-Programmteil und Java-Programmteil müssen viele benötigte Datenstrukturen umgewandelt werden. Diese Umwandlungen in beide Richtungen und die Berechnungen in der Java-Funktion kosten teilweise zu viel Rechenzeit. Stehen die auszugebenen Kraftwerte jedoch nicht rechtzeitig zur Verfügung, so erzeugt dieses ein Stottern des haptischen Gerätes.

Um dieses Problem zu Lösen wird eine neu programmierte JNI-Einbindung von OpenHaptics in die Java-Anwendung benutzt. Zur Steigerung der Effizienz werden dabei nicht alle, sondern nur die benötigten Funktionen von OpenHaptics bereitgestellt. Zusätzlich werden so wenig wie möglich Daten zwischen dem Java- und dem C++-Teil ausgetauscht. Die Servo-Programmschleife wird direkt in C++ in einem eigenen Programmthread mit hoher Priorität implementiert. Diese kann dauerhaft das haptische Rendering durchführen, selbst wenn der Java-Teil der Anwendung die benötigten Daten nicht schnell genug verarbeiten kann. Ein weiterer Vorteil dieser eigenen OpenHaptics-Einbindung ist die Erweiterbarkeit auf haptische Geräte anderer Firmen, durch das Einbinden anderer Haptik-Bibliotheken auf der C++-Ebene. Die Schnittstelle zur Client-Anwendung bleibt dabei unverändert.

5.3.8 Dreidimensionale Darstellung

Auf dem Bildschirm soll ein Modell des Roboterarmes dargestellt werden, dass dessen aktuellen Bewegungszustand widerspiegelt. Bewegungsabläufe sollen damit auch geplant werden können, bevor sie von der Hardware ausgeführt werden. Dadurch soll der Benutzer in der Lage sein, zu überprüfen, was der Roboterarm tatsächlich aus den Benutzereingaben macht. Dieser Teil des Clients erfüllt den Zweck eines Predictive-Displays (siehe Abschnitt 3.3). Eine Anzeige von Kamerabildern ist im Rahmen dieser Arbeit nicht vorgesehen. Zu diesem Zweck können parallel andere Programme genutzt werden, die für diese Aufgabe spezialisiert sind.

Ein wichtiger Entscheidungsfaktor für die Programmierung der Client-Anwendung in Java ist die geplante Verwendung der Java 3D API [JavaSoft 2002]. Die Java 3D API ist genau wie Java von Sun Microsystems entwickelt worden. Java 3D bietet einen großen Funktionsumfang zur Programmierung von dreidimensionaler Darstel-

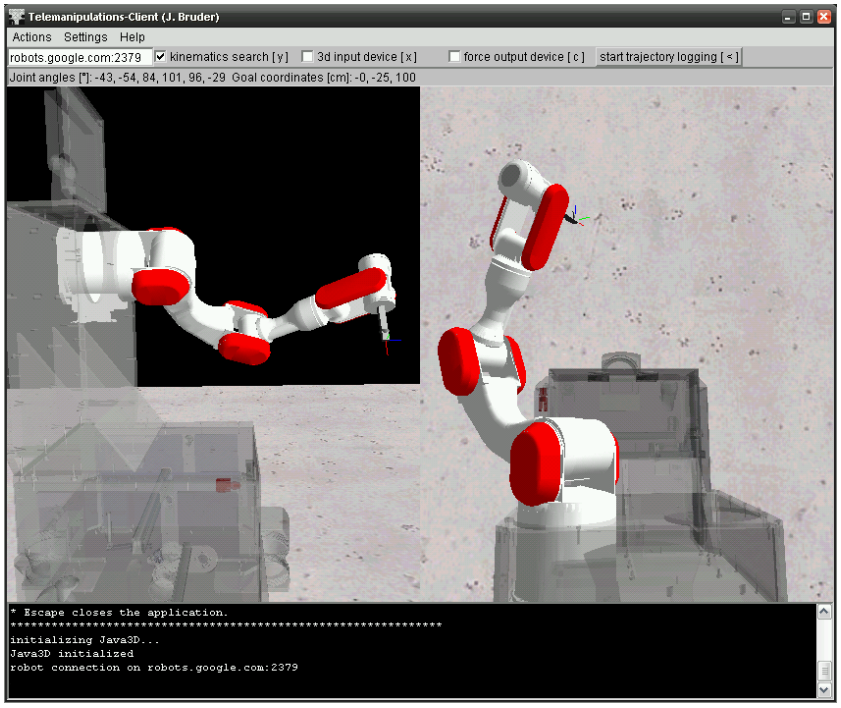


Abbildung 5.9: Client-Anwendung mit 3D-Darstellung des Roboter-Modelles.

lung auf zweidimensionalen Bildschirmen. Für die Darstellung werden moderne 3D-Programmier-Schnittstellen, wie OpenGL und DirectX, verwendet, die eine effiziente Hardwareunterstützung nutzen. Auf einer höheren Abstractions-Ebene, als bei diesen Schnittstellen, können in Java 3D virtuelle Welten erzeugt werden, ohne dass der Programmierer sich um die einzelnen Linien und Dreiecke der einzelnen Objekte kümmern muss. Es wird eine dynamische Datenstruktur verwendet, in der Szenen-Objekte, Oberflächenbeschaffenheiten, Bewegungsabläufe und Benutzerinteraktionen beschrieben werden können. Eine genauere Beschreibung von Java 3D ist nicht Thema dieser Arbeit, darum sei hier auf die zahlreichen ausführlichen Veröffentlichungen wie [JavaSoft 2002, Schnell und Strasser 2008] verwiesen.

Java 3D nutzt eine baumförmige Datenstruktur zur Verarbeitung der zu modellierenden virtuellen Welt. Diese Baumstruktur besteht, der objektorientierten Programmierung entsprechend, aus unterschiedlichen speziellen Java 3D-Objekten, die durch Referenzen miteinander verknüpft sind. Diese können Objekte, unter anderem auch Gruppen von Koordinaten-Transformationen darstellen, die auf Gruppen von dreidimensionalen Körpern angewendet werden. Mit diesem Konzept kann die kinematische Kette, die in dem virtuellen Modell der Client-Anwednung gespeichert ist (siehe 5.3.2), einfach auf das Java 3D-Modell übertragen werden. Die geometrischen Formen der einzelnen Kettenelemente werden aus externen Dateien geladen. Hierfür wird ein

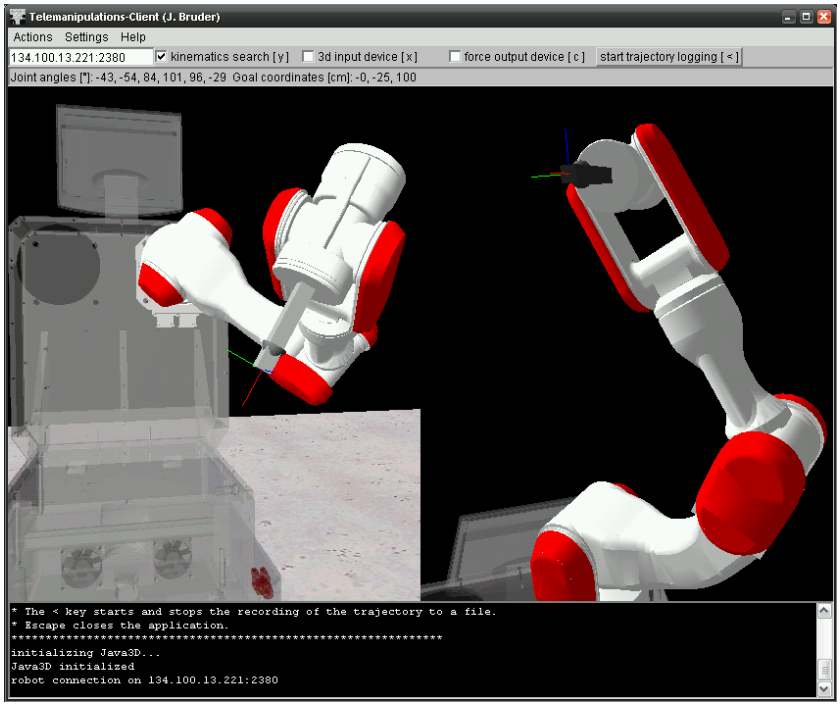


Abbildung 5.10: Client-Anwendung mit 3D-Darstellung des Roboter-Modelles aus anderer Perspektive.

Dateiformat der Grafiksoftware 3D Studio Max benutzt, da dieses auch von vielen anderen Programmen verarbeitet werden kann und somit eine hohe Kompatibilität bietet. Auch VRML-Dateien sind ein weit verbreiteter Standard, der hierfür ebenfalls benutzt werden kann.

Durch die realistisch wirkende, perspektivische Darstellung des dreidimensionalen Robotermodells kann der Benutzer die Haltung des Manipulators meistens sehr gut erkennen. Der Blickwinkel kann durch Mausbewegungen den Bedürfnissen des Benutzers angepasst werden. Die Tiefeninformation geht bei einem zweidimensionalen Bildschirm jedoch verloren. Die perspektivische Darstellung ermöglicht es dem Benutzer zwar, große Entfernungsunterschiede zu rekonstruieren, da alle dargestellten Körper feste, bekannte Ausmaße besitzen. Doch für filigrane Telemanipulations-Aufgaben reicht dieses nicht aus. Es wird daher eine zweite Darstellung aus einem andern Blickwinkel angezeigt. Die Blickwinkel beider Ansichten können so eingestellt werden, dass alle relevanten Informationen erkennbar sind. Die Abbildungen 5.9 und 5.10 zeigen das Fenster der Client-Anwendung mit verschiedenen Blickwinkeln auf das selbe Motiv. Erkennbar ist hier auch, dass nicht nur der Manipulator sondern auch der Roboter und der Fußboden modelliert sind.

5.3.9 Grafische Bedienung des Clients

Den größten Teil der Fläche der grafischen Bedienoberfläche, die nachfolgend als GUI bezeichnet wird, nimmt also die Darstellung des Roboters ein. Die weiteren Elemente der GUI folgen Standardkonzepten von aktuellen grafisch orientierten Computer-Betriebssystemen. Sie sind mit dem Abstract Windowing Toolkit (AWT) von Java implementiert, da dieses effizient alle benötigten Elemente durch das Betriebssystem erzeugt. Die Anwendung verwendet ein Programm-Fenster, dessen Größe und Position auf dem Bildschirm verändert werden kann. Am oberen Rand des Fensters ist ein Auswahlmenü für die Befehlseingabe vorhanden, über das auch Programmeinstellungen vorgenommen werden können. Darunter befindet sich ein Feld zur Eingabe der IP-Adresse und des UDP-Ports. Hierbei wird bewusst das Erscheinungsbild von häufig benutzten Internet-Browser-Anwendungen nachgeahmt, um eine intuitive Bedienung zu unterstützen. Zusätzlich gibt es noch einige Statusanzeigen und eine konsolenartige Textausgabe, die für den Benutzer wertvolle Programmereignisse auflistet.

Die wichtigsten Funktionen können zusätzlich auch durch das Drücken einzelner Tasten auf der Computer-Tastatur aufgerufen werden. Die Computer-Maus mit den GUI-Elementen zu benutzen, um diese Funktionen aufzurufen nimmt meistens mehr Zeit in Anspruch.

5.3.10 Architektur des Clients

Für den Telemanipulations-Client kann nun eine Software-Struktur erstellt werden, die die Aufgaben des Clients bewältigt und die Überlegungen der vorangegangenen Abschnitte umsetzt. Abbildung 5.11 zeigt die entwickelte Struktur in einer vereinfachten Form. Wie schon für den Server sind wieder die einzelnen Komponenten und ihre Beziehungen, sowie die Anbindung an Programm-Bibliotheken und das Netzwerk verbildlicht.

Das Programm beginnt von dem zentralen Client-Thread aus, in dem die Inizialisierungen der benötigten Datenstrukturen und der grafischen Bedienoberfläche (GUI) durchgeführt werden. Zu diesen Datenstrukturen gehört auch das virtuelle Robotermodell. Anschließend erzeugt dieser Thread den neuen Connection-Thread und kümmert sich selbst nur noch um die Aufgaben der GUI. Der Connection-Thread versucht automatisch eine TCP/IP-Verbindung zum Telemanipulations-Server aufzubauen. Ist eine solche Verbindung hergestellt, erzeugt er einen neuen Thread, der die UDP/IP Pakete von dem Server empfängt. In diesen Paketen sind die jeweils aktuellen Werte der Roboter-Hardware $\vec{s}_t$ codiert. Die Werte werden dann direkt auf das virtuelle Modell übertragen. Der Connection-Thread verwaltet parallel dazu die

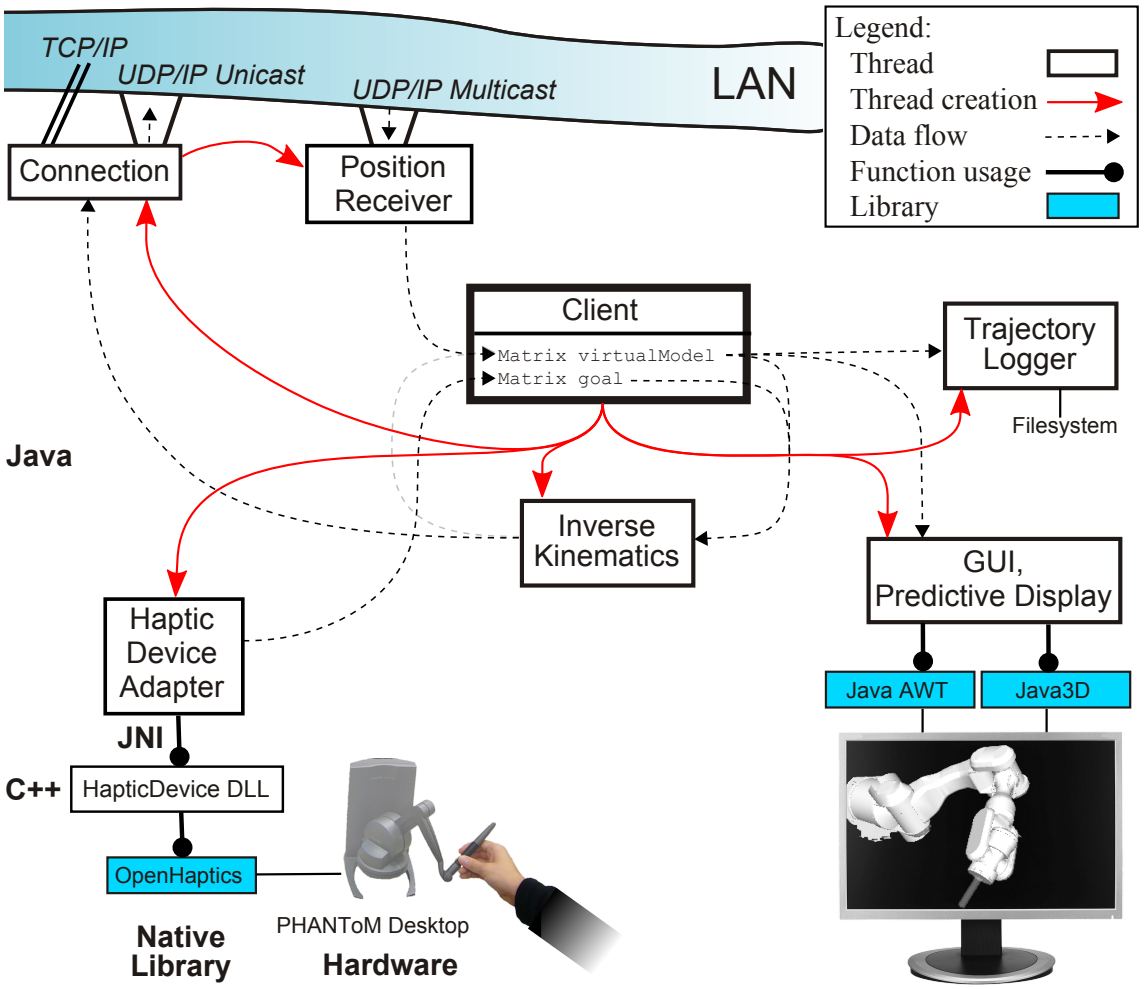


Abbildung 5.11: Vereinfachstes Schema des Telemanipulations-Clients.

TCP/IP-Verbindung und setzt das Senden von Steuerbefehlen über UDP/IP-Pakete um.

Die GUI bietet nun dem Benutzer die Möglichkeit einzeln weitere Komponenten zu aktivieren oder wieder zu deaktivieren. All diese optionalen Komponenten sind als einzelne Programm-Threads realisiert, die parallel zu dem bestehenden System ausgeführt werden können. Zu ihnen gehören die Suche der inversen Kinematik mit der Kollisionskontrolle, die Anbindung eines haptischen Gerätes und der Trajektorien-Aufzeichner, mit dem Armbewegungen gespeichert werden können.

5.4 Zusammenfassung

Das in diesem Kapitel vorgestellte verteilte Software-System bedient sich verschiedener Programmiersprachen und Programmier-Bibliotheken, wie Multi-RCCL, OpenHaptics und Java3D. Es realisiert eine flexibel einsetzbare Umsetzung des geplanten Telemanipulations-Systems. Einzelne wichtige Komponenten des Systems, wie die Implementation der permanent berechneten inversen Kinematik, ein neues Telemanipulations-Protokoll über UDP/IP, die Manipulator-Kontrolle und die Kollisionsvermeidung, sind beschrieben worden. Die Benutzung eines virtuellen Robotermodells für eine realistische Darstellung ist beschrieben und darüber hinaus ihr Wert für Simulationszwecke aufgezeigt worden.

In Abbildung 5.12 sind nur die Systemkomponenten verbildlicht, die für die Verarbeitung der Steuerungsdaten relevant sind. Dieses dient der Verdeutlichung des Datenflusses innerhalb des gesamten Systems. Dabei sind zusätzlich die Umwandlungen zwischen kartesischen Koordinaten und Gelenkwinkelkoordinaten hervorgehoben. Die Vektoren $\vec{s}$ und $\vec{s}$ stellen Gelenkwinkel dar und die Matrizen **S** und **X** enthalten kartesische Koordinaten mit Orientierung. Diese Variablen sind in einer speziellen Reihenfolge miteinander gekoppelt und erzeugen dadurch einen Datenfluss. Der Benutzer erzeugt die Eingabe **X**, aus dem dann die inverse Kinematik $\vec{j}$ berechnet wird. Diese steuert dann über das Netzwerk den Manipulator, der wiederum über das Netzwerk mit $\vec{j}$ und **S** die Kraftausgabe an den Benutzer beeinflusst. An dieser Stelle entsteht die Rückkopplung des Systems, da die Kraftausgabe des haptischen Gerätes wiederum die Eingabe **X** des Benutzers beeinflusst. Besteht keine Verbindung zum Server, so kann der Telemanipulations-Client dennoch in einem Simulationsmodus benutzt werden, um Telemanipulationen zu planen und zu üben. Zu diesem Zweck kann die haptische Rückkopplung direkt die Steuerdaten der inversen Kinematik verwenden.

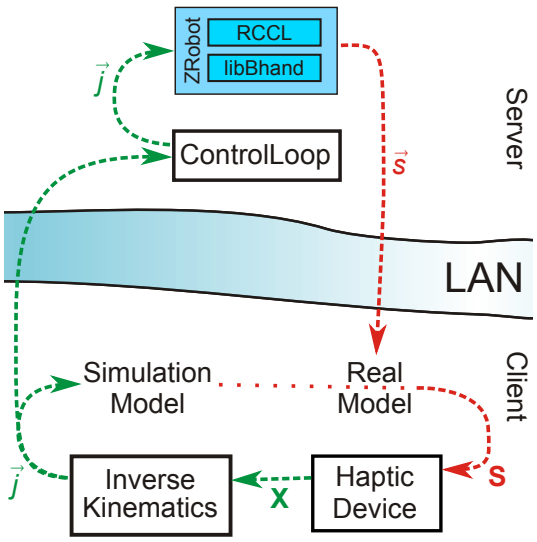


Abbildung 5.12: Datenfluss-Schema des Gesamtsystems.

In Abbildung 5.13 ist abschließend das Zusammenwirken aller Systemkomponenten und der Datenfluss in dem entwickelten Telemanipulations-System verdeutlicht.

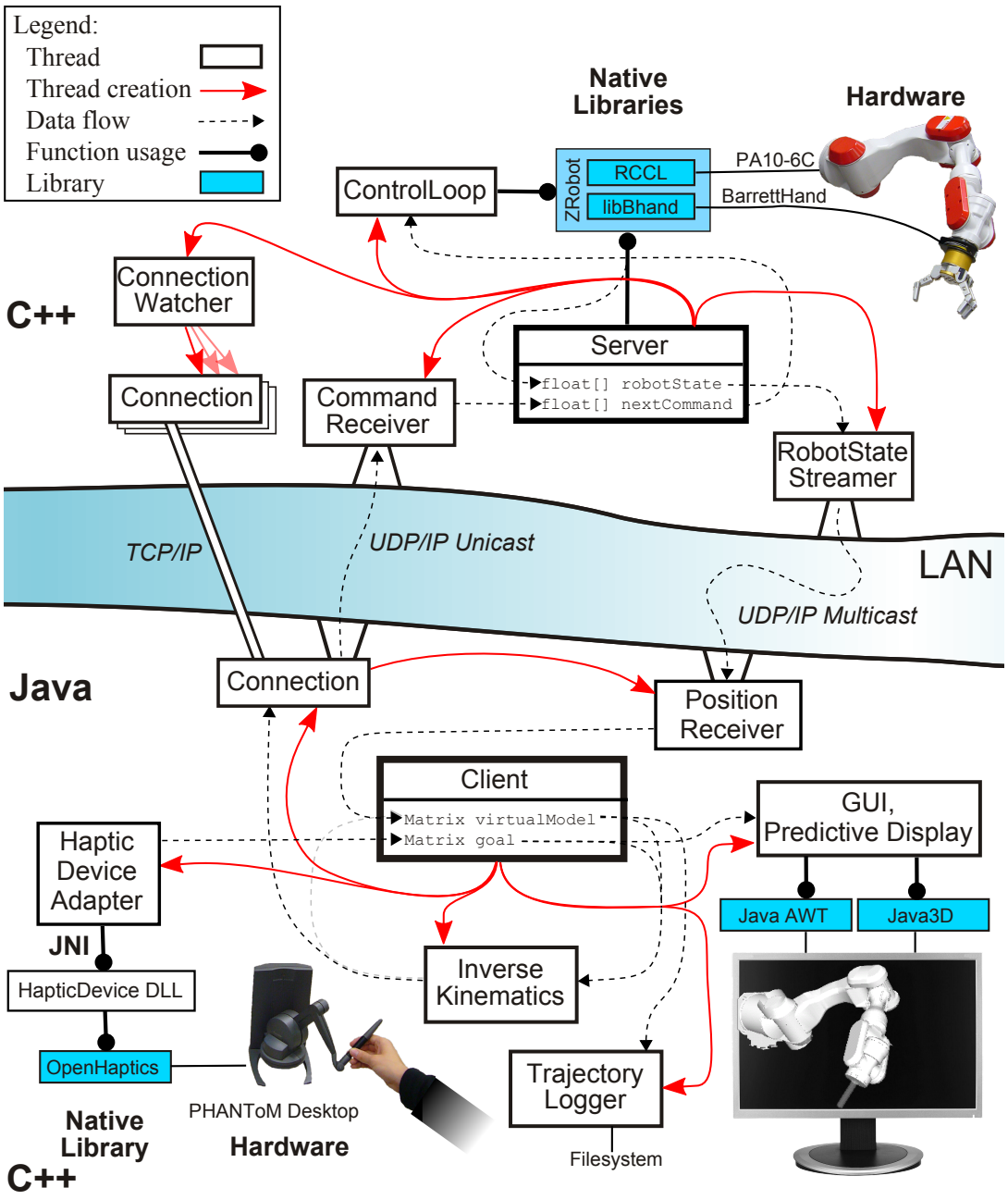


Abbildung 5.13: Schema des verteilten Telemanipulations-Systems mit der Server-Anwendung im oberen und der Client-Anwendung im unteren Bereich.

6 Ergebnisse

Das in dieser Arbeit entwickelte Telemanipulations-System wird in diesem Kapitel auf einige seiner wichtigen Eigenschaften überprüft. Bisher sind die Eigenschaften des Systems nur theoretisch behandelt worden. In diesem Abschnitt werden dazu Ergebnisse praktischer Experimente genutzt. Das Telemanipulations-System wird dabei evaluiert und auf die Qualität der Implementation hin untersucht.

6.1 Geschwindigkeit der inversen Kinematik

Das in den Abschnitten 5.3.3 und 2.2.3 präsentierte Verfahren zur permanenten Lösung der inversen Kinematik findet durch die beschriebene Umsetzung immer Lösungen. Die Echtzeitfähigkeit des gesamten Systems hängt jedoch entscheidend von der Geschwindigkeit ab, mit der die Lösungen gefunden werden. Die Parameter Geschwindigkeitsfaktor c, Orientierungsanteil β, Rauschfaktor ϵ und Differenzenquotient-Abstand dx aus der Gleichung (2.8) beeinflussen allgemein das Verhalten des Verfahrens und c und dx beeinflussen dabei speziell auch dessen Geschwindigkeit. Die Dauer der einzelnen Iterations-Schritte des Verfahrens ist von der Effizienz der Implementation und der Geschwindigkeit und Auslastung des verwendeten Computers abhängig. Die Abbildungen 6.1 und 6.2 zeigt, dass bei der Wahl geeigneter Parameter die Anzahl der nötigen Iterationsschritte stark verringert werden kann. Das iterative Verfahren verkleinert den Abstand zum Ziel fortlaufend bis zu dem Wert Null. Lediglich die Auswertung von Experimenten erfordert die folgende Definition der Genauigkeit von gefundenen Zwischenlösungen. Das Ereignis des Erreichens eines Zieles entspricht dabei dem Ereignis einer gefundenen Lösung. Es ist hierbei sinnvoll die Bedingung dieses Ereignisses der gewünschten räumlichen Auflösung der Manipulatorbewegung anzupassen. Zum Beispiel sind bei einfachen, alltäglichen Manipulationen, wie dem Greifen eines Bechers, Positionsabweichungen von einem Millimeter akzeptabel. Daher kann für die Untersuchung ein erreichter Abstand der kleiner als ein Millimeter ist als Lösung gewertet werden. Bei sicherheitskritischeren Anwendungen, wie zum Beispiel in der Medizin, müsste dieser Wert von einem Millimeter jedoch entsprechend verringert werden.

Bei den im Folgenden beschriebenen Experimenten wird der Tool Center Point des virtuellen Modelles jeweils zu der Ausgangsposition $(-30\text{cm}, -5\text{cm}, 110\text{cm})^T$ gebracht. Als Ziel wird der 30 Zentimeter entfernte Punkt $(-30\text{cm}, -35\text{cm}, 110\text{cm})^T$ eingegeben. Die Orientierung der beiden Punkte zeigt nach vorne vom Roboter weg, entlang der z-Achse. Nach dem Start der inversen Kinematiksuche wird die Spaltensumme des Positionsvektors von dem Tool Center Point aufgezeichnet. Abbildung 6.1 zeigt den Verlauf dieses Messwertes für die ersten 350 Iterationsschritte. Bei jeder Wiederholung des Experimentes ist der Geschwindigkeitsfaktor c ausgehend von $c = 0{,}1$ erhöht. Bei Werten zwischen 0,4 und 0,8 zeigt das Verfahren gute Konvergenzgeschwin-

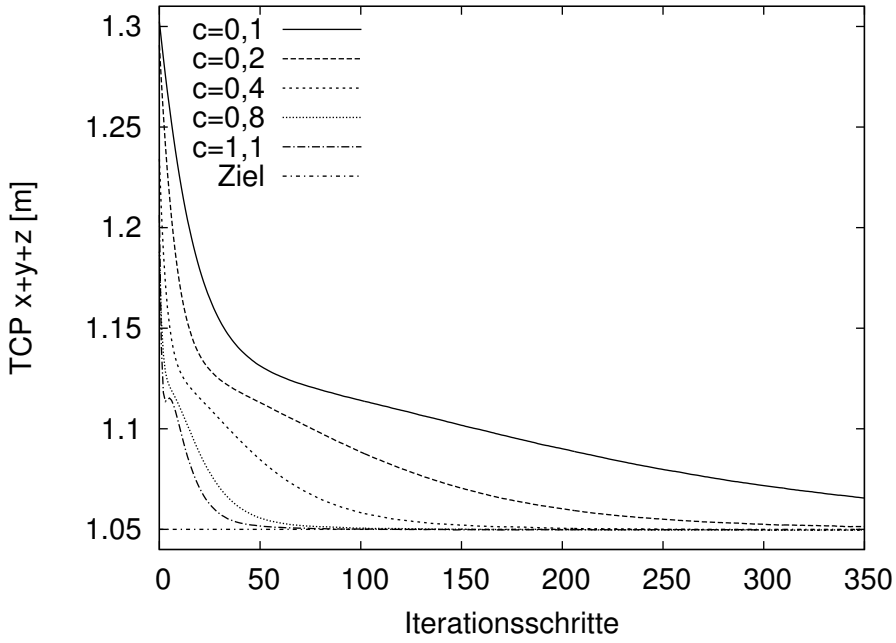


Abbildung 6.1: Suchlauf der inversen Kinematik mit unterschiedlichen Werten für den Geschwindigkeitsfaktor c, wobei d$x = 0{,}5°$, $\epsilon = 0{,}0001$ und $\beta = 0{,}1$ konstant sind.

digkeiten von etwa 40 bis 100 Iterationen für das Unterschreiten der 1-Zentimeter-Marke. Das Experiment ist auf einem Computer mit einem Pentium M Hauptprozessor mit 1,6 Gigahertz durchgeführt worden. Dieser hat für jeden Iterationsschritt durchschnittlich etwa 7,6 Millisekunden gebraucht. Durch das Abschalten der Kollisionsvermeidung kann dieser Wert sogar auf 5,2 Millisekunden verbessert werden. Für $c > 1$ neigt die Suche zu unstabilem Verhalten und liefert keine guten Lösungen. Bei Veränderungen der Werte für dx zwischen 0,01° und 45° sind ähnliche Auswirkungen in Abbildung 6.2 zu erkennen.

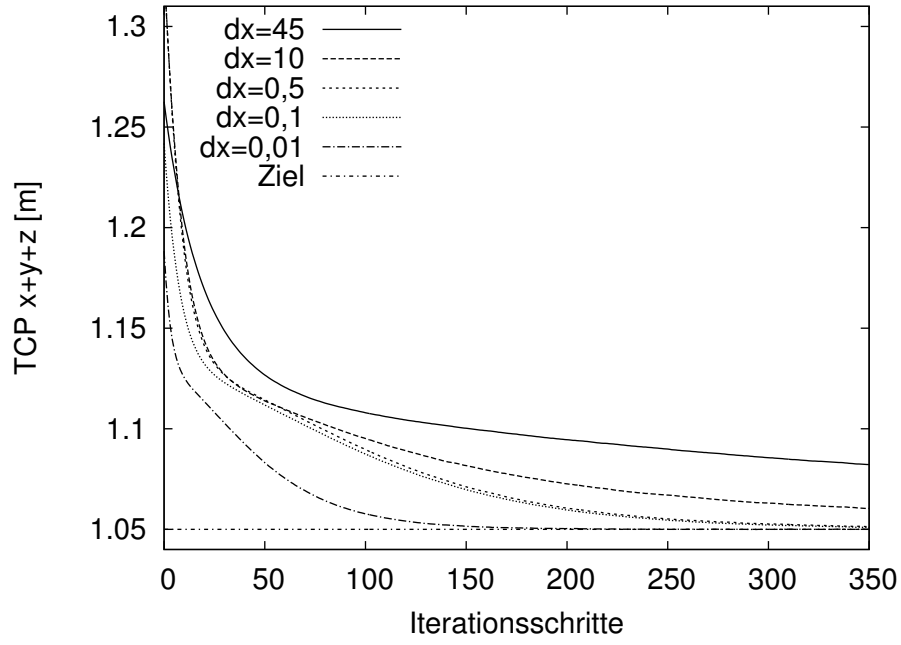


Abbildung 6.2: Suchlauf der inversen Kinematik mit unterschiedlichen Werten für den Differenzenquotient-Abstand dx, wobei $c = 0{,}2$, $\epsilon = 0{,}0001$ und $\beta = 0{,}1$ konstant sind.

6.2 Geschwindigkeit der Kommunikation

Eine Messung der Übertragungsdauer der UDP/IP-Datenpakete $\vec{s}$ in dem verwendeten Computer-Netzwerk hat einen Wert von unter einer Millisekunde ergeben. Dieses ist ein sehr positives Ergebnis. Unter dieser Bedingung besteht generell die Möglichkeit, die Werte für die haptische Ausgabe innerhalb eines haptischen Ausgabezyklus (1000 Hertz) zu übertragen. Mit dieser geringen Zeitverzögerung sollte der Benutzer das System als sehr transparent wahrnehmen. Bei praktischen Experimenten hat sich jedoch gezeigt, dass diese ideale Zeitverzögerung bei der Datenübertragung nicht erreicht wird. Die Dauer der nötigen Verarbeitungsschritte beim Senden und Empfangen der Pakete ist größer und muss zur Dauer des Übertragungsprozesses hinzugezählt werden. Die Anzahl der tatsächlich empfangenen Pakete pro Sekunde beträgt durchschnittlich etwa 48 Pakete. Eine Messung bei geringer Netzwerkauslastung durch andere Anwendungen hat ergeben, dass die Pakete in sehr gleichmäßigen Zeitabständen empfangen werden. Es sind demnach ungefähr zwanzig Millisekunden nötig, um den folgenden Prozess zu durchlaufen.

- Aktuelle Zeit in einer Variablen speichern.
- Die aktuellen Gelenkwinkel $\vec{s}$ der Hardware auslesen.

- Werte in die Datenstruktur eines Paketes übertragen.
- Paket versenden.
- Paket übertragen.
- Paket empfangen.
- Typ des Paketes identifizieren.
- Byte-Werte in Java-Variablen umwandeln.
- Gelenkwinkel in dem virtuellen Modell des realen Manipulators speichern.

Danach muss noch die Vorwärtskinematik berechnet werden, bevor eine Kraftausgabe erzeugt werden kann. Bei einer so großen Zeitverzögerung ist eine stotternde Bewegung des haptischen Geräts zu bemerken.

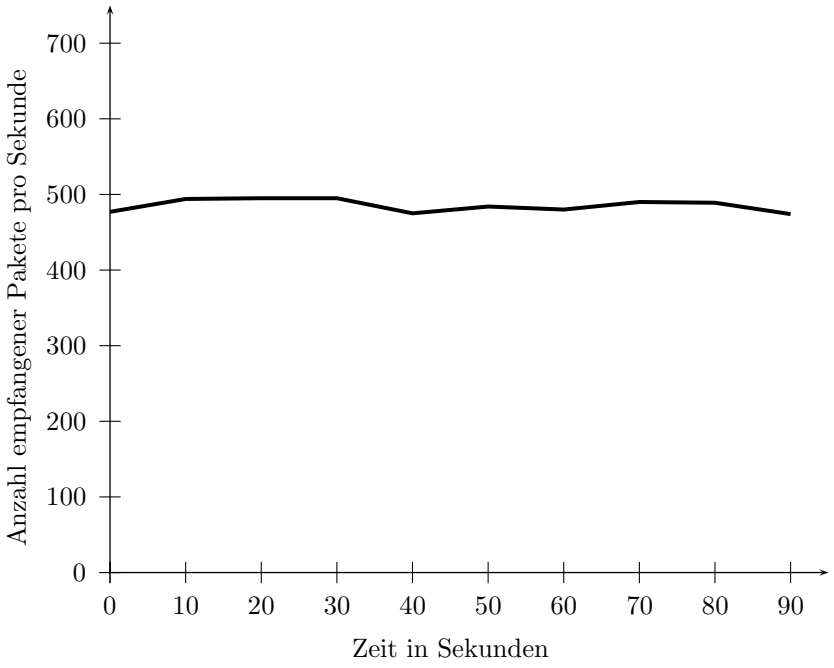


Abbildung 6.3: Messungsergebnis der Datenpaketrate.

Um diesem Problem entgegenzuwirken und dem Benutzer eine bessere Transparenz zu bieten, kann das virtuelle Modell der Eingabe für die haptische Ausgabe verwendet werden, statt dem Modell des realen Manipulators. Somit ist nicht nur die grafische Ausgabe, sondern auch die haptische Ausgabe, vorhersagend im Sinne eines Predictive-Displays. Die Kraftausgabe erfolgt dadurch stotterfrei. Darüber hinaus wirkt diese Verkürzung des Rückkopplungsweges stabilisierend auf das System, da weniger Komponenten rückgekoppelt werden, die eine variierende Zeitverzögerung erzeugen.

6.3 Prozessor-Auslastung des Roboters

Der Telemanipulations-Server muss mindestens für die Dauer der Telemanipulation auf dem Roboter ausgeführt werden. Verschiedene andere Prozesse müssen parallel dazu ausgeführt werden, die für den allgemeinen Betrieb des Roboters nötig sind. Der Server ist daher so konzipiert, dass er verhältnismäßig wenig Rechenzeit in Anspruch nimmt. Bei dem verwendeten Computer des Roboters, einem Pentium IV mit 2,4 Gigahertz Taktfrequenz, sind Messungen der Prozessorauslastung durchgeführt worden. Dabei kann prozentual die anteilig verbrauchte Rechenzeit, die sogenannte CPU-Auslastung, einzelner Prozesse untersucht werden. Die Messungen über einen Zeitraum von einer Stunde haben das Folgende ergeben.

Ohne Verbindung zu einem Telemanipulations-Client liegt die benötigte CPU-Auslastung des Telemanipulations-Servers bei unter einem Prozent. Doch auch bei normaler Belastung des Servers durch mehrere verbundene Clients und die Durchführung von Telemanipulationen steigt dieser Wert nicht über ein Prozent hinaus. Dieses Ergebnis zeigt, dass der entwickelte Telemanipulations-Server seine Anforderungen erfüllt. Sogar seine dauerhafte Ausführung als Hintergrundprozess, die bei Server-Anwendungen üblich ist, ist möglich.

6.4 Intuitive Bedienbarkeit

Es sind Versuche durchgeführt worden, in denen fünf verschiedene Personen einfache Telemanipulations-Aufgaben ausführen sollten. Die Personen haben das System vorher nicht gekannt. Die Aufgaben haben das Berühren von mehreren Objekten an unterschiedlicher Position und aus unterschiedlicher Richtung beinhaltet. Abbildung 6.4 zeigt die Anordnung dieser Objekte, dargestellt als Kugeln. Dabei handelt es sich um ein Objekt auf dem Fußboden, zwei Objekte in Tischhöhe und eines in Kopfhöhe. Bei dem Experiment hat jeder Proband den Manipulator so steuern müssen, dass jedes Objekt zehn mal nacheinander berührt worden ist. Dabei sind den Probanden nur die Bildschirmdarstellung des virtuellen Modelles und die haptischen Ausgaben geboten worden. In die Bildschirmdarstellung sind zu diesem Zweck die einzelnen Objekte integriert worden.

Insgesamt sind bei diesem Experiment 120 einzelne Trajektorien aufgezeichnet worden. Alle Probanden sind dabei in der Lage gewesen, die gestellte Aufgabe nach einigen Testläufen auszuführen. Die Lernphase für die Bedienung hat unterschiedlich viele Testläufe in Anspruch genommen. Dieses hat scheinbar hauptsächlich mit der ungewohnten dreidimensionalen Darstellung zu tun. Auch die Tatsache, dass nicht

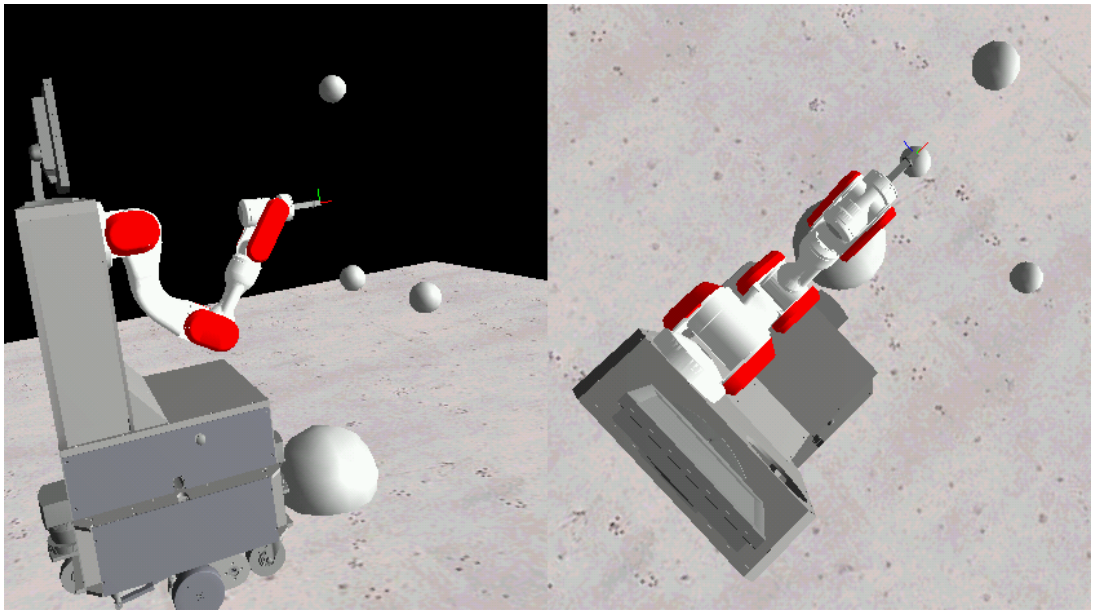


Abbildung 6.4: Anordnung der Telemanipulations-Aufgaben zur Untersuchung der intuitiven Bedienbarkeit. Probanden müssen den Manipulator zur Berührung der vier kugelförmigen Objekte steuern.

jedem Proband die Eigenschaften des Manipulators, wie die Beschränkungen einzelner Gelenkwinkel, bekannt gewesen sind, hat teilweise zu Verwirrung geführt. Es wurden jedoch nie mehr als drei Übungsdurchläufe benötigt, was dafür spricht, dass die Eingabemethode sehr intuitiv bedient werden kann.

6.5 Genauigkeit der Telemanipulation

Bei Experimenten mit realen Telemanipulationen kann erneut die Spaltensumme der Endeffektor-Position gemessen werden, um die Korreliertheit der Manipulatorbewegung mit der Eingabe zu untersuchen. Als Beispiel-Manipulation soll der Transport eines Objektes zwischen zwei Positionen auf einem Tisch betrachtet werden. Diese Aufgabe ist einfach genug, um häufiger trainiert und ausgeführt zu werden. Dennoch enthält sie viele wichtige Teilaspekte typischer Telemanipulationen. Diese umfassen die exakte Ansteuerung eines bestimmten Punktes im Raum aus einer bestimmten Richtung, das Schließen des Greifers, ein erneutes Ansteuern eines zweiten Punktes bei Einhaltung einer bestimmten Orientierung auf der kompletten Trajektorie und das öffnen des Greifers. Bei all diesen Bewegungen ist zu beachten, dass auf das zu greifende Objekt keine zu großen Kräfte wirken dürfen. Es könnte sich dabei um ein zerbrechliches Glas oder um eine Tasse, deren Inhalt nicht verschüttet werden

darf, handeln. Darüber hinaus dürfen andere Objekte, wie der Tisch oder andere auf ihm befindliche Gegenstände nicht berührt werden. Aus Sicherheitsgründen wird hierfür ein künstliche Versuchsumgebung benutzt, in der zwei robuste, gut handhabbare Gegenstände auf einem gepolsterten Tisch stehen. Zur Verdeutlichung ist in Abbildung 6.5 eine Folge von Fotos präsentiert, die den kompletten Versuchsablauf zeigen.

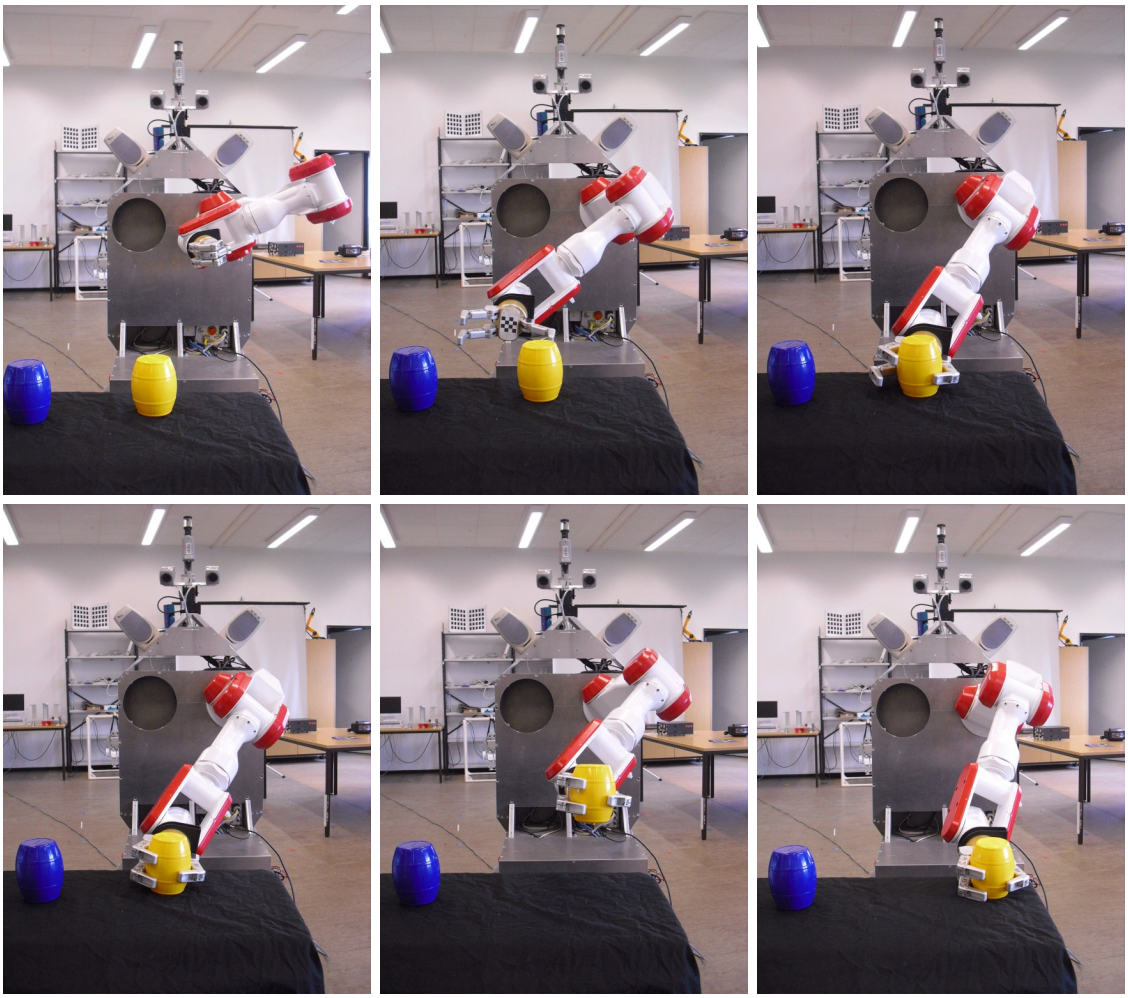


Abbildung 6.5: Fotofolge einer Telemanipulation.

Deutlich ist auf den oberen Fotos erkennbar, dass der Manipulator dem gelben Gegenstand nahe genau kommt, um ihn mit dem Greifer erfassen zu können. Der Benutzer schafft es dabei, den Manipulator so präzise zu steuern, dass der Gegenstand nicht umfällt, obwohl dieser auf dem weichen und unebenen Polster keinen stabilen Stand hat. In der unteren Reihe ist der Transport des Gegenstandes und sein zufriedenstellendes Ablagern dargestellt. Zwischen einzelnen Wiederholungen derartiger Versuche können die Parameter des Systems bestimmten Bedürfnissen des Benutzers angepasst

werden. Dabei zeigt sich, dass der Verstärkungsfaktor der Kraftausgabe und die Konvergenz-Geschwindigkeiten der Manipulator-Regelschleife und der Suche der inversen Kinematik einen erheblichen Einfluss auf die Stabilität und die Genauigkeit haben. Eine geringere Kraftausgabe des haptischen Gerätes minimiert zum Beispiel das Auftreten von Resonanzschwingungen des kompletten Systems. Diese entstehen bei der Verstärkung von Signalen in rückgekoppelten Systemen. Eine stärkere Kraftausgabe hilft dem Benutzer jedoch die Manipulatorposition genauer wahrzunehmen. Genau wie bei der inversen Kinematik und den beiden Regelschleifen zur Hardwaresteuerung können für alle Systemparameter geeignete Kompromisse zwischen Stabilität und Genauigkeit experimentell ermittelt werden.

In Abbildung 6.6 sind die Trajektorien der Eingabe (Ziel), der Steuerbefehle aus der inversen Kinematiksuche und der resultierenden Manipulatorbewegung dargestellt. Die abgebildeten Werte entsprechen auch hierbei der Spaltensumme der Raumpositionen. Diese Werte sind während der zuvor beschrieben Telemanipulation in Abständen von zehn Millisekunden gemessen worden.

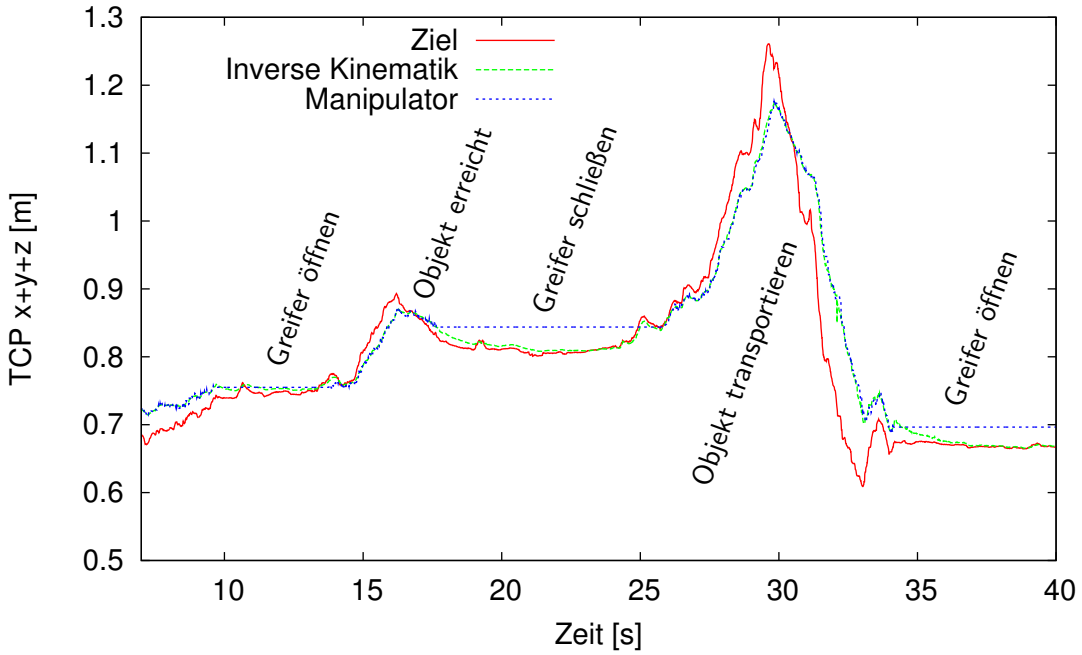


Abbildung 6.6: Bewegungsverlauf bei einer Telemanipulation.

Zur Erinnerung, diese drei Werte sind in der folgenden Reihenfolge miteinander gekoppelt. Der Benutzer steuert das Ziel, aus dem dann die inverse Kinematik berechnet wird. Diese steuert dann über das Netzwerk den Manipulator, der wiederum über das Netzwerk die Kraftausgabe an den Benutzer, und damit die Eingabe, beeinflusst (siehe Abbildung 5.12). Die Trajektorie der inversen Kinematiksuche zeigt eine leichte

Tiefpasswirkung, welche durch ihre zeitlich verzögernde Berechnung und durch eventuell unlösbare Zieleingaben entsteht. Die Manipulatortrajektorie entspricht hingegen sehr genau den Steuerungswerten aus der inversen Kinematik. Die einzigen Abweichungen sind an den Stellen zu erkennen, an denen der Greifer aktiv ist. Das ist jedoch eine vorgesehene, implementationsbedingte Schwäche des Systems. Die Implementation Echtzeit-Hardwaresteuerung, die die Hand und den Manipulator parallel kontrolliert, ist in dem zeitlichen Rahmen dieser Arbeit nicht vorgesehen gewesen. Um dennoch Greifbewegungen ausführen zu können, werden einfache vorhandene Funktionen benutzt, die den Manipulator blockieren bis sie beendet sind.

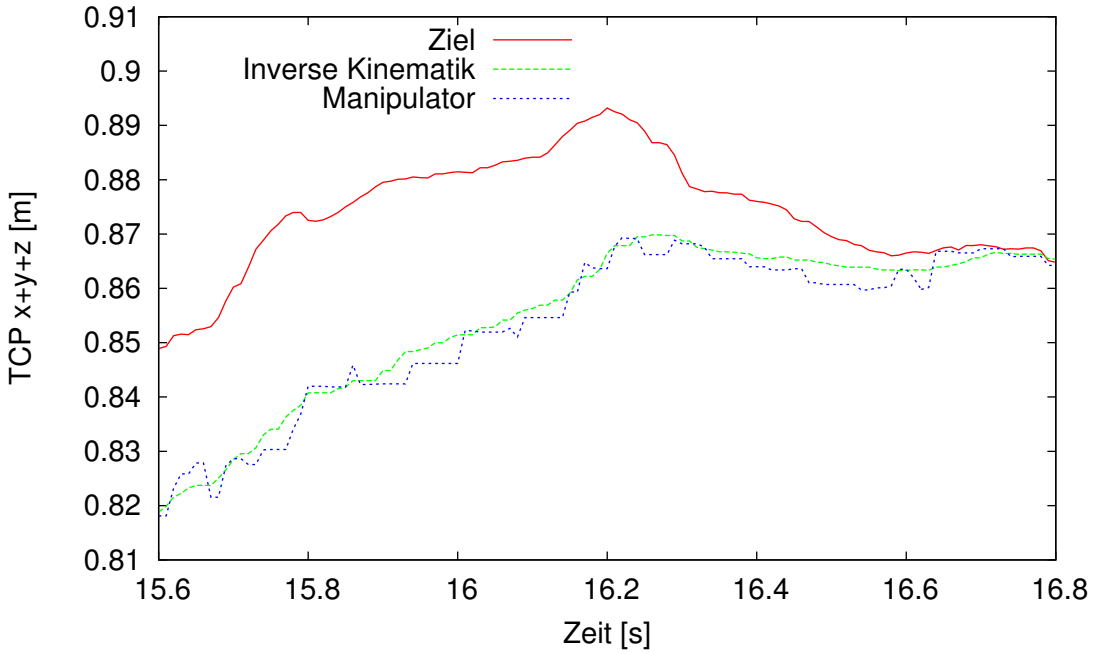


Abbildung 6.7: Stark vergrößerter Bewegungsverlauf bei einer Telemanipulation.

Doch aus dem restlichen Manipulatorverlauf geht hervor, dass die Datenübertragung und die Hardwaresteuerung des Manipulators sehr gute Ergebnisse erzielen. Nur bei einer stärker aufgelösten Betrachtung der selben Trajektorien sind die kleinen Abweichungen des Manipulators zu erkennen. Die maximale Fehler beträgt dabei nur 8 Millimeter und ist nach maximal 20 Millisekunden behoben. Siehe hierzu Abbildung 6.7.

7 Fazit und Ausblick

Die vorliegende Arbeit dokumentiert die Entwicklung eines neuen interaktiven Telemanipulations-Systems. Drei besondere Aspekte der Implementation sind die portable, eigenständige grafische Bedienoberfläche in Form einer Clientanwendung, die dreidimensionale Roboter-Simulation mit grafischer Darstellung und die Integration des Systems in einen autonomen Serviceroboter. Der autonome Serviceroboter wird dadurch um eine Möglichkeit der manuellen, unautonomen Steuerung erweitert, die es bei ihm bisher nicht gegeben hat.

Die Nutzung des Netzwerkprotokolls Internet Protocol hat zu der Verwendung einer erweiterbaren Client/Server-Struktur mit einem neuen Telemanipulations-Protokoll auf der Anwendungs-Protokollschicht geführt. Dieses Protokoll ermöglicht eine schnelle Übertragung der Steuerungsdaten über UDP/IP. Die Fragestellung der Realisierbarkeit eines solchen Systems ist diskutiert und positiv beantwortet worden. Dieses konnte auch durch Experimente bestätigt werden. Für Telemanipulations-Server ist eine generelle Architektur vorgestellt worden, die anhand einer Beispielimplementation untersucht worden ist. Der entwickelte Telemanipulations-Server ist besonders im Hinblick auf die Effizienz und die Schonung von Resourcen entworfen worden. Seine klare strukturelle Einfachheit bietet Möglichkeiten der Erweiterbarkeit und Wiederverwendbarkeit.

Zusätzlich wurde eine Telemanipulations-Client-Anwendung geschaffen, die in dieser ersten Version nur einen Mitsubishi PA10-6C simulieren kann. In Verbindung mit dem entwickelten Server kann der reale Manipulator mit dieser Client-Anwendung interaktiv gesteuert werden. Die Verwendung standardisierter Datenformate ermöglicht es, diese Anwendung für beliebige andere Manipulatoren mit serieller kinematischer Kette anzupassen. Durch die Denavit-Hartenberg-Parameter kann das System an andere Manipulatoren angepasst werden. Deren geometrische Daten können durch entsprechende Standard-Dateiformate für dreidimensionale Geometrien eingebunden werden. Durch Ansätze der modularen Programmierung können auch die Komponenten der Client-Anwendung einzeln ausgetauscht werden. So können zum Beispiel Anbindungen an neue Eingabegeräte oder neue Verfahren für die Lösung der inversen Kinematik oder die Kollisionsvermeidung integriert werden, ohne die ganze Anwendung zu verändern. Die Realisierung der Lösung der inversen Manipulator-Kinematik

in der Client-Anwendung bietet sowohl Vorteile, bei der Interaktivität und der Predictive-Display-Technik, als auch Nachteile bei der Effizienz. Doch besonders die daraus resultierende Nutzungsmöglichkeit zu Simulations-, Planungs- und Trainingszwecken und die offene Client/Server-Struktur sorgen dafür, dass das gesamte System als eine Art Rahmenwerk für weitere Forschungen und Entwicklungen angesehen werden kann.

Die Struktur und Funktionalität ist unter Berücksichtigung von aktuellen Forschungsergebnissen aus dem Bereich Telerobotik entwickelt worden. Die experimentellen Ergebnisse dieser Arbeit bestätigen dabei viele Untersuchungsergebnisse anderer Arbeiten zu diesem Thema. Die Fragen der Stabilität und Transparenz von Internet-Telemanipulations-Systemen sind nicht allgemein gültig gelöst worden. Es sind lediglich experimentell optimierte Bedingungen für die Arbeit mit dem Telemanipulator geschaffen worden. Es konnte gezeigt werden, dass der Hauptverursacher von auftretenden Genauigkeitsproblemen der untersuchten Telemanipulationen das Verfahren zur Suche der inversen Kinematik ist. Die Netzwerkkommunikation und die Hardwaresteuerung erzielen viel genauere Ausgaben.

Ausblick

Nachdem die Ergebnisse dieser Arbeit gezeigt haben, dass das Telemanipulations-System erfolgreich realisiert werden kann, sollen in diesem Abschnitt mögliche, sinnvolle Erweiterungen und Verbesserungen des entwickelten Systems aufgezeigt werden. Die bestehende Implementation sollte auch über diese Diplomarbeit hinaus weiter entwickelt werden. Es hat bereits vor der Fertigstellung dieses Textes und der Beispiel-Implementation andere Forschungs- und Entwicklungsarbeiten gegeben, die auf die Ergebnisse der vorliegenden Arbeit aufbauen. Hierbei geht es unter anderem um das maschinelle Lernen der Trajektorien von Manipulations-Aufgaben oder die kollisionsvermeidende Steuerung des zweiten Manipulators. Im Folgenden werden einige Vorschläge, für die weitere Verwendung des Telemanipulations-Systems diskutiert. Dieser Ausblick beschreibt dabei wichtige Verbesserungen, die generelle Weiterverwendbarkeit des Systems und die Anregungen zu grundsätzlichen Forschungen, die mit diesem System zu tun haben.

In den Bereich der wichtigen Verbesserungen fällt zum Beispiel die Implementation und der praktische Vergleich anderer Verfahren zur Lösung der inversen Kinematik. Eine einfache Erweiterung des entwickelten Verfahrens könnte zum Beispiel durch die parallele Ausführung mehrerer unabhängiger Suchläufe realisiert werden. Diese unterscheiden sich dabei in ihren Parametern und ihren Startwerten. Dadurch könn-

ten Probleme mit lokalen Minima gelöst werden. Auch wäre eine Auslagerung dieses Verfahrens in einen C++-Programmteil denkbar, um eine effizientere Programmierung zu ermöglichen. Eine allgemein effizientere Implementierung dieser Komponente würde die Genauigkeit des Systems erhöhen.

Auch eine Verbesserte Einbindung der Barrett-Hand und besonders ihrer Kraftsensoren würde den Nutzen des Gesamtsystems sehr stark erhöhen. Zu diesem Zweck sollte die lineare Datenstruktur des virtuellen Modells des Manipulators in eine baumförmige Datenstruktur umgewandelt werden, um auch die Finger der Hand komplett modellieren zu können. Die Hardware-Steuerung sollte sich dabei an der Hardware-Steuerung des Manipulators orientieren, denn diese hat sehr gute Ergebnisse gezeigt (siehe 5.2.4).

Ein neu entwickeltes Software-System benötigt generell einen gewissen Wartungsaufwand. Es gibt beispielsweise neue Versionen der Haptik-Programmier-Bibliotheken OpenHaptics und JTouchToolkit. Es wäre interessant zu untersuchen, ob mit diesen eine bessere Steuerung des haptischen Gerätes mit Java möglich ist. Auch eine noch stärkere Integration von anderen Funktionen des Roboters wäre denkbar. Zum Beispiel könnte die Fahrtkontrolle der mobilen Plattform in die Telemanipulations-Anwendung einbezogen werden.

Weiterhin ist es erwähnenswert, dass für den entwickelten Server und den Client auch einzeln Verwendungsmöglichkeiten bestehen. So könnten sich zum Beispiel andere Anwendungen den Telemanipulations-Server und das Telemanipulations-Netzwerkprotokoll zunutze machen, um die aktuelle Stellung des Roboterarmes zu überwachen, oder sogar um Steuerungsbefehle zu senden. Der Server schränkt solche Zugriffe nicht ein. Auch der Client kann, wie bereits vielfach erwähnt, zum Beispiel zur Planung oder einfach zur Visualisierung genutzt werden.

Interessante Untersuchungen mit dem System könnten sich mit den Problemen der Steuerung bei größeren Distanzen Generell häufige Nutzung und Untersuchung

Die bereits in Abschnitt 3.7 angesprochene umfangreiche, experimentelle Untersuchung der Systemeigenschaften im Spektralbereich, könnte wertvolle Hinweise für neue Ansätze zur Stabilisierung von Telemanipulations-Systemen liefern.

Literaturverzeichnis

[Adelstein u. a. 2002] Adelstein, Bernard D. ; Colgate, J. E. ; Hannaford, Blake ; Hollerbach, John: 10th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, IEEE Computer Society, März 2002. – ISBN 0-7695-1489-8

[Anderson und Spong 1989] Anderson, Robert J. ; Spong, Mark W.: Bilateral Control of Teleoperators with Time Delay. In: *IEEE Transactions on Automatic Control* Bd. 34, 1989, S. 494–501

[Archer 2007] Archer, Ian J.: *Java Haptik API JTouchToolkit*. Birmingham City Univertity, User-Lab. 2007. – Internet: `https://jtouchtoolkit.dev.java.net/` Zugriff Dezember 2008

[Baier u. a. 2006] Baier, Tim ; Hüser, Markus ; Zhang, Jianwei: Learning of Demonstrated Grasping Skills by Stereoscopic Tracking of Human Hand Configuration. In: *IEEE International Conference on Robotics and Automation, ICRA*, Mai 2006, S. 2795–2800

[Bartsch 2004] Bartsch, Hans-Jochen: *Taschenbuch Mathematischer Formeln*. 20. Fachbuchverlag Leipzig im Carl Hanser Verlag, 2004. – ISBN 3-446-22891-8

[Berestesky u. a. 2004] Berestesky, B. ; Chopra, N. ; Spong, Mark W.: Discrete Time Passivity in Bilateral Teleoperation over the Internet. In: *IEEE International Conference on Robotics and Automation*, 2004, S. 4557–4564

[Bilbao 2004] Bilbao, Stefan D.: *Wave and Scattering Methods for Numerical Simulation*. S. 25–48, John Wiley and Sons, 2004. – ISBN 0-470-87017-6

[Buzan und Sheridan 1989] Buzan, Forrest T. ; Sheridan, Thomas B.: A Model-Based Predictive Operator Aid for Telemanipulators with Time Delay. In: *IEEE International Conference on Man and Cybernetics Systems* Bd. 1, 1989, S. 138–143

[Craig 2005] Craig, John J.: *Introduction to Robotics: Mechanics and Control, International Edition*. 3. Upper Saddle River, Pearson, Prentice Hall, 2005. – ISBN 0-13-123629-6

[Ehrsson 2007] Ehrsson, H. H.: The Experimental Induction of Out-of-Body Experiences. In: *Science* 317 (2007), Nr. 5841, S. 1048

[Elias 2004] Elias, Hugo: *Inverse Kinematics - Improved Methods*. Juli 2004. – Internet: `http://freespace.virgin.net/hugo.elias/models/m_ik2.htm` Zugriff November 2008

[Ferre 2008] Ferre, Manuel: Haptics: Perception, Devices and Scenarios, 6th International Conference, EuroHaptics, Springer, 2008. – ISBN 3-540-69056-5

[Ferre u. a. 2007] Ferre, Manuel ; Buss, Martin ; Aracil, Rafael ; Melchiorri, Claudio ; Balaguer, Carlos: *Springer Tracts in Advanced Robotics*. Bd. 31: *Advances in Telerobotics*. Springer-Verlag Berlin Heidelberg, 2007. – ISBN 3-540-71363-8

[Furuta u. a. 1987] Furuta, Katsuhisa ; Kosuge, Kazuhiro ; Shiote, Yoshinori ; Hatano, Hiromu: Master-Slave Manipulator based on Virtual Internal Model Following Control Concept. In: *Proceedings of IEEE Intanational Conference on Robotics and Automation*, IEEE Computer Society, 1987, S. 567–572

[Gat u. a. 1994] Gat, Erann ; Desai, Rajiv ; Ivlev, Robert ; Loch, John ; Miller, David P.: Behavior Control for Robotic Exploration of Planetary Surfaces. In: *IEEE Transactions on Robotics and Automation* Bd. 10. 1994, S. 490–503

[Ghodoussi u. a. 2002] Ghodoussi, Moji ; Butner, Steven E. ; Wang, Yulun: Robotic Surgery - the Transatlantic Case. In: *IEEE International Conference on Robotics and Automation* Bd. 2, 2002, S. 1882–1888. – ISBN 0-7803-7272-7

[Girod u. a. 1997] Girod, Bernd ; Rabenstein, Rudolf ; Stenger, Alexander: *Einführung in die Systemtheorie*. Teubner, 1997. – ISBN 3-519-06194-5

[Goldberg und Siegwart 2002] Goldberg, Ken ; Siegwart, Roland: *Beyond webcams : an introduction to online robots*. MIT Press, Cambridge, Mass., 2002. – ISBN 0-262-07225-4

[Handlykken und Turner 1980] Handlykken, M. ; Turner, T.: Control System Analysis and Synthesis for a Six Degree-of-Freedom Universal Force-Reflecting Hand Controller. In: *19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, Dezember 1980, S. 1197–1205

[Hannaford 1989a] Hannaford, Blake: A Design Framework for Teleoperators with Kinesthetic Feedback. In: *IEEE Transactions on Robotics and Automation* 5 (1989), S. 426–434

[Hannaford 1989b] Hannaford, Blake: Stability and Performance Tradeoffs in Bi-lateral Telemanipulation. In: *IEEE International Conference on Robotics and Automation*, 1989, S. 1764–1767

[Hashtrudi-Zaad und Salcudean 1996] Hashtrudi-Zaad, K. ; Salcudean, S. E.: Adaptive Transparent Impedance Reflecting Teleoperation. In: *IEEE Intemational Conference on Robotics and Automation*, April 1996, S. 1369–1374

[Hayward und Paul 1984] Hayward, Vincent ; Paul, Richard P.: Introduction to RCCL: A Robot Control CLibrary. In: *IEEE International Conference on Robotics and Automation* Bd. 1, 1984, S. 293–297

[JavaSoft 2002] JavaSoft, a Sun Microsystems, Inc. Business: *The Java 3D API Specification*. 1.3. 901 San Antonio Road Palo Alto, CA 94303 USA: , 2002. – Internet: `http://java.sun.com/javase/technologies/desktop/java3d/` Zugriff November 2008

[Kosuge u. a. 2002] Kosuge, Kazuhiro ; Kikuchi, Jun ; Takeo, Koji: *Beyond Webcams*. Kap. VISIT: A Teleoperation System via the Computer Network, S. 215–226, Massachusetts Institude of Technology, 2002. – ISBN 0-262-07225-4

[Krüger und Stark 2007] Krüger, Guido ; Stark, Thomas: *Handbuch der Java-Programmierung*. 5. Addison-Wesley, 2007. – Internet: `http://www.javabuch.de` Zugriff Oktober 2008. – ISBN 3-8273-2373-8

[Lawrence 1993] Lawrence, Dale A.: Stability and Transparency in Bilateral Teleoperation. In: *IEEE Transactions an Robotics and Automation* 9 (1993), Nr. 5, S. 624–637

[Lee und Suh 2006] Lee, Dai G. ; Suh, Nam P.: *Axiomatic Design and Fabrication of Composite Structures*. S. 513, Oxford University Press, 2006. – ISBN 0195178777

[Lloyd und Hayward 1995] Lloyd, John ; Hayward, Vincent: *Multi-RCCL User's Guide*. RCCL Release 4.2, 1995. – Internet: `http://www.cs.ubc.ca/~lloyd/rccl.html` Zugriff März 2009

[Luenberger 2003] Luenberger, David G.: *Linear and Nonlinear Programming*. S. 227–230, Springer, 2003. – ISBN 1402075936

[Mitra u. a. 2007] Mitra, Probal ; Gentry, Diana ; Niemeyer, Günter: User Perception and Preference in Model Mediated Telemanipulation. In: *Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2007, S. 268–273

[Ni und Wang 2002] Ni, Liya ; Wang, David W. L.: A Gain-Switching Control Scheme for Position-Error-Based Force-Reflecting Teleoperation. In: *10th International Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems*, IEEE Computer Society, 2002, S. 239–246. – ISBN 0-7695-1489-8

[Niemeyer und Slotine 1991] Niemeyer, Günter ; Slotine, Jean-Jacques E.: Stable Adaptive Teleoperation. In: *IEEE Journal of Oceanic Engineering* 16 (1991), Januar, Nr. 1, S. 152–162

[OpenHaptics 2004] SensAble Technologies Inc.: *OpenHaptics Toolkit*. 2004. – Internet: `http://www.sensable.com/products-openhaptics-toolkit.htm` Zugriff Dezember 2008

[Park 2008] Park, Diego: *Inverse Kinematics*. Dezember 2008. – Internet: `http://diegopark.googlepages.com/computergraphics/` Zugriff November 2008

[Postel 1980] Postel, Jon: *UDP/IP, RFC 768*. 1980. – Internet: `http://tools.ietf.org/html/rfc768` Zugriff November 2008

[Postel 1981] Postel, Jon: *TCP/IP, RFC 793*. 1981. – Internet: `http://tools.ietf.org/html/rfc793` Zugriff November 2008

[Preusche und Hirzinger 2007] Preusche, Carsten ; Hirzinger, Gerd: Haptics in Telerobotics - Current and Future Research and Applications. In: *Visual Comput* Bd. 23. Springer-Verlag, März 2007, S. 273–284

[Schlag und Graschew 1999] Schlag, Peter M. ; Graschew, G.: Tele- und Computergestützte Chirurgie. In: *Symposium Chirgurgie im 21. Jahrhundert*, Springer Berlin, 1999. – ISBN 3-540-65342-2

[Schnell und Strasser 2008] Schnell, Christopher ; Strasser, Sascha: *Java 3D Ein Überblick der API*, November 2008. – Internet: `http://java3d.j3d.org/tutorials/` Zugriff November 2008

[Schulzrinne u. a. 2003] Schulzrinne, H. ; Casner, S. ; Frederik, R. ; Jacobson, V.: *RTP, RFC 3550*. 2003. – Internet: `http://tools.ietf.org/html/rfc3550` Zugriff Dezember 2008

[Shen u. a. 2007] Shen, Weimin ; Gu, Jason ; Ma, Yide: 3D Kinematic Simulation for PA10-7C Robot Arm Based on VRML. In: *IEEE International Conference on Automation and Logistics*, August 2007, S. 614–619. – ISBN 978-1-4244-1531-1

[Smidt 1998] Smidt, Wolfgang: *Verallgemeinerte inverse Kinematik für Anwendungen in der Robotersimulation und der virtuellen Realität*. Diplomarbeit. 1998. – Internet: `http://www.worldwidewolf.de/diplom.pdf` Zugriff November 2008

[Stahl und Robat 2008] Stahl, Ted ; Robat, Cornelis: *Timeline of Robotics*. Dezember 2008. – Internet: `http://www.thocp.net/reference/robotics/robotics.html` Zugriff Januar 2009

[Stevens 2004] Stevens, W. R.: *TCP/IP : der Klassiker; Protokollanalysen, Aufgaben und Lösungen*. Hüthig, 2004. – ISBN 3-8266-5042-5

[Tanenbaum 2003] Tanenbaum, Andrew S.: *Computernetzwerke*. Pearson Studium, 2003. – ISBN 3-8273-7046-9

[Thomas u. a. 2002] Thomas, U. ; Maciuszek, I. ; Wahl, F. M.: *Eine Systematik zur universellen Beschreibung für serielle, parallele und hybride Roboterstrukturen*. 2002. – Universität Braunschweig, VDI-Berichte Robotik

[Uchimura und Yakoh 2004] Uchimura, Yutaka ; Yakoh, Takahiro: Bilateral Robot System on the Real-Time Network Structure. In: *IEEE Transactions on Industrial Electronics* Bd. 51, Oktober 2004, S. 940–946

[Ullenboom 2008] Ullenboom, Christian: *Java ist auch eine Insel*. 7. Galileo Computing, 2008. – Internet: `http://openbook.galileocomputing.de/javainsel7/` Zugriff November 2008. – ISBN 978-3-8362-1146-8

[Vertut u. a. 1981] Vertut, J. ; Micaelli, A. ; Marchal, P. ; Guittet, J.: *Short Transmission Delay on a Force Reflective Bilateral Manipulator*. 4th Rom-An-Sy. 1981

[Wang und Chen 1991] Wang, Li-Chun T. ; Chen, Chih C.: A Combined Optimization Method for Solving the Inverse Kinematics Problem of Mechanical Manipulators. In: *IEEE Transactions on Robotics and Automation* Bd. 7, August 1991, S. 489–499

[Ward 2008] Ward, Jamie: *The Frog Who Croaked Blue*. Routledge. März 2008. – Internet: `http://www.thefrogwhocroakedblue.com` Zugriff März 2009

[Wegener 2005] Wegener, Ingo: *Lecture Notes in Computer Science*. Bd. 3580. Kap. Simulated Annealing Beats Metropolis in Combinatorial Optimization, S. 589–601, Springer, 2005. – ISBN 978-3-540-27580-0

[Wirz u. a. 2008] Wirz, Raul ; Ferre, Manuel ; Marín, Raul ; Barrio, Jorge ; Claver, José M. ; Ortego, Javier: Efficient Transport Protocol for Networked Haptics Applications. In: Ferre, Manuel (Hrsg.): *Haptics: Perception, Devices and Scenarios, 6th International Conference, EuroHaptics*, Springer, 2008, S. 3–12. – ISBN 3-540-69056-5

[Yashiro und Ohnishi 2008] Yashiro, Daisuke ; Ohnishi, Kouhei: L_2 Stable Four-Channel Control Architecture for Bilateral Teleoperation with Time Delay. In: *10th IEEE International Workshop on Advanced Motion Control*. 2008, S. 324–329. – ISBN 978-1-4244-1702-5

[Yokokohji u. a. 2002] Yokokohji, Yasuyoshi ; Tsujioka, Teruhiro ; Yoshikawa, Tsuneo: Bilateral Control with Time-Varying Delay including Communication Blackout. In: *10th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, IEEE Computer Society, 2002, S. 285–294. – ISBN 0-7695-1489-8

[Yokokohji und Yoshikawa 1990] Yokokohji, Yasuyoshi ; Yoshikawa, Tsuneo: Bilateral Control of Master-Slave Manipulators for Ideal Kinesthetic Coupling. In: *IEEE International Workshop on Intelligent Robots and Systems, IROS*, 1990, S. 605–620

[You und Sung 2008] You, Yonghee ; Sung, Mee Y.: Haptic Data Transmission Based on the Prediction and Compression. In: *IEEE International Conference on Communications*, 2008, S. 1824–1828

[Zhu und Salcudean 1995] Zhu, M. ; Salcudean, S. E.: Achieving Transparency for Teleoperator Systems under Position and Rate Control. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems* Bd. 2, August 1995, S. 7–12

Mathematische Notationen A

Hier folgt eine Liste der, dieser Arbeit verwendeten, mathematischen und physikalischen Symbole und Ausdrücke.

Ausdruck	Bedeutung
$\mathbb{N}$, $\mathbb{R}$, $\cdots$	Menge der natürlichen Zahlen, Menge der reelen Zahlen und Menge einzeln aufgeführter Werte.
(a,b), $[a,b]$	Offenes, geschlossenes Intervall der Werte zwischen a und b.
a, b, c, α, β, $\gamma, \ldots$	Skalare Variablen aus $\mathbb{R}$. Kleine griechische Buchstaben und bestimmte, häufig dafür verwendete lateinische Buchstaben.
i, j, k, n, $m, \ldots$	Ganzzahlige Variablen aus $\mathbb{N}$. Bestimmte häufig dafür genutzte kleine Buchstaben des lateinischen Alphabets.
x, y, z	Häufig verwendete Symbole für Raumkoordinaten.
v, a, F	Häufig verwendete Symbole für die physikalischen Größen Geschwindigkeit, Beschleunigung und Kraft.
m, cm, mm, s, ms, N, $^\circ$	Physikalische Einheiten für Längen (Meter, Zentimeter, Millimeter), Zeiten (Sekunden, Millisekunden), Kräfte (Newton) und Winkel (Grad).
$\vec{v}$, $\vec{v}^T$, $\vec{0}$	Spaltenvektor ($\vdots$), Zeilenvektor ($\cdots$), Nullvektor.
$\mathbf{A}$	Matrix ($::$)
$\vec{v}_k$, $\mathbf{A}_{i,j}$	Index. Das k-te Element des Vektors $\vec{v}$, Das Element aus der Matrix $\mathbf{A}$ aus der i-ten Zeile und der j-ten Spalte.
$\mathbf{A}^T$	Transponierte Matrix, an der Hauptdiagonalen gespiegelt.
$\mathbb{R}^n, \mathbb{R}^{n\times m}$	Menge der Vektoren mit n Elementen, Menge der Matrizen mit n Zeilen und m Spalten.

$f(x)$	Reelwertige Funktion von $x \in \mathbb{R}$.
h_t	Folge reeler Zahlen mit $t \in \mathbb{N}$. Häufig als diskrete Form einer reelwertigen Funktion $f(x)$ verwendet.
$\vec{f}(x),\ \vec{h}_t$	Vektorwertige Funktion oder Folge.
$\mathbf{A}(x)$	Matrixwertige Funktion.
$\lvert x\rvert,\ \lVert x\rVert$	Absolutbetrag, Vektornorm.
$x{+}y{+}z$	Spaltensumme $\sum_{i=1}^{n} \vec{v}_i$ eines Vektors. Speziell einer dreidimensionalen Positionsangabe.
$x',\ \dot{x},\ \frac{\mathrm{d}}{\mathrm{d}t}x$	Erste Ableitung von x, oft Ableitung nach der Zeit.
$\frac{\partial}{\partial x}f$	Partielle Ableitung von f nach x.
$\vec{\nabla},\quad \mathbf{J}$	Operatoren für den Gradient (Vektor aller partiellen Ableitungen einer Funktion $f\colon \mathbb{R}^n \to \mathbb{R}$) und die Jacobi-Matrix (Matrix aller partiellen Ableitungen einer Funktion $f\colon \mathbb{R}^n \to \mathbb{R}^m$)

Danksagung

Die Erstellung dieser Diplomarbeit ist mir durch viele Menschen erleichtert oder überhaupt erst ermöglicht worden. Diesen möchte ich an dieser Stelle meinen Dank mitteilen.

Ich bedanke mich zuerst bei den beiden Betreuern dieser Arbeit Prof. Dr. Jianwei Zhang und Dr. Peer Stelldinger für die vertrauensvolle Bereitstellung der Arbeitsmaterialien (Roboter, Computer, Räumlichkeiten). Sie beide haben mir sehr viel Freiheit bei der Verwirklichung meiner Vorstellungen gelassen und haben bei Fragen immer ein offenes Ohr und gute Ratschläge gehabt.

Ich bedanke mich auch bei allen TAMS Mitarbeitern und Studenten für ihre immer währende Hilfsbereitschaft und das Interesse an dem Thema dieser Arbeit. Besonders hervorzuheben sind dabei: Sascha Jockel, Denis Klimentjew, Martin Weser, Hannes Bistry, Bernd Schütz, Dr. Andreas Mäder, Felix Lindner, Gunter Labes und Daniel Westhoff. Und ohne Tatjana Tetsis wäre diese Arbeit wohl kaum möglich gewesen.

Für sehr schnelles und zuverlässiges Korrekturlesen bedanke ich mich sehr bei Waldemar Stegat und Martin Noeske und wiederholt bei Sascha Jockel und Dennis Klimentjew.

Bei meiner Familie und meinen Freunden, insbesondere meiner Freundin Stefanie Stegat, muss ich mich dafür bedanken, dass sie immer zu mir gehalten, sich um mich gekümmert und mich unterstützt haben, auch wenn ich zeitweise nichts davon erwidern konnte.

Eidesstattliche Erklärung

Ich, Jan Bruder, versichere hiermit, dass ich die vorstehende Arbeit selbstständig und ohne fremde Hilfe angefertigt und mich anderer als der im beigefügten Verzeichnis angegebenen Hilfsmittel nicht bedient habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht.

Ich bin mit einer Einstellung in den Bestand der Bibliothek des Fachbereiches einverstanden.

Hamburg, den _____ Unterschrift: _____