Technik
Informatik & Medien
Hochschule Ulm
University of
Applied Sciences

Teleoperation des humanoiden Roboters NAO per Kinect-Kamera

Bachelorarbeit an der
Hochschule Ulm
Fakultät Informatik
Studiengang Technische Informatik

vorgelegt von
Simon Päusch

November 2011

1. Gutachter: Prof. Dr. Christian Schlegel
2. Gutachter: Prof. Dr. Georg Schied

Eigenständigkeitserklärung

Hiermit erkläre ich, daß ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfaßt und keine anderen als die im Literaturverzeichnis angegebenen Hilfsmittel verwendet habe. Insbesondere versichere ich, daß ich alle wörtlichen und sinngemäßen Übernahmen aus anderen Werken als solche kenntlich gemacht und mit genauer Quellenangabe dargelegt habe.

Ulm, den 30. November 2011 Simon Päusch

Zusammenfassung

In der vorliegenden Arbeit wird untersucht, wie eine geeignete Mensch-Maschine-Schnittstelle zur Ansteuerung eines Roboters ausgearbeitet werden kann. Hierbei werden zwei unterschiedliche Methoden kombiniert. Die erste Methode überträgt eine Bewegung 1:1 auf den Roboter. Dabei findet die Kinect Kamera Anwendung, um die Bewegungen des Benutzers zu erfassen. Die zweite Methode dient der Ausführung von Bewegungsabläufen auf dem Roboter, die nach dem jeweils hinterlegten Muster ablaufen. Durch die Kombination der beiden Methoden ist es möglich, die Nachteile zu kompensieren und die Vorteile beider zu nutzen. So können nicht erfassbare Bewegungen über die Bewegungsabläufe, aber auch spontane und intuitive Interaktionen über die Benutzersteuerung realisiert werden. Beide Methoden sind durch ein Umschalten in den jeweiligen Modus über die Sprachsteuerung auswählbar. Zusätzlich sind gewisse Funktionen aber auch Modusübergreifend abrufbar. Das Gesamtszenario einer Einkaufsszene zeigt das Zusammenspiel beider Methoden. Dabei ist zu erkennen, wie diese sich gegenseitig ergänzen können und wann welche Methode sinnvoll angewendet werden kann.

Der vorgestellte Ansatz verdeutlicht durch seine Ergebnisse die Funktionsfähigkeit, sodass dieser zukünftig Anwendung finden kann.

Danksagung

Zunächst einmal möchte ich mich bei Herrn Prof. Dr. rer. nat. Christian Schlegel für dieses äußerst interessante und spannende Thema bedanken, sowie für die erstklassige Organisation und Betreuung der Bachelorarbeit.

Weiter gilt ein riesen Dank meinem Betreuer Herrn M.Sc Siegfried Hochdorfer, der mir jederzeit mit Rat und Tat zur Seite stand. Ebenso waren die Gespräche und Kommentare stets hilf- und lehrreich und gaben immer weitere Denkanstöße, die mich immer weiter brachten. Hierfür ein persönliches: Danke!

Ein nicht zu vernachlässigender Dank geht an alle Mitarbeiter des ZAFH Labors, die durch ihr stetiges Arbeiten von früh morgens bis spät abends und ihre ansteckende Motivation die Zeit wie im nu verfliegen ließ und auf Fragen stets eine passende Antwort fanden.

Ebenfalls möchte ich mich bei meinem persönlichen Umfeld danken, das mich stets moralisch und seelisch motiviert hat.

Auch danke ich allen, die mir in der Zeit Gutes getan und mich unterstützt haben.

Zu guter Letzt bin ich all denjenigen zu Dank verpflichtet, die meine Bachelorthesis Korrektur gelesen haben; Danke dafür.

Inhaltsverzeichnis

1 Einleitung **1**

2 Stand der Technik **3**
2.1 Humanoide Roboter . . . 3
2.2 Skeleton Tracking . . . 4
2.2.1 Softwarelösungen . . . 4
2.3 Teleoperation von humanoiden Robotern . . . 5
2.3.1 Teleoperation mit Hilfe von Skeleton Tracking . . . 5
2.3.2 Teleoperation mit Hilfe von Spracheingabe . . . 5
2.4 Aktuelle Arbeiten zur Teleoperation des NAOs . . . 6

3 Grundlagen **7**
3.1 Kinect Sensor . . . 7
3.2 Humanoider Roboter NAO . . . 8
3.2.1 Freiheitsgrade des NAOs . . . 10
3.2.2 Kommunikation mit NAO . . . 11

4 Methode **13**
4.1 Aufgabenstellung . . . 13
4.2 Potentielle Ansätze und Lösungen . . . 14
4.2.1 Mögliche Frameworks für das Skeleton Tracking . . . 14
4.2.1.1 Auswahlkriterien des Frameworks . . . 14
4.2.1.2 Beschreibung der Frameworks . . . 14
4.2.2 Mögliche Arten der Teleoperation . . . 15
4.2.2.1 Steuerung über direkte (1:1) Übertragung von Bewegungen . . . 15
4.2.2.2 Steuerung über Behavior . . . 16
4.2.3 Möglichkeiten zum Abruf der Behavior . . . 17
4.2.3.1 Tastatur/Controller . . . 17
4.2.3.2 Spezialgeste . . . 17
4.2.3.3 Spracheingabe . . . 17
4.3 Ausgewählter Ansatz „Direct and Behavioral Teleoperation“ . . . 17
4.3.1 Auswahl des Frameworks . . . 18
4.3.2 Auswahl der direkten 1:1 Umsetzung . . . 18
4.3.3 Auswahl der Behavior Umsetzung . . . 18
4.3.4 Steuerung der Behavior durch Spracheingabe . . . 19
4.4 Realisiertes Konzept . . . 22

4.4.1 Erfassung der Gelenkpunkte . . . 22
4.4.1.1 Gelenkwinkelberechnung aus 3D Punktwolke . . . 22
4.4.2 Spracherkennung für die Behaviorsteuerung . . . 25
4.4.2.1 Transformation der Sprachbefehle in Behavior . . . 25
4.4.3 Datenversand an den NAO . . . 25
4.4.4 Teleoperation . . . 25
4.4.5 Softwareübersicht . . . 26

5 Ergebnisse **31**
5.1 Genauigkeit der errechneten Winkel . . . 31
5.1.1 Erkenntnisse . . . 33
5.2 Erkennungsrate der Speech Recognition . . . 33
5.2.1 Fehlerkennungsrate . . . 35
5.3 Teilszenarien . . . 36
5.3.1 Ansteuern von Ziel Positionen . . . 36
5.3.2 Greifen des Einkaufwagens . . . 38
5.3.3 Laufen mit dem Einkaufswagen . . . 38
5.3.4 Greifen und ablegen eines Objekts . . . 41
5.4 Einkaufsszenario . . . 44
5.4.1 Überblick über das Szenario . . . 44
5.4.2 Ablauf des Szenarios . . . 44

6 Zusammenfassung und Ausblick **51**
6.1 Zusammenfassung . . . 51
6.2 Ausblick . . . 51

Kapitel 1

Einleitung

Die Anwendungsgebiete der Robotik sind bereits heute vielfältiger, als vielleicht allgemein angenommen. So werden Roboter bereits auf dem Mars, unter Wasser, in Lagerhallen oder im Haushalt eingesetzt [SN04, Seite 1 ff.]. Bei der Teleoperation, also dem fernsteuern eines Roboters, ist es dem Benutzer möglich, den Roboter aktiv zu steuern und direkt in die Handlungen einzugreifen. In der vorliegenden Arbeit wird untersucht, wie eine natürliche und einfache Schnittstelle zwischen Mensch und Maschine realisiert werden kann (siehe Abbildung 1.1). Der Schwerpunkt liegt darin, zu un-

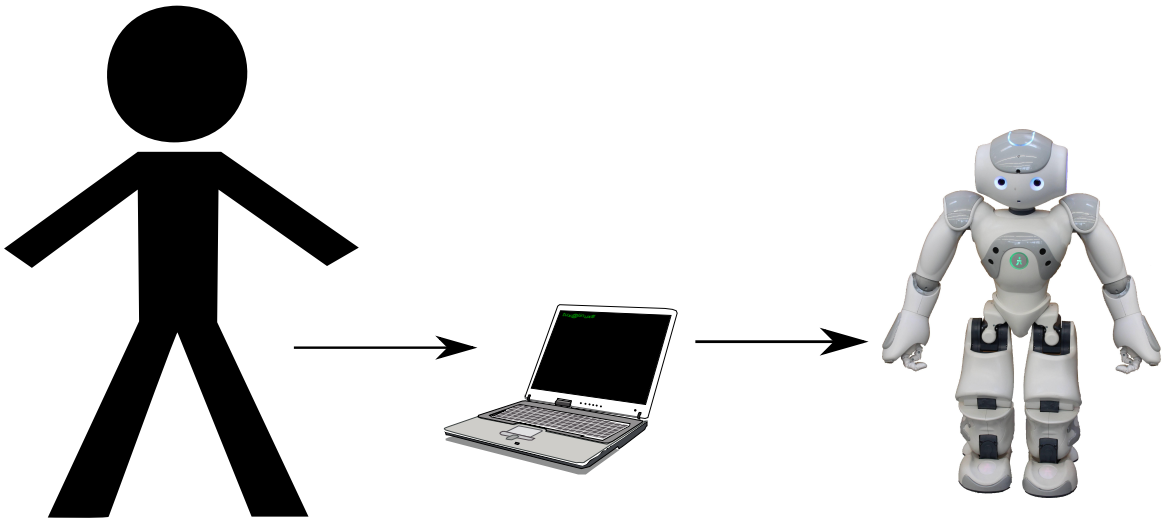

Abbildung 1.1: Grobüberblick über die Aufgabe

tersuchen, wann welche Ausführungsart von Bewegungen eingesetzt wird. So kann zum einen die Bewegung des Menschen erfasst und auf gleiche Art und Weise auf dem Roboter ausgeführt, zum anderen durch Aufrufen eines Bewegungsmusters realisiert werden. Es werden Techniken zur Erkennung der menschlichen Bewegung vorgestellt und untersucht, welche für diese Arbeit am geeignetsten sind. Weiter wird evaluiert, welche Eingabemöglichkeit für die Auswahl eines auszuführenden Bewegungsablaufs in Frage kommt.

Ziel der Arbeit ist, eine intuitive und einfache Steuerung des humanoiden Roboters NAO, der französischen Firma Aldebaran, zu erstellen und zu zeigen, wann welche Steuerungsart am sinnvollsten ist.

Kapitel 2

Stand der Technik

2.1 Humanoide Roboter

Ein humanoider Roboter ist ein, der menschlichen Gestalt nachempfundener, Roboter (siehe Abbildung 2.1). So besteht ein humanoider Roboter meist aus zwei Beinen, einem Torso sowie zwei Armen und einem Kopf. Es gibt auch abgewandelte Arten von humanoiden Robotern, die beispielsweise nur einen Torso mit Kopf und Armen besitzen. Das Ziel der Entwicklung humanoider Roboter ist, dem Menschen einen helfenden Begleiter zur Seite zu stellen, der es ermöglicht, den Menschen unterstützend zu entlasten (vgl. [KFH+08, Seite 1307 ff.]). So ist es vor allem in der alternden Bevölkerung Deutschlands interessant, einen humanoiden Roboter im Bereich der Seniorenbetreuung und -pflege einzusetzen, um die anfallenden Arbeiten zu erleichtern. Es ist denkbar, dass humanoide Roboter beispielsweise Objekte von einem Ort zu einem anderen bringen, die Spülmaschine ausräumen oder das Essen servieren.

Abbildung 2.1: Humanoider Roboter NAO der Firma Aldebaran Robotics

Heutige humanoide Roboter sind bereits unter anderem in der Lage, selbstständig zu laufen oder gar zu rennen (siehe Roboter ASIMO von Honda [Hon]), Treppen hoch zu steigen [OGHB11], Hindernisse zu erkennen und auszuweichen [MBS11], schwere Gegenstände zu heben [SNK08], zu tanzen [NNYI04], Instrumente [DSRI06] und Fussball zu spielen [ZMLA11].

Auf das Fussball spielen hat sich der RoboCup spezialisiert, ein international ausgetragener Wettstreit, gegründet 1997, dessen Ziel es ist, im Jahre 2050 in einem Fussballspiel mit seiner Robotermannschaft das menschliche Weltmeisterteam zu besiegen [Bir10]. In der „Standard Platform League“

treten alle Teams mit denselben Robotern an. Hier kommt seit 2007 der auch in dieser Arbeit verwendete humanoide Roboter NAO zum Einsatz.

2.2 Skeleton Tracking

Das präzise Erkennen und Digitalisieren von menschlichen Bewegungen (Skeleton Tracking) stößt auf vielseitige Anwendungsgebiete. So wird beispielsweise das Aufnehmen von Bewegungen von Menschen dazu genutzt, Animationen von Figuren für 3D Videospiele oder für Filme zu erstellen. Auf diese Weise wurde beispielsweise die Figur „Gollum“ im Film „Herr der Ringe“ erschaffen [Sco03]. Ein anderes Anwendungsgebiet findet sich in der Medizin wieder. So ist die Vision von Vitopsy, dass durch Bewegungen der Hand, ohne Berührungen, auf einem Bildschirm zwischen Bildern gewechselt werden kann. Dies kann beispielsweise im Operationssaal Verwendung finden [Gro].

Ein bekannter, aber aufwendiger Ansatz ist das Anbringen von Reflexionsmarkern an den Gelenken des Menschen, die über eine Kamera erfasst werden können. In seiner Untersuchung verbessert Yueting et al. [ZLP99] die Technik, indem auf Marker verzichtet wird, die Gelenke jedoch am Computer in der ersten aufgenommenen Bildsequenz markiert werden müssen. Von da an rechnet der Computer aus den zweidimensionalen Bildern die Gelenkpunkte, durch Kenntniss des menschlichen Skeletts und dem Loch-Kamera Modell („pin-hole model“), in dreidimensionale Punkte um. Eine ähnliche Lösung bietet die Firma NaturalPoint mit ihrem Softwarepacket „Arena“ an. Hierbei werden mehrere Kameras benötigt, die auf eine Szene gerichtet sind. Durch manuelles Hinzufügen von Reflexionsmarkern am Computer ist auch hier das Erfassen von Bewegungen möglich [Nat].

Da bereits Sensoren existieren, die 3D Informationen einer Szene ermitteln können, gibt es einfachere Möglichkeiten die benötigten Informationen zu erhalten als die mittels der Marker. Der zeitliche Aufwand für das Anbringen der Marker an der Kleidung oder über den Computer ist doch recht hoch. Ebenso ist der Installationsaufwand der Kameras und deren Kalibrierung nicht zu unterschätzen, sodass in dieser Arbeit auf ein anderes Verfahren gesetzt wird.

Eine weitere Möglichkeit, die menschlichen Bewegungen zu erfassen bietet die Firma Xsens mit Hilfe eines Anzugs an. Der Anzug ermöglicht durch die integrierten Sensoren die Wahrnehmung der Bewegungen der im Anzug befindlichen Person. Ein Vorteil hierbei ist, dass keine Kamera benutzt wird und Verdeckungen vor der Kamera daher nicht auftreten. Ebenso kann der Anzug innerhalb, aber auch außerhalb von Gebäuden, verwendet werden [Xse]. Im Gegensatz dazu, sind die Techniken mit den Tiefenkameras, die das natürliche Lichtspektrum nutzen, unter Sonneneinstrahlung nicht störungsfrei einsetzbar.

Der Anzug bietet die Möglichkeit, intuitive Bewegungen zu erfassen. Dennoch besteht ein Aufwand, den Anzug anziehen zu müssen. Ebenso gestaltet sich ein zügiger Wechsel der zu erfassenden Person als zeitaufwendig. Des Weiteren kann das Erfassen bei Personen verschiedener Körpergrößen, die denselben Anzug nutzen, zu ungenauen Ergebnissen führen. Deshalb findet diese Technik in der vorliegenden Arbeit keine Verwendung.

2.2.1 Softwarelösungen

Einen einfacheren und intuitiveren Ansatz bieten verschiedene Softwarepakete, die aus den 3D Informationen der verwendeten Hardware Sensoren, wie beispielsweise 3D Kameras, dreidimensionale Punkte eines Menschen errechnen. So bietet beispielsweise die Firma PMD Technologies mit ihrer 3D Kamera PMD[vision]®CamCube [PMD] und dem Software Development Kit (SDK) Omek Beckon™SDK der Firma Omek Interactive [Int] eine Lösung für das Skeleton Tracking an. Micro-

soft stellt das Skeleton Tracking mit Hilfe ihres SDKs „Kinect for Windows beta SDK“ [Micc] und dem, von Microsoft verkauften, Kinect Kamera Sensor zur Verfügung. Weitere Softwarepakete wie OpenNI [Opec] der Firma Primesense, CLNui [Lab] von Code Laboratories oder Libfreenect [Opea], entwickelt vom OpenKinect Projekt nutzen ebenfalls den Kinect Sensor, um 3D Informationen zu erhalten.

Für die Anforderungen an die vorliegende Arbeit scheinen diese Techniken, die dreidimensionalen Daten des Menschen zu erhalten, geeignet zu sein. So wird in Kapitel 4 evaluiert, welche Softwarelösung am besten geeignet ist.

2.3 Teleoperation von humanoiden Robotern

Das Fernsteuern von Robotern findet beispielsweise in der Medizin für präzise Operationen an schwer erreichbaren Orten, im Militär anhand von Drohnen zur Aufklärung und Überwachung, oder aber in der Fertigungsindustrie wie beispielsweise in Autowerken, ein breites Anwendungsgebiet. Ein Ansatz, einen humanoiden Roboter fern zu steuern, liefert Neo et al. [NYK+05]. Mit Hilfe von zwei Joysticks ist es gelungen, den Roboter zu bedienen, wobei die Ausführungsart der Gelenke über „Inverse Kinematic“ realisiert wird. Inverse Kinematic bedeutet das Bestimmen von Werten für die Gelenke, so dass der Endeffektor die gewünschte Position erreicht. Dies bedeutet am Beispiel des Menschen, dass der Finger, hier der Endeffektor, in eine bestimmte Position gebracht werden soll. Dabei werden für die restlichen Gelenke Werte so errechnet, dass der Finger die gewünschte Position erreicht.

Das Verwenden von Joysticks zur Teleoperation ist für eine intuitive und natürliche Steuerung, die direkte Bewegungen des Menschen auf den Roboter überträgt, nur bedingt geeignet. So können beispielsweise nicht zeitgleich mehrere Gliedmaßen unterschiedlich bewegt werden, sondern sind nur einzeln über ein Auswählen ansteuerbar. In dieser Arbeit soll eine Mensch-Maschine-Schnittstelle ausgearbeitet werden, die eine natürliche und intuitive Steuerung des Roboters ermöglicht. Voraussetzung dafür ist eine gleichzeitige Ansteuerung der verschiedenen Gliedmaßen. Daher scheidet diese Art der Steuerung für die vorliegende Arbeit aus.

2.3.1 Teleoperation mit Hilfe von Skeleton Tracking

In ihrem Ansatz verbindet Pollard et al. [PHRA02] die Erkennung von Bewegungen des Menschen, um einen humanoiden Roboter teleoperativ steuern zu können. Hierbei wird wieder auf die bewährte Technik mit dem Anbringen von Reflexionsmarkern an den Gelenken zurückgegriffen. Die anschließend errechneten Winkel werden an den Roboter gesendet, was eine gleichartige Bewegung erzeugt.

Die Reflexionsmarkertechnik wurde bereits im Verlauf dieses Kapitels diskutiert und für die Zielsetzung dieser Arbeit als nicht geeignet eingestuft.

2.3.2 Teleoperation mit Hilfe von Spracheingabe

Das Erkennen von Sprache gewinnt vor allem im Alltag immer mehr an Bedeutung. So sind heutige Smartphones, Navigationsgeräte oder Computer bereits in der Lage, Befehle anhand einer Spracheingabe auszuführen. Lu et al. [LLCH10] benutzt diese Technik, um den humanoiden Roboter BHR-02 zu steuern. Unter den von ihm angegebenen Vorteilen befindet sich der Nichtgebrauch von Händen und Augen, sowie das einfache Erlernen der Steuerung. Die Spracheingabe wird dazu verwendet, vorprogrammierte Bewegungen, wie beispielsweise „rechten Arm anheben“ oder „Kopf nach links drehen“, auf dem Roboter auszuführen.

Den Roboter per Sprachkommandos zu steuern ist sehr einfach und komfortabel. Die Sprache ist eine natürliche Kommunikationsform des Menschen und wird im Kapitel 4 als mögliche Steuerungsart diskutiert.

2.4 Aktuelle Arbeiten zur Teleoperation des NAOs

Derzeitig befassen sich mehrere andere Arbeiten mit dem Thema der Teleoperation des humanoiden Roboters NAO, welche bisher noch nicht publiziert wurden, sondern nur in Form von Internet Videos verfügbar sind. Die folgenden Beschreibungen erläutern die gezeigten Techniken. Dabei werden mögliche Ungenauigkeiten nicht berücksichtigt, da die diesbezüglichen Informationen lediglich aus den entsprechenden Videos stammen.

Koenemann und Bennewitz von der Universität Freiburg im Breisgau zeigen die Steuerung eines NAOs mit Hilfe eines Anzugs der erwähnten Firma Xsens. Hierbei wird die Bewegung des Benutzers über den Anzug erfasst und auf den Roboter übertragen. Die errechneten Winkel werden vom NAO über inverse Kinematic umgesetzt. Hierbei wird inverse Kinematic so angewandt, dass bei der Ausführung der Bewegung eine stabile Position sichergestellt wird. Eine stabile Position bedeutet beispielsweise das aufrechte Stehen, ohne umzufallen [Youb].

Suay und Chernova suchen in ihrer Arbeit nach einer Möglichkeit, humanoide Roboter auf die „bestmögliche natürliche Art“ (laut Beschreibungstext des Videos) zu steuern. Hierfür wird die Kinect Kamera verwendet, um sowohl Bewegungen direkt auf dem Roboter, als auch Bewegungsmuster auszuführen. Die Bewegungsmuster werden zum einen durch spezielle Gesten, zum anderen über Bewegungen, wie beispielsweise dem vorwärts Laufen, aufgerufen [Youa].

In einem weiteren Video wird gezeigt, wie mit Hilfe von zwei Wii-Controllern die Teleoperation des NAOs über das Ausführen von Bewegunen ermöglicht wird. Die Armbewegungen werden von der Kinect Kamera erfasst und auf dem Roboter ausgeführt. Nicht erkannte Bewegungen wie die Dreh-Bewegung der Unterarme werden mit Hilfe der Wii-Controller, der die Dreh-Bewegung über einen Lagesensor bestimmen kann, erfasst [Youc].

Die aufgeführten Arbeiten befassen sich mit einer ähnlichen Thematik, nämlich der direkten Ausführung von Bewegungen auf dem Roboter, teils auch mit dem Aufrufen von hinterlegten Bewegungsmustern. Allerdings sind die verwendeten Techniken über die Videos nicht vollständig ersichtlich. Darüber hinaus werden in der vorliegenden Arbeit die Bewegungsmuster weder über Spezialgesten oder Spezialbewegungen, noch über zusätzliche Hardware-Controller aufgerufen. Eine Ergänzung dieser Arbeit ist die Untersuchung, wann eine vom Menschen direkt übertragene, oder über Bewegungsmuster vollzogene Bewegungsausführung Sinn macht.

Kapitel 3

Grundlagen

3.1 Kinect Sensor

Der im November 2010 auf dem US-Markt erschienene und unter dem Namen „Natal“ entwickelte Kinect Sensor (siehe Abbildung 3.1) ist ursprünglich für den Gebrauch mit der XBOX-360 Spielekonsole von Microsoft gedacht. Doch schon einige Zeit nach der Veröffentlichung waren Treiber, wie beispielsweise CLNUI [Lab], im Internet zu finden, die es ermöglichen, die von den Komponenten des Kinect Sensors gelieferten Daten wie beispielsweise das Tiefenbild auszulesen. Die folgenden Beschreibungen zur Kinect Kamera sind den Quellen [Micd], [Mice], [SKM+, Seite 7-10], sowie [ZSMG07] entnommen.

Abbildung 3.1: Komponenten des Kinect Sensors (nach [Mice]).

Der Tiefensensor (1): Der Tiefensensor besteht wie in Abbildung 3.1 zu sehen aus zwei Komponenten: zum einen aus dem Infrarot (IR) Projektor auf der linken Seite, zum anderen aus einer IR Kamera auf der rechten Seite. Der IR Projektor sendet ein infrarot Muster in die Umgebung aus. Die IR Kamera erfasst dieses. Da sich die IR Kamera nicht parallel, sondern schräg zum IR Projektor befindet, erfasst diese das infrarot Muster aus einem anderen Winkel als es der IR Projektor gesendet hat. Dadurch ist es möglich, Tiefeninformationen über das von der IR Kamera erfasste Muster zu errechnen. Der Tiefensensor liefert bis zu 30 Bilder pro Sekunde (fps), mit einer maximalen Auflösung von 640x480 Pixeln.

Die RGB-Kamera (2): Die integrierte RGB-Kamera ermöglicht das Auslesen von Farbbildern mit einer Bildrate von 30fps und einer maximalen Auflösung von 1280x1024 Pixeln.

Ein Mikrofonarray (3): Eine Reihe von Mikrofonen (Mikrofonarray) unten an der Vorderkante des Kinect Sensors ermöglichen Spracheingaben. Der Vorteil der vier Mikrophone gegenüber einem einzelnen sind:
- bessere Rausch- und Echounterdrückung.
- Lokalisierung der akustischen Quelle.

Motorisierter Sockel (4): Der sich im Sockel der Kinect Kamera befindlichliche mechanische Antrieb neigt bei Bedarf die Sensorleiste automatisch nach oben und nach unten. Dies ist für die Zielanwendung Spiele wichtig, da so der von der Kinect Kamera erfassbare Sichtbereich je nach Situation automatisch angepasst werden kann.

Durch den niedrigen Einführungspreis von 149 Euro und die Vielzahl an auslesbaren Daten des Sensors wurde Microsofts Kinect zum am schnellsten verkauften IT-Gerät weltweit, das sogar einen Eintrag in das Guinness-Buch der Rekorde erhielt [Tan]. Als bekannteste Treiber beziehungsweise Frameworks gelten wie bereits erwähnt Libfreenect, CLNui, das Omek SDK, OpenNI und das Microsoft Kinect beta SDK. Ein Beispiel für die von der Kinect gelieferten Bilddaten sind in Abbildung 3.2 ersichtlich. Oben ist das Tiefenbild zu sehen, über das beispielsweise das Microsoft Kinect beta SDK die Punktdaten des Skeletts berechnet. Unten im Bild ist das RGB Bild zu sehen.

3.2 Humanoider Roboter NAO

Mit der Größe des NAOs, bezogen auf die vorliegende Version H21, von 58cm und einem Gewicht von 5KG ist es Aldebaran gelungen, einen kompakten kleinen humanoiden Roboter zu entwickeln. Dieser verfügt über zwei eingebaute Kameras über die beispielsweise eine Gesichtserkennung möglich ist, zwei Lautsprecher, sowie 4 Mikrophone, die für die Spracherkennung genutzt werden können. Des Weiteren besitzt er mehrere LEDs, Ultraschall-Sensoren um Entfernungen zu messen, sowie Drucksensoren (Bumpers) an den Füßen, die bei Berührungen von Gegenständen warnen können. Die Kommunikation mit dem Roboter kann sowohl per WLAN, als auch Ethernet vollzogen werden. Der Roboter verfügt über mehrere Gelenke durch die versucht wird, dem Roboter die größtmöglichen Freiheiten in seinen Bewegungen zu bieten [Roba].

Der NAO verfügt über eine API die mit C++, Phyton, Urbiscript aber auch .Net über die auf dem NAO integrierte Middleware NAOqi angesprochen werden kann [Robd]. Der große Hardware Umfang schlägt sich auch bei der Software nieder. So wird die Software „Choregraphe“ mitgeliefert, die unter anderem das Erstellen vorprogrammierter Bewegungsmuster, sogenannte Behavior, ermöglicht. Zusätzlich wird auch die Simulation eines 3D NAO Roboters angeboten [PMGM09]. Ein einfaches Behavior besteht aus Zeitstempeln, zu denen die Werte der Gelenkwinkel zugeordnet sind. Die Zeitstempel sind auf einer Zeitachse aufgereiht. Beim Aufruf eines Bewegungsablaufs werden entlang der Zeitachse die Zeitstempel aufgerufen und die Winkel entsprechend den hinterlegten Werten gesetzt. Im Choregraph ist es möglich die Werte der Gelenkwinkel manuell einzutragen. Zusätzlich können die Werte aber auch aus der Pose des NAOs übernommen werden. So ist es möglich, den simulierten oder echten NAO in die gewünschte Pose zu versetzen und alle aktuellen Werte der Gelenkwinkel im gewünschten Zeitstempel zu speichern. Darüber hinaus können zusätzliche Daten ausgewertet werden. Ein Beispiel ist der Gyroskop, über den ausgelesen werden kann in welcher Lage sich der NAO befindet. Fällt dieser beispielsweise um, kann dies erkannt werden. Dadurch ist es möglich, angemessen

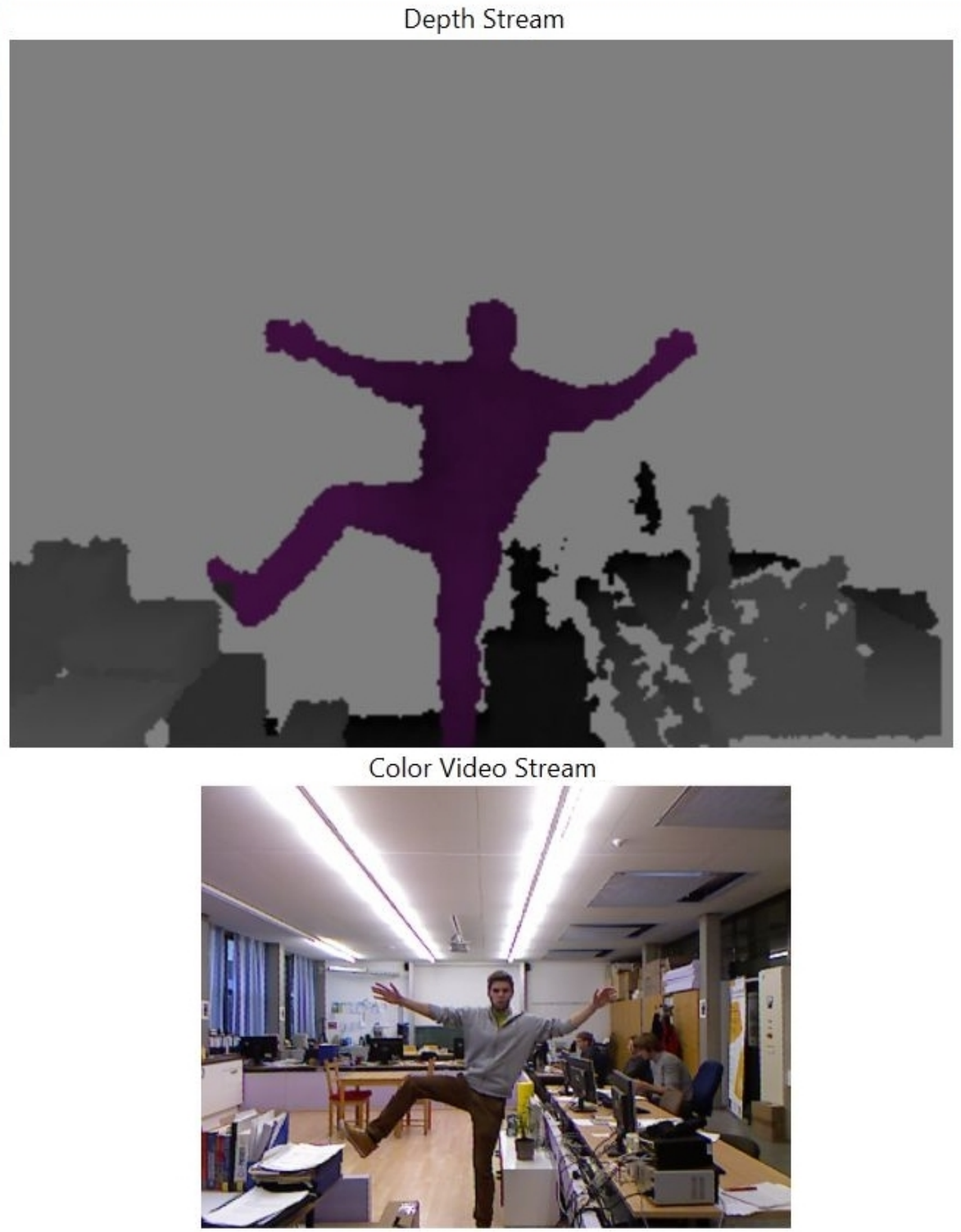

Abbildung 3.2: Überblick über die von der Kinect gelieferten Bilddaten

zu reagieren und eine Beschädigung des NAOs zu verhindern. Die verwendeten Behavior mit Beinbewegungen überwachen zusätzlich die Drucksensoren (Bumpers) an den Füßen. Auf diese Weise kann eine Kollision der Füße mit Gegenständen erkannt werden. Berühren die Füße einen Gegenstand, wird der Bewegungsablauf gestoppt was ein Sicherheitsgewinn des ganzen Systems darstellt. So ist es dem NAO nun möglich, eigenständig auf Kollision mit den Füßen zu reagieren und angemessen zu handeln.

3.2.1 Freiheitsgrade des NAOs

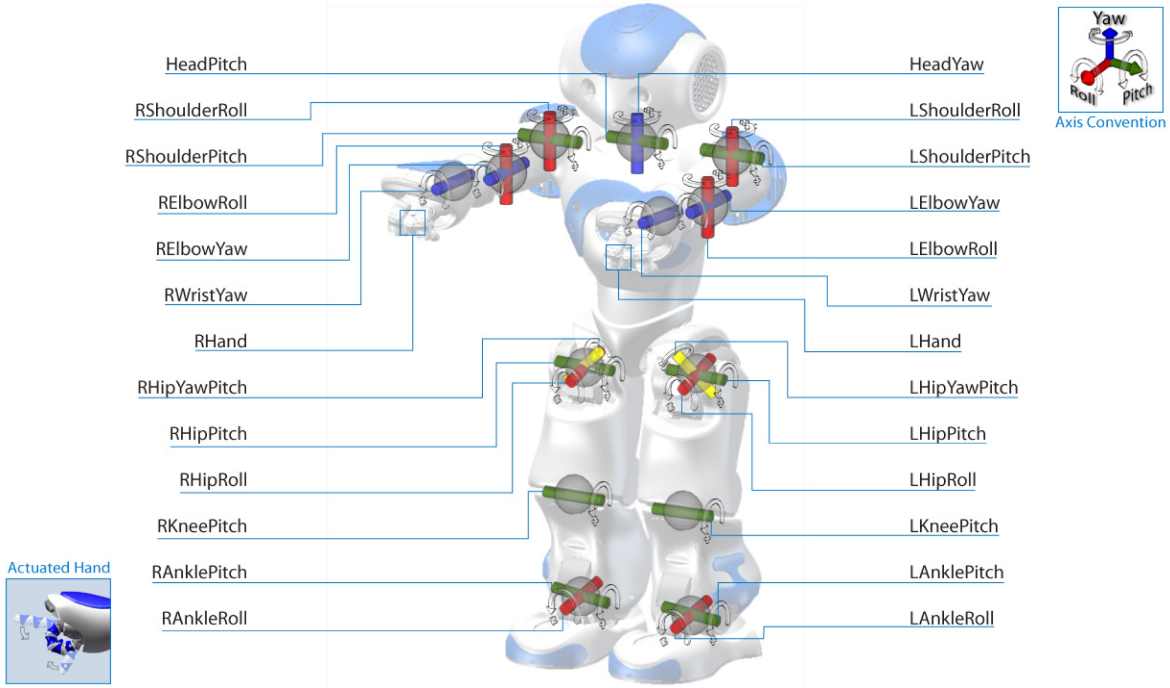

Abbildung 3.3: Überblick über die Gelenke des NAOs ([Robc, Seite 3])

Auch wenn ein humanoider Roboter in der Gestalt einem Menschen sehr ähnlich ist besitzt er meistens andere Proportionen, Gelenke und somit auch andere Bewegungsmöglichkeiten. Abbildung 3.3 zeigt die vorhandenen Gelenke des NAOs, über die Bewegungen vollzogen werden können. Die folgende Tabelle 3.1 listet die möglichen Freiheitsgrade (Wertebereiche) auf, die die aufgezeigten Gelenke annehmen können. Es zeigt sich, dass manche Bewegungen vom Roboter ausführbar sind, die dem Menschen nicht möglich sind. Beispielsweise gelingt es nicht den Unterarm um mehr als ca. 180° zu drehen, ohne dabei den Oberarm mit zu bewegen. Der NAO hat hier jedoch einen Freiheitsgrad von 209° (L/R WristYaw). Aber auch wir Menschen haben teilweise größere Bewegungsfreiheiten als der NAO. So sind bei diesem die Hüftgelenke (R/L HipYawPitch) nicht unabhängig ansteuerbar, da es sich um ein und denselben Motor handelt [Robb]. Hieraus ergeben sich Einschränkungen in den Bewegungen des Menschen oder des Roboters, die berücksichtigt werden müssen. Dadurch können eventuell nicht alle Bewegungen gleich ausgeführt werden, sodass eine direkte Übertragung der Bewegung des Menschen auf den Roboter nicht immer Sinn macht (siehe Kapitel 4.2.2).

Joint name	Motion	Range (degrees)	Range (radian)
HeadYaw	Head joint twist (Z)	-119.5 to 119.5	-2.0857 to 2.0857
HeadPitch	Head joint front and back (Y)	-38.5 to 29.5	-0.6720 to 0.5149
RShoulderPitch	Right shoulder joint front and back (Y)	-119.5 to 119.5	-2.0857 to 2.0857
RShoulderRoll	Right shoulder joint right and left (Z)	-94.5 to -0.5	-1.6494 to -0.0087
RElbowYaw	Right shoulder joint twist (X)	-119.5 to 119.5	-2.0857 to 2.0857
RElbowRoll	Right elbow joint (Z)	0.5 to 89.5	0.0087 to 1.5621
RWristYaw	Right wrist joint (X)	-104.5 to 104.5	-1.8238 to 1.8238
RHand	Right hand	Open And Close	Open And Close
LShoulderPitch	Left shoulder joint front and back (Y)	-119.5 to 119.5	-2.0857 to 2.0857
LShoulderRoll	Left shoulder joint right and left (Z)	0.5 to 94.5	0.0087 to 1.6494
LElbowYaw	Left shoulder joint twist (X)	-119.5 to 119.5	-2.0857 to 2.0857
LElbowRoll	Left elbow joint (Z)	-89.5 to -0.5	-1.5621 to -0.0087
LWristYaw	Left wrist joint (X)	-104.5 to 104.5	-1.8238 to 1.8238
LHand	Left hand	Open And Close	Open And Close
LHipYawPitch	Left hip joint twist (Y-Z 45°)	-65.62 to 42.44	-1.145303 to 0.740810
RHipYawPitch	Right hip joint twist (Y-Z 45°)	-65.62 to 42.44	-1.145303 to 0.740810
LHipRoll	Left hip joint right and left (X)	-21.74 to 45.29	-0.379472 to 0.790477
LHipPitch	Left hip joint front and back (Y)	-101.63 to 27.73	-1.773912 to 0.484090
LKneePitch	Left knee joint (Y)	-5.29 to 121.04	-0.092346 to 2.112528
LAnklePitch	Left ankle joint front and back (Y)	-68.15 to 52.86	-1.189516 to 0.922747
LAnkleRoll	Left ankle joint right and left (X)	-44.06 to 22.79	-0.769001 to 0.397880
RHipRoll	Right hip joint right and left (X)	-42.30 to 23.76	-0.738321 to 0.414754
RHipPitch	Right hip joint front and back (Y)	-101.54 to 27.82	-1.772308 to 0.485624
RKneePitch	Right knee joint (Y)	-5.90 to 121.47	-0.103083 to 2.120198
RAnklePitch	Right ankle joint front and back (Y)	-67.97 to 53.40	-1.186448 to 0.932056
RAnkleRoll	Right ankle joint right and left (X)	-22.27 to 45.03	-0.388676 to 0.785875

Tabelle 3.1: Übersicht über Winkel des NAOs (aus [Robb])

3.2.2 Kommunikation mit NAO

Die Kommunikation mit dem NAO ist auf komfortable Art möglich. So kann als Übertragungstechnik sowohl Wireless LAN als auch ein Ethernet Kabel zum Einsatz kommen. Des Weiteren bietet der NAO mit seiner API reichlich Möglichkeiten zur Ansteuerung an. Zum Aufrufen von API-Funktionen werden Proxies eingesetzt, die es auch ermöglichen, über die bereits erwähnten verschiedenen Programmiersprachen zuzugreifen. Zum initialisieren eines Proxies unter C# müssen lediglich die Roboter IP Adresse und Port übergeben werden, wobei es möglich ist, sowohl den simulierten als auch den echten NAO anzusteuern. Der simulierte NAO hat jedoch Einschränkungen gegenüber dem echten NAO, so ist es beispielsweise nicht möglich, eine Sprachausgabe über den TextToSpeech-Proxy auszugeben.

Kapitel 4

Methode

4.1 Aufgabenstellung

Die Aufgabe dieser Arbeit ist eine Möglichkeit zu finden, den humanoiden Roboter NAO intuitiv zu steuern, was verschiedene Fragen aufwirft:

1. Wie kann eine vom Benutzer ausgeführte Bewegung intuitiv und sinnvoll auf dem Roboter ausgeführt werden?
2. Wie können Bewegungen, die nicht direkt ausgeführt werden können, anders realisiert werden?
3. Wie können diese anderen Bewegungen aufgerufen werden?

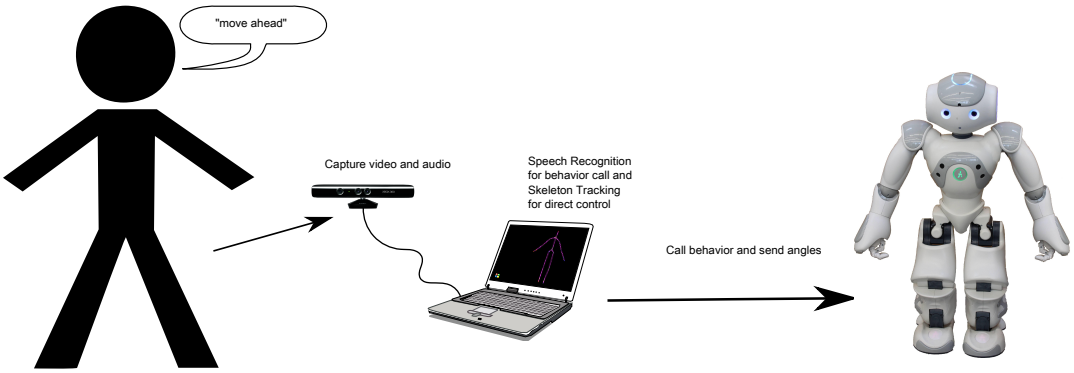

Abbildung 4.1: Überblick über die Gesamtaufgabe

Die in Abbildung 4.1 gezeigten Teilbereiche müssen jeweils getrennt betrachtet und bewertet werden. Die folgende Tabelle 4.1 identifiziert die einzelnen Teilbereiche.

Erkennen und Erfassen der menschlichen Bewegungen
Art der Steuerung (direkt oder über Behavior)
Technik zum Abrufen der Behavior

Tabelle 4.1: Identifizieren der zu untersuchenden Teilbereiche

4.2 Potentielle Ansätze und Lösungen

Wie bereits im Kapitel 2 erklärt, existieren einige Ansätze, um die dreidimensionalen Informationen einer, von einem Menschen ausgeführten Bewegung zu erhalten. Die mit einer Tiefenkamera verwirklichten Lösungen bieten die benötigten Randbedingungen wie beispielsweise ein geringer Installations- und Kalibrierungsaufwand, sowie ein günstiger Anschaffungspreis im Falle der Kinect Kamera.

4.2.1 Mögliche Frameworks für das Skeleton Tracking

In diesem Abschnitt werden die Frameworks vorgestellt und beschrieben, sowie die Kriterien die zur Auswahl eines der Frameworks führen.

4.2.1.1 Auswahlkriterien des Frameworks

Betriebssystem: Die Entwicklung mit dem Framework unter einem beliebigen Betriebssystem ist von Vorteil. So ist das Entwickeln der Software an kein spezielles Betriebssystem und dessen Lizenzen gebunden. Dieses Kriterium ist nicht notwendig, stellt aber einen kleinen Vorteil dar.

Dokumentation, Samples und Community: Ein sehr wichtiger Punkt für die Auswahl des Frameworks stellen die Anforderungen an eine verständliche und möglichst vollständige Dokumentation dar. Zusätzlich ist es von Vorteil, wenn eine große Community und Samples (kleine Beispielprogramme) vorhanden sind die zum schnellen Verständnis des Frameworks beitragen und somit die Einarbeitungszeit drastisch reduzieren.

Spracherkennung (Speech Recognition): Eine bereits im Framework integrierte Möglichkeit, gesprochene Sprache zu erkennen kann dafür verwendet werden, erweiterte Befehle aufzunehmen. Die erweiterten Befehle können dazu verwendet werden, Bewegungen, die nicht direkt vom NAO ausführbar sind, über zuvor erstellte Bewegungsabläufe, sogenannte Behavior, zu ersetzen. Beispielsweise kann so das Laufen realisiert werden, was die Einschränkung von nicht direkt ausführbaren Bewegungen kompensiert.

Skeleton Tracking: Der wichtigste Punkt zur Auswahl des Frameworks stellt das Skeleton Tracking dar. So ist es unabdingbar, dass sich dieses in einem soliden Stadium befindet, möglichst fehlerfrei, zuverlässig und genau arbeitet. Durch das Skeleton Tracking ist es möglich, Bewegungen auf eine intuitive Weise zu erfassen und die gelieferten Daten sinnvoll zu verwenden.

Datenverarbeitung: Ein nicht unwesentlicher Punkt ist die Geschwindigkeit, in der die von der Kinect Kamera erfassten Daten verarbeitet werden. Sowohl die Erfassung der Bilddaten in Bildern pro Sekunde als auch deren Genauigkeit sind von entscheidender Bedeutung.

4.2.1.2 Beschreibung der Frameworks

CLNUI ist eine Ansammlung von Treibern, welche nur unter Windows lauffähig sind und von Code Laboratories entwickelt wurden. Mithilfe der Treiber ist es möglich, Zugriff auf die Bildsensoren der Kinect Kamera sowie auf die Mikrophone und den Motor zu erhalten. Die Treiber sind mit wenigen Beispielprogrammen und wenig Dokumentation versehen. Darüber hinaus existiert keine Implementierung des Skeleton Trackings [Lab].

Libfreenect ist ein Framework des OpenKinect Projekts, welches sowohl unter Windows als auch Linux / Unix lauffähig ist. Auch hier befindet sich keine Implementierung des Skeleton Trackings, weshalb auch dieses Framework nicht näher betrachtet wird [Opea].

OpenNI, von der Firma Primesense, die auch bei der Entwicklung der Kinect Kamera beteiligt war, entwickelt, ist unter Windows, Linux und Mac OS X einsetzbar. Mit der, ebenfalls von Primesense entwickelten Middleware NITE [Opeb], wird ein umfangreiches Softwarepaket geboten. Neben dem bereits integrierten Skeleton Tracking bietet das Framework eine Vielzahl von Zusatzfeatures wie Gesture Recognition (wie beispielsweise dem „hand tracking“). Das Framework nutzt nicht die Möglichkeit, Zugriff auf die Audio Daten der Kinect Kamera zu erhalten. Hier muss im Falle einer Spracheingabe auf zusätzliche Komponenten zurückgegriffen werden [Opec].

Kinect for Windows beta SDK ist ein von Microsoft angebotenes Framework, das ebenfalls ein integriertes Skeleton Tracking anbietet. Um das Skeleton Tracking zu aktivieren wird keine Initialisierungspose, wie es bei OpenNI der Fall ist, benötigt. Zusätzlich bietet Microsoft noch eine Spracherkennung an. Es verzichtet allerdings sowohl auf die Lauffähigkeit unter Linux, als auch auf integrierte Gesture Recognition. Da sich die Version noch im Beta Stadium befindet, ist es noch nicht möglich, kommerzielle Produkte herzustellen ohne gegen die Lizenzbestimmungen des SDKs zu verstoßen [Micb]. Erst mit dem Veröffentlichen der finalen Version 2012 soll es auch möglich sein, kommerzielle Software zu entwickeln [Mica]. Das Framework bietet den größtmöglichen Zugriff auf alle Kinect Sensoren (Motor, LEDs, Audio, Video) und befindet sich trotz seiner Beta Version in einem sehr stabilen Stadium [Micc].

Das Omek SDK der Firma Omek bietet eine Vielzahl von Möglichkeiten. So ist es möglich, verschiedenste 3D Kameras beziehungsweise Sensoren System anzusteuern ohne an einen Hersteller oder ein spezielles Produkt gebunden zu sein. Auch ist es möglich, Körperbewegungen als Gesten zu interpretieren und diese in den Programmen abzurufen. Des Weiteren besitzt es ebenfalls ein integriertes Skeleton Tracking das keine Initialisierungspose benötigt [Int].

4.2.2 Mögliche Arten der Teleoperation

In diesem Kapitel wird erläutert, welche Vor- und Nachteile sowohl eine direkte Übertragung der Bewegungen an den Roboter, als auch das Ausführen von vordefinierten Bewegungsabläufen hat.

4.2.2.1 Steuerung über direkte (1:1) Übertragung von Bewegungen

Die direkte 1:1 Umsetzung von Bewegungen bietet einige Vorteile gegenüber dem Aufrufen von Bewegungsmustern. So liegt die Kontrolle der Bewegungen beim Benutzer selbst, der aktiv und auch spontan auf Ereignisse reagieren kann. Ein Hauptvorteil ist, dass keine Behavior vorprogrammiert werden müssen wie beispielsweise Arme heben, anwinkeln oder ähnliches. Darüber hinaus ist die 1:1 Umsetzung der vorgemachten Bewegung auf dem Roboter intuitiv und eine Steuerung des Roboters leicht zu erlernen. Die präzise Ausführung der Bewegung des Roboters hängt vor allem von der Erkennung der Bewegung des Benutzers ab. Ist die Erkennung ungenau oder sprunghaft können ruckartig ausgeführte Bewegungen auftreten. Dies wäre beim Abruf eines vorprogrammierten und auf dem Roboter installierten Behaviors nicht gegeben. Hier müssen keine Bewegungen erkannt werden, sondern lediglich das im Behavior hinterlegte Muster ausgeführt werden. Die 1:1 Umsetzung ist somit

stets auf die Erkennung der vorgemachten Bewegung angewiesen, was bedeutet, dass nicht erkannte Bewegungen nicht umgesetzt werden.

Die folgende Abbildung 4.2 zeigt die direkte Steuerung an einem simulierten NAO. Im linken Teil der Bilderserie ist das vom Framework erfasste Skelett des Menschen, rechts davon der simulierte NAO zu erkennen. Die Bilderserie zeigt eine einfache Bewegung der Arme.

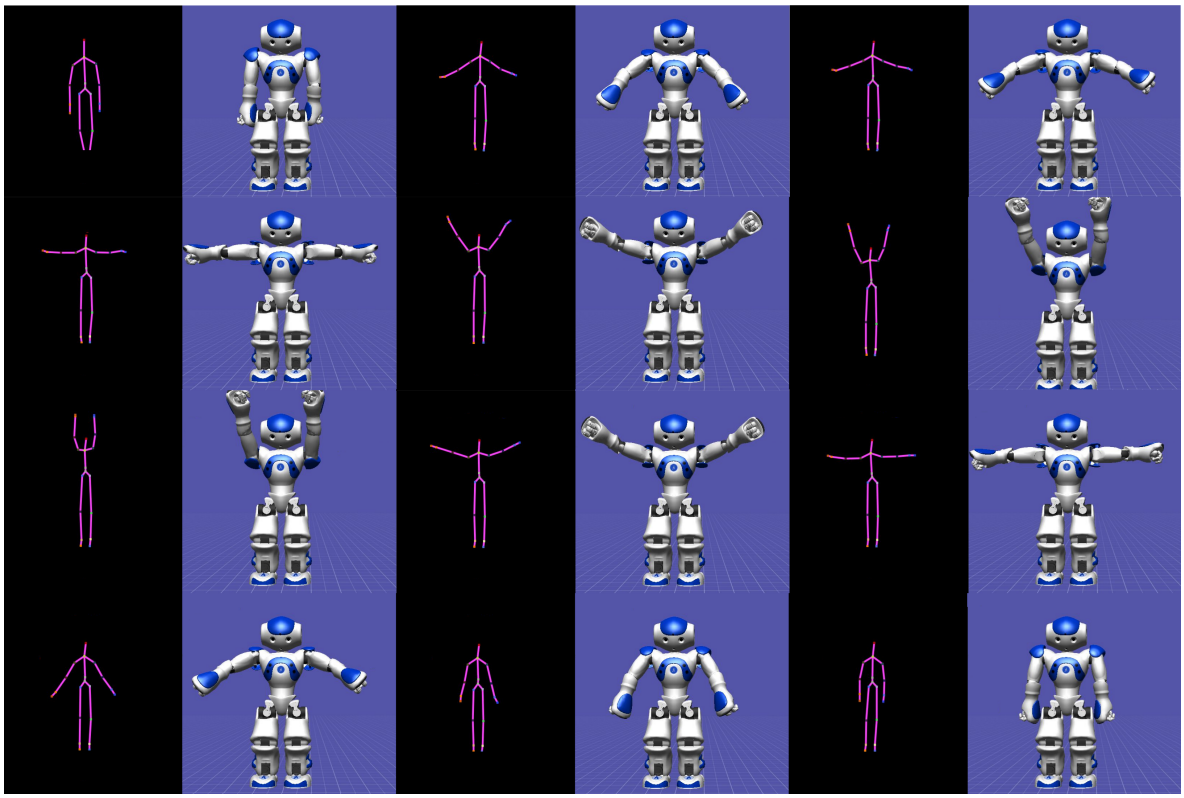

Abbildung 4.2: Bilderserie einer 1:1 Übertragung.

4.2.2.2 Steuerung über Behavior

Die Umsetzung einer vorprogrammierten Behaviorsteuerung beruht auf dem Gedanken, weniger intuitiv und spontan gesteuerte Bewegungen über diese starre und vorgegebene Technik auszuführen. So liegt ein wichtiges Kriterium für die Beine eines Roboters beispielsweise darin, ihn in einer sicheren Position zu halten und unsichere, dem Roboter schädliche Bewegungen zu vermeiden oder von vorne herein auszuschließen. Daher ist es unsinnig, eine Bewegung, wie beispielsweise das auf einem Bein stehen, direkt auf den Roboter zu übertragen. Da dieser unterschiedliche Proportionen und Massen hat, besitzt dieser auch zum Menschen unterschiedliche stabile Lagen. Bei einer direkten Ausführung der Bewegung ist es wahrscheinlich, dass der Roboter instabil wird und umfällt. Ebenso kann die direkten Umsetzung einer Beinbewegung aufgrund von Verzögerungen bei der Übertragung oder der Berechnung der Daten zu einer instabiles Position führen. Dies kann zur Folge haben, dass der Roboter umfällt und diesem dadurch Schaden zugefügt wird.

4.2.3 Möglichkeiten zum Abruf der Behavior

In diesem Abschnitt werden Vor- und Nachteile verschiedener Abrufmöglichkeiten für vorprogrammierte Behavior diskutiert.

4.2.3.1 Tastatur/Controller

Das Abrufen der einzelnen Behavior per Tastatur, Joystick oder mit Hilfe anderer technischer Geräte, würde der intuitiven und natürlichen Steuerung entgegenstehen. So müssen beim Auswählen der Behavior die Finger benutzt und die Augen auf die zu drückenden Tasten oder Knöpfe gerichtet werden. Zusätzlich besteht der Zwang sich in Kabelreichweite zu befinden, sofern keine kabellose Bedienung möglich ist.

4.2.3.2 Spezialgeste

Über Spezialgesten beziehungsweise Spezialposen ein Behavior aufzurufen birgt den großen Nachteil, dass die möglichen freien Bewegungen dadurch eingeschränkt werden. So ist es nicht ohne weiteres möglich, alle denkbaren Bewegungen auszuführen, weil dabei eventuell eine Spezialgeste erkannt wird. Eine Abhilfe kann hierbei ein Modus Wechsel schaffen, der sich zwischen „Auswahl eines Behaviors über eine Spezialgeste“ und „Direkte Ausführung der vorgemachten Bewegung“ hin- und herwechseln lässt. Diese Art des Aufrufs eines Behaviors ermöglicht es nicht, alle Bewegungen uneingeschränkt auszuführen. So kann durch eine Bewegung versehentlich eine Spezialpose erfasst werden, was zum Aufruf eines Behaviors führen kann.

4.2.3.3 Spracheingabe

Die Sprache als Aufruf eines auszuführenden Behaviors zu realisieren, kann einer intuitiven und einfachen Steuerung gerecht werden. So ist es für den Menschen eine natürliche Form, per Sprache zu kommunizieren. Wird ein Befehl nicht exakt erkannt, kann dieser dennoch Verwendung finden. So ist es denkbar, dass der Roboter den Befehl ausspricht und die Frage stellt, ob der ausgesprochene Befehl korrekt erkannt oder interpretiert wurde. Durch die Bestätigung des Benutzers kann der Befehl als richtig erachtet und ausgeführt werden. Durch die Rückfrage des Roboters kann so die Kommunikation zu einem Dialog erweitert werden. Bei der Benutzung von Sprache zur Steuerung sind keine Zusatzeingabegeräte wie beispielsweise eine Tastatur oder eine Fernbedienung nötig. Ebenso wird die räumliche Gebundenheit aufgelöst, da nur die Hörreichweite eingehalten werden muss.

4.3 Ausgewählter Ansatz „Direct and Behavioral Teleoperation“

Direct and Behavioral Teleoperation (DBT) bedeutet, den Roboter auf zwei unterschiedliche Arten teleoperativ zu steuern. Die erste Möglichkeit ist sinnvoll, um eine Bewegung direkt 1:1 auf den Roboter zu übertragen. Die zweite Möglichkeit dient der Ausführung von Behavior auf dem Roboter. Der ausgewählte Ansatz versucht bestmöglich, die beiden Arten der Steuerung zu kombinieren, um die Vorteile beider Methoden nutzen zu können und die jeweiligen Nachteile zu kompensieren.

4.3.1 Auswahl des Frameworks

	OpenNi	Libfreenect	MS SDK	CLNUI	Omek SDK
Betriebssystem (Linux/Mac OS X)	++	++	-	-	-
Betriebssystem (Windows)	+	+	+ (nur Windows 7)	+	+
Docs, Samples	+	+	++	+	nicht gefunden
Speech Recognition	-	-	+	-	-
Skeleton Tracking	+	-	++	-	+
Datenverarbeitung:	+	+	+	+	nicht getestet
Ergebnis:	6	5	7	3	2
Legende: ++ umfassend vorhanden, + vorhanden, - nicht vorhanden					

Tabelle 4.2: Features der Frameworks

Die gezeigte Tabelle 4.2 veranschaulicht die verglichenen Frameworks sowie deren Bewertung. Der Vergleich zeigt, dass nur OpenNI, das Omek SDK oder Kinect for Windows beta SDK in Frage kommen, da die anderen beiden Frameworks kein Skeleton Tracking anbieten.

OpenNi kann sich lediglich bei seiner Lauffähigkeit sowohl unter Windows, Mac OS X, als auch Linux, von den anderen beiden Frameworks absetzen.

Das Omek SDK ist nicht installiert und getestet worden, so dass lediglich die von der Firma bereitgestellten Informationen auf deren Webseite [Ome], als Anhaltspunkt dienen. Momentan ist für das Omek SDK kein Forum zu finden, weshalb vermutlich keine Community existiert und die Entwicklung unter diesem SDK (noch) exklusiv erscheint. Ebenso sind auf der Webseite keine Beispielprogramme oder eine Dokumentation zu finden, weshalb dieses SDK für die Entwicklung der Software ausscheidet.

Microsoft setzt sich klar mit einer großen Dokumentation und dem präzisen Skeleton Tracking von den beiden anderen Frameworks ab. Zusätzlich ist eine große Community vorhanden, sowie viele anschauliche Samples. Ein weiterer Vorteil ist, dass keine Initialisierungspose benötigt wird, die das Skeleton Tracking aktiviert, wie es bei OpenNi der Fall ist. Ausschlaggebend für die Entscheidung, das Microsoft Framework für die Software Entwicklung zu nutzen, sind daher das Skeleton Tracking, die solide Speech Recognition sowie die ausführliche Dokumentation und die zahlreichen Samples.

4.3.2 Auswahl der direkten 1:1 Umsetzung

Die direkte 1:1 Umsetzung wird in dieser Arbeit für den Oberkörper verwendet. So ist es eine sehr natürliche und intuitive Art, die Bewegungen der Arme direkt vom Menschen, auf den Roboter zu übertragen. Der Vorteil hierbei ist, dass die Kontrolle stets beim Menschen liegt und spontan auf Ereignisse reagiert werden kann, was sich bei fest definierten Bewegungsabläufen als schwierig darstellt.

4.3.3 Auswahl der Behavior Umsetzung

Ein vorprogrammiertes Behavior hat den Vorteil, komplexe Bewegungen, wie dem Gehen beispielsweise, in spezieller und vorgegebener Form sicher auszuführen. Ebenso sind nicht erkennbare Be-

wegungen, die beispielsweise durch andere Körperteile verdeckt sind, durch ein Behavior dennoch ausführbar. Des Weiteren sind die Behavior reproduzierbar, d.h. sie werden immer gleich ausgeführt. Die Steuerung über Behavior wird genau da eingesetzt, wo auf eine intuitive und spontane Steuerung verzichtet werden kann. Sofern sich der Untergrund nicht ändert ist das Gehen eine Tätigkeit, die immer gleich ausgeführt werden kann. Hierbei ist vor allem wichtig, eine stabile Position zu wahren und nicht umzufallen oder zu stolpern. Die Behavior können auch da eingesetzt werden, wo die Erkennung der menschlichen Bewegung momentan noch an ihre Grenzen stößt. So kann beispielsweise das Drehen der Unterarme oder das Öffnen und Schließen der Hände über ein Behavior realisiert werden. In dieser Arbeit werden Behavior vorallem für die grundlegenden Beinbewegungen, wie, sich um die eigene Achse drehen, seitwärts und vor und zurück laufen, verwendet.

4.3.4 Steuerung der Behavior durch Spracheingabe

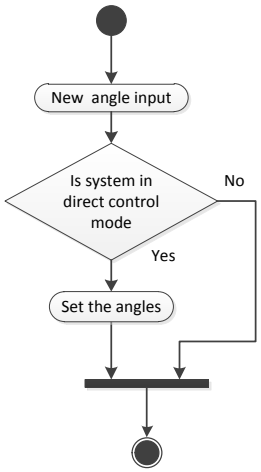

Abbildung 4.3: Ablauf bei neuen Winkel Daten

Ein wesentlicher Vorteil für die Benutzung der Spracheingabe zum Abrufen eines Behaviors ist der, dass die Eingabe der direkten Bewegung (über die Erkennung der Körperbewegung) und die der aufzurufenden Behavior (über die Erkennung der Sprache) getrennt stattfindet. So ist es möglich zeitgleich direkte Bewegungen auszuführen, sowie ein Behavior aufzurufen sofern dieses nicht blockierend ist. Blockierend bedeutet, dass das Behavior nur ausgeführt werden kann, wenn zeitgleich das Ausführen von direkten Bewegungen verhindert wird. Hierzu werden zwei Modi verwendet. So ermöglicht der „direct control mode“ die direkte Übertragung der Bewegung an den Roboter. Der „behavior control mode“ hingegen erlaubt die Ausführung von Bewegungen mit Hilfe der Behavior. Die Abbildung 4.3 zeigt den Ablauf, wenn neue Winkeldaten vorliegen. Sofern der direct control mode aktiv ist, können die Winkel an den Roboter übertragen werden, wo sie dann auf die verschiedenen Motoren der Gelenke übertragen werden. Ist der behavior control mode aktiv, können keine Winkel direkt gesetzt werden. So sind in diesem Fall Bewegungen nur noch über die Behavior realisierbar.

Die Abbildung 4.4 stellt den Ablauf, bei der Eingabe eines neuen Behaviorbefehls, dar. Liegt ein neuer Behavior Aufruf vor, wird zunächst ermittelt, ob das vorliegende Behavior blockierend ist,

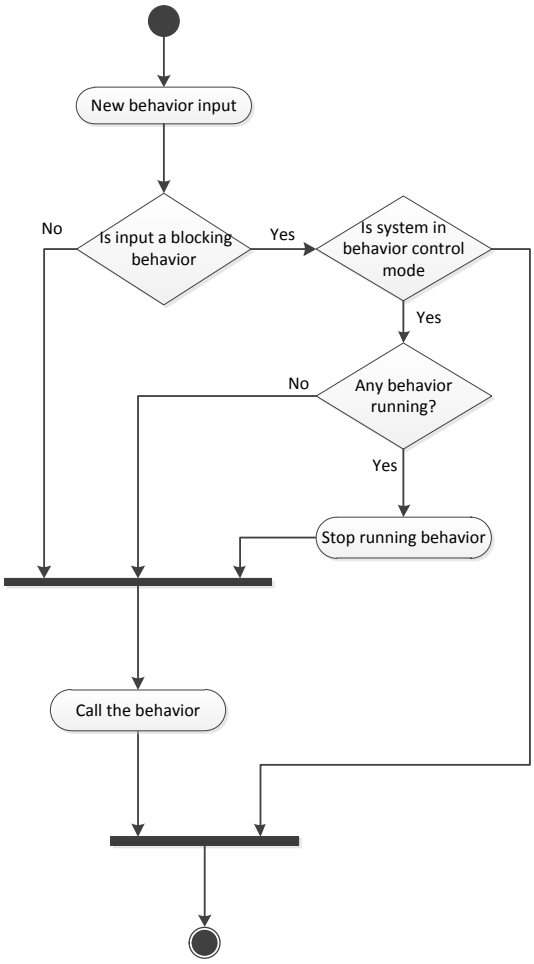

Abbildung 4.4: Ablauf bei neuen Behavior Daten

also eines, das nur ausführbar ist, wenn der behavior control mode aktiv ist. Handelt es sich um ein nicht blockierendes Behavior wie beispielsweise „open right hand“, kann dieses sowohl im direct control mode, als auch im behavior control mode ausgeführt werden. Die beiden Betriebsmodi sind also dazu da, den Roboter in einer robusten Art und Weise zu steuern. Hierdurch werden ungeeignete Steuerungen von vorne herein ausgeschlossen. So ist es beispielsweise nicht möglich, die Arme des Roboters zu kontrollieren während dieser im behavior control mode aufsteht. Dies hat den Vorteil, dass das „Aufsteh“ Behavior unbeeinflusst ausgeführt wird. Es ist ungeschickt, wenn die Arme dabei ruckartig bewegt werden können, da sich der Roboter beim Aufstehen selbst balanzieren muss. Hier können schnelle Armbewegungen das Gleichgewicht beeinträchtigen was zum Umfallen führen kann. Die Tabelle 4.3 zeigt die ausführbaren Behavior, deren Beschreibung und ob sie blockierend oder nicht blockierend sind.

Durch das Verwenden der Spracheingabe ist das Aufrufen der Behavior sehr einfach zu erlernen, orientieren sich doch die meisten Befehle genau an dem auszuführenden Behavior. So wird für das Behavior „Hinsetzen“ beispielsweise der Befehl „sit down“ verwendet.

Behavior:	Beschreibung:	Blockade Eigenschaft:
stop	hält alle laufenden Behavior an	nicht blockierend
stand up	lässt den Roboter aufstehen	blockierend
sit down	lässt den Roboter hinsetzen	blockierend
rotate left	dreht den Roboter entgegen den Uhrzeigersinn	blockierend
rotate right	dreht den Roboter im Uhrzeigersinn	blockierend
move sideway left	lässt den Roboter seitwärts nach links laufen	blockierend
move sideway right	lässt den Roboter seitwärts nach rechts laufen	blockierend
move ahead	lässt den Roboter vorwärts laufen	blockierend
move back	lässt den Roboter rückwärts laufen	blockierend
open left hand	öffnet die linke Hand	nicht blockierend
open right hand	öffnet die rechte Hand	nicht blockierend
close left hand	schließt die linke Hand	nicht blockierend
close right hand	schließt die rechte Hand	nicht blockierend
take the shopping cart	greift nach dem Einkaufswagen	blockierend
release the shopping cart	lässt den Einkaufswagen los	blockierend
turn left hand left	rotiert linken Unterarm links	nicht blockierend
turn left hand right	rotiert linken Unterarm rechts	nicht blockierend
turn right hand left	rotiert rechten Unterarm links	nicht blockierend
turn right hand right	rotiert rechten Unterarm rechts	nicht blockierend

Tabelle 4.3: Auflistung der Behavior und ihre Eigenschaft

4.4 Realisiertes Konzept

4.4.1 Erfassung der Gelenkpunkte

Das Microsoft Kinect beta SDK verfügt bereits über ein Skeleton Tracking, das 20 Gelenkpunkte (Joints) erkennt (siehe Abbildung 4.5). Über Algorithmen ist es möglich, anhand der Sensoren der Kinect Kamera die dreidimensionalen Positionen der Gelenkpunkte zu bestimmen. Mit Hilfe der Positionen können die Winkel zwischen den Gliedmaßen berechnet werden. Die Werte der Winkel können dann an den Roboter gesendet werden. Dieser setzt seine Winkel entsprechend den erhaltenen Winkeln, wodurch ermöglicht wird, direkte vom Menschen vorgemachte Bewegungen, auf dem Roboter auszuführen.

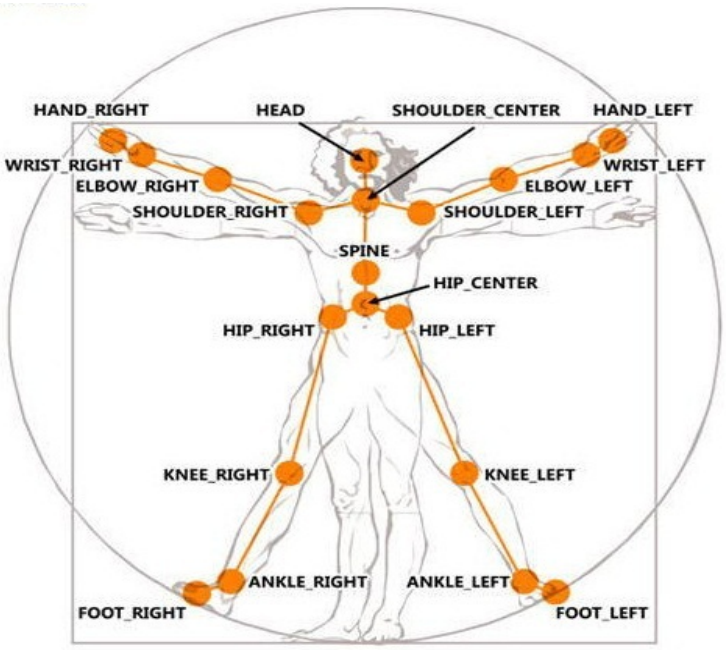

Abbildung 4.5: Überblick über vom Framework erkannte Gelenkpunkte [Micd, Seite 20].

4.4.1.1 Gelenkwinkelberechnung aus 3D Punktwolke

Durch die vom Skeleton Tracking gelieferten Gelenkpunkte ist es möglich, Vektoren aufzustellen. So wird beispielsweise vom linken Ellenbogen zur linken Hand und vom linken Ellenbogen zur linken Schulter jeweils ein Vektor aufgestellt (siehe Abbildung 4.6). Die Vektoren werden dabei so aufgestellt, dass sie weg vom zu bestimmenden Winkel ausgerichtet sind. Abbildung 4.7 zeigt den eingeschlossenen Winkel β zwischen den beiden Vektoren BC und BA. Die Vektoren werden durch die Stecke zwischen den Punkten A, B und C bestimmt. Im illustrierten Beispiel (Abbildung 4.6) wird der Winkel für den linken Ellenbogen mit den Vektoren *v1 = WRIST_LEFT - ELBOW_Left* und *v2 = WRIST_LEFT - SHOULDER_LEFT* errechnet. Hierfür wird die Arcus-Cosinus Funktion verwendet, die den eingeschlossenen Winkel β zwischen den zwei gegebenen Vektoren v1 und v2 wie folgt berechnet:

$$\beta = \arccos\left(\frac{v1 \cdot v2}{|v1| \times |v2|}\right) \tag{4.1}$$

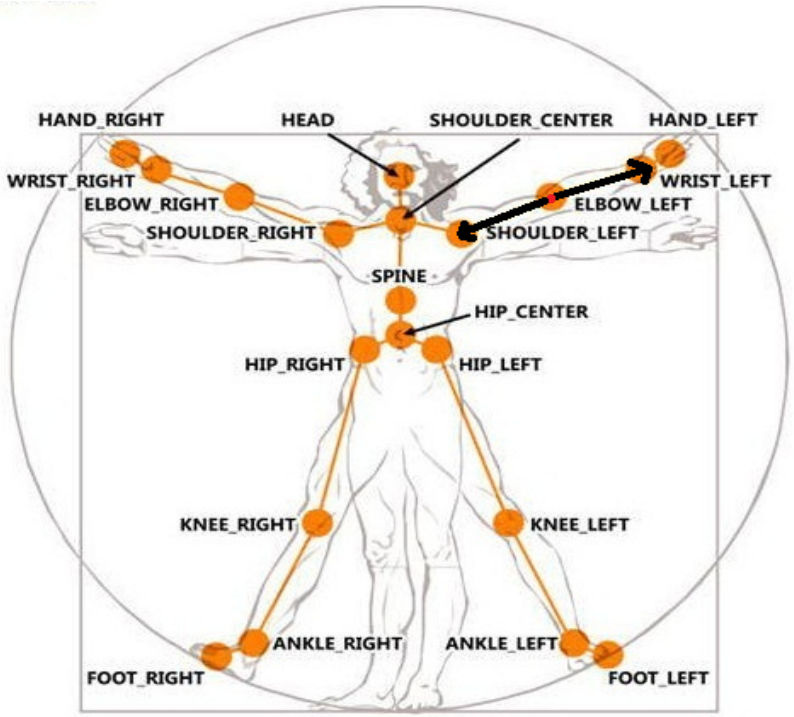

Abbildung 4.6: Aufgestellte Vektoren am linken Ellenbogen

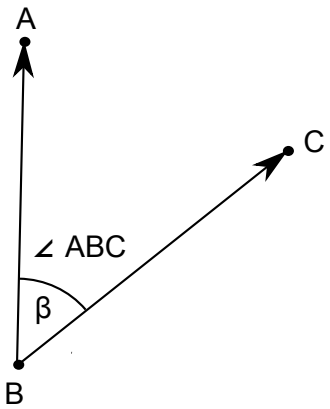

Abbildung 4.7: Eingeschlossener Winkel β zwischen Vektoren $\vec{BA}$ und $\vec{BC}$

Da der Arcus-Cosinus symmetrisch ist, gilt: $\arccos(-x) = \pi - \arccos(x)$ was zu einem Wertebereich von $0 \leq f(x) \leq \pi$ führt. In der Tabelle 4.4 sind die möglichen Wertebereiche der Gelenkwinkel des NAOs aufgelistet. Die verwendeten Gelenke sind grün markiert. Dabei ist zu erkennen, dass die meisten Winkel einen maximalen Wertebereich von $0 \leq \pi$ aufweisen. Der Grund, nicht alle Winkel zu berechnen ist auf die Entscheidung, die direkte Steuerung der Bewegung nur für die Arme zu verwenden, zurückzuführen. Wie bereits im Kapitel 3 erläutert kann der Mensch nicht immer alle Gelenke genauso anwinkeln, wie es dem NAO möglich ist. Dadurch ist die Einschränkung der Winkelberechnung, die einen Halbwinkel, also stets $0 \leq \pi$ berechnet, recht gering. Um die errechneten Winkel später korrekt an den NAO schicken zu können, werden die Daten der Winkel zusätzlich noch mit einem Namen und einem Wert für die Geschwindigkeit der Gelenkmotoren versehen. Für das ruhige Bewegen der Arme ist für jeden benutzten Gelenkwinkel eine Interpolation durch das arithmetische Mittel über die letzten 10 Werte des Gelenkwinkels verantwortlich.

Joint	Range in degrees
HeadYaw	-119.5 to 119.5
HeadPitch	-38.5 to 29.5
RShoulderPitch	-119.5 to 119.5
RShoulderRoll	-94.5 to -0.5
RElbowYaw	-119.5 to 119.5
RElbowRoll	0.5 to 89.5
RWristYaw	-104.5 to 104.5
RHand	Open And Close
LShoulderPitch	-119.5 to 119.5
LShoulderRoll	0.5 to 94.5
LElbowYaw	-119.5 to 119.5
LElbowRoll	-89.5 to -0.5
LWristYaw	-104.5 to 104.5
LHand	Open And Close
LHipYawPitch	-65.62 to 42.44
RHipYawPitch	-65.62 to 42.44
LHipRoll	-21.74 to 45.29
LHipPitch	-101.63 to 27.73
LKneePitch	-5.29 to 121.04
LAnklePitch	-68.15 to 52.86
LAnkleRoll	-44.06 to 22.79
RHipRoll	42.30 to 23.76
RHipPitch	-101.54 to 27.82
RKneePitch	-5.90 to 121.47
RAnklePitch	-67.97 to 53.40
RAnkleRoll	-22.27 to 45.0

Tabelle 4.4: Mögliche Wertebereiche der Gelenkwinkel des NAOs (vgl. [Robb]).

4.4.2 Spracherkennung für die Behaviorsteuerung

Mit Hilfe der von Microsoft angebotenen Spracherkennungsalgorithmen ist es möglich, ausgesprochene Sprache zu erkennen und diese gewünschten Befehlen zuzuordnen. Da der direkte Einblick in die Spracherkennungsalgorithmen von Microsoft momentan nicht gestattet ist, werden einige Details dazu ausgelassen und der Gesamtaufbau nur grob beschrieben. Zu Beginn wird eine Grammatik mit den gewünschten zu erkennenden Befehlen (wie beispielsweise „sit down“) gebildet, die dann an die Sprachengine übergeben wird. Im vorliegenden Fall werden die Befehle als Sätze abgelegt. So ist es nicht notwendig alle hinterlegten einzelnen Wörter zu erkennen, sondern lediglich die in der Grammatik hinterlegten Sätze. Dies hat den Vorteil, dass die Sprachengine nur die kompletten Sätze und nicht zwischen einzelnen Wörtern unterscheiden muss. Durch die Vorgabe der Grammatik wird der zu erkennenden Wortraum eingeschränkt.

4.4.2.1 Transformation der Sprachbefehle in Behavior

Der Aufruf eines Behaviors auf dem NAO geschieht über dessen Behavior-Proxy. Dazu wird lediglich der Behavior Name benötigt. Ein von der Spracherkennung erkannter Befehl wird als string angegeben. Da die Grammatik der Spracherkennung mit den Namen der Behavior erstellt wurde, kann der erkannten Befehl direkt für den Aufruf eines Behaviors verwendet werden. Zuvor wird jedoch ermittelt, ob das Behavior blockierend ist oder nicht. Ist dies der Fall, wird ein Flag gesetzt, dass je nach aktivem Modus verhindert, dass das blockierende Behavior ausgeführt und die direkte Ausführung von erfassten Bewegungen behindert wird.

4.4.3 Datenversand an den NAO

Wie bereits im Kapitel 3 erläutert, geschieht die Datenübertragung an den NAO wahlweise per WLAN oder Ethernet. Die API des NAOs wird per Proxies aufgerufen, wodurch es möglich ist, verschiedene Programmiersprachen zur Ansteuerung zu verwenden. In der vorliegenden Arbeit wird der NAO per WLAN angesteuert um eine kabellose Übertragung zu erhalten.

Zur Ansteuerung der API werden lediglich 4 Proxies genutzt (siehe Abbildung 4.9), darunter der Video-Device Proxy, um die Kamerabilder der im NAO integrierten Kameras zu erhalten. Hierdurch ist es möglich, den NAO auch über entfernte Netzwerke zu steuern, ohne direkten Sichtkontakt zu haben und lediglich die Kameras als Feedback zu nutzen. Aufgrund der Signalverzögerung im Netzwerk und die geringe Bildrate der Kameras (im vorliegenden Fall ca. 8fps) kann dies jedoch zu einer erschwerten und verzögerten Bedienung führen. Der TextToSpeech Proxy wird dazu genutzt um Sprachausgaben auf dem Roboter zu erzeugen, die als Feedback dem Benutzer dienen. So wird zum Beispiel beim Modus Wechsel der dann aktive Modus genannt, woraufhin der Benutzer erfährt, ob er den Roboter nun über Behavior oder über die direkte Abbildung der vorgemachten Bewegung steuert. Zusätzlich wird der Behavior Proxy zum Aufrufen der Behavior und der Motion Proxy, über den die Gelenke direkt über die Winkeleingabe gesetzt werden können, verwendet.

4.4.4 Teleoperation

Das Steuern des NAOs ohne direkten Sichtkontakt wird in dieser Arbeit auch aufgegriffen und berücksichtigt. So gibt es beispielsweise über die GUI des, für diese Arbeit entwickelten Programms „Teleoperation of NAO via Kinect“, die Möglichkeit, die Videobilder der im NAO eingebauten Kameras, anzeigen zu lassen. Dadurch ist es möglich, Feedback über die aktuelle Position des NAOs zu erhalten und seine Umgebung zu sehen. Ebenso sind Sicherheitsfeatures eingebaut wie beispielsweise

die bereits erwähnten „Bumpers“ (vgl. Kapitel 3.2) an den Füßen, die Laufbewegungen automatisch abbrechen, wenn Druck auf diese ausgeübt wird. Dieses Sicherheitsfeature wird durch Anpassung der Behavior mittels des Programms Choregraphe ermöglicht. Ein akustisches Feedback über die von der Spracherkennung erkannten Befehle werden über die Lautsprecher des Computers ausgegeben. Der NAO selbst gibt den auszuführenden Befehl ebenfalls über seine Lautsprecher aus. Zusätzlich ermöglicht er akustisches Feedback beim Moduswechsel. So ist es dem Benutzer möglich, zu erfahren, welcher Modus aktiv ist. Durch die Feedbacks kann der Benutzer somit über größere Entfernung und ohne direkten Sichtkontakt zum NAO diesen steuern. Hierbei muss allerdings beachtet werden, dass Verzögerungen bei der Übertragung der Steuerbefehle, aber auch der Feedbacks auftreten und die Steuerung beeinflussen können.

4.4.5 Softwareübersicht

Um die vorhandenen Hardware Ressourcen effizent auszunutzen wird beim Softwareentwurf darauf geachtet, wenn möglich parallelisierte Programmabläufe zu implementieren. Das Komponentendiagramm in Abbildung 4.8 zeigt die angedachten Komponenten. Jede Komponente läuft in einem eigenen Thread, um einen flüssigen Ablauf des Programms gewährleisten zu können. Im Diagramm ist unter anderem die Hauptkomponente, der Kinect-Manager, zu finden. Dieser ist für die kompletten Eingabedaten verantwortlich, was bedeutet, dass er die von der Kinect Kamera gelieferten Bild- aber auch die Audiodaten nutzt. Über die im Framework integrierte Skeleton Tracking Algorithmen bietet der Kinect-Manager die „Skeleton-Data“ dem Angle-Calculator zur Verfügung. Dieser errechnet aus der dreidimensionalen Punktwolke die Gelenkwinkel und bietet diese dem Proxy-Manager an. Der Kinect-Manager stellt der View die, von der Kinect Kamera gelieferten Videobilder, zur Verfügung, die dort angezeigt werden. Des Weiteren nutzt die Speech-Recognition Komponente die Audio Daten vom Kinect-Manager um Befehle zu erkennen. Wird ein Befehl erkannt, kann dieser sowohl über die Lautsprecher des Computers ausgegeben, als auch in ein Behavior transformiert und an den Proxy-Manager weitergeleitet werden. Im Proxy-Manager wird über die API des NAOs auf dessen Proxies zugegriffen, um die Winkel zu setzen oder ein Behavior aufzurufen. Ein Behavior wird auf dem NAO über dessen Lautsprecher angekündigt, um den Benutzer über die bevorstehenden Bewegungsabläufe des NAOs zu informieren. Über den Videoproxy des NAOs erhält der Proxy-Manager die Videodaten der im NAO integrierten Kameras, die an die View weitergeleitet werden, um diese dem Benutzer visuell zu präsentieren.

Der Ansatz scheitert in der praktischen Umsetzung. So ist es nach derzeitigem Kentnisstand nicht möglich, die View vom Haupt-Thread zu trennen. Ebenso ist die Anbindung der Kinect Kamera über das Natural User Interface (NUI) an den Haupt-Thread gebunden. Ob der Ansatz über ein anderes Projekt Template als dem Windows Presentation Foundation (WPF) realisiert werden kann, konnte in dem kurzen Zeitrahmen, den diese Arbeit bietet, letztlich nicht untersucht werden. Ebenso liegt der Fokus nicht auf den Implementierungen, sondern vielmehr auf der Untersuchung, wie eine Mensch-Maschine-Schnittstelle mit der Kinect Kamera ausgearbeitet werden kann. Eine Abwandlung des ursprünglichen Komponentendiagramms ist in Abbildung 4.9 zu sehen. Hier ist zu erkennen, dass sich die Struktur stark vereinfacht und der Datenfluss durch Zusammenkoppeln von Komponenten stark reduziert hat. So kann die Komponente UI direkt auf die Daten der Kinect zugreifen und bietet dem Benutzer nur noch das Skeleton Tracking, sowie dem Angle-Calculator die Skeleton-Data an. Das UI greift über einen Backgroundworker-Thread auf den Videoproxy des NAOs zu, um ein flüssiges Darstellen der Videos und gleichzeitig ein ruckelfreies Skeleton Tracking zu ermöglichen. Die Speech-Recognition Komponente greift direkt auf die Audio Daten des Mikrofons der Kinect Kamera zu, um bei erkannten Befehlen diese an den Proxy-Manager zu senden. Hier werden auch die vom

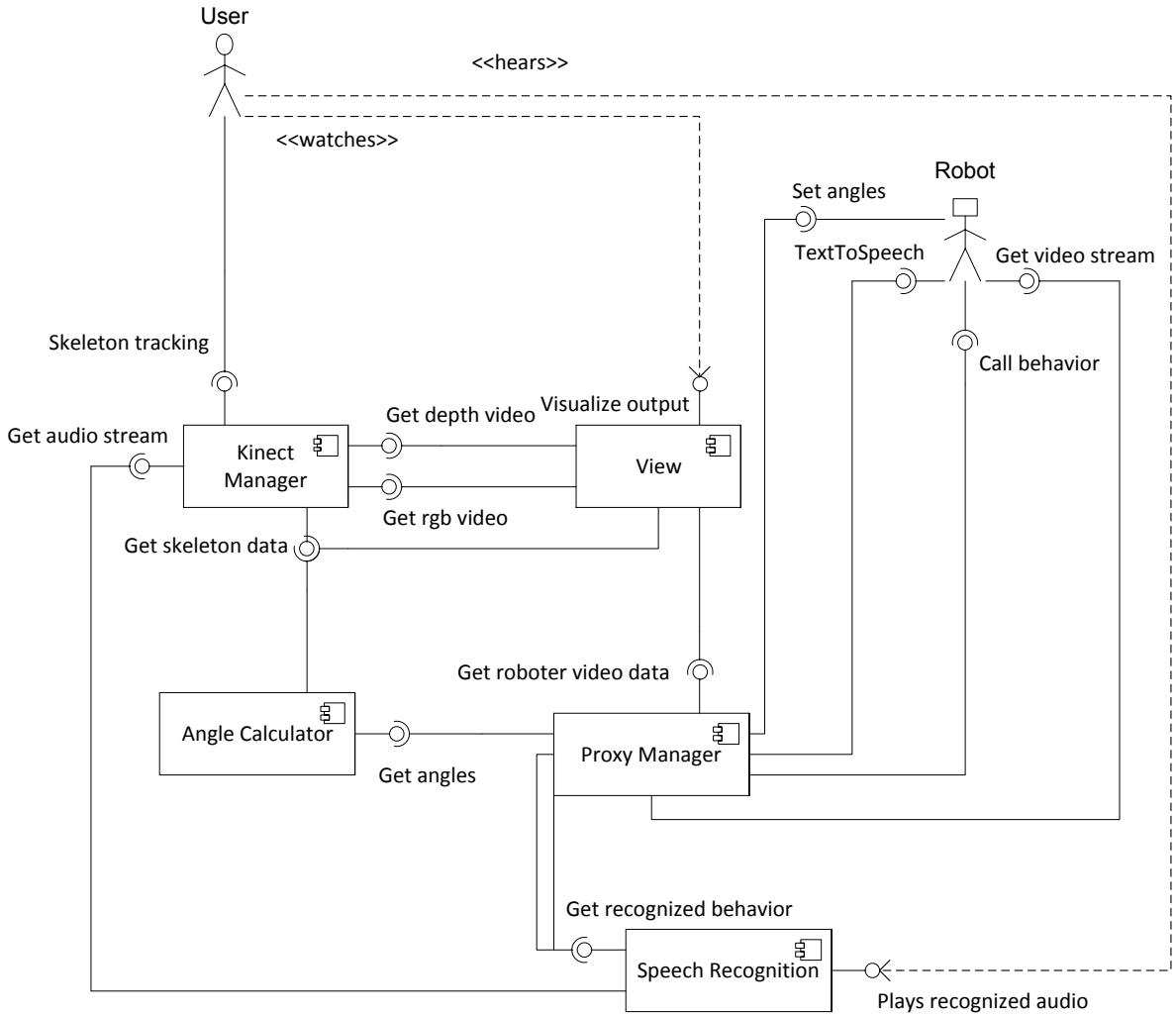

Abbildung 4.8: Komponentendiagramm des ursprünglichen Softwareentwurfs.

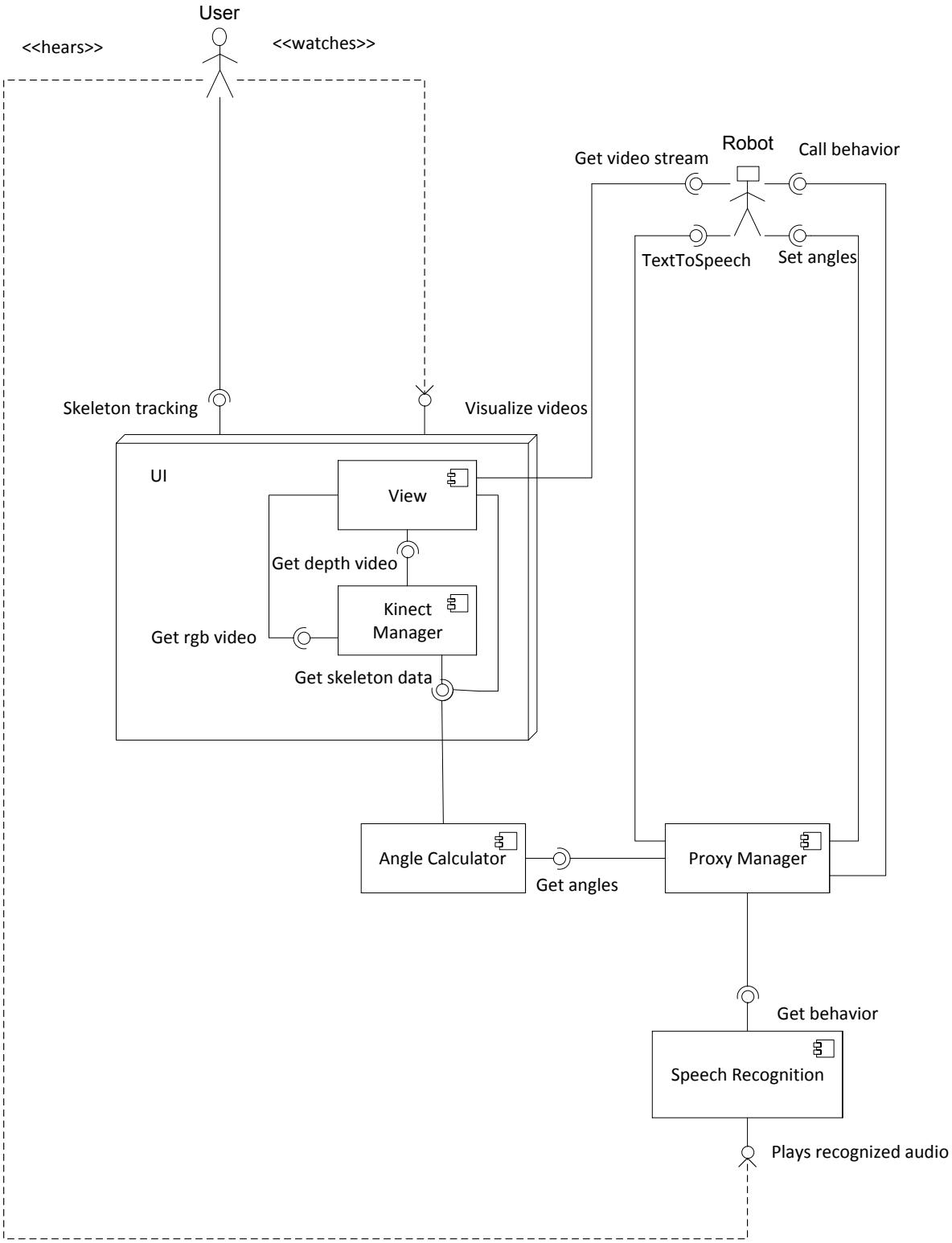

Abbildung 4.9: Komponentendiagramm des realisierten Softwareentwurfs.

Angle-Calculator berechneten Winkel weitergeleitet. Der Proxy-Manager sendet sowohl die Winkel, als auch die Behavior, über die entsprechenden Proxies, an den NAO. Der Benutzer erhält das gleiche Feedback, wie im ursprünglichen Softwareentwurf angedacht. Dies bedeutet, dass der NAO die empfangenen Behavior ausspricht, und die Speech Recognition die erkannten Befehle über die Computer Lautsprecher ausgibt.

Kapitel 5

Ergebnisse

5.1 Genauigkeit der errechneten Winkel

Abbildung 5.1: Manuell gesetzte Vektoren auf RGB Bild.

In dem folgenden Test wird untersucht, wie groß die Differenz zwischen den errechneten und den tatsächlichen Werten der Winkel ist. Die für die Tests verwendeten Winkel werden so gewählt, dass die für die Winkelberechnung eingesetzten Vektoren in einer Ebene senkrecht zu den Sensoren der Kinect Kamera liegen. Dies hat den Vorteil, dass keine Tiefeninformation, also die Entfernung zur Kamera, zu berücksichtigen ist. Erst dadurch ist es möglich, den tatsächlichen Winkel über das zweidimensionale RGB Bild zu erhalten. Zunächst erfolgt über ein Bildbearbeitungsprogramm das manuelle Setzen der Vektoren im Bild. Dabei ist zu beachten, dass die Vektoren stets in der Mitte der Körperteile liegen. Hierdurch ist es möglich, den tatsächlichen Winkel über das Bildbearbeitungsprogramm zu ermitteln und mit dem vom Programm errechneten Winkel zu vergleichen. Die manuell gesetzten Vektoren sind in roter Frabe in der rechten Hälfte der Abbildung 5.1 zu erkennen. Im oberen Teil des Bildes sind die Werte für den Winkel sowie die Länge der Vektoren ersichtlich. Die von der Kinect erfassten Werte sind mit den Werten aus den RGB Bildern in Abbildung 5.2 graphisch dargestellt. Die Abbildung

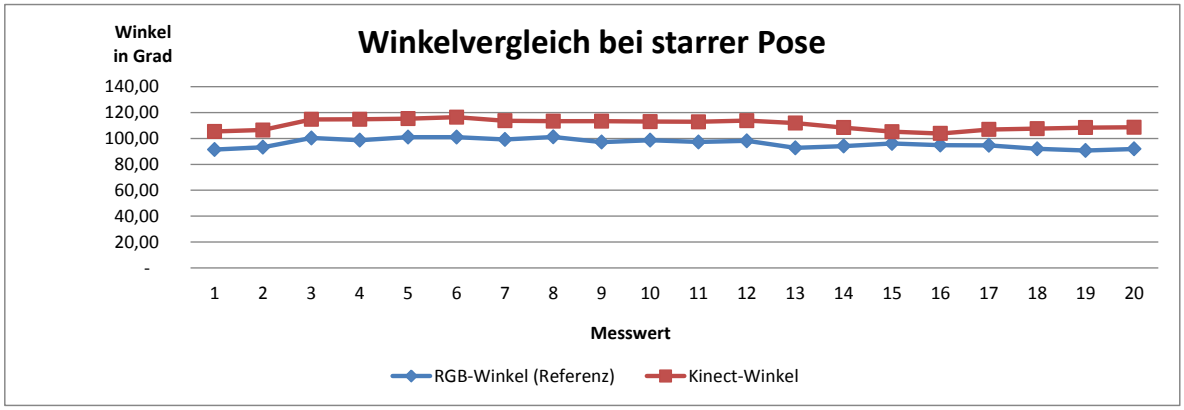

Abbildung 5.2: Messwerte der Winkel während einer starren Pose.

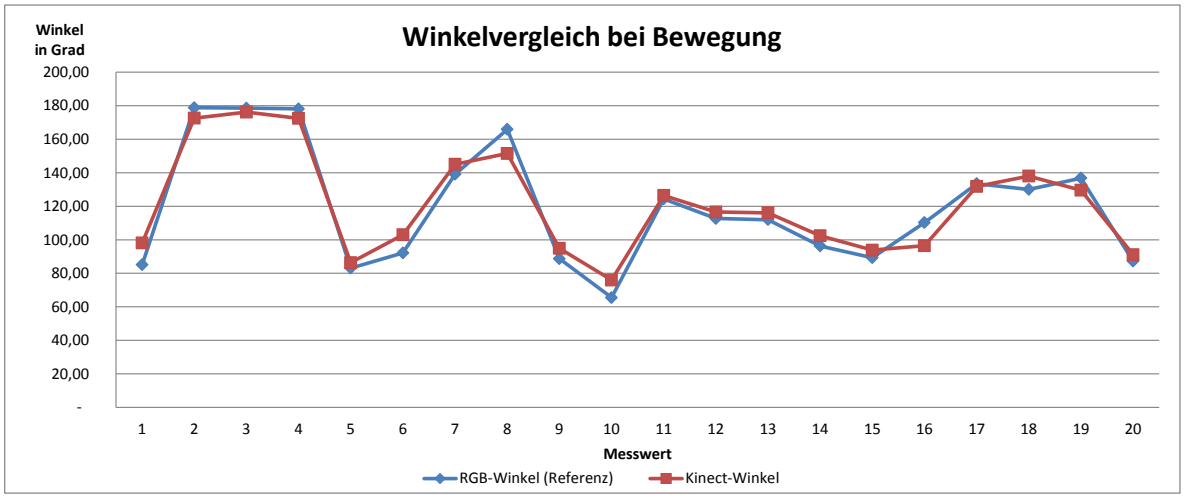

Abbildung 5.3: Messwerte der Winkel während einer Bewegung.

zeigt die Winkeldaten während einer starren Pose. Abbildung 5.3 veranschaulicht die Messdaten der Winkel für eine Bewegung. Zur Mittelwertbildung über die 20 Testwerte wird das quadratische Mittel genutzt. Dabei wird die Differenz zwischen den beiden Winkeln als Wert verwendet. Das quadratische Mittel kann wie folgt berechnet werden:

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} = \sqrt{\frac{x_1^2 + x_2^2 + \cdots + x_n^2}{n}} \tag{5.1}$$

Für die starre Pose ergibt sich danach das quadratische Mittel von 14,70°, für die Bewegung 7,68°. Interessant hierbei ist, dass bei einer Bewegung das quadratische Mittel kleiner ist, als bei einer starren Pose. Das bedeutet, dass die Kinect Kamera bei Bewegungen eine geringere Abweichung vom realen Wert hat, als bei einer festen Pose. Genauere Analysen zu dieser Abweichung können derzeit nicht durchgeführt werden, da momentan keine Möglichkeit besteht, Einblick in die Algorithmen des Skeleton Trackings im Microsoft Kinect beta SDK zu erhalten.

5.1.1 Erkenntnisse

Die Ungenauigkeit der von der Kinect erkannten Daten lassen momentan kein blindes steuern eines Roboters zu. Dies hat zur Folge, dass der Benutzer stets auf das visuelle Feedback des Roboters angewiesen ist, also bei der Steuerung wissen muss, wie die Arme des Roboters aktuell positioniert sind. Bei Bewegungen ist die Differenz zwischen der tatsächlichen Bewegung und der erkannten Bewegung kaum wahrnehmbar. Hier kann schnell und intuitiv nachgesteuert beziehungsweise nachjustiert werden, ohne dies bewusst wahrzunehmen.

5.2 Erkennungsrate der Speech Recognition

Die Erkennungsrate der Speech Recognition wird erfasst, indem 25-mal hintereinander, mit einer zeitlichen Verzögerung von mehreren Sekunden, derselbe zu erkennende Befehl ausgesprochen wird. Die Tabelle 5.1 zeigt, mit welcher Wahrscheinlichkeit die Speech Recognition den erkannten für den tatsächlich ausgesprochenen Befehl hält, was durch den Confidence Level angezeigt wird. Der Confidence Level zeigt also das Vertrauen in den erkannten Befehl an.

ausgesprochener Befehl:	Confidence Level:	erkannter Befehl
interrupt	0,9846181	interrupt
interrupt	0,9857874	interrupt
interrupt	0,9877933	interrupt
interrupt	0,9855625	interrupt
interrupt	0,9946504	interrupt
interrupt	0,9945194	interrupt
interrupt	0,9717951	interrupt
interrupt	0,9887742	interrupt
interrupt	0,9883341	interrupt
interrupt	0,9953384	interrupt
interrupt	0,9887477	interrupt
interrupt	0,9937552	interrupt
interrupt	0,9923376	interrupt
interrupt	0,9818094	interrupt
interrupt	0,9927278	interrupt
interrupt	0,9798459	interrupt
interrupt	0,9899032	interrupt
interrupt	0,9941162	interrupt
interrupt	0,9897043	interrupt
interrupt	0,9952732	interrupt
interrupt	0,9949704	interrupt
interrupt	0,9951068	interrupt
interrupt	0,9954007	interrupt
interrupt	0,9883948	interrupt

Tabelle 5.1: Auflistung der zu erkennenden und erkannten Befehle.

In der Tabelle ist zu sehen, dass der ausgesprochene Befehl zu 100% richtig erkannt wurde und

das Vertrauen in den erkannten Befehl mindestens bei 97% liegt. Wichtig hierbei ist zu erwähnen, dass die Grammatik der Spracherkennung nicht nur aus dem Befehl „interrupt“ besteht sondern noch weitere Befehle enthält. Ist nur ein Befehl in der Grammatik hinterlegt, kann die Spracherkennung auch nur diesen einen Befehl erkennen.

Im folgenden Testfall werden in zwei Durchläufen einige Befehle der Reihe nach ausgesprochen.

ausgesprochener Befehl:	Confidence Level:	erkannter Befehl:
open right hand	0,9949887	open right hand
open left hand	0,9919863	open left hand
close right hand	0,9831538	close right hand
close left hand	0,8455623	close right hand
move ahead	0,8747697	move ahead
move back	0,9904276	move ahead
move sideway left	0,9298196	move sideway left
move sideway right	0,9415983	move sideway right
rotate left	0,9884729	rotate left
rotate right	0,9848642	rotate right
stand up	0,9796394	stand up
break	0,9944098	break
break up	0,9944	break up
interrupt	0,9920613	interrupt
change mode	0,993807	change mode
take the shopping cart	0,8139592	take the shopping cart
release the shopping cart	0,9183079	release the shopping cart
open right hand	0,9911859	open right hand
open left hand	0,9930848	open left hand
close right hand	0,9924797	close right hand
close left hand	0,9899356	close left hand
move ahead	0,8927851	move ahead
move back	0,9753404	move ahead
move sideway left	0,9952771	move sideway left
move sideway right	0,9283205	move sideway right
rotate left	0,9818163	rotate left
rotate right	0,9764307	rotate right
stand up	0,9920824	stand up
break	0,9948415	break
break up	0,9841295	break up
interrupt	0,9942848	interrupt
change mode	0,9944898	change mode
take the shopping cart	0,8929281	take the shopping cart
release the shopping cart	0,8873061	release the shopping cart

Tabelle 5.2: Auflistung der zu erkennenden und erkannten Befehle bei variierenden Befehlen.

Ziel ist es, zu ermitteln, wie zuverlässig der ausgesprochene Befehl bei variierenden Befehlen

erkannt wird. In der Tabelle 5.2 werden die ausgesprochenen und die erkannten Befehle sowie deren Confidence Level aufgelistet. Erkannte Befehle, die nicht den ausgesprochenen entsprechen, sind grau markiert. Die Erkennungsrate liegt bei diesem Test bei ca. 91%, wobei die Erkennung von der genauen Aussprache abhängt. Je undeutlicher ein Befehl ausgesprochen wird, desto geringer ist die Wahrscheinlichkeit, den Befehl korrekt zu erkennen. Ebenso ist die Erkennungsrate abhängig von den herrschenden Umgebungsgeräuschen. Der Test zeigt, dass die Wahrscheinlichkeit, den korrekten Befehl zu erkennen, abnimmt, wenn verschiedene Befehle nacheinander ausgesprochen werden. Da der Befehl „move back“ in diesem Testlauf nie erkannt wird ist es durchaus möglich, dass dieser unverständlich oder falsch ausgesprochen wurde. Dabei sollte dann jedoch der Confidence Level niedriger sein, was darauf hinweisen könnte, dass der erkannte Befehl nicht dem ausgesprochene Befehl entspricht. Erstaunlich ist, dass nur einmal „left“ und „right“ falsch erkannt wurde und hierdurch der ausgesprochene Befehl „close left hand“ als „close right hand“ erkannt wurde. Es ist deshalb bemerkenswert, da sich die Befehle lediglich in diesen beiden Wörtern unterscheiden und hier die Wahrscheinlichkeit groß ist, left und right nicht korrekt zu erkennen.

5.2.1 Fehlerkennungsrate

Der folgenden Test wird während des normalen Labor Betriebs durchgeführt. Dabei wird die Spracherkennung für 5 Minuten eingeschaltet um zu testen, ob die Umgebungsgeräusche fehlerhaft als Befehle erkannt werden. Die folgende Tabelle 5.3 listet die erkannten Befehle, sowie ihren Confidence Level auf.

Befehl:	Confidence Level:
stand up	0,8448197
move ahead	0,8548739
move sideway left	0,9024703
stand up	0,9516287
move ahead	0,8856524
take the shopping cart	0,9126665
stand up	0,7951838
close right hand	0,8852706
move sideway left	0,8644001
open left hand	0,9437479
open left hand	0,8702832
open left hand	0,9457393
open left hand	0,8603933
move sideway left	0,8294297
move sideway right	0,8396908
open right hand	0,8475412
move ahead	0,8619782
open left hand	0,7932479
open left hand	0,8348107

Tabelle 5.3: Auflistung der fehlerhaft erkannten Befehle bei einem Test über fünf Minuten

Bei einem erkannten Befehl wird eine Audiodatei erzeugt um später analysieren zu können, ob

Ähnlichkeiten zu den in der Sprachgrammatik hinterlegten Befehlen besteht. Dies ist jedoch bei keinem der fehlerhaft erkannten Befehle der Fall. Wie die Tabelle zeigt, werden doch recht häufig Befehle erkannt, obwohl diese gar nicht ausgesprochen werden. Dies verdeutlicht auch nochmals die Entscheidung, der Einführung der beiden Modi. Diese verhindern, dass während des direct control mode blockierende Behavior, also beispielsweise Behavior die etwas mit Beinbewegung zu tun haben, aufgerufen werden und so unvorhergesehene Ereignisse eintreten können. Der Confidence Level der fehlerhaft erkannten Befehle liegt nicht über 95% was zulässt, Befehle mit einem Confidence Level unter 95% zu ignorieren und so fehlerhaft erkannte Befehle auszuschließen.

5.3 Teilszenarien

In diesem Abschnitt werden Teilszenarien vorgestellt, wie beispielsweise dem Anlaufen von Zielen, dem Greifen des Einkaufwagens, dem Laufen mit dem Einkaufwagen oder dem Greifen von Objekten. Jedes Teilszenario wird jeweils ohne Unterbrechungen mit einer Videokamera aufgenommen. Die folgenden Abbildungen zeigen Ausschnitte aus den Videos. Die Teilszenarien sind Voraussetzung für das Gesamtszenario, das ohne die erfolgreiche Ausführung der Teilszenarien nicht realisierbar ist, da das Gesamtszenario größtenteils aus den Teilszenarien besteht. Der Ablauf der Szenen ist visualisiert, wobei Abbildung 5.4 die Bedeutung der verwendeten Symbole erläutert.

Abbildung 5.4: Bedeutung der Symbole in den Visualisierungen der Abläufe

5.3.1 Ansteuern von Ziel Positionen

Das hier in Abbildung 5.5 gezeigte Teilszenario „Ansteuern von Ziel Positionen“ wird durch eine Kombination von 4 vorprogrammierten Behaviors realisiert. Jede Zeile in der Abbildung zeigt ein zusammenhängendes Behavior, wobei links immer der Anfang des Behaviors zu sehen ist und rechts das Ende. Der Ablauf des Szenarios ist in Abbildung 5.6 visualiziert.

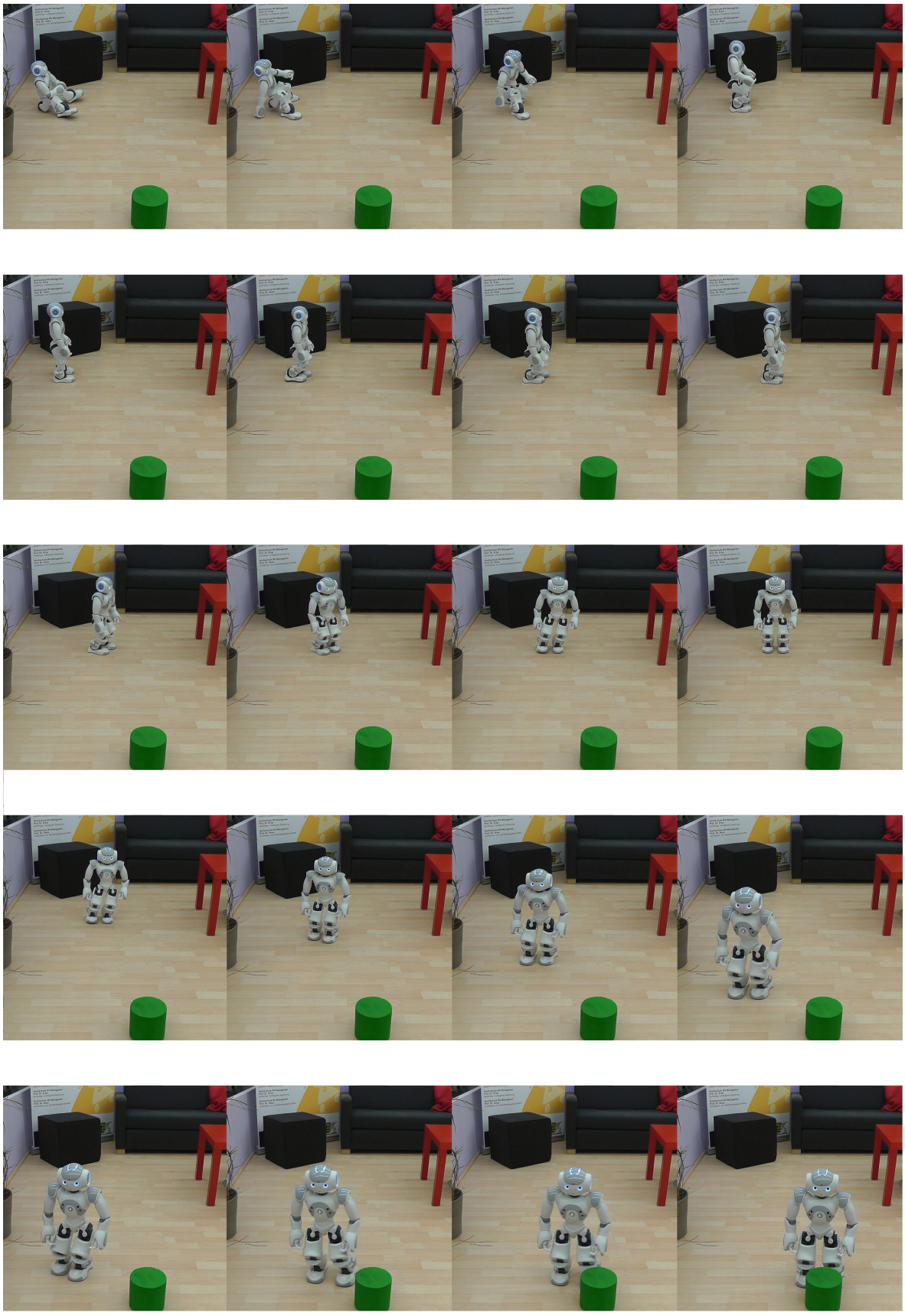

Abbildung 5.5: Bilderserie für das Teilszenario „Ansteuern von Ziel Positionen“. Jede Zeile zeigt von links, nach rechts ein Behavior.

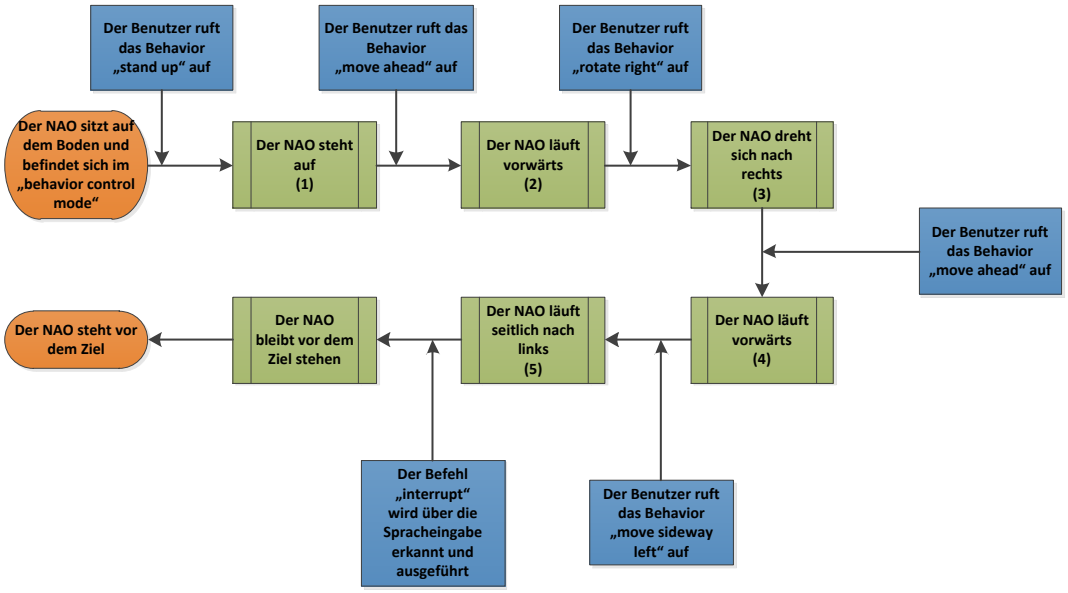

Abbildung 5.6: Visualisierung des Ablaufs der Abbildung 5.5. Die sich in Klammern befindenden Zahlen stehen für die zu beschreibende Zeile der Abbildung.

5.3.2 Greifen des Einkaufwagens

Die Abbildung 5.8 zeigt das Teilszenario „Greifen des Einkaufwagens“ das ebenfalls über ein Behavior realisiert werden kann, die Visualisierung des Ablaufs ist in Abbildung 5.8 zu sehen. Wird diese Bewegung im direct control mode ausgeführt, ist es möglich, dass sich die Roboter Arme nicht in der korrekten Ausgangsstellung befinden und sich der NAO beispielsweise zu sehr gegen den Einkaufswagen stemmt. Dies kann beim Laufen zum Umfallen führen, was eine Unsicherheit darstellt. Abhilfe kann durch ein Behavior geschaffen werden, dass das Gleichgewicht nach dem Greifen wieder herstellt. Des Weiteren ist es komfortabler, eine sich immer wiederholende Aufgabe automatisiert ablaufen zu lassen. Das Teilszenario zeigt, dass eine nicht triviale Bewegung, wie es das Greifen eines Einkaufswagens darstellt, mit Hilfe eines Behaviors möglich ist.

5.3.3 Laufen mit dem Einkaufswagen

Das folgende Teilszenario baut auf das Teilszenario „Greifen des Einkaufwagens“ auf und wird direkt im Anschluss ohne Unterbrechung mit der Videokamera weiter aufgezeichnet. Das bedeutet, dass für dieses Teilszenario notwendigerweise vorher das Teilszenario „Greifen des Einkaufwagens“ erfolgreich ausgeführt werden muss. Danach kann, wie in Abbildung 5.10 zu sehen, das Behavior „move ahead“ ausgeführt werden, was den NAO, den Einkaufswagen schiebend, vorwärts laufen lässt. Der Ablauf des Szenarios ist in Abbildung 5.9 visualiziert. Mit diesem Teilszenario wird gezeigt, dass die Behavior kombinierbar sind und nacheinander aufgerufen, einen komplexen Bewegungsablauf, wie es das Greifen nach einem Einkaufswagen und das Schieben selbigen darstellt, ermöglichen.

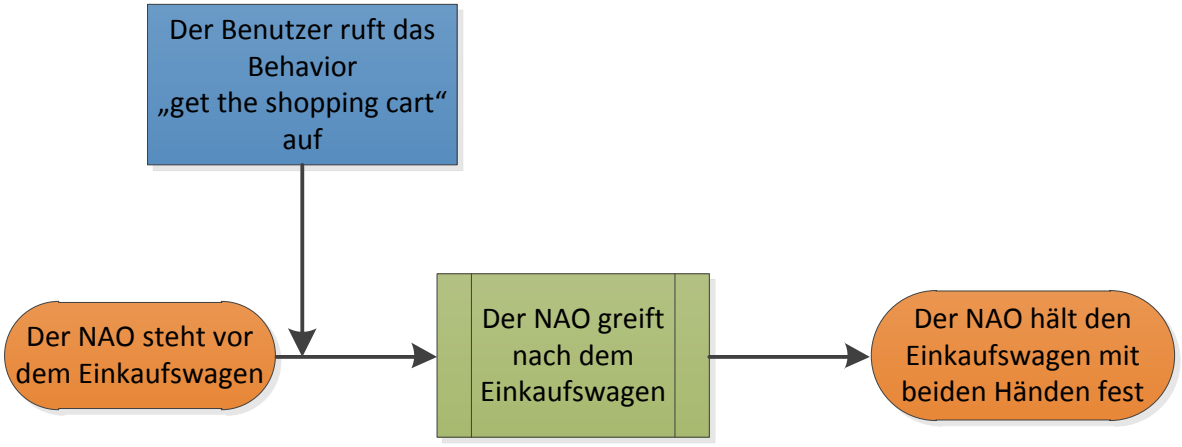

Abbildung 5.7: Visualisierung des Ablaufs der Abbildung 5.8.

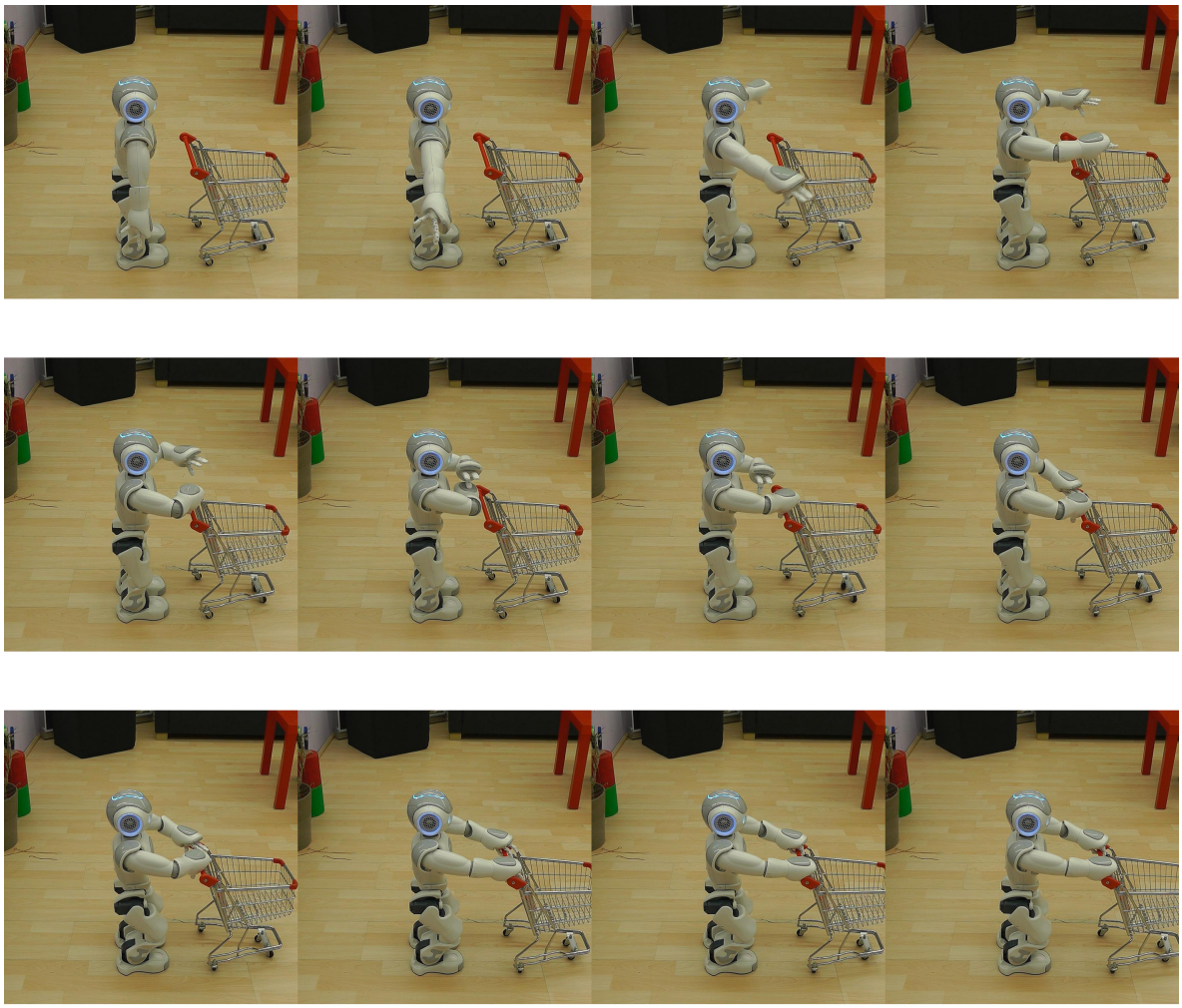

Abbildung 5.8: Bilderserie für das Teilszenario „Greifen des Einkaufwagens“. Das Szenario fängt links oben an und endet rechts unten.

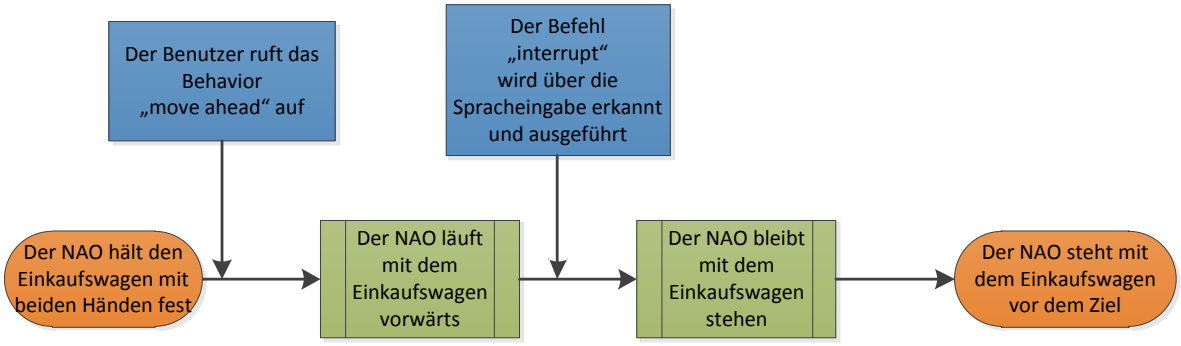

Abbildung 5.9: Visualisierung des Ablaufs der Abbildung 5.10

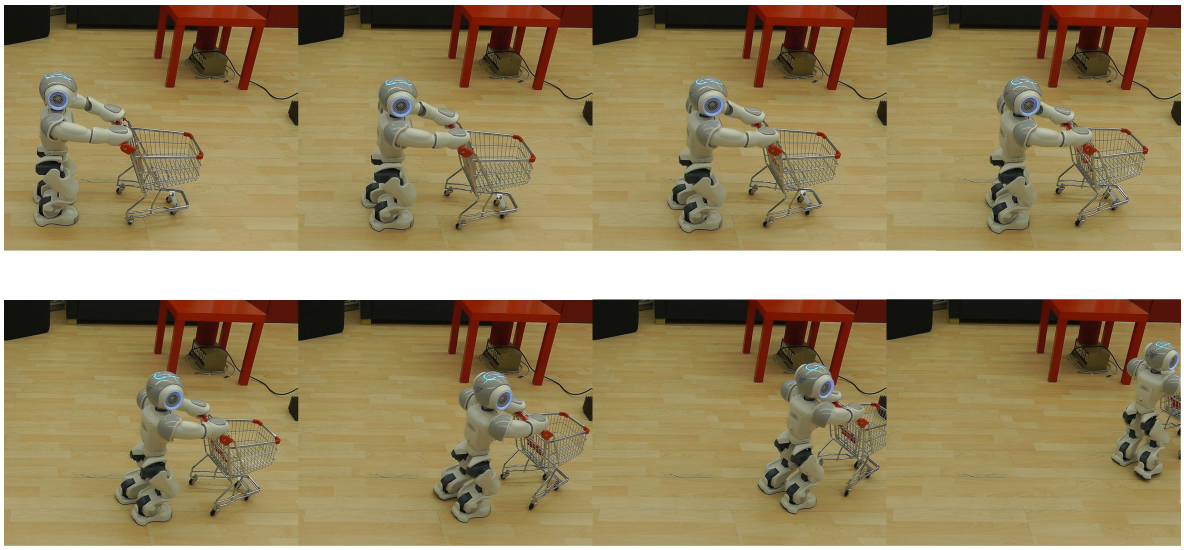

Abbildung 5.10: Bilderserie für das Teilszenario „Laufen mit dem Einkaufswagen“. Das Szenario fängt links oben an und endet rechts unten.

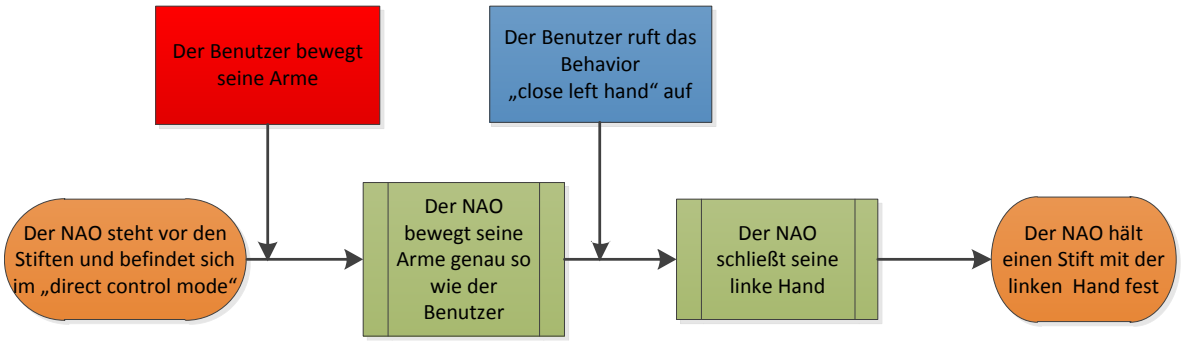

Abbildung 5.11: Visualisierung des Ablaufs der Abbildung 5.12.

5.3.4 Greifen und ablegen eines Objekts

Dieses Teilszenario zeigt eine Bewegung im direct control mode. Abbildung 5.12 zeigt, wie per Skeleton Tracking die Roboterarme so in Position gebracht werden, dass über das Behavior „ close right hand“ die rechte Hand geschlossen wird und das Objekt festgehalten wird. Die Abbildung 5.11 visualisiert den Ablauf der Abbildung 5.12. Die Abbildung 5.14 zeigt, wie wieder über das Skeleton Tracking die Roboterarme über den Einkaufswagen manövriert werden, um hier die Hand zu öffnen und das Objekt in den Einkaufswagen abzulegen. Die Abbildung 5.13 visualisiert diesen Ablauf. In diesen beiden Teilszenarien wird der direct control mode gezeigt. Ebenso wird das Zusammenspiel beider Techniken (direct control und behavioral control) durch das Behavior „open right hand“ und der direkten Übertragung der Armbewegungen demonstriert. Die Teilszenarien zeigen, dass das Greifen und Positionieren von Objekten über die direct control Technik möglich und sinnvoll ist, da das Verwenden der Behavior hier aufwendig sein kann, da alle möglichen Anwendungsszenarien vorprogrammiert sein müssten.

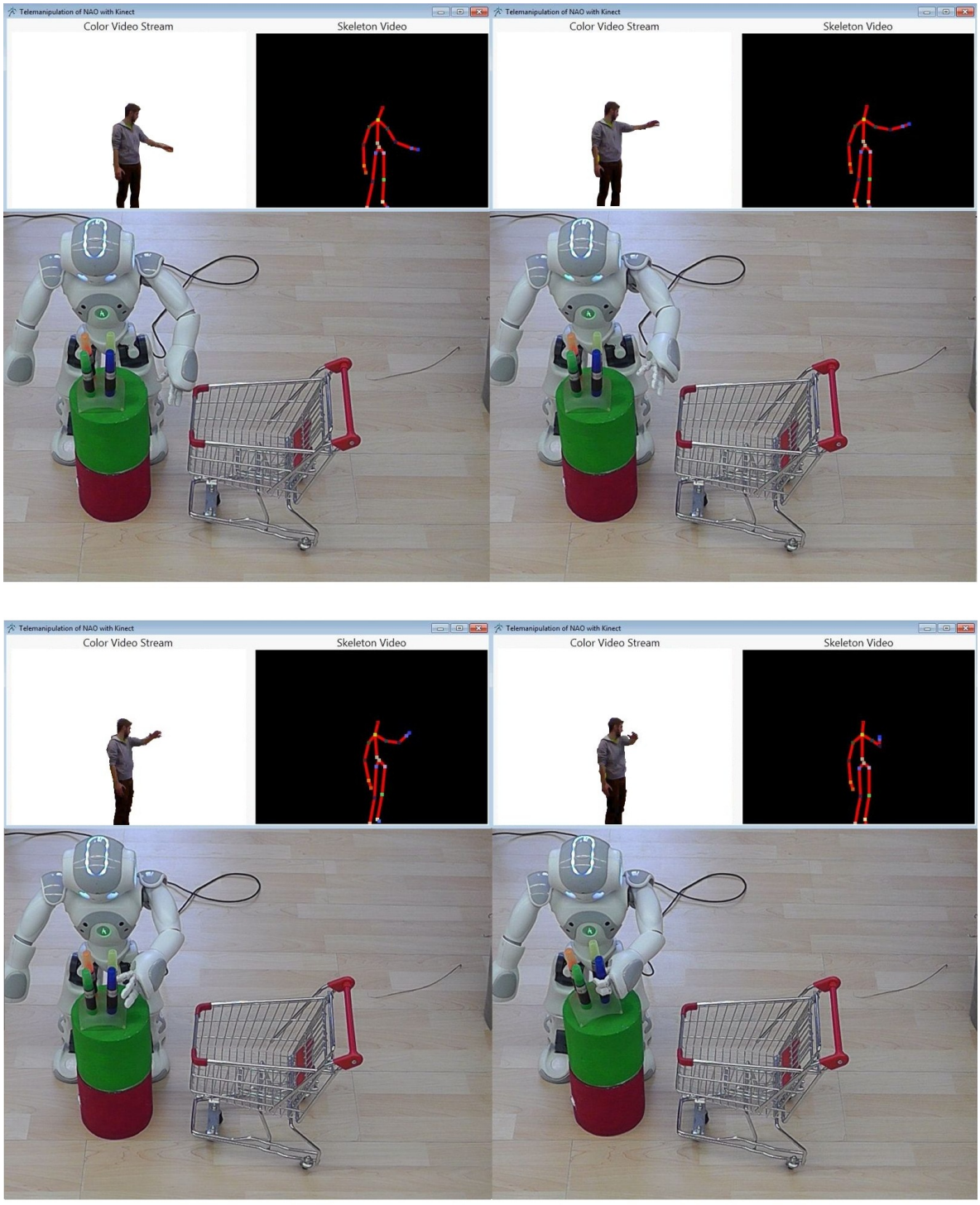

Abbildung 5.12: Bilderserie für das Teilszenario „Greifen und ablegen eines Objekts“. Das Szenario fängt links oben an und endet rechts unten.

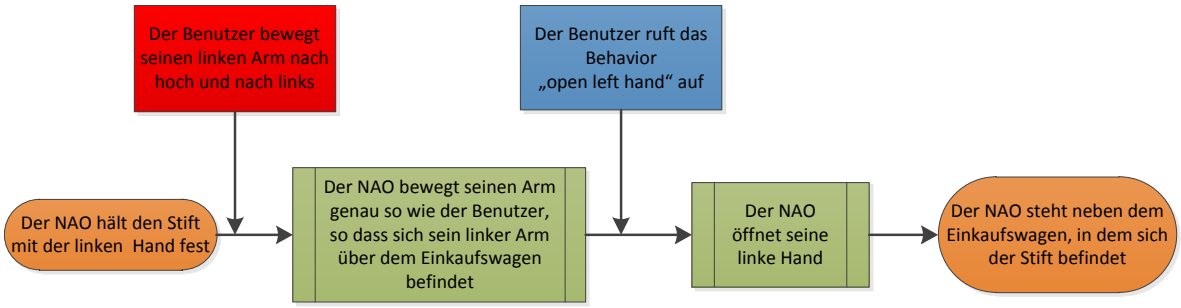

Abbildung 5.13: Visualisierung des Ablaufs der Abbildung 5.14

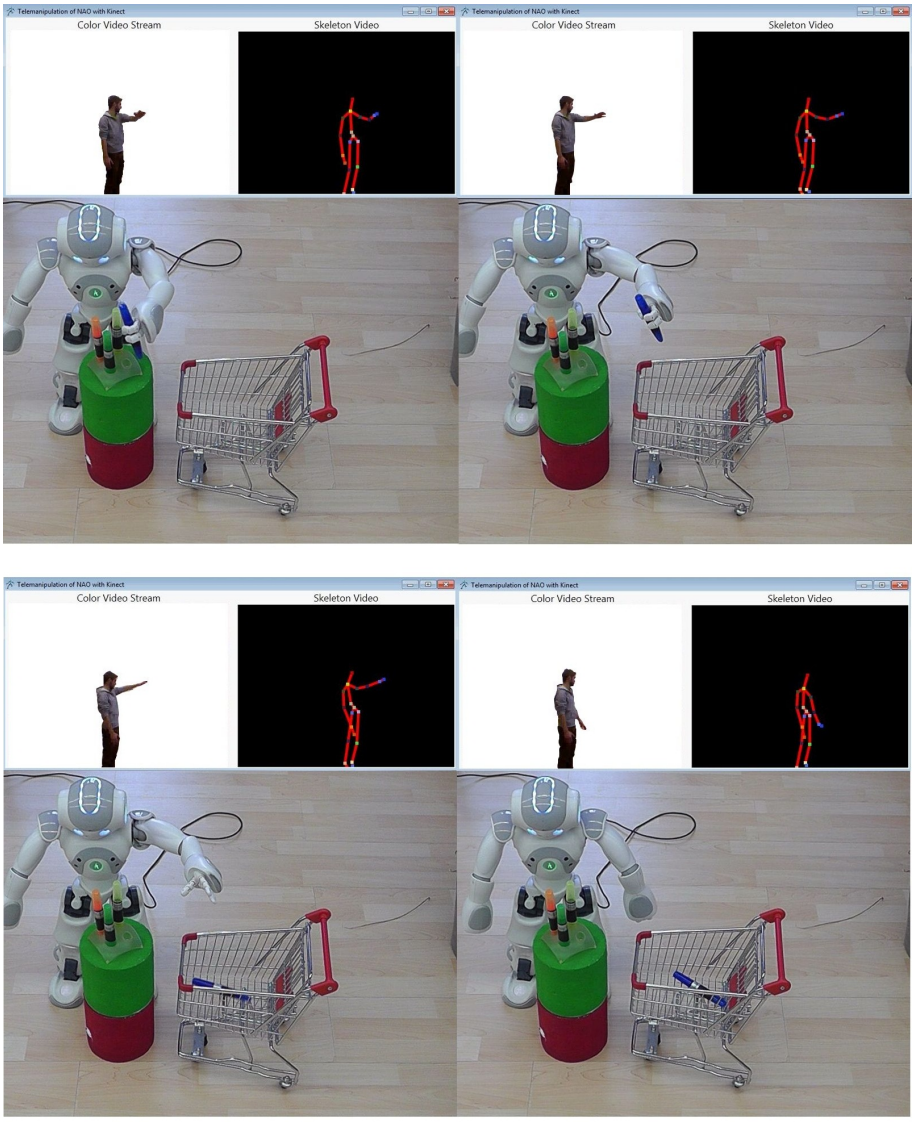

Abbildung 5.14: Bilderserie für das Teilszenario „Greifen und ablegen eines Objekts“. Das Szenario fängt links oben an und endet rechts unten.

5.4 Einkaufsszenario

Das Gesamtszenario zeigt, in Form einer realistischen Einkaufsszene, die Komplexität dieser Arbeit auf. Hierbei wird das Zusammenspiel beider Steuerungsarten, also der direkten Steuerung und der, des Abrufens der Behavior, ersichtlich. Ein solches Szenario könnte durchaus in der realen Welt eingesetzt werden und verdeutlicht die Idee, beide Steuerungsarten zu vereinen. So macht es Sinn, den Roboter über Bewegungsmuster in den Supermarkt laufen zu lassen, dort jedoch über die direkte Kontrolle selbst die gewünschten Produkte in den Einkaufswagen zu legen.

5.4.1 Überblick über das Szenario

Der NAO läuft zu einem Einkaufsregal und bleibt davor stehen. Hier greift er nach den beiden gewünschten Produkten, dem Ball und einem Stift und läuft mit diesen zu einem Einkaufswagen, wo er diese hinein legt. Dann positioniert es sich vor dem Einkaufswagen, so dass er diesen greifen kann. Nachdem er den Einkaufswagen gegriffen hat, schiebt er diesen davon.

5.4.2 Ablauf des Szenarios

Das folgende Szenario wird ebenfalls mit einer Videokamera zusammenhängend aufgezeichnet. Dies hat zur Folge, dass keine Unterbrechung während der Aufnahme stattfindet. Die Abbildungen zeigen Bildausschnitte aus dem Video. Das Szenario verläuft nicht, wie anfangs geplant, zeigt aber dadurch, wie interagiert wird, um dennoch das Ziel zu erreichen. Die in Klammer stehenden Zahlen zeigen, welche Zeile in den beiden Abbildungen 5.16 und 5.18 im Text beschrieben wird. Der Ablauf des Szenarios ist in den Abbildungen 5.15 und 5.17 visualisiert.

Der NAO läuft in die Szene und positioniert sich vor dem Regal. Hier wird nun vom behavior control mode in den direct control mode umgeschaltet, sodass die Armbewegungen des Benutzers direkt an den NAO übertragen und ausgeführt werden (1). Dann wird versucht, die Arme so zu positionieren, dass der Ball gegriffen werden kann. Da die Perspektive des Benutzers ungünstig ist, besteht keine direkte Sicht zum zu greifenden Objekt (2). Dies resultiert in zwei Fehlversuchen, den Ball zu greifen, da jedes Mal daneben gegriffen wird. Wie bereits im Abschnitt 5.1.1 die Erkenntnis gewonnen wurde, ist blindes steuern des NAOs momentan nicht möglich. Dies wird an diesem konkreten Beispiel nochmals deutlich. Um aber den Einkauf fortzusetzen, wird auf den Ball verzichtet und nach dem nächsten Objekt gegriffen. Dabei handelt es sich um den Stift, der sich nach dem ersten Versuch, diesen zu greifen, fest in der Hand des NAOs befindet (3). Anschließend wird der NAO wieder in den behavior control mode versetzt, um ihn zum Einkaufswagen zu steuern (4). Dort angekommen wird im direct control mode die Hand gehoben, um den Stift in den Einkaufswagen zu legen (5). Nach Umschalten in den behavior control mode wird der NAO vor dem Einkaufswagen so bewegt, dass das Behavior „get the shopping cart“ erfolgreich ausgeführt werden kann (6). Das Resultat des Behaviors ist nicht das gewünschte, da der NAO den Einkaufswagen nur mit der rechten, statt mit beiden Händen festhält. Der Benutzer versucht den NAO vorwärts laufen zu lassen. Durch das einarmige Greifen dreht sich der Einkaufswagen jedoch, sodass ein weiterschieben nicht sinnvoll erscheint (7). Hier wird wieder auf den direct control mode zurückgegriffen, dank dessen spontan interagiert werden kann. Der Benutzer bewegt seine Arme so, dass der NAO den Einkaufswagen wieder gerade rückt (8). Nach vier kleinen Schritten steht dieser direkt hinter dem Einkaufswagen. Anschließend wird das „get the shopping cart“ Behavior abermals aufgerufen, was bei diesem Versuch zum erfolgreichen Greifen führt. Der NAO hält den Einkaufswagen mit beiden Händen fest (9). Über das Behavior „move ahead“ schiebt der NAO den Einkaufswagen mit dem darin platzierten Stift davon (10).

Das Szenario zeigt eindrücklich die Komplexität der zusammengesetzten Teilszenarien und die geringe Wahrscheinlichkeit, dass das Szenario von Anfang bis zum Ende reibungslos abläuft, da eben doch immer unerwartete Ereignisse eintreten können. Durch die Verdeckung des Balls mit der Hand ist es dem Benutzer nicht gelungen, zu sehen, ob die Hand bereits korrekt positioniert ist, oder sich neben dem Ball befindet. Hier zeigt sich die Stärke des intuitiven und spontanen Ansatzes des direct control mode. Dieser ermöglicht dem Benutzer sich spontan gegen den Kauf des Balls zu entscheiden und den Einkauf fortzusetzen. Durch das erfolgreiche Greifen des Stifts im ersten Versuch zeigt sich, dass die Problematik nicht an einer ungenauen Erkennung der menschlichen Bewegung liegt. Das Scheitern, den Ball zu greifen, liegt stattdessen an der Verdeckung des Balls und der somit verbundenen blinden Steuerung des NAOs.

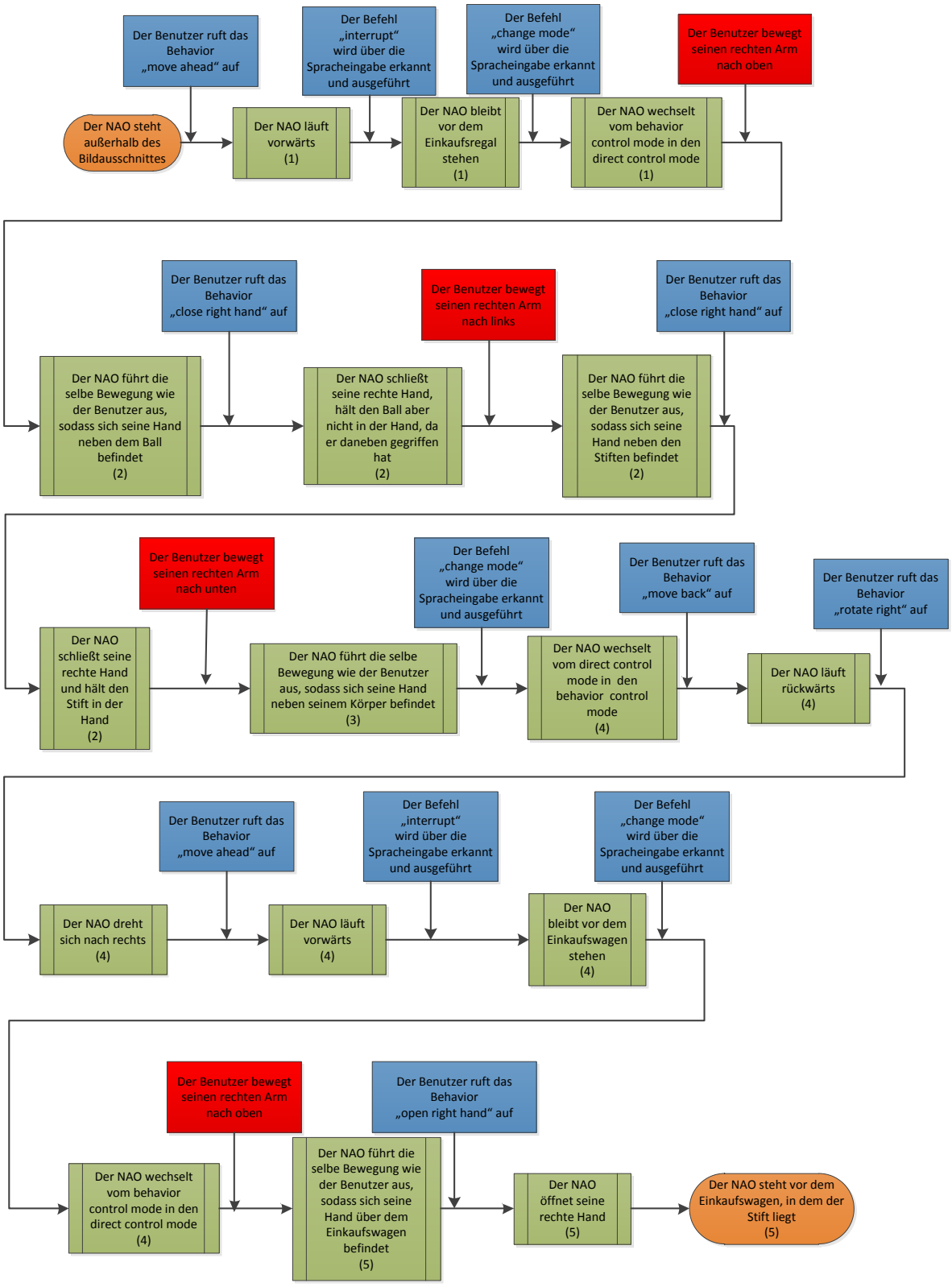

Abbildung 5.15: Visualisierung der Abbildung 5.16

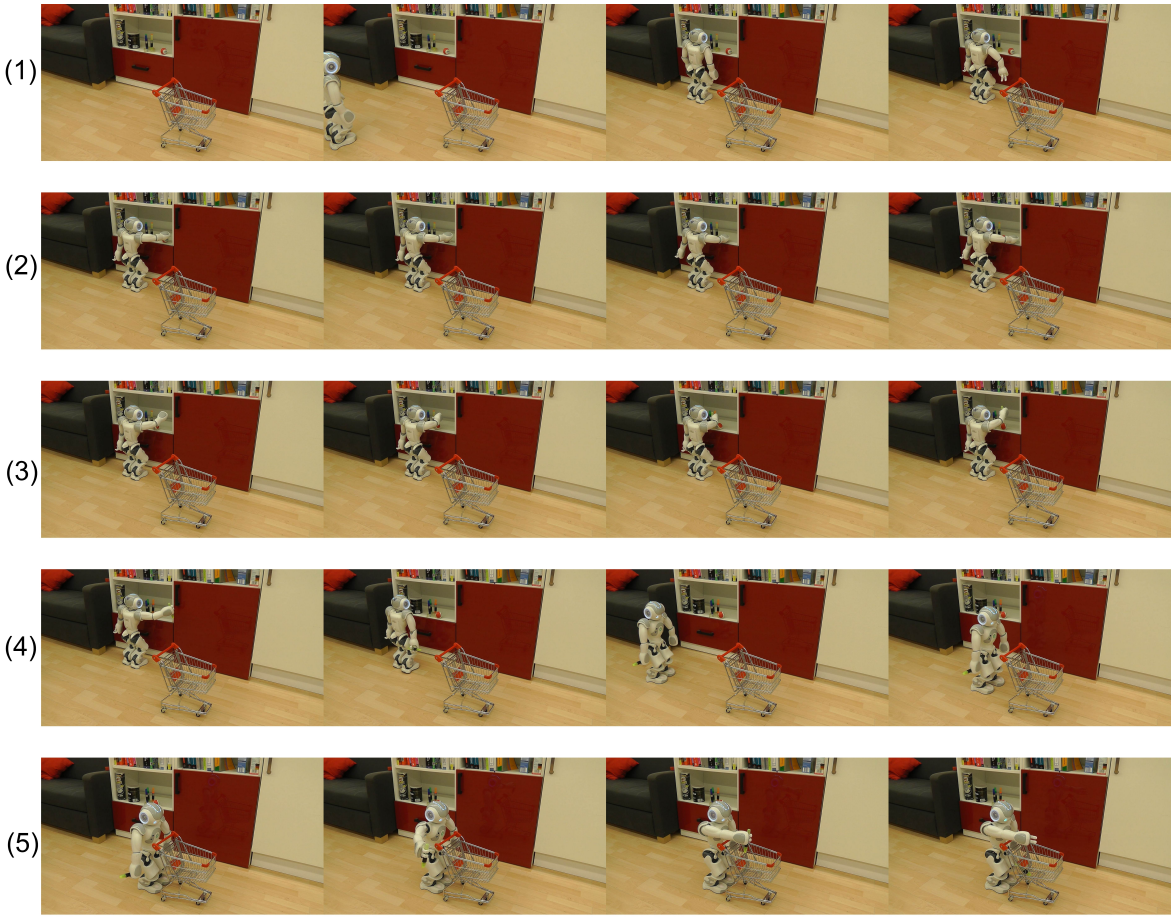

Abbildung 5.16: Bilderserie für das Gesamtszenario, Teil I. Das Szenario fängt links oben in dieser Abbildung an und endet rechts unten in der darauf folgenden Abbildung.

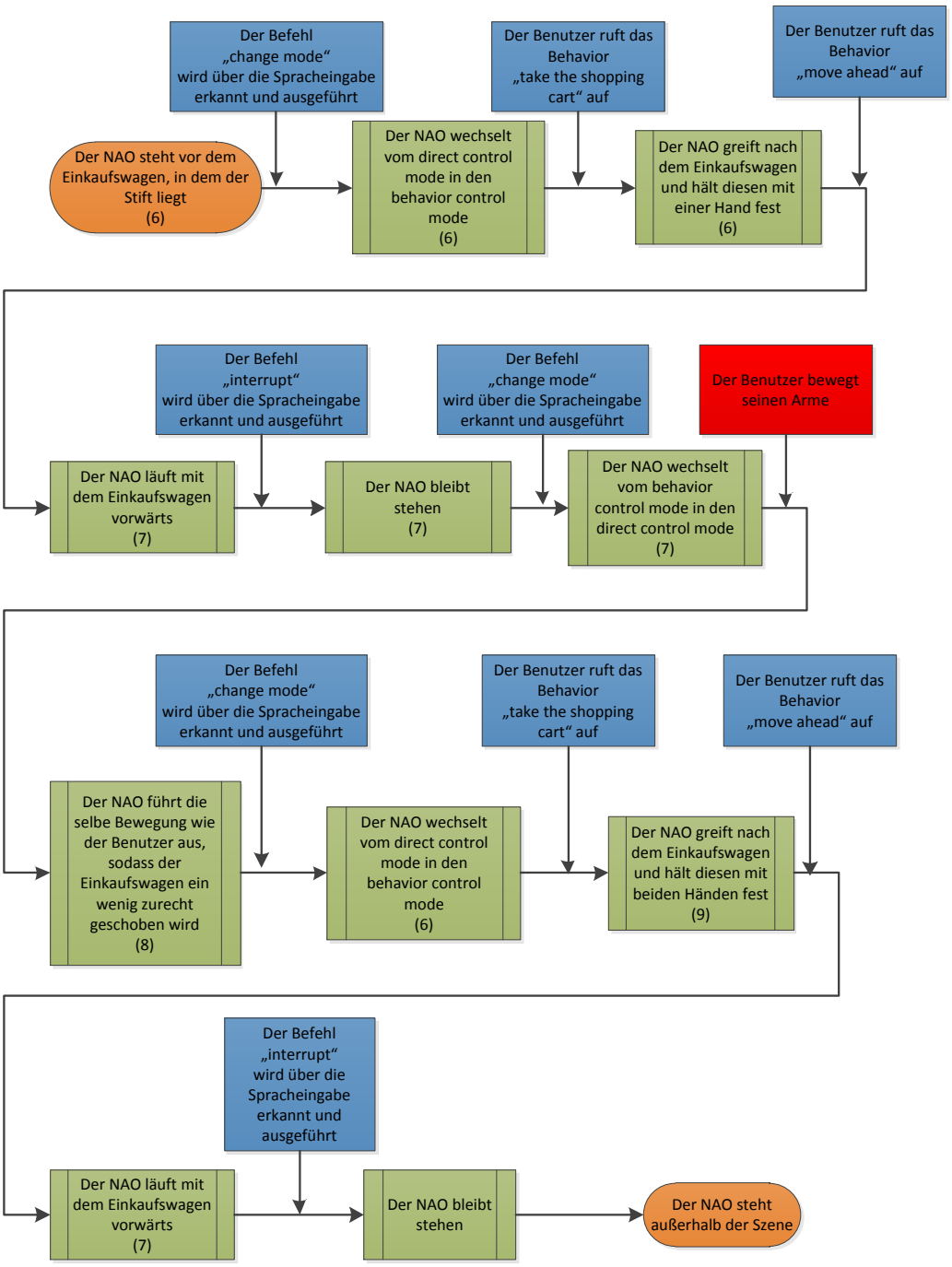

Abbildung 5.17: Visualisierung der Abbildung 5.18

Abbildung 5.18: Bilderserie für das Gesamtszenario, Teil II. Das Szenario fängt links oben in der vorhergehenden Abbildung an und endet in dieser Abbildung rechts unten.

Kapitel 6

Zusammenfassung und Ausblick

6.1 Zusammenfassung

Der in dieser Arbeit vorgestellte Ansatz ist, wie in Kapitel 5 aufgezeigt, in einfachen alltäglichen Szenarien einsetzbar. Die Erkennung der menschlichen Bewegung geschieht mittels des Spielesensors Kinect Kamera von Microsoft. Das von derselben Firma angebotene Framework Kinect SDK beta bietet ein robustes Skeleton Tracking an, das dreidimensionale Punkte des Menschen liefert. Dadurch ist es möglich, die Gelenkwinkel der Gliedmaßen zu berechnen. Diese Werte können dann an die entsprechenden Gelenkmotoren des Roboters gesendet werden, was zu der selben wie der vom Menschen vorgemachten Bewegung führt.

Da nicht alle Bewegungen 1:1 auf den Roboter übertragbar sind, hat dieser doch gänzlich andere Proportionen und instabile Posen, wird dieses Fehlen durch vorprogrammierte Bewegungsmuster (Behavior) kompensiert. Da der Programmieraufwand der Behavior variabel und durchaus aufwendig sein kann, werden derzeitig nicht alle möglichen Bewegungsabläufe wie beispielsweise dem Treppen steigen abgedeckt.

Es wurde gezeigt, dass es sinnvoll ist, beide Techniken, die direkte 1:1 Ausführung von Bewegungen und das Abrufen der Behavior, zu vereinen, da beide Stärken und Schwächen aufweisen, die sich gegenseitig ausgleichen können. So sind momentan nicht erfasste Bewegungen dennoch über Behavior realisierbar. Um intuitiv und spontan interagieren und den Programmieraufwand für die Behavior reduzieren zu können, kann auf die direkte Übertragung von Bewegungen zurückgegriffen werden.

Die Ergebnisse verdeutlichen die Stärke dieses Ansatzes. So ist es durchaus möglich, ein Szenario, das spontane Interaktionen fordert, erfolgreich zu absolvieren. Es hat sich erwiesen, dass die Bedienung über die Spracherkennung zur Auswahl des auszuführenden Behaviors spontan und natürlich efolgen kann. Auch wenn fehlerhafte Befehle erkannt werden ist es durch den Einsatz beider Techniken möglich, den NAO auf eine intuitive und einfache Weise zu bedienen.

6.2 Ausblick

Wie bereits in Kapitel 2 erwähnt, existieren aktuell eine Vielzahl von Ansätzen für einzelne Bewegungsmuster, wie beispielsweise dem Treppen steigen [OGHB11]. Diese können allesamt problemlos in die vorliegende Arbeit integriert werden, was dadurch die Leistungsfähigkeit des Roboters steigern würde.

Eine neue Winkelberechnung, die einen Vollwinkel (also einen Winkel zwischen 0° und 360°) zurückliefert, könnte die Freiheitsgrade des Roboters vollständig ausnutzen. Hierdurch könnten noch

größere Bewegungsfreiheiten bei der direkten Steuerung ermöglicht werden.

Eine erhebliche Erweiterung würde eine Eigenkollisionsberechnung mit sich bringen. Damit wäre es unter anderem nicht möglich, durch Bewegungen den Roboter in Eigenkollision zu bringen und diesen dadurch zu beschädigen. Dies würde zu einer größeren Einflussnahme des Roboters auf Entscheidungen führen, ob Bewegungsausführungen zulässig sind und dementsprechend ausgeführt werden oder nicht.

Durch den Einsatz eines Richtmikrofons wäre es möglich, die fehlerhafte Erkennung von Umgebungsgeräuschen als Befehlseingabe zu minimieren. Hier kann auch überlegt werden, ob ein Headset Sinn machen würde. Durch ein Headset könnte die örtliche Gebundenheit noch weiter aufgelöst werden, da man sich nicht mehr in Hörreichweite des Mikrofons sondern nur noch in Hör- und Signalreichweite des Headsets befinden müsste.

Eine Möglichkeit, Behavior zu erstellen, ist das Erlernen beziehungsweise Antrainieren eines Behaviors. So ist es denkbar, dass die direkte 1:1 Bewegung dazu genutzt wird zu gewissen Zeitpunkten die Gelenkwinkel auf einer Zeitachse abzuspeichern Anschließend wäre das so erstellte Behavior einsetzbar.

Abbildungsverzeichnis

1.1 Grobüberblick über die Aufgabe 1

2.1 Humanoider Roboter NAO der Firma Aldebaran Robotics 3

3.1 Komponenten des Kinect Sensors. 7
3.2 Überblick über die von der Kinect gelieferten Bilddaten 9
3.3 Überblick über die Gelenke des NAOs 10

4.1 Überblick über die Gesamtaufgabe 13
4.2 Bilderserie einer 1:1 Übertragung. 16
4.3 Ablauf bei neuen Winkel Daten 19
4.4 Ablauf bei neuen Behavior Daten 20
4.5 Überblick über vom Framework erkannte Gelenkpunkte. 22
4.6 Aufgestellte Vektoren am linken Ellenbogen 23
4.7 Eingeschlossener Winkel β zwischen Vektoren $\vec{BA}$ und $\vec{BC}$ 23
4.8 Komponentendiagramm des ursprünglichen Softwareentwurfs. 27
4.9 Komponentendiagramm des realisierten Softwareentwurfs. 28

5.1 Manuell gesetzte Vektoren in RGB Bild 31
5.2 Messwerte der Winkel während einer starren Pose. 32
5.3 Messwerte der Winkel während einer Bewegung. 32
5.4 Bedeutung der Symbole in den Visualisierungen der Abläufe 36
5.5 Bilderserie für das Teilszenario „Ansteuern von Ziel Positionen“ 37
5.6 Visualisierung des Ablaufs der Abbildung 5.5. 38
5.7 Visualisierung des Ablaufs der Abbildung 5.8. 39
5.8 Bilderserie für das Teilszenario „Greifen des Einkaufwagens“. 39
5.9 Visualisierung des Ablaufs der Abbildung 5.10. 40
5.10 Bilderserie für das Teilszenario „Laufen mit dem Einkaufswagen“. 40
5.11 Visualisierung des Ablaufs der Abbildung 5.12 41
5.12 Bilderserie für das Teilszenario „Greifen und ablegen eines Objekts“. 42
5.13 Visualisierung des Ablaufs der Abbildung 5.14 43
5.14 Bilderserie für das Teilszenario „Greifen und ablegen eines Objekts“. 43
5.15 Visualisierung der Abbildung 5.16 46
5.16 Bilderserie für das Gesamtszenario, Teil I. 47
5.17 Visualisierung der Abbildung 5.18 48
5.18 Bilderserie für das Gesamtszenario, Teil II. 49

Tabellenverzeichnis

3.1 Übersicht über Winkel des NAOs 11

4.1 Identifizieren der zu untersuchenden Teilbereiche 13
4.2 Features der Frameworks 18
4.3 Auflistung der Behavior und ihre Eigenschaft 21
4.4 Mögliche Wertebereiche der Winkel 24

5.1 Auflistung der zu erkennenden und erkannten Befehle 33
5.2 Auflistung der zu erkennenden und erkannten Befehle bei variierenden Befehlen . . 34
5.3 Auflistung der fehlerhaft erkannten Befehle bei einem Test über fünf Minuten 35

Literaturverzeichnis

[Bir10] A. Birk. The True Spirit of RoboCup. *Robotics Automation Magazine, IEEE*, 17(4):108, 2010.

[DSRI06] S. Dégallier, C.P. Santos, L. Righetti, and A. Ijspeert. Movement generation using dynamical systems : a humanoid robot performing a drumming task. In *International Conference on Humanoid Robots, IEEE-RAS*, pages 512 –517, 2006.

[Gro] Vitopsy Forensic Research Group. Touch free navigation through radiological images. Verfügbar unter: http://www.virtopsy.com/. [Stand 08. November 2011].

[Hon] Honda. Humanoider Roboter ASIMO. Verfügbar unter: http://world.honda.com/index.html. [Stand 08. November 2011].

[Int] Omek Interactive. Omek Beckon SDK. Verfügbar unter: http://www.omekinteractive.com/products.html. [Stand 08. November 2011].

[KFH+08] C.C. Kemp, P. Fitzpatick, H. Hirukawa, K. Yokoi, K. Harada, and Y. Matsumoto. *Springer Handbook of Robotics*, chapter 56. Humanoids, pages 1307–1333. Springer, 2008.

[Lab] Code Laboratories. CL-NUI-Platform-1.0.0.1210. Verfügbar unter: http://codelaboratories.com/downloads/. [Stand 7. September 2011].

[LLCH10] Y. Lu, L. Liu, S. Chen, and Q. Huang. Voice Based Control for Humanoid Teleoperation. In *International Conference on Intelligent System Design and Engineering Application (ISDEA)*, volume 2, pages 814 –818, 2010.

[MBS11] D. Maier, M. Bennewitz, and C. Stachniss. Self-supervised Obstacle Detection for Humanoid Navigation Using Monocular Vision and Sparse Laser Data. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1263 –1269, 2011.

[Mica] Microsoft. Kinect for Windows SDK v1.0 Beta 2 - Overview. Verfügbar unter: http://www.microsoft.com/download/en/details.aspx?id=27876. [Stand 26. November 2011].

[Micb] Microsoft. Microsoft Eula. Verfügbar unter: http://kinectforwindows.org/download/EULA.htm. [Stand 22. November 2011].

[Micc] Microsoft. Microsoft Kinect beta SDK KinectSDK32 v1.00.12. Verfügbar unter: http://research.microsoft.com/en-us/um/redmond/projects/kinectsdk/download.aspx. [Stand 7. September 2011].

[Micd] Microsoft. Microsoft Programming Guide. Verfügbar unter: http://research.microsoft.com/en-us/um/redmond/projects/kinectsdk/docs/ProgrammingGuide_KinectSDK.pdf. [Stand 25. Oktober 2011].

[Mice] Microsoft. Xbox Support Webseite. Verfügbar unter: http://support.xbox.com/de-DE/kinect/more-topics/kinect-sensor-components. [Stand 25. Oktober 2011].

[Nat] NaturalPoint. Arena. Verfügbar unter: http://www.naturalpoint.com/optitrack/products/arena/. [Stand 08. November 2011].

[NNYI04] S. Nakaoka, A. Nakazawa, K. Yokoi, and K. Ikeuchi. Leg Motion Primitives for a Dancing Humanoid Robot. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 610 – 615, 2004.

[NYK+05] E.S. Neo, K. Yokoi, S. Kajita, F. Kanehiro, and K. Tanie. A Switching Command-Based Whole-Body Operation Method for Humanoid Robots. *Transactions on Mechatronics, IEEE/ASME*, 10(5):546 – 559, 2005.

[OGHB11] S. Osswald, A. Gorog, A. Hornung, and M. Bennewitz. Autonomous Climbing of Spiral Staircases with Humanoids. In *International Conference on Intelligent Robots and Systems (IROS), IEEE/RSJ*, pages 4844 –4849, 2011.

[Ome] Omek. *Omek Beckon Development Suite - Datasheet*. Verfügbar unter: http://www.omekinteractive.com/content/Datasheet-Omek-BeckonDevelopmentSuite.pdf. [Stand 15. November 2011].

[Opea] OpenKinect. OpenKinect-libfreenect-v0.1.1-0-gdbfd4ce. Verfügbar unter: https://github.com/OpenKinect/libfreenect. [Stand 7. September 2011].

[Opeb] OpenNI. NITE-Win32-1.4.1.2-Dev. Verfügbar unter: http://www.openni.org/Downloads/OpenNIModules.aspx OpenNI Compliant Middleware Binaries, Stable, X86. [Stand 7. September 2011].

[Opec] OpenNI. OpenNI-Win32-1.3.2.1-Dev. Verfügbar unter: http://www.openni.org/Downloads/OpenNIModules.aspx OpenNI Binaries, Stable, X86. [Stand 7. September 2011].

[PHRA02] N.S. Pollard, J.K. Hodgins, M.J. Riley, and C.G. Atkeson. Adapting Human Motion for the Control of a Humanoid Robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1390 – 1397, 2002.

[PMD] PMDTechnologies. PMD[vision] CamCube. Verfügbar unter: http://www.pmdtec.com/products-services/pmdvisionr-cameras/pmdvisionr-camcube-bundle/. [Stand 08. November 2011].

[PMGM09] E. Pot, J. Monceaux, R. Gelin, and B. Maisonnier. Choregraphe: a Graphical Tool for Humanoid Robot Programming. In *The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pages 46 –51, 2009.

[Roba] Aldebaran Robotics. Datasheet NAO H21. Verfügbar unter: http://aldebaran-robotics.com/Downloads/Download-document/104-NAO-H21-datasheet.html. [Stand 26. Oktober 2011].

[Robb] Aldebaran Robotics. Kinematics 3.2 - Joints information. Verfügbar unter: http://users.aldebaran-robotics.com/docs/site_en/reddoc/hardware/joints-names.html. [Stand 26. Oktober 2011].

[Robc] Aldebaran Robotics. *NAO Academics Edition - Datasheet*. Verfügbar unter: http://aldebaran-robotics.com/Downloads/Download-document/99-NAO-Academics-Datasheet.html. [Stand 15. November 2011].

[Robd] Aldebaran Robotics. *NAO SOFTWARE SUITE - Naoqi*. Verfügbar unter: http://www.aldebaran-robotics.com/Downloads/Download-document/109-NAOqi-SDK-datasheet.html. [Stand 15. November 2011].

[Sco03] R. Scott. Sparking Life Notes on the Performance Capture Sessions for The Lord of The Rings: The Two Towers. *ACM SIGGRAPH Computer Graphics*, 37(4):17–21, 2003.

[SKM+] E. Steinbach, M. Kranz, W. Maier, F. Schweiger, N. Alt, A. Al-Nuaimi, and C. Schuwerk (Hrsg.). ADVANCES IN MEDIA TECHNOLOGY - VISUELLE UMGEBUNGSMODELLIERUNG FÜR ROBOTIK- UND SPIELEANWENDUNGEN. Verfügbar unter: http://www.lmt.ei.tum.de/courses/hsmt/proceedings/aimt_ss11.pdf. [Stand 24. November 2011].

[SN04] R. Siegwart and I.R. Nourbakhsh. *Introduction to Autonomous Mobile Robots*. The MIT Press, 2004.

[SNK08] M. Stilman, K. Nishiwaki, and S. Kagami. Humanoid Teleoperation for Whole Body Manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3175 –3180, 2008.

[Tan] Stewart Tansley. Microsoft Kinect beta SDK Community Update and Next Steps. Verfügbar unter: http://research.microsoft.com/en-us/events/fs2011/gupta_kinect4_windows.pdf. [Stand 25.Oktober 2011].

[Xse] Xsens. MVN - Inertial Motion Capture. Verfügbar unter: http://www.xsens.com/en/general/mvn. [Stand 08. November 2011].

[Youa] Youtube-Channel: hbenersuay. Humanoid Robot Control and Interaction using Depth Camera. Verfügbar unter: http://www.youtube.com/watch?v=GdepIXZTJsw. [Stand 24. November 2011].

[Youb] Youtube-Channel: HumanoidsFreiburg. Whole-body Imitation of Human Motions with a Nao Humanoid: Real-time teleoperation. Verfügbar unter: http://www.youtube.com/watch?v=dC16A6u8WA8. [Stand 24. November 2011].

[Youc] Youtube-Channel: taylorveltrop. Improved Humanoid Robot Teleoperation with NAO and Kinect. Verfügbar unter: http://www.youtube.com/watch?v=TmTW61MLm68. [Stand 24. November 2011].

[ZLP99] Y. Zhuang, X. Liu, and Y. Pan. Video Motion Capture Using Feature Tracking and Skeleton Reconstruction. In *International Conference on Image Processing (ICIP)*, volume 4, pages 232 –236, 1999.

[ZMLA11] J.M.I. Zannatha, L.E.F. Medina, R.C. Limón, and P. M. Álvarez. Behavior Control for a Humanoid Soccer Player using Webots. In *21st International Conference on Electrical Communications and Computers (CONIELECOMP)*, pages 164–170. IEEE, 2011.

[ZSMG07] Z. Zalevsky, A. Shpunt, A. Maizels, and J. Garcia. Method and System for Object Reconstruction. Verfügbar unter: http://www.wipo.int/patentscope/search/en/WO2007043036, April 2007. [Stand 24. November 2011].