**Koppelung einer Bewegungsentwicklung**

**für humanoide Roboter**

**mit der Bewegungserkennung**

**von Personen durch die Kinect**

**Studienarbeit**

für die Prüfung zum

Bachelor of Engineering

des Studienganges Informatik

Studienrichtung Informationstechnik

an der Dualen Hochschule Baden-Württemberg Karlsruhe

von

**Dennis Alles**

und

**Karolin Edigkaufer**

12.05.2014

Matrikelnummer 3934520 (Dennis Alles)

6589515 (Karolin Edigkaufer)

Kurs TINF11B3

Betreuer Prof. Dr. Hans-Jörg Haubner

Michael Schneider

Ehrenwörtliche Erklärung

gemäß § 5 (3) der „Studien- und Prüfungsordnung DHBW Technik“ vom 22. September 2011.

Ich habe die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Ort, Datum Unterschrift

Ort, Datum Unterschrift

Abbildungsverzeichnis

[Abb. 1: Die Sensoren der Kinect [6] 4](#_Toc382302175)

[Abb. 2: Skeleton mit 20 Punkten [7] 5](#_Toc382302176)

[Abb. 3: Die Lage der Motoren beim Nao [10] 7](#_Toc382302177)

[Abb. 4: Die Hardwarekomponenten des Nao [11] 8](#_Toc382302178)

[Abb. 5: Nao Bibliothek nutzen 9](#_Toc382302179)

Tabellenverzeichnis

[Code 1: Installation des Puppetmasters 18](#_Toc366502778)

[Code 2: Installation Foreman 19](#_Toc366502779)

[Code 3: Manifest für den Knoten KASTILXTEST03 22](#_Toc366502780)

Abkürzungsverzeichnis

DHBW Duale Hochschule Baden-Würrtemberg

SDK Software Development Kit

VGA Video Graphics Array

Inhaltsverzeichnis

[Ehrenwörtliche Erklärung II](#_Toc381796082)

[Abbildungsverzeichnis III](#_Toc381796083)

[Tabellenverzeichnis IV](#_Toc381796084)

[Abkürzungsverzeichnis V](#_Toc381796085)

[Inhaltsverzeichnis VI](#_Toc381796086)

[1 Einleitung 1](#_Toc381796087)

[1.1 Motivation 2](#_Toc381796088)

[1.2 Aufgabenumfeld 2](#_Toc381796089)

[1.3 Aufgabenbeschreibung und Ziele 2](#_Toc381796090)

[1.4 Vorgehensweise 2](#_Toc381796091)

[2 Grundlagen 3](#_Toc381796092)

[2.1 Kinect 3](#_Toc381796093)

[2.1.1 Sensoren der Kinect 4](#_Toc381796094)

[2.1.2 SDK 6](#_Toc381796095)

[2.1.3 Direkte Möglichkeit mit der SDK-Nutzung 6](#_Toc381796096)

[2.2 Humanoide Roboter 6](#_Toc381796097)

[2.2.1 Nao 6](#_Toc381796098)

[2.3 Programmiersprache 7](#_Toc381796099)

[3 Konzeption/Planung 7](#_Toc381796100)

[4 Umsetzung 8](#_Toc381796101)

[4.1 Nao 8](#_Toc381796102)

[4.2 Kinect 8](#_Toc381796103)

[4.3 Programm 8](#_Toc381796104)

[5 Fazit 8](#_Toc381796105)

[Literaturverzeichnis IX](#_Toc381796106)

# Einleitung

Sowohl Roboter als auch die direkte Interaktion zwischen Anwender und Computer liegen voll im Trend in der Informatik. Die starre Programmierung eines Rechners wird es zwar immer geben, aber oftmals rückt das Thema den Anwender selbst mit einzubeziehen mehr und mehr in den Vordergrund. Hierbei liegt die große Herausforderung in der Wahrnehmung und Nachahmung von menschlichen Aktionen [1].

Die Studienarbeit beschäftigt sich mit dem System Kinect von Microsoft und einem humaoiden Roboter Nao. Die Kinect ist eine Hardware zur Gestik- und Mimik Erkennung.

ROBOTER NAO…

## Motivation

Die fehlt derzeit noch…

## Aufgabenumfeld

Diese Studienarbeit entsteht an der Dualen Hochschule Baden Württemberg (DHBW) und wird von zwei Studenten gemeinsam betreut. Zur Verfügung steht ein Labor mit einem Nao und einem Rechner, auf dem alle relevanten Dokumente und Programme genutzt werden können. Zu dem können die Programme auch auf den privaten Rechnern installiert werden, so dass mit Hilfe von Simulationssoftware auch von zu Hause aus gearbeitet werden kann.

## Aufgabenbeschreibung und Ziele

Ziel der Arbeit ist es, eine Anwendung zu programmieren, die auf spielerische Weise die Zusammenarbeit zwischen dem Roboter, einem Spieler und der Kinect verknüpft.

So werden dem Roboter Nao verschiedene Bewegung beigebracht, die er auf Befehl automatisiert ausführen kann.

Die Kinect ist dafür zuständig, die Bewegungen des davorstehenden Spielers aufzunehmen.

Der Ablauf des programmierten Spiels kann in drei Schritte unterteilt werden. Im ersten Schritt führt Nao eine der programmierten Bewegungen vor. Sobald die Roboterbewegung vollendet ist, beginnt Schritt zwei, in dem die Kinect die Bewegung des Spielers aufnimmt. Gleichzeitig läuft eine festgelegte Zeit herunter. Die Aufgabe des Spielers ist es innerhalb der Zeit vom Roboter durchgeführte Bewegung möglich ähnlich nachzuahmen. Nach dem Ablauf der Zeit beginnt Schritt drei, bei dem die getätigte Benutzerbewegung mit der programmierten Roboterbewegung vom System verglichen wird.

Dieser Ablauf mit anschließender Auswertung wird in einem spielerischen Umfeld dargestellt und auf einem Monitor angezeigt.

## Vorgehensweise

Grobe Zusammenfassung sollten wir da schon schreiben.

# Grundlagen

Dieses Kapitel schildert die für das Verständnis dieser Arbeit notwendigen Grundlagen. Zu diesem Zweck wird in Kapitel 2.1 die „Kinect-Kamera“ und in Kapitel 2.2 der Roboter NAO vorgestellt.

## Kinect

Die Kinect ist eine Hardware aus dem Hause Microsoft in Zusammenarbeit mit dem Unternehmen PrimeSense. Sie wurde zunächst für die Steuerung der Spielekonsole Xbox360 entwickelt. Die Kinect setzt wie auch die Nintendo Wii oder Sonys Move auf den Körpereinsatz des Benutzers. Der große Unterschied liegt aber darin, dass der Benutzer keinen Controller in der Hand halten muss. Bei der Kinect wird der ganze Körper zum „Controller“. Die Kinect-Kamera ist eine Leiste aus vielen Sensoren (siehe Abbildung XXX) und erkennt so die Bewegungen der davorstehenden Personen [2], [3].

Nachdem die Kinect so erfolgreich für die Spielekonsole verkauft wurde, ist sie seit Februar 2012 auch für Windows erhältlich. Wie Microsoft selbst schreibt, erhält der Windows-Computer durch die Kinect Augen, Ohre und auch ein Gehirn. Zusätzlich zu der Hardware wird auch ein Software Development Kit (SDK) mitgeliefert, was nichts anderes ist als eine Sammlung von Werkzeugen und Anwendungen, um eine Software zu erstellen. Zusätzlich befindet sich noch eine Dokumentation im Angebot. Das Windows-SDK soll die Software-Entwicklung für Kinect-Anwendungen erleichtern [4], [5].

### Sensoren der Kinect

In der ca. 20 cm breiten Kinect sind ein Infrarot-Projektor, eine Infrarot-Kamera, eine Farb-Kamera sowie vier Mikrofone verbaut. Zwischen der Kameraleiste und dem Standfuß befindet sich außerdem ein Motor, über den die Sensoren-Leiste ausgerichtet werden kann.

Durch die beiden Infrarot-Komponenten werden die Tiefendaten, also ein 3D-Bild erzeugt. Hierzu wird von dem Infrarot-Projektor eine definierte Punkt-Matrix in den Raum gesendet. Die Infrarot-Kamera nimmt das Muster des Infrarot-Lichts wieder auf und gibt die Daten an den internen Chip weiter. Von ihm werden dann aus den Kameradaten die räumlichen Koordinaten ausgerechnet. Bei dem Projektor wurde auch darauf geachtet, dass die Laserstrahlen selbst bei einem Einfall ins Augeninnere des Benutzers nicht schädlich sind.

Die Farbkamera ist für die Aufnahme von 2D-Bildern zuständig. Die Kamera besitzt hierbei eine Video Graphics Array (VGA)-Auflösung von 640x480 Pixeln.

Die vier Mikrofone werden für eine Audioortung benutzt. Somit lässt sich ein Sprachbefehl eines Benutzers genau zuordnen. Durch die Mikrofone ist es auch möglich ein Xbox-Spiel per Sprache zu starten oder von vorne zu beginnen.



Abb. : Die Sensoren der Kinect [6]

Die Kamera ist mithilfe des Infrarot-Sensors in der Lage Menschen von leblosen Objekten zu unterscheiden. Es wird für jeden Spieler, der vor der Kinect steht, ein Skelett mit 20 Gelenken erstellt.



Abb. : Skeleton mit 20 Punkten [7]

Mithilfe dieser 20 Punkte können die Körperteile eines Spielers unterschieden und so Bewegungen verarbeitet werden. Außerdem ist es möglich die Gesichter von verschiedenen Personen zu unterscheiden und Aussage über die Mimik zu treffen.

<http://www.xboxaktuell.de/content,7,xbox_360_kinect_hintergruende_fakten.html>

<http://www.microsoft.com/en-us/kinectforwindows/discover/features.aspx>

<http://www.gamepro.de/xbox/spiele/xbox-360/kinect-adventures/artikel/kinect,46324,1967600.html>

<http://wiki.zimt.uni-siegen.de/fertigungsautomatisierung/index.php/Einsatzm%C3%B6glichkeiten_einer_3D-Kamera_in_der_Produktionstechnik_am_Beispiel_der_Kinect-Kamera>

<http://www.game7.de/kinect/news/technische-details-kinect-16238.php>

<http://www.generationrobots.com/de/401430-microsoft-kinect-sensor-microsoft.html>

<http://msdn.microsoft.com/en-us/magazine/jj851072.aspx>

### SDK

Vorgeschichte…

Es gibt zwei konkurrierende SDKs. (Vorgeschichte erläutern) Das eine stammt von Microsoft, das andere ist quelloffen. [Microsoft Kinect Seite 4]

### Direkte Möglichkeit mit der SDK-Nutzung

## Humanoide Roboter

Ein humanoider Roboter ist ein Roboter, der dem Mensch nachempfunden ist und dessen Bewegungen die eines Menschen ähneln. Genutzt werden humanoide Roboter zumeist in der Servicerobotik, zum Beispiel als Haushaltshilfe für häufig anfallende Aufgaben. Staubsaugen, Spülmaschine ausräumen oder Toilette putzen sind Aufgaben, die in Zukunft von Robotern ausgeführt werden könnten. Zahlreiche Unternehmen und Einrichtungen befassen sich derzeit mit der Entwicklung und dem Testen von humanoiden Robotern. Darunter sind auch namenhafte Firmen und Universitäten, wie das Karlsruher Institut für Technologie, das Frauenhofer Institut, Toyota und Google [7] [8] [9].

### Nao

Nao ist ein 58cm großer humanoider Roboter der Firma Aldeberan Robotics aus Frankreich. Die erste Version des Nao erschien .... to be continued.

#### Hardware

Der humanoide Roboter Nao hat einen integrierten Akku, welcher ca. eine Stunde genutzt werden kann und ca. fünf Stunden zum vollen Aufladen benötigt.

Er ist außerdem mit einem Intel „ATOM Z530“ Prozessor ausgestattet mit 512KB Cache Speicher und 1 GB RAM. Der eingebaute Flashspeicher hat Platz für 2GB Daten.

Zur Kommunikation zwischen Computer und Roboter verfügt Nao über Ethernet und WIFI.

Wie in zu sehen, hat Nao 14 eingebaute Motoren, um die Bewegungen so humanoid als nur möglich aussehen zu lassen. Sechs davon sind für Armbewegungen, sechs für Hüft- und Beinbewegungen und zwei für den Kopf zuständig. Dabei gibt es drei unterschiedliche Arten von Motoren.



Abb. : Die Lage der Motoren beim Nao [10]

Zu dem hat er mehrere Kameras, Mikrophone und sogar einen Lautsprecher, so dass er mit seiner Umwelt interagieren kann.

Neben den Aktoren hat der Nao auch einige Sensoren, die es ihm ermöglichen Hindernisse zu erkennen, die Winkeleinstellungen zu messen und vieles mehr.

Die Hardwarekomponenten sind wie folgt verteilt:

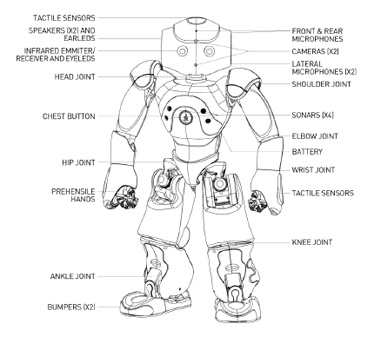


Abb. : Die Hardwarekomponenten des Nao [11]

#### Software

Auf dem Nao selbst läuft ein Linux Embedded Betriebssystem, welches auf Gentoo basiert. Nao ist ein 32-Bit System und kann direkt mit Python und C++ angesprochen werden. Über eine Programmierschnittstelle versteht Nao aber auch Java, MathLab und .NET Sprachen, wie C#. Choreographe ist die Software, die vom Hersteller mitgeliefert wird. Im Choreographen ist es möglich, vorgespeicherte Bewegungen auszuführen, Nao Texte sprechen zu lassen und gewisse Abläufe auf ihm zu speichern, so dass diese auch nach Neustarten des Roboters noch verfügbar sind.

#### Nao SDK

Der Roboter bringt seine eigene Bibliothek für andere Programmiersprachen mit, so dass man zum Beispiel auch mit Hilfe der Programmiersprache C# den Roboter relativ leicht ansteuern kann. Dies funktioniert in Visual Studio wie in Abb. 3 zu sehen. In diesem Beispiel wird der Nao den Text „Hello World from c sharp“ sprechen.



Abb. : Nao Bibliothek nutzen

#### NaoQi

#### Webots für Nao

Webots ist die Simulation von einem Nao. Bevor ein Stück Programmcode auf dem Nao ausgeführt wird, kann man sich so vergewissern, dass das Programm das tut, von dem man ausgeht und ob dies eventuell zu Problemen auf dem Nao führen kann. So kann verhindert werden, dass der Roboter versucht Bewegungen auszuführen, für die seine Motoren nicht geeignet sind oder ob der Roboter eventuell schaden nimmt, zum Beispiel durch Umfallen.

In der Simulationswelt ist es dabei auch möglich Hindernisse einzubauen, um so nicht nur die Motoren, sondern auch die Sensoren zu testen.

#### Sensoren

#### Aktoren

.

## Programmiersprache

Microsoft empfiehlt die Nutzung von C++, C# oder Visual Basic für die Kinect, da es für diese Programmiersprachen vorgefertigte Klassen im Software Development Kit (SDK) gibt, auf welche man zurückgreifen kann. Für den Roboter Nao kann man wie bereits beschrieben ebenfalls C# nutzen.

Da sowohl die Kinect, als auch der Nao mit C# arbeiten (können), ist auch die Anwendung in C# programmiert. So ist es möglich, dem Nao Daten zur Ausführung zu senden und hat diese dann im gleichen Format vorliegen, wie sie später von der Kinect aufgezeichnet und an die Anwendung gesendet werden. Es ist also keine Konventierung im Programm mehr nötig.

# Konzeption/Planung

Evtl Untergliederung in Kinect und Nao

Zu Beginn wird die Entwicklungsumgebung eingerichtet, inklusive der SDKs mit Kinect und Nao. Dann werden dem Nao einige Bewegungen einprogrammiert, dabei sollen auch die entsprechenden Winkel berechnet werden und an eine Klasse übermittelt werden. Die Kinect bringt einige fertige Programmierungen mit, unter anderem eine Programmierung, in der Skelette von Personen sowie deren Entfernung erkannt wird. Dieses Programm soll als Grundlage dienen, mit dem der Nutzer seine Bewegungen selbst sieht und das Programm die Winkel der Nutzer errechnen kann. Diese sollen dann mit den Werten des Nao verglichen werden. Es bleibt zu überlegen ob verschiedene Schwierigkeitsstufen darüber abgebildet werden, dass es unterschiedlich schwere Bewegungen zum Nachmachen gibt, die stetig komplizierter und/oder länger werden oder über die prozentuale Übereinstimmung der beiden Bewegungen bzw. derer Winkel. Dabei werden die Winkel nicht nur zu Beginn und Ende der Bewegung gemessen und verglichen, sondern auch an verschiedenen Zwischenpunkten.

# Umsetzung

## Nao

Hier Text Nao.

## Kinect

## Programm

## Test

Jede Bewegung wurde 20 mal vor der Kinect durchgeführt. Die beste Bewegung war das Heben der rechten Hand, wobei die Bewegung 18 mal als erfolgreich anerkannt wurde. Am schlechtesten wurde die Bewegung … mit 8 Bewegungen erkannt.

# Fazit

Das folgt am Ende.

Literaturverzeichnis

|  |  |
| --- | --- |
| [1] | Karlsruher Institut für Technologie, „Imitationslernen in der Robotik,“ 2012. [Online]. Available: http://www.hyperraum.tv/tag/autonome-roboter/. [Zugriff am 15 01 2014]. |
| [2] | Wikipedia, „Kinect,“ 10 01 2014. [Online]. Available: http://de.wikipedia.org/wiki/Kinect. [Zugriff am 15 01 2014]. |
| [3] | Microsoft, „Kinect für Windows,“ [Online]. Available: http://www.microsoft.com/en-us/kinectforwindows/. [Zugriff am 15 01 2014]. |
| [4] | Golem, „SDK - Software Development Kit,“ [Online]. Available: http://www.golem.de/specials/sdk/. [Zugriff am 15 01 2014]. |
| [5] | J. Roßberg, „Alle Informationen zur neuen Bewegungssteuerung,“ 01 11 2010. [Online]. Available: http://www.gamepro.de/xbox/spiele/xbox-360/kinect-adventures/artikel/kinect,46324,1967600.html. [Zugriff am 15 01 2014]. |
| [6] | ifixit.com, „Kinect ohne Gehäuse,“ 05 01 2012. [Online]. Available: http://wiki.zimt.uni-siegen.de/fertigungsautomatisierung/index.php/Datei:S910310\_abb2.png. [Zugriff am 10 03 2014]. |
| [7] | N. Demuth, „Die Welt,“ 27 08 2013. [Online]. Available: http://www.welt.de/wissenschaft/article119430009/Mit-solchen-Maschinen-werden-wir-alleine-sein.html. [Zugriff am 05 03 2014]. |
| [8] | Die Zeit, „zeit.de,“ 04 12 2013. [Online]. Available: http://www.zeit.de/wirtschaft/unternehmen/2013-12/google-roboter-android-erfinder-andy-rubin. [Zugriff am 05 03 2014]. |
| [9] | Frauenhofer Institut, „Frauenhofer Institut - Humaniode Roboter,“ [Online]. Available: https://www.iosb.fraunhofer.de/servlet/is/5093/. [Zugriff am 05 03 2014]. |
| [10] | Aldebaran Robotics, „Aldebaran Robotics Nao Software Documentation,“ [Online]. Available: https://community.aldebaran-robotics.com/doc/1-14/family/robots/motors\_robot.html. [Zugriff am 06 03 2014]. |
| [11] | Aldebaran Robotics, „Aldebaran Robotics,“ [Online]. Available: http://www.aldebaran-robotics.com/en/Discover-NAO/Key-Features/hardware-platform.html. [Zugriff am 06 03 2014]. |

Alle Abbildungen, Tabellen, o.ä., die keine Literaturangabe enthalten, sind eigene oder firmenintern verwendete Darstellungen.