Hochschule für Technik, Wirtschaft und Kultur Leipzig
Fakultät Informatik, Mathematik und Naturwissenschaften

Prüfungsprojekt für das Wahlpflichtmodul Robotik

Dokumentation des CM-BOT

B.Sc. Matthias Rick
B.Sc. Christof Pieloth

6. Dezember 2010

Inhaltsverzeichnis

1 Aufgabenstellung

Als Praxisaufgabe für das Wahlpflichtmodul Robotik an der HTWK Leipzig ist ein Hexapod zu entwickeln.

Für jedes Beinpaar dieses sechsbeinigen Roboters ist ein eigener Controller zu verwenden. Die Controller sollen über eine geeignete Kommunikationsstruktur Daten austauschen können. Außerdem ist die direkte und inverse Kinematik zu lösen. Unter Nutzung der zuvor erarbeiteten Erkenntnisse ist abschließend ein Laufalgorithmus zu implementieren.

2 Vorbetrachtung

TODO

- Projekt auf BerliOS
- Servos/ Robokit
- Controller
- Hardware
- Kommunikationsstruktur
- Servosanbindung

3 Hardware

- Servoanbindung
- Netzteil
- Kommunikationsstruktur
- Maße Chassis
- EAGLE-Schaltungen

4 Kinematik

4.1 Direkte Kinematik

4.1.1 Denavit-Hartenberg-Parameter

	G_1	G_2	G_3
l	-14	0	0
a	50	85	55
α	-90	0	0
ϑ	v	v	v

4.1.2 Denavit-Hartenberg-Matrix

Aus den Denavit-Hartenberg-Parametern ergeben sich die folgenden Matrizen:

$$D_{01} = \begin{pmatrix} \cos(\vartheta_1) & 0 & -\sin(\vartheta_1) & 50\cos(\vartheta_1) \\ \sin(\vartheta_1) & 0 & \cos(\vartheta_1) & 50\sin(\vartheta_1) \\ 0 & -1 & 0 & -14 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$D_{12} = \begin{pmatrix} \cos(\vartheta_2) & -\sin(\vartheta_2) & 0 & 85\cos(\vartheta_2) \\ \sin(\vartheta_2) & \cos(\vartheta_2) & 0 & 85\sin(\vartheta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$D_{12} = \begin{pmatrix} \cos(\vartheta_3) & -\sin(\vartheta_3) & 0 & 55\cos(\vartheta_3) \\ \sin(\vartheta_3) & \cos(\vartheta_3) & 0 & 55\sin(\vartheta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$D_{03} = \begin{pmatrix} c(\vartheta_1)\ c(\vartheta_2)\ c(\vartheta_3) - c(\vartheta_1)\ s(\vartheta_2)\ s(\vartheta_3) & -c(\vartheta_1)\ c(\vartheta_2)\ s(\vartheta_3) - c(\vartheta_1)\ c(\vartheta_3)\ s(\vartheta_2) & -s(\vartheta_1) & 50\,c(\vartheta_1) + 85\,c(\vartheta_1)\ c(\vartheta_2) - 55\,c(\vartheta_1)\ s(\vartheta_2)\ s(\vartheta_3) + 55\,c(\vartheta_1)\ c(\vartheta_2)\ c(\vartheta_3) \\ c(\vartheta_2)\ c(\vartheta_3)\ s(\vartheta_1) - s(\vartheta_1)\ s(\vartheta_2)\ s(\vartheta_3) & -c(\vartheta_2)\ s(\vartheta_1)\ s(\vartheta_3) - c(\vartheta_3)\ s(\vartheta_1)\ s(\vartheta_2) & c(\vartheta_1) & 50\,s(\vartheta_1) + 85\,c(\vartheta_2)\ s(\vartheta_1) - 55\,s(\vartheta_1)\ s(\vartheta_2)\ s(\vartheta_3) + 55\,c(\vartheta_2)\ c(\vartheta_3)\ s(\vartheta_1) \\ -c(\vartheta_2)\ s(\vartheta_3) - c(\vartheta_3)\ s(\vartheta_2) & s(\vartheta_2)\ s(\vartheta_3) - c(\vartheta_2)\ c(\vartheta_3) & 0 & -85\,s(\vartheta_2) - 55\,c(\vartheta_2)\ s(\vartheta_3) - 55\,c(\vartheta_3)\ s(\vartheta_2) - 14 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$x_{ex} = f_{11} = \cos(\vartheta_1)\ \cos(\vartheta_2)\ \cos(\vartheta_3) - \cos(\vartheta_1)\ \sin(\vartheta_2)\ \sin(\vartheta_3)$$

$$y_{ex} = f_{12} = -\cos(\vartheta_1)\ \cos(\vartheta_2)\ \sin(\vartheta_3) - \cos(\vartheta_1)\ \cos(\vartheta_3)\ \sin(\vartheta_2)$$

$$z_{ex} = f_{13} = -\sin(\vartheta_1)$$

$$p_{ex} = f_{13} = 50\ \cos(\vartheta_1) + 85\ \cos(\vartheta_1)\ \cos(\vartheta_2) - 55\ \cos(\vartheta_1)\ \sin(\vartheta_2)\ \sin(\vartheta_3) + 55\ \cos(\vartheta_1)\ \cos(\vartheta_2)\ \cos(\vartheta_3)$$

$$x_{ey} = f_{21} = \cos(\vartheta_2)\ \cos(\vartheta_3)\ \sin(\vartheta_1) - \sin(\vartheta_1)\ \sin(\vartheta_2)\ \sin(\vartheta_3)$$

$$y_{ey} = f_{22} = -\cos(\vartheta_2)\ \sin(\vartheta_1)\ \sin(\vartheta_3) - \cos(\vartheta_3)\ \sin(\vartheta_1)\ \sin(\vartheta_2)$$

$$z_{ey} = f_{23} = \cos(\vartheta_1)$$

$$p_{ey} = f_{23} = 50\ \sin(\vartheta_1) + 85\ \cos(\vartheta_2)\ \sin(\vartheta_1) - 55\ \sin(\vartheta_1)\ \sin(\vartheta_2)\ \sin(\vartheta_3) + 55\ \cos(\vartheta_2)\ \cos(\vartheta_3)\ \sin(\vartheta_1)$$

$$x_{ez} = f_{31} = -\cos(\vartheta_2)\ \sin(\vartheta_3) - \cos(\vartheta_3)\ \sin(\vartheta_2)$$

$$y_{ez} = f_{32} = \sin(\vartheta_2)\ \sin(\vartheta_3) - \cos(\vartheta_2)\ \cos(\vartheta_3)$$

$$z_{ez} = f_{33} = 0$$

$$p_{ez} = f_{33} = -85\ \sin(\vartheta_2) - 55\ \cos(\vartheta_2)\ \sin(\vartheta_3) - 55\ \cos(\vartheta_3)\ \sin(\vartheta_2) - 14$$

4.2 Inverse Kinematik

Die Lösung des inversen kinematischen Problems wird durch ein geometrisches Verfahren berechnet. Durch Abbildung des 3-dimensionalen Koordinatensystems auf zwei 2-dimensionale Koordinatensyteme können die Gelenkwinkel mit Hilfe der Trigonometrie berechnet werden.

In den folgenden Formeln und Erläuterungen werden diese Bezeichnungen verwendet:

- $H \triangleq$ Hüftgelenk, $\Theta_1 \triangleq$ Winkel des Hüftgelenks
- $K \triangleq$ Kniegelenk, $\Theta_2 \triangleq$ Winkel des Kniegelenks
- $F \triangleq$ Fußgelenk, $\Theta_3 \triangleq$ Winkel des Fußgelenks
- $T \triangleq$ Endpunkt
- $HK \triangleq$ Abstand bzw. Strecke Hüft- zu Kniegelenk
- $KF \triangleq$ Abstand bzw. Strecke Knie- zu Fußgelenk
- $FT \triangleq$ Abstand bzw. Strecke Fußgelenk zu Endpunkt
- $P_1 \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} \triangleq$ anzufahrender Punkt

Schritt 1

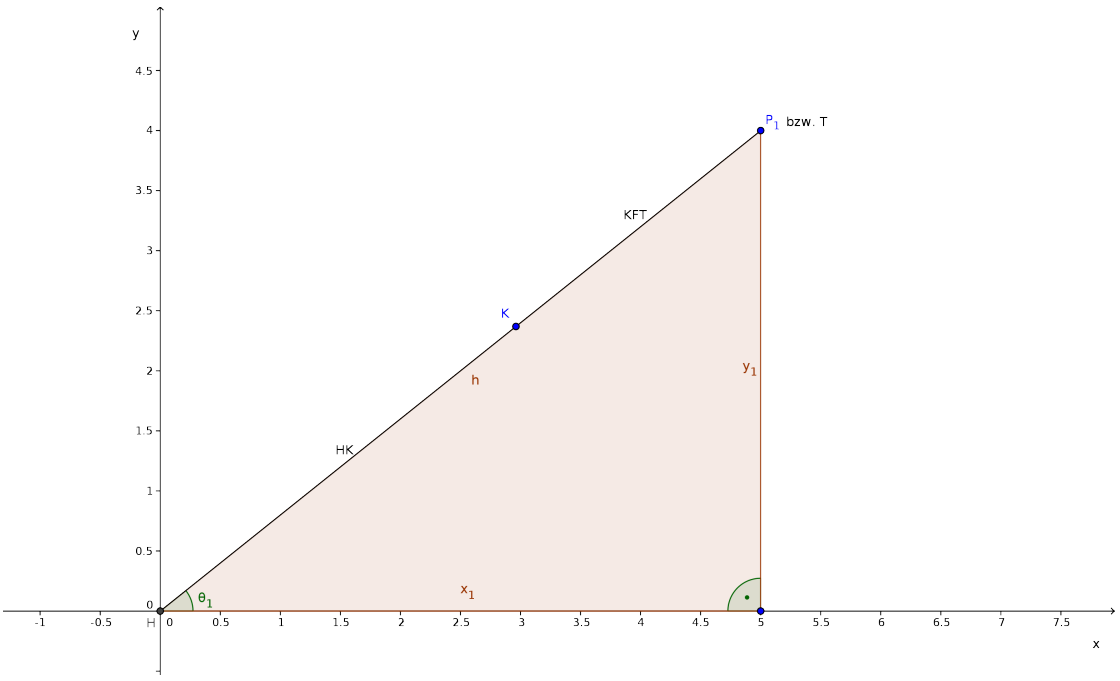

Abbildung 4.1: Schritt 1: x-y-Ebene

In Abbildung 4.1 ist der erste Schritt der Berechnung dargestellt. Hierbei wird das Bein in der x-y-Ebene ausgerichet, sodass eine gedachte Gerade, welche durch alle Achsen eines Beines verläuft, den anzufahrenden Punkt in dieser Ebene schneidet. Dafür muss der Winkel für das Hüftgelenk berechnet werden.

Die Strecken x_1, y_1 und HK sind bekannt. Somit können der Winkel für das Hüftgelenk Θ_1 und die Strecke HT, oder kurz h, berechnet werden:

- $\Theta_1 = \arctan\left(\frac{y_1}{x_1}\right)$
- $h = \sqrt{{x_1}^2 + {y_1}^2}$

Schritt 2

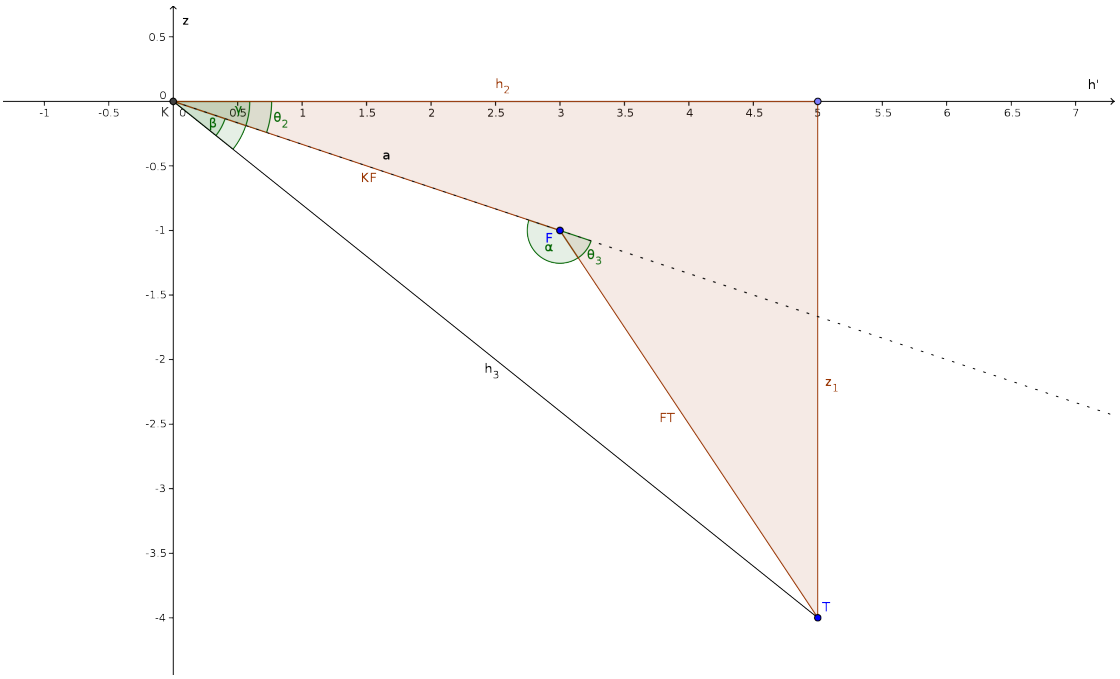

Abbildung 4.2: Schritt 2: h-z-Ebene

In Abbildung 4.2 ist der zweite Schritt der Berechnung dargestellt. Hierbei bildet die gedachte Gerade, aus dem Schritt 1, eine neue Koordinatenachse h'. Das Kniegelenk ist der Nullpunkt auf dieser Achse. Die zweite Achse in dem gedachten Koordinatensystem bildet die z-Achse. Um den Punkt P_1 anfahren zu können, muss mit Hilfe des Knie- und des Fußgelenks noch die „Entfernung" auf der h'-Achse, sowie die „Höhe" auf der z-Achse eingestellt werden. Da in dieser Ebene zwei Freiheitsgrade vorhanden sind, kann der Punkt P_1 auf zwei Wegen angefahren werden. In dieser Berechnung wird ein Weg fest vorgegeben.

Im zweiten Schritt sind die Strecken KF, FT und z_1 bekannt. Um die „Entfernung" auf der h'-Achse anfahren zu können, muss zunächst h_2 berechnet werden. Anschließend können die Winkel Θ_2 für das Kniegelenk und Θ_3 für das Fußgelenk berechnet werden:

- $h_2 = h - HK$
- $h_3 = \sqrt{{h_2}^2 + {z_1}^2}$
- $\alpha = \arccos\left(\frac{-(h_3{}^2)+FT^2+KF^2}{2*FT*KF}\right)$
- $\beta = \arcsin\left(\frac{FT/h_3}{\sin\alpha}\right)$
- $\gamma = \arcsin\left(\frac{|z_1|}{h_3}\right)$
- $\Theta_2 = \gamma - \beta$
- $\Theta_3 = \pi - \alpha$

5 Programmierung

In diesem Kapitel wird die Programmierung des Roboters beschrieben. Zu Beginn wird kurz die erstellte API vorgestellt. Anschließend wird die Umsetzung eines einfachen Laufalgorithmus erläutert.

5.1 API

Eine genaue Beschreibung der Dateien und deren Funktionen kann aus den Sourcecode mit Hilfe des Tools Doxygen[1] erstellt werden. Der Sourcecode sowie die aktuelle API im HTML-Format ist im BerliOS-Projekt[2] verfügbar.

communication.h Methoden zur Kommunikation der CPUs

datatypes.h Abstraktion der elementaren Datentypen und Definition von benötigten Strukturen

dynamixel.h Methoden und Protokoll zur Ansteurung der Dynamixel AX-12

kinematics.h Lösungsmethoden für inverse sowie direkte Kinematik

movement.h Allgemeine Methoden für den Ablauf eines Laufalgorithmus

remote.h Methoden für die Nutzung des Gamepads

usart_driver.h Erweiterter Beispieltreiber von Atmel für die USART

utils.h Hilfsmethoden zur Ausgabe und Umrechnung

xmega.h Spezifische Methoden zur Ansteuerung und Initialisierung der Komponenten des Mikrocontrollers ATXmega

5.2 4-Punkte-Lauf

Der 4-Punkte-Lauf ist ein sehr einfacher Laufalgorithmus, der den Roboter ein Geradeauslaufen durch das Anfahren von vier verschiedenen Koordinaten ermöglicht. Bei einem Hexapod müssen sich zu jedem Zeitpunkt mindestens drei Beine und mindestens eins auf jeder Seite auf dem Boden befinden.

Jedes der sechs Beine benötigt entweder vier Roboterkoorinaten oder vier Weltkoordinaten, welche koordiniert angefahren werden müssen. In Abbildung 5.1 auf der nächsten Seite ist diese Bewegung verdeutlicht. Alle Beine mit einem ausgemalten schwarzen Kreis befinden sich auf dem Boden und alle anderen haben keinen Bodenkontakt.

In der konkreten Implementierung, `movement4Points.c`, wird mit Weltkoordinaten gearbeitet. Der Nullpunkt des Weltkoordinatensystems ist die Mitte des Chassis und ein Punkt dieses Systems wird im folgendem als globaler Punkt bezeichnet. Jedes Bein besitzt sein eigenes Roboterkoordinatensystem, dessen Punkte als lokale Punkte bezeichnet

[1] `http://www.doxygen.org`
[2] `http://developer.berlios.de/projects/cm-bot/`

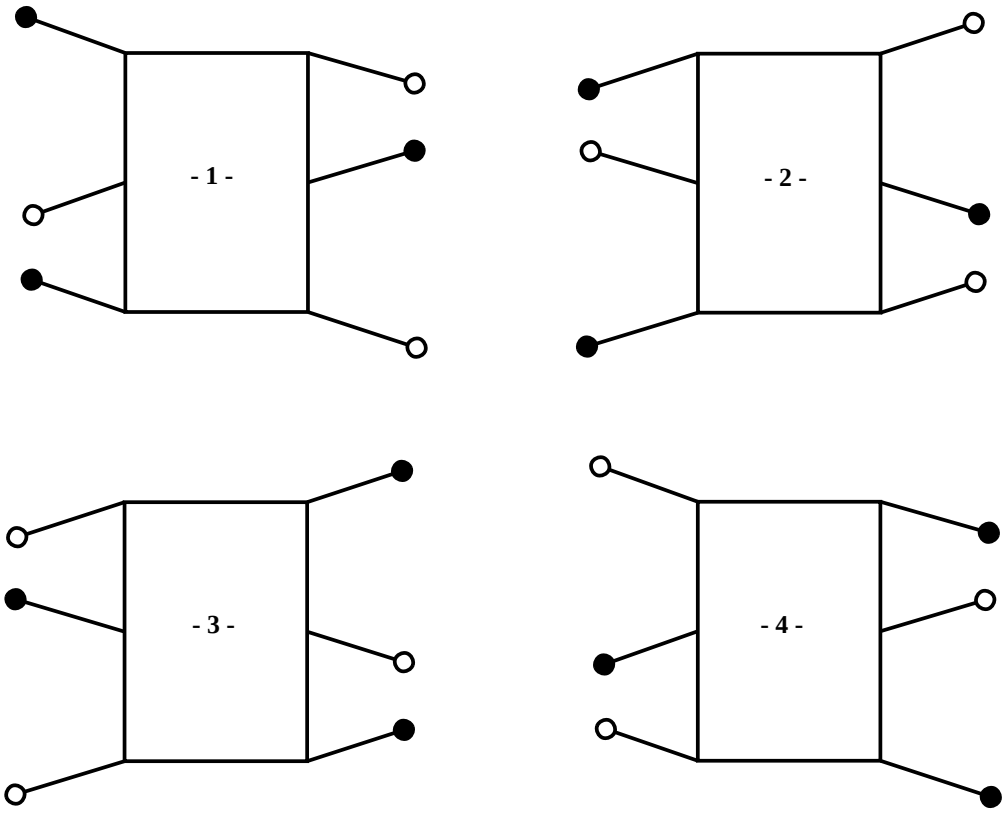

Abbildung 5.1: Darstellung einer Laufbewegung in vier aufeinander folgenden Schritten.

werden. Aus diesen lokale Punkten können die Roboterkoordinaten, dass heisst die anzufahrenden Gelenkwinkel für dieses Bein, errechnet werden.

Theoretisch könnte nur mit vier Roboterkoordianten geareitet werden, da für jedes Bein die selben Roboterkooridnaten genutzt werden könnten. In der praktischen Umsetzung werden jedoch globale Punkte genutzt, um die Transformation dieser in lokale Punkte und die nötige Berechnung der Roboterkoordinaten zu demonstrieren.

Damit nicht für jedes Bein vier Weltkoordinaten ermittelt werden müssen, wird das rechte mittlere Bein als Basis genutzt. Aus den globalen Punkten für dieses Bein können durch Addition bzw. Subtraktion der bekannten Abstände in der x-y-Ebene der Beine zueinander die globalen Punkte für jedes weitere Bein errechnet werden. Diese globalen Punkte werden durch einen CPU, den Master, in einer bestimmten Reihenfolge an die anderen CPUs, die Slaves, gesendet. Diese Punkte müssen nun zuerst von allen CPUs in lokale Punkte für das bestimmte Bein transformiert werden, um anschließend aus diesen die Roboterkoordinaten zu berechnen und die geforderte Position anfahren zu können.

Der Master steuert hierbei die gesamte Bewegung, indem er jedem Beinpaar die korrekten Punkte in einer festen Reihenfolge zuteilt. Die Slaves haben in dieser Implementierung keine Kenntnis über den verwendeten Laufalgorithmus, da sie nur die empfangenen Befehle des Masters ausführen. Somit kann auf allen Slaves das selbe Programm verwendet werden und durch eine Umprogrammierung des Masters, kann der Laufalgorithmus geändert werden.