Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projekt 1, SoSe2012

Jan Ruhnke

Project AMEE a Quadruped Rough Terrain Robot

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Jan Ruhnke

ruhnke@enapse.de

Project AMEE a Quadruped Rough Terrain Robot

Studienarbeit eingereicht im Rahmen des Masterprojekts 1

im Studiengang Master of Science Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. rer. nat. Gunter Klemke
Prüfer : Prof. Dr. rer. nat. Kai von Luck

Abgegeben am 30.08.2012

Inhaltsverzeichnis

1 Einführung

Dieses Projekt führt die Bachelor Thesis von Björn Bettzüche [Bet10] und Jan Ruhnke [Ruh11] fort und setzt einige Konzepte der Vorarbeiten ([Bet12] [Ruh121] [Ruh12] und folgende) experimentell um.

1.1 Motivation

Nach Unfällen, Naturkatastrophen und Terroranschlägen entsteht oft eine Situation, in der sich die Helfer selbst in Lebensgefahr begeben. Für die Helfer ist es primär, sich einen Überblick zu verschaffen. Für diese Aufgabe sollen sogenannte USAR-Roboter eingesetzt werden. Diese Roboter müssen extrem agil sein und sich sicher auf unebenem Untergrund bewegen können. Das *Projekt AMEE* beschäftigt sich mit der Entwicklung dieser Roboter. In der ersten Phase des Projekts soll der vierbeinige Roboter *AMEE* realisiert werden. Diese Art von Robotern wird als *Quadruped Rough Terrain Robot* (QRTR) bezeichnet. Die Roboter *BigDog* und *AlphaDog* der Firma Boston Dynamics [Bos10] werden vom Projektteam als Herausforderung und Referenz angesehen.

1.2 Inhalt des Projektberichts

Es wurde experimentell das *Robotics Development Studio R4 von Microsoft®* (MS-RDS) auf die Verwendbarkeit in einem QRTR geprüft. Für erste Tests des Kontrollerkonzepts [Bet10] [Ruh121] wurde eine verkleinerte und sehr simple Konstruktion eines Laufsystems mit dem Lego® Mindstorm® Bausystem erstellt *(Little-Lego® AMEE)*. Die Grundzüge des Deliberative- / Reactive-Layer Konzepts [Reb08] wurden implementiert. Weiterhin wurde eine Marktsondierung der Antriebe für die Realisierung des Roboters AMEE durchgeführt.

1.3 Ziele im Masterprojekt 1

Die Frage, ob sich ein Eventsystem mit dem von Microsoft® angebotenem *Concurrency and Coordination Runtime (CCS)* und *Decentralized Software Services (DSS)* Framework aus dem MS-RDS in einem Laufsystem realisieren lässt, wurde zwar in der Arbeit von Björn Bettzüche [Bet10] geklärt. Auch wurde in der Arbeit von Jan Ruhnke [Ruh11] gezeigt, dass eine Trennung von zeitkritischen und zeitunkritischen Tasks mit „intelligenten“ und physikalisch externen Kontrollern realisiert werden kann. In jeder dieser Arbeiten wurde die andere Arbeit aber nur skizziert. Hauptziel des Projekts 1 ist es zu überprüfen, ob beide Konzepte zusammen real funktionsfähig sind.

2 Projekt AMEE, Proof of Concept

Um die finanziellen Risiken gering zu halten, wird im Projekt 1 eine verkleinerte Version des AMEE QRTR entwickelt. Dazu wurde das Lego® Mindstorm V2 System ausgewählt. Als externe Kontroller wurden zwei NXT V2 mit der Lego® Firmware verwendet. In diesem Kapitel wird ein Proof of Concept durchgeführt.

2.1 Little-Lego®-AMEE

Die mechanische Konstruktion von AMEE kann mit dem Mindstorm® System schwer nachgebaut werden. Das geplante AMEE Roboterlaufsystem verwendet sechzehn Antriebe in den Gelenken. Für den AMEE QRTR ist dies die minimale Anzahl, um sicher im Gelände laufen zu können [Ruh11]. Um die Grundkonzepte experimentell zu testen, ist es ausreichend feste Schrittlängen zu erproben, da nur die Interaktion zwischen zeitkritischen Kontrollern und dem eventgetriebenen Hauptsystem von Interesse sind.

2.2 Hardware Aufbau

Das mechanische AMEE Konzept wurde für Little-Lego®-AMEE, auf insgesamt vier Antriebe, abstrahiert. Jeder Fuß kann damit nur auf einer festen Kreisbahn bewegt werden. Dies wird in Abbildung 2-1 mit der unterbrochenen grünen Line dargestellt.

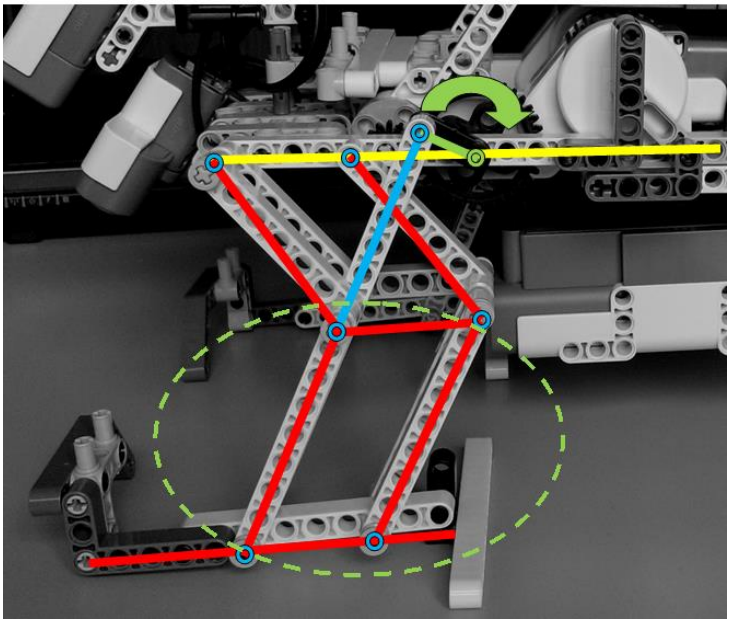

Abbildung 2-1 Aufbau des Little-Lego® AMEE Beins

Der Fuß wird über eine Stützkonstruktion (rote Linien, Abbildung 2-1) immer parallel zum Torso (gelbe Line) des Roboters gehalten. Die grüne Line (oben) Kennzeichnet einen Exzenter, der die Drehbewegung des Antriebs auf die Beinkonstruktion überträgt.
Die Antriebe des Mindstorm® V2 Systems bieten zwar genügend Drehmoment für diese Anwendung. Da aber dieses Baukastensystem auch von Kindern benutzt wird, reagiert der NXT V2 Kontroller sehr empfindlich auf Drehmomentspitzen. Das Verhalten gleicht einem Integralregler. Vermutlich soll damit einer möglichen Verletzungsgefahr vorgebeugt werden. Aus diesem Grund wurde das Drehmoment mit zwei zusätzlichen Zahnrädern auf 3:1 untersetzt[1]. Mit diesem Aufbau können die vier Beine unabhängig bewegt werden. Der Lego® NXT V2 Kontroller kann drei Antriebe verwalten. Für diese Konstruktion werden zwei NXT Kontroller verwendet, die jeweils das vordere und das hintere Beinpaar ansteuern (Abbildung 2-2, NXT.a & NXT.b). Für weitere Experimente wurden der Ultraschall-entfernungssensor und der Farbsensor als „Kopf“ montiert (Abbildung 2-2). Diese Sensoren werden aber in diesem Test nicht benötigt.

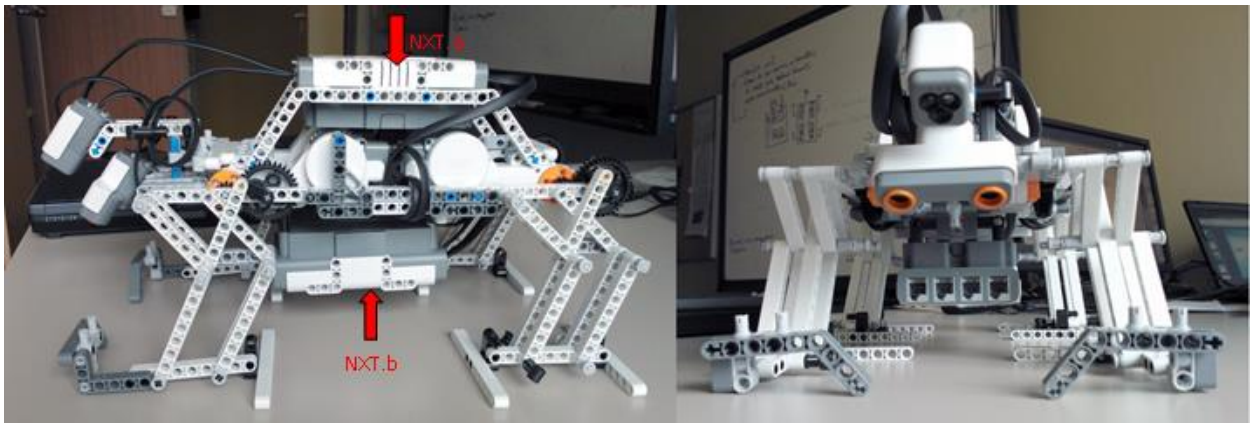

Abbildung 2-2 Aufbau von Little-Lego® AMEE

Die NXT Kontroller kommunizieren (über Bluetooth®) mit dem Hauptrechner, auf dem das MS-RDS Framework arbeitet.

2.3 Software Architektur und Modellierung

Die NXT Kontroller regeln die Antriebe. In der Firmware ist ein Automat implementiert, der zyklisch die Aktoren und Sensoren abarbeitet. Der Kontroller arbeitet die gesendeten Events von MS-RDS ab. Soll beispielweise ein Motor um einen bestimmten Winkel gedreht werden,

[1] Dem Projekt Team liegt der Source-Code des NXT V2 vor. Der Aufwand diesen Code zu verändern erschien uns aber zu groß, da die Drehmomentkontrollen an unterschiedlichen Stellen mehrfach implementiert wurden.

sendet MS-RDS einen abstrakten Befehl (drehe Motor A um 45°). Der NXT Kontroller empfängt diese Nachricht und sendet eine Empfangsbestätigung zurück. Der Kontroller regelt dann selbständig den Antrieb auf den angegeben Winkel. Ist dieser Auftrag abgearbeitet, sendet er den Erfolg (oder Misserfolg) an MS-RDS zurück [Bet10]. Damit stellt der NXT Kontroller den zeitkritischen Echtzeitanteil des Systems dar und ist eine vereinfachte Form des AMEE-Beinkontrollers [Ruh11]. Im Deliberative- / Reactive-Layer Konzept [Reb08] ist hier der Reactive-Layer in einen Echtzeit- (Beinkontroller) und Eventteil (MS-RDS) getrennt [Ruh121].
Da das Ziel eine Steuerung über ein Eventsystem ist, werden die Einzelbewegungen als Zustände betrachtet. Die Motor-Services von MS-RDS bieten die Möglichkeit, die Antriebe um eine frei wählbare Anzahl an Umdrehungen zu verstellen. Drei Umdrehungen des Antriebs entsprechen einem kompletten Bewegungsablauf eines Beins. Damit eindeutige Events erzeugt werden können, wurde ein Schritt des Beins in folgende Zustände abstrahiert (Abbildung 2-3).

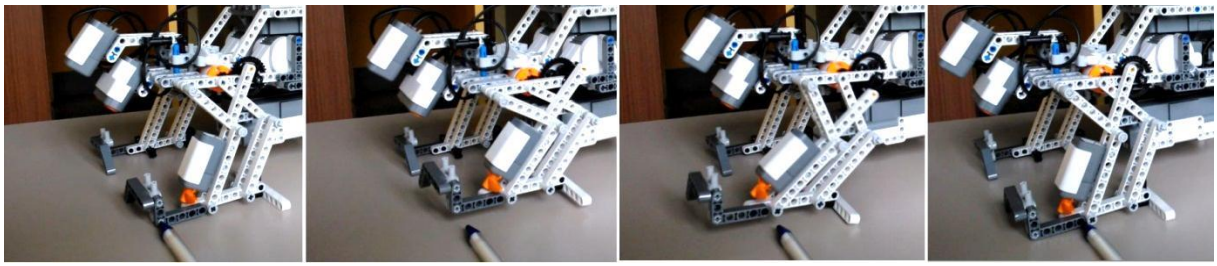

Abbildung 2-3 Bewegungsablauf(l.n.r.) Idle, Raising, Lowering, Pulling/Idle

In der linken Abbildung ist das Bein vollständig ausgestreckt und wird als *Idle* Zustand definiert. Im zweiten Bild (von links) ist der Fuß an den Körper gezogen (*Raising* Zustand). Im dritten Bild (von links) wird der Fuß, nach vorn und unten, in Laufrichtung bewegt (*Lowering* Zustand). Der Zustand *Idle* wird wieder, über den Zustandsübergang *Pulling* hergestellt. Dabei wird mit *Pulling* der Torso des Roboters nach vorn gezogen. Diese Reihenfolge ist ein kompletter Schritt des Beins.
Es wurde ein generischer Bein-Service (Abbildung 2-5) implementiert, mit obigen Zuständen als Automat. Der Automat überwacht auch die logische Abfolge der Einzelbewegungen. Damit kann über den Service *LittleAMEE.Devices* ein *LittleAMEEBeinService* erzeugt werden, der die nötigen Methoden und Schnittstellen für MS-RDS enthält. Die Services wurden in .net C# 4.0 implementiert und können auch im *Visual Programming Language* Editor (VPL) [Bet10] verwendet werden.

Die Koordination der Beine wurde mit VPL realisiert. Es wurde für jedes Bein ein spezifizierter Bein-Service erzeugt (Abbildung 2-4, „Vorn Rechts Bein Service“ usw.). Dadurch kann jedes Bein unabhängig, über abstrakte Events, bewegt werden.
Es wurden zwei übergeordnete Services für die Laufmuster *Fast-Walk* und *Save-Walk* implementiert. Wird ein *Fast-Walk*- oder *Save-Walk*-Event ausgelöst, regelt der jeweilige Service die nötige Schrittfolge, indem er die entsprechenden Events für jedes Bein auslöst. Pro Aufruf durchlaufen alle vier Beine einen Bewegungszyklus der Bein-Services. Der *Save-Walk* ([Ruh11] Abs. 2.5.1.) bewegt nur ein Bein zurzeit und setzt den Fuß in die vorgreifende Position. Sind alle vier Beine platziert worden, wird mit allen Beinen der Torso synchron nach vorn gezogen. Damit wird eine sichere Dreipunktauflage sichergestellt. Der *Fast-Walk* bewegt Beinpaare synchron. Ein Beinpaar ist jeweils, ein Vorderbein und das gegenüberliegende Hinterbein. Dabei sind die Bewegungszyklen zueinander versetzt ([Ruh11] Abs. 2.5.2. Fast Static Walk). Dies ermöglicht schnelleres Laufen, auf Kosten der Stabilität des Roboters.

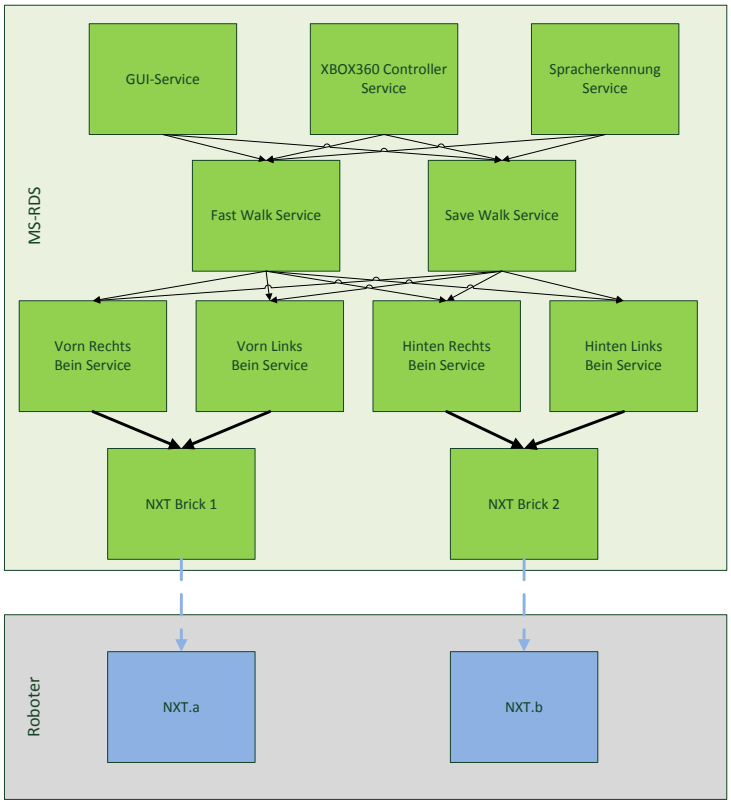

Abbildung 2-4 Schematische Darstellung der Services

Der Roboter Little-Lego®-AMEE kann über mehrere Mensch-Maschine-Schnittstellen gesteuert werden. Es wurde eine einfache GUI implementiert. Zudem kann auch ein XBOX® 360 Controller verwendet werden. Weiterhin wurde eine Sprachsteuerung mit dem Spracherkennungsservice von MS-RDS implementiert. Der Roboter kann mit den Sprachbefehlen: „AMEE schnell“ im *Fast-Walk* laufen, „AMEE langsam“ im *Save-Walk* laufen, „AMEE Stop“ anhalten und mit „AMEE gib Laut“ eine Melodie spielen. Auch dies erfolgt über das Eventsystem. Nähere Details zum MS-RDS und das Zusammenspiel von Services, CCS und DSS wird ausführlich in der Bachelor Thesis von Björn Bettzüche [Bet10] beschrieben. Die Erzeugung von Schrittmustern, Mechanik und die Softwarearchitektur der Beinkontroller werden in der Bachelor Thesis von Jan Ruhnke [Ruh11] genauer erläutert.

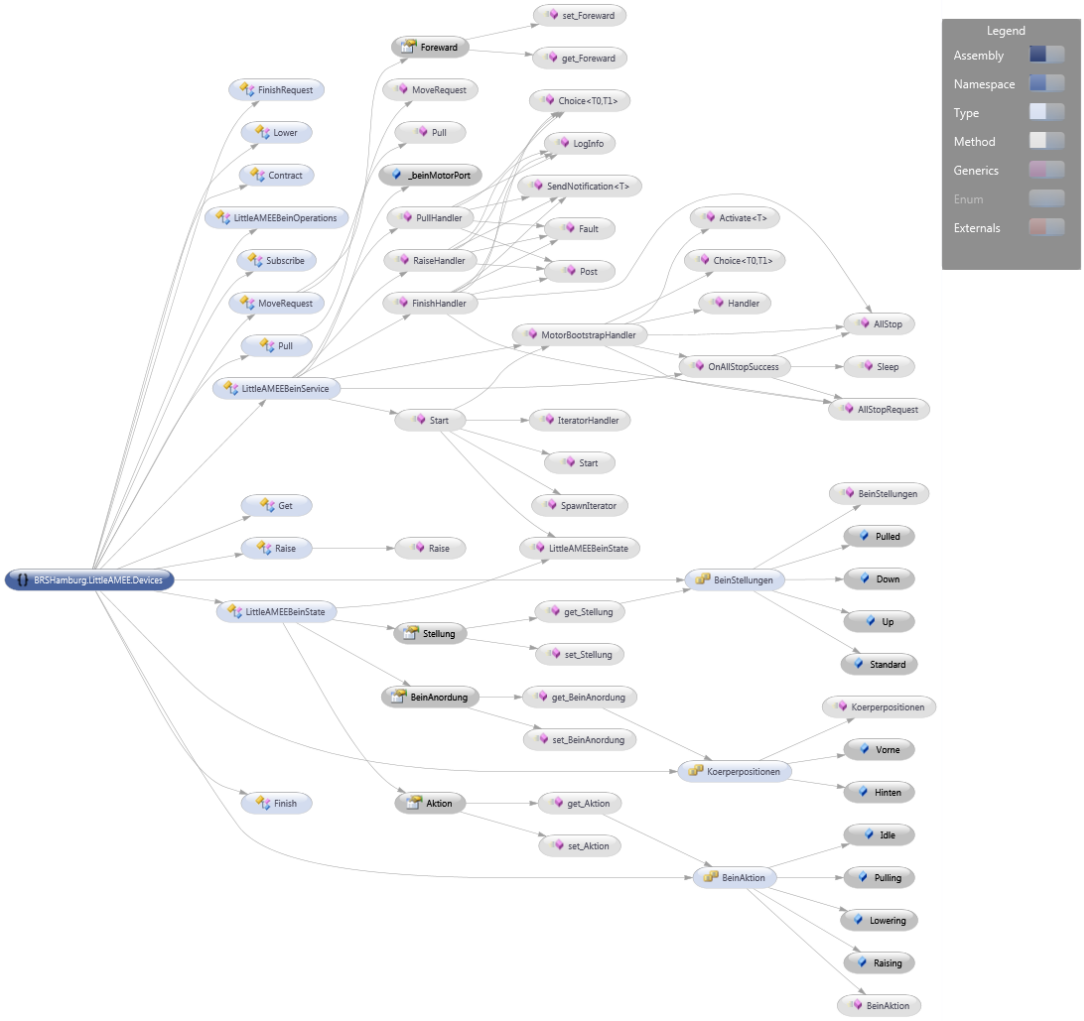

Abbildung 2-5 Aufbau des Bein-Services

2.4 Ergebnisse mit Little-Lego®-AMEE

Die Versuche mit dem Roboter Little-Lego®-AMEE haben das Konzept aus den Bachelor Thesis bestätigt. Es ist möglich, ein vierbeiniges Roboterlaufsystem mit einem verteilten Eventsystem zu steuern. Die Kapselung der zeitkritischen Task in physikalisch externe Kontroller ist erfolgreich. Beim Eventsystem konnte keine Eventflut oder Eventüberholung festgestellt werden.

Obwohl keine Maßnahmen zur Synchronisation der Antriebe unternommen wurden, konnte eine maximale Abweichung von 500ms gemessen[2] werden. Dies ist umso positiver zu sehen, da MS-RDS auf einem Windows®7 SP1 x64-OS betrieben wurde. Windows® 7 bevorzugt im Scheduler, Programme vor Services und das Reaktive-System läuft als Windows-Service im Hintergrund. Zudem liefen circa fünf weitere Hintergrundprogramme.

Wie bereits im Abs. 2.2 erwähnt, reagiert der NXT V2 Kontroller empfindlich auf Drehmomentspitzen. Diese Reaktion wird noch von einem Spannungseinbruch der Batterien verstärkt, wenn diese unter 70% der vollen Ladung absinken. So konnte mehrfach beobachtet werden, dass ein Bein schnell zur *Idle* Position fuhr. Der Fuß dann aber beim ersten Bodenkontakt zurückgezogen wurde. Leider wird dabei kein negativer Event ausgelöst, sondern der NXT Kontroller fuhr nochmal die Position sehr langsam und erfolgreich an. Als Folge warteten andere Events auf die Rückmeldung und der Roboter ist zeitweise sehr langsam gelaufen. Dies könnte mit der Veränderung der Drehmoment-überwachung (im NXT Kontroller) korrigiert werden. Weiterhin bietet MS-RDS noch die Möglichkeit, einen Timeout festzulegen. Dies wurde nicht mehr realisiert, da unsere Ausgangsfrage und das Proof of Concept bestätigt wurden.

Es konnte auch die Frage nach der Feingranularität der Services beurteilt werden. Für die *full size* Realisierung von AMEE bedeutet dies, dass nicht für jedes Bein ein Service erstellt wird. Es wird nur einen Lauf-Service geben, der die reaktiven Funktionen in sich vereint. Dieser Service wird wahrscheinlich den Reactive-Layer [Ruh121] [Ruh12] [Reb08] beinhalten und nur Schnittstellen zu MS-RDS anbieten. Nach unserer Ansicht macht es hier nur Sinn, Prozesse mit hoher Last durch ein Eventsystem zu verteilen. Dies begründen wir mit dem relativ hohen Programmieraufwand für einen eigenen Service (Siehe Abbildung 2-5). Die hohe Verarbeitungsgeschwindigkeit [Bet10] von MS-RDS Events ermöglicht hier jeglichen Grad der Granularität, ohne die Performance des Systems signifikant zu beeinflussen.

[2] Gemessen durch Auswertung von Videoaufnahmen des Roboters.

3 Full size AMEE - Marktsondierung Antriebe

Da das aufgestellte Konzept bestätigt werden konnte, wird der Roboter AMEE in voller Größe realisiert [Ruh11]. Seit der Bachelor Thesis [Ruh11] konnte mit Matlab® [Ruh12] ein Simulationstool erstellt werden, mit dem beliebige Gliederlängen und Gewichte des Laufsystems simuliert werden können. Es werden in der Drehmomentspitze 44Nm pro Antrieb, bei einem Gesamtgewicht des Roboters von ca. 36kg, benötigt. Die Antriebe im Prototypenbein liefern ein Drehmoment von 22Nm. Aus diesem Grund müssen neue Antriebe für die full size Realisierung gefunden werden.

Für einen QRTR sind die Antriebe eine der wichtigsten Komponenten, die das gesamte Design bis zur Software beeinflussen. Folgende Antriebskriterien sind dabei wichtig.

Drehzahl: Die Antriebe werden als Direktantriebe in den Gelenken verwendet([Ruh11]Abs. 3.3.3.). Damit ist die Ausgangsdrehzahl hinter dem Untersetzungsgetriebe direkt die Geschwindigkeit bzw. die Winkelgeschwindigkeit der Glieder. Aber um die maximale Ausgangsdrehzahl festzulegen, muss die gesamte Reaktionszeit des Systems berücksichtigt werden. Die maximale Reaktionszeit des Systems konnte mit 1,6ms gemessen werden ([Ruh11] Abs. 5.3.5.4.). Ab der Hüfte des Roboters ist eine Gliederlänge von 665mm geplant. Dies bedeutet: Ein Antrieb mit 30U/min^{-1} bewegt das Bein noch ca. 3,5mm. Erst dann leiten die Beinkontroller die aktiven Bremsvorgänge überhaupt ein.

$$360° \cdot \frac{\frac{30\ Umdrehungen}{Minute}}{60\ Sec} = \frac{180°}{sec};\ \frac{2 \cdot 665mm \cdot \pi \cdot \frac{180°}{sec}}{360°} = \frac{2089mm}{sec};\ \frac{2089mm}{sec} \cdot 0{,}0016\,\mathrm{sec} = 3{,}48mm$$

Der aktive Bremsvorgang ist primär vom Gewicht des Ankers (im Elektromotor) abhängig. Dies wird oft als *Läuferträgheitsmoment* bezeichnet. Bei Versuchen konnte das Prototypenbein in 300ms zum Stillstand gebracht werden. Wird ein Bein bei maximaler Geschwindigkeit in der Hüfte gestoppt, wird noch ein Weg von **630mm** zurückgelegt. Dies ist besonders bei schnellen Punkt-zu-Punkt-Bewegungen kritisch und bereitete schon bei der Beinrealisierung Probleme. Der Antrieb kann zwar langsamer gefahren werden, aber das hohe Anlaufdrehmoment liegt sofort an und fährt über das Ziel hinaus. Es muss ein ausgewogenes Verhältnis zwischen Drehmoment und Drehzahl gewählt werden. Versuche zeigten, das 30U/min^{-1} die maximal handhabbare Ausgangsdrehzahl für die Beinantriebe darstellt ([Ruh11]Abs. 6.4).

Untersetzung: Die Elektroantriebe sollen möglichst leicht und kompakt sein. Ein Elektromotor, ohne ein Untersetzungsgetriebe, wäre mit dem nötigen Drehmoment zu schwer. Das Untersetzungsgetriebe hebt das Ausgangsdrehmoment eines Motors auf Kosten

der Drehzahl an. Zudem sorgt das Getriebe mit einem sog. Schneckenradsatz dafür, dass der Antrieb im stromlosen Zustand nicht gedreht werden kann. Dies bedeutet, dass AMEE keine Energie „zum Stehen“ benötigt. Problematisch ist in diesem Zusammenhang, die kinetische Energie im Anker beim Abbremsen des Motors. Bei einer sehr hohen Untersetzung kann der Motor, bei gleichem Ausgangsdrehmoment, kleiner und damit leichter gewählt werden. Dies benötigt aber wiederum eine höhere Motordrehzahl und damit verlängert sich auch die Abbremszeit des Antriebs (siehe oben Drehzahl) [Gro11]. Hier ist die Auswahl auf die angebotenen Kombinationen aus Motor und Getriebe beschränkt. Marktüblich sind hier Kombinationen von 2.500U/min^{-1} (mit einer Untersetzung von 70:1) bis zu 4.500U/min^{-1} (mit einer 200:1 Untersetzung).

Drehmoment: Entscheidend für das Laufsystem ist das Drehmoment der Antriebe. Benötigt wird ein Antrieb mit einem minimalen Anlaufdrehmoment von 44Nm. Unter Volllast sollte der Antrieb noch 8 – 10Nm leisten können. Ein geringeres Drehmoment würde den Workspace der Beine erheblich einschränken, wie Tests in der Simulation [Ruh12] ergeben haben.

Gewicht: Die obigen Zusammenhänge und Faktoren beeinflussen das Gesamtgewicht des Antriebs. Große Untersetzungen benötigen ein größeres und schweres Getriebe, aber haben einen leichteren Motor. Auch der Umkehrschluss ist hier gültig. Kostengünstige Antriebe aus dem Automobilbau, die obige Kenndaten bieten, haben ein Gewicht von ca. 3 bis 5kg. Eine Ausnahme bilden hier hochpreisige Motoren mit Neodym-Permanentmagneten. Bei den gesuchten Antriebstypen wiegen diese im Durchschnitt 1kg weniger, als herkömmliche Antriebe mit Ferrit-Permanentmagneten.

Bewertung: Der ideale Antrieb wäre der Antrieb aus dem Prototypenbein [Ruh11], mit einer abgewandelten Untersetzung von 400:1. Dies entspräche genau einem Ausgangsdrehmoment von 44Nm. In der Simulation konnte ermittelt werden, dass dieser Antrieb bis zu einem Kilogramm wiegen darf. Der Antrieb aus dem Prototypenbein wiegt 0,7kg. Dieser Antrieb konnte uns noch nicht angeboten werden (Aug. 2012). Alternativ kommen nur schwerere Antriebe in die engere Wahl. Damit würde aber auch das Gesamtgewicht des Roboters auf 75-80kg steigen. Für dieses Gesamtgewicht konnte, durch die Simulation, ein nötiges Ausgangsdrehmoment von circa 70Nm ermittelt werden (Abbildung 4-1, im Anhang). Zurzeit warten wir auf Angebote von einigen großen Herstellern (Stand Aug. 2012).

4 Aussichten

Nach der Beschaffung der Antriebe, muss die Konstruktion der Beine angepasst werden. Diese neuen Antriebe haben wahrscheinlich eine andere Befestigung und Maße. Erst danach können die mechanischen Bauteile gefertigt werden. Dazu wird erst ein Bein mit den neuen Antrieben gefertigt und montiert. Dieses Bein wird auf Funktionsfähigkeit geprüft. Dann werden die anderen Beine und der Torso gefertigt. Für die Fertigung ist die Nutzung einer CNC-Fräse geplant. Dazu wird die Einarbeitung in AutoCAD® von Jan Ruhnke fortgesetzt. Diese CAD/CAM Softwarepakte bieten eine vollständige physikalische Simulation der Bauteile. Die nötigen CAM Dateien können (laut Hersteller) automatisch erzeugt werden. Parallel dazu wird die Software des Beinkontrollers auf eine leistungsfähigere MCU (Atmel® AT32UC3C0512C) angepasst. Hierfür wird auch ein neues Platinenlayout mit integrierter Ethernetschnittstelle und Leistungselektronik entwickelt. Es wurde bereits ein Experte aus dem Fachbereich Elektrotechnik angeworben. Diese Arbeit kann aber auch erst erfolgen, wenn die Frage des Antriebstyps geklärt ist. Da einige Antriebe die jetzige maximale Stromaufnahme von 30A pro Antrieb übersteigen.

Nach diesen Schritten erfolgt die Montage des Roboters. Hierzu müssen über einhundert Kleinteile beschafft werden, bei denen aber die Zulieferer bekannt sind.

Jetzt kann die Implementierung des Hauptkontrollers erfolgen. Parallel dazu werden einige Sensoren beschafft. Besonders der Sensor zur 3D Geländeerfassung ist zurzeit noch schwierig. Hier befindet sich das Team aber schon im Gespräch mit einem Hersteller. Björn Bettzüche wird sich dabei verstärkt mit dem Sensor und der Merkmalsextraktion für die beratende KI [Bet12] [Ruh121]beschäftigen.

Das AMEE-Projektteam versucht weitere Technologiepartner und Teammitglieder zu gewinnen.

Ein genauer Zeitplan wurde zwar erstellt, ist aber von nicht beeinflussbaren Faktoren[3] abhängig. Aus diesem Grund sehen wir Februar / März 2013 als realistischen Termin für die Endmontage an.

[3] Bei der Bauteilbeschaffung müssen die Bauteile sehr umsichtig ausgewählt werden. Da in Europa Erfahrungen mit Laufsystemen fehlen, können die Zulieferer nur wenig Beratung leisten.

Literaturverzeichnis & Quellen

[Bet12] **Bettzüche, Björn. 2012.** *Foothold Selection for Quadruped Locomotion.* [PDF] Hamburg, Germany : HAW-Hamburg, 2012.

[Bet10] **—. 2010.** *Machbarkeitsprüfung zur Entwicklung von SW-Anwendungen mit MS-Robotics Developer Studio für das Robocup Rescue Szenario.* [PDF] s.l., Hamburg : HAW Hamburg, Technische Informatik, Juli 2010.

[Bos10] **Boston Dynamics Inc. 2008, 2010.** *BigDog Overview.* [Internet] 78 Fourth Avenue, Waltham, MA, 02451-7507, US : Boston Dynamics Inc., 2008, 2010. http://www.bostondynamics.com.

[Gro11] **Grote, Karl-Heinrich und Feldhusen, Jörg. 2011.** *Dubbel.* Berlin : Springer Verlag, 2011. ISBN 978-3-642-17305-9.

[Reb08] **Rebula, John R., et al. 2008.** *A Controller for the LittleDog Quadruped Walking on Rough Terrain.* [PDF] Florida 32502, USA : Florida Institute for Human and Machine Cognition, 2008. jrebula@alum.mit.edu.

[Ruh121] **Ruhnke, Jan. 2012.** *A Walksystem for a Quadruped Rough Terrain Robot, Controller Concepts AW1.* [PDF] Hamburg, Germany : HAW-Hamburg, 2012. AW1.

[Ruh11] **—. 2011.** *Entwicklung und Realisierung eines vierbeinigen USAR-Roboter-Laufsystems.* [PDF] Hamburg, Germany : HAW-Hamburg Dep. Technische Informatik, 02. Juni 2011. Bachelor Arbeit.

[Ruh122] **—. 2012.** *Quadruped Rough Terrain Robot,Controller Concepts, related Works AW2.* [PDF] Hamburg, Germany : HAW-Hamburg, Department Informatik, 30. 08 2012.

[Ruh12] **—. 2012.** *Simulation der Drehmomente in einem Roboter Bein, Ausarbeitung Modellierung Technische Systeme .* HAW-Hamburg : s.n., 2012.

Anhang

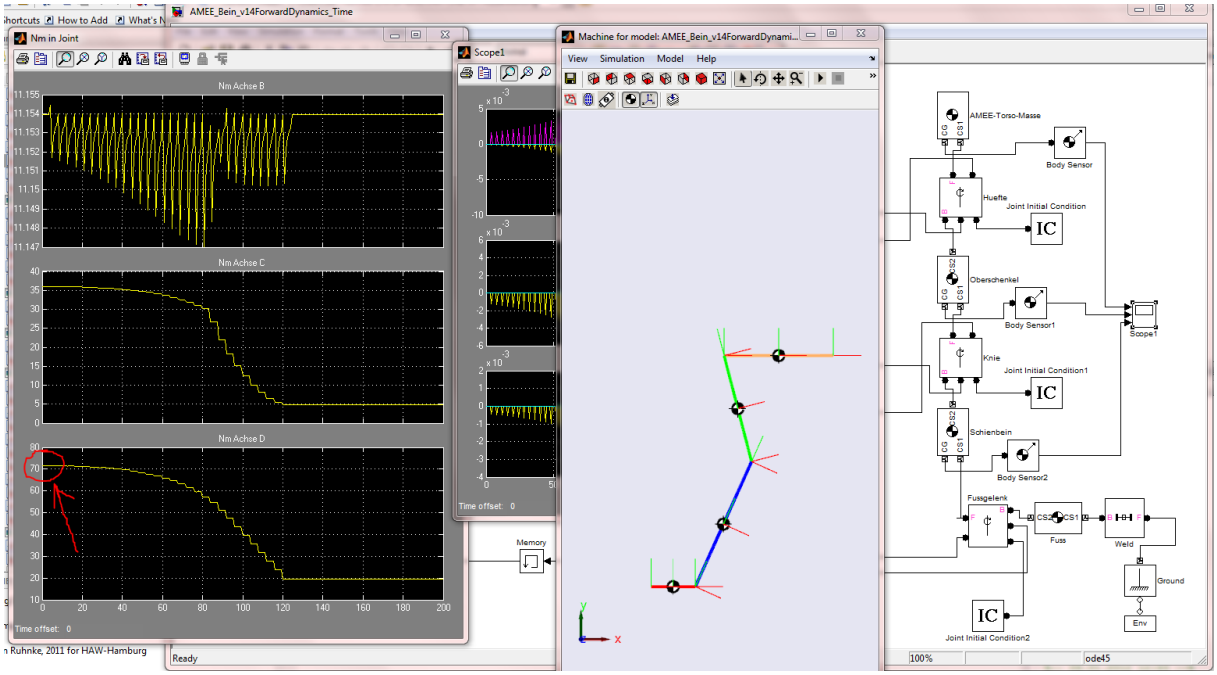

Abbildung 4-1 Simulation der Drehmomente

Hinweis

Das Projekt AMEE wird von der HAW-Hamburg finanziell gefördert und von Microsoft® unterstützt.
Herzlichen Dank an unseren Mentor Gunter Klemke, da er uns jederzeit mit Rat und Tat zur Seite steht.