



Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Studienarbeit Masterseminar

Jan Ruhnke

A CONTROLLER FOR QUADRUPEL LOCOMOTION OVER
ROUGH TERRAIN – Master Thesis Outline

Jan Ruhnke

ruhnke@enapse.de

A CONTROLLER FOR QUADRUPED LOCOMOTION OVER ROUGH TERRAIN



Studienarbeit eingereicht im Rahmen der Veranstaltung Master Seminar

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 28.02.2013

Jan Ruhnke

Thema der Ausarbeitung

A CONTROLLER FOR QUADRUPED LOCOMOTION OVER ROUGH TERRAIN

Stichworte

USAR – Roboter, vierbeiniges Laufsystem, Quadruped Rough Terrain Robot, Controller

Kurzzusammenfassung

Diese Arbeit gibt einen kurzen Ausblick, was in der Master-Thesis bearbeitet werden soll. Der Themenschwerpunkt ist ein Hauptcontroller für einen vierbeinigen Roboter, der über unebenes Gelände laufen kann.

Jan Ruhnke

Title of the paper

A CONTROLLER FOR QUADRUPED LOCOMOTION OVER ROUGH TERRAIN

Keywords

Quadruped robot inspired by mammals, Search and Rescue Robotics, Quadruped Rough Terrain Robot, Controller

Abstract

This work gives a short view, what should be edited in the master thesis. The thematic focus is a main controller for a Quadruped Rough Terrain Robot.

Inhaltsverzeichnis

1	Forschungsgebiet	1
1.1	Rough Terrain Robots	1
1.1.1	Quadruped Rough Terrain Robots	1
1.2	Project AMEE	2
2	Fragestellungen / Behauptung	2
2.1	Behauptung 1	2
2.2	Behauptung 2	5
3	Projektstand / Versuchsaufbau	6
3.1	Realisiert (Stand Feb. 2013)	6
3.2	Noch zu Realisieren	6
3.3	Roboter AMEE XW2	7
4	Schwerpunkt der Master Thesis	8
4.1	Hauptkontroller (Main-Controller)	9
4.1.1	Higher Logic-Layer	9
4.1.2	Deliberative Layer	9
4.1.3	Reactive Layer (Pre Online & Online)	11
5	Risiken	13
	Literaturverzeichnis	A
	Anhang	C
	Sponsoring	D

1 Forschungsgebiet

Die Master-Thesis wird sich mit einem speziellen Bereich der Robotik beschäftigen. Es sollen Fragestellungen und Verfahren für einen Quardruped Rough Terrain Robot (QRTR) erörtert werden. Dabei werden neue und modifizierte Verfahren an einer realen Maschine getestet. Hierfür muss eine eigene Roboterplattform entwickelt und realisiert werden. Die Entwicklung und Realisierung wird in der Master-Thesis durch die Sicht der Informatik getrieben. Dank der Vorarbeiten wird der Schwerpunkt der Thesis auf dem Hauptkontroller liegen. Die Summe aller Arbeiten des AMEE Teams sollte es aber ermöglichen jeden Schritt der Entwicklung nachzuvollziehen.

1.1 Rough Terrain Robots

Im Gegensatz zu Industrierobotern, sind Rough Terrain Robots für den Einsatz im Außenbereich konzipiert. Viele dieser Maschinen agieren im urbanen Umfeld und es können bestimmte Umweltbedingungen vernachlässigt werden. Ein weiterer Zweig der Forschung beschäftigt sich mit Robotern, die extremen Umweltbedingungen ausgesetzt sind. Hier liegt der Schwerpunkt auf dem Einsatz in nicht erschlossenen Umgebungen, wie beispielsweise einem Wald oder im Gebirge.

Ein simples Ziel ist dabei, dass Roboter autonom jeden Punkt der Erde erreichen können. In die Klasse der Rough Terrain Robots werden fliegende, schwimmende, tauchende, fahrende, laufende und steigende (kletternde) Roboter eingeteilt. Steigende Roboter werden in Klassen nach Beinanzahl eingeteilt. In diesem Zweig der Forschung werden Roboter betrachtet, die meist von Säugetieren abgeleitet bzw. inspiriert sind.

1.1.1 Quardruped Rough Terrain Robots

In den letzten Jahren wurden die Forschungsarbeit an QRTRs verstärkt, da die Lösung der Natur, sich mit vier Beinen fortzubewegen, einige Vorteile gegenüber anderen bietet [Ruh11]. ORTRs scheinen zurzeit der vielversprechendste Ansatz zu sein, damit eine Maschine beispielweise Schutt und Geröll überwinden kann. Dieser Punkt ist von besonderem Interesse im sog. USAR-Szenario. Dort handelt es sich zwar um eine urbane Umgebung, die aber durch Unfälle oder Naturkatastrophen zerstört wurde und damit wieder in den Bereich der unerschlossenen Umwelt fällt. Aus diesem Grund erörtern fast alle Arbeiten aus diesem Bereich das Laufen im Gelände und betrachtet das USAR-Szenario vorerst als sekundär.

1.2 Project AMEE

Im studentischen Project AMEE wird ein QRTR komplett neu entwickelt und realisiert. Für diesen Roboter (AMEE XW2) wurde eine mechanische Konstruktion entwickelt (Projekt2). Auch die Elektronik, Controller und Hauptcontroller wurden bzw. werden neu entwickelt.

Das System basiert auf einem verteilten System, dass durch Coprozessoren zeitkritische von zeitunkritischen Komponenten trennt [Ruh122] [Ruh121]. Hierfür wurden eigene Beincontroller realisiert, die die Regel- und Steueraufgaben stark vom Hauptcontroller kapseln. Diese Kapselung ist so stark ausgeprägt, dass der Hauptcontroller kein „Wissen“ über die interne Steuerung der Beine besitzt. Die Beine (Beincontroller inkl. Motion Controller) kommunizieren mit dem Hauptcontroller über Events, die nur Informationen enthalten. Der Hauptcontroller bearbeitet meistens zeitunkritische Aufgabe. Abstrahiert auf das Laufen werden hier Schrittpositionen, unter Beachtung der Stabilität des Roboters, bearbeitet. Natürlich werden hier auch Aufgaben der Orientierung, Pfadplanung usw. ablaufen.

Die Master-Thesis wird sich im Schwerpunkt mit dem Hauptcontroller und dessen Verhalten mit den Beincontrollern beschäftigen, da die Entwicklung der Beincontroller abgeschlossen ist [Ruh11]. Der entwickelte Roboter AMEE XW2 stellt damit einen aufwendigen Versuchsaufbau dar.

2 Fragestellungen / Behauptung

Wie bereits angedeutet ergeben sich zwei Hauptfragestellungen für die Master-Thesis. Diese Hauptfragestellungen umfassen in der Folge weitere Subfragestellungen. In der Master Thesis sollen einige der folgenden Fragestellungen geklärt bzw. die aufgestellten Behauptungen bestätigt werden.

2.1 Behauptung 1

Ein statischer / pseudodynamischer Läufer ist einem dynamischen Läufer im unebenen Gelände überlegen.

Warum?

Dynamischer Läufer: Vereinfacht betrachtet, fällt ein dynamischer Läufer in Laufrichtung. Die Beine fangen dabei den Sturz ab (ZMP [Ruh122] [Ruh121]). In den letzten 20 Jahren konzentrierte sich die Forschung mehr auf dynamische Läufer, da Menschen und viele Säugetiere [Hon07] [Nac10] so energieeffizient laufen. Dies trifft aber nur auf das Laufen auf

relativ glattem Untergrund zu. Sobald eine Steigung oder ein Gefälle von über ~20% auftritt, wechseln fast alle Läufer in ein statisches Verhalten. Dabei wird der Schwerpunkt näher am Untergrund gehalten und die Beinsetzung zueinander vergrößert [Beobachtung]. Unter bestimmten Umständen gelingt dies auch einem expliziten dynamischen Läufer (z.B. einem Pferd), indem er sich springend fortbewegt. Hier treten aber extreme mechanische Belastungen auf. Zudem sind auch die zeitkritischen Anforderungen an ein solches System extrem hoch. Der dynamische Läufer ist in dieser Umgebung zudem benachteiligt, weil er nur begrenzt eine eingeleitete Aktion (Schritte) rückgängig machen bzw. austesten kann. Es muss vorher feststehen, dass die nächste Aktion nicht die Stabilität des Roboters gefährdet. Wie Tests mit QRTR BigDog [Bos10] zeigten, muss ein dynamischer Läufer in 200ms eine neue Fußposition finden, wenn ein Fuß angehoben wurde. Gelingt dies nicht, fällt er um. Bei der Firma Boston Dynamics konnte bei BigDog die Zeit auf 3-10 Sekunden verlängert werden, indem der ganze Körper gegen die Fallrichtung gesenkt wird. Dadurch kippte der Roboter aber nur langsamer. Dies ist kein stabiler Stand. Zudem können auch hier ausgeführte Aktionen erst korrigiert werden, wenn alle vier Füße wieder Bodenkontakt haben. Ein zurückziehen des Fußes ist nicht möglich, da der Roboter bedingt durch seinen hohen Schwerpunkt, sofort in die aktive Beinrichtung kippt.

Statischer / pseudodynamischer Läufer: Ein vierbeiniger statischer Läufer kann auf drei Beinen sicher stehen und sich bewegen, ohne zu kippen. Dies wird beispielsweise durch einen niedrigen Schwerpunkt und große Füße erreicht. Bei einem vierbeinigen Laufsystem mit einem Bein in Bewegung, entsteht ein sog. Schwerpunktdreieck (sicherer Bereich). Dieser sichere Bereich ist so groß, dass der Schwerpunkt des Roboters diesen niemals überschreitet [Ruh11] [Ruh121]. Die Steuerung eines statischen Läufers ist einfacher und nicht zeitkritisch. Es kann an jedem Punkt eines Bewegungsablaufs verharrt werden. Leider werden diese Vereinfachungen mit einem höheren Energiebedarf in den Beinen erkauft, da der Roboterkörper aktiv in die Bewegungsrichtung „gezogen“ werden muss. Weiterhin kann ein statischer Läufer nicht so schnell laufen wie ein dynamischer Läufer.

Ein *pseudodynamischer Läufer* vereint die Eigenschaften von statischen und dynamischen Läufern. Er hat einen geringeren Schwerpunkt und einen größeren sicheren Bereich, als ein dynamischer Läufer. Aber der pseudodynamische Läufer kippt ab einer fest definierten Beinauslenkung in Laufrichtung. Da der sichere Bereich nur minimal überschritten wird, kippt der Roboter sehr langsam. Dies ist zwar zeitkritisch, aber ein Steuer- / Regelsystem (Kontroller) hat durchschnittlich ein Zeitfenster von zwei bis fünfzehn Sekunden um einzugreifen. Zudem kann durch die Größe der Auslenkung (Schrittlänge) bestimmt werden, ob der Roboter überhaupt in den dynamischen Bereich wechselt. Ein wesentlicher Vorteil des

pseudodynamischen Läufers ist aber, dass Aktionen (Bewegungen) jederzeit reversibel sind. Damit kann beispielsweise ein Schritt getestet werden [Behauptung].

Zusammenfassung der Behauptung 1:

Damit ein vierbeiniger Roboter im Gelände sicher laufen und Hindernisse überwinden kann, scheint ein statischer / pseudodynamischer Läufer besser geeignet zu sein, als ein dynamischer Läufer. Vor allem könnte es vorteilhaft sein, den Boden vorher zu Testen und Aktionen rückgängig machen zu können. Ein Indiz für die obige Behauptung ist auch der Nachfolgetyp des BigDog (Abbildung 2-1). Der AlphaDog (LS3 [Bos13]) wirkt¹ wie ein pseudodynamischer Läufer. Der Schwerpunkt ist näher am Boden und die Beingeometrie wurde vergrößert.



Abbildung 2-1 Boston Dynamics BigDog (links), AlphaDog-LS3 (rechts) [Bos10] [Bos13]

Wie diese Behauptung beweisen?

Die unterschiedlichen Verhaltensweisen der Läuferarten scheinen sich überproportional mit der Größe des Roboters zu verstärken [Ale11] [And09] [Kol09]. Um die Behauptung 1 zu Beweisen oder zu widerlegen, muss an einem realen Roboter getestet werden. Roboter in dieser Größenklasse sind nicht auf dem freien Markt erwerbbar. Auch Simulationen sind hier nicht zielführend [Ruh121]. Es muss ein eigener vierbeiniger Roboter entwickelt und realisiert werden. Hier ist es der Roboter AMEE XW2.

¹ Da der LS3 eine Entwicklung für die US-Streitkräfte ist, werden kaum Informationen veröffentlicht. Der LS3 (AlphaDog) wird seit dem Sommer 2012 von einer Spezialeinheit und Boston Dynamics unter realen Bedingungen getestet [Bos13].

2.2 Behauptung 2

Die Auslagerung von zeitkritischen Tasks in physikalische Controller, beschleunigt die Entwicklung erheblich.

Warum?

Soweit bekannt, scheinen alle QRTRs ein monolithisches Softwarekonzept zu haben. Es wird mit Zeitfenstern und einem absoluten Timing in einem Realtime-OS gearbeitet [Ale11] [And09] [Kol09] [Shk10] [Reb08].

Da diese Roboter dynamische Läufer sind, ist ein exaktes / absolutes Timing zwingend erforderlich, wenn es nur einen zentralen Hauptcontroller gibt. Dies hat wiederum zwingend ein Realtime-OS zur Folge, da nur hiermit ein absolutes Timing garantiert werden kann. Bei den komplexen Abläufen in einem Laufsystem ist es wünschenswert, klar gegliederte und gekapselte Module zu implementieren. Dabei werden auch nicht zeitkritische Aufgaben, wie beispielsweise die Pfadplanung, den Zwängen der Timeslots unterworfen. Um aber dieses Timing einzuhalten, wird versucht Teile von teuren Berechnungen in andere Module auszulagern. Dies führt zu einem „verwaschenen“ System, das keine klaren und überschaubaren Module mehr hat². Es ist sehr zeitintensiv obige Systeme zu testen bzw. zu debuggen. Die schlussendliche Folge ist ein hoher und zeitintensiver Entwicklungsaufwand.

Was wäre aber, wenn zeitkritische Tasks dort bearbeitet werden wo sie auftreten? In vierbeinigen Robotern sind zeitkritische Aufgaben immer an die Beine gebunden. Die Schlussfolgerung hier ist, dass auch jedes Bein einen eigenen physikalischen Prozessor (z.B. eine MCU) erhält [Ruh11]. Damit ist es möglich, klare und gekapselte Module zu implementieren. Diese Prozessoren (Bein-Kontroller) kommunizieren über einen Hauptcontroller, um sich bei einigen Aktionen zu synchronisieren [Ruh11]. Im Hauptcontroller kann ein **nicht** Realtime-OS verwendet werden und die dortigen Module sind alle **nicht** zeitkritisch. Durch die starke Kapselung der Module bleibt die Übersicht erhalten und kann leichter implementiert werden. Dies sollte den Entwicklungsaufwand verringern und damit auch beschleunigen. Zudem sinkt die Komplexität der Module, was die Fehleranfälligkeit verringert.

Wie diese Behauptung beweisen?

Der absolute Beweis der Behauptung kann auch hier nur eine Realisierung erbringen. Dies

² Der Autor möchte hier kein Beispiel nennen. Diese Feststellung ist als Behauptung zu betrachten.

wäre zwar auch in einer Simulation möglich. Trotzdem würden immer Zweifel an der Realitätsnähe der Simulation bleiben.

3 Projektstand / Versuchsaufbau

Das Kapitel 2 zeigt deutlich, dass eine Realisierung des Roboters AMEE nicht nur für die obigen Behauptungen von Interesse ist. Die Realisierung kann auch Ergebnisse liefern die in anderen Themenbereichen neue Wege aufzeigen könnten. Beispielsweise könnte diese Arbeit ein besseres Verständnis für biologische Regelsysteme liefern. Wie bereits in den Veröffentlichungen zu Honda ASIMO [Hon07] als Beispiel erwähnt wurde, ist das Rückenmark mit seinen Reflexen äquivalent zu den Co-Prozessoren (bei AMEE, Bein-Kontroller) im System zu sehen. Diese Reflexe werden parallel in einem anderen „Systembereich“ des Körpers bearbeitet und laufen nicht über das Gehirn (Hauptkontroller). Dies ist natürlich eine starke Abstraktion, zeigt aber wie das System von biologischen Systemen inspiriert wird.

Aus diesem und obigen Gründen sehen wir den hohen Kosten- und Zeitaufwand für eine komplette Realisierung als gerechtfertigt.

3.1 Realisiert (Stand Feb. 2013)

Das Beinkontrollersystem ist fertig implementiert und getestet [Ruh11]. Das System soll auf eine leistungsfähigere MCU portiert werden. Die Konstruktion des Roboters ist abgeschlossen (Projekt 2). Die Fertigung der Mechanik begann im Februar 2013 und sollte im März 2013 abgeschlossen sein. Das Eventsystem für den Hauptkontroller, auf Basis des Microsoft CCS & DSS Frameworks [Bet10], ist auf Verwendbarkeit getestet [Bet10]. In einem Testaufbau war die Systemintegration von Event- und Echtzeitsystem erfolgreich (Projekt 1). Eine Bewegungssimulation mit SolidWorks® wurde fertiggestellt (Projekt 2). Eine Drehmomentsimulation wurde mit MatLab® realisiert [Ruh12].

3.2 Noch zu Realisieren

Die Portierung des Beinkontrollersystems auf eine ARM basierte MCU von ATMEL®. Das Erstellen eines Platinenlayouts für diese MCU und die externe Fertigung dieser. Die Leistungselektronik ist zu ca. 70% abgeschlossen und wird auch extern gefertigt. Die Fertigung des Roboters in der HAW-Zentralwerkstatt läuft. Der Anbau und die Anpassung von 3D Sensoren sind noch nicht realisiert. Für erste Tests wird hier eine Microsoft Kinect for Windows® verwendet. Die Auswahl und der Einbau der Stromversorgung (Akkus) sind noch offen. Als letzter Punkt fehlen noch die Beschaffung und der Einbau eines leistungsfähigen

Hauptrechners in den Roboter. Der Hauptpunkt der Master-Thesis, die Implementierung und Tests des Hauptkontrollers, ist noch offen und sollte erst nach der Realisierung der Mechanik beginnen. Hier liegen aber schon einige Codefragmente und Testcodes vor.

3.3 Roboter AMEE XW2



Abbildung 3-1 Rendering der CAD-Dateien, AMEE XW2

Der Roboter AMEE XW2³ wird in Abbildung 3-1 dargestellt. Die Abbildung zeigt den Roboter als „Rohkarosse“ ohne Sensoren, Verkleidung, Akkus und Hauptrechner. Der Roboter wird ca. 85kg mit Akkus wiegen. Jeder Antrieb (Typ XDW092001-01) verfügt über ein Drehmoment von 90Nm. Von diesen Antrieben werden 16 Stück verbaut, die von der Firma Ott-Antriebe dem Projekt-Team zum Selbstkostenpreis überlassen wurden. Damit ergibt sich für jedes Bein ein DOF 4 System. Theoretisch wäre eine Stromaufnahme von 9.500W möglich. Dies wird im Betrieb nicht erreicht, da nicht alle Antriebe gleichzeitig voll belastet werden. Das Drehmoment der Antriebe ist absichtlich überdimensioniert. Die Simulation [Ruh12] [Projekt1] ergab ein benötigtes, maximales Drehmoment von 70Nm. Der Roboter wird 1,1m lang und hat eine Schulterhöhe von maximal 1m. Als Winkelsensoren für die Glieder wurden Folienpotentiometer vom Typ *Sensofoil*[®] direkt in den Gelenken verwendet. Diese wurden von der Firma Hoffmann + Krippner GmbH speziell für das Projekt angefertigt und gestiftet. Als Größenvergleich wurde in Abbildung 3-1 ein Avatar mit einer Körpergröße von ca. 1,7m eingefügt.

³ Bedeutung der Abkürzung AMEE XW2: Autonomous, Mapping, Exploration, Evasion - Experimental, Walker, Version 1.2

4 Schwerpunkt der Master Thesis

Wie in Kapitel 2 und 3 erörtert, wird der Schwerpunkt der Master Thesis sich mit dem Hauptkontroller für einen QRTR beschäftigen. Eine konkrete Implementierung soll für den Roboter *AMEE XW2* realisiert werden, um die Behauptungen und Annahmen zu untermauern. Als Basis dienen hier die Konzepte aus den Arbeiten von Kolter [And09] [Kol09], Rebula + Team [Reb08] und vielen anderen Arbeiten aus dem Umfeld von Boston Dynamics und dem LittleDog [Ruh122], wie sie in den Arbeiten zu AW1 [Ruh121] und AW2 [Ruh122] beschrieben wurden. Hierzu werden die Konzepte des Deliberative und Reactive-Layer,

um die eigenen Konzepte erweitert und modifiziert. Diese Modifikationen betreffen hauptsächlich den verteilten Aufbau des Gesamtsystems, wie er in Abbildung 4-1 dargestellt wird und in vorherigen Arbeiten [Bet10] [Ruh121] [Ruh11] [Ruh122] beschrieben wurde.

Der Hauptkontroller kommuniziert mit den Beincontrollern nur über Events. Die eigentlichen Steuer- und Regelaufgaben bleiben vollständig in den Bein-Kontrollern (blau, Abbildung 4-1) gekapselt. Der Hauptkontroller kontrolliert nur die globalen Daten im Roboter, wie beispielweise die Lage des gesamten Roboters. Aufgrund dieser Daten werden neue Fußpositionen und Laufmuster errechnet.

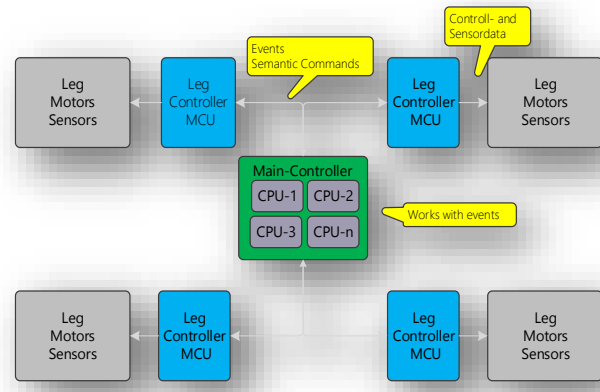


Abbildung 4-1 Systemüberblick AMEE XW2

4.1 Hauptkontroller (Main-Controller)

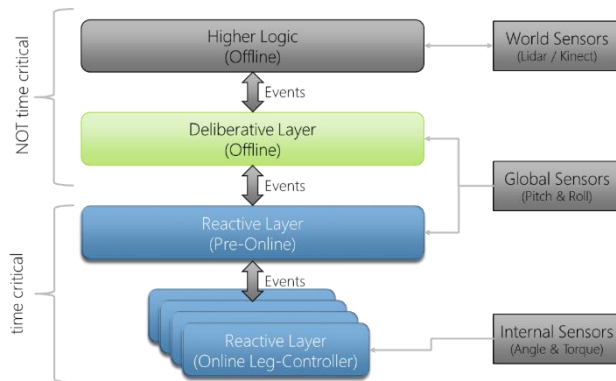


Abbildung 4-2 Layerübersicht des Hauptkontrollers

Der Hauptkontroller für das Laufsystem wird streng in drei Layer strukturiert. In Abbildung 4-2 wird eine Layerübersicht des Hauptkontrollers skizziert. Ein Merkmal dieser Modellierung ist die Teilung von zeitkritischen und nicht zeitkritischen Layern, die wiederum Module nach dieser Kategorisierung enthalten. Eine feinere Unterteilung wird nach Aufgaben getroffen. Hier wird in den Delinerative- und Reactive Layer

unterteilt. Der Reactive Layer ist nochmals in einen Pre-Online und Online Layer unterteilt. Die Layer sollen folgende Funktionen trennen.

4.1.1 Higher Logic-Layer

In dieser Darstellungsform sind hier alle höheren Funktionen zusammengefasst, die keine direkte Auswirkung auf das Laufsystem haben. Als höhere Funktionen werden beispielweise die Erfassung im Raum, die Pfadplanung und die KI angesehen. Hierbei handelt es sich nicht direkt um eine strenge Schicht, wie in den unteren Layern, sondern um lose gekoppelte Module. Es fließen Daten über die Umgebung des Roboters durch die *World Sensors* ein. Für die unteren Layer sind diese Daten weniger relevant. Im Higher Logic-Layer findet auch die Ermittlung einer Strategie zum Überwinden eines Hindernisses statt (optional für die Master Thesis).

4.1.2 Deliberative Layer

Dieser Layer kommuniziert nicht direkt mit den Beinen und wird deshalb auch als Offline Layer bezeichnet. In diesem Layer werden die Schritte und das Laufmuster geplant. Diese Planung umfasst aber nur eine Schrittfolge, also ein Bewegungsablauf pro Bein. Es werden, in schneller Folge, Schrittfolgen auf Durchführbarkeit geprüft. Diese Überprüfung beinhaltet die Stabilitätskontrolle und verwendet dafür die *globalen Sensoren* (z.B. die Neigung) des Roboters. Zudem wird auf eine mögliche Kollision der Beine untereinander geprüft. Wichtig

ist hierbei, dass keine festen Schrittmuster verwendet werden. Es wird mit einer Gewinnmaximierung gearbeitet. Gewinnmaximierung bedeutet hier, die maximal mögliche Bewegung des gesamten Roboters in Laufrichtung. Dies kann in maximale Schrittweiten zerlegt werden und die Auswahl des Beins erfolgt über das Maximum. Fehlerhafte Lösungen, die die Stabilität gefährden oder zu einer Kollision mit anderen Beinen führen, werden sofort verworfen und der Algorithmus beginnt von neuem mit einer geringeren Schrittweite. Erfolgreiche Lösungen werden zwischengespeichert und für den Reactive-Layer bereitgehalten. Dieser Layer wird zwar als planende Schicht bezeichnet, er plant aber mit aktuellen Daten aus den globalen Sensoren und reagiert damit nur auf die geänderten Daten. Damit könnte man behaupten, dass dieses Konzept kein Bruch mit der klassischen Subsumtion [Rod86] darstellt.

Die Modellierung wird mit einer beratenden KI erweitert, die gelernte erfolgreiche Schritte, diesem Layer als vorteilhaft vorschlägt. Diese KI wird im höheren Logic-Layer arbeiten. Dabei soll die maximale Schrittweite mit dem Vorschlag der KI verrechnet werden. Die Arbeiten an der KI werden von meinem Team-Kollegen Björn Bettzüge vorangetrieben und werden in seiner Arbeit genauer erläutert.

Geplante Module im Deliberative Layer: (Dargestellt in Abbildung 5-1 im Anhang C)

Die folgenden Module sind als einzelne sequentielle Automaten modelliert und stark durch die Arbeiten von Kolter [Kol09] und Rebula [Reb08] aus dem LittleDog [Bos10] Forschungsprojekt [Def05] geprägt. Der Higher Logic Layer übergibt die geplante Laufrichtung des Roboters an den Deliberative Layer. Das Modul *Gait Selection* wählt zwischen den beiden Laufmustern *Fast-Walk* [Ruh11] und *Save-Walk* [Ruh11] aus. Die Auswahl wird aus der Historie der vorherigen Schritte und der Empfehlung der KI ausgewählt. Hier scheint ein Schwellwert von drei Fehlritten ein gutes Kriterium zu sein, wann von schnellem Laufen auf die sichere Gangart mit nur einer Beinbewegung zurzeit umgeschaltet wird. Danach wird das entsprechende Beinpaar (bzw. Bein) im *Leg Selection* Modul ausgewählt. Dies erfolgt, wie oben beschrieben, über eine maximal mögliche Schrittweite. Vom Higher Logic-Layer wird hierfür ein Set mit sicheren Fußpositionen bereitgehalten. Dieses Set umfasst einen Radius von ca. 150cm um den 3D Sensor. Hierzu wurde im Higher Logic Layer eine 3D Punktwolke (z.B. aus einem Kinect Sensor) auf gerade Flächen untersucht (z.B. mit dem RANSAC Algorithmus [Yin10]). Zudem ist dieses Set schon von der KI auf besonders sichere Trittpositionen gefiltert worden. Die resultierende virtuelle Bewegung wird dem Modul *Body Shift* übergeben. Hier wird geprüft, ob der gesamte Roboter mit einer gemeinsamen Bewegung der Beine in Laufrichtung gezogen werden kann. Ist dies der Fall, weil alle Beine ihr Maximum beigetragen haben, werden diese geplanten Bewegungsdaten an das Modul *Step*

& *Body Plan* übergeben. In diesem Modul werden die Beinbewegungen virtuell ausgeführt und auf Kollisionen mit dem Torso und den Beinen untereinander geprüft. Dazu werden die Beine und der Torso des Roboters als Vektoren mit einer Hülle betrachtet. Wie hoch die Auflösung der Hülle und der Bewegung an sich sein wird, muss noch in Tests ermittelt werden. Denkbar wäre auch hier ein Eingreifen der beratenden KI. Ist der Bewegungsplan kollisionsfrei, werden diese Daten an das *Pitch & Roll Plan* Modul übergeben. Jetzt werden die Bewegungen mit den zurzeit anliegenden Neigungsdaten des Roboters geprüft. Es wird ein simples Schwerpunktmodell mit Vektoren betrachtet, welches ermittelt ob der Roboter bei einer geplanten Bewegung (in diesem realen Moment) kippen würde. Ist auch diese Prüfung erfolgreich, werden diese Trittpositionen für den Reactive Layer in einem Cache bereitgehalten. Nun beginnt der gesamte Ablauf erneut, um immer sichere und aktuelle Trittpositionen bereitzuhalten. Wird in irgendeinem Modul ein Fehler entdeckt, kann jedes Modul sofort den kompletten Ablauf neu anstoßen. Um dabei schnell zu einer sicheren Lösung zu gelangen, wird dann die maximale Schrittweite reduziert.

Ein Ziel der Realisierung, ist es hier eine Laufzeit von ca. 5ms über alle Module anzustreben. Umso mehr mögliche Schrittfolgen geprüft werden können, desto sicherer kann der Roboter im Gelände laufen [Behauptung]. Tests von Boston Dynamics [Bos10] und anderen Arbeiten zeigten: Eine grobgranulare aber schnelle Prüfung ist vorteilhafter als eine feingranulare aber langsame Prüfung. Da AMEE XW2 durchschnittlich 1-2 Sekunden gefahrlos kippen kann [Schätzung und Simulation], werden in dieser Zeit schon 200 bis 400 mögliche Aktionen geprüft die den Sturz abfangen könnten. Zusammenfassend kann dieser Layer als Brute Force charakterisiert werden. Die einzelnen Module in diesem Layer wurden schon in der Arbeit zu AW2 [Ruh122] genauer erläutert.

4.1.3 Reactive Layer (Pre Online & Online)

Erst dieser Layer kommuniziert direkt mit den Beinkontrollern. Durch die Auslagerung des Echtzeitanteils in die Beinkontroller, befindet sich im Hauptkontroller nur der Pre Online-Layer bzw. der Teil der für die Kommunikation nötig ist. Der restliche Online-Anteil des Reactive Layers ist in die Beinkontrollern implementiert und wurde ausführlich in der vorangegangenen Bachelor-Thesis [Ruh11] erläutert.

Der im Hauptkontroller verbleibende Teil, kontrolliert nochmals die reale Lage des Roboters über die Neigungssensoren und kann sofort eine sichere Trittposition aus dem Deliberative Layer abrufen. Die Modulübersicht wird in Abbildung 5-2 im Anhang dargestellt. In einem normalen Ablauf, ohne eine kritische Lage zu erreichen, ruft der Pre-Online Anteil

vorausberechnete Schrittpositionen aus dem Deliberative Layer ab und sendet diese an den Online Anteil in die Beincontroller. Dabei wartet er auf die Erfolgsmeldung, dass das Bein an dieser oder an einer leicht korrigierten Stelle platziert wurde. Durch die Ungenauigkeiten der Sensoren oder nachgebenden Boden werden die berechneten Positionen von den realen Positionen abweichen. Mit Sensoren in den Füßen des Roboters kann ermittelt werden, wann der Fuß den Boden real erreicht hat. Dieses Feedback ist ein Belastungsvektor mit der Verteilung des Gewichts auf den Fuß. Der Belastungsvektor wird an die KI übermittelt und bildet einen Teil des Belohnungs- und Bestrafungssystems des Lernalgorithmus [Bet12].

Sollte eine Bewegung zum Kippen des Roboters führen, ruft dieser sofort eine neue vorgehaltene Position aus dem Deliberative Layer ab. Da der Deliberative Layer ständig ca. 200 neue Positionen pro Sekunde ermittelt hat, besteht eine 99,8%ige Wahrscheinlichkeit [Shk101] [Shk10] [Reb08] dass schon eine Schrittposition berechnet wurde die den Sturz verhindert. Dieses Verhalten wird nicht nur für den Extremfall ausgenutzt, sondern wird aktiv genutzt, um mit einem mechanisch statischen Läufer das pseudo dynamische Verhalten zu erzeugen. Vereinfacht ausgedrückt: Hier wird der Roboter von der Beinmasse in Laufrichtung gekippt und der Sturz vom aktiven Bein abgefangen. Durch dieses Verhalten müssen auch keine sog. Ausfallschritte modelliert werden, da diese das zwangsläufige Resultat des oben beschriebenen Verhaltens sind.

Auch hier wurden die einzelnen Module schon in der AW2 [Ruh122] Arbeit beschrieben.

5 Risiken

- Das Hauptrisiko ist bei der Realisierung die Zeit. Jede kleine Störung gefährdet den Zeitplan. Abgemildert wird dieses Risiko durch ein hochmotiviertes beratendes Team von Experten aus der Informatik, Maschinenbau, Fertigung und Elektrotechnik. Damit sollte fast jedes auftretende Problem lösbar sein.
- Für einige Bewegungen ist es nötig, dass die Beinbewegungen synchron ausgeführt werden. Geplant ist hier eine Synchronisation über das Ethernet-Bussystem des Roboters. Ob die Laufzeiten klein genug sind, ist nur theoretisch berechnet worden. Schlägt dies fehl, könnte der Roboter nicht gerade laufen. Als Alternative wurde aber schon eine extra Synchronisationsleitung vorgesehen, die sternförmig die Bein-Kontroller (MCUs) direkt verbindet.
- Es könnte Probleme mit der Laufzeit im Deliberative Layer geben, wenn die Glieder zu feingranular modelliert werden müssen. Die Überschlagsrechnungen gehen von einfachen Vektoren mit einer Hülle aus, um beispielsweise eine Kollision im Voraus zu erkennen. Reicht diese Abstraktion nicht aus, wird sich die Laufzeit erhöhen. Um dieses Problem abzufangen, wird hier einfach auf mehr Rechenleistung des Hauptrechners⁴ gesetzt.
- Problematisch könnte auch die Fertigung der angepassten Elektronik und Leistungselektronik sein. Hier werden heute Bauteilgrößen verwendet, die keiner im Team manuell fertigen kann (löten). Hier hoffen wir auf eine kostengünstige Lösung durch den Auftragsfertiger der rohen Platine (Bestückungsservice). Das Problem könnte mit erheblich kostenintensiveren Fertigteilen gemildert werden. Leider werden diese Fertiglösungen nicht die geplante nötige Leistung erbringen. Es müssen Abstriche in der Leistungsfähigkeit des Roboters gemacht werden.
- Die in den zitierten Arbeiten vorgestellten Konzepte könnten nicht mehr auf einen Roboter in dieser Größe zutreffen. Dann können nur neue, komplett eigene Konzepte entwickelt werden. Dies ist für den Notfall mit zwei zusätzlichen Semestern (ab jetzt Feb. 2013) eingeplant.

⁴ In vielen nordamerikanischen Ansätzen nicht unüblich. Aufgrund der hohen Kapazität der geplanten Akkus, kann auch ein leistungsstarker Serverprozessor verwendet werden.

Literaturverzeichnis

- [Ale11] **Alexander Shkolnik, Michael Levashov, Ian R. Manchester and Russ Tedrake. 2011.** *Bounding on Rough Terrain with the LittleDog Robot.* [PDF] MASSACHUSETTS,USA : MASSACHUSETTS INST OF TECH, 2011. DOI: 10.1177/0278364910388315.
- [Bet12] **Bettzüge, 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.
- [Bos13] **Boston Dynamics. 2013.** Boston Dynamics, The LS3. [Online] Boston Dynamics, 02 09, 2013. [Cited: 02 09, 2013.] http://www.boston-dynamics.com/robot_ls3.html.
- [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>.
- [Def05] **Defense Advanced Research Projects Agency. 2005.** Learning Locomotion. USA : Government USA, 2005.
- [Hon07] **Honda Motor Co., Ltd. September 2007.** *ASIMO Technical Information.* [PDF] s.l. : Honda Motor Co., Ltd., September 2007. kA.
- [And09] **Kolter, J. Zico, Andrew, Y. and Youngjun, Kim. 2009.** *Stereo Vision and Terrain Modeling for Quadruped Robots.* [PDF] CA 94305, USA : Stanford University, Stanford, 2009.
- [Kol09] **Kolter, J. Zico, Rodgers, Mike P. and Andrew, Y. 2009.** *A Control Architecture for Quadruped Locomotion Over Rough Terrain.* [PDF] CA 94305 USA : Computer Science Department, Stanford University, Stanford, 2009.
- [Mur08] **Murphy, Robin R., et al. 2008.** Search and Rescue Robotics. [book auth.] Bruno Siciliano and Khatib Oussama. *Springer Handbook of Robotics.* Berlin : Springer, 2008, pp. 1151-1173.
- [Nac10] **Nachtigall, Werner. 2010.** *Bionik als Wissenschaft.* Heidelberg : Springer-Verlag, 2010. ISBN 978-3-642-10319-3 .

- [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.
- [Rod86] **Rodney A., Brooks. 1986.** *A robust layered control system for a mobile robot*. Massachusetts, USA : Journal of IEEE, 1986. ISSN: 0882-4967.
- [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, Juni 02, 2011. Bachelor Arbeit.
- [Ruh122] —. **2012.** *Quadruped Rough Terrain Robot, Controller Concepts, related Works AW2*. [PDF] Hamburg, Germany : HAW-Hamburg, Department Informatik, 08 30, 2012.
- [Ruh12] —. **2012.** *Simulation der Drehmomente in einem Roboter Bein, Ausarbeitung Modellierung Technische Systeme* . HAW-Hamburg : s.n., 2012.
- [Shk10] **Shkolnik, Alexander, et al. 12.2010.** *Bounding on Rough Terrain with the LittleDog Robot*. [PDF] MASSACHUSETTS USA : MASSACHUSETTS INST OF TECH , 12.2010. DOI: 10.1177/0278364910388315 The International Journal of Robotics Research.
- [Shk101] **Shkolnik, Alexander, et al. 2010.** *Motion Planning for Bounding on Rough Terrain with the LittleDog Robot*. [PDF] Cambridge, MA 02139, USA : Computer Science and Artificial Intelligence Lab, MIT, 2010.
- [Yin10] **Ying Yang, Michael and Förstner, Wolfgang. 2010.** *Plane Detection in Point Cloud Data*. Bonn : University Bonn, 2010. TR-IGG-P2010-01.

Anhang

Module im Deliberative Layer

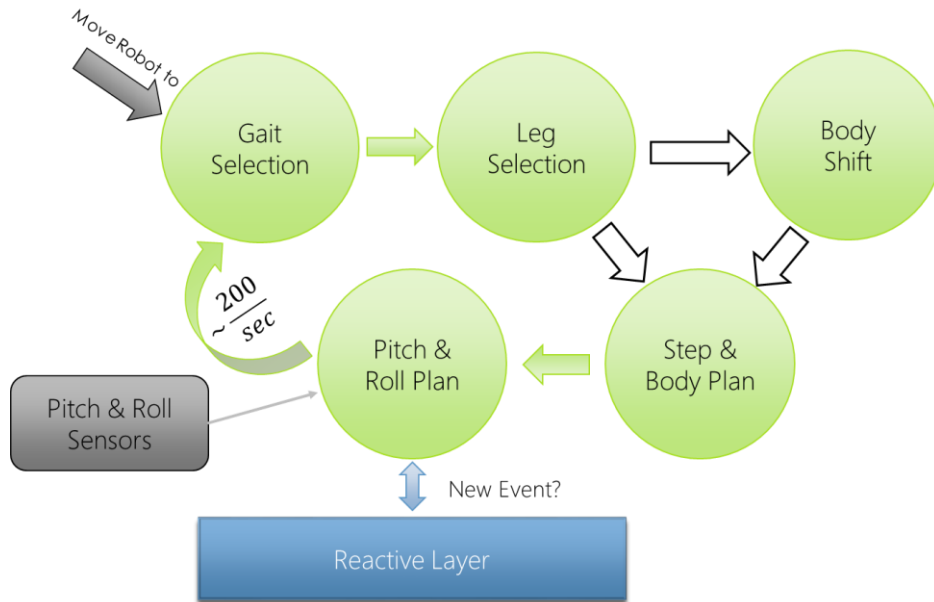


Abbildung 5-1 Module im Deliberative Layer

Module im Reactive Layer

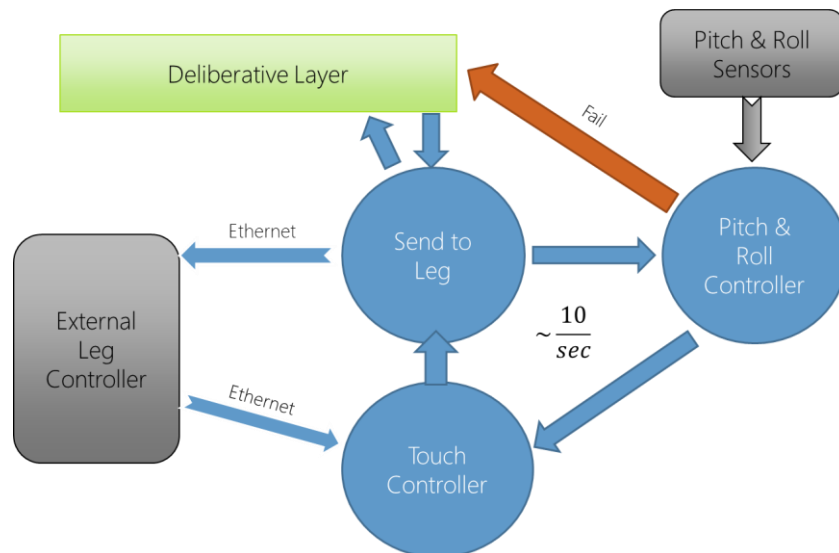


Abbildung 5-2 Module im Reactive Layer

Trends - Meinung:

Das Vorgehen, der Verteilung über spezialisierte Prozessoren, ist genau der gegenteilige Trend zur Automobilbauindustrie. Hier wird seit Jahren ein monolithisches System angestrebt, um die Kosten zu senken. Diese Kostensenkung wird mit immer höherem Entwicklungsaufwand erkaufte. Auch die Fehleranfälligkeit steigt seit Jahren bei Neuwagen (Quelle ADAC). Da aber in den letzten 2-3 Jahren erhebliche Sprünge in der Rechenleistung bei low-cost MCUs (z.B. ARM Kerne) erzielt wurden, bezweifle ich die Fortführung des Trends. Ich sehe unseren Ansatz als sicherere und kostengünstigere Alternative zur heutigen Praxis. (Heutige Fahrzeuge sind mit modernen Robotern vergleichbar.)

Sponsoring

Dieses Projekt wird gefördert durch:

HAW Hamburg

Hoffmann + Krippner GmbH

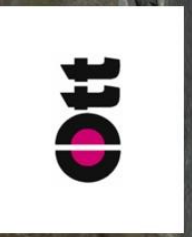
Ott Antriebe GmbH

Microsoft Corp. & Microsoft Deutschland GmbH

Meinen Vater, meinen Mentor Gunter Klemke und Björn Bettzüge meinen Team-Kollegen.
Danke!



Project AMEE



AMEE XW2

(C) 2013 Jan Ruhnke