Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

"Von Reis und Robbies"

Konzeptionierung eines Labors für Künstliche Intelligenz und Robotik

Studienarbeit zu diesem Thema

vorgelegt von

Bjørn Jensen[1]

betreut durch

Prof.Dr. Kai von Luck[2]

1 Matrikelnr.: 1563531, Kontakt via email: mirou@gmx.de
2 Kontakt via email: luck@informatik.haw-hamburg.de

Für Corina, meine Familie und meine Freunde

Für die Energie und Unterstützung, die sie mir gegeben haben

Inhaltsverzeichnis

Vorwort

Als ich im Sommer dieses Jahres aus einer kreativen Auszeit in Salzburg zurück nach Hamburg gekommen bin, lief mir dort ein wohl vertrauter Professor über den Weg: Kai von Luck. Da es früh am Tage war und sowohl er als auch ich offensichtlich noch nicht gefrühstückt hatten (war unsere persönliche Habe bei der zweiten Begegnung an diesem Tage um einiges Frühstücksutensil bereichert), beschlossen wir, uns bei einem gemeinsamen Frühstück zu unterhalten.
Tenor eben jenes Frühstücks war u.a. sowohl das Voranschreiten meines Studiums als auch meine Pläne für die Zukunft. Da ich nach meinem Abschluss an der HAW an der Universität Sinologie und Philosophie studieren möchte, suchten wir gemeinsam nach einer Möglichkeit, diesen Themenkomplex miteinander zu verbinden. So kamen auch die Studien- und die Diplomarbeit zur Sprache. Ein Thema für meine Studienarbeit hatte ich schon in Arbeit, denn da ich im Rahmen des Seminars für Angewandte Informatik einen Vortrag über Quantenkryptographie gehalten hatte, beschloss ich, mich mit diesem Thema weiterhin zu befassen. Doch der Vorschlag, den Kai mir offerierte, bewog mich dazu, begonnene Arbeit liegenzulassen, um ein komplett anderes Thema in Angriff zu nehmen. Dieses Thema ist äußerst interdisziplinär und stellt auf Grund dieser Komplexität eine große Herausforderung dar. Doch worum geht es eigentlich?

Einleitung

Die Hochschule für Angewandte Wissenschaften Hamburg (HAW)[3] hat mehrere Partnerhochschulen, die über den gesamten Globus verteilt sind. Eine dieser Hochschulen ist die University of Shanghai for Science and Technology of Shanghai (USST)[4]. Nun gibt es diverse Abkommen / Vereinbarungen mit den Partnerhochschulen der HAW. Eine dieser Vereinbarungen besagt, dass sich jede dieser Hochschulen zu einem Tausch des Lehrdeputats verpflichtet, d.h. dass mindestens einmal im Jahr ein Kontingent an Lehrkörpern aus Shanghai nach Hamburg kommt und umgekehrt. Kurz bevor ich mit Kai das Thema der Arbeit fixierte, war ein Professor aus Shanghai vor Ort, der sich u.a. die Veranstaltungen über Künstliche Intelligenz incl. zugehöriger Praktika zu Gemüte führte. Dabei wurde ihm auch ein neues Labor am Fachbereich Elektrotechnik / Informatik präsentiert: Das „Robot Building Laboratory“.
Das Wesen dieses Labors ist die praktische Umsetzung der Kenntnisse aus den Lehrveranstaltungen zur Künstlichen Intelligenz und das Heranführen an die Implementierung (teil-)autonomer Systeme.
Besagter Professor war von diesem Labor und den Veranstaltungen sehr angetan und erzählte Kai und Gunter, dass er sich solch ein Labor auch an der USST vorstellen könnte. Und genau hier greift meine Diplomarbeit. Ich beschäftige mich in dieser Arbeit mit der Konzeptionierung eines Labors für Künstliche Intelligenz und Robotik, welches an die USST

3 In Zukunft wird, wenn von der Hochschule für Angewandte Wissenschaften gesprochen wird, nur noch die Abkürzung HAW verwendet werden.
4 Auch diese Hochschule wird in Zukunft nur noch mit ihrem Kürzel USST angesprochen werden.

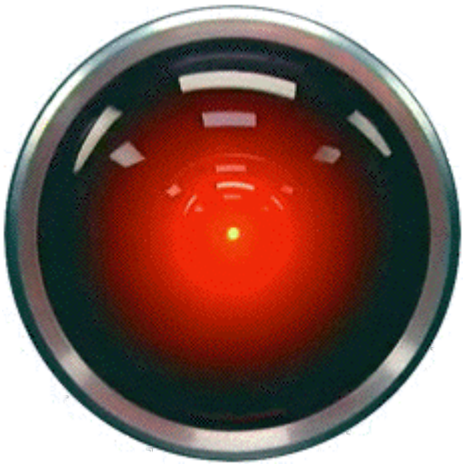
angepasst ist. Mitte Dezember wird dieses Konzept dann einer Delegation chinesischer Professoren und Dekanen präsentiert werden. In dem Fall, dass diese den Sinn und die Notwendigkeit für dieses Labor ebenso einsehen, wie man es an der HAW macht, und die Investition somit für sinnvoll erachtet, wird dann die Realisierung dieses Konzeptes Gegenstand meiner Diplomarbeit sein.

Vision

Wenn man nun an die Planung einer solchen Einrichtung herangeht, dann sollte man sich gerade in Anbetracht der Komplexität der Aufgabe gewisse Einschränkungen, was das technische Detail angeht, hinnehmen. Die Verwirklichung eigener Vorstellungen und Ideale in diesem Kontext sollte somit auf ein Minimum beschränkt werden.
Ich habe für meine Person folgende Dinge festgelegt, an deren Umsetzung ich weitestgehend festhalten möchte.

- Vier Computerarbeitsplätze
- Ein Werkstattteil
- Vier Roboter, die sich seitens der Anschaffungskosten im unteren Investitionsniveau befinden
- Verwendung einer Hochsprache in Bezug auf die Implementierung einer (vermeintlichen) Intelligenz der Roboter
- Ziel einer Veranstaltung: Realisierung fußballspielender Roboter der Junior-League mit einer Teamstärke von 1:1
- Ausschließliche Verwendung von freier Software und OpenSource

“Ich kriege schon Kopfschmerzen, wenn ich bloß versuche, mich auf euer Niveau runterzudenken.” (Marvin, paranoider Androide in "Per Anhalter durch die Galaxis" von Douglas Adams)



Künstliche Intelligenz

Doch warum ist der Themenkomplex "Künstliche Intelligenz" so prädestiniert? Was ist das Besondere daran und macht ihn von anderen Themenkomplexen der Informatik so verschieden, dass er eben jene exponierte Stellung genießt, in die er durch das besondere Augenmerk meiner Arbeit gehoben worden ist?

Die "Künstliche Intelligenz" (im Folgenden auch KI) zeichnet sich durch ihre starke Interdisziplinarität aus, denn neben der Informatik sind auch die Psychologie, Neurologie, Linguistik, Soziologie und Philosophie daran beteiligt. Ihr zum Gegenstand gereicht die Untersuchung intelligenten Verhaltens und dessen Implementierung auf Computern zum besseren Verständnis intelligenten Verhaltens von Lebewesen und von Ingenieursseite her die Konstruktion von Hard- und Softwaresystemen, die intelligentes Verhalten aufweisen.

Beginnen wir mit der wohl bekanntesten Definition von "Künstliche Intelligenz":

"Künstliche Intelligenz ist die Kunst Maschinen zu schaffen, die Aufgaben lösen, zu deren Lösung Intelligenz notwendig ist, wenn sie von Menschen ausgeführt werden."[5] [Kurzweil1993]
Schon beim ersten Hinsehen erkennt man, dass diese Definition Begrifflichkeiten in zur Klärung eines Begriffes nutzt, die durch die Definition erklärt werden sollen. Es ist einleuchtend, was sich hinter der Künstlichkeit in dem Begriff "Künstliche Intelligenz" verbirgt: Die Bildung von Artefakten zur Bewältigung von vorgegebenen Aufgaben- bzw. Problemstellungen.
Der Begriff der Intelligenz wird hingegen nur durch sich selbst erklärt. Jochen Heinsohn und Rolf Socher-Ambrosius nennen jedoch die Dinge, in denen man sich einig ist, dass sie intelligentes Verhalten ausmachen [Schneider2000]:

- Lernfähigkeit
- Fähigkeit zu logischen Schlussfolgerungen
- Planungsfähigkeit
- Problemlösungsfähigkeit
- motorische Intelligenz

Es gibt zwei Hauptansätze der KI:

- Symbolverarbeitung
- Subsymbolischer Ansatz

5 Diesen Wortlaut bzw. einen ähnlichen kann man den meisten Lehrbüchern zu diesem Thema entnehmen. Eine Gegendarstellung lautet: "Künstliche Dummheit (KD) kann als der Versuch von Computerwissenschaftlern angesehen werden, Computerprogramme zu entwerfen, die in der Lage sind, die Art von Problemen zu erzeugen, die normalerweise in der menschlichen Denkweise begründet sind." [Marsha1987]

Symbolischer Ansatz

Dies ist der klassische Ansatz der KI. Das Wissen wird hier mit symbolischen Strukturen dargestellt. Wichtigster Repräsentationsformalismus ist die Sprache der Prädikatenlogik [s.a. Schöning2000]. Mit diesem Ansatz lassen sich die Fähigkeiten zu logischen Schlussfolgerungen und die Planungsfähigkeit modellieren.

Subsymbolischer Ansatz

Bei diesem Ansatz lassen sich Wissenseinheiten (Begriffe oder Konzepte) nicht in einzelnen Symbolen lokalisieren, sondern nur über ein gesamtes Netz verteilt. Das bekannteste Konzept zur subsymbolischen KI ist das der neuronalen Netze. Dieser Ansatz betont die evolutionäre Entwicklung der natürlichen Intelligenz und versucht diese nachzuahmen. Auf diese Weise lassen sich Lernfähigkeit und motorische Intelligenz modellieren [Haykin1999].

Der kritischen Betrachtung Motivation

Wenn man sich mit dem Thema “Künstliche Intelligenz” und der Schöpfung derselbigen befasst, sollte man sich grundsätzlich die Frage stellen, welche Beweggründe hinter der Anforderung von KI stecken und welche Auswirkungen die Schaffung einer solchen auf Mensch und Gesellschaft haben kann.

Hypothetisch gesprochen ist der Einsatz von KI nichts anderes als der Ersatz von menschlicher Produktivität unter Aufwand von Zeit und Geld. Es wird also, im Positiven gesehen, die Zeit und das Können eines Menschen als derart kostbar angesehen, dass man diesem zur Steigerung der Produktivität, quasi zur Unterstützung, ein System zur Seite stellt, welches in gewissen Maße über eine Art von KI verfügt.
Negativ gesehen wird das Individuum nur als Kostenfaktor gesehen, welcher mit zunehmender Arbeitszeit fehleranfälliger wird. Eine Maschine wird zwar auch mit der Zeit anfälliger für Fehler, jedoch ist die Zeitspanne theoretisch größer als beim Menschen. Zudem sind die fortlaufenden Kosten für einen Menschen evtl. höher als die sporadisch bzw. einmalig anfallenden Kosten eines Roboters. Das wiederum könnte bedeuten, dass zunehmende Entwicklung Ursache sein kann für ein größeres Maß an Arbeitslosigkeit, wobei der KI bzw. den damit verbundenen Forschungseinrichtungen keine Pauschalschuld angelastet werden soll. Der Mensch als solches wäre also ersetzbar, der Wert des Individuums gleich seiner Produktivität im Verhältnis zu den individuell verursachten Kosten.
Auch ist die Frage zu stellen, wofür ein entwickeltes KI-System eingesetzt wird. Es stellt sich also nicht die Frage, wen es ersetzt, sondern wem es schaden könnte. Bespielsweise kann man hier bekannte Einsatzgebiete, die in höchstem Maße fraglich erscheinen, nennen: Der militärische Einsatz oder der Einsatz von KI im Walfang.

Als reines Forschungsgebiet ist die KI eines der vielfältigsten und interessantesten. Jedoch sollte jeder, der sich damit ernsthaft beschäftigt, die Fragen nach dem Sinn und dem Hintergrund stellen. So ist der Mensch am Ende doch sich selbst der ärgste Feind[6].
Die Zeichen der Zeit stehen auf Sturm, wenn man sich das Bild der Gesellschaft und die selbstvergessene Position der Wissenschaft in eben jener vor Augen führt. Leistung bestimmt den Status quo, wobei Leistung sich als maximale Produktivität unter minimalem Zeitaufwand zu verstehen ist. Qualität verliert das Element der Güte und die Sichtweise auch und gerade eben jener Gesellschaftsschicht, die der kritischen Sichtweise auf Grund des Bildungsstandes her verpflichtet ist, tritt wissentlich aus ihrem kritischem Element heraus und beugt sich der Wirtschaft. Wie sonst ist es möglich, dass Wissenschaft unter dem Gesichtspunkt der Wirtschaft beurteilt wird und es keinen Widerstand gibt, der jedoch hierbei absolut notwendig ist (s.a. [Adorno2001]).

Zu guter Letzt sollen die Phantasien der Verschwörungstheoretiker auch nicht vorenthalten werden. Gemäß dem Fall, dass die Rekonstruktion Mensch gelingt, die Ratio obliegt und der Eros verkümmert, so stellt sich die Frage, ob Humunculus [Goethe1998] nicht der Meinung sein kann, das emotionale Original auf Grund seiner emotionsbedingten Fehleranfälligkeit in Bezug auf rationale Entscheidungen zu eliminieren. Hollywood bedient sich dieses Themas nur zu gern und schickt u.a. seinen frisch gewählten Gouverneur in den Krieg mit den Maschinen[7].

Jedoch sind auch andere Sichtweisen auf die hier angesprochenen Artefakte geläufig. In fernöstlichen Regionen wird es gemeinhin so gesehen, dass jedem Gegenstand eine Art Seele innewohnt. So auch dem Gegenstand dieser Diskussion. Diese Sichtweise impliziert, dass man in dem Roboter eher eine Art Freund & Helfer sieht und führt zu einer wesentlich schnelleren Integration in den menschlichen Alltag und einer höheren, widerspruchfreieren Akzeptanz.

6 Anspielung auf das Zitat von Platon, welches Thomas Hobbes in *Leviathan* verwendet: Dem Sinn nach bedeutet es, dass der Mensch dem Menschen ein Feind ist, was ja, wenn man sich evtl. unmittelbar bevorstehender negativer Konsequenzen des Einsatzes von KI bewusst ist und deren Einsatz trotzdem zulässt, auch untermauert scheint.

7 Man schaue sich die Produkte der Unterhaltungsindustrie einmal genauer an: 2001 – Odyssee im Weltraum, Terminator-Trilogie, Matrix-Trilogie u.a. In ihnen ist das Feindbild die Maschine. Ähnliches Verhalten Mitte der 1990er Jahre in der Spieleindustrie: Das Feindbild Mensch in Egoshootern und militärischen Strategiespielen muss durch Roboter und/oder andere Nichthumanoide Lebensformen ersetzt werden.

Corpus delicti humunculus...

Der Einsatz von kybernetischen Konstrukten, also Robotern, ist aus diversen Gründen zur idealen Einführung in verschiedene Bereiche der Künstlichen Intelligenz geeignet. Jedoch muss man sich auch der Grenzen der Rekonstruktion Mensch bewusst sein. Der Mensch zeichnet sich nicht nur durch reine Funktion, sondern u.a. auch durch Emotion und Intuition aus.
Es wird möglich, alles der Ratio Entsprungene auf ein oder mehrere Artefakte abzubilden, doch der Mythos des Eros wird der Maschinenwelt weitestgehend verschlossen bleiben. Selbst wenn der Humunculus ein vermeintlich emotionales Verhalten an den Tag legt, so kann dies ebenso wie intuitives Verhalten nur approximiert werden. Dies begründet sich darin, dass die Ratio den Gesetzen der Logik wie auch immer gehorcht, der Eros sich diesen jedoch nicht unterwirft. Intuition kann als Erfahrungsschatz verstanden werden, von daher ist es für einen Algorithmus sicherlich einfacher, eine einmal gemachte Erfahrung auf einen ähnlichen Fall anzuwenden, doch inwiefern die angewandte Erfahrung redefiniert werden kann, ist fraglich.
Der Einsatz von Robotern ist jedoch unter dem Gesichtspunkt der Gefahrenprävention oder dem Einsatz in für den Menschen unerreichbare Gebiete wie große Meerestiefen oder andere Planeten (Mars) durchaus zu befürworten und hat hier einen deutlich positiven Nutzen für die Gesellschaft. Jedenfalls könnte das der Fall sein, wenn die Auswertung dessen, was ein Roboter erkundet und übermittelt, erfolgreich und verwertbar ist. Doch dies ist rein spekulativ und bewegt sich höchstgradig im Konjunktiv.
Wenn man jedoch ein wenig genauer hinsieht, kommt man nicht umhin, dass sich die These „Sie sind unter uns“ aufdrängt, erfährt doch, wenn auch nicht als Humunculus, sondern in Form einer nicht jedermann zugänglichen Sichtbarkeit, der Einsatz von intelligenten Artefakten eine gesellschaftlich hohe Akzeptanz, jedoch eher implizit denn explizit: Sie kommen bspw. in Kraftfahrzeugen (BMW etc.) oder Flugzeugen (Airbus etc.) seit Jahren zum Einsatz. Und jedermann ist z.B. über die reibungslose Funktionalität des in seinem Pkw befindlichen ABS durchaus erfreut.
Doch rücken wir den Gegenstand des Kapitels in den Vordergrund: den Roboter. Die Entwicklung von Robotern gehört in das Gebiet der Robotik, wobei die Robotik zwar eng mit der Künstlichen Intelligenz verbunden ist, jedoch keinen Teilbereich der KI darstellt.

Roboterkategorien

Um auch in dem Gebiet der Kybernetik granularer zu werden, gibt es auch hier diverse Kategorien von Robotern, die sich grundsätzlich voneinander unterscheiden:

- Manueller Manipulator
- Roboter mit festem Aktionsablauf
- Roboter mit variablem Aktionsablauf
- Playback-Roboter
- Roboter mit numerischer Steuerung
- Autonome und Intelligente Roboter

Nachfolgend sollen kurz die signifikanten Merkmale der unterschiedlichen Kategorien vorgestellt werden, um deutlich zu machen, welche Kategorie für das von mir betrachtete Fallbeispiel in Frage kommt. Tiefergehende Erläuterungen sind auch [Nehmzow2002] und [Dudek2000] zu entnehmen.

Manueller Manipulator

Unter einem manuellem Manipulator versteht man eine Vorrichtung mit mehreren Freiheitsgraden, wobei Freiheitsgrad eine Einschränkung der Bewegungsfreiheit der Vorrichtung bedeuten, die von einem Bediener bewegt wird. Ein Beispiel hierfür, was jeder kennt, sind die Automaten auf dem Jahrmarkt, aus denen man mittels einer Greifvorrichtung einen Gewinn angeln kann. Hierzu muss die Greifvorrichtung über dem Objekt der Begierde positioniert und ausgelöst werden. Einmal ausgelöst, greift diese Vorrichtung zu und fährt zur Ausgangsposition zurück. Dort wird das gegriffene Objekt in eine dem Spieler zugängige Vorrichtung fallen gelassen.

Roboter mit festem Aktionsablauf

Im Gegensatz zum manuellen Manipulator handelt dieser Manipulator nach einer festen Methodik, mittels der eine bestimmte Aufgabe bewältigt werden kann. Diese Methodik ist jedoch fest vorgegeben und somit nur schwer zu manipulieren.

Roboter mit variablem Aktionsablauf

Diese sind dem Wesen der zweiten Kategorie sehr ähnlich. Der einzige Unterschied ist der, dass Teile der Methodik leicht modifiziert werden können.

Playback-Roboter

Hierbei handelt es sich im Prinzip um ein Konglomerat der Kategorien 1 und 3. Es ist absolut notwendig, dass ein Bediener dem Roboter in der Initialisierungsphase, also im ersten Lauf, manuell den Weg zur Bewältigung seiner Aufgabe vermittelt. Der Roboter zeichnet diesen Weg auf, um anhand dessen ab Beendigung der Initialisierung die Aufgaben autonom zu bewältigen.

Roboter mit numerischer Steuerung

Hier kommt eine Form der Programmierung zum Zuge. Anstatt mit dem Roboter die Lösung einer Aufgabe schrittweise durchzugehen, entwirft der Bediener ein Programm, in dem alles an Informationen vorhanden ist, was der Roboter zur Bewältigung der Aufgabe benötigt. Dieses Programm wird dann im Roboter installiert.

Autonome und Intelligente Roboter

Hier handelt es sich um Roboter, die in der Lage sind, ihre Umgebung zu verstehen und Veränderungen in dieser zu erfassen und trotz dieser Veränderungen die Aufgabe, die es zu bewältigen gilt, erfolgreich zu lösen. Die Unterscheidung zwischen Autonomie und Intelligenz ist hier, dass ein autonomes System zwar in der Lage ist, seine Umwelt zu erfassen, jedoch nur unter der Prämisse, dass es eine Veränderung in dieser Umwelt nicht wirklich wahrnimmt und darauf im Vorfeld reagiert, sondern wie bspw. der Staubsauger „Roomba“ erst gegen einen Gegenstand fahren muss, um diesen wahrzunehmen. Intelligente Systeme erweitern also die Autonomität.

Umgebungen

Hat man eben eine Kategorisierung der Roboter kennengelernt, so lässt sich auch die Umgebung, in der ein Roboter agieren muss, kategorisieren. Hier sollen jedoch nur drei dieser Kategorien vorgestellt werden:

- modifizierte, strukturierte Umgebungen
- unmodifizierte, semistrukturierte Umgebungen
- unmodifizierte Umgebungen

Modifizierte, strukturierte Umgebungen

Es handelt sich bei dieser Form der Umgebung um eine solche, die in keiner Weise natürlich ist. Der Bediener gibt hier die Grenzen und das Wesen vor.

Unmodifizierte, semistrukturierte Umgebungen

Bei dieser Form der Umgebung ist eine Approximation der natürlichen Umwelt Gegenstand. Die Einschränkung erfolgt lediglich minimal durch Markierungen (bspw. durch Sensorik), Induktionsschleifen o.ä.

Unmodifizierte Umgebungen

Eine Abbildung der Natur an sich, denn es gibt hier keine Einschränkungen und kann als schwierigste Voraussetzung gesehen werden. Hier hat der Roboter nicht nur mit der Bewältigung der Aufgabe zu tun, sondern auch mit der Wahrnehmung seiner Umgebung und mit auftretenden Veränderungen.

Bei den für das Fußballspiel verwendeten Robotern handelt es sich unter Berücksichtigung der vorgestellten Kategorien also um autonome (ergo intelligente), mobile Roboter, die in unmodifizierten, teilstrukturierten Umgebungen agieren müssen.

Kritisch betrachtet muss man jedoch sagen, dass es unmodifizierte Umgebungen im eigentlichen Sinne auf unserem Heimatplaneten durch GPS nicht mehr gibt. Einzige Ausnahmen sind hier Meerestiefen, in denen ein GPS-Signal nicht mehr registriert werden kann.

Das “Robot Building Laboratory” der HAW Hamburg

Die Hochschule für Angewandte Wissenschaften Hamburg verfügt seit 1996 über ein eigenes Labor, das der Forschung und Entwicklung im Bereich “Künstliche Intelligenz” und “Robotik” verschrieben ist: das “Robot Building Laboratory”.

In Anbetracht der vorhandenen Räumlichkeiten stellt sich die Frage, ob diese Investition denn nötig gewesen sei, sind doch Labore für die Softwaretechnik vorhanden und die Benutzung eben solcher zur Implementierung der Intelligenz eines Robots gängige Praxis.

Doch ist die Praxis nicht unbedingt vorteilhaft. In den Räumen des Softwarelabors herrscht eine hohe Fluktuation an Studenten, Professoren, akad. Mitarbeitern und anderen Personen. Dieses gereicht nicht nur der Konzentration an einem Robotprojekt Arbeitenden zum Nachteil, es stellt auch ein hohes Risiko dar, denn die Anschaffung eines Roboters ist nicht gerade günstig und somit ist man ständig darum bemüht, einen Blick auf diese Investition zu haben. Auch dieses beeinträchtigt die Aufmerksamkeit der Involvierten und kann die praktische Umsetzung der erlernten Kenntnisse aus den begleitenden Vorlesungen verlangsamen.

Vorteile eines Labors für KI und Robotik

Ein solches Labor hat jedoch auch immense Vorteile. Es bietet die Möglichkeit, zum einen die Kenntnisse aus den Vorlesungen praktisch umzusetzen und Erfahrung darin zu sammeln, inwieweit Theorie und Praxis voneinander abweichen können, bzw. was neben aller Theorie noch zu berücksichtigen ist. Auch kann sich der Dozent voll auf die Studenten konzentrieren, ohne die o.g. Beeinträchtigung.

Die im Kapitel über KI vorgestellte Interdisziplinarität kann hier voll zum Zuge kommen. Es ergeben sich hier die Möglichkeiten, ungestört die Hintergründe eines Problems oder einer Aufgabe auch aus dem Blickwinkel der anderen Wissenschaften zu sehen. Zumindest sollte dies von Zeit zu Zeit der Fall sein.

Es ist auch von großem pädagogischem Wert, dass die Studenten das in den begleitenden Veranstaltungen erlernte Wissen simultan in die Praxis umsetzen können. Es hat zu dem auch großen pädagogischen Wert, bekommen doch die Studenten ad hoc ein Gefühl des Erfolgs vermittelt und die handwerklichen Fähigkeiten werden weiter ausgebaut.
Evtl. auftretende Problematik bei Umsetzung des Erlernten sorgt für eine Förderung der Kreativität, denn der Ehrgeiz zum Erreichen eines (Teil-) Ziels ist geweckt. Da die meisten dieser Arbeiten im Team stattfinden, fördert die Arbeit in einem solchen Labor die Team- und Kommunikationsfähigkeit.

Auch ist das Labor eine Erleichterung der Wissensvermittlung durch den Dozenten,denn durch regelmäßige Termine in den Räumen des Labors und das Betreuen der Studenten kann dieser den Themenkomplex wesentlich konkreter und nicht auf dem hohen Abstraktionsniveau der Vorlesung an den Studenten heranführen. Es handelt sich dabei sozusagen um “Wissenschaft zum Anfassen”.

Zu guter Letzt versetzt die aktive Mitarbeit in dem Komplex “KI & Robotik” den Studierenden in die Situation, aktiv über die Problematiken der jeweiligen Aufgabenstellungen nachzudenken. Dabei erschließen sich den Studierenden noch andere Erkenntnisse als bei der reinen Nachbereitung einer Vorlesung. Wie (kleinere) Teams erfolgreich und produktiv zusammenarbeiten, entnehme man [Beck2000].

In seiner Studienarbeit unterteilt Matthias Stolt die Motivation für ein derartiges Labor in zwei Bereiche [Stolt2001]. So gibt es zum Einen die Gruppe der pädagogisch/didaktischen, zum Anderen die Gruppe der technischen Motivationen, wobei die erste in der Robotik eher ein Mittel zum Zweck sieht, zweite jedoch ein Reihe von Möglichkeiten bietet, das Spektrum der Möglichkeiten des Baus von Robotern und der Berührung informatikrelevanter Themen kennenzulernen. Stolt stellt folgende Unterteilungen vor:

Pädagogisch/didaktische Motivationen

Aus diesem Bereich stellt Stolt fünf Punkte vor, deren Inhalt durch solch ein Labor abgedeckt werden.

Steigerung der Teamfähigkeit

Da in den vorlesungsbegleitenden Veranstaltungen grundsätzlich in einer Gruppe von mind. zwei Personen zusammengearbeitet wird, ist es nur allzu offensichtlich, dass die Teilnahme an einem solchen Projekt die Fähigkeit der Zusammenarbeit mit anderen Personen mit sich bringt. Diese Teamfähigkeit wird den sog. 'soft skills' zugeordnet und gehört zu den Fähigkeiten, die in der heutigen Zeit von jeder Person verlangt werden. In den meisten Vorlesungen und Praktika wird diese Fähigkeit jedoch nicht gefördert, was zum Teil laut Stolt in der Art der Aufgabenstellung bzw. der mangelnden Komplexität und somit schlechten Teilbarkeit der typischen Aufgaben begründet ist.

Alternative Lernmethodik

In vielen Veranstaltungen wird eine Lernmethodik verwendet, die als 'telling and testing' von Stolt bezeichnet wird. Hierbei kommt es zu einer Sequenz von Vermittlung der Lehrstoffs und der Überprüfung/Umsetzung desselbigen. Dies verhindert bzw. behindert jedoch die explorative Wissensermittlung durch eigenständige Untersuchungen und Überprüfungen.
In den Praktika zur Ki und Robotik wird jedoch diese zweite, als 'exploration and construction' bezeichnete Methode angewandt.

Steigerung der Kreativität

Auch steigert die Arbeit in einem Robotik-Projekt die Kreativität der Teilnehmer. Anders als in den meisten Veranstaltungen gibt es hier nämlich nicht nur Fragestellungen, die mit nur einer Antwort geklärt sind, sondern auch Probleme, die mehrere Lösungen akzeptieren. Dies hat eine kreative und kritische Auseinandersetzung mit einem Problem zu Folge, welche durchaus auch in der normalen Arbeits- und Umwelt auftreten können.

Steigerung der Weltwahrnehmung

Die Ingenieurs- und Naturwissenschaften arbeiten in der Regel immer mit Modellen der realen Welt. In der Regel sind dies abstrakte, also stark vereinfachte Modelle, ähnlich der aus Platons Höhlengleichnis.
In der Robotik werden diese Modelle wieder sehr deutlich mit der realen Welt in Kontakt oder auch in Konflikt gebracht. Der Teilnehmer an einem Robotikprojekt erfährt dadurch, dass die Welt der Theorien sich nicht unbedingt einfach auf die der realen Bedingungen übertragen lässt.

Steigerung der Designfähigkeiten

Das Design kann je nach Aufgabenstellung recht vielfältig sein, gleich den Aufgaben eines Ingenieurs in der Berufswelt. Die Kursteilnehmer können in dem Labor die Möglichkeiten des Designs ausprobieren und Vor- bzw- Nachteile eines jeweiligen Designs direkt erfahren. Hierbei ist der Student dazu verpflichtet, sein Design entweder vollständig zu planen oder sich quasi evolutionär einer Lösung zu nähern.
Die Erfahrung hat gezeigt, das die am häufigsten genutzte Variante der Lösung die evolutionäre Annäherung ist.

Informatikrelevante Motivationen

Stolt zeigt in seiner Arbeit jedoch nicht nur die Motivationen der erstgenannten Gruppe auf, sondern beschreibt auch viele Elemente der zweiten Gruppe.

Imperative Programmierung

Da die meisten der heute eingesetzten Programmiersprachen auf der imperative Programmierung beruhen, also auf Programmen, die auf den Elementen Anweisung, Variable und bedingte Verzweigung beruhen, können mit diesen Grundlagen schon einfache Robotermodelle und Aufgabenstellungen programmiert werden. Das bedeutet, dass die Roboterprogrammierung auch schon für Einstiegskurse in die Programmierung genutzt werden kann. Alle Konzepte der imperativen Programmierung können so in einem stark motivierendem Umfeld erlernt werden.

Programmiermethodik

Auch die Programmiermethodik kann durch Teilnahme an einem Robotikprojekt verbessert werden. Duch die mögliche Komplexität der Aufgabe ist der Teilnehmer dazu aufgefordert, Methodiken der strukturierten Analyse anzuwenden. Dabei werden Probleme in Teilaufgaben und deren Beziehungen unterteilt. Diese werden dann weiterbearbeitet. Im Kontext der Robotik ergeben sich recht häufig Teilprobleme, z.B. der Bewegunsapparat, die Zielfindung, die Wegplanung und die Manipulatorsteuerung.
In direktem Zusammenhang mit der strukturierten Analyse steht die strukturierte Programmierung. Hier werden alle Programmkonstrukte sauber geschachtelt zusammengefügt werden.
Eine andere Ebene der strukturierten Programmierung ist die der Datenstrukturen bzw. noch weitergehend die Ebene der abstrakten Datentypen, die letztlich in der Objektorientierten Programmierung mündet.
Auch können moderne Methoden der Programmierung erlernt werden, beispielsweise aus dem Bereich des Softwareengineering das *Extreme Programming* [Beck2000], einer stark teamorientierten Methode.

Parallele Abläufe

Viele Aufgaben können oder sollten in einem Roboter parallel verarbeitet werden. So gibt es beispielsweise eine Architektur für Programmierung, in der ein Task Sensordaten übermittelt und vorverarbeitet, ein anderer Task die Reaktion auf die vorverarbeiteten Daten veranlaßt und ein weiterer Task die übergeordnete Planung übernimmt. In anderen Architekturen laufen parallel verschiedene Verhaltensweisen ab, die sich unabhängig um den Zugriff auf Resourcen bewerben.
Die Kommunikation zwischen diesen Tasks, der Ausschluß von Blockaden und Kontrolle über Resourcen sind hier Thema der Programmierung und eine stark erweiterte Herausforderung gegenüber der Programmierung 'einfacher' einzeln ablaufender Programme.

Meß-, Steuer- und Regelungstechnik

Da die Roboter natürlicherweise als Teil eines Regelkreises agieren – jeder ernstzunehmende Roboter reagiert auf seine Umwelt und manipuliert sie, ist also ein Glied eines Regelkreises -, kann man die Steuer- und Regelungstechnik hier sehr praxisnah einsetzen. Allerdings ist es nicht notwendig, zunächst einen theoretischen unterbau zu schaffen, denn die absolut notwendigen Regelungen und deren Parameter sind auch durch Experimente ermittelbar bzw. sie sind mit Hilfe des 'gesunden' Menschenverstandes herleitbar. Wenn es aber gewünscht ist, so kann die ganze Tiefe der Steuer- und Regelungstechnik ausgelotet werden. Ebenso intensiv kann man sich mit Sensorik auseinandersetzen. Über die Filtertechniken und differentielle Messungen hin zu elektronischen Schaltungen für besondere Vorverarbeitungen und Aufbereitung der Signale. Ebenfalls ein Thema der Regelungstechnik, das mit Robotern bearbeitet werden kann, ist die Fuzzy-Logic-Programmierung.

Echtzeitbedingungen

Roboter können auch sehr gut als Echtzeitprobleme verstanden und behandelt werden. Typischerweise muß ein Roboter in Echtzeit auf eine aufgetretenes Signal reagieren. Als Beispiel sei hier ein Roboter angeführt, der sich frei auf einer Tischplatte bewegen soll, ohne vom tisch zu herunterzufallen. Die Signale der Sensorik, die die Tischkante erkennt, müssen in Echtzeit verarbeitet werden, d.h. es muß innerhalb einer harten Zeitgrenze eine Reaktion der Steuerung der Mototren erfolgen, sonst fällt der Roboter vom Tisch und die Aufgabe ist nicht gelöst.

behavior based / subsumption architecture

Speziell für die Bedürfnisse der Programmierung von Robotern, die sich in einer realen Umwelt bewegen sollen, wurde das 'behavior based programming' bzw. die 'subsumption architecture' entwickelt. Hier werden unterschiedliche Verhaltensweisen entworfen und programmiert. Diese werden dann konkurrierend parallel betrieben. Über Prioritäten und ide Anknüpfung von Verhalten an externe Signale wird nun das Verhalten ausgewählt, das den Roboter kontrollieren soll. Diese Art der Programmierung/Systembetrachtung ist dem Behaviorismus entlehnt. Sehr wichtig ist hier, dass nicht ein Modell von Verhalten im Ganzen entworfen wird,sondern nur kleine einfache Verhaltensweisen und die jeweilige Notwendigkeit für das Auftreten einer dieser Verhaltensweisen. Dies dient ebenfalls der Reduzierung von Komplexität eines Gesamtverhaltens auf kleine beherrschbare Verhaltensweisen. Es muß hierbei eine Regelung des Zugriffes auf beschränkte Resurcen (z.B. den Motorantrieb) eingeführt werden, damit sich konkurrierende Verhaltensweisen (Angriff und Flucht) nicht gegenseitig blockieren [Brooks1991].

Reaktive Systeme

Die formalen Modelle für reaktive Systeme nach der 'subsumption architecture' sind Zustandsautomaten nach Mealy und Moore. Hier werden spezielle Zustände definiert und Übergänge zwischen diesen Zuständen, die auf Grund von (externen oder internen) Signalen ausgelößt werden. Außerdem werden Aktionen definiert, die bei Betreten oder Verlassen eines Zustandes oder bei Durchführen eines Überganges ausgeführt werden. Die Automatentheorie der theoretischen Informatik beschreibt die Zustandsautomaten vollständig und gibt Methoden an die Hand, um in solchen Automaten z.B. Verklemmungen zu entdecken oder die Automaten zu optimieren. Von Harel stammt eine Erweiterung von hierarchischen Zustandautomaten, die besonders zur Darstellung (und Implementierung) von reaktiven Systemen geeignet ist. Die Harel-Automaten sind auch Bestandteil der Unified Modelling Language (UML), jedoch sind diese dort leicht modifiziert.

Künstliche Intelligenz und Bildverarbeitung

Die KI hält in Robotern im Zusammenhang mit höheren Funktionen Einzug. Hierzu zählen vor allem Planungsstrategien für das Erreichen von Zielen. Ebenfalls spielt in der Robotik das Thema Bildverarbeitung und -erkennung sowie der Bereich der kognitiven Systeme eine große Rolle. Weitere Schlagwörter dazu sind Neuronale Netze, Selbstlernende Systeme, Evolutionäre Programme und Genetische Algorithmen.

Braitenberg Vehikel

Ein interessantes Experiment aus der experimentellen Psychologie kann mit den Robotern ebenfalls durchgeführt werden. Es handelt sich um die sogenannten Braitenberg Vehikel. Dies sind einfache Wagen mit einem Differenzantrieb und einfachen Sensoren, die mit einfachen internen Verschaltungen (Programmierungen) sehr interessante Verhaltensweisen an den Tag Legen. Die Analyse bzw. die Synthetisierung der gewünschten Verhaltensweisen ist sehr spannend, da aus sehr einfachen inneren Strukturen ein sehr komplexes Verhalten entstehen kann. Die Experimente mit den Braitenberg Vehikeln eignen sich auch als Einführung in dynamische Systeme.

Multiagenten Systeme

Aus dem kooperativen Zusammenspiel mehrerer Roboter ergeben sich Szenarien von Multiagenten Systemen. Hier ist die Ebene der Kommunikationsprotokolle interessant, ebenso wie verteilte Planunsstrategien oder die Robustheit gegen Kommunikationsfehler (Problem der Byzantischen Generäle).

Mindestanforderungen an ein Robot-Laboratory

Zu Anfang der Arbeit wurden schon ein paar Gesichtspunkte genannt, an denen ich festhalten möchte, wenn es um die Realisierung eines Labors für KI und Robotik geht.

Dabei sollte folgendes gewährleistet sein:

- Die Arbeitsmöglichkeit für bis zu 8 Studenten mit der Option auf Erweiterung
- Labor und Werkstattteil
- Möglichst lokale & regionale Unabhängigkeit der Realisierung
- Verwendung von Low - Cost - Roboterbaukästen
- Implementierung der Intelligenz in einer Hochsprache
- Die Möglichkeit der Realisierung fußballspielender Roboter der Junior-League mit einer Teamstärke von 1:1
- Ausschließliche Verwendung von freier Software und OpenSource

Das Labor & die Werkstatt

In Hinsicht auf die Infrastruktur bedeutet das die Anschaffung bzw. Bereitstellung folgender Komponenten bzw. Räumlichkeiten:

Ein Raum mit einer Grundfläche von mindestens 35 m^2 für das Labor und einen Raum mit einer Grundfläche von mindestens 5 m^2 für den Werkstattteil. Der Werkstattteil muss auf jeden Fall vorhanden sein, damit die Roboter um Sensorik und andere Technologie, bei Verwendung eines neuen Boards oder der Erforschung neuer Technologie problemlos erweitert werden können.
Im Prinzip genügt es, einen Arbeitsplatz für 2 Personen in der Werkstatt bereitzustellen, da die Erfahrung gezeigt hat, dass meist nicht mehr Studierende zeitgleich darin arbeiten.

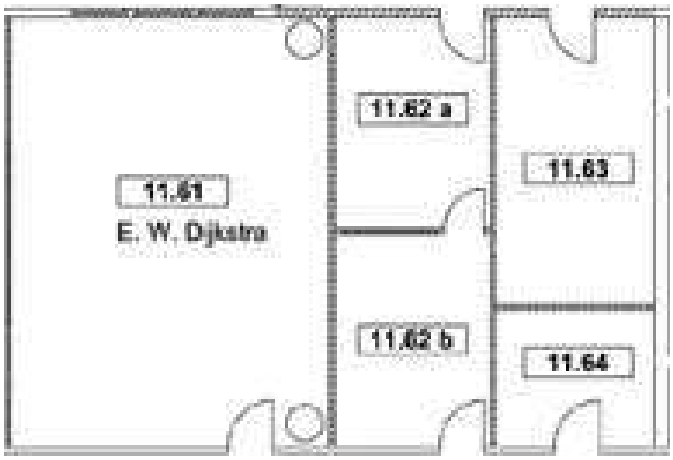


Abbildung 1: Robot Building Laboratory der HAW Hamburg

Zur Erläuterung:

Raum 11.61 “E.W.Dijkstra”[8] ist das Labor, Raum 11.64 die Werkstatt.

8 Edsger Wybe Dijkstra, niederländischer Informatiker, geboren 11. Mai 1930 in Rotterdam, gestorben am 6. August 2002, in Neunen (Niederlande).

Dijkstra studierte Mathematik und theoretische Physik an der Universität Leiden. Von 1952 bis 1962 arbeitete er als Programmierer am »Mathematisch Centrum« (heute Centrum voor Wiskunde en Informatica) in Amsterdam. Danach wurde er Mathematikprofessor an der T.H. Eindhoven, 1984 wechselte er auf den Schlumberger Centennial Chair in Computer Sciences an der Universität von Texas in Austin.

Zu seinen Beiträgen zur Informatik gehören Dijkstras Algorithmus zur Berechnung des kürzesten Weges in einem Graphen und seine Abhandlungen über den Goto-Befehl und warum er nicht benutzt werden soll.

Im Jahre 1972 erhielt Dijkstra den Turing-Preis.

“E.W.Dijkstra” 11.61 – Das Labor

Da das, was sich dem Besucher des *Robot Building Laboratory* präsentiert, in Worten wesentlich schwerer darzustellen ist als in Bildern, soll an dieser Stelle das Labor in Bildern präsentiert werden.
Die Computerarbeitsplätze sind gegenüber der Eingangstür sowohl zentral als auch an der gegenüberliegenden Außenwand und der rechten Wand angesiedelt.

Abbildung 2: Computerarbeitsplätze im Robot Building Laboratory in der Mitte des Raumes

Abbildung 3: Computerarbeitsplätze im Robot Building Laboratory an der rechten Außenwand

Neben den Computerarbeitsplätzen, deren Mindestanforderungen ich noch vorstellen werde, gibt es die Möglichkeit, strukturierte Umgebungen in dem Labor zu realisieren, ohne dass diese bei Beendigung eines Praktikums wieder abgebaut werden müssen[9]. Möglichkeiten der Umgebungsrealisierung und des Stauraumes für Roboter und Umgebungsgegenstände können hier betrachtet werden. Sie liegen linker und rechter Hand des Eingangsbereiches.

9 Was jedoch nicht zu pauschalisieren ist, denn immerhin hat auch diese Räumlichkeit nur begrenzt Platz zur Verfügung. Wenn dieses Platzangebot ausgereizt ist, muss selbstverständlich die eine oder andere Umgebung demontiert werden. Eine Unterbringung der Umgebungsgegenstände in dem Labor ist jedoch möglich (unter den bekannten Voraussetzungen).

Umgebungen

Abbildung 4: Die Fußballumgebung

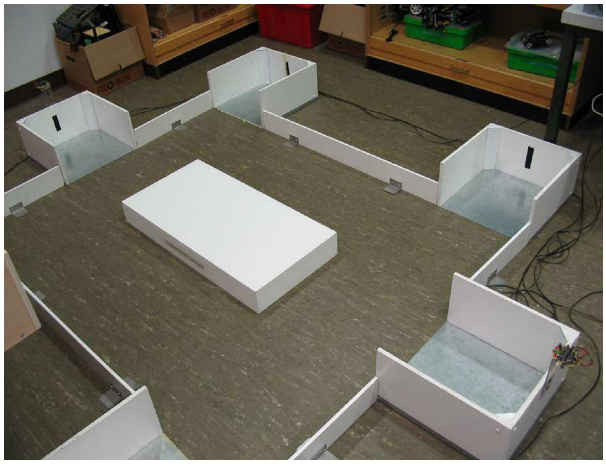


Abbildung 5: Eine weitere Umgebung, bei der ein Robot seine zugehörige "Garage" finden und nutzen muss

Stauraum

Abbildung 6: Diverse Unterbringungsmöglichkeiten im Robot Building Laboratory

Die Werkstatt

Die Werkstatt dient im Prinzip nur der Modifikation bzw. Reparatur der Roboter oder der involvierten Technologie (Sensorik, Mechanik). Auch in diesem Raum sind Werkzeuge, Arbeitsflächen und Stauraum vorhanden. Da jedoch diese Räumlichkeit eher selten und wenn, dann von maximal zwei Studierenden genutzt wird, fällt auch die Ausstattung des Raumes geringer aus.

Um konstruktiv und effektiv zu arbeiten, sind folgende Ausrüstungsgegenstände mindestens nötig:

- Ein Arbeitstisch 2m x 0,8m
- Ein Werkzeugsatz (div. Schraubendreher, Zangen...)
- Ein Lötkolben mit regelbarer Temperatur
- Ein Miniaturschraubstock
- Eine Lampe mit integrierter Lupe
- Ein Zwei-Strahl-Speicher-Oszilloskop
- Ein Multimeter
- Ein Trenntrafo
- Ein regelbares Netzteil
- Ersatzteilvorrat

Die Werkstatt sieht in der HAW wie folgt aus:

Abbildung 7: Einsichten in die Werkstatt des Robot Building Laboratory

Leider sind in der Werkstatt des *Robot Building Laboratory* nicht alle von mir geforderten Ausrüstungsgegenstände vorhanden. Aber letztendlich muss das Institut über die Ausrüstung entscheiden.

Der Computerarbeitsplatz

Bei den Computerarbeitsplätzen handelt es sich im Grunde um ganz “normale” PCs. Damit das Arbeiten den Studierenden auch ein wenig Spaß macht, gibt es aber auch an diese PCs gewisse Mindestanforderungen, denn jeder, der schon einmal mit einer JAVA-Entwicklungsumgebung gearbeitet hat, und das auf einem Rechner mit wenig Arbeitsspeicher, weiß, wie unlustig und frustrierend so etwas sein kann. Und Spaß ist nun einmal ein sehr motivierender und nicht zu vernachlässigender Faktor.

Hier also die Mindestanforderungen an einen dieser Comuterarbeitsplätze:

- 256 MByte RAM, eher 512 MByte
- CPU mit 533 MHz, je schneller, desto besser
- 20 GByte HD
- 17” -Monitor
- CD-ROM
- keine besondere Grafikkarte

Wenn man aktuelle (Stand: November 2003) Preislisten durchgeht, bieten viele Distributoren Komplettsysteme an, die diesen Mindestanforderungen bei weitem Genügen. Eine in Hamburg ansässige Firma verlangt für ein System mit folgender Ausstattung:

- 256 MByte DDR-RAM
- CPU mit 2,0 GHz
- 40 GByte HD
- CD-ROM
- 17”-Monitor

insgesamt 408,- Euro ohne Betriebssystem. Gemäß der von mir veranschlagten Mindestanforderungen benötigt man 4 Rechner dieser Art und würde Rabatt bekommen.

Das Betriebssystem

Wie schon in der Darstellung der Vision angekündigt, präferiere ich den Einsatz von OpenSource und freier Software. Damit liegt es auf der Hand, als Betriebssystem Linux zu verwenden. Da ich am vertrautesten mit der SuSE-Distribution bin, würde ich diese auch in ihrer neuesten Version (9.0)[10] einsetzen.

Nun gibt es jedoch ein Problem: Wenn man die Mindstorm-Entwicklungsumgebung "RobotLab" installieren möchte, erkennt man, dass diese eine Windows-Variante zum Betrieb benötigt.

Doch die Lösung dieses Problem ist nicht weiter schwierig: Es gibt unter Linux einiges an Software, die ein Windows-Betriebssystem emulieren kann.

Gängige Beispiele hierfür sind:

- Wine
- Win4Lin
- WineRack (nur SuSE 9.0)

Somit ist gewährleistet, dass auch die Nutzung von Windows-kompatiblen Programmen unter Linux möglich ist.

JAVA

Wie bereits erwähnt, soll zur Implementierung der Logik und (vermeintlichen) Intelligenz eine Hochsprache verwendet werden. Dies hat mehrere Gründe. Um den StudentInnen, die in einem solchen Labor ihrem Studieninhalt nachgehen (oder diesen vielleicht sogar zu ihrer Leidenschaft entwickeln), nicht nur ein Werkzeug zur Umsetzung der Vorlesungsinhalte einer Veranstaltung zum Thema „Künstliche Intelligenz" oder „Robotik" , sondern auch Methodik und Kenntnis aus dem Themenkomplex der Programmierung und des Software Engineering u.a. zu vermitteln. Und um diesen Bereich auch zu nutzen, habe ich eine Hochsprache gewählt, die dieses Unterfangen bestmöglich unterstützt: JAVA.

JAVA eignet sich auf Grund folgender Eigenschaften hervorragend zur Implementierung und zum Erlernen bzw. Vertiefen von Programmierfertigkeit[11]:

- strenge Objekt-Orientierung[12]
- Eliminierung von nicht-Objekt-orientierten und problematischen Features (Makros, Header-Dateien, globale Variablen und Funktionen,...)
- keine Zeiger

10 In Informatikerkreisen ist es eine unausgesprochene Regel, niemals eine x.0-Version einer SuSE–Distribution zu verwenden. Da ich dieses Betriebssystem jedoch selbst verwende und damit gute Erfahrungen gemacht habe, setze ich mich bewusst über diese Regel hinweg.

11 Hauptsächlich wurde hier der Vergleich JAVA vs. C++ betrachtet.

12 In diesem Hinblick wäre SMALLTALK noch besser geeignet, da es sich hierbei zu 100% um Objekt-Orientierung handelt.

- Garbage-Collection
- Typ-Sicherheit
- Runtime-Checks (Überprüfung von Feldgrenzen, Nullpointer, Type-Castings,...)
- Multi-Threading
- verbessertes Exception-Handling
- keine Mehrfachvererbung

Zudem hat JAVA seit seiner Geburtsstunde bzw. seit der Lizenzierung durch Netscape 1995 eine sehr starke Verbreitung erfahren. So gibt es allein im World Wide Web etliche Foren, die sich nur mit JAVA beschäftigen. Ebenso sind unter diesen eine Vielzahl, die sich dem Einsatz von JAVA in Bezug auf die Robotik verschrieben haben[13].

Die Entwicklungsumgebung

JAVA als Paradigma, so meine Vorstellung. Wenn man sich jedoch auf dem Markt umsieht, so gibt es diverse Entwicklungsumgebungen, die einem das Blaue vom Himmel versprechen, doch entweder sind diese kommerziell und entfallen der Auswahl durch meine eingangs erwähnte Beschränkung oder sie sind bei weitem nicht so komfortabel wie man es als Softwareentwickler gern hätte.

Seit dem letzten Jahr macht jedoch ein Projekt diverser kommerzieller Firmen von sich reden. Diese haben sich zusammengetan, um eine Entwicklungsumgebung gemeinsam zu entwickeln und gratis zur Verfügung zu stellen: Eclipse.

Eclipse ist eine Entwicklungsumgebung, die durch die Möglichkeit der individuellen Erweiterung über eine Vielzahl von eben solchen verfügt und durch den Komfort, den man sonst von JBuilder oder anderen großen kommerziellen Entwicklungsumgebungen gewohnt ist, zu einem Werkzeug herangewachsen ist, welches man nicht ignorieren sollte.

Für dieses Projekt kommt es uneingeschränkt in Frage, da es sogar für Eclipse schon Erweiterungen für die Programmierung der Lego© Mindstorm™ gibt.[14]

13 Siehe auch hierzu die Links zu Lego & Robotik

14 Wer sich über das Eclipse-Projekt informieren will, besuche bitte: http://www.eclipse.org

Dokumentation

Ein leidiges Thema der Entwickler ist die Dokumentation. Softwareentwickler berufen sich dabei meist auf die Kommentare in ihrem Sourcecode, die jedoch auch sehr gern vernachlässigt werden. Andere wiederum schwören auf JavaDoc etc.

Letztendlich ist es eine Glaubensfrage. Den Kommentar im Sourcecode einer Dokumentation gleichzusetzen, ist ein absolut unzumutbarer Fehler, denn niemand wird sich die Mühe machen, bei einer Unmenge von produziertem Code die Kommentare zu suchen, was schon voraussetzt, dass der Leser weiß, wie ein Kommentar aussieht.

Die beiden meist vertretenen Varianten sind meiner Meinung nach die Benutzung einer gängigen Office-Software oder LaTeX.

Für LaTeX gibt es diverse Programme unter Linux. Die Verwendung von gängiger Office-Software (gängig heißt hier kompatibel mit Microsoft Office) bedeutet nicht unbedingt eine große Investition. Denn auch hier gibt es ein Projekt, welches durchaus interessant ist und eine Alternative zu MS Office darstellt, ist es doch trotz Verwendung eines eigenen Standards kompatibel zu den von MS Office produzierten Dateien: Das Open-Office-Projekt[15].

Dieses Office-Paket liegt im Internet gratis zum Download in verschiedenen Sprachen bereit.

Roboterfamilien

Mehrere Möglichkeiten der Realisierung eines in einem Robotikprojekt entstandenen Roboters sind verbreitet. Davon kommen sehr viele auf Grund des finanziellen Aufwandes für ein solches Labor nicht in Frage, es sei denn, der Geldgeber verfügt über den nötigen finanziellen Hintergrund.

Es werden im folgenden zwei weit verbreitete Familien aus der unteren Preisklasse vorgestellt und ein paar Alternativen dazu genannt. In diesen Roboterfamilien werden Bausteine eingesetzt, mit denen viele Menschen schon in frühester Kindheit Kontakt hatten: Lego und Fischertechnik.

15 Hierzu kann folgende Internetseite besucht werden: http://www.openoffice.org

Lego© Mindstorm™

Warum die Verwendung von Lego© Mindstorm™? Nun, damit hat es folgende Bewandtnis: Mit Lego© assoziieren die meisten Personen Erinnerungen an ihre Kindheit und insofern keine negativen Erfahrungen mit Lego© gemacht worden sind, erinnert sich jeder gerne an diese Zeit.

Aber es hat auch einen praktischen Nutzen. Lego ist weit verbreitet und es gibt diverse Möglichkeiten der Erweiterung. Auch die Modifizierung durch Eigenkreation ist mit minimalem Aufwand durchaus möglich.

Lego© Mindstorm™ ist so etwas wie eine Erweiterung der Lego© Technik™-Palette um die Intelligenz. Doch ist dies keineswegs Spielzeug im reinen Sinne, denn die Roboter sind zum Experimentieren weit verbreitet. Eine Möglichkeit des Einsatzes sehen wir in der folgenden Vorstellung:

Jitter[16]

Jitter ist ein Roboter, der im Kontext eines 2001 von den Firmen LEGO, Siemens, Hitachi und Intospace ausgeschriebenen Wettbewerbs entstanden ist. Dabei wurden die Vorgaben gemacht, dass ausschließlich LEGO-Teile Verwendung finden sollten, ein Maximalegewicht von 1500 g und eine maximale Größe von 300 mm^3 nicht überschritten werden durfte und der Roboter eine autonome Arbeitsweise vorweisen sollte.
Jitter hat diesen Wettbewerb gewonnen und ist zur Internationalen Raumstation ISS geflogen, wo er dann zum Einsatz gekommen ist. Seine Aufgabe war es, kleine herumfliegende Teile einzusammeln. Hierzu musste Jitter in der Lage sein, selbstständig auf der ISS zu operieren. Die Programmierung von Jitter erfolgte in JAVA unter Verwendung des noch vorgestellten leJOS.

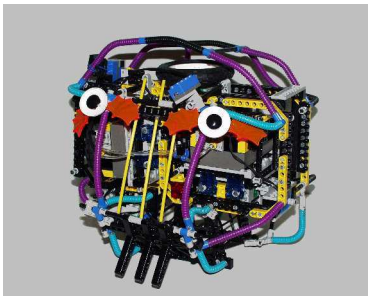


Abbildung 8: Der LEGO-Robot Jitter

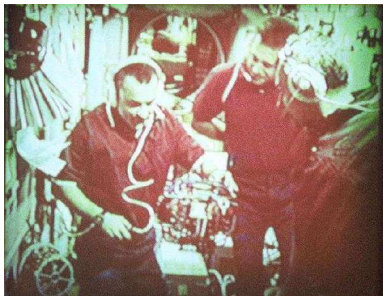


Abbildung 9: Jitter im Einsatz auf der ISS

16 Informationen über JITTER können im Internet abgerufen werden unter:
http://lejos.sourceforge.net/assets/robots/jitter.html
http://www.javamagazin.de/itr/online_artikel/psecom,id,232,nodeid,11.html

Der RCX

Der RCX stellt das “Gehirn” des Lego© Mindstorm™ dar. LEGO liefert in Kombination mit dem Bausatz das RIS- (Robotics Invention System) und das LabVIEW – Software-Paket aus. Diese sind jedoch durch ihre Eignung ab einem Alter von 12 Jahren somit nicht so komplex, wie es dem motivierten Entwickler genügt.

Das “spirit.ocx”-Modul von LEGO setzt die Interaktion des Rechners mit dem RCX-Baustein über die Infrarotschnittstelle um und kann auf dem RCX-Baustein Befehle ausführen, neue Programme und Firmware in den RCX laden und Daten vom RCX entgegennehmen [Koch2003].

Es gibt weltweit diverse Projekte, die sich damit beschäftigen, die Möglichkeiten des RCX durch Bereitstellung neuer Betriebssysteme zu erweitern. Eines davon ist leJOS.

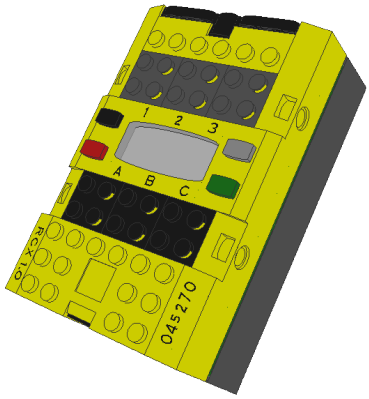


Abbildung 10: Der RCX (mit LDraw visualisiert)

Programmierung des RCX

Der RCX kann und muss von dem Entwickler eines Lego© Mindstorm™-Roboters programmiert werden, um überhaupt über eine (vermeintliche) Intelligenz zu verfügen. Um auch dem unerfahrenen Programmierer ein Werkzeug zur Hand zu geben, mit dem eben diese Programmierung möglich ist, hat man dem Baukasten von Lego© zwei Entwicklungs-umgebungen beigelegt.

Um den Anforderungen der von Lego© erwünschten Zielgruppe zu entsprechen und die jüngeren und/oder unerfahrenen Programmierer zu berücksichtigen, hat man sich dazu entschlossen, dem Nutzer eine Entwicklungsumgebung zur grafischen Programmierung bereitzustellen. Dabei wird, im Gegensatz zur textuellen Programmierung, das Programm via Drag & Drop zusammengepuzzelt,was sehr leicht und intuitiv erlernbar ist. Außerdem ist die Programmstruktur während der Entwicklung des Programms direkt sichtbar, in Form von Flußdiagrammen oder Programmabläufen.

Einige Möglichkeiten der Programmierung des RCX sollen in den folgenden Kapiteln vorgestellt werden.

Das Robotic Invention System (RIS)

Diese für Windows-Betriebssysteme entwickelte Programmierumgebung liegt dem Baukasten eines Lego© Mindstorm™-Roboters bei. Dabei handelt es sich um eine Entwicklungsumgebung für grafische Programmierung. Die Metapher des Puzzles passt hier sehr gut, da in Form von Blöcken programmiert wird, wobei jeder Block für eine Instruktion steht. Die Blöcke werden via Drag & Drop zu einem Programm verbunden, welches dann auf den RCX übertragen werden kann.

Das Programm präsentiert eine Art Flussdiagramm. Im Prinzip gibt es vier verschiedene Formen von Blöcken:

- Befehlsblöcke : grün, sind für einfache Anweisungen
- Sensorblöcke : blau, für die Sensorik
- Stack-Control-Blöcke : rot, für Schleifen und bedingte Anweisungen
- Makroblöcke : gelb, zur Speicherung von Subroutinen etc.

In der Version 1.5 verfügt das *RIS* nicht über Variablen oder Datenspeicher. Berechnungen und komplexe Programmierung sind nicht möglich. Die dem *RIS* eigene Datenstruktur wird im Prinzip nur durch die Sensorwerte, Timer und einen einzigen Zähler dargestellt. In der Version 2.0 hat man das *RIS* um ein einfaches Variablenkonzept erweitert.

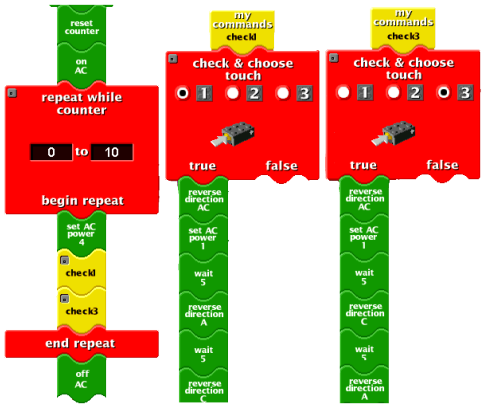


Abbildung 11: Programmbeispiel eines RIS-Programmes

ROBOLAB

Diese Entwicklungsumgebung ist auch für die grafische Programmierung konzipiert worden mit dem vornehmlichen Einsatzbereichs des Schulunterrichts. Es basiert auf *LabVIEW*, einer industriellen Standardsoftware für Mess-, Steuer- und Regelungstechnik. Visuell unterscheiden sich die beiden Umgebungen der grafischen Programmierung nur dadurch, dass *ROBOLAB* nicht mit so grellen Tönen wie *RIS* versehen ist und dass *ROBOLAB* eine kleinere Symbolik verwendet.

Funktional sind die Unterschiede schon größer:

- keine automatische Verbindung der Symbole, nur individuell
- individuelle Auswahl des Programmierniveaus (2 Ebenen mit je 4 Schwierigkeitsgraden)
- ab Version 2.0 eine Ebene zur Erfassung und Auswertung von Messdaten
- kontextsensitive Einschränkung der Symbolauswahl nach Schwierigkeitsgrad
- GoTo-Befehlsvariante (jedoch nur 5 pro Programm)
- Datenspeicher für Integerwerte, sog. Container (3 pro Programm)
- einfache Berechnungen möglich

Auch diese Entwicklungsumgebung hat durchaus ihre Vorteile, besonders für Einsteiger in die Programmierung, jedoch werden weder *RIS* noch *ROBOLAB* den Ansprüchen von erfahrenen Entwicklern nicht annähernd gerecht. Die Zielgruppe Schule gereicht hier zum Nachteil.

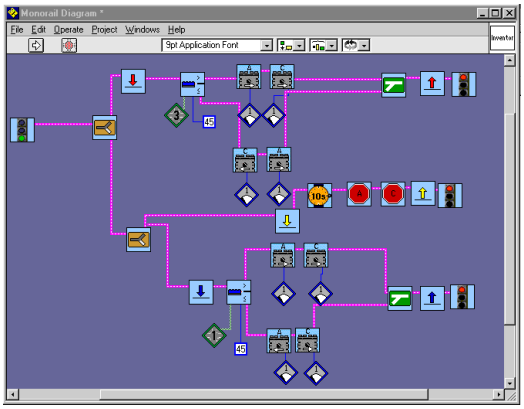


Abbildung 12: Screenshot eines ROBOLAB-Programmes

Auch kann man mit diesen Entwicklungsumgebungen den Anforderungen dieser Arbeit nicht gerecht werden. Es gilt also, sich hinsichtlich der Implementierung mehr Gedanken zu machen. Der erste Schritt in diese Richtung soll gleichfalls ein Schritt in die Nähe der Firmware sein.

NQC

Not Quite C ist eine an C angelehnte Programmiersprache. Sie besitzt jedoch nicht den vollen Sprachumfang noch alle Datentypen des „Originals“. Mit dieser Sprache ist es möglich, auf Basis der Firmware des Lego© Mindstorm™ komplexere Programme zu implementieren. *NQC* produziert Bytecode entsprechend der Firmware.

NQC hat jedoch auch eine Reihe von Einschränkungen:

- Rekursive Aufrufe nicht möglich, da Lego©-Firmware dies nicht vorsieht
- Globale Variablen innerhalb eines Programms auf 32 beschränkt
- Lokale Variablen innerhalb eines Programms auf 16 beschränkt
- Berechnungen mit Floats nicht möglich

Entwicklungsumgebungen im eigentlichen Sinne gibt es hier in großer Zahl, wenn auch nicht immer im herkömmlichen Sinn. Weit verbreitet ist die Programmierung mit einem einfachen Texteditor und Compilierung über den Befehl „make“ unter LINUX.

Da sich das Betriebssystem des RCX auszutauschen lässt, muss man sich bei der Suche nach einer geeigneten Programmiersprache nicht durch die Lego©-Firmware einschränken lassen.

Die folgenden beiden Kapitel beschreiben die Möglichkeit der Erweiterung durch Austausch des eigentlichen RCX-Betriebssystems.

BrickOS

Unter dem Namen *LegOS* wurde erstmals ein OpenSource-Betriebssystem für den RCX zur Verfügung gestellt. Auf Grund von rechtlichen Komplikationen wurde der Name des Betriebssystems jedoch am 18. Juli 2002 geändert und lautet seitdem *BrickOS*.

BrickOS ist ein Alternativbetriebssystem zur eigentlichen Lego©-Firmware. Die Programmierung erfolgt hier in C/C++. Bei *BrickOS* handelt es sich um eine komplette Multitasking-Umgebung.

Als Entwicklungsumgebung wird hierfür eine Variante des gcc bzw. g++ verwendet.

Da dem *BrickOS* weder Lego©-Firmware zugrunde liegt, noch Bytecode erzeugt wird, sind die Einschränkungen des Alternativbetriebssystems lediglich in der Konstruktion des RCX begründet.

Mit dem *BrickOS* wäre den Anforderungen dieser Arbeit zum Teil Genüge getan, wird dadurch die Verwendung einer Hochsprache gewährleistet. Doch liegt der persönliche Fokus eher auf der Verwendung von JAVA als Hochsprache denn auf C/C++. Die Möglichkeit der Verwendung von JAVA stellt das folgende Kapitel dar.

leJOS

Auch leJOS ist ein Alternativbetriebssystem zum eigentlichen Kern des RCX. Wie sein Vorgänger (TinyVM), ist leJOS eine kleine, von Jose Solorzano entwickelte, virtuelle Maschine. Sie funktioniert als Ersatz für die offizielle Lego-Firmware des Lego© Mindstorm™ RCX. Der programmierbare Baustein "RCX" enthält einen Hitachi H8300-Prozessor und 32 Kb RAM, von denen 28 Kb von der Firmware genutzt werden können.
Die Ziele der TinyVM weichen von denen der leJOS ab.
leJOS erlaubt es, mit einem von der TinyVM verschiedenen Design der API und einem anderen Usability-Modell zu experimentieren. Die TinyVM veranschaulicht, wie klein eine angemessene Java-Runtime sein kann, während leJOS mehr aussagt über die Menge der Features, die man in eine RCX-Java-Runtime stecken kann, bevor die Platz-Ressourcen erschöpft sind.
Folgende Eigenschaften haben leJOS und TinyVM gemeinsam:

- Objekt-orientierte Sprache (Java)
- Preemptive Threads (tasks)
- Arrays (auch multi-dimensionale)
- Rekursion
- Synchronisation
- Exceptions
- eine gut dokumentierte API

Die wichtigsten Features, die in leJOS implementiert sind, in der TinyVM jedoch fehlen, sind:

- eine Windows-Version
- Gleitkommazahlen (32 Bit)
- String-Konstanten.
- Casten von long nach int und umgekehrt.
- markiert Referenzen im Stack (was das Implementieren einer garbage collection möglich macht).
- Herunterladen mehrerer Programme.
- Mathematische Funktionen (java.lang.Math): sin, cos, tan, atan, pow, etc.
- andere APIs.

Beide Virtuelle Maschinen stellen umfangreiche API's zur Steuerung des RCX. Diese umfassen z.B. die Kontrolle von Motoren, Sensoren, LCD und Buttons, des Weiteren einen Sensor-Listener und die Kommunikation zwischen mehreren RCX-Bausteinen oder einem RCX und einem PC [Ferrari2002].

LePoMUX

LePoMUX stellt eine Erweiterung der I/O-Datenleitungen des RCX dar und ist 2003 an der HAW von Dietmar Cordes und Gunter Lemm in ihrer Studien- und Diplomarbeit entwickelt worden.

Er ist durch sein Stecksystem direkt zu den Lego© Mindstorm™ kompatibel und erweitert den RCX von 3 auf bis zu 16 I/O-Datenleitungen. Mit dem LePoMUX wurde eine RCX-I/O-Erweiterung mit folgenden Eigenschaften geschaffen:

- IR-Beacon-Detektor
- vier Lego©-kompatible Eingänge
- vier Eingänge für Sharp-Distanzsensoren
- vier Lego©-kompatible Motorausgänge
- Steuerung von vier Modellbauservos
- schnelles Übertragungsprotokoll zur Steuerung des LePoMUX mit bis zu 80 Bytes/Sekunde

Der Lepomux ist zur problemlosen Erweiterung der Lego© Mindstorm™-Roboter besonders interessant[17]. Der Anschaffungspreis liegt derzeit bei unter 200,- Euro.

Das HandyBoard

Bei diesem am MIT entwickelten Board handelt es sich im Prinzip um den Urahn des RCX.

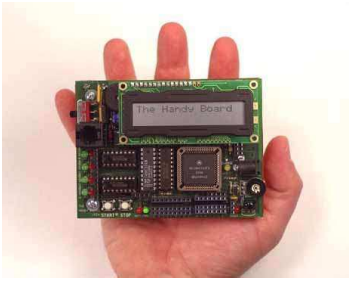


Abbildung 13: Das Handy Board des MIT

17 Weitere Informationen unter: http://www.lepomux.org

Die Programmierung des Prozessors erfolgt beispielsweise in C oder Assembler. Folgende (Anschluss-)Möglichkeiten bietet das HandyBoard:

- 4 Ausgänge für Motoren
- 2 Taster
- 9 digitale Eingänge
- 7 analoge Eingänge
- IR-Sender-Anschluss
- IR-Empfänger
- 1 Drehknopf
- LCD-Display

Es gibt zudem noch die Möglichkeit, das Board zu erweitern durch eines Zusatzmoduls. Dieses besitzt folgende (Anschluss-)Möglichkeiten:

- Ausgänge für 6 Servos
- 9 digitale Ausgänge
- 4 Eingänge für Lego-Sensoren
- 12 analoge Eingänge

Allerdings entfallen bei der Erweiterung die ersten beiden Analogeingänge des Basisboards.

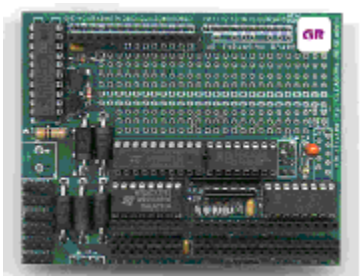


Abbildung 14: Das Erweiterungsmodul des HandyBoards

Man kann das HandyBoard komplett beziehen. Es gibt jedoch auch die Möglichkeit, das Board selbst zu bauen[18].

18 Siehe hierzu auch die Links zu Robotik

JCX

Dieses Board ist noch in der Entwicklung[19] und soll zur Unterstützung von JAVA dienen. Auch hier ist der Einsatz in Bezug auf die Lego© Mindstorm™-Roboter Initiator gewesen. Es verspricht unter anderem:

- Verwendung von Echtzeit-JAVA
- mehr Speicher als der RCX
- mehr Leistung als der RCX

Man sollte diese Entwicklung durchaus im Auge behalten, da dieses Board gerade in Bezug auf die Anforderungen dieser Arbeit recht interessant scheint.

Abbildung 15: Das JCX-Board von Systronix

FischerTechnik

Als Konkurrenz zu LegoTechnik ist gemeinhin FischerTechnik bekannt. Zwar wird dieser nachgesagt, sie sei wesentlich anspruchsvoller und präziser, jedoch ist FischerTechnik nicht so verbreitet wie Lego und hat auch nicht solch eine breite Produktpalette. Dadurch ist FischerTechnik nicht so flexibel wie Lego. Doch ist die Verwendung bei größeren Firmen beliebt: Firmen wie Staudinger entwickeln große, modulare Modelle zur CIM-Simulation auf der Basis von FischerTechnik.

1984 wurde bei FischerTechnik ein erstes Universalinterface angeboten, welches an eine serielle Schnittstelle angeschlossen werden konnte, die zu diesem Zeitpunkt sehr verbreitet war. Sie ermöglichte die Ansteuerung mit Computern wie dem Commodore 64, dem Atari ST oder auch dem IBM PC.

1997 erschien das so genannte Intelligent Interface zum Anschluss an eine serielle Schnittstelle, das einen Intel 80C32-Mikroprozessor enthält und ähnliche Eingaben und

19 Man besuche hierzu auch:
http://jcx.systronix.com/

Ausgaben erlaubt -allerdings mit etwas anderen elektrischen Eigenschaften- wie das parallele Interface. Heute können beide Geräte mit der gleichen grafischen Oberfläche LLWin angesteuert werden, die an das industriell verwendete System iCon-L angelehnt ist. Für das serielle Interface kennt man ein Byte-Protokoll, mit dem man das Interface von einem Computer aus steuern kann, der über eine serielle Schnittstelle verfügt.

Der Bau mobiler Roboter wird dadurch ermöglicht, dass man LLWin-Programme auch in das serielle Interface laden kann. Diese Programme können dort autonom betrieben werden.

Doch so bekannt die Firmware der RCX ist, so unbekannt ist das Innenleben der seriellen Schnittstelle von FischerTechnik. So ist es anscheinend nicht möglich, die von LLWin verwendete Firmware direkt anzusteuern, zu umgehen oder gar zu ersetzen.

Ein weiterer Grund, der Fischertechnik zum Nachteil gereicht, ist die Tatsache, dass der Support für die Robotikpalette eingestellt wurde.

Alternativen

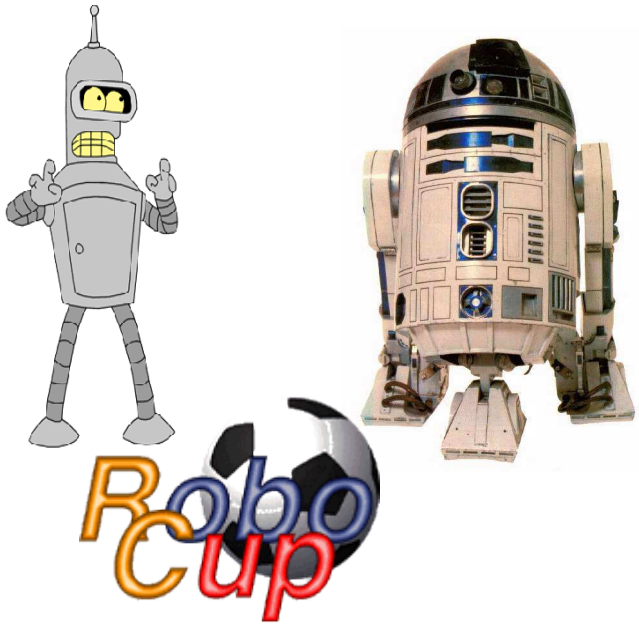
Es gibt noch weitere Alternativen als jene, die ich hier vorgestellt habe. Bei der Auswahl habe ich mich auf die Varianten beschränkt, die einen relativ großen Support, einen hohen Verbreitungsgrad und eine große Gemeinschaft (auch über das WWW) haben. Einzige Ausnahme bildet hier die FischerTechnik.

In Bezug auf Lego oder das HandyBoard sollte sicherlich noch erwähnt werden, dass die Fachhochschule Brandenburg ein eigenes Board entwickelt. Diese Entwicklung sollte durchaus weiter im Auge behalten werden.

Ansonsten sollte man in Bezug auf die Realisierung eines Labors für Künstliche Intelligenz und Robotik erst einmal die hier vorgestellten Varianten prüfen, bevor man nach weiteren Möglichkeiten Ausschau hält.

“Gott ist rund” Titel einer Fußballanalyse

“Bei einem Fußballspiel verkompliziert sich alles durch die Anwesenheit der gegnerischen Mannschaft.” Jean Paul Sartre



Roboter spielen Fußball

Wenn man so will, so ist das, was einen motivierten Wissenschaftler ausmacht, nichts anderes als der Spieltrieb eines Kindes. Wenn man sich dann die Möglichkeiten anschaut, mit denen ein Teil derjenigen, die sich mit KI und Robotik auseinander setzen, seine Ergebnisse präsentiert, so fühlt man sich unweigerlich bestätigt.

Eine dieser Vorführungen ist das mittlerweile in mehreren Ligen eingeteilte Robotfußball (RoboCup). Hier spielen 1 bis n Roboter pro Mannschaft gegeneinander Fußball. Seien es nun die “Kleinen”, die auf ihren Rädern über das Spielfeld huschen oder die “Großen”, die auf zwei Beinen dem runden Gott huldigen.

Die Spielekategorien

Damit ein möglichst großer Forschungsbereich abgedeckt werden kann, wurden verschiedene Spielekategorien – RoboCup leagues – definiert.

Es gibt mehrere Kriterien der Unterscheidung:

- Robotergröße
- Spieleranzahl
- Spielfeldgröße
- Anforderungsprofil an die Roboter

Daraus wurden sieben Kategorien abgeleitet:

- Middle Size Soccer Robot League
- Small Size Soccer Robot League
- Soccer Simulation League
- Humanoid Soccer Robot League
- Sony Legged Robot League
- Rescue Robot League und Rescue Simulation League
- Junior League

Im Folgenden sollen die einzelnen Ligen kurz vorgestellt werden.

Middle Size Soccer Robot League

Es spielen jeweils vier Spieler mit einem maximalen Durchmesser von 50 cm auf einem 5 x 10 Meter großen Spielfeld gegeneinander. Die Roboter sind mit Kameras ausgestattet und haben die Möglichkeit der direkten Kommunikation.

Abbildung 16: RoboCup Middle Size Soccer Robot League

Small Size Soccer Robot League

Auch hier sind pro Mannschaft vier Spieler im Einsatz. Das Spielfeld hat ungefähr die Größe einer Tischtennisplatte. Die Grundfläche eines Roboters beträgt maximal 180 cm^2. Es ist erlaubt, über einen externen Rechner drahtlos zu kommunizieren, humanoide Eingriffe sind jedoch verboten.

Zur Bilderkennung ist über dem Spielfeld eine Videokamera angebracht, doch seit einigen Jahren verfügen die Roboter zunehmend über individuelle Kameras.

Abbildung 17: RoboCup Small Size Soccer Robot League

Soccer Simulation League

Hierbei sind keine Roboter im Einsatz. Es treten je elf Programme gegeneinander auf dem von einem “Soccerserver” simulierten Spielfeld gegeneinander an. Dabei hat jeder Spieler sein eigenes Programm – die individuelle Spielstrategie. In dieser Liga wird auch Abseits “gepfiffen”.

Abbildung 18: Screenshot einer SoccerSimulation-Applikation

Humanoid Soccer Robot League

In dieser Kategorie treten humanoide Roboter gegeneinander an. Diese Roboter bewegen sich auf zwei Beinen und spielen mit einer Mannschaftsstärke von bis zu drei Spielern.

Die Mannschaftsstärke soll jedoch noch auf elf Spieler erweitert werden.

Es gibt unterschiedlich zu lösende Aufgaben für die Humanoiden Roboter und eine Einteilung in Unterkategorien entsprechend ihrer Größe.

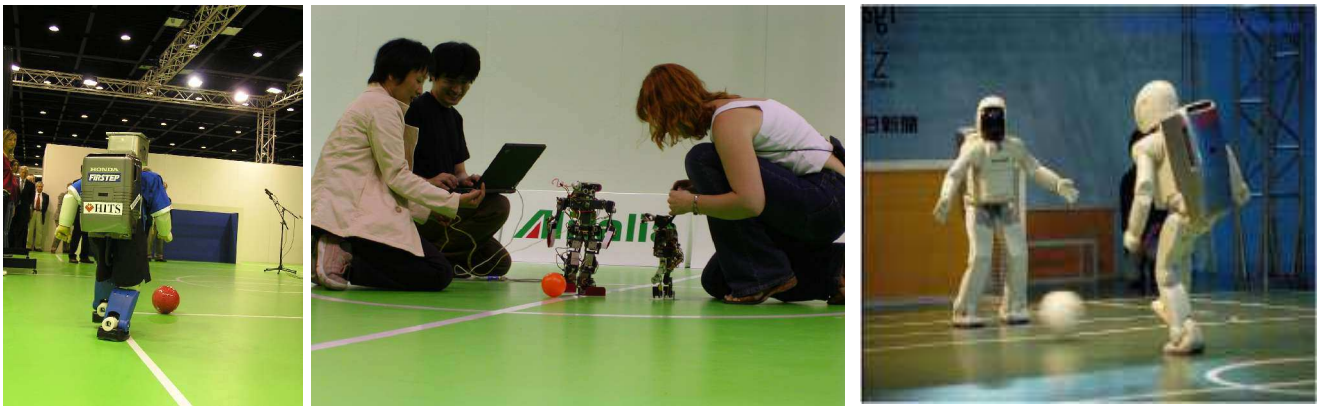


Abbildung 19: Human Soccer Robot League

Sony Legged Robot League

Eine markenspezifische Liga: Hier treten jeweils drei SONY AIBO-Roboterhunde gegeneinander auf einem Spielfeld an. Die Hunde sind durch eine komplizierte Steuerung in der Lage, den Ball ins Tor zu kicken.

Abbildung 20: Sony Legged Robot League

Rescue Robot League und Rescue Simulation League

Hier werden Such- und Rettungsszenarios dargestellt, welche entweder mit Robotern oder mit Programmen in der Simulation durchgeführt werden.

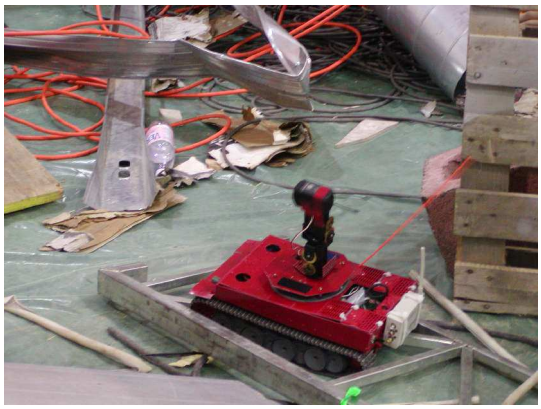


Abbildung 21: Rescue Robot League

Junior League

Auch hierbei gibt es verschiedene Aufgaben zu lösen:

Soccer

Die Mannschaftsstärke beträgt hier bis zu zwei Spieler pro Mannschaft. Das Spielfeld ist 90 x 150 cm groß und in Graustufen gehalten.

Abbildung 22: Junior Soccer League

Dance

Ein oder mehrere Roboter imitieren tanzähnliche Bewegungen zur Musik. Hier zählt auch das Äußere (das Design).

Rescue

Hier werden Opfer von Robotern aus künstlichen Katastrophenszenarios gerettet. Es gibt verschiedene Schwierigkeitsstufen von einfachen Linienverfolgern auf flachem Untergrund bis hin zu Robotern, die sich auf unebenem Gelände ihren Weg an Hindernissen vorbei suchen müssen.

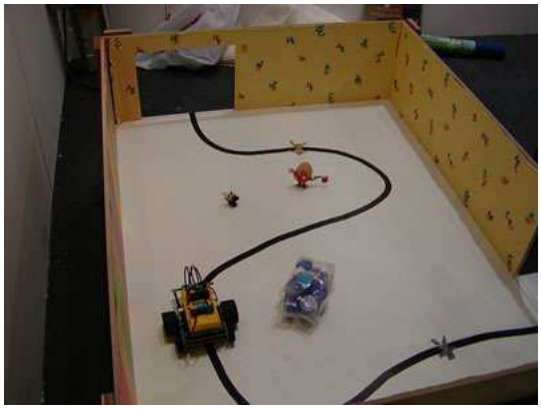


Abbildung 23: Junior Rescue League

Bei den Robotern der RoboCup Junior League handelt es sich meist um Lego©-Mindstorm™-Roboter[Koch2003].

Das Spielfeld der Junior League Soccer

Das von mir angestrebte Ziel ist es, Roboter zu bauen, die den Anforderungen des RoboCup Junior League Soccer genügen. Doch zu einem Roboter gehört auch die Umgebung, in der dieser operieren soll: das Spielfeld.

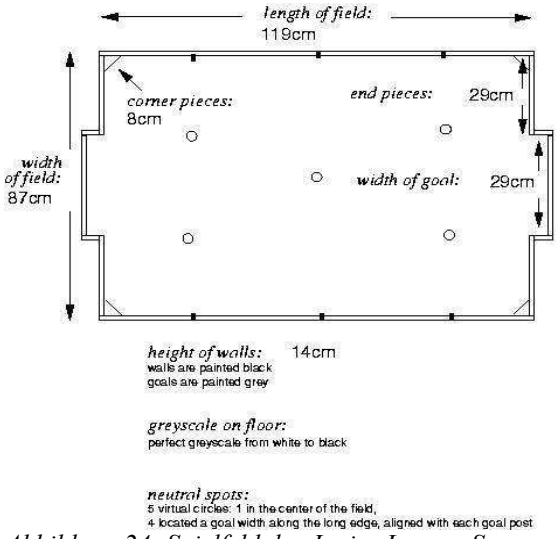


Abbildung 24: Spielfeld der Junior League Soccer

Das Spielfeld für die Mannschaftsstärke 1 sieht aus wie folgt[20]:

Man kann dieses mit geringem Aufwand selbst bauen oder aber sich entsprechendes Zubehör bspw. aus England schicken lassen.

Das Spielfeld im Robot Building Laboratory der HAW Hamburg wurde an den Toren mit jeweils einem Sender ausgestattet, anhand dessen die Roboter erkennen, welches Tor sie gerade anvisieren, wenn sie es denn tun.

20 Die Abmessungen für das Spielfeld finden sich in größerer Darstellung im Anhang

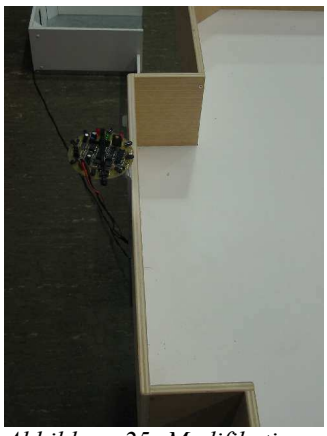


Abbildung 25: Modifikation eines Tores

Nachdem die Spieler und das Feld, auf dem sich die Roboter gegenüberstehen, vorgestellt wurden, fehlt noch etwas entscheidendes:

Der Ball

Bei dem Ball, der in der HAW eingesetzt wird und auch so eine relativ große Verbreitung in der Welt des Robotfußballs erfahren hat, handelt es sich um einen Kunststoffball, der 105 g schwer ist und einen Durchmesser von 80,0 ±0,5 mm aufweist. In seinem Inneren befinden sich Leuchtdioden (LEDs), die durch eine Batterie gespeist werden und durch die der Ball von den Spielern wahrgenommen werden kann.

Abbildung 26: Das Objekt der Begierde

Kostenübersicht

Bei den hier aufgestellten Kosten handelt es sich um die Kosten, die für die Anschaffung der von mir eingeplanten Komponenten nötig sind.

Dabei muss berücksichtigt werden, dass die hier aufgeführten Kosten vermutlich nicht der Aktualität entsprechen, da auch in diesem Bereich preislich eine permanente Bewegung auszumachen ist.

Außerdem wurde hier nur ein Anbieter zu Rate gezogen. Preise sind ohne Frachtkosten. Unterhaltungskosten für Räumlichkeiten und Einrichtung bzw. Ausstattung sind hier ebenfalls nicht (komplett) enthalten.

Bezeichnung	Menge	Einzelpreis	Gesamtpreis
Werkstatt			
Multimeter	1	149,00 EUR	149,00 EUR
Werkzeugsatz	1	36,90 EUR	36,90 EUR
Lötkolben	1	249,00 EUR	249,00 EUR
Miniaturschraubst.	1	24,95 EUR	24,95 EUR
Oszilloskop	1	999,00 EUR	999,00 EUR
Regelb. Netzteil	1	969,00 EUR	969,00 EUR
Zwischensumme$_1$			**2.427,85 EUR**
Labor			
LeJOS	4	kostenlos	kostenlos
Eclipse	4	kostenlos	kostenlos
LePoMUX	4	200,00 EUR	800,00 EUR
SoccerSet[21]	4	1.530,00 EUR	6.120,00 EUR
PC incl. Monitor	4	408,00 EUR	1.632,00 EUR
SuSE 9.0 Campus	1	49,95 EUR	49,95 EUR
SuSE 9.0 WineRack	4	39,95 EUR	159,80 EUR
Zwischensumme$_2$			**8.761,75 EUR**
Gesamtsumme			**11.189,00 EUR**

21 Nähere Angaben zu diesem Set erhält man unter: http://www.edex.com.au/products/item/index.cfm?action=detail&item_edp=14881

„Informatik-vollständig“

Die Einrichtung eines solchen Labores ist nicht nur auf Grund des ureigenen Spieltriebs eine sinnvolle Investition. Der Umfang der Bereiche, die im Kontext der Computerwissenschaften dadurch berührt werden, verdient durchaus die Bezeichnung „Informatik-vollständig“, denn es werden alle Bereiche berührt, sei es sowohl die Robotik oder Künstliche Intelligenz, die in dieser Arbeit schon vorgestellt wurden, als auch das Software Engineering, die verteilten Systeme, Netzwerktechnik, Programmiertechnik & -methodik, Algorithmen und Datenstrukturen, Mathematik, Prozesslenkung, Betriebssysteme etc.

Man sieht also, dass man nicht für jedes Spezialgebiet ein Labor benötigt, sondern im Grunde mit einer Einrichtung alle Bereiche abdecken kann.

Fazit

Es ist also problemlos mit dem nötigen finanziellen Hintergrund und dem technischen Wissen möglich, sich sein eigenes Labor für KI und Robotik aufzubauen. In Bezug auf meine Planung habe ich sicherlich einigen Idealismus (in Bezug auf JAVA und OpenSource) an den Tag gelegt, aber wenn man sich an die Konzeptionierung macht, muss man ja auch gewisse Ziele verfolgen, seien es nun wirtschaftliche oder technische.
Abschließend habe ich eigentlich nur noch zu sagen:

An die Nachgeborenen

Sollte es dazu kommen, dass tatsächlich anhand dieses Konzeptes ein Labor für Künstliche Intelligenz und Robotik geplant wird, muss berücksichtigt werden, dass es sich beim Inhalt dieser Arbeit zum größten Teil um Theorie handelt. Die lokale Umsetzung muss dann noch geprüft werden in Hinblick auf die Rechtmäßigkeit des Einsatzes der hier vorgeschlagenen Ausrüstungsgegenstände. Auch sind die Preise immer zu prüfen.
Diese Arbeit wurde primär durch eine mögliche Realisierung an der University of Shanghai aof Science and Technology (USST) motiviert. Der dort interessierte Fachbereich ist der Fachbereich für Elektrotechnik. Fachlich ist zu überprüfen, über welche Programmiermethodiken und Kenntnisse der Programmiersprachen (insbesondere JAVA) die dortigen Studierenden verfügen. Auch sollte man prüfen, ob die gesteckte Aufgabe des Roboterfussballs die Studierenden vor Ort nicht überfordert. Auch muss ein Skript zur Vorlesung bzw. Veranstaltung erstellt werden.

Literaturverzeichnis

In diesem Verzeichnis finden sich nicht nur die Bücher wieder, die ich bei dieser Arbeit zu Rate gezogen habe, sondern auch Bücher, die mit den angesprochenen Themenkomplexen in Verbindung stehen und hierbei recht nützlich sein und unterstützen können.

[Adorno2001] Theodor W. Adorno, Max Horkheimer. Dialektik der Aufklärung, S.Fischer Verlag GmbH, 2001

[Alpert1998] Sherman R. Alpert, Kyle Brown, Bobby Woolf. The Design Patterns SMALLTALK Companion, Addison Wesley, 1998

[Baum2000] Dave Baum, Michael Gasperi, Ralph Hempel und Luis Villa. Extreme Mindstorms, Apress, 2000

[Beck2000] Kent Beck. Extreme Programming – Revolutionäre Methode für Softwareentwicklung in kleinen Teams, Addison Wesley, 2000

[Bloch2002] Joshua Bloch. Effektiv Java programmieren, Addison Wesley, 2002

[Brooks1991] R.A.Brooks, „How to build complete creatures rather than isolated cognitive simulators“, in K.VanLehn (ed.), Architectures for Intelligence, Lawrence Erlbaum assosiates, Hillsdale, NJ, 1991

[Böhm2002] Oliver Böhm. Java Software Engineering unter Linux, SuSE PRESS 2002

[Cordes2003] Dietmar Cordes, Gunther Lemm. Konzeption von I/O-Erweiterungen für Lego© Mindstorms™, Studienarbeit an der Hochschule für Angewandte Wissenschaften Hamburg, 2003[22]

[Daum2003] Berthold Daum. Java-Entwicklung mit Eclipse 2, dpunkt.verlag, 2003

[Dröge2002] Nils Dröge. Partizipative, evolutionäre Entwicklung im Robot-Art Umfeld, Studienarbeit an der Hochschule für Angewandte Wissenschaften Hamburg, 2002[23]

[Dudek2000] Gregory Dudek, Michael Jenkin. Computational principles of mobile

22 Die Arbeit kann heruntergeladen werden unter: http://www.informatik.haw-hamburg.de/~robots/lepomux/lepomux.pdf

23 Die Arbeit kann heruntergeladen werden unter: http://www.informatik.haw-hamburg.de/~robots/droege/studien.pdf

robots, Cambridge University Press, 2000

[Ferrari2002] Giulio Ferrari, Andy Gombos, Søren Hilmer, Jürgen Stuber, Mick Porter, Jamie Waldinger, Dario Laverde. Programming LEGO® Mindstorms™ with Java, Syngress Publishing, Inc., 2002

[Gamma1996] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Entwurfsmuster, Addison Wesley, 1996

[Goethe1998] Johann Wolfgang Goethe, Faust – Der Tragödie erster und zweiter Teil, Insel Verlag Frankfurt am Main und Leipzig, 1998

[Gulbins2003] Jürgen Gulbins, Karl Obermayr, Snoopy. Linux, Springer-Verlag Berlin, 2003

[Günther2002] Karsten Günther. LaTeX gepackt, mitp-Verlag, Bonn, 2002

[Haykin1999] Simon Haykin. Neural Networks – A comprehensive foundation, Prentice-Hall, Inc., 1999

[Heinsohn1999] Jochen Heinsohn, Rolf Socher-Ambrosius. Wissensverarbeitung – Eine Einführung, Spektrum Akademischer Verlag, 1999

[Horstmann1999] Cay S. Horstmann, Gary Cornell. Core JAVA Bd. 1 – Grundlagen, Markt und Technik Verlag, 1999

[Horstmann2000] Cay S. Horstmann, Gary Cornell. Core JAVA Bd. 2 – Expertenwissen, Markt und Technik Verlag, 2000

[Koch2003] Birgit Koch. Einsatz von Robotikbaukästen in der univeritären Informatikausbildung am Fallbeispiel “Hamburger Robocup: Mobile autonome Roboter spielen Fußball”, Diplomarbeit an der Universität Hamburg, 2003[24]

[Kurzweil1993] Raymond Kurzweil. KI – Das Zeitalter der Künstlichen Intelligenz, Carl Hanser Verlag München Wien, 1993

[Manger2003] Michael Manger. Design und Realisierung von “kostengünstigen” fussballspielenden Robotern, Studienarbeit an der Hochschule für Angewandte Wissenschaften Hamburg, 2003[25]

24 Die Arbeit kann heruntergeladen werden unter: ftp://ftp.informatik.uni-hamburg.de/pub/unihh/informatik/TIS/koch/da.zip

25 Die Arbeit kann heruntergeladen werden unter:

[Marshal1987] Wallace Marshal. Journal of Irreproducable Results,Barnes & Noble, 1987

[Marx2003] Ben Marx (Hrsg.). Linux Manager Guide, SuSE PRESS, 2003

[Nehmzow2002] Ulrich Nehmzow. Mobile Robotik – Eine praktische Einführung, Springer-Verlag Berlin Heidelberg, 2003

[Nilsson1998] Nils J. Nilsson. Artificial Intelligence: A New Synthesis, Morgan Kaufmann Publishers, Inc., 1998

[Schneider2000] Uwe Schneider, Dieter Werner. Taschenbuch der Informatik, Fachbuchverlag Leipzig im Carl Hanser Verlag München Wien, 2000

[Schöning2000] Uwe Schöning. Logik für Informatiker, Spektrum Akademischer Verlag, 2000

[Schreiber2002] Hendrik Schreiber. Performant Java programmieren, Addison Wesley, 2002

[Stolt2001] Matthias Stolt. Roboter im Informatikunterricht, Studienarbeit an der Hochschule für Angewandte Wissenschaften Hamburg, 2001[26]

[Weizenbaum1978] Joseph Weizenbaum. Die Macht der Computer und die Ohnmacht der Vernunft, Suhrkamp Taschenbuch Verlag, 1978

http://www.informatik.haw-hamburg.de/~robots/manger/studienarbeit.pdf

26 Die Arbeit kann heruntergeladen werden unter: http://www.informatik.haw-hamburg.de/~lego/Projekte/Stolt/studienarbeit.pdf

Nützliche Quellen im Internet

Lego & Robotik

HandyBoard http://handyboard.com/

LepoMux http://www.lepomux.org

leJOS http://lejos.sourceforge.net

RCXTools http://rcxtools.sourceforge.net/d_home.html

Lego Community:

RCX & Java http://news.lugnet.com/robotics/rcx/java/

Lugnet http://www.lugnet.com/

LDraw http://www.ldraw.org/

Künstliche Intelligenz

HAW Hamburg http://www.informatik.haw-hamburg.de/~robots/

Robotfußball

RoboCup http://www.robocup.org

Universität Freiburg http://www.cs-freiburg.de/

Educational Experience http://www.edex.com.au/

Anhang

Die Regeln des RoboCup Junior League Soccer

RoboCupJunior 2003 SOCCER rules

last updated: Sun Jan 19 22:44:54 EST 2003 (thomas/sklar)

Note: These rules apply to both 2-on-2 and 1-on-1 Leagues. Differences are noted, where necessary.

1. Playing Field.

1.1. Size.

1.1.1. The playing field for the 1-on-1 League is 87 cm by 119 cm (oversize A0) (1-on-1 field diagram).

1.1.2. The playing field for the 2-on-2 League is 122 cm by 183 cm (2-on-2 field diagram).

1.1.3. As shown in the diagrams, each corner is a triangle of 8cm on each of the sides parallel to the walls.

1.2. Floor.

1.2.1. The floor of the playing field is covered with a printed, matte greyscale.

1.2.2. The playing field should be placed so that it is flat and level. The field may be placed on a table or on the floor.

Hint: It is recommended that teams design their robots to cope with slight imperfections up to 3mm on the surface.

1.3. Walls.

1.3.1. Walls are placed all around the field, including behind the goals.

1.3.2. The walls are 14 cm high.

1.3.3. The walls are painted matte black.

1.4. Goals.

1.4.1.The width of each goal for the 1-on-1 League is 29 cm, centered on the shorter end of the field.

1.4.2.The width of each goal for the 2-on-2 League is 45 cm, centered on the shorter end of the field.

1.4.3.The back and sides and floor of the goal (inside the field) and are painted matte

grey: 75% matte white and 25% matte black.

1.5. Neutral Spots.

1.5.1. For the both leagues, there are five (5) neutral spots defined in the field.

1.5.2. One (1) is in the center of the field.

1.5.3. Four (4) are adjacent to each corner, located a goal width along the long edge of the field, aligned with each goal post; i.e., for the 1-on-1 League, 29cm towards the middle of the field from each goal post (see drawing in 1.1.1); for the 2-on-2 League, 45cm towards the middle of the field from each goal post (see drawing in 1.1.2).

1.5.4. The neutral spots are positions on the field where the referee can place robots or the ball in case play is interrupted (see Interruption of Game Play).

1.5.5. The spots are marked by a small blue cross on the floor of the field.

1.5.6. The ball is to be placed on the goal neutral spots if an interruption occurs while it is in the goal area. The ball is placed in the central neutral spot if an interruption occurs while it is in the Centre Area. See the diagram in 1.1.1 or 1.1.2.

1.6. Lighting.

1.6.1. Teams must come prepared to calibrate their robots based on the lighting conditions at the venue.

1.6.2 Every effort will be made to keep ambient light to a low level with infra-red (IR) sources from incandescent lights and natural lighting minimized.

1.6.3 The organizing committee will release the range of light conditions to be expected, at least one month prior to the event.

1.7. Magnetic Conditions.

1.7.1 Every effort will be made by organizers to locate soccer fields away from magnetic fields such as under floor wiring and metallic objects. However sometimes this cannot be avoided.

Hint: It is recommended that teams design their robots to cope with variations in lighting and magnetic conditions, as these vary from venue to venue. Teams should come prepared to calibrate their robots based on the conditions at the venue.

2. Robots.

2.1. Diameter.

2.1.1. For the 1-on-1 League, the upright robot must fit inside an upright 18cm diameter cylinder.

2.1.2. For the 2-on-2 League, the upright robot must fit inside an upright 22cm diameter cylinder.

2.1.3. Robots will be measured with all parts fully extended.

2.2. Height.

2.2.1. The robot height must be 22cm or less.

2.3. Control.

2.3.1. Robots must be controlled autonomously.

2.3.2. Robots must be started manually by humans.

2.3.3. The use of remote control any kind is not allowed.

2.4. Marking/Coloring.

2.4.1. Competitors are required to mark or decorate their robots to identify them as belonging to the same team.

2.4.2. Colors of robots and/or light transmitters must not interfere with the light sensors readings of other robots. Transmitters on LEGO light sensors must be covered. Blue Tac is recommended.

2.5. Team.

2.5.1. For the 1-on-1 League, a team shall consist of one and only one (1) robot.

2.5.2. For the 2-on-2 League, a team shall consist of no more than two (2) robots.

2.6. Construction.

2.6.1. Any robot kit or building blocks may be used, as long as the robot fits the above specifications and as long as the design and construction are primarily and substantially the original work of the student(s) (see section 4.3).

2.6.2. Construction from raw electronic and hardware components is also allowed, as long as the robot fits the above specifications and as long as the design and construction are primarily and substantially the original work of the student(s) (see section 4.3).

2.6.3. Robot pieces may be permanently attached with glue, screws, etc.

2.7. Ball Capturing Zones.

2.7.1. Ball capturing zones are defined as any internal space created when a straight edge is placed on the protruding points of a robot.

2.7.2. The ball cannot penetrate the Ball Capturing Zone by more than 2cm.

3. Ball.

3.1. Specification.

3.1.1. A well-balanced electronic ball shall be used.

3.1.2. The ball will transmit infra-red (IR) light.

3.2. Suppliers.

There are two electronic balls have been tested by the RoboCupJunior Technical Committee. Both are similar in performance.

3.2.1. IR Roboball MK2 made by Wiltronics (http://www.wiltronics.com.au/productindex.asp?c=54).

3.2.2. RoboSoccer ball made by EK Japan (email: info@elekit.co.jp).

3.2.3. Acroname Inc., USA is also supplying the EK RoboSoccer ball (http://www.acroname.com/robotics/parts/R194-ROBO-BALL.html).

3.3. Competition ball.

3.3.1 For RoboCupJunior-2003 the official ball will be announced prior to the tournament.

4. Inspection.

4.1. Schedule.

4.1.1. The robots will be examined by a panel of referees before the start of the tournament to ensure that the robots meet the constraints described above.

4.1.2. It is the responsibility of teams to have their robots re-inspected if their robots are modified at any time during the tournament.

4.2. Robot configuration.

4.2.1. While being inspected, each robot must be upright and at its maximum size; i.e., anything that protrudes from the robot must be fully extended.

4.3. Students.

4.3.1. Students will be asked to explain the operation of their robots in order to verify that the construction and the programming of the robot is their own work.

4.3.2. Students will be asked questions about their preparation efforts, and they will be requested to answer surveys and participate in video-taped interviews for research purposes.

4.3.3. (previously numbered section 4.3.2.) Commercial kits may be used but must be substantially modified by the students.

4.3.4. (previously numbered section 4.3.3.) Robots must be predominantly constructed and programmed by the students.

4.4. Violations. (previously numbered section 4.5.)

4.4.1. Any violations of the inspection rules will prevent that robot competing until modifications are effected.

4.4.2. However, modifications must be made within the time schedule of the tournament and teams must not delay game play while making modifications.

4.4.3. If a robot fails to meet all specifications (even with modification), the robot will be disqualified for that game (but not the tournament).

4.4.4. If there is excessive mentor assistance or the work on the robots is not substantially original work by the students, then the team will be disqualified from the tournament.

5. Game Play.

5.1. Pre-game setup.

5.1.1. Organizers will make every effort to provide the teams access to the competition area at least two hours before the start of the competition.

5.1.2. Organizers will make every effort to allow at least 10 minutes of setup time before each game.

Participants should be aware, however, that situations may arise where these conditions cannot be met; and so participants should arrive prepared to cope under conditions that are less than ideal.

5.2. Length of Game.

5.2.1. The game will consist of two 10-minute halves.

5.2.2. There will be a 5-minute break in between the halves.

5.2.3. The game clock will run for the duration of the game (two 10-minutes halves), without stopping (except as noted in Damaged Robots).

5.2.4. The game will run on a central time clock.

5.2.5. Teams can be penalized one goal per minute at the referee's discretion if they are late.

5.2.6. If a team does not report within 5 minutes of the game start, it will forfeit the game and the winning team awarded a 5-0 score line.

5.3. Start of Game.

5.3.1. At the start of the first half of the game, the referee will toss a coin and the team first mentioned in the draw shall call the coin while it is in the air.

5.3.2. The winner of the toss can choose either (a) which end to kick to, or (b) to kick off first.

5.3.3. The loser of the toss will decide the other option.

5.3.4. The team not kicking off in the first half of the game will kick off to begin the second half of the game.

5.4. Kick-Offs.

5.4.1. Each half of the game begins with a kick-off.

5.4.2. All robots must be in located on their own side of the field.

5.4.3. All robots must be halted.

5.4.4. The ball is positioned by the referee in the center of the field.

5.4.5. All robots on the team not kicking off must be at least 30cm away from the ball.

5.4.6. The team not kicking off places their robots on the field first. Robots cannot be placed nor remain behind the goal line.

5.4.7. The team kicking off will place one robot near the ball.

5.4.8. The referee may adjust the placement of the robots.

5.4.9. On the referee's command, all robots will be started by human team members.

5.5. Humans.

5.5.1. In general, movement of robots by humans is not acceptable.

5.5.2. Humans can only move robots at the instruction of the referee.

5.5.3. Before the start of each match, teams should designate one human who will act as "Captain", and be allowed to start, place, remove and replace robots during the game, based on the stated rules and as directed by the referee.

5.5.4. Other team members within the vicinity of the playing field are to remain seated while the ball is in play, unless otherwise directed by the referee.

5.6. Ball Movement.

5.6.1. A robot cannot "hold" a ball.

Hint: Holding a ball means taking a full control of the ball by removing all of its degrees of freedom. For example, this would mean fixing a ball to the robot's body, surrounding a ball using the robot's body to prevent access by others, encircling the ball or somehow trapping the ball with any part of the robot's body. If a ball stops rolling while a robot is moving, it is a good indication that the ball is trapped.

5.6.2. The ball cannot be underneath a robot.

5.6.3. The ball must be visible at all times.

5.6.4. Other players must be able to access the ball.

5.7. Scoring.

5.7.1. The ball must be free rolling to score a goal otherwise it will be deemed "pushed" by the referee and disallowed.

5.7.2. The only exception to this is when a robot makes first contact with the ball less than 15cm in front of the goal.

5.7.3. The referee will blow the whistle when a goal is scored.

5.7.4. After a goal is scored, a kick-off will occur.

5.7.5. The non-scoring team will be awarded the ball.

5.7.6. "Own goals" will be treated as a goal to the opposition, even if the ball is "pushed"

into the goal.

5.8. Interruption of Game Play.

5.8.1. The situations listed in sections 5.9-5.12 may cause play to be interrupted, usually resulting in the movement of the ball to a neutral position while play is allowed to continue.

5.8.2. Play may also be stopped by the referee blowing a whistle, but the game clock is not stopped, all at the discretion of the referee. All robots must be stopped immediately and returned to their position when the whistle was blown.

5.8.3. After a stoppage in play, play will resume on the referee's command and all robots are started simultaneously.

5.9. Lack of Progress.

5.9.1. This occurs if the ball is stuck between multiple robots or between robot(s) and the wall and the ball is deemed by the referee to have no chance of being freed.

5.9.2. Lack of Progress also occurs if the ball has not been touched by any robot for at least 20 seconds and it appears that no robots are likely to hit the ball.

5.9.3. In the case of Lack of Progress, the ball will be moved to the nearest unoccupied neutral zone according to section 1.5 (Neutral Zones).

5.9.4. If the ball has not been touched for two periods of 20 seconds, all stuck robots will be freed using minimal movement by the referee. Goalies should be maintained with the same alignment.

5.9.5. When Lack of Progress is called, any robots sitting behind the goal line will be moved forward out of the goal area.

5.10. Damaged Robots.

5.10.1. If a robot does not move for a period of at least 20 seconds and/or it does not respond to the ball, it will be deemed damaged by the referee.

5.10.2. If a robot is fixed at the wall, because it has no sensors to detect the wall, the robot is not damaged.

5.10.3. If a robot continually returns to the area within the goals, it will be deemed damaged by the referee.

5.10.4. The referee or players may remove damaged robot(s) from the field.

5.10.5. A damaged robot must remain off the field for at least one minute.

5.10.6. A damaged robot may be returned with the referees permission to the neutral spot that is closest to the position on the field from where the robot was removed and does not advantage that robot.

5.10.7. Goalies may be returned to the area in front of the goal.

5.10.8. Play may continue during removal, repair and replacement. Note that the referee may choose to interrupt play if robot damage occurred because of a collision with an opposition robot.

5.10.9. If a robot turns over on its own accord, it will be treated as a damaged robot and removed. If the robot is tipped over after a collision with another robot, it can be righted by the referee and continue playing.

5.11. Multiple Defense (2-on-2 only).

5.11.1. Multiple Defense occurs if more than one robot from the defending side enters the region near the goal and substantially affects the game.

5.11.2. For a "Multiple Defense", the robot having the least influence on play is moved to the nearest neutral spot. In the case where a goalie is involved, the other player will be moved.

5.12. Pushed Goal.

In the event of a pushed goal (see section 5.7), play will be stopped with the referee's whistle. The referee will explain the decision. The goal will not be allowed. The ball is replaced on the nearest available neutral spot before play is resumed.

5.13. Fouls.

5.13.1. If a robot utilizes a device or an action which continuously attacks or charges a robot not in possession of the ball, the referee will call "Foul". The team captain must then remove the robot from the playing field for at least one minute and correct the problem; play will continue (as in 5.10 "Damaged Robots").

5.13.2. If the robot continues to Foul, it will be permanently removed from the game. In 1-on-1, that team will forfeit the game.

5.14 Free Kicks. (previously numbered section 5.9.)

There are no free kicks.

5.15. Penalty Kicks. (previously numbered section 5.10.)

There are no penalty kicks.

5.16. Offside. (previously numbered section 5.11.)

There are no offside rules.

5.17. Timeouts. (previously numbered section 5.12.)

There are no timeouts in the game.

5.18. Substitution. (previously numbered section 5.13.)

Substitution of robots at any time during a tournament is strictly forbidden.

6. Conflict Resolution.

6.1. Referee.

6.1.1. During game play, the referee's decisions are final.

6.2. Rule clarification.

Rule clarification may be made by members of the RoboCupJunior International Technical Committee.

6.3. Special Circumstances. (previously numbered section 6.4.)

Specific modifications to the rules to allow for special circumstances, such as unforeseen problems and/or capabilities of a team's robots, may be agreed to at the time of the tournament, provided a majority of the contestants agree.

7. Documentation.

7.1. All teams must bring written documentation describing their preparation efforts.

7.2. Teams will be given public space (approximately 1 by 2 meters) to display their materials on a poster board.

7.3. Officials will review the documentation and discuss the contents with team members. A prize will be awarded to teams with outstanding presentations.

7.4. Teams are encouraged to visit each other's posters.

8. Code of Conduct. (previously numbered section 7.)

8.1. Fair Play.

8.1.1. Robots that cause deliberate interference with other robots or damage to the field or the ball will be disqualified.

8.1.2. Humans that cause deliberate interference with robots or damage to the field or the ball will be disqualified.

8.1.4. It is expected that the aim of all teams is to play a fair and clean game of robot soccer.

8.2. Behavior.

8.2.1. All movement and behavior is to be of a subdued nature within the tournament venue.

8.2.2. Competitors are not to enter setup areas of other leagues or other teams, unless expressly invited to do so by team members.

8.2.3. Participants who misbehave may be asked to leave the building and risk being disqualified from the tournament.

8.2.4. These rules will be enforced at the discretion of the referees, officials, conference organizers and local law enforcement authorities.

8.3. Mentors.

8.3.1. Mentors (teachers, parents, chaperones and other adult team-members) are not allowed in the student work area.

8.3.2. Sufficient seating will be supplied for Mentors to remain in a supervisory capacity around the student work area.

8.3.3. Mentors are not to repair robots or be involved in programming of student's robots.

8.3.4. Mentor interference with robots or referee decisions will result in a warning in the first instance. If this recurs, the team will risk being disqualified.

8.4. Sharing. (previously numbered section 7.3.)

8.4.1. An understanding that has been a part of world RoboCup Competitions is that any technological and curricular developments should be shared with other participants after the competition.

8.4.2. Any developments may be published on the RoboCupJunior web site after the event.

8.4.3. This furthers the mission of RoboCupJunior as an educational initiative.

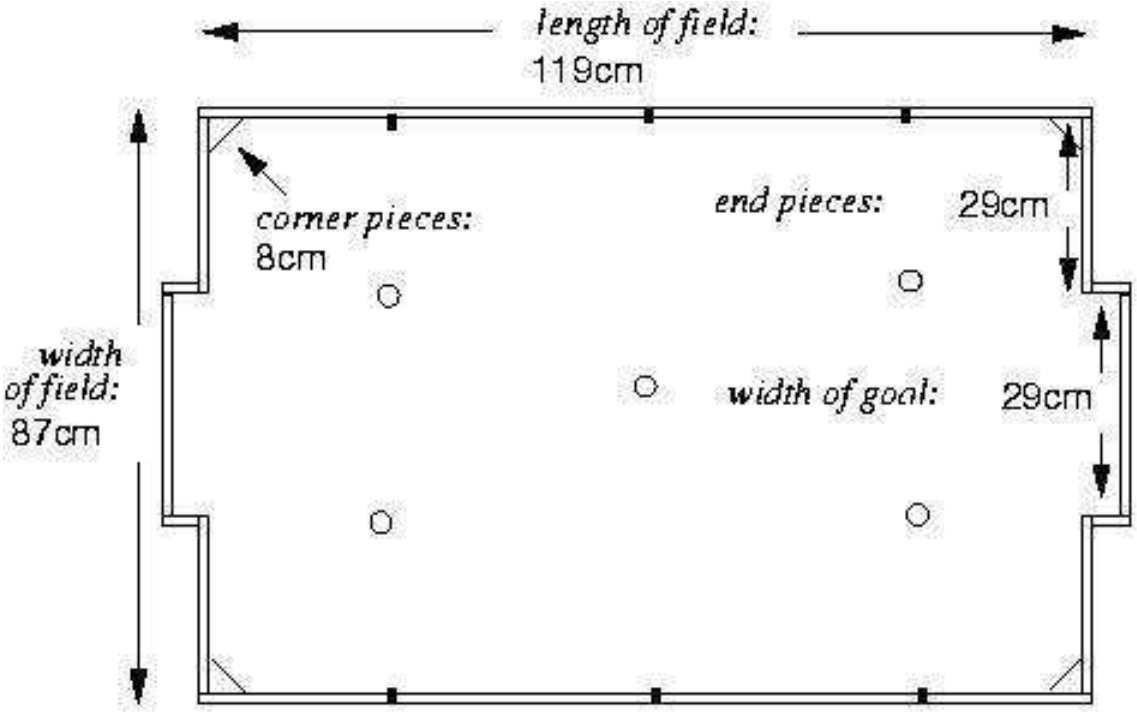
8.5. Spirit. (previously numbered section 7.4.)

8.5.1. It is expected that all participants, Students and Mentors alike, will respect the RoboCupJunior mission.

8.5.2. The referees and officials will act within the spirit of the event.

8.5.3. It is not whether you win or lose, but how much you learn that counts!

Das Spielfeld des Junior League Soccer 1 x 1

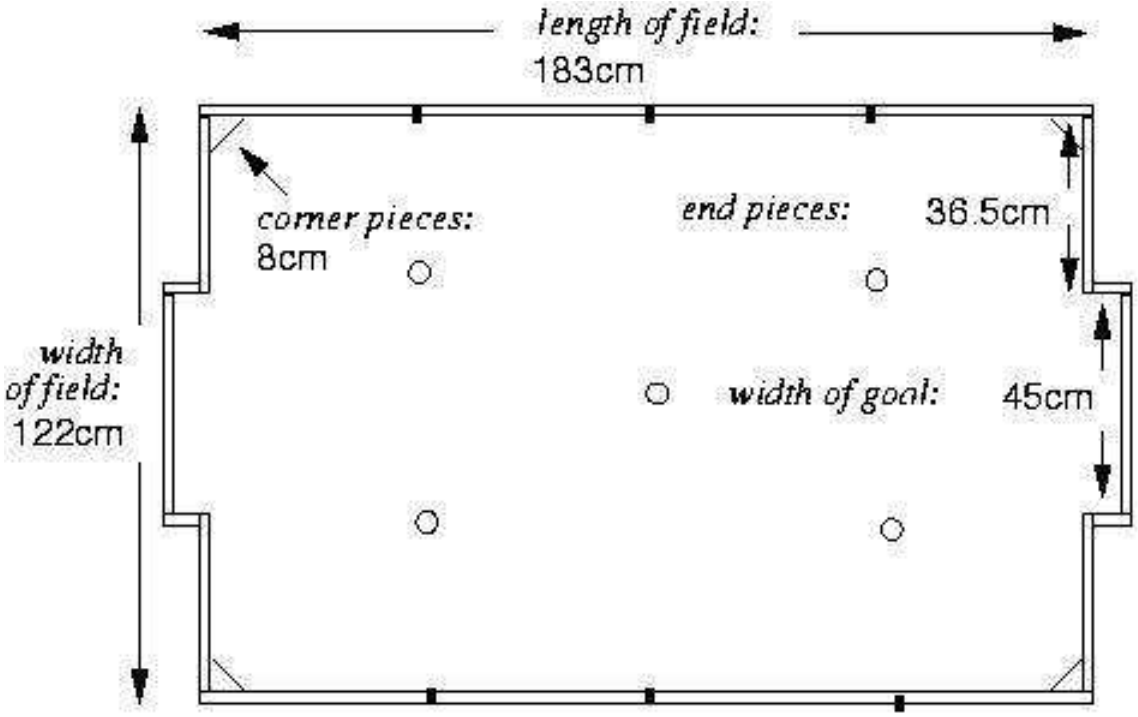


height of walls: 14cm
walls are painted black
goals are painted grey

greyscale on floor:
perfect greyscale from white to black

neutral spots:
5 virtual circles: 1 in the center of the field,
4 located a goal width along the long edge, aligned with each goal post

Das Spielfeld des Junior League Soccer 2 x 2



height of walls: 14cm
walls are painted black
goals are painted grey

greyscale on floor:
15cm band of white along width of one end
15cm band of black along width of other end
perfect greyscale from white to black in between

neutral spots:
5 virtual circles: 1 in the center of the field,
4 located a goal width along the long edge, aligned with each goal post