Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Kai Rosseburg

Entwicklung einer Programmierumgebung für roboterbasierten Informatikunterricht an Schulen

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Kai Rosseburg

Entwicklung einer Programmierumgebung für roboterbasierten Informatikunterricht an Schulen

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

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

Abgegeben am 23. August 2007

Kai Rosseburg

Thema der Bachelorarbeit

Entwicklung einer Programmierumgebung für roboterbasierten Informatikunterricht an Schulen

Stichworte

Robot-Building-Lab, Roboter, Schule, LEGO NXT, Microsoft Robotics Studio, Not eXactly C

Kurzzusammenfassung

Informatische Grundbildung ist in unserer modernen Zeit wichtiger denn je. Zur besseren Verankerung von Informatik im regulären Schulunterricht wurde in dieser Arbeit eine Programmierumgebung für roboterbasierten Informatikunterricht in Schulen entwickelt. Sie ermöglicht es Schülerinnen und Schülern ohne Vorkenntnisse erste Erfahrungen in der Informatik zu sammeln.

Kai Rosseburg

Title of the paper

Development of a Programming Enviroment for robot-based Computer Science Lessons in School

Keywords

Robot-Building-Lab, Robots, School, LEGO NXT, Microsoft Robotics Studio, Not eXactly C

Abstract

Knowledg in Computer Science ist more important than ever. For a better integration of Computer Science in regular school lessons a Programming Enviroment for robot-based Computer Science was developed. It allows pupils who are not familiar with programming and Computer Science to undertake their first experiences in Computer Science.

Danksagung

Ein Dank geht an die Müller-Reitz-Stiftung für die finanzielle Unterstützung dieser Bachelorarbeit.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Diese Arbeit entstand im Rahmen des Projektes Robot-Building-Lab, welches mit Mitteln der Müller-Reitz-Stiftung von Sommer 2006 bis Frühjahr 2008 finanziert wird. Ziel dieses Projektes ist es, Schülerinnen und Schüler der Mittelstufe an naturwissenschaftliche und technische Fragestellungen spielerisch heranzuführen und so gewonnenes Wissen anzuwenden. Dieses Ziel beinhaltet weitere wesentliche Ziele, beispielsweise die Steigerung der Abstraktionsfähigkeit, Teamarbeit und selbstgesteuertes eigenverantwortliches Lernen. Die 2007 vom Bundesministerium für Wirtschaft und Technologie veröffentlichte Studie zum Arbeitskräftemangel belegt, nach Angaben der Süddeutschen Zeitung[1], dass der Mangel an Fachkräften Deutschland jährlich bis zu einem Prozent des Bruttoinlandsproduktes (Stand 2007: ca. 20 Milliarden Euro) kostet. Bedarf besteht hauptsächlich in den Disziplinen Mathematik, Informatik, Naturwissenschaft und Technik. Dieser Umstand verdeutlicht die außerordentliche Bedeutung dieses Projektes. Um die Projektziele zu erreichen, konstruieren und programmieren die Schülerinnen und Schüler autonome mobile Roboter, die dann diverse Aufgaben lösen sollen.

Seit 1999 gibt es von Professoren der HAW Hamburg systematisch Kooperationen mit Schulklassen im Informatik- und Physikunterricht an vereinzelten Schulen. Diese Unterrichtseinheiten werden zumeist als gesondertes Projekt im Rahmen von Projekttagen oder ähnlichem abgehalten. Im Gegensatz zu den bisherigen Ansätzen, solche Programme als gesonderte Projekte durchzuführen, ist es Ziel dieses Ansatzes, das Robot-Building-Lab in den regulären Unterricht der Sekundarstufe I zu integrieren. Grundlage dieser Projektidee ist die Einrichtung eines mobilen

[1] Süddeutsche Zeitung, 20.08.2007

Robot-Building-Labs. Dieses Labor ist nicht im herkömmlichen Sinne als festgelegte Räumlichkeit zu verstehen, sondern vielmehr als integriertes Gesamtkonzept zur Unterstützung des Informatikunterrichts in der Schule.

Ziel dieser Arbeit ist die Entwicklung und Evaluation eines mobilen Robot-Building-Labs. Es soll eine integrierte benutzerfreundliche Programmierumgebung geschaffen werden, die im regulären Schulunterricht eingesetzt werden kann. Sie soll das Erreichen der beschriebenen Projektziele unterstützen. Hierfür müssen die Anforderungen, die das schulische Umfeld an die Umgebung stellt, berücksichtig werden. Die Programmierumgebung muss einfach gehalten sein, damit sie sowohl von Schülerinnen und Schülern, als auch von Lehrkräften benutzt werden kann.

In Kapitel 2 wird der heute vorherrschende Informatikunterricht dargestellt. Es wird eine Definition des Begriffs Informatik gegeben, die Notwendigkeit von Informatik im Schulunterricht dargestellt und Anforderungen an den Informatikunterricht formuliert. Abschliessend werden projekthafte Ansätze, Informatik im Schulunterricht zu verankern, dargestellt.

Pädagogisch-didaktische und informatikthemenbezogenen Motivationen zum Einsatz von Robotern im schulischen Umfeld werden im 3. Kapitel aufgezeigt.

In Kapitel 4 werden die die Erkenntnisse der vorangegangen Kapitel für den *Entwurf einer Workbench für ein Robot-Building-Lab an Schulen* genutzt. Zunächst werden die Komponenten der Workbench vorgestellt. Ein beispielhafter typischer Workflow der Benutzung der Workbench wird präsentiert. Anschließend werden verschiedene Architekturen zur Programmierung von Robotern diskutiert, die dann zur Einführung der Klasse 'Robot' angewendet werden.

Im 5. Kapitel wird der Einsatz des Robot Building Lab evaluiert. Der Vortest im Rahmen der Herbsthochschule 2006 und der Einsatz im Luisengymnasium und am Kurt-Körber-Gymnasium werden beschrieben und bewertet.

Im letzten Kapitel werden die Ergebnisse besprochen und bewertet und Ausblick für einen weiteren Verlauf des Projektes gegeben.

2 Informatik im Schulunterricht

In diesem Kapitel wird die momentane Situation der Informatik im Schulunterricht dargestellt. Es wird zunächst der Begriff Informatik definiert und die Notwendigkeit von Informatik im Schulunterricht begründet. Im Anschluss werden Anforderungen an den Informatikunterricht formuliert und abschliessend projekthafte Ansätze zur Verankerung von Informatik im Schulunterricht vorgestellt.

2.1 Definition von Informatik

Es gibt keine einheitliche Definition für diese noch relativ junge Wissenschaft.[Wei05] Volker Claus und Andreas Schwill definieren im Duden Informatik wie folgt:

> „Informatik ist die Wissenschaft der systematischen Verarbeitung und Speicherung von Informationen, besonders der automatischen Verarbeitung mit Hilfe von Computern.“[CS01]

Im Positionspapier der Gesellschaft für Informatik e.V. (GI) heißt es:

> „Die Wissenschaft Informatik befasst sich mit der Darstellung, Speicherung,Übertragung und Verarbeitung von Information. Dabei untersucht sie die unterschiedlichsten Aspekte: elementare Strukturen und Prozesse, Prinzipien und Architekturen von Systemen, Interaktionen in kleinen, mittleren und weltumspannenden Netzen, die Konzeption, Entwicklung und Implementierung von Hardware und Software bis hin zu hochkomplexen Anwendungssystemen und der Reflexion über ihren Einsatz und die Auswirkungen.“[Ie06]

Wilfried Brauer und Siegfried Münch definieren allgemeiner im Studien- und Forschungsführer Informatik:

> „Informatik ist die (Ingenieur-)Wissenschaft von der theoretischen Analyse und Konzeption, der organisatorischen und technischen Gestaltung sowie der konkreten Realisierung von (komplexen) Systemen aus miteinander und mit ihrer Umwelt kommunizierenden (in gewissem Maß intelligenten und autonomen) Agenten oder Akteuren, die als Unterstützungssysteme für den Menschen in unsere Zivilisation eingebettet werden müssen - mit Agenten/Akteuren sind Software-Module, Maschinen oder roboterartige Geräte gemeint.“[BM96]

Nahezu allen Definitionen ist gemein, dass sie die Informatik als Wissenschaft definieren, die sich mit der Verbreitung und Speicherung von Informationen befasst. Sie ist sowohl Grundlagen- als auch Ingenieurswissenschaft und ähnlich der Mathematik eine Formalwissenschaft. Im Gegensatz zur Mathematik befasst sich die Informatik vorwiegend mit diskreten Problemen. Die untersuchten Probleme sind klar gegeneinander abgrenzbar und in endlicher Zeit lösbar oder beschreibbar. Ihre Produkte hingegen sind ebenso abstrakt und immateriell wie ihr 'Rohstoff'; die Information. Die Informatik versucht hierbei konkrete Anwendungsprobleme in abstrakte Systeme zu übersetzen.Trotzdem die Produkte, wie beispielsweise Software, nicht 'greifbar' sind, haben sie konkrete Auswirkungen auf andere Wissenschaften. Viele Denkweisen der Informatik sind mittlerweile Bestandteil anderer Wissenschaften und führen zu neuen Modellen und Darstellungen. Die Informatik ist eine interdisziplinäre Wissenschaft. Sie arbeitet mit fast allen Wissenschaftsbereichen zusammen oder stellt Dienstleistungen zur Verfügung. Beispielhaft seien Wettersimulationen, das Internet oder einfache Textverarbeitungsprogramme genannt. (vgl. [Wei05] und [Ie06])

2.2 Notwendigkeit von Informatik

Die Gesellschaft für Informatik e. V. (GI) sieht eine Notwendigkeit das „Fach Informatik an den allgemein bildenden Schulen gleichberechtigt zu anderen Fächern einzuführen“[Ie04]. Das Schulfach Informatik kann den Schülerinnen und Schülern helfen, sich in unserer modernen Informationsgesellschaft zu orientieren. Diese moderne Gesellschaft zeichnet sich dadurch aus, dass sie zunehmend von Informations- und Kommunikationssystemen geprägt wird. Informatik ist in viele Lebensbereiche vorgedrungen. Dies gilt insbesondere auf dem Arbeitsmarkt. Um auch in Zukunft ein Teil der Informationsgesellschaft zu bleiben und auf dem Arbeitsmarkt bestehen zu können ist eine informatische Grundbildung notwendig, da hierfür verstärkt fundierte informatische Kompetenzen benötigt werden. Deshalb darf niemand mehr ohne grundlegende Kenntnisse und ein grundlegendes Verständnis moderner digitaler Hilfsmittel bleiben. Um eine „digitalen Spaltung“[1] der Gesellschaft zu verhindern soll das Schulfach Informatik durchgängig als Pflichtfach in der Sekundarstufe I an allen allgemein bildenden Schulen bundesweit eingeführt werden. Durch das Schulfach Informatik können den Schülerinnen und Schülern Kenntnisse über die grundlegenden Funktionsweisen von Informatiksystemen vermittelt werden. Sie sollen befähigt werden, diese Systeme effizient zu nutzen, verantwortungsvoll mit ihnen umzugehen und abschätzen zu können, welche Chancen und Risiken moderne Informatiksysteme beinhalten. Das Verständnis von Informatiksystemen kann durch die Vermittlung von grundlegenden Methoden und Sichtweisen im Schulfach Informatik erreicht werden. Diese Kompetenzen werden in unserer heutigen Gesellschaft zunehmend von jeder und jedem Einzelnen verlangt.[Ie04]

2.3 Anforderungen an Informatikunterricht

Die Behörde für Bildung und Sport in Hamburg sieht das Hauptziel des Wahlpflichtfaches Informatik folgendermaßen:

[1] [Ie04]

> „Der Informatikunterricht hat das Ziel, Schülerinnen und Schülern einen Zugang zur Nutzung, Analyse, Beschreibung und Modellierung komplexer Informationsstrukturen aus Wissenschaft, Wirtschaft und Gesellschaft zu ermöglichen.“[HH03]

Hierbei setzt sich der Nutzen aus verschiedenen Teilbereichen zusammen. Das Verständnis der Wirkungsweisen von modernen Kommunkations- und Informationstechniken soll durch informatische Denkweise geschult werden. Es sollen Problemlösungsmethoden erarbeitet werden, die den Schülerinnen und Schülern im späteren Alltag, Wirtschaft oder Wissenschaft Vorteile verschaffen können. Ziel ist es, eine gleichberechtigte Teilhabe am gesellschaftlichen Leben zu ermöglichen und den Grundstein für lebenslanges Lernen zu legen. Dabei vermittelt der Informatikunterricht den Schülerinnen und Schülern ein Orientierungswissen mit dem sie befähigt werden, sich elektronisch verfügbare Informationen leichter zu erschließen, zu strukturierern und zu erarbeiten. Durch die erworbene Methodenkompetenz können sich die Schülerinnen und Schüler leichter in komplexen und vernetzten Systemen zurechtfinden un ihre Handlungsfähigkeiten in diversen Problemsituationen stärken. Durch den Informatikunterricht erlernen die Schülerinnen und Schüler die Informatik im Kontext ihrer Anwendung, beispielsweise der Roboterprogrammierung zu begreifen. Sie entwickeln einen Blick für das gesamte System. Der Informatikunterricht stellt einen wichtigen Baustein zum Erlangen von Medienkompetenz dar. Die Schülerinnen und Schüler analysieren die Wechselwirkungen und die Struktur von Informatiksystemen und erstellen eigene mediale Produkte und Systeme. Ein interdisziplinärer Unterricht ist in allen genannten Punkten immanent vorhanden. Der Informatikunterricht stellt die Werkzeuge zur Strukturierung und Verknüpfung verschiedener Informationen zur Verfügung. Durch die anwendungsorientierte Nutzung des erworbenem Wissens wird die Wahrnehmung, Durchdringung und Bearbeitung komplexer Problemstellungen gefördert.

Die Informatik ist eine Wissenschaft, die permanenten Veränderungen unterliegt und sich ständig und schnell entwickelt. Die Schülerinnen und Schüler werden im Unterricht mit sich wiederholenden Konzeptionen und Wirkungsprinzipien kon-

frontiert. Dies ermöglicht ihnen sich schnell in die Nutzung neuer Systeme einzuarbeiten. Die Informatik bietet neben den handlungsorientierten Vorteilen kreative Gestaltungsmöglichkeiten. Die selbstständige Analyse, Modellierung und Beschreibung von Lösungsansätzen, sowie deren Reflexion und Weiterentwicklung fördern kreatives und schöpferisches Denken. Die komplexen Problemstellungen der Informatik ermöglichen arbeitsteiliges Vorgehen. Die Schülerinnen und Schüler erlernen soziale Fähigkeiten, wie Kommunikation und Kooperationsfähigkeit. Sie lernen, dass diese Kompetenzen wesentlich sind, um Aufgaben und Projekte erfolgreich zu bewältigen. Außerdem werden wichtige Aspekte und Denkansätze der Projektarbeit, wie Planung, Entscheidungen treffen, Präzision, Aufteilung von Aufgaben und Einhaltung von Absprachen, vermittelt. Der Unterricht gibt den Schülerinnen und Schülern Hilfestellung, sich in der Berufswelt zu orientieren. Wesentliche Grundlagen der Kommunikations- und Informationstechniken und typische Problemlösungen schaffen die Basis für den Einstieg in diverse Berufsfelder. Die so erhaltenen Einblicke in informatische Berufsfelder helfen den Schülerinnen und Schülern neue Möglichkeiten auf dem Arbeitsmarkt zu erkennen und bei der späteren Berufswahl. [Mäh07]

2.4 Ansätze Informatik projekthaft im Unterricht zu verankern

In der Sekundarstufe I wird Informatik fast ausschließlich in Form von Projekten unterrichtet. Entweder im Rahmen einer Projekt-Woche, während Projekt-Tagen oder als Vorbereitung auf Wettbewerbe wie die First-Lego-League[2] oder den Bundeswettbewerb Informatik[3]. Die Dauer der Projekte ist dabei zumeist zeitlich auf mehrere aufeinander folgende Tage begrenzt. Der Wettbewerb- bzw. Projektcharakter fördert die Motivation der Schülerinnen und Schüler. Projekte sind nicht curricular aufnehmbar, daher zielen diese Ansätze zumeist auf eine Art Arbeitsge-

[2] http://www.hands-on-technology.de/firstlegoleague
[3] http://www.bwinf.de/

meinschaft, die nach dem regulären Unterricht durchgeführt werden kann. Projekthafter Unterricht schafft es nicht die Infomatik gleichberechtigt zu anderen Fächern dauerhaft im Fächerkanon zu integrieren. Eine feste Verankerung der Informatik im curricularen Unterricht ist jedoch aus den aufgezeigten Gründen notwendig, um Kompetenzen der Schülerinnen und Schüler auszubilden.

3 Motivation Roboter im Schulunterricht einzusetzen

Seit mehr als 20 Jahren wird am Massachusetts Institute of Technology (MIT) in der 'Epistemology Learning Group' über den Zusammenhang von Lernumgebung und erlernten Fähigkeiten geforscht. Als einer der Pioniere des Einsatzes von Computern im Unterricht gilt der Mathematiker Seymour Papert. Papert, der von 1958 - 1963 mit dem Schweizer Entwicklungspsychologen und Pädagogen Jean Piaget an der Universität Genf zusammenarbeitete, kritisierte schon früh die herkömmliche Schulpädagogik, da der aus ihr resultierende Unterricht am Lernverhalten der Kinder vorbeigeht. Kinder lernen am besten, wenn sie etwas konstruieren können. [Koc03, S.37]

Der Einsatz von Robotern im Unterricht läßt sich durch zwei Gruppen von Motivationen begründen. Die pädagogisch-didaktischen und informatikthemenbezogenen. Bei den pädagogisch-didaktischen stellen der Roboter, seine Konstruktion und die Programmierung nur das Medium, um andere Inhalte zu vermitteln. Die informatikthemenbezogenen Motivationen hingegen bieten die Möglichkeit, durch die Konstruktion und die Programmierung Informatikthemen aufzugreifen und zu vermitteln, die im herkömlichen Unterricht kaum eine Rolle spielen.[Sto01, S.5]

3.1 Pädagogisch-didaktische Motivationen

Roboter sind faszinierend und besitzen daher ein sehr großes Motivationspotential für Schülerinnen und Schüler, sich mit ihnen und Informatik zu befassen. Herköm-

licher Unterricht ist zumeist frontal und vermittelt viel Lehrstoff. Der Lernerfolg wird durch kurz zu beanwortende Fragen überprüft. Diese meistgenutze Methode wird 'telling & testing' genannt. Durch den Einsatz von Robotern im Unterricht sind andere Methoden der Wissenvermittlung, wie projekthaftes, eigenverantwortliches, beziehungsweise selbstgesteuertes und exploratives Lernen möglich.

Wie in anderen Natur- oder Ingenieurwissenschaften wird in der Robotik auch mit vereinfachten Modellen der realen Welt gearbeitet. Der Unterschied besteht darin, dass diese Modelle wieder mit der realen Welt und den potentiellen Problemen, die durch sie entstehen können, in Kontakt gebracht werden. Die Schülerinnen und Schüler erfahren, dass sich theoretisch einfache Probleme in der Realität nicht ebenso einfach lösen lassen. In der momentan vorherrschenden schulischen Ausbildung wird das Prinzip der 'single answer questions' angewandt. Auf die im Unterricht gestellte Frage gibt es genau eine Antwort. Dieses Frage-Antwort-Prinzip hat nur sehr wenig mit den realen Bedingungen der Berufswelt zu tun und verhindert, dass die Schülerinnen und Schüler über alternative Lösungswege nachdenken. Durch die Art der Aufgabenstellung und die Komplexität des Themas ist, anders als im herkömlichen Unterricht, Teamarbeit möglich und häufig notwendig, um die Aufgaben zu lösen. Dies ist eine weitere Motivation für den Einsatz von Robotern, da viele Berufe in unserer modernen arbeitsteiligen Gesellschaft Teamfähigkeit voraussetzen.[1] Schlußendlich ermöglichen Roboter den Schülerinnen und Schülern, unterschiedliche Designansätze auszuprobieren. Das Design eines Roboters ist keine triviale Aufgabe und kann je nach Anforderungen an den Roboter vielfältig sein, sehr ähnlich den Designaufgaben, die ein Ingenieur in der realen Welt vorfindet. So können die Schülerinnen und Schüler anhand vielschichtiger Problemstellungen Designerfahrungen sammeln. Die Designentwicklung kann entweder durch komplettes Durchplanen oder evolutionäres Annähern betrieben werden. [Sto01, S. 5-7]

[1] In diesem Kontext kann auch das von Kent Beck entwickelte pair programming angewandt werden.[Bec00]

3.2 Informatikthemenbezogene Motivationen

Der Einsatz von Robotern im Schulunterricht ermöglicht es, viele Themenkomplexe der Informatik zu behandeln. Bei der Programmierung der Roboter können einfache imperative Programmiersprachen eingesetzt werden. Mit ihnen kann ein Grundverständnis für Algorithmen und Datenstrukturen vermittelt werden.[2] Um komplexer werdende Aufgabenstellungen zu lösen, wurde die Strukturierte Analyse entwickelt. Sie ermöglicht es, Aufgaben in Teilaufgaben und deren Beziehungen zueinander zu zerlegen. Diese weniger komplexen Teilaufgaben können dann weiter bearbeitet werden. Auch in der Robotik bietet es sich an, die komplexen Aufgabenstellungen in Teilaufgaben zu zerlegen. Solche Teilaufgaben können beispielsweise die Zielfindung, Wegplanung oder der Bewegungsapparat sein.

Roboter reagieren typischerweise in Echtzeit auf ihre Umwelt, deshalb können auch Echtzeitprobleme sehr gut mit ihnen verstanden und gelernt werden. Sensordaten müssen in Echtzeit behandelt werden, d.h. es gibt eine harte Zeitkante in der die Reaktion erfolgen muss. Hierbei kann sowohl nebenläufig, beziehungsweise parallel, als auch verhaltensbasiert gearbeitet werden. Parallel werden die Daten, die ein Sensor liefert, von einem Programmstrang (Task) vorverarbeitet und ein anderer Task löst dann die Reaktion auf die Signale aus. Beide Programmstränge konkurrieren um Prozessorzeit. Bei verhaltensbasierter Programmierung werden unterschiedliche Verhaltensweisen implementiert, die konkurrierend parallel ablaufen. Durch Prioritätenvergabe und Signalerkennung wird dann das jeweils notwendige Verhalten ausgeführt. Der Roboter reagiert also auf seine Umwelt und kann als reaktives System begriffen werden. Weitere Informatikthemen sind künstliche Intelligenz oder Multiagentensysteme. [Sto01, S.7-11]

[2] Beispiele für Datenstrukturen sind Anweisungen, Variablen und (bedingte) Verzweigungungen. Viele Programmiersprachen beruhen auf diesem Konzept.

4 Entwurf einer Workbench für ein Robot-Building-Lab an Schulen

In den beiden voran gegangenen Kapiteln wurde die Notwendigkeit der Informatik und ihre bisher nur unzureichende Verankerung im Schulunterricht dargestellt. Auch wurde gezeigt, dass Roboter viele Motivationen bieten um Informatik im Unterricht zu vermitteln. Im folgenden Kapitel wird der Entwurf einer Workbench skizziert. Die Workbench ist das Kernstück des Robot-Building-Lab und bietet eine integrierte Möglichkeit Informatik im curricularen Unterricht zu verankern. Zunächst werden die Anforderungen an die Workbench dargestellt, anschließend kurz die vier Komponenten des Robot Building Lab beschrieben. Dann wird anhand einer typischen Welt die Benutzung der Workbench verdeutlicht. Dies geschieht durch einen beispielhaft-typischen Workflow.

Durch die Diskussion von Architekturen zur Roboterprogrammierung wird die Grundlage zur Einführung einer Steuerklasse 'Robot' gegeben, deren Struktur und Implementierungen dargestellt werden. Bevor die Klasse 'Robot' eingeführt wird, wird der zu programmierende Roboter, der LEGO NXT vorgestellt. Anschließend werden drei Entwicklungsumgebungen und die aus ihnen resultierenden Ansätze der Roboterprogrammierung diskutiert und eine Auswahl für das Projekt getroffen.

Zum Abschluss wird die Testumgebung und eine Implementierung einer simulierten Welt dargestellt.

4.1 Anforderungen an die Workbench

Die Workbench soll Schülerinnen und Schüler, die keine, oder nur geringe Vorkenntnisse besitzen, spielerisch an die Informatik heranführen. Die Schülerinnen und Schüler sollen sich durch eigenes Experimentieren ein Grundwissen in der Informatik aneignen. Sie sollen befähigt werden, eigenständig komplexe Problemstellungen mit Hilfe von informatischen Mitteln zu begreifen und zu lösen. Die Vermittlung von Grundlagen der Informatik, wie der Aufbau und die Funktionsweise eines Computers oder informatische Konstrukte, wie Schleifen, Variablen und Verschachtelungen, steht hierbei im Vordergrund. Die ersten Programmiererfahrungen sollen gesammelt werden. Des Weiteren soll das Abstraktionsvermögen der Schülerinnen und Schüler geschult werden.

Die Anforderungen an die Workbench sollen durch eine testgetriebene Entwicklung in Kombination mit der Methode des 'pair-programming'[Bec00] erfüllt werden.

Die dabei einzusetzende Entwicklungsumgebung muss möglichst einfach gestaltet sein. Das Übersetzen des Programm-Codes, die Ansteuerung des Roboters oder das Laden des Simulators muss automatisch erfolgen. Eine Lösung, die beispielsweise externe Compiler verwendet, ist nicht akzeptabel, da dies die Schülerinnen und Schüler eventuell überfordert und sie sich nicht vollständig auf die informatischen Probleme konzentrieren können. Auch ist eine integrierte Lösung für die Lehrkraft leichter zu überschauen.

Primär soll die Roboterprogrammierung in der Sekundarstufe I eingesetzt werden. Eine Fortführung des Unterrichts in höheren Klassen bis zur Oberstufe ist jedoch wünschenswert. Die Anforderungen an Schülerinnen und Schüler der jeweiligen Jahrgansstufe sind dabei ebenso unterschiedlich wie ihre Fähigkeiten. Deshalb muss es möglich sein, die Komplexität der Roboterprogrammierung für niedrige Jahrgangsstufen zu verbergen, sie jedoch in höhren anzeigen zu können.

4.2 Komponenten des Robot-Building-Lab

Das Robot-Building-Lab besteht aus vier Komponenten. Dem Roboter, einer Welt in der sich der Roboter befindet, einem Host-Rechner und der Dokumentation.

4.2.1 Welten

Welten mit denen Roboter interagieren, können sehr verschieden sein. Sie können sowohl real als auch simuliert sein. Allen Welten gemein ist, dass sie eine Aufgabe beinhalten, die verhaltensbasiert lösbar ist. Es müssen ausreichend Hinweise vorhanden sein, die es dem Roboter ermöglichen die Aufgabe zu lösen. Hinweise können beispielsweise Wände sein, an denen der Roboter entlang fahren kann oder eine schwarze Linie, der er folgen kann.

4.2.2 Roboter

Mobile autonome Roboter sind Maschinen, die aus drei Gruppen von Komponenten aufgebaut sind. Sensoren, Aktoren und einer Steuerungseinheit. Die Sensoren werden in zwei Gruppen unterteilt; interne und externe Sensoren.
Interne Sensoren versorgen den Roboter mit Informationen über seinen Zustand, wie beispielsweise den Batterieladestand. Die externen Sensoren liefern Informationen über die Welt, in der sich der Roboter befindet, beispielsweise die Farbe des Untergrunds oder den Luftdruck.
Aktoren werden als das Gegenstück zu den Sensoren begriffen, da sie den Zustand des Roboters oder seiner Umwelt verändern. Interne Aktoren werden für die Veränderung des Roboter-Zustands, beispielsweise das Neusetzen eines Kreiselkompass oder versetzen in den Ruhezustand, verwendet. Externe Aktoren können beispielsweise Motoren sein, die den Roboter fort- oder einen Greifarm bewegen.
Die Steuerungseinheit verarbeitet die Signale der Sensoren und steuert die Aktionen der Aktoren, sie stellt das Herzstück des Roboters dar. Die Steuereinheit muss dabei nicht direkt im Roboter verbaut sein. Sie kann auch extern auf einem anderen Rechner installiert sein.

4.2.3 Host-Rechner

Zur Programmierung des Roboters wird ein Host-Rechner benötigt. Die Programme werden auf dem Host-Rechner in einer entsprechenden Entwicklungsumgebung geschrieben und dann entweder in ein vom Roboter lesbares Format übersetzt und auf den Roboter heruntergeladen, oder direkt auf dem Host-Rechner ausgeführt. Im letzteren Fall übernimmt dann der Host-Rechner die Steuerung des Roboters und stellt die Steuereinheit des Roboters dar.
Eine weitere Aufgabe des Host-Rechners ist es, simulierte Welten zu berechnen und zur Verfügung zu stellen.

4.2.4 Dokumentation

Da die Workbench im schulischen Umfeld eingesetzt wird, muss auch das Hintergrundmaterial für die Lehrenden umfassend und leicht verständlich sein. Roboter-Kurse können zumeist nicht in jedem Jahr von derselben Person angeboten werden. Um ein kontinuierliches Angebot zu gewährleisten, muss es anderen Lehrkräften möglich sein, sich schnell und ohne größeren Aufwand in die Thematik einzuarbeiten. Durch die Workbench sollen Schülerinnen und Schüler ohne Vorkenntnisse an die Informatik herangeführt werden und ihre ersten Programmiererfahrungen sammeln. Auch ist das begleitende Unterrichtsmaterial besonders wichtig und muss der jeweiligen Jahrgangsstufe angepasst und entsprechend umfangreich sein. Einmal erstelltes Unterrichtsmaterial erleichtert ebenfalls die Übernahme eines Kurses durch eine andere Lehrkraft, da die neue Lehrkraft sich voll auf die Unterrichtsvorbereitung konzentrieren kann und keine Ressourcen mit der Erstellung von Unterrichtsmaterial gebunden werden. Für das Projekt am Luisengymnasium wurde u.a. eine Materialiensammlung erstellt, die sich im Anhang dieser Arbeit findet. Sie richtet sich an die Lehrkraft und versucht eine Einführung in den Aufbau und die Funktionsweise der Workbench zu geben, klärt aber auch technische Sachverhalte.

4.3 Typische Welt & Aufgabe

Nachfolgend wird eine typische Welt dargestellt. Die Welt muss genügend Anreize, wie Kästen oder Außenwände, bieten, darf aber nicht zu kompliziert sein. Die hier dargestellte Welt ist nicht real sondern simuliert. Sie besteht aus einem Kasten, der von Banden umgeben ist, um so eine Spielfläche abzugrenzen. Der Roboter kann sich innerhalb der Banden, zwischen Banden und Kasten fortbewegen. Er besitzt einen Laser-Range-Finder als Sensor und zwei Motoren als Aktoren. Aufgabe ist es den Roboter um einen Kasten fahren zu lassen.

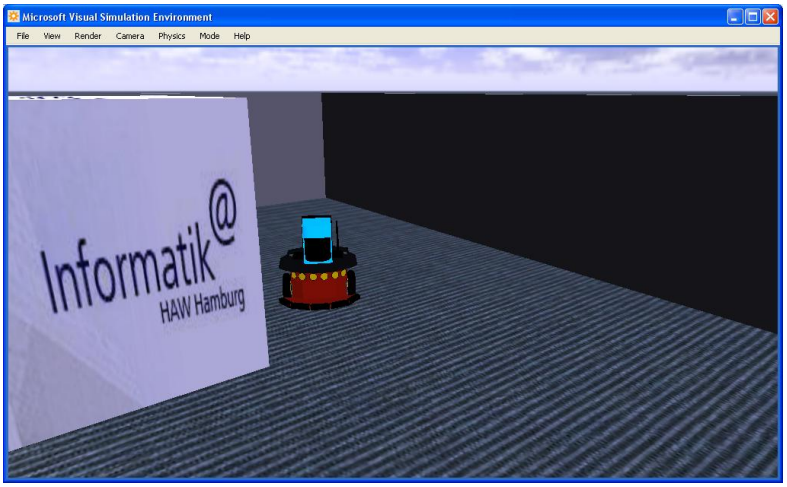

Abbildung 4.1: Typische Welt

4.4 Typischer Workflow

Der testgetriebene 'pair-programming'-Ansatz wird im Folgenden durch ein Beispiel verdeutlicht. Die in 4.3 dargestellte Welt und die beschriebene Aufgabe sind Grundlage für den nachfolgend vorgestellten Workflow.

Zunächst geben die Schülerinnen und Schüler dem Roboter den Befehl vorwärts zu fahren.

```
{
    MoveForward();
}
```

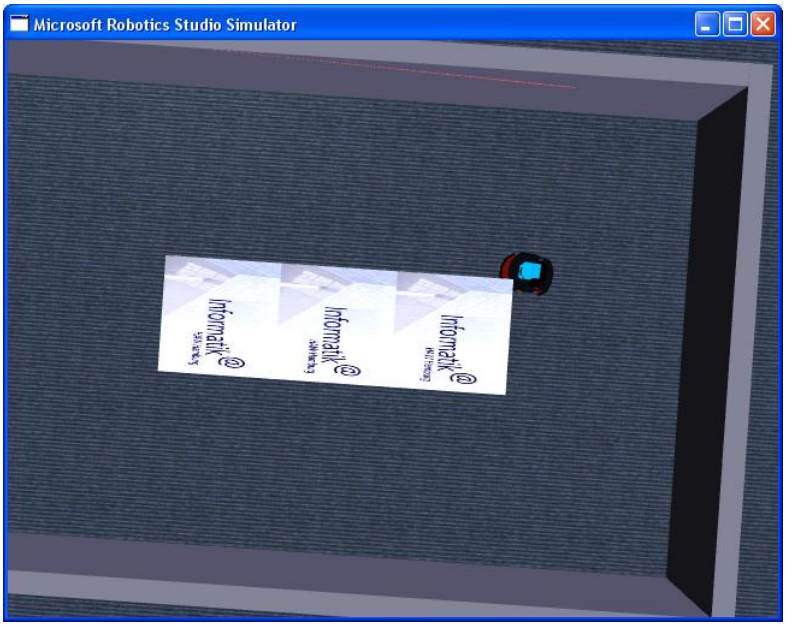

Abbildung 4.2: Workflow - Bild 01

Dies führt dazu, dass der Roboter am Kasten hängen bleibt. Im nächsten Schritt fügen die Schülerinnen und Schüler eine Entfernungsabfrage durch den Laser-Range-Finder hinzu. Außerdem wird eine Verzweigung hinzugefügt, die ausgeführt wird, wenn die Entfernung unterschritten wird. In diesem Fall soll sich der Roboter nach rechts drehen.

```
{
    if (LaserFront() > 3000)
    {
        MoveForward();
```

```
        }
        else
        {
            MoveRight();
        }

    }
```

Abbildung 4.3: Workflow - Bild 02

Die abgefragte Entfernung ist jedoch zu groß, deshalb dreht sich der Roboter permanent im Kreis. Im nächsten Schritt ändern die Schülerinnen und Schüler den Wert in der Entfernungsabfrage.

```
    {
        if (LaserFront() > 1500)
        {
            MoveForward();
```

```
        }
        else
        {
            MoveRight();
        }

    }
```

Jetzt fährt der Roboter um den Kasten herum. In weiteren Schritten würden die Schülerinnen und Schüler versuchen den Roboter, durch Abstandsmessungen zu den Seiten, mittig durch den Parcours fahren zu lassen.

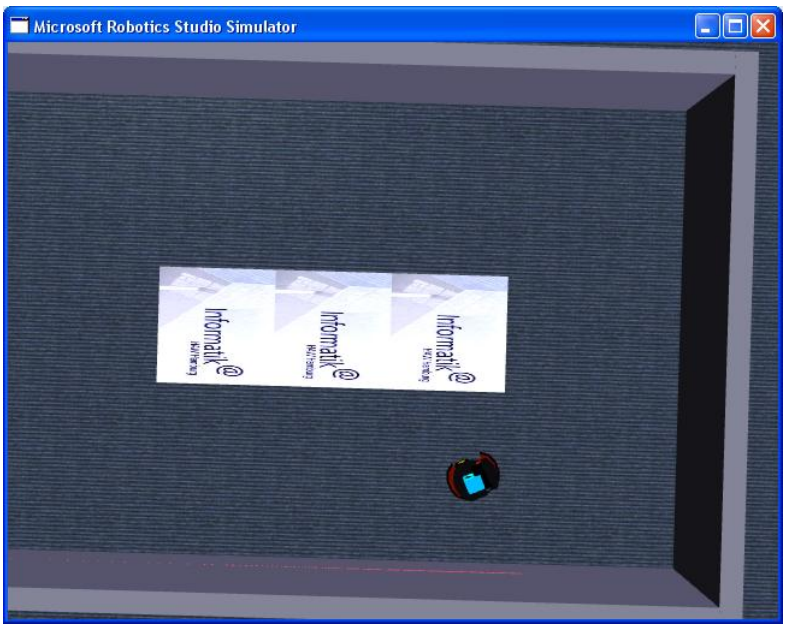

Abbildung 4.4: Workflow - Bild 03

4.5 Roboter Architekturen

Zur softwaretechnischen Realisierung existieren mehrere Ansätze. Die beiden meist genutzten Architekturen sind die deliberative und die reaktive, deren bekanntester Vertreter die Subsumptionsarchitektur ist. [1]

4.5.1 Deliberative Architekturen

In deliberativen Architekturen macht sich der Roboter zunächst ein Bild von der Welt in der er sich befindet. Der Roboter erstellt durch seine Sensoren ein mehr oder weniger exaktes internes Modell der Welt. Im zweiten Schritt plant der Roboter die Aktionen, um sein Ziel zu erreichen. Um dies durchführen zu können, ist es notwendig, dass der Roboter über Informationen verfügt, in welche Einzelschritte Aufgaben zerlegt werden können. Außerdem muss er über Planungsalgorithmen verfügen. Nachdem der Roboter nun ein Bild von seiner Umwelt und einen Plan entworfen hat, beginnt der Roboter mit der Ausführung des Plans.

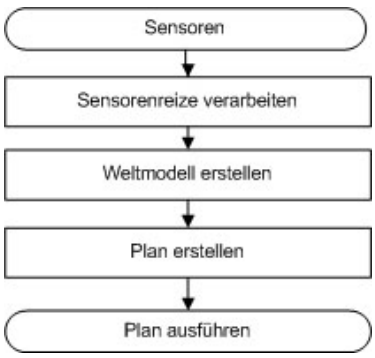

Abbildung 4.5: Schematische Darstellung einer deliberativen Architektur

Hierbei kann es zu Problemen kommen, wenn sich die Welt nach Ende der Planung

[1] Hybrid Architekturen versuchen die Vorteile beider Architekturen zu kombinieren. Sehr lesenswert in diesem Zusammenhang ist die Arbeit von Oliver Köckritz. ([Köc05])

und vor Beginn der Ausführung verändert hat. Im folgenden Beispiel wird diese Problematik verdeutlicht. Ein Roboter hat die Aufgabe beim Fussball ein Tor zu erzielen. Er kann um dieses Ziel zu erreichen einen Ball auf das Tor schiessen. In dem Tor befindet sich in der linken Ecke der Torwart. Der Roboter erstellt sich zunächst das interne Modell der Welt, in diesem Fall des Spielfeldes. Er weiß jetzt, dass der Ball vor ihm liegt und er in Richtung Tor schaut. Die Position des Torwartes ist ihm auch bekannt. Jetzt erarbeitet er den Plan um die Aufgabe 'Schieß ein Tor' zu erfüllen. Der Plan lautet 'Schieß in die rechte Ecke'. Der Plan wird erfolgreich ausgeführt werden, solange sich der Torwart nicht bewegt. Wenn der Torwart sich bewegt, hat der Roboter keine Möglichkeit darauf zu reagieren. Erst nach einer erneuten Berechnung der Welt und der Erstellung eines neuen Plans ist der Roboter in der Lage die Aufgabe zu lösen, sofern sich der Torwart nicht wieder bewegt.

In sich ständig ändernden Welten muss demzufolge das Weltmodell häufig neu erstellt werden. Je nach Komplexität kann dies relativ hohe Anforderungen an die Hardware und Software des Roboters und des Host-Rechners stellen, da viele Ressourcen für die Berechnungen benötigt werden. [Köc05, S.40-41]

4.5.2 Reaktive Architekturen

In reaktiven Architekturen wird das Verhalten des Roboters durch Reiz-Reaktions-Regeln definiert. Die Reize der Sensoren werden über jede Regel iteriert, dabei produziert jede Regel eine Ausgabe. Die Regeln sind dabei sehr kurz und einfach gehalten. Die Kombination dieser Ausgaben ergibt dann das Verhalten des Roboters. Im Unterschied zu deliberativen Architekturen besitzt der Roboter keine interne Abbildung der Welt, planmäßiges Handeln ist nicht möglich. Daher sind die Anforderungen an die Hard- und Software deutlich geringer als bei deliberativen Architekturen. Dies stellt den größten Vorteil der reaktiven Architekturen dar. Da wenig Rechenleistung erforderlich ist, können sie sehr leichtgewichtig sein und haben deutliche Geschwindigkeitsvorteile gegenüber planungsbasierten Archi-

tekturen.

4.5.3 Subsumptionsarchitektur

Ein bekannter reaktiver Ansatz ist der Subsumptionansatz von Rodney Brooks und der Arbeitsgruppe für mobile Roboter am MIT Artificial Intelligence Laboratory.[Bro86] Bei diesem Ansatz werden die Reaktionen des Roboters in Prioritätenlisten geordnet. Höhere Prioritäten verdrängen niedrigere.
Eine einfache Subsumptionsarchitektur ist in Abildung 4.6 dargestellt. Ein Roboter folgt einer schwarzen Linie, auf der sich auch Hindernisse befinden können. Wenn der Ultraschall- oder Kontaktsensor Signale liefert, sich also Hindernisse auf der Route befinden, ändert er sein Verhalten und versucht dem Hindernis auszuweichen. Hierbei ist das Verhalten, welches durch den Ultraschallsensor ausgelöst wird, ein anderes, als jenes, das durch den Kontaktsensor ausgelöst wird. Die Sub-

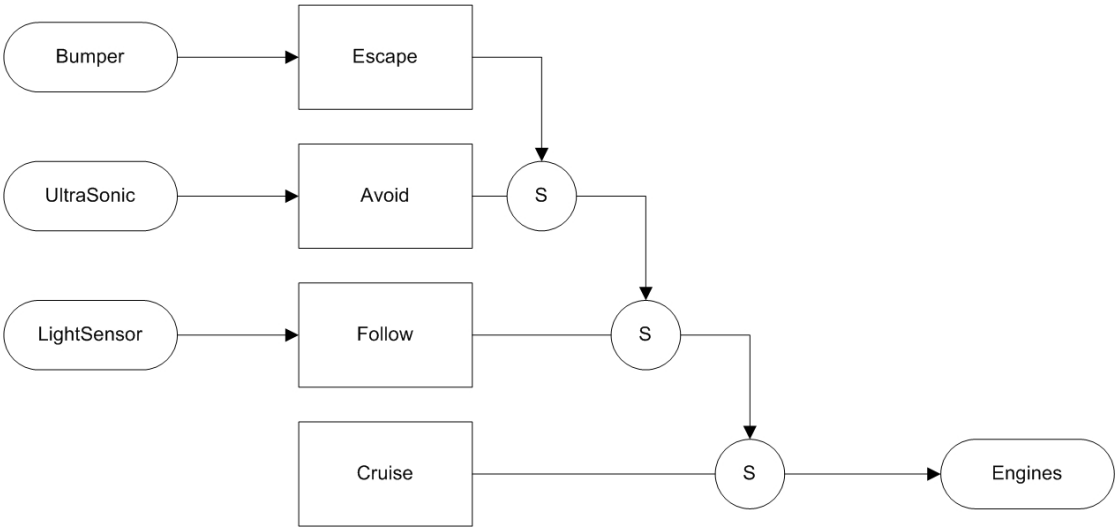

Abbildung 4.6: Einfaches Beispiel für die Subsumptionsarchitektur

sumptionsarchitektur kann sowohl durch nebenläufige Prozesse, als auch serialisiert durch Kaskaden von wenn-dann-Abfragen, realisiert werden. Bei serialisierter Umsetzung wird eine 'while(alive)-Schleife' eingeführt, die über die Sensor-Reize

iteriert.

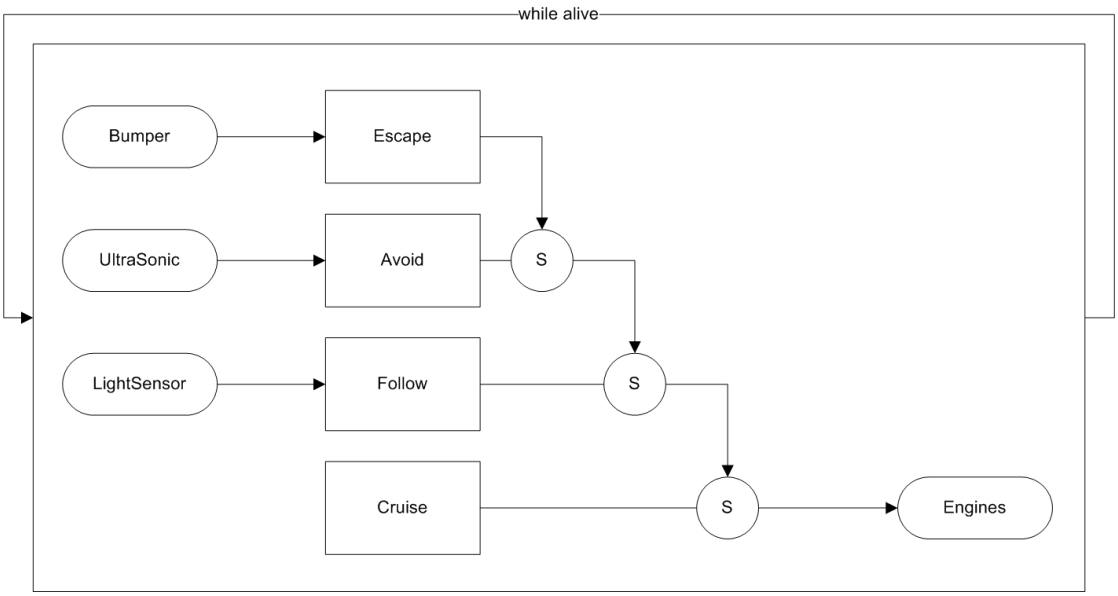

Abbildung 4.7: Einfaches Beispiel für die Subsumptionsarchitektur, erweitert um 'while-alive-Schleife'

4.6 LEGO NXT

Der LEGO NXT ist die Weiterentwicklung der LEGO Mindstorms und des Robotics Invention System, welches von LEGO in Zusammenarbeit mit dem Massachusetts Institute of Technology MIT entwickelt und 1998 auf den Markt gebracht wurde. Viele Aspekte der Forschungen von Seymour Pappert ('learning by doing') sind in den LEGO Mindstorms Baukästen verwirklicht.[Koc03, S.16] Das Herzstück der Baukästen ist ein programmierbarer LEGO-Stein, der Programmable Brick, beziehungsweise RCX-Baustein. Der RCX besitzt einen Hitachi-H8/3292-Microcontroller mit 16 KiloByte ROM und 32 KiloByte RAM als CPU. Außerdem verfügt er über eine Infrarot-Schnittstelle, mit der für den RCX geschriebene Programme zur CPU des RCX heruntergeladen werden können. Dadurch kann

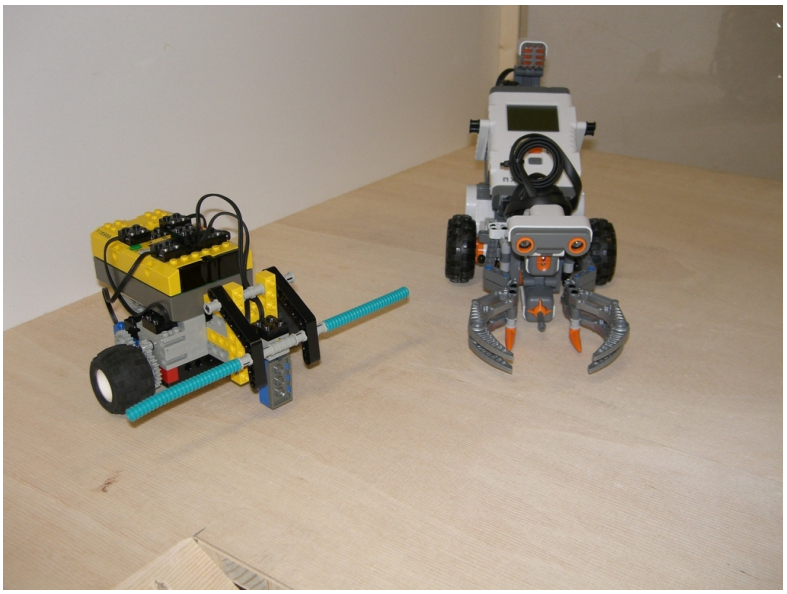

Abbildung 4.8: LEGO RCX & LEGO NXT

der RCX völlig autonom handeln und auf äußere und innere Ereignisse reagieren. Die Infrarot-Schnittstelle kann auch zur Kommunikation zwischen zwei oder mehr RCX genutzt werden, wodurch kooperatives Handeln möglich ist. Der RCX verfügt über drei Ausgänge für Motoren oder Lampen sowie über drei analoge Eingänge mit 10-bit A/D-Wandlern für den Anschluß von Sensoren. Die Leistung der Motorausgänge wird über Pulsweitenmodulation gesteuert. Der RCX verfügt über zwei Motoren und jeweils einen optischen Licht- und einen Tastsensor.[Koc03, S.16-17]

Der LEGO NXT ist seit Oktober 2006 im Handel erhältlich. Das Herzstück des NXT ist der programmierbarer LEGO-NXT-Stein. Der NXT besitzt einen 32-bit ARM7-Microcontroller mit 256 Kilobyte FLASH-Speicher und 64 KiloByte RAM als CPU und einen 8-bit AVR-Microcontroller mit 4 KiloByte FLASH-Speicher und 512 Byte RAM als Co-Prozessor. Der NXT verfügt über eine Bluetooth-Schnittstelle und einen USB 2.0-Anschluss, über die für den NXT geschriebene Programme zur CPU des NXT heruntergeladen werden können. Dies ermöglicht dem NXT ebenso wie dem RCX völlig autonom zu handeln und auf äußere und

innere Ereignisse reagieren zu können. Über die Bluetooth-Schnittstelle kann der NXT mit anderen Geräten, wie z.B. Mobiltelefonen, gekoppelt werden. Der NXT besitzt drei digitale Augänge für Motoren sowie über vier digitale Eingänge für den Anschluß von Sensoren. Außerdem besitzt NXT drei Servo-Motoren und jeweils einen Ton-, Ultraschall-, Licht- und Tastsensor.[LEG]

4.7 Entwicklungsumgebungen & Programmiersprachen

4.7.1 LEGO Entwicklungsumgebung & LEGO-Sprache

LEGO liefert den NXT mit einer in Kooperation mit National Instruments entwickelten Programmierumgebung aus. Die Programmiersprache ist eine auf Symbolen beruhende grafische Programmiersprache, bei der per Drag-and-Drop die Programme zusammengestellt werden, basierend auf der graphischen Programmiersprache LaBVIEW.[LEG] Diese einfache, sehr intuitiv zu benutzende Sprache

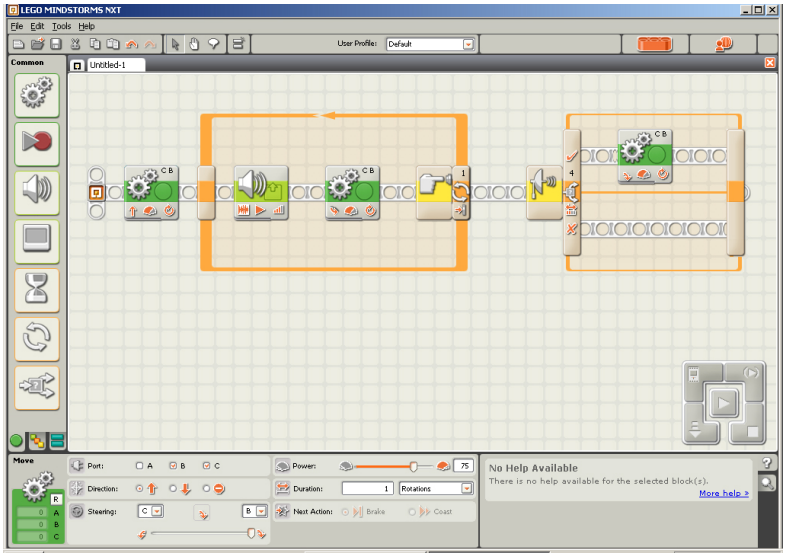

Abbildung 4.9: LEGO Programmierumgebung

implementiert einige Konzepte der Informatik, wie beispielsweise Schleifen, (bedingte) Anweisungen oder Blöcke. Andere, wie Variablen, fehlen jedoch komplett oder sind nicht offensichtlich als solche erkennbar. Die LEGO-Sprache verbirgt die Informatik vor dem Benutzer. Sie besitzt kaum Ähnlichkeit zu anderen Programiersprachen, eine offentsichtliche Verwandschaft zu textuellen Programiersprachen ist nicht gegeben. Daher ist Generalisierung des Gelernten schwierig. Der Transfer in eine andere Programmiersprache ist kaum möglich. Außerdem besteht durch den simplen Aufbau des Programms und die fehlende Tiefe die Gefahr der Unterforderung der SchülerInnen.

4.7.2 Bricx Command Center & NXC

NXC steht für Not eXactly C. NXC ist eine einfache Programmiersprache, die extra für den LEGO NXT entwickelt wurde. Der LEGO NXT besitzt einen von LEGO bereitgestellten Bytecode Interpreter, der dazu genutzt werden kann, Programme auf dem LEGO NXT auszuführen. Der NXC Compiler übersetzt ein in NXC geschriebenes Programm in NXT Bytecode, so dass dieses Programm dann auf den NXT heruntergeladen werden und dort ausgeführt werden kann. Die Syn-

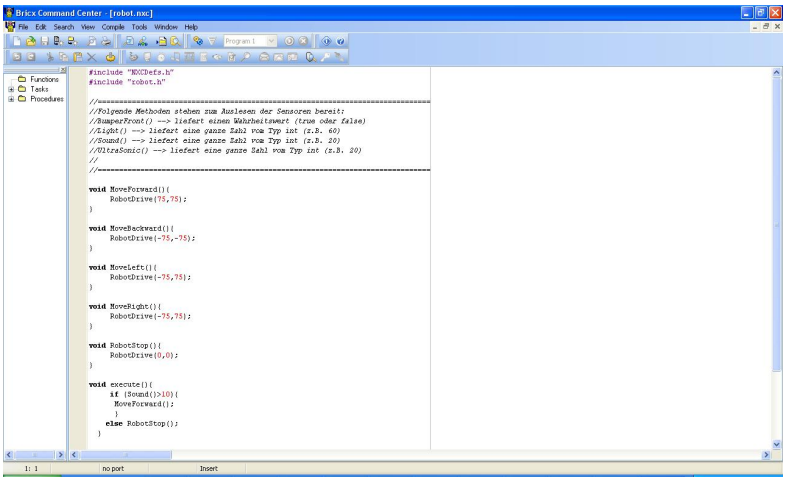

Abbildung 4.10: Bricx Command Center

tax von NXC ähnelt dabei sehr stark der Programmiersprache C. Obwohl NXC C

sehr ähnlich ist, ist NXC keine 'General Purpose Language'. Sie ist also nur für bestimmte Problemstellungen und nicht universell einsetzbar. Diese Einschränkungen liegen im NXT Bytecode Interpreter begründet.[Han07]

Das Bricx Command Center (BricxCC) ist eine integrierte Entwicklungsumgebung zur Programmierung diverser Roboter, wie des LEGO RCX oder des LEGO NXT. Es ist sehr einfach gehalten und beschränkt sich auf die für die Programmierung notwendigen Elemente. Die Oberfläche ist dabei klar strukturiert und sehr übersichtlich. Das Erstellen und Herunterladen von Programmen kann per Knopfdruck durchgeführt werden. BricxCC ist nur in englischer Sprache verfügbar.
NXC und das Brixc Command Center sind unter der Open Source Lizenz frei erhältlich.

4.7.3 Visual Studio, Microsoft Robotics Studio & C#

C# ist eine von Microsoft für das .NET-Framework entwickelte objektorientierte Programmiersprache. C# ähnelt anderen objektorientierten Programmiersprachen wie Java oder C++ und kombiniert viele Eigenschaften dieser Sprachen. Die Sprache ist von der European Computer Manufacturers Association (ECMA) standardisiert und somit ein Industriestandard. C# setzt auf das .NET-Framework auf, welches neben C# noch weitere Programmiersprachen zur Verfügung stellt, die auf der Binärebene interoperabel sind. Dies bedeutet, dass alle .NET-Sprachen prinzipiell miteinander interagieren können. C# selbst verfügt über keine Klassenbibliotheken, kann aber durch das .NET-Framework auf sehr viele bereits vorhandene Bibliotheken, die beispielsweise in C++ oder Visual Basic geschrieben worden sind, zurückgreifen.[Leh02]
C# ist eine 'General Purpose Language' und wird vermehrt im professionellen Umfeld eingesetzt.

Das Microsoft Robotics Studio ist eine windowsbasierte Entwicklungsumgebung für die Entwicklung von Roboter-Programmen. Das Robotics Studio stellt ei-

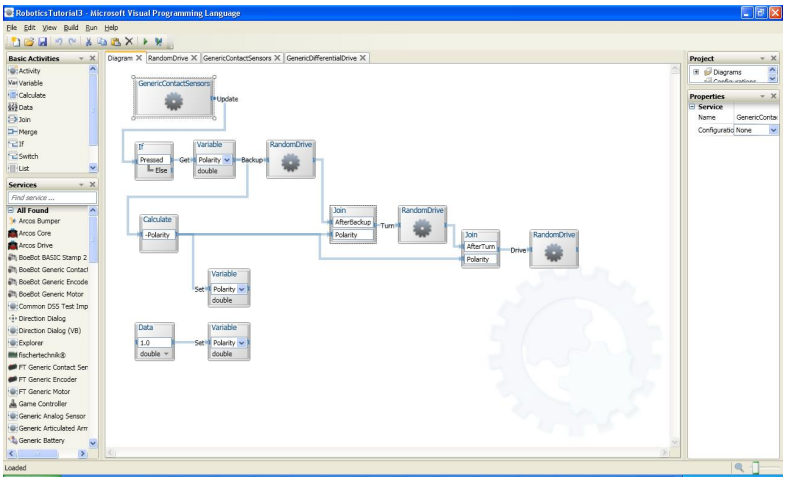

Abbildung 4.11: Microsoft Visual Programming Language

ne Sammlung von Diensten, Klassen und Beispielen bereit, mit denen man eine große Anzahl von unterschiedlichen Roboter-Plattformen programmieren kann. Unterstützt werden bisher Roboter von Fischer Technik, Pioneer und LEGO. Das Robotics Studio abstrahiert dabei vollständig von der anzusprechenden Roboter-Hardware und stellt einheitliche Schnittstellen zur Verfügung. So ist es ohne Probleme möglich ein Programm, das einen Pioneer ansteuert, für einen LEGO NXT zu benutzen, sofern vergleichbare Sensoren und Aktoren verwendet werden. Diese Abstraktion ist nur möglich, da die Programme nicht in ein vom Roboter ausführbares Format übersetzt werden. Das Robotics Studio führt die Programme aus und sendet, beispielsweise über Bluetooth, nur Steuerbefehle an den Roboter. Ein auf dem Roboter laufendes Programm (virtuelle Maschine) nimmt diese dann entgegen und führt sie aus. Durch den integrierten Simulator ist es möglich auch ohne Roboter Programme zu entwickeln und zu testen. Der Simulator ist eine physikalisch korrekt berechnete 3D-Umgebung, die in directX programmiert und leicht anpassbar ist. Das Robotics Studio unterstützt die von Microsoft im .NET-Framework angebotenen Programmiersprachen, wie Visual Basic oder C# und die eigens für das Robotics Studio entwickelte graphische Programmiersprache 'Microsoft Visual Programming Language' (MVPL), die das Programmieren

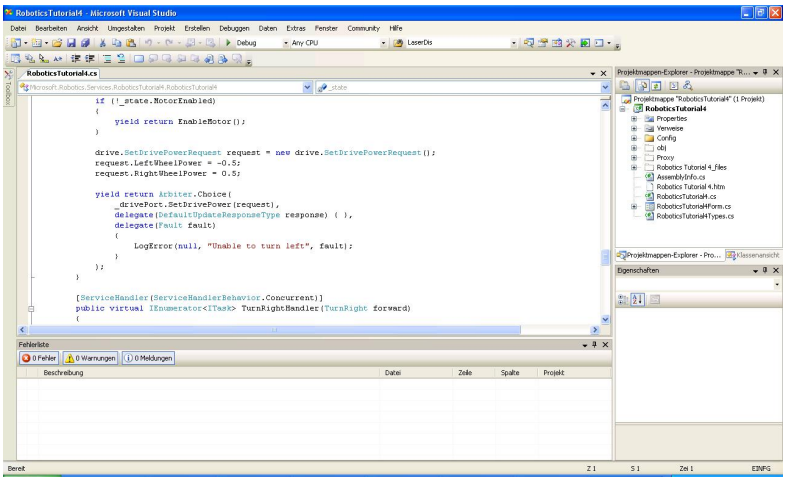

Abbildung 4.12: Microsoft Visual Studio

in Flussdiagrammen ermöglicht. MVPL implementiert analog zur LEGO-Sprache viele Informatik-Konzepte. Ebenso gilt aber auch für MVPL, dass diese Konzepte und somit die gesamte Informatik, teilweise nicht offentsichtlich erkennbar sind und eine Unterforderung der Schülerinnen und Schüler zu befürchten ist.[Mic]

Das Microsoft Visual Studio ist eine integrierte Entwicklungsumgebung, die in nahezu allen Bereichen eingesetzt werden kann. Verbreitet ist sie in der kommerziellen Software-Entwicklung. Sie bietet dem Entwickler ein breites Spektrum an Möglichkeiten zur Entwicklung von Software, deshalb ist sie nicht einfach gehalten und beschränkt sich nicht auf die für die Programmierung notwendigen Elemente. Die Oberfläche ist zwar klar strukturiert, aber für unerfahrene Benutzer sehr unübersichtlich. Das Erstellen und Ausführen von Programmen kann per Knopfdruck durchgeführt werden. Zur Steigerung der Übersichtlichkeit können Teile der Oberfläche ausgeblendet werden.
Das .NET-Framework ist kostenlos von Microsoft beziehbar, das Robotics Studio ist für eine nicht gewerbliche Nutzung ebenfalls kostenlos. Visual Studio ist eine teure vollwertige Entwicklungsumgebung, die es seit kurzem aber auch in einer Express-Version gibt, die wiederum für den privaten Gebrauch kostenlos ist.

4.7.4 Auswahl für das aktuelle Projekt am Luisengymnasium

Die zu befürchtende Unterforderung der Schülerinnen und Schüler, sowie die nicht erkennbare Informatik der LEGO-Sprache waren ausschlaggebend, sie nicht zu verwenden. NXC und C# sind von der Syntax sehr ähnlich. Das Bixc Command Center ist einfacher zu bedienen und übersichtlicher als das Visual Studio von Microsoft. Die Vorteile vom Robotics Studio und C# sind der integrierte Simulator und die Ähnlichkeit von C# zu Java. Der Simulator ermöglicht es den Schülerinnen und Schülern sich vollständig auf die Programmierung zu konzentrieren. Die maschinenbaulichen Anteile können zunächst ausgeblendet werden. Das Luisengymnasium lehrt in der Sekundarstufe II Java. Durch die Ähnlichkeit von C# und Java ist ein späterer Umstieg leichter möglich und es kann an bereits erworbenes Wissen angeknüpft werden. Außerdem können die vorhandenen Vorkenntnisse der Lehrkräfte optimal genutzt werden. Deshalb wird am Luisengymnasium das Robotics Studio mit dem Visual Studio und C# zum Einsatz kommen.

4.8 Klasse 'Robot'

Die Subsumptionsarchitektur ist eine einfache Architektur zur Programmierung von Robotern. Deshalb wird zur Steuerung und Kontrolle des Roboters eine Klasse 'Robot' eingeführt, die diesen Ansatz implementiert. Sie stellt Methoden bereit, um die Sensoren des Roboters auszulesen und die Aktoren zu manipulieren. Sie soll den Schülerinnen und Schülern mit keiner oder geringer Programmiererfahrung ermöglichen, den Roboter ohne lange Einarbeitungszeiten zu programmieren. So können die Schülerinnen und Schüler ihren Fokus auf die Informatik und die Lösung der informatischen Probleme legen. Dies ist notwendig, um eine Überforderung der Schülerinnen und Schüler zu vermeiden und sie nicht durch zu viele Anforderungen schnell zu demotivieren. Um diese einfache Benutzbarkeit herzustellen werden den Schülerinnen und Schülern nur ein begrenzter Befehlssatz und ausgewählte Kontrollstrukturen zur Verfügung gestellt. Der Befehlssatz beschränkt sich auf die Ansteuerung der Motoren und die Statusabfrage bestimmter Sensoren.

Als Kontrollstrukturen werden Abfragen, Verzweigungen und Schleifen zur Verfügung gestellt. Auch Variablen sind vorhanden.
Zur Realisierung der Subsumptionsarchitektur wird die Methode 'execute()' eingeführt. Diese Methode wird im Hintergrund permanent in einer 'while(alive')'-Schleife aufgerufen. Den Schülerinnen und Schülern bleibt dieser permanente Aufruf zunächst verborgen. Hierdurch lernen die Schülerinnen und Schüler unbewußt die Subsumptionsarchitektur durch Ausprobieren und Testen. Neben dem beschränkten Befehlssatz und der Methode 'execute()' besitzt die Klasse 'Robot' keine weiteren Methoden oder Variablen. So kann sie sehr kompakt gehalten werden und die genaue Funktionsweise des gesamten Programms verbergen. Abbildung 4.13 veranschaulicht die Funktionsweise der Klasse 'Robot'. Die Schülerinnen und Schüler arbeiten in dem rot umrandeten Bereich. Dafür werden ihnen Methoden zur Verfügung gestellt. Die Klasse 'Robot' kümmert sich dann um die Ansteuerung der Motoren und das Auslesen der Sensoren. Dies alles bleibt für die Schülerinnen und Schüler verborgen.

4.8.1 Konkrete Implementierungen

C#

Eine Implemtierung der Klasse 'Robot' in C# ist im Folgenden dargestellt. Diese Methoden können verwendet werden:

```
void RobotDrive(double left, double right)
void MoveForward()
void MoveBackward()
void MoveLeft()
void MoveRight()
int LaserLeft()
int LaserFront()
int LaserRight()
bool FrontBumper()
```

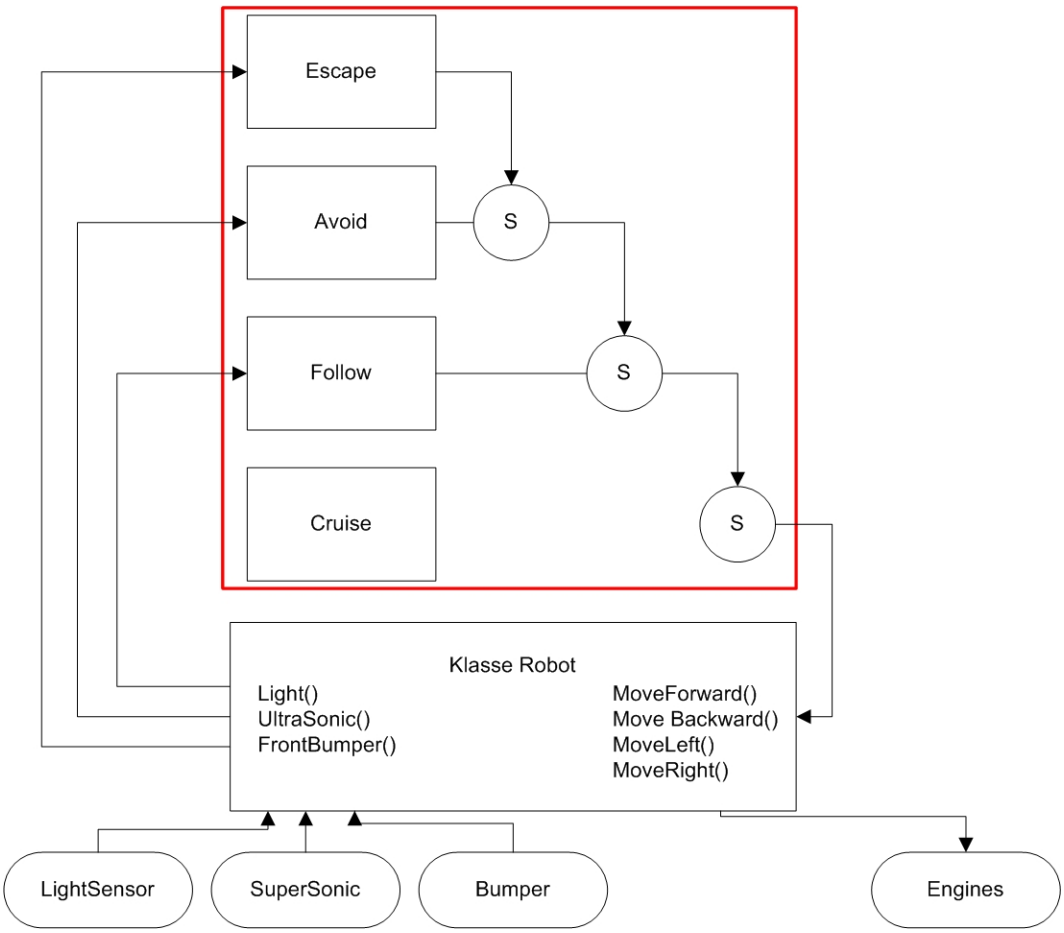

Abbildung 4.13: Darstellung der Klasse 'Robot'

```
bool RearBumper()
```

`RobotDrive(left, right)` ist die Methode mit der der linke und der rechte Motor des Roboters angesprochen werden können. Die Methode erwartet zwei Argumente `left` und `right` vom Typ `double`, die die Geschwindigkeit des jeweiligen Motors darstellen.

Zu Beginn einer Unterrichtseinheit sind die Methoden `MoveForward()`, `MoveBackward()`, `MoveLeft()` und `MoveRight()` noch nicht implementiert. Die Implementierung

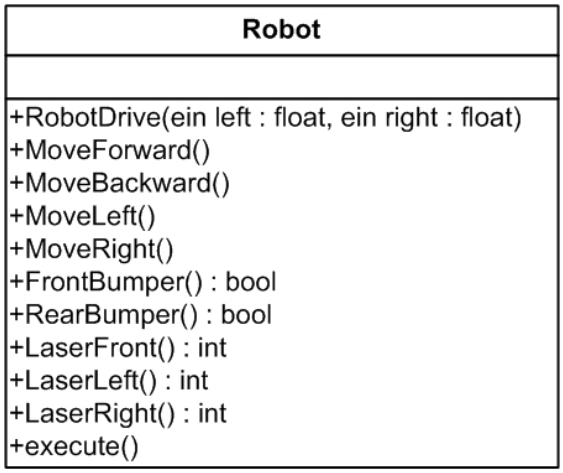

Abbildung 4.14: UML-Klassendiagramm der Klasse Robot in C#

kann von den Schülerinnen und Schülern geleistet werden, indem sie die Methode `RobotDrive(left, right)` nutzen.
Als sinnvoll haben sich folgende Werte für die einzelnen Methoden erwiesen:

`MoveForward()`	`RobotDrive(0.4, 0.4)`
`MoveBackward()`	`RobotDrive(-0.4, -0.4)`
`MoveLeft()`	`RobotDrive(0.0, 0.2)`
`MoveRight()`	`RobotDrive(0.2, 0.0)`

Diese Werte sind nur als Richtwerte zu verstehen und können im Laufe der Unterrichtseinheit den jeweiligen Bedürfnissen angepasst werden.

In der Simulation verfügt der Roboter über einen Laser-Range-Finder, der zur Abstandsmessung verwendet wird. Der LRF deckt einen Bereich von 180° ab. Die Methoden `LaserLeft()`, `LaserFront()` und `LaserRight()` geben die Entfernung zum nächsten linken, mittigen oder rechten Hindernis in mm zurück.

Sowohl in der realen als auch in der simulierten Welt verfügt der Roboter über einen Kontaktsensor, den sog. Bumper (in der simulierten sind es sogar zwei - ei-

ner vorne und einer hinten). Die Methoden `FrontBumper()` für den vorderen und `RearBumper()` für den hinteren Bumper geben einen Wahrheitswert zurück.

NXC

Eine Implemtierung der Klasse 'Robot' in NXC wird im Folgenden dargestellt. Folgende Methoden können verwendet werden:

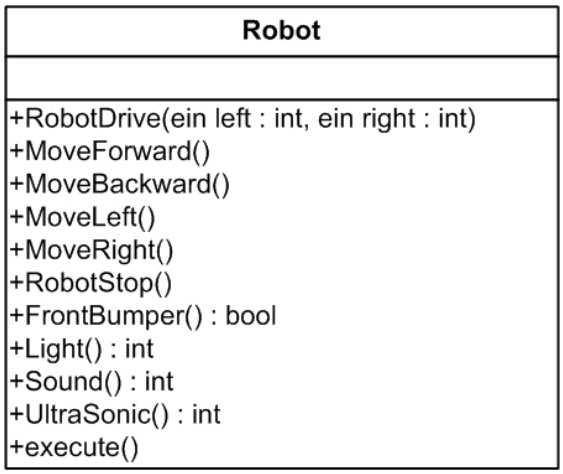

Abbildung 4.15: UML-Klassendiagramm der Klasse Robot in NXC

```
void RobotDrive(int left, int right)
void MoveForward()
void MoveBackward()
void MoveLeft()
void MoveRight()
void RobotStop()
bool FrontBumper()
int Light()
int Sound()
int UltraSonic()
```

`RobotDrive(left, right)` ist analog zur Implementierung in C# die Methode mit der der linke und der rechte Motor des Roboters angesprochen werden können. Sie erwartet im Unterschied zur Implementierung in C# zwei Argumente vom Typ `int`.

Neben der Methode `FrontBumper()` die den Status des vorderen Kontaktsensors zurückgibt, sind Weitere zum Auslesen des Licht-, Sound-, und Ultraschallsensors implementiert, die jeweils eine ganze Zahl zurückgeben, die angibt wie 'dunkel' das vom Lichtsensor Gesehene ist.

4.9 Testumgebung

Nachdem die Schülerinnen und Schüler ihr Programm entworfen haben, folgt die Testphase. Der Parser integrierter Entwicklungsumgebungen überprüft zunächst den geschriebenen Programm-Code auf formale Richtigkeit. Syntaxfehler werden angezeigt. Ist der Code syntaktisch korrekt, wird Programm übersetzt und entweder auf dem Host-Rechner oder dem Roboter ausgeführt. Der Roboter interagiert mit der Welt in der er sich befindet. Dabei kann beobachtet werden, ob der Roboter sich wie erwartet verhält.

4.9.1 Reale Welten

Reale Welten können sehr unterschiedliche Anforderungen an den Roboter und seine Programmierung stellen. Die Welt in Abbildung 4.16 besitzt eine Rampe, die der Roboter hochfahren kann. Hierfür ist es notwendig, dass der Roboter einen tiefen Schwerpunkt besitzt. Die Welt in Abbildung 4.17 hingegen ist ebenerdig.

4.9.2 Simulierte Welten

Damit die Schülerinnen und Schüler sich zunächst auf die Informatik und nicht die maschinenbaulichen Anteile des Robot-Building-Lab konzentrieren, ist es sinn-

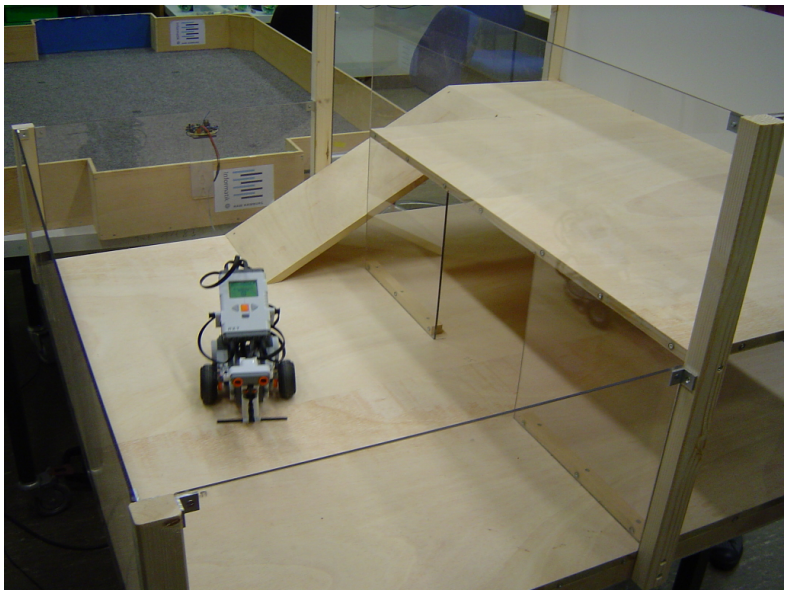

Abbildung 4.16: NXT in realer zweistöckiger Welt

voll einen Simulator zu verwenden. Dieser erinnert stark an Videospiele, die den meisten Schülerinnen und Schülern vertraut sind. Auch scheint es positive Zusammenhänge zwischen Lernerfolg und der Nutzung von Videospielen als Medium zu geben.[May07] Durch den Simulator ist es möglich, sich zunächst vollständig auf die Lösung der Informatik-Probleme zu konzentrieren, ohne durch andere Probleme, beispielsweise durch die Roboter-Hardware oder Probleme der Umgebung gestört zu werden. Simulierte Welten können so gestaltet werden, dass sie realen Welten zum Verwechseln ähnlich sehen und sich dank phsyikalisch korrekter Modelle auch wie reale Welten verhalten. Die in Abbildung 4.18 dargestellte simulierte Welt ist eine Nachbildung der in Abbildung 4.17 abgebildeten realen Welt.

4.9.2.1 Konkrete Implentierung

Zur Gestaltung eigener Simulations-Welten stehen im Robotics Studio folgende Methoden zur Verfügung:

```
void AddSky()
```

Abbildung 4.17: NXT in realer Welt mit Kasten

```
void AddGround()
void AddCameras()
void AddBox(Vector3 position,
       Vector3 dimensions, float mass, String name)
void AddTexturedBox(Vector3 position,
       Vector3 dimensions, float mass, String name)
void AddPioneer3DXRobot(Vector3 position)
private void PopulateWorld()
```

`AddSky()` fügt der simulierten Welt einen Himmel hinzu.

`AddGround()` fügt der simulierten Welt einen einfachen, planen Boden hinzu.

`AddCameras()` fügt der simulierten Welt eine Kamera hinzu. Ohne Kamera kann die Simulation nicht betrachtet werden.

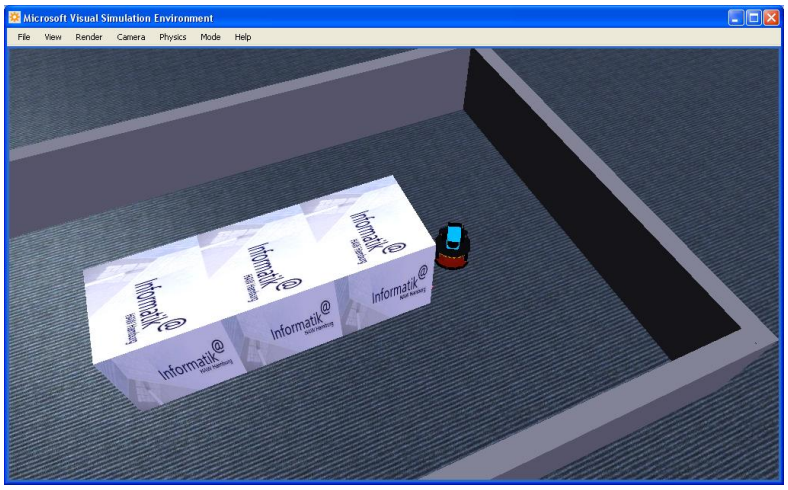

Abbildung 4.18: Pioneer in simulierter Welt mit Kasten

`AddBox(position, dimensions, mass, name)` ist die Methode mit der eine texturlose Box in die Simulation eingefügt werden kann. Die Methode erwartet vier Argumente `position` und `dimensions` vom Typ `Vector3`, `mass` vom Typ `float`, sowie `name` vom Typ `String`.
Mit `position` wird die Position in einem dreidimensionalen Koordinatensystem angegeben, mit `dimensions` wird die räumliche Ausdehnung bzw. Größe der Box festgelegt. Die Positions- und Größenangaben beziehen sich jeweils auf den Mittelpunkt der Box. Mit `mass` wird die Masse der Box festgelegt, je größer die Masse, desto gringer ist die Wahrscheinlichkeit, dass der Roboter sie verschieben kann.
Jedes Objekt in der simulierten Welt braucht einen eindeutigen Namen, deshalb muss dieser mit `name` übergeben werden.

`AddTexturedBox(position, dimensions, mass, name)` erwartet die exakt gleichen Argumente wie `AddBox(position, dimensions, mass, name)`, der einzige Unterschied ist, dass mit `AddTexturedBox(position, dimensions, mass, name)` texturierte Boxen in die Welt eingefügt werden können. Momentan ist die Textur noch fest implementiert und kann nicht verändert werden.

`AddPioneer3DXRobot(position)` ist die Methode mit der ein Modell des Pioneer-Roboters in die simulierte Welt eingefügt werden kann. Die Methode erwartet das Argument `position` vom Typ `Vector3`, welches die Position des Roboters innerhalb der simulierten Welt festlegt. Das Modell des Pioneer verfügt über einen Kontaktsensor (Bumper) und einen Laser-Range-Finder.
`PopulateWorld()` ist die Methode, in der die simulierte Welt zusammen gefügt wird. Alle Methoden müssen innerhalb der `PopulateWorld()` aufgerufen werden. Eine beispielhafte `PopulateWorld()` sie wie folgt aus:

```
{
        AddSky();
        AddGround();
        AddCameras();

        AddBox(new Vector3(0.0f,0.5f,0.0f),
                new Vector3(5.0f,1.0f,0.2f),1,"box");

        AddPioneer3DXRobot(new Vector3(0,0.1f,-2));
}
```

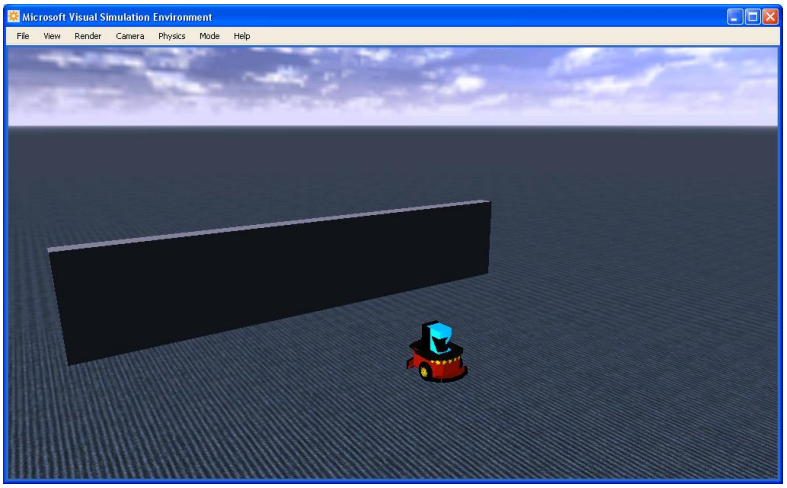

Abbildung 4.19: Die resultierende einfache simulierte Welt

5 Evaluation durch Unterricht vor Ort

Die Entwicklung der Workbench findet parallel zum Unterricht am Luisengymnasium und dem Kurt-Körber-Gymnasium statt. Im Folgenden wird die Evaluation dargestellt. Zunächst wird der Vortest im Rahmen der Herbsthochschule an der HAW Hamburg beschrieben und bewertet. Anschließend wird der Unterricht am Luisengymnasium betrachtet. Der Unterricht am Kurt-Körber-Gymnasium wird kurz anhand eines Beispiels diskutiert. Im Anhang dieser Arbeit finden sich Erfahrungsberichte der betreuenden Lehrkräfte Werner Baum (Luisengymnasium) und Christine Klemm (Kurt-Körber-Gymnasium).

5.1 Vortest Herbsthochschule

Im Rahmen der Herbst-Hochschule an der Hochschule für Angewandte Wissenschaften Hamburg, die vom 16. bis 20.Oktober 2006 stattfand, konnte das Robot-Building-Lab, als Generalprobe vor dem Start im Luisengymnasium, getestet werden. Die Fakultäten Technik und Informatik am Berliner Tor, sowie Life Sciences in Bergedorf der HAW Hamburg boten in den Schulherbstferien 2006 ein auf interessierte Schülerinnen zugeschnittenes Schnupperstudienprogramm an, das den Mädchen eine aktiv teilnehmen ermöglichte.

Die HAW Hamburg verfolgt mit diesem Angebot das Ziel, den Schülerinnen den Zusammenhang zwischen Naturwissenschaften und Technik zu veranschaulichen. Außerdem möchte sie die Besonderheiten ihres praxisorientierten Studiums durch

die Kombination von Theorie, Labor und Übungen vermitteln. Dieses vielfältige Programm wurde durch den gemeinsamen Einsatz von Professoren, Labormitarbeitern und Studierenden entwickelt und umgesetzt.

Abbildung 5.1: Teilnehmerinnen der Herbsthochschule

Neun Schülerinnen der Jahrgangsstufen 11 - 13 nehmen an diesem ersten Testlauf des Robot-Building-Lab teil. Die Gruppe ist, bezüglich der Programmiererfahrung, sehr heterogen zusammengesetzt. Einige der Schülerinnen haben bereits Informatikkurse in der Schule belegt oder privat Erfahrungen in Programmiersprachen wie Java oder Delphi gesammelt, andere sind ohne jede Vorkenntnis.

5.1.1 Zielsetzungen des Testlaufs

Die Herbsthochschule bietet die Möglichkeit das Robot-Building-Lab zum ersten Mal unter annähernd realen Bedingugen zu erproben. Mit dieser Generalprobe sollen eventuell noch vorhandene Probleme erkannt, und vor dem Einsatz in der Schule behoben werden. Dies geht über das reine Testen der technischen Funktio-

nalität hinaus. Auch die Dauer der Einarbeitung in die Entwicklungsumgebung, der testgetriebene Arbeitsansatz und die Lernerfolge werden betrachet. Probleme, die hier auftreten, können ebenso weitreichende Auswirkungen auf das Robot-Building-Lab haben, wie fehlerhafte Software oder defekte Roboter. Ein solches Problem kann auftreten, wenn die Schülerinnen sich nicht in angemessener Zeit in der Programmierung zurechtfinden.

5.1.2 Aufgabenstellung

Die Aufgabe für die Schülerinnen besteht darin, einen Roboter innerhalb einer simulierten Welt um einen Kasten fahren zu lassen. Hierführ steht ihnen ein Laser-Range-Finder als Sensor und zwei Motoren als Aktoren zur Verfügung.

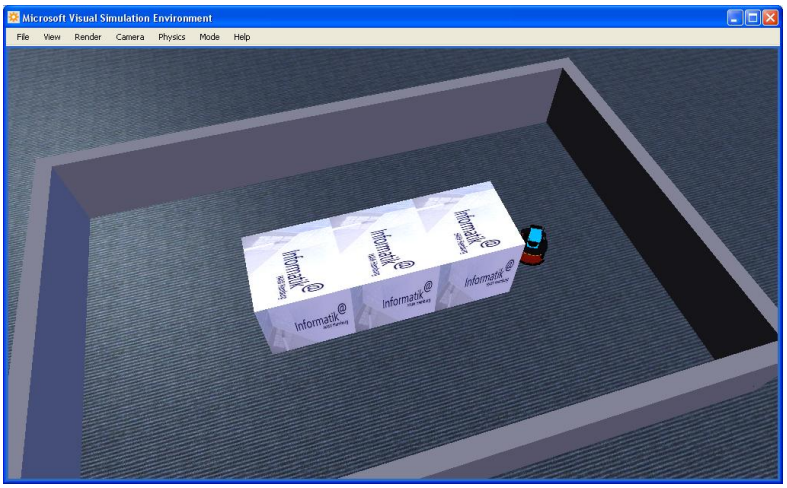

Abbildung 5.2: Simulation mit Roboter

5.1.3 Ablauf

Der Kurs für die Schülerinnen ist für einen Tag, also 8 Zeitstunden, angesetzt. Der erste Teil des Kurses besteht aus einer theoretischen Einführung in die Informatik und Roboter. Zunächst werden den Schülerinnen allgemeine Informationen zum

Fachbereich Informatik und Robotern gegeben, in dem unterschiedliche Roboterarten, Projekte der Hochschule und Möglichkeiten des Einsatzes von Robotern in einem Vortrag dargestellt werden. Anschließend wird die Aufgabe vorgestellt und eine interaktive kurze Einführung in die Programmiersprache und die Entwicklungsumgebung gegeben. Nach diesem Theorieteil beginnen die Schülerinnen in Zweierteams an den Rechnern zu arbeiten. Bevor die eigentliche Aufgabe bearbeitet werden kann, implentieren die Schülerinnen zunächst eigenständig und unter Anleitung der betreuenden studentischen Hilfskräfte Methoden zur Ansteuerung der Motoren. Dies geschieht im Prinzip durch Kopieren und Anpassen von in der Einführung verwendeten Methoden. Nach anfänglichen Schwierigkeiten lösen alle Teams die erste Teilaufgabe.

Nach der Mittagspause beginnen die Schülerinnen mit der eigentlichen Aufgabe, mit deren Bearbeitung die Schülerinnen bis zum Ende des Kurses beschäftigt sind. Bei der Bearbeitung zeigt sich die Heterogenität der Gruppe. Die Teams mit Mitgliedern, die Vorkenntnisse einbringen können, sind deutlich schneller in der Lage erste Lösungsversuche zu präsentieren. Die Teams ohne Vorkenntnisse müssen sich zunächst in der Programmierung zurechtfinden. Zur Unterstützung wird jedem Team individuelle, begleitende Hilfestellung durch die anwesenden studentischen Hilfskräfte gegeben. Dabei werden keine Lösungen durch die studentischen Hilfskräfte präsentiert, sondern nur Tipps gegeben. Auch zwischen den Teams herrscht ein reger Austausch über Probleme, die während der Bearbeitung auftreten. Am Ende des Kurses haben alle Teams eine lauffähige Lösung erarbeitet, einige Teams nehmen ihre Lösungen mit nach Hause, um dort weiter Programmieren zu können.

5.1.4 Gesamteindruck

Bewertung des Testlaufs

Der Testlauf im Rahmen der Herbsthochschule ist insgesamt sehr erfreulich. Die Schülerinnen konnten innerhalb der vorgegeben Zeit die Aufgabe lösen. Die Gruppen waren mit Begeisterung dabei und teilweise so in die Arbeit vertieft, dass sie

bei Unterbrechungen des Unterrichts diese gar nicht wahrgenommen haben und weiter arbeiteten.

Technische Bewertung

Bei der ersten Erprobung gab es kaum technische Probleme. Während der Testphase bemerkte ein Team, dass der Laser nicht punktgenau abzufragen war. Deshalb wurde noch während des Kurses das Sichtfeld des Roboters in Sektoren aufgeteilt, die vergleichbar mit der Funktionsweise von Sonar sind.

Umsetzungsproblem

Die Schülerinnen hatten keine Umsetzungsprobleme. Alle konnten sich zügig in der Materie zurechtfinden und schnell selbstständig arbeiten.

5.2 Szenarien

Die Evaluation des Projektes findet an zwei Hamburger Schulen statt, dem Kurt-Körber-Gymnasium in Hamburg-Billstedt und dem Luisengymnasium in Hamburg-Bergedorf. Das Hauptaugenmerk dieser Arbeit wird auf das Luisengymasium gerichtet, da hier im Gegensatz zum Kurt-Körber-Gymnasium die komplette Workbench zum Einsatz kommt.

5.2.1 Luisengymnasium

Das Luisengymnasium befindet sich im Hamburger Stadtteil Bergedorf. Die Schule liegt in einem ruhigen Wohngebiet direkt am Bergedorfer Gehölz. In einem vom Hamburger Architekten und Stadtplaner Fritz Schumacher entworfenen Gebäude lernen momentan ca. 900 Schülerinnen und Schüler. Das Luisengymnasium besitzt sehr unterschiedliche Bildungsschwerpunkte, beziehungsweise Bildungsprofile. Neben dem Musikzweig ab der 5. Klassenstufe gibt es einen naturwissenschaftlichen - technischen Wahlbereich. Außerdem ist das Luisengymnasium ein Aufbaugymnasium, dies bedeutet Schülerinnen und Schüler mit Realschulabschluß können,

nach einem Aufbaujahr, in die gymnasiale Oberstufe übergehen und ihre schulische Ausbildung mit dem Abitur abschließen. Der naturwissenschaftliche Sektor ist am Luisengymnasium sehr wichtig. Jede Disziplin, ob nun die Biologie, Chemie oder Physik, nähert sich dabei auf unterschiedliche Weise den jeweiligen Themen an, wobei Bewußtsein, Respekt und Verständnis für Natur und Umwelt geschaffen werden sollen. Schülerexperimente, deren Reflexion oder die Durchführung und anschließende Diskussion von Lehrerexperimenten, sind elementare Bestandteile des Unterrichts. Die interdisziplinären Wahlpflichtfächer 'Natur und Technik' in den Klassenstufen 5 bis 8, sowie das 'naturwissenschaftliche Praktikum' in den Klassen 9 und 10 bieten den Schülerinnen und Schülern die Möglichkeit sich naturwissenschaftliche und technische Themenkomplexe durch praktische Experimente zu erarbeiten. Zudem wird in den folgenden Jahrgangsstufen ein Ergänzungskurs 'Naturwissenschaftliches Experimentieren' angeboten und die Zusammenarbeit mit dem Bergedorfer Modell[1] ermöglicht.
In der Oberstufe werden in jedem Jahrgang Grund- und Leistungskurse in allen naturwissenschaftlichen Fächern bis zum Abitur angeboten. Nach der Reform der gymnasialen Oberstufe zum Schuljahr 2009/2010 wird es ein naturwissenschaftlich-informationstechnisches Profil geben. Das Angebot an Informatik soll im Zuge dieser Profilbildung erweitert werden. [Mäh07]

5.2.1.1 Wahlpflichtfach 'Natur & Technik'

Das Robot-Building-Lab kommt im Wahlpflichtkurs 'Natur und Technik' der Klassenstufe 9 zum Einsatz. Der Kurs wurde von vier Schülerinnen und 15 Schülern gewählt. Die Schülerinnen und Schüler haben keine Programmiervorkenntnisse. Ein Teil der Schüler hat in Klassenstufe 8 den Kurs 'Gestaltung' belegt, in dem u.a. HTML gelernt wird.

[1] Regionales Programm des Arbeitsamtes in Kooperation mit Bergedorfer Unternehmen, Schulen, dem Bezirksamt und Parteien, um Jugendliche in den Arbeitsmarkt zu integrieren. U.a. werden Praktika und Job-Patenschaften vermittelt.

5.2.1.2 Räumlichkeiten & technische Ausstattung

Abbildung 5.3: Physikraum Luisengymnasium

Der Unterricht am Luisengymnasium findet in den Physikräumen statt. Diese sind nur begrenzt für ein Robot-Building-Lab geeignet. Die Tische und Stühle sind in festen Reihen zum Lehrerpult hin ausgerichtet und es gibt keinen Platz, an dem eine Spielfläche für Roboter permanent aufgebaut werden kann. Die Spielfläche wird daher neben dem Lehrerpult für jede Unterrichtseinheit aufgebaut. Der Mangel an Platz in den Räumlichkeiten des Luisengymnasiums macht die Nutzung von Notebooks notwendig, da sonst Entwickeln und Roboterbau parallel nicht möglich ist. Die enge Anordnung der Tische und Stühle bereitet einige Probleme. Diese Anordnung erschwert es, der Lehrkraft und den Tutoren zu den einzelnen Teams zu gelangen, um ihnen Hilfestellung zu geben. Andere Arbeitsgruppen werden dadurch häufig gestört, weil hinter ihnen Bewegungen stattfinden. Auch ist die Unfallgefahr relativ hoch, da die Roboter und Notebooks sehr dicht am Gang stehen und ein unbeabsichtigtes herunterreißen sehr leicht möglich ist. Auch die Spielfläche ist für die einzelnen Teams schwierig zu erreichen. An der Spielfläche

selbst besteht die Gefahr, dass Schülerinnen und Schüler unbeabsichtigt auf die Roboter anderer Teams treten oder diese umstoßen, wenn sie mit dem Testen ihres eigenen Roboters beschäftigt sind. [Mäh07] Eine Anordnung der Tische und Stühle im Kreis oder Halbkreis mit der Spielfläche im Inneren wäre sinnvoll. Leider besitzt das Luisengymnasium keine entsprechenden Räume. [Gué06]

Die Notebooks sind mit dem Microsoft Visual Studio, dem Microsoft Robotics Studio und dem Brixc Command Center ausgestattet. Der zu programmierende Roboter ist der LEGO NXT.

5.2.2 Kurt-Körber-Gymnasium

Das Kurt-Körber-Gymnasium liegt im Hamburger Stadtteil Billstedt. Das Gymnasium wurde 1968 als Gymnasium Billstedt gegründet und trägt seit 2007 den Namen Kut-Körber-Gymnasium. Schülerinnen und Schüler mit Migrationshintergrund stellen etwa zwei Drittel der Schülerschaft. Zum Einzugsgebiet der Schule gehören neben den Stadtteilen Mümmelmannsberg und Billstedt auch die 1994 errichteten Dörfer für Aussiedler aus den ehemaligen Sowjetrepubliken wie beispielsweis Kasachstan.
Das Kurt-Körber-Gymnasium hat als eine der ersten Schulen Informatik in das Unterrichtsangebot aufgenommen. Seit 1982 wird Informatik unterrichtet und 1993 wurde ein Netzwerk für den Unterricht in Informatik und ITG[2] eingerichtet.
Das Kurt-Körber-Gymnasium sieht als zentralen Punkt seines Wirkens "die Entwicklung und Förderung der Gestaltungskompetenz der Schülerinnen und Schüler im Hinblick auf zukünftige Aufgaben und Anforderungen".[Mäh07, S. 16] Die Gestaltungskompetenz setzt sich dabei aus 3 Schlüsselkompetenzen zusammen; Selbstständiges erfolgreiches Handeln, erfolgreicher Umgang mit Kommunikations- und Wissensinstrumenten und erfolgreiches Handeln in sozial heterogenen Gruppen.

[2]Informationstechnische Grundbildung

5.2.2.1 Informatikunterricht

Das Robot-Building-Lab kommt im Informatikunterricht der Klassenstufe 8 zum Einsatz. Das Kurt-Körber-Gymnasium bietet im Gegensatz zu anderen Schulen schon in der Sekundarstufe I Informatikunterricht an. Die Klasse besteht zu etwa gleichen Teilen aus Schülerinnen und Schülern, insgesamt sind es 27. Die Schülerinnen und Schüler haben keine Programmiervorkenntnisse.

5.2.2.2 Räumlichkeiten & technische Ausstattung

Der Unterricht am Kurt-Körber-Gymnasium findet in den zwei vorhandenen Informatikräumen statt. Diese sind gut für ein Robot-Building-Lab geeignet. Die Tische und Stühle sind im Kreis angeordnet, mit der Blickrichtung zur Wand. Die Spielfläche steht in der Mitte des größeren Raumes. So ist ein sehr gutes Arbeiten möglich. Die Schülerinnen und Schüler können an den Rechnern konstruieren, programmieren und direkt zur Spielfläche gelangen, um das gerade Entworfene zu testen. Die Lehrkräfte können sehr schnell zu den jeweiligen Arbeitsplätzen gelangen und den Teams leicht Hilfestellung geben. Der einzige Nachteil ist, dass die Klasse aufgrund ihrer Größe auf zwei Räume aufgeteilt werden muss.

5.3 Erfahrungen aus der Schule

5.3.1 Luisengymnasium

5.3.1.1 Ablauf

Der Unterricht im Wahlpflichtfach „Roboter“ setzt sich aus zwei Einheiten zusammen. Im ersten Abschnitt wird ausschließlich mit dem Simulator des Microsoft Robotics Studio gearbeitet. Insgesamt dauert dieser Abschnitt neun Wochen und wird vor Weihnachten abgeschlossen. Im zweiten Abschnitt wird der LEGO NXT programmiert, dieser Abschnitt dauert 15 Wochen und wird am Ende des Schuljahres abgeschlossen.

Arbeiten mit dem Simulator

In der ersten Unterrichtseinheit wird eine theoretische Einführung in Informatik und Roboter gegeben. In einem Vortag wird zunächst eine Übersicht über allgemeine Informationen zum Fachbereich Informatik und Roboter. Roboterarten, Projekte und Möglichkeiten des Einsatzes von Robotern werden dargestellt. Im Anschluß stellen die beiden studentischen Hilfskräfte die Simulation des Microsoft Robotics Studio, die Entwicklungsumgebung Visual Studio, die Programmiersprache C# und die Aufgabenstellung vor; die Schülerinnen und Schüler sollen, wie auch die Teilnehmnerinnen der Herbsthochschule, einen Roboter innerhalb einer simulierten Welt um einen Kasten fahren lassen. Hierführ stehen ihnen ein Laser-Range-Finder und ein Kontaktsensor als Sensoren, sowie zwei Motoren als Aktoren zur Verfügung. Am Ende der ersten Unterrichtseinheit sind die ersten Fragen geklärt, die Schülerinnen und Schüler haben einen ersten Einblick bekommen und haben Gruppen mit einer Stärke von zwei bis drei Personen gebildet. Zu

Abbildung 5.4: Simulation mit Roboter

Beginn der zweiten Unterrichtseinheit beginnen die Schülerinnen und Schüler die

Methoden zur Steuerung der Roboter-Motoren zum implementieren. Nach dieser handwerklichen Übung beginnen die Teams die Aufgabe zu lösen. Die Begriffe der Aufgabenstellung werden geklärt und mögliche Probleme aufgezeigt. Die Bearbeitung der Aufgabe wird immer wieder individuell oder allgemein unterbrochen, um auftretende Probleme zu klären, wobei darauf geachtet wird, dass die Schülerinnen und Schüler einen möglichst grossen Anteil an Lösungsvorschlägen einbringen. Die Schülerinnen und Schüler sollen sich die Lösungen selbstständig erarbeiten. Die Lernbegleiter geben deshalb nur wegweisende Hilfestellung, präsentieren keine vollständigen Lösungen. Damit wird ihre Aufmerksamkeit für Programmierung und die dabei auftretenden Probleme geschult. Unbemerkt von den Schülerinnen und Schülern werden sie auf neue Betrachtungsperspektiven hingeführt, die für spätere Aufgabenstellungen nützlich sein könnten. Die Gruppen erkennen, dass in der simulierten Welt natürliche Gesetze, wie z.B. Schwerkraft, gelten und sie auf reale Probleme Rücksicht nehmen müssen. Am Ende der Doppelstunde haben alle Gruppen die Aufgabenstellung erfolgreich bearbeitet und können Lösungen präsentieren.

In der dritten Unterrichtseinheit wird eine Theorieeinheit abgehalten. Begriffe der Informatik werden geklärt und weiterführende Definitionen gegeben. Teile des von den Schülerinnen und Schülern erstellten Programmcodes werden im Plenum besprochen. Die Notebooks sind während dieser Phase des Unterrichts nicht in Betrieb, um Ablenkung zu vermeiden. Der bisherige Unterricht war dadurch gekennzeichnet, dass die Schülerinnen und Schüler weitgehend Programmteile reproduzieren, um einen ersten Eindruck von Programmierung und eine gewisse Sicherheit mit dem Umgang der Programmierumgebung zu bekommen. Durch das Zerlegen und Erklären des bisher erarbeiteten Programmcodes sollen die Schülerinnen und Schüler in die Lage versetzt werden, leichte Transferaufgaben zu bearbeiten und zu lösen. Deshalb werden innerhalb dieser Erklärungen Elemente der neuen erweiterten Aufgabenstellung mit den Schülerinnen und Schülern zusammen erarbeitet. Eigene Ideen und Fragen finden hierbei Berücksichtigung. Zum Ende dieser Einheit beginnen die Gruppen die erweiterte Aufgabenstellung zu bearbeiten. Dies-

mal nicht nur reproduzierend, sondern indem sie versuchen das vorher Gehörte anzuwenden. Hierbei stehen wieder die Lehkräfte und studentischen Hilfskräfte begleitend zur Verfügung.

In der folgenden vierten Unterrichtseinheit werden Theorieelemente wiederholt, um das bisher erlangte Wissen zu festigen. Die Aufgabenstellung wird erneut umrissen und offene Fragen der Klasse geklärt. Den restlichen Teil der Stunde bearbeiten die Schülerinnen und Schüler die Aufgabe. Die dabei entstehenden Fragen werden individuell oder im Plenum geklärt. Hierbei ist zu Beobachten, dass die Schülerinnen und Schüler sich gegenseitig bei der Lösung der Aufgabe unterstützen. Die Aufgabe betreffende Tipps und Erfahrungen werden ausgetauscht. Das Interesse der Schülerinnen und Schüler besteht dabei nicht allein in der Lösung ihrer eigenen Aufgaben und Probleme. Es entwickelt sich weiter zur Suche nach neuen selbstgestellten Aufgaben. Die Schülerinnen und Schüler probieren Vieles einfach aus. Daneben entwickeln die Schülerinnen und Schüler ein reges Interesse an der Simulation und verwenden relativ viel Zeit auf die freiwillige Hilfestellung bei anderen Gruppen. Diese gegenseitige Hilfestellung ist erwünscht und bietet mehr Vorteile für die Schülerinnen und Schüler, als die Hilfestellung durch die Lehrkraft oder die studententischen Hilfskräfte. Die Schülerinnen und Schüler können ihre Ideen miteinander diskutieren und eigene Vorschläge genauer betrachten. Wenn die eigenen Ideen und Gedanken mit der Lehrkraft besprochen werden, kann es schnell dazu kommen, dass die eigenen Ideen verworfen werden, da die Argumente der Lehrkraft als richtig angesehen werden. Die Gründe hierfür liegen zum einen in der gewohnten Unterrichtsform, sowie dem Verhältnis zwischen Lehrendem und Lernendem, zum anderen im vermeintlichen Erfahrungsschatz, den die Schülerinnen und Schüler der Lehrkraft unterstellen. Am Ende dieser Unterrichtseinheit sind viele Ideen der einzelnen Gruppen umgesetzt.

In der fünften Unterrichtseinheit setzen die Gruppen die Bearbeitung der Aufgabenstellung fort. Auffällig ist, dass alle Gruppen konzentriert bei der Sache sind, obwohl es schon die fünfte Doppelstunde ist, in der sie mit dem Simulator arbeiten.

Die Befürchtungen der Planungsphase, dass sich die Schülerinnen und Schüler in dieser Phase des Unterrichts zu langweilen beginnen und Anreize, sich zu beteiligen, veloren gehen könnten, erwiesen sich als unbegründet.

Während der sechsten Unterrichtseinheit wird das Nassi-Shneiderman-Diagramm[3] eingeführt. Nach der Zeit der praktischen Arbeit, soll durch eine Theorieeinheit das vorhandene Wissen wieder aufgefrischt und eine neue Sichtweise auf die Programmierung eingeführt werden. Das Nassi-Shneiderman-Diagramm ist hierfür sehr

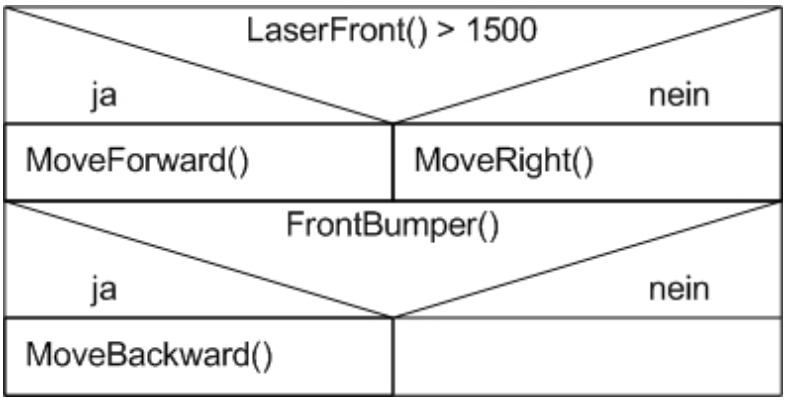

Abbildung 5.5: Nassi-Shneiderman-Diagramm

gut geeignet. Die nach diesem Modell aufgestellten Diagramme visualisieren die Strukturen des Programms. Einige Fragen der Schülerinnen und Schüler können so dargestellt und geklärt werden. Zunächst werden die Diagramme gemeinsam definiert und im Plenum mit Hilfe der Diagramme Beispiel-Programme erstellt. Nach dieser gemeinsamen Übung sind die Gruppen in die Lage versetzt, ihr eigenes Programm als Nassi-Shneiderman Diagramm darzustellen. Die Programme werden so übersichtlicher und die Schülerinnen und Schüler beginnen Strukturen und Abläufe der Programmierung zu begreifen. Auch sind nun die Programme

[3] Nassi-Shneiderman-Diagramme sind eine Entwurfsmethode für die strukturierte Programmierung, die Dr. Isaac Nassi und Dr. Ben Shneiderman 1972/1973 entwickelten. Das Gesamtproblem wird in immer kleinere Teile zerlegt, bis nur noch einzelne Anweisungen oder Kontrollstrukturen übrig bleiben.Vgl. auch [Deu85].

anderer Gruppen schneller zu überblicken und zu begreifen, die Fehlersuche wird erleichtert. Neue Ideen für die Programme werden zunächst auf dem Papier programmiert. Für die Klasse ist dies eine neue Erfahrung, der gegenüber nicht alle aufgeschlossen sind. Die Arbeit mit der Simulation ist für viele Gruppen deutlich reizvoller, da sie einen starken spielerischen Charakter hat. Das Programmieren auf dem Papier wurde von einigen als langweilig empfunden. Die Einführung des Nassi-Shneiderman-Diagramms ist jedoch wichtig, da so auf dieses Basiswissen in späteren Klassen zurückgegriffen werden kann und weitere Aspekte der Informatik eingeführt werden können. Außerdem besteht die Möglichkeit, ohne Computer zu programmieren. Ein weiterer Grund für die Einführung ist, dass den Schülerinnen und Schülern viele Teile der Programmierung, wie bespielsweise Schleifen, unklar sind. Die Diagramme sollen daher eine Hilfestellung im Umgang mit Programmierung und im Verständnis mit ihr geben.

Die Gruppen erstellen zu jeder Einheit Protokolle, in denen die Schülerinnen und Schüler darstellen, was sie gemacht haben, was funktionierte und welche Probleme es bei der Umsetzung gab. Die Protokolle gelten als Lernerfolgskontrolle und bilden die Grundlage für die spätere Benotung der Schülerinnen und Schüler. Die Protokolle geben sehr gute Hinweise auf Verständnisprobleme der einzelnen Gruppen, auf die in weiteren Unterrichtseinheiten eingegangen wird.

Arbeiten mit dem LEGO NXT

In der letzten Doppelstunde vor den Weihnachtsferien wird mit dem Zusammenbau der LEGO NXT begonnen. Den Schülerinnen und Schülern wird dabei keine Bauanleitung vorgelegt, sie sind in der Konstruktion des Roboters vollkommen frei. Einzige Bedingung ist, dass der Roboter einen tiefen Schwerpunkt haben soll, damit er eine Rampe bewältigen kann. Diese Freiheit in der Konstruktion führt dazu, dass sich die Gruppen Gedanken über die Nützlichkeit, Technik, Statik und das Design des Roboters machen. Da nicht alle Schülerinnen und Schüler über die gleichen Erfahrungen im Umgang mit LEGO verfügen, ist die Dauer der jeweiligen Bauphasen sehr unterschiedlich. Die Konstruktion des Roboters wird nach

den Ferien fortgesetzt. In den Ferien haben sich viele der Schülerinnen und Schüler weitere Gedanken über den Zusammenbau des Roboters gemacht, teilweise gab es Konstruktionszeichnungen oder Modellentwürfe mit eigenen LEGO-Steinen. Herausgestochen ist eine Gruppe, die sich in den Ferien neben den Gedanken zur Konstruktion des Roboters auch Gedanken zur Programmierung des Roboters gemacht hat. Ihre komplette Programm-Idee haben die Schüler auf drei DIN A4 Seiten niedergeschrieben. Dies unterstreicht noch einmal die Bedeutung der Nassi-Shneiderman Einheit. Die Gruppe präsentiert in der folgenden Doppelstunde ihren Programmentwurf, ohne die Hilfsmittel Computer oder Roboter zu verwenden. Dies geschieht einzig mit Tafelschwamm und Hindernisparcours aus Tafelkreide auf dem Lehrerpult. Das Beispiel verdeutlicht die Begeisterung der Schülerinnen und Schüler für die Thematik und das spielerische Erlernen von informatischen Grundlagen. Die Klasse benötigt noch zwei weitere Doppelstunden, um den Bau des Roboters abzuschließen. Während dieser Zeit finden bereits erste Funktionstests der Roboter statt. Die Lehrkraft und die studentischen Hilskräfte geben den Schülerinnen und Schülern individuelle Konstruktionstipps.

Microsoft Robotics Studio

Nachdem die Roboter fertiggestellt sind, stellen die studentischen Hilfskräfte die Verbindung zwischen Computer und LEGO NXT vor. Alle Gruppen verbinden nach anfänglichen technischen Schwierigkeiten den Roboter mit dem Laptop. Einige Computer können zunächst keine Verbindung mit dem LEGO NXT aufbauen. Die Deaktivierung der windowseigenen Firewall schafft bei fast allen Abhilfe, ein Notebook muss komplett neu eingerichtet werden. Zum Ende der Doppelstunde können einige Gruppen schon mit der Programmierung beginnen. Die Schülerinnen und Schüler arbeiten weiterhin in der für sie gewohnten Entwicklungsumgebung Visual Studio und mit der Programmiersprache C#. Hierbei zeigt sich, dass die Schülerinnen und Schüler sich zwar noch gut in der Entwicklungsumgebung zurecht finden, jedoch viele der bereits im Unterricht behandelten Programmierinhalte nicht mehr präsent sind. Dies scheint durch die lange Bauphase und die

Schulferien bedingt zu sein. Deshalb werden in der folgenden Unterrichtseinheit die Grundlagen der Programmierung wieder aufgefrischt. Die geschieht anhand von Lösungen der ersten Aufgaben, die im Plenum besprochen werden. Nach dieser kurzen Auffrischung beginnen die Gruppen mit der Programmierung des Roboters. Aufgabe ist es, den Roboter nur unter zu Hilfenahme des Kontaktsensors um einen Kasten fahren zu lassen. In den folgenden Unterrichtseinheiten sind die

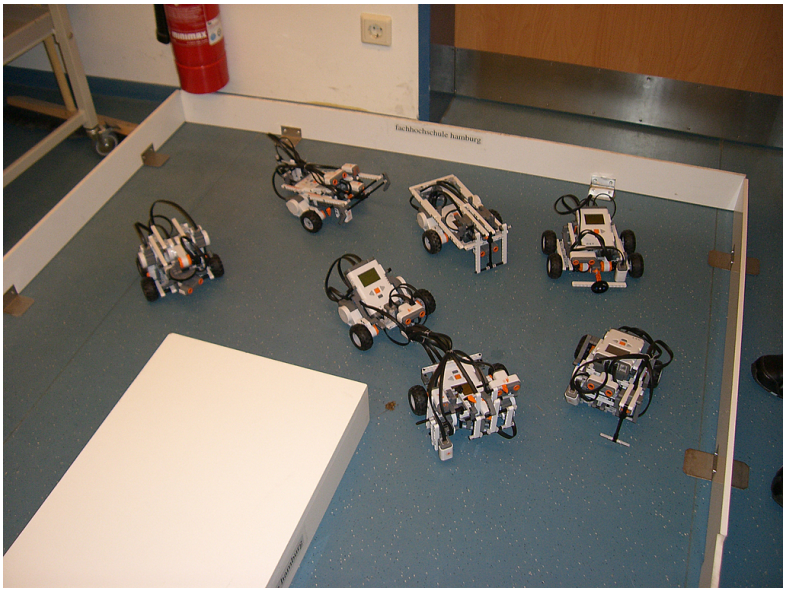

Abbildung 5.6: Von Schülerinnen und Schülern konstruierte Roboter in der Welt

Gruppen mit der Programmierung und der baulichen Anpassung des Roboters beschäftig. Viele Gruppen bemerken bei den ersten Tests ihrer Programme, dass ihr Roboter-Design noch nicht den Anforderungen genügt. Typische Probleme sind schleifende Räder, Roboter, die nicht in der Spur bleiben oder nicht gut um Kurven fahren können. Auch haben manche Teams den Kontaktsensor so angebaut, dass er nicht auslösen kann. Neben diesen sehr wertvollen Erfahrungen des Testens und Ausprobierens ist diese Phase des Unterrichts auch durch, für die Schülerinnen und Schüler sehr frustierende, technische Probleme gekennzeichnet. Die Bluetooth-Verbindung zwischen Roboter und Computer erweist sich als sehr fehlerträchtig. Signale werden teilweise nicht oder mit starker Verzögerung übertragen. Die von

den Schülerinnen und Schülern erstellten Programme funktionieren nicht, da der Roboter beispielsweise nicht 'merkt' dass er gegen die Wand gefahren ist.

Umstieg von C# auf NXC

Die anhaltenden technischen Probleme machen es unmöglich die Roboter in Echtzeit auf die Welt reagieren zu lassen. Deshalb wird drei Unterrichtseinheiten vor Ende des Schuljahres ein Wechsel der Programmiersprache und Entwicklungsumgebung gewagt. Der Umstieg erfolgt weg vom Microsoft Robotics Studio mit C# hin zu dem Bricx Command Center mit NXC. Um den Umstieg für die Schülerinnen und Schüler so leicht wie möglich zu gestalten, sind alle Methoden in der neuen Umgebung genauso benannt und erwarten auch die gleichen Argumente wie die Methoden in der alten Umgebung. Die Syntaxen von C# und NXC ähneln sich in dem von den Schülern genutzten Teilen sehr stark, Kontrollstrukturen, Variablen, Variablenzuweisungen und logische Abfragen werden identisch dargestellt.

Zu Beginn der Stunde wird den Schülerinnen und Schülern eine Einführung in die neue Entwicklungsumgebung gegeben, die Unterschiede und Gemeinsamkeiten werden benannt. Auch wird gezeigt, wie der Roboter mit dem Computer verbunden werden kann. Die neue Verbindung ist kabelgebunden. Das erstellte Programm wird in eine Sprache übersetzt, die der Roboter verarbeiten kann und anschließend auf den Roboter geladen. Auf die neue Programmiersprache wird kaum eingegangen. Es werden nur die neuen Methoden zum Auslesen der neuen Sensoren eingeführt. Für die Schülerinnen und Schüler ändert sich in diesem Sinne also nur die Entwicklungsumgebung, die Programmiersprache ändert sich in der Wahrnehmung der Schülerinnen und Schüler nicht.

Nach dieser kurzen Einführung beginnen die Gruppen mit der Programmierung der Roboter. Vereinzelt kommt es zu Verständnisfragen bezüglich der Bedienung der Entwicklungsumgebung. Diese können individuell geklärt werden. In der Mitte der Doppelstunde fahren die ersten Roboter. Am Ende der Unterrichtseinheit haben alle Gruppen Fortschritte bei der Lösung der ursprünglichen Aufgabenstellung

gemacht, oder eigene Aufgabenstellungen entwickelt.

In den letzten beiden Unterrichtseinheiten entwickeln die Gruppen ihre Programme weiter. Einige schaffen es, den Roboter sauber durch die Welt fahren zu lassen, andere kommen nicht so weit. Ein Team arbeitet sehr vertieft an der Lösung eigener Aufgaben und vergisst hierüber die eigentliche Aufgabe in Angriff zu nehmen.

5.3.1.2 Gesamteindruck

Bewertung des Unterrichts

Der Unterricht am Luisengymnasium verlief insgesamt sehr erfreulich. Die Schülerinnen und Schüler arbeiteten an Aufgaben, die zum einen selbstgestellt und zum anderen von der Lehrkraft vorgegeben wurden. Neue Lösungswege wurden immer wieder selbstständig gesucht, wobei nach dem Versuch und Irrtum Prinzip vorgegangen wurde. Die Klasse wurde während des ganzen Kurses nicht müde, sich mit dem Thema Robotik auseinanderzusetzen. Das große Interesse der Schülerinnen und Schüler an der Thematik führte dazu, dass sie unbewusst viele Grundlagen der Informatik erlernten und sich mit Technikproblemen und Designfragen auseinandersetzten. Die Schülerinnen und Schüler arbeiteten weitgehend eigenständig, wobei sie ständig Hilfestellungen und Begleitung durch den Lehrer oder die anwesenden studentischen Hilfskräfte bekommen konnten. Komplexe Fragestellungen, die auftraten, wurden im Plenum geklärt. Die Erfahrungen im Umgang mit dem Computer und dem Programmiercode geben den Schülerinnen und Schülern Sicherheit in ähnlichen Situationen und ermöglichen ihnen andere Problemstellungen selbstständig zu lösen.

Die Reaktion der Schülerinnen und Schüler auf das Thema Roboter war sehr gespannt. Die erste Neugier, was sie in diesem Kurs erwarten würde, wurde nach kurzer Zeit in Interesse gewandelt. Die Gruppenarbeit wurde, wie bereits erwähnt, möglichst selbstständig gehalten. Viele Gruppen konnten die Lehrkräfte mit ihren Lösungen überraschen. Der Ideenreichtum und die Einsatzbereitschaft einiger

Schüler ging dabei so weit, dass in den Weihnachtsferien Lösungen und Programme für den realen Roboter überlegt und verschriftlicht wurden, die dann nach den Ferien präsentiert wurden. Ein Gruppe Schülerinnen hat mit einer extrem kurzen und prägnanten Lösung für Begeisterung gesorgt. Die spielerische Lernatmosphäre im Kurs führte dazu, dass sich Ideen besser als im herkömmlichen Unterricht entfalten konnten. Der Lerncharakter wurde nicht durch strenge Vorgabe des Unterrichtmaterials bestimmt, sondern durch Anregungen der Schülerinnen und Schüler mitgestaltet. Die Klasse hatte vor Beginn des Kurses keine Erfahrung im Programmieren, die fehlenden Grundlagen mussten daher weiterhin in den Unterricht eingestreut werden. Dies darf aber nicht zu Lasten des spielerischen Charakters gehen, da die Schülerinnen und Schüler hierdurch unbewußt viel Wissen sammelten. Die Schülerinnen und Schüler haben nicht nur fachliche Kompetenzen erworben. Auch im sozialen Bereich konnte die Klasse Erfahrungen sammeln. Durch die selbstständig durchgeführte Gruppenarbeit konnten viele Lernaspekte wie beispielweise Planung der Vorgehensweise, Entscheidungen, Verteilung und Präzisierung von Aufgabenstellungen, die Einhaltung von Absprachen und die Kommunikation und das Vertrauen zwischen den Gruppenmitgliedern, vereint werden. Die unterschiedlichen Perspektiven, die sich durch die Gruppenarbeit ergaben, mussten besprochen werden, so dass die Gruppenmitglieder lernten, eigene Ideen einzubringen und sich gegebenenfalls zurückzustellen. Diese Fähigkeiten werden besonders dann wichtig, wenn die gesamte Gruppe, wie es hier der Fall ist, sich neu in ein Thema einarbeiten muss.

Technische Bewertung

Im Gegensatz zum Vortest sind im Verlauf des Unterrichts am Luisengymnasium größere technische Probleme aufgetreten. Die Probleme betrafen hauptsächlich das Microsoft Robotics Studio, welches sich während des Projektes noch in der Entwicklung befand. Monatlich wurde eine neue Version veröffentlicht, die vorherige Probleme löste, aber auch neue aufwarf. Dies führte dazu, dass die vorhandene Klasse 'Robot' permanent angepasst werden musste. Im Verlauf des Projektes wurden sechs Versionen des Robotics Studio veröffentlicht und die Entwicklung wird

von Microsoft weiter vorangetrieben. Gerade für einen Einsatz in der Schule mit den beschränkten Personalresourcen ist dies als sehr problematisch einzustufen. Dies bemerkten die Schülerinnen und Schüler kaum, für die Lehrkräfte bedeutete dies einen großen Arbeitsaufwand, der neben der Unterrichtsvorbereitung zu leisten war. Neben den ständigen Änderungen bereitete die Ansprache der Roboter über das Robotics Studio die größten Probleme. Die Bluetooth-Verbindung erwies sich als sehr fehlerträchtig und machte schlußendlich eine Programmierung in Echtzeit unmöglich. Der Simulator hingegen erwies sich als sehr gutes Werkzeug und bereitete kaum Probleme.

Umsetzungsproblem

Die Schülerinnen und Schüler sind nach anfänglichen Schwierigkeiten gut mit den Aufgaben und der Programmierumgebung zurecht gekommen. Ein Grundverständnis für informatische Problemstellungen konnte entwickelt werden. Um die Schülerinnen und Schüler besser zu unterstützen, ist es erforderlich, das vorhandene Lernmaterial weiter auszubauen und zu überarbeiten. So kann der große Schatz an Wissen, den sich die Schülerinnen und Schüler im Verlauf des Kurses angeeignet haben besser bewahrt werden.

Auch die Lehrkraft ist insgesamt gut mit der Programmierumgebung zurecht gekommen. Ein gewisses Maß an Unsicherheit blieb jedoch über weite Strecken des Kurses erhalten, da die Einarbeitungszeit sehr kurz und die Dokumentation nicht ausreichend war. Um dies für weitere Kurse dieser Art zu verbessern, muss ein Augenmerk auf die Lehrerfortbildung und Schulung geworfen werden.

Bewertung Umstieg

Der Wechsel der Programmierumgebung und -sprache verlief äußerst erfolgreich. Die Schülerinnen und Schüler konnten nach einer sehr kurzen Eingewöhnungsphase ihre Arbeit fortsetzen. Die Programmierprobleme, die mit der neuen Umgebung auftraten waren identisch mit den Problemen, die sie vorher hatten. Sie sind also

nicht auf den Wechsel zurückzuführen. So konnte das angesammelte Wissen der Schülerinnen und Schüler bewahrt werden und die Programmierung ohne größere zeitliche Verluste fortgesetzt und weiter entwickelt werden.

5.3.2 Kurt-Körber-Gymnasium

Am Kurt-Körber-Gymnasium ist die Bewertung ähnlich positiv zu sehen. Bis auf die technische Umsetzung sind keine nennenswerten Unterschiede festzustellen. Da die Roboter nach Vorlage gebaut wurden, wurde nicht so viel Zeit auf die Konstruktion verwendet. Technisch verlief das Projekt störungsfreier, da komplett auf NXC gesetzt wurde.

Nacht des Wissens

Die Schülerinnen und Schüler hatten die Möglichkeit ihre selbsterstellten Roboter auf der Nacht des Wissens[4] vorzustellen. Die Präsentation war ein sehr großer Erfolg und im Vorfeld eine sehr große Motivation für die gesamte Klasse.

[4]Die Nacht des Wissens bietet einmal im Jahr der interessierten Öffentlichkeit die Möglichkeit bei Hamburger Wissenschaftseinrichtungen hinter die Kulissen zu blicken.

Abbildung 5.7: Gäste der Nacht des Wissens

6 Zusammenfassung & Ausblick

In dieser Arbeit ging es darum, ein mobiles Robot-Building-Lab für den Einsatz im schulischen Umfeld zu entwickeln. Evaluiert wurde diese Workbench an zwei Hamburger Gymnasien, in der Hauptsache am Luisengymnasium in Hamburg-Bergedorf und als Quervergleich, in abgeänderter Form, am Kurt-Körber-Gymnasium in Hamburg-Billstedt.

Im Gegensatz zu traditionellen Formen der Wissenvermittlung, die auf Frontalunterricht und dem Frage-Antwort-Prinzip beruhen, setzt das Robot-Building-Lab auf einen spielerischen und selbstgesteuerten Ansatz. Dieser Ansatz führt dazu, dass Schülerinnen und Schüler vorurteilsfrei an mathematische, informatische, naturwissenschaftliche und technische Problemstellungen herangeführt werden. Des Weiteren fördert diese Art des Unterrichts Kompetenzen, die für das spätere Berufsleben unabdingbar sind. Diese Kompetenzen sind beispielsweise Teamfähigkeit, Eigenverantwortlichkeit, die Analyse komplexer Aufgabenstellungen, die daraus resultierende Abstraktionsfähigkeit und Kreativität.

In der Evaluation zeigte sich, dass erste informatische Grundlagen vermittelt wurden und die Schülerinnen und Schüler komplexe, teilweise selbstgestaltete, Aufgabenstellungen kreativ und eigenverantwortlich lösen konnten. Um sich vollständig auf die Informatik-Anteile des Robot-Building-Labs zu konzentrieren und der Umstand, dass simulierte Welten eine starke Ähnlichkeit zu Videospielen besitzen, führte dazu, dass in der ersten Unterrichtsphase mit dem Simulator des Mircrosoft Robotics Studio gearbeitet wurde. Das spielerische Heranführen an die Informatik baute erste Berührungsängste ab. In der zweiten Phase des Unterrichts wurde

der reale Roboter zunächst mit dem Microsoft Robotics Studio programmiert. Technische Probleme bei der Ansteuerung der Roboter führten zu einem Wechsel der Programmierumgebung und der Programmiersprache. In den verbleibenden Unterrichtseinheiten wurde mit NXC gearbeitet, welches in diesem Bereich ausgereifter ist. Der Ansatz zunächst, in der Simulation des Microsoft Robotics Studio zu arbeiten und dann den realen Roboter mit NXC zu programmieren hat sich als gangbarer Weg erwiesen. So konnten die Schülerinnen und Schüler zunächst in der simulierten Welt auf die realen Probleme vorbereitet werden. Die Vorteile beider Entwicklungsumgebungen konnten kombiniert werden.

Das mobile Robot-Building-Lab ist eine gute Möglichkeit informatische Inhalte curricular im Unterricht zu verankern. Ein deutlicher Hinweis ist die Tatsache, dass der Kurs im folgenden Schuljahr weitergeführt wird. Die Aufnahme der Informatik in den regulären Stundenplan einer Schule ist wichtig, um die informatische Grundbildung zu verstetigen. Dies, kombiniert mit dem spielerischen Unterrichtsansatz, fördert das Grundinteresse an mathematischen, informatischen, naturwissenschaftlichen und technischen Berufsbildern und kann so einen wichtigen Baustein für die Ingenieurausbildung liefern.

Die Zukunft wird zeigen, inwieweit Projektansätze, wie das Robot-Building-Lab die Ingenieurausbildung fördern und stärken können und so helfen, den Fachkräftemangel in den mathematischen, informatischen, naturwissenschaftlichen und technischen Disziplinen zu beheben. Eine empirische Betrachtung der Zusammenhänge zwischen dem Einsatz informatischer Schulprojekte, dem Lernerfolg und der späteren Studienwahl kann für die weitere Entwicklung von informatischen Lernkonzepten nützlich sein. Dies beschränkt sich nur auf die Informatik, sondern kann auf viele andere Themengebiete angewendet werden.

Literaturverzeichnis

[Bec00] Beck, Kent: *Extreme Programming.* Addison-Wesley, 2000

[BM96] Brauer, Wilfried ; Münch, Siegfried: *Studien- und Forschungsführer Informatik.* 3. Edition. Berlin : Springer Verlag, 1996

[Bro86] Brooks, Rodney: A Robust Layered Control System for a Mobile Robot. In: *IEEE Journal of Robotics and Automation* 2 (1986), März, Nr. 1. – auch MIT AI Memo 864, September 1985

[CS01] Claus, Volker ; Schwill, Andreas: *Duden Informatik.* 3. Edition. Mannheim : Dudenverlag, 2001

[Deu85] Deutsches Institut für Normung e.V. DIN: *Informationsverarbeitung, Sinnbilder für Struktogramme nach Nassi-Shneiderman.* November 1985. – DIN 66261

[Gué06] Guérin, Philippe: Raumanalyse für die Luisenschule Bergedorf / HAW Hamburg. 2006. – Forschungsbericht

[Han07] Hansen, John: *Not eXactly C (NXC) Programmer's Guide.* Version 1.0.1 b31, Juli 2007. – http://bricxcc.sourceforge.net/nbc/ - Zugriffsdatum 12.08.2007

[HH03] Freie und Hansestadt Hamburg, Behörde für Bildung und S. *Rahmenplan Wahlpflichtfach Informatik.* 2003

[Ie04] für Informatik e.V., Gesellschaft. *Digitale Spaltung verhinderrn - Schulinformatik stärken.* 2004

[Ie06] FÜR INFORMATIK E.V., Gesellschaft. *Was ist Informatik? Unser Positionspapier*. 2006

[Köc05] KÖCKRITZ, Oliver: *Visuomotorische Bewegungskoordination für mobile Roboter*, HAW Hamburg, Diplomarbeit, 2005

[Koc03] KOCH, Birgit: *Einsatz von Robotikbaukästen in der universitären Informatikausbildung am Fallbeispiel "Hambuger Robocup: Mobile autonome Roboter spielen Fußball"*, Universität Hamburg, Diplomarbeit, 2003

[LEG] LEGO: *Dokumentation LEGO NXT*. http://mindstorms.lego.com/Overview/. – Zugriffsdatum 11.08.2007

[Leh02] LEHR, Thomas: *C#: Spracheigenschaften und Vergleich mit Java*. 2002. – FH Wedel - http://www.fh-wedel.de/ si/-seminare/ws02/Ausarbeitung/4.csharp/csharp0.htm - Zugriffsdatum: 11.08.2007

[May07] MAYO, Merrilea J.: Games for Science and Engineering Education. In: *Communications of the ACM* 50 (2007), Juli, Nr. 7, S. 31–35

[Mäh07] MÄHL, Inga. *Zwischenbericht - Robot-Building-Lab*. 2007

[Mic] MICROSOFT: *Microsoft Robotics Developer Center*. – http://msdn2.microsoft.com/de-de/robotics/default.aspx - Zugriffsdatum: 11.08.2007

[Sto01] STOLT, Matthias: Roboter im Informatikunterricht / HAW Hamburg. 2001. – Forschungsbericht

[Wei05] WEICKER, Nicole: *Material zum vierten Vorlesungstermin "Didaktik der Informatik"*. 2005. – Universität Stuttgart

A Materialiensammlung

Roboter an Schulen - Materialiensammlung

Kai Rosseburg

22. August 2007, Hamburg

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Microsoft Robotics Studio

Das Microsoft Robotics Studio ist ein „Software Development Kit“ für die Entwicklung von Roboter-Programmen. Es stellt eine Sammlung von Diensten und Klassen bereit, mit denen man eine große Anzahl von unterschiedlichen Roboter-Plattformen programmieren kann.

Unterstützt werden bisher Roboter von Fischer Technik, Pioneer und LEGO.

Das Robotics Studio abstrahiert vollständig von der anzusprechenden Roboter-Hardware. So ist es ohne Probleme möglich ein Programm, das einen Pioneer ansteuert für einen LEGO NXT zu benutzen.

1.1 Die Simulation

Durch den integrierten Simulator ist es möglich auch ohne Roboter Programme zu entwickeln und zu testen. Der Simulator ist eine physikalisch korrekt berechnete 3D-Umgebung, die mit in directX programmiert und leicht anpassbar ist.

1.1.1 Simulations-Dateien

Die Quelltext- und Projekt-Dateien der simulierten Welt befinden sich im Ordner:

```
C:\Microsoft Robotics Studio (1.0)\samples\SimulationTutorials\Tutorial2
```

Die folgenden Tabelle gibt einen kurzen Überblick über die Dateien und ihre Bedeutung.

Dateiname	Bedeutung
AssemblyInfo.cs	Info-Datei der einzelnen Programmteile
Simulation Tutorial 2.htm	Ursprüngliche Beschreibung des Simulation Tutorial 2 von Microsoft
SimulationTutorial2.cs	Quelltext der simulierten Umgebung
SimulationTutorial2.csproj	Projekt-Datei
SimulationTutorial2.csproj.user	Benutzer-Datei
SimulationTutorial2.sln	Interne VisualStudio-Datei
SimulationTutorial2.sou	Interne VisualStudio-Datei

Für das Benutzen und Erstellen eigenner Simulationen sind nur SimulationTutorial2.csproj und SimulationTutorial2.cs von Interesse. Nachdem die Projekt-Datei geöffnet wurde kann die die Quelltext-Datei SimulationTutorial2.cs bearbeitet werden.

1.1.2 Gestaltung eigener Simulations-Welten

Zur Gestaltung eigener Simulations-Welten stehen folgende Methoden zur Verfügung:

void AddSky()
void AddGround()
void AddCameras()
void AddBox(Vector3 position, Vector3 dimensions, float mass, String name)
void AddTexturedBox(Vector3 position, Vector3 dimensions, float mass, String name)
void AddPioneer3DXRobot(Vector3 position)
private void PopulateWorld()

AddSky() fügt der simulierten Welt einen Himmel hinzu.

AddGround() fügt der simulierten Welt einen einfachen, planen Boden hinzu.

AddCameras() fügt der simulierten Welt eine Kamera hinzu. Ohne Kamera kann die Simulation nicht betrachtet werden.

AddBox(position, dimensions, mass, name) ist die Methode mit der eine texturlose Box in die Simulation eingefügt werden kann. Die Methode erwartet vier Argumente *position* und *dimensions* vom Typ *Vector3*, *mass* vom Typ *float*, sowie *name* vom Typ *String*.
Mit *position* wird die Position in einem dreidimensionalen Koordinatensystem angegeben, mit *dimensions* wird die räumliche Ausdehnung bzw. Größe der Box festgelegt. Die Positions- und Größenangaben beziehen sich jeweils auf den Mittelpunkt der Box. Mit *mass* wird die Masse der Box festgelegt, je größer die Masse desto gringer ist die Wahrscheinlichkeit, dass der Roboter sie umfahren kann.
Jedes Objekt in der simulierten Welt braucht einen eindeutigen Namen, deshalb muss dieser mit *name* übergeben werden.

AddTexturedBox(position, dimensions, mass, name) erwartet die exakt gleichen Argumente wie *AddBox(position, dimensions, mass, name)*, der einzige Unterschied ist, dass mit *AddTexturedBox(position, dimensions, mass, name)* texturierte Boxen in die Welt eingefügt werden können. Momentan ist ist die Textur noch fest implementiert und kann nicht verändert werden.

AddPioneer3DXRobot(position) ist die Methode mit der ein Modell des Pioneer-Roboters in die simulierte Welt eingefügt werden kann. Die Methode erwartet das Argument *position* vom Typ *Vector3*, welches die Position des Roboters innerhalb der simulierten Welt festlegt. Das Modell des Pioneer verfügt über einen Kontaktsensor (Bumper) und einen LaserRangeFinder.
PopulateWorld() ist die Methode in der die simulierte Welt 'zusammen gebaut'

wird. Alle o.g. Methoden müssen innerhalb der *PopulateWorld()* aufgerufen werden.
Eine beispielhafte *PopulateWorld()* könnte wie folgt aussehen:

```
private void PopulateWorld()
{
AddSky();
AddGround();
AddCameras();

AddBox(new Vector3(0.0f, 0.5f, 0.0f), new Vector3(5.0f, 1.0f, 0.2f), 1, "box");

AddPioneer3DXRobot(new Vector3(0, 0.1f, -2));
}
```

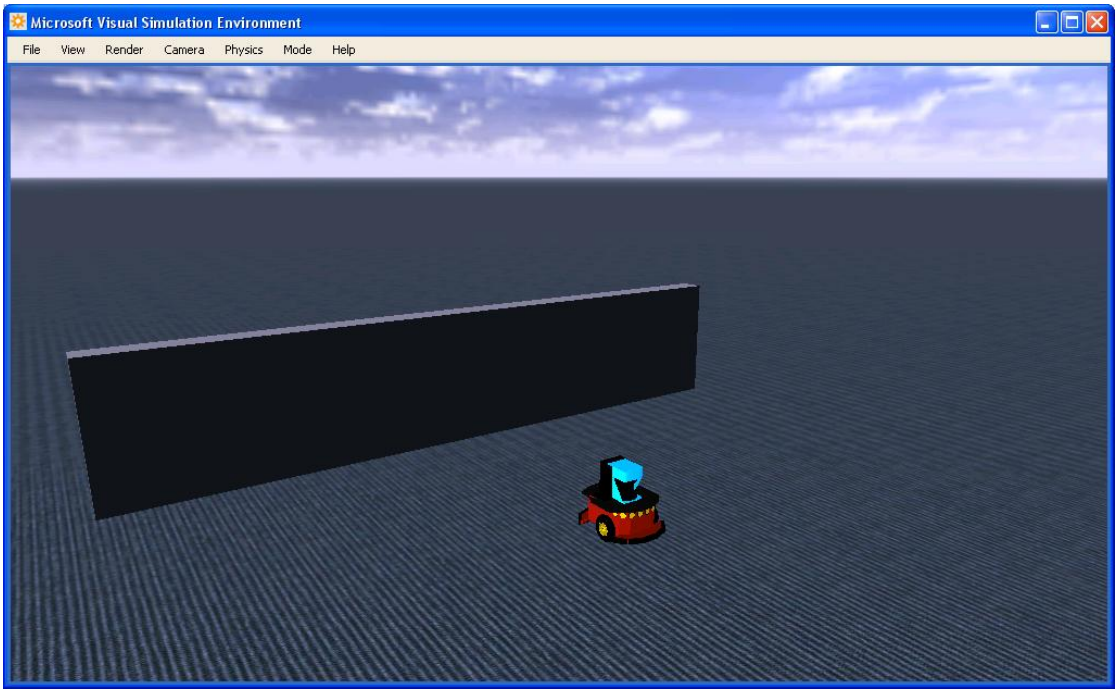

Abbildung 1.1: Die resultierende einfache simulierte Welt

1.2 Verhaltensbasierte Robotik

1.2.1 Verhaltens-Dateien

Die Quelltext- und Projekt-Dateien befinden sich im Ordner:

```
C:\Microsoft Robotics Studio (1.0)\samples\RoboticsTutorials\Tutorial3\CSharp
```

Die folgenden Tabelle gibt einen kurzen Überblick über die Dateien und ihre Bedeutung.

Dateiname	Bedeutung
AssemblyInfo.cs	Info-Datei der einzelnen Programmteile
Robotics Tutorial 3.htm	Ursprüngliche Beschreibung des Robotics Tutorial 3 von Microsoft
RoboticsTutorial3.cs	Quelltext der simulierten Umgebung
RoboticsTutorial3.csproj	Projekt-Datei
RoboticsTutorial3.csproj.user	Benutzer-Datei
RoboticsTutorial3.sln	Interne VisualStudio-Datei
RoboticsTutorial3.sou	Interne VisualStudio-Datei

Um den Roboter steuern zu können sind nur RoboticsTutorial3.csproj und RoboticsTutorial3.cs von Interesse. Nachdem die Projekt-Datei geöffnet wurde, kann die Quelltext-Datei RoboticsTutorial3.cs bearbeitet werden.

1.2.2 Roboter-Programmierung

Folgende Methoden können verwendet werden:

void RobotDrive(double left, double right)
void MoveForward()
void MoveBackward()
void MoveLeft()
void MoveRight()

int LaserLeft()
int LaserFront()
int LaserRight()
bool FrontBumper()
bool RearBumper()

RobotDrive(left, right) ist die Methode mit der der linke und der rechte Motor des Roboters angesprochen werden können. Die Methode erwartet zwei Argumente *left* und *right* vom Typ *double*, die die Geschwindigkeit des jeweiligen Motors darstellen.

Zu Beginn einer Unterrichtseinheit sind die Methoden *MoveForward()*, *MoveBackward()*, *MoveLeft()* und *MoveRight()* noch nicht implementiert. Die Implementierung kann von den Schülern geleistet werden, indem die o.g. Methode *RobotDrive(left, right)* benutzt wird.
Als sinnvoll haben sich folgende Werte für die einzelnen Methoden erwiesen:

MoveForward()	*RobotDrive(0.4, 0.4)*
MoveBackward()	*RobotDrive(-0.4, -0.4)*
MoveLeft()	*RobotDrive(0.0, 0.2)*
MoveRight()	*RobotDrive(0.2, 0.0)*

Diese Werte sind nur als Richtwerte zu verstehen und können im Laufe der Unterrichtseinheit den jeweiligen Bedürfnissen angepasst werden.

In der Simulation verfügt der Roboter über einen Laser Range Finder, der zur Abstandsmessung verwendet wird. Der LRF deckt einen Bereich von 180° ab. Mit den Methoden *LaserLeft()*, *LaserFront()* und *LaserRight()* geben die Entfernung zum nächsten linken, mittigen oder rechten Hindernis in mm zurück.

Sowohl in der realen als auch in der simulierten Welt verfügt der Roboter über einen Kontaktsensor, den sog. Bumper (in der simulierten sind es sogar zwei - ei-

ner vorne und einer hinten). Die Methoden *FrontBumper()* für den vorderen und *RearBumper()* für den hinteren Bumper geben einen Wahrheitswert zurück.

"Bumper ist gedrückt."$\Longrightarrow$ WAHR
"Bumper ist nicht gedrückt."$\Longrightarrow$ FALSCH

B Erfahrungsberichte aus den Schulen

Erfahrungsbericht aus dem Luisengymnasium

Die Praxisphase der Unterrichtseinheit gliederte sich in die folgenden Abschnitte:

- Arbeiten mit dem Microsoft-Robotics-Studio und der Programmiersprache C# in der Simulationsumgebung
- Aufbau der Lego-NXT-Roboter
- Steuerung der Roboter über das Microsoft-Robotics-Studio

Die Praxisphase wurde eingeleitet mit einem Vortrag von Prof. Dr. Kai von Luck der einen Bogen spannte von der Verwendung von Mikrochips in den Gegenständen des täglichen Lebens bis hin zum Einsatz von Robotern. Den Umgang mit der Simulationsumgebung erlernten alle Schüler sehr schnell. Die studentischen Mitarbeiter stellten dazu die nötigen Handgriffe und Befehlsfolgen kurz vor, die Schüler vollzogen diese dann nach. Bei der Bearbeitung der gestellten Programmieraufgaben zeigte sich aber sehr schnell ein differenzierteres Bild. Während eine Gruppe von zwei Mädchen in kürzester Zeit mit einem überraschend knappen Programm die Aufgabe meisterte, den virtuellen Roboter ohne „Crash“ um ein Hindernis fahren zu lassen, ergaben sich bei anderen Gruppen unterschiedlichste Programmier- und Verständnisprobleme. Zum Teil gelang es den Schülern durch Hilfestellung untereinander die Probleme zu lösen, in anderen Fällen war der Rat der Lehrkraft und die Hilfestellung der Mitarbeiter der HAWnötig. Schwierig bei dieser äußerst eigenständigen Gruppenarbeit ist, dass beim gleichzeitigen Auftreten von Schwierigkeiten in mehreren Arbeitsgruppen diese in eine Warteschleife gelangen, die den Fortschritt der Arbeit in dieser Gruppe behindert, Frustration erzeugen kann und auch ein Auseinanderlaufen der Wissensstände des Kurses zur Folge haben kann. Bisher ist es gelungen, eine zu starke Differenzierung innerhalb der Schülergruppe zu verhindern. Die Motivation der Schüler durch die Simulationsumgebung ist allerdings auch außerordentlich

hoch, nicht zuletzt durch ihre Ähnlichkeit mit einem Computerspiel. Dadurch war die Gruppe bisher durchaus bereit, sich sehr weit auf die Auseinandersetzung mit dem theoretischen Hintergrund einzulassen und auch hartnäckig bei auftretenden Schwierigkeiten am Programm weiterzuarbeiten. Einzuräumen ist, das sich der zunächst aufgestellte Katalog des theoretischen Hintergrundwissens als zu anspruchsvoll erwies. Der Begriff der Variablen als Speicher für Werte wurde noch akzeptiert, der Sinn der verschiedenen Typen von Variablen blieb für die Schüler jedoch zunächst zum großen Teil im Dunklen. Brav wurden auch die verlangten Nassi-Shneiderman-Diagramme angefertigt, die Nützlichkeit diesesWerkzeuges zur Programmanalyse erschloss sich den Schülern aber bisher nicht vollständig. Auch der Begriff der Funktion war bisher nur schwer zu vermitteln. Dabei können die Schüler Variablen und Funktionen in ihren Programmen rein pragmatisch sinnvoll benutzen und als Werkzeug zur Problemlösung einsetzen. Wichtig wird in der Fortsetzung der Unterrichtseinheit sein, durch ein fortgesetztes Hin- und Herschalten zwischen praktischer Arbeit und theoretischer Reflektion auch das theoretische Verständnis weiterzuentwickeln, ohne den spielerischen Charakter zu verlieren. Gerade hier, das zeigt die bisherige Erfahrung, liegen die Stärken des Projektes.

Beim praktischen Aufbau der Roboter mit den Lego-Bausätzen zeigte sich der hohe Aufforderungscharakter des Materials. Die Schüler und Schülerinnen waren mit Feuereifer bei der Sache und konstruierten ausgesprochen kreativ; sie lösten sich dabei sehr schnell völlig von den von Lego mitgelieferten Vorlagen und entwickelten eigene, unterschiedliche Modellvarianten, die den Vorgaben: ein möglichst niedriger Schwerpunkt und die Fähigkeit auf eine Rampe zu fahren, Rechnung tragen sollten. Nicht alle Konstruktionen erwiesen sich als uneingeschränkt brauchbar; die Vorzüge und Schwachstellen der entwickelten Roboter diskutierte Prof. v. Luck aus der Ingenieurssicht in einer Doppelstunde mit den Schülern. Interessant ist in diesem Zusammenhang die Bemerkung einer Schülerin: „Lego-Technik kannte ich bisher nur vom Einwickeln der Schachteln als Geschenk, wenn ich zu einem Jungen zum Geburtstag eingeladen wurde.“ Die „gefühlte Kompetenz“ der Jungen war

hier zunächst durch den vertrauteren Umgang mit dem Technikbaukasten größer, ein Faktum, das in vielen Bereichen des naturwissenschaftlichen- und Technikunterrichtes festzustellen ist. Dennoch konstruierten auch die Mädchen ausdauernd und erfolgreich mit. Dieses nicht zuletzt deshalb, da sie durch ihre lauffähigen Simulationsprogramme in der Einführungsphase eben doch Ansätze eines positiven Selbstkonzeptes in diesem Bereich entwickeln konnten.

Die Roboter über die Bluetooth-Schnittstelle anzusprechen gelang im Wesentlichen problemlos. Schwierigkeiten ergaben sich aber beim Übertragen der Signale der Sensoren vom Roboter via Bluetooth auf den Rechner; z.T. wurden hier Signale völlig verschluckt oder aber zeitlich so stark verzögert, dass eine sinnvolle Reaktion nicht programmierbar war.

Die Ursache dieses Problems muss jetzt gefunden und behoben werden. Für die Schüler waren diese letzten Schulstunden, in denen sich dieses unberechenbare Verhalten der Roboter zeigte, sehr unbefriedigend. Deutlich zeigt sich an dieser Stelle der Forschungscharakter des gesamten Vorhabens. In dieser Form ist das Projekt noch nicht praxistauglich.

Es wäre außerordentlich schade, wenn sich bei der Analyse der Signalübertragung zeigen sollte, dass hier ein grundsätzliches Manko der verwendeten Hardware liegt, hat sich doch der Einstieg über die Simulationsumgebung und die Programmierung mit dem Robotics-Studio unter der Verwendung von C# als sehr tragfähig und uneingeschränkt positiv erwiesen.

Werner Baum, Mai 2007

Erfahrungsbericht aus dem Kurt-Körber-Gymnasium

Am Kurt-Körber-Gymnasium ist, abweichend von der Stundentafel für Gymnasien in Hamburg, das Fach Informatik ab der 8. Klasse Pflicht für alle Schüler. Für die Teilnahme an diesem Projekt wurde eine 8. Klasse ausgewählt, so dass die Schülerinnen und Schüler noch keine Vorkenntnisse mitbrachten. Im ersten Halbjahr erfolgte eine Einführung in die Grundlagen der Textverarbeitung und in das Arbeiten mit Präsentationssoftware. Zu Beginn des zweiten Halbjahres wurde zunächst die Herkunft des Begriffs "Roboteräusführlich geklärt und anhand von Alltags-Algorithmen an den Algorithmusbegriff herangeführt. Danach erfolgte eine sechsstündige Einführung in das Programmieren anhand eines speziell für den Anfangsunterricht geeigneten Robotermodells (WinNiki). Es ermöglicht das Kennenlernen von Befehlsabläufen, die Abfrage einfacher Sensoren und das Formulieren von Wiederholungsanweisungen (ohne Zählschleife) und Fallunterscheidungen.
Für das Arbeiten mit den NXT-Robotern kamen verschiedene Oberflächen infrage. Das Robotics Visual Studio von Microsoft konnte nicht verwendet werden, da die vorhandenen Grafikkarten in den Schulcomputern nicht die gewünschten Anfordungen erfüllten. Erst im Herbst diesen Jahres wird die Anschaffung neuer Notebooks möglich werden. Aus diesem Grund fiel die Entscheidung zugunsten des Einsatzes von NXC, der Nachfolgeumgebung von NQC. Auch diese lief zunächst nicht problemlos, da sie die Installation eines Treibers für die Roboter benötigt, der in der LEGO-Software enthalten ist. Diese konnte nur unter Schwierigkeiten installiert werden, da nur XP Service pack 1 auf den Schulrechnern vorhanden war und die Software eine Schreibberechtigung in den Ordner Ëigene Dateien "verlangt. Dieser war jedoch aus pädagogischen Gründen vor den Schülern verborgen worden, damit sie die Rechner nicht verbotenerweise mit Spielen oder anderem belasten.

Für die Schülerarbeit stehen 10 Baukästen für 27 Schülerinnen und Schü-

ler zur Verfügung. Sie durften ihre Gruppenzugehörigkeit frei wählen, wobei keine gemischt-geschlechtliche Gruppe gebildet wurde. Zur Zeit arbeiten eine beamtete Lehrkraft (Frau OStR' Klemm), eine Pädagogin, die das Projekt begleitet, und eine Schülerin aus der 12. Klasse am Projekt mit. Die Schülerin hat donnerstags in den ersten beiden Stunden keinen Unterricht und ist freiwillig und unentgeldlich anwesend, betreut die Achtklässler bei technischen Schwierigkeiten und überwacht die Aus- und Rückgabe der Robotersets. Zeitweilig ist auch eine Informatikerin mit eingebunden, die als Quereinsteigerin in den Schuldienst im Herbst mit dem Vorbereitungdienst beginnen möchte. Wir hoffen, dass sie unserer Schule zugewiesen wird und dann im Rahmen des selbständig zu erteilenden Unterrichts in das NXT-Projekt einsteigen wird.

Die Schülerinnen und Schülern waren zunächst alle sehr begeistert. Sie bauen sehr konzentriert und fleißig. Zwei Gruppen waren frustriert von dem Gefühl, dass Teile im Bausatz fehlen, obwohl auch diese Schüler original verpackte Sätze erhalten hatten.

Mit der Zeit kristallisierten sich sich einige Schwierigkeiten heraus: Die NXT-Bausätze enthalten sehr viele Kleinteile und sind deutlich unübersichtlicher als die alten LEGO-Bausätze (die Roboter sind aber wesentlich stabiler, das ist ein Vorteil). Die Schüler suchen sehr lange nach dem richtigen Teil, das als nächstes eingebaut werden muss; die Doppelstunden wirken sehr kurz. Die Gesamtgruppe wird zum Basteln auf zwei benachbarte Räume verteilt. Eine angemessene Betreuung wäre durch eine einzige Lehrkraft kaum möglich. Ein Vorteil ist, dass den Schülern zwei große Arbeitsflächen (in der jeweiligen Raummitte) zur Verfügung stehen.

Die Durchführung einer Doppelstunde beginnt immer mit einem Plenum, in dem der Stand der einzelnen Gruppen bekannt gemacht wird. Danach werden Information über das Stundenziel an die Schüler gegeben und es erfolgt eine gemeinsame Erarbeitung der Sachinformationen, die benötigt werden, um das Stundenziel zu erreichen.

Dann erfolgt die Ausgabe der LEGO-Kästen und die Verteilung auf die beiden Räume. Da die Gruppen sehr unterschiedliche Arbeitsgeschwindigkeiten haben, ist schon ein deutliches inhaltliches Auseinanderdriften zu beobachten. Etwas ungünstig hat sich auf unseren Zeitplan ausgewirkt, dass wegen eines Frankreichaustausches am Anfang des Halbjahres einige Schüler zweimal gefehlt haben und zu einem Termin französische Gastschüler zusätzlich untergebracht werden mussten.

Die Gruppe bekam am Anfang als ferne Zielsetzung, eine Vorführung in der HAW am 9. Juni im Rahmen der "Langen Nacht der Wissenschaften zu gestalten", was einige Schüler zusätzlich stark motiviert.

Aufgrund der vielen positiven Erfahrungen soll das Projekt im nächsten Schuljahr unbedingt weitergeführt werden, wobei über die Klassenstufen erst nach der Lehrerzuweisung entschieden werden kann.

Christine Klemm, Mai 2007

C Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 23. August 2007	
Ort, Datum	Unterschrift