Autoren: Löwenstein, Bernhard / Di Angelo, Monika.

Titel: Lego Mindstorms-Roboter – Coole Klassenkameraden im Programmierunterricht.

Quelle: Blaschitz, Edith / Brandhofer, Gerhard / Nosko, Christian / Schwed, Gerhard (Hrsg.): Zukunft des Lernens. Wie digitale Medien Schule, Aus- und Weiterbildung verändern. Glückstadt 2012, S. 293-313.

Verlag: Verlag Werner Hülsbusch.

Die Zahlen in eckigen Klammern kennzeichnen das Seitenende der Originalausgabe.

Bernhard Löwenstein und Monika Di Angelo

Lego Mindstorms-Roboter – Coole Klassenkameraden im Programmierunterricht

Der Einstieg in die Programmierung erfolgt im schulischen Anfängerunterricht in der Sekundarstufe II traditionellerweise in Form eines Sprachkurses. Beginnend mit einem einfachen „Hello World"-Programm führen die Lehrenden bei dieser Vorgehensweise Schritt für Schritt die wesentlichen Sprachkonstrukte und Programmierkonzepte ein. Nicht selten müssen sie dabei aber feststellen, dass schon recht bald der Großteil der Lernenden das Interesse am eigentlich spannenden Thema der Computerprogrammierung verloren hat.

In den letzten Jahren entwickelten sich deshalb verschiedene alternative Ansätze zur Einführung von Anfängerinnen und Anfängern in die Programmierung. Einer davon ist der Einsatz von *Lego Mindstorms*-Baukästen. Solche Sets beinhalten einen programmierbaren Mikrocontroller, an den sich verschiedene Aktoren und Sensoren anschließen lassen. Ganz der *Lego*-Philosophie folgend lassen sich damit unterschiedliche Roboter zusammenbauen, deren Verhalten anschließend programmiert werden kann. Für die verschiedenen Zielgruppen stehen unterschiedliche Umgebungen und Sprachen zur Verfügung.

Der Beitrag geht der Frage nach, wie sich diese technologiegestützten Lernbehelfe zur Organisation und Gestaltung eines attraktiven und zeitgerechten Programmierunterrichts einsetzen lassen. Er beleuchtet weiterhin die Vor- und Nachteile und zeigt, dass die Nutzung von solchen Robotern nicht unbedingt Zukunftsmusik sein muss.

Einleitung

Die immer rasantere technologische Weiterentwicklung stellt besondere Herausforderungen an die Lehrenden technischer Fächer. War es früher noch ausreichend reines Faktenwissen im Schulunterricht zu vermitteln, so genügt dies heute nicht mehr. Zu schnell verliert das technische Wissen an Wert, darüber hinaus nimmt die Stoffmenge stetig zu. Auch der Zusammenarbeit [293] mit anderen Experten, sogar solchen auf fachfremden Gebieten, kommt im späteren beruflichen Umfeld immer größere Bedeutung zu. Um für die Zukunft gerüstet zu sein, müssen sich also Schülerinnen und Schüler neben den fachlichen Kompetenzen noch weitere aneignen. Als die wichtigsten Schlüsselqualifikationen neben der Fachkompetenz lassen sich die Methodenkompetenz, die Sozialkompetenz und die persönliche Kompetenz nennen (vgl. Klippert 2004: 39 ff.). Damit diese Qualifikationen trainiert und verbessert werden, ist allerdings eine entsprechende Umgestaltung des Unterrichts vonnöten. Anstatt wie bisher den Stoff „auf dem goldenen Tablett" zu servieren, müssen die Lernenden vielmehr zum eigenverantwortlichen Arbeiten und Lernen angeleitet werden. So gerüstet sollte es ihnen dann möglich sein, sich in einer Welt, die lebenslanges Lernen erfordert, bestens zurechtzufinden.

Speziell das Gebiet der Informatik ist von raschen Veränderungen betroffen. Das bisher Geschriebene gilt demnach für diese Disziplin ganz besonders. Leider wird dieser Tatsache aber nicht immer Tribut gezollt und an vielen Schulen nach wie vor nur kurzfristiges Anwenderwissen anstatt langfristigem Konzeptwissen vermittelt. Das ist zum einen darauf zurückzuführen, dass das Gelernte im Beruf direkt anwendbar sein soll, zum anderen besteht auf Seiten der Lehrenden die Angst, dass die Vermittlung von Konzepten zu abstrakt und wenig greifbar ausfällt. Genau so sollte Informatikunterricht natürlich nicht sein, denn ein solcher schreckt die Schülerinnen und Schüler vor der an und für sich interessanten Materie ab. Erfreulicherweise muss der Unterricht aber auch nicht so sein. Es steht heutzutage eine Vielzahl an technologiegestützten Lernbehelfen zur Verfügung, die einen interessanten Informatikunterricht ermöglichen und es gleichzeitig erlauben, nicht nur Anwenderwissen, sondern auch Konzeptwissen zu lehren. Des Weiteren lässt sich speziell der Informatikunterricht so gestalten, dass das eigenverantwortliche Arbeiten und Lernen im Zentrum steht, wenngleich dies möglicherweise gewisser organisatorischer Anpassungen bedarf.

Dieser Artikel beschäftigt sich damit, wie sich der schulische Programmierunterricht unter Berücksichtigung der bisher genannten Punkte attraktiver und nachhaltiger gestalten lässt. Weit verbreitet ist nach wie vor der klassische Sprachkurs, bei dem die Lernenden Schritt für Schritt mit den Konzepten einer Programmiersprache vertraut gemacht werden. Einigen Schülerinnen und Schülern mag diese Vorgehensweise zwar zusagen, doch viele verlieren schon recht bald das Interesse an der Computerprogrammierung. Zurückzuführen ist das vor allem auch auf die oftmals nicht besonders [294] attraktiven Beispiele, anhand derer die Konzepte eingeführt und in weiterer Folge vertieft werden. Der Einsatz von *Lego Mindstorms*-Roboter schafft Abhilfe.

Die coolen Klassenkameraden kommen nicht nur bei den Lernenden bestens an, sie leisten auch von didaktischer Seite her ihren Beitrag, damit sich Anfängerinnen und Anfängern das Thema Programmierung einfacher erschließt. Der Artikel gibt eine Einführung in *Lego Mindstorms* und stellt zwei konkrete Ansätze für den Schulunterricht vor: Einer setzt auf *NXT-G*, beim zweiten werden die Roboter mithilfe der Programmiersprache *Java* programmiert. Ein Vergleich zwischen *Lego Mindstorms* und anderen Vorgehensweisen rückt anschließend in den Fokus.

Lego Mindstorms im Überblick

Bei *Lego Mindstorms* handelt es sich um eine Produktreihe der dänischen Firma *Lego*. Als geistiger Vater der Baukästen gilt Seymour Papert – ein südafrikanischer Visionär, der sich schon in den späten 1960ern im Rahmen seiner Forschungstätigkeit damit beschäftigte, wie man Kindern und Jugendlichen auf interessante Weise die Programmierung näherbringen kann. Der *Lego*-Erfinder suchte eine interessante Anwendung für seine Programmiersprache und stieß dabei auf die Roboter. Mit *Lego* fand er einen passenden Partner zur Realisierung seines Vorhabens. Mittlerweile gibt es mehrere Generationen dieser Baukästen. Die aktuelle Version ist *Lego Mindstorms NXT 2.0* (Modellnummer: 8547), auf die auch dieser Artikel Bezug nimmt. Gemeinsam ist allen Sets, dass sie neben einer großen Zahl an *Lego-Technic*-Elementen jeweils einen programmierbaren Mikrocontroller beinhalten, an den sich verschiedene Aktoren und Sensoren anschließen lassen. Ganz der *Lego*-Philosophie folgend lassen sich mit einem solchen Baukasten unterschiedliche Roboter konstruieren, die den Kategorien Humanoide, Tier, Fahrzeug oder Maschine zugeordnet werden können. Über spezielle Programme lässt sich das Verhalten der Roboter programmieren. Für die verschiedenen Altersgruppen stehen unterschiedliche Umgebungen und Sprachen zur Programmierung bereit. Zur Faszination an diesen Baukästen trägt daher maßgeblich die große Zahl an Variationsmöglichkeiten bei. Darüber hinaus sind noch Erweiterungssets sowie Sensoren von Fremdanbietern erhältlich, die die Grenzen des Machbaren noch weiter ausdehnen. [295]

Das Herzstück des Systems stellt der programmierbare *NXT*-Stein dar. In ihm steckt ein 32-Bit-Mikroprozessor, der mit 48 MHz läuft und auf 256 KB Flashspeicher und 64 KB Arbeitsspeicher zurückgreifen kann. Auf dem *NXT*-Stein findet sich ein monochromes LCD mit einer Auflösung von 100 x 64 Pixel sowie vier Navigationsknöpfe, die sich bei Bedarf genauso wie das Display individuell programmieren lassen. Ein Lautsprecher komplementiert den Baustein. Für den Anschluss von Aktoren stehen drei Ausgangsports zur Verfügung, für Sensoren bietet er vier Eingänge. Zum Transfer des Programms vom Programmiercomputer auf den Baustein steht USB sowie das drahtlose *Bluetooth* bereit. Als Aktoren sind dem Baukasten drei Servomotoren beigepackt. Dank ihres Getriebes kann beim Betrieb zwischen Geschwindigkeit und Leistung gewählt werden. Des Weiteren verfügen sie über einen eingebauten Drehzahlmesser. Dem *NXT* 2.0-Baukasten sind ein Ultraschallsensor, zwei Tastsensoren und ein Farbsensor beigefügt. Ersterer erlaubt Abstandsmessungen, der Tastsensor reagiert

auf Druckkontakt und dem Farbsensor kommen sogar gleich drei Funktionen zu: er kann Farben und Lichteinstellungen identifizieren und als Lampe eingesetzt werden. Der früher mitgelieferte Geräuschsensor zur Messung des Schalldruckpegels ist nicht mehr im Lieferumfang enthalten. [296]

Abb. 1 Der „Alpha Rex" in schulischer Mission

Das auf *LabVIEW* basierende *NXT-G* stellt die standardmäßige Programmierumgebung für *Lego Mindstorms* dar. *NXT-G* ist ein typischer Vertreter für eine ikonische Programmierumgebung und zielt auf Kinder und Erwachsene ohne jegliche Programmiererfahrung ab. Von den Nutzenden ist hierbei kein Programmcode zu schreiben, sondern das Programm wird großteils mittels entsprechender Mausaktionen erstellt. Durch das Verschieben und Verbinden der einzelnen Funktionsblöcke entsteht nach und nach das gewünschte Programm. Sind Texteingaben vonnöten, erfolgen diese dialogbasiert. Neben Variablen stehen auch Kontrollstrukturen wie Schleifen und Verzweigungen in Blockform zur Verfügung. Zusätzlich lässt sich auf bestimmte Ereignisse warten und reagieren. *NXT-G* erlaubt die Umsetzung bereits recht anspruchsvoller Aufgaben.

leJOS NXJ ermöglicht die Programmierung der *NXT*-Roboter unter *Java*. Es stellt den Nutzerinnen und Nutzern alle benötigten Komponenten bereit – einerseits eine modifizierte Firmware, durch die auf dem *NXT*-Baustein eine JVM installiert wird, und andererseits ein API, über das die Entwickelnden Zugriff auf die einzelnen Roboterteile erhalten. Zusätzlich bietet *leJOS NXJ* noch verschiedene Werkzeuge, die beispielsweise den Firmwaretausch oder den Programmupload ermöglichen. Nachdem es ein *leJOS NXJ*-Plugin für *Eclipse* gibt, empfiehlt sich die Nutzung dieser frei verfügbaren

IDE. Nach der Installation des Plugins finden sich einige zusätzliche Menüpunkte in *Eclipse*, über die sich einfach und bequem die gleichen Aktionen wie mit den externen *leJOS*-Tools durchführen lassen. Da nach dem Firmwaretausch eine JVM ohne nennenswerte Einschränkungen auf dem *NXT*-Stein läuft, lassen sich alle Konzepte der Programmiersprache Java damit demonstrieren.

In den vergangenen Jahren hat *Lego Mindstorms* an vielen Schulen, Universitäten und sonstigen Ausbildungsstätten Einzug gehalten. Die Baukästen werden dabei zur Durchführung unterschiedlichster Projekte genutzt (vgl. Flowers & Gossett 2002: 45 ff.; Altman 2005: 56 ff.; Stolzlechner 2007: 3 ff.). Auch für den fächerübergreifenden Unterricht eignen sie sich bestens (vgl. Schmidt 2009: 20 ff.). Darüber hinaus konnten sich diverse Wettkämpfe etablieren, deren Anwendungsbereiche von Roboterfußball bis hin zur Algorithmenoptimierung reichen. Diese Bewerbe motivieren nicht nur Kinder und Jugendliche, sondern auch Erwachsene sich intensiver mit den Baukästen zu beschäftigen. [297]

Konkrete Ansätze für den Schulunterricht

Es folgen zwei konkrete Ansätze. Bei einem wird *NXT-G* zur Einführung der Schülerinnen und Schüler in die Programmierung verwendet, beim zweiten *leJOS NXJ*. Bei beiden werden die verschiedenen Konzepte isoliert eingeführt, ehe sie in Kombination anzuwenden sind. Des Weiteren nimmt die Wahlfreiheit der Teilnehmenden mit jeder Aufgabe zu. Beide Vorgehensweisen wurden in der Praxis ausprobiert. Die Versuche können als bestens gelungen bezeichnet werden.

Ikonische Programmierung / NXT-G

Dieser Ansatz gelangte im Februar 2012 an der Montessori-Schule KreaMont in St. Andrä-Wördern zur Ausführung. An dem Programmier-Workshop nahmen insgesamt zwölf Lernende der fünften bis achten Schulstufe teil. Die Schülerinnen und Schüler hatten keinerlei Programmiererfahrung vorzuweisen. Der Kurs dauerte fünf Schulvormittage jeweils von 8:30 bis 12:40 Uhr. Als Lehrende fungierten zwei Lehramtsstudierende, die sich tageweise abwechselten. Das Ziel des Kurses war die Teilnehmenden spielerisch mit der Computerprogrammierung vertraut zu machen und ein Interesse für diese Materie zu wecken.

Der erste Tag beginnt mit der Vorstellung verschiedener *Lego Mindstorms*-Roboter. Das geschieht einerseits durch eine Live-Präsentation verschiedener Robotertypen im Klassenzimmer, andererseits durch das Vorzeigen von Videos mit spektakulären Robotern. Die Lernenden bekommen dadurch einen ersten Eindruck, was sich mithilfe der Roboter alles anstellen lässt. Danach werden gemeinsam der grundsätzliche Aufbau eines *Lego Mindstorms*-Roboters analysiert und seine Hauptbestandteile unter die Lupe genommen. Die Schülerinnen und Schüler lernen hierbei die Bedeutung des programmierbaren Mikrocontrollers sowie der Motoren und Sensoren kennen. Danach dürfen sie sich auch schon selbst aktiv betätigen. In Zweiergruppen bauen sie den so-

genannten 30-Minuten-Roboter zusammen – ein einfaches Roboterfahrzeug, das aber bereits alle Sensoren an Bord hat. Damit werden dann die ersten Programmierübungen durchgeführt. Am Beginn steht eine kurze Einführung in die Programmierumgebung, bei der die Lernenden auch erfahren, was man überhaupt unter einer solchen versteht und was ein Programm ist. Unter Anleitung des/der Lehrenden entwerfen die Lernenden ihr erstes Programm, laden es auf den Roboter hoch und führen es aus. Es [298] bewirkt lediglich, dass der Roboter ein paar Sekunden geradeaus fährt und dann stehenbleibt. Dieses einfache Programm ist von den Kindern und Jugendlichen nun Schritt für Schritt zu erweitern, bis der Roboter ein am Boden mit Isolierband abgeklebtes Quadrat nachfährt. Verschiedene kreative Erweiterungen dieser Aufgabe sorgen für jede Menge Spaß unter den Teilnehmenden. So kann man beispielsweise den Roboter bei jeder 90-Grad-Drehung einen Sound von sich geben oder ihn einen Text am Display ausgeben lassen. Wunderbar lässt sich bei dieser Aufgabe das wichtige Konzept der Schleife einführen. Erfahrungsgemäß wird es in diesem Kontext auch schnell verstanden, da die Lernenden eine Notwendigkeit für ein solches Konzept erkennen können und es weiterhin für ein deutlich kompakteres Programm sorgt.

Der zweite Tag beginnt mit einer kurzen Wiederholung. Danach werden die einzelnen Sensoren genauer unter die Lupe genommen. Für jeden der drei Sensortypen demonstriert der/die Lehrende seine konkrete Anwendung anhand eines einfachen Beispiels. Darauf aufbauend bekommen die Schülerinnen und Schüler eigene Aufgaben zu lösen. So soll das Fahrzeug beispielsweise sofort stehen bleiben, wenn der Tastsensor gedrückt wurde. In Verbindung mit dem Ultraschallsensor soll sich das Fahrzeug selbstständig drehen, wenn es ein Hindernis vor sich wahrnimmt, um diesem auszuweichen. Mithilfe des Farbsensors sollen verschiedenfärbige Kugeln erkannt und in Abhängigkeit der Farbe jeweils eine unterschiedliche Aktion ausgeführt werden. Hierbei wird nun auch das Konzept der Verzweigung eingeführt, denn ohne dieses steht man bald an. Auch hier zeigen sich erfahrungsgemäß kaum Verständnisprobleme.

Der dritte Tag startet wiederum mit einer gemeinsamen Wiederholung des Stoffs vom Vortag und steht noch ganz im Zeichen der vorgegebenen Aufgaben, wenngleich zur Lösungsfindung nun jeweils mehrere Aktoren und Sensoren zusammenwirken müssen. So soll nun ein Programm entwickelt werden, durch das der Roboter einer beliebigen, am Boden festgeklebten Linie folgen kann. Wer das zufriedenstellend gelöst hat, darf aus seinem fahrenden Roboter den „RoboGator" (Roboterkrokodil) bauen und dieses gemäß den Vorgaben programmieren.

Am vierten Tag geht es mit einem Arbeitsblatt los, das verschiedene Aufgaben zu den Themen Schleifen und Verzweigungen vorgibt. Diese Aufgaben lösen die Lernenden genauso wie alle bisherigen Aufgaben im Zweierteam. Für den/die Lehrende bietet dies die Möglichkeit, vereinzelte Verständnisprobleme ausmachen und individuell darauf einzugehen. Danach geht es recht frei weiter. Jedes Team kann sich für einen der vorgegebenen Themen-[299]blöcke entscheiden und auf dem gewählten Gebiet eigenverantwortlich und großteils selbst entdeckend weiterarbeiten. So können die

Schülerinnen und Schüler nun beispielsweise den „Alpha Rex" (humanoider Roboter), eine Jukebox oder eine *Lego*-Gitarre bauen und programmieren. Hierzu gibt es zwar noch Anleitungen, diese müssen aber nicht unbedingt 1:1 umgesetzt werden. Die Kinder und Jugendlichen dürfen ihrer Fantasie freien Lauf lassen. Nachdem es sich hierbei um anspruchsvollere Aufgaben handelt, müssen die Lernenden zwischendurch noch mit dem Variablenkonzept vertraut gemacht werden. Das geschieht durch Einzelcoaching, sobald der/die Lehrende den Bedarf seitens der jeweiligen Gruppe feststellt.

Am letzten Tag setzen die Kinder und Jugendlichen ihre Arbeit vom Vortag fort und vollenden ihr Werk. Danach beginnen die Vorbereitungen für die Abschlusspräsentation, bei der jedes Team seinen in den letzten Tagen konstruierten Roboter präsentieren darf. Vor den Augen der restlichen Lehrerinnen und Lehrer sowie Schülerinnern und Schüler der Schule, diverser Vertreter aus Politik, Wirtschaft und Presse sowie der Eltern, Geschwister und Freunde ist das für die Kursteilnehmenden ein würdiger Abschluss nach den anstrengenden Tagen.

Textuelle Programmierung / leJOS NXJ

Dieser Ansatz gelangte im Jänner 2012 an der TU Wien zur Ausführung. An dem Programmier-Workshop nahmen insgesamt 27 Lernende der achten bis zehnten Schulstufe von vier verschiedenen Schulen teil. Die Schülerinnen und Schüler wurden gemäß ihrem Vorwissen auf zwei Kurse aufgeteilt. In Folge ist nur der Anfängerkurs mit dreizehn Teilnehmenden von Bedeutung. Er dauerte, genauso wie der Fortgeschrittenenkurs, vier Tage jeweils von 9:00 bis 16:30 Uhr. Als Lehrende fungierten an jedem Tag jeweils zwei bis drei verschiedene Lehramtsstudierende. Das Ziel des Kurses war die Teilnehmenden mit den wesentlichen Konzepten der objektorientierten Programmierung und im Speziellen mit deren konkreter Umsetzung in *Java* vertraut zu machen.

Am Beginn des ersten Tages wird der Ablauf für die vier Tage erklärt. Danach geht es zwecks Kennenlernen mit einer Art Speed-Dating weiter. Die Lehrenden mischen sich hierbei ebenfalls unter die Kinder und Jugendlichen. In mehreren Durchgängen versucht jedes für eine Runde zusammengeloste Paar jeweils eine Minute lang möglichst viel über sein/ihr Gegenüber zu [300] erfahren. Nachdem das erste Eis gebrochen ist, folgen eine Diskussion über das Thema Roboter sowie eine Demonstration verschiedener *Lego Mindstorms*-Roboter. Die Schülerinnen und Schüler werden dabei mit dem grundsätzlichen Aufbau einer solchen Maschine konfrontiert und erkennen, dass erst durch das Zusammenwirken von Hardware und Software so etwas wie situationsabhängiges Verhalten erreicht werden kann. Anschließend erklären die Lehrenden die wichtigsten Bauteile des *Lego Mindstorms*-Baukastens. Danach sind wieder die Schülerinnen und Schüler gefordert. Durch ein Rollenspiel verinnerlichen sie die Konzepte des sequentiellen Programmablaufs sowie des fix vorgegebenen Befehlssatzes. Zwei der Kursteilnehmenden ziehen dabei einen Drehstuhl, auf dem eine weitere Person sitzt. Die beiden Ziehenden repräsentieren jeweils einen Motor. Durch Zuruf von vor-

ab festgelegten Befehlen, die funktional mit den eigentlichen *Java*-Kommandos korrespondieren, wirken die Außenstehenden steuernd auf die „Motoren" ein und lösen so die vorgegebenen Aufgaben. Eine dieser Aufgaben sieht beispielsweise vor, dass der Drehstuhl in Quadratform im Raum bewegt wird. Anschließend wird der Schwierigkeitsgrad der Aufgaben noch etwas erhöht, indem die Schülerinnen und Schüler nicht mehr live ins Geschehen eingreifen dürfen, sondern die Lösung der Aufgabe vorab auf einen Zettel niederschreiben müssen, was man als Programmieren am Zettel bezeichnen könnte. Die am Drehstuhl sitzende Person liest nun die Befehle in der entsprechenden Reihenfolge vor und steuert so ihre „Motoren". Im letzten Schritt wird der bisherige Pseudocode in bereits sehr *Java*-nahen Code übergeführt – das erste *Java*-Programm ist somit entstanden, ohne dass eine einzige Zeile am Computer implementiert worden ist. Es folgt eine kurze Einführung in *Eclipse*. Für die ersten Schritte dort werden einfache, lauffähige Programme bereitgestellt, die das gleiche Verhalten der Roboter wie der Pseudocode davor bewirken. Unter Anleitung der Lehrenden nehmen die Kinder und Jugendlichen, wiederum in Zweierteams, kleine Ergänzungen und Adaptierungen an den Quellen vor, laden ihre modifizierten Programme auf den *NXT*-Stein hoch und führen sie aus. Um den Einstieg zu beschleunigen, werden bei diesem Ansatz fertig zusammengebaute Roboter vom Typ „Tribot" bereitgestellt.

Der zweite Tag beginnt mit einer kurzen Wiederholung des Stoffes vom Vortag. An diesem Tag sollen die Teilnehmenden ein gutes Verständnis für die objektorientierten Konzepte Klasse und Objekt entwickeln. Sie bekommen hierzu halbfertige Programme beigestellt. So kapselt eine eigene Roboterklasse die Funktionalitäten des Roboters, während in einer anderen [301] Klasse, die das Hauptprogramm repräsentiert, ein Objekt der Roboterklasse instanziiert und ihre Methoden zur Erfüllung der vorgegebenen Aufgaben aufgerufen werden. Manche der Methoden sind bereits fertig implementiert, andere müssen von den Lernenden noch korrigiert beziehungsweise mit sinnvollem Quelltext gefüllt werden. Ein Arbeitsblatt begleitet die Schülerinnen und Schüler bei ihren Schritten und gibt ihnen Halt und Orientierung. Aufgabenmäßig müssen sie hierbei wiederum Fahrbeispiele lösen. Gegen Ende des Tages wird noch behutsam das Prinzip der Vererbung eingeführt. Hierzu bekommen die Teilnehmenden eine weitere Klasse beigestellt, die sich direkt von der Roboterklasse ableitet und verschiedene Erweiterungen bietet. Nach Versuchen mit dieser neuen Roboterklasse sollen sie eine eigene Roboterklasse erstellen, die die wichtigsten Funktionalitäten von der Basisroboterklasse erbt. Erfahrungsgemäß klappt dies nach den Vorarbeiten ganz gut.

Am dritten Tag geht es erneut mit einer Wiederholung los. In der restlichen Unterrichtszeit rücken die Sensoren und Kontrollstrukturen in den Mittelpunkt des Geschehens. Zuerst wird der Tastsensor vorgestellt, danach der Ultraschallsensor und zu guter Letzt der Farbsensor. Nachdem die *leJOS*-API jeden dieser Sensoren jeweils durch eine eigene Klasse kapselt, sind keine allzu großen Verständnisprobleme zu erwarten. Dank der von den Kindern und Jugendlichen zu lösenden Aufgaben – beispielsweise soll der Roboter so lange geradeaus fahren, bis er auf ein Hindernis stößt und dann augenblicklich stehenbleiben – wird bald die Forderung nach neuen Konzepten laut.

Die typischen Kontrollstrukturen wie die While-Schleife und die If-Verzweigung lassen sich so einführen, ohne dass dies wie so oft „auf Vorrat" passieren muss. Darüber hinaus lernen sie auch Variablen kennen. Durch verschiedene kleinere Übungsaufgaben werden die neuen Konzepte vertieft. Idealerweise endet dieser Tag mit einer Führung durch ein regionales Unternehmen, in dem reale Industrieroboter zum Einsatz gelangen.

Der letzte Tag beginnt mit einem Stationenbetrieb, bei dem das Wissen der Schülerinnen und Schüler auf spielerische Weise überprüft wird. So bekommen sie beispielsweise mehrere kleine Blättchen überreicht, auf denen einzelne Codezeilen stehen. Diese müssen sie in die richtige Reihenfolge bringen, damit die Gesamtmethode die gewünschte Funktionalität erfüllt. Den Lehrenden dient das Ganze wiederum dazu, feststellen zu können, wo es noch Verständnisprobleme gibt. Bei ausgemachten Mängeln wird durch Einzelcoaching nachgeschult. Nach dem Stationenbetrieb geht es an die Ausarbeitung der Gesamtaufgabe. Die Kinder und Jugendlichen können hierbei zwischen vier Themen wählen: „RoboArt" (malender Roboter), „RoboDan-[302]ce" (tanzender Roboter), „RoboLab" (Labyrinth-durchfahrender Roboter) und „RoboPath" (Pfad-verfolgender Roboter). Auch dieser Ansatz endet analog zu vorhin mit einer Abschlusspräsentation.

Der vorgestellte Ansatz steht stellvertretend für viele weitere Ansätze mit *leJOS*, die sich in der Literatur finden lassen (vgl. Dietzel & Rinkens 2001: 193 ff.; Diethelm et al. 2003: 225 ff.; Granerud 2005: 40 ff.; Bagnell 2007: 58 ff.).

Vergleich mit anderen Ansätzen

Die Verwendung der *Lego Mindstorms*-Baukästen als Lernbehelf im Programmierunterricht ist mit diversen Pros und Kontras verbunden. Der Artikel nimmt folgend dazu Stellung und versucht die Vorteile und Nachteile mit den restlichen Ansätzen in Verbindung zu setzen. Zu nennen sind in diesem Zusammenhang besonders der klassische Sprachkurs sowie der Einsatz einer Mikrowelt zur Einführung in eine Programmiersprache (vgl. Löwenstein & Di Angelo 2011: 30 ff.).

Vorteile von Lego Mindstorms

Der Robotereinsatz im Unterricht wirkt stark motivierend auf Schülerinnen und Schüler (vgl. Dietzel & Rinkens 2001: 198). Schon seit jeher üben die Roboter, im Speziellen die menschenähnlichen Humanoiden, eine Faszination auf den Menschen aus und so ist es nicht verwunderlich, dass es für Lernende ein besonderes Erlebnis ist, wenn sie die Kontrolle über eine solche Maschine übernehmen können. Die Tatsache, dass diese im Vergleich zu den ausschließlich am Bildschirm ablaufenden, virtuellen Simulationen wie beispielsweise den Mikrowelten physikalisch angreifbar sind, trägt das Übrige dazu bei. Alles in allem stellen die Roboter also aus Schülersicht eine äußerst interessante Anwendung dar.

Ein weiterer großer Pluspunkt ist das direkte Feedback, das die Roboter an die Schülerinnen und Schüler zurückliefern (vgl. Barnes 2002: 148). Das Programm muss lediglich hochgeladen und aufgerufen werden. Danach kann auch schon beobachtet werden, ob es das macht, was es tun soll. Das Testen macht den Teilnehmenden so Spaß, und sie achten außerdem auf ein robusteres Programmdesign (vgl. Lawhead et al. 2002: 193). Bei einem Fehler kann es schon einmal passieren, dass der Roboter nicht nur sprichwörtlich gegen [303] die Wand fährt. Gerade dieses klare und direkte Feedback wirkt wiederum sehr motivierend auf die Kinder und Jugendlichen, da sie auf einen Blick erkennen können: Das implementierte Programm funktioniert – oder funktioniert nicht. Speziell im Vergleich zu den weit verbreiteten Sortieralgorithmen-Beispielen, die oft in Verbindung mit dem klassischen Sprachkurs zum Einsatz gelangen, ist das ein großer Vorteil. Auch bei den Mikrowelten kann es passieren, dass einer der Hauptdarsteller gegen die Wand läuft – aber das passiert halt nur virtuell.

Ganz besonders hervorzuheben ist, dass es sich bei *Lego Mindstorms* um ein hervorragend skalierbares Medium handelt. Das beginnt damit, dass sich die Roboter mit verschiedenen Sprachen und Umgebungen programmieren lassen und sich diese deshalb in Verbindung mit verschiedenen Zielgruppen einsetzen lassen. Das Spektrum reicht von Personen, die noch nie mit der Computerprogrammierung zu tun hatten, bis hin zu Studierenden, die komplexe Algorithmen wie beispielsweise Schwarmintelligenz-Verfahren anschaulich testen möchten. Des Weiteren können die Lehrenden dank der vielfältigen Verwendungsmöglichkeiten von *Lego Mindstorms* durch entsprechende Aufgabenwahl entscheidenden Einfluss auf den Schwierigkeitsgrad des Unterrichts nehmen (vgl. Hirst et al. 2003: 125 f.). Auch die gefürchtete Heterogenität innerhalb der Gesamtgruppe lässt sich beim richtigen Einsatz dieser Technologiestütze bestens kompensieren. Speziell wenn die Lehrenden Erfahrung mit *Lego Mindstorms* mitbringen, sollten sie aus dem Stegreif kreative Erweiterungen der Basisaufgaben von den schneller arbeitenden Schülerinnen und Schülern einfordern können. Das kommt bei diesen erfahrungsgemäß sehr gut an und die Adaptierungen werden meist gerne vorgenommen. Bei didaktisch richtigem Einsatz ist es auch überhaupt kein Problem, Lernende verschiedener Schulstufen gemeinsam zu unterrichten. Die gute Skalierbarkeit dieses Mediums gepaart mit der natürlichen Attraktivität der Roboter ist weiterhin ein Grund dafür, dass die Baukästen im Rahmen von Projekten eingesetzt werden, bei denen Gender-Lernunterschiede besonders beachtet werden sollen (vgl. Di Angelo 2011: 44 ff.) oder bei denen es um die Förderung von Hochbegabten geht (vgl. Di Angelo et al. 2010: 275 ff.). Auch beim klassischen Sprachkurs lassen sich die Beispiele der Zielgruppe gemäß anpassen, allerdings ist der Umgang mit einer heterogenen Gruppe deutlich schwieriger zu bewältigen. In Verbindung mit gängigen Mikrowelten muss überhaupt festgehalten werden, dass diese generell zu beschränkt sind, um über einen längeren Zeitraum im Unterricht eingesetzt werden zu können (vgl. Löwenstein 2011: 35). [304]

Abb. 2 Lego Mindstorms begeistert sowohl Mädchen als auch Burschen

Weiters ist von Vorteil – besonders im Vergleich zum klassischen Sprachkurs und den Mikrowelten –, dass die Kinder und Jugendlichen neben dem klassischen Desktop eine weitere Ausprägung des Computers im Schulunterricht kennen und programmieren lernen (vgl. McNally 2006: 62). Gerade in einer Zeit, in der immer mehr Geräte auf den Markt kommen, die vom Aussehen her immer weniger mit dem klassischen Computer zu tun haben, ist das wichtig. In Zukunft werden noch weitere, meist mobile Geräte erscheinen, die augenscheinlich immer weniger mit der ursprünglichen Rechenmaschine zu tun haben werden. Der interne Aufbau solcher Geräte korrespondiert aber nach wie vor mit dem des Computers.

Ebenfalls positiv zu erwähnen ist, dass die Schülerinnen und Schüler sowohl mit Hardware- als auch mit Softwarebelangen konfrontiert werden (vgl. Kurebayashi et al. 2006: 139). Durch das Erfordernis des Programmuploads wird für sie auch erkennbar, dass die Hardware und die Software grundsätzlich voneinander unabhängig sind. Trotzdem wird mit *Lego Mindstorms* sehr stark der Systemgedanke betont und vermittelt. Allzu oft lernen die Kinder und Jugendlichen im Schulunterricht Hardware und Software als etwas voneinander Isoliertes kennen. In Wahrheit ist das Gegenteil der Fall: ohne Hardware könnte Software nicht ablaufen, ohne Software würde Hardware keinen Nutzen erbringen. Die Grenzen zwischen diesen Welten ver-[305] schwimmen auch zunehmend (Stichwort: FPGA). Für die Endnutzerinnen und Endnutzer ist diese formale Unterscheidung auch belanglos. Der Erfolg von *Apple* zeigt, dass es heute um Gesamtsysteme geht, die für ihre Anwenderinnen und Anwender durch das geschickte Zusammenspiel zwischen Hardware und Software einfach nutzbar sein sollen. Durch den Einsatz von *Lego Mindstorms*-Robotern als Lernbehelf im Unterricht

wird das Zusammenspiel zwischen Hardware und Software für die Lernenden direkt nachvollziehbar. Nehmen sie beispielsweise an ihrem Roboter eine bauliche Veränderung vor, so sind nicht selten auch die Grenzwerte im Programm anzupassen, da die Sensoren plötzlich andere Werte liefern. Beim traditionellen Sprachkurs bekommen die Schülerinnen und Schüler von der Hardwareschicht nur am Rande etwas mit, bei Nutzung einer Mikrowelt kommt sogar noch eine weitere Abstraktionsschicht dazu.

Wer *NXT-G* als Programmierumgebung einsetzt, kann damit die Programmierkonzepte Variable, Schleife und Verzweigung vermitteln. Setzt man auf *leJOS NXJ*, so kann man überhaupt aus dem Vollen schöpfen, da der gesamte Funktionsumfang der Programmiersprache *Java* zur Verfügung steht. Dank des didaktisch vorbildlichen Aufbaus der *leJOS*-API können den Schülerinnen und Schülern auch die objektorientierten Konzepte gut vermittelt werden (vgl. Löwenstein 2010: 138 ff.). Und zwar existieren zu den realen Roboterobjekten virtuelle Programmierobjekte, über die die Lernenden die Umgebung abfragen und steuernd auf den Roboter einwirken können. Das Vorhandensein der realen Roboterobjekte und die Möglichkeit diese anfassen zu können, trägt maßgeblich zum Verständnis bei, was man sich unter einem Objekt vorzustellen hat. Nicht umsonst spricht man umgangssprachlich davon, etwas begriffen zu haben. Den Unterschied zwischen dem Klassen- und Objektbegriff kann man mithilfe der dem Baukasten in mehrfacher Ausführung beigelegten Servomotoren gut erklären. Gleiches gilt für die Vererbung. Hier erstellt man einen Roboter und fügt ihm eine weitere Funktionalität hinzu. Dieser physikalischen Erweiterung wird programmmäßig durch Ableitung von der ursprünglichen Roboterklasse Tribut gezollt. Erwähnt werden muss hier, dass sich die aufgezählten Konzepte auch durch einen Sprachkurs und unter Einsatz einer Mikrowelt gut vermitteln lassen. Ein Nachteil ist allerdings, dass man sich aus der virtuellen Welt nicht herausbewegt und die Schülerinnen und Schüler die Objekte dadurch nicht real anfassen können.

Zu guter Letzt lassen sich *Lego Mindstorms*-Baukästen und moderne Unterrichtsformen gemeinsam im Rahmen des Programmierunterrichts einsetzen. Die Arbeit in Zweierteams sehen fast alle Unterrichtskonzepte vor,[306] die auf die Roboter als Technologiestütze setzen. Das fördert die Sozialkompetenz der Schülerinnen und Schüler. Durch entsprechende Gestaltung der Unterrichtseinheiten, nämlich indem den Lernenden von Beginn an gewisse Freiräume zum Entdecken und Ausprobieren eingeräumt werden und mit Fortschreiten des Kurses die Aufgaben immer freier zu lösen sind, wird das eigenverantwortliche Lernen adressiert. Erfahrungsgemäß macht den meisten das Lösen der recht frei gestalteten Gesamtaufgabe großen Spaß, da sie hier ihrer Kreativität freien Lauf lassen können. Es zeigt sich aber auch, dass manche mit diesem Freiraum nicht umgehen können, da sie einen solchen im Schulunterricht nicht gewohnt sind. Das Lösen einer größeren Aufgabe erfordert von den Teilnehmenden aber auch bereits gewisses ingenieursmäßiges Arbeiten. So müssen sie beispielsweise vorab planen, was ihr Roboter können soll und auch die Arbeitsaufteilung mit dem zweiten Teammitglied erfordert ständige Koordination. Die Vorbereitung auf die Abschlusspräsentation und diese selbst schult ihre persönliche Kompetenz. Bei eini-

gen Teilnehmenden merkt man erfahrungsgemäß deutlich, dass sich ihr Stolz darüber, einen Roboter beherrschen zu können, positiv auf ihr Selbstvertrauen auswirkt und dies zu einem positiv veränderten Auftreten innerhalb des Klassenverbundes führt. Dank der vielfältigen Möglichkeiten, die *Lego Mindstorms* bietet, lässt sich der Unterricht weiterhin sehr abwechslungsreich gestalten. Den traditionellen Sprachkurs könnte man aus didaktischer Sicht auf ähnliche Beine stellen, leider wird das jedoch meist nicht gemacht, sondern die Konzepte werden den Schülerinnen und Schülern direkt vorgesetzt. Ansätze mit einer Mikrowelt als Einstiegsumgebung sehen moderne Unterrichtsformen zwar häufig vor, durch die Beschränktheit dieser virtuellen Welten werden sie den Nutzerinnen und Nutzern aber meist schnell langweilig.

Nachteile von Lego Mindstorms

Doch wo viel Licht, dort ist auch Schatten. So sind diese Baukästen natürlich nicht umsonst und deren Anschaffung stellt viele Schule vor Probleme (vgl. Brinda 2004: 31). Das *NXT 2.0*-Set kostet derzeit rund 250 Euro, die speziell für den Schulunterricht vertriebenen Educational Sets sind sogar noch teurer. Um sinnvoll arbeiten zu können, braucht jedes Zweierteam einen eigenen Baukasten. Bei Standardklassen sind da schnell sechs bis acht solche *Lego Mindstorms*-Baukästen vonnöten. Gerade in Zeiten, in denen das Schulbud-[307]get knapp ist, scheitern viele Schulen an der Finanzierung der Baukästen. Die Werkzeuge, die bei der Abhaltung eines Sprachkurses eingesetzt werden, sind zumindest für die Programmiersprache *Java* kostenlos erhältlich. Auch für die Nutzung der gängigen Mikrowelten fallen keine Kosten an.

Es ist weiterhin nicht möglich, alle Schülerinnen und Schüler mit einem solchen Baukasten auszurüsten, sodass diese auch zuhause damit arbeiten können (vgl. McNally 2006: 61). Die Verwendung der *Lego Mindstorms*-Roboter ist dadurch an die Öffnungszeiten des schulischen Computerlabors gebunden. Bestimmte Experimente wie Projekte außerhalb der Schulzeit sind somit nicht beziehungsweise nur schwer durchführbar. Diese Nachteile gibt es in Verbindung mit dem Sprachkurs und den Mikrowelten nicht. Nachdem die Tools kostenlos verfügbar sind, lassen sie sich beliebig vervielfachen und somit auch zuhause von den Kindern und Jugendlichen einsetzen.

Nicht zu unterschätzen ist der Aufwand, um die Teile mehrerer Baukästen zusammenzuhalten (vgl. Brinda 2004: 31). Ein Durchmischen der Teile mehrerer Baukästen ist jedenfalls unbedingt zu vermeiden. Speziell wenn die Teams sehr knapp beieinander sitzen, ist das nur schwierig erreichbar. Dazu kommt, dass die Verwaltung der Batterien für den Roboterbetrieb Vorbereitungs- und Nachbereitungszeit in Anspruch nimmt. Setzt man das Educational Set ein, so entspannt sich die Situation ein bisschen, da hier ein Akkupaket zum Einsatz kommt. All diese Aufwände fallen bei einem Sprachkurs oder bei der Nutzung einer Mikrowelt nicht an. Dort sind die erforderlichen Programme einmalig zu installieren, und das war es aufwandsmäßig meist schon für die nächsten Jahre.

Weiters ist fraglich, ob der gewünschte Wissenstransfer auf eine abstraktere Programmierumgebung klappt, das heißt ob die Schülerinnen und Schülern die unter NXT-G visualisierten Konzepte dann auch in einer textuellen Programmierumgebung wiedererkennen würden. Mit diesem Problem hat man allerdings sowieso immer zu kämpfen. Es ist aber auch fraglich, wie die Programmierung von Computern in Zukunft erfolgen wird. In der Industrie ist jedenfalls ein gewisser Trend weg von der rein textuellen Programmierung feststellbar. Programmierung ist heute ein vielgestaltiger Prozess, bei dem zum einen Teil designt, zum anderen Teil codiert wird.

Manche Lernenden verlieren nach einigen Tagen das Interesse an den Robotern (vgl. Granerud 2005: 2). Zwar macht das Zusammenbauen und Programmieren den meisten anfangs großen Spaß, aber mit dem Fortschreiten des Projekts sinkt die Motivation. Mit dieser Problematik hat man allerdings in Verbindung mit einem Sprachkurs und einer Mikrowelt noch viel [308] mehr zu kämpfen. Wichtig ist auf alle Fälle, dass sich die Lehrenden bei der Erstellung des Unterrichtskonzeptes Gedanken über den Spannungsbogen machen. Durch entsprechende Abwechslung hinsichtlich Methodik und Aufgabenstellung lässt sich jedenfalls diesem Problem gut vorbauen.

Nachdem das Programm nach jeder Änderung zum Testen auf den *NXT*-Stein hochzuladen ist, erhöht sich der Testaufwand gegenüber reinen Softwarelösungen (vgl. McNally 2006: 62). Dazu kommt, dass im Normalfall für den Programmupload das USB-Kabel einzustecken und nach dem Hochladen wieder abzuziehen ist (sofern nicht *Bluetooth* zur Übertragung genutzt wird). Des Weiteren schaltet sich der *NXT*-Stein nach einem gewissen Timeout automatisch aus, um Energie zu sparen. Nicht selten passiert es, dass man vergessen hat das USB-Kabel anzustecken oder dass man versucht das Programm auf den ausgeschalteten Baustein hochzuladen. Das sind keine wirklichen Probleme, aber diese Routinearbeiten werden umso lästiger, je länger der Kurs dauert. Bei den üblicherweise mit Sprachkursen eingesetzten Tools sowie bei den Mikrowelten haben die Schülerinnen und Schüler nicht mit solchen Schwierigkeiten zu kämpfen, da es sich um reine Softwaresysteme handelt.

Eine gewisse Kritik muss sich auch die *Lego Mindstorms*-Hardware gefallen lassen. Einerseits liefern die Sensoren zum Teil recht ungenaue Messergebnisse, andererseits schafft es *Lego* seit Jahren nicht, dem *NXT*-Stein einen ordentlichen Speicher zu gönnen. Anstatt sich auf die eigentliche Programmerstellung konzentrieren zu können, geht viel Zeit damit verloren, die richtigen Sensorgrenzwerte herauszufinden. Und in Zeiten, in denen üblicherweise in Gigabyte gerechnet wird, ist es unverständlich, dass der *NXT*-Baustein seinen Nutzenden weiterhin nur ein paar Kilobytes zu bieten hat. Gerade das Abspielen von Sounddateien während der Programmausführung sorgt für Unterhaltung bei den Kindern und Jugendlichen und beeindruckt bei der Präsentation. Nach nur wenigen hochgeladenen Sounds ist derzeit der Speicher aber leider schon voll.

Wer eine Technologiestütze wie *Lego Mindstorms* im Unterricht einsetzt, sollte diese selbst gut beherrschen. Idealerweise bieten die schulischen Fortbildungseinrichtun-

gen entsprechende Fortbildungskurse an. Ab Herbst 2012 wird es an der PH Niederösterreich hierzu einen Kurs geben. Ansonsten bleibt noch der Weg des Selbststudiums, der zwar ein wenig mühsamer, aber für engagierte Lehrerinnen und Lehrer sicherlich bestreitbar ist. Im Vergleich zum klassischen Sprachkurs beziehungsweise zur [309] Einarbeitung in eine Mikrowelt ist der Aufwand bei *Lego Mindstorms* sicherlich höher.

Last but not least ist festzuhalten, dass sich der Einsatz der *Lego Mindstorms*-Roboter im Rahmen des wöchentlichen zweistündigen Informatikunterrichts eher schwierig gestaltet. Hier geht einerseits zu viel Zeit mit dem Her- und Wegräumen verloren, andererseits ist der inhärente Zeitdruck dem eigenverantwortlichen und entdeckenden Lernen alles andere als zuträglich. Lehrende, die *Lego Mindstorms* im Unterricht einsetzen möchten, sollten also für passende Rahmenbedingungen sorgen. Idealerweise dauert eine zusammenhängende Lektion mindestens vier Unterrichtseinheiten lang. Der Ansatz unterscheidet sich diesbezüglich von den restlichen im Artikel angeführten Ansätzen, denn dort kommt man mit zwei Unterrichtseinheiten gut über die Runden.

Zusammenfassung und Fazit

Der Beitrag beschäftigte sich mit dem Einsatz von *Lego Mindstorms*-Robotern als Technologiestütze im Programmierunterricht. Nach einer kurzen Einführung in diese Produktreihe wurden zwei konkrete Ansätze für den Schulunterricht vorgestellt. Einer basierte auf *NXT-G*, also der standardmäßig mitgelieferten ikonischen Programmierumgebung, der andere setzte auf *leJOS NXJ*, das die Programmierung der Roboter unter *Java* ermöglicht. Es zeigte sich, dass sich durch den Robotereinsatz nicht nur ein attraktiver Unterricht gestalten lässt, sondern dieser bei Anwendung verschiedener konstruktivistischer Techniken und Methoden auch als didaktisch am Puls der Zeit bezeichnet werden kann. Durch das von den Schülerinnen und Schülern eingeforderte eigenverantwortliche Arbeiten und Lernen werden neben der Fachkompetenz auch die immer wichtiger werdenden Kompetenzen trainiert und verbessert: die Methodenkompetenz, die Sozialkompetenz und die persönliche Kompetenz. L*ego Mindstorms* ist kein Garant für modernen Unterricht, es ermöglicht aber einen solchen.

Danach wurden die Vorteile und Nachteile des Robotereinsatzes unter die Lupe genommen und festgestellt, dass der Einsatz solcher Baukästen stark motivierend auf die Kinder und Jugendlichen wirkt – einerseits da Roboter grundsätzlich faszinieren, andererseits da sie ihren Nutzerinnen und Nutzern direktes Feedback bezüglich der korrekten Funktionsweise des implementierten Programms zurückliefern. Als einer der ganz großen Pluspunkte wurde [310] die ausgezeichnete Skalierbarkeit dieses Mediums ausgemacht. So lässt es sich mit unterschiedlichen Zielgruppen nutzen. Auch die Heterogenität innerhalb einer Gruppe kann damit recht gut ausgeglichen werden, sofern es didaktisch richtig eingesetzt wird. Gender-Lernunterschiede lassen sich ebenfalls berücksichtigen und auch zur Begabtenförderung ist *Lego Mindstorms* einsetzbar. Durch die Nutzung solcher Roboter im Schulunterricht lernen die Schülerinnen und Schüler weiterhin eine andere Ausprägung des Computers kennen. Sie lernen

außerdem ein System kennen, bei dem ganz offensichtlich sowohl Hardware als auch Software ihren Beitrag zum Gelingen des Gesamtsystems beitragen. Auch die objektorientierten Konzepte lassen sich mittels *leJOS NXJ* ausgezeichnet vermitteln. Dem stehen die nicht zu vernachlässigenden Kosten zur Anschaffung der Baukästen sowie die Gebundenheit der Projekte an den Ort Schule gegenüber. Auch der Aufwand zur Instandhaltung der Baukästen und Roboter sollte nicht unterschätzt werden. Darüber hinaus ist das Testen eines Programms durch das Erfordernis des Programmuploads aufwendiger als bei reinen Softwaresystemen. Verschiedene Probleme in Verbindung mit der *Lego Mindstorms*-Hardware wurden ebenfalls bemängelt. Die Notwendigkeit zur Umorganisation des Informatikunterrichts bei Einsatz derartiger Roboter rundete das Kapitel ab. Sinnvollerweise dauert eine zusammenhängende Lektion nämlich mindestens vier Unterrichtseinheiten.

Können *Lego Mindstorms*-Roboter also für die Zukunft des Lernens einen Beitrag leisten? Definitiv, sofern sie didaktisch richtig eingesetzt werden. Die Schülerinnen und Schüler nehmen erfahrungsgemäß ihre neuen, coolen Klassenkameraden mit Freude in den Klassenverbund auf. Wünschenswert wäre allerdings, wenn derartige Unterrichtssequenzen nicht die Zukunft wären, sondern längst allgegenwärtig in den Schulen anzutreffen wären. Sollte das derzeit noch nicht der Fall sein: Die Zukunft beginnt heute!

Literatur

Altman, Aender (2005). Didaktische Konzepte von Lernsoftware – Konstruktionismus und LEGO MINDSTORMS. Diplomarbeit, TU Wien.

Bagnell, Brian (2007). Maximum Lego NXT. Building Robots with Java Brains. Variant Press.

Barnes, David J. (2002). Teaching Introductory Java through LEGO MINDSTORMS Models. In: ACM SIGCSE Bulletin, 34 (1), S. 147-151. [311]

Brinda, Torsten (2004). Didaktisches System für objektorientiertes Modellieren im Informatikunterricht der Sekundarstufe II. Dissertation, Universität Siegen.

Di Angelo, Monika et al. (2010). Begabtenförderung mit LEGO Mindstorms. In: Gerhard Brandhofer et al. (Hrsg.). 25 Jahre Schulinformatik, OCG, S. 275-279.

Di Angelo, Monika (2011). Robolab: LEGO Mindstorms im AHS-Informatikunterricht. Magisterarbeit, TU Wien.

Diethelm, Ira et al. (2003). Fujaba goes Mindstorms: Objektorientierte Modellierung zum Anfassen. In: Peter Hubwieser (Hrsg.). Informatische Fachkonzepte im Unterricht, INFOS 2003, 10. GI-Fachtagung Informatik und Schule, 17.-19. September 2003 in Garching bei München, S. 225-235.

Dietzel, Ruth/Rinkens, Tim (2001). Eine Einführung in die Objektorientierung mit Lego Mindstorms Robotern. Erfahrungsbericht aus dem Unterricht. In: Reinhard Keil-Slawik/Johannes Magenheim (Hrsg.). Informatikunterricht und Medienbildung, INFOS 2001, 9. GI-Fachtagung Informatik und Schule, 17.-20. September 2001 in Paderborn, S. 193-199.

Flowers, Thomas R./Gossett, Karl A. (2002). Teaching Problem Solving, Computing, and Information Technology with Robots. In: Journal of Computing Sciences in Colleges, 17 (6), S. 45-55.

Granerud, Roar (2005). Teaching object oriented programming to youths using the control technology Lego Mindstorms. Diplomarbeit, Universität Oslo.

Hirst, Anthony et al. (2003). What is the best programming environment/language for teaching robotics using Lego Mindstorms?. In: Artificial Life and Robotics, 7 (3), S. 124-131.

Klippert, Heinz (2004). Eigenverantwortliches Arbeiten und Lernen. Bausteine für den Fachunterricht. Beltz Verlag.

Kurebayashi, Shuji et al. (2006). Learning Computer Programming with Autonomous Robots. In: Roland Mittermeir (Hrsg.). Informatics Education – The Bridge between Using and Understanding Computers, International Conference in Informatics in Secondary Schools – Evolution and Perspectives, ISSEP 2006, Vilnius, Lithuania, November 7-11, 2006, Proceedings, S. 138-149.

Lawhead, Pamela B. et al. (2002). A Road Map for Teaching Introductory Programming Using LEGO Mindstorms Robots. In: ACM SIGCSE Bulletin, 35 (2), S. 191-201.

Löwenstein, Bernhard (2010). Objektorientierung und LEGO Mindstorms. Vermittlung objektorientierter Konzepte mittels LEGO Mindstorms im Schulunterricht. VDM Verlag Dr. Müller.

Löwenstein, Bernhard (2011). „Was nutzt ein klassischer Sprachkurs, wenn die halbe Klasse auf der Strecke bleibt". Dietrich Boles im Interview. In: Java Magazin, 2011 (5), S. 34-35.

Löwenstein, Bernhard/Di Angelo, Monika (2011). Mikrowelten. Schüler mittels alternativer Ansätze für die objektorientierte Programmierung mit Java begeistern. In: Java Magazin, 2011 (5), S. 30-33.

McNally, Myles et al. (2006). Do Lego MindStorms Robots have a Future in CS Education?. In: ACM SIGCSE Bulletin, 38 (1), S. 61-62.

Schmidt, Peter (2009). Einsatz von Lego Mindstorms im Unterricht der HTL für Informationstechnologie. Diplomarbeit, TU Wien.

Stolzlechner, Hans (2007). Forschendes Lernen – Ein alternativer Weg zum Erlernen einer Programmiersprache. Arbeitspapier, PTS Tamsweg.