oaBricksle
Der knuffige LEGO-Roboter

Olivia Ley, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Ziel des LEGO-Mindstorms-Projektseminars war es, einen LEGO-Roboter aus dem NXT-Bausatz zu bauen. Dazu den programmierbare LEGO-'Stein', im Original 'Brick', mit MATLAB zu programmieren. Ein Zweck des Roboters war nicht vorgegeben und so entstand oaBricksle, der knuffige aber sinnlose (und auf Dauer etwas nervige) LEGO-Roboter.**

Knuffig und niedlich sollte er sein, da es in der Welt einfach zu wenig Knuffigkeit gibt. Nichts produktives zu können ist ein weiteres, häufiges Merkmal knuffiger Dinge wie z.B. junger Katzen und daher ein wichtiger Aspekt für die letztendliche Knuffigkeit.

Von da aus war es auch nur noch ein kleiner Schritt bis zur Nervigkeit und nun existiert das oaBricksle, der knuffige LEGO-Roboter, welcher fröhlich piepend durch den Raum fährt, Aufmerksamkeit verlangt und allein gelassen bald schlechte Laune bekommt und dann anfängt 'rumzuheulen'.

Natürlich freut er sich auch, wenn man ihn streichelt, er schnurrt dann sogar.

***Schlagwörter*—LEGO-Mindstorms, Roboter, niedlich, sinnlos, NXT**

I. Einleitung

DAS oaBricksle findet vor allem in privaten Haushalten Anwendung, aber auch in sozialen Umfeldern wie Kindergärten oder Altenheimen könnte er durchaus eine Bereicherung des Umfelds darstellen.

So soll er den Alltag seiner Besitzer erfreulicher und angenehmer machen, eine Bezugsperson bis zu einem gewissen Grad positiv emotional beeinflussen können und mehr Knuffigkeit in die Welt tragen. Es gibt bis zu einem Punkt bereits künstliche Intelligenz, die uns das Leben vereinfacht und angenehmer gestaltet, doch häufig werden auch Zweifel laut, ob wir nicht zu bequem werden, wenn wir so vieles den neuen Technologien überlassen.

Das oaBricksle schafft diesen Vorwurf durch seinen beständigen Hunger nach Aufmerksamkeit beiseite und verbessert mit seiner Knuffigkeit auch das Leben seines Besitzers. So ermöglicht er eine positive emotionale Bindung nicht nur für Personen, die sich wegen Allergien, der Hausordnung oder anderen Gründen kein Haustier anschaffen können, sondern auch für die, die einfach keine Tiere, dafür aber Knuffigkeit mögen.

An Knuffigkeit und schönen Dingen fehlt es dieser Welt und den Nachrichten an erfreulichen Tagesthemen. Die Inspiration zum oaBricksle fand sich im in Abbildung 1 dargestellten Webcomic.

Abbildung 1. Die Idee, mit dem Roomba als Vorbild [1]

Der dort gezeigte Roboter ist an den Saugroboter Roomba angelehnt. So nach Aufmerksamkeit zu suchen, wurde zu einem Hauptzweck des oaBricksle.

Einem Wesen so einfach und schnell eine Freude bereiten zu können, löst positive Gefühle aus, welche wiederum wichtig für die Gesundheit sind. Denn wer sich gut fühlt, lebt damit auch ein kleines bisschen gesünder.

DOI: 10.24352/UB.OVGU-2020-039

II. Vorbetrachtungen

In den Vorbetrachtungen werden die bereits vorhandenen Lösungen kurz vorgestellt und mit einer Literaturquelle belegt. Ebenso werden für die eigene Lösung genutzte Verfahren erklärt.

A. LEGO-Mindstorms

LEGO-Mindstorms ist eine programmierbare Reihe der LEGO Serie, die die Bausteine des LEGO-Technik Bausatzes verwendet.

Außerdem sind ein programmierbarer LEGO-Stein, 'Brick' im Englischen, Sensoren und Motoren sowie Kabel Teil des Systems.

Über eine eigene Umgebung von LEGO kann der Brick dann mittels Codebausteinen programmiert werden. Hier wurde die zweite Generation, der NXT genutzt.

Durch die Einbindung einer von der RTWH Aachen zur freien Verfügung gestellten Bibliothek kann der Brick auch über die Software MATLAB angesprochen und gesteuert werden.

Mittels entsprechender Befehle können damit die Sensoren abgefragt und Motoren angesprochen werden, wie es bei der originalen Software möglich ist.

B. Roombas und die Amöbenstrategie

Roomba ist der geschützte Name der Saugroboter der Firma iRobot [2].

Da vor allem das Prinzip der Funktion von Saugrobotern, beziehungsweise deren Fortbewegung durch die Wohnung, und weniger die Marke als Grundlage diente, ist dies also nur ein bekannteres Beispiel mit hohem Erkennungswert.

Im nachfolgenden Text wird der Name daher etwas weitgreifender benutzt.

Um einen Saugroboter durch die Wohnung zu leiten werden unterschiedliche und unterschiedlich effiziente Strategien genutzt [3]. Häufig werden mehrere Strategien kombiniert, um auch schwierigere Situationen simpel lösen zu können und Festfahren zu vermeiden.

So gibt es Strategien, die bestimmte vorgegebene Wege abarbeiten lassen, zum Beispiel die Mäander-Strategie [4].

Bei der Wandverfolgung wird wiederum eng mit der Sensorik zusammen gearbeitet, um den richtigen Weg zu finden [5].

Um solche Wege zu vereinfachen und den Ablauf effizienter zu gestalten, werden häufig gleich Karten der Räume und Hindernisse erstellt, auf die bei der nächsten Fahrt wieder zugegriffen werden kann.

Allerdings braucht ein knuffiger Roboter keine Effizienz.

Deshalb wurde hierfür auf das Prinzip der Zufallsfahrt zurückgegriffen. Die Zufallsfahrt wird in zwei Varianten unterschieden: der zeitabhängigen Random-Strategie [6] und der Amöben-Strategie [7].

Bei beiden wird beim Eintreten festgelegter Ereignisse ein zufälliger Richtungswechsel eingeleitet.

Beide Varianten nutzen einen Frontschild zur Kollisionserkennung, der die vordere Hälfte des Roboters abdeckt. Bei der Random-Strategie wird das Ereignis außerdem beim Erreichen einer zuvor festgelegten, maximalen Fahrtzeit ausgelöst.

Die Amöben-Strategie erhielt ihren Namen über das Muster der Spur die bei längeren Laufzeiten entsteht, welche bei der Draufsicht dem Aussehen einer Amöbe ähnelt.

III. Aufbau und Funktionen

Das Konzept der etwas nervigen, aber knuffigen Verhaltensweise wurde stufenweise aufgebaut und implementiert. Damit stehen auch, im Rahmen der technischen Umsetzbarkeit, beliebige Erweiterungen offen.

A. Fahren

Die grundlegende Eigenschaft des Konzepts war die eigene Fortbewegung im Raum, die etwa der eines simplen Staubsaugeroboters ähneln sollte.

Er sollte sich im Raum bewegen können und bei Zusammenstoß mit einem Hindernis, auch 'andotzen', die Richtung ändern.

Zum Vorwärtsfahren mussten beide Ketten, auf denen sich oaBricksle bewegen sollte, mit der gleichen, positiven Motorleistung betrieben werden. Eine negative Motorleistung lässt die Elektromotoren lediglich mit entsprechend angegebener Leistung rückwärts drehen.

Drehen wurde durch gegenläufige Ketten realisiert. Im Programmcode werden dafür die Ketten einmal mit der positiven und einmal mit der negativen Motorleistung angetrieben. Rückwärtsfahren war ohne Wenden zu müssen mit entsprechend negativer Motorleistung möglich.

Um Hängenbleiben an Hindernissen zu vermeiden, fährt er zunächst zurück, hält an und wendet dann. Das Halten zwischendurch ist erforderlich, um nicht zu viele Anweisungen zeitgleich an die Motoren zu senden und sie damit zu überfordern.

Andotzen ist die Kollision des Bumpers, der Frontschild des oaBricksle, mit einem Hindernis, wie einer Wand, und das damit verbundene Auslösen des Tasters.

Statt des Tasters welcher durch den damit verbundenen Bumper die gesamte Vorderseite abdeckt wäre auch ein Ultraschallsensor möglich gewesen. Die Möglichkeit wurde nicht weiter verfolgt, da die Ultraschallsensoren nur einen stark beschränkten Winkel und eine geringe Höhe erfassen und zudem nicht exakt genug arbeiten.

Anstatt das Umfahren von Objekten und Wegkommen von Wänden dann noch zu optimieren, wurde der Winkel, um den sich der Roboter neu ausrichtet, um einen Zufallsfaktor erweitert. Dieses etwas verschusselt und hilflos anmutende Verhalten ist häufig bei knuffigen Wesen anzutreffen.

B. Piepen / Akustische Rückmeldung

Es gibt wenig Dinge, die als niedlicher empfunden werden als ein maunzendes Kätzchen, ein freudig bellender Welpe, oder ein glücklich piepender Roboter - wie in diesem Fall.

Außerdem ist es eine hervorragende Möglichkeit, den Roboter zurückgeben zu lassen, was er gerade tut, ohne auf die Ausgabe in der Befehlszeile achten zu müssen.

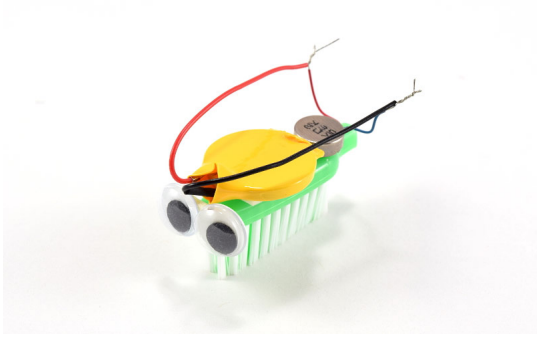

Abbildung 2. Ein Bristlebot oder Zahnbürstenroboter [8]

Vom Konzept her sollte er also ab und zu ein fröhliches Pfeifen von sich geben. Über die Angabe von Frequenz und Länge sowie Pausen dazwischen wurden einige eingängige, längere und kürzere Melodien umgesetzt und getestet.

Dabei schnitten kürzere, eigens geschriebene Stücke deutlich besser ab, da sie keine bereits bestehenden, negativen Assoziationen wach rufen.

Neben dem Grundzustand des normalen Fahrens, der von einem fröhlichen Dreiton begleitet wurde, gab es auch ein tieferes Brummen, das etwa einem Brummeln oder Schnurren (dazu später noch mehr) entsprach.

Um dem Ganzen einen nicht ganz erzwungenen Charakter zu verleihen, hätte nun wieder ein Algorhythmus geschrieben werden können.

Da aber einerseits Auslöser für solche Reaktionen nicht ganz klar sind und andererseits schwieriger sowie zeitaufwändiger umzusetzen wären, wurde auch hier wieder eine Zufallsfunktion dafür implementiert.

C. Schnurren / pwrr-o-Mat

Viele Bestandteile von oaBricksle, für die der pwrr-o-Mat ein exzellentes Beispiel darstellt, haben etwas verwirrende Namen. Diese sind ebenso wichtiger Bestandteil des Konzeptes.

So lässt sich der programmierbare NXT-'Brick' auch über Bluetooth mit einem PC verbinden und kann dann angesteuert werden.

In Heimarbeit beim Basteln stellt das noch kein Problem dar. Allerdings wird man bei einer größer konzipierten Veranstaltung wie dieser schnell erkennen, dass einheitlich benannte Geräte fatal hinsichtlich des Wiedererkennungswertes für einzelne Geräte sind.

Kurz: Welcher NXT, bei dem Bluetooth aktiviert war, war nun der Richtige?

Über die hauseigene Software von LEGO ließ sich der Brick dann am Computer über Kabel umbennen.

Im Sinne der knuffigen Namensgebung sollte der Brick dann den Namen Bricksley tragen, was sich jedoch als ein Zeichen zu viel erwies. Für den Namen waren nur acht Zeichen vorgesehen.

Und da der Name 'Bricksle' bereits etwas nach bayrischem Dialekt klang, wurde als Variable für die Verbindung dann 'oaBricksle' genommen. Abgesehen vom Nutzen als Namen des Roboters hat die Bezeichnung also tatsächlich auch einen Nutzen im Code.

Ursprünglich war geplant, das Schnurren, also eine Vibration über das Grundprinzip der Funktionsweise eines einfachen Roboters auf Borsten zu nutzen.

Diese Roboter werden in der simpelsten Ausführung auch Bristlebot oder Zahnbürstenroboter genannt.

Sie bestehen dabei lediglich aus dem abgeschnittenen Kopf einer Zahnbürste, einer Knopfzelle und einem Vibrationsmotor. Ein Beispiel ist in Abbildung 2 zu sehen, es wurden außerdem Verzierungen angebracht.

Dabei kann die Vibration auch durch einen beliebigen anderen Motor erzeugt werden. Dafür müssen lediglich die Umdrehungen pro Minute der Rotationsachse hoch genug sein und mit einen asymmetrisch daran platzierten Schwungstück eine Unwucht erzeugt werden, die dann die Vibration hervorruft.

Dieser häufig unerwünschte Effekt zerlegt allerdings über einen längeren Zeitraum ziemlich zuverlässig starre Systeme.

Ein nicht zum LEGO Bauset gehörender Motor war samt Batterien und Schwungstück vorhanden.

Das Einschalten und Einbauen erwies sich allerdings als nur schwer umsetzbar und es standen noch viele Zahnräder und ein dritter Motor zur Verfügung.

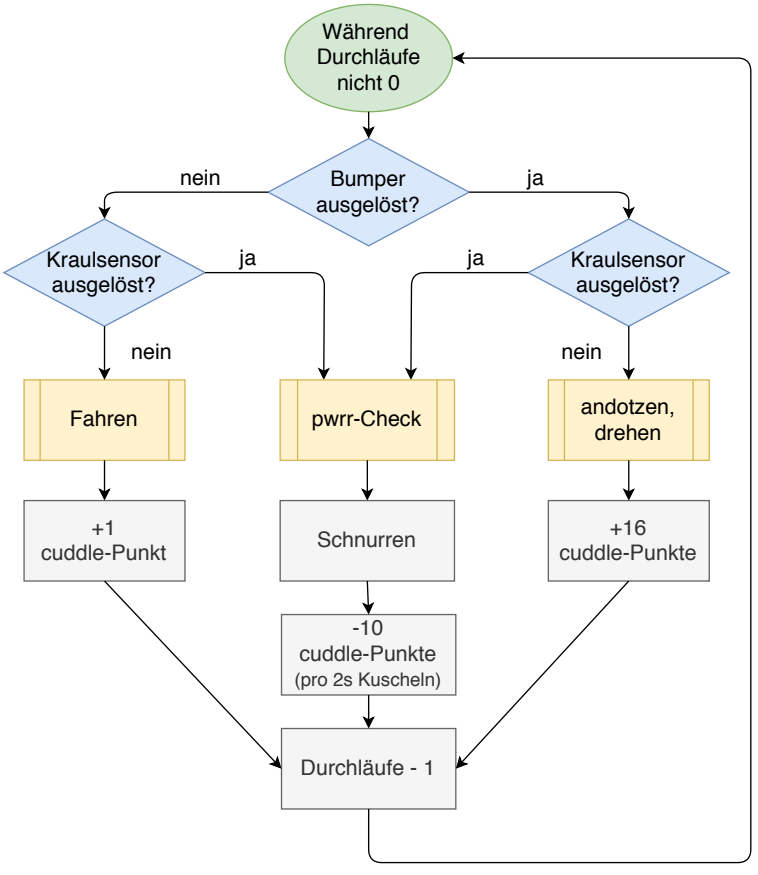

Abbildung 3. Programmablaufplan zur Unterscheidung von Fahren, Kollision und Kraulen

Summa summarum dreht beim Schnurren jetzt ein etwas wackliger Propeller schnell (jedoch ohne viel Kraft dahinter) und bringt den oaBricksle fühl- und hörbar zum Schnurren.

D. Cuddle-Meter / die Persönlichkeit

Zum Konzept des knuffigen Roboters gehört auch, dass er wütend wird, beziehungsweise anfängt 'rumzuheulen'. Dies wird ausgelöst durch Vernachlässigung oder wenn ihm schlimme Dinge passieren, wie zum Beispiel die Kollision mit einer Wand. Diese Grundaggressivität ließ sich am Besten über eine Geschwindigkeitserhöhung realisieren.

Allerdings brauchte es dazu ein Gedächtnis, hier das Cuddle-Meter, das den Kuschelwert angibt und damit ein Maß für die Unzufriedenheit ist. Das Cuddle-Meter ist jeweils die Gesamtsumme der cuddle-Punkte am Ende eines Durchlaufs.

Die Anzahl der Durchläufe wird beim Start des Programms festgelegt.

Nach jedem erfolgten Durchlauf wird die Anzahl der verbleibenden Durchläufe um eins verkleinert, bis das Programm schließlich bei null Durchläufen endet. Damit wird die gesamte Laufzeit begrenzt. Zu Testzwecken wurden üblicherweise 300 Durchläufe genutzt.

Für jeden Durchlauf wird dabei zunächst entschieden, ob oaBricksle gerade gegen ein Hindernis gefahren ist oder nicht. Nach jedem per Kollision erreichten Hindernis wendet oaBricksle, piept enttäuscht und das Cuddle-Meter wird für diese Runde um 16 Punkte erhöht.

Wenn kein Hindernis gefunden wurde, fährt er eine Runde normal weiter und das cuddle-Meter wird um einen Punkt hochgezählt. Die Beträge, um die das cuddle-Meter verändert wird, sind willkürlich gewählt, sie stellten sich nach mehreren Testläufen als geeignet heraus.

Um eine höhere Priorität zu erreichen wird der Kraulsensor, wie in Abbildung 3 zu sehen ist, erst nach dem Bumper abgefragt.

Statt beide Sensoren nacheinander abzufragen sind so die Reaktionszeiten bei gleichzeitig weniger Durchläufen kürzer. Der Code benötigt dann weniger Ressourcen.

Wird der Kraulsensor nicht ausgelöst, werden die normalen Aktionen weiter ausgeführt. Andernfalls wird die Funktion pwrr-Check aufgerufen. Diese aktiviert den pwrr-o-Mat, den Schnurrmotor von oaBricksle und sorgt für gelegentliches zufriedenes Brummeln.

Außerdem sorgt diese Funktion dafür, dass cuddle-Punkte für das Kraulen vom cuddle-Meter abgezogen werden. Für je zwei Sekunden Kraulen sinkt es um zehn Punkte. oaBricksle wird wieder zufriedener.

E. Aufregen / furity

Anhand des cuddle-Meters lässt sich erkennen, wie aufgeregt oaBricksle zu welchem Zeitpunkt ist. Um das auch äußerlich umzusetzen, berechnet die Funktion furity die momentane Geschwindigkeit anhand des cuddle-Meters.

Dafür wird ausgewertet, in welchem Bereich die cuddle-Punkte sich gerade befinden und die Grundgeschwindigkeit wird entsprechend angepasst. Damit wird oaBricksle dann nach mehrmaliger Kollision mit Hindernissen schneller werden, durchquert damit wiederum schneller den Raum und gelangt dadurch schneller zum nächsten Hindernis.

Durch die nächste Kollision steigt das cuddle-Meter wieder, oaBricksle wird noch schneller und aufgeregter und in überschaubarer Zeit ist er bei der maximalen Geschwindigkeit angekommen. Ein Kreislauf, von dem er nur zeitweise durch Kraulen abgebracht werden kann.

IV. Zusammenfassung und Fazit

Aus einer Idee, viel LEGO, Code und Tee entstand in den zwei Wochen des Seminars die erste voll funktionsfähige Version des knuffigen Roboters oaBricksle. Er kann sich eigenständig durch einen Raum bewegen, Hindernisse umfahren und einfache Interaktionen mit Menschen durchführen sowie seine momentane Laune ausdrücken.

Verbesserung ist hinsichtlich der knuffigen Optik ebenso notwendig, wie bessere und ausgefeiltere Sensorik. Dadurch lassen sich dann noch mehr Interaktionsmöglichkeiten einbinden.

Die im LEGO-Bausatz enthaltene Sensorik waren doch sehr begrenzt und simpel gehalten.

Beispielsweise könnte die Stimmung besser über ein Display angezeigt, eine Sprach- und Bilderkennung für Unterhaltungen und ein Belohnungssystem eingefügt und mehr Speicherplatz für komplexere Sprachausgaben eingefügt werden.

Das noch recht simple Verhalten lässt Raum für weitere Feinheiten und komplexere, ja sogar unterschiedliche Persönlichkeiten.

Vielleicht hängt ja der eine oaBricksle der nächsten Generation gerne in staubigen Ecken ab, während ein anderer gerne Kissen erklimmt und ein weiterer sich vielleicht hungrig über die auf dem Fußboden liegenden Krümel der letzten Mahlzeit her macht?

Der Kreativität sind ja bekanntlich keine Grenzen gesetzt, und was sich vielleicht doch alles umsetzen lässt, wird sich zeigen.

Begeisterte Nachfragen gab es bereits einige.

Anhang

Literaturverzeichnis

[1] STARFLEETRAMBO ON TUMBLR: *Webcomic: ok but imagine a roomba...* https://starfleetrambo.tumblr.com/post/171767831093/starfleetrambo-starfleetrambo-ok-but-imagine-a. Version: März 2018

[2] IROBOT GERMANY GMBH: *Roomba Saugroboter*. https://www.irobot.de/roomba. Version: März 2020

[3] U.FRITZ: *Saugroboter auf Zufallsfahrt*. http://saugrobot.de/bewegung-zufallsfahrt.php. Version: März 2005-2012

[4] U.FRITZ: *Mäander-Fahrstrategie*. http://saugrobot.de/saugroboter-maeander.php. Version: April 2005-2012

[5] U.FRITZ: *Wandverfolgung*. http://saugrobot.de/an-der-wand-entlang.php. Version: April 2005-2012

[6] U.FRITZ: *Zufalls-Fahrstrategie*. http://saugrobot.de/random-fahrstrategie.php. Version: März 2005-2012

[7] U.FRITZ: *Amöben-Fahrstrategie*. http://saugrobot.de/amoeben-fahrstrategie.php. Version: März 2005-2012

[8] SCIENCE BUDDIES: *bristlebot-googly-eyes*. https://cdn.sciencebuddies.org/Files/7937/15/bristlebot-googly-eyes.jpg. Version: März 2020