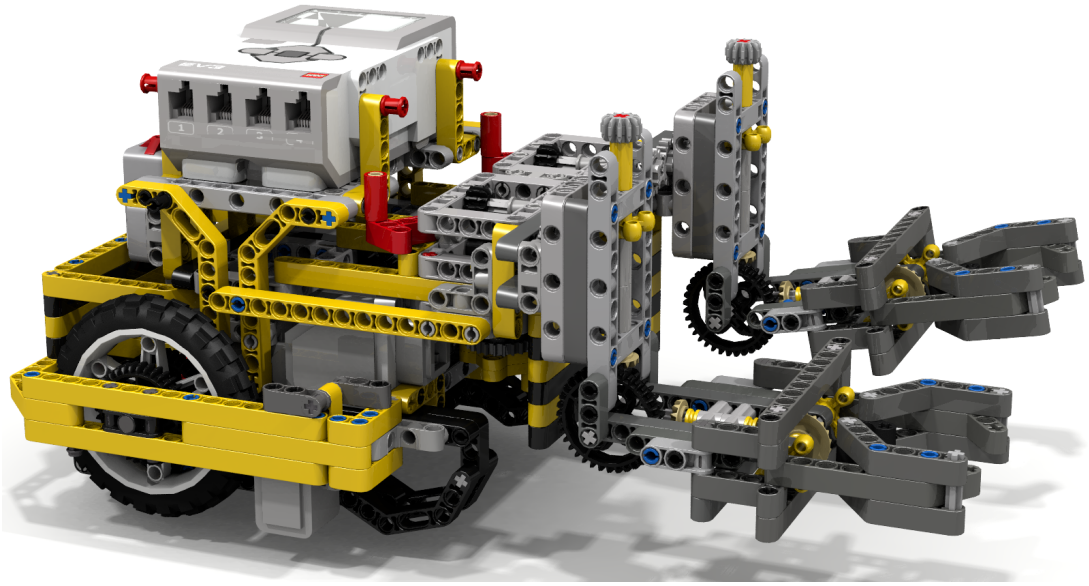
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG

EIT FAKULTÄT FÜR ELEKTROTECHNIK UND INFORMATIONSTECHNIK

# LEGO-Praktikum. entwickeln + programmieren + optimieren

## Berichte der Studierenden zum Projektseminar Elektrotechnik/Informationstechnik





Eine Schriftenreihe der Otto-von-Guericke-Universität Magdeburg, Fakultät für Elektrotechnik- und Informationstechnik, Institut für Medizintechnik

Herausgeben von: Mathias Magdowski, Marcus Schmidt und Enrico Pannicke

Band 1 vom Wintersemester 2017/2018

# Inhaltsverzeichnis

## Impressum

Herausgeber: Mathias Magdowski, Marcus Schmidt und Enrico Pannicke, Institut für Medizintechnik, Fakultät für Elektro- und Informationstechnik, Otto-von-Guericke-Universität Magdeburg, Postfach 4120, 39016 Magdeburg

DOI: 10.24352/UB.OVGU-2018-038

Redaktionsschluss: April 2018

Seminarzeitraum: 12.–23. Februar 2018

Bezug: Open Access, Digitale Hochschulbibliothek Sachsen-Anhalt
`http://edoc2.bibliothek.uni-halle.de/`



1. Auflage, Magdeburg, Otto-von-Guericke-Universität, 2018

Erstellung des Sammelbandes mittels LaTeX, `hyperref` und `pdfpages`

# Bau eines Bergungsroboters „Franky“

Nodom Notouom Danielle Ingrid, Elektrotechnik und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

*Abstract*— **Für Retter ist es nicht einfach, sowohl Leute in menschenfeindliche Umgebungen zu helfen als auch Gefahren zu beseitigen. Dafür brauchen Sie Zeit und Mut. Wir haben uns das Thema angenommen und haben uns entschieden einen Roboter aufzubauen, der das schnell und problemlos erledigen kann. Der Einsatz von Robotersystemen in menschenfeindliche Umgebungen erfordert ein höchstes Maß an Autonomie dieser Systeme, das heißt, die Arbeitsaufgaben sollen möglichst selbständig und ohne Eingriff des Menschen gelöst werden.**

***Schlagwörter*—Autonomie, Gefahren, menschenfeindlich, Retter, Roboter.**

## I. Einleitung

Katastrophengebiete, wie nach dem Tōhoku-Erdbeben 2011 in Japan, erfordern Helfer, die ihr eigenes Leben aufs Spiel setzen, um Vermisste zu suchen und Verschüttete zu bergen. Zwar kann ein Roboter diese Fähigkeiten besitzen und schneller bei der Rettung sein. Der Bergungsroboter sollte Überlebende in Menschenfeindliche Umgebungen retten, dazu gehört auch die Rettung ohne Gefahren, andere Funktionen sind: die Beseitigung von Gefahren und frei Räumung des Rettungswegs, damit es für die Retter einfacher wird Leute zu helfen.
Rote Objekte sind als Gefahren betrachtet und müssen vom Roboter aus dem Weg weggeräumt und grüne Objekte muss er in Sicherheit bringen.

## II. Vorbetrachtungen

Allerdings sind wir nicht die einzige, die gefunden haben, dass es nötig wird einen Roboter aufzubauen, der in Katastrophengebiete eingesetzt werden kann. Auf die Idee sind auch andere Ingenieure gekommen.

### *A. Arduino Hexapod-Zelos*

Zelos[1] ist ein sechsbeiniger Bergungsroboter, der vom Schüler des Gymnasiums Oberalster, Janning Meinert (18) im Jahr 2015 aufgebaut wurde. Damit Roboter hat er großen Erfolg gehabt: Als erster Schüler des Alstertaler Gymnasiums hat er bei dem Wettbewerb „Jugend forscht“ nicht nur den Regional-, sondern auch den Landeswettbewerb gewonnen. Zelos sollte ein eifriger Begleiter von Bergungsteams sei, die in Katastrophengebieten Menschenleben retten. Zwar hat er den Roboter drei Mal entwickelt:

“Zelos1“ der hauptsächlich aus Aluminium besteht. Er verfügt über drei Motoren je Bein, so dass er nicht nur vorwärts und rückwärts, sondern aus dem Stand auch seitwärts gehen kann.

DOI: 10.24352/UB.OVGU-2018-037 

“Zelos2“ der noch eindrucksvoller aussieht. Die Beine sind in einem 3D-Drucker aus Kunststoff hergestellt worden. Dafür konnte Janning die Firma „MakerBot“ gewinnen

“Zelos3“ mit austauschbare Füße, um sich an den Untergrund anzupassen – Gumminoppen für steinigen Untergrund, tellerartige Skier für Schnee.

### *B. Packbot 510*

Der Packbot 510[2] ist ein Roboter des US-Unternehmens iRobot, der in Katastrophengebieten eingesetzt werden kann. Er kann mit verschiedenen Sensoren und Kameras ausgestattet werden. Als Bergungsroboter wurde er für das erste Mal in Japan nach Erdbeben getestet.

### *C. Der Roboterkäfer*

Willi Zschiebsch entwickelte den Prototyp eines Roboters, der die Vorteile eines Wurmroboters und die einer Laufmaschine in sich vereint. Er besitzt einen Bewegungsmechanismus, der sich am biologischen Vorbild des Hundertfüßers [3] orientiert. Der Nachwuchsingenieur konstruierte einen elektronischen Helfer, der Hindernisse wie steile Wandabschnitte, kleine Schluchten und enge Felsspalten durchqueren kann, um beispielsweise zu Opfern solcher Naturkatastrophen zu gelangen. Er kann auch kleine Gegenstände transportieren und ist mit einer Kamera ausgestattet.

## III. Hauptteil

### *A. Der Aufbau*

Für den Aufbau unserem Roboter haben wir folgende Bauelemente benutzt: einen Farbensensor, einen Ultraschalsensor, drei Motoren und einen NXT benutzt. Die anderen Elemente, wie die Kette für die Stabilität, die Klauen zum Greifen hatten wir aus Lego.

Mithilfe der Farbensensor sollte der Roboter überprüfen, ob die Farbe des Objektes grün, oder rot ist. Mit dem Ultraschallsensor sollte er die Entfernung zu einem Objekt messen, bevor er anfängt zu fahren und die Motoren, helfen ihm sich zu bewegen.

### *B. Positionsbestimmung.*

Der Roboter haben wir so programmiert, dass er die Entfernung zum Objekt misst und zu ihm fährt, ohne das Objekt zu stoßen. Zuerst haben wir auf dem Tisch versucht und dann auf den Boden, mit derselben Drehzahl des Motors. War der Versuch auf dem Tisch erfolgreich, dann war es auf dem Boden leider nicht und der Grund dafür ist die Reibung. wir haben uns entschieden der Roboter auf dem Boden fahren

zu lassen, weil es da mehr Platz gab und Um das Problem zu lösen haben wir uns die folgende Formel gedacht:

$$Dist=(entf-4) * 35$$

Wobei Dist: die zufahrende Entfernung ist
und Entf: die gemessene Entfernung
Zur gemessenen Entfernung haben wir eine Zahl abgezogen, je nach Bodenbelag, in diesem Fall ist es 4, und das ganze Mal 35(Die Grad Zahl der Umdrehungen, 35=1cm) gerechnet

## IV. Ergebnisdiskussion

Der Roboter konnte genau tun, was er tun sollte d.h. zum Objekt fahren, die Farbe überprüfen, ist die Farbe Rot, das Objekt beseitigen und wenn das grün ist, es in Sicherheit bringen. Außerdem rechnen noch damit, das Problem der ungenauen Sensoren und Motoren. Die Motoren sind auch leider, je nach der Größe des Objekts zu schwach zum Greifen.

## V. Zusammenfassung und Fazit

Zusammenfassend, haben wir geschafft unseren Roboter genauso zu bauen und zu programmieren, wie wir uns gedacht hatten.

Würde man ihn in der Realität aufbauen, dann wäre es besser, dass man genauere Sensoren und Motoren nutzt. Die Motoren sollten auch zum hochheben stärker sein. Zusätzlich ist die Geschichte nur ein Objekt tragen zu können auch Problem.

Natürlich würde das in der Realität mit Farben Problematik sein, weil es passieren kann, dass die zuretende Person etwas Rotes angezogen hat und in diesem Fall wird der Roboter sie als Gefahr betrachten und sie nicht retten.

## Anhang

*Abbildung1: Ansicht des Roboters von Vorne*

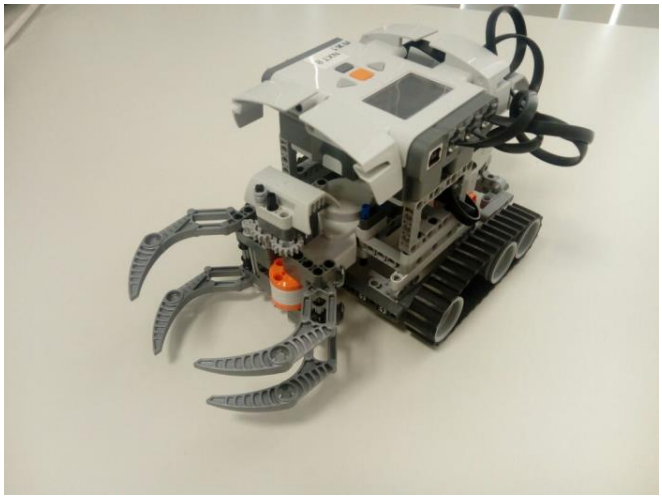


*Abbildung2: Ansicht des Roboters von Hinten*

## Literaturverzeichnis

*[1]* *http://www.heimatecho.de/artikel/4815*

*[2]* http://www.robonews.de/2011/03/bergungsroboter-japan-erdbeben-einsatz/

*[3]* *http://oiger.de/2015/03/28/junger-sachse-baut-rettungs-roboterkaefer/73209*

# *SUCH- UND RETTUNGSROBOTER*

***Bericht zum Projekt des Lego Mindstorms Seminars***

***Mustafa Mustafa , Elektrotechnik und Informationstechnik , 220409***

## *Kurzfassung :*

Such- und Rettungsroboter werden für eine Vielzahl von Aufgaben im 21. Jahrhundert eingesetzt. Sie werden entworfen, um bestimmte Aufgaben zu tun, die Menschen könnten, aber nicht tun, oder sogar Dinge, die Menschen nicht tun können. Unser Ziel war es, einen Roboter zu konstruieren , der in einer Notfalleinsatzmission eingesetzt werden kann, und sich in Gelände und Umgebungen wagen kann , die für menschliche Einsatzkräfte zu gefährlich sind. Die Hauptfunktion des Rettungsroboters besteht darin, ein Zielobjekt zu suchen, mit Farbsensor und Ultrasonic-sensor zu erkennen , es mittels seiner Klauen zu greifen ,und es zum Startpunkt (Sicherheitszone) zurückzubringen .

## *1 _ Einleitung :*

Die Verwendungen von Robotern sind nicht begrenzt . Von einfachen Aufgaben im Haushalt wie Staub saugen bis zu komplexen Aufgaben in großen Industrieanlagen . Das Ziel ist oft, menschliche Arbeitskräfte zu entlasten beziehungsweise vor Gefahren zu schützen. Nach einem Erdbeben in einem Wohngebiet oder einem Unfall in einem Chemielabor müssen die Rettungskräfte möglicherweise an diese Orte gehen, um Überlebende zu finden. Mit einem Roboter , der in ähnlichen Szenarien eingesetzt wird , können nicht nur Überlebende gefunden, sondern auch das unnötige Risiko von anderen Menschen vermieden werden. Zusammen mit D.Notouom und K.Harders haben diese Projekt im Zeitraum vom 12.Februar bis 23.Februar bearbeitet . wir wollten damit solche Szenario modellhaft darstellen . Das Ziel war einen Roboter zu konstruieren , der nach bestimmte Gegenstände sucht , ortsfeste Hindernisse erkennt und diesen ausweicht . Die Gegenstände sind farbig und sie sollen gefunden werden . Falls ein Gegestand rot ist , wird er beseitigt und falls er grün ist , bringt der Roboter ihn zum Startpunkt zurück, ansonsten (falls andere Farbe) sind die Gegenstände ortsfeste Hindernisse und werden ausgewichen. Die grüne Gegenstände können als Personen die zu bregen sind interpretiert werden . Der Roboter wird mit LEGO unter verwendung des NXT-Bausteins aufgebaut und in MATLAP programmiert .

## *2 _ Vorbetractungen :*

Es existieren bereits viele unterschiedliche Rettungsroboter mit verschiedenen Funktionsweisen. Eine davon besteht darin, Überlebende mit Echtzeit-Videoübertragung zu erkennen und den Standort des Roboters über einen Bluetooth-GPS-Empfänger zu senden. Die Videokamera wird auch drahtlos mit dem PC verbunden, um Videos in Echtzeit zu übertragen. Das GPS sendet auch Standortdaten an den PC und es wird der Ort erhalten, an den die Retter gesendet werden müssen. Der Roboter wird vom Microcontroller gesteuert und wird über Echtzeit-Video-Feedback am PC überwacht. Der Nachteil hier ist die fortgesetzte starke Abhängigkeit vom menschlichen

DOI: 10.24352/UB.OVGU-2018-039 

Operator. Darauf wollten wir bei unserem Projekt verzichten. Wir haben uns mehr auf Sensoren gestützt, damit der Roboter seine Aufgaben mit der geringstmöglichen Hilfe des Menschen erledigen kann.

## *3 _ Mechanische Umsetzung :*

Der Aufbau war ganz simple (Abbildung 1 ). Die Unterschicht besteht aus zwei Motoren, die nebeneinander vertikal angebracht wurden und da vorne werden zwei Sensoren fixiert, hinten ist ein dritter Motor mit die Klauen verbunden und auf der Oberschicht ist der NXT-Baustein festgelegt .

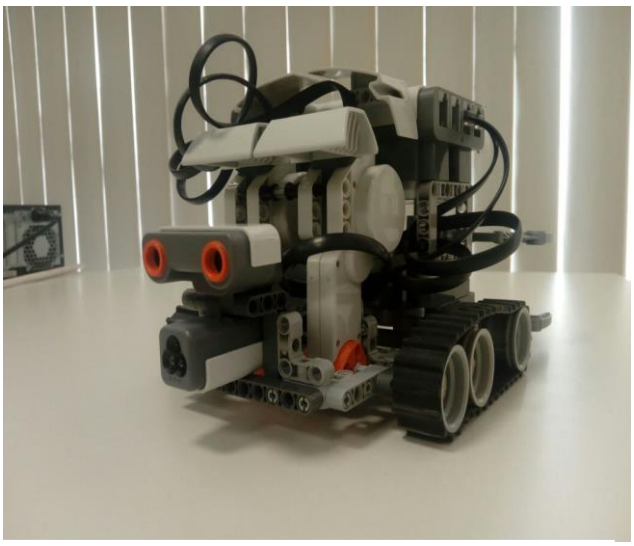


**Abbildung 1 : Das Fahrzeug (die Vorderseite)**

***3.1 _ Die Ketten :*** Um den Roboter besser bewegen zu können , haben wir anstelle von Rädern ein Paar Gummiketten verwendet. Diese Rettungsroboter bewegen sich oft in rauer Umgebung , so dass Ketten eine geeignetere Option als Räder sind, was eine bessere Fahrzeugstabilität über verschiedenem Gelände als bei Rädern bereitstellt . Der Nachteil ist hier die große Nahtfläche zwischen Kette und Boden , die den Reibwert erhöht und damit den Energieverbrauch auch erhöht. Diese Ketten sind mit einem Paar Motoren verbunden , die synchronisiert oder unabhängig arbeiten können , um die Fahrzeugbewegung in alle Richtungen zu ermöglichen.

***3.2 _ Die Klauen :*** (Abbildung 2 ). Die Klauen können kleine Objekte , die gefunden werden , greifen . Sie werden von einem einzigen Motor gesteuert , sie öffnen und schließen sich horizontal und sind in der Lage , verschiedene kleine Gegenstände zu halten , deren Durchmesser 8 cm nicht überschreitet . Die Klauen sind an der Rückseite des Roboters befestigt , um zu verhindern , dass Sensoren an der Vorderseite blokiert werden , aber dies zwingt den Roboter , sich zu drehen , wenn ein Objekt gefunden wird , und erhöht somit unnötige Bewegungen .

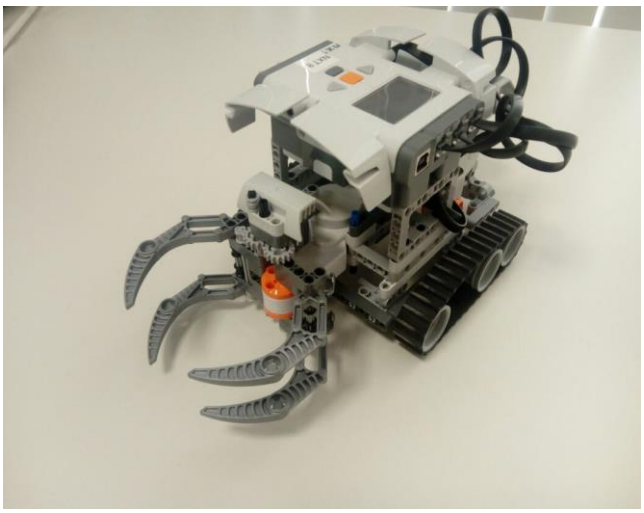


**Abbildung 2 : Das Fahrzeug (die Rückseite)**

## *4 _ Der Programmablaufplan :*

Das Steuerprogramm ist in zwei Hauptabschnitte unterteilt, der erste Abschnitt enthält die Suche nach dem Ziel und der zweite Abschnitt führt den Prozess des Zurückbringens des Ziels an den Startpunkt aus.

***4.1 _ Suchvorgang :*** (Abbildung 3). Vor dem Start: Der Roboter misst die Entfernung bis zum davor liegenden Körper und speichert sie, die gemessene Entfernung ist in Zentimetern angegeben und der Ultrasonic-Sensor kann keine Objekte erkennen, die mehr als 225cm entfernt sind. Nach dem Messen der Entfernung wandelt das Programm diese Entfernung unter Verwendung der Gleichung [Grad=(Entfernung-2)*35] in Grad um. 1cm entsprechen 35 Grad. Wir subtrahieren (2cm) vom Entfernungswert, um Kollisionen mit Objekten zu verhindern. Somit können die Motoren relativ genau geste- uert werden, indem sie um die errechneten Grad gedreht werden. Damit wird das Fahrzeug genau bewegt, um das Ziel zu erreichen. Jetzt bewegt sich der Roboter vorwärts , bis er den Körper erreicht und stoppt dann und prüft die Farbe dieses Körpers durch den Farbsensor. Hier verarbeitet das Programm drei Fälle.

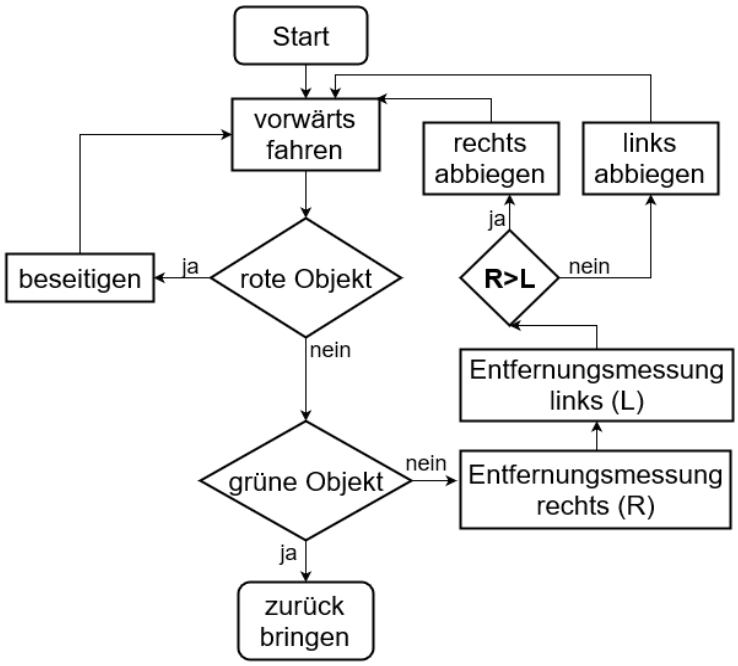



**Abbildung 3 : Der Programmablaufplan**

***Erstens:*** Hat das Objekt eine andere Farbe als Grün oder Rot, wird es als ortsfestes Hindernis betrachtet, eine Wand zum Beispiel, und der Roboter soll sie vermeiden. Er dreht sich zuerst nach rechts um 90° und misst die Entfernung bis zum nächsten Körper und speichert sie, dann dreht er sich um 180 Grad , misst die Entfernung wieder. Nachdem er die Entfernung in beiden Richtungen gemessen hat, vergleicht er die beiden gemessenen Entfernungen und wählt dann das am weitesten entfernte Objekt, fährt dorthin und überprüft die Farbe. Der Grund für die Wahl der größeren Entfernung, ist den möglichen Suchraum zu vergrößern.

***Zweitens:*** Rotes Objekt: Es wird als ein Hindernis angesehen, das beseitigt werden muss. Das Fahrzeug dreht sich um 180 Grad, um das Objekt zu greifen, da sich die Klauen auf der Rückseite des Roboters befinden. Nachdem er den Gegenstand gegriffen hat, misst der Roboter links und rechts, um sicherzustellen, dass genug Platz ist, um das Objekt zur Seite wegzuräumen. Wenn die hier gemessene Distanz in einer Richtung größer als 15 cm ist, bedeutet das, dass der Roboter hier genug Platz dafür hat. Nach dem Abladen des Objekts kehrt der Roboter zu seinem ursprünglichen Weg zurück, und folgt seiner Bewegung direkt zum nächsten Objekt.

***Drittens:*** Das Objekt ist grün: Die grüne Farbe zeigt an, dass das Objekt gerettet oder zurückgebracht werden muss. Der Roboter dreht sich um 180 Grad, um ihn zu greifen, und beginnt die Rückfahrt auf demselben Pfad, um zum Startpunkt zurückzukehren.

***4.2 _ Rückfahrt :*** Sobald das gewünschte Ziel gefunden wurde, muss es zum Startpunkt (Sicherheitszone) gebracht werden. Die Rückfahrt hängt nicht von den Sensoren ab, sondern von den während des Suchvorgangs gespeicherten Entfernungen und Richtungen, sodass das

Fahrzeug denselben Weg zurück zum Startpunkt fahren kann. Während der Suche werden die vom Roboter ausgeführten Schritte als Werte in zwei Vektoren gespeichert. In dem ersten Vektor werden die gemessenen Entfernungen gespeichert, in dem zweiten Vektor werden die vom Roboter gewählten Richtungen, darin werden die Werte (-1, 0, 1) verwendet, um die Trends (links, geradeaus, rechts) in Folge zu bezeichnen. Anhand der in beiden Vektoren gespeicherten Werte kann der Roboter seinen Weg ohne Hilfe finden, und es besteht keine Notwendigkeit, Sensoren zu verwenden oder nach einem anderen Weg zu suchen.

## *5 _ Die Ergebnisse :*

Bei den praktischen Tests des Roboters in einer modellierten Umgebung, die wir im Voraus vorbereitet haben, war es in den meisten Fällen möglich, das gewünschte Ziel zu finden und ohne menschlichen Eingriff zum Ausgangspunkt zurückzukehren. Aber die Ergebnisse waren nicht alle befriedigend. In einigen Fällen hatte der Roboter einige Probleme, das Ziel zu finden. Zum Beispiel könnte der Roboter die entgegengesetzte Richtung zum Zielobjekt wählen und daher könnte es sehr lange dauern, um es zu finden. In anderen Fällen kann er in einer unendlichen Suchschleife festhängen. Aufgrund der einfachen Steuerung des Roboters durch eine GUI könnte das Steuerungsprogramm direkt und schnell modifiziert werden und der Suchalgorithmus kann verbessert werden, um diese Probleme zu vermeiden.

## *6 _ Zusammenfassung :*

Trotz des enormen Potenzials und der Flexibilität der LEGO-Mindstormbauteile, müssten wir viele Änderungen und Verbesserungen in allen Bereichen vornehmen, um unseren Roboter in Zukunft einsetzen zu können. Eine Verbesserungsmöglichkeit wäre, den Farbsensor durch einen spezifischen Satz von Sensoren zu ersetzen, sowie PIR-Sensoren, die verwendet werden, um die Bewegung des menschlichen Körpers zu erkennen, Temperatursensoren, die zur Messung der menschlichen Körpertemperatur verwendet werden und Herzschlagsensoren, die verwendet werden, um eine digitale Ausgabe des Herzschlags zu messen, wenn ein Finger darauf platziert wird. All diese Sensoren geben Auskunft über die Existenz eines lebendigen menschlichen Körpers, somit wäre der Roboter in der Lage, zwischen starren Objekten und lebenden Körpern zu unterscheiden d.h. er kann Überlebende effektiver finden und damit in vielen katastrophalen Szenarien eingesetzt werden und viele Leben retten.

## *Quellenangaben :*


1) CoSpace Robot Rescue Simulator 2016 (Secondary) : http://cospacerobot.org/documents/CSR-Rescue%202016%20Help%20(Secondary)/index.html#!welcomeToCospaceRescue
2) EE-10 Senior Design Page , https://sites.google.com/a/temple.edu/sd-ee10-2010/home
3) International Journal of Research and Scientific Innovation (IJRSI) | Volume IV, Issue XII,December 2017 | ISSN 2321–2705 : http://rsisinternational.org/journals/ijrsi/digital-library/volume-4-issue-12/28-32.pdf
4) WIKIPEDIA , https://de.wikipedia.org/wiki/Roboter#Technische_Grundlagen

# „Lego-Kassettenrekorder“ Lego-Mindstorms-Praktikum

Inga Brockhage, ETIT
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Mithilfe des Lego Mindstorms NXT, mehreren Sensoren und der Programmierumgebung MatLab konnte im Zuge des Lego-Mindstorms-Praktikums mit unserem Kassettenrekorder die ursprüngliche Funktionsweise eines Kassettenrekorders wieder aufleben. Die Handhabung ist dabei ebenfalls dem Original nachempfunden, sodass auch unser Kassettenrekorder intuitiv bedient werden kann. Die Informationscodierung basiert dabei allerdings auf einem doppelten Farbcode, durch den jede Note einzeln codiert ist. Durch diese Codierung werden zwei Oktaven abgedeckt, sodass verschiedenste Melodien wiedergegeben werden können. Zwei Farbsensoren ober- und unterhalb der Stelle, durch die das Tape läuft, lesen den entsprechenden Code ein. Durch unser Programm werden daraufhin die entsprechenden Töne über den NXT wiedergegeben. Letztendlich entstand ein voll funktionsfähiger Kassettenrekorder mit zwei passenden Tapes, die problemlos gewechselt werden können.**

***Schlagwörter*—Kassette, Kassettenrekorder, Lego Mindstorms Praktikum, MatLab, Musik, Tapes**

## I. Einleitung

DAS Ziel dieses Praktikums war es, eine selbstgewählte Projektidee mittels der Produktserie Lego-Mindstorms umzusetzen. Dazu gehört der Steuerungscomputer NXT, der durch den Einsatz von MatLab programmiert werden kann, bis zu drei Motoren und verschiedensten Sensoren, sodass damit ein großes Spektrum an Projektmöglichkeiten abgedeckt wurde. Zusammen mit meinem Kommilitonen Max Tzschoppe entstand die Idee, die Funktionsweise eines Kassettenrekorders vollständig nachzubilden. Dabei galt es eine neue Informationscodierung zu wählen, da originale Tapes mit den zu Verfügung gestellten Sensoren nicht ausgelesen werden können. So entstand ein doppelter Farbcode, der aus der Kombinationen von fünf Farben besteht und jeweils bestimmten Noten beziehungsweise den Systemanweisungen Start, Pause, Stopp und Leerlauf zugeordnet ist. Diese Codierung kann über Farbsensoren erkannt werden und somit über ein Programm den entsprechenden Funktionen zugeordnet werden. Damit können die jeweiligen Frequenzen über den NXT wiedergegeben werden.

## II. Vorbetrachtungen

Erste Inspiration gab es durch ein Projekt aus dem Vorjahr, bei dem ein Schallplattenspieler realisiert wurde, da auch bei diesem Projekt die Wiedergabe von Tönen und dessen Codierung mit dem NXT behandelt wurde. Als Vorlage für unseren Kassettenrekorder diente in erster Linie natürlich auch ein echter Kassettenrekorder mit seiner Funktionsweise in Kombination mit den Kassetten.

DOI: 10.24352/UB.OVGU-2018-040 

### A. Lego NXT Music Player (MATLAB turntable)

Bei dem Projekt einer Gruppe aus dem Vorjahr wurde ein Schallplattenspieler mithilfe des Lego-Mindstorms nachgebaut, der in Abbildung 1 zu sehen ist. Dafür verwendete die Gruppe einen doppelten Farbcode, der sich auf der Ober- und Unterseite der Schallplatte befindet, als Toncodierung. Durch einen Motor wird die selbstgemachte Schallplatte in Rotation versetzt. An einem Arm befinden sich zwei Farbsensoren, die so angebracht sind, dass die Ober- und Unterseite der selbstgemachten Schallplatte gescannt wird. Ein zweiter Motor bewegt den Arm langsam in Richtung Mitte der Schallplatte, sodass die gesamte Melodie abgespielt wird. Da ein Schallplattenspieler schon umgesetzt wurde, entwickelte sich die Idee des Lego-Kassettenrekorders, bei dem wir ebenfalls eine doppelte Farbcodierung der Töne verwenden.

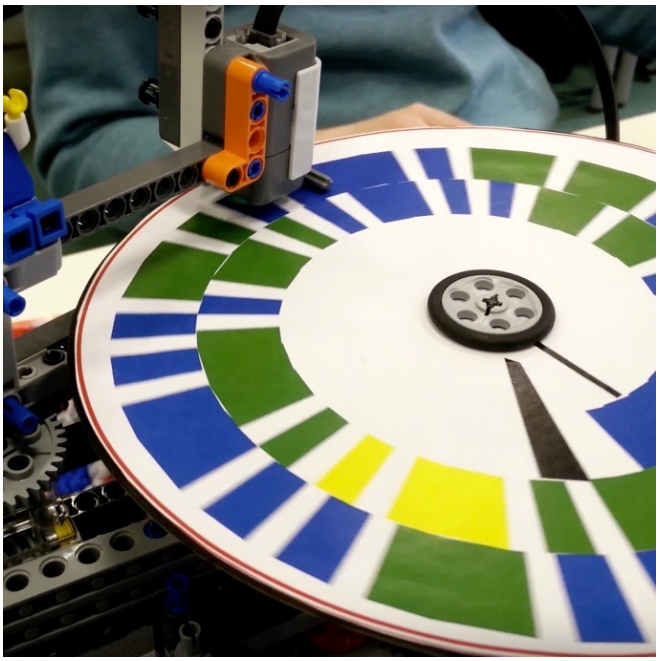


Abbildung 1: Lego-Schallplattenspieler [1]

### B. Kassettenrekorder

Als Vorlage für den Bau unseres Kassettenrekorders diente in erster Linie selbstverständlich ein originaler Kassettenrekorder und dessen Kassetten. Unser Lego-Kassettenrekorder sollte seinem Vorbild in Design und Funktionsweise weitestgehend entsprechend der Möglichkeiten nachkommen. Somit ist einer

der wichtigsten Punkte das Wechseln der Kassetten, aber auch die für einen Kassettenrekorder typische Funktion des Zurückspulens sollte möglich gemacht werden. Allgemein war es auch ein Ziel, möglichst kompakte Konstruktionen zu entwickeln.

## III. Hauptteil

### A. Grundlagen

Um den Kassettenrekorder mit Hilfe von Lego wieder aufleben zu lassen, lag der Fokus vor allem auf dem Prinzip der Kassetten und der damit verbunden Möglichkeit diese zu wechseln, um unterschiedliche Lieder wiedergeben zu können. Auch sollten natürlich die mit einem Kassettenrekorder in Verbindung gebrachten Standartfunktionen eingebracht werden. Besonders wichtig ist dabei, neben den selbstverständlichen Funktionen Play und Stopp, natürlich die Zurückspulen-Funktion, damit die Tapes nicht per Hand zurückgedreht werden müssen. Desweiteren gibt es noch die Funktion des Vorspulens.

Um die Wiedergabe von Tönen grundsätzlich zu realisieren, entschlossen wir uns einen doppelten Farbcode zu benutzen, der entsprechend auf der Ober- und Unterseite des Tapes gedruckt wird. Damit der Code wieder in ein Tonsignal umgewandelt wird, werden zwei Farbsensoren benutzt, zwischen denen das Tape hindurch läuft. Bei den Farben haben wir uns auf Blau, Rot, Grün, Schwarz und Weiß beschränkt, da diese am besten von den Farbsensoren erkannt werden. Letztendlich konnten durch die Kombination dieser fünf Farben 21 Noten und damit zwei Oktaven codiert werden. Dazu kommen noch die Codierungen von Start, Pause, Stopp und Leerlauf. Ein beispielhafter Teil unseres Codes ist in Abbildung 2 zu sehen. Grundsätzlich ist jedes Tape gleich aufgebaut. Das erste Stück des Tapes hat die Codierung für Leerlauf. Dann kommt die Startcodierung, die Farbcodierungen für die Noten und die Stoppcodierung. Auch hinter der Stoppcodierung folgt erneut ein Abschnitt die Codierung für Leerlauf.

| Code | C | D | E | Start | Stopp |
|---|---|---|---|---|---|
| oberer Farbcode | | | | | |
| unterer Farbcode | | | | | |

Abbildung 2: Beispielhafter Ausschnitt des Farbcodes

### B. Tapeproduktion

Die Produktion der Tapes beruhte auf der Trial-and-Error-Methode. Da bei dem Druck der Tapes auf A4 dieses in mehrere Abschnitte geteilt werden muss, stellte sich die Frage der besten Verbindung der einzelnen Tape-Abschnitte. Die Verbindung soll sich nicht wieder lösen und gleichzeitig nur geringfügig dicker sein als der Rest des Tapes, sodass die Kanten nicht während des Abspielens hängen bleiben. Letztlich funktionierte es am besten, zwischen zwei Abschnitten als Verbindung ein Stück Papier zu kleben und die Kante mit Tesafilm festzukleben. Auch stellte sich heraus, dass es besser ist Ober- und Unterseite der Tapes nebeneinander zu drucken und dann diese zusammenzufalten, statt zwei einzelne Streifen zusammenzukleben.

### C. Hardware

Im Bereich der Hardware gab es zwei Konstruktionen zu meistern, die entsprechend auf einander abgestimmt sind. Zum einen musste aus Legosteinen eine Art Kassette gebaut werden, die in Abbildung 3 zu sehen ist. In dieser sollte das Tape von einer Rolle auf eine weitere Rolle aufgerollt werden. Während dieses Vorgangs erfolgt das Ablesen der Informationen. Durch die beschränkten Möglichkeiten des Legos ähnelte unsere Kassette von der Größe her letztendlich eher einer alten Videokassette als einer Musikkassette, doch das Prinzip wurde verwirklicht. Zwischen den zwei Rollen befinden sich mehrere Liftarme [2], um das Tape in Position zu halten. Später wurde ein Teil der Liftarme durch Führungsrollen ersetzt, wodurch das Tape gleichmäßiger bewegt wird und die Verbindungen der Tape-Abschnitte nicht mehr hängen bleiben können.

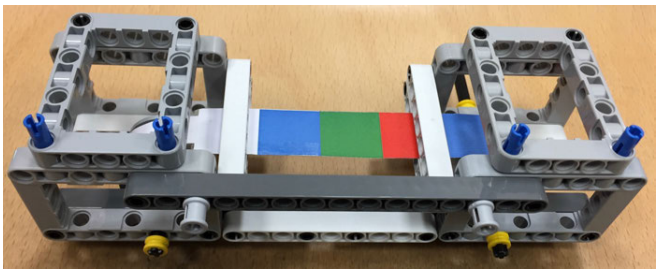


Abbildung 3: Kassette mit Tape

Passend zu der Kassette entstand der Kassettenrekorder (siehe Abbildung 4 beziehungsweise Abbildung 5). Dieser sollte möglichst kompakt sein und besteht im Grunde lediglich aus den zwei Farbsensoren mittig auf der Vorderseite und einem Motor auf der Rückseite hinter dem NXT, der sich in der Mitte des Kassettenrekorders befindet. Durch den Motor werden beide Rollen der Kassette angetrieben. Die Verbindung des Motors mit den Rollen stellte sich dabei als recht kompliziert heraus. Da die Kassetten ohne weiteren Aufwand gewechselt werden sollen, darf die Verbindung nicht fest verbaut sein. So entstand letztendlich eine einfache Steckverbindung von zwei Achsen pro Rolle. Eine jeweils fest an der Kassette verbaut, während die andere über zwei rechtwinklig zueinander stehenden Zahnrädern fest mit dem Motor verbaut ist und über einen Stecker mit der sich an der Kassette befestigten Achse verbunden werden kann. Zusätzlich befinden sich auf dem Kassettenrekorder noch zwei Tastsensoren, die die Funktion der Play- und Zurückspulen-Taste übernehmen, sodass unsere Konstruktion noch mehr einem wirklichen Kassettenrekorder gleicht. Damit die Kassetten nicht mit einander verwechselt werden können, befinden sich an der Vorderseite der Kassetten verschieden farbige Legosteine, wie sie an der Kassette in Abbildung 5 zu sehen sind.

### D. Software

Das passende Programm in der Programmierumgebung MatLab besteht, wie im Programmablaufplan in Abbildung 6 zu

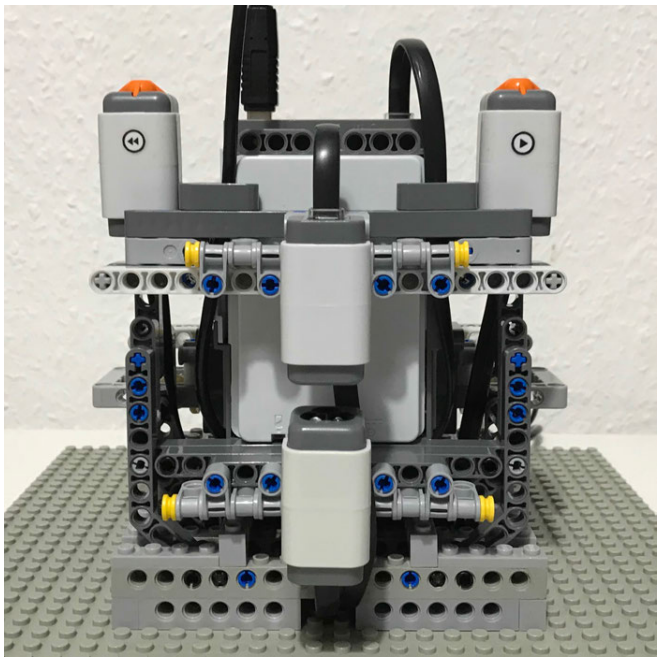


Abbildung 4: Frontansicht des Kassettenrekorders ohne Kassette

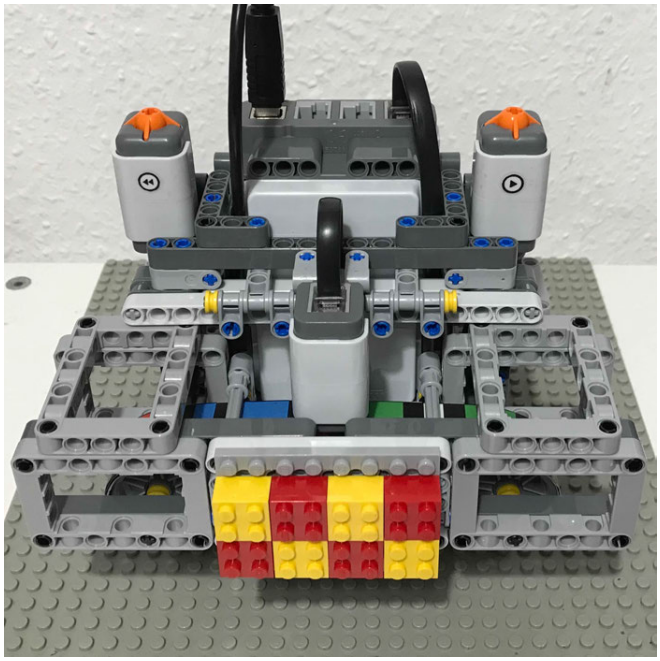


Abbildung 5: Frontansicht des Kassettenrekorders mit Kassette

sehen, hauptsächlich aus einem String Vergleich der Ausgaben der Sensoren. Dazu befinden sich in einer while Schleife mehrere if Abfragen, in denen die jeweiligen Farbkombinationen abgefragt werden und der entsprechenden Note beziehungsweise Frequenz [3] zugeordnet wird, die dann ausgegeben wird. Die Ausgabe von Tönen wird erst gestartet, sobald die Startcodierung gescannt wird. Als Abbruchkriterium fungiert die Stoppcodierung. Beim Zurückspulen erfolgt keine Tonausgabe, obwohl die Sensoren aktiv sind. Wird bei dem Scannvorgang die Startcodierung gescannt, stoppt das Zurückspulen und das Tape ist bereit erneut abgespielt zu werden. Um die Tonlänge zu variieren, wird diese durch die Farbcodelänge codiert. Unsere erste Idee war es, durch eine Zeitmessung zu bestimmen, wie lang eine Farbcode ist und dementsprechend die Tonlänge festzulegen. Doch um einiges leichter ist die finale Lösung, dass durch ständiges Scannen ein Ton so lange ausgegeben wird, wie die entsprechende Codierung gescannt wird. Da nun die Töne in einander übergehen würden, gibt es zwischen zwei Tönen jeweils einen kurzen Abschnitt mit der Codierung für Pause. Zu dem Kassettenrekorder gehört auch eine grafische Benutzeroberfläche (GUI), die die Tasten Play, Stopp, Zurückspulen und Vorspulen enthält. Die GUI kann parallel zu den zwei Tastsensoren, als Knöpfe für Play und Zurückspulen, benutzt werden und erleichtert damit die Bedienung des Kassettenrekorders. Mit dieser Codierung haben wir die Titelmelodien von der „Sendung mit der Maus“ [4] und von „Löwenzahn“ [5] verwirklichen können. Beide Stücke sind einstimmig und auch leicht zu erkennen.

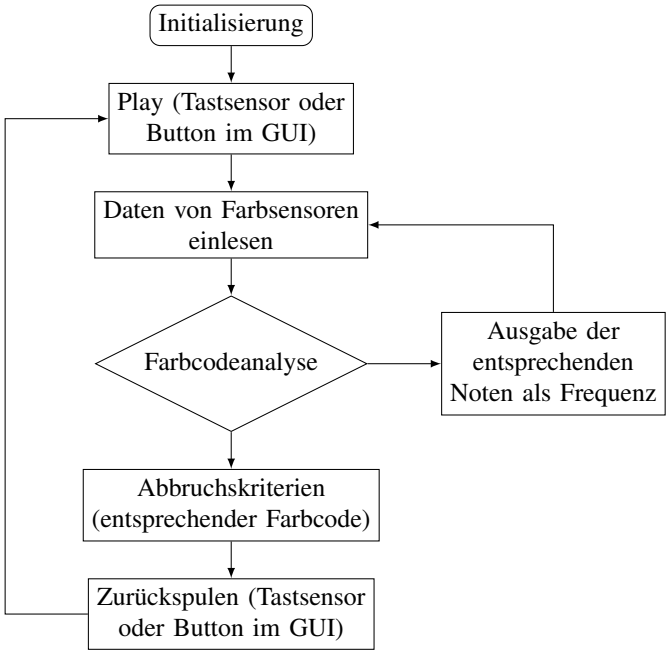



Abbildung 6: PAP des Kassettenrekorders

## IV. Ergebnisdiskussion

Am Ende der einwöchigen Arbeit an dem Projekt steht ein fertiger und vollständig funktionsfähiger Kassettenrekorder und zwei dazu passenden Kassetten mit den Tapes zu den Titelmelodien von der „Sendung mit der Maus“ und „Löwenzahn“. Zwar ist die Informationscodierung eine andere, so ist dennoch die deutliche Parallele zum Konzept der Kassetten nicht zu übersehen. Durch die Farbcodierung sind im Gegensatz zum Original lediglich einstimmige Melodien möglich, in denen sich keine Töne überlagern. Allerdings gibt es keine nennenswerte Längeneinschränkung bei unseren Tapes, wie es eventuell zu erwarten war.

Zusammenfassend gab es zwei wesentliche Probleme, die teilweise behoben beziehungsweise reduziert werden konnten. Als erstes machte die Tapeproduktion an sich Probleme. Zwar hat normales Druckerpapier eine gute Dicke, sodass das Tape ohne weiteres aufgerollt werden kann und beide Sensoren auch nur die Farbe auf der jeweiligen Seite scannen. Auch war somit die Produktion der Tapes grundsätzlich vergleichsweise einfach, da diese abschnittsweise ausgedruckt werden können und somit die Farben immer denselben Farbton haben, deckend und gleichmäßig sind. Ein komplettes Tape besteht damit allerdings aus vier beziehungsweise sieben Stücken, die miteinander verbunden werden müssen, was wiederrum zu Problemen führte. Um Ober- und Unterseite miteinander zu verbinden, werden diese nebeneinander gedruckt, gefaltet und dann zusammengeklebt. Damit ist sichergestellt, dass die Codierungen auch wirklich exakt übereinander liegen. Um diese Abschnitt schließlich miteinander zu verbinden wird ein Stück Papier zwischen die jeweiligen Enden geklebt. Damit die Kanten nicht hoch stehen und hängen bleiben können, fixiert ein dünner Tesafilm-Streifen den Übergang. Andere Klebe- und Verbindungsarte führten zu Problemen beim Aufwickeln, da die Verbindung zum Beispiel zu steif war oder es kam zu Problemen bei der gleichmäßigen Bewegung, wenn sich die Verbindung zu sehr von dem restlichen Tape unterscheidet.

Das zweite und auch nur teilweise gelöste Problem entstand durch die Farbsensoren. Die Ergebnisse der Sensoren waren fehlerhaft. Vor allem Schwarz wurde öfter als Blau oder Grün erkannt. So entstanden sehr viele Störgeräusche als Schwarz-Schwarz die Codierung für Pause war, während die Codierung Weiß-Weiß wesentlich weniger Störgeräusche hervorrief, wenngleich Weiß auch mal als Gelb interpretiert wurden. Sicherlich sind diese Fehler durch Veränderung des Abstandes zum Tape und der Umgebungshelligkeit zu reduzieren, dennoch brachten unsere Versuche dazu kaum Besserung. Um möglichst viele Störgeräusche zu verhindern, gibt es nun zwei Programmversionen, jeweils für jedes Tape eine, in der nur die Codierungen einem Ton zugeordnet sind, die auch wirklich gebraucht werden. Damit ist das Problem zumindest teilweise gelöst, da die Anzahl an möglichen Störfrequenzen reduziert ist.

## V. Zusammenfassung und Fazit

Mit diesem Projekt wurde erfolgreich eine doppelte Farbcodierung zur Informationsübertragung in Form von Tönen erreicht. Dies geschah in der Form eines Kassettenrekorders mit wechselbaren Kassetten. Zu dessen Funktionsweise gehört eine GUI zur Steuerung des Abspielens, wobei die zwei wichtigsten Funktionen auch über zwei Tastsensoren erreicht werden. Somit ist dieser Kassettenrekorder vollständig funktionsfähig und mit seinem Vorbild vergleichbar. Zu ergänzen sind selbstverständlich noch weitere Tapes, die problemlos entsprechend der Codierung hergestellt werden können. Um der GUI noch einen größeren Nutzen zuzuschreiben, wäre eine Ausgabe der aktuellen Note beziehungsweise der bereits ausgebenden Noten möglich. Auch wäre eine Verbesserung der Ergebnisse der Farbsensoren durch Nutzung anderer Sensoren oder dauerhafte Veränderung der Umgebungshelligkeit definitiv sinnvoll. Um Störgeräusche noch weiter zu verringern ließe sich auch die Scannhäufigkeit noch verändern und perfekt auf die Motorgeschwindigkeit anpassen, damit nur so häufig wie unbedingt nötig gescannt wird. Trotz diesen Verbesserungsmöglichkeiten ist der Lego-Kassettenrekorder auch jetzt schon ein vollständiges System und somit in der Lage als Kassettenrekorder verwendet zu werden und Melodien abzuspielen.

## Literaturverzeichnis


[1] Nihilianth: *Lego NXT Music Player (MATLAB turntable)*. Internet. https://www.youtube.com/watch?v=6J-YEVI1Kms. Version: März 2018, Abruf: 02.03.2018

[2] Buckdahn, Tobias: *Lego-Vokabular*. Internet. https://www.brickup.de/vokabular. Version: März 2018, Abruf: 01.03.2018

[3] Sandler, Anna: *How to Make Lego NXT to Play a Sound*. Internet. http://www.robotappstore.com/Knowledge-Base/How-to-Make-Lego-NXT-to-Play-a-Sound/35.html. Version: Februar 2018, Abruf: 20.02.2018

[4] Fritz, Emonts: *Posegga Hans: Die Maus - Titelmusik der Sendung mit der Maus*. Internet. https://www.musikalienhandel.de/noten/klavier-(4ms)-(klav-(4ms))/die-maus-titelmusik-der-sendung-mit-der-maus--ED+8491.htm. Version: Februar 2018, Abruf: 23.02.2018

[5] Hirte, Patrick: *Matthias Raue: Löwenzahn Thema*. Internet. https://musescore.com/user/1651211/scores/4113256. Version: Februar 2018, Abruf: 22.02.2018

# „Kassettenrekorder“ Lego-Mindstorms-Praktikum

Max Tzschoppe, ETIT
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Das Projekt „Kassettenrekorder“ entstand im Rahmen des Lego-Mindstorms-Praktikums und hat das Ziel einen original Kassettenrekorder zu rekonstruieren. Die Funktionen des Kassettenwechsels und des Zurückspulens sind so integriert, dass sie dem Benutzer ein einfaches Bedienen ermöglichen. Bei der Umsetzung der Konstruktion eines Tapes wurde ein doppelter Farbcode entwickelt. Dieser ermöglicht eine Codierung von komplexen Melodien über zwei Oktaven und enthält zusätzliche Sonder-Codierungen für die Steuerung am Start und am Ende des Tapes. Mit Hilfe von zwei Farbsensoren wird der Code von oben und von unten eingescannt. Die entsprechend dazu entwickelte Software decodiert den eigescannt Farbcode und gibt über den NXT den entsprechenden Ton aus.**

***Schlagwörter*—Kassette, Kassettenrekorder, Lego-Mindstroms-Praktium, MATLAB, Musik, Tapes**

## I. Einleitung

Im Rahmen des Lego-Mindstorms-Praktikums galt es ein selbstgewähltes Projekt zu realisieren, um dem Thema „Elektro- und Informationstechnik“ näher zu kommen. Zur Verfügung stand der Lego-Mindstorms-Roboter, der aus einem Hauptcomputer, genannt „NXT“, besteht. Dieser wurde von der RWTH Aachen entwickelt und mit MATLAB programmiert. An den NXT lassen sich viele verschiedene Sensoren anschließen. Die Daten der Sensoren werden von dem NXT erfasst und können mit dem Programm „MATLAB“ ausgelesen werden. Das Ziel unseres Projektes war den Kassettenrekorder wieder aufleben zu lassen. Zusammen mit Inga Brockhage entwickelten wir einen Kassettenrekorder, der dem Original weitestgehend nachempfunden ist und dessen Standardfunktionen alle vorhanden sind.

## II. Vorbetrachtungen

Zu Beginn des Lego-Mindstorms-Praktikums ließen wir uns von mehreren schon existierenden Projekten inspirieren. Bei der Präsentation der Projekte des letzten Jahres wurde ein Plattenspieler vorgestellt. Das Interesse wahr sehr groß, woraufhin uns nach einigen Überlegungen die Idee für den Bau eines Kassettenrekorders kam. Daraufhin recherchierten wir im Internet nach ähnlichen Projekten und stießen dabei auf ein Gerät. Dieses Gerät ähnelte einem Kassettenrekorder und wir übernahmen die Idee der Konstruktion und Umsetzung in unser Projekt.

### A. Projekt: „Schallplattenspieler“

Das Konstruieren eines Schallplattenspielers haben sich zwei Studenten der OvGU-Magdeburg im Jahre 2017 zur Aufgabe gemacht. Dabei entwickelten sie einen doppelten Farbcode, mit dem sie die Toncodierung vornahmen. Zur Decodierung setzten sie zwei Farbsensoren von Lego-Mindstorms ein. Durch einen Motor wird die mit dem Farbcode bestückte Schallplatte in Rotation versetzt. Die Farbsensoren scannen dabei den Farbcode und mittels einer Software wird der Farbcode decodiert und die Melodie wird auf dem Lego-Mindstorms-NXT wiedergegeben. Besonders die Idee der Codierung der Noten mit Hilfe des doppelten Farbcodes übernahmen wir für unser Projekt und entwickelten daraus ein Tape.

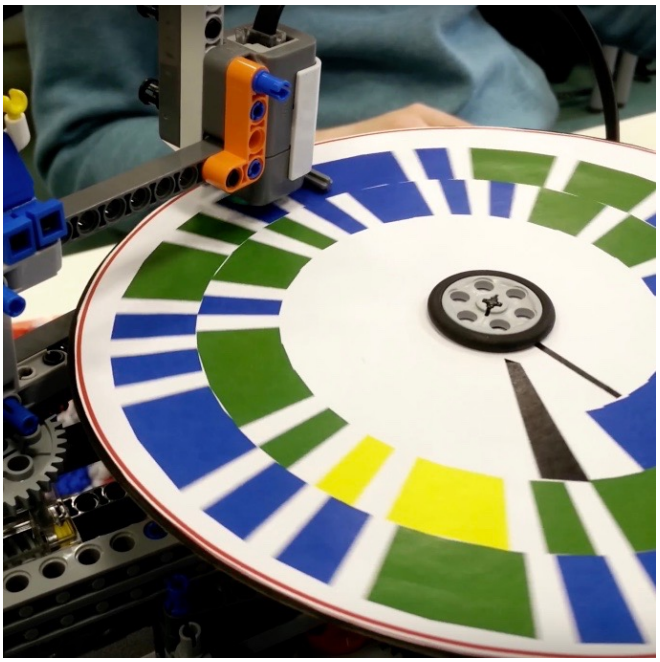


Abb. 1: Schallplattenspieler [1]

### B. Original Kassettenrekorder

Der Original Kassettenrekorder sollte für unser Projekt als Vorbild dienen. Unser Ziel war es einen Kassettenrekorder, soweit es mit Lego-Mindstorms möglich ist, nachzubauen. Dabei sollten die Funktionen des Wechseln einer Kassette und die Möglichkeit den Rekorder über Tasten zu bedienen in unserem Projekt umgesetzt werden. Zudem sollte die Kassette wie beim Original aufgebaut sein. Das heißt, die Tapeaufwicklung wird über zwei Rollen vorgenommen. Der Rekorder sollte außerdem in der Lage sein die Kassette vor dem Wechseln zurückzuspulen.

DOI: 10.24352/UB.OVGU-2018-041 

## III. Hauptteil

Bei der Umsetzung des Projekts einen Kassettenrekorder zu bauen, achteten wir stets darauf das Konzept weitestgehend einzuhalten. Der Anspruch lag darin, einen Kassettenrekorder, soweit es die Möglichkeiten mit Lego erlauben, nachzubauen und dabei die üblichen Standardfunktionen eines Kassettenrekorders mit in das Projekt einfließen zu lassen. Mehrere Kassetten zu bauen und Tapes zu entwickeln gehörte ebenso zum Konzept wie auch ein Kassettenrekorder zu bauen, in dem die Funktionen des Abspielens sowie des Vor- und Zurückspulens integriert sind. Ein Augenmerk lag dabei auf dem Separieren von Kassettenrekorder und Kassette. So ist es möglich verschiedene Kassetten zu benutzen, wobei der Rekorder universell einsetzbar ist. Durch die entsprechenden Sensoren, die an dem Kassettenrekorder angebracht sind, kann ein beliebiges Tape eingescannt und die Tonabfolge wiedergegeben werden. Das Tape besteht aus einer doppelseitigen Farbcodierung. Somit können verschiedene Töne codiert und die entsprechenden Frequenzen [2] über den NXT wiedergegeben werden.

### *A. Mechanik und Konstruktion*

Zuerst entstand ein kassettenähnlicher Protoyp, der in Abb. 3 abgebildet ist und nahezu der endgültigen Kassettenkonstruktion entspricht. Das Prinzip basiert auf dem Aufbau einer originalen Kassette. Zwei Rollen wickeln das Tape auf. Um die Rollen befindet sich ein Kasten, der für die nötige Stabilität sorgt. Die Rollen sind auf einer Achse gelagert, die ungefähr zwei bis drei Zentimeter über die Konstruktion herausragt. Dies ermöglicht die Kassette in den Rekorder zu stecken und es entsteht somit über die Achse eine stabile Verbindung zwischen der Kassette und dem Rekorder. Zwischen den beiden Rollen ist etwas Abstand. Dieser Raum wird benötigt, damit die Sensoren so platziert werden können, dass sie über und unter dem Tape liegen. Die zwei Farbsensoren sind in der Mitte in Abb. 4 zu erkennen. Um ein waagerechtes und sauberes Führen des Tapes zu gewährleisten, sind an beiden Seiten Führungsrollen angebracht. Nach oben sind die Führungsrollen mit einem Liftarm [3] begrenzt. Diese Konstruktion bietet eine optimale Führung des Tapes und verhindert Ungenauigkeiten beim Durchlaufen des Tapes. Damit die Kassetten unterscheidbar sind, sind an der Vorderseite unterschiedlich farbige Steine angebracht.

| Code | C | D | E | Start | Stopp |
|---|---|---|---|---|---|
| oberer Farbcode | | | | | |
| unterer Farbcode | | | | | |

Abb. 2: Farbcode

Danach begann der Bau des Kassettenrekorders. Zuerst entwickelten wir ein Gerüst, das zwei Lichtsensoren so positioniert, dass sie beim Einführen einer Kassette genau über und unter dem Tape liegen (vgl. Abb. 4). Links- und rechtsseitig befindet sich eine Konstruktion, die das Einführen der Achsen der Kassette ermöglicht. Hinter den Farbsensoren ist der NXT gelagert. Durch die mittige Position können die jeweiligen Sensoren optimal angeschlossen werden und es ist möglich direkt auf dem NXT Einstellungen vorzunehmen. Auf der Rückseite montierten wir den Motor, der die Kassette antreibt. Der Motor ist über eine Zahnrad-Konstruktion mit der Achse verbunden, die seitlich am NXT vorbei zur Kassette führt. Zum Schluss wurden zwei Tastsensoren montiert, die in Abb. 4 im oberen Drittel abgebildet sind. Diese sollen dem Nutzer ermöglichen, den Rekorder mittels einer Start- und Zurückspulfunktion zu bedienen.

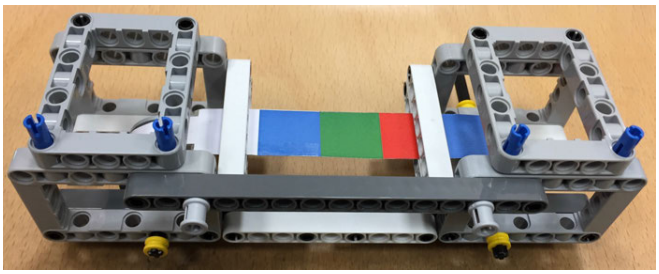


Abb. 3: Kassette

Abschließend fehlte nur noch ein Tape. Dafür entwickelten wir einen Farbcode, der ausschnittweise in Abb. 2 dargestellt ist. Aufgrund des eingeschränkten Farbspektrums blieben die Farben Rot, Blau, Grün, Weiß und Schwarz zur Auswahl. Die Farben codieren die einzelnen Töne und Schwarz und Weiß erhielten Sondercodierungen für Start, Stopp, Leerlauf und Pause. Nach der Festlegung des Farbcodes für die entsprechenden Noten, die für die Beispiellieder „Die Maus - Titelmelodie von der Sendung mit der Maus“ [4] [5] und „Löwenzahn Thema“ [6] benötigt werden, erstellten wir die Farbcodeabfolge mit Hilfe einer Excel-Tabelle. Am Anfang des Tapes befindet sich die Leerlauf- und Startcodierung und am Ende die Stopp- und Leerlaufcodierung. Zwischen den Tönen sind kurze weiß-weiß Codierungen, damit eine minimale Pause zwischen den Tönen entsteht und die Töne nicht ineinander klingen. Um die verschiedenen Tonlängen umzusetzen, variierten wir die jeweilige Länge der Codierungen. Alles in allem wird die Steuerung von der Software übernommen und die Bedienung für den Benutzer ist sehr einfach. Es muss nur der Starttaster ausgelöst werden und die Kassette kann automatisch abgespielt werden.

### *B. Software*

Das Programm für den Kassettenrekorder besitzt eine einfache Struktur. Zu Beginn wird die Initialisierung vorgenommen und der NXT wird angesteuert. Danach läuft das Programm in eine while-Schleife. In dieser Schleife befinden sich mehrere if-Abfragen, die die eingescannten Farben der Sensoren analysieren. Erst wenn die Startcodierung eingescannt wird, spielt das Programm nach den entsprechenden Codierungen die jeweilige Tonfrequenz [2] der Noten ab. Die Schleife stoppt beim Abbruchskriterium und das Programm ist am Ende. Wenn der Befehl des Zurückspulens vorliegt, wird der Motor solange in Rotation versetzt, bis die Startcodierung erreicht ist. Der dazugehörige Programmablaufplan (PAP) ist in Abb.

6 dargestellt. Damit die Steuerung und Bedienung benutzerfreundlich ist, erstellten wir eine graphische Benutzeroberfläche (GUI). Über diese kann der Benutzer den Kassettenrekorder starten oder stoppen und gegebenenfalls vor oder zurückspulen. Die Bedienung erfolgt über die jeweiligen Buttons in der GUI. Alternativ können die Funktionen des Startens und des Zurückspulens über die Tastsensoren aktiviert werden. Das Programm erkennt automatisch, von welchem Bedienfeld die Eingabe vorgenommen wird und führt dann die entsprechenden Befehle durch.

## IV. Ergebnisdiskussion

Das Endergebnis ist ein voll funktionsfähiger Kassettenrekorder, der eine beliebige Kassette mit dem entsprechenden Tape abspielen kann. Durch die Taster am Gerät und durch die GUI lässt sich der Rekorder bedienen. Als Muster stehen zwei verschiedene Kassetten zur Verfügung. Einmal die Titelmelodie von der Sendung mit der Maus und die Melodie von Löwenzahn. Es ist durchaus möglich, beliebig viele andere Lieder oder Melodien abzuspielen, vorausgesetzt die Töne liegen in dem Bereich des festgelegten Farbcodes.

Während des Projektes tauchten immer wieder neue Probleme auf. Am schwierigsten stellte sich die Tapeproduktion heraus. Zu Beginn mussten grundlegende Fragen zur Papierbeschaffenheit, der Verbindungen von einzelnen Farbcodeabschnitten und der Befestigung geklärt werden. Normales Druckerpapier erwies sich von der Dicke des Papiers am besten, um daraus ein Tape zu konstruieren und es aufwickeln zu können. Die Farben druckten wir mit einem Laserdrucker auf das Papier. Da die Tapelänge nicht auf ein Papierformat passt, erstellten wir für das „Maus-Tape“ vier und für das „Löwenzahn-Tape“ sieben Abschnitte. Schlussendlich fügten wir die Teilabschnitte mit Hilfe eines Klebestiftes zusammen,

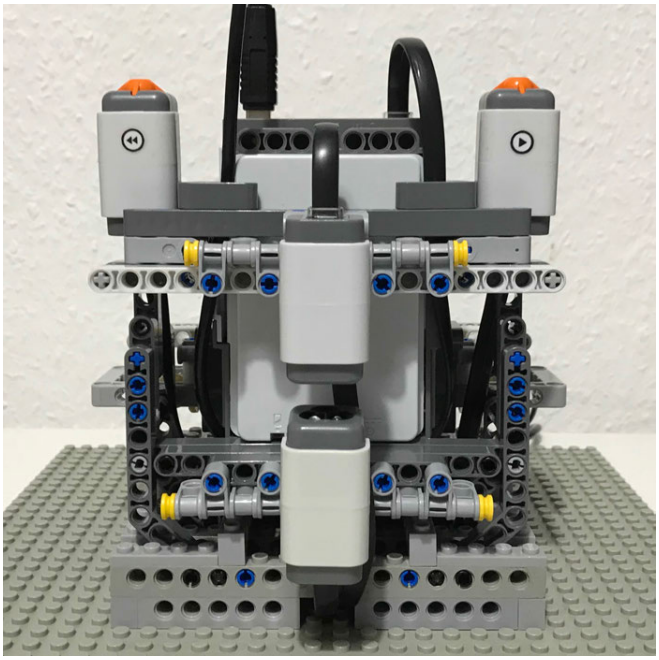


Abb. 4: Sensoren

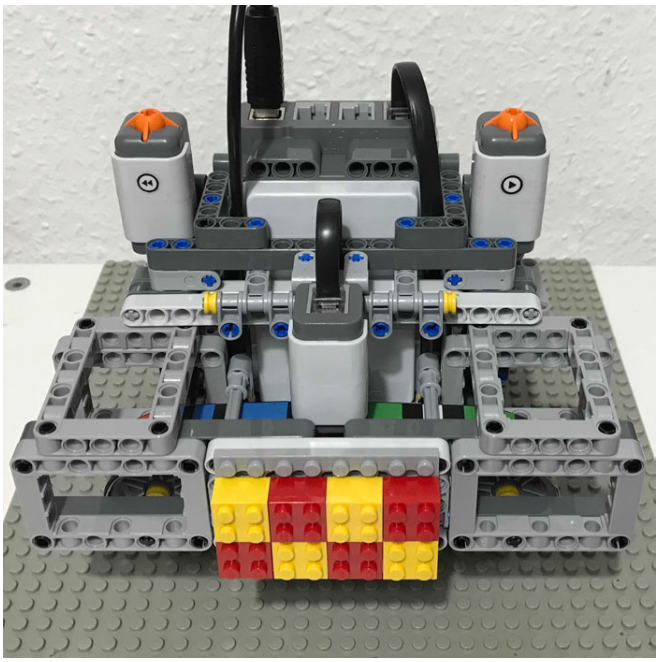


Abb. 5: Frontansicht

da es mit durchsichtigem Klebestreifen zu einer Kantenbildung kam, die zu Fehlern beim Einlesen des Codes führten. Ein weiteres Problem bestand darin, dass die kurzen weiß-weiß Abschnitte zwischen den Tönen teilweise nicht von den Farbsensoren erkannt wurden. Damit erreichten wir nicht den gewünschten Effekt einer minimalen Pause zwischen den Tönen. Um dieses Problem zu umgehen, stellten wir das „Maus-Tape“ mit einer schwarz-schwarz Codierung und das „Löwenzahn-Tape“ mit einer weiß-weiß Codierung für die Pausen her. Zusätzlich änderten wir die Breite der Pausencodierungen. Letztendlich stellte sich heraus, dass sich bei einer weiß-weiß Codierung für Pausen geringere Störgeräusche ergeben. Bei der schwarz-schwarz Codierung ergab sich, dass durch die Doppelcodierung mehrfach Codierungen von den Farbsensoren für Töne eingescannt werden, die in der Realität nicht existierten. So wurde von den Sensoren des Öfteren anstatt Schwarz Blau oder Grün eingescannt. Dadurch entstanden etliche Störgeräusche. Diese konnten wir bis zum Schluss nicht vollständig unterbinden, was schlussendlich an den etwas ungenauen Sensoren gelegen hat. Um diese Störgeräusche zu vermeiden, erstellten wir zwei unterschiedliche Programme, in denen nur die Codierungen vorhanden sind, die für die jeweilige Melodie notwendig sind. Zusätzlich gab es mit den wechselnden Umgebungshelligkeiten immer wieder neue Fehler, die zu weiteren Ungenauigkeiten beim Scannen und zu Störgeräuschen führten. Gleichwohl sind die Melodien klar erkennbar und wir haben somit unser Ziel erreicht, die beiden Melodien abspielen zu können.

## V. Zusammenfassung und Fazit

Während des Lego-Mindstorms-Praktikums entstand ein kompakter Kassettenrekorder, der mehrere Kassetten abspielen kann. Die Bedienung kann sowohl über die entsprechenden

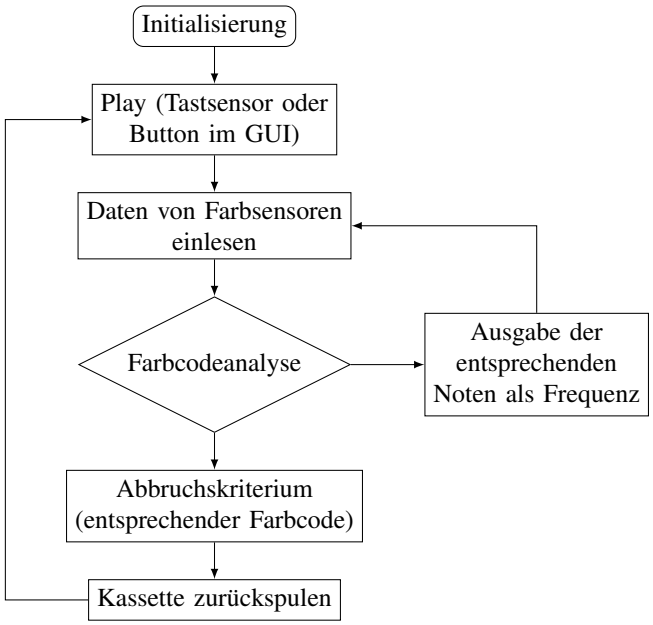



Abb. 6: PAP des Kassettenrekorders

Tasten am Rekorder als auch über die Software vorgenommen werden. Das Tape besteht aus einer doppelten Farbcodierung, die über zwei Farbsensoren eingelesen wird. Das dafür geschriebene Programm decodiert den Farbcode und spielt den entsprechenden Ton über den NXT ab.

Es bestehen einige Verbesserungsmöglichkeiten. Um zum Beispiel ein besseres Klangerlebnis erreichen zu können, müssen die vorhandenen Störgeräusche minimiert werden. Das könnte mit der entsprechenden Umstrukturierung des Programmes erreicht werden. Eine mögliche Erweiterung ist eine Ausgabe in der GUI, die die eingescannten Noten ausgibt und es so dem Benutzer einfacher macht die Melodie zu verfolgen oder mitzusingen. Außerdem ist es nur möglich einstimmige Melodien mit dem Kassettenrekorder abzuspielen. Mit der entsprechenden Umstrukturierung des Tapes und des Kassettenrekorders könnte eine neue Konstruktion und ein neues Tape entwickelt werden, woraufhin mehrstimmige Musikstücke mit dem Kassettenrekorder abgespielt werden könnten.

Das Ziel, einen funktionstüchtigen Kassettenrekorder zu konstruieren, ist erreicht und es ist somit gelungen den Kassettenrekorder wieder aufleben zu lassen. Auch wenn in der Praxis durch die Störgeräusche einige Einschränkungen existieren, ist es möglich mit dem Kassettenrekorder Musik zu hören.

## LITERATURVERZEICHNIS

[1] NIHILIANTH: *Lego NXT Music Player (MATLAB turntable)*. Internet. https://www.youtube.com/watch?v=6J-YEVI1Kms. Version: März 2018, Abruf: 02.03.2018

[2] SANDLER, Anna: *How to Make Lego NXT to Play a Sound*. Internet. http://www.robotappstore.com/Knowledge-Base/How-to-Make-Lego-NXT-to-Play-a-Sound/35.html. Version: Februar 2018, Abruf: 20.02.2018

[3] BUCKDAHN, Tobias: *Lego-Vokabular*. Internet. https://www.brickup.de/vokabular. Version: März 2018, Abruf: 01.03.2018

[4] FRITZ, Emonts: *Posegga Hans: Die Maus - Titelmusik der Sendung mit der Maus*. Internet. https://www.musikalienhandel.de/noten/klavier-(4ms)-(klav-(4ms))/die-maus-titelmusik-der-sendung-mit-der-maus--ED+8491.htm. Version: Februar 2018, Abruf: 23.02.2018

[5] REEG, Thommy: *Posegga Hans: Die Maus - Titelmusik der Sendung mit der Maus*. Internet. https://www.stretta-music.com/hans-die-maus-titelmelodie-aus-der-sendung-mit-der-maus-nr-552464.html. Version: Februar 2018, Abruf: 20.02.2018

[6] HIRTE, Patrick: *Matthias Raue: Löwenzahn Thema*. Internet. https://musescore.com/user/1651211/scores/4113256. Version: Februar 2018, Abruf: 22.02.2018

# Kartographie mit Lego-Mindstorms

Fabian Schimke, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Im Rahmen des Lego-Mindstorms Projekts wurde ein Roboter zur Kartographierung gebaut. Der mit Hilfe von einem Ultraschallsensors Entfernungen von sich zu Wänden oder Gegenständen um sich herum misst.Durch das Anfahren mehrerer Punkte, von denen aus gemessen werden soll, soll der Roboter auch komplexe Räume und Umgebungen aufzeichnen können. Aus den gemessenen Entfernungen soll eine zweidimensionale Karte erstellt werden. Der Roboter soll autonom von externen Ortungsdiensten(z.B. GPS) arbeiten können um viele Anwendungsfälle abdecken zu können. Das erstellen einer Karte kann zum Beispiel Anwendung finden in Räumen und Umgebungen bei denen keine Karte vorhanden ist oder durch ein Ereignis wie Erdbeben keine Aktuelle Karte/Grundriss vorhanden ist. Dies soll bei der Untersuchung oder Einsatzplanung in Höhlen oder eingestürzten Gebäuden helfen.**

***Schlagwörter*—Kartographie, Koordinatentransformation, Lego-Mindstorms, Ortungssystem,Roboter , Ultraschall**

## I. Einleitung

In vielen fällen liegt weder eine Karte noch Lageplan geschweige denn ein 3D-Modell vor. Sei es das diese verloren gegangen, nie erstellt worden oder durch Ereignisse wie Erdbeben nicht mehr aktuell sind. Durch das Erstellen dieser sollen Rettungskräfte und Forschungsgruppen unterstützt werden und vor eventuellen Gefahren(z.B. Nachbeben) geschützt werden. Die Aufzeichnung von Räumen oder Erdformationen kann in vielen Bereichen hilfreich sein. So können erstellte Lagepläne von eingestürzten Gebäuden die Rettungskräfte unterstützen und deren Vorbereitung für einen einstieg erleichtern indem sie schon vorher die passende Ausrüstung mit sich führen können. Des weiteren kann die automatisierte Kartographie nicht nur bei Rettungsaktionen unterstützen sondern auch beim Erfassen von Gebäuden die noch intakt sind, bei denen jedoch Pläne verloren gegangen sind oder nie erstellt worden sind,hilfreich sein. Dies kann von neu Bauten als auch von historischer Architektur der Fall sein. Somit könnten auch historische Bauten, mit Blick auf neue Technologien wie Augmented Reality oder Virtual Reality, für jeden erlebbar und untersuchbar gemacht werden, falls diese Zerfallen oder schwer zu erreichen sind. Die Einsatz Möglichkeiten sin vielfältig, um den Roboter zur Kartographie unabhängiger einsetzen zu können und soll er auf externe Ortungssysteme(GPS, Galileo, Erdmagnetfeld, u.Ä.) verzichten können da nicht jeder Anwendungsfall und/oder Ort den den Empfang dieser ermöglichen kann oder beim Verbindungsabriss unvorhersehbar reagieren könnte. Um dies zu ermöglichen muss der Roboter seine zurückgelegte strecke aufzeichnen oder sich an markanten Punkten orientieren und wieder erkennen.

DOI: 10.24352/UB.OVGU-2018-042 

## II. Vorbetrachtungen

### A. Abbildung und Verzerrung

Bei dem Erstellen von Karten oder Plänen, wie sie uns meisten begegnen, werden Vereinfachungen und gewisse annahmen gemacht die einem oft nicht bewusst sind. Im Rahmen dieser Arbeit werden nur 'kleine' Räume und Umgebungen beachtet die sich in einer Ebene befinden. Wird der Raum jedoch größer müssen weitere Faktoren in Betracht gezogen werden und auch gemessen werden. Dies spielt in der Nachbearbeitung und Darstellung der aufgenommenen Messwerte eine Rolle. Als eine der bekanntesten und am stärksten aus geprägte Verzerrungen sei hier die Mercator-Projektion genannt. Die den Globus in eine Rechteck projiziert und eine Winkel- aber nicht Flächentreue Karte umwandelt.

### B. Höhlenforschung

Um einen unbekannten Raum zu vermessen wurde sich am Beispiele der Höhlen Vermessung orientiert da diese grundsätzlich für unbekannten Räume genutzt wird. Hier gibt es verschiedene Ansätze für unterschiedliche Höhlenabschnitte auf zu zeichnen um sie später als Karte wiedergegeben zu können. Die Auf der Abbildung 1 dargestellten Methoden sind für die manuelle Vermessung gedacht in Kombination mit Maßband und Winkelmesser. So kann für die Vermessung mit dem Roboter weit mehr Messungen vorgenommen werden, da die einzelnen Messungen durch die Nutzung von Ultraschall schneller vorgenommen werden können als mit einem Maßband, bei dem die Strecke zurück gelegt werden muss. Durch den Aufbau des Roboters ist die Sternvermessung naheliegend.Die Sternvermessung ist für einen größeren Hohlraum gedacht bei der manuellen Vermessung und wird daher selten für für schmalere Gänge genutzt siehe 'Zick-Zack-Vermessung'.

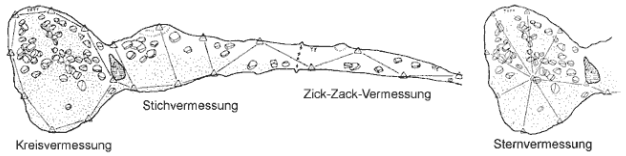



Abbildung 1. Vermessungsmethoden der Höhlenvermessung

### C. Sensorwahl und Positionierung

Um die Entfernung zumessen wurde der Ultrasonic-Sensor von Lego NXT genutzt. Dieser sendet einen Ultraschallimpuls aus und errechnet durch die Laufzeit der Reflexion die Entfernung der Oberfläche von der die Ultraschallwele reflektiert wurde. Der Messbereich reicht von 0 bis 255cm.

Um mehrere Messungen machen zu können ohne den Roboter selbst zu bewegen wurde der Sensor drehbar mittig über dem Roboter angebracht. Somit kann Rund um den Roboter die Umgebung vermessen werden ohne das sich der Roboter relativ zum Raum bewegt. Diese Position wurde auch gewählt um die Sternvermessung zu begünstigen. Würde der Sensor

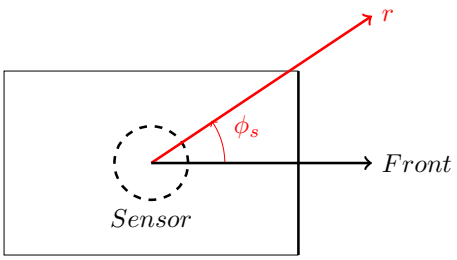



Abbildung 2. Schematische Darstellung des Roboters und position gemessener Werte

beispielsweise fest am Roboter angebracht sodass der Roboter sich um seine eigene Achse drehen müsste, müsste der genaue Drehpunkt ermittelt werden. Wohingegen der drehbar gelagerte Sensor immer auf der gleichen Bahn sich um den Roboter dreht und der systematische Fehler durch eine einzelne Messung ermittelt werden kann. So werden nun jeweils die Entfernungen und der dazu gehörige Winkel zwischen Fahrtrichtung und Sensorausrichtung aufgezeichnet. Die Werte Paare die sich bei der Entfernungsmessung ergeben können direkt als Polarkoordinaten dargestellt werden. Die für die leichtere Darstellung als Karte in der Nachbearbeitung zu kartesischen Koordinaten transformiert werden.

$$\begin{bmatrix} x \\ y \end{bmatrix} = r \cdot \begin{bmatrix} cos(\phi_s) \\ sin(\phi_s) \end{bmatrix} \tag{1}$$

Der zuvor genannte systematische Fehler entspricht dem Radius der Kreisbahn $r_k$ auf der sich der Sensor um den Roboter bewegt.

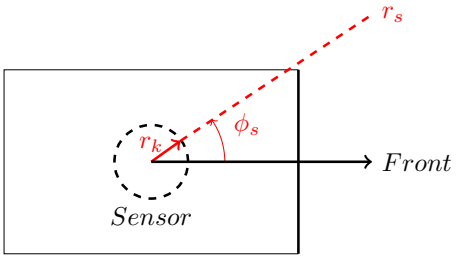



Abbildung 3. Darstellung des systematischen Fehlers bei der Messung der Entfernung durch die Drehung des Sensors auf einer Kreibahn

$$\begin{bmatrix} x \\ y \end{bmatrix} = r_s \cdot \begin{bmatrix} cos(\phi_s) \\ sin(\phi_s) \end{bmatrix} + r_k \cdot \begin{bmatrix} cos(\phi_k) \\ sin(\phi_k) \end{bmatrix} \tag{2}$$

Darüber hinaus kann durch die Anordnung des Sensors die gemessene Entfernung als die Normale auf der Kreisbahn des Sensors betrachtet werden womit

$$\phi_s = \phi_k$$

ist. Wodurch die Transformation (2) zu

$$\begin{bmatrix} x \\ y \end{bmatrix} = (r + r_k) \cdot \begin{bmatrix} cos(\phi_s) \\ sin(\phi_s) \end{bmatrix} \tag{3}$$

vereinfacht werden kann. Durch diese Transformation wurden nun aus polar Koordinaten kartesische Koordinaten. Ein weiterer Sensor der verwendet wurde ist der Gyroscopic Sensor von HiTec der die Winkelgeschwindigkeit messen kann. Womit beim Anfahren verschiedener Messpunkte die Verdrehung des Roboter zu seiner Vorherigen Position gemessen werden kann.

### *D. Koordinaten Transformation*

Da der Raum von mehreren Punkten aus vermessen wird, muss zwischen einzelnen Messreihen unterschieden werden. Alle Messungen von einem Punkt aus werden als lokale Koordinaten bezeichnet, sie beschreiben die unmittelbare Umgebung um den Roboter. Um die Darstellung zu verbessern oder auch weitere Räume zuerkennen müssen mehre Punkte angefahren von denen aus lokale Koordinaten auf genommen werden. Diese, lokalen polar Koordinaten, können mit (1) in der Nachbearbeitung in kartesische Koordinaten transformiert werden. Zunächst in lokale kartesische Koordinaten danach in globale kartesische Koordinaten. Global beschreibt hierbei das räumliche Verhältnis der einzelnen lokalen Messreihen zueinander. Um die einzelnen lokalen kartesischen Koordinaten zusammenzufügen zu globalen kartesischen Koordinaten wird der Punkt von dem aus gemessen wurde, die Roboter Position, der einzelnen lokalen Messreihen als Translation interpretiert. So muss nun jede Messreihe um die neue Roboter Position verschoben werden. Um diese Rechenoperationen durch Matrixmultiplikation zu vereinfachen werden die lokalen Koordinaten um eine $z$ Koordinate erweitert. Die erweiterte $z$ Koordinate hat keinen Einfluss auf die Messwerte bei den Transformationen, sie hat den Betrag *1*, gleichzeitig ermöglicht sie die spätere Erweiterung auf eine 3. Dimension.

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{4}$$

Bei dieser Translation wird jedoch nicht die Verdrehung des Roboter beachtet die entstehen kann wenn ein neuer Messpunkt angefahren wird. Daher werden die neuen lokalen koordinaten beim umrechnen in globale Koordniaten ebenso mit einer Rotation um die vom Gyroscopic Sensor aufgenommene Drehung des Roboters erweitert.

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{5}$$

Die Translation(4) und Rotation(5) können zu (6) zusammen gefasst werden.

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & t_x \\ -\sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{6}$$

### E. Odemetrie

Im Rahmen des Blockseminars wurde eine Aufgabe bearbeitet die sich mit Odemetrie befasst hat und dem daraus entstehenden Fehler. So ist auch hier, ähnlich wie unter 'Abbildung und Verzerrung', zu erwähnen das durch Übersetzungen, Spiel und Schlupf geringe Fehler bei der Ausrichtung des Roboters als auch der Sensor Positionierung auftreten können und somit eine hohe Genauigkeit nicht gegeben ist. So wurden experimentell ermittelt das 20000 MotorCounts den Sensor auf seiner Kreisbahn einmal um den Roboter fahren, was theoretisch zu einer Auflösung von

$$\frac{360\,\mathrm{deg}}{20000 MotorCount} = 0,018 \frac{\mathrm{deg}}{MotorCount}$$

führt, die aber im Angesicht der verwendeten Bauteile nicht realistisch erscheint. Ebenso ist anzunehmen das es nicht genau 20000 MotorCounts sind.

## III. Hauptteil

Einzelne Funktionsgruppen wie Roboter, Messwertaufnahme, Koordinaten Transformation und Motorenansteuerungen wurden Schrittweise programmiert und separat getestet und schrittweise zusammengeführt.

### A. Konstruktion

Das Roboter Fahrgestell wurde zu großen Teilen dem Standard Lego-Mindstorms NXT Set entnommen. Lediglich der dritte Motor wurde rückwärtig angebracht um eine hinzugefügte Sensorplattform die sich oben am Roboter befindet anzutreiben. Der antrieb ist durch Ketten realisiert die unabhängig von einander angesteuert werden können. Der Gesamt Aufbau wurde möglichst flach und zentralisiert gehalten damit der Ultrasonicsensor sich frei und ohne Hindernisse über den Roboter drehen kann. Ebenso wurde der Gyroscopic Sensor von HiTec mittig vom Roboter angebracht um die Abweichung zwischen dem gemessenen Drehwinkel und dem Drehwinkel des Roboters so gering wie möglich zuhalten. Die Sensoreinheit zur Entfernungsmessung besteht aus einer Drehscheibe mit Kabeldurchführung damit das Sensorkabel nicht vor den Empfänger geraten kann.

### B. Entfernungsmessung

Die Drehscheibe mit dem Ultrasonic Sensor wird über eine Schnecke angetrieben was zu einer langen Zeit für eine 360° Drehung führt (*22-23 Sekunden*), die es jedoch ermöglichte durchgehend Messungen vorzunehmen. In ersten Konstruktionen, bei denen der Sensor, je nach vorgegebener Auflösung, sich ein Stück gedreht und dann eine Messung vorgenommen hat, musste Anfahren und Abbremsen der Sensorhalterung abgewartet werden, da diese zu einer Verfälschung des am Motor gemessenen Winkels und dem tatsächlichen Winkel geführt hat. Die ersten Konstruktionen waren noch nicht durch eine Schnecke angetrieben. Der Antrieb über eine Schnecke erwies sich als hilfreich, da sich so die Sensoreinheit nicht leicht durch Außeneinwirkung verdrehen lässt. Die korrekte Zuordnung der Winkel und Entfernungen wurde experimentell überprüft durch den Vergleich von dargestellten polar Koordinaten und Gegenständen in der Messumgebung.

### C. Verifizierung von Algorithmen

Für die Nachbearbeitung der Messwerte ist es wichtig das jeder einzelne Algorithmus für sich funktioniert und Messwerte nicht verfälscht oder unbrauchbar gemacht werden. Um die Transformationen zu verifizieren wurde der Algorithmus mit Daten aus der vorherigen Entfernungsmessung gespeist und mit festen Rotationen und Translation transformiert. Ebenso wurde eine Array erstellt das sich nach jeder Messwertaufnahme erweitert um neue Messwerte nach der Transformationen von lokalen in globale Koordinaten abzuspeichern.

### D. Transformation in Verbindung mit Roboterbewegung

Die Aufzeichnung einer Messreihe, sprich einem Satz lokaler Koordinaten, ist in sich kohärent. Die erste Messreihe lokaler Koordinaten entspricht den gleichen in globalen Koordinaten, da der Roboter sich noch nicht bewegt hat und keine Messungen zuvor eingetragen hat mit denen er sie in Relation bringen muss. Wird nun ein zweiter Satz lokaler Koordinaten hinzugefügt muss die neue position des Roboters genau bestimmt werden. Hier für wurden zwei Möglichkeiten getestet. Bei der ersten Methode wurde die zurück gelegte Strecke bei vorgegebenen MotorCount vermessen und mehrmals wiederholt und gemittelt. Bei dieser Methode wird ein zufälliger Fehler erzeugt. Bei gleichbleibenden MotorCount Variierte die zurück gelegte Strecke um einen bis zwei Centimeter auf 14cm Strecke. Die zweite Methode nutzt einen zweiten Ultrasonic Sensor der in die Fahrrichtung zeigt und somit die zurückgelegte Strecke ermittelt durch ein Objekt das sich in der Fahrtrichtung befindet. Diese Methode hat eine offensichtliche Einschränkung, wenn sich kein Objekt oder Wand in der Fahrt Richtung befindet kann die zurück gelegte strecke nicht ermittelt werden. Bei versuchen ergaben sich durch abweichende Messungen um Teil ein doppelt so hohe zurückgelegte strecke wie tatsächlich zurückgelegte Strecke. Auffällig waren das die stark Abweichenden Messwerte beim anfahren und abbremsen auf traten, da der Roboter zum Nicken neigt.

## IV. Erpobung

Um die Raumvermessung zu erproben und Fehler zu erkennen wurde eine Testumgebung aufgebaut. Diese beinhaltete eine lange Wand und auf der anderen Seite zufällig aufgestellte Rechteckige Kartons. Der Test lauf beinhaltete die Aufnahme von lokalen Koordinaten, ein Stück grade ausfahren und wieder Koordinaten aufnehmen und dies vier weitere male. Auffällig waren Messwerte die unabhängig von der Umgebung in einem Kreis von 20cm entfernung um den Roboter waren. Durch eine Anpassung der Sensoraufnahme konnten diese Fehler unterdrückt werden. Dabei wurde $r_k$ vergrößert. Bei weiteren Erprobungen konnte aus den Messdaten kann eine Punktwolke erzeugt werden. Die sich ergebene Punktwolke lässt den vermessenen Raum erahnen jedoch weicht die Punktwolke stark von dem tatsächlich vorliegendem Raum ab. So gibt es Verzerrungen und 'wandernde' und 'plötzlich' auftauchende Oberflächen.

### A. *Verzerrung*

Am deutlichsten sind die Verzerrungen an geraden Wänden zu erkennen die als Bögen dargestellt werden. Die Ursache für diese Verzerrungen ist das Messverfahren mit Ultraschall. Die Abstrahlung von Ultraschall erfolgt nicht Linienförmig wie Abbildung 2 nahelegt. Die Ultraschallwellen breiten sich vom Sensor in einem Kegel aus. So kommt es dazu das der Ultraschallimpuls, der ausgesendet wurde, an einer Oberfläche reflektiert wird auf die der Sensor nicht ausgerichtet ist. Die Entfernung die gemessen wird stimmt jedoch nicht der ihr zu geordnete Winkel.

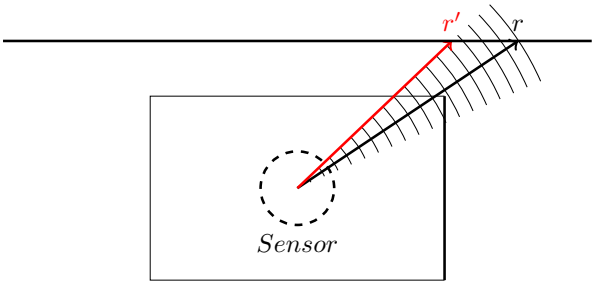



Abbildung 4. Aliasing von Oberflächen am Beispiel einer geraden Wand

### B. *'Wandernde' Oberflächen*

Die 'wandernde' Oberflächen bzw. Messwerte stimmten mit der Position einer in leichte Plastikfolie gehüllte Mülltonne am rand. So scheint die Messung mit Ultraschall nicht für leichte und bewegliche Oberflächen geeignet. Um dem auf den Grund zu gehen müssen noch weitere Oberflächen und Vergleichsmessungen vorgenommen werden.

### C. *'plötzlich' auftauchende Oberflächen*

Während den Messungen wurde ein Karton in den ersten Positionen nicht erkannt, der Sensor die maximale Entfernung ausgab, jedoch in einer weiteren Messung den Karton direkt neben sich anzeigte. So ist hier der Winkel in dem die Ultraschallwellen auf die Oberflächen treffen die Ursache. Ist der Winkel unter dem die Ultraschallwelle auf die Oberfläche trifft zu spitz reicht die Amplitude, der zum Sensor zurück geworfenen Welle, nicht aus um die Sensorschaltung zu Triggern.

### D. *Positionsbestimmung*

Beide zuvor erwähnten Möglichkeiten um die zurückgelegten strecken zu messen wurden getestet ebenso der Gyroscopic Sensor um die Drehungen des Roboters aufzunehmen. Die gemittelte zurückgelegte Strecke hat wie schon erwähnt von Messung zu Messung eine Streuung die immer größer wird. Die gemessene zurückgelegte Strecke, mit dem zusätzlichen Sensor, machte die gesamte Transformation unbrauchbar, ein Raum konnte nicht erkannt werden. Das selbe gilt auch für den Gyroscopic Sensor.

## V. Zusammenfassung und Fazit

Es ist möglich mit dem Lego-Mindstorm NXT einen Roboter zur Kartographierung zu konstruieren. Die Ergebnisse aus dem Projektseminar eignen sich noch nicht um eine Karte oder Lageplan zu erstellen. Das Problem besteht nicht in der Messwertaufnahme oder der Transformation. Hier stellt die Interpretation der Messwerte die größere und komplexere Aufgabe dar. Um die zurückgelegte Strecke und auch den Drehwinkel des Roboters besser zu erfassen können weitere Algorithmen entwickelt werden die auch die Umgebung mit einbeziehen um zum Beispiel Verzerrungen zurück zurechnen oder sich and markanten Punkten zu orientieren.

## Quellen

Abbildung 1 :[On Station, George Dasher, NSS 1994, S. 45]

# Kartographie mit Lego-Mindstorms

Jacob Rühe, Elektro- und Informationstechnik, 218349

***Zusammenfassung*—Im Rahmen des Lego-Mindstorms-Projekts war es das Ziel einen Kartographierungsroboter, aus einem Legobausatz, zu entwickeln. Dazu wurden verschiedene Transformationsmatrizen verwendet. Diese werden dazu benutzt, die Messwerte vom polaren Koordinatensystem in das globale kartesische Koordinatensystem umzuwandeln. Am Ende wird ein ungefährer Grundriss der Umgebung erstellt. Dieser Roboter kann in der Höhlenforschung sowie zur Rettung von Menschen in eingestürzten Gebäuden benutzt werden.**

***Schlagwörter*—Lego-Mindstorms, Ultraschall, Kartographie, Koordinatentransformation**

## I. Einleitung

In der heutigen Zeit wird alles automatisiert, so auch die Kartographierung. Wo früher alles mit Hand aufgezeichnet wurde, können heute Roboter eine Grundrisskarte erstellen. Die Intention meines Partners Fabian Schimke und mir war es, einen Kartographierungsroboter zu konstruieren, welcher genau dies bewerkstelligen kann. Dieser wäre zum Beispiel in der Höhlenforschung einsetzbar. Der Roboter könnte in eine unbekannte, gefährliche Höhlenöffnung gesetzt werden und so den Forschern erste Daten über das Innere der Höhle liefern, ohne dass die Forscher sich selbst in Gefahr begeben. Ein weiteres Anwendungsgebiet wäre die Rettung von Menschen in eingestürzten Gebäuden, in Erdbebengebieten. Das Fahrzeug könnte in enge und gefährliche Spalten gelangen und so einen Grundriss des Raumes erstellen um verletzte Menschen zu lokalisieren. So wären die Helfer in der Lage einen Rettungsplan zu entwerfen. Um dies zu schaffen, müsste der Roboter in der Lage sein, Wände bzw. Gegenstände zu erkennen und diesen selbständig ausweichen. Des Weiteren müsste der Roboter Menschen erkennen und diese dann bei der Kartographierung des Gebietes auf der Karte erkenntlich machen. Außerdem müsste das Fahrzeug in verschiedenen Gelände fahren und überall seine Messwerte aufnehmen können. Die Messwerte des Roboters müssten so umgewandelt werden, dass selbst bei translatorischen Bewegungen oder Rotation des Fahrzeuges die Messwerte immer zum Anfang zurückgerechnet werden. Durch zusätzliche Transformation der Messwerte würde dann zum Schluss eine ungefähre Grundrisskarte erstellt werden.

## II. Vorbetrachtungen

Es gibt ein paar Geräte und Techniken, womit man bestimmte Gebiete kartographieren kann. Meistens ist der Zeitaufwand dieser Geräte hoch , da viele Messungen noch von Hand getätigt werden.

### A. Höhlenforschung

In der Höhlenforschung müssen die Forscher meist alles manuell vermessen. Das heißt, es wird jeder einzelne Abstand oder Ecke mit Maßband und Winkelmesser vermessen. Die Daten müssen dann in einer ungefähren Skizze übertragen werden. Erst später werden dann die Messwerte und die grobe Skizze zusammen, zu einer genaueren Skizze zusammengefügt. Dies ist im großen und ganzen sehr zeitaufwendig. Daher liegt es nahe einen Roboter zu konstruieren, welcher diese Vermessungen mit Hilfe von Ultraschallmessungen schneller erledigen kann. Die Ultraschallmesswerte können dann anschließend in eine Karte umgewandelt werden. Zum Vermessen gibt es mehrere Messmethoden. Durch den Aufbau des Roboters empfiehlt es sich die Sternvermessungsmethode anzuwenden, welche in Abbildung 1 veranschaulicht ist

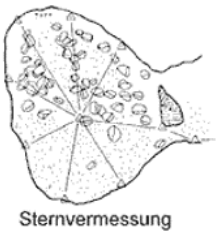



Abbildung 1. Sternvermessung

Die Sternvermessung wird bei manuellen Vermessungen von großen Hohlräumen eingesetzt. Da sich bei unserem Roboter der Ultraschallsensor, welcher die Messwerte aufnimmt, oben am Fahrzeug befindet und sich um 360 Grad dreht, ist die Anwendung in großen Hohlräumen mit der Sternvermessung naheliegend. Bei der Sternvermessung wird ein zentraler Punkt im Raum ausgemessen, wo sich der Messroboter oder das Messgerät befindet. Anschließend wir in alle Richtungen gemessen, wobei die Ultraschallmessung wesentlich schneller ist. Aus den Messwerten ergibt sich dann eine ungefähre Grundrisskarte der Höhle. Bei schmaleren Gängen wird die 'Zick-Zack-Vermessung' verwendet, welche in Abbildung 2 unter anderem abgebildet ist.

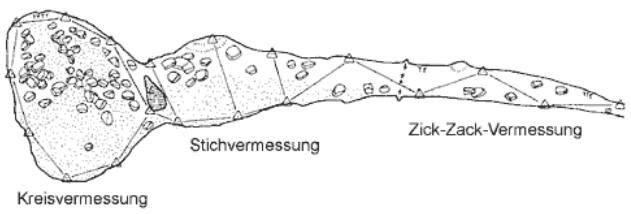



Abbildung 2. Vermessungsmethoden der Höhlenvermessung

Bei dieser Vermessungsmethode werden immer abwechselnd an den Wänden bei markanten Stellen, wie zum Beispiel eine hervorstehende Kante, Farbpunkte gesetzt. Anschließend werden anhand dieser Punkte Vermessungen angestellt.

DOI:10.24352/UB.OVGU-2018-043 

### B. Gleichung für Rotation und Translation

Damit die Messwerte am Ende, trotz Rotationsbewegungen und Translationsbewegungen, an der Ausgangsstellen zurückgerechnet werden und so eine ungefähre Grundrisskarte erkennen lassen, benötigt man folgende Formel.

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & t_x \\ -\sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (1)$$

In dieser Formel befindet sich die Formel der Rotations Koordinaten

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (2)$$

sowie die Formel der Translation Koordinaten

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (3)$$

zusammengefasst. Mit Hilfe dieser zusammengefassten Formel werden die Messwerte, auch bei einer Drehung oder vorwärts Bewegung des Roboters, zu dem Ausgangspunkt des Roboters zurückgerechnet. So entsteht schließlich eine ungefähre Grundrisskarte.

### C. Odemetrie

Odemetrie ist eine Methode, um anhand der Motoren eines Fahrzeugs, die ungefähre Position dieses Fahrzeuges im Raum zu bestimmen. In unserem Projekt verwendeten wir unter anderem diese Methode. Da wir mit dem uns zu Verfügung stehenden Materialien keine allzu hohe Genauigkeit erzielen konnten, weichen die Daten vom realistischen ab. Die Methode der Odemetrie wendeten wir beim oberen Sensor an, welcher um 360 Grad drehbar ist. Durch experimentelles Probieren fanden wir heraus, dass 20000 MotorCounts nötig sind, um diesen Sensor einmal um die eigenen Achse drehen zu können. Dies führt theoretisch zu einer Auflösung von

$$\frac{360deg}{20000MotorCount} = 0.018\frac{deg}{MotorCount} \quad (4)$$

Allerdings ist weder die Auflösung noch der MotorCount realistisch, wenn man bedenkt welche Bauteile verwendet wurden.

## III. Hauptteil

Unsere Idee war es einen Roboter zu konstruieren, der in einer unbekannten Umgebung eine Grundrisskarte dieses Gebietes erstellen kann. Realisiert haben wir dies mit dem Fahrzeug, welches sie in Abbildung 6 sehen können. Diese Abbildung finden sie im Anhang wieder.

Zu sehen ist ein Kettenfahrzeug, welches von zwei unabhängigen Motoren betrieben wird. Dadurch ist der Roboter in der Lage sich um seine eigene Achse zu drehen, um so seinen Kurs zu ändern. Des Weiteren haben wir dieses Projekt mit zwei Ultraschallsensoren und einem Gyroskopicsensor verwirklicht. Der Gyroskopicsensor befindet sich zentral an der Unterseite des Fahrzeugs. Dieser Sensor ist in der Lage zu messen, um wie viel Grad sich das Fahrzeug um seine eigene Achse dreht. Damit diese Messung so genau wie möglich ist, muss sich der Sensor möglichst zentriert im Roboter befinden. Die Gradzahl wird benötigt, um die Messwertpunkte bei einer Drehung des Fahrzeugs zurück auf die Anfangsmessung zu rechnen. Somit ergibt sich dann eine nicht verdrehte Darstellung. Der erste Ultraschallsensor, an der Front des Fahrzeugs angebracht, dient zur Messung der Entfernung zu Wänden oder Gegenständen. Bei einem bestimmten Mindestabstand zu einer Wand oder einem Gegenstand ist das Fahrzeug in der Lage diesen auszuweichen, indem es sich um eine festgelegte Gradzahl um seine eigene Achse dreht. Der zweite Ultraschallsensor befindet sich oben auf dem Roboter. Dieser ist so montiert, dass er sich um 360 Grad drehen kann. Diese Drehung benötigt man, um in alle Richtungen Messwerte aufnehmen zu können und so eine Karte vom Raum zu erstellen. Außerdem befindet sich der Ultraschallsensor am Ende einer Schiene, wodurch die Aufnahme von Störgeräuschen vom Fahrzeug selber vermindert wird. Die Messwerte werden vom zweiten Ultraschallsensor so umgewandelt, dass am Ende eine Darstellung bzw. ein ungefährer Grundriss vom Raum erkennbar wird. Zuerst werden die Messwerte in polaren Koordinaten wiedergegeben.

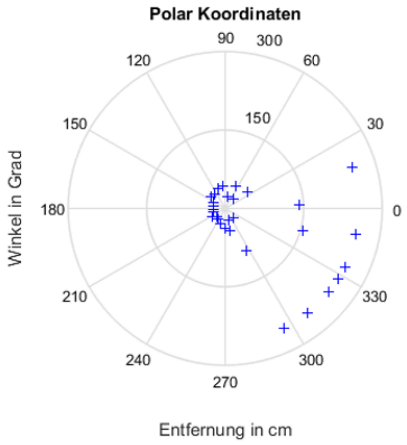



Abbildung 3. Polare Koordinaten

Diese werden mithilfe von Matrixmultiplikation in lokalen kartesischen Koordinaten transformiert. Um die Messwerte von polaren Koordinaten in kartesische Koordinaten umzuwandeln, benötigt man folgende Formel

$$\begin{bmatrix} x \\ y \end{bmatrix} = r \cdot \begin{bmatrix} cos(\phi) \\ sin(\phi \end{bmatrix} \quad (5)$$

Der obere Ultraschallsensor befindet sich auf eine Kreisbahn und nicht genau im Zentrum des Roboters. Daher muss die Formel 5 noch um den Abstand des Sensors zum Fahrzeugmittelpunkt und den Winkel ergänzt werden. Dadurch entsteht folgende Formel

$$\begin{bmatrix} x \\ y \end{bmatrix} = r \cdot \begin{bmatrix} cos(\phi) \\ sin(\phi) \end{bmatrix} + r_s \cdot \begin{bmatrix} cos(\phi) \\ sin(\phi) \end{bmatrix} \quad (6)$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = (r + r_s) \cdot \begin{bmatrix} cos(\phi) \\ sin(\phi) \end{bmatrix} \quad (7)$$

Mit Hilfe der Formel 7 werden die polaren Koordinaten in kartesische Koordinaten umgewandelt.

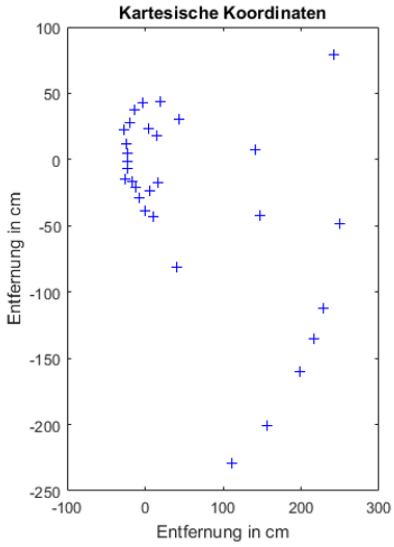



Abbildung 4. Kartesische Koordinaten

Der Unterschied der lokalen kartesischen Koordinaten zu den polaren Koordinaten ist der, dass die Messwerte der lokalen kartesischen Koordinaten den Roboter als Bezugspunkt haben. Da sich nun der Roboter allerdings nach jeder Messung um 13 cm nach vorne bewegt und dadurch die Messwerte am Ende keinen Raum erkennen lassen würden, muss eine erneute Transformation der lokalen kartesischen Koordinaten in globale kartesische Koordinaten erfolgen. Bei den globalen kartesischen Koordinaten werden die Messwerte immer wieder zur Anfangsmessung zurückgerechnet und die Punkte bilden jeweils untereinander einen Bezug zueinander. Damit die Messwerte bei einer Translation des Roboters zurückgerechnet werden, benötigt man folgende Formel

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (8)$$

So bleibt der Roboter, der bei der Darstellung im Mittelpunkt sich befindet, auch dort und verschiebt nicht seinen Mittelpunkt immer wieder bei jeder neuen translatorischen Bewegung mit, wie bei den lokalen kartesischen Koordinaten. Des Weiteren werden mithilfe folgender Formel die Messwerte bei einer Rotation übereinandergelegt.

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (9)$$

Die beiden Formeln 8 und 9 werden der Formel 10 zusammengefasst.

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & t_x \\ -\sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (10)$$

So entsteht am Ende aus den globalen kartesischen Koordinaten eine ungefähre Darstellung bzw. Grundriss des Gebietes.

## IV. Ergebnisdiskussion

Das Ergebnis unserer Arbeit ist ein Roboter, welcher einen ungefähren Grundriss seiner Umgebung erstellen kann, sowie Hindernisse erkennt und ihnen selbständig ausweicht. Allerdings ist dieses Fahrzeug nur für eine gerade Ebene geschaffen. Des Weitern ist die Reflexion der Ultraschallwellen an schrägen Objekten ein Problem. Dadurch werden die Ultraschallwellen nicht gerade zum Ultraschallsensor zurückgeworfen. Somit erkennt der Roboter dieses Objekt nicht. Ein weiteres Problem besteht in der Erkennung von Ecken. Durch die Nichterkennung der Ecken wird der Raum oval förmig wiedergegeben. Außerdem werden die Sensorkennlinien statisch verzerrt.

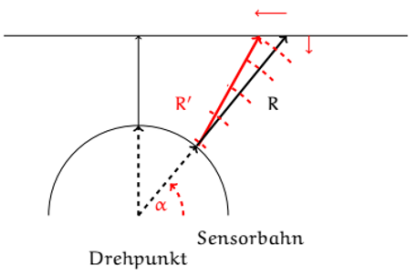



Abbildung 5. Verzerrung von Linien

Das heißt, dadurch das der Ultraschallsensor, welcher die Messwerte aufnimmt, nicht genau im Mittelpunkt des Roboters befestigt ist, entsteht ein kleiner Abstand, welcher bei den Messungen nicht berücksichtigt wird. Auch treffen die Ultraschallwellen zuerst auf das Objekt, da die Wellen sich mit zunehmender Entfernung immer weiter ausbreiten. Dadurch wird dann ein Messwert gemessen, der allerdings weiter rechts oder links von der eigentlichen Sensorkennlinie ist und somit die Ultraschallwellen früher zurückwirft. Die Sensorkennlinie ist der gerade Abstand vom Sensor bis hin zum Gegenstand.

## V. Zusammenfassung und Fazit

Zusammenfassend lässt sich sagen, dass wir einen Kartographierungsroboter gebaut haben, welcher in der Lage ist eine ungefähre Grundrisskarte seiner Umgebung zu erstellen. Dazu werden mit Hilfe eines um 360 Grad drehbaren Ultraschallsensors Messwerte aufgenommen. Diese werden zuerst in einem polaren Koordinatensystem dargestellt und mit Transformationsmatrizen erst in ein lokales kartesisches Koordinatensystem und dann in ein globales kartesisches Koordinatensystem umgewandelt. Durch zu Hilfenahme eines Gyroskopicsensors und eines weiteren Ultraschallsensors werden die Messwerte zurück zum Ausgangspunkt gerechnet.

Dadurch entsteht am Ende, auch wenn sich der Roboter bewegt, keine große verzerrte Grundrisskarte. Allerdings könnte man die Qualität verbessern, indem man mehrere Punkte aufnimmt. Des Weiteren könnten Gegenstände sowie Ecken genauer abgefahren werden zu der besseren Erkennung dieser. Auch würde ein 3. Ultraschallsensor, welcher ebenfalls um 360 Grad drehbar ist und Messwerte aufnimmt, helfen, um bei einem Vergleich der beiden drehbaren Ultraschallsensoren genauere Daten zu erzielen. Zum anderen wäre die Befestigung eines Wärmesensors eine Verbesserung, um damit Menschen zu erkennen. Eine weitere Überlegung wäre die Anwendung einer Heatmap, damit die Darstellung noch deutlicher wird. Bei der Heatmap werden bei einer bestimmten Dichte von Punkten diese mit einer Farbe dargestellt. So könnte man schlussendlich auch eine dreidimensionale Darstellung anstreben.

## VI. Anhang

Abbildung 6. Kartographierungsroboter

## VII. Literaturverzeichnis

(1) Abbildung 1 und 2: Einführung in die Höhlenvermessung, Autor: Jochen Hartig, DAV Höhlengruppe Frankfurt/Main, http://caverender.de/vermess/ vermess.htm

(2) Transformations matrix, Wikipedia, 22.03.2018, 10:02,

# Anfertigung eines Kehrroboters mithilfe von Lego Mindstorms und MatLab

Thomas Schreiber, Elektrotechnik und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Ziel des Projektes war es, einen Roboter aus Lego zu bauen und ihn anschließend mit MatLab zu programmieren. Dazu entschieden wir uns für den Bau eines Kehrroboters, der selbstständig Verschmutzungen beseitigen soll und gleichzeitig noch in der Lage ist, Hindernisse zu erkennen und auf diese durch ein Ausweichmanöver zu reagieren. Demenentsprechend sollte es sich auf Ketten bewegen und einen Ultraschallsensor zur Entfernungsmessung besitzen, darüberhinaus soll es auch eine Vorrichtung zum Aufsammeln der Verschmutzungen, zum Beispiel einen Staubwedel, besitzen.**

***Schlagwörter*—Haushaltsroboter, Kehrmaschine, Kehrroboter, Reinigungsroboter, Saugroboter, Staubsauger**

## I. Einleitung

Einst lediglich eine Zukunftsfantasie aus Science-Fiction Romanen, wo sie alle möglichen Aufgaben übernehmen, stehen sie heute immer mehr im Mittelpunkt der Gesellschaft: Die Rede ist von autonomen Maschinen, oder auch Roboter genannt. Eingesetzt in den unterschiedlichsten Bereichen, von Industrie, über den Handel bis hin zu den Privathaushalten, wo sie die lästigen Aufgaben übernehmen wie das Rasenmähen oder das Staubsaugen. In der Industrie werden sie hingegen hauptsächlich für Präzisionsaufgaben verwendet, die ein Mensch wesentlich ungenauer machen würde als der Roboter. Aber das ist für unser Projekt nur nebensächlich.

Ausschlaggebend für unser Projekt sind die Haushaltsroboter, insbesondere der Saugroboter, den wir in Form eines Kehrroboters während des Lego Mindstorms Seminars vom 11.02.2018 bis zum 23.02.2018, bauen und programmieren wollten. Dabei war unsere Idee, der Roboter fährt geradeaus und lässt vorneweg einen Staubwedel rotieren, der den vorliegenden Müll aufsammelt. Wenn er in der Nähe eines Hindernisses kommt, so sollte er die Möglichkeit kriegen, dieses zu erkennen und entsprechend durch ein Ausweichmanöver darauf zu reagieren. Und zu guter letzt sollte er sich auf Ketten bewegen, damit er in der Lage ist, sich auf der Stelle zu drehen, was unnötige Umwege ersparen würde. Als Versuchsmüll zogen wir kleinere Legoteile in Betracht, da diese keinen anderen Abfall produzieren, wie in etwa Flecken auf den Tischen des Computerlabors.

Zur Programmierung des Roboters stellt die RWTH Aachen ein Toolkit zur Verfügung, was es ermöglicht, den intelligenten Stein von Lego NXT über Matlab anzusteuern, was eine Vielzahl an neuen Möglichkeiten zum Nutzen von Lego NXT hervorbringt.

DOI: 10.24352/UB.OVGU-2018-044 

## II. Vorbetrachtungen

Egal ob für das Rasenmähen, zum Staubsaugen oder auch neuerdings mit Amazons Alexa zum Shoppen: Roboter dringen immer mehr in alle Lebensbereiche vor. Schon seit etlichen Jahren sind anfangsgenannte Roboter auf den Markt erhältlich und vereinfachen das Leben von vielen Personen, nicht zuletzt aus dem Grund, dass sie lästige Tätigkeit abnehmen, zu der man nie Lust hat oder auch eine Hilfe für Menschen, die gesundheitlich (z.B. durch eine körperliche Einschränkung) nicht in der Lage sind, bestimmte Tätigkeiten auszuführen.

Folgende zwei Apparate erwiesen sich dabei als äußerst hilfreich bei den Vorbetrachtungen zu unserem Roboter:

### A. Saugroboter

Einst nur fähig, grob zu reinigen und nur durch Kollision in der Lage auszuweichen, so sind sie heute standardmäßig mit einigen Sensoren zur Umgebungserkennung ausgrüstet. Die Rede ist von Staubsaugerroboter oder auch Saugroboter genannt. Selbst einfache Modelle sind heute ausgestattet mit allerlei Sensoren, um ein möglichst präzises Putzverhalten zu simulieren. Teurere Modelle sind unter anderem auch ausgestattet mit einer Vielzahl an anderen Funktionen wie etwa das Ablesen des Akkustandes, um bei Erschöpfung automatisch zur Ladestation zu fahren und sich dort anzudocken, um danach den Putzdienst fortsetzen zu können an der Stelle, wo sie aufgehört haben. Größentechnisch sind sie orientiert an den Abstand von Stuhlbeinen. [1]

Es mag zwar nun den Anschein besitzen, sie würden das Staubsaugen komplett ersetzen, dem ist aber nicht so. Zwar bieten sie neben den oben genannten Dingen noch weitere Vorteile, wie etwa die Tatsache, dass sie leicht verstaubar sind und wenig Strom verbrauchen, jedoch besitzen sie nicht annähernd die Saugkraft eines richtigen Staubsaugers und kommen mitunter auch nicht oder kaum in verwinkelte Ecken eines Raumes, was sich zum Beispiel dann bemerkbar macht, wenn der Raum viele Hindernisse wie etwa Möbel beinhaltet. [2]

### B. Kehrmaschine

Der grobe Aufbau und die Funktionsweise sind simple gehalten: Ein rotierender Bürstenkopf sammelt den vorliegenden Schmutz ein und transportiert es zu einem Sammenbehälter. Darunter gibt es dann die verschiedensten Arten von Kehrprinzipien, wovon ich nur eine kurz erläutern will, die für unseren Roboter am Interessantesten war: das Kehrschaufelprinzip. Hierbei rotiert eine vorliegende Bürstenrolle, um den Schmutz auf ein dahinterliegendes Kehrblech zu befördern. [3]

### C. Roboter im Privathaushalt

Nun noch zum Punkt, in welchen Bereichen können Roboter im Haushalt noch unterstützen? Gerade wurde schon erwähnt, sie unterstützen in Form von Staubsauger und Rasenmäher, aber es gibt noch andere, bekannte Einsatzorte für Roboter im Haushalt: Beispielsweise zur Überwachung des Hauses bei Nacht oder falls man nicht da ist, zum Putzen von Fenstern oder auch wie das verwendete Lego Mindstorms in Form von Spielzeug für Kinder. [4], [5]

## III. Hauptteil

In den Vorbertrachtungen bin ich kurz auf die Inspirationsquellen unseres Roboters eingegangen. Nun aber die Frage, wie haben wir es realisiert? Wie schon eingangs erwähnt, wollten wir einen Roboter bauen, der sich selbtständig bewegen kann und dabei eine Fläche über das Kehrschaufelprinzip reinigt. Dabei sollte er Hindernisse erkennen können und mittels einer Manövrierfunktion diesen ausweichen können.

Dazu nutzten wir zur Fortbewegung Ketten, damit er sich auf der Stelle per Differentiallenkung drehen konnte. Dies ersparte uns unnötige Umwege bei der Steuerung des Fahrzeugs. Zusätzlich dienten sie auch zur Fahrt über unwegsameres Gelände, beispielsweise wenn er Kleinteile wegschob statt sie auf das Kehrblech zu befördern, was des Öfteren aus bautechnischen Gründen, insbesondere wegen der Kehrschaufel, passierte. Angesteuert wurden die Ketten mit insgesamt zwei Motoren, jede Seite separat steuerbar, was eine Differentiallenkung ermöglicht hat.

Objekte wahrnehmen beziehungsweise die Distanzen zu Objekten messen konnte unser Kehrroboter durch einen Ultraschallsensor, der oben über den NXT-Stein angebracht war. Dadurch war er in der Lage, seine Umgebung wahrzunehmen und Objekten auszuweichen. Zudem verhinderte die hochgelegene Position, dass er Objekte am Roboter als Hindernis erkennen würde, wie zum Beispiel die Kehrschaufel.

Die Anbringung des Staubwedels am vorderen Ende des Roboters erschien uns zunächst als problematisch, da wir zunächst statische Probleme bei der Anbringung des Armes hatten, damit dieser den Staubwedel rotieren lassen konnte. Gelöst haben wir das Problem durch die Anbringung weiterer Stützen am Arm. Dann mussten wir versuchen, den Staubwedel an den Motor zu befestigen. Dies gestaltete sich etwas schwierig, weil der Staubwedel kein Lego-Standardteil ist. Auch das konnten wir mittels einer kleinen Konstruktion lösen, in dem wir in den Schaumstoff, dass im Inneren des Wedels war, ein Legoteil reingesteckt haben und schließlich von außen eine kleine Konstruktion entworfen haben, um ihn dann an den Motor zu befestigen. Somit war auch dieses Problem für uns erstmal gelöst.

Die Kehrschaufel wurde von uns aus Pappe selbst gebastelt, was den Vorteil hatte, dass wir sie leicht an den Roboter anbringen konnten. Dies brachte das Problem mitsich, dass hin und wieder Teile unter das Kehrblech landeten, was teilweise eine Blockierung der Ketten zufolge hatte, aber zurückzuführen war auf die Tatsache, dass wir das Kehrblech selbst gebastelt haben, worauf ich später nochmal genauer eingehen werde, wenn es um die Probleme geht, die währrend des Baus aufgetreten sind. Desweiteren hatten wir so eine einfache Gelegenheit, ein Kehrblech zu befestigen, was gleichzeitig mit nur geringen Aufwand auswechselbar ist, was sich im Alltag wohl als einigermaßen nützlich erweisen würde.

Das war soweit der Aufbau unseres Roboters, nun ging es an die Programmierung. Zunächst über USB angesteuert, später dann über Bluetooth, haben wir ihn so eingestellt, dass er geradeaus fährt und sollte er ein Hindernis erkennen, soll er diesen ausweichen, sobald der Ultraschallsensor eine zu geringe Entfenrung zu dem Objekt misst. Bei USB klappte es hervorragend ohne nennenswerte Probleme, bei Bluetooth hingegen gab es einige Probleme, insbesondere mit den Übertragungsverzögerungen, die dadurch zustande kamen. Gelöst haben wir diese, indem wir den Roboter mehr Zeit für die Reaktion gaben. Genaueres zu dem Problem folgt in der Ergebnisdiskussion. (Abb.1)

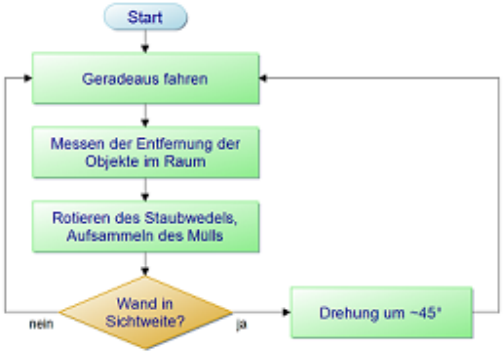



Abb.1: Schematischer Programmablauf

Im Test schließlich stellten Legoteile den Müll dar, den es vom Roboter einzusammeln gilt. Dies gelang auch in meisten den Fällen recht gut, einige Male trat jedoch das bereits genannte Problem auf, dass einige Teile unter das Kehrblech geraten sind. Für dieses Problem konnten wir keine allgemeine Lösung finden, die das Problem dauerhaft hätte lösen können. Grund dafür war wohl, dass es sich eben nicht um ein handelsübliches Kehrblech handelte.

Endergebnis unseres Projektes war also ein funktionsfähiger Kehrroboter, der in der Lage war, Kleinteile aufzusammeln und bei Bedarf, wenn Objekte sich näherten, diesen mittels Differentiallenkung auszuweichen. (Abb.2)

Abb. 2: Der fertige Kehrroboter

## IV. Ergebnisdiskussion

Wie gerade schon erwähnt, das Ergebnis unseres Projektes war ein einigermaßen funktionsfähiger Kehrroboter, der in der Lage war, grob den Schmutz zu kehren. Wie vorhin schon angesprochen, traten beim Bau und auch bei der Programmierung des Roboters diverse Probleme auf, die ich im folgenden Punkt etwas diskutieren möchte und auch auf unsere Lösungen dazu eingehen werde.

Zunächst einmal wäre da unser Staubwedel. Ein minderwertiges Modell aus der Drogerie, vom Stab abgetrennt und ein Legoteil reingesteckt, um es schließlich mittels einer kleinen Konstruktion an einen Motor zu befestigen. Dabei stellte sich die Frage: Wie sollen wir es am besten an den Roboter anbringen? Versuchen wollten wir es über ein Arm, was ohne Motor ganz gut geklappt hat, mit Motor stellten sich allerdings Stabilitätsprobleme ein, die wir mittels zusätzlicher Stützen am Arm beheben konnten. Schließlich gelang es uns, den Arm samt Staubwedel und Motor an den Kehrroboter zu befestigen und das Problem schien zunächst gelöst. Jetzt kam jedoch das Problem hinzu, dass der Staubwedel zu tief hing, weil er nur an einer Seite befestigt war und die andere Seite lose hinunter hing. Das Problem konnten wir leider nicht lösen, da uns keine Idee einfiel, wie wir eine zusätzliche Stütze anbringen konnten und gleichzeitig eine dadurch entstehende Blockierung des Staubwedels zu verhindern wussten. Wir haben versucht, einen zusätzlichen Arm auf der anderen Seite anzubringen. Nun trat das Problem auf, dass im Schaumstoff nur Platz für ein Legoteil war, welches wir schon auf der anderen Seite benutzt haben. Zudem würde es, wie grad schon erwähnt, die Rotation des Staubwedels blockieren, was zum Nachteil für den gesamten Kehrprozess wäre. Aber dazu später mehr.

Erstmal komme ich zum Kehrblech. Statt ein handelsübliches Kehrblech zu nehmen, wie wir es anfangs geplant haben, hatten wir erstmal vor, ein provisorisches aus Pappe anzufertigen, welches wir später durch ein richtiges, handelsübliches Kehrblech ersetzen wollten. Jedoch hätten sich bei einem richtigen Kehrblech weitere Probleme ergeben, zum Beispiel hätte es Probleme gegeben, das Kehrblech an den Roboter zu befestigen, da wir auch dazu erst eine Legokontruktion hätten anfertigen müssen, was wieder zum Problem geführt hätte, dass es vermutlich zu hoch liegt und es den Staubwedel und damit den Kehrprozess gehemmt hätte. Bei unserem Kehrblech aus Pappe hatten wir bereits Öffnungen für den Anbau vorhanden, die genau den richtigen Abstand zueinader hatten, sodass wir in der Lage waren, es einfach und auswechselbar an den Kehrroboter anzubringen. Zudem hatten wir so auch die Möglichkeit, die Größe des Kehrbleches anzupassen, was uns auch zugute kam, da wir so das Kehrblech an den Roboter anpassen konnten und mussten nicht die komplette Grundkonstruktion verwerfen und neu konstruieren, um ein richtiges Kehrblech anzubringen. Jedoch trat ein Problem dabei auf: Das vordere Ende verbog sich leicht und das hatte zur Folge, das hin und wieder einzelne Teile unter das Kehrblech gerieten, die dann teilweise für die Blockierung der Ketten und der damit verbundenen verhinderten Weiterfahrt des Roboters die Schuld tragen. Das Problem war auf die Eigenkonstruktion des Kehrblechs zurückzuführen und nur wenig behandelbar, lediglich durch immer weiteres Nachknicken, was aber auch zur Folge hat, dass die Pappe an der Knickstelle immer weicher wurde.

Das waren die Probleme , die währrend der Konstruktion des Roboters aufgetreten sind. Nun komme ich zu der Programmierung und die Probleme, die im Zuge dessen aufgetreten sind.

Zunächst hatten wir es über USB-Kabel angesteuert, was nahezu ohne Probleme funktioniert hat. Er konnte sich bewegen, den Staubwedel rotieren lassen und auch Hindernisse erkennen und diesen ausweichen. Das einzige Problem, welches wir im Zuge dessen hatten, war ein Problem mit dem Ultraschallsensor, welches jedoch ein Fehler seitens unserer Programmierung war, da wir dem Sensor nicht genügend Zeit gegeben haben, um die Entfernung akkurat zu messen.

Später entschieden wir uns dann dazu, den Roboter über Bluetooth anzusteuern, was einige neue Probleme mit sich brachte. Zum Beispiel wäre da die Verzögerung des Bluetooth, was zur Folge hatte, dass der Roboter zum Beispiel ein Objekt zu spät erkennt und erst, nachdem er gegengefahren ist, darauf durch ein Ausweichmanöver reagiert. Dies war nicht unser Ziel und so sollte er sich auch nicht verhalten. Gelöst haben wir das Problem, indem wir den Roboter mehr Zeit für die Reaktion gegeben haben. Zum Beispiel haben wir den Abstand vergrößert, in denen er Hindernissen ausweichen soll. So gaben wir dem Roboter genügend Spielraum, Hindernissen rechtzeitig auszuweichen. Zugleich gab es auch Probleme, dass der Roboter zuviele Befehle aufeinmal bekam, erkennbar an dem Piepen des Lego NXT-Steins. Lösen konnten wir dieses Problem auch immer nur temporär, indem wir die Reaktionszeit höher gestellt haben. Jedoch trat es immer wieder auf und wir konnten keine dauerhafte Lösung finden, in der es komplett weg war. Im folgenden Absatz wird dieses Problem nochmals genauer beleuchtet.

Das waren zunächst Probleme, die hauptsächlich beim Stillstand des Roboters auftraten. Als wir ihn mittels USB-Kabel angesteuert haben und ihn auf dem Tisch fahren ließen, konnten wir keine weiteren Probleme feststellen. Wie vorhin schon erwähnt, verhielt er sich genau so, wie er sollte. Weitere Probleme traten erst mit der Ansteuerung über Bluetooth auf, zum Beispiel das gerade erwähnte Piepen aufgrund zu vieler Befehle oder auch noch weitere Blockierungen des Staubwedels, wie vorhin schon erwähnt wurde. Aufgrund der Tatsache, dass der Wedel zu weit nach unten hängt und es dadurch zu einer Blockierung der Drehbewegung gekommen ist und schließlich wurde dadurch die Kehrfunktion des Roboters gehemmt. Auch kam es durch das Piepen vor, dass er Befehle nicht weiter ausgeführt hat und der Apparat in eine Art Schleife gelandet ist, aus der er sich so schnell nicht wieder befreien konnte. Dies trat hauptsächlich dann auf, wenn er mehrmals hintereinander ein Ausweichmanöver einleiten musste, zum Beispiel in der Ecke eines Raumes. Zurückzuführen ist das, wie eben schon erwähnt, auf die Verzögerung über Bluetooth. Lösen konnten wir es nicht zu 100 Prozent, lediglich mehr Zeit für das Ausführen der Befehle konnten wir den Roboter geben. Das Problem kam jedoch immer wieder vor und es lag womöglich an der geringen Übertragungsrate des Bluetooths.

## V. Zusammenfassung und Fazit

Wie im Hauptteil schon erwähnt, hatten wir am Ende des Lego Mindstorms Seminars einen funktionsfähigen Roboter, der in der Lage war, grob den Schmutz wegzukehren und somit für etwas Ordnung und Sauberkeit zu sorgen. Und das trotz unserer Probleme, die oft erst mit dem Bau und der Programmierung des Roboters am Ende miteinhergingen. Einige Fehler sind auch schlichtweg auf eine Ungenauigkeit der Legosensoren zurückzuführen, beispielsweise eine fehlerhafte Messung des Abstandes nahekommender Objekte. Zudem erwies es sich teilweise auch als problematisch, dass der Lego NXT Kernstein nur jeweils ein Befehl auf einmal verarbeiten konnte, was nicht nur einmal zu Problemen geführt hat, wie ich in der Ergebnisdiskussion beschrieben habe.

Nun noch die Frage, wie könnte man den Roboter noch verbessern beziehungsweise noch leistungsfähiger machen? Zunächst einmal könnte man die Effizienz beim Einsammeln von Verschmutzungen dadurch erhöhen, indem man einen höherwertigen Staubwedel verwendet statt eines minderwertigen, wie wir es gemacht haben.

Zudem wäre es noch möglich, den Roboter so zu programmieren, dass er sich seine bereits befahrenen Routen merkt und diese dementprechend nicht nochmal abfährt, sie also auch sozusagen als Hindernis anerkennt und so behandelt. So würde sich schließlich auch die Problematik mit der Drehung in immer derselben Richtung beheben, da er so weniger die Gefahr läuft, im Kreis zu fahren und effizienter und vor allem mehr Fläche abfährt. Eine weitere Verbesserungsmöglichkeit wäre, den Roboter seinen Akkustand ablesen zu lassen und bei Bedarf zu seiner Ladestation zu fahren, sich selbstständig andocken und so sich selbst aufladen. Dann müsste er noch die Position speichern, an der er bis vor kurzem war, und dann könnte er direkt da weitermachen, wo er vor dem Laden aufgehört hat.

Generell wäre es für die Effizienz besser, wenn er sich generell auch gefahrene Routen, diverse Objekte wie Schränke, Couchs usw. merkt und diese demensprechend im Algorithmus immer wieder abruft. Das funktioniert natürlich nur, wenn man nicht viel im Raum verändert oder verschiebt.

Zudem wäre es auch möglich, weitere Bürsten an der Seite oder am hinteren Ende anzubringen, mit Verbindungen zur vorderen Kehrschaufel versteht sich, sodass der Roboter auch in der Lage ist, die Ecken besser zu säubern beziehungsweise generell auch stärkere Verschmutzungen zu beseitigen. Letzteres wäre schon möglich, wenn man, wie im letzten Absatz erwähnt, einen höherwertigen Staubwedel verwendet.

## Literaturverzeichnis

[1] *http://de.wikipedia.org/wiki/Staubsaugerroboter*. – Eingesehen am 26.02.2018
[2] *http://http://saugroboter-staubsauger.de*. – Eingesehen am 26.02.2018
[3] *http://de.wikipedia.org/wiki/Kehrmaschine*. – Eingesehen am 27.02.2018
[4] *https://de.wikipedia.org/wiki/Serviceroboter*. – Eingesehen am 1.03.2018
[5] *http://www.haushaltroboter.com/*. – Eingesehen am 1.03.02.2018

# Bau eines Cleaning Bot
# Bericht zum LEGO Mindstoms Seminar

Malte Schwank, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

**Den Haushalt sauber zu halten erfordert viel Arbeit. Um uns bei dieser Arbeit helfen zu lassen bauen wir Roboter, die uns einige Aufgaben abnehmen. Im Rahmen des LEGO Mindstorms Seminar der Otto von Guericke Universität befassten wir uns mit dem Projekt einen Roboter zu bauen, der den Boden säubert. Diesen Roboter haben wir den Projektnamen Cleaning Bot gegeben und das besondere an ihn: Er besteht aus LEGO. Im folgenden Beitrag zum Projekt wird der Bau des Cleaning Bots erläutert. Es folgt eine Beschreibung der Funktionsweise sowie die Betrachtung der aufgetretenen Probleme. Nachfolgend gibt es eine Diskusion darüber wie sich der Cleaning Bot zu bereits existierenden Produkten auf dem Markt verhält.**

**Kettenfahrzeug, Konstruktion, LEGO, Saugroboter, Ultraschallsensor**

## I. Einleitung

ROBOTER werden gebaut um den Menschen von Arbeit zu entlasten. Dies gilt sowohl in der Industriebranche als auch im Haushalt. Um einen bequemeren Alltag zu ermöglichen wurden Roboter erfunden um Staub zu saugen, den Boden zu wischen oder den Rasen zu mähen. Diese gelten als praktisch und daher haben Thomas Schreiber und ich, es uns zur Aufgabe gemacht einen Roboter zu bauen der für uns sauber macht.
Dieses Projekt haben wir in den LEGO Mindstorms Seminar das im Zeitraum vom 12.02.2018 bis zum 23.02.2018 statt, fand, bewältigt. Das Ziel war es ein Fahrzeug zu konstruieren, das einen Raum abfährt und ihn säubert. Es muss in der Lage sein Hindernisse zu erkennen und dementsprechend ausweichen zu können. Die Konstruktion selbst ist inspiriert durch Fahrzeuge der Stadtreinigung. Müll wird durch einen rotierenden Staubwedel aufgenommen und auf einem integrierten Kehrblech gelagert.
Der Roboter wurde mit LEGO unter Verwendung des NXT-Bausteins aufgebaut. Der Quellcode wurde mit Matlab geschrieben. Mithilfe des von der RWTH Aachen bereit gestellten Toolkit (vgl. [1]) konnten wir auf die Sensoreingänge und Motorausgänge des NXT –Bausteins zugreifen, was uns letztendlich das Programmieren mit Matlab erst ermöglichte.

## II. Vorbetrachtungen

Für den Algorithmus in dem der LEGO Cleaning Bot dienten Saugroboter als Vorbild. Die älteren Modelle bewegen sich immer geradeaus bis ein Objekt oder eine Wand den Weg blockiert anschließend wechseln sie die Richtung. Eine Folge dieses Systems ist, dass einige Flächen nicht abgefahren werden, da der Saugroboter ohne System durch den Raum fährt. Außerdem saugen sie solange bis man sie manuell abschaltet.
Die neueren Modelle sind wesentlich effizienter. Sie bewegen sich intilligent durch den Raum. Das bedeutet, dass sie sich in einem bestimmten Muster auf einer Fläche bewegen. Ob diese Muster Spiralen oder Bahnen sind, ist bei jedem Sauroboter unterschiedlich. Den Weg die sie bereits abgefahren sind merken sie sich. Dadurch hören sie erst auf,sobald sie auch die ganze Fläche gereinigt haben. Zudem können sie sich die Position ihrer Ladestation merken. Sobald die letzte Bahn von dem Roboter bereinigt wurde, fährt er selbstständig zurück in seiner Ladestation.
Die neuen Saugroboter können außerdem auch Wände und schmalere Hindernisse wie zum Beispiel Tischbeine unterscheiden. Das ist dafür praktisch, dass der Roboter eine Ausweichkurve um das Hindernis herum fährt, anstatt umzukehren, weil er eine Wand sieht. Dies kann man auf Abbildung 1 sehen. Einige kann man außerdem noch mit seinem Handy von unterwegs starten lassen. Die neuen Modelle haben also sehr viele Komfortfunktionen, der LEGO Cleaning Bot wird daher, aufgrund der zeitlichen Beschränkung, eher an die Älteren erinnern.
Um einen verschmutzten Boden zu simulieren benutzten wir, weil es sich nun mal in so einem Seminar ergibt, LEGO Bauteile. Um ein möglichst großes Spektrum darzustellen benutzten wir sowohl sehr kleine Teile als auch größere Teile wie längliche Stäbe.

10.24352/UB.OVGU-2018-045 


Abbildung 1: Darstellung der Bewegung eines neueren Modells

## III. HAUPTTEIL

Bevor wir mit dem Bau an unserem Cleaning Bot beginnen konnten, mussten wir einen Handelsüblichen Staubwedel für unser Projekt kaufen. Nachdem wir uns für einen Staubwedel entschieden haben, konnte der Bau loslegen. Das endgültige Ergebnis wie der Cleaning Bot aussieht kann man auf Abbildung 2 beobachten.

Das konstruierte Fahrzeug wird mit drei Motoren betrieben die an den Motorausgängen des NXT-Bausteins angeschlossen sind. Der NXT befindet sich oberhalb des Fahrzeugs Gestells. Er ist so befestigt das der Bildschirm nach oben zeigt. Zwei Motoren sind für die Fortbewegung zuständig. Damit sich das Fahrzeug auf der Stelle drehen kann, was durchaus Notwendig ist z.B. in Ecken, bewegt es sich auf Ketten. Die Ketten haben sich zusätzlich noch als nützlich ergeben, beim überfahren von liegen gebliebenden Legoteile, die der Cleaning Bot nicht aufsammeln konnte. Das Wenden oder Drehen auf der Stelle ist dadurch möglich, da beide Seiten ihren eigenen Motor besitzen. Somit kann die linke Seite zum Beispiel vorwärts fahren und die rechte Seite gleichzeitig rückwärts.

Vor dem Gestell, an dem die Ketten und der NXT befestigt sind, befindet sich das Kehrblech. Das Kehrblech ist selbstgebastelt aus Pappe und lässt sich ganz leicht an dem Fahrzeug wieder an- und abbringen. Die richtige Anbringung des Kehrblechs hat sich als schwierig erwiesen, da es sich in erster Linie um ein nicht LEGO Bauteil handelt, aber auch da das Kehrblech in der richtigen Höhe sein muss. Anfangs hing das Kehrblech zu Hoch sodass der Dreck unterhalb des Blechs durch gerutscht ist. Zusätzlich musste das Kehrblech in einem bestimmten Winkel zum Boden ausgerichtet sein, weil es ansonsten nicht die LEGO Bauteile richtig greift.

Rechts vom Blech befindet sich der Arm des Cleaning Bots. An diesem ist der 3. Motor befestigt. Der Motor dient dazu den Staubwedel zu rotieren. Indem der Staubwedel rotiert schmeißt er Dreck auf das Kehrblech drauf. Den Staubwedel haben wir an den Motor befestigt indem wir ihn erst von seinem Stab getrennt haben und anschließend aufgeschnitten haben. In dem Aufgeschnittenen Staubwedel befand sich ein hohler Schaumstoffzylinder. In diesem konnte sich ein längliches LEGO Bauteil so befestigen, dass sich der gesamte Wedel dreht sobald man den Stab dreht. Den LEGO Stab konnte man dann bequem an den Motor anstecken.

An dem Sensoreingang des NXT-Bausteins befindet sich ein Ultraschallsensor. Er befindet sich seitlich neben dem NXT. Er ist dazu zuständig die Entfernung zwischen dem Fahrzeug und der nächsten Wand oder einem Hindernis zu messen. Der Sensor kann jedoch nur die Entfernung messen sofern das Hindernis gerade aus vor ihm liegen. Wenn das Fahrzeug in einem zu flachem Winkel auf einer Wand zufährt, dann kann der Sensor die Wand nicht erkennen und der Roboter schleift gegen die Wand.


Abbildung 2: Cleaning Bot

Generell funktioniert der Algorithmus des Cleaning Bots sehr simpel. Er ist in Abbildung 3 in Form einer PAP dargestellt. Sobald man den Bot startet, startet auch der Sensor. Er überprüft dauerhaft die Distanz zum nächsten Objekt. Wenn die Entfernung über einem bestimmten Schwellenwert befindet, fährt das Fahrzeug los und bringt den Staubwedel ins Rotieren. Es fährt nun solange bis sich die Entfernung unter dem Schwellenwert befindet. Sobald das passiert hört der Staubwedel auf sich zu rotieren, der Bot fährt einige Zentimeter nach hinten und macht eine Drehung um ca. 45°. Ab dann wiederholt sich die Schleife und der Ultraschallsensor misst erneut die Entfernung zur nächsten Wand und überprüft ob er weiterfahren kann.

Ursprünglich war der NXT mit dem Computer der Matlab und den Quellcode enthielt mit einem Kabel verbunden. Da es aber zeitlich noch möglich war, sind wir auf einer Verbindung durch Bluetooth gewechselt. Dies hatte den Vorteil, dass wir das Gerät in der Praxis auf größeren Flächen ausprobieren konnten. Es sind jedoch auch Probleme durch die Verzögerung aufgetreten. Der Schwellwert für die Entfernung zwischen Wand und Fahrzeug musste erhöht werden, da die Entfernung zu spät übertragen wird. Das hat dazu geführt, dass der Cleaning Bot schon gegen die Wand gefahren ist bevor er die Drehung einleiten konnte. Ursprünglich lag der Wert bei 27 Zentimeter, nachdem wir ihn aber mit Bluetooth angesteuert haben, mussten wir den Wert um 10 Zentimeter erhöhen.

Weitere Probleme die aufgetreten sind, sind Aufgrund der Konstruktion aufgetreten. In einer anfänglichen Konstruktion konnte das Gestell, auf dem der NXT befestigt ist, das Gewicht nicht standhalten. Die Folge davon war, dass das

Fahrzeug in der Mitte durchhing, was dazu führte, dass die Ketten nicht vollständig auf dem Boden auflagen. Deshalb konnte der Cleaning Bot anfangs nicht mal gerade fahren. Das nächste Problem ergab sich dann aus ähnlichen Gründen. Der Arm, an dem der Staubwedel befestigt ist, war zu schwach und hing durch. Sobald wir ihn jedoch stabiler angebracht haben hing er zu hoch. Es erwies sich als schwierig die richtige Höhe für den Staubwedel zu finden. Letztendlich haben wir eine Lösung für den Arm gefunden, aber es verabreichte den Cleaning Bot ein provisorischeres Aussehen, als er ohnehin schon hat. Ein immer noch bestehendes Problem ist, dass der Ultraschallsensor nicht in der Lage ist schmale Hindernisse wie ein Tischbein zu erkennen. Es kommt dadurch oftmals dazu, dass der Cleaning Bot sich an Tischbeinen aufhängt. Niedrigere bzw. flache Gegenstände werden ebenfalls nicht vom Ultraschallsensor erkannt. Das kommt davon, dass der Ultraschallsensor zu weit oben befestigt ist.
Für den LEGO Cleaning Bot bieten sich noch einige Verbesserungsmöglichkeiten. Um effizienter beim Reinigen einer großen Fläche zu sein würde sich anbieten, dass man das Fahrzeug sich, wie die neueren Modelle der Saugroboter, auf Bahnen bewegen lässt und bereits befahrene Wege sich merkt. Dadurch, dass das Fahrzeug sich bei einer Wand immer nur wegdreht besteht die Gefahr, dass es letztendlich nur im Kreis fährt. Um weitere Kollisionen mit Wänden zu entgehen kann man den Bot mit Sensoren für die Seite ausstatten. Eine weitere Idee wäre, dass man der Bot die Füllmenge seines Kehrblechs ablesen lässt. Dadurch kann der Bot aufhören sobald sein Kehrblech voll beladen ist. Man würde unnötiges weiterfahren vermeiden.

## IV. Ergebnisdiskussion

Der LEGO Cleaning Bot ist ein Kettenfahrzeug, das selbstständig fahren kann.
Er ist in der Lage auf der Stelle lenken zu können. Wände, Schränke und anderen große Objekte im Raum kann er problemlos ausweichen, jedoch ist er nicht in der Lage schmale oder flache Gegenstände zu erkennen. Dank der Bluetooth-Verbindung ist das Fahrzeug fähig sich dem Laptop, der den Quellcode enthält, für ausreichende Distanzen zu entfernen.
Trotzdem ist der LEGO Cleaning Bot im Verhältnis zu den handelsüblichen Saugroboter keine Konkurrenz. Besonders die neuen Modelle bieten sehr viel mehr Funktionen, die den LEGO Nachbau einfach fehlen. Er kann jedoch mit den älteren Modellen mithalten. Besonders kurz vor Wände, wo einige Saugroboter Probleme kriegen, da sie nicht ganz bis zur Wand saugen, überzeugt der Cleaning Bot. Das System mit dem Kehrblech funktioniert überraschend gut. Nur selten sind Teile entwischt. Wenn sie dann aber doch entwischt sind, dann Aufgrund dessen, dass der Staubwedel etwas größer als das Kehrblech ist. Der größte Nachteil des Kehrblechs ist aber seine Füllmenge. Sie ist zu schnell erreicht und sorgt dafür, dass in der Praxis, trotz maschineller Unterstützung, sich dauerhaft bewegen muss,da man das Kehrblech leeren muss. Aufpassen müsste man natürlich auch, sobald eine Treppe ins Spiel kommt. Das Fahrzeug hat keine Sensoren nach unten und würde daher gnadenlos blind abstürzen.

## V. Zusammenfassung und Fazit

Einen eigenen Cleaning Bot zu bauen, der aus LEGO besteht, ist uns im Rahmen des LEGO Mindstorms Seminar gelungen. Wir konnten ein Kettenfahrzeug konstruieren, das tatsächlich den Boden von Kleinteilen reinigt. Er hat unsere Erwartungen erfüllt. Wenn man ihn alleine in einem Raum lässt, kann man erwarten, dass der Raum hinterher sauberer ist. Die meisten Objekte und Wände kann er ausweichen, auch wenn er noch Probleme mit schmaleren Objekten hat. Das Kehrblech bietet zwar nicht nicht sonderlich viel Ablagerungsraum für den aufgesammelten Dreck, nimmt jedoch zufriedenstellend die meisten Teile auf.
So effizient wie ein handelsüblicher Saugroboter ist er dennoch nicht. Besonders bei dem System wie die Saugroboter vorgehen beim saugen stecken mehr Überlegungen dahinter. Bei unseren Cleaning Bot kann es mit Pech sein, dass der Raum nach drei Stunden noch nicht an jeder Stelle sauber ist. Trotz alledem funktionierte der LEGO Bau ganz gut und überraschte dadurch was man mit einer simplen Konstruktion alles machen kann.

## Anhang

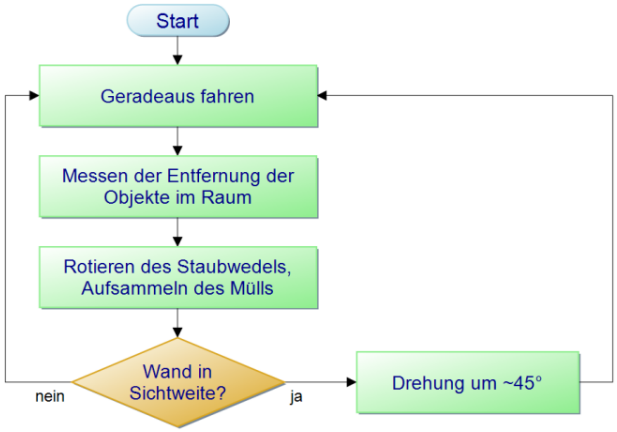



Abbildung 3: PAP für die Befehlskette des Cleaning Bots

## Literaturverzeichnis

*Bildquelle für Abbildung 1:*
http://www.irobot.de/Haushaltsroboter/Staubsaugen?gclid=Cj0KCQjwqM3VBRCwARIsAKcekb2GVrML4U9xbKbbah04WhY_NPq6EGJamBsi77ZfLjrMi9LvShksypIaAn0lEALw_wcB [22.03.2018]

[1] *RWTH Aachen: „RWTH - Mindstorms NXT Toolbox“ http://www.mindstorms.rwth-aachen.de/ [23.03.2018]*

# Geometrie-Scanner

Leander Bartsch, ETIT
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—In dem Paper „Geometrie-Scanner“ wird die Konstruktion und Programmierung eines Scanners mithilfe von Lego Mindstorms und dem Matlab NXT Toolkit beschrieben. Der Aufbau erfolgt mit einem Lego Mindstorms NXT und Lego Standartbauteilen. Er ermöglicht das Scannen eines DIN-A4-Dokumentes, welches die einfachen geometrischen Formen Kreis, Dreieck und Quadrat enthält. Es wird erklärt, wie ein Blatt pixelweise durch einen Lichtsensor digitalisiert werden kann. Das Hauptaugenmerk liegt auf der Entwicklung einer Software, die die Verarbeitung der eingescannten geometrischen Formen mit Matlab ermöglicht. So wird die Vorverarbeitung, wie das Umwandeln eines Graustufenbildes, in ein Schwarz-/Weiß-Bild mit einem dynamischen Schwellwert beschrieben, außerdem das Nummerieren und anschließende Isolieren von Objekten. Durch das Berechnen von formspezifischen Verhältnissen wird das Erkennen der geometrischen Formen erklärt. Die Verhältnisse sind die Kompaktheit und der Flächeninhalt des umliegenden Rechtecks zum Objektflächeninhalt. Außerdem wird der Flächeninhalt der erkannten Formen berechnet.**

***Schlagwörter*—Matlab Bildverarbeitung, Flächeninhalt, Lego Mindstorms, Objekterkennung, Scanner**

## I. Einleitung

SCANNER dienen der Erfassung und Digitalisierung von Informationen. Überall müssen Informationen elektronisch eingelesen und verarbeitet werden. Dabei ist der Begriff Scanner nicht auf die weitverbreiteten Flachbettscanner für Dokumente beschränkt, sondern umfasst einen weiten Bereich. So gehören auch 3D-Scanner, Handscanner und Radar dazu. [1] In dieser Arbeit soll es um die Entwicklung eines Scanners für DIN-A4-Dokumente im Rahmen des Projektseminars ETIT gehen, weshalb die folgenden Ausführungen auf Dokumentenscanner beschränkt sind.

Zu einem Scanner gehört auch immer eine Software, die im einfachsten Fall das eingescannte Dokument anzeigt. Oft soll die entstandene Bild-Datei weiterverarbeitet werden. So wird Software zum Dokumentenmanagment, zur Bildverbesserung oder zur Texterkennung eingesetzt. [2] Ebenso sind ingenieurstechnische Anwendungen denkbar, wie das Einscannen von technischen Zeichnungen. Dabei kann es wünschenswert sein, dass maßstabsgetreue Objekte vermessen werden. Neben Papierdokumenten können natürlich auch flache Körper zweidimensional erfasst werden.

Die Idee besteht darin, einen Scanner mit einem Lego Mindstorms NXT zu konstruieren, der ein DIN-A4-Blatt einscannen kann. Auf diesem Dokument sollen die einfachen geometrischen Formen Dreieck, Kreis und Quadrat abgebildet sein. Die Software des Scanners soll ein solches Dokument erfassen und die geometrischen Formen finden. Sie sollen voneinander isoliert und unterschieden werden. Es soll also das Erkennen der drei geometrischen Formen möglich sein. Die Software soll außerdem den Flächeninhalt der Objekte berechnen können.

DOI: 10.24352/UB.OVGU-2018-046 

## II. Vorbetrachtungen

### A. Scanner und Bildverarbeitungssoftware

Zum Scannen von Dokumenten sind zwei Scanner-Bauformen sehr verbreitet. Beim Flachbettscanner fährt ein lichtempfindlicher Sensor die Vorlage ab, welche selbst nicht bewegt wird. [1] Der Sensor ist dabei so breit wie das Dokument selbst. Bei einem Einzugscanner ist es genau andersherum. Das Dokument wird eingezogen und dabei an der Sensorzeile entlangbewegt. [1] Der Nachteil ist, dass nur einzelne Seiten und keine Bücher oder ähnliches eingescannt werden können. [1]

Neben der Software zur Kommunikation mit dem Scanner existiert diverse Software zur Weiterverarbeitung von Scans. Eine wichtige Anwendung ist die Texterkennung, auch als OCR (Optical Character Recongnition) bezeichnet. [3] Durch Texterkennung sollen die Buchstaben in einem Bild erkannt werden. Das ermöglicht die elektronische Weiterverarbeitung der eingescannten Texte. So ist nach einer Texterkennung das Nutzen einer Übersetzungssoftware möglich. [1] In anderen Bereichen existiert Software zur Verarbeitung von mikroskopischen Aufnahmen. [4] Sie ermöglicht das Zählen und Vermessen von Objekten. [4] Etwas ähnliches soll auch die in dieser Arbeit beschriebene Software leisten.

### B. Lego-Mindstorms-Geometrie-Scanner

Der DIN-A4-Scanner soll mit Lego Mindstorms aufgebaut werden. In den Vorjahren wurden beim Lego-Projektseminar schon Scanner konstruiert. [5] Allerdings lag das Hauptaugenmerk immer auf dem Scannen selbst und nicht auf der Weiterverarbeitung der Scans.

Als optischer Sensor zur Erfassung von Graustufen steht ein Lichtsensor zur Verfügung. Dieser kann nur einen kleinen Bereich erfassen. Deshalb muss das Prinzip des Einzugscanners und des Flachbettscanners kombiniert werden, da nur ein einzelner Pixel, keine ganze Blattzeile erfasst werden kann. So soll, wie beim Flachbettscanner, der Sensor über das Blatt bewegt werden, um eine Zeile des Dokumentes zu scannen. Wie beim Einzugscanner soll aber auch das Blatt bewegt werden, wodurch der Zeilenwechsel erfolgen soll. Mit diesem Vorgehen ist es möglich, die zwei nötigen Dimensionen mit dem ein Pixel großen Sensor abzutasten.

Die Software teilt sich in zwei Bereiche. Zum einen wird eine Software benötigt, die die Scanner-Hardware steuert. Außerdem ist eine Software zur Verarbeitung des Scans nötig. Die Entwicklung der benötigten Software erfolgt vollständig mit Matlab. Zur Programmierung des Lego Mindstorms NXT

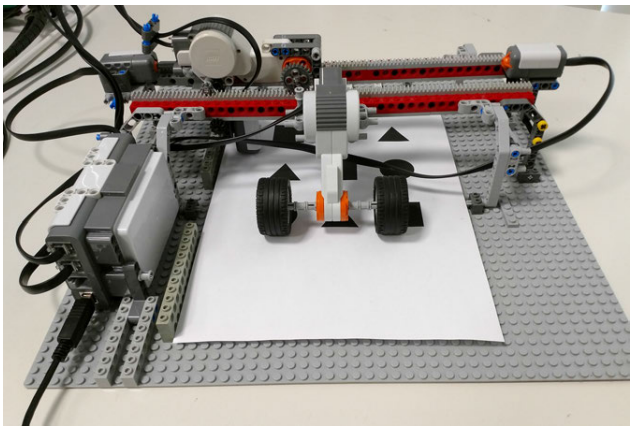


Abbildung 1. Hardware des Scanners mit eingezogenem Blatt

mit Matlab wurde die Mindstorms NXT Toolbox der RWTH Aachen genutzt.

## III. Hauptteil

### A. Hardware des Scanners

Zunächst soll die Funktionsweise des Lichtsensors erläutert werden. Im passiven Modus wird durch eine Fotodiode die Intensität des ankommenden Lichtes gemessen. [6] Im aktiven Modus wird Licht durch eine LED ausgestrahlt. [6] Dieses Licht wird durch eine Oberfläche reflektiert und zurück auf die Fotodiode geworfen, die wiederum die Intensität bestimmt. [6] Für den Scanner wird der aktive Modus genutzt. Die Intensität wird einheitenlos als Wert zwischen 0 für ganz dunkel und 1023 für ganz hell angegeben.

Wie schon im vorherigen Abschnitt erklärt, muss der Sensor in zwei Dimensionen über das Blatt bewegt werden, da dieser nur einen einzelnen Pixel messen kann. Dazu wurde auf einer Lego-Grundplatte ein Portal gebaut. Dieses ist breiter als ein DIN-A4-Blatt, weshalb das Blatt unter dem Portal eingezogen werden kann. Dies erfolgt mit zwei motorisierten Reifen, wobei diese nur locker auf dem Blatt aufliegen, um große Kräfte zu vermeiden. Auf dem Portal fährt ein Schlitten auf Zahnschienen hin und her. Der Schlitten kann den Lichtsensor 90 Grad entgegen der Einzugrichtung des Papiers über das Dokument bewegen. So tastet der Lichtsensor eine Zeile des Blattes nach der anderen ab. Unter dem Schlitten befindet sich der Lichtsensor zum Aufnehmen der Grauwerte. Dieser ist so tief befestigt, dass der Abstand zwischen Lichtsensor und Blatt möglichst gering ist. Auf dem Portal ist jeweils links und rechts ein Tastsensor angebracht, der als Endschalter für den Schlitten dient.

Die Abbildung 1 zeigt die vollständig aufgebaute Hardware. In der Mitte ist der Motor mit den Reifen für den Blatteinzug zu erkennen. Der links darüber liegende Motor ist der des Schlitten, welcher sich aktuell am linken Anschlag befindet.

### B. Software zum Scannen eines Dokumentes

Der Scan-Vorgang lässt sich in zwei Phasen unterteilen. In der ersten Phase fährt der Schlitten über das Blatt und der Sensor zeichnet die Helligkeitsdaten auf. In der zweiten Phase wird das Blatt ein Stück eingezogen und es geht wieder mit der ersten Phase weiter.

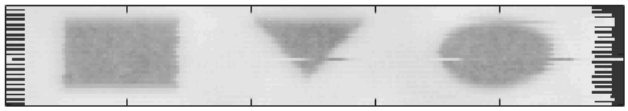


Abbildung 2. Eingescanntes Graustufenbild

In der ersten Phase muss zunächst der Motor zum Bewegen des Schlittens gestartet werden. Dabei soll der Schlitten sowohl bei der Fahrt von links nach rechts als auch bei der Fahrt von rechts nach links eine neue Zeile aufzeichnen. So kann auf das Zurückfahren des Schlittens verzichtet werden, was Zeit spart. Die Startposition ist immer links. Die Software muss also für den Scan einer Zeile folgende Aufgaben erfüllen: Der Motor des Schlittens muss entgegen der vorigen Richtung eingeschaltet werden. Gleichzeitig muss die Aufzeichnung der Helligkeitswerte beginnen. Die Helligkeitswerte werden mit einem konstanten Zeitabstand in einer eindimensionalen Matrix fester Größe gespeichert. Es findet keine Synchronisation der Position des Schlittens und der Messwerte statt. Deshalb muss die Fahrzeit des Schlittens möglichst konstant gehalten werden, wozu die in der NXT Toolbox verfügbare Drehzahlregelung aktiviert wird. Trotzdem schwankt die Anzahl der Messwerte von Zeile zu Zeile. Um keine Messwerte zu verlieren, muss die Matrix etwas größer als nötig gewählt werden. Dies hat zur Folge, dass einige Felder in der Matrix nicht genutzt werden. Deshalb verbleiben am Rand Initialisierungswerte, wie im Beispielscan in Abbildung 2 zu sehen. Beim Scannen von rechts nach links ist das Umdrehen und Verschieben der Pixel notwendig. Als letzter Schritt wird die eindimensionale Matrix als neue Zeile einer zweidimensionalen Matrix hinzugefügt.

Damit ist die erste Phase abgeschlossen und es folgt Phase zwei. Das Blatt wird ein Stück eingezogen, indem sich der Motor für den Einzug um einen festen Winkel dreht. Nun beginnt der Scan der nächsten Zeile in Phase eins. Die zwei Phasen können beliebig oft wiederholt werden, wobei die zweidimensionale Matrix immer größer wird. Die Breite dieser Matrix ist durch die Größe der eindimensionalen Matrix festgelegt. Hingegen ist die Zeilenanzahl beliebig. In der Software kann die Anzahl der zu scannenden Zeilen nach Bedarf eingestellt werden.

### C. Erkennung der geometrischen Formen und Berechnung des Flächeninhaltes

Nachdem der Scanvorgang abgeschlossen wurde, beginnt die Auswertung. Um das Ziel, die Unterscheidung verschiedener geometrischer Formen und Ermittlung derer Flächeninhalte zu erreichen, sollen im Folgenden die nötigen Algorithmen beschrieben werden.

Die Ausgangssituation ist eine zweidimensionale Matrix, deren Felder jeweils den Graustufenwert eines Pixels enthalten. In dieser müssen die eingescannten geometrischen Formen gefunden und isoliert werden. Dieser Verarbeitungsschritt wird als Segmentierung bezeichnet. [7] Das im folgenden beschriebene Verfahren gehört zu den pixelorientierten Verfahren zur Segmentierung. [7] Pixelorientiertes Verfahren meint,

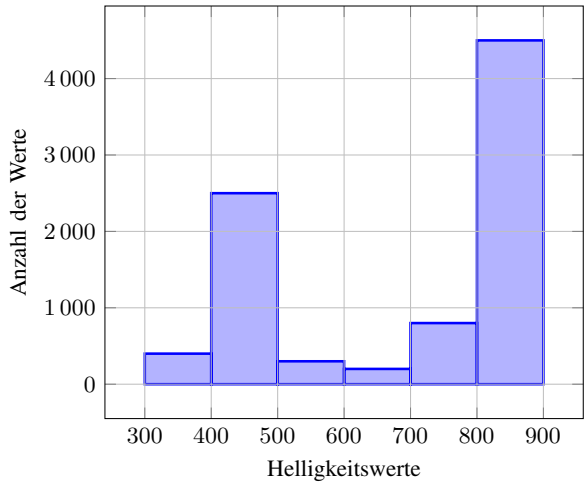



Abbildung 3. Histogramm der Helligkeitswerte (Grauwerte)

dass für jeden Pixel einzeln überprüft wird, ob dieser zum Objekt gehört oder nicht. [7] Im Fall der schwarzen Objekte auf weißem Hintergrund bietet sich eine Binarisierung an. Binarisierung bedeutet, dass jeder Pixel einem von zwei Segmenten zugeordnet wird. [8] Das eine Segment sind alle schwarzen geometrischen Formen. Das zweite Segment ist der Hintergund. Um die Binarisierung durchzuführen, wird den Grauwerten der Pixel, je nachdem ob sie kleiner oder größer eines Schwellwertes sind, eine Eins (gehört zum Objekt) oder eine Null (gehört zum Hintergrund) zugeordnet. [7] Es wird also das Graustufenbild in ein Schwarz-/Weiß-Bild umgewandelt, was auch als Threshold bezeichnet wird. Der Schwellwert kann mit Hilfe eines Histogrammes der Grauwertverteilung gefunden werden. [7] Ein solches Histogramm ist in Abbildung 3 zu sehen und gibt Aufschluss über die Verteilung der Grauwerte. [7] Der Schwellwert findet sich in dem Histogramm in Form des Tiefpunktes beim Helligkeitswert von etwa 600 wieder. Der Hochpunkt links davon, also die Helligkeitswerte um 400, sind die Objekte. Die Grauwerte um 800 hingegen gehören zum weißen Hintergrund. Das Finden des Schwellwertes erfolgt dynamisch durch die Software, indem der Tiefpunkt im Histogramm im Bereich von 400 bis 800 gesucht wird. Das Histogramm wird aus der Scan-Matrix erstellt. Damit ist ein erster wichtiger Schritt der Bildverarbeitung abgeschlossen. Das Ergebnis ist ein Bild, welches nur noch aus weißen Hintergrundpixeln und schwarzen Objektpixeln besteht.

Das Problem ist, dass eine unbekannte Anzahl $n$ an Objekten vorhanden ist. Im nächsten Schritt müssen also die Objekte voneinander isoliert werden. Dazu werden alle zusammenhängenden Pixel, die zu einem Objekt gehören, mit einer Nummer versehen. Diesen Schritt nennt man Labeln der Objekte und ist in Matlab als Funktion vorhanden. [9] Es werden die Einsen der Pixel durch die Nummer $n$ des Objektes ersetzt. Die Hintergrundpixel bleiben Null aber die Pixelwerte der Objekte sind nun 1 bis n, je nachdem zu welchem Objekt sie gehören.

Alle folgenden Schritte werden für alle 1 bis $n$ Objekte einzeln durchgeführt. Zunächst werden alle bis auf ein Objekt

Abbildung 4. Dreieck nach der Faltung mit dem Laplace-Kern

aus dem Bild gelöscht. Daraufhin wird durch Zählen der Pixel des verbleibenden Objektes der Flächeninhalt $A_O$ ermittelt. Das Objekt wird übersprungen, wenn das Objekt weniger als 50 Pixel groß ist (wenn $A_O < 50$). So werden Fragmente, die kein Objekt sein können, herausgefiltert.

Es folgt die Ermittlung des Flächeninhaltes des umliegenden Rechteckes $A_R$. Dazu werden alle Zeilen und Spalten der Matrix, die nur Nullen enthalten, gelöscht. In diesem Schritt werden auch Einzelpixel abgeschnitten, die als einzige außerhalb des Objektes liegen und damit wahrscheinlich durch einen Zeilenversatz entstanden sind. Es bleibt eine kleinere Matrix übrig, deren Größe der Flächeninhalt $A_R$ ist und deren Breite und Höhe der maximalen Breite und Höhe des Objektes entspricht.

Nun soll durch eine lineare Faltung die Kante isoliert und anschließend der Umfang $u$ des Objektes ermittelt werden. Eine Faltung ist eine mathematische Operation, für die ein Bild und ein Faltungskern benötigt wird. [10] Sowohl das Bild als auch der Faltungskern ist eine Matrix. [10] Durch Verrechnen beider Matrizen entsteht ein Ergebnisbild, welches eine Veränderung des Bildes je nach Art des Faltungskernes darstellt. [10] Wenn als Faltungskern der Laplace-Kern genutzt wird, bleibt im Ergebnisbild nur die Kante des Objektes über. Es bleiben nur die Pixel schwarz, welche an einem Farbübergang im Ausgangsbild, also an einer Kante, liegen. [10] Die Ermittlung des Umfangs $u$ erfolgt, indem die Anzahl der verbliebenen Pixel ermittelt wird. Es werden also die Konturpixel des Objektes gezählt. Die Abbildung 4 zeigt ein ausgeschnittenes Dreieck nach der Faltung mit dem Laplace-Kern.

Damit ist die Vorverarbeitung zur Identifizierung eines Objektes abgeschlossen. Die Identifizierung erfolgt nun durch die Berechnung von zwei Verhältnissen; die jeweils spezifisch für eine bestimmte geometrische Form sind. Das erste Verhältnis $V_1$ (vgl. (1)) ergibt sich aus dem Quadrat des Objektumfanges $u^2$ zu dem Flächeninhalt des Objektes $A_O$.

$$V_1 = \frac{u^2}{A_O} \tag{1}$$

Dieses Verhältnis wird Kompaktheit genannt und ermöglicht die Unterscheidung von Objekten. [7] Das Verhältnis ergibt für symmetrische, geometrische Formen, wie Kreis, Quadrat oder gleichseitiges Dreieck einen eindeutigen Wert.

Das Verhältnis $V_2$ (vgl. (2)) ist der Flächeninhalt vom Objekt $A_O$ zum Flächeninhalt des umliegenden Rechtecks $A_R$.

$$V_2 = \frac{A_O}{A_R} \tag{2}$$

Dieses Verhältnis ist nicht so eindeutig wie $V_1$. Dennoch ist es notwendig, da die Bestimmung des Umfangs $u$ nicht genau genug ist und damit die Toleranzen für $V_1$ zu groß sind.

Die eigentliche Identifizierung erfolgt nun durch ein Punktesystem. In der Software sind Wertebereiche für die Verhältnisse $V_1$ und $V_2$ für die drei zu identifizierenden Objekte hinterlegt. Für jedes mögliche Objekt, also Kreis, Quadrat und möglichst gleichseitiges Dreieck, wird ein Score berechnet. Das mögliche Objekt mit dem höchsten Score ist die vorliegende geometrische Form. Je schlechter der Scan ist, desto kleiner und damit uneindeutiger sind die Scores.

Als letzter Schritt erfolgt die Berechnung der Flächeninhalte. Dazu wird der Flächeninhalt $A_O$, also die Pixelanzahl, mit einem Faktor multipliziert. Dieser Faktor skaliert die Pixelanzahl des Bildes auf die reale Objektgröße. Je nachdem, als welche geometrische Form das Objekt klassifiziert wurde, wird der Faktor leicht variiert. Dies ist notwendig, da aufgrund des Rauschens an den Objektkanten der Flächeninhalt $A_O$ formabhängig schwankt.

## IV. Ergebnisdiskussion

Die an den Scanner gestellten Anforderungen werden fast vollständig erfüllt. So ist es möglich, ein DIN-A4-Blatt einzuscannen. Auch funktioniert die Weiterverarbeitung der eingescannten geometrischen Formen. Die Software ermöglicht die Erkennung der drei Objektarten Kreis, Quadrat und Dreieck. Allerdings funktioniert es nur, wenn das Quadrat und das Dreieck mit einer Seite parallel zur Bewegungsrichtung des Schlittens liegen. Diese Bedingung resultiert daraus, dass nur so das Objekt richtig ausgeschnitten und damit der Flächeninhalt des umliegenden Rechtecks berechnet werden kann. Außerdem ist die Erkennung von Dreiecken auf fast gleichseitige Dreiecke beschränkt.

Im Folgenden sollen Probleme bei der Konstruktion der Hardware erläutert werden. Zum Bau des Portals waren einige Versuche notwendig, um es stabil genug zu bauen. Das Konstruktionskonzept mit den Zahnschienen setzt voraus, dass sich der Motor direkt am Schlitten befindet. Deshalb muss neben dem Kabel des Lichtsensors auch das Kabel des Motors immer mit bewegt werden. Dies stellt ein Problem dar, weil die Lego Mindstorms Kabel sehr steif sind und so einen hohen Zug auf den Schlitten ausüben. Das Problem konnte reduziert werden, indem eine Kabelführung gebaut wurde.

Der räumliche Abstand zwischen den Messwerten in einer Zeile und der Vorschub zwischen den Zeilen ist verschieden groß. So werden, wie in Abbildung 2 zu sehen, alle Objekte verzerrt. Sie erscheinen flacher als sie wirklich sind. Dieses Problem konnte nur bedingt gelöst werden, da der Vorschub zwischen den Zeilen nicht beliebig klein wählbar ist. Auch konnte der Abstand der Messwerte in einer Zeile nicht einfach vergrößert werden, da so die Qualität des Scans stark abnahm.

Das wohl größte Problem war der Versatz zwischen den eingescannten Zeilen. Dieser resultiert daraus, dass der Datensatz aus dem Lichtsensor und die Position des Schlittens nicht miteinander synchronisiert werden. Das Problem wurde nicht vollständig gelöst, konnte aber gut genug reduziert werden. Zum einen werden die Zeilen um einen experimentell ermittelten Wert gegeneinander verschoben. Dieses Verschieben gleicht einen systematischen Fehler beim Scannen von der linken und rechten Richtung aus. Zum Anderen werden einzelne stark verschobene Zeilen von den Objekten abgeschnitten, um die Flächeninhaltsberechnung des umliegenden Rechtecks nicht zu verfälschen. So eine einzelne stark verschobene Zeile ist in der Abbildung 4 zu sehen.

Ein weiteres Problem war die Verschiebung des Grauwertbereiches durch unterschiedliches Umgebungslicht. Daraus resultierte ein anderer Schwellwert für die Binarisierung. Das Problem wurde durch das dynamische Ermitteln des Tiefpunktes im Histogramm gelöst.

Theoretisch reicht die Kompaktheit (vgl. (1)) aus, um einfache geometrische Formen zu unterscheiden. Das Berechnen des Umfanges ist sehr ungenau, da dieser nur durch Zählen der Randpixel ermittelt wird. Daraus resultiert auch eine ungenaue Kompaktheit, weshalb es nötig war, das Verhältnis $V_2$ einzuführen. Leider war es damit nicht mehr möglich, schief liegende geometrische Formen zu unterscheiden, da bei diesen die Berechnung des Flächeninhaltes des umliegenden Rechtecks sehr komplex wird.

## V. Zusammenfassung und Fazit

Das Paper beschreibt den Bau eines Scanners für DIN-A4-Dokumente mit Lego Mindstorms und einer Matlab-Software zur Bildverarbeitung. Die beschriebene Konstruktion ermöglicht die Erfassung eines DIN-A4-Blattes. Dazu wurde in der Arbeit zum Projektseminar ETIT ein Verfahren zur Segmentierung von Formen erklärt. Außerdem wurde ein sehr einfaches Verfahren zum Unterscheiden der Objekte auf Basis zweier Verhältnisse aus geometrischen Größen beschrieben. Dieses funktioniert zwar prinzipiell, ist aber recht unzuverlässig. Das Ergebnis verbessert sich mit zunehmender Größe der Objekte bzw. zunehmender Auflösung der Scans, da die Kanten sauberer werden. So ließe sich das Problem lösen, indem die Qualität der Scans verbessert wird oder indem durch eine verbesserte Vorverarbeitung die Kanten sauberer werden. Außerdem wäre es wünschenswert, wenn weitere und beliebig liegende Objekte erkennbar wären. Dazu müsste das Berechnen des umliegenden Rechtecks für beliebige Objekte ermöglicht werden. Trotz alledem ist das Endergebnis sehr zufriedenstellend.

## Literaturverzeichnis


[1] Wikipedia, the free encyclopedia: *Scanner (Datenerfassung)*. https://de.wikipedia.org/wiki/Scanner_(Datenerfassung). Version: 12. Januar 2018

[2] Wikipedia, the free encyclopedia: *Dokumentenscanner*. https://de.wikipedia.org/wiki/Dokumentenscanner. Version: 12. Januar 2018

[3] ubntuusers: *Texterkennung*. https://wiki.ubuntuusers.de/Texterkennung/. Version: 16. März 2018

[4] GSA GmbH: *GSA Image Analyser*. https://image.analyser.gsa-online.de/?lang=de. Version: 16. März 2018

[5] Otto-von-Guericke-Universität Magdeburg: *Lego-Praktikum an der Otto-von-Guericke-Universität Magdeburg — OVGU*. https://www.youtube.com/watch?v=nVnWycZoGAs. Version: 18. März 2018

[6] Bangemann, Felix: *Lego Mindstorms in der FEIT*. Otto von Guericke Universität Magdeburg Fakultät für Elektrotechnik und Informationstechnik, 2018

[7] Meyer, Michael: *Segmentierung*. http://www.weblearn.hs-bremen.de/risse/AWI/SEGMENT/SEGMENTI.HTM, 8. März 2018

[8] Hendrik Horn, Jörgen K.: *Segmentierung von Bilddaten*. http://mnementh.de/static/data/Segmentierung.pdf, 2004

[9] Xiaolu Zhao, JuanLu: *Labeling mit MATLAB*. http://pille.iwr.uni-heidelberg.de/~ocr02/Labeling%20mit%20MATLAB.html. Version: 13. März 2018

[10] rn-wissen: *Bildverarbeitung Tutorial*. http://rn-wissen.de/wiki/index.php?title=Bildverarbeitung_Tutorial. Version: 16. März 2018

# Bau eines flächenberechnenden Scanners

Chris-Marvin Hamann, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Abstract*— In dem LEGO Mindstorms Projekt welches von der Otto-von-Guericke-Universität durchgeführt wurde, sollte man sich selbst kreativ betätigen, in dem sich jede Gruppe eine individuelle, praxisbezogene Anwendung der gegebenen Bauteile ausdenkt und diese dann auch umsetzt. Bezogen auf diese Aufgabenstellung haben wir uns in unserer Gruppe entschlossen einen Scanner zu entwerfen, jedoch wurden schon in den Vorjahren des Projekts Scanner gebaut, welche zum Beispiel Bilder eingescannt haben, somit wurde innerhalb unserer Gruppe beschlossen automatische Flächenberechnung und –erkennung als neue Funktionen hinzuzufügen. Im folgenden Beitrag wird beschrieben wie diese Idee konstruktionstechnisch, sowie auch in Hinsicht auf das Programmieren umgesetzt haben. Nachfolgend wird diskutiert, inwiefern diese Idee besser umsetzbar gewesen wäre, so dass mögliche Fehler minimiert werden könnten. Dazu wird genauer auf den Aufbau des Scanners eingegangen und die Flächenerkennung anhand unserer Programmierung und verschiedenen Messwerten näher erklärt und wodurch dabei Fehler aufgetreten sind.**

***Schlagwörter*—Flächenberechnung, Formenerkennung, Lego, Mindstorms, Matlab, Scanner**

## I. Einleitung

SCANNER gewinnen in der heutigen Gesellschaft zunehmend an Bedeutung, so findet man sie nicht nur in der Industrie, sondern auch schon im Alltag. So besitzt ein Großteil der Bevölkerung einen Drucker und in diesen ist meist auch gleich eine Scanfunktion mit enthalten, diese kann zum Beispiel auch für das Universitätsleben nützlich sein, da es vorkommen kann, das man im Krankheitsfall die Aufzeichnungen eines Kommilitonen nutzen möchte oder einfach ein Skript für einen Vorlesung braucht. Manchmal fällt es einem somit schwer Drucker beziehungsweise Scanner aus dem Alltag wegzudenken. In der Industrie ist wohl der größte Nutzen von Scannern das Ein- und Auslesen von Daten, wie zum Beispiel bei einfachen Barcodescannern. Da in unserem Fall der Scanner jedoch zum Flächenberechnung und – erkennung genutzt werden soll, könnte man sich dies zum Beispiel im technischen zeichnen vorstellen und entworfene Räume im Maßstab zu berechnen oder auch in der Logistik um eventuell Pakete nach ihrer Größe und Form zu sortieren und somit eine bessere Lagerung zu ermöglichen.
In dem Projekt, welches ich zusammen mit Leander Bartsch bearbeitete und im Rahmen des LEGO Mindstorms Seminars vom 12. Februar bis zum 23. Februar 2018 entstand, wollten wir nun einen wie oben beschriebenen Scanner bauen. Somit haben wir unter Betrachtung der vorgegebenen Bauteile und des beschränkten Arbeitszeitraum beschlossen, einen Scanner zu bauen, welcher ein Blatt mit von uns ausgedruckten geometrischen Formen, was in diesem Fall Kreise, Quadrate und gleichseitige Dreiecke umschloss, automatisch einzuziehen und durch einen auf der Konstruktion fahrenden motorbetriebenen Lichtsensor einzuscannen. Der angeschlossene Computer soll dann die ermittelten Messwerte auswerten und über verschiedene Schritte, welche im Hauptteil ausführlicher beschrieben werden, den Flächeninhalt der Fläche erkennen und um welche geometrische Form es handelt.
Der Scanner wurde unter Verwendung des uns zur Verfügung gestellten NXT-LEGO-Bausteins gebaut und daraufhin mit Matlab programmiert. Das gescannte Bild, sowie auch die einzelnen Schritte, wie berechneter Flächeninhalt und erkannte Form, sollten auf einem Interface auf dem Bildschirm des Rechners ausgegeben werden.

## II. Vorbetrachtungen

Wie im vorherigen Abschnitt bereits erwähnt, gab es in früheren Jahrgängen innerhalb dieses Seminares schon Versuche Scanner zu bauen. Da uns die Informationen zu diesen Scannern mündlich von unseren Gruppenleitern weitergetragen wurden, sind sie nicht sonderlich detailliert und nicht belegt, den zweiten Scanner kann man in einem unter [1] im Literaturverzeichnis angegebenen Link sehen.
Beide Scanner hatten, im Gegensatz zu unserem, das Ziel ein Schwarz-Weiß Bild einzuscannen und danach digital wiedergeben zu können. Der Unterschied der beiden lag vor allem in der Art und Weise in der der Lichtsensor zum Scannen transportiert werden sollte.

### *A. Lichtsensor an hydraulischem Arm befestigt*

In diesem Beispiel aus dem Jahr 2016, ist der Lichtsensor an einem Arm befestigt, welcher gleichmäßig hin und her fährt und durch einen Motor betrieben ist. Eine Schwierigkeit hierbei ist gewesen, dass sich der Arm in gleichmäßiger Geschwindigkeit bewegen soll. Dazu musste nämlich eine Funktion aufgestellt werden, welche von den damaligen Projektteilnehmern empirisch ermittelt werden musste. Dies ist notwendig, da es sonst zu gewissen Messungenauigkeiten kommen würde, was vor Allem beim Einscannen eines Bildes fatal wäre, da dort sehr genaue Messwerte benötigt werden. Man sieht außerdem, dass sie den Lichtsensor so eingestellt haben, dass er im Aktivmodus ist, wo er selbst Licht ausgibt und das reflektierte Licht misst. Für das Einscannen eines

DOI: 10.24352/UB.OVGU-2018-047 
DOI: 10.24352/UB.OVGU-2018-047 Lizenz: CC BY-SA 4.0

Schwarz-Weiß Bildes ist dies optimal, da die gemessenen Lichtwerte die Schwarz- und Weißinhalte des Bildes wiedergeben können. Bei bunten Bildern müssen nämlich die Blau-, Grün- und Rotanteile des Bildes ermittelt werden, was ein deutlich größerer Rechenaufwand wäre und mit dem NXT dementsprechend lange dauern würde. Zudem ist der Einfluss äußerer Lichtquellen dadurch abgeschwächt. Diese Umsetzung soll, nach Angaben unserer Projektleiter sehr detailliert gewesen sein.

### *B. Lichtsensor fährt auf Zahnschienen*

Dies ist die Anwendung welche auch wir genutzt haben, dort befindet sich der Lichtsensor direkt mit dem Motor verbunden auf Zahnschienen. Der Motor betreibt dann Zahnräder durch sich der Lichtsensor dann bewegt. Der Vorteil bei dieser Methode ist, dass man für eine gleichmäßige Bewegung lediglich eine bestimmte Geschwindigkeit einstellen muss. Man muss jedoch darauf achten, dass die Zahnräder auch fest auf die Schienen eingespannt sind, da es sonst zu Ungenauigkeiten kommen kann. Dieser Scanner soll nach den Aussagen unserer Gruppenleiter ziemlich ungenau gewesen sein, dies kann natürlich an Konstruktionsfehlern liegen oder daran, dass die Ungenauigkeiten, die durch die Transportationsart entstehen, zu groß sind um Bilder einzuscannen, da die Unterschiede zwischen den Grautönen relativ klein sei können und dadurch nicht akkurat erfasst

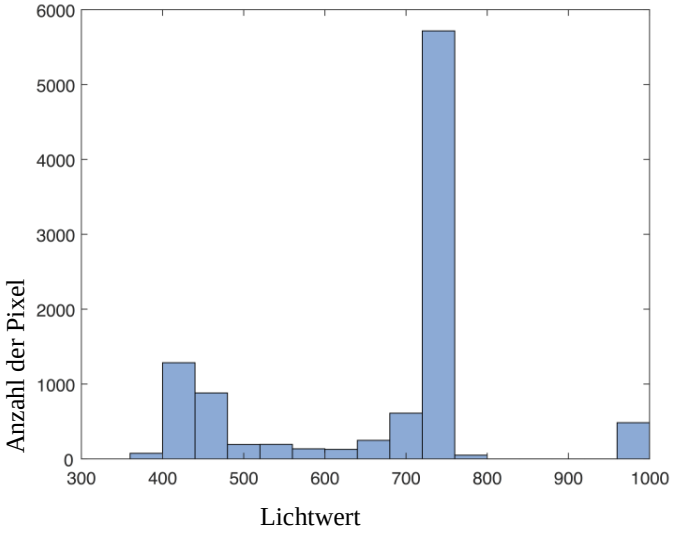



Abbildung 1: Vom Lichtsensor gemessene Werte

werden können.

## III. HAUPTTEIL

Nun werde ich genauer auf unsere Umsetzung eingehen.
Am Anfang erwähnte ich, dass der Blatteinzug automatisch erfolgen sollte. Dies haben wir so realisiert, dass das oberste Ende des einzuziehenden Blattes unter einen Reifen geklemmt wird, welcher dann auf Knopfdruck das Blatt einzieht, außerdem wird das Blatt immer etwas weiter weingezogen, sobald der Lichtsensor eine Zeile des Blattes gescannt hat. Glücklicherweise lässt sich der Motor in Hinsicht auf Drehwinkel und Drehgeschwindigkeit ziemlich genau einstellen, so dass die Scangenauigkeit dadurch nicht beeinträchtigt wurde.
Wie in den Vorbetrachtungen erwähnt, haben auch wir den Lichtsensor auf Zahnschienen fahren lassen, des Weiteren haben wir zwei Tastsensoren genutzt, damit der Motor stoppt und dann in die andere Richtung weiter fährt. Je nach dem in welche Richtung der Lichtsensor fährt, werden die Werte auch spiegelverkehrt dargestellt, weshalb wir im Programm implementiert haben, dass die jeweils zweite Reihe gedreht wird.

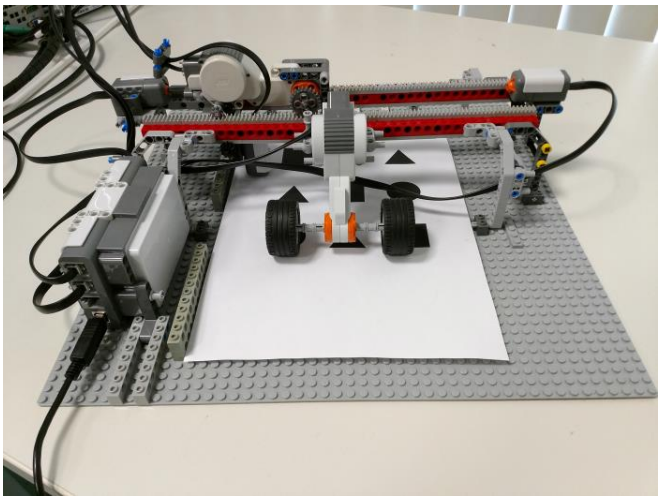


Abbildung 2: Scanner mit eingezogenem Blatt

Während der Lichtsensor hin- und herfährt, werden die reflektierten Lichtwerte gemessen, welche man in der Abbildung 1 in einem Diagramm dargestellt sehen kann. Dort kann man sehen dass die Werte von 0 bis 1023 gehen, dabei ist null sehr dunkel, also schwarz und 1023 weiß. Durch die vor Ort gegebenen Helligkeitsverhältnisse liegen die Werte welche Schwarz entsprechen bei circa 400 bis 500 und für Weiß bei 700 bis 800. Das Programm sucht dann automatisch einen Schwellenwert aus dem Diagramm, mit welchem die eindeutige Unterteilung in Schwarz und Weiß für die weitere Bearbeitung erfolgen kann. Dies erreicht man, in dem das Programm den niedrigsten Wert von dem Intervall zwischen ca. 400 bis ca. 750 sucht, welcher dann als eine Art Grenze zwischen den beiden Farben dient.

Abbildung 3: Durch Lichtwert dargestellte Formen

Abbildung 4: Formen nach Unterteilung

In den obigen Abbildungen 2 und 3 sieht man, wie das Bild vor und nach der Bearbeitung aussieht. Hier wird jedoch Weiß als Gelb und Schwarz als Blau dargestellt. Man kann zudem erkennen, dass im ersten Bild dunkelblaue Ränder vorliegen

und im zweiten nicht. Dies sind die Initialisierungswerte des Lichtsensors, wo er noch keine Messungen vollführen kann, diese wandelt das Programm sofort in Weiß um.
Nachdem diese eindeutige Unterteilung stattgefunden hat, wird jedes Objekt mit einer bestimmen Menge von zusammenhängenden schwarzen Pixeln mit einer spezifischen Zahl versehen. Durch diesen Schritt werden die einzelnen Objekte isoliert, wodurch die Formerkennung durchgeführt werden kann, außerdem werden zu kleine Objekte dadurch übersprungen, welche eventuell durch Fehlmessungen entstanden sind. Somit findet sich dann jedes Objekt ausgeschnitten auf einem einzelnen Bild vor.
Das Programm führt nun eine Faltung zur Kantenerkennung durch, dies wird mit einer in Matlab vorgegebenen Funktion durchgeführt. Hierbei wird der Laplace-Kern genutzt, welcher eine Faltungsmatrix speziell zur Kantenglättung ist. Dadurch werden nun also die Kanten des Objektes ausgeschnitten und man erhält die Hülle des Objektes wie sie in Abbildung 5 zu sehen ist.
Nun hat man alle Vorbedingungen geschaffen um nun die Flächeninhaltsberechnung und die Formerkennung durchzuführen. Hierfür werden zuerst zwei formspezifische Verhältnisse berechnet

$$.V_1 = \frac{u^2}{A} \qquad (1)$$

$$V_2 = \frac{A}{A_R} \qquad (2)$$

Das u was in Gleichung (1) gegeben ist, entspricht der Pixelmenge der zuvor bestimmten Hülle des Objektes und das A entspricht der Pixelanzahl des ursprünglich eingescannten Objektes, das $A_R$ welches in Gleichung (2) gegeben ist, entspricht zudem der Pixelzahl eines um die Objekte gelegtem Rechtecks. Da alle diese Werte in Pixeln angegeben sind, haben sie noch keine direkte Aussagekraft über die Größe des Objektes. Nun wird aus den berechneten Verhältnissen ein Score für jedes Objekt bestimmt, so dass nun jedes Objekt einen Score für jede Form zugewiesen hat. Also wird nach beiden Berechnungen geguckt, welcher Score jeweils am größten ist und dadurch wird letztendlich die spezifische Form des Objektes erkannt. Nun muss nur noch der Flächeninhalt bestimmt werden, dies geschieht indem der als A bestimmte Wert mit einem empirisch bestimmen Wert multipliziert wird. Welcher das Maß ist, wie viele $mm^2$ in einem Pixel enthalten sind. Danach hat man somit auch den Flächeninhalt gegeben, das fertige Ergebnis ist in Abbildung 5 zu sehen.


Abbildung 5: fertiges Scanergebnis

## IV. Ergebnisdiskussion

Wie in Abbildung 5 zu sehen, kann die Formerkennung, da dort ein Quadrat als solches erkannt wurde, und auch die Flächeninhaltsberechnung, welcher hier mit 862 $mm^2$ von ursprünglichen 900 $mm^2$ erkannt wurde, relativ verlässlich ausgeführt werden. Jedoch gibt es selbstverständlich auch hier einige mögliche Fehlerquellen.
Zum einen kann man in Abbildung 3 und 4 starke Verschiebungen bei dem Kreis und dem Dreieck sehen, welche das Ergebnis verfälschen können, ob diese durch eventuelle Messungenauigkeiten des Lichtsensors selbst kommen oder durch den Transport des Sensors auf Zahnschienen entstanden, konnten wir nicht herausfinden. Dieses Problem haben wir insofern versucht zu lösen, indem wir die Verschiebungen, sobald sie stark genug aus dem Objekt herausragen, einfach rauslöschen. Doch vor Allem bei dem Dreieck machte dies nur beschränkt Sinn, da wir das Löschen nur bei in einer Spalte vereinzelt aufgetretenen Messwerten durchgeführt haben und dies bei der Form des Dreiecks somit nicht so gut funktioniert. Außerdem konnte es bei dem Dreieck passieren, dass durch zu starke Verschiebungen die Spitze des Dreiecks abschnitten wurde, was auch zu Problemen bei der Formerkennung geführt hat.
Ein weiteres Problem war die Ermittlung des Schwellwertes, da diese bei starken Veränderungen des Lichteinflusses nicht unbedingt geklappt hat. So zum Beispiel, als die Sonne ungeschützt auf den Scanner eingestrahlt hat oder wenn man in einen anderen Raum mit anderer Beleuchtung gegangen ist. Unsere Lösung hierbei war das Suchen eines Tiefpunkts in einem bestimmten Intervall zwischen den beiden Maxima, aber bei zu großen Lichtveränderungen hat dies nicht zur Völle funktioniert. Eine andere Lösungsmethose wäre, dass man, wie bei handelsüblichen Druckern samt Scanner, den Sensor komplett einbaut, um ihn vor äußeren Einflüssen zu schützen, jedoch hatten wir dafür weder genug Bauteile noch genug Zeit, um solch große Konstruktionsänderung durchzuführen.
Zu guter Letzt war die Genauigkeit des Scanners an sich auch ein Problem. Um eine Genauigkeit zu erreichen um immer perfekte Ergebnisse zu erreichen, müsste der Scanner dementsprechend langsam hin- und herfahren um scharfe Messwerte zu erhalten, doch man musste einen Mittelwert zwischen Messpräzision und –geschwindigkeit finden. So hätte man wahrscheinlich für ein 3 cm hohes Objekt circa 10 Minuten gebraucht, um ein perfekt scharfes Bild zu erhalten. Da wir den Scanner jedoch vorstellen mussten, haben wir die Geschwindigkeit erhöht, weshalb also gewisse Ungenauigkeiten entstanden sind. So hätte man des Ergebnis, würde man nur auf die Akkuratesse des Messergebnisses bedacht sein, so hätte man das Ergebnis durchaus noch etwas verbessern können.

## V. Zusammenfassung und Fazit

Zusammenfassend kann man also sagen, dass wir in unserer Gruppe das Ziel durchaus erreicht haben. Wir waren relativ kreativ bei der Projektfindung, indem wir ein Projekt nahmen welches durchaus schon in den vorherigen Jahrgängen genutzt

wurde und haben aber noch einen individuelle Veränderung an der Arbeitsweise des Apparates vorgenommen. Die Konstruktion hätte man zudem durchaus auch noch schöner gestalten können, da wir den NXT verhältnismäßig lose an der Seite montiert haben, aber im Endeffekt geht Effizienz vor Design. Das Programm hätte man zudem auch definitiv noch effizienter und übersichtlicher schreiben können, aber da sowieso beim Programmieren eines Projektes solcher Größe nie Alles auf Anhieb funktioniert, musste man Wege finden die vielleicht nicht zeiteffizient oder übersichtlich sind, aber ihren Zweck erfüllten. So haben wir zum Schluss eine Genauigkeit von circa 60 bis 70 Prozent bei der Formerkennung und der Flächeninhalt wich maximal zu 60 mm ab, unter den gegebenen Umständen kann man mit diesem Ergebnis also durchaus zufrieden sein. Wenn man diesen Scanner für die Industrie nutzbar machen möchte, so müsste man definitiv mehr Bauteile brauchen, da er dort zum einen viel größere Objekte scannen müsste und zudem genauer Scannen müsste, was den Einsatz besserer Sensoren voraussetzen würde.

Zudem hat mich Matlab als Programmiersprache positiv überrascht, da wir sie schließlich vorher in der Uni noch nicht behandelten. Sie ist kurz gehalten und hat mit der Seite der Entwickler Mathworks eine gute Alternative um nach geeigneten Befehlen zu recherchieren. Was man jedoch etwas bemängeln kann, ist die gelegentlich nicht wirklich kontinuierliche Notation, vor Allem im Bezug darauf ob man eckige, runde oder doch mal geschweifte Klammern setzen soll.

Das Projektseminar an sich, konnte ich am Ende der zwei Wochen auch mit positiven Eindruck verlassen, da es meiner Meinung nach einen geeigneten Einblick in den Studiengang der Elektro- und Informationstechnik bietet, wobei der Schwerpunkt natürlich auf dem programmieren gelegt war. So hat man die Gelegenheit gehabt wie es ist, ein eigenes Projekt zu entwerfen und dieses auch irgendwie zu realisieren. Außerdem haben wir einen Ausblick drauf gehabt, was man mit unserem aktuellen Wissensstand schon realisieren kann und außerdem was später vielleicht noch so möglich wäre. Zudem musste man auch auf das Zeitmanagement achten, was für Studenten später auch definitiv wichtig wird. So werde ich den Bericht mit diesem positiv ausfallenden Fazit in Hinsicht auf die Projektdurchführung durch die Unimitarbeiter und auch auf die Leistung unserer Gruppe in dem kleinen Zeitraum beenden.

## LITERATURVERZEICHNIS

[1] Otto-von-Guericke-Universität Magdeburg | OVGU : Lego Praktikum an der Otto-von-Guericke-Universität Magdeburg | OVGU . https://www.youtube.com/watch?v=nVnWycZoGAs (1:26) . Version: 21.03.2018

# Bau eines Spinnenroboters

## Bericht zum Projekt des Lego Mindstorms Seminars

Philipp Schümann, ET-IT
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Moderne Roboter verwenden immer öfter Bilderkennung um sich zu Orientieren und ihre Aufgabe zu erfüllen. Durch immer höhere Kameraauflösungen können selbst kleine Details besser erkannt werden. Die Auswertung von Bilddaten verbraucht viel Rechenleistung und nicht immer müssen kleinste Details erkannt werden. Im Folgenden wird versucht eine Möglichste schnelle Zielerfassung mithilfe von Bildverarbeitung zu realisieren. Die Bilder werden durch ein Gütekriterium in der Größe variiert, um so die Auswertung zu beschleunigen.**

***Schlagwörter*—Bildauswertung, Lego, Matlab, Spinne, Zielerfassung**

## I. Einleitung

DIE Bildauswertung wird bei Robotern immer wichtiger. Vor allem in der Industrie ermöglicht diese dem Roboter immer komplexere Aufgaben zu lösen, doch auch privat kommen immer häufiger Roboter mit Bildauswertungstechnik zum Einsatz. In der Industrie ermöglichen Roboter das Arbeiten an gefährlichen Orten, ohne dabei Menschen zu gefährden. Auch ist Bildauswertung an Robotern sehr gut geeignet um das Zusammenarbeiten zwischen dem Roboter und Menschen zu ermöglichen.

In dem Projekt im Rahmen des LEGO Mindstorms Seminars wurde ein Roboter entworfen und programmiert, der in der Lage sein sollte, auf ein Ziel, welches mithilfe von Bildauswertung erkannt werden sollte, sich auszurichten und zu schießen.

Die Konstruktion des Roboters in Form einer Spinne kann für unwegsames Gelände verwendet werden, in dem normale Roboter mit Ketten oder Rädern sich nicht fortbewegen können.

Der Roboter wurde aus LEGO gebaut, zum Antrieb des Roboters wurden ein NXT-Baustein und drei NXT-Motoren verwendet. Programmiert wurde der Roboter in Matlab mit der Toolbox der RWTH Aachen [3]. Zusätzlich zu Legoteilen wurde eine IP-Webcam in Form eines Handys sowie eine kleine Batteriebox verbaut.

## II. Vorbetrachtungen

Die Idee zu dem Spinnenroboter hat sich aus zwei Teilideen zusammengesetzt. Diese werden hier kurz vorgestellt.

### A. Roboter mit Zielerkennung

Bereits im vorherigen Jahr haben Studenten im Rahmen des Lego-Projektes einen Roboter gebaut, der sich selbstständig auf ein Ziel ausrichtet. Dieser ist in Abbildung 1 zu sehen. Das wurde damals mit Ultraschallsensoren zur Zielerfassung umgesetzt. Ultraschallsensoren haben nur ca. 25 cm Reichweite. Dies war uns zu Wenig, deshalb wurde anstatt der Ultraschallsensoren eine IP-Webcam verwendet.

DOI: 10.24352/UB.OVGU-2018-048 

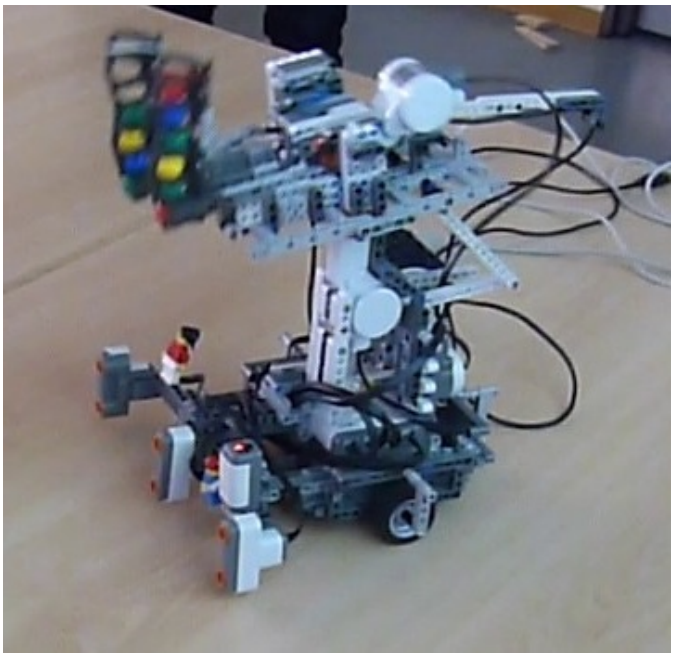


Abbildung 1. Legoroboter aus dem Mindstorms-Seminar 2017 [2]

### B. Spinnenroboter

Das Design des Roboters wurde durch den Film ”Wild Wild Westïnspiriert. In dem Film wurde von Wissenschaftlern eine sehr große schwer bewaffnete Metallspinne gebaut. In Abbildung 2 ist ein Nachbau der Spinne mit Lego zu sehen. Diese ist aber nicht mechanisiert.

## III. Hauptteil

### A. Konstruktion

Während der Umsetzung des Projektes wurde schnell klar, dass beim Design ein paar Abstriche gemacht werden mussten, so hat das Spider-Vehicle nur 6 anstatt der für eine Spinne üblichen 8 Beine bekommen. Diese Beine konnten auch keine Gelenke bekommen, sondern sind nur starr gerade. Gelenke hätten die Stabilität der Beine verringert und auch fehlten die Steuermöglichkeiten für diese. Ein Gesamtbild des Roboters ist in Abbildung 3 zu sehen. Bei dem Roboter war vor allem die Stabilität entscheidend, da das ganze Gewicht nur auch den 6 Beinen lag, von denen meistens nur 3 Beine auf dem Boden standen. Die anderen 3 Beine berührten den Boden meist aufgrund der Laufbewegung nicht. Die Unterseite der Füße ist mit Gummi ausgerüstet wie in Abbildung 4 zu sehen ist. Diese verhindern das Rutschen der Füße bei

Abbildung 2. Lego Spinnenroboter nachgebaut aus dem Film ”Wild Wild West”[1]

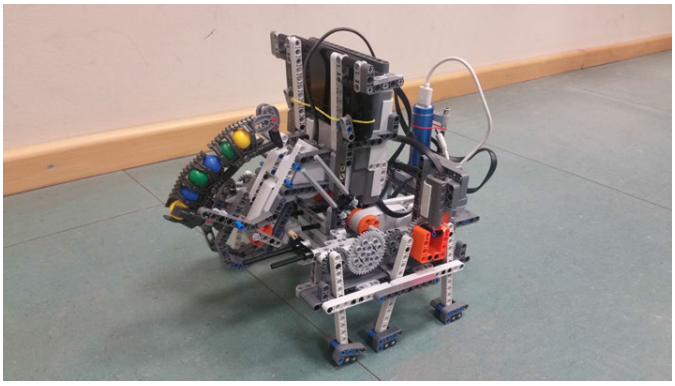


Abbildung 3. Roboter

Abbildung 4. Roboterfüße

Lauf- oder Drehbewegungen des Roboters. Auch die Präzision der Bewegungen konnte damit erhöht werden. Der nächste Punkt, der mit der Stabilität des Roboters einhergeht ist der Schwerpunkt des Roboters. Aufgrund der vorne angebauten Kanone, sowie der IP-Webcam in Form eines Handys, musste ein entsprechendes Gegengewicht geschaffen werden. Das wurde durch die Befestigung der Motoren, sowie eine kleine Batteriebox hinten am Roboter ausgeglichen. Die Batteriebox hat zudem noch den Vorteil, dass die IP-Webcam noch mit zusätzlichem Strom versorgt wird. Die Kanone ist so angebracht, dass sie in einem 45° Winkel schießt, siehe Abbildung 5. Der 45° Winkel ermöglicht Schüsse mit 150 cm Reichweite. Der waagerechte Schuss dagegen hatte nur eine Reichweite von 40 cm. Da ein NXT auf drei Motoren beschränkt ist, war es nicht möglich die Kanone im Schusswinkel Variabel zu machen. Das Ziel des Roboters wurde im Design eines Panzers

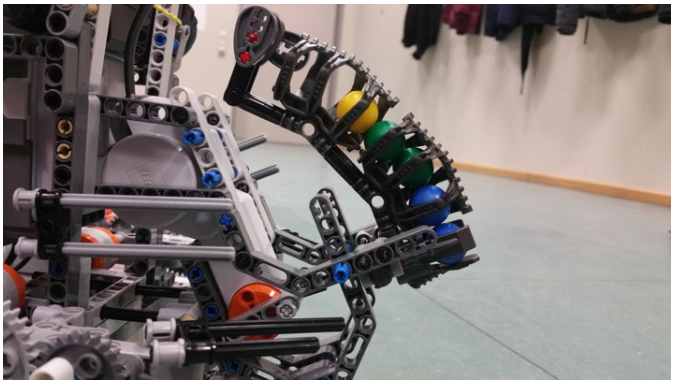


Abbildung 5. Kanone des Spinnenroboters

gestaltet wie in Abbildung 6 zu sehen ist. Diese Form wurde einerseits aus Designgründen gewählt, aber auch um die Größe des Ziels zu vergrößern und damit die Wahrscheinlichkeit das Ziel nicht zu treffen zu verringern.

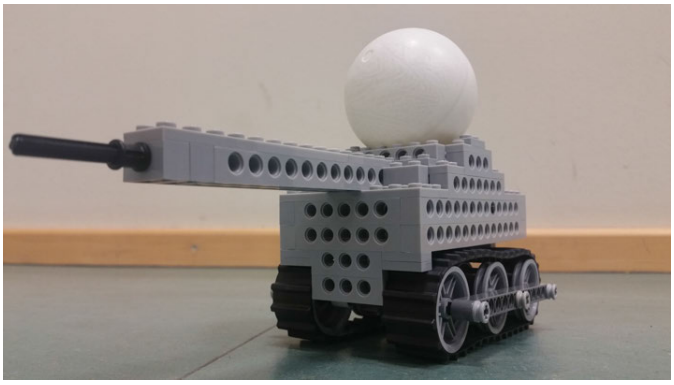


Abbildung 6. Roboterfüße

### *B. Programmierung*

In Bild 7 ist die grafische Benutzer Schnittstelle zu sehen. Links oben befinden sich die Steuerelemente nur manuellen Steuerung des Roboters. Wenn eine Bewegungsaktion aktiviert wird, bewegt sich der Roboter so lange, bis entweder eine andere Bewegungsaktion gestartet wird, oder der Roboter mithilfe des Stop-Buttons angehalten wird. Der Stopp-Button stoppt den Roboter aber nicht direkt, da dieser sich bei einem Stopp in eine stabile Standposition begibt. Durch Drehungen des Roboters ist es möglich, das die Füße eine Stellung einnehmen, in der der Roboter umkippen kann. Au der rechten Seite sieht man das Bild von der IP-Webcam. Auf dem Bild ist mit dem blauen Kreis das aktive Ziel markiert, alle weiteren

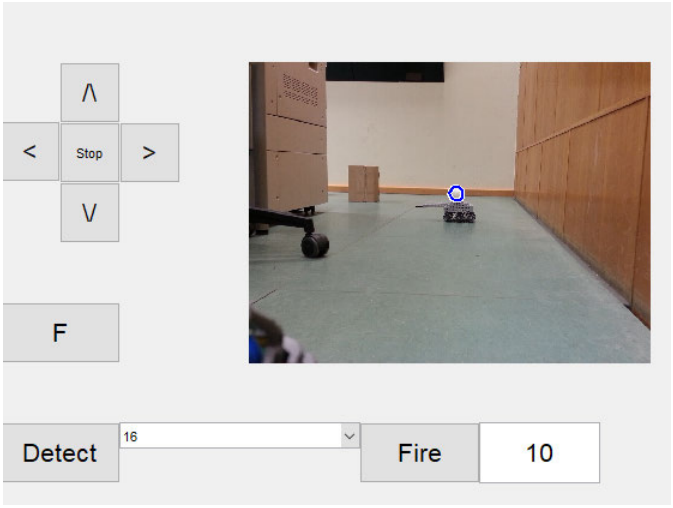



Abbildung 7. GUI

möglichen Ziele würden mit einem roten Kreis markiert werden.

Die unteren Schaltflächen sind für die automatische Zielfindung. Mit dem Detect-Button wird ein aktuelles Bild von der IP-Webcam abgerufen, alle Ziele werden auf dem Bild markiert und dann rechts in der GUI angezeigt. In dem Dropdown-Menü werden alle möglichen gefunden Ziele angezeigt. Das ausgewählte Ziel wird blau markiert. Die Namen für die Ziele sind Angaben zur Größe des Ziels, sodass die Zielauswahl dadurch erleichtert wird.

Im Input-Feld ganz rechts kann die Anzahl der Schüsse, die auf das Ziel abgegeben werden, eingegeben werden. Der letzte Button ”Fireßtartet den Algorithmus, mit dem sich der Roboter auf sein Ziel ausrichtet.

Der Algorithmus startet immer mit einem neuen Abruf von der IP-Webcam. Auf dem Bild wird das ausgewählte Ziel erneut gesucht, auf Basis von der Position und Größe des Ziels, das zuvor ausgewählt wurde.

Die Bildverarbeitung übernimmt die Imageprocessing-Toolbox in Matlab. Die Funktion $imfindcircles$ markiert alle kreisförmigen Objekte auf dem Bild und gibt deren Mittelpunkt und Radius in einer Matrix zurück. Da die Zielfindung über die Form und nicht über die Farbe läuft muss die Funktion nur zwei mal aufgerufen werden, um alle kreisförmigen Ziele zu finden.

Die Funktion wird einmal für Objekte mit hellerer Farbe als der Hintergrund und einmal mit dunklerer Farbe als der Hintergrund aufgerufen. Das Bild wird vor der Auswertung in ein Graustufenbild umgewandelt. Um nicht das Ziel bei fälschlicherweise neu gefundenen Zielen zu verlieren wird nicht das vollständige Bild ausgewertet, sondern nur ein Teil des Bildes auf Basis eines Gütekriteriums der vorherigen Bewegung. Sollte kein Ziel mit dem aktuellen Gütekriterium gefunden werden wird das Gütekriterium verschlechtert. Nach fünf Fehlschlägen in Folge wird der komplette Zielfindungsalgorithmus abgebrochen und muss neu gestartet werden.

Die Sollposition des Ziels ist die Mitte des Bildes und ist somit bekannt. Für die ersten Schritte des Algorithmus wird auch nur die x-Achse des Bildes beachtet und somit wird die Höhe des Ziels ignoriert. Sollte das Ziel sich links von der Sollposition befinden dreht sich der Roboter nach links, analog dazu, wenn sich das Ziel rechst von der Sollposition befindet, dreht sich der Roboter nach rechts. Sobald sich das Ziel im mittleren Drittel des Bildes befindet, geht der Algorithmus zu Schritt zwei über und das Gütekriterium wird verbessert, sollte das nicht der Fall sein, wird der aktuelle Schritt wiederholt.

Im zweiten Schritt des Algorithmus wird der Abstand zum Ziel angepasst. Der Abstand wird aus der Bildgröße im Vergleich zur bekannten Größe des Ziels bestimmt. Ist das Ziel zu Klein auf dem Bild, bewegt sich der Roboter auf das Ziel zu. Wieder analog dazu bewegt sich der Roboter rückwärts, wenn das Ziel zu Groß auf dem Bild ist. Falls Aufgrund der Vorwärts- oder Rückwärtsbewegung das Ziel das mittlere Drittel des Bildes verlässt, wird das Gütekriterium verschlechtert und der erste Schritt wird erneut ausgeführt. Wenn die Entfernung des Ziels innerhalb des zulässigen Bereichs liegt, ist die grobe Ausrichtung abgeschlossen, das Gütekriterium wird erneut verbessert und die Feinausrichtung beginnt.

Die Feinausrichtung ist relativ langsam. Aufgrund der Konstruktion des Roboters mit den Beinen sind minimale Bewegungen schwierig zu realisieren, da sich bei einer Drehung auch immer der Abstand der Kanone zum Boden verändert, sowie auch der Winkel der Kanone variiert je nach Position der Beine. Der Abstand der Kanone zum Boden variiert nur wenig und kann daher vernachlässigt werden. Die Variation des Winkels der Kanone muss aber beachtet werden, da die Kanone sonst sehr leicht über das Ziel schießen kann, oder auch nur den Boden vor dem Ziel treffen kann. Da der Winkel der Kanone nicht wirklich verändert werden kann, wurde ein Bereich definiert, in dem ein Treffen des Ziels möglich ist. Der Winkel wird aus dem Rotationssensor der IP-Webcam ausgelesen. Die IP-Webcam bietet eine REST-Api Schnittstelle, um deren Sensordaten im JSON-Format auszulesen. Vor jedem Abruf von der IP-Webcam, Sensordaten und Bilder, muss bei der Feinausrichtung eine kurze Pause eingelegt werden, da sonst aufgrund der Verzögerung der IP-Webcam bzw. der Übertragung ein nicht aktuelles Bild oder ein verschwommenes Bild übertragen wird.

Die Feinausrichtung ist nochmals in 3 Unterstufen unterteilt, abhängig davon wie nah das Ziel an der Sollposition ist. Auf 150 cm Reichweite mit einer Kugel mit 1 cm Radius ein Ziel zu treffen, das einen Radius von 3 cm hat. Ein Grad Abweichung ergibt auf 150 cm eine seitliche Verschiebung von 3 cm, auf Grund von der Größe des Ziels steht damit auch der Zielbereich fest, da die maximale Abweichung, mit der das Ziel noch getroffen werden kann, somit bei $+-1\%$ liegt. Da dieser Bereich sehr klein ist und aufgrund von der Konstruktion des Roboters mit 6 Beinen dreht sich der Roboter mehrfach über das Ziel, verbessert dabei aber immer die Distanz zwischen der Soll- und Istposition des Ziels. Wenn eine Ausrichtung gefunden wurde wird der Winkel der Kanone mithilfe der Sensordaten von der IP-Webcam geprüft. Sollte der Winkel außerhalb des gültigen Bereichs liegen wird versucht diesen zu verbessern, indem sich der Roboter rückwärts oder vorwärts bewegt, da dadurch sich die Zielposition möglichst wenig verändert. Wenn das Ziel trotzdem nicht mehr im Bereich $+-1\%$ liegt, wird die

Feinausrichtung fortgesetzt. Jeder Zyklus der Feinausrichtung dauert inklusive Bewegung des Roboters ca. 4 Sekunden.

Da der Datenverkehr zwischen der IP-Webcam und dem Rechner sehr hoch ist, wurde ein eigenes kleines Netzwerk für diesen Zweck eingerichtet. Durch das private Netzwerk ist es auch sichergestellt, dass niemand sonst zugriff auf die IP-Webcam des Roboters bekommt.

## IV. Ergebnisdiskussion

Das Endergebnis ist ein Spinnenroboter mit 6 Beinen. Die Gewichtsverteilung und Stabilität des Roboters konnten ausreichend optimiert werden, obwohl die baulichen Möglichkeiten begrenzt waren.

Aufgrund der baulichen Begrenzungen ist unser Spinnenroboter nicht Geländetauglich. Die Schritthöhe ist dafür zu niedrig und auch haben die Beine keine Gelenke.

Mithilfe der GUI kann der Roboter manuell gesteuert werden, da der Roboter über Bluetooth verbunden ist und über eine IP-Webcam verfügt muss man den Roboter selbst beim Steuern nicht unbedingt im Blick haben.

Die GUI kann auch die automatische Zielerfassung starten. Diese hat beim Schießen mit hoher Zuverlässigkeit das Ziel getroffen.

Der Roboter war in der Lage die ihm gestellte Aufgabe, das Suchen und Abschießen eines Ziels, zu erfüllen.

## V. Zusammenfassung und Fazit

Im Rahmen des Legoprojektes ist es gelungen einen Spinnen-Roboter zu bauen, der in der Lage ist, ein Ziel mithilfe eines Bildes zu erfassen und dieses abzuschießen. Die Zielerfassung läuft über ein mehrschrittiges System mit einem Gütekriterium um den Rechenaufwand und somit auch die Rechenzeit zur Bildauswertung zu reduzieren. Die Bildauswertung könnte unter Zuhilfenahme eines Grafikprozessors noch weiter beschleunigt werden. Die Zielerfassung funktioniert nur für stationäre Ziele die durch einen Kreis oder eine Kugel markiert sind. Die Zielerfassung ist somit noch verbesserungswürdig, dies Würde aber über den Rahmen des Praktikums hinausgehen und auch die Motoren wären für diesen Schritt nicht gut geeignet. Allgemein war die Konstruktion des Roboters in Form einer Spinne nur wenig geeignet für eine Feinausrichtung, mit einem Kettenfahrzeug wäre eine Ausrichtung einfacher möglich gewesen, aber es wurde mit dem Spinnenfahrzeug realisiert.

## VI. Quellen

[1] Lego Mechanische Spinne von Dr. Loveless
https://i2.wp.com/farm8.staticflickr.com/7354/11898981874_beca3b88bd_z.jpg
Version: 2018.03.23

[2] Lego Roboter aus dem Mindstorm-Seminar 2017
https://i.ytimg.com/vi/E7v3r9kEaco/maxresdefault.jpg
Version: 2018.03.23

[3] RWTH Aachen Mindstorms NXT Toolbox
http://www.mindstorms.rwth-aachen.de/
Version: 2018.03.23

# LEGO Mindstorms Projekt OVGU

Julian Reek, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Abstract*—**
Wir haben es uns zur Aufgabe gemacht, einen Roboter zu entwerfen, welcher in der Lage ist, Ziele zu lokalisieren, diese als solche zu erkennen und schlussendlich unter Beschuss zu nehmen. Bei der Konstruktion & Planung traten immer wieder verschiedene Probleme (wie z.B. die genaue Zielanpeilung des Geschützes) auf, welche wir aber alle beheben konnten. Unser Ziel war es einen Mittelweg zu finden, welcher nicht zu sehr von unserer ursprünglichen Idee abweichen, auf der anderen Seite uns die Umsetzung aber nicht unmöglich machen sollte. In dieser Arbeit werden die einzelnen Schritte der Planung, die Idee dahinter, aufgetretene Probleme und die Umsetzung genauer erläutert. Weiterhin zeige ich die verschiedenen Anwendungsgebiete unseres Roboters auf.

## I. EINLEITUNG

Der Roboter ist darauf ausgelegt, sowohl im zivilen, als auch im militärischen Bereich eingesetzt zu werden. Dabei soll er verschiedene Aufgaben erfüllen und vielfältig eingesetzt werden, beispielsweise als Geländefahrzeug, Roboter für Spezialkräfte bis hin zum militärischen Kampffahrzeug. Dabei war es uns wichtig, den Roboter multifunktional zu gestalten, um ihn für eine größere Zielgruppe interessant zu machen.
Zu seinen Aufgaben, bzw. Anforderungen zählen also das Scannen der Umgebung, die automatische Erfassung von Zielen, die Durchquerung von schwierigem Gelände, der Einsatz in Kampfgebieten, sowohl die Durchsuchung von Häusern im Zuge von Polizeieinsätzen. Der Roboter muss daher vielen Anforderungen gerecht werden, darunter Echtzeit-Übertragung der Kamera- und Sensordaten, er muss über große Agilität und Stabilität verfügen, sich auch bei unebenen Gelände bewegen können und manuell steuerbar sein.
Zudem muss er über ein Geschütz verfügen (oder zumindest die Möglichkeit gegeben sein, ein solches zu montieren). Bei unserem Prototyp ist es natürlich im Rahmen des Projektes aus Lego gebaut. Darüber hinaus muss der Roboter je nach Anforderung passend ausgerüstet oder umgebaut werden können, sprich leicht zu bedienen sein. Außerdem muss er auch schnell agieren können, da dies für die Erfüllung seiner Aufgaben erforderlich ist.
Es ist auch wichtig zu erwähnen, dass er je nach Bedürfnis und Anforderung in unterschiedlicher Größe zum Einsatz kommt; ein 1m großer Roboter wäre kein besonders tauglicher Geländewagen. Unsere ursprüngliche Idee basiert auf den Nachbau einer mechanischen Spinne. [1]

## II. VORBETRACHTUNGEN

### *A. Fortbewegung*

Eines der größten Probleme welches wir zwischenzeitlich hatten, war die Frage, wie sich der Roboter fortbewegen soll, da er sich später zur Anvisierung des Zieles mit sehr kleinen Bewegungen ausrichten sollte. Ketten und Räder erwiesen sich bei näherer Betrachtung als ungeeignet. Deshalb haben wir uns für Beine entschieden. Daraus ergab sich die Idee mit der Spinne. Allerdings mussten wir aufgrund der beschränkten Anzahl von Teilen und kleineren Problemen auf eine Variante mit 6 Beinen zurückgreifen. Bei der anderen Variante war aufgrund zu vieler Zahnräder die Motorbelastung zu groß, sodass sie nur bedingt funktionierte (da uns nur 1 NXT und 3 Motoren zur Verfügung standen & einer davon schon für das Geschütz verplant war).

### *B. Geschütz*

Die eigentliche Hauptaufgabe des Roboters war es, Ziele zu beschießen, welche von uns vorgegeben wurden. Am Anfang hatten wir das Geschütz in einem 90° Winkel an dem Roboter befestigt, da dies am einfachsten zu realisieren war. Allerdings verfügte der Roboter aufgrund der nicht allzu großen Motorenleistung nur über eine geringe Schussweite. Deswegen mussten wir unseren Roboter so umkonstruieren, dass es möglich war, das Geschütz in einem 45° Winkel zu montieren. Dank dieses Umbaus wurde es möglich, das Munitionslager zu vergrößern; so konnten wir auch ein mehrfaches Schießen hintereinander realisieren. Anschließend mussten wir nur noch eine Testreihe an Versuchen durchführen und dabei die verschiedenen Faktoren wie Motorleistung und -Umdrehung variieren. Mit diesen Messergebnissen konnten wir nun den ganzen Prozess der Zielerfassung programmieren. Es wurden insgesamt 200 Versuche durchgeführt bis wir verwertbare Ergebnisse hatten.

DOI: 10.24352/UB.OVGU-2018-049 

### *C. Kamera*

Um die Umgebung zu scannen und somit auch die Ziele zu erfassen, haben wir versuchsweise Licht-, Farb- und schließlich Ultraschallsensoren zu verwenden. Leider lieferten alle nicht ausreichende Resultate, weshalb wir beschlossen, ein Handy auf dem Roboter zu befestigen und mit der Kamera die nötigen Daten via Bluetooth zu übertragen. Dies bot sogar noch den Vorteil, dass wir zusätzliche Sensoren (wie z.B. den Rotationssensor) mitnutzen konnten. Dank der besseren Bildqualität verringerte sich auch die Zeitspanne, die der Roboter zum Erkennen des Zieles benötigte. Außerdem hatten wir so nun eine Echtzeitübertragung, welche sich bei den späteren Anwendungen als sehr brauchbar erwies.

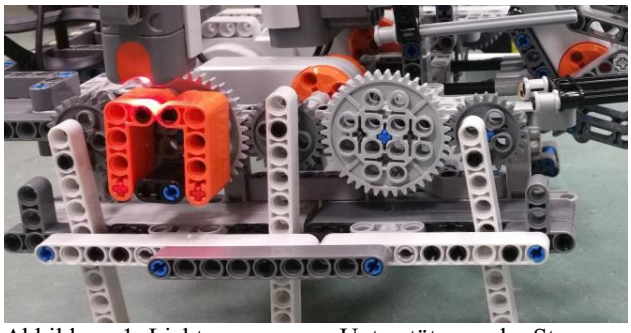


Abbildung 1: Lichtsensoren zur Unterstützung der Steuerung

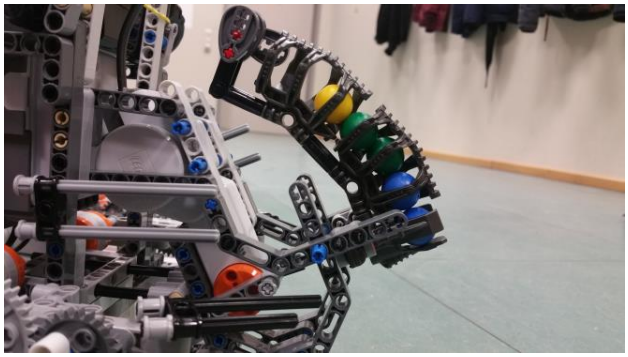


Abbildung 2: Geschütz mit vergrößertem Munitionslager

## III. HAUPTTEIL

Wir haben damit angefangen zu überlegen, welche Aufgaben unser Roboter erfüllen soll und welche Dinge wir deswegen bei der Umsetzung beachten müssen. Erst später ergab sich ein konkreter Plan. Als wir die Idee mit den Beinen und der Spinne herausgearbeitet hatten, wurde schnell klar, dass die Belastung auf den einzelnen Beinen und Zahnräder sehr groß sein wird, weswegen wir den kompletten unteren Teil des Roboters verstärkt haben.
Wir mussten dabei aber auch auf das Gewicht Rücksicht nehmen, da wir für alle Bewegungsabläufe nur 2 Motoren hatten. Deswegen haben wir auf beiden Seiten jeweils 5 Zahnräder als Verbindung der einzelnen Beine genommen. Dabei verwendeten wir abwechselnd große, damit mindestens immer 3 Beine Kontakt zum Boden hatten. Auf die Idee von meinem Partner hin haben wir für bessere Stabilität kleine Gummiuntersätze an die Füße montiert, um auch auf einer glatten Oberfläche die nötige Haftung zu bekommen. Wie bereits erwähnt, bestand unsere eigentliche Idee darin, eine mechanische Spinne zu bauen, welche auf dem Vorbild des Films „Wild Wild West“ [1] beruht. Allerdings haben wir schnell gemerkt, dass eine Variante mit 8 Beinen viele Probleme barg. So haben sich die einzelnen Füße beim Fortbewegen gegenseitig behindert, da die Gesamtgröße aufgrund von beschränkten Bauteilen begrenzt war. Als wir dieses Problem gelöst hatten, standen wir vor einem neuen. Nach dem einmaligen Benutzen des Roboters mussten wir die Beine immer wieder mühselig in Grundstellung bringen, bis uns die Idee kam, zusätzliche Lichtsensoren an die Seiten zu bauen. Diese registrieren eine bestimmte Farbe und lösen den Stopp des Motors aus. Diese Markierung steht für eine komplette Umdrehung des Zahnrads (siehe Abbildung 1: das schwarze Segment zwischen orange). Dies war programmiertechnisch nicht besonders aufwendig, ersparte uns allerdings eine Menge Arbeit. Nachdem der untere Teil fertiggestellt worden war, mussten wir überlegen wo und wie genau wir die massiveren Teile, wie NXT, Kamera und Powerbank (für eine lange Bildübertragung) anbauen können, dabei aber die Konstruktion so leicht wie möglich halten. Wir haben versucht, alles um den Schwerpunkt zu bauen, damit der Roboter nicht auf einer Seite mehr belastet wird, als auf der anderen. Nachdem das erledigt war, wurde das Geschütz konstruiert (wie genau habe ich in den Vorbetrachtungen in Punkt B erläutert). Als wir im Anschluss auch das Handy angebracht haben und via Bluetooth das Bild übertragen konnten, war die größte Herausforderung den Roboter so zu programmieren, dass er anhand der gesammelten und verarbeiteten Daten die Flugbahn selbst ermitteln konnte und sich demensprechend bewegt und ausrichtet. Dank der Beine und des Motors, welchen man in sehr kleinen Schritten steuern kann, ist es dem Roboter möglich, sich zentimetergenau aufs Ziel auszurichten. Dadurch konnten wir das Ziel schneller und leichter treffen, was in einem Kampfgebiet (späteres Anwendungsgebiet) überlebenswichtig ist. Darüber hinaus konnten wir beim Umbau des Geschützes (Anbringung in einem 45°Winkel) auch noch das Munitionslager vergrößern, was es dem Roboter nun erlaubte, Ziele nicht nur schneller, sondern auch länger unter Beschuss zu nehmen. Dieser Umbau hatte den einfachen Grund, da wir mit Einzelschüssen nicht so viele Treffer hatten und die Versuchsreihe so schneller beendet werden konnte. Das genaue Zielen und Schießen erfolgte über eine Formel der Flugbahn, welche wir mit der Höhe des Roboters kombinierten und dann in den Quellcode mit einbauten.

Allerdings musste man immer wieder kleinere Änderungen vornehmen, je nachdem in welche Höhen wir das Ziel packten (bei uns war es ein weißer Ball).

## IV. Ergebnisdiskussion

Nachdem wir mit der eigentlichen Arbeit fertig waren, nahmen wir nur noch kleine Verbesserungen vor, wie die Optimierung der Zielerfassung. Dort wollten wir mithilfe globaler Koordinaten und dem Echtzeitbild ein schnelleres Finden und Ausrichten auf das Ziel ermöglichen. Jedoch funktionierte diese Variante nur minimal besser. Als nächstes sollte der Roboter einen stabileren Aufbau bekommen, weil sich dieser aufgrund des Eigengewichtes in der Mitte zunächst leicht durchbog. Diese Verbesserung sieht man in Abbildung 2. Am Ende mussten wir nur noch testen, ob der Roboter allen Aufgaben gerecht werden konnte, für die er vorgesehen war. Bei dem Test sahen wir, dass er in der Lage war, Ziele in mehreren Metern Entfernung zu erkennen und sich darüber hinaus automatische auf diese ausrichtete. Als nächstes wurde getestet, ob der Roboter sich auch störungsfrei fortbewegen kann, da dies eines der größten Probleme war, das wir am Anfang hatten. Nach dem Umbau der Füße und der Variante mit nur 6 Beinen funktionierte auch das problemfrei. Nur eine Sache gab es noch zu beheben, nämlich die, dass wie bereits erwähnt der Roboter nach dem einmaligen Benutzen in einer zufälligen Position stehen blieb und danach die Bewegungen bei anschließenden Test sehr stockend verliefen. Das lag an dem vorprogrammierten Schrittablauf, welcher so aus dem Takt kam. Als die Lichtsensoren an den Seiten befestigt wurden, stoppte der Roboter nur dann, wenn er seine ursprüngliche Ausgangslage erreicht hatte. Der Gang wirkte nun auch flüssiger und das Handy, welches wir oben befestigt hatten, bewegte sich auch nicht mehr.

Da keine Ketten oder Räder verwendet wurden, ließ sich der Roboter auf das Ziel viel leichter ausrichten. Mit einem von uns vorher gebauten Kettenfahrzeug dauerte dieser Vorgang manchmal mehrere Minuten, was eindeutig zu lang war. Natürlich bestand weiterhin die Möglichkeit, den Roboter manuell fernzusteuern. Dies war aber nicht im unserem Interesse, weswegen wir die Idee mit den Ketten ziemlich schnell verwarfen. Zudem wollten wir etwas Einmaliges konstruieren, worin auch ein gewisser Schwierigkeitsgrad lag. Nachdem mehrere Tests erfolgreich abgeschlossen waren, befassten wir uns mit dem nächsten Problem: dem Schießen. Hierbei gab es mehrere Probleme. Zum einen hatte der Motor eine ungleichmäßige Leistung, wodurch die Kugel manchmal weiter, kürzer aber auch mal gar nicht flog, da der Motor sich aufhing und die Kugel so im Geschütz stecken blieb. Dies konnten wir lösen, indem wir im Quellcode die Umdrehungszahl des Motors von 2160 (360x6 Umdrehung für die sechs Kugeln im Lauf) auf 2880 (8 Umdrehungen) setzten. So wurden alle Kugeln abgefeuert, auch wenn sich der Motor einmal aufhängte. Das Problem der ungleichmäßigen Leistungen des Motors bestand aber weiterhin. Deswegen haben wir entschieden, dass bei allen Testläufen sämtliche Kugeln abgeschossen wurden, da dies die Trefferwahrscheinlichkeit erhöhte und in etwa 90% der Fälle auch funktionierte. Das nächste Problem beim Schießen war, dass der Roboter manchmal mehrere Ziele erfasste, welche gar nicht vorhanden waren. Dies erklären wir uns damit, dass wir den Roboter auf weiße, kreisförmige Ziele (in unserem Fall die besagte weiße Kugel) programmiert hatten. So erkannte dieser auch manchmal auf weißen oder zu hellen Flächen mit niedrigem Kontrast zu anderen Farben mögliche Ziele. Dieses Problem konnten wir beheben, indem festgelegt wurde, dass Ziele, die zu weit oder kurz entfernt, oder schlichtweg zu groß oder zu klein waren, nicht als Ziele markiert werden sollten.

Als letztes gab es nur noch ein Problem und das war der hohe Stromverbrauch des angebrachten Handys. Da wir beinah täglich den Roboter über mehrere Stunden testeten war das Handy auch immer an, wodurch es viel Energie verlor. Zuerst kam der Gedanke, es dauerhaft an ein Ladekabel anzuschließen, doch das Kabel behinderte die Testläufe und so kam es zu der Idee mit der Powerbank. Allerdings mussten wir, wie bereits erwähnt, auf das Gewicht und die Platzierung achten. Außerdem gingen uns zum Schluss allmählich die Teile aus, um auch noch eine Powerbank anzubauen. Wir mussten uns letzten Endes noch zusätzlich Teile borgen, um dieses Problem zu beheben. Nachdem der Roboter nun einsetzbar war, wurde noch die Benutzerfläche überarbeitet, mit dem Ergebnis, dass der Nutzer den Roboter jetzt auch leicht fernsteuern konnte und direkt daneben die Liveübertragung vom Roboter sehen konnte.

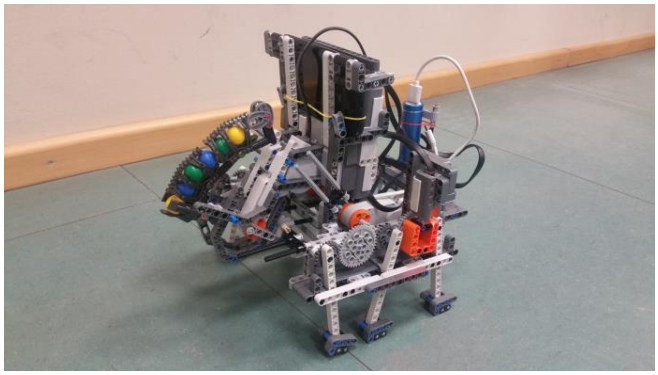


Abbildung 3: Roboter kurz nach dem letzten Umbau

## V. ZUSAMMENFASSUNG UND FAZIT

Zusammenfassend kann man sagen, dass es uns gelungen ist, einen multifunktionalen Roboter nach unseren Vorstellungen zu bauen, welcher in vielen Situationen eingesetzt werden kann. Er ist in der Lage, sich in alle Richtungen zu bewegen, scannt seine Umgebung, erkennt automatisch seine Ziele und kann sich letztendlich auch auf diese ausrichten und sie beschießen. Dies ist gelungen, durch die Benutzung von Beinen als Fortbewegungsmittel (nicht wie eigentlich geplant mit Ketten oder Rädern), durch die Verwendung und Anbringung des Handys, welches zur Echtzeitübertragung der Kamera und –Sensordaten benötigt wird und durch das Montieren eines Geschützes, welches Ziele treffen kann, die bis zu 1,70m entfernt sind. Zudem ist der Roboter in der Lage, bis zu 6 Kugeln hintereinander auf ein erfasstes Ziel abzufeuern. Zudem speichert er alle Bewegungen ab, damit er immer weiß, wo er sich gerade befindet. Was unserer Ansicht nach noch fehlte, war die Funktion, dass der Roboter in der Lage ist, Hindernisse zu erkennen und ihnen eigenständig ausweicht, um dann erneut seine Umgebung zu scannen. Dafür verblieb uns nicht genug Zeit. Was auch noch fehlte, war die Möglichkeit, den Roboter auf natürlichem, unebenen Gelände zu testen, um zu wissen, was man noch verbessern könnte.

Eine weitere Idee bestand darin, den Roboter an der Seite und oben mit sich bewegenden Ketten zu schützen, damit dieser im Falle eines Sturzes keinen zu großen Schaden nimmt und sich auch alleine wieder aufrichten kann. Zudem wäre es wichtig, bei einer größeren Version (als Geländewagen dienlich) zusätzliche Sitzmöglichkeiten für Fahrer einzubauen. Das würde Sinn machen, wenn man die Idee weiter verfolgen und ein solches Fahrzeug auch wirklich zu bauen würde.

## LITERATURVERZEICHNIS

[1] Barry Sonnenfeld Produzer, “Wild Wild West”, 1999

# Tracking Roboter FT18

Florian Miegel, Elektrotechnik und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Untersuchungsgegenstand dieses Papers ist ein LEGO-Roboter, der FT18, dem ein Menschenverfolgungs-Algorithmus zugrunde liegt. Mithilfe einer Logitech-C270-Webcam werden die Bilder auf eine Zielfarbe untersucht. Ketten und der mobil gelagerte Kopf (Kippstuhl) werden so angesteuert, dass das Ziel in der Bildmitte verbleibt, während der Roboter auf das Ziel zufährt. Zusätzlich werden Anwendungsbereiche für assistive Roboter im Dienstleistungssektor definiert.**

***Schlagwörter*—Aktives Sehen, Bildverarbeitungssystem, Color Space Thresholding, Mensch-Roboter-Interaktion, Service-Roboter, Ziel-Tracking**

## I. Einleitung

In einer Zeit, die durch den demographischen Wandel geprägt ist, besteht heutzutage trotz technischen Fortschritts ein Fach- und Pflegekräftemangel. In der Industrie wird der technische Fortschritt durch wirtschaftliche Interessen vorangetrieben. Roboter eignen sich zur Kompensation des Fachkräfte- und Personalmangels. Um die Automatisierung im Dienstleistungssektor weiterführend zu etablieren, müssen die Maschinen außerhalb des sekundären Sektors weiterentwickelt werden. Ziel ist also das Lösen von Problemen im sozialen Sektor mithilfe der Robotik. Als Lösungsansatz wurde daher der Roboter FT18 im Rahmen des Lego-Mindstorms-Projektes entwickelt.

Die frühesten wissenschaftlichen Auseinandersetzungen mit der Thematik reichen bis in die 1980er Jahre zurück. Die Relevanz des „aktiven Sehens“ [1] wird 1988 schon analysiert. Eine weitere Schwierigkeit besteht im Extrahieren der relevanten Informationen aus den Datenmengen, die beim aktiven Sehen aufgenommen werden. Die Rechenleistung soll dementsprechend auf das Extrahieren relevanter Informationen konzentriert werden und irrelevante Informationen sollen ausgeblendet werden [2]. Diese relevanten Informationen werden bei FT18 als Farben definiert. Der Roboter bedient sich also einer renommierten Methode der Farbsegmentierung, nämlich des „Color Space Thresholding“ [3].

## II. Vorbetrachtungen

Um im Dienstleistungssektor Arbeiten zu übernehmen, reicht das aktive Sehen [1] allein natürlich nicht aus. Durch Kombination mit anderen Funktionen können allerdings unterstützende Funktionen wahrgenommen werden.

### A. Assistenzbot Flo

Mit ähnlichen Zielen wurde im Jahr 2000 „Flo“ für den Bereich der Alterspflege entwickelt [4]. Eine fahrbare Plattform, ausgestattet mit zwei Kameras, 16 Ultraschallsensoren, Onboard-PCs und ausgefeilten Algorithmen, um Arbeiten der Pflegekräfte unterstützen zu können. Problematisch sind hier Baukosten, der Stand der Technik und die „soziale Kompetenz“, also die Probleme der Bedienung durch Laien bei Spracherkennung und Benutzeroberfläche.

### B. Assistenzbot SPENCER

Im Rahmen eines aktuellen Projektes wird State-of-the-Art-Technologie in SPENCER verbaut, einer Service-Plattform, die im Flughafen Anwendung finden soll [5]. Die wissenschaftlichen Auseinandersetzungen mit der Thematik können als Zielvision für den LEGO-Tracking-Roboter angenommen werden. Neben Personenerkennung werden hier natürlich von Kartografierung bis zur Analyse von Gruppendynamiken keine Funktionen ausgelassen. Die optimierte Programmierung im Bereich der Personenerkennung wird in der Ergebnisdiskussion als Vorbild genutzt werden.

### C. Tracking Roboter FT18

Flo und SPENCER sind zwei von zahlreichen wissenschaftlichen Projekten im Bereich der Robotik, welche verschiedene Technologiestadien abbilden. Die Funktion, die durch den Tracking-Roboter FT18 realisiert werden soll, ist das Erkennen und Verfolgen eines Ziels durch die Fähigkeit des Sehens und die Fortbewegung über elektronische Antriebe.

## III. Hauptteil

### A. Mechanik

Das Grundgerüst des Tracking-Roboters wurde als Kettenfahrzeug ausgelegt, um eine möglichst hohe Manövrierfähigkeit zu gewährleisten. Durch das kompakt-stabile Design mit weit nach unten verlagertem Schwerpunkt werden Vibrationen bei der Bewegung ausreichend kompensiert. Eine Webcam ist auf einem Kippstuhl befestigt. Dieser Kippstuhl ist durch einen Motor in einem festgelegten Bereich ansteuerbar. Außerdem ist ein Ultraschallsensor zur Hinderniserkennung im Einsatz. Dem Roboter wurde intern ein Koordinatensystem zugrunde gelegt, dabei wird die Vorwärtsbewegung als Bewegung in x-Richtung festgelegt. Die z-Achse deutet vom Mittelpunkt des Roboters nach oben (siehe Abbildung 1).

### B. Elektronik und Programmierung

Die Sensorik und Aktorik von FT18 wird durch einen NXT, einen Steuerungscomputer der Lego-Mindstorms Produktserie, angesprochen. Programmiert wurde der Computer mit Hilfe von MATLAB R2016a.
Der Start der grafischen Nutzeroberfläche leitet die Initialisierung des Roboters ein. Der Startbefehl gibt die dargestellte Befehlskette frei (siehe Abbildung 2).

DOI: 10.24352/UB.OVGU-2018-050 

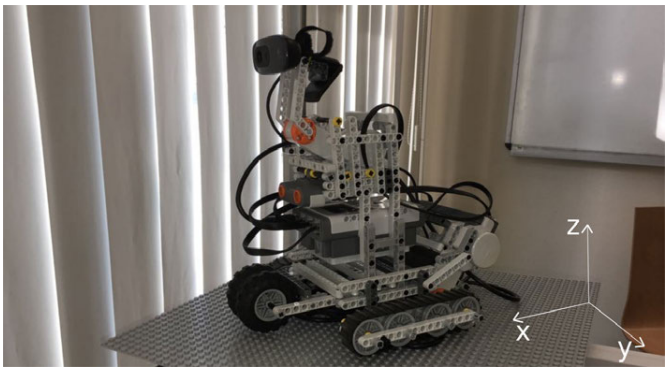



Abbildung 1. Trackingroboter

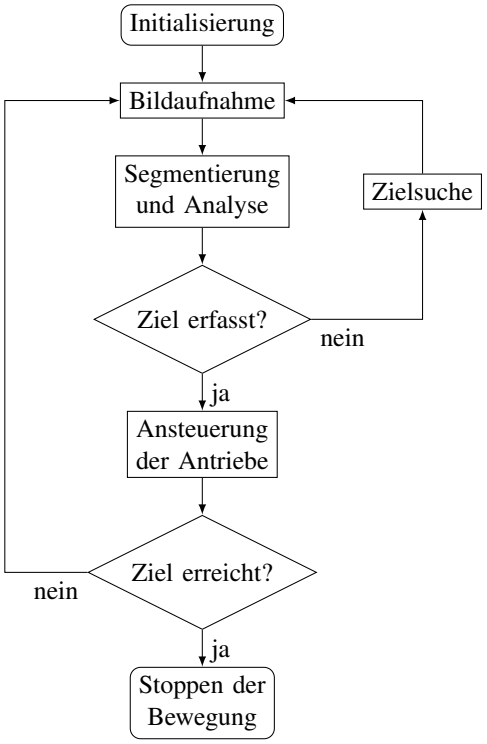



Abbildung 2. Programmablauf

*1) Bildverarbeitung:* Die Webcam Logitech-C270 ist für 30 fps ausgelegt. Drei dieser Bilder werden pro Sekunde abgefragt und ausgewertet (siehe Abbildung 3 links). Ein aufgenommenes Bild wird nach der Methode des „Color Space Thresholding“ [3] gefiltert und auf einen Farbbereich reduziert (siehe Abbildung 3 rechts). Auf der Basis wissenschaftlicher Untersuchungen wurde das HSV-Farbspektrum als sehr effektive Grundlage der Farbsegmentierung verwendet [6].

Nach vollendeter Segmentierung wird das Bild auf binäre Informationen, „Zielfarbe“ und „Nicht-Zielfarbe“, in den einzelnen Bildpunkten reduziert. (siehe Abbildung 4 links). Diese Informationen werden gewichtet und daraus ein „Zielmittelpunkt“ errechnet, der das Bild in vier Segmente mit der jeweils gleichen Anzahl an Farbpixeln zerlegt (siehe Abbildung 4 rechts).

Abbildung 3. Farbsegmentierung

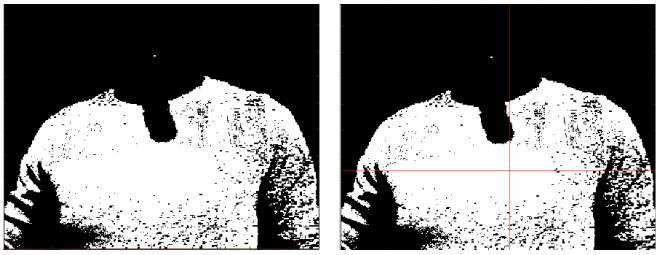


Abbildung 4. Binärdaten

*2) Zielerkennung:* Um die erforderliche Rechenleistung zu minimieren, muss ein möglichst einfacher Algorithmus zur Bildauswertung genutzt werden: Wenn die Gesamtanzahl an Farbpixeln einen Schwellenwert überschreitet, wird der Trackingprozess eingeleitet. Wird der Schwellenwert unterschritten, so wird der dargestellte Prozess der Zielsuche eingeleitet (siehe Abbildung 5).

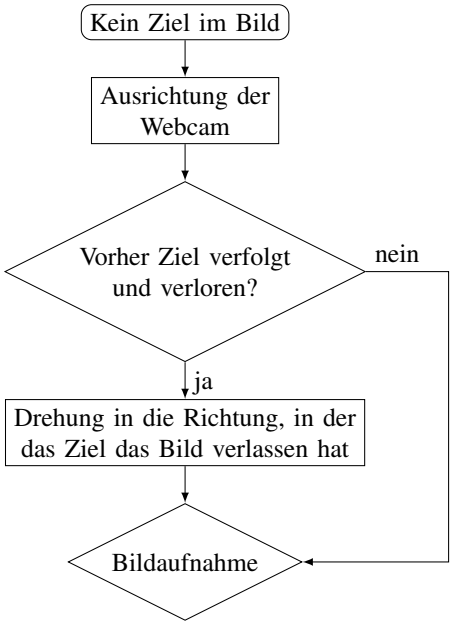



Abbildung 5. Zielsuche

*3) Regelung der Antriebe:* Die Linse der Webcam ist nah an die Rotationsachse des Roboters angeglichen worden, sodass der Bildmittelpunkt als Referenzpunkt eingesetzt werden kann. Demnach kann aus der Differenz der Koordinaten von Bildmittelpunkt und Zielmittelpunkt ein Fehler in

y-Richtung und ein Fehler in z-Richtung errechnet werden. Um aktives Sehen [1] zu gewährleisten, muss die Kamera auf den Bildbereich mit den meisten relevanten Informationen ausgerichtet werden.

Die Motorenregelung wird in Abhängigkeit von der Größe des Fehlers gestuft. Insgesamt gibt es drei Regelungsstufen. Bei großem Fehler, der dritten Regelungsstufe, bleibt eine der Ketten stehen und die andere Kette wird mit einem, von dem Fehler linear abhängigen, Spannungswert angesteuert. Ist der Fehler relativ gering, so werden beide Ketten angesteuert und eine Drehung erfolgt durch unterschiedliche Ansteuerung der Motoren. Wie auf der höchsten Regelungsstufe ist auch hier eine lineare Abhängigkeit von der Größe des Fehlers implementiert. Bei Stufe 1 unterschreitet der Fehler einen Schwellenwert und beide Antriebe werden mit maximaler Spannung versorgt.

Die Drehung des Kippstuhls ist weniger komplex. Ist das Ziel zu weit vom Bildmittelpunkt entfernt, wird die Kamera um einen konstanten Winkel nach oben bzw. nach unten korrigiert. Die Endlagen des Kippstuhls werden bei der Initialisierung des Roboters automatisch berechnet.

*4) Ziel:* Beendet wird der Verfolgungsvorgang durch den „Stop“-Befehl oder durch Erreichen des Ziels, was durch das Unterschreiten einer Distanzschranke des Ultraschallsensors realisiert wird. Mit dem Erreichen des Ziels hat der Roboter seine Aufgabe erfüllt

*5) Bedienung:* Gesteuert wird der Roboter mit Hilfe einer grafischen Nutzeroberfläche. Die „Start“- und „Stop“-Befehle können über diese Oberfläche eingegeben werden. Außerdem können diverse auftretende Störungen quittiert werden (siehe Abbildung 6).

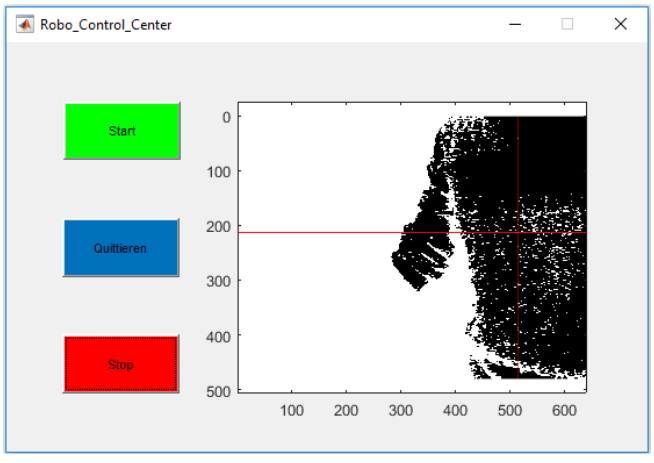



Abbildung 6. Grafische Nutzeroberfläche

## IV. Ergebnisdiskussion

Der entwickelte Trackingroboter FT18 erfüllt seine vordefinierte Funktion im Rahmen eines studentischen Projektes unter konstanten Lichtbedingungen und anderweitig unter Laborbedingungen.

Allerdings ist die Programmierung nicht auf die vorhandene Rechenleistung von Mikroprozessor des NXTs und der CPU des Computers angepasst. Die Begrenzung auf drei ausgewertete Bilder pro Sekunde löst starke Überschwingungen bei der Richtungsvorgabe aus, die im gegebenen Zeitrahmen nur durch eine Verlangsamung des Regelprozesses kompensiert wurden. Eine Veränderung der Lichtverhältnisse führt zu starkem Bildrauschen (siehe Abbildung 3), welches zum Verlust des Zieles führen kann. Zur Verminderung des Rauschens ist eine stärkere Einschränkung des Farbbereichs [7] denkbar. Um das Problem zu umgehen, können auch RGB-D-Daten und HOG-Daten erfasst werden, bei denen eine Bildtiefenanalyse, eine Analyse des Bodens und der Größe der im Sichtbereich vorhandenen Objekte und eine Untersuchung der Objekte im Speziellen durchgeführt werden [5], [8]. Im Vergleich mit diesem technisch fortgeschrittenen Algorithmus, mit dem SPENCER Personen erkennt, ist die Verwendung des „Color Space Thresholding“ störanfälliger, aber weniger rechenintensiv.

Ein letzter Kritikpunkt ist die Funktionsarmut. Die Fähigkeit ein Ziel zu erkennen und zu verfolgen reicht nicht aus, um einfachste Unterstützungsarbeiten außerhalb von Laborbedingungen zu übernehmen. Selbst als mobiler „Packesel“ ist er nicht geeignet,da ohne starke Geschwindigkeitseinbußen ein Beladen mit Alltagslasten nicht möglich ist.

## V. Zusammenfassung und Fazit

Im Rahmen des Lego-Mindstorms-Projektes wurde ein Lego-Roboter errichtet, der die Fähigkeit der Zielerkennung, Zielfixierung und der Zielverfolgung besitzt. Die mechanischen Ressourcen wurden bei der Errichtung des Roboters fast vollständig ausgelastet und harmonieren mit der einprogrammierten Funktionalität. Das Grundmodell des Roboters ist auf vielen verschiedenen Ebenen um Funktionen und Hardware erweiterbar. Die stark begrenzte mechanische Belastbarkeit der LEGO-Bausteine begrenzt allerdings mögliche Einsatzgebiete stark. Zum Erlernen der Programmierung einer Erkennungssoftware könnte ein Baukasten mit den nötigen Materialien und Bauanleitung in Bildungseinrichtungen verwendet werden.

Um sich mit hochintelligenten Robotern, wie den Zylonen [9], biomechanischen Daleks [10] und Weiteren, wie sie die Filmindustrie zu bieten hat, vergleichen zu können, reicht selbst eine Erweiterung um State-of-the-Art-Technologie [5] nicht aus. Es muss noch erhebliche wissenschaftliche Arbeit in die Robotik-Branche investiert werden, um Roboter zu entwickeln, die zuverlässig Menschen im Dienstleistungssektor unterstützen können.

## LITERATURVERZEICHNIS


[1] BAJCSY, Ruzena: Active perception. In: *Proceedings of the IEEE* 76 (1988), Nr. 8, S. 966–1005

[2] CROWLEY, James L. ; BEDRUNE, Jean M. ; BEKKER, Morten ; SCHNEIDER, Michael: Integration and control of reactive visual processes. In: *European Conference on Computer Vision* Springer, 1994, S. 47–58

[3] BRUCE, James ; BALCH, Tucker ; VELOSO, Manuela: Fast and inexpensive color image segmentation for interactive robots. In: *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on* Bd. 3 IEEE, 2000, S. 2061–2066

[4] ROY, Nicholas ; BALTUS, Gregory ; FOX, Dieter ; GEMPERLE, Francine ; GOETZ, Jennifer ; HIRSCH, Tad ; MARGARITIS, Dimitris ; MONTEMERLO, Michael ; PINEAU, Joelle ; SCHULTE, Jamie u. a.: Towards personal service robots for the elderly. In: *Workshop on Interactive Robots and Entertainment (WIRE 2000)* Bd. 25, 2000, S. 184

[5] TRIEBEL, Rudolph ; ARRAS, Kai ; ALAMI, Rachid ; BEYER, Lucas ; BREUERS, Stefan ; CHATILA, Raja ; CHETOUANI, Mohamed ; CREMERS, Daniel ; EVERS, Vanessa ; FIORE, Michelangelo u. a.: Spencer: A socially aware service robot for passenger guidance and help in busy airports. In: *Field and service robotics* Springer, 2016, S. 607–622

[6] SURAL, Shamik ; QIAN, Gang ; PRAMANIK, Sakti: Segmentation and histogram generation using the HSV color space for image retrieval. In: *Image Processing. 2002. Proceedings. 2002 International Conference on* Bd. 2 IEEE, 2002, S. II–II

[7] SIDENBLADH, Hedvig ; KRAGIC, Danica ; CHRISTENSEN, Henrik I.: A person following behaviour for a mobile robot. In: *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on* Bd. 1 IEEE, 1999, S. 670–675

[8] JAFARI, Omid H. ; MITZEL, Dennis ; LEIBE, Bastian: Real-time RGB-D based people detection and tracking for mobile robots and head-worn cameras. In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on* IEEE, 2014, S. 5636–5643

[9] LARSON, Glen A. ; THURSTON, Robert: *Battlestar Galactica*. Berkley Trade Pub, 1978

[10] PARSONS, Paul: *The Science of Doctor Who*. Icon, 2006

# Bau eines Tracking Roboters

Thorben Krause, Elektrotechnik und Informationstechnik

**Tracking Roboter— In diesem Paper wird der Bau und die Programmierung eines Lego Tracking Roboters thematisiert. Ein Farbverfolgungs-Algorithmus wertet die Bilder einer Logitech C270 Webcam aus und reguliert so die Ansteuerung der drei Motoren. Diese treiben jeweils separat zwei Ketten und den Kippstuhl der Webcam an, damit sich das Fahrzeug auf sein Ziel zubewegt. Zusätzlich definieren wir Anwendungsbereiche, die für den Roboter denkbar sind.**

***Schlagwörter*— Automatisierungstechnik, Bildverarbeitungs-system, Color space thresholding, Autonomes fahren, Rettungsroboter, Ziel-Tracking**

## I. Einleitung

In der heutigen Zeit schreitet die Technik immer schneller voran. Viele Aufgaben, die für den Menschen zu gefährlich sind, sollen von Robotern übernommen werden. Auch in der Automatisierungstechnik wächst die Nachfrage nach mechanischen „Arbeitskräften“, da diese präzise arbeiten und nicht ermüden. Ähnlich verhält es sich in der Automobil-Industrie. Autonom fahrende Autos sollen den Straßenverkehr sicherer gestalten, da so menschliches Versagen als Unfallquelle ausgeschlossen werden kann.

In all diesen Anwendungsbereichen kristallisiert sich eine Problematik heraus: Roboter müssen in der Lage sein, verschieden Situationen zu analysieren und daraus folgend eine Entscheidung treffen, wie auf die entsprechende Situation reagiert werden soll.

Fehlentscheidungen können hierbei dramatische Personenschäden oder auch finanzielle Schäden verursachen. Ein präzise arbeitender Algorithmus ist daher unerlässlich. Diese Problematik hat uns dazu veranlasst, einen Algorithmus zu schreiben, der eingehende Informationen verarbeitet, auswertet und so Entscheidungen trifft, damit der Roboter seine Aufgabe erfüllen kann. Bei unserem Roboter ist diese Aufgabe ein bestimmtes Ziel zu verfolgen.

## II. Vorbetrachtungen

Um in den beschriebenen Anwendungsbereichen arbeiten zu können, ist mehr als nur ein Farbverfolgungs-Algorithmus notwendig. Jedoch kann dieser mit anderen Funktionen kombiniert werden.

### *A. Projekt Tracloc*

[1] Nach dem Einsturz eines Gebäudes sinkt die Überlebenswahrscheinlichkeit der Verschütteten mit jeder Minute. Das Suchen in den Trümmern ist für die Rettungskräfte sehr gefährlich. Hier setzt das im September 2010 gestartete Projekt “Traloc” der ETH Zürich an. Sechs Studenten entwickelten einen Raupenförmigen Roboter, welcher mit zwei Wilde-VGA-CMOS-Kameras ausgestattet ist. Er besteht aus fünf gleichen Segmenten. An jeder Seite befindet sich eine Kette. So ist eine optimale Fortbewegung im Trümmerfeld möglich.

### *B. Industriekamera „USB 3 uEye CP Rev. 2*

[2] Die „USB 3 uEye CP Rev. 2“ von der Firma IDS Imaging Development Systems GmbH findet unzählige Anwendungen in der Industrie. Vor allem die Qualitätskontrolle kann von der Kamera übernommen werden. Beispielsweise in der pharmazeutischen Packmittelkontrolle. Die Kamera erkennt Farbe, Größe und Form. So kann Ausschussware aussortiert werden.

### *C. AutoX*

[3] Autonomes fahren ist in der heutigen Zeit ein großes Thema. Viele Unternehmen arbeiten an der Technik und wollen zeitnah solche Fahrzeuge auf den Markt bringen. Aufgrund preisintensiver und hochmoderner Sensoren ist die Entwickelung sehr kostenintensiv. Hier knüpft die Arbeit von Jianxiong Xiao, einem ehemaligen Professor der Princeton University, an. Sein Startup entwickelt ein autonom fahrendes Auto, welches auf Webcams setzt. Diese sollen für 50 US-Dollar auf dem Markt erhältlich sein und somit die Kosten gering halten..

## III. Hauptteil

### *A. Mechanik*

Der Roboter steht auf zwei Ketten. Diese sind für die Fortbewegung zuständig. Jede Kette wird jeweils von einem Motor angesteuert. So wird die Manövrierfähigkeit des Roboters gewährleistet. Die Webcam befindet sich auf einem Kippstuhl, welcher durch einen dritten Motor verstellt werden kann. So ist die Bilderkennung auch in der horizontalen Ebene möglich. Ein Ultraschallsensor wurde an der Front des Roboters befestigt. Somit ist es möglich, die Entfernung zum Ziel zu messen und im Abstand von zehn Zentimetern zu stoppen. Allgemein wurde der Schwerpunkt so tief wie möglich gehalten. Zusammen mit einem Stützrad je an der Front und am Heck soll so vermieden werden, dass der Roboter umkippen kann. Die Bewegung kann mittels eines Koordinatensystems beschrieben werden. Als X-Achse wird hierbei die Fahrtrichtung beschrieben. Vom Mittelpunkt des Roboters nach oben verläuft die Z-Achse.

DOI: 10.24352/UB.OVGU-2018-051 

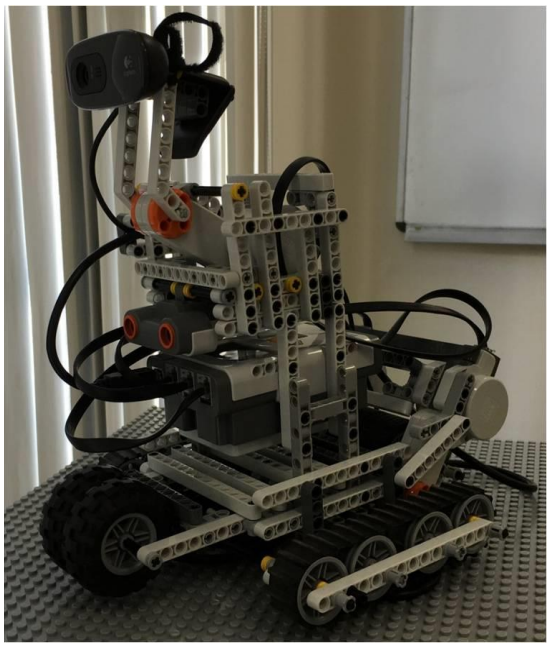


Abbildung 1: Trackingroboter

### B. *Elektronik und Programmierung*

Mit dem Start der graphischen Nutzeroberfläche wird die Initialisierung des Roboters eingeleitet. Der Startbefehl gibt die dargestellte Befehlskette frei. (Abbildung 2.)

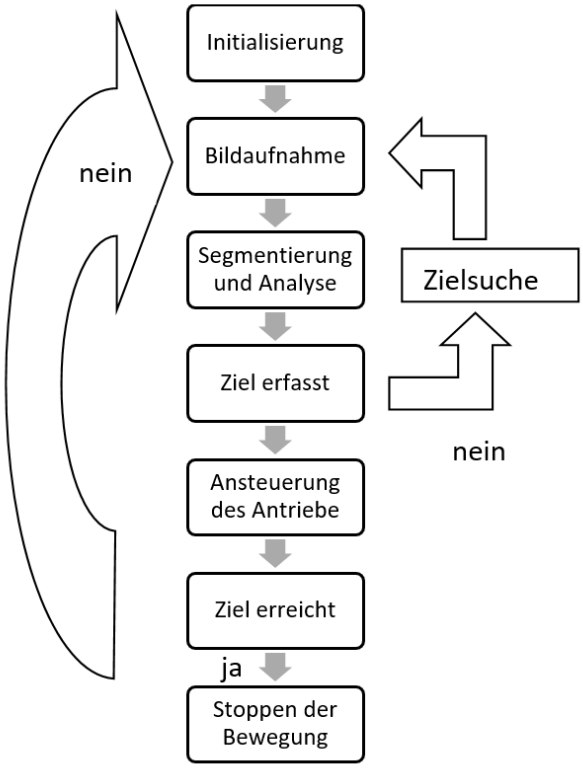



Abbildung 2: Programmablaufplan

#### 1) *Bildverarbeitung*

Für die Bildaufnahme nutzt der Roboter eine Logitech C270 Webcam. Diese ist in der Lage 30 Bilder pro Sekunde aufzunehmen. Um die Rechenzeit gering zu halten, arbeitet das Programm mit drei Bildern pro Sekunde. Die aufgenommenen Bilder werden dann mithilfe der „Color Space Thresholding“ Methode [4] gefiltert und auf einen Farbbereich reduziert. Auf Grundlage einiger Versuche kam es zu dem Ergebnis, dass sich das HSV-Farbspektrum [5] sehr gut zur Farbsegmentierung eignet. (Abbildung 3)
Das zu verarbeitende Bild wird auf binäre Informationen reduziert. Übrig bleiben nur „Zielfarbe“ und „nicht Zielfarbe“. Dieses Bild wird in vier Segmente unterteilt. Der Zielmittelunkt wird errechnet, indem die Farbpixel in gleicher Anzahl auf die Segmente aufgeteilt werden. (Abbildung 4)

Abbildung 3: Farbsegmentierung

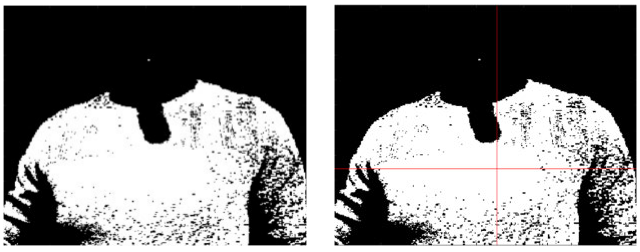


Abbildung 4: Binärdaten

#### 2) *Zielerkennung*

Wenn Anzahl der Farbpixel in einem Segmenten abweicht, wird der Trackingprozess eingeleitet. Sind keine Farbpixel im Bild zu erkennen, startet der dargestellte Zielsuchprozess. (Abbildung 5.)

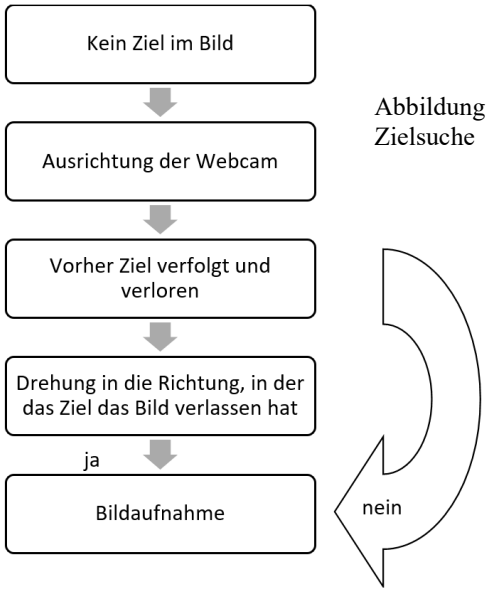



Abbildung 5: Zielsuche

*3) Regelung der Antriebe*

Die Webcam ist nahe der Rotationsachse des Roboters montiert. Somit kann der Bildmittelpunkt als Referenzpunkt verwendet werden. Dieser wird anhand eines Koordinatensystems mit dem Zielmittelpunkt verglichen. So kann ein Fehler in „y-Richtung“ und ein Fehler in „z-Richtung„ errechnet werden. Damit der Roboter seine Aufgabe optimal erfüllen kann, muss die Webcam auf den Bildbereich mit den meisten relevanten Informationen ausgerichtet werden.

Die Ansteuerung der beiden Antriebsmotoren ist von der Größe des Fehlers abhängig und kann grob in drei Stufen gegliedert werden. Ist der Fehler sehr groß, so bleibt der Roboter stehen und nur der Motor einer Kette wird mit einem linear abhängigen Spannungswert angesteuert. Bei geringem Fehler werden die Motoren beider Ketten angesteuert, jedoch mit unterschiedlichem Spannungswert, um die Richtung zu korrigieren. Der Spannungswert ist hierbei wieder linear abhängig von der Größe des Fehlers. Unterschreitet der Fehler einen bestimmten Stellenwert, werden die Motoren mit maximaler Spannung angetrieben.

Die Bedienung des Kippstuhls erfolgt auf ähnlicher Weise. Je nachdem, auf welcher Höhe sich der Zielmittelpunkt befindet, wird der Winkel des Kippstuhls nach oben, beziehungsweise nach unten korrigiert. Bei der Initialisierung wird der Kippstuhl in seine Ausgangsposition gebracht, indem er nach oben bewegt wird, bis er einen Tastsensor aktiviert.

*4) Ziel*

Der Roboter hat seine Aufgabe erfüllt, wenn er sein Ziel erreicht hat. Diese Information erhält er durch die Distanzschranke des Ultraschallsensors. Manuelles beenden der Zielverfolgung ist mit dem Befehl „Stopp“ möglich.

*5) Bedienung*

Die Steuerung des Roboters erfolgt über eine grafische Nutzeroberfläche. Diese beinhaltet des Befehle „Start“ und „Stopp“. Der Befehl „Quittieren“ beseitigt diverse auftretende Störungen. (Abbildung 6.)

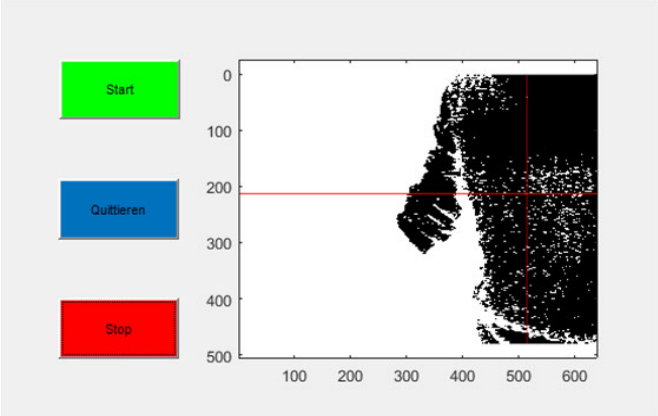



Abbildung 6: GUI

## IV. Ergebnisdiskussion

Der Trackingroboter ist in der Lage, die ihm angedachte Funktion im Rahmen der gegebenen technischen und zeitlichen Möglichkeiten des Praktikums auszuführen. Die Funktionalität ist jedoch lediglich unter Laborbedingungen gegeben. Sich ändernde Lichtverhältnisse sind hierbei der größte Störfaktor, da diese zu Bildrauschen führen. Der Verlust des Ziels ist hierbei möglich. Um das Problem zu umgehen, können auch RGB-D Daten und HOG Daten erfasst werden, bei denen eine Bildtiefenanalyse, eine Analyse des Bodens und der Größe der im Sichtbereich vorhandenen Objekte und eine Untersuchung der Objekte im Speziellen durchgeführt werden [6] [7]. Da das Programm zur Reduzierung der Rechendauer mit nur drei Bildern pro Sekunde arbeitet, kann es zur Übersteuerung bei der Richtungskorrektur kommen. Das Resultat ist eine Pendelbewegung des Roboters.

Selbstverständlich ist es dem Roboter nicht möglich, die in der Vorbetrachtung erwähnten Aufgaben zu übernehmen. Jedoch ließe sich der Algorithmus bei besseren technischen Möglichkeiten übertragen, verbessern und weitere Funktionen könnten ergänzt werden.

Im Rahmen des Praktikums entspricht der Roboter jedoch allen Anforderungen. Die Aufgabe eines lustigen Gadgets erfüllt er ohne Zweifel.

## V. Zusammenfassung und Fazit

Im Rahmen des Lego Mindstorms Projekts wurde ein funktionsfähiger Trackingroboter konstruiert und programmiert. Er ist in der Lage Ziele zu erfassen, fixieren und verfolgen. Die gegebenen Ressourcen wurden nahezu vollständig verbaut und führen, die im Rahmen der Programmierung zugeteilten Aufgaben aus. Eine Implementierung weiterer Funktionen ist durchaus denkbar und könnte den Roboter zumindest in der Spielzeugindustrie marktfähig machen.

## Literaturverzeichnis

[1] INSPECTONLINE: „Retter der Not“ *http://www.inspectonline.com/topstories/vision/retter-der-not* vom 20.03.2018

[2] IDS: „Die neue CP – unsere Highend-Kamera“ *https://de.ids-imaging.com/cp-inspiration.html* vom 20.03.2018

[3] GETMOBILITY: „AutoX: Selbstfahrendes Auto mit 50$ Webcams“ http://getmobility.de/20170420-autox-selbstfahrendes-auto-mit-50-dollar-webcams-guenstig/ vom 20.03.2018

[4] BRUCE, James ; BALCH, Tucker ; VELOSO, Manuela: Fast and inexpensivecolorimagesegmentationforinteractiverobots. In:Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on Bd. 3 IEEE, 2000, S. 2061–2066

[5] SURAL, Shamik ; QIAN, Gang ; PRAMANIK, Sakti: Segmentation and histogram generation using the HSV color space for image retrieval. In: Image Processing. 2002. Proceedings. 2002 International Conference on Bd. 2 IEEE, 2002, S. II–II

[6] TRIEBEL, Rudolph ; ARRAS, Kai ; ALAMI, Rachid ; BEYER, Lucas ; BREUERS, Stefan ; CHATILA, Raja ;

CHETOUANI, Mohamed ; CREMERS, Daniel ; EVERS, Vanessa ; FIORE, Michelangelo u.a.: Spencer: A socially aware service robot for passenger guidance and help in busy airports. In: Field and service robotics Springer, 2016, S. 607–622

[7] JAFARI, Omid H. ; MITZEL, Dennis ; LEIBE, Bastian: Real-time RGB-D based people detection and tracking for mobile robots and head-worn cameras. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on IEEE, 2014, S. 5636–5643

# Aufräumbot NXT_Bert

Jack Knobbe, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Kurzübersicht*** **- Im folgendem Dokument wird das Projekt „Aufräumbot NXT_Bert“, welches im Rahmen des LEGO-Mindstorms Seminar der Otto von Guericke Universität erarbeitet wurde, vorgestellt. Das Projekt wird in dieser wissenschaftlichen Arbeit kurz vorgestellt, die Idee dahinter, sowie die Funktionsweise und der Aufbau. Es handelt sich hierbei um ein mit Ketten gesteuerten, aus LEGO-Bauteilen konstruierten Roboter, der einfache und leichte Dinge sucht, erfasst, aufnimmt und zu einem Zielort bewegt. Dabei kann komplett auf manuelle Steuerung verzichtet werden, da der Roboter selbstständig Dinge mit Hilfe von Farben erkennt und ansteuert. Des Weiteren wird auf den Aufbau, die Programmierung und die Anwendung, sowie Probleme in der Praxis eingegangen.**

***Aufbau, Funktionsweise, Idee, Lösungen, Probleme, Zukunft***

## I. Einleitung

Jeder hatte schon mal die Situation, dass überall Sachen und Gegenstände rum stehen und liegen und die Wohnung eigentlich unbedingt aufgeräumt werden muss, einem aber schlichtweg die Motivation fehlt sich hoch zu raffen, aufzustehen oder nach einem langen Tag auf der Arbeit oder in der Uni alles da hin zu räumen, wo es eigentlich hin gehört. So hat sich die Motivation und Idee hinter dem Projekt „Aufräumbot NXT_Bert“ entwickelt. Der Roboter war Bestandteil des LEGO-Mindstorms-Praktikum der Otto von Guericke Universität Magdeburg. Die Idee hinter dem Roboter war, dass der Roboter mithilfe von Sensoren autonom in seiner Umgebung umherfahren und dabei Objekte erkennen und zuordnen können soll. Dabei sollte er noch unterscheiden, ob es sich um einen Gegenstand handelt, welchen er aufnehmen und bewege kann oder um ein Hindernis, welches er umfahren muss. Natürlich ist die Anwendung nicht nur auf den privaten Bereich beschränkt zum Aufräumen von Wohnungen und Zimmern, sondern könnte auch in größerem Umfang zum Bergen von Objekten und Menschen in Gefahrengebieten dienen. Der Ansatz war, dass der Roboter eine oder zwei Farben mithilfe eines Sensors o.Ä. erkennen soll, diese ansteuert, in einem bestimmtem Abstand stehen bleibt und ihn aufnimmt. Danach wird nach einer dritten Farbe gescannt, welche den Ablageort markiert, wo das gleiche Prinzip angewandt werden soll. Diese beiden Schritte werden wiederholt, bis alle Objekte mit den dementsprechenden Farben beiseite geschafft wurden. Um auch andere bzw. alle Objekte erkennen zu können, muss nur der Parameter auf die dementsprechende Farbe im Programm angepasst werden, welche das Objekt besitzt, welches vom Roboter angesteuert werden soll.

## II. Vorbetrachtungen

### A. *Roboter als Filmhelden*

Während der Suche nach einem möglichem Projekt stießen wir auf Bilder und Videos von Wall-E und Nachbauten des kleinen, gelben Helfers. Die Vorstellung eines sich aufklappenden, gelben Würfels, welcher sich als Aufräum- und Bergungsroboters entpuppt war leider nicht umsetzbar, doch die Grundidee und Umsetzung blieb die gleiche. So war der erste Prototyp sehr an das Design von Wall-E angelehnt, musste jedoch aufgrund des großen Greifarms und der Webcam abgeändert werden. [1] Auch in anderen Filmen, wie „I Robot” [2] mit Will Smith sind Roboter fast selbstverständliche Helfer im Alltag, welche die einfachsten und doch nötigen Aufgaben im Haushalt übernehmen und das Leben erleichtern.

### B. *Packbot510, Quince…*

Heutzutage ist der Einsatz von Robotern fast schon selbstverständlich. Die meisten Industriezweige, wie zum Beispiel der Automobilbau werden heutzutage von komplett autonomen Robotern ausgeführt, da deren Arbeit wesentlich präziser und fehlerfreier ist, als die von Menschen. Dies trifft auch auf Roboter zu, die in Gefahrengebieten, wie bei Naturkatastrophen oder bei Unfällen zum Einsatz kommen. So zum Beipsiel der japanische Packbot510, der eigentlich japanische Einsatztruppen im Zielgebiet unterstützen sollte, nun aber auch auf die Möglichkeit zum Einsatz bei Katastrophen zur Bergung von Menschen und Bewegung von Schutt getestet werden soll. So wird überlegt, ob der Greifarm zur eigentlichen Entschärfung von Bomben und Mienen auch zur Räumung von Schuttteilen genutzt werden kann. [2]

### C. *Roboscooper*

Ähnlich wie unsere Idee und Ausführung ist auch der Roboscooper von der Firma Wowwee [3] konstruiert. Dieser macht grundsätzlich das gleiche wie unser Projekt: Objekte erkennen, ansteuern und aufnehmen. [4]

DOI: 10.24352/UB.OVGU-2018-054 

Abb. 2: Aufräumroboter „Roboscooper" von Wowwee

## III. HAUPTTEIL

Zwei Ketten auf jeweils zwei Rädern mit einem Motor dazwischen, der über eine Achse die vorderen Räder antreibt und alles improvisiert befestigt, dass es hält und fährt. Dies war der erste Zusammenbau zum Testen der Aufbaumöglichkeiten. Nachdem die Aufgaben des MATLAB Handbuchs, welches uns zur Verfügung gestellt wurde abgearbeitet waren und die Idee zum Projekt gefasst war wurde erst einmal getestet und geschaut, was für Möglichkeiten bestehen einen Roboter zu bauen, der den Vorstellungen und Anwendungen der Idee entsprachen. Schnell wurde klar, dass die größte Stabilität über Ketten als „Fundament" gewährleistet wird und der Roboter so am geländegängingsten war. Schnell wurde auch erkannt, dass ein einziger Motor als Antrieb zwar grundsätzlich für die Fortbewegung reichen würde, jedoch keinerlei Steuerung möglich war. So wurde aus einem Motor zwei und aus zwei Ketten gleich vier, sodass jeder Motor zwei Ketten antrieb und über die Drehrichtung der Motoren die Bewegung in andere Richtungen als vor und zurück möglich war. Nachdem dies erkannt war wurden erste Versuche gestartet beide Motoren gleichzeitig über den NXT anzusteuern. Einen Motor anzusteuern war Bestandteil der Übungsaufgaben im MATLAB Handbuch und stellte daher keine Schwierigkeit mehr dar. Nach erfolgreicher Ansteuerung beider Motoren wurde die differentielle Bewegung der Motoren programmiert, sodass sich beide auch gleichzeitig in entgegengesetzte Richtungen drehten und die Steuerung gewährleistet war. Währenddessen wurden die beiden Bauteile bestehend aus jeweils zwei Ketten zusammengebaut, sodass ein einziges Konstrukt entstand, an das weiter an- und aufgebaut werden konnte. Zuerst wurde überlegt, wie man den NXT in den Aufbau einbinden könne, sodass die Reichweite nicht nur auf wenige Zentimeter Länge der Verbindungskabel zwischen NXT und den Motoren beschränkt war. So entstand der Aufbau eine Art Gerüst auf den Verbindungsstreben der Kettenbauteile anzubringen, in welches der NXT eingesetzt und befestigt werden konnte. So stand das Grundgerüst, an welches reintheoretisch nur noch die Sensoren und der Greifarm angebaut werden sollten. Die Ketten wurden für bessere Koordinierung auf unebenem Untergrund in eine Dreiecksform gebracht, welche später wieder verworfen wurde, da der Zusammenbau zu instabil aufgrund des Materials der LEGO-Bauteile war.

Danach war der erste Versuch der Hindernisumgehung mit Hilfe eines Ultraschallsensors, welcher die Entfernung misst, indem Ultraschallwellen, ähnlich wie bei Fledermäusen, ausgesandt werden und reflektiert werden. Die Dauer, bis die Reflektion wieder vom Sensor wahrgenommen wird bestimmt den Abstand, der zwischen dem Sensor und dem Hindernis lag. In der Programmierung wurde angegeben, dass sobald der gemessene Abstand unter einem selbst bestimmten Abstand liegen sollte, die Motoren differentiell angesteuert werden, sodass sich der Roboter um einen bestimmten Winkel auf der Stelle dreht und nach dem bestimmten Winkel weiter fährt. Der Betrag des Winkels konnte nicht über die Umdrehungen der Motoren bestimmt werden, sondern musste über die Pause bestimmt werden, welche angibt, wie lange der Befehl für die Steuerung der Motoren ausgeführt werden soll. So musste der richtige Winkel über Probieren mit verschieden langen Pausen im Quellcode ermittelt werden.

Das autonome Bewegen durch Hindernisse war somit also gewährleistet und die Steuerung sollte aber noch etwa benutzerfreundlicher gestaltet werden. So programmierte man eine GUI in welcher ein Button eingefügt wurde, welcher mit dem Befehl zur Ansteuerung des NXTs verknüpft war, sodass der Roboter gestartet werden konnte. Ein weiterer Knopf diente zum Starten der Bewegung und noch einer zum wieder beenden aller Befehle, sodass der Roboter anhielt. Später wurde eine Messtabelle eingeführt, welche Messwerte „live" vom Roboter übertrug, in welcher der Benutzer ablesen konnte, wie viel Power die Motoren momentan haben, wie viele Umdrehungen in einem bestimmten Zeitraum erreicht wurden usw. Da der Roboter aber nicht nur autonom fahren sollte, sondern auch der Benutzung in die Bewegung eingreifen können sollte, wurden der GUI Schieberegler hinzugefügt, welche die Richtung und Geschwindigkeit anpassen ließen und die Richtung bestimmen konnte, sodass der NXT_Bert nicht nur autonom fahren konnte, sondern auch manuell gesteuert werden konnte.

Die nächste Überlegung galt der Verbesserung der Objekterkennung. So entschied man sich gegen den Ultraschallsensor und befestigte eine Webcam seitlich am Roboter, welche zum tracken von Farben genutzt werden sollte. Dieses System sollte auf dem RGB-Prinzip beruhen, welches alle Farben in die drei Primärfarben zerlegt: Rot, Blau, Gelb und durch Aufaddieren der einzelnen Anteile der drei Primärfarben die zu trackende Farbe definiert. Die geschah durch folgendes Prinzip: die Kamera wurde auf eine bestimmte, möglichst prägnante Farbe gepolt, um Verwechslungen durch unterschiedliche Lichtverhältnisse oder Ungenauigkeit der Kamera möglichst auszuschließen. Die Farbe wurde über die Rot-, Grün- und Blauwerte im Quellcode bestimmt, sodass man sich für die Farbe Gelb entschied. Der Roboter bekam den Befehl sich solange im 360°-Winkel zu drehen, bis die Kamera ein prägnant gelbes Objekt erkannte. War dies der Fall ging der Befehl an den NXT raus die Motoren zu stoppen und die Pixel im Bild der Kamera zu markieren, welche die Farbe Gelb hatten. Von diesem markierten Bild aller gelben Pixel wurde der Mittelwert ermittelt. Der NXT sollte nun den Befehl bekommen den Mittelwert der gelben Pixel in das Zentrum des Kamerabildes zu fokussieren und danach die Motoren

ansteuern sich vorwärts zu bewegen. Ab einem bestimmten Abstand, sollten die Motoren stoppen und die restlichen Befehle zur Steuerung des Greifarms an den NXT ausgegeben werden. Die Abstandsermittlung erfolgte wie folgt: Die markierten gelben Pixel wurden ausgezählt. Sobald sich der Roboter nun auf das gelbe Objekt zubewegt erhöht sich die Anzahl der gelben Pixel im Bild der Kamera. Sobald eine bestimmte Anzahl gelber Pixel erkannt wird, wird ein Befehl an den NXT ausgegeben die Motoren zu stoppen und die Befehle zur Steuerung des Greifarms auszuführen.

Abb. 3: Bild der Webcam mit verschiedenfarbigen Bällen

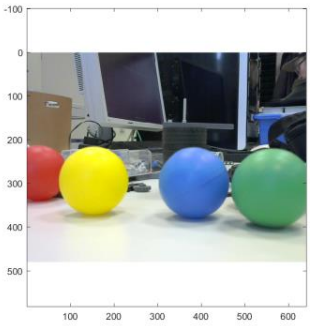
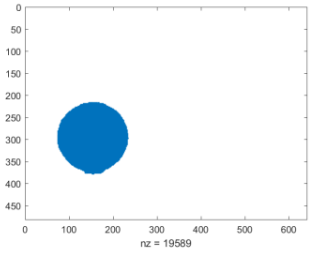
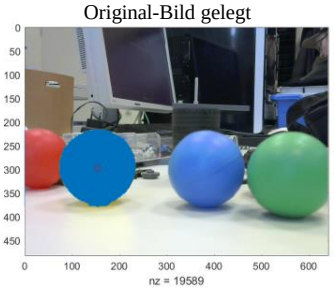


Abb. 4: Markieren der Farbe, die getrackt wurden (hier: gelb)




nz = Anzahl der markierten Pixel

Abb. 5: Bild der markierten Pixel über Original-Bild gelegt




Zu guter letzt wurde der Greifarm konstruiert und seitlich am Roboter, direkt über der Webcam montiert. Die differentielle Bewegung der Hoch- und Runterbewegung und des Auf- und Zuklappens des Arms erschien über die Steuerung eines einzelnen Motors und einer Kupplung zu kompliziert, so wurde sich für eine Steuerung mit zwei Motoren entschieden. Einer für jede Bewegungsart. Da die NXTs standardmäßig nur über drei Slots zur Verbindung mit Motoren verfügen, wurde ein zweiter NXT an die Rückseite des Aufräumbots angebracht. Danach musste nur noch die Ansteuerung zweier NXTs gleichzeitig erfolgen bis die Feinarbeit des Greifarms erfolgen konnte.

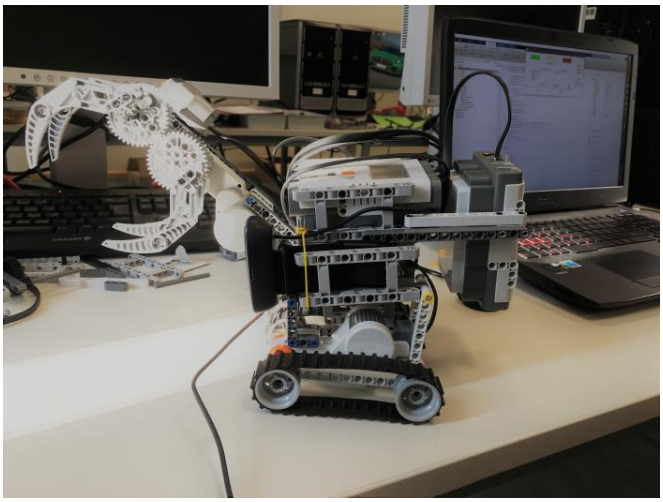


Abb. 6: Seitenansicht Aufräumbot NXT_Bert

Über einige Bedingungsverknüpfungen im Code gab der NXT im dementsprechend richtigen Moment den Befehl an den Greifarm raus, wann er hoch/runter bzw. auf/zu auszuführen hat. Durch die Verknüpfung all dieser Befehle war nun gewährleistet, dass der Roboter ein gelbes Objekt trackt, anvisiert, darauf zusteuert, im richtigen Moment davor anhält, dann die Klaue öffnet, den Greifarm herunterlässt, die Klaue wieder schließt und den Greifarm anhebt.

Genau das gleiche Prinzip wurde zum weiterem Verfahren genutzt. So drehte er sich wieder, mit dem Objekt in der Klaue, so lange um die eigene Achse, bis er die zweite vorgegebene Farbe trackte. In diesem Fall prägnant grün. So markierte er wieder die diesmal grünen Pixel, nahm den Mittelpunkt, steuerte ihn an und hielt im dementsprechendem Abstand an. Da als Ablageort eine Art Käfig gewählt wurde war es unnötig den Greifarm vor dem Ablegen zu senken, sodas nur noch der Befehl zur Öffnung der Klaue ausgeführt wurde.

Um die Steuerung des NXTs zu erleichtern und die Anwendung in der Praxis zu verbessern, wurden die beiden NXTs über Bluetooth angesteuert.

## IV. Ergebnisdiskussion

Alles in allem wich das Endergebnis von unserer ursprünglichen Idee etwas ab, was jedoch im Nachhinein nur zum Positiven zu bewerten ist, da die neuen Ideen den Anwendungsbereich des Aufräumbots erweiterten. Der Roboter tut, wofür er konstruiert wurde, sodass er mit besserem Material auch in größerem Maßstab entwickelt und zusammengebaut werden könnte.

Probleme stellt, wie bereits erwähnt das Material der Bauteile da, da dies unter größerer Belastung zu Verformungen neigt.

Zudem war es nicht möglich eine Unterscheidung von Objekten mit den verwendeten Farben vorzunehmen, sodass der Roboter alles anvisiert und ansteuern will, was zuerst dementsprechend prägnant gelb bzw. grün erscheint, so steuerte der Roboter nicht den von uns vorgesehenen grünen Käfig-Ablageort an, sondern den Bildschirm des vor uns sitzenden Kommilitonen, welcher aus vorwiegend grünen Bildern

bestand.

Weiteres Problem war die Befestigung der Webcam am Roboter, da diese nicht zum LEGO-Baukasten dazu gehörte. Die Befestigung gelang, jedoch war die Genauigkeit durch Wackeln beim Fahren nicht immer zu 100% korrekt.

Die Genauigkeit der Steuerung der Ketten war durchaus präzise, jedoch unterscheidet sich die Leistung der unterschiedlichen Motoren voneinander, da sie über die Zeit ihrer Benutzung unterschiedlich beansprucht wurden.

Das einzige Problem, welches wir zum Ende hin endgültig beheben konnten, war der Delay und die lange Verbindungszeit der beiden NXTs mit dem Computer per Bluetooth. Der Delay, der durch die kabellose Verbindung entstand war zum Glück vernachlässigbar klein, nur die Verbindungszeit mit zwei NXTs gleichzeitig betrug etwas über 40 Sekunden. Diese konnten wir jedoch umgehen, indem die Initialisierung nicht abgebrochen wurde, wenn die NXTs nicht mehr angesteuert wurden, sodass die Verbindung nur mit Drücken des NXT-Knopfes unterbrochen werden konnte.

Eine Einschränkung der Reichweite gab es trotz der Verbindung per Bluetooth, denn die Webcam gehört nicht zum Baukasten, kann somit nicht mit dem NXT verbunden werden und musste über ein USB-Kabel mit dem Computer verbunden bleiben. Natürlich ist dieses Problem ganz einfach mit der Montierung einer bluetoothfähigen Webcam zu beheben.

## V. Zusammenfassung und Fazit

Trotz einiger Probleme während der Arbeit an dem Projekt konnten wir einige schon von Anfang an verhindern, oder im Nachhinein ganz einfach eliminieren. Übrig gebliebene Probleme sind meist den Bauteilen zu verschulden, sodass auch diese in größerem Rahmen mit besseren Möglichkeiten wegfallen würden. Nur die Probleme in der Programmierung, wie das tracken der spezifischen Objekte und nicht Aller würde einen höheren Zeitaufwand mit sich bringen und stellt tatsächliche Probleme dar. Trotz allem funktioniert der Roboter wie erhofft und geplant und muss mit nur leichten Justierungsarbeiten, wie der Kamera unter dem Greifarm, in Stand zu halten. Die Bedienung ist einfach, egal ob autonom oder manuell, da bei „autonom“ nur ein Knopf gedrückt wird und der Roboter von alleine fährt und seine Aufgabe erledigt und auch bei „manuell“ sind die Buttons so beschriftet, dass jeder erkennen kann, was zu tun ist.

Dennoch kann der Roboter noch optimiert und erweitert werden. Die Ketten können stabilisiert werden, der Greifarm für höhere Genauigkeit zentral auf dem Korpus platziert werden und eine bluetoothfähige Webcam könnte in Zukunft verbaut werden. Auch die Steuerung des Greifarms über nur einen Motor mit Hilfe einer Kupplung wäre denkbar du mit mehr Zeit umsetzbar gewesen.

Somit lässt sich sagen, dass sich das Projekt auch mit einigen Verbesserungsmöglichkeiten durchaus sehen lassen kann und anwendbar ist.

## Anhang

```
frame=getsnapshot(camera);

camera.ReturnedColorSpace;
rotf=frame(:,:,3);
gruenf=frame(:,:,2);
blauf=frame(:,:,1);

for n=1:480
for m=1:640

gelb(n,m)=rotf(n,m)<gruenf(n,m)-100 &&
rotf(n,m)<blauf(n,m)-100;
end
end

nz=sum(sum(gelb));
```

Quellcode 1: Farberkennung gelb

## Literaturverzeichnis

[1]https://g.co/kgs/hdjnTJ
https://www.youtube.com/watch?v=UWBU7BSTwu4

[2]https://de.wikipedia.org/wiki/I,_Robot_(Film)

[3]http://www.robonews.de/2011/03/bergungsroboter-japan-erdbeben-einsatz/

[4]https://wowwee.com/

[5]http://www.handelsblatt.com/technik/forschung-innovation/aufraeumroboter-roboscooper-packt-den-muell-weg/3506314.html

# Bau des Aufräumroboters „BERT“ im Rahmen des LEGO-Mindstorm-Praktikums

Tommy Gaede, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Das vorliegende Paper beschäftigt sich mit Bau und Programmierung eines Aufräumroboters aus Lego. Ziel dieses Roboters ist es mithilfe eines Greifarmes und einer Webcam Zielobjekte zu erkennen, aufzunehmen und zu einem bestimmten Ort zu bringen. Dafür wurde mithilfe von MATLAB ein Bildanalyseprogramm implementiert, welches den Roboter koordiniert. Es handelt sich um ein Kettenfahrzeug mit Differentiallenkung was Stabilität und Manövrierfähigkeit ermöglicht. Des Weiteren werden die beiden Steuereinheiten des Roboters per Bluetooth angesteuert, was eine verbesserte Reichweite zwischen Laptop und Roboter sichert. Es werden außerdem Bezüge auf bereits heute existierenden Robotern im Haushalt und militärischen Robotern genommen. Schwerpunkt dieser Arbeit ist dabei die Farberkennung, welche in der Bildanalyse implementiert ist.**

***Schlagwörter* - Aufräumroboter, Bildanalyse, Farberkennung, Greifarm, Webcam**

## I. Einleitung

Wir befinden uns in einer Zeit, die durch Technologie stark geprägt ist. Schon heute sind Roboter im Haushalt eingebunden und in der Industrie sind sie in vielen Branchen nicht mehr wegzudenken. Die Entwicklung wird durch immer leistungsfähigere Hardware zunehmend schneller, sodass der Fortschritt in naher Zukunft Roboter vollends in unseren Alltag einbindet, um uns bei den täglichen Pflichten zu unterstützen. So sind Aufräumroboter schon länger ein Thema für viele Konzerne und Saug- oder Wischroboter finden in Deutschland ebenfalls mehr und mehr Anklang. Auch das Militär greift auf Roboter zurück, beispielsweise bei der Bombenräumung. Somit wird klar, dass Roboter schon heute einige Aufgaben für uns übernehmen können.

Jedoch gibt es bisher keine vollständig autonomen Roboter im freien Handel, die ein beliebiges Objekt zurück an seinen ursprünglichen Ort bringen. Diese wäre eine große Erleichterung für alte oder kranke Menschen, welche es nicht alleine schaffen, sich ohne Probleme um den Haushalt zu kümmern. So haben wir es uns zum Ziel gemacht, diesen Ansatz zu verfolgen und einen Aufräumroboter gebaut, der diese Aufgabe vereinfacht lösen kann.

Dabei spielt die Bildverarbeitung eine große Rolle. Wir nehmen die Welt über unsere Sinne wahr, so auch mit dem optischen Sinn, welcher das Gesehene über das Gehirn zu Informationen verarbeitet. Anders am Computer: Dort sendet eine Kamera das aufgenommene Bild an einen Computer, welcher dann mithilfe von uns entwickelter Algorithmen das Bild analysiert und zu Informationen verarbeitet [4]. Genau dies machen wir uns zu Nutze, um unseren Roboter mit einem digitalen optischen Sinn auszurüsten, sodass dieser sich orientieren kann.

DOI: 10.24352/UB.OVGU-2018-055 

## II. Vorbetrachtungen

### A. Roboter im Haushalt

*“Die meisten Roboterinteressierten würden sich vom technischen Helfer das leidige Staubsaugen abnehmen lassen (68 Prozent). Jeder Zweite würde den Roboter zum Fußboden wischen einsetzen. Etwas weniger sehen darin einen idealer Fensterputzer (47 Prozent). Deutlich weniger Bürger können sich Roboter zum Blumengießen (15 Prozent), zum Aufräumen (12 Prozent) und als Butler (9 Prozent) vorstellen.”*[1] Somit wird deutlich, dass Roboter einen immer größeren Stellenwert in der Gesellschaft genießen. Sie können Aufgaben im Haushalt übernehmen und uns so das Leben erleichtern.

Seit Ende der 1990er Jahre wird an voll automatischen Saugrobotern geforscht. Sie sind derzeit die beliebtesten Serviceroboter und erleichtern uns den Alltag. Die ersten Saugroboter nutzten eine einfache Federmechanik, um Kollisionen zu erkennen und den Hindernissen auszuweichen. In modernen Geräten dagegen werden bereits komplexere Sensoren, wie Ultraschall-, Infarot- oder Lasersensoren, verbaut, um diese Kollision zu umgehen [2].

### B. Roboter des Militärs

Das Militär nutzt Roboter, da diese, im Falle einer Bedrohung, ohne die Gefährdung von Menschen agieren können. Des Weiteren können sie aus der Entfernung gesteuert werden. Es gibt verschiedene Arten von unbemannten Landfahrzeugen, auf welche das Mitlitär zurückgreift, u.a. EOD-Roboter (Explosive Ordnance Disposal), welche zur Bombenentschärfung dienen, SUGV (Small Unmanned Ground Vehicle), welche zur Erkundung von Gebieten genutzt werden oder UGCV (Unmanned Ground Combat Vehicle), welche als Kampfroboter genutzt werden. Somit werden die zahlreichen Möglichkeiten, die Roboter bieten auch vom Militär ausgeschöpft, was die Forschung stark voranbringt. EOD-Roboter werden häufig in Krisengebieten genutzt, aber auch in öffentlichen Gebäuden oder Einrichtungen wie

Flughäfen, wenn eine Bombe vermutet wird. So kann sichergestellt werden, dass sich kein Mensch in Gefahr begeben muss. Diese Roboter haben uns als Vorlage für unsere Konstruktion gedient [3].

### C. *Bildanalyse durch Farberkennung*

Die Farberkennung beruht auf der Analyse der additiven Farbmischung von Objekten. So lassen sich alle Farben in ihre Grundbestandteile aus den Primärfarben zerlegen. Dieses Prinzig beruht auf einem RGB-System (rot-gelb-blau). Diese drei Farben können alle anderen Farben durch mischen erzeugen, lassen sich selbst allerdings durch vermengen nicht produzieren. Dieses Prinzip nutzen wir bei der Bildanalyse mithilfe einere Webcam. Diese legt für jeden Pixel, den die Kamera erfasst, einen Wert im RGB-Spektrum an, das heißt, es werden jedem Pixel drei Werte für seine Rot-, Gelb- und Blauanteile zugeordnet, welche zwischen 0 und 255 liegen. So können wir jede Farbe anvisieren und analysieren.

## III. HAUPTTEIL

### A. *Mechanik*

Der Aufräumroboter „BERT“ ist als Kettenfahrzeug konstruiert, um eine möglichst hohe Manövrierfähigkeit mittels der Differentiallenkung zu ermöglichen. Des weiter mussten wir, gewichtsbedingt, jeweils zwei Ketten pro Seite anbringen, da durch die Nutzung vieler Komponenten der Schwerpunkt erhöht gelegen hat und die Stabilität so gewährleistet werden kann. Allerdings erschwert dies das Manövrieren, da die äußeren Ketten schneller laufen müssten als die Inneren, was wir aber nicht umsetzen konnten. So kann es zu kleinen Ungenauigkeiten kommen. Unser Ziel war es, den Roboter so kompakt wie möglich zu halten, was uns weitestgehend gut gelungen ist. Der Greifarm, mit welchem der Roboter die Zielobjekte wegräumt, wird durch zwei Motoren betrieben, um eine Auf- und Abwärtsbewegung des Armes zu ermöglichen und zur Steuerung der Kralle. Somit waren wir auch gezwungen eine zweite NXT-Einheit zu verbauen, um vier Motoren gleichzeitig ansteuern zu können. Zur Umgebungsanalyse haben wir eine Webcam seitlich des Konstrukts, auf Greifarmhöhe angebracht (Abbildung 1).

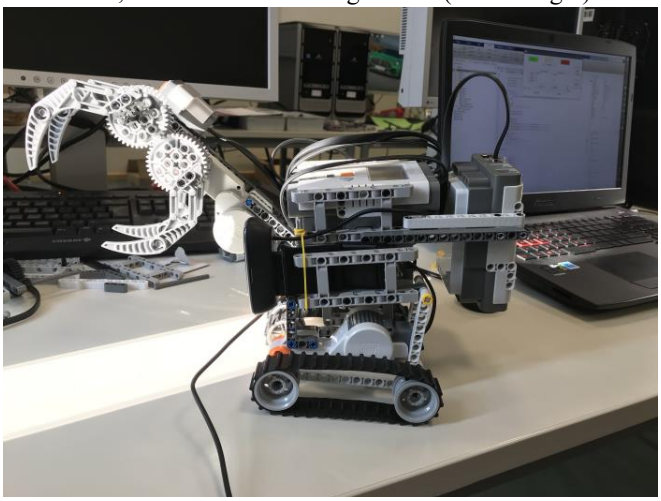


Abbildung 1: Aufbau des Roboters

### B. *Programmierung und Umsetzung*

Zum Ansteuern des Roboters per Bluetooth haben wir zunächst eine Initialisierung ausgeführt, welche vor dem Start des eigentlichen Programmes separat ausgeführt wurde, da diese mit einem Delay von ca. 40 Sekunden zu langwierig gewesen ist, um diese in den Programmablauf hinein zu beziehen. Diese verbindet den Laptop mit den Steuereinheiten (NXT-Einheiten) des Roboters. Mit dem Start der grafischen Benutzeroberfläche (GUI), kann man den dargestellten Programmablauf starten (siehe Abbildung 2).

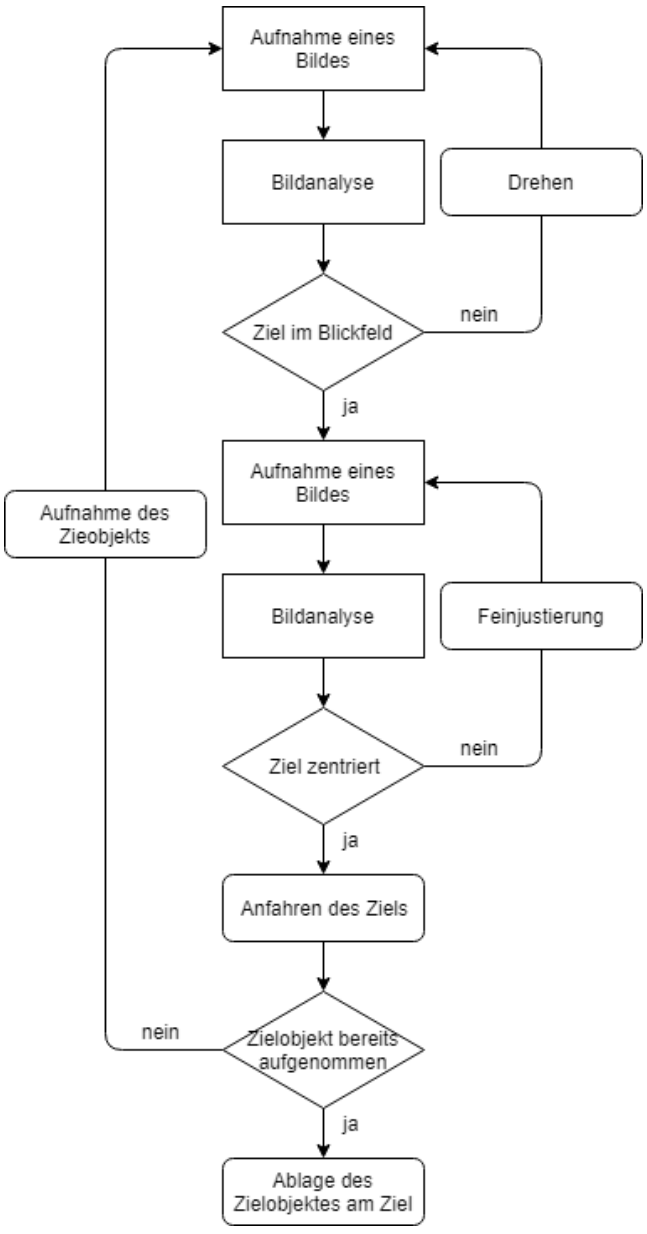



Abbildung 2: Programmablaufplan

Die Bildanalyse (siehe Quellcode A) erfolgt durch die Einbindung einer Logitech C270 Webcam. Wir haben uns bei der Auswertung der Bilder für die Methode der

Farberkennung entschieden, bei welcher besonders auffällige Farben, wie gelb, blau, rot oder grün analysiert werden. Dies ist notwendig, um eine möglichst hohe Genauigkeit zu gewährleisten. Zunächst wird dabei das Bild als Matrix mit den Maßen 640 Spalten und 480 Zeilen erstellt, da die Auflösung der Kamera 640x480 Pixel beträgt (siehe Abbildung 3).

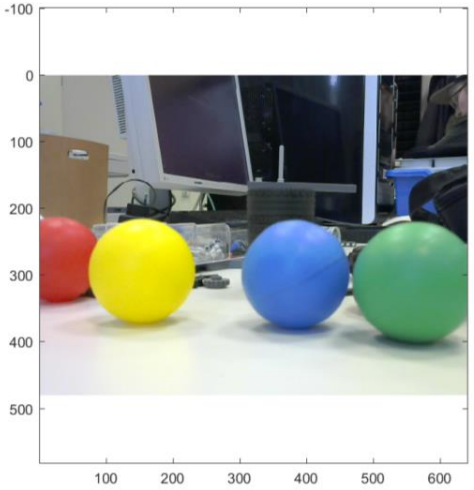



Abbildung 3: Aufgenommenes Bild der Webcam

Wie in der Vorbetrachtung bereits erläutert werden nun die Farbwerte, durch das RGB-System, elementweise miteinander verglichen, sodass genau ein Farbwert herausgefiltert wird. In die Matrix wird daraufhin der vorher festgelegte Farbwert eingetragen, sodass an allen Stellen, an welchen dieser besonders hoch sind eine 1 eingesetzt, wobei der Rest mit Nullen gefüllt wird (siehe Abbildung 4).

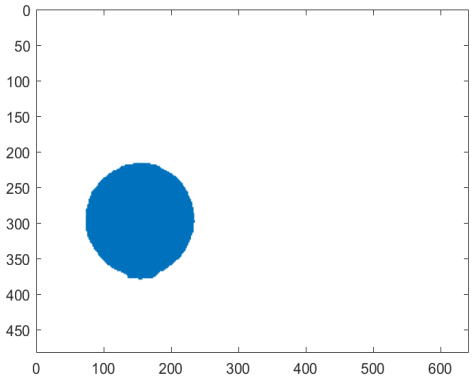



Abbildung 4: Markierung der gelben Punkte

Sobald solch eine Matrix erstellt wurde, wird die Summe der markierten Zeilen (für die y-Achse) und der Spalten (für die x-Achse) errechnet, sodass Vektoren entstehen. Daraufhin wird der Mittelwert aus den Vektoren gezogen, sodass sich ein Mittelpunkt klar bestimmen lässt, welcher sich nun in die Matrix übernommen wird (siehe Abbildung 5). So kann der Roboter sich auf diesen Punkt ausrichten. Um eine möglichst hohe Genauigkeit zu gewährleisten wird dieser Vorgang alle 0,1 Sekunden wiederholt.

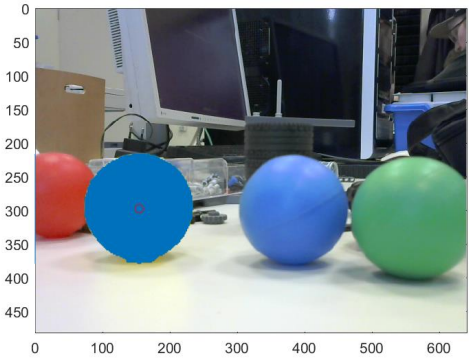



Abbildung 5: Analysiertes Bild mit berechnetem Mitelpunkt

Allerdings gibt es technisch bedingte Einschränkungen zu berücksichtigen, da die Kamera erst auf einer Distanz von unter 150 cm genau arbeiten kann. Damit dies realisierbar ist, haben wir einen Schwellwert implementiert, bei welchem die anschließenden Funktionen erst ab einer Anzahl von 500 markierten Pixeln erfolgen.

Die Feinjustierung beginnt, sobald ein Ziel durch die Bildanalyse gefunden wurde. Bei dieser drehen sich die Ketten in kleinen Zeitabständen relativ langsam, da durch die Verbindung per Bluetooth ein gewisser Delay zu berücksichtigen ist. Sobald der Mittelpunkt des Zieles mit einer Genauigkeit von 5 Pixeln im Bildmittelpunkt ist, fährt der Roboter auf das Ziel zu.

Sobald er einen bestimmten Abstand, welcher durch das Auszählen der markierten Pixel herausgefiltert wird, vom Ziel hat, hält er an und der Greifprozess beginnt. Dabei wird zunächst die Klaue durch einen Zahnradmechanismus geöffnet. Der Arm fährt herunter, die Klaue schließt sich wieder und der Arm kann nach oben fahren. Bei der Programmierung war dies mit Problemen verbunden, da die Motoren nicht immer gleichmäßig stark und schnell arbeiten und wir dafür zwei NXT-Einheiten ansprechen mussten.

Um das aufgenommene Objekt auch zu einem Ziel zu bringen, beginnt der Prozess erneut, wobei nun eine andere prägnante Farbe als Ziel dient und der Greifprozess ohne das Herauf- und Herunterlassen des Armes umgesetzt wird.

## IV. Ergebnisdiskussion

Das Projekt war für uns ein Erfolg. Wir haben es geschafft in der vorgegebenen Zeit einen funktionsfähigen, autonomen Aufräumroboter zu bauen und programmieren, welcher unter Laborbedingungen ohne Probleme arbeiten konnte.

Der Bildanalysealgorithmus war dabei eine der größten Hürden, da manche Farben nicht deutlich genug erkannt wurden oder aber die Kamera zu unscharfe Bilder auf große Entfernung schießt. Des Weiteren gab es kleinere Rückschläge durch die Instabilität der LEGO-Bauteile, welche nur sehr geringen Belastungen standhielten, oder durch die Ungenauigkeit der Motoren, welche nicht mit konstant gleicher Stärke gearbeitet haben, wodurch der Roboter teilweise nicht gerade gefahren ist oder aber die Klaue nicht

weit genug auf oder zu machen konnte. Hinzu kommt die geringe aber dennoch bemerkbare Verzögerung durch die Ansteuerung über Bluetooth. Auch wenn diese uns letztendlich viele Vorteile, vor allem im Thema Mobilität gegeben hat.

Doch diese Probleme lassen sich größtenteils einfach Beheben, wenn die Zeit dafür da wäre, sodass unser Roboter unter Realbedingungen gut umsetzbar wäre. Somit ist die Idee, einen voll autonomen Aufräumroboter für Zuhause zu konstruieren, gelungen und es lässt sich sagen, dass Roboter immer benutzerfreundlicher werden. Woran früher Wissenschaftler jahrelang gearbeitet haben, lässt sich durch moderne Technik heute von Studenten in zwei Wochen umsetzen.

## V. Zusammenfassung und Fazit

Während des zweiwöchigen Projektseminars ist ein funktionstüchtiger, autonomer LEGO-Roboter entstanden, welche automatisiert Gegenstände erkennt, aufnimmt und aus dem Weg räumt. Durch das Einbinden einer Webcam ist seine Funktionalität stark vergrößert worden. Die Programmierung und der Bau harmonieren miteinander, sodass er ohne große Probleme seine Aufgabe erfüllen kann.

Es gibt noch viele Erweiterungsmöglichkeiten. So könnte man durch eine stärkere Konstruktion, z.B. durch die Verwendung anderer Materialien, die Stabilität erhöhen, wodurch auch schwerere Gegenstände ohne Probleme aufgenommen werden könnten. Dafür wären allerdings auch leistungsfähigere Motoren nötig, welche auch die Genauigkeit erhöhen würden. Durch eine Umkonstruktion des Greifarmes könnte man diesen universeller gestalten, sodass auch andere Objekte mit anderen Formen aufgenommen werden könnten. Des Weiteren wäre es möglich, den gesamten Corpus drehbar zu lagern, wodurch einen Anhänger angebracht werden könnte, welcher die aufgenommenen Gegenstände sammelt. Hinzu kommt, dass eine genauere Kamera bessere Ergebnisse liefern würde. Das heißt, die Genauigkeit bei der Bildanalyse würde steigen, ebenso wie der Arbeitsradius, aber auch Arbeitsaufwand.

Somit wird deutlich, dass wir viel erreicht haben, aber auch noch viel verbessern könnten. Dies gilt nicht nur für unser Experiment, sondern auch für die gesamte Wissenschaft. Durch den rasanten technologischen Fortschritt, sind uns immer weniger Grenzen gesetzt.

## Anhang

```
frame=getsnapshot(camera);
camera.ReturnedColorSpace;
rotf=frame(:,:,3);
gruenf=frame(:,:,2);
blauf=frame(:,:,1);
for n=1:480
for m=1:640
gelb(n,m)=rotf(n,m)<gruenf(n,m)-100 &&
rotf(n,m)<blauf(n,m)-100;
end
end
nz=sum(sum(gelb));
framerotx=any(gelb,1);
frameroty=any(gelb,2);
posx=(find(framerotx,1,'first')+find(framerotx
,1,'last'))/2;
posy=(find(frameroty,1,'first')+find(frameroty
,1,'last'))/2;
```

Quellcode A: Bildanalyse

## Literaturverzeichnis


[1] CreditPlus - Studie: Fast drei von vier Deutschen interessieren sich für Haushaltsroboter: *https://www.creditplus.de/ueber-creditplus/newsroom/pressemitteilungen/presse-detail/news/studie-fast-drei-von-vier-deutschen-interessieren-sich-fuer-haushaltsroboter/*
Zitat: Zeile 7-11
Stand: 19.03.2018

[2] Wikipedia, the free Encyclopedia: Staubsaugerroboter *https://de.wikipedia.org/wiki/Staubsaugerroboter*
Stand: 19.3.2018

[3] Wikipedia, the free Encyclopedia: Unbemanntes Landfahrzeug: *https://de.wikipedia.org/wiki/Unbemanntes_Landfahrzeug*
Stand: 19.3.2018

[4] ELEARNING OvGU: Matlab Handbuch *https://elearning.ovgu.de/pluginfile.php/115111/mod_resource/content/2/Main.pdf*
Stand: 19.3.2018

[5] Prof. Dr. Elke Hergenröther: "Wie schreibt man einen wissenschaftlichen Artikel („paper")?" *https://www.fbi.h-da.de/fileadmin/personal/e.hergenroether/VertiefungAktuellerThemenCG/WieSchreibtManEinenWissenschaftlichenArtikel.pdf*
Stand: 19.3.2018

# Bau und Programmierung des autonomen Aufräumroboters „BERT“

Phillip Schulz, Elektro - und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Autonome Roboter bestimmen unseren Alltag immer mehr. Im Haushalt erledigen sie vor allem Aufgaben, die für den Menschen sehr zeitintensiv und eintönig sind. Daher wurde im Rahmen des Projektseminars Elektrotechnik / Informationstechnik (LEGO Mindstorms) ein LEGO Roboter konstruiert und programmiert, welcher mit Hilfe eines Greifarms und Greifers Objekte aufnimmt und zu einem bestimmten Ziel transportiert. Die Erkennung der Objekte erfolgte auf Grund deren Farbe. Dazu wurde eine Webcam verwendet und dessen Bild so bearbeitet, dass das Programm erkennt, wo sich das farbige Objekt im Bild befindet und wie groß seine scheinbare Größe ist. Dies ermöglicht eine präzise Ansteuerung an das Objekt.**

**In diesem Paper wird die Konstruktion und schwerpunktmäßig die Programmierung dieses Roboters dargestellt und erläutert. Es wird dabei auch auf die Farberkennung des Roboters eingegangen.**

***Schlagwörter*—Autonomisierung, Differenziallenkung, Farberkennung, Greifarm, Kettenfahrzeug, Webcam**

## I. Einleitung

AUTONOME Roboter werden mehr und mehr Bestandteil unseres Lebens. Sie sind unter anderen in der Industrie, im Militär oder im Haushalt anzutreffen. Dabei übernehmen sie Aufgaben, die für Menschen zu gefährlich oder einfach zu anstrengend oder zeitaufwändig sind. Im Haushalt werden sie vor allem eingesetzt, um Menschen die Arbeit abzunehmen. Das erhöht die Lebensqualität, da erheblich viel Zeit eingespart werden kann, um sinnvolleren Tätigkeiten nachgehen zu können. Besonders Staubsaugroboter oder rasenmähende Roboter sind hier zu nennen.

Als eine weitere zeitaufwändige Aufgabe im Haushalt ist auch das Aufräumen anzusehen. Es gab schon verschiedene Ansätze, dies durch Roboter zu erledigen, sei es in Filmen, wie Wall - E, wo der gleichnamige Roboter, die von den Menschen verlassene Erde aufräumt oder beispielsweise den Spielzeugroboter Roboscooper, welcher von der Firma Wowwee entwickelt wurde. Dieser kann Kleinobjekte mit seinen Greifarmen hochheben und auf eine Laderampe legen. Doch er ist trotzdem nur als Spielzeug anzusehen, da er nur sehr leichte Objekte, mit einem Gewicht von nicht über 30 Gramm heben kann. Außerdem ist er nicht in der Lage, seine Laderampe wieder selbstständig zu leeren.

Daran angelehnt wollen wir einen Aufräumroboter entwickeln, welcher Bälle aufnehmen kann und diese in einen Behälter legt. Sie können unter anderem für Müll stehen, welcher in einem Müllbehälter gebracht werden soll. Diesen Roboter wollen wir unter Verwendung von LEGO Mindstorms bauen und mit Matlab programmieren.

DOI: 10.24352/UB.OVGU-2018-056 

## II. Vorbetrachtungen

### A. Additive Farbmischung

Beim additiven Farbmischen kann jede Lichtfarbe in verschiedene Anteile bestimmter Grundfarben, auch Primärvalenzen, zerlegt werden. Nach dem RGB – System sind diese Primärvalenzen Rot, Grün und Blau. Sie sind linear unabhängig. Das heißt, dass keine Farbe durch die Kombination anderer Farben entstehen kann. Durch das additive Mischen entstehen immer hellere Farben. So entsteht aus Grün und Rot Gelb, aus Rot und Blau Violett und aus Blau und Grün Cyan. Weiß ist eine Mischung aus allen 3 Farben in gleichen Anteilen, während Schwarz das Fehlen jeglichen Farbanteil ist.

### B. Bild einer Webcam

Der Film einer Webcam besteht aus vielen hintereinander gezeigten Bildern. Diese Bilder sind in verschiedene Pixel geteilt. Bei der von uns verwendeten waren es 640 x 480 Pixel. Jedem Pixel wird eine Farbe zugeordnet. Diese Farbe wird auf Grundlage der additiven Farbmischung bestimmt. Das heißt für jeden Pixel werden die drei RGB Werte angegeben. Jeder Wert ist eine natürliche Zahl zwischen 0 und 255. Dies ist sinnvoll, da man mit einem Byte Speicherplatz genau 256 verschiedene Zeichen oder Zahlen codieren kann. Mathematisch ist dies als drei überlagerte Matrizen darstellbar, wobei jede Matrix einen Farbwert beinhaltet. Daher gibt die Webcam 3 Farbwertmatrizen an das Programm weiter.

## III. Hauptteil

### A. Bau des LEGO Roboters

Der Roboter wurde auf zwei Kettenblöcken aufgebaut. Jeder dieser Blöcke besteht aus zwei Ketten, die gemeinsam von einem Motor angetrieben werden. Die Nutzung von zwei Ketten pro Kettenblock, also insgesamt vier Ketten, war sinnvoll, um die Stabilität des Roboters zu gewährleisten, da dieser auf Grund der Verwendung von zwei NXT - Einheiten eine erhöhte Masse aufweist. Ein Problem dieser Konstruktion ist es aber, dass die inneren Ketten bei einer Drehung des Roboters langsamer laufen müssten, als die äußeren, da diese einen geringeren Radius zurücklegen müssen. Dies ist nicht realisierbar, so dass es bei Drehungen zu leichten Abweichungen kommen kann.

Die Kettenblöcke wurden darauffolgend miteinander verbunden, indem ein dritter LEGO Mindstorms Motor zwischen

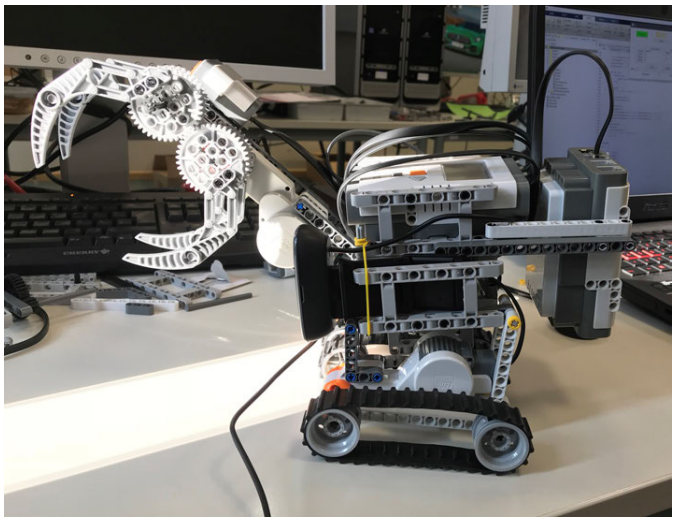


Abbildung 1. Foto des Aufräumroboters ”Bert”, aufgenommen am 20.02.2018

diesen befestigt wurde. Eine Differentiallenkung ist dadurch möglich, indem sich die Motoren der beiden Kettenblöcke in unterschiedliche Richtungen oder mit unterschiedlicher Geschwindigkeit drehen. Das gewährleistet, dass sich der Roboter um die eigene Achse drehen kann. Der Motor des linken Kettenblockes wird im Folgenden mit Motor A, der des rechten Motorblockes mit Motor B bezeichnet. Über dem Konstrukt mit den beiden Kettenblöcken wurde die erste NXT-Einheit angebracht. Diese ist mit den drei genannten Motoren verbunden.

Der Motor, welcher die beiden Kettenblöcke miteinander verbindet, ist für die Bewegung des Greifarms verantwortlich. Daher wurde der Greifarm an diesem Motor befestigt. Der Greifer ist am Ende des Greifarms, leicht nach links versetzt befestigt. Dieser besteht aus je zwei gegeneinander gestellten Greifzangen, die durch einen weiteren Motor bewegt werden. Dieser Motor ist auch am Greifarm befestigt. Da eine NXT Einheit nur maximal drei Motoren ansteuern kann, wird der Motor des Greifers von einer weiteren NXT Einheit angesteuert, welche hinten am Roboter befestigt wurde. Dies sorgt gleichzeitig für ein Gegengewicht zum Greifarm und damit für mehr Stabilität des Roboters.

Der Aufräumroboter besitzt keine LEGO Mindstorms Sensoren. Dafür eine Logitech C270 Webcam, welche alle Daten für das Programm liefert. Auf Grund der Form der Webcam wurde diese um 90° gedreht angebracht. Durch die Befestigung an der linken Seite ist sichergestellt, dass sich die Webcam genau unter der Greifzange befindet, was es dem Roboter erleichtert ein Zielobjekt exakt anzusteuern.

### *B. Farberkennung*

Um das Bild der Webcam nutzen zu können, muss es bearbeitet werden. Für den Algorithmus des Roboters (siehe Kapitel Programm des Roboters) sind die Anzahl der gelben bzw. grünen Pixel und der geometrische Mittelpunkt dieser gelb oder grün gefärbten Pixel notwendig.

Da die Auflösung der Webcam 640 x 480 beträgt, haben die Farbmatrizen 640 Spalten und 480 Zeilen. Um zu ermitteln,

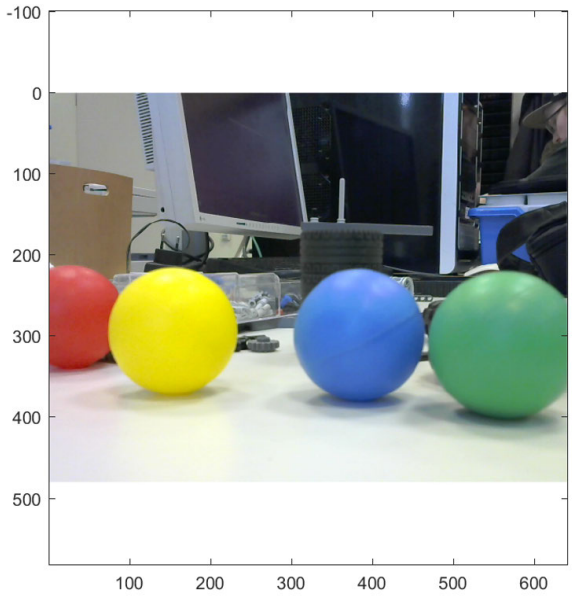



Abbildung 2. Bild der Wecam ohne Farberkennung, aufgenommen am 20.02.2018

welche Pixel gelb sind, müssen die drei Farbwertmatrizen miteinander verglichen werden. Da für die Farbe Gelb laut der additiven Farbmischung die Farbwerte für Rot und Grün deutlich höher sein müssen, als der Farbwert von Blau, wurden die Farbmatrizen der Webcam elementweise verglichen. Da dies elementweise durchgeführt wird, entsteht durch das Vergleichen auch eine 640 x 480 Matrix, gefüllt mit booleschen Werten. Der Wert eins resultiert, wenn der Farbwert für Rot an dieser Stelle größer ist als der um 100 erhöhte Farbwert für Blau, was auch für den Farbwert für Grün an dieser Stelle gelten muss. Ist dies nicht der Fall, beträgt der boolesche Wert an dieser Stelle 0. Der Wert 1 steht dann dafür, dass dieser Pixel gelb ist, der Wert 0 steht für das Gegenteil. Die Addition von 100 zum blauen Farbwert beim Vergleichen sichert, dass dieser deutlich kleiner ist, als die anderen beiden. Durch Ausprobieren wurde ermittelt, dass dies gut geeignet ist, um die Kugel bei unterschiedlichen Lichtverhältnissen zu erkennen und gleichzeitig deutlich von verschiedenen Hintergründen abzuheben. Um nun die Anzahl der gelben Pixel zu bestimmen, wurde die Summe aus allen Werten der booleschen Matrix gebildet. Die Position des Mittelpunktes der gelben Fläche setzt sich aus der Spalte und der Zeile zusammen, in welcher sich dieser befindet. Um die Spalte des Mittelpunktes zu bestimmen, wurden die Spalten höchster und niedrigster Ordnung herausgesucht, miteinander addiert und durch zwei dividiert. Dasselbe wurde mit den Spalten gemacht, so dass der Mittelpunkt in der resultierenden Spalte und der resultierenden Zeile des Bildes liegt.

Um immer einen hinreichend aktuelle Werte für die Anzahl der farbigen Pixel und die Position des Mittelpunktes zu erhalten, wird dieser Prozess der Farberkennung alle 0.1 Sekunden am aktuellen Webcambild wiederholt.

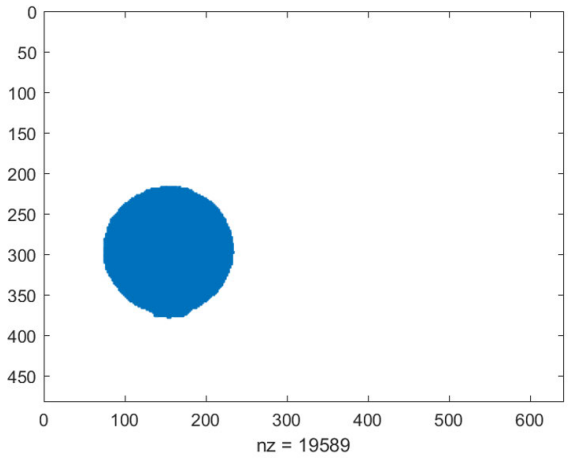



Abbildung 3. vgl. Abbildung 2; Schritt bei der Bildbearbeitung: alle gelben Pixel wurden blau gefärbt; nz entspricht der Anzahl der farbigen Pixel

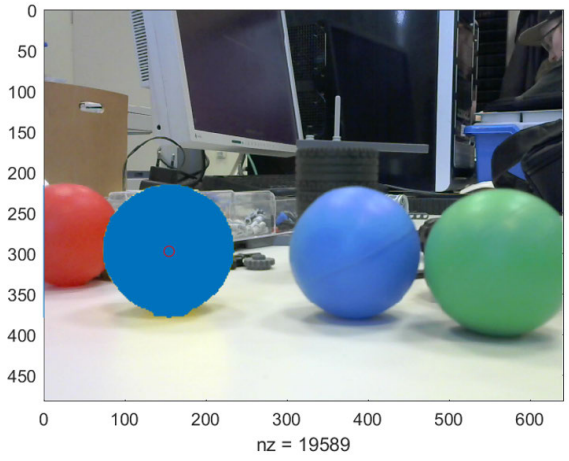



Abbildung 4. vgl. Abbildungen 2 und 3; Resultat nach der Bildbearbeitung: Abbildung 3 wurde auf Abbildung 2 gelegt und der geometrische Mittelpunkt der gefärbten Pixel wurde mit einem roten Kreis markiert

### *C. Programm des LEGO Roboters*

Ziel des Roboters ist es, ein Objekt auf Grund seiner Farbe zu erkennen, dieses automatisch anzusteuern, aufzunehmen und in ein bestimmtes Ziel zu legen. Alle dazu benötigten Eingabeinformationen erhält er durch die unter dem Greifarm am Roboter befestigten Webcam.

In der Bildbearbeitung werden die Farben gelb und grün herausgefiltert. Das Programm erhält so die Information, welche und wie viele Pixel gelb bzw. grün sind. Zudem berechnet es einen statistischen Mittelpunkt aller gelb bzw. rot gefärbten Pixel. Da die verwendete Webcam eine Auflösung von 640 x 480 Pixel hat und diese um 90° gedreht angebracht wurde, gibt sie den Mittelpunkt der gefärbten Pixel in einer 480 x 640 Matrix an. Mit Hilfe dieser Eingabeinformationen läuft ein Algorithmus ab, durch den der Roboter automatisiert gesteuert wird.

Im ersten Schritt muss sichergestellt werden, dass der Greifarm hochgefahren ist, damit er nicht im Blickfeld der Webcam hängt. Dies wird mit einer Drehung des Greifarmmotors in positive Drehrichtung realisiert. Eine Drehung in positive Drehrichtung erfolgt entgegen des Uhrzeigersinnes, eine Drehung in negative Drehrichtung erfolgt im Uhrzeigersinn.

Darauffolgend scannt der Roboter nach einem gelben Ball, welcher das aufzuräumende Objekt symbolisiert. Dazu dreht er sich rechtsseitig um die eigene Achse, bis die Anzahl der gelb gefärbten Pixel im Bild der Webcam den Wert von 500 überschreitet. Um dies zu realisieren, dreht sich der Motor A mit 20% Leistung in positive Drehrichtung, während sich Motor B mit 20% Leistung in negative Drehrichtung dreht.

Nach Abschluss des Scann – Schrittes beginnt der Schritt der Feinjustierung auf das Ziel. Dazu wird die Position der Mitte der gelben Pixel entlang der horizontalen benötigt, also die Spalte der 480 x 640 Matrix, in welcher sich die Mitte der gelben Pixel befindet. Das Ziel der Feinjustierung ist es, dass sich dieser Mittelpunkt in Spalte 235 bis 245 befindet. Ist dies erreicht, fährt der Roboter zum aufzunehmenden Objekt. Um dieses Ziel zu erreichen, wird immer nach Beendigung jedes Drehschrittes des Roboters geprüft, wo sich der Mittelpunkt befindet. Befindet er sich in einer Spalte geringerer Ordnung als 235, so dreht sich der Roboter nach rechts. Dazu bewegen sich Motor A mit 5% Leistung in positive Drehrichtung und Motor B mit 5% Leistung in negative Drehrichtung. Dies geschieht 0.7 Sekunden lang. Befindet sich der Mittelpunkt in einer Spalte höherer Ordnung, so drehen sich Motoren A und B 0.3 Sekunden lang in die entgegengesetzte Drehrichtung. Also dreht sich der Roboter 0.3 Sekunden nach links. Die Drehzeit in die entgegengesetzte Richtung ist geringer, um zu verhindern, dass sich der Roboter so dreht, dass der Mittelpunkt immer zwischen zu weit links und zu weit rechts hin- und herspringt.

Um zum aufzunehmenden Objekt zu fahren, drehen sich die Motoren A und B mit 41% bzw. 40% Leistung so lange, bis die Anzahl der gelben Pixel größer gleich 50000 ist. Die Rotationsgeschwindigkeit von Motor A ist leicht größer als die von Motor B, da der Roboter konstruktionsbedingt einen leichten Linksdrall hat. Dieser wird durch die stärkere Benutzung von Motor A ausgeglichen. Dieser Linksdrall entsteht durch ein leichtes Übergewicht des Roboters auf der linken Seite, da Webcam und Greifer aus Sicht des Roboters vorne links angebracht sind.

Darauffolgend nimmt der Roboter den Ball auf, in dem er die Greifklaue öffnet, den Greifarm herunterfährt, die Klaue schließt und den Greifarm darauffolgend hochfährt. Dies wird durch kurze Bewegungen der für den Greifarm und der Greifklaue zuständigen Motoren realisiert. Das Schließen der Klaue, sowie das Herunterfahren des Greifers, erfolgt durch Bewegung der Motoren in die entgegengesetzte Richtung wie beim Öffnen des Greifers bzw. Hochfahren des Greifarms.

Der nächste Schritt des Roboters ist es, den Ball zum Ziel zu bringen. Dazu geht er analog zum Aufnehmen des Balles vor.

Das Ziel wurde mit grünen LEGO – Steinen markiert. Daher werden diesmal grüne Objekte angesteuert. Ist der erforderliche Abstand erreicht, öffnet der Roboter die Klaue und dreht sich um 180°, was den Abschluss des Algorithmus entspricht.

## IV. Ergebnisdiskussion

Das Endergebnis ist ein Kettenfahrzeug mit Greifarm, welches gelbe Bälle automatisiert aufsammeln und in einem grün markierten Behälter ablegen kann. Des Weiteren ist es möglich, den Roboter auf Objekte anderer Farben und Größen, sowie andersfarbige und unterschiedlich große Farbmarkierungen, einzustellen. Doch jeder Änderung der Größe, sowohl des Zielobjektes als auch des Behälters, haben eine längere Einstellung auf diesen zu Folge. Dies ist darin begründet, dass das Programm die Entfernung zum angestrebten Objekt auf Grund dessen relativer Größe bestimmt. Dabei sind aber Größe und Farbe der Objekte nur im bestimmten Rahmen frei wählbar. Die Farben sollten so gewählt werden, dass sie nicht in der näheren Umgebung des Roboters vorkommen. Sonst erkennt die Farberkennung auch diese Objekte als Zielobjekte an, so dass der Roboter sie ansteuert. Gleichzeitig sollten die Farben hell und kräftig sein. Das erleichtert das Erkennen der Farbe durch die Software, da sich solche Farben gewöhnlich stark von der Umgebung abheben. Und auch die Größe der Objekte und Farbmarkierungen muss so gewählt werden, dass sie ausreichend groß sind, so dass sie von der Farberkennung erkannt werden können, aber nicht zu groß für den Greifer sind. Auch zu weite Entfernungen stellen aus zwei Gründen ein Problem für den Roboter dar. Da entfernte Objekte auf Grund der Perspektive kleiner erscheinen, müssen die Zielobjekte eine auch für die Entfernung eine gewisse Größe haben. Doch auch in dem Prozess der Feinjustierung auf das Objekt treten bei großen Entfernungen Probleme auf. Dies liegt darin begründet, dass die Feinjustierung auf weit entfernte Objekte sehr genau sein muss, um diese anzusteuern. Das ist auf Grund gewisser Ungenauigkeiten der LEGO Motoren und den leichten Abweichungen auf Grund der Verwendung von vier Ketten nicht möglich.

## V. Zusammenfassung und Fazit

Das Ziel der Arbeit war ein Roboter, welcher kleinere Objekte auf Grund seiner Farbe erkennt, diese aufnimmt und zu einem Ziel bringt, welches er ebenfalls auf Grund dessen Farbe erkennt. Dies wurde mit unserem LEGO Roboter umgesetzt. Gewisse Ungenauigkeiten auf Grund der verwendeten Bauteile waren vorherzusehen und nicht vermeidbar. Im Grunde ist unser Roboter ein Modell, welches bestimmte Anforderungen und Problemstellungen aufzeigt, die bei einer industriellen Entwicklung eines solchen Roboters beachtet werden müssen.

Unser gewählter Ansatz in der Konstruktion und der Programmierung ist eine Möglichkeit diesen Anforderungen und Problemen zu begegnen. Dabei ist unser Roboter auf zwei Kettenblöcken, welche zur Fortbewegung dienen, aufgebaut. Um die Objekte aufzunehmen ist ein Greifarm integriert, welcher sich nach oben und unten bewegen lässt und einen Greifer an der Spitze öffnen und schließen kann. Dem Roboter liegt ein Programm zu Grunde, welches mit Hilfe der Informationen einer Webcam diesen automatisiert steuert. Dazu ist das Programm in der Lage, gesuchte Farben im Bild der Webcam zu erkennen.

Weitere Ergänzungsmöglichkeiten des Roboters wären unter andern ein schwenkbarer Greifarm, um Objekte nicht nur hoch – und runter bewegen zu können, sondern auch zur Seite. Dies wäre besonders kombiniert mit einer Ladefläche für kleine Objekte sinnvoll, um diese auch zwischenlagern zu können. Auch das Programm des Roboters bietet Erweiterungsmöglichkeiten. Als besonders sinnvoll ist die Ergänzung mit einer Formerkennungssoftware zu nennen, um Fehlermöglichkeiten zu beseitigen, welche in der Erkennung umstehender Objekte oder anderer im Hintergrund erkennbarer Farben (z.B. Farbe der Wände, andere Roboter, usw.) liegen. Auch eine Erhöhung der Reichweite des Roboters wäre denkbar, zum Beispiel, in dem man nach einer bestimmten Fahrstrecke eine weitere Feinjustierung auf das Objekt vornimmt. So ist es möglich, weit entfernte Objekte ohne sehr genaue Feinjustierung anzusteuern.

## Literaturverzeichnis

[1] Andreas Donath: Roboscooper packt den Müll weg. $http://www.handelsblatt.com/technik/forschung-innovation/aufraeumroboter-roboscooper-packt-den-muell-weg/3506314.html$. Version: März 2018

[2] RWTH Aachen: „RWTH - Mindstorms NXT Toolbox“. $http://www.mindstorms.rwth-aachen.de$ Version: Februar 2018

[3] Wikipedia, the free Encyclopedia: Autonomer, mobiler Roboter. $https://de.wikipedia.org/wiki/Autonomer_mobiler_Roboter$. Version: März 2018

[4] Spektrum.de: autonomer Roboter. $http://www.spektrum.de/lexikon/neurowissenschaft/autonomer-roboter/1159$. Version: März 2018

# *Bau eines autonomen Spielfeldzeichners*

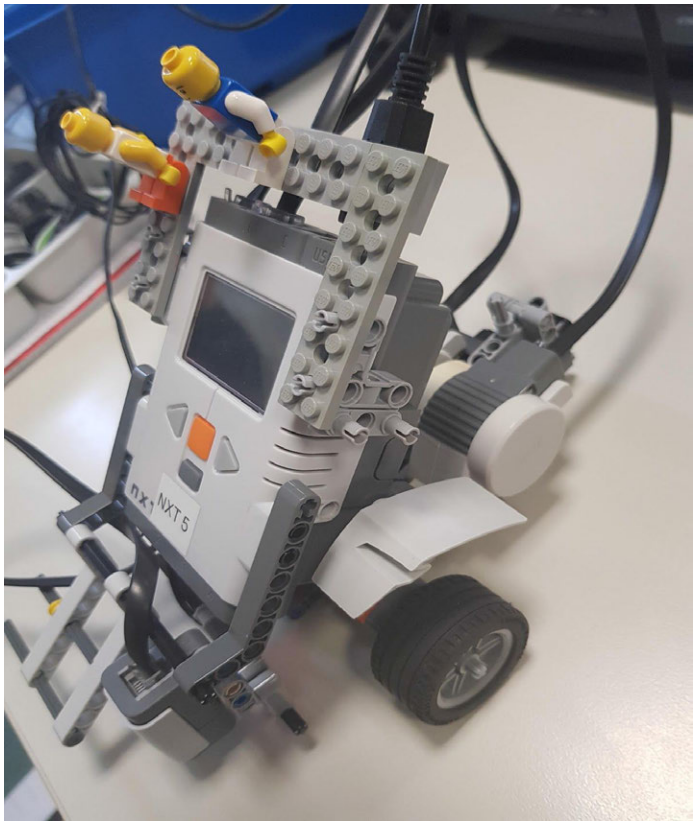
Ali Kharnoub , ETIT

Otto-von-Guericke-Universität Magdeburg

***Abstract***

**Noch nie haben die Extreme von Zeitnot und Langeweile das Leben so stark geprägt, wie in unserer beschleunigten Gesellschaft. Wie viel Tempo verträgt der Mensch? Selbständige Roboter können Hilfe leisten und eine sinnvolle Ergänzung sein.**
**In der modernen Industrie werden Prozesse automatisiert, um sie von Fehlern zu bewahren. Anlagen sollen 24 Stunden laufen können, effizient, effektiv und sicher arbeiten können. Dabei kommt es entscheidend darauf an, vorgegebene Werte einzuhalten, um eine qualitativ hochwertige, umweltgerechte und sichere Produktion zu gewährleisten.**
**Die Programmierungsmöglichkeiten erlauben in der Folgezeit genau das, was man sich als Ziel setzt. z.B. die komplexe Montage innerhalb der Automobilfertigung zu erfüllen.**

Nicht nur in der Industrie werden wir immer mehr Robotertechnik in den nächsten Jahren erleben, sondern auch im privaten Bereich von der Waschmaschine bis zum Staubsaugroboter.
Dienstleistungen, Pflege, Zustelldienste und alle Routinetätigkeiten werden künftig Maschinen erledigen.
Eine mögliche Anwendung ist das Zeichnen von Spielfeldern in Stadien oder Sporthallen.
Diese Routinearbeit kostet Zeit, Geld und kann anstrengend sein aber braucht keine menschliche Intelligenz, weil die benötigten Informationen und Messungen schon angegeben sind.
Deswegen können Spielfeldbegrenzungen von Automaten gezeichnet werden. Außerdem ist der Ersatz eines Roboters in diesem Fall effizienter, schneller und günstiger.
Im Zeitraum von 12. Februar bis 23. Februar 2018 wollten wir solche Maschine Bauen.
Das Projekt wurde im Rahmen des LEGO-Mindstorms Seminars bearbeitet und das Ziel war, einen Roboter zu konstruieren bzw. Programmieren.
Der Roboter sollte eine schwarze Linie auf weißen Untergrund abfahren und in einer bestimmten Weise sie speichern können. Danach kann er diese Linie reproduzieren.
Außerdem sollte auf den Roboter ein Stift fixiert werden, um die Linie nachzeichnen zu können.
Der Aufbau erfolgt mit Lego unter Verwendung von NXT-Baustein. Mithilfe des von der RWTH Aachen entwickelten Toolkits wird der Roboter in Matlab programmiert.
Die Anwendung dieses Toolkits unterstützt den direkten Zugriff auf die Sensoren und Motoren unter Matlab.
Simpel aber fein, so war der Aufbau.

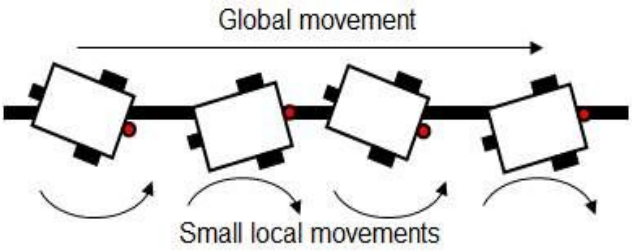



Die Unterschicht besteht aus zwei Motoren, die nebeneinander angebracht wurden und auf jeden Motor ist ein Rad angewiesen.
Zur Unterstützung von Stabilität und Flexibilität wurde ein drittes freies Rad verbunden, das sich um 180 Grad drehen kann. Auf der Unterschicht ist der NXT-Baustein festgelegt und da vorne wird der Farbsensor fixiert.

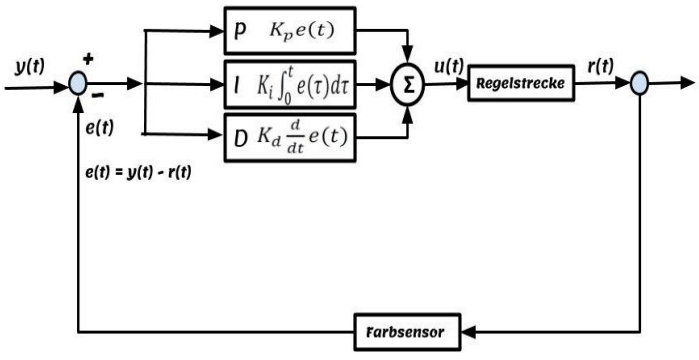
Die Stifthaltung war ein umstrittenes Thema.
Der Stift sollte entweder neben dem Farbsensor oder in der Mitte festgestellt werden. Zwar wäre die Fixierung in der Mitte besser, weil dieser Punkt im Zentrum liegt . dies hätte die Stabilität unterstützt und hätte zur genaueren Nachzeichnung der Linie aber das war sehr aufwendig und könnte Störungen wegen der zusätzlicher Reibung bewirken.

Das Grundkonzept eines Linienfolgers stellt sich wie folgt dar: Der Roboter folgt nicht der Linie selbst, sondern ihrer Grenze und fährt in Form von Zickzack. d.h. statt einer globalen Bewegung wird er in kleine lokale Bewegungen gesetzt.

DOI: 10.24352/UB.OVGU-2018-058 




Zuerst erkennt der Sensor den Farbwert und je nach dem eingelesenen Farbwert wird die Spannung der Motoren eingesetzt.
Zwar kann die Bewegung durch diesen Prozess theoretisch korrigiert werden aber es reicht nicht aus, die Bewegung realisieren zu können, da solches Verfahren wie eine einfache boolesche Variable wirkt und Ungenauigkeit bzw. Fehlern beim folgen der Linie, besonders bei geraden und rechteckigen Strecken verursacht. Zur Vermeidung der Fehlern in diesem Prozess, ist es sinnvoller, die Motorsteuerung durch effizienteren Regler zu bestimmen und aus diesem Grund besteht der Bedarf an der Regelungstechnik.
Die Regelungstechnik automatisiert verschiedene Vorgänge so, dass eine gewünschte Betriebsart gestoppt wird oder erhalten bleibt.
Die Regelungstechnik versteckt sich in vielen technischen und natürlichen Systemen und beschäftigt sich mit dem Problem, dass bei den meisten Systemen der gewünschte Einfluss des Eingriffs auf die Stellgröße nicht direkt erzielt werden kann, sondern einen Zeitverzug zwischen dem Eingriff und seinen bemerkbaren Einflüssen auftritt, dabei dient ein Regelkreis dazu, die Regelgröße auf die gewünschte Führungsgröße zu bringen und auf dieser Ebene zu halten.
Bei der Verwendung des Regelkreises im Linienfolger ist das Ziel, die Regeldifferenz zwischen der vorgegebenen Fahrspur (Sollwert) und dem eingelesenen Farbwert (Istwert) auf Null zu setzen.




**PID Regelkreis**

Die Regelstrecke ist die Störgröße, die vom Regler ausgeregelt werden soll.
Es existieren bereits viele unterschiedliche Regler mit verschiedenen Art und Funktionsweise. Einer davon ist der in unserem Projekt verwendeten PID-Regler, der aus drei Glieder besteht, und zwar P,I und D. Diese Buchstaben stehen für Proportional, Integral und Differenzial. Zugleich gibt es auch die drei zugehörigen Parameter Kp, Ki und Kd.

## Parameter Tabelle

| | |
|---|---|
| e(t) | Regeldifferenz |
| y(t) | Führungsgröße (Sollwert) |
| u(t) | Stellgröße |
| r(t) | eingelesener Farbwert (Istwert) |
| Kp,Ki,Kd,dT | PID-Regler Beiwerte |
| L_Power | Kraft des linken Rads |
| R_Power | Kraft des eichten Rads |

Das P-Glied ist die einfachste Teil. aber gleichzeitig ist er der Hauptanteil, der die Gegenwart betrachtet und für eine schnelle Reaktion sorgt. Er misst die Regeldifferenz, multipliziert sie mit dem vorgegebenen Proportionalbeiwert und ergibt den ersten Anteil der Stellgröße.

$$e(t) = y(t) - r(t)$$
$$u(t) = K_p\, e(t)$$

Das I-Glied berücksichtigt die alten Messwerte und sorgt dafür, dass der sollwert erreicht werden kann. d.h. die bei proportionalen Regelungen bleibenden Störungen können durch die Verwendung eines I-Reglers behoben werden.

$$u(t) = K_i \int_0^t e(\tau)\, d\tau$$

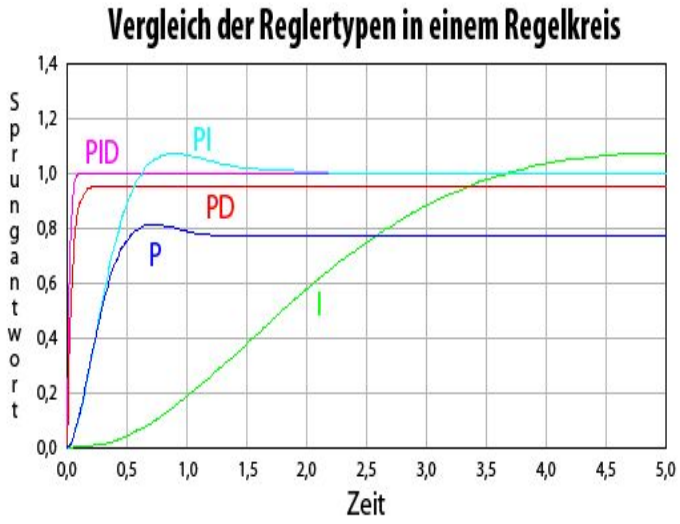
Das D-Glied sorgt für die künftigen Messwerte und reagiert auf die Änderungsgeschwindigkeit der Regeldifferenz. Alleine lässt sich das D-Glied als Regler nicht einsetzen.
Außerdem ermöglicht es eine schnelle Nachregelung bei plötzlich eingreifenden Störeinflüssen.

$$u(t) = K_d \frac{d}{dt}\, e(t)$$

Die Ausgangsgröße eines idealen PID-Reglers sieht folgenderweise aus (Die Allgemeine Formel) :

$$u(t) = K_p(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{d}{dt} u(t)$$

Je nach dem Bedarf lassen sich die Anteile des PID-Reglers auch als einzige Regler verwenden, wie z.B. P-, I-, PI- und PD-Regler.
Zwar ist es schwierig, ein PID-Regler zu bilden und es reicht manchmal aus zweipunktigen Regler zu verwenden. Aber er bietet ein paar Vorteile an.
PID-Regler verknüpft die guten Eigenschaften der anderen Reglern, außerdem verfügt über gute Genauigkeit und reagiert sehr schnell. Zuletzt kann man mit einem guten Pid-Regler fast jedes System in den Griff bekommen.



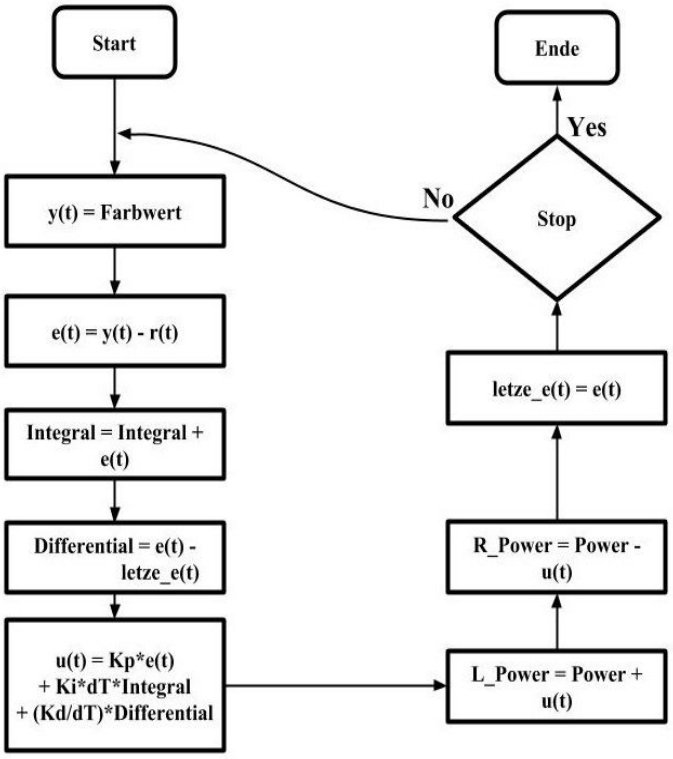

PID-Regler betrachtet man als stetige lineare Regler. Es gibt auch zwei andere Arten von Reglern:

* Nichtlineare Reglern wie Fuzzy-Regler, adaptive Regler und Extremwertregler.
* Unstetige Regler wie Zweipunktregler, Dreipunktregler und Mehrpunktregler.

In unserem Roboter wurde ein GUI ( Graphical User Interface) benutzt. Die GUI verfügt über drei Knöpfe : Abfahren, Stop, Nachzeichnen, jeder Knopf für eine Funktion.
Das Hauptprogramm verbirgt sich in Abfahren-Funktion und besteht aus einer Schleife.
Sowohl die idealen Beiwerte des PID-Reglers als auch die passenden Spannungen der Motoren haben wir schon durch Experimente bestimmt und in Hauptprogramm als Konstante hinzugefügt.
Die Anfangswerte wie Integral,Differenzial und Regeldifferenz wurden zum Beginn natürlich gleich Null eingesetzt.

Normalerweise benutzt man im Linienfolger einen Lichtsensor aber wir haben einen Farbsensor verwendet, weil er während der Versuche bessere Genauigkeit zeigte.

Am Anfang wird auch der Roboter so gelegt, dass der Farbsensor senkrecht über der linie hängt, damit er zuerst das Schwarz erkennt. Der erkannte Farbwert wird im Programm als Führungsgröße (Sollwert) deklariert, dann fährt der Roboter an und der Farbsensor erkennt neue Werte (Istwert).
Im Programm wird der Regeldifferenz zwischen dem Sollwert und dem zeitlichen Istwert berechnet und zur Integral addiert, dann wird der neue Integralwert gespeichert .
Im nächsten Phase wird die von der vorherigen Schleife entstehenden Regeldifferenz von der neuen Regeldifferenz subtrahiert und als die neuen Differential eingesetzt.
Nun kommt der PID-Regler ins Spiel. Er berechnet die Stellgröße und steuert den Motoren so, dass sie das Fahrzeug drehen, um die Regeldifferenz so weit wie möglich um Null zu nähern.




**Die Funktion von Linienfolger**

Man kann sagen, dass die Stellgröße entspricht der Drehung. Anschließend wird die neue Regeldifferenz für die nächste Runde angemerkt und wiederholt sich der Prozess.

Die während der Funktion eingelesenen Farbwerte werden in einem globalen Array gespeichert.
Die Stop-Funktion hat nur die Auftgabe, die Schleife zu beenden..
In der Nachzeichnen-Funktion sieht es ähnlich wie in der Abfahren-Funktion aus. der Unterschied besteht nur darin, dass anstatt neue Farbwerte abzulesen, werden die im Array schon gespeicherten Farbwerte in die Schleife durchgeführt.

Beim Abfahren der Linie bewies der Roboter die Effektivität des PID-Reglers und folgte der Linie nahtlos aber leider tritt ein Problem beim Nachzeichnen auf.

Eigentlich konnte der Roboter die abgefahrene Strecke ganz genau reproduzieren aber nach der Fixierung des Stifts wurden Störungen beobachtet.
Aufgrund der von Stift entstehenden Reibung und der Ungenauigkeit der Motoren wich der Roboter ab und fuhr in falsche Richtungen.

Wir versuchten mit verschiedenen Stiften die Fixierungsweise zu ändern aber es gelingt uns nicht.
Mangels der Zeit konnte das Problem nicht behoben werden aber eine mögliche Lösung wäre, einen Regler zu gestalten, um die Störungen zu eliminieren und den Prozess in Ordnung zu bringen.

Das Ergebnis ist ziemlich erfreulich trotz der begrenzten baulichen Möglichkeiten und der Ungenauigkeit von Motoren und Sensoren.
Außerdem ist der Roboter aufgrund der GUI simpel zu steuern und lässt sich mit ein paar Erweiterungen in vielen Branchen wie Lieferservice, Fahrdienst oder Reinigungsdienst einsetzen.
Eine mögliche Erweiterung wäre z.B. ein Planierschild mit einem Drucksensor und einem Ultraschallsensor. Der Roboter soll auf solche Weise programmiert werden, dass er Hindernisse erkennt und sie entweder zur Seite verschiebt oder umfährt, wie ein autonomer Bulldozer oder Schneepflug.
Eine weitere Anwendungsmöglichkeit wäre beispielsweise beim Begleiten von Sehbehinderten oder auch von Senioren in Altenwohnheimen.

### *Fazit*

**Zusammenfassend kann man feststellen, dass den selbstständigen Robotern in unserer modernen Gesellschaft immer mehr Bedeutung eingeräumt wird. Niemand weiß wie unser Leben morgen aussieht aber viele Entwicklungen zeichnen sich schon heute ab. Vielleicht kann unserer Roboter nun nur Feldbegrenzungen zeichnen, aber wer weiß, was die Zukunft für ihn bringt.**
**Die Erfindungen von heute sind die Basis für die Anwendungen von Morgen und Die Zukunft bleibt unbekannt, doch sicher ist, dass die Innovationen der Robotik, die Bildverarbeitung und Montagetechnik unseres Lebens nachhaltig beeinflussen, Schon heute aber noch viel stärker morgen**.

## Literatur

[1] W. Schumacher , M. Maurer : “ Grundlagen der Regelungstechnik” , Institut für Regelungstechnik IFR, https://www.ifr.ing.tu-bs.de/static/files/lehre/vorlesungen/gdr/Skript_GdR.pdf ,Stand: 18.09.2014

[2] Mondada Francesco, Michael Bonani, Xavier Raemy, James Pugh, Christopher Cianci, Klaptocz Adam, Stephane Magnenat, Jean-Christophe Zufferey, Dario Floreano, Alcherio Martinoli, "The E-puck, a Robot Designed for Education in Engineering", Proceedings of the 9th conference on Autonomous Robot Systems and Competitions, Vol. 1, pp: 59-65, 2009.

[3] Zimu Wang , “A MODEL OF LINE FOLLOWING ROBOT USING PID CONTROLLER” , Stand : 2015, http://hig.diva-portal.org/smash/record.jsf?pid=diva2%3A921842&dswid=8312

[4] Alexander Behrens, Linus Atorf, Robert Schwann, Bernd Neumann, Rainer Schnitzler, Johannes Ballé, Thomas Herold, Aulis Telle, Tobias G. Noll, Kay Hameyer, and Til Aach, “MATLAB Meets LEGO Mindstorms—A Freshman Introduction Course Into Practical Engineering”, IEEE Transactions on Education, vol. 2, no. 2, May 2010.

[5] Ziegler J.G and Nichols N. B., Optimum Settings for Automatic Controllers, pp: 759–768, trans. ASME, 1942.

[6] Karl Heinz Fasol, Klaus Diekmann (Hrsg.), “ Simulation in der Regelungstechnik “, ISBN 978-3-642-84261-0

[7] International Federation of Robotics, World Robotics 2015 Industrial Robots http://www.ifr.org/industrial-robots/statistics/

[8] Mike Williams, History of Robotics. A class assignment, Ball University. ITDPT 303 Manufacturing Systems. http://www.bsu.edu/web/mawilliams/history.html

[9] RWTH - Mindstorms NXT Toolbox for MATLAB. Available: http://www.mindstorms.rwth-aachen.de/trac/wiki/Download

# Der Connect4-Roboter- Die Unterhaltung der Zukunft

Christoph Andres, Elektro und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

Auch in diesem Jahr 2018 wurde das Lego Mindstorms Praktikum von der Otto von Guericke Universität durchgeführt. Viele sehr unterschiedliche Roboter wurden geplant, praktisch umgesetzt und vor einer Juri und Studenten vorgestellt. So auch der Connect4-Roboter, mit dem sich in diesem Paper auseinandergesetzt wird. Dabei wurden die Scherpunkte auf die Planung der Idee und die Umsetzung des Konzeptes gelegt. Der Connect4-Roboter, welcher in der Lage sein sollte einen zweiten menschlichen Spielpartner für das bekannte Vier-Gewinnt Spiel zu ersetzen, brachte in seiner Umsetzung viele Probleme mit sich. Diese Probleme und ihre Lösungen werden im nachfolgendem Hauptteil aufgegriffen und erläutert. Dennoch ist es den Entwicklern gelungen, am Ende des zwei wöchigen Projektseminars, einen autonom fahrenden und spielenden Roboter zu präsentieren.

## *I. Einleitung*

Die Welt im 21. Jahrhundert ist eine sehr technisch versierte Welt. Der wissenschaftliche und technische Fortschritt bestimmt den Markt. Ob in Industrie oder privaten Haushalt, überall werden die neuesten Innovationen angewendet und benötigt. Auch, wenn meiner Meinung nach, jeder gesunde Mensch in der Lage sein sollte seine Wohnung noch selber zu reinigen, sind staubsaugende Roboter zeitsparend. So werden unteranderem nicht nur die lästigen Tierhaare beseitigt, sondern auch die Verursacher beschäftigt. Sicherlich profitieren auch ältere Menschen von den intelligenten Putzmaschinen, da sie ihnen körperliche Arbeit abnehmen und erleichtern können.
Auch die Unterhaltungsbranche bietet der Technik eine große Spielfläche. Ob Virtual Reality Brillen oder die neuesten Smartphones zeigen, zu was der Mensch in der Lage ist. Da sich vor allem die Technik in der Unterhaltungsbranche als aktuelles und interessantes Thema für uns darstellte, wollten wir einen Roboter erfinden, der die Fähigkeit besitzt, dem Menschen in seiner Freizeit zu dienen. Aus diesem Grund entwickelte sich die Idee, einen Roboter zu konstruieren, welcher in der Lage sei das Gesellschaftsspiel Vier-Gewinnt selbständig zu spielen.

DOI: 10.24352/UB.OVGU-2018-059 

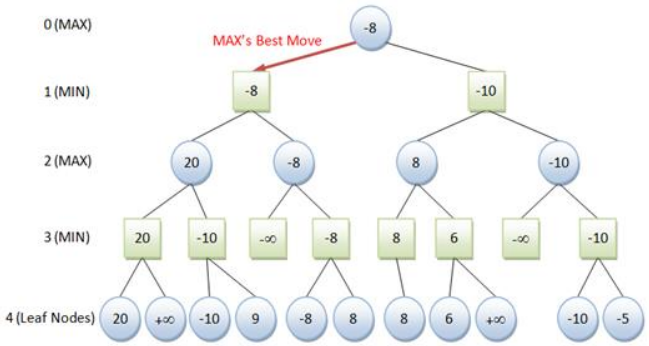
Der sogenannte „Connect4-Roboter“ wäre also eine gute Option für ältere Menschen oder Kinder, welche nicht oder noch nicht auf dem höchsten Stand der Technik stehen. Auch könne er von Menschen, die oft alleine sind sinnvoll genutzt werden. Zur Entwicklung dieses Vorhabens würde also eine präzise und kluge Anlage benötigt werden, welche den Anforderungen der heutigen Gesellschaft nachkommen könnte.

## *II. Hauptteil*

Die Idee des Connect4-Roboters war es, dass ein zweiter menschlicher Spielpartner für das Vier-Gewinnt-Spiel, durch den Roboter ersetzt werden kann. Dadurch wäre es Menschen möglich auch in der reellen Welt und nicht nur auf virtueller Ebene, alleine so genannte „Multiplayer“ Spiele zu spielen.
Der Roboter sollte sich autonom bewegen können und in der Lage sein, die eigenen Spielchips einzuwerfen. Um den Roboter noch attraktiver zu gestalten, sollte er Anfangs die gegnerischen Spielsteine mit einem hoch und herunterrunter fahrendem Arm erkennen und diese Information an den Rechner weitergeben. Dazu sollte er eine ordentliche Software verpasst bekommen. Er sollte klug und intelligent spielen können und nicht nur als passiver Spielpartner agieren. Das bedeutet, dass der Roboter auch selber versucht sich ein System an Spielchips aufzubauen und nicht einfach nur die „Vier in einer Reihe“ seines Spielpartners zu verhindern.
Die Erkennung der gegnerischen Spielsteine sollte über einen Farbsensor, der an diesem Arm angebracht war, realisiert werden. Das hieße, dass der Vier-Gewinnt spielende Roboter sich oberhalb des Spieles auf einer gedachten X-Ebene und der Farbsensor in Y-Richtung bewegen müsste. Dafür bräuchte man ein Podest auf dem der Roboter stehen könnte und gegebenenfalls eine Schiene oder andere Vorrichtung, damit er möglichst gradlinig fahren kann. Dies erschien als sehr wichtiger Aspekt, da bei nicht gradliniger Bewegung, der Roboter nach wenigen Spielzügen anfangen würde die Steine nicht mehr genau in die schmalen und kleinen Spalten zu schmeißen. Um zu gewährleisten, dass die Spielchips des Roboters auch nahezu immer in die ermittelte Spalte des Spiels fallen, sollte auch die Abschuss- und Einwurfvorrichtung ziemlich genau konstruiert werden.

Der sich in die y-Richtung bewegende Arm mit integriertem Farbsensor sollte immer die oberste Reihe, in der keine Spielsteine liegen, abfahren. Dazu würde natürlich auch die Bewegung des gesamten Roboters in x-Richtung erforderlich sein. So könnte Zeit gespart werden, da er nicht jedes einzelne Feld abfahren müsste und neu überprüfen müsste welche Chipfarbe sich dort befindet. Mit dieser Art der „Chipfindung“ würde er lediglich den vom Gegner neu eingeworfenen Spielstein ausfindig machen.
Die Abschussvorrichtung sollte auch von einem Motor betrieben werden und die Chips aus dem Roboter herausschieben. Dieser Vorgang dürfte nicht zu schnell aber auch nicht zu langsam geschehen, damit die Chips immer mit der gleichen Geschwindigkeit in die Einwurfvorrichtung geschoben werden könnten. Würde die Geschwindigkeit variieren, könnte dadurch die Trefferquote des Roboters abgesenkt werden.

Da bekannt war, dass das Vier-Gewinnt-Spiel aus 6x7 Feldern bestand, wollten wir eine Matrix in dieser Größe erstellen, um so die Basis für unser Programm zulegen. Die Matrix würde also aus 42 Feldern bzw. „Arrays“ bestehen und beinhaltet dadurch die Information, dass das Spiel aus maximal 42 Zügen bestehen könne.
Bei den Überlegungen zu dem Programm, entwickelte sich immer stärker die Frage, wie intelligent der Roboter spielen sollte bzw. könnte. Der perfekt programmiert und spielende Roboter wäre also in der Lage, im Falle der gegebenen Matrix, 42 Züge in die Zukunft zuschauen. Um jedoch so einen „perfekten Gegenspieler“ zu entwickeln würde mehr Zeit gebraucht werden. Auch die fehlenden bzw. nicht ausreichenden Kenntnisse der Programmiersprache MATLAB stellten ein Problem da. Daher wurden die Hauptaspekte in der Planung daraufgelegt, sich an dem Min-Max-Algorithmus (Abb.1) zu orientieren. Diese Art von Algorithmus wird in vielen solcher Spiele verwendet. Dazu gehören zum Beispiel „Schach“ oder „Tic Tac Toe“. Bei dem Min-Max Algorithmus werden der Spielsituation Werte zugeordnet. Durch diese Werte kann die beste oder schlechteste Position ermittelt werden, an der der Spielchip oder ein Kreuz (Tic Tac Toe) gelegt bzw. gesetzt werden kann.
Diese Art von Algorithmus funktioniert jedoch nur, wenn auch der menschliche Spielpartner versucht perfekt zu spielen. Mit dem Min-Max Algorithmus wäre es auch möglich einen perfekten Spielpartner zu entwickeln, der 42 Züge in die Zukunft schauen kann. Dennoch bietet er die Möglichkeit den Roboter nicht ganz so klug zu programmieren und ihn nur drei bis fünf Züge in die „Zukunft blicken“ zu lassen. Es könnte selber bestimmt werden, wie intelligent der Roboter letztendlich programmiert wird. Da in der Planungsphase die zeitliche Einteilung der Realisierung noch nicht absehbar war, erwies er sich als geeignetes Mittel, für unser LEGO Mindstorms Projektseminar.



(Abb. 1) Der Min-Max Algorithmus

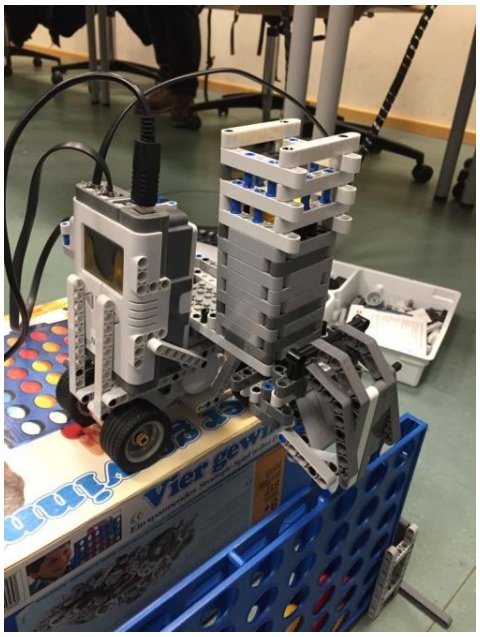
Die Chips sollten in einem Spielsteinmagazin oder Lager vor dem Spiel aufgefüllt werden, sodass sie ohne Eingreifen des menschlichen Spielpartners nachgeladen werden. Die Spielsteine sollten übereinander liegen, damit sie von alleine nachfallen und nicht extra ein „Stopper“ oder eine andere Vorrichtung benötigt wird. Diese Bauweise könnte einen reibungsfreien Ablauf des Abwurfes ermöglichen.
Um den Connect4-Roboter möglichst „alltagstauglich“ zu bauen, sollte er bestenfalls stabil und nicht zu schwer sein. Die Form und Größe sollten so gewählt werden, dass er auch von Kindern oder älteren Personen leicht transportiert werden könnte. Der Roboter sollte zudem aus mehreren, einzelnen Teilen bestehen, sodass er schnell und einfach auf und ab zubauen wäre. Auch sollten die Vorrichtungen für den Menschen leicht zugänglich sein, damit er bei Fehlern oder Versagen des Roboters schnell und einfach eingreifen könnte.
Insgesamt müsste die Vier-Gewinnt spielende Anlage sehr präzise arbeiten, um einen möglichst flüssigen Spielfluss zu erzielen.
Die benötigten Bauteile wären laut diesem Konzept, der NXT, das Gehirn des Roboters, drei interaktive Servomotoren für die Bewegungen in X- und Y- Richtung und der Farbsensor. Zusätzlich wird natürlich ein Vier-Gewinnt-Spiel benötigt mit Spielsteinen in zwei unterschiedlichen Farben und Lego Bausteine zum konstruieren des Roboters. Darüber hinaus müsste ein Podest gebaut werden auf dem der Roboter sicher stehen und fahren könnte. Ob dieses aus Lego Bausteinen errichtet werden sollte oder gegebenenfalls aus anderen Materialen, wie z.B. Büchern, müsste noch in der Gruppe diskutiert werden.

## *Die Realisierung*

Die Einführung von der Programmiersprache MATLAB und die Planung des Konzeptes wurden in der ersten Woche des zwei wöchigen Praktikums durchgeführt. In der zweiten Woche musste nun angefangen werden die Idee des Roboters praktisch umzusetzen. Hierzu bekamen die einzelnen Gruppen die Lego Mindstroms Baukästen. In diesen Kästen waren die Sensoren, Motoren, Kabel, der NXT und die normalen Lego Bausteine enthalten. Es wurden jeder Gruppe drei interaktive Servomotoren, zwei Tastsensoren, ein Farbsensor und ein Lichtsensor zur Verfügung gestellt.

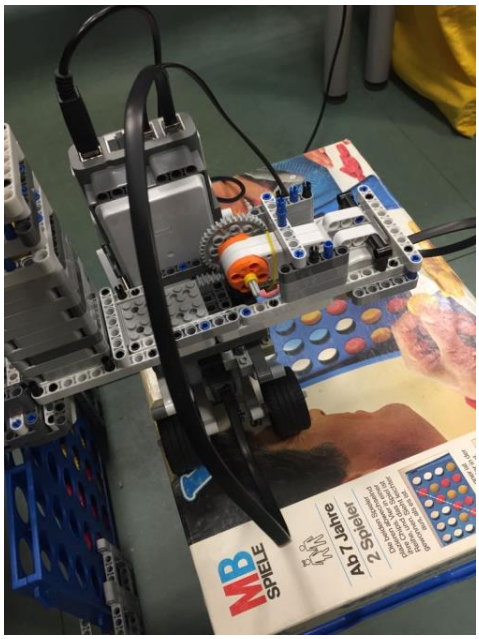
Begonnen wurde mit der Konstruktion des Bauteils, an dem später der Farbsensor angebracht werden sollte. Dafür benötigte man einen von den drei Motoren und ein System von Zahnrädern, welches ermöglichen sollte, den Farbsensor präzise in jedes einzelne Feld steuern zu können. Dabei traten viele Probleme auf. Anfangs war die Konstruktion nicht stabil genug, um ihr Eigengewicht mit Farbsensor zu tragen. Der Arm musste im Verhältnis zum Grundgerüst des Roboters sehr lang sein, damit er auch die unterste Zeile des Spieles erreichen konnte. Bei dem hoch und herunter fahren des Armes blieb das Kabel, welches die Verbindung zwischen NXT und Farbsensor war, oft an den Bauteilen hängen und riss den Arm auseinander. Deswegen wurde nach den aufgetretenen Problemen nach einer Alternative gesucht. Diese Alternative musste schnell gefunden werden, denn der Bau des eigentlichen Roboters hatte noch nicht begonnen. Herr Magdowski gab dann den Tipp eine Webcam zu verwenden, die immer wieder die Spielsituation neu aufzeichnen würde. Mit ihr und ein paar Kenntnissen im Thema Bildbearbeitung in MATLAB wäre es möglich, den vom menschlichen Spielpartner eingeworfenen Chip ausfindig zu machen. Aus diesem Grund wurde der Bau des Armes nicht weiter fortgesetzt und es begann die Konstruktion des eigentlichen Roboters.
Hierbei stellte sich zuerst die Frage, ob der Roboter auf Rädern, Ketten oder Schienen gebaut werden sollte. Da die Konstruktion auf Rädern als zeitlich schnellste und sicherste Variante erschien, wurde der Connect4-Roboter mit vier Rädern ausgestattet. Diese vier Räder wurden mit einem Motor angetrieben, was bedeutete, dass der Roboter nicht in der Lage sei zu lenken, da dafür mindestens zwei Motoren benötigt werden. Auf diesem Gerüst von Moter und Rädern wurden auf der einen Seite der zweite Motor für die Abschussvorrichtung und auf der anderen Seite das Chipmagazin angebracht. Diese Konstruktion hatte positive Auswirkungen auf die Statik des Roboters, da sowohl das Spielsteinlager und der Motor ein hohes Eigengewicht hatten. Dieses Gleichgewicht wirkte sich wiederum positiv auf die gradlinige Bewegung des gesamten Roboters aus.
Das Magazin für die Chips wurde sehr hoch konstruiert, damit es auch alle 21 Chips beinhalten konnte. Zuerst wurden in dem „Turm“ aus Legobausteinen Lücken gelassen, damit bei falschen herabfallen, eingegriffen werden könne. Diese Bauart erwies sich später als Fehlerquelle und wurde deshalb zum Teil korrigiert. Daher hatte das Lager, seit dem Umbau, den Nachteil, dass die Spielsteine mit großem Zeitaufwand und großer Vorsicht in das Lager eingelassen werden mussten. Lag erst einmal ein Chip schief in dem Magazin, der ohne Abbau einzelner Teile nicht zu erreichen war, so musste das ganze Lager abgebaut werden.

Auch bei der praktischen Umsetzung der mechanischen Einwurfvorrichtung traten mehrere Komplikationen auf. Sie wurde auf der äußeren Seite des Chipmagazins angebracht. Zuerst hatte die Vorrichtung einen zu großen Abstand zu dem Magazin. Durch diesen Abstand erreichten viele Spielsteine nicht einmal die Vorrichtung. Sie blieben auf dem Lauf einfach liegen, da nicht bei allen Stößen der Abwurfvorrichtung, dieselbe Kraft an den Steinen umgesetzt wurde. Das Problem, dass zwar alle Steine mit derselben Kraft aus dem Lager herausgedrückt wurden aber der Schwung bzw. die Geschwindigkeit der Spielchips stark variierte, werde ich an späterer Stelle noch genauer erläutern. Deshalb wurde nun die Einwurfvorrichtung direkt am Lager befestigt, damit eine höhere Trefferquote erzielt werden konnte. Auch der Spalt aus dem die Chips rutschen und in das Spielen fallen sollten musste sehr präzise konstruiert werden. Dafür wurden auch mehrere Anläufe benötigt.


(Abb. 2) Das Spielsteinlager und die Einwurfvorrichtung

Die Abschussvorrichtung, welche auf der anderen Seite arretiert wurde, musste mit einem Motor betrieben werden. Dieser lag waagerecht auf dem Grundgerüst des Roboters. An dem Motor wurde ein Zahnrad montiert. Das Zahnrad wurde auf eine Zahnschiene gedrückt, die bei rotieren des Motors bewegt werden konnte. An dieser Zahnschiene wurde eine Platte aus Legosteinen angebracht, welche die Breite der Spielsteine besaß. Diese Vorrichtung schob die Platte in das Spielsteinlager hinein und somit immer einen Chip heraus. Nach diesem Vorgang, musste die Platte mit demselben Tacholimit zurückgezogen werden, damit ein Spielstein nachfallen konnte. Dadurch war der Roboter in der Lage, die Spielsteine einzeln aus dem Lager heraus zu „schießen“. Auch hier tauchten natürlich Probleme auf. Oft blieb die Platte in dem Spielsteinlager hängen, weil die Chips zu schnell nachgefallen sind. Die Konstruktion von Zahnschiene und Platte war nicht sehr stabil und brach des Öfteren auseinander. Das größte Problem jedoch war, dass die Spielchips kleine Zacken an ihren Rändern hatten und sich so untereinander verkanten konnten. Waren sie verkanntet wurden sie oft entweder gar nicht aus dem Lager herausbefördert oder mit sehr großer Geschwindigkeit. Im Falle des Letzteren, konnte auch die Platte nicht immer zurückgezogen werden, da sie durch die ruckartige Bewegung bei dem Lösen des Steines zu weit nach vorne befördert wurde.

Dadurch hatte das Zahnrad nicht mehr genug Angriffsfläche auf der Schiene, um diese wieder nach hinten zu ziehen. Trotz dieser Schwierigkeiten, erzielte die Abschussvorrichtung eine hohe Trefferquote.



(Abb. 3) Die Abschussvorrichtung

Nachdem der grobe Bau des Roboters am Mittwochnachmittag abgeschlossen war, musste sich nun mit der Webcam und der Bildverarbeitung in MATLAB auseinandergesetzt werden.
Leider brachte auch diese Idee viele Komplikationen mit sich. Das Kalibrieren der Webcam und dem Vier-Gewinnt Spiel dauerte immer sehr lange, da die Bildpunkte immer genau in den einzelnen Spielfeldern platziert werden mussten. Bei der kleinsten Berührung an der Webcam oder dem Spiel selbst, musste sie neu eingestellt werden. Mit den Bildpunkten, die auf die Größe der Matrix bzw. die des Spiels abgestimmt waren, war es möglich die Farben in den Feldern zu unterscheiden. Änderten sich jedoch nur geringst die Lichtverhältnisse in der Umgebung, so war das Programm nicht mehr in der Lage, die roten und gelben Chips auseinander zu halten. Dies führte wiederrum zu zeitlichen Blockaden. Aus diesen Gründen dauerte es bis Donnerstagabend an, bis unser Connect4-Roboter zum ersten Mal selbstständig spielen konnte. Da es uns erlaubt wurde, den Roboter mit nach Hause zunehmen, um ihn fertig zu stellen, drehten wir unser Back-Up Video bei mir in der Wohnung. Dort waren immer gleiche Lichtverhältnisse gegeben und es stand mehr Platz zur Verfügung.
Durch die aufgetretenen Komplikationen, war es leider nicht möglich, den Roboter bei den Abschluss-Präsentationen selbstständig ein Spiel, spielen zu lassen. Was natürlich sehr bedauerlich war.
Auch das Programm konnte in der vorhandenen Zeit, nicht komplett fertig gestellt werden. Das Ziel den Roboter fünf Züge in die Zukunft blicken zu lassen wurde demnach nicht erreicht. Der Roboter war in der Lage, sich Vier Spielsteine in einer Reihe zu erarbeiten und den Gewinn des Gegners zu verhindern.

## *III. Zusammenfassung und Fazit*

Abschließend lässt sich sagen, dass der Bau des Connect4-Roboters mit einem positiven Eindruck beendet wurde. Natürlich gibt es viele Dinge, die bei genügend Zeit, noch verbessert werden könnten. Die Abschuss und Einwurfvorrichtung könnte effizienter gestaltet werden. Zudem wäre es sinnvoll den kompletten Roboter alltagstauglicher zu konstruieren. Das Programm müsste überarbeitet und verbessert werden, damit er noch intelligenter spielen könnte. Insgesamt war das Lego Mindstorms Praktikum eine interessante und aufschlussreiche Veranstaltung.

*Literaturverzeichnis:*

Abb.1
https://www3.ntu.edu.sg/home/ehchua/programming/java/images/GameTTT_minimax.png

Abb. 2-3
Eigene Aufnahmen

# Entwicklung eines Vier-Gewinnt Roboters

Hannes Schreiber, Elektrotechnik und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—In diesem Paper soll die Entwicklung eines selbstständig spielenden Vier-Gewinnt Roboters beschrieben werden. Ausgehend von der mathematischen Spieltheorie wird ein Algorithmus entwickelt, mit dem der Roboter intelligent spielen soll. Anschließend wird die Realisierung des Einwurfmechanismus sowie der Erkennung der Spielsteine mittels Webcam erläutert. Die Funktionalität des Roboters wird an einem menschlichen Gegenspieler getestet.**

***Schlagwörter*—Künstliche Intelligenz, Lego Mindstorms, Projektseminar, Spieltheorie, Vier Gewinnt**

## I. Einleitung

KÜNSTLICHE Intelligenz spielt heutzutage eine zentrale Rolle bei Computerspielen. Häufig wird nicht gegen menschliche Gegner, sonder gegen Computergegner gespielt. Dabei ist es für den eigenen Spielspaß und für ein realistisches Spielgefühl besonders wichtig, dass der künstliche Gegner ein möglichst menschliches Spielverhalten aufweist. Einen Bereich von Computerspielen bilden dabei Gesellschaftsspiele, die gegen einen Computergegner gespielt werden. Bei solchen Spielen kommt häufig die mathematische Spieltheorie zum Einsatz, um die künstliche Intelligenz zu entwickeln. Beispiele für solche Computerspiele sind Schach, Poker oder auch das in diesem Paper behandelte Vier-Gewinnt. Für gewöhnlich findet dabei das Spielerlebnis lediglich auf dem Bildschirm statt. In unserem Projekt sollte die Spielerfahrung dahingehend erweitert werden, dass wie beim echten Spiel auf einem realen Spielfeld mit physischen Spielsteinen gespielt werden kann. Dazu wurde ein Roboter entwickelt, der mittels einer Webcam die aktuelle Position aller Spielsteine auf dem Feld erkennt. Anschließend wird der nächste Spielzug berechnet, den der Roboter dann selbstständig ausführt, indem er zur jeweiligen Position fährt und seinen Stein einwirft.

## II. Vorbetrachtungen

Zur Entwicklung einer künstlichen Intelligenz wurden Ansätze aus der mathematischen Spieltheorie genutzt. Dazu muss das behandelte Spiel zunächst genauer beschrieben werden.

### A. Das Spiel Vier-Gewinnt

Bei Vier-Gewinnt spielen zwei Spieler gegeneinander und versuchen, 4 Steine ihrer Farbe horizontal, vertikal oder diagonal in eine Reihe zu bekommen. In Abbildung 1 ist ein Spielfeld mit einer Spielsituation, in der der Spieler mit den roten Steinen gewinnen würde, dargestellt. Es handelt sich um ein endliches Zwei-Personen-Nullsummenspiel mit perfekter Information. Endlich bedeutet, dass es einen Maximalwert an

DOI: 10.24352/UB.OVGU-2018-060 

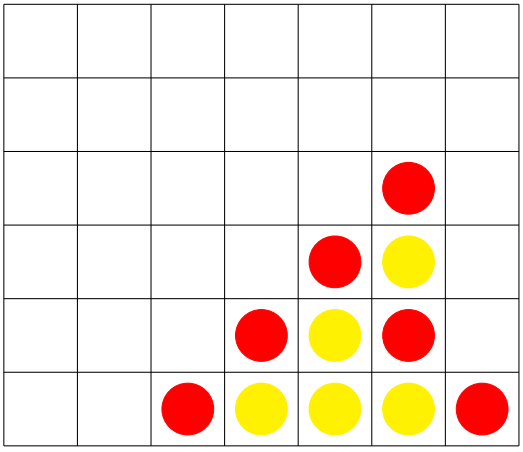


Abbildung 1. Vier-Gewinnt Spielfeld mit einer beispielhaften Spielsituation

Zügen gibt, nach denen das Spiel beendet ist. Nullsummenspiel beschreibt ein Spiel, bei dem die Summe der Gewinne und Verluste beider Spieler zusammen null ist. Ein Gewinn für den einen Spieler bedeutet also den gleichen Verlust für den anderen Spieler. Bei einem Spiel mit perfekter Information ist die komplette Spielsituation jederzeit für alle Spieler einsehbar (beispielsweise im Gegensatz zu Poker).

### B. Bewertung einer Spielsituation

Zur Nutzung von Algorithmen aus der Spieltheorie ist es nötig, verschiedene Spielsituationen zu bewerten. Sei $\mathbf{A}$ eine 6x7 Matrix, die das Spielfeld mit den aktuellen Spielsituation repräsentiert. Für die Situation aus Abbildung 1 würde die Matrix beispielsweise folgendermaßen aussehen:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 & 1 & 1 & -1 \end{bmatrix}$$

Welche Zahl welcher Spielfarbe zugeordnet wird, ist dabei zunächst frei wählbar, hier wurde für die roten Steine die Zahl -1, für die Gelben die Zahl 1 und für ein leeres Feld die Zahl 0 gewählt. Eine ideale Bewertungsfunktion $f$

$$g = f(\mathbf{A}) = \begin{cases} 1, & \text{Spieler 1 gewinnt} \\ -1, & \text{Spieler 2 gewinnt} \\ 0, & \text{Unentschieden} \end{cases} \quad (1)$$

würde einer Spielsituation $\mathbf{A}$ den Wert $g$ zuordnen. Damit der Algorithmus Züge die möglichst schnell zum Sieg führen bevorzugt, wird die Bewertung um einen Faktor erweitert.

$$g = f(\mathbf{A}) \cdot (43 - n) \tag{2}$$

Dabei gibt $n$ die Nummer des jeweiligen Zuges an (von 1 bis maximal 42). Je früher ein möglicher Sieg im Spiel auftritt, desto höher wird dieser also bewertet. In der Praxis ist es jedoch häufig zu aufwändig, alle möglichen Spielsituationen zu berechnen und zu bewerten. Daher werden von der aktuellen Spielsituation ausgehend die möglichen Spielsituationen nur für eine bestimmte Anzahl an Zügen betrachtet. Eine mögliche Erweiterung besteht jetzt noch darin, ein Unentschieden genauer zu bewerten, also abzuschätzen, ob die jeweilige Spielsituation eine gute oder schlechte Ausgangssituation für die weiteren Züge darstellt. Somit wird auch einem Unentschieden ein Wert ungleich 0 zugeordnet und $f$ hat mehr als 3 mögliche Ergebnisse.

### C. MinMax-Algorithmus

Wenn die möglichen Spielsituationen einmal bewertet sind, kann der MinMax-Algorithmus genutzt werden, um den besten aktuell möglichen Zug zu ermitteln. Der MinMax-

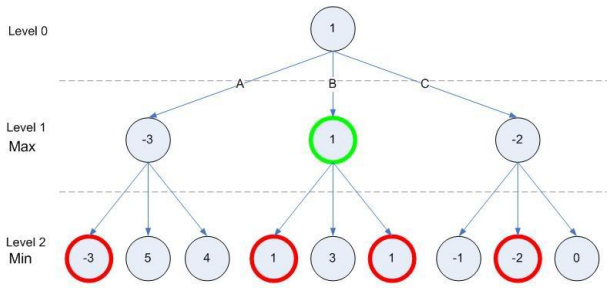



Abbildung 2. Beispielhafter Suchbaum für den MinMax-Algorithmus

Algorithmus basiert auf der Annahme, dass jeder Spieler stets versucht, seinen Gewinn zu maximieren bzw. seinen Verlust zu minimieren. Wenn der Spieler, der gerade an der Reihe ist, seinen Zug plant, kann er also davon ausgehen, dass er selbst stets die beste Situation, also die mit dem höchsten Wert für $g$, wählt. Er selbst will ja seinen Gewinn maximieren. Der Gegner wird stets die Situation mit dem niedrigsten Wert für $g$ wählen, da er ja seinen Verlust minimieren will. Das führt dazu, dass in dem Suchbaum stets abwechselnd der niedrigste und der höchste Wert gewählt wird. Ein solcher Suchbaum ist in Abbildung 2 beispielhaft dargestellt. Es ist dazu zu sagen, dass aus Platzgründen nur ein Suchbaum mit einer Breite von 3 dargestellt ist. Das bedeutet, dass es bei jedem Zug 3 Möglichkeiten gibt. Bei Vier Gewinnt wäre es ein Suchbaum mit einer Breite von 7, jedoch ändert das nichts am Algorithmus selbst. Die Suchtiefe beträgt in diesem Beispiel den Wert 2. Man sieht, dass für jeden Zweig zunächst der minimale Wert der darunter liegenden Möglichkeiten übernommen wird. Anschließend wird für die Ausgangssituation der maximale Wert der 3 darunter liegenden Möglichkeiten übernommen. In diesem Beispiel würde der Algorithmus also den Zug B als beste Möglichkeit auswählen. [1]

## III. Hauptteil

Der Roboter sollte also nicht nur ausgehend von der aktuellen Spielsituation einen Spielzug wählen, sondern auch die Steine selbstständig einwerfen. Dazu waren verschiedene Arbeitsschritte nötig.

### A. Das Anfahren einer Spielposition

Damit der Roboter eine bestimmte Spielposition anfahren kann, muss er sich horizontal entlang des Spielfeldes bewegen können. Dazu wurde der Roboter mit vier Rädern ausgestattet, welche über einen Motor angetrieben wurden. Damit ein einwerfen von oben möglich war, wurde aus Kartons eine Plattform gebaut, auf welcher der Roboter fahren konnte. Es war von großer Wichtigkeit, dass die Bewegung des Roboters parallel zum Spielfeld erfolgt, da sich ansonsten der Abstand des Roboters zu den Schlitzen, in die die Spielsteine geworfen werden müssen, verändert, sodass das Einwerfen der Spielsteine nicht mehr funktioniert. Es hat sich gezeigt, dass trotz großer Sorgfalt kein exaktes paralleles Fahren realisiert werden konnte. Da sich beim Hin- und Herfahren die Fehler gegenseitig ungefähr aufheben, konnte trotzdem ein sicheres Einwerfen der Spielsteine realisiert werden. Es war jedoch wichtig, den Roboter zu Beginn des Spiels möglichst parallel zum Spielfeld hinzustellen. Im Programm wurde das Anfahren so realisiert, dass eine Funktion geschrieben wurde, die als Parameter die aktuelle Position und die vom Algorithmus berechnete, anzufahrende Position übergeben bekommt und daraus die Strecke ermittelt, die gefahren werden muss.

### B. Das Einwerfen der Spielsteine

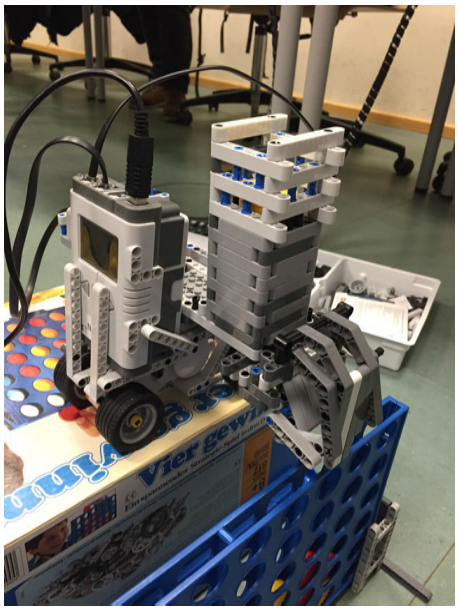


Abbildung 3. Das Spielsteinlager

Ist der Roboter zu der gewünschten Spielsituation gefahren, muss er einen Spielstein in den vorgesehenen Schlitz des Spielfeldes werfen. Damit der Roboter ein gesamtes Spiel ohne menschliches Eingreifen spielen kann, wurde ein Lager gebaut,

das Platz für die 21 Spielsteine hat. In Abbildung 3 ist das Spielsteinlager zu sehen. Aus diesem Lager sollte nun immer ein Stein eingeworfen werden. Das erwies sich als besonders schwierig, da die Spielsteine, da sie natürlich nicht für solch eine Nutzung gedacht sind, kleine Einkerbungen haben, damit sie sich ineinander verhaken. Somit konnte der unterste Stein im Lager nicht einfach herausgeschoben werden, da er durch die Einkerbungen und den Druck der darüber liegenden Steine verhakt war. Als Lösung wurde ein Schieber gewählt, der,

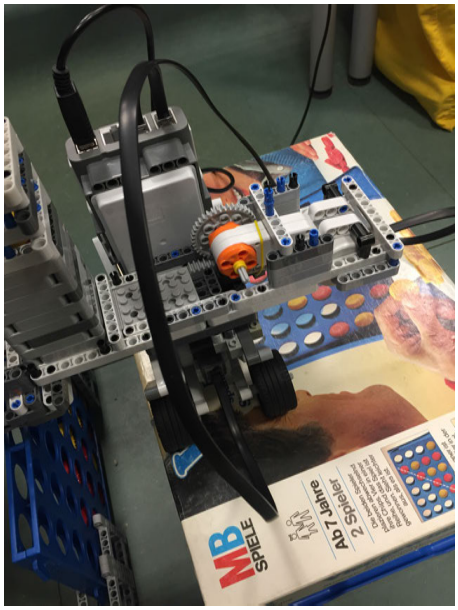



Abbildung 4. Die Abwurfvorrichtung

angetrieben durch einen Motor, einen sehr ruckartigen, starken Stoß ausführt. Problematisch war hierbei, dass die nötige Kraft für das Herausschießen des untersten Steines mit steigender Spieldauer sinkt, da der Druck auf den Stein kleiner wird, wenn weniger Spielsteine über ihm liegen. Der Motor für den Schieber musste also eine Drehmoment aufwenden, dass groß genug war, um den untersten Stein herauszuschießen. Jedoch wurden die Spielsteine so eben mit sinkender Anzahl der Reststeine im Lager immer weiter herausgeschossen. Um trotzdem bei jedem Zug den Schlitz zu treffen, wurde eine Vorrichtung gebaut, die die Flugbahn der Spielsteine so umlenkt, dass sie immer im Schlitz endet. Die Abwurfvorrichtung ist in Abbildung 4 dargestellt.

### C. Das Erkennen der aktuellen Spielsituation

Das Erkennen der Spielsteine sollte zunächst durch einen Farbsensor realisiert werden. Dazu sollte der Sensor an einem Arm befestigt werden, der sich, angetrieben durch einen Motor, vertikal entlang des Spielfeldes bewegt und so die Position der Spielsteine und deren Farbe erkennt. Die Idee war dabei, dass sich das Programm die vorherige Spielsituation merkt, sodass nicht alle Felder, sonder nur die Felder, in denen ein Stein liegen könnte, abgefahren werden. Im Laufe des Seminars wurde jedoch entschieden, dass das Erkennen der Spielsituation mit Hilfe einer Webcam realisiert werden soll. Das brachte eine Erleichterung im mechanischen Teil des Projekt mit sich, da keine Vorrichtung zum Abfahren der Felder gebaut werden musste. Andererseits brachte es zusätzliche Arbeit im Bereich der Programmierung, da zusätzlich eine Bildverarbeitung durchgeführt werden musste. Für die Kalibrierung der Kamera wurde ein recht einfacher Weg gewählt. Für den Nutzer wurde ein Kamerabild mit markierten Bildpunkten angezeigt. Zur Kalibrierung musste der Nutzer nun die Webcam so ausrichten, dass die Markierungen genau in den Feldern lagen. Im Programm wurden nun die Helligkeitswerte an den einzelnen Markierungen überprüft und daraus konnte entschieden werden, ob es sich um einen roten oder gelben Spielstein oder um ein leeres Feld handelt. Zur besseren Erkennung des leeren Felder wurde ein schwarzes Blatt hinter das Spielfeld geklebt.

### D. Das Finden des besten Zuges

Aus Zeitgründen wurde ein eher einfacher Algorithmus gewählt. Der Algorithmus erkennt, wenn er eine Gewinnsituation hat und nutzt diese, außerdem erkennt er die Gewinnsituation des Gegners und verhindert diese. Sollte nichts davon der Fall sein, wird versucht, den eigenen Stein an eine bereits bestehende Reihe anzureihen, es wird also versucht, eine Gewinnsituation aufzubauen. Das entspricht einem MinMax-Algorithmus mit einer Suchtiefe von 2.

### E. Das gesamte Programm

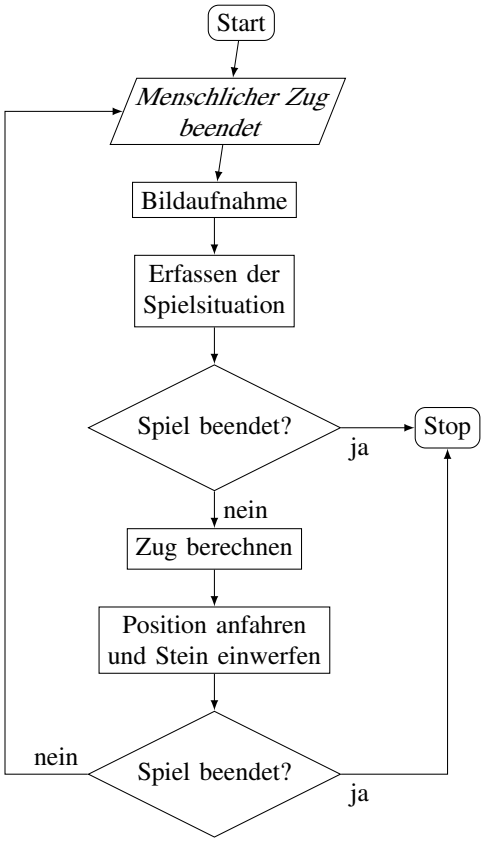



Abbildung 5. Programmablaufplan für das Spielprogramm

In Abbildung 5 ist ein Programmablaufplan für das gesamte Programm dargestellt. Nachdem die Kamera kalibriert wurde

wird das Spiel gestartet. Der Gegner muss über eine Benutzeroberfläche signalisieren, wenn er seinen Zug beendet hat. Das Programm wartet solange auf die Eingabe 'Menschlicher Zug beendet'. Anschließend wird mittels Webcam die aktuelle Spielsituation erfasst. Danach wird geprüft, ob der Gegner bereits gewonnen hat. Falls das der Fall ist, stoppt das Programm. Falls das nicht der Fall ist, berechnet das Programm den bestmöglichen Zug, fährt zu dieser Position und wirft einen Stein ein. Anschließend wird geprüft, ob der Roboter gewonnen hat. Sollte das der Fall sein, wird das Programm gestoppt. Ist das nicht der Fall, wird wieder gewartet bis der menschliche Spieler signalisiert, dass er seinen Zug beendet hat.

## IV. Ergebnisdiskussion

Am Ende des Seminars war ein Roboter entwickelt worden, der gegen einen menschlichen Spieler selbstständig spielen konnte. In einem Testspiel schaffte es der Roboter, die Testperson zu besiegen, wobei die Testperson allerdings absichtlich Gewinnchancen des Roboters nicht verhinderte. Es zeigte sich, dass der Roboter erfolgreich Gewinnsituationen des Gegners verhinderte und eigene Gewinnsituationen kreierte und diese dann auch nutzte. Große Probleme bereitete zunächst die Konstruktion der Einwurfvorrichtung. Es erwies sich aufgrund der in Unterabschnitt III-B beschriebenen ungleichmäßigen Flugweiten der Spielsteine als schwierig, eine Einwurfvorrichtung zu entwickeln, die unabhängig von der Anzahl der bereits getätigten Züge immer den Schlitz trifft. Dazu wurde eine Schiene konstruiert, die die Flugbahn der Steine lenken sollte. Letztendlich konnte so eine sehr hohe Trefferquote erreicht werden. Ein weiteres Problem stellte das Spielsteinlager dar. Dieses wurde zuerst mit einigen Öffnungen an der Seite konstruiert. Dies stellte sich jedoch später alsm Fehlerquelle heraus, da die Steine in dieses Öffnungen häufig stecken blieben. Deshalb wurde das Lager so verändert, dass es komplett geschlossen war. Das brachte den Nachteil mit sich, dass das Befüllen des Lagers sehr lange dauerte und dass bei verkanteten Steinen das Lager abgebaut werden musste. Auch die Erkennung der Spielsituation mittels Webcam stellte ein Problem dar. Die Kalibrierung der Kamera dauerte immer sehr lang, da die markierten Bildpunkte genau in den einzelnen Feldern liegen mussten. Außerdem musste die Kamera, nachdem sie selbst oder das Spielfeld auch nur ein kleines bisschen berührt wurde, neu kalibriert werden. Außerdem funktionierte die Erkennung der Farben der Spielsteine nur bei bestimmten Lichtverhältnissen. Änderten sich die Lichtverhältnisse, konnte die Bildverarbeitung nicht mehr zwischen roten und gelben Spielsteinen unterscheiden. Aufgrund dieser Probleme blieb relativ wenig Zeit für den eigentlichen Algorithmus, der den besten Zug ermittelt. Daher ist dieser leider ziemlich einfach gehalten, sodass der Roboter nicht wirklich gut spielen konnte.

## V. Zusammenfassung und Fazit

In dem LEGO Mindstorms Seminar wurde erfolgreich ein Roboter entwickelt, der selbstständig Vier Gewinnt gegen einen menschlichen Gegner spielt. Der Roboter könnte auf verschieden Weisen verbessert werden. Zum Einen könnte die Kalibrierung der Webcam etwas automatisiert werden. Ein Vorschlag eines Betreuers wäre, die Vier Eckpunkte des Spielfeldes durch den Nutzer im Kamerabild markieren zu lassen. Aus diesen vier Punkten könnte dann die Lage der einzelnen Felder berechnet werden. So wäre eine schnelle Kalibrierung unabhängig vom Abstand und Winkel der Kamera zum Spielfeld möglich. Außerdem könnte die Kamera zusätzlich auf die Lichtverhältnisse kalibriert werden. So könnte man bei der Kalibrierung zwei unterschiedlich farbige Steine in zwei festgelegten Felder platzieren. Das Programm könnte dann die Helligkeitswerte dieser Steine ermitteln und die Werte dann, mit einer gewissen Toleranz, als Grenzen für einen roten bzw. gelben Stein nutzen. Desweiteren könnte das Anfahren der Positionen so verbessert werden, dass der Roboter immer parallel zum Spielfeld fährt. Dazu könnte eine Schiene gebaut werden, die parallel zum Spielfeld verläuft und verhindert, dass der Roboter von seinem Kurs abkommt. Außerdem könnte das Spielsteinlager so verändert werden, dass es nicht mehr gerade nach oben sondern schräg verläuft. So würde der Druck auf den untersten Stein durch die darüber liegenden Steine stark verringert werden. Desweiteren könnte des Algorithmus, der den besten Zug ermittelt, verbessert werden. Mit einer richtigen Implementierung des MinMax-Algorithmus wäre es möglich, einen Roboter zu erschaffen, der weitaus besser spielt. Es wäre denkbar, dass die Suchtiefe über eine Benutzeroberfläche verändert werden kann, sodass man einstellen kann, wie gut der Roboter spielt. Bei einer ausreichenden Suchtiefe wäre sogar ein perfekt spielender Roboter denkbar. Außerdem könnte man einbauen, dass der Roboter einen ungültigen Zug des Gegners erkennt. Über den Vergleich der vorherigen Anzahl der Spielsteine und der aktuellen würde der Roboter es erkennen können, wenn der Gegner keinen oder zwei Steine eingeworfen hat.

## Literaturverzeichnis

[1] Wikipedia, *Minimax-Algorithmus*, https://de.wikipedia.org/wiki/Minimax-Algorithmus (Abruf: 10.03.2018)
[2] H. Baier, *Der Alpha-Beta-Algorithmus und Erweiterungen bei Vier Gewinnt*, 2006

# Bau einer autonomen Räumraupe

Bennett Sattler, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Abstract*— In der heutigen Zeit werden mehr und mehr Arbeitsschritte von selbstständigen Robotern übernommen um die Produktion zu beschleunigen und menschliche Arbeitskräfte einzusparen. Aus menschengesteuerten Maschinen werden so autonome Roboter. Wir haben „die Räumraupe“, das Modell eines autonomen Räumfahrzeuges entwickelt, da diese später in der Praxis in Katastrophenfällen helfen können Straßen für andere Hilfsfahrzeuge wieder befahrbar zu machen. In unserer Arbeit ging es nun darum auf Grundlage der Fuzzy-Logik die „Räumraupe“ die Linie verfolgen zu lassen. Sollten sich nun auf diesem Weg Hindernisse befinden so erkennt der Roboter sie rechtzeitig, nähert sich ihnen langsam und testet dann ob er sie verschieben kann oder nicht. Sollte ersteres der Fall sein dann dreht sich der Roboter um 180° und nutzt die Schiebevorrichtung um das Hindernis von der Fahrbahn herunter zu schieben.**

**Als Bausatz nutzen wir das “Lego Mindstorms Education Set“ und entwickelten das dazu gehörige Programm in Matlab. Um die Kommunikation zwischen Matlab und dem Lego-NXT Baustein zu gewährleisten nutzten wir ein Matlabpaket der RTWH Aachen.**

**Als Ergebnis erhielten wir ein funktionstüchtiges Modell und noch einige Anmerkung, auf Basis unseres Entwicklungsprozesses, die erfüllt werden sollten um das Räumfahrzeug in der Praxis nutzen zu können. Zum einen wären es zusätzliche Sensoren um Fußgänger oder andere Dinge zu erkennen die keine Hindernisse sind und zum anderen wäre es eine Satellitensender um zu erkennen welche Fläche bearbeitet werden muss und wo zum Beispiel die Erde hingebracht werden kann.**

***Schlagwörter*—Autonomes Fahren, Fuzzy-Logik, Lego Mindstorms, Matlab, Räumfahrzeug**

## I. Einleitung

Der Einsatz von Robotern in Arbeitsbereichen die relativ leicht automatisiert werden können ist zum Beispiel in der Automobilproduktion nichts neues mehr. Da es aber noch andere simple Tätigkeiten gibt die durch Roboter ausgeführt werden könnten um den Menschen für wichtigere beziehungsweise kompliziertere Tätigkeiten freizustellen, besteht in dieser Richtung noch Handlungsbedarf.

Als mögliches Beispiel sei das Freiräumen von Straßen aufgeführt. Das ist eine, für den Menschen eintönige aber dennoch wichtige Tätigkeit. Hier liegt einer guter Ansatz für den Roboter vor, da die Entwicklung für autonom fahrende Autos schon fortgeschritten ist, bedarf es hier theoretisch nur ein paar Zusätze um aus einem autonomen Auto ein autonomes Räumfahrzeug zu machen.

DOI: 10.24352/UB.OVGU-2018-061 

Genau dort setzt das Projekt „Räumraupe“ an, welches vom 12.02. bis zum 23.02 im Lego Mindstorms Seminar bearbeitet wurde. Ziel war es solch einen Prototypen modellhaft zu erschaffen und damit ein autonomes Räumfahrzeug herzustellen. Dieses Fahrzeug soll, während es einer Linie folgt (als vereinfachte Straße) selber Hindernisse erkennen und testen ob diese, für das Fahrzeug selber, verschiebbar sind oder umfahren werden müssen. Als Hindernisse dienten leere Pappschachteln und beschwerte Dosen, die zum Beispiel als Geröll oder Ähnliches gesehen werden könnten. Der Roboter wird mithilfe des NXT-Education Sets aufgebaut und ausgeführt. Die Programmierung erfolgt über Matlab und von der RTWH-Aachen wurde ein Paket bereitgestellt um die Kommunikation zwischen Matlab und dem NXT-Baustein realisieren zu können. So konnten einzelne Motoren- und Sensoreingänge angesteuert werden.

## II. Vorbetrachtungen

### A. Linienverfolgung

Als Grundlage zur Linienverfolgung diente die Fuzzy-Logik [1] um einen flüssigeren Ablauf zu garantieren und so die Fehlerhaftigkeit zu reduzieren. Der Unterschied zur herkömmlichen Linienverfolgung ist, dass nicht nur zwischen schwarz und weiß unterschieden wird, sondern auch Graustufen hinzugezogen werden (siehe Abbildung 1). Somit kann schneller und feiner auf Abweichungen reagiert werden und daher wurde dieses Verfahren von uns genutzt.

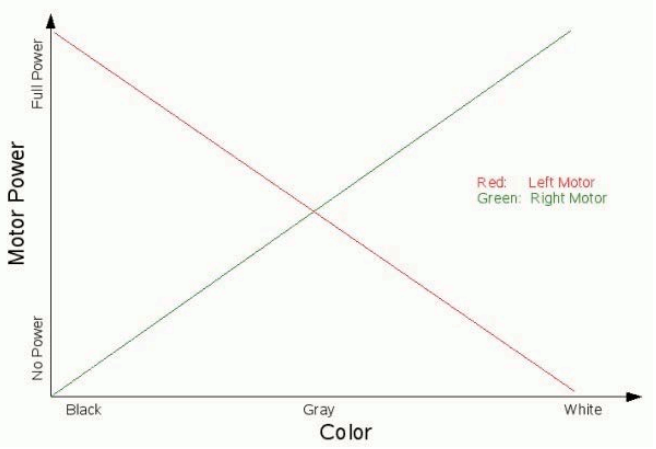



Abbildung 1: Motorenkraft in Abhängigkeit vom Farbwert

Zu beachten ist aber, dass bei mittelmäßigen Lichtverhältnissen in Räumen der Schattenwurf die Lichtsensoren leicht verwirrt und die Linienverfolgung dadurch beeinträchtigt wird.

### *B. Grundlegende Programmierung*

Zur grundlegenden Programmierung wurde die Befehlsübersicht [2] genutzt. In dieser sind neben der Kommunikation zwischen Matlab und dem NXT-Baustein, auch die grundlegenden Programmabläufe, sowie das Einbinden und Nutzen von Funktionen beschrieben. Was aber während des Erarbeitungsprozesses auffiel, war das der Roboter sich oft in einer Schleife fest fuhr und das Problem nicht gefunden werden konnte, da auch kein Error ausgegeben wurde. Es musste eine Maßnahme zur Ausgabe der aktuell ablaufenden Prozesse gefunden werden.

## III. HAUPTTEIL

Als grundlegendes Konzept lag ein Fahrzeug vor welches eine Differenziallenkung besitzt um ein das Manövrieren leichter und die Wendekreise kleiner zu machen. Des weiteren wurde auf einen kompakten, da robusten Bau geachtet denn ein niedriger Schwerpunkt macht das Verschieben von Gegenständen leichter und verringert die Gefahr dass das Fahrzeug umkippt und beschädigt wird. Dies wird in bei diesem Modell nicht der Fall sein, wird aber in der Praxis von Bedeutung sein, wenn der Roboter über unebenes Gelände fahren muss.

Abbildung 2: Seitenansicht der Räumraupe

In Abbildung 2 ist zu sehen, dass alles sehr nah am Untergrund gehalten ist, um den oben genannten Problemen aus dem Weg zu gehen. Nach dem dies festgelegt war folgte als erste Aufgabe, das Fahrzeug die Linie verfolgen zu lassen, da die restlichen Problemstellungen darauf aufbauten.

Wie in Punkt II.A bereits beschrieben programmierten wir nach der Fuzzy-Logik (“verschwommene Logik”). Aber um dem Schattenwurf und den damit einhergehenden Beeinträchtigungen aus dem Weg zu gehen ließen wir den Lichtsensor (in diesem Fall ein “umfunktionierter” Farbsensor) selber ein blaues Licht ausstrahlen und konnten so “schattenfrei” die Farbwerte messen. Um den einzelnen Farbwerten Motorkräfte zuzuordnen, nahmen wir zwei Testwerte (die schwarze Linie und der weiße Untergrund) und ordneten diese jeweils der Motorkraft 0 dem einen und der Motorkraft 54 dem anderen Motor zu. (siehe Anhang 1)

Weiterhin sollten sich die Motorkräften gegensätzlich und gleichmäßig verändern. Das hat zur Folge, dass bei einem sehr hellen beziehungsweise sehr dunklen Untergrund stärker gelenkt wird als bei einem Mittelwert. Der Roboter versucht nun auf der Kante zwischen weiß und schwarz zu fahren. Gegenüber der herkömmlichen Variante wird die Unterlage angestrahlt und verhindert so den größten Teil von Fehlmessung und sollte daher auch später in die Praxis übernommen werden.

Danach ging es darum den Roboter Hindernisse erkennen zu lassen und später diese auf Verschiebbarkeit zu testen. Dafür sind folgende NXT-Sensoren wichtig:

- zwei Ultraschallsensoren
- ein Farbsensor (zur Linienverfolgung)
- ein zusätzlicher Motor
- ein Tastsensor

Abbildung 3: Draufsicht der Räumraupe

Die Ultraschallsensoren befinden sich jeweils unter dem Tastsensor (Abbildung 3 rechte Seite) und einmal seitlich zur Fahrtrichtung (Abbildung 3 untere Mitte). Erstere dient dazu um zu erkennen ob sich ein Hindernis auf der Fahrbahn befindet und letzterer dient wird benötigt um zu erkennen wann der Roboter das Hindernis mit einer Rechtskurve umfahren kann oder ob es sich immer noch auf der Linie befindet.

Der Farbsensor dient wie schon beschrieben zur Verfolgung der Linie und befindet sich daher vorne in der Mitte des Fahrzeugs (Abbildung 3 zwischen dem Tastsensor und dem NXT-Baustein).

Für die Schiebevorrichtung wird noch ein zusätzlicher Motor benötigt um das Objekt nicht nur von dem Roboter vor sich her schieben zu lassen, sondern auch aus dem Weg, das heißt von der Linie wegzuschieben. (siehe Anhang 2)

Das wird dadurch erreicht, dass ab einem bestimmten Punkt im Prozess der Motor aktiviert wird und dadurch die Schiebevorrichtung etwas anstellt, d.h. nicht mehr senkrecht zur Fahrtrichtung, sondern in einem steileren Winkel, sodass das Objekt von der Linie gedrängt wird.

Als letztes wird noch ein Tastsensor benötigt um zu testen ob das Objekt verschiebbar ist oder nicht (darauf wird später genauer eingegangen). Dieser befindet sich vorne über dem Ultraschallsensor (Abbildung 3 rechte Seite). Die Vorrichtung wurde so gebaut das er sich schwerer auslösen lässt, dies

wurde erreicht indem wir zwischen der Kontaktplatte (die mit dem Hindernis in Berührung kommt) und dem Tastsensor ein Bauteil aus Gummi einbauten, welches zusätzliche Kraft erfordert um eingedrückt zu werden und letztendlich diese Kraft weiter auf den Tastsensor zu übertragen und diesen damit auszulösen. Der Aufbau wird in der Abbildung 5 noch etwas genauer dargestellt.

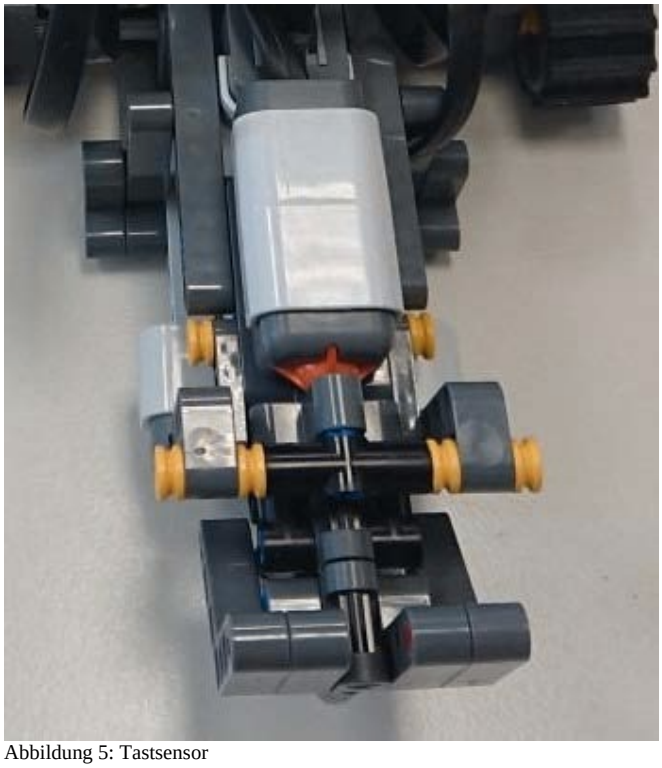


Abbildung 5: Tastsensor

Nachdem der Bau des Fahrzeugs im groben abgeschlossen war ging es an die Programmierung, da es unpassend wäre den ganzen Quellcode zu erklären, wird nur der Inhalt des Quellcodes anhand eines normalen Testablaufes des fertigen Prototyps der “Räumraupe“ erklärt.

Die Räumraupe fährt auf das Hindernis hinzu und sobald der Ultraschallsensor eine Wand erkennt hält diese an und dreht sich um ca. 45°nach links und rechts und misst die Entfernung in kleinen Abschnitte. An dem Punkt an dem die niedrigsten Entfernung gemessen wurde bleibt der Roboter stehen, da er nun senkrecht im Bezug zum Hindernis steht (die Strecke ist im geometrischen Sinn das Lot). Daraufhin fährt der Roboter langsam geradeaus mit gleichbleibender Geschwindigkeit. Sollte innerhalb eines kurzen Zeitfensters nicht der Tastsensor ausgelöst werden (das Zeitfenster reicht aus das Hindernis zu erreichen und kurz dagegen zu fahren), fährt die Räumraupe zurück und dreht sich um 180° um den Schieber nutzen zu können. Nun fährt der Roboter geradeaus (unabhängig von der Unterlage) und verdreht dabei langsam den Schieber um das Hindernis aus dem Weg zu schieben. Nach einer bestimmten Zeit fährt er wieder ein Stück zurück bis die Linie gefunden wurde. Nun dreht sich die Räumraupe wieder um 180° zurück in ihre Ausgangsposition und verfolgt die Linie.

Ist das Objekt nicht verschiebbar so umfährt der Roboter das Hindernis in dem er um 90° (nach links) wendet und solange in diese Richtung fährt bis der Ultraschallsensor einen viel größeren Entfernungswert ausgibt. Dann dreht sich der Roboter um 90° in die andere Richtung (rechts) und dies wird solang wiederholt (immer nach rechts) bis der Farbsensor die Linie wieder findet er sich nun dann wieder um 90° dreht und der Linie weiter folgen kann.

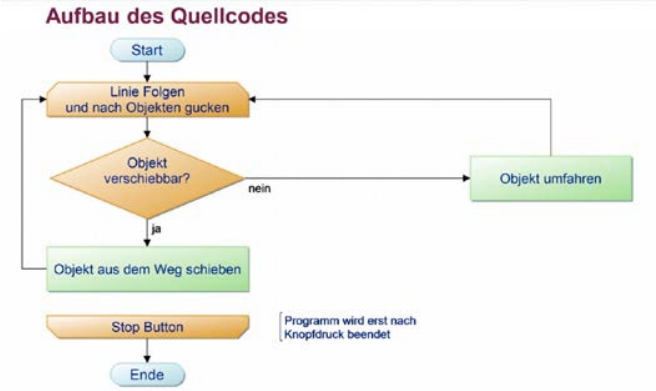



Abbildung 6: Aufbau des Quellcodes

## IV. Ergebnisdiskussion

Herausgekommen ist ein autonomer Roboter der selbstständig seinen Weg findet und den vorgegebenen Weg (soweit ihm möglich) von Hindernissen befreit. (Beispiel: siehe Abbildung 7)

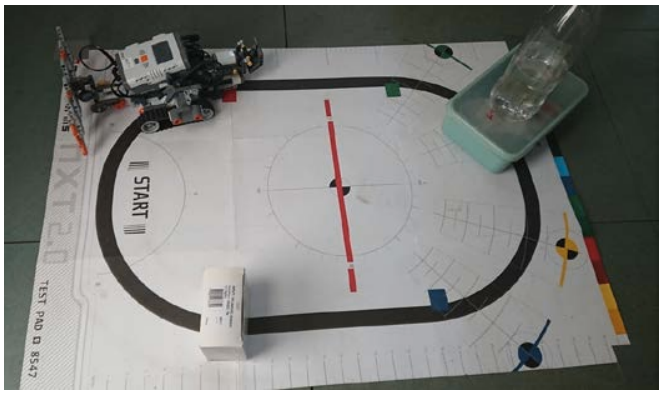



Abbildung 7: Vorführungsaufbau

Als größten Probleme stellten sich 3 Sachverhalte dar.

1. Ungenauigkeiten der Motoren und Sensoren
2. Unklarheiten des aktuellen Ablaufs
3. Umfahren der Hindernisse

Zu 1.: Die Sensoren und Motoren arbeiteten meist nicht so genau wie gewünscht von daher musste, zum Beispiel bei einer Wende des Fahrzeugs die Anzahl der Umdrehungen für jeden Fall spezifisch durch Testen herausgefunden werden und konnte nicht einfach aus der Theorie (sowie es hätte eigentlich möglich sein sollen) umgesetzt werden. Selbiges galt auch für das Messen von Farbwerten oder der Entfernung.

Zu 2.: Anfangs kam es regelmäßig dazu dass der Roboter sich in bestimmten Schleifen “festfuhr“ und so Prozesse immer wieder ausgeführt wurden (zum Beispiel: „suche Wand“). Das dies keinen Error in Mathlab ergab war es schwer in kurzer Zeit fehler zu finden und zu beheben. Deshalb bauten wir in der GUI („Graphical User Interface“) ein Textfeld ein welche die aktuelll laufenden Prozesse ausgab, das heißt dass die einzelnen Schleifen beziehungsweiße Prozesse benannt werden mussten.

Zu 3.: Durch die beiden vorhergehenden Probleme erwieß sich diese Aufgabe als äußerst schwierig. Denn neben der logischen Abfolge musste noch der Wenderadius beachtet werden und nur durch häufiges Testen konnte schließlich das Wenden um genau 90° absolviert werden.

## V. Zusammenfassung und Fazit

Mit dem Verwenden der Fuzzy-Logik als Grundlage zur Linienverfolgung entstand ein autonomer Roboter der anhand eines bestimmten Ablaufes von Fragen und Aufgaben (siehe Abbildung 6) einen gekennzeichneten Weg frei von Hindernissen räumt. Sollte ihm dies nicht möglich sein findet er selbstständig einen Weg und setzt danach seinen alten Weg auf dem gekennzeichneten Abschnitt fort.

Dieser kann als grundlegender Prototyp für autonome Straßenreinigungsfahrzeuge gelten, insofern im Alltag zusätzliche Probleme wie Verkehr oder Fußgänger beachtet werden. Ein wichtiger Faktor ist unserer Meinung nach noch dass außerdem in der Praxis nicht einfach das Hindernis von der Straße auf den Fußgängerweg oder die andere Fahrbahn geschoben werden. So wäre es eine Möglichkeit das Fahrverhalten zu verbessern indem man mithilfe der Satellitentechnik den Verkehr mit einbezieht, mehrere Sensoren anbringt um Fußgänger oder Tiere zu erkennen.

Für die Entscheidungsfindung sollte außerdem noch ein Weg gefunden werden um klar zwischen einem Störfaktor (Äste oder Geröll) und einem normalem Hindernis (Auto oder Motorrad). Des weiteren wäre es nötig in Städten den gefundenen „Dreck“ von der Räumraupe tatsächlich entfernen zu lassen und nicht nur an den Rand zu schieben. Daher wäre ein besserer Einsatz ein Bereich in dem lediglich das bereinigen der Straße gilt und keine Verkehr im Weg steht (zum Beispiel nach einem Erdrutsch auf einer Landstraße).

## Anhang

Aus formatierungstechnischen Gründen sind hier noch zwei Abbildungen angefügt:

| | A | B | C |
|---|---|---|---|
| 1 | Lichtwert | Speed-A | Speed |
| 2 | | | |
| 3 | 27 | 0 | 54 |
| 4 | 36 | 2 | 52 |
| 5 | 45 | 4 | 50 |
| 6 | 54 | 6 | 48 |
| 7 | 63 | 8 | 46 |
| 8 | 72 | 10 | 44 |
| 9 | 81 | 12 | 42 |
| 10 | 90 | 14 | 40 |
| 11 | 99 | 16 | 38 |
| 12 | 108 | 18 | 36 |
| 13 | 117 | 20 | 34 |
| 14 | 126 | 22 | 32 |
| 15 | 135 | 24 | 30 |
| 16 | 144 | 26 | 28 |
| 17 | 153 | 28 | 26 |
| 18 | 162 | 30 | 24 |
| 19 | 171 | 32 | 22 |
| 20 | 180 | 34 | 20 |
| 21 | 189 | 36 | 18 |
| 22 | 198 | 38 | 16 |
| 23 | 207 | 40 | 14 |
| 24 | 216 | 42 | 12 |
| 25 | 225 | 44 | 10 |
| 26 | 234 | 46 | 8 |
| 27 | 243 | 48 | 6 |
| 28 | 252 | 50 | 4 |
| 29 | 261 | 52 | 2 |
| 30 | 270 | 54 | 0 |

Anhang 1: Lichtwerte und dazugehörige Motorenkraft

Anhang 2: Schiebevorrichtung

## Literaturverzeichnis


- Stephan Bracher, “NXT Linienfolger programmiert in Fuzzylogik,” http://www.stefans-robots.net/de/nxt-linienfolger-in-fuzzy-logik.php (Stand: 22.03.18)
- FEIT OVGU (2018, Februar) [Online]. Unterlagen zum Lego Mindstorms Praktikum in der FEIT