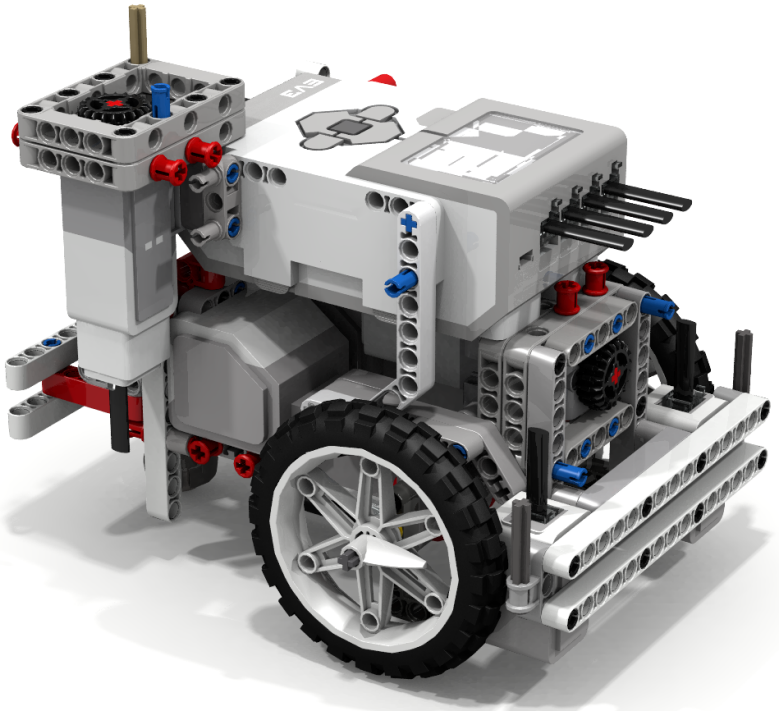
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG

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 sowie Institut für Elektrische Energiesysteme

Herausgeben von:
Mathias Magdowski, Thomas Schallschmidt und Thomas Gerlach

Band 6 vom Wintersemester 2022/2023

Inhaltsverzeichnis

Impressum

Herausgeber:	Mathias Magdowski, Thomas Schallschmidt und Thomas Gerlach Institut für Medizintechnik, Institut für Elektrische Energiesysteme Fakultät für Elektro- und Informationstechnik Otto-von-Guericke-Universität Magdeburg Postfach 4120, 39016 Magdeburg
DOI:	10.24352/UB.OVGU-2023-037
ISSN:	2629-6160
Redaktionsschluss:	April 2023
Seminarzeitraum:	30. Januar – 14. Februar 2023
Bezug:	Open Access, Digitale Hochschulbibliothek Sachsen-Anhalt `http://edoc2.bibliothek.uni-halle.de/`

1. Auflage, Magdeburg, Otto-von-Guericke-Universität, 2023

Erstellung des Sammelbandes mittels LaTeX, `hyperref` und pdfpages

Selbstbalancierender Roboter - Der intelligente Weg zum Ausgleich

Ahmed Abdelgaber Abdou M. Badawy, Mechatronik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Zur Zeit besteht ein wachsendes Interesse für Anwendungsgebiete wie Service-Robotik, Logistik- und Produktionsindustrie. Mit hohen Präzision und Schnelligkeit können intellegente Roboter Aufgabe erledigen, die für Menschen zu gefährlich oder zu anstrengend sind. Da selbstbalancierender Roboter leicht und leistungsfähig, können solche Aufgaben ausgeführt werden.**

Die Structure des selbstbalancierenden Roboter entspricht die Kombination des linearen umgekehrten Pendels und des mobilen Roboters mit Rädern. Da der Roboter von Natur aus unstable ist, wird aus der Regelungstechnik ein Regler verwendet, damit der Roboter auf nur zwei Rädern aufrecht halten kann. Bestehend aus Sensoren und Aktoren kann der Roboter durch ständiges Ausgleichen der vertikalen Neigung ein stabiles Zustand erreichen. Im Rahmen des Projektseminars wurde ein selbstbalancierenden Roboter mit Hilfe eines Gyroscope gebaut und mit MATLAB programmiert.

***Schlagwörter*—zweirädriger Roboter, PID-Regler, Regelungstechnik, Self-balancing Robot, Segwayroboter**

I. Einleitung

INNERHALB der Robotik-Forschung werden immer neue Entwicklungen erstanden, welche zu einem wachsenden Bedarf an Roboter führt. Sie können unterschiedliche Formen und Größen annehemn, von kleinen Armrobotern bis zu großen industriellen Maschinen. Insbesondere in der Lagerverwaltung übernehmen immer Roboter komplexe Aufgaben und stellen maximale Effizienz und Produktivität sicher [1]. Allerdings ist das Problem, Lagerroboter erfordern große Auflagefläche, haben hohe Anschaffungskosten und aufgrund der technischen Komplexität benötigen regelmäßige Wartung und Reparatur. Zur Losung dieses Problems kommt der selbstbalanciernde Roboter zum Einsatz. Selbstbalanciernde Roboten sind sehr flexibel, platzsparend und sehr wendig. Ein gutes Beispiel dafür ist der zweirädrige Roboter Handle, der von Boston Dynamics entwickelt wurde [2]. Die Kombination der Räder und der Gelenke in den Beinen ermöglicht ihm, bis zu 45 Kilogramm zu heben und eine Geschwindigkeit von bis zu 14,5 km/h zu erreichen. Der Roboter wird hauptsächlich zum Entladen von LKW und das Transportieren von Waren eingesetzt.

Innerhalb des LEGO-Mindstorms-Projekts ist ein selbstbalancierender Roboter mit einem Gyroscope zu bauen, der stabil aufrecht bleiben und äußere Störungen ertragen kann. Dieser wird mit MATLAB programmiert und die Konstruktion erfolgte mit LEGO Mindstorms.

DOI:10.24352/UB.OVGU-2023-016

II. Vorbetrachtungen

Aus der Regelungstechnik wird ein PID-Regler zur Stabilisierung des Roboters verwendet. Das mechanische Design ist von entscheidender Bedeutung. Es hat einen Einfluss auf die Stabilität des gesamten Systems.

A. Segwayfunktionsweise

Durch den Einsatz von Gyroskop- und Beschleunigungssensoren kann der Segway das Gleichgewicht halten, indem die Geschwindigkeit der beiden Räder reguliert wird. Die Position und Bewegungen des Fahrers sind kontinuierlich durch die Steuerungselektronik zu überwachen. In Abbildung 1 sieht man, dass der Roboter sich aufgrund des Drehmoments nach vorne umkippen lässt. Dieses Drehmoment wird durch die Gewichtskraft $mg \cdot \sin(\theta)$ erzeugt. Die Motoren kommen hier zum Einsatz, um das Drehmoment entgegenzuwirken und der Roboter aufrecht halten zu können.

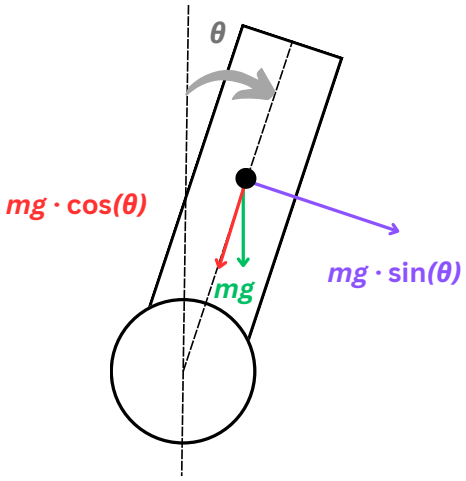


Abbildung 1. Modell eines selbstbalancierenden Roboters

B. PID-Regler

In vielen Bereichen der Technik und Industrie lassen sich Regler eingesetzt. Ein Regler hat die Aufgabe, die Ausgangsgröße eines Systems oder Prozesses zu halten oder zu steuern. Der häufigste ist der PID-Regler, der aus 3 verschiedenen Regler besteht [3]. Der P-Regler steht für den Proportionsanteil und verhält sich proportional zur Regelabweichung. Der I-Anteil minimiert den Fehler im Laufe der Zeit durch Integration der

Regelabweichung. Um schnellere Reaktionen zu erzielen, ist der D-Regler dafür geeignet, der die Ableitung der Regelabweichung nach der Zeit nimmt.

III. UMSETZUNG

A. Aufbau

Der selbstbalancierende Roboter besteht hauptsächlich aus ein LEGO-Stein, Gyroscopesensor und 2 Elektromotoren. Beim Bauen müssen viele Faktoren betrachtet werden. Eine gleichmäßige Verteilung des Gewichts ist entscheidend, um der Roboter stabil zu halten und die Störungen aufgrund des Gewicht zu reduzieren. Dies kann dazu führen, dass das gesamte System zu Schwingung kommt. Bei dem ersten und zweiten Prototyp lag der LEGO-Stein vertikal mit der Radachse. Als der Roboter aufrecht gehalten und losgelassen wird, fällt er viel schnell zu Boden. Das liegt daran, dass eine Reihe von Batterien sich auf der Rückseite des Steins befindet, was aufgrund des Gewichts ein hohes Drehmoment erzeugt. Deshalb wurde die beiden Konstruktionen verworfen. Allerdings beim dritten Prototyp wurde der Stein senkrecht zur Radachse und parallel zum Boden platziert, sodass der Schwerpunkt so nahe wie möglich zum Boden, um eine bessere Stabilität zu gewährleisten. Durch Halten und Loslassen lässt sich deutlich erkennen, dass der Roboter langasmer nach vorne kippt. Zur Bestimmung der Position des Roboters kann ein Gyroscop- oder Lichtsensor verwendet. Anfänglich wurde die Sensoren verbaut und deren Werte aufgezeichent, um zu sehen, welche Sensor bessere Ergebnisse ausgibt. Nach einigen Test konnte man erkennen, dass der Gyroscope genauere Werte aufweist. aber es wird im Laufe der Zeit ungenau wegen des Drifts. Der Gyroscope ist auf der Radachse angebracht, wodurch die Rotationsbewegungen des Roboters in Echtzeit richtig erfasst werden können. Durch die Montage von kleinen Reifen war es deutlich, dass das von den Motoren erzeugte Moment nicht ausreichte, um die Position des Roboters gut genug zu korrigieren. Um das Problem zu lösen, sollten große Reifen verbaut werden, um ein hohes Moment zu gewährleisten [4]. Der endgültige Aufbau ist in Abbildung 5 zu sehen.

B. Funktionsweise des PID-Reglers

Eine zentrale Komponente des selbstbalancierenden Roboters ist der PID-Regler. Der Regler ermöglicht es, dass der Roboter kontinuierlich die notwendigen Anpassungen der Motorleistungen vornimmt, damit das Gleichgewicht des Roboters aufrecht bleiben kann. Ein Regler besteht typischerweise aus drei Anteilen, die jeweils einen individuellen Fehler betrachten. Diese drei Fehler bezeichnet man als Proportionalfehler, Integralfehler und Derivatfehler. Die Definitionen der einzelnen Fehlerarten lautet wie folgt [5]:

$$\text{ProportionalFehler} = \text{Sollwert} - \text{Istwert} \tag{1}$$

$$\text{IntegralFehler} = \text{IntegralFehler} + \text{ProportionalFehler} \tag{2}$$

$$\text{Derivativfehler} = \text{ProportionalFehler} - \text{vorherigeProportionalFehler} \tag{3}$$

Die nächste Formel ergibt sich durch die Multiplikation der einzelnen Fehler mit ihrer jeweiligen Konstante, gefolgt von einer Addition aller Produkte.

$$\begin{aligned} \text{X} &\rightarrow \text{ProportionalFehler} \\ \text{Y} &\rightarrow \text{IntegralFehler} \\ \text{Z} &\rightarrow \text{Derivativfehler} \end{aligned}$$

$$\text{Stellgröße} = (K_\text{P} \cdot \text{X}) + (K_\text{I} \cdot \text{Y}) + (K_\text{D} \cdot \text{Z}) \tag{4}$$

Der Proportionalitätsfaktor K_P kontrolliert die Reaktionsgeschwindigkeit des Reglers auf Abweichungen zwischen der gewünschten Ausgangsgröße und der tatsächlichen Ausgangsgröße. Der Integralitätsfaktor K_I hingegen korrigiert Fehler, die sich im Verlauf der Zeit aufsummieren. Durch Einstellung des Differentialitätsfaktors K_D wird die Geschwindigkeit gesteuert, mit der der Regler auf Änderungen im Fehler reagiert. Um die optimalen Werte für K_P , K_I und K_D zu bestimmen, kann man erst ein Wert für K_P feststellen und dann beobachtet man das Verhalten des Systems. Schließlich stellt man K_I und K_D ein, falls das System eine Regelabweichung hat oder die Reaktiongeschwindigkeit auf Fehler beeinflusst werden soll.

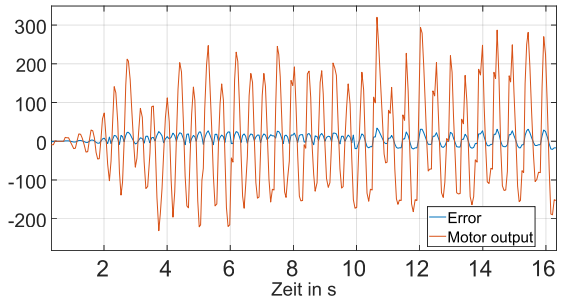


Abbildung 2. Versuch 1 mit $K_\text{P} = 6$ und $K_\text{D} = 0.9$

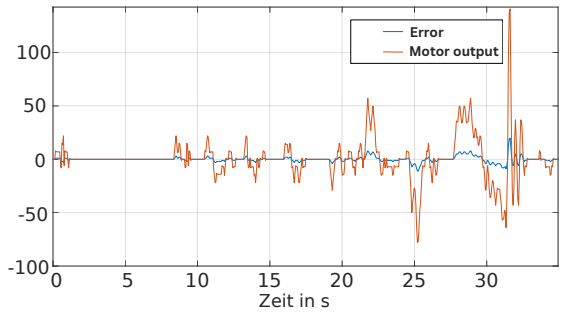


Abbildung 3. Versuch 2 mit K_P = 9 und K_D = 0.4

In Abbildung 2 und 3 werden die Differenz zwischen Soll- und Istwert sowie die Stellgröße dargestellt. Die blaue Kurve repräsentiert den Fehler oder den Eingangswert des Reglers,

während die orangefarbene Kurve die Motorwerte oder den Ausgangswert des Reglers anzeigt. Abbildung 2 zeigt ein kontinuierliches Auftreten von Fehlern, was zu Schwingungen führt. Dadurch wird der Regler stark beeinflusst und instabil. Im Gegensatz dazu sind in Abbildung 3 geringe Fehler und schnelle Reaktionen zu beobachten. Ein erfolgreicher Durchlauf zeichnet sich durch eine orangefarbene Kurve aus, die größer als die blaue Kurve ist und durch ein seltenes Auftreten von Fehlern gekennzeichnet ist.

IV. Programmierung

Zu Beginn wurde die Programmiersprache MATLAB eingesetzt. Allerdings kam es bei Ubermittlung der Daten zu Problemen, da ein Kabel mit dem Stein verbunden war. Dieses Kabel zog den Roboter nach vorne und beeinflusste dadurch das gesamte System als Störgröße. Diese Störung führte zu Regelabweichungen, welche möglicherweise Schwingungen im System hervorrufen konnten. Nach Recherchen wurde festgestellt, dass das MATLAB-Skript nicht auf den LEGO-Brick hochgeladen werden kann. Daher war die LEFO-Software die einzige Lösung, dass das Programm reibungslos ausgeführt werden konnte. Der Algorithmus kann auf beide Softwares angewendet werden.

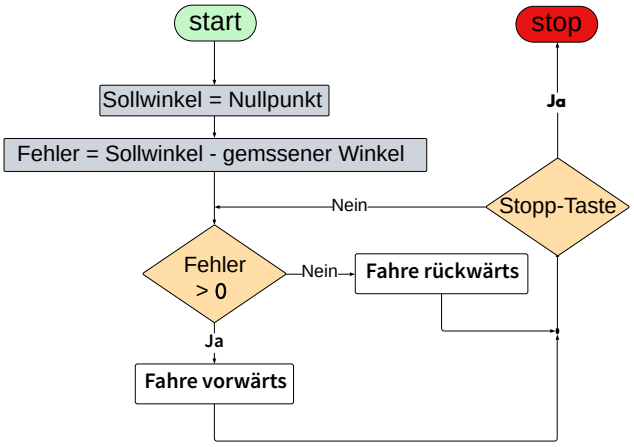


Abbildung 4. Programmablaufplan

In Abbildung 4 wird der Programmablauf für sowohl MATLAB als auch die LEGO Mindstorms Software dargestellt. Am Anfang wird der Winkel oder der Sollwinkel auf 0 gesetzt, um sicherzustellen, dass der Roboter in der aufrechten Position bei Position 0 bleibt. Es gibt zwei Möglichkeiten, um den aktuellen Winkel zu bestimmen. Die erste Möglichkeit besteht darin, die interne Integration der Winkelgeschwindigkeit zu nutzen. Diese Methode kann jedoch aufgrund von Drift zu Fehlern in der Berechnung des Winkels führen, was die Genauigkeit beeinträchtigen kann. Eine genauere Methode zur Berechnung des Winkels besteht darin, die Winkelgeschwindigkeit zu integrieren. Hierbei kann soll die Zeitfunktion definiert werden, um die Zeitdifferenz $(dt = t - to)$ zu berechnen. In einer Endlosschleife lässt sich nun der aktuelle Winkel kontinuierlich abfragen und mit dem Sollwinkel vergleichen. Wenn eine Abweichung zwischen dem aktuellen und dem Sollwinkel besteht, wird dies mithilfe des PID-Reglers korrigiert. Anschließend wird das korrigierte Signal an das System zurückgegeben, um die Position des Roboters entsprechend anzupassen und zu regeln.

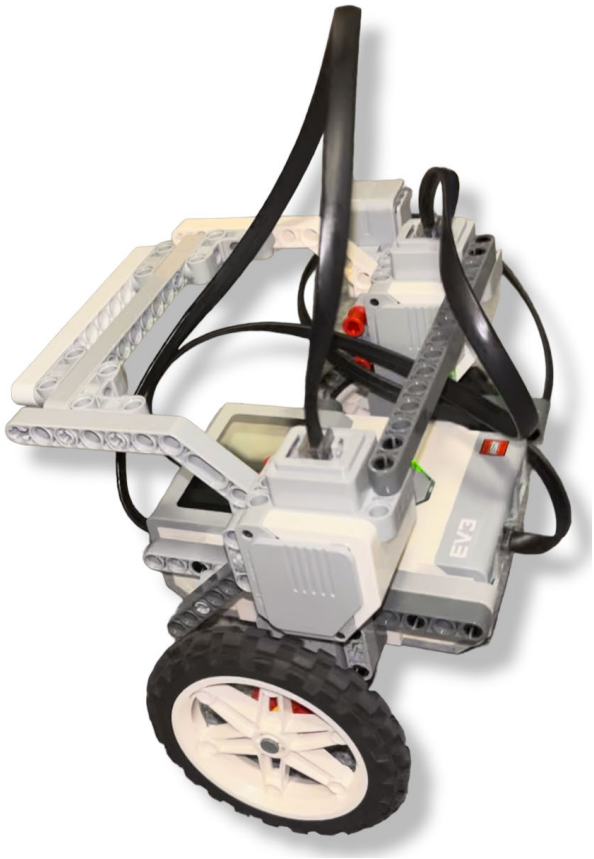


Abbildung 5. dritter Prototyp

V. Ergebnisdiskussion

Abschließend konnte der selbstbalancierende Roboter trotz großer Herausforderungen die gestellten Anforderungen erfüllen. Die Idee war, einen Roboter mit einem Gyrosensor zu bauen, der auf zwei Rädern fahren und Störungen tolerieren kann. Im Laufe der Zeit entstehen jedoch Fehler aufgrund des Drifts des Gyrosensors, wodurch der Roboter nur für eine begrenzte Zeit aufrecht bleiben kann, bevor der Winkel driftet. Um dieses Problem zu lösen, kann die Messung des Gyrosensors mit anderen Sensorinformationen wie beispielsweise Beschleunigungsdaten eines Accelerometers kombiniert und gefiltert werden, um eine genauere Messung des Winkels zu erzielen und den Drift zu minimieren. Mit Hilfe des geplotteten Graphen in MATLAB ermöglicht es, eine stetige Verbesserung der Regelparameter des Roboters. Die Übertragung des MATLAB-Codes auf den LEGO Mindstorms verlief reibungslos und dadurch konnte letztendlich das Hauptproblem, das durch Störungen aufgrund der Verbindung des Kabels verursacht wurde, behoben werden.

VI. Zusammenfassung und Fazit

Dieses Dokument beschreibt den Bau und die Steuerung eines selbstbalancierenden Roboters. Das Ziel war ein Roboter

zu entwickeln, der in aufrechter Position halten und Störungen von außen tolerieren kann. Anfangs wurde das Regelungssystem mit MATLAB programmiert, jedoch wurde das Programm auf die LEGO-Software übertragen, um auftretende Probleme zu lösen. Aufgrund der Drift des Gyrosensors wurde der Winkel durch Integration der Winkelgeschwindigkeit berechnet, um genauere Messungen zu erzielen.

Eine Erweiterung könnte das Hinzufügen von Greifarmen sein, um Gegenstände aufzuheben und zu manipulieren. Dafür könnten zusätzliche Sensoren ausstattet werden, dass der Roboter eine bessere Navigation und Hindernisvermeidung haben kann. Eine weitere Möglichkeit wäre die Integration eines dritten Motors mit einem Bein, sodass der Roboter sich aus eigner Kraft aufrichten kann.

ANHANG

[1] wlw inside business: https://www.wlw.de/de/inside-business/praxiswissen/logistikmanagement/roboterloesungen-fuer-die-logistik-produktiver-und-widerstandsfaehiger

[2] engineering.com: https://www.engineering.com/story/boston-dynamics-shows-off-logistics-version-of-handle-robot

[3]temperatur-profis:https://temperatur-profis.de/temperaturregler/reglertypen-pid/

[4] Autodesk Instructables:https://www.instructables.com/Self-Balancing-Robot-1/

[5] wikipedia:https://de.wikipedia.org/wiki/Regler

https://de.wikipedia.org/wiki/Regler

Regelung eines selbstbalancierenden Roboters

Saeid Miri, MTK
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Der zweirädrige Balanceroboter ist zu einem sehr beliebten Projekt in den Bereichen Mechatronik und Robotik geworden. Durch die Nutzung des umgekehrten Pendels gehört es zu den klassischen Systemen in Dynamik und Regelung. Allerdings stellt dieses System aufgrund seiner inhärenten Instabilität und Nichtlinearität eine der herausforderndsten Probleme in der Regelungstechnik dar. Die Stabilität des Roboters wird durch eine Kombination aus mechanischem Design, Rädern, Motoren, Sensoren und fortschrittlicher Software erreicht, die es ihm ermöglicht, sein Schwerpunkt in Echtzeit anzupassen. Das Projekt ist in zwei Hauptgruppen unterteilt: Hardware, Softwareimplementierung. Es wird eine detaillierte Betrachtung des Fortschritts jeder Phase erfolgen.**

***Schlagwörter*—Gyroskop-Sensor, zweirädriger Roboter, PID-Regler, Regelungstechnik, Self-balancing Robot, Segwayroboter**

I. Einleitung

IN In den letzten Jahren hat die Robotik weltweit bedeutende Fortschritte gemacht. Die Durchführung von internationalen Robotikwettbewerben in verschiedenen Bereichen sowie die Veröffentlichung wertvoller Arbeiten zur Robotik sind ein Beweis für diese Behauptung. Das Konzept des Balance-Roboters basiert auf dem umgekehrten Pendel [1]. Im Gegensatz zu einem gewöhnlichen Pendel neigt ein umgekehrtes Pendel dazu, instabil zu sein.

Während des Lego-Mindstorms-Projekts sollte ein zweirädriger selbstbalancierender Roboter gebaut werden, der sich mithilfe von Sensoren und Aktoren aufrecht hält und stabil auf seiner eigenen Achse bewegen werden kann. Im Vergleich zu Robotern mit drei oder vier Rädern benötigen sie weniger Komponenten und belegen weniger Platz. Aus diesem Grund haben zweirädrige Roboter einen größeren Freiheitsgrad und sind in kleinen Räumen äußerst nützlich. Eine gute Idee könnte die Verwendung dieser Roboter als Rollstuhl sein [2]. Außerdem sollte dieser Roboter in der Lage sein, einer Linie zu folgen und auf ihr hin- und zurückzufahren. Die Linienverfolgung konnte aufgrund von Zeitbeschränkungen leider nicht implementiert werden.

Da im Bereich der Mechatronik viel mit Steuerung und Regelung gearbeitet wird, ist dieses Projekt besonders relevant und lehrreich für uns.

II. Vorbetrachtungen

Roboter mit zwei Rädern weisen aufgrund ihrer Physik einige Vorteile auf. Auf Basis von ihrem geringeren Gewicht und ihrer wenigeren Komponenten haben diese Roboter einen niedrigeren Energieverbrauch und können zu einer guten Option im Transportsystem werden. Segway ist ein praktisches und kommerzielles Beispiel für diesen Roboter, welche die Neigungswinkel durch den Sensor ständig kontrolliert würde.

Es wurde von uns ein Roboter gebaut, der auf denselben Prinzipien basiert und durch den Gyro-Sensor wird den Winkel ständig gemessen. Um das Verständnis des selbstbalancierenden Roboters zu verbessern, ist es erforderlich, zunächst das Konzept des Selbstausgleichsmodus, des Gyrosensors und des Regelkreises zu verstehen.

A. Selbstausgleichsmodus

Der Selbstausgleichsmodus ist das beeindruckendste Merkmal von zweirädriger Robotern. Um zu verstehen, wie dieses System funktioniert, kann man feststellen, dass es eine ähnliche Funktion wie der menschliche Körper aufweist. Das Gleichgewicht geht verloren, wenn aus einer aufrechten Position nach vorne gelehnt wird, aber in der Regel wird man dabei nicht flach auf das Gesicht fallen. Nachdem das Gleichgewicht verloren wurde, beginnt das Gehirn wahrzunehmen und das Bein nach vorne zu bringen, um ein Fallen zu verhindern. Ein zweirädriger Roboter macht fast das gleiche, außer dass er anstelle von Beinen Räder hat, anstelle von Muskeln einen Motor und anstelle von Gehirn eine Sammlung von Mikroprozessoren und Sensoren hat. In Abbildung 1 ist zu erkennen, dass der Roboter aufgrund des Drehmoments nach vorne zu kippen droht. Dieses Drehmoment entsteht durch die Gewichtskraft $mg \cdot \cos(\theta)$. Um dem entgegenzuwirken und den Roboter aufrecht zu halten, werden die Motoren verwendet.

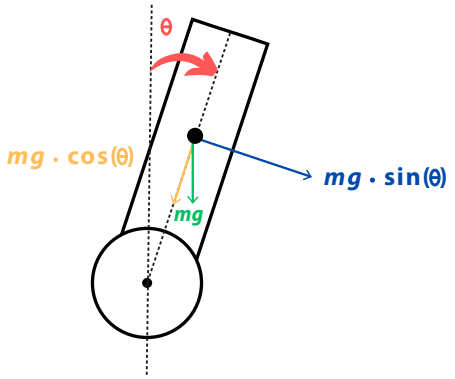


Abbildung 1. Self-balancing Robot model

B. Gyrosensor

Ein Gyroskop erfasst die Rotationsgeschwindigkeit und den Neigungswinkel um eine Achse. Die Drehung um eine Achse wird normalerweise in Grad pro Sekunde gemessen und eine Rotation in entgegengesetzter Richtung erzeugt ein Signal

DOI: 10.24352/UB.OVGU-2023-017

mit umgekehrtem Vorzeichen. Wenn die Rotation gestoppt wird, wird ein Wert von Null gemessen. Gyrosensoren können neben der Erfassung der Rotationsgeschwindigkeit auch die Bewegung des Körpers messen. Um Bewegungen genau zu messen, werden Gyrosensoren oft mit Beschleunigungssensoren kombiniert.

C. Regelkreis

Ein Regelkreis hat den Zweck, automatisch ein stabiles System zu generieren und sicherzustellen, dass das gewünschte Ausgangssignal oder die Referenz erreicht wird. Das System misst kontinuierlich den Ist-Zustand eines Prozesses und vergleicht ihn mit einem Soll-Zustand. Falls eine Abweichung erkannt wird, wird das System mittels einer Rückkopplung angepasst, um die Abweichung zu minimieren und den Prozess in Richtung des Soll-Zustands zu steuern.

III. Hardware

A. Aufbau

Der selbstbalancierende Roboter basiert hauptsächlich auf einem LEGO-Stein, welche über einen leistungsstarken ARM9-Prozessor verfügt [3], der schnelle Berechnungen und komplexe Programme ermöglicht. Zusätzlich verfügt er über einen Sensor, um die Position des Roboters zu messen. Zudem sind zwei Elektromotoren verbaut, die mit einer Geschwindigkeit von $160\,\mathrm{U/min}$ bis $170\,\mathrm{U/min}$ und einem Drehmoment von $20\,\mathrm{N\,cm}$ laufen [3]. Beim Bau des Roboters müssen viele Faktoren berücksichtigt werden, um eine stabile und gut steuerbare Konstruktion zu gewährleisten. Eine wichtige Überlegung ist eine gleichmäßige Gewichtsverteilung und der Schwerpunkt muss in der Mitte und nahe am Boden liegen. Im ersten Prototyp wurde der EV3-Stein vertikal und weit oben positioniert, was zu einer instabilen Konstruktion und erhöhte das Risiko des Umkippens. Bei dem zweiten Prototyp wurde der Stein im unteren Bereich des Roboters angebracht, aber aufgrund des vertikalen Aufbaus des Steins war das Gewicht ungleichmäßig verteilt. Letztendlich wurde, wie in Abbildung 2 zu sehen ist, der EV3-Stein horizontal auf der Radachse positioniert, um den Schwerpunkt nah am Boden zu halten und das Gewicht gleichmäßig zu verteilen. Zusätzlich sollte bei der Auswahl der Räder darauf geachtet werden, dass größere Räder verwendet werden, um aus dem Hebelgesetz (1) [4] ein höheres Drehmoment zu erzeugen. Diese Konstruktion sorgt für eine höhere Stabilität und eine bessere Steuerung des Roboters.

$$M = F \cdot r \tag{1}$$

B. Auswahl des Sensors

Für die Bestimmung der Position des Roboters wurde ursprünglich ein Lichtsensor verwendet, der aufgrund seiner Empfindlichkeit gegenüber Veränderungen in der Umgebungshelligkeit sowie Schattenwürfen durch den Roboter ungenauere Werte lieferte. Daher wurde anstelle des Lichtsensors ein Gyrosensor auf die Radachse gebaut, um den Winkel und die Drehgeschwindigkeiten des Roboters präzise zu messen und zu steuern. Obwohl der Gyrosensor im Laufe der Zeit aufgrund des Drifts ungenauere Werte liefert, erwies er sich als besser geeignet als der Lichtsensor.

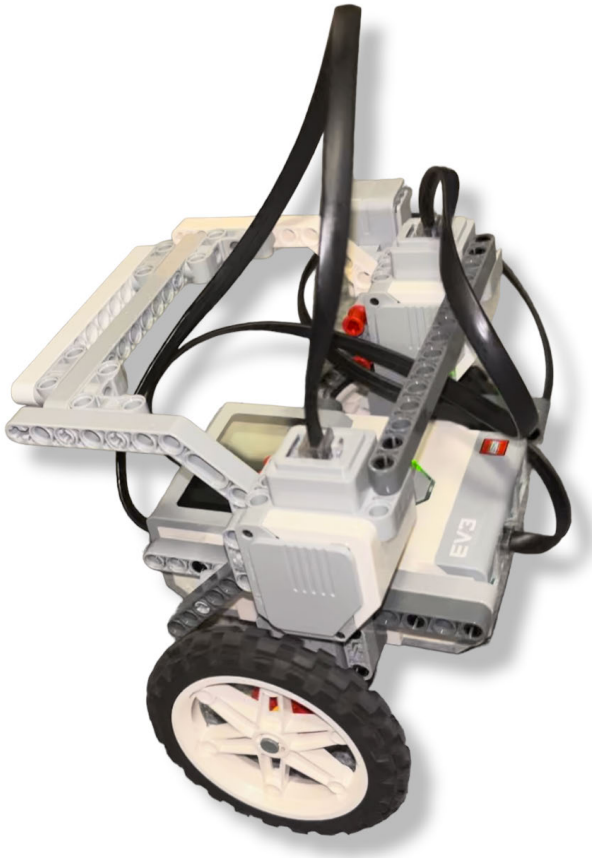


Abbildung 2. Robot Aussehen

IV. Software

A. Programmierung und Funktionsweise eines PID-Reglers

Ein PID[1]-Regler ist ein kombinierter Regler, Der drei Komponenten basiert. Um den PID-Regler für einen selbstbalancierenden Roboter zu programmieren, müssen zunächst die richtigen Parameter für die drei Komponenten gefunden werden. Die integrale Komponente berücksichtigt die zeitliche Komponente des Fehlers, die derivativen Komponente berücksichtigt die Geschwindigkeit des Fehlers und die proportionale Komponente gibt die Größe der Reaktion des Reglers auf den Fehler an. Die Struktur des PID-Reglers, welche in Abbildung 3 dargestellt ist, zeigt, dass die Integral, Proportional und Differential-Steuerungen parallel arbeiten und am Ende ihre Signale addiert werden, um das Steuersignal zu erzeugen. Es ist selten erforderlich, eine integrale Größe zu verwenden. Diese Größe kann in der Regel vernachlässigt werden. Nur wenn die Beschleunigung ein wichtiger Faktor für den Roboter ist, wird diese Größe benötigt. Aus diesem Grund wird für den Balance-Roboter ein PD-Regler gegenüber einem PID-Regler bevorzugt. Um den PD-Regler auf einem EV3-Stein zu implementieren, wird es folgendermaßen vorgegangen:

$$X \rightarrow \mathrm{ProportionalFehler}$$

$$Z \rightarrow \mathrm{Derivativfehler}$$

[1]Proportional, Integral, derivative

$$X = \text{gewünschter Winkel} - \text{aktueller Winkel} \quad (2)$$

$$Z = \text{ProportionalFehler} - \text{vorherige ProportionalFehler} \quad (3)$$

$$\text{Motor Leistung} = (K_\text{P} \cdot X) + (K_\text{D} \cdot Z) \quad (4)$$

Um das Verständnis der Programmierung zu verbessern, würde ein Programmablaufplan erstellt. Wie in Abbildung 4 deutlich zu sehen ist, wird der Winkel des Roboters auf 0 Grad gesetzt, um sicherzustellen, dass er in der Aufrechten Position bleibt. Zunächst wird der Winkel des Roboters durch den Sensor gemessen. Anhand der gemessenen Sensorwerte werden im nächsten Schritt die PD-Werte berechnet. Der Proportionalanteil des PD-Reglers wird durch den Unterschied zwischen dem gewünschten Winkel und dem aktuellen Winkel bestimmt, während der Differentialanteil auf der Rate des Winkeländerung basiert. Die Kombination der beiden Anteile ergibt die Steuergröße, die zur Ansteuerung des Motors verwendet wird, um den Roboter vorwärts und rückwärts zu bewegen. Dieser Ablauf wird kontinuierlich wiederholt, um den Roboter stabil und balanciert zu halten.

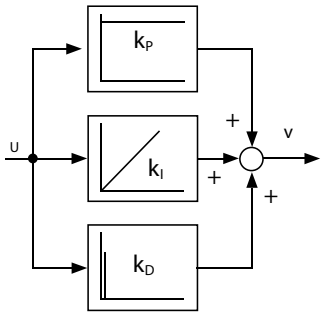


Abbildung 3. Blockdiagramm einen PID-Regler [5]

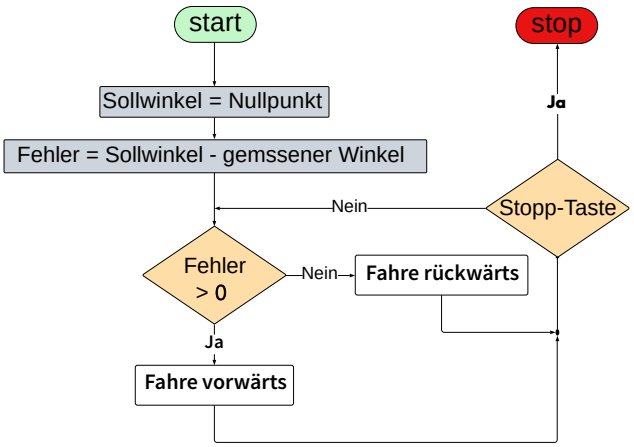


Abbildung 4. Programmablaufplan

B. Programmauswahl

Zu Anfang wurde MATLAB als Programmiersprache eingesetzt. Allerdings kam es aufgrund eines Kabels, das zwischen dem EV3-Stein und dem PC verbunden war, zu einer Störung im System, da das Kabel ständig den Roboter zog. Aufgrund dieser Störung war es schwierig, die exakten Parameterwerte für den PID-Regler auszuwählen. Nach Recherchen stellte sich heraus, dass das MATLAB-Skript nicht auf den Lego-Stein hochgeladen werden konnte. Darüber hinaus wurde als Ausgangswert des MATLAB-Skripts eine Geschwindigkeit erzeugt, obwohl für unsere Zwecke die Motorleistung benötigt wurde. Aus diesem Grund war die LEGO-Mindstorms Software die einzige Lösung, um das Programm reibungslos ausführen zu können, da die Daten direkt auf den LEGO-Stein hochgeladen werden konnten.

C. Parameterauswahl

Die Programmierung des PD-Reglers stellte eine Herausforderung dar, da es schwierig war, die optimalen Parameter für die zwei Komponenten zu bestimmen. Um den gewünschten Stabilitätsgrad zu erreichen, wurden mehrere Tests durchgeführt und die Parameter kontinuierlich angepasst. Wie aus Abbildung 5 ersichtlich wird, erhöhte sich im Laufe der Zeit die Fehleramplitude für die ausgewählten Werte und führte zu Schwingungen. Nach einigen Anpassungen konnten schließlich die korrekten Parameter gefunden werden, wodurch es möglich wurde, den Roboter erfolgreich selbstbalancierend zu mache, wie in Abbildung 6 dargestellt. Bei jeder Motorreaktion wurde die Error auf Null reduziert.

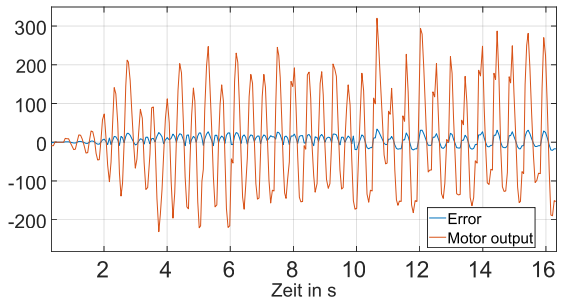


Abbildung 5. Graphische Darstellung des Fehlers und Motorleistung mit $K_\text{P} = 9$ und $K_\text{d} = 0,1$

.

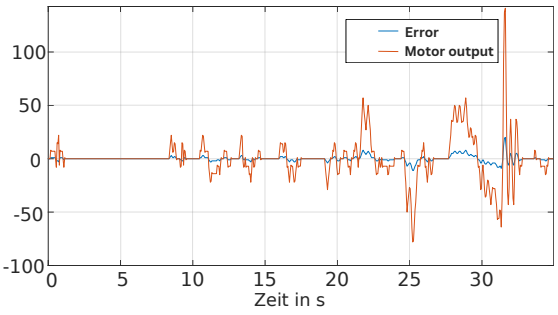


Abbildung 6. Graphische Darstellung des Fehlers und Motorleistung mit $K_\mathrm{P} = 7$ und $K_\mathrm{d} = 0,2$

V. Ergebnisdiskussion

Letztendlich wurde ein selbstbalancierender Roboter gebaut, der in der Lage war, sich aufrecht zu halten. Es war jedoch erforderlich, vor jedem Start zu überprüfen, ob der gewünschte Winkel auf Null lag, da der Lego-Gyrosensor zu Drift neigte.

VI. Zusammenfassung und Fazit

Angesichts des Fortschritts der Technologie und des Einsatzes moderner Fahrzeuge ist es offensichtlich, dass die Welt in Richtung Fahrzeuge mit höheren Fähigkeiten und geringerem Verbrauch geht. Aus diesem Grund wurde eines der am häufigsten verwendeten und kostengünstigsten Fahrzeuge in kleinen Größen in dieser Studie untersucht, nämlich der Segway.

In diesem Artikel wurde Gyrossensor verwendet, um den Neigungswinkel des Roboters zu bestimmen. Als Regelsystem wurde zusätzlich ein PD-Regler eingesetzt, welcher in der Industrie weit verbreitet ist. Der PD-Regler gilt als bewährtes und zuverlässiges System zur Regelung und zeichnet sich durch seine einfache Struktur und hohe Effektivität aus. Eine Möglichkeit zur Entwicklung eines intelligenteren Balancing-Roboters in der Zukunft wäre die Kombination von Beschleunigungssensor und Gyrosensor zur Messung von Winkeln mit höherer Präzision. Eine andere Option wäre die Integration eines zweiten PID-Reglers, um einen zweirädrigen Selfbalancing-Roboter in der Lage zu machen, sich möglicherweise in alle vier Richtungen (vorwärts, rückwärts, links und rechts) bewegen könnte. Des Weiteren könnte die Verwendung eines Lichtsensors in Kombination mit einem PID-Regler sein, eine Linienverfolgung zu ermöglichen.

Literaturverzeichnis

[1] Chan, Ronald Ping M. ; Stol, Karl A. ; Halkyard, C R.: Review of modelling and control of two-wheeled robots. In: *Annual reviews in control* 37 (2013), Nr. 1, S. 89–103

[2] Goher, KM ; Tokhi, MO ; Siddique, NH: Dynamic modeling and control of a two wheeled robotic vehicle with a virtual payload. In: *ARPN Journal of Engineering and Applied Sciences* 6 (2011), Nr. 3, S. 7–41

[3] Team, Lego: *Lego Mindstorm*, https://education.lego.com/en-us/lessons/ev3-space/turn-using-sensor

[4] Wikipedia: *Hebelgesetz*, https://de.wikipedia.org/wiki/Hebel_(Physik)

[5] Heinrich, Berthold ; Linke, Petra ; Glöckler, Michael ; Heinrich, Berthold ; Linke, Petra ; Glöckler, Michael: Regelungstechnik. In: *Grundlagen Automatisierung: Erfassen-Steuern-Regeln* (2020), S. 123–229

Automatischer Seifenspender

Abrar Ahmed, Elektrotechnick und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Ziel des Projektseminars Elektrotechnik/Informationstechnik (LEGO-Mindstorms) 2023 an der Otto-von-Guericke-Universität Magdeburg war die Entwicklung eines Roboters. Basierend auf der Programmierung und Konstruktion mit dem zur Verfügung gestellten LEGO-NXT Set und MATLAB sollte der Roboter bestimmte Funktionen ausführen. Ein automatisierter Seifenspender war die Wahl für dieses Projekt und der Prozess, die damit verbundenen Herausforderungen und die Ergebnisse werden im Folgenden beschrieben. Um das Projekt abzuschließen, wurde der entwickelte automatische Seifenspender in Form einer Präsentation vorgestellt und seine Funktionalität demonstriert.**

***Schlagwörter*—LEGO-Mindstorms, MATLAB, Programmierung, Roboter, Automatisierung, Seifenspender, Ultraschallsensor**

I. Einleitung

In der heutigen Welt nach der Pandemie ist die Verbreitung von Keimen ein großes Problem im Alltag der Menschen. Die Verbesserung der Hygiene und die Minimierung aller möglichen unnötigen Kontakte, die wiederum die Verbreitung von Keimen und Bakterien verhindern würden, sollte eine Priorität sein. Einer der wichtigsten Faktoren ist der Kontakt mit Alltagsgegenständen, die von vielen Menschen gemeinsam benutzt werden. Ironischerweise dient gerade ein solcher gemeinsamer Gegenstand der Verbesserung der Hygiene: der Seifenspender. Ob zu Hause oder in öffentlichen Toiletten, es kommt immer wieder vor, dass Menschen sich einen manuellen Seifenspender teilen, bei dem eine Person auf den Spender drücken muss, um die Seife herauszubekommen. Dieser unnötige Körperkontakt mit einem Gegenstand in einem meist unsauberen Bereich ist genau die Art von Körperkontakt, die durch Automatisierung vermieden werden kann.

Genau aus diesem Problem heraus entstand die Idee des automatischen Seifenspenders. Mit dem Endprodukt dieses Projekts wird der gesamte manuelle physische Kontaktprozess zugunsten eines berührungslosen, hygienischeren Seifenspendeprozesses vermieden. In diesem Fall werden Sensoren zur Erkennung der Anwesenheit einer Hand unter dem Seifenspender verwendet, was wiederum die Aktivierung des Seifenspenders und die kontaktlose Abgabe der Seife ermöglicht. Hier werden Ultraschallsensoren zur Abstandsmessung verwendet und unter bestimmten Bedingungen funktioniert das Gerät wie gewünscht. Diese Bedingungen werden mit MATLAB programmiert und an den LEGO-Mindstorm Block übertragen, der das Gerät steuert.

II. Vorbetrachtungen

Das Projekt beschäftigt sich mit der Automatisierung eines konventionellen Seifenspenders und den dafür notwendigen Schritten. Es wird ein kurzer Überblick über die verwendeten mechanischen LEGO-Komponenten und Sensoren gegeben.

DOI: 10.24352/UB.OVGU-2023-018

A. konventioneller Seifenspender

Vor der Konstruktion eines automatischen Seifenspenders muss die Funktionalität eines manuellen Seifenspenders untersucht werden. Es ist wichtig zu sehen, wie dieser verbessert werden kann, um die gewünschten Ziele der kontaktlosen Seifenabgabe zu erreichen. Ein normaler Seifenspender hat einen Behälter, der mit Seife gefüllt ist, die dann durch ein Rohr nach oben und aus dem Kopf des Spenders heraus befördert wird, wenn der Kopf physisch gedrückt wird. In diesem Fall ist das Ziel das Erkennen der Anwesenheit einer Hand und das Drücken des Kopfes des Spenders durch programmierte mechanische LEGO-Steine, anstatt ihn mit der Hand drücken zu müssen. Auf diese Weise kann ein direkter Kontakt vermieden werden.

B. Funktionsprinzip des Ultraschallsensors

Zur Messung des Abstandes zwischen Zapfsäulenkopf und Hand wird ein Ultraschallsensor verwendet. Das Funktionsprinzip eines Ultraschallsensors beruht auf dem Echo-Prinzip. Dabei wird ein Ultraschallsignal ausgesendet, das von einem Objekt reflektiert wird. Die Zeit, die das Signal benötigt, um zum Sensor zurückzukehren, wird gemessen und kann zur Berechnung der Entfernung des Objekts verwendet werden. Die kurzen Ultraschallsignale, die in diesem Fall ausgesendet wurden, wurden in sehr kurzen Abständen gesendet, um die An- oder Abwesenheit der Hand regelmäßig zu erfassen. Dies wird in einem späteren Abschnitt näher erläutert.

Abbildung 1: LEGO-Ultraschallsensor [1]

C. Schneckengetriebe

Der oben erwähnte konventionelle Seifenspender erforderte einen erheblichen Kraftaufwand. Diese Kraft ist normalerweise von den Motoren, die im LEGO-Set enthalten sind, nur schwer aufzubringen, um den Kopf des Seifenspenders erfolgreich zu drücken. Deshalb wurde das LEGO-Schneckengetriebe wie in Abbildung 2 verwendet. Ein Schneckengetriebe ist ein Getriebe, das aus einer schraubenförmigen Welle (der Schnecke) und einem zylindrischen Rad (dem Schneckenrad)

besteht. Die Schnecke wird von einer Antriebswelle angetrieben, die das Schneckenrad dreht [2]. Das Schneckengetriebe wird im Modellbau verwendet, weil es eine gute Belastbarkeit, eine präzise Positionierung und eine hohe Kraftübertragung ermöglicht. Außerdem kann durch das Schneckengetriebe eine starke Drehbewegung auf kleinstem Raum übertragen werden, was bei dieser kleinen Konstruktion wichtig ist.

Abbildung 2: LEGO-Schneckengetriebe [3]

III. Konstruktion und Programmierung

Im Folgenden wird detailliert auf den Aufbau der Mechanik und die Programmierung der entsprechenden Prozesse eingegangen.

A. Aufbau

Der Aufbau des Seifenspenders lässt sich im Wesentlichen in 3 Teile gliedern. Das Unterteil hält den manuellen Seifenspender so fest, dass er sich bei Krafteinwirkung nicht bewegt. Das Unterteil ist so konstruiert, dass es im Bedarfsfall zum Auswechseln eines leeren Behälters leicht geöffnet werden kann. Während des gesamten Vorgangs wird eine beträchtliche Kraft auf den Seifenspenderkopf ausgeübt. Es ist daher sehr wahrscheinlich, dass sich der Seifenspenderkopf bewegt oder seitlich wegrutscht.

Abbildung 3: Aufbau des Seifenspenders

Der Kopf des Seifenspenders wird mit Gummibändern fixiert, damit er während des Spendevorgangs nicht wackelt. Die Motoren, das Schneckengetriebe und alle Zahnräder werden im mittleren Teil der Konstruktion gehalten. Der wichtigste Teil der Konstruktion ist genau dieser Teil. Er erfährt eine große nach oben gerichtete Reaktionskraft, wenn der LEGO-Stein auf den Seifenspenderkopf drückt. Es muss daher unbedingt darauf geachtet werden, dass dieser Teil der Konstruktion gut befestigt ist und sich während des Vorgangs nicht nach oben bewegt.

Für diese spezielle Konstruktion wurden zwei separate Motoren verwendet: Einer ist der Hauptmotor, der die Zahnräder dreht, und der andere ist als Ersatzmotor vorhanden. Wenn der erste Motor aus irgendeinem Grund ausfällt, können die Kabel ganz einfach vom ersten auf den zweiten Motor umgesteckt werden und die Maschine arbeitet normal weiter. Außerdem bringt der zweite Motor mehr Symmetrie und Stabilität in die Gesamtstruktur, da das Gewicht gleichmäßig auf beide Seiten verteilt ist. Die Zahnräder sind mit einem LEGO-Stein verbunden, der sich je nach Drehrichtung der Motoren auf und ab bewegt und den Seifenspenderkopf entsprechend drückt und loslässt. Im oberen Teil der Konstruktion befindet sich der Ultraschallsensor, der nach unten auf den Boden gerichtet ist. Schließlich sind alle diese Motoren und Sensoren mit dem zentralen LEGO-NXT-Block auf der Rückseite des Geräts verbunden, der die Codes zur Steuerung aller Funktionen des Geräts enthält. Der LEGO-NXT-Block wird an den Computer angeschlossen, wo der MATLAB-Code eingelesen wird.

B. Ablaufplan

Die Idee des automatischen Seifenspenders lässt sich kurz wie folgt erklären. Der Ultraschallsensor erkennt das Vorhandensein einer Hand unter dem Seifenspenderkopf. Wenn eine Hand erkannt wird, wird Seife für eine festgelegte Zeit ausgegeben und dann gestoppt. Es wird keine weitere Seife ausgegeben, bis die Hand unter dem Sensor weggezogen wird und der Vorgang erneut gestartet werden kann. Damit diese Idee funktioniert, sind im Grunde zwei Schleifen erforderlich, deren detaillierter Ablauf inklusive aller einzelnen Schleifen im Folgenden erläutert wird.

C. Algorithmus

Der Ultraschallsensor misst den Abstand zwischen sich und dem, was sich unter ihm befindet. Ist der gemessene Abstand größer als 15 cm, wird keine Hand vom Sensor erkannt und somit auch keine Seife ausgegeben. Ist der Abstand jedoch kleiner als 15 cm, interpretiert das Programm dies so, dass sich in diesem Moment eine Hand unter dem Seifenspenderkopf befindet. Dies führt dazu, dass sich der Motor im Uhrzeigersinn mit einer maximalen Leistung von 100 für eine eingestellte Anzahl von Umdrehungen dreht. Das „Tacho Limit“ bestimmt die Anzahl der Umdrehungen des Motors, die in diesem Fall auf 4500 Grad oder 12,5 Umdrehungen eingestellt ist. Dadurch drehen sich die Zahnräder und bewegen den LEGO-Stein über dem Kopf des Seifenspenders nach unten, wodurch die Seife ausgegeben wird. Danach ist der Motor so programmiert, dass er sich automatisch um die gleiche Anzahl von Umdrehungen

gegen den Uhrzeigersinn dreht und wieder in seine Anfangsposition zurückkehrt. Der Kopf des Seifenspenders wird während dieses Teils ebenfalls freigegeben und das Gerät ist bereit, erneut gedrückt zu werden, wenn bestimmte Bedingungen wieder erfüllt sind.

Der Sensor fährt dann mit der Abstandsmessung fort. Das Gerät ist so programmiert, dass der gesamte Vorgang erst dann wieder von vorne beginnt, wenn der Sensor erkennt, dass sich die Hand nicht unter das Gerät bewegt hat, d. h. ein Abstand von mehr als 15 cm gemessen wird. Ist diese Bedingung erfüllt, kehrt das Programm in den Ausgangszustand zurück, und der Vorgang kann von neuem beginnen.

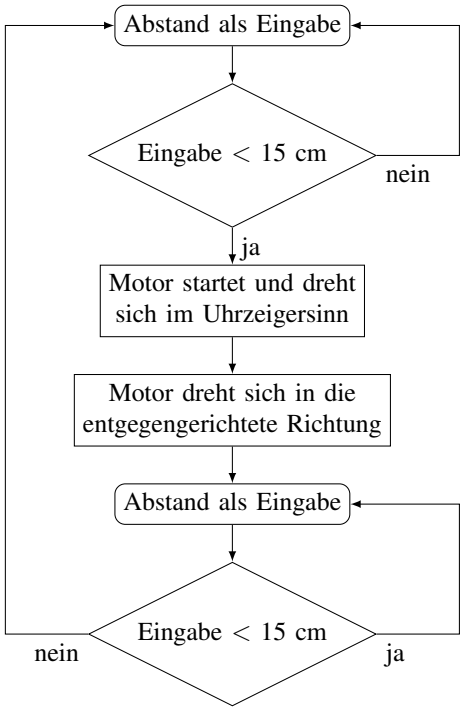


Abbildung 4: Ablaufplan des Algorithmus

D. Zufälliger Eingabefehler

Beim Programmieren und Testen des Gerätes wurde festgestellt, dass es immer wieder zu zufälligen Eingabefehlern kam, die durch den Ultraschallsensor verursacht wurden. Die Idee des Spenders war, dass nach der Seifenspende die Hand unter dem Sensor wegbewegt werden muss, bevor der gesamte Vorgang, wie oben beschrieben, von vorne beginnen kann. Bei den Tests wurde jedoch festgestellt, dass der Seifenspender auch dann plötzlich mit der Abgabe der Seife begann, wenn die Hand nach der Abgabe der Seife nicht weggezogen wurde und der vom Sensor gemessene Abstand weniger als 15 cm betrug. Dies ist darauf zurückzuführen, dass es gelegentlich zu einer zufälligen Messung kam, bei der der vom Ultraschallsensor gemessene Abstand einen Wert von mehr als 15 cm ergab. Ein solches Beispiel ist in Abbildung 5 dargestellt.

```
Abstand: 6 cm
Warten auf Wegnehmen
Abstand: 6 cm
Warten auf Wegnehmen
Abstand: 11 cm
Warten auf Wegnehmen
Abstand: 22 cm
Hand ist wieder weg
Warten auf Wegnehmen
Abstand: 7 cm
Warten auf Wegnehmen
Abstand: 9 cm
```

Abbildung 5: Beispiel - Zufälliger Eingabefehler

Um sicherzustellen, dass dieser zufällige Fehler die ursprüngliche Idee des Gerätes nicht stört, wurde eine sekundäre Schleife in den Code programmiert, wie in Abbildung 6 zu sehen ist. Bleibt die Hand nach der Abgabe unter dem Sensor, beginnt ein neuer Zähler mit einer Variablen n. Für jede aufeinanderfolgende Messung mit einem Wert größer als 15 cm wird der Wert von n um eins erhöht. Wenn es drei aufeinanderfolgende Messungen mit einem gemessenen Abstand von mehr als 15 cm gibt, d. h. $n = 3$, bedeutet dies, dass die Hand tatsächlich unter dem Sensor verschwunden ist und der gesamte Vorgang von vorne beginnen kann. Ist der Wert von n kleiner als 3, so bedeutet dies, dass ein zufälliger Eingabefehler des Sensors aufgetreten ist und diese Messwerte ignoriert werden können, da die Hand nicht tatsächlich unter dem Sensor verschwunden ist.

```
n=0;
        while 1
            d=GetUltrasonic(SENSOR_2);
            if d>limit
                disp('Hand ist wieder weg')
                n=n+1;
                if n==3
                    break;
                end
            else
                n=0;
            end
```

Abbildung 6: Problemlösungsschleife

IV. Ergebnisdiskussion

Das Endergebnis dieses Projekts, wie es in der Präsentation und in dem obigen Dokument beschrieben ist, ist ein automatischer Seifenspender zur Abgabe von Seife bei Anwesenheit einer Hand. Dank der oben erwähnten Ergänzungen und Korrekturen des Codes gibt der Spender bei Anwesenheit der Hand korrekt Seife aus und wartet dann, bis die Hand entfernt wird, bevor er den gesamten Vorgang von neuem beginnt. Der Ultraschallsensor liefert dem Programm genaue und zuverlässige Messwerte. Dies ist möglich, weil Lösungen

für zufällige Eingabefehler zur Verfügung stehen. Die Konstruktion kann auch für andere Zwecke modifiziert werden, um beispielsweise Wurstsoße abzugeben, und bleibt nicht auf Seife beschränkt.

V. Zusammenfassung und Fazit

Ziel dieses Projekts war es, einen konventionellen Seifenspender mit Hilfe von MATLAB-Programmierung und dem LEGO-NXT-Set in einen automatischen Seifenspender umzuwandeln. Insgesamt wurde die Grundidee gut umgesetzt und die gewünschten Ergebnisse erzielt. Darüber hinaus wurden einige Verbesserungen und Erweiterungen der Grundkonstruktion des Seifenspenders angedacht, die jedoch in der zur Verfügung stehenden Zeit nicht umgesetzt werden konnten. Zu den Verbesserungen gehören unter anderem die Messung der verbleibenden Seifenmenge in der Flasche und die Benachrichtigung, wenn der Behälter nachgefüllt werden muss.

Anhang

```
while 1
    pause(0.2)
    d = GetUltrasonic(SENSOR_2);
    disp(['Abstand: ',num2str(d),' cm'])
    if d>limit
        % Warten auf das Hinhalten der Hand
        a.Power = 0;
        a.SendToNXT(handle);
    else
        % Hand wurde erkannt
        % Seife spenden
        disp('Seife spenden')
        a.Power = -100;
        a.TachoLimit=4500;
        a.SendToNXT(handle);
        a.WaitFor
        disp('fertig und zur ckfahren')
        % zurucksetzen
        a.Power = 100;
        a.TachoLimit=4500;
        a.SendToNXT(handle);
        a.WaitFor
        disp('Original Zustand')
```

Abbildung 7: Hauptschleife des Programms

Literaturverzeichnis

[1] Torok, Rob: *NXT Sensors*. http://www.legoengineering.com/nxt-sensors/. Version: May 2019

[2] Wikipedia, the free encyclopedia: *Schneckengetriebe*. https://de.wikipedia.org/wiki/Schneckengetriebe. Version: 2023

[3] *Schneckengetriebe*. https://www.steinpalast.eu/lego-technic/differentiale-getriebe?p=1

Automatischer Seifenspender

Joan Llaguri, Elektrotechnik und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—In der heutigen Welt schreitet die Technologie mit einem beispiellosen Tempo voran und hat das Ziel, das menschliche Leben zu verbessern und zu vereinfachen. Sogar einfache Aufgaben werden mittlerweile von elektrischen Maschinen übernommen. Ein gutes Beispiel dafür sind automatische Seifenspender, die heute in vielen öffentlichen Waschräumen, Krankenhäusern, Schulen, Büros und anderen öffentlichen Gebäuden zu finden sind. Obwohl automatische Seifenspender eine einfache Aufgabe erfüllen, tragen sie erheblich dazu bei, die Hygiene zu verbessern, die Verbreitung von Krankheiten zu reduzieren und eine gesündere und hygienischere Umwelt zu schaffen.**

Der Mechanismus eines automatischen Seifenspenders basiert hauptsächlich auf der Idee, dass ein Sensor die Anwesenheit von Händen erkennt und dann eine bestimmte Menge Seife durch einen Motor abgibt, ohne dass der Benutzer den Spender berühren muss. Im Rahmen des Projektseminars soll ein automatischer Seifenspender mit Hilfe von LEGO-Mindstorm gebaut und mit MATLAB programmiert werden.

***Schlagwörter*—Seifenspender, Ultraschallsensor, Hygiene, LEGO-Mindstorm, MATLAB, Projektseminar.**

I. Einleitung

IN der heutigen Zeit ist es wichtiger denn je, auf Hygiene und Sauberkeit zu achten. Besonders während der Corona-Pandemie wird deutlich, wie wichtig es ist, die Ausbreitung von Bakterien und Viren zu reduzieren. Regelmäßiges Händewaschen ist eine einfache und sehr wirksame Maßnahme, um die Verbreitung von Infektionskrankheiten zu verringern. Daher sollte jeder die Reinigung der Hände zur Gewohnheit machen und so dazu beitragen, die Ausbreitung von Infektionskrankheiten einzudämmen. Automatische Seifenspender sind eine praktische Lösung, um diesen Prozess sowohl hygienischer als auch effizienter zu gestalten. Anstelle der manuellen Betätigung des Seifenspenders ist eine automatische Dosierung der Seife durch Erfassung der Handbewegung des Benutzers über einen Sensor möglich. Eine automatische Seifenspenderlösung bietet eine Vielzahl von Vorteilen in Bezug auf Zeitersparnis und Kosteneffizienz. Durch die automatische Dosierung wird eine bestimmte Menge an Seife ausgegeben, was eine Überdosierung verhindert und somit die Kosten für den Nachschub senkt.

Im Zuge des LEGO-Mindstorms-Projekts wurde eine automatische Lösung für Seifenspender gebaut. Zur Automatisierung der Seifendosierung wurde ein Ultraschallsensor integriert, der in der Lage ist, die Handbewegungen des Benutzers zu erkennen.

II. Vorbetrachtungen

Es ist wichtig, beim Pumpen der Seife ein ausreichendes Drehmoment zu erzeugen, da der Motor allein nicht in der Lage ist, genug Kraft zum Auspressen der Seife aufzubringen. Daher wird oft ein Getriebe eingesetzt, um das benötigte Drehmoment zu erzeugen. Um die Anwesenheit von Händen zu erkennen, kann ein Ultraschallsensor verwendet werden.

A. Ultraschallsensor

Ein Ultraschallsensor ist ein Gerät, das Schallwellen nutzt, um Entfernungen zu messen und Objekte zu erkennen. Wie in Abbildung 1 zu sehen ist, sendet es Ultraschallwellen aus und erfasst dann die reflektierten Wellen, um die Entfernung zum Objekt zu bestimmen. Diese Technologie ist ähnlich wie die von Radarsensoren, aber Ultraschallsensoren verwenden Schallwellen anstelle von Radiowellen [1]. Ultraschallsensoren werden in zahlreichen Anwendungsbereichen eingesetzt, von der Messung von Entfernungen bis hin zur Erkennung von Objekten. Insbesondere in der Automobil- und Industriebranche kommen sie häufig zum Einsatz, beispielsweise um Hindernisse zu identifizieren oder Roboter zu steuern.

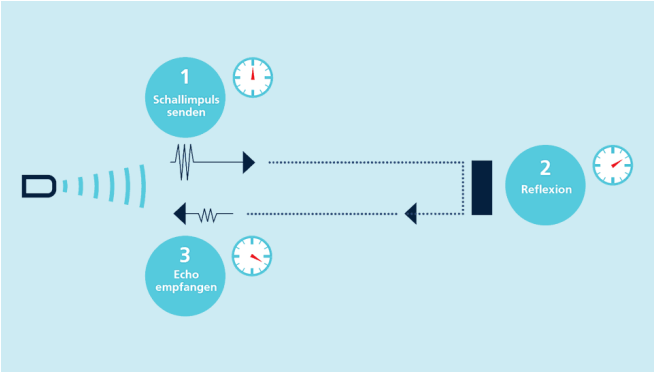


Abbildung 1. Funktionsweise und Technologie von Ultraschallsensoren [2].

B. Getriebe

Getriebe sind ein wesentlicher Bestandteil von mechanischen Systemen und werden in vielen Bereichen eingesetzt, um die Drehzahl, das Drehmoment und die Richtung der Kraftübertragung zu ändern. Der von dem NXT Motor gelieferte Antrieb erzeugt leider nur ein geringes Drehmoment, das für das Auspressen von Seife nicht ausreichend ist. Daher wurde entschieden, ein Schneckengetriebe zu verwenden. Ein Schneckengetriebe besteht aus einer Schneckenwelle und einem Schneckenrad, wie in Abbildung 2 dargestellt. Durch die spiralförmige Form der Schnecke und des Schneckenrads dreht sich das Schneckenrad langsamer als die Schneckenwelle, wodurch das Drehmoment verstärkt wird.

DOI: 10.24352/UB.OVGU-2023-019

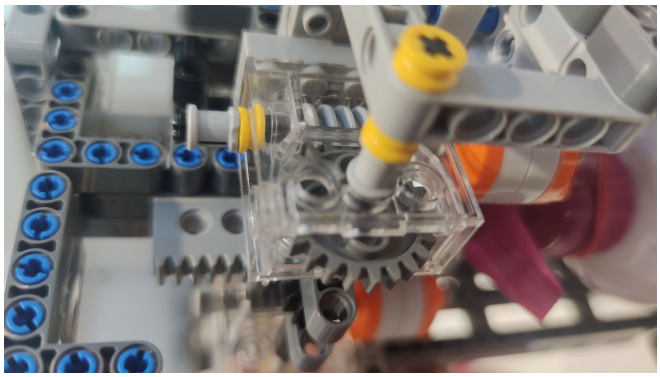


Abbildung 2. Schneckengetriebe

III. Realisierung

A. Aufbau

Das ganze Projekt bestand aus einem NXT-Block, 2 Motoren, einem Ultraschallsensor, der LEGO-Struktur und einem normalen Seifenspender. Die Konstruktion der Hauptstruktur war eine Fusion aus einem normalen Seifenspender und der LEGO-Struktur, die von uns gebaut wurde. Zu Beginn bestand das erste Problem darin, wie genügend Kraft für das Drücken des Knopfes erzeugt werden kann. Dies wurde mit einem Schneckengetriebe gelöst, das das Drehmoment stark erhöhte und das Drücken des Pumpenkopfes ermöglichte. Nach dem Bau der gesamten Struktur wurde ein weiteres Problem festgestellt. Es zeigte sich, dass der obere Teil der Struktur nach oben ging, wenn der LEGO-Stein den Pumpenkopf nach unten drückte. Der Grund für diese Reaktion war die vom Pumpenkopf ausgehende Reaktionskraft.

Die Lösung für dieses Problem bestand einfach darin, die Struktur zu fixieren und sie durch Hinzufügen weiterer LEGO-Steine fester zu machen. Es war nicht ganz einfach, den oberen Teil der Struktur zu befestigen, da das Schneckengetriebe ein alter LEGO-Stein war, während die anderen Teile aus neuen Steinen bestanden. Dies würde immer eine halbe Blockverschiebung mit sich bringen. Obwohl in der Abbildung 3 zwei Motoren zu sehen sind, wird nur einer für den gesamten Prozess verwendet. Der andere Motor wird aus verschiedenen Gründen verwendet, z. B. um die Konstruktion besser sichtbar zu machen, um das Gleichgewicht zu halten und um das System zu entlasten. Wenn etwas mit dem verwendeten Motor schief geht, ist der andere Motor bereit, effektiv zu laufen.

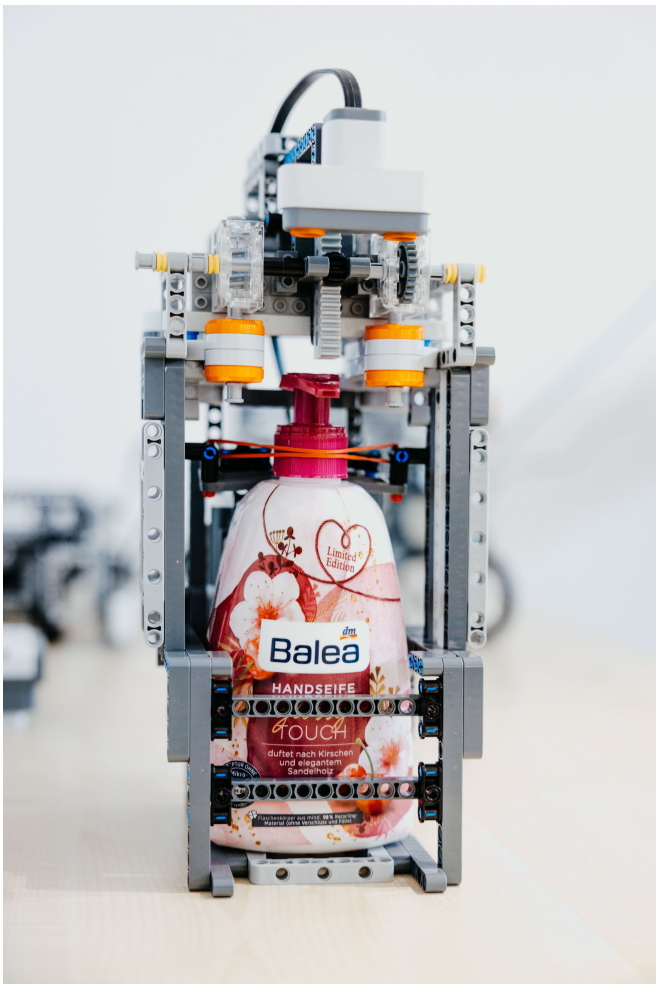


Abbildung 3. Finale Version des automatischen Seifenspenders

B. Programmierung

Der Code wurde entwickelt, um eine automatische Seifenspenderfunktion mit einer NXT-Plattform und einem Ultraschallsensor zu implementieren. Der Code beginnt mit dem Anschluss an das NXT-Gerät und der Initialisierung eines Motors und eines Ultraschallsensors. Die Variable „limit“ definiert die maximale Entfernung, in der ein Objekt, z. B. eine Hand, vom Sensor erkannt werden soll. Ist der Abstand größer als der Grenzwert, wartet das Programm, bis eine Hand erkannt wird, indem es den Motor mit einer Leistung von 0 anhält. Wird jedoch eine Hand erkannt, wird der Motor zur Ausgabe von Seife aktiviert.

Der Motor dreht sich 4500 Umdrehungen lang mit einer Leistung von -100, bevor er in seinen Ausgangszustand zurückkehrt. Das Programm wartet dann darauf, dass die Hand entfernt wird, bevor es erneut auf die Erkennung wartet. Wird die Hand bei mindestens drei aufeinanderfolgenden Messungen nicht erkannt, wird davon ausgegangen, dass sie entfernt wurde, und das Programm wartet darauf, dass die Hand erneut festgehalten wird.

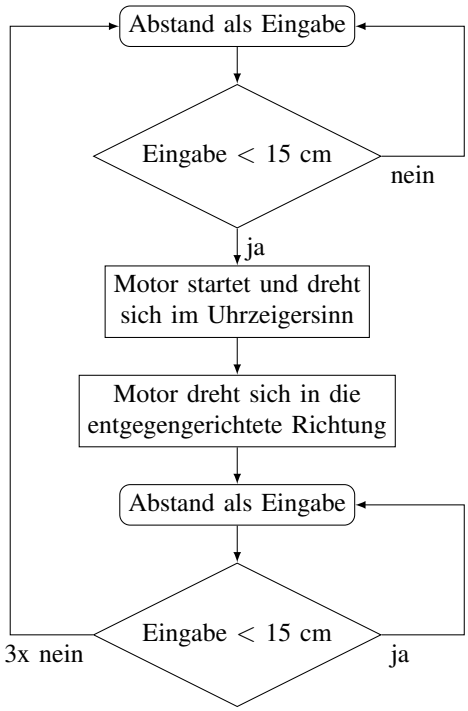


Abbildung 4. Programmablaufplan

1) Zufällige Variable: Das Ziel der Idee war die Implementierung einer Funktion, bei der der gesamte Prozess während der Platzierung der Hand unter dem Sensor und dem Warten auf die Entfernung der Hand ausgeführt werden sollte. Während der Entwicklung traten jedoch Schwierigkeiten auf: In der Abbildung 5 ist beispielsweise der Abstand einer Hand unter dem Ultraschallsensor zu sehen, während der Prozess ausgeführt wird. Der Ultraschallsensor verursachte gelegentlich zufällige Messfehler und startete den Prozess erneut, solange sich die Hand noch unter dem Sensor befand. Um dieses Problem zu lösen, wurde ein Abschnitt des Codes hinzugefügt, wie in Abbildung 6 dargestellt. Er sicherstellt, dass der Sensor mindestens drei aufeinander folgende Messungen des Objekts durchführt, bevor das Programm davon ausgeht, dass es sich um eine echte Erkennung handelt. Sobald die Hand entfernt wird, wartet das Programm, bis es mindestens drei aufeinanderfolgende Messungen durchgeführt hat, bevor es erneut wartet, um zu sehen, ob es „erkannt“ wurde. Dieser Zusatz verbessert die Zuverlässigkeit des Systems und verringert die Wahrscheinlichkeit, dass der Sensor ungenaue Messwerte liefert und den Prozess stört.

IV. Ergebnisdiskussion

Entwicklung eines automatischen Seifenspenders mit integriertem Ultraschallsensor wurde die gestellte Aufgabe erfolgreich umgesetzt. Benutzer können nun Seife ohne physischen Kontakt mit dem Spender dosieren, was besonders in Umgebungen mit vielen Benutzern von Vorteil ist. Ultraschallsensoren haben sich im Rahmen des Projekts als effektives und zuverlässiges Mittel zur Verbesserung der Funktionalität von automatischen Seifenspendern erwiesen. Es gab jedoch einige Herausforderungen bei der Implementierung, wie z. B. die

```
Abstand: 6 cm
Warten auf Wegnehmen
Abstand: 6 cm
Warten auf Wegnehmen
Abstand: 11 cm
Warten auf Wegnehmen
Abstand: 22 cm
Hand ist wieder weg
Warten auf Wegnehmen
Abstand: 7 cm
Warten auf Wegnehmen
Abstand: 9 cm
```

Abbildung 5. Zufällige Fehler

```
n=0;
        while 1
            d=GetUltrasonic(SENSOR_2);
            if d > limit
                disp('Hand ist wieder weg')
                n=n+1;
                if n==3
                    break;
                end
            else
                n=0;
            end
```

Abbildung 6. Codeabschnitt der Lösung für die zufällige Fehler

Feinabstimmung des Sensors und die korrekte Programmierung des Systems.

V. Zusammenfassung und Fazit

Das Ziel des Projekts bestand darin, einen automatischen Seifenspender zu entwickeln, der bei Erkennung einer Hand eine bestimmte Menge Seife abgibt. Im Vergleich zu herkömmlichen Spendern kann die Verwendung eines automatischen Spenders das Risiko einer Keimübertragung zwischen Personen reduzieren, da der Benutzer den Spender nicht berühren muss. Ein solcher Spender kann auch für andere Substanzen wie Saucen verwendet werden, aber es sind verschiedene Faktoren zu berücksichtigen, wie beispielsweise die Konsistenz der Sauce, die Größe der Öffnung und die Leistung des Motors.

Literaturverzeichnis

[1] Baumer: *Funktionsweise und Technologie von Ultraschallsensoren*, https://www.baumer.com/ch/de/service-support/funktionsweise/funktionsweise-und-technologie-von-ultraschallsensoren/a/Know-how_Function_Ultrasonic-sensors

[2] Microsonic: *Ultraschallsensoren*, https://www.microsonic.de/de/service/ultraschallsensoren/prinzip.htm

Hinderniserkennungsroboter

Xu Pai, Elektromobilität
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—In diesem Paper wird ein Ultraschall-Hindernisvermeidungssystem für das autonome Fahren vorgestellt. Das System ist in der Lage, Hindernisse zu erkennen und eine Vermeidung zu ermöglichen. Die Ergebnisse zeigen, dass das System eine hohe Genauigkeit aufweist und eine zuverlässige Hindernisvermeidung ermöglicht.**

***Schlagwörter*—Hinderniserkennung, Ultraschall, Roboter, Matlab, Programm**

I. Einleitung

DAS selbstfahrende Auto hat in den letzten Jahren erhebliche Fortschritte bei der Entwicklung selbstfahrender Autos gemacht. Autonome Fahrzeuge sind in der Lage, ohne menschliche Intervention zu fahren und können so das Unfallrisiko erheblich reduzieren. Eines der wichtigsten Probleme bei der Entwicklung selbstfahrender Autos ist die Sicherheit. Ein Ultraschallhindernisvermeidungssystem ist eine der wichtigsten Technologien, die bei der Entwicklung selbstfahrender Autos eingesetzt wird. Das Ziel dieses Papiers ist es, ein Ultraschall-Hindernisvermeidungssystem für das Selbstfahren vorzustellen und zu evaluieren.

II. Vorbetrachtung

Diese Realisierung der Funktion basiert auf der Sensorik des Systems. Derzeit gibt es verschiedene Ansätze, um autonomes Fahren zu ermöglichen: kamerabasiertes, lidarbasiertes, ultraschallbasiertes autonomes Fahren. Wir konzentrieren uns heute auf das ultraschallbasierte autonome Fahren.

A. Hintergrund

Die Natur ist eine unerschöpfliche Inspirationsquelle für die technologische Entwicklung des Menschen. So wird beispielsweise die Biosonartechnologie von Fledermäusen und Delphinen genutzt, um Ultraschallwellen zur Objekterkennung einzusetzen. [1] Tatsächlich verfügt der Mensch über natürliche Sinne, wie die Augen als Kamerasensoren, um Bilder zu erfassen und zu verarbeiten. Und während der Mensch sich auf Radar- und Lidarsensoren als wichtige Sensoren für die Objekterkennung verlässt, ist es wichtig zu beachten, dass diese Sensoren Lichtwellen verwenden. Der Unterschied besteht jedoch darin, dass Radar elektromagnetische Wellen verwendet, während Lidar auf Laserlicht basiert. Im Vergleich zu Kameras und Ultraschallsensoren bieten Lidar und Radar eine höhere Erkennungsgenauigkeit und Störfestigkeit, allerdings zu höheren Kosten.

DOI: 10.24352/UB.OVGU-2023-020

B. Eigene Lösung

In dieser Arbeit konzentrieren wir uns auf die Verwendung von Ultraschall als Sensortyp, um einen Lego Mindstorms Roboter zur Hindernisvermeidung zu entwickeln. Mithilfe von Ultraschallsensoren und MATLAB-Programmierung kann der Roboter Hindernisse erkennen und ihnen ausweichen.

C. Aktuellemodul

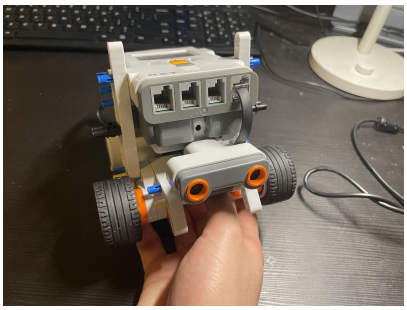


Abbildung 1. Aufgebautes Modellauto

Die Abbildung 1 zeigt die Ansicht des aufgebauten Autos. Die Abbildung 2 zeigt, dass das Modellauto aus drei Teilen besteht: Sensoren, Antriebssystem und Steuerungssystem. Die Sensoren sind Ultraschallsensoren, die alle 0,1 Sekunden Ultraschallwellen aussenden. Das Antriebssystem besteht aus zwei unabhängigen Motoren, einer Antriebsstange und Gummireifen. Das Steuerungssystem sendet Befehle an den NXT-Stein, der dann die beiden anderen Systeme steuert.

Abbildung 2. Bestehende Teile

III. Design und Umsetzung

Die Abbildung 3 zeigt ein Ablaufdiagramm für die Prüfung der Bedingung, ob ein Hindernis vorhanden ist. Wenn kein Hindernis vorhanden ist, fährt das System kontinuierlich vorwärts. Wird jedoch ein Hindernis erkannt, dreht das System um einen kleinen Winkel, bis kein Hindernis mehr erkannt wird.

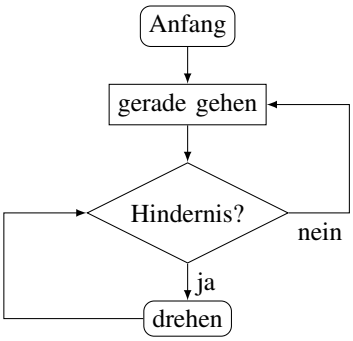


Abbildung 3. Programmablaufplan

A. Gleichungen

Das Prinzip der Ultraschall-Entfernungsmessung beruht darauf, dass Schallwellen mit einer Frequenz oberhalb des menschlichen Hörbereichs ausgesendet werden. Diese Schallwellen werden vom Objekt reflektiert und von einem Empfänger aufgenommen. Durch die Messung der Zeitdauer zwischen dem Aussenden und Empfangen der Schallwellen kann die Entfernung zum Objekt berechnet werden. [1].

$$2s = vt \tag{1}$$

Mit

- s: Abstand zwischen Objekt und Sensor
- v: Geschwindigkeit der Ultrawelle
- t: die Zeit, die der Ultraschall benötigt, um diese Strecke zurückzulegen

B. Programmablauf

In diesem Experiment wird der Steuerungsbefehl in drei Teile unterteilt: Erkennung, Verarbeitung und Ausgabe. Zunächst wird mit dem Ultraschallsensor der Abstand zwischen dem Hindernis und dem Roboter ermittelt, um festzustellen, ob der Roboter eine Drehung ausführen soll. Anschließend wird die Drehung des Roboters durch die Anzahl der Motorumdrehungen und die Stärke der Drehkraft gesteuert. Schließlich kann die Drehbewegung abgeschlossen werden, indem die beiden unabhängigen Motoren in entgegengesetzte Richtungen gedreht werden.

IV. Ergebnisdiskussion

Während der Identifikationsphase kann es gelegentlich zu Problemen bei der Erkennung kleiner Hindernisse kommen, da die Leistung oder Genauigkeit des Sensors zu gering ist. In der Verarbeitungsphase kann es zu einem Problem kommen, wenn der Abstand zum Startpunkt nicht vorher in MATLAB definiert wurde, was zu einem Abbruch der Schleife führen kann. Wenn der Abstand zwischen dem Ultraschallsensor und dem Hindernis zu klein ist, kann der Roboter abstürzen, weil das NXT-Stein zu viele Entscheidungen treffen muss. Schließlich kann ein Rad blockieren und in den Bremszustand übergehen, wenn die Drehzeit zu lang ist. Dieses Problem kann gelöst werden, indem die Motordrehkraft erhöht und somit die Drehung beschleunigt wird.

V. Zusammenfassung und Fazit

Obwohl dieser Roboter noch sehr einfach ist, kann er das Grundprinzip eines hindernisvermeidenden Roboters und einen Teil seiner Anwendungsbereiche darstellen. Das bedeutet, dass er mit Hilfe von Sensoren seine Position im Raum oder seinen Abstand zu Hindernissen bestimmen kann und daraufhin eine Reihe von Reaktionen ausführt. Hinderniserkennungsroboter sind ein wichtiger Bestandteil der Automatisierung in verschiedenen Bereichen. Mit dem technologischen Fortschritt und der Entwicklung neuer Sensoren werden diese Roboter immer leistungsfähiger und können in einer Vielzahl von Anwendungen eingesetzt werden. Es ist zu erwarten, dass Roboter zur Hinderniserkennung in Zukunft noch wichtiger werden und in immer mehr Bereichen eingesetzt werden.

Anhang

```
handle = COM_OpenNXT()
COM_SetDefaultNXT(handle)
MotorA = NXTMotor('A');
MotorC = NXTMotor('C');

port = SENSOR_4;
OpenUltrasonic(port);
distance = 100;
//setzen den Anfangswert,
sonst startet das Roboter nicht
while (true) //Bedingung
    distance = GetUltrasonic(port);
    if (distance < 25)
//drehen
        MotorA.Power = 60;
        MotorC.Power = -60;
        MotorA.TachoLimit = 115;
        MotorC.TachoLimit = 115;
        MotorA.SendToNXT;
        MotorC.SendToNXT;
//gerade gehen
    else
        MotorA = NXTMotor('A','Power',60);
        MotorC = NXTMotor('C','Power',60);
        MotorA.TachoLimit = 0;
        MotorC.TachoLimit = 0;
        MotorA.SendToNXT();
        MotorC.SendToNXT();
    end
end
```

Literaturverzeichnis

[1] Weiwen Zhu: *Auswertung des Ultraschall-Abstandssensors für Auto-Rückfahrkollisionsschutzsysteme*. Guangzhou, China: Guangzhou city transportation occupation school, 2021

Artemis: Der Bogenschießroboter

Owis Alsabbagh, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Im Rahmen des diesjährigen LEGO-Mindstorms-Praktikums wurde mit Hilfe des LEGO-Baukastens ein Roboter konstruiert, der Mithilfe von NXT-Steuerung und MATLAB-Programmierung in der Lage ist, einen Pfeil abzuschießen und mit Hilfe eines Abstandssensors ein Obejkt in einem vorgegebenen Bereich zu erkennen.In den folgenden Abschnitten werden die einzelnen Schritte der Entwicklung des Roboters Artemis beschrieben.**

***Schlagwörter*—Artemis, Bogenschütze, LEGO-Mindstorms, MATLAB, Ultraschallsensor**

I. Motivation

DIE griechische Göttin Artemis des Olymp wurde in der Mythologie mit der Wildnis , der Jagd und dem Mond assoziiert. Mit dem goldenen Bogen Khryselakatos und ihrem Pfeilregen des Todes Iokheaira schoß sie ihre Ziele nieder [1]. Die Erzählungen und Gedichte über Artemis haben viele wissenschaftliche Projekte inspiriert, die ihren Namen tragen, und dieses Projekt ist keine Ausnahme. Im Laufe der Geschichte sind je nach Region verschiedene Bogenformen entstanden.Mit der Entwicklung der Waffenindustrie wurden die Bögen leistungsfähiger, präziser und leichter zu handhaben, so dass später Armbrüste entwickelt wurden und moderne, vollautomatische Bogenschießanlagen entstanden. Außerhalb der Waffenindustrie ist das Bogenschießen eine hoch angesehene Sportart, die heute in vielen Kulturen zu finden ist. Auch wenn die Form des Bogens unterschiedlich dargestellt wird, ist die Funktion die gleiche: Eine elastische Sehne wird zwischen zwei festen Punkten eines gebogenen Stabs gespannt. Durch den Rückstoß eines stabförmigen Projektils kommt es zu einer Überspannung des Fadens, die nach dem Loslassen dazu führt, dass das Projektil „geschossen“ wird. Ziel des Projektes ist es, einen Roboter zu entwickeln, der diesen Schussvorgang autonom durchführen kann.

II. Vorbetrachtungen

Bereits zu Beginn der Entwicklung des Roboters stellte sich die erste Frage. Ist es möglich, eine solche Konstruktion mit LEGO-Steinen zu bauen? Und wenn ja, wie? Im folgenden Abschnitt werden die Ergebnisse der Recherche vorgestellt und anschließend der selbst gewählte Lösungsansatz präsentiert.

A. Die Inspiration

Die ersten Recherchen führten zu einer Quelle mit dem Design (Abb. 1) des Online-Entwicklers Pedro Rosa [2]. Dieses Design diente als Ausgangspunkt für die Realisierung der eigenen Lösung.

DOI: 10.24352/UB.OVGU-2023-021

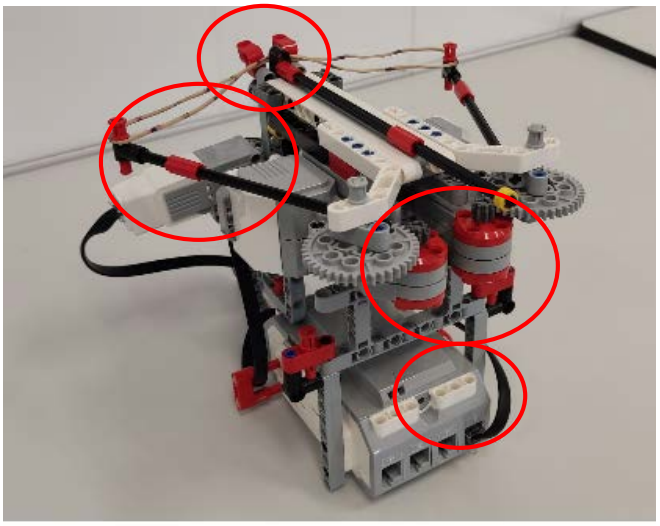


Abbildung 1. Design von Pedro Rosa [2]

Das Design besteht aus einer Konstruktion mit zwei Motoren für die Bewegung der Bogenarme und einem Motor für das Heben und Senken des „Triggers“. Mit dieser Konstruktion ist die erste Frage der Vorüberlegungen beantwortet und es werden Lösungsansätze zur Erweiterung der Konstruktion von Pedro Rosa untersucht.

B. Eigener Lösungsansatz

Um die Funktion bzw. den Einsatzbereich des Roboters zu optimieren, ist es notwendig, die Anzahl der Motoren, die für den Schussvorgang verantwortlich sind, zu reduzieren, da der NXT-Block nur über drei Motoranschlüsse verfügt. Durch den Umbau des Bewegungsmechanismus der Bogenarme von zwei Motoren auf einen Motor kann der Arbeitsbereich des Roboters erweitert werden, so dass der dritte Motor für die Fortbewegung verwendet werden kann.

III. Konstruktion

In diesem Abschnitt wird die Konstruktion des Roboters Schritt für Schritt vom Prototyp bis zum Endprodukt beschrieben.

A. Der Prototyp

In Abb. 2 ist der erste Aufbau von Artemis zu sehen. Der Prototyp wurde gebaut, um sich mit dem Design von Herrn Rosa [2] vertraut zu machen bzw. als Ausgangspunkt für weitere Entwicklungsschritte zu dienen.

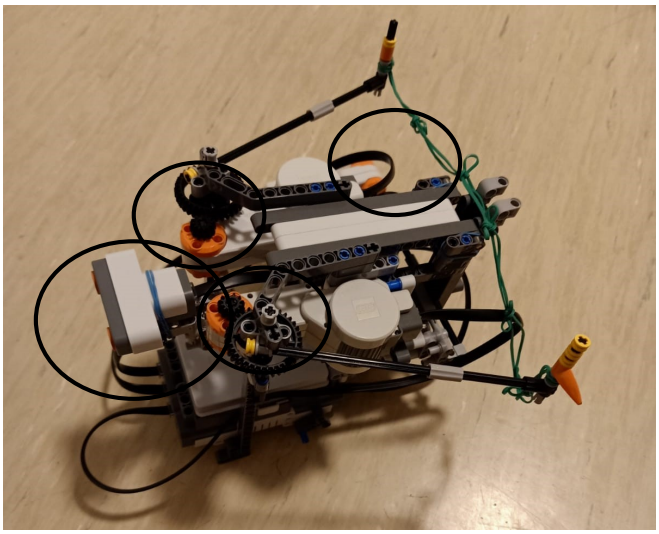


Abbildung 2. Prototyp

Bei genauerer Betrachtung zeigten sich bereits die ersten Einschränkungen. Die Bewegung der Bogenarme über zwei Motoren war trotz der Einstellung auf eine Motorkraft asynchron, was daran liegt, dass der NXT-Block immer nur eine Codezeile verarbeiten kann und die Befehle im MATLAB-Programm mit zwei getrennten Codeabschnitten realisiert sind. Dies führt zu einem Versatz in der Bewegung der Arme und damit zu einer Beeinträchtigung der Funktion. Im folgenden Abschnitt wird ein möglicher Lösungsansatz für das oben genannte Problem sowie erste eigene Entwicklungsschritte vorgestellt.

B. Umbau des Spannmechanismus

Um die Anzahl der eingesetzten Motoren zu reduzieren, wurde folgende Konstruktion (Abb. 3) für den Spannmechanismus der Bogenarme gebaut.

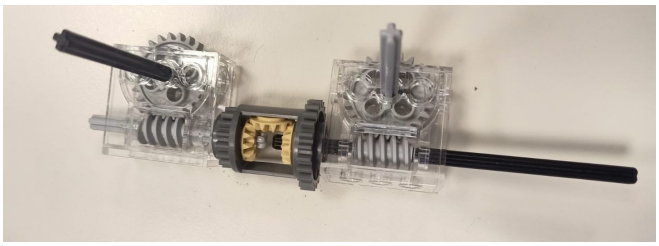


Abbildung 3. LEGO-Konstruktion für die Bewegung der Arme

Durch diesen Umbau wird eine synchrone Bewegung der Arme erreicht. Da die Bewegung der „Trigger“ bereits mit der ersten Konstruktion erfolgreich war, wurde das erste Ziel erreicht. Die zweite Version von Artemis ist in Abb. zu sehen. 4 zu sehen.

C. Finale Version

In Abb. 5 ist die letzte und endgültige Version von Artemis-der Bogenschießroboter zu sehen. Nach dem erfolgreichen

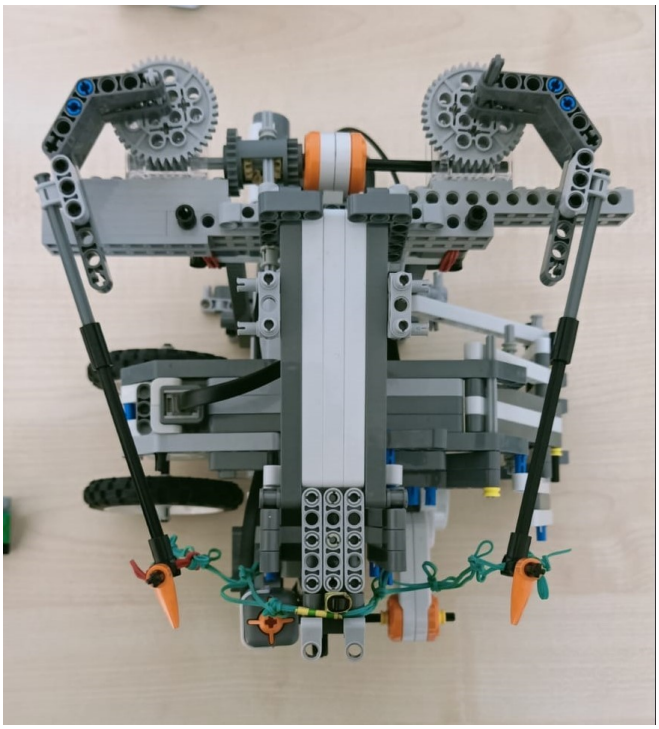


Abbildung 4. Artemis 2.0

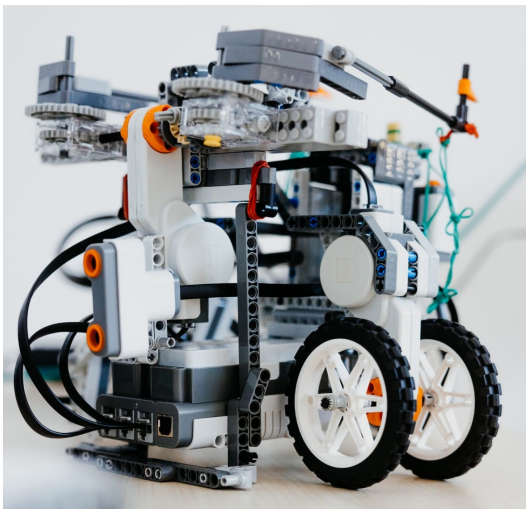


Abbildung 5. Artemis – der Bogenschießroboter [3]

Umbau der ersten Konstruktion war das Hauptziel bereits erreicht und zwei neue Ziele wurden in Angriff genommen. Erstens die Auslösung des Schusses durch den Ultraschallsensor und zweitens die Fortbewegung des Roboters. Da die Montage des Sensors einfach zu realisieren war, konnte das zweite Ziel mit Hilfe eines einfachen MATLAB-Programms erreicht werden. Außerdem wurde der verbleibende Motoranschluss für die Fortbewegung genutzt, indem auf der einen Seite des Roboters ein Motor mit angeschlossenen Rädern und auf der anderen Seite eine Stützkonstruktion angebracht wurde. Damit war das dritte Ziel erreicht. Diese Eigenschaften sind in Abb. 4 und 5 dargestellt.

IV. Software

In diesem Abschnitt wird das MATLAB-Programm kurz erläutert. Die verschiedenen Operationen, die das Gerät ausführen soll, sind in den folgenden MATLAB-Funktionen kodiert:

`Schiessen();`	Schussvorgang
`Reset();`	Zurückführen der Bogenarme
`move();`	Bewegung in eine Richtung
`move2();`	Bewegung in die Gegenrichtung
`killswitch();`	Abbruch aller Abläufe

Diese MATLAB-Funktionen wurden in einem While-Loop in einer bestimmten Reihenfolge mit einer If-Bedingung codiert, die den Bewegungsablauf unterbricht und den Schuss auslöst, wenn der Signalwert des Ultraschallsensors einen bestimmten Wert unterschreitet. Der folgende Programmablaufplan veranschaulicht die Funktionen des Gerätes.

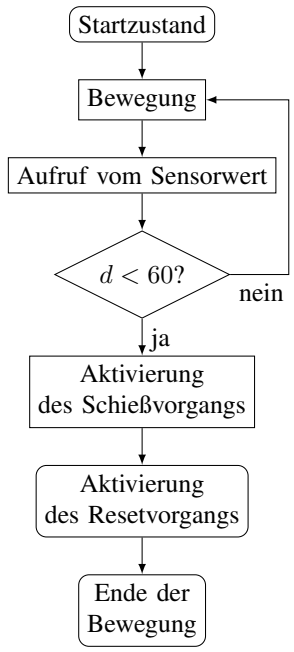


Abbildung 6. Programmablaufplan

V. Ergebnisdiskussion

Nach zahlreichen Funktionstests wurde festgestellt, dass der Roboter die gestellten Anforderungen erfolgreich erfüllen konnte. Allerdings führte die Stützfunktion auf der rechten Seite des Roboters zu einer bogenförmigen Bewegung anstelle der erwarteten geradlinigen Bewegung. Darüber hinaus stellten die physikalischen Grenzen der LEGO-Steine bzw. des NXT-Blocks ein ständiges Hindernis während der gesamten Entwicklung dar. Das Gewicht der zusätzlichen Räder auf der rechten Seite hätte dazu geführt, dass der Motor nicht genügend Kraft hätte aufbringen können, um Artemis zu bewegen. Aus diesem Grund wurde die oben beschriebene Stützkonstruktion gewählt. Die Verarbeitungsgeschwindigkeit des NXT-Blocks konnte mit dem Ablauf der Operationen nicht mithalten, dieses Problem wurde durch das Einfügen von Pausen in das MATLAB-Programm gelöst.

VI. Zusammenfassung und Fazit

Zusammenfassend kann gesagt werden, dass die endgültige Konstruktion des Gerätes alle gesteckten Ziele erreichen konnte. Dies wurde durch die Programmierung in MATLAB erreicht. Eine Überlegung für die Weiterentwicklung wäre der Wechsel von der NXT-Steuerung zur EV3, da diese eine höhere Anzahl an Motoranschlüssen besitzt und somit zwei Motoren für die Bewegung verwendet werden könnten. Ein weiterer Vorschlag wäre der Einbau eines Farbsensors oder einer Webcam, um einen reibungslosen Ablauf der Bewegung zu gewährleisten.

Anhang

```
OpenUltrasonic(3);
OpenSwitch(2);
while true
move();
pause(0.5);
if GetUltrasonic(3) <= 60
    motorC = NXTMotor('C');
    motorC.Power = 0;
    motorC.SendToNXT();
    pause(2);
    Schiessen();
    pause(2);
    Reset();
    break
elseif GetSwitch(2) == 1
    killswitch();
    break
end
pause(2);
move2();
pause(1.5);
end
```

Abbildung 7. MATLAB-Programm

Literaturverzeichnis

[1] Theoi Greek Mythology: *Artemis*. https://www.theoi.com/Olympios/Artemis.html. Version: Februar 2017

[2] Pedro Rosa: *Archery*. https://sites.google.com/view/profpedromrosa-lego/. Version: April 2022

[3] Theile, Hannah: *MKM, OVGU*. https://www.picdrop.com/janaduennhaupt/BeWVACovJi. Version: Februar 2023

Artemis – der automatisierte Bogenschütze Roboter zum Treffen der Ziele

Hisham Mohamed Eid, Elektrotechnik und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Während des jährlichen Projektseminars „Elektrotechnik und Informationstechnik“ an der Otto-von-Guericke-Universität war ein Bogenschützen-Roboter aufzubauen, der durch den Einsatz der NXT-Steine die Kompetenz besaß, Ziele mit Pfeilen zu treffen. Im vorliegenden Bericht ist die Umsetzung des Roboters beschrieben. Hierbei werden insbesondere die Konstruktionsmerkmale, wichtige Programmteile und auftretende Probleme betrachtet.**

***Schlagwörter*—Automatisierung, Bogenschütze, Pfeil, Design, Schusstechnik, Zielerkennung**

I. Einblick in die Motivation

DAS Bogenschießen ist ein alter Sport mit einer jahrtausendealten Geschichte. Ursprünglich als Jagd- und Kriegswaffe eingesetzt, hat sich das Bogenschießen in den letzten Jahren zu einem beliebten Freizeit- und Wettkampfsport entwickelt. Es erfordert Konzentration, Geschicklichkeit, Willenskraft und eine gute Körperhaltung. Ziel dieses Projekts ist es, einen automatisierten Bogenschützen zu entwickeln, der in der Lage ist, sich zu bewegen, Objekte zu erkennen, Pfeile abzuschießen und wieder in den Ausgangszustand zurückzukehren. Es soll gezeigt werden, wie eine Lösung zur perfekten Beherrschung der Schusstechnik entwickelt werden kann, um ein breites Publikum für diese Technologie zu begeistern und das Verständnis für die Funktionsweise eines Bogenschützen zu fördern.

II. Vorbetrachtungen

A. Schusstechnik im 21. Jahrhundert

Im 21. Jahrhundert nimmt die Entwicklung der Robotik im Bogensport deutlich zu. Mittlerweile gibt es automatisierte Bogenschützen, die speziell für das Training und den Wettkampf entwickelt wurden. Diese Roboter können dazu beitragen, das Training der Schützen zu verbessern. Einige moderne Bogenroboter können verschiedene Arten von Bögen verwenden und Pfeile mit unterschiedlicher Geschwindigkeit und Kraft abschießen. Der Einsatz von Bogenrobotern verbessert nicht nur das Training der Bogenschützen, sondern macht den Sport auch für alle zugänglicher. Menschen, die körperlich behindert oder nicht in der Lage sind, das traditionelle Bogenschießen auszuüben, können nun den Bogensport ausüben.
Es gibt eine andere außergewöhnliche Sportart, die unser Projekt zum Ausdruck bringt, und es ist Armbrustsport. Die bloße Erwähnung von „Armbrust“ zeichnet bei den meisten Menschen ein geistiges Bild, oft eine Waffe aus dem Mittelalter, die für die Kriegsführung entwickelt wurde. Allerdings hat sich die Armbrust zu einem hochentwickelten Sportgerät entwickelt. Während das alte Prinzip erhalten bleibt, ist seine Anwendung in der Neuzeit weit von seinen Ursprüngen entfernt. Eine Mischung aus Tradition und Präzisionssport, beschreibt Markus Peschel, ein begeisterter Armbrustschütze aus Freising, es als eine einzigartige und wettbewerbsintensive Aktivität [1].

DOI: 10.24352/UB.OVGU-2023-022

B. LEGO® und Design Thinking

Da das Projekt unter Verwendung von LEGO®-Teilen und dem Hauptteil, der mit den Motoren und Sensoren verbunden ist, nämlich NXT 2.0, durchgeführt wurde, wurde zunächst überlegt, wie die folgenden Hauptkomponenten des Bogenschützen realisiert werden können. Zunächst gibt es Bögen in vielen verschiedenen Varianten. Dazu gehören der Langbogen, der Recurvebogen und der Compoundbogen - jeder mit seinen eigenen Eigenschaften und Zwecken. Mit jedem Bogentyp werden verschiedene Arten des Bogenschießens ausgeführt. Pfeile gibt es in verschiedenen Gewichten, Schäften und Federfarben. Sie werden entsprechend der Sportart ausgewählt, für die sie verwendet werden. Einige Pfeile haben eine Spitze, einen Schaft und Stabilisierungsfedern. Ein Visier ist ein einfaches Gerät, das am Bogen befestigt wird und die Sicht des Schützen verbessert. Es besteht aus einem Rahmen, einem Stift und einer Skala, mit der der Schütze die Entfernung zum Ziel einstellen kann. Um auf die Umsetzung der Idee mit LEGO® zurückzukommen, begannen wir mit der Umsetzung des ersten Modells unseres Projekts, wobei zwei Motoren verwendet wurden, um das Gummiseil zu ziehen, und ein dritter Motor, um die Welle nach unten zu ziehen, die das Gummiseil hält, so dass der Pfeil getroffen wird. [2].

C. Kreativität im Fokus

Als Alternative zur Realisierung des Projekts mit der geringsten Anzahl von Motoren, die benötigt werden, um den Pfeil auf das Ziel zu schießen, und aufgrund einiger Probleme, die beim Prototyp auftraten, wurde eine praktischere Idee gefunden, einen Motor zum Ziehen des Gummiseils zu verwenden.

III. Technischer Aufbau und Funktionsprinzip

Jetzt ist es möglich, Schritt für Schritt zu erklären, wie der Roboter gebaut, programmiert und in Betrieb genommen wurde.

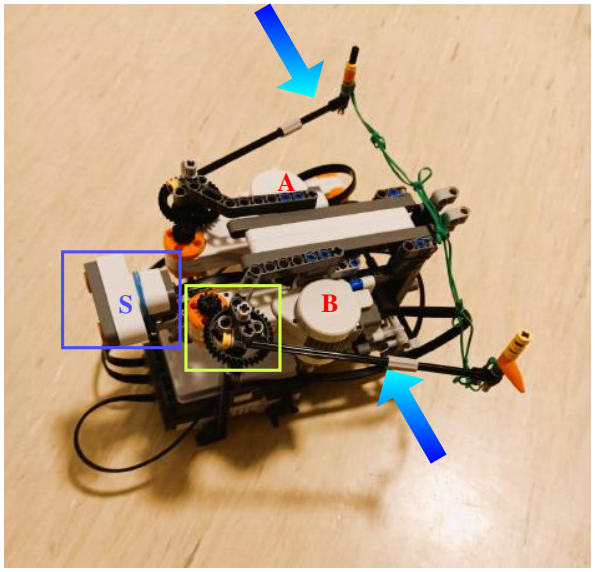


Abbildung 1. Draufsicht des ersten Modells

A. Aufbau des Prototyps

Abbildung 1 zeigt den ersten Entwurf eines automatischen Bogenschützen, bei dem die Motoren (A) und (B) deutlich sichtbar sind. Diese Motoren steuern die Bewegung der Bogenarme, deren Enden mit einem Gummiband befestigt sind, in entgegengesetzte Richtungen In dem gelben Kästchen in der oberen Abbildung befinden sich zwei Zahnräder, die ineinander greifen. Das eine ist klein und mit dem Motor verbunden, das andere ist groß und mit dem Bogenarm verbunden. Der blaue Buchstabe (S) in der Abbildung 1 bezeichnet den Abstandssensor, der zur Erkennung des Ziels und damit zum Auslösen des Pfeils verwendet wird. Abbildung 2 zeigt die

Abbildung 2. Seitenansicht des ersten Modells

fehlende Seite der ersten Abbildung, auf der der dritte Motor (C) des Roboters deutlich zu sehen ist. Dieser Motor ist an einer horizontalen Säule befestigt, die auf eine vertikale Säule drückt, hinter der sich die Mitte des Gummiseils befindet, bis es gespannt ist. Wenn der Abstandssensor das Ziel erkennt, dreht sich der Motor (C) in einem bestimmten Winkel und mit einer bestimmten Geschwindigkeit nach unten, zieht die vertikale Welle nach unten und schießt den Pfeil auf das Ziel. In Abbildung 2 ist die Position, an der der Pfeil abgeschossen wird, durch einen Pfeil mit dem Buchstaben (P) daneben gekennzeichnet. Der Buchstabe (T) weist auf die Taste hin, mit der der Pfeil alternativ direkt abgefeuert werden kann.

B. Vom Prototyp zur Vollendung

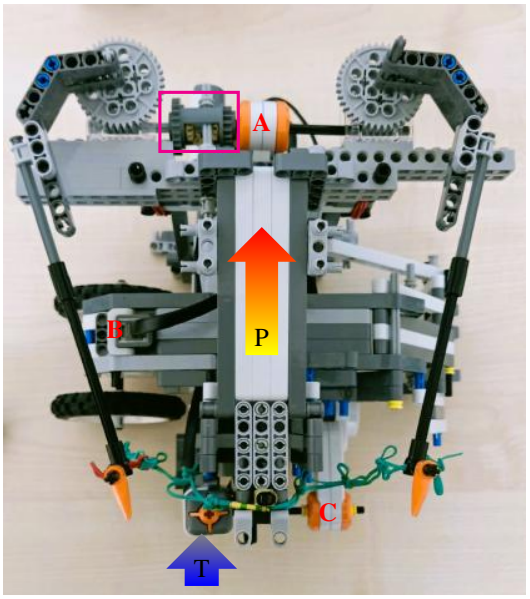


Abbildung 3. Draufsicht des endgültigen Modells

Abbildung 3 zeigt das endgültige Design des Bogenschützenroboters, das einige Unterschiede zum ursprünglichen Design aufweist, jedoch effizienter ist und den Mechanismus des Roboters verbessert. Der erste offensichtliche Unterschied besteht darin, dass der Motor B vertikal auf zwei Rädern auf der linken Seite des Roboters montiert ist, um den Roboter auf einer gekrümmten Bahn zu bewegen, damit er das Ziel in seiner Umgebung besser sehen kann. Motor A ist nun allein für die Bewegung der beiden Bogenarme verantwortlich, da er mit einem Getriebe mit zwei gegenläufigen Wellen verbunden ist.
Die endgültige und ideale Form des Bogenschützenroboters ist in Abbildung 4 gut zu erkennen. Die Position des Abstandssensors in dieser Abbildung unterscheidet sich von der in Abbildung 1, da er um 90 Grad gedreht und etwas tiefer angebracht wurde, um den Pfeil beim Abschuss nicht zu behindern.

C. Programmierung

Für die Programmierung des Projekts wurde MATLAB verwendet. Zuerst wurde eine Reihe von Funktionen mit unterschiedlichen Eigenschaften erstellt. Die erste Funktion steuerte die Bewegung des Geräts, indem sie die Rotationsgeschwindigkeit von Motor B für eine bestimmte Zeit in eine bestimmte Richtung steuerte und dann mit der gleichen

Abbildung 4. Bogenschütze [3]

Geschwindigkeit in die Ausgangsposition zurückkehrte. Die zweite Funktion implementierte das Abschießen des Pfeils auf das Ziel, sobald es erkannt wurde, indem zuerst die Bewegung von Motor A gesteuert wurde und dann die vertikale Welle, die mit Motor C verbunden war, bewegt wurde, um das Gummiseil zu lösen. Die Erkennung des Ziels erfolgte durch den Abstandssensor, der prüfte, ob sich das Ziel innerhalb oder außerhalb des festgelegten Bereichs von $60\,\mathrm{cm}$ befand. Bei der Programmierung traten keine Probleme auf.

IV. Ergebnisdiskussion

Der Roboter bewegt sich, erkennt das Ziel und schießt den Pfeil darauf. Natürlich ist dieser Roboter nicht vollständig automatisiert, da es notwendig ist, die Mitte des Gummiseils hinter der vertikalen Säule neu auszurichten und den Pfeil manuell an seine Position zu bringen. Trotz der mäßigen Qualität des Sensors und der Motoren war die Leistung des Roboters zufriedenstellend.

Beim ersten Modell des Projekts traten einige mechanische Probleme auf, die jedoch beim endgültigen Modell behoben wurden. Das Hauptproblem bestand darin, dass die Bewegungen der Motoren A und B nicht synchronisiert waren, was sich negativ auf den Schussvorgang auswirkte. Das zweite Problem waren die fehlenden Eingänge für die Kabelverbindungen im Hauptteil des NXT 2.0, was eine Änderung des ersten Entwurfs erforderlich machte. Im endgültigen Design wurden die beiden Bogenarme von einem Motor bewegt, während ein dritter Motor

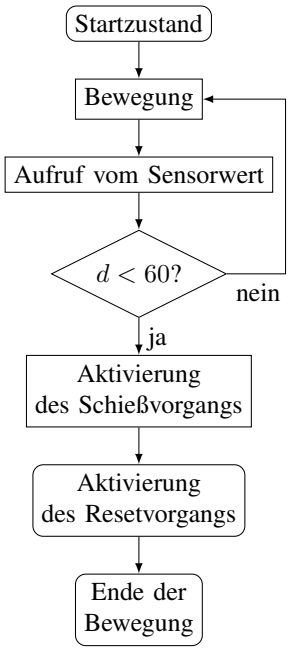


Abbildung 5. Programmablaufplan

für die Bewegung des Roboters verwendet wurde. Schließlich gab es noch ein letztes Problem, bei dem die Bewegung der vertikalen Welle zur Steuerung der Bewegung des Gummiseils dazu führte, dass das kleine Zahnrad durch ein größeres ersetzt werden musste, um das Problem zu lösen.

V. Zusammenfassung und Fazit

Die Ziele des Projektseminars waren erfolgreich erreichbar, indem ein Bogenschützenroboter entwickelt wurde, der alle ihm übertragenen Aufgaben erfolgreich bewältigte. Obwohl es einige anfängliche Probleme gab, wurden diese Probleme erfolgreich gelöst.

Literaturverzeichnis

[1] Tobias Meindl, Freising: *Tradition trifft Präzision*. https://www.sueddeutsche.de/muenchen/erding/freising-tradition-trifft-praezision-1.4805654. Version: Februar 2020

[2] Bogensportwelt: *Bogenschießen: Die Grundlagen und Wissenswertes*. https://www.bogensportwelt.de/magazin/bogenschiessen-die-grundlagen-und-wissenswertes. Version: 2021

[3] Hannah Theile, MKM, OVGU: *LEGO Abschlussveranstaltung*. https://www.picdrop.com/janaduennhaupt/BeWVACovJi. Version: Februar 2023

Smart Car

Omar Marzouk, Elektro- und Informationstechnik

Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Das Projektseminar Elektrotechnik und Informationstechnik wird jedes Jahr an der Otto-von-Guericke Universität Magdeburg durchgeführt. Im Rahmen des diesjährigen Projektseminars wurde ein Auto erstellt. Dieses hilft dabei , die Anzahl der Verkehrsunfälle zu verringern. Die Entwicklung und Konstruktion dieses Auto erfolgten auf Basis von LEGO-Mindstorms-Sets sowie des LEGO-NXT-Steuerungscomputers. Die softwareseitige Implementierung wurde über MATLAB realisiert. In diesem Paper werden der Aufbau und die Funktionsweise des Autos vorgestellt. Es wird weiterhin auf einige Herausforderungen während des Konstruktionsprozesses, sowie auf deren Lösungsansätze eingegangen.**

***Schlagwörter* Smart Car, MATLAB, LEGO-Mindstorms, Farbsensor, Ultraschallsensor.**

I. EINLEITUNG

JEDER glaubt – auf die eine oder andere Weise – dass selbstfahrende Autos die Zukunft sind, und deshalb sieht man, dass die meisten Autohersteller versuchen, diese Technologie zu entwickeln und ihre Erkenntnisse zu überprüfen. Und das ist, was dieses Projekt auch wollte.

Laut Weltgesundheitsorganisation (WHO) sterben jährlich etwa 1,3 Millionen Menschen an den Folgen von Autounfällen, und anderen Quellen zufolge gibt es in Deutschland jährlich mehr als 2 Millionen Autounfälle [1]. All dies lässt die Alarmglocken läuten und unterstreicht die Bedeutung der Zusammenarbeit, um dieses Problem zu lösen.

Autounfälle können durch verschiedene Faktoren verursacht werden, wie Ablenkung, Geschwindigkeit, Alkohol- und Drogenkonsum, Wetterbedingungen, technische Probleme und Fahrunfähigkeit. Es gibt noch viele weitere Ursachen für Autounfälle, aber diese Beispiele zeigen, wie verschiedene Faktoren dazu beitragen können, dass ein Unfall passiert.
Somit ist ersichtlich, dass diese Technologie – das Selbstfahren – so viel wie möglich bei der Lösung dieses Problems helfen wird.

Abbildung 2: Weltweit führende Unternehmen, die an selbstfahrenden Autos arbeiten [2]

DOI: 10.24352/UB.OVGU-2023-023

Abbildung 1: Nissan Invisible-to-Visible-Technologie auf der CES 2019

II. VORBETRACHTUNGEN

In diesem Teil werden die wichtigsten Tools, die in diesem Projekt verwendet wurden, und die Funktion jedes einzelnen kurz erläutert

Allgemein über LEGO-NXT:

LEGO NXT ist eine Serie von LEGO-Robotik-Bausätzen, die für Bildung und Freizeit entwickelt wurden. NXT steht für "NeXt Generation Robotics" und bezieht sich auf die fortgeschrittene Technologie und Programmierbarkeit der Roboter. Diese Bausätze ermöglichen es Benutzern, komplexe Robotermodelle aus LEGO-Teilen zu bauen und diese mit einer Programmieroberfläche zu programmieren, um sie zu steuern und Aufgaben auszuführen. Die NXT-Serie wurde 2006 von LEGO auf den Markt gebracht und hat seitdem viele Iterationen und Verbesserungen erfahren. Sie wird häufig in Bildungseinrichtungen für den MINT-Unterricht (Mathematik, Informatik, Naturwissenschaften und Technik) eingesetzt und hat auch bei Robotik-Enthusiasten und Hobbyisten eine große Anhängerschaft gefunden.

LEGO NXT enthält verschiedene Arten von Motoren und Sensoren, die es Benutzern ermöglichen, ihre Robotermodelle zu programmieren und zu steuern. Diese Motoren und Sensoren können durch die Verwendung von LEGO NXT-Softwareprogrammen programmiert werden, um bestimmte Aktionen auszuführen oder auf bestimmte Eingaben zu reagieren.

III. KONSTRUKTION UND REALISIERUNG

A. Aufbau

Ein Auto aus LEGO-Bausteine zu bauen ist eigentlich gar nicht so schwer. Alles, was man braucht, sind zwei Motoren,

die die gleiche Funktion erfüllen, und auch einige Sensoren wie z.B. Ultraschallsensor und Farbsensor. Aber diese Teile müssen sorgfältig installiert werden, damit jedes von ihnen seine richtige Position einnimmt (siehe Abbildung 3 und 5).

Abbildung 3: Smart Car aus LEGO-Bausteine

1. *Motoren*

Die Motoren haben eine Aufgabe: sie drehen sich um. Dabei kann man die Geschwindigkeit durch Programmierung auswählen. Der Standard-Motor kann sich in beide Richtungen drehen, um Vorwärts- und Rückwärtsbewegungen zu erzeugen.

Farbsensor scannt die Farbe einer Oberfläche und gibt einen RGB-Wert zurück, welcher möglichst die Farbe der Oberfläche darstellt. Damit lassen sich die Farben einer Ampel erkennen.

2. *Farbsensor*

Mit diesem Sensor kann der Roboter die Farben von Objekten erkennen und darauf reagieren. Der Farbsensor arbeitet, indem er Licht von einem Objekt aufnimmt und analysiert, welches Farbspektrum reflektiert wird. Der Sensor kann sechs verschiedene Farben erkennen: Rot, Grün, Blau, Gelb, Weiß und Schwarz.

3. *Ultraschallsensor*

Dieser Sensor verwendet Ultraschallwellen, um die Entfernung von Objekten zu messen und auf Bewegungen zu reagieren. Der Sensor sendet einen Ultraschallimpuls aus, der von einem Objekt reflektiert wird, bevor er zum Sensor zurückkehrt. Der Sensor berechnet dann die Entfernung des Objekts basierend auf der Zeit, die für das Zurückkehren des Signals benötigt wird.

B. Programmierung

Wie schon vorher ernannt wurde: Ziel diese Projekt Programmierung in MATLAB zu lernen. Deshalb die Programmierung ist der wichtigste Teil dieses Projekt und darauf wurde konzentriert Studenten Fähigkeit zu entwickeln.
Am Anfang haben sich die Studenten eine allgemein MATLAB durchgeführt. Dann mit Hilfe der RWTH- Mindstorms-NXT-Toolbox könnten die Studenten anfangen, NXT-Roboter zu programmieren.

Also diese Toolbox ermöglicht die Kommunikation zwischen MATLAB und dem NXT-Roboter über eine Bluetooth-Verbindung oder direkt durch USB-B Kabel (siehe Abbildung 4).

Abbildung 4: NXT-Gerät durch USB-B Kabel mit PC verbinden

C. Programmablauf

(Um deutlicher zu verstehen, wie der Code funktioniert, siehe Abbildung 5). Das Programm wurde mit while und if Schleifen erstellt. Es begann mit der Definition des NXT-Geräts über das RWTH Mindstorms NXT Toolbox. Dann wurden die Motoren und Sensoren nacheinander definiert. Für jeden Motor wurden eine bestimmte Geschwindigkeit, Farbe und Entfernung entsprechend der erforderlichen Aufgabe eingestellt, sodass der Roboter alle erforderlichen Aufgaben konfliktfrei korrekt ausführen konnte.

D. Problem

Die in Lego verwendeten Sensoren und Motoren haben im Vergleich zu den Sensoren in richtigen Autos eingeschränkte Funktionen.
Beispielsweise kann sich der Motor nicht nach rechts oder links drehen, sondern nur vorwärts und rückwärts. Um dieses Problem zu lösen, müssen zwei Motoren verwendet und zwischen ihnen synchronisiert werden, indem die Geschwindigkeit des ersten verringert und die Geschwindigkeit des zweiten erhöht wird, um sich zu drehen.
Zu den Problemen mit den Sensoren gehört zum Beispiel: Beim Farbsensor erkennt er die Farbe nur aus nächster Nähe, was nicht dem Design echter Ampeln entspricht, die immerhin etwa 3 Meter von dem Auto entfernt sind. Also Die Form des Designs musste geändert werden, sei es für das Auto oder auch für die Ampel (siehe Abbildung 7). Außerdem ist der Abstand des Ultraschallsensors begrenzt und gibt sogar ungenaue Zahlen aus, was eine Neuerstellung des Codes gemäß dem, was für diesen Sensor geeignet ist, erforderlich macht.

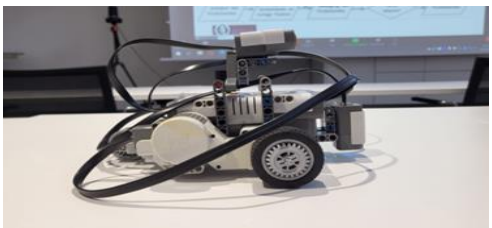


Abbildung 5: Seitliche Sicht auf das Smart Car

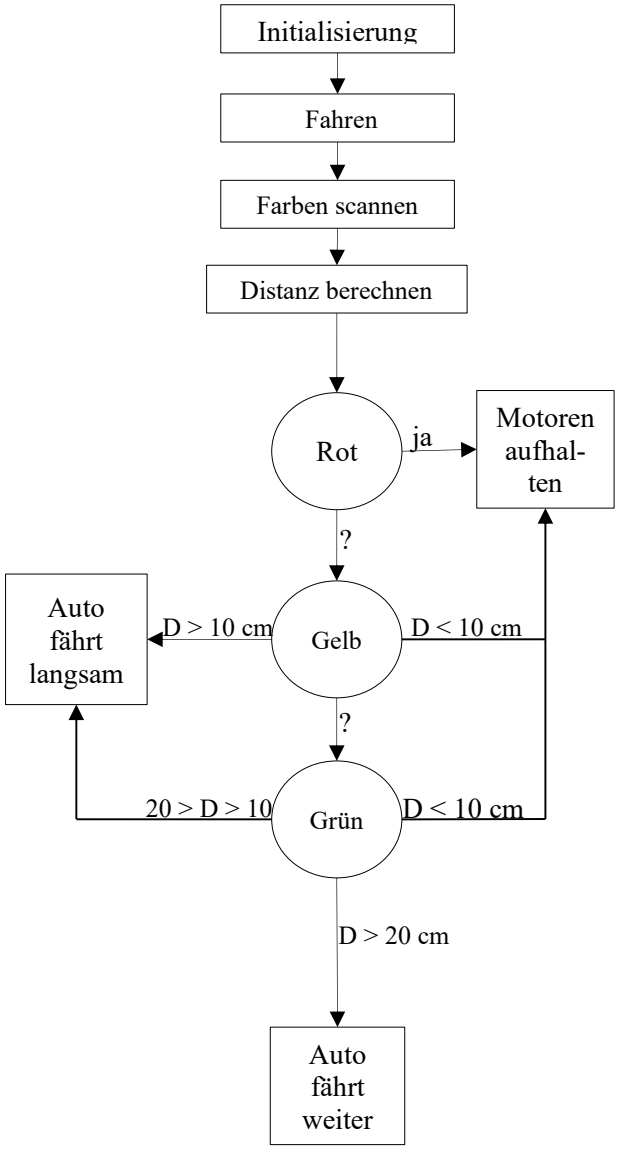


Abbildung 6: Programmablaufplan

IV. ERGEBNISDISKUSSION

Das angestrebte Ziel ist erreicht. Die Ergebnisse sahen positiv aus. Somit wurde das Ziel des Praktikums erreicht. Das Auto funktioniert einwandfrei. Die Motoren und Sensoren sind miteinander synchronisiert. Das heißt, eine Anfangsform des sogenannten Smart Cars ist erreicht. Es besteht kein Zweifel, dass es möglich war, einige andere Sensoren hinzuzufügen, da bekannt ist, dass intelligente Autos reich an nützlichen Sensoren sind. Zu den Sensoren, die hinzugefügt werden könnten, gehört eine Kamera zur Erkennung von Verkehrszeichen, die aber schwierig zu programmieren war und die Zeit dabei nicht half.

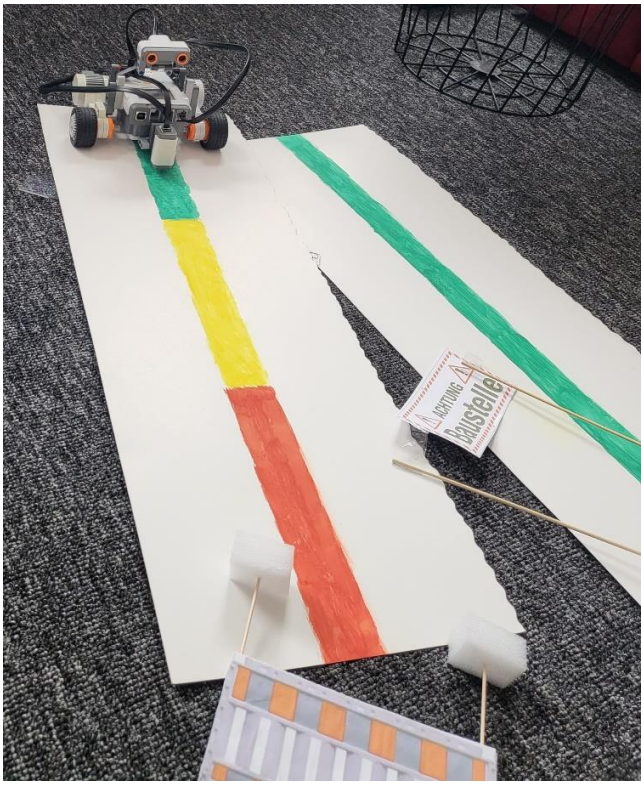


Abbildung 7: Ersatz zur Ampel

V. ZUSAMMENFASSUNG UND FAZIT

Am Ende war nach Anstrengung die endgültige Form des Autos erreicht, die ihre Aufgabe perfekt erfüllte. Keine Frage, die Zeit war knapp, zumal innerhalb von Tagen eine neue Sprache erlernt werden musste, dann eine Idee entstand, ein Roboter installiert und programmiert wurde. Natürlich hätte das Auto mit mehr Sensoren und anderen Funktionen entwickelt werden können, aber je nach verfügbarer Zeit ist dies der maximal mögliche Fortschritt. Das Ziel dieses Projekts war es, die Sprache MATLAB zu lernen und praktisch anzuwenden, indem ein Roboter gebaut wurde, und dieses Ziel wurde tatsächlich vollständig erreicht.

LITERATURVERZEICHNIS

[1] Destatis: Verkehrsunfälle (https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Verkehrsunfaelle/_inhalt.html) Version: Januar 2023

[2] KC Cheung in Algorithm-XLAB (https://algorithm-mxlab.com/blog/worlds-top-33-companies-working-on-self-driving-cars/) Version: 23.10.2020

Smart Car

Mhd Esmail Omar, Elektro- und Informationstechnik

Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*— Das Projektseminar Elektrotechnik und Informationstechnik findet jährlich an der Otto-von-Guericke-Universität Magdeburg statt. Im Rahmen des diesjährigen Projektseminars wurde ein Auto entwickelt. Dieses trägt dazu bei, die Zahl der Verkehrsunfälle zu reduzieren. Die Entwicklung und Konstruktion des Autos erfolgte auf Basis von LEGO Mindstorms-Sets und dem LEGO NXT-Steuerungscomputer. Die softwaretechnische Umsetzung wurde mit MATLAB realisiert. In diesem Papier werden der Aufbau und die Funktionsweise des Autos vorgestellt. Des Weiteren werden einige Herausforderungen während des Konstruktionsprozesses sowie deren Lösungsansätze diskutiert.**

***Schlagwörter*— Smart Car, MATLAB, LEGO Mindstorms, Farbsensor, Ultraschallsensor.**

I. EINLEITUNG

Menschliches Fehlverhalten ist die häufigste Unfallursache im Straßenverkehr, da der Fahrer oder die Fahrerin das Fahrzeug lenkt und somit die Kontrolle über das Fahrzeug hat. Ein Fehler oder eine Unaufmerksamkeit des Fahrers oder der Fahrerin kann schnell zu einer gefährlichen Situation und zu einem Unfall führen. Häufige menschliche Fehler, die zu Verkehrsunfällen führen, sind z.B. Ablenkung (z.B. durch ein Handy), überhöhte Geschwindigkeit, Alkohol- oder Drogenkonsum, falsches Überholen, Ignorieren von Verkehrszeichen oder Missachtung der Vorfahrt. Obwohl es auch andere Ursachen für Verkehrsunfälle gibt (z. B. technische Mängel am Fahrzeug oder schlechte Straßenverhältnisse), bleibt menschliches Fehlverhalten die Hauptursache für Verkehrsunfälle [1]. Um das Risiko von Verkehrsunfällen zu minimieren, ist es daher sinnvoll, Autos mit modernen Technologien wie Sensoren auszustatten, um Fehler des Fahrers zu vermeiden, bevor sie passieren.

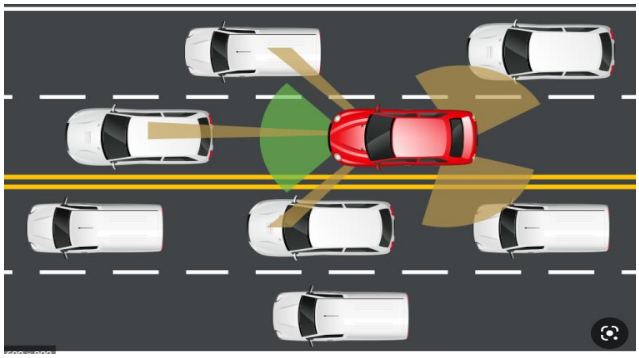


Abbildung 2: Auto mit Sensoren

DOI: 10.24352/UB.OVGU-2023-024

Abbildung 1: Ablenkung beim Fahren

II. VORBETRACHTUNGEN

A. Sensoren

Mit Sensoren ausgestattete Fahrzeuge sind heute immer häufiger im Straßenverkehr anzutreffen. Mit Hilfe von Sensoren und anderen fortschrittlichen Technologien können moderne Fahrzeuge ihre Umgebung erfassen und auf Veränderungen in der Umgebung, wie z. B. andere Fahrzeuge oder Hindernisse, reagieren. Mit Hilfe von Kameras, Radarsystemen und Lidar-Technologie können Autos ihre Umgebung in Echtzeit erfassen und so eine schnellere Reaktionszeit gewährleisten. Der Einsatz von Sensoren kann auch dazu beitragen, Unfälle zu vermeiden, indem sie den Fahrer vor potenziellen Gefahren warnen und in einigen Fällen sogar automatisch eingreifen, um Kollisionen zu verhindern. Insgesamt ermöglichen mit Sensoren ausgestattete Fahrzeuge dem Fahrer ein sichereres und effizienteres Fahren.

B. LEGO-Mindstroms-Set

Das LEGO-Mindstroms-Set ist eine Kombination aus Bausteinen und Elektronik. Es enthält eine programmierbare Steuereinheit (den NXT-Baustein), drei Motoren, verschiedene Sensoren (Licht-, Tast-, Ultraschall- und Farbsensoren) und eine Vielzahl von LEGO-Steinen zum Bau von Robotern [2].

III. KONSTRUKTION UND REALISIERUNG

A. Aufbau

Um ein Auto zu bauen, das schneller reagiert und bessere Entscheidungen trifft als ein menschlicher Fahrer, werden zwei Motoren, ein Farbsensor, ein Ultraschallsensor, NXT-Gerät und LEGO-Steine benötigt.

- *Farbsensor*

Der Farbsensor scannt die Farbe einer Oberfläche und gibt einen RGB-Wert zurück, welcher möglichst die Farbe der Oberfläche darstellt. Damit können die Farben einer Ampel erkannt werden.

- *Ultraschallsensor*

Der Ultraschallsensor eignet sich zur Messung von Entfernungen und Abständen zu anderen Objekten. Dazu wird ein Ultraschallsignal ausgesendet und dessen Echo empfangen. Um den Abstand zum Objekt messen zu können, muss die Laufzeit des Signals gemessen und daraus der Abstand berechnet werden. Auf diese Weise wird ein Hindernis erkannt.

- *Motoren*

Beide Motoren verrichten die gleiche Arbeit. Sie machen genau die vorgegebene Anzahl von Umdrehungen. Damit das Auto fährt.

B. Programmierung

Der NXT ist ein Steuerungscomputer. Er besitzt Anschlüsse für mehrere Sensoren und Motoren sowie USB- und Bluetooth-Schnittstellen. Die Steuerung des NXT-Gerätes und der Sensoren/Motoren erfolgt über MATLAB mit Hilfe der RWTH-Mindstorms-NXT Toolbox. Damit können die Werte der Sensoren ausgelesen und die Motoren angesteuert werden.
Durch die Kennzeichnung der Anschlüsse mit Buchstaben (für Motoren) und Nummern (für Sensoren) können diese elektronischen Elemente definiert werden. Die RWTH Toolbox stellt verschiedene Befehle zur Verfügung. Mit diesen werden die Sensoren und Motoren in Betrieb genommen. Beispiele sind „`OpenNXT2Color`“ zur Farberkennung mit dem Farbsensor oder „`GetUltrasonic`“ zur Abstandsberechnung mit dem Ultraschallsensor [3]. MATLAB ist eine Plattform für Programmierung und numerische Berechnungen. Ähnlich wie andere Programmiersprachen verfügt sie über Schleifen wie z.B. While-Schleife, die es erlauben, eine Folge von Anweisungen wiederholt auszuführen, ohne die entsprechenden Anweisungen mehrmals schreiben zu müssen. Sie verfügt auch über Anweisungen wie die if-Anweisung, mit der bestimmte Teile eines Programms nur dann ausgeführt werden, wenn bestimmte Bedingungen erfüllt sind.
Die NXT-Einheit empfängt die Befehle vom Computer und leitet sie an die Motoren weiter. Sie fungiert als Übersetzer zwischen dem in MATLAB programmierten Skript und dem Roboter.

C. Programmablauf

Das Programm wird durch Klicken auf Run in MATLAB gestartet. Zuerst beginnt der Farbsensor mit der Farberkennung und der Ultraschallsensor mit der Abstandsmessung. Durch die if-Anweisung (siehe Abbildung 4) werden die Motoren nur gedreht, wenn die erkannte Farbe Grün ist und gleichzeitig der Abstand zwischen dem Auto und dem Objekt, das sich vor ihm befindet, größer als 20 cm ist (siehe Abbildung 5).

Abbildung 3: Aufbau des Autos

Analog, wenn die Farbe Gelb ist und der Abstand mehr als 10 cm beträgt, fährt das Fahrzeug langsam, und wenn die Farbe jedoch Rot ist, muss der Motor gestoppt werden.

Anhang

```
while 1
   color = GetNXT2Color(port,handle);
   distance = GetUltrasonic(SENSOR_2);
   if strcmp(color,"GREEN") && (distance > 20)
        motorA.Power=50;
        motorA.SendToNXT(handle);
        motorB.Power=50;
        motorB.SendToNXT(handle);
```

Abbildung 4: Kurzer Ausschnitt des Quelltextes

D. Problem

Wie jeder weiß, sind Ampeln mindestens 3 Meter von Autos entfernt, manchmal sogar 100 Meter. Leider kann der Farbsensor des LEGO Bausatzes Farben nur aus sehr kurzer Entfernung erkennen. Aus diesem Grund wurde er an der Vorderseite befestigt und nach unten gerichtet, wo eine Linie mit drei Farben (Rot, Gelb, Grün) gezeichnet ist. Diese Linie dient als Ersatz für die Ampel (siehe Abbildung 6).

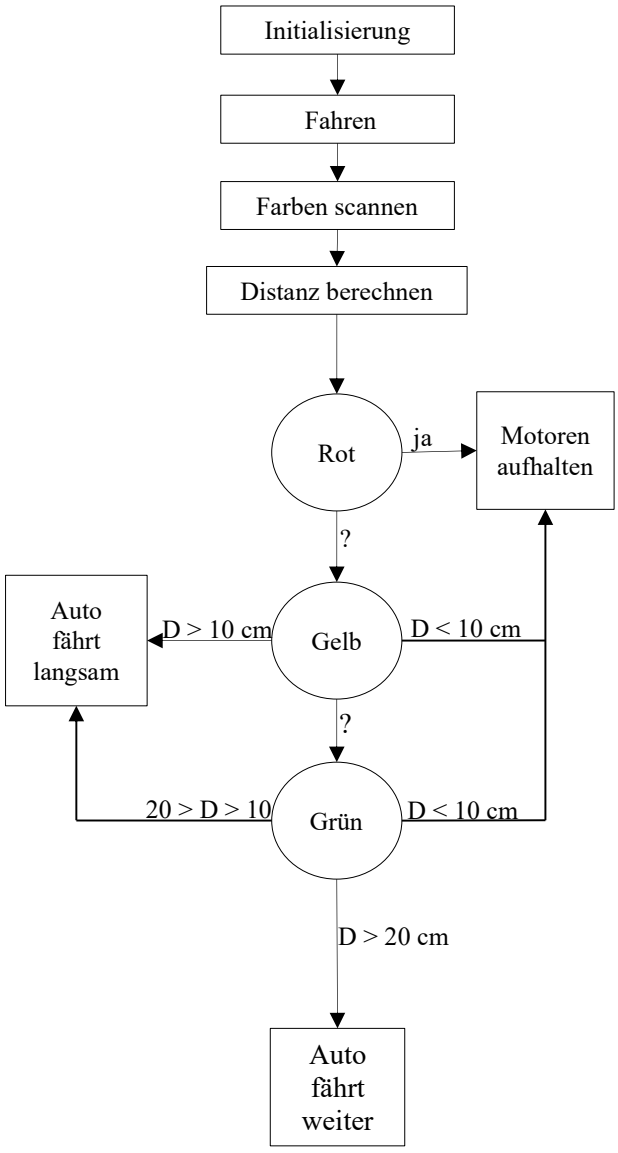


Abbildung 5: Programmablaufplan

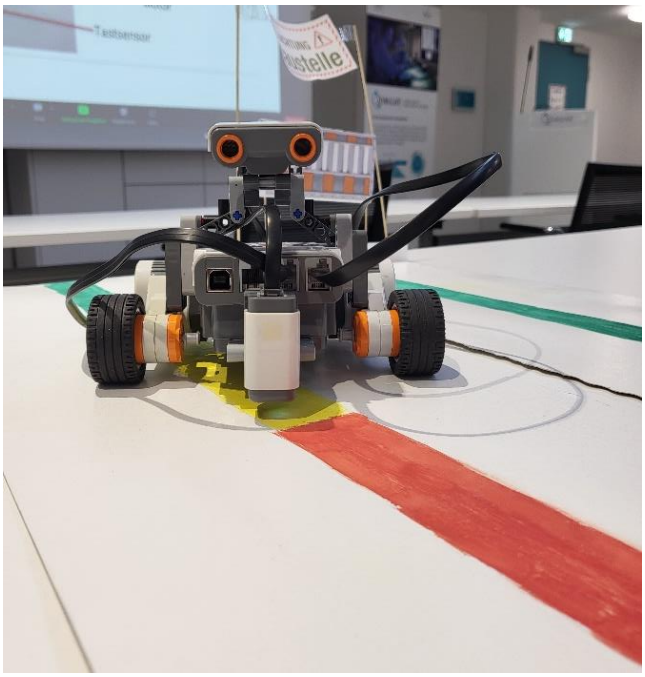


Abbildung 6: Ersatz für Ampel

IV. ERGEBNISDISKUSSION

Nach zwei Wochen Arbeit wurde ein smartes Auto gebaut, das vor einem Hindernis oder einer roten Ampel bremst. Das Ziel des Praktikums wurde erreicht. Das Auto trägt dazu bei, den Straßenverkehr sicherer zu machen. Es könnte noch mit einer Kamera und einem Tonsensor ausgestattet werden. Damit kann das Auto Verkehrsschilder und Alarmsignale von Rettungs- und Feuerwehrautos erkennen. So kann das Auto noch mehr zur Verkehrssicherheit beitragen.

V. ZUSAMMENFASSUNG UND FAZIT

Zusammenfassend kann festgestellt werden, dass die Ziel des Projekts zufriedenstellend erreicht wurde. Hätte das Projektseminar länger als zwei Wochen gedauert, wäre es möglich gewesen, komplexere Maschinen zu bauen. Abschließend kann man sagen, dass wir, obwohl das Auto wie ein Spielzeug aussieht, in diesem Projekt viel über selbstfahrende Autos gelernt haben. Außerdem ist zu erwähnen, dass das LEGO Mindstorm-Projektseminar eine gute Möglichkeit ist, Studierende mit der Programmierung in MATLAB vertraut zu machen.

LITERATURVERZEICHNIS

[1] Marie Maier: Menschliches Versagen verursacht die meisten Verkehrsunfälle (https://www.fahrschule-online.de/nachrichten/menschliches-versagen-verursacht-die-meisten-verkehrsunfaelle-2911045) Version: Juli 2021

[2] WIKIPEDIA, THE FREE ENCYCLOPEDIA: LEGO Mindstorms NXT. (https://en.wikipedia.org/wiki/Lego_Mindstorms_NXT) Version: November 2022

[3] Alexander Behrens (2023). RWTH - Mindstorms NXT Toolbox (https://www.mathworks.com/matlabcentral/fileexchange/18646-rwth-mindstorms-nxt-toolbox) Version: Februar 2023.

Farbsortierroboter für Socken

Kilian Borde, Elektromobilität
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Die folgende Arbeit beschäftigt sich mit der Planung, Konstruktion und Programmierung eines Farbsortierroboters, der mithilfe des LEGO Mindstorms EV3 gebaut wurde. Dieser soll in der Lage sein, die Farbe von Socken zu erkennen und diese zu einer entsprechenden Ablagefläche zu transportieren, sodass sie farbrein sortiert werden. Dazu werden verschiedene Sensoren zur Farberkennung und Programmausführung sowie Motoren zum Greifen und Bewegen benötigt. Die Programmierung erfolgt mit MATLAB.**

***Schlagwörter*—LEGO Mindstorms, EV3, Projektseminar, Farbsortierroboter, Socken**

I. Einleitung

DIE Automatisierung schreitet immer weiter voran und hält in allen Bereichen Einzug. Besonders in der Industrie findet sie viele Anwendungen, wodurch Personal und Kosten eingespart werden können. Sie kann aber auch den Menschen unterstützen und somit eine Erleichterung im Alltag darstellen. Im Rahmen des Projektseminars wurde mithilfe eines LEGO-Mindstorms-Sets ein Roboter entwickelt, der den Menschen beim Sortieren von Strumpfwaren unterstützen kann. Außerdem kann er eine Hilfe für Menschen mit Sehbehinderung sein.

II. Vorbetrachtungen

A. Farbsortierung

”Farbe ist ein subjektiver Eindruck. Er entsteht, wenn unser Sehsystem den Sinnesreiz verarbeitet, der durch Licht verschiedener Wellenlängen auf der Netzhaut des Auges ausgelöst wird.” [1] Farben haben einen starken Einfluss auf das menschliche Gehirn. So können Farben wie Grün oder Blau eine positive Wirkung auf den Menschen haben und zu Entspannung führen. Rot hingegen kann bei längerer Betrachtung zu Aggression und Wut führen [2]. Auch in der Industrie ist die Sortierung nach Farben von großer Bedeutung. Zum Beispiel in der Kunststoffherstellung, wo die Aussortierung verschiedenfarbiger Granulatpartikel sehr wichtig ist, um besonders reine Ergebnisse zu erzielen. Das Projekt sollte die grundsätzliche Funktion eines industriellen Farbsortierers, zu sehen in Abbildung 1, nachbilden und dabei eine realistische Aufgabe bewältigen.

B. Zielsetzung

Ziel des Projektes war es, einen automatisierten Farbsortierroboter zu bauen. Um die Aufgabe zielgerichteter lösen zu können, wurde die Sortierung eines bestimmten Gegenstandes gewählt. Aufgrund des Formfaktors wurden Socken ausgewählt. Das Ziel des Farbsortierroboters für Socken ist also die fehlerfreie Erkennung der Sockenfarbe und die darauf folgende korrekte Ansteuerung der richtigen Ablagefläche. Darüber hinaus soll der Roboter in der Lage sein, diesen Vorgang beliebig oft zu wiederholen, ohne dass sich auftretende Fehler summieren und zu einem Ausfall der Automatisierung führen.

DOI: 10.24352/UB.OVGU-2023-025

Abbildung 1. Industrieller Farbsortierer [3]

C. Umsetzungsplan

Um das Projekt durchführen zu können, wurden im Vorfeld umfangreiche Recherchen und Programmierübungen mit dem Programm MATLAB durchgeführt. Dabei wurden auch die Kenntnisse über die Motoren und Sensoren des LEGO Mindstorms vertieft. Das zugrundeliegende Ziel war es, einen Roboter zu bauen, der die Anforderungen der Farberkennung und Bewegung erfüllt. Der Roboter sollte in der Lage sein, die Farbe von Socken zu erkennen, zum richtigen Ablageplatz zu fahren und die Socken dort abzulegen. Die Programmierung sollte sicherstellen, dass dieser Vorgang beliebig oft wiederholt werden kann. Zunächst beschränkte man sich auf die Auswahl von drei bis vier Farben, um den Aufwand gering zu halten, aber dennoch die Funktionsfähigkeit demonstrieren zu können.

III. Programmierung und Konstruktion

A. Programmierung der Farberkennung

Da bei der Entwicklung des Farbsortierroboters für Socken die richtige Programmierung von großer Bedeutung war, wurde zuerst das Programm geschrieben und dann die Konstruktion entsprechend angepasst. Besonderes Augenmerk wurde dabei auf die korrekte Ansteuerung des LEGO Mindstorms EV3 und das korrekte Einlesen der Sensordaten gelegt. Außerdem war es wichtig, sowohl die von MATLAB gespeicherten Werte und Parameter als auch die Rotationszustände der Motoren

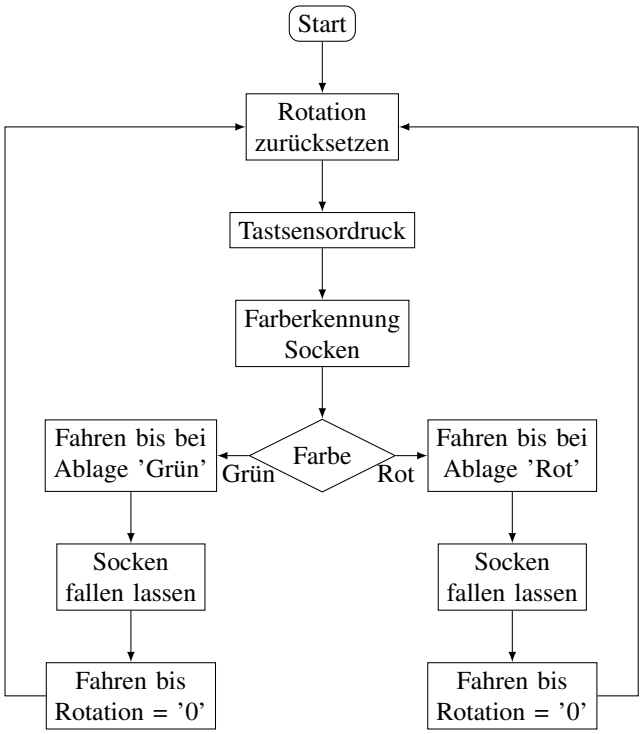


Abbildung 2. Programmablaufplan des Programmes für den Farbsortierroboter

bei jedem Programmstart zurückzusetzen. Um theoretisch zu gewährleisten, dass die Funktion der Sockensortierung unendlich oft wiederholt werden kann, wurde eine While-Schleife verwendet, die die gesamte Sortieraufgabe umschließt. Die Ausführung der Farberkennung mittels Farbsensor beginnt mit dem Druck auf den zuvor initialisierten Tastsensor. Anschließend speichert das Programm die erkannte Farbe, woraufhin mittels einer If-Abfrage an die entsprechende Stelle im Programm gesprungen wird, welche die Ansteuerung der richtigen Ablagefläche ausführt.

B. Programmierung der Sockenablage

Anschließend fährt der Roboter linear zur richtigen Ablagefläche. Diese erkennt er mithilfe eines zweiten Farbsensors. Ist er an der richtigen Ablagefläche angekommen, stoppt er und der Greifarm öffnet sich. Dies wurde durch eine While-Schleife realisiert, die den Motor so lange bewegt, bis die Dreherkennung im Motor einen entsprechenden Wert erreicht hat. Danach fährt der Roboter linear in seine Ausgangsposition zurück. Dies wird ebenfalls durch eine While-Schleife gelöst, die so lange ausgeführt wird, bis der Rotationssensor im Motor den Wert '0' erreicht. Danach wird die gesamte While-Schleife zur unendlichen Wiederholung erneut gestartet, sodass das Programm durch erneutes Drücken des Taktsensors wieder gestartet werden kann. Das Programm ist in Abbildung 2 anhand eines Programmablaufplans grafisch dargestellt.

C. Montage der Motoren

Die Konstruktion des Farbsortierroboters für Socken wurde an die Programmierung angepasst. Es wurde darauf geachtet, den Roboter so kompakt wie möglich zu bauen. Um eventuelle Abweichungen in der Drehung des Motors zu vermeiden, wurde nur ein Motor für die Bewegung des Roboters verwendet, was möglich ist, da der Roboter sich nur linear vor- und zurückbewegen muss. Der Tastsensor für den Programmstart wurde an der Seite des Roboters angebracht und mit einer größeren Auflagefläche versehen, um einen besseren Druck zu erzielen. Ursprüngliche Pläne, den Farbsortierroboter mit einem von zwei Motoren gesteuerten Greifarm auszustatten, wurden wegen Gleichgewichtsproblemen verworfen. Dieser Greifarm ist in Abbildung 3 dargestellt. In der endgültigen Konstruktion wurde der Greifarm durch einen einfachen Greifer ersetzt, auf dem die Socken abgelegt werden können, woraufhin sie nur noch fallen gelassen werden müssen.

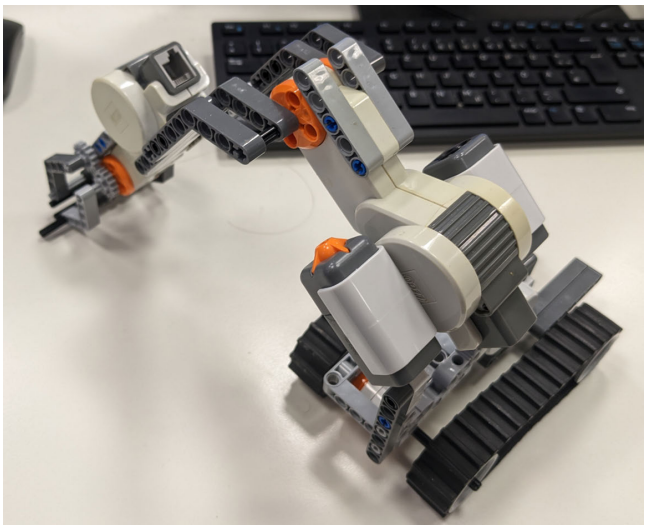


Abbildung 3. Greifarm mit Gleichgewichtsproblemen

D. Montage der Farbsensoren

Außerdem wurden zwei Farbsensoren benötigt, einer zur Erkennung der Sockenfarbe und ein zweiter zur Erkennung der Ablagefläche. Die Ablageflächen sind in Abbildung 4 dargestellt. Socken, deren Farbe der Farbsensor nicht eindeutig erkennen kann, werden auf der gelben Fläche abgelegt. Da die Farbsensoren nur auf sehr kurze Distanzen (ca. $0{,}1\,\mathrm{cm}$ bis $0{,}5\,\mathrm{cm}$) zuverlässige Farbwerte liefern, mussten sie sehr nahe am zu erkennenden Objekt angebracht werden. Aus diesem Grund wurde der Farbsensor zur Farberkennung der Socken direkt unter dem Greifer montiert. Ein weiterer Vorteil dieser Konstruktion ist, dass der Farbsensor beim Herunterdrücken der Socken unterstützend wirkt. Die fertige Konstruktion ist in Abbildung 5 zu sehen.

IV. Ergebnisdiskussion

Am Ende des Projektes konnte ein weitgehend funktionsfähiger Farbsortierroboter für Socken präsentiert werden. Der angestrebte Funktionsumfang wurde fast vollständig erreicht, allerdings gibt es noch einige Probleme, die bei einer längeren Bearbeitungszeit hätten gelöst werden können. Zum einen ist die Zuverlässigkeit der Farberkennung teilweise

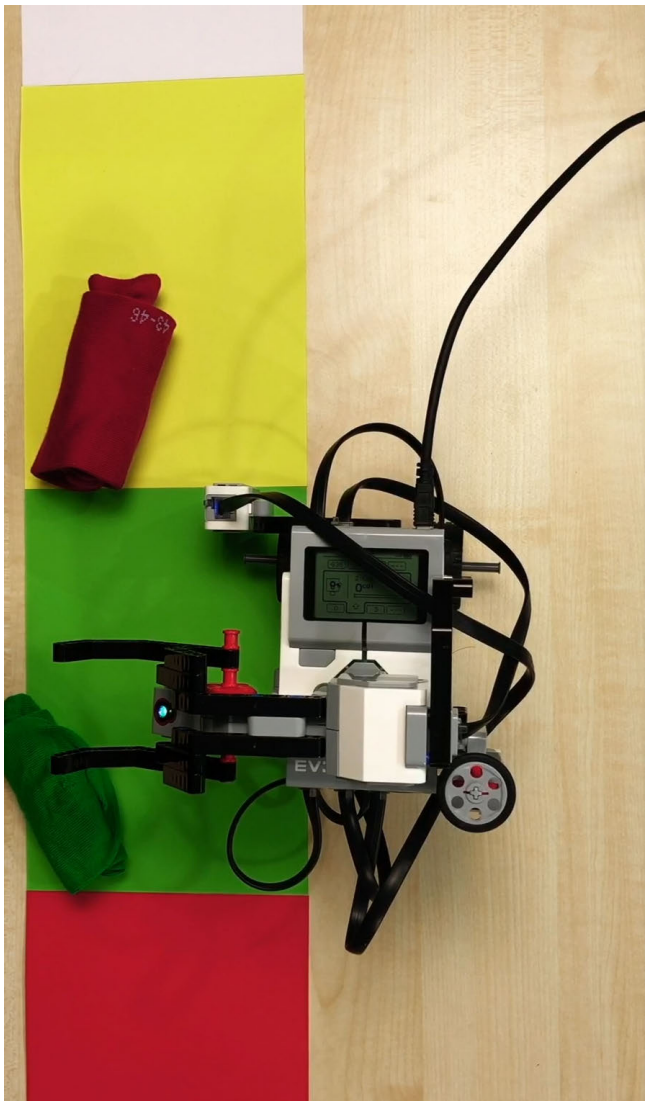


Abbildung 4. Farbsortierroboter mit Ablagefläche [4]

eingeschränkt. So kann der Roboter aufgrund der Nähe des ersten Farbsensors zum Boden zwar die exakte Ablagefläche anfahren, jedoch wird die Farbe der Socken nicht immer korrekt erkannt. Dies liegt vor allem an der Entfernung des Farbsensors zu den Socken. In einer zukünftigen Version des Roboters könnte dies durch bessere Sensoren oder durch das Einlesen der Farben mithilfe einer Webcam behoben werden. Außerdem können nur einfarbige Socken sortiert werden. Des Weiteren ist es in der aktuellen Version des Roboters nur möglich, eine begrenzte Anzahl von Farben zu sortieren. Diese sind: Blau, Rot und Grün. Alle nicht erkannten Farben werden auf einem weiteren gelben Lagerplatz abgelegt. Die Anzahl der Ablageflächen könnte jedoch in Zukunft beliebig skaliert werden. Ein weiteres Problem ist, dass der Roboter aufgrund der Schwerpunktverlagerung in Richtung Greifer nicht perfekt geradeaus fährt. Dies könnte in einer zukünftigen Version durch eine Schiene gelöst werden, auf der sich der Roboter geradeaus und rückwärts bewegt. Schließlich gibt es noch die Einschränkung, dass der Roboter nur einzelne Socken

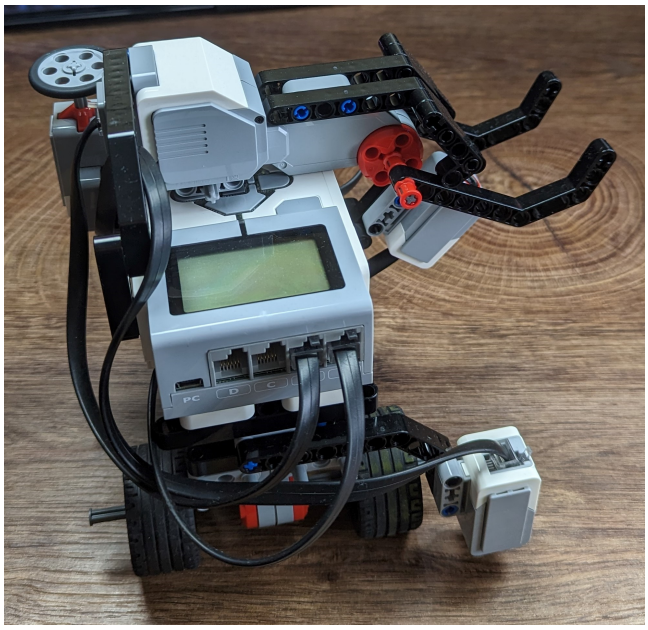


Abbildung 5. Farbsortierroboter mit reduziertem Greifer

einsortieren kann, da der Greifer nur eine kleine Auflagefläche hat. Dieses Problem kann leicht gelöst werden, indem in Zukunft ein größerer Greifer gebaut wird.

V. Zusammenfassung und Fazit

Während des Projektseminars wurde ein Roboter entwickelt, der Strumpfwaren nach Farben sortieren kann. Diese Funktion wurde grundlegend erfüllt, sodass ein funktionierender Roboter präsentiert werden konnte. Dabei wurde viel über die Programmierung mit MATLAB gelernt und Kenntnisse über die Konstruktion konnten erworben und gefestigt werden. Unter idealen Bedingungen ist der Roboter in der Lage, die Farbe von Socken zu erkennen, diese auf eine zugewiesene Ablagefläche zu befördern und diesen Vorgang zu wiederholen. Mit leichten Anpassungen könnte der Roboter in einer zukünftigen Version jedoch noch in Bezug auf Zuverlässigkeit und Wiederholbarkeit verbessert werden.

Literaturverzeichnis

[1] *Was ist Farbe und wie entsteht sie?* Wissenschaft im Dialog, 2023.02.21 https://www.wissenschaft-im-dialog.de/projekte/wieso/artikel/beitrag/was-ist-farbe-und-wie-entsteht-sie-warum-ist-beispielsweise-eine-tomate-rot/

[2] *Einrichten in Rot: Was muss ich beachten?* Pharao24.de, 2023.02.21 https://www.pharao24.de/magazin/einrichten-in-rot-was-muss-ich-beachten/

[3] *Ehexianwiki*. CC BY-SA 4.0, 2023.02.22 https://commons.wikimedia.org/wiki/File:Color_Sorter.jpg?uselang=de

[4] *LEGO-Praktikum 2023 Abschlusspräsentationen live aus dem Speicher B*. mathiasmagdowski, 2023.02.23 https://www.twitch.tv/mathiasmagdowski

Farbsortierroboter für Socken

Anh Minh Nguyen, , Elektrotechnik und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

Das Seminarprojekt der Fakultät für Elektrotechnik und Informationstechnik der Otto-von-Guericke-Universität Magdeburg findet im Wintersemester eines jeden Studienjahres statt. An diesem Projekt müssen Studierende teilnehmen, die jeweils eine Projektgruppe aus zwei Studierenden wählen. Jede Gruppe findet ihr eigenes Projekt und muss dieses vor den Dozenten und anderen Studenten präsentieren. Mein Team wird an einem Projekt zur Farbkorrektur der Farben Rot, Gelb und Blau mit Hilfe eines LEGO-MINDSTORMS-EV3 arbeiten.

Schlagwörter— Farbsortierroboter, LEGO-EV3, Otto-von-Guericke Universität

1. EINLEITUNG

Im modernen Leben setzen die Menschen Roboter für viele Zwecke ein, z. B. in der Produktion in Fabriken und bei alltäglichen Tätigkeiten. Wie alles im Leben hat auch die Robotik ihre Vor- und Nachteile, die den Menschen helfen oder schaden können. Aber man kann sagen, dass der Mensch ohne Roboter nicht leben kann, weil sie Zeit, Geld und Mühe sparen und die Arbeitsproduktivität erheblich steigern. Und diese Projektgruppe hat unter anderem die Idee eines Sortierroboters. Roboter werden den Menschen helfen, die Produktivität zu verbessern, zum Beispiel: Roboter sortieren verdorbenes Obst nach Farbe. Das Ziel dieser Präsentation ist also die Farbkorrektur.

2. AUFBAU UND FUNKTIONEN

Dieser Abschnitt beschreibt den Aufbau und die Funktionsweise.

2.1 Aufbau

2.1.1 Motoren und Sensoren

Grundsätzlich verwendet der Roboter 2 Motoren, 2 Farbsensoren, Knopfsensoren, LEGO-EV3 und LEGO-Steine.

Motor (d) wird verwendet, um sich vorwärts und rückwärts zum Startpunkt zu bewegen, wie in Abbildung 1 gezeigt.
Der Motor (e) wird verwendet, um die Socke auf die vom Farbsensor bestimmte Farbposition abzusenken. Wie in Abbildung 1 gezeigt.
Der Farbsensor (a) wird verwendet, um die Farbe der Socken am LEGO-Arm zu bestimmen. Wie in Abbildung 1 gezeigt.
Der Berührungssensor (c) ist für die Inbetriebnahme des LEGO-EV3 verantwortlich. Wie in Abbildung 1 gezeigt.
Farbsensor (b): Dieser Sensor ist zuständig, nachdem der Farbsensor (a) die Farben der Socken (rot, gelb und grün) erkannt hat. Dieser Sensor bestimmt die Farben der Socke im Ablagesplatz auf die von Sensor (a) identifizierten Farben. Der Vorteil des Farbsensors (b) besteht darin, dass er die Socken entsprechend unseren Anforderungen auf den richtigen Ablageplatz legt.

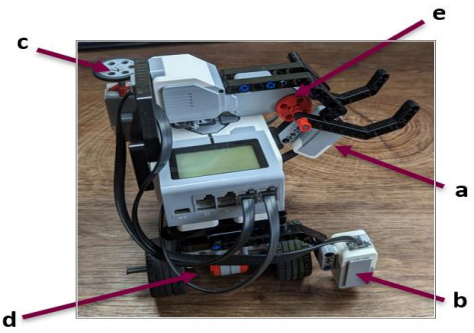


Abbildung 1 : LEGO-Mindstroms-EV3

Abbildung 2 : Farbsensor

Abbildung 3 : Berührungssensor

2.1.2 EV3-Gerät

EV3 ist das wichtigste und schwierigste Teil des Projekts, es ent-scheidet über Erfolg oder Misserfolg. Wir können EV3 das Gehirn von LEGO nennen, es verbindet die Sensoren über Kabel mit dem Motor.

2.2 Programmablauf

DOI: 10.24352/UB.OVGU-2023-026

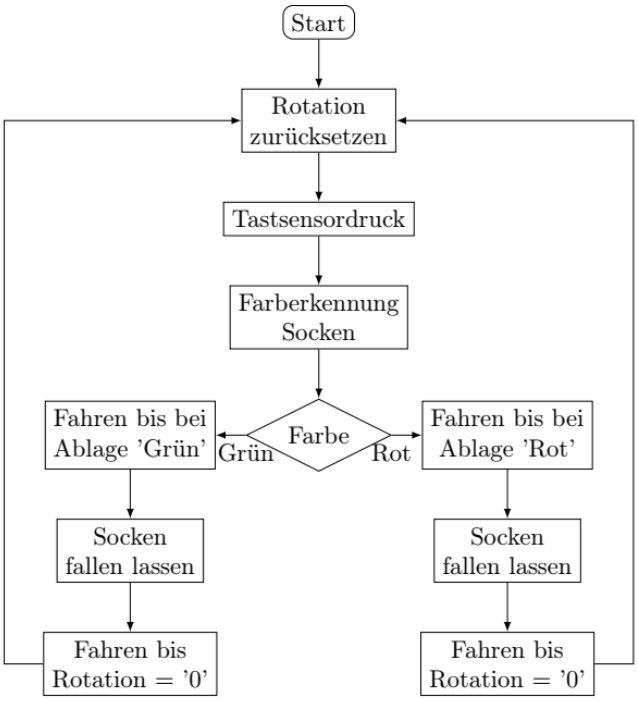


Abbildung 4 : Programmablauf

Um LEGO zum Laufen zu bringen, wird das Programm durch Drücken der Start-Taste im MATLAB-Programm gestartet. Nach ca. 5 bis 6 Sekunden drücken wir den Berührungssensor und der LEGO-EV3 Roboter beginnt mit dem Sortieren der Socken, basierend auf dem in MATLAB geschriebenen Code. Der Roboter kann nur 3 Grundfarben unterscheiden: Rot, Blau und Grün. In diesem Projekt werden die Socken auf den Roboterarm gelegt, dann erkennt der Farbsensor unter dem Arm die Farbe der Socke. Als nächstes bewegt sich der Bewegungsmotor (d) zu den Ablageplatz mit den 3 Farben rot, blau und grün. Wenn sich der Roboter bewegt, erkennt der Farbsensor (b) gleichzeitig die Farben dieser 3 Ablageplatze, dieser Sensor unterscheidet auch nur die 3 Farben rot, blau und blau. Wenn der Farbsensor (b) die vom Farbsensor (a) erhaltene Farbe richtig erkannt hat, senkt der Motor (d) des Roboterarms die Socken auf dieses Ablageplatzes. Nachdem der Roboter die Farbe der Socken erkannt hat, fährt der Motor (d) den Roboter in seine Ausgangsposition zurück. Der Vorgang ist beendet, die Socken werden wieder angezogen und es wird wie oben beschrieben mit der Sortierung der farbigen Socken fortgefahren.

Während der Präsentation konnten Sie auch den Roboter in Aktion sehen, der die Farbe von Socken unterscheidet, einschließlich der Probleme des Roboters, auf die später in diesem Artikel eingegangen wird.

3. PROBLEME

Im Folgenden erfahren Sie mehr über die Probleme, die bei der Entwicklung des Farbsortierroboters auftraten, und wie der Roboter überwunden wurde. . Das erste und schwierigste Problem bei der Entwicklung eines Roboters ist das Gewicht des Roboterarms. Bei der Entwicklung des Roboters LEGO-Mindstrom-EV3 wurde versucht, einen Arm zu konstruieren, der farbige Socken (rot, gelb und grün) greifen, halten und heben kann. Dieser Versuch scheiterte jedoch (siehe Abbildung 5), da der Arm zu schwach ist, um z. B. eine lange, schwere Socke anzuheben: Wintersocken, sie sind aus Baumwolle. Und das größte Problem mit dem Gewicht des Roboterarms ist, dass der gesamte Roboter aus dem Gleichgewicht gerät. Während der Entwicklung war der Roboter wirklich unausgewogen, um alle drei beweglichen Motoren bedienen zu können, sie kippten nach vorne. Um den Roboter weiterzuentwickeln, wurde daher ein Motor zum Greifen und Halten entfernt (siehe Abbildung 1). Das Endergebnis des Projekts ist, dass der Roboter nur einen beweglichen Motor verwendet, um die Socken in das richtige Fach der richtigen Farbe fallen zu lassen. Das ist das Problem, auf das das Team gestoßen ist und das es gelöst hat. Das zweite Problem, auf das das Team stieß, war die Farbgenauigkeit des Farbsensors nach Tests, die auf dem in MATLAB geschriebenen Code basierten. Die Gruppe vermutete einen Fehler im in MATLAB geschriebenen Code. Nach Hinweisen von Dozenten und Kommentaren im Internet wurde der Fehler jedoch durch den Farb- und Lichtsensor im Raum verursacht. Der Roboter konnte das Problem überwinden, indem er den Farbsensor näher an den Socken- und Ablageplatz platzierte. Zusätzlich zu den oben genannten Problemen hat der Roboter ein Problem mit der Wiederholung des Prozesses ohne menschliches Eingreifen. Der Roboter kann den Vorgang nicht vollautomatisch wiederholen, es sei denn, das Team zieht die Socke auf den Roboterarm und klickt auf den Berührungssensor. Während des Trainings kippt der Roboter manchmal nach links oder rechts, wenn das Team nicht parallel zur Ablagefläche steht. Das letzte Problem, auf das das Team stieß, war, dass das USB-Kabel zu kurz war, was dazu führte, dass der Roboter nur innerhalb des zulässigen Bereichs arbeiten konnte, der der Länge des Kabels zwischen Laptop und LEGO-Mindstrom-EV3 entspricht.

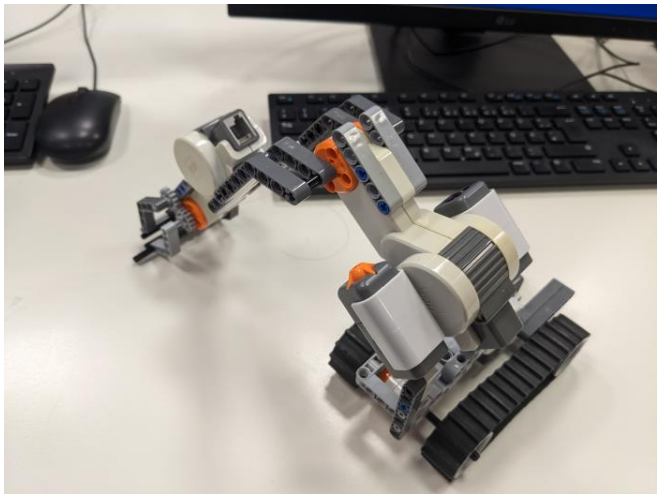


Abbildung 5 : Der erste Aufbau des Roboters

4. ERGEBNIS

Für dieses Projektseminar baute das Team einen LEGO-Roboter, der farbige Socken in farblich passende Schalen sortieren kann. Wie bereits erwähnt, wurde das Roboterprojekt rechtzeitig fertig gestellt. Das Design des Farbsortierroboters muss jedoch verbessert werden. Um effizienter zu arbeiten,

braucht der Roboter mehr Sensoren und bessere Farbsensoren, um die menschlichen Eingriffe zu begrenzen.

5. PRAKTISCHE ANWENDUNG

Farbsortierroboter werden in vielen Bereichen des täglichen Lebens eingesetzt.
Im Folgenden werden einige praktische Anwendungen beschrieben.

- Früchte sortieren: Dank der Farbunterscheidung weiß der Roboter, welche Früchte in Ordnung oder beschädigt sind, z.B.: Eine Reihe von Äpfeln muss durch den Farbunterscheider, die Äpfel werden sofort braun, werden sofort aussortiert und zu organischem Dünger verarbeitet. So wird die Qualität des Obstes verbessert und gleichzeitig die Umwelt geschont.
- Warenklassifizierung: Im Zeitalter der Industrialisierung werden immer mehr Produkte hergestellt und in Kartons verpackt, um die Effizienz zu steigern und die Kosten zu optimieren. Dabei fällt eine große Menge an repetitiver Arbeit an. Mit Hilfe von Farbsortierrobotern können Menschen Produkte in Kartons mit bestimmten Farben verpacken, die dem Produkt entsprechen. Roboter helfen den Menschen, diese Kartons genau zu inventarisieren und ins Lager zu transportieren.

6. ZUSAMMENFASSUNG UND FAZIT

Am Ende des Projektseminars wurde der Farbsortierroboter für den vorgesehenen Einsatzzweck entwickelt. Allerdings weist der Roboter noch die in Teil 3 genannten Probleme auf, so dass er in der Praxis nicht eingesetzt werden kann. Durch das Projektseminar haben die teilnehmenden Studenten jedoch viel Erfahrung in der Arbeit in Gruppen, im Schreiben von Software auf der Basis von MATLAB und, was noch wichtiger ist, in der Anwendung auf andere Arten von Automatisierungsrobotern gesammelt, um die Lebensqualität und Produktivität am Arbeitsplatz zu verbessern.

LITERATUR

[1] DeepL Write : https://www.deepl.com/write
[2] Abbildung 3 des Berührungssensors aus LEGO https://www.lego.com/de-de/product/touch-sensor-9843
[3] Abbildung 2 des Farbsensors aus Legopedia https://lego.fandom.com/de/wiki/NXT_Farbsensor_9694

ANHANG

Fotos und Videos zur Entwicklung des Farbsortierroboters auf Instagram hochgeladen:
https://www.instagram.com/legopraktikumgruppe6/

Nachfolgend wird der Programmcode gezeigt, mit dem der Roboter in Bewegung gesetzt wurde.

```
while 1
   if readTouch(touchsensor) == 1
      resetRotation(motorA);
      resetRotation(motorB);
      motorA.Speed = 0;
      motorB.Speed = 0;
      start(motorA);
      start(motorB);
      %Farbe erkennen
      sockenfarbe = readColor(colorsensor1);
      disp(sockenfarbe);
      pause(2);
      %Greifen
      while readRotation(motorB) > -75
         motorB.Speed = -40;
      end
      motorB.Speed = 0;
      pause(2);
      %zur Ablage fahren
      if strcmp(sockenfarbe,'red')
         %solange bis an der richtigen Ablagefläche fahren
         while readRotation(motorA) > -2000
            motorA.Speed=-20;
            color = readColor(colorsensor2);
            if strcmp(color,'red')
               motorA.Speed = 0;
               disp('bei ROT');
               %Socken ablegen
               while readRotation(motorB) < 0
                  motorB.Speed = 20;
               end
               motorB.Speed = 0;
               pause(2);
               %Zurück Fahren
               while readRotation(motorA) < 0
                  motorA.Speed=20;
               end
               motorA.Speed=0;
               stop(motorA);
               stop(motorB);
               disp('FERTIG!');
               break;
            end
         end
      elseif strcmp(color,'blue')
         %solange bis an der richtigen Ablagefläche fahren
         while readRotation(motorA) > -720
            motorA.Speed=-20;
            color = readColor(colorsensor2);
            if strcmp(color,'blue')
               motorA.Speed = 0;
               disp('bei BLAU');
               while readRotation(motorB) < 0
                  motorB.Speed = 20;
               end
               motorB.Speed = 0;
               pause(2);
               while readRotation(motorA) < 0
                  motorA.Speed=20;
               end
               motorA.Speed=0;
               stop(motorA);
               stop(motorB);
               disp('FERTIG!');
               break;
            end
         end
      elseif strcmp(color,'green')
         %solange bis an der richtigen Ablagefläche fahren
         while readRotation(motorA) > -720
            motorA.Speed=-20;
```

```
            color = readColor(colorsensor2);
            if strcmp(color,'green')
               motorA.Speed = 0;
               disp('bei GRÜN');
               while readRotation(motorB) < 0
                  motorB.Speed = 20;
               end
               motorB.Speed = 0;
               pause(2);
               while readRotation(motorA) < 0
                  motorA.Speed=20;
               end
               motorA.Speed=0;
               stop(motorA);
               stop(motorB);
               disp('FERTIG!');
               break;
            end
         end
      else
         %solange bis an der richtigen Ablagefläche fahren
         while readRotation(motorA) > -720
            motorA.Speed=-20;
            color = readColor(colorsensor2);
            if strcmp(color,'GELB')
               motorA.Speed = 0;
               disp('bei REST');
               while readRotation(motorB) < 0
                  motorB.Speed = 20;
               end
               motorB.Speed = 0;
               pause(2);
               while readRotation(motorA) < 0
                  motorA.Speed=20;
               end
               motorA.Speed=0;
               stop(motorA);
               stop(motorB);
               disp('FERTIG!');
               break;
            end
         end
      end
   end
end
```

Robotik trifft Kunst: Der Lightpainting-Roboter 2.0

Sunny Lou Seidler, Elektromobilität
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*— Im Rahmen des LEGO-Mindstorms-Praktikums 2023 wurde der letztjährige Lightpainting-Roboter „Sally can Draw“ weiterentwickelt. Dieser ist in der Lage, mittels einer Lichtquelle zweidimensionale Formen und Zeichnungen darzustellen. Die dazu verwendeten Konzeptideen sowie deren technische Umsetzungen, Verbesserungen und Probleme werden im Folgenden chronologisch dargestellt.**

***Schlagwörter*— LEGO-Mindstorms, Lightpainting, Roboter, Kunst, Plotter, MATLAB**

I. Einleitung

LIGHTPAINTING [1] ist eine Kunstform, bei der Fotografien in der Nacht oder in abgedunkelten Räumen durch gezielten Einsatz von Licht und Langzeitbelichtung entstehen. Die Künstler verwenden dabei Lichtquellen wie Taschenlampen, LEDs oder Leuchtstäbe, um die Umgebung mit Licht zu bemalen. Da der Fotograf in der Regel viele Versuche benötigt, ein Lightpainting-Bild zu erstellen, wurde bereits 2022 von den Studentinnen Michelle Horn und Anna-Lena Thalhofer, im Rahmen dieses Lego-Projekts ein Prototyp mit dem Namen „*Sally can Draw*“ [2] entwickelt. Dieser sollte neben einer Verkürzung der Gesamtdauer eines Lightpainting-Bildes, auch eine präzisere Linienführung des Lichts ermöglichen, um mit wenigen Versuchen das ideale Bild zu erstellen. Da der Prototyp noch viel Verbesserungspotential hat und das Endergebnis des Bildes nicht optimal aussieht, soll eine 2. Variante des Roboters entwickelt werden. Dieser soll eine stabilere Konstruktion und ein anderes Konzept in der programmtechnischen Umsetzung haben und neben einfachen Formen auch Freihandzeichnungen umsetzen können.

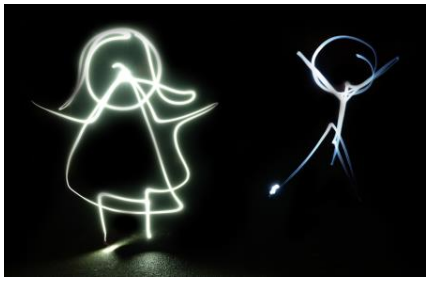


Abbildung 1. Beispielbild zum Lightpainting [3]

II. Vorbetrachtungen

Bevor diese Änderungen am Roboter näher erklärt werden können, sind einige Vorüberlegungen zum Prototyp und zum bereitgestellten Equipment notwendig, die im Folgenden erläutert werden.

A. Hard- und Software

1) LEGO-Mindstorms-NXT: Für die mechanische Umsetzung bildet das LEGO-Mindstorms-NXT-Set die Basis der Hardware. Neben verschiedenen Sensoren und Bauteilen stehen auch Motoren zur Verfügung, die alle mit dem NXT-Stein verbunden werden können. Die Motoren bieten viele Einstellmöglichkeiten. Sie können sich z. B. über den Power-Wert schneller oder langsamer bewegen oder über das Tacho-Limit um eine bestimmte Gradzahl drehen.

2) MATLAB und RWTH – Toolbox: Die Programmierung des NXT-Steins erfolgt mit der numerischen Rechen- und Programmiersoftware *MATLAB* und der zur Verfügung gestellten *RWTH-Toolbox* aus Achen [4], mit der die Motoren und Sensoren über den NXT-Stein angesteuert werden können.

3) Leuchtmittel: Für die Realisierung der Lichtstrahlen ist der *Calliope mini* [5] eine gute Alternative zur normalen LED. Er lässt sich einfach in verschiedenen webbasierenden Entwicklungsumgebungen programmieren. Die Farben sind manuell einstellbar und es gibt gute Befestigungsmöglichkeiten.

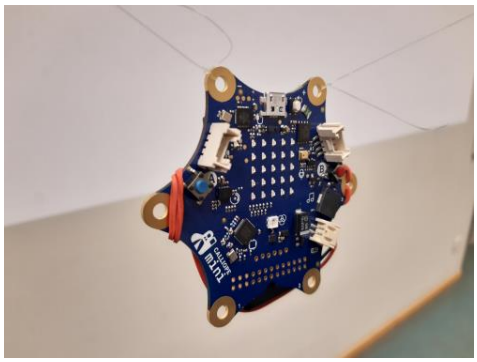


Abbildung 2. Calliope mini

DOI: 10.24352/UB.OVGU-2023-027

B. Sally can Draw

Der Roboter besteht aus einem Metallgerüst, in dessen Mitte der NXT-Stein befestigt ist. Außen sind zwei Motoren angebracht, die den Calliope mini über eine Schnur durch Auf- und Abrollen in die gewünschte Position bringen. Die softwaretechnische Umsetzung erfolgt in zwei Schritten. Dabei handelt es sich zum einen um die Bildverarbeitung und zum anderen um den Algorithmus, der für die Bewegung der Lichtquelle verwendet werden soll. Zunächst wird das Bild eingelesen und so gefiltert, dass alle Kanten erkannt werden. Anschließend werden die Kanten segmentiert, so dass ein binäres Bild entsteht. Eine „1“ steht für eine Stelle, an der sich eine Kante befindet, ansonsten stehen dort Nullen. Mit der Hilfe des Binärbildes fährt der Roboter jeden Pixel der Kante ab, bis er wieder an der Ausgangsposition angelangt ist. Dabei berechnet das Programm für jede Position des Pixels die Fadenlänge. Dazu werden die Motoren um 3° pro Pixel bewegt.

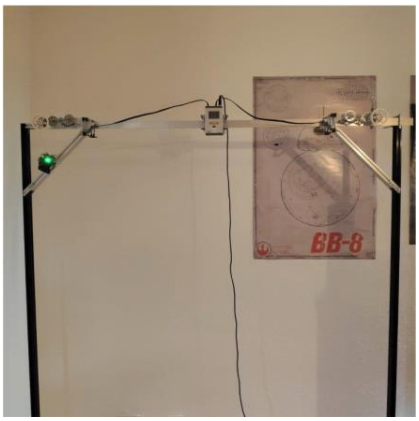


Abbildung 3. Sally can Draw

C. Potentielle Probleme

1) Konstruktion: Durch die Anordnung der Motoren innerhalb des Gerüstes kann der Platz nicht voll ausgenutzt werden, da die Verbindung zum NXT-Stein durch die Kabellänge begrenzt ist. Außerdem ist es schwierig, die Leistung der Motoren auf die entsprechende Seillänge einzustellen, da sie über Umwege über die Zahnräder auf die Rolle übertragen werden muss. Ein weiteres potentielles Problem ist die Instabilität der Konstruktion. Da die Klammern zur Befestigung der Motoren außen angebracht sind, kann die Rolle nach unten rutschen und das Endergebnis verschlechtern.

2) Programmierung: Durch die Ansteuerung der Motoren über das Tacho-Limit, werden die Formen nur sehr zeitaufwendig dargestellt und runde Objekte sind kaum realisierbar. Besser wäre es die Motoren über Power-Werte zu steuern, um eine kontinuierliche Linie zu erzeugen.

III. Technische Umsetzung

Die dazu notwendigen Vorüberlegungen und Konzepte sowie die technische Umsetzung zur Verbesserung dieser Probleme werden im Folgenden dargestellt.

A. Mechanische Konstruktion

Das Grundgerüst des Lightpainting-Roboters besteht wie beim Prototyp aus einem Metallrahmen, in dessen Mitte sich oben der NXT-Stein befindet. Die beiden Motoren sind hier jedoch direkt am NXT-Stein befestigt und wickeln den Calliope mini über eine Angelschnurrolle und zwei Umlenkrollen, die außen am Rahmen befestigt sind, in die gewünschte Position. Dadurch wurde zum einen die Platzausnutzung verbessert und zum anderen das Problem der Kabellänge gelöst. Außerdem bietet das Gerüst mehr Stabilität, da die Schienen auch zur Befestigung der Bauteile dienen. In der Konstruktion befinden sich zudem 4 Tastsensoren, mit denen die Motoren per Knopfdruck vorwärts bzw. rückwärts gesteuert werden können, um den Calliope mini vor dem Start des Programms in die Startposition zu bringen.

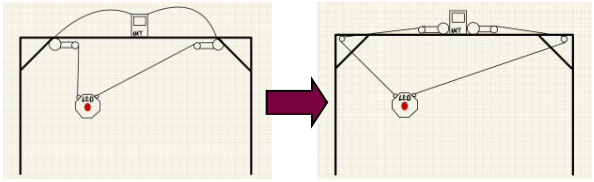


Abbildung 4. Aufbau: 1. Konzept (links), finales Konzept (rechts)

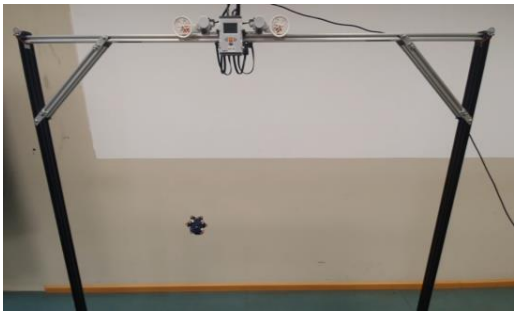


Abbildung 5. Gesamtkonstruktion (oben), manuelle Steuerung (links), NXT-Baustein und zwei Motoren mit Rollen (rechts)

B. Vorgehensweise für die Programmierung

Um eine präzise Linienführung zu realisieren und die Seillängen so anzupassen, dass der Lightpainting-Roboter die Form möglichst genau abfährt, müssen die in der Abbildung 3 dargestellten geometrischen Zusammenhänge angewendet werden. Daraus ergeben sich die Gleichungen (1) und (2), die für jede Koordinate der abzufahrenden Form die Längen der Seile berechnen. Aus diesen Längenänderungen und der zeitlichen

Differenz von Koordinate zu Koordinate ergeben sich die jeweiligen Geschwindigkeiten. Damit man diese an den Motor übertragen kann, muss zunächst eine Motorkalibrierung durchgeführt werden, die die Winkelgeschwindigkeit pro Power berechnet.

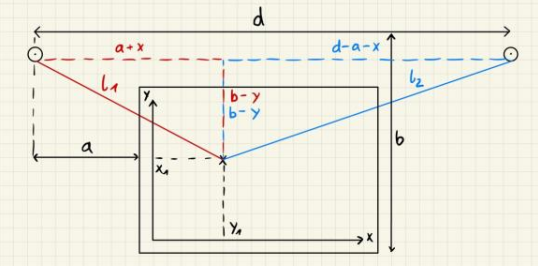


Abbildung 6. Geometrische Zusammenhänge

$$l_1 = \sqrt{(a+x)^2 + (b-y)^2} \qquad (1)$$

$$l_2 = \sqrt{(d-a-x)^2 + (b-y)^2} \qquad (2)$$

C. *Motorkalibrierung*

Die Kalibrierung wird mittels einer For-Schleife umgesetzt. Dabei wird bei jedem Schleifendurchlauf der Power-Wert des Motors erhöht und proportional dazu auch das Tacho-Limit. Sobald der Motor nach dem Start das Tacho-Limit erreicht, wird die Zeit gestoppt und ein neuer Schleifendurchlauf beginnt. Die Schleife endet, wenn der Motor einen Power-Wert von 100 % angenommen hat. Aus den aufgezeichneten Werten kann die Winkelgeschwindigkeit pro Power berechnet werden. Da sich der Motor nur bis ca. 80 % linear verhält, wie in Abbildung 4 zu sehen ist, ist dies auch der maximale Wert, der eingestellt werden kann. Andernfalls werden die linearen Abweichungen zu groß.

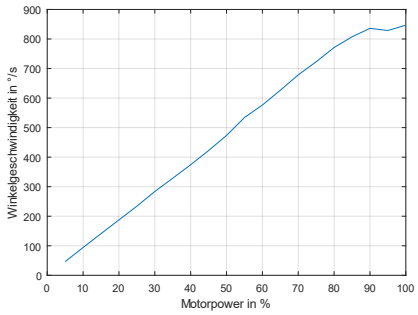


Abbildung 7. Winkelgeschwindigkeit in Abhängigkeit der Motorpower

D. *Programmablauf*

Der Benutzer hat die Möglichkeit, das Programm so einzustellen, dass die maximale Leistung des Motors abgerufen wird, so dass der Prozess in kürzester Zeit abgeschlossen werden kann. Er kann aber auch eine beliebige Zeit wählen, zu der das Programm beendet sein soll. Außerdem besteht die Möglichkeit, zwischen verschiedenen Formen und Größen zu wählen oder Freihandzeichnungen [6] in das Programm einzufügen. Nach der Auswahl wird die Form geplottet und für die jeweiligen Koordinaten werden die Längendifferenzen mit den Gleichungen (1) und (2) berechnet. Daraus ergeben sich die Winkeländerungen für die Motoren und mit dem Wert der Winkelgeschwindigkeit pro Power und der zeitlichen Differenz von Koordinate zu Koordinate, die sich aus den beiden Optionen am Anfang ergibt, die Power-Werte für die Motoren.

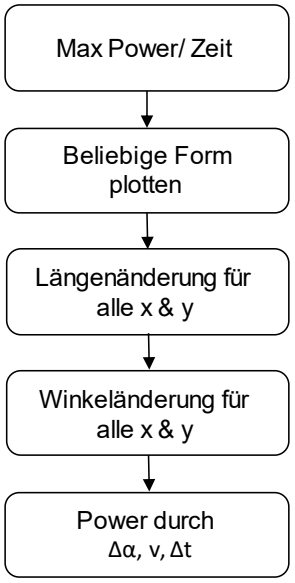


Abbildung 8. Ablaufplan des Algorithmus

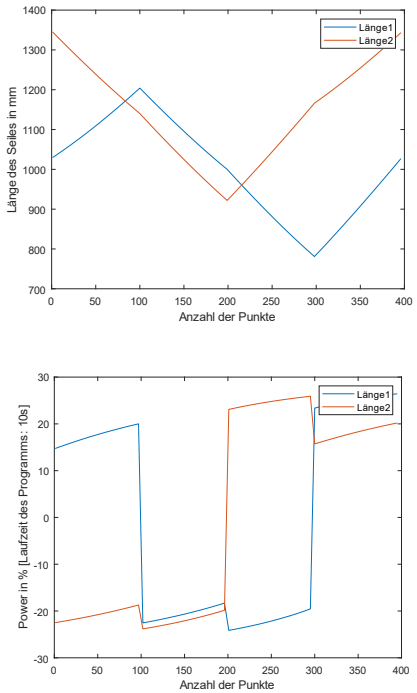


Abbildung 9. Länge der Seile (oben) und Power-Werte der Motoren (unten) beim Quadrat (a = 50 cm, 100 Punkte pro Seite)

IV. Ergebnisdiskussion

Am Ende des Projekts ist ein Lightpainting-Roboter entstanden, bei dem die Funktionen und der technische Aufbau des Prototyps verbessert wurden. Neben der maximalen Ausnutzung der Fläche und der Verbesserung der Stabilität des Gestells werden nun Endergebnisse mit klaren Linienführungen und vollständiger Wiedergabe der Formen erzielt. In der Abbildung 10 ist auch zu sehen, dass der Roboter neben dem Kreis, Stern, Quadrat und dem Dreieck auch Freihandzeichnungen darstellen kann. Das Problem dabei ist jedoch, dass die Ergebnisse meist sehr verwackelt sind, weil die Kamera nur maximal 30 Sekunden mit Langzeitbelichtung aufnehmen kann und der Roboter daher mit hektischen Bewegungen versucht, das Bild in dieser Zeit abzufahren. Neben diesem Problem gab es bei der Entwicklung des Programms noch viele weitere Schwierigkeiten. Zum einen stellten die Berechnungen für die Motorsteuerung eine große Herausforderung dar, aber auch das Plotten der Formen und die unzähligen Bugs während der Programmierphase kosteten viel Mühe und Zeit bei der Bearbeitung.

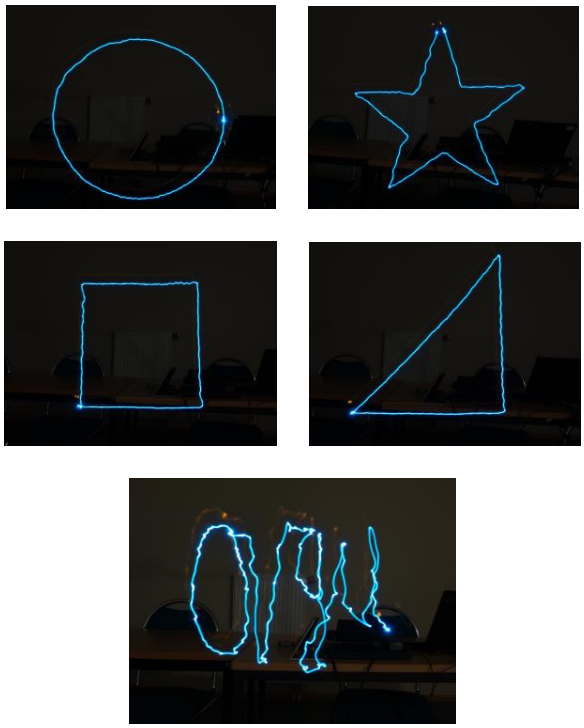


Abbildung 10. Ergebnisbilder: vier vorgefertigte Formen (oben), Freihandzeichnung (unten)

V. Zusammenfassung und Fazit

Ziel des Projektes war es, die Idee des Lightpainting-Roboters aus dem Jahr 2022 umzusetzen und weiterzuentwickeln. Dieses Ziel wurde in vollem Umfang erreicht. Sowohl die konstruktiven als auch die softwaretechnischen Verbesserungen werden aufgezeigt. Darüber hinaus werden aufgetretene Probleme und Endergebnisse dargestellt. Für die Zukunft bietet der Roboter jedoch noch Verbesserungspotential. Zum einen wäre es möglich, dass der Roboter während der Bewegung für eine gewisse Zeit nicht zeichnet, da der Calliope mini über Bluetooth gesteuert werden kann. So könnten alle möglichen Zeichnungen und Schriftarten realisiert werden. Eine weitere Verbesserungsidee, die allerdings schwieriger umzusetzen ist, wäre die Erweiterung von 2- ins 3-Dimensionale, indem man die Schiene mit weiteren Motoren beweglich macht.

Literaturverzeichnis

[1] Wikipedia: *Light Painting.* https://de.wikipedia.org/wiki/Light_Painting. Version: Dezember 2022

[2] Magdowski Mathias Schallschmidt Thomas, Gerlach Thomas, Pannicke Ennrico: *LEGO-Praktikum. Entwickeln, programmieren, optimieren: Berichte der Studierenden zum Projektseminar Elektrotechnik/Informationstechnik.* https://journals.ub.ovgu.de/index.php/LEGO/article/view/2072/2065. Version: August 2022

[3] Mundus: *Praxistest: Light-Painting.* https://www.foto-mundus.de/blog/praxistests/praxistest-light-painting-fujifilm-x-t10-fuji-xf10-24mm-f4.0-r-ois. Version: Januar 2016

[4] Behrens, Alexander: *RWTH - Mindstorms NXT Tool-box* https://www.mathworks.com/matlabcentral/fileexchange/18646-rwth-mindstorms-nxt-toolbox. Version: Oktober 2011

[5] Calliope gGmbH: *Calliope mini, technische Daten.* https://calliope.cc/calliope-mini/technische-daten. Version: Februar 2023

[6] MathWorks: *How can I draw with mouse as we do in paint?* https://de.mathworks.com/matlabcentral/answers/320227-how-can-i-draw-with-mouse-as-we-do-in-paint. Version: Februar 2023

Light-Painting-Robot 2.0

Tobias Wagner, Elektromobilität
Otto-von-Guericke-Universität Magdeburg

***Abstract*— Im Rahmen des jährlich stattfindenden Projektseminars Elektrotechnik/Informationstechnik wurde die Idee eines Light-Painting-Roboters weiterentwickelt. Entstanden ist der Light-Painting-Robot 2.0, der mit Hilfe eines LEGO-Mindstorms-Sets und einer MATLAB-Programmierung umgesetzt wurde. Mit dem Roboter können verschiedene geometrische Formen sowie einfache selbstgezeichnete Bilder realisiert werden.**

***Schlagwörter*—Calliope mini, LEGO Mindstorms, Light-Painting, MATLAB, Roboter**

I. Einleitung

LIGHT-PAINTING [1] bezeichnet eine Technik aus der Fotografie. Mittels Langzeitbelichtung können Formen oder Linien erzeugt werden, indem eine Lichtquelle vor der Kamera bewegt wird. Die besten Ergebnisse werden bei Dunkelheit erzielt. Das Gezeichnete wird erst im fertigen Bild sichtbar. Dadurch ist es für den Künstler schwierig zu erkennen, wo er bereits gezeichnet hat. Möchte man eine geschlossene Form zeichnen, ist es nur mit Glück und vielen Versuchen möglich, dass sich Anfangs- und Endpunkt der Zeichnung in einem Punkt treffen. Auch die Belichtungszeit spielt eine wichtige Rolle. Je länger diese ist, desto mehr Rauschen ist im Bild zu sehen [2]. Der Light-Painting-Robot 2.0 soll diese Probleme lösen. Mit ihm ist es möglich, möglichst schnell ein präzises Bild mit klaren Linien zu zeichnen, um die Belichtungszeit zu minimieren.

II. Vorbetrachtungen

Zur Realisierung des Roboters wurde ein bereits bestehendes mechanisches Konzept verwendet. Zusätzlich zu den Teilen, die in dem LEGO Bausatz enthalten sind, wurden noch einige weitere Materialien verwendet.

A. Inspiration

Die ursprüngliche Idee eines Light-Painting-Roboters stammt aus dem Projekt „Sally can Draw: Automatisierter Light-Painting Roboter“ [3] aus dem Jahr 2022. Nach mehreren Konstruktionsansätzen fehlte im letzten Jahr die Zeit, um das Endergebnis zu perfektionieren. An diesem Punkt setzt der Light-Painting-Robot 2.0 an, wobei die mechanische Grundstruktur des Roboters erhalten bleibt. Eine schrittweise Optimierung der Mechanik und eine neue Software sind die wesentlichen Unterschiede.

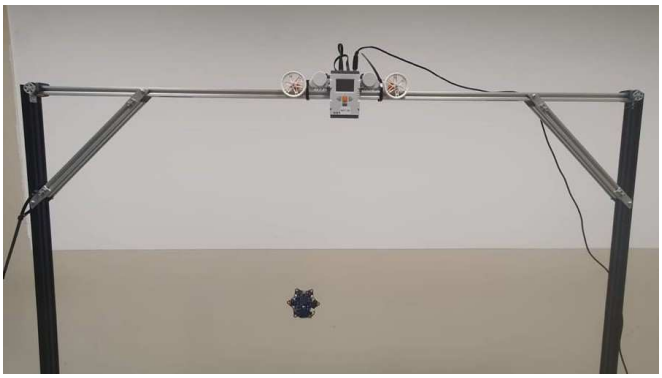
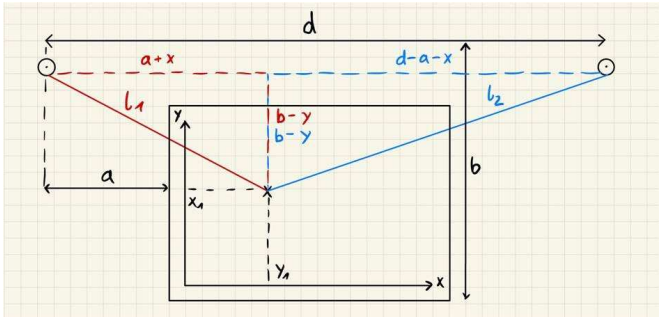


Abbildung 1: Aufbau des Light-Painting-Robot 2.0

B. Geometrie

Um zeichnen zu können, benötigt der Roboter zunächst einen vorgegebenen Rahmen. Dieser Rahmen ist in diesem Fall ein Koordinatensystem, da das MATLAB-Programm dieses benötigt, um die Größe und Position des zu zeichnenden Bildes bestimmen zu können. Um die LED wie gewünscht bewegen zu können, müssen die Schnüre entsprechend auf- oder abgerollt werden. Für jeden Punkt im Koordinatensystem ergibt sich eine bestimmte Länge für die linke und eine für die rechte Schnur. Als Bezugspunkte für die Längen dienen die Umlenkrollen, die am Rand des Gestells angebracht sind. Mit Hilfe des Satzes des Pythagoras können die erforderlichen Schnurlängen für jede x/y-Kombination berechnet werden. Abbildung 2 zeigt die zugrunde liegende Berechnung. Bestimmt man die Längendifferenz der Schnüre zwischen zwei Punkten, so kann die LED von einem Punkt zum anderen bewegt werden. Dieses Prinzip bildet die Grundlage für das Zeichnen verschiedener Formen.



$$l1 = \sqrt{(a+x)^2 + (b-y)^2} \qquad l2 = \sqrt{((d-a-x)^2 + (b-y)^2}$$

Abbildung 2: Geometrie zur Berechnung der Schurlängen

10.24352/UB.OVGU-2023-028

C. Material

Da der Roboter nicht allein mit LEGO Mindstorms realisierbar ist, werden Zusatzkomponenten benötigt. Die größte Komponente ist das Metallgestell, auf dem die LEGO-Komponenten montiert werden. Als Lichtquelle wird ein Calliope mini [4] verwendet. Dieser Mikrocontroller bietet viele Funktionen, wobei in diesem Projekt der Schwerpunkt auf der Konstruktion und Programmierung des Roboters liegt und daher nur die RGB-LED des Calliope mini verwendet wird. Die Lichtquelle wird mit Hilfe von zwei Angelschnüren aufgehängt. Diese sind sehr dünn und reißfest und daher optimal geeignet. Die Stromversorgung des Calliope mini erfolgt über einen Akku, der auf der Rückseite angebracht ist.

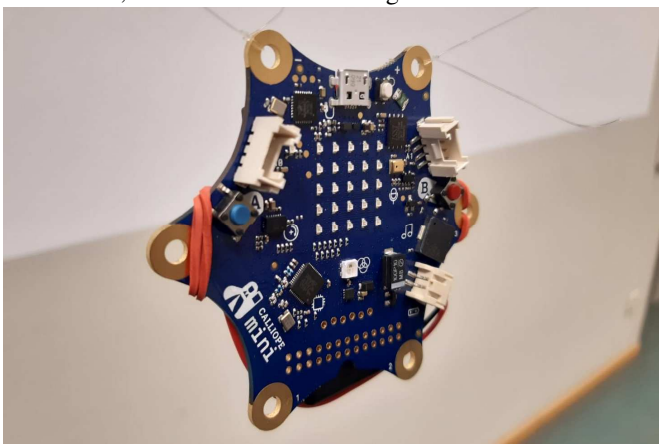


Abbildung 3: Calliope mini an Schnüren befestigt

III. Entwicklungsprozess

Im Bereich der mechanischen Konstruktion konnten schrittweise Verbesserungen erzielt werden, und im Bereich der Programmierung konnte nach mehreren Ansätzen eine Lösung gefunden werden.

A. Konstruktion

Ausgehend von der letztjährigen Konstruktion wurde mit der Entwicklung begonnen. Durch die Platzierung der Motoren auf dem NXT-Stein wurde das Problem der begrenzten Kabellänge zur Verbindung der Komponenten umgangen. An jedem Motor ist ein Rad befestigt, auf dem die Schnur aufgewickelt wird. Durch Umlenkrollen am äußeren Rand des Gestells ist es möglich, fast die gesamte Breite des als Zeichenfläche zu nutzen. Eine drehbare Lagerung der Umlenkrollen minimiert die Reibung der Schnur und reduziert somit Vibrationen. Die Inkompatibilität der LEGO Elemente mit dem Metallrahmen stellt ein Problem bei der Befestigung dar. Im Entwicklungsprozess wurden die ursprünglich verwendeten Kabelbinder durch eine Haltekonstruktion aus LEGO ersetzt. Diese Konstruktion sowie die Umlenkrollen werden mit Gummibändern fixiert, um Bewegungen der Bauteile auf dem Metallrahmen zu verhindern. Auf der Rückseite des NXT-Steins befinden sich vier Schalter, mit denen die LED manuell ausgerichtet werden kann. So kann im Voraus festgelegt werden, wo der Roboter mit dem Zeichnen des Bildes beginnen soll.

Abbildung 4: NXT und Motoren

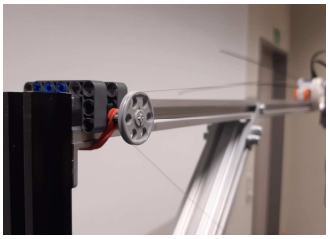


Abbildung 5: Umlenkrolle

B. Programm

Bei der Erstellung des MATLAB-Codes wurden zwei Ansätze verfolgt. Diese unterscheiden sich im Wesentlichen in der Art der Motoransteuerung.

1) Motoransteuerung durch Tacholimit: Mit dieser Methode war es nicht möglich präzise Linien abzufahren. Beim ersten Ansatz werden die Motoren mit einer konstanten Leistung und einer vorher berechneten Tachobegrenzung für die zu fahrende Strecke versehen. Die Motoren müssen also eine bestimmte Anzahl von Umdrehungen bei konstanter Drehzahl ausführen und dann anhalten. Bei diesem Ansatz erreichen die Motoren ihr Tacholimit jedoch zu unterschiedlichen Zeitpunkten. Mit dieser Methode war es nicht möglich, präzise Linien abzufahren.

2) Motoransteuerung durch Power: Bei dieser Variante werden die Motoren in kurzen Zeitabständen mit einer jeweils neu berechneten Motorpower beaufschlagt. Dazu muss zunächst eine Motorkalibrierung durchgeführt werden. Dabei wird die Leistung in Fünferschritten von 0 auf 100 % erhöht. Proportional dazu wird das Tacholimit erhöht. Theoretisch sollte die Zeit bis zum Erreichen des Tacholimits konstant sein. In der Praxis war dies jedoch nicht der Fall. Als Grund für die aufgetretenen Ungenauigkeiten kann die unzureichende Datenübertragungsgeschwindigkeit mit dem in die Jahre gekommenen NXT-Stein genannt werden. Trotzdem sind die Ergebnisse bis zu einer Leistung von 80% ausreichend genau, weshalb diese als maximale Power definiert wurde. Der größte Teil der Programmierung besteht aus Berechnungen, die dem eigentlichen Zeichnen vorausgehen. Zuerst wird die Form, die gezeichnet werden soll, in MATLAB gezeichnet. Diese Form wird in n Teile aufgeteilt, so dass viele kleine Abschnitte entstehen. Später kann für jeden zu zeichnenden Abschnitt eine spezifische Power an die beiden Motoren gesendet werden. Zuvor wird die erforderliche Längenänderung der Schnüre und die daraus resultierende Winkeländerung am Motor berechnet. Die resultierende Power wird anschließend noch geglättet, um einen weicheren Übergang zwischen den Ansteuerungen zu erreichen. Zusätzlich ist die Zeit, die der Roboter zum Zeichnen des Bildes benötigt, variabel einstellbar. Dies ermöglicht eine einfache Anpassung an die gewünschte Belichtungszeit, mit der das Bild aufgenommen werden soll. Auf diese Weise können geometrische Formen gezeichnet werden. Um ein eigenes Bild vom Roboter zeichnen zu lassen, wird auf ein externes MATLAB-Programm [5] zurückgegriffen, welches die eigene Zeichnung einliest und so anpasst, dass sie vom Roboter gezeichnet werden kann.

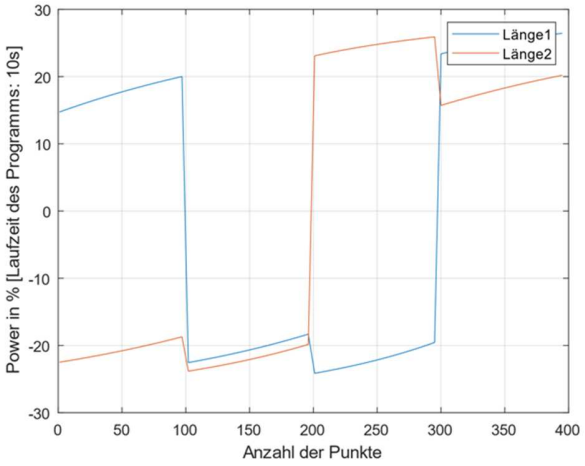


Abbildung 6: Motorpower beim Zeichnen eines Rechtecks

IV. Ergebnisdiskussion

Das Ergebnis ist angesichts der begrenzten Zeit von zwei Wochen sehr zufriedenstellend. Der Roboter ist in der Lage, die vorgegebenen Formen sehr genau und gut reproduzierbar zu zeichnen. Die Probleme mit dem Wackeln des Calliope mini wurden durch mehrere Maßnahmen minimiert. Zum einen durch den auf der Rückseite angebrachten Akku. Durch das erhöhte Gewicht ist die Lichtquelle weniger anfällig für auftretende Vibrationen. Auch die Glättung der Übergänge bei Richtungsänderungen sowie die drehbar gelagerten Umlenkrollen erzielen diesen Effekt. Das Ergebnis des Freihandzeichnens ist noch verbesserungswürdig. Das Zeichnen von Schriftzügen ist schwierig, da man schon bei wenigen Buchstaben an die Grenze der maximal sinnvollen Belichtungszeit stößt. Außerdem muss eine sehr hohe Power gewählt werden, was zu einem stärkeren Wackeln der Lichtquelle und damit zu ungenauerem Zeichnen führt. Nachdem die endgültige Programmiermethode festgelegt war, traten keine größeren Probleme mehr auf. Die Aufgabe bestand fortan darin, die Funktion zu optimieren und kleinere Fehler zu beheben. Die Verbesserung der Benutzerfreundlichkeit wurde schließlich durch die Erstellung von Funktionen in MATLAB und das Hinzufügen der manuellen Steuerung erreicht.

Abbildung 7: Ergebnisbild Stern

V. Zusammenfassung und Fazit

Zusammenfassend kann gesagt werden, dass der Light-Painting-Robot 2.0 seinen Zweck gut erfüllt. Er stellt eine Weiterentwicklung gegenüber dem letztjährigen Projekt dar und bietet zudem mehr Funktionen. Die Zeichenfläche ist größer und die gesamte Konstruktion ist stabil. Verschiedene Formen können gezeichnet werden. Erst bei komplexeren Zeichnungen stößt der Roboter an seine Grenzen. Hier gibt es den größten Verbesserungsbedarf. Auch das Ergebnisbild des Sterns (Abbildung 7) kann sicherlich noch perfekter gezeichnet werden. Mit einigen Verbesserungen in der Programmierung oder der Konstruktion sollte hier ein nahezu perfektes Bild im Bereich des Möglichen liegen. Durch eine Implementierung des Calliope mini wäre es möglich, die Farben während des Zeichnens zu ändern oder die LED zu bestimmten Zeiten ein- und auszuschalten. Damit wäre es auch möglich, Bilder zu erzeugen, die nicht aus einer geschlossenen Linie bestehen. Im Großen und Ganzen funktionieren die Grundfunktionen des Light-Painting-Robot 2.0 sehr gut und können mit mehr Zeit noch weiter ausgebaut und perfektioniert werden.

Literaturverzeichnis

[1] WIKIPEDIA: *Light Painting.* https://de.wikipedia.org/wiki/Light_Painting. Stand: Februar 2023.

[2] Cyber-shot Benutzeranleitung: *Bulb.* https://helpguide.sony.net/dsc/1210/v1/de/contents/02/03/07/07.html. Stand: Februar 2023

[3] LEGO-Praktikum. Entwickeln, programmieren, optimieren: *Gruppe 3.* https://journals.ub.ovgu.de/index.php/LEGO/article/view/2072/2065 Stand Februar 2023.

[4] Calliope: *Calliope mini Übersicht.* https://calliope.cc/calliope-mini/uebersicht Stand: Februar 2023.

[5] MathWorks: *How can I draw with mouse as we do in paint? :* https://de.mathworks.com/matlabcentral/answers/320227-how-can-i-draw-with-mouse-as-we-do-in-paint Stand: Februar 2023.

Farbsortiermaschine

Sebastian Knospe, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Während des LEGO-Praktikums erhielten die Studierenden nicht nur eine theoretische Einführung in die Programmierung mit MATLAB, einer Software zur grafischen Darstellung von Ergebnissen und zur Lösung mathematischer Probleme, sondern sie konnten auch ihrer Kreativität beim praktischen Bau eigener LEGO-Roboter freien Lauf lassen. Im Verlauf des zweiwöchigen Praktikums wurde eine Maschine entwickelt, die mit Hilfe von LEGO Minstorms Sensoren und Motoren in der Lage ist, Objekte nach ihrer Farbe zu sortieren.**

***Schlagwörter*—Farberkennung, LEGO Mindstorms, MATLAB, Programmierung, Sortiermaschine**

I. Einleitung

DER stetige technologische Fortschritt hat zu einer kontinuierlichen Steigerung der Produktionsgeschwindigkeiten in der industriellen Fertigung geführt.
Diese Entwicklung erfordert eine gleichzeitige Anpassung der Qualitätssicherungsmaßnahmen, um den höchsten Standards und Anforderungen gerecht zu werden und den hohen Geschwindigkeiten zu entsprechen.
In vielen Fällen reicht eine manuelle Prüfung der Produkte nicht mehr aus, um den hohen Taktzahlen der Fertigungsprozesse gerecht zu werden. Die Qualitätskontrolle muss daher zunehmend von automatisierten Systemen wie Robotern übernommen werden, die aufgrund ihrer herausragenden Eigenschaften – wie unvergleichliche Präzision und außerordentliche Schnelligkeit – in der Lage sind, die Anforderungen an eine präzise und effiziente Qualitätssicherung zu erfüllen.
Aus einer außerordentlichen Leidenschaft für das Qualitätsmanagement entstand die Idee, die Sortierung von Objekten auf intelligente Weise effizienter zu gestalten. Diese Idee beinhaltet die Entwicklung einer Sortiermaschine, die in der Lage ist, Objekte anhand ihrer Farbe zu erkennen und gezielt zu sortieren.

II. Vorbetrachtungen

Im folgenden Abschnitt wird die Kernidee des Konzeptes im Hinblick auf die Schaffung eines klaren Verständnisses für die Umsetzung erläutert. Insbesondere werden die für die Realisierung der Farbsortiermaschine notwendigen Komponenten vorgestellt.

A. Was soll die Sortiermaschine alles können?

Die Farbsortiermaschine hat die Aufgabe, Objekte nach ihrer Farbe in die dafür vorgesehene Farbkiste zu sortieren. Dazu wird das Objekt auf ein Förderband gelegt, welches nach Betätigung eines Tasters in Bewegung gesetzt wird. Mit einer Kamera wird dann ein Foto des Objektes erstellt, aus dem die Farbe des Objektes extrahiert wird. Nach der Ermittlung der Farbe werden die Sortierkisten so positioniert, dass das Objekt entsprechend seiner Farbe in die dazugehörige Kiste fällt.

DOI: 10.24352/UB.OVGU-2023-029

B. Welche Komponenten werden dafür benötigt?

Für die Realisierung steht ein LEGO-Mindstorms-Set zur Verfügung, welches eine Vielzahl verschiedener Elemente enthält. Diese sind in Abbildung 1 dargestellt, darunter verschiedene Sensoren, Motoren, LEGO-Bausteine und vieles mehr. Der NXT-Stein fungiert als unsere zentrale Steuereinheit, an der sämtliche Motoren, Sensoren und schließlich auch der Laptop, auf dem das Programm läuft, angeschlossen sind. Ein Motor wird verwendet, um das Förderband anzutreiben, und ein weiterer wird verwendet, um die Kisten für die Sortierung zu bewegen. Ein Ultraschallsensor und ein Tastsensor dienen zur genauen Positionsbestimmung der Sortierkisten, während ein zweiter Tastsensor zum Starten des Förderbandes verwendet wird. Zudem wird eine Kamera zur Farberkennung eingesetzt.

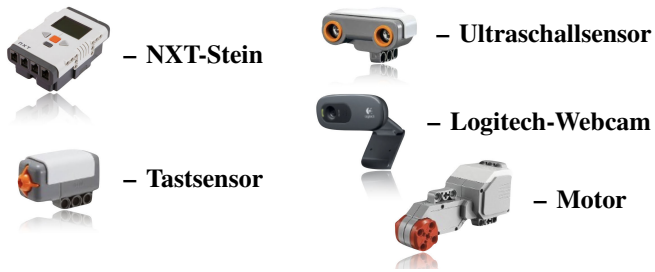


Abbildung 1. LEGO-Mindstorms-Komponenten entnommen aus [3]–[7]

III. Umsetzung

In diesem Abschnitt wird die Umsetzung der Farbsortiermaschine in einzelnen Schritten erläutert, beginnend mit einer detaillierten Funktionsdarstellung und dem Aufbau der Maschine. Dieser beinhaltet eine Funktionsbeschreibung der Komponenten. Abschließend wird auf die Programmierung im Hinblick auf die Arbeit mit einer Kamera eingegangen.

A. Funktionsweise

Die Funktionsweise der Sortiermaschine ist im folgendem Programmablaufplan dargestellt, der in Abbildung 2 veranschaulicht wird.

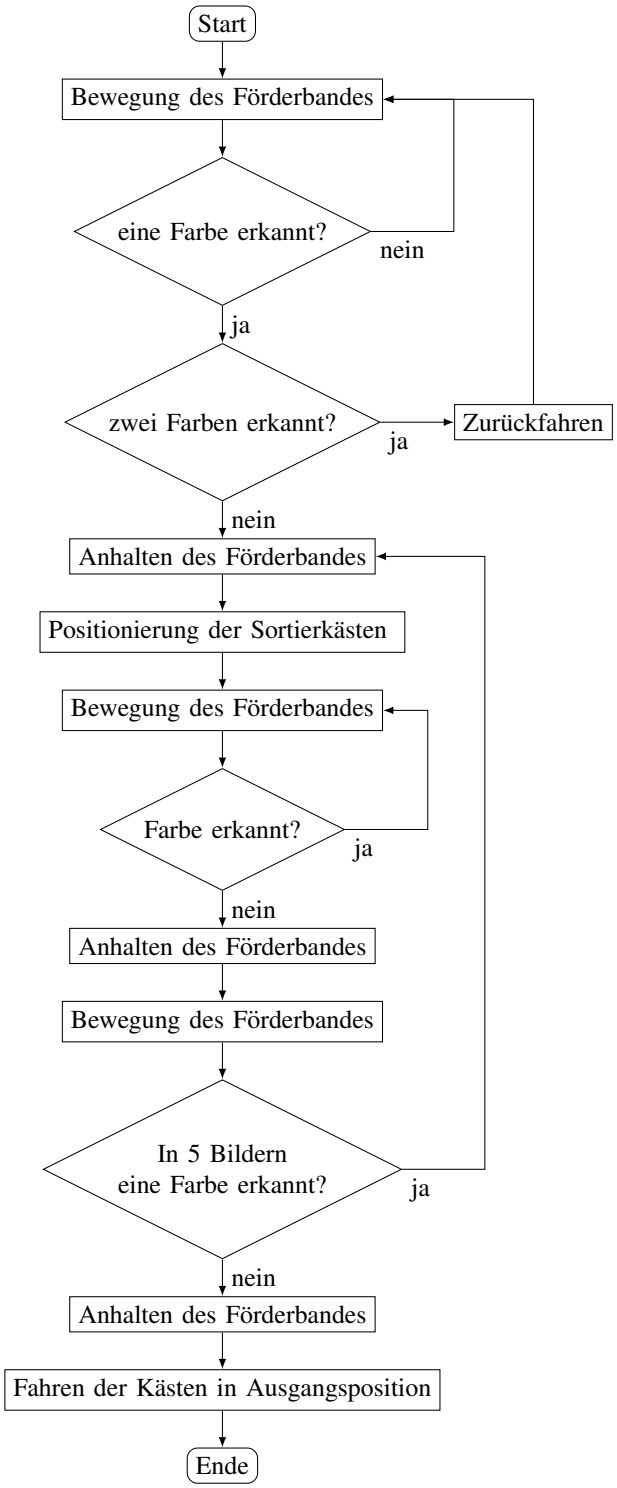


Abbildung 2. Programmablaufplan des Sortiervorganges

B. Aufbau und Funktion der Komponenten

Die Sortiermaschine in der Abbildung 3 besteht aus 3 Hauptkomponenten, dem Förderband, der Kamera und den beweglichen Sortierkisten. Das Förderband dient zum Transport der farbigen Objekte und besteht aus zwei hintereinander geschalteten Förderbändern. Diese beiden Bänder sind über ein einfaches Getriebe miteinander verbunden, das aus drei Zahnrädern, zwei Wellen (einer Antriebs- und einer Abtriebswelle) und einer zentralen Achse zur Verbindung der Komponenten besteht. Das Förderband wird von einem Motor angetrieben, der erst nach Betätigung eines Tasters in Gang gesetzt wird.
Die Farberkennung der auf dem Förderband transportierten Objekte erfolgt durch eine Logitech Webcam, die in regelmäßigen Abständen Bilder aufnimmt. Diese Kamera ist extern mit einem Laptop verbunden und wird über MATLAB, einer Software zur Datenanalyse und -verarbeitung, in den Arbeitsablauf integriert. Obwohl auch ein Farbsensor eine Option gewesen wäre, hätte dies die Objekte in ihrer Form und Größe stark eingeschränkt. Aus diesem Grund wurde die Webcam als optimale Lösung zur Farberkennung gewählt.
Am Ende des Förderbandes befinden sich die Sortierkisten, welche auf einem Fahrzeug montiert sind. Dieses kann mit Hilfe eines Motors bewegt werden. Wenn ein Objekt eine bestimmte Farbe hat, wird das Fahrzeug so gesteuert, dass die Sortierkisten in die richtige Position fahren und das Objekt in die entsprechende Kiste fällt. Um sicherzustellen, dass die Position der Sortierkisten immer korrekt ist, wird ein Ultraschallsensor verwendet, der kontinuierlich den Abstand zum Fahrzeug misst. Allerdings misst dieser auf kurze Distanz sehr ungenau, weshalb zusätzlich ein Tastsensor eingesetzt wird. Dieser Tastsensor dient als Anschlag und bewirkt bei Betätigung durch das Fahrzeug, dass dieses zum Stillstand kommt.
Schließlich werden alle Komponenten (Sensoren und Motoren) an den NXT-Stein angeschlossen, der als zentrale Steuereinheit fungiert.

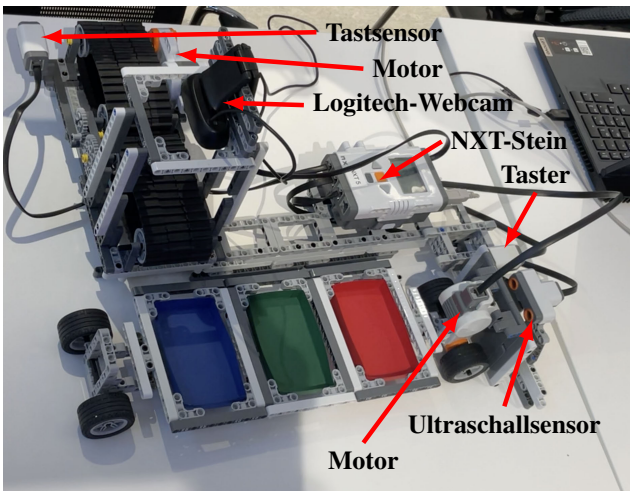


Abbildung 3. Aufbau der Farbsortiermaschine

C. Wie bekommt man aus einem Bild eine Farbe?

Gegenstand dieses Abschnittes ist die programmiertechnische Umsetzung, wobei eine Kamera zur Erkennung von Farben eingesetzt werden soll.

1) Bild mit einer Kamera erstellen: Der Beispielcode III-C1 zeigt die Erstellung eines Bildes mit den zugehörigen Farbwerten in MATLAB. Zunächst wird ein Kamera-Objekt mit dem Namen `camera` angelegt. In der nächsten Zeile des Codes wird mit der Kamera ein Bild erzeugt und mit dem `image`-Befehl in MATLAB angezeigt.

```
camera = videoinput('winvideo')
frame = getsnapshot(camera);
image (frame);
R = [frame(:,:,1)];
G = [frame(:,:,2];
B = [frame(:,:,3];
```

Beispielcode: Zum Erstellen und Anzeigen eines Bildes sowie zum Auslesen von Farbwerten

2) Farbwert aus einem Bild auslesen: In MATLAB wird ein Koordinatensystem über das Bild gelegt, das es ermöglicht, für jedes Pixel den entsprechenden Farbwert mit Hilfe der `Frame`-Funktion auszulesen. Die `frame`-Funktion besteht aus drei Komponenten (vgl. Abbildung 4), beginnend mit der X-Koordinate, gefolgt von der Y-Koordinate im Bild und schließlich dem Farbkanal (1 rot, 2 grün, 3 blau). Aufgrund der fehlenden Angabe konkreter X- und Y-Werte werden die Farbwerte des gesamten Bildes erfasst und in einer Matrix mit dem Namen `R` gespeichert.

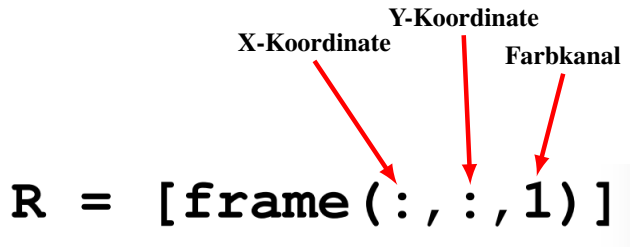


Abbildung 4. Aufbau der Frame Funktion

3) Farbbereiche in einem Bild markieren: Zunächst wird für jede vordefinierte Farbe (Rot-, Grün- und Blautöne) ein Farbbereich aus der Mischung der drei Grundfarbwerte (Rot-, Grün- und Blauwerte) erzeugt, um Farbbereiche in einem Bild zu kennzeichnen. Diese Werte werden anschließend in die `mask`-Funktion integriert (siehe die farbig hinterlegten Werte im Beispielcode III-C3). Dabei werden die Farbwerte aus dem gesamten Bild mit Hilfe der `frame`-Funktion ausgelesen. Liegen alle drei Farbwerte des Pixels im vordefinierten Bereich für Rot, Grün oder Blau, wird dieses Pixel durch die `mask`-Funktion markiert und erhält den Wert 1. Liegen die Werte außerhalb des Bereichs, erhält das Pixel den Wert Null. Alle Einsen und Nullen werden in einer neuen Matrix gespeichert.

```
mask = (frame(:,:,1)>=160)
   & (frame(:,:,1)<=255)
   & (frame(:,:,2)<=55) &
   (frame(:,:,3)<=55);
if any(mask(:))
r = 1;
end
```

Beispielcode: Zur Markierung roter Farbbereiche mithilfe der `mask`-Funktion

Um zu prüfen, ob eine spezifische Farbe im Bild vorhanden ist, wird die `if`-Funktion im Beispielcode III-C3 angewendet. Dabei wird die neu erzeugte Matrix, welche ausschließlich aus Nullen und Einsen besteht, auf das Vorhandensein von Einsen untersucht. Das Vorkommen einer einzigen 1 in der Matrix ist bereits ausreichend, um die im Bild vorhandene Farbe zu erkennen.

D. Schnelle Verarbeitung von Bildern

Im Folgenden geht es darum, möglichst schnell Bilder zu generieren und diese jeweils auf die Anwesenheit einer der vordefinierten Farben zu überprüfen.
Der in Abbildung III-D dargestellte Code ermöglicht eine effiziente Erstellung und Verarbeitung der Daten. Mit Hilfe der While-Schleife können Bilder erzeugt werden, solange eine bestimmte Bedingung erfüllt ist. Allerdings kann mit dieser Methode nur etwa alle 3 Sekunden ein Bild erstellt werden.
Der Grund dafür ist, dass bei alleiniger Verwendung der Schleife bei jedem Durchlauf das Kameraobjekt geöffnet, eine Aufzeichnung gestartet, ein Bild erzeugt, die Aufzeichnung gestoppt und das Kameraobjekt wieder geschlossen werden müsste. Hinzu kommt die Datenverarbeitung, die sehr zeitintensiv ist.
Die Verwendung des Befehls `triggerconfig (camera, 'manual')` beschleunigt die Datenverarbeitung erheblich, da nur einmalig eine Verbindung zum Gerät aufgebaut und eine Konfiguration durchgeführt wird. Außerdem werden die Daten nur einmal gespeichert, was eine schnelle Verarbeitung in MATLAB ermöglicht. Durch die Verwendung des `trigger`-Befehls ist es möglich, jede Sekunde ein Bild zu erzeugen. Erst durch diese Methode ist eine schnelle Farberkennung mit einer Kamera realisierbar.

```
triggerconfig(camera,'manual');
while (Bedingung)
frame = getsnapshot(camera);
end
```

Beispielcode: Für ständige Bilderstellung

E. Herausforderungen

Im Zuge des Entwicklungsprozesses traten verschiedene Herausforderungen auf, die es zu bewältigen galt. Bei der Konstruktion der Farbsortiermaschine erwies sich zunächst die Konstruktion des Getriebes, einschließlich der Auswahl geeigneter Zahnräder, als anspruchsvoll. Auch die exakte Positionierung und Ausrichtung der Kamera war komplex, da dies die Belichtung des Objekts und die Größe des Kamerabildes beeinflusst. Die Stabilisierung und Fixierung des Ultraschall- und Tastsensors stellten sich ebenfalls als Schwierigkeit heraus, da sie notwendig waren, um die Sensoren auszurichten und ihre Stabilität gegenüber den Bewegungen

des Fahrzeugs zu gewährleisten.
Ebenso war die Interpretation der Messdaten des Ultraschallsensors herausfordernd, da dieser für die exakte Positionierung der Sortierkisten verantwortlich ist. Darüber hinaus erwies sich die Inkonsistenz der Messwerte des Ultraschallsensors auf kurze Distanz als Herausforderung, da für die Ausgangsposition eine kurze Distanz vom Fahrzeug zum Ultraschallsensor erforderlich war.

IV. Ergebnisdiskussion

Am Ende waren die Herausforderungen gemeistert und eine funktionsfähige Farbsortiermaschine war das Ergebnis. Diese ist in der Lage, Objekte nach den Farben Rot, Blau und Grün mittels einer Kamera zu erkennen und in dafür vorgesehene Kisten zu sortieren.
Neben einfarbigen Objekten werden auch mehrfarbige Objekte erkannt und durch den Rückwärtslauf des Förderbandes zur Ausgangsposition zurückbefördert, wo sie dann manuell aussortiert werden können. Das Aussortieren von mehrfarbigen Objekten funktioniert allerdings nur im ersten Durchlauf, da die While-Schleife, die den Großteil des Programms umfasst, erst nach der Unterscheidung, ob mehrere Farben vorhanden sind, beginnt.
Ebenso besteht die Möglichkeit, die Bildverarbeitung zu beschleunigen, da diese fast nur noch durch die Prüfung, ob eine Farbe vorhanden ist, eingeschränkt wird. Dazu wäre es beispielsweise möglich, sich bei der Farberkennung auf die Bildmitte zu beschränken, wodurch deutlich weniger Pixel auf ihre Farbe hin überprüft werden müssten. Dies hätte zur Folge, dass Bilder in kürzeren Zeitabständen erstellt werden können und gleichzeitig vermieden wird, dass Objekte aufgrund zu großer Zeitintervalle bei der Bilderzeugung nicht erkannt werden.
Ebenfalls könnte durch die Integration weiterer Farben mit ihren jeweiligen Farbbereichen ausgeschlossen werden, dass bei mehrfarbigen Objekten nur eine der beiden Farben erkannt wird und diese somit in die falsche Kiste einsortiert werden. Dies war jedoch in diesem Fall nicht möglich, da die Länge des Kabels, das den Motor der Sortierkisten mit dem NXT-Stein verbindet, die Anzahl der Farben begrenzt. Zudem würde bei einer Erweiterung der Farbpalette die Bildverarbeitung wieder länger dauern.
Dies zeigt überaus deutlich, dass Prozesse oft ineinander greifen und sich somit gegenseitig beeinflussen, weshalb Verbesserungen immer im Gesamtsystem und nicht nur im Einzelschritt betrachtet werden müssen.

V. Zusammenfassung und Fazit

Zum Ende des LEGO-Praktikums war die Farbsortiermaschine, mit der Objekte nach ihrer Farbe sortiert werden können, fertiggestellt. Die Sortiermaschine hat noch viel Potenzial für Verbesserungen, wie bereits in der Ergebnisdiskussion festgestellt wurde. Die Umsetzung des selbst gesteckten Ziels wurde erreicht und das LEGO-Praktikum konnte erfolgreich abgeschlossen werden. Dabei wurde nicht nur die grundlegende Programmierung in MATLAB erlernt, sondern auch die Anwendung dieser Kenntnisse bei der Steuerung der selbstgebauten LEGO-Mindstorms-Roboter.

Literatur

[1] MathWorks:*triggerconfig*, http://de.mathworks.com/help/imaq/imaqdevice.triggerconfig.html?s_tid=srchtitle_triggerconfig_1 (Stand: Februar 2023)
[2] Mathworks: *Aquiring a Single Image in a Loop*, https://de.mathworks.com/help/imaq/acquiring-a-single-image-in-a-loop.html (Stand:Februar 2023)
[3] Joachim Bone: NXT-Stein:https://www.brick-shop.de/Elektrik-4951.html?language=de (Stand:Februar 2023)
[4] LEGO:Tastsensor:https://www.lego.com/cdn/cs/set/assets/bltee56bc5628e9b8bf/9843.jpg (Stand:Februar 2023)
[5] Amazone:LEGO NXT Ultraschallsensor Ultrasonic sensor: https://www.amazon.de/LEGO-NXT-Ultraschallsensor-Ultrasonic-sensor/dp/B000PM8I8O (Stand:Februar 2023)
[6] Amazone:Logitech-C270-Webcam:https://www.amazon.de/Logitech-C270-Webcam-720p-schwarz/dp/B01BGBJ8Y0 (Stand:Februar 2023)
[7] Brick Owl:LEGO-Mindstorms EV3 Groß Motor:https://www.brickowl.de/catalog/lego-%20ev3-%20large-%20servo-%20motor-%C2%A0set-%C2%A045502 (Stand:Februar 2023)

LEGO-Farbsortiermaschine

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

***Zusammenfassung*—Im Rahmen des LEGO-Praktikums sollten die Studierenden nicht nur das theoretische Programmieren mit MATLAB, einer Software zur Lösung mathematischer Probleme und zur grafischen Darstellung der Ergebnisse, erlernen, sondern sich auch kreativ und praktisch mit dem Bau von selbstgewählten LEGO Robotern betätigen. Dabei entstand während des zweiwöchigen Praktikums eine Farbsortiermaschine, die mit Hilfe verschiedener LEGO-Mindstorms-Sensoren Objekte nach ihrer Farbe sortieren kann.**

***Schlagwörter*—Automatisierung, Farbsortiermaschine, LEGO-Mindstorms, Lego-Praktikum, MATLAB, NXT**

I. Einleitung

MIT der Erfindung des Computers und dem immer schnelleren Fortschritt der Technik wurde die Entwicklung von Robotern nahezu mühelos und rasant vorangetrieben. Heute werden Roboter in vielen Bereichen der Industrie und des menschlichen Lebens eingesetzt. Die Aufgaben von Robotern sind dabei vielfältig, da sie Anforderungen mit hoher Präzision, Schnelligkeit und geringer Fehlerquote erfüllen können.
Doch was passiert, wenn in einem industriellen Fertigungsprozess fehlerhafte Produkte entstehen, die in Größe, Form oder Farbe voneinander abweichen?
Um eine gleichbleibend hohe Qualität eines Produktes zu gewährleisten, muss fehlerhafte Ware erkannt und aussortiert werden. Diese Qualitätsprüfung erfolgt meist noch durch Menschenhand, in zeitaufwendiger und mühsamer Arbeit. Dies kann jedoch schnell zu Messungenauigkeiten und Fehlern führen, weshalb auch in der Qualitätskontrolle zunehmender Roboter die menschliche Arbeitskraft ersetzen. Mit modernster Bildverarbeitung, Sensorik und künstlicher Intelligenz vermessen und überprüfen sie große Warenmengen und sortieren fehlerhafte Produkte in einem automatisierten Prozess aus. So stellen sie eine gleichbleibende Qualität der zu produzierenden Produkte sicher.
Aus der Faszination für Sortierroboter entstand das Ziel, eine Maschine zu entwickeln, die in der Lage ist, die Farben verschiedener Objekte zu erkennen und diese in dafür vorgesehene Kisten zu sortieren.

II. Vorbetrachtungen

A. Was soll die Sortiermaschine alles können?

Die Sortiermaschine soll verschiedene Objekte nach ihrer Farbe in bestimmte, für die jeweilige Farbe vorgesehene Kisten sortieren. Dazu werden die Objekte auf ein Förderband gelegt, das durch einen Startknopf in Bewegung gesetzt wird. Eine Kamera soll dann ein Bild des Gegenstandes erstellen, aus dem eine Farbe extrahiert werden soll. Wird im Kamerabild eine Farbe erkannt, soll die jeweilige Box unter das Förderband gefahren werden, so dass das Objekt mit der Farbe in die für diese Farbe vorgesehene Box fällt. Um Fehlsortierungen zu vermeiden, müssen also auch mehrfarbige Objekte erkannt und aussortiert werden.

B. Welche Komponenten werden dafür benötigt?

Zur Realisierung der Sortiermaschine steht ein LEGO-Mindstorms-Set mit verschiedenen Sensoren und Motoren zur Verfügung. Die zentrale Steuereinheit bildet der NXT-Stein, an den alle eingebauten Motoren und Sensoren angeschlossen sind. Die Programmierung des NXT-Steins erfolgt mit der Programmiersoftware MATLAB, die es mit Hilfe der Toolbox der RWTH Aachen ermöglicht, die Sensoren und Motoren über die MATLAB-Umgebung anzusprechen und zu konfigurieren. Zusätzlich dient ein Taster als Startknopf und eine Kamera nimmt Bilder der Objekte auf. Für die exakte Positionierung der Sortierboxen wird ein Ultraschallsensor verwendet. Damit sich nun die Kästen in die richtige Position bewegen und das Förderband die Objekte transportieren kann, sind zwei NXT-Motoren verbaut.

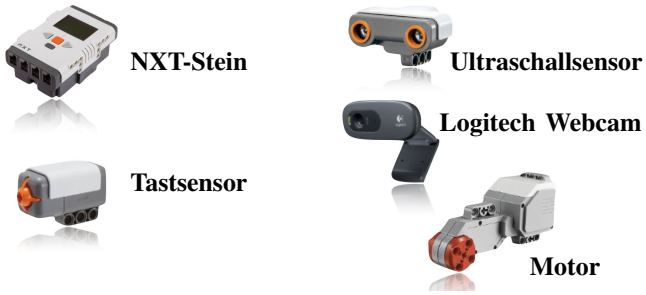


Abbildung 1. LEGO-Mindstorms-Komponenten entnommen aus [4]–[8]

III. Umsetzung

Im Folgenden soll die Umsetzung von der Konstruktion über die Funktionsweise bis hin zur Programmierung der Farbsortiermaschine erläutert werden.

A. Funktionsweise

Der folgende Programmablaufplan in Abbildung 2 zeigt die Funktionsweise der Sortiermaschine.

DOI: 10.24352/UB.OVGU-2023-030

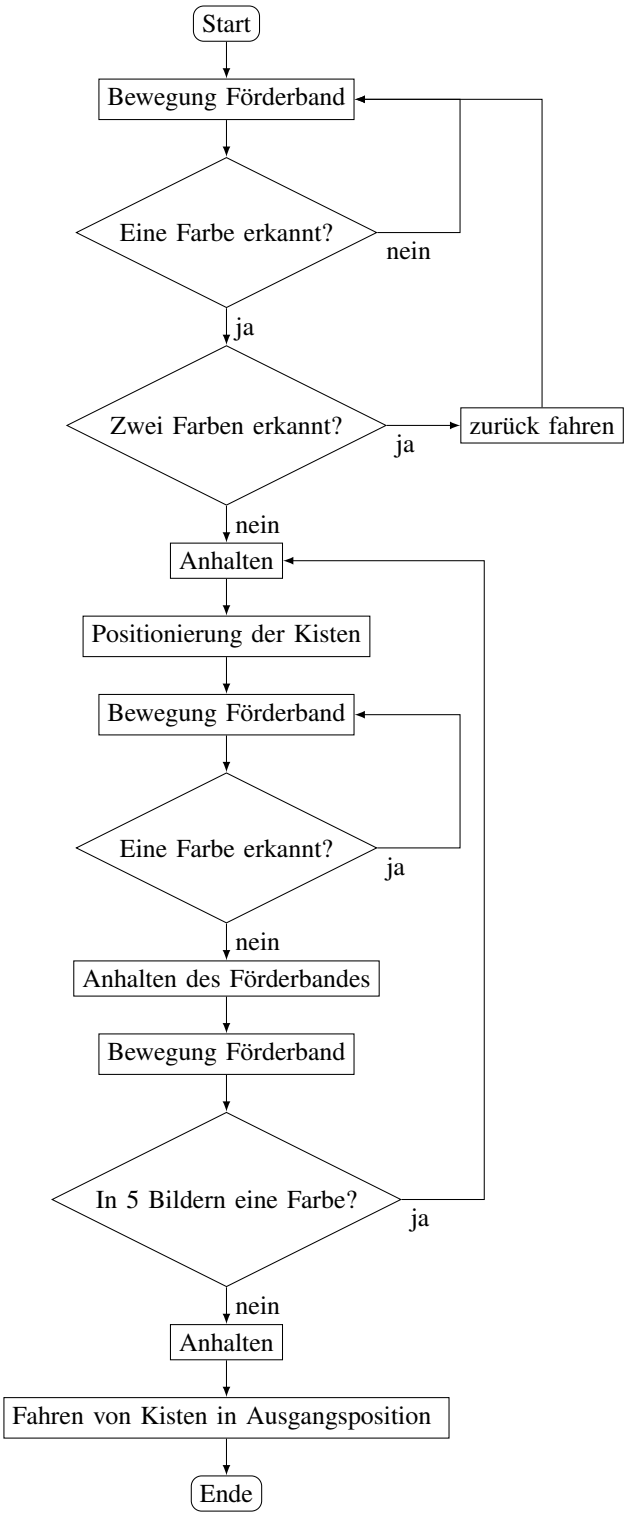


Abbildung 2. Programmablaufplan zur Funktionsweise

B. Aufbau und Funktion der Komponenten

Die Farbsortiermaschine in Abbildung 3 kann man sich aus drei Teilen, bestehend aus zwei hintereinander geschalteten Förderbändern, der Kamera und dem Fahrzeug für die Sortierboxen, vorstellen.

Zwei Förderbänder dienen dem Transport der farbigen Objekte. Damit auch größere Objekte nach ihrer Farbe sortiert werden können, sind zwei Förderbänder über ein Getriebe miteinander verbunden. Als Antrieb für das Band dient ein NXT-Motor, der durch Betätigen des Tastsensors gestartet wird.
Um Bilder der Objekte zu erstellen, wird eine Logitech Webcam verwendet, die extern an den Computer angeschlossen wird und in MATLAB integriert werden kann. Die Bilder der Kamera sind auf die sehr kurze Distanz nicht scharf, müssen es aber auch nicht sein, da es ausreicht eine Farbe im Bild zu erkennen, nicht aber das Objekt genau zu identifizieren.
Die Verwendung eines LEGO-Mindstorms-Farbsensors anstelle einer Kamera wäre nicht möglich gewesen, da die Objekte sehr nah an den Sensor gehalten werden müssten, um eine Farbe zu erkennen.
Damit die Objekte auch sortiert werden können, befindet sich nach dem Förderband ein Fahrzeug, an dem die Sortierboxen befestigt sind. Dieser fährt je nach erkannter Farbe die richtige Box vor das Förderband. Damit die exakte Positionierung der Boxen auch reibungslos funktioniert, ist zusätzlich ein Ultraschallsensor angebracht. Dieser misst permanent den Abstand zum Fahrzeug. Durch die Kopplung des Ultraschallsensors mit dem Fahrzeugmotor können die Kästen punktgenau vor das Förderband gefahren werden.
Da jedoch die Messergebnisse des Ultraschallsensors auf kurze Distanzen teilweise sehr ungenau sind, ist neben dem Sensor noch ein weiterer Tastsensor installiert, der als Anschlagpunkt dient. Fährt das Fahrzeug gegen den Tastsensor, kommt es zum Stillstand.
Damit aber alle verbauten Komponenten auch zusammenarbeiten können, müssen sie miteinander verbunden werden. Dies geschieht über den NXT-Stein, der als zentrale Steuereinheit dient. An ihn werden nicht nur alle Sensoren und Motoren angeschlossen, sondern er stellt auch die Schnittstelle zum MATLAB-Programm dar, mit dem die Komponenten programmiert werden.

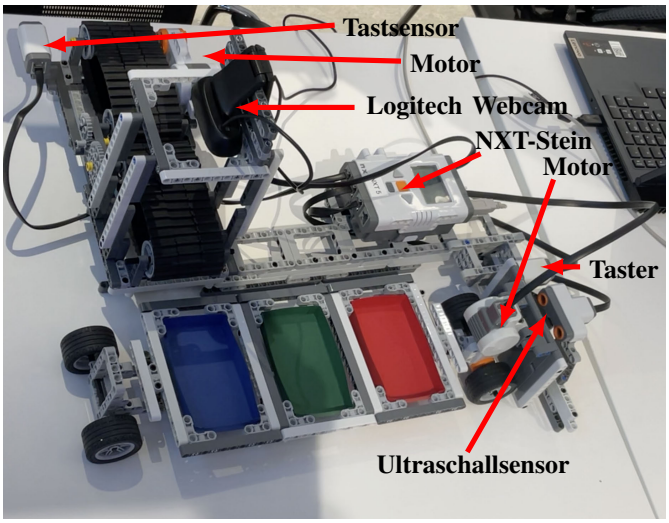


Abbildung 3. Aufbau

C. Wie bekommt man aus einem Bild eine Farbe?

Nachdem der Aufbau und die Idee der Funktionsweise erläutert wurden, besteht die Aufgabe nun darin, mit der Kamera Bilder aufzunehmen und in diesen Farbbereiche zu markieren sowie Aktionen abzurufen, wenn ausgewählte Farben erkannt werden.

```
camera = videoinput('winvideo')
frame = getsnapeshot(camera);
image (frame);
R = [frame(:,:,1)];
G = [frame(:,:,2)];
B = [frame(:,:,3)];
```

(Codebeispiel zum Erstellen von Bildern und auslesen von Farbwerten)

1) Bild mit einer Kamera erstellen: Mit dem abgebildeten Code kann eine Kamera in MATLAB eingebunden werden. Dazu wird zunächst ein neues Kameraobjekt mit dem Namen „camera“initialisiert. Mit den weiteren Befehlen kann dann ein Bild mit der Kamera erstellt und in MATLAB angezeigt werden. Nachdem ein Frame erzeugt wurde, kann aus diesem mit dem Frame-Befehl ein Farbwert ausgelesen werden.

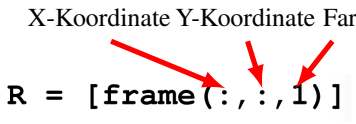


Abbildung 4. Aufbau des Frame-Befehls zum Auslesen roter Farbwerte

2) Farbwert aus einem Bild auslesen: In Matlab wird ein Koordinatensystem über das Bild gelegt, so dass mit Hilfe der Frame Funktion für jeden Bildpunkt ein Farbwert ausgelesen werden kann.
Der Frame-Befehl (siehe Abbildung 4) besteht aus drei Komponenten. An erster Stelle steht die X-Koordinate, an zweiter Stelle die Y-Koordinate im Bild und an dritter Stelle der Farbkanal (1 rot, 2 grün, 3 blau). Da hier keine konkreten X- und Y-Werte angegeben sind, werden die Farbwerte des gesamten Bildes ausgelesen und in einer Matrix mit dem Namen R gespeichert.

```
mask = (frame(:,:,1)>=160)
   & (frame(:,:,1)<=255)
   & (frame(:,:,2)<=55) &
   (frame(:,:,3)<=55);
if any(mask(:)) r=1;
end
```

(Codebeispiel zur Markierung von roten Farbbereichen)

3) Farbbereiche in einem Bild markieren: Um nun Farbbereiche in einem Bild markieren zu können, wird die Maskenfunktion verwendet (siehe Codebeispiel zur Markierung von roten Farbbereichen). Dabei werden die Farbwerte mit Hilfe der Frame-Funktion aus dem gesamten Bild ausgelesen. Damit aber nur vordefinierte Farben wie Rottöne, Grüntöne und Blautöne im Bild erkannt werden, muss zuvor über einen RGB-Farbmischer für jede Farbe ein Farbbereich aus der Mischung aller drei Farbwerte (Rot-, Grün-, und Blauwert) erzeugt werden. Diese mit dem Farbmischer definierten Werte werden verwendet, um Farbbereiche zu definieren, so dass nur Farbwerte in diesem Bereich von der Frame-Funktion ausgelesen werden. Liegen alle drei Farbwerte des Pixels in dem vordefinierten Bereich für Rot-, Grün- oder Blauwert, so wird dieses Pixel durch die „Mask-Funktion“mit dem Wert 1 markiert. Wenn die Werte außerhalb des Bereichs liegen, wird dem Pixel der Wert Null zugewiesen. Alle Einsen und Nullen werden in einer neuen Matrix gespeichert.
Um zu prüfen, ob eine bestimmte Farbe im Bild vorhanden ist, wird die if Anweisung (siehe Codebeispiel zur Markierung von roten Farbbereichen) verwendet. Dabei wird geprüft, ob in der neuen Matrix Einsen und Nullen vorkommen. Eine Eins in der Matrix reicht aus, um eine Farbe im Bild zu erkennen.

D. Schnelle Verarbeitung von Bildern

Nach dem Aufnehmen von Bildern und dem Erkennen von Farben in einem Bild, geht es nun darum, kontinuierlich und schnell Bilder aufzunehmen und bei jedem neuen Bild zu prüfen, ob die gewünschte Farbe im Bild vorhanden ist.

```
triggerconfig(camera, 'manual');
while
(Bedingung) frame =
   getsnapeshot(camera);
end
```

(Codebeispiel zur kontinuierlichen Erzeugung von Bildern)

Der obige Code ermöglicht eine schnelle Erzeugung und die anschließende schnelle Verarbeitung der Daten. Die while-Schleife erzeugt Bilder, solange eine bestimmte Bedingung erfüllt ist. Würde man zur Bilderzeugung nur die while-Schleife verwenden, müsste bei jedem neuen Durchlauf der Schleife das Kameraobjekt geöffnet, eine Erfassung gestartet, ein Frame erstellt, die Erfassung beendet und das Kameraobjekt geschlossen werden. Hinzu kommt noch die Datenverarbeitung. Durch die zeitaufwendigen Schritte entsteht so nur ca. 1 Bild in 3 Sekunden, was viel zu langsam ist, um die Farben der Objekte rechtzeitig zu erkennen und dann sortieren zu können. Mit dem Befehl „triggerconfig (camera,'manual')“wird die Verarbeitung weniger zeitintensiv, da nur einmal eine Verbindung zum Gerät hergestellt und eine Konfiguration durchgeführt wird. Man kann sich vorstellen, dass die Kamera im Bereitschaftsmodus gehalten wird. Außerdem werden die Daten nur einmal gespeichert, was eine schnellere Verarbeitung der Bilder in MATLAB ermöglicht. Durch die Verwendung des „Triggermodus“ist es möglich, 1 Bild in ca. 1 Sekunde zu erzeugen. Erst dadurch wird eine schnelle Farberkennung mit einer Kamera möglich gemacht.

E. Herausforderungen

Bei der Entwicklung der Maschine waren nicht nur konstruktive, sondern auch programmiertechnische Herausforderungen zu bewältigen. Dazu gehörte zunächst die konstruktive Schwierigkeit, die beiden Förderbänder über ein Getriebe miteinander zu verbinden, da die Zahnräder oft nicht die gewünschte Größe aufwiesen.

Das programmiertechnische Problem, das es zu lösen galt, bestand zum einen darin, Farbbereiche in einem Kamerabild zu erkennen und zu markieren, so dass bei Erkennen einer bestimmten Farbe eine ganz bestimmte Aktion ausgeführt wird. Zum anderen war die Interpretation der Messdaten des Ultraschallsensors und die Ansteuerung des Motors, zur exakten Positionierung der Kisten eine kleine Herausforderung. Erschwert wurde dies durch die teilweise ungenaue Messung des Ultraschallsensors auf kurze Distanzen, was durch einen nachträglich eingebauten Tastsensor behoben werden konnte.

IV. Ergebnisdiskussion

Nachdem alle bestandenen Herausforderungen und Probleme fast gelöst sind, kann sich das Endergebnis sehen lassen. Die Farben der Objekte darunter verschiedene Blau-, Grün- und Rottöne, werden in den Kamerabildern schnell erkannt und die Objekte entsprechend ihrer Farbe in Boxen sortiert. Auch mehrfarbige Objekte werden erkannt und aussortiert, indem das Förderband rückwärts läuft. Problematisch ist nur, dass das Aussortieren von mehrfarbigen Objekten nur zu Beginn des Sortiervorgangs funktioniert. Das bedeutet, wenn nach einem Sortiervorgang wieder mehrfarbige Körper auf das Förderband gelegt werden, werden diese nicht mehr erkannt und somit auch nicht mehr aussortiert. Des Weiteren besteht die Problematik, dass teilweise bei mehrfarbigen Körpern nur eine vordefinierte Farbe erkannt wird und somit diese Körper nach ihrer erkannten Farbe sortiert werden, was jedoch einen Sortierfehler darstellt, da nur rein farbige Objekte sortiert werden sollen. Nicht nur bei der Farberkennung können Fehler auftreten, sondern auch bei der Bilderzeugung. Da die Kamera nur ein Bild pro Sekunde aufnimmt, kann es vorkommen, dass ein Bild aufgenommen wird, bevor sich das Objekt im Bildbereich der Kamera befindet und erst wieder, wenn es den Bildbereich verlassen hat. Dies kann dazu führen, dass die Farben der Objekte nicht erkannt werden und ein Sortiervorgang nicht gestartet wird.

V. Zusammenfassung und Fazit

Das Ergebnis des zweiwöchigen Praktikums war eine Sortiermaschine, die verschiedene Objekte nach ihrer Farbe (blau, grün und rot) sortieren kann.
In Zukunft könnte an einigen Stellen noch eine Optimierung erfolgen. Zum Beispiel könnte der Code zur Farberkennung in MATLAB noch verkürzt und überarbeitet werden, um so noch schneller und effektiver Farben in einem Bild erkennen zu können. Des Weiteren könnte die Farberkennung mehrfarbiger Objekte erweitert werden, indem der Farbanteil der Farben am Objekt berechnet wird, so dass erst ab einem bestimmten Farbanteil sortiert wird und es nicht zu einer falschen Sortierung von mehrfarbigen Objekten kommt. Auch könnten noch weitere Farben hinzugefügt und die Anzahl der Sortierboxen erweitert werden.
Letztendlich wurde aber das Ziel, die Idee einer Farbsortiermaschine umzusetzen, erreicht. Somit war das LEGO-Praktikum ein voller Erfolg und hat zu dem auch viel Spaß gemacht.
Während des Praktikums konnte man nicht nur seiner Kreativität praktisch freien Lauf lassen, sondern lernte gleichzeitig die Grundlagen, einer neuen und in naturwissenschaftlichen Bereichen wichtige Programmiersprache, kennen.
In diesem Bericht wird die gesamte Arbeit, der zwei Wochen von der Entwicklung und den konstruktiven Ergebnissen über die Funktionsweise bis hin zu den wichtigsten Programmierschritten der Farbsortiermaschine dargestellt und näher erläutert.

Literaturverzeichnis

[1] MathWorks:*triggerconfig*, https://de.mathworks.com/help/imaq/imaqdevice.triggerconfig.html?s_tid=srchtitle_triggerconfig_1 (Stand: Februar 2023)
[2] MathWorks:*Acquiring a Single Image in a Loop*, https://de.mathworks.com/help/imaq/acquiring-a-single-image-in-a-loop.html (Stand:Februar 2023)
[3] Prasadi Gunawardana:*Geschichte der Roboter* (31.07.2022), https://wandelbots.com/de/geschichte-der-roboter (Stand:Februar 2023)
[4] Amazone:*Logitech-C270-Webcam* https://www.amazon.de/Logitech-C270-Webcam-720p-schwarz/dp/B01BGBJ8Y0
[5] Amazone:*LEGO NXT Ultraschallsensor Ultrasonic sensor* https://www.amazon.de/LEGO-NXT-Ultraschallsensor-Ultrasonic-sensor/dp/B000PM8I8O
[6] Joachim Boné:*NXT-Stein* https://www.brick-shop.de/Elektrik-4951.html?language=de
[7] LEGO:*Tastsensor* https://www.amazon.de/Logitech-C270-Webcam-720p-schwarz/dp/B01BGBJ8Y0
[8] Brick Owl:*LEGO-Mindstorms EV3 Groß Motor* https://www.brickowl.de/catalog/lego-ev3-large-servo-motor-set-45502

Halbautomatischer Getränkespender

Laurent Herms, ETIT
Otto-von-Guericke-Universität Magdeburg

***Abstract*— Aufgrund des demographischen Wandels in der heutigen Gesellschaft ist festzustellen, dass die qualifizierten Pflegekräfte dem steigenden Bedarf an Arbeitsaufgaben nicht mehr abdecken können. Die Pflegeeinrichtungen sind nachgewiesenermaßen so überlastet, dass die Bedürftigen nicht angemessen gepflegt werden können. Laut [2] werden bereits 2035 500.000 qualifizierte Pflegekräfte fehlen, um den steigenden Bedarf zu decken. Wäre es möglich, dass man weniger anspruchsvolle Tätigkeiten in Pflegeeinrichtungen, wie z.B. das Einschenken von Getränken, automatisiert werden könnten, um das Fachpersonal zu entlasten?**

***Schlagwörter*— Getränkespender, Kippvorrichtung, Automatisierung, Infrarotsensor, Tastsensor, Sprachsteuerung**

I. Einleitung

Das Einsatzgebiet dieses Gerätes ist die Alten- und Krankenpflege. Viele ältere Patienten haben Bedürfnisse wie z.B. das Trinken. Manchmal ist das Pflegepersonal so überlastet, dass diese Bedürfnisse nicht gehört oder vergessen werden. Mit dem halbautomatisierten Getränkespender könnte das Personal ein Behälter für den Patienten an dessen Bett bereitstellen. Bei Bedarf kann der Patient per Sprachsteuerung sich etwas einschenken lassen und dabei per Knopfdruck sein Glas nach Belieben füllen. Das Personal muss nur noch bei Gelegenheit das Gefäß wechseln.

II. Vorbetrachtungen

Laut [1] hat die Firma BRITA hat ein Gerät entwickelt, dass bei Annäherung eines Gefäßes an die Einfüllöffnung (Signal des dort eingebauten Ultraschallsensors) automatisch ausgießt. Das Gerät ist stationär und kann von jeder nicht bettlägerigen Person bedient werden. Der folgende Lösungsansatz hat für jeden bettlägerigen Patienten ein separates Gerät vorgesehen, mit dem das Glas des Patienten berührungslos per Sprachbefehl transportiert wird. Diese Lösung ist auch hygienisch, da das Gerät durch die Sprachsteuerung nicht vom Patienten selbst berührt wird. Nach einem ähnlichen Prinzip funktioniert auch der automatische Seifenspender. Bei Annäherung der Hand an den Sensor sendet dieser ein Signal und die Seife fließt aus einer Öffnung.

A. Aufbau

Der halbautomatische Getränkespender setzt sich zum einen aus dem Trinkglas-Transportfahrzeug (1) zusammen, welches ein Trinkglas (2) transportiert und den anzutreibenden Motor A (3). Und zum anderen das Gießgerät (4), welches aus dem Behälter (5), den Motor B (6), den Sprachsensor (7), den Tastsensor 2 (8), dem Steuergerät (9) und den Kipparm (10).

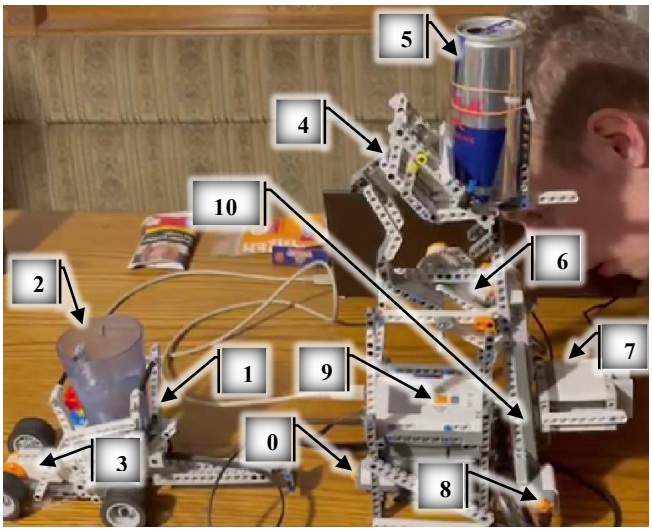


Abbildung 1: Aufbau

B. Funktionsweise

Mittels Sprachbefehl wird der Sprachsensor (7) gesteuert. Dadurch startet der Motor A (3) des Fahrzeuges (1). Stößt das Fahrzeug (1) bei der Fahrt zum Gießgerät (4) gegen den Tastsensor 1 (0), kommt das Fahrzeug zum Stehen, da der Motor nicht mehr rotiert bei der Berührung dieses Tastsensors (0). Gleichzeitig startet der Motor B (6), welcher eine Drehachse antreibt, die wiederum mechanisch mit dem Kipparm (10) des Gießgerätes (4) verbunden ist. Durch die Rotation des Motors (6) wird der Kipparm (10) samt dem Behälter (5) geneigt. Aufgrund dieser Neigung des Kipparms (10) wird der Inhalt des Behälters (5) ins Trinkglas (2) entleert. Bei Betätigung des Tastsensors 2 (8) rotiert der Motor B (6) zurück zu seiner Ausgangstellung, wodurch wiederum der Kipparm (10) zurück geneigt wurde. Gleichzeitig fährt das Fahrzeug (1) zurück.

DOI: 10.24352/UB.OVGU-2023-031

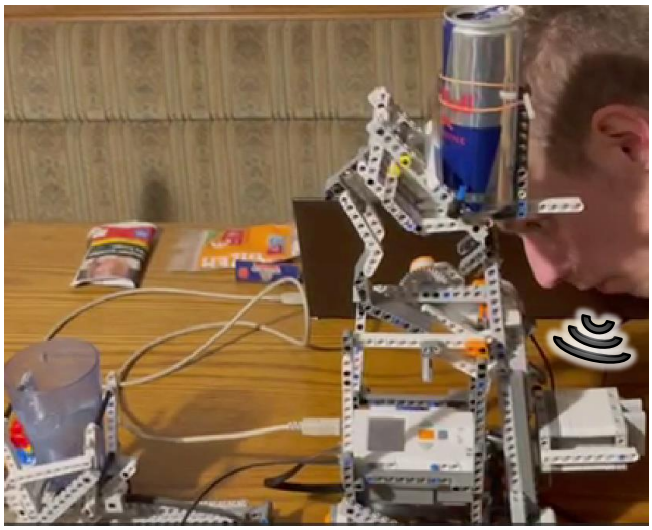


Abbildung 2: Per Sprachsteuerung wird der Prozess gestartet

Der Motor A des Trinkglas-Transportfahrzeuges wird per Sprachbefehl gesteuert. Dies ist links in Abbildung 2 zu sehen. Das Fahrzeug bewegt sich daraufhin in Richtung des Gießgerätes.

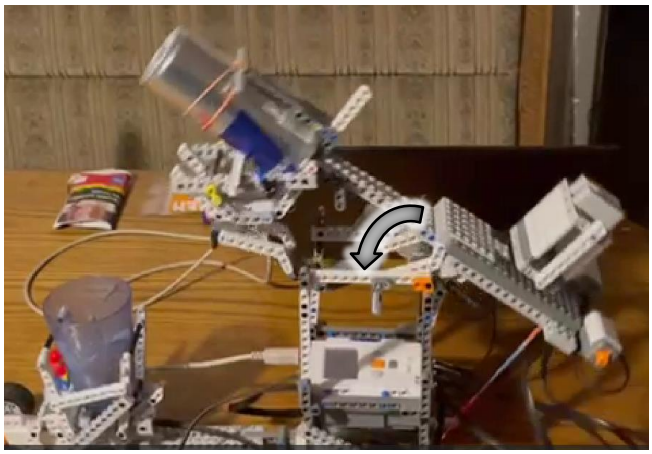


Abbildung 3: Kipparm kippt nach Berührung des Fahrzeugs

Löst das Fahrzeug bei Annäherung an das Gießobjekt den Tastsensor 1 aus, so stoppt es. Durch das Auslösen des Tastsensors 1 wird der Motor B des Gießgerätes gestartet und dreht sich bis zu einer bestimmten Drehzahl. Dabei treibt der schon zuvor genannte Motor eine Drehachse an, welche mit dem Gießarm mechanisch verbunden ist. Mit der Drehung des Motors wird somit der Gießarm bewegt. Dies ist in Abbildung 3 dargestellt. Je nachdem wie viel eingegossen werden soll, wird zeitlich verzögert der Knopf gedrückt.

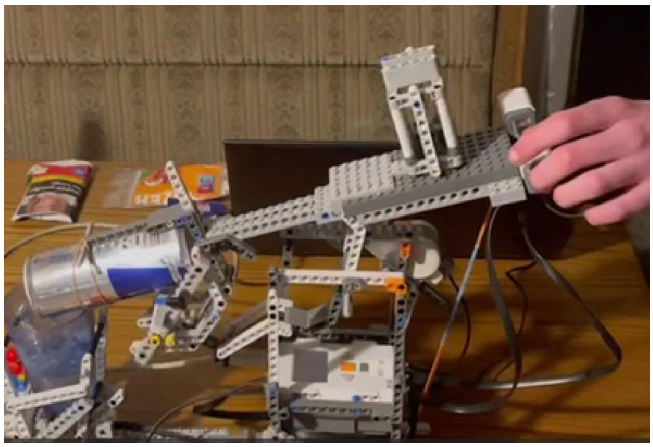


Abbildung 4: Nach dem Einfüllvorgang wird Knopf gedrückt

Wie in Abbildung 4 dargestellt ist, wird der Tastsensor 2 (8) gedrückt.

Abbildung 5: Kipparm kippt zurück nach Knopfdruck

Danach dreht sich der Motor B wieder in seine Ausgangsstellung zurück, wodurch auch der Gießarm nach hinten gekippt wird. Der letzte Schritt dieses Prozesses, wurde erfolgreich durchgeführt wie es in Abbildung 5 dargestellt ist.

C. Programmablaufplan

Zum besseren Verständnis ist der Programmablauf in Abb. 6 dargestellt. Wird die Sprachsteuerung nicht aktiviert, geschieht nichts. Sobald der Sprachsensor durch einen Sprachbefehl gestartet wird, startet der Motor des Trinkglas-Transportfahrzeuges mit dem Powerwert 15. Erst wenn das Fahrzeug durch Berührung den Tastsensor 1 aktiviert, kommt der Motor A zum Stehen. Durch die Aktivierung des Tastsensors 1 wird zudem der Motor B des Gießgerätes, welches mit einem Tacholimit von 360° programmiert wurde, gestartet. Sobald der Tastsensor 2 des Gießgerätes durch Knopfdruck aktiviert wurde, rotiert der Motor B zurück zu seiner Ausgangsstellung. Weiterhin wird der Motor A dadurch mit dem negativen Powerwert 15 angesteuert, wodurch das Fahrzeug sich wieder vom Gießgerät entfernt.

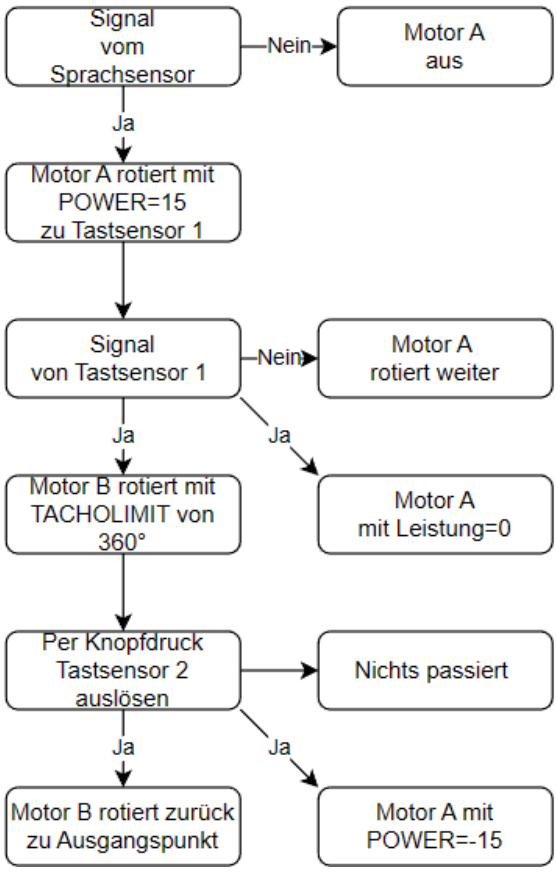


Abbildung 6: Programmablaufplan

D. Realisierbarkeit

Der entscheidende Nachteil dieses Gerätes ist, dass es halbautomatisch arbeitet. Das bedeutet, dass der Tastsensor am Gegengewicht des Gerätes (durch Fremdeinwirkung) gedrückt werden muss, wie in Abbildung 3 dargestellt ist. Die Lösung für dieses Problem ist eine Fernbedienung, die betätigt wird, wenn genügend Schüttgut vorhanden ist. Dazu muss der Tastsensor am Gegengewicht durch einen Infrarotsensor ersetzt werden.

III. Ergebnisdiskussion

Das größte Problem bestand darin, ein Gegengewicht für die relativ schwere volle Dose zu finden. Außerdem muss die Dose in einem bestimmten Winkel auf dem Kipparm befestigt werden. Dies wurde durch geschickte Montage von Teilen erreicht, wie in Abbildung 3 dargestellt ist. Des Weiteren muss die Dose so weit geneigt werden, dass möglichst viel von ihrem Inhalt entleert werden kann. Dabei gab es zum einen das Problem, dass die Dose in das Glas hineingekippt wurde und zum anderen, dass die Dose zu stark auf das Glas neigte, wenn der Kipparm sein Kipppunkt überschritt. Diese Probleme wurden mit Hilfe von Spanngummis behoben, wie in Abbildung 3 zu sehen ist [3].

IV. Zusammenfassung und Fazit

Per Sprachbefehl wird ein Trinkglastransporter zur Gießmaschine gefahren. Dabei wird der Tastsensor 2 an der Gießmaschine ausgelöst, der das Fahrzeug stoppt und den Motor B der Gießmaschine startet. Der am Kipparm befestigte Behälter wird durch das Gegengewicht ausbalanciert. Der Kipparm ist mechanisch mit dem Motor B verbunden. Durch äußere Einwirkung (Drücken des Tastsensors 2), kehrt der Kipparm in seiner Ausgangsstellung zurück. Wie bereits erwähnt ist die Grundidee der Funktionsweise der Vorrichtung an sich nicht schlecht, aber es gibt viele Möglichkeiten, die Vorrichtung zu optimieren. So könnte man z.B. einen Drucksensor unter dem Behälter anbringen, der bei Druckverlust (Behälter wird durch Entleeren des Inhaltes leichter) am Drucksensor diesen auslöst. Durch diesen Mechanismus könnte ein Signal ausgelöst werden, wenn der Behälter leer ist, so dass das Personal den Behälter nur dann wechselt, wenn es notwendig ist. Darüber hinaus könnten die Tastsensoren am Gießgerät durch Infrarotsensoren ersetzen werden, so dass sie über eine Fernbedienung gesteuert werden könnten. Dadurch müsste sich der Patient dem Gerät nicht mehr nähern (hygienisch besser), da zum einen der Sprachsensor nicht empfindlich genug ist und zum anderen der Patient selbst den Vorgang per Knopfdruck beenden muss. Eine weitere Möglichkeit wäre es das Gerät vollautomatisch zu konstruieren, was durch den Einbau eines Füllsensors möglich wäre. Wenn dann beim Entleeren ein bestimmter Füllstand im Trinkglas erreicht wird, erhält ein solcher Sensor ein Signal. Mit diesem Signal neigt sich der Kipparm sofort zurück.

In Zukunft wäre auch der Einsatz von künstlicher Intelligenz (KI) denkbar. In der Form, dass das Gerät aus dem Trinkverhalten des Patienten lernt, automatisch das Trinkglas entsprechend dem geschätzten Bedarf automatisch befüllt und dem Personal die Befüllung des Vorratsbehälters und den Bedarf vorschlägt.

Anhang

So wurde der in Abbildung 6 dargestellte Prozess in MATLAB mithilfe von [4] programmiert:

```
COM_CloseNXT('all')
handle=COM_OpenNXT();
COM_SetDefaultNXT(handle)
 while true
        OpenSound(SENSOR_1,'DB')
        sound = GetSound(SENSOR_1);
         a=sound;
       if a==1023
          motorA=NXTMotor('A','Power',-10);
          motorA.SendToNXT()
       end
       OpenSwitch(SENSOR_2);
        switchState = GetSwitch(SENSOR_2);
       CloseSensor(SENSOR_2);
        b=switchState;
        if b==1
          CloseSensor(SENSOR_1)
          motorA=NXTMotor('A','Power',0);
```

```
        motorA.SendToNXT()
        if b==1
            motorB=NXTMotor('B','Power',25);
            motorB.SendToNXT()
        end
    end
    OpenSwitch(SENSOR_3)
    switchState=GetSwitch(SENSOR_3);
    CloseSensor(SENSOR_3);
    c=GetSwitch(SENSOR_3);
    if c==1
        CloseSensor(SENSOR_2);
        motorB=NXTMotor('B','Power',-15);
        motorB.TachoLimit=360;
        motorB.ActionAtTachoLimit='brake';
        motorB.SendToNXT()
        motorA=NXTMotor('A', 'Power',10);
        motorA.SendToNXT()
    end
end
```

Abbildung 7: Verwendeter MATLAB-Programmcode

LITERATURVERZEICHNIS

[1] Firma Brita, “Wasserspender für Krankenhäuser” https://www.brita.de/wasserspender/branchen/kliniken-und-gesundheitswesen

[2] Statista, “Prognostizierter Bedarf an stationären und ambulanten Pflegekräften* in Deutschland bis zum Jahr 2035” https://de.statista.com/statistik/daten/studie/172651/umfrage/bedarf-an-pflegekraeften-2025/

[3] H. Vogel, “Gerthsen Physik” 18. Auflage, Springer Verlag Berlin, 1995.

[4] Projektseminar Elektrotechnik/Informationstechnik (LEGO Mindstorms), “ Matlab-Handbuch” https://elearning.ovgu.de/course/view.php?id=178

Automatisierter Getränkebefüller aus Lego

Märkisch Jannis, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Abstract*- Wir leben in einer Gesellschaft, in der vieles für die Menschen, die darin leben, geregelt ist. Aber nicht alle Menschen haben dieses Glück. Menschen mit körperlichen Einschränkungen können oft nicht alles so machen wie andere. So leben allein in Deutschland, laut der [1] ,7,8 Millionen Menschen mit körperlichen Einschränkungen. Um diesen Menschen zu helfen, überlegen sich die Menschen immer wieder neue Dinge, um auch ihnen das Leben so einfach wie möglich zu machen. So sind wir auch auf die Idee eines Getränkeabfüllautomaten gekommen, um diesen Menschen zu helfen und ihnen das Leben zu erleichtern**.

I. Einleitung

DIE Idee, die wir umgesetzt haben, soll diesen Menschen helfen, indem sie ihnen Getränke einschenken. So wären die Menschen in der Lage, sich ihr Getränk selbst oder ohne fremde Hilfe einzuschenken.So können sie sich jederzeit etwas zu trinken zubereiten. Die einzige Voraussetzung ist das Wechseln der Flasche, das von einer externen Person durchgeführt werden muss.

II. Vorbetrachtungen

In der Wissenschaft kommen immer mehr Getränkespender auf dem Markt. Eine Firma die vorne in der Entwicklung mit spielt, ist die Firma BRITA. Sie entwickeln schon seit längerer Zeit, Wasserspender speziell für den Einsatz in Krankenhäuser [2]. Die von mir entwickelte Lösung beinhaltet ein kleines Auto und ein Gerät mit Kippmechanismus. Speziell auch für Kinder mit körperlichen Einschränkungen hat die Idee auch ein Spaß- bzw. Ablenkungsfaktor. Somit macht man durch, dass fahrende Auto den Kleineren eine kleine Freude und lässt eine kleine Ablenkung zu.

A. Fahrzeug

Als erstes wurde das Fahrzeug gebaut, das für den Transport des Getränkebehälters bestimmt ist. Dabei wurde die erste Idee umgesetzt (Abb. 1). Die Schwierigkeit bestand darin, die Höhe bzw. den Abstand zum Behälter so einzustellen, dass das Getränk genau im Becher landet und nicht daneben.

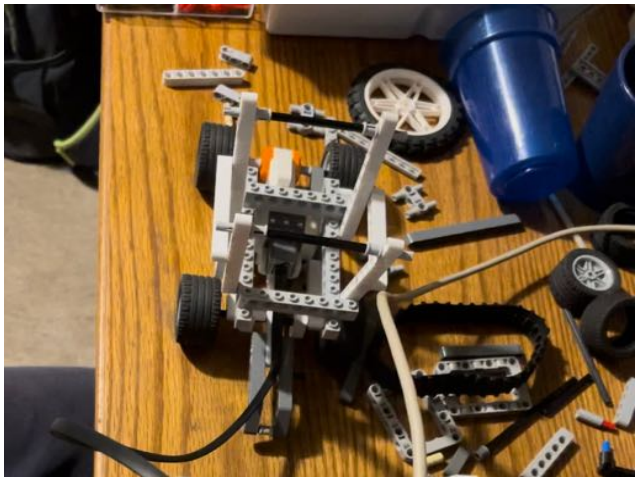


Abb. 1 Fahrendes Auto

B. Fahrzeug

Als erstes wurde das Fahrzeug gebaut, das für den Transport des Getränkebehälters bestimmt ist. Dabei wurde die erste Idee umgesetzt (Abb. 1). Die Schwierigkeit bestand darin, die Höhe bzw. den Abstand zum Behälter so einzustellen, dass das Getränk genau im Becher landet und nicht daneben.

C. Grundstruktur

Es gab einige Probleme mit der Grundstruktur, mit einigen Umstrukturierungen der Grundstruktur. Die ursprüngliche Idee war eine Art „Flaschenzug“. Der eine Flasche hochzieht und darüber kippt. Doch nach einigen Versuchen wurde schnell klar, dass die Umsetzung in der zur Verfügung stehenden Zeit nicht zu realisieren war. So kam die Idee einer Zahnradkonstruktion. Aber auch diese wurde nach einigen Versuchen wieder verworfen. Es wurde eine Plattform gebaut (Abb. 2), die aber nicht funktionierte und so entstand die neue Plattform (Abb. 3). Eine volle Dose war jedoch zu schwer für die Zahnradkonstruktion, so dass die Zahnräder nach unten gedrückt wurden und der Motor nicht mehr in der Lage war, die Zahnräder anzutreiben. Daraus entstand die schließlich umgesetzte Idee eines Gerüstes nach dem Gegengewichtsprinzip. Dabei wurde auch die zuvor entworfene Plattform wiederverwendet. Somit besteht die Grundstruktur aus der Plattform (Abb. 3) und dem Rahmen mit dem Gegengewicht, an dem auch der Lego-NXT-Stein befestigt wurde. Wobei der Lego-NXT-Stein auch leichte Stabilitätsprobleme verursachte. Dies konnte durch einfaches Verschieben des Steins behoben werden.

DOI: 10.24352/UB.OVGU-2023-032

Abb. 2 Alte Plattform

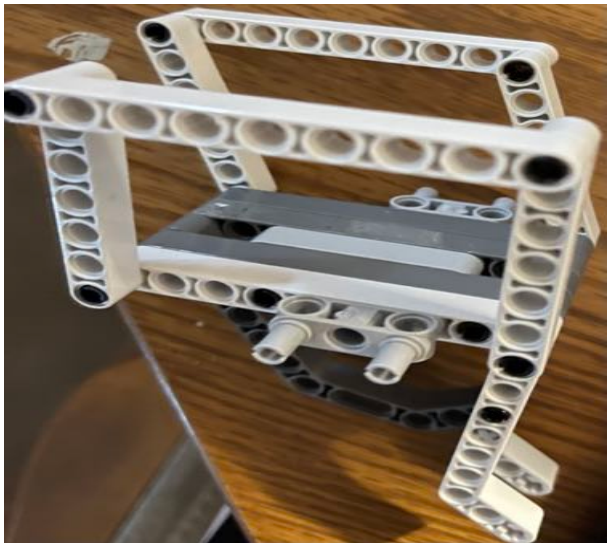


Abb. 3 Neue und verbesserte Plattform

D. Ideenfindung

Die Idee hinter dem Getränkeausgießer war, wie bereits erwähnt, den Menschen zu helfen bzw. das Leben zu erleichtern. Ein Ziel war es auch, die Konstruktion so klein bzw. kompakt wie möglich zu halten. Damit man es auch auf einen kleinen Tisch stellen kann, wenn nicht unbedingt ein „großer“ Tisch zur Verfügung steht. Das Gerät ist natürlich auch für normale bzw. alltägliche Situationen einsetzbar. So ist es auch auf jeder Party ein Hingucker. So ist auch die Idee entstanden. In geselliger Runde sind schon viele Ideen entstanden. So ist auch die Idee zum Getränkeeinschänker entstanden.

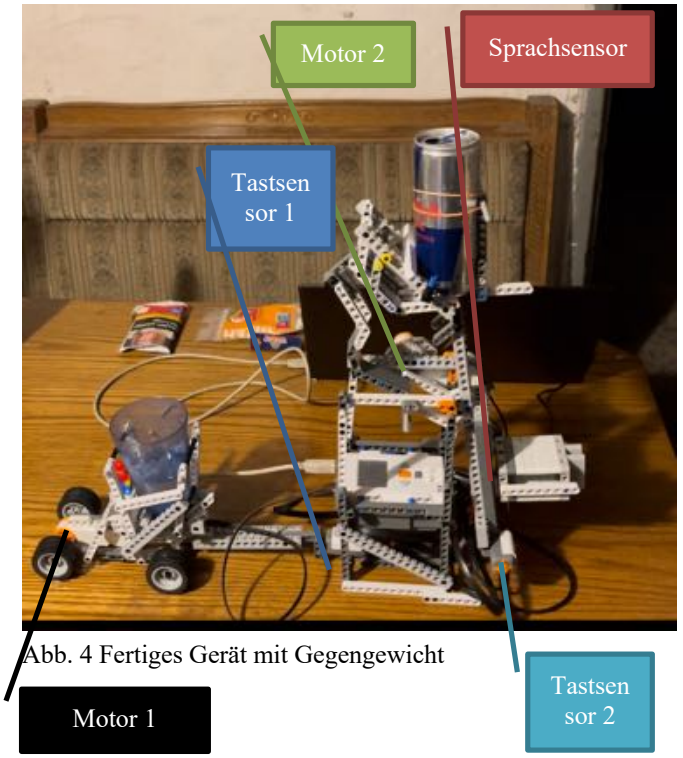


Abb. 4 Fertiges Gerät mit Gegengewicht

III. Bau und Funktionsweise

Die Bedienung ist so einfach wie möglich gehalten. Das Gerät wird über einen Sprachsensor gestartet. Dieser Sensor misst die Lautstärke im Raum, d.h. durch lautes Sprechen wird der Sensor ausgelöst. Sobald der Sensor ausgelöst wurde, fährt ein kleines Auto los, auf dem der Behälter steht, in den das Getränk eingefüllt werden soll. Das Auto berührt dabei nach kurzer Zeit einen Tastsensor. Nach dem Berühren des Tastsensors stoppt das Auto und der Kippmechanismus wird ausgelöst. Das Getränk beginnt zu fließen. Sobald die gewünschte Menge des Getränks im Glas ist, kann der Einfüllvorgang durch einen weiteren Tastsensor unterbrochen bzw. beendet werden. Nach dem Betätigen des Tastsensors fährt die Flasche wieder zurück und das Fahrzeug fährt wieder ein Stück vor, um ein bequemes Herausnehmen des Bechers zu ermöglichen. Danach kann der Vorgang so oft wiederholt werden, bis die Flasche mit dem Getränk leer ist. Sie muss nur ausgetauscht werden. Danach ist das Gerät wieder einsatzbereit.

Erklärung der Funktionsweise:

Bei der Idee bzw. bei der Umsetzung ist aufgefallen, dass es sich bei dem Gerät um ein „halbautomatisches“ Modell handelt. „Halbautomatisch“ in dem Sinne, dass man den Taster 2 drücken muss, um den Vorgang abzubrechen (siehe Abb. 1 und Abb. 4), eine völlig autonome Bedienung des Gerätes ist also noch nicht möglich. Eine Lösung für dieses Problem wäre die Möglichkeit, den Kippmechanismus durch externe Elemente wie Fernbedienungen oder die Steuerung durch eine externe Handy-Applikation. In beiden Fällen müsste ein zusätzlicher Sensor am Sensor angebracht werden. Für die Fernbedienung müsste man einen Infrarotsensor (der mit der Fernbedienung kompatibel ist). Für die Handy-App könnte man zum einen,

einen programmierbaren Sensor anbringen, der bei richtiger Programmierung wie der Tastsensor den Prozess manuell stoppt. Zum anderen könnte ein Mechanismus gebaut werden, der ebenfalls mit einer Handy-App funktioniert und einen dritten Motor antreibt, der mit dem Tastsensor 2 betätigt wird (siehe Abb. 1 und Abb. 4).

IV. Programm

Das Programmieren wurde in einzelnen Schritten gemacht. Der Anfang des Programmierens war die Einstellung des Autos.

```
COM_CloseNXT('all')
handle=COM_OpenNXT()
COM_SetDefaultNXT(handle)


while true
    OpenSound(SENSOR_1);
    a=sound;
    if a==1023
     motorA=NXTMotor('A','Power',-15);
        motorA.SendToNXT()
    end
```

Als erstes würde der Sprachsensor bzw. der Sensor die Lautstärke misst eingestellt. Sobald dieser den durch laute Geräusche bzw. Sprechen ausgelöst wird. Sobald die Bedingung erfüllt ist, startet den Motor A damit, das Auto ranfährt.

Danach wurde der Motor 2 (Abb. 4) programmiert und eingestellt.

```
  OpenSwitch(SENSOR_2);
    switchState=GetSwitch(SENSOR_2);
    b=switchState
    if b==1
        CloseSensor(SENSOR_1);
        motorA=NXTMotor('A','Power',0);
        motorA.SendToNXT()
    end
    if b==1
        motorB=NXTMotor('B','Power',50)
        motorB.SendToNXT()
    end
```

Dabei würde zuerst der Tastsensor 1 eingestellt. Sobald dieser betätigt bzw. ausgelöst wird, setzt sich der Motor A auf 0 und bleibt somit stehen. Gleichzeitig wir der Motor B gestartet und der Prozess des einkippen wird gestartet.

Somit kippt das Getränk jetzt ein. Jetzt fehlt nur noch der letzte Schritt, das Zurückfahren des Getränkes und das Vorfahren des Autos.

```
OpenSwitch(SENSOR_3)
    switchState=GetSwitch(SENSOR_3)
    c=GetSwitch(SENSOR_3)
    if c==1
        CloseSensor(SENSOR_2);
        motorB=NXTMotor('B','Power',-50);
        motorB.TachoLimit=720;

motorB.ActionAtTachoLimit='brake';
        motorB.SendToNXT()

motorA=NXTMotor('A','Power',10);
        motorA.TachoLimit=360;
        motorA.SendToNXT()
    end
end
```

Wie hier zu sehen ist, wird auch hier zuerst der Sensor, in diesem Fall ein weiterer Tastsensor (Abb. 4), geöffnet. Sobald dieser Tastsensor betätigt wurde, wird der zweite Tastsensor geschlossen. Nach Betätigung des Tastsensors 2 fährt Motor B mit einer Tachobegrenzung von 720 (entspricht 2 Umdrehungen) rückwärts, bis das Gegengewicht so groß ist, dass die Plattform (Abb. 2) wieder in die Ausgangsposition zurückfährt, während Motor A wieder vorwärtsfährt und nach einer Motorumdrehung zum Stillstand kommt. Danach ist das Programm beendet und beginnt von neuem.

V. Ergebnissdiskussion:

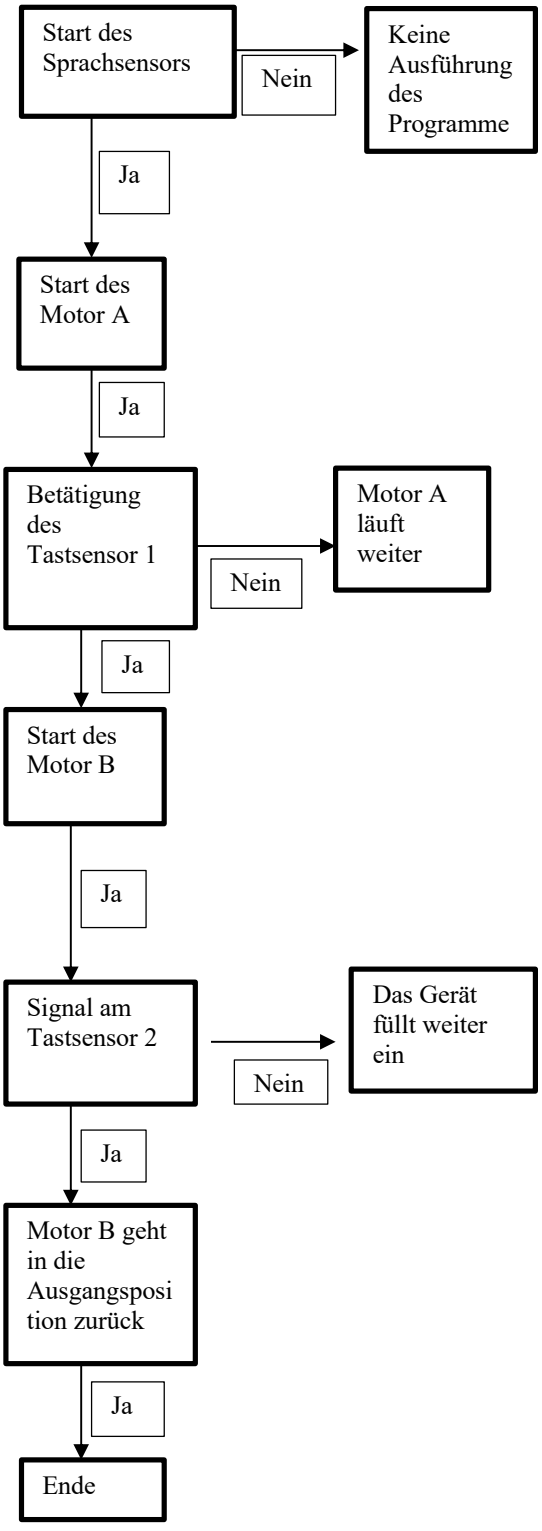
Eines der größten Probleme war es, das richtige Gegengewicht für die Dose zu finden. Bei einer vollen Dose muss das Gegengewicht deutlich größer sein als bei einer halb vollen Dose. Daraus ergab sich auch ein Problem bei der Einstellung des Motors 2 (siehe Abb. 1). Bei einer halbvollen Dose ist die Motorleistung deutlich zu hoch und die Dose würde auf den Behälter fallen und evtl. den ganzen Behälter umstoßen bzw. durch den Aufprall den Behälter verschieben und somit die ganze Flüssigkeit daneben laufen.

VI. Literaturverzeichniss

[1] **Statistisches Bundesamt:https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Gesundheit/Behinderte-Menschen/_inhalt.html (letzter Aufruf:23.02.2023)**

[2] Firma Brita: https://www.brita.de/wasserspender/branchen/kliniken-und-gesundheitswesen (letzter Aufruf:23.02.2023)

VII. ANHANG(ABLAUFPLAN)



Die LEGO-Sortiermaschine mit Klappen

Finn Sackewitz, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Im Rahmen des LEGO-Praktikums wurde eine Sortiermaschine für LEGO-Steine entwickelt. Dazu wurde ein LEGO-Gestell mit 2 Motoren gebaut. Diese Motoren können insgesamt 4 Türen steuern, wodurch eine Sortierung der LEGO-Steine nach Farben möglich wurde. Die Farberkennung wird durch eine Webcam ermöglicht. Wird nun ein LEGO-Stein in die Sortiermaschine gelegt, so wird die Farbe erkannt und die falschen Bahnen werden blockiert. Der LEGO-Stein rutscht nun auf einer bestimmten Bahn in den dafür vorgesehenen Auffangbehälter. Dieses Funktionsprinzip hat viele Anwendungs- und Erweiterungsmöglichkeiten. Mit diesem prinzipiellen Aufbau lassen sich vielerlei Dinge sortieren und somit in Automatisierungsprozesse eingliedern.**

***Schlagwörter*—Automatisierung,Farberkennung,LEGO-Mindstorms,sortieren,Webcam**

I. Einleitung

JEDER der schon einmal mit LEGO-Steinen gebaut hat, kennt es: Es herrscht großes Durcheinander. Oft landen alle LEGO-Steine einfach gesammelt in einer großen Kiste. Möchte man nun etwas Neues bauen, ist man meist eine sehr lange Zeit damit beschäftigt, die passenden LEGO-Steine zu suchen. Für den Menschen ist es sicherlich sehr einfach diese zu sortieren. Doch wer hat dazu die Motivation? Diese Problemstellung lässt sich auch aus den Grenzen der LEGO-Welt heraustragen. Egal wo der Mensch ist, gibt es Unordnung.
Dieser Problemstellung soll sich die LEGO-Sortiermaschine stellen.

II. Vorbetrachtungen

Solche Technologien wie die LEGO-Sortiermaschine existieren bereits und werden auch sehr Häufig in der Industrie angewandt. Fast jeder automatisierte Prozess benötigt einen Sortiermechanismus.

A. Beispiele

Ein erstes Beispiel dafür wäre die industrielle Mülltrennung. Gerade im Zuge des Klimawandels ist eine fachgerechte Mülltrennung sehr wichtig. Dieser Herausforderung stellte sich eine Gruppe aus Studenten im Auftrag von Remondis. Hierbei wurde eine automatische Mülltrennung mit künstlicher Intelligenz entwickelt. Im Inneren arbeitet dabei eine Kamera, die den Müll optisch erfasst. Daraufhin wird das Bild durch einen Algorithmus auf Basis künstlicher Intelligenz ausgewertet, wodurch der Müll in den richtigen Behälter sortiert werden kann [1]. Eine weiteres Beispiel ist hier eine Sortiermaschine für Bälle [2]. Hierbei handelt es sich um eine einfache Sotiermaschine für farbige Bälle. Die Erkennung der Farbe wird hier ebenfalls über eine Kamera sichergestellt. Durch ein Pappgestell und ein Gefälle wird schließlich der Weg der Bälle festgelegt und durch Türen kann der richtige Weg ausgewählt werden. Dieses kleine Projekt des Linkedin Users stellt die grundlegende Inspiration für dieses Projekt der LEGO-Sortiermaschine dar.

DOI: 10.24352/UB.OVGU-2023-033

B. Funktionsprinzip der LEGO-Sortiermaschine

Im Grunde handelt es sich bei der LEGO-Sortiermaschine um einen Mechanismus, welcher durch Öffnen und schließen von Türen LEGO-Steine in eine bestimmte Richtung lenkt. Diese Richtung wird mittels einer Webcam durch Erkennung einer Farbe bestimmt. Wird nun z. B. die Farbe Rot erkannt, so werden z. B. die Türen eins und drei betätigt. Diese Türen werden durch LEGO-NXT-Motoren gesteuert und sind mittels mechanischer Mittel verbunden. Die LEGO-Steine werden durch die Schwerkraft zum Rutschen gebracht.

III. Entwicklungsprozess

Im Hauptteil werden der Aufbau, die Entwicklung und die Problemlösung des Projekts beleuchtet. Das Ziel dieses Abschnittes ist es, die Funktionsweise und die Entwicklung der LEGO-Sortiermaschine chronologisch darzustellen und zu erklären.

A. Aufbau

Die LEGO-Sortiermaschine besteht im Wesentlichen aus einem LEGO-Rahmen und einer darüber liegenden Konstruktion aus Pappe und Papier. Die LEGO Konstruktion dient als Träger und zur Steuerung der Türen. Die Türen sind in drei Ebenen unterteilt. Auf der ersten Ebene befindet sich eine Tür. Die erste Tür dient dazu, den LEGO-Stein festzuhalten, während die Farbe erkannt wird. Die zweite Tür, die sich ebenfalls in der zweiten Ebene befindet, dient der ersten Sortierunterscheidung. In der dritten Ebene befinden sich zwei Tore. Diese sorgen schließlich für die insgesamt 4 Sortiermöglichkeiten. Des Weiteren sind in dieser LEGO-Konstruktion zwei Motoren und zwei Tastsensoren verbaut. Die Motoren dienen zur Betätigung der insgesamt vier Türen und die Tastsensoren zur Steuerung des Programms. Die Motoren sind hier mechanisch mit den Türen verbunden. Um das „Rutschen“ der LEGO-Steine in der Sortiermaschine zu gewährleisten, muss ein Gefälle in die Konstruktion eingebaut werden. Auf die LEGO-Konstruktion wird außerdem die Webcam angebracht, um die Farbe der LEGO-Steine zu erkennen. An das Ende der LEGO-Sortiermaschiene werden kleine Papierbehälter aufgestellt, um die LEGO-Steine aufzufangen.

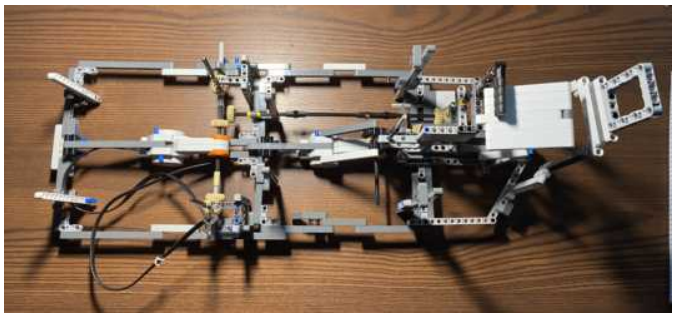


Abbildung 1. LEGO-Konstruktion der Sortiermaschine

B. Konstruktions-Entwicklung

1) LEGO-Gestell: Die Entwicklung der LEGO-Sortiermaschine begann mit dem Bau der Konstruktion. Dabei musste vor allem auf Stabilität und Funktionalität geachtet werden. Der Bau begann am ersten Motor und setzte sich am zweiten Motor fort. In der Zwischenzeit wurde durch regelmäßige Tests sichergestellt, dass auf diese Konstruktion auch eine Pappauflage passt und dass diese auch fest in ihrer Position bleibt. So konnte auch sichergestellt werden, dass die Steigung groß genug war. Es musste außerdem genau auf die richtige Größe geachtet werden, damit die Mechaniken und Geräte genug Platz haben, um sich zu bewegen. Die Oberfläche muss eine bestimmte Größe haben, damit die LEGO-Steine darauf gleiten können.

2) Mechanik: Im zweiten Schritt wurde die Mechanik der LEGO-Sortiermaschine konstruiert. Diese basiert im Großen und Ganzen auf einer Übertragung des Drehmoments der Motoren mittels Achsen und Zahnrädern. Die Motoren sind zentral angeordnet, um einen möglichst großen Einsatzbereich zu gewährleisten. Dies ist gut in der Abbildung 1 nachvollziehbar. Der in der Abbildung 1 linke Motor steuert dabei die erste und dritte Ebene der Türen und der rechte Motor die zweite Ebene. Da der linke Motor insgesamt drei Türen steuert, konnte auf einen dritten oder vierten Motor verzichtet werden. Hier musste vor allem darauf geachtet werden, dass jedes Zahnrad genau in das andere übergreift. Es darf hier keine (oder nur sehr geringe) Fehler geben, da die Rückdrehung der Türen sonst nicht wieder in ihren Ausgangszustand geht. In diesem Fall wären mehrfache Durchläufe nicht möglich.

3) Farberkennung: Die Entwicklung der Farberkennung begann mit Versuchen, den LEGO-Farbsensor zu verwenden. Dies erwies sich schnell als kompliziert, da der Farbsensor Farben nur aus sehr kurzer Entfernung erkennen kann. Aus diesem Grund wurde eine Webcam verwendet. Diese gewährleistet eine größere Erkennungsentfernung und weitreichende Anpassungsmöglichkeiten. Die Webcam wurde am rechten oberen Ende der LEGO Konstruktion angebracht und auf die erste Tür ausgerichtet. Die LEGO Steine werden dann genau vor der Tür platziert, wo ihre Farbe von der Webcam erkannt werden kann.

4) Pappauflage: Als letztes wurde die entgültige Pappauflage entwickelt. Im Verlaufe der Entwicklung wurde immer eine Testpappauflage verwendet, um die Funktionsweise der jeweiligen Zwischenschritte sicherzustellen. Bei der Pappauflage wurde darauf geachtet, dass kein LEGO-Stein an den Rändern hängen bleibt und dass sie exakt auf das Gestell passt. Außerdem mussten Löcher für die Türen und Stützen hineingeschnitten werden. Der schematische Aufbau dieser Auflage ist in Abbildung 2 zu sehen.

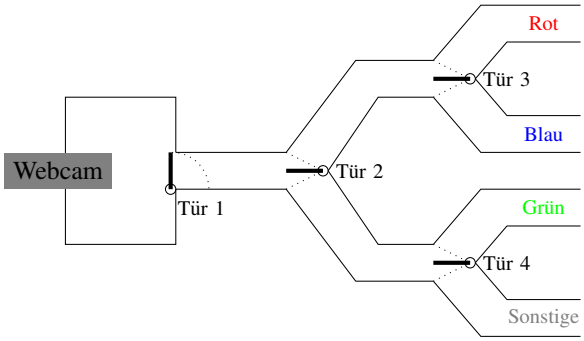


Abbildung 2. Schematischer Aufbau der Sortiermaschine

C. Programmierungs - Entwicklung

Nach dem Bau der Konstruktion ging es darum, die Funktionsweise der Sortiermaschine zu implementieren bzw. zu entwickeln. Der allgemeine Programmablauf ist in Abbildung 3 dargestellt und wird im Folgenden kurz erläutert. Die Ansteuerung der einzelnen NXT-Komponenten wurde durch eine Toolbox der RWTH Aachen ermöglicht [3].

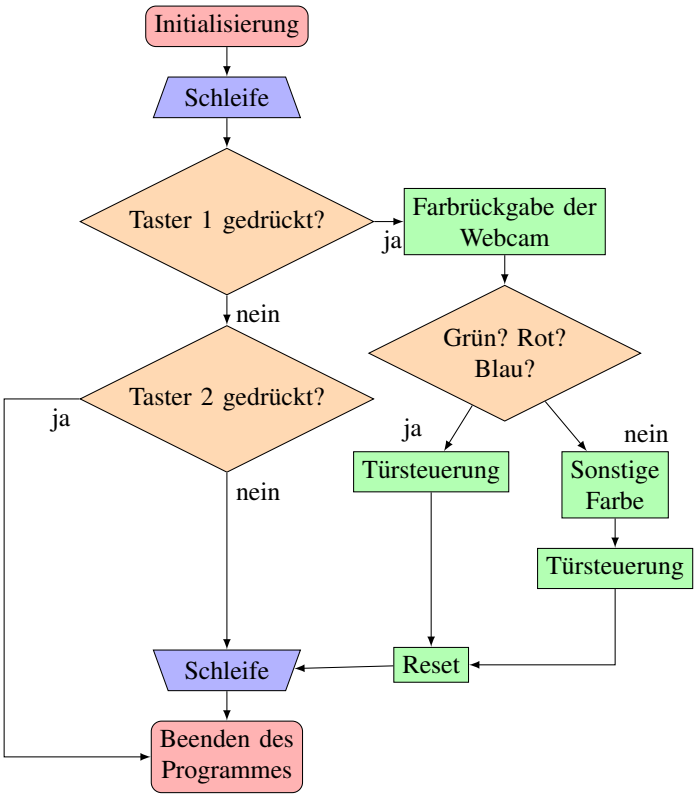


Abbildung 3. Allgemeiner Programmablaufplan der LEGO-Sortiermaschine

Mit der Initialisierung des Programms wird eine Schleife gestartet. Die Schleife läuft zunächst endlos weiter. Erst wenn

einer der beiden Taster betätigt wird, wird eine Aktion ausgelöst. Durch Betätigen des zweiten Taster wird das Programm lediglich beendet. Betätigt man den ersten Taster, wird ein LEGO-Stein sortiert. Als erstes wird die Farbe durch die Webcam erkannt. Abhängig von dieser Erkennung werden die nächsten Schritte ausgeführt. Wird Grün, Blau oder Rot erkannt, so wird eine bestimmte Kombination von Türen geöffnet. Diese ist nicht für jede Farbe gleich, wie es der Programmablaufplan vermuten lässt. Dies wurde hier aus Platzgründen vereinfacht dargestellt. In dem Fall, dass keine der abgefragten Farben erkannt wird, wird der LEGO-Stein in eine für sonstige Farben vorgesehene Kiste sortiert. Nachdem einer dieser vier Fälle eintrat, werden alle Türen wieder auf ihre Ausgangsposition gesetzt. Die Schleife wartet nun auf eine Eingabe über einen der beiden Taster.
Das Programm wurde nicht als Ganzes geschrieben. Es wurden verschiedene Unterprogramme geschrieben, die die jeweiligen Teilprobleme lösen. Diese wurden dann am Ende in einem Hauptprogramm zusammengeführt. Im Folgenden werden die einzelnen Teilprogramme erläutert.

1) Türensteuerung: Die Türen werden durch eine Drehung des Motors bewegt. Diese Drehung wird durch die *Power* des Motors bestimmt. Umso höher die *Power* ist, desto schneller dreht sich der Motor. Weist man dem Motor eine negative Power zu, so dreht dieser sich in die entgegengesetzte Richtung. Nun soll sich der Motor aber nicht einfach nur drehen, sondern bei einem bestimmten Punkt stoppen. Das wird durch das *TachoLimit* des Motors gesteuert. Dazu wird das *TachoLimit* auf die gewünschte Gradzahl gesetzt und der Motor dreht sich bis zu diesem Punkt.

2) Webcamsteuerung: Die Webcam lässt sich relativ einfach in Matlab einbinden. Durch den Befehl *getsnapshot*() wird das Bild der Webcam eingelesen. Das Bild wird als Matrix gespeichert. In dieser Matrix befinden sich dann die jweiligen RGB-Werte des dazugehörigen Pixels. So lässt sich durch Auswählen eines bestimmten Bereiches die Farbe eines LEGO-Steins bestimmen. Um eine möglichst fehlerfreie Messung durchzuführen, wurde der Mittelwert über den eingelesenen Bereich gebildet.

D. Probleme

Während des Entwicklungsprozesses traten vielfältige Probleme auf. Diese mussten gelöst werden. Das wohl größte Problem der LEGO-Sortiermaschine war wohl die mechanische Stabilität. Es musste akribisch darauf geachtet werden, dass z. B. alle Zahnräder ineinander greifen. Da die LEGO-Stäbe nur eine begrenzte Festigkeit haben, war dies nicht immer so einfach. Dies ließ sich durch eine geschickte Konstruktion bewältigen. Ein weiteres Problem stellte die Farberkennung der Webcam dar. Die Webcam erkennt, je nach Belichtung, immer einen anderen RGB-Wert, obwohl das selbe Objekt vor der Webcam liegt. Dadurch konnte es passieren, das z. B. ein roter LEGO-Stein nicht als solcher erkannt wurde und in die Kategorie „Sonstige“ einsortiert wurde. Wichtig ist, dass das eingelesene Feld der Webcam immer genau den LEGO-Stein trifft. Dies musste leider immer manuell sichergestellt werden. Auch die Belichtung sollte immer durch eine externe Lichtquelle sichergestellt werden. Dies geschah z. B. mit einer Schreibtischlampe. Ein weiteres Problem war die Kontrolle der LEGO-Steine. Damit ist gemeint, dass ein LEGO-Stein nicht durch eine geschlossene Tür oder über eine Bande rutscht. Dieses Problem kann nicht vollständig vermieden werden. Es wird jedoch durch eine gut angepasste Pappauflage stark minimiert.
Weitere Probleme stellen die Erkennung von anderen Farben (Farben, die nicht rot, grün oder blau heißen), große LEGO-Steine und eine automatisierte Sortierung dar.

E. Ausbaumöglichkeiten

Die LEGO-Sortiermaschine bietet große Ausbaumöglichkeiten. Zum einen lassen sich die zu erkennenden Farben erweitern auf z. B. 8 oder sogar 16. Dafür müsste man eine bis zwei weitere Ebenen von Türen einbauen. Dies ließe sich auch mit LEGO-NXT lösen, da bei dem gegenwärtigem Projekt bisher nur zwei Motoren verwendet wurden. Ein Motor der bisher verwendeten Motoren könnte schließlich noch eine weitere Ebene von Türen steuern. So könnten mit dem dritten Motor noch zwei zusätzliche Ebenen gesteuert werden.
Noch eine weitere Ausbaumöglichkeit stellt eine Automatisierung der Hineingabe von LEGO-Steinen in die Sortiermaschine dar. So könnte diese aus vielen Steinen immer einen LEGO-Stein auswählen und diese in die Sortiermaschine hineingeben. Dazu müsste allerdings die Farberkennung für bewegte Objekte weiter angepasst werden.

IV. Ergebnisdiskussion

Als Ergebnis ist eine Sortiermaschine für vier Farben entstanden, welche aber noch vielfältige Erweiterungsmöglichkeiten bietet. Der Mechanismus lässt sich über zwei Taster steuern und ist unendlich oft wiederholbar. Um die Sortiermaschine allerdings im alltäglichen Leben einzusetzen bedarf es noch weitere Optimierungen.

V. Zusammenfassung und Fazit

Das Ziel des Projektes war es, eine Maschine zu bauen, welche zuverlässig LEGO-Steine nach Farben sortieren kann. Dieses Ziel wurde erreicht. Dennoch gibt es noch weitreichende Optimierungs- und Erweiterungsmöglichkeiten für diese Sortiermaschine. Dafür sind Punkte wie Automatisierung und Erweiterung der Farberkennungsmöglichkeiten zu nennen. Aufgrund der begrenzten Zeit und Ressourcen ist dies dennoch ein gutes Ergebnis was am Ende des Projektseminars 2023 entstanden ist. Man muss hierbei auch bedenken, dass die LEGO-Sortiermaschine in Einzelarbeit entstanden ist, im Gegensatz zu anderen Projekten, die in Gruppenarbeit entstanden sind.

Literaturverzeichnis

[1] Remondis: automatische Mülltrennung durch künstliche Intelligenz
https://kreativgesellschaft.org/cross-innovation-hub/cases/remondis-automatische-mulltrennung-ki/
[2] Linkedin: Beitrag von Jeff He
https://www.linkedin.com/posts/jeff-he-00ba13200_diyprojects-hardware-activity-6888684109066043392-7HXC/
[3] Mathworks: RWTH Aachen–Mindstorms NXT Toolbox
https://de.mathworks.com/matlabcentral/fileexchange/18646-rwth-mindstorms-nxt-toolbox

Der Stabelgabler

Jamie Georg, ETIT
Otto-von-Guericke-Universität Magdeburg

***Abstract*— Im Rahmen des Projektseminars Elektrotechnik und Informationstechnik wurde die Aufgabe gestellt, einen Automaten zu entwerfen, der Lego Mindstorms (Abbildung [1]) als strukturelle Basis verwendet. Dieser sollte mit Hilfe von Matlab als Programmiersoftware umgesetzt werden. Der in diesem Projekt realisierte Automat ist ein selbstfahrender Gabelstapler. Im Folgenden wird sowohl auf die Hard- und Softwarekonzepte als auch auf die Umsetzung, die dabei aufgetretenen Probleme und deren Lösungen eingegangen. Abschließend werden die Einsatzmöglichkeiten und mögliche Verbesserungen diskutiert.**

***Schlagwörter*— autonom, Gabelstapler, innerbetrieblich, Roboter, Transport**

Abbildung [1]: NXT-Kontrollstein

I. Einleitung

Die Industrialisierung war vor 250 Jahren (in Großbritannien) der Beginn einer Epoche, in der körperliche Arbeit durch Maschinen ersetzt wurde und dadurch eine exponentielle Steigerung der Produktion möglich wurde. In den letzten Jahrzehnten ist diese Automatisierung bereits in die nächste Entwicklungsphase eingetreten, in der der Mensch vollständig von der Ausführung entfernt wird und die Maschinen durch komplexere Programmierung und künstliche Intelligenz in der Lage sind, Routinen selbstständig auszuführen und sogar teilweise autonom auf irreguläre Ereignisse zu reagieren.

Ein Anwendungsbereich, in dem die Automatisierung bereits weit fortgeschritten ist, ist die Lagerhaltung (Sortieren und Transportieren von Waren). Bisher wurden für den innerbetrieblichen Transport von schweren Lasten Gabelstapler eingesetzt, die jedoch von Menschen bedient werden mussten. Hier setzt der Automat dieses Projektes an, der, wie in Abbildung [2] gezeigt wird, den Menschen ersetzen soll.

Abbildung [2]: Autonomer Gabelstapler der Marke Agilox

II. Vorbetrachtungen

Um den Gabelstapler automatisieren zu können, werden im Folgenden die Funktionen definiert, die für den Ersatz des Menschen angepasst bzw. neu implementiert werden müssen.

Die beiden Hauptfunktionen, die automatisiert werden müssen, sind die Orientierung im Raum und das Be- und Entladen der richtigen Güter. Dabei stellt die Orientierung das weitaus größere Problem dar. Denn der Automat muss den optimalen Weg zu seinen Stationen finden und zufällige oder unnötige Wege, die den Betrieb und damit die Effizienz mindern, müssen vermieden werden. Die Orientierung kann auf unterschiedliche Art und Weise erfolgen, z.B. optisch oder durch Positionsbestimmung (GPS). In diesem Projekt wurde aufgrund technischer Einschränkungen eine optische Variante gewählt, bei der der Stapler einer Linie folgt, um entlang einer bestimmten Farbe zur entsprechenden Abteilung zu gelangen.

Im nächsten Schritt muss die Sensorik und Programmierung der Lenkung an die Antriebsart des Fahrzeugs angepasst werden, damit der Roboter der Führungslinie exakt folgen kann. Der Antrieb bietet hier verschiedene Möglichkeiten. Zum einen können Antrieb und Lenkung kombiniert werden, indem beide Räder von separaten Motoren angesteuert werden. Dadurch kann die Auslenkung des Gabelstaplers durch den unterschiedlichen Schub der beiden Motoren verändert werden, ähnlich wie bei einem Panzer. Eine andere Möglichkeit besteht darin, den Antrieb auf ein oder mehrere schwenkbare Hinterräder zu verlagern, die dann die Richtung des Staplers ändern. Diese Technik wird bereits bei konventionellen Gabelstaplern eingesetzt.

Sobald sich der Roboter vor dem gewünschten Regal positioniert hat, muss er das gesuchte Modul erkennen.

DOI: 10.24352/UB.OVGU-2023-034

Hierfür bietet sich die Warenerfassung mittels Codierungen an, wie sie bereits im Verkauf in Form von Barcodes oder QR-Codes eingesetzt werden. Daraus ergibt sich, dass mindestens zwei Sensoren benötigt werden, einer für die Linienverfolgung und einer für die Erkennung der gesuchten Palette. Außerdem sind mindestens drei Motoren erforderlich, einer für die Gabel und zwei für den Antrieb und die Lenkung.

Das Design der Führungslinie wurde dreifarbig gestaltet. Schwarz definiert den Idealpfad und Blau bzw. Rot die jeweilige Abweichung nach links bzw. rechts von der Spur.

III. Umsetzung

Da die Richtung der Abweichung durch Farben bestimmt wird, muss bei der Umsetzung des Projekts nicht nur die Konstruktion und Programmierung des Gabelstaplers, sondern auch die Gestaltung der Strecke berücksichtigt werden.

A. Erstellung der Linie

Das Grundkonzept der Bahn sieht vor, dass auf dem Weg zum Regal die rote Linie auf der rechten Seite und die blaue Linie auf der linken Seite in Bezug zur schwarzen Linie in der Mitte stehen. Außerdem sind Anfang und Ende der schwarzen Linie durch gelbe Punkte definiert, die dem Gabelstapler signalisieren, dass er eine Station erreicht hat.

Zu Beginn betrug die Breite jeder Linie 1 cm. Da es bei späteren Versuchen zu Problemen kam, bei denen das Fahrzeug durch einen annähernd senkrechten Anfahrwinkel die äußeren Linien nicht erkannte und überfuhr, wurden alle Linien auf 3cm verbreitert. Daher mussten alle Linien auf 3 cm verbreitert werden. Trotzdem musste eine exakte Position vor dem Regal gewährleistet sein, weshalb sich die schwarze Linie, wie in Abbildung [3] erkennbar ist, am Anfang und am Ende zu den gelben Markierungen hin auf 1 cm verjüngt.

Abbildung [3]: Anfangs-/Endsequenz der Führungslinie

B. Konstruktion

Da für diesen autonomen Gabelstapler eine konventionelle Lenkung gewählt wurde, stellt sich die Frage, wie der für den Antrieb zuständige Motor die Rotationsenergie auf die Räder übertragen soll. Eine Möglichkeit wäre, den Motor über Gelenke in der Antriebswelle und ein Differentialgetriebe mit den Rädern zu verbinden. Die schließlich gewählte Möglichkeit sieht vor, dass sowohl die Räder als auch der Antriebsmotor um den Lenkmotor schwenken, so dass der Schub immer direkt vom Antriebsmotor auf die Räder übertragen wird (siehe Abbildung [4]).

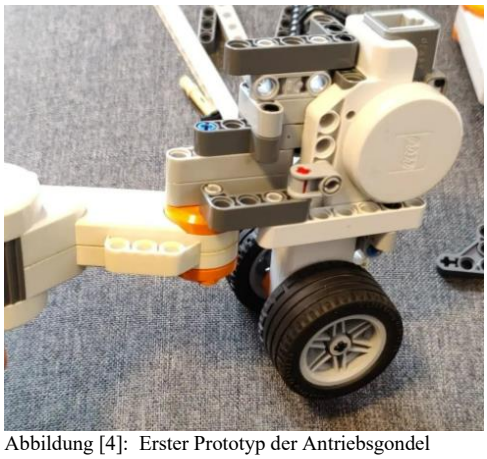


Abbildung [4]: Erster Prototyp der Antriebsgondel

Für die Struktur bedeutet dies, dass der Antriebsmotor nicht nur stabil, sondern auch starr mit dem beweglichen Teil des Lenkmotors verbunden sein muss, um die Last des darüber liegenden NXT und der restlichen Struktur tragen zu können. Außerdem muss der Platzbedarf dieser Verbindung minimiert werden, um den Auslenkungsgrad der Antriebsgondel zu maximieren. Ein maximaler Schwenkwinkel von 180° wird angestrebt, um das Wenden an einem Punkt zu ermöglichen.

Bei der Konstruktion kam es jedoch zu Konflikten zwischen der Stabilität und der Bewegungsfreiheit der Gondel, da der Antrieb zwar annähernd um 180° geschwenkt werden konnte, die Konstruktion jedoch unter dem Gewicht des NXT nachgab. Daher musste schließlich ein Kompromiss gefunden werden, der eine maximale Auslenkung von 150° vorsieht. Außerdem wurde der Motor, der für die Lenkung zuständig ist, weit nach vorne und unter die restliche Karosserie verlegt, was zwar den Anschluss des Motors nach vorne verlegt, aber die Stabilität des Gabelstaplers weiter verbessert. Das Fahrzeug ist dadurch kürzer und die Hebelwirkung auf die Verbindung ist geringer. Aus diesem Grund konnte der Farbsensor für die Linienverfolgung nicht direkt auf Höhe der Vorderachse, sondern vorne angebracht werden. Dadurch könnte es theoretisch zu leichten Abweichungen beim Lenken kommen. Dies konnte jedoch durch die bereits diskutierte Linienverbreiterung behoben werden.

Zuletzt musste die Position des dritten Motors festgelegt werden, der die Gabel hebt und senkt. Es wurde schnell klar, dass der Motor hoch montiert werden musste, um einen möglichst großen Hub der Gabel zu gewährleisten und mehrere Module im Regal anfahren zu können. Die Übersetzung der Kreisbewegung des Motors in eine lineare Bewegung der Gabel wurde realisiert, indem ein Zahnrad direkt am Motor angebracht wurde und die Gabel an der Rückseite mit Zähnen versehen wurde, auf die das Zahnrad zugreifen kann.

Zusätzlich wurden an beiden Seiten Schienen angebracht, die die Gabel fixieren. Da die Gabel trotz der Schienen auf einer Seite nach unten fiel, wurde parallel zum Motor ein weiteres Zahnrad angebracht, das das Problem schließlich löste.

C. *Programmierung*

Bei der Programmierung stellte die Lenkung bzw. das Spurhalten aufgrund der gewählten Antriebsvariante die größte Herausforderung dar. Denn im Gegensatz zur Lenkung mit zwei separaten Antriebsmotoren ist bei der gewählten Variante ein punktgenaues Drehen nicht möglich, so dass immer ein gewisser Weg eingeplant werden muss, um die Richtung des Fahrzeugs zu ändern.

Phase 1: Zu Beginn speichert der Rechner mit Hilfe des im Lenkmotor integrierten Drehwinkelsensors die gerade Ausrichtung der Antriebsgondel als Nullposition. Da der Grad der Auslenkung variabel sein sollte, so dass der Roboter bei leichten Abweichungen von der Ideallinie nur geringe Richtungskorrekturen vornehmen muss, war es notwendig, die Dauer der Auslenkung des Roboters zu speichern. Da jedoch die TicToc-Funktion von Matlab das Auslesen der Zeit bei laufender Stoppuhr nicht erlaubt, wurden Matrizen verwendet, um die gescannten Farbwerte zu speichern.

```
if rot > 0 && rot <= 5
    gesucht = -15
elseif rot > 5 && rot <= 10
    gesucht = -25
elseif rot > 10 && rot <= 15
    gesucht = -35
elseif rot > 15 && rot <= 20
    gesucht = -45
end
```

Abbildung [5]: Stufen der Einlenkung

Das Programm entscheidet dann in Abhängigkeit vom Verhältnis der Farben zueinander (Abbildung [5]), wie groß die Auslenkung der Gondel sein soll. Dies kann dazu führen, dass der Lenkmotor automatisch die Nullposition anfährt, da beim Auffinden der Ideallinie Blau und Rot aus der Matrix verdrängt werden, was auf geraden Strecken zu einer immer geringeren Richtungsanpassung führt.

```
if gesucht - aktuell > 2
    mot2.Power = 5;
    SendToNXT(mot2);
elseif gesucht - aktuell < 2
    mot2.Power = -5;
    SendToNXT(mot2);
else
    mot2.Power = 0;
    SendToNXT(mot2)
end
```

Abbildung [6]: Schub des Lenkmotors bis Zielauslenkung

Sobald die gewünschte Auslenkung ermittelt ist, wird der Fahrbefehl an den Lenkmotor gegeben, der je nach Verhältnis der aktuellen Position zur ermittelten Auslenkung (kleiner oder größer) solange Schub gibt, bis der Zielwert erreicht ist (Abbildung [6]). Da die Motoren von Lego Mindstorms relativ ungenau sind, ist eine Abweichung von +/- 2° zwischen Zielwert und aktueller Position eingeplant.

Der Antriebsmotor gibt ab dem Zeitpunkt, an dem der Startpunkt erkannt wurde, einen konstanten Schub, bis der Zielpunkt wieder erreicht ist. Hierbei trat wiederholt das Problem auf, dass der Farbsensor aufgrund eines unbekannten Fehlers fälschlicherweise gelbe Signale auf der schwarzen Fahrspur erkannte, wodurch der Gabelstapler ungewollt auf der Fahrspur stoppte und in die Routine der Palettenauswahl überging. Dies konnte umgangen werden, indem Phase 1 erst dann beendet wird, wenn der Farbsensor über einen längeren Zeitraum Gelb erkannt hat.

Phase 2: Bei Erreichen des Regals wird der Farbsensor an der Gabel zur Palettenerkennung aktiviert und der Motor, der die Gabel hebt bzw. senkt, gibt einen konstanten Schub. Erkennt der Sensor die gesuchte Palette, wird die Phase 2 beendet und die Palette aus dem Regal entnommen. Wird die gesuchte Palette nicht gefunden und die zuvor ermittelte maximale Gabelhöhe überschritten, wird Phase 2 abgebrochen.

Phase 3: Sobald die richtige Palette gefunden ist, wird der Farbsensor an der Gabel deaktiviert und die Gabel um einen bestimmten Wert abgesenkt, der durch mehrmaliges Testen ermittelt wurde. Dadurch positioniert sich die Gabel unter der Palette, ohne an der darunter liegenden Palette hängen zu bleiben. Anschließend fährt der Roboter vor, hebt die Gabel leicht an und fährt zurück, um wieder frei vor dem Regal zu stehen. Nun wendet das Fahrzeug, indem es mit maximaler Rechtskurve zurückfährt, bis der Gabelstapler rechtwinklig zum Fahrweg steht. Anschließend fährt der Gabelstapler mit der gleichen Länge nach links, bis er wieder in der Spur steht.

Phase 4: Schließlich wird Phase 1 mit umgekehrter Richtungszuordnung der Linienfarben wiederholt, bis der Startpunkt erkannt wird. Dies markiert das Ende der Routine. Danach fährt der Roboter ein Stück vorwärts, fährt die Gabel in die gespeicherte Startposition zurück, geht in den Standby-Modus und wartet auf ein neues Startsignal (Abbildung [7]).

IV. Auswertung und mögliche Verbesserungen

Das Ergebnis der Vorführung war mehr als zufriedenstellend. Die Spur konnte erfolgreich verfolgt und die Zielpunkte in ca. 19 von 20 Fällen erkannt werden.

Die gewählte Lenkungsoption zeigte jedoch trotz der Verbesserung einige Grenzen auf. Beispielsweise ist es mit der gewählten Lenkung nicht möglich, in der Nullposition auf der Stelle zu wenden, was mit der zuvor genannten Option leicht möglich gewesen wäre. Dies hatte zur Folge, dass der Roboter zeitweise Probleme hatte, in engen Kurven die Spur zu halten. Um dieses Problem zu minimieren, musste der Antriebsschub deutlich reduziert werden. Dadurch kann der „Stabelgabler“ nur sehr langsam fahren.

Auf der Seite der Sensorik und Programmierung wäre eine mögliche Verbesserung, die Farbsensoren durch

Videokameras zu ersetzen. Damit wäre es möglich, den Linienverlauf als Grafik einzulesen und je nach Position und Steigung die Lenkung anzupassen. Dadurch würde sich die Breite der Linie drastisch reduzieren und es wäre jeweils nur eine Linie zur Anpassung notwendig. Außerdem würde sich die Möglichkeit eröffnen, mehrere Linien zu verwenden, um durch unterschiedliche Farben mehrere Regale/Ziele anzusteuern.

V. Zusammenfassung und Fazit

Der Gabelstapler (sog. „Stabelgabler“) (Abbildung [7]) funktioniert gut und macht seine Arbeit für die zwei Wochen, die für das Projekt vorgesehen sind, schon sehr gut. Mit mehr Bearbeitungszeit wäre es möglich gewesen, die Verarbeitung der Farben zu beschleunigen, so dass der Gabelstapler schneller fahren könnte. Außerdem wäre eine bessere Lenkanpassung möglich gewesen. Dies würde bedeuten, dass die Auslenkung nicht nur in Stufen unterteilt, sondern komplett variabel und in jedem Grad steuerbar wäre. Dies würde es dem Roboter ermöglichen, beim Auftreffen auf die schwarze Linie so gegenzusteuern, dass der Gabelstapler annähernd gerade über die schwarze Linie fährt.

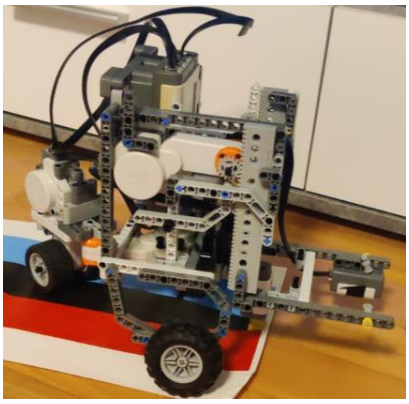


Abbildung [8]: Das Endresultat des Projekts

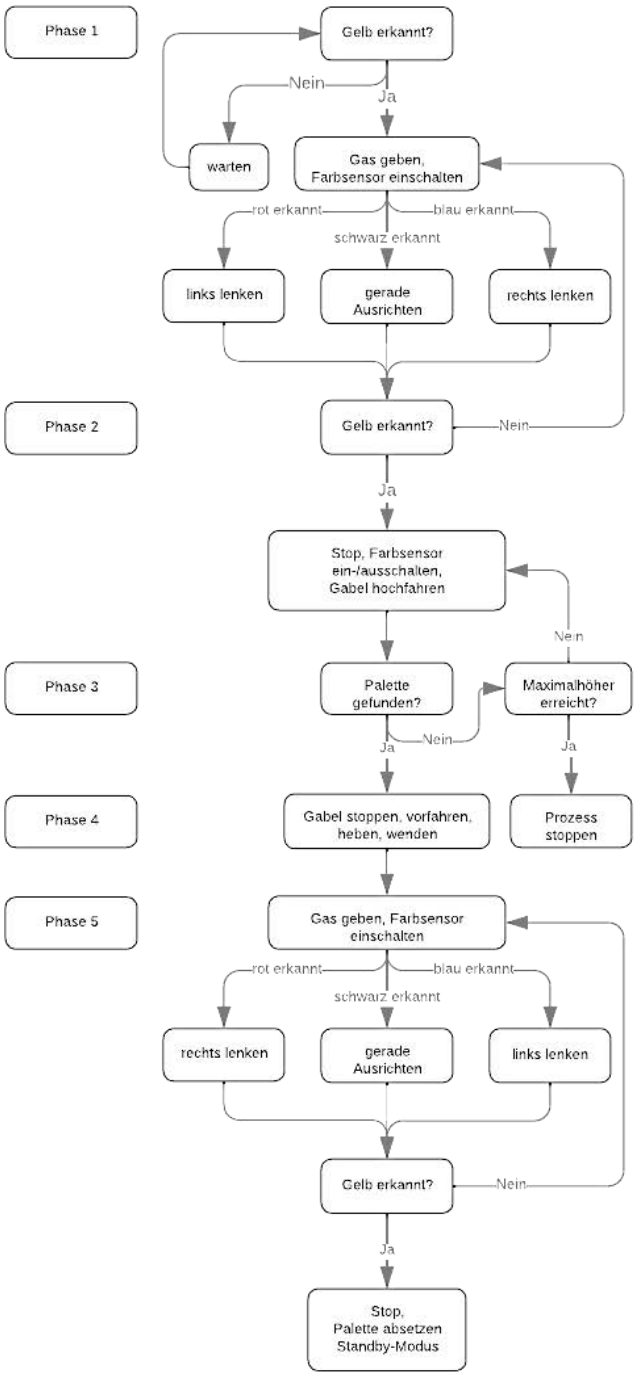


Abbildung [7]: Programmablauf als Flussdiagramm

Literaturverzeichnis

Bildquellen:

[1] https://www.brick-shop.de/images/product_images/original_images/9841-NXT-Brick.jpg (16.04.2023)

[2] https://automationspraxis.industrie.de/allgemein/autonomer-gabelstapler-mit-schwarmintelligenz/ (23.02.2023)

Literatur:

[3] Freddie Wilkinson. (2022, Juni). “Industrial Revolution and Technology”. Website: https://education.nationalgeographic.org/resource/industrial-revolution-and-technology/

Der Stabelgabler
-
Ein Ansatz für den innerbetrieblichen Transport

Robin Kaldyk, Elektrotechnik und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Abstract*— Innerhalb des Projektseminares Elektro- und Informationstechnik soll ein Roboter konstruiert werden. Dieser besteht aus einfachen LEGO-Bauteilen und Sensoren, welche mit einem NXT-Kontrollstein (Abbildung [1]) interagieren können. Im Zuge des Seminars musste dieser Stein durch die Programmier-Software MATLAB konfiguriert werden. Im Folgenden werden Konstruktion, aufgetreten Probleme und deren Lösungen eines Gabelstaplers erläutert, sowie sich mit Anwendungsmöglichkeiten und potenzieller Verbesserungen auseinandergesetzt.**

***Schlagwörter*— autonom, Gabelstapler, innerbetrieblich, Roboter, Transport**

Abbildung [1]: NXT-Kontrollstein

I. Einleitung

Roboter sind heutzutage unersetzbare Bestandteile der Industrie. Sie ermöglichen fehlerfrei Prozesse zu automatisieren, wodurch eine schnelle und saubere Verarbeitung von Massenware garantiert werden kann. Im Bereich der Logistik ist der innerbetriebliche Transport ein wichtiger Bestandteil der Unternehmen. Um dort die Arbeitseffizienz zu steigern und den Materialfluss zu beschleunigen, kommt es zum Einsatz von automatisierten Transportsystemen (Abbildung [2]). Durch die Verwendung von Sensoren können diese Systeme eigenständig Waren und Güter innerhalb des Betriebes organisieren und zu den benötigten Einsatzorten liefern. Der in diesem Projekt aus LEGO gebaute Gabelstapler verwendet genau solche Sensoren, um sich in seiner Umgebung zu orientieren.

DOI: 0.24352/UB.OVGU-2023-035

Abbildung [2]: Autonomer Gabelstapler der Marke Agilox

II. Vorbetrachtungen

Die Automatisierung des Gabelstaplers ist durch die verfügbaren Bauteile stark begrenzt. Es gibt nur wenige sinnvolle Möglichkeiten den Roboter annähernd präzise zu steuern. Außerdem ist es nicht möglich zwei unterschiedliche Prozesse über einen Motor anzusteuern, weshalb insgesamt drei Motoren verwendet werden. Einer ist für die Lenkung zuständig, ein anderer für den Antrieb und ein dritter zum Heben bzw. Senken der Gabel.

A. Antrieb und Lenkung

Grundsätzlich gibt es viele verschiedene Möglichkeiten den Bau der Lenkung und des Antriebs umzusetzen. Für einen Gabelstapler ist ein kleiner Wendekreis jedoch essenziell. Dieser ermöglicht es auch zwischen engen Regalen unkomplizierte Lenkbewegungen auszuführen, wodurch eine genaue Ausrichtung vor dem Regal möglich wird. Um dies zu erreichen kann entweder ein Raupenfahrwerk oder aber eine Hecklenkung, an dem sich gleichzeitig auch der Antriebsmotor befindet, verwendet werden. In dieser Version wurde sich für die Konstruktion einer Hecklenkung entschieden.

B. Sensorik

Der Roboter muss in der Lage sein, sich innerhalb des definierten Raumes zu orientieren und die richtige Ware aus dem Regal zu entnehmen, ohne dabei andere Gegenstände oder sich selbst zu beschädigen. Ursprünglich wurde diskutiert, eine Kombination aus Farb-, Licht- und Ultraschallsensor zu verwenden, letztendlich kamen jedoch nur zwei Farbsensoren zum Einsatz. Ein Sensor dient dabei der Orientierung anhand einer auf den Boden gezeichneten Linie, während der andere nach dem richtigen Farbmerkmal der gesuchten Palette scannt.

III. Umsetzung

Im folgenden Abschnitt werden die Zusammenhänge zwischen Konstruktion und Programmierung erläutert und es werden auf Probleme und deren Lösungen eingegangen.

A. Erstellung der Linien

Um festzustellen, in welche Richtung der Stapler lenken muss und gleichzeitig Fehlinterpretationen der Farben durch den Sensor zu vermeiden (der Sensor erkannte oft "weiß" als "gelb"), war schnell klar, dass eine dreifarbige Linie entworfen werden musste, deren Farben der Sensor eindeutig erkennen kann. Es wurde sich für die Linienfarben Blau (links), Schwarz (in der Mitte) und Rot (rechts) entschieden. Die ursprünglich zu schmal gewählten Linien (ca. 1cm) wurden nachträglich vergrößert (auf ca. 3cm), um eine präzisere Erkennung des Sensors zu gewährleisten. Zusätzlich verringert sich die Breite gegen Anfang und Ende der Linie und ist außerdem durch einen gelben Bereich gekennzeichnet. Dieser dient zur späteren Erkennung des Start- bzw. Endpunktes und ermöglicht auch die Ausrichtung vor dem Regal (Abbildung [3]).

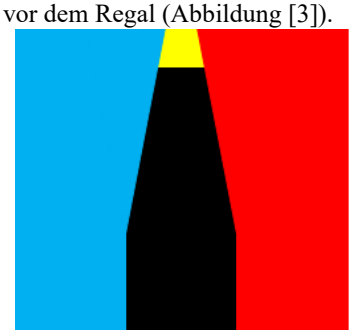


Abbildung [3]: Ausschnitt der Orientierungslinie

B. Konstruktion

Da bereits zu Beginn feststand, den Antriebsmotor direkt am Lenkmotor anzubringen, mussten diese beiden Teile sehr stabil miteinander verbunden werden, um das Gewicht des NXT-Steines tragen zu können und Verbiegungen an der Verbindungsstelle zu vermeiden (Abbildung [4]). Gleichzeitig war es wichtig, dass der Gabelstapler kompakt bleibt, um den Wendekreis so gering wie möglich zu halten und die Stabilität der gesamten Konstruktion zu gewährleisten. Aus diesem Grund wurde der Lenkmotor sehr weit vorne und unterhalb der eigentlichen Karosserie montiert.

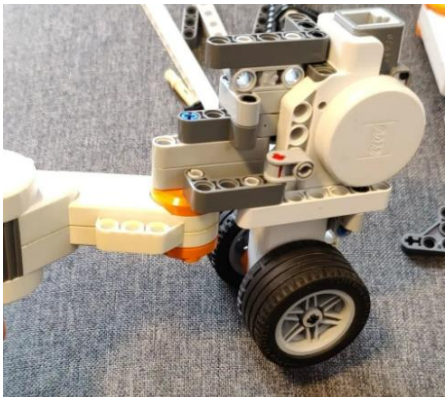
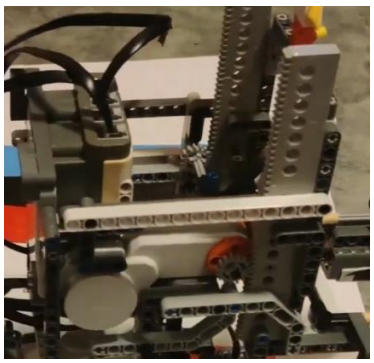


Abbildung [4]: Prototyp des Lenk- und Antriebsmechanismus

Um ausreichend Platz für den Lenkmotor zu schaffen, musste der Farbsensor, der für die Ermittlung der Position auf den Linien zuständig war, noch weiter vorne montiert werden. Folglich kam es zu Abweichungen zwischen der eigentlichen Position des Gabelstaplers und der erkannten Farbe durch den Sensor. Diese konnten jedoch durch die breiteren Linien minimiert werden.

Der dritte Motor, welcher für das Heben und Senken der Gabel zuständig ist, wurde über ein Zahnrad, mit den auf der Rückseite der Gabel, befestigten Zähnen verbunden. Somit kann die Rotationsbewegung des Motors in eine lineare umgewandelt werden. Die Stabilität der Gabel wurde weiter verbessert, indem auf beiden Seiten Schienen konstruiert wurden, auf denen die Gabel gleiten konnte. Dabei stützt ein mitlaufendes Zahnrad die Konstruktion und gewährleistet einen ständigen Kontakt zwischen Gabel und Motor (Abbildung [5]).



Abbildung[5]: Verbindung zwischen Gabel und Motor

C. Programmierung

Das größte Problem stellte die Auswertung der Lenkbewegung, auf Grund des gewählten Lenk- und Antriebsmechanismus, dar. Zur Lösung die allgemeine Vorgehensweise der Regelungstechnik angewandt. Um eine möglichst genaue Lenkung zu bewilligen, wurde zunächst die gerade Ausrichtung mittels des, im Motor, eingebauten Rotationssensors ermittelt und diese als Nullposition festgelegt.

Phase 1: Sobald der gelbe Startpunkt erkannt wurde, wird ein Startbefehl an den Antriebsmotor gesendet. Die erkannten Farben werden in einer Matrix (mit insgesamt 20 Elementen) gespeichert, um die aktuelle Bewegung des Staplers zu verfolgen und zu steuern. Basierend auf der Anzahl von roten, blauen und schwarzen Elementen innerhalb der Matrix wird bestimmt, in welche Richtung sich der Motor drehen muss, um die Lenkbewegungen auszugleichen und wie stark diese sein muss. Der Motor bewegt sich nach rechts bei der Farbe „blau“, nach links bei „rot“ und in Richtung Nullposition bei „schwarz“. (Tabelle 1)

Tabelle 1
Stärke des Einlenkens am Beispiel der Farbe Rot

Menge der roten Elemente in der Matrix	Größe des Einlenkwinkels (nach links)
größer 0 und kleiner 6	-15°
größer 5 und kleiner 11	-25°
größer 10 und kleiner 16	-35°
größer 15 und kleiner 21	-45°

Phase 2: Wenn nun erneut die Farbe "gelb" erkannt wird, hat der Stapler die Zielposition erreicht und der Antriebsmotor wird gestoppt. Der Lenkmotor kehrt dann in die Nullposition zurück und richtet somit den Gabelstapler vor dem Regal aus. Aufgrund der Ungenauigkeit der Motoren kann jedoch eine Abweichung von +/-2° auftreten.

Phase 3: Ist das Regal erreicht, wird der vordere Farbsensor aktiviert. Nun fährt die Gabel in einem konstanten Tempo nach oben, bis die richtige Palette durch die Farbcodierung der Palette erkannt wurde. Wird jedoch die vorher definierte Maximalhöhe überschritten bzw. die gesuchte Farbe nicht gefunden, wird der Vorgang abgebrochen.

Phase 4: Durch mehrfaches Testen wird die Gabel nun so weit gesenkt, dass der Stapler beim Hineinfahren in das Regal und anheben der Gabel nur die gewünschte Palette greift und nicht gegen andere Paletten innerhalb des Regals steuert. Anschließend fährt er zurück und wendet.

Phase 5: Ähnlich wie in Phase 1 folgt er nun der Linie, wobei die Lenkbewegungen vertauscht werden mussten. Erkennt er erneut den Start/Endpunkt, wird die Routine beendet und er geht in den Standby-Modus über, um ein erneutes Durchlaufen der Routine zu ermöglichen, ohne das Programm neu zu starten (Abbildung [6]).

IV. Auswertung und mögliche Verbesserungen

Das Projekt war im Großen und Ganzen mehr als zufriedenstellend. Der Gabelstapler konnte die Routine in ca. 19 von 20 Versuchen ausführen.

Durch die verwendete Lenkung mussten wir dennoch auf einige Dinge verzichten. So war es nicht möglich der Stapler schneller als 5% der maximalen Geschwindigkeit fahren zulassen, da andernfalls scharfe Kurven nicht erreicht werden konnten.

Auch gibt es sensorische Verbesserungsmöglichkeiten. So würde die Verwendung von Kameras zu genaueren Ergebnissen bei der Farberkennung führen und demnach auch die Lenkung schon etwas verbessern.

V. Zusammenfassung und Fazit

Der Gabelstapler (Abbildung [7]) (auch „Stabelgabler“) funktioniert für den zweiwöchigen Zeitraum gut und erfüllt die gestellten Aufgaben. Mit mehr Bearbeitungszeit könnte man mit Sicherheit die Lenkung noch weiter verbessern und die Geschwindigkeit, mit der der Stabelgabler fährt, etwas erhöhen.

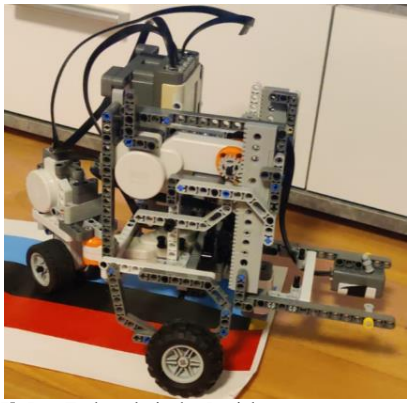


Abbildung [7]: Das Endergebnis des Projekts

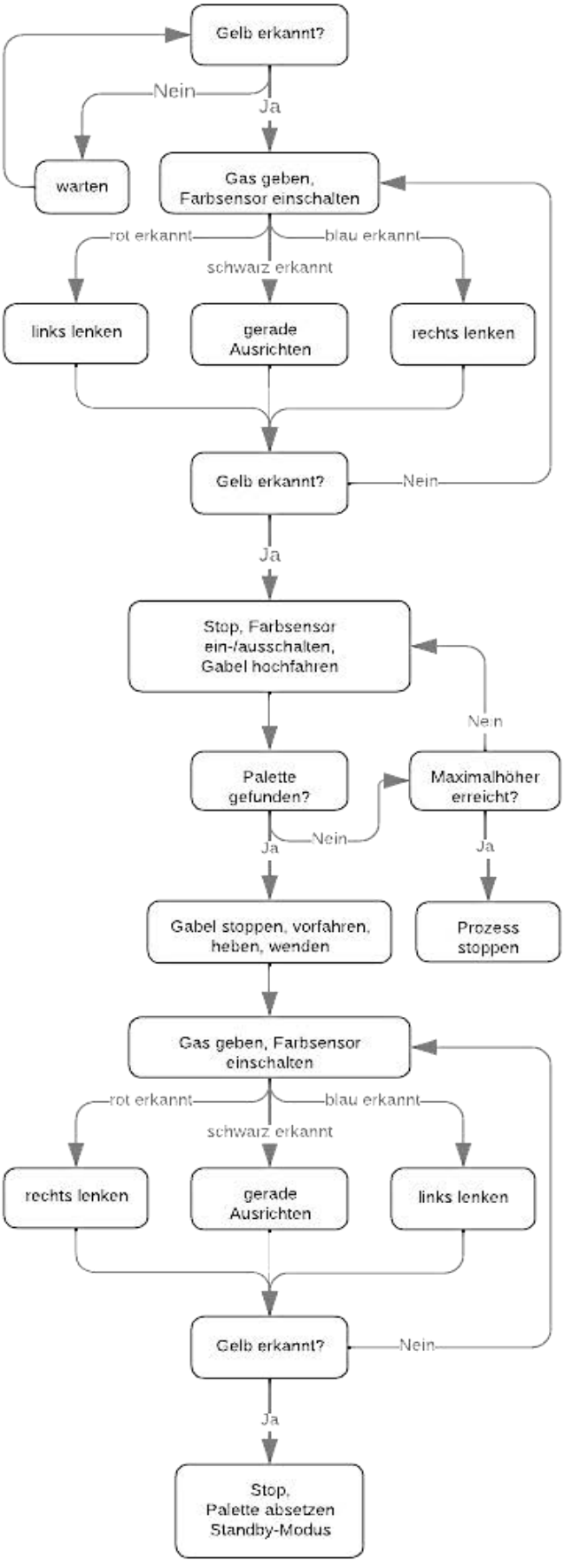


Abbildung [6]: Programmablauf als Flussdiagramm

LITERATURVERZEICHNIS

[1]
https://www.brick-shop.de/images/product_images/original_images/9841-NXT-Brick.jpg (01.03.23)
[2]
https://automationspraxis.industrie.de/allgemein/autonomer-gabelstapler-mit-schwarmintelligenz/ (02.03.23)

LEGO Drink Partner

Oleksii Sasin, Mechatronik
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Im Rahmen des jährlichen Lego-Seminars wurde ein lang ersehntes, aber aus Platzgründen nicht realisierbares Projekt verwirklicht: der Bau eines Roboters, der als Begleiter für Heißgetränke dienen sollte. Mit Hilfe von Legosteinen, einem NXT-Controller, drei Sensoren und drei Motoren konnte der Prototyp gebaut und getestet werden. Im Laufe des Projekts wurde deutlich, dass der erste Prototyp mit entsprechenden Weiterentwicklungen ein echtes Potenzial für einen nützlichen automatisierten Roboterassistenten hat. In den folgenden Abschnitten werden die technischen Daten des Projekts im Detail vorgestellt und sein Anwendungspotenzial beschrieben.**

I. Einleitung

DIE Zeit der Einschränkungen durch Covid-19 ist noch nicht vergessen und hat das soziale Leben von uns allen verändert. Für Studierende und junge Menschen zum Beispiel waren die Einschränkungen durch die “stay at home”-Regelungen eine besondere Herausforderung. Online-Lernen ohne die Möglichkeit, den Campus zu besuchen, das Verbot, sich mit Freunden zu treffen, und die Einschränkung, neue soziale Kontakte zu knüpfen, waren Faktoren, die dazu führten, dass sich einigen Studien zufolge die Gruppe der 18- bis 30-Jährigen in dieser Zeit am einsamsten fühlte [5]. In der Folge leidet die jüngere Generation unter schweren Depressionen und einer geringeren sozialen Lebensqualität.

Wie könnte der Roboter dieses Problem lösen? Die rein physische Aufgabe des Roboters besteht zunächst darin, ein Partner beim Trinken von Heißgetränken zu sein. Das bedeutet, dass er ein Objekt direkter physischer Interaktion wird, was während der Corona-Einschränkungen ein großer Mangel war. Eine Person drückt einen Knopf, der Prozess der Bewegung des Behälters zur Quelle des Getränks beginnt, das Getränk wird eingefüllt und nach einem Sprachsignal endet der Einfüllvorgang und der Behälter kehrt in seine Ausgangsposition zurück.

Zweitens ist es eine interessante und nicht triviale Möglichkeit, mit anderen Menschen zu interagieren, die nicht in der Nähe sind. Mit anderen Worten, es ist eine eindrucksvolle Art, einen „Anwesenheitseffekt“ zu erzeugen. Wenn sich zwei oder mehr Personen auf gegenüberliegenden Seiten des Bildschirms befinden, könnte man mit diesem Roboter gemeinsame „Drink-Partys“ veranstalten. Betrachtet man jedoch das industrielle Potenzial dieses Projekts, so bietet es für bestimmte Kundengruppen große Vorteile. So könnte ein solcher Roboter, wenn er weiter modernisiert wird, das Leben von bettlägerigen Patienten oder Behinderten erleichtern, die nicht in der Lage sind, allzu viele Bewegungen auszuführen. Eine Person, die an einen Rollstuhl oder ein Pflegebett gefesselt ist, muss nur einen Knopf drücken und rechtzeitig „Stopp“ sagen, um sich ein Glas Wasser einzuschenken.

DOI: 10.24352/UB.OVGU-2023-036

Wie dieses System funktioniert und vor welchen Herausforderungen es stand, wird im Folgenden ausführlich beschrieben.

II. Vorbetrachtungen

Die Inspiration für dieses Projekt kam von der Figur des Androiden-Barmanns Arthur, gespielt von Michael Sheen, aus dem Film Passengers. Der Mann, der allein an Bord des Raumschiffs geblieben war, fand Trost in der Hilfe des Androiden und fühlte sich nicht einsam. Natürlich reichten die Möglichkeiten von LEGO und das bisherige Wissen nicht aus, um ein solches Ergebnis zu erzielen, also wurde in der Anfangsphase im Internet recherchiert und versucht, den Entwurf so weit wie möglich zu vereinfachen, damit der Roboter eine bestimmte Aufgabe erfüllen konnte - beim Servieren helfen und Gesellschaft leisten. Denn wie bekannt ist: Wenn man allein trinkt, ist es Alkoholismus, aber wenn man es zu zweit tut, ist es eine Party.

A. Grundlegende Strukturelemente

Nach einigem Brainstorming kristallisierte sich das Design des gesamten Gerätes heraus. Zunächst waren es drei Hauptelemente:

- der Ständer mit dem eingebauten Controller (bildet das Zentrum des Entwurfs)
- ein beweglicher „Träger“ des Bechers, der über eine starre Kupplung mit einem festen Steuergerät verbunden ist
- der Mechanismus, der die Flüssigkeit aus der Flasche in das Glas gießt

Die gesamte Struktur sollte wie folgt funktionieren:
Der Benutzer stellt ein leeres Glas an die entsprechende Stelle und drückt die Starttaste, woraufhin sich das Glas auf den Füllmechanismus zubewegt und an der gewünschten Stelle anhält, woraufhin der Füllmechanismus seine Arbeit verrichtet, bis der Benutzer einen Sprachbefehl gibt, woraufhin das gefüllte Glas zum Benutzer zurückkehrt.

Nach der Festlegung des Gesamtkonzepts konnte die Umsetzung Schritt für Schritt erfolgen[2].

B. NXT und MAC

Da der NXT-Controller die Hauptsteuereinheit des gesamten Systems ist, musste als erstes die Kommunikation zwischen ihm, dem Computer und MATLAB hergestellt werden. Klingt einfach, oder? In Wirklichkeit war der erste Schritt jedoch schwierig. Der NXT wurde nämlich erst hergestellt, als 64-Bit-Betriebssysteme allgemein verfügbar waren, und alle Treiber, selbst für Windows, mussten angepasst werden, so dass ein einfaches Plug-and-Play nicht mehr möglich war. Außerdem

lief mein PC unter MacOS, was die Sache noch komplizierter machte.

Die übliche NXT-Kabelverbindung funktionierte überhaupt nicht, da es keine passenden Treiber gab. Die Situation wurde jedoch durch das Bluetooth-Modul des NXT gerettet. Es stellte sich heraus, dass der Aufbau der Bluetooth-Verbindung viel einfacher war und nicht einmal zusätzliche Treiber erforderte, nur eine kleine Konfiguration des Kanals mit Hilfe der unterstützenden Literatur [1], die Erstellung einer .ini-Datei im Hauptverzeichnis des NXT Toolpacks für MATLAB und alles funktionierte wie durch ein Wunder[3].

Das anfängliche Problem wandelte sich schließlich in einen Vorteil, da in diesem Fall keine Testverbindung erforderlich war und der Computer nicht von seinem üblichen Standort entfernt werden musste.

Sobald die Verbindung zwischen NXT und MATLAB hergestellt war, konnte mit der Konfiguration aller grundlegenden Elemente begonnen werden.

C. Sensoren und Aktoren

Die wichtigsten Elemente in diesem Projekt sind zwei Drucksensoren (im Folgenden S1 und S2 genannt), ein Schallsensor (im Folgenden SD genannt) und drei Aktoren (im Folgenden M1, M2 und M3 genannt). Mit Hilfe der NXT Toolbox Befehlsbibliothek für MATLAB wurden zunächst kleine Testskripte erstellt, um die Funktion der einzelnen Elemente zu testen. Nachfolgend ein Beispielcode zum Testen des Lärmsensors.

Der Befehl „Plot“ wurde verwendet, um das vom Sensor empfangene Signal visuell darzustellen (siehe Abbildung 1). Diese Informationen können verwendet werden, um die Bedingungen der Tonbefehle zu bestimmen. Der Schallsensor in MATLAB hat zwei Modi: dB und dB(A). Der dB-Modus wurde im Projekt verwendet.

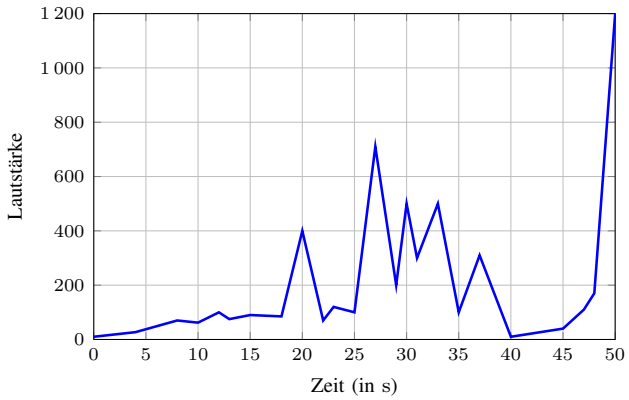


Abbildung 1. Audiosignal im „Db“-Modus [6]

III. Zusammenbau und Programmierung

Nachdem alle Elemente initialisiert und konfiguriert waren, konnte man den Algorithmus für den Betrieb des Systems entwerfen (Abbildung 2). Das Funktionsprinzip ist also sehr einfach:

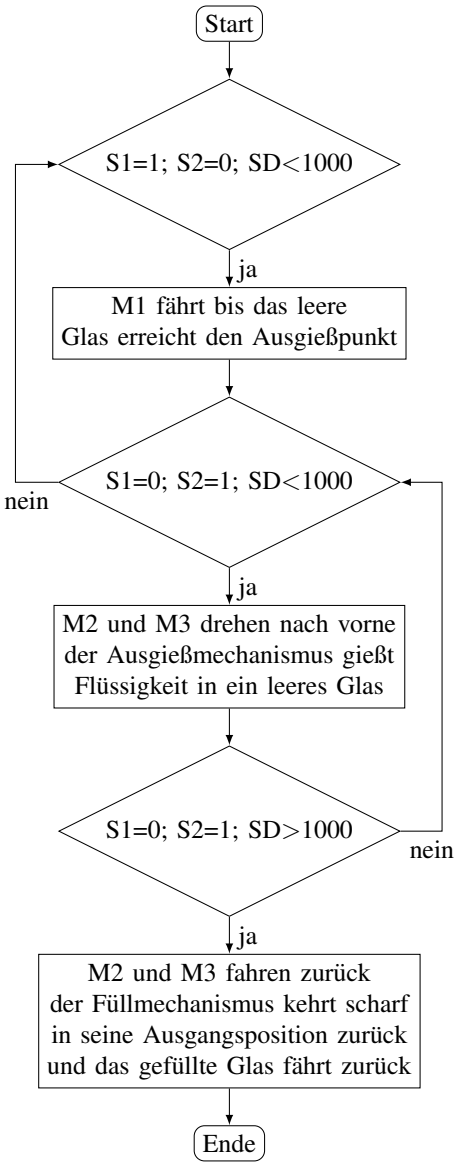


Abbildung 2. Programmablaufplan eines Verfolgungsalgorithmus

Nach dem Start des NXT-Programms wird ständig der Status der drei Sensoren abgefragt. Solange keiner der Sensoren betätigt wird, befindet sich das System im Ruhezustand. Sobald der Benutzer die erste Taste (S1) drückt, läuft der Motor M1 an, bis der Benutzer die zweite Taste (S2) drückt. Dann drehen die Motoren M2 und M3 synchron den Füllmechanismus und stoppen in der unteren Position, um das Getränk zu füllen, bis der Benutzer laut „STOP“ sagt. Das Programm wird dann beendet. Nun werden die einzelnen Elemente betrachtet.

A. Glasträger

Seine Konstruktion ist so einfach wie möglich. Er basiert auf einem Motor aus einem LEGO-Set und wurde durch einen kleinen LEGO-Kasten ergänzt, in den der Tumbler passt.

Wie in der Abbildung 3 zu sehen ist, treibt der Motor die beiden Räder auf der Mittelachse an, während ein drittes Rad zur strukturellen Stabilität eingebaut ist.

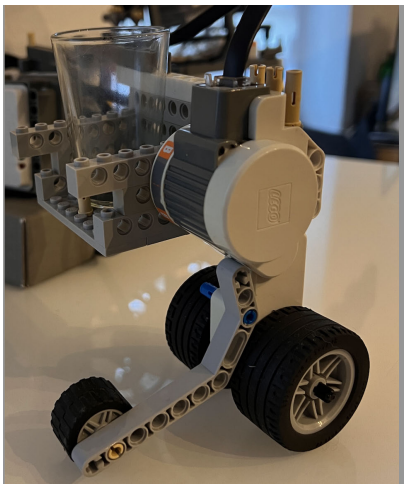


Abbildung 3. Glasträger

B. Füllmechanismus

Die Herstellung dieses Elements erwies sich als der schwierigste Teil des gesamten Projekts. Erstens musste die Konstruktion stabil und stark genug sein, um die Wasserflasche zu halten, und zweitens musste die Konstruktion ohne eine Mittelachse (da die Flasche sonst nicht installiert werden konnte) und ohne eine „Abdeckung“ (da die Flasche sich frei bewegen können musste) entworfen werden. Die Lösung bestand darin, das Drehmoment des Motors in einem Winkel von 90 Grad zu übertragen und zwei Motoren gleichzeitig zu verwenden.

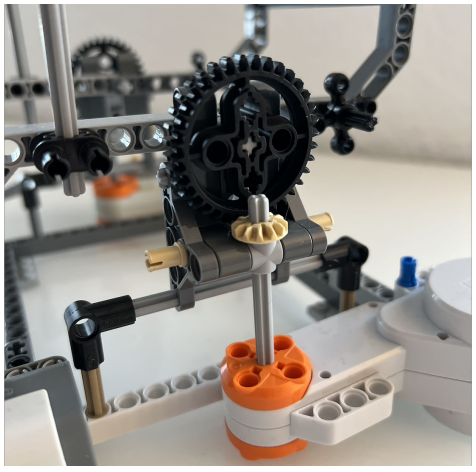


Abbildung 4. Drehmomentübertragung bei 90 Grad

Die beiden in der Abbildung 4 gezeigten Elemente sind einander gegenüber angeordnet und drehen sich jeweils um ihre eigene Achse. Abbildung 5 zeigt die endgültige Version dieser Vorrichtung.

C. Montage und Testen

Nachdem alle Grundelemente fertiggestellt waren, konnte mit der Montage begonnen werden. Der Deckel des Kastens diente als Basis, auf der alle anderen Elemente befestigt wurden. Plötzlich tauchte eine weitere Schwierigkeit auf, die es zu lösen

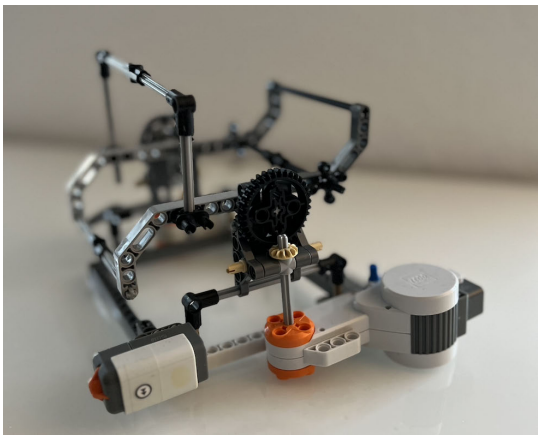


Abbildung 5. Füllmechanismus

galt, nämlich die Verlegung der Kabel. Mehrmals musste man Drähte unterschiedlicher Länge austauschen und Hilfskanäle in den Karton schneiden, um die Platzierung zu erleichtern. Der S2-Taster war ebenfalls in der Box eingebaut und wurde während des Tests mit einer kleinen Verlängerung versehen, um eine präzisere Bedienung zu ermöglichen.Am Ende war das gesamte System wie in Abbildung 6 dargestellt.

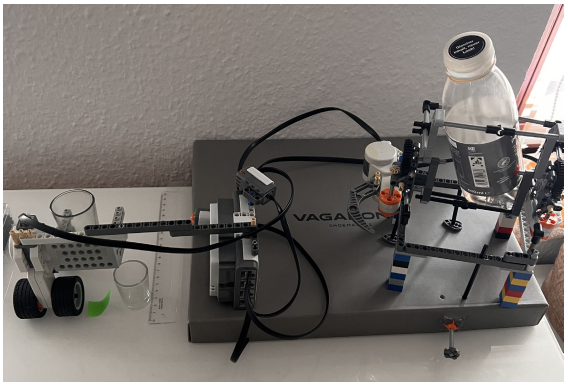


Abbildung 6. Zusammengebaute System

Bevor alles zusammen getestet wurde, wurden mehrere Hilfsskripte in MATLAB geschrieben, um jedes Element einzeln zu testen. Dies war nützlich um :

- Die Abstürze und Elementbrüche zu vermeiden.
- Alle Abstände und Winkel der Motoren zu überprüfen.
- Die Lautstärke für den Sprachbefehl festzustellen.

Diese Vorgehensweise würde so jedem empfehlen, denn nach der Fehlersuche in den einzelnen Schritten verliefen die Testläufe reibungslos. Alles funktionierte wie geplant, was bei der Verteidigung des Projekts unter Beweis gestellt wurde.

IV. Ergebnisdiskussion

Natürlich handelt es sich nur um einen Prototyp, aber beim Testen der endgültigen Version wurden einige Elemente entdeckt, die in zukünftigen Versionen des Projekts verbessert werden könnten:

1) Die Verbindung zwischen dem Controller und dem Becherträger muss stabiler sein, da alles unter Last etwas wackelig war
2) Die Konstruktion des Gießmechanismus könnte durch eine Rückwand ergänzt werden, um die Gesamtfestigkeit zu erhöhen.
3) Es wäre toll, wenn man die Möglichkeit hätte, mehrere Ausgießmechanismen zu bedienen, so dass der Benutzer ein Getränk auswählen und per Sprachbefehl auswählen könnte.

Insgesamt ist das Hauptziel des Projekts jedoch erreicht worden.

V. Zusammenfassung und Fazit

Dank der direkten Unterstützung des Betreuers und einer Fülle an unterstützender Literatur war es möglich, einen Prototyp eines kleinen Roboters zu bauen, zu programmieren, zu testen und seine Vor- und Nachteile zu analysieren. Es hat Spaß gemacht, selbst in die Rolle eines Ingenieurs zu schlüpfen und alle Phasen des Projektzyklus von der Idee über das Konzept bis hin zum Prototyp zu durchlaufen.

Dieses Projekt ermöglichte es vor allem, sowohl die Fähigkeiten im Umgang mit Matlab in einem verständlichen Format zu verbessern, als auch unschätzbare Erfahrungen bei der Lösung von Problemen zu sammeln, die bei der Entwicklung eines Projekts auftreten, und einen unschätzbaren Einblick in die Bedeutung kreativen Denkens zu gewinnen. Es war ein Vergnügen, an einem solchen Projekt teilzunehmen, und es würde schön sein, wenn solche Fächer auch in der Schule angeboten würden, damit sich neue Generationen von Ingenieuren dafür interessieren.

Literatur

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.
[2] Mathias Magdowski, Thomas Schallschmidt, Thomas Gerlach ,Ennrico Pannicke *LEGO-Praktikum. Entwickeln, programmieren, optimieren* Bd. 5 (2022): Wintersemester 2021/2022
[3] Enrico Pannicke *MatLab Handbuch* E-Learning-Portal
[4] *Unterlagen zum Praktikum LEGO Mindstorms in der FEIT* E-Learning Portal
[5] Buecker, S. et al., 2020 in Brakemeier, E.-L. et al., 2020
[6] Till Tantau, Joseph Wright, Vedran Miletić, „The beamer class“ , version 3.69 February 20, 2023