

Roboterlader

Hlib Horbachov, Elektro- und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

Zusammenfassung—Jedes Jahr veranstaltet die Otto-von-Guericke-Universität Magdeburg eine Projektwerkstatt für Elektro- und Informationstechnik. Im Rahmen des diesjährigen Seminars wurde ein spezieller Mechanismus, ein so genannter „Roboterlader“, entwickelt. Er hilft beim Transport von Lasten, bewegt sich selbstständig auf einer schwarzen Linie und parkt sich auch selbstständig in einer von drei vordefinierten Positionen ein. Für den Entwurf und die Entwicklung des Mechanismus wurden LEGO Mindstorms Sets und ein LEGO NXT Steuercomputer verwendet. Die Software wurde mit MATLAB implementiert. In diesem Beitrag werden der Aufbau und die Funktionsweise des Mechanismus vorgestellt. Darüber hinaus werden die Probleme, die während des Entwurfsprozesses aufgetreten sind, und die Möglichkeiten zu ihrer Lösung analysiert.

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

I. EINLEITUNG

Die moderne Robotik gibt uns die Möglichkeit, verschiedene alltägliche Aufgaben effektiv zu automatisieren, sei es in Unternehmen, im Haushalt oder in der Medizin. Im Jahr 2020, während der COVID-19-Pandemie, erlebte der Online-Handel einen Aufschwung, und viele Unternehmen sahen sich mit dem Problem verspäteter Lieferungen konfrontiert. Amazons Lagerhäuser haben die Situation jedoch mit autonomen Gabelstaplerrobotern gerettet, die in Logistikzentren auf der ganzen Welt eingesetzt werden [1].

Diese von Kiva Systems [2] (2012 von Amazon gekauft) entwickelten Roboter (Abbildung 1) sind in der Lage, schnell die richtigen Waren zu finden, sie den Mitarbeitern zum Verpacken zu übergeben und die Lagerprozesse zu optimieren. Dank dieser Roboter konnte Amazon die Geschwindigkeit der Auftragsabwicklung verdoppeln und Millionen von Menschen helfen, ihre Weihnachtsgeschenke rechtzeitig zu erhalten.

Dieser Fall zeigt, wie Robotik nicht nur die Effizienz steigern, sondern auch den Menschen echte Vorteile bringen kann, insbesondere in schwierigen Zeiten!

II. VORBETRACHTUNGEN

In diesem Abschnitt werden die wichtigsten Werkzeuge vorgestellt, die in diesem Projekt verwendet werden, sowie ihre Funktionen zur Durchführung bestimmter Aufgaben und zur Verarbeitung von Eingabedaten.

Roboterprogrammierung. Die Technologie basiert auf der Verwendung von LEGO Bausteinen, elektronischen Modulen und Software zur Erstellung und Konfiguration von Robotersystemen. In diesem Projekt wurde LEGO NXT verwendet, das trotz einer neueren Version von EV3 eine robuste und funktionelle Entwicklungsplattform darstellt. NXT bietet eine breite Palette von Sensoren und Motoren sowie eine visuelle



Abbildung 1. Amazon Laderoboter

Programmierung und ist damit ein praktisches Werkzeug für die Konzeption und Realisierung von Roboterlösungen. Im Rahmen des Universitätsprojekts Otto von Guericke 2024 wurde die LEGO-NXT-Plattform zur Entwicklung eines Roboters verwendet, der in der Lage ist, Aufgaben effizient auszuführen, den Konstruktionsprozess zu beschleunigen und den Maschinenbau zu vereinfachen.

MATLAB (Matrix Laboratory) ist eine von MathWorks entwickelte Softwareumgebung und Programmiersprache, die Funktionen für numerische Analyse, Modellierung und Datenverarbeitung bietet [3]. MATLAB enthält Werkzeuge für die Matrix-, Signal- und Bildverarbeitung und ist daher für die Analyse und Steuerung von Robotersystemen sehr nützlich.

III. KONSTUKTION UND PROGRAMMIERUNG

A. Aufbau

Das Design des Roboters zeichnet sich durch technische Einfachheit aus. Um effektiv arbeiten zu können, muss das System jedoch nicht nur externe Signale wahrnehmen, sondern auch in geeigneter Weise darauf reagieren, um Bewegungen zu ermöglichen. Motoren und Sensoren spielen dabei eine Schlüsselrolle, denn sie ermöglichen es dem Roboter, Lasten zu kontrollieren, präzise Manipulationen durchzuführen und sich an Veränderungen in der Umgebung anzupassen. Im Folgenden werden die wichtigsten Sensoren und Motoren und ihre Eigenschaften vorgestellt, die für eine optimale Roboterleistung erforderlich sind. Für den Zusammenbau des Mechanismus sind drei große Motoren erforderlich. Zwei davon steuern die Vorwärts- und Rückwärtsbewegung sowie die Drehung des Roboters, während der dritte Motor für die

Funktion der Klaue verantwortlich ist. Diese Kombination sorgt für präzise Manövrierbarkeit und hohe Anpassungsfähigkeit, so dass der Roboter seine Aufgaben effizient erfüllen kann. Nachstehend finden Sie ein Blockdiagramm (Abbildung 2 der Roboterlogik.

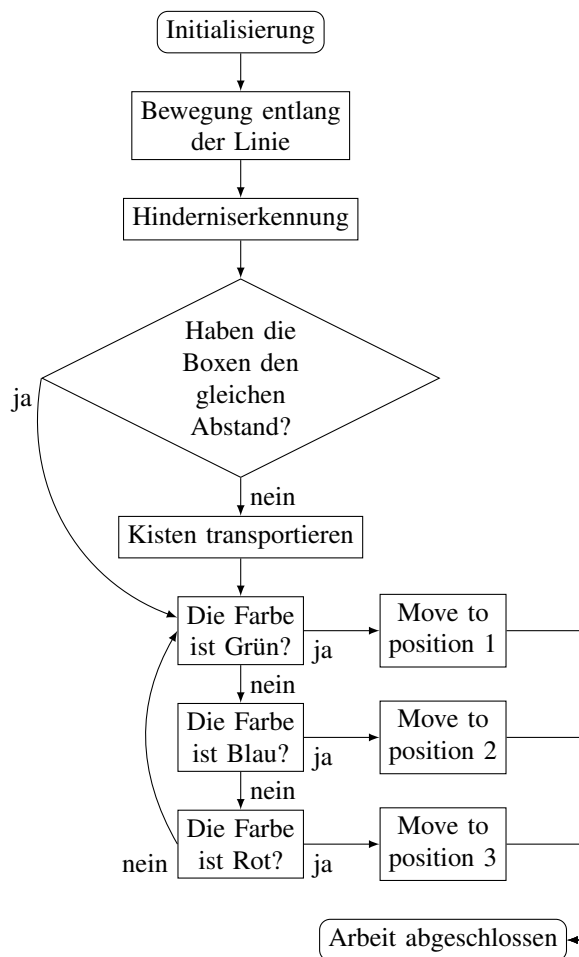


Abbildung 2. Blockdiagramm des Roboterbetriebs

B. Motoren

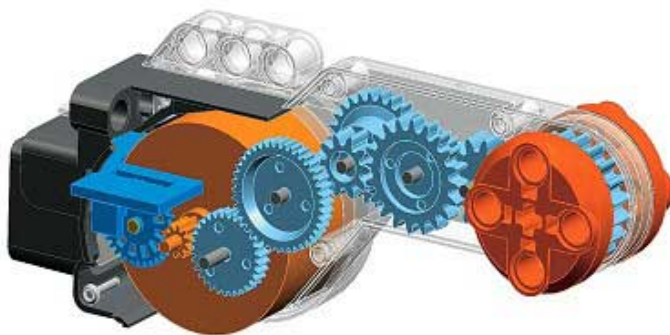


Abbildung 3. großer Motor

Für den Zusammenbau des Mechanismus sind drei große Motoren (Abbildung 3) erforderlich. Wie in [4] dargestellt,

zeigt das Bild die interne Struktur des NXT-Motors. Zwei davon steuern die Vorwärts- und Rückwärtsbewegung sowie die Drehung des Roboters, während der dritte Motor für die Funktion der Klaue verantwortlich ist. Diese Kombination sorgt für präzise Manövrierbarkeit und hohe Anpassungsfähigkeit, so dass der Roboter seine Aufgaben effizient erfüllen kann.

C. Lichtsensoren



Abbildung 4. Lichtsensor

Zwei Lichtsensoren (Abbildung 4) sind symmetrisch am unteren Teil des Roboters angebracht, um seine Bewegung präzise zu steuern. Ihre Hauptfunktion besteht darin, eine schwarze Linie zu erkennen, die dazu beiträgt, den Roboter auf einer vorgegebenen Bahn zu halten. Wenn einer der Sensoren eine schwarze Linie erkennt, passt der Roboter automatisch seine Richtung an, um auf dem Kurs zu bleiben. Dieses Kontrollsystem gewährleistet eine stabile Navigation und verhindert Abweichungen von der Route.

D. Ultraschallsensor Sensor



Abbildung 5. Ultraschallsensor

Moderne Roboter erkennen nicht nur Objekte, sondern interagieren auch gekonnt mit ihnen. Ein Ultraschallsensor (Abbildung 5) wird eingesetzt, um den Abstand zu den Kisten zu bestimmen und sie genau zu positionieren. Er misst zunächst den Abstand zur ersten Kiste, dann zur zweiten, analysiert die Daten und entscheidet, ob eine Anpassung notwendig ist. Wenn die Abstände nicht übereinstimmen, aktiviert der Roboter eine Klaue, die die zweite Kiste sanft an den richtigen Platz bewegt. Dank dieses Verfahrens werden die Objekte mit hoher Genauigkeit platziert, was die Effizienz der automatisierten Frachtbewegung deutlich erhöht. Diese Technologie findet breite Anwendung in der Logistik und in Produktionslinien, in denen Schnelligkeit und Präzision gefragt sind. Darüber hinaus minimiert dieses System das menschliche Eingreifen, was die Fehlerwahrscheinlichkeit verringert und die Arbeitsabläufe optimiert.

E. Programmierung

Der Roboter basiert auf dem NXT-Controller; er ermöglicht es dem Roboter, alle oben genannten Geräte in einem System zu kombinieren. Zur Programmierung habe ich eine Bibliothek mit Funktionen für den Roboter verwendet, die RWTH Mindstorms NXT Toolbox. Mit seiner Hilfe konnte ich viele Funktionen für den Roboter erstellen, um verschiedene Aufgaben auszuführen, und diese zu einem vollständigen Skript kombinieren, um mit der zugewiesenen Aufgabe zu arbeiten. Unten können Sie den Roboterlader auf dem Arbeitsfeld beobachten (Abbildung 6)

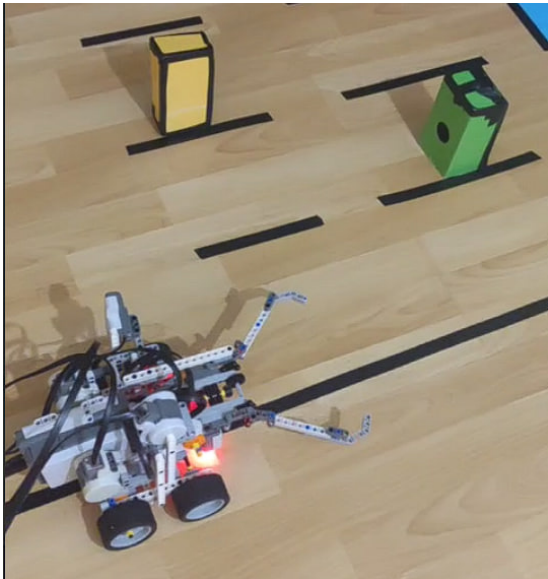


Abbildung 6. Roboterlader verrichtet Arbeit

Im Folgenden werden einige Funktionen vorgestellt, die zur Vermeidung von Wiederholungen im Code entwickelt wurden.

```
function moveDirekt(motorLeft,motorRight,speed)
    motorLeft.Power = -speed;
    motorRight.Power = -speed;
    motorLeft.SendToNXT();
    motorRight.SendToNXT();
end
```

Abbildung 7. moveDirekt Funktion

Diese Funktion (Abbildung 7) fährt die Räder des Roboters mit der eingestellten Geschwindigkeit vorwärts.

```
function correctLeft(motorLeft, motorRight)
    motorLeft.Power = 0;
    motorRight.Power = -10;
    motorLeft.SendToNXT();
    motorRight.SendToNXT();
    fprintf('CorrectLeft');
    pause(0.5);
    moveForward(motorLeft, motorRight, 20);
end
```

Abbildung 8. correctLeft Funktion

Diese Funktion (Abbildung 8) dient dazu, den Roboter nach links auszurichten, wenn der linke Sensor die schwarze Linie berührt. Während seiner Ausführung korrigiert das rechte Rad die Position des Roboters relativ zur Linie.

```
function correctRight(motorLeft, motorRight)
    motorRight.Power = 0;
    motorLeft.Power = -10;
    motorLeft.SendToNXT();
    motorRight.SendToNXT();
    fprintf('CorrectRight');
    pause(0.5);
    moveForward(motorLeft, motorRight, 20);
end
```

Abbildung 9. correctRight Funktion

Die Spiegelfunktion (Abbildung 9), die ausgeführt wird, wenn der rechte Sensor die schwarze Linie trifft.

```
COM_CloseNXT all;
h = COM_OpenNXT();
COM_SetDefaultNXT(h);

motorLeft = NXTMotor('C');
motorRight = NXTMotor('B');

OpenUltrasonic(SENSOR_1);
OpenLight(SENSOR_4, 'ACTIVE');
OpenLight(SENSOR_3, 'ACTIVE');

while true

    lightValueRight = GetLight(SENSOR_3);
    lightValueLeft = GetLight(SENSOR_4);

    if lightValueLeft < 450 && lightValueRight >= 450
        fprintf('Left\n');
        correctLeft(motorLeft, motorRight);
    elseif lightValueLeft >= 450 && lightValueRight < 450
        fprintf('Right\n');
        correctRight(motorLeft, motorRight);
    elseif lightValueLeft <= 500 && lightValueRight <= 500
        moveForward(motorLeft, motorRight, 20

        pause(1);
        stopMotors(motorLeft, motorRight);
        pause(2);
```

```

    if (lightValueLeft > 530 && lightValueLeft <
        630) || (lightValueRight > 530 &&
        lightValueRight < 650) % Red
        fprintf('Red\n');
        moveForward(motorLeft, motorRight, -20)
        ;

        pause(13);
        stopMotors(motorLeft, motorRight);

    end
    if (lightValueLeft > 350 && lightValueLeft <
        430) || (lightValueRight > 350 &&
        lightValueRight < 430) % blau
        fprintf('Blue\n');
        moveForward(motorLeft, motorRight, -20)
        ;

        pause(10.5);
        stopMotors(motorLeft, motorRight);
    end
    if (lightValueRight > 470 && lightValueRight <
        500) || (lightValueLeft > 470 &&
        lightValueLeft < 500) % green
        fprintf('green\n');
        moveForward(motorLeft, motorRight, -20)
        ;

        pause(8.3);
        stopMotors(motorLeft, motorRight);
    end

    CloseSensor (SENSOR_1);
    CloseSensor (SENSOR_3);
    CloseSensor (SENSOR_4);

    COM_CloseNXT(h);

    error('Program_finished.');
```

```

else
    fprintf('Forward\n');
    moveForward(motorLeft, motorRight, 20);
end
end
end

```

Abbildung 10. moveLinie Script

Ein Codeblock (Abbildung 10) aus dem vollständigen Roboterskript, der es dem Roboter ermöglicht, entlang der schwarzen Linie zu fahren sowie die Farbe und das entsprechende Verhalten des Roboters zu bestimmen.

IV. PROBLEME

Während der Entwicklung des Roboters traten verschiedene Probleme auf. Eines der interessantesten war die Verwendung eines EV3-Lichtsensors im NXT, was zu falschen Messwerten führte. Dieses Problem konnte jedoch leicht gelöst werden, indem man einen für den NXT geeigneten Sensor verwendete und diesen ersetzte.

Sobald der Roboter in der Lage war, der schwarzen Linie zu folgen, stand er vor der Aufgabe, zwei Kisten zu zählen, ohne unnötige Werte in Variablen einzugeben. Zunächst wurde die Verwendung einer Identifikationsmarkierung in Form einer Flagge in Betracht gezogen, die den Abstand angibt, in dem der Codeblock ausgeführt werden sollte, um die Position der Kisten zu fixieren. Es stellte sich jedoch heraus, dass der Roboter dieselbe Box mehrmals erkennt, so dass diese Methode ungeeignet war. Stattdessen wurde das Problem dadurch gelöst,

dass die Abstandserkennung nur aktiviert wurde, wenn der gemessene Wert unter einem bestimmten Schwellenwert lag. Die Messung wurde dann so lange gestoppt, bis der Roboter den Raum zwischen den Kisten durchquerte und erneut einen Abstand unterhalb des festgelegten Schwellenwerts erkannte.

Probleme gab es auch mit den Farbsensoren, da die verwendeten Sensoren den Farberkennungsmodus nicht unterstützten. Um die Farbe zu bestimmen, wurde der Bereich der Lichtreflexion verschiedener Farben analysiert und zur Identifizierung der Farben verwendet.

V. ERGEBNISDISKUSSION

Der Roboterlader hat die Aufgabe erfolgreich abgeschlossen. Die Sensoren ermittelten Farbe und Entfernung zu den Kisten korrekt und die Motoren bewegten die Ladung reibungslos und stellten sie im dafür vorgesehenen Bereich ab. Der Algorithmus arbeitete exakt gemäß der angegebenen Befehlsfolge und gewährleistete so die Zuverlässigkeit und Effizienz des Betriebs. Das System ist bereit für die Integration in reale Logistikprozesse. Das Projektziel wurde erfolgreich erreicht.

LITERATURVERZEICHNIS

- [1] HABR: *Artikel über Roboterlader in Amazon-Lagern während der Covid-19-Pandemie*. <https://habr.com/ru/companies/pochtoy/articles/493094/>. Version: 2020. – Zugriff am 13. März 2025
- [2] WIKIPEDIA: *Amazon Robotics*. https://en.wikipedia.org/wiki/Amazon_Robotics. Version: 2025. – Zugriff am 13. März 2025
- [3] MATHWORKS: *MATLAB zur Robotersteuerung – Dokumentation zur Arbeit mit NXT über MATLAB, Einrichten von Sensoren und Motoren*. <https://www.mathworks.com/solutions/robotics.html>. Version: n.d.. – Zugriff am 13. März 2025
- [4] PHILO: *NXT Motor Analysis – Bildquelle*. <https://www.philohome.com/nxtmotor/nxtmotor.htm>. Version: n.d.. – Zugriff am 13. März 2025