Manipulator

Mykyta Samiliak, Elektrotechnik und Informationstechnik
Otto-von-Guericke-Universität Magdeburg

***Abstract* — Diese Arbeit beschäftigt sich mit der Planung, dem Design und schließlich der Programmierung eines Robotermanipulators, der mit einem LEGO-Mindstorms-EV3-Controller interagieren kann. Das Hauptziel ist es, ein Objekt mit einem Berührungssensor zu erkennen und es mit Hilfe von Motoren und Greifern zu bewegen. Das Programm wurde mit MATLAB geschrieben.**

***Schlagwörter* — LEGO Mindstorms, EV3, Projektseminar, Manipulator, Roboter, Greifer.**

I. Einleitung

Seit langer Zeit verbessern und entwickeln die Menschen Maschinen, die sie bei ihrer harten Arbeit unterstützen. Diese Maschinen werden in der Industrie eingesetzt und helfen, Zeit, Geld und menschliche Energie zu sparen. Heutzutage sind Roboter ein unverzichtbarer Bestandteil der Industrie. Sie ermöglichen die Automatisierung von Prozessen, was eine schnelle und präzise Verarbeitung von Massenprodukten ohne Fehler gewährleistet. Eine besonders wichtige Rolle spielen dabei innerbetriebliche Transportmittel wie Pick-up-Trucks (siehe Abbildung 1). Sie steigern nicht nur die Effizienz der Arbeit, sondern beschleunigen auch den Materialfluss innerhalb des Unternehmens. Zur Erreichung dieses Ziels werden verschiedene automatisierte Transportsysteme mit Sensoren eingesetzt. Diese Systeme sind in der Lage, Waren und Güter eigenständig zu organisieren und an die benötigten Orte innerhalb des Unternehmens zu liefern. Somit nutzen Pick-up-Trucks, ähnlich wie andere robotergesteuerte Transportmittel, modernste Technologien, um ihre Aufgaben in der Industrie effizient zu erfüllen.

Abbildung 1[1]: Manipulator Auto

II. Vorbetrachtungen

Ziel des Projekts ist die Entwicklung eines Roboters, der Objekte autonom laden kann. Der Roboter soll Objekte mit Hilfe eines hexagonalen Lademechanismus aufnehmen und aufnehmen und transportieren können. Die Motivation hinter diesem Projekt ist Aufgaben im industriellen Umfeld zu automatisieren, um die Effizienz zu steigern zu steigern und menschliche Arbeitskräfte zu entlasten. Durch die Verwendung eines Objekterkennung und drei Motoren - ein kleiner Motor zum Greifen und zwei große Motoren für die Bewegung - soll eine kostengünstige und dennoch soll eine kostengünstige und dennoch effektive Lösung erreicht werden. Der Lademechanismus ist speziell für das seitliche Anheben von Objekten ausgelegt. was die Vielseitigkeit des Roboters erhöht und die Integration in verschiedene Arbeitsumgebungen ermöglicht.

A. Beschreibung des Designprozesses

Die ursprüngliche Idee war, einen Manipulator zu entwickeln und ihn auf die Maschine zu setzen. Aber wie ein Mann sagte: "Man kann nicht reinstopfen, was man nicht reinstopfen kann". Daher werden in diesem Abschnitt die Ergebnisse der Versuche vorgestellt und anschließend der unabhängig gewählte Lösungsansatz präsentiert.

Wie bereits erwähnt, funktionierte alles mit Zahnrädern, und das Hauptproblem bei der einfachen Konstruktion war die Größe der Klauen und die Frage, wo sie angebracht werden sollten.

B. Eigener Lösungsansatz

Um die Funktion und die Anwendung des Roboters zu optimieren, muss die Anzahl der Teile, aus denen die Klauen bestehen, reduziert, die Motoren richtig positioniert und die Griffigkeit verbessert werden.

III. Konstruktion

Dieser Abschnitt beschreibt die Entwicklung eines Roboters Schritt für Schritt vom Prototyp bis zum Endprodukt.

A. Mechanismus zum Greifen und Laden

Im Prinzip gibt es viele Möglichkeiten, das Greifen von Objekten zu realisieren. Es war jedoch wichtig, dass der Mechanismus nicht zu schwer ist, so dass der kleine Motor ihn bei etwa 180 Grad anheben kann. Wichtig war auch, dass er einen ausreichend großen Gegenstand aufnehmen kann. Zu diesem Zweck wurde beschlossen, einen Mechanismus zu

DOI: 10.24352/UB.OVGU-2024-036

verwenden, der das Objekt ergreift und es dank eines Getriebes (siehe Abbildung 2) anhebt.

In Abbildung 2 ist auch deutlich zu erkennen, dass der gesamte Mechanismus zum Greifen von Gegenständen durch einen kleinen Motor angetrieben wird. Das große schwarze Zahnrad ist für das senkrechte Anheben des Objekts zuständig. Kleinere Zahnräder sind für das Schließen der Klauen und für das Öffnen der Klauen bei der Rückkehr in die Ausgangsposition zuständig.

Die Bewegung der Klauen war ungenau und kehrte nicht in die Ausgangsposition zurück, was darauf zurückzuführen ist, dass die EV3-Einheit nur eine Codezeile auf einmal verarbeiten kann und die Befehle im MATLAB-Programm als separate Funktionen geschrieben wurden.

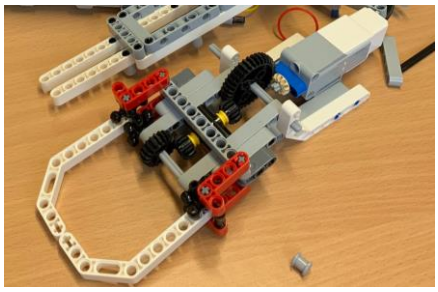

Abbildung 2: Mechanische Krallen

B. Sensor für Objekterkennung

Der erste Sensor, mit dem die Maschine erkennt, wann sich ein Objekt im Erfassungsbereich befindet, war ein Ultraschallsensor. Er war unter den Klauen angebracht, so dass er sie ständig sah und sehr ungenau war. Es wurde dann beschlossen, einen Farberkennungssensor zu verwenden. Sein Problem war, dass er Farben erst ab einer Entfernung von 1 cm oder weniger erkannte. Da diese Entfernung sehr kurz ist, konnten die Krallen das Objekt nicht erfassen. Schließlich wurde beschlossen, eine Taste mit einer Verlängerung zu verwenden, durch deren Betätigung der Transportvorgang gestartet wurde. Abbildung 3 zeigt ein Beispiel.

Abbildung 3: Sensor für die Objekterkennung

C. Finale Version.

Abbildung 4 zeigt die letzte und endgültige Version des Pickup-Tracks. Nach erfolgreicher Installation der Klauen an der Maschine wurde ein spezieller Behälter gebaut, in den das Objekt gelegt wurde. Zwei Probleme waren noch zu lösen: erstens die Installation des schwenkbaren Teils. Mechanisch war es schwierig, es zu installieren, da die gesamte Struktur ziemlich schwer und nicht drehbar war. Zweitens galt es, einen MATLAB-Code zu schreiben, da es Probleme beim Greifen des Objekts gab. Dies wurde durch die Verwendung spezieller Funktionen Abbildung 5 aus der Mindstorms EV3 Toolbox Dokumentation erreicht [1].

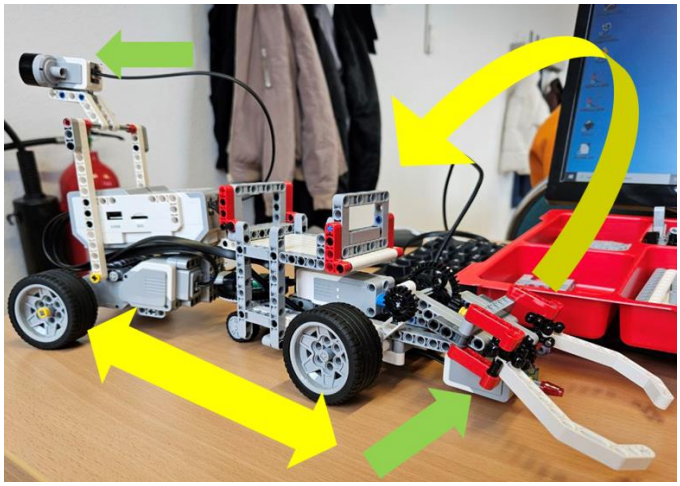

Abbildung 4: Schema der Objektaufhebung

Außerdem verfügte die endgültige Version über einen Ultraschallsensor, der für das Drehen zuständig war. Wenn der Abstand weniger als 50 cm betrug, wendete das Auto sanft nach rechts, bis der Abstand auf 50 cm erhöht wurde. Wenn der Knopf berührt wurde, schloss der kleine Motor die Klauen, die das Objekt ergriffen und sie um 170 Grad horizontal nach oben anhoben. Am Scheitelpunkt öffneten sie sich, der Gegenstand fiel in die Zelle, und eine Sekunde später kehrten sie in ihre Ausgangsposition zurück.

```
brick.motorB.smoothStart = 1;
brick.motorB.start();
```

Abbildung 5: SmoothStart, um das Objekt zu greifen.

IV. Software

In diesem Abschnitt wird die MATLAB-Software kurz beschrieben. Die verschiedenen vom Gerät ausgeführten Operationen sind in den folgenden MATLAB-Funktionen kodiert (siehe Abbildung 6).

V. Ergebnisdiskussion

Nach zahlreichen Funktionstests erwies sich der Roboter als erfolgreich. Allerdings führte die Rechtsdrehfunktion des Roboters zu einer ruckartigen bogenförmigen Bewegung anstelle der erwarteten gleichmäßigen Drehung. Darüber hinaus stellten die physikalischen Grenzen der LEGO-Steine und des EV3-Blocks während des gesamten Entwicklungsprozesses ein ständiges Hindernis dar.

Auch dank der Smoothstart-Funktion waren die Klauen nicht scharf und griffen das Objekt sanft. Das Problem, dass MATLAB jeweils nur eine Aktion ausführen kann, wurde durch die Funktion `waitFor()` behoben. Dabei wurden alle überflüssigen Teile entfernt, was die Maschine erheblich erleichterte.

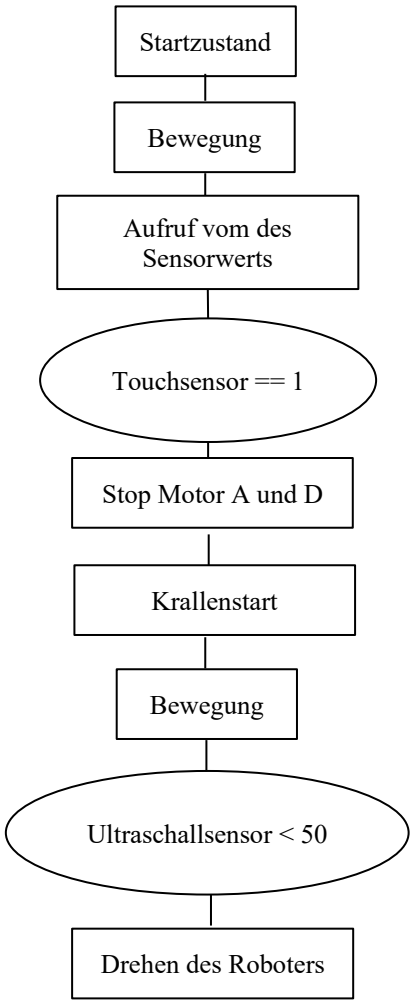

Abbildung 6: Programmablaufplan

VI. Zusammenfassung und Fazit

Zusammenfassend lässt sich sagen, dass alle Ziele mit Hilfe der MATLAB-Programmierung erreicht wurden. Die Bluetooth-Verbindung wurde ebenfalls hinzugefügt, aber um Probleme zu vermeiden, wurde die Verbindung über Kabel verwendet. In Zukunft ist es möglich, den rotierenden Teil zu verbessern und auch den Entladevorgang zu gestalten. Die Idee war, einen Magneten in das Objekt einzufügen und einen zweiten großen Magneten als separate Station zu errichten, bei dessen Annäherung das Objekt nach oben gezogen wird und das Auto entladen wird.

Anhang

```
brick.motorA.power = -30;
brick.motorD.power = -30;
brick.motorA.start;
brick.motorD.start;
% Richtung den MotorB;
motorB direction = 1;
```

Abbildung 7: Fahrtrichtung vor dem Erwerb des Ziels

```
Brick.motorB.limitValue = 170;
Brick.motorB.power = 50;

Brick.motorB.smoothStart = 1;
Brick.motorB.start();
%brick.motorB.waitFor();
Pause(2);
%Verzögerung, um mehrfache Taste
Brick.motorB.power = -60;
Brick.motorB.start();
Brick.motorB.waitFor();
Pause(1);
%Zuruck motorA
```

Abbildung 8: Der Prozess des Anhebens des Objekts und der Rückführung der Klauen in ihre ursprüngliche Position

```
if brick.sensor4.value () < 50
  brick.motora.stop () ;
  brick.motorD.stop ();
  brick.motorD.power = - 97;
  brick.motorA.power = 100;
  brick.motorA.start ();
  brick.motorD.start ();
  pause (10) ;
  brick.motorA.stop ();
  brick.motord.stop ();
  break
end
```

Abbildung 9: Drehfunktion, in einer bestimmten Entfernung

Literaturverzeichnis

[1] Der Manipulator Auto ist ein Spezialfahrzeug für das Be- und Entladen sowie den Transport von Fahrzeugen, URL: https://goo.su/6dwbh

[2] RWTH Aachen, “Mindstorms EV3 Toolbox”, Documentation, Release v1.0, Jan 27, 2020, URL : https://git.rwth-aachen.de/mindstorms/ev3-toolbox-matlab/-/blob/master/MindstormsEV3Toolbox.pdf?ref_type=heads