

Automatischer Gabelstapler

Yehor Popovych, Elektromobilität
Otto-von-Guericke-Universität Magdeburg

Im Rahmen des Design-Workshops an der Otto-von-Guericke-Universität Magdeburg wurde ein automatisierter Gabelstapler entwickelt. Ziel des Projekts ist die Optimierung des innerbetrieblichen Transports durch eine autonome Lösung.

Zur Umsetzung kamen LEGO-Mindstorms-Bausätze und der NXT-Steuerungscomputer zum Einsatz. Dieser Artikel beschreibt den Aufbau, die Funktionen sowie die wichtigsten Herausforderungen und Lösungsansätze.

Schlagwörter – Berührungssensoren, Elektromotoren, Farbsensor, Gabelstapler, LEGO, MATLAB.

I. IDEE

In diesem Projekt wird ein vollautomatischer Gabelstapler aus LEGO-Teilen konstruiert. Für die vollständige Automatisierung werden LEGO-Mindstorms-NXT-Komponenten verwendet, darunter Farbsensoren, Berührungssensoren und Elektromotoren. Durch den Einsatz dieser Sensoren kann der Roboter das Vorhandensein einer Last auf der Plattform erkennen, die Farbe der Palette erfassen und sie an einen bestimmten Ort im „Lager“ transportieren. Das Hauptmerkmal des Systems ist die Fähigkeit, vergleichsweise schwere Objekte, wie beispielsweise ein iPhone, anzuheben. Beim Entwurf des Designs wurden keine externen Prototypen aus dem Internet verwendet.

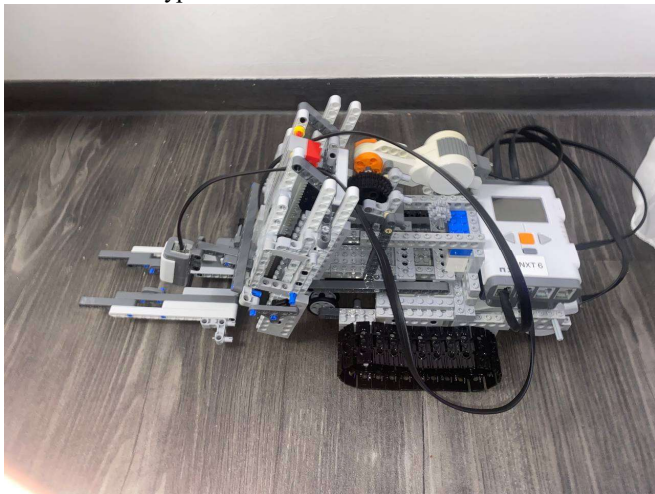


Abbildung 1: Seiten- und Draufsicht des Roboters

II. KONSTRUKTION

Das Design wurde unabhängig und ohne die Verwendung von Prototypen Dritter entwickelt (Abb. 1).

A. Fahrtrieb

Bei der Konstruktion des Fahrtriebs müssen die spezifischen Anforderungen des Roboters berücksichtigt werden, insbesondere die Fähigkeit, sich in alle Richtungen zu bewegen, auf engem Raum zu manövrieren und auf der Stelle zu drehen. Für diese Anforderungen ist ein Raupenfahrwerk am besten geeignet, da es eine hohe Wendigkeit und Stabilität gewährleistet.

Der Antrieb erfolgt über zwei Elektromotoren, jeweils einen pro Raupe. Diese Lösung verbessert die Manövrierfähigkeit sowie die Geländegängigkeit auf begrenztem Raum. Zudem erhöht der Einsatz von zwei Motoren die Leistungsdichte des Roboters, sodass er auch unter Last höhere Geschwindigkeiten erreichen kann.

B. Hubantrieb

Der Hubantrieb muss so ausgelegt sein, dass er das Heben schwerer Gegenstände ermöglicht. Zu diesem Zweck wird ein einzelner Elektromotor verwendet, der ein Getriebe antreibt. Um das Übersetzungsverhältnis zu optimieren und die Reibung zu erhöhen, kommen zwei Zahnradpaare zum Einsatz, die durch ihre Rotation die an der Plattform befestigte Zahnstange vertikal bewegen.

Die gesamte Konstruktion ist horizontal geneigt, um die Hubkapazität am oberen Endpunkt zu maximieren. Durch diese Neigung wird das Gewicht der Plattform auf die Zahnräder übertragen, was die Stabilität und Effizienz des Hebevorgangs verbessert.

C. Plattform

Die Plattform muss so konstruiert sein, dass sie das Anheben von Paletten und deren Transport zu den Regalen ermöglicht. Zu diesem Zweck wurde eine Bauweise mit zwei Gabelzinken und darauf befindlichen Befestigungslaschen entwickelt. Diese Konstruktion erlaubt es, verschiedene Lasten sicher anzuheben, zu fixieren und zu den vorgesehenen Regalen zu transportieren.

D. Konstruktionsmerkmale

Die Hauptsteuereinheit sowie drei Elektromotoren wurden an die Rückseite des Roboters verlegt, um ein Gegengewicht zu erzeugen (Abb. 2). Diese Anordnung trägt zur Stabilisierung des Roboters bei und verbessert seine Balance während des Betriebs.



Abbildung 2: Seitenansicht des Roboters

E. Strukturelle Probleme und deren Lösungen

Ein zentrales Problem bei der Konstruktion stellte das Durchrutschen der Ritzel und Zahnräder im Hubmechanismus dar. Aufgrund der unzureichenden Stabilität des Hebemechanismus sowie des Spiels zwischen den Lego-Teilen hatte der Roboter Schwierigkeiten, schwere Objekte anzuheben.

Zur Lösung dieses Problems wurde der Hubmechanismus nach hinten geneigt, sodass das Gewicht der Last im Verhältnis zur Hubbewegung auf den Kontaktpunkt zwischen Zahnrädern und Zahnstange drückt. Zusätzlich wurde die gesamte Rahmenstruktur verstärkt und stabilisiert, um unerwünschte Spielräume zu minimieren und die Gesamtstabilität zu verbessern.

III. AUTOMATISIERUNG

A. Automatisierungsmöglichkeiten für Lego-Roboter

Die Lego-NXT-Steuereinheit wird zur Programmierung und Automatisierung des Roboters verwendet. Zur Realisierung der Funktionalität eines Gabelstaplerroboters kommen Drucksensoren zum Einsatz, um das Vorhandensein einer Last auf der Plattform zu erkennen. Zusätzlich wird ein Farbsensor integriert, der die Sortierung der Paletten ermöglicht.

B. Tastsensor

Das Lego-NXT-System verfügt über einen Tastsensor, der an der Hebebühne befestigt ist. Beim Annähern des Roboters an eine Palette wird eine Taste betätigt, wodurch die nachfolgenden Schritte im Algorithmus aktiviert werden. Auf diese Weise erfolgt die Erkennung einer Last auf der Plattform vollständig automatisiert.

C. Farbsensor

Eine weitere Aufgabe des Roboters besteht darin, die Last an einen definierten Ort zu transportieren. Zur Sortierung der Paletten wird ein Farberkennungssystem eingesetzt. Der Farbsensor des Lego-NXT-Systems erfasst die Farbe der Palette und steuert den Transport an den entsprechenden Zielort.

Die Farbmarkierung auf der Palette ist so positioniert, dass sie vom Sensor zuverlässig erfasst werden kann. Beispielsweise wird eine blaue Palette dem unteren Regal zugewiesen, während eine rote Palette im oberen Regal platziert wird.

D. Probleme und Lösungen

Ein zentrales Problem bei der Installation der Sensoren ist der begrenzte Erfassungsradius des Farbsensors. Experimentelle Untersuchungen haben gezeigt, dass der Sensorstrahl in einem Abstand von etwa 2 Zentimetern zuverlässig arbeitet (Abb. 3). Diese Einschränkung stellt besondere Anforderungen an die Konstruktion. Daher muss der Montageort des Sensors auf Grundlage der Abmessungen der Palette sowie der Position der Farbmarkierung präzise berechnet werden, um eine korrekte Farberkennung zu gewährleisten.

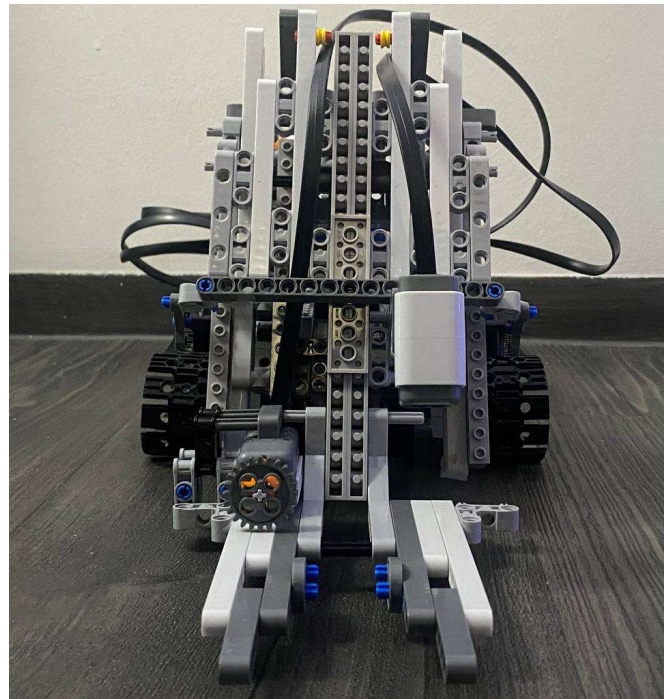


Abbildung 3: Vorderansicht des Roboters 1

IV. ROBOTER-ALGORITHMUS

Zu Beginn führt der Algorithmus des Roboters eine Demonstration aller seiner Funktionen und Eigenschaften durch.

A. Algorithmus

Der Ablauf des Roboters folgt dem folgenden Algorithmus:

1. Start an der Ausgangsposition
2. Rückwärtsfahrt
3. Annäherung an die erste Palette (rot auf dem Foto)
4. Anheben der Palette nach Betätigung des Sensors
5. Farbscan der Palette
6. Drehung in Richtung „Lager“

7. Platzierung der Palette auf dem vorgesehenen Regal (z. B. oberstes Regal)
8. Fahrt zur zweiten Palette (grün auf dem Foto)
9. Anheben der Palette nach Betätigung des Sensors
10. Farbscan der Palette
11. Drehung in Richtung „Lager“
12. Platzierung der Palette auf dem vorgesehenen Regal (z. B. unterstes Regal)
13. Abschluss des Algorithmus

B. Zusätzliche Gebäude

Zur Demonstration der Funktionsweise des Roboters werden verschiedene zusätzliche Strukturen auf der Lego-Plattform eingesetzt. Die wichtigste davon sind die Paletten, die speziell an die Größe der Gabelzinken des Gabelstaplers angepasst wurden und auf denen beliebige Objekte mit geeigneten Abmessungen platziert werden können. Jede Palette ist mit einer farbigen Markierung versehen, die für die Sortierung erforderlich ist.

Die Paletten werden an definierten Positionen aufgestellt, den sogenannten „Ladeterminals“ (Abb. 4). Zudem gibt es ein Regallager, in dem die Paletten entsprechend ihrer Farbmarkierung entweder im oberen oder unteren Bereich einsortiert werden. Darüber hinaus befindet sich auf der Plattform eine Startstation des Roboters, von der aus der gesamte Algorithmus initiiert wird.

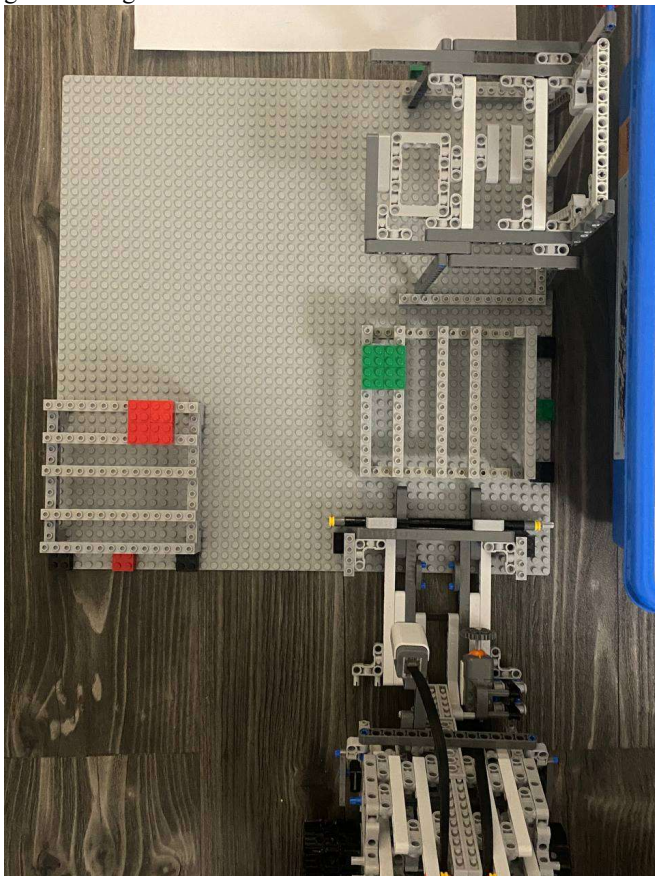


Abbildung 4: Ansicht des „Ladeterminals“ von oben

V. PROGRAMMIERUNG

Die Programmierung des Roboters erfolgt mithilfe der Software MATLAB unter Verwendung der zusätzlichen Bibliothek RWTH Mindstorms NXT Toolbox. Diese Bibliothek ermöglicht die Integration der Programmierungsumgebung mit dem Lego-NXT-System und erlaubt eine effiziente Steuerung des Roboters.

Im Folgenden wird ein Beispiel für einen Teil des Programmiercodes des Roboters vorgestellt (Abb. 5):

```
state = 0;

lastButtonState = -1;

while true

    currentButtonState = GetSwitch(SENSOR_1);

    if currentButtonState == 1 && lastButtonState == 0

        if state == 0

            mMoveA.Power = MOVE_MOTOR_POWER;

            mMoveB.Power = MOVE_MOTOR_POWER;

            mMoveA.SendToNXT();

            mMoveB.SendToNXT();

            state = 1;

        elseif state == 1

            mMoveA.Stop('brake');

            mMoveB.Stop('brake');

            mLiftC.Power = LIFT_MOTOR_POWER;

            mLiftC.TachoLimit = DEGREES_PER_CM;

            mLiftC.SendToNXT();

            mLiftC.WaitFor();

            mMoveA.Power = ROTATION_POWER;

            mMoveB.Power = ROTATION_POWER;

            mMoveA.TachoLimit = 540;

            mMoveB.TachoLimit = 540;
```

Abbildung 5: Teil des Programmiercodes

VI. LITERATURVERZEICHNIS

- [1] Lego Mindstorm Wikipedia
https://en.wikipedia.org/wiki/Lego_Mindstorms
- [2] RWTH - Mindstorms NXT Toolbox
<https://de.mathworks.com/matlabcentral/fileexchange/18646-rwth-mindstorms-nxt-toolbox>
- [3] Roboter-Konstruktionen mit Lego Mindstorms NXT
<https://www.generationrobots.com/de/content/60-roboter-konstruktionen-mit-mindstorms-nxt-v2>
- [4] Navigation von Wegpunkten mit einem Lego MINDSTORMS NXT Roboter
<https://de.mathworks.com/matlabcentral/fileexchange/42835-navigating-waypoints-with-a-lego-mindstorms-nxt-robot>