

# Flunky-Ball-Roboter

Wird nicht in der Lage sein, Flunky-Ball zu spielen, wird aber dabei helfen, Genauigkeit zu entwickeln.

Hennadii Shypunov, Elektrotechnik und Informationstechnik  
Otto-von-Guericke-Universität Magdeburg

**Zusammenfassung**—Im Rahmen des Projektseminars Elektrotechnik/Informationstechnik (LEGO Mindstorms) 2024 an der Otto-von-Guericke-Universität Magdeburg wurde mit Hilfe des LEGO-Baukastens ein Roboter konstruiert, der durch seine Bewegungen die Präzision menschlicher Würfe verbessern könnte.

Eine in MATLAB entwickelte Software ermöglicht es dem Roboter, sich entlang einer vorgegebenen Route zu bewegen. Diese Bewegung unterstützt die Verbesserung der Präzision bei menschlichen Ballwürfen, indem sie ein dynamisches Trainingsszenario bietet. Zudem verfügt der Roboter über eine Selbstabschaltfunktion, um Energie zu sparen.

Zum Abschluss des Projekts wurde der entwickelte Flunky-Ball-Roboter in einer Präsentation vorgestellt und seine Funktionalität demonstriert.

**Schlagwörter**—LEGO-Mindstorms, MATLAB, Präsentation, Präzision, Roboter.

## I. EINLEITUNG

UM einen Ball präzise zu werfen, muss eine Person mehr als 30 Muskeln einsetzen. Die Genauigkeit des Wurfs hängt von der Koordination und der Reaktion des Werfers ab. Gleichzeitig haben die Menschen zu Hause nicht immer die Möglichkeit, Bälle in voller Größe zu werfen, und der nächste geeignete Ort kann weit von ihrem Wohnort entfernt sein.

Durch das Üben mit Flunkyball-Robotern kann man Wurfgenauigkeit, Ausdauer und die Fähigkeit, geduldig zu sein, entwickeln, ohne das Haus zu verlassen. In diesem Fall ist das Risiko, Möbel zu beschädigen oder sich zu verletzen, minimal.

Der Roboter verfügt über einen stabilen Radstand, der einen Korb mit Sensor entlang einer vorgegebenen Strecke bewegt. Zusätzlich zur Bewegung der Basis selbst dreht sich auch der Sensorbehälter um  $180^\circ$ , was den Vorgang etwas erschwert. Ein im Korb integrierter Sensor ermöglicht es mehreren Personen, abwechselnd leichte Plastikbälle in verschiedenen Farben zu werfen, was das Spiel interessanter macht. Wenn der Ball nicht geworfen wird, bleibt der Roboter am Ende der Strecke stehen. Dadurch wird sichergestellt, dass der Roboter nicht mit Objekten außerhalb seines Bewegungsbereichs kollidiert.

## II. VORBETRACHTUNGEN

Damit ein Roboter die ihm übertragenen Aufgaben erfolgreich ausführen kann, muss man genau verstehen, welche Möglichkeiten dafür zur Verfügung stehen, und auch sicherstellen, dass die Idee als Ganzes umgesetzt werden kann.

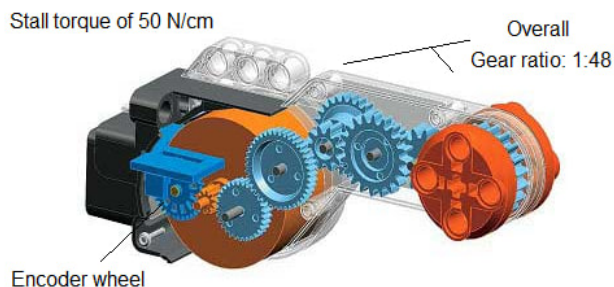


Abbildung 1. Innenansicht des Motor [1]

### A. Einführung in Motoren und Sensoren

Bevor mit dem Zusammenbau des ersten Modells begonnen wurde, wurde beschlossen, die Funktionsweise der NXT-Motoren (siehe Abbildung 1) zu untersuchen und herauszufinden, welche Geschwindigkeiten sie erreichen können. Auch dem Farbsensor wurde besondere Aufmerksamkeit gewidmet, da er ein sehr wichtiges und fragiles Element des Projekts war. Dank einer so kleinen Studie war es möglich zu verstehen, welche Möglichkeiten uns der LEGO-Konstrukteur für die weitere Arbeit damit bietet. Dieses Wissen ermöglichte auch ein tieferes Verständnis der Softwarekomponente des Projekts, was es später ermöglichte, zuverlässigen und verständlichen Quelltext zu erstellen.

### B. Roboteraufgaben

Es ist wichtig, die Aufgaben, die der Roboter ausführen muss, klar zu definieren, da Unklarheiten oder Unsicherheiten dazu führen können, dass eine zunächst erfolgreiche Idee in einer Reihe von Versuchen, sie zu verbessern, stecken bleibt. Daher wurde beschlossen, ein starkes konzeptionelles Grundgerüst zu erstellen und diesem dann Quelltext und LEGO-Elemente hinzuzufügen.

### C. Technische Umsetzung und Softwarekomponenten

Trotz der Vielzahl an Designideen waren die Möglichkeiten des Projekts zunächst durch die Fähigkeiten von MATLAB und die spezifischen Herausforderungen beim Zusammenbau von LEGO-Strukturelementen begrenzt. Diese Faktoren schränkten auch die Fähigkeiten des Roboters ein. Außerdem ergaben sich während der Programmierung des Roboters Herausforderungen durch spezielle Funktionen in MATLAB, was das Team dazu zwang, den Quelltext mehrmals zu ändern.

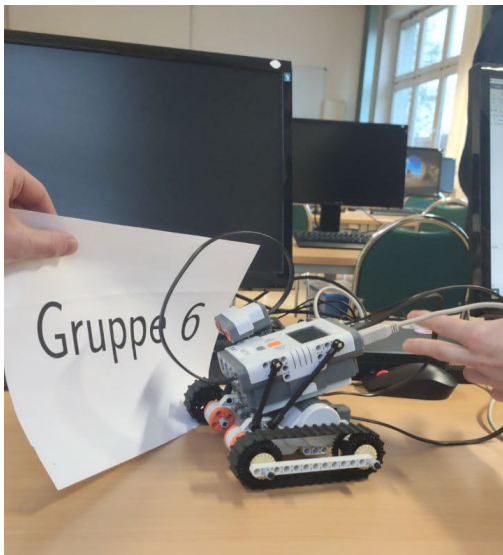


Abbildung 2. Erster Prototyp auf Raupenbasis

### III. KONSTRUKTION UND REALISIERUNG

#### A. Design und erster Prototyp

Ursprünglich war der Roboter auf Raupenbasis konzipiert, wobei zur Erhöhung der Stabilität ein Stein mit leichter vertikaler Abweichung befestigt wurde (siehe Abbildung 2). Anschließend wurde am Stein selbst eine zusätzliche Halterung angebracht, die den dritten Motor hielt, der für die Bewegung des Korbes mit dem Sensor zuständig ist. Unmittelbar nach dem Zusammenbau der Struktur wurden Probleme mit der Spannung der Raupen festgestellt, wodurch die Befestigungen, die die Rollen hielten, geschwächt wurden. Leider gelang es dem Roboter nach den ersten Tests nicht, ausreichend Geschwindigkeit zu entwickeln, und später stellte sich heraus, dass die gesamte Struktur nach einer kurzen Bewegung schwächer wurde (siehe Abbildung 2).

Aus diesem Grund wurde beschlossen, die Basis des Roboters zu ändern und zwei Motoren anzubringen, die dafür verantwortlich waren, den Roboter direkt zum Stein zu bewegen, sowie einen dritten, der den Korb mit dem Sensor drehte. Dadurch konnte die Anzahl der Verbindungen reduziert werden, was die Struktur stabiler machte und auch den Einfluss von Vibrationen durch Elektromotoren verringerte (siehe Abbildung 3).

#### B. Modul mit Sensor zur Farberkennung

Neben dem Radstand ist ein weiterer wichtiger Teil des Roboters der Farbsensor, dessen Aufgabe es ist, die Farbe des Balls zu lesen, der in den Korb fällt. Da alle Sensoren direkt mit dem Stein verbunden sein müssen, ist die Möglichkeit, sie um eine volle Umdrehung zu drehen, eingeschränkt. Aus diesem Grund wurde ein Drehwinkel von  $180^\circ$  gewählt (siehe Abbildung 4). Da die Teilnahme von zwei Personen oder zwei Teams geplant war, liest der Sensor beliebige Farben, aber erst wenn Blau oder Rot erkannt wird, stoppt der Roboter. Durch die sichere Montage des Korbes mit dem Sensor konnte sichergestellt werden, dass dieser auch bei einem direkten

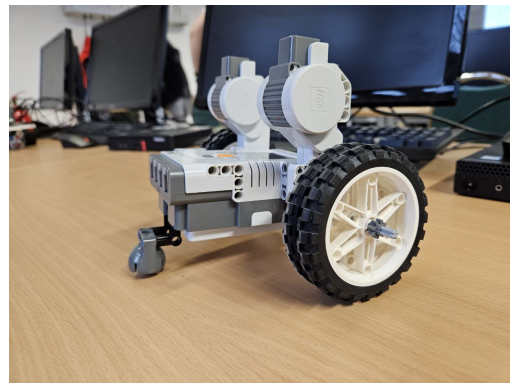


Abbildung 3. Verbesserte Version mit Radstand

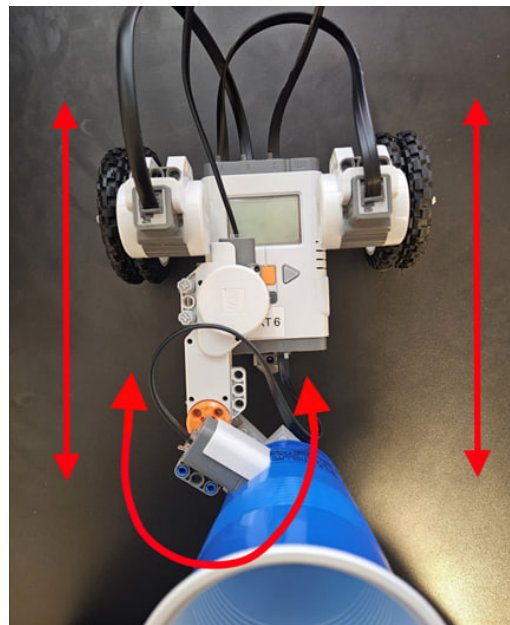


Abbildung 4. Zwei Motoren bewegen die Basis des Roboters und der dritte dient zum Bewegen des Sensorkorbs

Aufprall auf den Sensor an Ort und Stelle bleibt und bereit ist, die Farben der auftreffenden Bälle abzulesen. Lediglich bei einem Element des gesamten Roboters wurde kein Material verwendet, das in einem LEGO-Set enthalten wäre. Nämlich ein Plastikbecher, der als Korb fungiert, in den der Spieler den Ball werfen muss.

#### C. Quelltext in MATLAB

Beim Schreiben des Quelltextes wurde beschlossen, dem Roboter eine klare Bewegungsrouten zu geben. Dadurch konnte sich der Roboter in der Mitte eines  $0,8\text{ m}$  bis  $0,8\text{ m}$  Quadrats bewegen. Das einzige, was den Roboter und den Computer verband, war das Kabel, ohne das es unmöglich war, Befehle von MATLAB zu übertragen. Ein Versuch, Bluetooth zu verwenden, wurde ebenfalls unternommen, führte jedoch zu keinem positiven Ergebnis. Es konnte keine Verbindung zum drahtlosen Netzwerk durch den Roboter hergestellt werden, daher blieb ein Kabel die einzige Option. Der Quelltext besteht aus Elementen, die denen in der Abbildung 5 ähneln. Jedes

Element gibt die Richtung, Geschwindigkeit und Distanz an, die der Roboter zurücklegen muss.

Der gesamte Quelltext besteht aus kleinen Abschnitten, die angeben, in welche Richtung, mit welcher Geschwindigkeit sich die Räder drehen und wie sich der Sensorkorb dreht. Es wurde eine while-Schleife verwendet. Es war notwendig, eine Route in Form eines Quadrats anzulegen. Zusätzlich wurde "if ~OBJC.ReadFromNXT().IsRunning" verwendet, um den Sensorkorb zu bewegen, bei Stopps und Wendemanövern. Darüber hinaus wurde "if strcmp(color, 'BLUE')" und "elseif strcmp(color, 'RED')" verwendet, um Bedingungen zu schaffen, unter denen der Roboter anhält, wenn der Ball den Korb trifft, und ihn scannt. Außerdem wurden dem Quelltext Zeilen hinzugefügt, die einen Piepton mit verschiedenen Tönen abspielen, um zwischen blauen und roten Balltreffern zu unterscheiden (siehe Abbildung 5).

Es ist zu beachten, dass die Bewegung des Roboters nicht mit dem endet, was im folgenden Quelltextelement angegeben ist. Der Roboter dreht sich auch auf der Stelle, bewegt sich geradlinig in verschiedene Richtungen und mit unterschiedlicher Geschwindigkeit und führt auch andere Manöver aus. Aufgrund seiner Größe ist es nicht möglich, den gesamten Quelltext einzufügen.

```
while i < 4
if ~OBJC.ReadFromNXT().IsRunning
    OBJC.Power = -OBJC.Power;
    OBJC.TachoLimit = 180;
    OBJC.SendToNXT()

    OBJA = NXTMotor('A', 'Power', 50)
    OBJB = NXTMotor('B', 'Power', 50)
    OBJA.TachoLimit = 600;
    OBJB.TachoLimit = 600;
    OBJA.SendToNXT()
    OBJB.SendToNXT()
    OBJA.WaitFor()
    OBJB.WaitFor()

    OpenNXT2Color(port, 'FULL')
    color = GetNXT2Color(port)
    if strcmp(color, 'BLUE')
        NXT_PlayTone(300, 760)
        StopMotor('all', 'off')
        break
    elseif strcmp(color, 'RED')
        NXT_PlayTone(800, 760)
        StopMotor('all', 'off')
        break
end
```



Abbildung 5. Flunkyball-Roboter

#### IV. ERGEBNISDISKUSSION

Nachdem die Programmierung und Montage abgeschlossen waren, wurde der Roboter getestet. Im Korbbefestigungselement wurde eine strukturelle Schwachstelle identifiziert und anschließend behoben.

Nach mehreren Tests mit Leuten aus anderen Teams wurde entschieden, dass der Roboter alle Anforderungen erfüllte. Seine Bewegung wurde nur durch das mit dem Laptop verbundene Kabel begrenzt, und der Farbsensor bestimmte zuverlässig die Farbe des Balls, woraufhin sich der Roboter abschaltete.

#### V. ZUSAMMENFASSUNG UND FAZIT

Zusammenfassend lässt sich sagen, dass das endgültige Design des Geräts alle seine Ziele erreicht hat. Durch das Studium von MATLAB konnten wichtige Kenntnisse erworben werden, die in Folgeprojekten eine wichtige Rolle spielen werden. Der Roboter hat Entwicklungspotenzial. Zukünftig besteht Möglichkeit, den Bewegungsweg des Roboters im Code nicht mehr strikt vorzuschreiben, sondern Abstandssensoren hinzuzufügen, entlang derer sich der Roboter im Raum bewegt und bei Bedarf selbstständig Hindernisse ausweicht.

#### LITERATURVERZEICHNIS

- [1] Philippe Hurbain: *NXT® motor internals*.  
<https://www.philohome.com/nxtmotor/motor1-3.jpg> Version: Januar 2023
- [2] WIKIPEDIA: *Flunkyball*  
<https://de.wikipedia.org/wiki/Flunkyball> Version: Februar 2024