

# Self-Balancing Robot on a Ball

Der Roboter mit der Gelegenheit einer weiteren innovativen Entwicklung, der die Menschheit zukünftig assistieren kann.

Artem Hrach, Elektrotechnik und Informationstechnik  
Otto-von-Guericke-Universität Magdeburg

**Zusammenfassung** — Die Hauptidee dieses Projekt war die Entwicklung eines Roboters, der während des Projektseminars für Elektrotechnik/Informationstechnik 2025 aus Legobauteile (LEGO-Mindstorms) gebaut wurde und auf einem Ball balancieren kann, um weitere technische Entwicklungen in der Robotik zu fördern und den Menschen zu helfen.

Das Programm, das dieser Roboter in Bewegung setzen konnte, wurde in MATLAB geschrieben. Mit Hilfe des Programms kann Der Roboter jeden seiner nächsten Schritte erfassen, was besonders wichtig ist, da seine Funktion schnelle berechnung benötigt, um ein Umkippen zu verhindern.

Zum Abschluss des Projekts wurde ein Prototyp vom Roboter in einer Präsentation vorgestellt, bei der seine Funktionen in Aktion demonstriert wurden.

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

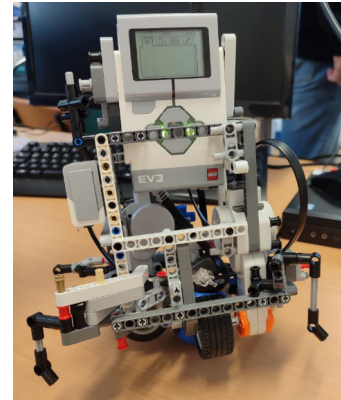


Abbildung 1. Komplett gebauter Roboter mit den Stützungsstangen

## I. EINLEITUNG

**D**er Balancier-Roboter erfordert eine präzise Steuerung und die Koordination mehrerer Motoren, um das Gleichgewicht aufrechtzuerhalten. Um stabil auf einer Kugel zu balancieren, muss er kontinuierlich seine Position und Neigung prüfen und entsprechende Korrekturen vornehmen. Dies erfordert eine schnelle Reaktionsfähigkeit und eine schwierige Regelungstechnik.

Gleichzeitig haben viele Menschen zu Hause nicht die Möglichkeit, komplexe Konzepte wie Gleichgewicht und Stabilität praktisch zu erforschen. Der Roboter ermöglicht es, diese Prinzipien anschaulich zu erleben, ohne teure Laborgeräte zu benötigen.

Falls eine Neigung erkannt wird, regelt er sich durch gezielte Motorbewegungen, um das Gleichgewicht wiederherzustellen. Dank dieses Mechanismus kann er auf einer Kugel balancieren, ohne umzukippen. Dies macht ihn nicht nur zu einem spannenden Demonstrationsobjekt für technische und physikalische Experimente, sondern auch zu einem unterhaltsamen Gerät für interaktive Anwendungen. Erster Versuch einen Roboter zu testen, ist auf der Abbildung 1 demonstriert.

## II. VORBETRACHTUNG

Um einen Roboter zu bauen und programmieren, bräuchte man zuerst die ganze Idee zu verstehen. Dabei auch was einem für die Ziele in Verfügung steht, um alles ordentlich zusammenzusetzen, mit die Möglichkeiten des LEGO-Mindstorm-Sets.

Sie können über Lego Mindstorms EV3 mehr auf Wikipedia [2] entdecken.

### A. Aufgabe des Roboters

Die Hauptfunktion des Roboters besteht natürlich darin, auf einem Ball balancieren, ohne übermäßige bewegung auszuführen. Falls dennoch solche unerwünschte Bewegung auftritt, dann bringen die Motoren den Roboter zurück ins Gleichgewicht, um diese Neigung zu vermeiden. Nachdem wir die wichtigste aufgabe erreicht haben, können wir andere aufgaben dazu hinzufügen. Beispielsweise, könnte der Roboter, mit Hilfe eines Tablett Gegenstände auf dem „Kopf“ transportieren oder einfach einige Objekte für bestimmte Zeit halten.

### B. Bestandteile

Die Teile, von denen der Roboter gebastelt wurde, sind:  
Legobauteile

2 Motoren  
2 Gyrosensoren  
2 Räder  
EV3-Stein  
Kabel inkl.(PC-USB verbindung)

Mehr info über die Teile [1]

### C. Funktionsweise des Roboters

Der Balancier-Roboter funktioniert durch eine präzise Kombination aus Sensorik, Regelungstechnik und Antriebssystemen, um dauerhaft auf einer Kugel zu balancieren. Seine Hauptkomponenten sind:

1. Gyrosensoren zur Lageerkennung: Der Roboter verwendet zwei Gyrosensoren, um kontinuierlich die Neigung und Rotationsbewegung zu messen, was hilft ihm die Situation zu verstehen. Falls eine Abweichung vom Gleichgewichtszustand erkannt wird, sendet das System sofort Signale an die Motorsteuerung, die dann ihn zurückziehen sollen.

2. Regelungssystem: Der wichtigste Teil von Befehlsgebung ist ein PID-Controller (Proportional-Integral-Diverivative-Regler) verarbeitet die Sensordaten und berechnet in Echtzeit die notwendigen Korrekturen, um das Gleichgewicht wiederherzustellen. Durch schnelles Nachjustieren der Motoren kann der Roboter auch auf kleinen Neigungen reagieren und stabil bleiben.

3. Motoren und Antriebsmechanismus: Der Roboter ist mit mehreren Motoren ausgestattet, die unabhängig voneinander gesteuert werden. Sie bewegen sich in verschiedene Richtungen, um das Gleichgewicht zu halten, indem sie kleine Anpassungen auf der Kugel vornehmen. Je nach Konstruktion können dies omnidirektionale Räder oder spezielle Rollen sein, die eine präzise Bewegung ermöglichen.

### III. TECHNISCHE UMSETZUNG

Der Roboter ist ein Prototyp, weil er noch nicht fertiggestellt wurde, und kann nicht seine wichtigste Funktion richtig umsetzen. Es gab Herausforderungen auf unserem Weg, sowohl mit der Konstruktion als auch mit der Programmierung in MATLAB Software.

#### A. Konstruktion des Roboters

Der Roboter wurde solche Weise gebastelt, dass er richtige Gewichtsverteilung hat. Seine Form erinnert an ein Quadrat bzw. länglicher kubischer Körper, was hilft bei der Verteilung und somit kann man die bauteile, wie Motoren und Sensoren in bestimmte Positionen aufsetzen. Die Räder befestigten sich auch in spezifischen Positionen, damit wenn der Roboter in verschiedenen Richtungen neigt, kann ein einziger Motor je vom bestimmten Winkel sich drehen und gekippte Seite zurück in die Anfangsposition zu bringen. Abbildung 2 zeigt besseren Anblick der Motoren.

#### B. Motoren und Gyrosensoren

1. Motoren. Sie sorgen für gezielte Bewegungen, um das Gleichgewicht auf dem Ball zu halten. In diesem Fall werden Servomotoren verwendet. Großer EV3-Servomotor ist ein leistungsstarker Motor mit eingebautem Rotationssensor, der eine Genauigkeit von einem Grad aufweist.

1. Gyrosensoren. Er misst die Drehgeschwindigkeit und Neigung des Roboters. Wenn der Roboter sich zur Seite neigt, erkennt der Gyrosensor diese Bewegung und sendet ein Signal an den EV3-Stein Controller. Dieser berechnet die nötige Korrektur und gibt den Motoren Anweisungen, um das Gleichgewicht wiederherzustellen.

Die Kombination aus Gyrosensoren und Motorsteuerung ermöglicht es dem Roboter, schnell auf Veränderungen zu reagieren und stabil auf dem Ball zu balancieren.

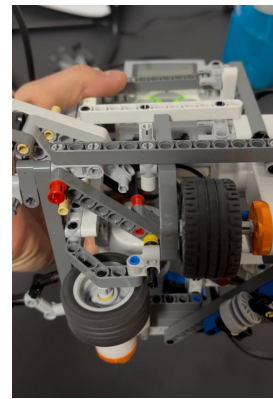


Abbildung 2. Konstruktion (Anblick von unten)

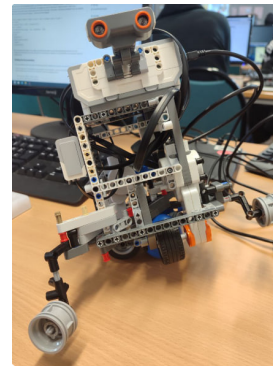


Abbildung 3. Problem mit dem Schwerpunkt

#### C. Probleme und Herausforderungen

Unsere größte Herausforderung war in MATLAB programmieren. Natürlich war dies auch die Hauptidee des Projekts - Code schreiben zu können und konzept des Programms zu verstehen. Danach überlegten wir uns, wie man den Roboter in die Balance bringen könnte. Dabei half uns der PID-Controller, jedoch mussten wir auch die richtigen  $k_i, k_p$  und  $k_d$  Werte zu finden. Ein weiteres Problem gab es mit der Konstruktion, insbesondere mit dem EV3-Stein, da er zu schwer war und dadurch der Schwerpunkt gestört wurde (Abbildung 3). Um dies Problem zu vermeiden, schoben wir ihn so weit nach unten wie möglich - und das gelang uns. Das größte und problematischste Herausforderung war jedoch, dass die Gyrosensoren nicht richtig funktionieren. Nämlich die Werte steigen bei jedem Testlauf an. Deshalb konnte man keine verlässlichen Anfangswerte bestimmen. Beispielsweise, wenn Anfangswert  $0^\circ$  und  $1^\circ$  ist, dann zeigt der Sensor nach mehreren Tests bereits  $2^\circ$  und  $-4^\circ$  an. Das lässt sich auf dargestellten Diagramm auch gut anzeigen (Abbildung 4).

### IV. PROGRAMMIERUNG IN MATLAB

Der Code soll so funktionieren, dass wenn der Roboter sich in eine beliebige richtung kippt, dann werden ausgemessenen Werte der beiden Gyrosensoren mit Hilfe eines Befehls ausgelesen und die Motoren in Bewegung setzen. Welcher Motor seiner Drehung startet, hängt vom Neigungswinkel ab. Beispielsweise, wenn der Roboter nach vorne neigt, sodass der Wert von des ersten Gyrosensors positiv und der des zweiten

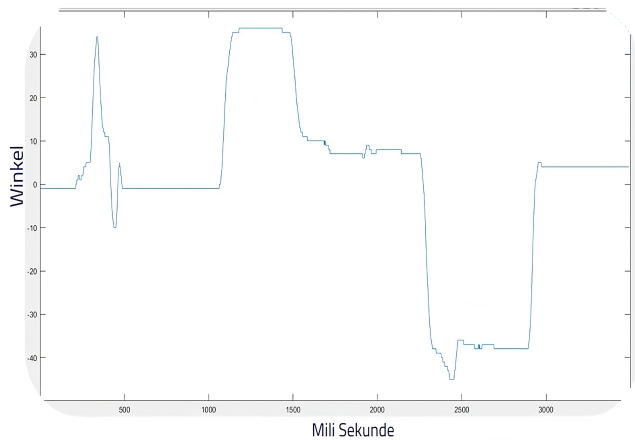


Abbildung 4. Diagramm der wechselnden Werte.

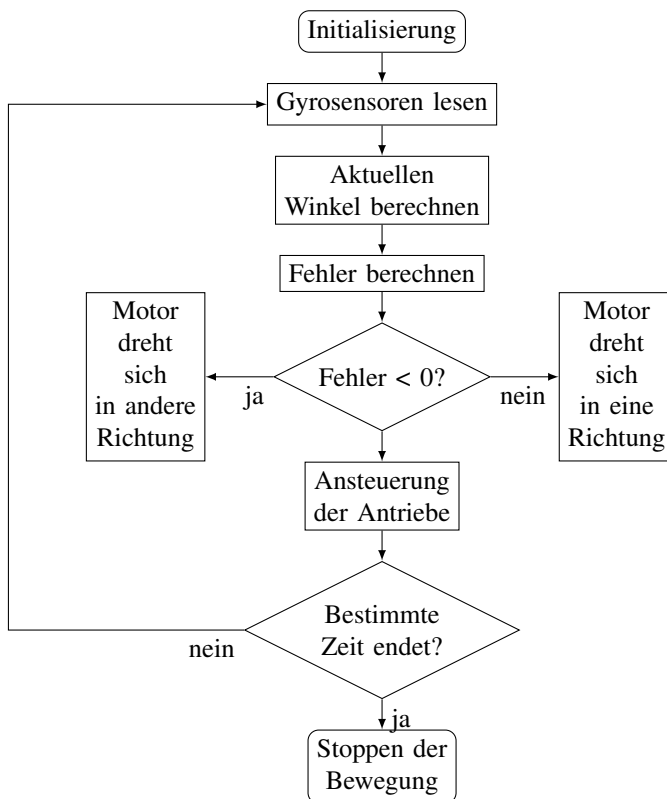


Abbildung 5. Verfolgungsalgorithmus des Programms für Balancier-Roboter

negativ ist, dann dreht sich nur Frontmotor, da in diese Situation kein Backmotor benötigt wird. Genauer kann man auf der Abbildung 5 sehen. Der Kern des Programms ist der PID-Controller. Er hat dafür auch 3 Werte.

1) *Der Proportional-Anteil  $k_p$* : Reagiert direkt auf die aktuelle Abweichung vom Sollwert (z. B. gewünschter Neigungswinkel ist gleich  $0^\circ$ ). Je größer  $K_p$ , desto stärker reagiert das System auf eine Abweichung. Ist  $K_p$  jedoch zu hoch, kann es zu Schwingungen oder Instabilität kommen.

2) *Der Integral-Anteil  $k_i$* : Summiert kleine Abweichungen über die Zeit auf. Korrigiert systematische Fehler (Offset), die der Proportional-Anteil allein nicht ausgleichen kann. Ein zu hoher  $K_i$ -Wert kann zu Überschwingungen oder trägern Verhalten führen.

3) *Der Differential-Anteil  $k_d$* : Betrachtet, wie schnell sich der Fehler ändert. Dämpft abrupte Bewegungen und hilft, frühzeitig gegenzusteuern. Ist  $K_d$  zu hoch, kann das System nervös oder ruckartig reagieren.

Der PID-Regler vergleicht ständig die tatsächliche Neigung mit der idealerweise  $0^\circ$ -Neigung. Basierend auf  $k_p$ ,  $k_i$  und  $k_d$  berechnet er das Korrektursignal für die Motoren. So kann der Roboter selbst bei leichten Stößen oder unebenen Untergründen schnell reagieren und im Gleichgewicht bleiben.

## V. ERGEBNISDISKUSSION

Nachdem alles erledigt war, wurde der Roboter getestet. Es war auch eindeutig klar, dass der Code funktioniert und die Motoren drehen sich relativ zum Winkelswert. Aber natürlich konnte das nicht lang gut laufen, denn es gibt das Problem mit Winkel und zwar mit einer Fehlerberechnung. Eine Lösung für dieses Problem ist noch nicht gefunden, aber uns ist eine Idee eingefallen. Man kann den Code direkt auf dem EV3-Stein übertragen und dann noch einige Tests machen. Es könnte natürlich sein, dass MATLAB kann nicht so schnell rechnen oder die Gyrosensoren übertragen ihre Werte nicht so schnell wie es bedarf.

## VI. ZUSAMMENFASSUNG UND FAZIT

Der Roboter ist ein beeindruckendes Beispiel dafür, wie Technik und Mathematik zusammenwirken, um autonome Systeme stabil und reaktionsschnell zu gestalten. Man bekam neue Erfahrung mit dem MATLAB-Software, was auch in der Zukunft nützlich und wertvoll für weitere Projekte sein werden. Ein sehr spannendes Projekt, das einem beibringt, wie man Probleme löst und gleichzeitig hilft, moderne Kenntnisse in der Roboterprogrammierung zu erwerben. Diese Erkenntnisse sind nicht nur für den Bereich der Robotik relevant, sondern bieten auch einen praxisnahen Einblick in moderne Steuerungstechniken. Zum Beispiel, Mobile Transportmittel: Ähnliche Prinzipien wie bei selbstbalancierenden Segways werden genutzt, um kompakte, wendige Fahrzeuge zu entwickeln. A

## ANHANG

Nur kurze Anhang, wie das Programm PID berechnet:

```

error1 = desiredAngle1 - currentAngle1;
error2 = desiredAngle2 - currentAngle2;
integral1 = integral1 + error1 * elTime;
integral2 = integral2 + error2 * elTime;
der1 = (error1 - prevError1) / elTime;
der2 = (error2 - prevError2) / elTime;
  
```

## LITERATURVERZEICHNIS

- [1] . EV3-Mindstorm Roboterteile. 2025. URL: <https://describd.com/document/422734718/RobotParts-grade10>.
- [2] Wikipedia contributors. *Lego Mindstorms EV3* — Wikipedia, The Free Encyclopedia. März 2025. URL: [https://en.wikipedia.org/wiki/Lego\\_Mindstorms\\_EV3](https://en.wikipedia.org/wiki/Lego_Mindstorms_EV3).