Mobiler Solartracker mit LEGO Mindstorms

Lennart Thies Christian Brehmer, Elektromobilität
Otto-von-Guericke-Universität Magdeburg

***Zusammenfassung*—Anlässlich des Projektseminars Elektrotechnik/Informationstechnik an der Otto-von-Guericke-Universität wurde ein Roboter entwickelt, welcher in der Lage ist eine Ebene abzufahren und sich in dieser zum Licht hin auszurichten. In folgender Ausarbeitung wird die Realisierung mithilfe von LEGO Mindstorms und MATLAB sowie die Ansteuerung über eine grafische Benutzeroberfläche näher betrachtet. Des weiteren wird auf die entstandenen Probleme und deren Lösung eingegangen. Zuletzt werden mögliche Weiterentwicklungen und praxisnahe Einsatzgebiete diskutiert.**

***Schlagwörter*—Elektromobilität, LEGO, MATLAB, Solar, Solartracker, Zukunft**

I. Motivation

WÄHREND die Autoindustrie das Zeitalter des Verbrennungsmotors hinter sich lässt und Milliarden von Euro in neue elektrisch betriebene Automobile investiert [1] stellt sich die Frage wie die Menschheit den neuen Bedarf an Energie decken kann. In Deutschland wurden 2020 zwar schon 46% der gesamten Stromversorung aus erneuerbaren Rescourcen gewonnen [2], allerdings ist dies noch unzureichend um den immer näher kommenden irrevisiblen Klimawandel aufzuhalten. Eine mögliche Lösung ist die verstärkte Förderung von Solarenergie. Dafür müssten zuerst die momentanen Probleme dieser alternativen Energiegewinnung gelöst werden [3]. Unter anderem kann nur während des Tages die Sonne als brauchbare Energiequelle benutzt werden. Dies ist nicht ideal, da der größte Energieverbrauch am Abend und in der Nacht auftritt. Nicht bewegliche Solaranmodule verringern zudem ihre Effizienz im Laufe des Tages, da sich der Einfallswinkel der Sonne ändert. Moderne Solaranlagen besitzen daher sogenannte Solartracker (siehe Abschnitt II-A), welche diese zum Stand der Sonne hin ausrichten. Der Roboter soll diese weiterentwickeln und eine weitere Ebene an Mobilität zu solchen Solartrackern hinzufügen. Somit könnte das Problem der sich ändernden Wetterlage und die damit auftretenden Veränderungen der Lichtverhältnisse, welche den Ertrag an Energie drastisch verringern, gelöst werden. Um dies zu realisieren, soll sich der Roboter nicht nur stationär zur Sonne hin ausrichten, sondern auch eine Ebene abfahren, um so aus schattigen Stellen zu entkommen können.

II. Vorbetrachtungen

Im folgendem Abschnitt wird der aktuelle Stand moderner Solartracker kurz vorgestellt sowie ein kleiner Einblick in die verschiedenen LEGO-Mindstorms-Varianten gewährleistet.

A. Solartracker

Um maximale Effizienz moderner Solaranlagen zu erhalten, werden Solartracker verwendet. [4] Diese sind in der Lage ein Objekt, wie z. B. ein Solarmodul, sehr genau zur Sonne hin auszurichten, damit der Einfallswinkel zwischen Solarplatte und dem Sonnenlicht möglich gering ist. Somit soll der Energieverlust sehr stark verringert werden. Man unterscheidet weithin zwischen einachsigen und zweiachsigen Systemen, wobei erstere aufgrund ihres Preises weiter verbreitet sind. Ein einfaches Beispiel ist in Abbildung 1 zu sehen. Obwohl die zweiachsigen System effektiver sind, machen die höheren Produktionskosten eine kommerzielle Nutzung unattraktiver.

Abbildung 1. Solartracker von Suntactics [5]

B. LEGO Mindstorms

LEGO Mindstorms ist eine programmierbare Produktreihe des dänischen Spielwarenherstellers LEGO, um erste Erfahrungen in dem Bereich der Robotik zu erhalten [6]. Es können mithilfe eines zentralen Computers verschieden Sensoren und Motoren angesteuert werden. Angefangen hat die Produktreihe im Jahr 1998 mit dem RCX-Baustein, welcher 2006 von dem leistungsstärkeren NXT-Baustein ersetzt wurde, welchen in Abbildung 2 zu sehen ist. Die aktuellste Variante, der EV3-Baustein, kam 2013 auf dem Markt. Die verschieden Systeme unterscheiden sich in der Leistungsstärke ihrer eingebauten Prozessoren sowie Speichern. Die Anzahl kompatibler Sensoren und Motoren hat sich über die Jahre ebenfalls erhöht [7]. Beide Bausteine besitzen ein auf Linux basierendes Betriebssystem. Wobei der EV3-Baustein standardmäßig mithilfe von Python angesteuert wird, wurde für den NXT-Baustein eine eigene Programmierumgebung namens „NXT-G“ angeboten, welche auf LabView basiert [8]. Es werden auch andere Programmiersprachen wie MATLAB unterstützt. Allerdings benötigt man hierfür z. B. die von der RTWH Aachen entwickelten Bibliothek [9].

III. Realisierung

In diesem Abschnitt wird auf die Realisierung des Roboters eingegangen. Hierfür wird zuerst ein kurzer Überblick der

DOI: 10.24352/UB.OVGU-2021-039

Abbildung 2. Lego-Mindstorms NXT-Baustein [10]

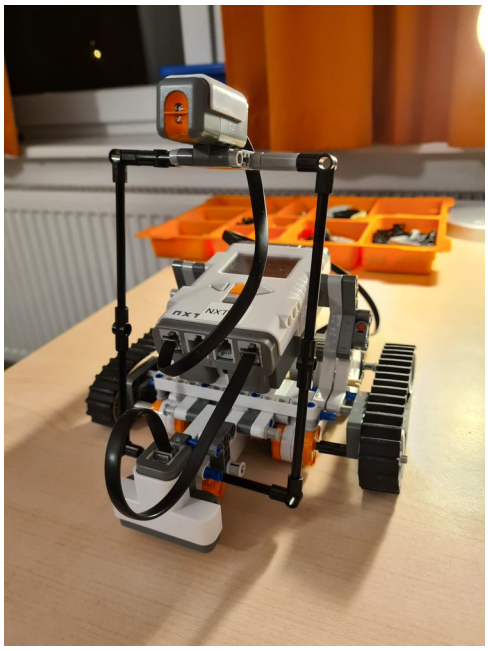

Abbildung 3. Konstruktion des Roboters

Konstruktion betrachtet. Danach werden die automatisierten Prozesse vorgestellt. Zuletzt wird die grafische Benutzeroberfläche erläutert und deren Funktion im Detail erklärt.

A. *Konstruktion*

Bei der Konstruktion des Roboters, welche bei Abbildung 3 zu sehen ist, wurde vor allem auf Funktionalität geachtet. Daher werden zur Fortbewegung zwei Ketten verwendet, welche jeweils von einem Motor angetrieben werden und somit separat ansteuerbar sind. Dadurch ist es dem Roboter möglich sich auf der Stelle zu drehen. Die Konstruktion besitzt zudem einen dritten Motor, welcher über zwei Stangen den Lichtsensor hoch und runter bewegen kann. Besagter Motor wurde zentral an der Vorderseite des Roboters angebracht, um ein eventuelles Anschließen eines Solarmodules zu simulieren. An der Vorderseite wurde ebenfalls zentral ein Ultraschallsensor befestigt. Dieser zeigt in einem 90° Winkel nach unten und soll zur Kantenerkennung dienen. Gesteuert wird der Roboter über den zentral mittig angebrachten NXT-Baustein.

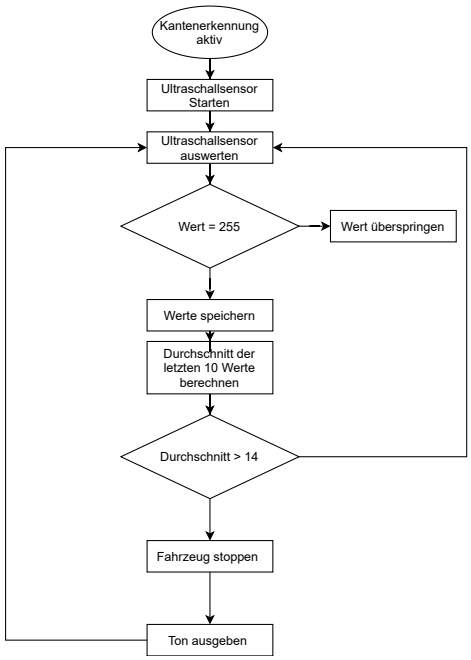

Abbildung 4. Programmablaufplan der Kantenerkennung

B. *Programmierung*

1) Kantenerkennung: Der Roboter verfügt über eine Kantenerkennung, wie in Abbildung 4 beschrieben, welche mithilfe eines Ultraschallsensors realisiert wurde. Hierfür misst der Sensor konstant die Entfernung zum Untergrund und speichert die Werte in einem Array ab. Sobald die ersten zehn Werte abgespeichert wurden, wird der Durchschnitt aus diesen berechnet, um Ungenauigkeiten des Sensors auszugleichen. Aus dem selben Grund wird auch eine Fallunterscheidung bei dem Maximalwert vorgenommen, welche diesen in der Berechnung überspringt. Wenn vom Roboter eine Kante erkannt wird, werden die Motoren, welche die Ketten ansteuern, umgehend gestoppt und es wird ein Ton abgespielt. Dieser wird solange abgespielt bis keine Kante mehr erkannt wird.

2) Lichtwert plotten: Eine weitere Funktion des Roboters ist es, mithilfe des Lichtsensors Werte zu messen, zu verwerten und auszugeben. Letzteres erfolgt über ein Koordinatensystem. Damit das Plotten des Funktionsgraphen in Echtzeit erfolgen kann, müssen die Lichtwerte in einem Array gespeichert und mit jeder Ausführung des Skripts um einen Wert erweitert werden.

3) Reset des Lichtsensors: Der Lichtsensor des Roboters wird beim Start auf einen bestimmte Anfangsposition gebracht, damit dieser nicht vor der Konstruktion sitzt und die Stabilität während des Fahrens beeinflusst. Hierfür wird der Sensor solange hochgefahren bis die Drehzahl des Motors 0 beträgt. Dieses wurde realisiert, indem sowohl die Position des Motors an zwei verschieden Zeitpunkten, als auch die dafür benötigte Zeit gemessen und daraus die Drehzahl berechnet wurde. Um Ungenauigkeiten zu vermeiden, wird auch hier der Durchschnittswert betrachtet. Sobald die ausgerechnete Drehzahl den Schwellenwert erreicht hat, wird das Tacholimit auf 1 gesetzt und der Motor wird gestoppt. Das wird gemacht, da

die Motoren sonst ihn einer Endlosschleife gefangen wären, was die weitere Ansteuerung unmöglich machen würde.

4) Automatisches Ausrichten auf der Stelle: Der Roboter ist in der Lage sich automatisch stationär zum Licht auszurichten. Hierfür dreht er sich einmal um 360° und misst dabei die Lichtintensität mithilfe des Lichtsensors. Gleichzeitig wird die Postion der Motoren ausgewertet, welche die Ketten antreiben. Die Werte werden in Arrays gespeichert und nachdem der Roboter seine erste Drehung vollendet hat dreht er sich zu der Position zurück, an der der höchste Lichtwert gemessen wurde.

5) Automatisches Abfahren einer Ebene: Beim automatischen Abfahren einer Ebene fährt der Roboter zuerst vorwärts bis er eine Kante erkennt (siehe Abschnitt III-B1). Sobald dies geschehen ist, fährt der Roboter kurz rückwärts und dreht sich um 180°. Gleichzeitig setzt er einen internen Zähler hoch, welcher für die Anzahl erkannter Kanten steht. Nun fährt der Roboter wieder vorwärts bis er eine weitere Kante erkennt. Allerdings werden jetzt mithilfe des Lichtsensors die Lichtwerte und die dazugehörige Postionen in einem Array gespeichert. Sobald eine weitere Kante erreicht wurde, schließt sich der Lichtsensor und der Roboter dreht sich wieder um 180°. Anschließend fährt er zu der Postion zurück, an der er den höchsten Lichtwert gemessen hat und dreht sich dort um 90°. Daraufhin wiederholt der Roboter die selben Schritte, um auch hier die Postion mit der größten Lichtintensität zu finden. Wenn diese gefunden wurde, dreht sich der Roboter allerdings nicht mehr um 90° sondern führt die Funktion zur automatischen Ausrichtung auf der Stelle aus (siehe Abschnitt III-B4). Zuletzt wird der Lichtsensor nochmals separat ausgerichtet, indem er von unten nach oben bewegt wird. Der passenden Programmablaufplan für das automatische Abfahren einer Ebene ist bei Abbildung 5 zu sehen.

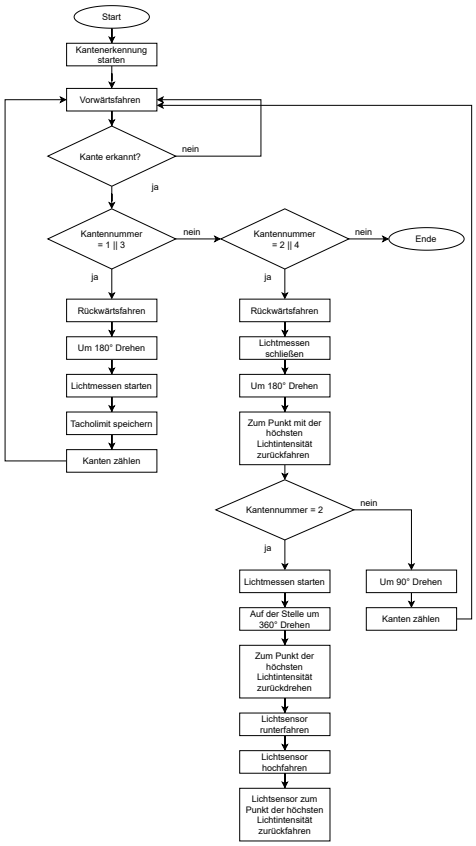

Abbildung 5. Programmablaufplan des automatischen Abfahren einer Ebene

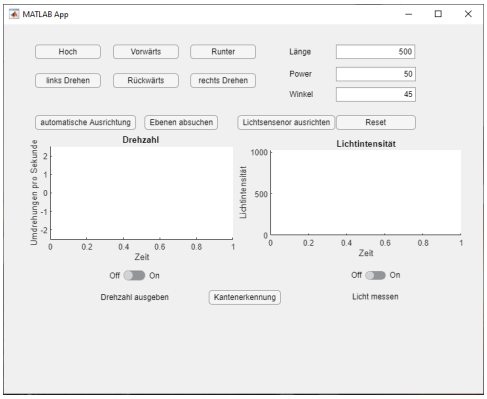

Abbildung 6. GUI zur Steuerung des Roboters

C. Grafische Benutzeroberfläche

Der Roboter kann komplett über die in Abbildung 6 zu sehenden Benutzeroberfläche gesteuert werden, welche mithilfe des in MATLAB vorhandenem App Designer erstellt wurde. Links oben kann der Roboter manuell gesteuert werden. Hierbei kann nicht nur die Bewegung des gesamten Roboters gesteuert werden, sondern auch die Ausrichtung des Lichtsensors. Oben rechts befinden sich Eingabefelder, welche benutzt werden um den Motoren-Parameter wie z. B. den gewünschten Winkel zu übermitteln. Damit diese nicht bei jedem Start erneut eingegeben werden müssen, wurden Startwerte festgelegt. Unter den Knöpfen befinden sich die Ansteuermöglichkeiten für die automatisierten Bewegungsabläufe. Dort kann neben dem automatischen Absuchen der Ebenen (siehe Abschnitt III-B5) auch die Zurücksetzung des Lichtsensors (siehe Abschnitt III-B3) oder das automatische Ausrichten auf der Stelle (siehe Abschnitt III-B4) angesteuert werden. Darunter können links die Drehzahl der Motoren und rechts die vom Lichtsensor gemessenen Werte in Echtzeit graphisch ausgewertet werden. Um dies zu steuern, befindet sich unter jedem Koordinatensystem ein Schalter. Zuletzt kann die Kantenerkennung (siehe Abschnitt III-B1) mittels eines Zustandsknopfes bedient werden.

IV. Ergebnisdiskussion

Am Ende des Projektes ist es dem Roboter möglich, eine Ebene wie z. B. einen Tisch abzufahren, ohne von dort herunter zu fallen. Dabei kann er die Lichtintensität mithilfe eines Lichtsensors messen und sich die Position auf dem Tisch merken, an der der Maximalwert gemessen wurde und sich automatisch dorthin zurückbewegen. So ist es ihm möglich sich z. B. automatisch aus einer schattigen Stelle herauszubewegen. Wie in diesem Video [11] zu sehen, fährt der Roboter nicht sehr gerade. Dies liegt zum einem an der Konstruktion, da die Ketten nicht perfekt symmetrisch sind und der NXT-Baustein nicht schnell genug die MATLAB-Befehle bearbeiten kann,

zum anderen daran, dass die Drehung über die Eingabe eines bestimmten Tacholimits erfolgt, welches durch Ausprobieren nur bedingt gut einen genauen Drehwinkel beschreiben kann. Eine Möglichkeit dies zu lösen wäre die Nutzung eines Gyrosensors, um genauere Daten zum Drehwinkel zu erhalten. Ein weiteres Problem ist die Kabelverbindung von Roboter zu Computer. Diese kann theoretisch durch eine Bluetooth-Verbindung ausgetauscht werden. Allerdings führt die dadurch erhöhte Latenz dazu, dass wichtige Funktionen wie die Kantenerkennung (siehe Abschnitt III-B1) nicht funktionieren, da die Daten der Sensoren zu lange brauchen, um vom NXT-Baustein zum Computer gesendet zu werden und der Computer diese zwar auswerten, aber erst zu spät neue Befehle zurück senden kann. Des weiteren ist es dem Roboter nicht möglich, gleichzeitig Funktionsgraphen zu plotten und die Kantenerkennung aktiv zu haben, da beide Funktion mithilfe von Endlosschleifen realisiert wurden und daher nicht parallel ausgeführt werden können. Bei dem automatischem Abfahren einer Ebene ist das kein Problem, da diese Funktion in die der Kantenerkennung (siehe Abschnitt III-B1) mit integriert ist und somit mithilfe von Verschachtelungen in der Schleife mit ausgeführt wird. Eine mögliche Lösung könnte hier die parallel Computing Toolbox [12] oder die Nutzung eines Interrupt-Befehls sein.

V. Zusammenfassung und Fazit

Während des Projektseminars LEGO Mindstorms wurde erfolgreich ein Roboter konstruiert und programmiert, welcher in der Lage ist, automatisch in einer Ebene den hellsten Punkt zu suchen und sich dorthin zubewegen. Im Zuge dessen traten vor allem durch technischen Limitationen der Sensorik und des NXT-Bausteins Probleme auf, welche nur bedingt gelöst wurden konnten. Des weiteren kann man den Roboter über eine grafische Benutzeroberfläche ansteuern. In dieser sind nicht nur die automatisierten Prozesse verfügbar sondern auch die manuelle Ansteuerung sowie die Auslesung der Drehzahl der verschieden Motoren und die vom Lichtsensor gemessene Lichtintensität mittels Funktionsgraphen in Echtzeit. In Zukunft könnte im Rahmen unter dem Lichtsensor ein Solarmodul angebracht werden, welches Elektrizität in einen Energiespeicher senden und so z. B. den Roboter selbst laden könnte.

Anhang

Im Rahmen des Projektseminars wurde eine Playlist erstellt, welche die Funktionen aller entstandenen Roboter darstellt [13].

Quellenverzeichnis

[1] Germis, Carsten: *VW nimmt das Rennen mit Tesla in der Elektromobilität auf*. 2020 https://www.faz.net/aktuell/wirtschaft/auto-verkehr/vw-investiert-35-milliarden-euro-in-elektromobilitaet-17051526.html

[2] Energie, Bundesministerium für Wirtschaft u.: *Erneuerbare Energien*. 2021 https://www.bmwi.de/Redaktion/DE/Dossier/erneuerbare-energien.html

[3] Follett, Andrew: *Top 11 problems for wind and solar*. 2015 https://www.bmwi.de/Redaktion/DE/Dossier/erneuerbare-energien.html

[4] Wikipedia: *Solar tracker*. 2021 https://en.wikipedia.org/wiki/Solar_tracker

[5] Adsala: *Suntastics Solar tracker*. 2016 https://commons.wikimedia.org/wiki/File:Suntactics_solar_tracker.jpg

[6] LEGO: *LEGO MINDSTORMS*. 2021 https://education.lego.com/de-de/products/lego-mindstorms-education-ev3-set/5003400#ev3-set

[7] Wikipedia: *Lego Mindstorms*. 2021 https://de.wikipedia.org/wiki/Lego_Mindstorms

[8] Wikipedia: *Lego Mindstorms NXT*. 2021 https://de.wikipedia.org/wiki/Lego_Mindstorms_NXT

[9] RWTH: *Mindstorms NXT Toolbox*. 2019 https://www.mindstorms.rwth-aachen.de/

[10] Löwenstein, Bernhard: *LEGO MINDSTORMS NXT*. 2013 https://jaxenter.de/lego-mindstorms-nxt-2662

[11] Magdowski, Mathias: *Lichtfinder aus dem Lego-Praktikum 2021 an der OVGU Magdeburg sucht die hellste Stelle einer Fläche*. 2021 https://www.youtube.com/watch?v=9ruyJ5D16nY

[12] TheMathWorks: *Parallel Computing Toolbox*. 2021 https://de.mathworks.com/products/parallel-computing.html

[13] Magdowski, Mathias: *Lego-Praktikum 2021*. 2021 https://www.youtube.com/watch?v=RAflL1oAXsM&list=PLWCaO6Bpqy-7nJr45PQG828J6OdIsfCIi