DHBW
Duale Hochschule
Baden-Württemberg
Ravensburg
Campus Friedrichshafen

Realisierung eines Vier-Gewinnt Roboters

ggf. Untertitel mit ergänzenden Hinweisen

Studienarbeit T3_3200

Studiengang Elektrotechnik

Studienrichtung Automation

Duale Hochschule Baden-Württemberg Ravensburg, Campus Friedrichshafen

von

Simon Gschell / Patrik Peters

Abgabedatum:	3. Juli 2025
Bearbeitungszeitraum:	01.01.2025 - 31.06.2025
Matrikelnummer:	123 456
Kurs:	TEA22
Betreuerin / Betreuer:	Prof. Dr. ing Thorsten Kever

Erklärung

gemäß Ziffer 1.1.14 der Anlage 1 zu §§ 3, 4 und 5 der Studien- und Prüfungsordnung für die Bachelorstudiengänge im Studienbereich Technik der Dualen Hochschule Baden-Württemberg vom 29.09.2017 in der Fassung vom 24.07.2023.

Ich versichere hiermit, dass ich meine Studienarbeit T3_3200 mit dem Thema:

Realisierung eines Vier-Gewinnt Roboters

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Friedrichshafen, den 3. Juli 2025

Simon Gschell / Patrik Peters

Kurzfassung

Diese Studienarbeit wurde im Rahmen der sechsten Akademiephase angefertigt. Im ersten Teil der Arbeit wurden verschiedene Spieltheorien miteinander verglichen. Aufbauend auf den Erkenntnissen der ersten Studienarbeit, lag der Schwerpunkt diesmal in der praktischen Umsetzung eines Vier-Gewinnt-Roboters. Das Ziel dieser Arbeit bestand darin, einen Roboter zu entwickeln, der eigenständig Spielzüge beim Spiel „Vier gewinnt“ ausführen kann und somit in der Lage ist, gegen einen anderen Roboter anzutreten.

Für die Realisierung des Projekts wurde ein LEGO Spike Prime Set verwendet. Die Konstruktion des Roboters besteht aus verschiedenen zusammengesetzten LEGO-Bauelementen. Vereinzelt wurden auch Elemente selbst entworfen und mittels 3D-Drucker angefertigt. Die Steuerung und Überwachung der Aktoren, wie zum Beispiel der Motoren und Sensoren, erfolgt mithilfe eines LEGO Spike Hub. Dieser Mikrocontroller verarbeitet die eingehenden Sensordaten und steuert die Bewegungen des Roboters entsprechend der programmierten Logik.

Die Programmierung erfolgte in der Sprache MircoPython in der LEGO Spike App. Das Kernstück des Programms ist der Alpha-Beta-Algorithmus, mit ihm wird der nächste optimale Zug berechnet. Durch die Kombination aus mechanischer Konstruktion und programmierter Software entstand ein funktionsfähiger Prototyp, der die gestellten Anforderungen erfüllt und einen Spielzug eigenständig ausführen kann.

Abstract

English translation of the „Kurzfassung“.

Inhaltsverzeichnis

1 Einleitung

2 Grundlagen

In diesem Kapitel werden die zentralen Begriffe und Methoden ausführlich vorgestellt. So wird das notwendige Grundlagenwissen vermittelt, auf dem die weitere Ausarbeitung aufbaut.

2.1 Mikropython

MicroPython ist eine speziell für Mikrocontroller angepasste Version der Programmiersprache Python. Im Gegensatz zur Desktop-Variante lässt sich MicroPython-Code direkt auf Hardware mit begrenzten Ressourcen ausführen.[SS23][PLJ23] Im Unterschied zu Standard-Python 3 bringt MicroPython allerdings nur einen Teil der gewohnten Python-Standardbibliotheken mit. Wodurch es weniger Speicherplatz benötigt. Zudem besitzt MircoPython einen eigenen Interpreter, der direkt auf einem Mirkocontroller ausgeführt werden kann. Dadurch eignet sich MicroPython besonders gut für die Programmierung des LEGO Spike Hubs.[Bel24]

2.2 der Mikrocontoller

LEGO® spike Hub:

Der Große LEGO® Technic Hub ist mit einem leistungsfähigen 32-Bit ARM Cortex-M4 Mikrocontroller ausgestattet, der mit einer Taktfrequenz von 100 MHz arbeitet. Darüber hinaus verfügt der Hub über 1 MB Flash-Speicher und 128 KB RAM, was für viele schulische und experimentelle Anwendungen mehr als ausreichend ist. Sechs Anschlüsse für Motoren und Sensoren sowie Bluetooth Low Energy (BLE) zur drahtlosen Kommunikation machen den Hub vielseitig einsetzbar.

Der Hub ist das Steuerungselement des Spike Prime Systems. Er umfasst sechs Ein-/Ausgänge, welche den Anschluss von Peripheriegeräten, wie Sensorik und Aktorik, ermöglicht. Mit einem Micro-USB-Anschluss und Bluetooth wird die Kommunikation mit kompatiblen Endgeräten hergestellt. Der Hub besitzt ein integriertes MicroPython-Betriebssystem mit einem 100-MHz-Prozessor. Weitere Ausstattungen sind:

- Individuell anpassbaren Lichtmatrix (5x5)
- Aufzeigen von wichtigen Informationen und Statusmeldungen
- Tasten ermöglichen eine einfache Navigation und Steuerung durch Menüs
- Lautsprecher
- 6-achsigen Kreiselsensor

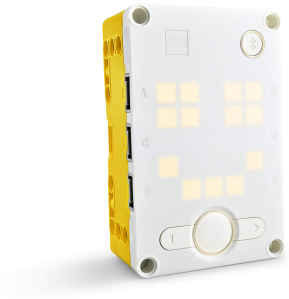

Abbildung 2.1: großer LEGO® Technic Hub

2.3 Sensorik

LEGO® Technic Farbsensor:

Dieser Sensor kann durch die Intensität des reflektierten Lichts, welche er durch den eingebauten Lichtring erzeugt, bis zu acht Farben erkennen und unterscheiden (erkennbare Farben: schwarz, blau, rot, weiß, braun, gelb, pink und grün). Er wird vor allem bei Anwendungen, wie Sortieren nach Farben, Linienverfolgung von farbigen Streifen und zum Ausführen von farbcodierten Befehlen eingesetzt. Um die Farberkennung optimal auf die entsprechende Umgebung anzupassen, wird das Umgebungslicht zuvor ausgewertet.

Abbildung 2.2: LEGO® Technic Farbsensor

2.4 Aktorik

2.4.1 LEGO® Technic Großer Winkelmotor

Für die Bewegungssteuerung wird dieser Winkelmotor mit hoher Drehkraft und präziser Steuerung verwendet. Dieser Motor ist dafür ausgelegt, um schwere und komplexe Konstruktionen exakt in Geschwindigkeit und Position verfahren zu können. Dies wird durch den internen Winkel- und Rotationssensor ermöglicht. Der Anwendungsbereich ist vielseitig, unter anderem wird dies als Antrieb von Rädern, Gelenke, Greifarme, Hebevorrichtungen und Drehscheibe angewandt. Vor allem für Aufgaben, welche eine genau und wiederholbare Aktion ausüben müssen.

Abbildung 2.3: LEGO® Technic Großer Winkelmotor

3 der Algorithmus

Struktur und Funktionalität des Python-Codes für ein 4-Gewinnt-Spiel

1. Initialisierung und Konfiguration

Zu Beginn werden grundlegende Variablen und Datenstrukturen definiert, die den Zustand des Spiels abbilden. Das Spielfeld wird als zweidimensionale Liste mit 6 Zeilen und 7 Spalten realisiert, wobei freie Felder mit dem Symbol " gekennzeichnet sind. Zusätzlich wird ein Dictionary zur Zuordnung der numerischen Spielwerte (0 für den Computer und 2 für den Menschen) zu deren Symbolen („O“ bzw. „X“) erstellt. Die Bewertungsmatrix `werte` dient zur heuristischen Gewichtung der Spielfeldpositionen, wobei zentrale Felder höher bewertet werden als Randpositionen, da sie strategisch vorteilhafter sind. Diese Initialisierung schafft die Grundlage für die spätere Spiellogik und die Bewertung von Spielsituationen.

2. Ausgabefunktion

Die Funktion `ausgabe()` visualisiert das aktuelle Spielfeld in der Konsole. Dabei werden numerische Spielzustände in die zugehörigen Zeichen („O“ oder „X“) umgewandelt und zeilenweise ausgegeben. Nicht belegte Felder erscheinen als " und am unteren Rand wird eine Spaltenbeschriftung (0 bis 6) zur Orientierung angezeigt. Die Funktion

dient sowohl der Information als auch der Interaktion mit dem menschlichen Spieler. Sie ermöglicht die Nachvollziehbarkeit der Spielzüge und den aktuellen Spielverlauf.

3. Bewertungsfunktion

Die Funktion `bewertung()` liefert einen numerischen Wert, der die aktuelle Spielsituation heuristisch einschätzt. Für jedes belegte Feld wird basierend auf der Bewertungsmatrix ein Wert addiert oder subtrahiert: positive Werte für den Computer, negative für den Menschen. Dadurch entsteht ein Maß für die Positionierungsvorteile des Computers gegenüber dem Gegner. Die Funktion kommt im Minimax-Algorithmus zum Einsatz, wenn die maximale Suchtiefe erreicht ist oder keine Gewinnsituation erkannt wurde. Sie ermöglicht eine differenzierte Einschätzung von Zwischenständen jenseits reiner Gewinnbedingungen.

4. Auswertungsfunktion

Die Funktion `auswertung()` überprüft, ob eine Spielsituation vorliegt, in der ein Spieler vier Spielsteine in einer Reihe platzieren konnte. Dabei werden horizontale, vertikale und diagonale Kombinationen systematisch untersucht. Ergibt sich eine solche Konstellation, so wird der entsprechende Spieler (0 für Computer, 2 für Mensch) als Gewinner zurückgegeben. Liegt keine Gewinnsituation vor, wird zudem überprüft, ob das Spielfeld voll ist, was auf ein Unentschieden hinweist (Rückgabewert 1). In allen anderen Fällen wird -1 zurückgegeben, was den Fortbestand des Spiels signalisiert.

5. Zugfunktionen

Die Funktion `zug(y, s)` simuliert das Einwerfen eines Spielsteins in die Spalte `y` für den Spieler `s`. Dabei wird die unterste freie Position in der Spalte gesucht und entsprechend belegt. Die komplementäre Funktion `zugRueckgaengig(y)` entfernt den

obersten Spielstein aus der Spalte und stellt somit den vorherigen Spielzustand wieder her. Diese Rücknahmefunktion ist essenziell für die Implementierung des Minimax-Algorithmus, da im Rahmen der rekursiven Bewertung zahlreiche hypothetische Züge getestet und wieder verworfen werden müssen. Beide Funktionen gewährleisten eine realitätsnahe Simulation von Spielzügen.

6. Minimax-Algorithmus

Die beiden Funktionen `maxFunktion(tiefe)` und `minFunktion(tiefe)` implementieren gemeinsam den klassischen Minimax-Algorithmus mit begrenzter Suchtiefe. `maxFunktion` wird vom Computer aufgerufen und versucht, den bestmöglichen (höchsten) Bewertungswert zu erzielen, während `minFunktion` die besten Gegenreaktionen des menschlichen Spielers simuliert. Beide Funktionen überprüfen vor jedem Zug, ob bereits ein Gewinn oder Unentschieden vorliegt, um die Rekursion ggf. zu beenden. Wenn die maximale Suchtiefe erreicht ist, erfolgt die Bewertung über die Heuristik. Diese Logik bildet das Kernstück der künstlichen Intelligenz im Spiel.

7. Zugauswahl des Computers

Die Funktion `besterZug()` bestimmt den optimalen Zug des Computers unter Anwendung des Minimax-Verfahrens mit einer festen Suchtiefe von vier. Für jede gültige Spalte wird simuliert, wie sich der Zug auf die Spielsituation auswirken würde, wobei anschließend der resultierende Spielzustand durch den Minimax-Algorithmus bewertet wird. Der Zug mit dem geringsten Wert aus Sicht des Gegners (also dem besten aus Computersicht) wird als nächster Spielzug ausgewählt. Diese Funktion stellt sicher, dass der Computer stets eine taktisch sinnvolle Entscheidung trifft. Sie verbindet die Simulation potenzieller Spielverläufe mit einer strategischen Bewertung.

8. Spielsteuerung

Die Funktion `spiele()` koordiniert den gesamten Spielablauf zwischen Mensch und Computer. Nach der Initialisierung des Spielfelds folgt eine Spielschleife, in der der Mensch und der Computer abwechselnd ihre Züge tätigen. Der Spieler gibt seine Eingabe über die Konsole ein, während der Computer seinen Zug automatisch über die `besterZug()`-Funktion berechnet. Nach jedem Zug wird der Spielzustand neu ausgegeben und auf ein mögliches Spielende überprüft. Die Spielsteuerung bildet somit die zentrale Benutzerschnittstelle und kontrolliert die Spiellogik.

9. Programmausführung

Der abschließende Block `if __name__ == "__main__"` sorgt dafür, dass das Spiel nur dann automatisch gestartet wird, wenn das Skript direkt ausgeführt wird. Dies ist in Python eine gängige Praxis, um eine Datei sowohl als importierbares Modul als auch als eigenständig ausführbares Programm nutzen zu können. Dadurch wird verhindert, dass das Spiel beim Import des Skripts in andere Module unbeabsichtigt gestartet wird. Dieser Mechanismus fördert die Wiederverwendbarkeit und Modularität des Codes. In wissenschaftlichen Anwendungen ist dies ein Merkmal guter Programmierpraxis.

4 Umsetzung und Ergebnisse

Je nach Art der Arbeit kann diese Kapitelüberschrift auch „Ergebnisse“ lauten, z. B. bei rein messtechnischen Aufgaben.

Beschreibung der Umsetzung des zuvor gewählten Vorgehens (theoretische Untersuchung, Erhebungen, Durchführung von Experimenten, Prototypenaufbau, Implementierung eines Prozesses, etc.).

Verifikation anhand der zuvor erarbeiteten Anforderungen und Validierung in Bezug auf das zuvor gestellte Ziel. Diskussion der Ergebnisse. Spätestens hier auch auf die Zuverlässigkeit der gewonnenen Erkenntnisse eingehen (z. B. anhand der Genauigkeit von Messergebnissen).

5 Zusammenfassung

Auf zwei bis drei Seiten soll auf folgende Punkte eingegangen werden:

- Welches Ziel sollte erreicht werden
- Welches Vorgehen wurde gewählt
- Was wurde erreicht, zentrale Ergebnisse nennen, am besten quantitative Angaben machen
- Konnten die Ergebnisse nach kritischer Bewertung zum Erreichen des Ziels oder zur Problemlösung beitragen
- Ausblick

In der Zusammenfassung sind unbedingt klare Aussagen zum Ergebnis der Arbeit zu nennen. Üblicherweise können Ergebnisse nicht nur qualitativ, sondern auch quantitativ benannt werden, z. B. „. . . konnte eine Effizienzsteigerung von 12 % erreicht werden.“ oder „. . . konnte die Prüfdauer um 2 h verkürzt werden“.

Die Ergebnisse in der Zusammenfassung sollten selbstverständlich einen Bezug zu den in der Einleitung aufgeführten Fragestellungen und Zielen haben.

Literaturverzeichnis

[Bel24] Charles A. Bell. *MicroPython for the Internet of Things: A Beginner's Guide to Programming with Python on Microcontrollers*. English. 2nd. Berkeley, CA: Apress, 2024. ISBN: 9781484298619.

[PLJ23] Ignas Plauska, Agnius Liutkevičius und Audronė Janavičiūtė. „Performance Evaluation of C/C++, MicroPython, Rust and TinyGo Programming Languages on ESP32 Microcontroller". In: *Electronics* 12.1 (2023). Open Access Article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license, S. 143. DOI: 10.3390/electronics12010143. URL: https://www.mdpi.com/2079-9292/12/1/143.

[SS23] Albin Samefors und Felix Sundman. *Investiating Energy Consumption and Responsiveness of low power modes in MicroPython for STM32WB55*. Bachelor's thesis. 15 hp (first-cycle education). Jönköping, Sweden, Juni 2023. URL: https://www.diva-portal.org/smash/get/diva2:1778426/FULLTEXT01.pdf.

Abbildungsverzeichnis

Tabellenverzeichnis

A Nutzung von Künstliche Intelligenz basierten Werkzeugen

Im Rahmen dieser Arbeit wurden Künstliche Intelligenz (KI) basierte Werkzeuge benutzt. Tabelle 1.1 gibt eine Übersicht über die verwendeten Werkzeuge und den jeweiligen Einsatzzweck.

Tabelle 1.1: Liste der verwendeten KI basierten Werkzeuge

Werkzeug	**Beschreibung der Nutzung**
ChatGPT	• Grundlagenrecherche zu bekannten Prinzipien optischer Sensorik zur Abstandsmessung (siehe Abschnitt ...) • Suche nach Herstellern von Lidar-Sensoren (siehe Abschnitt ...) • ...
ChatPDF	• Recherche und Zusammenfassung von wissenschaftlichen Studien im Themenfeld ... • ...
DeepL	• Übersetzung des Papers von [...]
Tabnine AI coding assistant	• Aktiviertes Plugin in MS Visual Studio zum Programmieren des ... • ...
...	• ...

B Ergänzungen

2.1 Details zu bestimmten theoretischen Grundlagen

2.2 Weitere Details, welche im Hauptteil den Lesefluss behindern

C Details zu Laboraufbauten und Messergebnissen

3.1 Versuchsanordnung

3.2 Liste der verwendeten Messgeräte

3.3 Übersicht der Messergebnisse

3.4 Schaltplan und Bild der Prototypenplatine

D Zusatzinformationen zu verwendeter Software

4.1 Struktogramm des Programmentwurfs

4.2 Wichtige Teile des Quellcodes

E Datenblätter

Auf den folgenden Seiten wird eine Möglichkeit gezeigt, wie aus einem anderen PDF-Dokument komplette Seiten übernommen werden können, z. B. zum Einbindungen von Datenblättern. Der Nachteil dieser Methode besteht darin, dass sämtliche Formateinstellungen (Kopfzeilen, Seitenzahlen, Ränder, etc.) auf diesen Seiten nicht angezeigt werden. Die Methode wird deshalb eher selten gewählt. Immerhin sorgt das Package *„pdfpages“* für eine korrekte Seitenzahleinstellung auf den im Anschluss folgenden „nativen“ LaTeX-Seiten.

Eine bessere Alternative ist, einzelne Seiten mit „*\includegraphics*“ einzubinden.

2

3

4