

Robotik

Leistungsnachweis 3

Dokumentation

Autonomes Parken mit EV3



Verfassende Personen:
Gruppennummer:
Klasse:
Studiengang:
Semester:
Institution:
Dozent TEKO:
Einreichdatum:

Salihi Sirwan
5
OTEL-22a
Dipl. Techniker/in HF Elektrotechnik
6
TEKO Schweizerische Fachschule Olten
Benjamin Engler
20.09.2025

Inhaltsverzeichnis

1.	Initialisierung	3
1.1.	Auftragsklärung	3
1.2.	Ziele.....	4
2.	Realisierung	5
2.1.	Informationssammlung / Recherche.....	5
2.2.	Aufbau / Komponenten.....	6
2.3.	Programmcode	10
2.4.	Test.....	15
3.	Abschluss.....	16
3.1.	Auswertung der Ziele.....	16
3.2.	Zeitliche Aufwendungen	16
3.3.	Reflexion.....	17
3.3.1.	Sachreflexion.....	17
3.3.2.	Leistungsreflexion	17
3.3.3.	Lernreflexion.....	17
4.	Fazit	17
5.	Quellenverzeichnis	18
5.1.	Literaturverzeichnis	18
5.2.	Abbildungen.....	18
5.3.	Tabelle.....	18
5.4.	Hilfsmitteln	18
6.	Eigenständigkeitserklärung	19
7.	Anhang	20

1. Initialisierung

1.1. Auftragsklärung

Im Rahmen des Fachs Robotik soll eine Projektarbeit durchgeführt werden, bei der das im Unterricht erlernte Wissen praxisnah angewendet wird. Das Ziel besteht darin, mit dem LEGO EV3 ein funktionsfähiges Fahrzeug aufzubauen und zu programmieren. Dieses Fahrzeug soll selbstständig bestimmte Aufgaben ausführen können, welche an ein reales Szenario angelehnt sind.

Die Aufgabenstellung orientiert sich an einem typischen Kundenauftrag, wie er auch in einer Firma vorkommen könnte. Dabei steht die Planung, der Aufbau, die Programmierung sowie das Testen und Dokumentieren der Arbeit im Vordergrund. Das Projekt verbindet somit die theoretischen Kenntnisse aus dem Unterricht mit einer praktischen Umsetzung.

Für die Realisierung wurde die Idee gewählt, ein Fahrzeug zu bauen, das selbstständig in eine Parkhalle fahren und dort einen freien Parkplatz erkennen kann. Mit Hilfe von Sensoren soll das Fahrzeug in der Lage sein, seine Umgebung zu analysieren. Dazu wird ein Ultraschallsensor eingesetzt, der auf einem Motor drehbar gelagert ist. Dieser kann sowohl nach links als auch nach rechts prüfen, ob ein Parkplatz frei ist. Ergänzt wird das System durch einen Farbsensor, der Markierungen am Boden erkennt. Diese Markierungen dienen dem Fahrzeug als Orientierung, an welcher Stelle es prüfen soll, ob sich daneben ein Parkplatz befindet oder einfach das es gerade ausfahren kann.

Das Fahrzeug soll also eine vorgegebene Strecke abfahren, an der Markierung anhalten und mithilfe seiner Sensoren eine Entscheidung treffen. Wird ein freier Parkplatz erkannt, so muss es korrekt einfahren und sich selbstständig positionieren. Damit entsteht eine Simulation, die das Prinzip eines autonomen Parkassistenten nachstellt.

Der Aufbau wird ausschliesslich mit den Bauteilen des EV3-Sets realisiert. Dadurch wird sichergestellt, dass das Projekt mit den zur Verfügung stehenden Mitteln umsetzbar ist und am Ende wieder vollständig zurückgegeben werden kann. Für die Programmierung wird Python eingesetzt, da diese Sprache im Unterricht verwendet wurde und eine flexible, saubere Umsetzung der Logik ermöglicht.

Die Projektarbeit ist zeitlich begrenzt und umfasst insgesamt fünf Unterrichtstage mit etwa 30 Lernstunden pro Person. Sie wird als Einzelarbeit durchgeführt. Zu den abzugebenden Ergebnissen gehören ein Pflichtenheft, eine technische Dokumentation mit Testprotokoll sowie eine Präsentation mit Live-Demonstration des Fahrzeugs.

Mit dieser Aufgabenstellung soll nicht nur ein funktionierendes Roboterfahrzeug entstehen, sondern auch das Verständnis für Sensorik, Programmierung und logische Abläufe vertieft werden. Das Projekt bietet die Möglichkeit, ein praxisnahes Problem zu bearbeiten und dabei den kompletten Prozess von der Planung über die Umsetzung bis zur Präsentation zu durchlaufen.

1.2. Ziele

Richtziel:

1. *Vollständige Dokumentation mit pünktlicher Abgabe bis 20.09.2025-23:59*
2. *Es liegt ein vollständig zusammengebautes EV3-Fahrzeug vor, bei dem alle Sensoren (Ultraschall, Farbsensor) korrekt angeschlossen sind, eingelesen werden können und zuverlässig funktionieren.*
3. *Der Ultraschallsensor ist so programmiert, dass er sich um 90° nach rechts und 90° nach links drehen kann und somit einen Drehwinkel von 180° abdeckt.*
4. *Der Farbsensor ist in der Lage, Farben zuverlässig zu unterscheiden und dient als Steuersignal für die Programmabläufe.*
5. *Das Fahrzeug kann eine vorgegebene Strecke bis zu einer Markierung selbstständig abfahren, dort anhalten und eine Entscheidung treffen.*
6. *An der erkannten Markierung prüft das Fahrzeug, ob auf der Seite ein Parkplatz frei ist.*
7. *Falls ein Parkplatz frei ist, positioniert sich das Fahrzeug korrekt und fährt in diesen ein.*
8. *Alle definierten Funktionen (Sensorerkennung, Spurfolgen, Parkvorgang) sind innerhalb des vorgegebenen Zeitrahmens programmiert und getestet.*
9. *Abgabe des Pflichtenheftes erfolgte termingerecht am 24.08.2025.*
10. *Abgabe von Dokumentation und Source Code erfolgte fristgerecht bis 20.09.2025, 23:59 Uhr.*

– **Dozent Benjamin Engler**

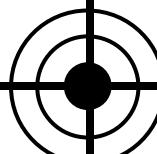
Endergebnisse

Sinn und Zweck

1. *Praxisnahe Umsetzung des Gelernten im Fach Robotik*
2. *Dokumentation des Projekts als Nachweis und Referenz*
3. *Vertieftes Verständnis für Programmierung in Python entwickeln*

Kunde

Erfolgskriterien



1. *Funktionsfähigkeit: Das EV3-Fahrzeug ist vollständig aufgebaut, alle Sensoren sind angeschlossen, eingelesen und funktionieren zuverlässig.*
2. *Ultraschall-Drehung: Der Ultraschallsensor deckt einen Drehwinkel von 180° ab (Abweichung ≤ 10 %).*
3. *Farbsensor-Steuerung: Farben werden korrekt erkannt und zur Programmsteuerung eingesetzt.*
4. *Spurfolgen: Das Fahrzeug folgt einer schwarzen/weißen Linie zuverlässig bis zur Markierung und hält dort an.*
5. *Entscheidungslogik: An der Markierung trifft das Fahrzeug eine programmierte Entscheidung (z. B. Parkplatz prüfen).*
6. *Parkvorgang: Bei freiem Parkplatz positioniert sich das Fahrzeug korrekt und fährt selbstständig ein.*
7. *Dokumentation: Pflichtenheft, Source Code und Projektdokumentation sind termingerecht abgegeben, nachvollziehbar und vollständig.*
8. *Präsentation: Ergebnisse wurden im Unterricht am 23.09.2025 präsentiert; Funktionsfähigkeit konnte demonstriert werden.*
9. *Zuverlässigkeit: Alle definierten Muss-Kriterien (ME) sind erfüllt; Kann-Kriterien (KE) wurden soweit möglich umgesetzt.*

Legende:

ME: Muss erfüllt werden
KE: Kann erfüllt werden

2. Realisierung

2.1. Informationssammlung / Recherche

Die Ideenfindung für dieses Projekt entstand eher zufällig. Beim Anschauen von Videos in sozialen Medien bin ich auf Beispiele von Tesla-Fahrzeugen gestossen, die autonom fahren und als Taxis eingesetzt werden. Viele dieser Fahrzeuge sind in der Lage, selbstständig zu parkieren. Dabei nutzen sie ähnliche Prinzipien wie in diesem Projekt. Das Fahrzeug fährt eine Strecke ab, überprüft mit seinen Sensoren, ob links, rechts oder vorne ein Parkplatz frei ist, und manövriert sich anschliessend präzise in die Parkfläche.

Natürlich ist der Vergleich zum realen Leben nur bedingt möglich. In der Praxis sind wesentlich mehr Faktoren zu berücksichtigen, etwa die Verkehrssituation, andere Fahrzeuge, Fußgänger oder unvorhersehbare Hindernisse. Diese erfordern eine grosse Anzahl zusätzlicher Sensoren, hochentwickelte Software und eine viel komplexere Logik. Das Niveau solcher Systeme liegt daher weit über dem eines EV3-Schulprojekts. Dennoch ist die Grundidee vergleichbar und bietet eine gute Basis, um das Prinzip in vereinfachter Form nachzustellen.

Für die Umsetzung habe ich zunächst das EV3-Set genau studiert und die passenden Bauteile ausgewählt. Die Grundkonstruktion des Fahrzeugs basiert auf den Vorgaben und Erklärungen von LEGO Education. Auf der offiziellen EV3-Website finden sich zahlreiche Beispiele, Anleitungen, technische Beschreibungen und Datenblätter, die sehr ausführlich erklären, wie die einzelnen Komponenten funktionieren. Diese Unterlagen waren eine wertvolle Grundlage, um zu verstehen, welche Möglichkeiten das Set bietet und wie die Bauteile am besten kombiniert werden können.

Nach der Auswahl der Hauptkomponenten begann die praktische Arbeit. Dabei habe ich Schritt für Schritt ausprobiert, wie das Fahrzeug aufgebaut werden kann. Der Farbsensor wurde an der Vorderseite angebracht, um Markierungen am Boden zuverlässig zu erkennen. Der Ultraschallsensor wurde auf einem Motor befestigt, sodass er sich drehen und verschiedene Bereiche seitlich absuchen kann. Durch wiederholtes Testen, Anpassen und Ausprobieren wurde die Konstruktion kontinuierlich verbessert, bis sie stabil und funktional war.

Zusätzliche Ideen und Hinweise erhielt ich von unserem Dozenten, Herrn Engler, der uns bei Fragen unterstützte. Darüber hinaus halfen Recherchen im Internet sowie moderne Hilfsmittel wie KI-gestützte Plattformen, Lösungsansätze zu entwickeln und Schwierigkeiten zu überwinden. Auf diese Weise konnte ich nicht nur auf bestehendes Wissen zurückgreifen, sondern auch neue Ansätze entdecken und eigenständig weiterentwickeln.

2.2. Aufbau / Komponenten

Der Aufbau besteht aus mehreren Sensoren und Akteuren.

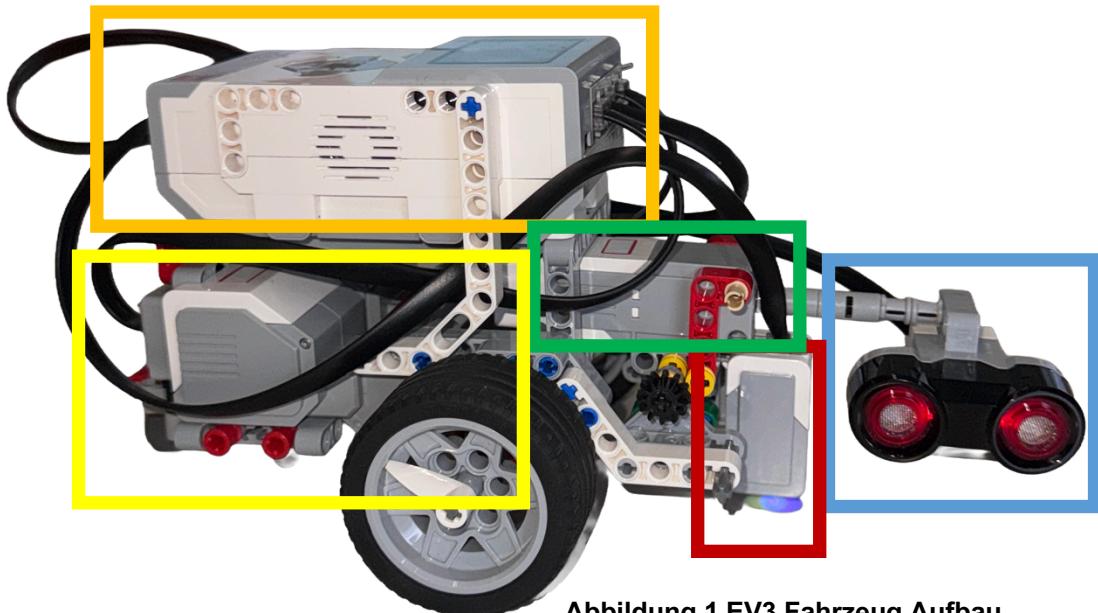
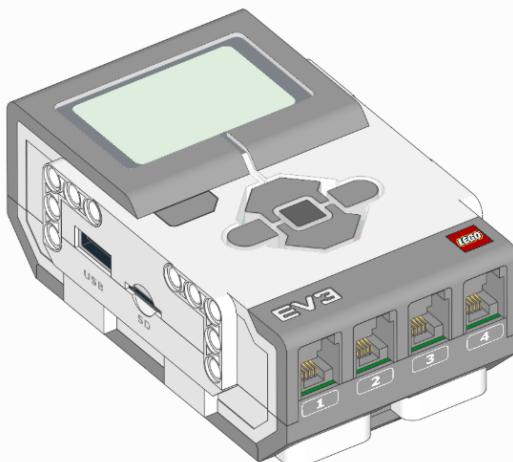


Abbildung 1 EV3 Fahrzeug Aufbau

Was	Wie heisst es	Wozu	Farbe
uC	EV3	Dient dazu, die Programmierung auszuführen und die Akteure anhand der Sensordaten zu steuern. Er ist das „Gehirn“ des gesamten Systems.	Yellow
Aktor	2x Starke Motor auf beiden Achsen	2x Grosser Motoren auf beiden Achsen Sorgen für den Antrieb der Räder auf beiden Seiten und ermöglichen die Fortbewegung des Fahrzeugs.	Yellow
Aktor	Mittlerer Motor	Dieser Motor bewegt den Ultraschallsensor um insgesamt 180° – also 90° nach links und 90° nach rechts –, damit die Umgebung gezielt abgesucht werden kann.	Green
Sensor	Farbsensor	Dient zur Erkennung von Farben, damit der EV3 weiß, wann das Fahrzeug anhalten muss und welcher Linie es folgen soll.	Red
Sensor	Ultraschallsensor	Misst Abstände und prüft, ob ein Parkplatz frei oder belegt ist.	Blue

EV3-Stein:**Abbildung 2 EV3**

Der EV3-Stein bildet das zentrale Steuerungselement des Roboters. Er verfügt über ein farbiges Display, mehrere Knöpfe zur Bedienung sowie eine kleine Status-LED, die den aktuellen Betriebszustand anzeigt. Ergänzt wird dies durch einen eingebauten Lautsprecher, über den Signale oder Töne ausgegeben werden können.

Für die Verbindung mit der Außenwelt sind verschiedene Anschlüsse vorhanden wie man im Abbildung 2 erkennt. Auf der Oberseite befinden sich die Ports S1 bis S4, an die unterschiedlichen Sensoren angeschlossen werden können, während auf der Unterseite die Ports A bis D für Motoren vorgesehen sind. Darüber hinaus bietet der Stein Schnittstellen für USB, SD-Karte und Netzteil sowie Möglichkeiten für drahtlose Verbindungen über WLAN oder Bluetooth.

In seiner Funktion übernimmt der EV3-Stein die Rolle des „Gehirns“ des Roboters. Er wertet die eingehenden Signale der Sensoren aus, verarbeitet die programmierten Abläufe und steuert die Motoren entsprechend. Auf diese Weise koordiniert er sämtliche Aktionen des Fahrzeugs und stellt sicher, dass Bewegungen und Reaktionen präzise ausgeführt werden.

Eine besondere Eigenschaft ist, dass der EV3-Stein sowohl direkt über das Display bedient und Programme gestartet werden können als auch die Möglichkeit besteht, neue Programme über einen PC mit der EV3-Software oder in Python aufzuspielen und auszuführen. Dadurch ist er flexibel einsetzbar und sowohl für einfache Tests als auch für komplexere Anwendungen geeignet.

Mittlere Motor

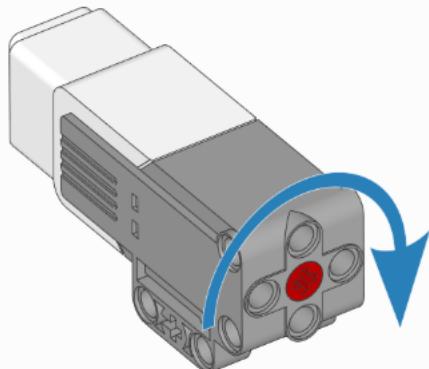


Abbildung 3 Mittlere Motor

Der mittlere Motor des LEGO MINDSTORMS EV3-Systems ist kleiner, leichter und schneller als der grosse Motor, wie in Abbildung 3 erkennbar ist. Durch seinen integrierten Drehsensor mit einer Genauigkeit von 1° eignet er sich besonders für Aufgaben, die eine präzise Steuerung erfordern. Eine volle Umdrehung entspricht 360° , wodurch Bewegungen exakt kontrolliert und überwacht werden können. In diesem Projekt wird der Motor als Servo eingesetzt, um den Ultraschallsensor um 180° nach links und rechts zu drehen. Dadurch kann das Fahrzeug prüfen, ob ein Parkplatz frei ist. Dank seiner schnellen Reaktionsfähigkeit eignet er sich ideal für solche Anwendungen, bei denen es auf Genauigkeit und Beweglichkeit ankommt. Ausgestaltung / Umsetzung

Starke Motor

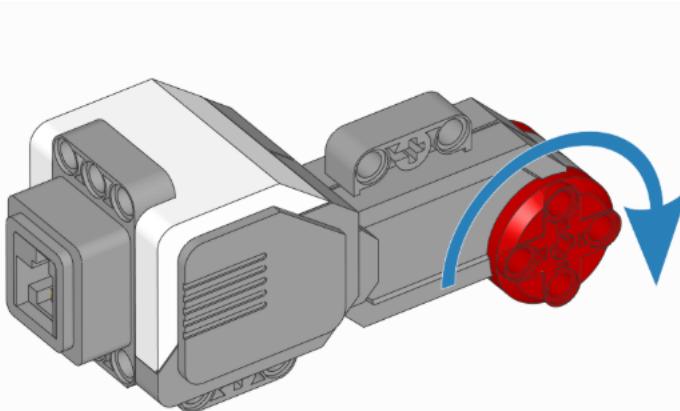


Abbildung 4 Starke/Grosse Motor

Der grosse Motor des LEGO MINDSTORMS EV3-Systems im Abbildung 4 wird an einem der Ports A–D am EV3-Stein angeschlossen, über die er sowohl mit Strom als auch mit Steuersignalen versorgt wird. Es handelt sich um einen Servo-Motor mit integriertem Drehwinkelsensor, wodurch er nicht nur mit einer bestimmten Geschwindigkeit betrieben werden kann, sondern auch sehr präzise auf einen gewünschten Winkel oder eine exakte Position gesteuert wird. Er erlaubt die Kontrolle von Geschwindigkeit, Drehrichtung und Drehwinkel, liefert über seinen eingebauten Tacho-Sensor ein präzises Feedback und bietet eine Bremsfunktion, mit der Bewegungen abrupt oder sanft gestoppt werden können. In diesem Projekt werden die grossen Motoren für den Antrieb der Räder des Fahrzeugs eingesetzt und sorgen damit für die zuverlässige Fortbewegung des Roboters. Dank ihrer Robustheit und Genauigkeit eignen sie sich besonders für wiederholgenaue Bewegungsabläufe.

Ultraschallsensor

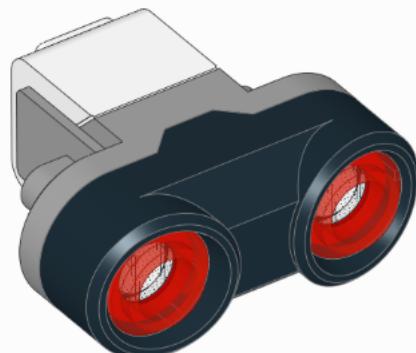


Abbildung 5 Ultraschallsensor

Der Ultraschallsensor des LEGO MINDSTORMS EV3-Systems im Abbildung 5 wird an einem der Eingangs-Ports S1–S4 des EV3-Steins angeschlossen und misst Entfernung zu Objekten mithilfe von Schallwellen. Er sendet dazu Ultraschallimpulse aus und berechnet anhand der zurückgeworfenen Echos die Distanz. Die Messwerte können in Zentimetern oder Zoll ausgegeben werden, wobei der Sensor Distanzen bis zu etwa 250 cm erfassen und mit einer Genauigkeit von wenigen Millimetern unterscheiden kann.

Der Sensor eignet sich besonders gut für Anwendungen, bei denen Hindernisse erkannt oder Abstände präzise gemessen werden müssen. Er ermöglicht sowohl eine kontinuierliche Abstandsmessung als auch die Abfrage, ob sich ein Objekt innerhalb eines bestimmten Bereichs befindet.

In diesem Projekt wird der Ultraschallsensor eingesetzt, um die Umgebung nach freien Parkplätzen abzusuchen. Mithilfe eines zusätzlichen Motors kann er nach links und rechts gedreht werden und überprüft so einen Bereich von 180°. Erkennt er dabei genügend Platz, signalisiert er dem Fahrzeug, dass es einparken kann. Damit übernimmt der Sensor eine zentrale Rolle bei der Umsetzung des autonomen Parkvorgangs.

Farbsensor

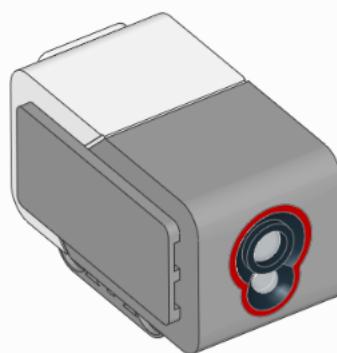


Abbildung 6 Farbsensor

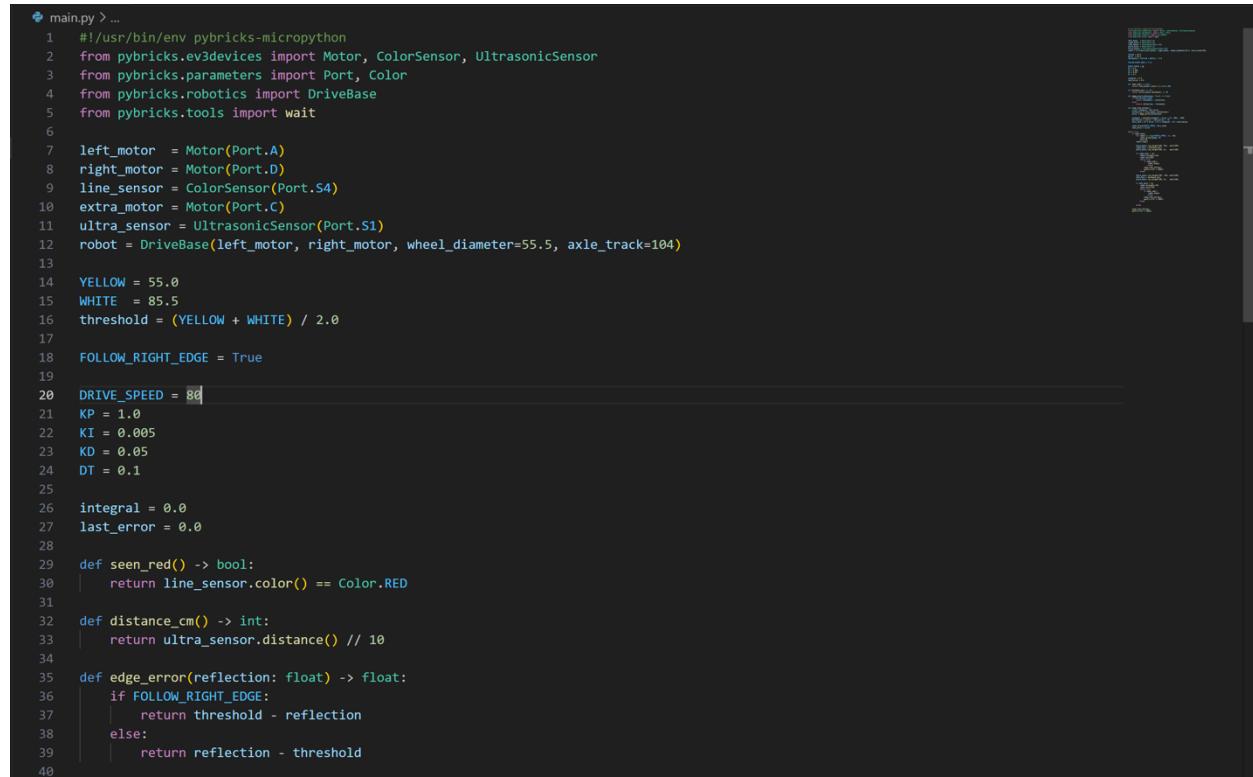
Der Farbsensor des LEGO MINDSTORMS EV3-Systems wird an einem der Eingangs-Ports S1–S4 des EV3-Steins angeschlossen und kann sowohl Farben erkennen als auch die Helligkeit (Reflexionswert) messen. Er arbeitet mit einem integrierten Lichtsender und einem Empfänger, die gemeinsam bestimmen, wie viel Licht von der Oberfläche zurückgeworfen wird. Dadurch ist es möglich, verschiedene Farben wie Rot, Blau, Grün, Weiss

oder Schwarz zuverlässig zu unterscheiden. Zusätzlich kann der Sensor auch den Umgebungslichtwert messen, was ihn vielseitig einsetzbar macht.

Der Farbsensor eignet sich besonders für Aufgaben, bei denen Linien erkannt oder Farbsignale als Steuerungselemente verwendet werden sollen. Typische Anwendungen sind Linienfolger, Stoppmarkierungen oder das Auslösen von Aktionen bei bestimmten Farben.

In diesem Projekt wird der Farbsensor eingesetzt, um die Markierungen am Boden zu erkennen. Diese Markierungen geben dem Fahrzeug die Information, wann es anhalten und den Ultraschallsensor aktivieren soll, um nach Parkplätzen zu suchen. Zudem dient er zur Orientierung auf der Strecke, indem er zwischen den Farben Weiss und Gelb unterscheidet. Damit spielt der Farbsensor eine entscheidende Rolle bei der Navigation und Steuerung des autonomen Parkvorgangs.

2.3. Programmcode



```

❶ main.py > ...
1  #!/usr/bin/env pybricks-micropython
2  from pybricks.ev3devices import Motor, ColorSensor, UltrasonicSensor
3  from pybricks.parameters import Port, Color
4  from pybricks.robotics import DriveBase
5  from pybricks.tools import wait
6
7  left_motor = Motor(Port.A)
8  right_motor = Motor(Port.D)
9  line_sensor = ColorSensor(Port.S4)
10 extra_motor = Motor(Port.C)
11 ultra_sensor = UltrasonicSensor(Port.S1)
12 robot = DriveBase(left_motor, right_motor, wheel_diameter=55.5, axle_track=104)
13
14 YELLOW = 55.0
15 WHITE = 85.5
16 threshold = (YELLOW + WHITE) / 2.0
17
18 FOLLOW_RIGHT_EDGE = True
19
20 DRIVE_SPEED = 80
21 KP = 1.0
22 KI = 0.005
23 KD = 0.05
24 DT = 0.1
25
26 integral = 0.0
27 last_error = 0.0
28
29 def seen_red() -> bool:
30     return line_sensor.color() == Color.RED
31
32 def distance_cm() -> int:
33     return ultra_sensor.distance() // 10
34
35 def edge_error(reflection: float) -> float:
36     if FOLLOW_RIGHT_EDGE:
37         return threshold - reflection
38     else:
39         return reflection - threshold

```

Abbildung 7 Programmcode Teil 1

Wie man in Abbildung 7 sieht, werden zu Beginn des Programms werden die notwendigen Bibliotheken eingebunden. Diese stellen die Bausteine für die Motoren, Sensoren und Steuerbefehle zur Verfügung. So können später Fahrbewegungen, Farberkennung, Abstandsmessungen und Wartezeiten umgesetzt werden.

Anschliessend folgt die Definition der Hardware. Hier wird genau festgelegt, welches Bauteil an welchem Anschluss des EV3-Steins verbunden ist. Der linke Fahrmotor befindet sich an Port A, der rechte Fahrmotor an Port D. Für die Linienerkennung wird ein Farbsensor am Port S4 verwendet. Zusätzlich ist am Port C ein Motor angeschlossen, der den Ultraschallsensor bewegt. Der Ultraschallsensor selbst ist an Port S1 verbunden. Damit die beiden Antriebsmotoren koordiniert gesteuert werden können, wird ein

DriveBase-Objekt erzeugt, in dem zusätzlich Raddurchmesser und Achsabstand einge tragen sind.

Im nächsten Abschnitt werden die Farbwerte für die Linienerkennung definiert. Dazu werden die gemessenen Reflexionswerte für Gelb und Weiss gespeichert. Aus diesen beiden Werten wird eine Schwelle berechnet, die später dabei hilft, zu unterscheiden, ob sich der Roboter mehr auf Gelb oder auf Weiss befindet. Diese Schwelle ist die Grundlage für die Steuerung des Linienfolgers.

Darauf folgt die Festlegung von Grundeinstellungen. Mit der Variablen „FOLLOW_RIGHT_EDGE“ wird entschieden, ob sich der Roboter am rechten Rand orientieren soll. In diesem Projekt ist dies auf „True“ gesetzt. Zusätzlich wird die Fahrgeschwindigkeit auf 80 festgelegt. Danach werden die Parameter für den PID-Regler bestimmt. Dieser Regler ist notwendig, damit der Roboter die Linie nicht nur grob, sondern möglichst präzise und stabil verfolgen kann. Dazu werden die Werte für den proportionalen, integralen und differenziellen Anteil (KP, KI, KD) sowie das Zeitintervall DT angegeben.

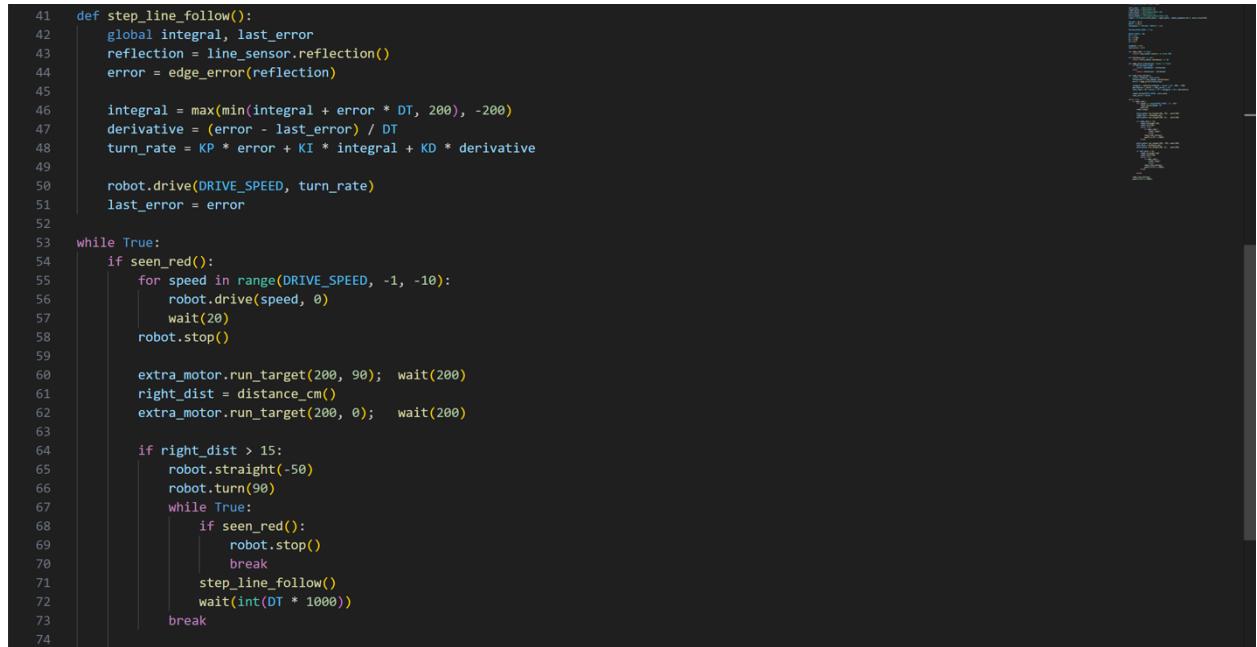
Im Anschluss werden noch zwei Variablen vorbereitet, die für die Berechnungen des Reglers benötigt werden: das Integral und der letzte Fehlerwert. Beide sind zu Beginn auf null gesetzt, da noch keine Berechnungen durchgeführt wurden.

Schliesslich folgen kleine Hilfsfunktionen, die das Programm übersichtlicher machen. Mit der Funktion „seen_red“ kann der Roboter prüfen, ob der Farbsensor die Farbe Rot erkannt hat. Mit „distance_cm“ wird die Distanz gemessen, die der Ultraschallsensor erfasst, und das Ergebnis in Zentimetern zurückgegeben. Die dritte Funktion „edge_error“ berechnet, wie weit der Roboter von der gewünschten Fahrspur abweicht. Je nachdem, ob der rechte oder linke Rand verfolgt wird, ergibt sich eine andere Berechnung. Damit ist der Grundstein des Programms gelegt: Die Sensoren und Motoren sind korrekt zugeordnet, die Farben sind kalibriert, die Fahrgeschwindigkeit und Reglerparameter

stehen fest, und es gibt Hilfsfunktionen, mit denen der Roboter Farben erkennen, Abstände messen und seine Spurabweichung berechnen kann.

Zusammengefasst:

- Import der Bibliotheken (Zeilen 1–5)
- Definition der Hardware und DriveBase (Zeilen 7–13)
- Farbwerte und Schwelle (Zeilen 15–17)
- Grundeinstellungen inkl. PID-Werte (Zeilen 19–24)
- Initialisierung von Regler-Variablen (Zeilen 26–27)
- Die drei kleinen Hilfsfunktionen (seen_red, distance_cm, edge_error) (Zeilen 29–39)



```

41 def step_line_follow():
42     global integral, last_error
43     reflection = line_sensor.reflection()
44     error = edge_error(reflection)
45
46     integral = max(min(integral + error * DT, 200), -200)
47     derivative = (error - last_error) / DT
48     turn_rate = KP * error + KI * integral + KD * derivative
49
50     robot.drive(DRIVE_SPEED, turn_rate)
51     last_error = error
52
53 while True:
54     if seen_red():
55         for speed in range(DRIVE_SPEED, -1, -10):
56             robot.drive(speed, 0)
57             wait(20)
58         robot.stop()
59
60         extra_motor.run_target(200, 90); wait(200)
61         right_dist = distance_cm()
62         extra_motor.run_target(200, 0); wait(200)
63
64     if right_dist > 15:
65         robot.straight(-50)
66         robot.turn(90)
67         while True:
68             if seen_red():
69                 robot.stop()
70                 break
71             step_line_follow()
72             wait(int(DT * 1000))
73         break
74

```

Abbildung 8 Programmcode Teil 2

Im nächsten Teil des Programms wird die Funktion `step_line_follow()` definiert. Diese Funktion ist der eigentliche Linienfolger, der dafür sorgt, dass das Fahrzeug immer der Kante zwischen Gelb und Weiss folgt. Zuerst wird der aktuelle Reflexionswert des Farbsensors gemessen. Dieser Wert wird mit dem zuvor berechneten Schwellenwert verglichen, wodurch sich ein Fehler ergibt: Je nachdem, ob der Roboter zu weit links oder zu weit rechts von der Spur ist, wird der Fehler positiv oder negativ.

Anhand dieses Fehlers berechnet der PID-Regler die Korrektur. Der proportionale Anteil reagiert direkt auf die Abweichung, der integrale Anteil berücksichtigt die Fehler über die Zeit und verhindert ein dauerhaftes Abdriften, während der differenzielle Anteil schnelle Änderungen ausgleicht. Am Ende ergibt sich daraus ein Steuerwert, der bestimmt, wie stark der Roboter nach links oder rechts lenken soll. Dieser Wert wird zusammen mit der Fahrgeschwindigkeit an die Funktion `robot.drive()` übergeben. So fährt der Roboter gleichzeitig vorwärts und korrigiert ständig seine Richtung, um der Linie zu folgen. Der zuletzt gemessene Fehler wird gespeichert, damit er beim nächsten Rechenschritt berücksichtigt werden kann.

Im Anschluss daran beginnt mit `while True:` das Hauptprogramm. Dieses läuft in einer Endlosschleife, solange das Fahrzeug eingeschaltet ist. Als erstes prüft der Roboter mit der Funktion `seen_red()`, ob er eine rote Markierung erkannt hat. Falls das der Fall ist, reduziert er seine Geschwindigkeit Schritt für Schritt, bis er vollständig zum Stehen kommt. Dieses Abbremsen geschieht sanft, indem die Geschwindigkeit in kleinen Schritten nach unten geregelt wird.

Sobald das Fahrzeug steht, wird der Kopf-Motor aktiviert, der den Ultraschallsensor bewegt. Zunächst schaut der Sensor nach rechts, und der Abstand wird gemessen. Danach wird der Sensor wieder in die Ausgangsposition zurückgedreht. Mit dem gemessenen Abstand kann entschieden werden, ob auf der rechten Seite ein Parkplatz frei ist.

Wenn rechts genügend Platz vorhanden ist, fährt der Roboter ein kleines Stück zurück und dreht sich nach rechts. Danach setzt er seine Fahrt fort, wobei er wieder dem

Linienfolger folgt. In dieser Phase prüft er laufend, ob erneut eine rote Markierung erscheint. Wird wieder Rot erkannt, hält er an und die Entscheidungsschleife beginnt von vorne. Damit ist sichergestellt, dass das Fahrzeug nur an den vorgesehenen Stellen seine Entscheidung trifft.

Falls jedoch rechts kein Parkplatz frei ist, wird an dieser Stelle das Programm mit einem break unterbrochen, sodass im nächsten Schritt die linke Seite geprüft werden kann.

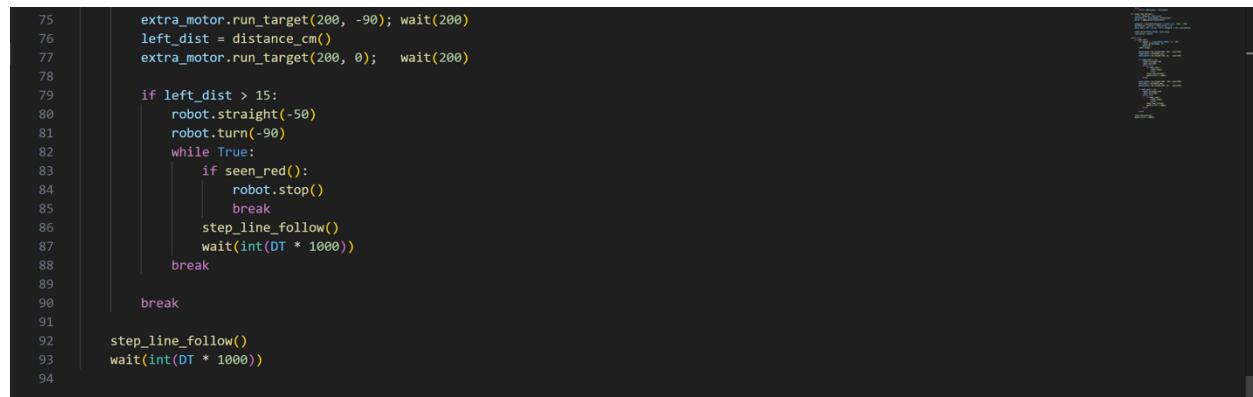
Zusammengefasst:

Definition der Linienfolger-Funktion (Zeilen 41–51)

- In der Funktion step_line_follow() wird gemessen, wie stark der Roboter von der Linie abweicht.
- Mit Hilfe des PID-Reglers (KP, KI, KD) wird eine Korrektur berechnet.
- Diese Korrektur bestimmt den Lenkwinkel, sodass der Roboter sauber der Linie folgt.
- Am Ende fährt der Roboter mit der vorgegebenen Geschwindigkeit weiter, während er sich ständig selbst korrigiert.

Start des Hauptprogramms (Zeilen 53–74)

- Mit while True: beginnt die Endlosschleife, in der der Roboter dauerhaft seine Aufgaben ausführt.
- Zuerst prüft der Farbsensor mit seen_red(), ob eine rote Markierung erkannt wurde.
- Falls ja, bremst der Roboter langsam ab und stoppt vollständig.
- Danach wird der Kopf-Motor bewegt, damit der Ultraschallsensor nach rechts schaut. Der Abstand wird gemessen und der Sensor kehrt in die Ausgangsposition zurück.
- Wenn rechts ein freier Platz erkannt wird (Abstand grösser als 15 cm), fährt der Roboter leicht zurück, dreht sich nach rechts und folgt der Linie erneut.
- Während dieser Fahrt prüft er fortlaufend, ob wieder eine rote Markierung erscheint, um gegebenenfalls erneut anzuhalten.
- Wenn rechts kein Platz frei ist, bricht der Ablauf hier ab, damit anschliessend die linke Seite geprüft werden kann.



```

75     extra_motor.run_target(200, -90); wait(200)
76     left_dist = distance_cm()
77     extra_motor.run_target(200, 0);   wait(200)
78
79     if left_dist > 15:
80         robot.straight(-50)
81         robot.turn(-90)
82         while True:
83             if seen_red():
84                 robot.stop()
85                 break
86             step_line_follow()
87             wait(int(DT * 1000))
88         break
89
90     break
91
92     step_line_follow()
93     wait(int(DT * 1000))
94

```

Abbildung 9 Programmcode Teil 3

Nachdem die rechte Seite geprüft wurde, folgt im Programm die Untersuchung der linken Seite wie in Abbildung 9 ersichtlich ist. Dazu wird der Kopf-Motor so gesteuert, dass sich der Ultraschallsensor um 90 Grad nach links dreht. In dieser Position wird der Abstand gemessen und als Wert gespeichert. Danach fährt der Motor den Sensor wieder in seine Ausgangsstellung zurück.

Liegt auf der linken Seite genügend Platz vor, also wenn der gemessene Abstand grösser als 15 Zentimeter ist, führt der Roboter ein Einparkmanöver nach links durch. Dazu fährt er zunächst ein kurzes Stück zurück und dreht sich anschliessend um 90 Grad nach links. In der darauffolgenden Schleife fährt er wieder mit dem Linienfolger, bis eine rote Markierung erkannt wird. Sobald der Farbsensor Rot sieht, stoppt der Roboter, beendet die Schleife und bleibt in der neuen Position stehen.

Falls jedoch auch auf der linken Seite kein Parkplatz frei ist, wird der Ablauf beendet. In diesem Fall gibt es keinen verfügbaren Platz und der Roboter bleibt nach dem Abbruch einfach stehen.

Wenn dagegen überhaupt keine rote Markierung erkannt wurde, läuft der normale Fahrbetrieb weiter. Dabei wird die Funktion `step_line_follow()` regelmässig aufgerufen, sodass der Roboter der Linie folgt. Zwischen den Aufrufen gibt es eine kurze Pause entsprechend der festgelegten Taktzeit. So bewegt sich das Fahrzeug konstant entlang der Fahrspur, bis es irgendwann erneut auf eine rote Markierung trifft und wieder in den Entscheidungsmodus übergeht.

Zusammengefasst:

Prüfung auf linken Parkplatz (Zeilen 75–77)

- Der Kopf-Motor dreht den Ultraschallsensor diesmal nach links.
- Der Abstand zur linken Seite wird gemessen.
- Danach wird der Sensor wieder in die Ausgangsposition zurückgedreht.

Einfahren nach links (Zeilen 79–87)

- Wenn links genügend Platz vorhanden ist (mehr als 15 cm Abstand), führt das Fahrzeug ein Einparkmanöver aus.
- Es fährt ein kleines Stück zurück und dreht sich um 90° nach links.
- Danach folgt es wieder der Linie, bis eine rote Markierung erkannt wird.
- Bei Rot stoppt es und verlässt die Schleife, womit der Parkvorgang abgeschlossen ist.

Abbruch bei keinem Parkplatz (Zeilen 88–90)

- Wenn weder rechts noch links ein Parkplatz gefunden wird, verlässt das Programm die Schleifen.
- Damit bleibt das Fahrzeug stehen, da keine Parkmöglichkeit vorhanden ist.

Normaler Fahrbetrieb (Zeilen 92–93)

- Falls keine rote Markierung erkannt wird, fährt der Roboter ganz normal weiter.
- Er ruft dafür immer wieder die Funktion `step_line_follow()` auf und wartet kurz entsprechend der vorgegebenen Taktzeit.
- Dadurch fährt er kontinuierlich am Rand entlang, bis er erneut eine rote Markierung entdeckt.

2.4. Test

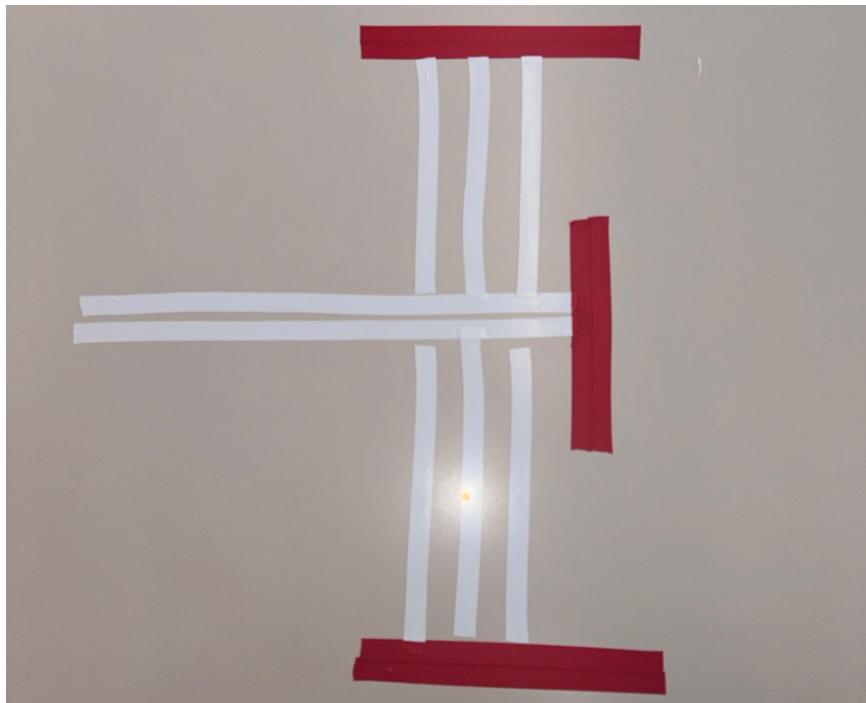


Abbildung 10 Fahrbahn

Das gesamte Projekt wurde erfolgreich wie auf der Abbildung 10 ersichtlichen Fahrbahn getestet und die Funktionen arbeiten zuverlässig. Das Fahrzeug folgt der gelben Linie, die durch weisse Ränder begrenzt ist, bis es eine rote Markierung erkennt. An dieser Stelle hält es an und dreht den Ultraschallsensor nach rechts, um zu prüfen, ob ein Parkplatz frei ist. Wird kein Hindernis erkannt, fährt das Fahrzeug ein Stück zurück, dreht nach rechts ein und folgt der gelben Linie bis zur nächsten roten Markierung. Dort hält es erneut an und beendet das Parkmanöver, indem es korrekt in der vorgesehenen Parkposition steht.

Ist der rechte Parkplatz besetzt, wiederholt sich derselbe Ablauf auf der linken Seite. Das Fahrzeug prüft die Distanz, und wenn genügend Platz vorhanden ist, führt es das Einparkmanöver spiegelverkehrt durch. Damit ist sichergestellt, dass das Fahrzeug sowohl auf der linken als auch auf der rechten Seite korrekt reagieren kann.

Schlussfolgerung:

Eine mögliche Erweiterung wäre, dass das Fahrzeug nicht dauerhaft stehen bleibt, wenn beide Parkplätze belegt sind. Stattdessen könnte es nach einer definierten Wartezeit, beispielsweise einer Minute, den Suchprozess wiederholen oder weiterfahren, bis eine nächste Station erreicht wird, um dort erneut eine Parkplatzsuche durchzuführen.

3. Abschluss

3.1. Auswertung der Ziele

Ziel-Nr.	Beschreibung	Wertung (ME / KE)	ist
1	Ein vollständig zusammengebautes EV3-Fahrzeug ist vorhanden, bei dem alle Sensoren korrekt angeschlossen sind, eingelesen werden können und zuverlässig funktionieren.	ME	Erfüllt
2	Der Ultraschallsensor kann sich um 90° nach rechts und 90° nach links drehen, sodass insgesamt ein Drehwinkel von 180° abgedeckt wird.	ME	Erfüllt
3	Der Ultraschallsensor wird nur dann aktiviert, wenn der Farbsensor eine Markierung erkennt. Die Ausrichtung erfolgt bei +90° oder -90°.	KE	Erfüllt
4	Der Farbsensor ist in der Lage, verschiedene Farben zuverlässig zu unterscheiden. Diese Farberkennung dient als Steuerungssignal für die Programmabläufe.	ME	Erfüllt
5	Das Fahrzeug kann die Strecke bis zu einer Markierung selbstständig abfahren, dort anhalten und eine Entscheidung treffen.	ME	Erfüllt
6	Farbsensor erkennt die Farben Weiss und Schwarz, um der Strecke folgen und anhalten zu können.	ME	Erfüllt
7	An der erkannten Markierung prüft das Fahrzeug, ob auf der Seite ein Parkplatz frei ist.	ME	Erfüllt
8	Falls ein Parkplatz frei ist, soll das Fahrzeug in diesen einfahren und sich korrekt positionieren.	KE	Erfüllt
9	Abgabe Pflichtenheft: 24.08.2025	ME	Erfüllt
10	Abgabe Dokumentation und Source Code. 20.09.2025 - 23:59	ME	Erfüllt
11	Präsentieren im Unterricht 23.09.2025	ME	-

Tabelle 1 SOLL und IST

Ziel 6 nicht die Farbe Schwarz, sondern die verfügbaren Farben verwendet werden. Zudem wurde die Anforderung von zwei auf drei Farben erweitert. Das Fahrzeug erkennt nun Weiss, Gelb und Rot.

3.2. Zeitliche Aufwendungen

Arbeit	Stunden
Pflichtenheft	4
LEGO Aufbau Projekt	4
Programmcode	5
Test	1-2
Korrekturen	1
Dokumentation	6

Tabelle 2 Aufwand

3.3. Reflexion

3.3.1. Sachreflexion

Bei der Arbeit mit dem EV3 hat mich vor allem die richtige Zusammenstellung der Bauteile gefordert. Es musste darauf geachtet werden, dass sich die Komponenten nicht gegenseitig behindern und die Sensoren zuverlässig arbeiten. Ein Beispiel dafür ist der Ultraschallsensor, den ich mit etwas Abstand eingebaut habe, damit andere Bauteile die Messungen nicht verfälschen. Diese Herausforderung konnte ich erfolgreich lösen und der Aufbau funktionierte danach einwandfrei.

Etwas schwieriger war die Arbeit mit dem Farbsensor. Anfangs hatte ich Probleme, weil er die Unterschiede der Farben nicht zuverlässig erkannt hat. Durch gründliche Recherchen und intensives Nachlesen habe ich jedoch gelernt, wie er richtig eingesetzt werden muss, sodass er schliesslich stabil und zuverlässig funktioniert hat.

3.3.2. Leistungsreflexion

Mit meinem Arbeitsfortschritt bin ich insgesamt sehr zufrieden. Ich konnte meine Arbeitsschritte planmäßig umsetzen und die Ergebnisse entsprachen meinen Erwartungen.

Besonders positiv war, dass das Fahrzeug zuverlässig die Linien erkannte, die roten Markierungen beachtete und die Parkvorgänge korrekt ausführte. Meine Arbeitsweise war konzentriert und zielgerichtet, wodurch ich die Aufgaben effizient bearbeiten konnte. Insgesamt bin ich stolz auf das erreichte Resultat und freue mich über den Erfolg des Projekts.

3.3.3. Lernreflexion

Während des Projekts habe ich viele wertvolle Erfahrungen gesammelt. Ich habe gelernt, wie wichtig ein sauberer und durchdachter Aufbau ist, damit Sensoren und Motoren korrekt arbeiten können. Außerdem habe ich erkannt, dass gezielte Recherche und das Nachlesen in Dokumentationen entscheidend sind, um Probleme zu lösen und Sensoren optimal einzusetzen. Besonders motivierend war, dass mir die Arbeit mit dem EV3 viel Spass gemacht hat und ich dadurch auch mehr Interesse an der Robotik gewonnen habe. Für zukünftige Projekte nehme ich mir vor, noch stärker mögliche Erweiterungen mitzudenken, zum Beispiel zusätzliche Funktionen zu planen, wenn beide Parkplätze belegt sind oder wenn nach einer gewissen Zeit ein neuer Suchvorgang gestartet werden soll. Insgesamt war dieses Projekt für mich ein sehr guter Einstieg in die Robotik und hat mir gezeigt, wie viele Möglichkeiten im EV3 stecken.

4. Fazit

Das Projekt „Autonomes Parken mit dem EV3“ war ein voller Erfolg. Alle definierten Muss-Ziele wurden erreicht, und auch einige Erweiterungen konnten umgesetzt werden. Das Fahrzeug war in der Lage, selbstständig einer Linie zu folgen, an Markierungen anzuhalten und Parkplätze sowohl auf der rechten als auch auf der linken Seite zu erkennen und anzufahren. Damit konnte die Kernfunktionalität – ein vereinfachtes Einparkassistenzsystem – praxisnah umgesetzt werden.

Besonders wertvoll war, dass die Tests bestätigten, dass das Fahrzeug zuverlässig arbeitet und die Abläufe stabil bleiben. Auch wenn kleinere Optimierungen möglich wären, zum Beispiel eine Wiederholungsprüfung bei belegten Parkplätzen oder eine

Weiterfahrt zur nächsten Station, erfüllt das System bereits alle wesentlichen Anforderungen.

Darüber hinaus hat das Projekt mir gezeigt, wie vielschichtig die Arbeit mit Robotiksystemen ist: Von der Planung über den Aufbau bis hin zur Programmierung war jeder Schritt wichtig und trug zum Gesamtergebnis bei. Ich konnte technisches Wissen anwenden und erweitern, gleichzeitig aber auch meine Arbeitsweise verbessern und mehr Sicherheit im Umgang mit Sensorik und Programmabläufen gewinnen.

Insgesamt war die Projektarbeit für mich eine sehr lehrreiche und motivierende Erfahrung. Sie hat mir nicht nur fachliches Können vermittelt, sondern auch gezeigt, wie spannend und praxisnah die Umsetzung von autonomen Systemen sein kann.

5. Quellenverzeichnis

5.1. Literaturverzeichnis

5.2. Abbildungen

Abbildung 1 EV3 Fahrzeug Aufbau.....	6
Abbildung 2 EV3	7
Abbildung 3 Mittlere Motor	8
Abbildung 4 Starke/Grosse Motor.....	8
Abbildung 5 Ultraschallsensor	9
Abbildung 6 Farbsensor.....	9
Abbildung 7 Programmcode Teil 1.....	10
Abbildung 8 Programmcode Teil 2.....	12
Abbildung 9 Programmcode Teil 3.....	13
Abbildung 10 Fahrbahn.....	15

5.3. Tabelle

Tabelle 1 SOLL und IST	16
Tabelle 2 Aufwand	16

5.4. Hilfsmitteln

Für die grammatischen Korrekturen, sprachliche Überarbeitung und für Ideen oder bei Schwierigkeiten wurden digitale Hilfen (z. B. KI-gestützte Tools), Google, Unterrichtsfolien sowie der Dozent als Unterstützung genutzt.

6. Eigenständigkeitserklärung

Die verfassenden Personen bestätigen mit ihrer Unterschrift, dass die vorliegende Arbeit selbstständig, ohne fremde Hilfe und ohne Benutzung anderer als die angegebenen Hilfsmittel erstellt wurde.

Die aus fremden Quellen (einschliesslich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit ist in gleicher oder ähnlicher Form noch nicht vorgelegt worden.

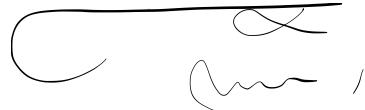
Ort, Datum

Vorname Nachname

Unterschrift

Biel, 20.09.2025

Sirwan Salihi



7. Anhang

Aufbau Auto Im TEAMS: Allgemein -> General -> Kursmaterialien -> 12 Übungsserien -> Übung 08