HTW D Hochschule für Technik und Wirtschaft Dresden University of Applied Sciences

Diplomarbeit

Entwicklung einer Roboter-Ansteuerung

Vorgelegt von:	Yizhe Gu
Ort:	HTW Dresden
Fakultät	Maschinenbau
Studiengang:	Fahrzeugtechnik
Betreuer	Prof. Dr. rer. Nat. Toralf Trautmann
	Dr. Duo Fu
	Dipl.-Ing. Dirk Engert
	Dipl.-Ing. Franziskus Mendt
Abgabedatum	02.09.2024

Inhaltverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Formelzeichen und Symbole

Zeichen	Einheit	Bedeutung
W	°	aktueller Kurswinkel
ω	°/s	Winkelgeschwindigkeit der z-Achse des Gyroskops
s	m	Entfernung des Ultraschallsensors zum gemessenen Objekt
v_w	m/s	Schallgeschwindigkeit des Ultraschalls
t	s	Übertragungszeit des Ultraschalls
n	r/s	Drehzahl des Motors
Δt	s	die Zeitdifferenz zwischen zwei aufeinanderfolgenden steigenden Flanken
t'	ms	Dauer des High-Pegels
t'_{min}	ms	Minimale Dauer des High-Pegels
t'_{max}	ms	Maximale Dauer des High-Pegels
P_{out}	/	Ausgang des Proportionalreglers im PID-Steuerungssystem
K_p	/	Proportionalverstärkung
e(t)	/	Aktueller Fehlerwert
I_{out}	/	Ausgang des Integralreglers im PID-Steuerungssystem
K_i	/	Integralverstärkung
D_{out}	/	Ausgang des Differentialreglers im PID-Steuerungssystem
K_d	/	Differentialverstärkung
PID_{out}	/	Gesamtausgang des PID-Reglers
P	/	Vertrauenswahrscheinlichkeit
$t_{0.68}$	/	t-Wert bei einer Vertrauenswahrscheinlichkeit von 0,68
$t_{0.95}$	/	t-Wert bei einer Vertrauenswahrscheinlichkeit von 0,95
$t_{0.99}$	/	t-Wert bei einer Vertrauenswahrscheinlichkeit von 0,99
$\bar{x}$	/	Mittelwert
σ_x	/	Standardabweichung
u_A	/	Typ-A-Standardunsicherheit
UA	/	Typ-A-Unsicherheit
Tl/tl	/	Tastverhältnis des linken Motors des Roboters
Tr/tr	/	Tastverhältnis des rechten Motors des Roboters
re_w	°	Referenzwinkel
re_v	r/s	Referenzdrehzahl
W_{out}	°	Tatsächlicher Drehwinkel des Roboters
W_{in}	°	Der mittels Gyroskop berechnete Winkel
w	m	Zurückgelegte Strecke des Roboters
w_y	m	Auf die vorgegebene Trajektorie projizierte Strecke
α_1	°	Kursabweichungswinkel des Roboters am Anfang
p	°	Aktueller Kursabweichungswinkel des Roboters
Winkel_delta	°	Winkelkorrektur zur Rückkehr auf die geplante Trajektorie
v	m/s	Momentangeschwindigkeit
w1	°	Fortlaufend iterierter Winkel der vorgegebenen Trajektorie
w2	°	Kurswinkel bei der Rückkehr auf die Trajektorie
Winkel_delta1	°	Differenz zwischen aktuellem Kurswinkel und Trajektorienwinkel

d1/d2	m	der vertikale Abstand zur Wand
h	/	Markierungsparameter, 0 vor und 1 nach der Wende
E	/	Gewichtetes Bewertungsergebnis
z	/	Die Obergrenze des Zählers (=50)
k1	°/s	Winkeländerungsrate
a1	°	Erst aufgezeichneter Winkel
a2	°	Winkel nach 50 Aktualisierungen
$\bar{x}$	/	Mittelwert
σ^2	/	Varianz

Abkürzungsverzeichnis

RPi	Raspberry Pi
VLR	Vorwärtsdrehung des linken Rads
VRR	Vorwärtsdrehung des rechten Rads
RLR	Rückwärtsdrehung des linken Rads
RRR	Rückwärtsdrehung des rechten Rads
TV	Tastverhältnis
HP	High-Pegel
AMRK	Aluminium-Motorhalterungs- und Rad-Kit
BSB	Blei-Säure-Batterien
DHCP	Dynamic Host Configuration Protocol
l	link
r	recht
TVO	Trajektorienverfolgungsoptimierung
SSM	Stufenschwächungsmethode
AA	Anzahl der Anpassungen
VAW	der vertikale Abstand zur Wand
SET	zurückgelegte Strecke entlang der Trajektorie
RKZ	Rückkehrzeit
SÜ	Straßenüberquerung
GE	Gästeempfang
SZ	Steuerungszeit

1 Einleitung

1.1 Motivation

Mit der rasanten Entwicklung der Robotikindustrie werden Roboter aufgrund ihrer Vorteile in Bezug auf Arbeitsraum und Kosten in verschiedenen Bereichen wie Industrie, Dienstleistungssektor und Forschung breit eingesetzt. So können beispielsweise Pflegeroboter Patienten pflegerische Dienstleistungen bieten, während Saugroboter im Alltag den Menschen Bequemlichkeit verschaffen[1]. Ein gemeinsames Merkmal dieser Roboter bei ihrer Arbeit ist die Spurverfolgung. Diese Problematik hat in der Robotikforschung große Beachtung gefunden und umfasst verschiedene Module wie Kommunikation, Antrieb und Steuerung. Die koordinierte Zusammenarbeit dieser Module gewährleistet die Genauigkeit der Bewegungsbahnen des Roboters. Vor diesem Hintergrund zielt diese Arbeit darauf ab, die Bewegungsbahn des Arlo-Roboters zu entwerfen.。

1.2 Projektziel

Im autonomen Fahrversuchsprojekt müssen Dummys eingesetzt werden, um die Aufgaben des autonomen Fahrens zu testen. Die derzeit verwendeten Dummys sind jedoch nur in der Lage, relativ einfache Bewegungsmuster auszuführen und können sich nicht gemäß den Anforderungen der Experimente bewegen. Um die Dummys dazu zu bringen, vielfältige Bewegungsanforderungen zu erfüllen, wie beispielsweise das „Überqueren der Straße“, stützt sich dieses Projekt auf die mobile Roboterplattform Arlo, die von der Firma Parallax entwickelt wurde. Und dem Roboter Befehle übermitteln, um eine Spurverfolgung zu gewährleisten. Das Projekt wird von Prof. Dr. rer. nat. Toralf Trautmann geleitet und von Dr.-Ing. Duo Fu, Dr.-Ing. Dirk und Dr. Ing. Franziskus Mendt betreut.

2 Grundlagen

Als zentrales Robotik-Plattform dieses Projekts bietet das Arlo-Robotersystem eine solide Grundlage für den Bau mittelgroßer mobiler Roboter in Fakultäten und Forschungseinrichtungen. Dieses Kit umfasst eine Plattform mit einem Durchmesser von 45cm, zwei Gleichstromantriebe mit luftbereiften Rädern, einen frei programmierbaren Motorcontroller und Ultraschallsensoren. Zwei Batterien liefern eine Energie von 170Wh. Der Hauptschalter, der Spannungsregler und die Klemmen sind auf einer Verteilerplatte montiert. Kugellager und leicht zu bedienende Laufrollen verleihen der Plattform die notwendige Stabilität[2].

2.1 Raspberry Pi und Zubehör

2.1.1 Raspberry Pi

In dieser Arbeit wurde der Einplatinencomputer RPi 3B verwendet (siehe Hardware 1 in Anlage 1), der über einen 8GB Arbeitsspeicher und einen 64-Bit-Zentralprozessor (CPU) verfügt. Dieser Einplatinencomputer ist mit einer Vielzahl von GPIO-Schnittstellen ausgestattet und unterstützt die Konfiguration und Installation verschiedener Deep-Learning-Frameworks, was dem Experiment eine leistungsstarke Rechen- und Erweiterungsfähigkeit verleiht.

In der Regel kann die Entwicklung von Robotern auch mit Arduino erfolgen, jedoch weist Arduino im Vergleich zum RPi einen signifikanten Nachteil auf. Ohne die zusätzliche Installation von drahtlosen Verbindungsmodulen wie ESP8266 oder ESP32 kann Arduino nur über ein USB-Datenkabel mit dem Computer verbunden werden, was den Bewegungsraum des Roboters während der Entwicklung einschränken kann. Aus diesem Grund wurde in diesem Projekt der RPi 3B als zentrale Entwicklungsplattform für den Roboter gewählt, da er über ein drahtloses Netzwerk mit dem Computer verbunden werden kann. Dies bedeutet, dass der Roboter Befehle vom Computer empfangen und ausführen kann, solange sich beide im selben lokalen Netzwerk befinden. Für die Verbindung wurde RealVNC Connect verwendet[3].

2.1.2 Ultraschalsensor

Im Entwicklungsprozess der Spurverfolgung bei Robotern spielt das Distanz-Feedback eine entscheidende Rolle. In diesem Projekt wird der HC-SR04 Ultraschallsensor (sehe Anlage 3) eingesetzt, um dieses Ziel zu erreichen. Der Sensor verfügt über einen Signalausgang, einen Signalempfangsbereich und vier Pin-Anschlüsse, die jeweils für VCC, Trig, Echo und GND vorgesehen sind.

- VCC：Dies ist der Funktionsanschluss des Distanzmessungsmoduls, der das Modul mit einer Spannung von etwa 5V versorgt. Im Experiment wird dieser Anschluss mit dem 5V GPIO des Raspberry Pi verbunden.

- Trig：Dieser Pin empfängt das Steuersignal vom Raspberry Pi. Im Experiment ist er mit GPIO27 verbunden.
- Echo：Dieser Pin sendet das Messergebnis der Distanzmessung an den Raspberry Pi. Im Experiment wird er nach einer Parallelverbindung mit einem Spannungsteilerkreis an den Raspberry Pi übertragen.
- GND：Dieser Pin ist für die Erdung (Masse) zuständig und stellt die Verbindung zur Erdung des gesamten Schaltkreises her.

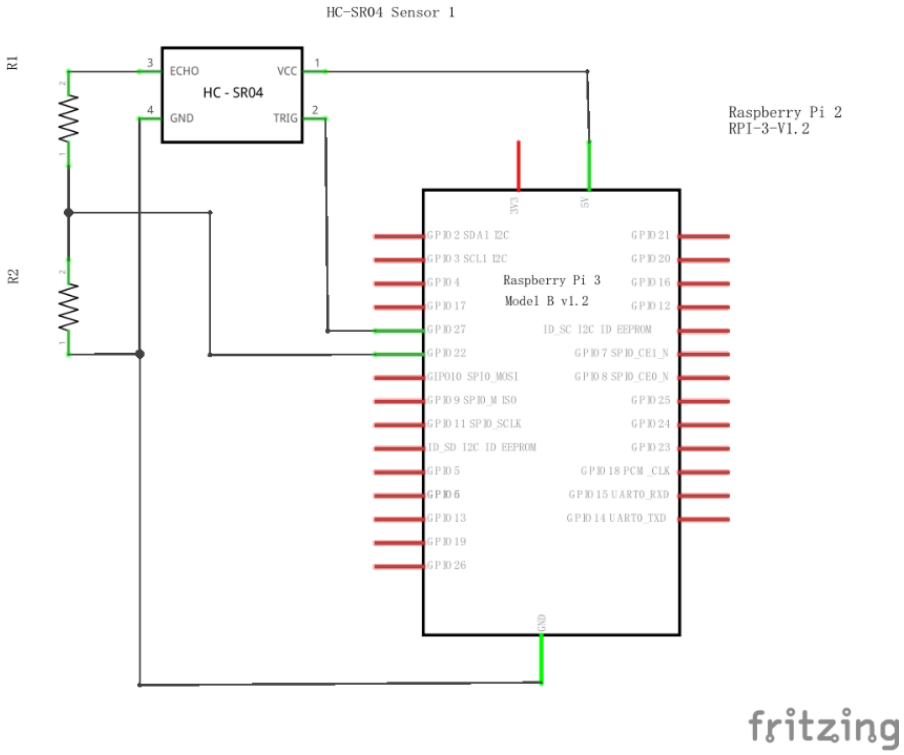


Abbildung 2.1 Schaltplan zwischen HC-SR04 und Raspberry Pi

Wie in Abbildung 2.1 und Anlage 4 dargestellt, können die Schnittstellen VCC, Trig und GND direkt mit den entsprechenden Ports des Raspberry Pi (RPi) verbunden werden. Da jedoch die GPIO-Ports des RPi maximal nur 3,3V Spannung akzeptieren können, muss am Echo-Pin ein Spannungsteiler-Schaltkreis hinzugefügt werden, um die Spannung des Rücksignals auf einen sicheren Bereich zu reduzieren. In der Abbildung wurden hierfür ein 1kΩ- und ein 2kΩ-Widerstand verwendet. Die spezifische Berechnungsmethode ist wie folgt:

Der Ausgangsspannung des Echo-Pins beträgt 5V. Um diese Spannung auf unter 3,3V zu reduzieren, kann ein Spannungsteiler verwendet werden.

Nach der Spannungsteiler-Formel:

$$V_{out} = V_{in} * \frac{R_2}{R_1 + R_2} \tag{2.1}$$

Dabei ist V_{out} die Ausgangsspannung, V_{in} die Eingangsspannung, und R_1 und R_2 sind die Widerstandswerte im Spannungsteiler. In diesem Fall gilt V_{in}=5V, R_1 = 1kΩ, und R_2 = 2kΩ. Setzt man diese Werte in die Formel ein, erhält man:

$$V_{out} = 5V * \frac{2k\Omega}{1k\Omega + 2k\Omega} = 3.33V \quad (2.2)$$

Auf diese Weise wird durch die Verwendung von 1kΩ- und 2kΩ-Widerständen die vom Echo-Pin an den Raspberry Pi gesendete Spannung auf 3,33V reduziert, wodurch sichergestellt wird, dass die Signalspannung innerhalb des sicheren Empfangsbereichs des Raspberry Pi liegt. Durch diese Verbindungen kann der HC-SR04 Ultraschallsensor während der Bewegung des Roboters in Echtzeit Entfernungsdaten bereitstellen, sodass der Roboter präzise der vorgegebenen Spur folgen kann[4].

2.2 Antriebseinheit

Das wichtigste Antriebselement in diesem Projekt ist der Motor, der in diesem Kapitel zusammen mit den anderen im Projekt verwendeten Geräten behandelt wird.

2.2.1 Aluminium-Motorhalterungs- und Rad-Kit [5]

Das Antriebssystem des Arlo-Roboters umfasst ein Aluminium-Motorhalterungs- und Rad-Kit (Motor Mount and Wheel Kit – Aluminum, kurz AMRK), das mit zwei versiegelten Blei-Säure-Batterien (Sealed Lead Acid Battery, kurz BSB) verbunden ist. Die Batterien haben eine Betriebsspannung von 12V und eine Kapazität von 7,2Ah, was die benötigte Energie für das gesamte Antriebssystem bereitstellt. Die dazugehörigen Motoren haben eine Leistung von 96W und können zwei Räder mit einem Durchmesser von 150mm antreiben, wodurch ein synchroner Antrieb der linken und rechten Räder ermöglicht wird. Das Antriebssystem kann eine maximale Geschwindigkeit von 75 cm/s erreichen, was den Anforderungen der meisten mobilen Roboter entspricht. Dieses Kit ist unter der in Anlage 1 gezeigten Plattform montiert. Nach dem Ausbau weist es Struktur wie Anlage 5 auf.

2.2.2 36-Positionen-Encoderscheibe

Auf dem AMRK ist auch ein für die Geschwindigkeitsmessung sehr wichtiges Modul montiert – der optische Encoder (siehe die zwei Chips in Anlage 5). Dieses Modul umfasst eine 36-Positionen-Encoderscheibe, die an der Motorwelle befestigt ist, sowie eine fest montierte Sensoreinheit (siehe Anlage 6).

Die Sensoreinheit des optischen Encoders besteht aus einem Infrarot-Emitter und einem Infrarot-Detektor. Diese Komponente nutzt das Prinzip der Abschattung und Durchlässigkeit von Infrarotlicht, um die Position der Encoderscheibe zu erkennen.

- Infrarot-Emitter: Sendet einen Infrarotstrahl aus, der durch die Markierungen oder Löcher auf der Encoderscheibe hindurchtritt.

- Infrarot-Detektor: Empfängt den Infrarotstrahl. Wenn der Strahl durch die Erhebungen auf der Encoderscheibe blockiert wird, kann der Detektor das Infrarotsignal nicht empfangen.

Wenn sich die Encoderscheibe zusammen mit dem Motor dreht, blockieren die 36 Erhebungen auf der Scheibe nacheinander den Infrarotstrahl. Die Sensoreinheit markiert den blockierten Zustand als Low-Pegel (0V) und den durchlässigen Zustand als High-Pegel (1V). Somit erzeugt der Sensor ein Rechteckwellensignal, bei dem sich 0 und 1 abwechseln, während sich die Motorwelle dreht. Anhand dieses Signals kann die Drehzahl des Motors weiter berechnet werden. Der Berechnungsansatz ist wie folgt:

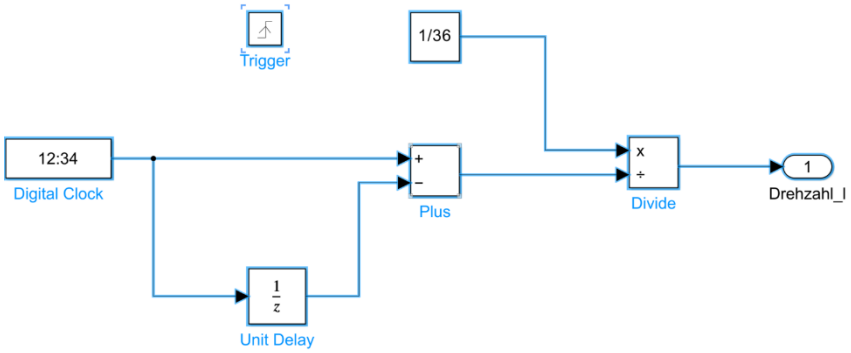


Abbildung 2.2 Algorithmus zur Berechnung der Drehzahl des Motors

In dieser Logik durchsucht das Trigger-Modul kontinuierlich den vom HC-SR04 bereitgestellten Rechteckwellensignal nach steigenden Flanken. Sobald eine steigende Flanke erkannt wird, wird die Berechnungslogik des Triggers aktiviert, um die Zeitdifferenz zwischen zwei aufeinanderfolgenden steigenden Flanken zu berechnen, wodurch die Zeit Δt ermittelt wird, die die Encoderscheibe benötigt, um eine Position zu durchlaufen. Da die Encoderscheibe 36 Gitter pro Umdrehung hat, ergibt sich folgende Formel zur Berechnung der Drehzahl:

$$n = \frac{1}{\Delta t * 36} r/s \tag{2.3}$$

Nach der Berechnung der Drehzahl kann die lineare Geschwindigkeit der linken und rechten Räder durch die folgenden Formeln ermittelt werden.

$$v = 2 * \pi * r * n \tag{2.4}$$

Angewendet auf das spezifische Simulink-Programm ergibt sich das folgende Programm.

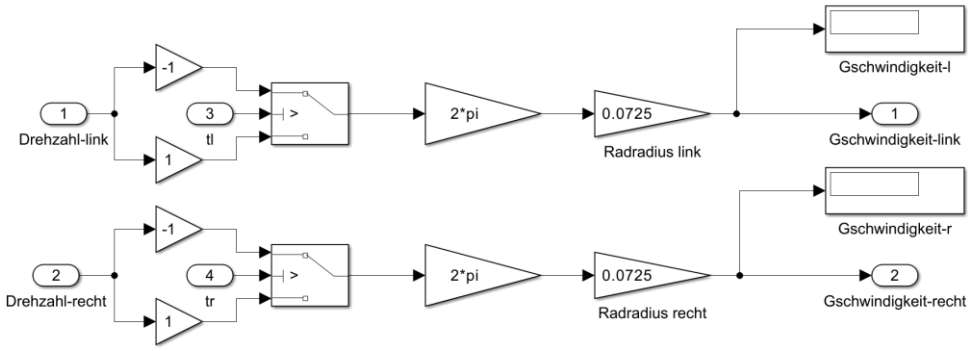


Abbildung 2.3 Geschwindigkeit-Berechnung

Nachdem die Geschwindigkeit der linken und rechten Räder berechnet wurde, wird die zurückgelegte Strecke des Roboters über einen bestimmten Zeitraum durch Akkumulation berechnet[6]. Während des Betriebs wird die Strecke des Roboters kontinuierlich durch die Integration der Geschwindigkeit über die Zeit aktualisiert. Um diesen Prozess zu realisieren, wurde in Simulink mithilfe des MATLAB Function-Blocks ein Streckenakkumulator erstellt, wie in der folgenden Abbildung dargestellt.

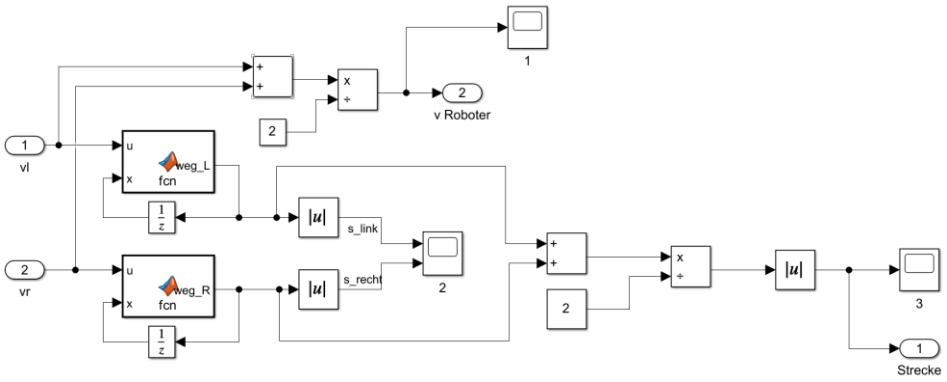


Abbildung 2.4 Streckenakkumulator

Die linearen Geschwindigkeiten der linken und rechten Räder werden jeweils in den entsprechenden MATLAB Function-Block eingespeist. In diesem Block summiert ein vordefiniertes Programm das Produkt aus Geschwindigkeit und Abtastzeit, um die während des gesamten Betriebs vom linken und rechten Rad akkumulierten Strecken präzise zu berechnen [7].

```
function weg_L = fcn(vl,x)
x=x+vl*0.001;
weg_L = x;
```

```
function weg_R = fcn(vr,x)
x=x+vr*0.001;
weg_R = x;
```

Abbildung 2.5 Streckenakkumulationsprogramm

Im Experiment wurde eine Abtastzeit von 0,001s verwendet. Bei jeder Abtastung addiert das Programm die in diesem Zeitraum zurückgelegte Strecke des Roboters zur Streckenvariable x.

Schließlich wird die Gesamtstrecke, die der Roboter während des gesamten Experiments zurückgelegt hat, durch die Berechnung des Durchschnitts der Strecken vom linken und rechten Rad ermittelt.

2.3 Verteilerboard und Motorsteuerung

2.3.1 Verteilerboard und DC-DC-Abwärtswandler

Das Verteilerboard (siehe Anlage 7) ist eine relativ komplexe Vorrichtung, die aus Wippschalter (Rocker Switch), Halter, Mini-Sicherung (Holder, Mini Fuse), 1kΩ-, 5,3kΩ- und 7,15kΩ-Widerständen, Anschlussbuchse (Terminal Receptacle), 2-Positionen-Schraubklemme (2 Position Screw Terminal) und weiteren Komponenten besteht. Bei korrekter Verbindung kann es 12V Gleichstrom freigeben, um den Roboter mit der notwendigen Energie für seine Bewegung zu versorgen.

Um eine drahtlose Steuerung zu ermöglichen, wird der RPi ebenfalls von einer Batterie mit Strom versorgt. Da die Nennbetriebsspannung des RPi jedoch bei etwa 5V liegt, muss eine DC-DC-Abwärtswandler (siehe Anlage 8) zwischengeschaltet werden, um sicherzustellen, dass die dem RPi zugeführte Spannung konstant bei 5V bleibt.

2.3.2 Motorsteuerung DHB-10

Im Projekt verwendet der Arlo-Roboter das Motorsteuerung DHB-10 (10 Ampere Dual-H-Brücke-Motorsteuerung, siehe Anlage 9), um den Motoren das für die Steuerung der Drehbewegung erforderliche PWM-Signal (Pulsweitenmodulation) bereitzustellen.

Wenn die linken und rechten Motoren jeweils mit den Motor1- und Motor2-Schraubklemmen verbunden sind, kann der Roboter serielle und PWM-Eingaben von CH1 und CH2 empfangen und somit beide Motoren antreiben.

PWM (Pulsweitenmodulation) ist ein Signal, bei dem die grundlegende Idee darin besteht, die mittlere Spannung in einem Stromkreis durch kontinuierliche Veränderung der Pulsbreite zu steuern, um so eine präzise Kontrolle des Stromkreises zu erreichen. Die wichtigsten Parameter sind dabei die Frequenz und das Tastverhältnis (Duty Cycle, kurz TV).

- Frequenz: Die Anzahl der Wiederholungen des Signals pro Sekunde.

- TV (Duty Cycle): Der Anteil der Zeit, in der das Signal auf einem hohen Spannungsniveau bleibt, gemessen über die gesamte Periodendauer des Signals.

2.4 Kommunikation zwischen PC und den Roboter

Die Verbindung zwischen dem Roboter und dem PC wird im Experiment über WLAN realisiert. Im Folgenden wird das Topologiediagramm der drahtlosen Netzwerkverbindung dargestellt:

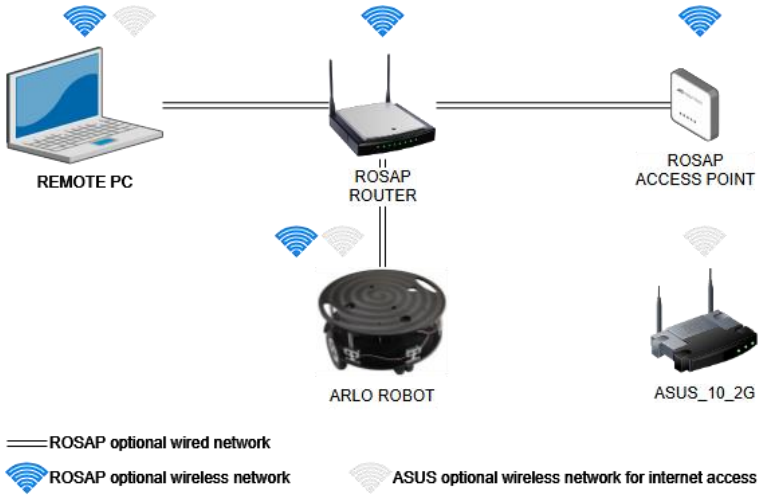


Abbildung 2.6 Local Network (Mendt, Franziskus, 2023)[8]

In dieser Netzwerk-Topologie sind mehrere Geräte über drahtlose Kommunikation miteinander verbunden. Das zentrale Gerät ist ein Router namens „ROSAP ROUTER“, der über ein Wi-Fi-Signal mit den anderen Geräten verbunden ist und ein lokales Netzwerk bildet. Die Geräte in diesem WLAN umfassen:

- Laptop: Als ein Client-Gerät im Netzwerk ist der Laptop über Wi-Fi mit dem ROSAP-Router verbunden. Er kann über den Router auf andere Geräte im Netzwerk zugreifen.
- ROSAP ACCESS POINT: Dieses Gerät fungiert als ein drahtloser Netzwerkzugangspunkt, der über Wi-Fi mit dem ROSAP-Router kommuniziert und den Abdeckungsbereich des Netzwerks erweitert, sodass mehr Geräte eine Verbindung zum Netzwerk herstellen können.
- ARLO ROBOT: Dieses Gerät ist der Roboter, der im Experiment verwendet wird. Der RPi auf der Roboterplattform (siehe Anlage 1) ist über Wi-Fi mit dem ROSAP-Router verbunden, was bedeutet, dass der Roboter mit anderen Geräten im Netzwerk, wie dem Laptop oder dem Zugangspunkt, kommunizieren kann.
- ASUS_10_2G: Dies ist ein weiterer Wi-Fi-Router, das unabhängig vom ROSAP-Netzwerk arbeitet. Seine Anwesenheit stellt sicher, dass ein alternatives drahtloses Netzwerk zur Verfügung steht, um die Kommunikation zwischen dem Roboter und dem PC zu gewährleisten, falls das ROSAP-WLAN Probleme bereitet.

Während des Experiments, wenn sich der Roboter im Labor befindet, kann das drahtlose Netzwerk ASUS in den meisten Fällen die Kommunikationsanforderungen zwischen dem Roboter und dem PC erfüllen. Durch Eingabe des Befehls „ipconfig“ in die Eingabeaufforderung auf dem

PC wird die IP-Adresse des PCs im aktuellen Netzwerkumfeld angezeigt. Die folgende Abbildung zeigt die IP-Adresse des PCs, wenn er mit dem drahtlosen Netzwerk ASUS verbunden ist.

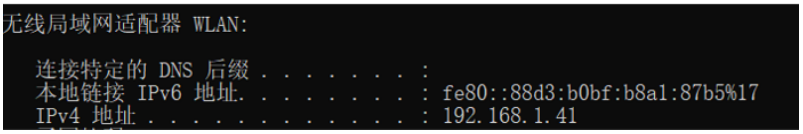


Abbildung 2.7 IP-Adresse des PCs im Netzwerkumfeld ASUS

Nachdem der RPi mit einem Bildschirm, einer Maus und einer Tastatur verbunden wurde, kann durch Eingabe des Befehls „ifconfig“ im Terminalfenster die IP-Adresse des RPi im aktuellen Netzwerkumfeld angezeigt werden. Die folgende Abbildung zeigt die IP-Adresse des RPi im ASUS-Netzwerk:

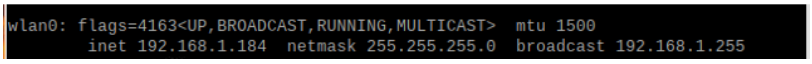


Abbildung 2.8 IP-Adresse des Roboters im Netzwerkumfeld ASUS

In Bereichen ohne ASUS-WLAN-Signal wird das ROSAP-Wi-Fi zur Kommunikation zwischen Roboter und PC verwendet. Der Router ROSAP und der zugehörige Access Point werden durch eine Powerbank betrieben. Sobald der Roboter im ROSAP-WLAN-Bereich ist, sollte im Terminal „sudo nano /etc/wpa_supplicant/wpa_supplicant.conf“ eingegeben werden, um das gewünschte Netzwerk auszuwählen(siehe Anlage 10). Nach einem Systemneustart wird die Verbindung automatisch hergestellt.

Nach erfolgreicher Herstellung der Verbindung können die IP-Adressen des PCs und des Roboters im ROSAP-Netzwerkumfeld auf ähnliche Weise abgerufen werden. Da im DHCP die Anfangs- und Endwerte für die IP-Adressvergabe wie in der Anlage 11 festgelegt sind, haben der PC und der Roboter, wenn sie nacheinander in das drahtlose Netzwerk ROSAP eingebunden werden, die folgenden IP-Adressen.

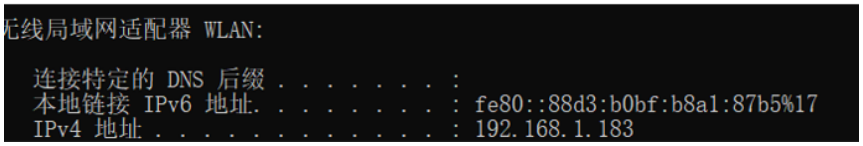


Abbildung 2.9 IP-Adresse des PCs im Netzwerkumfeld ROSAP

Abbildung 2.10 IP-Adresse des Roboters im Netzwerkumfeld ROSAP

Die IP-Adresse des Roboters wird in beide Netzwerkumfeld auf „192.168.1.184“ festgelegt, was den Kommunikationsprozess zwischen dem Roboter und dem PC vereinfacht. Da die IP-Adresse bei jeder Kommunikation über VNC nicht geändert werden muss, erhöht diese Einstellung die Effizienz der Experimente erheblich.

3 Konzept

3.1 Winkelvermessung

Auf dem RPi wurde zusätzlich das Sense HAT-Modul installiert. Dieses Modul integriert verschiedene Sensoren wie Gyroskop, Beschleunigungsmesser, Magnetometer, Thermometer und Barometer. Für dieses Projekt ist insbesondere der Gyroskopsensor LSM9DS1 von Bedeutung. Wie in der Anlage 2 gezeigt, ist das Sense HAT direkt auf der Oberseite des Raspberry Pi montiert. Das Koordinatensystem ist wie folgt definiert: Die positive x-Achse zeigt nach links, die positive y-Achse nach hinten und die positive z-Achse nach unten. Wenn das Sense HAT auf dem Roboter montiert ist, entsprechen die drei Achsen weiterhin den Richtungen links, hinten und unten. In den Experimenten wurden die Daten der Winkelgeschwindigkeit um die z-Achse verwendet. Durch die Integration der Winkelgeschwindigkeit kann der aktuelle Winkel des Roboters ermittelt werden.

$$W = \int_{t_0}^{t} \omega * dt \tag{3.1}$$

Es ist zu beachten, dass der Roboter beim Messen des Winkels stets die aktuelle Richtung der Vorderseite als 0° betrachtet. Im Messprozess wird die Drehung im Uhrzeigersinn als negativ und die Drehung gegen den Uhrzeigersinn als positiv gewertet. Um eine präzise Navigation zu gewährleisten, ist es außerdem erforderlich, während der Programmierung eine zusätzliche Kalibrierung vorzunehmen, um die Genauigkeit der Winkelmessung sicherzustellen

3.2 Abstandvermessung

Die wichtigste Funktion des HC-SR04 in diesem Projekt besteht darin, den Abstand zwischen dem Arlo-Roboter und der Wand zu messen. Das Prinzip der Distanzmessung wird in der folgenden Abbildung erläutert:

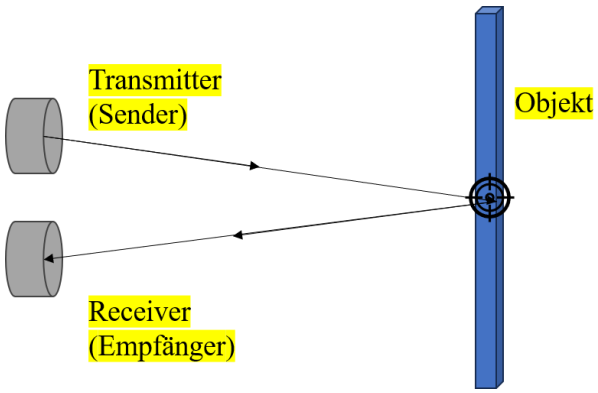


Abbildung 3.1 Prinzip der Ultraschall-Entfernungsmessung

Der Sender (Transmitter) emittiert Ultraschallwellen, die beim Auftreffen auf ein Objekt reflektiert werden. Der Empfänger (Receiver) erfasst die reflektierten Wellen und das System berechnet

intern die dafür benötigte Zeit. Mit Hilfe dieser Zeit und der Ausbreitungsgeschwindigkeit der Ultraschallwellen in der Luft kann die Entfernung des HC-SR04 Ultraschallsensors zum gemessenen Objekt berechnet werden. Die Herleitung der entsprechenden Formel lautet wie folgt:

$$s = v_w * \frac{t}{2} \tag{3.2}$$

$$v_w = \left(331{,}5 + 0{,}6 * \frac{\vartheta}{°C}\right) m/s \tag{3.3}$$

Bei einer Umgebungstemperatur von 20°C beträgt die Schallgeschwindigkeit in Luft etwa:

$$v_w = \frac{\left(331{,}5 + 0{,}6 * \frac{20°C}{°C}\right) m}{s} = 343{,}5 m/s \tag{3.4}$$

Durch das Einsetzen der Schallgeschwindigkeit bei 20°C in die Gleichung (3.2) ergibt sich:

$$s = 343{,}5 * \frac{t}{2} \tag{3.5}$$

Das Prinzip zur Bestimmung von t in der Gleichung (3.5) ist wie folgt:

Wenn der Trig-Pin ein 10 µs langes TTL-Signal (Transistor-Transistor Logic) vom Raspberry Pi erhält, beginnt der Sender, Ultraschallimpulse auszusenden, und gleichzeitig wird der Echo-Pin auf ein hohes Spannungsniveau (5V) gesetzt. Sobald der Empfänger das reflektierte Signal erfasst, kehrt der Echo-Pin auf ein niedriges Spannungsniveau (0V) zurück. Daher kann der Wert von t als die Zeit definiert werden, während der der Echo-Pin auf einem hohen Spannungsniveau bleibt.

Um den oben beschriebenen Prozess besser zu verstehen, wird im Folgenden ein konkretes Beispiel erläutert:

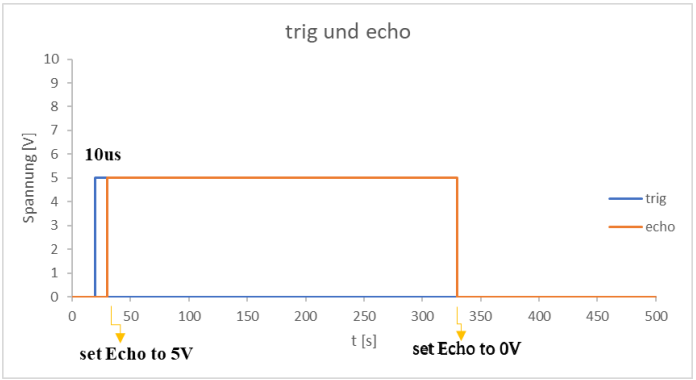


Abbildung 3.2 Prinzip der Zeitmessung des HC-SR04

Gemäß den Daten in Abbildung 3.2 beträgt die Zeit, in der der Echo-Pin auf einem hohen Spannungsniveau bleibt, 300 µs. Unter Verwendung der Gleichung (3.5) ergibt sich die Berechnung wie folgt:

$$s = 343{,}5m/s * \frac{0{,}0003s}{2} = 0.052m \tag{3.6}$$

Im Experiment müssen noch einige Details beachtet werden. Nachdem der RPi einen 10-Mikrosekunden-TTL-Puls an den HC-SR04 gesendet hat, sendet das Modul zyklisch 8 Ultraschallimpulse mit einer Frequenz von 40 kHz aus. Die Verwendung des „8-Puls“-Verfahrens hat keinen direkten Einfluss auf die Messung selbst, verbessert jedoch die Messgenauigkeit. Ein einzelner Ultraschallimpuls könnte durch Umgebungsstörungen wie Lärm beeinträchtigt werden, was zu instabilen Echo-Signalen führen könnte. Durch die Erhöhung der Anzahl der Impulse auf 8 wird die Signalstärke verstärkt, was zu genaueren und zuverlässigeren Messergebnissen führt.

In den folgenden Experimenten wird dieser Ultraschallsensor an der rechten Seite des Roboters montiert, um den Abstand zur rechten Wand, der senkrecht zur Fahrtrichtung verläuft, zu messen. Diese Anordnung und Messmethode helfen dem Roboter, während der Fahrt einen konstanten Abstand zur Wand beizubehalten, wodurch die Navigationsstabilität und -sicherheit verbessert wird.

In der Praxis müssen folgende Punkte beachtet werden:

1. Montageposition: Stellen Sie sicher, dass der Ultraschallsender und -empfänger des Sensors direkt auf das zu messende Ziel ausgerichtet sind und dass keine Hindernisse die Sichtlinie blockieren.

2. Umgebungsgeräusche: Führen Sie die Messungen möglichst in einer Umgebung mit wenig Lärm durch oder verwenden Sie Filter, um die Auswirkungen von Umgebungsgeräuschen zu minimieren.

3. Temperatur und Luftfeuchtigkeit: Da die Ausbreitungsgeschwindigkeit von Ultraschallwellen durch Temperatur und Luftfeuchtigkeit beeinflusst wird, sollten die Messungen möglichst unter konstanten Umgebungsbedingungen durchgeführt werden. Alternativ können entsprechende Kompensationsberechnungen vorgenommen werden.

4. Datenverarbeitung: Durch Mittelung mehrerer Messungen kann die Messgenauigkeit weiter verbessert werden

Durch diese Maßnahmen lässt sich sicherstellen, dass der HC-SR04 Ultraschallsensor genaue und zuverlässige Entfernungsdaten liefert, die eine effektive Unterstützung für die Navigation des Roboters bieten.

3.3 Bewegungssteuerung

3.3.1 Grundprinzip

Laut der Bedienungsanleitung des Arlo-Roboters besteht folgender Zusammenhang zwischen der Dauer des High-Pegels (kurz HP) des PWM-Signals und der Drehzahl sowie der Drehrichtung des Motors:

- Wenn die Dauer des HPs zwischen 1ms und 1,5ms liegt, dreht sich der Motor vorwärts, und die Drehzahl nimmt mit zunehmender Dauer des HPs ab. Bei einer Dauer von 1ms erreicht der Motor seine maximale Vorwärtsgeschwindigkeit.
- Wenn die Dauer des HPs 1,5ms beträgt, steht der Motor still.
- Wenn die Dauer des HPs zwischen 1,5ms und 2ms liegt, dreht sich der Motor rückwärts, und die Drehzahl nimmt mit zunehmender Dauer des HPs zu.
- Wenn die Dauer des HPs 2ms erreicht, erreicht der Motor seine maximale Rückwärtsgeschwindigkeit.

Im Rahmen der Projektforschung wurde die PWM-Frequenz auf 250 Hz eingestellt. Um sicherzustellen, dass sich der Motor vorwärts dreht, muss die Dauer des HPs t′ folgenden Bereich erfüllen:

$$\frac{t'_{min}}{250Hz} < t' < \frac{t'_{max}}{250Hz} \tag{3.7}$$

$$\frac{1ms}{250Hz} < t' < \frac{1.5ms}{250Hz} \tag{3.8}$$

$$0.25 < t < 0.375 \tag{3.9}$$

In ähnlicher Weise, um eine Rückwärtsdrehung des Motors zu erreichen, muss die Dauer des High-Pegels t′ unter Verwendung der Formel (3.7) in folgendem Bereich liegen:

$$\frac{1.5ms}{250Hz} < t' < \frac{2ms}{250Hz} \tag{3.10}$$

$$0.375 < t' < 0.5 \tag{3.11}$$

3.3.2 Rotation

Bei der Steuerung des Verhaltens des Roboters zeigt dieser eine Rotation um einen bestimmten Mittelpunkt, wenn die PWM-Eingaben für die linke und rechte Rad unterschiedlich sind. Zum Beispiel, wenn das linke Rad ein PWM-TV von 0,25 und das rechte Rad ein PWM-TV von 0,4 erhält, können gemäß den Gleichungen (4.9) und (4.12) folgende Drehzahlen abgeleitet werden:

$$v_l = 1.68r/s \tag{3.12}$$

$$v_r = -0.27r/s \tag{3.13}$$

Basierend auf den vorliegenden Daten wird eine grafische Darstellung der Bewegungsanalyse erstellt.

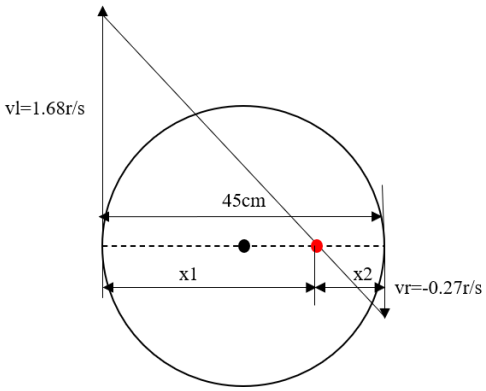


Abbildung 3.3 Kinematische Analyse

Aus Abbildung 3.3 ist ersichtlich, dass sich der Arlo-Roboter um den im Bild gezeigten roten Punkt dreht, wenn sich die beiden Räder mit Drehzahlen von 1,68 r/s und -0,27 r/s bewegen. Die Position des roten Punktes kann mithilfe des Prinzips der Ähnlichkeit von Dreiecken ermittelt werden. Es ist bekannt, dass der Durchmesser des Arlo-Roboters 45cm beträgt.

$$\frac{x_1}{x_2} = \frac{vl}{vr} = \frac{1.68}{0.27} \tag{3.14}$$

$$x_1 + x_2 = 45cm \tag{3.15}$$

$$x_1 = 38.77; x_2 = 6.23 \tag{3.16}$$

In diesem Beispiel befindet sich das Rotationszentrum des Roboters 38,77cm rechts vom linken Rad, und der Roboter dreht sich um diesen Mittelpunkt. Im formellen Experiment wurde zur Vereinfachung der Berechnungen für das linke und rechte Rad eine gleiche, jedoch entgegengesetzte Drehzahl eingestellt, um sicherzustellen, dass das Rotationszentrum im Mittelpunkt des Roboters liegt.

3.4 PID-Regelung

Der PID-Regler ist ein weit verbreiteter klassischer Regelungsalgorithmus in automatischen Steuerungssystemen. Sein Name leitet sich von den drei wesentlichen Regelparametern ab: Proportional (P), Integral (I) und Differenzial (D). Durch die koordinierte Arbeit dieser drei Parameter kann der PID-Regler die dynamischen Eigenschaften eines Systems optimieren. In diesem Kapitel wird das Funktionsprinzip des PID-Reglers sowie seine Anwendung in Regelungssystemen ausführlich erläutert.

Zunächst einmal wird die proportionale Regelung (P) durch die Anpassung der Steuergröße in Abhängigkeit von der aktuellen Abweichung (also der Differenz zwischen dem Sollwert und dem Istwert) realisiert. Die Hauptfunktion der Proportionalregelung besteht darin, schnell auf

Änderungen der Abweichung zu reagieren und somit die Abweichung des Systems zu reduzieren. Die Ausgangsformel für die Proportionalregelung lautet:

$$P_{out} = K_p * \mathrm{e(t)} \tag{3.17}$$

Dabei ist K_p der Proportionalverstärkung, und e(t) die Abweichung zum aktuellen Zeitpunkt. Obwohl die proportionale Regelung die Abweichung schnell reduzieren kann, führt ihre alleinige Anwendung oft zu einer bleibenden statischen Abweichung (Steady-State Error).

Zweitens zielt die Integralregelung (I) darauf ab, durch die Anpassung der Steuergröße basierend auf der über die Zeit akkumulierten Abweichung die statische Abweichung zu beseitigen. Die Integralregelung summiert die vergangenen Abweichungen und passt die Steuergröße so an, dass das System den Sollwert erreicht und dort bleibt. Die Ausgangsformel für die Integralregelung lautet:

$$I_{out} = K_i * \int_0^t e(\tau) d\tau \tag{3.18}$$

Dabei ist K_i der Integralverstärkung, und $\int_0^t e(\tau) d\tau$ stellt das Integral der Abweichung von der Zeit null bis zum aktuellen Zeitpunkt dar. Obwohl die Integralregelung dazu beitragen kann, die statische Abweichung zu beseitigen, kann ihre übermäßige Anwendung zu einer Übersteuerung und Schwingungen im System führen.

Schließlich zielt die Differentialregelung (D) darauf ab, durch Berechnung der Änderungsrate der Abweichung die zukünftigen Abweichungstendenzen vorherzusagen und so vorausschauende Anpassungen an der Steuergröße vorzunehmen. Der Hauptzweck der Differentialregelung besteht darin, die Stabilität der Systemreaktion zu verbessern und Übersteuerungen sowie Schwingungen zu minimieren.

Die Ausgangsformel für die Differentialregelung lautet:

$$D_{out} = K_d * \frac{de(t)}{dt} \tag{3.19}$$

Dabei ist K_d der Differentialverstärkung, und $\frac{de(t)}{dt}$ stellt die Änderungsrate der Abweichung dar. Die Differentialregelung trägt wesentlich zur Verbesserung der Systemstabilität bei, ist jedoch empfindlich gegenüber Rauschen.

Der Gesamtausgang des PID-Reglers ergibt sich aus der Summe der drei Komponenten und lässt sich durch folgende Gleichung darstellen:

$$PID_{out} = K_p * \mathrm{e(t)} + K_i * \int_0^t e(\tau) d\tau + K_d * \frac{de(t)}{dt} \tag{3.20}$$

Die angemessene Einstellung der Parameter K_p, K_i und K_d in dieser Gleichung ist entscheidend für die Leistung des PID-Reglers. Die Parametereinstellung erfolgt in der Regel durch Experimente und Erfahrungswerte, um die optimale Steuerung des Systems zu erreichen[9][10].

3.4.1 Winkel-PID-Regelung

Das System verwendet ein Gyroskop zur Winkelsteuerung. Dabei wird ein PID-Algorithmus eingesetzt, um den Roboter auf dem Zielwinkel zu halten. Das entsprechende Programm ist in der folgenden Abbildung dargestellt.

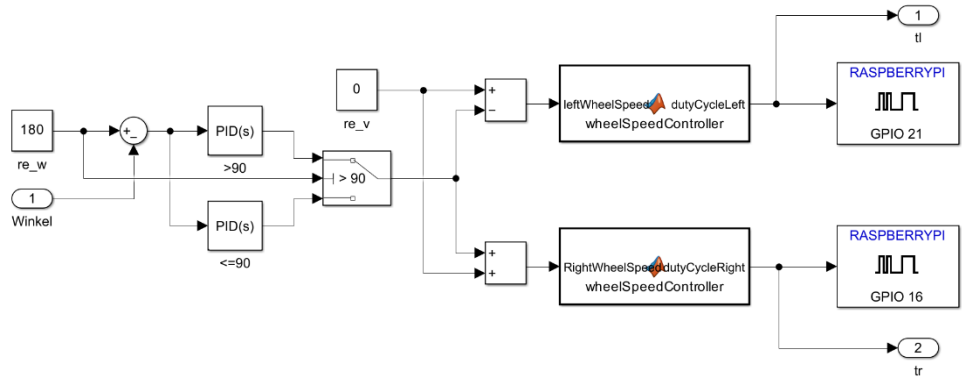


Abbildung 3.4 Winkel-PID-Algorithmus

Wie in der Abbildung dargestellt, wurde im Experiment ein PID-Regler für die Winkelsteuerung eingesetzt. Dabei wurden für eine Sollwinkeleinstellung von über 90° die folgenden PID-Parameter verwendet: Proportionalverstärkung K_p = 0,05, Integralverstärkung K_i = 0,0003 und Differenzialverstärkung K_d = 0,006. Bei einer Sollwinkeleinstellung von 90° oder weniger wurden die Parameter wie folgt angepasst: Proportionalverstärkung K_p = 0,034, Integralverstärkung K_i = 0 und Differenzialverstärkung K_d = 0,00015.

Durch diese Einstellung kann der Roboter präzise auf den Zielwinkel ausgerichtet werden. Wie in Abbildung 14 dargestellt, beträgt der Referenzwinkel 180°. Nach Ausführung des Programms dreht sich der Roboter um 180° und behält seine Ausrichtung bei, selbst unter äußeren Einflüssen.

3.4.2 Distanz-PID-Regelung

Ähnlich wie bei der Winkel-PID-Regelung dient der Abstand des Roboters zur Wand als Referenzparameter, um das Verhalten des Roboters zu steuern. Ist der Abstand des Roboters zur Wand geringer als 1,55m, reduziert der Algorithmus die Geschwindigkeit des linken Rads und erhöht die des rechten Rads, wodurch der Roboter nach links abweicht. Umgekehrt, wenn der Abstand größer als 1,55m ist, erhöht der Algorithmus die Geschwindigkeit des linken Rads und verringert die des rechten Rads, sodass der Roboter nach rechts abweicht und sich der vorgegebenen Trajektorie annähert.

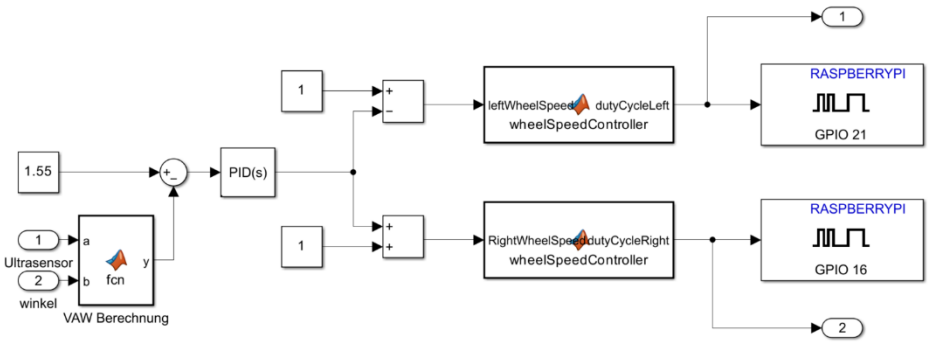


Abbildung 3.5 Logik des Distanz-PID-Berechnung

Die Modulstruktur entspricht grundsätzlich derjenigen der Winkel-PID-Regelung. Der Differenzwert der Distanz wird in den PID-Regler eingespeist, der daraufhin basierend auf den PID-Ausgaben die Drehgeschwindigkeit der linken und rechten Räder des Roboters anpasst. Anschließend berechnet eine MATLAB-Funktion die jeweiligen TVs für die linken und rechten Räder.

Wichtig ist dabei, dass der Differenzwert der Distanz nicht direkt aus den vom Ultraschall-Entfernungsmesser erfassten Daten und dem Referenzabstand gebildet werden kann. Da die gemessenen Daten senkrecht zur aktuellen Fahrtrichtung des Roboters erfasst werden, müssen diese Messwerte mithilfe trigonometrischer Funktionen auf die Senkrechte zur Wand projiziert werden, bevor sie in die weitere Berechnung einfließen.

4 Vorversuch

Um eine präzise Steuerung der Geschwindigkeit und des Wendewinkels des Arlo-Roboters zu ermöglichen, müssen vor Beginn der Forschungsarbeiten detaillierte Parameterkalibrierungsexperimente durchgeführt werden. Erst nachdem durch diese Experimente genaue Daten gewonnen wurden, ist es möglich, den Roboter basierend auf den Kalibrierungsergebnissen präzise zu steuern. Die Kalibrierungsexperimente umfassen hauptsächlich Versuche zur Drehzahl, zum Drehwinkel sowie zur Messung des Winkels durch die experimentellen Geräte.

4.1 Kalibrierungsexperiment zur Drehzahl und zum Tastverhältnis

Im Experiment wird der Arlo-Roboter auf einer Schaumstoffplatte angehoben, sodass die beiden Räder in der Luft hängen (siehe Anlage 17), um die Parameterkalibrierung durchzuführen. Mit Hilfe der Simulink-Software werden direkt PWM-Signale an die GPIO-Pins 21 und 16 des RPi gesendet, um das Drehmoment für die linken und rechten Räder des Roboters zu erzeugen. In der weiteren Programmgestaltung wird auch die Ausgabe der Radgeschwindigkeit und der Reifengeschwindigkeit integriert, um die notwendigen Daten für weitergehende Untersuchungen zur Bewegungssteuerung zu liefern.

- **Experimentbeschreibung**

Folgendes ist das im Experiment verwendete PWM-Eingabeprogramm:

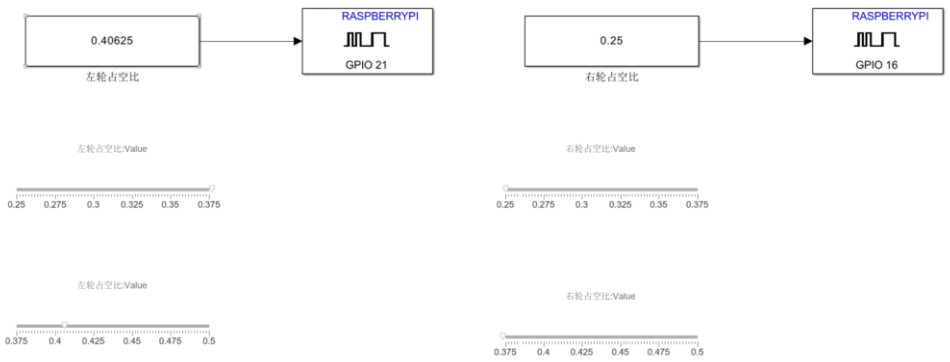


Abbildung 4.1 Manuelle PWM-Eingabe

Die Änderung des PWM-TVses kann entweder durch manuelle Eingabe oder durch die Steuerung eines Schiebereglers erfolgen[11].

- **Experimentergebnisse und Datenanalyse**

In der Anlage 12 sind die in Simulink gemessenen Drehzahldaten von 10 Versuchsreihen aufgeführt.

Auf Basis der Daten wird die A-Klasse-Ungewissheit für die Vertrauensniveaus P=0,68 P=0,95 und P=0,99 berechnet. Bei einer Anzahl von 10 Messwerten sind die t-Werte für die entsprechenden Vertrauensniveaus wie folgt gegeben: $t_{0,68}$=1,06, $t_{0,95}$=2,26 und $t_{0,99}$=3,25.

$$\bar{x} = \frac{1}{n} * \sum_{i=1}^{n} x_i \tag{4.1}$$

$$\sigma_x = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n * (n-1)}} \tag{4.2}$$

$$u_A = \frac{\sigma_x}{\sqrt{n}} \tag{4.3}$$

$$U_A = t_p * u_A \tag{4.4}$$

Basierend auf der obigen Formel ergeben sich die Parameter in der Anlage 13

Nach den Durchschnittswerten der Drehzahl kann die funktionale Beziehung zwischen der Drehzahl des Reifens im Vorwärts- und Rückwärtslauf und dem TV berechnet werden.

Aus Anlage 14 können die folgenden Tabellen aufgestellt werden (Gleichungen (4.5)-(4.8) dienen zur Berechnung des TVses anhand der Drehzahl, Gleichungen (4.9)-(4.12) zur Berechnung der Drehzahl anhand des TVses):

VLR	$y = -0.072x + 0.3727$	(4.5)
RLR	$y = -0.0746x + 0.3778$	(4.6)
VRR	$y = -0.072x + 0.3726$	(4.7)
RRR	$y = -0.0748x + 0.3776$	(4.8)
VLR	$y = -13.867x + 5.1689$	(4.9)
RLR	$y = -13.364x + 5.0465$	(4.10)
VRR	$y = -13.865x + 5.1669$	(4.11)
RRR	$y = -13.334x + 5.0321$	(4.12)

Die Werte der Unsicherheit UA bei P=0.68, P=0.95 und P=0.99 sind in einem Streudiagramm darzustellen:

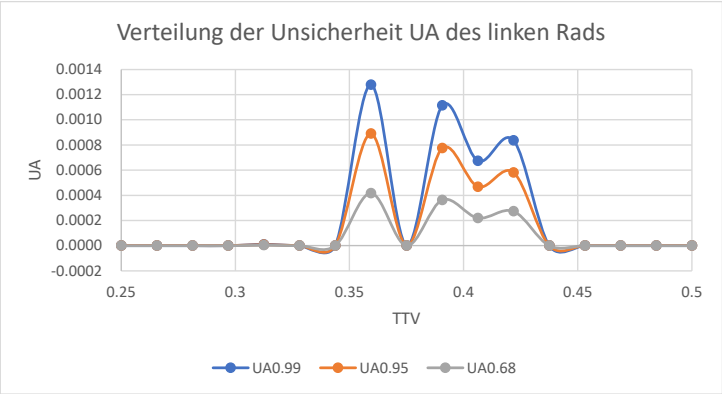


Abbildung 4.2 Verteilung der UA am linken Rad

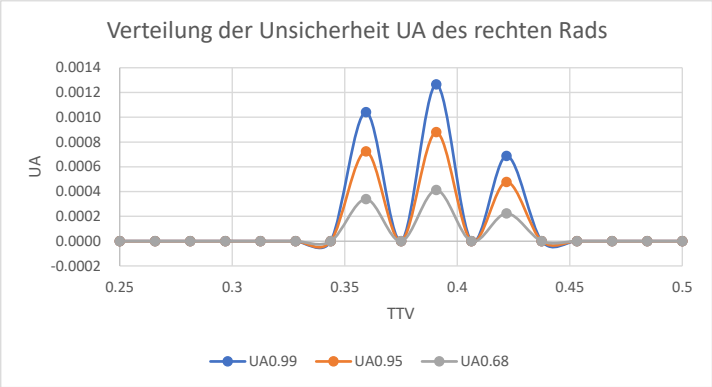


Abbildung 4.3 Verteilung der UA am rechten Rad

Basierend auf den experimentellen Daten konzentrieren sich die von null abweichenden Werte der Unsicherheit UA hauptsächlich um einen PWM-TV von 0,375. Dies deutet darauf hin, dass bei niedrigen Drehzahlen des Arlo-Roboter-Motors eine gewisse Unsicherheit bei den von Simulink erfassten und berechneten Daten bestehen könnte. Allerdings beträgt der gemessene Wert selbst bei höheren P-Werten nur 0,0013, was darauf hinweist, dass die Zuverlässigkeit der Experimentergebnisse hoch ist. Daher ist es eine sinnvolle Entscheidung, die experimentell ermittelte durchschnittliche Drehzahl als Kalibrierungsparameter zu verwenden[12][13].

4.2 Statisches Kalibrierungsexperiment für den Drehwinkel

Um die Änderung des Drehwinkels klar darzustellen, wurde in diesem Experiment eine Bodenmessmethode verwendet. Der Roboter wurde auf den Testpunkt gesetzt, sodass die rote Markierung auf der Roboterplattform mit der 90°-Markierung des Testfeldes übereinstimmt. Anschließend wurde ein geeigneter PWM-TV eingegeben. Mithilfe des in einem iPhone 15 integrierten Kompasses wurde der Drehwinkel des Roboters über einen bestimmten Zeitraum gemessen (siehe Anlage 18).

- **Experimentbeschreibung**

Abbildung 4.4 Skizze des statischen Kalibrierungsexperiments

Die Kompassfunktion des Smartphones wird aktiviert und innerhalb des roten Rahmens auf dem Roboter platziert. Um die Rotationswinkel des Roboters präzise steuern zu können, wurde die Drehgeschwindigkeit des Roboters im Experiment auf ein Minimum reduziert. Die spezifischen Einstellungsparameter sind wie folgt:

in Uhrzeigersinn	PWM	gegen Uhrzeigersinn	PWM
linkes Rad	0.355	linkes Rad	0.395
rechtes Rad	0.395	rechtes Rad	0.355

Tabelle 4.1 Einstellung des PWM für die Drehbewegung

Unter Verwendung der oben genannten Parameter können die Drehzahlen der linken und rechten Räder des Roboters gemäß den Gleichungen (4.5) bis (4.8) berechnet werden:

in Uhrzeigersinn	Drehzahl[r/s]	gegen Uhrzeigersinn	Drehzahl[r/s]
linkes Rad	0.231	linkes Rad	0.206
rechtes Rad	0.210	rechtes Rad	0.231

Tabelle 4.2 Drehzahlberechnung

- **Experimentergebnisse und Datenanalyse**

Nach der Eingabe der entsprechenden Parameter in Simulink und der Durchführung des Experiments wurden die Daten in der Anlage 15 erhalten.

Mithilfe der Gleichungen (4.1)-(4.4) werden der Mittelwert $\bar{x}$, die Standardabweichung der Messreihe σ_x, die Standardabweichung des arithmetischen Mittels, uA und die Typ-A-Unsicherheit UA berechnet. (siehe Anlage 16)

Basierend auf dem ermittelten Mittelwert kann das folgende Diagramm erstellt werden:

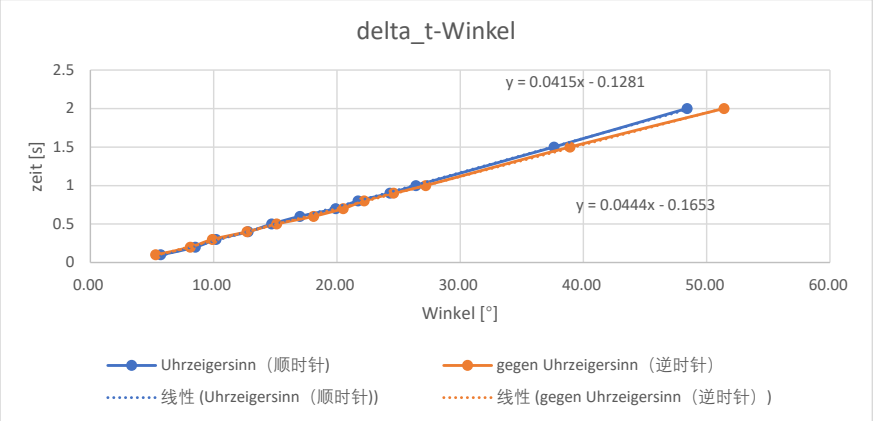


Abbildung 4.5 Beziehung zwischen Steuerungszeit und Drehwinkel im statischen Kalibrierungsexperiment

Wie in der Abbildung gezeigt, ist der Drehwinkel annähernd proportional zur benötigten Zeit. Mit zunehmendem Drehwinkel erhöht sich die benötigte Zeit proportional. Daraus ergibt sich die folgende Gleichung zur Berechnung der benötigten Zeit:

In Uhrzeigersinn $$y = 0.0415 * x - 0.1281 \quad (4.13)$$

Gg. Uhrzeigersinn $$y = 0.0444 * x - 0.1653 \quad (4.14)$$

Bei einer Konfidenzwahrscheinlichkeit von 0,99 liegt die gemessene Typ-A-Unsicherheit der einzelnen Drehwinkel zwischen 0,10 und 0,32. Dieser Bereich übersteigt insgesamt die Typ-A-Unsicherheit des Kalibrierungsexperiments für das TV der Drehzahl. Angesichts der Tatsache, dass die Genauigkeit des Kompasswerkzeugs im iPhone 15 bei 1° liegt und somit eine erhebliche Fehlerwahrscheinlichkeit besteht, kann diese Unsicherheit jedoch als akzeptabel angesehen werden.

4.3 Dynamisches Kalibrierungsexperiment für den Drehwinkel

Die in Experiment 3.3 gewonnenen Daten basieren auf der Annahme, dass die Anfangsgeschwindigkeit v0 beider Räder des Roboters gleich null ist. In der Praxis muss der Roboter jedoch häufig während der Vorwärtsbewegung Richtungsänderungen vornehmen. Wie in Anlage 17 dargestellt, verfügt der Arlo-Roboter neben den beiden Hauptantriebsrädern über je ein zusätzliches Stützrad vorne und hinten, um die Stabilität während der Fahrt zu gewährleisten. Diese Stützräder können jedoch beim Drehen des Roboters einen gewissen Einfluss ausüben. Daher ist es erforderlich, für den Roboter in dynamischen Situationen eine zusätzliche Winkelkalibrierung durchzuführen.

- **Experimentbeschreibung**

Um sicherzustellen, dass das Smartphone während des Experiments nicht durch die Bewegung des Roboters beeinträchtigt wird und damit die Messergebnisse verfälscht, wurde das Smartphone an seinem Platz zusätzlich mit Klebeband fixiert (siehe Anlage 19). Dies gewährleistet, dass das Messgerät während der Messung stabil bleibt und somit die Genauigkeit der Experimentergebnisse erhöht wird.

Der Programmablaufplan sieht wie folgt aus:

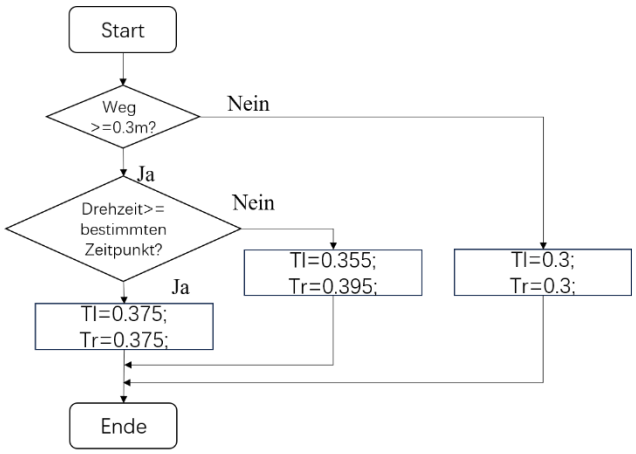


Abbildung 4.6 Programmablaufplan des dynamischen Kalibrierungsexperiments für den Drehwinkel

Im Experiment werden zunächst beide Räder des Roboters mit derselben Drehzahl von 1 r/s betrieben, um eine Strecke von 0,3m zurückzulegen. Während dieses Vorgangs bleiben die beiden Stützräder parallel zur Bewegungsrichtung des Roboters. Anschließend wird dem Roboter ein Rotationsbefehl erteilt, und nach einer bestimmten Zeitspanne wird die Rotation gestoppt. Der dabei erreichte Drehwinkel wird abgelesen und aufgezeichnet. Jeder Zeitabschnitt wird 10Mal wiederholt, um die Zuverlässigkeit und Genauigkeit der Daten sicherzustellen.

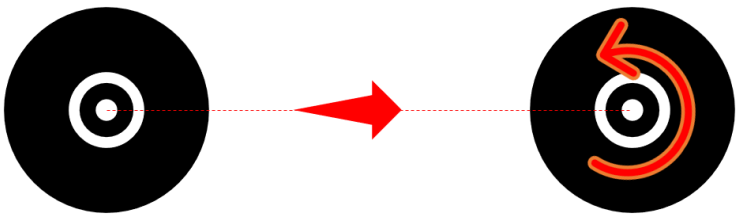


Abbildung 4.7 Skizze des dynamischen Kalibrierungsexperiments

Das Experiment verwendet ein Zustandsautomat-Steuermodul (Statflow)[14]. Dieses Modul enthält den Hauptzyklus des Versuchsablaufs, um die Aktionen des Roboters in den verschiedenen Zuständen präzise zu steuern. Durch diese Verbesserung kann das Experiment das Drehverhalten des Roboters in unterschiedlichen Zeitabschnitten besser simulieren und messen, was genauere Daten für die weitere Analyse liefert.

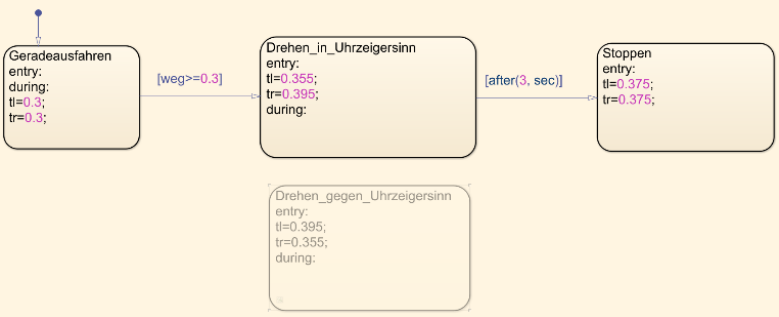


Abbildung 4.8 Interne Logikstruktur des Zustandsautomaten

Für die drei Betriebszustände des Roboters – Geradeausfahren, Drehen und Stoppen – werden im Zustandsautomaten die folgenden PWM-Eingaben vorgenommen:

In Uhrzeigersinn	Geradeausfahren	Drehen	Stoppen
TV des linken Rads	0.3	0.355	0.375
TV des rechten Rads	0.3	0.395	0.375

Tabelle 4.3 PWM-Eingabeeinstellungen in Uhrzeigersinn

Gegen Uhrzeigersinn	Geradeausfahrt	Drehen	Stoppen
TV des linken Rads	0.3	0.395	0.375
TV des rechten Rads	0.3	0.355	0.375

Tabelle 4.4 PWM-Eingabeeinstellungen gegen Uhrzeigersinn

Im Steuerungsalgorithmus erfolgt der Übergang des Roboters vom Geradeausfahren zum Drehen sowie vom Drehen zum Anhalten durch die Überwachung der zurückgelegten Strecke und der Programmlaufzeit. Wenn der Roboter eine Strecke von 0,3m zurückgelegt hat, wechselt der Zustand des Roboters von „Anfang“ zu „Lenken_in_Uhrzeigersinn“, und der Roboter wechselt vom Geradeausfahren in den Zustand einer Rechtsdrehung (im Uhrzeigersinn). Nach einer bestimmten Rotationszeit stoppt der Roboter seine Drehbewegung. Im Zustandsautomaten wurde außerdem der Zustand „Lenken_gegen_Uhrzeigersinn“ eingeführt, um den Anforderungen für Experimente mit Linksdrehungen (gegen den Uhrzeigersinn) gerecht zu werden.

- **Experimentergebnisse und Datenanalyse**

In der Anlage 20 sind die im Experiment gemessenen Daten aufgeführt.

Nach der Datenverarbeitung wurde die Tabelle wie Anlage 21 erstellt:

Das Diagramm des Drehwinkels des dynamischen Roboters in Abhängigkeit von der Rotationszeit wurde basierend auf den Mittelwerten erstellt:

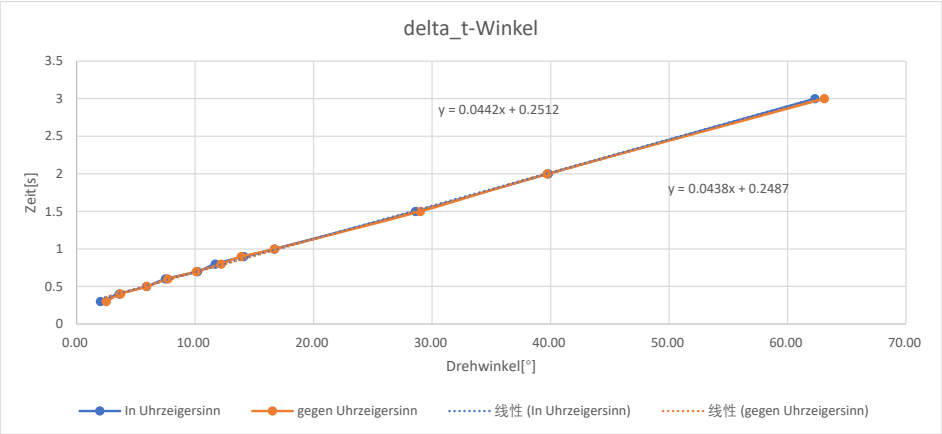


Abbildung 4.9 Beziehung zwischen Steuerungszeit und Drehwinkel im statischen Kalibrierungsexperiment

Aus der obigen Abbildung lässt sich erkennen, dass die Beziehung zwischen der Rotationszeit und dem Drehwinkel des Roboters bei der Drehung im Uhrzeigersinn und gegen den Uhrzeigersinn ähnlich ist. Diese Beziehungen können durch die folgenden Funktionsgleichungen beschrieben werden:

Im Uhrzeigersinn $$y = 0.0442 * x + 0.2512 \quad (4.15)$$

Gg. Uhrzeigersinn $$y = 0.0438 * x + 0.2487 \quad (4.16)$$

Die Versuchsergebnisse zeigen, dass die Winkel-Zeit-Beziehung des Roboters bei der Drehung im Uhrzeigersinn und gegen den Uhrzeigersinn annähernd linear ist und die linearen Regressionsgleichungen für beide Drehrichtungen nahezu identisch sind. Dies bedeutet, dass die Bewegungseigenschaften des Roboters in beiden Drehrichtungen weitgehend symmetrisch sind, was auf eine gute Rotationskonsistenz hinweist.

Diese Konsistenz ist für die präzise Steuerung und Navigation des Roboters von großer Bedeutung, da sie sicherstellt, dass das erwartete Verhalten des Roboters bei der Ausführung von Drehmanövern mit der tatsächlichen Leistung übereinstimmt. Dadurch wird die Zuverlässigkeit und die Steuerungsgenauigkeit des Roboters erhöht.

4.4 Gyroskop-Winkelkalibrierungsexperiment

In diesem Experiment wurde das im RPi integrierte „Sense HAT“-Gyroskop einer Kalibrierung unterzogen. Die Untersuchungen zeigten, dass die Messgenauigkeit des Sense HAT-Gyroskops begrenzt ist und signifikante Abweichungen zwischen den aufgezeichneten und den tatsächlichen Drehwinkeln bestehen. Ziel des Experiments ist es daher, eine Kalibrierung der gemessenen Winkelwerte vorzunehmen, um die Genauigkeit der Winkelbestimmung zu verbessern und somit die Präzision der Spurverfolgung des Arlo-Roboters zu optimieren.

- **Experimentbeschreibung**

Die spezifische Konfiguration des Experiments ist wie folgt:

Integriertes Gyroskop des Roboters: Dient zur Aufzeichnung der gemessenen Winkel.

Sensor-Plugin in MATLAB Online: Wird verwendet, um den tatsächlich vom Roboter zurückgelegten Winkel zu messen.

Vorbereitende Maßnahmen:

1. Es wird sichergestellt, dass sowohl das im Roboter integrierte Gyroskop als auch das Sensor-Plugin in MATLAB Online ordnungsgemäß funktionieren.
2. Die Inbetriebnahme des Roboters

Datenerfassung:

1. Der Roboter wird gemäß dem festgelegten Programmablauf in Bewegung gesetzt und führt eine Rotationsbewegung aus.
2. Das im Roboter integrierte Gyroskop zeichnet während der Rotationsbewegung die entsprechenden Messdaten des Drehwinkels auf.
3. Parallel dazu wird der tatsächliche Drehwinkel des Roboters mithilfe des in MATLAB Online integrierten Sensor-Plugins in Echtzeit erfasst.

Datenanalyse und Kalibrierung:

1. Die vom Gyroskop erfassten Messwinkel werden mit den tatsächlichen, durch das Sensor-Plugin ermittelten Winkeln verglichen.
2. Durch Datenanpassung und -analyse wird ein Kalibrierungsmodell entwickelt, das die Beziehung zwischen den gemessenen Winkeln und den tatsächlichen Winkeln präzise abbildet.

In den folgenden Abbildungen sind das im Experiment verwendete Programm sowie das zugehörige Programmablaufplan dargestellt.

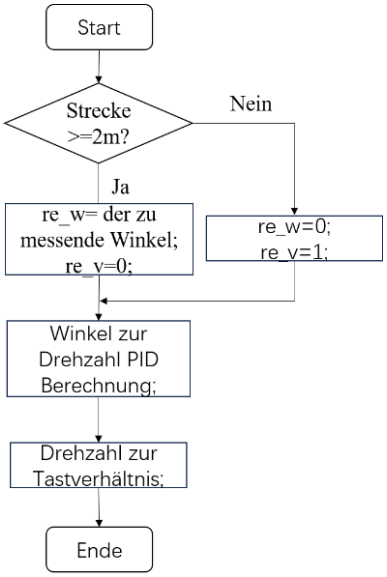


Abbildung 4.10 Programmablaufplan des Winkelkalibrierungsexperiments

Im Rahmen dieses Versuchs ist der entscheidende Parameter die vom Roboter zurückgelegte Strecke. Solange diese Strecke unter 2m bleibt, wird die Referenzdrehzahl des Winkel-PID-Reglers auf 1r/s und der Referenzwinkel auf 0° festgelegt. Der PID-Regler passt die Drehgeschwindigkeiten der linken und rechten Räder entsprechend an, um die Fahrtrichtung des Roboters stabil zu halten. Sobald die Strecke von 2m erreicht ist, wird die Referenzdrehzahl auf 0 r/s reduziert, während der Referenzwinkel je nach Versuchsgruppe variiert und in Schritten von 30° von 30° bis 300° eingestellt wird.

Anschließend wird die im Kalibrierungsexperiment ermittelte Beziehung zwischen Drehzahl und TV genutzt, um die Drehzahl in das entsprechende TV umzuwandeln, was eine präzise Steuerung der Motoren ermöglicht. Die untenstehende Abbildung zeigt das spezifische Programm, das im Experiment verwendet wurde.

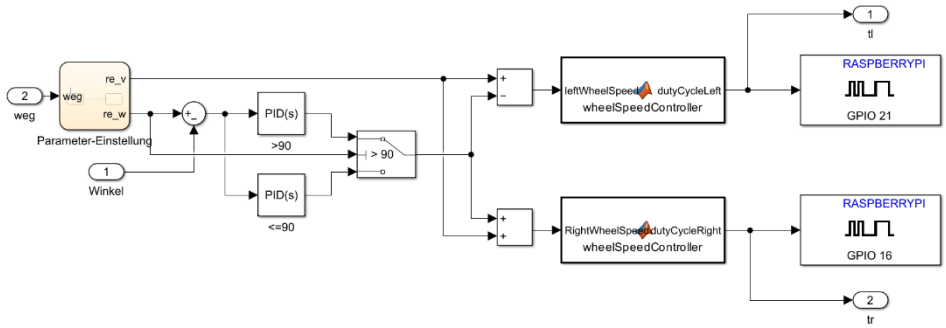


Abbildung 4.11 Simulink-Programm zur Kalibrierung des Gyroskopwinkels

Wie in Abbildung 4.11 dargestellt, erfolgt die Parametrierung für die verschiedenen Phasen des Roboterbetriebs über die Zustandsautomat „Parameter-Einstellung“. Es ist dabei von besonderer Bedeutung, dass der PID-Regler für Drehwinkel, die größer als 90° sind, sowie für solche, die 90° oder weniger betragen, unterschiedliche Werte für die Proportional-, Integral- und Differentialverstärkungen (K_p, K_i, K_d) aufweist. Für Drehwinkel über 90° wurden die Verstärkungen wie folgt festgelegt: K_p = 0,05, K_i = 0,0003, K_d = 0,006. Im Gegensatz dazu

lauten die Werte für Drehwinkel bis einschließlich 90°: K_p = 0,034, K_i = 0, K_d = 0,00015. In beiden Fällen wurde der Filterkoeffizient auf 100 gesetzt, um den Einfluss von Hochfrequenzrauschen auf das System zu minimieren.

- **Analyse der Versuchsergebnisse**

Nach Abschluss des Experiments wurden die vom Kompassprogramm berechneten Winkelwerte sowie die von MATLAB Online erfassten tatsächlichen Drehwinkel dokumentiert. Die Beziehung zwischen diesen beiden Datensätzen wird anschließend in Form eines Streudiagramms veranschaulicht.

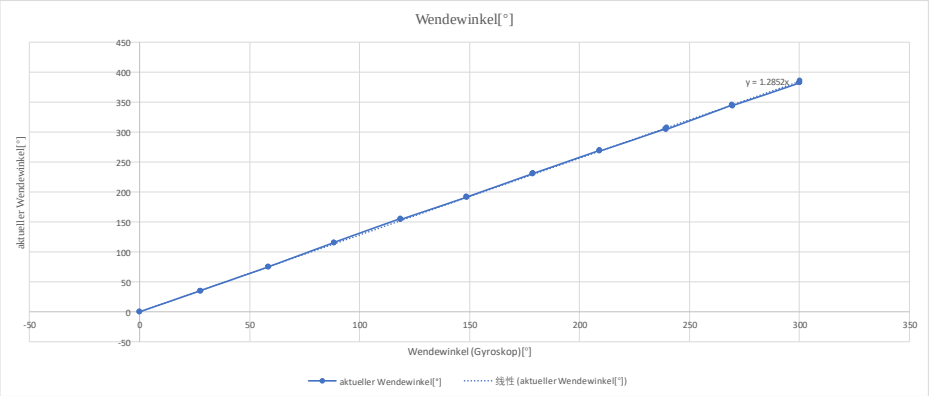


Abbildung 4.12 Experimentergebnisse des Winkelkalibrierungsexperiments

Basierend auf den Experimentergebnissen lässt sich feststellen, dass der tatsächliche Drehwinkel des Roboters proportional zum gemessenen Winkel ist, wobei der Proportionalitätsfaktor 1,2852 beträgt. Da der tatsächliche Drehwinkel bei einem gemessenen Winkel von 0 ebenfalls 0 ist, beträgt der Achsenabschnitt null. Folglich ergibt sich die folgende Beziehung zwischen dem tatsächlichen Drehwinkel und dem gemessenen Winkel:

$$W_{out} = W_{in} * 1.2852 \tag{4.17}$$

5 Umsetzung

Wie bereits dargelegt, ist ein zentrales Ziel dieses Projekts die Implementierung einer zuverlässigen Spurverfolgung für den Roboter. Dieses Ziel wird in zwei Hauptmodule unterteilt: den grundlegenden Trajektorienalgorithmus und den Optimierungsalgorithmus für die Robotertrajektorie. Der grundlegende Trajektorienalgorithmus bildet das Herzstück des Projekts und stellt sicher, dass der Roboter in der Lage ist, einer vorgegebenen Route grob zu folgen. Der Optimierungsalgorithmus hingegen dient dazu, die Genauigkeit und Effizienz der Spurverfolgung weiter zu steigern, wodurch die Kernfunktionalität des Systems insgesamt optimiert wird.

5.1 Der grundlegende Trajektorienalgorithmus

In dieser Untersuchung wurden zwar keine spezifischen Trajektorien vorgegeben, doch lassen sich anhand der beiden Szenarien „Straßenüberquerung“ (kurz SÜ) und „Gästeempfang“ (kurz GE) zwei mögliche Routen ableiten.

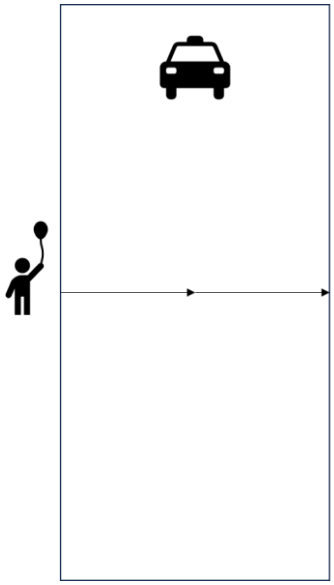


Abbildung 5.1 Straßenüberquerung

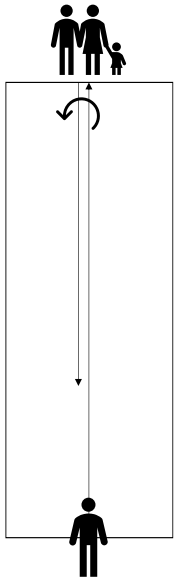


Abbildung 5.2 Gästeempfang

Ausgehend von der obigen Abbildung lässt sich den Hauptablaufplan der beiden Routen ableiten.

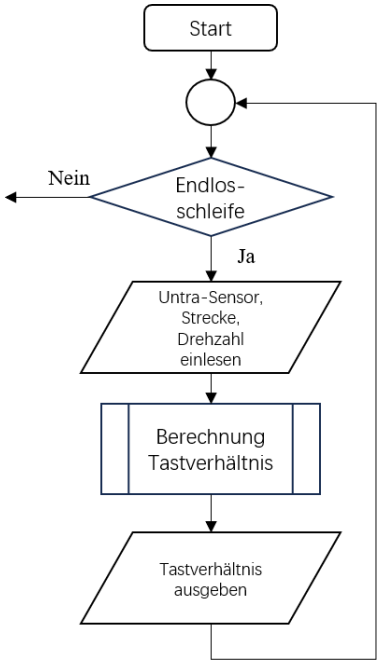


Abbildung 5.3 Hauptablaufplan [15]

Nach dem Programmstart werden zunächst die Daten für Abstand, Strecke und Geschwindigkeit erfasst. Diese Daten dienen anschließend zur Berechnung des TVses, das letztendlich ausgegeben wird.

5.1.1 „SÜ“-Route

Die erste Route besteht darin, dass der Roboter von einem Ende der Straße zum anderen fährt, wobei er einer geraden Linie mit einer Länge von 5m folgt. Diese Route wird durch den folgenden Programmablaufplan unterstützt.

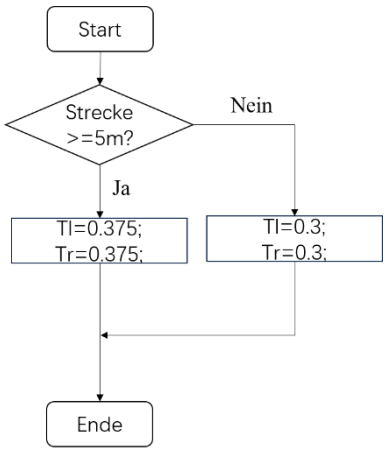


Abbildung 5.4 „SÜ“-Route Programmablaufplan

Für die SÜ wird basierend auf der erfassten Strecke das PWM-Signal zur Ansteuerung der linken und rechten Räder des Roboters entsprechend angepasst. Konkret bedeutet dies,

- Im Fall einer zurückgelegten Strecke von weniger als 5m wird an beide Räder des Roboters ein PWM-Signal mit einem TV von 0,3 angelegt. Dies gewährleistet eine konstante Vorwärtsbewegung des Roboters mit einer Geschwindigkeit von 0,47m/s.
- Beträgt die zurückgelegte Strecke 5m oder mehr, so wird das TV für beide Räder des Roboters auf 0,375 eingestellt. Dadurch wird der Roboter zum Stillstand gebracht.

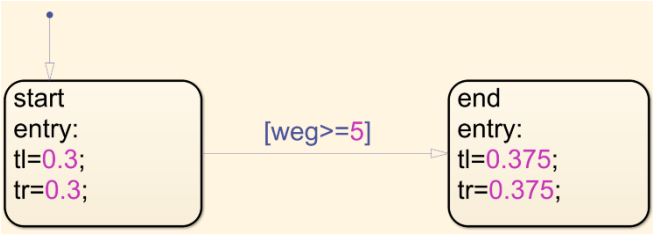


Abbildung 5.5 „SÜ“-Route Zustandsautomat

5.1.2 „GE“-Route

Die zweite Route sieht vor, dass der Roboter nach einem geraden Weg den Besucher zu seinem Sitzplatz führt. Konkret bedeutet dies, dass der Roboter zunächst 5m geradeaus fährt und anschließend eine Wendung vollzieht, um weitere 3m in entgegengesetzter Richtung zurückzulegen. Ein ähnliches logisches Diagramm wie bei der Route zur SÜ unterstützt diesen Bewegungsablauf

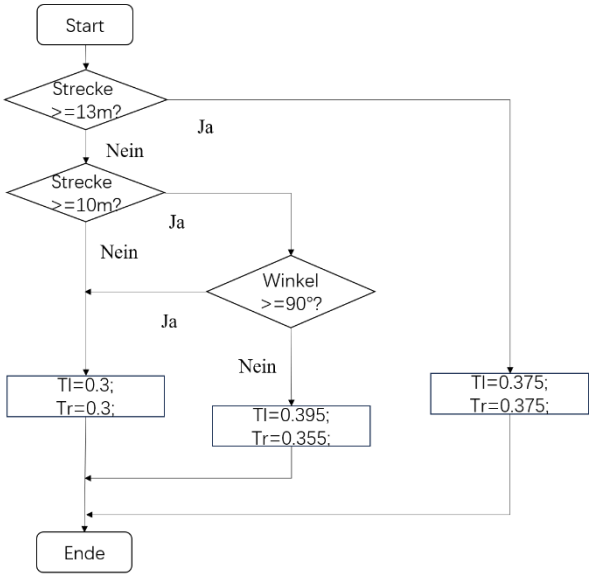


Abbildung 5.6 Programmablaufplan „GE“-Route

Für die Route zur Begleitung der Gäste wird das TV der PWM-Signale für die linken und rechten Räder des Roboters in Abhängigkeit von den erfassten Streckedaten angepasst. Die spezifischen Einstellungen sind wie folgt:

- Wenn die zurückgelegte Strecke weniger als 5m beträgt, wird das TV der PWM-Signale für beide Räder des Roboters auf 0,3 eingestellt.
- Wenn die zurückgelegte Strecke 5m oder mehr beträgt, wird das TV der PWM-Signale für die linken und rechten Räder des Roboters auf 0,395 bzw. 0,355 eingestellt, wodurch der Roboter beginnt, sich um sein symmetrisches Zentrum im Gegenuhrzeigersinn zu drehen.
- Nachdem der Roboter eine Drehung von 90° vollzogen hat, wird das TV der PWM-Signale für beide Räder wieder auf 0,3 zurückgesetzt.
- Wenn festgestellt wird, dass die zurückgelegte Strecke 8m oder mehr beträgt, wird das TV der PWM-Signale für die linken und rechten Räder auf 0,375 eingestellt, wodurch der Roboter zum Stillstand kommt.

Diese Methode stellt sicher, dass der Roboter bei unterschiedlichen Wegstrecken gemäß dem vorgesehenen Verhaltensmuster arbeitet und eine zuverlässige Empfangsroute bietet. Für diese Route wurden die folgenden zwei Zustandsautomaten-Designs entwickelt:

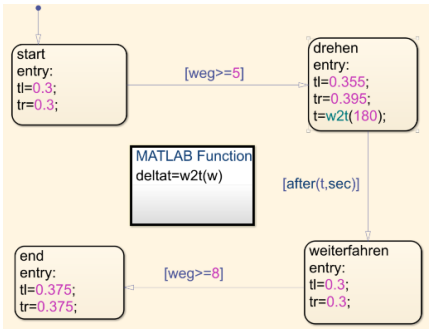


Abbildung 5.7 „GE“ Wendemethode 1

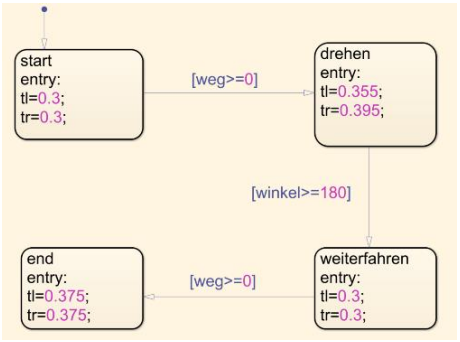


Abbildung 5.8 „GE“ Wendemethode 2

Im ersten Entwurf wird das Ergebnis des in Abschnitt 3.4 beschriebenen Kalibrierungsexperiments zur dynamischen Bestimmung des Drehwinkels und der Rotationszeit verwendet, um die für eine 180°-Drehung im Uhrzeigersinn erforderliche Zeit zu berechnen. Der Roboter wechselt nach einer bestimmten Rotationszeit vom Zustand 'drehen' in den Zustand 'weiterfahren' und setzt die Geradeausfahrt fort. Im zweiten Entwurf wird hingegen ein externer Kompassmodul verwendet, um den Winkel direkt zu messen. Der Zustandstransfer erfolgt hier, sobald der Roboter eine Drehung von 90° vollzogen hat.

5.1.3 Vergleich und Analyse der Wendemethode

In diesem Experiment wurden zwei verschiedene Ansätze verwendet, um den Roboter im Rahmen eines Richtungswechseltests zu steuern und den Drehwinkel zu messen, nachdem das Wendemanöver ausgelöst wurde. Ziel des Experiments war es, die Genauigkeit der Robotersteuerung mittels Gyroskop zu bewerten und zu entscheiden, ob das Gyroskop in zukünftigen Experimenten zur Unterstützung der Roboter-Navigation verwendet werden sollte. Konkret wurde festgelegt, dass das Gyroskop in nachfolgenden Experimenten eingesetzt wird, wenn es den Roboter präziser als die direkte Berechnung der Drehzeit nach den Formeln (4.13) und (4.14) um 180° drehen kann.

In der Vergleichsstudie wurde der Geradeausfahrteil weggelassen, sodass der Roboter in beiden Ansätzen direkt eine 180°-Drehung ausführte. Die tatsächlich gedrehten Winkel wurden mithilfe einer Gyroskop-Kalibrierungsmethode gemessen. Zu diesem Zweck wurde MATLAB Online auf einem Smartphone geöffnet und die Richtungssensorfunktion verwendet, wobei das Smartphone in der Mitte der Roboterplattform platziert wurde. Nachdem der Roboter die Drehung abgeschlossen hatte, wurde die Änderung des angezeigten Winkels berechnet und die Experimentergebnisse wie folgt dokumentiert:

Die gyroskopbasierte Richtungssteuerung		
EG	Wendewinkel	x̄
1	178.3	174.78
2	176.6	Abweichung
3	168.7	5.22
4	173.5	Varianz
5	176.8	11.6776

Abbildung 5.9 Die gyroskopbasierte Richtungssteuerung

Mit der Hilfe von Formel		
EG	Wendewinkel	x̄
1	191	191.14
2	188.2	Abweichung
3	197.9	11.14
4	182.1	Varianz
5	196.5	32.9624

Abbildung 5.10 Richtungssteuerung mit der Hilfe von Formel

Anhand der experimentellen Daten wurden der durchschnittliche Drehwinkel des Roboters, die durchschnittliche Abweichung vom Sollwinkel von 180° sowie die Varianz für beide Ansätze berechnet. Basierend auf den experimentellen Ergebnissen wurde die Genauigkeit der beiden Ansätze mittels einer gewichteten Analyse bewertet, wobei:

- Der durchschnittliche Winkelabweichung und die Varianz werden als Referenzkriterien für die gewichtete Analyse der Richtungssteuerung herangezogen. Dabei fließt die durchschnittliche Winkelabweichung mit einem Gewicht von 60% und die Varianz mit einem Gewicht von 40% in die Bewertung ein.
- Eine durchschnittliche Winkelabweichung im Bereich von 0° bis 3° wird mit 9 Punkten bewertet, im Bereich von 3° bis 6° mit 6 Punkten, und eine Abweichung von 6° oder mehr wird mit 3 Punkten bewertet.
- Eine Abweichung im Bereich von 0 bis 10 wird mit 9 Punkten bewertet, im Bereich von 10 bis 20 mit 6 Punkten, und eine Abweichung von mehr als 20 wird mit 3 Punkten bewertet.

Die Berechnung ergab die Punktzahl für die gyroskopbasierte Richtungssteuerung:

$$E = 6 * 0.6 + 6 * 0.4 = 6 \tag{5.1}$$

Die Berechnung ergab die Punktzahl für die Steuerung der Rotationszeit des Roboters durch die Formel:

$$E = 3 * 0.6 + 3 * 0.4 = 3 \tag{5.2}$$

Die Berechnungsergebnisse zeigen, dass das gyroskopbasierte Steuerungskonzept eine höhere Genauigkeit aufweist. Daher wird in den nachfolgenden Experimenten die Robotersteuerung ausschließlich mittels Gyroskops erfolgen.

5.2 TVO-Algorithmus

Bei ausschließlicher Anwendung des Basisalgorithmus fehlt dem Roboter während des Betriebs ein notwendiges Feedbacksystem. Dieses Fehlen von Rückmeldungen kann sich nachteilig auf die Optimierung der Spurverfolgung des Roboters auswirken. Daher wird in diesem Abschnitt die entscheidende Rolle von Feedbackmechanismen bei der Optimierung der Spurverfolgung des Roboters erörtert. Das Kernziel der Spurverfolgungsoptimierung besteht darin, die Übereinstimmung der tatsächlichen Robotertrajektorie mit der vorgegebenen Trajektorie zu erreichen. Um dieses Ziel zu erreichen, muss der Roboter seine Position dynamisch anhand verschiedener aus der Umgebung empfangener Feedbacksignale (einschließlich Abstandsdaten und Kursabweichung) anpassen, um die gewünschten Effekte der Spurverfolgungsoptimierung zu erzielen.

5.2.1 Methode 1: Stufenschwächungsmethode (kurz SSM)

Dieses Konzept nutzt hauptsächlich die Funktionen des Ultraschall-Entfernungsmessmoduls sowie des Gyroskops. Durch kontinuierliche Messung des Abstands zwischen dem Roboter und der rechten Wand werden die Position und die Kursinformationen des Roboters bestimmt. Basierend auf diesen Informationen werden entsprechende Anpassungen an der Fahrtrajektorie des Roboters vorgenommen.

- **„SÜ“-Route TVO**

Für die Trajektorie des Straßenüberquerens ist vorgesehen, dass der Roboter eine Geradlinienfahrt parallel zur Wand ausführt, wobei ein Abstand von 1,55m zur Wandmitte des Versuchsfeldes eingehalten wird. Es ist jedoch zu berücksichtigen, dass sich der Roboter üblicherweise nicht in der idealen Ausgangsposition befindet und die Ausrichtung des Roboters zur Wand anfänglich nicht vollständig parallel gewährleistet sein kann. Bei der Konzeption des Programms wurde dieser Umstand berücksichtigt, indem ein Feedback-Modul integriert wurde, welches es dem Roboter ermöglicht, auf Basis der Rückmeldungen seine Kursrichtung und Position entsprechend anzupassen.

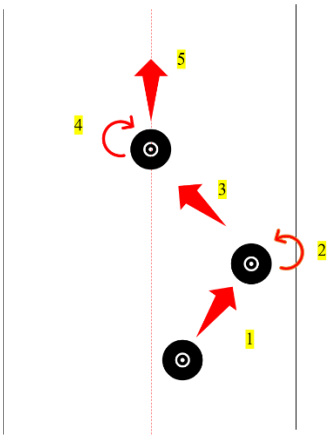


Abbildung 5.11 Schaubild der Roboter-Trajektorienverfolgung

Wie in der Grafik dargestellt, beginnt der Roboter seine Bewegung durch das Empfangen eines PWM-Signals mit einem TV von 0,3. Damit fürht eine Vorwärtsbewegung von 0,3 m durch. Das Feedbackmodul erfasst die Kursabweichung und berechnet basierend auf der zurückgelegten Distanz sowie der Veränderung des Abstands zur Wand die aktuelle Position des Roboters. Anhand dieser Daten passt der Roboter seinen Winkel an, um zur vorgegebenen Trajektorie zurückzukehren. Der Roboter setzt seine geradlinige Bewegung fort, sobald die Trajektorie korrigiert ist. Dieser Prozess wird bei jeder weiteren Abweichung wiederholt, bis der Roboter eine Gesamtstrecke von 5 m erreicht.

In Abbildung 5.11 wird eine der möglichen Szenarien dieses Optimierungsansatzes dargestellt. Im Experiment könnten jedoch die folgenden sechs Szenarien auftreten:

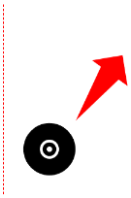


Abbildung 5.12 rechts von der Route nach rechts

Abbildung 5.13 rechts von der Route nach links

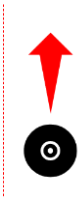


Abbildung 5.14 rechts von der Route vorwärts

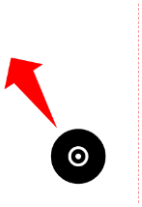


Abbildung 5.15 links von der Route nach links

Abbildung 5.16 links von der Route nach rechts

Abbildung 5.17 links von der Route vorwärts

Für diese sechs unterschiedlichen Ausgangsbedingungen trifft das Simulink-Programm entsprechende Entscheidungen und steuert den Roboter so, dass er dem jeweiligen Algorithmus folgt, um zur vorgesehenen Trajektorie zurückzukehren.

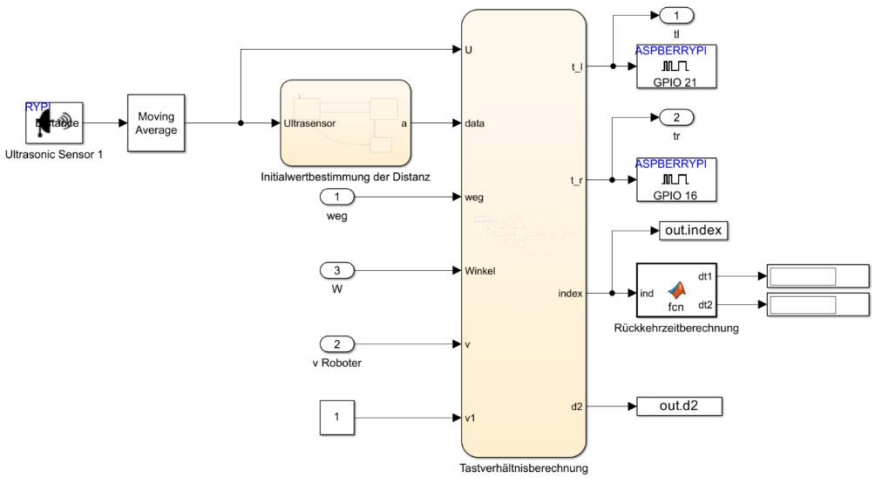


Abbildung 5.18 „SÜ“-Programm ohne PID

Wie in der Abbildung dargestellt, umfasst die Gesamtstruktur des Programms zur „SÜ“-Route die Zustandsautomaten „Bewegungssteuerungsalgorithmus“ und „Initialwertbestimmung der Distanz“. Nach dem Einlesen der Sensordaten wie Entfernung, Strecke, Winkel und Geschwindigkeit berechnet der Zustandsautomat „Bewegungssteuerungsalgorithmus“ basierend auf diesen Eingangsparametern das entsprechende TV.

Beim Einsatz des Ultraschallmoduls kann es zu Beginn der Messung zu sprunghaften Änderungen der Messwerte kommen, d. h., die Distanz kann in sehr kurzer Zeit von 0 auf den korrekten Messwert ansteigen. Um die Auswirkungen dieser anfänglichen Sprünge auf das Steuerungsprogramm zu minimieren, erfolgt vor dem eigentlichen Steuerungsprozess eine zusätzliche Berechnung der Initialwerte. Dieser Prozess wird als erste Zustandsautomat implementiert und stellt sicher, dass das System von stabilen Anfangswerten ausgeht, wodurch die Genauigkeit und Zuverlässigkeit des gesamten Steuerungsprozesses erhöht wird.

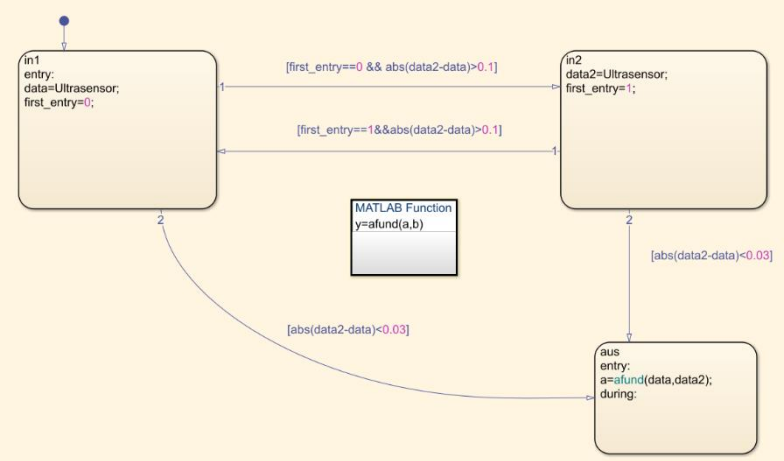


Abbildung 5.19 Zustandsautomat „Initialwertbestimmung der Distanz“

Wie in der Abbildung dargestellt, zeigt die interne Logik des Zustandsautomaten „Initialwertbestimmung der Distanz“, dass die beiden Zustände „in1“ und „in2“ kontinuierlich Daten vom Ultraschall-Entfernungsmessmodul auslesen und die Differenz der beiden Parameter berechnen. Sobald diese Differenz einen Wert von weniger als 0,03 erreicht, kann der exakte Initialwert der Distanz ermittelt werden.

Der in diesem Zustandsautomaten berechnete Distanzinitialwert sowie die Geschwindigkeit, die zurückgelegte Strecke, der Winkel und die weiteren Distanzrückmeldungen des Ultraschallentfernungsmessers werden anschließend in den zweiten Zustandsautomaten „Bewegungssteuerungsalgorithmus“ eingespeist, um weiterführende Berechnungen durchzuführen. Da die Struktur dieses Zustandsautomaten komplexer ist, wird sie in Gruppen entsprechend ihrer Funktionalität erläutert.

Zunächst wird der Prozess 1 aus Abbildung 5.11 analysiert. Hier wird das Beispiel der Ausgangsbedingung betrachtet, bei der der Roboter nach rechts von der rechten Seite abweicht:

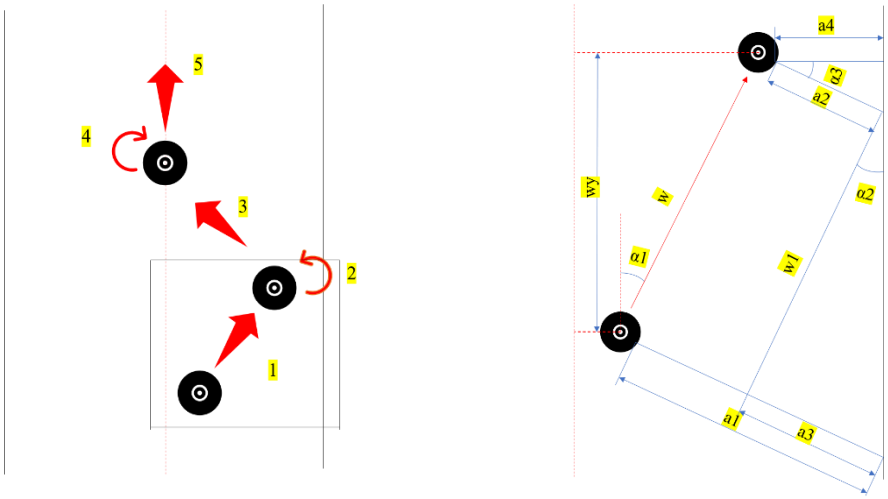


Abbildung 5.20 Prozess 1 und 2 und Parameterberechnung

Es wird die erste Route aus Abbildung 5.11 analysiert. Der Roboter bewegt sich entlang der initialen Fahrtrichtung über eine Strecke von w = 0,3. Während dieser Strecke erfolgt eine kontinuierliche Abstandsmessung mithilfe des Entfernungsmessers. Dabei werden der anfängliche Abstand a_1 sowie der Endabstand a_2 erfasst, um die Differenz der Distanzen vor und nach der Fahrt zu bestimmen.

$$a_3 = a_1 - a_2 \tag{5.3}$$

Auf Grundlage der ermittelten Daten a_3 und der zurückgelegten Strecke w wird der Winkel α_2, den die Strecke w_1 mit der Wand einschließt, mithilfe der Arkustangens-Funktion (arctan) berechnet. Bei der Berechnung wird eine Drehung im Uhrzeigersinn als positiv und eine Drehung gegen den Uhrzeigersinn als negativ definiert.

$$\alpha_2 = \tan^{-1}\frac{a_3}{w} \tag{5.4}$$

Basierend auf dem Ähnlichkeitssatz der Dreiecke werden sowohl der aktuelle Kursabweichungswinkel des Roboters als auch der Winkel α_3 ermittelt.

$$\alpha_1 = \alpha_3 = \alpha_2 \tag{5.5}$$

Der Abstand zwischen dem Roboter und der rechten Wand am Ende dieser Strecke wird mithilfe der Kosinusfunktion ermittelt.

$$a_4 = a_2 * \cos\alpha_3 \tag{5.6}$$

Auf Grundlage des oben beschriebenen mathematischen Modells wird das entsprechende Programm entwickelt.

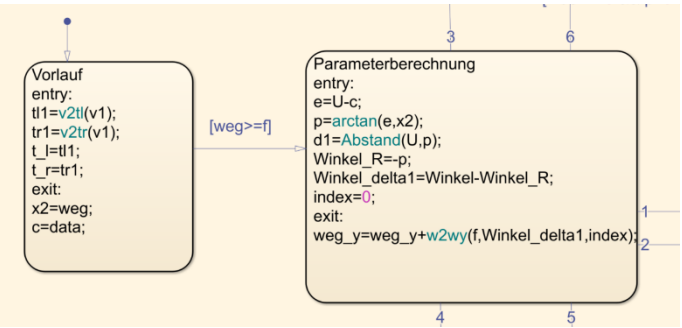


Abbildung 5.21 Zustand „Vorlauf“ (l) und Parameterberechnung (r)

Beim Eintritt in den ersten Zustand „Vorlauf“ ruft das Programm zunächst die benutzerdefinierten MATLAB-Funktionen „v2tl“ und „v2tr“ auf, um die den vorgegebenen Drehzahlen entsprechenden PWM-Tastverhältnisse zu berechnen. Diese werden anschließend an die Motoren auf der linken und rechten Seite ausgegeben, um den Roboter in Bewegung zu setzen. Nachdem der Roboter die vorgegebene Strecke von f Meter (initial auf 0,3m festgelegt) zurückgelegt hat, wird die aktuelle Fahrstrecke aufgezeichnet, und das Programm wechselt in den nächsten Zustand „Parameterberechnung“.

Im Zustand „Parameterberechnung“ wird zunächst die Änderung der vom Ultraschallentfernungsmesser gemessenen Distanz e berechnet. Anschließend werden mithilfe der benutzerdefinierten MATLAB-Funktionen „arctan“ und „Abstand“ der Kursabweichungswinkel p des Roboters sowie die vertikale Distanz d_1 zur Wand ermittelt. Die Richtung der vorgegebenen Trajektorie wird daraufhin als -p definiert, also als negativer Wert des berechneten Winkels.

Im nächsten Schritt wird die Differenz zwischen dem aktuellen Kurswinkel und dem Winkel der Trajektorie, Winkel_delta1, berechnet, um die Grundlage für die Berechnung der weiteren Fahrstrecke zu schaffen. Der Indexwert index wird auf 0 gesetzt, und der Zustandsautomat wechselt basierend auf der aktuellen Position und dem Kursabweichungswinkel des Roboters in den entsprechenden Folgezustand.

Bevor der Zustand „Parameterberechnung“ verlassen wird, erfolgt mithilfe der benutzerdefinierten MATLAB-Funktion „w2wy“ die Berechnung der tatsächlich zurückgelegten Strecke des Roboters in Richtung der vorgegebenen Trajektorie, um die notwendigen Daten für den anschließenden Zustandsübergang und die Trajekturkorrektur bereitzustellen.

Im Folgenden werden die spezifischen MATLAB-Funktionen, die in diesem Prozess aufgerufen werden, detailliert analysiert.

Für die MATLAB-Funktionen „v2tl“ und „v2tr“ lauten die entsprechenden Programme wie folgt:

```
function bl1=v2tl(v1)
if v1>0
    bl1 = - 0.072*v1 + 0.3727;
else
    bl1= - 0.0746*v1 + 0.3778;
end
```

```
function br1=v2tr(v1)
if v1>0
    br1 = -0.072*v1+0.3726;
else
    br1 = -0.0748*v1+0.3776;
end
```

Abbildung 5.22 TV-Berechnung v2tl (l) und v2tr (r)

Die in den oben genannten Programmen verwendeten Formeln basieren auf den Ergebnissen des Kalibrierungsexperiments zur Bestimmung der Beziehung zwischen TV und Drehzahl. Die beiden Programme berechnen jeweils das erforderliche TV für die gewünschte Geschwindigkeit, das in die Motoren der linken bzw. rechten Seite des Roboters eingegeben wird.

Die MATLAB-Funktionen „arctan“ und „Abstand“ dienen zur Berechnung von Winkel und Distanz:

```
function winkel=arctan(x,y)
 tW=x/y;
 winkel=rad2deg(atan(tW));
```

```
function dis1=Abstand(a,W)
W = deg2rad(W);
dis1=abs(a*cos(W));
```

Abbildung 5.23 Winkelberechnung „arctan“ (l) und Distanzberechnung „Abstand“ (r)

Bei der Berechnung des Winkels wird zunächst das Verhältnis der beiden Katheten des Dreiecks mithilfe trigonometrischer Funktionen ermittelt. Anschließend wird der Winkel mithilfe der in MATLAB integrierten Arkustangens-Funktion „atan“ berechnet und im Gradmaß ausgegeben.

Für die Berechnung der Distanz wird der eingegebene Winkel zunächst in Bogenmaß umgewandelt. Anschließend wird der Abstand zur Wand gemäß der Formel (5.6) berechnet.

Die MATLAB-Funktion „w2wy“ wird eingesetzt, um im Prozess 1 die vom Roboter entlang der vorgegebenen Trajektorie zurückgelegte Strecke zu berechnen.

```
function weg_y1=w2wy(weg1,W,ind)
W = deg2rad(W);
if ind==0
    weg_y1=abs(weg1*cos(W));
else
    weg_y1=0;
end
```

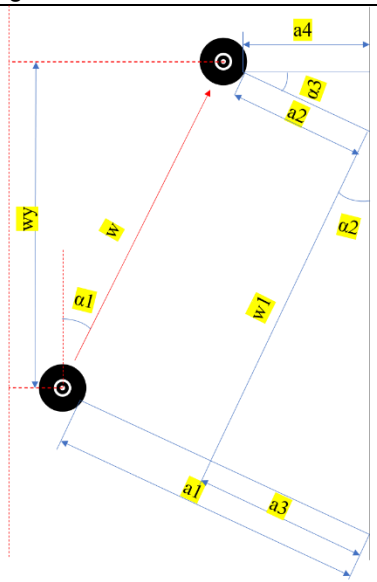


Abbildung 5.24 Streckenberechnung entlang der vorgegebenen Trajektorie w2wy (l) und Prinzip (r)

Im Rahmen des Projekts wird für die Trajektorie der SÜ eine Strecke von 5m entlang der vorgegebenen Trajektorie gefordert. Um diese Strecke präzise zu berechnen, muss der in Prozess 1 zurückgelegte Weg auf die vorgegebene Trajektorie projiziert werden. Die Funktion „w2wy“ berechnet die Länge dieser Projektion unter Verwendung der Formel (5.7).

$$w_y = |w * cos\alpha_1| \tag{5.7}$$

Die Bewegungsprozesse 2 und 3 beschreiben den Vorgang, bei dem der Roboter zur Zieltrajektorie zurückkehrt.

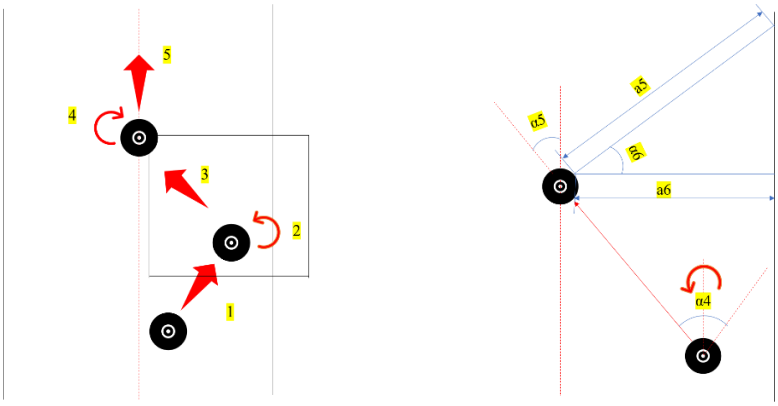


Abbildung 5.25 Rückkehr des Roboters zur vorgegebenen Trajektorie

Nach dem Eintritt in Prozess 2 rotiert der Roboter in die negative Richtung um einen Winkel, der dem Doppelten des zuvor gemessenen Kursabweichungswinkels entspricht. Diese Drehung zielt darauf ab, den Roboter wieder präzise auf die vorgegebene Trajektorie auszurichten. Nach der Winkelanpassung setzt der Roboter seine Bewegung mit der ursprünglich festgelegten Geschwindigkeit fort, bis er wieder auf die vorgegebene Trajektorie zurückkehrt.

Der entscheidende Parameter zur Bestimmung, ob der Roboter die vorgegebene Trajektorie wieder erreicht hat, ist der vertikale Abstand zur Wand. Wenn dieser Abstand den festgelegten

Bedingungen entspricht, 1,55m erreicht, kann davon ausgegangen werden, dass der Roboter die Zieltrajektorie erreicht hat. Während dieses Prozesses überwacht und berechnet der Roboter kontinuierlich seinen Kursabweichungswinkel, der in Echtzeit nach demselben Verfahren wie in Prozess 1 aktualisiert wird. Anhand der berechneten Ergebnisse und der vom Ultraschallsensor erfassten Distanz wird der vertikale Abstand des Roboters zur Wand berechnet.

$$a_6 = |a_5 * \cos \alpha_6| \tag{5.8}$$

Das folgende Programm wurde basierend auf den oben erläuterten Prinzipien entwickelt:

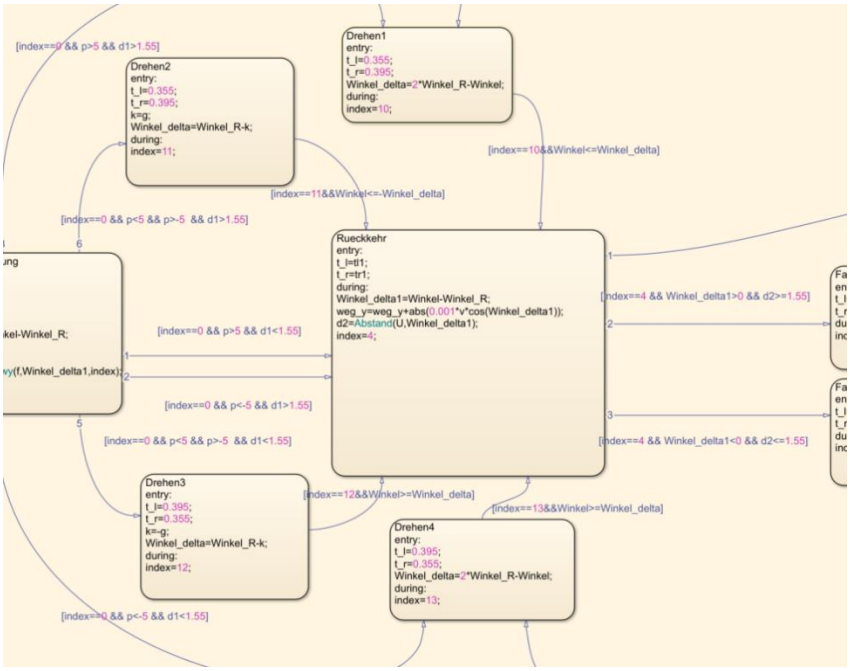


Abbildung 5.26 Fahrtrichtungsanpassungsprogramm

Im Prozess 1 wurde der Winkel der vorgegebenen Trajektorie als -p definiert. Im Zustand „Drehen 1“ oder „Drehen 4“ muss der Winkel nach der folgenden Formel :

$$\Delta W = 2 * (-p) - W \tag{5.9}$$

berechnet werden, um sicherzustellen, dass der Roboter sich um das Doppelte des anfänglichen Kursabweichungswinkels dreht.

In der Gleichung steht W für den vom Ultraschall-Entfernungsmessmodul in Echtzeit berechneten Kurswinkel. Nachdem der Roboter in diesem Zustand den Winkel ΔW durchlaufen hat, wechselt er in den nächsten Zustand „Rückkehr“. Bei den Ausgangszuständen „links parallel“ und „rechts parallel“, d. h. bei einem anfänglichen Kursabweichungswinkel zwischen -5° und 5°, wäre der Drehwinkel bei Anwendung derselben Methode zu gering, was die Rückkehr zur vorgegebenen Trajektorie verlängern würde. Daher wird im Zustand „Drehen 2“ oder „Drehen 3“ der Roboter um 15° in Richtung der Trajektorie gedreht. Für den Ausgangszustand „links parallel“ gilt somit:

$$\Delta W = -p - 15° \tag{5.10}$$

Für den Ausgangszustand „rechts parallel“ gilt somit:

$$\Delta W = -p + 15° \tag{5.11}$$

Beim Verlassen dieser Zustände ist der Roboter in jedem Fall auf die vorgegebene Trajektorie ausgerichtet. Im darauffolgenden Zustand „Rückkehr“ bewegt sich der Roboter mit den ursprünglich festgelegten TVs tl1 und tr1 entlang der vorgegebenen Trajektorie und aktualisiert dabei kontinuierlich die Strecke $w_y$in Richtung der Trajektorie. Die Methode zur Aktualisierung von w_y in diesem Zustand unterscheidet sich von der im Prozess 1. Da der Kursabweichungswinkel nun festgelegt ist, kann w_y iterativ mithilfe der Integrationsmethode berechnet werden:

$$w_{y1} = w_{y0} + |0.001 * v * \cos \Delta\alpha| \tag{5.12}$$

In der Gleichung steht 0,001 für die Abtastzeit des Programms, also die Zeitdauer einer einzelnen Iteration. Der Wert v repräsentiert die aktuell gemessene Momentangeschwindigkeit, während $\Delta\alpha$ den aktuellen Kursabweichungswinkel beschreibt.

Es ist zudem wichtig zu beachten, dass für den Fall, dass der Roboter bereits zu Beginn in Richtung der vorgegebenen Trajektorie fährt, sowie bei einer linksseitigen Rechtsabweichung und einer rechtsseitigen Linksabweichung kein „Rückkehr“-Zustand erforderlich ist, da der Roboter bereits in Richtung der Trajektorie ausgerichtet ist.

Sobald das Feedback-Modul erkennt, dass a6 ≥ 1,55m erreicht wurde, wechselt der Roboter in den nächsten Zustand.

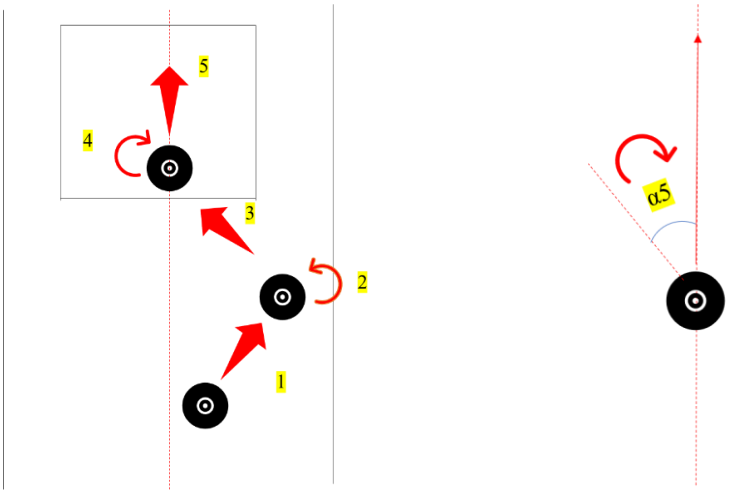


Abbildung 5.27 Ausrichtung und Trajektorienverfolgung

Basierend auf den vom Feedback-Modul erfassten Informationen zum Kursabweichungswinkel wird der Roboter in dieser Phase so ausgerichtet, dass seine Bewegungsrichtung mit der vorgegebenen Trajektorie übereinstimmt. Anschließend setzt der Roboter seine Bewegung in dieser Richtung fort. Da es trotz der vorgenommenen Anpassungen von Kurs und Position während des Betriebs möglich ist, dass der Roboter aufgrund externer Einflüsse von der vorgegebenen Trajektorie abweicht, wird das Programm zur Trajekturkorrektur bei einer erneuten

Kursabweichung reaktiviert, um sicherzustellen, dass der Roboter stets nahe an der vorgesehenen Trajektorie bleibt.

Auf Grundlage dieser Prinzipien wurde das folgende Programm entwickelt:

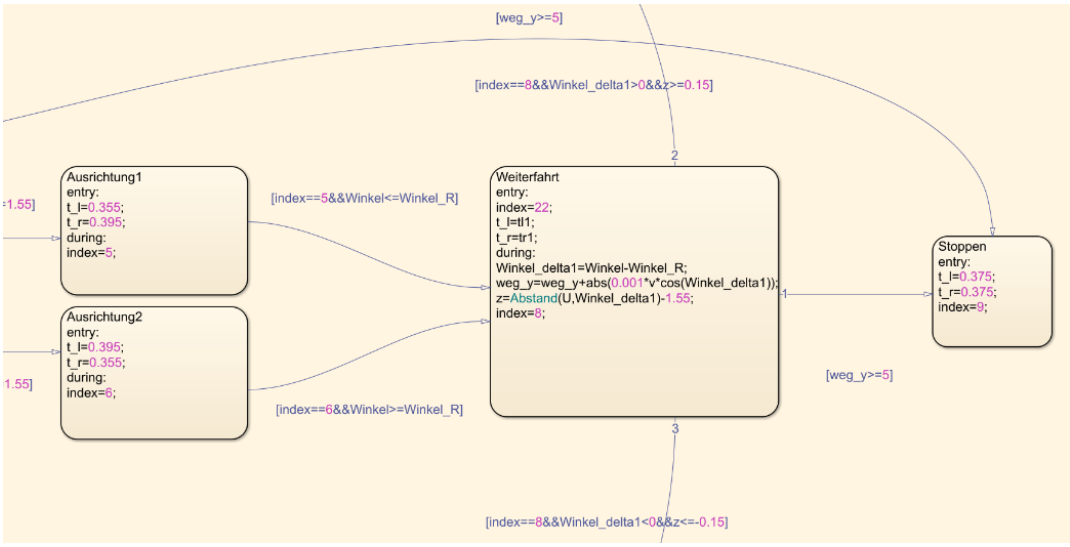


Abbildung 5.28 Ausrichtungsmanöver-Programm

Im Programm adressieren die Zustände „Ausrichtung 1“ und „Ausrichtung 2“ jeweils die Situation, in der der Kurswinkel des Roboters nach der Rückkehr zur vorgegebenen Trajektorie relativ zur Trajektorie nach links bzw. nach rechts abweicht. Nach der Korrektur des Kurswinkels wechselt der Roboter in den Zustand „Weiterfahrt“, in dem die linken und rechten Räder mit den TVs tl1 und tr1 geradeaus fahren. Weicht der Roboter um 0,15m von der Trajektorie ab, kehrt das Programm in den entsprechenden „Drehen“-Zustand zurück, um den Kurs des Roboters erneut anzupassen.

Die Zustände „Rückkehr zur Trajektorie“ und „Weiterfahrt“ beinhalten jeweils eine Übergangsbedingung zum Zustand „Stoppen“, die aktiviert wird, sobald wy ≥ 5m erreicht ist. Das bedeutet, dass der Roboter stoppt, sobald er eine Strecke von 5m entlang der Trajektorie zurückgelegt hat.

- **Verstärkungsstrategie für SSM bei „SÜ“-Route**

Im vorherigen Kapitel wurde das Trajektverfolgungssystem vorgestellt, das bereits in der Lage ist, die erwarteten Ergebnisse zu erzielen. Allerdings zeigt sich, dass dieses System gegenüber äußeren Störungen eine geringe Resistenz aufweist. Sobald eine externe Kraft den Kurswinkel des Roboters kurzfristig verändert, benötigt das Programm eine erhebliche Zeit, um die Trajektorie zu korrigieren. Daher besteht weiterhin Optimierungspotenzial für das oben beschriebene System.

Eine relativ einfache Optimierungsmöglichkeit besteht in der Integration eines PID-Regelungsprinzips. Dabei wird in den verschiedenen Zuständen ein PID-Regler eingesetzt, um die Auswirkungen äußerer Störungen zu minimieren. Das Grundprinzip besteht darin, dass der PID-Regler die Differenz zwischen dem aktuellen Winkel und dem Referenzwinkel berechnet und

daraus die Geschwindigkeitsänderungen der beiden Räder ableitet. Diese Geschwindigkeitsänderungen werden anschließend mithilfe der Beziehung zwischen TV und Drehzahl in TV umgewandelt und an die Motoren der linken und rechten Seite ausgegeben.

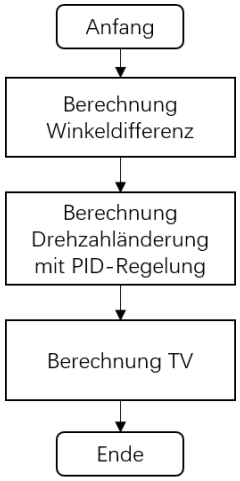


Abbildung 5.29 Programmablaufplan Winkel PID-Regelung

Wie in der Abbildung 3.4 dargestellt, wurde im Experiment ein PID-Regler für die Winkelsteuerung eingesetzt. Durch die Integration des Winkel-PID-Reglers in das im vorherigen Kapitel verwendete Programm entstand das folgende neue Programm:

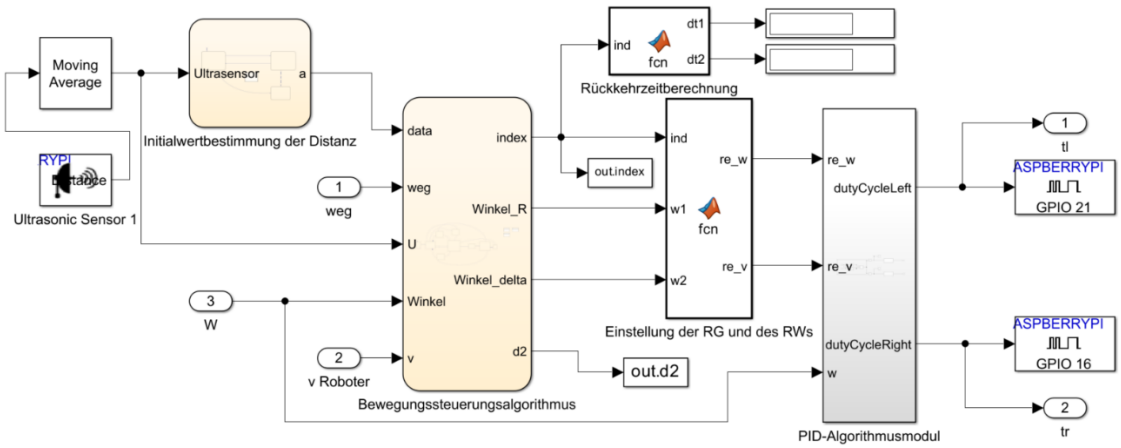


Abbildung 5.30 SSM-Verstärkungsstrategie

Im neuen Programm werden die ausgegebenen TV-Werte alle durch das PID-Algorithmusmodul berechnet. Allerdings werden in den Zuständen „Drehen“ und „Stoppen“ die Eingangsparameter angepasst. In diesen Zuständen wird im PID-Modul eine Referenzgeschwindigkeit von 0 eingestellt, während in den übrigen Zuständen eine Referenzgeschwindigkeit von 1 r/s verwendet wird.

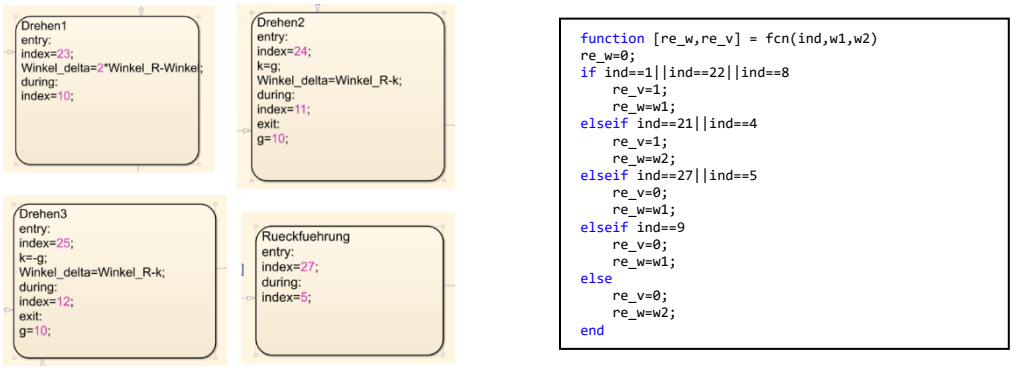


Abbildung 5.31 Zustände des Zustandsautomaten und Einstellung der Referenzparameter

Es ist wichtig zu beachten, dass der Referenzwinkel in den verschiedenen Zuständen variiert. Die spezifischen Referenzwinkel basieren auf den Winkelberechnungen, die in den einzelnen Prozessen der SSM durchgeführt wurden, und werden entsprechend in der MATLAB-Funktion „re_w“ eingestellt, wie in der Abbildung dargestellt. Dabei stellt w1 den im Programm kontinuierlich iterierten Winkel der vorgegebenen Trajektorie dar, während w2 den Kurswinkel des Roboters beim Rückkehren zur Trajektorie repräsentiert.

- **„GE“-Route TVO**

Die Route „GE“ stellt eine Weiterentwicklung der „SÜ“-Route dar. In der Routenplanung wurde die „GE“-Route um das zusätzliche Manöver eines Wendens und anschließenden Geradeausfahrens erweitert.

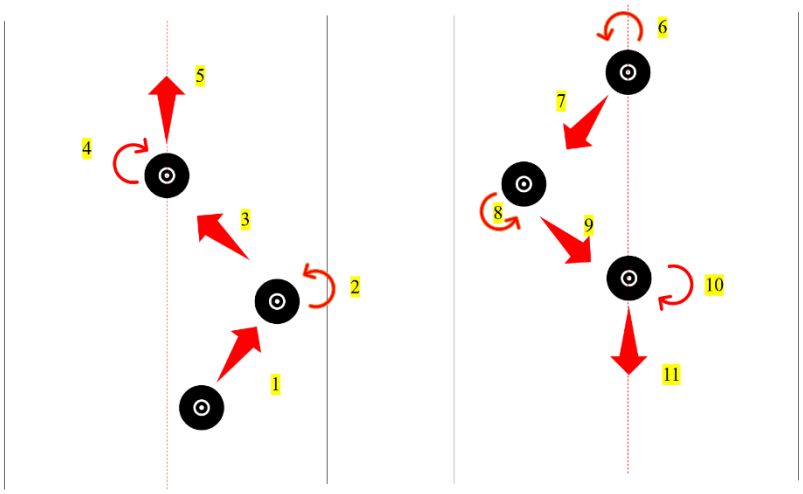


Abbildung 5.32 Grundlegender Ablauf des „GE“

Wie in der Abbildung dargestellt, dreht sich der Roboter nach Abschluss der Geradlinienstrecke in Prozess 6 gegen den Uhrzeigersinn um 180°. Da die Genauigkeit dieser Drehung nicht vollständig gewährleistet werden kann, wiederholen die Prozesse 7 bis 11 im Wesentlichen die Berechnungen der Prozesse 1 bis 5, um die Position und den Kursabweichungswinkel des Roboters zu ermitteln und ihn zurück auf die vorgegebene Trajektorie zu bringen.

Im Unterschied zur TVO bei der SÜ erfordert die Route „GE“, dass der Roboter nach dem Wenden eine zusätzliche Strecke zurücklegt, nämlich nach einer 180°-Drehung gegen den Uhrzeigersinn noch 3m geradeaus fährt. Im Folgenden wird das entsprechende Programmalgorithmus-Design erläutert.

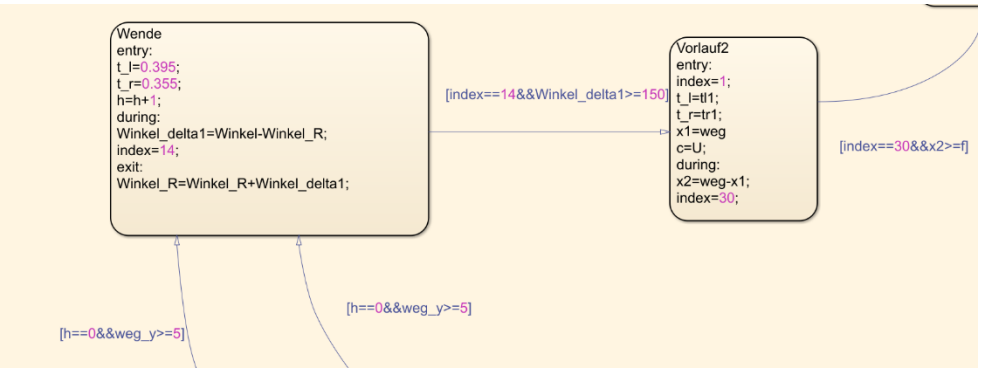


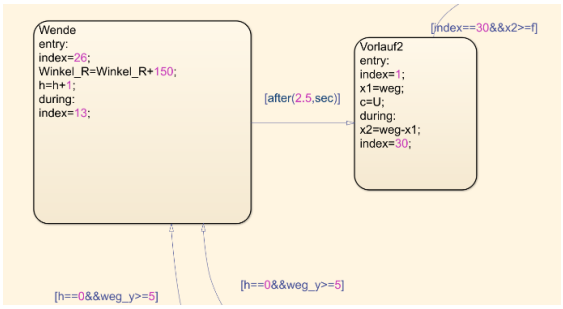
Abbildung 5.33 Wendemanöver-Programm

Im Verlauf des Experiments wird, sobald die gefahrene Strecke des Roboters 5m erreicht, automatisch das Drehprogramm gestartet, und der Zustand „Wende“ aktiviert. Dabei werden die TVs für die linken und rechten Räder auf 0,395 bzw. 0,355 gesetzt, um eine angemessene Drehgeschwindigkeit zu gewährleisten. Während der Drehung berechnet das System fortlaufend die Differenz zwischen dem aktuellen Kurswinkel und dem Winkel der vorgegebenen Trajektorie. Sobald eine Winkelabweichung von 180° erreicht wird, wechselt das Programm in den Zustand „Vorlauf 2“, in dem der Roboter seine aktuellen Parameter neu berechnet und die Schritte 1 bis 5 des Straßenüberquerungsprogramms erneut durchläuft.

In der Praxis zeigte sich jedoch, dass die Verwendung eines Referenzwinkels von 180° häufig zu einer Überdrehung des Roboters führte, wodurch dieser von der vorgegebenen Trajektorie abwich. Um dieses Problem zu beheben, wurde der Referenzwinkel im Experiment auf 150° angepasst. Diese Änderung reduzierte effektiv das Risiko einer Überdrehung und verbesserte die Genauigkeit und Stabilität der Roboter-Navigation unter komplexen Bedingungen. Auf diese Weise konnte der Roboter besser auf die Anforderungen des Experimentierfeldes reagieren und die Erfolgsquote des Experiments insgesamt steigern.

- **Verstärkungsstrategie für SSM bei „GE“-Route**

Der PID-Regelungsalgorithmus zur Stabilisierung des Roboters während der linearen Bewegung bleibt aktiv. Um sicherzustellen, dass der Roboter auch während der Drehung gegen den Uhrzeigersinn weniger anfällig für äußere Störungen ist, wird der PID-Algorithmus auch in diesen Prozess integriert.




```
function [re_w,re_v] = fcn(ind,w1,w2)
re_w=0;
if ind==1||ind==22||ind==8||ind==30
    re_v=1;
    re_w=w1;
elseif ind==21||ind==4
    re_v=1;
    re_w=w2;
elseif ind==27||ind==5||ind==26||ind==13
    re_v=0;
    re_w=w1;
elseif ind==9
    re_v=0;
    re_w=w1;
else
    re_v=0;
    re_w=w2;
end
```

Abbildung 5.34 Wendemanöver-PID Regelung

Während der Wende wird das PID-Algorithmusmodul verwendet, das auf der „Stufenschwächung“-PID-Regelungsstrategie basiert, welche bereits im Trajekturverfolgungssystem für die SÜ eingesetzt wurde. Die ursprünglichen Einstellungen der Proportional-, Integral- und Differenzialkoeffizienten K_p， K_i und K_d bleiben dabei unverändert. In der Parametrierung des PID-Moduls wird der Trajektorienwinkel der Geradeausfahrt um 150° erweitert und als Referenzwinkel in den PID-Regler eingespeist. Da sich der Roboter in diesem Zustand in einer stationären Drehbewegung befindet, wird die Referenzgeschwindigkeit auf null gesetzt. Auf dieser Basis berechnet das PID-Algorithmusmodul das entsprechende TV, um sicherzustellen, dass der Roboter während des Drehvorgangs sowohl stabil als auch präzise bleibt.

- **Versuchsdaten und Datenanalyse**

Im Experiment wurde der Roboter mit einem Kursabweichungswinkel von 15° nach rechts und einem Abstand von 0,3m zur rechten Seite der vorgegebenen Trajektorie positioniert und getestet (siehe Abbildung 5.35). Dabei wurden die relevanten Parameter gemessen und aufgezeichnet. Für die Route „SÜ“ wurden die Rückkehrzeit (kurz RKZ), Anzahl der Anpassungen (kurz AA), der maximale und minimale vertikale Abstand zur Wand nach der Rückkehr (kurz VAW) sowie die entlang der Trajektorie zurückgelegte Strecke (kurz SET) erfasst. Für die Route „GE“ wurden, unter Berücksichtigung der erforderlichen Neuausrichtung des Roboters nach der Drehung, die entsprechenden Daten sowohl vor als auch nach dem Wenden aufgezeichnet.

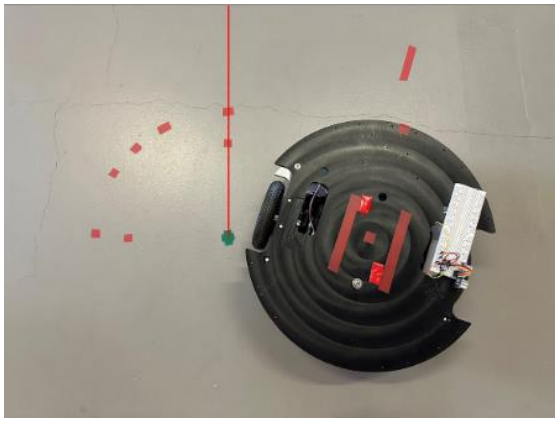


Abbildung 5.35 Anfangsposition des Roboters und die Makierung auf dem Boden

Wie in Abbildung 5.35 dargestellt, ist der Mittelpunkt des Roboters auf den am Boden markierten roten Punkt ausgerichtet, der eine Entfernung von 0,3m zur Trajektorie anzeigt. Die Verbindungslinie der beiden roten Punkte auf der schwarzen Plattform des Roboters ist mit der Winkelmarkierungslinie ausgerichtet, welche einen Kursabweichungswinkel von 15° nach rechts darstellt.

Nach Abschluss des Experiments wurden die folgenden Daten erfasst:

Straßenüberquerung Stufenschwächung ohne PID					
VG	RKZ[s]	AA	VAWmax[m]	VAWmin[m]	SET[m]
1	8.055	4	1.55	1.29	7.8
2	6.913	2	1.73	1.55	7.9
3	8.157	3	1.55	1.29	7.7
4	8.378	3	1.56	1.29	7.9
5	7.747	3	1.56	1.32	8
$\bar{x}$	7.85	3	1.59	1.348	7.86
σ^2	0.32591	0.5	0.00615	0.01292	0.013

Tabelle 5.1 Versuchsdaten- „SÜ“ SSM ohne PID

Straßenüberquerung Stufenschwächung mit Winkel PID					
VG	RKZ[s]	AA	VAWmax[m]	VAWmin[m]	SET[m]
1	6.691	1	1.64	1.55	5.6
2	6.295	1	1.63	1.49	5.57
3	6.608	3	1.65	1.43	6.13
4	6.365	2	1.63	1.49	6.07
5	6.492	3	1.64	1.41	6.7
$\bar{x}$	6.4902	2	1.638	1.474	6.014
σ^2	0.02699	1	7E-05	0.00308	0.21393

Tabelle 5.2 Versuchsdaten- „SÜ“ SSM mit Winkel PID

Die vergleichende Analyse der experimentellen Daten zeigt signifikante Unterschiede zwischen den Ergebnissen der „SÜ“-Experimente mit und ohne PID-Regelung. Wie in der Tabelle dargestellt, benötigt der Roboter in der Gruppe ohne PID-Regelung durchschnittlich 7,85s, um zur vorgegebenen Trajektorie zurückzukehren, was etwa 1,3s langsamer ist als in der Gruppe mit Winkel-PID-Regelung. Nach der Rückkehr zur Trajektorie passt der Roboter seine Fahrtrichtung basierend auf seiner Position an. In Bezug auf die Anzahl der Anpassungen benötigt der Roboter in der Gruppe ohne PID-Regelung durchschnittlich etwa drei Anpassungen, während der Roboter in der Gruppe mit PID-Regelung nur etwa zwei Anpassungen benötigt, was auf eine höhere Stabilität hinweist.

Die Messungen des Abstands zur Wand bestätigen diese Beobachtungen. In der Gruppe ohne PID-Regelung beträgt der maximale Abstand zur Wand nach der Rückkehr zur Trajektorie 1,59m, der minimale Abstand 1,35m, was zu einer Trajektorienschwankung von 0,24m führt. Im Gegensatz dazu weist die Gruppe mit Winkel-PID-Regelung einen maximalen Abstand von 1,64m und einen minimalen Abstand von 1,47m auf, wodurch die Trajektorienschwankung auf 0,17m reduziert wird.

Hinsichtlich der zurückgelegten Strecke zeigt sich, dass der Roboter in der Gruppe mit Winkel-PID-Regelung näher an der Zielstrecke von 5m liegt, mit einer tatsächlichen Fahrtstrecke von etwa 6,01m, was einer Überschreitung von etwa 20% entspricht. Im Vergleich dazu überschreitet

der Roboter in der Gruppe ohne PID-Regelung die Strecke um 2,9m, was einer Abweichung von 58% entspricht. Dies unterstreicht die höhere Präzision der Winkel-PID-Regelung.

Obwohl die Gruppe ohne PID-Regelung hinsichtlich der Varianz stabilere Daten aufweist, was auf eine höhere Konsistenz der Ergebnisse hinweist, zeigen die Ergebnisse der Gruppe mit Winkel-PID-Regelung insgesamt eine bessere Annäherung an die angestrebten Werte. Daher erweist sich die SSM mit PID-Regelung als besser geeignet, um die experimentellen Ziele zu erreichen.

Während des Experiments wurden auch der vertikale Abstand des Roboters zur Wand und die Indexwerte erfasst und im MATLAB-Arbeitsbereich gespeichert. Diese Daten wurden zur Erstellung eines Streudiagramms verwendet, das die Veränderung der Position des Roboters über die Zeit hinweg darstellt. Durch die Eingabe dieser Informationen in ein entsprechendes Grafikprogramm (siehe Anlage 26) lassen sich die Positionsveränderungen des Roboters im Zeitverlauf visuell darstellen und analysieren, was eine umfassendere Bewertung seiner Bewegungsleistung und der Wirksamkeit der Steuerungsstrategien ermöglicht.

Die im Hauptprogramm des Experiments gesammelten Daten zum vertikalen Abstand zur Wand „d2“ und den Indexwerten „index“ werden in das Analyseprogramm importiert. Anschließend wird der Punkt identifiziert, an dem index = 22 erreicht wird, was den Zeitpunkt der Rückkehr zur Trajektorie markiert. Dieser Punkt dient als Ausgangspunkt für die weitere Analyse, um die entsprechenden Abstand- und Zeitdaten zu extrahieren.

In den gefilterten Daten werden dann der maximale und der minimale Abstand sowie die entsprechenden Zeitpunkte berechnet. Schließlich werden die gefilterten Daten im Diagramm mit einer roten Linie hervorgehoben, wobei die maximalen und minimalen Werte zusätzlich gekennzeichnet werden.

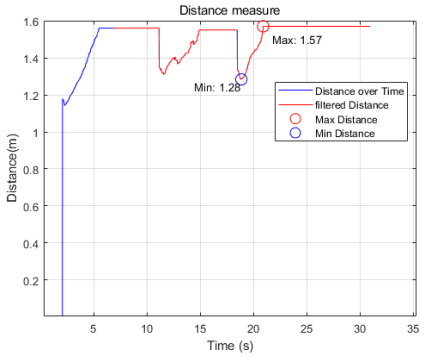


Abbildung 5.36 Streudiagramm der VAW ohne PID bei der SSM (SÜ)

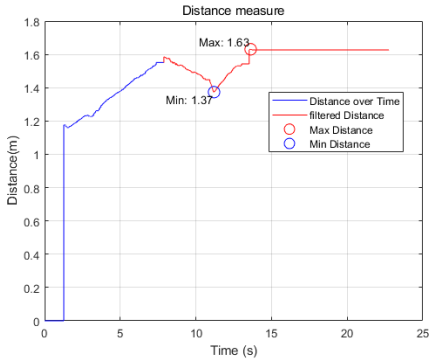


Abbildung 5.37 Streudiagramm der VAW mit Winkel PID bei der SSM (GE)

Aus der obigen Abbildung lässt sich ebenfalls erkennen, dass der Roboter in der Versuchsgruppe mit Winkel-PID-Kontrolle deutlich bessere Leistungen zeigt. Die Anzahl der notwendigen Anpassungen der Fahrtroute ist geringer, und die Schwankungen im vertikalen Abstand zur Wand sind in der Gruppe mit PID-Kontrolle geringer als in der Gruppe ohne PID-Kontrolle. Dies zeigt, dass die Winkel-PID-Kontrolle einen signifikanten Vorteil bei der Verbesserung der Stabilität und Genauigkeit der Robotertrajektorie bietet.

Für die zwei Versuchsgruppen der „GE“-Route wurden die folgenden beiden Datensätze erfasst:

Gästeempfang Stufenschwächung ohne PID					
VG	RKZ1[s]	AA1	VAWmax1[m]	VAWmin1[m]	SET1[m]
1	8.047	4	1.77	1.24	8.9
2	8.127	3	1.83	1.24	7.8
3	8.318	4	1.74	1.55	8.1
4	7.645	3	1.75	1.54	7.6
5	7.975	3	1.78	1.54	8.7
$\bar{x}$	8.0224	3.4	1.774	1.422	8.22
σ^2	0.0609	0.3	0.00123	0.02762	0.317

Tabelle 5.3 Versuchsdaten bei der SSM ohne PID (GE) 1

RKZ2[s]	AA2	VAWmax2[m]	VAWmin2[m]	SET2[m]
nan	nan	nan	nan	4.9
3.811	2	1.61	1.33	5
4.563	3	1.55	1.41	4.6
nan	nan	nan	nan	nan
nan	nan	nan	nan	nan
4.187	2.5	1.58	1.37	4.83333333
0.282752	0.5	0.0018	0.0032	0.04333333

Tabelle 5.4 Versuchsdaten bei der SSM ohne PID (GE) 2

Gästeempfang Stufenschwächung mit Winkel PID					
VG	RKZ1[s]	AA1	VAWmax1[m]	VAWmin1[m]	SET1[m]
1	7.211	1	1.63	1.53	5.5
2	6.812	1	1.65	1.53	6
3	7.027	1	1.65	1.53	5.8
4	7.745	2	1.65	1.33	6.4
5	7.866	1	1.63	1.43	5.8
$\bar{x}$	7.3322	1.2	1.642	1.47	5.9
σ^2	0.20845	0.2	0.00012	0.008	0.11

Tabelle 5.5 Versuchsdaten bei der SSM mit Winkel PID (GE) 1

RKZ2[s]	AA2	VAWmax2[m]	VAWmin2[m]	SET2[m]
4.687	1	1.62	1.31	3.6
4.065	2	1.56	1.31	4
2.803	2	1.63	1.34	5
3.33	3	1.71	1.17	5.1
4.585	2	1.7	1.51	4.3
3.894	2	1.644	1.328	4.4
0.660987	0.5	0.00383	0.01472	0.415

Tabelle 5.6 Versuchsdaten bei der SSM mit Winkel PID (GE) 2

In diesem Experiment wurden die Daten in zwei Gruppen unterteilt: Die ersten Datensätze repräsentieren die Ergebnisse vom Beginn der Roboterbewegung bis unmittelbar vor dem Wenden, während die zweiten Datensätze die Daten vom Ende der Drehung bis zum Ende des Experiments umfassen. Die Ergebnisse zeigen, dass die Winkel-PID-Regelung in beiden Datengruppen eine höhere Stabilität und Genauigkeit aufweist.

Im Detail ergaben sich bei den Robotern ohne PID-Regelung nach dem Wenden bei drei Versuchsdurchläufen „nan“-Werte, was darauf hindeutet, dass der Roboter aufgrund systematischer Fehler nach der Drehung nicht mehr in der Lage war, zur vorgegebenen Trajektorie zurückzukehren. Die Roboter mit Winkel-PID-Regelung kehrten deutlich schneller zur Trajektorie zurück als die Roboter ohne PID-Regelung. Die durchschnittliche Rückkehrzeit der Roboter mit Winkel-PID-Regelung betrug 7,33s und 3,89s, während die Roboter ohne PID-Regelung durchschnittlich 8,02s und 4,19s benötigten, was allgemein etwa 0,5s langsamer war. Zudem mussten die Roboter ohne PID-Regelung nach der Rückkehr zur Trajektorie häufiger Anpassungen vornehmen als die Roboter mit Winkel-PID-Regelung.

Nach der Rückkehr zur Trajektorie betrugen die durchschnittlichen maximalen und minimalen Abstände der Roboter ohne PID-Regelung zur Wand 1,774m und 1,422m sowie 1,58m und 1,37m, während die Roboter mit Winkel-PID-Regelung 1,64m und 1,47m sowie 1,64m und 1,33m erreichten. Diese Ergebnisse zeigen, dass die Abweichungen der Roboter ohne PID-Regelung deutlich größer waren als die der Roboter mit Winkel-PID-Regelung.

Darüber hinaus betrugen die erwarteten Strecken, die die Roboter vor und nach dem Wenden entlang der vorgegebenen Trajektorie zurücklegen sollten, jeweils 5m und 3 m. In beiden Versuchsgruppen überschritten die Roboter diese Strecken, jedoch war die Überschreitung bei den Robotern mit Winkel-PID-Regelung geringer, was darauf hindeutet, dass sie die Ziele des Experiments genauer erreichten.

Für dieses Experiment gibt es ebenfalls ein Programm zur Erstellung eines Streudiagramms, das den vertikalen Abstand des Roboters zur Wand darstellt. (sehe Anlage 27)

Im Vergleich zur Analyse der „SÜ“-Route wurde im Programm für diese Route der Markierungswert h eingeführt (h = 0 zeigt an, dass der Roboter noch nicht gewendet hat, h = 1 weist darauf hin, dass das Wenden abgeschlossen ist). Um den Start- und Endpunkt der Trajektorie nach der Rückkehr zur vorgegebenen Trajektorie vor dem Wenden zu bestimmen, wird die Position gesucht, an der index = 22 und h = 1 erreicht sind. Die Daten nach dem Wenden werden ab dem Punkt h = 1 extrahiert, und in diesen Daten wird die Position ermittelt, an der index = 22 erreicht ist, um den Startpunkt der zweiten Trajektorie zu bestimmen.

Die beiden Datensätze werden dann in der Grafik mit roten und hellvioletten Linien dargestellt. In den gefilterten Datensätzen werden der maximale und der minimale Abstand sowie die entsprechenden Zeitpunkte berechnet und in der Grafik markiert.

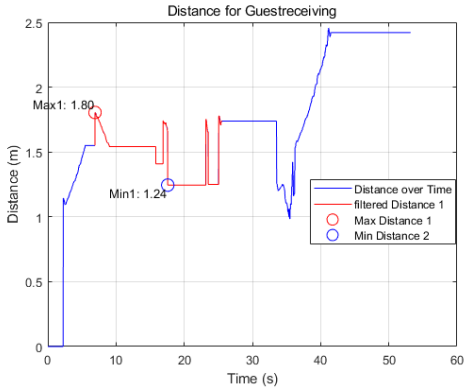


Abbildung 5.38 Streudiagramm der VAW ohne PID bei der SSM (GE)

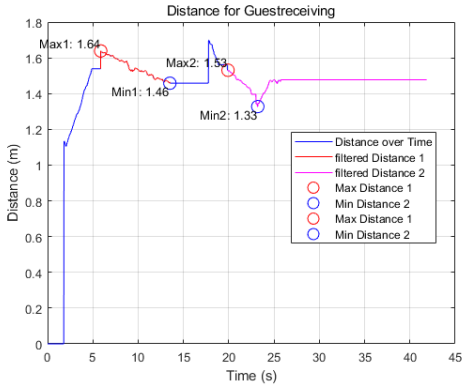


Abbildung 5.39 Streudiagramm der VAW mit Winkel PID bei der SSM (GE)

Auch aus der Abbildung lässt sich erkennen, dass der Roboter in der Versuchsgruppe ohne PID-Regelung erhebliche Schwankungen in der Trajekturverfolgung aufweist. Nach dem Wenden gelingt es dem Roboter nicht, die vorgegebene Trajektorie beizubehalten, was zur Folge hat, dass die Parameter nach dem Wenden nicht berechnet werden können. Im Gegensatz dazu kann der Roboter mit Unterstützung der Winkel-PID-Regelung während des Betriebs die Abweichungen von der vorgegebenen Trajektorie reduzieren, was zu einer stabileren Fahrt führt. Der vertikale Abstand nach der Rückkehr zur Trajektorie bleibt dabei innerhalb eines akzeptablen Bereichs. Im Experiment, das in Abbildung 5.39 dargestellt ist, wurde der vertikale Abstand vor dem Wenden auf 1,64m bis 1,46m und nach dem Wenden auf 1,53m bis 1,33m begrenzt.

Zusammenfassend zeigt sich, dass der Roboter mit Winkel-PID-Regelung in Bezug auf die Rückkehrzeit, die Anzahl der Anpassungen sowie die Fahrgenauigkeit dem Roboter ohne PID-Regelung überlegen ist und eine höhere Stabilität und Genauigkeit aufweist.

5.2.2 Methode 2: Die segmentierte PID-Methode

Im Rahmen der SSM erfolgt die Kontrolle des Abstands zur Wand durch das Design des Programms selbst. Dies erforderte im Experiment eine erhebliche Menge an Zeit für die Anpassung des Programms. Im Gegensatz dazu wird in der segmentierten PID-Methode der Abstand des Roboters zur Wand auch durch das PID-Algorithmusmodul geregelt, was den Zeitaufwand für die Programmjustierungen im Vorfeld erheblich reduziert.

- **Programmentwicklung**

In beiden Routen ist die verwendete Programmstruktur weitgehend identisch. Es werden die Daten des Ultraschallsensors zur Distanz, die aktuelle Geschwindigkeit, die zurückgelegte Strecke sowie der Kurswinkel erfasst. Basierend auf diesen Informationen analysiert der Zustandsautomat „Verhaltenssteuerung“ die Situation und veranlasst den Roboter, in den entsprechenden Zustand zu wechseln. Der Roboter entscheidet anhand der vom Zustandsautomaten bereitgestellten Informationen, welches PID-Modul zum Einsatz kommt. Dabei handelt es sich entweder um den Winkel-PID-Regler oder den Distanz-PID-Regler.

Der Ablauf sieht vor, dass der Roboter zunächst 0,3m vorfährt, um seine Position und den Kursabweichungswinkel zu bestimmen. Bei einer erheblichen Abweichung vom vorgegebenen Trajektorium wird der Distanz-PID-Regler aktiviert, der mithilfe der berechneten TVs den Roboter näher an die vorgegebene Trajektorie heranführt. Sobald der Roboter wieder auf der vorgegebenen Trajektorie erkannt wird, übernimmt der Winkel-PID-Regler die Steuerung, um die Fahrtrichtung des Roboters zu kontrollieren.

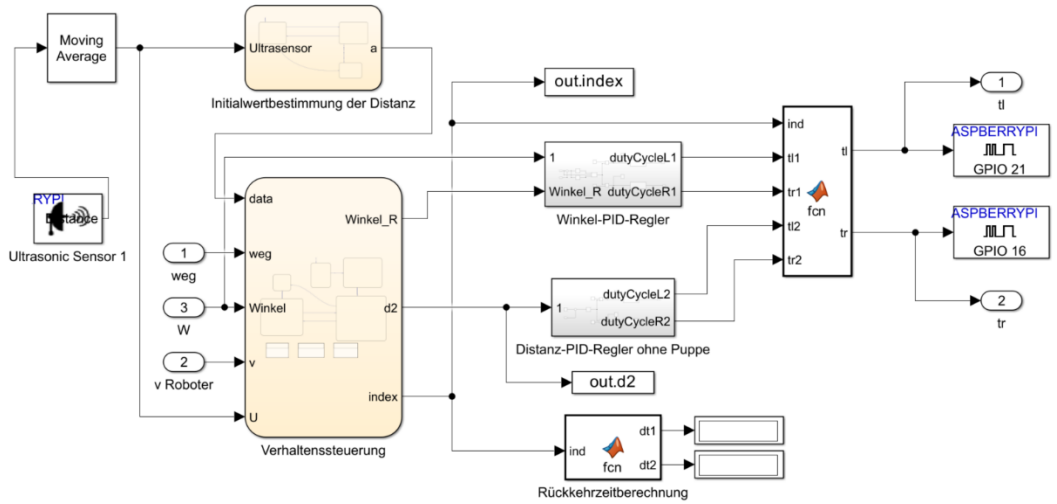


Abbildung 5.40 Segmentiertes PID-Steuerungsprogramm (SÜ)

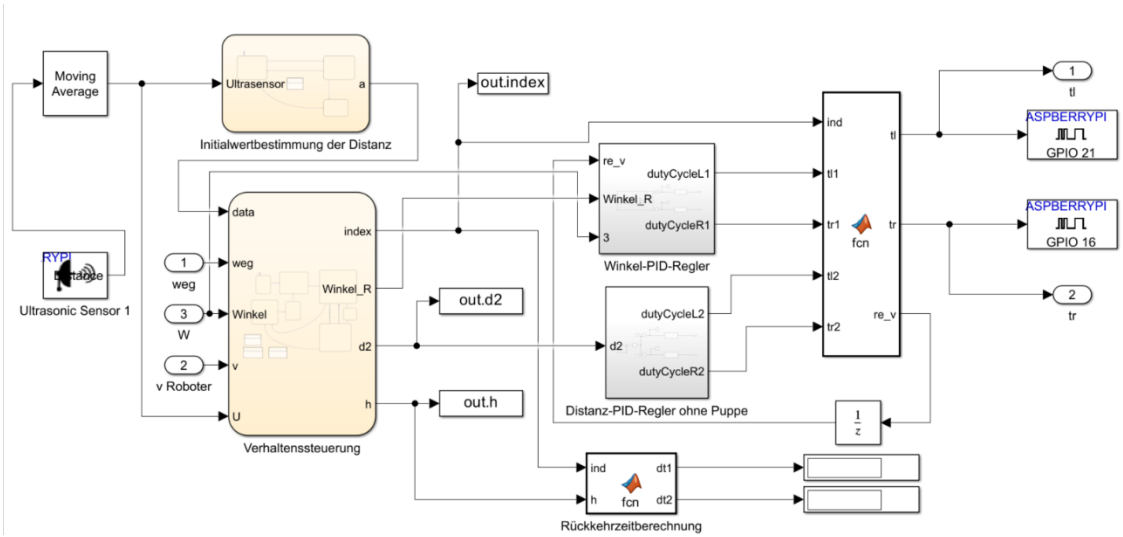


Abbildung 5.41 Segmentiertes PID-Steuerungsprogramm (GE)

Die Struktur beider Programme ist im Wesentlichen gleich. Da der Roboter in der „GE“-Route jedoch einen Wendemanöver durchführt, muss während dieses Prozesses der Winkel-PID-Regler eingesetzt werden, um die Richtung des Roboters anzupassen. Dabei muss die Referenzgeschwindigkeit auf 0 gesetzt werden. Daher wird am Eingang des Winkel-PID-Reglers eine zusätzliche Eingabe für die Referenzgeschwindigkeit integriert.

- **Die segmentierte PID-Methode bei der SÜ**

Im Experiment werden die Informationen zu Distanz, Winkel, Geschwindigkeit und anderen relevanten Parametern in den Zustandsautomaten „Verhaltenssteuerung“ eingespeist. Der Zustandsautomat führt folgende Berechnungen durch:

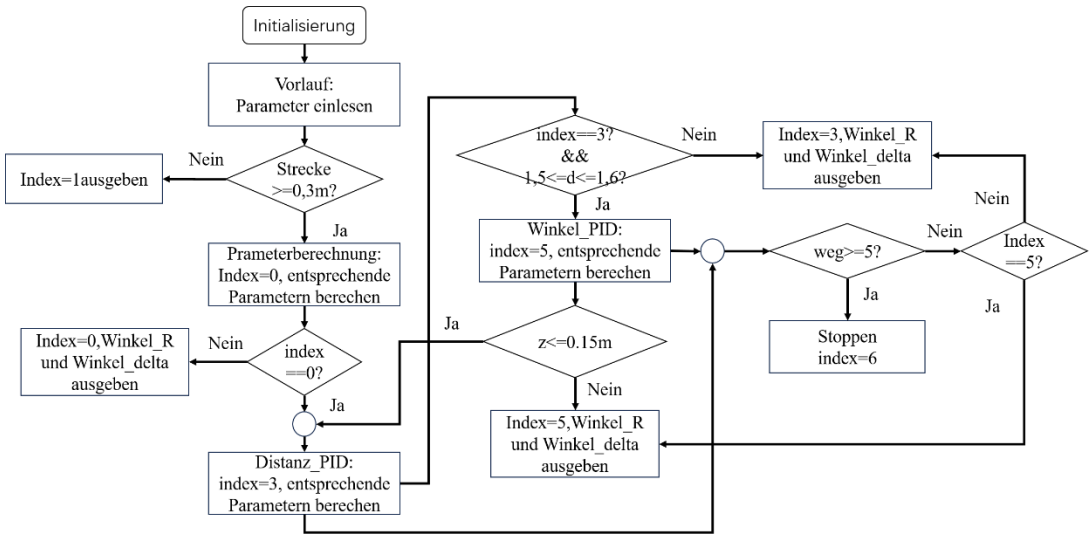


Abbildung 5.42 Programmablaufplan für Zustandsautomat „Verhaltenssteuerung“ bei der segmentierten PID-Methode (SÜ)

Nachdem im Zustand „Vorlauf“ und im Zustand „Parameterberechnung“ der Kursabweichungswinkel und die Position des Roboters ermittelt wurden, aktiviert der Zustandsautomat den „Distanz-PID“-Modus. Sobald der gemessene vertikale Abstand des Roboters zur Wand „d2“ zwischen 1,6m und 1,5m liegt, wird der aktuelle Zustand des Roboters als „Rückkehr zur vorgegebenen Trajektorie“ definiert. In diesem Fall wechselt das Programm in den Zustand „Winkel-PID“, in dem der Roboter seine Ausrichtung auf Basis der im „Vorlauf“ ermittelten Winkelparameter korrigiert, um die Parallelität zur vorgegebenen Trajektorie sicherzustellen. Sollte während des Betriebs eine Kursabweichung auftreten, d. h., wenn d2 kleiner als 1,40m oder größer als 1,70m wird, kehrt das Programm in den Zustand „Distanz-PID“ zurück, um die Position des Roboters erneut zu justieren. (Das entsprechende Simulink-Programm ist wie Anlage 22)

- **Die segmentierte PID-Methode beim GE**

Basierend auf dem „SÜ“-Programm mit segmentierter PID-Methode wird nun ein Wende-Manöver integriert.

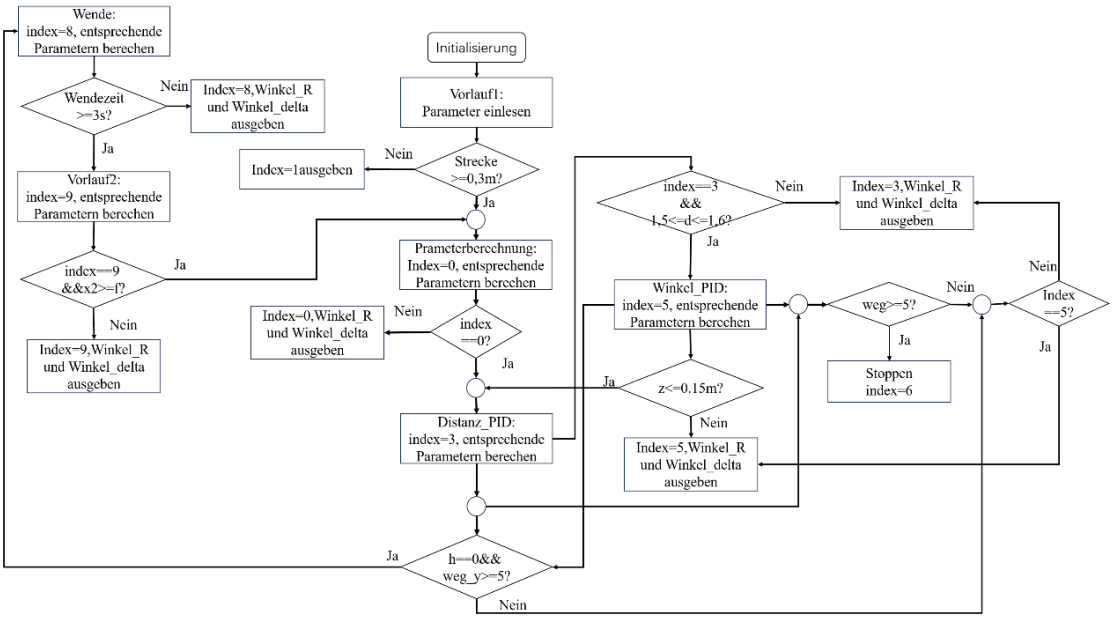


Abbildung 5.43 Programmablaufplan für Zustandsautomat „Verhaltenssteuerung“ bei der segmentierten PID-Methode (GE)

Vor dem Wenden ist das segmentierte PID-Steuerungsschema der „GE“-Route weitgehend identisch mit dem der „SÜ“-Route. Nachdem der Roboter jedoch eine Strecke von 5m zurückgelegt hat, wechselt das Programm, unabhängig davon, ob sich der Roboter im „Distanz-PID“- oder „Winkel-PID“-Zustand befindet, in den „Wende“-Zustand.

Ursprünglich war geplant, dass der Roboter eine Drehung um 180° durchführt. Im Experiment zeigte sich jedoch, dass eine Eingabe von 180° zu einer übermäßigen Drehung des Roboters führte. Daher wurde der Drehwinkel auf 150° angepasst, um eine präzisere Richtungsänderung zu gewährleisten.

Nach dem Wenden durchläuft der Roboter die Zustände „Vorlauf 2“ und „Parameterberechnung“, um seine aktuelle Position und den Kursabweichungswinkel neu zu bestimmen. Anschließend wechselt er erneut in den Zustand „Distanz-PID“ und setzt den Prozess des Geradeausfahrens fort, bis der Roboter eine Gesamtdistanz von 8m zurückgelegt hat und schließlich anhält. (Das entsprechende Simulink-Programm ist wie Anlage 23)

- **Versuchsdaten und Datenanalyse**

Zur Bewertung der Leistungsfähigkeit der beiden segmentierten PID-Steuerungsprogramme wurden jeweils fünf Versuchsreihen für jede der beiden Strategien durchgeführt. Die experimentellen Daten sind in die Tabellen dargestellt.

Straßenüberquerung					
VG	RKZ[s]	AA	VAWmax[m]	VAWmin[m]	SET[m]
1	8.962	1	1.55	1.39	6.2
2	10.02	1	1.56	1.47	6.8
3	10.18	1	1.55	1.43	6.7
4	9.945	1	1.58	1.42	6.4
5	7.878	1	1.61	1.37	6.3
$\bar{x}$	9.397	1	1.57	1.416	6.48
σ^2	0.94953	0	0.00065	0.00148	0.067

Tabelle 5.7 Versuchsdaten bei der segmentierten PID-Methode (SÜ)

Gästeempfang					
VG	RKZ1[s]	AA1	VAWmax1[m]	VAWmin1[m]	SET1[m]
1	10.7	1	1.57	1.49	6.4
2	9.86	2	1.58	1.37	6.7
3	8.07	1	1.58	1.36	6.7
4	10.11	1	1.58	1.49	6.2
5	10.33	2	1.58	1.36	6.9
$\bar{x}$	9.814	1.4	1.578	1.414	6.58
σ^2	1.04563	0.3	0.00002	0.00483	0.077

Tabelle 5.8 Versuchsdaten bei der segmentierten PID-Methode vor der Wende (GE)

RKZ2[s]	AA2	VAWmax2[m]	VAWmin2[m]	SET2[m]
1.9641	1	1.6	1.52	3.5
1.715	1	1.56	1.5	3.3
2.007	2	1.71	1.57	3.5
2.172	2	1.57	1.38	3.5
1.796	1	1.57	1.43	3.3
1.93082	1.4	1.602	1.48	3.42
0.0324584	0.3	0.00387	0.00565	0.012

Tabelle 5.9 Versuchsdaten bei der segmentierten PID-Methode nach der Wende (GE)

In jeder Versuchsreihe wurden die wesentlichen Leistungskennzahlen erfasst, darunter die Rückkehrzeit, die Anzahl der Anpassungen, die maximale und minimale Distanz nach der Rückkehr sowie die zurückgelegte Strecke entlang der Trajektorie. In dieser Studie wurde zunächst eine experimentelle Bewertung der Roboterbewegungen auf der „SÜ“-Route und der „GE“-Route vor dem Wenden vorgenommen. Beide Routen wurden unter den gleichen Ausgangsbedingungen getestet: Der Roboter startete mit einem Winkel von 15° von einem Punkt, der 0,3m links der vorgegebenen Trajektorie lag.

In der Anfangsphase, nachdem der Roboter im Vorlauf seine Position und den Kursabweichungswinkel bestimmt hatte, bewegte er sich auf beiden Routen gemäß den vom Distanz-PID-Regler berechneten TVs entlang der vorgegebenen Trajektorie. In beiden Versuchsreihen benötigte der Roboter etwa 10s, um zur Trajektorie zurückzukehren.

Nach der Rückkehr zur Trajektorie folgte der Roboter den vom Winkel-PID-Regler berechneten TVs und bewegte sich entlang der Trajektorie. In dieser Phase betrug der durchschnittliche maximale Abstand zur Wand für die „SÜ“-Route 1,562m, der durchschnittliche minimale Abstand 1,412m, und die durchschnittliche maximale Abweichung von der vorgegebenen Trajektorie 0,138 m. Auf der „GE“-Route betrug der durchschnittliche maximale Abstand zur Wand 1,612m, der durchschnittliche minimale Abstand 1,47m, und die maximale Abweichung von der vorgegebenen Trajektorie lag bei 0,08 m. Die gemessene Gesamtdistanz, die der Roboter auf beiden Routen zurücklegte, betrug 6,6 m.

Nach Abschluss dieses Abschnitts stoppte der Roboter in der „SÜ“-Versuchsgruppe, während der Roboter in der „GE“-Versuchsgruppe eine 180°-Drehung gegen den Uhrzeigersinn ausführte und dann weiterfuhr. In dieser Gruppe benötigte der Roboter 1,4s, um nach dem Wenden zur Trajektorie zurückzukehren. Der durchschnittliche maximale und minimale Abstand zur Wand nach der Rückkehr betrug 1,602m bzw. 1,48m und die maximale Abweichung von der vorgegebenen Trajektorie lag bei 0,07 m. Nach der Rückkehr zur Trajektorie legte der Roboter 3,42m entlang der vorgegebenen Trajektorie zurück. Diese Daten zeigen, dass der Roboter in der „GE“-Versuchsgruppe nach der Drehung schnell und präzise zur vorgegebenen Trajektorie zurückkehren und dabei eine hohe Genauigkeit in der Trajekturverfolgung beibehalten konnte.

Die erhaltenen Abstände zur Wand sowie die entsprechenden Indexdaten werden in den entsprechenden Programmen zur Erstellung von Streudiagrammen importiert. (sehe Anlage 28 und Anlage 29)

Nach dem Import der Daten analysiert das Programm den Betriebszustand des Roboters und unterteilt dessen Trajektorie in verschiedene Phasen. Konkret wird die Trajektorie des Roboters in die Abschnitte „Rückkehr“ und „Trajekturverfolgung“ unterteilt. Für die „SÜ“-Route wird jeweils eine Phase der „Rückkehr“ und der „Trajekturverfolgung“ definiert. Im „Verhaltenssteuerungs“-Zustandsautomaten des Straßenüberquerungsprogramms (siehe Abbildung 5.42) wird der Index auf 4 gesetzt, sobald der Zustand „Weiterfahrt“ erreicht ist. Das Programm zur Erstellung von Streudiagrammen sucht nach dem ersten Datensatz mit einem Indexwert von 4 und markiert diesen als den Beginn der „Trajekturverfolgung“. Alle nachfolgenden Daten werden ebenfalls als „Trajekturverfolgung“ gekennzeichnet und im Diagramm mit einer roten Linie dargestellt. Zudem identifiziert das Programm die maximalen und minimalen Abstände in diesem Abschnitt.

Für die „GE“-Route erweitert das Programm das Streudiagramm-Programm der „SÜ“-Route, um den „Trajekturverfolgungs“-Abschnitt nach dem Wenden zu identifizieren. Im „Verhaltenssteuerungs“-Zustandsautomaten der segmentierten PID-Algorithmusstruktur für die „GE“-Route (siehe Abbildung 5.43) wird der Markierungswert h bei einer Wende um 1 erhöht (der Anfangswert ist 0). Da der Roboter nur bei h=0 in den „Wende“-Zustand wechselt, kann h als Indikator dafür verwendet werden, ob eine Wende stattgefunden hat.

Das Programm zur Erstellung des Streudiagramms für die „GE“-Route identifiziert nach dem ersten Auftreten von h=1 den Punkt als „Ende der ersten Trajekturverfolgung“. Sobald es den ersten Datensatz mit h=1 und einem Indexwert index= 4 findet, markiert es alle nachfolgenden Daten als „zweite Trajekturverfolgung“. Schließlich werden die beiden „Trajekturverfolgungs“-Abschnitte im Diagramm dargestellt, und das Programm identifiziert die maximalen und minimalen vertikalen Abstände zur Wand in beiden Abschnitten.

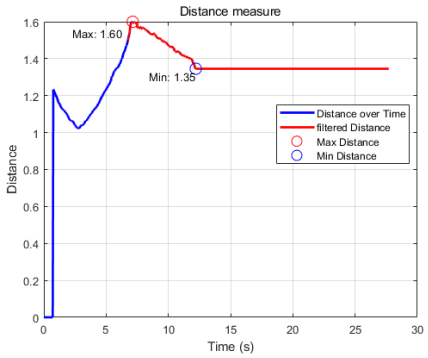


Abbildung 5.44 VAW bei der segmentierten PID-Methode (SÜ)

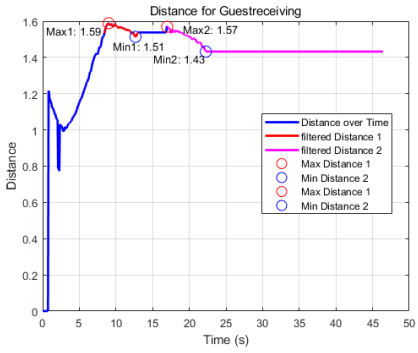


Abbildung 5.45 VAW bei der segmentierten PID-Methode (GE)

Wie in Abbildung 5.44 für die „SÜ“-Route dargestellt, beginnt der Roboter nach Abschluss der anfänglichen Parameterberechnungsphase mit der Rückmeldung des vertikalen Abstands zur Wand. Unter der Kontrolle des segmentierten PID-Programms verringert sich der Abstand zunächst von 1,2m auf etwa 1m und steigt dann allmählich auf die vorgegebene Trajektorie von 1,55m an. Aufgrund von Fehlern bei der Winkelsteuerung nimmt der vertikale Abstand zwar leicht ab, bleibt jedoch dank der Begrenzungen des Programms weiterhin in der Nähe der Trajektorie. Nach etwa 12s stoppt der Roboter und der vertikale Abstand stabilisiert sich bei 1,35 m.

In der „GE“-Route, wie in Abbildung 5.45 dargestellt, zeigt der Winkel-PID-Regler des Roboters eine noch bessere Steuerungsleistung als im Experiment zur „SÜ“-Route. Nach der Rückkehr zur Trajektorie verringert sich der Abstand von 1,59m auf 1,51m, und nach dem Wenden von 1,57m auf 1,43m, wobei die maximale Abweichung nicht mehr als 0,14m beträgt.

Die beiden Experimentreihen zeigen, dass der Roboter in den „Trajekturverfolgungs“-Abschnitten eine hohe Stabilität und Genauigkeit aufrechterhalten kann, wobei die Trajektorie zwischen 1,41m und 1,61m schwankt. Allerdings beträgt die Rückkehrzeit zur Trajektorie auf der „SÜ“-Route sowie die erste Rückkehrzeit zur Trajektorie auf der „GE“-Route etwa 10s, was relativ lang ist. Im Vergleich dazu ist die zweite Rückkehrzeit auf der „GE“-Route deutlich kürzer, nämlich etwa 1,4s, was hauptsächlich auf die hohe Stabilität und Genauigkeit zurückzuführen ist, die der Roboter im ersten Abschnitt der „Trajekturverfolgung“ zeigt.

5.2.3 Vergleich der Methoden

Sowohl die „Stufenschwächung“-Methode als auch die „segmentierte PID“-Methode ermöglichen es dem Roboter, der vorgegebenen Trajektorie zu folgen. In diesem Kapitel werden die beiden Methoden anhand der experimentellen Daten verglichen, um die beste auszuwählen, die anschließend für den Betrieb des gesamten Systems, bestehend aus Roboter und Dummy, eingesetzt wird.

Es wird die Methode der gewichteten Analyse verwendet. Dabei machen die Gesamtergebnisse der „SÜ“ -Route 40% und die der „GE“-Route 60% der Bewertung aus. Für die beide Route gelten die Bewertungskriterien in der Anlage 24.

Wie in Anlage 24 gezeigt, sieht die gewichtete Analyse unterschiedliche Kriterien für den Roboter vor, je nachdem, ob er sich vor oder nach dem Wenden befindet. Da sich der Roboter nach dem Wenden bereits nahe der vorgegebenen Trajektorie befindet, benötigt er weniger Zeit, um zur Trajektorie zurückzukehren. Daher werden unterschiedliche Bewertungskriterien für die beiden Phasen angewendet. „Gästeempfang 1“ bezieht sich auf die Bewertung der Roboterleistung vor dem Wenden, während „Gästeempfang 2“ die Bewertung nach dem Wenden beschreibt.

Auf Grundlage dieser Bewertungsmethode sowie der Daten in den Tabelle 5.2, Tabelle 5.5, Tabelle 5.6 wurden die jeweiligen Bewertungen der „Stufenschwächung“-Methode für beide Routen ermittelt.

Stufenschwächungsmethode						
SÜ	RKZ[s]	AA	VAWmax[m]	VAWmin[m]	SET[m]	Sum
$\bar{x}$	6	6	3	3	6	57
σ^2	9	6	9	6	3	
GE1	RKZ1[s]	AA1	VAWmax1[m]	VAWmin1[m]	SET1[m]	Sum
$\bar{x}$	6	6	3	3	9	108
σ^2	3	9	9	3	6	
GE2	RKZ2[s]	AA2	VAWmax2[m]	VAWmin2[m]	SET2[m]	
$\bar{x}$	6	6	6	3	6	
σ^2	3	9	6	3	3	

Tabelle 5.10 Die Bewertung der SSM

Die Endbewertung der „Stufenschwächung“-Methode wird gemäß der gewichteten Bewertungsregel anhand der folgenden Formel berechnet:

$$E_{Stuf} = 40\% * \sum_{1}^{n} P_{S\ddot{U}} + 60\% * \sum_{1}^{n} P_{GE} \tag{5.13}$$

$$E_{Stuf} = 40\% * 57 + 60\% * 108 = 87{,}6 \tag{5.14}$$

Der „Stufenschwächung“-Methode wurde eine Endbewertung von 87,6 zugewiesen.

Auf Grundlage der Tabelle 5.7, Tabelle 5.8 und Tabelle 5.9 wurden die jeweiligen Bewertungen der „segmentierten PID“-Methode für beide Routen ermittelt.

segmentierten PID-Methode						
SÜ	RKZ[s]	AA	VAWmax[m]	VAWmin[m]	SET[m]	Sum
$\bar{x}$	3	6	9	3	3	63
σ^2	6	9	9	6	9	
GE1	RKZ1[s]	AA1	VAWmax1[m]	VAWmin1[m]	SET1[m]	Sum
$\bar{x}$	3	9	3	3	3	123
σ^2	3	9	9	6	9	
GE2	RKZ2[s]	AA2	VAWmax2[m]	VAWmin2[m]	SET2[m]	
$\bar{x}$	9	6	6	3	9	
σ^2	9	9	6	3	6	

Tabelle 5.11 Die Bewertung der segmentierten PID-Methode

Ebenso wird die Endbewertung der „segmentierten PID“-Methode gemäß der gewichteten Bewertungsregel nach folgender Formel berechnet:

$$E_{seg} = 40\% * 63 + 60\% * 123 = 99 \tag{5.15}$$

Die „segmentierte PID“-Methode erzielte eine Endbewertung von 99.

Auf Grundlage der Berechnungsergebnisse lässt sich feststellen, dass die „segmentierte PID“-Methode eine höhere Gesamtbewertung als die „Stufenschwächung“-Methode erreicht hat. Daher wird die „segmentierte PID“-Methode für die weitere Steuerung des Gesamtsystems, bestehend aus Dummy und Roboter, eingesetzt.

5.3 Die Verbindung zwischen Dummy und Roboter

5.3.1 Entwurf des Verbindungskonzepts

Das ultimative Ziel dieses Experiments besteht darin, eine synchronisierte Bewegung zwischen dem Dummy und dem Roboter zu realisieren, sodass der Roboter den Dummy in einer koordinierten Bewegung transportieren kann. Der zentrale Ansatz dieser Untersuchung ist die Konstruktion und Implementierung einer mechanischen Struktur, die es dem Dummy ermöglicht, in einer stabilen, vertikalen Position auf der oberen Plattform des Roboters zu stehen. Dadurch wird gewährleistet, dass der Dummy während der Bewegung des Roboters sich koordiniert mit dem Roboter bewegt.

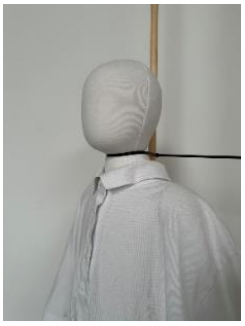


Abbildung 5.46 Befestigung des Dummy-Kopfs

Abbildung 5.47 Befestigung der Dummy-Beine

Abbildung 5.48 Verbindungs- und Stützsäule

Im Experiment wurde ein strukturiertes Stützsystem entwickelt, um eine stabile Verbindung zwischen dem Dummy und dem Arlo-Roboter sicherzustellen. Dieses System verwendet eine 1,5m lange und 15mm im Durchmesser messende runde Holzstange als Verbindungselement. Der Dummy-Kopf und die Dummy-Beine wird mithilfe von Kabelbindern und einem Stoffgurt fest an der Holzstange fixiern, um eine stabile Verbindung zu gewährleisten. Am unteren Ende der Holzstange werden etwa 7cm freigelassen, um durch ein Loch mit einem Durchmesser von 15mm zu passen, das sich in der Nähe des Zentrums der oberen Plattform des Roboters befindet (siehe Abbildung 5.46, Abbildung 5.47 und Abbildung 5.48).

Zur weiteren Erhöhung der Stabilität zwischen dem Holzstab und dem Roboter wird unterhalb des Lochs eine blaue Stütze installiert. Diese Stütze bietet zusätzliche Unterstützung, indem der

freigehaltene Teil des Stabs in die innere Öffnung der Stütze eingeführt und mit einer Zylinderkopfschraube fixiert wird, was sicherstellt, dass der Holzstab während der Bewegung des Dummys nicht verrutscht oder sich lockert.

Zusätzlich befinden sich an den vier Ecken der Plattform vier dünne Stützen. Zwei weiße Stützen sind durch Öffnungen an ihren oberen und unteren Enden zwischen der oberen und unteren Plattform des Roboters befestigt, was die dreiachsige Bewegungsfreiheit der oberen Plattform einschränkt und zusätzliche Stabilität gewährleistet. Zwei rote Stützen sind lediglich mit der oberen Plattform des Roboters verbunden, wobei ihre unteren Enden auf der unteren Plattform aufliegen, was die Tragfähigkeit der Plattform und die Gesamtstabilität weiter erhöht.

Das Gesamtergebnis ist in der Anlage 25 dargestellt. Die zeigt, dass der Dummy stabil auf der oberen Plattform des Roboters steht. Basierend auf den in der Analyse der Vergleichsmethoden (Kapitel 5.2.3) gewonnenen Ergebnisse wird der Roboter nun mit der segmentierten PID-Steuerungsmethode weiterentwickelt und optimiert. Da die Masse des Dummys und der Verbindungsstruktur im Experiment einen signifikanten Einfluss auf die Systemleistung haben, ist es erforderlich, die PID-Verstärkung im segmentierten PID-Steuerungsprogramm anzupassen. Durch eine erneute Feinabstimmung dieser Verstärkung kann die durch Masseeinflüsse bedingte Veränderung der Systemreaktion effektiv kompensiert werden, sodass das Robotersystem während der Bewegung weiterhin stabil und präzise arbeitet.

Für die Distanzregelung auf der „SÜ“-Route wurden die PID-Parameter wie folgt aktualisiert: Die Proportionalverstärkung (K_p) wurde auf 0,083, die Integralverstärkung (K_i) auf 0 und die Differentialverstärkung (K_d) auf 0,76 gesetzt. Hinsichtlich des Winkel-PID-Reglers wurden sowohl auf der „SÜ“-Route als auch auf der GE-Route identische Anpassungen vorgenommen. Die Parameter für die Regelung bei kleinen Winkeln wurden auf K_p=0,027, K_i=0 und K_d=0,00016 aktualisiert, während die Parameter für Winkel größer als 90° unverändert blieben.

Auf der GE-Route wählt der Distanz-PID-Regler unterschiedliche Steuerungsparameter je nachdem, ob der Roboter die Wende abgeschlossen hat oder nicht. Wenn h≤0 ist (dies ist im Experiment nur bei h=0 der Fall), bedeutet dies, dass der Roboter die Wende noch nicht abgeschlossen hat. In diesem Fall wurden die PID-Parameter auf K_p=0,083, K_i=0 und K_d=0,76 gesetzt. Wenn h>0 ist (im Experiment nur h=1, was anzeigt, dass der Roboter die Wende abgeschlossen hat), wurden die PID-Parameter auf K_p=0,3, K_i=0 und K_d=0,72 gesetzt. Durch die Erhöhung der Proportionalverstärkung und eine leichte Reduzierung der Differentialverstärkung wird sichergestellt, dass der Roboter schnell und präzise zur vorgegebenen Trajektorie zurückkehrt. (Das Gesamtprogramm ist wie Anlage 30, Anlage 31, Anlage 32 und Anlage 33.)

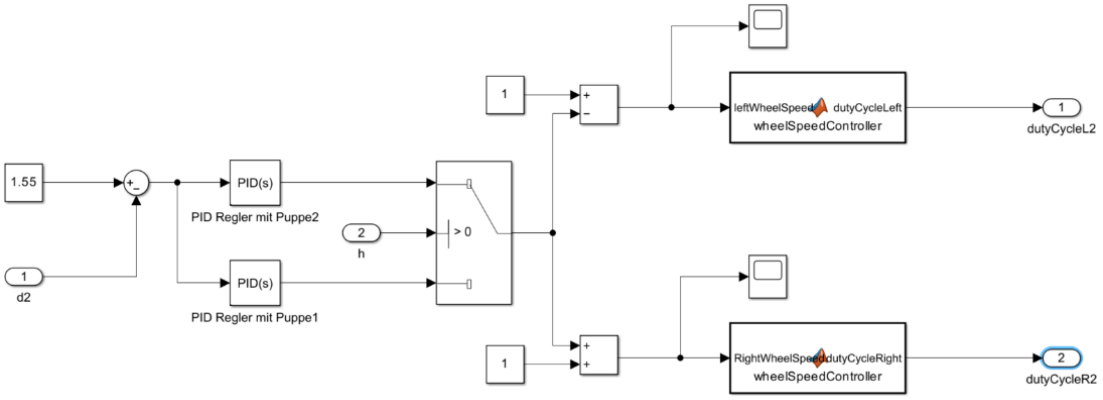


Abbildung 5.49 Distanz-PID-Regler (mit Dummy, „GE“-Route)

5.3.2 Bewertung der Versuchsdaten

Der Dummy wird auf der rechten Seite der vorgegebenen Trajektorie in einem Abstand von 0,3m zur Trajektorie positioniert und so ausgerichtet, dass seine Bewegungsrichtung entlang der 15°-Markierungslinie verläuft (siehe Abbildung 2.1). Anschließend wird das Programm gestartet und jeweils fünf Experimente sowohl für die „SÜ“-Route als auch für die GE-Route durchgeführt. Nachfolgend werden die Ergebnisse der Experimente zur „SÜ“-Route präsentiert:

Straßenüberquerung					
VG	RKZ[s]	AA	VAWmax[m]	VAWmin[m]	SET[m]
1	8.962	1	1.55	1.39	6.2
2	10.02	1	1.56	1.47	6.8
3	10.18	1	1.55	1.43	6.7
4	9.945	1	1.58	1.42	6.4
5	7.878	1	1.61	1.37	6.3
$\bar{x}$	9.397	1	1.57	1.416	6.48
σ^2	0.94953	0	0.00065	0.00148	0.067

Tabelle 5.12 Versuchsergebnisse der segmentierten PID-Methode (mit Dummy, SÜ)

Basierend auf den in der Tabelle dargestellten experimentellen Daten lässt sich die „SÜ“-Route aus verschiedenen Perspektiven analysieren:

1. Rückkehrzeit:

Die Rückkehrzeiten der Versuchsgruppen 1 bis 5 variieren zwischen 8,962s und 10,18s, mit einem Mittelwert von 9,397s und einer Varianz von 0,949527. Dies weist darauf hin, dass die Rückkehrzeiten zwischen den einzelnen Experimenten gewisse Schwankungen aufweisen, insgesamt jedoch relativ stabil bleiben.

2. Anzahl der Anpassungen:

In allen Versuchsgruppen wurde jeweils nur eine Anpassung registriert. Dies deutet darauf hin, dass der Roboter unter den gegebenen Bedingungen konsistent in der Lage ist, nach einer einzigen Anpassung den Weg zur vorgegebenen Trajektorie erfolgreich zurückzufinden.

3. Maximaler und minimaler Abstand nach der Rückkehr:

Der durchschnittliche maximale vertikale Abstand beträgt 1,57m, der durchschnittliche minimale vertikale Abstand 1,416m. Die Varianz des maximalen Abstands liegt bei 0,00065, die des minimalen Abstands bei 0,00148. Diese Werte zeigen, dass die Abstandsregelung nach der Rückkehr zur Trajektorie relativ stabil ist und nur geringe Abweichungen aufweist.

4. Entlang der Trajektorie zurückgelegte Strecke：

Die durchschnittlich zurückgelegte Strecke beträgt 6,48m, was das Ziel von 5m um etwa 24,7% übertrifft, jedoch im akzeptablen Bereich liegt. Die Varianz beträgt 0,067, was darauf hindeutet, dass die zurückgelegte Strecke zwischen den einzelnen Experimenten relativ konsistent ist und nur geringe Schwankungen aufweist.

Gästeempfang					
VG	RKZ1[s]	AA1	VAWmax1[m]	VAWmin1[m]	SET1[m]
1	10.7	1	1.57	1.49	6.4
2	9.86	2	1.58	1.37	6.7
3	8.07	1	1.58	1.36	6.7
4	10.11	1	1.58	1.49	6.2
5	10.33	2	1.58	1.36	6.9
$\bar{x}$	9.814	1.4	1.578	1.414	6.58
σ^2	1.04563	0.3	0.00002	0.00483	0.077

Tabelle 5.13 Versuchsergebnisse der segmentierten PID-Methode (GE, vor der Wende, mit Dummy)

RKZ2[s]	AA2	VAWmax2[m]	VAWmin2[m]	SET2[m]
1.438	1	1.58	1.54	3.3
2.994	1	1.58	1.41	3.4
4.667	1	1.58	1.46	3.7
0.964	1	1.58	1.55	3.8
4.12	1	1.58	1.53	3.4
2.8366	1	1.58	1.498	3.52
2.6212418	0	0	0.00367	0.047

Tabelle 5.14 Versuchsergebnisse der segmentierten PID-Methode (GE, nach der Wende, mit Dummy)

Basierend auf den experimentellen Daten zur GE-Route lässt sich folgende Analyse durchführen:

1. Rückkehrzeit:：

Erste Rückkehrzeit: Die Rückkehrzeiten der Versuchsgruppen 1 bis 5 variieren zwischen 8,07s und 10,7s, mit einem Mittelwert von 9,814s und einer Varianz von 1,046. Dies weist darauf hin, dass es bei der ersten Rückkehr zur Trajektorie auf der GE-Route gewisse Unterschiede in den Rückkehrzeiten gibt, diese jedoch insgesamt um die 10s liegen und somit relativ stabil sind.

Zweite Rückkehrzeit: Die Rückkehrzeiten liegen zwischen 0,964s und 4,667s, mit einem Mittelwert von 2,84s und einer Varianz von 2,621. Die größere Schwankung der zweiten Rückkehrzeiten zeigt, dass in dieser Phase die Leistungen der einzelnen Versuchsgruppen weniger konsistent sind, was wahrscheinlich auf die variierenden Abstände des Roboters zur Wand nach dem Wenden zurückzuführen ist; bei größeren Abständen dauert die Rückkehr zur Trajektorie länger.

2. Anzahl der Anpassungen:

Anpassung bevor der Wende: Die Anzahl der Anpassungen variiert zwischen 1 und 2, mit einem Mittelwert von 1,4 und einer Varianz von 0,3. Dies zeigt, dass es bei der ersten Rückkehr zur

Trajektorie leichte Unterschiede in der Anzahl der erforderlichen Anpassungen gibt, in den meisten Fällen jedoch nur eine Anpassung notwendig ist.

Anpassung nach der Wende: Bei der zweiten Rückkehr zur Trajektorie war in allen Versuchsgruppen nur eine Anpassung erforderlich, was darauf hindeutet, dass der Anpassungsbedarf in dieser Phase relativ einfach und konsistent ist.

3. Maximaler und minimaler vertikaler Abstand nach der Rückkehr:

Maximaler vertikaler Abstand nach der ersten Rückkehr: Die VAW_{max} nach der ersten Rückkehr der Versuchsgruppen liegen zwischen 1,57m und 1,58m, mit einem Mittelwert von 1,578m und einer Varianz von 0,000022. Dies zeigt, dass die maximalen Abstände nach der ersten Rückkehr nahezu identisch sind.

Minimaler vertikaler Abstand nach der ersten Rückkehr: Die VAW_{min} nach der ersten Rückkehr variieren zwischen 1,36m und 1,49m, mit einem Mittelwert von 1,414m und einer Varianz von 0,00483. Obwohl leichte Unterschiede bei den minimalen Abständen bestehen, zeigt der Roboter insgesamt eine stabile Kontrolle der minimalen Abweichungen.

Maximaler vertikaler Abstand nach der zweiten Rückkehr: In allen Versuchsgruppen beträgt der VAW_{max} nach der zweiten Rückkehr 1,58m, was auf eine konsistente und stabile Leistung des Roboters hindeutet.

Minimaler Abstand nach der zweiten Rückkehr: Die $VAW_{min}2$ liegen zwischen 1,41m und 1,55m, mit einem Mittelwert von 1,498m und einer Varianz von 0,00367. Dies zeigt, dass die VAW_{min} nach der zweiten Rückkehr leichte Unterschiede aufweisen, insgesamt jedoch stabil bleiben.

4. Entlang der Trajektorie zurückgelegte Strecke:

Erste zurückgelegte Strecke: Die zurückgelegten Strecken variieren zwischen 6,2m und 6,9m, mit einem Mittelwert von 6,58m und einer Varianz von 0,077m. Dies zeigt, dass die Steuerung der zurückgelegten Strecke in der ersten Phase relativ konsistent ist. Obwohl die zurückgelegte Strecke das Ziel von 5m um etwa 26% übertrifft, liegt sie dennoch im akzeptablen Bereich.

Zweite zurückgelegte Strecke: Die zurückgelegten Strecken variieren zwischen 3,3m und 3,8m, mit einem Mittelwert von 3,52m und einer Varianz von 0,047m. Diese zweite Strecke liegt näher am Ziel von 3m, mit einer durchschnittlichen Überschreitung von nur 0,52m (etwa 17%) und zeigt insgesamt eine stabile Leistung.

Die im Experiment gemessenen vertikalen Abstände des Roboters zur Wand werden in einem Streudiagramm dargestellt:

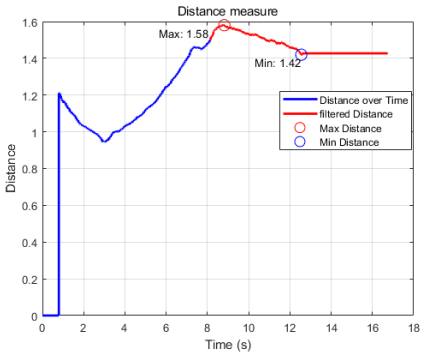


Abbildung 5.50 VAW-Streudiagramm (SÜ, mit Dummy)

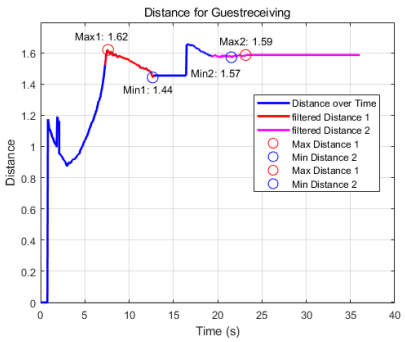


Abbildung 5.51 VAW-Streudiagramm (GE, mit Dummy)

In der „SÜ“-Route, wie in Abbildung 5.50 dargestellt, beginnt der Roboter nach Abschluss der anfänglichen Parameterberechnungsphase damit, den VAW zu erfassen. Basierend auf den in Abbildung 3.3 dargestellten Prinzipien ist es aufgrund der zwei angetriebenen Räder des Roboters unvermeidlich, dass der Roboter bei der Anpassung seiner Position und Fahrtrichtung mithilfe des Distanz-PID-Reglers entlang einer Kurve fährt. Der VAW verringert sich zunächst von 1,2m auf etwa 0,9m, bevor er allmählich auf den vorgegebenen Wert von 1,55m zurückkehrt. Während der anschließenden Steuerung durch den Winkel-PID-Regler führt ein kleiner Winkelberechnungsfehler dazu, dass die Ausrichtung des Roboters nicht exakt mit der vorgegebenen Trajektorie übereinstimmt, was eine leichte Verringerung des Abstands zur Folge hat. Dennoch bleibt der Roboter nahe der vorgegebenen Trajektorie, und der Abstand bleibt im akzeptablen Bereich. Nach etwa 12s stoppt der Roboter, und der VAW stabilisiert sich bei 1,42m.

In der GE-Route, wie in Abbildung 5.51 Abbildung 5.51gezeigt, trat beim Dummy während der Positionsanpassung durch den PID-Regler ein Messfehler auf. Der VAW sprang innerhalb von etwa 2s von 1m auf 1,2m. Dieser Fehler beeinträchtigte jedoch nicht die nachfolgende Bewegung des Roboters, der nach 7,3s erfolgreich zur vorgegebenen Trajektorie zurückkehrte. Nach der Steuerung durch den Winkel-PID-Regler führte der Roboter eine Wende durch und maß den vertikalen Abstand zur gegenüberliegenden Wand. Der Distanz-PID-Regler wurde erneut aktiviert, und der Abstand verringerte sich allmählich von etwa 1,65m auf 1,55m. Während der anschließenden Steuerung durch den Winkel-PID-Regler blieb die Fahrtrichtung des Roboters weitgehend stabil und deckte sich mit der vorgegebenen Trajektorie, sodass sich der VAW kaum veränderte.

Der Roboter zeigte sowohl in der SÜ- als auch in der GE-Route, dass er mithilfe des segmentierten PID-Programms schnell zur vorgegebenen Trajektorie zurückkehren konnte. Selbst bei geringfügigen Messfehlern blieb der Roboter in der Lage, wie gefordert zur Trajektorie zurückzufinden, was auf die hohe Störfestigkeit des Systems hinweist.

Die gesammelten Daten belegen, dass der Roboter bei der Ausführung der „SÜ“-Route insgesamt stabil arbeitet, insbesondere in Bezug auf die AA, die Kontrolle der VAW nach der Rückkehr sowie die SET. Obwohl die RKZ zwischen den einzelnen Experimenten leicht variierten, blieben die Schwankungen insgesamt gering, was auf eine gute Vorhersehbarkeit und Zuverlässigkeit des Systems in Bezug auf die Zeitsteuerung hinweist. In der GE-Route zeigte der Roboter während der ersten Rückkehrphase eine konsistente Leistung, mit relativ stabilen RKZ

und einer geringen AA. Obwohl die RKZ in der zweiten Phase größere Schwankungen aufwiesen, blieb die Leistung des Roboters bei der Anpassung und der Abstandsregelung stabil. Insgesamt erfüllte der Roboter in beiden Routen die Erwartungen, auch wenn in Bezug auf die Kontrolle der SET entlang der Trajektorie noch Verbesserungspotenzial besteht.

5.3.3 Beschreibung des Experimentes auf dem Prüffeld

Das ultimative Ziel dieses Projekts besteht darin, den Roboter so zu befähigen, dass er an Fahrzeugtests auf dem Prüffeld teilnehmen kann. Dies bedeutet, dass das kombinierte System aus Dummy und Roboter sowie die dazugehörigen Programme in der Lage sein müssen, sich an die Umgebungsbedingungen des Prüffelds anzupassen. Die experimentellen Ergebnisse zeigten jedoch, dass der Roboter nicht in der Lage war, der vorgegebenen Route gemäß dem Programm zu folgen und ständig eine Tendenz aufwies, nach links von der vorgesehenen Trajektorie abzuweichen. Nach einer gründlichen Untersuchung wurde festgestellt, dass das Problem vom Ultraschallsensor herrührt.

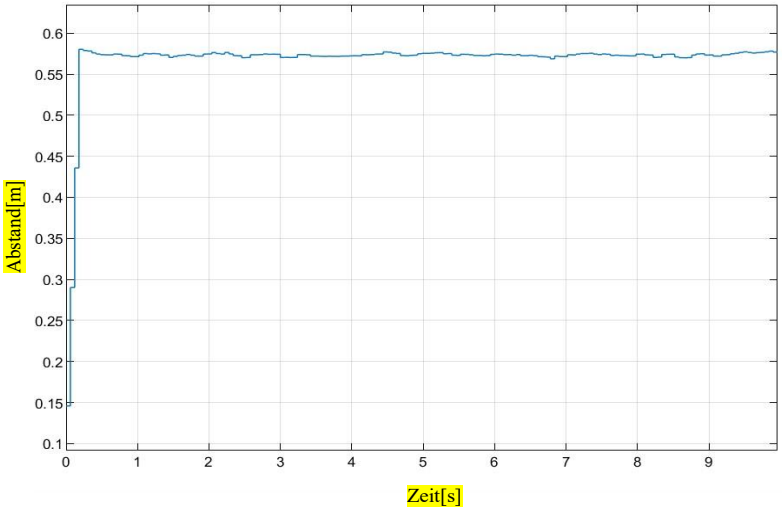


Abbildung 5.52 Messwerte des Ultraschallsensors für eine Entfernung von 0,6m (auf dem Prüffeld)

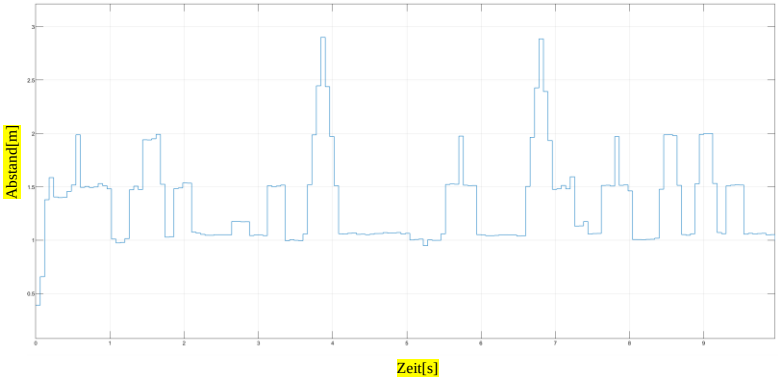


Abbildung 5.53 Messwerte des Ultraschallsensors für eine Entfernung von 3m (auf dem Prüffeld)

Im Experiment wurden zudem die Messergebnisse des Ultraschallsensors für andere Entfernungen getestet. Die entsprechenden Ergebnisse sind in Anlage 34 dokumentiert.

Die Analyse der Diagrammdaten zeigt, dass der Ultraschallsensor im Bereich von 0 bis 0,9 m die Entfernung zwischen dem Roboter und der Wand präzise messen kann. Sobald jedoch dieser Bereich überschritten wird, nimmt die Messabweichung zwischen dem Ultraschallsensor und der tatsächlichen Entfernung signifikant zu und zeigt eine hohe Schwankungsbreite. Diese Schwankungen verstärken sich mit zunehmender Entfernung des Roboters zur Wand. Wenn der Roboter eine Entfernung von 3 m zur Wand erreicht, übersteigt die Messschwankung sogar 1,3 m. Offensichtlich ist eine solche Messgenauigkeit unzureichend, um eine Echtzeitsteuerung des Roboters zu unterstützen. Daher wird bei der Anwendung des Roboters im Außenbereich (auf dem Prüffeld) der Ultraschallsensor nicht mehr als bevorzugte Option verwendet; stattdessen wird hauptsächlich auf das Gyroskop zurückgegriffen, um die Bewegungsrichtung des Roboters zu steuern. Das spezifische Steuerungsprogramm ist in der folgenden Abbildung dargestellt.

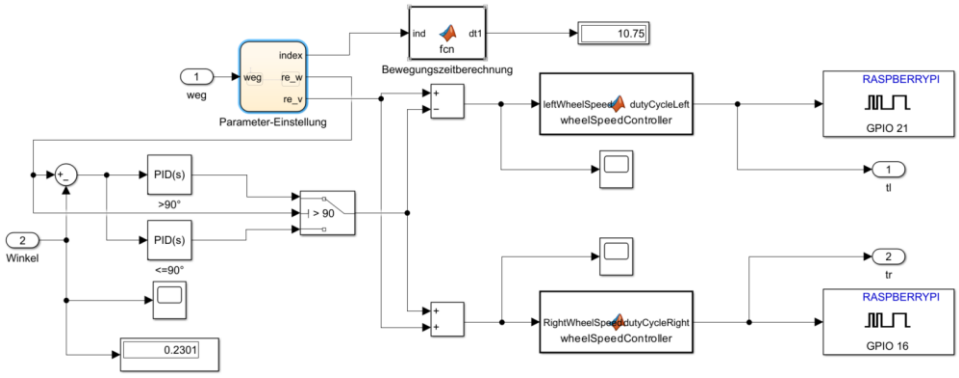


Abbildung 5.54 Programm des kombinierten Systems aus Dummy und Roboter (auf dem Prüffeld)

Das Programm passt die Richtung des Roboters hauptsächlich durch eine Winkel-PID-Steuerung an, wobei die Proportional-, Integral- und Differentialverstärkung des PID-Reglers den Einstellungen in Kapitel 3.4.1 entsprechen. Konkret wird bei einer Differenz zwischen dem tatsächlichen und dem Referenzwinkel von weniger als oder gleich 90° die Proportionalitätsverstärkung K_p auf 0,034, die Integralverstärkung K_i auf 0 und die Differentialverstärkung K_d auf 0,00015 eingestellt. Bei einer Winkelabweichung von mehr als 90° wird K_p auf 0,5, K_i auf 0,0003 und K_d auf 0,006 gesetzt. Die Einstellungen des Referenzwinkels und der Referenzdrehzahl des Motors während der Bewegung des Roboters werden durch den Zustandsautomaten „Parameter-Einstellung“ vorgenommen. Die Unterschiede in den Programmen für die Überquerungsroute und die Gästeempfangsroute werden ebenfalls in diesem Zustandsautomaten berücksichtigt.

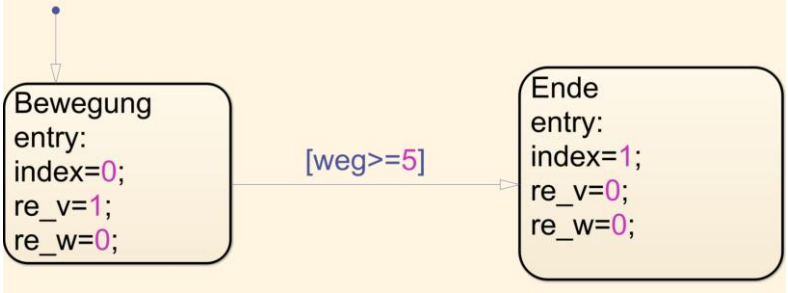


Abbildung 5.55 Einstellung der Referenzparameter für „SÜ“

Der Zustandsautomat umfasst zwei Zustände: „Bewegung“ und „Ende“. Wenn der Roboter erkennt, dass die zurückgelegte Strecke 5m erreicht hat, wechselt der Zustandsautomat vom Zustand „Bewegung“ in den Zustand „Ende“. Während der Bewegung fährt der Roboter stets geradeaus, daher wird in den Zuständen „Bewegung“ und „Ende“ der Referenzwinkel jeweils auf 0° gesetzt. Sobald der Roboter die 5m zurückgelegt hat, stoppt er, weshalb die Referenzgeschwindigkeit im Zustand „Bewegung“ auf 1 r/s und im Zustand „Ende“ auf 0 r/s eingestellt wird.

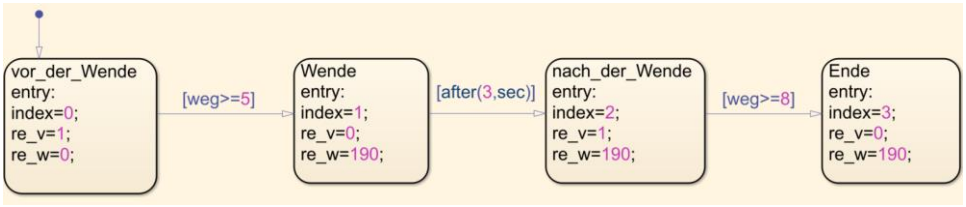


Abbildung 5.56 Einstellung der Referenzparameter für „GE“

In der Gästeempfangsroute umfasst die Bewegung des Roboters vier Phasen, nämlich die Zustände „vor der Wende“, „Wende“, „nach der Wende“ und „Ende“. Der Zustand „vor der Wende“ entspricht dem Zustand „Bewegung“ in der „SÜ“-Route. Nachdem der Roboter in diesem Zustand 5m zurückgelegt hat, wechselt er in den Zustand „Wende“. Während des Wendemanövers sollten theoretisch beide Räder auf einen Referenzwinkel von 180° eingestellt werden. Aufgrund äußerer Einflüsse muss der Referenzwinkel in der Praxis jedoch auf 190° gesetzt werden, um sicherzustellen, dass der Roboter auf dem Prüffeld tatsächlich eine 180°-Wende vollzieht. In den nachfolgenden Zuständen bleibt der Referenzwinkel weiterhin bei 190°.

Um sicherzustellen, dass der Roboter die Wende an Ort und Stelle durchführen kann, wird im Zustand „Wende“ die Referenzgeschwindigkeit auf 0 gesetzt. Nach einer Dauer von 3s in diesem Zustand (um sicherzustellen, dass die Wende abgeschlossen ist), wechselt der Roboter in den Zustand „nach der Wende“. In diesem Zustand beginnt der Roboter, in entgegengesetzter Richtung zu fahren, wobei die Referenzgeschwindigkeit auf 1 r/s eingestellt ist. Sobald die zurückgelegte Strecke 8m erreicht, stoppt der Roboter und die Referenzgeschwindigkeit wird auf 0 r/s gesetzt.

Es ist wichtig zu beachten, dass aufgrund des Fehlens einer Positionierungsfunktion durch den Ultraschallsensor die Anfangsrichtung und -position des Roboters vor Beginn des Experiments (siehe Anlage 35) manuell festgelegt werden müssen. In diesem Experiment wird durch das Ausrichten der Außenseiten der linken und rechten Räder des Roboters an zwei weißen Linien

(siehe Anlage 36) sichergestellt, dass die Fahrtrichtung und die Position des Roboters den vorgegebenen Anforderungen entsprechen. Diese Methode trägt dazu bei, die Genauigkeit und Wiederholbarkeit des Experiments zu gewährleisten.

5.3.4 Versuchsdatenanalyse

Auf dem Prüffeld wurden für beide Routen jeweils fünf Experimente durchgeführt, bei denen die folgenden Versuchsdaten erfasst wurden.

Straßenüberquerung		
VG	SZ[s]	SET[m]
1	10.87	5.29
2	10.86	5.24
3	10.83	5.25
4	10.79	5.19
5	10.75	5.23
$\bar{x}$	10.82	5.24
σ^2	0.0025	0.0013

Tabelle 5.15 Versuchsdaten der „SÜ“-Route (auf dem Prüffeld)

Tabelle 33 zeigt die experimentellen Ergebnisse des kombinierten Systems aus Roboter und Dummy bei der „SÜ“-Route und konzentriert sich dabei auf zwei Hauptindikatoren: die Steuerungszeit (kurz SZ) und die tatsächlich zurückgelegte Strecke entlang der Trajektorie. Die Ergebnisse zeigen, dass die durchschnittliche Steuerungszeit des Systems für das Absolvieren der Route 10,82s beträgt, mit einer Varianz von 0,0025, was auf eine geringe Schwankung der Zeitdaten und eine stabile Systemleistung hinweist. Die durchschnittliche zurückgelegte Strecke des Systems beträgt 5,24m, mit einer Varianz von 0,0013. Obwohl diese Strecke leicht über dem Zielwert von 5m liegt, spiegelt sie insgesamt eine hohe Genauigkeit in der Streckenkontrolle wider.

Gästeempfang				
VG	SZ1[s]	SET1[m]	SZ2[s]	SET2[m]
1	10.74	5.39	7.873	3.22
2	10.76	5.3	7.96	3.19
3	10.78	5.36	8.006	3.24
4	10.75	5.41	7.985	3.3
5	10.77	5.4	7.858	3.25
$\bar{x}$	10.76	5.372	7.9364	3.24
σ^2	0.00025	0.00197	0.0044823	0.00165

Tabelle 5.16 Versuchsdaten der „GE“-Route (auf dem Prüffeld)

Die Tabelle zeigt die experimentellen Daten des kombinierten Systems aus Roboter und Dummy bei der „GE“-Route, unterteilt in die Steuerungszeit und die tatsächlich zurückgelegte Strecke vor (1) und nach (2) der Wende.

Vor der Wende beträgt die durchschnittliche Steuerungszeit des Systems 10,76s, mit einer Varianz von 0,00025, was auf eine geringe Schwankung und eine relative Stabilität des Systems

in dieser Phase hinweist. Die durchschnittlich zurückgelegte Strecke liegt bei 5,375m, mit einer Varianz von 0,00197, was geringfügig über der angestrebten Zielstrecke von 5m liegt, jedoch eine insgesamt hohe Präzision in der Streckenkontrolle aufzeigt.

Nach der Wende beträgt die durchschnittliche Steuerungszeit des Systems etwa 7,94s, mit einer Varianz von 0,0045, was im Vergleich zur Phase vor der Wende eine leichte Abnahme der Stabilität der Zeitdaten anzeigt. Die durchschnittlich zurückgelegte Strecke nach der Wende beträgt 3,24m, mit einer Varianz von 0,00165, während die Zielstrecke 3m beträgt. Dies zeigt, dass, obwohl eine leichte Abweichung besteht, das System dennoch eine hohe Genauigkeit in der Streckenkontrolle beibehält.

Die Analyse der experimentellen Daten zeigt, dass das System auf beiden Routen die jeweiligen Fahraufgaben in jeder Phase präzise ausführt und die Daten in jedem Experiment innerhalb eines bestimmten Rahmens bleiben, was auf eine hohe Stabilität hinweist. Dies spiegelt die gute Anpassungsfähigkeit des Systems an die Umgebungsbedingungen des Prüffelds wider und unterstützt seine Anwendung in Fahrzeugtests erheblich.

6 Problemanalyse und Lösungsvorschlag

Der Nullpunktdrift eines Gyroskops stellt eine häufige Fehlerquelle dar und hat einen erheblichen negativen Einfluss auf die Genauigkeit von Messsystemen. Unter Nullpunktdrift versteht man das Phänomen, bei dem das Ausgangssignal des Gyroskops ohne externe Drehbewegung im Laufe der Zeit abweicht. Diese Abweichung wird hauptsächlich durch Faktoren wie Temperaturänderungen im Inneren des Gyroskops, mechanischen Stress, elektronisches Rauschen sowie durch Unregelmäßigkeiten im Herstellungsprozess verursacht.

Der durch Nullpunktdrift verursachte Messfehler tritt in der Regel als kumulativer Fehler auf. Da das Ausgangssignal des Gyroskops integriert wird, um Winkel oder Positionen zu berechnen, führt eine vorhandene Nullpunktdrift dazu, dass auch ohne tatsächliche Drehbewegung der Integrationsprozess Fehler akkumuliert. Dies bewirkt, dass die Messwerte allmählich vom tatsächlichen Wert abweichen. Solche Fehler nehmen meistens im Laufe der Zeit linear zu und beeinträchtigen die langfristige Stabilität und Genauigkeit des Systems erheblich.

Um den durch Nullpunktdrift verursachten Fehler im Messprozess des Gyroskops zu kompensieren, wird vor jedem Experiment ein Korrekturverfahren angewendet, um die Winkelgeschwindigkeit entlang der z-Achse zu justieren. Hierunten ist das zugehörige Programm:

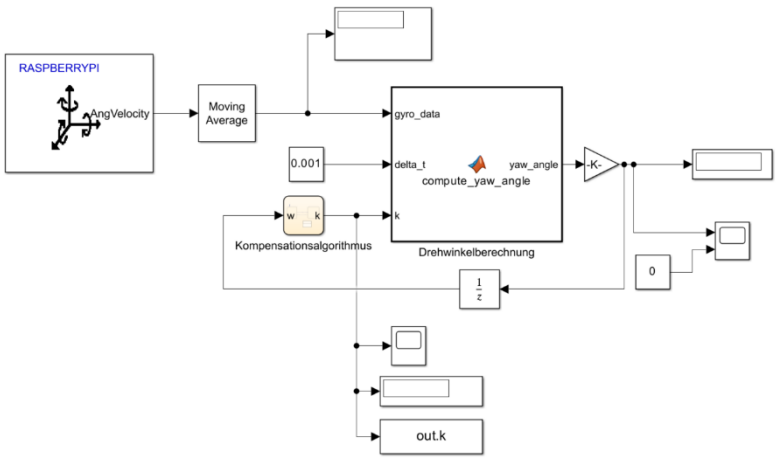


Abbildung 6.1 Berechnungsprogramm für die kompensierte Winkelgeschwindigkeit

In den durchgeführten Experimenten wurde festgestellt, dass das verwendete Gyroskop-Modul eine Winkelgeschwindigkeit von etwa -1,8°/s misst, obwohl keine tatsächliche Drehbewegung stattfindet. Um den durch diesen Fehler verursachten Einfluss so weit wie möglich zu minimieren, wurde der folgende Kompensationsalgorithmus entwickelt:

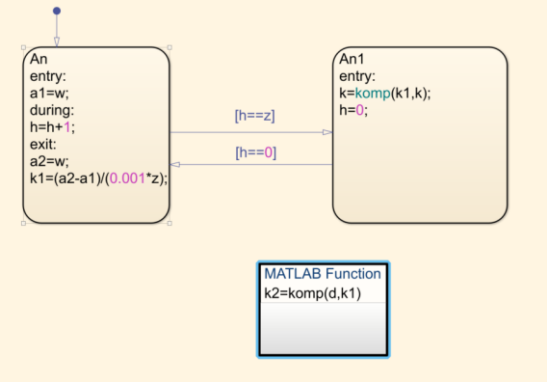


Abbildung 6.2 Zustandsautomat Kompensationsalgorithmus

Das Programm besteht aus zwei Zuständen und einer MATLAB-Funktion. Im ersten Zustand wird der aktuelle Winkel gespeichert, und während der Dauer des Zustands wird ein Zähler inkrementiert (Der Zähler hat eine Obergrenze von z=50.). Beim Verlassen des Zustands wird die Winkeländerungsrate k1 berechnet. Es wird die Steigung der über 50 Messungen aufgezeichneten Winkelwerte berechnet.

$$k1 = \frac{a_2 - a_1}{0.001 * z} \tag{6.1}$$

Wenn der Zähler die Obergrenze z erreicht, erfolgt der Übergang in den zweiten Zustand, in dem die Funktion „komp“ aufgerufen wird, um die kompensierte Winkelgeschwindigkeit k (anfänglich auf 1,8 eingestellt) anzupassen.

```
function k3=komp(k1,k2)
if k1>0
    k2=k2-0.001;
else
    k2=k2+0.001;
end
```

Abbildung 6.3 Anpassung der kompensierten Winkelgeschwindigkeit

Wenn die berechnete Winkeländerungsrate k1 größer als 0 ist, wird die kompensierte Winkelgeschwindigkeit k2 um 0,001 verringert, andernfalls wird k2 um 0,001 erhöht. Schließlich wird k3 als neue kompensierte Winkelgeschwindigkeit in die Berechnung des Drehwinkels einbezogen.

In Abbildung 6.1 wird außerdem kontinuierlich die aktualisierte kompensierte Winkelgeschwindigkeit k erfasst. Sobald die gemessenen Winkelwerte stabil sind und das Programm beendet wird, werden alle Daten im MATLAB-Arbeitsbereich gespeichert. Mithilfe des folgenden Programms wird der Durchschnitt der letzten 20 aufgezeichneten kompensierten

Winkelgeschwindigkeiten k berechnet. Dieser Durchschnittswert dient dann als kompensierte Winkelgeschwindigkeit für das Steuerungsprogramm des Roboters.

```
%% Berechnung der kompensierten Winkelgeschwindigkeit des
Nullpunkts
data = k.Data;
% Bestimmung der Länge des Arrays
n = length(data);

% Wenn die Länge des Arrays größer oder gleich 20 ist, wird
der Durchschnitt der letzten 20 Werte berechnet
if n >= 20
last_20_avg = mean(data(end-19:end));
else
error('Die Länge des Arrays ist kleiner als 20');
end
```

Abbildung 6.4 Berechnung des Durchschnittswerts der kompensierten Winkelgeschwindigkeit

Durch das oben beschriebene Kompensationsprogramm kann der Nullpunktdriftfehler des Gyroskops im Ruhezustand effektiv korrigiert werden, was die Messgenauigkeit und -zuverlässigkeit erhöht. Diese Methode berücksichtigt nicht nur die Veränderungen der Echtzeitdaten, sondern korrigiert den Fehler auch mithilfe eines mathematischen Modells. Dadurch wird eine präzise Kontrolle und Korrektur der Messfehler des Gyroskops erreicht.

7 Fazit und Ausblick

Diese Studie konzentriert sich auf die Entwicklung und Anwendung eines Interaktionssystems zwischen einem Dummy und dem Arlo-Roboter im Bereich der Fahrzeugtests, mit dem Ziel, die Dynamik und Präzision der Experimente zu verbessern. Durch die Einführung der „Stufenschwächung“ und der „Segmentierten PID“-Steuerungsstrategien konnten die Bewegungssteuerungsprobleme des Dummys auf den „SÜ“- und „GE“-Routen effektiv gelöst werden. Die experimentellen Ergebnisse zeigen, dass das entwickelte System über eine gute Trajektorienausrichtungsfähigkeit verfügt, innerhalb kurzer Zeit zur vorgegebenen Trajektorie zurückkehrt und eine stabile Bewegungsrichtung beibehält. Dadurch wird der Bedarf an mehrfacher Positionsanpassung des Dummys, wie er in herkömmlichen Experimenten häufig erforderlich ist, reduziert. Dieses Ergebnis durchbricht in gewisser Weise die Einschränkungen traditioneller Fahrzeugtests, bei denen der Dummy in einer festen Position steht, und ermöglicht dem Dummy eine Bewegung entlang vorgegebener Trajektorien, was innovative Ansätze und Methoden für den Einsatz von Robotern in Fahrzeugexperimenten bietet.

Trotz der bedeutenden Fortschritte, die in der Entwicklung und Steuerung des kombinierten Systems erzielt wurden, besteht weiterhin Optimierungsbedarf in Bezug auf die Steuerungsgenauigkeit. Konkret weist das System in Innenumgebungen bei der Kontrolle der Strecke entlang der vorgegebenen Trajektorie noch gewisse Fehler auf. In Außenumgebungen fehlt dem System derzeit eine Distanzkontrolle, was ebenfalls weiter erforscht und verbessert werden sollte. Darüber hinaus muss bei der Behandlung von Nullpunktdrift im Experiment vor jedem Versuch ein Winkelgeschwindigkeitskompensationsprogramm ausgeführt werden, um eine geeignete Kompensationsgeschwindigkeit zu berechnen. Obwohl diese Maßnahme die Genauigkeit erhöht, verlangsamt sie den Fortschritt der Experimente und beeinträchtigt deren Effizienz. Zukünftige Forschungen sollten sich daher darauf konzentrieren, die Steuerungsgenauigkeit des Systems zu verbessern, insbesondere im Bereich der Trajektoriendistanzkontrolle, und die Experimentabläufe zu optimieren, um die Gesamteffizienz zu steigern. Diese Verbesserungen könnten die Anwendung des Interaktionssystems zwischen Dummy und Roboter in Fahrzeugexperimenten und verwandten Bereichen weiter vorantreiben und so breiteren praktischen Anforderungen gerecht werden.

Ausblickend gibt es viele vielversprechende Bereiche für die Weiterentwicklung des Interaktionssystems zwischen Dummy und Arlo-Roboter. Erstens könnten in der Steuerungsalgorithmik fortschrittlichere intelligente Steuerungsmethoden wie adaptive Steuerung, Fuzzy-Logik oder maschinelles Lernen eingeführt werden, um die Anpassungsfähigkeit des Systems in komplexen Umgebungen zu verbessern. Durch die Implementierung solcher intelligenten Algorithmen könnte das System eine höhere Präzision und Flexibilität bei sich dynamisch ändernden Experimentalszenarien erreichen.

Zweitens könnte man angesichts der aktuellen Nullpunktdrift-Problematik in zukünftigen Experimenten effizientere und automatisierte Kompensationsmechanismen entwickeln, um den Bedarf an manuellen Anpassungen vor den Experimenten zu verringern. Dies würde nicht nur die

Effizienz der Experimente erhöhen, sondern auch die Echtzeitfähigkeit und Stabilität des Systems verbessern und somit die Anforderungen in der Praxis besser erfüllen.

Auf Anwendungsebene könnte das Interaktionssystem zwischen Dummy und Roboter auf breitere Bereiche ausgeweitet werden, wie zum Beispiel im Bereich der intelligenten Mobilität, Sicherheitsprüfungen für autonomes Fahren oder bei der Ausführung komplexer Aufgaben in der Mensch-Maschine-Interaktion. Insbesondere vor dem Hintergrund der zunehmenden Reife autonomer Fahrtechnologien birgt dieses System großes Potenzial, da es dynamische und realistische Testmodelle für Sicherheitsprüfungen in verschiedenen Szenarien bieten könnte.

Schließlich könnte eine interdisziplinäre Zusammenarbeit neue Möglichkeiten für die weitere Optimierung und Innovation dieses Systems eröffnen. Die Integration mit Bereichen wie künstliche Intelligenz, Computer Vision und Sensortechnologie könnte zu intelligenteren und umfassenderen Systemen führen, die ihre Entscheidungsfähigkeit und Betriebsleistung in komplexen Umgebungen verbessern.

Zusammenfassend lässt sich sagen, dass das Interaktionssystem zwischen Dummy und Arlo-Roboter eine vielversprechende Zukunft hat. Durch kontinuierliche technologische Innovation und Anwendungserweiterung könnte dieses System eine bedeutende Rolle in Fahrzeugtests und verwandten Bereichen spielen und so zur Förderung des Fortschritts in diesen Industrien beitragen.

Literatur- und Quellenverzeichnis

[1] Jie Kuang:
服务业机器人的发展. 2020.

[2] o.V.:
Arlo Robot-Robuste Plattform für mobile Roboter. Webseite.2021. URL:
https://elmicro.com/de/arlo-robot.html (besucht am 09.06.2024)

[3] Jack N RealVNC:
How do I get started with RealVNC Connect on Windows? Webseite. 2024. URL:
https://help.realvnc.com/hc/en-us/articles/360003474552-How-do-I-get-started-with-RealVNC-Connect-on-Windows#on-the-device-you-want-to-control-0-0 (besucht am 11.06.2024)

[4] Wenzheng Li:
HC-SR04 超声波测距模块说明. Webseite.2017. URL:
https://blog.csdn.net/weixin_39513374/article/details/78645941 (besucht am 13.06.2024)

[5] Parallax Inc.:
Arlo Power Distribution Board. Webseite. 2024. URL:
https://learn.parallax.com/tutorials/robot/arlo/arlo-robot-assembly-guide/section-4-arlo-power-distribution-board (besucht am 24.06.2024)

[6] o.V.:
Simulink 中计时器的 5 种实现方式. Webseite.2020. URL:
https://zhuanlan.zhihu.com/p/179205039 (besucht am 13.06.2024)

[7] o.V.:
MATLAB Function. Webseite.2024. URL:
https://ww2.mathworks.cn/help/simulink/slref/matlabfunction.html?s_tid=doc_ta (besucht am 13.06.2024)

[8] Franziskus Mendt:
target_mover_V2. Webseite. 2023. URL:
https://gitlab.com/htw_nives/test_technology_center/target_mover_v2 (besucht am 28.08.2024)

[9] o.V.:
Proportional–integral–derivative controller. Webseite.2024. URL:
https://en.wikipedia.org/wiki/Proportional%E2%80%93integral%E2%80%93derivative_controller (besucht am 18.06.2024)

[10] Grant Maloy Smith:
Was ist ein PID-Regler? Webseite. 2024. URL: https://dewesoft.com/de/blog/was-ist-ein-pid-regler (besucht am 18.06.2024)

[11] Zhiqiang Zhu:
基于 MATLAB/Simulink 闭环小车控制系统设计. Webseite. 2016. URL:
https://m.fx361.com/news/2016/1114/17721457.html (besucht am 28.06.2024)

[12] University of Science and Technology of China:
测量的不确定度 Webseite. O.J. URL:
https://jxzy.ustc.edu.cn/video/file/notice/uncertainty.pdf (besucht am 23.06.2024)

[13] Klaus Eckhardt:
Messunsicherheit. Webseite.20. URL:
https://www.aufgabomat.de/physik/Messunsicherheit.pdf (besucht am 23.06.2024)

[14] o.V.:
Chart. Webseite.2024. URL:
https://ww2.mathworks.cn/help/stateflow/ref/chart.html?searchHighlight=chart&s_tid=srchtitle_support_results_1_chart (besucht am 10.05.2024)

[15] Dirk Engert:
„Vorlesungsskript Beschreibung von Programmabläufen“. 2023.

[16] Matlab Help Center:
LSM9DS1 IMU Sensor. o.J. URL:
https://ww2.mathworks.cn/help/simulink/supportpkg/raspberrypi_ref/lsm9ds1imusensor.html?s_tid=doc_ta (besucht am 06.07.2024)

Eidesstattliche Erklärung

Das vorliegende Dokument wurde an dem K-Gebäude der HTWD unter der Leitung von Prof. Dr. rer. nat. Toralf Trautmann, Dr. -Ing. Duo Fu, Dr.-Ing. Dirk Engert und Dr. Ing. Franziskus Mendt angefertigt.

Hiermit erkläre ich, dass ich die vorliegende Arbeit zum Thema

„Entwicklung einer Roboter-Ansteurung“

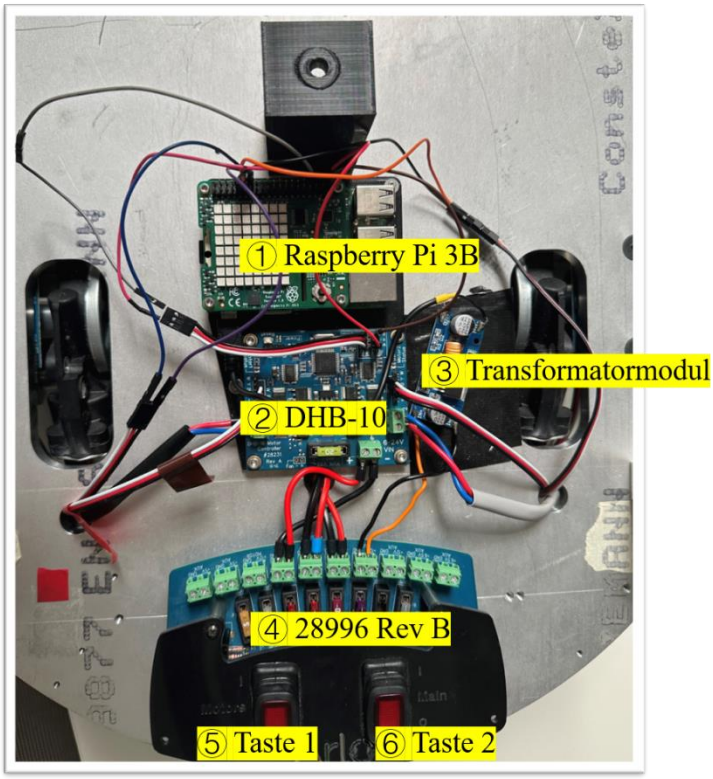
selbstständig und ohne Benutzung anderer Quellen und Hilfsmittel als angegeben angefertigt habe.

Insbesondere versichere ich, dass ich alle wörtlichen und sinngemäßen Übernahmen als solche kenntlich gemacht habe.

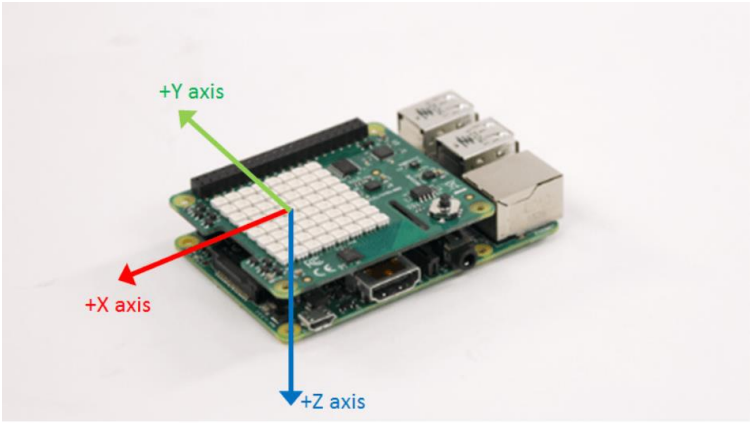
Ferner gestatte ich der Hochschule für Technik und Wirtschaft Dresden, die vorliegende Belegarbeit unter Beachtung insbesondere urheber-, datenschutz- und wettbewerbsrechtlicher Vorschriften für Lehre und Forschung zu nutzen.

Anlagenverzeichnis

Anlage 1 - Hardwarestruktur der Plattformoberfläche

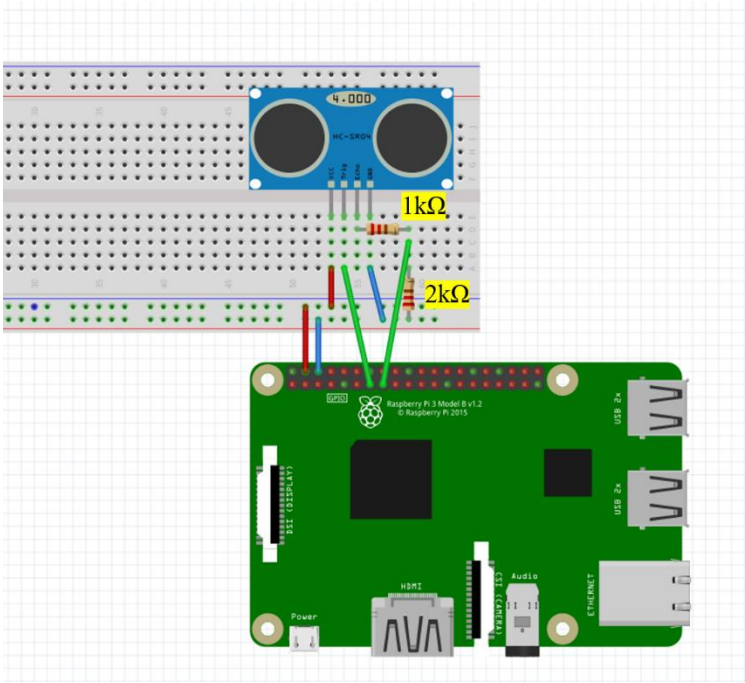


Anlage 2 - die drei Achsen des Sense HAT (Matlb Help Center, o.J.)[16]



Anlage 3 - HC-SR04

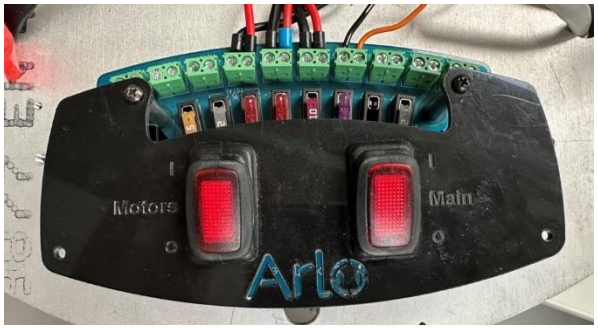
Anlage 4 - Verbindungsmethode zwischen HC-SR04 und Raspberry Pi



Anlage 5 - Motorhalterung und Rad-Kit[2]

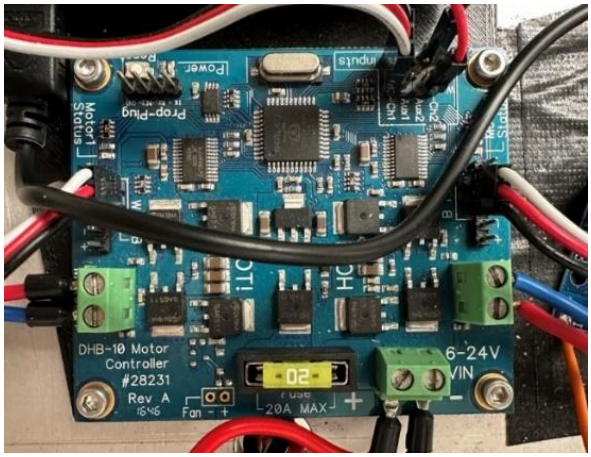
Anlage 6 - Optischer Encoder und dessen Installationsposition[5]

Anlage 7 - Arlo Power Distribution Board Kit Rev B



Anlage 8 - DC-DC-Abwärtswandler

Anlage 9 - DHB-10 Dual H-Bridge



Anlage 10 - Netzwerkwechsel

```
  GNU nano 5.4          /etc/wpa_supplicant/wpa_supplicant.conf
country=CN
ctrl_interface=DIR=/var/run/c GROUP=netdev
update_config=1
network={
        ssid="ROSAP_2G"
        psk="[redacted]"
        priority=10
}
```

Anlage 11 - DHCP Einstellung

DHCP Settings

DHCP Server:	○ Disable ◉ Enable	
Start IP Address:	192.168.1.183	
End IP Address:	192.168.1.184	
Address Lease Time:	120	minutes (1~2880 minutes, the default value is 120)
Default Gateway:	192.168.1.4	(Optional)
Default Domain:		(Optional)
Primary DNS:	8.8.8.8	(Optional)
Secondary DNS:	9.9.9.9	(Optional)

Anlage 12 - PWM-Drehzahlkalibrierungsexperiment

PWM-Drehzahlkalibrierungsexperiment (links)										
TV \ EG	1	2	3	4	5	6	7	8	9	10
0.25	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850
0.265625	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026
0.28125	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927
0.296875	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486
0.3125	0.8294	0.8294	0.8294	0.8294	0.8293	0.8294	0.8294	0.8294	0.8294	0.8294
0.328125	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039
0.34375	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912
0.359375	0.1647	0.1653	0.1630	0.1611	0.1684	0.1684	0.1639	0.1663	0.1715	0.1578
0.375	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.390625	0.1447	0.1537	0.1411	0.1466	0.1478	0.1447	0.1432	0.1470	0.1485	0.1477
0.40625	0.3472	0.3530	0.3458	0.3482	0.3463	0.3481	0.3466	0.3462	0.3481	0.3486
0.421875	0.5728	0.5737	0.5751	0.5728	0.5745	0.5762	0.5762	0.5766	0.5808	0.5788
0.4375	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170
0.453125	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288
0.46875	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352
0.484375	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620
0.5	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886

PWM-Drehzahlkalibrierungsexperiment (links)

PWM-Drehzahlkalibrierungsexperiment (rechts)										
TV \ EG	1	2	3	4	5	6	7	8	9	10
0.25	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850	1.6850
0.265625	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026	1.5026
0.28125	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927	1.2927
0.296875	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486	1.0486
0.3125	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170
0.328125	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039	0.6039
0.34375	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912	0.3912
0.359375	0.1634	0.1675	0.1699	0.1624	0.1615	0.1634	0.1657	0.1643	0.1684	0.1705
0.375	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.390625	0.1447	0.1522	0.1561	0.1543	0.1510	0.1510	0.1526	0.1584	0.1570	0.1515
0.40625	0.3516	0.3516	0.3516	0.3516	0.3516	0.3516	0.3516	0.3516	0.3516	0.3516
0.421875	0.5728	0.5792	0.5759	0.5763	0.5764	0.5754	0.5792	0.5779	0.5791	0.5788
0.4375	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170	0.8170
0.453125	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288	1.0288
0.46875	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352	1.2352
0.484375	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620	1.4620
0.5	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886	1.5886

PWM-Drehzahlkalibrierungsexperiment (rechts)

Anlage 13 - Bewertungsparameter

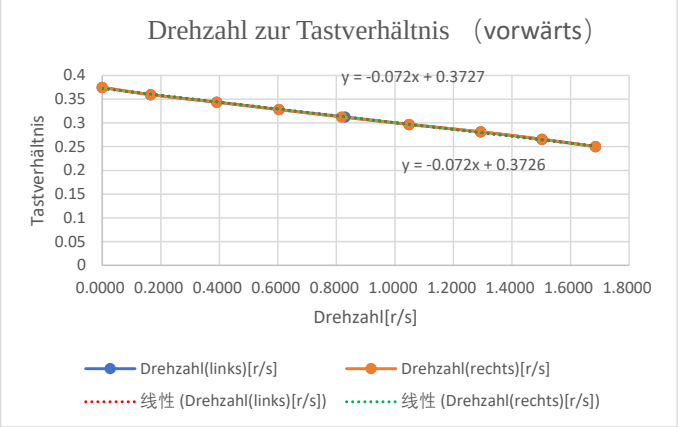
$\bar{x}$	σx	uA	UA0.68	UA0.95	UA0.99
1.6850	0.0000	0.0000	0.0000	0.0000	0.0000
1.5026	0.0000	0.0000	0.0000	0.0000	0.0000
1.2927	0.0000	0.0000	0.0000	0.0000	0.0000
1.0486	0.0000	0.0000	0.0000	0.0000	0.0000
0.8294	0.0000	0.0000	0.0000	0.0000	0.0000
0.6039	0.0000	0.0000	0.0000	0.0000	0.0000
0.3912	0.0000	0.0000	0.0000	0.0000	0.0000
0.1650	0.0012	0.0004	0.0004	0.0009	0.0013
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.1465	0.0011	0.0003	0.0004	0.0008	0.0011
0.3478	0.0007	0.0002	0.0002	0.0005	0.0007
0.5758	0.0008	0.0003	0.0003	0.0006	0.0008
0.8170	0.0000	0.0000	0.0000	0.0000	0.0000
1.0288	0.0000	0.0000	0.0000	0.0000	0.0000
1.2352	0.0000	0.0000	0.0000	0.0000	0.0000
1.4620	0.0000	0.0000	0.0000	0.0000	0.0000
1.5886	0.0000	0.0000	0.0000	0.0000	0.0000

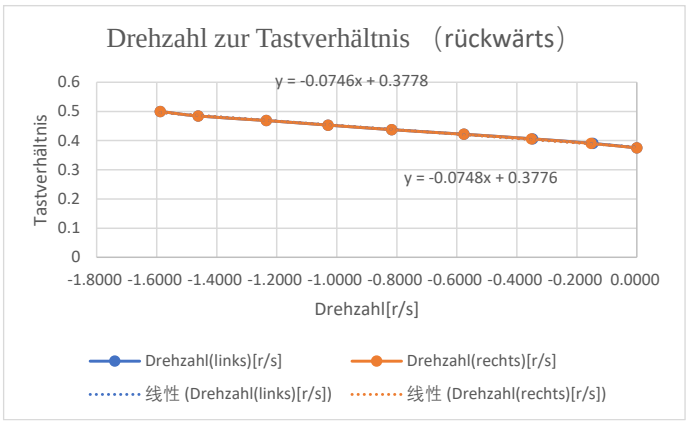
Bewertungsparameter für das linke Rad

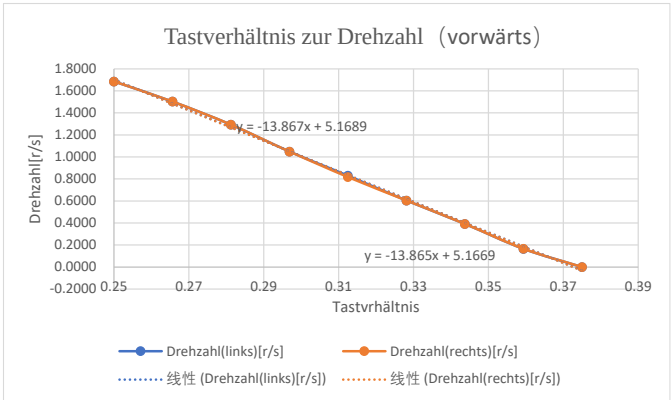
$\bar{x}$	σx	uA	UA0.68	UA0.95	UA0.99
1.6850	0.0000	0.0000	0.0000	0.0000	0.0000
1.5026	0.0000	0.0000	0.0000	0.0000	0.0000
1.2927	0.0000	0.0000	0.0000	0.0000	0.0000
1.0486	0.0000	0.0000	0.0000	0.0000	0.0000
0.8170	0.0000	0.0000	0.0000	0.0000	0.0000
0.6039	0.0000	0.0000	0.0000	0.0000	0.0000
0.3912	0.0000	0.0000	0.0000	0.0000	0.0000
0.1657	0.0010	0.0003	0.0003	0.0007	0.0010
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.1529	0.0012	0.0004	0.0004	0.0009	0.0013
0.3516	0.0000	0.0000	0.0000	0.0000	0.0000
0.5771	0.0007	0.0002	0.0002	0.0005	0.0007
0.8170	0.0000	0.0000	0.0000	0.0000	0.0000
1.0288	0.0000	0.0000	0.0000	0.0000	0.0000
1.2352	0.0000	0.0000	0.0000	0.0000	0.0000
1.4620	0.0000	0.0000	0.0000	0.0000	0.0000
1.5886	0.0000	0.0000	0.0000	0.0000	0.0000

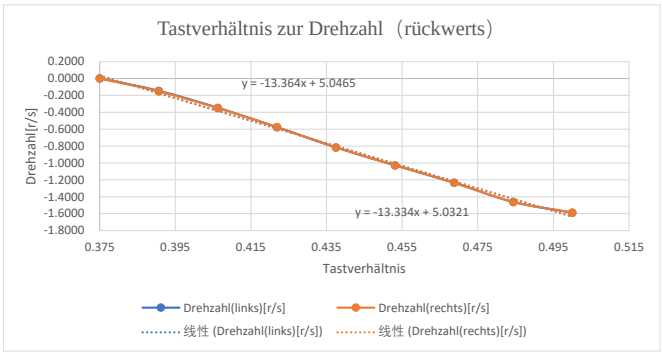
Bewertungsparameter für das rechte Rad

Anlage 14 - Diagrammformat für die Beziehung zwischen TV und Drehzahl









Anlage 15 - Experimentelle Daten des statischen Kalibrierungsexperiments

in Uhrzeigersinn										
Steuerungszeit/s	1st/°	2/°	3/°	4/°	5/°	6/°	7/°	8/°	9/°	10/°
0.1	6	5	6	6	5	5	6	6	6	6
0.2	9	9	8	9	9	8	9	8	8	8
0.3	10	11	11	10	10	10	10	10	10	10
0.4	14	13	13	14	13	13	12	12	12	12
0.5	15	14	14	15	14	15	15	16	14	15
0.6	17	17	18	17	16	17	16	17	17	18
0.7	19	21	18	19	21	21	20	22	19	19
0.8	22	22	21	21	21	21	23	22	22	22
0.9	25	24	25	25	24	24	24	25	24	23
1	28	27	26	26	27	26	27	26	26	25
1.5	39	37	37	38	38	38	36	38	38	37
2	49	47	50	49	48	50	49	47	49	46

Messdaten: Drehwinkel und Steuerungszeit im Uhrzeigersinn

gegen Uhrzeigersinn										
Steuerungszeit/s	1st/°	2/°	3/°	4/°	5/°	6/°	7/°	8/°	9/°	10/°
0.1	6	5	5	6	3	5	6	5	6	6
0.2	8	8	8	9	8	8	8	8	8	8
0.3	10	9	10	10	10	10	10	10	10	10
0.4	13	13	13	13	13	12	12	13	13	12
0.5	14	15	15	15	14	15	16	16	16	15
0.6	18	18	18	17	17	19	18	19	19	18
0.7	20	20	20	21	21	21	20	20	21	21
0.8	22	22	23	22	21	22	23	21	23	23
0.9	25	24	24	25	25	26	25	24	24	24
1	27	28	28	27	26	28	27	27	27	27
1.5	38	39	40	39	39	38	38	41	40	37
2	51	53	50	51	51	52	51	51	51	53

Messdaten: Drehwinkel und Steuerungszeit gegen Uhrzeigersinn

Anlage 16 - Datenanalyse des statischen Kalibrierungsexperiments

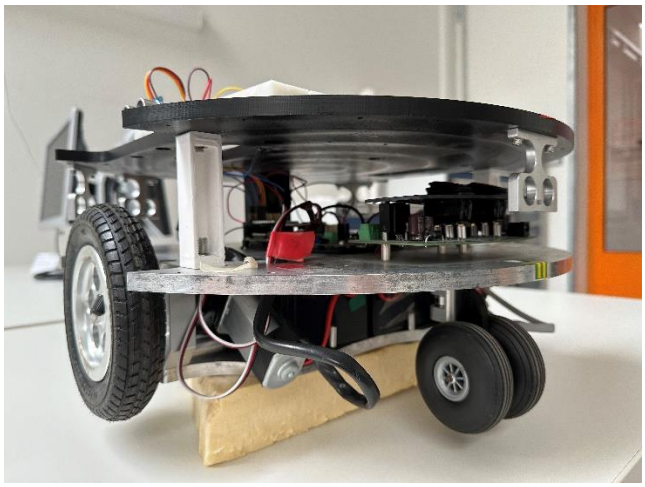
$\bar{x}$	σ	uA	UA0.68	UA0.95	UA0.99
5.70	0.1528	0.0483	0.0512	0.1092	0.1570
8.50	0.1667	0.0527	0.0559	0.1191	0.1713
10.20	0.1333	0.0422	0.0447	0.0953	0.1370
12.80	0.2494	0.0789	0.0836	0.1783	0.2564
14.70	0.2134	0.0675	0.0715	0.1525	0.2194
17.00	0.2108	0.0667	0.0707	0.1507	0.2167
19.90	0.4069	0.1287	0.1364	0.2908	0.4182
21.70	0.2134	0.0675	0.0715	0.1525	0.2194
24.30	0.2134	0.0675	0.0715	0.1525	0.2194
26.40	0.2667	0.0843	0.0894	0.1906	0.2741
37.60	0.2667	0.0843	0.0894	0.1906	0.2741
48.40	0.4269	0.1350	0.1431	0.3051	0.4387

Datenanalyse: Drehwinkel und Steuerungszeit im Uhrzeigersinn

x̄	σ	uA	UA0.68	UA0.95	UA0.99
5.30	0.3000	0.0949	0.1006	0.2144	0.3083
8.10	0.1000	0.0316	0.0335	0.0715	0.1028
9.90	0.1000	0.0316	0.0335	0.0715	0.1028
12.70	0.1528	0.0483	0.0512	0.1092	0.1570
15.10	0.2333	0.0738	0.0782	0.1668	0.2398
18.10	0.2333	0.0738	0.0782	0.1668	0.2398
20.50	0.1667	0.0527	0.0559	0.1191	0.1713
22.20	0.2494	0.0789	0.0836	0.1783	0.2564
24.60	0.2211	0.0699	0.0741	0.1580	0.2272
27.20	0.2000	0.0632	0.0670	0.1429	0.2055
38.90	0.3786	0.1197	0.1269	0.2706	0.3891
51.40	0.3055	0.0966	0.1024	0.2183	0.3140

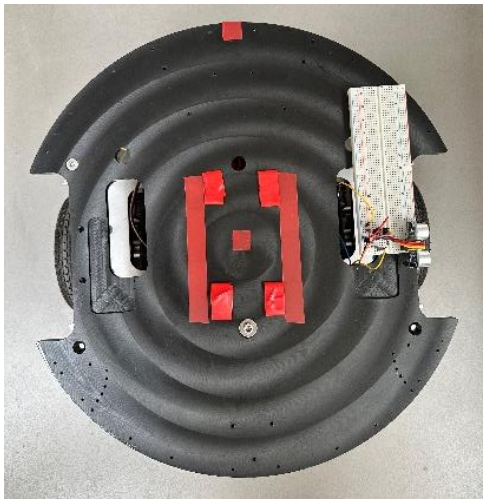
Datenanalyse: Drehwinkel und Steuerungszeit gegen Uhrzeigersinn

Anlage 17 - Radaufhängung



Anlage 18 - Kalibrierungsexperiment des Drehwinkels

Anlage 19 - Roboterplattform mit Klebemarkierung



Anlage 20 - Experimentelle Daten des dynamischen Kalibrierungsexperiments

In Uhrzeigersinn										
Zeit[s]	1st[°]	2[°]	3[°]	4[°]	5[°]	6[°]	7[°]	8[°]	9[°]	10[°]
0.3	2	1	3	3	1	3	1	2	2	2
0.4	2	4	5	3	3	4	3	3	4	5
0.5	6	6	7	7	6	5	4	7	6	5
0.6	7	7	9	8	9	7	7	7	7	7
0.7	10	10	10	12	11	10	9	9	11	10
0.8	12	11	11	11	12	13	11	12	12	12
0.9	14	14	15	14	15	14	15	14	13	13
1	18	17	16	18	15	16	17	16	16	18
1.5	30	28	27	29	27	27	29	29	30	30
2	41	39	39	39	41	40	38	40	40	41
3	62	62	65	61	62	63	63	61	61	63

Messdaten: Drehwinkel und Steuerungszeit im Uhrzeigersinn

gegen Uhrzeigersinn										
Zeit[s]	1st[°]	2[°]	3[°]	4[°]	5[°]	6[°]	7[°]	8[°]	9[°]	10[°]
0.3	2	4	2	2	2	4	2	2	3	2
0.4	3	4	4	3	3	3	5	4	4	4
0.5	6	5	7	6	5	6	7	5	7	5
0.6	7	7	8	9	7	8	8	8	7	8
0.7	10	9	11	10	11	10	9	10	11	10
0.8	12	12	12	12	12	12	11	13	13	13
0.9	13	15	13	15	14	13	14	15	13	14
1	18	17	16	18	15	16	17	16	16	18
1.5	28	30	28	29	29	30	30	29	30	27
2	39	39	39	41	42	37	41	37	39	43
3	62	63	61	65	64	63	64	63	64	62

Messdaten: Drehwinkel und Steuerungszeit gegen Uhrzeigersinn

Anlage 21 - Datenanalyse des dynamischen Kalibrierungsexperiments

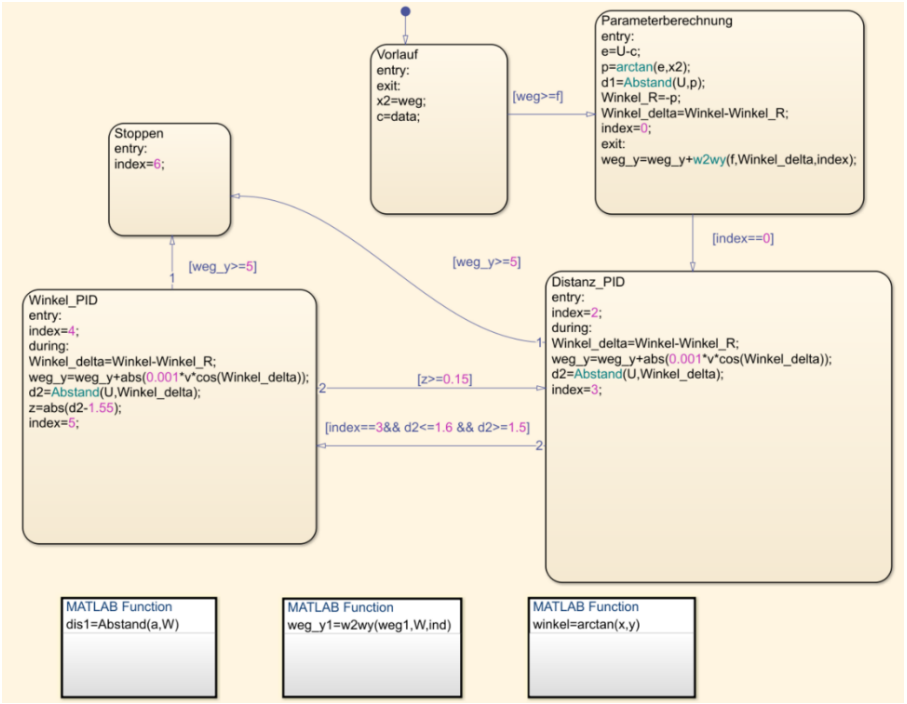
Zeit[s]	x̄	σ	uA	UA0.68	UA0.95	UA0.99
0.3	2.00	0.2582	0.0816	0.0865	0.1845	0.2654
0.4	3.60	0.3055	0.0966	0.1024	0.2183	0.3140
0.5	5.90	0.3145	0.0994	0.1054	0.2247	0.3232
0.6	7.50	0.2772	0.0876	0.0929	0.1981	0.2848
0.7	10.20	0.2906	0.0919	0.0974	0.2077	0.2987
0.8	11.70	0.2134	0.0675	0.0715	0.1525	0.2194
0.9	14.10	0.2333	0.0738	0.0782	0.1668	0.2398
1	16.70	0.3350	0.1059	0.1123	0.2394	0.3443
1.5	28.60	0.4000	0.1265	0.1341	0.2859	0.4111
2	39.80	0.3266	0.1033	0.1095	0.2334	0.3357
3	62.30	0.3958	0.1252	0.1327	0.2829	0.4068

Datenanalyse: Drehwinkel und Steuerungszeit in Uhrzeigersinn

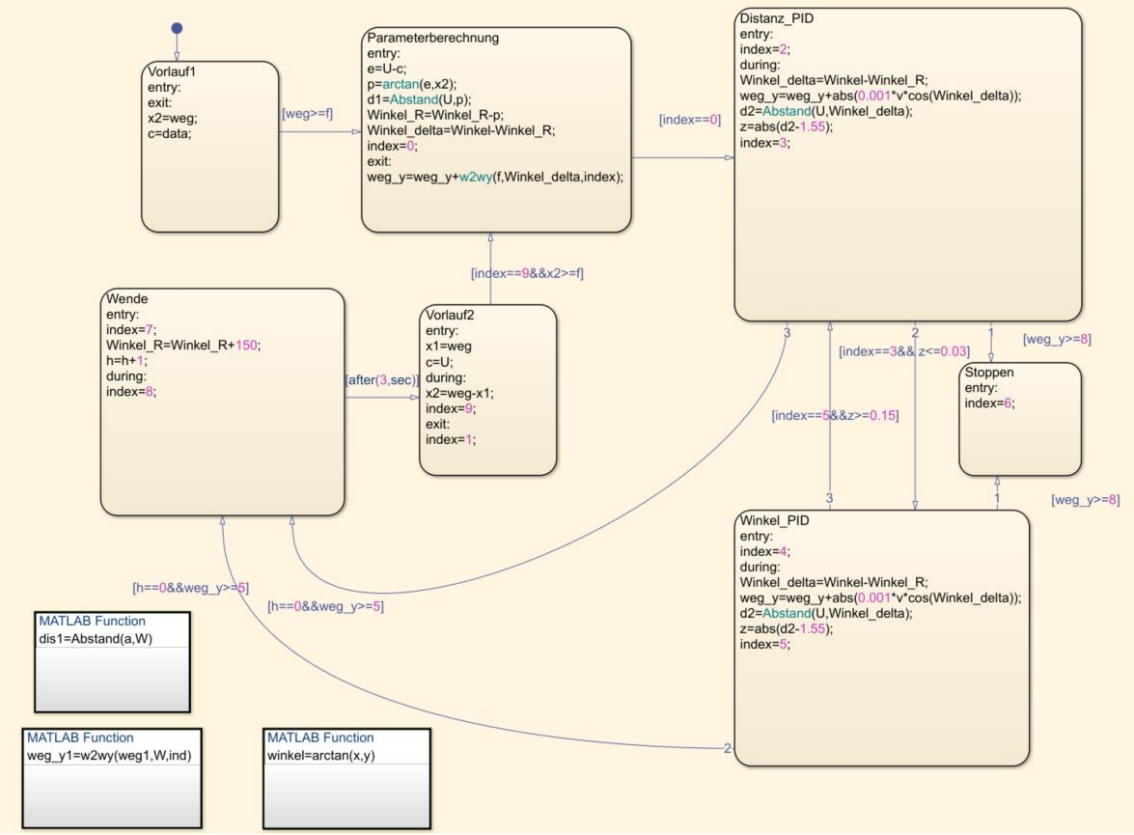
Zeit[s]	x̄	σ	uA	UA0.68	UA0.95	UA0.99
0.3	2.50	0.2687	0.0850	0.0901	0.1921	0.2762
0.4	3.70	0.2134	0.0675	0.0715	0.1525	0.2194
0.5	5.90	0.2769	0.0876	0.0928	0.1979	0.2846
0.6	7.70	0.2134	0.0675	0.0715	0.1525	0.2194
0.7	10.10	0.2333	0.0738	0.0782	0.1668	0.2398
0.8	12.20	0.2000	0.0632	0.0670	0.1429	0.2055
0.9	13.90	0.2769	0.0876	0.0928	0.1979	0.2846
1	16.70	0.3350	0.1059	0.1123	0.2394	0.3443
1.5	29.00	0.3333	0.1054	0.1117	0.2382	0.3426
2	39.70	0.6333	0.2003	0.2123	0.4526	0.6509
3	63.10	0.3786	0.1197	0.1269	0.2706	0.3891

Datenanalyse: Drehwinkel und Steuerungszeit gegen Uhrzeigersinn

Anlage 22 - Zustandsautomat „Verhaltenssteuerung “ bei der segmentierten PID-Methode



Anlage 23 - Zustandsautomat „Verhaltenssteuerung “ bei der segmentierten PID-Methode (GE)



Anlage 24 - Gewichtete Bewertungskriterien

SÜ	RKZ[s]	AA	VAW_{max}[m]	VAW_{min} [m]	SET[m]
$\bar{x}$	4-6：9； 6-8：6； >8：3	0-1：9； 1-2：6； >2：3；	1.55-1.58：9； 1.58-1.61：6； >1.61：3	1.52-1.55：9； 1.49-1.52：6； <1.49：3	5-6：9； 6-6.5：6； >6.5: 3
σ^2	0-0.05：9； 0.05-0.1：6； >0.1：3	0-0.5：9； 0.5-1：6； >1：3；	0-0.002：9； 0.002-0.004：6； >0.004：3	0-0.002：9； 0.002-0.004：6； >0.004：3	0-0.1：9； 0.1-0.2：6； >0.2: 3

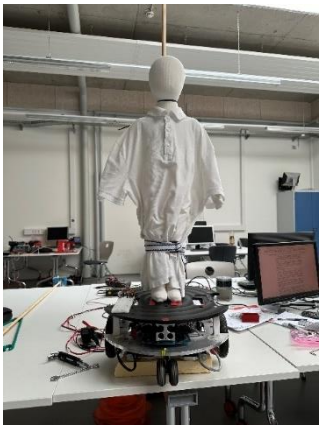
Gewichtete Bewertungskriterien („SÜ“-Route)

GE1	RKZ1[s]	AA1	VAW_{max}1 [m]	VAW_{min}1 [m]	SET1 [m]
$\bar{x}$	4-6：9； 6-8：6； >8：3	0-1：9； 1-2：6； >2：3；	1.55-1.58：9； 1.58-1.61：6； >1.61：3	1.52-1.55：9； 1.49-1.52：6； <1.49：3	5-6：9； 6-6.5：6； >6.5: 3
σ^2	0-0.05：9； 0.05-0.1：6； >0.1：3	0-0.5：9； 0.5-1：6； >1：3；	0-0.002：9； 0.002-0.004：6； >0.004：3	0-0.002：9； 0.002-0.004： 6； >0.004：3	0-0.1：9； 0.1-0.2：6； >0.2: 3
GE2	RKZ2[s]	AA2	VAW_{max}2 [m]	VAW_{min}2 [m]	SET2 [m]
$\bar{x}$	0-2：9； 2-4：6； >4：3	0-1：9； 1-2：6； >2：3；	1.55-1.58：9； 1.58-1.61：6； >1.61：3	1.52-1.55：9； 1.49-1.52：6； <1.49：3	3-4：9； 4-4.5：6； >4.5: 3
σ^2	0-0.05：9； 0.05-0.1：6； >0.1：3	0-0.5：9； 0.5-1：6； >1：3；	0-0.002：9； 0.002-0.004：6； >0.004：3	0-0.002：9； 0.002-0.004： 6； >0.004：3	0-0.1：9； 0.1-0.2：6； >0.2: 3

Gewichtete Bewertungskriterien („GE“-Route)

Anlage 25 - Verbindungsergebnis

Verbindung und Installation der Säule mit dem Holzstab



Gesamteffekt

Anlage 26 - Programm zur Erstellung von Streudiagrammen für SSM bei der SÜ

```
%% Stufenschwächungsmethode(SÜ)
% Daten einlesen
data1 = [d2.Time,d2.Data];
data2 = [index.Time,index.Data];
% Extraktion von Zeit und Entfernung
time1 = data1(:, 1);
distance = data1(:, 2);
time2 = data2(:,1);
ind = data2(:,2);
% Finde den ersten Index 22
startIndex = find(ind == 22, 1);
% Extrahiere die entsprechenden Entfernungswerte
filtered_distance = distance(startIndex:end);
filtered_time = time1(startIndex:end);
% Finde den Maximalwert und Minimalwert sowie die dazugehörigen Zeiten
[max_distance, max_idx] = max(filtered_distance);
[min_distance, min_idx] = min(filtered_distance);
max_time = filtered_time(max_idx);
min_time = filtered_time(min_idx);
% Zeichnen der ursprünglichen Daten
figure;
plot(time1, distance, 'b-', 'DisplayName', 'Distance over Time');
hold on;
% Zeichnen der gefilterten Daten
plot(filtered_time, filtered_distance, 'r-', 'DisplayName', 'filtered
Distance');
% Markiere Maximalwert und Minimalwert
plot(max_time, max_distance, 'ro', 'MarkerSize', 10, 'DisplayName', 'Max
Distance');
plot(min_time, min_distance, 'bo', 'MarkerSize', 10, 'DisplayName', 'Min
Distance');
% Anmerkungen hinzufügen
text(max_time, max_distance, sprintf(' Max:%.2f', max_distance),
'VerticalAlignment', 'bottom', 'HorizontalAlignment', 'right');
text(min_time, min_distance, sprintf(' Min:%.2f', min_distance),
'VerticalAlignment', 'top', 'HorizontalAlignment', 'right');
% Titel und Beschriftungen hinzufügen
title('Distance measure');
xlabel('Time (s)');
ylabel('Distance(m)');
legend;
% Gitter anzeigen
grid on;
hold off;
```

Anlage 27 - Programm zur Erstellung von Streudiagrammen für SSM beim GE

```
%% Stufenschächungsmethode (GE)
data1 = [d2.Time,d2.Data];
data2 = [index.Time,index.Data];
data3 = [h.Time,h.Data];
% Extraktion von Zeit und Entfernung
time1 = data1(:, 1);
distance = data1(:, 2);
time2 = data2(:,1);
ind = data2(:,2);
time3 = data3(:,1);
h1 = data3(:,2);
% Bestimmung des Auftretens des Indexwerts 22
startIndex1 = find(ind == 22, 1);
endIndex1 = find(h1 == 1, 1);
% Extraktion des ersten Abschnitts der Entfernungswerte
filtered_distance1 = distance(startIndex1:endIndex1);
filtered_time1 = time1(startIndex1:endIndex1);
filtered_ind = ind(endIndex1:end);
% Bestimmung des Auftretens des Indexwerts 22 nach der Richtungsänderung
startIndex2 = find(filtered_ind == 22, 1) + endIndex1;
filtered_distance2 = distance(startIndex2:end);
filtered_time2 = time1(startIndex2:end);
% Bestimmung der maximalen und minimalen Werte sowie der entsprechenden
Zeitpunkte
[max_distance1, max_idx1] = max(filtered_distance1);
[min_distance1, min_idx1] = min(filtered_distance1);
[max_distance2, max_idx2] = max(filtered_distance2);
[min_distance2, min_idx2] = min(filtered_distance2);
max_time1 = filtered_time1(max_idx1);
min_time1 = filtered_time1(min_idx1);
max_time2 = filtered_time2(max_idx2);
min_time2 = filtered_time2(min_idx2);
% Darstellung der Originaldaten
figure;
plot(time1, distance, 'b-', 'DisplayName', 'Entfernung über die Zeit');
hold on;
% Darstellung der gefilterten Daten
plot(filtered_time1, filtered_distance1, 'r-', 'DisplayName', 'Gefilterte
Entfernung 1');
plot(filtered_time2, filtered_distance2, 'm-', 'DisplayName', 'Gefilterte
Entfernung 2');
% Markierung der maximalen und minimalen Werte
plot(max_time1, max_distance1, 'ro', 'MarkerSize', 10, 'DisplayName',
'Maximale Entfernung 1');
plot(min_time1, min_distance1, 'bo', 'MarkerSize', 10, 'DisplayName',
'Minimale Entfernung 1');
plot(max_time2, max_distance2, 'ro', 'MarkerSize', 10, 'DisplayName',
'Maximale Entfernung 2');
plot(min_time2, min_distance2, 'bo', 'MarkerSize', 10, 'DisplayName',
'Minimale Entfernung 2');
% Hinzufügen von Anmerkungen
text(max_time1, max_distance1, sprintf(' Max1:%.2f', max_distance1),
'VerticalAlignment', 'bottom', 'HorizontalAlignment', 'right');
text(min_time1, min_distance1, sprintf(' Min1:%.2f', min_distance1),
'VerticalAlignment', 'top', 'HorizontalAlignment', 'right');
text(max_time2, max_distance2, sprintf(' Max2:%.2f', max_distance2),
'VerticalAlignment', 'bottom', 'HorizontalAlignment', 'right');
text(min_time2, min_distance2, sprintf(' Min2:%.2f', min_distance2),
'VerticalAlignment', 'top', 'HorizontalAlignment', 'right');
% Hinzufügen von Titel und Beschriftungen
title('Entfernung während des Gästeempfangs');
xlabel('Zeit (s)');
ylabel('Entfernung (m)');
legend;
grid on;
hold off;
```

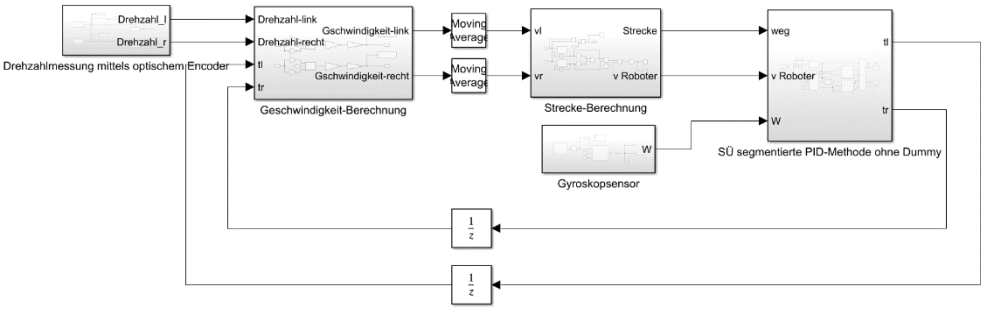
Anlage 28 - Programm zur Erstellung von Streudiagrammen für die segmentierte PID-Methode bei der SÜ

```
%% Die segmentierte PID-Methode (SÜ)
% Angenommen, das gegebene Array ist eine Nx2-Matrix, wobei die erste Spalte die
Zeit und die zweite Spalte die Entfernung darstellt.
data1 = [d2.Time,d2.Data];
data2 = [index.Time,index.Data];
% Extraktion von Zeit und Entfernung
time1 = data1(:, 1);
distance = data1(:, 2);
time2 = data2(:,1);
ind = data2(:,2);
% Bestimmung des ersten Auftretens des Indexwerts 4
startIndex = find(ind == 4, 1);
% Extraktion der entsprechenden Entfernungswerte
filtered_distance = distance(startIndex:end);
filtered_time = time1(startIndex:end);
% Bestimmung der maximalen und minimalen Werte sowie der entsprechenden
Zeitpunkte
[max_distance, max_idx] = max(filtered_distance);
[min_distance, min_idx] = min(filtered_distance);
max_time = filtered_time(max_idx);
min_time = filtered_time(min_idx);
% Darstellung der Originaldaten
figure;
plot(time1, distance, 'b-', 'LineWidth', 2, 'DisplayName', 'Entfernung über die
Zeit');
hold on;
% Darstellung der gefilterten Daten
plot(filtered_time, filtered_distance, 'r-', 'LineWidth', 2, 'DisplayName',
'Gefilterte Entfernung');
% Markierung der maximalen und minimalen Werte
plot(max_time, max_distance, 'ro', 'MarkerSize', 10, 'DisplayName', 'Maximale
Entfernung');
plot(min_time, min_distance, 'bo', 'MarkerSize', 10, 'DisplayName', 'Minimale
Entfernung');
% Hinzufügen von Anmerkungen
text(max_time, max_distance, sprintf(' Max:%.2f', max_distance),
'VerticalAlignment', 'bottom', 'HorizontalAlignment', 'right');
text(min_time, min_distance, sprintf(' Min:%.2f', min_distance),
'VerticalAlignment', 'top', 'HorizontalAlignment', 'right');
% Hinzufügen von Titel und Beschriftungen
title('Entfernungsmessung');
xlabel('Zeit (s)');
ylabel('Entfernung');
legend;
% Aktivierung des Gitternetzes
grid on;
hold off;
```

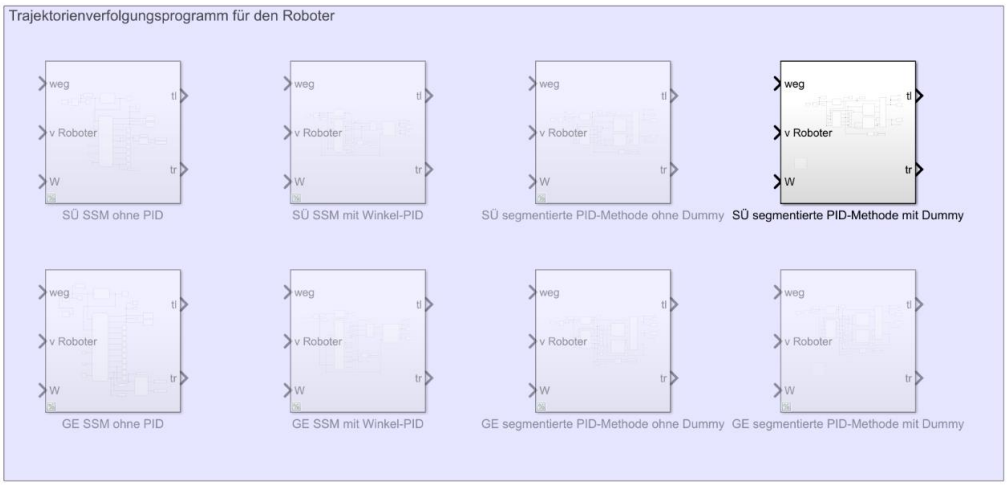
Anlage 29 - Programm zur Erstellung von Streudiagrammen für die segmentierte PID-Methode beim GE

```
%% Die segmentierte PID-Methode (GE)
data1 = [d2.Time,d2.Data];
data2 = [index.Time,index.Data];
data3 = [h.Time,h.Data];
time1 = data1(:, 1);
distance = data1(:, 2);
time2 = data2(:,1);
ind = data2(:,2);
time3 = data3(:,1);
h1 = data3(:,2);
% Bestimmung des ersten Auftretens des Indexwerts 4
startIndex1 = find(ind == 4, 1);
endIndex1 = find(h1 == 1, 1);
% Extraktion des ersten Abschnitts der Entfernungswerte
filtered_distance1 = distance(startIndex1:endIndex1);
filtered_time1 = time1(startIndex1:endIndex1);
filtered_ind = ind(endIndex1:end);
% Bestimmung des ersten Auftretens des Indexwerts 4 nach der Wende
startIndex2 = find(filtered_ind == 4, 1) + endIndex1;
% Extraktion des zweiten Abschnittse
filtered_distance2 = distance(startIndex2:end);
filtered_time2 = time1(startIndex2:end);
% Bestimmung der maximalen und minimalen Werte sowie der entsprechenden
Zeitpunkte
[max_distance1, max_idx1] = max(filtered_distance1);
[min_distance1, min_idx1] = min(filtered_distance1);
[max_distance2, max_idx2] = max(filtered_distance2);
[min_distance2, min_idx2] = min(filtered_distance2);
max_time1 = filtered_time1(max_idx1);
min_time1 = filtered_time1(min_idx1);
max_time2 = filtered_time2(max_idx2);
min_time2 = filtered_time2(min_idx2);
% Darstellung der Originaldaten
figure;
plot(time1, distance, 'b-', 'LineWidth', 2, 'DisplayName', 'Entfernung über die
Zeit');
hold on;
% Darstellung der gefilterten Daten des
plot(filtered_time1, filtered_distance1, 'r-', 'LineWidth', 2, 'DisplayName',
'Gefilterte Entfernung 1');
plot(filtered_time2, filtered_distance2, 'm-', 'LineWidth', 2, 'DisplayName',
'Gefilterte Entfernung 2');
% Markierung der maximalen und minimalen Werte
plot(max_time1, max_distance1, 'ro', 'MarkerSize', 10, 'DisplayName', 'Maximale
Entfernung 1');
plot(min_time1, min_distance1, 'bo', 'MarkerSize', 10, 'DisplayName', 'Minimale
Entfernung 2');
plot(max_time2, max_distance2, 'ro', 'MarkerSize', 10, 'DisplayName', 'Maximale
Entfernung 2');
plot(min_time2, min_distance2, 'bo', 'MarkerSize', 10, 'DisplayName', 'Minimale
Entfernung 2');
% Hinzufügen von Anmerkungen
text(max_time1, max_distance1, sprintf(' Max1:%.2f', max_distance1),
'VerticalAlignment', 'bottom', 'HorizontalAlignment', 'right');
text(min_time1, min_distance1, sprintf(' Min1:%.2f', min_distance1),
'VerticalAlignment', 'top', 'HorizontalAlignment', 'right');
text(max_time2, max_distance2, sprintf(' Max2:%.2f', max_distance2),
'VerticalAlignment', 'bottom', 'HorizontalAlignment', 'right');
text(min_time2, min_distance2, sprintf(' Min2:%.2f', min_distance2),
'VerticalAlignment', 'top', 'HorizontalAlignment', 'right');
% Hinzufügen von Titel und Beschriftungen
title('Entfernungsmessung für den Gästeempfang');
xlabel('Zeit (s)');
ylabel('Entfernung');
legend;
% Aktivierung des Gitternetzes
grid on;
hold off;
```

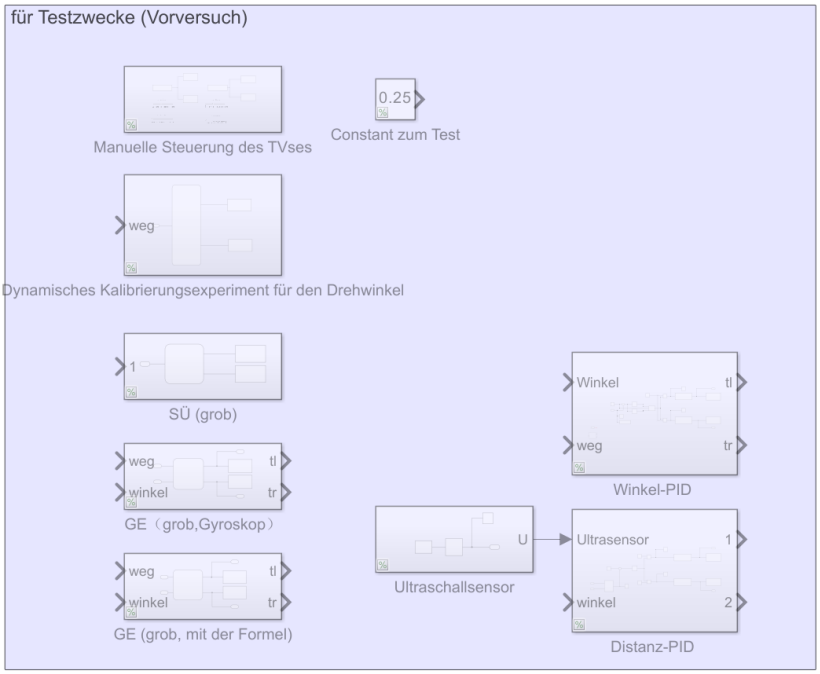
Anlage 30 - Gesamtstruktur des Simulink-Programms



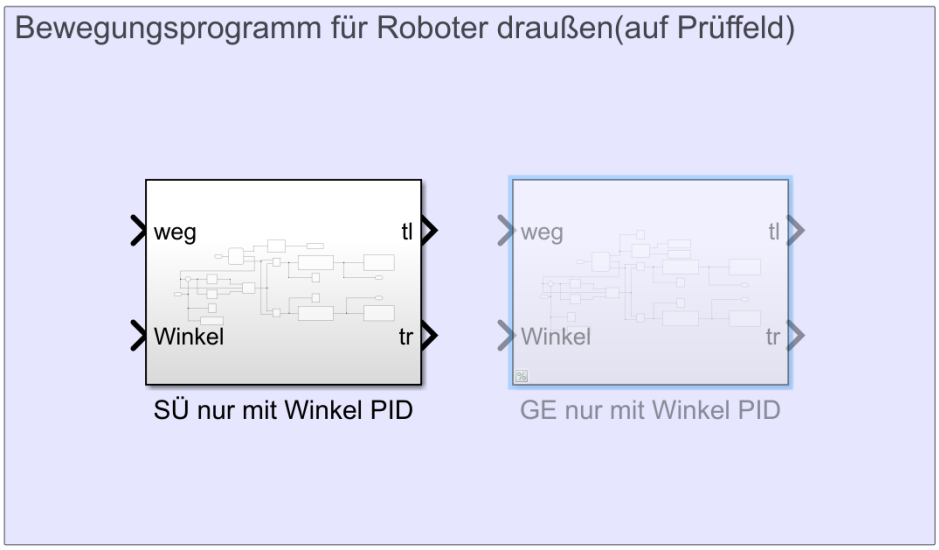
Anlage 31 – Trajektorienverfolgungsprogramme für den Roboter



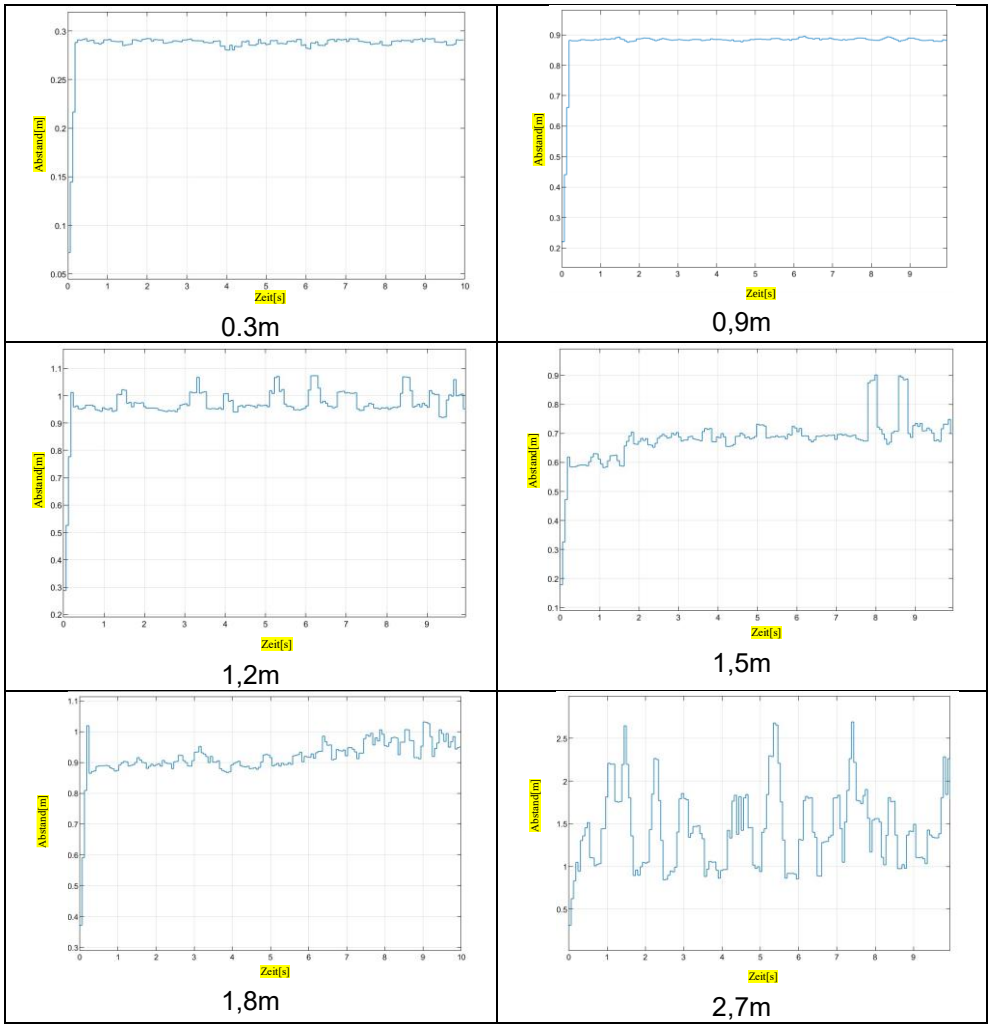
Anlage 32 - Simulinkmodule für Testzwecke



Anlage 33 - Trajektorienverfolgungsprogramme für den Roboter (auf dem Prüffeld)

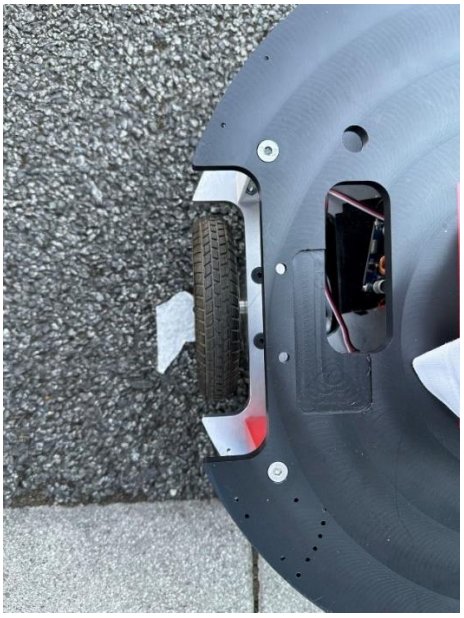


Anlage 34 - Die Messergebnisse der Entfernungsmessung (auf dem Prüffeld)

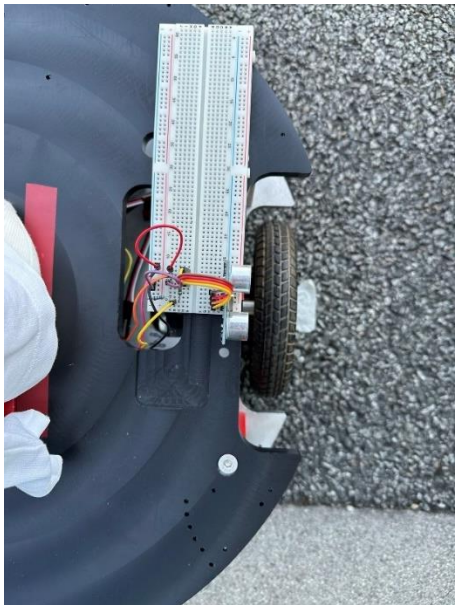


Anlage 35 - Versuch auf dem Prüffeld

Anlage 36 - Bestimmung der Anfangsrichtung und -position des Roboters



Linkes Rad



Rechtes Rad