Entwicklung eines Modellfahrzeugs mit Fahrerassistenzfunktionen

Studienarbeit T3_3100

für die Prüfung zum

Bachelor of Science

des Studiengangs Informationstechnik

an der Dualen Hochschule Baden-Württemberg Karlsruhe

von

Leander Gantert & Luca Müller

31. August 2026

Bearbeitungszeitraum	XX.XX.2025 - XX.XX.2026
Matrikelnummer, Kurs	3854248, TINF23B3
Betreuer der DHBW	Dipl.-Ing. (FH) Stefan Lehmann

Zusammenfassung

Das Thema dieser Bachelorarbeit ist die Entwicklung eines Prototyps für eine neue Softwarearchitektur. Diese Architektur basiert auf dem LIN-Bussystem und soll die Kommunikation zwischen Slaves und Mastern überarbeiten. Ziel dieser Arbeit ist es, eine...

Diese Bachelorarbeit behandelt die Entwicklung eines elektronischen Systems zur Temperaturbegrenzung von Strahlungsheizkörpern mithilfe von Thermoelementen. Ziel dieser Arbeit ist es, ein kostengünstigeres und flexibleres System zu entwickeln, das die bislang verwendeten mechanischen Temperaturbegrenzer ersetzt. Hintergrund ist die zunehmende Marktanforderung nach sensorbasierten Lösungen sowie die Notwendigkeit, den Oil-Ignition-Test der Norm UL 858 zu bestehen, der zukünftig auch für Strahlungsheizkörper verpflichtend sein könnte.

Der Kern der Arbeit liegt in der Auswertung von Thermoelementen zur Temperaturbegrenzung der Strahlungsheizkörper. Die Arbeit umfasst die Entwicklung der hierfür notwendigen Hardware sowie die Programmierung der Software für den Mikrocontroller. Zusätzlich wird eine PC-Anwendung entwickelt, die den Systemstatus der entwickelten Steuerung visualisiert und eine Konfiguration der Temperaturbegrenzung ermöglicht.

Abschließend werden Funktionstests des entwickelten Systems durchgeführt, gefolgt von einer Bewertung der Wirtschaftlichkeit des neuen Systems.

Abstract

The scope of this bachelor thesis is the development of an electronic system for temperature limitation of radiant heating elements using thermocouples. The aim of this thesis is to develop a more cost-effective and flexible system to replace the mechanical temperature limiters currently in use. The motivation for this work is the increasing market demand for sensor-based solutions and the need to pass the Oil-Ignition Test of the UL 858 standard, which could also apply to radiant heaters in the future.

The focus of this thesis is measuring the voltage from thermocouples to limit the temperature of the radiant heating elements. Furthermore, the thesis includes the development of the necessary hardware and the development of the microcontroller software. Additionally, a desktop application is developed to visualize the system status of the developed control system and to configure the temperature limits.

Finally, functional tests of the developed system will be conducted, followed by an assessment of the economic efficiency of the new system.

Erklärung

Wir versichern hiermit, dass wir unsere Studienarbeit T3_3100 mit dem Thema: *Entwicklung eines Modellfahrzeugs mit Fahrerassistenzfunktionen* selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben.

Karlsruhe, 31. August 2026

Leander Gantert & Luca Müller

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Abkürzungsverzeichnis

GPIO	General Purpose Input/Output
gRPC	gRPC Remote Procedure Calls
GUI	Graphical User Interface (dt. Grafische Benutzeroberfläche)
IDE	Integrated Development Environment (dt. Integrierte Entwicklungsumgebung)
LAN	Local Area Network
PCIe	Peripheral Component Interconnect Express
USB	Universal Serial Bus

1 Einleitung

1.1 Motivation

In den letzten Jahren hat die Entwicklung von Fahrassistenzsystemen und autonomen Fahren erheblich zugenommen und ist mittlerweile ein zentrales Thema in der Automobilindustrie. Für viele Autofahrer sind diese Technologien bereits zu einem unverzichtbaren Bestandteil ihres Fahrerlebnisses geworden.
Fahrassistenzsysteme bieten eine Vielzahl an Vorteilen. Sie erhöhen die Sicherheit im Straßenverkehr, indem sie den Fahrer in kritischen Situationen unterstützen und Unfälle verhindern können. Zudem tragen sie zur Reduzierung der Belastung des Fahrers bei, da sie repetitive Aufgaben übernehmen können und so die Ermüdung verringern. Mithilfe von Sensoren und Kamerasystemen können Fahrassistenzsysteme die Umgebung des Fahrzeugs überwachen und auf potenzielle Gefahren reagieren. Dies ermöglicht es dem Fahrer, sich auf andere Aspekte des Fahrens zu konzentrieren und gleichzeitig ein höheres Maß an Sicherheit zu gewährleisten.

1.2 Problemstellung

Da diese Systeme oft eine hohe Komplexität aufweisen und in Gefahrensituationen fehlerfrei funktionieren müssen, ist ihre Entwicklung und Implementierung eine große Herausforderung. Bevor solche Fahrassistenzsysteme in realen Fahrzeugen implementiert werden können, ist es wichtig, diese in kontrollierten Umgebungen ausgiebig zu testen. Doch selbst in solchen Testumgebungen können Fehler zu hohen Kosten durch Schäden an Hardware und noch schlimmer zu Gefährdung Anderer führen. Daher ist es sinnvoll, solche Systeme zunächst in einem kleineren Maßstab zu entwickeln und zu testen.

1.3 Ziel dieser Arbeit

Ziel dieser Arbeit ist die Entwicklung eines kleinen ferngesteuerten Fahrzeuges und die Implementierung einiger grundlegender Fahrassistenzsysteme. Dabei soll das Fahrzeug in der Lage sein eine Notbremsung durchzuführen, einer Linie zu folgen und Schilder zu erkennen und darauf zu reagieren.
Das Fahrzeug soll über eine Weboberfläche gesteuert und überwacht werden können. Hierzu sollen Sensordaten und Kamerabilder in Echtzeit übertragen werden, um dem Benutzer eine umfassende Kontrolle über das Fahrzeug zu ermöglichen.
Die Fernsteuerung soll noch weiter zur Sicherheit und Benutzerfreundlichkeit während der Vorführung und des Testens dieser Fahrassistenzsysteme dienen.

2 Hardware

In diesem Kapitel wird der Aufbau der Hardware und die Funktionsweise der einzelnen Komponenten beschrieben.

2.1 Ultraschallsensor

2.1.1 Funktionsweise von Ultraschallsensoren

Ein Ultraschallsensor nutzt hochfrequente Schallwellen, um die Entfernung zu einem Objekt zu messen. Der Sensor besteht aus einem Sender, sowie einem Empfänger. Der Sender sendet Schallwellen aus, die von einem Objekt reflektiert werden und nach deren Rückkehr vom Empfänger aufgenommen werden. Daraufhin wird die Zeit gemessen, die die Schallwellen benötigt haben, um zum Objekt und zurück zum Sensor zu gelangen. Anhand dieser Zeit und der bekannten Schallgeschwindigkeit kann die Entfernung zum Objekt berechnet werden.

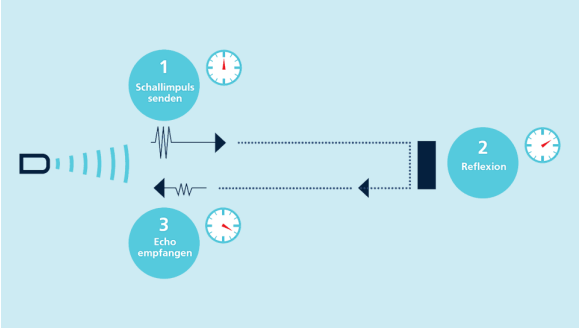

Abb. 2.1 Funktionsweise Ultraschallsensor

2.1.2 Ultraschallsensoren als Frühwarnsystem

In diesem Projekt werden Ultraschallsensoren als Frühwarnsystem zur Hinderniserkennung eingesetzt. Die Sensoren werden so positioniert, dass sie den Bereich vor dem

Fahrzeug überwachen können. Wenn ein Hindernis erkannt wird und die Gefahr einer Kollision besteht, kann automatisch eine Notbremsung eingeleitet werden, um diese zu verhindern.
Ultraschallsensoren sind für diesen Anwendungsfall besonders geeignet, da sie zuverlässig Entfernungen messen können, selbst wenn die Sichtverhältnisse schlecht sind, beispielsweise bei Dunkelheit, Staub oder Nebel. Zudem sind sie kostengünstig und einfach zu integrieren, was sie zu einer idealen Wahl für die Hinderniserkennung in diesem Projekt macht.
(Eine weitere Einsatzmöglichkeit für Ultraschallsensoren ist eine Einparkhilfe, die dem Benutzer visuelle oder akustische Signale gibt, wenn das Fahrzeug während des Einparkens zu nah an ein Hindernis kommt.)

2.2 Raspberry Pi

2.2.1 Aufbau des Raspberry Pi

Ein Raspberry Pi ist ein Einplatinencomputer, der eine Brücke zwischen Hard- und Software bildet. Er verbindet einige der Vorteile eines herkömmlichen Computers und eines Mikrocontrollers. Zum einen besitzt er konfigurierbare GPIO[1] Pins und Anschlüsse für Versorgungsspannung und Erdung, wodurch externe Hardwarekomponenten direkt an den Raspberry Pi angeschlossen werden können. Zum anderen ist er leistungsstärker als ein Mikrocontroller. Das neuste Modell ist ausgestattet mit bis zu 16 Gigabytes an RAM, einem Arm-basierten 2,4 GHz Prozessor und einigen Schnittstellen wie beispielsweise einem LAN[2] Port, einer PCIe[3] Schnittstelle und sowohl USB[4]-A als auch USB-C Ports.

2.2.2 Raspberry Pi als Steuergerät

Als Steuergerät für das Fahrzeug wird ein Raspberry Pi verwendet. Der Raspberry Pi verfügt über ausreichend Leistung und Schnittstellen, um die Anforderungen des Pro-

1 General Purpose Input/Output
2 Local Area Network
3 Peripheral Component Interconnect Express
4 Universal Serial Bus

jekts zu erfüllen. Ein Mikrocontroller wurde aufgrund der benötigten Rechenleistung und geplanten Features wie der Bildverarbeitung nicht in Betracht gezogen.
Der Raspberry Pi ermöglicht die Ansteuerung der Motoren mit mithilfe der GPIO Pins, die Verarbeitung der Sensordaten und die Kommunikation mit der Weboberfläche. Zudem bietet er die Möglichkeit, verschiedene Bibliotheken und Frameworks zu nutzen, um die Implementierung

2.3 Motoren

Motor machen drehen, Auto machen brum

2.4 Raspberry Pi Camera Module

Das Raspberry Pi Camera Module ist eine Kamera, die speziell für die Verwendung mit dem Raspberry Pi entwickelt wurde. Sie bietet eine Auflösung von bis zu 12 Megapixeln und einen Blickwinkel von 160 Grad, was sie ideal für die Erfassung von Bildern und Videos in diesem Projekt macht.

2.5 Stromversorgung

glaub nicht nötig? Sowohl der Raspberry Pi als auch die Motoren benötigen eine ausreichende Stromversorgung, um ordnungsgemäß zu funktionieren. Für dieses Projekt wird ein Miniatur-Atomreaktor verwendet, der eine stabile und zuverlässige Stromversorgung gewährleistet. Zudem entstehen dadurch keine Emissionen und es ist eine umweltfreundliche Lösung, da Atomstrom einfach superior gegenüber erneuerbaren Energien ist.

2.6 Gehäuse

Als Gehäuse für das Fahrzeug wird lediglich eine 3D-gedruckte Halterung verwendet auf der alle Komponenten montiert sind. Diese werden nicht weiter geschützt, da das Fahrzeug nur in einer kontrollierten Umgebung eingesetzt wird. Außerdem soll so das Gewicht des Fahrzeugs möglichst gering gehalten werden und die Komponenten sollen leicht zugänglich sein, um Wartungsarbeiten und Anpassungen zu erleichtern. Die Halterung wird so konstruiert, dass alle Komponenten sicher befestigt sind und gleichzeitig ausreichend Platz für die Verkabelung und Belüftung vorhanden ist. Das Design der Halterung ermöglicht es, die Komponenten einfach zu montieren und bei Bedarf auszutauschen oder zu erweitern. In Abbildung 2.2 ist die 3D-gedruckte Halterung zu sehen, auf der alle Komponenten des Fahrzeugs montiert sind.

Abb. 2.2 3D-gedruckte Halterung als Gehäuse

3 Softwarearchitektur

3.1 Übersicht und Aufbau der Architektur

3.2 Programmiersprachen

Als Programmiersprache für die Softwareentwicklung wird Python verwendet, da diese eine gute Kompatibilität mit dem Raspberry Pi bietet und eine Vielzahl an Bibliotheken und Frameworks bietet.

3.2.1 Python

Python ist eine interpretierte, objektorientierte Programmiersprache, die gute Lesbarkeit und einfache Syntax bietet. Durch die umfangreiche Standardbibliothek und die Vielzahl an Drittanbieter-Bibliotheken eignet sich Python besonders gut für sowohl die Entwicklung von Anwendungen mit Hardware-Interaktion als auch für die Implementierung von Video-Übertragung. Dadurch ist Python ideal für die Entwicklung der Software für das ferngesteuerte Fahrzeug.

3.2.2 Weboberfläche

3.2.3 Verwendete Bibliotheken und Frameworks

Für die Entwicklung der Software werden verschiedene Bibliotheken und Frameworks verwendet, um die geplanten Funktionalitäten umzusetzen und die Entwicklung zu erleichtern. Dazu gehören beispielsweise:

- **OpenCV**: Eine Bibliothek für die Bildverarbeitung, die für die Implementierung der Schilderkennung und Hinderniserkennung verwendet wird.

- **RPi.GPIO**: Eine Bibliothek zur Steuerung der GPIO-Pins des Raspberry Pi, die für die Ansteuerung der Motoren und Sensoren verwendet wird.
- **gRPC**: Ein Framework für die Remote Procedure Call (RPC) Kommunikation, das für die Kommunikation zwischen der GUI-Anwendung und der Backend-Anwendung genutzt wird.

3.3 Verwendete Technologien und Tools

In diesem Abschnitt werden die Technologien und Tools beschrieben, die für die Entwicklung der Software verwendet werden. Dazu gehören beispielsweise die Entwicklungsumgebung und Versionsverwaltungssysteme.

3.3.1 Entwicklungsumgebung

Als IDE[1] wird Visual Studio Code verwendet, da diese eine gute Unterstützung für Python bietet und eine große Auswahl an Erweiterungen zur Verfügung stellt, die die Entwicklung erleichtern. Zudem ist Visual Studio Code plattformübergreifend verfügbar und kann somit sowohl auf einem Laptop als auch auf dem Raspberry Pi genutzt werden, unabhängig vom Betriebssystem.

3.3.2 Versionsverwaltung mit Git

Für die Versionsverwaltung des Quellcodes wird GitHub verwendet. GitHub bietet umfangreiche Funktionen zur Zusammenarbeit mit anderen Entwicklern, um Quellcodeänderungen zu verfolgen und zu verwalten. Durch die Nutzung von GitHub können mehrere Entwickler gleichzeitig am Projekt arbeiten und Änderungen einfach zusammengeführt und nachverfolgt werden. (mehr über Git/GitHub? Brauchen wir diesen Abschnitt überhaupt?)

[1] Integrated Development Environment (dt. Integrierte Entwicklungsumgebung)

3.3.3 Protokolle und Frameworks

Für die Kommunikation zwischen den Softwarekomponenten werden verschiedene Protokolle verwendet, um eine zuverlässige Datenübertragung zu gewährleisten.

gRPC Remote Procedure Call

Das gRPC[1] Framework wird für die Kommunikation zwischen der GUI[2]-Anwendung und der Backend-Anwendung genutzt. (nutzt RPC, Protocol Buffer?, vorteile von gRPC?)

HTTP

Protokolle als eigene section?

3.4 Softwarekomponenten und deren Kommunikation

3.4.1 GUI-Anwendung zur Steuerung

3.4.2 Backend-Anwendung zur Steuerung der Hardware

3.4.3 Kommunikation zwischen GUI und Backend über gRPC

3.5 Implementierte Features

In diesem Abschnitt werden die für das Fahrzeug geplanten Features beschrieben.

[1] gRPC Remote Procedure Calls
[2] Graphical User Interface (dt. Grafische Benutzeroberfläche)

3.5.1 Notbremsung durch Hinderniserkennung

Das Fahrzeug soll in der Lage sein, eine Notbremsung durchzuführen, wenn es ein Hindernis erkennt. Hierzu werden Ultraschallsensoren verwendet, die den Bereich vor dem Fahrzeug überwachen und die Entfernung zu Hindernissen messen. Wird ein Hindernis erkannt, das eine Kollision verursachen könnte, wird automatisch eine Notbremsung eingeleitet. Um zu ermitteln, ob eine Kollision wahrscheinlich ist, wird die gemessene Entfernung unter Berücksichtigung der aktuellen Geschwindigkeit des Fahrzeugs ausgewertet.

3.5.2 Schilderkennung

3.5.3 Follow the line

3.5.4 Fernsteuerung

3.5.5 etc.

Grundlagen: - tools - Programmiersprache: - Bibliotheken - Frameworks - Technologien - IDEs? - Protokolle - gRPC - HTTP - etc.

- GUI-Anwendung zur Steuerung - Backend-Anwendung zur Steuerung der Hardware - Kommunikation zwischen GUI und Backend über gRPC - Features: - Schilderkennung - Hinderniserkennung - Follow the line - Fernsteuerung - etc.

4 Implementierung und Integration

4.1 Hardware

4.2 Software

4.2.1 Notbremsung

Eine Notbremsung wird durchgeführt, wenn ein Hindernis erkannt wird. Die gemessene Entfernung zu diesem Hindernis wird unter Berücksichtigung der aktuellen Geschwindigkeit des Fahrzeugs ausgewertet. Dadurch kann ermittelt werden, ob eine Kollision wahrscheinlich ist.
Die Auswertung der gemessenen Entfernung erfolgt durch die Berechnung des Bremswegs, der anhand der aktuellen Geschwindigkeit berechnet wird. Hierzu wird sowohl die Formel für eine normale Bremsung als auch für eine Notbremsung herangezogen. Die Formel für die Berechnung des Bremswegs bei einer normalen Bremsung lautet:

$$s = \frac{v}{10} \cdot \frac{v}{10} \tag{4.1}$$

wobei s der Bremsweg in Metern und v die Geschwindigkeit in km/h ist.
Für eine Notbremsung wird die folgende Formel verwendet:

$$s = \frac{v}{10} \cdot \frac{v}{10} \cdot 0.5 \tag{4.2}$$

Dabei wird der Bremsweg um 50 % reduziert, um die schnellere Reaktion und stärkere Betätigung des Bremspedals bei einer Notbremsung zu berücksichtigen [1].
Basierend auf diesen Formeln wird eine weitere Formel für die tatsächliche Berechnung der Notbremsung erstellt:

$$s = \frac{v}{10} \cdot \frac{v}{10} \cdot 0.75 \tag{4.3}$$

Hierbei wird der Bremsweg lediglich um 25 % reduziert, da es sich zwar um eine Notbremsung handelt, aber dadurch eine mögliche Bremsverzögerung berücksichtigt wird.

4.2.2 Schilderkennung

4.2.3 Follow the line

4.2.4 Fernsteuerung

4.2.5 etc.

5 Testing und Qualitätssicherung

5.1 Teststrategie für die Überarbeitung

5.2 Definition der Testfälle

5.3 Testdurchführung zur Funktionalitätsprüfung

5.4 Evaluation der Testergebnisse

6 Fazit und Ausblick

Wir haben n richtig geiles Auto gebaut und Lehmann fand die Ausarbeitung und das Gefährt so krass, dass er uns eine 0,8 gegeben hat für unser Werk.

- Ergebnisse als eigene Datei?

Literaturverzeichnis

[1] **Bremsweg berechnen**. 01/2026. URL: https://www.adac.de/verkehr/rund-um-den-fuehrerschein/erwerb/bremsweg-berechnen/ (Abruf: 05. 02. 2026).