

Weiterentwicklung eines selbstfahrenden Fahrzeuges mit Lidar und anderen Sensoren

Studienarbeit

über die ersten drei Quartale des 3. Studienjahres

an der Fakultät für Technik
im Studiengang Informationstechnik

an der DHBW Ravensburg
Campus Friedrichshafen

von

Justin Serrer - 5577068 - TIT21
Timo Waibel - 8161449 - TIT21
Janik Frick - 4268671 - TIT21

Sperrvermerk

gemäß Ziffer 1.1.13 der Anlage 1 zu §§ 3, 4 und 5 der Studien- und Prüfungsordnung für die Bachelorstudiengänge im Studienbereich Technik der Dualen Hochschule Baden-Württemberg vom 29.09.2017 in der Fassung vom 25.07.2018.

„Der Inhalt dieser Arbeit darf weder als Ganzes noch in Auszügen Personen außerhalb des Prüfungsprozesses und des Evaluationsverfahrens zugänglich gemacht werden, sofern keine anders lautende Genehmigung vom Dualen Partner vorliegt.“

Ort, Datum

Unterschrift

Selbständigkeitserklärung

gemäß Ziffer 1.1.13 der Anlage 1 zu §§ 3, 4 und 5 der Studien- und Prüfungsordnung für die Bachelorstudiengänge im Studienbereich Technik der Dualen Hochschule Baden-Württemberg vom 29.09.2017 in der Fassung vom 25.07.2018.

Ich versichere hiermit, dass ich meine Hausarbeit mit dem Thema

Weiterentwicklung eines selbstfahrenden Fahrzeuges mit Lidar und anderen Sensoren

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

Ort, Datum

Unterschrift

Gender-Erklärung

Das in dieser Arbeit gewählte generische Maskulinum bezieht sich zugleich auf die männliche, die weibliche und andere Geschlechteridentitäten. Zur besseren Lesbarkeit wird auf die Verwendung männlicher und weiblicher Sprachformen verzichtet. Alle Geschlechteridentitäten werden ausdrücklich mitgemeint, soweit die Aussagen dies erfordern.

Inhaltsverzeichnis

Abkürzungsverzeichnis

GPIO general-purpose input/output

LiDAR Light Detection and Ranging

ROS Robot Operating System

Abbildungsverzeichnis

Listings

5.1	Beispiel Listing	12
-----	----------------------------	----

1 Einleitung

Die Automatisierung im Straßenverkehr befindet sich in stetigem Wachstum, wobei selbstfahrende Fahrzeuge zunehmend an Bedeutung gewinnen.

Auch im Bereich der Modell-Autos und Roboter ist Automatisierung eine präsentes Thema.

Zwar ist die Umsetzung eines selbstfahrenden Modell-Autos oder Roboters, aufgrund der vorhersehbareren und weniger komplexen Umgebung, einfacher als die Umsetzung eines selbstfahrenden PKW, jedoch stellt sie trotzdem eine Herausforderung dar.

2 Problemstellung, Ziel und Umsetzung

In diesem Abschnitt wird auf die Problemstellung, das generelle Ziel und die geplante Umsetzung der Arbeit eingegangen. Des weiteren wird Erläutert, weshalb das Ziel der Arbeit wichtig ist, wie die Arbeit aufgebaut ist und welche Probleme und Schwierigkeiten durch bereits getätigte Versuche einer Umsetzung des Arbeits-Ziels bereits bekannt sind.

2.1 Problemstellung

Bisherige Versuche, ein selbstfahrendes Auto, im Rahmen einer Studienarbeit zu entwerfen und einen entsprechenden Algorithmus zu programmieren, sind gescheitert. Die beiden Hauptprobleme der bisherigen Arbeiten, war zum einen der Bau eines geeigneten Fahrzeugs und zum Anderen die Entwicklung einer Software, welche es dem Auto ermöglicht, ohne manuelle Steuerung, Hindernisse zu erkennen und um diese herum zu navigieren. Da es sich bei den Studenten der bisherigen Gruppen ausschließlich um Studenten mit einem Schwerpunkt in Elektrotechnik handelte, war die Entwicklung und Implementierung der Software zum autonomen Fahren die größere Herausforderung.

Da keine der bisherigen Gruppen, das Problem der Software lösen konnte, wurde sich dazu entschieden, die Aufgaben aufzuteilen. Die Aufgabe, der Erstellung eines funktionsfähigen Modell-Autos und einer Schnittstelle, zur Steuerung des Autos, wurde einer Gruppe von Studenten mit einem Schwerpunkt in Elektrotechnik zugeteilt. Somit ist das Hauptproblem dieser Arbeit, die Entwicklung und Implementierung eines Ausweichalgorithmus, welcher das Auto, über die, von der anderen Gruppe zur Verfügung gestellten Schnittstelle, steuern soll und so eine autonome Hindernis-Detektierung und Vermeidung ermöglicht.

Da die Aufgabe auf mehrere Gruppen aufgeteilt wurde, ist ein weiteres Problem die Kommunikation zwischen den Gruppen. Um einen reibungslosen und effizienten Ablauf gewährleisten zu können, sollte diese möglichst umfangreich sein.

Ein weiteres Problem, welches durch die Aufteilung auf verschiedene Gruppen entsteht, ist die eingeschränkte Verfügbarkeit der Hardware. Da die Gruppe, welche den Hardware-Teil der Aufgabe übernimmt, diese erst bauen und anschließend auch testen muss, ist sie nur eingeschränkt verfügbar. Daher ist zur Entwicklung der Software, eine Abstrahierung der Hardware notwendig. Konkret bedeutet das, dass die Schnittstelle zur Steuerung, sowie die Daten der Sensoren simuliert werden müssen, um ein Testen des Algorithmus auch ohne Verfügbarkeit der Hardware zu ermöglichen.

2.2 Ziel

Das Hauptziel der Arbeit ist die Entwicklung und Implementierung eines Algorithmus für ein Modell-Auto. Dieser soll mit Hilfe der Daten diverser Sensoren, Hindernisse erkennen und das Auto, unter Verwendung einer bereitgestellten Schnittstelle, um die Hindernisse herum navigieren.

Da der Hardware-Teil der Arbeit von einer anderen Gruppe an Studenten übernommen wird, ist es notwendig, die Hardware zu abstrahieren um so ein Testen des Algorithmus möglich zu machen. Somit ist das Entwickeln einer solchen Simulation der Hardware ein weiteres Ziel der Arbeit.

2.3 Umsetzung

Um eine umfangreiche Kommunikation zwischen den Gruppen zu ermöglichen, müssen Kommunikationswege so früh wie möglich erstellt werden. Des Weiteren sollten Termine für regelmäßige Meetings festgelegt werden, um einen konstanten Austausch von Informationen zwischen den Gruppen zu gewährleisten.

Zur Umsetzung der Aufgabe selbst, stehen, neben einem RPLiDAR A1M8-R6 der Firma Slamtec, auch weitere Sensoren, wie Ultraschall-, Lenkwinkel- und Geschwindigkeits-Sensoren, sowie ein Raspberry PI 4 zur Verfügung. Bevor die eigentliche Arbeit an einem Algorithmus beginnen kann, muss die gegebene Hardware getestet werden. Zudem ist es, um das weitere Vorgehen planen zu können, notwendig, sich mit der Hardware vertraut zu machen. Zu wissen, welche Daten von den Sensoren, wann gesendet werden, ermöglicht es, präziser zu planen, wodurch die Entwicklung des Algorithmus effizienter wird.

Nachdem verstanden wurde, wie die Hardware funktioniert, muss eine Möglichkeit, diese zu simulieren, entwickelt werden. Hierbei ist es wichtig, die, für den Algorithmus notwendige Hardware, so genau wie möglich zu simulieren. Je genauer die Simulation ist, desto unwahrscheinlicher treten Probleme bei der Zusammenführung von Hard- und Software auf.

3 Überblick Hardware und Software

In diesem Kapitel wird auf die verschiedene Hardware und Software, welche zur Entwicklung des Ausweichalgorithmus verwendet werden kann, eingegangen.

3.1 Hardware

Dieser Abschnitt beschreibt die Hardware, welche für die Entwicklung des Ausweichalgorithmus relevant ist.

1. Raspberry Pi 4 Model B

Der Raspberry Pi 4 ist ein single-board-computer, welcher im Jahr 2019 auf dem Markt erschien. Er besitzt eine ARM-basierte 64-bit CPU, welche mit 1.5GHz getaktet ist. Das Modell, welches im Rahmen unserer Studienarbeit genutzt wird, besitzt 4GB Arbeitsspeicher. Außerdem verfügt der Raspberry Pi 4 über 40 **gpio!** (**gpio!**) Pins, welche zur Kommunikation mit den Sensoren und der Steuerungs-Schnittstelle genutzt werden können. [RasPi2024]

2. Slamtec RPLiDAR A1M8-R6

Der RPLiDAR A1M8-R6 von Slamtec ist ein zweidimensionaler Laser-Scanner, welcher mittels **lidar!** (**lidar!**), ein 360° Scan der Umgebung erstellen kann. [Slamtec2020] Er hat eine effektive Reichweite von 0.15 bis 12 Meter und bei einer Scan-Rate von 5.5 Scans pro Sekunde, sowie eine Scan-Frequenz von 8000 Hz, eine Auflösung von weniger als einem Grad. [Slamtec2020]

3. Weitere Sensoren

Da der **lidar!**-Sensor nur zweidimensionale Scans macht, können Hindernisse, welche kleiner wie die Scan-Höhe des **lidar!** sind, von diesem nicht erfasst werden. Daher sind weitere Sensoren, wie z.B. Ultraschall-Sensoren notwendig, um auch niedrige Hindernisse erkennen zu können. Außerdem wäre eine Sensor zur Bestimmung des aktuellen Lenkwinkels und ein weiterer Sensor zum Bestimmen der aktuellen Geschwindigkeit sinnvoll. Die Daten dieser Sensoren könnten bei der Ermittlung der Position im Raum von Nutzen sein.

3.2 Software

In diesem Abschnitt wird auf die Software eingegangen, welche zur Entwicklung des Algorithmus zur Verfügung steht.

1. **ros! (ros!)**

ros! ist eine Ansammlung von Werkzeugen und Bibliotheken, wie Treiber und Algorithmen, welche bei der Entwicklung von Roboter-Anwendungen helfen sollen. Hierbei ist ROS vollständig Open-Source und bietet zudem eine ausführliche Dokumentation, Foren und eine große Community. [Ros2024] Des Weiteren bietet Slamtec, der Hersteller des zur Verfügung stehenden **lidar!**-Sensors, eine Bibliothek, zur Nutzung des **lidar!**-Sensors, in Kombination mit verschiedenen Versionen des **ros!** an. [RplidarRos2023]

2. **Slamtec RPLIDAR Public SDK**

Slamtec bietet, neben der **ros!**-Bibliothek, auch eine öffentlich zugängliche SDK für sämtliche RPLiDAR-Produkte an. Die SDK ist in C++ geschrieben und unter der BSD 2-clause Lizenz lizenziert. [RplidarSDK2023]

4 SLAM-Algorithmus

1. Was ist SLAM?
2. Wie funktioniert SLAM?
3. Welche SLAM-Algorithmen gibt es?
4. Welcher SLAM-Algorithmus ist für uns geeignet?

5 Simulation des Ausweichalgorithmus

1. Warum simulieren?
2. Was simulieren?
3. Woher bekommen wir die Daten?
4. Wie werden die Daten verarbeitet?
5. Wie werden die Daten visualisiert?
6. Wie wird der Algorithmus getestet und validiert

5.1 Was ist eine Simulation?

Bevor damit begonnen werden kann die verschiedenen Aspekte einer Simulation zu beleuchten, ist zu klären, was eine Simulation ist. Nach der Aussage von A. Maria ist eine Simulation eine Ausführung eines Modells eines Systems [maria1997introduction][p. 1, ch. 2]. Der Begriff des Modells wird ebenfalls in der Arbeit beschrieben. Ein Modell ist eine vereinfachte, funktionierende Repräsentation des Systems, das betrachtet werden soll [maria1997introduction][p. 1, ch. 1].

In der Simulationstechnik gibt es unterschiedliche Arten von Simulation. In diesem Kontext von Bedeutung ist die Unterscheidung zwischen realer Simulation und Computersimulation. Reale Simulationen kommen zum Einsatz, wenn durch einen Fehler keine Gefahr für Personen und Umwelt besteht. Außerdem kann es sein, dass ein Nachstellen der Umweltbedingungen so komplex ist, dass eine Nachbildung am Computer nicht ausreichend möglich oder zeitlich zu aufwendig ist. Computersimulationen kommen dann zum Einsatz, wenn ein Fehler schädliche Folgen für Personen und Umwelt herbeiführen könnten und sich die Einflussfaktoren auf das System am Computer nachahmen lassen [Britannica2023]. Eine Computersimulation kann auch dann genutzt werden, wenn das Erstellen eines realen Modells nicht möglich oder nicht rentabel ist. Ein weiterer Anwendungsfall einer Computersimulation tritt ein, wenn das reale Modell noch in der Entwicklungsphase ist. In diesen Fällen stellt die Simulation sicher, dass erste Versuche mit Algorithmen, die unabhängig vom Modell funktionieren, möglich sind. Dadurch kann damit begonnen werden an Technologien und Methodiken zu arbeiten, ohne auf eine reale Umsetzung warten zu müssen.

5.2 Simulation im Kontext eines Ausweichalgorithmus

Im Rahmen dieser Arbeit ist eine Simulation geplant. Die Konstellation des Teams das am selbstfahrenden Modell-Auto beteiligt ist, begünstigt den Einsatz einer Simulation. Der verfügbare Zeitraum für Hard- und Software ist gleich lang. Aus diesem Grund ist es nicht möglich mit der Entwicklung des Ausweichalgorithmus zu warten, bis das reale Modell-Auto einsatzbereit ist. Wie im vorherigen Kapitel beschrieben, ist in solchen Fällen eine Simulation ein guter Weg für eine simultane Entwicklung der Algorithmik und des Modells. Die Simulation hilft dabei den Algorithmus zu entwickeln und in einer ersten Form zu validieren. Neben dem zeitlichen Faktor ist auch ein möglicher Schaden im Fehlerfall ein Grund für den Einsatz einer Simulation. Würde es in einem Test zu einem Fehler kommen, könnte das Modell-Auto je nach Szenario so beschädigt werden, dass eine aufwendige Reparatur notwendig oder sogar unmöglich wäre. Dadurch würde die weitere Entwicklung verzögert werden. Um dieses Szenario zu vermeiden ist es wichtig den Algorithmus mit Hilfe der Simulation so weit zu entwickeln, dass die Chance für das Eintreten eines solchen Szenarios möglichst gering ist. Auch für den Fall, dass das Modell am Ende nicht einsatzbereit ist, sorgt die Simulation dafür, dass der Ausweichalgorithmus zumindest im Rahmen einer Simulation überprüft und getestet werden kann. Zusätzlich sorgt die Simulation für einen reduzierten Kommunikationsaufwand, denn die Entwicklung und Verwendung der Simulation erfolgt unabhängig vom Aufbau des Modells.

Der Einsatz einer Simulation scheint in diesem Kontext eine gute und einfache Option zu sein, wie Hard- und Software parallel zueinander entwickelt werden können. Allerdings gibt es auch Faktoren die eine Simulation erschweren. Der Aufbau einer Umgebung in der der Algorithmus erprobt und entwickelt werden soll, ist grundlegend wenig herausfordernd. Allerdings muss die Umgebung so aufgebaut werden, dass für diese Umgebung Sensordaten generiert werden können, mit denen der Algorithmus ausgeführt werden kann. Zusätzlich müssen die simulierten Daten in Umfang und Frequenz mit denen des späteren Modells übereinstimmen, um die Funktionalität später auf das Modell übertragen zu können.

5.3 Aufbau der Simulation

Um die Simulation für eine erste Validierung der Algorithmik nutzen zu können, ist die Voraussetzung dass die simulierte Funktionalität auch der Funktionalität entspricht, die später im realen Einsatz genutzt wird. Gibt es Abweichungen zwischen den Funktionalitäten, so kann nur eine Schätzung

vorgenommen werden, wie sich der Algorithmus im praktischen Einsatz verhält.

Wie zu Beginn der Arbeit formuliert, soll zunächst ein Algorithmus entwickelt werden, der das Ausweichen auf gerader Linie ermöglicht. Da es aktuell keine Möglichkeit gibt, ein Ziel einzugeben, wird die aktuelle Ausrichtung des Fahrzeugs als Fahrtrichtung verwendet. Das ermöglicht eine Funktionalität unabhängig von der Position des Fahrzeugs. Dann fährt das Auto in diese Richtung. Wird auf der Strecke ein Hindernis erkannt, versucht der Algorithmus dem Hindernis auszuweichen. Ist dies möglich wird das Auto auf die ursprüngliche Gerade zurückgeführt.

Um eine Simulation zu implementieren, ist zu klären, ob diese Funktionalität in einer Simulation realitätsnah möglich ist, oder ob eine vereinfachte Version simuliert werden muss.

5.3.1 Kritische Funktionalitäten

Zunächst müssen die Funktionalitäten identifiziert werden, die in der Simulation Probleme verursachen könnten.

1. Sensordaten
2. Umgebung
3. Ausweichen
4. Lokalisierung
5. Fahrzeug

Sensordaten Die Sensordaten bilden die Grundlage für die gesamte Simulation. Auf den Sensordaten basiert die Lokalisierung im Raum und das Erkennen und Ausweichen eines Hindernisses. Die Simulation dieser Daten stellt damit die größte Herausforderung in der Simulation dar, da die Daten in Frequenz und Aufbau den realen Daten möglichst genau entsprechen sollten. Vor allem der Aufbau der Daten sollte den realen Daten so nahe wie möglich kommen, da zusätzliche oder fehlende Daten in der Qualität der Auswertung deutlich zu erkennen sein könnten. Gibt es Unterschiede in der Frequenz sind die Auswirkungen weniger problematisch. Eine geringere oder höhere Frequenz kann durch die möglichen Geschwindigkeiten des Modell-Autos oder an anderen Stellen ausgeglichen werden.

Umgebung Die Umgebung ist ebenfalls ein essentieller Bestandteil. Denn die Umgebung muss so simuliert werden, dass diese von den Sensoren erkannt werden kann. Ist das nicht der Fall, ist jede Simulation der Sensorik unbrauchbar. Die genaue Implementierung ist unabhängig von der realen Welt. Dennoch ist darauf zu achten, dass die für die Sensorik notwendigen vorhanden sind.

Ausweichen Die Logik des Ausweichens ist die eigentliche Funktionalität die implementiert werden soll. Diese ist der zentrale Baustein im System. Ist diese nicht vorhanden, ist die Simulation nicht brauchbar, um die Funktionalität zu validieren.

Lokalisierung Die Lokalisierung des simulierten Fahrzeugs ist ebenfalls ein wichtiger Bestandteil der Simulation, denn die Lokalisierung ist dafür verantwortlich zu entscheiden ob das Ausweichmanöver abgeschlossen ist.

Fahrzeug Das Fahrzeug dient in der Simulation nur der Visualisierung. Physische Eigenschaften wie zum Beispiel der Kurvenradius und Reibung der Reifen sind nicht von Bedeutung. Außerdem kann das Auto visuell stark vereinfacht dargestellt werden.

5.3.2 Prüfung der kritischen Funktionalitäten

In diesem Abschnitt werden die einzelnen Funktionalitäten auf Umsetzbarkeit geprüft. Ist eine Umsetzung möglich, kann diese Funktionalität so implementiert werden, andernfalls muss eine Alternative erarbeitet werden.

Sensordaten Folgende Sensordaten müssen simuliert werden:

- LIDAR: Der Lidar Sensor erzeugt durch Rotation und das Aussenden von Laserstrahlen eine Punktwolke die die Umgebung abbildet. Diese Daten sind gut zu simulieren, da hierfür ausgehend von der aktuellen Position in jede Richtung Strahlen simuliert werden können, die die Entfernung von Objekten bestimmen.
- Ultraschall: Bei der Implementierung der Simulation des Ultraschall-Sensors ist mehr Aufwand nötig, da der Sensor nicht in jede Richtung Signale sendet, sondern konstant in eine Richtung. + Die Komplexität kommt durch das Verhalten der Schallwellen, die sich im Raum ausbreiten. Trotz der zusätzlichen Komplexität sollte eine ausreichend gute Simulation möglich sein.
- Geschwindigkeit: Die Komplexität der Ermittlung der Geschwindigkeit ist

gering, da diese Information als Variable direkt im simulierten Auto hinterlegt werden kann.

- Lenkwinkel: Kann ähnlich wie die Geschwindigkeit als Variable gespeichert werden.

Umgebung Die Komplexität einer vereinfachten Umgebung ist als gering einzuschätzen, da für die Erkennung die Positionsdaten, sowie geometrische Form und Längen- und Breitenmaße ausreichend sind, um eine Überschneidung mit Strahlen oder Wellen mathematisch zu berechnen. Die vereinfachte Umgebung enthält nur Gegenstände, deren Form eine einfache Berechnung zulässt. Dazu gehören zum Beispiel Quadrate und Rechtecke.

Soll die Komplexität der Umgebung erhöht werden, steigt der Aufwand der mathematischen Berechnungen. Die Komplexität der Simulation sollte aber auf einem ähnlichen Niveau bleiben.

Ausweichen Das Ausweichen weist eine geringe Komplexität auf, da es für diese Aufgabe bereits viele verschiedene Lösungsstrategien gibt.

Lokalisierung Die Lokalisierung erfolgt auf Basis der empfangenen Sensordaten. Die Korrektheit ist abhängig von der Qualität der empfangenen Daten. Aus diesem Grund ist an dieser Stelle mit der größten Abweichung von Simulation und Realität zu rechnen. Daraus resultiert die Herausforderung die Position an so vielen Parametern wie möglich zu bestimmen. Denn je mehr Parameter genutzt werden, desto geringer der Einfluss von verfälschten Daten.

Fahrzeug Das Fahrzeug ist wenig komplex in der Implementierung, da eine vereinfachte Visualisierung, zum Beispiel mit einem Punkt, der sich bewegen kann, ausreichend ist.

5.3.3 Auswertung der Prüfung

Auf Basis der vermuteten Umsetzbarkeit der kritischen Funktionalitäten in Kapitel ?? stellt sich die Simulation des Ausweichalgorithmus und der dazugehörigen Komponenten als machbar heraus.

Es gibt Teilaufgaben in der Simulation deren Komplexität höher ist, als die anderer Bestandteile, aber auch diese befinden sich in einem Umfang der umsetzbar ist.

```
class a {  
};
```

Listing 5.1: Beispiel Listing



Abbildung 1: Beispiel Bild

6 Implementierung des Ausweichalgorithmus

1. Zugriff auf die Daten
2. Verarbeitung der Daten
3. Ausweichalgorithmus
4. Welche Technologien werden verwendet?
5. Validierung und Tests?

7 Fazit

1. Was haben wir erreicht?
2. Entspricht das dem erhofften Ergebnis?
3. Welche Einschränkungen gibt es in der Verwendung?

A Anhang