Universität der Bundeswehr München
Fakultät für Luft- und Raumfahrttechnik
Institut für Technik Autonomer Systeme

Kooperative Lokalisierung und Kartierung in unstrukturierter Umgebung

Patrick Burger, M.Sc.

Vollständiger Abdruck der von der Fakultät für Luft- und Raumfahrttechnik der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

1. Gutachter: Univ.-Prof. Dr.-Ing. Hans-Joachim Wünsche
2. Gutachter: Univ.-Prof. Dr.-Ing. Uwe Stilla

Diese Dissertation wurde am 06.12.2021 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Luft- und Raumfahrttechnik am 15.04.2022 angenommen. Die mündliche Prüfung fand am 04.05.2022 statt.

Vorwort

Die vorliegende Dissertation entstand während meiner Tätigkeit am Institut für Technik Autonomer Systeme der Fakultät für Luft- und Raumfahrttechnik an der Universität der Bundeswehr München.

Dem Betreuer dieser Arbeit, Herrn Prof. Dr.-Ing. Hans-Joachim Wünsche, danke ich herzlich für den Freiraum, den er mir bei der Wahl meines Forschungsthemas gegeben hat, für die Schaffung von ausgezeichneten Rahmenbedingungen und für die fortlaufende Unterstützung meiner wissenschaftlichen Arbeit.

Bei meinen Kolleginnen und Kollegen bedanke ich mich für die offene und angenehme Arbeitsatmosphäre, die große Hilfsbereitschaft, sowie für die zahlreichen und spannenden Diskussionen. Besonderer Dank geht an Benjamin Naujoks, Philipp Berthold, Felix Ebert, Martin Michaelis, Carsten Fries, Lukas Beer, Matthias Schmitz und Lukas Roth für die intensive Zusammenarbeit. Es war mir eine Freude!

Mein ganz besonderer Dank gilt meiner Partnerin Alisa, meiner Familie und meinen Freunden, die mit ihrer Unterstützung und Motivation maßgeblich zum Gelingen dieser Arbeit beigetragen haben.

Patrick Burger

Kurzfassung

Gängige SLAM (Simultaneous Localization And Mapping)-Verfahren verwenden keine Vorabinformationen über die Umwelt. Sie schätzen die wahrscheinlichste Karte und Position aus einer Reihe von Beobachtungen. Um den akkumulierten Positionsfehler zu korrigieren, muss jedoch eine Schleifenschließung durchgeführt werden. Dies ist nicht immer möglich, vor allem bei autonomen Fahr-, Rettungs- und Überwachungseinsätzen, wenn eine vorgegebene Position schnell und direkt angefahren werden muss. Zudem werden für die sichere Lokalisierung und Navigation oft hochgenaue Karten benötigt. In unstrukturierter Umgebung mit viel Vegetation und abseits von Autobahn und Großstadt sind diese jedoch nur eingeschränkt oder gar nicht verfügbar.

In dieser Dissertation wird ein kooperatives SLAM-Verfahren vorgestellt, welches einem Fahrzeug die Lokalisierung an Orten ermöglicht, die noch nie zuvor aktiv mit Sensorik wahrgenommen wurden. Inspiriert von den kognitiven Fähigkeiten des Menschen erfolgt die globale Lokalisierung entlang einer Wegbeschreibung, der sogenannten Pfad-Karte. Die Pfad-Karte ist eine topologisch-metrische Beschreibung der Umgebung und kann automatisiert, beispielsweise auf Basis von OpenStreetMap Daten, generiert werden. Durch die Integration der Pfad-Karte im SLAM-Verfahren kann der akkumulierte Positionsfehler ausgeglichen und die lokale SLAM-Karte global referenziert werden, um die aktualisierte Karte anderen Fahrzeugen zur Verfügung zu stellen. Dafür werden die Ergebnisse einer Monte Carlo Lokalisierung mit neuartigem Messmodell zur globalen Lokalisierung, eines Multi-Target Trackingverfahrens zur Landmarkenbestimmung sowie einer B-Spline-basierten Straßenverlaufsschätzung in einem Graph-SLAM Ansatz probabilistisch fusioniert.

Für die Navigation in unstrukturierter Umgebung ist neben den Landmarken zur Lokalisierung auch ein Szenenverständnis der Umgebung nötig. Durch ein neuartiges LiDAR (Light Detection And Ranging)-sensorbasiertes Verfahren kann die Unterscheidung zwischen Hindernissen, Freiraum und vertikalen Objekten unmittelbar getroffen werden. Mithilfe rekursiver Schätzverfahren werden die Zustände aller Objekte in der Umgebung bestimmt. Auf Basis der Geschwindigkeitsschätzung und Zustandsunsicherheit werden anschließend statische Objekte erkannt und als Landmarken im SLAM-Verfahren und der Monte Carlo Lokalisierung berücksichtigt.

In mehreren Testfahrten über eine Gesamtstrecke von $56\,\mathrm{km}$ wurde das Gesamtsystem intensiv in verschiedenen Gebieten zu unterschiedlichen Jahreszeiten für den Anwendungsfall des autonomen und aufgesplitteten Konvois erprobt. Die Auswertungen belegen, dass topologisch-metrische Karten für die Navigation mit einem sensorbasierten Wahrnehmungssystem kombiniert werden können, um die Herausforderungen der autonomen Navigation in unstrukturierter Umgebung sowie bei schwierigen Witterungsbedingungen zu lösen. Es hat sich gezeigt, dass die Auswahl der abstrakten Landmarkenrepräsentation zur Lokalisierung eine robuste Kartenrepräsentation darstellt. Das Verfahren ist darüber hinaus universell einsetzbar und erlaubt die Erweiterung um beispielsweise visuelle Objektlandmarken.

Abstract

Conventional SLAM (Simultaneous Localization And Mapping) methods do not use prior information about the environment. They estimate the most likely map and position from a set of observations. In order to correct for the cumulative position error, loop closure must be performed. This is not always possible, especially in autonomous driving, rescue, and surveillance operations when a specific position needs to be approached quickly and directly. In addition, highly accurate maps are often required for reliable localization and navigation. However, in unstructured environments with a lot of vegetation and far away from highways and big cities, HD-maps are available only to a limited extent or not at all.

In this thesis, a cooperative SLAM method is presented that enables a vehicle to localize at locations that have never been actively observed with sensors before. Inspired by human cognitive abilities, global localization is performed along a path description called a path map. The path map is a topological metric description of the environment and can be generated automatically, for example, based on OpenStreetMap data. By integrating the path map into the SLAM process, the accumulated position error can be compensated and the local SLAM map can be globally referenced to provide the updated map to other vehicles. For this purpose, the results of Monte Carlo localization with novel measurement model for global localization, multi-target tracking method for landmark determination, and B-spline-based road course estimation are probabilistically fused in a graph SLAM approach.

For navigation in unstructured environments, scene understanding of the environment is required in addition to landmarks for localization. A novel LiDAR-based method can immediately make the distinction between obstacles, free space, and vertical objects. Using recursive estimation techniques, the states of all objects in the environment are determined. Based on the velocity estimation and state uncertainty, static objects are then detected and considered as landmarks in the SLAM method and Monte Carlo localization.

In numerous test drives covering a total distance of $56\,\mathrm{km}$, the system was extensively tested in various areas at different times of the year for the autonomous and split convoy use case. The evaluations demonstrate that topological metric maps for navigation can be combined with a sensor-based perception system to solve the challenges of autonomous navigation in unstructured environments as well as in difficult weather conditions. It has been shown that the selection of abstract landmarks for localization provides a robust map representation. Furthermore, the method is universally applicable and allows for extension to visual object landmarks.

Inhaltsverzeichnis

1 Einleitung

Inhalt

Roboter sind aus unserem Alltag nicht mehr wegzudenken. Neben der Verwendung in der Lieferlogistik und bei der Montage von Fahrzeugen reinigen sie z. B. in Form von kleinen Staubsaugerrobotern unsere Wohnung oder führen hochkomplexe chirurgische Eingriffe durch.

Über die Jahre hat sich die Robotik stetig weiterentwickelt. Bereits heute sind Roboter in Form von autonomen Fahrzeugen auf unseren Straßen unterwegs. In der nahen Zukunft werden sie die Art und Weise, wie wir uns fortbewegen, revolutionieren. Experten gehen davon aus, dass durch den vollständigen Ersatz von menschlichen Fahrern Verkehrsunfälle reduziert, der Fahrkomfort erhöht sowie die Umweltverschmutzung verringert werden kann [Winkle, 2015].

Nicht alle sich aktuell im Einsatz befindlichen Roboter arbeiten vollständig autonom, was u. a. an der fehlenden Robustheit liegt. Robustheit beschreibt die Fähigkeit eines Systems, mit allen auftretenden Fehlern und Unregelmäßigkeiten umgehen zu können, sodass die geforderten Tätigkeiten korrekt ausgeführt und zu keiner Zeit Menschen oder Tiere in Gefahr gebracht werden [Sünderhauf, 2012].

Der Grund für die fehlende Robustheit liegt vor allem im Entwicklungsprozess eines Roboters und der Komplexität der Aufgaben. Denn oftmals werden nicht alle Eventualitäten in der Entwurfsphase bedacht oder können schlicht nicht vorausgesehen werden. Folglich werden diese nicht modelliert und in Form von Software umgesetzt.

Die fünf Stufen des autonomen Fahrens Seit den 1980er Jahren arbeiten Ingenieure, Forscher und Techniker an der Automatisierung von Fahrzeugen und an den sogenannten Fahrassistenzsystemen. Dafür wurden die Entwicklungsschritte des autonomen Fahrens in fünf verschiedenen Stufen unterteilt [VDA, 2015]. Jede Stufe entspricht dabei einem Meilenstein, der fest definiert, in welchem Umfang das Fahrzeug die Aufgaben des Fahrers übernehmen kann und muss.

In der ersten Stufe, dem *Assistierten Fahren*, wird der Fahrer von Assistenzsystemen lediglich unterstützt. Diese Stufe ist in den meisten modernen Autos bereits zu finden, wobei keine direkten Steuereingriffe vorgenommen werden. Hierzu gehören

beispielsweise die aktive Geschwindigkeitsregelung mit Stop & Go Funktion, die den Abstand zum vorausfahrenden Fahrzeug sicher und selbständig regelt sowie einen automatischen Bremsvorgang einleitet, um eine Kollision zu verhindern. Die eingebauten Sensoren messen hierfür kontinuierlich die aktuelle Geschwindigkeit und den Abstand zum vorausfahrenden Fahrzeug. Die adaptive Abstands- und Geschwindigkeitsregelung (engl. Adaptive Cruise Control (ACC)) regelt als Assistenzfunktion automatisch das Beschleunigungs- und Bremsverhalten je nach Abstand zum Vorderfahrzeug.

In der zweiten Stufe, dem *Teilautomatisierten Fahren*, kann das Fahrzeug bereits automatisch beschleunigen, bremsen und im Vergleich zur ersten Stufe auch das Steuern teilautomatisiert übernehmen. Zur zweiten Stufe gehört somit ein Lenk- und Spurführungsassistent und ein Stauassistent. Diese ermöglichen bei Geschwindigkeiten von bis zu $60\,\mathrm{km/h}$, ohne die Hände des Fahrers am Lenkrad, die Spur zu halten. Bei diesen Systemen erfassen meist Kameras die Spurmarkierung und die Begrenzungslinien der eigenen Fahrspur. Das Fahrerassistenzsystem kann sich daran orientieren und der Spur teilautomatisiert folgen. Der Spurführungsassistent kann den Fahrer insbesondere im Stau entlasten und die Fahrt komfortabler und vor allem sicherer gestalten. Aktuell können diese Systeme jedoch nur dort eingesetzt werden, wo eine Fahrspur sicher anhand ihrer Markierung auf der Straße erkannt werden kann. In Bereichen ganz ohne Fahrspurmarkierungen oder bei schlecht erkennbaren und abgenutzten Markierungen, wie es oft in der Stadt, auf Landstraßen, aber auch auf Kreuzungen und Baustellen vorkommt, funktioniert das System noch nicht ausreichend.

In der dritten Stufe, dem *Hochautomatisierten Fahren*, ist es dem Fahrer erlaubt, unter bestimmten Voraussetzungen die Fahraufgabe vollständig an das Fahrzeug abzugeben, um sich dauerhaft vom Verkehrsgeschehen abwenden zu können. Der Fahrer darf sich jedoch nur insoweit von dem Fahrgeschehen abwenden, dass er innerhalb weniger Sekunden wieder eingreifen kann, beispielsweise um manuell durch eine Baustelle zu steuern. Das Fahrzeug muss somit über längere Strecken mit den unterschiedlichsten Verkehrssituationen selbständig zurechtkommen. Das hochautomatisierte Fahren im öffentlichen Straßenverkehr wird bereits seit einigen Jahren mit Forschungsfahrzeugen erprobt.

In der vierten Stufe, dem *Vollautomatisierten Fahren*, kann das Fahrzeug bereits zu einem überwiegenden Teil selbständig navigieren. Im Vergleich zur dritten Stufe, kann das Fahrzeug nun hochkomplexe urbane Verkehrssituationen meistern, z. B. eine plötzlich auftretende Baustelle. Der Fahrer muss jedoch trotzdem noch jederzeit bei Bedarf eingreifen können. Werden die Warnhinweise ignoriert oder ist der Fahrer sogar bewusstlos, so ist das Fahrzeug in dieser Stufe in der Lage, automatisch einen sicheren Zustand herzustellen, wie beispielsweise rechts heranzufahren und anzuhalten.

In der fünften Stufe, dem *Autonomen Fahren*, kommt das Fahrzeug vollständig ohne Fahrer aus. Im Vergleich zur dritten und vierten Stufe können somit Personen ohne Fahrerlaubnis oder Fahrtüchtigkeit transportiert werden. Das Fahrzeug ist in dieser Stufe vollständig autonom und kann folglich selbst Entscheidungen treffen.

Hochmoderne autonome Fahrsysteme sind aktuell noch in umfangreichem Maße auf detaillierte und sehr genaue Karten angewiesen. Außerhalb städtischer Gebiete ist es jedoch in der Regel schwierig und noch nicht rentabel, detaillierte Karten zu erstellen, zu speichern und zu übertragen. Darüber hinaus kann die Beibehaltung und Aktualisierung detaillierter Karten großer ländlicher Gebiete aufgrund der schnellen Geschwindigkeit, mit der sich das Erscheinungsbild dieser Umgebungen verändern kann, unpraktikabel sein.

Fehlende Trainings- und Validierungsdaten für unstrukturierte Gebiete grenzen die Verwendung neuronaler Netze oftmals ein, da bekannte Trainingsdatensätze wie beispielsweise *KITTI* von Geiger et al. [2013], *Cityscapes* von Cordts et al. [2016], *SemanticKITTI* von Behley et al. [2019], *A2d2* von Geyer et al. [2020] und *TUM-MLS-2016* von Zhu et al. [2020] in strukturierter Umgebung erstellt worden sind.

1.1 Grundfunktionalitäten eines autonomen Fahrzeugs

Damit sich ein Fahrzeug autonom in der Welt bewegen kann, sind einige grundlegende und entscheidende Fähigkeiten notwendig. Die Umgebungsrepräsentation ist dabei die Datengrundlage aller Fähigkeiten, die ein autonomes Fahrzeug besitzen muss. Sie wird mittels Sensormessungen der aktuell wahrgenommenen Umgebung erstellt.

So erlaubt eine Umgebungsrepräsentation idealerweise die Unterscheidung zwischen befahrbaren und nicht befahrbaren Oberflächen, sowie statischen und dynamischen Hindernissen. Denn dies sind die notwendigen und entscheidenden Informationen, um sicher navigieren zu können.

Auf Basis der Umgebungsrepräsentation lässt sich beispielsweise eine metrische Karte der Umgebung ableiten, welche die Planung für eine sichere Navigation zu einem gegebenen Ziel ermöglicht. Dies ist jedoch nur möglich, sofern die eigene Position innerhalb der Karte bekannt ist.

Die Aufgaben der Kartierung, der Lokalisierung und der Planung stellen somit die Grundfunktionalitäten eines autonomen Fahrzeugs dar.

In Abbildung 1.1 ist die gegenseitige Abhängigkeit der drei Komponenten Kartierung, Lokalisierung und Planung dargestellt.

Kartierung (engl. Mapping) Eine Herausforderung bei der Kartierung ist die Erfassung und Korrelation einer Vielzahl von Sensormessungen in einer gemeinsamen und konsistenten Kartendarstellung. Weit verbreitete Kartenrepräsentationen sind beispielsweise merkmalbasierte, geometrische und topologische Karten. Die Kartenrepräsentationen unterscheiden sich zwar in der Darstellung, jedoch müssen alle Methoden mit Sensorrauschen und Unsicherheiten umgehen können. Zusätzlich muss die eigene Positionsunsicherheit bei der Korrelation von Sensormessungen berücksichtigt werden. Kartierungsverfahren und verschiedene Kartentypen werden in Unterabschnitt 3.1.1 beschrieben.

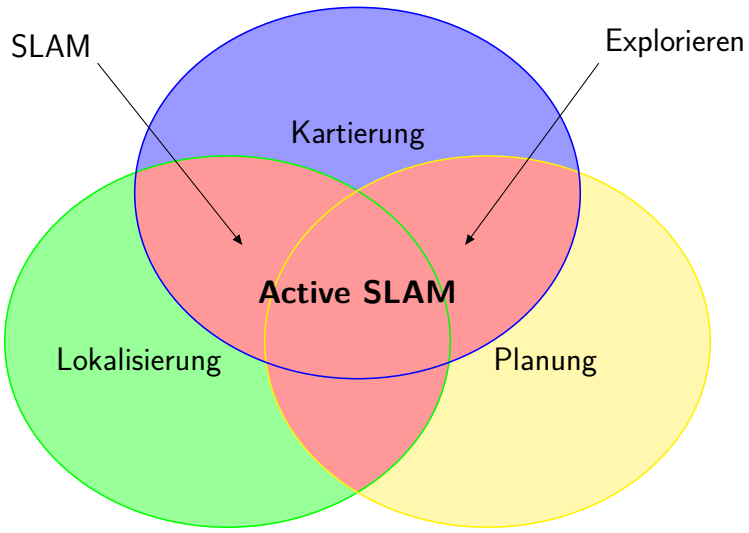


Abbildung 1.1:
Darstellung von Aufgaben, die von einem autonomen Fahrzeug gelöst werden müssen. Die sich überschneidenden Bereiche stellen Kombinationen aus Kartierungs-, Lokalisierungs- und Planungsaufgaben dar [Makarenko et al., 2002].

Lokalisierung (engl. Localisation) Die Lokalisierung erfolgt anhand von Sensormessungen, die eine Abschätzung der Position innerhalb einer Karte ermöglichen. Dabei muss das Sensorrauschen sowie die Unsicherheit der Karte berücksichtigt werden. In einigen Fällen kann die Lokalisierung möglicherweise nicht zwischen zwei oder mehr plausiblen Positionen und Orientierungen innerhalb der Karte unterscheiden. Dieser Fall muss erkannt und explizit behandelt werden und tritt insbesondere dann auf, wenn keine exakte Initialisierung innerhalb der Karte möglich ist. Ein Einblick in aktuelle Lokalisierungsmethoden wird in Unterabschnitt 3.1.2 gegeben.

Planung Die Planung entscheidet, welche Aktion als nächstes ausgeführt werden soll. Der Aktionsraum ist durch Randbedingungen limitiert. Hierzu gehören beispielsweise physische Einschränkungen (maximale und minimale Beschleunigung) sowie sicherheitsrelevante Aspekte (Kollisionsvermeidung). Zusätzlich muss die Planung auch mit Unsicherheiten in der Karte, der aktuellen Lokalisierung sowie dem Ergebnis von zuletzt ausgeführten Aktionen umgehen können. Eine wichtige Unterfunktion ist die Trajektorienplanung mit der Aufgabe, eine sichere und zweckmäßige Route zu einer bestimmten Zielposition zu finden. Darüber hinaus besitzt ein autonomes Fahrzeug eine sogenannte übergreifende Aufgabe, die weitgehend bestimmt, was als Nächstes passieren bzw. erreicht werden soll. Das kann beispielsweise ein übergeordnetes Navigationsziel sein, wie „Folge der Straße bis zur nächsten Tankstelle“.

Jedes genannte Element kann als unabhängiges Problem betrachtet werden, jedoch erlaubt erst die Kombination aller Elemente einem Fahrzeug sich autonom, effizient und vor allem sicher bewegen zu können.

Der Prozess, der aktiv Aktionen wählt, um Unsicherheiten bei der Kartierung als auch bei der Lokalisierung zu reduzieren, wird als active SLAM bezeichnet. Darüber hinaus umfasst active SLAM auch das Explorieren. Beim Explorieren werden neue Teile der Karte erkundet und die Karte erweitert. Die Kartierung und Lokalisierung sind dabei unmittelbar von der Planung abhängig, da die Planung bewusst Aktionen ausführen lassen kann, um die Unsicherheiten in der Karte sowie der Positionsschätzung zu reduzieren.

Simultane Lokalisierung und Kartierung (SLAM) SLAM gehört zu den Hauptforschungsgebieten im Bereich der mobilen Robotik und wird als die Aufgabe beschrieben, eine Karte zu erstellen (Kartierung) und dabei die Position des Fahrzeugs relativ zu dieser Karte abzuschätzen (Lokalisierung).

Das Fahrzeug ist dabei mit Sensoren, wie beispielsweise Light Detection And Ranging (LiDAR)-Sensoren, Radio Detection And Ranging (Radar)-Sensoren und Kameras, ausgestattet, die zum einen die Bewegungen mittels Odometrie schätzen und zum anderen die Umgebung wahrnehmen.

Um die gesammelten Informationen über die Umgebung in einer konsistenten Karte abbilden zu können, muss der Roboter seine Position und Orientierung relativ zur gleichen Karte kennen. Zu Beginn ist die Karte leer, da diese erst über die Zeit mit Informationen angereichert wird.

Dies ist eine komplexe Aufgabe, da eine Karte zur Lokalisierung des Fahrzeugs benötigt wird und gleichzeitig die Position und Orientierung des Fahrzeugs für die Erstellung der Karte essenziell ist [Alismail et al., 2014, Bardow et al., 2016, Blochliger et al., 2018, Brubaker et al., 2016, Carlone et al., 2014a, 2016, Engel et al., 2014, Forster et al., 2017a, 2014, 2017b, Gao et al., 2018, Guivant und Nebot, 2001, Holz und Behnke, 2010, Kerl et al., 2013, Kim et al., 2016b, Kümmerle et al., 2010, Li et al., 2012, Lu und Milios, 1997a, Makarenko et al., 2002, Milford und Wyeth, 2012, Montemerlo und Thrun, 2003, Montemerlo et al., 2002, 2003, Moosmann und Stiller, 2011, Mur-Artal et al., 2015, Pomerleau et al., 2013, Thrun et al., 2005, Valencia und Andrade-Cetto, 2018, Zhou et al., 2017].

Folglich ist eine genaue Lokalisierung für den Prozess von entscheidender Bedeutung, da eine ungenaue Lokalisierung auch eine ungenaue Karte zur Folge hat und sich das wiederum stark auf das zukünftige Lokalisierungsergebnis auswirkt.

Des Weiteren gilt SLAM als eine große Herausforderung, da aufgrund von Messrauschen und Systemfehlern keine der Messungen perfekt ist und somit nicht davon ausgegangen werden kann, dass die gemessene Bewegung oder wahrgenommene Umgebung der Realität entspricht. Zusätzlich ist im Allgemeinen die Zuordnung von sensorbasierten Beobachtungen und Kartenmerkmalen als unbekannt anzunehmen. Um die genannten Unsicherheiten zu reduzieren und Mehrdeutigkeiten aufzulösen, wird bei SLAM in der Regel mit probabilistischen Methoden und Filter-Techniken gearbeitet. Die Wahl falscher Datenassoziationen kann dabei zu großen Positionsfehlern bis hin zum Verlust des Fahrzeugs führen.

Abbildung 1.2: Autonome Konvoi-Fahrzeuge in unstrukturierter Umgebung.

1.2 Forschungsziel

1.2.1 Beschreibung

Das Forschungsziel dieser Arbeit ist die Entwicklung und Erprobung eines Gesamtkonzepts für die kooperative und simultane Lokalisierung und Kartierung der Umgebung von autonomen Konvoi-Fahrzeugen mit dem Fokus auf unstrukturierte Umgebungen. In Abbildung 1.2 sind die beiden Forschungsfahrzeuge MuCAR-3 und MuCAR-4 (Munich Cognitive Autonomous Robot) im autonomen Konvoi zu sehen, welche für die Evaluierung und Erprobung verwendet werden.

In dieser Arbeit wird untersucht und erprobt, inwieweit topologisch metrische Karten für die globale Navigation mit einem sensorbasierten Wahrnehmungssystem kombiniert werden können, um die Herausforderungen der autonomen Navigation in unstrukturierter Umgebung zu lösen, denn im Allgemeinen verwenden SLAM-Verfahren keine Vorabinformationen über die Umwelt. Sie schätzen die wahrscheinlichste Karte und Position angesichts einer Reihe von Beobachtungen. Um den akkumulierten Positionsfehler zu korrigieren, müssen SLAM-Verfahren über die Zeit das sogenannte Loop Closing (Schleifenschließung) durchführen. Beim Loop Closing erkennt das SLAM-System eine bereits erfasste Umgebung und der akkumulierte Positionsfehler kann korrigiert bzw. verringert werden.

Eine Schleifenschließung ist jedoch nicht immer möglich, wenn beispielsweise für autonome Fahr-, Rettungs- und Überwachungsszenarien eine vorgegebene, im globalem Raum liegende Position schnell erreicht werden muss, ohne dass eine Position über größere Distanzen zweimal besucht werden kann.

Um ein weit entferntes Ziel zügig zu erreichen, muss eine interpretierbare Karte oder eine Wegbeschreibung, die dem autonomen System die benötigten Navigationsanweisungen bereitstellt, bereits vorhanden sein. Darüber hinaus hängt die Verwendbarkeit sowie die Genauigkeit der Lokalisierung unmittelbar von der Auflösung und dem

Detaillierungsgrad der verwendeten Karte ab [Levinson et al., 2008, Schiotka et al., 2017].

Ein Hauptziel dieser Arbeit ist daher die globale Lokalisierung innerhalb einer gegebenen spärlichen Landmarkenkarte ohne die Berücksichtigung von Global Navigation Satellite System (GNSS) oder hochgenauem Kartenmaterial. Eine Randbedingung ist hierbei, dass die verwendeten und selbst erstellten Karten nur wenig Speicher benötigen dürfen, da sie mittels schmalbandigem Funk zwischen Fahrzeugen eines autonomen Konvois ausgetauscht werden sollen.

Zusätzlich soll das System die Integration und Verwendung von weiterem Kartenmaterial ermöglichen, um auch in Gebieten zu operieren, die noch nicht befahren und kartiert wurden. Hierzu gehören beispielsweise frei zugängliches Kartenmaterial wie das von OpenStreetMap (OSM) oder Kartenmaterial, das aus Luftbildern einer Drohne erstellt wird. Die Genauigkeit und Zuverlässigkeit dieser Karten kann jedoch oftmals stark schwanken, was beim Lokalisierungsverfahren zu berücksichtigen ist.

Ein weiteres Forschungsziel dieser Arbeit ist die Entwicklung eines Umgebungsmodells auf Basis von Sensormessungen, das eine sichere Lokalisierung und Navigation ermöglicht. Hierzu muss zuerst eine geeignete Sensorik ausgewählt und auf Basis der Eingangsdaten eine Reihe von Verfahren entwickelt werden, welche statische und dynamische Objekte sowie Flächen in der Umgebung erkennen. Zusätzlich müssen aus der Menge an erkannten Objekten Landmarken identifiziert werden.

Das übergeordnete Ziel dieser Arbeit ist die Erhöhung der Robustheit von autonomen Fahrzeugen speziell in unstrukturierten Umgebungen und abseits von Autobahnen und Städten, damit autonome Fahrzeuge in Zukunft universell und außerhalb von kontrollierten Umgebungen und genau kartierten Bereichen eingesetzt werden können.

1.2.2 Randbedingungen

Die Randbedingungen und die sich daraus ergebenden Systembedingungen können zusammengefasst und wie folgt definiert werden:

- Auf die Verwendung von hochgenauem GNSS und hochgenauem Kartenmaterial zur Lokalisierung muss verzichtet werden.
- Der benötigte Speicherplatz der Karte muss klein sein, damit diese über schmalbandigen Funk übertragen werden kann.
- Die Karte muss vom Menschen gelesen und erweitert werden können.
- Das Verfahren muss für alle Fahrzeuge eines aufgesplitteten Konvois verwendet werden können.
- Eine hinreichende Erprobung und Evaluierung mit verschiedenen Versuchsträgern in unterschiedlichem Terrain muss durchgeführt werden.

1.3 Beiträge der Dissertation

Die Beiträge dieser Arbeit liegen in den Bereichen simultane Lokalisierung und Kartierung sowie Umgebungswahrnehmungsmethoden für autonome Fahrzeuge. Im Folgenden sind die wesentlichen Beiträge dieser Arbeit zusammengefasst.

Im Themengebiet der Umgebungswahrnehmung:

- Methode zur Punktwolkensegmentierung, die das Modellwissen des charakteristischen Diodenmusters eines LiDAR-Sensors dazu benutzt, Freiräume von Hindernissen unterscheiden zu können.
- Graphen-basierte Clusterverfahren zur Objekterkennung als Erweiterung der Punktwolkensegmentierung, um auch bei weiten Entfernungen, schwierigen Betrachtungswinkeln oder sogar bei Verdeckungen Objektinstanzen sicher zu detektieren.
- Methode zur Erkennung von Straßensegmenten in einer Punktwolke und deren zeitliche Schätzung mit einem Unscented Kalman Filter (UKF) und einem B-Spline basierten Prädiktions- und Messmodell.

Im Themengebiet von SLAM:

- Eine Lokalisierungsmethode für autonome Fahrzeuge unter Verwendung einer Monte Carlo Lokalisierung (MCL), welche die gesamte Verarbeitungskette von der Landmarkenextraktion bis zur Verarbeitung der Fahrzeugposition umfasst.
- Untersuchung zur Verwendung alternativer Karten für autonome Fahrzeuge.
- Eine Kartierungsmethode zur Erstellung einer globalen und konsistenten Karte sowie zur Glättung der Fahrzeugtrajektorie mit einem graphbasierten SLAM-Verfahren. Als Datengrundlage dient neben den identifizierten Landmarken und der globalen Lokalisierungslösung der Monte Carlo Methode auch die Berücksichtigung des mit einem Kalman Filter (KF) geschätzten Straßenverlaufs.

Einige Themengebiete dieser Arbeit wurden mit Genehmigung der Fakultät bereits vorab auf internationalen Konferenzen veröffentlicht. Eine chronologische Auflistung ist im Folgenden zu finden:

- Fries, C., Burger, P., Kallwies, J., Naujoks, B., Luettel, T., und Wuensche, H. J. (2018). How MuCAR won the convoy scenario at ELROB 2016. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Seiten 1--7, Fries et al. [2018]
- Burger, P. und Wuensche, H. J. (2018). Fast Multi-Pass 3D Point Segmentation Based on a Structured Mesh Graph for Ground Vehicles. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 2150--2156, Burger und Wuensche [2018]

- Huang, H., Burger, P., Schmitz, M., Roth, L., Wünsche, H. J., und Mayer, H. (2018). Driving in unknown areas: From UAV images to map for autonomous vehicles. In *IWCTS 2018 - Proceedings of the 11th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, Seiten 39--42, Huang et al. [2018]
- Burger, P., Naujoks, B., und Wuensche, H. J. (2018). Fast Dual Decomposition based Mesh-Graph Clustering for Point Clouds. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Seiten 1129--1135, Burger et al. [2018]
- Mueller, G. R., Burger, P., und Wuensche, H. J. (2018). Continuous Stereo Self-Calibration on Planar Roads. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 1755--1760, Mueller et al. [2018]
- Naujoks, B., Burger, P., und Wuensche, H. J. (2019c). The greedy dirichlet process filter-An online clustering multi-target tracker. In *2018 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2018 - Proceedings*, Seiten 1233--1237, Naujoks et al. [2019c]
- Naujoks, B., Burger, P., und Wuensche, H.-J. (2019a). Combining Deep Learning and Model-Based Methods for Robust Real-Time Semantic Landmark Detection. In *22nd International Conference on Information Fusion (FUSION)*, Ottawa, ON, Canada, Naujoks et al. [2019a]
- Naujoks, B., Burger, P., und Wuensche, H.-J. (2019b). Fast 3D Extended Target Tracking using NURBS Surfaces. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, Auckland, New Zealand. ©IEEE, Naujoks et al. [2019b]
- Burger, P., Naujoks, B., und Wuensche, H. J. (2019b). Unstructured Road SLAM using Map Predictive Road Tracking. *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, Seiten 1276--1282, Burger et al. [2019b]
- Burger, P., Naujoks, B., und Wuensche, H. J. (2019a). Map-Aware SLAM with Sparse Map Features. *IEEE International Conference on Intelligent Robots and Systems*, Seiten 347--353, Burger et al. [2019a]

1.4 Gliederung

Die Gliederung dieser Arbeit orientiert sich an den verschiedenen Komponenten, die für das Lösen der Forschungsfrage nötig sind.

Begonnen wird in Kapitel 2 mit einem Grundlagenkapitel, in dem die wichtigsten Methoden und Grundlagen dieser Arbeit beschrieben werden.

Kapitel 3 gibt einen Einblick in den aktuellen Stand der Technik zu SLAM-Methoden, wobei detaillierter auf Verfahren zur Lokalisierung, Kartierung und der simultanen Ausführung eingegangen wird.

In Kapitel 4 wird das objektbasierte Umgebungsmodell zur Erfassung aller statischen und dynamischen Objekte vorgestellt. Die Erkennung von Freiräumen, Hindernissen und Straßenelementen bis hin zur Landmarkenextraktion sind einige der erforderlichen Komponenten dieser Arbeit, um in unstrukturierter Umgebung navigieren zu können.

Der Kern dieser Arbeit, das entwickelte SLAM-Framework, wird in Kapitel 5 vorgestellt. Einzelne Komponenten des Frontends und Backends des SLAM-Frameworks werden auf Basis der Top-Down Systemarchitektur beschrieben.

In Kapitel 6 wird die Struktur der Karte sowie das Geographic Information System (GIS) eingeführt. Neben den Datenbankmodellen wird hier auch auf die Kommunikation der Daten über Funk zwischen den Konvoi-Teilnehmern eingegangen.

In Kapitel 7 werden die mit dem Gesamtsystem in verschiedenen Einsatzgebieten ausgeführten Experimente beschrieben sowie die qualitativen Ergebnisse ausgewertet und vorgestellt.

Im abschließenden Kapitel 8 werden die erreichten Ergebnisse zusammengefasst und auf Basis der Forschungsfragen und Randbedingungen diskutiert.

2 Grundlagen

Inhalt

Im Folgenden werden grundlegende Notationen und Methoden beschrieben, die für das Verständnis dieser Arbeit und der darin abgehandelten Themen erforderlich sind. Zunächst werden einige probabilistische Theorien und die Bayes-Regeln beschrieben. Danach wird die Bayessche Wahrscheinlichkeitstheorie als Grundlage zur Erklärung der parametrischen und nichtparametrischen Bayesschen Filteralgorithmen verwendet. Darüber hinaus sind einige formale Herleitungen enthalten, um das Verständnis der vorgestellten Konzepte zu verbessern. Techniken zur Optimierung der kleinsten Quadrate für die Graphen-Optimierung bilden das Ende des Kapitels.

2.1 Notation und Abkürzungen

In dieser Arbeit werden Zustandsvektoren durch fett gedruckte Kleinbuchstaben und Zustandsräume durch fett gedruckte Großbuchstaben bezeichnet (z. B. ist $\boldsymbol{x}$ ein Zustandsvektor des Zustandsraums $\boldsymbol{X}$).

Mengen und Matrizen werden durch Großbuchstaben beschrieben. Die Menge an n Zuständen $\boldsymbol{x}^{(i)}$ wird dabei wie folgt geschrieben:

$$\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\}_{i=1}^{n}. \tag{2.1}$$

Für eine zeitliche Beschreibung wird der Index k verwendet. So bezeichnet beispielsweise $\boldsymbol{x}_k$ den Zustand zum Zeitpunkt k und $\boldsymbol{X}_K$ die Menge aller Zustände bis zum Zeitschritt k. Eine Menge von Zuständen lässt sich auch als Matrix beschreiben, wobei die Einträge jeder Zeile k einen Zustand repräsentieren.

2.1.1 Zustände und Beobachtungen

In der Robotik wird die Umgebungswahrnehmung in der Zustandsraumdarstellung dargestellt. Ein Zustandsraum beinhaltet eine Menge von Zuständen. Der Zustandsvektor $\boldsymbol{x}_k$ repräsentiert dabei den tatsächlichen Zustand zum Zeitschritt k von beispielsweise dem Ego-Fahrzeug, von Objekten in der Umgebung oder der Umgebung selbst. Zustandsgrößen ändern sich mit der Zeit, wie z. B. die Position und Geschwindigkeit des Ego-Fahrzeugs, was auch für andere dynamische Objekte in der Umgebung gilt.

Eine Beobachtung $\boldsymbol{y}$ hingegen ist eine mittels Sensor wahrgenommene Zustandsänderung oder der Zustand selbst. Beispielsweise liefern LiDAR-Sensoren im Allgemeinen Abstands- und Intensitätsinformationen von Objekten der Umgebung. Somit sind die Beobachtungen des LiDAR-Sensors als ein Reihe von n-Entfernungsmessungen zum Zeitschritt k wie folgt definiert:

$$\boldsymbol{Y}_k = \{r_k^{(i)}, \ldots, r_k^{(n)}\}_{i=1}^{n}. \tag{2.2}$$

Eine weitere wichtige Eingangsgröße ist der sogenannte Steuer- oder auch Aktionsvektor $\boldsymbol{u}_{k-1}$. Dieser enthält Informationen über eine Aktion oder Steuergröße, die zum Zeitschritt $k-1$ erzeugt wurde und sich zum Zeitschritt k auswirkt.

Da in dieser Arbeit kein *Active SLAM* betrieben wird und somit keine aktiven Steuerbefehle gegeben werden, wird $\boldsymbol{u}_{k-1}$ als Aktion interpretiert. Die Aktion setzt sich aus einer gemessenen Geschwindigkeit und Gierrate des Fahrzeugs zusammen, welche durch Trägheitssensoren zum Zeitschritt k gemessen werden.

2.1.2 Posen

In dieser Arbeit wird das Ego-Fahrzeug, dynamische Objekte und Landmarken als Posen (Koordinate mit Orientierung) und Positionen (Koordinate ohne Orientierung) im dreidimensionalen $\mathbb{R}^3$ sowie im zweidimensionalen $\mathbb{R}^2$ Raum modelliert.

Das gewählte Fahrzeugkoordinatensystem ist dabei ein dreidimensionales kartesisches Koordinatensystem, das fest mit dem Fahrzeug verbunden ist. Es wird ein rechtshändiges Koordinatensystem gewählt, wobei die x-Achse nach vorn und die z-Achse nach oben weist. Die Orientierung wird durch die drei Euler-Winkel Yaw (Gierwinkel), Pitch (Nickwinkel), Roll (Wankwinkel) definiert, siehe Abbildung 2.1.

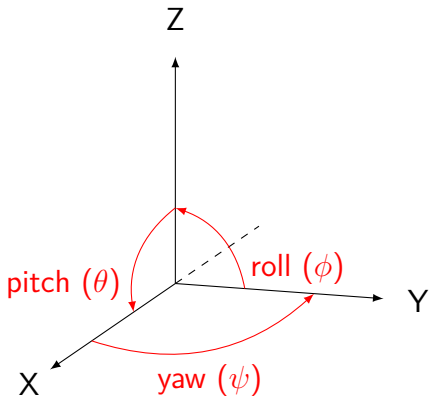


Abbildung 2.1:
Konvention der eulerschen Winkel. Pfeile geben die positive Richtung vor.

Eine beliebige starre Transformation kann in zwei Teile zerlegt werden: in eine starre Rotation und eine Translation. Folglich ist eine Pose in 2D definiert als $\boldsymbol{p}^3 = \left(p^x, p^y, p^\psi\right)^T$ und stellt in verkürzter Schreibweise die Parameter einer 3×3 großen homogenen Matrix $\mathbf{H}$ dar:

$$\mathbf{H} = \left(\begin{array}{c|c} \mathbf{R}_{2\times 2} & \boldsymbol{t} \\ \hline \mathbf{0}_{1\times 2} & 1 \end{array}\right) = \left(\begin{array}{cc|c} \cos\psi & -\sin\psi & t_x \\ \sin\psi & \cos\psi & t_y \\ \hline 0 & 0 & 1 \end{array}\right), \tag{2.3}$$

wobei die Translation t^x, t^y und die Rotation ψ die drei Freiheitsgrade bestimmen. Somit ist eine Pose in 2D, $\mathbf{SE}(2) = \mathbb{R}^2 \times \mathbf{SO}(2)$ mit Hilfe der *Special Euclidean Group*

deutlich einfacher zu beschreiben. Alle Rotationsmatrizen $\mathbf{R}_{2\times 2}$ deren Determinante eins ist, gehören per Definition zur *Special Orthogonal Group* $\mathbf{SO}(2)$.

Die 3D Pose $\mathbf{SE}(3) = \mathbb{R}^3 \times \mathbf{SO}(3)$ hingegen besitzt sechs Freiheitsgrade und ist wie folgt definiert:

$$\boldsymbol{p}^6 = \left(p^x, p^y, p^z, p^\phi, p^\theta, p^\psi\right)^T, \tag{2.4}$$

wobei die Euler-Winkel p^ϕ, p^θ, p^ψ die Drehung um die X-,Y- und Z-Achse beschreiben.

Im Folgenden werden, falls nicht anders gekennzeichnet, die Position von Fahrzeugen in SE(2) und Landmarken als 2D Punkte modelliert und die Degree of Freedom (DOF)-Notation weggelassen.

2.1.3 Komposition von Posen und Referenzsysteme

Um eine konsistente und präzise Darstellung der Umgebung zu erhalten, müssen die genauen Einbaulagen aller Sensoren bekannt sein. Eine verbreitete Methode ist dabei die Repräsentation aller Onboard-Sensorik in Bezug auf ein definiertes lokales Bezugssystem, den Fahrzeugreferenzpunkt. Werden jedoch beispielsweise Kameras in Bezug zu einem LiDAR-Sensor kalibriert, wird die Pose der Kamera unter Berücksichtigung der extrinsischen Einbaulage des LiDAR-Sensors (Referenzsensors) $\boldsymbol{p}^{\mathtt{lidar}'}$ beschrieben, dessen Pose zum Fahrzeugreferenzpunkt beispielsweise durch eine technische Zeichnung bekannt ist. Damit die Kameras nun im Fahrzeugreferenzpunkt beschrieben werden können, müssen mehrere Transformationen verkettet bzw. Posen zusammengesetzt werden.

Um die Position eines Fahrzeugs zu beschreiben, wird der Fahrzeugreferenzpunkt in einem Referenzsystem $\boldsymbol{p}_{ref}$ definiert. Ein gängiges Referenzsystem ist das Universal Transverse Mercator (UTM)-Koordinatensystem, bei dem die Welt in 60 Nord-Süd-Zonen eingeteilt ist, die jeweils sechs Längengrade breit sind. Innerhalb jeder Zone werden die Koordinaten als Nord und Ost in Metern gemessen.

Beim geografischen Koordinatensystem hingegen wird die Position eines Ortes auf der Erdoberfläche mithilfe von sphärischen Maßen der geografischen Länge und Breite beschrieben, unter der Annahme, dass die Erde eine Kugel ist. Dabei entspricht die auf einem Längengrad zurückgelegte Entfernung in Richtung der Pole nur am Äquator der auf einem Breitengrad zurückgelegten Entfernung. Da das UTM-System ein konstantes Entfernungsverhältnis an jeder Stelle der Karte bietet, ist dessen Verwendung deutlich praktikabler.

Beim landmarkenbasierten SLAM erfolgt die Kartierung von Landmarken ebenfalls in einem Referenzsystem. Messungen von Objekten werden im Sensorkoordinatensystem erfasst. Die Pose des LiDAR-Sensor $\boldsymbol{p}^{\mathtt{lidar}'}$ wird anhand der aktuellen Fahrzeugposition in den globalen UTM-Kartenraum transformiert:

$$\boldsymbol{p}^{\mathtt{lidar}} = \boldsymbol{p}^{\mathtt{utm}} \oplus \boldsymbol{p}^{\mathtt{lidar}'}, \tag{2.5}$$

wobei die Pose des Fahrzeugs $\boldsymbol{p}^{\mathtt{utm}}$ in UTM-Koordinaten mit der extrinsischen Pose des LiDAR-Sensors $\boldsymbol{p}^{\mathtt{lidar}'}$ zusammengesetzt wird, um die globale LiDAR-Sensor-Pose $\boldsymbol{p}^{\mathtt{lidar}}$ zu erhalten.

Die inverse Zusammensetzung ist dabei wie folgt definiert:

$$\boldsymbol{p}^{\mathtt{lidar}'} = \boldsymbol{p}^{\mathtt{lidar}} \ominus \boldsymbol{p}^{\mathtt{utm}}. \tag{2.6}$$

Die Landmarkenposition von $\boldsymbol{l} = \left(x, y, z\right)^T$ in UTM-Koordinaten lässt sich wie folgt beschreiben:

$$\boldsymbol{l} \equiv \boldsymbol{p}^{\mathtt{lidar}} \oplus \boldsymbol{l}', \tag{2.7}$$

mit $\boldsymbol{l}'$ als Positionsmessung der Landmarke im Sensorkoordinatensystem. Das dazugehörige Beispiel ist in Abbildung 2.2 visualisiert.

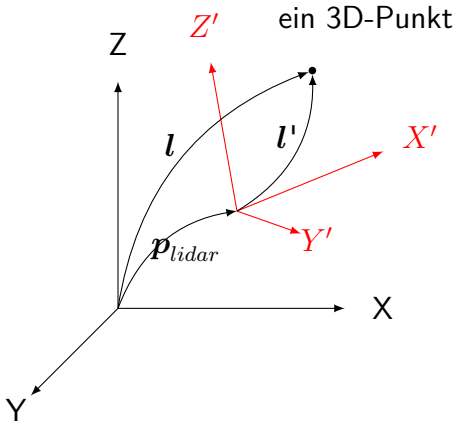


Abbildung 2.2:
Exemplarische Darstellung eines 3D-Punktes $\boldsymbol{l}$ im UTM-Referenzsystem des LiDAR-Sensors $\boldsymbol{p}_{\text{lidar}}$.

2.2 Wahrscheinlichkeitsverteilung über Zufallsvariablen

In der Robotik werden Zustände, Messungen und Aktionen als Zufallsvariablen modelliert, wobei diese wiederum selbst aus Beobachtungen bzw. von weiteren Zufallsvariablen abgeleitet werden können.

Im Folgenden ist X eine Zufallsvariable, die n verschiedene Werte annehmen kann. Die Wahrscheinlichkeitsverteilung $p\left(x\right)$ der Zufallsvariablen X drückt eine relative Wahrscheinlichkeit für ein bestimmtes Ergebnis aus $p\left(X = x\right) = p\left(x\right)$. Der Vektor dieser Zufallsvariablen $X_1 \ldots X_n$ wird als Multivariate bezeichnet.

2.2.1 Gemeinsame Wahrscheinlichkeit und absolute Unabhängigkeit

Weitere Begriffe in der Wahrscheinlichkeitstheorie sind die Verbundwahrscheinlichkeit (engl. joint probability) und die absolute Unabhängigkeit (engl. absolute independence) von Zufallsvariablen.

Die Verbundwahrscheinlichkeit $p(X = x \wedge Y = y) = p(x, y)$ beschreibt das Ereignis, dass die Zufallsvariable X den Wert x und Y den Wert y annimmt. Wenn die Zufallsvariablen X, Y absolut unabhängig sind, kann die gemeinsame Wahrscheinlichkeit wie folgt beschrieben werden: $p(x, y) = p(x)\, p(y)$.

2.2.2 Bedingte Wahrscheinlichkeit

Aristoteles war der erste Mensch, der die Kausalitätstheorie als eine Möglichkeit zum Verständnis der menschlichen Erfahrung der physischen Natur einführte [von Fritz und Weiss, 1944]. Kausalität kann als die Beziehung zwischen Ursache und Wirkung beschrieben werden. Sie umreißt die Abfolge von verwandten Ereignissen und Zuständen.

Ein Ereignis oder der Zustand *A* ist die Ursache für die Wirkung *B*, wenn *B* durch *A* bewirkt wird. Diese Methodik spielt eine grundlegende Rolle in der probabilistischen Robotik. Beispielsweise enthalten Sensormessungen Informationen über den Zustand des Roboters. Der Zustand des Roboters wird aus den Messungen abgeleitet, sodass z. B. die Positionsschätzung des Roboters von den Messungen abhängig ist und somit auch die Sensormessungen von der aktuellen Position im Raum.

Für den allgemeinen Fall der bedingten Wahrscheinlichkeit gilt:

$$p(x \mid y) = p(X = x \mid Y = y) = \frac{p(x, y)}{p(y)}, \tag{2.8}$$

wobei $p(y) \neq 0$. Wenn die Zufallsvariablen unabhängig sind, dann ist die bedingte Wahrscheinlichkeit wie folgt definiert:

$$p(x \mid y) = \frac{p(x, y)}{p(y)} = \frac{p(x)\, \cancel{p(y)}}{\cancel{p(y)}} = p(x)\,. \tag{2.9}$$

Gesetz der totalen Wahrscheinlichkeit Das Gesetz der totalen Wahrscheinlichkeit ergibt sich aus dem Grundsatz der Wahrscheinlichkeitstheorie und dem Gesetz der bedingten Wahrscheinlichkeit für unabhängige Zufallsvariablen, siehe Gleichung (2.9), und ist wie folgt definiert:

$$p(x) = \sum_{Y} p(x \mid y)\, p(y) \tag{2.10}$$

$$p(x) = \int_{Y} p(x \mid y)\, p(y)\, dy, \tag{2.11}$$

für diskrete bzw. kontinuierliche Zufallsvariablen. Darüber hinaus gilt, dass das Produkt $p(x \mid y)\, p(y)$ null ist, wenn eines der Elemente Null ist.

Kettenregel Die Kettenregel wird von der grundlegenden bedingten Wahrscheinlichkeitsregel abgeleitet und ermöglicht, die gemeinsame Verteilung einer Reihe von Zufallsvariablen $X_1, \ldots, X_n$ nur mit bedingten Wahrscheinlichkeiten zu berechnen:

$$p(x_1, \ldots, x_n) = p(x_n \mid x_{n-1}, \ldots, x_1)\, p(x_{n-1}, \ldots, x_1)\,. \tag{2.12}$$

Satz von Bayes Die Bayes-Regel ist die wichtigste Regel in der probabilistischen Robotik. Sie wird verwendet, um die a-posteriori Wahrscheinlichkeitsverteilung $p(x \mid y)$, die als die Wahrscheinlichkeitsverteilung über x bei gegebener Wahrscheinlichkeitsverteilung über y beschrieben wird, aus ihrer invers bedingten Wahrscheinlichkeit $p(y \mid x)$ zu bestimmen. Die invers bedingte Wahrscheinlichkeit ist auch als das generative Modell $p(y \mid x)$ bekannt, das die Beobachtung y zum aktuellen Zustand x beschreibt. Die Bayes-Regel erfordert $p(y) > 0$ und ist für diskrete und kontinuierliche Zufallsvariablen wie folgt definiert:

$$p(x \mid y) = \frac{p(y \mid x)\, p(x)}{p(y)} = \frac{p(y \mid x)\, p(x)}{\sum_{X'} p(y \mid x')\, p(x')} \tag{2.13}$$

$$p(x \mid y) = \frac{p(y \mid x)\, p(x)}{p(y)} = \frac{p(y \mid x)\, p(x)}{\int_{X'} p(y \mid x')\, p(x')\, dx}. \tag{2.14}$$

In der Bayesschen Statistik wird die Verteilung $p(x)$ als Prior bezeichnet, da sie Informationen über die Unsicherheit eines bestimmten Zustandes vor der Berücksichtigung von Erkenntnissen, z. B. aus Messungen, liefert. Darüber hinaus hängt $p(y)$ nicht von x ab und wird für alle x gleich sein. Daher werden die Gleichungen (2.13) und (2.14) wie folgt geschrieben:

$$p(x \mid y) = \eta p(y \mid x)\, p(x)\,, \tag{2.15}$$

wobei η eine Normalisierungskonstante ist, sodass das Ergebnis von Gleichung (2.15) auf eins normalisiert wird. Zusätzlich ist es möglich, die Regel auf mehrere Zufallsvariablen anzuwenden. Für zwei bedingte Zufallsvariablen X, Z ist die Bayes-Regel wie folgt definiert:

$$p(x \mid y, z) = \frac{p(y \mid x, z)\, p(x \mid z)}{p(y \mid z)} = \frac{p(z \mid x, y)\, p(x \mid y)}{p(z \mid y)}. \tag{2.16}$$

2.3 Erwartungswert und Varianz

Die in dieser Arbeit verwendeten Algorithmen setzen die Berechnung von Statistiken aus Wahrscheinlichkeitsverteilungen voraus. Die wichtigsten sind dabei der Erwartungswert und die Varianz bzw. Kovarianz.

2.3.1 Erwartungswert

Der Erwartungswert ist der erwartete Wert einer Zufallsvariablen für einen unendlich wiederholten Prozess und wird als gewichtetes Mittel aller möglichen Werte der Zufallsverteilung berechnet. Er wird für den diskreten und kontinuierlichen Fall von Zufallsvariablen wie folgt definiert:

$$E[X] = \sum_X xp\,(x) \tag{2.17}$$

$$E[X] = \int_X xp\,(x)\,dx. \tag{2.18}$$

Eine wichtige Eigenschaft des Erwartungswertes ist die Linearität in Bezug auf die Zufallsvariablen - es gilt $E[aX + b] = aE[X] + b$, wenn $a, b \in \mathcal{R}$.

2.3.2 Varianz und Kovarianz

Die Varianz σ^2 misst die quadrierte erwartete Abweichung σ einer einzelnen Zufallsvariablen von dem aus der Erwartungsstatistik erhaltenen Mittelwert.

Für multivariate Wahrscheinlichkeiten wird die Kovarianzmatrix $\boldsymbol{P} \in \mathbb{R}^{n\times n}, n \in \mathbb{N}$ verwendet. Sie stellt nicht nur die Varianz jeder einzelnen Zufallsvariablen dar, sondern auch die Korrelation zwischen jedem Variablenpaar. Ein Beispiel wird für zwei Zufallsvariablen X, Y gegeben:

$$\begin{aligned}\boldsymbol{P}(X,Y) =& E\left[(X - E[X])\,(Y - E[Y])\right] \\ =& \begin{bmatrix} \sigma^2(x) & \sigma(x,y) = \sigma(x)\sigma(y) \\ \sigma(y,x) = \sigma(y)\sigma(x) & \sigma^2(y) \end{bmatrix},\end{aligned} \tag{2.19}$$

wobei $\sigma(x, y)$ angibt, wie eine Zufallsvariable durch die Änderung einer anderen Zufallsvariable beeinflusst wird. Wenn die Korrelation positiv ist, ändern sich beide Variablen auf die gleiche Weise. Wenn die Korrelation negativ ist, wird bei einem Anstieg einer Zufallsvariable eine Abnahme der anderen angenommen. Wenn zwei Zufallsvariablen unabhängig (unkorreliert) sind, ist die Kovarianz Null.

Darüber hinaus gibt es einige Eigenschaften, die für Kovarianzmatrizen gelten:

- **Bilinear** für Konstanten a, b und Zufallsvariablen X, Y, Z gilt: $\sigma(ax+by, z) = a\sigma(x, z) + b\sigma(y, z)$.

- **Symmetrisch** $\sigma(x, y) = \sigma(y, x)$.
- **Positive semi-definite Matrix** $\sigma^2(x) = \sigma(x, x) \geq 0$ für alle Zufallsvariablen X und wobei $\sigma(x, x) = 0$ impliziert, dass X eine konstante Zufallsvariable ist.

Wenn außerdem eine Multivariate $\boldsymbol{x}$ durch eine lineare Transformationsmatrix $\boldsymbol{H}$ transformiert wird, muss die zugehörige Kovarianzmatrix aufgrund ihrer Ableitung der Erwartungsstatistik wie folgt transformiert werden:

$$\boldsymbol{P}(\boldsymbol{H}\boldsymbol{x}) = \boldsymbol{H}\boldsymbol{P}(\boldsymbol{x})\boldsymbol{H}^T. \tag{2.20}$$

2.4 Wahrscheinlichkeitsverteilungen

Das zugrundeliegende Prinzip der Bayesschen Inferenz ist die a-priori und a-posteriori Wahrscheinlichkeitsverteilung über Zufallsvariablen. Wenn der kontinuierliche Zufallsvektor $\boldsymbol{x} = (X_1, X_2, \ldots, X_n)$ eine multivariate Gauß'sche Verteilung $\boldsymbol{x} \sim \mathbb{N}(\boldsymbol{\mu}, \boldsymbol{P})$ hat, ist die Wahrscheinlichkeitsdichtefunktion (WDF) wie folgt definiert:

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{P}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{P}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right), \tag{2.21}$$

mit Mittelwert $\boldsymbol{\mu} \in \mathbb{R}^n$ und symmetrischen und positiver semidefiniter Kovarianzmatrix $\boldsymbol{P}$. Darüber hinaus summieren sich diskrete Wahrscheinlichkeitsverteilungen immer zu eins, während eine WDF immer zu eins integriert. Gauß'sche Modelle sind beispielsweise eine beliebte Darstellungsweise von Messunsicherheiten.

2.5 Bayes Filter

Der Bayes Filter ist ein Framework zur rekursiven Zustandsschätzung und es gibt verschiedene Implementierungen und Realisierungen. Im Folgenden werden einige grundlegende Konzepte des Bayes-Filter-Algorithmus beschrieben. Anschließend wird der Gauß'sche Filter sowie nicht parametrische Filter als Implementierungen von Bayes Filtern vorgestellt.

In einer perfekten endlichen Welt mit n Zuständen kann der Zustandsraum als $\boldsymbol{X} = \{\boldsymbol{x}_k\}_{k=0}^n$ beschrieben werden, wobei die Ausgangsbedingungen $\boldsymbol{x}_0$ zum Zeitschritt $k = 0$ bekannt sind. Die Zustandsübergangsfunktion f überführt den alten Zustand $\boldsymbol{x}_k$ in einen neuen Zustand $\boldsymbol{x}_{k+1}$:

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k). \tag{2.22}$$

Folglich können alle Zustände durch die kontinuierliche und akkumulierte Berechnung bestimmt werden.

Diese Formulierung beschreibt jedoch nicht die reale Welt, denn dort beeinflussen z. B. Umwelteinflüsse den Zustandsübergang maßgeblich. Folglich ist $\boldsymbol{X}$ nur eine Teilmenge der Welt und Gleichung (2.22) ist ein zu ungenaues Modell.

Für die meisten wissenschaftlichen Disziplinen ist es somit nötig, Modelle zu entwickeln, die eine exakte Vorhersage über Zustände ermöglichen. In der probabilistischen Robotik hängt der Erfolg des Roboters beispielsweise maßgeblich vom menschlichen Programmierer ab, der relevante kinematische Eigenschaften und externe Einflüsse anhand eines physikalisch korrekten Modells abbilden muss. Eine gängige Methode in der Robotik ist dabei die Annahme der geschlossenen Welt (engl. closed world formulation). Bei dieser Formulierung trifft man die vereinfachte Annahme, dass das Welt-Modell alle Parameter, welche für die korrekte Ausführung nötig sind, beinhaltet. Äußere Einflüsse oder nicht-deterministische Verhaltensweisen werden dabei mit einem Rauschterm $\boldsymbol{v}_k$ modelliert:

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k) + \boldsymbol{v}_k. \tag{2.23}$$

In der Praxis ist der Anfangszustand $\boldsymbol{x}_0$ oftmals nicht ausreichend genau bekannt, sodass dieser ebenfalls annähernd bestimmt werden muss. Die Prädiktion des Zustandes erfolgt dabei nur anhand des vorhergesagten Zustands $f(\hat{\boldsymbol{x}}_k)$ und den äußeren Einflüssen $\boldsymbol{v}_k$. Darüber hinaus ist das Prädiktionsmodell oftmals nicht ausreichend genau und die Schätzung weicht folglich vom wahren Zustand nach mehreren Zeitschritten stark ab.

Um diese Abweichung zu korrigieren sind Beobachtungen $\boldsymbol{y}_k$ bezüglich des tatsächlichen Zustands notwendig. Die Messgleichung ist wie folgt definiert:

$$\boldsymbol{y}_k = g(\boldsymbol{x}_k) + \boldsymbol{w}_k, \tag{2.24}$$

mit Messfunktion $g(\boldsymbol{x}_k)$ und Rauschterm $\boldsymbol{w}_k$.

Die Kombination der Gleichungen (2.23) und (2.24) und die Berücksichtigung von Unsicherheiten ist ein allgemeines Problem der Zustandsschätzung. Die Bayessche Inferenz ist dabei eine verbreitete Methode zur Lösung dieser Aufgabe.

2.5.1 Probabilistisches Zustandsraummodell

Das Ziel der Zustandsraummodellierung ist die optimale Schätzung eines unbekannten Zustands durch die Verwendung von Beobachtungen zu erhalten, die als rekursive Form der Bayessche Regel abgeleitet werden kann. Die Zustandsübergangswahrscheinlichkeit und die Messwahrscheinlichkeit erlauben dabei die Modellierung von Zustandsübergängen und Messungen.

2.5.1.1 Zustandsübergangswahrscheinlichkeit

Die Zustandsübergangswahrscheinlichkeitsverteilung geht von der Markov-Struktur des Zustands aus und wird wie folgt modelliert:

$$\boldsymbol{x}_k \sim p\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}\right) = p\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{0:k-1}, \boldsymbol{u}_{0:k-1}\right), \tag{2.25}$$

wobei der Kontrollvektor $\boldsymbol{u}_{k-1}$ eine Möglichkeit darstellt, eine Zustandsänderung vom alten Zustand $\boldsymbol{x}_{k-1}$ im vorherigen Zeitschritt $k-1$ zu einem neuen Zustand $\boldsymbol{x}_k$ zu bewirken, unter Annahme der Markov-Bedingung.

Die Markov-Bedingung ist eine Annahme in der Bayesschen Wahrscheinlichkeitstheorie, die besagt, dass der aktuelle Zustand unabhängig von allen Nicht-Eltern ist. Das bedeutet für ein dynamisches System, dass bei gegebenem aktuellem Zustand alle folgenden Zustände unabhängig von allen vergangenen Zuständen sind [Särkkä, 2010]. Folglich ist $\boldsymbol{x}_k$ ein vollständiger Zustand und speicherfrei, der alle Informationen beinhaltet, die zur Modellierung des Zustands erforderlich sind. Die Markov-Eigenschaft verstößt jedoch gegen die Geschlossene-Welt-Annahme, da es in der Praxis immer Ereignisse gibt, die nicht im Zustand kodiert sind und zukünftige Zustände beeinflussen können.

2.5.1.2 Messwahrscheinlichkeit

Die Messwahrscheinlichkeit gibt an, unter welcher Wahrscheinlichkeit und Abhängigkeit eine Messung erzeugt worden ist. Da der Zustand $\boldsymbol{x}_k$ als vollständig und speicherfrei modelliert wird, ist die Messung $\boldsymbol{y}_k$ bedingt unabhängig von vergangenen Messungen, Aktionen und Zuständen:

$$\boldsymbol{y}_k \sim p\left(\boldsymbol{y}_k \mid \boldsymbol{x}_k\right) = p\left(\boldsymbol{y}_k \mid \boldsymbol{y}_{1:k-1}, \boldsymbol{x}_{0:k}, \boldsymbol{u}_{0:k-1}\right). \tag{2.26}$$

Bei SLAM wird zusätzlich die Karte $\boldsymbol{M}$ berücksichtigt, da diese a-priori Informationen zu Objekten in der Umgebung und somit zu möglichen Messungen liefert.

2.5.1.3 Messmodell

Das Messmodell beschreibt, wie sich Messwerte aus dem Systemzustand ergeben und ermöglicht unter Berücksichtigung der aktuellen Schätzung der Fahrzeugposition $\boldsymbol{x}_k$ und der Karte $\boldsymbol{M}$ die Messung $\boldsymbol{y}_k$ zu beschreiben:

$$\boldsymbol{y}_k = g\left(\boldsymbol{x}_k, \boldsymbol{M}\right) + \boldsymbol{w}_k. \tag{2.27}$$

LiDAR-Sensoren liefern Distanzen, welche über die Laufzeit eines Lichtimpulses gemessen werden. Ein bekanntes Messmodell für einen LiDAR-Sensor ist z. B. die Berechnung von kartesischen Koordinaten anhand der Distanzmessung sowie dem horizontalen und vertikalen Einbauwinkel der Laserdiode. Im Idealfall ist die Laufzeit direkt proportional zur Distanz zu einem sich im Sichtstrahl befindlichen Hindernis. In der Realität weichen jedoch die gemessenen Distanzen durch systembedingte Störungen oder durch externe Einflüsse von der echten Distanz ab. Bei LiDAR-Sensoren zählen hierzu insbesondere reflektierende oder lichtdurchlässige Oberflächen. Aber auch durch Störeinflüsse von z. B. anderen LiDAR-Sensoren kommt es zu Fehlmessungen.

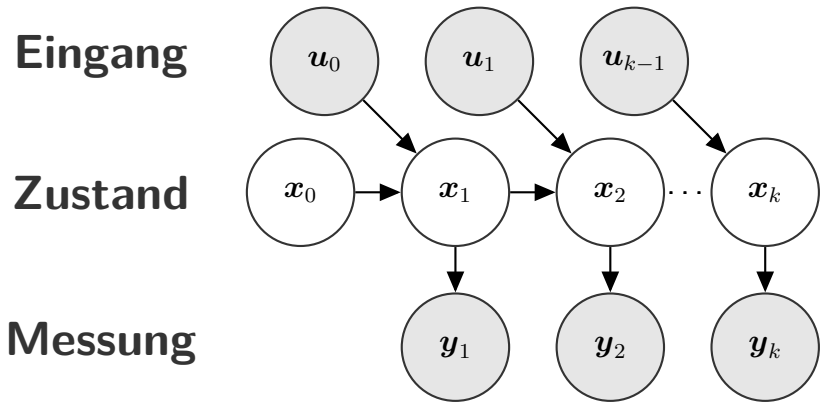


Abbildung 2.3:
Das Hidden-Markov-Modell (HMM) charakterisiert die Entwicklung von Aktionen, Zuständen und Messungen. Der Zustand im Zeitschritt k ist stochastisch abhängig vom Zustand im Zeitschritt $k-1$ und dem Steuereingang $\boldsymbol{u}_{k-1}$. Der Messwert $\boldsymbol{y}_k$ hängt stochastisch vom Zustand zum Zeitpunkt k ab, jedoch nicht von den vorherigen Messungen.

Darüber hinaus können vorverarbeitete Messdaten, wie beispielsweise die Position und Dimension von Objekten, als Messung im Messmodell verwendet werden, wobei die Menge an Sensormessungen $\boldsymbol{Y}_k$ wie folgt definiert ist:

$$\boldsymbol{Y}_k = \{\boldsymbol{y}_k\}_{k=0}^{K}. \tag{2.28}$$

Die in Unterabschnitte 2.5.1.1 und 2.5.1.2 vorgestellten Zustandsübergänge und Messwahrscheinlichkeiten bilden die Grundlage der Bayes Filter und können auch als Hidden Markov Model (HMM) dargestellt werden. Eine Visualisierung des HMM ist in Abbildung 2.3 dargestellt.

2.5.2 Belief Distribution und Bayes Filter

Die von Thrun et al. [2005] beschriebene *belief distribution* eines Zustands $\boldsymbol{x}_k$ ist eine elegante Art und Weise, die a-posteriori Wahrscheinlichkeitsverteilung zum Zeitschritt k zu beschreiben. Der gesuchte Zustand $\boldsymbol{x}_k$ kann meistens nicht direkt gemessen werden und muss daher auf Basis der vergangenen Messungen $\boldsymbol{y}_{1:k}$ und den vergangenen Aktionen $\boldsymbol{u}_{0:k-1}$ probabilistisch geschätzt werden:

$$\texttt{bel}\left(\boldsymbol{x}_k\right) = p\left(\boldsymbol{x}_k \mid \boldsymbol{y}_{1:k}, \boldsymbol{u}_{0:k-1}\right). \tag{2.29}$$

Die prädizierte *belief distribution*, also die vorhergesagte a-posteriori Verteilung auf Basis der aktuellsten Aktion und ohne Berücksichtigung der aktuellen Messung $\boldsymbol{y}_k$, ist wie folgt definiert:

$$\overline{\texttt{bel}}\left(\boldsymbol{x}_k\right) = p\left(\boldsymbol{x}_k \mid \boldsymbol{y}_{1:k-1}, \boldsymbol{u}_{0:k-1}\right). \tag{2.30}$$

Der allgemeine Bayes Filter ist als zweistufiger rekursiver Prozess formuliert, der die *belief distribution* $\texttt{bel}\,(\boldsymbol{x}_k)$ auf Basis von Messungen und Aktionen bestimmt:

$$\overline{\texttt{bel}}\,(\boldsymbol{x}_k) = \int p\,(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1})\,\texttt{bel}\,(\boldsymbol{x}_{k-1})\,d\boldsymbol{x}_{k-1} \tag{2.31}$$

$$\texttt{bel}\,(\boldsymbol{x}_k) = \eta\,p\,(\boldsymbol{y}_k \mid \boldsymbol{x}_k)\,\overline{\texttt{bel}}\,(\boldsymbol{x}_k)\,, \tag{2.32}$$

wobei die Prädiktion, siehe Gleichung (2.31), anhand der Chapman-Kolmogorov-Gleichung [Bronstejn, 2012] und dem Integral (Summe) über das Produkt der Zustandsübergangswahrscheinlichkeit, siehe Unterabschnitt 2.5.1.1, sowie dem a-priori *belief* des Zustand $\boldsymbol{x}_{k-1}$ bestimmt wird.

Die Bestimmung der *belief distribution* $\texttt{bel}\,(\boldsymbol{x}_k)$ anhand der prädizierten *belief distribution* $\overline{\texttt{bel}}$ in Gleichung (2.32) wird Messupdate oder Korrekturschritt genannt und basiert auf der Messwahrscheinlichkeit, siehe Unterabschnitt 2.5.1.2, der vorhergesagten *belief distribution* $\overline{\texttt{bel}}\,(\boldsymbol{x}_k)$ sowie der Normierungskonstante η, gemäß der Bayes-Regel von Gleichung (2.15).

Die Gleichung (2.31) und Gleichung (2.32) sind die Grundform der allgemeinen Bayes-Filterformulierungen. Eine ausführliche Herleitung ist in Thrun et al. [2005] beschrieben.

2.5.3 Kalman Filter

Das Kalman Filter (KF) wurde in den 1960er Jahren erfunden und ist eine Technik zur Filterung und Vorhersage von linearen Gauß'schen Systemen mit kontinuierlichen Zuständen [Kalman, 1960]. Es handelt sich um eine geschlossene Lösung des linearen Bayes'schen Filterproblems unter den Markov-Annahmen, um nicht direkt messbare Systemgrößen zu schätzen. Es werden hierzu kontinuierlich die prädizierten und die tatsächlichen Messwerte verglichen und der Mittelwert sowie die Kovarianz der Zustandsgrößen ermittelt. Initial sind die Rauschfaktoren sowie der Anfangszustand zu definieren. Im Vergleich zum diskreten Zustandsraum des HMM kann der KF mit kontinuierlichen Zustandsvariablen umgehen.

Das probabilistische Zustandsraummodell aus Gleichungen (2.25) und (2.26) kann durch die linearen Funktionsargumente als lineare Rekursionsgleichungen formuliert werden:

$$\boldsymbol{x}_k = \boldsymbol{\Phi}_{k-1}\boldsymbol{x}_{k-1} + \boldsymbol{B}_{k-1}\boldsymbol{u}_{k-1} + \boldsymbol{v}_{k-1} \tag{2.33}$$

$$\boldsymbol{y}_k = \mathbf{C}_k\boldsymbol{x}_k + \boldsymbol{w}_k, \tag{2.34}$$

mit den Zustandsvektoren $\boldsymbol{x}_k, \boldsymbol{x}_{k-1}$ und Aktionsvektor $\boldsymbol{u}_{k-1}$, der Übergangsmatrix des dynamischen Modells $\boldsymbol{\Phi}_{k-1}$, Eingangsmatrix $\boldsymbol{B}_{k-1}$, Prozessrauschen $\boldsymbol{v}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1})$, die Messmodell-Matrix $\boldsymbol{C}_k^{n\times n}$ und das additive Messrauschen $\boldsymbol{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$.

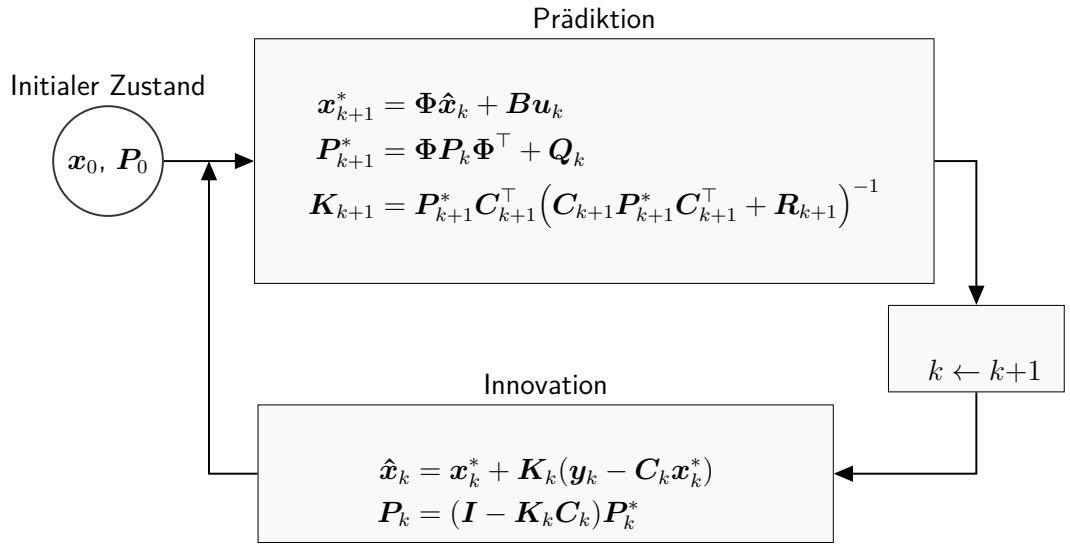


Abbildung 2.4:
Diskreter Kalman Filter zur rekursiven Zustandsschätzung mit dem prädizierten Zustand $\boldsymbol{x}^*_{k+1}$ und der prädizierter Kovarianzmatrix $\boldsymbol{P}^*_{k+1}$ sowie nach dem Innovationsschritt aktualisierter Zustand $\hat{\boldsymbol{x}}_k$.

Das probabilistische Zustandsraummodell ergibt sich daraus wie folgt:

$$p(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) = \mathcal{N}(\boldsymbol{x}_k \mid \boldsymbol{\Phi}_{k-1}\boldsymbol{x}_{k-1} + \boldsymbol{B}_{k-1}\boldsymbol{u}_{k-1}, \boldsymbol{Q}_{k-1}) \tag{2.35}$$

$$p(\boldsymbol{y}_k \mid \boldsymbol{x}_k) = \mathcal{N}(\boldsymbol{y}_k \mid \boldsymbol{C}_k\boldsymbol{x}_k, \boldsymbol{R}_k), \tag{2.36}$$

mit Systemkovarianzmatrix $\boldsymbol{Q}_{k-1} \in \mathbb{R}^{n\times n}$ und Messkovarianzmatrix $\boldsymbol{R}_k \in \mathbb{R}^{m\times m}$.

Für die Zustandsübergangswahrscheinlichkeit $p(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1})$ und die Messwahrscheinlichkeit $p(\boldsymbol{y}_k \mid \boldsymbol{x}_k)$ gilt dann:

$$p\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}\right) = \frac{1}{det(2\pi\boldsymbol{Q}_{k-1})} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_k - \boldsymbol{\mu}\right)^T {\boldsymbol{Q}_{k-1}}^{-1}(\boldsymbol{x}_k - \boldsymbol{\mu})\right), \tag{2.37}$$

$$p\left(\boldsymbol{y}_k \mid \boldsymbol{x}_k\right) = \frac{1}{det(2\pi\boldsymbol{R}_{k-1})} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_k - \mathbf{C}_k\boldsymbol{x}_k\right)^T {\boldsymbol{R}_{k-1}}^{-1}(\boldsymbol{x}_k - \mathbf{C}_k\boldsymbol{x}_k)\right), \tag{2.38}$$

wobei der Mittelwert der posteriori Verteilung wie folgt bestimmt wird: $\boldsymbol{\mu} = \boldsymbol{\Phi}_{k-1}\boldsymbol{x}_{k-1} + \boldsymbol{B}_{k-1}\boldsymbol{u}_{k-1}$.

Darüber hinaus gilt, dass die a-posteriori Verteilung Gauß-verteilt ist, wenn die initiale Verteilung ebenfalls eine Gauß-Verteilung ist, sodass gilt:

$$\boldsymbol{x}_0 \sim \mathcal{N}(\boldsymbol{x}^*_0, \mathbf{P}_0), \tag{2.39}$$

mit Mittelwert $\boldsymbol{x}^*_0 \in \mathbb{R}^n$ und initiale Kovarianzmatrix $\mathbf{P}_0 \in \mathbb{R}^{n\times n}$. Abbildung 2.4 stellt den rekursiven Schätzprozess als Blockschaltbild dar.

Zusammengefasst ist das KF ein optimaler Filter für Probleme der linearen Zustandsschätzung. Lineare Funktionen modellieren dabei die Transaktion von einem vorherigen Zustand zu einem neuen Zustand, wobei das Gleiche für Messungen gilt. Eine vollständige mathematische Herleitung des KF ist in [Kalman, 1960] zu finden.

In der realen Welt erfüllen jedoch nur die wenigsten Anwendungen die Bedingung der Linearität und somit wurden mehrere nichtlineare Erweiterungen entwickelt, siehe Bar-Shalom et al. [1990], Julier [2002], Särkkä [2010], Senne [1972], Wan und Van Der Merwe [2000]. Die gängigsten Implementierungen, das Extended Kalman Filter (EKF) und das Unscented Kalman Filter (UKF), werden im Folgenden kurz eingeführt.

2.5.4 Extended Kalman Filter

Das Extended Kalman Filter (EKF) ist das am häufigste verwendete Filter, um Messungen oder Beobachtungen von realen Prozessen mit Sensoren zu erfassen. Im Vergleich zum KF benötigt das EKF keinen linearen Zustandsübergang, da um den aktuellen Mittelwert eine Linearisierung mithilfe der Taylor-Approximation durchgeführt wird [Särkkä, 2010, Senne, 1972]. EKF's werden oftmals für die Eigenbewegungsschätzung verwendet, bei der die Übergangswahrscheinlichkeit $p\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}\right)$ mit einem physikalischen Bewegungsmodell approximiert wird.

Die Zustandsübergänge beim EKF sind wie folgt definiert:

$$\boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) + \boldsymbol{v}_{k-1} \tag{2.40}$$

$$\boldsymbol{y}_k = g(\boldsymbol{x}_k) + \boldsymbol{w}_k, \tag{2.41}$$

mit der nichtlinearen Zustandsgleichungen $f(\cdot)$ und der nichtlinearen Messgleichung $g(\cdot)$. Die Übergangsmatrix und die Messmodell-Matrix werden durch die Berechnung der entsprechenden Jacobi-Matrix approximiert:

$$\boldsymbol{\Phi}_{k-1} = \frac{\partial f}{\partial \boldsymbol{x}} \mid_{\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_{k-1}}, \tag{2.42}$$

$$\mathbf{C}_k = \frac{\partial g}{\partial \boldsymbol{x}} \mid_{\boldsymbol{x}_k^*} . \tag{2.43}$$

Folglich wird die Approximation der Funktionen f, g am Arbeitspunkt $\boldsymbol{\mu}_{k-1}$ und $\overline{\boldsymbol{\mu}}_k$ wie folgt definiert:

$$f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) \approx f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) + \boldsymbol{\Phi}_{k-1} \left(\boldsymbol{x}_{k-1} - \boldsymbol{\mu}_{k-1}\right) \tag{2.44}$$

$$g(\boldsymbol{x}_k) \approx g(\boldsymbol{x}_k) + \mathbf{C}_k \left(\boldsymbol{x}_k - \overline{\boldsymbol{\mu}}_k\right). \tag{2.45}$$

Der Hauptvorteil des EKF ist dessen Einfachheit für nicht-lineare Systeme. Für hochgradig nichtlineare Systeme ist die Leistung jedoch aufgrund der einfachen nichtlinearen Approximation der Ableitung erster Ordnung nicht ausreichend.

Im Folgenden wird das UKF vorgestellt, der den Mittelwert der Zielverteilung und die Kovarianz durch sogenannte Sigma-Punkte direkt approximiert. Das Aufstellen von nichtlinearen Abbildungsgleichungen entfällt dabei.

2.5.5 Unscented Kalman Filter

Das Unscented Kalman Filter (UKF) wurde zunächst von Wan und Van Der Merwe [2000] vorgestellt und hat seinen Ursprung in der Unscented Transform (UT). Die UT ist eine Methode zur Schätzung der Statistik einer Zufallsvariablen, die eine nicht-lineare Transformation durchläuft [Julier und Uhlmann, 1997].

Die grundlegende Idee ist, eine Gauß-Verteilung zu approximieren und nicht eine nichtlineare Funktion. Das UKF verwendet hierzu eine Menge von Sigma-Punkten, die aus dem aktuellen Mittelwert und der Kovarianz gezogen werden. Anschließend werden die Sigma-Punkte gemäß einer nichtlinearen Funktionen g transformiert, um den a-posteriori Mittelwert und die Kovarianz aus einer Gauß'schen Verteilung zu ermitteln, anstatt die nichtlineare Funktion durch eine Taylor-Reihe zu approximieren. Diese Transformation wird als UT bezeichnet, von der sich der Name UKF ableitet. Für die Multivariate $\boldsymbol{x}$ mit der Dimension n, die sich durch eine nichtlineare Funktion g propagiert, wird angenommen, dass $\boldsymbol{\mu}$ der Mittelwert und $\boldsymbol{P}$ die Kovarianz ist.

Um die Statistik der nichtlinearen Funktion zu berechnen, wird eine Matrix von $2n+1$ Sigma-Punkten erzeugt:

$$\mathcal{X}_0 = \boldsymbol{\mu} \tag{2.46}$$

$$\mathcal{X}_i = \boldsymbol{\mu} + \left(\sqrt{(n+\kappa)\,\boldsymbol{P}^*}\right)_i \qquad i = 1, \ldots, n \tag{2.47}$$

$$\mathcal{X}_i = \boldsymbol{\mu} - \left(\sqrt{(n+\kappa)\,\boldsymbol{P}^*}\right)_{i-n} \qquad i = n+1, \ldots, 2n, \tag{2.48}$$

mit Skalisierungsparametern $\lambda = \alpha^2\,(n+\kappa) - n$ und Konstante α die als Faktor Einfluss auf die Streuung der Sigma-Punkte um den Mittelwert $\boldsymbol{\mu}$ hat. Zur optimalen Approximation einer Gauß-Verteilung empfiehlt Julier [2002] für die Scaled Unscented Transform folgende Parameter zu wählen: $\kappa > 0$, $\alpha \in (0, 1)$ und $\beta = 2$.

Die Sigma-Punkte werden durch die nichtlineare Funktion g abgebildet:

$$\mathcal{Y} = g(\mathcal{X}_i) \quad i = 0, \ldots, 2n. \tag{2.49}$$

Der Mittelwert $\boldsymbol{\mu_U} \in \mathbb{R}^m$ und die Kovarianz $\boldsymbol{S_U} \in \mathbb{R}^{m \times m}$ werden anschließend wie nach Chang et al. [2013], Särkkä [2010] approximiert:

$$\boldsymbol{\mu}_U = \sum_{i=0}^{2n} W_i^l \mathcal{Y}^i \tag{2.50}$$

$$\boldsymbol{S}_U = \sum_{i=0}^{2n} W_i^c (\mathcal{Y}^i - \boldsymbol{\mu}_U)(\mathcal{Y}^i - \boldsymbol{\mu}_U)^\top, \tag{2.51}$$

wobei die Gewichte W_i wie folgt berechnet werden:

$$W_0^a = \lambda/n+\lambda \tag{2.52}$$
$$W_0^c = \lambda/n+\lambda + (1 - \alpha^2 + \beta) \tag{2.53}$$
$$W_i^a = W_i^c = 1/2(n+\lambda) \qquad i = 1, \ldots, 2n. \tag{2.54}$$

Zur vollständige Herleitung der in diesem Abschnitt verwendeten Gleichungen sei auf Chang et al. [2013], Julier und Uhlmann [1997], Särkkä [2010] verwiesen.

Durch die effektive Abbildung der Wahrscheinlichkeitsdichtefunktion (WDF) bietet das UKF eine genauere Approximation bis zur zweiten Ordnung des Mittelwertes und der Kovarianz, aber nur, wenn die entsprechenden Zufallsverteilungen durch eine Gaußsche-Verteilung approximiert werden können. Das UKF ist auch ein Gaußscher-Filter und daher nicht in der Lage, nicht-gaußsche multimodale Verteilungen zu schätzen. Aus diesem Grund wird im nächsten Abschnitt das Partikel-Filter (PF) vorgestellt.

2.5.6 Partikel-Filter

Das Partikel-Filter (PF) oder auch sequentielle Monte Carlo Methode (SMC) genannt ist ein nichtparametrischer rekursiver Bayes Filter. Die erste Implementierung dieses Filteransatzes wurde 1993 von Gordon et al. [1993] vorgestellt.

Die Grundidee ist, eine multimodale Wahrscheinlichkeitsverteilung durch eine beliebige Anzahl von Stichproben $\boldsymbol{x}_k^{(i)}$ (engl. random samples) zu approximieren. Folglich ist die Anzahl der Stichproben für die Genauigkeit der Approximation verantwortlich. Darüber hinaus steigen mit der Anzahl der Stichproben auch entsprechend die erforderlichen Rechenressourcen.

Die Kombination aus Stichprobe und dazugehörigem Gewicht $\{\boldsymbol{x}_k^{(i)}, w_k^{(i)}\}$ wird als Partikel bezeichnet und repräsentiert jeweils einen möglichen Zustand bzw. eine Stützstelle der gesuchten WDF. Im Idealfall gilt somit: je dichter eine Teilregion des Zustandsraums von Partikeln bevölkert ist, desto wahrscheinlicher ist es, dass der tatsächliche Systemzustand in dieser Teilregion zu finden ist [Thrun et al., 2005]. Das Gewicht drückt aus, wie gut das aktuelle Partikel den wahren Zustand approximiert, gegeben die aktuelle Messung.

In seiner einfachsten Form propagiert ein Partikel-Filter Partikel, welche sich dem Prior im Zustandsraum nähern, unter Verwendung eines Zustandsübergangsmodells. Im nächsten Schritt werden die Partikelgewichte unter Berücksichtigung der Beobachtungswahrscheinlichkeit bestimmt. Dieser rekursive Prozess führt zu einem verbesserten Schätzergebnis der gesuchten Zustände des dynamischen Systems.

Die Formeln der in dieser Arbeit verwendeten Partikel-Filter-basierten Lokalisierungsmethode werden in Unterabschnitt 3.1.2.4 beschrieben.

2.6 Versuchsträger

Abbildung 2.5:
Die Versuchsfahrzeuge MuCAR-3 (rechts) und MuCAR-4 (links) sind ausgestattet mit verschiedener Sensorik, um das Umfeld wahrzunehmen und die eigene Position und Geschwindigkeit zu schätzen.

Im Folgenden, werden die in dieser Arbeit verwendeten autonomen Fahrzeuge beschrieben. Die Institutsfahrzeuge mit den Namen MuCAR-3 und MuCAR-4 können sowohl als Führungs- sowie als Folgefahrzeug im autonomen Konvoi eingesetzt werden. Das MuCAR-3 basiert auf einem serienmäßigen VW Touareg mit einem V6 TDI-Motor, der so modifiziert wurde, dass mittels Drive-by-Wire die Steuerung der Lenkung, Bremse, Gas und des Automatikgetriebes möglich ist.

Das MuCAR-4 ist ein VW Tiguan und besitzt vergleichbare Sensorik sowie einen ähnlichen Aufbau wie das MuCAR-3. Beide Fahrzeuge sind jeweils mit einem Velodyne HDL-64 LiDAR-Sensor ausgestattet, der auf dem Dach montiert ist. Die Befestigungsposition und der Sichtbereich der Sensoren sind jedoch durch die Bauart der Dachkonstruktion und weiterer auf dem Dach montierter Sensorik, die den Sichtbereich einschränken, verschieden. Daher wird im folgenden exemplarisch nur auf die technische Ausstattung des MuCAR-3 eingegangen.

2.6.1 Wahrnehmungssensorik

Um die Umgebung wahrzunehmen, sind Kameras, Stereo-Kameras, eine Wärmebildkamera und LiDAR-Sensoren auf den Versuchsfahrzeugen verbaut, siehe Abbildung 2.5.

Das Hauptkamerasystem von MuCAR-3 ist die MarVEye-8, ein multifokales Aktiv-Reaktiv-Sichtsystem, das zwischen der Windschutzscheibe und dem Rückspiegel montiert ist [Unterholzner, 2015]. Das System besteht aus drei Farbkameras, wobei zwei Kameras mit Weitwinkelobjektiven für Stereosehen ausgestattet sind und eine Kamera mit einem Teleobjektiv.

Zusätzlich ist eine Wärmebildkamera auf dem Dach montiert, die die Erkennung von Objekten bei schlechten Lichtverhältnissen ermöglicht. Neben dem Velodyne HDL-64 LiDAR-Sensor ist im MuCAR-3 noch ein seriennaher Achtlinien-LiDAR-Sensor der Firma IBEO[1] sowie ein Radio Detection And Ranging (Radar)-Sensor von smartmicro[2] im vorderen Stoßfänger verbaut.

In dieser Arbeit werden ausschließlich die 3D-Punktwolken des Velodyne HDL-64 LiDAR-Sensors betrachtet, um statische und dynamische Objekte sowie die Bodenebene und die Straße, der zu folgen ist, zu erkennen. Der Sensor liefert Distanzmessungen auf eine Entfernung von bis zu $120\,\mathrm{m}$, mit einem horizontalen Sichtbereich von $360°$ und vertikalem Sichtbereich von $-24.8°$ bis $2°$.

Die Gruppierung von Entfernungsmessungen zu homogenen Objekten ist aufgrund einer ungleichmäßigen Abtastdichte der Laserstrahlen und durch fehlende Strukturinformationen der Umgebung eine besonders komplexe Aufgabe.

Diese Arbeit soll einen wichtigen Beitrag bei der LiDAR-gestützten Umgebungswahrnehmung für autonome Fahrzeuge liefern. Denn je besser die einzelnen Wahrnehmungssysteme sind, umso sicherer wird das Gesamtsystem und umso wahrscheinlicher wird das Erreichen der fünften Stufe für autonome Fahrzeuge, siehe Kapitel 1.

2.6.2 Software- und Hardwarearchitektur

In den Institutsfahrzeugen wird die Echtzeitdatenbank KogMo-Real-time Database (RTDB) von Goebl und Färber [2007] als Middleware verwendet. Über sogenannte RTDB-Objekte kommunizieren die Prozesse und Anwendungen miteinander. Der Datenaustausch erfolgt dabei über einen geteilten Speicherbereich, was die Übertragung sehr effizient gestaltet. Jedes gespeicherte Objekt besitzt neben den zu übertragenen Daten einen Datenzeitstempel sowie eine eindeutige Kennung [Goebl, 2009].

[1] https://www.ibeo-as.com/
[2] https://www.smartmicro.com/

3 Stand der Technik

Inhalt

3.1 Kartierung und Lokalisierung

Die Themen der Kartenerstellung und der Lokalisierung sind in der Forschungsgemeinschaft und insbesondere im Bereich des autonomen Fahrens immer noch eine herausfordernde und spannende Fragestellung. Im Folgenden werden daher die Grundsätze von SLAM besprochen sowie aktuelle und vergangene Entwicklungen anhand der Komponenten Kartierung, Lokalisierung und deren simultane Ausführung beschrieben.

3.1.1 Kartierung

Obwohl intelligente Maschinen die Fähigkeit haben, viele Dinge effizienter zu tun als der Mensch, haben sie noch immer Probleme bei der Szeneninterpretation. Menschen können intuitiv Situationen deutlich besser einschätzen und darüber hinaus überlebensnotwendige Entscheidungen in Echtzeit treffen. Hierzu gehören beispielsweise zur Vermeidung eines Unfalls das Anhalten des Fahrzeugs an der richtigen Stelle und im richtigen Moment.

Als Teil des komplexen Entscheidungsprozesses eines autonomen Fahrzeugs unterstützen Informationen aus Karten speziell in kritischen Szenarien. Konventionelle Landkarten, aber auch digitale Karten aus klassischen Navigationsgeräten, können für eine robuste Fahrzeuglokalisierung und autonome Navigationsaufgaben jedoch nicht verwendet werden. Diese Karten sind einfache und für den Menschen entwickelte Navigationsanweisungen. Im heutigen digitalen Zeitalter werden daher neuartige Karten benötigt, die gezielt für Maschinen und Roboter entwickelt und abgestimmt sind.

Abbildung 3.1: High Definition (HD)-Map von der Firma TomTom.

In den letzten Jahrzehnten wurde aus diesem Grund intensiv an Karten und deren Verwendung geforscht, die speziell für mobile Roboter, autonome Fahrzeuge sowie Fahrassistenzsystemen verwendet werden können [Bauer et al., 2016, Bender et al., 2014, Blochliger et al., 2018, Cummins und Newman, 2008, Douillard et al., 2010, Fisher, 1990, Gran, 2019, Hornung et al., 2013, Huang und Mayer, 2017, Ilci und Toth, 2020, Jang et al., 2018, Matthaei et al., 2014, 2015, Montemerlo und Thrun, 2006, Pathak et al., 2009, Poggenhans et al., 2018, Schindler, 2013, Seif und Hu, 2016, Thrun, 2001, 2002, 2003, TomTom, 2017, Trevor et al., 2014, Waymo Inc., 2017].

Hierzu gehören sogenannte digitale High Density (HD)-Karten, die insbesondere für autonome Fahrzeuge sehr präzise und detaillierte Informationen liefern können. Neben der exakten Geometrie aller Fahrspuren, Fahrspurmarkierungen und vertikalen Landmarken beinhalten HD-Karten oftmals mehrere Datenschichten, die neben Sensor-spezifischen Informationen auch Zusatzinformationen zu Verkehr, Befahrbarkeit der Straße, Geschwindigkeitsvorgaben und auch topologische Informationen bereitstellen. Eine Beispielvisualisierung einer HD-Karte von der Firma TomTom[1] ist in Abbildung 3.1 dargestellt.

Die Erstellung von HD-Karten erfordert Messfahrten mit hochpräziser Sensorik und die Fusion von mehreren Datenquellen zu einer konsistenten Kartenrepräsentation. In aufwendigen Nachbearbeitungsschritten werden die aufgenommenen Daten aufbereitet, registriert und Informationen abstrahiert sowie in einem Weltkoordinatensystem referenziert. Darüber hinaus ist für die Kartenerstellung eine exakte Lokalisierung im globalen Raum notwendig. Kartenanbieter verwenden daher, neben hochpräzisen LiDAR-Sensoren und Kamerasystemen zur Kartierung der Umgebung, differentielles GNSS oder Marker, deren exakte Position bekannt ist. Diese liefern bereits bis in den Sub-Zentimeterbereich genaue Positionsinformationen in globalen Koordinaten. Um darüber hinaus kleine Positionsfehler zu korrigieren, wird in der Nachbearbeitung oftmals ein sogenanntes *Smoothing* durchgeführt. Weitere Informationen hierzu sind in Unterabschnitt 3.2.2 zu finden.

[1] `https://www.tomtom.com/products/hd-map/`

Die Erstellung einer HD-Karte gilt im Allgemeinen als sehr zeit- und ressourcenaufwendig, was sich in hohen Kosten widerspiegelt. Das Angebot von Kartenanbietern ist somit aktuell größtenteils auf Karten von dicht besiedelten Regionen und Autobahnen begrenzt. Auf Baustellen, in ländlichen oder unterentwickelten Gebieten ist daher oftmals gar kein oder kein aktuelles Kartenmaterial verfügbar. Darüber hinaus können Karten von verschiedenen Herstellern sehr unterschiedlich sein, was u. a. an dem verwendeten Fahrzeug, den Sensoren, der Umgebung bzw. dem Einsatzort sowie dem Verwendungszweck liegen kann.

Um diese Herausforderungen heute und in der Zukunft zu meistern, müssen autonome Fahrzeuge die Umgebung mit Sensoren wahrnehmen und intelligent interpretieren, um Unsicherheiten bei der Lokalisierung für eine sichere Navigation zu überbrücken. Insbesondere eine Kombination aus Merkmalskarten und topologischen Karten bieten die benötigte Skalierbarkeit.

Dieser Abschnitt gibt einen Überblick über eine Reihe von Kartentypen, die in der vergangenen und gegenwärtigen Forschung zu SLAM-Verfahren verwendet werden. Grundsätzlich wird zwischen metrischen und topologischen Karten unterschieden. Metrische Karten beschreiben die geometrischen Eigenschaften der Umgebung, wobei topologische Karten die Verbundenheit einzelner Orte repräsentieren [Thrun, 2003].

3.1.1.1 Belegtheitsgitterkarten

Belegtheitsgitterkarten (engl. Occupancy Grid Maps (OGM)), lassen sich sehr einfach aus Entfernungsmessdaten von z. B. Sonar- oder LiDAR-Sensoren erstellen und eignen sich besonders für Umgebungen mit einer dichten und gleichmäßigen Struktur, wie sie beispielsweise in Innenräumen oder auch auf Autobahnen vorkommen.

In den 1980er Jahren waren Sonar-Sensoren für SLAM-Systeme sehr beliebt, da sie einfach verfügbar und kostengünstig waren. Zudem wurde für die Datenverarbeitung nicht viel Rechenleistung benötigt. Occupancy Grid Maps (OGM) waren daher eine gängige Kartenrepräsentation der Umwelt und wurden neben der Lokalisierung auch zur Navigation in realen Szenarien und Wettbewerben eingesetzt [Kammel et al., 2009, Montemerlo et al., 2008, Moravec und Elfes, 1985, Thrun, 2006, Urmson et al., 2009a, Wille et al., 2010].

Bei dieser Kartendarstellung wird die Umgebung in sogenannte metrische diskrete Zellen $m_i \in \mathbb{N} \mid \mathbb{N} = \{1, 2, 3, \ldots\}$ unterteilt, die eine bestimmte Fläche der Umgebung abdecken. Jeder Zelle m_i ist ein Wert zugeordnet, der die Wahrscheinlichkeit $p\left(m_i \mid y_{0:k}, x_{0:k}\right)$ angibt, ob die Zelle m_i von einem Hindernis besetzt ist, diese frei ist oder ob die Zelle noch nicht beobachtet wurde [Bresenham, 1965, Murphy, 1998]. Dabei werden alle bisherigen Sensorbeobachtungen $y_{0:k}$ und Roboterposen $x_{0:k}$ berücksichtigt. Unbeobachtete Zellen werden üblicherweise mit dem Wert von 0.5 initialisiert.

Die Abbildung 3.2 zeigt die Verwendung einer Belegtheitsgitterkarte als Umgebungsrepräsentation, die mittels zweidimensionalen LiDAR-Messungen erstellt worden ist. Der blaue Punkt repräsentiert die Position des Sensors. Die roten Linien stellen den

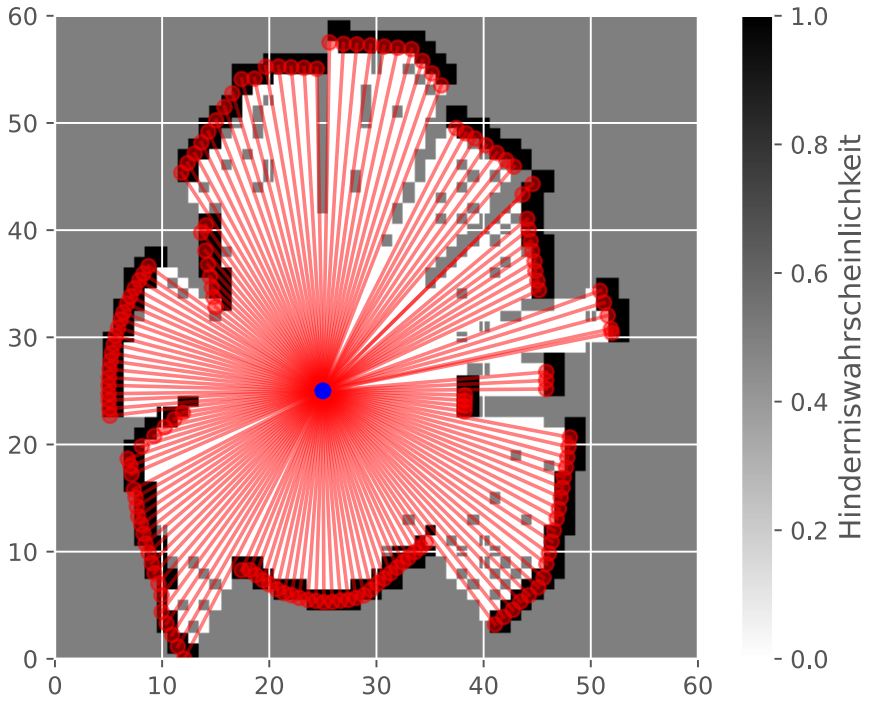


Abbildung 3.2:
Darstellung einer Belegtheitsgitterkarte auf Basis von 2D Entfernungsmessungen eines LiDAR-Sensors. Die Graustufen geben die Hinderniswahrscheinlichkeit pro Zelle an. Schwarze Bereiche repräsentieren Hindernisse, weiße Bereiche Freiraum und graue Bereiche wurden noch nicht beobachtet.

Lichtstrahl eines LiDAR-Sensors dar. Trifft ein Lichtstrahl auf ein Hindernis (roter Kreis), so ist die Entfernung zu diesem Punkt bekannt. Schwarze Zellen sind in diesem Beispiel als Hindernis (Hinderniswahrscheinlichkeit ist eins) und weiße Zellen (Hinderniswahrscheinlichkeit ist null) als Freiraum dargestellt. Graue Zellen hingegen entsprechen Zellen, über die noch keine Informationen (unbeobachtet) vorhanden sind. Dies kann zum einen an der Strahlendichte, an Verdeckungen oder an der Sensorreichweite liegen. Aufgrund ihrer einfachen Struktur werden OGM auch für die Bewegungsplanung und Hindernisvermeidung eingesetzt [Doherty et al., 2016, Fassbender et al., 2016, Jaspers, 2021, Jaspers et al., 2017b, Tanzmeister et al., 2014, Urmson et al., 2009b].

Darüber hinaus gibt es auch dreidimensionale OGM Repräsentationen, die aus 3D-Punktwolken von beispielsweise LiDAR-Sensoren, Stereokameras oder RGB-D Kameras erzeugt werden. Hierzu gehören sogenannte Voxelrepräsentationen.

Ein Voxelgrid beinhaltet Voxel, die einen Körper (Volumen) in einem dreidimensionalen Raum beschreiben. Um die Datenstruktur zu komprimieren, wird nach Hornung et al. [2013] eine sogenannte Baumstruktur erzeugt. Jeder Knoten dieser Datenstruktur repräsentiert ein bestimmtes Volumen und kann dabei bis zu acht weitere Unterteilungen (Kinder) besitzen. Ein Knoten wird jedoch nur dann erweitert und weiter unterteilt, wenn in seinem Volumen eine Messung liegt. Bereiche mit

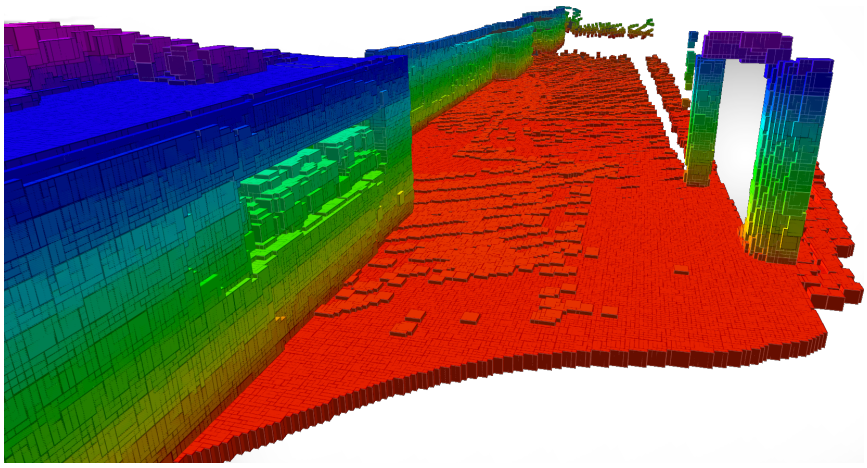


Abbildung 3.3:
3D Voxel-Repräsentation einer Punktwolke aufgenommen im Zwischengeschoss der U-Bahn am Marienplatz München.

vielen Messungen können somit deutlich detaillierter abgebildet werden und Bereiche ohne Messung belegen keinen zusätzlichen Speicher.

Die in Abbildung 3.3 gezeigte dreidimensionale OGM ist ein sogenanntes Voxelgrid, das aus über 90.000 LiDAR-Messungen erstellt wurde. Jeder sichtbare Würfel ($0.5\,\mathrm{cm}^3$) repräsentiert dabei einen Knoten. Die Baumstruktur enthält 20.000 Knoten, was im Vergleich zu den ursprünglichen 90.000 Distanzmessungen eines LiDAR-Sensors eine erhebliche Reduzierung darstellt. Mit Voxelrepräsentationen können auch überhängende Strukturen abgebildet werden, was mit zweidimensionalen OGM nicht möglich ist.

Ein Hybrid stellt die 2.5D OGM dar, die es ermöglicht neben der Wahrscheinlichkeit über die Belegtheit einer Gridzelle weitere Parameter wie beispielsweise Farb- oder Höheninformation in zusätzlichen Schichten zu speichern [Badue et al., 2019, Fox et al., 1999, Jaspers, 2021, Jaspers et al., 2017b, Kammel et al., 2009].

3.1.1.2 Merkmalskarten (Landmarkenkarten)

Merkmalskarten enthalten extrahierte und aussagekräftige Merkmale der Umgebung [Burger et al., 2019a, Fisher, 1990, Jaspers et al., 2017a]. Merkmalskarten eignen sich besonders gut für Bereiche, die deutlich erkennbare und unterscheidbare Merkmale enthalten. So wird die Karte $\mathbb{M}$ durch eine Menge von n Merkmalen $\mathbb{M} = \{\boldsymbol{l}_i\}_{i=0}^{i=n}$ beschrieben. $\boldsymbol{l}_i$ ist der Zustandsvektor eines Merkmales, der beispielsweise die Pixelposition u, v für Bildmerkmale oder die 3D-Position mit x, y, z eines Objekts beinhaltet.

Im Allgemeinen lässt sich zwischen drei verschiedenen Arten von Merkmalen bzw. Landmarken unterscheiden, auf die im Folgenden näher eingegangen wird. Die Merkmale hängen dabei stark von den Sensoren ab, mit denen sie beobachtet werden.

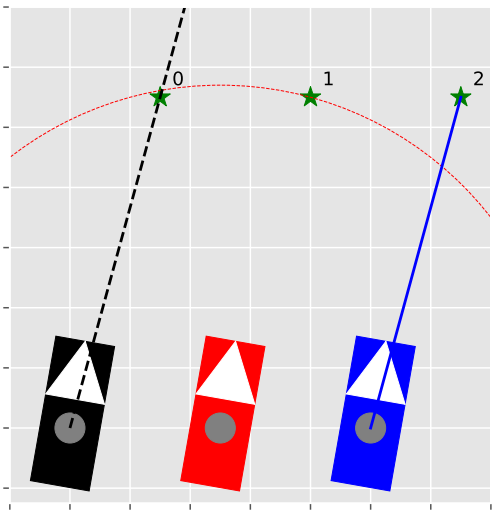


Abbildung 3.4:
Direkte Gegenüberstellung der drei Landmarken-Typen: Richtungsmerkmal (ID=0), Distanzlandmarke (ID=1) und die Positionslandmarke (ID=2).

Es wird zwischen Richtungsmerkmalen, Distanzlandmarken und Positionslandmarken unterschieden. In Abbildung 3.4 sind die verschiedenen Arten grafisch dargestellt.

Richtungsmerkmale sind typischerweise visuelle Merkmale, die aus Kamerabildern extrahiert werden. In den letzten Jahren sind visuelle Merkmale für SLAM populär geworden [Engel et al., 2014, Forster et al., 2014, 2017b, Gálvez-López und Tardós, 2012, Gao et al., 2018, Milford und Wyeth, 2012, Mur-Artal et al., 2015, Se et al., 2001, Sünderhauf et al., 2015, Zhou et al., 2014]. Bei diesem Messverfahren ist nur die Richtung des Merkmals in Bezug auf das Kamera-Koordinatensystem bekannt und nicht die Distanz. In der Computer Vision Literatur gibt es eine Vielzahl von Detektoren und Deskriptoren, die stabile Merkmale aus Bildern extrahieren [Agrawal et al., 2008, Bay et al., 2006, Lowe, 1999, Rosten und Drummond, 2005].

Distanzlandmarken werden auf Basis von Laufzeitmessungen erzeugt, die durch sogenanntes Pseudoranging zu beispielsweise GPS-Satelliten ermittelt werden. Darüber hinaus gibt es auch funkbasierte Verfahren wie WiFi oder Bluetooth, die über die Signalstärke und Laufzeit die Distanz ermitteln [Amini et al., 2014, Biswas und Veloso, 2010, Ferris et al., 2007, Parker und Valaee, 2006]. Um die Eigenposition über Triangulation bestimmen zu können, müssen jedoch mindestens drei Satelliten oder drei Basisstationen erreichbar sein. Zusätzlich ist eine genaue Uhr im GPS-Empfänger nötig, um die Laufzeit korrekt bestimmen zu können. Ist dies nicht der Fall, wird ein vierter Satellit benötigt, um eine möglichst genaue Position bestimmen zu können.

Positionslandmarken gehören zu den stabilsten Landmarkentypen, da die relative Position gut beobachtbar ist. Beispiele sind neben Kantenmerkmalen in LiDAR-Punktwolken auch 3D-Objekte, die mittels Cluster-Verfahren oder Lernverfahren aus der Punktwolke extrahiert werden können [Burger et al., 2018, Burger und Wuensche,

2018, Naujoks et al., 2019a]. Aber auch visuelle Landmarken aus Stereokamerabildern gehören zu dieser Gruppe [Lemaire et al., 2007, Wu et al., 2008].

Landmarken, welche sich in der Bewegungsrichtung des Fahrzeugs befinden, stellen eine besondere Herausforderung dar. Dies liegt daran, dass die Positionsunsicherheit des Fahrzeugs in der Längsrichtung höher ist, jedoch liefern sie wertvolle Informationen über die eigene Orientierung.

Ist nur die Richtung oder Entfernung einer Landmarke beobachtbar, ist die Position nicht sofort eindeutig und exakt bestimmbar. Das Gleiche gilt für Landmarken mit einer hohen Positionsunsicherheit. Die Landmarken müssen dann über die Zeit von verschiedenen Positionen beobachtet werden, damit eine eindeutige Positionsbestimmung z. B. über Triangulation möglich ist. Das bedeutet auch, dass eine Landmarke nicht automatisch in die Karte aufgenommen werden kann, wenn sie zum ersten Mal beobachtet wird. Ein Algorithmus muss in diesen Fällen entscheiden, ob und wann die Kartierung der Landmarke sinnvoll ist.

3.1.1.3 Topologische Karten

Topologische Karten bieten eine prägnante Darstellung der Welt, indem sie nur Informationen über relevante Orte und deren Beziehungen enthalten. Ein Musterbeispiel sind beispielsweise die Linienpläne eines U-Bahnnetzes. Alle Bahnhöfe sind hier über ein Streckennetz (Topologie) miteinander verbunden und der Fahrgast kann, ohne die genauen Abstände zu kennen, zu jedem Bahnhof navigieren.

Auch in der mobilen Robotik und im Bereich des autonomen Fahrens werden topologische Karten vermehrt zur autonomen Navigation sowie zur Lokalisierung eingesetzt [Blochliger et al., 2018, Cummins und Newman, 2008, Fassbender et al., 2016, 2015, 2014, Fraundorfer et al., 2007, Frese et al., 2005, Krajnik et al., 2014, Ort et al., 2018]. Die Verwendung einer topologischen Karte stellt dabei eine robuste und effiziente Kartenrepräsentation speziell für großflächige Außenszenen in unstrukturierter Umgebung dar, in der keine genauen Karteninformationen vorhanden sind, obgleich die Anforderungen an die Wahrnehmung und Interpretation der aktuellen Umgebung deutlich höher sind.

3.1.2 Lokalisierung

Zur Zeit von Christoph Kolumbus, zum Ende des 15. Jahrhundert, wurde die Himmelsnavigation gerade erst von den Portugiesen entwickelt. Zuvor navigierten die meisten Seefahrer unter Verwendung des Koppelnavigationsprinzips. Bei der Koppelnavigation findet der Seefahrer seine Position, indem er den Kurs und die Entfernung misst, die er von einem bekannten Punkt aus zurückgelegt hat.

Ausgehend von diesem bekannten Punkt, z. B. einem Hafen, misst er seinen Kurs sowie die Entfernung von diesem Punkt auf einer Karte und sticht mit einer Nadel in die Karte, um die neue Position zu markieren. Die Endposition eines jeden Tages

ist der Ausgangspunkt für die Kurs- und Distanzmessung am nächsten Tag. Die sogenannte Odometrie ist eine grundlegende Methode, die auch von autonomen Fahrzeugen bei der Navigation verwendet wird. Die Lageschätzung erfolgt dabei anhand Daten des Vortriebsystems. Legt das Fahrzeug längere Strecken zurück, wird jedoch die Unsicherheit über die eigene Position größer. Vergleichbar ist diese Situation, wenn sich ein Mensch mit geschlossenen Augen in einem Raum bewegt und nur die Schritte gezählt werden. Auch hier gilt: je weiter man geht, desto unsicherer ist man über den aktuellen Standort. Man muss somit von Zeit zu Zeit die Augen öffnen, um das eigene Verständnis über den Standort zu korrigieren. Im Allgemeinen gilt, dass Odometrie-basierte Lösungen ausreichende Genauigkeit für kurze Strecken liefern. Bei größeren Distanzen erhöht sich jedoch die Unsicherheit, und die Position muss mit Messungen (externe Information) von z. B. Landmarken korrigiert werden. Dieser Prozess wird als Lokalisierung bezeichnet.

Die heutzutage gängigste und am weitesten verbreitete Möglichkeit der Lokalisierung ist mittels eines GNSS. In vielen Einsatzorten kann jedoch nicht auf GNSS oder Karten zur Lokalisierung zurückgegriffen werden. Neben der gezielten Störung von Satellitensignalen gibt es sowohl in urbanen wie auch in unstrukturierten Gebieten Abschattung und Multipfadeffekte, welche die GNSS-basierte Lokalisierung stören [Henkel und Sperl, 2016, Mekik und Can, 2010]. Bis heute gelten Zentimeter-genaue Lokalisierungssysteme mittels GNSS als nicht ausreichend robust oder sehr teuer.

3.1.2.1 Globale und lokale Lokalisierung

Bei der Lokalisierung lässt sich zwischen einem lokalen und globalen Lokalisierungsproblem unterscheiden. Bei der globalen Lokalisierung existieren keine Vorkenntnisse über die Startposition. Die Startposition kann dabei überall in der Karte sein und muss mithilfe der Karte sowie verfügbaren Sensorinformationen erst ermittelt werden. In der Regel kann eine Re-Lokalisierung nicht sofort durchgeführt werden, insbesondere in Umgebungen mit vielen ähnlichen und sich wiederholenden Strukturen. Diese Art von Problem wird auch als *Kidnapped-Robot-Problem* bezeichnet. Zur Erreichung einer stabilen Anfangsposition müssen dazu oftmals mehrere Hypothesen aufrechterhalten werden.

In heutigen Anwendungen für autonome Fahrzeuge tritt das *Kidnapped-Robot-Problem* in der Regel nicht auf. Bei der lokalen Lokalisierung ist die anfängliche Position annähernd bekannt, z. B. durch ein GNSS. Odometriefehler werden während der Navigation kompensiert. Häufig ist die Pose bis zu einer kleinen Verschiebung und einem kleinen Fehler bereits bekannt. Von einem Verlust der aktuellen Positionsschätzung können sich solche Systeme normalerweise nur schwer erholen.

3.1.2.2 Ortsbestimmung

Eine spezielle Variante zum Lösen des Lokalisierungsproblems ist die sogenannte Ortsbestimmung (engl. place recognition). Ortserkennung bedeutet, dass der Roboter

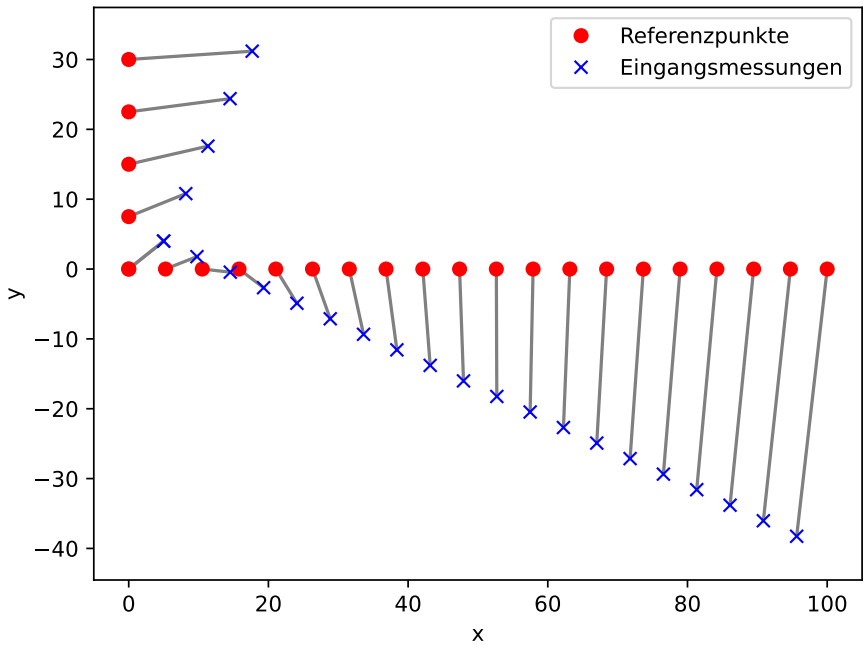


Abbildung 3.5:
ICP-Scan-Matching Verfahren einer 2D-Punktmenge. Beim Scan-Matching wird die Korrespondenz und Transformation zwischen den Eingangsmessungen (blaue Kreuze) sowie den Referenzpunkten (rote Punkte) auf Basis der gefundenen Korrespondenzen (graue Linien) bestimmt.

einen Ort aus einer bekannten Menge von Orten identifizieren muss. Im Folgenden werden drei verschiedene Varianten vorgestellt, die sich neben der Methodik auch in der Ausführung zwischen Single-Frame und Multi-Frame unterscheiden.

Scan-Matching-Techniken werden häufig angewendet, wenn der Roboter mit einem Laser-Entfernungsmesser ausgestattet ist [Holz und Behnke, 2010, Li et al., 2012, Moosmann und Stiller, 2011, Segal et al., 2010, Takeuchi und Tsubouchi, 2006, Ulas und Temeltas, 2011, 2013]. Der Abgleich von LiDAR-Scans ist ein grundlegender Bestandteil vieler Lokalisierungs- und Kartierungsalgorithmen. Die meisten Scan-Matching-Algorithmen erfordern das Finden von Korrespondenzen zwischen Merkmalen wie Punkten oder Linien. Beim Scan-Matching wird mittels eines Iterative Closest Point (ICP)-Verfahrens die Transformation zwischen zwei Punktwolken bestimmt und ein Fehlermaß berechnet [Segal et al., 2010]. In Abbildung 3.5 ist ein Beispiel dargestellt.

Eine weitere bekannte Methode ist die Normal Distributions Transform (NDT). Ähnlich zu OGM, wird das Umfeld in einem 2D-Raster unterteilt. Jeder Zelle wird mit einer Normalverteilung modelliert, welche die lokale Messwahrscheinlichkeit eines Punktes repräsentiert [Biber, 2003, Takeuchi und Tsubouchi, 2006, Ulas und Temeltas, 2011, 2013]. Das Ergebnis der Normal Distributions Transform (NDT) ist eine stückweise kontinuierliche und differenzierbare Wahrscheinlichkeitsdichte. Mithilfe des Newtonverfahrens kann die Transformation zwischen zwei LiDAR-Scans

bestimmt werden. Der große Vorteil dabei ist, dass keine expliziten Korrespondenzen hergestellt werden müssen.

Bei der **Visuellen Ortsbestimmung** wird das visuelle Erscheinungsbild der Umgebung mithilfe sogenannter Vokabelsätze (engl. Bag Of Words (BOW)) beschreiben [Yang et al., 2007]. Bei BOW wird die Anzahl jedes Wortes gezählt, das in einem Dokument vorkommt. Somit lässt sich beispielsweise über die Häufigkeit jedes Wortes ein signifikanter Schlüssel erstellen.

Bei Bag Of Visual Words (BOVW) hingegen werden aus Bildern sogenannte Schlüsselpunkte (engl. keypoints) extrahiert, Deskriptoren der umgebenen Pixel bestimmt und mit einem K-Means oder Density-Based Spatial Clustering of Applications with Noise (DBSCAN) Clusteringverfahren zusammengefasst [Agrawal et al., 2008, Bay et al., 2006, Jain, 2010, Jaspers, 2021, Lowe, 1999, Rosten und Drummond, 2005]. Die Zentren der Cluster werden als sogenannte visuelle Wörter geführt und bilden die Datenbank, die auch als Wörterbuch oder visuelles Vokabular bezeichnet wird. Um nun eine Ortsbestimmung durchzuführen, werden Schlüsselpunkte und Deskriptoren der aktuellen Szene bestimmt und mithilfe der Datenbank abgeglichen, indem die ähnlichsten Szenen anhand statistischer Maße gefunden werden [Gálvez-López und Tardós, 2012, Gao et al., 2018, Kim und Eustice, 2013].

Ein bekanntes Problem bei visuellen Lokalisierungsmethoden sind die sogenannten Falsch-Positive. Diese treten dann auf, wenn zwei Orte sehr ähnlich aussehen, sich diese aber tatsächlich an zwei verschiedenen Positionen befinden [Cummins und Newman, 2008, Milford et al., 2015, Milford und Wyeth, 2012, Warren et al., 2014]. Besonders in vom Menschen erbauten Umgebungen, die sich oft ähnlich sind und wiederholen, sind falsche Ortserkennungen sehr wahrscheinlich.

3.1.2.3 Filter-basierte (rekursive) Lokalisierung

Neben den klassischen Single-Frame Verfahren ist die rekursive Positionsschätzung eine weit verbreitete Methode in der Robotik. Für das Lösen der lokalen Lokalisierung werden oftmals Kalman Filter (KF) verwendet [Barfoot, 2017, Chen, 2012, Martinelli et al., 2007, Montemerlo et al., 2002, Roumeliotis und Bekey, 2000, Sola, 2013, Thrun et al., 2004].

Für die globale Lokalisierung eignen sich Partikel-Filter (PF) deutlich besser, da es mit ihnen möglich ist, eine multimodale Verteilung zu approximieren [Aulinas et al., 2008, Blanco et al., 2008, Burger et al., 2019a, Chen, 2012, Fairfield et al., 2007, Montemerlo und Thrun, 2003, Montemerlo et al., 2003, Rormero et al., 2018, Stachniss et al., 2005, 2016, Wu et al., 2008].

Eines der ersten Verfahren auf Basis einer Monte Carlo Lokalisierung (MCL) wurde von Dellaert et al. [1999] für ein autonomes System vorgestellt. Bei der MCL wird die Wahrscheinlichkeitsdichte durch eine Menge von Stichproben (Partikel) bestimmt, die zufällig gezogen werden. Um einen Roboter innerhalb einer Gridmap zu lokalisieren, werden Distanzmessungen von Sonar- und Laserscannern verwendet.

3.1.2.4 Monte Carlo Lokalisierung

Die Monte Carlo Lokalisierung (MCL) ist eine Partikel-Filter (PF) basierte Lokalisierungsmethode, um den Belief über die Fahrzeugposition $\boldsymbol{x}_k$ zum Zeitpunkt k zu bestimmen:

$$\text{bel}_{\boldsymbol{M}}\left(\boldsymbol{x}_k\right) = p\left(\boldsymbol{x}_k \mid \boldsymbol{y}_{1:k}, \mathbf{u}_{0:k-1}, \boldsymbol{M}\right), \tag{3.1}$$

auf Basis der Messungen $\mathbf{y}_{1:k}$, den Aktionen $\mathbf{u}_{0:k-1}$ und einer Karte $\boldsymbol{M}$.

Mit der erweiterten Bayes'schen Filtergleichungen aus Unterabschnitt 2.5.2 kann die posteriore Wahrscheinlichkeitsverteilung wie folgt rekursiv geschätzt werden:

$$\overline{\text{bel}}_{\boldsymbol{M}}\left(\boldsymbol{x}_k\right) = \int p\left(\boldsymbol{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_{k-1}\right) \text{bel}_{\boldsymbol{M}}\left(\mathbf{x}_{k-1}\right) \mathrm{d}\mathbf{x}_{\mathbf{k-1}}, \tag{3.2}$$

$$\text{bel}_{\boldsymbol{M}}\left(\boldsymbol{x}_k\right) = \eta \cdot p\left(\boldsymbol{y}_t \mid \boldsymbol{x}_k, \boldsymbol{M}\right) \overline{\text{bel}}_{\boldsymbol{M}}\left(\boldsymbol{x}_k\right). \tag{3.3}$$

In Abbildung 3.6 sind die verschiedenen Filterschritte (Initialisierung, Prädiktion, Update) grafisch dargestellt. Im Folgenden werden die einzelnen Schritte genauer beschrieben:

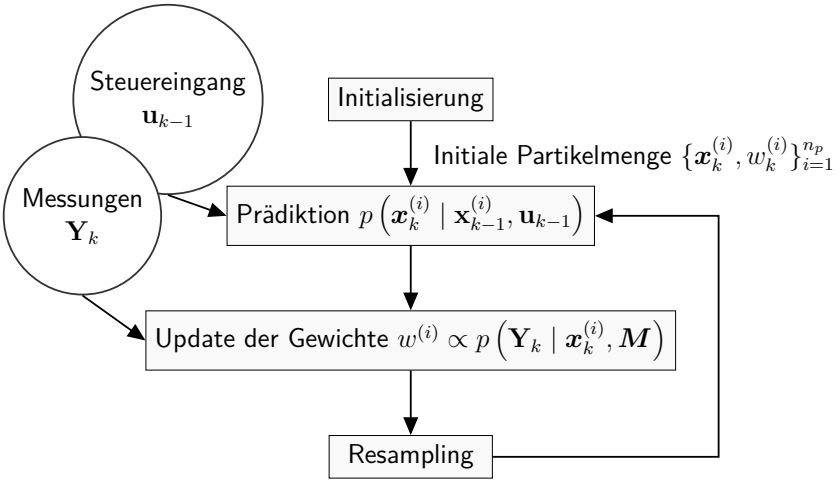


Abbildung 3.6: Darstellung der verschiedenen Verarbeitungsschritte eines PF.

Initialisierung Bei der Initialisierung wird eine Menge von Partikeln erzeugt $\{\boldsymbol{x}_k^{(i)}, w_k^{(i)}\}_{i=1}^{n_p}$. Liegt keine a-priori Information über die Verteilung vor, so werden Stichproben üblicherweise aus einer Gleichverteilung des Zustandsraums (Karte) gezogen. Für den Fall, dass bereits eine erste Positionsschätzung vorliegt, wird oftmals eine Gauß-Verteilung verwendet, welche zusätzlich die Unsicherheit der gegebenen anfänglichen Positionsschätzung berücksichtigt, um die Partikel zu streuen.

Prädiktion Im Prädiktionsschritt werden die Partikel anhand eines Bewegungsmodells prädiziert, wobei die gegebenen Unsicherheiten durch einen additiven Rauschterm berücksichtigt werden, siehe Unterabschnitt 3.2.3. Die neuen Zustände der

Partikel ergeben sich somit anhand der Zustandsübergangswahrscheinlichkeit, siehe Unterabschnitt 2.5.1.1.

Updateschritt Im Updateschritt wird die vorhergesagte Wahrscheinlichkeitsdichtefunktion (WDF) mit den aktuellen Messungen korrigiert. Im Innovationsschritt werden die Messungen zur Bewertung der Partikelzustände und somit zur Berechnung der Partikelgewichte herangezogen, sodass gilt:

$$w_k^{(i)} \propto p\left(\mathbf{y}_k \mid \boldsymbol{x}_k^{(i)}, \boldsymbol{M}\right). \tag{3.4}$$

Eine typische Methode zur Aktualisierung des Partikelgewichts besteht darin, das Gewicht aus dem vorherigen Zeitschritt mit der neuen Messwahrscheinlichkeit zu multiplizieren:

$$w_k^{(i)} = w_{k-1}^{(i)}\, p\left(\mathbf{y}_k \mid \boldsymbol{x}_k^{(i)}, \boldsymbol{M}\right). \tag{3.5}$$

Die Wahl einer sinnvollen Messwahrscheinlichkeitsfunktion $p\left(\boldsymbol{Y}_k \mid \boldsymbol{x}_k, \boldsymbol{M}\right)$ ist dabei der entscheidende Faktor für eine funktionierende MCL. Unter Berücksichtigung der verwendeten Sensoren und Karten sind dabei die verschiedensten Implementierungen möglich, siehe [Adiprawita et al., 2011, Burger et al., 2019a, Floros et al., 2013, Fox et al., 1999, Marchetti et al., 2007, Schiotka et al., 2017].

Resampling Um eine Degeneration des Filters zu vermeiden, wird ein sogenanntes Resampling durchgeführt, siehe [Adiprawita et al., 2011, Burger et al., 2019a, Marchetti et al., 2007, Senlet und Elgammal, 2011]. Die Degeneration tritt auf, wenn viele Partikel nur noch ein sehr geringes Gewicht besitzen. Daher werden beim Resampling Partikel mit einer Wahrscheinlichkeit gezogen, die ihrem Gewicht entsprechen. Das Resampling des Partikelfilters funktioniert nach dem *survival of the fittest*-Prinzip, bei dem eine neue Generation an Partikeln anhand der stärksten Partikel der vorherigen Generation für den nächsten Zeitschritt erzeugt werden.

Um festzustellen, ob ein Resampling durchgeführt werden sollte, wird in der Regel die effektive Anzahl der Partikel bestimmt:

$$n_{eff} = {}^{1}\!/\!{\sum_i \left(w^{(i)}\right)^2}, \tag{3.6}$$

und mit einem Schwellwert n_{thr} verglichen. Gilt $n_{eff} < n_{thr}$, so wird eine definierte Anzahl von Partikeln aus der aktuellen Partikelmenge gezogen, die proportional zu ihrem Gewicht sind.

Zusammengefasst sind die Vorteile des PF Ansatzes:

- kann mit Nichtlinearitäten umgehen,
- kann mit nicht-gaußschen Verteilungen umgehen,
- größtenteils parallelisierbar,
- einfach zu implementieren,
- konzentrieren sich im Gegensatz zu, Hidden Markov Model (HMM)-Filtern, adaptiv auf wahrscheinlichere Regionen des Zustandsraums.

In den letzten Jahrzehnten hat das PF daher kontinuierlich große Aufmerksamkeit in der Wissenschaft erhalten [Aulinas et al., 2008, Blanco et al., 2008, Carlone et al., 2010, 2014b, Manz et al., 2011, Marchetti et al., 2007, Montemerlo und Thrun, 2003, Montemerlo et al., 2003, Nuss et al., 2018, Petrovskaya und Thrun, 2009, Rormero et al., 2018, Särkkä, 2010, Stachniss et al., 2005, 2016, Steyer et al., 2018, Vallivaara et al., 2010, Weikersdorfer und Conradt, 2012, Wu et al., 2008].

3.1.2.5 Rao-Blackwellized Monte Carlo Lokalisierung

Die von Murphy und Russell [2001] entwickelte Rao-Blackwellized Monte Carlo Lokalisierung (RBMCL) ist eine Erweiterung der MCL. Die RBMCL verbindet die Vorteile das EKF zur modellbasierten Prädiktion und die Möglichkeit des PF eine multimodale Verteilung zu approximieren. Im Vergleich zur MCL werden die Partikelpositionen mithilfe eines EKF prädiziert, wodurch die Partikelposition schon vor dem Update verbessert und somit deren Varianz verringert wird. Jedes Partikel ist somit nochmals ein eigenständiger KF.

3.2 Simultane Lokalisierung und Kartierung (SLAM)

SLAM ist seit über drei Jahrzehnten ein sehr aktives Forschungsthema im Bereich der mobilen Robotik und des autonomen Fahrens [Alismail et al., 2014, Burger et al., 2019b, Engel et al., 2014, Forster et al., 2017b, Gao et al., 2018, Guivant und Nebot, 2001, Kerl et al., 2013, Kim et al., 2016b, Kümmerle et al., 2010, Lemaire et al., 2007, Makarenko et al., 2002, Montemerlo und Thrun, 2003, Montemerlo et al., 2002, 2003, Moosmann und Stiller, 2011, Mur-Artal et al., 2015, Thrun et al., 2005, Valencia und Andrade-Cetto, 2018, Wu et al., 2008]. SLAM-Verfahren verbinden dabei oftmals Ideen und Techniken aus verschiedenen Forschungsbereichen: von der Bayes'schen Wahrscheinlichkeitstheorie, dem Computersehen bis hin zu Neurowissenschaften.

SLAM-Verfahren können dabei in mehrere Klassen unterteilt werden. Die Unterscheidung ist abhängig von verschiedenen Kriterien, z. B. der Art der Karte, der Art der Sensoren sowie der Art der Methode, welche zum Lösen des SLAM-Problems verwendet wird. Nachdem in den vorherigen Abschnitten Verfahren zur Lokalisierung

und Kartierung vorgestellt worden sind, werden nun Konzepte für SLAM und die dabei entstehenden Herausforderungen dargestellt.

3.2.1 Frontend & Backend

SLAM-Verfahren lassen sich nach Sünderhauf und Protzel [2012] in zwei Funktionsblöcke, das Frontend und das Backend, unterteilen. Das Frontend beinhaltet die Verarbeitung der Sensordaten und führt die Datenassoziation der Landmarken mit der Karte durch. Zusätzlich hat das Frontend die Aufgabe, eine Schleifenschließung zu erkennen. Der zweite Teil eines SLAM-Systems wird als Backend bezeichnet. Das Backend hat die Aufgabe auf Basis der assoziierten Sensordaten die a-posteriori Wahrscheinlichkeitsdichte zu bestimmen. Abbildung 3.7 veranschaulicht das Konzept und den Informationsfluss.

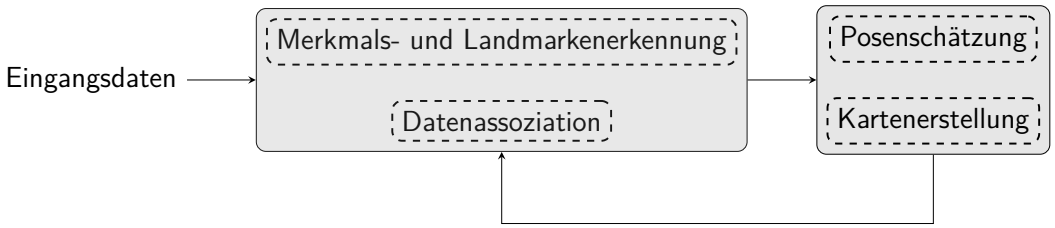


Abbildung 3.7: Frontend & Backend eines SLAM-Frameworks.

In den Anfängen von SLAM dominierten filterbasierte Methoden wie das FastSLAM Verfahren von Montemerlo et al. [2003]. Im letzten Jahrzehnt fand jedoch ein Wandel in der SLAM-Community hin zur Verwendung von modernen Optimierungsverfahren statt. Diese Ansätze modellieren das SLAM-Problem mithilfe einer Graphenstruktur und bauen auf effizienten Algorithmen zur nichtlinearen Optimierung der kleinsten Quadrate auf. Die rasante Entwicklung zu immer effizienteren Prozessoren beschleunigen die Verbreitung zusätzlich, da selbst große SLAM-Probleme in nur wenigen Sekunden auf Standard-Hardware gelöst werden können.

Je nach SLAM-Verfahren unterscheidet sich der Aufbau des Frontends und des Backends. Bei Filter-basierten Verfahren wird die Karte und die Fahrzeugposition durch einen rekursiven Schätzprozess im Backend bestimmt. Beim Graph-SLAM sind Faktorgraphen eine mittlerweile weit verbreitete Methode zur Darstellung des Optimierungsproblems [Kümmerle et al., 2011]. Der Graph wird vom Frontend unter Verwendung aller verfügbaren Sensordaten und Datenassoziationsergebnisse erstellt und anschließend an das Backend übergeben. Das Backend wiederum extrahiert Informationen und löst das im Graphen beschriebene Optimierungsproblem, mit dem Ziel, eine Karte sowie die zurückgelegte Trajektorie zu rekonstruieren.

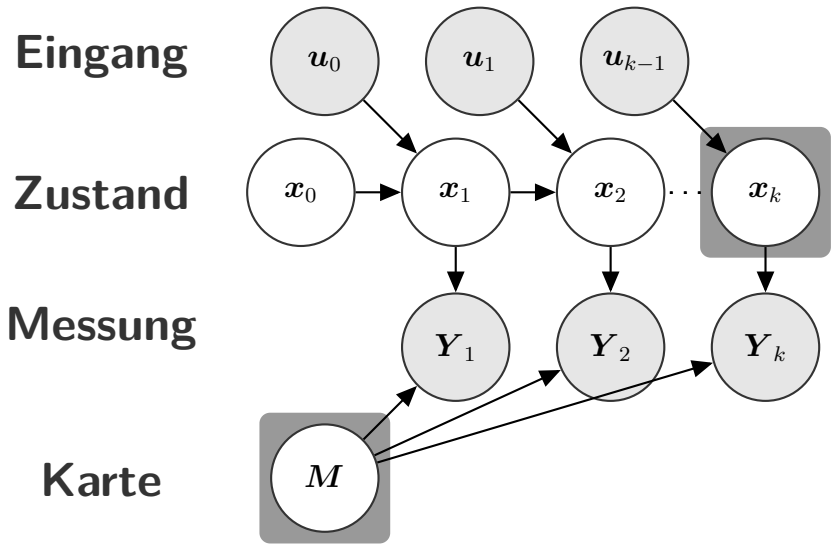


Abbildung 3.8:
Dynamisches Bayes'sche Netzwerk zur Repräsentation des SLAM-Problems. Die grauen Knoten repräsentieren die Beobachtungen (Sensormessungen), während weiße Knoten unbeobachtete Zufallsvariablen darstellen, z. B. die Roboterposition und die Landmarkenpositionen. Beim Online SLAM wird die aktuellste Position $\boldsymbol{x}_k$ sowie die Karte $\boldsymbol{M}$ gleichzeitig bestimmt, was hier durch die Umrandung mit einem dunkelfarbigen Quadrat dargestellt ist.

3.2.2 Überblick: Probabilistische Methoden

Das SLAM-Problem lässt sich mit Hilfe eines Dynamischen Bayes'schen Netzwerkes darstellen. In Abbildung 3.8 ist eine Dynamisches Bayes'sche Netzwerk (DBN)-Darstellung für das SLAM-Problem mit Landmarken für k Zeitschritte zu sehen [Dean und Kanazawa, 1988, Loeliger, 2004, Thrun et al., 2005].

Zustände und Beobachtungen werden dabei als Knoten in einem Graphen repräsentiert. Kanten hingegen repräsentieren zeitliche und bedingte Abhängigkeiten zwischen den Knoten. Darüber hinaus werden Beobachtungen als graue Knoten und unbeobachtete Zufallsvariablen als weiße Knoten dargestellt, ähnlich einem HMM.

Bei SLAM kann zwischen verschiedenen probabilistischen Schätzverfahren unterschieden werden. Alle Varianten haben gemeinsam, dass neben der Fahrzeugposition oder der vergangenen Trajektorie die Karte gleichzeitig geschätzt wird.

Im Folgenden werden diese Notationen verwendet:

- Menge an Fahrzeugposen $\boldsymbol{X}_K = \{\boldsymbol{x}_k\}_{k=1}^K$,
- Menge an Landmarkenmessungen $\boldsymbol{Y}_K = \{\boldsymbol{y}_k\}_{k=1}^K$,
- Menge an Odometriemessungen $\boldsymbol{U}_K = \{\boldsymbol{u}_{k-1}\}_{k=1}^K$,

mit Index k, wobei K die vollständige Menge aller Posen, Messungen und Aktionen kennzeichnet.

Full SLAM beschreibt den Prozess, bei dem die vollständige Karte $\boldsymbol{M}$ und die komplette Sequenz an Fahrzeugpositionen $\boldsymbol{X}_K$ gleichzeitig geschätzt werden. Diese Art, das SLAM-Problem zu lösen, gilt als sehr ressourcenaufwendig und wird daher auch als Offline-SLAM bezeichnet und primär für die Kartenerstellung verwendet. Die a-posteriori Wahrscheinlichkeitsverteilung unter Berücksichtigung aller Messungen $\boldsymbol{Y}_k$ und Aktion $\boldsymbol{U}_K$ ist wie folgt definiert und graphisch in Abbildung 3.9 als Dynamisches Bayes'sche Netzwerk (DBN) dargestellt:

$$p\left(\boldsymbol{X}_K, \boldsymbol{M} \mid \boldsymbol{Y}_k, \boldsymbol{U}_K\right). \tag{3.7}$$

Beim **Online SLAM** hingegen wird nur die aktuellste Fahrzeugposition $\boldsymbol{x}_k$ und die Karte $\boldsymbol{M}$ bestimmt:

$$\text{bel}\left(\boldsymbol{x}_k, \boldsymbol{M}\right) = p\left(\boldsymbol{x}_k, \boldsymbol{M} \mid \boldsymbol{Y}_k, \mathrm{U}_{k-1}\right), \tag{3.8}$$

unter Berücksichtigung der Messungen $\boldsymbol{Y}_k$ und den Aktionen U_{k-1}. Der Index k kennzeichnet die dynamische Menge an Zuständen und Messungen, welche zu jedem Zeitschritt k um eins erweitert werden.

Das Online-SLAM wird im Allgemein auch als Filterung bezeichnet, während Full-SLAM (offline) auch als Glättung (engl. smoothing) bezeichnet wird.

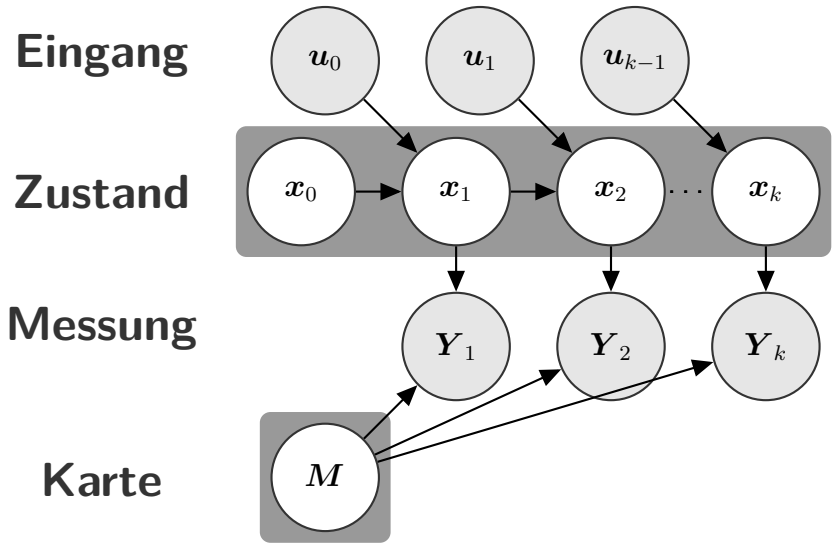


Abbildung 3.9:
Ein Dynamisches Bayes'sche Netzwerk (DBN) zur Darstellung eines Full SLAM Problems.

Aktuelle graphenbasierte **Hybrid SLAM**-Verfahren verbinden die Vorzüge von Online-SLAM und Full-SLAM. Sie stellen einen Hybriden-Ansatz dar und sind in der Lage, die aktuellste Position und gleichzeitig alle vergangenen Position bis zum aktuellen Zeitschritt k zu schätzen, sodass:

$$\text{bel}\left(\boldsymbol{X}_k, \boldsymbol{M}\right) = p\left(\boldsymbol{x}_k, \boldsymbol{M} \mid \boldsymbol{Y}_k, \mathrm{U}_{k-1}\right). \tag{3.9}$$

Zur Vollständigkeit wird an dieser Stelle erwähnt, dass im Allgemeinen bei SLAM-Methoden die Umgebung als statisch angesehen wird. Autonome Fahrzeuge bewegen sich jedoch oft in dynamischer Umgebung und es kann somit nicht davon ausgegangen werden, dass erkannte Merkmale oder Landmarken zu einem statischen Objekt gehören. Es müssen weitere Methoden wie ein Klassifizierungsverfahren oder auch filter-basierte Verfahren integriert werden, die aktiv dynamische Objekte aus der Menge von möglichen Merkmalen ausschließen. Dynamic SLAM ist dabei eine Erweiterung und beschreibt den SLAM-Prozess für dynamische Umgebungen [Xiang et al., 2015].

3.2.3 Bewegungsmodelle für SLAM

Die Fahrzeugposen $\boldsymbol{X}_K$ und die Karte $\boldsymbol{M}$ sind nicht a-priori bekannt. Sie müssen anhand der gegebenen Sensorinformationen geschätzt werden. Hierzu gehören beispielsweise Odometrieinformationen, welche die Bewegung zwischen einzelnen Posen beschreibt. Unter dem Begriff Odometrie sind nicht nur im klassischen Sinn Informationen gemeint, die von einfachen Rad-Enkodern geliefert werden, sondern auch Beschleunigungs- und Drehratenmessungen aus Inertialsensorik, oder Odometrieinformationen von komplexeren Ansätzen wie die der visuellen Odometrie oder Scan-Matching Verfahren [Bosse und Zlot, 2009, Holz und Behnke, 2010, Li et al., 2012, Lu und Milios, 1997b, Moosmann und Stiller, 2011, Pathak et al., 2009, Pomerleau et al., 2013, Ulas und Temeltas, 2011, Zhang und Singh, 2015].

Bei der visuellen Odometrie wird aus einer Abfolge von Bildern die relative Bewegung berechnet [Brubaker et al., 2016, Censi und Scaramuzza, 2014, Fraundorfer und Scaramuzza, 2012, Gao et al., 2018]. Beim Scan-Matching hingegen wird die Transformation (Bewegung) zwischen zwei Punktwolken, z. B. mithilfe eines ICP-Verfahrens, bestimmt.

Vektor $\boldsymbol{u}_{k-1}$ ist somit sehr stark vom Aufbau und dem tatsächlich verwendeten Sensorsystem abhängig. Neben einer im lokalen Koordinatensystem gemessenen Translation und Rotation $\boldsymbol{u}_{k-1} = (\Delta x, \Delta y, \Delta z, \Delta\phi, \Delta\theta, \Delta\psi)^T$, was beispielsweise durch ein Scan-Matching Verfahren bestimmt wird, misst eine Inertial Measurement Unit (IMU) Beschleunigungen und Drehraten, sodass $\boldsymbol{u}_{k-1} = \left(\ddot{x}, \ddot{y}, \ddot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi}\right)^T$.

Darüber hinaus liefert ein einfacher Rad-Enkoder die gezählten Raddurchläufe $\boldsymbol{u}_{k-1} = (c_{left}, c_{right})^T$ oder die Vorwärtsgeschwindigkeit und Drehrate $\boldsymbol{u}_{k-1} = (v_x, \omega)^T$. Aber auch Geschwindigkeit und Lenkwinkeländerungen aus Steuergeräten können verwendet werden $\boldsymbol{u}_{k-1} = \left(\dot{v_x}, \dot{\Psi}\right)^T$.

Die Beziehung zwischen zwei aufeinander folgenden Posen und der jeweiligen Odometriemessung wird normalerweise durch ein nichtlineares Bewegungsmodell beschrieben:

$$\boldsymbol{x}_{k+1} = f\left(\boldsymbol{x}_k, \boldsymbol{u}_{k-1}\right) + \boldsymbol{v}_{k-1}, \tag{3.10}$$

was dem Modell des nichtlinearen Zustandsübergangs aus Unterabschnitt 2.5.4 entspricht mit $\boldsymbol{v}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1})$ mit

$$\boldsymbol{x}_{k+1} \sim \mathcal{N}(f\left(\boldsymbol{x}_k, \boldsymbol{u}_{k-1}\right), \mathbf{Q}_{k-1}). \tag{3.11}$$

Je nach Fahrzeugkonstruktion können verschiedene Bewegungsmodelle verwendet werden [Bar-Shalom et al., 2002, Shimkin, 2009]. Ein oft verwendetes Modell in der mobilen Robotik ist das Constant Turn Rate and Velocity model (CTRV) von Schubert et al. [2008]. Für den Zustandsvektor und Steuereingang gilt $\boldsymbol{x}_{k-1} = (x, y, \psi)^T$ und $\boldsymbol{u}_{k-1} = (v, \omega)^T$:

$$\begin{aligned} \boldsymbol{x}_{k+1} =& f\left(\boldsymbol{x}_k, \boldsymbol{u}_{k-1}\right) \\ =& \boldsymbol{x}_k + \begin{cases} \begin{pmatrix} \frac{v_{k-1}}{\omega_{k-1}} \left(\sin\left(\psi_{k-1} + \omega_{k-1}\Delta t\right) - \sin\left(\psi_{k-1}\right)\right) \\ \frac{v_{k-1}}{\omega_{k-1}} \left(\cos\left(\psi_{k-1}\right) - \cos\left(\psi_{k-1} + \omega_{k-1}\Delta t\right)\right) \\ \omega_{k-1}\Delta t \end{pmatrix} & \omega_{k-1} \neq 0 \\ \begin{pmatrix} v_{k-1}\cos\left(\psi_{k-1}\right)\Delta t \\ v_{k-1}\sin\left(\psi_{k-1}\right)\Delta t \\ 0 \end{pmatrix} & \text{andernfalls.}, \end{cases} \end{aligned} \tag{3.12}$$

3.2.4 Filterbasierte SLAM-Verfahren

Eines der ersten Filter-basierten SLAM-Verfahren wurde von Smith et al. [1988] vorgestellt. Mithilfe eines EKF wurde aus verrauschten Sensormessungen die Fahrzeugposition und die Karte gleichzeitig bestimmt. Im Allgemeinen fordert der EKF-SLAM-Ansatz eine Linearisierung der Roboter- und Sensormodelle unter Annahme von Gauß'schem Rauschen. Diese vereinfachte Annahme kann jedoch problematisch sein, z. B. wenn die Verteilungen nicht unimodal ist oder große Unsicherheiten und starke Nichtlinearitäten beteiligt sind [Guivant und Nebot, 2001].

Die Datenassoziation für EKF-SLAM wird üblicherweise mit der Maximum-Likelihood-Methode gelöst. Jede Messung wird dabei einer Landmark $\boldsymbol{l}$ zugeordnet, die beispielsweise die kleinste Distanz aufweist und somit im Vergleich zu allen anderen die Likelihood-Funktion maximiert. Was im Falle einer von einem Parameter ϑ abhängigen Wahrscheinlichkeitsfunktion wie folgt definiert ist: $L\colon \Theta \to [0;1], \quad \vartheta \mapsto p\left(\boldsymbol{l} \mid \vartheta\right)$, gegeben der Landmarke $\boldsymbol{l}$ und dem möglichen Parameterraum Θ. Ist die Wahrscheinlichkeit einer Assoziation jedoch zu gering, wird die Messung als neue Landmarke betrachtet.

Darüber hinaus hat das EKF keinen Mechanismus, um die Unsicherheit der Datenassoziationen nativ zu berücksichtigen. Daher divergiert das Filter, wenn eine große Anzahl von Messungen den falschen Zuständen (Landmarken) zugeordnet wird. Ein weiteres bekanntes Problem von EKF-SLAM ist der Berechnungsaufwand, der insbesondere bei einer großen und immer weiter steigenden Anzahl von Landmarken auftritt.

Darauf folgend wurde der alternative Ansatz FastSLAM von Montemerlo et al. [2002] vorgestellt, der eine Echtzeit-Implementierung des SLAM-Problems aus Bayes'scher Sicht beschreibt. Die entwickelte Methode unterteilt, im Vergleich zum vorherigen EKF Ansatz, SLAM in ein Lokalisierungs- und ein Mapping-Problem. Die Idee dahinter ist, dass durch die Annahme von bekannten Posen das Mapping keine Herausforderung mehr darstellt. Die Position wird durch einen PF geschätzt, wobei jedes Partikel für jede Landmarke ein unabhängiges EKF besitzt und jedes Partikel eine mögliche Fahrzeugpose und die dazugehörigen Landmarken repräsentiert.

Die Prädiktion der Partikelposition erfolgt klassisch wie beim PF durch ein Bewegungsmodell, während im Updateschritt zur Bestimmung des Partikelgewichtes die Wahrscheinlichkeit über alle Messungen zu den Landmarken sowie deren Messunsicherheit berücksichtigt wird, siehe Unterabschnitt 2.5.6.

FastSLAM 2.0 von Montemerlo et al. [2003] ist eine Erweiterung, wobei die Posen unter Berücksichtigung der Eigenbewegung und den Messungen gesampelt werden. Dies hat zur Folge, dass weniger Partikel benötigt und ein robusteres und genaueres Ergebnis erzielt werden kann.

3.2.5 Graphenrepräsentation zum Lösen des SLAM-Problems

Beim graphenbasierten SLAM wird das SLAM-Problem durch eine graphische Darstellung modelliert und ausgedrückt [Dean und Kanazawa, 1988, Kaess et al., 2012, Kümmerle et al., 2011, Thrun und Montemerlo, 2006].

Levinson et al. [2008] erstellt beispielsweise eine hochauflösende Umgebungskarte in städtischer Umgebung und verwendet dabei GNSS-, IMU-, Rad-Odometer- und LiDAR als Eingangsdaten. Mittels Graph-SLAM wird eine global, hochgenaue und konsistente Karte über mehrere Kartierungsläufe erstellt. Der LiDAR-Sensor erfasst dabei Entfernungsdaten und Reflexionsstärken der Bodenfläche. Infolgedessen werden bewegliche Objekte, wie statische und dynamische Fahrzeuge, herausgefiltert. Die erstellte Karte besteht daher nur aus 3D-Infrarotbildern der Bodenreflektivität. Fahrbahnmarkierungen sind dabei besonders markante Merkmale, da sie ein viel höheres Reflexionsvermögen als die Straßenbeläge besitzen.

Die globale Lokalisierung in der zuvor erstellten Karte erfolgt jedoch mittels MCL, bei der das Partikelgewicht abhängig ist von der Übereinstimmung der wahrgenommenen und gespeicherten Gridzellenreflektivität.

3.2.5.1 Faktorgraphen

Faktorgraphen sind ungerichtete Graphen, die zwei Arten von Knoten enthalten: einen für Variablen und einen für die Beziehungen (Faktoren) zwischen den Variablen. Sie wurden als ein allgemeines Werkzeug zur Faktorisierung großer Funktionen von Dean und Kanazawa [1988] entwickelt und können ebenfalls auf probabilistische Probleme wie SLAM angewandt werden.

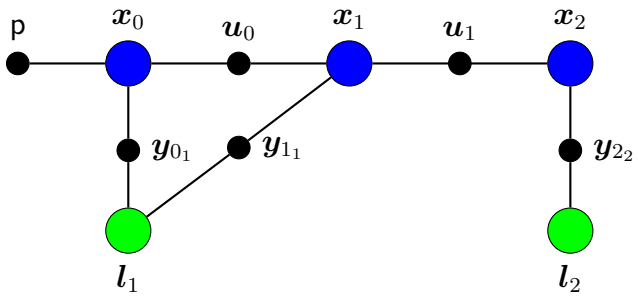


Abbildung 3.10:
Ein HMM eines Full SLAM-Problems mit Landmarkenmessungen dargestellt als Faktorgraph. Die blauen und grünen Knoten repräsentieren unbekannte Fahrzeug- und Landmarkenpositionen, während probabilistische Beziehungen zwischen ihnen durch schwarze Knoten dargestellt werden. Schwarze Knoten entsprechen Aktionen $\boldsymbol{u}_k$ und Landmarkenbeobachtungen $\boldsymbol{y}_{k_l}$.

In Abbildung 3.10 ist die Faktorgraphenrepräsentation für das Landmarken-SLAM Problem dargestellt. Farbige Knoten repräsentieren die unbekannten Variablen (Roboterzustände, Landmarkenpositionen oder Kartenzustand im Allgemeinen), während die schwarzen Knoten die bedingten Wahrscheinlichkeiten (Beziehungen) zwischen ihnen abbilden.

Das gemeinsame Wahrscheinlichkeitsmodell des Landmarken-SLAM-Problems ist nach Sünderhauf [2012] wie folgt definiert:

$$p\left(\boldsymbol{X}_K, \boldsymbol{M} \mid \boldsymbol{U}_K, \boldsymbol{Y}_K\right) = p\left(\boldsymbol{x}_0\right) \prod_k p\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}\right) \prod_{l,k} p\left(\boldsymbol{l}_j \mid \boldsymbol{x}_k, \boldsymbol{y}_{k_l}\right), \quad (3.13)$$

mit Prior $p\left(\boldsymbol{x}_0\right)$, Landmarkenmodell $p\left(\boldsymbol{l}_l \mid \boldsymbol{x}_k, \boldsymbol{y}_{k_l}\right)$ mit l-ter Landmarke $\boldsymbol{l}_l$ die zur Messung $\boldsymbol{y}_{k_l}$ gehört und die Karte $\boldsymbol{M}$ bildet, wobei $l \in 1, \ldots, L$ Landmarken.

Im Folgenden wird nun das SLAM-Problem als Optimierungsproblem der kleinsten Quadrate formuliert und exemplarisch für ein landmarkenbasiertes SLAM-Verfahren mithilfe einer Faktorgraphendarstellung gelöst [Choudhary et al., 2015]. Die nichtlineare Optimierung von Bayes'schen Netzwerken, verkörpert durch Faktorgraphen, ist eine allgemeine Technik, um die Maximum-A-Posteriori-Schätzung für eine Menge gegebener Beobachtungen zu finden.

3.2.5.2 Maximum-Likelihood SLAM

Beim graphenbasierten Maximum-Likelihood SLAM wird die Schätzung der A-posteriori-Wahrscheinlichkeit durch einen Graphen $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ beschrieben. Positionen $\mathbf{x}_{1:k}$ und die Landmarken $\mathbf{l}_{1:L}$ werden durch Knoten des Graphen $\mathbf{v}_i \in \mathbf{V} \mid i \in \mathcal{N}$ repräsentiert. Die Kanten $\mathbf{E}$ (engl. Edges) zwischen den Knoten beschreiben dabei Messungen wie beispielsweise Odometrie- oder Landmarkenmessungen.

In den meisten Fällen ist die Anzahl von Kanten größer als die Anzahl an Knoten. Durch Fehler im Messverfahren sind die Beziehungen zwischen den Knoten widersprüchlich. Das Ziel des Optimierungsschritts im graphbasierten SLAM besteht darin, eine Konfiguration der Knoten zu finden, die die Konstellation aller Knoten auf Basis der Beobachtungen maximal konsistent abbildet. Dies beinhaltet das Lösen eines Fehlerminimierungsproblems mit einer großen Anzahl von Unbekannten.

In der Faktorgraphendarstellung der Wahrscheinlichkeitsverteilung $p\left(\boldsymbol{X}, \boldsymbol{M}, \boldsymbol{U}, \boldsymbol{Y}\right)$ werden Faktoren durch $p\left(\boldsymbol{x}_0\right)$, $p\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}\right)$ und $p\left(\boldsymbol{y}_k \mid \boldsymbol{x}_{ki}, \boldsymbol{l}_{ji}\right)$ repräsentiert. Die Faktoren lassen sich auf Basis von sogenannten Fehlerfunktionen modellieren und die gemeinsame Wahrscheinlichkeitsverteilung kann wie folgt definiert werden:

$$p\left(\boldsymbol{X}, \boldsymbol{M}, \boldsymbol{U}, \boldsymbol{Y}\right) \propto \boldsymbol{F}(\mathbf{V}) = \sum_{\langle i,j \rangle \in C} \boldsymbol{e}\left(\mathbf{y}_{ij}, \mathbf{v}_i, \mathbf{v}_j\right)^T \boldsymbol{\Omega}_{ij} \boldsymbol{e}\left(\mathbf{y}_{ij}, \mathbf{v}_i, \mathbf{v}_j\right), \tag{3.14}$$

mit Fehlerfunktion $\boldsymbol{e}\left(\mathbf{y}_{ij}, \mathbf{v}_i, \mathbf{v}_j\right) = \mathbf{y}_{ij} - h_{ij}(\mathbf{v}_i, \mathbf{v}_j)$, welche die Differenz zwischen der erhaltenen Messung $\mathbf{y}_{ij}$ und der erwarteten Messung $h_{ij}(\mathbf{v}_i, \mathbf{v}_j)$ beschreibt und die durch die Informationsmatrix $\boldsymbol{\Omega}_{ij}$ gewichtet wird. Die Informationsmatrix stellt beispielsweise die Unsicherheit des Messfehlers zwischen der Pose i und der Landmarke j dar und ist der Kehrwert der Messkovarianzmatrix.

Zur Vereinfachung der Schreibweise wird im Folgenden die Schreibweise der Fehlerfunktion vereinfacht:

$$\boldsymbol{e}(\mathbf{y}_{ij}, \mathbf{v}_i, \mathbf{v}_j) \overset{\text{def.}}{=} \boldsymbol{e}_{ij}(\mathbf{v}_i, \mathbf{v}_j) \overset{\text{def.}}{=} \boldsymbol{e}_{ij}(\mathbf{v}). \tag{3.15}$$

Auf Basis der Messungen und Aktionen lässt sich dann die Maximum-A-Posteriori-Schätzung durch die Maximierung der gemeinsamen Wahrscheinlichkeitsverteilung $p\left(\boldsymbol{X}, \boldsymbol{M}, \boldsymbol{U}, \boldsymbol{Y}\right)$ bestimmen:

$$\mathbf{V}^* = \arg\max_{\mathbf{V}} p\left(\boldsymbol{X}, \boldsymbol{M}, \boldsymbol{U}, \boldsymbol{Y}\right) = \arg\min_{\mathbf{V}} \left(-log\, \boldsymbol{F}(\mathbf{V})\right), \tag{3.16}$$

was zu einem nichtlinearen Problem der kleinsten Quadrate führt und mit einer nichtlinearen Optimierungsmethode, wie dem Gauss-Newtown oder Levenberg-Marquardt Algorithmus, gelöst werden kann [Hu et al., 2013, Roweis, 1996].

Sofern ein Anfangszustand der Parameter bekannt ist, kann eine numerische Lösung für Gleichung (3.16) gefunden werden. Die Fehlerfunktion $\boldsymbol{e}_{ij}(\breve{\boldsymbol{v}})$ wird dabei durch ihre Taylor-Erweiterung erster Ordnung um einen Anfangswert approximiert, siehe Kümmerle et al. [2011]:

$$\boldsymbol{e}_{ij}(\breve{\boldsymbol{v}}_{ij} + \Delta\boldsymbol{v}_{ij}) = \boldsymbol{e}_{ij}(\breve{\boldsymbol{v}} + \Delta\boldsymbol{v}) \tag{3.17}$$

$$\simeq \boldsymbol{e}_{ij} + \mathbf{J}_{ij}\Delta\boldsymbol{v}, \tag{3.18}$$

mit Jakobi-Matrix $\mathbf{J}_{ij}$ und $\boldsymbol{e}_{ij} \overset{\text{def.}}{=} \boldsymbol{e}_{ij}(\breve{\boldsymbol{v}})$. Setzt man nun Gleichung (3.18) in die Fehlerfunktion von $\boldsymbol{F}_{ij}$ der Gleichung (3.14) ein, so erhält man:

$$\begin{aligned}
\boldsymbol{F}_{ij}(\breve{\boldsymbol{v}} + \Delta\boldsymbol{v}) =& \boldsymbol{e}_{ij}(\breve{\boldsymbol{v}} + \Delta\boldsymbol{v})^T \boldsymbol{\Omega}_{ij} e_{ij}(\breve{\boldsymbol{v}} + \Delta\boldsymbol{v}) && (3.19)\\
\simeq& (\boldsymbol{e}_{ij} + \mathbf{J}_{ij}\Delta)^T \boldsymbol{\Omega}_{ij} (\boldsymbol{e}_{ij} + \mathbf{J}_{ij}\Delta) && (3.20)\\
=& \underbrace{\boldsymbol{e}_{ij}^T \boldsymbol{\Omega}_{ij} \boldsymbol{e}_{ij}}_{c_{ij}} + \underbrace{2\boldsymbol{e}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{J}_{ij}}_{\boldsymbol{b}_{ij}} \Delta\boldsymbol{v} + \Delta\boldsymbol{v}^T \underbrace{\mathbf{J}_{ij}^T \boldsymbol{\Omega}_{ij} \mathbf{J}_{ij}}_{\boldsymbol{H}_{ij}} \Delta\boldsymbol{v} && (3.21)\\
=& c_{ij} + 2\boldsymbol{b}_{ij}\Delta\boldsymbol{v} + \Delta\boldsymbol{v}^T \boldsymbol{H}_{ij} \Delta\boldsymbol{v}. && (3.22)
\end{aligned}$$

Mit dieser lokalen Annäherung lässt sich Gleichung (3.14) umschreiben zu:

$$\begin{aligned}
\boldsymbol{F}(\breve{\boldsymbol{v}} + \Delta\boldsymbol{v}) =& \sum_{\langle i,j\rangle \in C} \boldsymbol{F}_{ij}(\breve{\boldsymbol{v}} + \Delta\boldsymbol{v}) && (3.23)\\
\simeq& \sum_{\langle i,j\rangle \in C} c_{ij} + 2\boldsymbol{b}_{ij}\Delta\boldsymbol{v} + \Delta\boldsymbol{v}^T \boldsymbol{H}_{ij} \Delta\boldsymbol{v} && (3.24)\\
=& c + 2\boldsymbol{b}^T \Delta\boldsymbol{v} + \Delta\boldsymbol{v}^T \boldsymbol{H} \Delta\boldsymbol{v}. && (3.25)
\end{aligned}$$

Die quadratische Form aus Gleichung (3.25) erhält man durch das Ersetzen von $c = \sum c_{ij}, \boldsymbol{b} = \sum \boldsymbol{b}_{ij}$ und $\boldsymbol{H} = \sum \boldsymbol{H}_{ij}$ aus Gleichung (3.24).

Die Gleichung kann minimiert werden durch das Lösen des folgenden Systems:

$$\mathbf{H}\Delta\boldsymbol{v}^* = -\boldsymbol{b}, \tag{3.26}$$

mit Informationsmatrix $\mathbf{H}$. Die Lösung ergibt sich durch Addition der Inkremente $\Delta\boldsymbol{v}^*$ zur Anfangsschätzung:

$$\boldsymbol{v}^* = \breve{\boldsymbol{v}} + \Delta\boldsymbol{v}^*. \tag{3.27}$$

Der Gauß-Newton-Algorithmus führt die Linearisierung in Gleichung (3.25), die Lösung in Gleichung (3.26) und den Aktualisierungsschritt in Gleichung (3.27) kontinuierlich aus. In jedem Schritt wird die vorherige Lösung als Linearisierungspunkt und als Anfangsschätzung verwendet, bis ein vorgegebenes Abbruchkriterium erfüllt ist.

Im Vergleich zum Gauß-Newton Verfahren erweitert das Levenberg-Marquardt Verfahren die Gleichung (3.26) um einen Dämpfungsfaktor λ, um einen direkten Einfluss auf das Konvergenzverhalten zu haben:

$$(\mathbf{H} + \lambda\boldsymbol{I})\Delta\boldsymbol{v}^* = -\boldsymbol{b}. \tag{3.28}$$

Die Idee hinter dem LM-Algorithmus ist, den Dämpfungsfaktor dynamisch zu steuern, um somit die Schrittweite im Falle von nicht-linearen Oberflächen besser anzupassen. Bei jeder Iteration wird der Fehler der neuen Konfiguration überwacht. Wenn der neue Fehler niedriger ist als der vorherige Fehler, wird λ für die nächste Iteration verringert. Andernfalls wird die Lösung rückgängig gemacht und λ wird erhöht.

Weitere Informationen und eine Herleitung sind in Hu et al. [2013], Kümmerle et al. [2011], Roweis [1996] gegeben.

Die beiden genannte Verfahren sind allgemeine Ansätze zur Minimierung einer multivariaten Funktion. Es wird angenommen, dass der Parameterraum euklidisch ist. Dies ist jedoch für verschiedene Probleme wie SLAM oder den Bündelausgleichung nicht möglich. Um mit Zustandsvariablen umzugehen, die sich über den nicht-euklidischen Raum erstrecken, ist ein gängiger Ansatz, die Inkremente $\Delta\boldsymbol{v}$ in einem anderen Raum als dem der Parameter $\boldsymbol{v}$ auszudrücken. Hierzu wird die Fehlerfunktion mit einem nichtlinearen Operator $\boxplus$ wie folgt beschrieben:

$$\boldsymbol{v}_i^* = \breve{\boldsymbol{v}} \boxplus \Delta\boldsymbol{v}_i^*, \tag{3.29}$$

wobei die Inkremente $\Delta\boldsymbol{v}$ die Störungen um $\breve{\boldsymbol{v}}$ sind [Kümmerle et al., 2011].

In dieser Arbeit entspricht der Box-Operator $\boxplus$ den in Unterabschnitt 2.1.3 eingeführtem Bewegungskompensationsoperator. Die Fahrzeugpose besteht dabei aus einem Translationsvektor (euklidischer Raum) und einer nichtlinearen Rotation, die sich über eine nicht-euklidische Rotationsgruppe SO(2) aufspannt. Es gilt:

$$\breve{\boldsymbol{v}} \boxplus \Delta\boldsymbol{v}_i^* \overset{def.}{=} \breve{\boldsymbol{x}} \oplus \Delta\boldsymbol{x}_i^*. \tag{3.30}$$

Die neue Fehlerfunktion lässt sich dann wie folgt definieren:

$$\boldsymbol{e}_{ij}(\Delta\boldsymbol{x}_i, \Delta\boldsymbol{x}_j) \overset{def.}{=} \boldsymbol{e}_{ij}(\breve{\boldsymbol{x}} \oplus \Delta\boldsymbol{x}) \tag{3.31}$$

$$\simeq \boldsymbol{e}_{ij} + \boldsymbol{J}_{ij}\Delta\boldsymbol{x}, \tag{3.32}$$

wobei $\breve{\boldsymbol{x}}$ über den ursprünglichen über-parametrisierten Zustandsraum aufgespannt ist. Die Jakobi-Matrix $\boldsymbol{J}_{ij}$ ergibt sich dann wie folgt:

$$\boldsymbol{J}_{ij} = \left.\frac{\partial \boldsymbol{e}_{ij}(\breve{\boldsymbol{x}} \boxplus \Delta\boldsymbol{x})}{\partial \Delta\boldsymbol{x}}\right|_{\Delta\boldsymbol{x}=\boldsymbol{0}}. \tag{3.33}$$

Zur Lösung des nichtlinearen SLAM-Problems wird das General Graph Optimization (g2o) Framework von Kümmerle et al. [2011] verwendet. g2o ist eine Open-Source-Software zur Optimierung graphbasierter nichtlinearer Fehlerfunktionen. g2o wurde so konzipiert, dass es leicht auf eine Vielzahl von Problemen erweiterbar ist. Es erlaubt die einfache Definition unterschiedlicher Räume für die Inkremente und die Zustandsvariablen und unterstützt damit transparent beliebige Parametrisierungen innerhalb desselben Problems. Unabhängig von der Wahl der Parametrisierung bleibt die Struktur der Hessischen $\boldsymbol{H}$ im Allgemeinen erhalten [Kümmerle et al., 2011].

4 Umgebungswahrnehmung

Inhalt

4.1 Einleitung

Ohne die Fähigkeit, die unmittelbare Umgebung zu erfassen, hätten autonome Fahrzeuge nicht die Möglichkeit zu wissen, wo sie abbiegen sollen, wann ein Spurwechsel möglich ist und wann die Bremse betätigt werden sollte. Autonome Fahrzeuge müssen alle Hindernisse und bewegte Objekte innerhalb ihrer Umgebung erkennen, um in Sekundenbruchteilen lebensrettende Entscheidungen treffen zu können. Messdaten von Wahrnehmungssensoren wie beispielsweise Kameras, Radar-, Ultraschall- und LiDAR-Sensoren liefern hierfür die maßgeblichen Eingangsdaten. Die Umgebungswahrnehmung ist somit eine entscheidende Schlüsselkomponente für intelligente Fahrzeuge.

Ein Umgebungsmodell ist eine virtuelle und abstrakte Modellierung der statischen und dynamischen Umgebung, das auf Basis von Sensordaten und Modellvorwissen erstellt wird. Die statische Umgebung beinhaltet beispielsweise Beschreibungen zu befahrbaren Freiflächen, zu Hindernissen, die umfahren werden sollten und zu Landmarken, die zur Lokalisierung verwendet werden können. Zu dynamischen Objekten gehören wiederum Verkehrsteilnehmer wie beispielsweise Fahrzeuge, Personen, aber auch Tiere, die sich mit einer Geschwindigkeit größer Null bewegen.

Um zwischen statischer und dynamischer Umgebung zu unterscheiden, müssen die Zustände aller Objekte und Merkmale erfasst, über die Zeit geschätzt und interpretiert werden. Die Schwierigkeit liegt hier u. a. in der exakten Bestimmung der

Abbildung 4.1:
Darstellung des objektbasierten Umgebungsmodells für eine Verkehrsszene. Farbige Punktwolken beschreiben einzelne vertikale Objektinstanzen. Instanzen mit einer Bounding-Box stellen andere Verkehrsteilnehmer dar, wobei der Pfeil die Richtung und Geschwindigkeit (Länge des Pfeiles) darstellt.

Geschwindigkeit, Position und Klasse von Objekten unter Berücksichtigung der Eingangsdaten, die aus unterschiedlichen Zeitschritten und verschiedenen Blickwinkeln aufgenommen wurden.

Im folgenden Abschnitt wird ein objektorientiertes Umgebungsmodell vorgestellt, bei dem sämtliche vertikale Hindernisse und Flächen als generalisierte Objekte dargestellt werden. Landmarken und dynamische Objekte werden dabei mit einer Bounding-Box sowie Freiflächen und Hindernisflächen durch konkave 2D-Polygone, die eine Höhe als Parameter besitzen, repräsentiert. In Abbildung 4.1 ist das Umgebungsmodell am Beispiel einer Verkehrsszene gezeigt.

Die generalisierte Repräsentation von Objekten in der Umgebung ermöglicht die Einhaltung der in Unterabschnitt 1.2.2 genannten Randbedingungen und erlaubt die Übermittlung des Umgebungsmodells als Objektliste zwischen den Teilnehmern eines Konvois.

Die Unterscheidung zwischen Hindernissen und befahrbarem Freiraum ist für autonome Fahrzeuge eine der grundlegenden Fähigkeiten in der Umgebungswahrnehmung. Die Definition von einem Hindernis kann je nach Fahrzeug sehr unterschiedlich aussehen. Im Allgemeinen sind Hindernisse Objekte oder Bereiche, die nicht befahrbar sind oder überfahren werden dürfen, um sich und andere nicht zu gefährden.

Eine robuste Freiraum- und Hinderniserkennung sind daher wichtige Voraussetzungen für autonome Fahrzeuge [Aeberhard, 2017, Cadena et al., 2015, Cheng et al., 2014, Cordts et al., 2016, Girshick et al., 2016, Himmelsbach et al., 2009, Jaspers, 2021, Kuang Chiu et al., 2020, Lai und Fox, 2010, Moosmann et al., 2009, Pillai und

Leonard, 2015, Redmon und Farhadi, 2017, Ren et al., 2017, Salas-Moreno et al., 2013, Steyer et al., 2018, Viola und Jones, 2001, Wang et al., 2007].

Da LiDAR-Messungen 3D-Koordinaten von Objekten liefern, haben sich insbesondere LiDAR-Punktwolken als wichtige Datenquelle für die Wahrnehmung und Erfassung von geometrischen Objekten in dreidimensionalen Szenen für autonome Fahrzeuge erwiesen. Clustering ist dabei der grundlegende Schritt zur Extraktion von Objekten aus 3D-Punktewolken.

Für Verfahren, welche neuronale Netze verwenden, stellen unstrukturierte Gelände eine besondere Herausforderung dar. Dies liegt u. a. daran, dass es noch keine umfassenden Trainingsdatensätze gibt [Caesar et al., 2020, Cordts et al., 2016, Geiger et al., 2013, Geyer et al., 2020, Huang et al., 2020, Lejeune et al., 2018, Maddern et al., 2017, Milioto et al., 2019, Pham et al., 2020, Qi et al., 2017, Romera et al., 2018, Wu et al., 2017a, Yu et al., 2020] und die Generalisierung deutlich schwieriger fällt.

Im folgenden Abschnitt wird daher ein Überblick über die verschiedenen Ansätze zur Unterscheidung zwischen Hindernis und Freiraum sowie der Objekterkennung auf Basis von Punktwolken gegeben. Die Ansätze lassen sich anhand der Art, wie die Daten gespeichert und verarbeitet werden, unterscheiden. In der Datenrepräsentation für Punktwolken gibt es zwei grundlegende Arten: Raster und Rohdaten. Darüber hinaus lassen sich Raster nochmal in ihrer Dimension 3D (Mesh, Voxel), 2D (Gridmaps) oder 2,5D (Gridmaps mit Zusatzinformationen) unterscheiden. Neben der metrischen Darstellung der Gridzellen sind Winkeldarstellungen, wie beispielsweise die sphärische Projektion einer Punktwolke, beliebte Repräsentationen. In Unterabschnitt 3.1.1 wurden hierzu bereits die verschiedenen Arten von Kartenrepräsentationen eingeführt.

Bereits seit der DARPA Urban Challenge sind Occupancy Grid Maps (OGM) eine beliebte Wahl zur Unterscheidung von Freiraum und Bereichen, die umfahren werden müssen [Montemerlo et al., 2008, Urmson et al., 2009a]. 3D-Messungen werden in ein 2,5D-Belegungsraster projiziert und Zellen mit einem Höhenunterschied unter einem vordefinierten Schwellenwert werden als Boden klassifiziert. Im Allgemeinen neigen diese Ansätze nach Chen et al. [2014] zu einer Übersegmentierung, insbesondere in unstrukturierten Bereichen und Bereichen mit unterschiedlichen Steigungen.

Die von Himmelsbach et al. [2010] vorgestellte Methode verwendet im Vergleich zu den quadratischen Gitterkarten eine Gitterkarte im Polarkoordinatensystem. Für jeden Winkelbereich wird die Bodenebene mithilfe einer Linienfunktion approximiert. Anhand des Abstands der Messungen und der Linien erfolgt die Klassifizierung zwischen Bodenfläche und Hindernis. Zusätzlich werden Gitterzellen mit ähnlichen Höhen gruppiert. Die Polarkoordinatenrepräsentation eignet sich besonders für 360° Sensoren.

Chen et al. [2014] erweiterten den Ansatz durch die Verwendung einer eindimensionalen Gaußprozess-Regression. Der vorgeschlagene Ansatz teilt dabei das Bodensegmentierungsproblem in viele einfache GP-Regressionsprobleme auf, die eine geringere Komplexität besitzen.

Im Verfahren von Neuhaus et al. [2009] erfolgt die Unterscheidung zwischen befahrbar und nicht befahrbar ebenfalls in einer grid-basierten Datenstruktur. Zusätzlich werden die als befahrbar klassifizierten Gridzellen auf die lokale Geländerauigkeit untersucht. Diese Zusatzinformation hilft der Pfadplanung zwischen unebenem oder schlammigem Gebiet und einer ebenen Straße zu wählen, was mit einer rein binären Befahrbarkeitsinformation nicht möglich wäre.

Um die Bodenfläche zu approximieren, stellen Zhang et al. [2015] ein kostenbasiertes Messmodell für 3D-LiDAR-Sensoren vor, das in ein Markov-Random-Field (MRF) integriert ist. Das Verfahren liefert eine robuste und nicht-parametrische Schätzung der Bodenhöhe, wobei die Punktwolke in einem Polargrid repräsentiert wird. Das Multi-Label-MRF beinhaltet dabei lokale Glattheits- und Neigungsannahmen, um Hindernisse herauszufiltern, während gleichzeitig scharfe Diskontinuitäten in der Bodenhöhe zugelassen werden. Die Autoren beschreiben, dass mit dem Verfahren sowohl mehrdeutige Situationen als auch Verdeckungen durch nähere Objekte modelliert werden können.

Die Autoren Sengupta und Sturgess [2015] verwenden hingegen ein hierarchisches und robustes MRF, das in ein Octree eingebettet ist. Die Octree-Darstellung ermöglicht ein dynamisches Update, eine hohe Datenkompression und die direkte Rekonstruktion von Oberflächen. Neben dem Volumen von Objekten werden zudem Objektklassen bestimmt.

Das Verfahren von Rieken et al. [2015] kombiniert die Datenrepräsentation der Punktwolke als sphärische Projektion mit einem metrischen Grid, um räumliche Fehlmessungen zu erkennen. Um die Bodenhöhe zu bestimmen, wird eine Neigungswinkelschätzung sowie ein Bordsteinerkennungsmodul vorgestellt.

Die Grundannahme klassischer OGM ist eine stationäre Umgebung. Um den dynamischen Zustand einer Gitterzelle zu repräsentieren und Bewegungen vorherzusagen, sind insbesondere Partikelfilter-basierte Verfahren eine verbreitete Methode.

Danescu et al. [2010, 2011] stellen Verfahren vor, um den dynamischen Zustand einer Gitterzelle mit einem Partikelfilter zu schätzen. Die Veröffentlichungen von Negre et al. [2014] und Tanzmeister et al. [2014] erweitern das Verfahren und berücksichtigen nur den dynamischen Teil einer Gitterkarte mit Partikeln.

Nuss et al. [2015] fusionieren LiDAR- und Radar-Messungen in einer dynamischen Gitterkarte unter strickter Trennung von bewegten und statischen Hindernissen. Die Gesamtleistung wird durch die Berücksichtigung der Dopplermessungen des Radars und der direkten Integration der gemessenen Geschwindigkeit deutlich verbessert. Zusammenfassend lässt sich sagen, dass bisherige Arbeiten zu dynamischen Rasterkarten auf der Basis von Partikeln vielversprechende Ergebnisse zeigen.

Bei dem Verfahren von Nuss et al. [2018] wird der Zustand mehrerer Gitterzellen als eine zufällige endliche Menge definiert, die es erlaubt, die Umgebung als ein stochastisches, dynamisches System mit mehreren Hindernissen zu modellieren, das von einem stochastischen Messsystem beobachtet wird. Die Filterung erfolgt mithilfe

eines Probability Hypothesis Density (PHD) und Multi-Instance Bernoulli (MIB) Ansatzes sowie der Anwendung eines Top-Down Verfahrens.

Steyer et al. [2018] kombinieren die Dempster-Shafer Methodik mit einem Low-Level Partikelfilter-Tracking-Ansatz, um zwischen statischen, freien und dynamischen Gridzellen sowie deren Kombination zu unterscheiden. Zusätzlich können direkt die Zellgeschwindigkeitsverteilungen bestimmt und die Bewegung von dynamischen Gridzellen prädiziert werden. Die Hypothesen werden dabei durch eine angepasste evidenzbasierte Filterung konsistent geschätzt und in einer dynamischen Rasterkarte akkumuliert, sodass zwischen statischer und dynamischer Belegung unterschieden werden kann.

Chen et al. [2020] stellen eine punktwolkenbasierte VoxelGrid-Methode für unstrukturierte Umgebungen vor. Mit einer Gauß-Kernel-Funktion werden Kantenmerkmale extrahiert, welche Hinweise auf Hindernisse geben. Im nächsten Schritt gruppiert ein euklidischer Clustering-Algorithmus einzelne Hindernissinstanzen. Merkmale der Hindernisse werden durch ein neuronales Netzwerk mithilfe der Levenberg-Marquardt-Backpropagation (LM-BP) detektiert.

Das Tiefenbild-basierte Verfahren von Bogoslavskyi und Stachniss [2016] erzeugt über die sphärische Projektion der Punktwolke das Tiefenbild. Im ersten Schritt werden Bodenpunkte durch eine Methode ähnlich wie bei Himmelsbach et al. [2010] entfernt. Als Nächstes wird eine Breitensuche innerhalb des Tiefenbildes durchgeführt und zusammenhängende Pixel unter Überprüfung eines lokalen Winkelkonvexitätskriteriums gruppiert.

Im Ansatz von Klasing et al. [2009] erfolgt die Segmentierung auf Basis der ursprünglichen Eingangsdaten eines LiDAR-Sensors unter Berücksichtigung der Diodenanordnung des Sensors. Nachbarschaftsbeziehungen lassen sich so leicht bestimmen und zusammengehörige 3D-Messungen werden unter Berücksichtigung des euklidischen Abstandes und mithilfe eines Winkelkriteriums bestimmt.

Unter Verwendung einer Graphenrepräsentation der Punktwolke verfolgt die Methode von Moosmann et al. [2009] das gleiche Prinzip. Horizontale Nachbarschaftsbeziehungen des Gridgraphen werden unter Berücksichtigung der kontinuierlichen Sensorrotation der Laserdioden und vertikalen Beziehungen unter Berücksichtigung des Elevationswinkel bestimmt. Ein generisches lokales Konvexitätskriterium entscheidet über zusammenhängende Nachbarschaftsbeziehungen. Um Hindernis- und Bodenpunkte zu unterscheiden, werden die z-Werte der Normalenvektoren in einem histogrammbasierten Ansatz untersucht.

4.2 Verfahren zur Hindernis- und Freiraumerkennung

Im Folgenden wird das Verfahren für die Hindernis- und Freiraumbestimmung, mit dem eine robuste und effiziente Segmentierung der Punktwolken durchgeführt werden kann, beschrieben. Das Verfahren kann gleichermaßen in unstrukturierten

oder städtischen Gebieten eingesetzt werden und ist auf alle mechanisch rotierenden LiDAR-Sensoren anwendbar.

4.2.1 Intrinsisches Sensormuster des Velodyne HDL-64

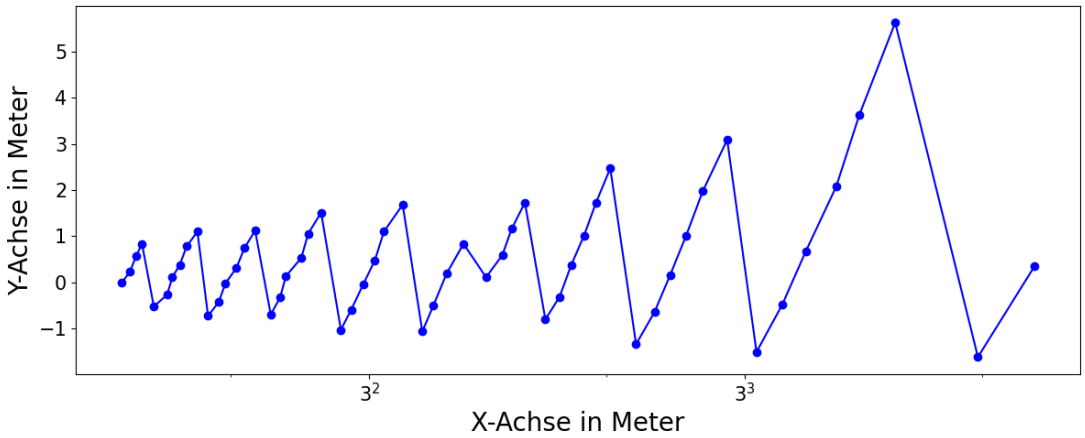
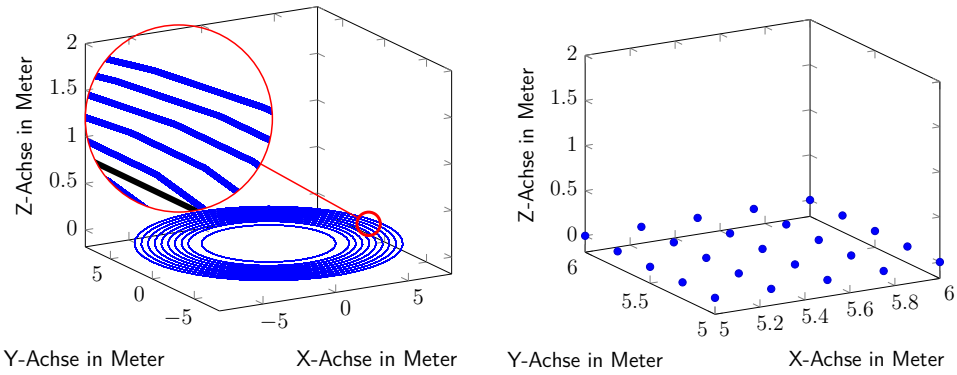


Abbildung 4.2:
Intrinsisches Sensormuster aus der Vogelperspektive: Visualisierung des Velodyne HDL-64 LiDAR-Sensormusters $\mathbf{S}$ mit 56 virtuellen Messungen (blaue Punkte) auf eine virtuelle Bodenebene. Zur besseren Darstellung ist die x-Achse logarithmisch skaliert. Die blaue Linie zeigt die vertikalen Nachbarschaftsbeziehungen mit aufsteigendem Elevationswinkel (entlang der X-Achse) an, wobei nur Laserstrahlen mit einem negativen Elevationswinkel auf die Bodenebene treffen.

Der in dieser Arbeit verwendete Velodyne HDL-64 LiDAR-Sensor ist ein rotierender Sensor mit 64 Laserdioden. Projiziert man alle Laserstrahlen auf eine virtuelle Bodenfläche, so entsteht ein einzigartiges Sensormuster. Das Sensormuster entsteht durch einen unterschiedlichen Azimut und Höhenwinkel der Laserdioden und ist in Abbildung 4.2 dargestellt. Die Messungen sind in Blau und die direkte vertikale Nachbarschaftsbeziehung zwischen den Dioden mit aufsteigendem Höhenwinkel ($-24.8°$ bis $2°$) entlang der X-Achse mit einer blauen Linie gekennzeichnet.

Durch die Rotation um die Sensorachse und die unterschiedlichen Höhenwinkel der Dioden erzeugt jeder Strahl einen idealen Kreis mit einem festen Radius auf einer idealen Bodenfläche. In Abbildung 4.3a ist exemplarisch eine stark vereinfachte Punktwolke eines mechanisch rotierenden LiDAR-Sensors in kartesischen Koordinaten dargestellt.

Der Betrieb von mechanisch rotierenden LiDAR-Sensoren mit 600 Umdrehungen pro Minute, was $10\,\text{Hz}$ entspricht, hat sich für mobile Systeme als ideale Frequenz bewährt. Die Auswahl der Rotationsgeschwindigkeit ist dabei ein Kompromiss zwischen Aktualität der Daten und Abtastdichte. Bei reinen Kartierungsfahrten wird oftmals eine langsamere Rotation gewählt, um die Umgebung dichter abzutasten (höhere horizontale Auflösung).



(a) Idealisiertes Modell **(b)** Vergrößerter Ausschnitt

Abbildung 4.3:
Idealisierte und vereinfachte Punktwolkendarstellung eines rotierenden mechanischen Sensors im kartesischen Koordinatensystem in einer flachen Welt. Durch die Rotation des Sensors um seine eigene Achse entstehen charakteristische Ringe. Jeder Kreisring, siehe Abbildung 4.3a, ist dabei einer Laserdiode zuzuordnen. Jeder Ring besteht dabei aus einer Reihe von Messungen. Die in Abbildung 4.3a vergrößert dargestellten Punktmessungen sind in Abbildung 4.3b als einfache 3D Punkte dargestellt.

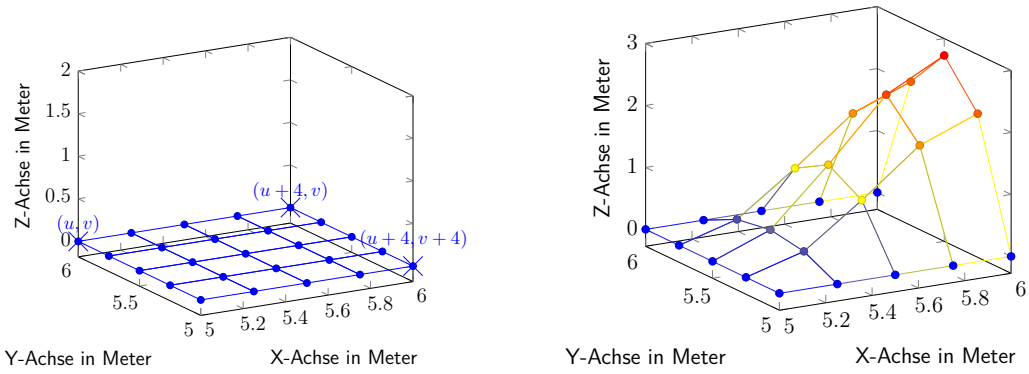
Jede Laserdiode sendet pro Umdrehung ca. $\mu_{\text{max}} = \lfloor {}^{t_{\text{acq}}}/_{\Delta t_{\text{f}}} \rfloor \mid \mu_{\text{max}} \in \mathbb{N}_0 = 2083$ Impulse aus, mit $t_{\text{acq}} = 100\,\text{ms}$ für die Dauer einer Umdrehung und $\Delta t_{\text{f}} = 0.048\,\text{ms}$ als die Zeit zwischen zwei aufeinander folgenden Lichtimpulsen. Die horizontale Winkelauflösung $\alpha_h = \frac{365^\circ}{2083} = 0.175\,312\,2^\circ$ berechnet sich dann aus dem horizontalen Sichtbereich sowie der Anzahl von Impulsen pro Diode.

Rechnerisch wird pro Umdrehung eine Punktwolke mit ca. 130.000 3D-Punkten erzeugt. Die Anzahl an Messungen ist jedoch im Durchschnitt deutlich geringer, da einige ausgesendete Laserimpulse ihren Weg nicht zurück zur Empfangseinheit des Sensors finden.

Der Dateneinzug erfolgt über die Netzwerkschnittstelle. Der Sensor liefert innerhalb einer Umdrehung ca. 348 User Datagram Protocol (UDP)-Datenpakete. Pro UDP-Datenpaket werden neben Distanzmessungen auch Informationen über die intrinsische Kalibrierung der Dioden sowie Daten über den Sensorzustand übermittelt. Jedes dieser Pakete enthält sechs aufeinander folgende Sensormuster $\mathbf{S}_\mu$, wobei jedes Sensormuster des Velodyne HDL-64 LiDAR-Sensors aus jeweils 64 Entfernungsmessungen und 64 Intensitätsmessungen besteht.

4.2.2 Punktwolkenrepräsentation mittels Mesh-Graph

Die Grundidee des entwickelten Ansatzes ist, einen idealisierten Mesh-Graphen auf Basis des Sensormusters zu erstellen, der die zeitliche und räumliche Zusammengehörigkeit der Entfernungsmessungen modelliert. Die idealisierte Projektion des Sensormusters auf eine flache Ebene bildet die Basis des Modells. Überträgt man die Punktwolkendarstellung aus Abbildung 4.3b auf eine Mesh-Graphen Darstellung,



(a) Idealisiertes Modell **(b)** Verformtes Modell

Abbildung 4.4:
In Abbildung 4.4a ist die idealisierte Projektion eines Teilstücks des Mesh-Graphen auf eine flache Ebene im kartesischen Koordinatensystem dargestellt. Blaue Knoten und Kanten entsprechen dem idealisierten Modell. Die Indizes u, v der Graph-Knoten sind exemplarisch für 3 Knoten gezeigt. In Abbildung 4.4b wird der Mesh-Graphen durch ein Hindernis verformt. Die Kanten-Beziehungen zwischen den Knoten und die Knotenpositionen sind stark verändert. Graph-Knoten im gelben und roten Bereich zeigen eine starke Abweichung zum idealisierten Modell an.

so werden die 3D-Messungen (blaue Punkte) durch Graph-Knoten ersetzt, siehe Abbildung 4.4a.

Der Aufbau des Mesh-Graphen ist dabei durch die intrinsische Diodenanordnung und Rotationsgeschwindigkeit des Sensors vorgegeben. Die horizontalen und vertikalen Linien (Kanten) zwischen den Knoten beschreiben die Beziehung zwischen den Messungen. So ergibt sich entlang der Kreisringe eine zeitliche und über den Radius der einzelnen Ringe eine statische räumliche Abhängigkeit. Die Graph-Kanten des idealisierten Modells aus Abbildung 4.4a sind sehr gleichmäßig aufgebaut. Anhand der Kantenbeziehungen kann das Graphen-Modell konstruiert werden.

Durch Objekte und Hindernisse in der realen Welt kommt es nach Burger und Wuensche [2018] zu Verformungen des Graphen und somit zu einer Verletzung der Modellannahmen. In Abbildung 4.4b ist die Verformung des Mesh-Graphen durch ein Objekt exemplarisch dargestellt. Graph-Knoten, die zu stark vom idealisierten Mesh-Graphen abweichen, können auf Basis der erkannten Abweichung als *Hindernis* klassifiziert werden. Dabei werden nicht nur die Positionen der Knoten, sondern insbesondere auch die Kantenbeziehungen zu allen Nachbarn berücksichtigt.

Der Mesh-Graph $\mathbf{G}$ besteht aus einer Menge von Knoten (engl. Vertices) $\mathbf{V}$ und Kanten (engl. Edges) $\mathbf{E}$ und ist wie folgt definiert:

$$\mathbf{G} = (\mathbf{V}, \mathbf{E}), \tag{4.1}$$

Der Mesh-Graph hat eine Größe von $u_{\text{max}} = 64 \mid u \in \{1, \ldots, 64\}$ Zeilen und $v_{\text{max}} = 2084 \mid v \in \{1, \ldots, 2084\}$ Spalten. Die vertikale und horizontale Anzahl an

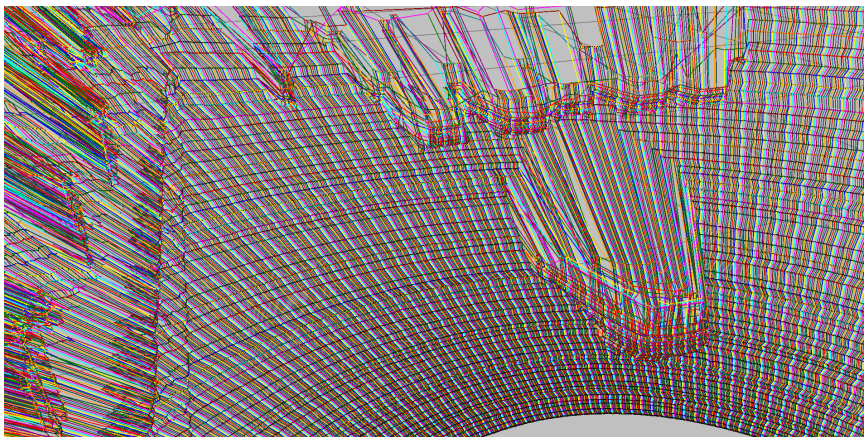


Abbildung 4.5:
Mesh-Graphen Darstellung der vertikalen und horizontalen Kanten. Hindernisse, wie beispielsweise ein Fahrzeug (rechts unten im Bild), verformen die Struktur des Graphen. Zur verbesserten Darstellung sind Zeilen und Spalten des Mesh-Graphen mit verschiedenen Farben gekennzeichnet.

Knoten des Mesh-Graphen ist durch die Anzahl von Laserdioden und die Anzahl von Messungen pro Diode und Sensorrotation vorgegeben, siehe Unterabschnitt 4.2.1.

Zur Vereinfachung der Notation wird zusätzlich zwischen horizontalen und vertikalen Kanten $\mathbf{E} = \{\mathbf{E}^h, \mathbf{E}^v\}$ unterschieden. Jede Eingangsmessung ist dabei genau einmal in der Menge aller Knoten enthalten und kann eindeutig durch die Indizes u, v zugeordnet werden.

In Abbildung 4.5 ist ein Ausschnitt des Mesh-Graphen am Beispiel einer Verkehrsszene gezeigt. Die farbigen horizontalen und vertikalen Linien repräsentieren die Kantenbeziehungen zwischen den Knoten. Die unterschiedlichen Kantenlängen und Verformungen des Mesh-Graphen durch Objekte sind sehr gut zu erkennen.

Der Aufbau des Mesh-Graphen ist dem eines Polargrids sehr ähnlich. Im Allgemeinen erfolgt die Einordnung von unstrukturierten 3D-Messungen von tiefen-gebenden Sensoren in 2D-Repräsentationen, wie beispielsweise bei Polargrids und Tiefenbildern, anhand des Azimut- und Elevationswinkels. Durch Diskretisierungsfehler, die insbesondere bei rotierenden LiDAR-Sensoren auftreten, werden fälschlicherweise verschiedene Messungen derselben Pixelzelle zugeteilt. Eine inertiale Korrektur der Punktwolke auf Basis der Eigenbewegung wirkt sich zusätzlich negativ auf das Ergebnis aus, da sich durch die Neuberechnung der euklidischen Position jeder Messung die Nachbarschaften der ursprünglichen Messungen ändern.

Die Berechnung der Indizes u, v erfolgt in dieser Arbeit jedoch anhand des Sensormusters und nicht z. B. durch die Bestimmung der Polarkoordinaten jeder Messung. Zusätzlich sind die Abstände (Kanten) zwischen den Graph-Knoten variabel und nicht durch die Datenstruktur vorgegeben, wie das beim Tiefenbild und Polargrid der Fall ist. Es kommt somit beim Erstellen der Datenstruktur zu keinem Diskretisierungsfehler und die Punktwolken bleiben in ihrer ursprünglichen Form erhalten.

Das im Folgenden vorgestellte Verfahren macht sich das Vorwissen über die Kantenbeziehungen zu Nutze, um Abweichungen zu detektieren. Im Vergleich zu Grid-basierten Ansätzen ermöglicht die Mesh-Graphen Repräsentation die Überprüfung und die Klassifizierung jedes einzelnen Punktes. Die Klassifizierung jeder einzelnen Messung erfolgt dabei durch den Vergleich der Modellannahme mit den echten Messungen.

Visualisiert man genau ein Sensormuster in einer Zylinderprojektion, wie in Abbildung 4.6 dargestellt, so ergibt sich keine einheitliche gerade vertikale Linie, sondern eine charakteristische Abfolge. Diese statische Verschiebung der Dioden innerhalb des Sensormusters wird zur Erstellung des Graphen und bei der punktweisen Segmentierung als Vorwissen berücksichtigt.

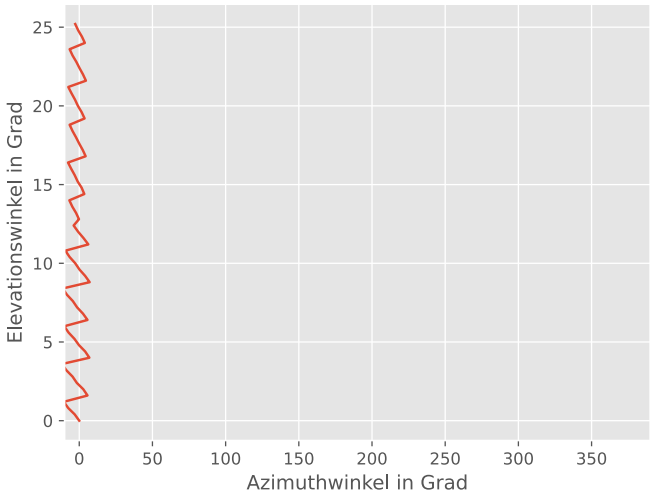


Abbildung 4.6:
Zylinderprojektion eines Sensormusters $\mathbf{S}_{u=1}$, das die Winkelverschiebung der einzelnen Dioden darstellt. Zur besseren Darstellung sind die Messungen pro Sensormuster durch eine Linie mit aufsteigendem Elevationswinkel verbunden.

Im Folgenden wird nun die Berechnung der statischen Winkelverschiebung pro Sensormuster beschrieben. Als Nullpunkt und Referenzpunkt wird die Diode $(u = 1)$, welche den kleinsten Elevationswinkel besitzt, gewählt. Der Index u ist direkt abhängig vom Elevationswinkel der Diode und gibt den vertikalen Index des korrespondierenden Knoten an. Die Referenzdiode des ersten Sensormusters $\mathbf{S}_{u=1}$ erzeugt den ersten Knoten im Graphen mit den Indizes $u = 1 \wedge v = 1$. Die Referenzdiode des zweiten Sensormusters $\mathbf{S}_{u=2}$ hat entsprechend die Indizes $u = 1 \wedge v = 2$.

Um den Index v zu bestimmen, wird die statische Winkelverschiebung aller Dioden zur Referenzdiode auf Basis der Azimutdifferenz berechnet. Diese statische Verschiebung (Offset) ist nur einmal pro Sensor zu bestimmen. Ändert sich die Diodenanordnung, muss der Offset ebenfalls neu bestimmt werden.

Der Azimutoffset jeder Diode $i \in \{2, 3, 4, 5, \ldots, 64\}$ zur Referenzdiode mit $u = 1$ lässt sich wie folgt bestimmen:

$$\Delta\Psi_i = \Psi_1 - \Psi_i. \tag{4.2}$$

Der horizontale Offset v_{offset} ergibt sich, indem der Azimutoffset durch die horizontale Winkelauflösung dividiert wird:

$$v_{\text{offset}} = \lceil {}^{\Delta\Psi_i}/_{\alpha_h} \rceil. \tag{4.3}$$

Der horizontale Index v lässt sich dann mithilfe folgender Berechnung für jede Diode des Sensormusters S_u in Abhängigkeit des statischen und horizontalen Offsets v_{offset} und des vertikalen Indexes u bestimmen:

$$v = \begin{cases} u & v_{\text{offset}} \in \{0, u_{\text{max}}\} \\ u + 2v_{\text{max}} - v_{\text{offset}} & v_{\text{offset}} > u_{\text{max}} \\ u + v_{\text{max}} + v_{\text{offset}} & v_{\text{offset}} < 0 \\ u + v_{\text{offset}} & else \end{cases}. \tag{4.4}$$

Die in Abbildung 4.7 dargestellte vollständige Zylinderprojektion der Sensormuster einer Rotation bildet somit die Grundlage des Mesh-Graphen. Die akkumulierte Menge aller Sensormuster pro Rotation wird dabei als eine Zeitreihe betrachtet, welche die kontinuierliche Messung aller Dioden beschreibt.

Die horizontalen Graph-Kanten $\mathbf{E}^h$ sind abhängig von der kontinuierlichen Drehung des Sensors und bilden die direkte zeitliche Nachbarschaft zwischen zwei Messungen der gleichen Diode:

$$\mathbf{E}^h = \left\{ (\mathbf{v}_{u,v_i}, \mathbf{v}_{u,v_{i+1}}) \mid i = \{1, \ldots, v_{\text{max}-1}\} \right\}. \tag{4.5}$$

Die vertikalen Kanten können dann wie folgt definiert werden:

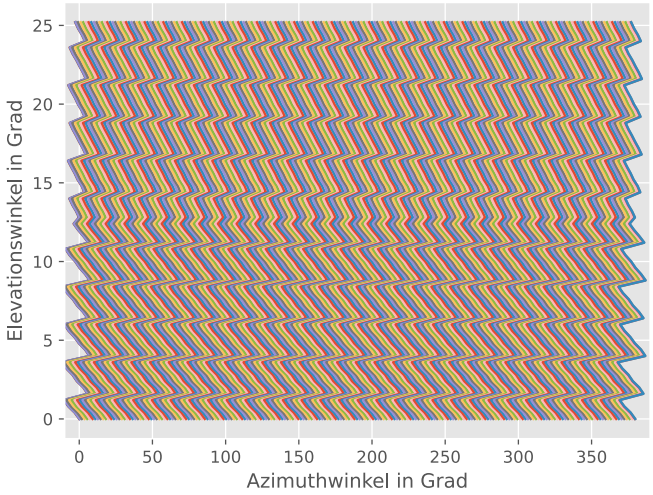
$$\mathbf{E}^v = \left\{ (\mathbf{v}_{u,v_1}, \mathbf{v}_{u,v_2}) \middle| |v_1 - v_2| = 1 \right\}. \tag{4.6}$$

Aus Effizienzgründen erfolgt die Erstellung der Graphenstruktur einmal zu Beginn der Programmlaufzeit. Die Reihenfolge der Distanzmessungen ist deterministisch und so können die Indizes für jede Messung einmal bestimmt und in einer Lookup-Tabelle gespeichert werden. Dies ermöglicht die Konstruktion und das Füllen des Mesh-Graphen innerhalb von $1\,\text{ms}$, da Knoten und Kanten nur noch aktualisiert werden müssen.

Jeder Graph-Knoten besitzt neben den räumlichen und zeitlichen Beziehungen zu den Nachbarn auch Attribute wie die Entfernungsmessung r, die anhand der Sensorkalibrierung berechneten kartesischen Koordinaten x, y, z sowie ein Label l:

$$\mathbf{v} = \{x, y, z, r, l\}, \mathbf{v} \in \mathbf{V}. \tag{4.7}$$

Das Label l ordnet jedem Knoten eine Klasse zu, die folgende Werte annehmen können: *Freiraum*, *Hindernis*, *Rauschen*, *Kante* oder *Ungültig*.



(a) $t = 100\,\text{ms}$

Abbildung 4.7:
Zylinderprojektion des Mesh-Graphen unter Berücksichtigung des intrinsischen Sensormusters $\mathbf{S}_u \mid u \in \{1, \ldots, 2084\}$ für die Dauer einer vollständigen Rotation von $360°$.

4.2.3 Mehrstufiger Segmentierprozess

In diesem Abschnitt wird das mehrstufige Verfahren zur Echtzeit-Segmentierung auf der Basis des zuvor beschriebenen Mesh-Graphen vorgestellt. Um die Abweichungen zu erkennen wird nun im Folgenden die Bestimmung des Modells auf Basis einer Modellfunktion beschrieben und anschließend der vollständige Prozess genau erläutert.

4.2.3.1 Modellfunktion

Um Abweichungen des Sensormusters festzustellen, wird eine Modellfunktion auf Basis des idealisierten Sensormusters bestimmt. In Abbildung 4.8 ist das idealisierte Sensormuster über den vertikalen Graphen-Index u aufgetragen.

Der Index u ordnet mit aufsteigendem Höhenwinkel die Graph-Knoten eines Sensormusters an. Die grünen Punkte repräsentieren die Höhenmessungen pro Diode und Sensormuster. Die blauen Punkte repräsentieren die Distanzmessungen des idealisierten Sensormusters und entsprechen der geometrischen 2D-Darstellung, die bereits in Abbildung 4.2 gezeigt wurde.

Im flachen Weltmodell liegen alle Messungen auf der Höhe Normal-Null (lila Linie) und die Distanzmessungen pro Diode entsprechen dem Verlauf einer Funktion höherer Ordnung, dargestellt in Orange. Diese Funktion, im folgenden Modellfunktion genannt, kann durch ein Polynom achten Grades $f(\tilde{r}) = \sum_{j=0}^{8} a_j \tilde{r}^j$ approximiert werden. Zur Bestimmung der Koeffizienten a_j wurde ein Gauß-Newton-Verfahren

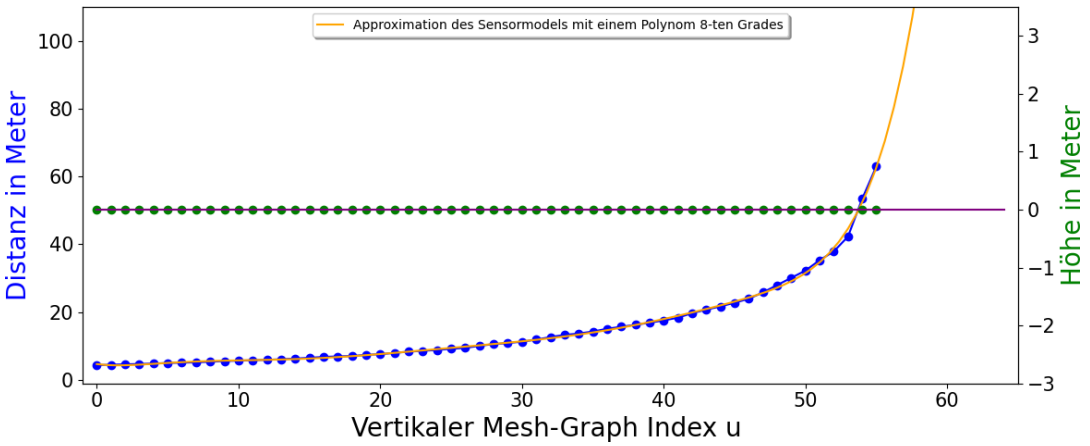


Abbildung 4.8:
Die Abbildung zeigt die Modellfunktion f (Orange) des idealen Sensormusters. Die blauen und grünen Punkte repräsentieren die aus dem Modell bestimmte Distanzmessung $\tilde{r}$ sowie Höhenmessung z auf der Y-Achse und die vertikale Spalte des Graphen mit fortlaufendem Index u auf der X-Achse.

nach der Methode der kleinsten Quadrate verwendet. Die Koeffizienten des in dieser Arbeit verwendeten Sensors sind in Tabelle 4.1 aufgelistet.

Tabelle 4.1:
Koeffizienten der Modellfunktion $f(r_u)$ eines Velodyne HDL-64 LiDAR-Sensor.

$a_8 = 5.60519e-11$	$a_7 = -1.05775e-08$	$a_6 = 8.20012e-07$
$a_5 = -3.34620e-05$	$a_4 = 7.65166e-04$	$a_3 = -9.46870e-03$
$a_2 = 5.94245e-02$	$a_1 = -3.52446e-02$	$a_0 = 5.12710$

Befinden sich vertikale Objekte im Sichtbereich des Sensormusters, kommt es zu Abweichungen bei den Graph-Knoten Distanzen r zur Modellfunktion f sowie bei den Höhen z zur Höhe Normal-Null. In Abbildung 4.9 sind die Veränderungen des Sensormusters zum Modell dargestellt. Darüber hinaus ist nicht für jede Diode eine gültige Messung vorhanden.

Um die Abweichung zur Modellfunktion zu berechnen, wird eine Kostenfunktion verwendet, die den quadratischen Fehler der echten Distanzmessungen r_u an den Stützpunkten u der idealisierten Modellfunktion $f(\tilde{r}_u)$ bestimmt:

$$E = \sum_{u=1}^{64} |f(\tilde{r}_u) - r_u|^2. \tag{4.8}$$

Im idealisierten Modell treten keine Verformungen auf und die Kosten sind $E = 0$.

In der geometrischen zweidimensionalen Darstellung in Abbildung 4.10 ist das idealisierte Sensormuster in Blau mit dem der echten Messungen in Grau gegenübergestellt. Als Hindernis klassifizierte Graph-Knoten sind in Rot eingefärbt. An den Graph-Kantenbeziehungen sind die Abweichungen zwischen den Sensormustern zu erkennen.

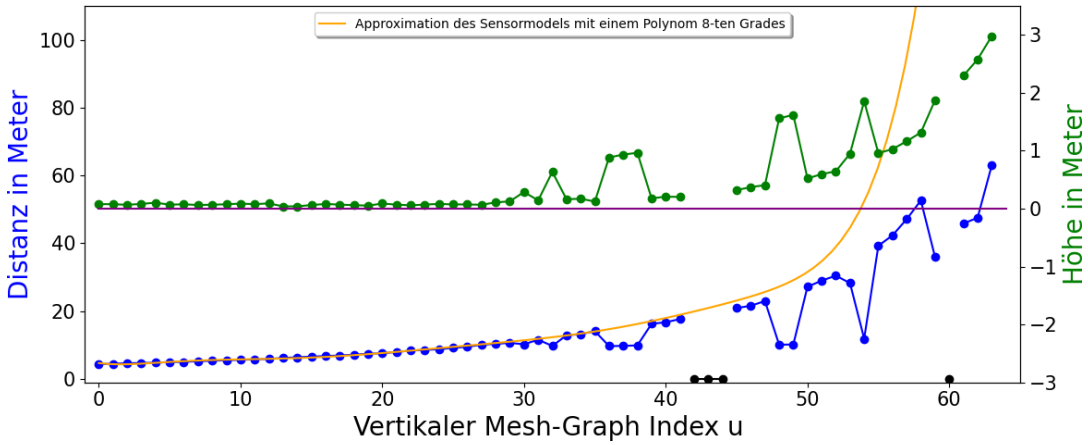


Abbildung 4.9:
Die Abbildung zeigt die Veränderung der Graph-Knoten durch ein Hindernis im Sichtbereich des Sensormusters. Die Graph-Knoten des Sensormusters, in Blau und Grün dargestellt, entsprechen nicht mehr der Modellfunktionen (orangefarbenes Polygon). Ungültige bzw. fehlende Messungen sind als schwarze Punkte dargestellt.

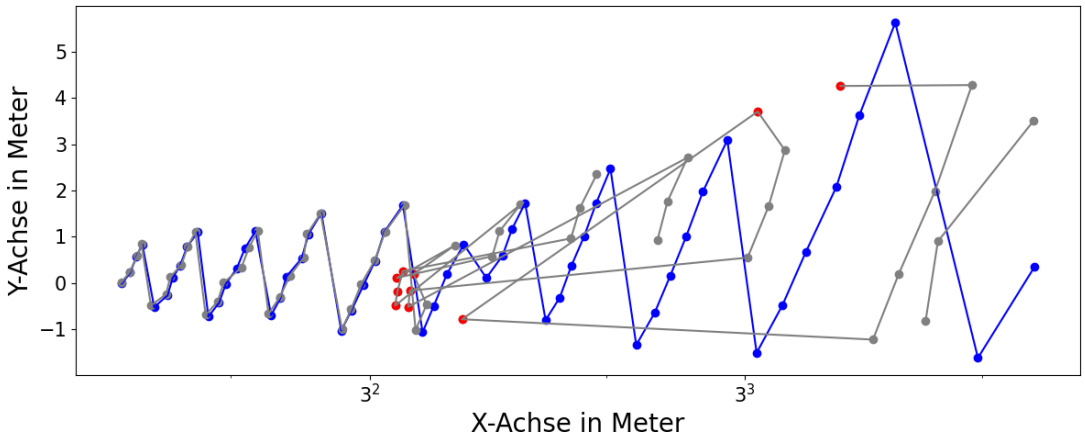


Abbildung 4.10:
Vergleich des idealisierten Sensormusters im flachen Weltmodell (blau) mit einem Sensormuster, bei dem sich ein vertikales Objekt im Sichtbereich befindet (grau). Als Hindernis klassifizierte Graph-Knoten sind in Rot eingefärbt unter Berücksichtigung der grauen Kantenbeziehungen, die stark vom Modell abweichen.

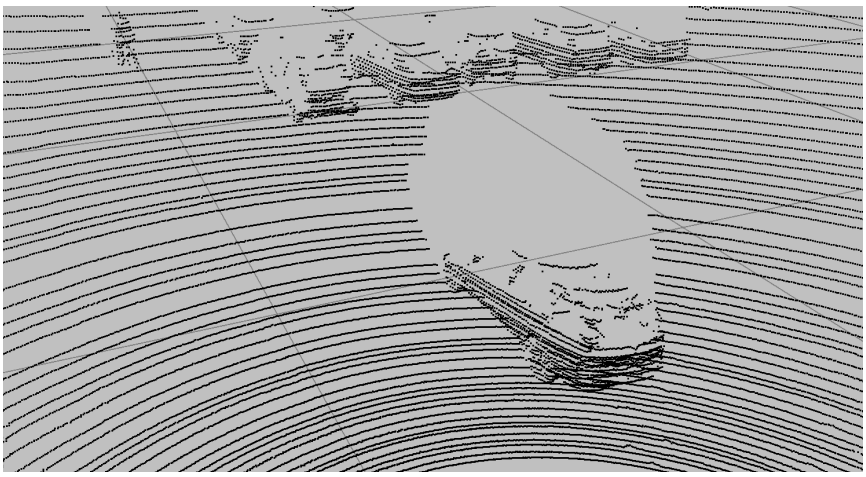


Abbildung 4.11:
Eingangspunktwolke, zu der alle Graph-Knoten das Label *Freiraum* (schwarze Farbe) besitzen.

Die Klassifizierung anhand des Funktionsvergleiches ist trotz Verteilung auf acht CPU-Kerne rechenintensiv. Die durchschnittliche Laufzeit liegt bei ca. $30\,\text{ms}$, was bereits $1/3$ der möglichen Zykluszeit des SLAM-Verfahrens entspricht.

Um die Prozesslaufzeit weiter zu reduzieren, wurden auf Basis der Kostenfunktion E und der idealisierten Modellfunktion f Heuristiken abgeleitet, welche die Laufzeit der Segmentierung auf $1/6$ reduzieren und insbesondere das Gesamtergebnis in unstrukturierter Umgebung deutlich verbessern konnten.

Der Klassifizierungsprozess lässt sich in drei Schritte unterteilen:

1. Identifizierung von Schlüsselknoten, die mit hoher Wahrscheinlichkeit zu einem Hindernis gehören.
2. Agglomeratives hierarchisches Clusteringverfahren, das unter Berücksichtigung der Schlüsselknoten Hindernisscluster identifiziert.
3. Verfeinerung durch erweiterten Modellvergleich.

In jedem einzelnen Schritt des Verfahrens wird iterativ und kontinuierlich die Modellvorstellung überprüft und bei Abweichung der Graph-Knoten als *Hindernis* klassifiziert. Iterativ bedeutet, dass die Ergebnisse auf den vorherigen Schritten aufbauen und so kontinuierlich das Ergebnis verbessert wird. Zu Beginn des Verfahrens besitzen alle Graph-Knoten das Label *Freiraum* und sind als schwarze 3D-Punkte in Abbildung 4.11 dargestellt.

Um die Segmentierungsaufgabe zu lösen, wird jede vertikale Spalte v des Mesh-Graphen separat betrachtet und mit dem idealisierten Modell verglichen. Die Ausführung erfolgt dabei spaltenweise, was eine parallele Bearbeitung ermöglicht. Zur Vereinfachung der Schreibweise werden die vertikalen Knoten im Folgenden $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{64}$ nur noch über ihren vertikalen Index $u \in \mathbf{U} \mid \mathbf{U} = \{1, \ldots, 64\}$ beschrieben, sodass $\mathbf{e}_u = (\mathbf{v}_u, \mathbf{v}_{u+1}) \mid \mathbf{E}^v = \{\mathbf{e}_1, \ldots, \mathbf{e}_{63}\}$.

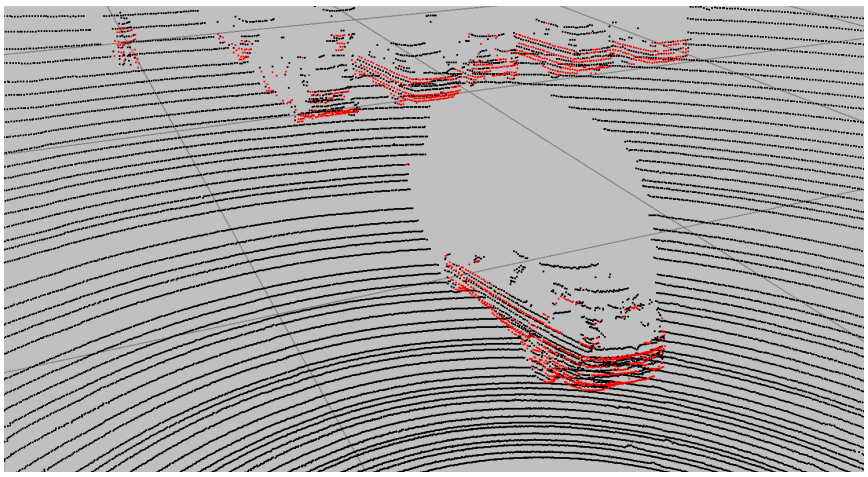


Abbildung 4.12:
Schlüsselknoten (in Rot) liefern wichtige Stützpunkte bei der Klassifizierung.

4.2.3.2 Schlüsselknoten

Im ersten Durchlauf wird eine Menge an Schlüsselknoten $\mathbb{K}$ identifiziert, die sehr stark die Annahme der flachen Welt verletzen. Dies ist gegeben, sofern die Distanz r eines Graph-Knoten $\mathbf{v}_{u+1}$ kleiner ist als die eines Knoten $\mathbf{v}_u$. Bezogen auf die Modellfunktion aus Gleichung (4.8) bedeutet dies, dass die Reihenfolge sowie die Anordnung über die Distanzmessung r der Graph-Knoten mit aufsteigendem Elevationswinkel gegenüber der Modellvorstellung verletzt ist.

Das Kriterium wird für jede vertikale Kante $(\mathbf{v}_u, \mathbf{v}_{u+1})$ überprüft:

$$l_{u+1} = \begin{cases} \textit{Hindernis} & r_u > r_{u+1} \\ 0 & \text{else.} \end{cases} \tag{4.9}$$

Ist die Bedingung erfüllt, so wird der Knoten $\mathbf{v}_{u+1}$ zur Menge von Schlüsselknoten hinzugefügt $\mathbf{v}_{u+1} \in \mathbb{K}$. In Abbildung 4.12 ist das Ergebnis dargestellt, wobei die Schlüsselknoten zur Gruppe der Hindernisknoten gehören.

Durch die paarweise Klassifizierung werden nicht alle Knoten gleichermaßen berücksichtigt und das Ergebnis ist speziell bei vertikalen Objekten noch nicht zufriedenstellend. Im zweiten Schritt wird eine multimodale Verteilung der Graph-Knoten für jede vertikale Spalte bestimmt um Gruppen von Knoten zu identifizieren, die einem Hindernis entsprechen.

4.2.3.3 Agglomeratives hierarchisches Clustering

Um alle weiteren Knoten auf Basis der Schlüsselknoten zu klassifizieren, wurde ein für die Anwendung optimiertes agglomeratives hierarchisches Clustering entwickelt. Agglomerativ bedeutet, dass jeder Knoten in einem eigenen Cluster beginnt und Paare von Clustern zusammengeführt werden [Bouguettaya et al., 2015, Shalizi, 2009]. Ein

Vorteil des Verfahrens im direkten Vergleich zu k-means Clustering ist, dass vorab keine Kenntnis über die Anzahl von Clustern benötigt wird [Lloyd, 1982]. Zusätzlich ermöglicht das Verfahren, ganze Clusterhierarchien abzubilden [Bouguettaya et al., 2015].

Das Clustering-Verfahren berücksichtigt neben der Dichte der zugrunde liegenden Punktverteilung auch die intrinsische Anordnung des Musters in Abhängigkeit der Distanz, der Höhe und dem vertikalen Index u jeder Diode. Die Schlüsselknoten bilden dabei die initialen Cluster. Um die Anzahl der Cluster zu reduzieren und zusammengehörige Cluster zu fusionieren, sind mehrere Schritte und Kriterien nötig.

Das Distanzmaß zwischen Clusterpaaren wird mithilfe des euklidischen Abstands (Metrik) der jeweiligen Zentroiden von zwei Clustern $\mathcal{A}, \mathcal{B}$ bestimmt, wobei ein Abstand von $0\,\mathrm{m}$ der maximalen Ähnlichkeit entspricht:

$$D(\mathcal{A},\mathcal{B}) = \frac{1}{|\mathcal{A}| \cdot |\mathcal{B}|} \sum_{\boldsymbol{a}\in\mathcal{A}} \sum_{\boldsymbol{b}\in\mathcal{B}} d(\boldsymbol{a},\boldsymbol{b}) \mid \boldsymbol{a},\boldsymbol{b} \in \{\boldsymbol{v}_1,\ldots,\boldsymbol{v}_{64}\}, \tag{4.10}$$

mit $|\mathcal{A}|, |\mathcal{B}|$, als die Kardinalität der Cluster. Die Modellannahme und das Kriterium, wann zwei Cluster fusioniert werden, ist wie folgt definiert:

$$\mathcal{A} \cup \mathcal{B} = \begin{cases} \text{Richtig} & \min(\mathcal{B},u) \in \{\min(\mathcal{A},u),\ldots,\max(\mathcal{A},u)\} \vee \\ & \min(\mathcal{B},u) > \max(\mathcal{A},u) \wedge \min(\mathcal{B},z) > \min(\mathcal{A},z) \\ \text{Falsch} & \text{andernfalls,} \end{cases} \tag{4.11}$$

unter Berücksichtigung des minimalen $\min(\mathcal{B},u)$, $\min(\mathcal{A},u)$ und maximalen $\max(\mathcal{B},u)$ vertikalen Index u pro Cluster, sowie den minimalen Höhen $\min(\mathcal{B},z), \min(\mathcal{A},z)$.

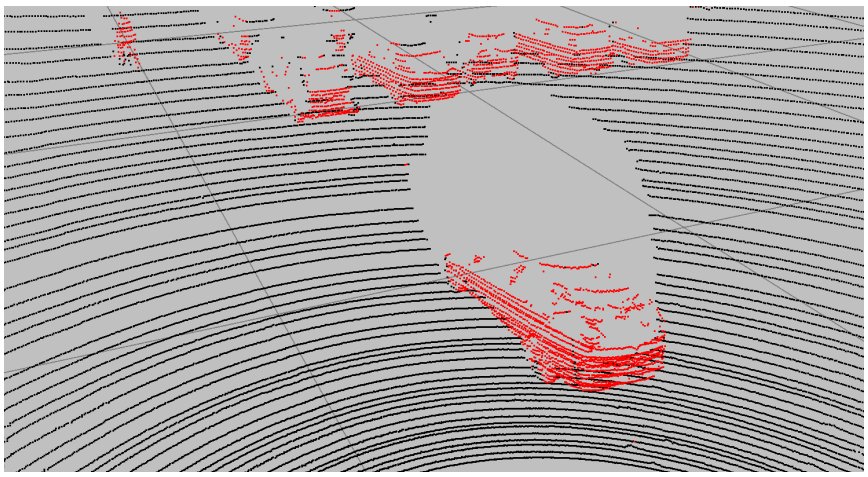


Abbildung 4.13:
Zu Beginn des Verfahrens sind alle Knoten als *Freiraum* markiert. Durch den mehrstufigen Prozess werden die Kanten zwischen den Graph-Knoten auf verschiedene Kriterien überprüft und bei Abweichung zu einem Modell als *Hindernis* (rot) klassifiziert.

Das agglomerative hierarchische Clusteringverfahren lässt sich in verschiedene Stufen unterteilen, die jeweils für jede Zeile des Graphen ausgeführt werden. Das Verfahren erfolgt nach dem Multi-Pass-Prinzip [Burger und Wuensche, 2018]. In mehreren Schritten werden die Graph-Knoten mit aufsteigendem Index u betrachtet. Jedes Cluster entspricht dabei einem Hindernis. Knoten, die am Ende des Verfahrens zu keinem Cluster gehören, werden als Freiraum klassifiziert. Im Folgenden sind die verschiedenen Schritte des abgewandelten Verfahrens beschrieben:

1. Starte mit n Clustern (n entspricht der Anzahl an Reihen des Mesh-Graphen), jedes Cluster besteht aus genau einem Graph-Knoten.
2. Berechne das Distanzmaß $D(\mathcal{A}_r, \mathcal{A}_s)$ als das Maß zwischen den Clustern mit $r, s = 1, 2, ..., 64$, mit $D = (D(\mathcal{A}_r, \mathcal{A}_s))$ als die quadratische Distanzmatrix.
3. Suche das ähnlichste Clusterpaar $\mathcal{A}_r$ und $\mathcal{A}_s$, sodass das Maß $D(\mathcal{A}_r, \mathcal{A}_s)$ unter allen paarweisen Distanzen minimal ist.
4. Überprüfe, ob die Modellannahme für das Clusterpaar mit dem geringsten Abstand gilt. Wenn nicht, überprüfe das Clusterpaar mit dem zweitgeringsten Abstand, bis ein Clusterpaar gefunden ist, für das die Bedingung erfüllt ist.
5. Wenn Distanz- und Modellkriterium erfüllt, fusioniere $\mathcal{A}_r$ und $\mathcal{A}_s$ zu einem neuen Cluster $\mathcal{A}_t$ zusammen. Berechne den Abstand zwischen den Clustern $D(\mathcal{A}_t, \mathcal{A}_k)$ für jedes vorhandene Cluster $\mathcal{A}_k \neq \{\mathcal{A}_r, \mathcal{A}_s\}$. Lösche die entsprechenden Zeilen und Spalten aus der Ähnlichkeitsmatrix, die zu den Clustern $\mathcal{A}_r, \mathcal{A}_s$ gehören. Füge eine neue Zeile und Spalte in $\mathbb{D}$ ein, die dem neuen Cluster $\mathcal{A}_t$ entspricht.
6. Wiederhole Schritt 3. $n - 1$-Mal, bis nur noch ein Cluster vorhanden ist.

Das Gesamtergebnis des Verfahrens bis zu diesem Zeitpunkt ist in Abbildung 4.13 dargestellt. Dabei ist zu erkennen, dass bereits ein Großteil der Graph-Knoten klassifiziert wurden. Im letzten Schritt werden zur Verfeinerung des Ergebnisses die restlichen Knoten nochmals gesondert betrachtet. Die bereits klassifizierten Graph-Knoten werden im Kontext der kontinuierlichen Verbesserung (Multi-Pass) berücksichtigt.

4.2.3.4 Verfeinerung der Ergebnisse

Nach den vorherigen Schritten ist bereits die Mehrheit der Graph-Knoten klassifiziert. Im letzten Schritt erfolgt die Verfeinerung der Ergebnisse mithilfe eines Entscheidungsbaums. Zu den Kriterien gehören zwei binäre Entscheidungsfunktionen d und s, die den Abstand unter Berücksichtigung der Modellfunktion f und der Steigung zwischen zwei aufeinander folgenden vertikalen Knoten $\mathbf{v}_u$ und $\mathbf{v}_{u+1}$ bewerten.

In Abbildung 4.14 ist das Ergebnis des Verfahrens dargestellt.

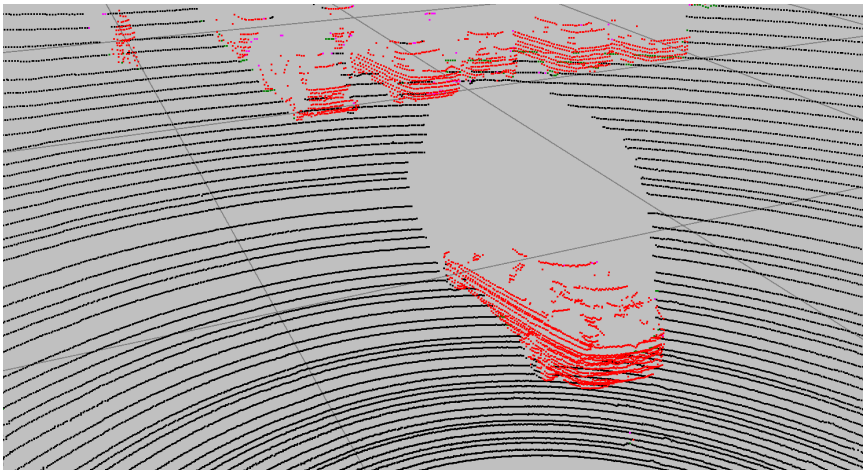


Abbildung 4.14:
Ergebnis des Verfahrens. Schwarze Punkte sind Messungen, die dem flachen Weltmodell entsprechen. Rot dargestellte Punkte verletzten die Modellvorstellung und tragen das Label *Hindernis*.

Die binäre Entscheidungsfunktion d gibt *Falsch* zurück, sofern eine ausreichende Distanzabweichung vom Muster gefunden wird:

$$d(\mathbf{v}_u, \mathbf{v}_{u+1}) = \begin{cases} \text{Falsch} & |r_u - f(u)| > d_{\text{t1}} f(u) \vee \\ & |r_{u+1} - r_u| > d_{\text{t2}} |f(u+1) - f(u)| \,, \\ \text{Richtig} & \text{andernfalls} \end{cases} \tag{4.12}$$

mit den Schwellenwerten d_{t1}, d_{t2}. Zusätzlich wird die Steigung bestimmt und mit einem Schwellenwert s_t verglichen:

$$s(\mathbf{v}_u, \mathbf{v}_{u+1}) = \begin{cases} \text{Richtig} & \text{Steigung}(\mathbf{v}_u, \mathbf{v}_{u+1}) < s_t \\ \text{Falsch} & \text{andernfalls} \end{cases}. \tag{4.13}$$

Der Schwellwert s_t ist abhängig von der zulässigen Steigung zwischen zwei vertikalen Graph-Knoten und verhindert insbesondere bei hochfrequenten Winkeländerungen, z. B. durch Nicken oder Wanken, eine fehlerhafte Klassifizierung.

Die Bedingungen der Zustandsübergänge sind in Tabelle 4.2 aufgelistet. Der Entscheidungsbaum hat insgesamt sechs verschiedene Zustände $\{A, \ldots, F\}$ und es werden jeweils nur die Graph-Knoten gewählt, deren Label $\{l_{u-1}, l_u, l_{u+1}\}$ den Eingangsbedingungen entsprechen. Der Zustandsübergang erfolgt durch die Prüfung von Bedingungen, die durch die Mesh-Graphen-Struktur vorgegeben sind und enden im END-Zustand.

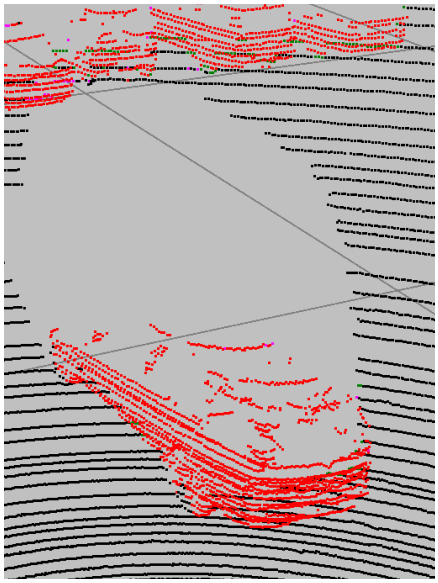
Tabelle 4.2:
Entscheidungsbaum für die Klassifizierung der N2 Nachbarschaft. Zur besseren Darstellung werden die Label mit ihrem Anfangsbuchstaben bezeichnet: *Freiraum* (f), *Hindernis* (h), *Unbekannt* (u).

Zustände \ Labels	l_{u-1}	l_u	l_{u+1}	$\hat{l}_{u-1}$	$\hat{l}_u$	$\hat{l}_{u+1}$	Bedingungen
A → END	f	h	h	h	h	h	$z_{u-1} > z_u$
B → END	h	f	h	h	h	h	$\neg d(\mathbf{v}_{u-1}, \mathbf{v}_u) \wedge \neg d(\mathbf{v}_u, \mathbf{v}_{u+1})$
B → END	h	f	h	h	u	h	andernfalls
C → A	f ∨ h	h	f	f ∨ h	h	h	$\neg s(\mathbf{v}_u, \mathbf{v}_{u+1}) \wedge d(\mathbf{v}_u, \mathbf{v}_{u+1})$
C → A	f ∨ h	h	f	f ∨ h	h	h	$s(\mathbf{v}_u, \mathbf{v}_{u+1}) < 0 \wedge \neg d(\mathbf{v}_u, \mathbf{v}_{u+1})$
D → C	h	f	f	h	h	f	$\neg d(\mathbf{v}_{u-1}, \mathbf{v}_u) \wedge \neg s(\mathbf{v}_{u-1}, \mathbf{v}_u)$
D → END	h	f	f	h	u	f	andernfalls
E → A	f	f	h	f	h	h	$\neg s(\mathbf{v}_{u-1}, \mathbf{v}_u) \wedge r_u < r_{u+1} \wedge \text{slope}(\mathbf{p}_u, \mathbf{p}_{u+1}) > 0$
F → C	f	f	f	f	h	f	$\neg s(\mathbf{v}_{u-1}, \mathbf{v}_u) \wedge \mid z_{u-1} - z_{u+1} \mid > 0.1$

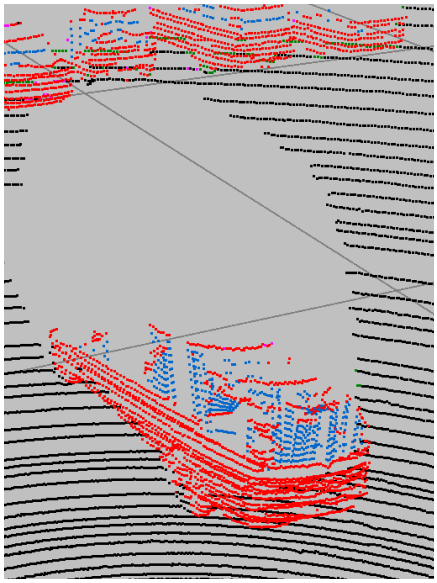
4.3 Erzeugung virtueller Messungen

Aufgrund von diffusen Reflexionen und spiegelnden Oberflächen finden nicht alle Laserstrahlen den Weg zurück zum LiDAR-Sensor. Fehlende Reflexionen, insbesondere an schwarzen Autos und Fahrzeugscheiben, führen zu fehlenden Entfernungsmessungen, was in Abbildung 4.15a zu sehen ist. Es entstehen Löcher in der Punktwolke bzw. dem Mesh-Graphen, was die stabile Form- und Richtungsschätzung erschwert.

Im Folgenden wird eine heuristische Methode vorgestellt, die die Position der fehlenden Graph-Knoten auf Basis des Sensormusters und der Mesh-Graphen Struktur approximiert. Die Repräsentation der fehlenden Messungen durch sogenannte virtuelle Messungen ermöglicht die Wiederherstellung der unvollständigen Nachbarschaftsbeziehung.



(a) Fehlende Messungen



(b) Virtuelle Graph-Knoten (blau)

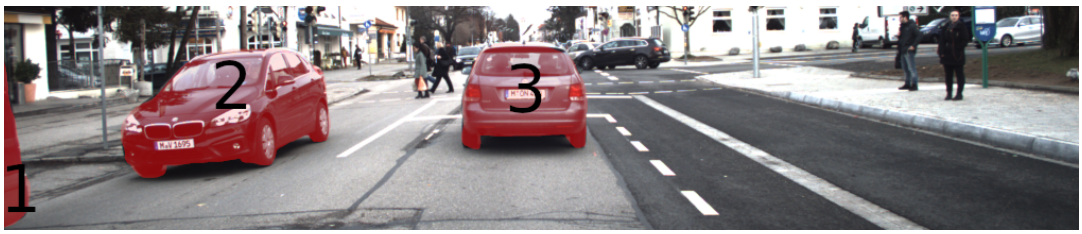
Abbildung 4.15:
Vergleich der Punktwolke vor dem Hinzufügen der virtuellen Messungen (a) sowie danach (b).

4.3.1 Verfahren

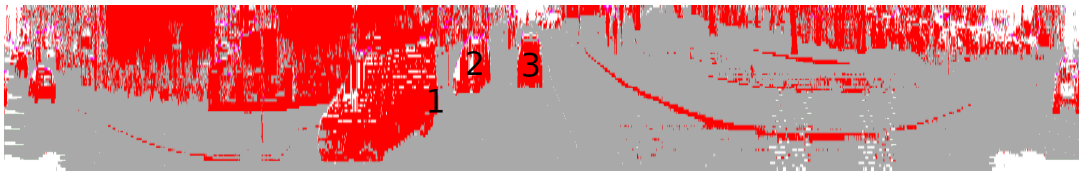
Das Ziel des Verfahrens ist, die 3D-Positionen der fehlenden Messungen zu approximieren, um so die Geometrie von Fahrzeugen stabiler erkennen zu können. Durch die Verwendung des Mesh-Graphen und die direkte Erzeugung der Graph-Knoten auf Basis der Sensorrohdaten, ist bereits im Dateneinzug bekannt, zu welchen Knoten keine Messung vorhanden ist. Diese Graph-Knoten tragen das Label *Ungültig*.

Um eine Gruppe von fehlenden Messungen im Mesh-Graphen zu detektieren wird ein Breitensuchalgorithmus verwendet. Bei der Breitensuche wird von einem Startknoten aus der gesamte Graph in die Breite nach einem Element durchsucht. Vom Startknoten aus wird jede Kante betrachtet und dabei getestet, ob der gegenüberliegende Knoten schon besucht wurde. Ist dies nicht der Fall, so wird der entsprechende Knoten einer Warteschlange hinzugefügt und im nächsten Schritt bearbeitet. Nachdem alle Nachbarschaften betrachtet wurden, wird der Prozess für den ersten Knoten der Warteschlange bearbeitet [Moore, 1959].

In Abbildung 4.15b ist das Ergebnis des Verfahrens dargestellt. Virtuelle Messungen sind mit der Farbe Blau gekennzeichnet, wobei für den maximalen Abstand zwischen den äußeren Kantenknoten $t_{d_{\text{max}}} = 1.3\,\text{m}$ gewählt wurde. Virtuelle Messungen sind in den folgenden Abbildungen bereits dem Mesh-Graphen hinzugefügt und als *Hindernis* gekennzeichnet.



(a) Kamerabild



(b) Spherische Darstellung des Mesh-Graphen

Abbildung 4.16:
Segmentierergebnis einer urbanen Verkehrsszene: (a) Kamerabild, (b) Ergebnis des Segmentierungsprozess dargestellt in einer Zylinderprojektion des Mesh-Graphen. Graph-Knoten mit dem Label *Hindernis* und *Freiraum* sind durch rote und graue Pixel dargestellt. Für weiße Pixel liegen keine Messdaten vor.

Die Ausführung der verschiedenen Schritte kann wie folgt beschrieben werden:

1. Erkennung von zusammengehörigen Graph-Knoten mit dem Label *Ungültig*, die von Graph-Knoten mit dem Label *Hindernis* umschlossen sind.
2. Überprüfung der Distanz aller Kanten-Knoten innerhalb einer vertikalen Spalte u, ob diese kleiner ist als der Schwellwert $t_{d_{\max}}$.
3. Überprüfung der Kriterien und falls erfüllt, lineare Approximierung der 3D-Positionen aller Knoten mit dem Label *Ungültig*. Hierzu wird eine ideale 3D-Linie zwischen den äußeren Kanten-Knoten gebildet und entlang der Geraden die virtuellen Messungen approximiert.
4. Wiederholung der Schritte 2 bis 4 bis alle Gruppen überprüft und die entsprechenden Knoten approximiert wurden.

Die vorgestellte Heuristik erlaubt eine deutlich verbesserte Datenassoziation der Graph-Knoten, was die Objektdetektion, insbesondere von schwarzen Autos, deutlich verbessert. Dies ist hauptsächlich auf die kombinierte Berücksichtigung der fehlenden LiDAR-Messungen im Dateneinzug sowie auf die gewählte Graphenstruktur zurückzuführen.

Das Gesamtergebnis der Hinderniserkennung ist für eine innerstädtische Verkehrsszene in Abbildung 4.16b gezeigt. Das Bild entspricht einer Zylinderprojektion des vollständigen Mesh-Graphen für den Sichtbereich des Sensors von $360°$. Die vertikale und horizontale Pixelauflösung ist proportional zu der Anzahl an Laserdioden pro Sensormuster und der Anzahl an Diodenmessungen innerhalb einer Umdrehung. Der Bildmittelpunkt entspricht dabei der Frontrichtung des Fahrzeugs. Rot gekennzeichnete Pixel entsprechen den als *Hindernis* klassifizierten Graph-Knoten und graue

Pixel den Freiraumknoten. Das Kamerabild einer Onboardkamera in Abbildung 4.16a zeigt die Verkehrsszene vor dem Fahrzeug.

Die in Abbildung 4.16a farblich gekennzeichneten Fahrzeuge entsprechen den in Abbildung 4.16b nummerierten Fahrzeugen (1-3). Das Fahrzeug mit der Nummer eins ist dabei kaum noch im Kamerabild sichtbar (linker Bildrand).

Auf Basis dieses Ausschnitts werden im Folgenden die weiteren Arbeitsschritte erklärt und visualisiert.

4.3.2 Freiraumflächenbestimmung

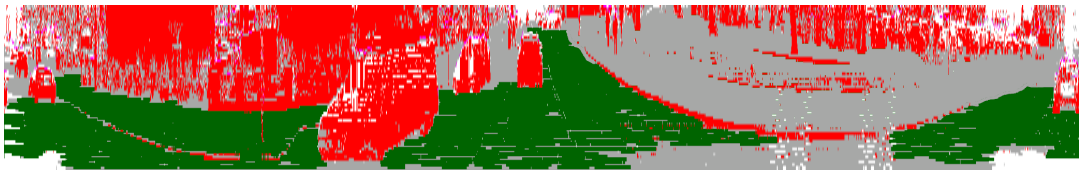


Abbildung 4.17:
Ergebnis der Freiraumflächenbestimmung. Hindernisse sind in Rot, zusammengehörige Freiraumflächen in Grün und Freiraum in Grau dargestellt.

Im Folgenden wird eine Methode zur Bestimmung von zusammengehörigen Graph-Knoten beschrieben, die eine Freiraumfläche bilden. Die Erzeugung der Freiraumflächen baut auf den Ergebnissen des mehrstufigen Segmentierungsprozesses auf, welcher in Unterabschnitt 4.2.3 beschrieben wurde.

Freiraumflächen sind definiert als Bereiche, die direkt vom Fahrzeug erreicht bzw. befahren werden können und bestehen aus einer Menge von Graph-Knoten. Bereiche, die beispielsweise durch Hindernisse versperrt werden, bilden eine Bodenfläche, jedoch keine Freiraumfläche. Zur Erzeugung der Freiraumflächen werden neben der eigenen Fahrzeugbreite Informationen über die Hindernisse der Umgebung berücksichtigt.

Um Freiraumflächen zu bestimmen, wird der Connected-Component Algorithmus von Wu et al. [2009] verwendet. Für jeden Graph-Knoten wird dabei die vertikale und horizontale N2 Nachbarschaft ausgewertet und eine Gewichtung bestimmt. Die horizontalen Kanten $\{\mathbf{e}_\mu^h\}_{\mu=1}^{2084} \in \mathbf{E}$ des Mesh-Graphen werden auf Basis des Winkelkriteriums $f(\mathbf{e}_\mu^h)$ überprüft und das Gewicht w_μ^h wie folgt bestimmt:

$$w_\mu^h = \begin{cases} f(\mathbf{e}_\mu^h) & \exists \mathbf{e}_\mu^h \in \mathbf{E} \\ 0 & \text{else.} \end{cases} \tag{4.14}$$

Die Gewichtung der vertikalen Kante $\mathbf{e}_\mu^v$ hingegen basiert auf der euklidischen Distanz $\|\mathbf{e}_\mu^v\|$ zwischen den Kanten-Knoten und dem Skalisierungsparameter λ_v:

$$w_\mu^v = \begin{cases} 1 - \|\mathbf{e}_\mu^v\|_2/(\lambda_v \|\mathbf{e}_\mu^v\|_2) & \exists \mathbf{e}_\mu^v \in \mathbf{E} \\ 0 & \text{else.} \end{cases} \tag{4.15}$$

Im letzten Schritt werden die Gewichte in einer Breitensuche für jede Graph-Knoten überprüft. Ist eine horizontale $w_\mu^h \leq \epsilon_h$ oder eine vertikale Diskontinuität $w_\mu^v \leq \epsilon_v$ erkannt, so wird ein neues Cluster erzeugt. Um die Freiraumflächen in Abhängigkeit der Fahrzeugbreite zu erzeugen, werden nur vertikale Komponenten verknüpft, wenn die maximale Distanz der äußeren Graph-Knoten pro Cluster kleiner ist als die gegebene Fahrzeugbreite. Eine weitere Annahme ist, dass Hindernisse Freiraumflächen begrenzen. Diese Betrachtungsweise vereinfacht den Prozess der Graph-Knotenassoziation deutlich, da einzelne Zweige der Breitensuche abgebrochen werden können, wenn ein Hindernisknoten erreicht wird. Dieser Prozess wird durchgeführt, bis jeder Graph-Knoten mindestens einmal überprüft wurde. Das Ergebnis ist eine Menge von l Freiraumflächen $\{\mathbf{a}_{k,l}\}_{l=1}^{m}$, die jeweils aus einer Menge an Graph-Knoten bestehen. Die Freiraumflächen sind in Abbildung 4.17 durch grüne Pixel dargestellt.

Im Umgebungsmodell sind die Freiraumflächen von hoher Bedeutung. Die Information über Freiraumflächen kann beispielsweise von der Pfadplanung verwendet werden, um die Trajektorie des Fahrzeugs zu planen oder als Vorabinformation für weitere Klassifizierungsaufgaben verwendet werden. Im objektbasierten Umgebungsmodell wird die Geometrie der Freiraumflächen durch Polygone beschrieben. Genauso wichtig sind für autonome Fahrzeuge die vertikale Objekte der Umgebung, die ein Hindernis darstellen. Im folgenden Abschnitt wird daher ein Verfahren vorgestellt, das auf Basis der vorsegmentierten Punktewolke Hindernisknoten zu zusammenhängenden Objekten bündelt und sogenannte Objektinstanzen erzeugt.

4.4 Objektinstanzen

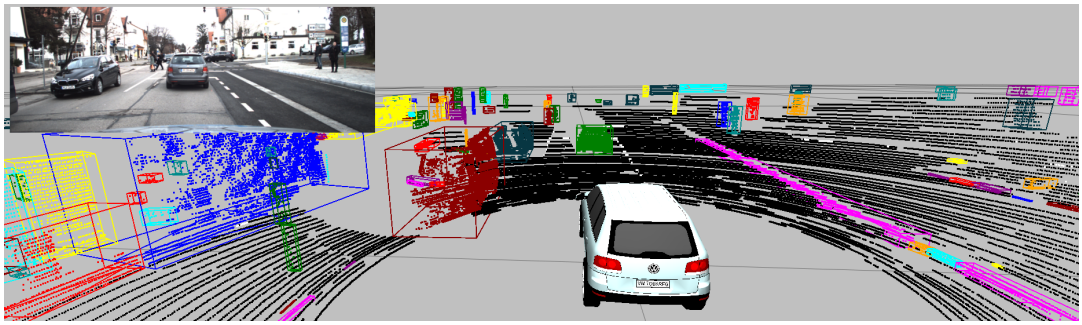


Abbildung 4.18:
Objektinstanzen als 3D-Darstellung. Bounding-Boxen beschreiben die ermittelte Objektdimension. Schwarze Punkte repräsentieren 3D-Messungen, die zu einer Bodenfläche gehören.

Das Ziel des Verfahrens ist eine robuste Extraktion von Objektinstanzen aus der Punktwolke. Eine Objektinstanz repräsentiert ein vertikales Objekt und kann durch eine Position x, y, z und den Parametern Länge l, Breite b und Höhe h beschrieben werden. In Abbildung 4.18 sind die Objektinstanzen durch farbige Bounding-Boxen und Punktwolken dargestellt.

Die Hauptaufgabe des hier vorgestellten Clusteringverfahrens ist, 3D-Punkte zu konsistenten Objektinstanzen zu gruppieren, damit diese später als Landmarken verwendet werden können. Das Verfahren soll gleichermaßen in unstrukturierter sowie strukturierter Umgebung eingesetzt werden können, ohne die Erzeugung und Verwendung von Trainingsdatensätzen. Im Allgemeinen steigt die Anforderung an die 3D-Objekterkennung, sobald das Fahrzeug die Autobahn oder die Innenstadt verlässt und die Umgebung unstrukturierter wird. Inhomogene Flächen sind dabei eine besondere Herausforderung, welche besonders bei Bäumen, Sträuchern, Schotterwegen und Wiesen mit flachem Gras auftreten. Darüber hinaus begrenzen Ausreißer, Messrauschen und Wettereinflüsse insgesamt die Leistung von punktwolkenbasierten Clusterverfahren.

Für eine schnelle und effektive Bearbeitung wird das Clustering der Graph-Knoten in ein vertikales und horizontales Problem aufgeteilt, in dem die vertikalen oder horizontalen Kanten des Mesh-Graphen getrennt betrachtet werden. Dies ermöglicht die parallele Verarbeitung und führt so zu einer Effizienzsteigerung.

In jedem Teilprozess werden dabei die Graph-Kanten und die dazugehörigen Knoten anhand von Regeln untersucht, die vom intrinsischen Sensormuster abgeleitet sind und auf der zeitlichen Abhängigkeit der Messungen basieren. Die Identifikation von zusammengehörigen Graph-Knoten erfolgt dabei mithilfe von *Scores*, die einen Wertebereich zwischen null und eins annehmen können und wie Wahrscheinlichkeiten behandelt werden. In einem finalen Fusionsschritt werden die Teilergebnisse wieder zu einem Gesamtergebnis zusammengeführt.

4.4.1 Vertikales Bottom-Up-Clusteringverfahren

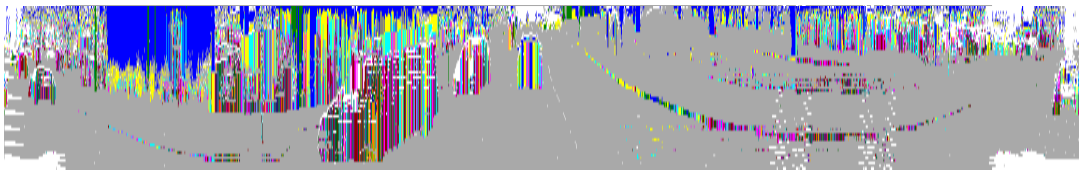


Abbildung 4.19:
Zylinderprojektion des Mesh-Graphen. Beim vertikalen Clusteringverfahren wird jede Spalte des Mesh-Graphen gesondert betrachtet. Das Ziel ist, in jeder Spalte die zusammengehörigen Graph-Knoten zu finden und zu gruppieren. Zusammengehörige Graph-Knoten sind mit der gleichen Pixelfarbe je Spalte dargestellt.

Der vertikale Clustering-Prozess ist eine Weiterentwicklung des in Sektion 4.2 vorgestellten Verfahrens mit dem Ziel, Cluster in den vertikalen Spalten des Graphen zu finden. In Abbildung 4.19 ist das Ergebnis mittels einer Zylinderprojektion des Mesh-Graphen dargestellt. Es baut auf dem Klassifizierungsergebnis des im vorherigen Abschnitt vorgestellten Verfahrens auf. Der Pseudocode der Methode ist in Algorithmus 4.1 beschrieben. Im Folgenden werden die einzelnen Schritte erläutert.

Zu Beginn wird für jeden Knoten einer vertikalen Spalte, der das Label *Hindernis* besitzt, ein eigenes Cluster $\mathbf{c}_{i=u} = \{\mathbf{v}_u\} \mid i \in \mathbf{U}$ erzeugt, siehe Zeile 3 Algorithmus

Algorithmus 4.1 : Vertikaler Bottom-Up-Clustering-Algorithmus. Die Berechnung erfolgt für jede vertikale Spalte des Mesh-Graphen getrennt. Die Abarbeitung wird parallel auf mehrere Rechnerkerne verteilt.

Result : Menge an Hindernis-Cluster $\mathbf{C}^v$ für jede vertikale Spalte v
Input : Menge an vertikalen Knoten $\mathbf{V}^v = \{\mathbf{v}_1, \ldots, \mathbf{v}_{64}\}$
Menge an Label-Knoten $\mathbf{L} = \{\}$;
Menge an Cluster $\mathbf{C}^v = \{\{\}\}$;
Clusterlabel Label=1;

1 **foreach** $\mathbf{v}_u \in \mathbf{V}^v \wedge l_u =$ Hindernis **do**
2 $\quad l_u^v \leftarrow$ Label ▷ Weise neues Label dem Cluster u zu
3 $\quad \mathbf{c}_i \leftarrow \{\mathbf{v}_u\}$ ▷ Initialisiere ein neues Cluster pro Knoten
4 $\quad \mathbf{C}^v \leftarrow \mathbf{C}^v \bigcup \mathbf{c}_i$ ▷ Füge Cluster dem Hindernis-Cluster $\mathbf{C}^v$ hinzu
5 $\quad$ Label $\leftarrow$ Label $+ 1$ ▷ Erhöhe Label um eins
6 **foreach** $\mathbf{c}_i \in \mathbf{C}^v \wedge \mathbf{c}_i \notin \mathbf{L}$ **do**
7 $\quad$ **foreach** $\mathbf{c}_j \in \mathbf{C}^v \wedge \mathbf{c}_j \notin \mathbf{L}$ **do**
8 $\quad\quad$ **if** $p(A \wedge B) > t_v$ **then**
9 $\quad\quad\quad l_j^v \leftarrow l_i^v$ ▷ Update Label j mit Clusterlabel i
10 $\quad\quad\quad \mathbf{c}_i \leftarrow \mathbf{c}_i \bigcup \mathbf{c}_j$ ▷ Update Cluster und berechne neue Position
11 $\quad\quad\quad \mathbf{L} \leftarrow \mathbf{L} \bigcup \{c_j\}$ ▷ Füge Cluster c_j zu Menge Label-Knoten hinzu.
12 $\quad\quad j = i + 1$
13 $\quad \mathbf{C}^v \leftarrow \mathbf{C}^v \bigcup \mathbf{c}_i$
14 $\quad i \leftarrow i + 1$

4.1. Ein Cluster besteht dabei aus einer Menge von Graph-Knoten und der Index i entspricht dem vertikalen Index u des initialen Graph-Knoten.

In den nächsten Schritten werden die Zusammengehörigkeiten von Clustern durch die Berechnung von zwei *Scores* A und B bestimmt, siehe Zeile 6 Algorithmus 4.1. Ist das Kriterium der Zusammengehörigkeit erfüllt, werden Cluster fusioniert und eine neue Cluster-Position $\mathbf{c}_i = \mathbf{c}_i \bigcup \mathbf{c}_j \mid j \in \mathbf{U}$ bestimmt, wobei gilt $i < j$, siehe Zeile 10 Algorithmus 4.1.

Ist das letzte Cluster der vertikalen Graphen-Spalte erreicht, startet der Algorithmus den gleichen Prozess mit dem nächsten Cluster, der nicht in der Menge der bereits gelabelten Cluster enthalten ist. Das Verfahren ist dem in Unterabschnitt 4.2.3 vorgestellten hierarchischen Clusteringverfahren ähnlich, jedoch wird auf die Berechnung der Distanzmatrix aus Effizienzgründen verzichtet. Zusätzlich ist das Clustern von Graph-Knoten abhängig von Bewertungsfunktionen den sogenannten *Scores*.

Im Folgenden wird die Berechnung der beiden *Scores* beschrieben. A ist der *Score*, der die Abhängigkeit von zwei Clustern berechnet:

$$A = \text{score}\left(d_2, k_a\right), \tag{4.16}$$

mit $d_2\left(\mathbf{c}_i^{\text{com}}, \mathbf{c}_j^{\text{com}}\right)$ als die zweidimensionale euklidische Distanz zwischen den Cluster-Zentren (engl. center of mass). Die Koordinaten der Cluster-Zentren werden über

den Mittelwert aller 3D-Koordinaten, die zu einem Cluster gehören, berechnet: $\mathbf{c}^{\text{com}} = {}^{1}/_{N} \sum_{i=0}^{N} \mathbf{c}_i$. Der Parameter k_a parametrisiert die Steigung der Kurve und erlaubt die Gewichtung in Abhängigkeit der Distanz. Die *Score*-Funktion selbst ist wie folgt definiert:

$$\text{score}\,(x, k) = 1 - \frac{|x|}{k + |x|}. \tag{4.17}$$

Das Ergebnis liegt dabei im Wertebereich zwischen $[0, 1[$ und gibt eine Art Likelihood an, wie stark die beiden Cluster zusammengehören. Die Bewertungsfunktion ist eine Approximation der hyperbolischen Tangentenfunktion. Die hyberbolische Tangentenfunktion wird sehr oft als sogenannte Aktivierungsfunktion in neuronalen Netzen verwendet, um das Ergebnis eines Neurons zu bewerten.

Im Vergleich zu der originalen hyperbolischen Tangentenfunktion $\tanh x = {}^{\sinh x}/_{\cosh x}$, erlaubt die Approximation eine deutlich schnellere Berechnung, da Funktionsaufrufe der Winkelfunktionen sehr teuer sind.

Der Score B bewertet den Höhenunterschied zwischen zwei Clustern unter Berücksichtigung der intrinsischen Diodenanordnung. Ist die Höhe des Clusters $c_i^{z,\text{com}}$ größer als das nachfolgende Cluster $c_j^{z,\text{com}}$, wird die Beziehung zwischen den Clustern mit einem niedrigen *Score* bewertet, da nicht davon auszugehen ist, dass es sich um ein zusammengehöriges Cluster handelt. Ist die Bedingung jedoch nicht erfüllt, wird die absolute Höhendifferenz der Clustermittelpunkte bewertet. Der *Score* B berechnet sich dabei wie folgt:

$$B = \text{score}\left(w_h(c_i^{z,\text{com}}, c_j^{z,\text{com}}), \tilde{k}_b\right), \tag{4.18}$$

unter Berücksichtigung der Gewichtungsfunktion:

$$w_h\left(c_i^{z,\text{com}}, c_j^{z,\text{com}}\right) = \begin{cases} |c_i^{z,\text{com}} - c_j^{z,\text{com}}| & c_i^{z,\text{com}} < c_j^{z,\text{com}} \\ +\infty & else \end{cases}. \tag{4.19}$$

Um die Höhendifferenz zwischen Clustern in Abhängigkeit der Distanz zu gewichten, erfolgt die Parameterbestimmung der Kurvensteigung $\tilde{k}_b$ anhand des euklidischen Abstands zum Clustermittelpunkt $c_i^{z,\text{com}}$ sowie dem Elevationswinkel zum Clustermittelpunkt $\arctan {}^{c_i^{y,\text{com}}}/_{c_i^{x,\text{com}}}$:

$$\tilde{k}_b = k_b + \|\mathbf{c}_i^{\text{com}}\| \cdot \sin\left(\arctan {}^{c_i^{y,\text{com}}}/_{c_i^{x,\text{com}}}\right). \tag{4.20}$$

Die Scorekurve wird über die Distanz der Messung geringer. Dies ermöglicht insbesondere bei Fahrten in unstrukturiertem Gelände und starken Nickwinkeländerungen falsche Datenassoziation zu reduzieren.

Um den finalen *Score* zu bestimmen, werden die beiden *Scores* A und B wie zwei absolut unabhängige Wahrscheinlichkeiten multipliziert:

$$A \wedge B = A \cdot B. \tag{4.21}$$

Je näher der *Score* sich dem Maximalwert von eins nähert, desto zuversichtlicher ist das Modell, dass die beiden Cluster zum gleichen Objekt gehören. Der Schwellwert t_v definiert die Grenze, bei dem ein Clusterpaar als zusammenhängend oder getrennt zu klassifizieren ist.

4.4.2 Horizontales Clusteringverfahren

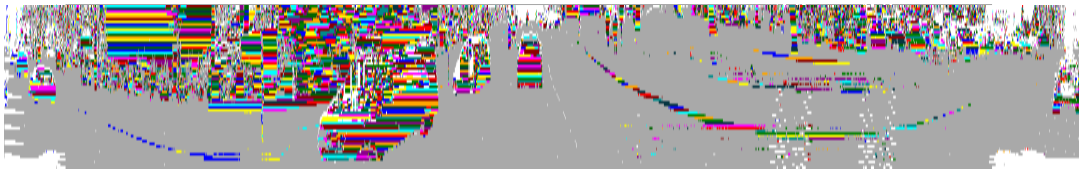


Abbildung 4.20:
Zylinderprojektion des Mesh-Graphen. Beim horizontalen Clusteringverfahren wird jede Zeile des Mesh-Graphen gesondert betrachtet. Das Ziel ist in jeder Zeile die zusammengehörigen Graph-Knoten zu finden und zu gruppieren. Zusammengehörige Graph-Knoten sind mit der gleichen Pixelfarbe je Zeile dargestellt.

Vergleicht man die horizontalen Kanten der Graph-Knoten mit den vertikalen Kanten des Mesh-Graphen, so sind die Graph-Knoten pro Kante deutlich homogener. Der Grund dafür ist, dass direkt aufeinander folgende horizontale Distanzmessungen zeitlich sehr nahe beieinander liegen und die Eigenbewegungskompensation nur einen sehr geringen Einfluss hat. Dies ermöglicht, im Vergleich zum vertikalen Verfahren, die Verwendung einer weniger aufwendigen Methode.

Um die Zusammengehörigkeit von horizontalen Graph-Knoten zu bestimmen, wurde ein weiteres Verfahren entwickelt, das neben der euklidischen Distanz auch verschiedene Winkel zwischen den Graph-Knoten berücksichtigt. Die Ausführung des Algorithmus erfolgt dabei pro horizontale Zeile und parallel auf mehrere Prozessorkerne verteilt.

Wie bereits im vertikalen Clusteringverfahren werden *Scores* bestimmt, die ein Maß für die Zusammengehörigkeit von Graph-Knoten geben. Die Berechnung erfolgt ebenfalls unter Berücksichtigung des intrinsischen Sensormusters unter Annahme des idealen Weltmodells. Der *Score* C berücksichtigt dabei die dreidimensionale euklidische Distanz zwischen zwei Graph-Knoten $(\mathbf{v}_v, \mathbf{v}_w)$, $v, w \in \mathbb{N} \mid v < w$:

$$C = \text{score}\left(\left\|\mathbf{v}_v - \mathbf{v}_w\right\|, k_c\right). \tag{4.22}$$

Zusätzlich wird die Annahme getroffen, dass drei benachbarte Graph-Knoten aus der gleichen horizontalen Spalte zum gleichen Cluster gehören, wenn sie auf einer virtuellen Linie liegen. Die gesuchten Winkel und die dazugehörigen Graph-Knoten sind in Abbildung 4.21 visualisiert. Der winkelabhängige *Score* D bestimmt sich anhand des Winkels $\beta_1 = \angle(\overrightarrow{\mathbf{v}_{i+1}\mathbf{v}_i}, \overrightarrow{\mathbf{v}_{i+1}\mathbf{v}_{i+2}})$ wie folgt:

$$D = \text{score}\left(\left\|\pi - \beta_1\right\|, k_d\right). \tag{4.23}$$

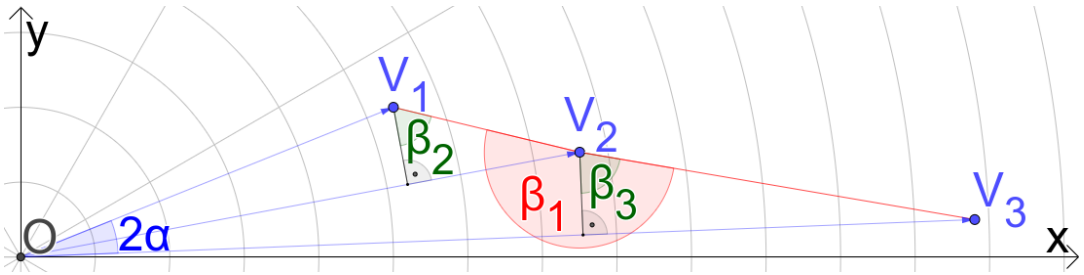


Abbildung 4.21:
Die Abbildung zeigt die Winkel β_1, β_2, β_3, wobei α der Azimutwinkel zwischen zwei aufeinander folgenden Messungen desselben Strahls ist. O stellt den Sensorursprung dar. V_1, V_2, V_3 beschreiben die horizontale Position der Knoten in 2D und rote Linien visualisieren die Graph-Kanten.

Die Bedingung $p\left(C \wedge D, \beta_1\right) > t_h$ ist nicht immer ausreichend und so wird in Abhängigkeit der *Scores* E_{β_2} und E_{β_3} ein verfeinertes und partielles Clustern durchgeführt. Die Winkel β_2, β_3 werden unter Berücksichtigung der minimalen und maximalen Entfernungsmessung $r_{2,\min}, r_{2,\max} = \texttt{minmax}\left(||\mathbf{v}_i||, ||\mathbf{v}_{i+1}||\right)$ zweier aufeinander folgenden horizontalen Graph-Knoten sowie dem Azimutwinkel α_h bestimmt [Bogoslavskyi und Stachniss, 2016]:

$$\beta_{\{2,3\}} = \arctan \frac{r_{\min} \sin \alpha_h}{r_{\max} - r_{\min} \cos \alpha_h}. \tag{4.24}$$

Die entsprechenden *Scores* berechnen sich dann in Abhängigkeit der Winkel β_2 und β_3 wobei gilt:

$$E = \frac{1}{1 + e^{-(k_e - x)}}. \tag{4.25}$$

Der vollständige Pseudocode der horizontalen Clustering-Methode ist in Algorithmus 4.2 angegeben und wird im Folgenden beschrieben.

Zu Beginn werden die ersten drei Graph-Knoten einer Spalte mit aufsteigendem vertikalem Index u gewählt $\{\mathbf{v}_u, \mathbf{v}_{u+1}, \mathbf{v}_{u+2}\}$. Die Auswahl der ersten Graph-Knoten innerhalb einer Zeile erfolgt dabei nach dem Zufallsprinzip. Durch die Auswahl von zufälligen Startpunkten wird einem deterministischen Clusterergebnis entgegengewirkt und das Gesamtergebnis deutlich verbessert. In den darauf folgenden Schritten werden kontinuierlich alle weiteren Graph-Knoten entlang der Spalte nach dem gleichen Prinzip ausgewertet, bis der Start-Knoten wieder erreicht ist. Das Ergebnis der horizontalen Clusterbildung einer Szene ist in Abbildung 4.20 gezeigt.

4.4.3 Fusion der vertikalen und horizontalen Ergebnisse

Im letzten Schritt werden die Teilergebnisse des vertikalen und horizontalen Verfahrens fusioniert [Burger et al., 2018]. Das Ziel ist, die in Abbildung 4.19 und Abbildung 4.20 gezeigten Teilergebnisse zu vereinen. Als Ausgangspunkt werden dabei die Graph-Knoten der vertikalen Cluster herangezogen und die Übereinstimmung der vertikalen und horizontalen Nachbarn ausgewertet.

Algorithmus 4.2 : Pseudocode des horizontalen Clusteringverfahrens. Die Ausführung erfolgt pro vertikale Reihe u des Mesh-Graphen.

Result : Set $\mathbf{L}$ of all labeled horizontal vertices.
Input : $\mathbf{V}^h = \{\mathbf{v}_1, \ldots, \mathbf{v}_{2084}\}$; $i = 1$, $\texttt{Label} = 1$

foreach $\{\mathbf{v}_i, \mathbf{v}_{i+1}, \mathbf{v}_{i+2}\} \in \mathbf{V}^h$ **do**
- $\beta_1 = \angle(\overrightarrow{\mathbf{v}_{i+1}\mathbf{v}_i}, \overrightarrow{\mathbf{v}_{i+1}\mathbf{v}_{i+2}})$ ▷ Berechne Winkelkriterium β_1
- $\mathit{Bedingung}_1 = p\,(C \wedge D, \beta_1) > t_h$ ▷ Überprüfe erste Bedingung
- $r_{2,\min}, r_{2,\max} = \texttt{minmax}\,(||\mathbf{v}_i||, ||\mathbf{v}_{i+1}||)$
  ▷ Berechne min/max Distanz der Knoten $i, i+1$
- $\beta_2 = \arctan \frac{r_{2,\min} \sin \alpha}{r_{2,\max} - r_{2,\min} \cos \alpha}$ ▷ Berechne Winkelkriterium β_2
- $\mathit{Bedingung}_2 = p\,(C) > t_h \wedge \beta_2 >= t_{ha}$ ▷ Überprüfe zweite Bedingung
- $r_{3,\min}, r_{3,\max} = \texttt{minmax}\,(||\mathbf{v}_{i+1}||, ||\mathbf{v}_{i+2}||)$ ▷ s.o.
- $\beta_3 = \arctan \frac{r_{3,\min} \sin \alpha}{r_{3,\max} - r_{3,\min} \cos \alpha}$ ▷ s.o.
- $\mathit{Bedingung}_3 = p\,(C \wedge D, \beta_3) > t_h$ ▷ Überprüfe dritte Bedingung
- **if** $\text{Bedingung}_1 \vee (\text{Bedingung}_2 \wedge \text{Bedingung}_3)$ **then**
  - $\{l_i^h, l_{i+1}^h, l_{i+2}^h\} \leftarrow \texttt{Label}$ ▷ Alle Graph-Knoten erhalten das gleiche Label
- **else if** Bedingung_2 **then**
  - $\{l_i^h, l_{i+1}^h\} \leftarrow \texttt{Label}$ ▷ Knoten $\{i, i+1\}$ erhalten das gleiche Label
  - $\texttt{Label} \leftarrow \texttt{Label} + 1$ ▷ Neues Label erzeugen
  - $l_{i+2}^h \leftarrow \texttt{Label}$ ▷ Knoten $i+2$ erhält neu erzeugtes Label
- **else if** Bedingung_3 **then**
  - $\texttt{Label} \leftarrow \texttt{Label} + 1$
  - $\{l_{i+1}^h, l_{i+2}^h\} \leftarrow \texttt{Label}$ ▷ Knoten $\{i+1, i+2\}$ erhalten das gleiche Label
- **else**
  - $l_i^h = \texttt{Label}$ ▷ Initialisiere erstes Label des Tripel
- $i \leftarrow i + 2$ ▷ Erhöhe horizontalen Graph-Knoten Index um 2

Das Gesamtergebnis des Verfahrens ist in Abbildung 4.22 dargestellt, wobei gleichfarbige Pixel zusammengehörige Graph-Knoten (Cluster) kennzeichnen.

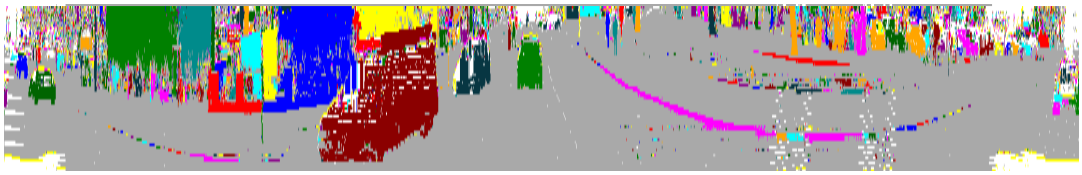


Abbildung 4.22:
Gesamtergebnis des Clusteringverfahrens. Benachbarte Pixel mit der gleichen Farbe gehören zur gleichen Objektinstanz. Graph-Knoten mit dem Label *Ungültig* und *Freiraum* haben die Farbe Weiß und Grau.

Während des Fusionsprozesses werden die Ergebnisse des vertikalen- und horizontalen Clusteringverfahrens für jeden Graph-Knoten ausgewertet. Der Algorithmus basiert ebenfalls auf einer Breitensuche, bei der in jedem Schritt die Label der direkten Nachbarn auf Übereinstimmung überprüft werden. Stimmen die Label des vertikalen $l_{u\pm1,v}^v$

und horizontalen $l^h_{u,v\pm1}$ Nachbarn mit dem des Graph-Knoten $l_{u,v}$ überein, werden die Label $l_{u\pm1,v}$ und $l_{u,v\pm1}$ der Nachbarknoten aktualisiert und auf das Cluster-Label $l_{u,v}$ gesetzt. Darüber hinaus wird der Nachbarschaftsknoten ebenfalls zur Warteschlange $\mathbf{Q}$ hinzugefügt, damit im nächsten Schritt alle weiteren Nachbarschaften überprüft werden können. Dieser Prozess erfolgt so lange, bis alle Nachbarschaftsknoten maximal zweimal besucht wurden. Der Pseudocode des Verfahrens ist in Algorithmus 4.3 beschrieben.

In Abbildung 4.23 wird die Auswertung am Graph-Knoten mit der Nummer fünf (in der Mitte) exemplarisch dargestellt. Die Ergebnisse des vertikalen- und horizontalen Verfahrens sowie der Fusion sind als einzelne Layer dargestellt (V, H ,R). Der Knoten fünf besitzt dabei in seiner vertikalen Nachbarschaft die Knoten zwei und acht. In der horizontalen Nachbarschaft befinden sich die Knoten vier und sechs. Die Knoten mit der Nummer eins, zwei, drei und sieben gehören aufgrund fehlender vertikaler oder horizontaler Kanten nicht zum gleichen Cluster.

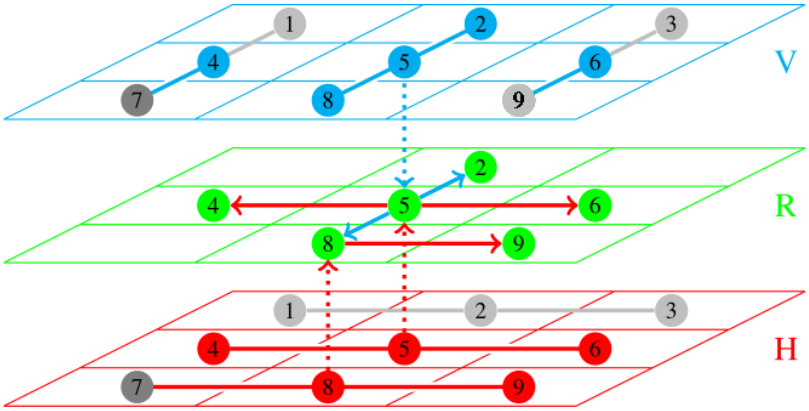


Abbildung 4.23:
Diese Abbildung veranschaulicht den Fusionsprozess der Teilergebnisse aus dem horizontalen und vertikalen Clusterverfahren. Die vertikale (V, in Cyan) und horizontale (H, in Rot) Struktur sowie das fusionierte Ergebnis (R, in Grün) sind als Schichten dargestellt, wobei jede Gitterzelle ein Knoten im Graphen darstellt. Im Schaubild symbolisieren gleiche Farben der Kanten sowohl im horizontalen und vertikalen Ergebnislayer die Zusammengehörigkeit der Knoten. Gestrichelte Linien visualisieren den Übergang der Ergebnisse von jeder Schicht auf die Gesamtergebnisschicht.

Algorithmus 4.3 : Pseudocode der Breitensuche zur Fusion des vertikalen und horizontalen Clusteringergebnis. Das Ergebnis der vertikalen und horizontalen Clusterung wird kombiniert, um dreidimensionale Objekt-Instanzen zu extrahieren.

```
1  Function ClusterFusionBreitensuche(V, C_u^v):
       Result : Menge L aller Graph-Knoten mit einem Clusterlabel
       Data : L = {}; Q = {}; Label = 1; Label ∈ ℕ
2      foreach C^v ∈ {C^0, ..., C^v_max} do
3          foreach v_{u,v} ∈ C^v | v_{u,v} ∉ L do
4              Q = Q ⋃ v_{u,v}                          ▷ Füge Cluster zu Q
5              while Q not empty do
6                  v_{u,v} ← Q.front()                  ▷ Nehme erstes Element in der Liste
7                  l_{u,v} ← Label;
8                  L = L ⋃ {v_{u,v}}                    ▷ Label
9                  NLabeling(Q, {v_{u±1,v}}, l^v_{u,v}, Label)
10                 NLabeling(Q, {v_{u,v±1}}, l^h_{u,v}, Label)
11                 Q.popfront()                         ▷ Entferne erstes Element
12             Label ← Label + 1                        ▷ Erzeuge neues Label
13 Function NLabeling(Q, N, l_ref, Label):
14     foreach v_{u,v} ∈ N ∧ v_{u,v} ∉ L do
15         if l^{v,h}_{u,v} = l_ref then
16             l_{u,v} ← Label;
17             L = L ⋃ {v_{u,v}}
18             Q = Q ⋃ {v_{u,v}}                        ▷ Füge Knoten der Warteschlange hinzu
```

4.4.4 Generierung von Objektinstanzen

Im letzte Schritt werden die Objektinstanzen auf Basis der Graph-Knoten erzeugt. Um die Position sowie die Länge, Breite und Höhe für jede Instanz bestimmen zu können, müssen zuerst die Graph-Knoten mit dem gleichen Label $l_{u,v}$ extrahiert werden.

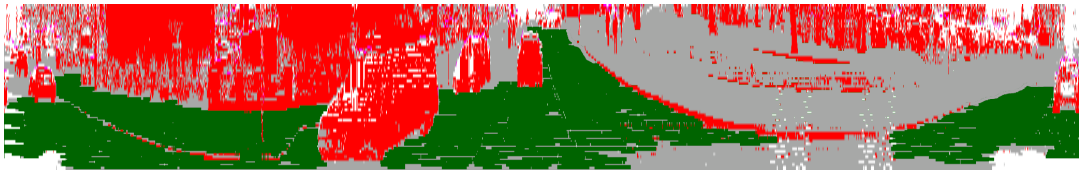
In Abbildung 4.24e sind die Label $l_{u,v}$ aller Graph-Knoten noch einmal farblich dargestellt, wobei benachbarte Pixel mit der gleichen Farbe zur gleichen Objektinstanz gehören. Mit dem Verfahren von Naujoks und Wuensche [2018] können dann auf Basis der 3D-Positionen aller Graph-Knoten einer Instanz die Bounding-Box und Position bestimmt werden. Das Ergebnis der Szene ist in Abbildung 4.24f zu sehen. Die Punktwolke ist farblich gekennzeichnet und die Bounding-Boxen entsprechend eingezeichnet.

Für eine bessere Übersichtlichkeit sind die in diesem Kapitel vorgestellten Teilergebnisse in der Zylinderprojektionsdarstellung für den Mesh-Graphen nochmals chronologisch in Abbildung 4.24 dargestellt.

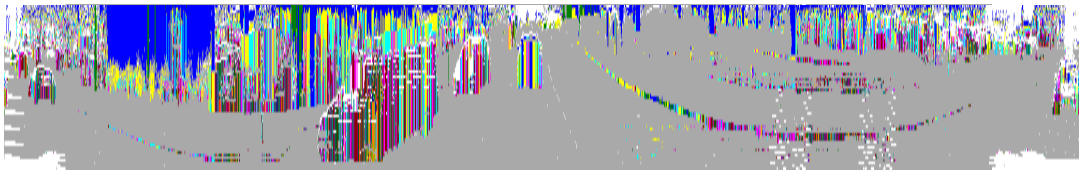
Zusammengefasst ist zu erwähnen, dass die in den vorherigen Abschnitten vorgestellte Verfahren ohne rekursive Filterung erfolgen. Diese Designentscheidung wurde bewusst getroffen. Pro Zeitschnitt können somit sehr schnell sowie unabhängig von weiteren Sensoren und Messgrößen Freiraumflächen und Objektinstanzen in der Punktwolke gefunden werden.

Die rekursive Schätzung der Umgebung erfolgt nach der Vorverarbeitung basierend auf einem Multi-Target Tracking (MTT)-Verfahren, das als Grundlage eine Objektliste verwendet, in welches die Objektinstanzen eingefiltert werden.

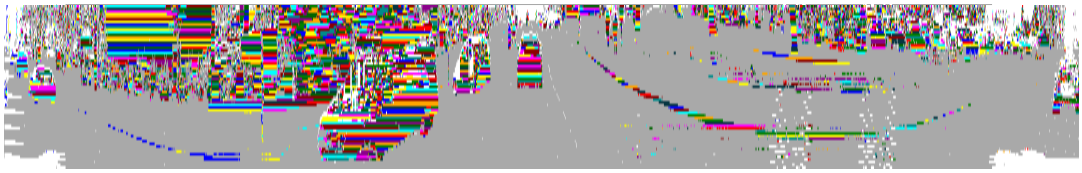
(a) Kamerabild einer urbanen Verkehrsszene



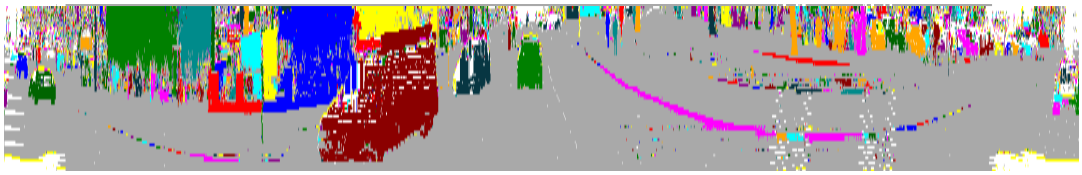
(b) Hindernisse (Rot), Freiraumflächen (Grün), Freiraum (Grau)



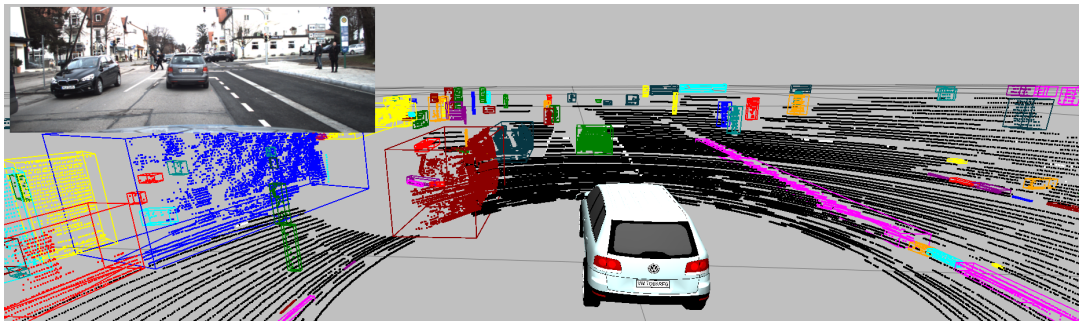
(c) Vertikales Segmentierungsergebnis



(d) Horizontales Segmentierungsergebnis



(e) Fusion der horizontalen und vertikalen Ergebnisse



(f) Objektinstanzen als 3D-Darstellung

Abbildung 4.24:
Segmentierergebnisse: (a) Kamerabild, (b) Punktlabelergebnis: Hindernisse (Rot), Freiraumflächen (Grün) (c)...(e) Segmentierungsschritte, (f) Segmentierungsergebnis in der dreidimensionalen Punktwolke. Objektinstanzen sind in Farbe und Freiraumpunkte sind in Schwarz dargestellt.

5 Kooperatives SLAM in unstrukturierter Umgebung

Inhalt

5.1 Einführung

Um die genannten Forschungsziele aus Unterabschnitt 1.2.1 zu erreichen, wurden im Zuge dieser Arbeit eine Reihe von neuen Methoden entwickelt, die im Zusammenspiel die kooperative Lokalisierung und Kartierung in unstrukturierter Umgebung ermöglichen. Im Kontext des aufgesplitteten Konvois wird diese Funktionalität erprobt und ausgewertet.

Bei dem in dieser Arbeit entwickelten SLAM-Verfahren wird eine Monte Carlo Lokalisierung (MCL) mit neuartigem Messmodell, ein Multi-Target Tracking (MTT)-Verfahren zur Landmarkenbestimmung sowie eine Straßenverlaufsschätzung probabilistisch in einem Graph-SLAM Ansatz fusioniert, um simultan eine Karte zu erstellen und sich dabei gleichzeitig global in einer Karte zu lokalisieren.

Die Datengrundlage des Verfahrens liegt in der Verwendung einer spärlichen Landmarkenkarte (engl. Sparse Feature Map), in der Landmarken als 3D Positionen mit Höhe, Länge, Breite sowie die Landmarkenklasse abgespeichert werden. Mithilfe einer rekursiven Zustandsschätzung werden vertikale Objekte und der aktuelle Straßenverlauf erfasst, als Landmarke identifiziert und dabei die benötigte Datenassoziation zwischen Landmarkenobjekten für das SLAM-Verfahren gelöst. Zusätzlich werden die Informationen über die Zustände bewertet, um robuste Landmarken zu identifizieren.

Zur Speicherung und Übermittlung der Karteninformationen zwischen den Fahrzeugen wird darüber hinaus ein einheitliches und im Hinblick auf den Speicherbedarf sparsames Datenmodell verwendet. Dies ist ebenfalls eine wichtige Voraussetzung für den aufgesplitteten Konvoi, damit das vorausfahrende Fahrzeug seinen gefahrenen Pfad inklusive erkannter und kartierter Landmarken an die Folgefahrzeuge über schmalbandigen Funk übermitteln kann. Zusätzlich kann das vorausfahrende Fahrzeug bei der Kartierungsfahrt vorhandene Karteninformationen wie z. B. eine OpenStreetMap (OSM) nutzen, um die selbst erstellte Karte im globalen Kartenraum zu registrieren.

Den Folgefahrzeugen steht nach erfolgreicher Übertragung eine aktualisierte und bei Bedarf erweiterte Karte zur Verfügung. Diese Karte beinhaltet neben dem Pfad und den Landmarken auch Informationen über die Zustandsunsicherheit. Diese Informationen können wiederum von den Folgefahrzeugen als Vorabwissen im eigenen SLAM-Verfahren integriert und fusioniert werden. Die Gewichtung der Sensorlandmarken und Kartenlandmarken erfolgt dabei unter Berücksichtigung verschiedener Faktoren, wie z. B. der Zustandsunsicherheit der Schätzprozesse.

5.2 Stand der Technik

Im Folgenden werden aktuelle Verfahren vorgestellt, die Ähnlichkeiten und Überschneidungen mit dem in dieser Arbeit vorgestellten Verfahren aufweisen. Der Fokus liegt dabei auf Methoden, die ebenfalls für den alleinigen bzw. teilweisen Einsatz in unstrukturierten Bereichen entwickelt wurden und zusätzliches Kartenmaterial zur Lokalisierung berücksichtigen.

Im Allgemeinen gilt, dass SLAM-Verfahren in unstrukturierter Umgebung besonderen Herausforderungen ausgesetzt sind, denn im direkten Vergleich zu urbanen bzw. suburbanen Szenarien fehlen oftmals einfach identifizierbare und stabile Landmarken zur Lokalisierung wie beispielsweise Straßenmarkierungen, Schilder, Laternen, Leitpfosten und Gebäude [Albrecht et al., 2020, Burger et al., 2019a, Caltagirone et al., 2018, 2017, Fassbender et al., 2014, Fox et al., 1999, Kim et al., 2016a, Manz

et al., 2010, Montemerlo et al., 2008, 2002, Moosmann et al., 2009, Neuhaus et al., 2009, Schneider et al., 2010, Thrun, 2003, 2006, Urmson et al., 2009a, Zhang et al., 2015].

Die genannten Merkmale bilden jedoch aktuell die Grundvoraussetzung für eine hochgenaue Lokalisierung sowie den in Kapitel 1 dargestellten Fahrfunktionalitäten. Die fünf Stufen des autonomen Fahrens und die geforderte Genauigkeit sind somit aktuell noch nicht auf das unstrukturierte Gelände übertragbar.

Darüber hinaus wirken sich die höheren Anforderungen an die Umgebungswahrnehmung auch proportional auf die Anschaffungskosten der notwendigen Sensorik aus, die neben einer hohen Auflösung auch einen hohen Sichtbereich und eine weite Messdistanz erfordern [Burger et al., 2019a, Manz et al., 2010, Urmson et al., 2009a].

Besonders Verfahren der visuellen Ortsbestimmung mit Kamerabildern, wie von Jaspers [2021] vorgestellt, gelten als nicht langzeitstabil, da sich das Erscheinungsbild sowie die Geometrie von Vegetationslandmarken in Abhängigkeit der unterschiedlichen Jahreszeiten in Mitteleuropa deutlich verändern kann. Das Gleiche gilt für die Lokalisierung in metrischen Occupancy Grid Maps (OGM), da sich die Belegtheit der Gridzellen sehr stark in Abhängigkeit des Erscheinungsbildes von Bäumen, Sträuchern und Büschen ändern kann.

Die Herausforderung in der sensorbasierten Umgebungswahrnehmung sowie der anschließenden Datenassoziation zur Lokalisierung ist somit in unstrukturierten Bereichen als sehr komplex anzusehen, insbesondere wenn es darum geht, Kartenmaterial zu erstellen und über längere Zeiträume zur Lokalisierung zu verwenden.

Um sich global in einer Karte, innerhalb von Gebäuden und in Außenumgebungen zu lokalisieren, verwendet der Ansatz von Kümmerle et al. [2010] öffentlich zugängliche Luftbilder. Um den durch die Luftbilder gegebenen Prior im Graph-SLAM Backend zu integrieren, wird eine MCL mit einem neuartigen Sensormodell vorgestellt. Das Modell berücksichtigt LiDAR-Punktwolken und gleicht diese mit extrahierten Gebäudeumrissen aus den Luftbildern ab. Auf diese Weise erreicht der Ansatz globale Konsistenz, ohne dass Schleifen geschlossen werden müssen.

Einer der ersten Ansätze, der das Straßennetz von OpenStreetMap (OSM) zur Fahrzeuglokalisierung verwendet, stammt von Floros et al. [2013]. Das vorgestellte Verfahren gleicht die Fahrzeugtrajektorie, die auf Basis visueller Odometrie ermittelt wurde, mit dem Straßennetzwerk von OSM ab, um eine robuste und genaue Schätzung der Fahrzeugposition zu erhalten. Der Hauptbeitrag besteht dabei in der Einbindung der Kartendaten als zusätzliche Information in das Beobachtungsmodell der MCL. Zusätzlich ist der Ansatz in der Lage, den Odometriefehler der visuellen Odometrie zu kompensieren, wodurch die Qualität der Lokalisierung deutlich verbessert werden kann.

Das Verfahren von Ruchti et al. [2015] löst das Lokalisierungsproblem im globalen Raum durch die Berücksichtigung des Straßennetzes von OSM. Hierzu wird ein Grid-basiertes Klassifizierungsverfahren vorgestellt, um Straßen von Nicht-Straßen

in den 3D-LiDAR-Punktwolken zu identifizieren. Die Lokalisierung erfolgt ebenfalls mit einer MCL. Im Vergleich zu anderen Ansätzen ermöglicht das Verfahren die Lokalisierung des Roboters, ohne direkt dem Straßenverlauf zu folgen.

In der Arbeit von Brubaker et al. [2016] werden die auf Basis von zwei Videokameras berechnete visuelle Odometrie und eine Straßenkarte dazu verwendet, das Fahrzeug global in einer großen Karte von mehreren $100\,\mathrm{km}$ Straßennetzwerk zu lokalisieren. Der Kern der Methode ist ein probabilistisches Modell, für das ein effizienter Inferenzalgorithmus vorgestellt wird. Dieser ist in der Lage, mit den gegebenen Unsicherheiten und dem Rauschen der visuellen Odometrie sowie den inhärenten Mehrdeutigkeiten der Karte umzugehen.

Das Verfahren von Vysotska und Stachniss [2016] erweitert ein Pose-Graph SLAM-Verfahren um globale Positionsbedingungen, die auf Basis von Gebäudeinformationen aus OSM gewonnen werden. Es ist somit in der Lage, die Kartenqualität zu verbessern und die Karte global zu registrieren. Die mittels ICP Verfahren bestimmte Transformation beschreibt dabei den Versatz zwischen Gebäudeumrissen aus der Karte und 3D-Entfernungsdaten eines LiDAR-Sensors.

OGM sind eine beliebte Methode zur Darstellung der Umgebung. Sie benötigen jedoch einen hohen Speicherbedarf, der quadratisch mit der Reichweite des Sensors wächst. Um den hohen Speicherbedarf zu reduzieren, führen Schiotka et al. [2017] eine speichereffiziente Kartendarstellung ein, die auf einer konstanten Menge von individuellen LiDAR-Scans basiert. Um das Kartierungsproblem zu lösen, wählt der Ansatz inkrementell Scans aus der Scan-Karte aus, basierend auf den Informationen, die aus den zuvor ausgewählten Scans extrahiert wurden. Zur Lokalisierung wird eine MCL mit einem optimierten Sensormodell für die scanbasierte Kartendarstellung implementiert.

Der probabilistische Ansatz von Suger und Burgard [2017] verwendet ebenfalls OSM zur autonomen Navigation. Mit einem LiDAR-Sensor werden semantische Geländeinformationen extrahiert und mit Teilstücken der OSM-Straßenkarte abgeglichen, unter Verwendung einer Markov-Chain Monte-Carlo Technik. Der Ansatz ist in der Lage, mit der Unsicherheit der Kartenmerkmale von OSM umzugehen.

Die Methode von Wu et al. [2017b] führt eine Graph-basierte Lokalisierungstechnik ein, die neben *Sparse Features* auch Fahrbahnmarkierungen auf Basis einer HD-Karte integriert. Mit einem Multi-Hypothesen-Ansatz wird das Datenassoziationsproblem gelöst und die Robustheit gegenüber Ausreißern erhöht. Die Verwendung von High-Level Merkmalen ermöglicht den adaptiven Einsatz auf Autobahnen und in Städten, unter deutlich geringerem Speicher- und Rechenaufwand.

Die Methode Rormero et al. [2018] verbessert die Fahrzeugposition, indem ein vorab erstelltes OGM im Lokalisierungsprozess berücksichtigt wird, um die Auswahl an möglichen Posen zu begrenzen und sich innerhalb dieser Karte durch eine MCL zu lokalisieren. Jedes Partikel wird basierend auf der Gültigkeit seiner aktuellen Position zur gegebenen Karte sowie unter Berücksichtigung der vergangenen Trajektorie gewichtet. Hierzu wird eine zusätzliche Gitterkarte erstellt, bei der jede Zelle einen

numerischen Wert erhält, der die Anzahl der Zellen zwischen ihr und dem freien Raum in der Karte darstellt.

Im Vergleich zu den Vorgestellten Verfahren von Ruchti et al. [2015] und Rormero et al. [2018] werden in dieser Arbeit die Partikel auf der Basis der Abweichung der vergangenen Trajektorie zum gegebenen Straßenverlauf gewichtet. Es müssen somit keine zusätzlichen Datenrepräsentationen, wie z. B. ein OGM, erzeugt werden und das Ergebnis ist nicht von der gewählten Gittergröße abhängig. Darüber hinaus erfordern gridbasierte Methoden im Allgemeinen einen beträchtlichen Speicherbedarf, insbesondere bei großen Karten. Die Verwendung von spärlichen Karten hingegen benötigt nur einen Bruchteil der Datenmenge und lässt sich somit deutlich einfacher mit anderen Teilnehmern austauschen. Ein Nachteil von spärlichen Karten ist, dass sich die Anzahl von Alleinstellungsmerkmalen pro Landmarke stark reduziert. Dies erhöht wiederum die Schwierigkeit der Datenassoziation. Um dieses Problem zu lösen, wird in dieser Arbeit das Vorwissen der vergangenen Trajektorie sowie eine Pfad-Karte berücksichtigt, welche aus OSM Daten oder durch ein anderes Fahrzeug erzeugt werden kann.

Die Autoren von Fassbender et al. [2015] stellen ein Kartierungs- und Navigationssystem vor, bei dem Straßenabschnitte, Kreuzungen und markante Strukturen wie Häuser und Bäume mithilfe von LiDAR-Daten erkannt und als abstrakte Objekt-Landmarken entlang eines Pfades gespeichert werden. Die Autoren beschreiben ihr Verfahren als reines Lokalisierungsverfahren. Die Lokalisierung entlang eines Pfades erfolgt mit einer MCL, wobei die erstellte Karte mittels Funk an die Folgefahrzeuge übertragen wird. Die aus LiDAR-Punktwolken erstellte OGM wird zur lokalen Hindernisvermeidung verwendet. Jaspers et al. [2017a] erweitert das Verfahren um abstrakte Bildmerkmale nach dem Konzept der Bag Of Visual Words (BOVW). Der Ansatz von Jaspers [2021] wiederum verwendet anstelle von LiDAR-Daten mehrere Stereokamerasysteme, um die Umwelt in 3D wahrzunehmen. Die Farbinformationen und 3D-Daten werden auch in einer OGM fusioniert.

Neben Hinderniswahrscheinlichkeiten beinhaltet das OGM Informationen über die Farbe, Höhe und Steigung des Terrains, mit denen ein komfortablerer und robusterer Pfad geplant werden kann. Darüber hinaus beschreibt die Arbeit von Jaspers [2021] ein landmarkenbasiertes Kartierungs- und Lokalisationssystem, das nach dem Ansatz von Fassbender et al. [2015] anderen Fahrzeugen entlang einem bereits kartierten Pfad folgen kann. Es nutzt hierzu Bildmerkmale und kamerabasierte Objektlandmarken, die während einer Kartierungsfahrt mit teurer Messsensorik erstellt werden. Die Lokalisierung basiert ebenfalls auf der in Fassbender et al. [2015] vorgestellten MCL und es handelt sich explizit um kein SLAM-Verfahren, sondern um ein reines Lokalisierungssystem.

Die grundlegende Methode von Kümmerle et al. [2010] ist der in dieser Arbeit vorgestellten Methode am ähnlichsten. Im Frontend wird eine MCL für den Lokalisierungsprozess verwendet, während ein Graph-basierter SLAM-Ansatz die Landmarken-Beobachtungen zu einer konsistenten Karte ausrichtet.

Der Hauptunterschied zu dem Verfahren besteht jedoch darin, dass nur *Sparse Features* verwenden werden und die Karte simultan zum Lokalisierungsprozess erstellt wird, wobei die Unsicherheit der Sensor-Landmarken und die geschätzte Fahrzeugposition probabilistisch integriert werden.

5.3 Verfahrensbeschreibung

Im folgenden Abschnitt wird der Einsatz des Verfahrens am Beispiel eines aufgesplitteten Konvois beschrieben. Zu Beginn besitzt das Führungsfahrzeug nur einen Zielpunkt. Mittel OSM wird eine globale Route geplant. Diese Route entspricht dem Wegverlauf der sogenannten virtuellen Pfad-Karte. Die virtuelle Pfad-Karte ist eine topologisch-metrische Repräsentation einer Wegstrecke und besteht aus Graph-Knoten, welche entlang eines Pfades verteilt und über eine Linie verbunden sind. Die erstellte virtuelle Pfad-Karte dient dabei als Wegbeschreibung, um ein gewünschtes globales Ziel zu erreichen. Der Pfad ist als virtuell bezeichnet, da er keine echte fahrbare Trajektorie beschreibt.

Die virtuelle Pfad-Karte wird somit immer auf Basis von externen Karteninformationen erzeugt, wenn keine eigenen Karteninformationen vorliegen. Die Existenzwahrscheinlichkeit und Klassenbeschreibung von Kartenobjekten ist dabei als unbekannt anzunehmen und die Graph-Knoten können nur entlang des Pfades approximiert werden.

Um virtuelle Ortsmerkmale zu erzeugen, werden im nächsten Schritt zu jedem Graph-Knoten die Sichtbarkeit von Landmarken auf Basis einer virtuellen Verdeckungsberechnung bestimmt. Die Datengrundlage für die Verdeckungsberechnung liefert ebenfalls OSM. Eine genaue Beschreibung bezüglich des Erstellungsprozesses der Pfad-Karte ist in Unterabschnitt 6.2.1.1 zu finden.

Nachdem die virtuelle Pfad-Karte erzeugt ist, steht das Ziel fest und das Führungsfahrzeug kann die Kartierungsfahrt beginnen. Das Führungsfahrzeug ist in Abbildung 5.1 mit einer schwarzen Bounding-Box und der virtuellen globalen Pfad-Karte mit einer schwarzen Linie sowie die Pfadknoten durch schwarze Punkte dargestellt. Jeder Graph-Knoten repräsentiert dabei einen Ort mit zugehörigen Landmarken (grüner Stern), die ebenfalls als Graph-Knoten modelliert werden.

Das Ziel des Führungsfahrzeuges ist nun, sich innerhalb und entlang der erstellten virtuellen Pfad-Karte mithilfe einer im Folgenden vorgestellten MCL global zu lokalisieren. Damit die MCL in der globalen Karte konvergiert und falsche Datenassoziationen den Schätzprozess nicht nachhaltig beeinflussen, wird das Führungsfahrzeug zur Initialisierung manuell eine gewisse Strecke entlang des Pfades gefahren. Sofern GNSS verfügbar ist, kann dieses auch zur initialen Lokalisierung verwendet werden.

Konvergiert die MCL, so sind globale Positionsinformationen der Eigenposition und der Landmarken bekannt. Diese werden dann als Bedingungen im Graph-SLAM-Backend berücksichtigt, um die lokal erstellte Karte in der globalen Karte zu registrieren. Die beiden Prozesse werden dabei kontinuierlich und parallel ausgeführt unter

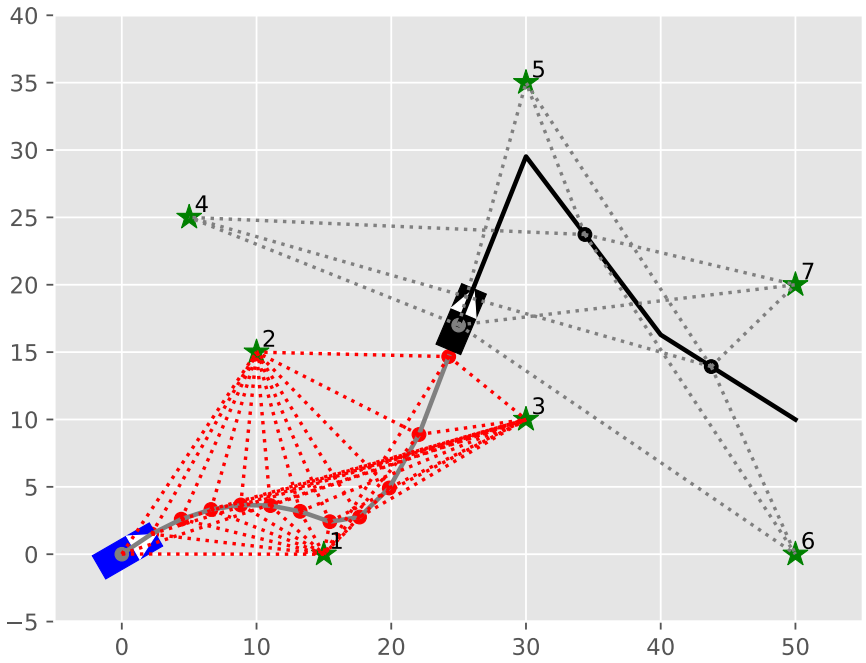


Abbildung 5.1:
Die Abbildung zeigt die kartierten Landmarken (in Grün - 0, 1, 2) sowie die dazugehörige Wegstrecke (durchgezogene graue Linie). Die einzelnen Wegpunkte der gefahrenen Trajektorie sind miteinander verbunden und haben eine vorgegebene Richtung. Die roten gestrichelten Linien zwischen den Wegpunkten und den kartierten Landmarken zeigen, welche Landmarken an welchem Wegpunkt sichtbar sind. Dies ermöglicht die Eingrenzung des Suchraums nach Landmarken. Die schwarze Linie hingegen zeigt die virtuelle Pfad-Karte, welche anhand der vorliegenden Karteninformationen erstellt wird. Die Sichtbarkeit von Landmarken an den verschiedenen Fahrzeugpositionen ist nicht bekannt, jedoch werden auf Basis einer Verdeckungsberechnung virtuelle Verknüpfungen erstellt (grau gestrichelte Linie). Erst wenn das Führungsfahrzeug dem Pfad folgt und sich erfolgreich lokalisiert hat, können die Kanten des Graphen aktualisiert werden.

Berücksichtigung der neu erstellten Karte (Graph-SLAM) und den neuen globalen Positionsinformationen (MCL).

Während der Fahrt wird durch das vorgestellte SLAM-Verfahren die Pfad-Karte auf Basis echter Beobachtungen aktualisiert und den Folgefahrzeugen zu Verfügung gestellt. Die Aktualisierung der Karte ist beispielsweise im reinen Kartierungs- und Lokalisierungsverfahren von Jaspers [2021] nicht vorgesehen.

Das Beispiel in Abbildung 5.1 zeigt das Führungsfahrzeug (Schwarz), nachdem es auf Basis des im Folgenden vorgestellten SLAM-Verfahrens bereits erfolgreich die Hälfte des gegebenen Pfades passieren und dabei den genauen Verlauf des Pfades (graue Linie) sowie die sichtbaren Pfadknoten und Landmarkenverknüpfungen aktualisieren konnte (in Rot gestrichelt).

Um nun dem Führungsfahrzeug folgen zu können und um sich auf Basis der erweiterten und aktualisierten Karte zu lokalisieren, wird dem Folgefahrzeug (Blau) die neue

Pfad-Karte per Funk übermittelt. Während das Führungsfahrzeug die Unsicherheiten der Kartenlandmarken noch als unbekannt annehmen musste, kann das Folgefahrzeug jetzt auf die bestimmten Zustandsunsicherheiten des Führungsfahrzeugs zurückgreifen, da diese ebenfalls kartiert werden. Im Hinblick auf die Softwarearchitektur gibt es keine Unterschiede zwischen dem Führungsfahrzeug und dem Folgefahrzeug und so können die Rollen der Fahrzeuge flexibel getauscht werden.

Für den Fall, dass für das Führungsfahrzeug keine Kartendaten vorliegen, ist das SLAM-Verfahren somit dennoch in der Lage, eine lokale Karte in einem eigenen Koordinatensystem zu erstellen. Über längere Distanzen müsste jedoch ein Loop-Closing durchgeführt werden, um die akkumulierten Fehler zu reduzieren.

5.4 Systemarchitektur

Der Aufbau des Verfahrens besteht, wie beim klassischen SLAM, aus einem Frontend und einem Backend. Das Frontend abstrahiert Sensorbeobachtungen, die für die Schätzung der Fahrzeugposition geeignet sind. Hierzu werden LiDAR-basierte Objektinstanzen und der Straßenverlauf aus der Punktwolke extrahiert und deren Zustand geschätzt sowie relevante Landmarken daraus abgeleitet. Zusätzlich wird die Datenassoziation mithilfe der rekursiven Schätzverfahren im Frontend gelöst. Die MCL bestimmt die globale Fahrzeugposition und Orientierung unter Berücksichtigung der vergangenen Trajektorie, vertikaler Objekte und einer Landmarkenkarte.

Das Backend hingegen basiert auf einer Graphen-basierten Formulierung des SLAM-Problems, bei der Pose, Odometrie und Landmarkenbeobachtungen als Knoten und Kanten repräsentiert werden. Die Aufgabe des Backend ist es, das zusätzliche Wissen der verfügbaren Pfad-Karte im Optimierungsprozess zu integrieren und zu nutzen, um sowohl die Fahrzeugtrajektorie als auch die beobachteten Landmarken zu kartieren und an einer globalen referenzierten Karte auszurichten. Im Optimierungsschritt wird dazu die optimale Knotenkonstellation gesucht, die den durch die Beobachtungen gegebenen Fehler minimiert. Es wird angenommen, dass die Beobachtungen durch Gauß'sches Rauschen beeinflusst sind und das Datenassoziationsproblem im Frontend des Frameworks bereits gelöst wurde.

Eine Übersicht der Systemarchitektur ist in Abbildung 5.2 gegeben. Die verschiedenen Funktionen sind als Blockdarstellung und der Datenfluss durch Pfeile dargestellt.

Die Ausführung der MCL sowie des Graph-SLAM Verfahren erfolgt parallel. Sofern die MCL konvergiert, können die globalen Positionsinformationen der Eigenposition sowie die der assoziierten Landmarken im Graph-SLAM Backend als statische Graph-Knoten repräsentiert und die Ergebnisse fusioniert werden. Hierbei wird die Kovarianz des Filterzustandes der MCL berücksichtigt. Die räumlichen Beziehungen werden in Form von Fehlerfunktionen zwischen Pose-Pose sowie Pose-Landmarke im Backend beschrieben, während globale Positionsinformationen der MCL als Bedingungen zum Graphen hinzugefügt werden, welche die lokale Karte im Universal Transverse Mercator (UTM)-Koordinatensystem beschreiben.

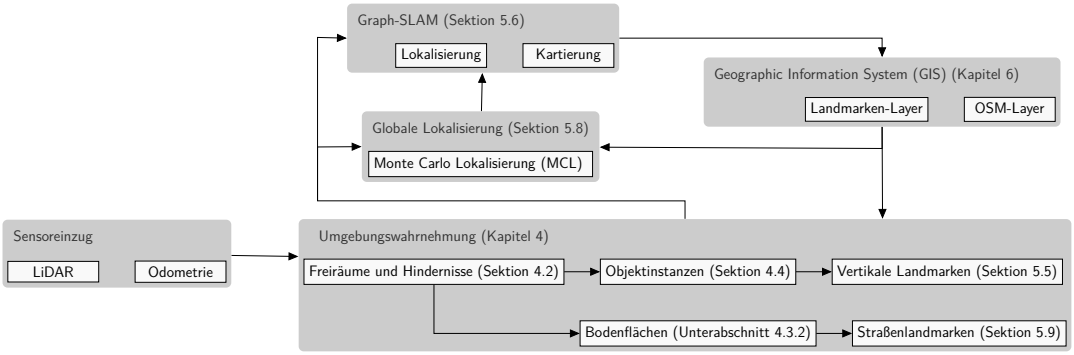


Abbildung 5.2:
Schematischer Aufbau und Zusammenspiel der einzelnen Komponenten des SLAM-Framework.

Das gesamte Verfahren lässt sich neben der SLAM-spezifischen Unterscheidung zwischen Frontend und Backend in drei Hauptfunktionsblöcke unterteilen.

Der erste Block beschreibt die Umgebungswahrnehmung und die Vorverarbeitung, die nötig ist, um ein konsistentes Umgebungsmodell aufzubauen und Landmarkenhypothesen zu generieren. Der Zustand der Landmarken und die Datenassoziationen zwischen mehreren Zeitschritten werden mithilfe eines Multi-Target Tracking (MTT)-Verfahrens bestimmt und gelöst, was ebenfalls zum Frontend des SLAM-Verfahrens gehört und im Folgenden beschrieben wird. Die Erzeugung der Straßenlandmarken ist ebenfalls Teil des Frontends. In Kapitel 4 wurden bereits die Vorverarbeitungsschritte zur Freiraum- und Hinderniserkennung sowie Bodenflächenerkennung beschrieben.

Der zweite Block besteht aus der MCL, welche als Eingangsdaten eine gegebene Pfad-Karte mit spärlichen Kartenlandmarken benötigt sowie die mittels MTT-Verfahren bestimmten vertikalen Landmarken. Die Kartendaten werden aus dem GIS geladen.

Zum dritten Block gehört die Erstellung des Graphen und das Lösen des Optimierungsproblems aller Teilprobleme. Nach jedem Optimierungsschritt wird die Karte des Graph-SLAM-Verfahrens in einer extra Datenschicht im GIS gespeichert bzw. aktualisiert.

In Abbildung 5.3 ist das SLAM-Verfahren als vereinfachtes dynamisches Bayes'sches Netzwerk mit allen Zuständen, Messungen und Aktionen visualisiert. Im Folgenden werden nun die einzelnen Komponenten detaillierter dargestellt.

5.5 Landmarkenerkennung

5.5.1 Einleitung

Die robuste und eindeutige Erkennung von Landmarken ist aufgrund mehrerer Faktoren eine Herausforderung. Sind viele Objekte sehr nahe beieinander, ist die eindeutige Abgrenzung erschwert und es kann zu Mehrdeutigkeiten kommen. Zusätzlich spielt im Allgemeinen die Erkennungswahrscheinlichkeit des verwendeten Sensors eine wichtige Rolle. Die Erkennungswahrscheinlichkeit beschreibt die Wahrscheinlichkeit, dass ein Objekt vom Sensor erkannt wird, sofern sich das Ziel innerhalb des Sichtfeldes des Sensors befindet. Ist die Wahrscheinlichkeit klein, so kann es sein, dass in einem Zeitschritt keine Detektion erzeugt wird, obwohl ein Zielobjekt vorhanden ist.

Zusätzlich ist die Sensorauflösung ein wichtiger Faktor. Ist beispielsweise die Winkelauflösung zu gering, kann es vorkommen, dass der Sensor nicht in der Lage ist zwei Objekte mit geringem Abstand voneinander zu unterscheiden. Als Ergebnis wird fälschlicherweise nur eine Detektion erzeugt. Dies verstößt gegen die gängige Annahme, dass jede Detektion nur einem Track zugeordnet werden kann und führt in der Folge zu unlösbaren Assoziationsproblemen. Darüber hinaus erzeugen Fehlmessungen und Clutter zusätzliche Detektionen, was die Komplexität der Datenzuordnung erhöht.

Im Hinblick auf den Erfassungsbereich und die Erkennungswahrscheinlichkeit liefern moderne $360°$ LiDAR-Sensoren eine hohe vertikale und horizontale Auflösung und ermöglichen hierdurch die robuste und sichere Erfassung der dreidimensionalen Umgebung auch bei schwierigen Belichtungsbedingungen. In Hinblick auf die gesetzten Forschungsziele dieser Arbeit wurde aus diesen Gründen primär auf die Verwendung eines einzelnen mechanischen LiDAR-Sensors gesetzt.

In dieser Arbeit wird daher auf Basis einer LiDAR-Punktwolke die Umgebung abgetastet und ein objektbasiertes Umgebungsmodell erstellt. Das objektbasierte Umgebungsmodell besteht dabei aus sogenannten Objektlisten, die Objektinstanzen sowie Bodeninstanzen beinhalten. Eine Objektinstanz beschreibt ein vertikales Objekt in der Umgebung und ist durch die Position x, y, z und durch die Parameter Länge l, Breite b und Höhe h definiert, siehe Unterabschnitt 4.4.4. Bodeninstanzen hingegen repräsentieren zusammengehörige Freiraumflächen (Bodenflächen), die durch eine Position, eine konvexe Hülle und der dazugehörigen Punktwolke repräsentiert werden, siehe Unterabschnitt 4.3.2.

Das Verfahren und die Vorverarbeitungsschritte zur robusten Erkennung von Objektinstanzen in der LiDAR-Punktwolke wurden bereits ausführlich in Kapitel 4 beschrieben. Im Folgenden wird das rekursive Filter zur Landmarkenerkennung beschrieben.

5.5.2 Multi-Target-Tracking Verfahren

In dieser Arbeit wird ein rekursives MTT-Verfahren verwendet, um die Zustände aller vertikalen Objekte zu bestimmen und implizit das Datenassoziationsproblem

des SLAM-Problems zu lösen. Die detektierten Objektinstanzen $\mathbf{Y}_k^{oi} = \{\mathbf{y}_n^{oi}\}_{n=0}^{N}$ werden als Beobachtung bzw. Detektion zum Zeitschritt k im MTT-Verfahren berücksichtigt.

Die Methode zur Erzeugung der Objektinstanzen neigt im Allgemeinen zur Übersegmentierung. Dies hat zur Folge, dass mehr Objektinstanzen erkannt werden als wahre Objekte vorhanden sind. Eine Anforderung an das MTT-Verfahren ist, mehrere Objektinstanzen einem Track zuweisen zu können, da im Durchschnitt die Anzahl von Objektinstanzen größer ist als die Anzahl der aktuellen Tracks. Ein Track beschreibt den Filterzustand eines Objektes, dessen Zustand über die Zeit rekursiv geschätzt wird.

Neben der Trackverwaltung ist die rekursive Zustandsschätzung (Filterung) die Hauptkomponente eines MTT-Systems. Das Greedy Dirichlet Process Filter (GDPF) von Naujoks et al. [2019c] ist ein MTT-Verfahren, bei dem die probabilistische Datenassoziation im Filterschritt erfolgt. Greedy bedeutet, dass die Messungen zur wahrscheinlichsten Komponente assoziiert werden. Die Zustandsschätzung ist neben dem UKF durch ein Augmented Coordinated Turn (ACT)-Prozessmodell realisiert, um neben der Position und Dimension auch die Geschwindigkeit und Orientierung der Objekte zu bestimmen [Bar-Shalom et al., 2001].

Auf Basis der Objektinstanzen führt eine Trackverwaltung eine Initialisierung, Bestätigung und Löschung der aktuellen Tracks durch. Nicht zugewiesene Objektinstanzen können neue vorläufige Tracks erzeugen und ein vorläufiger Track wird akzeptiert, wenn die Qualität des Tracks bestimmte Kriterien erfüllt. Tracks von geringer Qualität werden gelöscht, wenn sie beispielsweise länger nicht mehr beobachtet wurden und somit die Zustandsunsicherheit stark ansteigt. Jeder neue Track beginnt dabei in einem vorläufigen Zustand. Wenn einem vorläufigen Track genügend Objektinstanzen zugewiesen wurden, ändert sich der Status in *aktiv* und der MTT weist den Track als valides Objekt aus. Wenn innerhalb einer vorgebbaren Anzahl von Zeitschritten einem Track keine Detektionen mehr zugewiesen wurden, wird dieser gelöscht. Die Assoziationsfunktion berücksichtigt die Distanz sowie die Positions- und Dimensionsunsicherheit eines Tracks zu allen detektierten Objektdistanzen.

Durch die zusätzliche Berücksichtigung der Zustandsunsicherheiten bei der Datenassoziation ermöglicht das Verfahren die Objektinstanzen auch bei einer schwierigen Anordnung robust zuzuordnen. Die Assoziation erfolgt dabei kontinuierlich für alle Objektinstanzen, bis diese vollständig den vorhandenen Zuständen zugeordnet oder neue Zustände erzeugt wurden.

Der Zustandsvektor für jeden Track des UKF ist dabei wie folgt definiert:

$$\mathbf{x}^{oi} = (x, y, v_x, v_y, v_\psi, l, b, h)\,, \tag{5.1}$$

wobei x, y die kartesischen Positionen, v_x, v_y die kartesischen Geschwindigkeiten, v_ψ die Gierwinkelrate und l, b, h die 3D-Dimensionen der Objektinstanzen sind. Das Messmodell verwendet direkt die kartesischen Koordinaten sowie die Objektdimensionen der Objektinstanz $\mathbf{y}^{oi}$.

Um die Tracks den im Graph-SLAM kartierten Landmarken eindeutig zuordnen zu können, wird jedem Track eine eindeutige fortlaufende Identifikationsnummer zugewiesen. Zusätzlich erfolgt die Speicherung aller Zeitpunkte k an dem eine neue Messung dem Track zugeordnet wurde. So lassen sich neben einer eindeutigen Zuordnung, die Dauer der Sichtbarkeit, sowie die Dauer von Unterbrechungen für jeden Track bestimmen.

5.5.3 Auswahl von Tracks als Sensor-Landmarke

Die Auswahl der Tracks, die als Sensor-Landmarke infrage kommen, wird zu jedem Zeitschritt k für die komplette Menge an $1, \ldots, L \mid L \in \mathbb{N}$ Tracks $\mathbf{X}_k^{oi} = \{\mathbf{x}_{k,l}^{oi}\}_{l=1}^{L}$ durchgeführt, wobei l einem eindeutigen Index entspricht, der nur einmal in der Menge aller Tracks vorkommen kann.

Der Auswahlprozess erfolgt nach einer einfachen Heuristik, die verschiedene Kriterien berücksichtigt. Zuerst wird die Länge der Sichtbarkeit jedes Tracks überprüft. Ist ein Track länger als $t_{lms} = 3\,\mathrm{s}$ sichtbar, wird im nächsten Schritt überprüft, ob die absolute Geschwindigkeit kleiner ist als der Schwellwert von $v_{lms} = 0.5\,\mathrm{m/s}$. Der Schwellwert wurde bewusst größer als $0.0\,\mathrm{m/s}$ gewählt, um Objekte mit sogenannten Pseudogeschwindigkeiten herauszufiltern. Dies ist insbesondere auf das Bounding-Box Messmodell zurückzuführen, welches bei Vegetationslandmarken, bei denen sich das Erscheinungsbild zwischen zwei Zeitschritten stark ändern kann, die Objektgeometrie nicht ausreichend genug beschreibt.

Im letzten Schritt wird die Information über die Positions- und Dimensionsunsicherheit der einzelnen Tracks dafür verwendet, Landmarken zu identifizieren [Burger et al., 2019a, Naujoks et al., 2019a]. Beispielsweise ist eine hohe Unsicherheit in der Positionsschätzung auf eine fehlerhafte Datenassoziation oder auf die fehlerhafte Bestimmung des Objektmittelpunkts zurückzuführen. Das Gleiche gilt für eine hohe Unsicherheit in der Objektdimension.

Es wird vorausgesetzt, dass für eine Landmarke die Unsicherheit in der Position durch die erfolgreiche Assoziation weiterer Messungen geringer werden muss. Um dies zu überprüfen, wird in jedem Updateschritt k, der Mahalanobis-Abstand zwischen der zuletzt assoziierten Messung $\tilde{\mathbf{y}}_k^{oi}$ und dem Track $\mathbf{x}_{k,l}^{oi}$ unter Berücksichtigung der Kovarianzmatrix $\boldsymbol{P}_k^{-1}$ bestimmt:

$$\pi_{k,l}^{oi} = (\tilde{\mathbf{y}}_k^{oi} - \mathbf{x}_{k,l}^{oi})^T \boldsymbol{P}_{k,l}^{-1} (\tilde{\mathbf{y}}_k^{oi} - \mathbf{x}_{k,l}^{oi}). \tag{5.2}$$

Der Messvektor $\tilde{\mathbf{y}}_k^{oi}$ ist um die Geschwindigkeiten $v_x = 0.0, v_y = 0.0, v_\psi = 0.0$ zur Einhaltung der Vektorengröße erweitert.

Um nun eine Teilmenge an Objektinstanzen als Sensorlandmarke $\boldsymbol{Z}_k \subseteq \mathbf{X}_k^{oi}$ auszuwählen, wird das Gewicht w_l auf Basis der Standardabweichung für jeden Track berechnet:

$$w_l = \sqrt{\frac{1}{|\pi_{k,l}^{oi}| - 1} \sum_k (\pi_{k,l}^{oi} - \overline{\pi}_{k,l}^{oi})}. \tag{5.3}$$

Diejenigen Objektinstanzen, welche die kleinste Standardabweichung besitzen, werden als Sensor-Landmarke ausgewählt. Die besten Ergebnisse konnten mit mindestens vier und maximal 70 Sensorlandmarken erreicht werden.

Die Sensor-Landmarken werden anschließend als Eingang in der MCL zur Lokalisierung sowie zur simultanen Lokalisierung und Kartierung im Graphen-basierten Backend berücksichtigt.

5.6 Graph-SLAM Backend

Die Schätzung der a posteriori Wahrscheinlichkeitsverteilung wird im Backend wie in Burger et al. [2019a] durch einen dünn besetzten Graphen $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ dargestellt. Posen, Odometrie und Landmarkenbeobachtungen sind durch Knoten und Kanten repräsentiert, siehe Unterabschnitt 3.2.5.2.

In der gewählten Graphen-Darstellung werden Landmarken als 2D Punkte und die Fahrzeugposition als SE(2) repräsentiert. Die kombinierte Menge an Fahrzeugposen in SE(2) $\{\mathbf{x}_k\}_{k=1}^{K} \mid \mathbf{x}_k = \left(x_k^x, x_k^y, x_k^\psi\right)^T$ und die Menge an 2D Landmarken $\{\mathbf{l}_l\}_{l=1}^{L} \mid \mathbf{l}_l = (l_k^x, l_k^y)^T$ bilden die Graphknoten und sind wie folgt definiert:

$$\mathbf{V} = \{\mathbf{x}_k\}_{k=1}^{K} \cup \{\mathbf{l}_l\}_{l=1}^{L} \mid K, L \in \mathbb{N}. \tag{5.4}$$

Um den Graphen zu erstellen, werden in jedem Zeitschritt k neue Knoten und Kanten hinzugefügt. Kanten zwischen den Knoten entsprechen dabei Ereignissen, wie z. B. der Eigenbewegung, Beobachtung von Landmarkenobjekten sowie globalen Positionsinformationen, die mithilfe der MCL und der Straßen-Offsetbestimmung ermittelt werden.

Die Kanten gelten dabei als weiche Bedingungen, die vergleichbar sind mit Einträgen in einer Informationsmatrix eines linearen Gleichungssystems. Globale Positionsinformationen der Fahrzeugpose und Landmarken werden als statische Bedingungen modelliert, die nur bedingt verändert werden können.

5.6.1 Übergangsmodell

Das Übergangsmodell der Fahrzeugposen wird als eine Bewegungsbedingung (Kante) zwischen zwei aufeinanderfolgenden Posen modelliert. Die Pose $\mathbf{x}_k$ zum Zeitschritt k wird aus der vorherigen Pose $\mathbf{x}_{k-1}$ auf Basis des Aktionsvektors $\mathbf{u}_{k-1}$ propagiert.

Die neue Pose bestimmt sich dann wie folgt: $\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + v_{k-1}$, wobei v_{k-1} eine Gauß'sche Rauschvariable mit Mittelwert Null und Kovarianz $\mathbf{Q}_{k-1}$ ist. Der Aktionsvektor $\mathbf{u}_{k-1}$ setzt sich aus der Gierrate und der Geschwindigkeit des Fahrzeugs zusammen. Die Schätzung einer neuen Pose $f_k(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1})$ basiert dabei auf einem nichtlinearen Bewegungsmodell, welches in Unterabschnitt 3.2.3 beschrieben wurde. Die benötigten Beschleunigungen und Drehraten liefern ein lose gekoppeltes GNSS/Inertial Navigation System (INS)-System.

Unter der Annahme von normalverteiltem Rauschen gilt:

$$p\left(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}\right) \propto \exp -\frac{1}{2}||f_k(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) - \boldsymbol{x}_k||^2_{\mathbf{Q_{k-1}}}. \tag{5.5}$$

Die Pose-Pose Bedingung J_k^{odo} kann somit wie folgt modelliert werden:

$$J_k^{odo} = ||f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) - \boldsymbol{x}_k||^2_{\mathbf{Q_{k-1}}} = \boldsymbol{o}_k^T \mathbf{Q}_{k-1}^{-1} \boldsymbol{o}_k, \tag{5.6}$$

mit Fehlerfunktionen $\boldsymbol{o}_k = f_k(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) - \boldsymbol{x}_k$, wobei Funktion $f_k(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) = \boldsymbol{x}_{k+1} \ominus \boldsymbol{x}_k$ die ideale Odometrie beschreibt und Kovarianzmatrix $\mathbf{Q_{k-1}}$ des Bewegungsmodells. In Abbildung 5.4 ist die Pose-Pose Kante als rote gerichtete Linie dargestellt.

5.6.2 Landmarken-Messmodell

Das Landmarken-Messmodell kann wie folgt angenommen werden:

$$p\left(\boldsymbol{z}_{k,l} \mid \boldsymbol{x}_k, \boldsymbol{l}_l\right) \propto \exp -\frac{1}{2}||g_i(\boldsymbol{x}_k, \boldsymbol{l}_l) - \boldsymbol{z}_{k,l}||^2_{\mathbf{P}^{lm}_{k,l}}, \tag{5.7}$$

mit l-ter Landmarke $\boldsymbol{l}_l$ die zur Sensor-Landmarke $\boldsymbol{z}_{k,l}$ gehört und die Karte $\boldsymbol{M}$ bildet, wobei $l \in 1, \ldots, L$ Landmarken.

Die Pose-Landmarken-Bedingung, welche die Beobachtung einer Landmarke an einer definierten Pose beschreibt, wird wie folgt modelliert:

$$J^{lm}_{k,l} = w_l ||g\left(\boldsymbol{x}_k, \boldsymbol{l}_l\right) - \boldsymbol{z}_{k,l}||^2_{\mathbf{P}^{lm}_{k,l}} = w_l \, \boldsymbol{e}^T_{k,l} \mathbf{P}^{lm^{-1}}_{k,l} \boldsymbol{e}_{k,l}, \tag{5.8}$$

mit Messkovarianz $\mathbf{P}^{lm}_{k,n}$ der Sensor-Landmarke $\boldsymbol{z}_{k,l}$ des MTT-Verfahren, Fehlerfunktion $\boldsymbol{e}_{k,l} = g\left(\boldsymbol{x}_k, \boldsymbol{l}_l\right) - \boldsymbol{z}_{k,l}$ und Gewicht w_l.

Für die Posen-Landmarken Messfunktion gilt dann:

$$g\left(\boldsymbol{x}_k, \boldsymbol{l}_l\right) = \begin{pmatrix} (x^x_k - l^x_l)\cos x^\psi_k + (x^y_k - l^y_l)\sin x^\psi_k \\ -(x^x_k - l^x_l)\sin x^\psi_k + (x^y_k - l^y_l)\cos x^\psi_k \end{pmatrix}, \tag{5.9}$$

welche die Landmarke in das Koordinatensystem des Fahrzeugs transformiert.

Zur Betrachtung und Bewertung dynamischer Objekte wird zusätzlich die Beweglichkeit jeder Landmarke durch die latenten Variablen $\mathbf{W} = \{w_l\} \mid l \in 1, \ldots, L$ für jede l-te Landmarke auf Basis des MTT-Verfahrens bestimmt. Somit ist es möglich, Landmarken aus dem Graph zu entfernen, die fälschlicherweise als statische Landmarken erkannt wurden und zu einem späteren Zeitschritt erst als beweglich identifiziert werden. Der Prozess, um $\mathbf{W}$ für alle Sensor-Landmarken zu bestimmen, wurde im vorherigen Unterabschnitt 5.5.3 beschrieben.

5.6.3 Graphen-Beschreibung

Der Graph lässt sich somit auf Basis der Gleichungen (3.13), (3.14) und (3.16) als Summe von Bedingungen beschreiben:

$$\mathbf{V}^* = \arg\min_{\mathbf{V}} \sum_{k=1}^{K} J_k^{odo} \sum_{k=1}^{K} \sum_{l=1}^{L} J_{k,l}^{lm} \tag{5.10}$$

und mit einem Gauß-Newton Verfahren lösen. Nach dem Optimierungsschritt beschreibt der Graph $\boldsymbol{V}$ die optimale Anordnung aller Landmarken und Fahrzeugposen auf Basis der Landmarken- und Odometriemessungen.

5.7 Globale Lokalisierung und Karten-Registrierung

In diesem Abschnitt wird die globale Lokalisierung beschrieben. Das Verfahren basiert auf einer MCL, um die nicht-gauß'sche multimodale Verteilungen von möglichen Fahrzeugposen entlang der Pfad-Karte zu schätzen. Im Folgenden wird auf die Herleitung und Erklärung der Formeln verzichtet und stattdessen auf Unterabschnitt 3.1.2.4 verwiesen.

5.7.1 Monte Carlo Lokalisierung

Die Verteilung $\text{bel}_{\mathbf{M}}(\boldsymbol{x}_k)$ wird auf Basis aller vergangener Sensor-Landmarken $\mathbf{Z}_{1:k}$, Aktionsvektoren $\mathbf{u}_{0:k-1}$ und der (virtuellen) Pfad-Karte $\mathbf{M} = \{\boldsymbol{m}_l, \ldots, \boldsymbol{m}_L\}$ bestimmt:

$$\text{bel}_{\mathbf{M}}(\boldsymbol{x}_k) = p(\boldsymbol{x}_k \mid \mathbf{Z}_{1:k}, \mathbf{u}_{0:k-1}, \mathbf{M}). \tag{5.11}$$

Sofern neue Sensor-Landmarken $\mathbf{Z}_k = \{\mathbf{z}_{k,l}\}_{l=1}^{L}$ eintreffen, wird im Updateschritt die Gewichtung der Partikel auf Basis einer neuartigen Messfunktion unter Berücksichtigung der Sensor-Landmarken, der vergangenen Trajektorie und den Karten-Landmarken bestimmt. Die Prädiktion der Partikel $p(\boldsymbol{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_{k-1})$ basiert auf dem gleichen Bewegungsmodell wie im Graph-basierten SLAM-Verfahren für die Graph-Knoten der Fahrzeugposition, siehe Unterabschnitt 3.2.3.

Die einzelnen Schritte und Filtergleichungen sind in folgender Auflistung noch einmal zusammengefasst:

- Filtergleichungen:

  $$\overline{\text{bel}}_{\mathbf{M}}(\boldsymbol{x}_k) = \int p(\boldsymbol{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \, \text{bel}_{\mathbf{M}}(\mathbf{x}_{k-1}) \, \mathrm{d}\mathbf{x}_{\mathbf{k-1}},$$

  $$\text{bel}_{\mathbf{M}}(\boldsymbol{x}_k) = \eta \cdot p(\mathbf{Z}_k \mid \boldsymbol{x}_k, \mathbf{M}) \, \overline{\text{bel}}_{\mathbf{M}}(\boldsymbol{x}_k).$$

- Prädiktion der Partikel $p(\boldsymbol{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_{k-1})$ anhand eines Bewegungsmodells.

- Im Updateschritt erfolgt das Update der Partikelgewichte auf Basis eines Messmodells:

$$w \propto p\left(\mathbf{Z}_k \mid \boldsymbol{x}_k, \mathbf{M}\right).$$

- Das Resampling der Partikel basiert auf der Low-Variance Resampling Strategie von Thrun et al. [2005] und wird für 70% aller Partikel angewendet. Die restlichen 30% der Partikelmenge werden uniform entlang des Pfades gestreut.
- Nach jeder Iteration wird die aktuelle Fahrzeugpose $\hat{\mathbf{x}}_k = (x, y, \psi)$ innerhalb der gegebenen Karte auf Basis eines gewichteten Mittelwerts, der besten 33% der Partikel mit dem höchsten Gewicht, bestimmt.

5.7.2 Messmodell

Um das Fahrzeug entlang der (virtuellen) Pfad-Karte zu lokalisieren, wird neben den aktuell beobachteten Sensor-Landmarken $\mathbf{Z}_k$ auch die vergangene Trajektorie berücksichtigt.

Da die (virtuelle) Pfad-Karte $\mathbf{M}$ mehrere (virtuelle) Landmarken-Beobachtungen pro Graph-Knoten $\mathbf{m} \in \mathbf{M}$ speichert, ergibt sich ein Assoziationsproblem, bei dem die virtuellen Beobachtungen den Sensor-Landmarken $\mathbf{Z}_k$ zugeordnet werden müssen. Diese Assoziation ist unbekannt und kann wie folgt modelliert werden:

$$p\left(\mathbf{Z}_k \mid \boldsymbol{x}_k, \mathbf{M}\right) = \textstyle\sum_{\mathbf{m}\in\mathbf{M}} p\left(\mathbf{Z}_k \mid \boldsymbol{x}_k, \mathbf{m}\right) p\left(\mathbf{m} \mid \boldsymbol{x}_k\right). \tag{5.12}$$

Folglich wird die aktuelle wahrgenommene Landmarkenkonstellation $\mathbf{Z}_k$ mit jedem Pfad-Knoten-Tupel $\mathbf{m} \in \mathbf{M}$ abgeglichen und mit $p\left(\mathbf{m} \mid \boldsymbol{x}_k\right)$ gewichtet.

Um $p\left(\mathbf{Z}_k \mid \boldsymbol{x}_k, \mathbf{m}\right)$ zu berechnen, wird die Annahme getroffen, dass die Messungen bedingt unabhängig sind. Daher gilt:

$$p\left(\mathbf{Z}_k \mid \boldsymbol{x}_k, \mathbf{m}\right) \propto \textstyle\prod_{n=1}^{|\mathbf{Z}_k|} p\left(\mathbf{z}_{k,n}^{\mathsf{lm}} \mid \boldsymbol{x}_k, \mathbf{m}\right), \tag{5.13}$$

wobei $p\left(\mathbf{z}_{k,n}^{\mathsf{lm}} \mid \boldsymbol{x}_k, \mathbf{m}\right)$ definiert ist als die Wahrscheinlichkeit zum Partikelzustand $\boldsymbol{x}_k$, dass die Landmarkenbeobachtung $\mathbf{z}_{k,n}^{\mathsf{lm}}$ zur nächstgelegenen Kartenlandmarke $\tilde{l}_{\mathsf{n}}$ von allen Kartenlandmarken $\mathbf{l}_l \in \mathbf{m}$ gehört.

Die Wahrscheinlichkeitsdichte wird als Gauß-Verteilung mit Null-Mittelwert und Standard-Kartenunsicherheit σ_{zm} modelliert:

$$p\left(\mathbf{z}_{k,n}^{\mathsf{lm}} \mid \boldsymbol{x}_k, \mathbf{m}\right) \propto \beta_{\mathsf{zm}} \cdot \mathcal{N}\left(d_M\left(\mathbf{z}_{k,n}^{\mathsf{lm}}, \tilde{l}_{\mathsf{n}}, \mathbf{P}_{k,n}^{lm}\right); 0, \sigma_{zm}\right) \tag{5.14}$$

und unter Verwendung der Mahalanobis-Distanz berechnet:

$$d_M\left(\mathbf{z}_{k,n}^{\mathsf{lm}}, \tilde{l}_{\mathsf{n}}, \mathbf{P}_{k,n}^{lm}\right) = \sqrt{\left(\mathbf{z}_{k,n}^{\mathsf{lm}} - \tilde{l}_{\mathsf{n}}\right)^T \mathbf{P}_{k,n}^{lm\,-1} \left(\mathbf{z}_{k,n}^{\mathsf{lm}} - \tilde{l}_{\mathsf{n}}\right)}. \tag{5.15}$$

$\mathbf{P}_{k,n}^{lm}$ ist dabei die Zustandskovarianz der Sensor-Landmarke $\mathbf{z}_{k,n}^{\mathsf{lm}}$. Der Gewichtungsfaktor β_{zm} von Burger et al. [2019a] ist ein Maß über die Abweichungen der Dimensionen (Länge, Breite, Höhe) der Karten-Landmarke zur geschätzten Dimension der Sensor-Landmarke.

Zusätzlich werden nur die Kartenlandmarken berücksichtigt, welche die Wahrscheinlichkeit $p\left(\mathbf{Z}_k \mid \boldsymbol{x}_k, \mathbf{m}\right)$ maximieren. Die maximale Anzahl an Sensor-Landmarken ist jedoch auf die Anzahl der gegebenen Kartenlandmarken pro Pfad-Karten-Knoten beschränkt. Dies ermöglicht insbesondere eine bessere Konvergenz der MCL in Szenarien, in denen keine Kartenkorrespondenzen gefunden werden oder wo wesentlich mehr Sensor-Landmarken als Karten-Landmarken vorhanden sind.

Der Gewichtungsfaktor $p\left(\mathbf{m} \mid \boldsymbol{x}_k\right)$ aus Gleichung (5.12) wird auf Basis der Mahalanobis-Distanz $d_M(\mathbf{p}^m, \boldsymbol{x}_k, \hat{\mathbf{P}}_k)$ zwischen der Pfad-Knoten-Pose $\mathbf{p}^m$ des Pfad-Knotens $\mathbf{m}$, dem Partikel $\boldsymbol{x}_k$ sowie Filterkovarianz $\hat{\mathbf{P}}_k$ bestimmt:

$$p\left(\mathbf{m} \mid \boldsymbol{x}_k\right) = \mathcal{N}\left(d_M(\mathbf{p}^m, \boldsymbol{x}_k, \hat{\mathbf{P}}_k); 0, \sigma_{pm}\right). \tag{5.16}$$

Eine gängige Methode, um Mehrdeutigkeiten bei der Lokalisierung zu reduzieren, ist die vergangene Trajektorie zu berücksichtigen, um den Suchraum einschränken zu können. [Floros et al., 2013, Rormero et al., 2018, Ruchti et al., 2015, Suger und Burgard, 2017]. Die Ideen der Verfahren von Floros et al. [2013], Rormero et al. [2018] werden daher im Folgenden aufgegriffen und unter der Annahme, dass die vergangene Trajektorie einem Teilstück der gegebenen Pfad-Karte ähnlich ist und die Unsicherheit über die gefahrene Strecke ansteigt, um einen gewichteten Trajektorien-Ansatz erweitert.

Die Partikelgewichte bestimmen sich daher im Wesentlichen aus der geometrischen Ähnlichkeit zu den Pfad-Knoten und ihrer Ortssignatur sowie dem Abgleich der vergangenen Trajektorie mit dem Verlauf der Pfad-Karte. Die Trajektorie $\mathbf{T} = \boldsymbol{x}_k, \boldsymbol{t}_{k-1} \ldots, \boldsymbol{t}_1$ besteht dabei aus den vergangenen Fahrzeugpositionen, die im Koordinatensystem des Partikels $\boldsymbol{x}_k$ zum Zeitschritt k beschrieben werden [Burger et al., 2019a]. Die Transformation der vergangenen Fahrzeugpositionen ermöglicht die virtuelle Repräsentation der vergangenen Trajektorie zur aktuellen Partikelposition und Ausrichtung, was in Abbildung 5.5 durch die blauen Symbole visualisiert ist.

Um nun die vergangene Trajektorie im Updateschritt zu berücksichtigen, wird Gleichung (5.16) um das Produkt der gewichteten Trajektorien-Posen $\mathbf{T}_k$ erweitert:

$$p\left(\mathbf{m} \mid \mathbf{T}_k\right) = \mathcal{N}\left(d_M(\mathbf{p}^m, \mathbf{x}_k, \hat{\mathbf{P}}_k); 0, \sigma_{pm}\right) \cdot \prod_{k=1}^{|\mathbf{T}_k|} \beta_{\mathsf{tr,k}} \cdot \mathcal{N}\left(\perp\left(\mathbf{t}_k, \mathbf{p}_i^m, \mathbf{p}_{i+1}^m\right), \sigma_{pm}\right).$$

Als Maß wird der senkrechte Abstand $\perp\left(\mathbf{t}_k, \overrightarrow{\mathbf{p}_i^m \mathbf{p}_{i+1}^m}\right)$ zwischen der Trajektorienpose $\mathbf{t}_k \in \mathbf{T}_k$ und einer Linie zwischen den beiden nächstgelegenen Knoten $\mathbf{p}_i^m, \mathbf{p}_{i+1}^m$ zum

Trajektorienpunkt $\mathbf{t}_k$ herangezogen. Die zwei nächstgelegenen korrespondierenden Pfad-Knoten $\mathbf{p}_i^m, \mathbf{p}_{i+1}^m$ lassen sich wie folgt finden:

$$\mathbf{p}_i^m = \underset{\mathbf{p}^m \in \mathbf{M}}{\arg\min} \sqrt{(p_x^m - t_x)^2 + (p_y^m - t_y)^2 + \mathit{deg}(p_\psi - t_\psi)^2},$$

mit der Differenz der Orientierung $\mathit{deg}(p_\psi^m - t_\psi)$ in Grad, wobei der fortlaufende Index $i \in \mathbb{N}$ die Reihenfolge der Pfad-Knoten bezogen auf den euklidischen Abstand zum Trajektorienpunkt $\boldsymbol{t}_k$ beschreibt. Die gewählte Metrik ist nur eine Annäherung, vermeidet jedoch die Zuordnung von Pfad-Knoten mit einem großen Abstand und einer stark unterschiedlichen Ausrichtung.

Zusätzlich wird die Annahme getroffen, dass der Odometriefehler exponentiell mit der Länge der Trajektorie zunimmt. Aus diesem Grund wurde die Abstandsmetrik um eine exponentielle Gewichtungsfunktion $\beta_{\mathsf{tr,k}} = e^{(-\lambda_{\mathsf{tr}}||\mathbf{t}_k - \mathbf{t}_1||_2)}$ erweitert, welche die Trajektorien-Posen $\mathbf{t}_k$ entsprechend dem Abstand zur letzten Trajektorien-Pose $\mathbf{t}_1$ und dem Faktor λ_{tr} gewichtet. In Abbildung 5.5 sind die unterschiedlichen Gewichtungen durch die Radien der einzelnen Trajektorien-Posen (Blau) grafisch dargestellt.

5.8 Integration von globalen Positionsinformationen

Wenn die MCL konvergiert, sind neben der globalen Fahrzeugposition auch globale Positionsinformationen der Sensor-Landmarken bekannt. Diese globalen Positionsinformationen werden anschließend im SLAM-Graph als Bedingungen hinzugefügt, unter Berücksichtigung der Zustandsunsicherheit der geschätzten Eigenposition sowie der geschätzten Landmarkenpositionen. In Abbildung 5.6 ist der konstruierte Graph inklusive den globalen Bedingungen dargestellt.

Die Integration der globalen Positionsinformationen im Graphen-basierten Backend ermöglicht die Registrierung der mittels SLAM-Verfahren erstellten Karte - innerhalb des globalen Kartenraums.

5.8.1 Globale Positionsbedingungen

Die Bedingung zur globalen Fahrzeugpositionsschätzung sind wie folgt modelliert:

$$J_k^{\mathsf{gp}} = ||\mathbf{x}_k - \hat{\mathbf{x}}_k||^2_{\hat{\mathbf{P}}_k^{-1}} = \boldsymbol{d}_k^T \hat{\mathbf{P}}_k^{-1} \boldsymbol{d}_k, \tag{5.17}$$

mit $\hat{\mathbf{x}}_k$ als die beste Positionsschätzung der MCL und aktuellem Graph-Knoten $\mathbf{x}_k$.

5.8.2 Globale Landmarken-Bedingungen

Für die Graphen-Bedingungen der assoziierten Sensor-Landmarken zu den globalen Karten-Landmarken gilt:

$$J_{k,l}^{\mathtt{glm}} = ||\mathbf{l}_l - \hat{\mathbf{x}}_k \oplus \mathbf{z}_{k,n}^{\mathtt{lm}}||^2_{\hat{\mathbf{P}}_{k,l}^{-1}} = \boldsymbol{g}_l^T \hat{\mathbf{P}}_{k,l}^{-1} \boldsymbol{g}_l, \tag{5.18}$$

wobei die Operation $\hat{\mathbf{x}}_k \oplus \mathbf{z}_{k,n}^{\mathtt{lm}}$ die Sensor-Landmarke auf Basis der Positionsschätzung der MCL in den globalen Raum transfomiert unter Berücksichtigung der in den globalen Zustandsraum transformierten Kovarianzmatrix $\hat{\mathbf{P}}_{k,l}$ [Parois und Lutz, 2011].

Um die Konvergenz des Graph-SLAM Ansatzes zu verbessern, werden die Bedingungen für die Karten-Landmarken $\mathbf{l}_l$ jedoch nur hinzugefügt, wenn die Varianz der Positionsunsicherheit der getrackten Sensor-Landmarke $\mathbf{z}_{k,l}^{\mathtt{lm}}$ kleiner ist als die der vorherigen Schätzung $\mathbf{z}_{k-1,l}^{\mathtt{lm}}$.

5.8.3 Erweiterte Graphen-Darstellung

Die Formulierung aus Gleichung (5.10) wird somit um weitere Faktoren erweitert. Die neue Kostenfunktion des Graphen unter Berücksichtigung der Odometrie- und Landmarkenmessungen sowie den globalen Positionsinformationen ist dann wie folgt definiert:

$$\mathbf{V}^* = \arg\min_{\mathbf{V}} \sum_{k=1}^{K} J_k^{\mathtt{odo}} \sum_{k=1}^{K} \sum_{l=1}^{L} J_{k,l}^{\mathtt{lm}} + \sum_{k}^{K'} J_k^{\mathtt{gp}} + \sum_{k}^{K'} \sum_{l}^{L'} J_{k,l}^{\mathtt{glm}}. \tag{5.19}$$

Eine erfolgreiche Sensor-Landmarken zu Karten-Landmarken Assoziation erfolgt nicht zu jedem Zeitschritt k und für jede Landmarke mit Index l. Für die globalen Bedingungen werden somit nur die Menge über alle tatsächlichen Zeitschritten $K' \subseteqq K$ und Landmarken mit Index $L' \subseteqq L$ berücksichtigt.

5.9 Erweiterung um Straßenmerkmale

5.9.1 Einleitung

Um die Positionsschätzung in Bereichen zu unterstützen, in denen nur sehr wenige vertikale Landmarken vorhanden sind, wird der in den vorherigen Abschnitten beschriebene Ansatz um Straßenmerkmale erweitert.

Im Allgemeinen helfen Informationen über den Straßenverlauf die eigene Bewegung zu planen oder die von anderen Fahrzeugen zu prädizieren. Das Wissen über den wahrgenommenen Straßenverlauf, die Straßenbreite und die Anzahl der Spuren liefert darüber hinaus wichtige Informationen für eine spurgetreue Lokalisierung. Der Straßenverlauf ist jedoch nicht immer vollständig beobachtbar, beispielsweise

weil andere Verkehrsteilnehmer oder Objekte am Straßenrand die Sicht verhindern. Karten sind hier eine beliebte Wahl, um die Prädiktion zu stützen und den Suchraum für Bodenmerkmale einzuschränken.

In den letzten Jahrzehnten wurde intensiv an der Fahrbahn- und Fahrspurerkennung geforscht. Ein Überblick gibt Bar Hillel et al. [2014]. Kamera-basierte Systeme wie von Abramov et al. [2017], Behringer et al. [1992], Meis et al. [2010] eignen sich sehr gut, um Fahrbahnmarkierungen, wie sie in städtischen Gebieten und auf Autobahnen vorkommen, wahrzunehmen. Darüber hinaus kommen auch Stereokamerasysteme wie von Schreiber et al. [2014] zum Einsatz, um Straßenmarkierungen (wie Pfeile, Geschwindigkeitsbegrenzungen, Busspuren) zu erkennen und zur Lokalisierung oder für die Navigation zu verwenden.

Die Reflektivität von Straßenmarkierungen lässt sich sehr gut mit LiDAR-basierten Verfahren wahrnehmen [Kammel und Pitzer, 2008, Levinson et al., 2008, Veronese et al., 2018]. Kammel und Pitzer [2008] akkumulieren hierzu mehrere LiDAR-Messungen in einem OGM, um eine dichte Straßenkarte aus Intensitätsinformationen zu erzeugen. Mithilfe der Radon-Transformation wird die Position und Richtung der Fahrspurmarkierungen sowie die Abweichungen zu einem digitalen Straßennetz bestimmt. Veronese et al. [2018] hingegen wandeln ein Intensitäts-OGM in ein Binärbild um. Neben Heuristiken wird eine quadratische Regression verwendet, um die Polynomkoeffizienten zu finden, die am besten die Fahrspur abbilden. Neben den genannten Ansätzen unter Verwendung von maschinellem Lernen finden auch Deep-Learning-Ansätze immer weitere Verbreitung [Bar Hillel et al., 2014, Caltagirone et al., 2018, 2017, Chen et al., 2019].

Die meisten der genannten Publikationen konzentrieren sich auf die Autobahn mit mehreren Fahrspuren oder städtische Straßen mit Fahrbahnmarkierungen. Auf Nebenstraßen sowie in unstrukturierten Umgebungen sind oftmals keine Spurmarkierungen vorhanden und so müssen weitere Merkmale aus den Sensordaten extrahiert und der Straßenverlauf daraus abgeleitet werden. An Orten ohne genaue Karteninformationen, Landmarken und GNSS sind die gestellten Anforderungen an die lokale Wahrnehmung des Straßenverlaufs besonders hoch. Zusätzlich spielt die Erkennung und Unterscheidung zwischen befahrbaren Oberflächen und Hindernissen insbesondere auf unstrukturierten Straßen eine entscheidende Rolle, denn die sichere Navigation durch eine kollisionsfreie Pfadplanung ist nur durch die explizite Behandlung von Freiflächen innerhalb des Umgebungsmodells möglich.

Um in unstrukturiertem Gelände einer Straße autonom folgen zu können, fusioniert Manz et al. [2011] Punktwolken und Kamerabilder in einer grid-basierten Höhenkarte. Mithilfe eines Partikelfilters wird die Geometrie des Straßennetzes geschätzt. Zusätzlich wird das Straßennetz als Vorabinformation aus kommerziell erhältlichen GIS-Daten integriert, um eine grobe Vorstellung der Straßengeometrie zu erhalten. Der Schätzprozess nutzt dabei rand- und regionbasierte Bildmerkmale sowie Hindernisinformationen, die aus der eingefärbten OGM abgeleitet werden.

Der Ansatz von Ruchti et al. [2015] beschreibt ein Verfahren, das LiDAR-Messungen in einer OGM akkumuliert, um Segmente der Straße zu klassifizieren. Die Lokalisierung

erfolgt mit einer MCL innerhalb des OSM-Straßennetzes, wobei das Messmodell die im Grid klassifizierten Straßenzellen berücksichtigt.

Suger und Burgard [2017] stellen einen probabilistischen Ansatz zur autonomen Roboternavigation vor, der semantische Geländeinformationen aus 3D-LiDAR-Daten ableitet und sich mithilfe einer Markov-Chain-Monte-Carlo-Technik im Straßennetz von OSM lokalisiert. Dadurch ist der Roboter in der Lage, OSM für die Navigationsplanung zu nutzen und gleichzeitig bei der Ausführung von Navigationsanweisungen innerhalb der Spur zu bleiben.

Ort et al. [2018] detektieren Straßenkanten in einer 3D-Punktwolke durch die Frequenzanalyse der Entfernungsmessungen jeder Laserdiode. Mithilfe der Random Sample Consensus (RANSAC)-Methode werden Ausreißer entfernt und ein Spline für die linke und rechte Seite der Straßenbegrenzung bestimmt.

5.9.2 Rekursive Straßenverlaufsschätzung mit B-Splines

Das im Folgenden vorgestellte Verfahren hat das Ziel, den Zustand des Straßenverlaufs über die Zeit rekursiv zu schätzen, um diesem autonom folgen zu können. Im Vergleich zu den genannten Verfahren erfolgt keine globale Lokalisierung, sondern eine Offsetbestimmung (lokale Lokalisierung).

Durch die Kombination von filter-basiertem Straßentracking und Graph-SLAM kann der Positionsoffset stabil erkannt und korrigieren werden, um bei beispielsweise fehlenden vertikalen Landmarken das Fahrzeug trotzdem noch sicher in der Spur zu halten.

In Abbildung 5.7 ist die Straßenverlaufsschätzung mit B-Splines exemplarisch dargestellt. Der Straßenverlauf wird hierzu mit einem UKF und einem neuartigem Prozess- und Messmodell auf Basis einer B-Spline Repräsentation geschätzt [Burger et al., 2019b]. In unstrukturierter Umgebung ohne genormte Straßen und Wege sind B-Splines eine verbreitete Repräsentation, um komplexe Verläufe zu modellieren [Abramov et al., 2017, Alismail et al., 2014, Aly, 2008, Bibby und Reid, 2010, Burger et al., 2019b, Lu et al., 2013, Mueggler et al., 2015, Patron-Perez et al., 2015, Schindler, 2013].

5.9.2.1 Verfahrensbeschreibung

Abbildung 5.8 gibt einen Überblick über das Zusammenspiel der einzelnen Teilkomponenten des Verfahrens. Im Kontext eines SLAM-Systems gehört der Schätzprozess des Straßenverlaufs sowie die Offset-Bestimmung zum Frontend. Der Straßenverlauf (Straßenmittelpunkt) wird mit einem B-Spline modelliert, wobei die Kontrollpunkte des B-Splines und die linke und rechte Straßenbreite die Zustände des Filters sind. Der Schätzprozess erfolgt mit einem UKF, das im Messmodell sogenannte Freiraumflächen $\mathbf{a}_{k,l}$ berücksichtigt.

Hierzu werden zuerst Hypothesen aus der Menge an Freiraumflächen unter Berücksichtigung des aktuellen Straßenverlaufs ausgewählt und dann die äußeren Ränder der Fläche als Messung berücksichtigt.

Die Freiraumflächenerzeugung ist Teil der Punktwolkenvorverarbeitung, was bereits in Sektion 4.2 und Unterabschnitt 4.3.2 detailliert beschrieben wurde.

Die Prädiktion des Straßenverlaufs erfolgt entlang der gegebenen globalen Route auf Basis des OSM Straßennetzwerks oder entlang der kartierten Trajektorie des Führungsfahrzeuges. Sind keine Vorabinformationen des Straßenverlaufs verfügbar, so kann die aktuelle Krümmung und die inverse Eigenbewegung dazu verwendet werden, den Straßenverlauf vorherzusagen.

Die Verwendung der Routeninformation erhöht jedoch die Robustheit des Systems erheblich, insbesondere wenn große Bereiche der eigenen Fahrspur z. B. von anderen Fahrzeugen verdeckt sind oder wenn nicht eindeutig zwischen eigener Fahrspur und Abzweigungen bzw. Kreuzungen unterschieden werden kann.

5.9.2.2 Straßenverlaufsrepräsentation mittels B-Spline

Eine B-Spline-Kurve $B_k(u)$ besteht aus der gewichteten Summe von $n+1$ Basisfunktionen $N_{i,p}(u) \mid i = 0, \ldots, n$ mit dem Maximalgrad p ($p = 3$ für kubisch):

$$B_k(u) = \sum_{i=0}^{n} N_{i,p}(u) \cdot \mathbf{c}_{i,k}, \tag{5.20}$$

mit $n+1$ Kontrollpunkten $\mathbf{c}_{i,k}$ und den Knotenvektoren $\begin{pmatrix} u_0 & u_1 & \ldots & u_m \end{pmatrix} \mid m = n+p+1$ zum Zeitschritt k. Die Kontrollpunkte bestimmen den Verlauf der B-Spline Funktion. Der Zustand des UKF zur rekursiven Straßenverlaufsschätzung ist dann wie folgt definiert:

$$\mathbf{X}_k = \begin{pmatrix} \mathbf{x}_{0,k} & \ldots & \mathbf{x}_{n,k} \end{pmatrix}^T, \tag{5.21}$$

mit $i = 0, \ldots, n$ Unterzuständen $\mathbf{x}_{i,k} = \left(\mathbf{c}_{i,k}^T, \mathbf{w}_{i,k}^T\right)$. Jeder Unterzustand beinhaltet einen Kontrollpunkt in $\mathbf{c}_{i,k} = \begin{pmatrix} x_{i,k} & y_{i,k} \end{pmatrix}^T$ der Straßenmitte in kartesischen 2D-Koordinaten sowie die linke und rechte Breite der Fahrspur zur Fahrbahnmitte $\mathbf{w}_{i,k} = \begin{pmatrix} w_{i,k}^l & w_{i,k}^r \end{pmatrix}^T$ von jedem Kurvenpunkt, der am nächsten zum Kontrollpunkt ist. Zur besseren Veranschaulichung sind die Unterzustände in Abbildung 5.9 nochmal grafisch visualisiert. Im Folgenden wird das Prozess- und Messmodell der rekursiven Straßenschätzung beschrieben.

5.9.2.3 Prozessmodell

Auf Basis der Unterzustände (Position der Kontrollpunkte, Straßenbreite) lässt sich das Prozessmodell in zwei verschiedene Prädiktionsschritte unterteilen. Zur

Veranschaulichung sind in Abbildung 5.10 die einzelnen Schritte des Prozessmodells exemplarisch dargestellt.

Die Prädiktion der Kontrollpunkte $\mathbf{C}_k = \{\mathbf{c}_{i,k}\}_{i=0}^{n}$ erfolgt entlang des B-Splines $B_{\texttt{map}}$, welcher auf Basis der gegebenen Route erzeugt wird:

$$p\left(\mathbf{c}_{i,k} \mid \mathbf{c}_{i,k-1}, \mathbf{u_{k-1}}, B_{\texttt{map}}\right) = f(\mathbf{c}_{i,k-1}, \mathbf{u_{k-1}}, \mathbf{v}_{k-1}^{c}, B_{\texttt{map}}), \tag{5.22}$$

wobei für jeden Kontrollpunkt gilt $\mathbf{v}_{k-1}^{c} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1}^{c})$ mit Kovarianz des Prozessrauschens $\mathbf{Q}_{k-1}^{c}$.

Um die Kontrollpunkte zu prädizieren, wird als Erstes für jeden Kontrollpunkt der Knoten $\hat{u}$ ermittelt, der den Abstand zum B-Spline minimiert:

$$\hat{u} = \underset{u \in U}{\arg\min} \left\| \mathbf{c}_{i,k-1} - B_{\texttt{map}}(u) \right\|_2. \tag{5.23}$$

Anschließend wird der Kontrollpunkt auf Basis des Kurvenpunktes aktualisiert $\mathbf{c}_{i,k} = B_{\texttt{map}}(\hat{u})$, was in Abbildung 5.10b dargestellt ist.

Im nächsten Schritt wird unter Verwendung des Bewegungsmodells und $\mathbf{u_{k-1}}$ die zurückgelegte Wegstrecke geschätzt und jeder Kontrollpunkt entsprechend der ermittelten Länge entlang des Kurvenverlaufs von $B_{\texttt{map}}$ verschoben. Die prädizierten Kontrollpunkte sind in Abbildung 5.10c durch orangefarbene Punkte visualisiert.

Im zweiten Teil des Prädiktionsmodells werden die linke und rechte Breite des Straßenverlaufs vorhergesagt. Die Grundlage liefern zwei B-Splines die aus den Kontrollpunkten sowie der linken und rechten Breite des Filterzustands erzeugt werden:

$$\mathbf{w}_{i,k} = f(\mathbf{w}_{i,k-1}, \mathbf{C}_{k-1}, \mathbf{u_{k-1}}, \mathbf{v}_{k-1}^{w}) \tag{5.24}$$

mit der Kovarianz des Prozessrauschens $\mathbf{v}_{k-1}^{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1}^{w})$. Die neuen Kontrollpunkte $\tilde{\mathbf{C}}_i = \{\tilde{\mathbf{c}}_i\}_{i=0}^{n}$ des linken $B_{\texttt{Links}}(u)$ und rechten $B_{\texttt{Rechts}}(u)$ B-Splines lassen sich dann folgt erzeugen:

$$\tilde{\mathbf{c}}_i^{\{l,r\}} = {}^{k-1}\mathbf{H}_k \cdot \mathbf{c}_{i,k-1} + \mathbf{w}_{i,k-1}^{\{l,r\}} \cdot \frac{\hat{n}}{\|\hat{n}\|}. \tag{5.25}$$

Jeder Kontrollpunkt $\mathbf{C}_{k-1}$ wird dabei auf Basis der inversen Eigenbewegung ${}^{k-1}\mathbf{H}_k$ von Zeitschritt $k-1$ zu k transformiert. Die Eigenbewegung wird auf Basis der alten Position und Steuergröße $\mathbf{u_{k-1}}$ bestimmen, wobei sich der orthogonale Offset-Vektor durch Multiplikation der Breiten $\mathbf{w}_{i,k-1}^{\{l,r\}}$ und dem normalen Vektor wie folgt ergibt:

$$\hat{n} = \begin{cases} (\mathbf{c}_{i+1} - \mathbf{c}_i)^{\perp} & \texttt{wenn } \; i = 0, \ldots, n-1 \\ (\mathbf{c}_i - \mathbf{c}_{i-1})^{\perp} & \texttt{dann.} \end{cases} \tag{5.26}$$

Im letzten Schritt werden die linke und rechte Breite $\mathbf{w}_{i,k}^{\{l,r\}}$ für jeden Zustand auf Grundlage des euklidischen Abstands zwischen den neuen Kontrollpunkten

vorhergesagt $\tilde{\mathbf{c}}_i^{\{l,r\}}$ und dem nächstgelegenen Kurvenpunkt der neu erstellten B-Splines zugeordnet:

$$\mathbf{w}_{i,k}^{l} = \min_{u \in U} \|\tilde{\mathbf{c}}_i^{l} - B_{\texttt{Links}}(u)\|_2 \tag{5.27}$$

$$\mathbf{w}_{i,k}^{r} = \min_{u \in U} \|\tilde{\mathbf{c}}_i^{r} - B_{\texttt{Rechts}}(u)\|_2. \tag{5.28}$$

Die Vorhersage entlang des Straßennetzes ist in Abbildung 5.10c durch orangefarbene Punkte dargestellt. Die prädizierten Kurvenpunkte werden dann anschließend im Messmodell verwendet.

5.9.2.4 Messmodel

Freiraumflächen sind zusammengehörige Bodenflächen, die in einem Vorverarbeitungsschritt aus der Punktwolke extrahiert wurden und nun als Eingangsmessung im Messmodell berücksichtigt werden. Zur Vereinfachung werden im Folgenden nur Freiraumflächen betrachtet, die vor dem Fahrzeug liegen. Der Prozess zur Freiraumflächenerzeugung ist in Unterabschnitte 4.2.3 und 4.3.2 beschrieben.

Um aus der Menge von Freiraumflächen diejenige zu wählen, die am besten zum aktuellen Zustand der Straße passt, wird die Geometrie der äußeren Ränder jeder Freiraumfläche $\mathbf{a}_{k,l}$ mithilfe einer gewichteten quadratischen Kurve $\mathrm{q}_{k,l}$ approximiert. Diese Approximation ist im Vergleich zu B-Splines deutlich schneller und für den gewählten Ansatz ausreichend.

Um die Kurve entlang der äußeren Knoten $\mathbf{y}_u = \begin{pmatrix} x_u & y_u \end{pmatrix}^T$ jeder Freiraumfläche zu bestimmen, wird die Anordnung der Graph-Knoten innerhalb der Graphen-Struktur berücksichtigt. Eine beispielhafte Darstellung der Eckpunkte (Blau) ist in Abbildung 5.11b gezeigt. Im folgenden Abschnitt werden die Indizes k, l zur besseren Lesbarkeit weggelassen.

Die Formulierung der Methode der kleinsten Quadrate ist wie folgt definiert:

$$\min \sum_{j=1}^{n} w_j \left(y_j - (ax_j^2 + bx_j + c)\right)^2, \tag{5.29}$$

über die Anzahl von n Graphknoten. Die Initialisierung der Gewichte w_j erfolgt unter Berücksichtigung der quadratischen euklidische Distanz $d_j = \min_{u \in U} \|\mathbf{y}_j - B_k(u)\|_2$ zwischen Kanten-Knoten $\mathbf{y}_j$ und dem korrespondierenden Kurvenpunkt $B_k(u_i)$ mit dem geringsten Abstand:

$$w_j = \begin{cases} 1 - \frac{e^{d_j^2}}{\sum_{j=1}^{n} w_j} & \texttt{wenn} \quad 1 - \frac{e^{d_j^2}}{\sum_{j=1}^{n} w_j} > 0 \\ 0 & \texttt{ansonsten.} \end{cases} \tag{5.30}$$

Um die Robustheit zu erhöhen, wird der Likelihood $p\left(\mathtt{q} \mid B_k\left(u\right)\right)$ gesucht, der die Kurvenfunktion $\mathtt{q}(x) = ax^2 + bx + c$ am besten approximiert:

$$p\left(\mathtt{q} \mid B_k\left(u\right)\right) = e^{-\frac{1}{s_{max}} \sum_{x \in X} \min\limits_{u \in U} \|c_p(x) - B_k(u)\|_2}, \tag{5.31}$$

unter Berücksichtigung aller Kurvenpunkte $c_p(x) = \begin{pmatrix} x & \mathtt{q}(x) \end{pmatrix}^T$ zum B-Spline $B_k\left(u\right)$ des aktuellen Zustands und normalisiert anhand der Anzahl an Stichproben. Der Abstand zwischen den gesampelten Stichproben ist konstant gewählt $s_d = 0.25\,\mathrm{m}$ und die Menge an Stichproben ist wie folgt definiert:

$$X = \bigcup_{s=0}^{s_{max}} \{x_{0,k} + s \cdot s_d\}, \tag{5.32}$$

mit $s_{max} = \left\lfloor {|x_{n,k} - x_{0,k}|}/{s_d} \right\rfloor$ unter Berücksichtigung der maximalen Distanz in X-Richtung zwischen dem ersten und letzten Kontrollpunkt $x_{0,k}, x_{n,k}$.

Um aus der Menge an linken und rechten Kurvenelementen die Kurve auszuwählen, welche am besten zum aktuellen Straßenverlauf passt, wird der Likelihood $p(\mathtt{q}_{k,l} | B_k\left(u\right))$ in Abhängigkeit der linken $Q_k^{\mathtt{Links}}$ und rechten $Q_k^{\mathtt{Rechts}}$ quadratischen Kurven bestimmt. Die optimale linke Kurve $q_{k,l_{\mathtt{Links}}}$ bestimmt sich dann wie folgt:

$$\mathtt{q}_{k,l_{\mathtt{Links}}} = \underset{\mathtt{q}_{k,l} \in Q_k^{\mathtt{Links}}}{\arg\max}\, p(\mathtt{q}_{k,l} | B_k\left(u\right)), \tag{5.33}$$

wobei das gleiche Verfahren für alle rechten Kurven $Q_k^{\mathtt{Rechts}}$ angewendet wird, um die optimale rechte Kurve $\mathtt{q}_{k,l_{\mathtt{Rechts}}}$ auszuwählen.

Für jeden Kurvenpunkt $\mathtt{q}_{k,l}(x)$ wird der Kontrollpunkt des aktuellen Zustands $\mathbf{c}_{i',k}$ mit Index i' gesucht, der den Abstand wie folgt minimiert:

$$i' = \underset{i \in \{0,\ldots,n\}}{\arg\min} \left(\min_{u_i \in U_i} \|\mathtt{q}_{k,l} - B_k(u_i)\|_2 \right), \tag{5.34}$$

mit $U_i = \begin{bmatrix} u_i & u_{i+p+1} \end{bmatrix}$ als das u-Intervall des Kontrollpunktes mit Index i. Anschließend sagt die Messfunktion $g_{k,l}(\mathbf{w}_{i',k}, \mathtt{q}_{k,l})$ die Zustandsbreite des Index i' voraus, wobei berücksichtigt wird, ob der Kurvenpunkt zur linken oder rechten Approximation gehört:

$$g_{k,l}(\mathbf{w}_{i',k}, \mathtt{q}_{k,l}) = \begin{cases} w^l_{i',k} & \texttt{if } \mathtt{q}_{\mathtt{k,l}} \in \text{linker B-Spline} \\ w^r_{i',k} & \texttt{if } \mathtt{q}_{\mathtt{k,l}} \in \text{rechter B-Spline.} \end{cases} \tag{5.35}$$

Die Breite wird durch den euklidischen Abstand von $\mathtt{q}_{k,l}$ zum nächstgelegenen Kurvenpunkt der B-Spline $B_k\left(u\right)$ des Zustands gemessen:

$$\tilde{y}_{kl} = \min_{u_i \in U_i} \|\mathtt{q}_{k,l} - B_k(u_i)\|_2 \tag{5.36}$$

und die Messinnovation wird dann bestimmt durch $\delta = (\tilde{y}_{k,l} - g_{k,l}(\mathbf{w}_{i',k}, \mathtt{q}_{k,l}))$.

Abbildung 5.11 zeigt die einzelnen Verarbeitungsschritte des Straßentrackings, die B-Spline Repräsentationen, den Prädiktionsschritt sowie die Messungen und der Updateschritt grafisch und chronologisch von links nach rechts angeordnet.

5.9.3 Bestimmung und Korrektur der Fahrzeugposition

Im letzten Schritt wird der Offset zwischen dem ermittelten Straßenverlauf und der gegebenen globalen Route ermittelt. In Abbildung 5.12 ist das Ergebnis des rekursiven Straßentracking-Verfahrens mittels B-Splines gezeigt, mit linker (Rot), mittlerer (Weiß) und rechter (Grün) Fahrbahnkante sowie den 3D-LiDAR-Messungen einer Freiraumfläche (schwarze Punkte), die die Grundlage des Straßentrackings bilden.

Der globale Offset entspricht dabei einer Transformation, welche die aktuelle Schätzung der Straßenmitte $\mathtt{B}_{center}$ (weiße Linie) am besten auf den aktuell sichtbaren Routenabschnitt $\mathtt{B}_{\mathtt{map}}$ (blau gestrichelte Linie) unter Berücksichtigung der globalen Fahrzeugposition transformiert.

Hierzu ist, wie in Abbildung 5.12 dargestellt, die globale Route $\mathtt{B}_{\mathtt{map}}$ im Koordinatensystem der aktuellen Positionsschätzung repräsentiert. Der Prozess sowie die Graphen-Erstellung sind exemplarisch in Abbildung 5.13 dargestellt.

Um die 3x3 Transformationsmatrix $\mathbf{T}$ mit einem ICP-Verfahren zu bestimmen [Besl und McKay, 1992], werden zur Initialisierung äquidistant abgetastete Kurvenpunkte der B-Splines $\mathbf{B}_{\mathtt{center}}, \mathbf{B}_{\mathtt{map}}$ zum jeweiligen nächsten Nachbarn (kleinste euklidische Distanz) unter Berücksichtigung der aktuellen Fahrzeugausrichtung assoziiert.

Die ermittelte Transformation und Kovarianz des ICP-Verfahrens nach Censi [2007] wird im Backend wie folgt modelliert:

$$J_k^{\mathtt{icp}} = \left(\mathbf{x}_k - \mathbf{T}_k \mathbf{x}_k\right)^T \boldsymbol{\Sigma}_k^{-1} \left(\mathbf{x}_k - \mathbf{T}_k \mathbf{x}_k\right), \tag{5.37}$$

mit der inversen Kovarianzmatrix $\boldsymbol{\Sigma}_k^{-1}$ der ICP-Methode. Die Fehlerfunktion $\mathbf{x}_k - \mathbf{T}_k \mathbf{x}_k$ beschreibt dabei den Offset zwischen der aktuellen Fahrzeugpose $\mathbf{x}_k$ und der um den Offset korrigierten Pose $\mathbf{T}_k \mathbf{x}_k$.

Um die Konvergenz des Graph-SLAM Verfahren zu verbessern und Ausreißer zu eliminieren, werden die Bedingungen $J_k^{\mathtt{icp}}$ nur zum Graphen hinzugefügt, wenn der Fehler einen gewissen Schwellenwert nicht überschreitet. Dies ist besonders in Szenen wichtig, in denen der aktuell wahrgenommene Straßenverlauf stark vom gegebenen Straßenverlauf abweicht.

Um den Straßenverlauf zu kartieren, wird kontinuierlich ein Kurvenpunkt $\mathbf{q}_k = B_{map}(u)$ zum Graphen hinzugefügt, wobei der Kurvenpunkt am Anfang des B-Splines gewählt wird, mit $u = 0.2$, da sich das Filter über den Zustand am Anfang des B-Splines am sichersten ist. Dies liegt insbesondere am Mess- und Prozessmodell, da durch die Prädiktion auf Basis der inversen Eigenbewegung die weiter entfernten Zustände einen direkten Einfluss auf die Zustände mit kleiner werdendem Abstand erwirken.

Der Fehler $\mathbf{q}_k - h(\mathbf{x}_k, \mathbf{q}_k)$ stellt die Diskrepanz zwischen dem Kurvenpunkt der Karte in globalen Koordinaten $\mathbf{q}_k$ und dem erwarteten Kurvenpunkt $h(\mathbf{x}_k, \mathbf{q}_k) =$

$B_{map}(u = 0.2)$ auf Basis der aktuell geschätzten Fahrzeugposition $\mathbf{x}_k$ dar. Die Pose-zu-Kurvenpunkt-Bedingung wird wie folgt modelliert mit Einheitsmatrix $\boldsymbol{I}$:

$$J_k^{\texttt{road}} = (\mathbf{q}_k - h(\mathbf{p}_k, \mathbf{q}_k))^T \, \boldsymbol{I} \, (\mathbf{q}_k - h(\mathbf{p}_k, \mathbf{q}_k)) \,. \tag{5.38}$$

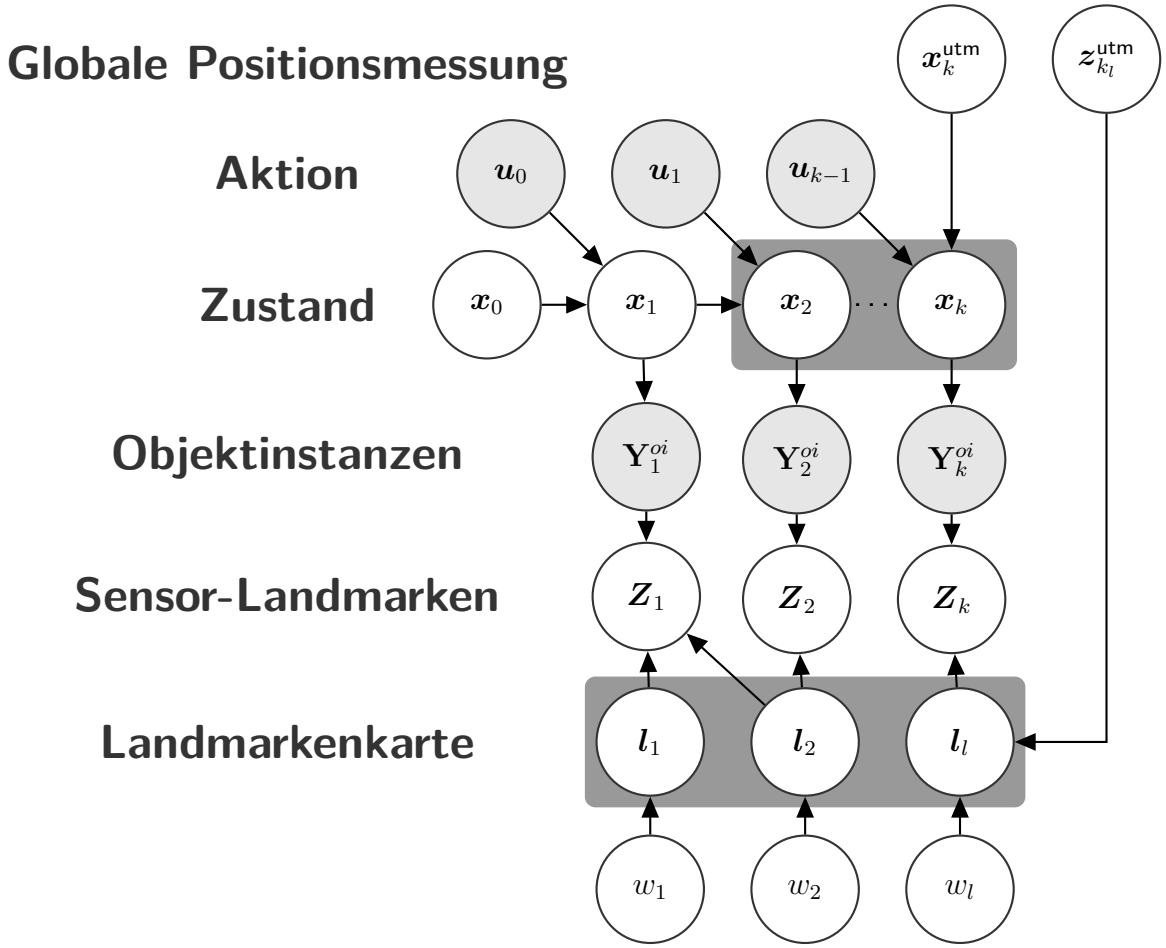


Abbildung 5.3:
Vereinfachte Repräsentation des SLAM-Verfahrens als dynamisches Bayes'sche Netzwerk. Die grauen Knoten repräsentieren die Beobachtungen (Sensormessungen), während weiße Knoten unbeobachtete Zufallsvariablen darstellen. Beim Graph-SLAM Verfahren wird die aktuellste Position $\boldsymbol{x}_k$ und vergangene Posen (Sliding-Window Ansatz) sowie die Karte $\boldsymbol{M} = \{\boldsymbol{l}_1, \ldots, \boldsymbol{l}_l\}$ gleichzeitig bestimmt, unter Berücksichtigung der latenten Variablen $w_1, \ldots, w_l$, welche die Wahrscheinlichkeiten über alle korrespondierenden Sensor-Landmarken zu den Karten-Landmarken beschreiben. Die Sensor-Landmarken werden auf Basis der Objektinstanzen und durch eine anschließende rekursive Filterung mit einem MTT-Verfahren bestimmt. Eine Untermenge der Sensor-Landmarken wird auf Basis der Objektzustände als Karten-Landmarken ausgewählt und die finale Position im Optimierungsschritt des Graph-SLAM Backend ermittelt. Parallel dazu wird die globale Fahrzeugposition sowie globale Landmarkenpositionen $\boldsymbol{x}_k^{\text{utm}}, \boldsymbol{z}_{k_l}^{\text{utm}}$ durch die MCL auf Basis der Sensor-Landmarken, der vergangenen Trajektorie sowie der gegebenen Pfad-Karte geschätzt und als globale Bedingungen im Graphen berücksichtigt. Die globalen Positionen $\boldsymbol{x}_k^{\text{utm}}, \boldsymbol{z}_{k_l}^{\text{utm}}$ sind ebenfalls wieder als ein in sich geschlossenes Hidden Markov Model (HMM) abbildbar, jedoch wurde die Darstellung an dieser Stelle wegen der besseren Übersichtlichkeit weggelassen. Konvergiert die MCL, so sind die assoziierten Zustände $\boldsymbol{x}_k$ direkt von den globalen Positionen abhängig.

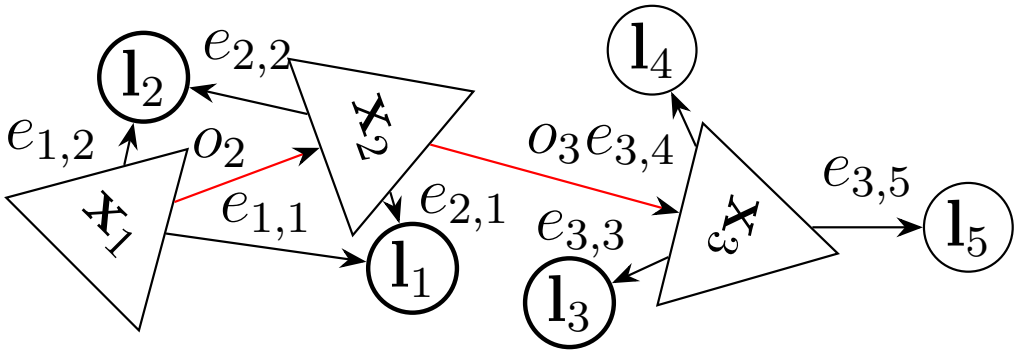


Abbildung 5.4:
Diese Abbildung visualisiert die Knoten und Kanten des Graphen-basierten Backend. Dreiecke repräsentieren Fahrzeugpositionen $\mathbf{x}_{1:k}$ und Kreise Landmarkenpositionen $\mathbf{l}_{1:L}$. Die roten Kanten repräsentieren die Odometriemessungen und schwarze Kanten stellen die Landmarkenmessungen dar.

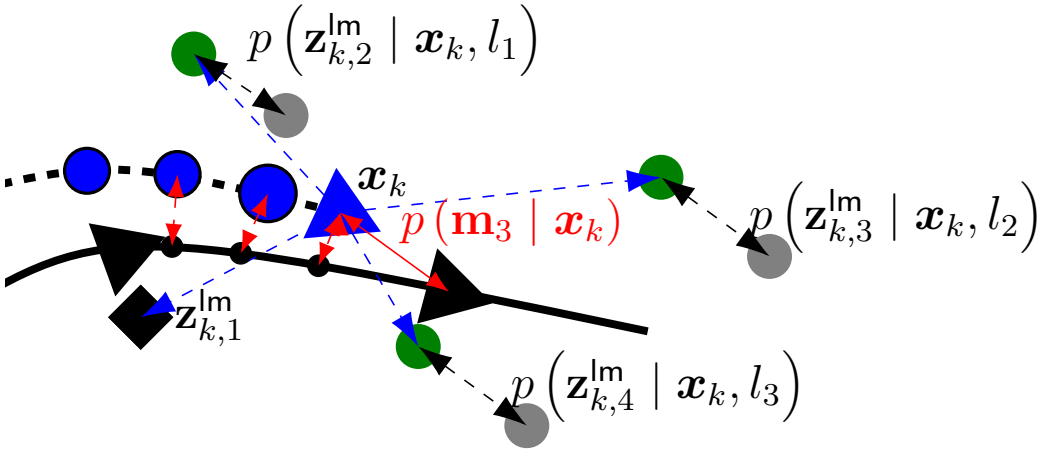


Abbildung 5.5:
Verbesserung der Lokalisierung durch Berücksichtigung der vergangenen Trajektorie. Jedes Partikelgewicht ist im Wesentlichen eine Funktion seiner räumlichen Nähe zur Pfad-Karte (hier in schwarz dargestellt). Die räumliche Zuordnung wird durch die roten Linien visualisiert, wobei jeder Trajektorienpunkt (Blau) ein zusätzliches Gewicht erzeugt. Das bestimmte Gewicht $p\left(\mathbf{m}_3 \mid \boldsymbol{x}_k\right)$ gibt Auskunft, wie gut der Pfad-Knoten $\mathbf{m}_3$ mit dem Partikelzustand $\boldsymbol{x}_k$ und der vergangenen Trajektorie (blaue Punkte) übereinstimmt. Die Durchmesser der blauen Trajektorienpunkte stellen darüber hinaus die Gewichte dar, die mit größer werdendem Abstand geringer werden. Um die Robustheit zu erhöhen, wird eine Maximum-Likelihood-Methode verwendet, bei der nur diejenigen Sensor-Landmarken (grüne Punkte) berücksichtigt werden, die die Gesamtwahrscheinlichkeit über alle Sensor-Landmarken und Karten-Landmarken maximieren, während alle anderen nicht berücksichtigt werden (schwarzes Quadrat).

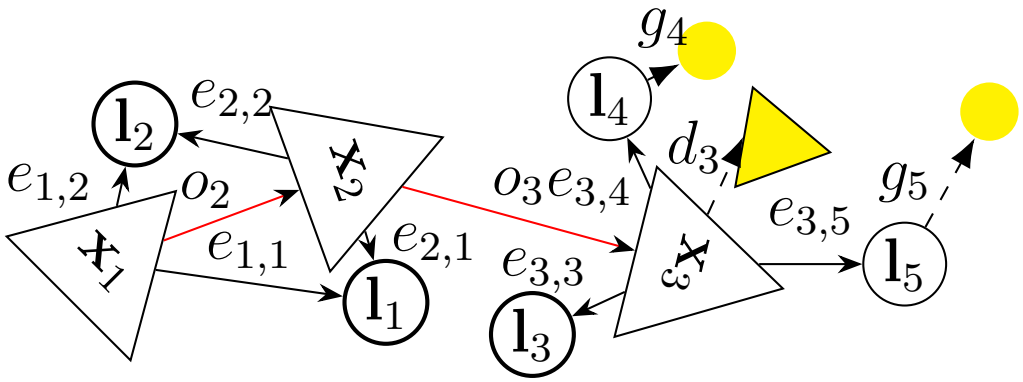


Abbildung 5.6:
Darstellung des erweiterten Graphen um globale Bedingungen, die anhand der MCL ermittelt wurden. Die gelb markierten Symbole repräsentieren dabei die globale Position, der mittels Kante verbundenen Knoten.

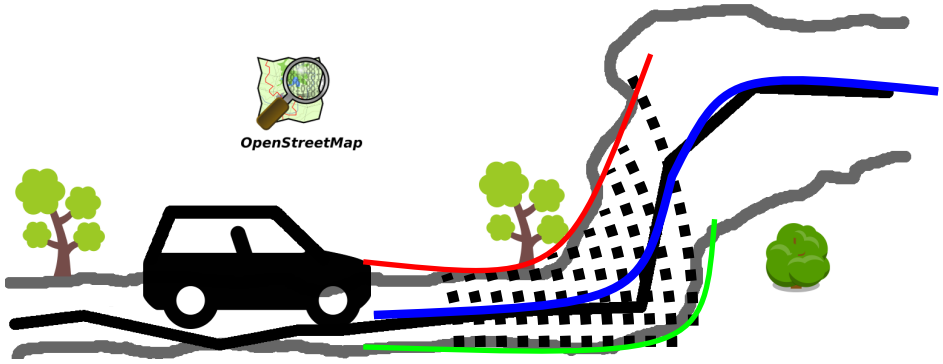


Abbildung 5.7:
Straßenverlaufsschätzung mit B-Splines zur Offsetbestimmung der Fahrzeugpose.

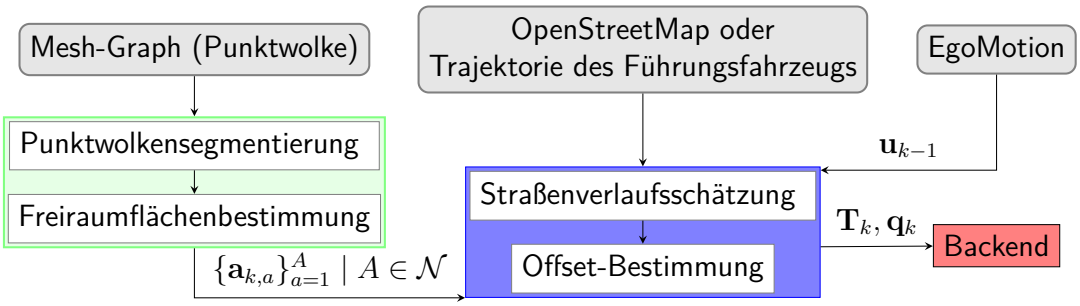


Abbildung 5.8:
Überblick der einzelnen Komponenten, die für die Offsetbestimmung nötig sind. Die Punktwolken-Vorverarbeitung ist in Grün, das Frontend in Blau und das Backend des SLAM-Verfahrens in Rot dargestellt. Die Aktualisierung der Karte erfolgt im Backend.

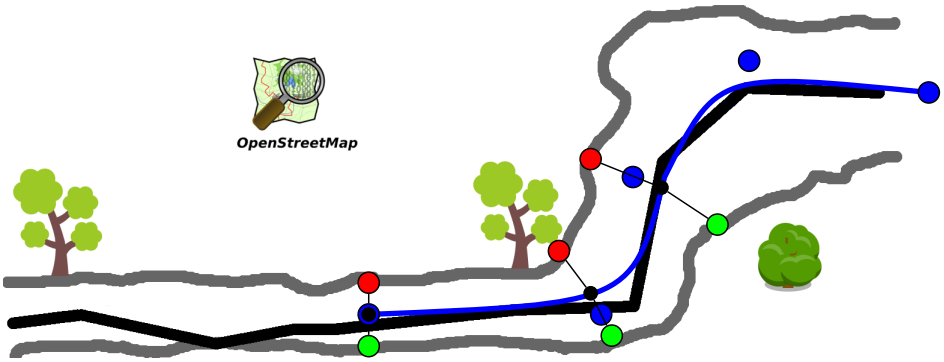


Abbildung 5.9:
Straßenrepräsentation mit einem kubischen B-Spline. Die schwarze Linie repräsentiert die globale Route zum Ziel. Die blaue Linie stellt den B-Spline des aktuellen Zustandes des UKF dar, mit Kontrollpunkten (Blau) sowie linker und rechter (Rot, Grün) Fahrbahnbreite zu jedem Kontrollpunkt.

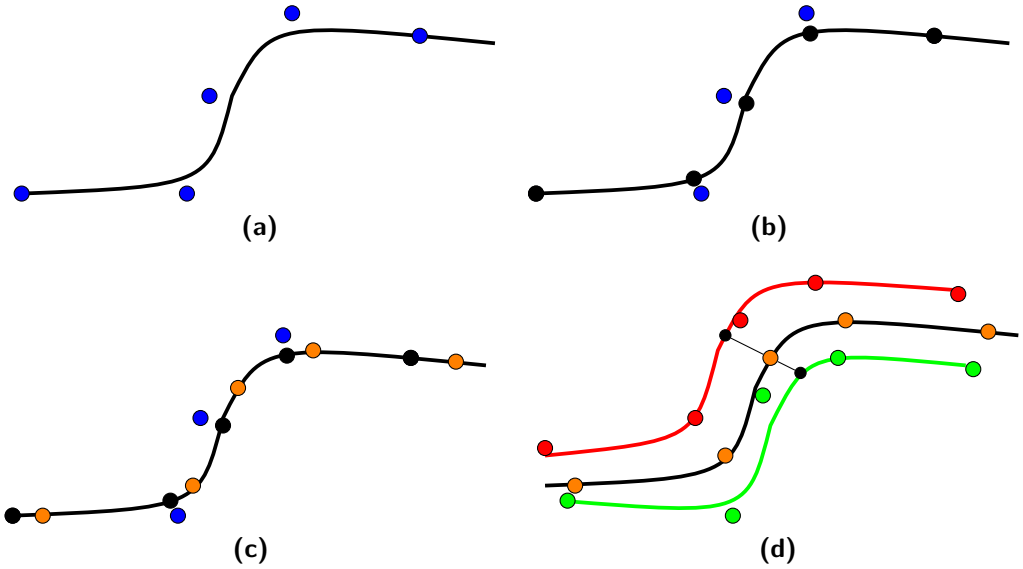
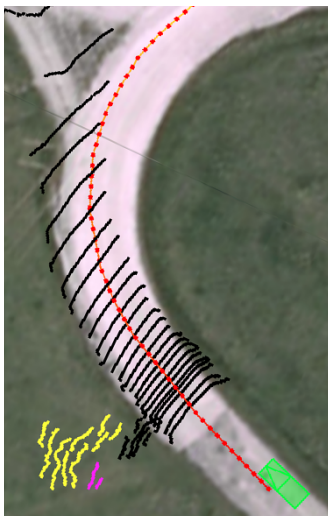
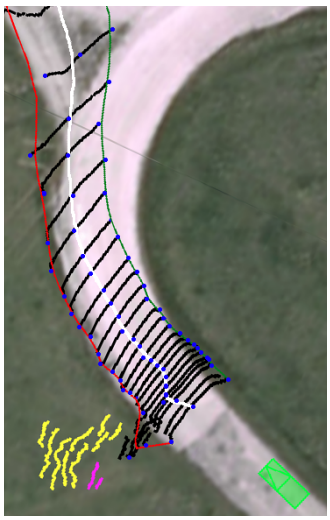


Abbildung 5.10:
Darstellung der Verarbeitungsschritte des Prozessmodells zur Prädiktion der Kontrollpunkte und Straßenbreite entlang eines Pfades (schwarz). Abbildung 5.10a zeigt die Kontrollpunkte (Blau) des aktuellen Zustands. Zur Prädiktion der Kontrollpunkte werden im ersten Schritt die nächsten Kurvenpunkte bestimmt (schwarze Punkte), siehe Abbildung 5.10b und entlang des gegebenen Pfades (schwarze Linie) prädiziert, mithilfe der Steuergröße. Das Ergebnis der Prädiktion ist durch orangefarbene Punkte in Abbildung 5.10c dargestellt. Im letzten Schritt erfolgt die Prädiktion der linken (Rot) und rechten (Grün) Straßenbreite auf Basis der prädizierten Knotenpunkte (Orange) und der Straßenbreite des vorherigen Zustandes.

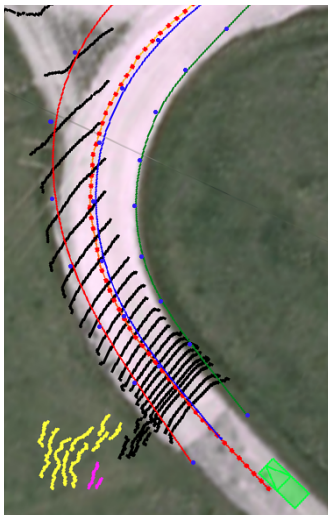


(a) Freiraumflächen

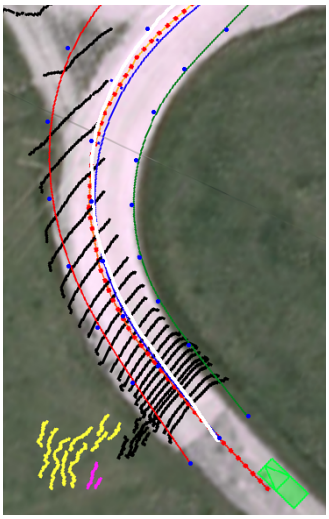


(b) 3D-Messungen

(c) Kurvenapproximation



(d) Prädiktion



(e) Korrektur

Abbildung 5.11:
Darstellung der verschiedenen Verfahrensschritte des Straßentrackings (von links nach rechts). Die grüne Bounding-Box repräsentiert die Position und Orientierung des Fahrzeugs. 5.11a zeigt die Freiraumflächen (schwarz, gelb, magenta), die aus der Punktwolke gewonnen werden. Die orangefarbene Linie mit roten Punkten repräsentiert die globale Route $B_{\mathtt{map}}$, der zu folgen ist. In 5.11b ist die Bestimmung der linken (Rot) und rechten (Grün) äußeren Begrenzungspunkte, exemplarisch für die schwarze Freiraumfläche, gezeigt. Die Begrenzungspunkte werden anschließend als Messung berücksichtigt. Die weiße Linie repräsentiert den Mittelpunkt der äußeren Begrenzungspunkte. 5.11c zeigt das Ergebnis der Vorauswahl der Freiraumflächen auf Basis der gegebenen Route und die Repräsentation der äußeren Ränder durch eine gewichtete quadratische Kurve. Abbildung 5.11d visualisiert das Ergebnis der Prädiktion des UKF basierend auf dem aktuellen Zustand (Blau) der B-Splines-Darstellung. Die linken (Rot) und rechten (Grün) B-Splines werden aus der B-Spline des Zustands (Blau) und der Breiteninformation für jeden Kontrollpunkt gewonnen, wobei die blauen Punkte die Kontrollpunkte aller dargestellten B-Splines repräsentieren. Abbildung 5.11e zeigt den Korrekturschritt des Filters, indem die Messungen aus 5.11c berücksichtigt werden, wobei die weiße Linie den korrigierten B-Spline des Zustands repräsentiert.

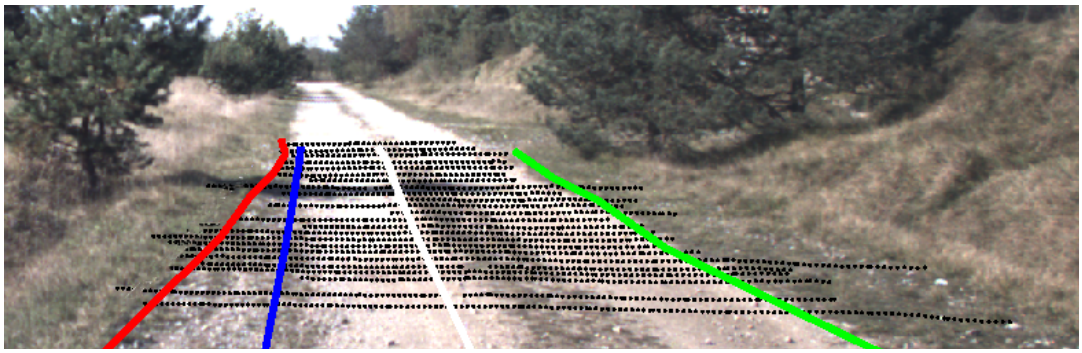


Abbildung 5.12:
Tracking des Straßenverlaufs in einem unstrukturierten Bereich, in dem der geschätzte Straßenverlauf (Weiß) sowie der linke und rechte Fahrbahnrand (Rot, Grün) durch B-Splines approximiert werden. Die Route (Blau) und die relevanten LiDAR-Messungen (schwarze Punkte) werden unter Berücksichtigung der aktuellen globalen Fahrzeugposition, der Kamera und der LiDAR-Kalibrierung (nur zur Visualisierung) in das Bild projiziert. Es ist deutlich zu sehen, dass bei der aktuellen Fahrzeugposition ein seitlicher Versatz zwischen dem geschätzten Zentrum (Weiß) und der Straße aus OSM (Blau) besteht. Dieser seitliche Versatz wird in einer Graph-SLAM Formulierung modelliert, um die Fahrzeugtrajektorie korrekt zu bestimmen und um den Straßenverlauf für die zukünftige Nutzung zu speichern.

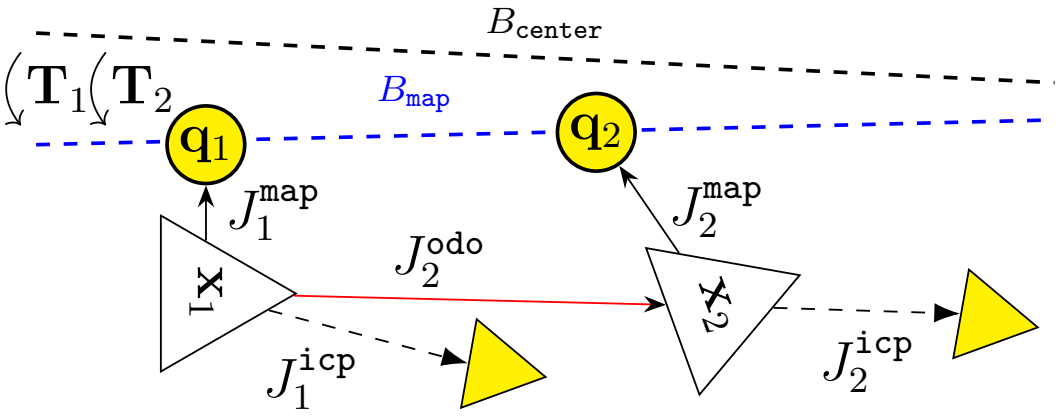


Abbildung 5.13:
Erweiterung des SLAM-Graphen um den Straßenverlauf sowie den daraus ermittelten globalen Positionsoffset über zwei aufeinander-folgende Zeitschritte $k = 1$ und $k = 2$. Zur besseren Übersicht sind die Sensor-Landmarken und Positionsbedingungen aus Sektion 5.8 nicht visualisiert.

5.10 Lösen des Optimierungsproblems

Nachdem nun sämtliche Knoten und Bedingungen zum Graphen hinzugefügt wurden, kann Gleichung (5.19) um die Straßen-abhängigen Offset-Bedingungen erweitert werden.

Der Graph besteht nun aus der Summe folgender Nebenbedingungen:

$$\mathbf{V}^* = \arg\min_{\mathbf{V}} \underbrace{\sum_{k=1}^{K} J_k^{\texttt{odo}} + \sum_{k=1}^{K}\sum_{l=1}^{L} J_{k,l}^{\texttt{lm}}}_{Landmarken-SLAM} + \underbrace{\sum_{k}^{K'} J_k^{\texttt{gp}} + \sum_{k}^{K'}\sum_{l}^{L'} J_{k,l}^{\texttt{glm}}}_{MCL} + \underbrace{\sum_{k}^{K''} J_k^{\texttt{icp}} + \sum_{k}^{K''} J_k^{\texttt{road}}}_{Offset-Bestimmung}, \tag{5.39}$$

wobei $K'' \subseteqq K$ die Untermenge aller Zeitschritte K beschreibt, an denen die Straßenmerkmale verfügbar waren.

Die Minimierung dieser Summe von nichtlinearen quadratischen Gleichungen erfolgt mit dem g2o-Framework von Kümmerle et al. [2011] unter Verwendung des Levenberg-Marquardt Algorithmus. Die optimierte Konfiguration der Graph-Knoten $\mathbf{V}^* = \arg\min_{\mathbf{V}} J(\mathbf{V})$ entspricht dann der optimierten Landmarkenpositionen $\mathbf{l}_{1:L}^*$ und der optimierten Fahrzeugtrajektorie $\mathbf{x}_{1:K}^*$ unter Minimierung aller Messungen.

Der Prozess wird in zwei Stufen durchgeführt. Zuerst wird die beste Konfiguration aller Posen und Landmarkenpositionen der lokalen Karte basierend auf den Sensor-Landmarken und den Odometriemessungen bestimmt, siehe Gleichung (5.10). Das Ergebnis dient als initialer Graph, der um die globalen Positions-Bedingungen erweitert und optimiert wird.

Durch den zweistufigen Prozess und die stärkere Gewichtung der mittels SLAM-Verfahren erstellten Karte konnte der Gesamtfehler reduziert und die Konvergenz des Verfahrens deutlich beschleunigt werden. Zusätzlich erfolgt die Graphen-Optimierung unter Berücksichtigung eines Sliding Window Ansatzes, bei dem nur ein Teil der gefahrenen Trajektorie sowie die zugehörigen Landmarken optimiert werden. Dies erfolgt unter der Annahme, dass immer nur lokale Teilstücke des Graphen stochastisch abhängig voneinander sind.

Die Pose $\hat{\mathbf{x}}_k$ zum Zeitschritt k entspricht der besten Schätzung der aktuellen Fahrzeugposition auf Basis der Messungen und den externen Kartendaten.

6 Geo-Informations-System

Inhalt

Ein Geographic Information System (GIS) ist nach Maguire [1991], Vatsavai et al. [2012] ein System zur Organisation, Analyse und Bearbeitung von räumlichen Daten. GIS werden in vielen Branchen und zu verschiedenen Verwendungszwecken eingesetzt, unter anderem in der Kartografie, Geographie, Logistik und auch in autonomen Fahrzeugen.

Für den in dieser Arbeit vorgestellten Ansatz wird ein GIS benötigt, das Landmarken und Fahrzeugposen sowie weitere Kartendaten effektiv speichern und laden kann. Die Hauptaufgabe des GIS ist die zur Lokalisierung benötigten Kartendaten dynamisch im Umkreis des Fahrzeugs zu laden.

Die Basis des GIS ist PostgreSQL, eine objektrelationale Datenbank [The PostgreSQL Global Development Group, 2021]. PostgreSQL unterstützt alle Funktionen des Structured Query Language (SQL) Standards sowie die Erstellung neuer Datentypen. So können selbst definierte Objekttypen wie Punktwolken, Landmarkenobjekte und Kamerabilder gespeichert werden [Ikawa et al., 2019].

PostgreSQL bietet darüber hinaus die Möglichkeit, den Funktionsumfang mit Erweiterungen zu vergrößern [Solarz und Szymczyk, 2020, Vatsavai et al., 2012]. PostGIS beispielsweise implementiert den vom Open Geospatial Consortium (OGC) definierten Standard (Simple Features for SQL) und erweitert den Funktionsumfang von PostgreSQL für Geodaten [Fröhlich, 2018].

Die großen Vorteile der geometrischen Erweiterung sind, dass die Größe der geladenen Karte sich nur marginal auf die Suchleistung auswirkt. Außerdem lassen sich neben Geometrie-Datentypen wie Punkt, Linie und Polygon auch weitere Datentypen und Funktionen implementieren.

Einer sehr hilfreiche Funktion für SLAM-Anwendungen sind insbesondere die Berechnung von Distanzen und die Möglichkeit, Objekte im definierten Umkreis von räumlichen Positionen und in Abhängigkeit der aktuellen Orientierung zu suchen [Hellerstein et al., 1995]. Berechnungen, wie eine Radiussuche oder die Suche der k-Nächsten-Nachbarn, werden von der Datenbank direkt ausgeführt und das Ergebnis direkt übermittelt. Dies ermöglicht relevante Landmarken aus der Karte beispielsweise

in Abhängigkeit des Sensorsystems sowie der aktuellen Position und Blickrichtung dynamisch zu laden.

Die Kombination von PostgreSQL und PostGIS sind somit ein wirkungsvolles Werkzeug für SLAM-Anwendungen, siehe Fröhlich [2018].

6.1 Verwendung von externen Karteninformationen

6.1.1 Einleitung

Ein Forschungsziel dieser Arbeit ist die Untersuchung und Erprobung, inwieweit frei zugängliches Kartenmaterial zur Lokalisierung in unstrukturierten Gebieten verwendet werden kann.

Die Hauptmotivation für die Verwendung von externem Kartenmaterial besteht darin, die verfügbaren Zusatzinformationen als Vorwissen bei der Objekterkennung, Datenassoziation aber auch bei der globalen Lokalisierung zu berücksichtigen, obgleich mit einer höheren Unsicherheit in der Positionsgenauigkeit und der semantischen Beschriftung zu rechnen ist [Canavosio-Zuzelski et al., 2013, Haklay, 2010, Scioscia et al., 2014].

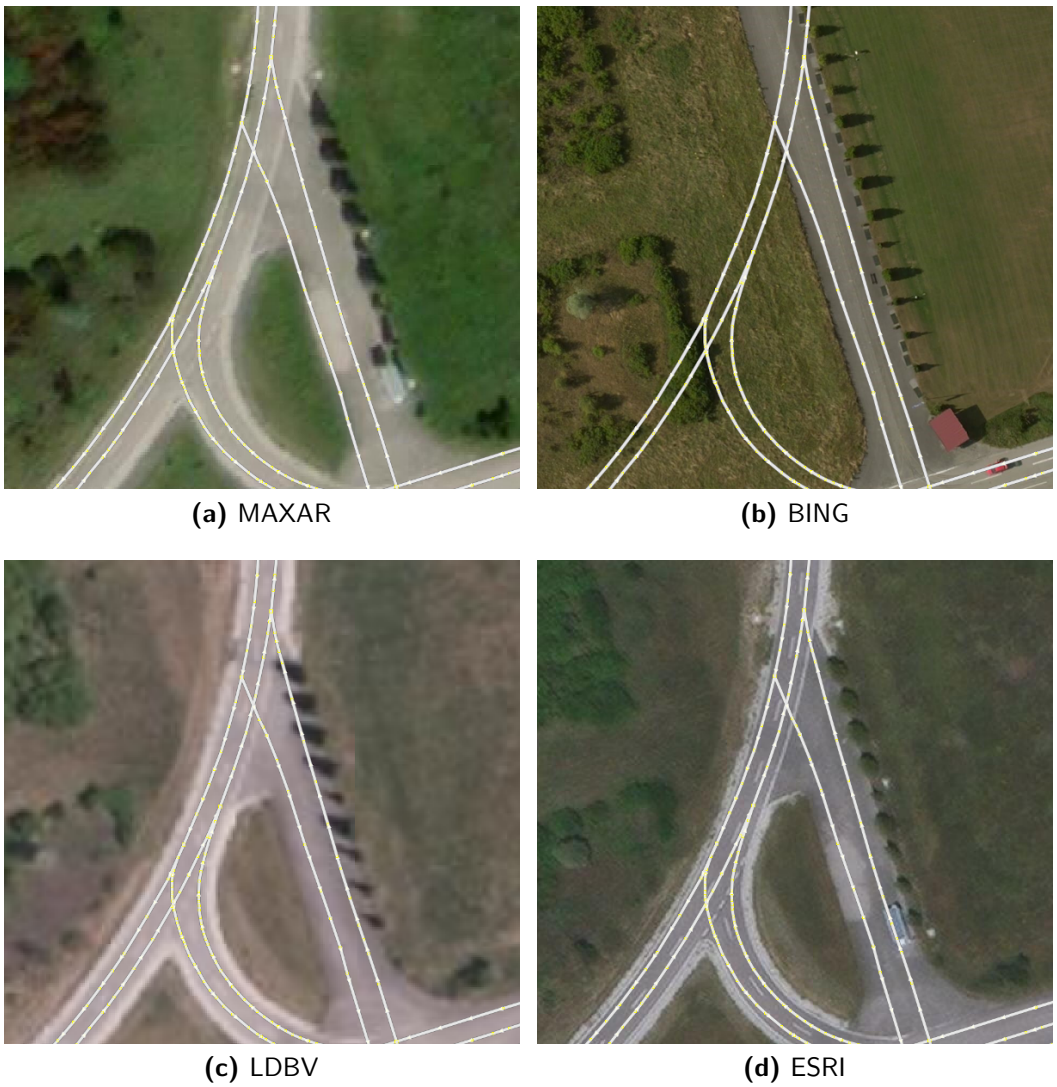
Für die Eigenpositionsbestimmung im unstrukturierten Bereich werden insbesondere Kartenlandmarken benötigt, die längerfristig ihr Erscheinungsbild erhalten, robust gegenüber den verschiedenen Jahreszeiten sind und die sich mit einem LiDAR-Sensor wahrnehmen lassen. Zusätzliche Informationen über befahrbare Straßen und Wege können in der Pfadplanung aber auch zur Lokalisierung verwendet werden.

Die Auswahlkriterien für die Verwendung externer Kartenquellen sind somit neben der Verfügbarkeit von Merkmalen, die sich mittels Onboard-Sensorik erfassen lassen, die Kartenauflösung bzw. Positionsgenauigkeit der Kartenmerkmale [Scioscia et al., 2014]. Landesgrenzen oder Naturschutzgebiete sind dabei weniger relevante Kartenmerkmale, da diese mit Sensoren nur im Ausnahmefall, z. B. durch ein Schild, wahrgenommen werden können.

Darüber hinaus ist bei den Kartenobjekten die korrekte semantische Bezeichnung wichtig. Falsche Objektklassen können zur Verwendung von Landmarken führen, die sich mittels Onboard-Sensorik nicht wahrnehmen lassen. Dies kann zu einer erhöhten Anzahl von inkorrekten Datenassoziationen führen, was sich wiederum auf den Positionsfehler auswirkt und bis hin zur Divergenz des Filters führen kann.

6.1.2 Verwendung und Integration von OpenStreetMap

OpenStreetMap (OSM) ist ein Open-Source-Projekt, bei dem täglich weltweite Nutzer neue Daten hinzufügen und so gemeinschaftlich eine digitale Karte erstellen. Die Karte beinhaltet neben vertikalen Objekten, die als Landmarke in Frage kommen, auch semantische Klassen der Objekte. Ein Großteil der Daten wird dabei händisch



(a) MAXAR **(b)** BING

(c) LDBV **(d)** ESRI

Abbildung 6.1:
Eine Übersicht verschiedener Kartenanbieter die von OSM-Mitgliedern zur Kartierung verwendet werden: Maxar [Maxar, 2020] in Abbildung 6.1a, Bing [Microsoft, 2020] in Abbildung 6.1b, Bavaria-80 des bayrischen Vermessungsamt [Bayrisches Vermessungsamt, 2020] in Abbildung 6.1c und ESRI [ESRI, 2020] in Abbildung 6.1d. Gezeigt wird ein Ausschnitt der Teststrecke Flight auf dem Universitätsgelände der Universität der Bundeswehr München.

aus einem Orthofoto annotiert [Mooney und Corcoran, 2012]. Orthophotos sind Luftbildaufnahmen, deren Höhenverzerrung durch ein Digitales Geländemodel im Postprocessing herausgerechnet wurde.

Im Folgenden werden Orthofotos von verschiedenen Kartenanbietern der gleichen Szene verglichen. In Abbildung 6.1 sind die einzelnen Anbieter gegenüber gestellt, die normalerweise von OSM-Nutzern zur Kartierung verwendet werden. Um die Unterschiede und die Qualität der Orthofotos besser zu verdeutlichen, wurden global referenzierte Fahrspuren (weiße Linien) mittels Real Time Kinematic (RTK)-GNSS kartiert und im georeferenzierten Bild eingezeichnet.

Es ist deutlich zu erkennen, dass das Orthofotos des Kartenanbieters Maxar Abbildung 6.1a eine mehrere Meter große Verzerrung aufweist. In Abbildung 6.1b ist ein scharfes, jedoch stark veraltetes Luftbild des Kartenanbieters Bing zu sehen. Vertikale Strukturen, wie Bäume und Gebäude, lassen sich sehr gut erkennen, jedoch entspricht das Bild nicht dem aktuellen Straßenverlauf und so fehlen komplette Straßenzüge.

Im Vergleich zu den vorherigen genannten Orthofotos liefert das bayrische Landesamt für Digitalisierung, Breitband und Vermessung ein exaktes und aktuelles Luftbild, siehe Abbildung 6.1c, jedoch bei deutlich geringerer Auflösung (in der kostenlosen Version).

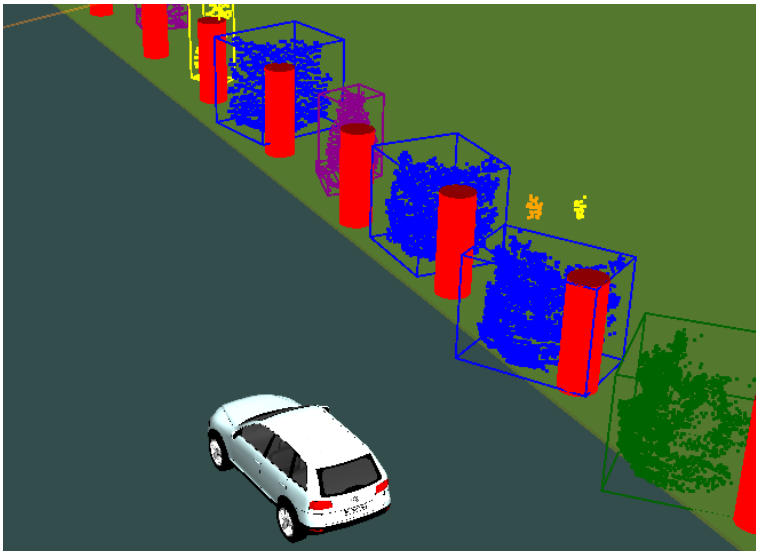


Abbildung 6.2:
Darstellung von Positionsfehler der OSM-Landmarken (Bäume). Das Fahrzeug ist mittels RTK-GNSS auf wenige Zentimeter genau in der Karte positioniert und die farblichen Punktwolken mit der dazugehörigen Bounding-Box stellen die mittels LiDAR-Sensor wahrgenommenen Landmarken dar.

In Abbildung 6.2 ist die selbe Szene unter Berücksichtigung einer zentimetergenauen RTK-GNSS Referenzlokalisierung des Fahrzeugs dargestellt. Die roten Zylinder repräsentieren Bäume der Baumreihe, welche rechts im Luftbild Abbildung 6.1b zu sehen ist. Die Bäume wurden auf Basis des Luftbildes händisch kartiert und global referenziert. Es ist ein deutlicher Positionsversatz zwischen Kartenlandmarken (rote Zylinder) und wahrgenommenen Objekten (farbige Bounding-Boxen) zu erkennen.

OSM ist der einzige Anbieter von frei zugänglichem Kartenmaterial welches großflächig verfügbar ist und eine zugängliche Schnittstelle bereitstellt. Zusätzlich wurde die Verwendbarkeit von OSM zur Lokalisierung von autonomen Robotern und autonomen Fahrzeugen bereits in mehreren Arbeiten bewiesen [Floros et al., 2013, Ruchti et al., 2015, Suger und Burgard, 2017, Vysotska und Stachniss, 2016, Yan et al., 2019]. Darüber hinaus bietet OSM eine hohe Datenvielfalt an Objekttypen wie Bäume, Sträucher, Verkehrsschilder, Wege, Straßen und Gebäude, die zur Lokalisierung

verwendet werden können [Goodchild, 2007], auch wenn die Qualität im Hinblick auf die Positionsgenauigkeit und Korrektheit in der Beschriftung unbekannt ist und sehr schwankt. Aus den genannten Gründen wurde auch in dieser Arbeit OSM als Kartenanbieter ausgewählt.

6.2 Datenschichten

Das GIS kann aus beliebig vielen Datenschichten bestehen. Jede Datenschicht beinhaltet Informationen, die zur Lokalisierung verwendet werden können. Neben metrischen Daten sind Topologien hinterlegt, welche die Beziehungen zwischen den kartierten Objekten beschreiben.

Die Schichtenarchitektur ermöglicht die strikte Trennung von verschiedenen Datenquellen. So erzeugt das Führungsfahrzeug eine eigene Datenschicht bei der Kartierung mittels SLAM-Verfahren. Neben dem Pfad des Führungsfahrzeuges werden die Landmarken entlang des Pfades gespeichert. Dabei stehen dem Führungsfahrzeug, falls verfügbar, Karteninformationen von OSM zu Verfügung, die bei der Pfadplanung aber auch bei der Landmarken-basierten Lokalisierung berücksichtigt werden können. Die OSM-Daten sind ebenfalls in einer eigenen Schicht gespeichert. Zusätzlich erzeugt jeder Konvoiteilnehmer eine extra Datenschicht, die neben der gefahrenen Trajektorie Landmarken enthält, die noch nicht in der Karte der vorausfahrenden Führungsfahrzeuge enthalten sind.

6.2.1 SLAM-Karten

Um die zur Lokalisierung benötigten Daten zu speichern und abzurufen, werden die Positionen des Führungsfahrzeugs, die dazugehörigen Landmarken und die Beziehungen zueinander innerhalb einer Datenschicht des GIS abgespeichert.

Jede kartierte Landmarke und jede Fahrzeugposition erhält, wie es für in einer Datenbank gespeicherte Daten üblich ist, eine eindeutige Kennung, die ID. Die ID wird im Zuge eines neuen Tracks vom MTT-Verfahren für die Landmarken erzeugt. Die ID's der Fahrzeugposen werden iterativ für jede neuen Graph-Knoten erhöht.

Sofern die globale Position der erstellten Pfad-Karte durch die MCL ermittelt wurde, können die globalen Posen der Landmarken und die der Fahrzeugpositionen im UTM-Koordinatensystem unter Verwendung der PostGIS-Geometrietypen gespeichert werden. Ist dies nicht der Fall, erfolgt die Kartierung in einem inertialen Raum mit metrischen Koordinaten.

Um die Abfragen nach geografischen Daten in der Datenbank zu beschleunigen, wird jeweils eine räumliche Indexierung der Positionstabellen durchgeführt [Fröhlich, 2018].

Zusätzlich werden die Dimensionen der Landmarken (Länge, Breite und Höhe), die semantische Klasse, die Anzahl der Assoziationen und die Unsicherheit über die

Position und Dimension der Landmarken aber auch die der eigenen Positionsschätzung gespeichert.

Eine Übersicht über alle Parameter wird in folgender Liste gegeben:

- Pfad des Führungsfahrzeugs

  SE2 Pose inkl. Unsicherheiten

  Zeitstempel, ID

- Landmarken

  2D-Position inkl. Unsicherheiten

  Zeitstempel, ID

  Dimension (Länge, Breite, Höhe) inkl. Unsicherheiten

  Existenzwahrscheinlichkeit

  Gesamtanzahl von Detektionen

  Landmarkenklasse

- Kanten, Beziehung zwischen den Knoten

  ID der Landmarke

  ID der Position

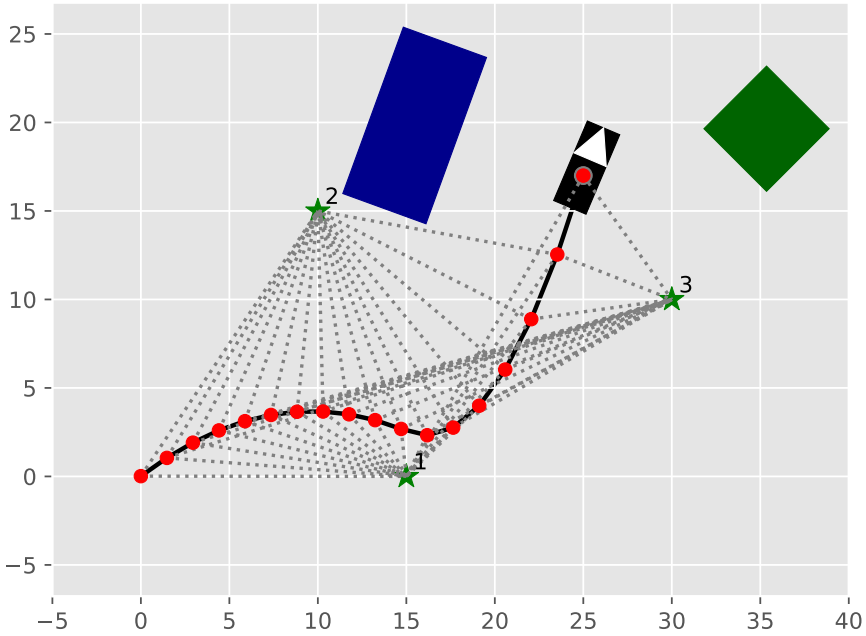


Abbildung 6.3:
Die Abbildung zeigt den erstellen Pfad-Graphen auf Basis des SLAM-Verfahrens. Das Fahrzeug ist als schwarze Bounding-Box mit Orientierung, Fahrzeugposen als rote Punkte und Landmarken als grüne Sterne gekennzeichnet. Die einzelnen Pfad-Knoten der gefahrenen Trajektorie sind miteinander verbunden und haben eine vorgegebene Richtung (schwarze Linie). Die grauen gestrichelten Linien verknüpfen die kartierten Landmarken und die Pfad-Knoten, an denen sie sichtbar waren. Die blaue und grüne Bounding-Box repräsentieren vertikale Objekte wie Gebäude, welche die direkte Sicht zu den Landmarken verhindern.

6.2.1.1 Erstellung der Pfad-Karte

Die Pfad-Karte ist eine virtuelle topologisch-metrische Repräsentation einer kartierten Wegstrecke. Die Pfad-Karte $\boldsymbol{M}$ entspricht dem SLAM-Graphen, der in Sektion 5.6 vorgestellt wurde. Fahrzeugposen und Landmarken werden ebenfalls als Knoten repräsentiert und mit einer N-N Beziehung abgebildet. Das bedeutet, dass bei der Kartierung neben der Fahrzeugposition, die dazugehörigen sichtbaren Landmarken als Verknüpfung abgespeichert werden. Umgekehrt kann eine Landmarke auch mit mehreren Fahrzeugpositionen verbunden sein, was bedeutet, dass eine Landmarke von verschiedenen Positionen aus sichtbar ist. In Abbildung 6.3 sind die Datenbankobjekte und die Relationen am Beispiel des Pfad-Graphen dargestellt.

Die Pfad-Karte wird vom Führungsfahrzeug erstellt, lokal im GIS abgespeichert und kontinuierlich vom GIS des Führungsfahrzeugs an das Folgefahrzeug übertragen. Zur Kommunikation können verschiedene Wege ausgewählt werden, neben schmalbandigem Funk und Mobilfunk ist auch eine Übermittlung über Wireless Local Area Network (WLAN) möglich. Die Übertragung der Fahrzeugposen und

der dazugehörigen Landmarken erfolgt in Abhängigkeit der gefahrenen Wegstrecke und dem aktuellen Lenkwinkel. Dies ist nötig, damit die Pfad-Knoten entlang des Pfades äquidistant verteilt sind und nicht in Abhängigkeit der Zykluszeit des SLAM-Verfahrens und der Fahrzeuggeschwindigkeit übertragen werden, siehe Abbildung 6.3. Zusätzlich führt dies zu einer Verringerung der Datenmenge, insbesondere auf geraden Strecken, bei geringen Geschwindigkeiten und kleinen Änderungen des Lenkwinkels. Die Auswirkungen der adaptiven Übertragung sind exemplarisch in Abbildung 6.4 dargestellt.

6.2.1.2 Übertragung der Daten vom Führungsfahrzeug an das Folgefahrzeug

Zur Übertragung der Pfad-Karte vom Führungsfahrzeug an das Folgefahrzeug kann zwischen zwei Varianten gewählt werden.

Rohdatenübertragung mittels UDP: Beim Versenden der Pfad-Karte werden die einzelnen Landmarken und Posen als eigene Datenstruktur innerhalb des UDP-Payloads versendet. Der Overhead für die Datenübertragung ist gering und die benötigte Bandbreite auf ein Minimum reduziert. Durch einen Prüfsummencheck kann zusätzlich die Robustheit und Sicherheit der Datenübertragung erhöht werden. Einen Nachweis für die Vollständigkeit der übertragenen Daten ist beim UDP nicht gegeben, was bei Verbindungsunterbrechungen zu einer unvollständigen Pfad-Karte führen kann. Das Führungsfahrzeug und das Folgefahrzeug müssen somit zusätzlich den aktuellen Stand der übermittelten Objekte austauschen und bei Bedarf fehlende Pakete noch einmal übermitteln.

Bei der **Replikations-basierten Übertragung** werden mittels logischer Replikation die Datenbanktabellen des Folgefahrzeugs mit denen des Führungsfahrzeuges synchronisiert. Bei der Synchronisierung werden nur die inkrementellen Änderungen übertragen, was die Payload auf ein Minimum reduziert. Die Übertragung erfolgt mithilfe einer direkten Transmission Control Protocol (TCP)-Verbindung zwischen den Fahrzeugen. Die Datenrate ist im Vergleich zu dem UDP-Verfahren höher. Die Funktionalität wird direkt von PostgreSQL bereitgestellt und ist auf die sichere Übertragung ausgelegt.

Um den Payload der übertragenen Daten zusätzlich zu reduzieren, wurden verschiedene Schritte unternommen:

- Reduktion der Datentypen auf die minimale mögliche Anzahl an numerischen Stellen unter Einhaltung der geforderten Genauigkeit ($1\,\mathrm{cm}$).
- Übertragung der Kovarianzmatrizen als Dreiecksmatrizen.
- Verwendung der Objekt-ID's als Primärschlüssel.

Abbildung 6.4:
Darstellung des übertragenen und interpolierten Pfad-Graphen. Zwischen den übertragenen Pfad-Knoten (hellblaue Kreise) werden die Posen linear interpoliert (Dunkelblau) und die Verknüpfung mit den Landmarken (Rot) auf Basis der echten Kanten (Hellgrün) des nächstgelegenen ursprünglichen Pfad-Knoten erstellt. Die dunkelgrünen Verknüpfungen stellen dabei die interpolierten Kanten dar.

6.2.1.3 Rekonstruktion der übermittelten Pfad-Karte

Je weiter der Abstand zwischen Pfad-Knoten ist, umso schwieriger wird es für die MCL, Mehrdeutigkeiten aufzulösen und zu einer Position zu konvergieren. Die Entfernung der Pfad-Knoten wirkt sich somit direkt proportional auf die Lokalisierungsgenauigkeit aus. Sind die Pfad-Knoten nicht gleichverteilt entlang der Pfad-Karte, kommt es zusätzlich zu einem Ungleichgewicht bei der Approximation der multimodalen Verteilung durch die MCL. Partikel in Bereichen mit einer höheren Anzahl an Pfad-Knoten werden proportional übergewichtet.

Um diesem Problem entgegenzuwirken, erfolgt die Rekonstruktion der Pfad-Karte durch eine lineare Interpolation und äquidistante Abtastung zwischen den übermittelten Pfad-Knoten und entlang der kompletten Pfad-Karte.

Den interpolierten Knoten werden jeweils die assoziierten Landmarken des nächstgelegenen originalen Pfad-Knotens zugeordnet, unter der Annahme, dass die Posen-Landmarkenbeziehungen lokal konstant sind.

Die Interpolation und die übertragene Pfad-Karte ist in Abbildung 6.4 dargestellt.

6.2.2 OpenStreetMap

Der OSM-Datenlayer besteht aus Landmarken, die zur Lokalisierung verwendet werden können. Die Kartenobjekte werden dabei als 2D-Punkt, 2D-Linie oder als

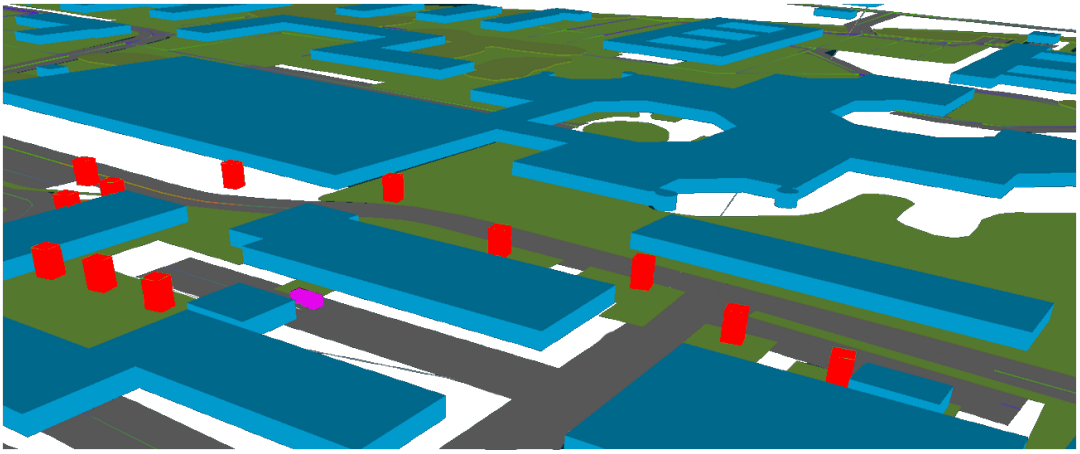


Abbildung 6.5:
Darstellung der Kartendaten von OSM. Straßen und Wege werden als graue Polygone visualisiert. Gebäude werden in Blau, Bäume in Rot, Grasflächen in Grün, hohe Vegetation in Dunkelgrün sowie die Bounding-Box des Fahrzeug in Magenta dargestellt.

2D-Polygonzug repräsentiert. In Abbildung 6.5 ist eine räumliche Darstellung von OSM-Daten gezeigt. Die Modellierung der Kartenobjekte in 3D anhand des Vorwissens (semantische Beschriftung) ermöglicht zusätzlich im SLAM-Verfahren eine verbesserte Objekterkennung und Datenassoziation mithilfe einer virtuellen Verdeckungsberechnung.

6.2.2.1 Datenrepräsentation

OSM besteht aus zwei grundlegenden Elementtypen, den Nodes sowie den Ways, die sämtliche Karteninformationen beschreiben. Jedes Element besitzt darüber hinaus eine Typenkennzeichnung (Tags) mit weiteren Meta-Informationen, wie die Objektklasse (Baum, Busch, Wegpunkt, Häuserecke, Schild, etc.) aber auch Informationen über die Anzahl der Stockwerk von Gebäuden oder die Öffnungszeiten von Geschäften.

Der Grundtyp ist der Node, der einen Punkt auf der Erdoberfläche in World Geodetic System 1984 (WGS 84) Koordinaten beschreibt. Die Positionsgenauigkeit der Koordinaten ist auf sieben Nachkommastellen limitiert, der numerische Fehler liegt somit bei maximal $\pm 0.01\,\mathrm{m}$.

Ein Way ist hingegen eine Folge von Nodes in einer bestimmten Reihenfolge, die Objekttypen wie Straßen, Wege, Gebäude oder Uferlinien repräsentieren können, siehe Abbildung 6.6.

Zusätzlich ermöglichen die Relations die Beziehung zwischen einem oder mehreren Grundelementen zu beschreiben, also beispielsweise die Verknüpfung von zwei Straßen und Flusslinien.

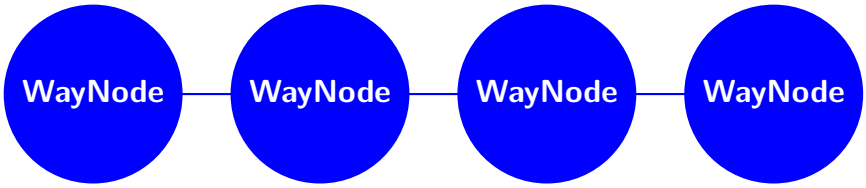


Abbildung 6.6:
Darstellung der OSM-Repräsentation für den Elementtyp Way. Ein Way besteht aus mehreren Nodes, die zum Beispiel eine Straße, Verkehrswege, Gebäude oder Uferlinien bilden.

Mit den zwei Grundelementen sowie den Relations ist es somit möglich, sämtliche Kartenobjekte darzustellen. Darüber hinaus beinhalten die Metadaten zum Beispiel genaue Typenbezeichnung eines Verkehrsschildes, Adressen von Gebäuden, Oberflächenbeschaffenheit von Straßen bis hin zu der Anzahl von Stockwerken pro Gebäude. Diese Metadaten liefern wichtige Zusatzinformationen, die bei der Objekterkennung und Objektklassifizierung sowie bei der Dateninterpretation unterstützen [Huang et al., 2018, Mooney und Corcoran, 2012, Suger und Burgard, 2017].

6.2.2.2 Datenimport

Um die OSM-Daten in das GIS zu integrieren, wird die Schnittstellenanwendung osm2pgsql[1] verwendet. Diese Anwendung überträgt die OSM-Rohdaten in die PostgreSQL Datenbank. Das Datenbankschema kann dabei individuell erweitert werden.

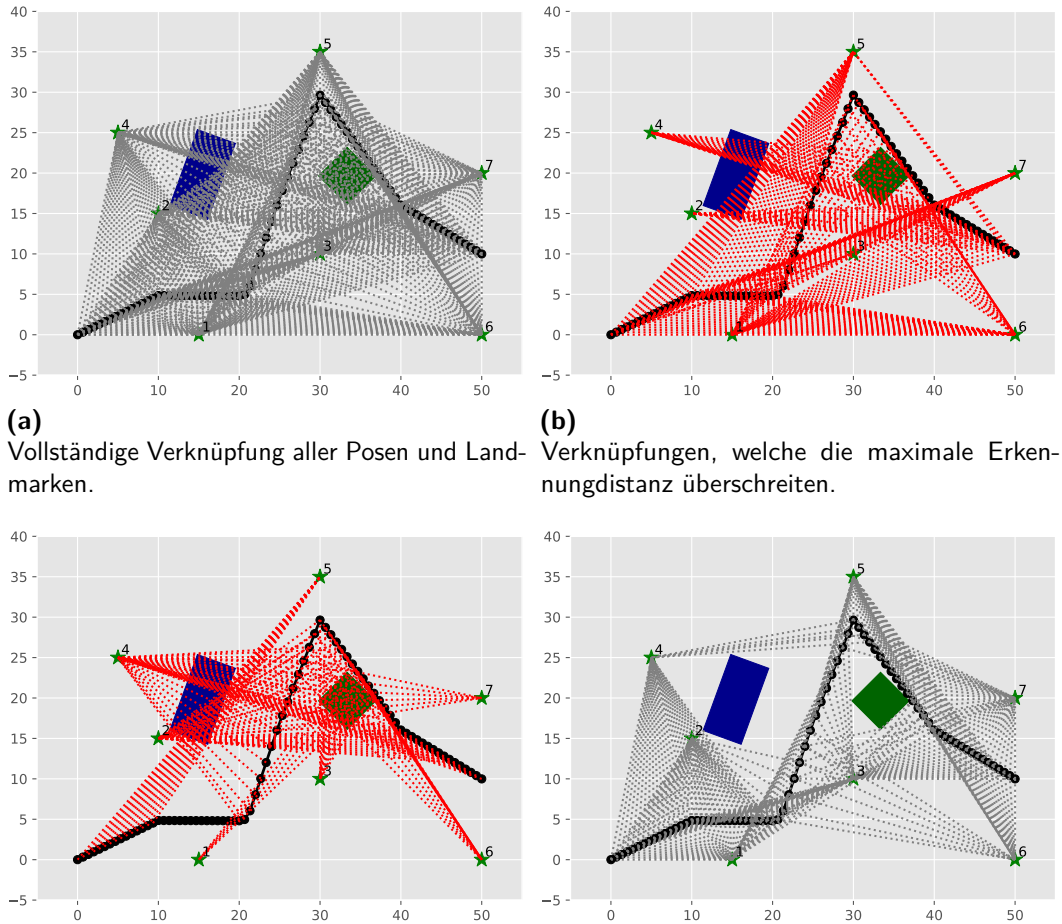
Um Karten-Landmarken aus der Datenbank abzufragen, wurde eine C++ Bibliothek entwickelt, die unter Verwendung von SQL-Befehlen Kartenobjekte um eine gewählte Position und einen gegeben Radius lädt und entsprechend dem Objekttyp eine eigene Objektklasse erstellt. Um die Datenbankabfrage zu beschleunigen, wird zusätzlich ein Suchbaum über die 2D-Positionen erstellt und sogenannte Indextabellen erzeugt, die den Zugriff beschleunigen.

6.2.2.3 Erstellung des virtuellen Pfad-Graphen

Im Folgenden wird die Erstellung des virtuellen Pfad-Graphen beschrieben. Die Basis liefert eine globale Route, die vom Führungsfahrzeug oder einem Planungs- und Verhaltensmodul vorgegeben ist und auf Basis des OSM-Straßennetzwerks erstellt wurde. Die globale Route bestimmt das übergeordnete Ziel und wird durch Wegpunkte beschrieben.

In einem ersten Schritt werden der virtuelle Pfad-Graph und die Pfad-Knoten entlang der Route äquidistant verteilt. Im Vergleich zum selbst kartierten Pfad-Graphen (siehe Unterabschnitt 6.2.1.1) ist vorab nicht bekannt, an welchen Orten welche Kartenlandmarken sichtbar sind. Aus diesem Grund wird unter Berücksichtigung der

[1] https://wiki.openstreetmap.org/wiki/Osm2pgsql



(a)
Vollständige Verknüpfung aller Posen und Landmarken.

(b)
Verknüpfungen, welche die maximale Erkennungdistanz überschreiten.

(c)
Verknüpfungen, die durch Objekte verdeckt werden.

(d)
Finaler Pfad-Graph unter Berücksichtigung der Distanz und Sichtbarkeit.

Abbildung 6.7:
Die Abbildungen zeigen den Prozess zum Erstellen des virtuellen Pfad-Graphen unter Berücksichtigung der Kartendaten aus OSM. Abbildung 6.7a zeigt die vollständige Verknüpfung aller OSM-Landmarken und den gesampelten Pfad-Knoten (schwarze Punkte) entlang der globale Route (schwarze Linie). Unter Berücksichtigung der maximalen Distanz unter der ein Objekt erkannt werden kann, siehe Abbildung 6.7b, werden Verknüpfungen entfernt. Zusätzlich wird eine Verdeckungsrechnung durchgeführt, die Wissen über Kartenobjekte, wie beispielsweise Gebäude (blaue Bounding-Box) oder hohe Vegetation (grüne Bounding-Box) dazu benutzt, die entsprechenden Verknüpfungen zu entfernen, siehe Abbildung 6.7c. Abbildung 6.7d zeigt den finalen Pfad-Graphen mit allen validen Verknüpfungen (grau gestrichelte Linien).

Kartendaten und mithilfe einer Verdeckungsrechnung die Sichtbarkeit von Objekten geschätzt. Zusätzlich wird die maximale Erkennungsdistanz der Objekterkennung verwendet, um zu weit entferne Landmarken auszuschließen.

Zur Feststellung, ob eine Verdeckung vorliegt, werden zuerst alle Verknüpfungen zwischen Landmarken und Posen erstellt, siehe Abbildung 6.7a. Im nächsten Schritt wird

für jede Verknüpfungslinie untersucht, ob eine Überschneidung mit Kartenobjekten vorliegt, die die Sichtlinie verdecken können [Antonio, 1992, Schneider und Eberly, 2003]. Kommt das Verfahren dabei zum Ergebnis, dass die Landmarke theoretisch sichtbar ist, so werden die entsprechenden virtuellen Messungen (Landmarkenverknüpfungen) hinzufügt. Das Ergebnis des Pfad-Graphen ist in Abbildung 6.7d dargestellt. Die grauen gestrichelten Linien zwischen den Wegpunkten und den Landmarken repräsentieren virtuelle Messungen und beschreiben, an welcher Position welche Landmarke sichtbar sein kann.

Ein Beispiel ist in Abbildung 6.7a visualisiert, mit Graph-Knoten als schwarze Punkte entlang des Pfades, Landmarken als grüne Sterne mit einer eindeutigen ID sowie die Verknüpfungen aller Landmarken und Graph-Knoten als graue gestrichelte Linien. Exemplarisch sind ebenfalls vertikale Hindernisse, wie ein Gebäude (Blau) und hohe Vegetation (Dunkelgrün), mit einer Bounding-Box dargestellt. Abbildungen 6.7b und 6.7c zeigen jeweils die Verknüpfungen, die in Abhängigkeit der maximale Distanz und der Sichtbarkeit entfernt wurden.

6.2.3 Karte aus Luftbildern

Als weitere Datenquelle wurde untersucht, inwieweit Luftbilder zur Erstellung von Landmarkenkarten verwendet werden können, um sich mit einem autonomen Landfahrzeug darin zu lokalisieren. Denn nicht immer ist Kartenmaterial für unstrukturierte Bereiche verfügbar und oftmals ist es nicht möglich, den Bereich vorab zu erkunden.

Die Vermessung mit einem Unmaned Aerial Vehicle (UAV) ist ein kostengünstiges und flexibles Datenerfassungsverfahren. UAVs erlauben die automatische und effiziente Erfassung und werden heutzutage oftmals für die Aufgabe der großflächigen Kartenerstellung verwendet [Boerner et al., 2019, Mancini et al., 2013, Nex und Remondino, 2014]. Die schnelle Erfassung mit einem UAV ist insbesondere dann interessant, wenn Gebiete nicht einfach zu befahren sind oder wenn sich seit der letzten Vermessung gravierende Veränderungen ergeben haben.

Durch die geringe Flughöhe von UAV und den eingesetzten hochauflösenden Kameras können Objekte sehr genau erfasst und modelliert werden. Die hochauflösenden Bilder bieten somit eine gute Datengrundlage für die Kartenerstellung. Zusätzlich können Objekte und Flächen klassifiziert und ihnen semantische Beschreibungen zugewiesen werden.

Das Ziel des Ansatzes ist, aus einer Menge von Luftbildern georeferenzierte zweidimensionale und dreidimensionale Objekte sowie Flächen zu bestimmen, die im Lokalisierungsprozess von Landfahrzeugen berücksichtigt werden können.

Die folgende Liste fasst die Ziele des Projektes zusammen:

1. Datenerfassung mittels UAV und Kartenerstellung für autonome Fahrzeuge.
2. Ergebnis ist eine vollständige 3D-Szenenbeschreibung mit genauer Geometrie- und Farbinformation aus hochauflösenden Bildern.

3. Kartendaten liefern wichtige Informationen für die Fahrzeuglokalisierung und Fahrzeugnavigation mittels Onboard-Sensorik.
4. Abstrakte und effiziente Datenrepräsentation in einem einheitlichen Kartenformat (angelehnt an OSM).

Das Verfahren zur Erstellung der Karte wurde von den Mitarbeitern des Instituts für Angewandte Informatik der Universität der Bundeswehr München von Univ.-Prof. Dr.-Ing. Helmut Mayer in einem gemeinsamen Projekt entwickelt und umgesetzt. Im folgenden werden die verschiedenen Schritte zur Erstellung der Karte beschrieben.

6.2.3.1 3D-Szenenrekonstruktion

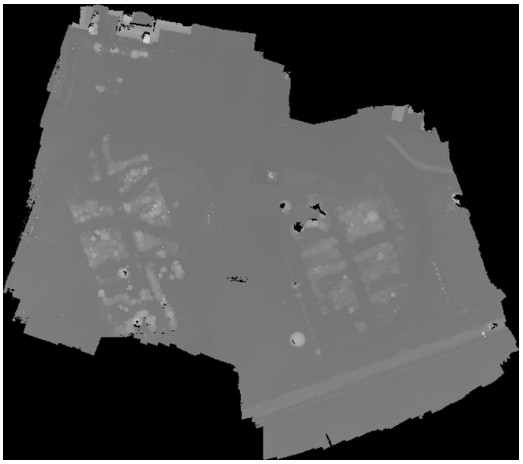
Um eine Punktwolke aus Einzelbildern zu erzeugen, wurde der Ansatz von Mayer et al. [2012] verwendet. Dieser Ansatz kombiniert Structure from Motion (SfM) mit Semiglobal Matching (SGM), um übereinstimmende Merkmale in mehreren Bildpaaren mit großem Basisabstand zu identifizieren und eine dichte Punktwolke zu erzeugen. SfM basiert auf den gleichen Prinzipien wie die stereoskopische Photogrammetrie, siehe Hirschmüller [2011]. In der Stereophogrammetrie wird die Triangulation verwendet, um die relativen 3-D-Positionen von Objekten aus Stereobildpaaren zu berechnen. Um eine 3D-Rekonstruktion zu erstellen, werden viele Bilder eines Bereiches oder eines Objektes mit einem hohen Grad an Überlappung benötigt, die aus verschiedenen Winkeln aufgenommen wurden. Dabei werden die gleichen Merkmale von Bild zu Bild verfolgt und zur Erstellung von Schätzungen der Kamerapositionen und Kameraausrichtungen sowie den 3D-Koordinaten der Merkmale verwendet. Die Summe der 3D-Koordinaten bildet dann die Punktwolke [Hirschmüller, 2011, Hoegner et al., 2016].

6.2.3.2 Szeneninterpretation und Objekterkennung

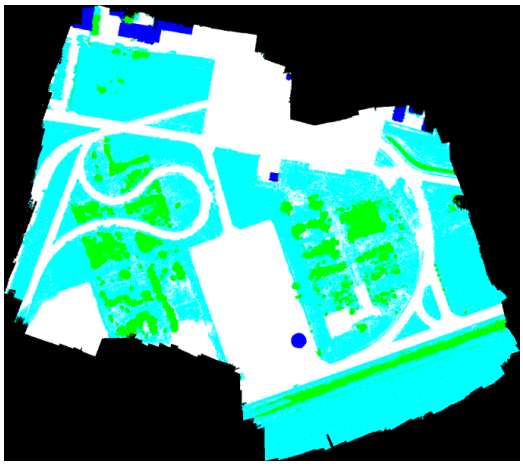
Um die Szene zu klassifizieren wird in einem ersten Schritt die Anzahl der 3D-Punkte reduziert und in ein Paar aus RGB- und Tiefenbildern umgewandelt, siehe Abbildungen 6.8a und 6.8b. Mit einer Erweiterung des Ansatzes von Huang und Mayer [2015] wird schließlich die Szene in Abhängigkeit der Höhe und Klasse als befahrbar und nicht befahrbar klassifiziert. In Abbildung 6.8c ist das Ergebnis der semantischen Klassifizierung nach dem Verfahren von Huang et al. [2018] gezeigt. Der Fokus liegt hier besonders auf vertikalen Objekten bzw. Landmarken, die zur Lokalisierung verwendet werden können.

Die statistische Objektrekonstruktion mittels generativer Primitive von Huang et al. [2011, 2013] dient der Erkennung von Gebäuden und Leitpfosten, aus denen die abstrakten Landmarken abgeleitet werden können (Abbildung 6.9). Die Detektion wird dabei auf der ursprünglichen dichten Punktwolke durchgeführt, da diese ausreichend Datenpunkte in der Vertikalen von Objekten enthält. Geometrien, die nicht für die Lokalisierung eines Fahrzeugs wichtig sind, werden jedoch bei der Extraktion ignoriert. Dies ist z. B. für die in Abbildung 6.9a gezeigte Dachform der Fall.

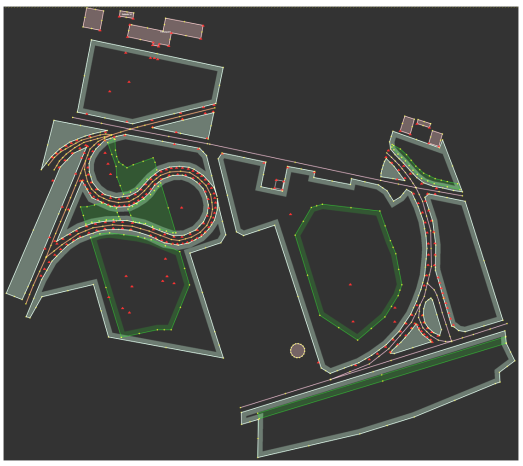
(a) Farbige Punktwolke



(b) Tiefenbild



(c) Klassifiziertes Orthobild

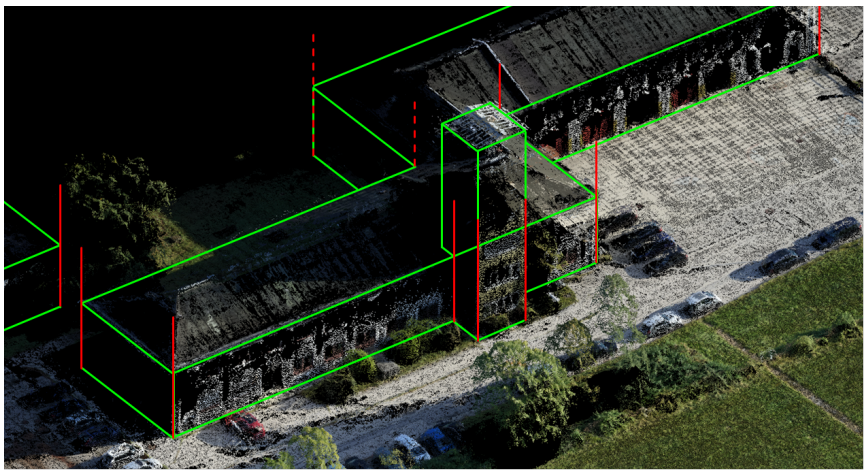


(d) Abstrahierte Karte

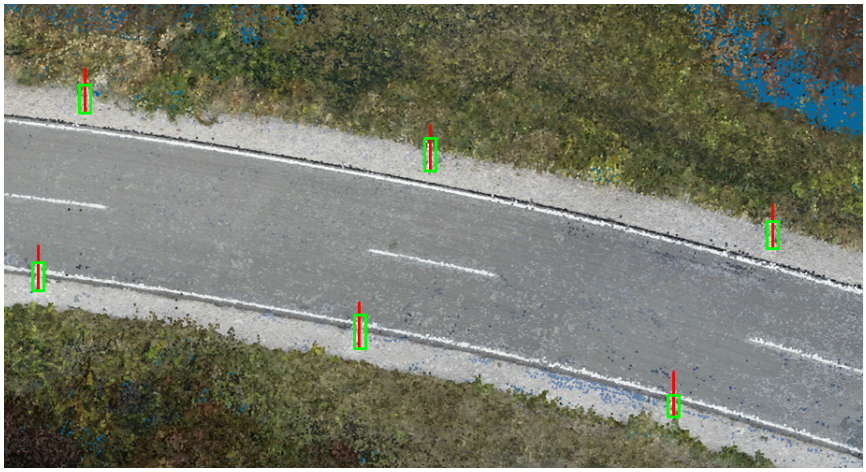
Abbildung 6.8:
Darstellung der einzelnen Verarbeitungsschritte zur Erstellung der Landmarken-Karte aus 756 Bildern, die mit einem UAV aufgenommen wurden. Die kartierte Fläche ist ein Teil der Universität der Bundeswehr München (UniBw M)-Teststrecke. Abbildungen 6.8a und 6.8b zeigen ein farbiges Orthofoto und Tiefenbild, welche mit Hilfe der dreidimensionalen Punktwolke erstellt wurden. Abbildung 6.8c zeigt die klassifizierte Punktwolke: geteerter Boden (Weiß), niedrige Vegetation/Grasland (Cyan), hohe Vegetation/Strauchwerk (Grün) und Gebäude (Blau). Abbildung 6.8d stellt die aus den Eingangsdaten abstrahierte Karte dar, welche zur Lokalisierung eines Bodenfahrzeugs verwendet werden kann.

6.2.3.3 Kartenerstellung

Zur Kartenerstellung werden die zur Lokalisierung verwendbaren Landmarken aus dem klassifizierten Orthobild abgeleitet und extrahiert. Vegetationsflächen werden mit Hilfe eines umhüllenden Polygons beschrieben. Vertikale Landmarken, die sich gut von der Umgebung abgrenzen lassen, werden als Punktmessung mit Höheninformation exportiert. Alle exportierten Daten sind global referenziert und besitzen eine



(a)



(b)

Abbildung 6.9:
Das Ergebnis der Kartenerstellung aus Luftbildern eines UAV. Dreidimensionale Modelle von Gebäuden (a) und Leitpfosten (b) sowie die abstrahierten Landmarken sind als grüne und rote Linie dargestellt.

semantische Beschreibung. Eine Visualisierung der erstellten Kartenobjekte sowie der einzelnen Arbeitsschritte wurde bereits in Abbildung 6.8 und Abbildung 6.8d gezeigt.

Der Export der Landmarken erfolgt im Extensible Markup Language (XML) Datenformat, was ebenfalls von OSM verwendet wird. Die Schnittstelle zum Laden der Kartenobjekte bleibt somit zu der in Unterabschnitt 6.2.2.2 beschriebenen Methode gleich. Die Karte kann somit in das vorhandene GIS integriert, verwendet und mit den Daten der OSM verglichen werden.

Bei der Kartenerstellung aus Luftbildern wurden sämtliche verfügbaren Merkmale kartiert, obgleich im vorgestellten Ansatz nur vertikale Landmarken berücksichtigt werden. Zusammengefasst beinhaltet die neu erstellte Karte aus Luftbildern folgende Merkmale:

- Straße (Asphalt) als Liniensegmente, Gebäude, mittlere Vegetation (Büsche), niedrige Vegetation (Grasland) als Polygone mit semantischen Beschriftungen und Bodenbeschaffenheit.
- Abstrakte Landmarken (Bäume, Gebäudeecken und Pfosten) als 3D-Punkte.
- Befahrbarkeit, in Abhängigkeit von verschiedenen Fahrzeugtypen als Polygone.
- Steigungsinformationen mit Hilfe von Höhenlinien.

7 Auswertung

Inhalt

7.1 Einführung

Das Ziel der Auswertung ist, die Qualität und Leistung des Gesamtsystem in verschiedenen Einsatzgebieten zu evaluieren. Für die Erprobung des Gesamtsystems wurden mehrere und verschiedene manuelle und autonome Testfahrten im aufgesplitteten Konvoi ausgewertet. Die ausgewählten Testszenarien besitzen unterschiedliche Bodenbeschaffenheiten und weisen einen starken Unterschied in der Landmarkenanzahl und Landmarkenqualität auf:

- Standortübungsplatz München in Garching-Hochbrück (unstrukturiertes Gelände, größere Grünflächen mit zum Teil wenigen vertikalen Objekten, Waldwegen, Schotterwegen),
- Aying (unstrukturiertes Gelände, dicht bewachsen, Waldwege),
- UniBw M (strukturiertes Gelände, Vorstadt-ähnliches Gebiet, geteerte Straßen, gepflasterte Wege, keine Straßenmarkierungen).

Zur Auswertung ist die initiale Position der Fahrzeuge in Real Time Kinematic (RTK)-Qualität gegeben, um die geschätzten Fahrzeugposen mit der Referenzlösung vergleichen zu können. Die Funktionsparameter der verschiedenen Komponenten des SLAM-Frameworks wurden für alle Testszenarien einheitlich festgelegt und werden zwischen den Szenarien und während der Laufzeit für alle Fahrzeuge nicht geändert. Auswertungen von Vorarbeiten, die in dieser Arbeit verwendet werden, sind in den Veröffentlichungen [Burger et al., 2018, 2019a,b, Burger und Wuensche, 2018, Naujoks et al., 2019a,c] zu finden.

Die Gesamtlaufzeit des Systems lag nach über 30 Messfahrten durchschnittlich bei 85 ms. Die Prozesse des Frontends und Backends werden jeweils in einem eigenen Prozess ausgeführt. Die Laufzeit der MCL-Lokalisierung und der LiDAR-basierten

Sensor-Landmarkenerkennung lag bei ≈ 65 ms und somit deutlich unterhalb der Laufzeit einer Rotation von 100 ms des mit 10 Hz rotierendem LiDAR-Sensors. Im Zeitschritt der Kartenerstellung stieg die Gesamtlaufzeit auf maximal 150 ms an. Diese ist abhängig von der gewählten Größe des Betrachtungszeitraums und basierend auf der Anzahl vergangener Fahrzeugposen sowie der Sensor-Landmarkenanzahl. Da die Erstellung der vollständigen Karte durchschnittlich nur jede Sekunde erfolgt und in einem eigenen Prozess ausgeführt wird, hat dies keinen nennenswerten Einfluss auf das Gesamtsystem.

7.2 Metriken

Die Auswertung der bestimmten Fahrzeugtrajektorie erfolgt durch Metriken. Der mittlere absolute Fehler (engl. Mean Absolute Error (MAE)) und der mittlere quadratische Fehler (engl. Root Mean Squared Error (RMSE)) sind zwei Metriken, die im Allgemeinen zur Evaluation von Lokalisierungsverfahren verwendet werden. Sowohl der MAE als auch der RMSE drücken den durchschnittlichen Modellvorhersagefehler in Einheiten der interessierenden Variablen aus. Beide Metriken können von null bis unendlich reichen und sind gleichgültig gegenüber der Richtung der Fehler. Das Ziel des hier vorgestellten Verfahrens ist, die Metriken (negativ orientierte Scores) zu minimieren. Denn je kleiner die Metrik, umso höher die Güte des Modells bzw. Verfahrens.

Der RMSE ist eine Maßzahl zur Beurteilung der Prognosegüte. Das Fehlermaß gibt an, wie gut ein Modell an die Referenzdaten angepasst ist und beschreibt in dieser Arbeit, wie stark die Positionsschätzung von den tatsächlichen Referenzpositionen im Durchschnitt abweicht. Berechnet wird der RMSE aus der Quadratwurzel des durchschnittlichen Prognosefehlers mit n als die Anzahl der betrachteten Variablen:

$$\mathrm{RMSE} = \sqrt{\frac{\Sigma_{i=1}^{n}(\mathbf{x}_i - \hat{\mathbf{x}}_i)^2}{n}}. \tag{7.1}$$

Das Ziehen der Quadratwurzel aus den durchschnittlichen quadrierten Fehlern hat einige interessante Implikationen für den RMSE. Durch das Quadrieren der Fehler vor der Mittelwertbildung wird großen Fehlern ein relativ hohes Gewicht zugeschrieben. Das bedeutet, dass der RMSE ein wichtiger Indikator ist, wenn insbesondere große Fehler besonders unerwünscht sind.

Der MAE gibt die durchschnittliche Größe der Fehler an und berechnet sich durch den Durchschnitt der absoluten Differenz zwischen der tatsächlichen Positionsschätzung und der Referenzposition, bei der alle Abweichungen gleich gewichtet sind:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|\mathbf{x}_i - \hat{\mathbf{x}}_i\right|. \tag{7.2}$$

Der Unterschied zwischen RMSE und MAE ist am größten, wenn der gesamte Vorhersagefehler von einer einzigen Stichprobe stammt. Der quadrierte Fehler entspricht

für diese einzelne Stichprobe dann $\left(\frac{1}{n}\sum_{i=1}^{n}|\mathbf{x}_i - \hat{\mathbf{x}}_i|\right)^2 \cdot n$ und 0 für alle anderen Stichproben.

Durch das Ziehen der Quadratwurzel ergibt sich der RMSE zu $\frac{1}{n}\sum_{i=1}^{n}|\mathbf{x}_i - \hat{\mathbf{x}}_i| \cdot \sqrt{(n)}$. Das kann problematisch sein, wenn RMSE-Ergebnisse verglichen werden sollen, die für eine verschieden große Anzahl von Stichproben berechnet wurden. Dies ist besonders in der Praxis häufig der Fall, wenn unterschiedlich lange Trajektorien aus verschiedenen Messfahrten zur Auswertung herangezogen werden. Aus diesem Grund wurden zur Auswertung ähnlich lange Sequenzen aus den Testfahrten verwendet.

7.3 Auswertung Gesamtsystem

Die Auswertung des Gesamtsystems erfolgt auf Basis von mehreren Mess- und Testfahrten mit einer Gesamtlänge von rund 56 km. Als Referenzlösung wurde ein lose gekoppeltes RTK-GNSS-INS-System verwendet, welches mit einem Korrektursignal arbeitet und nach Datenblatt eine Genauigkeit von bis zu 2.5 cm ermöglicht.

Zur Evaluierung erstellt das Führungsfahrzeug mittels SLAM-Verfahren eine Pfad-Karte online und lokalisiert sich lokal in dieser Karte. Die Pfad-Karte wird hierzu, gegeben der Referenzlösung, im globalen UTM Kartenraum registriert und zyklisch per Mobilfunk oder WLAN an das Folgefahrzeug übermittelt.

Die geschätzten Trajektorien des Führungsfahrzeugs und die des Folgefahrzeug werden zur Evaluierung getrennt ausgewertet. Um den Einfluss der gegebenen Karte auf das Gesamtsystem zu minimieren, erfolgt die Kartenerstellung des Führungsfahrzeugs auf Basis des Graph-SLAM-Backend aus Sektion 5.4 und unter Berücksichtigung einer manuell annotierten Referenzkarte. Die Landmarkenpositionen der Referenzkarte wurde auf Basis von Luftbildern ermittelt und über mehrere Messfahrten validiert. Durch die Integration von Kartenwissen kann der Positions- und Winkeldrift über längere Strecken reduziert und zusätzlich die selbst erstellte Karte global registriert werden. Die erste Fahrzeugpose ist dabei durch das RTK-GNSS-INS-System gegeben und die Pfad-Karte im UTM Koordinatensystem registriert.

Die Trajektorie des Folgefahrzeugs basiert hingegen auf der Fusion aller in Sektion 5.4 vorgestellten Komponenten. Die Auswertung beginnt jedoch erst, sofern die MCL konvergiert ist, da erst in diesem Zeitschritt eine globale Positionsschätzung vorliegt, welche sich mit der Referenzlösung vergleichen lässt.

Der Einfluss der reduzierten Datenübertragung auf die Positionsschätzung wurde ebenfalls in mehreren Experimenten untersucht und wird im Folgenden beschrieben. Zusätzlich wurde in mehreren Testfahrten die MCL im Hinblick auf Konvergenzverhalten in den verschiedenen Szenarien getestet.

(a) **(b)**

(c) **(d)**

Abbildung 7.1:
Beispielaufnahme einer Testsequenz auf dem Standortübungsplatz München.

7.3.1 Standortübungsplatz München

7.3.1.1 Beschreibung

Der Standortübungsplatz München ist eine Grasheide mit größtenteils flach gewachsener Vegetation sowie Bäumen und Sträuchern und eignet sich, um das Gesamtsystem in einem unstrukturiertem Gebiet mit viel Vegetation und ohne horizontale, vom Menschen geschaffene Strukturen, zu testen. Das Wegnetz besteht aus schmalen Waldwegen, Forststraßen und Schotterwegen. Beispielaufnahmen der Onboard-Kamera, welche die Umgebung zeigen, sind in Abbildung 7.1 zu finden.

7.3.1.2 Ergebnisse

Die Ergebnisse der Auswertung sind auf Basis von 12 verschiedene Testsequenzen bestimmt worden. Die Testsequenzen tragen im Folgenden die fortlaufende Nummerierung von 1 bis 12 und wurden zu unterschiedlichen Jahreszeiten im Sommer und Herbst 2020 (Testsequenz 1-8), Februar 2019 (Testsequenz 8-9) und Sommer 2018 (Testsequenz 10-12) aufgenommen und erprobt.

Zur Evaluierung wurde für jede Testsequenz der RMSE und der MAE ermittelt. Der gemittelte RMSE und der gemittelte MAE über alle Testsequenzen bei einer durchschnittliche Streckenlänge von ca. $2\,\mathrm{km}$ ist in Tabelle 7.1 zu finden.

In allen Testsequenzen war das Folgefahrzeug in der Lage, sich auf der gesamten Strecke entlang der Pfad-Karte zu lokalisieren. Der Abstand zwischen den Konvoifahrzeugen betrug dabei bis zu $400\,\mathrm{m}$ und größtenteils ohne direkten Sichtkontakt bei einer durchschnittlichen Geschwindigkeit von $7.1\,\mathrm{m/s}$.

Tabelle 7.1: Mittelwert der Metriken über alle Testsequenzen.

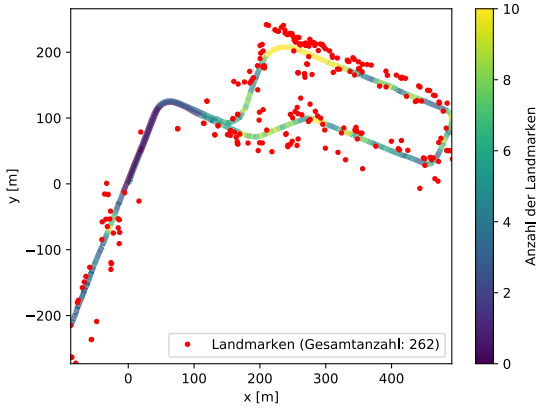
	RMSE	MAE
Positionsfehler MCL	$1.31\,\mathrm{m}$	$1.04\,\mathrm{m}$
Positionsfehler Graph-SLAM	$0.46\,\mathrm{m}$	$0.33\,\mathrm{m}$
Kursfehler MCL	$0.93°$	$0.68°$
Kursfehler Graph-SLAM	$0.24°$	$0.19°$

Die ermittelten Metriken der Sequenz 4 sind den Gesamtmetriken am ähnlichsten. Für die Testsequenz 4 werden somit exemplarisch die detaillierten Ergebnisse der Auswertung im Folgenden visualisiert und diskutiert. Das Luftbild in Abbildung 7.2 gibt einen Überblick über den gesamten Testbereich. Die gefahrene Wegstrecke der Testsequenz 4 ist in Blau sowie der Start, das Ziel und die Wegpunkte (1-5) sind eingezeichnet und beschriftet.

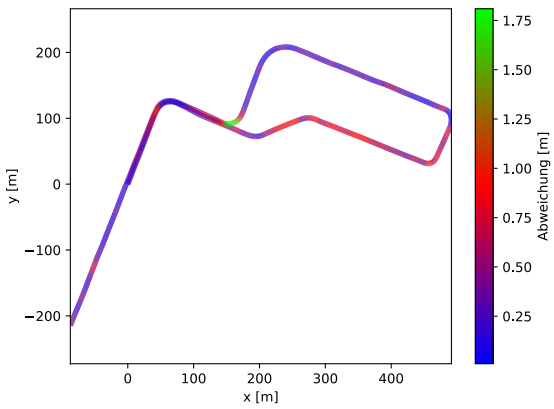
In Abbildung 7.3a ist die mittels Graph-SLAM erzeugte Pfad-Karte des Führungsfahrzeugs, die Anzahl von Karten-Landmarken pro Pfad-Knoten sowie alle Karten-Landmarken gezeigt. Die Farbkodierung von Dunkellila nach Hellgelb stellt die Anzahl der sichtbaren Karten-Landmarken pro Pfad-Knoten dar. In Abbildung 7.3b ist die Pfadabweichung des Folgefahrzeug zum Führungsfahrzeug dargestellt und gibt ein Maß, wie gut das Folgefahrzeug dem Führungsfahrzeug in der Spur gefolgt ist. Da die Lokalisierung entlang der gegebenen Pfad-Karte erfolgt, ist dies ebenfalls ein Faktor, der sich auf die Metriken auswirkt. Im Allgemeinen gilt, je weiter sich das

Abbildung 7.2:
Luftbild des Standortübungsplatzes München. Die in Blau visualisierte Linie stellt die gefahrene Strecke mit einer Gesamtlänge von $1850\,\mathrm{m}$ dar. Die Fahrt wurde an der Markierung **O** gestartet und es wurden die Wegpunkte WP 1 bis WP 5 nacheinander abgefahren, bis das Ziel **X** erreicht wurde.

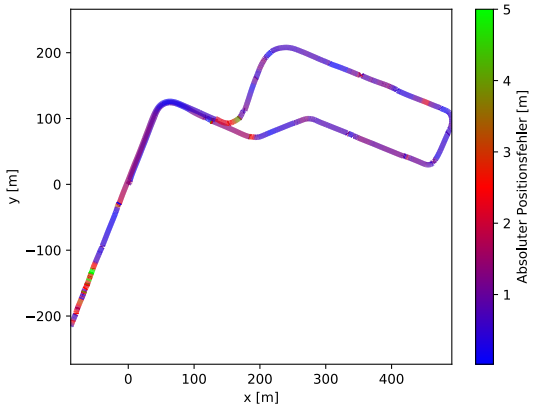
Folgefahrzeug vom Pfad des Führungsfahrzeugs fortbewegt, umso schwieriger ist es, eine korrekte Landmarkenassoziation durchzuführen.



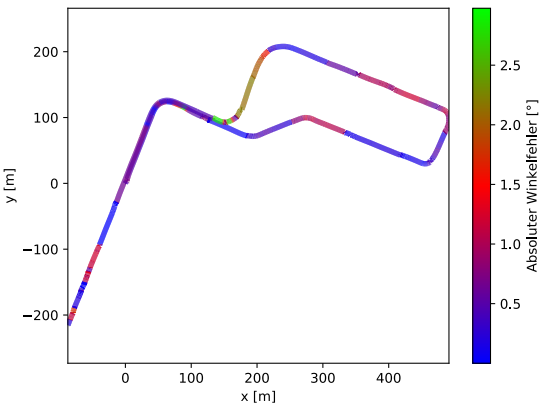
(a) Pfad-Karte des Führungsfahrzeugs



(b) Abweichung zum Führungsfahrzeugs



(c) Abs. Positionsfehler des Folgefahrzeug



(d) Abs. Kursfehler des Folgefahrzeug

Abbildung 7.3:
Testsequenz 4: Auswertung und Visualisierung der geschätzten Fahrzeugpositionen der MCL des Folgefahrzeugs auf Basis der Pfad-Karte des Führungsfahrzeugs. Der Maximalwert der Farbkodierung wurde für eine bessere Darstellung händisch gewählt und entspricht dabei nicht dem Maximalwert der Kennzahl.

In Abbildungen 7.3c und 7.3d ist der Positions- und Kursfehler des Folgefahrzeugs entlang der Pfad-Karte des Führungsfahrzeugs basierend auf der MCL dargestellt. Am Anfangsbereich der Testsequenz (an den Koordinaten $(x = 0, y = 0)$) sind sehr wenige, dafür jedoch eindeutige und stabile Landmarken sichtbar. Diese erlauben der MCL nach einer durchschnittlichen Wegstrecke von $14\,\mathrm{m}$ sich global in der gegebenen Karte zu lokalisieren.

Um Mehrdeutigkeiten zu lösen, die insbesondere bei der Initialisierung auftreten, wird das Resampling der MCL erst nach $10\,\mathrm{m}$ per Software aktiviert. Dies ermöglicht die Reduktion von fehlerhaften Initialisierungen, welche durch inkorrekte Sensor-Landmarken zu Karten-Landmarken Assoziationen und einer voreiligen Neugewichtung aller Partikel auftreten. Durch die Betrachtung einer Reihe von Landmarken

und Odometriemessungen entlang einer Trajektorie konnte dieser Effekt deutlich reduziert werden.

Nach Wegpunkt 1 und ca. an den Koordinaten $(x = 180, y = 100)$ fährt das Folgefahrzeug eine Linkskurve (nach Norden), wobei die Fahrspur nach Links deutlich geschnitten wird. Die laterale Abweichung zum Führungspfad beträgt dabei ca. $2.8\,\text{m}$.

Die Abweichung hat vorwiegend deshalb einen direkten Einfluss auf den Kurs- und Positionsfehler, weil die Abweichung direkt vor einer Kurve erfolgt und somit einen direkten longitudinalen Positionssprung durch ein Resampling der MCL impliziert.

Im Anschluss bleibt der Winkelfehler im Vergleich zum Positionsfehler deutlich länger bestehen, siehe Abbildung 7.3d. Dies liegt u. a. an der ungleichmäßigen Verteilung der Landmarken, die an dieser Stelle hauptsächlich vor dem Fahrzeug liegen. Die Zustandsunsicherheit in der Kursbestimmung ist somit deutlich erhöht und die MCL benötigt eine längere Strecke und eine erneute Winkeländerung (vor Wegpunkt 2 bei ca. $(x = 250, y = 200)$), bis der Winkelfehler korrigiert werden kann.

Eine weitere Auswertung hat ergeben, dass die Erkennung der Straßenlandmarken auf geraden Wegstücken die Lokalisierung deutlich stützen kann, insbesondere in Bereichen mit weniger als drei Landmarken oder in Bereichen, in denen die Landmarken sehr homogen verteilt sind. Darüber hinaus lässt sich erkennen, dass speziell bei Abzweigungen der Fehler durch die Straßenlandmarken erhöht wird. Dies liegt daran, dass die Straßenwahrnehmung abzweigende Wege nur unzureichend erkennt und das Messmodell diese nicht ausreichend gut modellieren kann.

Zusätzlich ist zu erwähnen, dass die Anzahl der Landmarken entlang der gesamten Pfad-Karte sehr unterschiedlich verteilt ist. Beispielsweise sind im Norden der Karte deutlich mehr Karten-Landmarken zu finden, da sich hier angrenzend ein Waldstück befindet.

Gegen das Ende der Wegstrecke (vor Ziel bei ca. $(x = -50, y = -130)$) kommt es erneut zu größeren Abweichungen. An dieser Stelle sind nur Karten-Landmarken und Sensor-Landmarken nordwestlich der Trajektorie zu finden. Die Geometrie der Sensor-Landmarken ist sehr ähnlich und die Distanz zwischen den Landmarken sehr gering. Hierbei kommt es zu Assoziationsfehlern, die sich auf das Gesamtergebnis negativ auswirken. Zusätzlich ist die Geschwindigkeit des Fahrzeugs an dieser Stelle deutlich reduziert, da mehrere Schlaglöcher die Fahrt erschweren. Es ist somit zusätzlich davon auszugehen, dass das gewählte Bewegungsmodell diese komplexen Bewegungen nicht ausreichend modelliert und es somit zu weiteren Fehlern kommt.

Mehrere Tests haben gezeigt, dass Lateralabweichungen zur Fahrspur von bis zu $5.5\,\text{m}$ und einer Dauer von ca. $15\,\text{s}$ nur einen geringen Einfluss auf das Gesamtergebnis haben, sofern das Fahrzeug anschließend wieder zurück auf den ursprünglichen Pfad fährt. Bei zweispurigen Straßen ist somit ein Überholen und das Ausweichen auf die andere Fahrspur ohne Probleme möglich. Bei Winkelabweichungen von mehr als $15°$ und einer Dauer von $4\,\text{s}$ bei einer durchschnittlichen Geschwindigkeit von $7.1\,\text{m/s}$

war das MTT Verfahren und die MCL nicht mehr in der Lage, die Datenassoziation zu lösen und zu einer globaler Kartenposition zu konvergieren.

In Abbildung 7.4 ist die Auswertung der Metriken für die MCL sowie das fusionierte Gesamtergebnis des Graph-SLAM-Verfahrens über die Zeit getrennt geplottet. Der RMSE der MCL ist im Vergleich zum fusionierten Ergebnis des Graph-SLAM-Backends höher. Das Graph-SLAM-Backend verwendet alle Messungen und die bestimmten Gewichte der Karten-Landmarken über einen definierten Zeithorizont (in die Vergangenheit), um die beste Schätzung aller Landmarken und Posen zu erhalten. Zusätzlich werden durch die Fusion aller Daten insbesondere Sprünge herausgefiltert und der Gesamtfehler im Vergleich zur reinen MCL reduziert. Dies ist auf die bessere Behandlung von Nichtlinearitäten und die Tatsache zurückzuführen, dass die Messungen alle lokal und für einen gewissen Zeitraum betrachtet werden (Graph-SLAM).

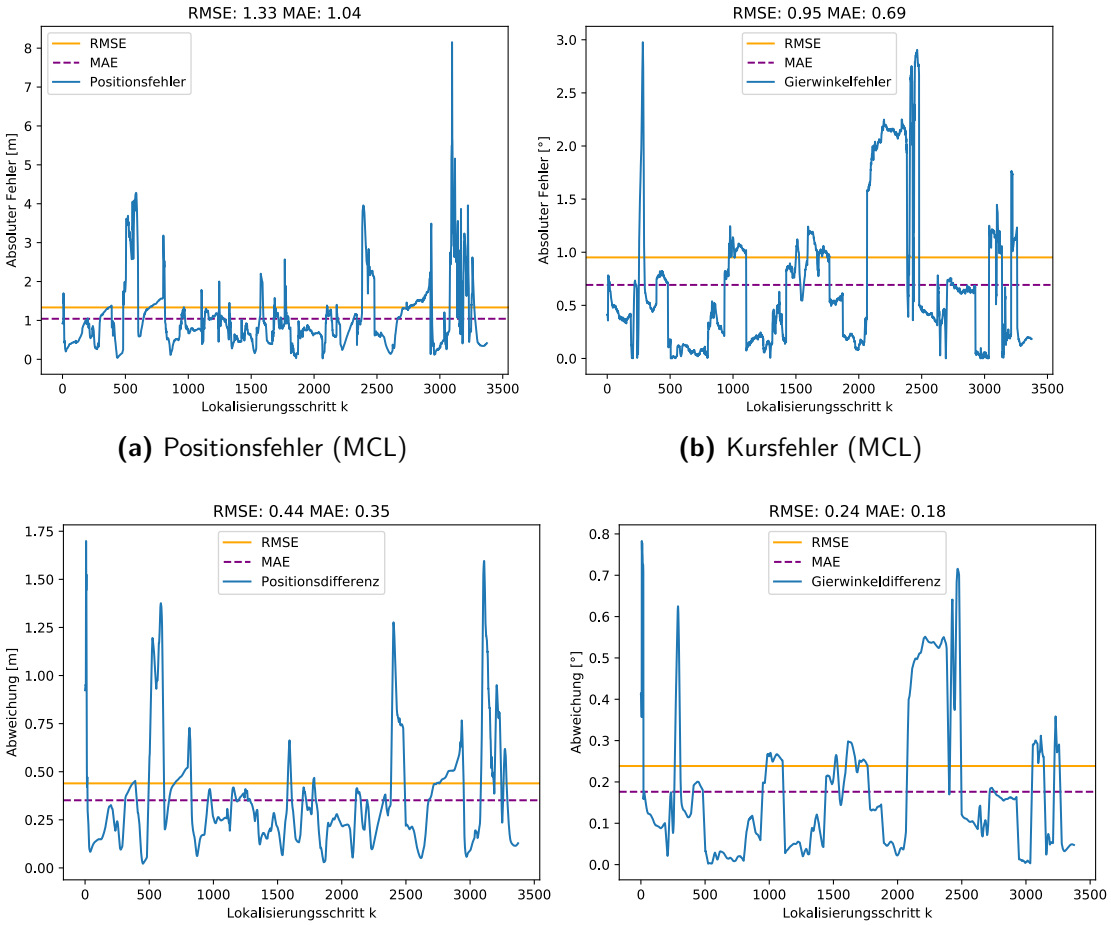
Die globalen Beschränkungen, welche durch die MCL hinzugefügt werden, helfen speziell bei der globalen Konvergenz und der Korrektur der auftretenden Driftfehler des Graph-SLAM Verfahrens.

7.3.2 Auswirkungen der reduzierten Datenübermittlung

Um die Auswirkung der Datenreduktion auf die MCL untersuchen zu können, wurden unterschiedliche Übertragungstests auf den Daten der Testsequenz 1 durchgeführt, die auf dem Standortübungsplatz München aufgenommen wurden.

Die Tabelle 7.2 beinhaltet die Gegenüberstellung der Positions- und Kursfehler für die ursprüngliche (nicht reduzierte) Pfad-Karte und der in Abhängigkeit der Kartierungsdistanz Δd_{max} reduzierten Pfad-Karte sowie die zur Übertragung benötigte Datenrate C_{UDP}. Die Kartierungsdistanz gibt an, in welchen Abständen die Pfad-Knoten vom Führungsfahrzeug an das Folgefahrzeug übertragen und entsprechend rekonstruiert werden. Für Testsequenz 1 konnte die Übertragungsrate von $13.23\,\mathrm{kbit/s}$ auf bis zu $0.81\,\mathrm{kbit/s}$ reduzieren werden. Die Übertragungsrate ist neben der Anzahl von Pfad-Knoten auch abhängig von der Anzahl von Landmarken. Da die Anzahl der Landmarken jedoch im Vergleich viel kleiner ist, sind die Auswirkungen zu vernachlässigen.

Die Kartierungsdistanz Δd_{max} gibt an, in welchen Abständen die Pfad-Knoten vom Führungsfahrzeug an das Folgefahrzeug übertragen und entsprechend rekonstruiert werden. Bei der nicht reduzierten Kartierungsdistanz (siehe erste Zeile der Tabelle 7.2) werden die Kartendaten in Abhängigkeit der Zykluszeit des Velodyne-Sensors geschrieben. Dabei erfolgt keine Berücksichtigung der aktuellen Geschwindigkeit, der Lenkwinkeländerung oder explizite Reduktion der Datentypen auf die minimale mögliche Anzahl an numerischen Stellen, was sich in einer erhöhten Übertragungsrate widerspiegelt. Die durchschnittliche Geschwindigkeit des Folgefahrzeugs für Testsequenz 1 betrug $4.5\,\mathrm{m/s}$ und die Höchstgeschwindigkeit $10.5\,\mathrm{m/s}$. Für die nicht reduzierte Pfad-Karten Übertragung ergibt sich eine durchschnittliche und maximale Kartierungsdistanz von $0.45\,\mathrm{m}$ und $1.05\,\mathrm{m}$.



(a) Positionsfehler (MCL)

(b) Kursfehler (MCL)

(c) Positionsfehler (Graph-SLAM)

(d) Kursfehler (Graph-SLAM)

Abbildung 7.4:
Vergleich der fusionierte Graph-SLAM und der MCL bestimmten Positionsschätzung im globalen UTM-Kartenraum für Testsequenz 4.

Tabelle 7.2:
Auswertung von verschiedenen Kartierungsdistanzen auf den RMSE und MAE für Testsequenz 1.

Δd_{max} [m]	Position RMSE [m]	Position MAE [m]	Kurswinkel RMSE [°]	Kurswinkel MAE [°]	C_{UDP} [kbit s^{-1}]
nicht reduziert	1,24	1,03	1,03	0,77	13,23
0,5	1,30	1,08	1,03	0,7	3,22
1,5	1,08	0,94	1,00	0,71	1,72
3,0	1,32	1,1	1,65	1,14	1,22
5,0	1,34	1,13	1,68	1,22	0,99
15,0	1,44	1,24	2,74	1,66	0,81

Durch die adaptive Änderung der Kartierungsdistanz unter Berücksichtigung der aktuellen Geschwindigkeit und Lenkwinkeländerung, lässt sich die Anzahl der Pfad-

Knoten und die Landmarkenverknüpfungen reduzieren. Die Erstellung der reduzierten Pfad-Karte wurde bereits in Unterabschnitt 6.2.1 und Unterabschnitt 6.2.1.2 beschrieben.

Das Folgefahrzeug war bis zu einer Kartierungsdistanz von $\Delta d_{\mathsf{max}} \leq 15\,\mathrm{m}$ in der Lage, sich entlang der Pfad-Karte zu lokalisieren. Bei einer größeren Interpolationsdistanz war das Verfahren hingegen nicht mehr in der Lage zu konvergieren.

Es ist festzuhalten, dass sich mit steigender Kartierungsdistanz der Lokalisierungsfehler erhöht. Betrachtet man den MAE für $\Delta d_{\mathsf{max}} = 0.5\,\mathrm{m}$ und $\Delta d_{\mathsf{max}} = 15\,\mathrm{m}$, so erhöht sich der Positionsfehler um $15\,\%$ und der Gierwinkel um $137\,\%$.

Durch eine zu groß gewählte Interpolationsdistanz werden somit wichtige Landmarkeninformationen verworfen, wodurch nicht mehr sichergestellt ist, dass zu jeder Zeit Karten-Landmarken sichtbar sind und diese auch richtig assoziiert werden können. Der Gierwinkelfehler steigt durch falsch assoziierte Landmarken überproportional stark an. Dies ist insbesondere in Bereichen festzustellen, wo sich viele Landmarken auf engem Raum befinden und es somit zu einer erhöhten Mehrdeutigkeit kommt. Durch eine falsche Zuordnung bleibt die Positionsdifferenz meistens gering, jedoch ändert sich der Gierwinkel stark.

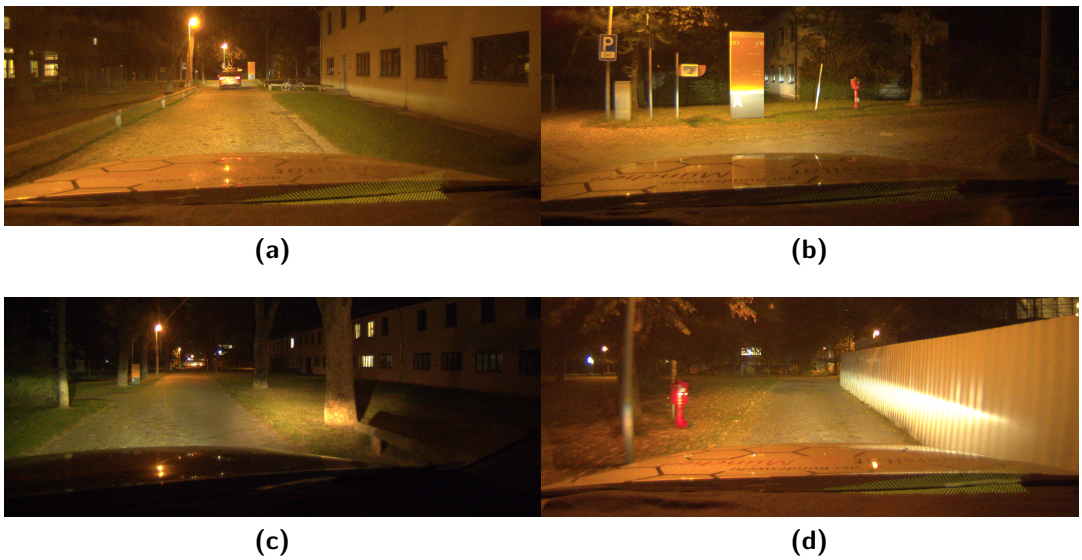


Abbildung 7.5:
Beispielaufnahme einer Testsequenz auf dem Campus der UniBw M bei Dunkelheit.

7.3.3 Campus der Universität der Bundeswehr München (UniBw M)

7.3.3.1 Beschreibung

Das Gelände der UniBw M ist als Campus-Universität angelegt. Alle Einrichtungen sowie Wohngebäude der Studenten befinden sich auf dem Universitätsgelände. Die Infrastruktur ist mit einer Kleinstadt zu vergleichen. Neben zweispurigen Hauptstraßen (ohne Straßenmarkierungen) verknüpfen Anliefer- und Fußwege die verschiedenen Gebäude. Im direkten Vergleich zum Standortübungsplatz München, sind neben Vegetationslandmarken viele zusätzliche vertikale stangenähnliche Landmarken vorhanden, die zur Lokalisierung berücksichtigt werden können. Um den Ansatz im Hinblick auf die unterschiedlichen Landmarken sowie Bodenbeschaffenheiten zu untersuchen, ermöglicht der UniBw M-Campus ein sehr abwechslungsreiches Erscheinungsbild.

Die Belichtungssituation bei Dunkelheit stellte durch die Verwendung eines LiDAR-Sensors keine besondere Herausforderung dar. In Abbildung 7.5 sind Onboard-Kamerabilder der Nachtsequenz 2 dargestellt, welche die schwierige Beleuchtung zeigen.

Um die Robustheit des Ansatzes zu demonstrieren, wurden Testfahrten bei Tageslicht und in Dunkelheit unternommen. Neben einer hohen Anzahl an weiteren Verkehrsteilnehmern (Fahrzeuge, Fußgänger) bei Tag, waren bei der Fahrt in Dunkelheit keine weiteren Verkehrsteilnehmer außerhalb des Konvois sichtbar.

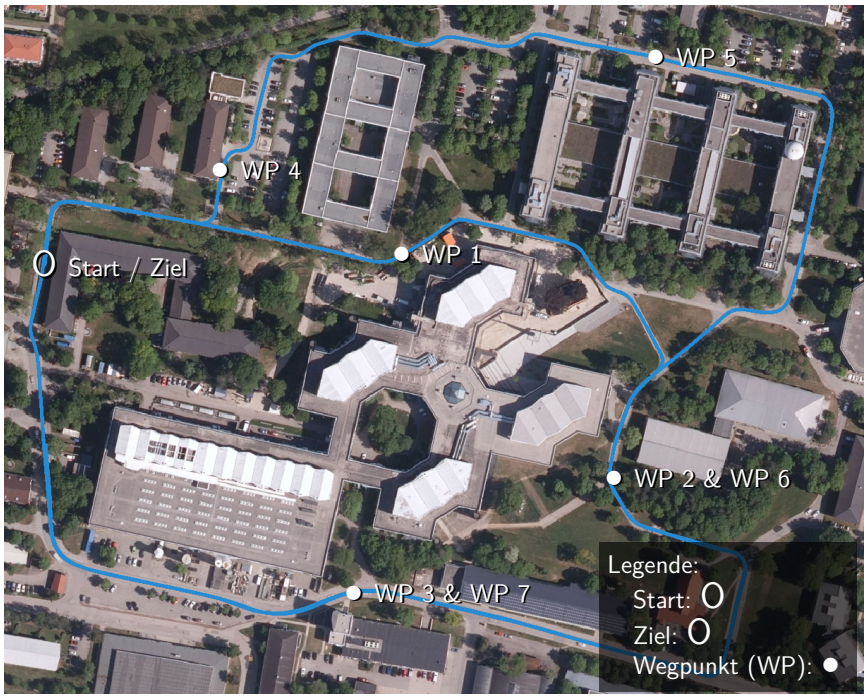


Abbildung 7.6:
Luftbild vom UniBw M Campus in Neubiberg bei München. Die in Blau visualisierte Linie stellt die gefahrene Strecke mit einer Gesamtlänge von $2300\,\mathrm{m}$ dar. Die Fahrt wurde an der Markierung **O** gestartet und es wurden die Wegpunkte WP 1 bis WP 7 nacheinander abgefahren, bis das Ziel **O** erreicht wurde. Bei der Fahrt wurde zweimal die Schleife (jeweils im Uhrzeigersinn) am Start und Zielpunkt geschlossen.

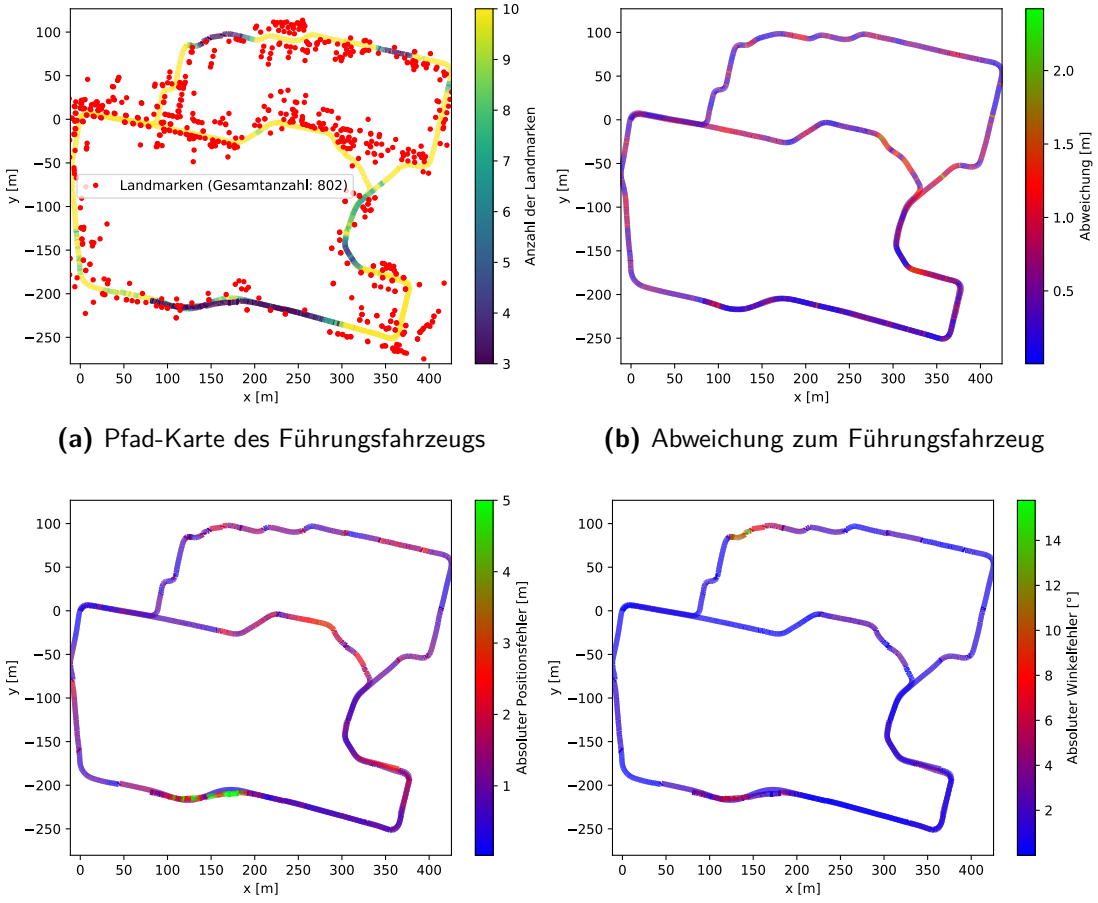
7.3.3.2 Ergebnisse

Zur Auswertung wurden Daten von acht verschiedene Testsequenzen mit einer durchschnittlichen Gesamtstrecke von $2300\,\mathrm{m}$ pro Testsequenz herangezogen, wobei die durchschnittliche Landmarkenanzahl bei 805 und die einfache Standardabweichung bei acht Landmarken lag. In Abb. 7.6 ist ein Luftbild des Campus sowie eine vollständige Trajektorie für Nachsequenz 2 zu sehen. Neben der gefahrenen Wegstrecke in Blau sind der Start, das Ziel sowie die Wegpunkte eingezeichnet.

In Abbildung 7.7d ist die vollständig Trajektorie des Folgefahrzeugs für die Nachtsequenz 2 dargestellt. Der Konvoi startet an den Koordinaten $(x = 0, y = 0)$ und die MCL konvergiert bereits nach $3\,\mathrm{m}$ Fahrt. Dies ist auf die eindeutige Anordnung der Landmarken um das Fahrzeug zurückzuführen, die an dieser Stelle hauptsächlich aus gut wahrnehmbaren Baumstämmen bestehen. Der initiale Positionsfehler des Führungsfahrzeugs und des Folgefahrzeugs liegt zur Referenzlösung bei unter einem Meter.

Die Trajektorie des Führungsfahrzeug führt vom Startpunkt $(x = 0, y = 0)$ nach Osten über Wegpunkt 1 $((x = 220, y = -25))$ und Wegpunkt 2 über den südlichen Wegpunkt 3 $((x = 190, y = -210))$ im Uhrzeigersinn zurück zum Startpunkt.

Um die Schleife erneut zu schließen, fährt das Führungsfahrzeug anschließend die äußere Schleife der in Abbildung 7.7a dargestellten Trajektorie über den nördlichen



(a) Pfad-Karte des Führungsfahrzeugs

(b) Abweichung zum Führungsfahrzeug

(c) Abs. Positionsfehler des Folgefahrzeug

(d) Abs. Kursfehler des Folgefahrzeug

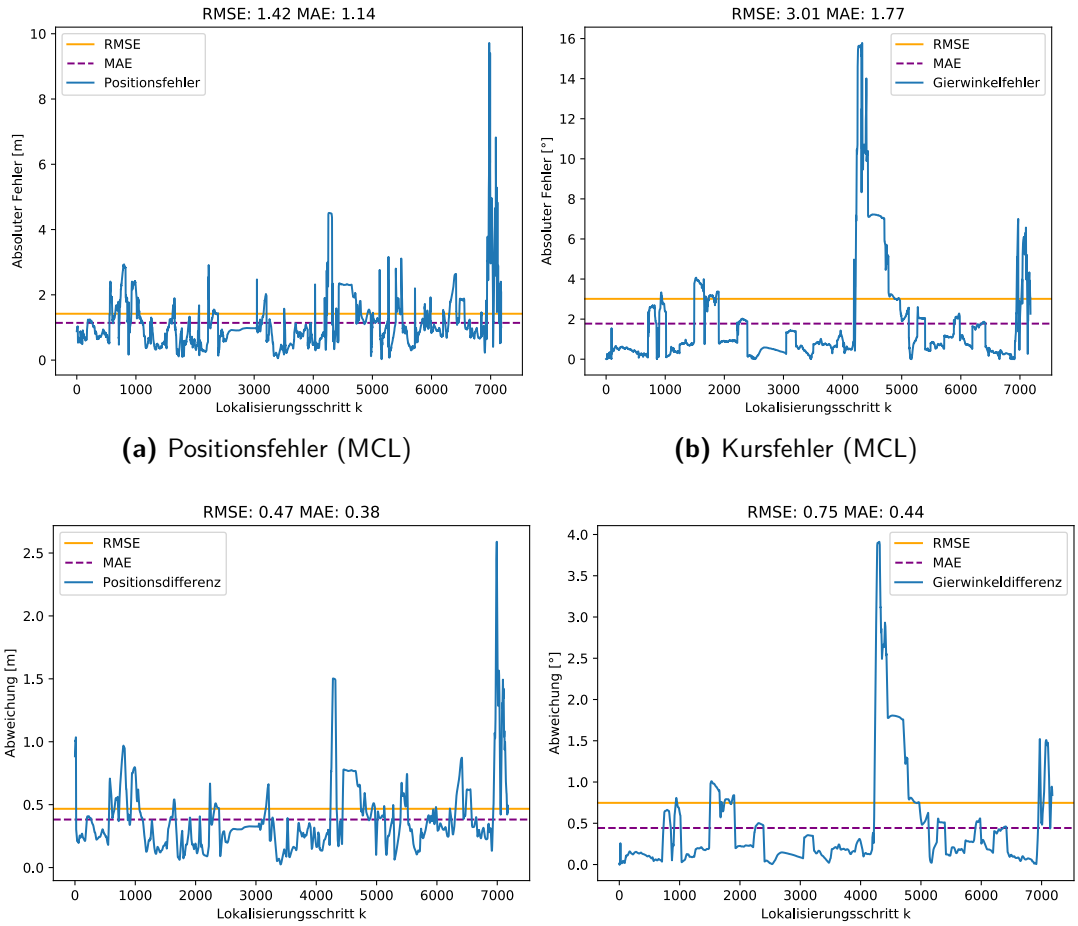
Abbildung 7.7:
Nachtsequenz 2: Auswertung und Visualisierung der geschätzten Fahrzeugpositionen der MCL des Folgefahrzeugs auf Basis der Pfad-Karte des Führungsfahrzeugs. Der Maximalwert der Farbkodierung wurde für eine bessere Darstellung händisch gewählt und entspricht dabei nicht dem Maximalwert der Kennzahl.

Teil an den Wegpunkten 4, 5, 6 und 7 vorbei bis zum Ziel, siehe Abbildung 7.6. Ab den Koordinaten $(x = 150, y = 90)$ sind die Landmarken um das Fahrzeug sehr ungleichmäßig verteilt und zusätzlich ist die Anzahl der Landmarken deutlich reduziert. Dies führt zu einer Verschlechterung der Positions- und Kursschätzung, die in allen Testsequenzen sichtbar ist. Insbesondere der Kursfehler steigt deutlich und bleibt länger bestehen. Der gleiche Effekt wurden bereits in Unterabschnitt 7.3.1.2 beobachtet.

Nach dem Passieren von Wegpunkt 7 ($(x = 300, y = -230)$ bis $(x = 70, y = -210)$), ist die Landmarkenanzahl deutlich reduziert, was auf ein Baustellenfahrzeug zurückführen ist, welches sich über eine längere Wegstrecke direkt vor dem Führungsfahrzeug befindet. Das Baustellenfahrzeug verdeckt dabei große Teile des Sichtbereichs und so werden Sensor-Landmarken erst sehr spät als Zustand im MTT aufgenommen und folglich verzögert im SLAM-Verfahren als Landmarke berücksichtigt und kartiert.

Vergleicht man Abbildung 7.7a und Abbildung 7.7c so ist ein direkter Zusammenhang zwischen Landmarkenanzahl und erhöhtem Positionsfehler für die gleichen Teilbereiche zu erkennen. Da die Lokalisierung entlang der Pfad-Karte erfolgt, werden zuvor kartierte Landmarken der gleichen Stelle nicht berücksichtigt, da diese nicht dem aktuellen Pfad zugewiesen wurden.

In Abbildung 7.8 sind die jeweiligen Metriken für die MCL und die fusionierte Graph-SLAM Lösung über die Zeit exemplarisch für die Nachtsequenz 2 dargestellt. Die Ergebnisse sind vergleichbar mit den im vorherigen Abschnitt erzielten Ergebnissen auf dem Standortübungsplatz München, siehe Abbildung 7.3c. Zusätzlich ist eine Verbesserung der Positionsschätzung der fusionierten Lösung des Graph-SLAM-Backends (RMSE: $0.47\,\mathrm{m}$) im Vergleich zur MCL (RMSE: $1.42\,\mathrm{m}$) zu erkennen.



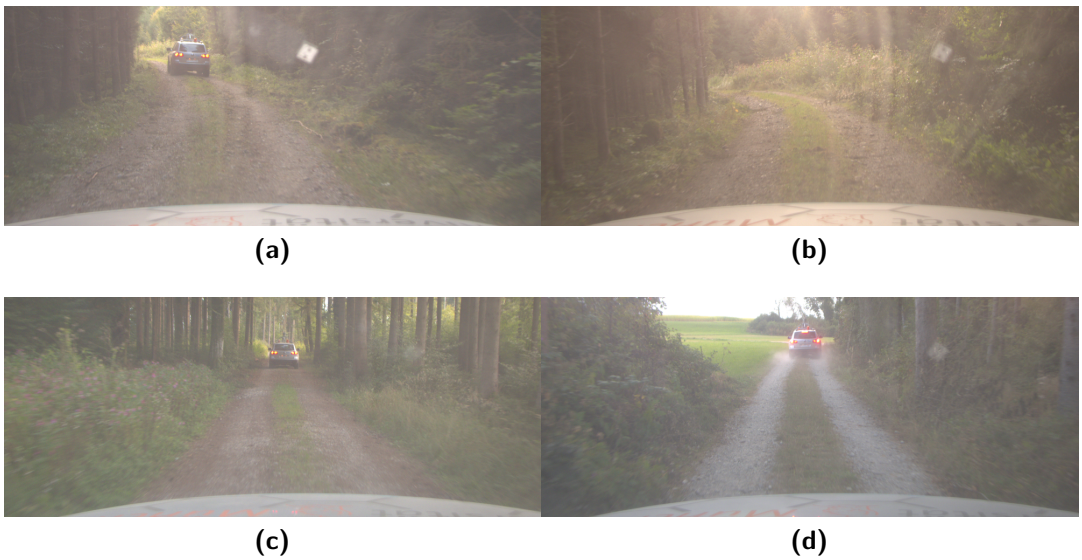
(a) Positionsfehler (MCL)

(b) Kursfehler (MCL)

(c) Positionsfehler (Graph-SLAM)

(d) Kursfehler (Graph-SLAM)

Abbildung 7.8:
Vergleich der fusionierte Graph-SLAM und der MCL bestimmten Positionsschätzung im globalen UTM-Kartenraum für Nachtsequenz 1.



(a) **(b)**

(c) **(d)**

Abbildung 7.9:
Beispielaufnahme der Testsequenz auf einem Waldweg in Aying, Bayern.

7.3.4 Waldweg in Aying

7.3.4.1 Beschreibung

Zur Erprobung und Auswertung des Gesamtsystems wurde eine weitere Testfahrt in einem Waldstück bei Aying in Bayern durchgeführt. Die Fahrt dauerte $120\,\mathrm{s}$ wobei eine Strecke von $480\,\mathrm{m}$ durch einen stark bewachsenen Wald und unstrukturiertes Gelände zurückgelegt wurde. Die in Abbildung 7.9 dargestellten Onboard-Kamerabilder zeigen das schwierige Gelände und die herausfordernden Belichtungsbedingungen. Die Verwendung der LiDAR-basierten Straßenschätzung und die daraus bestimmten Straßenlandmarken war aufgrund der Bodenbeschaffenheit mit den festgelegten Parametern nicht möglich und findet somit keine Anwendung im SLAM-Verfahren.

Zusätzlich war im Wald zu großen Teilen kein GNSS-Signal verfügbar, was zu Positions- und Kurssprüngen der Referenzlösung führte. Die ermittelten Fehler sind daher nur mit Vorsicht zu betrachten und eher als qualitative Merkmale zu verstehen.

7.3.4.2 Ergebnisse

Das Luftbild in Abbildung 7.10 stellt die gefahrene Strecke der Testsequenz 1 Aying dar, bei der insgesamt 359 Landmarken kartiert wurden. Zur Auswertung war eine Vorabkartierung anhand des Luftbildes aufgrund der dicht bewachsenen Vegetation nicht möglich. Aus der Luftaufnahme ist bereits zu erkennen, dass es sich um einen sehr dicht bewachsenen Wald handelt.

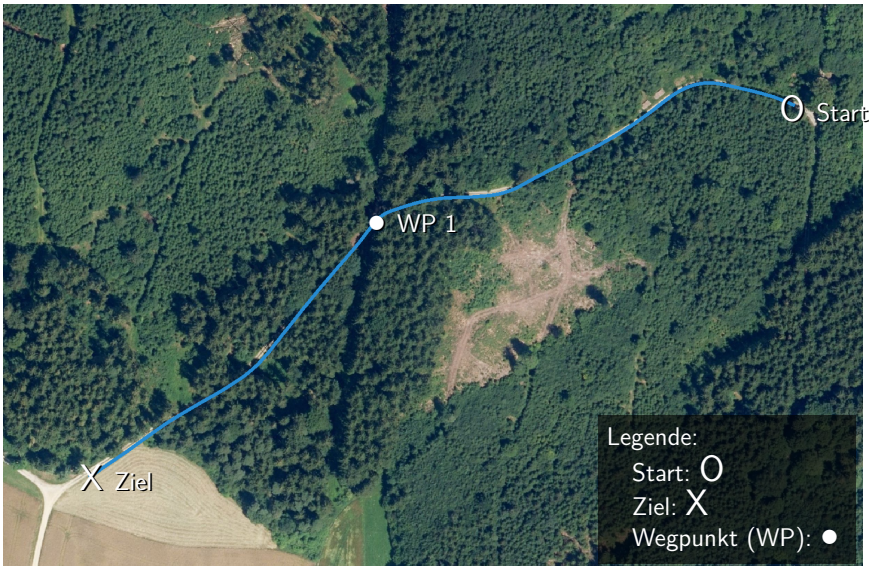


Abbildung 7.10:
Luftbildaufnahme und Trajektorie des Führungsfahrzeugs der Testsequenz Waldweg Aying. Die in Blau visualisierte Linie stellt die gefahrene Strecke mit einer Gesamtlänge von $480\,\text{m}$ dar. Die Fahrt wurde an der Markierung **O** gestartet bis das Ziel **X** erreicht wurde.

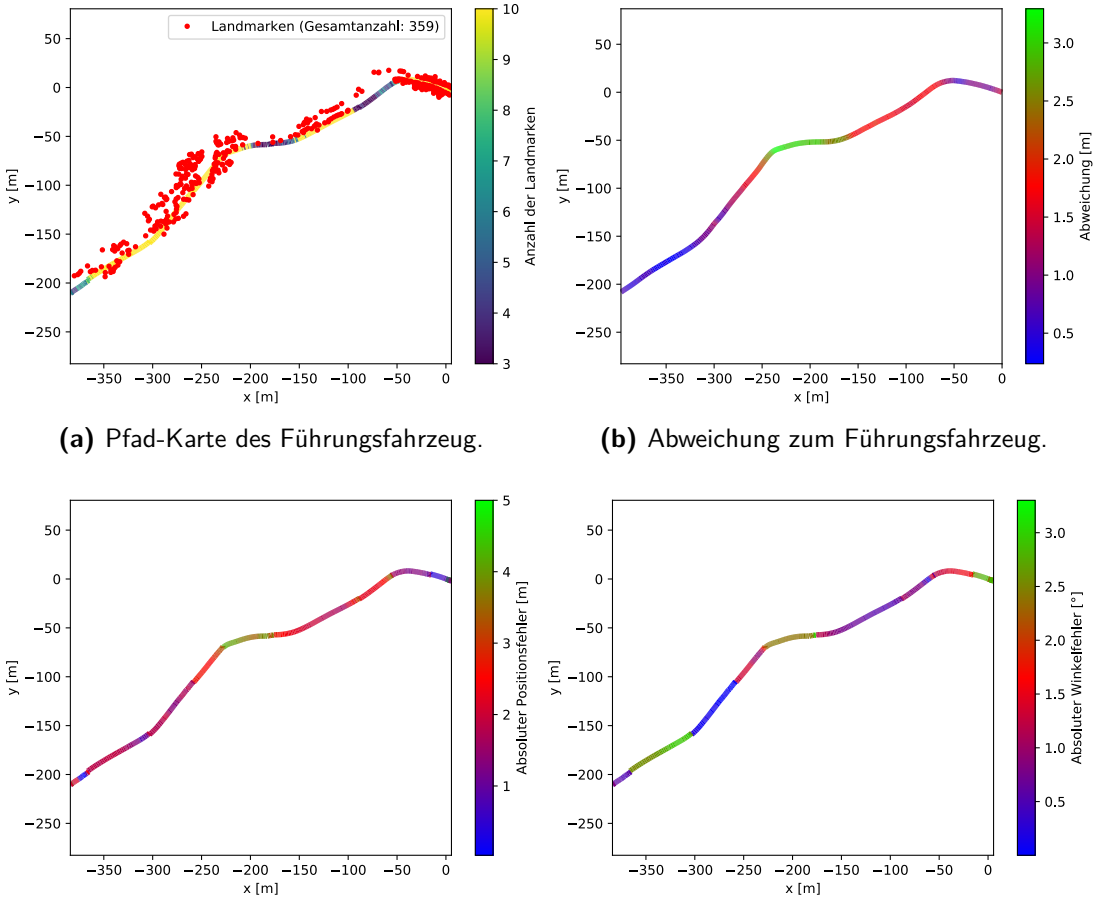
Die dargestellte Pfad-Karte des Führungsfahrzeugs in Abbildung 7.11a basiert somit nur auf der mittels Graph-SLAM ermittelten Fahrzeugtrajektorie und den Sensor-Landmarken. Zu Beginn wurden die Fahrzeugpositionen des Führungs- und Folgefahrzeugs jeweils mit der initialen GNSS-Position der Referenzlösung initialisiert.

Die Auswertung entlang der Pfad-Karte ist in Abbildung 7.11 aufgezeigt. Die Fahrt startet an den Koordinaten $(x = 0, y = 0)$.

Bei der Initialisierung des Systems kommt es zu einem großen Positions- und Kursfehler. Dies ist zum einen auf die mittels Referenzlösung initialisierte Pose des Führungs- und Folgefahrzeug zurückzuführen, die im Wald für beide Fahrzeuge einen geschätzten Fehler von $> 15\,\text{m}$ aufwies. Zum anderen wird dies durch eine fehlerhafte Sensor-Landmarken zu Karten-Landmarken Assoziation verursacht. Nach ca. $50\,\text{m}$ Fahrt war die MCL jedoch in der Lage, die Mehrdeutigkeiten aufzulösen. Durch einen stärkeren Richtungswechsel (bei ca. $(x = -50, y = 10)$), wird der Betrachtungsraum der Partikel stärker eingeschränkt und die vergangene Trajektorie hat einen erhöhten Einfluss auf die Gewichtung der Partikel.

Die Auswertung in Abb. 7.12 zeigt die bestimmten Metriken für die MCL und die fusionierte Graph-SLAM Lösung über die Zeit exemplarisch für die Testsequenz 1 Aying. Die Ergebnisse sind vergleichbar mit den im vorherigen Abschnitt erzielten Ergebnissen.

Zusätzlich ist eine Verbesserung der Positionsschätzung der fusionierten Lösung des Graph-SLAM-Backends (RMSE: $1.17\,\text{m}$) im Vergleich zur MCL (RMSE: $2.45\,\text{m}$) zu erkennen. Darüber hinaus ist insbesondere die durch das Graph-SLAM bestimmte Posenschätzung deutlich homogener und ohne größere Sprünge. Dies ist auf die fehlenden a-priori Karteninformationen aus Luftbildern oder OSM zurückzuführen,



(a) Pfad-Karte des Führungsfahrzeug.

(b) Abweichung zum Führungsfahrzeug.

(c) Abs. Positionsfehler des Folgefahrzeug.

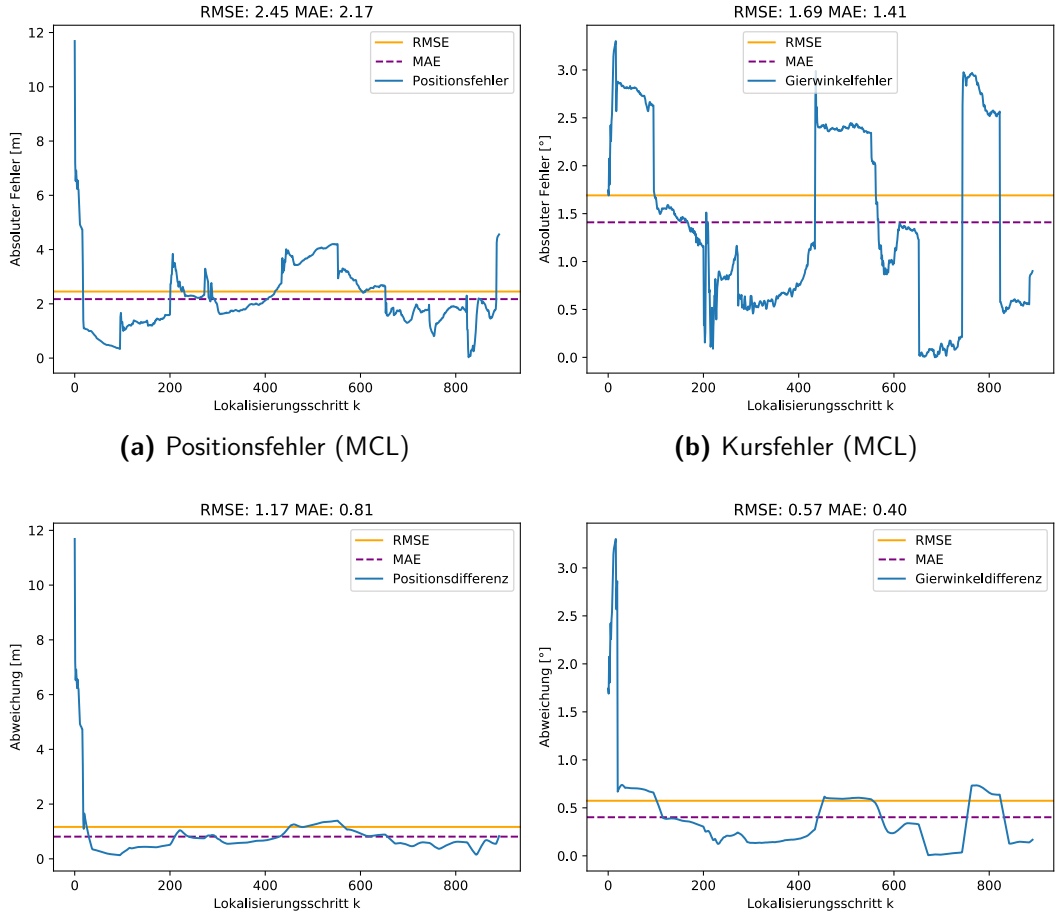
(d) Abs. Kursfehler des Folgefahrzeug.

Abbildung 7.11:
Testsequenz 1 Aying: Auswertung und Visualisierung der geschätzten Fahrzeugpositionen der MCL des Folgefahrzeugs auf Basis der Pfad-Karte des Führungsfahrzeugs. Der Maximalwert der Farbkodierung wurde für eine bessere Darstellung händisch gewählt.

die bei einer falschen Datenassoziation einen globalen Prior zum Graphen hinzufügen und das Ergebnis somit deutlich verschlechtern.

In Abbildung 7.13a ist die Referenzlösung (Folgefahrzeug mit roter Bounding-Box) gegenüber der Graph-SLAM Posenschätzung (Folgefahrzeug mit grüner Bounding-Box) für den gleichen Zeitschritt dargestellt. Es ist ein deutlicher Positionsversatz zu erkennen. Im nächsten Zeitschritt, siehe Abbildung 7.13b, korrigiert die Referenzlösung in Richtung der aktuell geschätzten Graph-SLAM Fahrzeugpose. Fehler in der Referenz-Lösung wirken sich somit direkt auf die bestimmten Metriken MAE und RMSE aus.

In Abb. 7.14 ist der Initialisierungsschritt des Folgefahrzeugs dargestellt. Es befinden sich eine hohe Anzahl von Bäumen (Rot) entlang der Führungsfahrzeug-Trajektorie (blaue Linie). Die Datenassoziation zwischen Sensor-Landmarken (Grün) mit den Karten-Landmarken (Rot) ist für die MCL nicht eindeutig lösbar.



(a) Positionsfehler (MCL) **(b)** Kursfehler (MCL)

(c) Positionsfehler (Graph-SLAM) **(d)** Kursfehler (Graph-SLAM)

Abbildung 7.12:
Testsequenz 1 Aying: Vergleich der fusionierte Graph-SLAM und der MCL bestimmten Positionsschätzung im globalen UTM-Kartenraum für die Testsequenz Aying.

(a) Zeitschritt k **(b)** Zeitschritt $k+1$

Abbildung 7.13:
Darstellung der Referenzlösung (Rot) im Vergleich zur Graph-SLAM Pose in zwei aufeinander folgenden Zeitschritten. Es ist gut zu erkennen, dass die Referenzlösung zwischen den Zeitschritten k und $k+1$ in Richtung der aktuellen Graph-SLAM Pose springt.

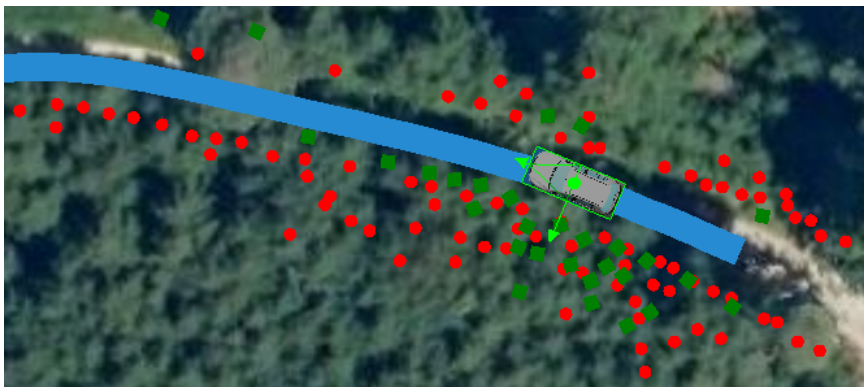


Abbildung 7.14:
Darstellung der Karten-Landmarken (Rot) und die Sensor-Landmarken (Grün) des Folgefahrzeugs. Die hohe Landmarkenanzahl sowie die Ähnlichkeit der Landmarkenkonstellation erschwert die eindeutige Zuordnung der Sensor-Landmarken mit den Karten-Landmarken.

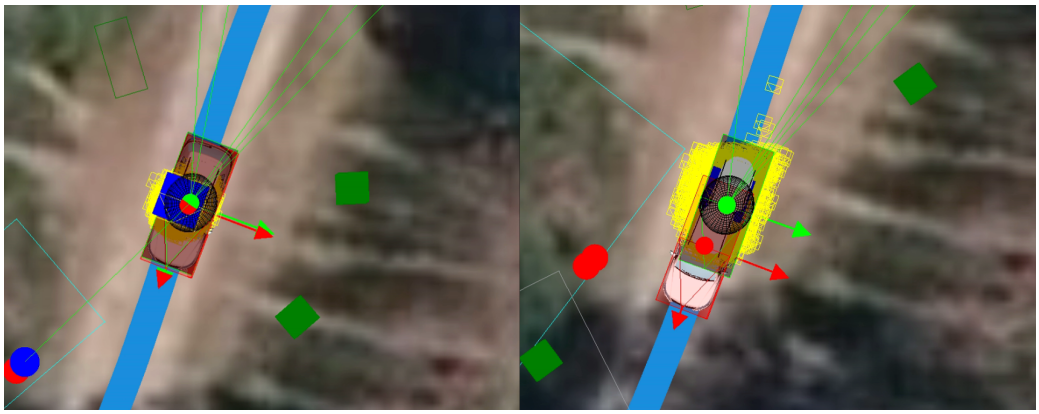
7.4 Beurteilung Gesamtsystem

Zusammenfassend ist festzuhalten, dass das System in allen ausgewerteten Testsequenzen (siehe Sektion 7.3) sowie in den darüber hinaus durchgeführten autonomen Testfahrten eine für den aufgesplitteten Konvoi in unstrukturiertem Gelände ausreichende Robustheit und Genauigkeit erzielen konnte.

An Teilstücken mit wenig robusten vertikalen Landmarken konnte die Positionsschätzung auf Basis der Straßenlandmarken gestützt werden. Darüber hinaus konnte gezeigt werden, dass die Kombination von MTT-Verfahren mit MCL speziell in komplexen Umgebungen und mit dynamischen Objekten die Datenassoziation lösen und bewegte Objekte aus dem Schätzprozess entfernen kann.

Zusätzlich hat sich gezeigt, dass die Anzahl der Landmarken sowie deren Konstellation einen direkten Einfluss auf die globale Positionsschätzung haben. Treten Landmarken in sehr hoher Anzahl und mit einem geringem Abstand auf, so ist eine eindeutige Zuordnung der Sensor-Landmarken und Karten-Landmarken nicht mehr möglich und es kann zu Abweichungen in der Position und Orientierung kommen. Das Modell, welches die Vegetationslandmarken nur durch eine Position, Dimension sowie die bestimmten Unsicherheiten beschreibt, kommt in diesem Fall an seine Grenzen.

Darüber hinaus ist ebenfalls die Landmarkenkonstellation entscheidend. Das Beispiel in Abbildung 7.15 zeigt die Auswirkung einer einseitigen Landmarkenkonstellation auf die MCL, wobei die Graph-SLAM Fahrzeugpose in Grün sowie die Referenzposition in Rot dargestellt ist. Die vom Fahrzeug abgehenden grünen Linien zeigen die Richtung zur aktuellen assoziierten Karten-Landmarke an. In Abbildung 7.15a sind eine Karten-Landmarke vor dem Fahrzeug sowie vier Karten-Landmarken hinter dem Fahrzeug sichtbar. Das Koordinatensystem der Positionsschätzung (hellgrüner Punkt mit X-Y Koordinaten) kommt der Referenzlösung (roter Punkt mit X-Y Koordinaten) sehr nahe. Die Partikel der MCL sind als gelbe Bounding-Box dargestellt und liegen sehr gut auf der Referenzlösung.



(a) Zeitschritt k **(b)** Zeitschritt $k + 10$

Abbildung 7.15:
Ungleichmäßig verteilte Landmarken haben eine direkte Auswirkung auf die Positionsgenauigkeit. In den gezeigten Abbildungen sind die Referenzlösung sowie die Graph-SLAM Schätzung durch Fahrzeugmodelle mit einer roten sowie grünen Hülle und gleichfarbigem Koordinatensystem dargestellt. In Abbildung 7.15a sind Landmarken vor und hinter dem Fahrzeug sichtbar. Der Positionsfehler zur Referenzlösung ist somit gering (grünes und rotes Zentrum der Koordinatensysteme liegen übereinander). In den folgenden Zeitschritten sind nur noch Landmarken hinter dem Fahrzeug sichtbar. Die Positionsunsicherheit nimmt zu und die MCL streut Partikel (Gelb). Der zeitliche Versatz zwischen den gezeigten Zeitschritten beträgt ca. $1\,\mathrm{s}$.

In Abbildung 7.15b ist die gleiche Szene zehn Zeitschritte später dargestellt. Es ist zu erkennen, dass die Karten-Landmarken nicht richtig mit den Sensor-Landmarken assoziiert werden können und die Assoziation nur für Landmarken hinter dem Fahrzeug erfolgt. Dies erhöht die Positionsunsicherheit der MCL, was durch die größere

Verteilung der Partikel dargestellt wird. Die geschätzte Pose (hellgrüner Punkt) weicht folglich von der Referenzlösung ab.

Zusätzlich ist anzumerken, dass auf Basis der genannten Einschränkungen die Anforderungen an die Landmarkenkonstellation sowie Landmarkenanzahl insbesondere für eine globale Initialisierung in einer gegebenen Karte gegeben sein muss.

8 Zusammenfassung und Diskussion

8.1 Zusammenfassung

In diesem abschließenden Kapitel werden die wesentlichen Ergebnisse zusammengefasst und auf Basis der in Unterabschnitt 1.2.1 definierten Forschungsziele besprochen. Das in dieser Arbeit vorgestellte Framework unterliegt den in Unterabschnitt 1.2.1 und Unterabschnitt 1.2.2 definierten Forschungszielen sowie Randbedingungen, welche vollständig eingehalten werden konnten.

In dieser Arbeit wurde ein SLAM-Framework vorgestellt, das eine spärliche Kartenrepräsentation im Schätzprozess integriert, damit Konvoi-Teilnehmer in unstrukturierten Gebieten sicher ein Ziel erreichen können. Neben dem Anwendungsfall aufgesplitteter Konvoi, kann das Framework auch für die normale autonome Fahrt verwendet werden.

Um dies zu ermöglichen, wurden mehrere Teilkomponenten des SLAM-Framework neu entwickelt. Das Gesamtsystem umfasst eine Monte Carlo Lokalisierung mit neuartigem Messmodell, ein Multi-Target-Tracking-Verfahren zur Landmarkenbestimmung und ein Graph-SLAM Verfahren, das die Ergebnisse probabilistisch fusioniert, um simultan eine Karte zu erstellen und sich dabei gleichzeitig entlang eines gegebenen Pfades global zu lokalisieren.

Das vorgestellte Framework verwendet dazu Landmarkentypen, die von Bäumen und Büschen in unstrukturierten Bereichen bis hin zu Schildern und stangenähnlichen Strukturen variieren. Der globale Pfad wird durch eine sogenannte topologisch-metrische Pfad-Karte beschrieben. Die Pfad-Karte besteht dabei aus spärlich verteilten Kartenlandmarken, die nur durch eine Position und Dimension sowie die Positions- und Dimensionsunsicherheit beschrieben werden. Die sparsame Datenrepräsentation wurde gewählt, damit mittels schmalbandigem Funk die Pfad-Karte zwischen den Konvoiteilnehmern ausgetauscht werden kann und die Karte von einem Menschen lesbar ist.

Das Gesamtsystem wurde vielfältig und intensiv in verschiedenen Gebieten, bei Tag und Nacht und zu unterschiedlichen Jahreszeiten erprobt und ausgewertet. Dabei kamen unterschiedliche Folge- und Führungsfahrzeuge zum Einsatz. Das echtzeitfähige Gesamtsystem zeigte sich beständig gegenüber schwierigen Witterungsbedingungen und Dunkelheit. Zusätzlich konnte die Erprobung des Systems auf Basis von Karten durchgeführt werden, die bis zu zwei Jahre alt waren und zu unterschiedlichen Jahreszeiten aufgenommen wurden. Es hat sich somit gezeigt, dass die Auswahl von abstrakten Landmarkenrepräsentation zur Lokalisierung im autonomen aufgesplitteten Konvoi eine robuste Kartenrepräsentation darstellt.

8.2 Diskussion

Ein Forschungsziel dieser Arbeit war die Entwicklung und Erprobung eines Gesamtkonzepts für die simultane Lokalisierung und Kartierung von autonomen und aufgesplitteten Konvois für den Einsatz in unstrukturierten Gebieten.

Es konnte im Zuge dieser Arbeit gezeigt werden, dass topologisch-metrische Karten für die globale Navigation mit einem sensor-basierten Wahrnehmungssystem kombiniert werden können, um die Herausforderungen der autonomen Navigation in unstrukturierter Umgebung zu lösen.

Nach ausgiebiger Analyse von verfügbaren Kartenlandmarken und verschiedenen Sensortechnologien wurde ein einzelner LiDAR-Sensor mit einem Sichtbereich von $360°$ und einer Erfassungsdistanz von $120\,\mathrm{m}$ für die Landmarkendetektion ausgewählt.

Um Landmarken aus der Punktewolke effizient zu extrahieren, wurden verschiedene Verarbeitungsschritte entwickelt. Die vorgestellten Verfahren zur Punktwolkensegmentierung und Objekterzeugung erzielten gleichermaßen gute Ergebnisse in unstrukturierter aber auch in strukturierter Umgebung. Bei der Erprobung und Evaluierung hat sich darüber hinaus gezeigt, dass kein anderes visuelles System in der Lage war, vertikale Vegetationslandmarken sowie den Straßenverlauf robust über längere Zeiträume, Entfernungsdistanzen sowie Belichtigungsbedingungen und innerhalb der Zykluszeit zu erkennen, um die gestellten Anforderungen zu erfüllen.

Festzuhalten ist jedoch, dass insbesondere die LiDAR-basierte Straßenverlaufserkennung bei unstrukturierten Feldwegen mit stark inhomogenen Oberflächen schnell an ihre Grenze gestoßen ist. Für die Weiterentwicklung des Gesamtsystems wird hier somit die Verwendung und Erweiterung mit bildgebenden Kamerasystemen angeraten.

Neben der mittels Führungsfahrzeug erzeugten Pfad-Karte konnte ebenfalls die Lokalisierung entlang einer virtuellen Pfad-Karte erprobt werden, die auf Basis von Luftbildern oder Straßeninformationen aus OpenStreetMap erstellt wurde. Wichtig ist hier zu erwähnen, dass die Landmarkengenauigkeit bezogen auf die Klasse und Position dabei von der Realität abweichen kann und die Positionsgenauigkeit direkt proportional stark von der Kartenqualität abhängt. Im Allgemeinen konnten jedoch sehr gute Ergebnisse in städtischen Gebieten mit OpenStreetMap-Daten erzielt werden. Insbesondere in ländlichen Gegenden, wie auf dem Standortübungsplatz München und Aying, waren jedoch keine oder nur sehr wenige verwendbare Karteninformationen verfügbar. Der Straßenverlauf sowie die Karten-Landmarken mussten, wie bei OpenStreetMap üblich, teilweise händisch aus dem Luftbild annotiert werden.

Ein weiteres Ziel dieser Arbeit war es, die globale Lokalisierung innerhalb der spärlichen Karte zu ermöglichen, ohne die Berücksichtigung von GNSS oder hoch genauem Kartenmaterial. Die entwickelte Monte Carlo Lokalisierung benötigt für die globale Lokalisierung innerhalb der Pfad-Karte kein GNSS. Es hat sich jedoch gezeigt, dass in der Initialisierungsphase ein GNSS mit einer Positionsunsicherheit von bis zu $100\,\mathrm{m}$, trotz der hohen Unsicherheit dabei helfen kann, die Konvergenz der Monte Carlo Lokalisierung zu beschleunigen und zu verbessern. Dies liegt zum einen daran, dass

die grobe Orientierung der Fahrzeugpose und die Berücksichtigung der vergangenen Trajektorie viele Partikelanordnungen eliminieren kann. Zusätzlich kann durch eine initiale GNSS-Position der Zustandsraum eingegrenzt werden.

Ist darüber hinaus der Startpunkt des autonomen Konvoi bekannt, lässt sich dies im Framework auch unabhängig von globalen Fahrzeugkoordinaten berücksichtigen. So wird zusätzlich die Kartenerstellung und Lokalisierung in einem inertialen Raum ohne die Registrierung in einem globalen Koordinatensystem ermöglicht.

Bei der Erprobung hat sich zudem gezeigt, dass nach der Initialisierungsphase das System selbständig in der Lage war, sich von größeren Positionssprüngen zu erholen. Die Berücksichtigung der vergangenen Trajektorie sowie der aktuellen Straßenverlaufsschätzung konnte besonders in Bereichen stützen, in denen viele Mehrdeutigkeiten bei der Sensor-Landmarken zu Karten-Landmarken-Assoziation auftraten, sowie in Bereichen mit nur sehr wenigen vertikalen Landmarken.

Zu erwähnen ist weiter, dass die Annahme der Übereinstimmung der vergangenen Trajektorie nur dann funktioniert, sofern das Fahrzeug dem globalen Pfad des Führungsfahrzeugs oder einem Pfad aus OpenStreetMap folgt. Treten größeren Abweichungen oder sogar ein Verlassen des Pfades auf, so ist das System nicht mehr in der Lage sich zu lokalisieren. Als Lösung ist denkbar, dass das System die Abweichung und das Verlassen automatisch erkennt und auf Basis des Straßennetzwerks von OpenStreetMap eine neue Route zurück zum gegebenen Pfad plant. Bei dem neu geplanten Teilstück können dann ebenfalls OpenStreetMap-Landmarken der virtuellen Pfad-Karte zugeordnet werden.

In besonders anspruchsvollem Terrain, wie beispielsweise durch Wälder oder entlang Forstwirtschaftsstraßen und Gegenden mit gleichzeitig sehr wenigen vertikalen Landmarken, konnte das SLAM-Verfahren die Fahrzeugbewegung nicht ausreichend abbilden. Es hat sich gezeigt, dass das verwendete Bewegungsmodell die Fahrzeugbewegung nur über einen kurzen Zeithorizont präzise genug vorhersagen kann und die längerfristige Prädiktion einen größeren Fehler verursacht. Als Erweiterung dieser Arbeit wird somit eine enge Kopplung des SLAM-Verfahrens mit einer Bewegungsschätzung unter Verwendung sämtlicher Messgrößen der Onboard-Sensorik wie beispielsweise INS, GNSS, Magnetometer sowie Drehradgeber vorgeschlagen.

Symbolverzeichnis

Abkürzungen

ACC	Adaptive Cruise Control
ACT	Augmented Coordinated Turn
BOVW	Bag Of Visual Words
BOW	Bag Of Words
CTRV	Constant Turn Rate and Velocity model
DBN	Dynamisches Bayes'sche Netzwerk
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DOF	Degree of Freedom
EKF	Extended Kalman Filter
ELROB	European Land Robotic
ESP	Elektronisches Stabilitätsprogramm
FoFa	Folgefahrzeug
FoV	Field of View
FueFa	Führungsfahrzeug
g2o	General Graph Optimization
GDPF	Greedy Dirichlet Process Filter
GIS	Geographic Information System
GNN	Global Nearest Neighbor
GNSS	Global Navigation Satellite System
HD	High Definition
HMM	Hidden Markov Model
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
JPDA	Joint Probability Data Association
KF	Kalman Filter
LiDAR	Light Detection And Ranging
MAE	Mean Absolute Error
MCL	Monte Carlo Lokalisierung
MEMS	Micro-Electro-Mechanical Systems
MIB	Multi-Instance Bernoulli
MRF	Markov-Random-Field
MTT	Multi-Target Tracking
NDT	Normal Distributions Transform
NN	Neuronale Netze
NNS	Nearest Neighbor Search
OGM	Occupancy Grid Maps
OSM	OpenStreetMap
PCL	PointCloudLibrary
PF	Partikel-Filter
PHD	Probability Hypothesis Density

Radar	Radio Detection And Ranging
RANSAC	Random Sample Consensus
RBMCL	Rao-Blackwellized Monte Carlo Lokalisierung
RFS	Random Finite Set
RMSE	Root Mean Squared Error
RTDB	Real-time Database
RTK	Real Time Kinematic
SfM	Structure from Motion
SGM	Semiglobal Matching
SLAM	Simultaneous Localization And Mapping
SMC	Sequential Monte Carlo
SQL	Structured Query Language
TAS	Technik Autonomer Systeme
TCP	Transmission Control Protocol
TOMHT	Tracker-Oriented Multiple Hypothesis Tracking
UAV	Unmaned Aerial Vehicle
UDP	User Datagram Protocol
UKF	Unscented Kalman Filter
UniBw M	Universität der Bundeswehr München
UT	Unscented Transform
UTC	Coordinated Universal Time
UTM	Universal Transverse Mercator
WDF	Wahrscheinlichkeitsdichtefunktion
WGS 84	World Geodetic System 1984
WLAN	Wireless Local Area Network
XML	Extensible Markup Language

Konventionen

Basisnotation

$(.)^i$	Index
$(.)_k$	Zeitschritt
$\boldsymbol{p}$	2D Pose
$\boldsymbol{p}^6$	3D Pose
$\boldsymbol{x}$	Zustandsvektor
x	Skalar

Bayes Filter

$(.)^*$	Praedizierter Zustand
$\boldsymbol{B}$	Eingangsmatrix
$\boldsymbol{K}$	Kalman gain
$\boldsymbol{P}$	Zustandskovarianzmatrix
$\boldsymbol{Q}$	Systemkovarianzmatrix
$\boldsymbol{R}$	Messkovarianzmatrix
$\boldsymbol{U}$	Aktionen

$\boldsymbol{X}$	Filterzustaende
$\boldsymbol{Y}$	Messungen
$\boldsymbol{\Phi}$	Uebergangsmatrix des dynamischen Modells
$\boldsymbol{u}$	Aktion
$\boldsymbol{v}$	Prozessrauschen
$\boldsymbol{w}$	Messrauschen
$\boldsymbol{x}$	Zustand
$\boldsymbol{y}$	Messung
$\boldsymbol{J}_{ij}$	Jacobi-Matrix
$\hat{(.)}$	Zustand nach dem Innovationsschritt
$f(\cdot)$	Nichtlinearen Zustandsgleichungen
$g(\cdot)$	Nichtlinearen Messgleichung

Graph-SLAM

$J^{\mathtt{glm}}$	Bedingung der globalen Landmarkenpositionsschaetzung
$J^{\mathtt{gp}}$	Bedingung der Fahrzeugpositionsschaetzung
$J^{\mathtt{lm}}$	Bedingung der Landmarkenmessungen
$J^{\mathtt{odo}}$	Bedingung der Odometrie
$\boldsymbol{G}$	Graph
$\mathbf{E}$	Graph-Kanten
$\mathbf{V}^*$	Beste Graph-Knoten Konstellation
$\mathbf{Y}^{oi}$	Menge an Objektinstanzen
$\mathbf{Z}$	Menge an Sensor-Landmarken
$\mathbf{v}$	Graph-Knoten
$\mathbf{x}^{utm}$	Globale Fahrzeugposition
$\mathbf{z}^{utm}$	Globale Landmarkenposition

Partikel Filter

$\boldsymbol{X}_K$	Menge an Partikeln
$\boldsymbol{T}$	Trajektorie
$\boldsymbol{Y}_K$	Menge an Landmarkenmessungen
$\boldsymbol{y}$	Sensor-Landmarke
β_{zm}	Landmarkendimensionsabweichungsfaktor
$\hat{\mathbf{P}}$	Zustandskovarianzmatrix im globalen Zustandsraum
$\mathbf{M}$	Pfad-Karte
$\mathbf{P}^{lm}$	Zustandskovarianzmatrix einer Sensor-Landmarke
$\boldsymbol{\Sigma}$	Informationsmatrix
$\hat{\mathbf{x}}$	Beste Positionsschaetzung
$\mathbf{l}$	Karten-Landmarke
$\mathbf{m}$	Pfad-Knoten
$\mathbf{p}^m$	SE2 Pose der Pfadknoten
σ_{zm}	Standard-Kartenunsicherheit

Literaturverzeichnis

Abramov, A., Bayer, C., Heller, C., und Loy, C. (2017). A flexible modeling approach for robust multi-lane road estimation. In *arXiv*.

Adiprawita, W., Ahmad, A. S., Sembiring, J., und Trilaksono, B. R. (2011). A novel resampling method for particle filter for mobile robot localization. *International Journal on Electrical Engineering and Informatics*, 3(2):165--177.

Aeberhard, M. (2017). Object-Level Fusion for Surround Environment Perception in Automated Driving Applications. In *Object-Level Fusion for Surround Environment Perception in Automated Driving Applications*, Seiten 1--201. VDI Verlag GmbH.

Agrawal, M., Konolige, K., und Blas, M. R. (2008). *CenSurE: Center surround extremas for realtime feature detection and matching*, volume 5305 LNCS. Springer Berlin Heidelberg.

Albrecht, C., Kraus, S., und Stilla, U. (2020). Conpect on Landmark Detection in Road Scene Images Taken From a Top-View Camera System. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B1-2020:205--209.

Alismail, H., Baker, L. D., und Browning, B. (2014). Continuous Trajectory estimation for 3D SLAM from actuated lidar. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 6096--6101. IEEE.

Aly, M. (2008). Real time detection of lane markers in urban streets. *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 7--12.

Amini, A., Vaghefi, R. M., De La Garza, J. M., und Buehrer, R. M. (2014). Improving GPS-based vehicle positioning for Intelligent Transportation Systems. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 1023--1029.

Antonio, F. (1992). IV.6 - Faster Line Segment Intersection. In Kirk, D., Herausgeber, *Graphics Gems III (IBM Version)*, Seiten 199--202. Morgan Kaufmann.

Aulinas, J., Petillot, Y., Salvi, J., und Lladó, X. (2008). The SLAM problem: A survey. *Frontiers in Artificial Intelligence and Applications*, 184(1):363--371.

Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., Jesus, L., Berriel, R., Paixão, T., Mutz, F., Veronese, L., Oliveira-Santos, T., und De Souza, A. F. (2019). Self-driving cars: A survey. *arXiv*, abs/1901.0.

Bar Hillel, A., Lerner, R., Levi, D., und Raz, G. (2014). Recent progress in road and lane detection: A survey. *Machine Vision and Applications*, 25(3):727--745.

Bar-Shalom, Y., Li, X.-R., und Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., New York, NY, USA.

Bar-Shalom, Y., Li, X.-R., und Kirubarajan, T. (2002). Estimation for Kinematic Models. In *Estimation for Kinematic Models*, Seiten 267--299. John Wiley & Sons, Inc.

Bardow, P., Davison, A. J., und Leutenegger, S. (2016). Simultaneous optical flow and intensity estimation from an event camera. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seiten 884--892. IEEE.

Barfoot, T. D. (2017). *State estimation for robotics*. Cambridge University Press.

Bar-Shalom, Y., Fortmann, T. E., und Cable, P. G. (1990). *Tracking and Data Association* , volume 87. Academic Press Professional, Inc., San Diego, CA, USA.

Bauer, S., Alkhorshid, Y., und Wanielik, G. (2016). Using high-definition maps for precise urban vehicle localization. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Seiten 492--497.

Bay, H., Tuytelaars, T., und Van Gool, L. (2006). SURF: Speeded up robust features. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 3951 LNCS, Seiten 404--417.

Bayrisches Vermessungsamt (2020). Bayrisches Vermessungsamt. `https://www.ldbv.bayern.de/`. Accessed: 2020-09-30.

Behley, J., Garbade, M., Milioto, A., Behnke, S., Stachniss, C., Gall, J., und Quenzel, J. (2019). SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences. In *arXiv*, Seiten 9297--9307.

Behringer, R., Holt, V. V., und Dickmanns, D. (1992). Road and relative ego-state recognition. *IEEE Intelligent Vehicles Symposium, Proceedings*, 14(2):385--390.

Bender, P., Ziegler, J., und Stiller, C. (2014). Lanelets: Efficient map representation for autonomous driving. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 420--425.

Besl, P. J. und McKay, N. D. (1992). A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239--256.

Bibby, C. und Reid, I. (2010). A hybrid SLAM representation for dynamic marine environments. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 257--264. IEEE.

Biber, P. (2003). The Normal Distributions Transform: A New Approach to Laser Scan Matching. *IEEE International Conference on Intelligent Robots and Systems*, 3(October):2743--2748.

Biswas, J. und Veloso, M. (2010). WiFi localization and navigation for autonomous indoor mobile robots. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 4379--4384.

Blanco, J. L., Fernndez-Madrigal, J. A., und Gonzalez, J. (2008). A novel measure of uncertainty for mobile robot SLAM with rao-blackwellized particle filters. *International Journal of Robotics Research*, 27(1):73--89.

Blochliger, F., Fehr, M., Dymczyk, M., Schneider, T., und Siegwart, R. (2018). Topomap: Topological Mapping and Navigation Based on Visual SLAM Maps. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 3818--3825.

Boerner, R., Xu, Y., Baran, R., Steinbacher, F., Hoegner, L., und Stilla, U. (2019). Registration of multi-sensor bathymetric point clouds in rural areas using point-to-grid distances. *ISPRS International Journal of Geo-Information*, 8(4).

Bogoslavskyi, I. und Stachniss, C. (2016). Fast range image-based segmentation of sparse 3D laser scans for online operation. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2016-Novem, Seiten 163--169.

Bosse, M. und Zlot, R. (2009). Continuous 3D scan-matching with a spinning 2D laser. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 4312--4319. IEEE.

Bouguettaya, A., Yu, Q., Liu, X., Zhou, X., und Song, A. (2015). Efficient agglomerative hierarchical clustering. *Expert Systems with Applications*, 42(5):2785--2797.

Bresenham, J. (1965). Algorithm for Computer Control of a Digital Plotter. *IBM Syst. J.*, 4:25--30.

Bronstejn, I. N. (2012). *Taschenbuch der Mathematik*. Harri Deutsch, Frankfurt am Main.

Brubaker, M. A., Geiger, A., und Urtasun, R. (2016). Map-based probabilistic visual self-localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(4):652--665.

Burger, P., Naujoks, B., und Wuensche, H. J. (2018). Fast Dual Decomposition based Mesh-Graph Clustering for Point Clouds. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Seiten 1129--1135.

Burger, P., Naujoks, B., und Wuensche, H. J. (2019a). Map-Aware SLAM with Sparse Map Features. *IEEE International Conference on Intelligent Robots and Systems*, Seiten 347--353.

Burger, P., Naujoks, B., und Wuensche, H. J. (2019b). Unstructured Road SLAM using Map Predictive Road Tracking. *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, Seiten 1276--1282.

Burger, P. und Wuensche, H. J. (2018). Fast Multi-Pass 3D Point Segmentation Based on a Structured Mesh Graph for Ground Vehicles. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 2150--2156.

Cadena, C., Dick, A., und Reid, I. D. (2015). A fast, modular scene understanding system using context-aware object detection. In *Proceedings - IEEE International Conference on Robotics and Automation*, number June in Proceedings ICRA, Seiten 4859--4866. IEEE.

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., und Beijbom, O. (2020). Nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seiten 11618--11628.

Caltagirone, L., Bellone, M., Svensson, L., und Wahde, M. (2018). LIDAR-camera fusion for road detection using fully convolutional neural networks. *arXiv*, 111:125--131.

Caltagirone, L., Scheidegger, S., Svensson, L., und Wahde, M. (2017). Fast LIDAR-based road detection using fully convolutional neural networks. In *arXiv*.

Canavosio-Zuzelski, R., Agouris, P., und Doucette, P. (2013). A Photogrammetric approach for assessing positional accuracy of OpenStreetMap© roads. *ISPRS International Journal of Geo-Information*, 2(2):276--301.

Carlone, L., Aragues, R., Castellanos, J. A., und Bona, B. (2014a). A fast and accurate approximation for planar pose graph optimization. *International Journal of Robotics Research*, 33(7):965--987.

Carlone, L., Calafiore, G. C., Tommolillo, C., und Dellaert, F. (2016). Planar Pose Graph Optimization: Duality, Optimal Solutions, and Verification. *IEEE Transactions on Robotics*, 32(3):545--565.

Carlone, L., Du, J., Kaouk Ng, M., Bona, B., und Indri, M. (2010). An application of Kullback-Leibler divergence to active SLAM and exploration with particle filters. In *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, Seiten 287--293. IEEE.

Carlone, L., Du, J., Kaouk Ng, M., Bona, B., und Indri, M. (2014b). Active SLAM and exploration with particle filters using Kullback-Leibler divergence. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 75(2):291--311.

Censi, A. (2007). An accurate closed-form estimate of ICP'S covariance. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 3167--3172.

Censi, A. und Scaramuzza, D. (2014). Low-Latency Event-Based Visual Odometry. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 703--710. IEEE.

Chang, L., Hu, B., Li, A., und Qin, F. (2013). Transformed unscented kalman filter. *IEEE Transactions on Automatic Control*, 58(1):252--257.

Chen, S. Y. (2012). Kalman filter for robot vision: A survey. *IEEE Transactions on Industrial Electronics*, 59(11):4409--4420.

Chen, T., Dai, B., Wang, R., und Liu, D. (2014). Gaussian-Process-Based Real-Time Ground Segmentation for Autonomous Land Vehicles. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 76(3-4):563--582.

Chen, W., Liu, Q., Hu, H., Liu, J., Wang, S., und Zhu, Q. (2020). Novel laser-based obstacle detection for autonomous robots on unstructured terrain. *Sensors (Switzerland)*, 20(18):1--18.

Chen, Z., Zhang, J., und Tao, D. (2019). Progressive LiDAR adaptation for road detection. *arXiv*.

Cheng, J., Xiang, Z., Cao, T., und Liu, J. (2014). Robust vehicle detection using 3D Lidar under complex urban environment. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 691--696. IEEE.

Choudhary, S., Indelman, V., Christensen, H. I., und Dellaert, F. (2015). Information-based reduced landmark SLAM. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2015-June, Seiten 4620--4627.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., und Schiele, B. (2016). The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seiten 3213--3223.

Cummins, M. und Newman, P. (2008). FAB-MAP: Probabilistic localization and mapping in the space of appearance. *International Journal of Robotics Research*, 27(6):647--665.

Danescu, R., Oniga, F., und Nedevschi, S. (2010). Particle grid tracking system for stereovision based environment perception. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 987--992.

Danescu, R., Oniga, F., und Nedevschi, S. (2011). Modeling and tracking the driving environment with a particle-based occupancy grid. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1331--1342.

Dean, T. L. und Kanazawa, K. (1988). Probabilistic Temporal Reasoning. In *Aaai*, AAAI'88, Seiten 524--529. AAAI Press.

Dellaert, F., Fox, D., Burgard, W., und Thrun, S. (1999). Monte Carlo localization for mobile robots. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2, Seiten 1322--1328.

Doherty, K., Wang, J., und Englot, B. (2016). Probabilistic map fusion for fast, incremental occupancy mapping with 3D Hilbert maps. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2016-June, Seiten 1011--1018.

Douillard, B., Underwood, J., Melkumyan, N., Singh, S., Vasudevan, S., Brunner, C., und Quadros, A. (2010). Hybrid elevation maps: 3D surface models for segmentation. In *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, Seiten 1532--1538.

Engel, J., Schöps, T., und Cremers, D. (2014). LSD-SLAM: Large-Scale Direct monocular SLAM. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, number PART 2 in Lecture Notes in Computer Science, Seiten 834--849. Springer.

ESRI (2020). Environmental Systems Research Institute. `https://www.esri.com/`. Accessed: 2020-09-30.

Fairfield, N., Kantor, G., und Wettergreen, D. (2007). Real-time SLAM with octree evidence grids for exploration in underwater tunnels. *Journal of Field Robotics*, 24(1-2):03--21.

Fassbender, D., Heinrich, B. C., und Wuensche, H. J. (2016). Motion planning for autonomous vehicles in highly constrained urban environments. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 4708--4713.

Fassbender, D., Kusenbach, M., und Wuensche, H. J. (2015). Landmark-based navigation in large-scale outdoor environments. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 4445--4450.

Fassbender, D., Mueller, A., und Wuensche, H. J. (2014). Trajectory planning for car-like robots in unknown, unstructured environments. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 3630--3635, Chicago, IL, USA.

Ferris, B., Fox, D., und Lawrence, N. (2007). WiFi-SLAM using Gaussian process latent variable models. In *IJCAI International Joint Conference on Artificial Intelligence*, Seiten 2480--2485.

Fisher, R. B. (1990). Geometric constraints from planar surface patch matching. *Image and Vision Computing*, 8(2):148--154.

Floros, G., Van Der Zander, B., und Leibe, B. (2013). OpenStreetSLAM: Global vehicle localization using OpenStreetMaps. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 1054--1059.

Forster, C., Carlone, L., Dellaert, F., und Scaramuzza, D. (2017a). On-Manifold Preintegration for Real-Time Visual-Inertial Odometry. In *IEEE Transactions on Robotics*, volume 33, Seiten 1--21.

Forster, C., Pizzoli, M., und Scaramuzza, D. (2014). SVO: Fast Semi-Direct Monocular Visual Odometry. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 15--22. IEEE.

Forster, C., Zhang, Z., Gassner, M., Werlberger, M., und Scaramuzza, D. (2017b). SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems. In *IEEE Transactions on Robotics*, volume 33, Seiten 249--265.

Fox, D., Burgard, W., Dellaert, F., und Thrun, S. (1999). Monte Carlo Localization: efficient position estimation for mobile robots. In *Proceedings of the National Conference on Artificial Intelligence*, Seiten 343--349.

Fraundorfer, F., Engels, C., und Nistér, D. (2007). Topological mapping, localization and navigation using image collections. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 3872--3877.

Fraundorfer, F. und Scaramuzza, D. (2012). Visual Odometry. Part II: Matching, Robustness, Optimization, and Applications. *IEEE Robotics and Automation Magazine*, 19(2):78--90.

Frese, U., Larsson, P., und Duckett, T. (2005). A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Transactions on Robotics*, 21(2):196--207.

Fries, C., Burger, P., Kallwies, J., Naujoks, B., Luettel, T., und Wuensche, H. J. (2018). How MuCAR won the convoy scenario at ELROB 2016. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Seiten 1--7.

Fröhlich, L. (2018). *PostGIS*. Carl Hanser Verlag GmbH und Co. KG.

Gálvez-López, D. und Tardós, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188--1197.

Gao, X., Wang, R., Demmel, N., und Cremers, D. (2018). LDSO: Direct sparse odometry with loop closure. In *arXiv*.

Geiger, A., Lenz, P., Stiller, C., und Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research*, 32(11):1231--1237.

Geyer, J., Kassahun, Y., Mahmudi, M., Ricou, X., Durgesh, R., Chung, A. S., Hauswald, L., Pham, V. H., Mühlegg, M., Dorn, S., Fernandez, T., Jänicke, M., Mirashi, S., Savani, C., Sturm, M., Vorobiov, O., Oelker, M., Garreis, S., und Schuberth, P. (2020). A2d2: Audi autonomous driving dataset. *arXiv*.

Girshick, R., Donahue, J., Darrell, T., und Malik, J. (2016). Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):142--158.

Goebl, M. (2009). *(German) Eine realzeitfähige Architektur zur Integration kognitiver Funktionen*. Dissertation, Technische Universität München, München.

Goebl, M. und Färber, G. (2007). A real-time-capable hard- and software architecture for joint image and knowledge processing in cognitive automobiles. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 734--740, Istanbul, Turkey.

Goodchild, M. F. (2007). Citizens as sensors: The world of volunteered geography.

Gordon, N. J., Salmond, D. J., und Smith, A. F. (1993). Novel approach to nonlinear/non-gaussian Bayesian state estimation. *IEE Proceedings, Part F: Radar and Signal Processing*, 140(2):107--113.

Gran, C. W. (2019). HD-Maps in Autonomous Driving. *IATSS Research*, 43(1):1--13.

Guivant, J. E. und Nebot, E. M. (2001). Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, 17(3):242--257.

Haklay, M. (2010). How good is volunteered geographical information? A comparative study of OpenStreetMap and ordnance survey datasets. *Environment and Planning B: Planning and Design*, 37(4):682--703.

Hellerstein, J. M., Naughton, J. F., und Pfeffer, A. (1995). Generalized Search Trees for Database Systems. In *Proceedings of the 21st International Conference of Very Large Databases VLDB*, Seiten 562--573.

Henkel, P. und Sperl, A. (2016). Precise RTK positioning with GPS/INS tight coupling and multipath estimation. In *Institute of Navigation International Technical Meeting 2016, ITM 2016*, volume 2, Seiten 1015--1023.

Himmelsbach, M., Hundelshausen, F. v., und Wuensche, H. J. (2010). Fast segmentation of 3D point clouds for ground vehicles. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 560--565, San Diego, CA, USA.

Himmelsbach, M., Luettel, T., und Wuensche, H. J. (2009). Real-time object classification in 3D point clouds using point feature histograms. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, Seiten 994--1000, St. Louis, MO, USA.

Hirschmüller, H. (2011). Semi-Global Matching Motivation, Developments and Applications. *Photogrammetric Week*.

Hoegner, L., Tuttas, S., und Stilla, U. (2016). 3D building reconstruction and construction site monitoring from RGB and TIR image sets. In *2016 12th International Symposium on Electronics and Telecommunications, ISETC 2016 - Conference Proceedings*, Seiten 305--308.

Holz, D. und Behnke, S. (2010). Sancta Simplicitas - On the efficiency and achievable results of SLAM using ICP-based incremental registration. *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 1380--1387.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., und Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3):189--206.

Hu, G., Khosoussi, K., und Huang, S. (2013). Towards a reliable SLAM back-end. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 37--43.

Huang, H., Brenner, C., und Sester, M. (2011). 3D building roof reconstruction from point clouds via generative models. In *GIS: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, Seiten 16--24, Chicago, IL, USA. ACM Press.

Huang, H., Brenner, C., und Sester, M. (2013). A generative statistical approach to automatic 3D building roof reconstruction from laser scanning data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 79:29--43.

Huang, H., Burger, P., Schmitz, M., Roth, L., Wünsche, H. J., und Mayer, H. (2018). Driving in unknown areas: From UAV images to map for autonomous vehicles. In *IWCTS 2018 - Proceedings of the 11th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, Seiten 39--42.

Huang, H. und Mayer, H. (2015). Robust and efficient urban scene classification using relative features. In *GIS: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, volume 03-06-Nove of *GIS '15*, Seiten 81:1----81:4, New York, NY, USA. ACM.

Huang, H. und Mayer, H. (2017). Towards automatic large-scale 3D building reconstruction: Primitive decomposition and assembly. In *Lecture Notes in Geoinformation and Cartography*, Lecture Notes in Geoinformation and Cartography, Seiten 205--221. Springer.

Huang, X., Wang, P., Cheng, X., Zhou, D., Geng, Q., und Yang, R. (2020). The ApolloScape Open Dataset for Autonomous Driving and Its Application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10):2702--2719.

Ikawa, G., Watanabe, Y., Yamada, S., und Takada, H. (2019). Performance Evaluation of Querying Point Clouds in RDBMS. In *2019 IEEE International Conference on Big Data and Smart Computing, BigComp 2019 - Proceedings*.

Ilci, V. und Toth, C. (2020). High definition 3D map creation using GNSS/IMU/LiDAR sensor integration to support autonomous vehicle navigation. *Sensors (Switzerland)*, 20(3).

Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8):651--666.

Jang, W., An, J., Lee, S., Cho, M., Sun, M., und Kim, E. (2018). Road Lane Semantic Segmentation for High Definition Map. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 1001--1006.

Jaspers, H. (2021). *Towards Autonomous Driving with Visual Landmarks -- Camera-based Mapping and Localization in Unknown Terrain*. Dissertation, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik, Neubiberg.

Jaspers, H., Fassbender, D., und Wuensche, H. J. (2017a). Visual navigation with efficient ConvNet features. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 5340--5345, Vancouver, BC, Canada.

Jaspers, H., Himmelsbach, M., und Wuensche, H. J. (2017b). Multi-modal local terrain maps from vision and LiDAR. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 1119--1125, USA.

Julier, S. J. (2002). The scaled unscented transformation. In *Proceedings of the American Control Conference*, volume 6, Seiten 4555--4559, Anchorage, AK, USA.

Julier, S. J. und Uhlmann, J. K. (1997). New extension of the Kalman filter to nonlinear systems. In Kadar, I., Herausgeber, *Signal Processing, Sensor Fusion, and Target Recognition VI*, volume 3068, Seite 182. International Society for Optics and Photonics, SPIE.

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J. J., und Dellaert, F. (2012). ISAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research*, 31(2):216--235.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering, Transactions of the ASME*, 82(1):35--45.

Kammel, S. und Pitzer, B. (2008). Lidar-based lane marker detection and mapping. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 1137--1142.

Kammel, S., Ziegler, J., Pitzer, B., Werling, M., Gindele, T., Jagzent, D., Schöder, J., Thuy, M., Goebl, M., Von Hundelshausen, F., Pink, O., Frese, C., und Stiller, C. (2009). Team AnnieWAY's autonomous system for the DARPA Urban Challenge 2007. *Springer Tracts in Advanced Robotics*, 56:359--391.

Kerl, C., Sturm, J., und Cremers, D. (2013). Dense visual SLAM for RGB-D cameras. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 2100--2106.

Kim, A. und Eustice, R. M. (2013). Real-time visual SLAM for autonomous underwater hull inspection using visual saliency. *IEEE Transactions on Robotics*, 29(3):719--733.

Kim, H., Leutenegger, S., und Davison, A. J. (2016a). Real-time 3D reconstruction and 6-DoF tracking with an event camera. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9910 LNCS, Seiten 349--364.

Kim, J. H., Cadena, C., und Reid, I. (2016b). Direct semi-dense SLAM for rolling shutter cameras. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2016-June, Seiten 1308--1315. IEEE.

Klasing, K., Wollherr, D., und Buss, M. (2009). Realtime segmentation of range data using continuous nearest neighbors. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 2431--2436.

Krajnik, T., Fentanes, J. P., Mozos, O. M., Duckett, T., Ekekrantz, J., und Hanheide, M. (2014). Long-term topological localisation for service robots in dynamic environments using spectral maps. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 4537--4542. IEEE.

Kuang Chiu, H., Prioletti, A., Li, J., Bohg, J., University, S., und Institute, T. R. (2020). Probabilistic 3D multi-object tracking for autonomous driving. *arXiv*.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., und Burgard, W. (2011). G2o: A general framework for graph optimization. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 3607--3613.

Kümmerle, R., Steder, B., Dornhege, C., Kleiner, A., Grisetti, G., und Burgard, W. (2010). Large scale graph-based SLAM using aerial images as prior information. *Robotics: Science and Systems*, 5(1):297--304.

Lai, K. und Fox, D. (2010). Object recognition in 3D point clouds using web data and domain adaptation. *International Journal of Robotics Research*, 29(8):1019--1037.

Lejeune, A., Verly, J. G., und Van Droogenbroeck, M. (2018). Probabilistic Framework for the Characterization of Surfaces and Edges in Range Images, with Application to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(9):2209--2222.

Lemaire, T., Berger, C., Jung, I. K., und Lacroix, S. (2007). Vision-based SLAM: Stereo and monocular approaches. *International Journal of Computer Vision*, 74(3):343--364.

Levinson, J., Montemerlo, M., und Thrun, S. (2008). Map-based precision vehicle localization in urban environments. In *Robotics: Science and Systems*, volume 3, Seiten 121--128, Atlanta, GA, USA.

Li, M., Li, W., Wang, J., Li, Q., und Nüchter, A. (2012). Dynamic VeloSLAM – Preliminary Report on 3D Mapping of Dynamic Environments. *IEEE Intelligent Vehicles*, Seiten 1--6.

Lloyd, S. P. (1982). Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129--137.

Loeliger, H. A. (2004). An introduction to factor graphs. *IEEE Signal Processing Magazine*, 21(1):28--41.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, Seiten 1150--1157.

Lu, F. und Milios, E. (1997a). Globally Consistent Range Scan Alignment for Environment Mapping. *Autonomous Robots*, 4(4):333--349.

Lu, F. und Milios, E. (1997b). Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 18(3):249--275.

Lu, W., Rodriguez Florez, S. A., Seignez, E., und Reynaud, R. (2013). An Improved Approach for Vision-Based Lane Marking Detection and Tracking. *2013 International Conference on Electrical, Control and Automation Engineering*, Seiten 382--386.

Maddern, W., Pascoe, G., Linegar, C., und Newman, P. (2017). 1 year, 1000 km: The Oxford RobotCar dataset. *International Journal of Robotics Research*, 36(1):3--15.

Maguire, D. J. (1991). An overview and definition of GIS. *Geographical information systems. Vol. 1: principles*, Seiten 9--20.

Makarenko, A. A., Williams, S. B., Bourgault, F., und Durrant-Whyte, H. F. (2002). An experiment in integrated exploration. In *IEEE International Conference on Intelligent Robots and Systems*, volume 1, Seiten 534--539.

Mancini, F., Dubbini, M., Gattelli, M., Stecchi, F., Fabbri, S., und Gabbianelli, G. (2013). Using unmanned aerial vehicles (UAV) for high-resolution reconstruction of topography: The structure from motion approach on coastal environments. *Remote Sensing*, 5(12):6880--6898.

Manz, M., Himmelsbach, M., Luettel, T., und Wuensche, H. J. (2011). Detection and tracking of road networks in rural terrain by fusing vision and LIDAR. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 4562--4568.

Manz, M., Von Hundelshausen, F., und Wuensche, H. J. (2010). A hybrid estimation approach for autonomous dirt road following using multiple clothoid segments. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 2410--2415, Anchorage, AK, USA.

Marchetti, L., Grisetti, G., und Iocchi, L. (2007). A comparative analysis of particle filter based localization methods. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 4434 LNAI, Seiten 442--449.

Martinelli, A., Tomatis, N., und Siegwart, R. (2007). Simultaneous localization and odometry self calibration for mobile robot. *Autonomous Robots*, 22(1):75--85.

Matthaei, R., Bagschik, G., und Maurer, M. (2014). Map-relative localization in lane-level maps for ADAS and autonomous driving. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 49--55.

Matthaei, R., Reschka, A., Rieken, J., Dierkes, F., Ulbrich, S., Winkle, T., und Maurer, M. (2015). Autonomous driving. In *Handbook of Driver Assistance Systems: Basic Information, Components and Systems for Active Safety and Comfort*, Seiten 1519--1556. Springer Berlin Heidelberg.

Maxar (2020). Maxar Technologies. `https://www.maxar.com/`. Accessed: 2020-09-30.

Mayer, H., Bartelsen, J., Hirschmüller, H., und Kuhn, A. (2012). Dense 3D reconstruction from wide baseline image sets. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7474 LNCS, Seiten 285--304, Berlin, Heidelberg. Springer Berlin Heidelberg.

Meis, U., Klein, W., und Wiedemann, C. (2010). A new method for robust far-distance road course estimation in advanced driver assistance systems. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, Seiten 1357--1362.

Mekik, C. und Can, O. (2010). Multipath effects in RTK GPS and a case study. *Journal of Aeronautics, Astronautics and Aviation*, 42(4):231--240.

Microsoft (2020). Microsoft Bing Maps. `https://https://www.bing.com/maps`. Accessed: 2020-09-30.

Milford, M., Lowry, S., Sunderhauf, N., Shirazi, S., Pepperell, E., Upcroft, B., Shen, C., Lin, G., Liu, F., Cadena, C., und Reid, I. (2015). Sequence searching with deep-learnt depth for condition-and viewpoint-invariant route-based place recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, volume 2015-Octob, Seiten 18--25.

Milford, M. J. und Wyeth, G. F. (2012). SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 1643--1649. IEEE.

Milioto, A., Vizzo, I., Behley, J., und Stachniss, C. (2019). RangeNet ++: Fast and Accurate LiDAR Semantic Segmentation. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 4213--4220.

Montemerlo, M., Becker, J., Shat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Hoffmann, G., Huhnke, B., Johnston, D., Klumpp, S., Langer, D., Levandowski, A., Levinson, J., Marcil, J., Orenstein, D., Paefgen, J., Penny, I., Petrovskaya, A., Pflueger, M., Stanek, G., Stavens, D., Vogt, A., und Thrun, S. (2008). Junior: The Stanford entry in the urban challenge. *Journal of Field Robotics*, 25(9):569--597.

Montemerlo, M. und Thrun, S. (2003). Simultaneous localization and mapping with unknown data association using FastSLAM. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2, Seiten 1985--1991.

Montemerlo, M. und Thrun, S. (2006). Large-scale robotic 3-D mapping of urban structures. In *Springer Tracts in Advanced Robotics*, volume 21, Seiten 141--150. Springer.

Montemerlo, M., Thrun, S., Koller, D., und Wegbreit, B. (2002). FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the National Conference on Artificial Intelligence*, Seiten 593--598.

Montemerlo, M., Thrun, S., Roller, D., und Wegbreit, B. (2003). FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *IJCAI International Joint Conference on Artificial Intelligence*, Seiten 1151--1156. Morgan Kaufmann Publishers Inc.

Mooney, P. und Corcoran, P. (2012). The Annotation Process in OpenStreetMap. *Transactions in GIS*, 16(4):561--579.

Moore, E. F. (1959). The shortest path through a maze. In *Int. Symposium on the Theory of Switching, Part II*, Seiten 285--292.

Moosmann, F., Pink, O., und Stiller, C. (2009). Segmentation of 3D lidar data in non-flat urban environments using a local convexity criterion. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 215--220.

Moosmann, F. und Stiller, C. (2011). Velodyne SLAM. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 393--398.

Moravec, H. P. und Elfes, A. (1985). High resolution maps from wide angle sonar. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2, Seiten 116--121.

Mueggler, E., Gallego, G., und Scaramuzza, D. (2015). Continuous-time trajectory estimation for event-based vision sensors. In *Robotics: Science and Systems*, volume 11.

Mueller, G. R., Burger, P., und Wuensche, H. J. (2018). Continuous Stereo Self-Calibration on Planar Roads. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 1755--1760.

Mur-Artal, R., Montiel, J. M., und Tardos, J. D. (2015). ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, 31(5):1147--1163.

Murphy, K. und Russell, S. (2001). Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. In *Sequential Monte Carlo Methods in Practice*. Springer.

Murphy, R. R. (1998). Dempster-Shafer theory for sensor fusion in autonomous mobile robots. *IEEE Transactions on Robotics and Automation*, 14(2):197--206.

Naujoks, B., Burger, P., und Wuensche, H.-J. (2019a). Combining Deep Learning and Model-Based Methods for Robust Real-Time Semantic Landmark Detection. In *22nd International Conference on Information Fusion (FUSION)*, Ottawa, ON, Canada.

Naujoks, B., Burger, P., und Wuensche, H.-J. (2019b). Fast 3D Extended Target Tracking using NURBS Surfaces. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, Auckland, New Zealand. ©IEEE.

Naujoks, B., Burger, P., und Wuensche, H. J. (2019c). The greedy dirichlet process filter-An online clustering multi-target tracker. In *2018 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2018 - Proceedings*, Seiten 1233--1237.

Naujoks, B. und Wuensche, H. J. (2018). An Orientation Corrected Bounding Box Fit Based on the Convex Hull under Real Time Constraints. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 415--420, Changshu, China.

Negre, A., Rummelhard, L., und Laugier, C. (2014). Hybrid sampling Bayesian Occupancy Filter. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 1307--1312.

Neuhaus, F., Dillenberger, D., Pellenz, J., und Paulus, D. (2009). Terrain drivability analysis in 3D laser range data for autonomous robot navigation in unstructured environments. In *ETFA 2009 - 2009 IEEE Conference on Emerging Technologies and Factory Automation*.

Nex, F. und Remondino, F. (2014). UAV for 3D mapping applications: A review.

Nuss, D., Reuter, S., Thom, M., Yuan, T., Krehl, G., Maile, M., Gern, A., und Dietmayer, K. (2018). A random finite set approach for dynamic occupancy grid maps with real-time application. *International Journal of Robotics Research*, 37(8):841--866.

Nuss, D., Yuan, T., Krehl, G., Stuebler, M., Reuter, S., und Dietmayer, K. (2015). Fusion of laser and radar sensor data with a sequential Monte Carlo Bayesian occupancy filter. In *IEEE Intelligent Vehicles Symposium, Proceedings*, volume 2015-Augus, Seiten 1074--1081.

Ort, T., Paull, L., und Rus, D. (2018). Autonomous Vehicle Navigation in Rural Environments Without Detailed Prior Maps. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 2040--2047.

Parker, R. und Valaee, S. (2006). Vehicle localization in vehicular networks. In *IEEE Vehicular Technology Conference*, Seiten 2713--2717.

Parois, P. und Lutz, M. (2011). Linear transformations of variance/covariance matrices. *Acta Crystallographica Section A: Foundations of Crystallography*, 67(4):383--390.

Pathak, K., Vaskevicius, N., Poppinga, J., Pfingsthorn, M., Schwertfeger, S., und Birk, A. (2009). Fast 3D mapping by matching planes extracted from range sensor point-clouds. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, Seiten 1150--1155.

Patron-Perez, A., Lovegrove, S., und Sibley, G. (2015). A Spline-Based Trajectory Representation for Sensor Fusion and Rolling Shutter Cameras. *International Journal of Computer Vision*, 113(3):208--219.

Petrovskaya, A. und Thrun, S. (2009). Model based vehicle detection and tracking for autonomous urban driving. *Autonomous Robots*, 26(2-3):123--139.

Pham, Q. H., Sevestre, P., Pahwa, R. S., Zhan, H., Pang, C. H., Chen, Y., Mustafa, A., Chandrasekhar, V., und Lin, J. (2020). A3D Dataset: Towards Autonomous Driving in Challenging Environments. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 2267--2273.

Pillai, S. und Leonard, J. J. (2015). Monocular SLAM supported object recognition. In *Robotics: Science and Systems*, volume 11.

Poggenhans, F., Salscheider, N. O., und Stiller, C. (2018). Precise Localization in High-Definition Road Maps for Urban Regions. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 2167--2174.

Pomerleau, F., Colas, F., Siegwart, R., und Magnenat, S. (2013). Comparing ICP variants on real-world data sets: Open-source library and experimental protocol. *Autonomous Robots*, 34(3):133--148.

Qi, C. R., Su, H., Mo, K., und Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, Seiten 77--85.

Redmon, J. und Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, Seiten 6517--6525.

Ren, S., He, K., Girshick, R., und Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., und Garnett, R., Herausgeber, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 39, Seiten 1137--1149. Curran Associates, Inc.

Rieken, J., Matthaei, R., und Maurer, M. (2015). Benefits of using explicit ground-plane information for grid-based urban environment modeling. In *2015 18th International Conference on Information Fusion, Fusion 2015*, Seiten 2049--2056.

Romera, E., Alvarez, J. M., Bergasa, L. M., und Arroyo, R. (2018). ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):263--272.

Rormero, A. R., Borges, P. V., Pfrunder, A., und Elfes, A. (2018). Map-Aware Particle Filter for Localization. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 2940--2947.

Rosten, E. und Drummond, T. (2005). Fusing points and lines for high performance tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, volume II, Seiten 1508--1515.

Roumeliotis, S. I. und Bekey, G. A. (2000). Bayesian estimation and Kalman filtering: a unified framework for mobile robot localization. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 3, Seiten 2985--2992.

Roweis, S. (1996). Levenberg-Marquardt Optimization. *Lecture Notes. University Of Toronto*.

Ruchti, P., Steder, B., Ruhnke, M., und Burgard, W. (2015). Localization on OpenStreetMap data using a 3D laser scanner. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2015-June, Seiten 5260--5265.

Salas-Moreno, R. F., Newcombe, R. A., Strasdat, H., Kelly, P. H., und Davison, A. J. (2013). SLAM++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seiten 1352--1359. IEEE.

Särkkä, S. (2010). *Bayesian filtering and smoothing*, volume 3. Cambridge University Press.

Schindler, A. (2013). Vehicle self-localization with high-precision digital maps. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 134--139.

Schiotka, A., Suger, B., und Burgard, W. (2017). Robot localization with sparse scan-based maps. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2017-Septe, Seiten 642--647.

Schneider, P. J. und Eberly, D. H. (2003). *Geometric Tools for Computer Graphics*. Elsevier Science Publishers B. V.

Schneider, S., Himmelsbach, M., Luettel, T., und Wuensche, H. J. (2010). Fusing vision and LIDAR - Synchronization, correction and occlusion reasoning. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 388--393, San Diego, CA, USA.

Schreiber, M., Poggenhans, F., und Stiller, C. (2014). Detecting symbols on road surface for mapping and localization using OCR. In *2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC 2014*, Seiten 597--602.

Schubert, R., Richter, E., und Wanielik, G. (2008). Comparison and evaluation of advanced motion models for vehicle tracking. In *Proceedings of the 11th International Conference on Information Fusion, FUSION 2008*, Seiten 1--6.

Scioscia, F., Binetti, M., Ruta, M., Ieva, S., und Di Sciascio, E. (2014). A Framework and a Tool for Semantic Annotation of POIs in OpenStreetMap. *Procedia - Social and Behavioral Sciences*, 111:1092--1101.

Se, S., Lowe, D., und Little, J. (2001). Vision-based mobile robot localization and mapping using scale-invariant features. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2, Seiten 2051--2058.

Segal, A. V., Haehnel, D., und Thrun, S. (2010). Generalized-ICP. In *Robotics: Science and Systems*, volume 5, Seiten 161--168.

Seif, H. G. und Hu, X. (2016). Autonomous Driving in the iCity—HD Maps as a Key Challenge of the Automotive Industry. *Engineering*, 2(2):159--162.

Sengupta, S. und Sturgess, P. (2015). Semantic octree: Unifying recognition, reconstruction and representation via an octree constrained higher order MRF. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2015-June, Seiten 1874--1879. IEEE.

Senlet, T. und Elgammal, A. (2011). A framework for global vehicle localization using stereo images and satellite and road maps. *Proceedings of the IEEE International Conference on Computer Vision*, Seiten 2034--2041.

Senne, K. (1972). *Stochastic processes and filtering theory*, volume 17 of *Mathematics in Science and Engineering*. Acad. Press.

Shalizi, C. (2009). Distances between Clustering , Hierarchical Clustering. *Data Mining*, (September):36--350.

Shimkin, N. (2009). Kinematic Models for Target Tracking. *Estimation and identification in dynamical Systems*, Seiten 1--9.

Smith, R., Self, M., und Cheeseman, P. (1988). Estimating Uncertain Spatial Relationships in Robotics. In *Machine Intelligence and Pattern Recognition*, volume 5, Seiten 435--461.

Sola, J. (2013). Simulataneous localization and mapping with the extended Kalman filter. *unpublished. Available: http://www. joansola. eu/JoanSola/eng/JoanSola. html*, Seiten 1--35.

Solarz, A. und Szymczyk, T. (2020). Oracle 19c, SQL Server 2019, Postgresql 12 and MySQL 8 database systems comparison. *Journal of Computer Sciences Institute*, 17:373--378.

Stachniss, C., Grisetti, G., und Burgard, W. (2005). Information gain-based exploration using rao-blackwellized particle filters. In *Robotics: Science and Systems*, volume 1, Seiten 65--72.

Stachniss, C., Leonard, J. J., und Thrun, S. (2016). Simultaneous localization and mapping. In Siciliano, B. und Khatib, O., Herausgeber, *Springer Handbook of Robotics*, chapter 37, Seiten 1153--1175. Springer.

Steyer, S., Tanzmeister, G., und Wollherr, D. (2018). Grid-based environment estimation using evidential mapping and particle tracking. *IEEE Transactions on Intelligent Vehicles*, 3(3):384--396.

Suger, B. und Burgard, W. (2017). Global outer-urban navigation with OpenStreetMap. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 1417--1422.

Sünderhauf, N. (2012). *Robust Optimization for Simultaneous Localization and Mapping*. PhD thesis, Technische Universität Chemnitz.

Sünderhauf, N. und Protzel, P. (2012). Towards a robust back-end for pose graph SLAM. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 1254--1261. IEEE.

Sünderhauf, N., Shirazi, S., Jacobson, A., Dayoub, F., Pepperell, E., Upcroft, B., und Milford, M. (2015). Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free. In *Robotics: Science and Systems*, volume 11, Rome, Italy.

Takeuchi, E. und Tsubouchi, T. (2006). A Fast Scan Matching in 3-D Space using 3D Normal Distributions Transform for Mobile Robotic Mapping. *Transform*, Seiten 2--7.

Tanzmeister, G., Thomas, J., Wollherr, D., und Buss, M. (2014). Grid-based mapping and Tracking in dynamic environments using a uniform evidential environment representation. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 6090--6095. IEEE.

The PostgreSQL Global Development Group (2021). https://www.postgresql.org/. URL: https://www.postgresql.org/.

Thrun, S. (2001). Learning occupancy grids with forward models. In *IEEE International Conference on Intelligent Robots and Systems*, volume 3, Seiten 1676--1681.

Thrun, S. (2002). Robotic Mapping: A Survey. In *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann.

Thrun, S. (2003). Learning occupancy grid maps with forward sensor models. *Autonomous Robots*, 15(2):111--127.

Thrun, S. (2006). Winning the DARPA grand challenge: A robot race through the mojave desert. In *Proceedings - 21st IEEE/ACM International Conference on Automated Software Engineering, ASE 2006*, Seite 11.

Thrun, S., Burgard, W., und Fox, D. (2005). Probabilistic robotics (intelligent robotics and autonomous agents series). *Intelligent robotics and autonomous agents, The MIT ...*, 45:52.

Thrun, S., Liu, Y., Koller, D., Ng, A. Y., Ghahramani, Z., und Durrant-Whyte, H. (2004). Simultaneous localization and mapping with sparse extended information filters. In *International Journal of Robotics Research*, volume 23, Seiten 693--716.

Thrun, S. und Montemerlo, M. (2006). The graph SLAM algorithm with applications to large-scale mapping of urban structures. *International Journal of Robotics Research*, 25(5-6):403--429.

TomTom (2017). TomTom High Definition Map for Autonomous Driving Now Covers Western Europe.

Trevor, A. J., Rogers, J. G., und Christensen, H. I. (2014). OmniMapper: A modular multimodal mapping framework. In *Proceedings - IEEE International Conference on Robotics and Automation*, Seiten 1983--1990. IEEE.

Ulas, C. und Temeltas, H. (2011). A 3D scan matching method based on multi-layered Normal Distribution Transform. In *IFAC Proceedings Volumes (IFAC-PapersOnline)*, volume 44, Seiten 11602--11607.

Ulas, C. und Temeltas, H. (2013). A fast and robust feature-based scan-matching method in 3d slam and the effect of sampling strategies. *International Journal of Advanced Robotic Systems*, 10.

Unterholzner, A. (2015). *Sensor orientation selection and adaptive control of an actuated sensor platform for autonomous vehicles*. PhD thesis, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik, Neubiberg.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M. N., Dolan, J., Duggins, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T. M., Kolski, S., Kelly, A., Likhachev, M., McNaughton, M., Miller, N., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Salesky, B., Seo, Y. W., Singh, S., Snider, J., Stentz, A., Whittaker, W., Wolkowicki, Z., Ziglar, J., Bae,

H., Brown, T., Demitrish, D., Litkouhi, B., Nickolaou, J., Sadekar, V., Zhang, W., Struble, J., Taylor, M., Darms, M., und Ferguson, D. (2009a). Autonomous driving in Urban environments: Boss and the Urban Challenge. *Springer Tracts in Advanced Robotics*, 56(8):1--59.

Urmson, C., Baker, C., Dolan, J., Rybski, P., Salesky, B., Whittaker, W. R., Ferguson, D., und Darms, M. (2009b). Autonomous driving in traffic: Boss and the urban challenge. *AI Magazine*, 30(2):17--28.

Valencia, R. und Andrade-Cetto, J. (2018). Active pose SLAM. In *Springer Tracts in Advanced Robotics*, volume 119, Seiten 89--108. IEEE.

Vallivaara, I., Haverinen, J., Kemppainen, A., und Röning, J. (2010). Simultaneous localization and mapping using ambient magnetic field. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Seiten 14--19. IEEE.

Vatsavai, R. R., Burk, T. E., Lime, S., Hugentobler, M., Neumann, A., und Strobl, C. (2012). Open-source GIS. In *Springer Handbook of Geographic Information*, Seiten 939--966. Springer Berlin Heidelberg.

VDA (2015). Automatisierung: Von Fahrerassistenzsystemen zum automatisierten Fahren. Technical report, Verband der Automobilindustrie e.V.

Veronese, L. D. P., Ismail, A., Narayan, V., und Schulze, M. (2018). An Accurate and Computational Efficient System for Detecting and Classifying Ego and Sides Lanes Using LiDAR. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 1476--1483.

Viola, P. und Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1.

von Fritz, K. und Weiss, H. (1944). Kausalitat und Zufall in der Philosophie des Aristoteles. *The Journal of Philosophy*, 41(16):439.

Vysotska, O. und Stachniss, C. (2016). Exploiting building information from publicly available maps in graph-based SLAM. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2016-Novem, Seiten 4511--4516. Institute of Electrical and Electronics Engineers Inc.

Wan, E. A. und Van Der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. URL: `https://www.seas.harvard.edu/courses/cs281/papers/unscented.pdf`.

Wang, C. C., Thorpe, C., Thrun, S., Hebert, M., und Durrant-Whyte, H. (2007). Simultaneous localization, mapping and moving object tracking. *International Journal of Robotics Research*, 26(9):889--916.

Warren, M., McKinnon, D., He, H., Glover, A., Shiel, M., und Upcroft, B. (2014). Large scale monocular vision-only mapping from a fixed-wing sUAS. In *Springer Tracts in Advanced Robotics*, volume 92, Seiten 495--510. Springer.

Waymo Inc. (2017). On the road to Fully Self-driving. *Waymo Safety Report*, Seite 43.

Weikersdorfer, D. und Conradt, J. (2012). Event-based particle filtering for robot self-localization. In *2012 IEEE International Conference on Robotics and Biomimetics, ROBIO 2012 - Conference Digest*, Seiten 866--870. IEEE.

Wille, J. M., Saust, F., und Maurer, M. (2010). Stadtpilot: Driving autonomously on braunschweig's inner ring road. In *IEEE Intelligent Vehicles Symposium, Proceedings*, Seiten 506--511.

Winkle, T. (2015). Sicherheitspotenzial automatisierter Fahrzeuge: Erkenntnisse aus der Unfallforschung. In *Autonomes Fahren*, Seiten 351--376. Springer Berlin Heidelberg.

Wu, B., Wan, A., Yue, X., und Keutzer, K. (2017a). SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud. In *arXiv*, Seiten 1887--1893. IEEE.

Wu, C., Huang, T. A., Muffert, M., Schwarz, T., und Grater, J. (2017b). Precise pose graph localization with sparse point and lane features. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2017-Septe, Seiten 4077--4082.

Wu, E. Y., Li, G. Y., Xiang, Z. Y., und Liu, J. L. (2008). Stereo vision based SLAM using Rao-Blackwellised particle filter. *Journal of Zhejiang University: Science A*, 9(4):500--509.

Wu, K., Otoo, E., und Suzuki, K. (2009). Optimizing two-pass connected-component labeling algorithms. *Pattern Analysis and Applications*, 12(2):117--135.

Xiang, L., Ren, Z., Ni, M., und Jenkins, O. C. (2015). Robust graph SLAM in dynamic environments with moving landmarks. In *IEEE International Conference on Intelligent Robots and Systems*, Seiten 2543--2549.

Yan, F., Vysotska, O., und Stachniss, C. (2019). Global Localization on OpenStreetMap Using 4-bit Semantic Descriptors. In *2019 European Conference on Mobile Robots, ECMR 2019 - Proceedings*.

Yang, J., Jiang, Y. G., Hauptmann, A. G., und Ngo, C. W. (2007). Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the ACM International Multimedia Conference and Exhibition*, Seiten 197--206.

Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., und Darrell, T. (2020). BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seiten 2633--2642.

Zhang, J. und Singh, S. (2015). LOAM: Lidar Odometry and Mapping in Real-time. In *Robotics: Science and Systems*.

Zhang, M., Morris, D. D., und Fu, R. (2015). Ground Segmentation Based on Loopy Belief Propagation for Sparse 3D Point Clouds. In *Proceedings - 2015 International Conference on 3D Vision, 3DV 2015*, Seiten 615--622.

Zhou, B., Lapedriza, A., Torralba, A., und Oliva, A. (2017). Places: An Image Database for Deep Scene Understanding. *Journal of Vision*, 17(10):296.

Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., und Oliva, A. (2014). Learning deep features for scene recognition using places database. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., und Weinberger, K. Q., Herausgeber, *Advances in Neural Information Processing Systems*, volume 1, Seiten 487--495. Curran Associates, Inc.

Zhu, J., Gehrung, J., Huang, R., Borgmann, B., Sun, Z., Hoegner, L., Hebel, M., Xu, Y., und Stilla, U. (2020). Tum-mls-2016: An annotated mobile lidar dataset of the tum city campus for semantic point cloud interpretation in urban areas. *Remote Sensing*, 12(11).