

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

Wahrnehmung der statischen Umgebung eines autonomen Fahrzeugs auf Feldwegen durch die Fusion von Kamera und LiDAR

Bianca Forkel, M.Sc.

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

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Gutachter: 1. Univ.-Prof. Dr.-Ing. Hans-Joachim Wünsche
2. Univ.-Prof. Dr.-Ing. Markus Ulrich

Diese Dissertation wurde am 06.06.2024 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Luft- und Raumfahrttechnik am 05.01.2025 angenommen. Die mündliche Prüfung fand am 07.01.2025 statt.

Danksagung

Die vorliegende Dissertation entstand während meiner Tätigkeit als wissenschaftliche Mitarbeiterin am Institut für Technik Autonomer Systeme (TAS) der Universität der Bundeswehr München. Ich danke Herrn Prof. Dr.-Ing. Hans-Joachim Wünsche für diese Gelegenheit, an seinem Institut Forschung an realen Versuchsfahrzeugen betreiben zu können. Sich letzten Endes von seinen eigenen Algorithmen fahren lassen zu können, ist die größte Belohnung für die Arbeit mit realen Messdaten und die Lösung von echten Problemen, die „nur“ in der Praxis auftreten. Dass diese Arbeit zudem zu einer Dissertation geführt hat, verdanke ich weiterhin Herrn Prof. Dr.-Ing. Markus Ulrich und Herrn Prof. Dr. rer. nat. Christian J. Kähler, die freundlicherweise das Zweitgutachten und den Vorsitz der Prüfungskommission übernommen haben.

Bei meiner Arbeit konnte ich im Laufe der Jahre auf die Mithilfe vieler Kollegen zählen: Mein herzliches Dankeschön an alle, die das Institut und die Versuchsträger am Laufen gehalten haben! Unser Tiguan hat mich bei keiner Demo im Stich gelassen. Im Hinblick auf diese Dissertation danke ich explizit Patrick Neufeld für den Bau der in Kapitel 3 verwendeten, höhenverstellbaren Kalibriertafel. Ein großer Dank geht zudem an Jan Kallwies für meine Einarbeitung und die produktive Zusammenarbeit in der Anfangsphase meiner Promotion. Danke, dass du deine vielen Ideen geteilt hast und wir einen gemeinsamen Grundstein legen konnten! Ein besonderer Dank gilt außerdem Thorsten Lüttel und Michael Kusenbach – nicht nur für die tatkräftige Unterstützung am Auto und für das Korrekturlesen meiner Dissertation, sondern auch für die Gesellschaft an vielen langen Abenden. Schließlich danke ich Thomas Rottmann für sein stets offenes Ohr und die moralische Unterstützung.
Abseits des Büros möchte ich Kusi, Martin und Philipp danken: Danke für die lustigen Mittagspausen, Dienstreisen und anderweitigen Abenteuer, die die letzten Jahre bei TAS verfliegen ließen! Ganz besonders danke ich Philipp, meinem zudem unermüdlichen Diskussionspartner und Korrekturleser, meinem vertrauensvollen Sicherheitsfahrer, meiner helfenden Hand zu jeder Zeit, meinem Mitstreiter, meinem Partner – danke, dass du an meiner Seite bist!

Außerhalb der UniBw danke ich nicht zuletzt Anna für ihre wertvollen Anregungen zu meiner Dissertation und die Geduld in Stressphasen. Ich denke an meinen Opa Georg, der lange vor mir wusste, wohin mich mein Weg einmal führt – und damit nun doch recht behalten hat. Und vor allem danke ich meinen Eltern Ute und Jochen: Danke für eure bedingungslose Unterstützung!

Kurzfassung

Während die Automobilindustrie mit viel Einsatz das autonome Fahren auf Straßen vorantreibt, ist das autonome Fahren auf Feldwegen auch aufgrund ihrer heterogenen und über die Zeit veränderlichen Erscheinung und unzureichender Karten noch immer eine kaum erforschte Herausforderung. Im Vordergrund steht dabei die Wahrnehmung der statischen Umgebung: Wo verläuft der Weg, gibt es Weggabelungen, wo befinden sich nicht passierbare Hindernisse, wie erstreckt sich das Terrain?

Für die robuste Wahrnehmung komplexer Feldwege werden in dieser Arbeit RGB-Kamerabilder mit den Messungen eines LiDAR-Sensors fusioniert. Dazu wird zunächst eine Rohdatenfusion untersucht: Um die Genauigkeit von Stereo-Punktwolken zu verbessern, wird ein kostenbasiertes Verfahren zur Fusion einer LiDAR-Punktwolke in die Stereoverarbeitung vorgestellt. Dieses wird auf einem urbanen Benchmark-Datensatz evaluiert und übertrifft dabei die echtzeitfähigen Verfahren zur Stereo-LiDAR-Fusion aus der Literatur in ihrer Genauigkeit, Korrektheitsrate und Effizienz.

Für die höchste Genauigkeit auf Feldwegen und für eine Zuverlässigkeit unter allen Wetterbedingungen wird anschließend eine Sensordatenfusion auf Merkmalsebene realisiert. Dazu werden vier Module zur Repräsentation der statischen Umgebung auf Feldwegen implementiert: ein Hindernis-Grid, eine Terrain-Schätzung, ein darauf aufbauendes Straßen- und Wegetracking sowie eine Kreuzungserkennung.
Das **Hindernis-Grid** schätzt mittels eines Bayes-Filters Hinderniswahrscheinlichkeiten für diskrete Grid-Zellen. Um auf Feldwegen zwischen befahrbarer Vegetation (z. B. auf einem Grünstreifen in der Mitte des Wegs) und tatsächlichen Hindernissen zu unterscheiden, wird dabei eine LiDAR-basierte Belegtheitswahrscheinlichkeit mit einer kamerabasierten Befahrbarkeitswahrscheinlichkeit fusioniert.
Die **Terrain-Schätzung** schätzt mittels Optimierung aus akkumulierten LiDAR-Punkten ein kontinuierliches Elevationsmodell des Bodens. Um auf Feldwegen auch kleinere negative Hindernisse wie Schlaglöcher modellieren zu können, wird in einer dynamischen Auflösungsanpassung die Anzahl Stützpunkte des Modells gezielt lokal erhöht. Für diese Entscheidung werden Informationen aus dem Kamerabild genutzt.
Das **Straßen- und Wegetracking** schätzt in einem Optimierungsverfahren die Posen der Weg- oder Fahrspurränder. Dazu werden robuste Farbmodi und Messverfahren zur Kantenmessung im Kamerabild erarbeitet. Um die Robustheit bei schwierigen Sichtverhältnissen zu erhöhen, werden zusätzlich Messungen aus der LiDAR-Punktwolke gewonnen und Informationen aus dem Hindernis-Grid und der Terrain-Schätzung in das Straßen- und Wegetracking fusioniert.
Die **Kreuzungserkennung** erkennt schließlich anhand der entwickelten Farbmodi Feldwegkreuzungen in einem mit dem Kamerabild eingefärbten Terrain-Grid.

Die einzelnen Komponenten der statischen Umfeldrepräsentation wurden auf den institutseigenen Versuchsträgern in anspruchsvollen Szenarien getestet. Durch die Fusion von Kamera und LiDAR konnte eine Vielzahl von unterschiedlichen Feldwegen und ländlichen Straßen bei verschiedensten Wetterbedingungen und Lichtverhältnissen robust getrackt werden. So konnte auch bei Dunkelheit und Schnee einem zuvor unbekannten Feldweg mehrere Kilometer lang autonom gefolgt werden.

Abstract

While the automotive industry has been making great efforts on advancing autonomous driving on roads, autonomous driving on dirt roads has hardly been researched. Due to their heterogeneous and changing appearance over time and the lack of reliable maps, dirt roads present a particular challenge. The key aspect is the perception of the static environment: Where does the road run? Are there any crossroads? Where are impassable obstacles? What is the terrain like?

This work develops a robust perception system for complex dirt roads by fusing camera images with LiDAR sensor measurements. First, a raw data fusion approach is investigated. To improve the accuracy of stereo point clouds, a cost-based method to fuse a LiDAR point cloud into stereo processing is presented. Evaluations on an urban benchmark dataset show that this approach outperforms state-of-the-art methods for real-time stereo LiDAR fusion in accuracy, correctness, and efficiency.

Ultimately, for maximum accuracy on dirt roads and for reliability in all weather conditions, the sensor data fusion is realized at feature level. Therefore, four modules that represent the static environment on dirt roads are developed: an obstacle grid, a terrain estimation, a subsequent road and dirt road tracking, and a crossroad detection.
The **obstacle grid** estimates obstacle probabilities for discrete grid cells using a Bayes filter. It distinguishes between drivable vegetation (e.g., grass in the middle of the path) and actual obstacles by fusing LiDAR-based occupancy probabilities with camera-based drivability probabilities.
The **terrain estimation** estimates a continuous elevation model of the ground surface based on accumulated LiDAR points by means of optimization. To model smaller negative obstacles on dirt roads, such as potholes, the number of support points of the model can be increased locally. Thus, the resolution of the terrain model can be dynamically adjusted. When deciding where to increase the resolution, information from the camera image is taken into account.
The **(dirt) road tracking** estimates the poses of road or lane boundaries in an optimization process. For this purpose, edge measurement methods in camera images and robust image color modes are developed. To improve robustness under difficult visibility conditions, additional measurements from LiDAR point clouds are obtained and fused into the road tracking. Moreover, information from the obstacle grid and the terrain estimation is incorporated.
Finally, the **crossroad detection** identifies crossroads by applying the developed color modes to a terrain grid colored with camera images.

The individual components of the static environment model were tested on the institute's own test vehicles in challenging scenarios. The fusion of camera and LiDAR made it possible to robustly track a wide variety of dirt roads and rural roads under diverse weather and lighting conditions. Consequently, even in darkness and snow, a previously unknown dirt road could be followed autonomously for several kilometers.

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Voraussetzungen . . . 2
1.2 Fusion auf Rohdatenebene . . . 3
1.3 Fusion auf Merkmalsebene . . . 3

2 Grundlagen **7**
2.1 Sensorik . . . 7
2.2 Koordinatensysteme und Transformationen . . . 8
2.3 Kameramodell . . . 10
2.4 Klothoiden . . . 12
2.5 Zustandsschätzung . . . 13

I Voraussetzungen **17**

3 Kamera-zu-LiDAR-Kalibrierung **19**
3.1 Kalibrierkörper . . . 19
3.2 Optimierung der relativen Einbaulage . . . 22
3.3 Ergebnisse . . . 25

II Fusion auf Rohdatenebene **27**

4 Eingefärbte LiDAR-Punktwolke **29**

5 Fusion von Stereo und LiDAR **31**
5.1 Einleitung . . . 31
5.2 Stand der Forschung . . . 34
5.3 Kostenbasierte Fusion einer LiDAR-Punktwolke in Stereo Matching mittels Semi-Global Matching . . . 36
5.3.1 Stereo-SGM: Berechnung der Stereo-Matching-Kosten und Semi-Global Matching . . . 36
5.3.2 LiDAR-only-SGM: Berechnung der LiDAR-Matching-Kosten 38
5.3.3 LiDAR-SGM: Fusion von Stereo- und LiDAR-Matching-Kosten 40
5.4 Ergebnisse . . . 40
5.4.1 Setup . . . 41
5.4.2 Einfluss der LiDAR-Informationen auf das Stereo Matching . 43
5.4.3 Wahl der Sensorik . . . 45
5.4.4 Vergleich mit anderen Ansätzen . . . 47
5.4.5 Rechenzeiten . . . 49
5.4.6 Skalierung der Eingangsbilder . . . 49
5.4.7 Qualitative Ergebnisse . . . 51
5.5 Diskussion . . . 51

III Fusion auf Merkmalsebene 57

6 Hindernis-Grid 59
- 6.1 Einleitung und Problemstellungen . . . 59
- 6.2 Stand der Forschung . . . 62
- 6.3 Schätzung von Belegtheitswahrscheinlichkeiten . . . 64
  - 6.3.1 Bayes-Filter . . . 65
  - 6.3.2 Messungen aus der LiDAR-Punktwolke . . . 67
  - 6.3.3 Update der Belegtheitswahrscheinlichkeiten . . . 68
  - 6.3.4 Behandlung dynamischer Objekte . . . 70
    - 6.3.4.1 Ignorieren getrackter dynamischer Objekte . . . 70
    - 6.3.4.2 Löschen des hinterlassenen Schweifs eines dynamischen Objekts vor Aufsetzen des Trackers . . . 71
- 6.4 Bestimmung einer Befahrbarkeitswahrscheinlichkeit . . . 71
  - 6.4.1 Von Klassenwahrscheinlichkeiten zu einer Befahrbarkeitswahrscheinlichkeit . . . 73
  - 6.4.2 Integration in das Hindernis-Grid . . . 75
- 6.5 Fusion zu einer Hinderniswahrscheinlichkeit . . . 76
- 6.6 Laufzeitverhalten . . . 77
- 6.7 Diskussion . . . 78

7 Terrain-Schätzung 81
- 7.1 Einleitung und Problemstellung . . . 81
- 7.2 Stand der Forschung . . . 85
- 7.3 Terrain-Schätzung mit dynamischer Auflösungsanpassung . . . 86
  - 7.3.1 Extraktion von Messungen aus der LiDAR-Punktwolke . . . 87
  - 7.3.2 Akkumulieren der Messungen in hochaufgelösten Mess-Grids 89
  - 7.3.3 Glättung von Terrain-Kacheln unterschiedlicher Auflösung . 89
  - 7.3.4 Dynamische Skalierung der Terrain-Kacheln . . . 94
  - 7.3.5 Schätzung einer Straßenwahrscheinlichkeit und Einfärben des Terrain-Grids . . . 96
- 7.4 Ergebnisse . . . 99
  - 7.4.1 Qualitative Ergebnisse . . . 99
  - 7.4.2 Laufzeit . . . 104
- 7.5 Diskussion . . . 104

8 Straßen- und Wegetracking 107
- 8.1 Problemstellung . . . 107
- 8.2 Stand der Forschung . . . 109
- 8.3 Modellierung und Tracking . . . 112
  - 8.3.1 Modellierung von Straßen und Wegen durch Fahrspurrandpunkte 112
  - 8.3.2 Straßen- und Wegetracking durch Optimierung . . . 114
  - 8.3.3 Initialisierung und Verlängerung von Straßen und Wegen . . 116
- 8.4 Messungen aus Kamerabildern . . . 117
  - 8.4.1 CRONOS-Kantendetektion auf Straßen . . . 119
  - 8.4.2 Orientierung der Messfenster und Linienmessung . . . 120

8.4.3 CRONOS-Messungen auf Feldwegen . . . 122
8.5 Messungen aus LiDAR-Informationen . . . 128
8.5.1 Informationen aus der Terrain-Schätzung . . . 128
8.5.2 Informationen aus dem Hindernis-Grid . . . 129
8.5.3 Messungen aus einer segmentierten LiDAR-Punktwolke . . . 130
8.6 Fusion von Kamera, LiDAR und Modellwissen . . . 135
8.7 Ergebnisse . . . 137
8.7.1 Qualitative Ergebnisse . . . 137
8.7.2 Laufzeiten und Vorausschauweite . . . 143
8.7.3 Grenzen des Verfahrens . . . 144
8.8 Ausblick: Straßentracking auf Mischverkehrswegen mittels Radar-Sensorik . . . 145
8.9 Diskussion . . . 146

9 Kreuzungserkennung 147
9.1 Problemstellung . . . 147
9.2 Stand der Forschung . . . 148
9.3 Konzept . . . 149
9.4 Integration der Farbmodi aus dem Wegetracking . . . 153
9.4.1 Sättigungsfeature . . . 153
9.4.2 Grünfeature . . . 154
9.4.3 Feature-Grids . . . 155
9.5 Ergebnisse . . . 157
9.5.1 Qualitative Ergebnisse . . . 157
9.5.2 Vergleich mit dem Ausgangsverfahren . . . 157
9.5.3 Grenzen des Verfahrens . . . 161
9.5.4 Laufzeit . . . 163
9.6 Diskussion . . . 163

IV Fazit 165

10 Zusammenfassung und Ausblick 167

Symbolverzeichnis III
Abkürzungen . . . III
Konventionen . . . III

Eigene Veröffentlichungen VII

Literaturverzeichnis IX

1 Einleitung

„Wahrnehmung der statischen Umgebung eines autonomen Fahrzeugs auf Feldwegen durch die Fusion von Kamera und LiDAR“ – was bedeutet das und warum macht man das?

Autonome Fahrzeuge Das Ziel der Forschung zu autonomen Fahrzeugen ist die Entwicklung von Fahrerassistenzsystemen, die ein Fahrzeug ohne menschlichen Fahrer selbstständig zum Ziel bewegen können. Zwar werden derzeit bereits die ersten hochautomatisierten Systeme für den Straßenverkehr zugelassen, diese können jedoch nur in stark eingeschränkten Szenarien eingesetzt werden. Ein Beispiel hierfür ist der Mercedes Drive Pilot, der nur auf der Autobahn und nur bis zu einer Geschwindigkeit von 60 km/h aktiviert werden kann, dabei aber auch nicht in Baustellen, Tunneln, in Anwesenheit von Fußgängern oder bei schlechtem Wetter verwendet werden kann [Rudschies, 2023]. Darüber hinaus muss der Fahrer jederzeit in der Lage sein, innerhalb von 10 s wieder selbst die Kontrolle über das Fahrzeug zu übernehmen.

Wahrnehmung Damit das Fahrzeug selbstständig fahren kann, besitzt es Sensoren, die die Umgebung des Fahrzeugs erfassen, sowie Aktoren, die das Fahrzeug steuern. Dazwischen werden verschiedene Softwareschichten benötigt: Die Sensoreinzüge übersetzen die eingehenden Datenströme der Sensoren und setzen daraus beispielsweise ein Kamerabild zusammen. Die Wahrnehmungsschicht interpretiert die Sensordaten und erkennt darin relevante Objekte und Strukturen in der Umgebung des Fahrzeugs. Anhand dieser wahrgenommenen Objekte und Strukturen entscheidet die Verhaltensplanung über die durchzuführende Fahraufgabe, beispielsweise einer Fahrspur zu folgen, ein vorausfahrendes Fahrzeug zu überholen oder einem Hindernis auszuweichen. Darauf basierend plant die Pfad- bzw. Trajektorienplanung, wo und gegebenenfalls wie schnell das Fahrzeug fahren soll. Die Fahrzeugregelung sorgt schließlich für die korrekte Umsetzung dieser Fahrabsichten durch die Aktorik.

Statische Umgebung Diese Arbeit entwickelt Methoden zur Wahrnehmung der statischen Umgebung. Im Gegensatz zur dynamischen Umgebung, die andere Verkehrsteilnehmer wie Fahrzeuge, Fahrradfahrer oder Fußgänger beinhaltet, umfasst die statische Umgebung unbewegte Hindernisse auf und neben der Straße, den Boden und insbesondere die Straße selbst. Das Ziel ist es also, wahrzunehmen, wo sich eine Straße befindet, der das Fahrzeug folgen könnte, welche Steigungen und Unebenheiten dabei auftreten, ob es die Möglichkeit gibt abzubiegen, und welche Hindernisse den befahrbaren Bereich einschränken.

Feldwege Der Fokus dieser Arbeit liegt dabei jedoch, anders als in den Entwicklungen durch die Automobilindustrie, auf dem autonomen Fahren auf Feldwegen. Die Wahrnehmung der statischen Umgebung auf Feldwegen ist zum einen weit weniger erforscht, zum anderen aber auch herausfordernder als auf größeren Straßen: Für Feldwege existieren oftmals keine oder nur sehr ungenaue Karten, zumal sich die Position und die Beschaffenheit eines Feldwegs im Laufe der Zeit verändern. Auch mit einer Karte ist es schwierig, sich präzise genug in dieser Karte zu lokalisieren, da

der GNSS-Empfang (Globales Navigationssatellitensystem, engl. *Global Navigation Satellite System*) insbesondere im Wald oft schlecht ist, und selten verlässliche Landmarken kartiert sind. Daher muss ein autonomes Fahrzeug auf Feldwegen in der Lage sein, ohne Kartenwissen auszukommen und eigenständig den Weg zu erkennen. Dabei hat ein Feldweg keine Straßenmarkierungen oder Bordsteine, die die Spurerkennung erleichtern, und ist meist nicht geteert, wodurch sein Aussehen unregelmäßig und die Wegränder unscharf sind. Zudem variiert das Erscheinungsbild von Feldweg zu Feldweg stark, während gleichzeitig kaum Trainingsdaten für die Feldwegerkennung verfügbar sind. Auch die Detektion von Hindernissen gewinnt auf Feldwegen zum einen an Relevanz, da beispielsweise mit umgestürzten Bäumen auf dem Weg gerechnet werden muss, zum anderen an Schwierigkeit, da zwischen befahrbarer, höherer Vegetation und tatsächlich unbefahrbaren Hindernissen unterschieden werden muss. Im Boden müssen außerdem anders als auf urbanen Straßen Steigungen, Schlaglöcher und Gräben detektiert werden.

Fusion von Kamera und LiDAR Um auch auf Feldwegen eine robuste Wahrnehmung der statischen Umgebung zu ermöglichen, wird in dieser Arbeit nicht nur eine Sensormodalität benutzt, sondern die Fusion eines farbgebenden Kamerabilds mit Informationen eines abstandsgebenden LiDAR-Sensors (*Light Detection And Ranging*) untersucht. Die präzisen Abstandsinformationen des LiDAR-Sensors helfen nicht nur dabei, Hindernisse und Boden zu lokalisieren, sondern auch die zunächst zweidimensionale (horizontaler und vertikaler Winkel) Messung einer einzelnen Kamera in den dreidimensionalen Raum zu übertragen. Die Sensordatenfusion kann dabei auf verschiedenen Ebenen erfolgen (vgl. [Darms, 2015]): Bei der Fusion auf Rohdatenebene werden die Sensordaten vor der Fusion lediglich minimal vorverarbeitet, und erst in den fusionierten Sensordaten werden weitere Merkmale extrahiert. Bei der Fusion auf Merkmalsebene werden aus beiden Sensoren unabhängig voneinander Merkmale der zu untersuchenden Objekte extrahiert und die Merkmale aus beiden Sensordaten fusioniert, um ein Objekt zu erkennen und seine Position zu bestimmen. Bei der Fusion auf Entscheidungs- oder Objektebene wird schließlich in beiden Sensoren unabhängig voneinander eine Objektposition bestimmt, die durch die Fusion beider Detektionen korrigiert wird. In dieser Arbeit werden die Fusion auf Rohdatenebene und die Fusion auf Merkmalsebene betrachtet.

Gliederung Kapitel 2 stellt zunächst die für die Sensordatenfusion verwendete Sensorik vor und führt zum besseren Verständnis dieser Arbeit wichtige formale und technische Grundlagen ein. Die weitere Arbeit gliedert sich dann in drei Hauptteile: Teil I behandelt die Schaffung der Voraussetzungen für die Sensordatenfusion, Teil II die Fusion auf Rohdatenebene und Teil III die Fusion auf Merkmalsebene.

1.1 Voraussetzungen

Kamera-zu-LiDAR-Kalibrierung (Kapitel 3) Für jede Art der Sensordatenfusion ist es essenziell, die Lage der Sensoren zueinander zu kennen. In Kapitel 3 wird daher vorbereitend ein Kalibrierungsverfahren zur extrinsischen Kalibrierung der Kamera zum LiDAR-Sensor entworfen.

1.2 Fusion auf Rohdatenebene

Anschließend werden zwei unterschiedliche Ansätze zur Rohdatenfusion diskutiert.

Eingefärbte LiDAR-Punktwolke (Kapitel 4) Die einfachste Fusion einer LiDAR-Punktwolke mit einem Kamerabild stellt die eingefärbte LiDAR-Punktwolke dar. Dabei werden auf Rohdatenebene den Punkten der LiDAR-Punktwolke Pixel des Kamerabilds zugeordnet, um die Punktwolke um Farbinformationen zu ergänzen. Auf die Vor- und Nachteile dieses Verfahrens wird in Kapitel 4 eingegangen.

Fusion von Stereo und LiDAR (Kapitel 5) Im Anschluss wird eine alternative Methode zur Rohdatenfusion von Kamera und LiDAR beleuchtet: die Fusion der LiDAR-Punktwolke in das Stereosehen. Ohne die Nutzung eines abstandsgebenden Sensors müsste die Entfernung eines in der Kamera beobachteten Punkts entweder über die Zeit geschätzt oder mithilfe einer zweiten Kamera mittels Stereoverarbeitung bestimmt werden. Um die Genauigkeit der durch das Stereosehen gewonnenen Entfernungsinformationen zu verbessern, wird untersucht, eine LiDAR-Punktwolke in die Stereoverarbeitung zu fusionieren. Hierfür wird in Kapitel 5 ein neues Verfahren entwickelt und evaluiert. Abbildung 1.1 zeigt ein Beispiel für ein mit diesem Verfahren geschätztes Disparitätsbild, aus dem für jedes Pixel eines Kamerabilds eine geschätzte Entfernung berechnet werden kann.

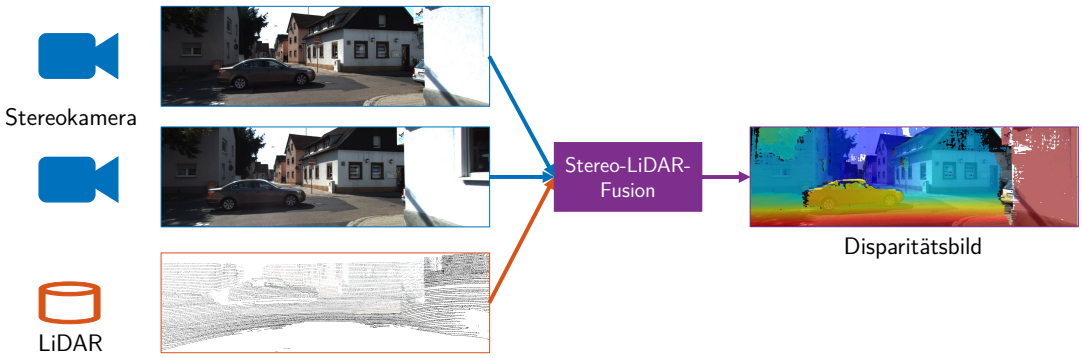


Abbildung 1.1:
Beispiel für die Rohdatenfusion von Stereo und LiDAR in Kapitel 5. Das Disparitätsbild visualisiert den geschätzten Abstand eines Pixels zur Kamera von rot (nah) zu blau (fern).

1.3 Fusion auf Merkmalsebene

Zur Wahrnehmung der statischen Umgebung wird jedoch schließlich der Ansatz der Merkmalsfusion verfolgt. Damit wird in Kapitel 6 bis 9 eine Repräsentation der statischen Umgebung aus vier Komponenten erarbeitet. Abbildung 1.2 zeigt ein Flussdiagramm dieser Umgebungsrepräsentation: Die zentrale Komponente für das autonome Fahren auf Feldwegen stellt ein Straßen- und Wegetracking dar. Dieses basiert auf einem Hindernis-Grid und einer Terrain-Schätzung. Vervollständigt wird die Umgebungsrepräsentation durch eine Kreuzungserkennung.

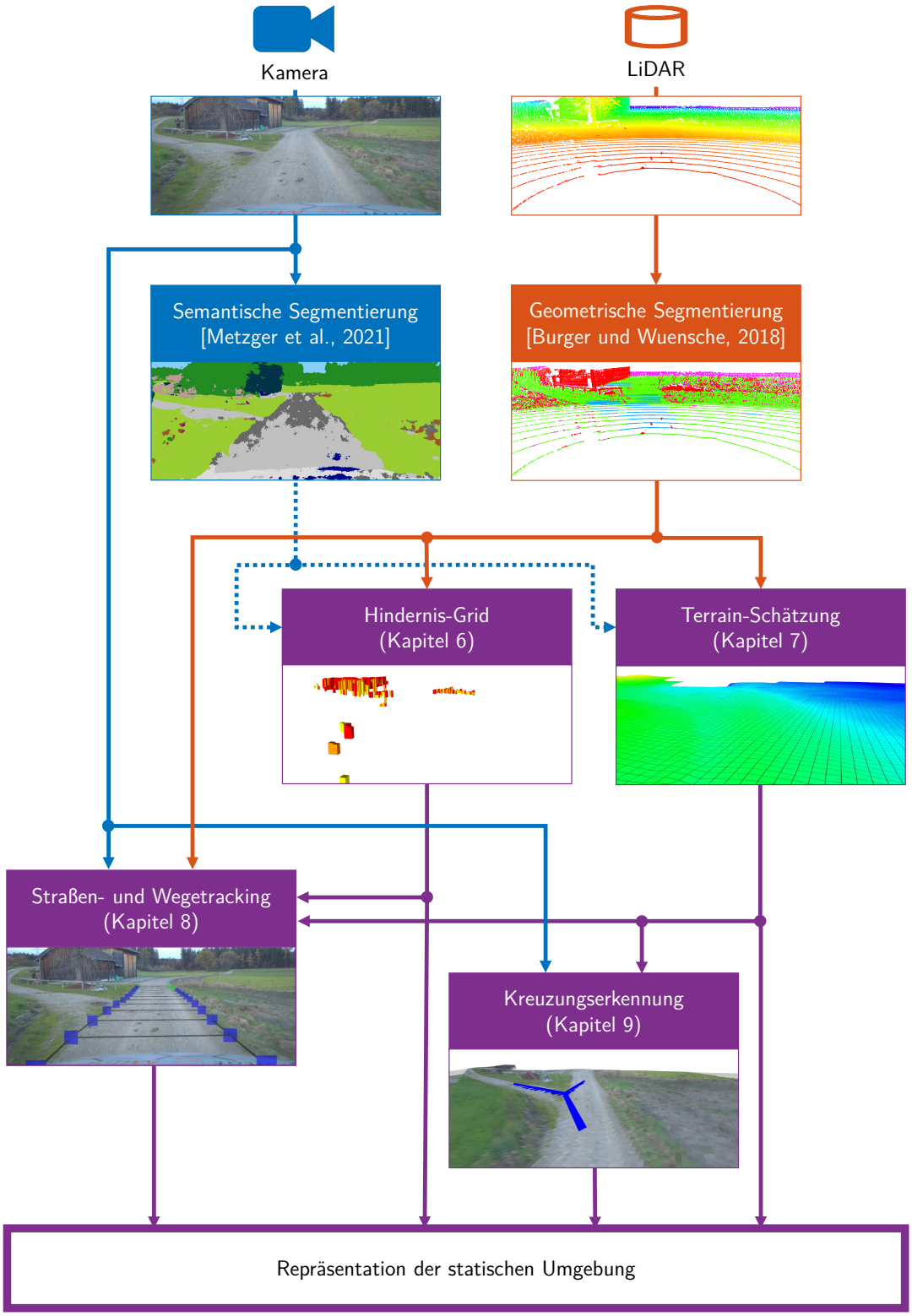


Abbildung 1.2:
Flussdiagramm der in dieser Arbeit entwickelten Repräsentation der statischen Umgebung durch Fusion (lila) von Merkmalen aus dem Kamerabild (blau) und der LiDAR-Punktwolke (rot). Optionale Datenströme werden gepunktet dargestellt.

Hindernis-Grid (Kapitel 6) Im Hindernis-Grid wird aus den vorverarbeiteten Punkten einer LiDAR-Punktwolke und optional einem vorverarbeiteten Kamerabild für jeden Bereich der Umgebung die Wahrscheinlichkeit dafür geschätzt, dass dieser Bereich ein statisches Hindernis enthält. Das Ergebnis ist ein Grid, das für jede Grid-Zelle eine Hinderniswahrscheinlichkeit sowie gegebenenfalls die vertikale Ausdehnung des Hindernisses in dieser Grid-Zelle enthält. Dieses Hindernis-Grid wird im Straßen- und Wegetracking dafür benötigt, einen hindernisfreien Weg zu erkennen.

Terrain-Schätzung (Kapitel 7) Aus denselben Eingangsdaten wird in einer davon unabhängigen Terrain-Schätzung für jeden Bereich der Umgebung der Verlauf der Bodenfläche geschätzt. Dazu muss die Bodenoberfläche von Hindernissen getrennt werden. Das Ergebnis ist ein zweites Grid, das für jede Grid-Zelle die Höhe und die Steigung des Bodens enthält. Diese hindernisfreie Terrain-Schätzung hilft im Straßen- und Wegetracking dabei, die fehlende Entfernungsinformation von Kameramessungen auszugleichen und die Lage des Wegs im dreidimensionalen Raum zu bestimmen.

Straßen- und Wegetracking (Kapitel 8) Mithilfe der Informationen aus dem Hindernis-Grid und der Terrain-Schätzung sowie anhand gezielter Messungen im Kamerabild und in der vorverarbeiteten LiDAR-Punktwolke können im Straßen- und Wegetracking ein befahrbarer Weg gesucht und die Pose seiner Spurbegrenzungen geschätzt werden. Dabei liegt der Fokus des Straßen- und Wegetrackings nicht nur auf Feldwegen, sondern es sollen einheitliche Messverfahren für Straßen und Feldwege gefunden werden. Zudem wird eine Robustheit auf unterschiedlichsten Wegen und bei allen Wetterbedingungen angestrebt und evaluiert.

Kreuzungserkennung (Kapitel 9) Schließlich werden bei der Kreuzungserkennung anhand des Kamerabilds und der Terrain-Schätzung mögliche Feldwegkreuzungen erkannt. Dabei können Erkenntnisse aus dem Straßen- und Wegetracking genutzt werden.

Kapitel 10 zieht letztlich ein Fazit und stellt zusammenfassend die Forschungsbeiträge der Arbeit heraus. Einige Beiträge dieser Arbeit wurden dabei bereits vorab auf internationalen Konferenzen veröffentlicht [Forkel et al., 2021a,b, Forkel und Wuensche, 2022a,b, 2023, Forkel et al., 2024].

2 Grundlagen

Dieses Kapitel führt Grundlagen ein, die für mehrere der folgenden Kapitel relevant sind. Weitere, spezifischere Grundlagen werden in den einzelnen Kapiteln behandelt.

2.1 Sensorik

Die Wahrnehmung der statischen Umgebung basiert in dieser Arbeit auf Kameras und LiDAR-Sensoren (*Light Detection And Ranging*). Während eine Kamera passiv das Umgebungslicht einfängt, um Farbinformationen zu messen, sendet ein LiDAR-Sensor aktiv Lichtimpulse aus, um anhand der Signallaufzeit des durch Objekte reflektierten Lichts den Abstand dieser Objekte zu messen (für die genaue Funktionsweise sei auf [Gotzig und Geduld, 2015] verwiesen). Am Institut für Technik Autonomer Systeme (TAS) werden rotierende LiDAR-Sensoren verwendet. Dabei werden bis zu 128 Laserdioden in 100 ms einmal 360° um die eigene Achse rotiert. Aus den Ergebnissen einer Umdrehung lässt sich eine Punktwolke von 3D-Punkten $\mathbf{p} = [x\ y\ z]^\mathsf{T}$ von der Umgebung des Fahrzeugs berechnen. Da sich der LiDAR-Sensor durch die Fahrt des Fahrzeugs gegenüber der Umwelt bewegt, wird die geschätzte Eigenbewegung des Fahrzeugs in der LiDAR-Punktwolke kompensiert, um korrigierte Punktmessungen in einem stationären Koordinatensystem zu erhalten (vgl. [Schneider et al., 2010]).

Bis auf Kapitel 5 wurden für die gesamte vorliegende Arbeit institutseigene Versuchsträger genutzt: MuCAR-3 (*Munich Cognitive Autonomous Robot Car 3rd Generation*) – ein VW Touareg – und MuCAR-4 (*Munich Cognitive Autonomous Robot Car 4th Generation*) – ein VW Tiguan. Das gemeinsame Sensorsetup dieser zwei Versuchsträger beinhaltet jeweils einen zentralen LiDAR-Sensor – zu Beginn der Entstehung dieser Arbeit (2019) ein Velodyne HDL-64E S2 mit 64 Laserdioden und 133 000 Punkten pro Umdrehung [Velodyne LiDAR, Inc., 2011], zum Ende der Entwicklungsarbeiten (2023) ein Velodyne Alpha Prime (VLS-128) mit 128 Laserdioden und 240 000 Punkten pro Umdrehung [Velodyne LiDAR, Inc., 2018] (es wird jeweils nur die erste empfangene Reflexion verwendet). Außerdem besitzen beide Versuchsträger hinter der Windschutzscheibe eine Kamera auf einer drehbaren Kameraplattform namens MarVEye (Multifokales aktives / reaktives Fahrzeug-Auge, engl. *Multifocal active / reactive Vehicle Eye*) (vgl. [Unterholzner, 2016]). In dieser Arbeit wird mittels des MarVEye die Kamera dem Lenkwinkel folgend nachgeführt, sodass der aktuelle Weg stets im Blickfeld bleibt. Dafür wird die Positionsänderung $(\Delta x, \Delta y)$ des Fahrzeugs relativ zur aktuellen Fahrzeugpose innerhalb der nächsten 5 s (und mindestens 10 m) prädiziert, und daraus der Soll-Gierwinkel $\Psi = \arctan2(\Delta y, \Delta x)$ der Kameraplattform berechnet. Durch das Nachführen der Kamera können Objektive mit geringerem Öffnungswinkel gewählt und damit höhere Winkelauflösungen erreicht werden: Die neuesten Kameras von MuCAR-3 und MuCAR-4 haben einen horizontalen Öffnungswinkel von 55° bzw. 53° bei 2448 px × 2048 px Auflösung. Alle Kameras beinhalten CMOS-Sensoren mit Global Shutter. Auf dem Dach von MuCAR-4 befinden sich drei zusätzliche Kameras als Teil eines horizontalen und vertikalen Stereosetups. Abbildung 2.1 zeigt ein Bild des Sensorsetups von MuCAR-4.

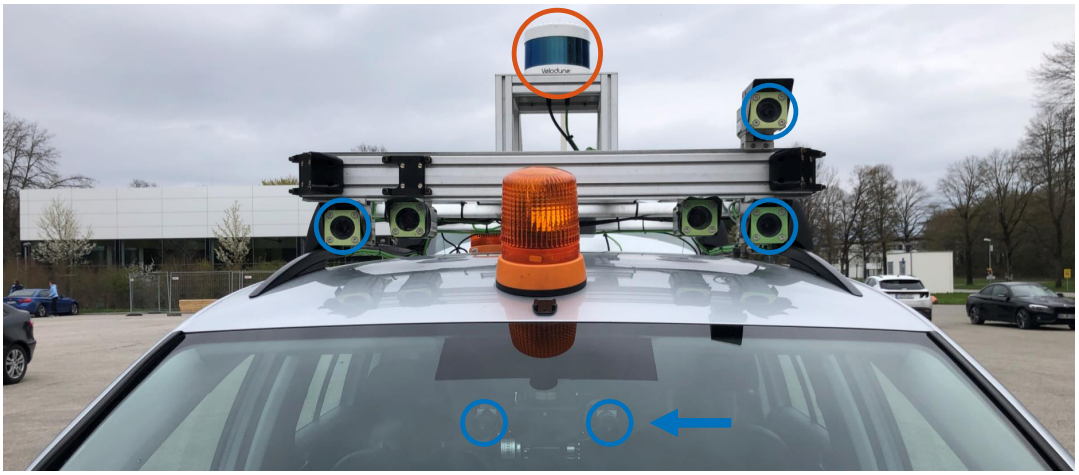


Abbildung 2.1:
Sensoraufbauten des Versuchsträgers MuCAR-4 im April 2023. Auf dem Dach befindet sich ein zentraler Velodyne Alpha Prime LiDAR-Sensor (rot) sowie ein Stereokamera-Setup bestehend aus einer Referenzkamera und je einer dazu horizontal und vertikal versetzten Kamera (blau). Hinter der Windschutzscheibe befinden sich zwei bewegte Kameras (blau). Der Pfeil markiert die in dieser Arbeit verwendete Kamera.

Zur Schätzung der Eigenbewegung besitzen die Versuchsträger schließlich ein Inertiales Navigationssystem, engl. *Inertial Navigation System* (INS), das mittels Beschleunigungs- und Drehratensensoren die relative Bewegung des Fahrzeugs schätzt und mittels RTK-GNSS (Globales Navigationssatellitensystem, engl. *Global Navigation Satellite System*) (Echtzeitkinematik, engl. *Real Time Kinematic*) eine globale Lokalisierung erlaubt. Die absolute Pose wird in dieser Arbeit jedoch lediglich zu Visualisierungszwecken verwendet, beispielsweise um Luftbilder zeigen zu können.

Aufgrund der Messrate des LiDAR-Sensors von 10 Hz sollen in dieser Arbeit alle Daten innerhalb von 100 ms verarbeitet werden, damit ein Verfahren als echtzeitfähig gilt. Die zeitlichen Auswertungen basieren dabei auf einer Intel© Core™ i7-8700K CPU (2017) mit 3,70 GHz und einer Nvidia GeForce GTX 1050Ti GPU (2016) mit dem Betriebssystem Ubuntu 20.04 und dem Framework ROS (*Robot Operating System*) in der Version Noetic. Das Framework dient dem zentralen Datenaustausch der einzelnen Softwaremodule.

2.2 Koordinatensysteme und Transformationen

Die Koordinaten jedes Punkts ${}^{\text{coord}}\mathbf{p} = [x\ y\ z]^{\mathsf{T}}$ beziehen sich auf ein bestimmtes Koordinatensystem *coord*. Wichtige Koordinatensysteme in dieser Arbeit sind das Fahrzeugkoordinatensystem $\mathbf{F}_{\text{ego}}$ im Schwerpunkt des Fahrzeugs, das Koordinatensystem des LiDAR-Sensors $\mathbf{F}_{\text{lidar}}$, das Kamerakoordinatensystem $\mathbf{F}_{\text{cam}}$ sowie das stationäre Koordinatensystem der Odometrie $\mathbf{F}_{\text{odom}}$. Das Odometriekoordinatensystem hat seinen Ursprung an der Position, an der das Fahrzeug eingeschaltet wird, und integriert die geschätzte Eigenbewegung auf. Dadurch driftet das Koordinatensystem zwar über die Zeit vom Weltkoordinatensystem weg, ist jedoch sprungfrei.

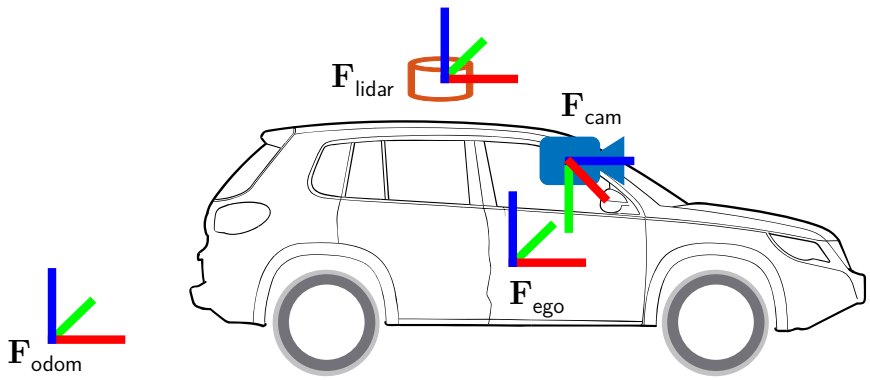


Abbildung 2.2:
Visualisierung der wichtigsten Koordinatensysteme dieser Arbeit. Die x-Achse wird rot dargestellt, die y-Achse grün und die z-Achse blau.

Alle Koordinatensysteme sind rechtshändig definiert. Bei $\mathbf{F}_{\text{ego}}$ und $\mathbf{F}_{\text{lidar}}$ zeigt die x-Achse entlang des Fahrzeugs nach vorne, die y-Achse nach links und die z-Achse nach oben. Auch bei $\mathbf{F}_{\text{odom}}$ zeigt die z-Achse nach oben. Lediglich $\mathbf{F}_{\text{cam}}$ ist gemäß der Konvention in ROS davon abweichend definiert: Hier zeigt die x-Achse im Kamerabild nach rechts, die y-Achse im Kamerabild nach unten, und die z-Achse in das Kamerabild hinein. Abbildung 2.2 visualisiert diese Koordinatensysteme.

Winkel werden entweder mittels einer Quaternion $\mathbf{q} = [q_{\text{x}}\ q_{\text{y}}\ q_{\text{z}}\ q_{\text{w}}]^{\text{T}}$ oder mittels des Rollwinkels Φ um die x-Achse, des Nickwinkels Θ um die y-Achse und des Gierwinkels Ψ um die z-Achse dargestellt. Dabei erfolgt zuerst die Rotation um die Gierachse (engl. *yaw*), dann die Rotation um die Nickachse (engl. *pitch*) und zuletzt die Rotation um die Rollachse (extrinsische XYZ-Drehung mit den Yaw-Pitch-Roll-Winkeln (Ψ, Θ, Φ) um raumfeste Achsen, entspricht der intrinsischen ZYX-Drehung mit den Eulerwinkeln $(\alpha = \Phi, \beta = \Theta, \gamma = \Psi)$ um körperfeste Achsen, vgl. [Waldron und Schmiedeler, 2008, S. 11]). Die Umrechnung einer Quaternion in Eulerwinkel (Φ, Θ, Ψ) wird mit der Funktion $\text{euler}(\cdot)$ notiert.

Um zwischen den Koordinatensystemen umzurechnen, wird aus der homogenen Translationsmatrix $\underline{\mathbf{T}}$ und der homogenen Rotationsmatrix $\underline{\mathbf{R}}$ eine Homogene Transformationsmatrix (HTM) berechnet:

$$
\begin{aligned}
{}^{\text{to}}\mathbf{H}_{\text{from}} &= \underline{\mathbf{T}}(t_1, t_2, t_3) \cdot \underline{\mathbf{R}}(\Phi, \Theta, \Psi) = \\
&\begin{bmatrix} \mathbf{I} & \mathbf{t}(t_1, t_2, t_3) \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{R}(\Phi, \Theta, \Psi) & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}(\Phi, \Theta, \Psi) & \begin{bmatrix} t_1\ t_2\ t_3 \end{bmatrix}^{\text{T}} \\ 0 & 1 \end{bmatrix},
\end{aligned} \tag{2.1}
$$

wobei $\mathbf{t}$ den Translationsvektor darstellt und $\mathbf{R}$ die Rotationsmatrix:

$$
\begin{aligned}
&\mathbf{R}(\Phi, \Theta, \Psi) = \mathbf{R}_{\text{x}}(\Phi) \cdot \mathbf{R}_{\text{y}}(\Theta) \cdot \mathbf{R}_{\text{z}}(\Psi) = \\
&\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\Phi) & \sin(\Phi) \\ 0 & -\sin(\Phi) & \cos(\Phi) \end{bmatrix} \begin{bmatrix} \cos(\Theta) & 0 & -\sin(\Theta) \\ 0 & 1 & 0 \\ \sin(\Theta) & 0 & \cos(\Theta) \end{bmatrix} \begin{bmatrix} \cos(\Psi) & \sin(\Psi) & 0 \\ -\sin(\Psi) & \cos(\Psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}.
\end{aligned} \tag{2.2}
$$

2.3 Kameramodell

Ein Pixel $\mathsf{p} = (u, v)$ bezieht sich auf das zweidimensionale Bildkoordinatensystem, das seinen Ursprung in der linken oberen Ecke des Kamerabilds hat. Dabei wird in dieser Arbeit von unverzerrten Kamerabildern ausgegangen – die Verzeichnung des Bilds durch das Objektiv wird bereits im Einzug näherungsweise korrigiert.

Zur Projektion eines 3D-Punkts ${}^{\mathsf{coord}}\mathbf{p} = [x\ y\ z]^\mathsf{T}$ in das Kamerabild wird der Punkt zunächst in einen homogenen Punkt ${}^{\mathsf{coord}}\underline{\mathbf{p}} = [x\ y\ z\ 1]^\mathsf{T}$ umgewandelt (dieser Schritt wird in Zukunft nicht mehr explizit angegeben). Der homogene Punkt wird mittels der HTM ${}^{\mathsf{cam}}\mathbf{H}_{\mathsf{coord}}$ in das Kamerakoordinatensystem transformiert:

$$ {}^{\mathsf{cam}}\underline{\mathbf{p}} = {}^{\mathsf{cam}}\mathbf{H}_{\mathsf{coord}} \cdot {}^{\mathsf{coord}}\underline{\mathbf{p}}\,, \tag{2.3} $$

wobei ${}^{\mathsf{cam}}\mathbf{H}_{\mathsf{coord}}$ statische Transformationen, bspw. vom Fahrzeugkoordinatensystem zur Kamera, wie auch dynamisch veränderliche Transformationen wie die Transformation vom Odometrie- zum Fahrzeugkoordinatensystem enthalten kann. Der Punkt im Kamerakoordinatensystem wird dann entsprechend des Lochkameramodells (vgl. Abbildung 2.3) zunächst in die durch die Brennweite f (in mm) definierte Bildebene ${}^{\mathsf{cam}}z_{\mathsf{plane}} = f$ projiziert. Aus der Anwendung des Strahlensatzes

$$ \frac{{}^{\mathsf{plane}}x}{f} = \frac{{}^{\mathsf{cam}}x}{{}^{\mathsf{cam}}z} \qquad \text{bzw.} \qquad \frac{{}^{\mathsf{plane}}y}{f} = \frac{{}^{\mathsf{cam}}y}{{}^{\mathsf{cam}}z} \tag{2.4} $$

ergibt sich die perspektivische Projektion

$$ {}^{\mathsf{plane}}\mathbf{p} = \begin{bmatrix} {}^{\mathsf{plane}}x \\ {}^{\mathsf{plane}}y \\ {}^{\mathsf{plane}}z \end{bmatrix} = \frac{f}{{}^{\mathsf{cam}}z} \begin{bmatrix} {}^{\mathsf{cam}}x \\ {}^{\mathsf{cam}}y \\ {}^{\mathsf{cam}}z \end{bmatrix} \tag{2.5} $$

des Punktes ${}^{\mathsf{cam}}\mathbf{p} = [{}^{\mathsf{cam}}x\ {}^{\mathsf{cam}}y\ {}^{\mathsf{cam}}z]^\mathsf{T}$ vom Kamerakoordinatensystem in die Bildebene. Diese Koordinaten in Millimetern werden anschließend anhand der horizontalen und vertikalen Pixelabstände (engl. *Pixel Pitch*) s_x bzw. s_y des Kamerachips in die Einheit Pixel umgerechnet [Steger et al., 2018, S. 221]:

$$ {}^{\mathsf{plane}}\mathsf{p} = ({}^{\mathsf{plane}}u, {}^{\mathsf{plane}}v) = \left(\frac{{}^{\mathsf{plane}}x}{s_\mathsf{x}}, \frac{{}^{\mathsf{plane}}y}{s_\mathsf{y}} \right) . \tag{2.6} $$

Das Bildebenenkoordinatensystem hat seinen Ursprung im Schnittpunkt der z-Achse des Kamerakoordinatensystems (der optischen Achse) mit der Bildebene. Dieser Punkt, der Hauptpunkt genannt wird, liegt grob in der Mitte des Bilds. Zur Umrechnung in das Bildkoordinatensystem, dessen Ursprung in der linken oberen Ecke des Bilds liegt, muss also zuletzt noch der Referenzpunkt verschobenen werden:

$$ {}^{\mathsf{img}}\mathsf{p} = {}^{\mathsf{plane}}\mathsf{p} + \mathsf{c} = ({}^{\mathsf{plane}}u + c_\mathsf{u}, {}^{\mathsf{plane}}v + c_\mathsf{v})\,, \tag{2.7} $$

wobei $\mathsf{c} = (c_\mathsf{u}, c_\mathsf{v})$ die Position des Hauptpunkts in Bildkoordinaten angibt.

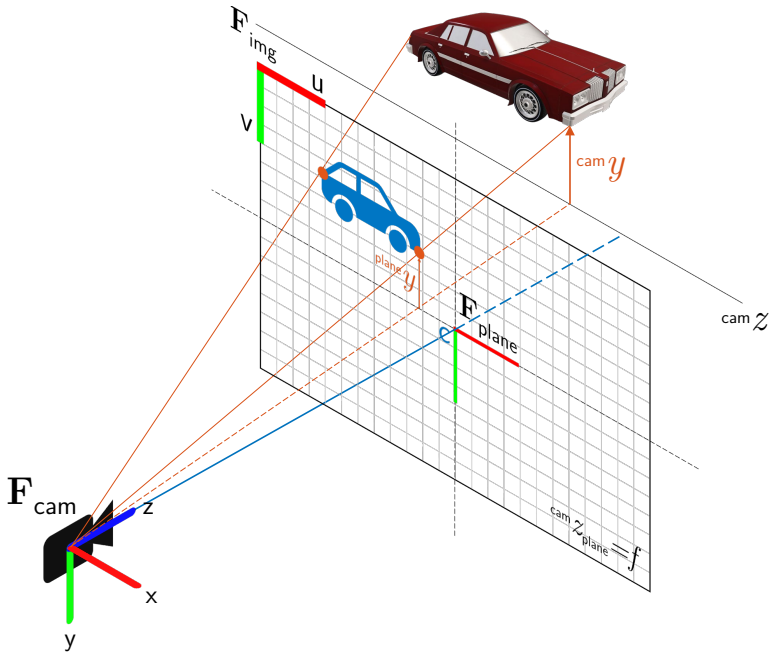


Abbildung 2.3:
Schematisches Beispiel der Projektion eines 3D-Objekts im Kamerakoordinatensystem $\mathbf{F}_{\text{cam}}$ in das Bildkoordinatensystem $\mathbf{F}_{\text{img}}$ auf der Bildebene $\mathbf{F}_{\text{plane}}$ nach dem Lochkameramodell.

Die Parameter f, s_{x}, s_{y}, c_{u} und c_{v}, auch intrinsische Kameraparameter genannt, müssen aufgrund mechanischer Ungenauigkeiten – genau wie die Verzeichnung – durch eine intrinsische Kalibrierung bestimmt werden. Hier werden meist direkt die Brennweiten $f_{\text{u}} = f/s_{\text{x}}$ und $f_{\text{v}} = f/s_{\text{y}}$ in Pixeln bestimmt. Damit ergibt sich die vollständige Projektion eines 3D-Punkts im Kamerakoordinatensystem in das entzerrte Kamerabild als:

$$^{\text{img}}\text{p} = (u, v) = \frac{1}{^{\text{cam}}z} \left(f_{\text{u}} \, ^{\text{cam}}x + c_{\text{u}}, f_{\text{v}} \, ^{\text{cam}}y + c_{\text{v}} \right) . \tag{2.8}$$

Zur Modellierung von Verzeichnungen sei auf [Steger et al., 2018, S. 219] verwiesen.

In Matrix-Schreibweise wird die Kamera-Projektionsmatrix

$$^{\text{img}}\mathbf{P}_{\text{cam}} = \begin{bmatrix} f_{\text{u}} & 0 & c_{\text{u}} & 0 \\ 0 & f_{\text{v}} & c_{\text{v}} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.9}$$

aufgestellt, um den in homogenen Kamerakoordinaten gegebenen Punkt $^{\text{cam}}\underline{\mathbf{p}}$ in (zunächst nicht normalisierte) homogene Bildkoordinaten umzurechnen:

$$^{\text{img}}\underline{\text{p}} = \begin{bmatrix} u' & v' & w \end{bmatrix}^{\text{T}} = {}^{\text{img}}\mathbf{P}_{\text{cam}} \cdot {}^{\text{cam}}\underline{\mathbf{p}} . \tag{2.10}$$

Abschließend werden die Bildkoordinaten normalisiert:

$$^{\text{img}}\text{p} = (u, v) = \left(\frac{u'}{w}, \frac{v'}{w} \right) . \tag{2.11}$$

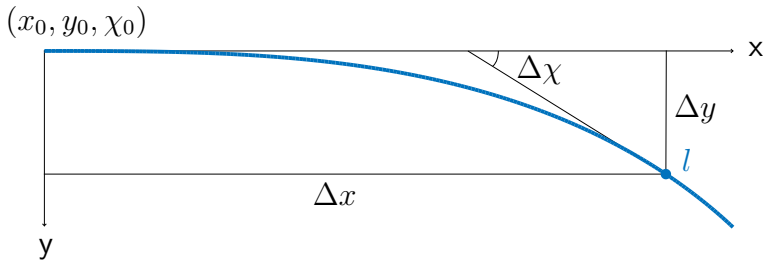


Abbildung 2.4: Beispiel einer Klothoide (blau) mit $c_0 = 0$, $c_1 = -0{,}015$ und $l = 10$.

In der weiteren Arbeit wird die Projektion eines 3D-Punkts in das Kamerabild über die Funktion $\mathrm{proj}(\cdot)$ abgekürzt.

2.4 Klothoiden

Teile dieser Arbeit beinhalten die Schätzung von Straßen- und Kreuzungsverläufen. Diese sind in Deutschland meist nicht willkürlich gebaut, sondern folgen dem sogenannten Klothoidenmodell (vgl. [Dickmanns und Zapp, 1987]). Das Klothoidenmodell ist ein lineares Krümmungsmodell: Die Krümmung, d. h. der inverse Kurvenradius einer Klothoidenkurve, nimmt mit der Länge der Klothoide linear zu oder ab (siehe Abbildung 2.4) – die Krümmungsänderung ist konstant. Entsprechend berechnet sich die Krümmung

$$c(l) = c_0 + c_1 \cdot l \tag{2.12}$$

aus einer Anfangskrümmung c_0 und einer konstanten Krümmungsänderung c_1 über die gesuchte Lauflänge der Klothoide l.

Aus dem Integral der Krümmung berechnet sich die Bahnrichtung

$$\chi = \int c(l)\, dl = \chi_0 + c_0 \cdot l + c_1 \cdot \frac{l^2}{2}\,, \tag{2.13}$$

wobei χ_0 die Anfangsrichtung angibt. Anhand der Bahnrichtung können die Änderungen der x- und y-Koordinaten über die Lauflänge l berechnet werden:

$$\Delta x = \int \cos(\chi)\ dl\,, \qquad \Delta y = \int \sin(\chi)\ dl\,. \tag{2.14}$$

Mit der Kleinwinkelnäherung $\sin(x) \approx x$ und $\cos(x) \approx 1$ können für Bahnrichtungsänderungen $|\Delta\chi| \leq 15°$ die Näherungen

$$\Delta x = l\,, \qquad \Delta y = \int \chi\ dl = \frac{1}{2} c_0 l^2 + \frac{1}{6} c_1 l^3 \tag{2.15}$$

getroffen werden.

Da die in dieser Arbeit verwendeten Krümmungen mit vielen Indizes versehen sind, werden für die bessere Lesbarkeit im Folgenden die Anfangskrümmung vereinfacht als c und die Krümmungsänderung als $\mathring{c}$ angegeben.

2.5 Zustandsschätzung

Die beschriebenen Klothoidenparameter können genau wie alle in dieser Arbeit gesuchten Zustandsvektoren $\mathbf{x}$ nicht direkt gemessen werden. Stattdessen werden Schätzverfahren benötigt, die den Zustand anhand verrauschter Messgrößen $\mathbf{y}$ rekursiv über die Zeit schätzen. Ein dynamisches System wird dazu mittels eines Prozessmodells

$$\dot{\mathbf{x}}(t) = \mathrm{f}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{s}(t)) \tag{2.16}$$

für die Zustandsänderung $\dot{\mathbf{x}}(t)$ zum kontinuierlichen Zeitpunkt t abhängig vom aktuellen Zustandsvektor $\mathbf{x}(t)$, dem Vektor $\mathbf{u}(t)$ der Steuergrößen sowie dem Vektor $\mathbf{s}(t)$ der Störgrößen modelliert. Der Zustandsvektor enthält alle Parameter, die zur vollständigen Beschreibung des Systemzustands benötigt werden. Die Steuergrößen stellen die Eingaben des Nutzers in das System dar – bei einem Fahrzeug beispielsweise Lenkwinkel und Sollgeschwindigkeit. Die Störgrößen modellieren zufällige Störungen im Systemprozess.

Im Zeitdiskreten lässt sich mittels dieses Prozessmodells ausgehend von einem geschätzten Zustandsvektor $\hat{\mathbf{x}}_{k-1}$ eines diskreten Zeitschritts $k-1$ eine Prädiktion

$$\mathbf{x}_k^* = \mathrm{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{v}_{k-1}) \tag{2.17}$$

für den Zustandsvektor des nächsten Zeitschritts k treffen, wobei $\mathbf{v}_{k-1}$ die diskrete Störgröße des Prozessrauschens darstellt. Im Falle des statischen Umfelds vereinfacht sich die Prädiktion je nach Zustandsraum oftmals zur Einheitsfunktion.

Der unbekannte tatsächliche Zustandsvektor $\mathbf{x}_k$ wird gleichzeitig durch Messungen $\mathbf{y}_k$ beobachtet, die jedoch ebenfalls gestört werden. Das Messmodell

$$\mathbf{y}_k = \mathrm{g}(\mathbf{x}_k) + \mathbf{w}_k \tag{2.18}$$

enthält daher neben der Abbildung des Zustandsvektors ein additives Messrauschen $\mathbf{w}_k$. Anhand dieses Messmodells kann aus dem mittels des Prozessmodells prädizierten Zustandsvektor eine Messung $\mathbf{y}_k^*$ prädiziert werden. Die Abweichung dieser prädizierten Messung von einer tatsächlich erhaltenen Messung wird schließlich von Schätzverfahren dazu genutzt, in einem Innovationsschritt die Schätzung des Zustandsvektors zu korrigieren. Dabei ist das Ziel, diesen Schätzfehler zu minimieren.

Kalman-Filter Ein bekanntes Schätzverfahren für lineare Systeme ist das Kalman-Filter [Kálmán, 1960]. Es beschreibt in der Variante für zeitdiskrete Systeme das Prozessmodell als Linearkombination

$$\mathbf{x}_k = \mathbf{\Phi}\mathbf{x}_{k-1} + \mathbf{B}_{k-1}\mathbf{u}_{k-1} + \mathbf{D}_{k-1}\mathbf{v}_{k-1}\,. \tag{2.19}$$

Die Übergangsmatrix $\mathbf{\Phi}$ gibt dabei an, wie sich der Zustandsvektor $\mathbf{x}_k$ ohne äußere Einflüsse aus einem bekannten Zustandsvektor $\mathbf{x}_{k-1}$ des letzten Zeitschritts berechnen würde. Die Eingangsmatrix $\mathbf{B}$ beschreibt den zusätzlichen Einfluss der Steuergrößen $\mathbf{u}$. Mit der Störeingangsmatrix $\mathbf{D}$ und dem Prozessrauschen $\mathbf{v}$ werden schließlich

zufällige Fehler bei diesem Zustandsübergang berücksichtigt. Auch das Messmodell ist eine Linearkombination:

$$\mathbf{y}_k = \mathbf{C}_k \mathbf{x}_k + \mathbf{w}_k \,. \tag{2.20}$$

Die Jacobi-Matrix der Messgleichungen $\mathbf{C}$ bildet den Zustandsvektor linear in den Messraum ab, d. h. sie gibt an, wie sich die erwartete Messung bei einem perfekten Modell aus dem Zustandsvektor berechnen würde. Das Messrauschen fügt dazu Störungen durch Messungenauigkeiten hinzu. Das Prozess- und Messrauschen werden dabei als normalverteilte Prozesse mit einem Erwartungswert von Null und bekannten Kovarianzmatrizen (Erwartungswert von $\mathbf{v}\mathbf{v}^\mathsf{T}$) $\mathbf{Q}$ des Prozessrauschens bzw. $\mathbf{R}$ des Messrauschens angenommen. Zusätzlich zum Zustandsvektor $\mathbf{x}_k$ schätzt das Kalman-Filter die Kovarianzmatrix des Schätzfehlers $\mathbf{P}_k$. Durch stochastische Betrachtungen dieser drei Kovarianzmatrizen wird eine Filtermatrix $\mathbf{K}$ berechnet, mittels der im Innovationsschritt der prädizierte Zustandsvektor $\mathbf{x}_k^*$ zu einem korrigierten Zustandsvektor $\hat{\mathbf{x}}_k$ verbessert wird:

$$\hat{\mathbf{x}}_k = \mathbf{x}_k^* + \mathbf{K}_k \left(\mathbf{y}_k - \mathbf{y}_k^* \right) \,. \tag{2.21}$$

Die Filtermatrix $\mathbf{K}$ gewichtet die Differenz zwischen der prädizierten mittleren Messung $\mathbf{y}_k^* = \mathbf{C}_k \mathbf{x}_k^*$ (aufgrund des mittelwertfreien Messrauschens ohne Rauschterm) und der tatsächlichen Messung anhand der Unsicherheit der Prädiktion relativ zur Messunsicherheit. Übertrifft die Kovarianz des Messrauschens die Kovarianz der Zustandsprädiktion, so wird die Messung weniger stark gewichtet als die Prädiktion. Ist die Kovarianz des Vorhersagefehlers größer als die Messunsicherheit, so wird die Messung stärker gewichtet. Durch die schrittweise, rekursive Korrektur der Zustandsschätzung mit immer neuen Messungen wird das mittelwertfreie Messrauschen über die Zeit heraus gemittelt. Mit steigender Anzahl Messungen sinkt im Normalfall die Kovarianz des Schätzfehlers und damit auch die Kovarianz der Zustandsprädiktion, sodass neue Messungen bei einem eingeschwungenen Kalman-Filter nur schwach gewichtet werden.

Für die Herleitung der Berechnung der Filtermatrix sowie der Prädiktion- und Innovation der Kovarianzen sei auf die Literatur (bspw. [Simon, 2006, S. 123]) verwiesen. Zum Nachschlagen sind die vollständigen Prädiktions- und Innovationsgleichungen dennoch hier gegeben.

Prädiktion von Zeitschritt $k-1$ zu Zeitschritt k:

$$\mathbf{x}_k^* = \mathbf{\Phi}_{k-1} \hat{\mathbf{x}}_{k-1} + \mathbf{B}_{k-1} \mathbf{u}_{k-1} \,, \tag{2.22}$$

$$\mathbf{P}_k^* = \mathbf{\Phi}_{k-1} \hat{\mathbf{P}}_{k-1} \mathbf{\Phi}_{k-1}^\mathsf{T} + \mathbf{D}_{k-1} \mathbf{Q}_{k-1} \mathbf{D}_{k-1}^\mathsf{T} \,. \tag{2.23}$$

Innovation mit Messung $\mathbf{y}_k$:

$$\mathbf{K}_k = \mathbf{P}_k^* \mathbf{C}_k^\mathsf{T} \left(\mathbf{C}_k \mathbf{P}_k^* \mathbf{C}_k^\mathsf{T} + \mathbf{R}_k \right)^{-1} \,, \tag{2.24}$$

$$\hat{\mathbf{x}}_k = \mathbf{x}_k^* + \mathbf{K}_k \left(\mathbf{y}_k - \mathbf{C}_k \mathbf{x}_k^* \right) \,, \tag{2.25}$$

$$\hat{\mathbf{P}}_k = \left(\mathbf{I} - \mathbf{K}_k \mathbf{C}_k \right) \mathbf{P}_k^* \,. \tag{2.26}$$

Erweitertes Kalman-Filter (EKF) Eine wichtige Erweiterung des Kalman-Filters stellt das EKF dar, das das Kalman-Filter für nichtlineare Prozess- und Messmodelle erweitert. Es berechnet die Übergangsmatrix $\mathbf{\Phi}$ und die Jacobi-Matrix der Messgleichungen $\mathbf{C}$ durch eine Linearisierung an der Stelle des Zustandsvektors (und der Eingangsgrößen). Dafür werden die nichtlinearen Funktionen $\mathrm{f}(\cdot)$ und $\mathrm{g}(\cdot)$ partiell nach dem Zustandsvektor $\mathbf{x}$ abgeleitet. In der Praxis erfolgt dies oft numerisch mithilfe des Differenzenquotienten. Die linearisierten Matrizen $\mathbf{\Phi}$ und $\mathbf{C}$ werden für die Berechnung der Kovarianz des Prädiktions- bzw. Schätzfehlers sowie der Filtermatrix genutzt. In der Prädiktion des Zustands und der Messungen werden jedoch die nichtlinearen Funktionen $\mathrm{f}(\cdot)$ und $\mathrm{g}(\cdot)$ verwendet:

$$\mathbf{x}_k^* = \mathrm{f}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}) \ , \tag{2.27}$$

$$\hat{\mathbf{x}}_k = \mathbf{x}_k^* + \mathbf{K}_k \left(\mathbf{y}_k - \mathrm{g}(\mathbf{x}_k^*)\right) \ . \tag{2.28}$$

Die restlichen Berechnungen des Kalman-Filters bleiben unverändert. In Kapitel 9 dieser Arbeit wird das EKF dazu verwendet, den Mittelpunkt einer Kreuzung zu schätzen.

Informationsfilter Eine Variante des Kalman-Filters, die in Arbeiten verwendet wurde, auf denen diese Arbeit aufbaut, ist das Informationsfilter. Dieses modelliert die Unsicherheiten des Schätz- und Messprozesses nicht über die Kovarianzen $\mathbf{P}$ und $\mathbf{R}$, sondern über die inversen Kovarianzen $\mathbf{P}^{-1}$ und $\mathbf{R}^{-1}$, genannt Information. Dies vereinfacht die Innovation zu einer Summe, die in einigen Fällen effizienter zu berechnen ist als die Innovation eines Kalman-Filters [Simon, 2006, S. 157]. Die umgeformten Prädiktions- und Innovationsgleichungen sind beispielsweise in [Simon, 2006, S. 157] oder [Bar-Shalom et al., 2001, S. 303] nachzulesen.

Partikelfilter Ein anderes, in mit dieser Arbeit verwandten Verfahren verwendetes Schätzverfahren ist das Partikelfilter. Das Partikelfilter kann im Gegensatz zum Kalman-Filter auch multimodale Wahrscheinlichkeitsverteilungen schätzen, indem eine Wahrscheinlichkeitsverteilung anhand von Stichproben im Zustandsraum abgebildet wird. Diese Stichproben möglicher Zustandsvektoren werden Partikel genannt. Zur Zustandsschätzung werden die Partikel zunächst entsprechend des Prozessmodells prädiziert. Bei einer neuen Messung wird dann für jedes Partikel eine sogenannte Likelihood berechnet, die aussagt, wie gut das Partikel zu den bisherigen Messungen passt. In der Nähe von Partikeln mit einer hohen Likelihood werden anschließend zusätzliche Partikel gestreut, während Partikel mit einer geringen Likelihood verworfen werden. Dies ersetzt die Innovation. Bei einem eingeschwungenen Partikelfilter markiert die größte Ansammlung an Partikeln den geschätzten Zustandsvektor. Das Aufstellen der Likelihoodfunktion zur Bewertung eines Partikels ersetzt die Formulierung der Messfunktion im Kalman-Filter. Der Nachteil dieses Verfahrens liegt in seiner hohen Rechenkomplexität. In jeder Dimension des Zustandsvektors müssen ausreichend viele Partikel gestreut und ihre Likelihood berechnet werden. Bei höherdimensionalen Zustandsvektoren wird dabei schnell die Grenze der Echtzeitfähigkeit überschritten. Unter dem Namen Monte-Carlo-Filter finden sich weiterführende Informationen in der Literatur, etwa in [Durrant-Whyte und Henderson, 2008, S. 593].

Optimierung In den Kapiteln 3 (Kamera-zu-LiDAR-Kalibrierung), 7 (Terrain-Schätzung) und 8 (Straßen- und Wegetracking) wird ein Optimierer zur Zustandsschätzung eingesetzt. Dabei wird das Schätzproblem als MAP-Problem (Maximum-a-posteriori) formuliert, um denjenigen Zustandsvektor zu finden, der unter Berücksichtigung der A-priori-Verteilung des Vorwissens und der Likelihoods der Messungen die höchste A-posteriori-Wahrscheinlichkeit besitzt. Dieses MAP-Problem wird ähnlich zu einem Kalman-Filter dadurch gelöst, dass die Abweichung eines Zustandsvektors von Vorwissen und Messungen minimiert wird. Dazu werden Vektoren von Residuen $\mathbf{r}(\mathbf{x})$ formuliert, die die Abweichungen eines Zustandsvektors $\mathbf{x}$ zu aufgestellten Bedingungen modellieren. Diese Residuen werden aufsummiert. Da verschiedene Residuen eines Schätzproblems verschiedene Einheiten und Skalierungen haben können, werden die Residuen in ihrer Funktionsdefinition mit Gewichten $\mathbf{w}$ gewichtet:

$$\mathbf{r}(\mathbf{x}) = \mathrm{f}(\mathbf{x}) \cdot \mathbf{w} \,. \tag{2.29}$$

Zusätzlich wird auf jeden Residuumsvektor eine Verlustfunktionen $\rho(\cdot)$ angewandt. Meist kommt in dieser Arbeit die quadratische Verlustfunktion

$$\rho_{\mathsf{T}}(\mathbf{r}) = \mathbf{r}^2 \tag{2.30}$$

zum Einsatz, die der Methode der kleinsten Quadrate folgt. Sollen große Ausreißer weniger stark gewichtet werden, so wird die von Huber [1964] vorgestellte Verlustfunktion

$$\rho_{\mathsf{H}}(\mathbf{r}) = \begin{cases} \mathbf{r}^2 & \mathbf{r} \leq 1 \\ 2\mathbf{r} - 1 & \mathbf{r} > 1 \end{cases} \tag{2.31}$$

verwendet. Da die Huber-Verlustfunktion ab einem Schwellwert von 1 (skalierbar) nicht mehr quadratisch, sondern nur noch linear wächst, steigt die Robustheit gegenüber Ausreißern. Eine noch schwächere Gewichtung von Ausreißern gewährleistet die logarithmische Cauchy-Verlustfunktion

$$\rho_{\mathsf{C}}(\mathbf{r}) = \log(1 + \mathbf{r}^2) \,. \tag{2.32}$$

Ein Optimierer sucht schließlich durch Gradientenabstiegsverfahren denjenigen Zustandsvektor, der die Summe der Residuen $\sum_{\mathbf{r}} \rho(\mathbf{r})$ minimiert. Der geschätzte Zustandsvektor bestimmt sich also als:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_{\mathbf{r}} \rho(\mathbf{r}(\mathbf{x})) \,. \tag{2.33}$$

Teil I

Voraussetzungen

3 Kamera-zu-LiDAR-Kalibrierung

Um Messungen verschiedener Sensoren fusionieren zu können, müssen diese Sensoren zueinander kalibriert werden. Dies umfasst zum einen die zeitliche Kalibrierung, die untersucht, zu welchen unterschiedlichen Zeitpunkten die Sensoren eine Szene aufnehmen, und zum anderen die räumliche Kalibrierung, die ermittelt, von welchen unterschiedlichen Positionen und Winkeln aus die verschiedenen Sensoren eine Szene aufnehmen.

Um den Zeitunterschied zwischen dem Kamerabild und den Messungen des LiDAR-Sensors zu minimieren, wird die Kamera so ausgelöst, dass zur Mitte der Belichtungszeit der rotierende LiDAR-Sensor die Bildmitte erreicht, vgl. [Schneider et al., 2010]. Durch diese Auslösesynchronisierung nehmen beide Sensoren die Szene annähernd zum selben Zeitpunkt auf, sodass keine zeitliche Kalibrierung nötig ist.

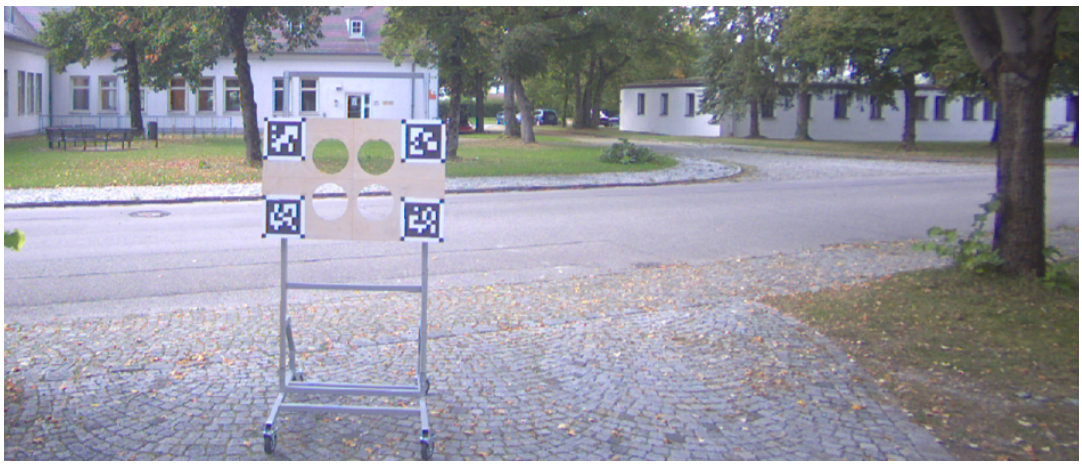
Ein Positions- und Lageunterschied der Sensoren kann jedoch nicht vermieden werden. In Zusammenarbeit mit Jan Kallwies, der in [Kallwies, 2023] die Kamera-zu-Kamera-Kalibrierung behandelt, wurde daher ein gemeinsames Kalibrierungstool entwickelt, in dem generisch die relative Einbaulage mehrerer (drehbarer) Kameras und LiDAR-Sensoren zu einem beliebigen Referenzsensor (Kamera oder LiDAR) sowie das intrinsische Kamera- und Linsenmodell kalibriert werden können. Diese Arbeit fokussiert sich jedoch ausschließlich auf die Kamera-zu-LiDAR-Kalibrierung, also die Bestimmung der relativen Einbaulage einer (festen) Kamera zu einem LiDAR-Sensor. Die intrinsische Kalibrierung wird als bereits bekannt angenommen.

Die Bestimmung der relativen Einbaulage der Sensoren erfolgt über einen Abgleich ihrer Messdaten. Der Messdatenabgleich hat im Gegensatz zur direkten Messung der Sensorposen den Vorteil, dass der Referenzpunkt der Sensoren nicht bekannt sein muss. Außerdem findet die Kalibrierung so in dem Messraum statt, der für die Sensordatenfusion relevant ist. Das Ziel ist es deshalb, die Projektion der LiDAR-Punktwolke in das Kamerabild zu optimieren. Abbildung 3.1 zeigt ein Beispiel.

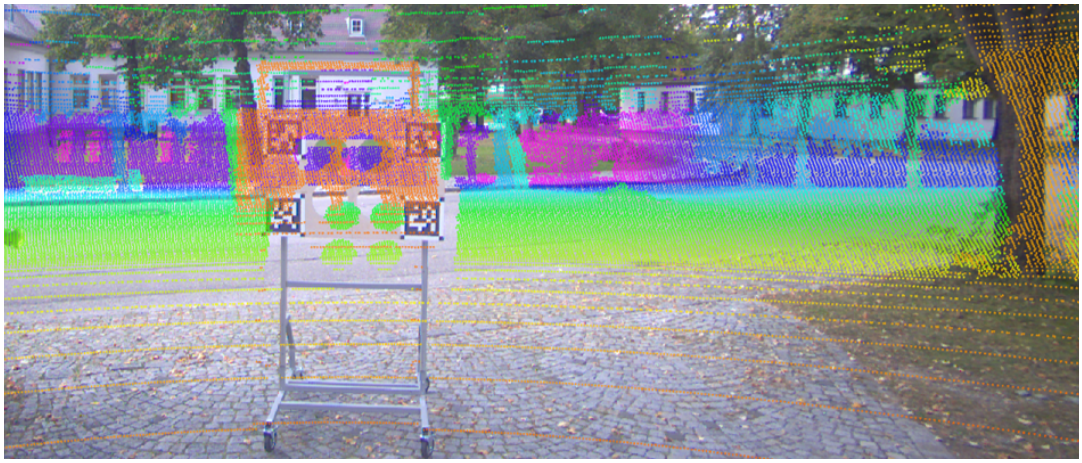
3.1 Kalibrierkörper

Für den Messdatenabgleich wird ein Kalibrierkörper eingesetzt, der Referenzpunkte an exakt bekannten Positionen bietet. Diese werden so gewählt, dass die einzelnen Sensoren sie gut erfassen und die Pose der Referenzpunkte in den jeweiligen Sensorkoordinatensystemen möglichst genau und einfach bestimmt werden kann. Da die Lage der Referenzpunkte auf dem Kalibrierkörper bekannt ist, lässt sich daraus auf die Pose des Kalibrierkörpers in den jeweiligen Sensorkoordinaten schließen. Dadurch können Korrespondenzen zwischen den Sensordaten gebildet werden, auch wenn jeder Sensor unterschiedliche Merkmale des Kalibrierkörpers wahrnimmt.

Für die Kamera-zu-LiDAR-Kalibrierung wird eine Kalibriertafel verwendet, die ursprünglich von Guindel et al. [2017] für die Kalibrierung von Stereokameras zu einem LiDAR-Sensor vorgestellt wurde. Dadurch ist sie so konzipiert, dass sie gut in



(a) Kamerabild der Szene



(b) Mit schlecht kalibrierter Einbaulage in das Kamerabild projizierte LiDAR-Punktwolke



(c) Mit gut kalibrierter Einbaulage in das Kamerabild projizierte LiDAR-Punktwolke

Abbildung 3.1:
Beispiel einer in das in (a) gezeigte Kamerabild projizierten LiDAR-Punktwolke eines Velodyne Alpha Prime (VLS-128) (b) vor und (c) nach einer Neu-Kalibrierung der relativen Einbaulage der Kamera zum LiDAR-Sensor. Die Farbe der LiDAR-Punkte visualisiert ihre x-Koordinate in Sensorkoordinaten (von nahen Punkten in orange zu entfernten Punkten in lila). Die unteren und oberen Laserstrahlen des Velodyne Alpha Prime haben eine größere vertikale Spreizung, weshalb die Punktwolke nach oben und unten dünner wird. Abgebildet ist die vorgestellte Kalibriertafel auf einem rollbaren, höhenverstellbaren Ständer.

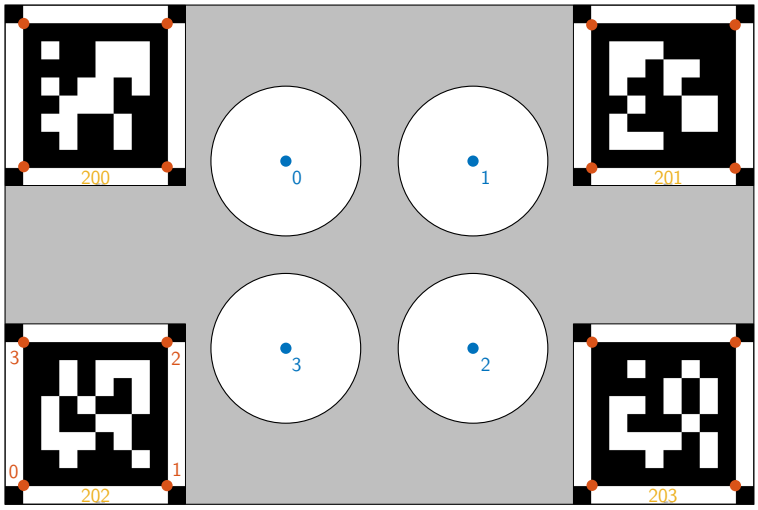


Abbildung 3.2:
Schematischer Aufbau des verwendeten Kalibrierkörpers mit Löchern zur Detektion in der LiDAR-Punktwolke (blaue Referenzpunkte) und AprilTags zur Detektion im Kamerabild (rote Referenzpunkte). Die Kalibriertafel ist 1,2 m × 0,8 m groß. Die Löcher haben einen Durchmesser von 24 cm und ihre Mittelpunkte sind jeweils 15 cm in x- und y- Richtung vom Mittelpunkt der Kalibriertafel versetzt. Die AprilTags der Familie 36h11 haben (ohne die Schachbrett-Ecken) eine Kantenlänge von 22,8 cm. Die IDs sind farbig angegeben.

Tiefendaten detektierbar ist. Damit die Kalibriertafel auch im Kamerabild präzise lokalisiert werden kann, wurde sie im Rahmen dieser Arbeit um künstliche Marker erweitert. Dies wurde inzwischen durch Beltrán et al. [2022] ähnlich vorgeschlagen.

Messung in der LiDAR-Punktwolke Die Kalibriertafel besteht aus einer rechteckigen Platte mit vier großen Löchern entsprechend des in Abbildung 3.2 gezeigten Schemas. Die Löcher dienen als Referenzpunkte für den LiDAR-Sensor. Diese sind (im Gegensatz zu beispielsweise den Ecken der Kalibriertafel) auch durch Sensoren oder in Messbereichen mit einem großen vertikalen Abstand zwischen den gemessenen LiDAR-Punkten präzise detektierbar. Das Verfahren von Guindel et al. [2017] weiterentwickelnd, werden aus der gesamten LiDAR-Punktwolke mittels des RANSAC-Verfahrens (*Random Sample Consensus*) [Fischler und Bolles, 1981] Ebenen segmentiert, in denen nach einer Fläche in der Größe des Kalibrierkörpers gesucht wird. In die Kantenpunkte dieser segmentierten Fläche werden schließlich wieder mittels RANSAC Kreise in der den Löchern des Kalibrierkörpers entsprechenden Größe gefittet. Werden vier Kreise in passender Anordnung gefunden, so dienen deren vier Mittelpunkte $h \in [0;3]$ als Messungen ${}^{\text{lidar}}\mathbf{y}_{\text{hole},h} = [x\ y\ z]^{\mathsf{T}}$ des Kalibrierkörpers in der LiDAR-Punktwolke.

Messung im Kamerabild Um die Pose des Kalibrierkörpers auch im Kamerabild möglichst exakt messen zu können, wurden in den Ecken der Kalibriertafel vier sog. AprilTags [Olson, 2011, Krogius et al., 2019] angebracht. Hierbei handelt es sich um optische Referenzpunkte in Form eines quadratischen Schwarz-Weiß-Musters, die aufgrund des bekannten Musters sehr präzise im Bild lokalisiert werden können. Ein

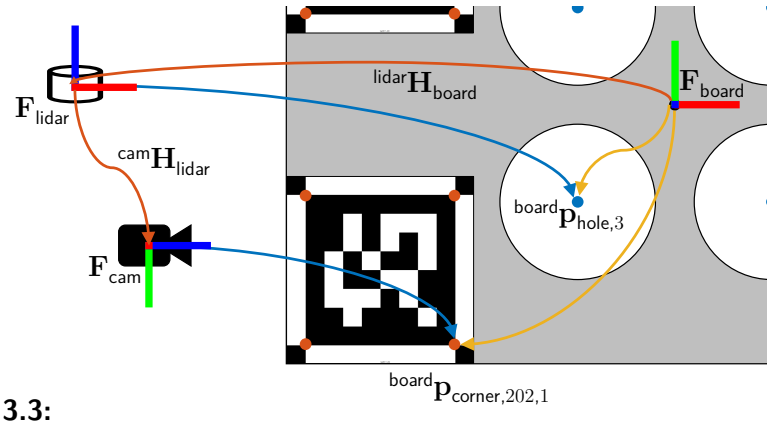


Abbildung 3.3:
Visualisierung der Transformationskette zur Kalibrierung der relativen Einbaulage einer statischen Kamera zu einem LiDAR-Sensor. Gesucht werden die roten Transformationen. Die gelben Positionen sind exakt bekannt. Die blauen Posen werden gemessen. Aus der Transformationskette lassen sich damit die roten Transformationen schätzen.

Softwarepaket dazu wurde in [Kallwies, Forkel, und Wuensche, 2020b] veröffentlicht. Abbildung 3.2 zeigt die Kalibriertafel mit den AprilTags. Die vier erkannten Ecken $c \in [0; 3]$ der vier AprilTags $t \in [200; 203]$ dienen als Messungen ${}^{\text{cam}}\mathbf{y}_{\text{corner},t,c} = [u\ v]^{\mathsf{T}}$ des Kalibrierkörpers im Kamerabild.

3.2 Optimierung der relativen Einbaulage

Für die Kalibrierung wird der Kalibrierkörper an verschiedenen Posen aufgestellt und es wird jeweils eine (über mehrere Zeitschritte gemittelte) Messung in der LiDAR-Punktwolke und dem Kamerabild durchgeführt. Mit dem entwickelten Kalibrierungstool können mehrere Kameras (und LiDAR-Sensoren) gleichzeitig kalibriert werden sowie mehrere Kalibriertafeln verwendet werden. Hier wird jedoch der Einfachheit halber lediglich die Kalibrierung einer Kamera mit einer Kalibriertafel betrachtet.

Aus den Messungen der Loch-Mittelpunkte durch den LiDAR-Sensor wird die jeweilige Pose des Kalibrierkörpers im LiDAR-Koordinatensystem bestimmt. Über die gesuchte, zunächst nur grob initialisierte Transformation ${}^{\text{cam}}\mathbf{H}_{\text{lidar}}$ und die bekannten Posen der AprilTags relativ zu den Löchern der Kalibriertafel kann daraus die Position der AprilTags im Kamerabild prädiziert werden. Die Abweichung dieser anhand der LiDAR-Messungen und der in Abbildung 3.3 visualisierten Transformationskette prädizierten Position der Ecken der AprilTags zu den tatsächlich gemessenen Eckpunkten – und damit den Fehler in ${}^{\text{cam}}\mathbf{H}_{\text{lidar}}$ – gilt es in der Kalibrierung zu minimieren.

Die Minimierung dieses Projektionsfehlers erfolgt mittels nichtlinearer Optimierung durch den *Ceres Solver* [Agarwal et al., 2020]. Der gesuchte Zustandsvektor enthält die relative Einbaulage der Kamera zum LiDAR-Sensor, modelliert durch die Position ${}^{\text{cam}}\mathbf{p}_{\text{lidar}}$ und die Quaternion ${}^{\text{cam}}\mathbf{q}_{\text{lidar}}$ sowie alle Posen $({}^{\text{lidar}}\mathbf{p}_{\text{board},k}, {}^{\text{lidar}}\mathbf{q}_{\text{board},k})$ des Kalibrierkörpers zu den einzelnen Zeitschritten k. Die einzelnen Posen der Kalibriertafel werden zwar durch die LiDAR-Messungen initialisiert, da diese jedoch auch Messfehler aufweisen, sind die exakten Posen aber dennoch Unbekannte.

Daher werden zwei Arten von Residuen minimiert:

Projektionsfehler der AprilTag-Ecken im Kamerabild Das Residuum

$$\mathbf{r}_{\text{corner},t,c,k} = \left({}^{\text{cam}}\mathbf{y}^{*}_{\text{corner},t,c,k} - {}^{\text{cam}}\mathbf{y}_{\text{corner},t,c,k}\right) \cdot \frac{1}{\sigma_{\text{corner}}} \tag{3.1}$$

modelliert den Fehler der aus der geschätzten Pose der Kalibriertafel im Zeitschritt k prädizierten Pixel-Position ${}^{\text{cam}}\mathbf{y}^{*}_{\text{corner},t,c,k}$ der Ecke c eines AprilTags t im Kamerabild zu der tatsächlich im Zeitschritt k durch die Kamera gemessenen Eckposition ${}^{\text{cam}}\mathbf{y}_{\text{corner},t,c,k}$. σ_{corner} stellt dabei die geschätzte Standardabweichung des Messrauschens der AprilTag-Eck-Messungen im Kamerabild dar und wird entsprechend der Evaluationen zur Genauigkeit von AprilTag-Detektionen in [Kallwies, Forkel, und Wuensche, 2020b] auf $\sigma_{\text{corner}} = 0{,}2\,\text{px}$ gesetzt. Diese Auswertung des Residuums in Bildpixeln entspricht der durch Himmelsbach et al. [2011] bzw. Schneider [2023] beschriebenen Punkt-zu-Strahl-Metrik. Die Prädiktion einer AprilTag-Ecke c im Kamerabild erfolgt durch die Projektion ($\text{proj}(\cdot)$, vgl. Abschnitt 2.3) der prädizierten Eckposition im Kamerakoordinatensystem ins Kamerabild. Die prädizierte Eckposition im Kamerakoordinatensystem berechnet sich aus der bekannten Position ${}^{\text{board}}\mathbf{p}_{\text{corner},t,c}$ der Ecke relativ zur Kalibriertafel, der geschätzten Pose ${}^{\text{lidar}}\hat{\mathbf{H}}_{\text{board},k}$ der Kalibriertafel im LiDAR-Koordinatensystem zum Zeitschritt k und der ebenfalls geschätzten Transformation ${}^{\text{cam}}\hat{\mathbf{H}}_{\text{lidar}}$ zwischen dem LiDAR-Koordinatensystem und dem Kamerakoordinatensystem:

$${}^{\text{img}}\mathbf{y}^{*}_{\text{corner},t,c,k} = {}^{\text{img}}\,\text{proj}\Big({}^{\text{cam}}\hat{\mathbf{H}}_{\text{lidar}} \cdot {}^{\text{lidar}}\,\hat{\mathbf{H}}_{\text{board},k} \cdot {}^{\text{board}}\mathbf{p}_{\text{corner},t,c}\Big)\ . \tag{3.2}$$

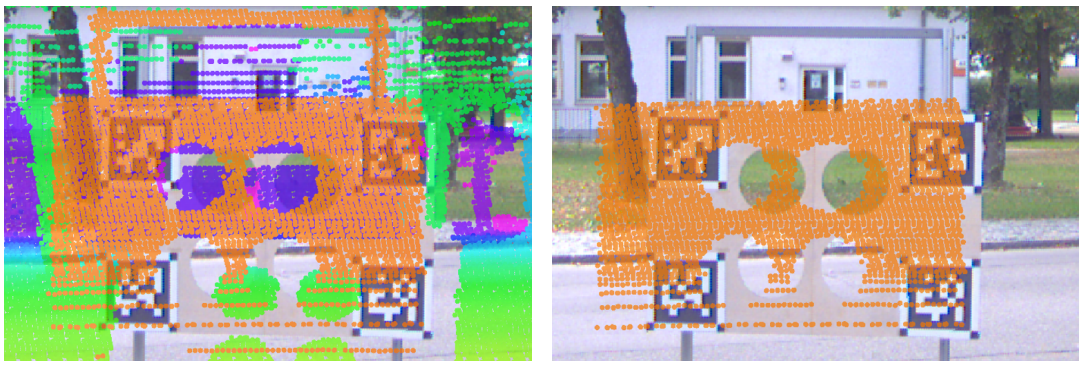
Bei einer Kamera auf einer rotierbaren Plattform, die zum Zeitpunkt der Messung um den bekannten Winkel $\Psi_{\text{platf.}}$ gedreht ist, wird die Transformation ${}^{\text{cam}}\hat{\mathbf{H}}_{\text{lidar}}$ aufgeteilt in ${}^{\text{cam}}\hat{\mathbf{H}}_{\text{platf.}} \cdot \Psi_{\text{platf.}} \cdot {}^{\text{platf.}}\hat{\mathbf{H}}_{\text{lidar}}$.

Messfehler der Kalibriertafel-Löcher in der LiDAR-Punktwolke Das Residuum

$$\mathbf{r}_{\text{hole},h,k} = \left({}^{\text{lidar}}\mathbf{y}^{*}_{\text{hole},h,k} - {}^{\text{lidar}}\mathbf{y}_{\text{hole},h,k}\right) \cdot \frac{1}{\sigma_{\text{hole}}} \tag{3.3}$$

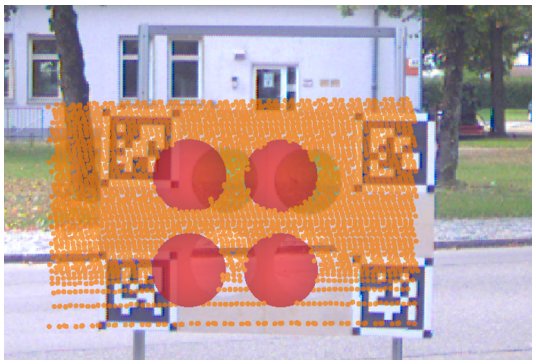
modelliert die Abweichung der aus der geschätzten Pose der Kalibriertafel im Zeitschritt k prädizierten 3D-Position ${}^{\text{lidar}}\mathbf{y}^{*}_{\text{hole},h,k}$ eines Loch-Mittelpunkts h zu dem tatsächlich im Zeitschritt k durch den LiDAR-Sensor gemessenen Loch-Mittelpunkt ${}^{\text{lidar}}\mathbf{y}_{\text{hole},h,k}$. $\sigma_{\text{hole}} = 0{,}01\,\text{m}$ stellt dabei die geschätzte Standardabweichung des Messrauschens der Loch-Mittelpunkt-Messungen durch den LiDAR-Sensor dar. Der prädizierte Loch-Mittelpunkt berechnet sich aus der bekannten Position ${}^{\text{board}}\mathbf{p}_{\text{hole},h}$ der Löcher auf der Kalibriertafel und der geschätzten Pose ${}^{\text{lidar}}\hat{\mathbf{H}}_{\text{board}}$ der Kalibriertafel im LiDAR-Koordinatensystem zum Zeitschritt k:

$${}^{\text{lidar}}\mathbf{y}^{*}_{\text{hole},h,k} = {}^{\text{lidar}}\,\hat{\mathbf{H}}_{\text{board},k} \cdot {}^{\text{board}}\mathbf{p}_{\text{hole},h}\ . \tag{3.4}$$

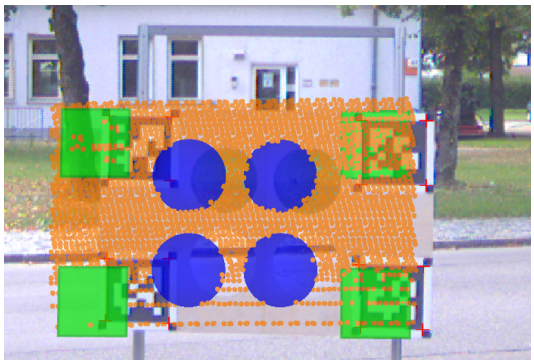


(a) Falsch kalibrierte Eingangsdaten

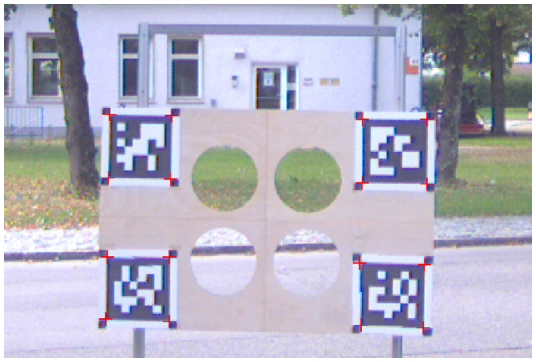
(b) Aus (a) segmentierter Kalibrierkörper



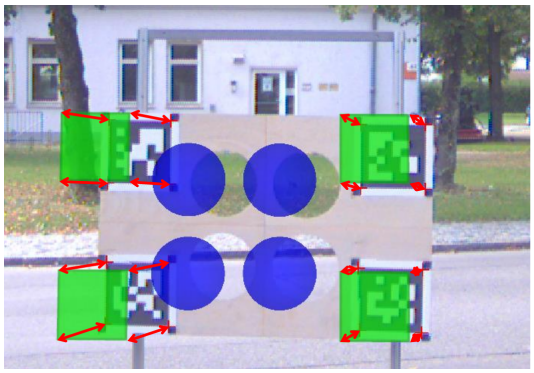
(c) In (b) detektierte Löcher



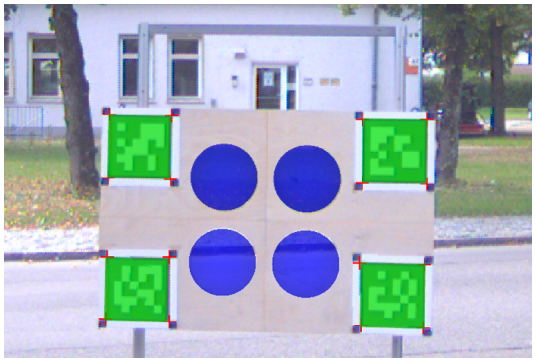
(d) Anhand (c) gemessene Boardpose



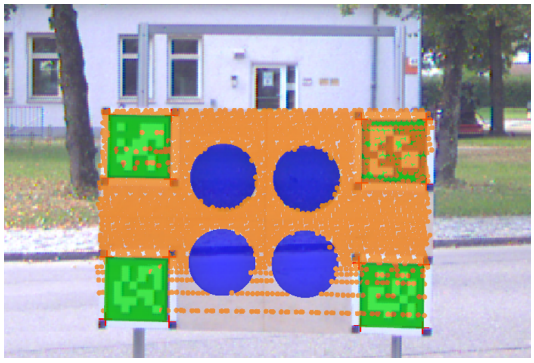
(e) Detektierte AprilTags



(f) Minimierung des Projektionsfehlers



(g) Korrigierte Projektion der Boardpose



(h) Neu kalibrierte Einbaulage

Abbildung 3.4:
Visualisierung der einzelnen Schritte zur Kalibrierung der relativen Einbaulage einer Kamera zu einem LiDAR-Sensor (vgl. Abbildung 3.1) anhand eines Kalibrierkörpers mit durch den LiDAR-Sensor detektierten Löchern (rote Kreise) und durch die Kamera detektierten AprilTags (rote Kreuze). Die blauen Kreise und die grünen Quadrate zeigen die geschätzten Posen der Löcher und der AprilTags anhand der gemessenen Boardpose und des Modells.

Mittels eines Optimierungsverfahrens wird schließlich der geschätzte Zustandsvektor $\hat{\mathbf{x}}$ als derjenige Zustandsvektor $\mathbf{x}$ bestimmt, für den die Summe aller quadrierten Residuen aller Messungen zu allen Zeitschritten minimal wird:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_{k} \left(\sum_{t} \sum_{c} \mathbf{r}^2_{\text{corner},t,c,k} + \sum_{h} \mathbf{r}^2_{\text{hole},h,k} \right) . \tag{3.5}$$

Abbildung 3.4 fasst das Verfahren zur Schätzung der relativen Einbaulage einer Kamera zu einem LiDAR-Sensor an einem Beispiel zusammen.

3.3 Ergebnisse

Zur Bewertung des entwickelten Kalibrierverfahrens soll dessen Wiederholgenauigkeit untersucht werden. Dafür wurden ausgehend von derselben Initialisierung fünf unabhängige Kalibrierungsdurchgänge für eine unbewegte Kamera durchgeführt. Dazu wurde die Kalibriertafel an jeweils 15 unterschiedlichen Posen gemessen. Abbildung 3.5 zeigt die einzelnen Kalibriertafel-Posen dieser fünf Durchgänge. In den ersten drei Durchgängen (a) bis (c) wurden möglichst ähnliche Posen gewählt: In drei Kreisbögen um das Fahrzeug herum in etwa 5 m, 10 m und 15 m Entfernung vom LiDAR-Sensor wurden je fünf Posen aufgenommen. In den Durchgängen (d) und (e) wurden die Posen des Kalibrierkörpers dagegen willkürlich vom Nutzer ausgewählt. Dabei enthält Durchgang (d) auch Messungen in mehr als 20 m Entfernung.

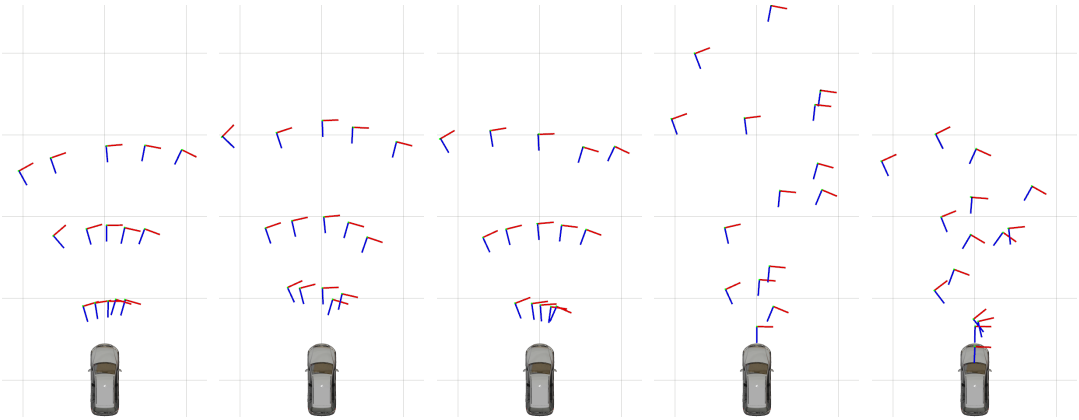


Abbildung 3.5:
Aufgenommene Posen des Kalibrierkörpers bei fünf unterschiedlichen Kalibrierungsdurchgängen. Das Grid im Hintergrund hat eine Zell-Kantenlänge von 5 m.

Tabelle 3.1 zeigt die Kalibrierungsergebnisse der fünf Kalibrierungsdurchgänge mit den je 15 Messungen sowie das Ergebnis einer Kalibrierung mit allen 75 Messungen. Die maximale Abweichung tritt zwischen den Ergebnissen der Durchgänge (b) und (d) auf: Dort unterscheiden sich die geschätzten relativen Einbaulagen um 1,14 cm und 0,1°. Dies liegt auch daran, dass der Kalibrierkörper bei den höheren Entfernungen in Durchgang (d) weniger präzise erkannt werden konnte.

Tabelle 3.1:
Geschätzte, relative Einbaulage der zu kalibrierenden Kamera zum LiDAR-Sensor unter Verwendung der jeweils in den Abbildungen 3.5a bis 3.5e gezeigten 15 Messungen sowie aller 75 Messungen. Zur besseren Einordnung sind die angegebenen Rotationswinkel dabei um die durch die unterschiedlichen Achsenrichtungen der LiDAR- und Kamerakoordinatensysteme verursachten Rotationen bereinigt – es wird für beide Sensoren die Fahrzeugkonvention (x-Achse entlang des Fahrzeugs, y-Achse nach links, z-Achse nach oben) angenommen.

	Position in m			Lage in °		
	x	y	z	Roll Φ	Pitch Θ	Yaw Ψ
(a)	0,9069	0,1106	−0,7450	−0,2050	4,0332	3,6745
(b)	0,9075	0,1058	−0,7460	−0,2770	4,0248	3,7030
(c)	0,9049	0,1142	−0,7421	−0,2035	4,0621	3,6524
(d)	0,9029	0,1126	−0,7380	−0,2066	4,1017	3,6825
(e)	0,9057	0,1081	−0,7455	−0,2028	4,0477	3,7000
Alle	0,9057	0,1101	−0,7433	−0,2208	4,0531	3,6826

Tabelle 3.2:
Abweichung der in den fünf Kalibrierungsdurchgängen geschätzten relativen Einbaulagen der zu kalibrierenden Kamera zum LiDAR-Sensor (Tabelle 3.1 (a)–(e)) zu der aus allen 75 Messungen geschätzten relativen Einbaulage (Tabelle 3.1 Alle).

	(a)	(b)	(c)	(d)	(e)
Abstand in m	0,0052	0,0053	0,0044	0,0065	0,0029
Winkel in °	0,0185	0,0674	0,0370	0,0506	0,0247

Im Vergleich mit der Kalibrierung mit allen 75 Messungen (Tabelle 3.2) ergibt sich für Durchgang (d) eine Abweichung der geschätzten Sensorpose von 0,7 cm und 0,05°. Die willkürliche Kalibrierung mit der teils schlecht lokalisierten Kalibriertafel wirkt sich also selbst in 50 m Entfernung (was der maximalen Vorausschau im entwickelten Straßentracking entspricht) nur als ein Versatz von etwa 5 cm aus. Das beste Ergebnis wird in Durchgang (a) erreicht: Hier beträgt die Abweichung zu der ausführlichen Kalibrierung auf 50 m Entfernung nur rund 2 cm. Diese Wiederholgenauigkeit auch mit weniger Messungen lässt auf eine ausreichend hohe Genauigkeit des entwickelten Kalibrierverfahrens schließen.

Damit ermöglicht diese Kalibrierung die Sensordatenfusion von Kamera und LiDAR. Sowohl der Kalibrierkörper als auch das Kalibrierungstool könnten darüber hinaus jedoch auch einfach für weitere Sensoren erweitert werden. So könnte beispielsweise mithilfe eines Radarreflektors oder eines Dopplersimulators die Pose der Kalibriertafel relativ zu einem Radar-Sensor (*Radio Detection And Ranging*) bestimmt werden.

Teil II

Fusion auf Rohdatenebene

Nach erfolgter Kalibrierung werden in den folgenden Kapiteln nun zwei Möglichkeiten für die Fusion von Kamera und LiDAR auf Rohdatenebene untersucht:

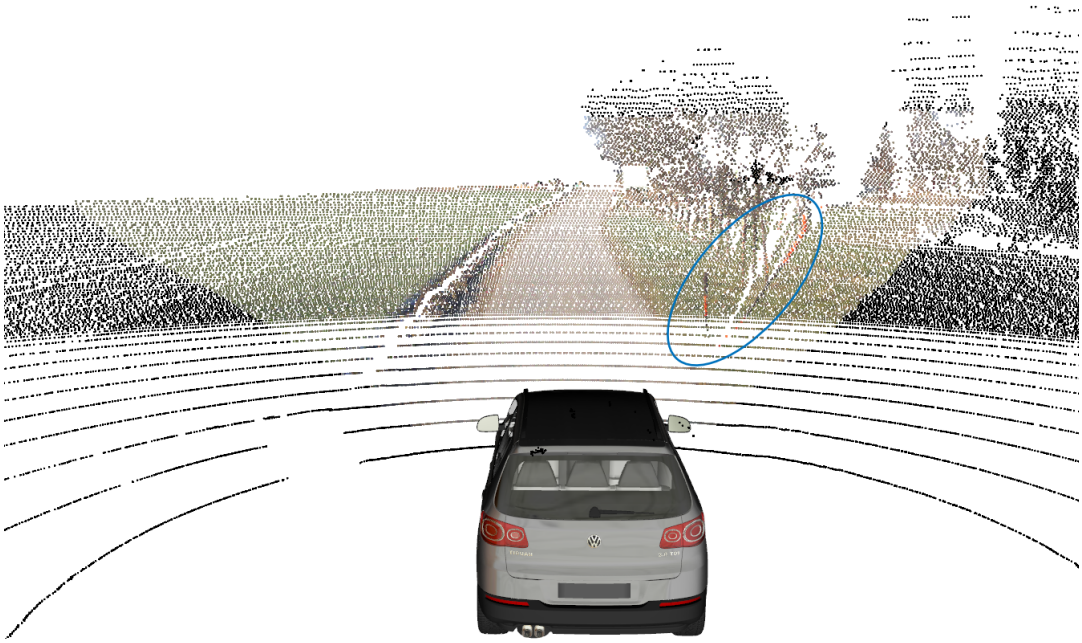
- Einfärben der LiDAR-Punktwolke mit dem Kamerabild (Kapitel 4)
- Fusion der LiDAR-Punktwolke in das Stereo Matching (Kapitel 5)

4 Eingefärbte LiDAR-Punktwolke

Eine Möglichkeit der Rohdatenfusion von Kamera und LiDAR wurde von Schneider et al. [2010] vorgeschlagen: die LiDAR-Punktwolke mit dem Kamerabild einzufärben. Hierbei werden den Punkten der LiDAR-Punktwolke, die im Sichtbereich einer Kamera liegen, Farbwerte zugeordnet. Dazu werden sie in das Kamerabild projiziert und erhalten die Farbe des entsprechenden Pixels. Auf diese Art und Weise wird die komplette 3D-Information der LiDAR-Punktwolke beibehalten und um Farbinformationen aus dem Kamerabild ergänzt. Abbildung 4.1 zeigt ein Beispiel.



(a) Kamerabild der Szene



(b) Mit dem Kamerabild eingefärbte LiDAR-Punktwolke

Abbildung 4.1:
Beispiel einer mit einem Kamerabild durch Projektion der einzelnen Punkte in das Kamerabild eingefärbten LiDAR-Punktwolke eines Velodyne Alpha Prime (VLS-128). Punkte außerhalb des Sichtbereichs der Kamera sind schwarz eingefärbt. Blau eingekreist ist ein Beispiel fehlerhafter Einfärbung durch fehlende Verdeckungsbehandlung.

Beim Einfärben der Punktwolke muss beachtet werden, dass die Objektverdeckung zwischen den beiden Sensoren abweicht: Da die zwei Sensoren an unterschiedlichen Positionen am Fahrzeug montiert sind und die Szene aus unterschiedlichen Blickwinkeln erfassen, verdeckt dasselbe Objekt in den beiden Messungen zwei verschiedene Bereiche. So verdeckt im Beispiel in Abbildung 4.1 der orange-schwarze Straßenpfosten für den LiDAR-Sensor einen schmalen Streifen Gras, in dem keine Punkte gemessen werden können, sodass die LiDAR-Punktwolke an dieser Stelle eine kleine (weiß illustrierte) Lücke aufweist. Im Kamerabild verdeckt der Straßenpfosten dagegen einen anderen Teil des Grases. Ohne eine explizite Verdeckungsbehandlung wird dieser schmale Streifen Gras schräg rechts neben dem Straßenpfosten fälschlicherweise orange-schwarz eingefärbt. Dies liegt daran, dass hier zwar LiDAR-Punkte auf dem Gras gemessen werden, das entsprechende Gras im Kamerabild jedoch durch den Pfosten verdeckt ist, weshalb die Projektion der Punkte in das Kamerabild genau auf dem Straßenpfosten liegt. Schneider et al. [2010] bestimmen daher vor dem Einfärben die Sichtbarkeit von LiDAR-Punkten im Kamerabild. Dafür werden die Ergebnisse eines Punktwolken-Clusterings von Himmelsbach et al. [2010] verwendet. Punkte, deren Projektion in das Kamerabild innerhalb der konvexen Hülle eines in das Kamerabild projizierten Objekt-Clusters liegt, werden von der Einfärbung ausgeschlossen. Infolgedessen, und da meist nicht das gesamte Blickfeld des LiDAR-Sensors durch Kameras abgedeckt ist, gibt es in der eingefärbten Punktwolke jedoch sowohl Punkte mit als auch Punkte ohne Farbinformationen.

Eine weitere Einschränkung der eingefärbten Punktwolke ist ihre im Vergleich zum Kamerabild signifikant reduzierte Auflösung, insbesondere im Nahbereich. Die vertikale Auflösung der Punktwolke hängt stark vom verwendeten LiDAR-Sensor ab. Im Beispiel wird die Punktwolke eines hochauflösenden Velodyne Alpha Prime mit 128 Strahlen gezeigt. Dieser misst im mittleren Sichtbereich zwar eine im Vergleich zu anderen Sensoren äußerst dichte Punktwolke, nach oben und unten hin sinkt die Auflösung jedoch stark. Während die maximale vertikale Winkelauflösung des Velodyne Alpha Prime bei 0,11° liegt (bei den mittleren Strahlen), wird zwischen den untersten beiden Laserstrahlen lediglich eine minimale Winkelauflösung von 5,5° erreicht [Velodyne LiDAR, Inc., 2018]. Doch auch die maximale vertikale Winkelauflösung reicht nicht an die Winkelauflösung der Kamera von 0,02° heran (vgl. Abschnitt 2.1). Die vertikale Winkelauflösung des Velodyne HDL-64E S2 liegt zwischen 0,33° und 0,5° [Velodyne LiDAR, Inc., 2011]. Auch horizontal liegt die Winkelauflösung der beiden Sensoren bei einer Rotationsfrequenz von 10 Hz nur bei etwa 0,2°. Damit kann die eingefärbte Punktwolke nicht die Auflösung des Kamerabilds erreichen.

Schließlich ist auch die Vorausschau der (eingefärbten) LiDAR-Punktwolke deutlich geringer als die eines Kamerabilds. So ist im Beispiel in Abbildung 4.1 im Kamerabild in etwa 150 m Entfernung ein Haus zu erkennen, das der LiDAR-Sensor jedoch nicht mehr erfasst. Wird nur noch die eingefärbte LiDAR-Punktwolke verwendet, so geht in der Fusion also die Vorausschau des Kamerabilds verloren.

Da insbesondere im Wegetracking sowohl eine hohe Auflösung als auch eine gute Vorausschau benötigt werden, werden zur Umgebungswahrnehmung auf Feldwegen im Folgenden andere Verfahren zur Fusion von Kamera und LiDAR untersucht.

5 Fusion von Stereo und LiDAR

Falls ein Stereokamerasetup zur Verfügung steht, liegt eine andere Möglichkeit Kamerabilder und LiDAR-Punkte auf Rohdatenebene zu fusionieren darin, die LiDAR-Punkte in das Stereosehen zu fusionieren. Dafür wurde im Rahmen dieser Arbeit ein neues Verfahren entwickelt und in [Forkel und Wuensche, 2023] vorab veröffentlicht.

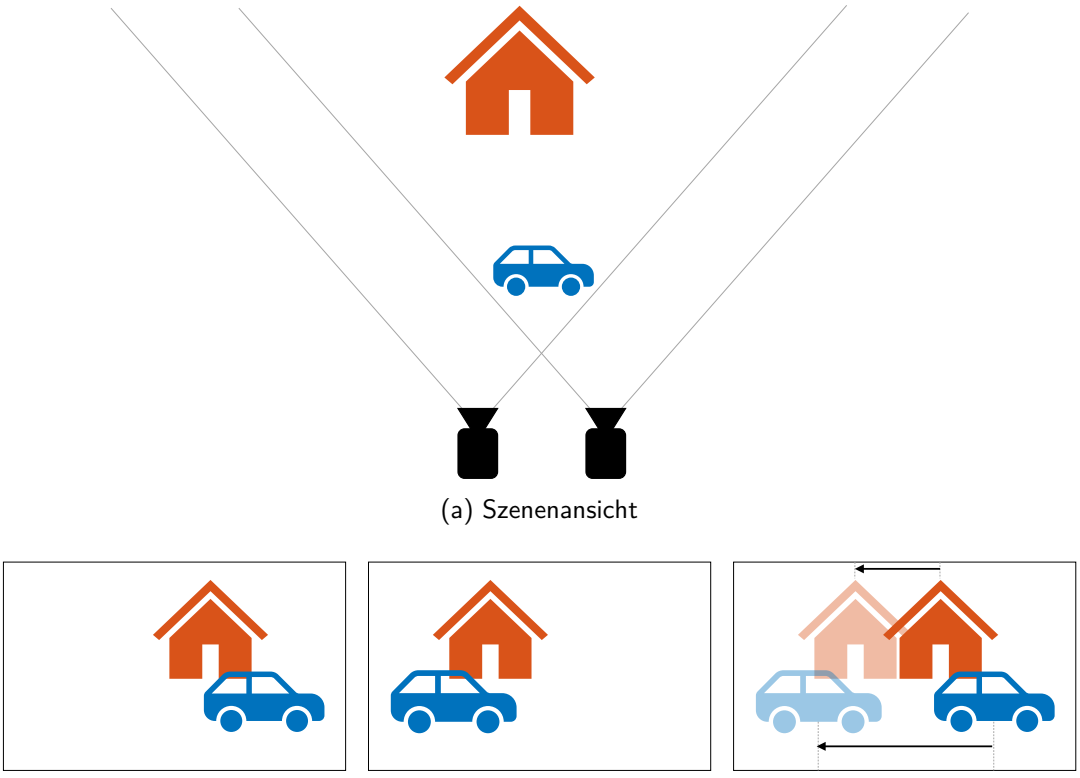
5.1 Einleitung

Stereo Vision Durch Stereosehen können aus Kameras nicht nur Farb-, sondern auch Tiefeninformationen gewonnen werden, indem dieselbe Szene von zwei leicht versetzten Kameras aufgenommen wird. Die Kameras sollten dabei nur entlang einer Achse versetzt sein und die Bildebenen möglichst präzise in einer Ebene liegen. Kleine Abweichungen in der Lage können durch eine sog. Rektifizierung ausgeglichen werden (vgl. [Steger et al., 2018, S. 247]). Typischerweise wird ein horizontales Kamera-Paar bestehend aus einer linken Referenzkamera und einer horizontal versetzten rechten Kamera verwendet (siehe auch dem Sensoraufbau von MuCAR-4 in Abschnitt 2.1).

In den Bildern dieser zwei Kameras werden anschließend Korrespondenzen gesucht, d. h. für jedes Pixel des linken Referenzbilds wird das entsprechende Pixel im rechten Bild gesucht, das denselben Welt-Punkt darstellt. Daraus wird für jedes Pixel eine Disparität berechnet, die angibt, um wie viele Pixel der abgebildete Punkt zwischen den beiden Bildern springt. Je näher ein Objekt der Kamera ist, desto höher ist seine Disparität. Ein Beispiel dafür ist in Abbildung 5.1 zu sehen. Ähnlich der menschlichen Daumensprung-Methode für das grobe Schätzen von Entfernungen kann aus der gemessenen Disparität bei bekannter Basisbreite (Abstand der beiden Kameras) die Entfernung eines Punkts abgeleitet werden.

Auf diese Art und Weise kann für viele Pixel eines Kamerabilds eine Tiefeninformation berechnet werden. Das Ergebnis ist eine dichte Punktwolke mit – aufgrund der hohen Winkelauflösung moderner Kameras – sehr hoher örtlicher Auflösung. Ein Beispiel einer solchen Stereo-Punktwolke ist in Abbildung 5.2a zu sehen. Die Fehler in der geschätzten Tiefe steigen jedoch quadratisch mit dem Abstand eines Punkts zur Kamera [Gallup et al., 2008], sodass die erhaltenen Tiefenpunkte oft ungenau sind und die Entfernungsschätzung rauscht. Weiterhin kann bei Objekten mit (in horizontaler Richtung) wenig oder periodischer Textur, die die Suche nach Korrespondenzen erschwert, oft keine Disparität und damit keine Tiefe gemessen werden.

Stereo-LiDAR-Fusion Um auch mögliche vertikale Texturen zur Korrespondenzsuche nutzen zu können und die Erkennung horizontaler Strukturen zu verbessern, wurde in [Kallwies, Engler, Forkel, und Wuensche, 2020a] zunächst eine dritte, vertikal versetzte Kamera hinzugefügt. Dies ist allerdings ein eher ungewöhnliches Setup und oft schwer auf dem Versuchsträger umzusetzen. Die meisten autonomen Fahrzeuge verfügen jedoch über einen LiDAR-Sensor. Daher wird nun untersucht, anstatt einer dritten Kamera diesen LiDAR-Sensor in das Stereo Matching zu fusionieren. Er kann aufgrund des abweichenden Messprinzips die Tiefe unmittelbar



(a) Szenenansicht

(b) Linkes Kamerabild (c) Rechtes Kamerabild (d) Disparitäten

Abbildung 5.1:
Exemplarisches Beispiel für die Messung einer Disparität mit zwei horizontal versetzten Kameras. Die Abbildung ist angelehnt an [Dumont, 2015, Figure 2.5].



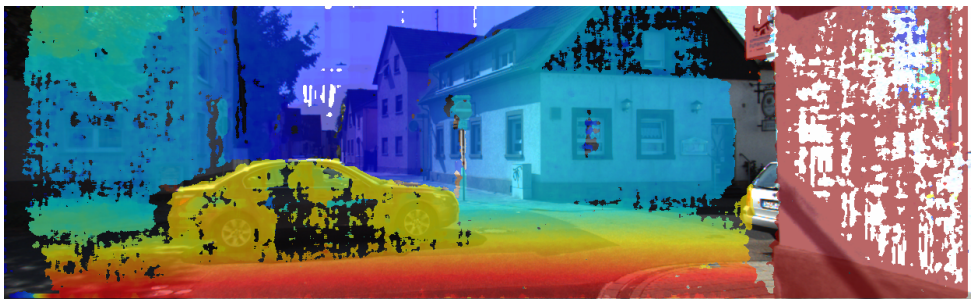
(a) Stereo-Punktwolke (b) LiDAR-Punktwolke

Abbildung 5.2:
Beispiel einer aus Stereo Matching berechneten Punktwolke im Vergleich zu einer von einem LiDAR-Sensor gemessenen Punktwolke derselben Szene. Die Stereo-Punktwolke ist aufgrund der geringen Textur auf dem Auto lückenhaft, während die LiDAR-Punktwolke so dünn ist, dass ihre Einfärbung anhand des Kamerabilds optisch bereits nicht mehr zu erkennen ist. Die Szene zeigt Realdaten aus dem KITTI-Datensatz [Geiger et al., 2013].

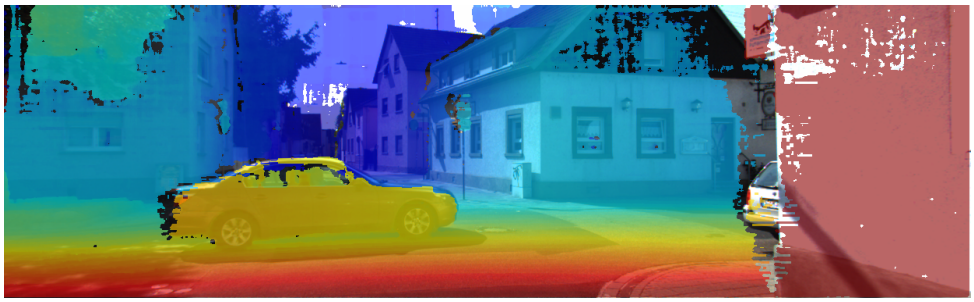
messen und damit genauer und annähernd unabhängig von der Textur eines Objekts bestimmen. Dafür sind die Messpunkte – wie bereits in Kapitel 4 betrachtet wurde – deutlich dünner verteilt, und es ist dadurch nicht für jedes Pixel eines Bilds eine Tiefeninformation verfügbar. Abbildung 5.2b zeigt ein Beispiel im direkten Vergleich zur Stereo-Punktwolke. Die Fusion der LiDAR-Informationen in das Stereo Matching soll die Schwächen beider Sensoren ausgleichen und für eine größere Anzahl von Pixeln eine genauere Disparität liefern. Um auf einem Fahrzeug eingesetzt werden zu können, muss die Fusion in Echtzeit erfolgen, d. h. die Bilder und Punktwolken müssen schneller verarbeitet werden, als bei einem Sensor-Zyklus von 10 Hz die nächste Szene aufgenommen wird.

Für das vorgestellte Verfahren wird Stereo mittels Semi-Global Matching (SGM) [Hirschmüller, 2005] (siehe Abschnitt 5.3.1) auf einem horizontalen Kamera-Paar berechnet. Die LiDAR-Punktwolke wird dabei nicht auf Objekt- oder Punktebene mit den berechneten Stereo-Tiefen oder auf Disparitätsebene mit fertigen Stereo-Disparitäten fusioniert, sondern als zusätzliche Information in das Stereo Matching integriert, um bereits dort Mehrdeutigkeiten aufzulösen. Dafür werden aus den Bildkorrespondenzen und der LiDAR-Punktwolke jeweils Kosten für mögliche Disparitäten berechnet und diese Kosten fusioniert. Alternative Ansätze aus der Literatur und die Vorteile der vorgestellten kostenbasierten Fusion gegenüber diesen Ansätzen werden in Abschnitt 5.2 beschrieben. Abschnitt 5.3 erklärt anschließend das zugrundeliegende SGM und die Ableitung von Matching-Kosten aus einer LiDAR-Punktwolke. In Abschnitt 5.4 werden dann mittels realer und simulierter Daten die Ergebnisse der Fusion untersucht und quantitativ mit anderen Verfahren verglichen.

Die Haupt-Vorteile der Fusion sind bereits in einem Beispiel in Abbildung 5.3 zu sehen: Verglichen mit traditionellem Stereo-SGM mit zwei horizontalen Kameras ist das resultierende Disparitätsbild des fusionierten LiDAR-SGMs deutlich dichter – gerade auf Objekten mit wenig Textur und in Bildbereichen, die nur in der Referenzkamera sichtbar sind (wie insbesondere dem Bildrand). Zudem sind die geschätzten Disparitäten genauer und weisen einen niedrigeren RMSE (Wurzel des mittleren quadratischen Fehlers, engl. *Root Mean Square Error*) auf.



(a) **Stereo-SGM**: 76,66 % Pixel mit korrekter Disparität, RMSE 1,00 px



(b) **LiDAR-SGM**: 93,12 % Pixel mit korrekter Disparität, RMSE 0,58 px

Abbildung 5.3:
Realdatenbeispiel, wie die Fusion von LiDAR-Informationen in das Stereo Matching die geschätzten Disparitäten verbessern kann. (a) zeigt die Disparitäten, die nur mittels horizontalem Stereo Matching berechnet wurden, (b) das Ergebnis der Fusion. Niedrige Disparitäten werden blau visualisiert, hohe Disparitäten rot. Die Szene ist Teil des KITTI-Datensatzes [Geiger et al., 2013]. Das obere Drittel des Bilds wird nicht durch den LiDAR-Sensor erfasst, weshalb die Disparitäten in diesem Bereich durch die Fusion nicht verbessert werden. Weitere sichtbare Effekte werden im Verlauf des Kapitels diskutiert.

5.2 Stand der Forschung

Zur Fusion von Stereo und LiDAR wurden bereits unterschiedliche Herangehensweisen veröffentlicht. Ein früher Ansatz, der ebenfalls auf SGM basiert, wurde durch Badino et al. [2011] entwickelt. In einer tiefen Fusion wird dabei vor dem Stereo Matching anhand der Entfernungsmessungen des LiDAR-Sensors der Suchraum für mögliche Disparitäten eines Pixels eingeschränkt. Dies hat im Vergleich zur kostenbasierten Fusion den Vorteil, dass das Matching beschleunigt wird. Da ein dichtes Tiefenbild benötigt wird, wird linear (horizontal und vertikal) zwischen den LiDAR-Messungen interpoliert. Der Disparitätssuchraum eines Pixels wird dann um diese (interpolierte) Tiefe herum eingeschränkt. Zudem werden im Stereo Matching Abweichungen der geschätzten Disparität von den (interpolierten) Tiefen an diesem Punkt bestraft. Die dadurch entstehenden zusätzlichen Matching-Kosten haben einen ähnlichen Effekt wie das in dieser Arbeit entwickelte Verfahren. Durch die harte Beschränkung des Suchraums und die Interpolation zwischen den LiDAR-Messungen werden jedoch für einige Pixel nur aufgrund von Interpolationen ohne direkte Messungen Disparitäten ausgeschlossen, und wahre Informationen aus Bildkorrespondenzen verloren. Leider fehlen in der Veröffentlichung quantitative wie auch zeitliche Auswertungen.

Maddern und Newman [2016] erweitern in einem anderen Ansatz das Stereo-Framework von Geiger et al. [2010] um die Verwendung von LiDAR-Stützpunkten. Dazu werden sowohl aus Bildkorrespondenzen als auch aus den LiDAR-Punkten sogenannte „Prior-Disparitäten“ und dazugehörige Standardabweichungen berechnet. Für ein dichteres Prior-Disparitätsbild wird in Dreiecken zwischen nahen Stützpunkten linear interpoliert. Wird für ein Pixel sowohl aus Bildkorrespondenzen als auch aus LiDAR-Messungen eine Prior-Disparität gewonnen, so wird diejenige mit der geringeren Standardabweichung verwendet, um ein Prior-Disparitätsbild zu erhalten. Die Prior-Disparitäten werden schließlich unter Verwendung von Bilddeskriptoren anhand der Stereobilder präzisiert. Insgesamt bietet dieses Verfahren allerdings keine hohe Genauigkeit (siehe Abschnitt 5.4.4), insbesondere in Bereichen mit fehlender LiDAR-Information [Park et al., 2018]. Dies liegt daran, dass ebenfalls nicht zwischen echtem Vorwissen durch Messungen und interpolierten Informationen unterschieden wird. Auch Courtois und Aouf [2017] erweitern das Stereo-Framework von Geiger et al. [2010] mit zusätzlichen Stützpunkten aus der LiDAR-Punktwolke. Sie verwenden jedoch ein bilaterales Filter anstatt der linearen Interpolation. Dadurch erhöht sich die Genauigkeit (siehe Abschnitt 5.4.4), solange die Stützpunkte nah genug beieinander liegen [Courtois und Aouf, 2017]. Das Verfahren ist jedoch nicht echtzeitfähig.

Die Rechenzeiten sind auch das Hauptproblem der in letzter Zeit beliebten Neuronalen Faltungsnetze, engl. *Convolutional Neural Networks* (CNNs) zur Stereo-LiDAR-Fusion. Die erste Fusion von Stereo mit LiDAR mittels eines CNNs erfolgte durch Park et al. [2018] auf Disparitätsebene. Basierend auf SGM wird ein Stereo-Disparitätsbild berechnet und anschließend mittels eines tiefen neuronalen Netzes mit einem von einer LiDAR-Punktwolke abgeleiteten Disparitätsbild fusioniert. Diese späte Fusion benötigt jedoch die zehnfache Rechenzeit des zugrundeliegenden SGMs und erreicht dennoch nicht die Genauigkeit des hier vorgestellten LiDAR-SGMs (vgl. Abschnitt 5.4.4).

Die auf dem Prinzip des Ende-zu-Ende-Lernens basierenden neuronalen Netze von Wang et al. [2019], Cheng et al. [2019] und Choe et al. [2021] benötigen jeweils mehr als eine Sekunde pro Szene (eine Szene besteht aus einem Bild-Paar und einer LiDAR-Punktwolke) und sind damit nicht für die Anwendung auf autonomen Fahrzeugen geeignet. Zhang et al. [2020] und Mai et al. [2021] geben gar keine Inferenzzeiten ihrer CNNs an, weshalb auch hier nicht mit einer Echtzeit-Anwendbarkeit zu rechnen ist. Weiter wird in [Meng et al., 2021] festgestellt, dass sich Ende-zu-Ende gelernte neuronale Netze zur Fusion von Stereo und LiDAR (insbesondere [Wang et al., 2019] und [Cheng et al., 2019]) auf einzelne Datensätze überspezialisieren.

Indem lediglich die Generierung von Features durch ein neuronales Netz übernommen wird, aber für die Kostenoptimierung wieder auf SGM zurückgegriffen wird, ist es Meng et al. [2021] zuletzt schließlich gelungen, mit einer CNN-basierten Stereo-LiDAR-Fusion Echtzeitfähigkeit zu erreichen. Aus dem Hamming-Abstand der gelernten Features werden Matching-Kosten gewonnen, aus denen mittels einer kreuzförmigen Aggregation von LiDAR-Stützpunkten und der Kosten-Aggregation durch Semi-Global Matching ein optimiertes Disparitätsbild extrahiert wird. Die Fehlerrate dieses Verfahrens steigt jedoch auch bei leerer LiDAR-Punktwolke kaum, sodass die erreichten Ergebnisse unabhängig von der Stereo-LiDAR-Fusion sind.

5.3 Kostenbasierte Fusion einer LiDAR-Punktwolke in Stereo Matching mittels Semi-Global Matching

Das in dieser Arbeit entwickelte Verfahren, LiDAR-SGM, greift auf analytische Weise direkt in SGM ein. Damit ist es sehr effizient und kommt ohne Trainingsdaten aus. Zudem werden die LiDAR-Daten nur unterstützend, nicht einschränkend verwendet, und nicht interpoliert.

5.3.1 Stereo-SGM: Berechnung der Stereo-Matching-Kosten und Semi-Global Matching

Um LiDAR-SGM zu verstehen, muss zuerst die Funktionsweise des zugrundeliegenden (Stereo-)SGMs betrachtet werden. Beim Semi-Global Matching [Hirschmüller, 2005] wird aus zwei Kamerabildern ein Disparitätsbild geschätzt durch:

1. Die pixelweise Berechnung von Matching-Kosten, mit
2. zusätzlichen, globalen Glattheits-Anforderungen und
3. die anschließende Suche nach den minimalen Gesamtkosten.

Berechnung der Stereo-Matching-Kosten Zunächst werden für jedes Pixel p Matching-Kosten $\boldsymbol{C}(\mathsf{p}, d)$ für alle möglichen Disparitäten d berechnet. Dafür wird die Ähnlichkeit eines Pixels im linken Kamerabild zu jedem Pixel in derselben Zeile des rechten Kamerabilds bestimmt. Dazu werden jeweils kleine Ausschnitte aus den Graubildern um die beiden Pixel herum verglichen. Um die Ähnlichkeit zweier Bildausschnitte zu bestimmen, wird hier die Census-Transformation [Zabih und Woodfill, 1994] verwendet. Andere oft verwendete Ähnlichkeitsmaße basieren auf der Transinformation (engl. *Mutual Information*) [Viola und Wells, 1995] oder der Rank-Transformation [Zabih und Woodfill, 1994]. Hirschmüller und Scharstein [2007] zeigen jedoch, dass die Rank-Transformation auf Bildern ohne radiometrische Transformationen zu weniger verwischten Objekt-Grenzen führt als die Transinformation. Weiter zeigen Banz et al. [2012], dass die Census-Transformation eine wiederum bessere Kantenschärfe als die Rank-Transformation liefert, da sie die räumliche Information über Grauwertunterschiede zu Nachbarpixeln erhält.

Bei der Census-Transformation wird für jedes Pixel p eine Bitfolge $R(\mathsf{p})$ erzeugt, indem sein Grauwert $\mathbf{Y}(\mathsf{p})$ einzeln mit den Grauwerten in seiner Nachbarschaft $\mathcal{N}_\mathsf{p}$ verglichen wird. Benachbarte Pixel $\mathsf{p}' \in \mathcal{N}_\mathsf{p}$ mit einem Grauwert $\mathbf{Y}(\mathsf{p}')$ kleiner oder gleich $\mathbf{Y}(\mathsf{p})$ ergeben eine binäre 1, Pixel mit einem höheren Grauwert eine binäre 0. Formal kann der Census-String definiert werden als

$$R(\mathsf{p}) = \bigotimes_{\mathsf{p}' \in \mathcal{N}_\mathsf{p}} \mathcal{H}(\mathbf{Y}(\mathsf{p}) - \mathbf{Y}(\mathsf{p}')) \ , \tag{5.1}$$

wobei $\mathcal{H}(\cdot)$ die Heaviside-Funktion $\mathcal{H}(x) = 0 \quad \forall\, x < 0, \ = 1 \quad \forall\, x \geq 0$ und $\otimes$ die Konkatenation der resultierenden Bits darstellt. Das Nachbarschafts-Fenster $\mathcal{N}_\mathsf{p}$ ist dabei um p zentriert, muss also eine ungerade Größe haben, und wird $9\,\mathrm{px} \times 9\,\mathrm{px}$ groß gewählt.

Aus diesen Census-Bitfolgen werden Matching-Kosten $\boldsymbol{C}(\mathsf{p}, d)$ – hier zur Konkretisierung Stereo-Matching-Kosten $\boldsymbol{C}_\mathsf{S}(\mathsf{p}, d)$ genannt – für jedes Pixel des linken Referenzbilds und jeweils jede mögliche Disparität $d \in [0, 128[$ px berechnet. Dafür wird der Hamming-Abstand $\Delta_{\mathsf{hamming}}(\cdot)$ der Census-Bitfolge des Pixels $\mathsf{p} = (u, v)$ im linken Bild und des um die Disparität d verschobenen Pixels $\mathsf{p}' = (u + d, v)$ im rechten Bild berechnet, d. h. die Anzahl unterschiedlicher Bits gezählt:

$$\boldsymbol{C}_\mathsf{S}(\mathsf{p}, d) = \Delta_{\mathsf{hamming}}(R_\mathsf{L}(u, v), R_\mathsf{R}(u + d, v)) \; . \tag{5.2}$$

Dies kann für jedes Pixel und jede Disparität komplett unabhängig voneinander berechnet werden und lässt sich daher gut parallelisieren. Das Ergebnis ist ein dreidimensionaler Kosten-Kubus der Größe (Bildbreite × Bildhöhe × 128).

Semi-Global Matching Die aus dieser Pixel-Ähnlichkeit resultierenden Kosten sind sehr verrauscht und oft mehrdeutig, weshalb sie nicht direkt für die Disparitätsschätzung verwendet werden. SGM stellt daher zusätzlich globale Glattheitsbedingungen auf und bestraft Änderungen in der Disparität zwischen benachbarten Pixeln. Das optimale Disparitätsbild $\mathbf{D}$, das für jedes Pixel die im Gesamtkontext aller Pixel optimale Disparität enthält, soll demnach die Energiefunktion

$$\mathrm{E}(\mathbf{D}) = \sum_{\mathsf{p}} \left(\boldsymbol{C}(\mathsf{p}, \mathbf{D}_\mathsf{p}) + \sum_{\mathsf{q} \in \mathcal{N}_\mathsf{p}} P_1 \, \mathrm{T}[|\mathbf{D}_\mathsf{p} - \mathbf{D}_\mathsf{q}| = 1] + \sum_{\mathsf{q} \in \mathcal{N}_\mathsf{p}} P_2 \, \mathrm{T}[|\mathbf{D}_\mathsf{p} - \mathbf{D}_\mathsf{q}| > 1] \right) \tag{5.3}$$

minimieren. $\mathrm{T}[\cdot]$ stellt dabei die Kronecker-Delta-Funktion dar, die als Einheitsimpuls genau dann 1 wird, falls die Bedingung in den eckigen Klammern erfüllt ist, und andernfalls 0 wird. Dadurch bestraft P_1 alle Pixel q in der Nachbarschaft $\mathcal{N}_\mathsf{p}$ von p, deren Disparität nur um 1 px abweicht, um auf Objekten möglichst glatte Disparitätsverläufe zu erhalten. Um Ausreißer zu unterdrücken, bestraft P_2 außerdem Sprünge in der Disparität um mehr als 1 px. Die Nachbarschaft $\mathcal{N}_\mathsf{p}$ kann dabei abweichend von der Nachbarschaft für die Census-Transformation gewählt werden.

Die Minimierung dieser Energiefunktion unter Berücksichtigung der Nachbarschaften aller Pixel ist jedoch ein NP-vollständiges Problem. Daher zerlegt SGM das Problem in einzelne 1D-Pfade im Kosten-Kubus: Um die optimale Disparität für ein Pixel p zu erhalten, werden für jede Disparität d Kosten $\boldsymbol{L}_\mathsf{r}(\mathsf{p}, d)$ entlang verschiedener Bildpfade r (vgl. Abbildung 5.4) aggregiert und die aggregierten Kosten minimiert:

$$\mathbf{D}(\mathsf{p}) = \arg\min_d \sum_{\mathsf{r}} \boldsymbol{L}_\mathsf{r}(\mathsf{p}, d) \, . \tag{5.4}$$

Die Kosten $\boldsymbol{L}_\mathsf{r}(\mathsf{p}, d)$ werden dabei rekursiv berechnet:

$$\begin{aligned} \boldsymbol{L}_\mathsf{r}(\mathsf{p}, d) = \boldsymbol{C}(\mathsf{p}, d) + \min \Big(& \boldsymbol{L}_\mathsf{r}(\mathsf{p} - \mathsf{r}, d), \\ & \boldsymbol{L}_\mathsf{r}(\mathsf{p} - \mathsf{r}, d - 1) + P_1, \; \boldsymbol{L}_\mathsf{r}(\mathsf{p} - \mathsf{r}, d + 1) + P_1, \\ & \min_i \boldsymbol{L}_\mathsf{r}(\mathsf{p} - \mathsf{r}, i) + P_2 \Big) - \min_j \boldsymbol{L}_\mathsf{r}(\mathsf{p} - \mathsf{r}, j) \, , \end{aligned} \tag{5.5}$$

wobei $i \in [0, 128[\backslash \{d - 1, d, d + 1\}$ und $j \in [0, 128[$.

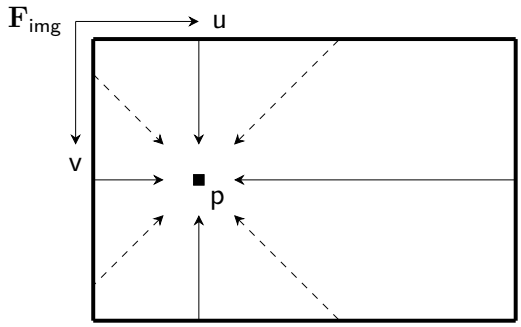


Abbildung 5.4:
Visualisierung der von SGM zur Kostenaggregation für das Pixel p verfolgten Bildpfade in alle Richtungen r im Bildkoordinatensystem $\mathbf{F}_{\text{img}}$. In jedem Fall werden vier horizontale und vertikale Pfade, optional zusätzlich die vier gestrichelt dargestellten, schrägen Pfade verfolgt. Die Abbildung ist angelehnt an [Hirschmüller et al., 2012, Figure 1].

Nachverarbeitung Im Anschluss an die Optimierung werden noch Plausibilitätsprüfungen durchgeführt. Bei einem sog. Eindeutigkeits-Check (engl. *Uniqueness Check*) wird die Disparität eines Pixels auf ungültig gesetzt, falls kein eindeutiges Minimum vorliegt, d. h. neben dem globalen Minimum ein zweites, lokales Minimum mit ähnlich niedrigen Kosten existiert. Zusätzlich wird beim sog. Links-Rechts-Konsistenz-Check (vgl. [Hirschmüller, 2005]) die Berechnung der Disparitäten mit dem rechten Kamerabild als Referenzbild wiederholt. Weicht die für ein Pixel p im linken Bild bestimmte Disparität d mehr als 1 px von der im rechten Bild für das Pixel $\mathrm{p} + (d, 0)$ bestimmten Disparität ab, wird die Disparität ebenfalls als ungültig betrachtet.

Zuletzt werden aus den ganzzahligen Pixeldisparitäten subpixel-genaue Disparitäten bestimmt. Dafür wird eine Parabel durch die Kosten der kostenoptimalen Disparität eines Pixels und die Kosten der nächsthöheren und -niedrigeren Disparitäten gelegt, und das Minimum der quadratischen Funktion bestimmt.

5.3.2 LiDAR-only-SGM: Berechnung der LiDAR-Matching-Kosten

Diese Kostenaggregation mit der globalen Glättung des Semi-Global Matchings soll für LiDAR-SGM nun auf LiDAR-Punktwolken angewandt werden. Dazu müssen aus den LiDAR-Daten pixelweise Matching-Kosten berechnet werden (anstelle der Census-Transformation auf den Bild-Daten). Unter Berücksichtigung der intrinsischen und extrinsischen Kalibrierung der Kamera zum LiDAR sowie der Eigenbewegung des Fahrzeugs wird dafür die Punktwolke in ein Kamerabild projiziert (vgl. Abschnitt 2.3). Dabei ist es wichtig, dass Kamera und LiDAR-Sensor zeitlich so synchronisiert sind, dass zur Mitte der Belichtungszeit der Kamera der rotierende LiDAR-Sensor die Mitte des Bilds erreicht. Da die Sensoren unterschiedliche Blickwinkel auf die Szene haben, können bei der Projektion mehrere LiDAR-Punkte in dasselbe Bildpixel projiziert werden. Um die Dichte der LiDAR-Punkte zu erhöhen, wird ein Tiefenpunkt außerdem in einer 9er Nachbarschaft nicht nur in ein Pixel projiziert, sondern auch in dessen direkte und diagonale Nachbar-Pixel dupliziert. Ein Beispiel für die auf diese Art und Weise gewonnene Projektion ist in Abbildung 5.5 zu sehen.

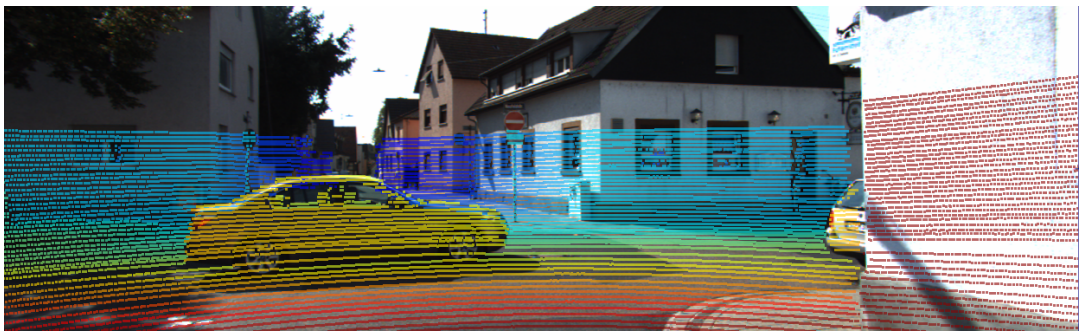


Abbildung 5.5:
Aus einer LiDAR-Punktwolke abgeleitete Disparitäten in ein Kamerabild projiziert, wobei eine Nachbarschaft von 9 Pixeln berücksichtigt wird. Die abgebildete Szene ist Teil des KITTI-Datensatzes [Geiger et al., 2013]. Man bemerke, dass die Punktwolke den oberen Teil des Bilds nicht abdeckt. Wie in Kapitel 4 diskutiert, verdecken Objekte in den Messdaten des erhöht eingebauten LiDAR-Sensors weniger von ihrem Hintergrund als im Kamerabild. Da hier auf eine explizite Verdeckungsbehandlung verzichtet wurde, kommt es dadurch beispielsweise an der Motorhaube des abgebildeten Fahrzeugs fälschlicherweise zu einer Projektion von Bodenpunkten auf das Fahrzeug (blaue Punkte auf der Motorhaube). Dadurch werden die Objektkanten durch die Fusion leicht unscharf (vgl. Abbildung 5.3).

Um SGM anwenden zu können, müssen für jedes Pixel p Kosten für jede mögliche Disparität d spezifiziert werden. Daher wird für jeden LiDAR-Punkt $\mathbf{p}$ basierend auf seiner Tiefe $z_\mathbf{p}$ eine virtuelle Disparität

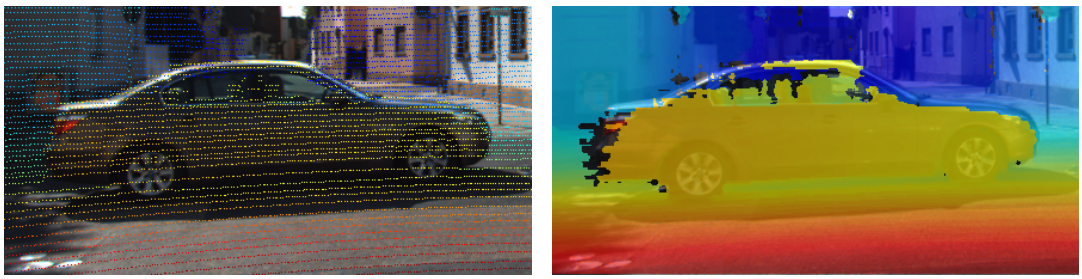
$$d_\mathbf{p} = \frac{f_\mathsf{u} b}{z_\mathbf{p}} \tag{5.6}$$

berechnet, wobei die Brennweite f_u und die Basisbreite b für eine spätere Fusion mit Stereo dem Stereo-Setup bzw. den rektifizierten Bildern entsprechend gewählt werden. Die Disparität muss an dieser Stelle nicht ganzzahlig diskretisiert werden. Die LiDAR-Matching-Kosten $\boldsymbol{C}_\mathsf{L}(\mathsf{p}, d)$ eines Pixels werden dann unter Berücksichtigung aller dem Pixel zugeordneten LiDAR-Punkte $\mathcal{P}$ durch die minimalen Kosten

$$\boldsymbol{C}_\mathsf{L}(\mathsf{p}, d) = \min_{\mathbf{p} \in \mathcal{P}} \left(\min \left(\frac{64}{\sigma^2} \left(d - d_\mathbf{p} \right)^2, 64 \right) \right) \tag{5.7}$$

bestimmt. Dies modelliert eine Gaußsche Normalverteilung der Disparitäten mit Mittelwert $d_\mathbf{p}$ und Standardabweichung σ durch quadratische Kosten mit den minimalen Kosten von $\boldsymbol{C}_\mathsf{L} = 0$ bei der Disparität $d = d_\mathbf{p}$ und nach außen hin maximalen Kosten von $\boldsymbol{C}_\mathsf{L} = 64$. 64 entspricht der Mitte des Wertebereichs der Kosten $\boldsymbol{C}(\mathsf{p}, d) \in [0, 128[$. σ wird auf 2 px gesetzt.

Auf diese LiDAR-Matching-Kosten könnte nun direkt die Kostenaggregation von SGM angewandt werden. Damit kann aus der dünnen LiDAR-Punktwolke ein dichtes Tiefenbild interpoliert werden, wie das Beispiel in Abbildung 5.6 zeigt. Wie später in Abschnitt 5.4 gezeigt wird, sind die resultierenden Disparitätsbilder dieses LiDAR-only-SGMs oft bereits korrekter als die durch Stereo Matching erhaltenen Disparitäten.



(a) LiDAR-Punktwolke (b) LiDAR-only-SGM

Abbildung 5.6:
Mittels Semi-Global Matching kann aus der dünn besetzten LiDAR-Punktwolke in (a) ohne weitere Informationen das dichte Tiefenbild in (b) interpoliert werden.

5.3.3 LiDAR-SGM: Fusion von Stereo- und LiDAR-Matching-Kosten

Um jedoch diese genaueren Tiefeninformationen mit den dichteren Informationen aus den Bildkorrespondenzen zu kombinieren, soll nun ein fusioniertes Disparitätsbild berechnet werden. Angelehnt an die Fusion zweier Stereopaare in [Kallwies et al., 2020a] erfolgt die Fusion auf Kostenebene: Die Stereo-Matching-Kosten und die LiDAR-Matching-Kosten werden kombiniert und eine gemeinsame Kostenaggregation auf fusionierten Kosten durchgeführt. Die fusionierten Kosten

$$\boldsymbol{C}(\mathsf{p}, d) = \alpha \boldsymbol{C}_{\mathsf{L}}(\mathsf{p}, d) + (1 - \alpha)\boldsymbol{C}_{\mathsf{S}}(\mathsf{p}, d) \tag{5.8}$$

berechnen sich dabei als Linearkombination der LiDAR-Matching-Kosten $\boldsymbol{C}_{\mathsf{L}}$ und der Stereo-Matching-Kosten $\boldsymbol{C}_{\mathsf{S}}$. Der LiDAR-Einflussfaktor α legt dabei die Gewichtung der beiden Sensormodalitäten fest. Die Wahl von α wird in Abschnitt 5.4.2 genauer untersucht. Während $\alpha = 0$ dem traditionellen Stereo-SGM entspricht, bewirkt $\alpha = 1$ eine reine Interpolation der LiDAR-Punktwolke (LiDAR-only-SGM). Durch die Fusion beider Kosten (LiDAR-SGM) kann nicht nur die Dichte der Disparitäten erhöht werden, sondern es können auch Mehrdeutigkeiten besser aufgelöst und die Genauigkeit erhöht werden, wie das Beispiel in Abbildung 5.7 zeigt.

5.4 Ergebnisse

Um beurteilen zu können, ob das entwickelte LiDAR-SGM zur Wahrnehmung der statischen Umgebung geeignet ist, müssen sowohl qualitative als auch quantitative Auswertungen erfolgen. Zunächst muss ein geeigneter LiDAR-Einflussfaktor α bestimmt werden. Dafür werden im Folgenden Dichte und Genauigkeit der resultierenden Disparitätsbilder in Abhängigkeit von der Gewichtung der Sensormodalitäten untersucht und LiDAR-SGM damit mit Stereo-SGM und LiDAR-only-SGM verglichen. Im Anschluss werden verschiedene Sensorsetups evaluiert. Das resultierende Verfahren wird dann mit anderen Methoden zur Stereo-LiDAR-Fusion aus der Literatur verglichen. Dabei wird auch die Rechenzeit untersucht. Zuletzt werden die Effekte der Stereo-LiDAR–Fusion qualitativ beurteilt.

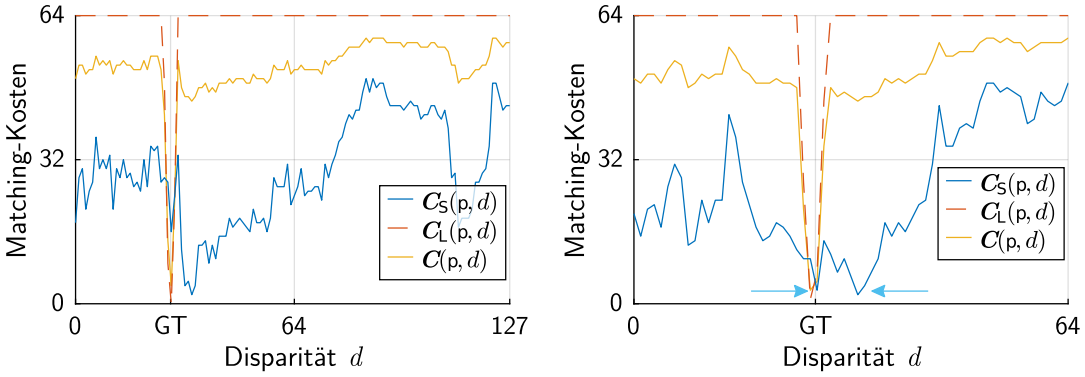


Abbildung 5.7:
Beispiel anhand eines exemplarischen Pixels p für die Fusion der traditionellen Stereo-Matching-Kosten aus der Census-Transformation (blau) mit den aus einer LiDAR-Punktwolke abgeleiteten LiDAR-Matching-Kosten (rot) zu den fusionierten Kosten (gelb), wobei $\alpha = 0{,}7$. Die Ground-Truth-Disparität ist mit GT markiert. In (a) kann die Disparität durch die Fusion mit den LiDAR-Informationen genauer bestimmt werden (Minimum näher an GT). In (b) helfen die LiDAR-Kosten weiterhin, eine Mehrdeutigkeit in den Stereo-Matching-Kosten (siehe blaue Pfeile) aufzulösen.

5.4.1 Setup

Aufgrund der besseren Verfügbarkeit von Datensätzen erfolgt die Auswertung auf urbanen Szenen. Dabei werden zwei Arten von Datensätzen herangezogen: Realdaten und simulierte Daten. Während die Realdaten die tatsächliche Einsetzbarkeit im Anwendungsfall sicherstellen, werden die synthetischen Daten benötigt, um eine wahre, mit den LiDAR-Daten unkorrelierte Ground-Truth zu erhalten. Zudem können in einer Simulation leicht zusätzliche Sensoren wie eine dritte Kamera oder unterschiedliche LiDAR-Sensoren hinzugefügt und evaluiert werden.

Realdaten Für Realdaten kann vom KITTI-Datensatz Gebrauch gemacht werden: Die Trainingsdaten des KITTI Stereo 2015 Benchmarks [Menze und Geiger, 2015] stellen Ground-Truth-Disparitäten für rektifizierte RGB-Bildpaare zur Verfügung (mit einer Basisbreite von 53,27 cm). 142 dieser RGB-Bildpaare finden sich in den Rohdaten des KITTI-Datensatzes [Geiger et al., 2013] wieder, sodass ihnen eine LiDAR-Punktwolke zugeordnet werden kann. Auf dieser Teilmenge der Trainingsdaten kann damit die Stereo-LiDAR-Fusion evaluiert werden. Die Ground-Truth-Disparitätsbilder basieren jedoch auf den akkumulierten LiDAR-Punktwolken. Dadurch kann die Stereo-LiDAR-Fusion nicht ganz unabhängig bewertet werden, da Teile der Ground-Truth in das Stereo-Matching eingefiltert werden. Der Datensatz ist dennoch interessant: Er ermöglicht einen einfachen Vergleich mit Ansätzen aus der Literatur [Maddern und Newman, 2016, Courtois und Aouf, 2017, Park et al., 2018, Wang et al., 2019, Cheng et al., 2019], die Ergebnisse hierfür angeben. Zudem ist es (Stand November 2022) der einzig öffentlich verfügbare Realdaten-Datensatz für Stereo Matching mit LiDAR-Punktwolken. Leider sind die LiDAR-Punktwolken nicht

eigenbewegungskorrigiert. Zwar wurde kürzlich ein Softwarepaket zur Eigenbewegungskorrektur veröffentlicht[1], jedoch wurde hier auf dessen Einsatz verzichtet, um die Vergleichbarkeit mit den anderen Verfahren und ihren Auswertungen zu erhalten. In den Auswertungen werden die Disparitäten aller Pixel evaluiert, für die eine Ground-Truth-Disparität verfügbar ist („occ“ im Benchmark). Da die Ground-Truth aus einer LiDAR-Punktwolke erzeugt wurde, deckt sie jedoch nur den unteren Bereich des Bilds ab und ist zudem sehr dünn – lediglich bei Fahrzeugen wurden die Disparitätsbilder zusätzlich anhand von CAD-Modellen aufgefüllt. Beispielbilder aus diesem Datensatz sind im ganzen Kapitel zu finden, ein Beispiel für die Ground-Truth-Disparitäten findet sich am Ende des Kapitels in der qualitativen Auswertung (Abbildung 5.13).

Simulierte Daten Für eine unkorrelierte Auswertung der Stereo-LiDAR-Fusion wurden mithilfe des CARLA Fahrsimulators [Dosovitskiy et al., 2017] zwei synthetische Datensätze mit je 500 statischen Szenen gerendert: *CARLA ClearSunset* und *CARLA HardRainNoon*. Diese zeigen dieselbe Stadt („Town03“) bei zwei verschiedenen Wetterbedingungen: leicht sonnig („ClearSunset“) und starker Regen („HardRainNoon“). Dies wirkt sich jedoch nur auf die simulierten Kamerabilder aus, nicht auf die LiDAR-Punktwolke. Aus den Ergebnissen einer perfekten Tiefenkamera wurden echte Ground-Truth-Disparitäten für das gesamte linke Kamerabild berechnet. Durch die statischen Aufnahmen wird keine Bewegungskorrektur benötigt.

Die RGB-Kameras haben eine Basisbreite von 80 cm und simulieren auch Effekte wie Jitter, Bloom und Lens Flair. Die Einstellungen für den LiDAR-Sensor orientieren sich am Velodyne HDL-64E [Velodyne LiDAR, Inc., 2011], der im KITTI Datensatz verwendet wird (64 Ringe, 1 333 000 Punkte in der Sekunde, 120 m maximale Reichweite). Zur Evaluation des Sensorsetups wurden ein weiterer LiDAR-Scanner mit 128 Ringen, 2 400 000 Punkten in der Sekunde und einer maximalen Reichweite von 245 m (ähnlich einem Velodyne Alpha Prime [Velodyne LiDAR, Inc., 2018]), sowie eine dritte Kamera 20 cm oberhalb der linken Kamera hinzugefügt. Für zukünftige Vergleiche wurden alle Daten der Öffentlichkeit zur Verfügung gestellt[2]. Beispielbilder sind in der qualitativen Auswertung in den Abbildungen 5.14 und 5.15 zu sehen.

Implementierung Die Implementierung von Stereo-SGM und LiDAR-SGM nutzt für die Census-Transformation und das SGM die CUDA-Implementierung libSGM[3]. Die Matching-Kosten werden entlang von 4 Pfaden (horizontal und vertikal) mit den Straffaktoren $P_1 = 10$ und $P_2 = 100$ aggregiert. Für bestmögliche Ergebnisse könnten die Straffaktoren auch für jedes Pixel unterschiedlich gewählt werden, bspw. mittels eines neuronalen Netzes [Seki und Pollefeys, 2017]. Für den Eindeutigkeits-Check wird ein uniqueness-Faktor von 0,95 gewählt, d. h. die Kosten im globalen Kosten-Minimum müssen weniger als 95 % der Kosten im zweitniedrigsten Minimum betragen. Um ein dichteres Disparitätsbild zu erhalten, wird auf das resultierende Disparitätsbild noch ein Median-Filter mit einer Fenstergröße von 5 px × 5 px angewandt. Auf eine weitere Filterung oder Prüfung der Disparitäten wird für die Evaluation verzichtet.

[1] `https://github.com/fracgawd/kitti_motion_compensation` (24. Februar 2024)
[2] `https://doi.org/10.60776/U8CTRE`
[3] `https://github.com/fixstars/libSGM` (Version vom 16. Januar 2018)

Metriken Für die Auswertung werden zwei Metriken herangezogen: der Anteil der Pixel (mit Ground-Truth), für die eine korrekte Disparität geschätzt wurde (Korrektheitsrate) und die Wurzel des mittleren quadratischen Fehlers (RMSE) dieser korrekten Pixel. Wie im KITTI Stereo Benchmark [Menze und Geiger, 2015] wird ein Pixel dabei als korrekt betrachtet, falls der Fehler in der geschätzten Disparität kleiner als 3 px ist oder weniger als 5 % der Ground-Truth-Disparität beträgt.

5.4.2 Einfluss der LiDAR-Informationen auf das Stereo Matching

Anhand dieser Metriken können nun verschiedene Verfahren und Parametrierungen bewertet werden. Dabei ist zunächst interessant, wie sich die Fusion der LiDAR-Informationen in das Stereo Matching auswirkt und welchen Einfluss die Gewichtung der Stereo- und der LiDAR-Matching-Kosten hat. Abbildung 5.8 zeigt hierfür die Korrektheitsrate und den RMSE für die drei Datensätze in Abhängigkeit vom LiDAR-Einflussfaktor α, wobei $\alpha = 0$ dem traditionellen, rein kamerabasierten, horizontalen Stereo Matching entspricht (Stereo-SGM) und $\alpha = 1$ das rein LiDAR-basierte SGM zur Punktwolkeninterpolation (LiDAR-only-SGM) darstellt.

Stereo-SGM Betrachtet man vorweg die Ergebnisse von Stereo-SGM, so fällt auf, dass diese nicht nur stark durch die Wetterbedingungen beeinflusst werden – die Korrektheitsrate und der RMSE sind auch bei den simulierten Eingangsdaten deutlich schlechter als bei den Realdaten. Dies liegt zum einen daran, dass die synthetischen Bilder von CARLA weniger Textur aufweisen. Zum anderen sind die Ground-Truth-Disparitäten des KITTI-Datensatzes sehr dünn und decken nicht das gesamte Bild ab, während in den CARLA-Datensätzen vollständige Disparitätsbilder vorliegen. Dadurch bezieht die Auswertung mehr, oft schwieriger zu erkennende Bereiche, wie beispielsweise den Himmel oder kleine Objekte, mit ein.

LiDAR-only-SGM Aus diesem Grund kann LiDAR-only-SGM bei dem Datensatz *CARLA ClearSunset* kaum mehr Pixel korrekt schätzen als Stereo-SGM, da die LiDAR-Punktwolke, die bei LiDAR-only-SGM ausschließlich genutzt wird, keine Informationen über den oberen Teil des Bilds enthält. Damit erhöht sich zwar der Anteil korrekter Pixel im unteren Teil des Bilds signifikant, da für den oberen Teil des Bilds jedoch ohne Messung keine Disparitäten geschätzt werden können, ist der Anteil korrekter Pixel des Gesamtbilds trotzdem nicht höher als bei Stereo-SGM. Auf den anderen beiden Datensätzen weist LiDAR-only-SGM eine signifikant höhere Korrektheitsrate und niedrigere RMSEs auf als Stereo-SGM. Allerdings wird bei *CARLA HardRainNoon* die LiDAR-Punktwolke im Gegensatz zu den Kamerabildern nicht durch das schlechte Wetter beeinflusst, und beim KITTI-Datensatz basiert die Ground-Truth direkt auf den LiDAR-Daten. Daher sind die Vergleiche auf diesen beiden Datensätzen nicht komplett fair. Dennoch zeigt sich insgesamt, dass LiDAR-only-SGM kamerabasiertes Stereo Matching darin übertreffen kann, dichte und korrekte Tiefeninformationen für ein Kamerabild zu erhalten, insbesondere im relevanten, unteren Teil des Bilds. Damit zeigt sich auch das Potenzial einer Stereo-LiDAR-Fusion.

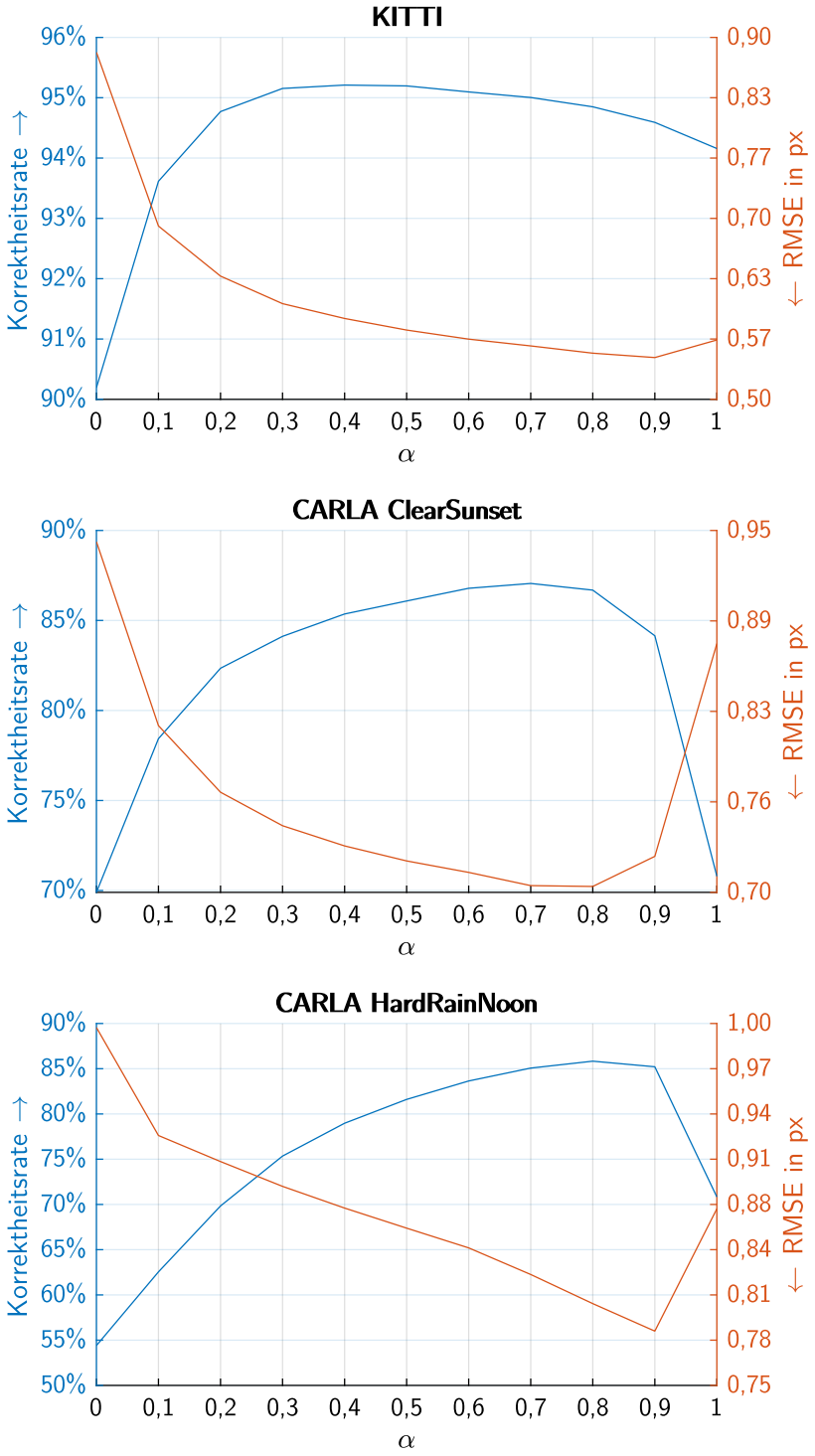


Abbildung 5.8:
Median des Anteils der Pixel, für die eine korrekte (vgl. Metriken in Abschnitt 5.4.1) Disparität geschätzt wurde (blau) und ihr RMSE (rot) in Abhängigkeit von der Gewichtung α der LiDAR-Matching-Kosten (und $(1-\alpha)$ der Stereo-Matching-Kosten) über alle Bilder der drei Datensätze. Man beachte die unterschiedlichen Achsen für die Korrektheitsrate (links) und den RMSE (rechts). Die Pfeile in der y-Achsenbeschriftung geben dabei an, ob höhere oder niedrigere Werte besser sind.

LiDAR-SGM Mit der Fusion von Stereo- und LiDAR-Informationen werden auf allen Datensätzen die mit Abstand besten Korrektheitsraten und RMSEs erzielt. Welche Gewichtung der Stereo-Matching-Kosten und der LiDAR-Matching-Kosten dabei am besten abschneidet, hängt vom Datensatz ab. Während auf dem KITTI-Datensatz die höchste Korrektheitsrate bei $\alpha = 0{,}4$ erreicht wird, liegt die maximale Korrektheitsrate bei den CARLA-Datensätzen bei $\alpha = 0{,}7$ und $\alpha = 0{,}8$.

Als bester Kompromiss wird ein LiDAR-Einflussfaktor von $\alpha = 0{,}7$ gewählt. Die genauen Werte für die Korrektheitsrate und den RMSE des resultierenden LiDAR-SGMs im Vergleich zu Stereo-SGM und LiDAR-only-SGM sind in Tabelle 5.1 angegeben: Der mittlere Anteil der Pixel mit falsch geschätzter Disparität kann durch die Fusion der LiDAR-Punkte in das Stereo Matching um bis zu 67 % reduziert werden (CARLA HardRainNoon). Gleichzeitig verringert sich der RMSE um mindestens 17 % (CARLA HardRainNoon). Auf dem KITTI-Datensatz kann der RMSE sogar um 36 % reduziert werden. Damit wird der Nutzen der Stereo-LiDAR-Fusion quantitativ deutlich. Qualitative Ergebnisse für den optischen Vergleich werden in Abschnitt 5.4.7 gezeigt.

Tabelle 5.1:
Median des Anteils der Pixel mit korrekt geschätzter Disparität und ihr RMSE für horizontales Stereo Matching ($\alpha = 0$), LiDAR-only-SGM ($\alpha = 1$) und das fusionierte LiDAR-SGM ($\alpha = 0{,}7$) für alle drei Datensätze. Die Tabelle gibt die genauen Zahlenwerte zu Abbildung 5.8 an.

	KITTI		CARLA ClearSunset		CARLA HardRainNoon	
	Korrekt	RMSE	Korrekt	RMSE	Korrekt	RMSE
Stereo-SGM ($\alpha = 0$)	90,19 %	0,88 px	69,90 %	0,94 px	54,40 %	1,00 px
LiDAR-only-SGM ($\alpha = 1$)	94,16 %	0,57 px	70,79 %	0,87 px	70,83 %	0,87 px
LiDAR-SGM ($\alpha = 0{,}7$)	95,00 %	0,56 px	87,05 %	0,70 px	85,07 %	0,83 px

5.4.3 Wahl der Sensorik

In vorherigen Arbeiten am Institut wurde zur Verbesserung der Stereodisparitäten anstelle des LiDAR-Sensors eine dritte, vertikal versetzte Kamera in die Stereoverarbeitung fusioniert. Damit konnte nicht nur die Erkennung von horizontalen Strukturen, sondern auch die Genauigkeit der geschätzten Disparitäten verbessert werden [Kallwies et al., 2020a]. Daher muss nun untersucht werden, ob der Nutzen des LiDAR-Sensors den der zuvor integrierten dritten Kamera erreicht oder übersteigt, oder ob sogar der Einsatz beider zusätzlicher Sensoren sinnvoll ist. Dazu wurde das Triple-SGM-Verfahren von Kallwies et al. [2020a] in die LiDAR-SGM-Applikation integriert und die simulierten CARLA-Datensätze wurden um zusätzliche Sensoren erweitert. Abbildung 5.9 zeigt nun für verschiedene Sensor-Setups die Verteilungen der Korrektheitsrate und des RMSEs über alle Bilder der CARLA-Datensätze.

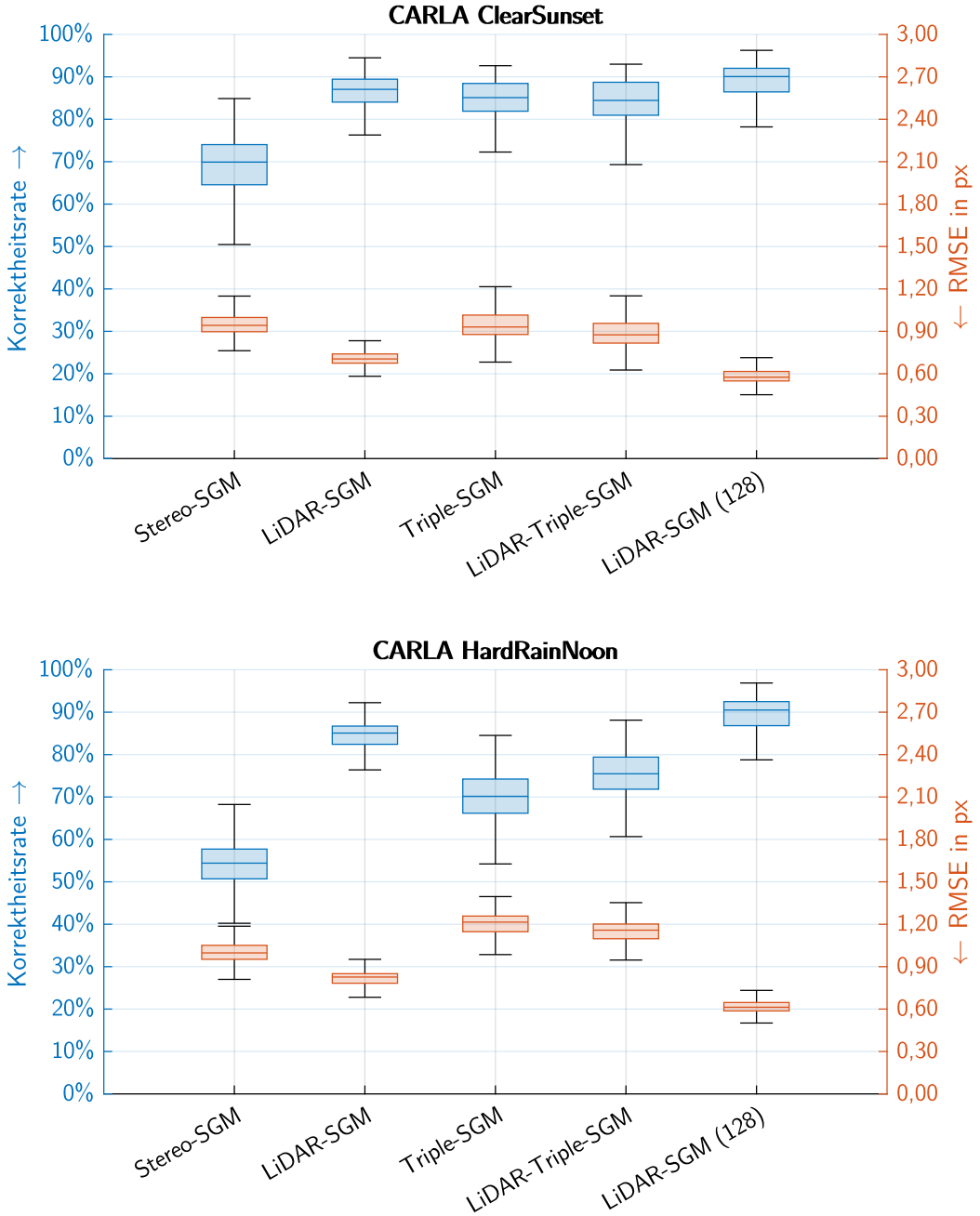


Abbildung 5.9:
Verteilungen der Korrektheitsrate und des RMSEs über alle Bilder der zwei synthetischen CARLA-Datensätze für Stereo-SGM ($\alpha = 0$), LiDAR-SGM (mit einem LiDAR-Scanner mit 64 Ringen, $\alpha = 0{,}7$), Triple-SGM [Kallwies et al., 2020a] ($\alpha = 0$), Triple-SGM fusioniert mit LiDAR-Informationen ($\alpha = 0{,}7$), und LiDAR-SGM mit einem LiDAR-Scanner mit 128 Ringen ($\alpha = 0{,}7$).

Anzahl der Kameras Vergleicht man zunächst die Ergebnisse von LiDAR-SGM, das zwei Kameras und einen LiDAR-Sensor fusioniert, mit den Ergebnissen von Triple-SGM, das drei Kameras verwendet, so gewinnt die multimodale Sensordatenfusion zweier unterschiedlicher Messprinzipien: In der höheren Korrektheitsrate von LiDAR-SGM zeigt sich, dass der LiDAR-Sensor die Schwächen des Stereo Matchings zum Beispiel bei wenig oder periodischer Textur besser ausgleichen kann als eine dritte Kamera. Zudem spiegelt sich die höhere Entfernungsgenauigkeit des LiDAR-Sensors in einem im Vergleich zu Triple-SGM deutlich geringeren RMSE wieder. Betrachtet man weiterhin die Ergebnisse von LiDAR-Triple-SGM, also der Fusion von drei Kameras und einem LiDAR-Sensor, zeigt sich im Vergleich zu Triple-SGM erneut der Nutzen des LiDAR-Sensors für die Genauigkeit und bei schlechten Wetterbedingungen. Dennoch werden schlechtere Ergebnisse erzielt als durch LiDAR-SGM mit nur zwei Kameras. Dies ist dadurch zu begründen, dass bei LiDAR-Triple-SGM durch die Fusion dreier Kosten-Tensoren (horizontale Stereo-Matching-Kosten, vertikale Stereo-Matching-Kosten, LiDAR-Matching-Kosten) die einzelnen Gradienten an Stärke verlieren. Der Verbau einer dritten Kamera bringt also keine Vorteile und ist bei der Verwendung des LiDAR-Sensors nicht mehr nötig. Qualitative Ergebnisse für den optischen Vergleich der Verwendung von zwei oder drei Kameras mit oder ohne LiDAR sind am Ende des Kapitels in den Abbildungen 5.14 und 5.15 zu sehen.

Art des LiDAR-Sensors Nachdem die CARLA-Simulation es ermöglicht, die Sensoren frei zu konfigurieren, kann als alternatives Sensor-Setup auch die Integration eines besseren LiDAR-Sensors untersucht werden. Die Konfiguration des bisher verwendeten LiDAR-Sensors ist dem im KITTI-Datensatz verwendeten Velodyne HDL-64E mit 64 Laserdioden nachempfunden. Moderne LiDAR-Sensoren bieten jedoch inzwischen 128 Dioden, eine höhere Reichweite und grob doppelt so viele Messpunkte. Wie sich dies auf LiDAR-SGM auswirkt, zeigen die Ergebnisse von LiDAR-SGM (128) in der rechten Spalte von Abbildung 5.9, für das eine dem Velodyne Alpha Prime nachempfundene LiDAR-Punktwolke mit 128 Ringen verwendet wurde. Es zeigt sich, dass mit dem besseren Sensor eine noch etwas höhere Korrektheitsrate und auch ein geringerer RMSE erzielt werden kann. Der Unterschied ist jedoch deutlich geringer als die durch die Fusion erreichten Verbesserungen. Im Folgenden werden weiterhin die Ergebnisse für die LiDAR-Punktwolke mit 64 Ringen gezeigt.

5.4.4 Vergleich mit anderen Ansätzen

Nachdem die CARLA-Datensätze die Evaluation verschiedener Sensor-Setups erlaubt haben, erlaubt der KITTI-Datensatz den Vergleich von LiDAR-SGM mit verwandten Verfahren. Die Arbeiten von Maddern und Newman [2016], Courtois und Aouf [2017], Park et al. [2018], Wang et al. [2019] und Cheng et al. [2019] verwenden in ihrer Evaluation dieselben in Abschnitt 5.4.1 beschriebenen 142 Szenen des KITTI Stereo 2015 Benchmarks mit den zugehörigen Punktwolken aus den Rohdaten. Dadurch können die angegebenen Ergebnisse direkt verglichen werden. Die Ergebnisse von Meng et al. [2021] sind leider nicht vergleichbar, da diese durch Abtasten der Ground-Truth eigene LiDAR-Daten erzeugen.

Tabelle 5.2:
Vergleich des entwickelten LiDAR-SGMs ($\alpha = 0{,}7$) mit anderen Verfahren zur Stereo-LiDAR-Fusion aus der Literatur mittels des KITTI Stereo 2015 Benchmarks [Menze und Geiger, 2015]. Die Zahlen stammen aus den entsprechenden Veröffentlichungen. Zur Referenz werden auch das zugrundeliegende Stereo-SGM ($\alpha = 0$) und LiDAR-only-SGM ($\alpha = 1$) ausgewertet. *Fehler$_{HG}$*, *Fehler$_{VG}$* und *Fehler$_{Ges.}$* geben jeweils den Anteil Pixel mit falscher (vgl. Abschnitt 5.4.1) Disparität im Hintergrund, im Vordergrund und insgesamt an. *Dichte* gibt den Anteil der Pixel (von allen Pixeln mit einer Ground-Truth-Information) an, für die eine (nicht zwangsläufig korrekte) Disparität geschätzt wurde. *Rechenzeit* gibt die Größenordnung der für eine Szene benötigten Rechenzeit an. Auf die Angabe der genauen Zahlen wird dabei verzichtet, da sich die verwendete Hardware zwischen den einzelnen Veröffentlichungen stark unterscheidet.

Methode	Fehler$_{HG}$	Fehler$_{VG}$	Fehler$_{Ges.}$	Dichte	Rechenzeit
Stereo-SGM ($\alpha = 0$)	3,64 %	9,78 %	4,54 %	92,23 %	<100 ms
LiDAR-only-SGM ($\alpha = 1$)	2,31 %	18,22 %	4,61 %	97,07 %	<100 ms
LiDAR-SGM ($\alpha = 0{,}7$)	2,06 %	14,47 %	3,87 %	97,46 %	<100 ms
Combined+Pyr [Maddern und Newman, 2016]	4,55 %	13,18 %	5,91 %	99,62 %	<100 ms
Tri_BF [Courtois und Aouf, 2017]	2,62 %	9,01 %	3,60 %	98,59 %	>0,5 s
Park et al. [2018]	-	-	4,84 %	99,8 %	<100 ms
Wang et al. [2019]	-	-	3,35 %	-	1 s
Cheng et al. [2019]	-	-	1,98 %	100,00 %	2 s

Das Entwicklungskit des KITTI Benchmarks [Menze und Geiger, 2015] bestimmt separate Fehlerraten für Pixel im Hintergrund, Pixel im Vordergrund und alle Pixel mit einer Ground-Truth. Tabelle 5.2 zeigt die Ergebnisse für LiDAR-SGM im Vergleich:

- Im Hintergrund übertrifft LiDAR-SGM nicht nur Stereo-SGM, sondern auch die Stereo-LiDAR-Fusionen aus [Maddern und Newman, 2016] und [Courtois und Aouf, 2017].
- Im Vordergrund verschlechtert die Fusion der LiDAR-Punktwolke in das Stereo-Matching dagegen das Ergebnis. Dies ist damit zu erklären, dass dort die leicht unterschiedlichen Blickwinkel der Sensoren einen stärkeren Einfluss haben und eine explizite Verdeckungsbehandlung benötigt würde. Zudem werden aufgrund der fehlenden Eigenbewegungskorrektur der LiDAR-Punktwolke einzelne Objekte ungenau abgegrenzt. So ist auch in [Courtois und Aouf, 2017] für das fusionierte Verfahren eine höhere Vordergrund-Fehlerrate angegeben als für das zugrundeliegende Stereoverfahren.
- Auf dem gesamten Bild unterbietet LiDAR-SGM die Fehlerrate von Stereo-SGM jedoch wieder. Darüber hinaus untertrifft die Gesamtfehlerrate von LiDAR-SGM auch die der beiden anderen echtzeitfähigen Algorithmen von Maddern und Newman [2016] und Park et al. [2018]. Die Fusionsmethoden von Courtois und Aouf [2017], Wang et al. [2019] und Cheng et al. [2019] erzielen zwar noch geringere Fehlerraten, sind jedoch weit von einer Echtzeitfähigkeit entfernt.

5.4.5 Rechenzeiten

Das entwickelte LiDAR-SGM benötigt auf den vollaufgelösten (1242 px × 375 px) Bildern der drei Datensätze durchschnittlich 24 ms pro Szene, verglichen mit 21 ms für das zugrundeliegende Stereo-SGM (auf der in Abschnitt 2.1 vorgestellten Hardware). Die Fusion mit den Informationen des LiDAR-Sensors benötigt nur zusätzliche 2,5 ms, wohingegen Triple-SGM [Kallwies et al., 2020a] 8,7 ms zusätzlich benötigt. Die genaue Verteilung der Rechenzeiten auf die einzelnen Schritte von LiDAR-SGM, Stereo-SGM und Triple-SGM wird in Tabelle 5.3 gezeigt.

Tabelle 5.3:
Mittlere Rechenzeiten der einzelnen Verarbeitungsschritte von LiDAR-SGM, Stereo-SGM und Triple-SGM (in derselben Implementierung) auf dem Datensatz *CARLA ClearSunset*.

Schritte	LiDAR-SGM	Stereo-SGM	Triple-SGM
Datenkopie auf die GPU + Initialisierung	1,2 ms	1,1 ms	2,0 ms
Berechnung der Stereo-Matching-Kosten	1,0 ms	1,0 ms	2,0 ms
Berechnung der LiDAR-Matching-Kosten	0,7 ms	–	–
Fusion der LiDAR-Matching-Kosten	1,7 ms	–	–
Fusion der Triple-Matching-Kosten	–	–	7,0 ms
Kostenaggregation	8,2 ms	8,2 ms	8,2 ms
Minimumsuche	6,7 ms	6,7 ms	6,5 ms
Nachverarbeitung	0,4 ms	0,4 ms	0,4 ms
Datenkopie von der GPU	0,8 ms	0,8 ms	0,8 ms
Median-Filter	3,2 ms	3,2 ms	3,2 ms
Gesamtzeit	23,9 ms	21,4 ms	30,1 ms

Eine Kostenaggregation auf acht im 45°-Winkel angeordneten SGM-Pfaden anstatt nur vier horizontalen und vertikalen Pfaden (vgl. Abbildung 5.4 und Abschnitt 5.4.1) erhöht die durchschnittliche Rechenzeit von LiDAR-SGM auf 32 ms. Die Ergebnisse sind mit einer Korrektheitsrate von 95,18 % und einem RMSE von 0,55 px auf dem KITTI-Datensatz jedoch kaum besser als mit vier Pfaden (95,00 % und 0,56 px).

5.4.6 Skalierung der Eingangsbilder

Für eine schnellere Verarbeitung hingegen können die Eingangsbilder verkleinert werden. Bei einer Skalierung der KITTI-Bilder auf 50 % der ursprünglichen Auflösung benötigt LiDAR-SGM (mit vier SGM-Pfaden) nur noch 6 ms, während die Korrektheitsrate und der RMSE nach Rückskalierung auf die volle Auflösung mit 94,35 % und 0,58 px nur marginal schlechter sind. Damit ist das Ergebnis immer noch signifikant besser als mit Stereo-SGM auf der vollen Auflösung. Wie in Abbildung 5.10 zu sehen ist, gilt dies sogar bis zu einer Reduktion der Auflösung auf 20 %, womit die Berechnung der Disparitätsbilder nur noch 1,5 ms benötigt. Die bei den Skalierungsfaktoren von 0,9 bis 0,5 zu beobachtenden leichten Schwankungen in der Korrektheitsrate sind damit zu erklären, dass Texturen mit unterschiedlicher Periodizität bei manchen Skalierungsfaktoren besser erhalten werden als bei anderen.

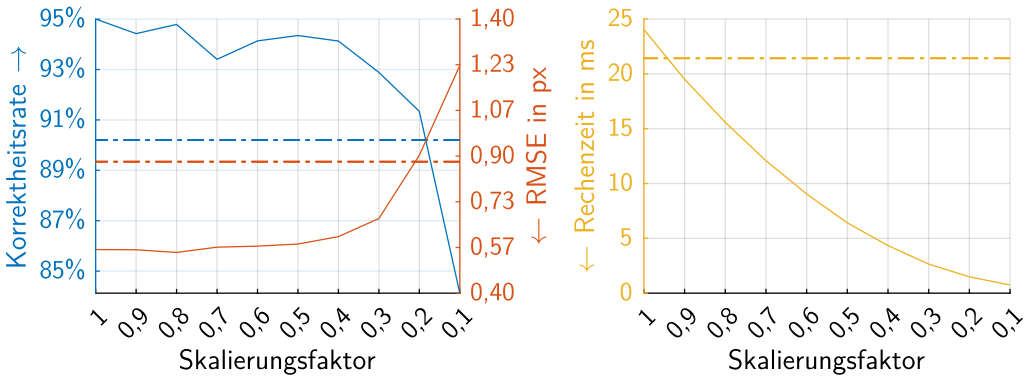
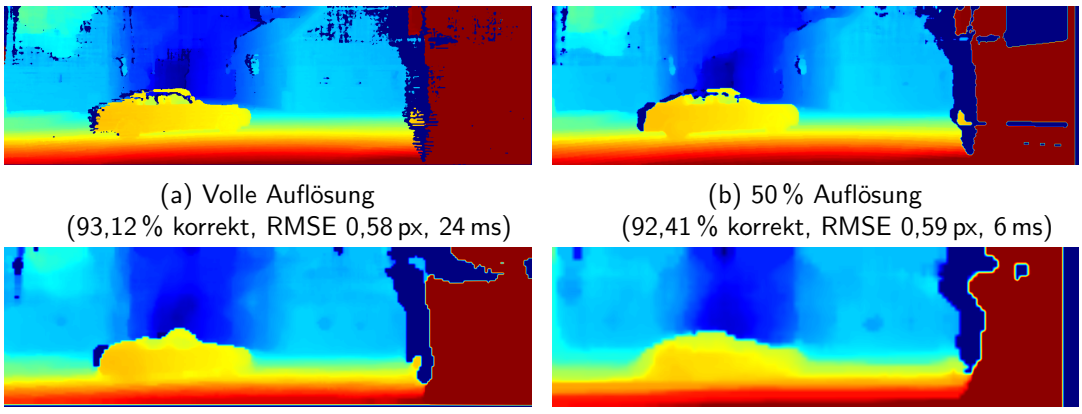


Abbildung 5.10:
Median des Anteils der Pixel, für die eine korrekte Disparität geschätzt wurde (blau), ihr RMSE (rot) und die für die Berechnung benötigte Rechenzeit (gelb) in Abhängigkeit der Skalierung der Eingangsbilder aus dem KITTI-Datensatz. Die resultierenden Disparitäten werden dazu auf die volle Auflösung zurück skaliert. Die gestrichelt/gepunkteten Linien zeigen zum Vergleich Korrektheitsrate, RMSE und Rechenzeit von Stereo-SGM.



(a) Volle Auflösung
(93,12 % korrekt, RMSE 0,58 px, 24 ms)

(b) 50 % Auflösung
(92,41 % korrekt, RMSE 0,59 px, 6 ms)

(c) 20 % Auflösung
(90,30 % korrekt, RMSE 0,93 px, 1,5 ms)

(d) 10 % Auflösung
(82,98 % korrekt, RMSE 1,24 px, 0,8 ms)

Abbildung 5.11:
Mit linearer Interpolation auf die volle Auflösung zurück skalierte Disparitätsbilder, die mittels LiDAR-SGM ($\alpha = 0{,}7$) aus verschiedenen Skalierungen desselben Eingangsbilds aus dem KITTI-Datensatz (vgl. Abbildung 5.3) berechnet wurden. Angegeben ist jeweils die Korrektheitsrate, der RMSE der korrekten Pixel und die benötigte Rechenzeit.

Abbildung 5.11 zeigt an einer Beispielszene die verschiedenen Disparitätsbilder, die durch LiDAR-SGM auf unterschiedlich skalierten Eingangsbildern berechnet werden. Bei 50 % Auflösung ist (im vom LiDAR-Sensor abgedeckten Bereich) optisch kaum eine Verschlechterung sichtbar. Im Gegenteil ist zu beobachten, dass das Disparitätsbild durch die Interpolation bei der Rückskalierung weniger kleine Lücken aufweist. Bei einer Restauflösung von 20 % gehen hingegen merklich Details verloren, wobei selbst bei einer Skalierung auf 10 % der vollen Auflösung noch eine grobe Distanz von Autos und Häusern geschätzt werden kann.

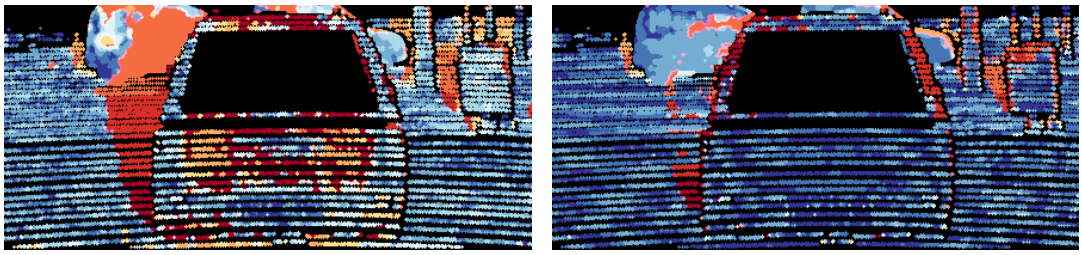
5.4.7 Qualitative Ergebnisse

Nachdem in allen vorherigen Auswertungen die Leistung des LiDAR-SGMs anhand von Zahlen demonstriert wurde, sollen zuletzt nun qualitative Beispiele zur eigenen Bewertung, ob und wo die Fusion der LiDAR-Messungen in das Stereo Matching vorteilhaft ist, gezeigt werden. Ein Effekt ist in der Betrachtung der Disparitätsfehler in Abbildung 5.12 zu sehen: Durch den horizontalen Versatz der beiden Kameras entstehen verdeckte Bereiche hinter bzw. neben Objekten. So kann für Bereiche im linken Kamerabild, die die rechte Kamera nicht einsieht, keine Disparität berechnet werden (siehe die roten Bereiche in Abbildung 5.12a). Der LiDAR-Sensor kann jedoch Tiefeninformationen für diese verschatteten Bereiche einbringen, sodass die Disparitätsbilder an dieser Stelle aufgefüllt werden (siehe Abbildung 5.12b).

In den Abbildungen 5.13 bis 5.15 sind weitere Effekte zu beobachten: Der linke Bildrand sowie die linke, untere Ecke des linken Kamerabilds sind im rechten Kamerabild ebenfalls nicht sichtbar, weshalb hier mit Stereo-SGM keine Disparität berechnet werden kann, mit LiDAR-SGM jedoch schon (sichtbar in allen Datensätzen). Weiter hat Stereo-SGM wie vermutet Probleme mit schwierigen Texturen in dunklen Bereichen (KITTI, Abbildung 5.13), hellen Bereichen mit Reflexionen von der Sonne (CARLA ClearSunset, Abbildung 5.14), nassem Untergrund (CARLA HardRainNoon, Abbildung 5.15) oder glänzenden Oberflächen (Bus in CARLA ClearSunset, Abbildung 5.14). Auch hier profitiert LiDAR-SGM stark von den zusätzlichen LiDAR-Informationen. Kleine Strukturen in höherer Distanz, wie Straßenlaternen oder Ampelpfosten, werden durch LiDAR-SGM jedoch etwas überglättet.

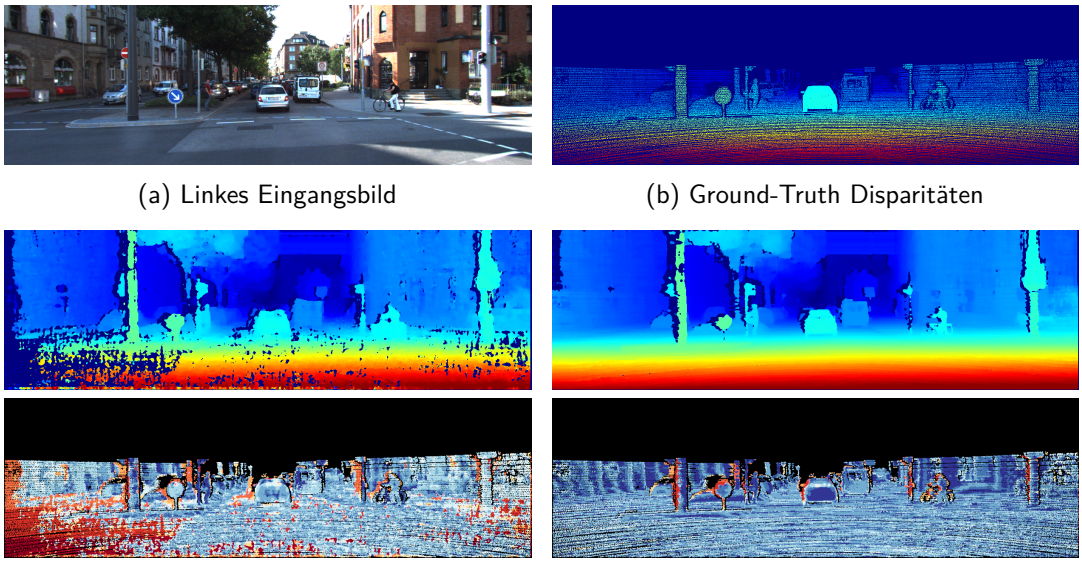
5.5 Diskussion

Mittels LiDAR-SGM ist es möglich, in Echtzeit LiDAR-Punktwolken in kamerabasiertes Stereo zu fusionieren. Das Ergebnis sind sowohl im Vergleich zum traditionellen, horizontalen Stereo, als auch zu Triple-Stereo mit drei Kameras, signifikant dichtere Disparitätsbilder mit einer höheren Genauigkeit, die dabei auch die von anderen weitgehend echtzeitfähigen Verfahren zur Fusion von Stereo und LiDAR übertrifft. Stände unbegrenzt Rechenzeit zur Verfügung, gäbe es noch bessere Methoden – die größte Genauigkeit erzielt ein CNN-basiertes Verfahren – für den Anwendungsfall des autonomen Fahrens sind diese jedoch ungeeignet.



(a) Stereo-SGM ($\alpha = 0$) (b) LiDAR-SGM ($\alpha = 0{,}7$)

Abbildung 5.12:
Visualisierung der Disparitätsfehler von Stereo-SGM und LiDAR-SGM in derselben Szene des KITTI-Datensatzes. Dunkelblau zeigt einen kleinen Fehler, dunkelrot einen großen Fehler. Die Farbkodierung ist über alle Abbildungen konstant (auch Abbildungen 5.13 bis 5.15).

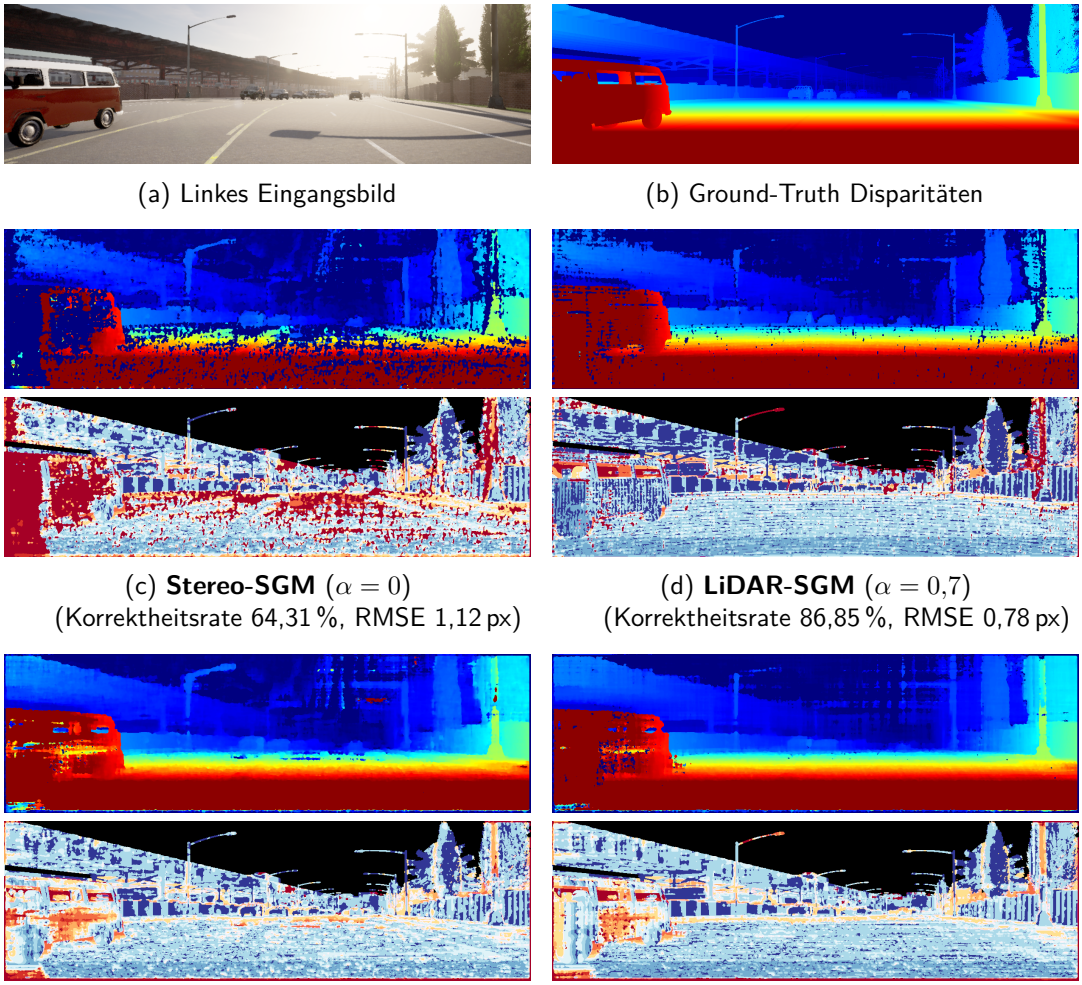


(a) Linkes Eingangsbild (b) Ground-Truth Disparitäten

(c) **Stereo-SGM** ($\alpha = 0$)
(Korrektheitsrate 78,45 %, RMSE 0,97 px)

(d) **LiDAR-SGM** ($\alpha = 0{,}7$)
(Korrektheitsrate 97,54 %, RMSE 0,53 px)

Abbildung 5.13:
Beispielhafte, qualitative Ergebnisse des **KITTI**-Datensatzes. (a) und (b) zeigen das linke Eingangsbild und die entsprechenden Ground-Truth-Disparitäten. (c) und (d) zeigen die Disparitäten (oben) zusammen mit ihrem Fehler zur Ground-Truth (unten) für das horizontale Stereo-SGM und das fusionierte LiDAR-SGM mit $\alpha = 0{,}7$. Die Disparitätsbilder sind farbkodiert von dunkelblau für eine Disparität von 0 zu dunkelrot für eine Disparität von 63 und höher. Die Fehlerbilder sind farbkodiert von dunkelblau für einen geringen Fehler zu dunkelrot für einen hohen Fehler. Die Farbkodierung ist über alle Abbildungen konstant (inkl. Abbildungen 5.14 und 5.15). Die Bildunterschriften geben die entsprechenden Korrektheitsraten und RMSEs der als korrekt eingestuften Pixel an.



(a) Linkes Eingangsbild

(b) Ground-Truth Disparitäten

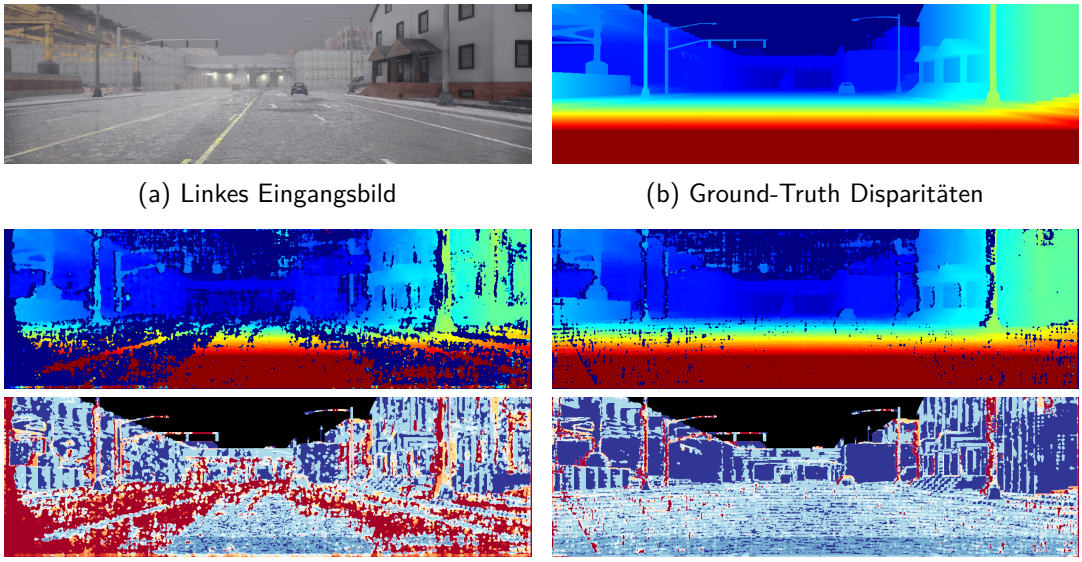
(c) **Stereo-SGM** ($\alpha = 0$)
(Korrektheitsrate 64,31 %, RMSE 1,12 px)

(d) **LiDAR-SGM** ($\alpha = 0{,}7$)
(Korrektheitsrate 86,85 %, RMSE 0,78 px)

(e) **Triple-SGM** ($\alpha = 0$)
(Korrektheitsrate 78,66 %, RMSE 1,09 px)

(f) **Triple-LiDAR-SGM** ($\alpha = 0{,}7$)
(Korrektheitsrate 79,77 %, RMSE 1,00 px)

Abbildung 5.14:
Beispielhafte, qualitative Ergebnisse des Datensatzes ***CARLA ClearSunset***. (a) und (b) zeigen das linke Eingangsbild und die entsprechenden Ground-Truth-Disparitäten. (c) und (d) zeigen die Disparitäten (oben) zusammen mit ihrem Fehler zur Ground-Truth (unten) für das horizontale Stereo-SGM und das fusionierte LiDAR-SGM mit $\alpha = 0{,}7$. (e) und (f) zeigen die resultierenden Disparitätsbilder (und ihren Fehler) von Triple-SGM [Kallwies et al., 2020a] und Triple-SGM fusioniert mit den LiDAR-Informationen ($\alpha = 0{,}7$). Die Disparitätsbilder sind farbkodiert von dunkelblau für eine Disparität von 0 zu dunkelrot für eine Disparität von 63 und höher. Die Fehlerbilder sind farbkodiert von dunkelblau für einen geringen Fehler zu dunkelrot für einen hohen Fehler. Die Farbkodierung ist über alle Abbildungen konstant (inkl. Abbildungen 5.13 und 5.15). Die Bildunterschriften geben die entsprechenden Korrektheitsraten und RMSEs der als korrekt eingestuften Pixel an.



(a) Linkes Eingangsbild

(b) Ground-Truth Disparitäten

(c) **Stereo-SGM** ($\alpha = 0$)
(Korrektheitsrate 60,87 %, RMSE 0,89 px)

(d) **LiDAR-SGM** ($\alpha = 0{,}7$)
(Korrektheitsrate 90,16 %, RMSE 0,69 px)

(e) **Triple-SGM** ($\alpha = 0$)
(Korrektheitsrate 73,45 %, RMSE 1,04 px)

(f) **Triple-LiDAR-SGM** ($\alpha = 0{,}7$)
(Korrektheitsrate 79,31 %, RMSE 0,96 px)

Abbildung 5.15:
Beispielhafte, qualitative Ergebnisse des Datensatzes ***CARLA HardRainNoon***. (a) und (b) zeigen das linke Eingangsbild und die entsprechenden Ground-Truth-Disparitäten. (c) und (d) zeigen die Disparitäten (oben) zusammen mit ihrem Fehler zur Ground-Truth (unten) für das horizontale Stereo-SGM und das fusionierte LiDAR-SGM mit $\alpha = 0{,}7$. (e) und (f) zeigen die resultierenden Disparitätsbilder (und ihren Fehler) von Triple-SGM [Kallwies et al., 2020a] und Triple-SGM fusioniert mit den LiDAR-Informationen ($\alpha = 0{,}7$). Die Disparitätsbilder sind farbkodiert von dunkelblau für eine Disparität von 0 zu dunkelrot für eine Disparität von 63 und höher. Die Fehlerbilder sind farbkodiert von dunkelblau für einen geringen Fehler zu dunkelrot für einen hohen Fehler. Die Farbkodierung ist über alle Abbildungen konstant (inkl. Abbildungen 5.13 und 5.14). Die Bildunterschriften geben die entsprechenden Korrektheitsraten und RMSEs der als korrekt eingestuften Pixel an.



(a) Stereo (b) LiDAR-SGM

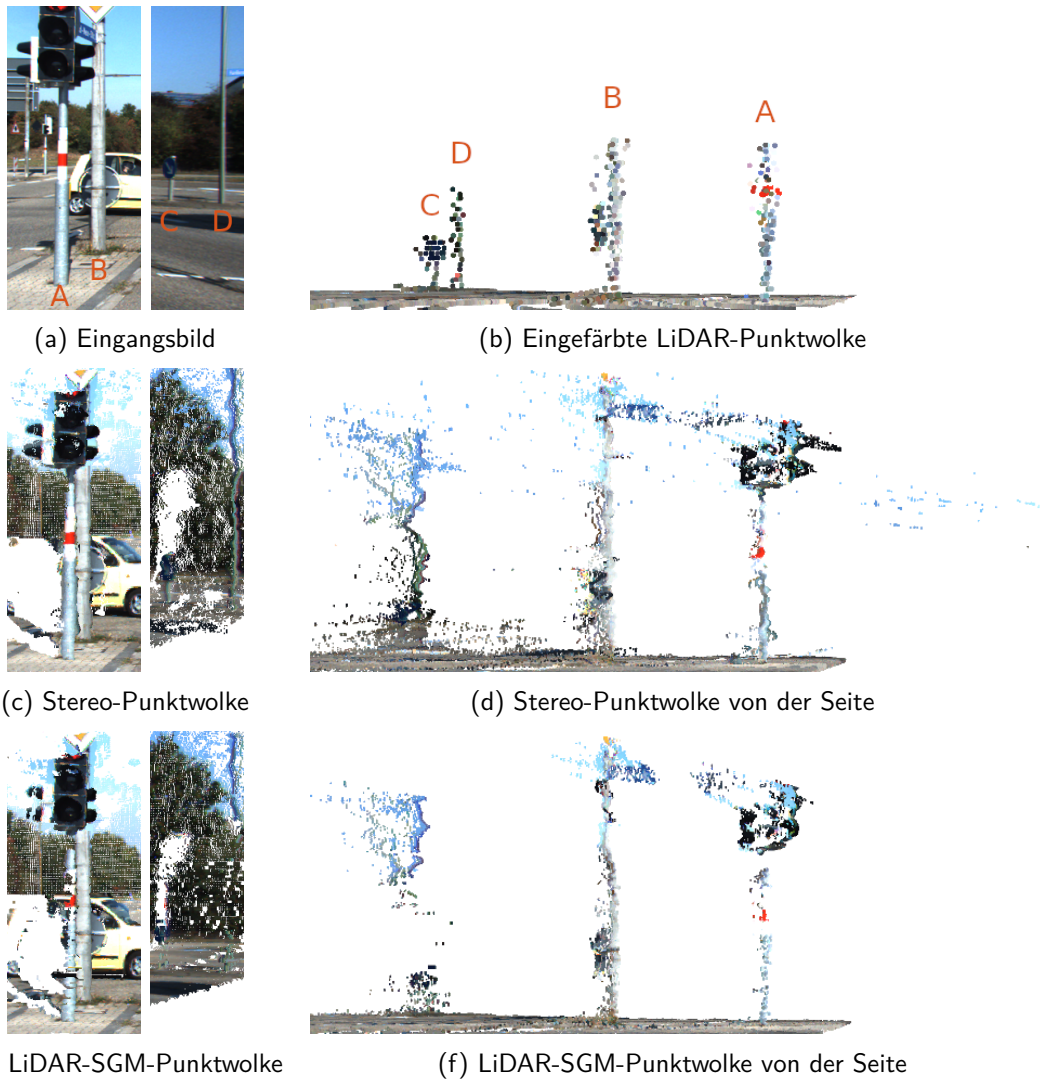
(c) LiDAR (eingefärbt) (d) Stereo (e) LiDAR-SGM

Abbildung 5.16:
Vergleich der (a) in Abbildung 5.2 abgebildeten, lückenhaften Stereo-Punktwolke mit (b) der durch LiDAR-SGM berechneten Punktwolke. Abbildungen (c) bis (e) zeigen den seitlichen Blick auf die Punktwolken vom Heck des Fahrzeugs aus.

Aus den Disparitätsbildern können letztendlich Punktwolken berechnet werden. Auch in dieser resultierenden Punktwolke zeigt sich eine deutliche Verbesserung durch die Fusion der LiDAR-Punktwolke in das Stereo Matching. Abbildung 5.16 greift die eingangs in Abbildung 5.2 gezeigte Szene auf und zeigt zu der Stereo-Punktwolke nun auch die LiDAR-SGM-Punktwolke. Außer bei den durchsichtigen Fenstern weist diese nun kaum mehr Lücken auf der Fahrzeugoberfläche auf. In der seitlichen Ansicht ist außerdem zu erkennen, dass das Tiefenrauschen sehr stark verringert werden konnte und die geschätzte Kontur des Fahrzeugs nun der in der LiDAR-Punktwolke erfassten Kontur ähnelt.

Das Heck des Fahrzeugs wird in der LiDAR-SGM-Punktwolke jedoch leicht abgeschnitten. Auch im Bereich der C-Säule entsteht eine neue Lücke. Dies liegt an dem in Abbildung 5.5 gezeigten Effekt: Durch die abweichenden Blickwinkel von Kamera und LiDAR, insbesondere aufgrund der höheren Einbaulage des LiDAR-Sensors, werden durch den LiDAR-Sensor gemessene Hintergrundpunkte, die im Kamerabild durch ein Objekt verdeckt sind, fälschlicherweise auf dieses Objekt projiziert. Dies ließe sich – wie bereits in Kapitel 4 für die eingefärbte LiDAR-Punktwolke diskutiert – durch eine Verdeckungsbehandlung vermeiden.

Abbildung 5.17 zeigt jedoch an einem weiteren Punktwolken-Beispiel ein anderes Problem. Die abgebildeten Ampel- und Straßenschildpfosten werden zwar durch LiDAR-SGM mit deutlich weniger Entfernungsrauschen gemessen als durch Stereo. Dennoch werden noch einige blaue Pixel aus dem Himmel fälschlicherweise Objekten zugeordnet, und die weiter entfernten Pfosten (C und D) haben immer noch eine hohe Tiefenunsicherheit. So weichen die abgebildeten Punkte des entferntesten Pfostens (D) in ihrer Tiefe stark von der vom LiDAR-Sensor gemessenen Tiefe ab: in der Abbildung erscheinen die Pfosten in den Stereo-Punktwolken deutlich weiter links. Weiterhin wird in der LiDAR-SGM-Punktwolke der mittlere Teil des Pfostens nicht erkannt. Gerade auf Feldwegen müssen jedoch oft dünne Pfosten neben dem Weg detektiert werden. Daher kann vor allem in der Hinderniserkennung



(a) Eingangsbild (b) Eingefärbte LiDAR-Punktwolke

(c) Stereo-Punktwolke (d) Stereo-Punktwolke von der Seite

(e) LiDAR-SGM-Punktwolke (f) LiDAR-SGM-Punktwolke von der Seite

Abbildung 5.17:
Vergleich der Genauigkeit verschiedener Punktwolken anhand der seitlichen Ansicht von Pfosten in einer Szene des KITTI-Datensatzes [Geiger et al., 2013], bei der sich das Ego-Fahrzeug selbst nicht bewegt. (a) zeigt Ausschnitte des Kamerabilds. Die Punktwolke in (b) wurde durch einen LiDAR-Sensor aufgenommen. Zur besseren Sichtbarkeit sind die LiDAR-Punkte vergrößert dargestellt. Da der LiDAR-Sensor nur bis zu 2° nach oben misst [Velodyne LiDAR, Inc., 2011], sind die Pfosten oben abgeschnitten. (c) und (d) zeigen eine frontale und eine seitliche Ansicht einer durch horizontales Stereo Matching berechneten Punktwolke. Die mittels LiDAR-SGM durch die Fusion der LiDAR-Punktwolke in das horizontale Stereo Matching berechnete Punktwolke ist in (e) und (f) zu sehen.

auf Feldwegen die LiDAR-SGM-Punktwolke die LiDAR-Punktwolke nicht ersetzen, so wie in Kapitel 4 auch die mit dem Kamerabild eingefärbte Punktwolke die Verwendung des Kamerabilds nicht ersetzen konnte. Beide Möglichkeiten zur Rohdatenfusion von Kamera und LiDAR führen also zu einem Informationsverlust. Aus diesem Grund wird in den nächsten Kapiteln stattdessen die Fusion von Kamera- und LiDAR-Informationen auf Merkmalsebene untersucht.

Teil III

Fusion auf Merkmalsebene

Bei der Fusion auf Merkmalsebene werden die Informationen aus dem Kamerabild und der LiDAR-Punktwolke direkt im jeweiligen Wahrnehmungsmodul fusioniert. Dazu werden aus allen Sensordaten für die jeweilige Anwendung relevante Merkmale extrahiert. Anhand aller Merkmale wird dann eine gemeinsame Schätzung getroffen. Für das autonome Fahren auf Feldwegen wurden vier wichtige Module zur Repräsentation der statischen Umgebung identifiziert:

- **Hindernis-Grid** (Kapitel 6): Im Hindernis-Grid werden statische Hindernisse kartiert, die die durch das autonome Fahrzeug befahrbare Fläche eingrenzen.
- **Terrain-Schätzung** (Kapitel 7): In der Terrain-Schätzung wird ein Modell der Bodenoberfläche geschätzt. Dieses enthält keine Hindernisse oder Objekte auf dem Terrain, jedoch negative Hindernisse wie Schlaglöcher oder Gräben. Das geschätzte Terrain-Modell gibt Aufschluss über die Geländesteigung auf und neben dem Weg. Da die aus dem Stereo Matching gewonnenen Entfernungsinformationen nicht genau genug für den Einsatz auf dem Fahrzeug waren, wird das Terrain-Modell außerdem dafür benötigt, eine Tiefeninformation für im Kamerabild erkannte Weg- und Kreuzungsmerkmale zu erhalten.
- **Straßen- und Wegetracking** (Kapitel 8): Das Straßen- und Wegetracking schätzt die Lage der Wegränder von Feld- und Waldwegen sowie der einzelnen Fahrspuren nicht-urbaner Straßen. Damit kann dem Weg oder der Straße autonom gefolgt werden.
- **Kreuzungserkennung** (Kapitel 9): Die Kreuzungserkennung detektiert schließlich Kreuzungen von Feldwegen und trackt ihre Position über die Zeit.

6 Hindernis-Grid

Abseits des strukturierten Straßenverkehrs ist es die Hauptaufgabe eines autonomen Fahrzeugs, befahrbare Bereiche zu identifizieren. Dabei müssen zuallererst mögliche Hindernisse erkannt werden. Auch wenn einem definierten, normalerweise hindernisfreien Feldweg gefolgt werden soll, muss beispielsweise in die Fahrspur ragender Vegetation, großen Steinen, umgefallenen Baumstämmen oder anderen unerwarteten Hindernissen ausgewichen werden können. Außerdem kann das Wissen über die Position statischer Hindernisse dabei helfen, den Suchraum nach Wegen einzugrenzen. Daher wird eine (lokale) Karte statischer Hindernisse benötigt. Um diese ohne Vorwissen lediglich aus der eigenen Wahrnehmung des Fahrzeugs aufzubauen, wurde ein sog. Hindernis-Grid implementiert.

6.1 Einleitung und Problemstellungen

Ausgangslage: Kombiniertes Umgebungsmodell Bisher wurden statische Hindernisse am Institut für Technik Autonomer Systeme (TAS) zusammen mit dem Boden in einer Geländekarte kartiert (vgl. [Himmelsbach et al., 2009, Jaspers et al., 2017]). Basierend auf klassischen Belegtheitsgittern (engl. *Occupancy Grids*) wird darin die unmittelbare Umgebung des Ego-Fahrzeugs in einem zweidimensionalen Grid (auch Gitter oder Raster genannt) rasterisiert, und für jede Grid-Zelle eine Wahrscheinlichkeit dafür geschätzt, dass dieser Bereich durch ein statisches Hindernis belegt ist. Aus diesen Belegtheitswahrscheinlichkeiten lässt sich beispielsweise das in Abbildung 6.1 visualisierte, ternäre Belegtheitsgitter ausgeben, das für jede Grid-Zelle angibt, ob diese frei oder belegt ist, oder ob ihr Zustand unbekannt ist.

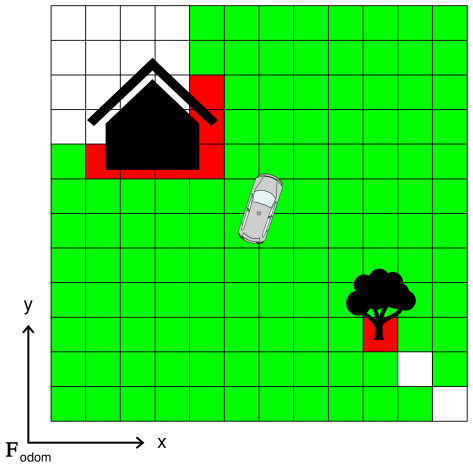


Abbildung 6.1:
Schematisches Beispiel eines ternären Belegtheitsgitters. Das statische Umfeld wird im ortsfesten Koordinatensystem der Odometrie $\mathbf{F}_{\text{odom}}$ in quadratische Grid-Zellen rasterisiert. In einem um das Ego-Fahrzeug zentrierten Ausschnitt des Rasters wird anhand von abstandsgebenden Sensoren die Belegtheit der Grid-Zellen geschätzt. Rot eingefärbte Grid-Zellen sind durch ein Hindernis belegt. Grüne Grid-Zellen werden als frei betrachtet. Weiße Grid-Zellen können nicht beobachtet werden, da ihre Sicht durch Hindernisse verdeckt ist.

Zusätzlich zur Schätzung von Belegtheitswahrscheinlichkeiten akkumulieren Geländekarten über die Zeit die höchsten und niedrigsten in eine Grid-Zelle fallenden 3D-Punktmessungen (aus LiDAR oder Stereo Matching). Daraus ergibt sich eine kombinierte Höhenkarte (engl. *Elevation Map*) von Hindernissen und Untergrund. Die Höhe eines Hindernisses wird aus der Differenz der minimalen und maximalen Höhe der entsprechenden Grid-Zelle bestimmt. Für die Modellierung des Bodens und die Ableitung von Terrain-Gradienten wird die minimale Höhe herangezogen.

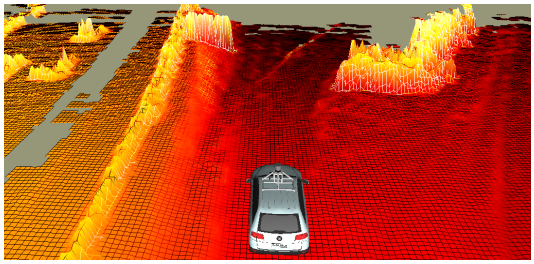
Problemstellung: Überhängende Hindernisse Auf Feldwegen stellt diese gemeinsame Kartierung von Hindernissen und Boden jedoch ein Problem dar. Wie in Abbildung 6.2b zu sehen ist, wird dadurch bei überhängenden Hindernissen – in diesem Fall einem Straßenschild – die gesamte Grid-Zelle als belegt markiert. Ragt nun beispielsweise ein Ast in die Straße, unter dem das Fahrzeug eigentlich problemlos durchfahren könnte, gilt die Grid-Zelle dennoch als durch ein Hindernis belegt und könnte durch das autonome Fahrzeug nicht befahren werden. Zwar könnten Punktmessungen über dem Ego-Fahrzeug verworfen werden, dies funktioniert in der Vorausschau jedoch nur bei ebenem Terrain. Zudem stehen unter Hindernissen keine Höheninformationen des Terrains zur Verfügung, wie sie jedoch im Wegetracking benötigt werden, um die Lage der Straße zu bestimmen.

Eine Lösungsidee bieten die hybriden Höhenkarten (engl. *Hybrid Elevation Maps*) von Douillard et al. [2010]: Ähnlich wie die Geländekarten bilden auch sie sowohl Hindernisse, als auch die Bodenoberfläche ab. Diese werden jedoch voneinander separiert: Für jede Grid-Zelle werden die minimale, die maximale und die mittlere z-Koordinate aller Punkte, die in die Grid-Zelle fallen, akkumuliert. Die mittlere Höhe dient als erste Schätzung der Bodenoberfläche. Anhand der Gradienten zwischen benachbarten Punkten werden daraus jedoch anschließend alle aus dem Terrain ragenden Hindernisse entfernt und ein hindernisbereinigtes Geländemodell erzeugt. Die Hindernisse werden stattdessen als separate Objekte modelliert. Hier dienen zunächst wie bei den Geländekarten die minimalen und maximalen Höhen zur Identifikation von Hindernissen. Zur Modellierung überhängender Hindernisse werden diese jedoch anschließend in dreidimensionalen Gittern rasterisiert, was die Entfernung des Bodens und die Modellierung überhängender Hindernisse ermöglicht.

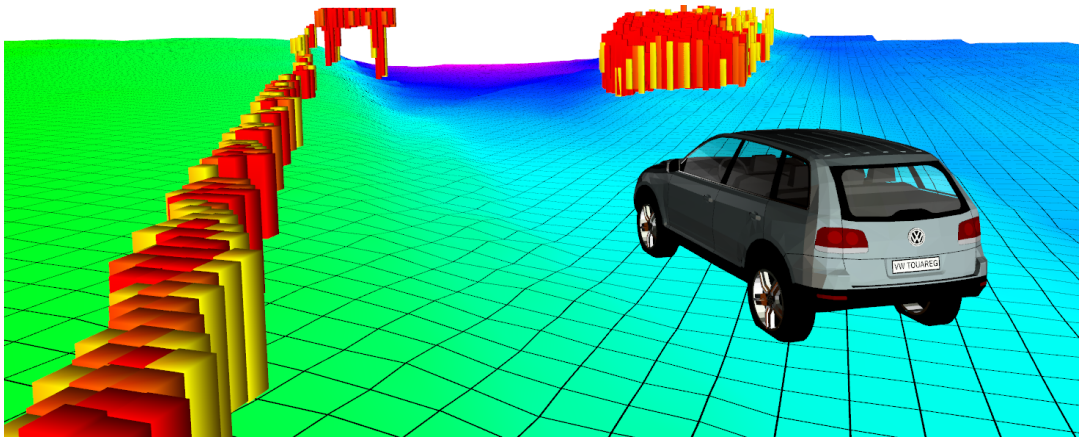
Trennung von Hindernis-Grid und Terrain-Schätzung Die Idee, Hindernisse und Boden unabhängig voneinander zu modellieren, soll hier aufgegriffen werden, um auch überhängende Hindernisse abbilden und darunter dennoch das Terrain modellieren zu können. Die beiden Aufgaben der Hindernis- und Bodenkartierung werden daher in dieser Arbeit getrennt. Das Umgebungsmodell setzt sich dadurch aus zwei separaten Grids zusammen: dem im Folgenden beschriebenen Hindernis-Grid, das alle positiven statischen Hindernisse enthält, und dem in Kapitel 7 geschätzten Terrain-Grid, das den Boden inklusive negativer Hindernisse (z. B. Löcher und Gräben) modelliert. Ein visueller Vergleich der aufgeteilten Grids zum bisherigen Umgebungsmodell ist in Abbildung 6.2 zu sehen. Durch die Aufteilung können für Hindernis-Grid und Terrain-Grid unterschiedliche Auflösungen gewählt werden. Während für das Terrain eine Auflösung von 0,5 m bis 1,5 m ausreicht, werden die Hindernisse mit einer Zell-Kantenlänge von 20 cm feiner aufgelöst.



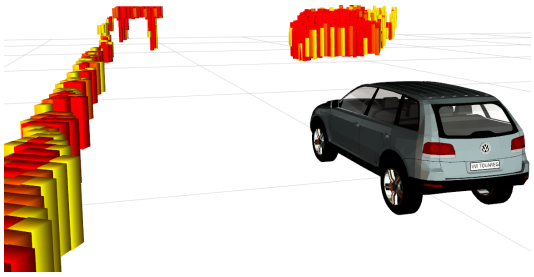
(a) RGB-Kamerabild als Referenz



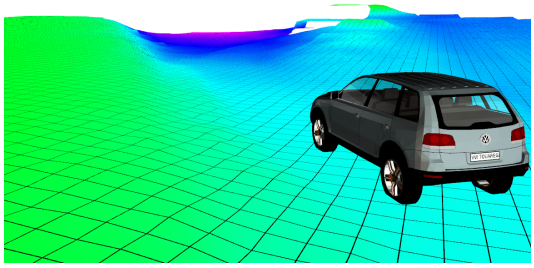
(b) Bisheriges Umgebungsmodell



(c) Neues Umgebungsmodell



(d) Hindernis-Grid



(e) Terrain-Schätzung

Abbildung 6.2:
Vergleich zwischen bisherigem und neuem Umgebungsmodell für die in (a) abgebildete Szene. Das in (b) gezeigte bisherige Umgebungsmodell enthält Boden und Hindernisse in einem Grid, farbkodiert von rot für niedrig zu gelb für hoch. Das in (c) gezeigte neue Umgebungsmodell besteht dagegen aus zwei getrennten Grids: (d) der Hinderniskarte (farbkodiert von gelb für eine Belegtheitswahrscheinlichkeit von 0,5 zu rot für eine Belegtheitswahrscheinlichkeit von 0,7 oder mehr), sowie (e) einem separatem Grid mit dem Ergebnis einer hindernisbereinigten Terrain-Schätzung (farbkodiert von lila für niedrig zu grün für hoch). Deutlich wird der Unterschied in der Modellierung des überhängenden Straßenschilds und der durchgängigen Terrain-Schätzung auch bei Abschattungen durch die Leitplanke.

Ausgangslage: LiDAR-basiertes Hindernis-Grid Die 3D-Punktwolke, anhand derer in einem Belegtheitsgitter die Belegtheit einer Grid-Zelle gemessen wird, kann entweder durch einen LiDAR-Sensor aufgenommen oder durch Stereo Matching berechnet werden. Ein Messmodell für ein stereobasiertes Umgebungsmodell wird von Jaspers [2021] vorgestellt. In Kapitel 5 wurde jedoch bereits festgestellt, dass die Entfernungsschätzungen des Stereo Matchings in größerer Entfernung unzureichend genau für die Hinderniskartierung sind. Daher wird im Folgenden analog zum bisherigen Umgebungsmodell ein LiDAR-basiertes Hindernis-Grid umgesetzt und für die Modellierung überhängender Hindernisse weiterentwickelt.

Problemstellung: Befahrbarkeit Auf Feldwegen stellt sich bei einem LiDAR-basierten Hindernis-Grid jedoch ein weiteres Problem: Oftmals wächst neben dem Weg hohes Gras, das in den Weg hinein ragt. Manchmal befindet sich sogar in der Mitte des Wegs ein Grünstreifen. Treffen dort LiDAR-Punkte auf höhere Grashalme, so wird im Hindernis-Grid die entsprechende Grid-Zelle als belegt markiert. Das autonome Fahrzeug könnte das Gras aber natürlich problemlos überfahren. Die LiDAR-Punkte messen also zwar die Belegtheit einer Grid-Zelle, es kann damit jedoch keine direkte Aussage darüber getroffen werden, ob die Grid-Zelle tatsächlich unpassierbar ist. Daher stellt sich die Herausforderung, zwischen befahrbaren und unbefahrbaren Hindernissen zu unterscheiden. In diesem Kapitel wird dazu untersucht, die Ergebnisse einer semantischen Segmentierung eines Kamerabilds in das Hindernis-Grid zu fusionieren, um zusätzliche Befahrbarkeitsinformationen zu erhalten.

Gliederung Nach der Literaturrecherche in Abschnitt 6.2 zur Modellierung von Hindernissen gliedert sich das Kapitel daher wie folgt: In Abschnitt 6.3 werden aus den LiDAR-Messungen Belegtheitswahrscheinlichkeiten berechnet. In Abschnitt 6.4 werden dann zusätzlich aus den Kamera-Messungen Befahrbarkeitswahrscheinlichkeiten bestimmt. Beide werden schließlich in Abschnitt 6.5 zu einer Hinderniswahrscheinlichkeit kombiniert. Zuletzt wird in Abschnitt 6.6 das Laufzeitverhalten des Hindernis-Grids untersucht.

6.2 Stand der Forschung

2D-Grids Die Verwendung zweidimensionaler Belegtheitsgitter zur probabilistischen Kartierung von Hindernissen geht zurück auf Moravec und Elfes [1985] (bzw. [Elfes, 1989]). Seit ihrem Einsatz auf dem Gewinner-Auto *Stanley* [Thrun et al., 2006] bei der DARPA Grand Challenge 2005, einem Wettbewerb unbemannter Roboterfahrzeuge in der Wüste, bilden 2D-Grids im autonomen Fahren die klassische Modellierungsform der statischen Umgebung. So verwendeten bei der nachfolgenden DARPA Urban Challenge 2007, bei der sich die Fahrzeuge im Stadtverkehr beweisen mussten, mehrere Teams 2D-Grids, darunter das Gewinner-Fahrzeug *Boss* [Urmson et al., 2008], das deutsche Team *AnnieWAY* [von Hundelshausen et al., 2008, Kammel et al., 2008] sowie das Fahrzeug *Junior* [Montemerlo et al., 2008]. Das Grundprinzip blieb seitdem bestehen. Weiterentwicklungen betreffen lediglich die Reduktion des Speicherbedarfs, beispielsweise durch die Anwendung räumlicher und zeitlicher Komprimierungsverfahren aus der Bildverarbeitung [Grewe et al., 2012].

2½D-Grids Oftmals enthalten Grid-basierte Darstellungen statischer Hindernisse zusätzlich Informationen über die Höhe der Hindernisse. Da die Hindernisse damit dreidimensional als Quader visualisiert werden können (vgl. Abbildung 6.2d), werden diese Belegtheitskarten mit Höheninformationen 2½D-Grids genannt. Diese Visualisierung wurde durch Badino et al. [2009] und Pfeiffer und Franke [2011] unter dem Namen „Stixel“ eingeführt. Darauf aufbauend ermöglichen Dryanovski et al. [2010] durch die Angabe einer Liste von Volumen die Modellierung mehrerer übereinander liegender Hindernisse in einer Grid-Zelle. Bouzouraa [2011] trackt darüber hinaus mehrere Hypothesen möglicher Hindernishöhen. Korah et al. [2011] bilden stattdessen in jeder Grid-Zelle ein Histogramm der gemessenen Höhen.

Moderne Belegtheitsgitter werden außer für die Schätzung einer Belegtheitswahrscheinlichkeit durch einen oder auch mehrere (vgl. [Matthaei et al., 2014]) abstandsgebende Sensoren gleichzeitig oft auch noch für andere Aufgaben verwendet. Mit Hilfe der in Abschnitt 6.1 beschriebenen, häufig simultanen Modellierung der Bodenoberfläche, dient das 2½D-Grid oft auch der Fahrbahn- (vgl. [Manz et al., 2011, Rieken et al., 2015]) und Kreuzungserkennung (vgl. [Bayerl und Wuensche, 2014]). Hierfür werden in dieser Arbeit jedoch dedizierte Module vorgestellt (Kapitel 8 und 9). Ferner werden Belegtheitsgitter auch zum Tracking dynamischer Objekte verwendet (vgl. [Tanzmeister et al., 2014, Berthold et al., 2020]). Diese basieren meist auf der Evidenztheorie, d. h. anstatt einer Belegtheitswahrscheinlichkeit werden eine Evidenz für die Belegtheit und eine Evidenz für die Freiheit der Grid-Zelle akkumuliert. Hierfür sei auf die Literaturrecherche in [Rieken, 2020] verwiesen.

Neben zweidimensionalen Grids (mit oder ohne Höheninformationen) gibt es inzwischen noch weitere, komplexere Möglichkeiten der Modellierung statischer Hindernisse, die im Folgenden kurz genannt werden sollen.

Voxel-Grids Bei Voxel-Grids, die durch Moravec [1996] und Roth-Tabak und Jain [1989] etabliert wurden, wird neben der x/y-Ebene auch die z-Achse rasterisiert und eine Belegtheitswahrscheinlichkeit für jeden Quader, angelehnt an das zweidimensionale Pixel (engl. *Picture Element*) im dreidimensionalen „Voxel“ (engl. *Volume Element*) genannt, geschätzt. Dies benötigt aufgrund der meist hohen Anzahl an Voxeln sehr viel Speicher und Rechenleistung. Der Speicherbedarf kann reduziert werden, indem in sog. Voxel-Listen, wie sie durch Ryde und Hu [2010] vorgestellt wurden, nur die Voxel gespeichert werden, für die mindestens eine Messung existiert.

Baumstrukturen Eine andere Möglichkeit die Anzahl gespeicherter Voxel zu reduzieren ist die Verwendung von Baumstrukturen, die es erlauben, verschiedene Bereiche des Raums unterschiedlich hoch aufzulösen. Bei Octrees wird jeder würfelförmige Voxel bei Bedarf in acht kleinere Voxel aufgeteilt. Große Freiräume (wie auch große Hindernisse, falls sie von allen Seiten beobachtet wurden) können dadurch mit sehr großen Voxeln modelliert werden, während für kleine Hindernisse und die Kanten von Hindernissen sehr kleine Voxel eine gute Genauigkeit bieten. Erstmalig für dreidimensionale Belegtheitskarten eingesetzt wurden Octrees durch Payeur et al. [1997], modernere Implementierungen erfolgten beispielsweise durch Schmid et al. [2010] und Wurm et al. [2010] bzw. Hornung et al. [2013].

Stochastische Prozesse Eine weitere Modellierungsart der statischen Umgebung stellen stochastische Prozesse dar. O'Callaghan und Ramos [2012] schlagen dafür Gauß-Prozesse vor, Ramos und Ott [2016] stellen sog. Hilbert Maps vor. Beide Darstellungen sind probabilistisch und durch die kontinuierliche Modellierung potenziell sehr genau, jedoch genau wie die dreidimensionalen Grids nicht echtzeitfähig.

Polygone und Linien Effizienter können Hindernisse durch ihre Kanten modelliert werden. Polygone und Polylinien, wie sie beispielsweise durch Kubertschak et al. [2014] verwendet werden, bieten in strukturierten Umgebungen ebenfalls eine gute Genauigkeit. Sie bilden eine Erweiterung früherer Ansätze der Modellierung der statischen Hindernisse durch zweidimensionale Linien, über die Sack und Burgard [2004] einen Überblick geben. Die Repräsentation durch Linien und linienförmige Polygone ist jedoch schlechter geeignet für geometrisch komplexere Hindernisse in unstrukturierter Umgebung wie Bäume und Büsche. Weiterhin kann die Schätzung der Belegtheitswahrscheinlichkeiten nicht in dieser Modellierung erfolgen, sondern erfordert ein zusätzliches 2D-Grid.

Modellierung des Freiraums Eine Vereinfachung der Liniendarstellung ist die indirekte Modellierung der statischen Hindernisse über den befahrbaren Freiraum, wie durch Schreier et al. [2013] und Nuss et al. [2014] implementiert. Dabei wird die Fahrbahn um das Ego-Fahrzeug herum mit einem Polygon modelliert, das bei statischen Hindernissen am Fahrbahnrand entsprechend verkleinert wird. Diese Darstellung ist damit nicht für das Fahren abseits der Straße geeignet und benötigt ebenfalls ein zusätzliches 2D-Grid. Auch die von Weiherer et al. [2013] vorgestellte Freiraummodellierung durch die laterale Unterteilung der Fahrbahn in Hindernis- und Freiraumintervalle ist nicht für das Fahren abseits gerader Straßen geeignet.

6.3 Schätzung von Belegtheitswahrscheinlichkeiten

Für eine effiziente und probabilistische Modellierung der statischen Hindernisse wird in diesem Kapitel ein bayessches 2½D-Belegtheitsgitter entwickelt, das der theoretischen Grundlage des bisherigen Umgebungsmodells des Instituts folgt. Im Gegensatz zur kombinierten Geländekarte aus [Jaspers et al., 2017] werden im Hindernis-Grid jedoch nur Punktmessungen auf Hindernissen betrachtet. Um überhängende Hindernisse zu modellieren, wird zudem nicht nur die maximale Ausdehnung eines Hindernisses nach oben, sondern auch seine minimale Höhe über dem Terrain kartiert. Diese 2½D-Darstellung ist im Gegensatz zu einem dreidimensionalen Grid sehr effizient. Die Kartierung der Hindernisse erfolgt im stationären, sprungfreien Koordinatensystem der Odometrie. Zur Implementierung der Hinderniskarte wurde die *Universal Grid Map Library* von Fankhauser und Hutter [2016] verwendet. Diese verwaltet Grids mit verschiedenen Ebenen und übernimmt die Indizierung und die Speicherverwaltung für sog. *Wrappable Maps*: Mittels eines zweidimensionalen Ringspeichers kann dabei der im Hindernis-Grid gezeigte Ausschnitt der Umgebung im Odometriekoordinatensystem mit dem Ego-Fahrzeug mitgeführt werden, ohne dass Daten verschoben werden müssen.

6.3.1 Bayes-Filter

Bei einem bayesschen Belegtheitsgitter wird der Zustand einer Grid-Zelle binär als entweder belegt (occ) oder nicht belegt modelliert und die Wahrscheinlichkeit $p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})$ dafür gesucht, dass eine Grid-Zelle c im Hinblick auf alle bisherigen Messungen $\mathbf{y}_{1:k}$ belegt ist. Dabei wird die Annahme getroffen, dass der Zustand einer Grid-Zelle unabhängig von allen anderen Grid-Zellen, der Fahrzeugpose und der Zeit ist. Nach dem Satz von Bayes zur bedingten Wahrscheinlichkeit von drei Ereignissen

$$p(A \mid B, C) = \frac{p(B \mid A, C)\; p(A \mid C)}{p(B \mid C)} \tag{6.1}$$

berechnet sich die Belegtheitswahrscheinlichkeit als

$$p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k}) = p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k, \mathbf{y}_{1:k-1}) = \frac{p(\mathbf{y}_k|\,\mathrm{occ}^{(\mathsf{c})}, \mathbf{y}_{1:k-1})\; p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})}{p(\mathbf{y}_k|\,\mathbf{y}_{1:k-1})}\,, \tag{6.2}$$

wobei gilt:

$$p(\mathbf{y}_k|\,\mathrm{occ}^{(\mathsf{c})}, \mathbf{y}_{1:k-1}) = p(\mathbf{y}_k|\,\mathrm{occ}^{(\mathsf{c})})\,, \tag{6.3}$$

da jede Messung unabhängig von allen vorherigen Messungen ist.

Das Messmodell $p(\mathbf{y}_k|\,\mathrm{occ}^{(\mathsf{c})})$, also die Wahrscheinlichkeit dafür, bei einer belegten Grid-Zelle eine bestimmte Messung zu erhalten, ist oft schwer aufzustellen. Daher wird der Satz von Bayes für zwei Ereignisse

$$p(A \mid B) = \frac{p(B \mid A)\; p(A)}{p(B)} \tag{6.4}$$

angewandt, um stattdessen das leichter zu bestimmende, inverse Sensormodell $p(\mathrm{occ}^{(\mathsf{c})}|\mathbf{y}_k)$, also die Wahrscheinlichkeit dafür, dass eine Grid-Zelle mit einer Messung $\mathbf{y}_k$ belegt ist, zu verwenden:

$$p(\mathbf{y}_k|\,\mathrm{occ}^{(\mathsf{c})}) = \frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)\; p(\mathbf{y}_k)}{p(\mathrm{occ}^{(\mathsf{c})})}\,. \tag{6.5}$$

Gleichungen (6.3) und (6.5) in Gleichung (6.2) eingesetzt, ergibt

$$p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k}) = \frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)\; p(\mathbf{y}_k)\; p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})}{p(\mathrm{occ}^{(\mathsf{c})})\; p(\mathbf{y}_k|\,\mathbf{y}_{1:k-1})}\,. \tag{6.6}$$

Um Gleichung (6.6) zu vereinfachen, wird das Chancenverhältnis (engl. *odds ratio*) $\frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})}{p(\neg\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})}$ der Belegtheit einer Grid-Zelle aufgestellt:

$$\frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})}{p(\neg\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})} = \frac{\dfrac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)\;\cancel{p(\mathbf{y}_k)}\;p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})}{p(\mathrm{occ}^{(\mathsf{c})})\;\cancel{p(\mathbf{y}_k|\,\mathbf{y}_{1:k-1})}}}{\dfrac{p(\neg\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)\;\cancel{p(\mathbf{y}_k)}\;p(\neg\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})}{p(\neg\mathrm{occ}^{(\mathsf{c})})\;\cancel{p(\mathbf{y}_k|\,\mathbf{y}_{1:k-1})}}} =$$

$$= \frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)}{p(\neg\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)} \frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})}{p(\neg\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})} \frac{p(\neg\mathrm{occ}^{(\mathsf{c})})}{p(\mathrm{occ}^{(\mathsf{c})})}. \tag{6.7}$$

Dadurch kürzen sich die Faktoren $p(\mathbf{y}_k)$ und $p(\mathbf{y}_k|\,\mathbf{y}_{1:k-1})$ weg. Da der Belegtheitszustand einer Grid-Zelle binär ist, gilt weiterhin

$$p(\neg\mathrm{occ}^{(\mathsf{c})}) = 1 - p(\mathrm{occ}^{(\mathsf{c})})\,, \tag{6.8}$$

und Gleichung (6.7) vereinfacht sich zu

$$\frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})}{1-p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})} = \frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)}{1-p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)} \frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})}{1-p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})} \frac{1-p(\mathrm{occ}^{(\mathsf{c})})}{p(\mathrm{occ}^{(\mathsf{c})})}. \tag{6.9}$$

Die A-priori-Wahrscheinlichkeit $p(\mathrm{occ}^{(\mathsf{c})})$ dafür, dass eine Grid-Zelle belegt ist, ohne dass weitere Informationen durch Messungen gegeben sind, wird auf $p_{\text{prior}} = 0{,}5$ gesetzt. Damit nimmt der letzte Term aus Gleichung (6.9) den Wert 1 an und die Gleichung reduziert sich auf

$$\frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})}{1-p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})} = \frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)}{1-p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)} \frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})}{1-p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})}\,. \tag{6.10}$$

Dies entspricht dem Korrekturschritt eines rekursiven Bayes-Filters.

Die Chance (engl. *Odds*) eines Ereignisses wird als $\mathrm{odds}(x) = \frac{p(x)}{1-p(x)}$ definiert. Die Belegtheitswahrscheinlichkeit als Chance ausgedrückt ergibt die odds-Repräsentation des Bayes-Filters:

$$\mathrm{odds}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k}) = \mathrm{odds}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k) \cdot \mathrm{odds}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})\,. \tag{6.11}$$

Die rechenaufwendige Multiplikation im Filterupdate lässt sich durch eine schnellere Addition ersetzen, indem auf beiden Seiten von Gleichung (6.11) der (natürliche) Logarithmus angewandt wird:

$$\begin{aligned}\log\Big(\mathrm{odds}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k})\Big) &= \log\Big(\mathrm{odds}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k) \cdot \mathrm{odds}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})\Big) = \\ &= \log\Big(\mathrm{odds}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)\Big) + \log\Big(\mathrm{odds}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})\Big)\,.\end{aligned} \tag{6.12}$$

Diesen natürlichen Logarithmus der Chance nennt man auch den Logit:

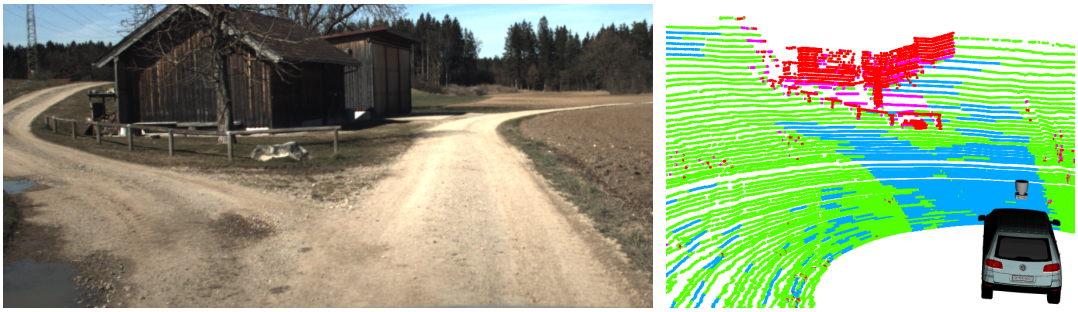
$$\mathrm{logit}(x) = \log(\mathrm{odds}(x)) = \log\left(\frac{p(x)}{1-p(x)}\right)\,. \tag{6.13}$$

Damit ergibt sich die log-odds-Repräsentation des Korrekturschritts eines rekursiven Bayes-Filters zur Schätzung der Belegtheitswahrscheinlichkeit einer Grid-Zelle zum Zeitschritt k anhand des Logits des vorherigen Zeitschritts $k-1$ und einer neuen Messung $\mathbf{y}_k$ als

$$\mathrm{logit}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k}) = \mathrm{logit}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k) + \mathrm{logit}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})\,. \tag{6.14}$$

6.3.2 Messungen aus der LiDAR-Punktwolke

Die Messungen $\mathbf{y}_k$ zur Belegtheit einer Grid-Zelle werden aus den Punkten einer LiDAR-Punktwolke gewonnen. Die eingehende LiDAR-Punktwolke ist bereits inertial korrigiert, d. h. sie wurde unter Berücksichtigung der Eigenbewegung des Fahrzeugs in das ortsfeste Koordinatensystem der Odometrie transformiert (vgl. Abschnitt 2.2). Um nun Hindernis- von Bodenpunkten zu unterscheiden, wird eine von Burger und Wuensche [2018] entwickelte, geometrische Segmentierung verwendet. Basierend auf den Gradienten zwischen horizontal und vertikal benachbarten Punkten ordnet sie jeden Punkt der LiDAR-Punktwolke einer der Klassen *Hindernis*, *Freiraum* oder *Bodenfläche* zu. Ein Beispiel für diese Segmentierung ist in Abbildung 6.3 zu sehen.



(a) RGB-Kamerabild als Referenz (b) Segmentierte Punktwolke

Abbildung 6.3:
Beispiel für die LiDAR-Punktwolken-Segmentierung von Burger und Wuensche [2018]. Bodenflächenpunkte werden in blau dargestellt, Freiraumpunkte in grün und Hindernispunkte in rot. In pink dargestellte Punkte wurden als Rauschen klassifiziert.

Aufgrund dieser Klassifizierung der Punkte als Hindernis oder Bodenfläche bzw. Freiraum wird anschließend entschieden, ob eine 20 cm $\times$ 20 cm große Grid-Zelle in dieser Messung als frei oder belegt gemessen wird oder unbeobachtet ist. Um die eingehende klassifizierte Punktwolke effizient zu verarbeiten, wurde eine modifizierte Version des Voxel-Grid-Filters der PCL-Bibliothek [Rusu und Cousins, 2011] implementiert. Das Ziel ist es dabei, die Eingangspunktwolke eines Zeitschritts k auf eine Messung $\mathbf{y}_k^{(\mathrm{c})}$ pro Grid-Zelle c zu reduzieren. Dazu werden die Punkte in einem ersten Schritt zunächst so sortiert, dass die Punkte, die in dieselbe Grid-Zelle fallen, benachbart im Speicher liegen. Dadurch werden die Punkte in dieselbe Cache-Zeile des Prozessors geladen, und es kann von einer deutlich geringeren Zugriffszeit profitiert werden, wenn in einem zweiten Durchlauf die jeweiligen LiDAR-Punkte in einer Grid-Zelle zu einer Messung zusammengefasst werden.

Eine Messung

$$\mathbf{y}_k^{(\mathrm{c})} = \left[\tilde{c}_k^{(\mathrm{c})} \;\; \tilde{h}_{\mathrm{min},k}^{(\mathrm{c})} \;\; \tilde{h}_{\mathrm{max},k}^{(\mathrm{c})}\right]^{\mathsf{T}} \tag{6.15}$$

besteht dabei aus einer Klassifizierung $\tilde{c}_k^{(\mathrm{c})}$, einer minimal gemessenen Höhe $\tilde{h}_{\mathrm{min},k}^{(\mathrm{c})}$ und der maximal gemessenen Höhe $\tilde{h}_{\mathrm{max},k}^{(\mathrm{c})}$. Die gemessene Klassifizierung der Grid-Zelle ergibt sich aus den einzelnen Klassifizierungen $\mathrm{class}(\cdot)$ der Punkte $\mathcal{P}_k^{(\mathrm{c})}$, die

in Zeitschritt k in die Grid-Zelle c fallen, gemäß der geometrischen Punktwolken-Segmentierung. Die Messung wird der Klasse *Hindernis* (o) zugeordnet, falls mindestens ein Hindernispunkt in die Zelle fällt, bzw. der Klasse *Boden* (g), falls lediglich Bodenflächen- und Freiraumpunkte in die Zelle fallen. Eine Zelle ohne Punktmessungen gilt zunächst als unbeobachtet (u). Dadurch ergibt sich die Klassifizierung

$$\tilde{c}_k^{(\mathsf{c})} = \begin{cases} \mathrm{o} & \exists\, \mathbf{p} \in \mathcal{P}_k^{(\mathsf{c})} : \mathrm{class}(\mathbf{p}) = \mathrm{Hindernis}\,, \\ \mathrm{g} & \nexists\, \mathbf{p} \in \mathcal{P}_k^{(\mathsf{c})} : \mathrm{class}(\mathbf{p}) = \mathrm{Hindernis} \wedge \mathcal{P}_k^{(\mathsf{c})} \neq \emptyset\,, \\ \mathrm{u} & \mathcal{P}_k^{(\mathsf{c})} = \emptyset\,. \end{cases} \tag{6.16}$$

Die minimal und maximal gemessenen Hindernishöhen berechnen sich aus den z-Koordinaten aller durch die Punktwolken-Segmentierung als Hindernis klassifizierten Punkte $\mathcal{H}_k^{(\mathsf{c})}$ im Odometriekoordinatensystem:

$$\tilde{h}_{\mathsf{min},k}^{(\mathsf{c})} = \min_{\mathbf{p} \in \mathcal{H}_k^{(\mathsf{c})}} (z_{\mathbf{p}})\,, \qquad \tilde{h}_{\mathsf{max},k}^{(\mathsf{c})} = \max_{\mathbf{p} \in \mathcal{H}_k^{(\mathsf{c})}} (z_{\mathbf{p}})\,. \tag{6.17}$$

6.3.3 Update der Belegtheitswahrscheinlichkeiten

Anhand der gemessenen Klassifizierung einer Grid-Zelle wird dann dem in [Burgard und Hebert, 2008, S. 855] beschriebenen, inversen Sensormodell und Jaspers et al. [2017] folgend die Wahrscheinlichkeit $p(\mathrm{occ}^{(\mathsf{c})} |\, \mathbf{y}_k)$ dafür bestimmt, dass eine Zelle c, gegeben die Messung $\mathbf{y}_k$ des Zeitschritts k, belegt ist:

$$p(\mathrm{occ}^{(\mathsf{c})} |\, \mathbf{y}_k) = \begin{cases} p_{\mathsf{occ}} & \text{wenn } \tilde{c}_k^{(\mathsf{c})} = \mathrm{o} \text{ und } \tilde{h}_{\mathsf{max},k}^{(\mathsf{c})} - \tilde{h}_{\mathsf{min},k}^{(\mathsf{c})} > \tau\,, \\ p_{\mathsf{free}} & \text{wenn } \tilde{c}_k^{(\mathsf{c})} = \mathrm{g}\,, \\ p_{\mathsf{prior}} & \text{andernfalls}\,. \end{cases} \tag{6.18}$$

Werden in einer Zelle mindestens zwei Hindernispunkte gemessen und übersteigt die Differenz der gemessenen Hindernishöhen einen Schwellwert τ, wird die Zelle mit $p_{\mathsf{occ}} = 0{,}95$ zu 95 % als belegt angenommen. Unter der Annahme, dass die meisten Hindernisse auf dem Boden liegen, wird der Schwellwert τ auf die Bodenfreiheit h_{gc} des Fahrzeugs gesetzt. Bei MuCAR-3 beträgt $h_{\mathsf{gc}} = 30\,\mathrm{cm}$. Für andere Fahrzeuge oder Anwendungen, bei denen bereits niedrigere Hindernisse problematisch wären, kann der Parameter angepasst werden. Werden in einer Zelle nur Boden- oder Freiraumpunkte gemessen, so wird die Zelle mit $p_{\mathsf{free}} = 0{,}4$ mit 60 % Wahrscheinlichkeit als frei angenommen. Die übrigen Zellen werden mit der A-priori-Belegtheitswahrscheinlichkeit von $p_{\mathsf{prior}} = 0{,}5$ gleichwahrscheinlich als belegt und als frei angenommen.

Oftmals werden nach dem Prinzip des zweidimensionalen Raytracings auch unbeobachtete Zellen auf einer Linie zwischen dem Sensor und einer belegten Zelle als frei angenommen, da davon ausgegangen wird, dass nähere Hindernisse den Laserstrahl sonst bereits vorher reflektiert hätten. Dies ist jedoch nur bei soliden, die gesamte Grid-Zelle belegenden Hindernissen der Fall. Durchlässige Hindernisse wie Zäune oder schmälere Hindernisse wie Pfosten blockieren nicht alle LiDAR-Strahlen in

dieser Grid-Zelle. Moderne LiDAR-Sensoren können zudem sogar mehrere Echos derselben Diode auflösen. Schließlich wird beim 2D-Raytracing ignoriert, dass der Strahl auch über oder unter einem Hindernis durchgegangen sein könnte. Ein dreidimensionales Raytracing, das die Höhe des Strahls berücksichtigt, wäre jedoch zu rechenaufwendig. Daher werden die entsprechenden Zellen hier als unbeobachtet angenommen. Mit hochauflösenden LiDAR-Sensoren kann Freiraum auch allein über die gemessenen Bodenpunkte bereits in großer Entfernung erkannt und über die Zeit und die Bewegung verifiziert und verdichtet werden. Dennoch kann das Raytracing optional aktiviert werden, falls es für den Anwendungsfall geeignet ist.

Mit den Messungen $\mathbf{y}_k$ und dem inversen Sensormodell $p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)$ können nun mittels des in Abschnitt 6.3.1 hergeleiteten, rekursiven Bayes-Filters die Belegtheitswahrscheinlichkeiten der einzelnen Grid-Zellen korrigiert werden. Aus dem Logit des inversen Sensormodells der neuen Messung $\mathbf{y}_k$ aus dem Zeitschritt k und dem gespeicherten Logit $\mathrm{logit}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})$ der Belegtheitswahrscheinlichkeit aus dem vorherigen Zeitschritt $k-1$ (bzw. im ersten Zeitschritt dem Logit der A-priori-Wahrscheinlichkeit $p(\mathrm{occ}^{(\mathsf{c})}) = p_{\mathsf{prior}} = 0{,}5$) wird gemäß Gleichung (6.14) der korrigierte Logit der Belegtheitswahrscheinlichkeit für den Zeitschritt k berechnet:

$$\mathrm{logit}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k}) = \log\left(\frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)}{1-p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)}\right) + \mathrm{logit}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k-1})\,. \tag{6.19}$$

Alternativ, wenn nicht mehr beobachtete Hindernisse nach einer gewissen Zeit vergessen werden sollen, kann, Jaspers et al. [2017] folgend, mittels eines Decay-Faktors γ ein exponentielles Vergessen realisiert werden:

$$\mathrm{logit}(\mathrm{occ}^{(\mathsf{c})}|\mathbf{y}_{1:k}) = \gamma \log\left(\frac{p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)}{1-p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k)}\right) + (1-\gamma)\,\mathrm{logit}(\mathrm{occ}^{(\mathsf{c})}|\mathbf{y}_{1:k-1})\,, \tag{6.20}$$

wobei $\gamma = 0{,}1$ gesetzt wird. Die Logits können zur Weiterverwendung anschließend in Belegtheitswahrscheinlichkeiten

$$p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k}) = 1 - \frac{1}{1+\exp(\mathrm{logit}(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_{1:k}))} \tag{6.21}$$

umgerechnet werden, wobei intern die Logit-Darstellung für die Verwendung im nächsten Zeitschritt gespeichert wird.

Zuletzt werden die minimale und die maximale Höhe $h^{(\mathsf{c})}_{\mathsf{min}}$ bzw. $h^{(\mathsf{c})}_{\mathsf{max}}$ eines potenziellen Hindernisses bestimmt. Diese Hindernishöhen werden für die als belegt gemessenen Grid-Zellen, also falls $p(\mathrm{occ}^{(\mathsf{c})}|\,\mathbf{y}_k) = p_{\mathsf{occ}}$, rekursiv anhand der gemessenen Höhen $\tilde{h}^{(\mathsf{c})}_{\mathsf{min},k}$ bzw. $\tilde{h}^{(\mathsf{c})}_{\mathsf{max},k}$ akkumuliert:

$$h^{(\mathsf{c})}_{\mathsf{min},k} = \min\Big(h^{(\mathsf{c})}_{\mathsf{min},k-1},\ \tilde{h}^{(\mathsf{c})}_{\mathsf{min},k}\Big)\,, \tag{6.22}$$

$$h^{(\mathsf{c})}_{\mathsf{max},k} = \max\Big(h^{(\mathsf{c})}_{\mathsf{max},k-1},\ \tilde{h}^{(\mathsf{c})}_{\mathsf{max},k}\Big)\,. \tag{6.23}$$

Die Aktualisierung der Hindernishöhen kann wie auch das Filterupdate in Matrix-Operationen effizient für das gesamte Grid gleichzeitig berechnet werden.



(a) Kamerabild (b) Hindernis-Grid

Abbildung 6.4:
Beispiel für den Schweif, den ein Führungsfahrzeug im Hindernis-Grid hinterlässt, wenn dynamische Objekte nicht gesondert behandelt werden. Zur Verdeutlichung ist das exponentielle Vergessen hier deaktiviert.

6.3.4 Behandlung dynamischer Objekte

Bislang wurde angenommen, dass alle Punkte, die als Hindernispunkt klassifiziert wurden, zu einem statischen Hindernis gehören. So wurden alle als Hindernis klassifizierten Punkte in das Hindernis-Grid eingefügt. Allerdings könnte ein Hindernispunkt auch von einem dynamischen Objekt wie einem anderen Fahrzeug stammen. Diese sollten nicht Teil des statischen Umgebungsmodells sein. Zum einen soll zwischen statischen Hindernissen, die den Weg blockieren und ein Umplanen des Pfades erfordern würden, und dynamischen Hindernissen, die sich im Straßenverkehr bewegen, unterschieden werden. Zum anderen hinterlassen dynamische Objekte über die Zeit zusätzlich einen Schweif im Grid, wie in Abbildung 6.4 zu sehen ist. Daher müssen dynamische Objekte gezielt behandelt und aus der statischen Hinderniskarte entfernt werden. Hierfür können die Ergebnisse eines Einzel- oder Multi-Objekt-Trackers, der die Position, Lage und Dynamik eines oder mehrerer dynamischer Objekte über die Zeit schätzt, verwendet werden. Dazu wurden Schnittstellen zu dem am Institut eingesetzten Multi-Objekt-Tracker von Reich und Wuensche [2021] sowie einem darauf basierenden Einzel-Objekt-Tracker implementiert.

6.3.4.1 Ignorieren getrackter dynamischer Objekte

Existiert eine Schätzung für den Zustand des Objekts, so kann daraus ein umschließendes Rechteck (engl. *Bounding Box*) des Objekts zum Zeitpunkt der Messung prädiziert werden. Dafür werden konstante Linear- $(\dot{x}, \dot{y}, \dot{z})$ und Winkelgeschwindigkeiten $(\dot{\Phi}, \dot{\Theta}, \dot{\Psi})$ angenommen, um die Homogene Transformationsmatrix (HTM)

$$ {}^{t_k}\mathbf{H}_{t_{\text{Track}}} = \underline{\mathbf{T}}(\dot{x}\Delta t, \dot{y}\Delta t, \dot{z}\Delta t) \cdot \underline{\mathbf{R}}(\dot{\Phi}\Delta t, \dot{\Theta}\Delta t, \dot{\Psi}\Delta t), \quad \Delta t = t_k - t_{\text{Track}} \tag{6.24} $$

zwischen den Objektpositionen zu den Zeitpunkten t_{Track} des letzten Zustandsupdates des dynamischen Objekts und t_k der Punktwolken-Messung zu berechnen (vgl.

Abschnitt 2.2). Aus dem mittels ${}^{t_k}\mathbf{H}_{t_{\mathsf{Track}}}$ für den Zeitpunkt t_k prädizierten, umschließenden Rechteck wird dann die Menge $\mathcal{D}$ der durch das dynamische Objekt belegten Grid-Zellen bestimmt. Um nun das dynamische Objekt nicht in das Hindernis-Grid einzutragen, werden alle Messungen in diesen Grid-Zellen kurzzeitig ignoriert, indem die Zellen als unbeobachtet angenommen werden:

$$p(\mathrm{occ}^{(\mathsf{c})} |\, \mathbf{y}_k) = p_{\mathsf{prior}} \qquad \forall \mathsf{c} \in \mathcal{D}\,. \tag{6.25}$$

Somit werden keine alten Informationen gelöscht, diese werden lediglich zeitweise nicht mehr durch neue Messungen korrigiert. Dadurch bleiben bereits vor Vorbeifahren des dynamischen Objekts beobachtete Hindernisse erhalten.

6.3.4.2 Löschen des hinterlassenen Schweifs eines dynamischen Objekts vor Aufsetzen des Trackers

Typischerweise braucht das Objekt-Tracking jedoch einige Zyklen, um ein dynamisches Objekt zu erkennen und einen Track aufzusetzen. In dieser Zeit hat das Objekt oft bereits einen Schweif im Hindernis-Grid hinterlassen. Auch wenn zukünftige Messungen ignoriert werden, bleibt ein störender Rückstand aus den Messungen vor dem Aufsetzen des Trackers. Die einzige Möglichkeit diesen hinterlassenen Schweif dennoch zu entfernen ist es, die Position des Objekts einige Sekunden zurückzuprädizieren, die Belegtheitswahrscheinlichkeit der betroffenen Zellen zu löschen und wieder auf die A-priori-Wahrscheinlichkeit zu initialisieren:

$$p(\mathrm{occ}^{(\mathsf{c})} |\, \mathbf{y}_{1:k}) = p_{\mathsf{prior}} \qquad \forall \mathsf{c} \in \mathcal{D}\,. \tag{6.26}$$

Dadurch werden jedoch auch alle vor Auftreten des dynamischen Objekts beobachteten Hindernisse in diesem Bereich gelöscht. Daher muss dazu der Bereich weiterhin gut durch Messungen beobachtbar sein, sodass eventuelle Hindernisse im nächsten Zeitschritt (bzw. wenn die Zelle nicht mehr im aufgrund des dynamischen Objekts ignorierten Bereich liegt) wieder in die Hinderniskarte eingetragen werden. Um die Unsicherheit in der Schätzung der Dynamik des Objekts im ersten Zeitschritt des Trackings auszugleichen, werden alle Schätzungen in der ersten Sekunde zurückprädiziert und ein etwas größerer Bereich um das dynamische Objekt herum gelöscht. Auf diese Weise kann ein Schweif von bis zu 5 s vor der Trackinitialisierung rückstandslos aus dem Hindernis-Grid entfernt werden, wie das Beispiel in Abbildung 6.5 zeigt.

6.4 Bestimmung einer Befahrbarkeitswahrscheinlichkeit

Die aus der geometrisch segmentierten LiDAR-Punktwolke gewonnene Belegtheitswahrscheinlichkeit gibt nun an, ob in einer Grid-Zelle oberhalb des Bodens LiDAR-Punkte reflektiert wurden, die nicht von einem (bekannten) dynamischen Objekt stammen. Dass in einer Grid-Zelle LiDAR-Punkte reflektiert wurden, muss aber dennoch nicht zwangsläufig heißen, dass diese Grid-Zelle für das autonome Fahrzeug unpassierbar ist. Abseits der geteerten Straße stammen die Reflektionen oftmals auch

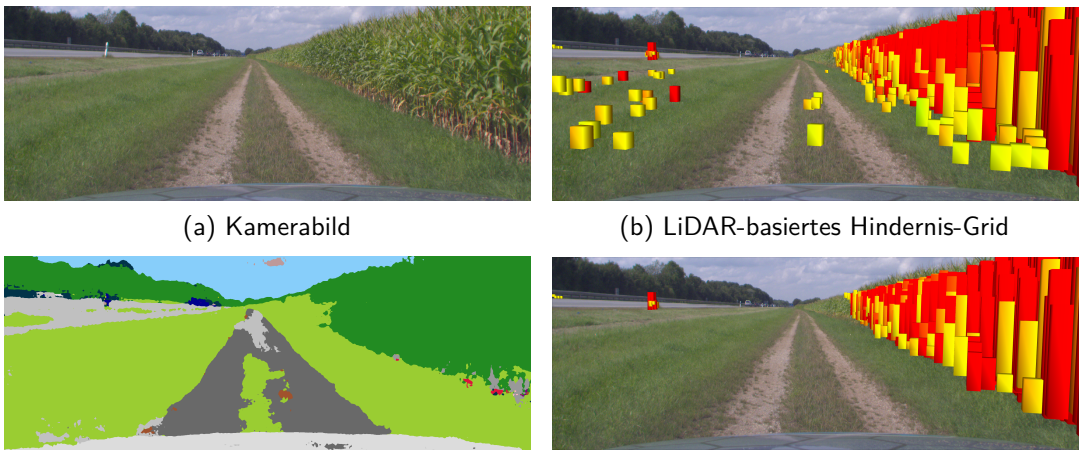


(a) Ignorieren neuer Messungen (b) Löschen der alten Spur

Abbildung 6.5:
Beispiel für die Behandlung eines dynamischen Objekts: (a) Anhand des Tracks eines Einzel-Objekt-Trackers (rotes Rechteck) wird ein etwas größeres umschließendes Rechteck für den Zeitpunkt der Messung prädiziert (grünes Rechteck), innerhalb dessen neue Messungen ignoriert werden. Dadurch endet der Schweif zu dem Zeitpunkt, an dem der Tracker aufgesetzt wurde. Um den Abbruch des Schweifs besser zu visualisieren, wurde das exponentielle Vergessen deaktiviert und das Ego-Fahrzeug vorwärts gefahren, obwohl die Fahrspur vermeintlich belegt ist. (b) Durch Rückprädiktion wird auch der vor dem Aufsetzen des Trackers hinterlassene Schweif gelöscht.

nur von hohem Gras. Auch wenn nicht durch das Gelände gefahren, sondern einem Feldweg gefolgt werden soll, kann dies bei Fahrzeugen mit einer geringen Bodenfreiheit vor allem bei Grünstreifen in der Mitte des Wegs oder an der Seite in den Weg ragenden Grasbüscheln zu Problemen führen. Wird die Grid-Zelle hier durch LiDAR-Messungen auf hohem Gras als belegt markiert, gilt der Weg als blockiert. Das Gras könnte jedoch eigentlich überfahren werden. Daher wird nach einer Möglichkeit gesucht, zwischen unbefahrbaren Hindernissen, die das Fahrzeug tatsächlich blockieren, und (zur Not) befahrbarer Vegetation zu unterscheiden. Metzger et al. [2021] haben dafür eine Oberflächenklassifikation in Form einer semantischen Segmentierung von Kamerabildern implementiert. Nun gilt es zu untersuchen, ob diese kamerabasierte Segmentierung in das LiDAR-basierte Hindernis-Grid fusioniert werden kann, um zwischen befahrbaren und nicht-befahrbaren Hindernissen zu unterscheiden.

Abbildung 6.6 zeigt eine Beispielszene, die im Folgenden zur Veranschaulichung verwendet werden soll. Rechts des Wegs befindet sich ein Maisfeld, das nicht befahren werden kann. Links des Wegs sowie auf einem schmalen Streifen in der Mitte des Weg befinden sich lediglich einige höhere Grashalme, über die der Versuchsträger fahren könnte. Diese haben jedoch eine signifikante Höhe, sodass die LiDAR-Punktwolken-Segmentierung darauf Hindernispunkte klassifiziert und bei einer reduzierten Bodenfreiheit von $h_{\mathrm{gc}} = 20\,\mathrm{cm}$ die entsprechenden Grid-Zellen im Hindernis-Grid als belegt markiert werden. In der semantischen Segmentierung des Kamerabilds werden die entsprechenden Pixel jedoch korrekt als befahrbare Vegetation klassifiziert und sind so eindeutig unterscheidbar von dem als nicht-befahrbar klassifizierten Maisfeld oder dem Leitpfosten. Diese Information soll nun im Hindernis-Grid verwendet werden.



(a) Kamerabild (b) LiDAR-basiertes Hindernis-Grid

(c) Semantisch segmentiertes Kamerabild (d) Fusioniertes Hindernis-Grid

Abbildung 6.6:
Beispiel für die Notwendigkeit der Unterscheidung zwischen befahrbarer und nicht-befahrbarer Vegetation bei Fahrzeugen mit einer geringen Bodenfreiheit. Dafür wird hier von einer Bodenfreiheit von $h_{\text{gc}} = 20\,\text{cm}$ ausgegangen. Abbildung (a) zeigt ein Kamerabild der Szene. Durch Verwendung der geometrisch segmentierten LiDAR-Punktwolke werden nicht nur das unbefahrbare Maisfeld rechts, sondern auch höhere Grashalme in das in (b) abgebildete Hindernis-Grid eingetragen. Damit liegen Hindernisse auf dem Weg. Mittels des in (c) gezeigten semantisch segmentierten Kamerabilds könnte hier jedoch zwischen befahrbarer Vegetation (hellgrün) und nicht-befahrbarer Vegetation (dunkelgrün) unterschieden werden. Farblegende: sky, asphalt, gravel, soil, grass, forest, car, ego_vehicle. Abbildung (d) zeigt das Ergebnis der im Folgenden vorgestellten Fusion des semantisch segmentierten Kamerabilds in das Hindernis-Grid. Die Hindernisse erscheinen nicht in der vollen Höhe, da die Punktwolken-Segmentierung die Punktwolke nach oben hin abschneidet.

6.4.1 Von Klassenwahrscheinlichkeiten zu einer Befahrbarkeitswahrscheinlichkeit

Das Ergebnis einer semantischen Segmentierung ist für jedes Pixel eines Kamerabilds ein Wahrscheinlichkeitsvektor

$$p = \left[p_{\text{asphalt}}\ p_{\text{gravel}}\ p_{\text{sand}}\ p_{\text{soil}}\ p_{\text{grass}}\ p_{\text{forest}}\ p_{\text{static}}\ p_{\text{dynamic}}\right] \tag{6.27}$$

mit den einzelnen Wahrscheinlichkeiten dafür, dass das Pixel verschiedenen, im Vorfeld definierten Klassen angehört. Die Summe aller Klassenwahrscheinlichkeiten ergibt dabei 1. Mögliche Klassen in der Vegetationserkennung von Metzger et al. [2021] sind *asphalt*, *gravel*, *sand* und *soil* für Wege, und *grass* und *forest* für befahrbare und nicht-befahrbare Vegetation. Weitere Hindernis-Klassen seien hier als p_{static} und p_{dynamic} zusammengefasst.

Klassisch wird meist nur die wahrscheinlichste Klasse betrachtet, wie sie auch in Abbildung 6.6c abgebildet ist. Dies ist jedoch fehleranfällig: Der in Abbildung 6.6 dargestellte Feldweg ist nicht eindeutig als solcher klassifizierbar, bei vielen Pixeln weist die Klasse *asphalt* die höchste Wahrscheinlichkeit auf, bei anderen dagegen die Klasse *gravel*, bei einzelnen auch *soil*. Aus diesem Grund soll nicht nur die

wahrscheinlichste Klasse betrachtet werden. Stattdessen werden aus den einzelnen Klassenwahrscheinlichkeiten die Wahrscheinlichkeiten

$$p_{\text{road}} = p_{\text{asphalt}} + p_{\text{gravel}} + p_{\text{sand}} + p_{\text{soil}} \,, \tag{6.28}$$

und

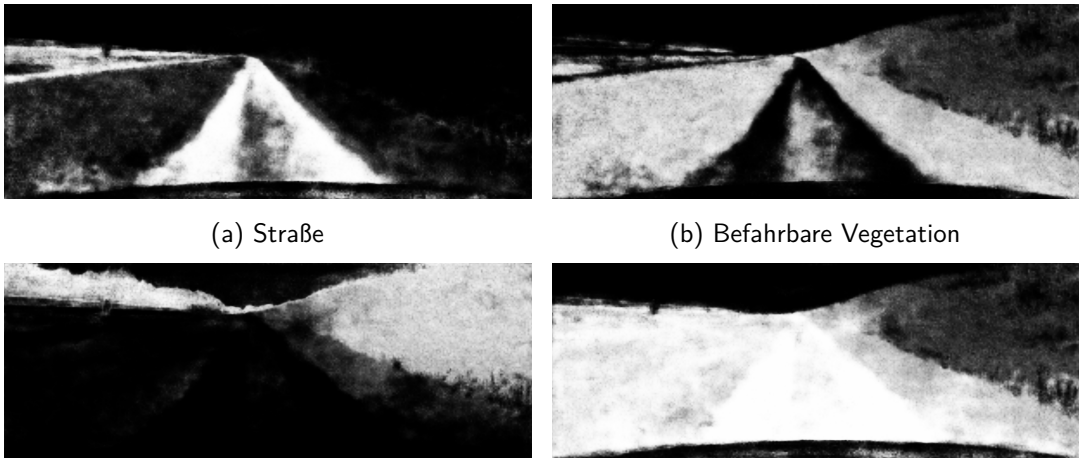
$$p_{\text{vegetation_traversable}} = p_{\text{grass}} \tag{6.29}$$

dafür berechnet, dass es sich unabhängig vom genauen Untergrund um Straße bzw. einen Weg oder befahrbare Vegetation handelt. Die Abbildungen 6.7a bis 6.7c visualisieren diese Wahrscheinlichkeiten in der Beispielszene.

Um im Hindernis-Grid zwischen befahrbaren und nicht-befahrbaren Grid-Zellen zu unterscheiden, werden die Straßenwahrscheinlichkeit p_{road} und die Wahrscheinlichkeit für befahrbare Vegetation $p_{\text{vegetation_traversable}}$ zu einer allgemeinen Befahrbarkeitswahrscheinlichkeit

$$p_{\text{traversability}} = p_{\text{road}} + p_{\text{vegetation_traversable}} \tag{6.30}$$

addiert (siehe auch Abbildung 6.7d). Diese Befahrbarkeitswahrscheinlichkeit stellt die Schnittstelle von der semantischen Segmentierung zum Hindernis-Grid dar und findet später auch in der Terrain-Schätzung Einsatz. Mit einer semantischen Segmentierung, die sich nicht wie die von Metzger et al. [2021] auf die Klassifikation von Vegetation, sondern auf die Erkennung von Wegen spezialisiert, könnte die Befahrbarkeitswahrscheinlichkeit als Schnittstelle auch für das Straßen- und Wegetracking oder die Kreuzungserkennung verwendet werden.



(a) Straße (b) Befahrbare Vegetation

(c) Nicht-befahrbare Vegetation (d) Befahrbarkeitswahrscheinlichkeit

Abbildung 6.7:
Aus den einzelnen Klassenwahrscheinlichkeiten der semantischen Segmentierung von Metzger et al. [2021] zusammengesetzte Pixelwahrscheinlichkeiten für Klassen der Kategorien (a) Straße, (b) befahrbare Vegetation und (c) nicht-befahrbare Vegetation für das in Abbildung 6.6 gezeigte Beispiel. Aus der Summe der Straßen-Klassenwahrscheinlichkeiten mit denen der befahrbaren Vegetation kann die in (d) gezeigte Befahrbarkeitswahrscheinlichkeit abgeleitet werden. Helle Bereiche zeigen jeweils eine hohe Wahrscheinlichkeit.

6.4.2 Integration in das Hindernis-Grid

Diese Befahrbarkeitswahrscheinlichkeitsmessungen werden nun in das Hindernis-Grid fusioniert, indem die einzelnen Hindernis-Stixel mit dem Befahrbarkeitswahrscheinlichkeitsbild eingefärbt werden. Dazu werden die Mittelpunkte (x- und y-Koordinaten) der Hindernis-Grid-Zellen (gemäß der anhand der Eigenbewegungsschätzung und der Kalibrierung geschätzten Kamerapose im Odometriekoordinatensytem zum Zeitpunkt der Bildaufnahme) in das Bild projiziert. Um das Bild nicht nur an einem willkürlichen Pixel abzutasten und um auch zu berücksichtigen, dass oft nicht das gesamte Hindernis im Kamerabild sichtbar ist, werden verschiedene z-Koordinaten der Hindernisse verwendet: Zwischen $h_{\mathsf{min}}^{(\mathsf{c})}$ und $h_{\mathsf{max}}^{(\mathsf{c})}$ wird das Hindernis auf $n = 5(h_{\mathsf{max}}^{(\mathsf{c})} - h_{\mathsf{min}}^{(\mathsf{c})})/0{,}2\,\mathrm{m}$ zufälligen Höhen abgetastet und der Mittelwert der Befahrbarkeitswahrscheinlichkeiten als Messung $\mathbf{y}_{\mathsf{sem},k}$ verwendet. Durch die zufällige Auswahl von z-Koordinaten werden in jedem Zeitschritt andere Pixel ausgewertet, und ein örtliches Rauschen in der Befahrbarkeitswahrscheinlichkeitsmessung wird durch räumliche und zeitliche Mittelung herausgefiltert. Die Anzahl gemittelter Werte hängt von der Hindernishöhe und der Gridauflösung ab. Für den 20 cm hohen Grashalm im Beispiel werden durch diese Abtastung fünf Pixel betrachtet. Dies ist nötig, da die Ausgabe der semantischen Segmentierung stark rauscht.

Um mögliche Verdeckungen zu berücksichtigen, wird beim Einfärben vom LiDAR-Sensor ausgehend radial entlang des Sehstrahls vorgegangen und bei Grid-Zellen mit einer Befahrbarkeitswahrscheinlichkeit kleiner als 0,5, die also nicht befahrbar und damit vermutlich auch nicht durchsichtig sind, abgebrochen. Auf diese Art und Weise wird für jede jemals belegte Zelle des Hindernis-Grids die Wahrscheinlichkeit

$$p(\mathrm{trav}^{(\mathsf{c})} |\, \mathbf{y}_{\mathsf{sem},k}) = p_{\mathsf{traversability}}(\mathsf{c}) \tag{6.31}$$

dafür bestimmt, dass eine Zelle c mit den gegebenen Messungen $\mathbf{y}_{\mathsf{sem},k}$ zum Zeitschritt k befahrbar ist. Nicht im Kamerabild beobachtbare Hindernis-Zellen erhalten eine A-priori-Befahrbarkeitswahrscheinlichkeit von $p(\mathrm{trav}^{(\mathsf{c})} |\, \mathbf{y}_{\mathsf{sem},k}) = 0{,}5$.

Anhand dieser Messungen werden die Befahrbarkeitswahrscheinlichkeiten analog zu den Belegtheitswahrscheinlichkeiten (vgl. Abschnitt 6.3.3) mittels eines Bayes-Filters über die Zeit geschätzt. Im Filterupdate werden dabei erneut über die log-odds-Repräsentation des Bayes-Filters rekursiv die Logits

$$\mathrm{logit}(\mathrm{trav}^{(\mathsf{c})} |\, \mathbf{y}_{\mathsf{sem},1:k}) = \log\left(\frac{p(\mathrm{trav}^{(\mathsf{c})} |\, \mathbf{y}_{\mathsf{sem},k})}{1 - p(\mathrm{trav}^{(\mathsf{c})} |\, \mathbf{y}_{\mathsf{sem},k})}\right) + \mathrm{logit}(\mathrm{trav}^{(\mathsf{c})} |\, \mathbf{y}_{\mathsf{sem},1:k-1}) \tag{6.32}$$

bestimmt, woraus sich die Befahrbarkeitswahrscheinlichkeiten berechnen lassen:

$$p(\mathrm{trav}^{(\mathsf{c})} |\, \mathbf{y}_{1:k}) = 1 - \frac{1}{1 + \exp\Big(\mathrm{logit}(\mathrm{trav}^{(\mathsf{c})} |\, \mathbf{y}_{\mathsf{sem},1:k})\Big)}\,. \tag{6.33}$$

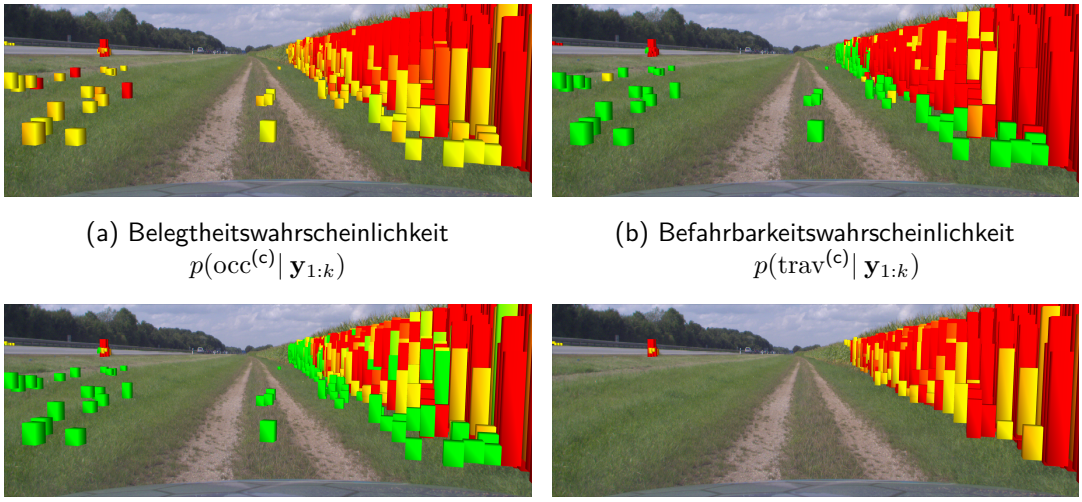
Abbildung 6.8b visualisiert die auf diese Weise bestimmten Befahrbarkeitswahrscheinlichkeiten für die Beispielszene.

6.5 Fusion zu einer Hinderniswahrscheinlichkeit

Anhand der Belegtheitswahrscheinlichkeit $p(\mathrm{occ}^{(c)}|\,\mathbf{y}_{1:k})$ aus der geometrischen Segmentierung der LiDAR-Punktwolke (Abbildung 6.8a) und der Befahrbarkeitswahrscheinlichkeit $p(\mathrm{trav}^{(c)}|\,\mathbf{y}_{1:k})$ aus der semantischen Segmentierung des Kamerabilds (Abbildung 6.8b) wird zuletzt eine kombinierte Hinderniswahrscheinlichkeit

$$p(\mathrm{obstacle}^{(c)}|\,\mathbf{y}_{1:k},\ \mathbf{y}_{\mathrm{sem},1:k}) = p(\mathrm{occ}^{(c)}|\,\mathbf{y}_{1:k})(1 - p(\mathrm{trav}^{(c)}|\,\mathbf{y}_{1:k})) \tag{6.34}$$

berechnet (Abbildung 6.8c). Diese gibt die Wahrscheinlichkeit dafür an, dass eine Grid-Zelle ein für das entsprechende Fahrzeug nicht passierbares Hindernis enthält. Im Beispiel ist die Hinderniswahrscheinlichkeit des hohen Grases nun sehr viel kleiner als die des Maisfelds, und der Weg wird als frei befahrbar erkannt. So ergibt sich als Ausgabe erkannter Hindernisse letztlich das Hindernis-Grid in Abbildung 6.8d, das nur den Leitpfosten und das Maisfeld enthält.



(a) Belegtheitswahrscheinlichkeit $p(\mathrm{occ}^{(c)}|\,\mathbf{y}_{1:k})$

(b) Befahrbarkeitswahrscheinlichkeit $p(\mathrm{trav}^{(c)}|\,\mathbf{y}_{1:k})$

(c) Hinderniswahrscheinlichkeit $p(\mathrm{obstacle}^{(c)}|\,\mathbf{y}_{1:k},\ \mathbf{y}_{\mathrm{sem},1:k})$

(d) Hinderniswahrscheinlichkeit mit Schwellwert $p(\mathrm{obstacle}^{(c)}|\,\mathbf{y}_{1:k},\ \mathbf{y}_{\mathrm{sem},1:k}) > 0{,}5$

Abbildung 6.8:
Hindernis-Grid für die in Abbildung 6.6 gezeigte Szene unter Verwendung der in Abbildung 6.7 gezeigten Befahrbarkeitswahrscheinlichkeit aus dem semantisch segmentierten Kamerabild. Grün visualisierte Grid-Zellen weisen eine hohe Befahrbarkeits- bzw. niedrige Belegtheits- und Hinderniswahrscheinlichkeiten auf, rot visualisierte Grid-Zellen entsprechend umgekehrt. Gelb entspricht einer jeweiligen Wahrscheinlichkeit von 0,5. (a) Aus der geometrischen LiDAR-Punktwolken-Segmentierung abgeleitete Belegtheitswahrscheinlichkeit. Hier wird hohes Gras als belegt erkannt. (b) Mit dem semantisch segmentierten Kamerabild bestimmte Befahrbarkeitswahrscheinlichkeit. Hier wird das Maisfeld rechts (bis auf vorgelagerte Grasbüschel) als eindeutig nicht-befahrbar, das Gras aber als eindeutig befahrbar erkannt. (c) Die aus Belegtheitswahrscheinlichkeit und Befahrbarkeitswahrscheinlichkeit fusionierte Hinderniswahrscheinlichkeit. Beim Maisfeld entspricht diese der Belegtheitswahrscheinlichkeit, der Weg und das Gras links werden dagegen als hindernisfrei erkannt. (d) Die verbleibenden erkannten Hindernisse, wenn Grid-Zellen mit einer Hinderniswahrscheinlichkeit von mehr als 50 % als erkannte Hindernisse ausgegeben werden.

6.6 Laufzeitverhalten

Um das Hindernis-Grid auf dem autonomen Fahrzeug einsetzen zu können, ist zuletzt noch die Echtzeitfähigkeit der Implementierung von Bedeutung. Die benötigte Rechenzeit hängt von der gewählten Größe des Hindernis-Grids ab. Abbildung 6.9a wertet daher die mittleren Laufzeiten der einzelnen Berechnungsschritte des Hindernis-Grids (ohne die vorhergehende Punktwolken-Segmentierung) in Abhängigkeit von der Grid-Größe aus. Erwartungsgemäß steigt die Gesamtlaufzeit mit der Seitenlänge des Grids – durch die effiziente Vorverarbeitung durch das Voxel-Grid-Filter jedoch nur linear. So ist die Berechnungszeit des Voxel-Grid-Filters annähernd unabhängig von der Grid-Größe, und gleichzeitig benötigt damit das anschließende Filterupdate nur noch so viel Zeit, wie bereits für die einfache Ausgabe der Grid-Nachricht benötigt wird. Die Gesamtverarbeitungszeit bleibt dadurch selbst bei Grid-Größen von 200 m × 200 m unter 50 ms pro LiDAR-Scan und damit sicher echtzeitfähig.

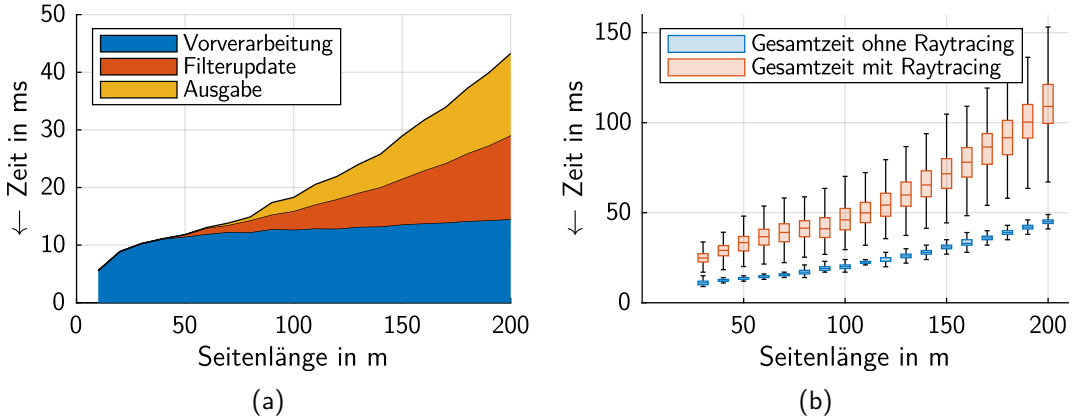


Abbildung 6.9:
Mittlere Laufzeiten des Hindernis-Grids in Abhängigkeit von der Grid-Größe: (a) Laufzeiten der einzelnen Berechnungsschritte ohne Raytracing. (b) Vergleich der Streuung der Gesamtrechenzeit mit und ohne Raytracing. Die Punktwolken wurden durch einen Velodyne HDL-64E S2 auf einem Waldweg mit vielen Hindernissen aufgenommen.

Würde man das (zweidimensionale) Raytracing aktivieren, das im Umgebungsmodell von Jaspers et al. [2017] zum Einsatz kam (vgl. Abschnitt 6.3.3), würde das Hindernis-Grid jedoch mit zunehmender Grid-Größe signifikant verlangsamt, wie Abbildung 6.9b zeigt. Die durch das Raytracing zusätzlich benötigte Zeit ist dabei stark von der Anzahl Hindernispunkte abhängig. Berücksichtigt man alle Ausreißer, so dürfte mit Raytracing die Seitenlänge des Grids maximal 140 m messen, um auch bei extrem vielen Hindernissen Echtzeitfähigkeit garantieren zu können.

Entsprechend der Anforderungen der das Hindernis-Grid nutzenden Module zum Wege- und Kreuzungstracking wird das Grid 100 m × 100 m groß gewählt und um das Ego-Fahrzeug zentriert. Bei dieser Grid-Größe ist das Hindernis-Grid in allen Fällen echtzeitfähig. Ohne Raytracing beträgt die mittlere Laufzeit 18 ms – bei einer Punktwolke eines Velodyne HDL-64E S2. Seit 2022 werden bei TAS auch LiDAR-Sensoren des Typs Velodyne Alpha Prime (VLS-128) eingesetzt. Damit erhöht sich

die Rechenzeit des Hindernis-Grids aber dennoch nur auf 24 ms – obwohl dieser Sensor etwa doppelt so viele Punkte misst (in derselben Szene, bei der beide Sensoren auf dem Fahrzeug montiert waren). Dies ist der effizienten Vorverarbeitung durch das Voxel-Grid-Filter zu verdanken, das die Punktwolke auf nur eine Messung pro 20 cm × 20 cm großer Grid-Zelle reduziert. In der Vorwärtsfahrt könnte die benötigte Rechenzeit sogar auf 15 ms (HDL-64) bzw. 17 ms (VLS-128) gesenkt werden, wenn man dafür nur Hindernispunkte vor dem Fahrzeug betrachten würde.

All diese genannten Laufzeiten beziehen sich auf die Berechnung der Belegtheitswahrscheinlichkeit aus der LiDAR-Punktwolke und die Ausgabe des Hindernis-Grids. Parallel dazu werden komplett unabhängig vom Sensor-Zyklus des LiDAR-Scanners die Befahrbarkeitswahrscheinlichkeiten aus dem semantisch segmentierten Kamerabild bestimmt. Zur Auswertung der dazu benötigten Rechenzeit zeigt Abbildung 6.10 die zum Einfärben der belegten Hindernis-Zellen mit der Befahrbarkeitswahrscheinlichkeit benötigte Zeit in Abhängigkeit von der Anzahl einzufärbender Hindernis-Zellen. Es zeigt sich, dass die Rechenzeit nur sehr schwach von der Anzahl einzufärbender Hindernis-Zellen und damit der abzutastenden Bildpixel abhängt. Der Flaschenhals liegt stattdessen im strahlenförmigen Iterieren durch das Hindernis-Grid im Speicher. Letztlich liegt die zum Einfärben benötigte Rechenzeit im Mittel bei 20 ms.

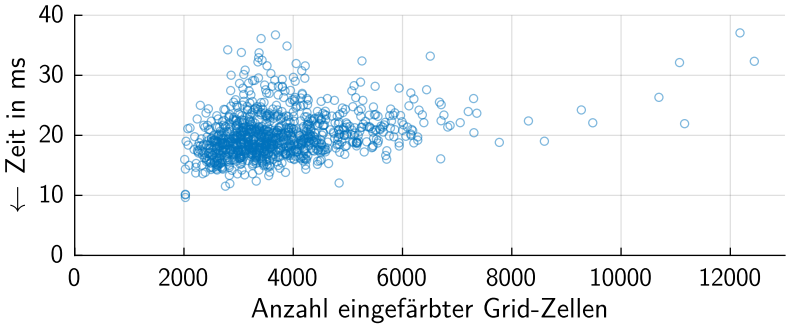


Abbildung 6.10:
Rechenzeiten für das Einfärben des Hindernis-Grids mit dem Befahrbarkeitswahrscheinlichkeitsbild in Abhängigkeit von der Anzahl eingefärbter Grid-Zellen. Wie in Abbildung 6.9 wird für eine pessimistische obere Schranke ein Waldweg mit vielen Hindernissen verwendet.

6.7 Diskussion

Mittels einer segmentierten LiDAR-Punktwolke können in einem 2½D-Grid statische Hindernisse mit ihrer vertikalen Ausdehnung modelliert und in Echtzeit probabilistische Belegtheitswahrscheinlichkeiten geschätzt werden. Dabei ist es wichtig, Messungen von dynamischen Objekten zu ignorieren. Um das Hindernis-Grid auch auf stark bewachsenen Feldwegen einsetzen zu können, wird in dieser Arbeit zudem eine kamerabasierte Oberflächenklassifikation in das Hindernis-Grid fusioniert, um eine zusätzliche Befahrbarkeitswahrscheinlichkeit zu schätzen. Mit der Befahrbarkeitswahrscheinlichkeit lässt sich nun aus der Belegtheitswahrscheinlichkeit eine tatsächliche Hinderniswahrscheinlichkeit dafür bestimmen, dass eine Grid-Zelle für das Fahrzeug tatsächlich unpassierbar ist, und nicht nur hohes Gras gemessen wurde.

Zur Bestimmung der Befahrbarkeitswahrscheinlichkeiten werden die in 20 cm × 20 cm großen Grid-Zellen rasterisierten Hindernisse in ein semantisch segmentiertes Bild projiziert und an einzelnen Pixeln eine aus den Klassenwahrscheinlichkeiten berechnete Befahrbarkeitswahrscheinlichkeit abgetastet. Auf diese Art und Weise werden in den getesteten Beispielen zuverlässig befahrbare Hindernisse klassifiziert, und das Hindernis-Grid wird auch auf ungepflegten Feldwegen einsetzbar gemacht. Die Kombination aus einem gröber rasterisierten Belegtheitsgitter mit einem deutlich feiner rasterisierten Befahrbarkeitswahrscheinlichkeitsbild ist jedoch eine mögliche Schwachstelle des Verfahrens, die zu Ungenauigkeiten führen kann. Abbildung 6.11 zeigt eine Beispielszene eines Feldwegs mit einem Verkehrsschild. Die aus der LiDAR-Punktwolke geschätzten Hindernisse mit einer Belegtheitswahrscheinlichkeit größer 50 % wurden als Quader entsprechend ihrer minimalen und maximalen Höhe in das Kamerabild projiziert. Dabei ist der Quader an der Stelle des Verkehrsschilds mit seiner Kantenlänge von 20 cm deutlich breiter als der Pfosten des Verkehrsschilds. Zudem muss der Pfosten nicht perfekt mittig innerhalb des Quaders liegen. Dadurch könnte das Befahrbarkeitswahrscheinlichkeitsbild fälschlicherweise neben dem Pfosten abgetastet und der Pfosten damit als befahrbar angenommen werden. Die Position des Pfostens genauer zu modellieren ist jedoch gegen die Idee der Rasterisierung der Hindernisse in einem Grid. Für eine Befahrbarkeitsmodellierung im urbanen Umfeld mit hauptsächlich schmalen Hindernissen könnte eine explizitere Modellierung der statischen Umgebung durch einzelne Objekte erfolgen. Im nicht-urbanen Umfeld mit vermehrt Buschwerk und untrennbarer Vegetation ist dies jedoch nicht zielführend.

Inwieweit sich der beschriebene Effekt auf Feldwegen auswirkt, kann nicht evaluiert werden, da die in [Metzger et al., 2021] vorgestellte semantische Segmentierung Verkehrsschilder nicht zuverlässig segmentiert. So verschmilzt in der in Abbildung 6.12 gezeigten Segmentierung von Abbildung 6.11 das Verkehrsschild mit dem Hintergrund. Dies liegt daran, dass die semantische Segmentierung primär für die Oberflächenklassifikation in unstrukturierter Umgebung entwickelt wurde und nur wenige Trainingsdaten von Verkehrsschildern erhalten hat. Um die Ergebnisse der Oberflächenklassifikation auf Straßen und Feldwegen zu verbessern, wird am Institut derzeit ein großer, annotierter Datensatz (German Outdoor and Offroad Dataset – GOOSE [Mortimer et al., 2024]) veröffentlicht, der sowohl unstrukturiertes Gelände als auch Feldwege und Straßen enthält. Damit soll die Oberflächenklassifikation für alle Einsatzgebiete erweitert und auch insgesamt die Robustheit erhöht werden.

Bis dahin wird mangels der Zuverlässigkeit der semantischen Segmentierung auf unbekannten Feldwegen in den folgenden Kapiteln auf den Einsatz der Oberflächenklassifikation verzichtet und ausschließlich die Belegtheitswahrscheinlichkeit herangezogen. Um dennoch die Detektion eigentlich befahrbarer Hindernisse auf Feldwegen mit Grünstreifen zu vermeiden, wird dabei unabhängig vom verwendeten Fahrzeug immer die Bodenfreiheit von MuCAR-3 von $h_{\text{gc}} = 30\,\text{cm}$ als minimale Hindernishöhe verwendet, bei der eine Grid-Zelle als belegt klassifiziert wird. Damit wurde kein hohes Gras mehr als Hindernis kartiert. Ansonsten könnte zur weiteren Robustheitssteigerung gegen hohes Gras auch in den abnehmenden Modulen der Schwellwert dafür, ab welcher Belegtheitswahrscheinlichkeit ein Hindernis erkannt wird, von 50 % auf 70 % erhöht werden.



Abbildung 6.11:
Beispielszene eines Feldwegs mit Verkehrsschild. Die geschätzten Hindernisse mit einer Belegtheitswahrscheinlichkeit im Belegtheitsgitter größer 0,5 sind entsprechend ihrer minimalen und maximalen Höhe als Quader ins Kamerabild projiziert dargestellt. Zur besseren Sichtbarkeit des Schildpfostens wurden hier LiDAR-Punkte oberhalb des Fahrzeugs abgeschnitten. Die untersten Punkte auf dem Pfosten wurden zudem durch die Segmentierung als Bodenpunkte klassifiziert, wodurch das kartierte Hindernis minimal zu weit oben liegt.

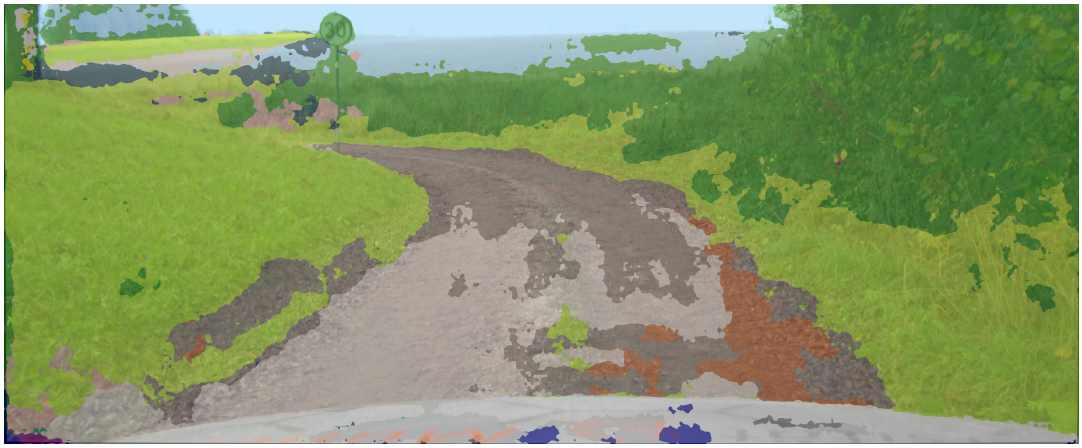


Abbildung 6.12:
Das Ergebnis der semantischen Segmentierung (wahrscheinlichste Klassen) des in Abbildung 6.11 gezeigten Kamerabilds eines Feldwegs mit Verkehrsschild als Überlagerung. Farblegende: sky, asphalt, gravel, soil, grass, forest, ego_vehicle, car, building, fence.

7 Terrain-Schätzung

Ob ein Gebiet durch ein autonomes Fahrzeug befahrbar ist, hängt außer von der eben untersuchten Hindernisfreiheit auch vom Gelände und sog. negativen Hindernissen wie Gräben oder Schlaglöchern ab. Während im Straßenverkehr der Boden meist als Ebene angenommen wird, kann auf Feldwegen nicht davon ausgegangen werden, dass der Boden flach ist. Daher muss ein Modell des Untergrunds erstellt werden, das den Verlauf des Bodens relativ zum Fahrzeug abbildet. Dieses Terrain-Modell ist nicht nur für die Trajektorienplanung und die Fahrzeugregelung interessant, sondern bereits im Straßen- und Wegetracking von Bedeutung: Da die durch das Stereo Matching gewonnenen Entfernungsschätzungen nicht den Genauigkeitsanforderungen entsprechen, muss die Entfernung von im Kamerabild detektierten Wegmerkmalen nun auf andere Art und Weise bestimmt werden. Hier hilft ein präzises Terrain-Modell, aus dem die Höhe und die Steigung des im Terrain liegenden Wegs abgeleitet werden können. Dadurch kann die Lage des Wegs im 3D-Raum bestimmt werden, auch wenn dieser noch nicht von verschiedenen Perspektiven betrachtet werden konnte. Das Wissen über Straßengräben kann außerdem dabei helfen, den Weg von der Umgebung abzugrenzen. Schließlich kann mit einem Modell des Untergrunds eine korrekte Draufsicht auf die Szene erzeugt werden. Diese findet in der Kreuzungserkennung Verwendung. Um ein solches Terrain-Modell anhand der Fahrzeugsensorik aufzubauen, wurde eine Terrain-Schätzung entwickelt, die in [Forkel et al., 2021b] und [Forkel und Wuensche, 2022b] bereits vorab veröffentlicht wurde.

7.1 Einleitung und Problemstellung

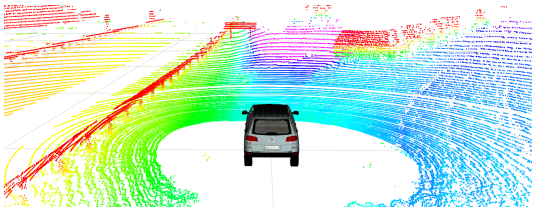
Ausgangslage: Geländekarte Im vorhergehenden Kapitel wurde die Höhe statischer Hindernisse in der Umgebung kartiert, indem in einem Grid Punktmessungen über die Zeit akkumuliert wurden. Der Untergrund wurde bisher am Institut für Technik Autonomer Systeme (TAS) im selben Grid akkumuliert [Jaspers et al., 2017]. Wie in Abbildung 6.2 bereits gezeigt wurde, weisen akkumulierte Bodenkarten jedoch Löcher und Ausreißer auf. Zudem wird für das Straßen- und Wegetracking eine hindernisbereinigte Terrain-Schätzung benötigt: Da der Weg unter überhängenden Bäumen oder hohen Straßenschildern verlaufen kann, muss die Terrain-Schätzung den Boden auch unterhalb von Hindernissen explizit modellieren, damit daraus die Lage der Straße abgeleitet werden kann.

Probabilistische Terrain-Schätzung (auf Straßen) In gemeinsamer Arbeit mit Jan Kallwies wurde daher eine vom Hindernis-Grid unabhängige Terrain-Schätzung entwickelt, bei der die zeitliche Akkumulierung von Bodenpunkten durch eine zusätzliche räumliche Glättung ergänzt wird, um eine probabilistische Schätzung einer glatten und kontinuierlichen Terrain-Oberfläche ohne Löcher und Hindernisse zu erhalten. Während in [Kallwies, 2023] das grundlegende Konzept der Terrain-Schätzung vorgestellt und die Terrain-Messung aus Stereo-Punktwolken behandelt wird, liegt der Beitrag dieser Arbeit in der Verwendung einer LiDAR-Punktwolke zur Messung des Terrains sowie der Weiterentwicklung der Terrain-Schätzung für Feldwege.

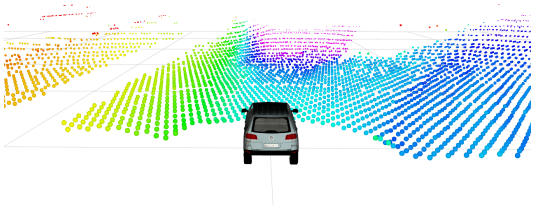
Das Ergebnis der Terrain-Schätzung ist erneut ein Grid, das nun jedoch einem anderen Zweck dient: Anders als die Belegtheit sind Höhe und Steigung räumlich kontinuierliche Zustände, die eng mit ihren Nachbarzellen verbunden sind. Daher werden in der Terrain-Schätzung die Grid-Zellen nicht wie im Hindernis-Grid zur Rasterisierung verwendet. Stattdessen bilden die Mittelpunkte der Grid-Zellen Stützpunkte für ein kontinuierliches Terrain-Modell. Dadurch kann die Auflösung des Terrain-Grids gröber gewählt werden als die des Hindernis-Grids, da zwischen den Stützpunkten entsprechend des Modells interpoliert werden kann. Der Zustand dieser Stützpunkte wird über zwei Schritte rekursiv geschätzt: Zunächst wird aus allen Bodenpunkten, die in die entsprechende Grid-Zelle fallen, eine Messung generiert und diese über die Zeit akkumuliert. Anschließend werden die akkumulierten Messungen mittels Optimierung zu einem kontinuierlichen Terrain-Modell geglättet. Abbildung 7.1 visualisiert diese Schritte an einem Beispiel.



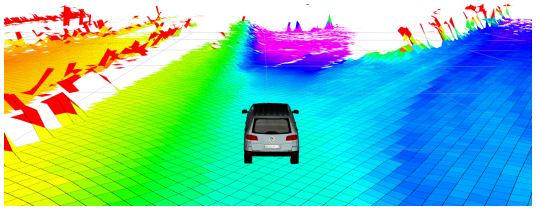
(a) RGB-Kamerabild



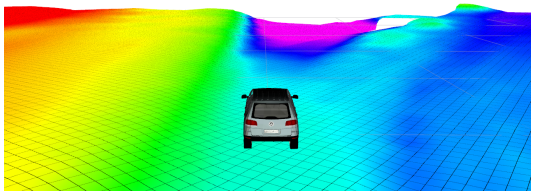
(b) Eingehende LiDAR-Punktwolke



(c) Extrahierte Bodenmessungen

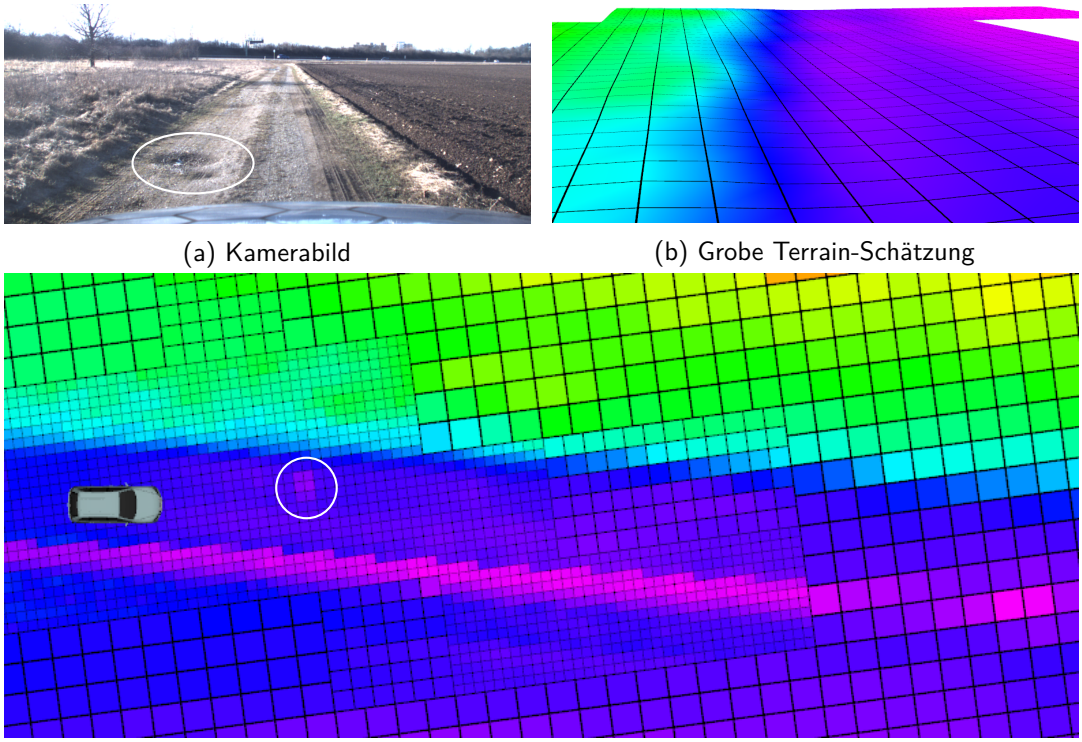


(d) Akkumulierte Höhen



(e) Geglättete Höhen

Abbildung 7.1:
Visualisierung der einzelnen Verarbeitungsschritte der Terrain-Schätzung. (a) Das Kamerabild zeigt, wie das Ego-Fahrzeug auf einem Feldweg neben einer durch eine Leitplanke abgegrenzten Autobahn fährt. (b) Der Ausgangspunkt ist eine LiDAR-Punktwolke. (c) Aus der LiDAR-Punktwolke werden Bodenpunkte extrahiert und je eine Messung für jeden Stützpunkt bestimmt. (d) Diese werden über die Zeit akkumuliert. Der Untergrund des Feldwegs wird verrauscht gemessen. Die Leitplanke verursacht eine Verdeckung, durch die die akkumulierten Höhen Lücken aufweisen. Fälschlicherweise als Bodenpunkt segmentierte Messungen der auf der Autobahn fahrenden Fahrzeuge sowie des Gebüschs führen außerdem zu Ausreißern im akkumulierten Terrain. (e) Die räumliche Glättung glättet Ausreißer und Rauschen und erzeugt eine durchgängige Terrain-Schätzung auch unter Hindernissen. Alle Bilder verwenden dieselbe Farbkodierung von pink für niedrige Höhenwerte über blau, grün, gelb, bis rot für die höchsten geschätzten Höhenwerte.



(a) Kamerabild (b) Grobe Terrain-Schätzung

(c) Terrain-Schätzung mit dynamischer Auflösung

Abbildung 7.2:
Beispiel für die für Feldwege weiterentwickelte Terrain-Schätzung mit dynamischer Auflösung (0,5 m bis 1,5 m) im Vergleich zu einer Terrain-Schätzung mit durchgehend grober Auflösung (1,6 m). Die geschätzten Terrain-Höhen sind einheitlich farbkodiert von pink (niedrig) zu orange (hoch). Abbildung (b) zeigt die Egoperspektive, (c) eine Draufsicht.

Problemstellung auf Feldwegen: Auflösung vs. Rechenzeit Bei der Anzahl der Stützpunkte und damit der Auflösung des Terrain-Grids muss ein Kompromiss zwischen Genauigkeit und Rechenzeit getroffen werden. Für den Einsatz auf Straßen wurde eine relativ geringe Auflösung von 1,6 m Zell-Kantenlänge gewählt. Wie das Beispiel in Abbildung 7.2b zeigt, ist dies auf Feldwegen jedoch zu ungenau. Bei zu geringer Auflösung werden Schlaglöcher und Gräben vollständig weggeglättet. Daher bedarf es für das autonome Fahren auf Feldwegen einer höheren Auflösung. Dies erhöht allerdings den Umfang des zu lösenden Optimierungsproblems und die dafür benötigte Rechenzeit deutlich. So ist es mit einem einzigen Prozessorkern nicht möglich, die für das Wegetracking benötigte Terrain-Fläche von 1500 m^2 in Echtzeit mit der angestrebten Auflösung von 0,5 m Zell-Kantenlänge zu schätzen.

Doch warum sollte das gesamte Terrain in der gleichen Auflösung modelliert werden? Da sich das autonome Fahrzeug auf dem Weg bewegen soll, sind die Bereiche des Terrains auf einem Weg und um einen Weg herum von höherem Interesse als nicht befahrbare Bereiche abseits des Wegs. Außerdem erfordern manche Bereiche, beispielsweise die mit kleinen, negativen Hindernissen, eine höhere Auflösung als andere. Daher sollen in der Terrain-Schätzung auf Feldwegen unterschiedliche Bereiche des Terrains unterschiedlich fein aufgelöst werden. Abbildung 7.2c zeigt dazu ein Beispiel der letztlich entwickelten Terrain-Schätzung mit dynamischer Auflösung.

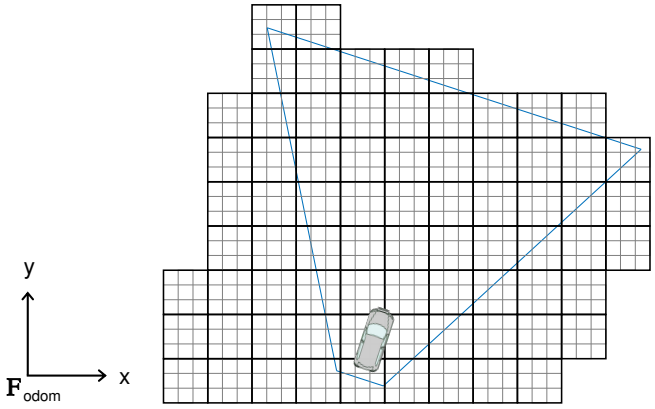


Abbildung 7.3:
Schematisches Beispiel eines in kleinere Kacheln aufgeteilten Terrain-Grids. Die Terrain-Kacheln haben eine feste Größe und eine feste Position im Odometriekoordinatensystem. Bewegt sich das Ego-Fahrzeug weiter, so werden neue Terrain-Kacheln erstellt, wobei immer mindestens für den durch das blaue Trapez visualisierten Bereich Kacheln existieren müssen. Eine Grid-Kachel besteht wiederum aus mehreren Grid-Zellen. Die Anzahl und Größe der Grid-Zellen innerhalb einer Terrain-Kachel soll dynamisch angepasst werden.

Durch die lokale Erhöhung der Auflösung können ein Straßengraben und ein Schlagloch korrekt kartiert werden. Gleichzeitig läuft die Terrain-Schätzung weiterhin in Echtzeit, da nicht das gesamte Terrain in hoher Auflösung geschätzt werden muss.

Problemstellung: Dynamische Auflösungsänderung Für diese lokale Auflösungserhöhung kann genutzt werden, dass das entwickelte Terrain-Grid aus einzelnen Teil-Grids besteht, die Kacheln genannt werden. Die Kacheln sind im Odometriekoordinatensystem stationär und können einzeln erstellt und geglättet werden. Dies diente bisher dazu, auch nur eine Teilmenge der Terrain-Kacheln glätten zu können. So wird auf die erneute Glättung von Kacheln hinter dem Fahrzeug, die bereits gut modelliert wurden, oder die Glättung von Kacheln in weiter Entfernung, die noch nicht relevant sind, aus Gründen der Rechenzeit verzichtet. Abbildung 7.3 zeigt ein schematisches Beispiel. Diese einzelnen Kacheln sollen nun unterschiedliche Auflösungen erhalten, d. h. die Kacheln sollen in unterschiedlich viele und große Grid-Zellen aufgeteilt werden. Dabei soll jede Kachel individuell nach Bedarf zwischen einer minimalen Auflösung von 1,5 m Zell-Kantenlänge und einer maximalen Auflösung von 0,5 m skaliert werden. Dadurch stellt sich das neue Problem, wie aus einzelnen Kacheln unterschiedlicher Auflösung ein kontinuierliches Grid-Modell erzeugt werden kann. Zudem muss ein Konzept entwickelt werden, wie die Auflösung der einzelnen Kacheln zur Laufzeit dynamisch bestimmt und laufend angepasst werden kann.

Gliederung In Abschnitt 7.2 folgt zunächst eine Literaturrecherche über Terrain-Modelle. Abschnitt 7.3 beschreibt anschließend die entwickelte Feldweg-Terrain-Schätzung: Abschnitt 7.3.1 stellt die Extraktion von Terrain-Messungen aus einer LiDAR-Punktwolke vor. Die Abschnitte 7.3.2 und 7.3.3 beschreiben das zeitliche Akkumulieren der Messungen und die räumliche Glättung der Terrain-Höhen. Abschnitt 7.3.4 zeigt anschließend, wie die Auflösung einer Terrain-Kachel bestimmt

wird. Abschnitt 7.3.5 befasst sich dazu mit der Schätzung einer Straßenwahrscheinlichkeit sowie dem Einfärben des Terrain-Grids mit dem Kamerabild. Schließlich zeigt Abschnitt 7.4 qualitative Ergebnisse sowie Laufzeitanalysen.

7.2 Stand der Forschung

LiDAR-basierte Terrain-Grids Zwei Verfahren zur Modellierung der Bodenoberfläche wurden bereits in Abschnitt 6.2 im Rahmen der Kartierung statischer Hindernisse vorgestellt: Douillard et al. [2010] und Jaspers et al. [2017] rastern die Bodenoberfläche in Grid-Zellen und mitteln die Höhe in diesen Zellen gemessener LiDAR-Punkte über die Zeit. Diese diskrete, Grid-basierte Repräsentation des Terrains anhand von LiDAR-Messungen geht zurück auf Bares et al. [1989] und Kweon und Kanade [1992]. Ye und Borenstein [2003] stellen dazu später Filtermethoden zur Eliminierung von Messfehlern vor. Triebel et al. [2006] schlagen weiter vor, nicht nur den Mittelwert, sondern auch die Varianz der Messungen einer Grid-Zelle zu berechnen. Aufbauend auf Ye und Borenstein [2003] berechnen Belter und Skrzypczyński [2011] schließlich unter Berücksichtigung von Messunsicherheiten Unsicherheiten zu den geschätzten Terrain-Höhen.

Kontinuierliche Terrain-Modelle Diese diskreten Terrain-Grids wurden primär für kleinere Schreitroboter entwickelt. Für größere Fahrzeuge, die sich schneller bewegen, werden stattdessen glatte, kontinuierliche Terrain-Modelle gesucht, die größere Bereiche repräsentieren können. Broggi et al. [2013] und Jaspers und Wuensche [2014] berechnen dazu mittels Basis-Splines Steigungen für die einzelnen Grid-Zellen, wodurch ein kontinuierliches Terrain-Modell entsteht. Eine andere verbreitete Methode ist die Modellierung des Terrains durch Gauß-Prozesse [Lang et al., 2007, Plagemann et al., 2009, Vasudevan et al., 2009, Hadsell et al., 2010, Yang et al., 2016]. Diese können zwar statistisch korrekt mit Unsicherheiten umgehen und Lücken durch unvollständige Sensordaten füllen, sind jedoch sehr rechenaufwendig. Für Echtzeitanwendungen können entweder nur kleinere Bereiche des Terrains modelliert werden, oder es müssen Näherungen getroffen werden. Zudem müssen Steigungen dabei weiterhin nachträglich berechnet werden.

Dynamische Auflösungsanpassung Ein Nachteil, den alle klassischen Grid-basierten Terrain-Modelle teilen, ist die feste Diskretisierung des gesamten Raums in derselben Auflösung. Um verschiedenen Sensorauflösungen und -reichweiten gerecht zu werden, verwenden Montemerlo und Thrun [2004] und Belter et al. [2012] daher mehrere überlagerte Terrain-Grids unterschiedlicher Größe und Auflösung. Im Gegensatz zur in dieser Arbeit vorgestellten dynamischen Auflösungsänderung müssen dabei jedoch mehrere Terrain-Grids aktualisiert werden, und die Auflösung der einzelnen Grids kann nicht dynamisch angepasst werden.

Die Idee, für einzelne Kacheln eines diskreten Terrain-Modells unterschiedliche Auflösungen zu verwenden, stammt aus dem Bereich der Computergrafik. In der Geländesimulation für Bodenfahrzeuge stellt sich ebenfalls die Herausforderung, das

Terrain mit möglichst geringer Rechenkomplexität möglichst genau abzubilden. Die Lösung von He [2000] und Aquilio [2006] ist es, das Terrain in Kacheln aufzuteilen und diese unterschiedlich fein aufzulösen. Dabei werden die quadratischen Grid-Zellen in einer Kachel entweder in zwei, in acht oder in zweiunddreißig rechtwinklige Dreiecke unterteilt. Das Konzept unterschiedlich fein aufgelöster Terrain-Kacheln wird in dieser Arbeit auf die Terrain-Schätzung angewandt.

7.3 Terrain-Schätzung mit dynamischer Auflösungsanpassung

Das Ziel ist eine adaptive Auflösung der Terrain-Schätzung: Zum einen sollen einzelne Terrain-Kacheln feiner aufgelöst werden können als andere. Zum anderen soll die Auflösung einer Terrain-Kachel aber auch nicht fest vorbestimmt sein, sondern bei Bedarf erhöht werden können, falls das gemessene Terrain mit der aktuellen Auflösung nicht ausreichend genau modelliert werden kann. Die Größe und die Anzahl der Grid-Zellen müssen also zur Laufzeit dynamisch angepasst werden.

Die Grid-Zellen bilden Stützpunkte für ein kontinuierliches Terrain-Modell. Für diese Stützpunkte werden zunächst Messungen aus der LiDAR-Punktwolke extrahiert. Dafür kommt wie bereits im Hindernis-Grid die Punktwolken-Segmentierung von Burger und Wuensche [2018] zum Einsatz (vgl. Abschnitt 6.3.2). Anschließend besteht die Terrain-Schätzung aus zwei Schritten: der Akkumulierung der Messungen über die Zeit, und der Optimierung der Stützpunkte des Terrain-Modells anhand der akkumulierten Messungen. Diese durch Kallwies [2023] konzipierten Schritte müssen für die dynamische Auflösungsanpassung modifiziert werden: Damit die Auflösung des Grids und damit die Anzahl der Stützpunkte zur Laufzeit erhöht werden kann, müssen die Messungen für diese bereits in der höchstmöglichen Stützpunkt-Auflösung vorliegen. Daher werden für jede Terrain-Kachel zwei Grids benötigt:

1. ein Mess-Grid, in dem die Messungen in der höchstmöglichen Auflösung (0,5 m) akkumuliert werden, und
2. ein Glättungs-Grid zur Optimierung, das von der minimalen Auflösung (1,5 m) bis hin zur Auflösung des Mess-Grids (0,5 m) skaliert werden kann.

Die Optimierung muss für die dynamische Auflösung daher mit wechselnden Grid-Auflösungen arbeiten können: Die Auflösung kann sowohl von einer Terrain-Kachel zur nächsten als auch vom Glättungs-Grid zum Mess-Grid variieren. Auf Feldwegen ergibt sich für die Terrain-Schätzung zudem die neue Herausforderung, beim Akkumulieren der Messungen feine Strukturen wie Schlaglöcher nicht herauszufiltern. Dieses Kapitel beschreibt daher eine weiterentwickelte Feldweg-Terrain-Schätzung.

Ein Ablaufplan des Gesamtsystems mit den beiden Grids ist in Abbildung 7.4 zu sehen. Für die dynamische Auflösungsanpassung werden zusätzlich zu den Informationen aus der LiDAR-Punktwolke die Ergebnisse der semantischen Segmentierung des Kamerabilds von Metzger et al. [2021] genutzt, um potenziell besonders relevante Bereiche des Terrains zu ermitteln. In diesen Bereichen wird dann die Abweichung des Glättungs-Grids zum Mess-Grid betrachtet, um die Auflösung einer Terrain-Kachel bedarfsgerecht zu wählen (vgl. Abschnitt 7.3.4).

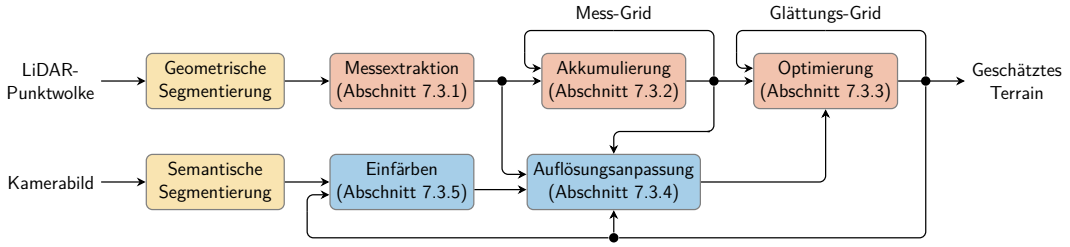


Abbildung 7.4:
Ablaufplan der entwickelten Feldweg-Terrain-Schätzung mit dynamischer Auflösung durch die Fusion segmentierter LiDAR-Punktwolken und Kamerabilder. Bei den gelben Modulen handelt es sich um externe Module zur Vorverarbeitung der Sensordaten. Die roten Module beinhalten die eigentliche LiDAR-basierte Terrain-Schätzung. Die blauen Module werden für die kameraunterstützte dynamische Wahl der Auflösung benötigt.

7.3.1 Extraktion von Messungen aus der LiDAR-Punktwolke

Zunächst werden aus der von Burger und Wuensche [2018] segmentierten LiDAR-Punktwolke Messungen für die Mess-Grid-Zellen extrahiert. Aus allen Punkten, die zum Zeitschritt k in eine Grid-Zelle $\mathsf{c_m}$ des Mess-Grids fallen und als Freiraum- oder Bodenflächenpunkt klassifiziert wurden, soll eine Messung $\mathbf{y}_k^{(\mathsf{c_m})} = [\tilde{h}_k^{(\mathsf{c_m})}]$ der Höhe des Stützpunkts im Mittelpunkt der Grid-Zelle $\mathsf{c_m}$ ermittelt werden. Eine Möglichkeit wäre es, aus allen Punktmessungen in einer Grid-Zelle eine Ebene zu berechnen und daraus die Höhe des Mittelpunkts zu bestimmen. Dies ist jedoch nur sinnvoll, wenn die Bodenpunkte keine falsch segmentierten Ausreißer enthalten, und zudem das Geländemodell stark geglättet sein soll. Kleine negative Hindernisse wie Schlaglöcher würden durch die Ebenenannahme vernachlässigt werden. Um negative Hindernisse hingegen zu betonen, wird zur Feldweg-Terrain-Schätzung stattdessen die niedrigste z-Koordinate aller in Zeitschritt k in die Mess-Grid-Zelle $\mathsf{c_m}$ fallenden Punkte $\mathcal{P}_k^{(\mathsf{c_m})}$ als Messung verwendet:

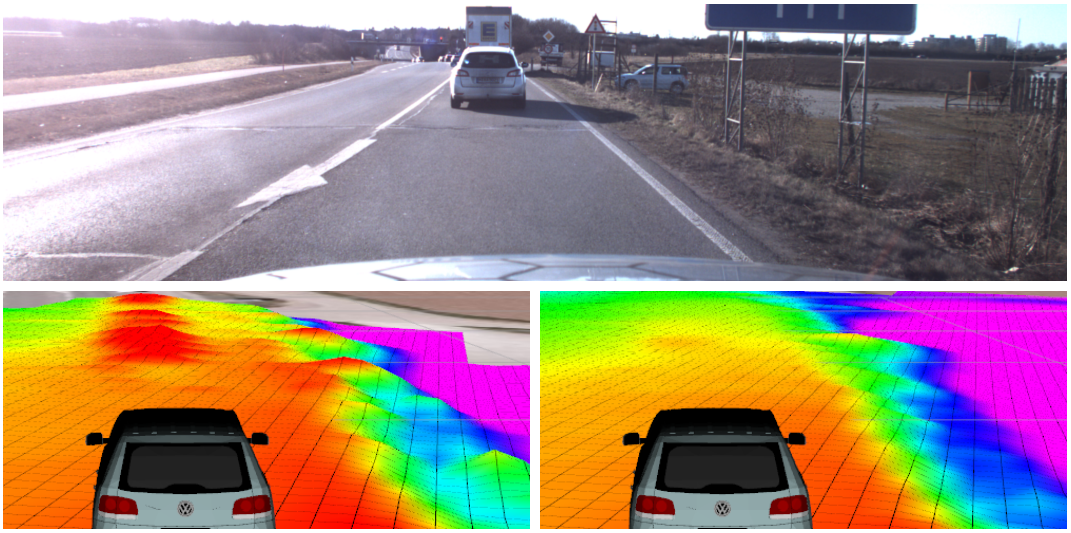
$$\tilde{h}_k^{(\mathsf{c_m})} = \min_{\mathbf{p} \in \mathcal{P}_k^{(\mathsf{c_m})}} (z_\mathbf{p}) \,. \tag{7.1}$$

Jede Messung besitzt eine zugehörige Information $\mathbf{R}_k^{(\mathsf{c_m})^{-1}}$, die die inverse Kovarianz der Messfehler angibt. Die Messfehler beinhalten dabei hier nicht nur die Messunsicherheit des LiDAR-Sensors, sondern auch Messfehler durch die falsche Klassifikation von Bodenpunkten und Messfehler durch die Wahl des Minimums. Daher wird hier eine erwartete Standardabweichung von $\sigma = 0{,}1\,\mathrm{m}$ angenommen. Da mit steigender Anzahl von Punktmessungen in einer Grid-Zelle die Wahrscheinlichkeit dafür steigt, dass das Minimum dieser Punktmessungen tatsächlich der minimalen Bodenhöhe in dieser Grid-Zelle entspricht (und nicht von einem Punkt neben einem potenziellen Schlagloch oder gar einem Ausreißer stammt), wird die Messinformation mit der Anzahl der in einer Zelle gemessenen Bodenpunkte n gewichtet. Zusätzlich wird die Messinformation mit der Abweichung der gemessenen Höhe $\tilde{h}_k^{(\mathsf{c_m})}$ von der akkumulierten Höhe $h_k^{(\mathsf{c_m})}$ (vgl. Abschnitt 7.3.2) skaliert:

$$\mathbf{R}_k^{(\mathsf{c_m})^{-1}} = \max\!\left(\frac{n}{\sigma^2}\left(\sigma - \left|\tilde{h}_k^{(\mathsf{c_m})} - h_k^{(\mathsf{c_m})}\right|\right), 1\right) , \tag{7.2}$$

wobei eine minimale Messinformation von 1 angenommen wird. Dies modelliert, dass die extrahierte Messung im schlimmsten Fall höchstens von einem Ausreißerpunkt zwei bis drei Meter über dem Boden stammen kann, und die Messung damit maximal eine Standardabweichung von 1 m haben kann.

Die (zeitlich akkumulierte) Messinformation bestimmt später bei der räumlichen Glättung des Terrains über die Gewichtung einer (akkumulierten) Messung im Vergleich zu den Messungen benachbarter Grid-Zellen und dem Terrain-Modell. Dies ist der Grund für die Skalierung der Messinformation mit der Abweichung der Messung von der akkumulierten Messung. Abbildung 7.5 zeigt eine Beispielszene, in der vor dem Ego-Fahrzeug andere Fahrzeuge fahren und den Boden für den LiDAR-Sensor verdecken. Dies führt zu einigen falsch klassifizierten Mess-Ausreißern auf den Fahrzeugen. Wird eine konstante Messinformation verwendet, so verursachen diese Ausreißer selbst nach der räumlichen Glättung falsche Erhebungen in der Geländeschätzung. Mit der skalierten Information erhalten die Ausreißer jedoch eine geringere Messinformation. Werden dann bei der räumlichen Glättung die akkumulierten Messungen mit der Summe ihrer Messinformationen gewichtet, so erhalten die entsprechenden Grid-Zellen ein geringeres Gewicht als benachbarte Grid-Zellen mit gut beobachteten Bodenpunkten. Auf diese Weise werden die Ausreißer weggeglättet. Die geringe Gewichtung betrifft dabei zum einen Zellen mit einer hohen Standardabweichung in ihren Messungen, zum anderen Zellen, für die zum ersten Mal eine Messung extrahiert wurde, wodurch keine Aussage getroffen werden kann, ob es sich um einen Ausreißer handelt.



(a) Konstante Information (b) Skalierte Information

Abbildung 7.5:
Vergleich des resultierenden geglätteten Terrains mit (a) einer konstanten Messinformation von $\mathbf{R}_k^{(c_m)^{-1}} = 1/\sigma^2 = 100$ und (b) der nach Gleichung (7.2) skalierten Information in einer Beispielszene. Mess- und Glättungs-Grid haben eine Auflösung von 0,5 m. Beide Visualisierungen nutzen zur Farbkodierung denselben Wertebereich, wobei pink niedrige Höhen und rot hohe Höhen anzeigt. Ohne die Skalierung führen durch ein vorausfahrendes Fahrzeug verursachte Falschmessungen zu falschen Hügeln (rot) im geschätzten Terrain.

7.3.2 Akkumulieren der Messungen in hochaufgelösten Mess-Grids

Der zweite Schritt der Terrain-Schätzung ist das Akkumulieren der im ersten Schritt gewonnenen Messungen einer Mess-Grid-Zelle und ihrer Information über die Zeit. Eine Möglichkeit wäre es, zu diesem Zweck ein Informationsfilter zu verwenden. Dieses bildet das gewichtete Mittel der Messungen, wodurch jedoch negative Hindernisse abgeschwächt werden, falls mehr Punkte neben einem Schlagloch oder einem Graben gemessen werden als innerhalb. Dies ist bedingt durch den Einfallswinkel eines LiDAR-Strahls in negative Hindernisse häufig der Fall. Deshalb wird für die Feldweg-Terrain-Schätzung für jede Grid-Zelle $\mathsf{c_m}$ des Mess-Grids stattdessen die niedrigste je gemessene Höhe als akkumulierte Höhe $h_k^{(\mathsf{c_m})}$ verwendet:

$$h_k^{(\mathsf{c_m})} = \min\left(\tilde{h}_k^{(\mathsf{c_m})},\, h_{k-1}^{(\mathsf{c_m})}\right) . \tag{7.3}$$

Mögliches Messrauschen wird dabei im anschließenden räumlichen Glättungsschritt eliminiert. Abgesehen von nasser Asphaltfläche, die für den LiDAR-Strahl als Spiegel wirkt, können keine Ausreißer in Form von zu niedrigen Punkten gemessen werden. Auf Feldwegen beschränkt sich daher das Messrauschen bei der Minimum-Akkumulierung auf Falschmessungen in Grid-Zellen, für die ausschließlich falsch segmentierte höhere Hindernispunkte als Messung extrahiert wurden, sowie das Messrauschen des LiDAR-Sensors. Letzteres ist mit etwa 2 cm (vgl. [Velodyne LiDAR, Inc., 2011] bzw. [Velodyne LiDAR, Inc., 2018]) sehr gering. Falschmessungen in Grid-Zellen, für die noch nie ein korrekter Bodenpunkt extrahiert wurde, könnte auch ein Filter nicht allein über die Zeit korrigieren. Hierfür wird zwingend eine räumliche Glättung benötigt.

Die der akkumulierten Messung zugehörige akkumulierte Information

$$\mathbf{P}_k^{(\mathsf{c_m})^{-1}} = \min\left(\mathbf{P}_{k-1}^{(\mathsf{c_m})^{-1}} + \mathbf{R}_k^{(\mathsf{c_m})^{-1}},\, \frac{50}{\sigma^2}\right) \tag{7.4}$$

bildet wie bei einem Informationsfilter die Summe der einzelnen Messinformationen. Da unabhängig von der Anzahl der Punkte eine bestimmte Genauigkeit der akkumulierten Messung nie überschritten werden kann, wird die akkumulierte Information auf einen Maximalwert von $50\sigma^{-2} = 5000$ begrenzt. Dies entspricht bei $\sigma = 0{,}1\,\mathrm{m}$ einer minimalen Standardabweichung der akkumulierten Höhe von etwa 1,4 cm.

7.3.3 Glättung von Terrain-Kacheln unterschiedlicher Auflösung

Die im Mess-Grid akkumulierten Höhen dienen als Messwerte für ein kontinuierliches Geländemodell. Dieses wird durch Stützpunkte beschrieben, deren Abstand im Rahmen der dynamischen Auflösungsanpassung variiert werden kann. Hierzu dient ein separates Glättungs-Grid, dessen Zell-Mittelpunkte die Stützpunkte bilden. Für jeden dieser Stützpunkte wird ein Zustandsvektor

$$\mathbf{x}_k^{(\mathsf{c})} = \left[h^{(\mathsf{c})}\ m_{\mathsf{x}}^{(\mathsf{c})}\ m_{\mathsf{y}}^{(\mathsf{c})}\right]^{\mathsf{T}} \tag{7.5}$$

geschätzt, der neben der Höhe $h^{(\mathsf{c})}$ eines Stützpunkts c auch Steigungen $m_{\mathsf{x}}^{(\mathsf{c})}$ und $m_{\mathsf{y}}^{(\mathsf{c})}$ zu den nächsten Stützpunkten in x- und y-Richtung des Odometriekoordinatensystems enthält. Basierend auf den Steigungen kann mit der Funktion

$$h_{\mathsf{model}}^{(\mathsf{c}\to\mathsf{c}^*)}(\mathbf{x}_k) = h^{(\mathsf{c})} + \begin{bmatrix} \mathrm{d}_{\mathsf{x}}(\mathsf{c},\mathsf{c}^*) \\ \mathrm{d}_{\mathsf{y}}(\mathsf{c},\mathsf{c}^*) \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} m_{\mathsf{x}}^{(\mathsf{c})} \\ m_{\mathsf{y}}^{(\mathsf{c})} \end{bmatrix} \tag{7.6}$$

die Höhe benachbarter Stützpunkte c^* prädiziert werden, wobei $\mathrm{d}_{\mathsf{x}}(\mathsf{c},\mathsf{c}^*)$ und $\mathrm{d}_{\mathsf{y}}(\mathsf{c},\mathsf{c}^*)$ die jeweilige Distanz vom aktuellen Stützpunkt in x- und y-Richtung angeben. Damit wird ein einfaches Geländemodell erster Ordnung modelliert, es könnten jedoch auf dieselbe Art und Weise auch Modelle höherer Ordnung geschätzt werden.

Die Schätzung des Zustandsvektors $\mathbf{x}_k$ aller Stützpunkte einer Terrain-Kachel erfolgt durch die Lösung des MAP-Problems (Maximum-a-posteriori)

$$\hat{\mathbf{x}}_k = \arg\min_{\mathbf{x}_k} \left\{ \sum_{\mathsf{c}} \sum_{\mathsf{c_m}} \mathbf{r}_{\mathsf{meas.}}^{(\mathsf{c_m}\to\mathsf{c})}(\mathbf{x}_k)^2 + \sum_{(\mathsf{c},\mathsf{c}^+)} \mathbf{r}_{\mathsf{consist.}}^{(\mathsf{c}\to\mathsf{c}^+)}(\mathbf{x}_k)^2 + \sum_{\mathsf{c}} \mathbf{r}_{\mathsf{prior}}^{(\mathsf{c})}(\mathbf{x}_k)^2 \right\}. \tag{7.7}$$

Dieses minimiert Residuen $\mathbf{r}_{\mathsf{meas.}}$, die die Abweichungen des Modells zu den Messungen bestimmen, Residuen $\mathbf{r}_{\mathsf{consist.}}$, die die Abweichung der Höhe benachbarter Stützstellen zu der aus den geschätzten Steigungen des aktuellen Stützpunkts prädizierten Höhe beschreiben, sowie Residuen $\mathbf{r}_{\mathsf{prior}}$, die die A-Priori-Verteilung der Steigungen modellieren. Im Folgenden werden diese Residuen im Detail erklärt.

Mess-Residuum Gestützt wird das gesamte Terrain-Modell durch die im vorherigen Schritt im Mess-Grid akkumulierten Höhen. Durch die dynamische Auflösung der Grid-Kacheln können das Mess- und das Glättungs-Grid unterschiedliche Auflösungen haben. Damit müssen die Stützpunkte des Mess-Grids und des Glättungs-Grids nicht übereinstimmen. Ist das Glättungs-Grid niedriger aufgelöst als das Mess-Grid, so werden einer Glättungs-Grid-Zelle c dementsprechend nach dem Schema in Abbildung 7.6 nicht nur eine Messung, sondern mehrere Mess-Grid-Zellen $\mathsf{c_m}$ zugeordnet. Für jeden Glättungs-Stützpunkt c und alle zugeordneten (und mit einer Messung gefüllten) Mess-Grid-Zellen $\mathsf{c_m}$ wird mittels Gleichung (7.6) aus der geschätzten Höhe und den geschätzten Steigungen des Glättungs-Stützpunkts die Höhe für die Position des Mess-Stützpunkts prädiziert. Im Residuum wird dann die Abweichung der prädizierten Höhe von der akkumulierten Höhe $h_k^{(\mathsf{c_m})}$ bestimmt:

$$\mathbf{r}_{\mathsf{meas.}}^{(\mathsf{c_m}\to\mathsf{c})}(\mathbf{x}_k)^2 = \left[h_k^{(\mathsf{c_m})} - h_{\mathsf{model}}^{(\mathsf{c}\to\mathsf{c_m})}(\mathbf{x}_k) \right]^2 \cdot \mathbf{P}_k^{(\mathsf{c_m})^{-1}}. \tag{7.8}$$

Dabei dient die akkumulierte Information $\mathbf{P}_k^{(\mathsf{c_m})^{-1}}$ als Gewichtung der Messung. Während bei übereinstimmenden Mess- und Glättungs-Stützpunkten zur Prädiktion lediglich die geschätzte Höhe benötigt wird, basiert die Prädiktion bei abweichender Auflösung auf dem gesamten Zustandsvektor. Damit erlaubt das Mess-Residuum direkte Rückschlüsse auf die Steigungen eines Stützpunkts, ohne dass benachbarte Stützpunkte betrachtet werden müssen. Dadurch ist das Optimierungsproblem bei einer von der Mess-Auflösung abweichenden Glättungs-Auflösung besser bestimmt.

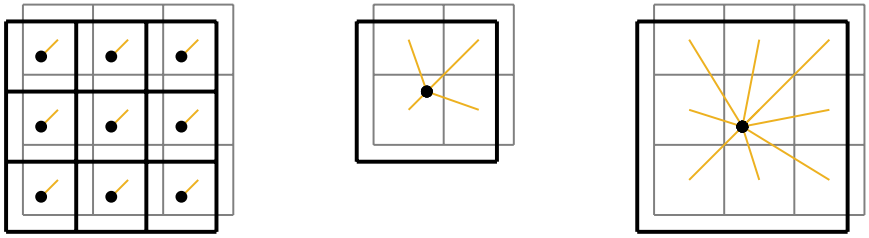


Abbildung 7.6:
Visualisierung der Zuordnung (gelb) von Mess-Grid-Zellen (grau) zu Glättungs-Stützpunkten (schwarz) bei verschiedenen Auflösungen des Glättungs-Grids. Während die Grid-Zellen des Mess-Grids eine feste Kantenlänge von 0,5 m besitzen, hat nur das linke Glättungs-Grid ebenfalls 0,5 m Auflösung, sodass jedem Glättungs-Stützpunkt genau eine Mess-Grid-Zelle zugeordnet werden kann. In der Mitte hat das Glättungs-Grid eine Auflösung von 1,0 m, rechts eine Auflösung von 1,5 m. Dadurch werden dem Glättungs-Stützpunkt entsprechend mehrere Mess-Grid-Zellen zugeordnet.

Konsistenz-Residuum Das Konsistenz-Residuum sorgt für die räumliche Glättung von Messrauschen und Ausreißern und bewirkt zudem eine Interpolation von Terrain-Höhen für Stützpunkte ohne Messungen. Um starke Terrain-Sprünge von einem Stützpunkt zum nächsten zu minimieren und ein kontinuierliches Modell zu erhalten, werden die Stützpunkte über das Terrain-Modell mit benachbarten Stützpunkten c^+ verbunden und ihre Konsistenz zum geschätzten Modell optimiert. Dazu wird das Residuum

$$\mathbf{r}_{\text{consist.}}^{(\mathrm{c}\to\mathrm{c}^+)}(\mathbf{x}_k) = \left[h_{\text{model}}^{(\mathrm{c}\to\mathrm{c}^+)}(\mathbf{x}_k) - h^{(\mathrm{c}^+)}\right] \cdot \mathbf{w}_{[\text{consist.}]} \tag{7.9}$$

minimiert. Es beschreibt die Abweichung der anhand der geschätzten Höhe und Steigungen des aktuellen Stützpunkts prädizierten Höhe eines benachbarten Stützpunkts zu seiner geschätzten Höhe. Dafür werden die Grid-Zellen einer Grid-Kachel zeilen- und spaltenweise verbunden und für jeden Stützpunkt ein Konsistenz-Residuum zum nächsten Stützpunkt in positive x-Richtung und zum nächsten Stützpunkt in negative y-Richtung ausgewertet. Der Gewichtungsfaktor $\mathbf{w}_{[\text{consist.}]}$ bestimmt durch die Skalierung der Abweichung den gewünschten Grad der durch die Minimierung des Residuums erreichten Glättung. Für die Anwendung auf Feldwegen wird der Gewichtungsfaktor entsprechend der in Abschnitt 7.3.1 gewählten Standardabweichung der Messungen von $\sigma = 0{,}1\,\mathrm{m}$ auf $\mathbf{w}_{[\text{consist.}]} = \sqrt{\sigma^{-2}} = 10$ gesetzt.

Der Zustandsvektor $\mathbf{x}_k$, der durch die Lösung des in Gleichung (7.7) beschriebenen MAP-Problems geschätzt wird, enthält alle Stützpunkte einer Terrain-Kachel. Sollen mehrere Terrain-Kacheln geglättet werden, werden mehrere, voneinander unabhängige Optimierungsprobleme gelöst. Da das geschätzte Terrain-Modell jedoch auch über die Kachel-Grenzen hinaus konsistent sein soll, müssen im Konsistenz-Residuum Verbindungen zur Nachbar-Kachel hinzugefügt werden. Dementsprechend werden für die Rand-Stützpunkte, die keine internen Nachbarn in der entsprechenden Richtung mehr haben, stattdessen Konsistenz-Residuen zu den äußersten Zellen der angrenzenden Kacheln berücksichtigt. Dazu kommt das in Abbildung 7.7 gezeigte Verbindungsschema zum Einsatz. Die geschätzten Höhen von Stützpunkten aus Nachbar-Kacheln werden dabei jedoch konstant gehalten und nicht durch das „fremde“ Optimierungsproblem verändert.

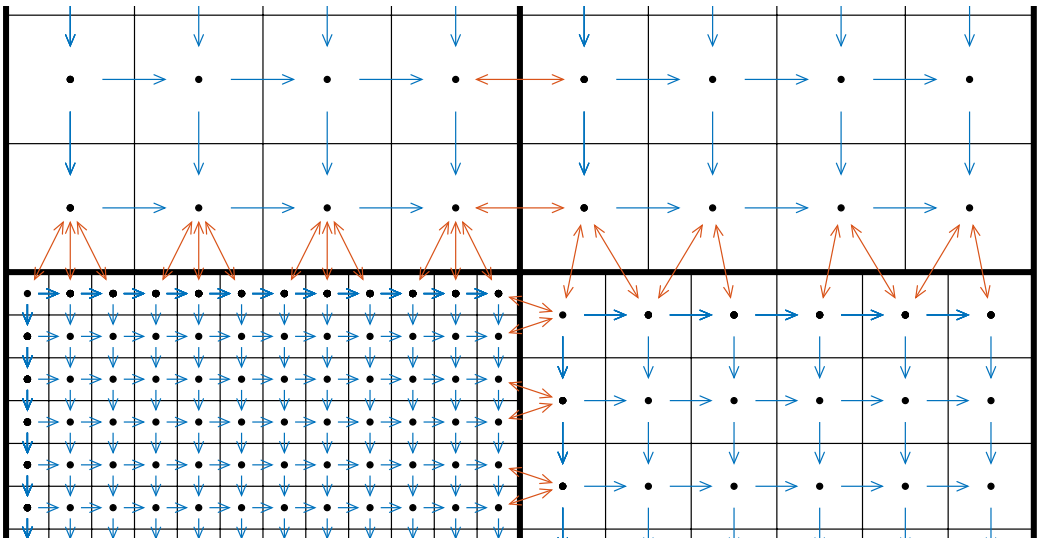


Abbildung 7.7:
Ausschnitt aus dem Verbindungsschema zwischen Glättungs-Stützpunkten. Jeder Pfeil entspricht einem zur Glättung ausgewerteten Konsistenz-Residuum. Die in schwarz dargestellten Glättungs-Grid-Kacheln sind unterschiedlicher Auflösung und haben unterschiedlich viele Grid-Zellen. Die blauen Linien zwischen den Stützpunkten zeigen Verbindungen innerhalb einer Grid-Kachel, die roten Linien zeigen Verbindungen zwischen Grid-Kacheln. Bei benachbarten Kacheln unterschiedlicher Auflösungen werden für die äußeren Stützpunkte entsprechend des Auflösungsunterschieds mehr als zwei Konsistenz-Residuen ausgewertet.

A-Priori-Residuum Mit dem dritten Residuum wird schließlich die A-Priori-Verteilung der Steigungen an den Stützpunkten modelliert. Unter der Annahme, dass die Steigungen normalverteilt mit einem erwarteten Mittelwert von 0 sind, bezieht das Residuum

$$\mathbf{r}^{(c)}_{\text{prior}}(\mathbf{x}_k) = \begin{bmatrix} m_x^{(c)} \\ m_y^{(c)} \end{bmatrix} \cdot \mathbf{w}_{[\text{prior}]} \tag{7.10}$$

die erwartete Standardabweichung $1/\mathbf{w}_{[\text{prior}]}$ der Steigungen in das MAP-Problem ein. Für die Terrain-Schätzung auf Feldwegen wird $\mathbf{w}_{[\text{prior}]} = 10$ gesetzt. Dies entspricht einer erwarteten Standardabweichung der Steigung von 10 % (10 cm auf 1 m Entfernung).

Lösen des Optimierungsproblems Das durch diese drei Residuengruppen aufgestellte lineare Optimierungsproblem wird schließlich mittels des in der Bibliothek *Ceres Solver* [Agarwal et al., 2020] implementierten Levenberg-Marquardt-Algorithmus [Levenberg, 1944, Marquardt, 1963] gelöst. Aufgrund der hohen Reichweite des LiDAR-Sensors können für sehr viele Terrain-Kacheln Höhenmessungen extrahiert werden. Es können jedoch rechenzeitbedingt nicht alle Kacheln optimiert werden. Daher muss eine Auswahl getroffen werden, welche Terrain-Kacheln geglättet werden sollen. Für die Terrain-Schätzung auf Feldwegen, die oft starke Kurven beinhalten, wird ein Trapez von Kacheln vor dem Ego-Fahrzeug zur Optimierung ausgewählt. Für Kacheln außerhalb des Trapezes werden zwar neue Messungen akkumuliert, diese werden jedoch nicht geglättet. Verlässt eine bereits optimierte Kachel das Trapez,

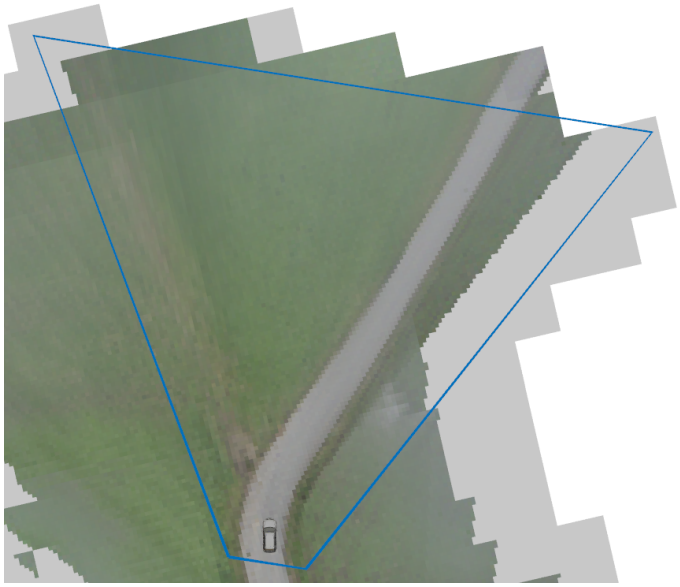


Abbildung 7.8:
Um den aktuellen Lenkwinkel rotiertes Trapez zur Bestimmung der zu optimierenden Terrain-Kacheln. Das dargestellte Terrain-Grid wurde mit einem Kamerabild eingefärbt (s. Abschnitt 7.3.5). Die grauen Bereiche wurden noch nicht von der Kamera aufgenommen.

wird die letzte Schätzung beibehalten. Abbildung 7.8 zeigt dieses Trapez, das dem Blickfeld einer Kamera ähnelt. Es reicht von 2 m hinter dem Ego-Fahrzeug bis 60 m vor das Fahrzeug, mit einer Breite von 10 m bzw. 80 m. Für die bessere Einsicht in Kurven wird das Trapez entsprechend des Gierwinkels der verwendeten drehbaren Kameraplattform MarVEye (Multifokales aktives / reaktives Fahrzeug-Auge, engl. *Multifocal active / reactive Vehicle Eye*) [Unterholzner, 2016] rotiert. Durch die gleiche Rotation der Kamera und des Trapezes wird stets eine aktuelle 3D-Information zum aktuellen Kamerabild bereitgestellt, und das Grid kann gleichzeitig gut mit dem Kamerabild eingefärbt werden. Das ist beispielsweise für die Erkennung von Kreuzungen interessant. In der Anwendung dieser Arbeit folgt das MarVEye dem Lenkwinkel. Auch ohne bewegte Kamera macht es Sinn, den zu optimierenden Bereich mit dem Lenkwinkel zu rotieren, um auch in engen Kurven eine gute Vorausschau zu erzielen. Damit die Terrain-Kacheln das Trapez gut abbilden können, und auch um die Auflösung nur für kleine lokale Bereiche erhöhen zu können, werden die einzelnen Terrain-Kacheln für die Feldweg-Terrain-Schätzung mit dynamischer Auflösung 6 m $\times$ 6 m groß gewählt. Eine Terrain-Kachel besteht also entweder aus 4 $\times$ 4 Grid-Zellen mit einer Kantenlänge von 1,5 m, oder aus 6 $\times$ 6 Grid-Zellen mit 1 m Kantenlänge oder aus 12 $\times$ 12 Grid-Zellen mit einer Auflösung von 0,5 m.

Die Optimierung der einzelnen Terrain-Kacheln kann in mehreren Threads parallelisiert werden. Um das komplette Trapez in der maximalen Auflösung von 0,5 m in Echtzeit, d. h. unter den 100 ms des LiDAR-Sensorzyklus, zu optimieren, werden drei Threads benötigt. Jedes Softwaremodul soll jedoch nur einen Thread verwenden, da die anderen Prozessorkerne für andere Aufgaben benötigt werden. Daher müssen einzelne Kacheln ausgewählt werden, die in dieser feinen Auflösung optimiert werden dürfen, und die verbleibenden Kacheln müssen gröber aufgelöst werden.

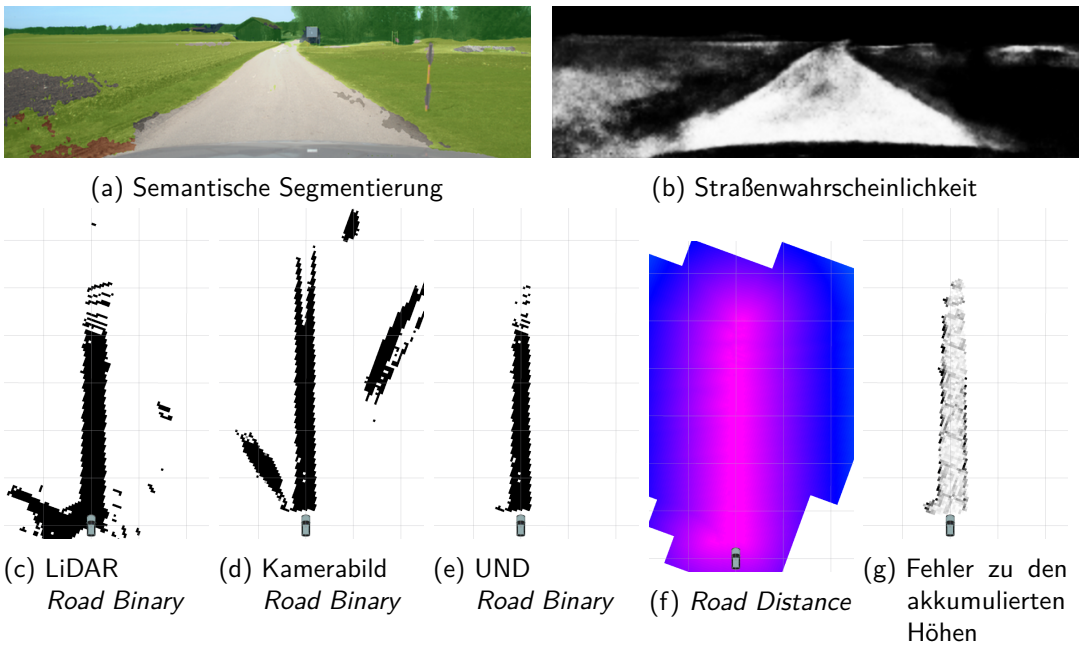
7.3.4 Dynamische Skalierung der Terrain-Kacheln

In der Anwendung des autonomen Fahrens liegt das Interesse vor allem auf Terrain-Bereichen auf und neben Straßen und Wegen. Diese beinhalten außerdem mit höherer Wahrscheinlichkeit unnatürliche Stufen wie Bordsteine, Straßengräben oder Schlaglöcher als die Felder und Wiesen in der Umgebung. Deshalb werden, falls nicht unter Verwendung mehrerer Threads das gesamte Terrain in der höchsten Auflösung geschätzt werden kann, bevorzugt Kacheln in der Nähe eines Wegs höher aufgelöst. Daher werden zunächst wahrscheinliche Straßenbereiche identifiziert, und dann untersucht, ob diese Bereiche mit der aktuellen Auflösung ausreichend genau modelliert werden können, oder ob die Auflösung lokal erhöht werden muss. Abbildung 7.9 zeigt an einem Beispiel das schrittweise Vorgehen.

Die Extraktion von Terrain-Messungen aus der LiDAR-Punktwolke basiert auf der geometrischen Punktwolken-Segmentierung von Burger und Wuensche [2018]. Wie in Abbildung 6.3 gezeigt wurde, segmentiert diese nicht nur Hindernis- und Freiraumpunkte, sondern anhand ebener Flächen auch Bodenflächenpunkte [Burger et al., 2019]. Diese können verwendet werden, um wahrscheinliche Straßenbereiche zu identifizieren. Dazu wird aus den segmentierten Bodenflächenpunkten im Mess-Grid der Terrain-Schätzung ein binäres Grid-Layer namens *Road Binary* erzeugt (Abbildung 7.9c), das anzeigt, ob in eine Mess-Grid-Zelle mindestens einmal ein Bodenflächenpunkt gefallen ist (1) oder nicht (0). Da die geometrische LiDAR-Punktwolken-Segmentierung jedoch lediglich ebene Flächen identifiziert, werden oft fälschlicherweise ebene Wiesen als Bodenfläche segmentiert, wodurch in der *Road Binary* zu viele Zellen als Straßen-Zellen markiert werden.

Deshalb wird dieses Merkmal aus der LiDAR-Punktwolke mit Merkmalen aus dem Kamerabild fusioniert: Anhand der Straßenwahrscheinlichkeit aus der semantischen Bild-Segmentierung von Metzger et al. [2021], die bereits zur Bestimmung einer Befahrbarkeitswahrscheinlichkeit im Hindernis-Grid verwendet wurde (vgl. Abschnitt 6.4), wird ein zweites *Road-Binary*-Layer erzeugt. Dazu wird das Mess-Grid mit der Straßenwahrscheinlichkeit aus der semantischen Segmentierung eingefärbt. Die daraus für eine Grid-Zelle erhaltenen Straßenwahrscheinlichkeiten werden dann über die Zeit akkumuliert. Dieses Vorgehen wird im folgenden Abschnitt 7.3.5 beschrieben. Alle Grid-Zellen mit einer akkumulierten Straßenwahrscheinlichkeit größer 0,5 werden dann im zweiten, kamerabasierten *Road-Binary*-Layer als mögliche Straßen-Zellen mit einer *Road Binary* von 1 markiert (Abbildung 7.9d).

Da jedoch, wie in Abschnitt 6.7 gezeigt wurde, auch die semantische Segmentierung des Kamerabilds sehr fehleranfällig ist, und, wie im Beispiel zu sehen ist, wiederum andere falsch-positive Zellen erzeugt, werden die zwei *Road Binary* Layers mit einem logischen UND kombiniert (Abbildung 7.9e). Dadurch werden nur durch beide Segmentierungen als Bodenflächenpunkt bzw. Straßenpixel segmentierte Grid-Zellen als mögliche Straßen-Zellen angenommen und so falsch-positive Straßen-Zellen minimiert. In anderen Anwendungsfällen könnte alternativ auch ein logisches ODER verwendet werden, um falsch-negative Straßen-Zellen zu reduzieren.



(a) Semantische Segmentierung (b) Straßenwahrscheinlichkeit

(c) LiDAR *Road Binary* (d) Kamerabild *Road Binary* (e) UND *Road Binary* (f) *Road Distance* (g) Fehler zu den akkumulierten Höhen

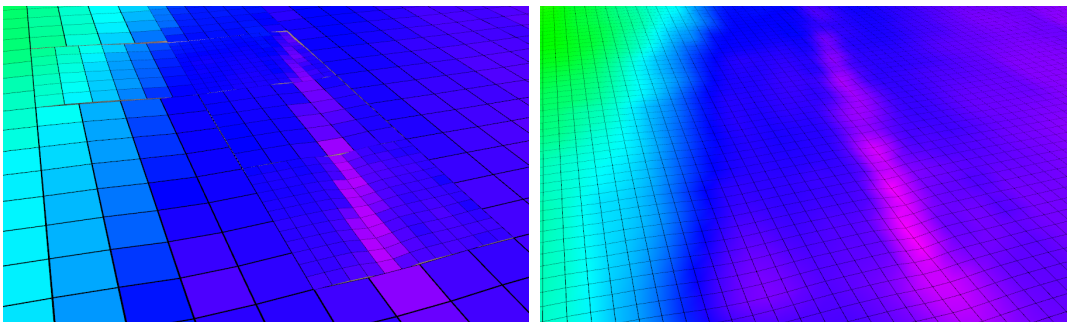
Abbildung 7.9:
Überblick über die einzelnen Schritte, um Zellen von Interesse und ihre Genauigkeit zu bestimmen. (a) und (b) zeigen ein Kamerabild der Szene mit seiner semantischen Segmentierung durch Metzger et al. [2021] und der daraus bestimmten Straßenwahrscheinlichkeit. Zur Bestimmung von Zellen von Interesse werden aus der LiDAR-Punktwolken-Segmentierung und dem semantisch segmentierten Kamerabild zwei *Road-Binary*-Layer (c) und (d) mit möglichen Straßen-Zellen (schwarz) akkumuliert und (e) über eine UND-Verknüpfung fusioniert. (f) Darauf basierend gibt die *Road Distance* die Distanz jeder Grid-Zelle zur nächsten wahrscheinlichen Straßen-Zelle aus (e) in Metern an (pink für 0 m bis blau für mehr als 1,5 m). Zellen mit einer *Road Distance* kleiner 1,5 m gelten als Zellen von Interesse. (g) Für die Zellen von Interesse wird der Fehler der geglätteten Höhen zu den akkumulierten Höhen betrachtet. Fehler größer als 5 cm sind schwarz dargestellt. Im Beispiel werden diese durch Gräben links und rechts eines Feldwegs verursacht. Für die entsprechenden Grid-Kacheln, die diese Zellen von Interesse mit Fehlern zu den akkumulierten Höhen größer als 5 cm enthalten, wird im Anschluss die Auflösung erhöht. Das in der Visualisierung überlagerte Gitter hat eine Kantenlänge von 10 m. Farblegende der semantischen Segmentierung: sky, asphalt, gravel, soil, grass, forest, ego_vehicle.

Aufgrund der Charakteristik einer LiDAR-Punktwolke ist die gewonnene fusionierte *Road Binary* lückenhaft. Zudem kann sie falsch-negative Bereiche auf dem Weg enthalten, sobald eine der beiden Segmentierungen den Bereich nicht erkennt. Daher kann nicht unmittelbar anhand der *Road Binary* entschieden werden, ob eine Terrain-Kachel eine höhere Auflösung erhalten soll oder nicht. Da nicht nur Grid-Zellen auf dem Weg von Interesse sind, sondern auch die Grid-Zellen knapp neben dem Weg noch relevant sind, wird stattdessen die Entfernung einer Grid-Zelle zur nächsten potenziellen Straßen-Grid-Zelle betrachtet: Mithilfe der Distanztransformation der Bildverarbeitungsbibliothek OpenCV [Bradski, 2000] wird für jede Grid-Zelle die Entfernung (in Metern) von dieser Grid-Zelle zur nächsten Grid-Zelle mit einer *Road Binary* von 1 bestimmt. Diese Entfernung stellt das Grid-Layer *Road Distance* dar

(Abbildung 7.9f). Grid-Zellen mit einer *Road Distance* kleiner als 1,5 m, also Zellen, die entweder auf oder maximal 1,5 m neben dem Weg liegen, gelten als Zellen von Interesse für die hochaufgelöste Terrain-Schätzung.

Diese Zellen von Interesse sollen in der Terrain-Schätzung so genau wie möglich modelliert werden. Deshalb wird für diese Zellen geprüft, wie stark die geschätzte, geglättete Höhe von der aus den Messungen akkumulierten Höhe abweicht (Abbildung 7.9g). Abweichungen von bis zu 5 cm sind durch Messrauschen und Fehler in der Akkumulierung des Minimums zu erklären. Eine Abweichung von mehr als 5 cm könnte jedoch implizieren, dass ein Bordstein oder ein negatives Hindernis weggeglättet wurden. Daher wird in diesem Fall das Glättungs-Grid der entsprechenden Terrain-Kachel auf die nächsthöhere Auflösung hochskaliert und die Kachel mit höherer Auflösung erneut optimiert. Dies wird wiederholt, bis der Fehler zu den akkumulierten Höhen weniger als 5 cm beträgt oder die maximale Auflösung erreicht wurde. Grid-Kacheln, die keine Zelle von Interesse enthalten, werden auf die minimale Auflösung herunterskaliert (ohne die Abweichung der geschätzten Höhen auszuwerten). Auf diese Art und Weise werden die einzelnen Terrain-Kacheln dynamisch zwischen 1,5 m, 1,0 m und 0,5 m Zell-Kantenlänge skaliert.

An andere Module ausgegeben wird jedoch schließlich für die einfachere Weiterverarbeitung ein Terrain-Grid einheitlicher Auflösung. Dazu werden für die Ausgabe die im Glättungs-Grid geschätzten Höhen aller Terrain-Kacheln durch Interpolation auf die höchste verwendete Auflösung hochskaliert. Ein Beispiel des Ergebnisses ist in Abbildung 7.10 zu sehen.



(a) Tatsächlich optimierte Grid-Kacheln (b) Skaliertes Ausgabe-Grid

Abbildung 7.10:
Die einzelnen in (a) gezeigten Grid-Kacheln werden zu einer gemeinsamen Auflösung von 0,5 m hochskaliert, um das Ausgabe-Grid in (b) zu erzeugen.

7.3.5 Schätzung einer Straßenwahrscheinlichkeit und Einfärben des Terrain-Grids

Zur Fusion des LiDAR-basierten Terrain-Grids mit Informationen aus dem Kamerabild kann das Terrain-Grid mit (Farb-)Bildern eingefärbt werden. Hierfür wird jeder Stützpunkt c des einzufärbenden Grids in das entsprechende Kamerakoordinatensystem transformiert und in das Bild projiziert, um eine Farbinformation auszulesen. Die Position eines Stützpunkts im Odometriekoordinatensystem bestimmt sich dabei aus seinen x- und y-Koordinaten sowie der geschätzten Höhe als z-Koordinate.

Farbiges Terrain-Grid Wird das Terrain-Grid mit einem RGB-Kamerabild eingefärbt, so kann ein farbiges Terrain-Grid erzeugt werden (siehe Abbildung 7.8), das beispielsweise in der Kreuzungserkennung in Kapitel 9 Anwendung findet. Hierfür wird eine hohe Auflösung benötigt. Daher wird zum Einfärben das von der Terrain-Schätzung ausgegebene Grid auf eine Auflösung von 0,2 m hochskaliert. Die durch die Projektion der Zell-Mittelpunkte ins Kamerabild erhaltenen Farbinformationen werden über die Zeit gemittelt. Entsprechend Bayerl und Wuensche [2014] berechnet sich am Beispiel des Rotkanals die akkumulierte Farbinformation $r_{\mathsf{RGB},1:k}$ aus der aus dem aktuellen Kamerabild ausgelesenen Farbinformation $r_{\mathsf{RGB},k}$ und der vorherigen akkumulierten Farbinformation $r_{\mathsf{RGB},1:k-1}$ als:

$$r_{\mathsf{RGB},1:k} = \frac{r_{\mathsf{RGB},1:k-1} \cdot (w-1) + r_{\mathsf{RGB},k}}{w} \,, \tag{7.11}$$

wobei

$$w = \min(4,\, k+1) \,. \tag{7.12}$$

Wird eine Grid-Zelle zum ersten Mal eingefärbt, wird die Farbinformation einmalig direkt übernommen. Die Berechnung des Grün- und Blaukanals erfolgt analog.

Schätzung einer Straßenwahrscheinlichkeit Um zu entscheiden, welche Terrain-Kacheln höher aufgelöst werden dürfen, muss das Terrain-Grid außerdem mit den Straßenwahrscheinlichkeiten aus einer semantischen Segmentierung eingefärbt werden. Darauf basierend wird im Mess-Grid über die Zeit eine Straßenwahrscheinlichkeit für jede Grid-Zelle geschätzt. Dazu wird für jede Zelle $\mathsf{c_m}$ des Mess-Grids anhand der Projektion ihres Zell-Mittelpunkts in das Straßenwahrscheinlichkeitsbild (unter Verwendung einer aus dem Glättungs-Grid interpolierten z-Koordinate) die Wahrscheinlichkeit

$$p(\mathrm{road}^{(\mathsf{c_m})} |\, \mathbf{y}_{\mathsf{sem},k}) = p_{\mathsf{road}}(\mathsf{c_m}) \tag{7.13}$$

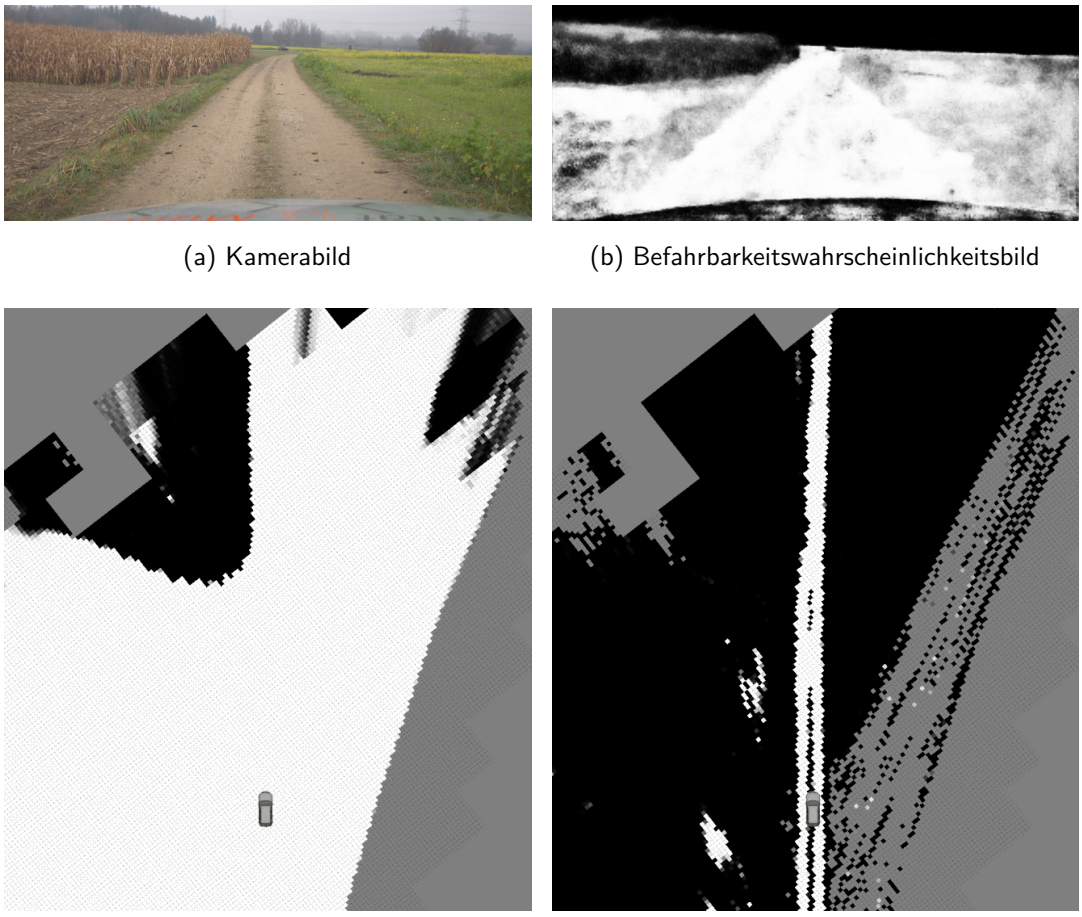
dafür bestimmt, dass die Zelle auf Basis der Messungen $\mathbf{y}_{\mathsf{sem},k}$ zum Zeitschritt k Teil einer Straße oder eines Wegs ist. Nicht im Kamerabild beobachtbare Grid-Zellen erhalten eine A-priori-Straßenwahrscheinlichkeit von $p(\mathrm{road}^{(\mathsf{c_m})} | \mathbf{y}_{\mathsf{sem},k}) = 0{,}5$. Analog zum Update der Belegtheits- und Befahrbarkeitswahrscheinlichkeit im Hindernis-Grid (vgl. Abschnitt 6.4.2) wird die Straßenwahrscheinlichkeit über die log-odds-Repräsentation des Bayes-Filters rekursiv als Logit

$$\mathrm{logit}(\mathrm{road}^{(\mathsf{c_m})} |\, \mathbf{y}_{\mathsf{sem},1:k}) = \log\left(\frac{p(\mathrm{road}^{(\mathsf{c_m})} |\, \mathbf{y}_{\mathsf{sem},k})}{1 - p(\mathrm{road}^{(\mathsf{c_m})} |\, \mathbf{y}_{\mathsf{sem},k})}\right) + \mathrm{logit}(\mathrm{road}^{(\mathsf{c_m})} |\, \mathbf{y}_{\mathsf{sem},1:k-1}) \tag{7.14}$$

geschätzt, woraus sich die geschätzte Straßenwahrscheinlichkeit

$$p(\mathrm{road}^{(\mathsf{c_m})} |\, \mathbf{y}_{\mathsf{sem},1:k}) = 1 - \frac{1}{1 + \exp\Big(\mathrm{logit}(\mathrm{road}^{(\mathsf{c_m})} |\, \mathbf{y}_{\mathsf{sem},1:k})\Big)} \tag{7.15}$$

berechnen lässt. Analog kann durch das Einfärben des Terrain-Grids mit einem Befahrbarkeitswahrscheinlichkeitsbild auch eine Befahrbarkeitswahrscheinlichkeit geschätzt werden. Abbildung 7.11 zeigt ein Beispiel.



(a) Kamerabild

(b) Befahrbarkeitswahrscheinlichkeitsbild

(c) Geschätzte Befahrbarkeitswahrscheinlichkeit

(d) Geschätzte Straßenwahrscheinlichkeit

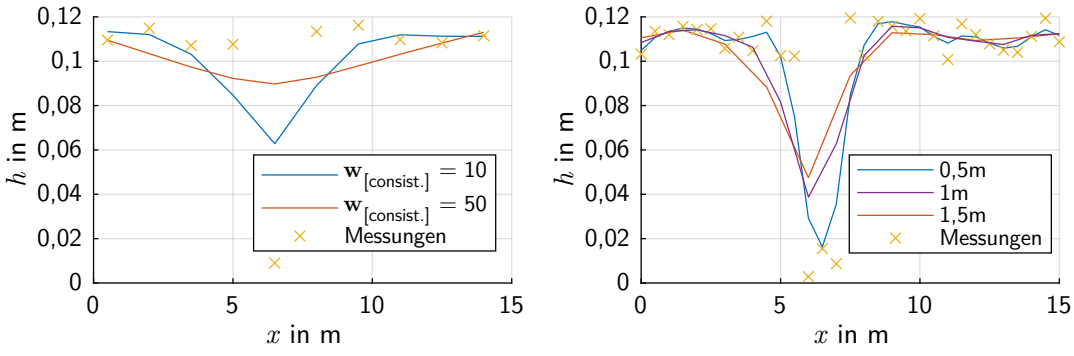
Abbildung 7.11:
Beispiel für das mit den Ergebnissen der semantischen Segmentierung von Metzger et al. [2021] eingefärbte Mess-Grid. (a) zeigt ein Kamerabild der Szene. (b) zeigt das aus der semantischen Segmentierung gewonnene Befahrbarkeitswahrscheinlichkeitsbild (vgl. Abschnitt 6.4.1). (c) zeigt die im Mess-Grid geschätzte Befahrbarkeitswahrscheinlichkeit und (d) die analog geschätzte Straßenwahrscheinlichkeit. In der Befahrbarkeitswahrscheinlichkeit ist deutlich der Rand des nicht-befahrbaren Maisfelds zu erkennen (innerhalb des Maisfelds sind die Artefakte der semantischen Segmentierung am oberen Rand des Maisfelds wiederzufinden). Die Straßenwahrscheinlichkeit zeigt deutlich den Feldweg mit Grünstreifen in der Mitte.

7.4 Ergebnisse

Abschließend sollen die Resultate der Terrain-Schätzung mit dynamischer Auflösungsanpassung ausgewertet und mit anderen Methoden verglichen werden. Da die Terrain-Schätzung ein vereinfachtes Modell der Bodenoberfläche schätzt und dieses Modell für verschiedene Anwendungen unterschiedlich stark geglättet sein soll, ist die Terrain-Schätzung nicht quantitativ zu bewerten. Stattdessen können qualitative Ergebnisse betrachtet sowie die Laufzeit des Verfahrens beurteilt werden.

7.4.1 Qualitative Ergebnisse

Zunächst soll die Wahl der Auflösung der Mess- und Glättungs-Grids veranschaulicht werden. Abbildung 7.12 zeigt dazu ein zweidimensionales Beispiel: Ein 1,5 m breites Loch von etwa 11 cm Tiefe soll durch die Terrain-Schätzung abgebildet werden. Bei groben Mess- und Glättungs-Grids mit 1,5 m Auflösung und einem hohen Glättungsfaktor wird das Loch als 15 m breite, sehr leichte Mulde modelliert, die sehr stark von den Messungen abweicht. Mit einem reduzierten Glättungsfaktor liegt die Terrain-Schätzung zwar etwas näher an den Messungen, das Loch kann aber dennoch nur mit seiner halben Tiefe modelliert werden, da nur eine Messung hinein fällt. Daher muss zur Erhöhung der Genauigkeit zwingend die Auflösung des Mess-Grids erhöht werden. Bei einer Mess-Grid-Auflösung von 0,5 m können die Messungen deutlich besser abgebildet werden – auch bei einer gleichbleibenden Glättungs-Grid-Auflösung von 1,5 m. Wird die Glättungs-Grid-Auflösung ebenfalls auf 0,5 m erhöht, so wird das Loch schließlich in seiner korrekten Breite und Tiefe modelliert. Die Terrain-Schätzung benötigt also ein feines Mess-Grid, damit die Genauigkeit dann mit steigender Auflösung des Glättungs-Grids weiter erhöht werden kann.



(a) 1,5 m Mess-Grid-Auflösung, 1,5 m Glättungs-Grid-Auflösung, verschiedene $\mathbf{w}_{[\text{consist.}]}$

(b) 0,5 m Mess-Grid-Auflösung, verschiedene Glättungs-Grid-Auflösungen, $\mathbf{w}_{[\text{consist.}]} = 10$

Abbildung 7.12:
Resultierende Terrain-Schätzung für die in gelb gezeigten Messungen bei verschiedenen Grid-Auflösungen und Glättungsfaktoren $\mathbf{w}_{[\text{consist.}]}$. Die Messungen für die reduzierte Grid-Auflösung in (a) wurden durch Mittelung der entsprechenden in eine Mess-Grid-Zelle fallenden Messungen in (b) gewonnen.

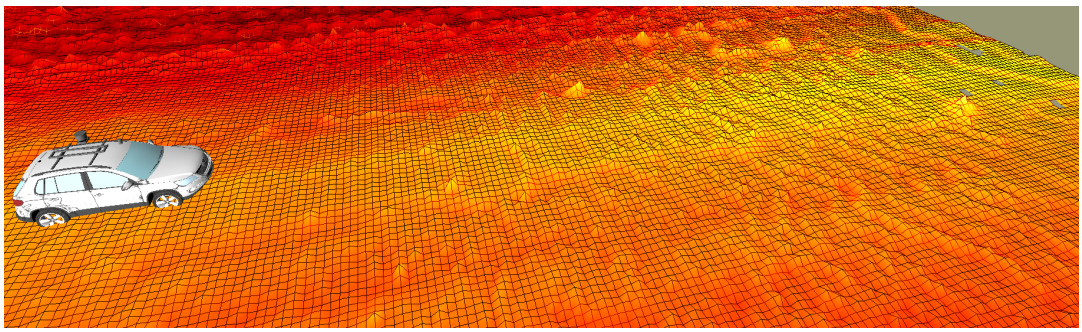
Wenn sich das Modell so stark an den Messungen orientieren soll, weshalb wird dann überhaupt eine glättende Terrain-Schätzung benötigt? Die Notwendigkeit der Glättung zeigt sich im Vergleich zwischen der entwickelten Terrain-Schätzung und der akkumulierten Geländekarte von Jaspers et al. [2017]. Abbildung 7.13 zeigt zwei Phänomene dieser Geländekarten an Realdatenbeispielen: Aufgrund des Rauschens sowohl der Messungen des LiDAR-Sensors als auch der Eigenbewegungsschätzung weisen die akkumulierten Terrainhöhen auch auf absolut ebenen Wegen ein Rauschen auf, insbesondere in größerer Entfernung (vgl. Abbildung 7.13a). Zudem führen einzelne Messausreißer direkt zu Ausreißern in der Geländekarte (vgl. Abbildung 7.13e). Das Rauschen wird zwar über die Zeit herausgefiltert und sinkt damit in näherer Entfernung, dadurch sinkt jedoch auch die effektive Vorausschau. Auch die Ausreißer können mit der zeitlichen Filterung nicht ausreichend gut aus der Geländekarte eliminiert werden. Die entwickelte Terrain-Schätzung erzeugt jedoch aufgrund der zusätzlichen räumlichen Glättung in beiden Beispielen ein rausch- und ausreißerfreies Terrain-Modell (vgl. Abbildungen 7.13b und 7.13f).

Ein weiterer Vorteil der entwickelten Terrain-Schätzung gegenüber den Geländekarten ist in Abbildung 7.14 visualisiert. Da die Geländekarten gleichermaßen den Boden und Hindernisse kartieren, erzeugt ein über den Weg ragender Ast eines Baums eine Erhebung in der Geländekarte. Dadurch ist keine Terrain-Information für den Boden unter dem Ast verfügbar. Bei der entwickelten Terrain-Schätzung jedoch, bei der lediglich der Boden modelliert wird und die Hindernisse in einem separaten Hindernis-Grid kartiert werden, ist das Terrain-Modell durchgängig. Dadurch stehen auch unter dem Baum Terrain-Informationen zur Verfügung, mit denen beispielsweise im Wegetracking die Lage der Straße bestimmt werden kann.

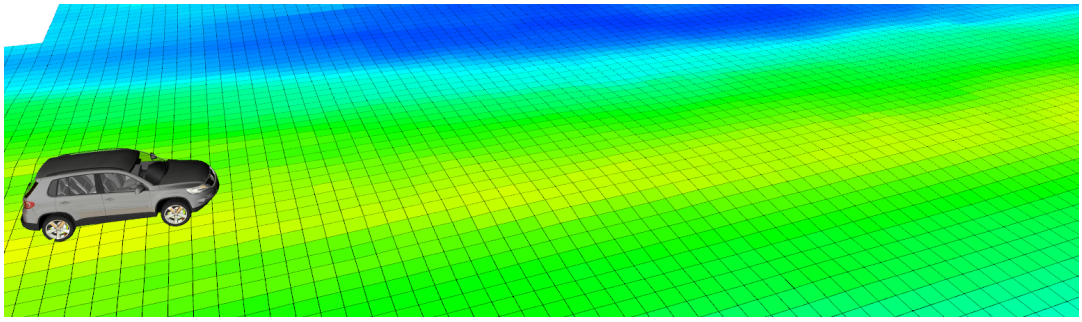
Da die Geländekarten keine räumliche Glättung durchführen, können diese sehr fein aufgelöst werden. So sind mit ihrer Auflösung von 0,2 m im Beispiel in Abbildung 7.14c deutlich die Gräben links und rechts des Wegs sichtbar. In dieser Auflösung wäre die räumliche Glättung durch Optimierung jedoch zu rechenaufwendig. Bei einer Terrain-Schätzung mit einer reduzierten Auflösung von 1,5 m werden diese Details wiederum weggeglättet (vgl. Abbildung 7.14f). Einen Kompromiss, bei dem die Gräben noch zu erkennen sind, bildet eine Auflösung von 0,5 m (vgl. Abbildung 7.14e). Damit kann das Terrain aber bereits nicht mehr in Echtzeit geglättet werden. Indem die Auflösung der Terrain-Schätzung dynamisch an die Gegebenheiten angepasst wird, können bei der Terrain-Schätzung mit dynamischer Auflösung jedoch die Gräben mit hoher Auflösung modelliert werden (vgl. Abbildung 7.14d), gleichzeitig aber durch eine Reduktion der Auflösung in anderen Bereichen dennoch die Echtzeitfähigkeit bewahrt werden. Das Ergebnis ist dabei optisch kaum von einer Terrain-Schätzung mit einer durchgängig hohen Auflösung zu unterscheiden.

Dies verdeutlicht sich auch im eingangs in Abbildung 7.2 gezeigten Beispiel eines Feldwegs mit einem Schlagloch, das in Abbildung 7.15 aufgegriffen wird. Auch hier zeigt sich kaum ein visueller Unterschied zwischen der Terrain-Schätzung mit dynamischer Auflösung und der Terrain-Schätzung mit 0,5 m Auflösung.

Weitere Beispiele finden sich in einem Demo-Video, das unter `https://www.youtube.com/watch?v=i2-XZ92jcd0` verfügbar ist.



(a) Geländekarte von Jaspers et al. [2017]



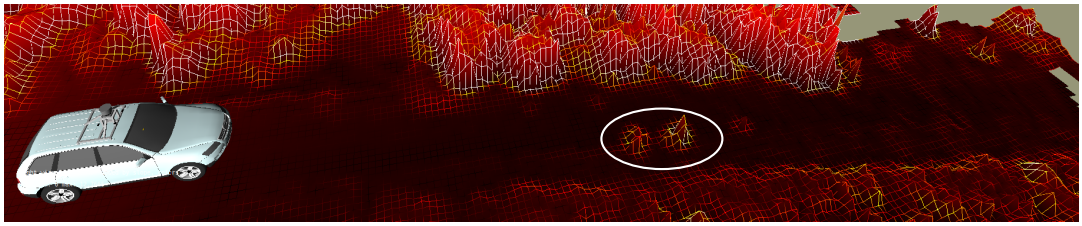
(b) Terrain-Schätzung



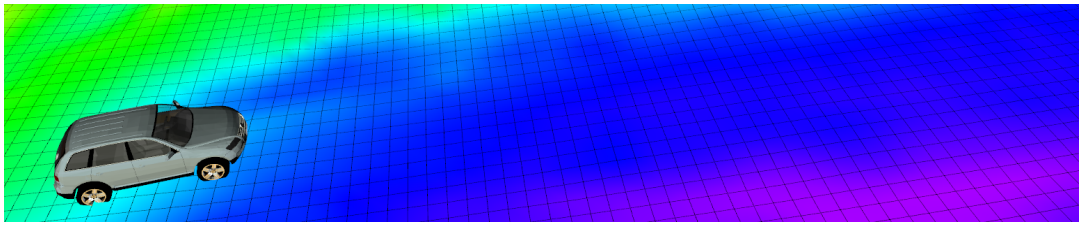
(c) Kamerabild zu (a) und (b)



(d) Kamerabild zu (e) und (f)



(e) Geländekarte von Jaspers et al. [2017]

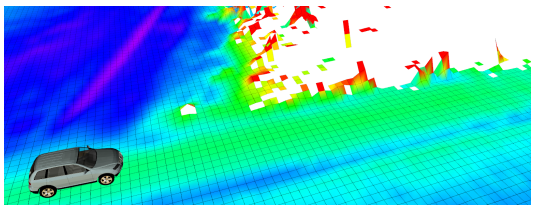


(f) Terrain-Schätzung

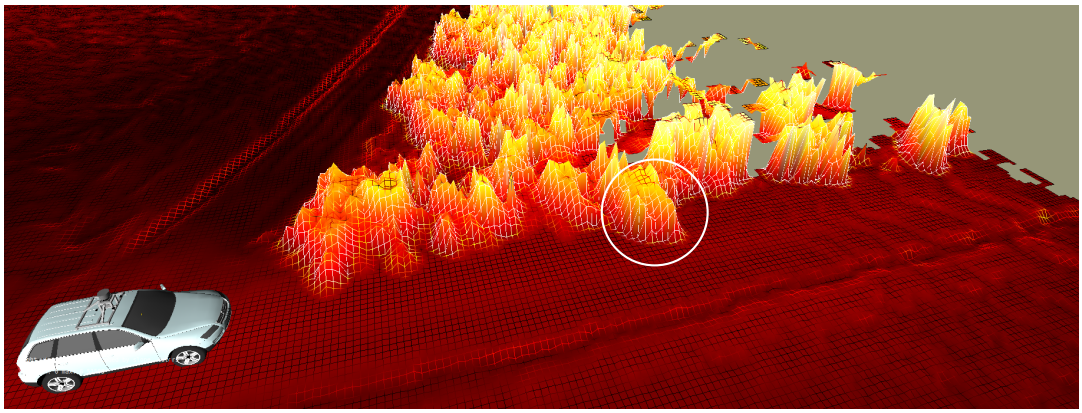
Abbildung 7.13:
Vergleich der entwickelten Terrain-Schätzung mit der Geländekarte von Jaspers et al. [2017] an zwei Beispielen (Abbildungen (a) bis (c) und Abbildungen (d) bis (f)). Die Farbe der Terrain-Schätzung zeigt die geschätzte Höhe in Odometriekoordinaten von lila für niedriges Terrain über blau und grün zu gelb für höhergelegenes Terrain. Die Farbskala der Geländekarte von Jaspers et al. [2017] geht von dunkelrot für niedrige Höhen bis gelb für höhere Höhen.



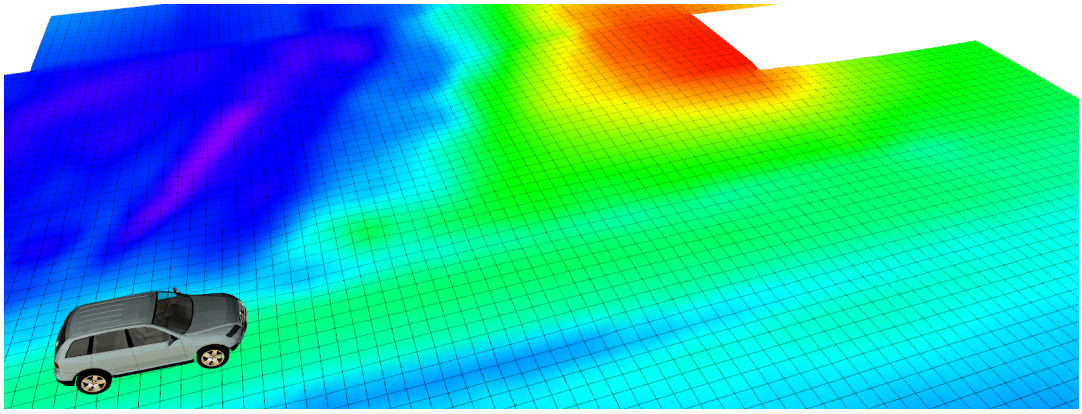
(a) Kamerabild



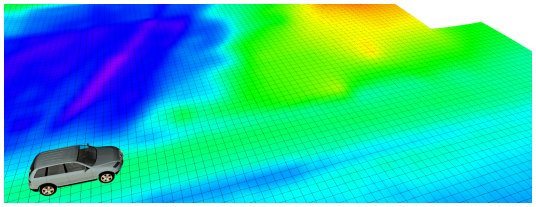
(b) Akkumulierte Messungen



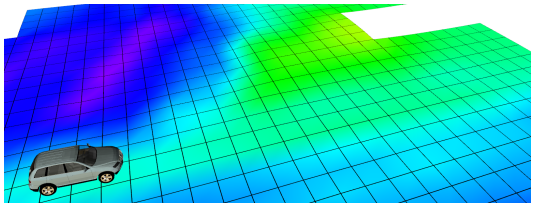
(c) Geländekarte von Jaspers et al. [2017]



(d) Terrain-Schätzung mit dynamischer Auflösung

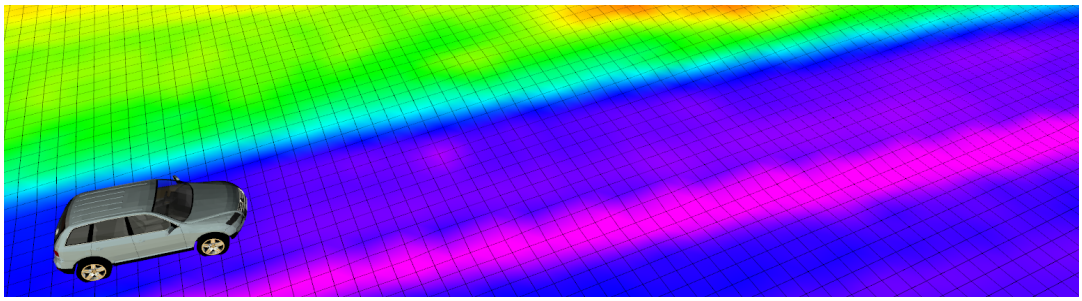


(e) Terrain-Schätzung mit hoher Auflösung

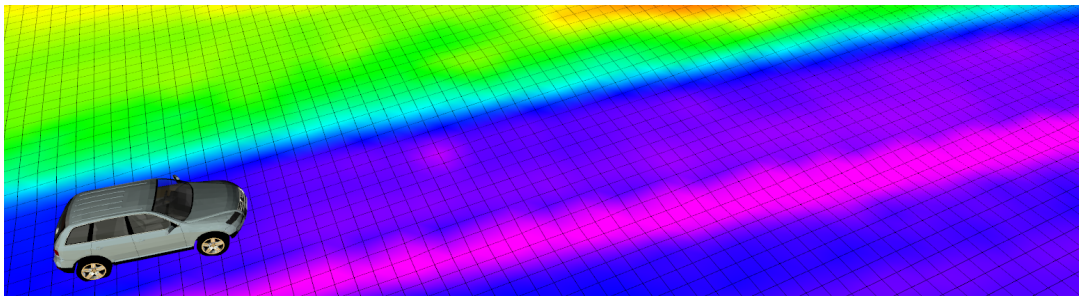


(f) Terrain-Schätzung mit niedriger Auflösung

Abbildung 7.14:
Vergleich der entwickelten Terrain-Schätzung mit der Geländekarte von Jaspers et al. [2017]. Gezeigt werden dazu das Ergebnis der Terrain-Schätzung mit dynamischer Auflösung (0,5 m bis 1,5 m Auflösung in der Optimierung, Ausgabe-Grid überall auf 0,5 m skaliert), mit hoher Auflösung (0,5 m) und mit niedriger Auflösung (1,5 m) sowie die im Mess-Grid (mit 0,5 m Auflösung) akkumulierten Messungen. Alle Bilder der Terrain-Schätzung teilen dieselbe Farbskala von lila für niedriges Terrain über blau und grün zu rot für höhergelegenes Terrain (in Odometriekoordinaten). Die Farbskala der Geländekarte von Jaspers et al. [2017] geht von dunkelrot für niedrige Höhen bis gelb für höhere Höhen.



(a) Terrain-Schätzung mit dynamischer Auflösung

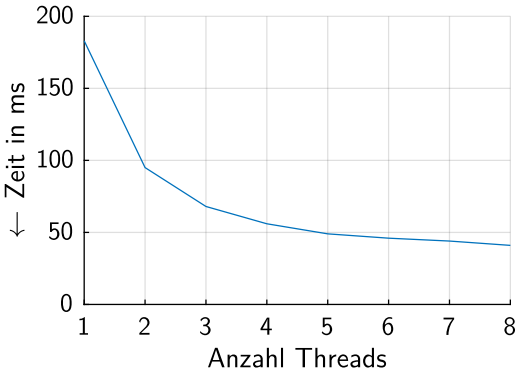


(b) Terrain-Schätzung mit hoher Auflösung

Abbildung 7.15:
Vergleich des resultierenden Terrain-Modells der Terrain-Schätzung mit dynamischer Auflösung (0,5 m bis 1,5 m) mit dem Ergebnis der Terrain-Schätzung mit durchgehend hoher Auflösung (0,5 m) für die eingangs dieses Kapitels gezeigte Szene mit einem Schlagloch (Abbildung 7.2). Während dort die einzelnen Kacheln in ihrer dynamischen Auflösung zu sehen waren, ist hier das auf eine Auflösung von 0,5 m skalierte Ausgabe-Grid abgebildet.

Auflösung	Rechenzeit
Niedrig (1,5 m)	22 ms
Hoch (0,5 m)	183 ms
Dynamisch	72 ms

(a)



(b)

Abbildung 7.16:
(a) Durchschnittliche Laufzeit der Terrain-Schätzung für das in den Abbildungen 7.2 und 7.15 gezeigte Szenario. Verglichen werden die benötigten Rechenzeiten bei einer durchgehend niedrigen Auflösung (1,5 m), bei einer durchgehend hohen Auflösung (0,5 m) und mit der dynamischen Auflösung, jeweils unter Verwendung eines Threads. Bei der dynamischen Auflösung werden durchschnittlich 77 Kacheln mit der niedrigsten Auflösung von 1,5 m geglättet, 2 Kacheln mit 1,0 m Auflösung und 15 Kacheln mit der höchsten Auflösung von 0,5 m. (b) Durchschnittliche Rechenzeit, die benötigt wird, um alle Terrain-Kacheln in höchster Auflösung zu optimieren, in Abhängigkeit von der Anzahl verwendeter Threads, auf die die zu optimierenden Terrain-Kacheln aufgeteilt werden.

7.4.2 Laufzeit

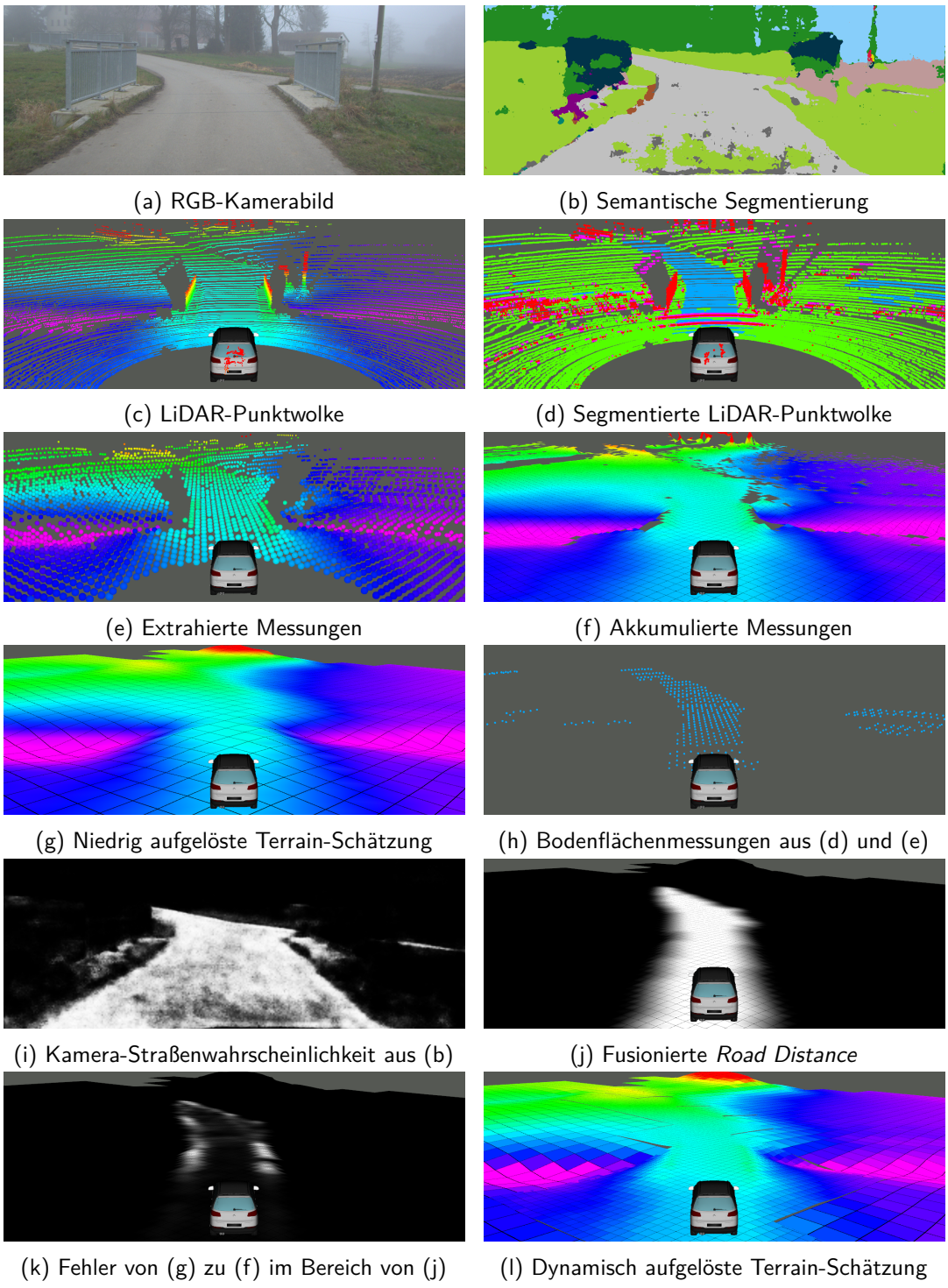
Während die Resultate der Terrain-Schätzung mit dynamischer Auflösung denen einer hochauflösenden Terrain-Schätzung kaum nachstehen, verringert sich die Rechenzeit jedoch erheblich: Wie Abbildung 7.16a zeigt, benötigt die Terrain-Schätzung mit dynamischer Auflösung nur 40% der Rechenzeit der hochauflösenden Terrain-Schätzung. Damit ist die Terrain-Schätzung mit dynamischer Auflösungsanpassung im Gegensatz zur hochauflösenden Terrain-Schätzung echtzeitfähig. Bei der hochauflösenden Terrain-Schätzung werden dazu, wie in Abbildung 7.16b zu sehen ist, mehrere parallel laufende Threads benötigt. Eine sichere Echtzeitfähigkeit ist erst ab drei Threads gewährleistet. Die Terrain-Schätzung mit dynamischer Auflösung erzielt jedoch ein sehr ähnliches Ergebnis mit nur einem Thread.

7.5 Diskussion

Zusammenfassend zeigt Abbildung 7.17 an einem zusammenhängenden Beispiel noch einmal alle verschiedenen Schritte und Zwischenergebnisse der vorgestellten Feldweg-Terrain-Schätzung mit dynamischer Auflösungsanpassung: Abbildung 7.17a zeigt zur Referenz ein Kamerabild der Szene. Eine LiDAR-Punktwolke (7.17c) der Szene wird durch Burger und Wuensche [2018] geometrisch segmentiert (7.17d). Aus den als Freiraum und Bodenfläche klassifizierten Punkten werden Messungen für die einzelnen Stützpunkte extrahiert (7.17e) und über die Zeit akkumuliert (7.17f). Daraus wird in der ersten Optimierung einer Terrain-Kachel ein niedrig aufgelöstes Terrain-Modell geschätzt (7.17g).

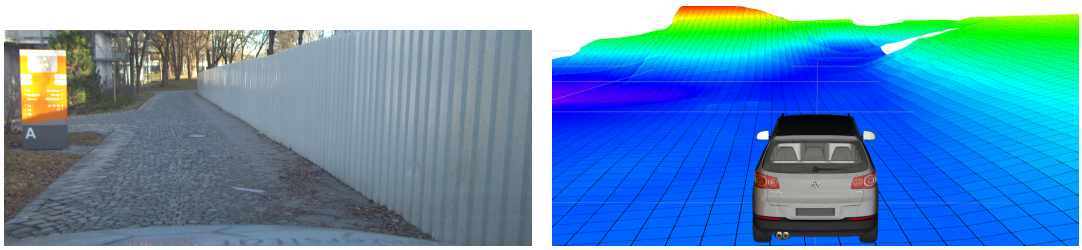
Aus den Bodenflächenpunkten der geometrischen LiDAR-Punktwolken-Segmentierung (7.17h) und der aus der semantischen Segmentierung des Kamerabilds durch Metzger et al. [2021] (7.17b) gewonnenen Straßenwahrscheinlichkeit (7.17i) wird der Abstand aller Grid-Zellen zu wahrscheinlichen Straßen-Zellen berechnet (7.17j). Im Bereich wahrscheinlicher Straßen wird der Fehler der geschätzten Höhen (7.17g) zu den akkumulierten Höhen (7.17f) ausgewertet (7.17k), und gegebenenfalls die Auflösung der Grid-Kachel erhöht. Auf diese Art und Weise wird die Feldweg-Terrain-Schätzung dynamisch aufgelöst (7.17l).

Die Terrain-Schätzung schätzt unter Berücksichtigung von Messrauschen und Ausreißern ein kontinuierliches Modell der Bodenoberfläche. Dabei wird auch der Untergrund unter Hindernissen geschätzt, wodurch auch auf Feldwegen mit überhängender Vegetation eine Terrain-Information bereitgestellt wird. Die Hindernisse werden stattdessen im in Kapitel 6 beschriebenen Hindernis-Grid kartiert (beispielsweise die Brückengeländer in Abbildung 7.17). Die Terrain-Schätzung modelliert wiederum eventuelle negative Hindernisse (wie beispielsweise dem unter der Brücke verlaufenden Bach in Abbildung 7.17). Mit der dynamischer Auflösungsänderung ist es dabei möglich, die Terrain-Oberfläche in Echtzeit so präzise zu schätzen, dass Schlaglöcher auf dem Weg oder Gräben neben dem Weg modelliert werden können. Dadurch kann die entwickelte Feldweg-Terrain-Schätzung im Folgenden im Wegetracking (Kapitel 8) sowie in der Kreuzungserkennung (Kapitel 9) zum Einsatz kommen.



(a) RGB-Kamerabild
(b) Semantische Segmentierung
(c) LiDAR-Punktwolke
(d) Segmentierte LiDAR-Punktwolke
(e) Extrahierte Messungen
(f) Akkumulierte Messungen
(g) Niedrig aufgelöste Terrain-Schätzung
(h) Bodenflächenmessungen aus (d) und (e)
(i) Kamera-Straßenwahrscheinlichkeit aus (b)
(j) Fusionierte *Road Distance*
(k) Fehler von (g) zu (f) im Bereich von (j)
(l) Dynamisch aufgelöste Terrain-Schätzung

Abbildung 7.17:
Zwischenergebnisse der entwickelten Feldweg-Terrain-Schätzung. In den Abbildungen (c), (e), (f), (g) und (l) zeigt die Farbe die Höhe des Punkts bzw. des Terrains von pink für niedrige Höhen über blau und grün bis rot für hohe Höhen (in Odometriekoordinaten), wobei die Skala für alle Abbildungen einheitlich ist. Farblegende für Abbildung (b): sky, asphalt, gravel, soil, grass, forest, fence, building, misc_object. Farblegende für Abbildungen (d) und (h): Hindernis, Freiraum, Bodenfläche, Rauschen. Abbildung (i) zeigt Straßenwahrscheinlichkeiten von 0 in schwarz bis 1 in weiß. Abbildung (j) zeigt eine *Road Distance* von 0 m in weiß bis 1,5 m in schwarz. Abbildung (k) zeigt Abweichungen von 0 cm in schwarz bis 5 cm in weiß. Das Sichtfeld der dreidimensionalen Visualisierungen weicht von dem der Kamera ab.



(a) Kamerabild (b) Terrain-Schätzung

Abbildung 7.18:
Beispiel für das Resultat der Terrain-Schätzung mit dynamischer Auflösung auf einem urbanen Campusweg mit viel Verdeckung aufgrund von Mauern und Gebäuden. Die Hindernisse wie das im Kamerabild abgebildete Schild und die Absperrung sind im Hindernis-Grid als Hindernis kartiert. Betrachtet man die Terrain-Schätzung hinter diesen Hindernissen, zeigen sich falsche Hügel aufgrund von vereinzelten Falschmessungen im sonst verdeckten Bereich. Eine Sensitivitätsanalyse würde dort jedoch eine hohe Unsicherheit aufzeigen.

Die Grenzen des Verfahrens zeigen sich bei urbanen Szenen wie dem in Abbildung 7.18 abgebildeten Beispiel. Auf engen Wegen mit vielen Hindernissen und Strukturen wird zwar der Weg selbst noch gut modelliert, für stark verdecktes Terrain neben dem Weg werden jedoch nicht vorhandene Hügel interpoliert. Dies liegt daran, dass dort die einzig verfügbaren Messungen von falsch segmentierten Hindernispunkten stammen, die ganz ohne tatsächliche Bodenmessungen nicht über die Zeit oder räumlich heraus gefiltert werden können. Daher ist die Terrain-Schätzung nur auf weitgehend offenem Terrain und dem Weg gültig (ohne zwingend die durch eine Sensitivitätsanalyse bestimmbare geschätzte Unsicherheit betrachten zu müssen). In allen Fällen stehen jedoch dem nachfolgenden Straßen- und Wegetracking Informationen über die Höhe und damit die Entfernung eines Wegs zur Kamera zur Verfügung.

8 Straßen- und Wegetracking

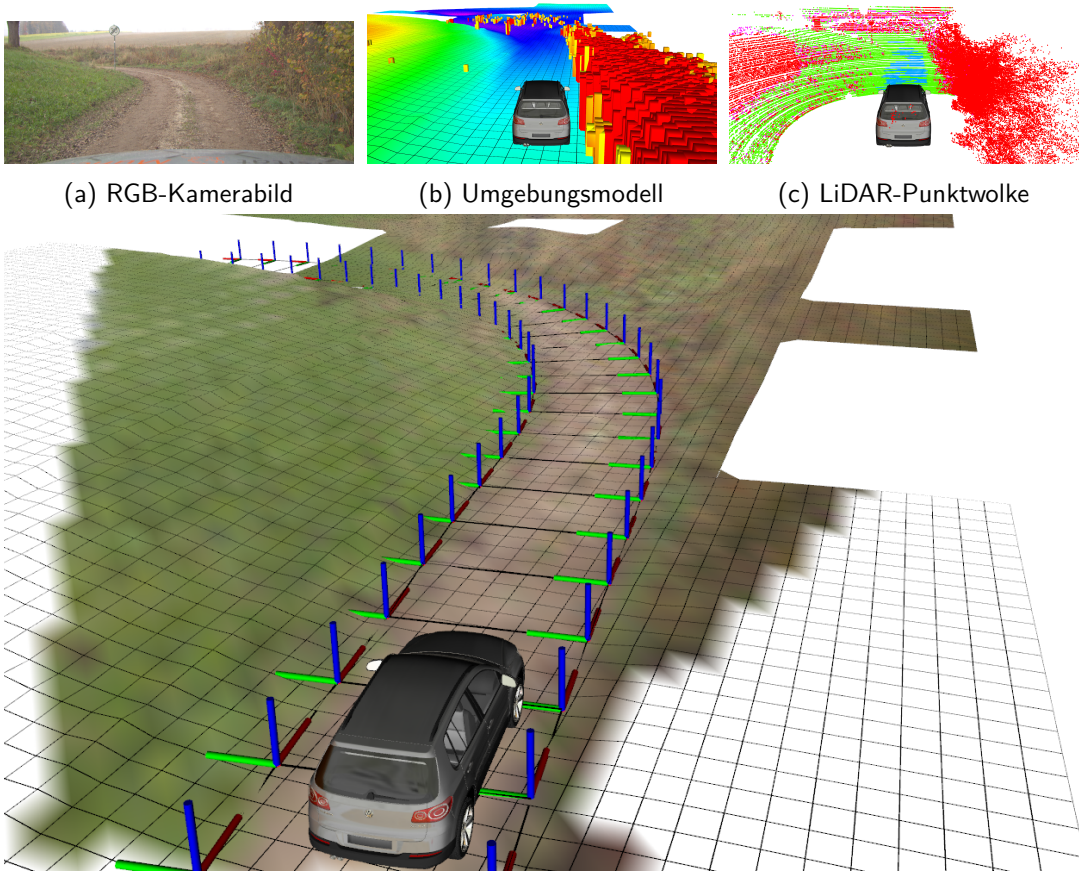
Das Kernelement in der Wahrnehmung der statischen Umgebung eines autonomen Fahrzeugs auf Feldwegen ist schließlich die Wahrnehmung des Feldwegs selbst. Das Fahrzeug soll ohne Kartenwissen in der Lage sein, einem Feldweg autonom zu folgen. Dazu muss die Position und Lage der Fahrbahn auf Basis der eigenen Wahrnehmung geschätzt werden. Hierfür wurde ein Framework zum Straßen- und Wegetracking anhand von Kamera- und LiDAR-Messungen entwickelt, das in [Forkel, Kallwies, und Wuensche, 2021a] und [Forkel und Wuensche, 2022a] vorab veröffentlicht wurde.

8.1 Problemstellung

In der Vergangenheit wurden am Institut für Technik Autonomer Systeme (TAS) spezialisierte Trackingverfahren für Feldwege in eingefärbten Geländekarten entwickelt [Manz et al., 2011, Bayerl et al., 2015]. Da Feldwege allerdings oftmals nur über befestigte Straßen erreichbar sind, sollen nun nicht mehr nur Feldwege, sondern auch ländliche Straßen erkannt werden. Auf Straßen wird jedoch mit höherer Geschwindigkeit gefahren. Dafür werden eine größere Vorausschau sowie eine höhere Genauigkeit benötigt, als mit den Geländekarten erreicht werden können. Im Gegensatz zu Feldwegen können Straßen außerdem aus mehreren Fahrspuren bestehen, weshalb die Modellierung und Erkennung einzelner Fahrspuren nötig wird.

In Zusammenarbeit mit Jan Kallwies wurde daher ein neues Trackingframework entwickelt, mit dem die Fahrspurränder von (mehrspurigen) Straßen und Feldwegen gleichermaßen getrackt werden können. Dabei werden die 6D-Posen einzelner Punkte der Fahrspurränder geschätzt. Während Kallwies [2023] den grundlegenden Schätzprozess einführt, liegt der Fokus dieser Arbeit auf der Entwicklung robuster Messverfahren für alle Arten von Straßen und insbesondere Wegen. Für Messungen im Kamerabild wird ein für Straßen entworfenes Kantenmessverfahren für die Anwendung auf Feldwegen weiterentwickelt. Zusätzlich wird ein Messverfahren in LiDAR-Punktwolken entwickelt. Die jeweiligen Messungen werden dann im Schätzverfahren fusioniert, wobei auch Informationen aus dem in den letzten beiden Kapiteln erarbeiteten Umgebungsmodell berücksichtigt werden. Darüber hinaus wird in dieser Arbeit das Straßenmodell für die Verwendung auf veränderlichen Wegen angepasst und die Robustheit und Vorausschau des Straßen- und Wegetrackings erhöht. Abbildung 8.1 zeigt ein Beispiel eines mit erhöhter Vorausschau getrackten Feldwegs.

Gliederung Nach einer Literaturrecherche in Abschnitt 8.2 beschreibt Abschnitt 8.3 den (für veränderliche Feldwege angepassten) Schätzprozess. Abschnitt 8.4 erklärt anschließend die kantenbasierten Messungen von Straßen und Feldwegen im Kamerabild. Abschnitt 8.5 stellt das Messverfahren in einer segmentierten LiDAR-Punktwolke und die Integration von Informationen aus dem Umgebungsmodell in den Schätzprozess vor. Abschnitt 8.7 zeigt schließlich, dass mithilfe dieser Sensordatenfusion eine Vielzahl an Straßen und Wegen robust getrackt werden kann. Ein Überblick über alle Module des Straßen- und Wegetrackings ist in Abbildung 8.2 zu sehen.



(a) RGB-Kamerabild (b) Umgebungsmodell (c) LiDAR-Punktwolke

(d) Posen der getrackten Fahrspurrandpunkte auf eingefärbtem Terrain-Grid

Abbildung 8.1:
Beispiel für das Tracking eines Feldwegs: Aus den in der oberen Reihe gezeigten Eingangsdaten – (a) einem RGB-Kamerabild, (b) dem LiDAR-basierten Umgebungsmodell und (c) der segmentierten LiDAR-Punktwolke – können die in (d) gezeigten 6D-Posen einzelner Punkte auf dem Wegrand geschätzt werden. Zum Vergleich der Vorausschauweite werden diese Posen auf dem mit dem Kamerabild eingefärbten Terrain-Grid gezeigt, dessen Vorausschau in der Kurve vom entwickelten Wegetracking deutlich übertroffen wird.

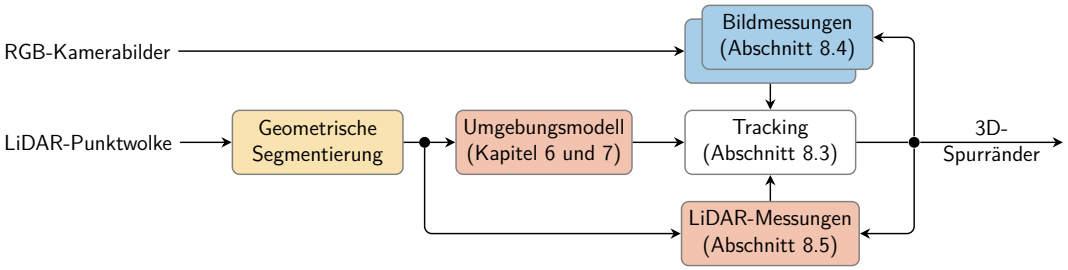


Abbildung 8.2:
Ablaufplan des entwickelten Straßen- und Wegetrackings durch die Fusion von Messungen aus Kamerabildern (blau) mit Messungen aus segmentierten LiDAR-Punktwolken (rot). Beide Messverfahren nutzen Vorwissen aus früheren Schätzresultaten.

8.2 Stand der Forschung

Die Anfänge des Straßen- und Wegetrackings gehen zurück bis in die frühen Achtzigerjahre – einen guten Überblick gibt hier [Mysliwetz, 1990]. Während frühe Ansätze der Straßenerkennung gleichermaßen für Feldwege funktionierten, werden dafür heute allerdings meist abweichende Messverfahren benötigt. Mit fortschreitender Technologie insbesondere bei digitalen Kameras und damit steigender Bildauflösung, Farbauflösung und allgemein Bildqualität stiegen auch die Unterschiede in der Wahrnehmung von Straßen und Feldwegen. Straßen werden nun vorwiegend über ihre weißen oder gelben Spurmarkierungen erkannt. Einige Ansätze hierfür werden in [Manz, 2013] genannt. Einen Überblick über die seltener erforschte Wahrnehmung von Feldwegen soll dagegen dieser Abschnitt bieten.

Erkennung von Wüstenpfaden Im Jahr 2004 rückte die Erkennung von unbefestigten Wegen erstmals in den Vordergrund: Bei den DARPA Grand Challenges 2004 und 2005 traten unbemannte Roboterfahrzeuge zum Wettbewerb in der Wüste an, wobei diese autonom einem vorgegebenen Pfad folgen sollten. Für die DARPA Grand Challenge 2004 entwickelten Rasmussen und Korah [2005] für das Fahrzeug *Bob* eine Methode zur Erkennung eines durch Reifenspuren definierten Wegs im Kamerabild. Dabei wird anhand der vorherrschenden Kantenorientierungen im Bild der Fluchtpunkt des Wegs bestimmt. Darauf basierend werden alle Linien im Bild zu diesem Fluchtpunkt extrahiert, deren Richtung den dominanten Kantenorientierungen entlang der Linie entspricht. Daraus wird eine zu verfolgende Mittellinie im Bild bestimmt. Für die DARPA Grand Challenge 2005 wurde dieses Verfahren von Rasmussen [2006] für das Fahrzeug *Alice* weiterentwickelt. Dabei wird anhand der im Kamerabild bestimmten Wegmitte im LiDAR-basierten Hindernis-Grid ein hindernisfreier Korridor in die entsprechende Richtung extrahiert. Auch bei dem Sieger-Fahrzeug *Stanley* [Thrun et al., 2006] wurde ein gerader Korridor aus dem LiDAR-basierten Hindernis-Grid extrahiert. Unter der Annahme, dass die Wegränder parallel zum vorgegebenen Pfad verlaufen, wird mittels eines Kalman-Filters der Versatz der Wegränder zur Sollspur geschätzt. Als Messung werden dabei aus dem Hindernis-Grid die äußersten, zur Sollspur parallelen Linienstücke extrahiert, die kein Hindernis enthalten. Viele Teams, darunter beispielsweise die Fahrzeuge *Sandstorm* und *H1ghlander* [Urmson et al., 2007], nutzten auch ausschließlich das Umgebungsmodell aus (statischen) Hindernissen und Terrain, um hindernisfreie, ebene Pfade zu extrahieren, ohne dediziert Wegränder zu erkennen. Das Fahrzeug *RASCAL* [Behringer et al., 2005] verfügte dagegen über eine Kantenerkennung im Kamerabild zur Identifikation der Wegränder. Auf dem Fahrzeug *CIMAR* [Crane III et al., 2006] erfolgte eine Klassifikation des Kamerabilds in befahrbare Bereiche und Hintergrund. Hierfür werden der Mittelwert und die Standardabweichung der RGB-Farbwerte mit vorgegebenen Referenz-Bildfenstern verglichen. Für das Fahrzeug *TerraMax* clusterten Caraffi et al. [2007] schließlich das Bild anhand der Farbunterschiede benachbarter Pixel im HLS-Farbraum (Farbton, Helligkeit, Sättigung, engl. *Hue, Lightness, Saturation*). Anhand der Homogenität, der Form und der Größe eines Clusters wird anschließend entschieden, ob ein Cluster Teil des Wegs ist.

Feldwegtracking Im Anschluss an die DARPA Challenges wurden die ersten speziell für Feldwege ausgelegten Verfahren zur Schätzung eines geometrischen Wegmodells veröffentlicht: Franke et al. [2007] greifen das bis dahin für das Tracking von markierten Straßen verwendete Prinzip auf, mittels eines Partikelfilters eine die Fahrbahn beschreibende Klothoide relativ zum Fahrzeug zu schätzen. Der Zustandsvektor beinhaltet die laterale Verschiebung und den Gierwinkel der Klothoide zum Fahrzeug, Anfangskrümmung, Krümmungsänderung und Vorausschauweite der Klothoide und die Wegbreite. Für die Bestimmung der Likelihood der Partikel führen sie neue Kriterien für Feldwege ein: Herangezogen werden die Intensität und der Farbton (engl. *Hue*) aus dem HSI-Farbraum (Farbton, Sättigung, Lichtintensität, engl. *Hue, Saturation, Intensity*) eines Pixels, die Textur sowie der Abstand zu Kanten gewichtet mit ihrer Stärke und Richtung. Manz et al. [2010] schätzen ebenfalls mittels eines Partikelfilters Krümmung, Krümmungsänderung und Breite einzelner Klothoidenstücke in fester Entfernung zum Fahrzeug. Die Likelihood der Partikel wird invers zur Sättigung im HSI-Farbraum sowie anhand der Stärke und Richtung von Kanten im RGB-Bild bewertet. Ein Erweitertes Kalman-Filter (EKF) schätzt schließlich aus den einzelnen Klothoidensegmenten eine kontinuierliche Klothoide relativ zum Fahrzeug. Um im selben Framework auch Kreuzungen von Feldwegen erkennen zu können, evaluieren Manz et al. [2011] in ihrer Weiterentwicklung die Partikel der Klothoidenstücke anhand von Messungen in einem eingefärbten Terrain-Grid. Neben der Sättigung im HSI-Farbraum wird nun der Unterschied in der Farbe im YUV-Farbmodell und im RGB-Farbraum zu dem Bereich unter dem Fahrzeug bewertet. Zur Fahrzeugregelung wird durch die getrackten Klothoidensegmente schließlich ein kubischer Spline gefittet. Bayerl et al. [2015] bauen zuletzt auf diesem Ansatz auf. Mittels eines Partikelfilters wird eine Klothoide geschätzt, die einen Weg fester Breite modelliert. Ebenfalls das eingefärbte Terrain-Grid nutzend, wird die Likelihood der Partikel zusätzlich zu den Features von Manz et al. [2011] anhand der Kantenstärke und invers zum Anteil des Grünkanals an der Intensität bewertet. Außerdem wird weitere Sensorik integriert: In der Akkumulation eines Wärmebilds einer Infrarot-Kamera im Grid wird die Kantenstärke sowie der Temperaturunterschied zu dem Bereich unter dem Fahrzeug bewertet. Weiterhin berücksichtigt die Likelihood aus dem Umgebungsmodell die Hindernisfreiheit, den Höhenunterschied zum Bereich unter dem Fahrzeug sowie die räumliche Frequenz des Entfernungssignals der einzelnen Dioden eines LiDAR-Sensors.

Messverfahren im Kamerabild Neben diesen vollständigen Algorithmen zum Wegetracking wurden auch zahlreiche Arbeiten nur über Messverfahren zur Erkennung von Feldwegen in Kamerabildern publiziert: Rasmussen [2002] setzt alle $10\,\mathrm{px} \times 10\,\mathrm{px}$ einen Stützpunkt. Für jeden Stützpunkt wird aus Farb-Histogrammen im RGB-Farbraum und der Stärke und Richtung der Textur verschieden großer Bildausschnitte um diesen Stützpunkt herum ein Featurevektor berechnet. Anhand dieses Featurevektors klassifiziert ein neuronales Netz Stützpunkte als Weg oder Nicht-Weg. Chaturvedi et al. [2004] klassifizieren befahrbare Bereiche anhand des mittleren Farbtons eines $4\,\mathrm{px} \times 4\,\mathrm{px}$ großen Bildausschnitts im HSI-Farbraum. Fernandez und Price [2005] segmentieren einen Feldweg, indem für ein Rechteck am unteren Rand des Bilds, von dem angenommen wird, dass es auf dem Weg liegt, die Mittelwerte

und Standardabweichungen des Farbtons, der Sättigung und der Intensität aus dem HSI-Farbraum berechnet werden. Anhand dieser Statistik werden die restlichen Pixel des Bilds als Weg oder Nicht-Weg klassifiziert. Tan et al. [2006] nehmen ebenfalls ein stets auf dem Weg liegendes Rechteck am unteren Bildrand an und klassifizieren die Bildpixel anhand von Histogrammen des mit der Intensität normalisierten Rot- und Grünkanals in diesem Rechteck. Song et al. [2006] nutzen den $c_1c_2c_3$-Farbraum von Gevers und Smeulders [1999], um in der Wüste sandige Wege von Vegetation zu unterscheiden. Lewis et al. [2012] nutzen ebenfalls den $c_1c_2c_3$-Farbraum: Eine *Support Vector Machine* (SVM) klassifiziert Bildausschnitte anhand bekannter Deskriptoren wie des *Histogram of Oriented Gradients* (HOG) für Kantenorientierungen und Lokale binäre Muster, engl. *Local Binary Pattern* (LBP) für die Textur sowie Farbhistogrammen auf dem RGB-Farbraum und dem $c_1c_2c_3$-Farbraum. Die klassifizierten Bildausschnitte werden mittels eines SIFT-Deskriptors (*Scale-Invariant Feature Transform*) zwischen den Bildern unterschiedlicher Zeitschritte getrackt. Fleischmann et al. [2016] segmentieren Bodenpixel anhand einer Stereo-Punktwolke. Diese Bodenpixel werden zur Klassifikation in verschiedenen Farbmodellen mit einem Rechteck am unteren Bildrand verglichen. Dafür werden der a^*- und der b^*-Kanal des CIELAB-Farbraums, die Kanäle des $c_1c_2c_3$-Farbraums, die Sättigung aus dem HSI-Farbraum sowie ein Texturfeature berücksichtigt. Wang und Park [2021] und Park et al. [2022] trainieren schließlich ein Neuronales Faltungsnetz, engl. *Convolutional Neural Network* (CNN) zur Segmentierung von Wegen in Kamerabildern.

Messverfahren in der LiDAR-Punktwolke Auch zur Feldwegerkennung in LiDAR-Punktwolken wurden bereits verschiedene Methoden vorgestellt: So beinhaltet der Featurevektor der bereits erwähnten Arbeit von Rasmussen [2002] neben den Farbinformationen auch Mittelwert und Standardabweichung von LiDAR-Punkten, deren Projektion in die unterschiedlich großen Bildausschnitte um die Stützpunkte herum fällt. Der gesamte Featurevektor dient als Input für ein CNN. Han et al. [2012] extrahieren aus der LiDAR-Punktwolke in Polarkoordinaten Linien-Segmente, die anhand ihrer Länge sowie ihrer Roll- und Nickwinkel entweder als Straße bzw. Weg oder als Hindernis klassifiziert werden. Ort et al. [2018] suchen in den einzelnen Ringen der LiDAR-Punktwolke Sprünge im Rauschen der Entfernung der Punkte zum Fahrzeug. Davon ausgehend, dass der Weg von ebenerer Beschaffenheit ist als seine Umgebung, weisen die Punkte auf dem Weg weniger Varianz in der Entfernungsmessung auf, wodurch an den Straßenrändern Sprünge in der ersten Ableitung der Entfernung entlang eines Rings zu finden sind. Mittels RANSAC (*Random Sample Consensus*) werden schließlich Splines in die auf diese Art und Weise erkannten Fahrbahnrandpunkte gefittet. Auch Burger et al. [2019] nehmen an, dass die Textur der Straße glatter als die der Umgebung ist. Anhand des Winkels zwischen horizontal und vertikal benachbarten Punkten werden ebene Flächen als Bodenflächenpunkte segmentiert und geclustert. Auf diesen Bodenflächenclustern basiert das in dieser Arbeit vorgestellte LiDAR-Messverfahren, das wahrscheinliche Wegbereiche bestimmt. Burger et al. [2019] nutzen das Clustering, um die Wegränder als quadratische Kurven durch die linken und rechten Randpunkte zu bestimmen. Yin und Zhu [2022] segmentieren Wege ebenfalls anhand ihrer Glattheit. Dafür wird die LiDAR-Punktwolke in zwei Grids unterschiedlicher Auflösung rasterisiert.

In jedem Grid werden dann die Höhenunterschiede der einzelnen LiDAR-Punkte innerhalb einer Grid-Zelle sowie der minimalen und maximalen Höhen benachbarter Grid-Zellen untersucht. Dabei werden – das inverse Problem lösend – Grid-Zellen mit großen Höhenunterschieden als Nicht-Weg klassifiziert. Anschließend wird nach den Fahrbahnrandpunkten gesucht. Park et al. [2022] trainieren schließlich ein CNN zur Segmentierung von Wegen in LiDAR-Punktwolken. Die Ergebnisse werden mit der semantischen Segmentierung eines Kamerabilds fusioniert.

8.3 Modellierung und Tracking

In diesem Abschnitt wird das entwickelte, rekursive Schätzverfahren zum Straßen- und Wegetracking beschrieben, für das in den nächsten Abschnitten robuste Messverfahren in Kamera und LiDAR erarbeitet werden. Das grundlegende Schätzverfahren wurde durch Kallwies [2023] konzipiert, wird hier jedoch für ein robusteres Tracking von unbefestigten Straßen und Feldwegen um die Schätzung einer entlang des Wegs variablen Straßenbreite und einer vertikalen Krümmung erweitert. Zudem wird die Vorausschauweite dynamisch angepasst.

8.3.1 Modellierung von Straßen und Wegen durch Fahrspurrandpunkte

Um im Gegensatz zu den bis dahin veröffentlichten Verfahren zum Wegetracking auch mehrspurige Straßen modellieren zu können, wird die Geometrie eines Wegs und seiner Fahrspuren durch die Lage der einzelnen Spurbegrenzungen bzw. Fahrspurränder definiert. Eine Spurbegrenzung k wird dabei durch einzelne, dreidimensionale Spurbegrenzungspunkte $\mathbf{x}_i^{(k)}$ modelliert. Diese werden durch Klothoidenstücke verbunden. Abbildung 8.3 visualisiert dies an einem Beispiel. Die Posen dieser Spurbegrenzungspunkte im stationären Koordinatensystem der Odometrie werden in einer rekursiven Zustandsschätzung geschätzt. Um den Zustandsvektor zu vereinfachen, enthält dieser jedoch nicht für alle Spurbegrenzungspunkte die vollständige Pose. Stattdessen wird eine der Spurbegrenzungen als Bezugsspurbegrenzung bestimmt. Diese besteht aus Bezugsspurbegrenzungspunkten $\mathbf{x}_i^{(0)}$ in einem konstanten Abstand l. Davon ausgehend werden Spurbegrenzungspunkte auf den anderen Spurbegrenzungen durch ihren orthogonalen Abstand zu den Bezugsspurbegrenzungspunkten definiert. Eine Straße mit n Fahrspuren und einer Bezugsspurbegrenzung der Länge $m \cdot l$ (in Metern) wird damit beschrieben durch ihre Bezugsspurbegrenzungspunkte $\mathbf{x}_i^{(0)}$ mit $i \in \{0, \dots, m\}$ und die dazugehörigen senkrechten Ablagen $s_i^{(k)}$ zu den übrigen Spurbegrenzungspunkten $\mathbf{x}_i^{(k)}$. Wird die rechteste Spurbegrenzung als Bezugsspurbegrenzung gewählt, ist $k \in \{1, \dots, n\}$. Ansonsten gilt: $k \in \{k_{\mathsf{min}}, \dots, k_{\mathsf{max}}\} \setminus 0$.

Durch die Schätzung einer variablen Ablage zwischen den Spurbegrenzungen können auch Feldwege modelliert werden, deren Breite sich abrupt ändert. Um ferner nicht nur flache Straßen, sondern auch Feldwege mit Steigungen modellieren zu können, werden die einzelnen Spurbegrenzungspunkte nicht nur wie in [Franke et al., 2007, Manz et al., 2010, 2011, Bayerl et al., 2015] mit horizontalen Klothoiden verbunden, sondern auch

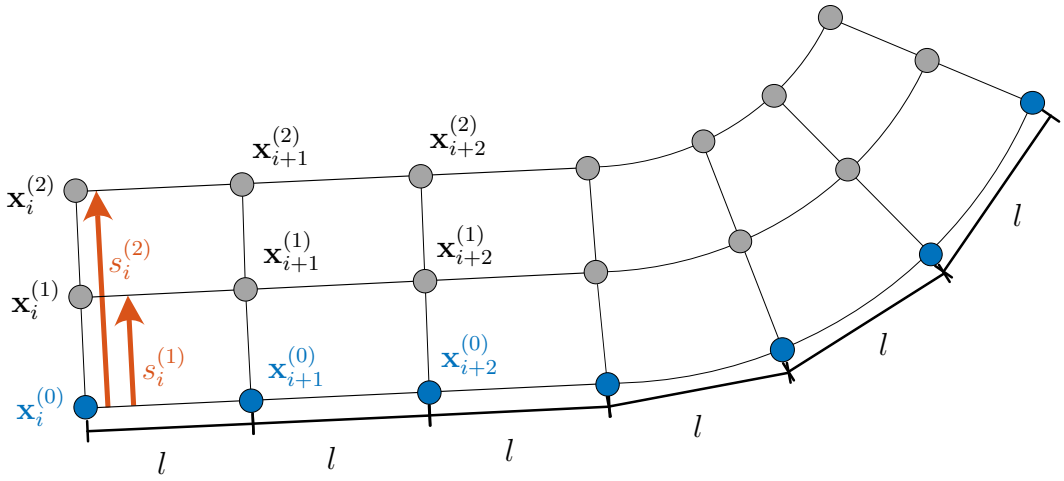


Abbildung 8.3:
Schematisches Beispiel der Modellierung einer zweispurigen Straße. Die Straße wird durch ihre Spurbegrenzungen definiert. Diese werden durch einzelne Spurbegrenzungspunkte modelliert, die durch Klothoidenstücke verbunden sind. Eine Gruppe lateral benachbarter Spurbegrenzungspunkte $\mathbf{x}_i = \{\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}\}$ wird durch den Bezugsspurbegrenzungspunkt $\mathbf{x}_i^{(0)}$ (blau) und die beiden senkrechten Verschiebungen $s_i^{(1)}$ und $s_i^{(2)}$ (rot) der beiden anderen Spurbegrenzungen zu dieser Bezugsspurbegrenzung beschrieben. Die Punkte auf der Bezugsspurbegrenzung haben einen konstanten longitudinalen Abstand l.

vertikal durch Klothoiden verknüpft. Durch die Schätzung einer vertikalen Krümmung kann das Wegetracking dadurch von den Informationen aus der Terrain-Schätzung über die Höhe und die Normalen des Untergrunds profitieren.

Insgesamt ergibt sich damit für jeden Bezugsspurbegrenzungspunkt der zu schätzende Zustandsvektor

$$\mathbf{x}_i = \begin{bmatrix} x & y & z & q_\mathsf{x} & q_\mathsf{y} & q_\mathsf{z} & q_\mathsf{w} & c_\mathsf{h} & c_\mathsf{v} & \mathbf{s} \end{bmatrix}^\mathsf{T} , \tag{8.1}$$

wobei ${}^{\text{odom}}\mathbf{p}_i^{(0)} = [x \ y \ z]^\mathsf{T}$ die Position des Bezugsspurbegrenzungspunkts $\mathbf{x}_i^{(0)}$ im ortsfesten Odometriekoordinatensystem angibt, ${}^{\text{odom}}\mathbf{q}_i^{(0)} = [q_\mathsf{x} \ q_\mathsf{y} \ q_\mathsf{z} \ q_\mathsf{w}]^\mathsf{T}$ seine Orientierung als Quaternion, $c_{\mathsf{h},i}$ die horizontale Krümmung zum nächsten Bezugsspurbegrenzungspunkt, $c_{\mathsf{v},i}$ analog die vertikale Krümmung und $\mathbf{s}_i$ den Versatz der lateral benachbarten Spurbegrenzungspunkte zum Bezugsspurbegrenzungspunkt (siehe Abbildung 8.3). Die Krümmungsänderung der verbindenden Klothoiden ergibt sich aus der Differenz der Krümmungen konsekutiver Spurbegrenzungspunkte.

Die Fahrspurränder sind nicht auf einen rechten, einen linken und einen optionalen mittleren Fahrspurrand begrenzt, sondern es können beliebig viele Fahrspurränder auch im laufenden Betrieb erzeugt und gelöscht werden. Dabei kann eine beliebige Spurbegrenzung als Bezugsspurbegrenzung gewählt werden. Der longitudinale Abstand l der Bezugsspurbegrenzungspunkte wird je nach Anforderung der Fahrzeugregelung auf $l = 1\,\text{m}$ oder $l = 2\,\text{m}$ gesetzt. Die Abstände der Punkte anderer Spurbegrenzungen vergrößern bzw. verkleinern sich entsprechend in Abhängigkeit von der Krümmung der Straße.

8.3.2 Straßen- und Wegetracking durch Optimierung

Mittels rekursiver Zustandsschätzung soll nun der Zustandsvektor $\mathbf{x}$ aller Gruppen von Spurbegrenzungspunkten $\mathbf{x}_i$ vor dem Fahrzeug geschätzt werden. In den Verfahren von Franke et al. [2007], Manz et al. [2010] und Bayerl et al. [2015] kommt zur Zustandsschätzung ein Partikelfilter zum Einsatz. Für ein Partikelfilter ist das vorliegende Schätzproblem jedoch zu hochdimensional: Bei einer zweispurigen Straße mit 40 m Vorausschau müssen bei einem Bezugspunktabstand von $l = 2\,\text{m}$ in jedem Zeitschritt $21 \cdot 11 = 231$ Zustandsparameter geschätzt werden. Damit müssten zu viele Partikel gestreut und bewertet werden, als dass eine Echtzeitanwendung möglich wäre. Gleichzeitig übersteigen die entwickelten Messverfahren auch die Möglichkeiten des effizienteren Kalman-Filters: Da für die einzelnen Spurbegrenzungspunkte nicht nur direkt auf den Zustandsvektor abbildbare Messungen berücksichtigt werden, sondern aus der LiDAR-Punktwolke und dem Hindernis-Grid auch wahrscheinlichere und – hochgradig multimodal – unwahrscheinlichere Bereiche für einen Weg extrahiert werden, lässt sich keine Messfunktion zur Berechnung einer erwarteten Messung aufstellen. Daher erfolgt die Zustandsschätzung stattdessen durch die Lösung des MAP-Problems (Maximum-a-posteriori)

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_i \Big[\mathbf{c}_{\text{meas.}}(\mathbf{x}_i) + \mathbf{c}_{\text{model}}(\mathbf{x}_i) \Big] \tag{8.2}$$

mittels nichtlinearer Optimierung. Dieses MAP-Problem minimiert zum einen die Kosten $\mathbf{c}_{\text{meas.}}$, die die Abweichungen der geschätzten Spurbegrenzungspunkte zu den Messungen evaluieren und die wahrscheinlichen und unwahrscheinlichen Wegbereiche einbeziehen. Zum anderen werden Kosten $\mathbf{c}_{\text{model}}$ minimiert, die die Abweichung der geschätzten Spurbegrenzungspunkte zum vorgegebenen Straßenmodell beschreiben. Zur besseren Lesbarkeit wird in diesem Kapitel auf die Verwendung eines Zeitindex verzichtet. Das Ergebnis der Optimierung des letzten Zeitschritts ist jedoch stets der Ausgangspunkt für die Schätzung im nächsten Zeitschritt. Dabei wird die Schätzung rekursiv durch die Berücksichtigung neuer Messungen verbessert.

Mess-Kosten Die Mess-Kosten $\mathbf{c}_{\text{meas.}}$ setzen sich aus Kosten durch Messverfahren im Kamerabild und Kosten durch 3D-Informationen aus der LiDAR-Punktwolke zusammen:

$$\mathbf{c}_{\text{meas.}}(\mathbf{x}_i) = \mathbf{c}_{\text{img}}(\mathbf{x}_i) + \mathbf{c}_{\text{3D}}(\mathbf{x}_i)\ . \tag{8.3}$$

Diese Messverfahren und die zugehörigen Residuen werden in den Abschnitten 8.4 und 8.5 beschrieben.

Modell-Kosten Die Modell-Kosten $\mathbf{c}_{\text{model}}$ setzen sich zusammen aus einem Residuum $\mathbf{r}_{\text{cloth.}}$ zur Konsistenz der geschätzten Fahrspurrandpunkte zum Klothoidenmodell, dem Residuum $\mathbf{r}_{\text{smooth}}$ zur Glattheit der Krümmungen und der Spurbreite entlang des Wegs sowie den Kosten $\mathbf{c}_{\text{limit}}$ zur Begrenzung der möglichen Krümmungen und der Spurbreite:

$$\mathbf{c}_{\text{model}}(\mathbf{x}_i) = \left(\mathbf{r}_{\text{cloth.}}(\mathbf{x}_i)\right)^2 + \left(\mathbf{r}_{\text{smooth}}(\mathbf{x}_i)\right)^2 + \mathbf{c}_{\text{limit}}(\mathbf{x}_i)\ . \tag{8.4}$$

Der Residuenvektor $\mathbf{r}_{\text{cloth.}}$ minimiert die Abweichung der Position und der Orientierung $(\mathbf{p}_i^{(0)}, \mathbf{q}_i^{(0)})$ eines Bezugsspurbegrenzungspunkts zu seiner aus den Krümmungen des vorhergehenden Bezugsspurbegrenzungspunkts prädizierten Pose $(\mathbf{p}_i^{*(0)}, \mathbf{q}_i^{*(0)})$:

$$\mathbf{r}_{\text{cloth.},[1,2,3]}(\mathbf{x}_i) = \left({}^{\text{odom}}\mathbf{p}_i^{*(0)} - {}^{\text{odom}}\mathbf{p}_i^{(0)}\right) \cdot \mathbf{w}_{[\text{cloth}_{\text{pos.}}]}\,, \tag{8.5}$$

$$\mathbf{r}_{\text{cloth.},[4,5,6]}(\mathbf{x}_i) = \text{euler}\left({}^{\text{odom}}\mathbf{q}_i^{*(0)} \cdot {}^{\text{odom}}\mathbf{q}_i^{(0)^{-1}}\right) \cdot \mathbf{w}_{[\text{cloth}_{\text{orient.}}]}\,. \tag{8.6}$$

Die Funktion $\text{euler}(\cdot)$ notiert dabei die Umrechnung der Quaternion in Euler-Winkel. Der Vektor $\mathbf{w}$ enthält Gewichtungsfaktoren, die die einzelnen Residuen des MAP-Problems unterschiedlich stark gewichten. Die Prädiktion eines Bezugsspurbegrenzungspunkts erfolgt anhand des Klothoidenmodells: Zwei longitudinal konsekutive Spurbegrenzungspunkte $\mathbf{x}_{i-1}^{(0)}$ und $\mathbf{x}_i^{(0)}$ werden durch die Klothoide verbunden, die durch die geschätzte Anfangskrümmung c_{i-1} und die lineare Krümmungsänderung

$$\mathring{c}_{i-1} = \frac{c_i - c_{i-1}}{l} \tag{8.7}$$

definiert ist. Aufgrund der geringen Krümmungen kann für die Berechnung dieser Klothoide die Näherung

$$\text{f}(l) = \frac{1}{2}cl^2 + \frac{1}{6}\mathring{c}l^3 \tag{8.8}$$

herangezogen werden. Die prädizierte Pose eines Bezugsspurrandpunkts $\mathbf{x}_i^{(0)}$ berechnet sich damit aus der geschätzten Pose $(\mathbf{p}_{i-1}^{(0)}, \mathbf{q}_{i-1}^{(0)})$ und den geschätzten Krümmungen $c_{\text{h},i-1}$ bzw. $c_{\text{v},i-1}$ des vorherigen Bezugsspurrandpunkts $\mathbf{x}_{i-1}^{(0)}$ als:

$${}^{\text{odom}}\mathbf{p}_i^{*(0)} = {}^{\text{odom}}\mathbf{p}_{i-1}^{(0)} + \mathbf{R}\left({}^{\text{odom}}\mathbf{q}_{i-1}^{(0)}\right) \cdot \Delta\mathbf{p}_{i-1\to i}^{*}\,, \tag{8.9}$$

$${}^{\text{odom}}\mathbf{q}_i^{*(0)} = {}^{\text{odom}}\mathbf{q}_{i-1}^{(0)} \cdot \Delta\mathbf{q}_{i-1\to i}^{*}\,, \tag{8.10}$$

wobei $\mathbf{R}(\mathbf{q})$ die der Quaternion $\mathbf{q}$ entsprechende Rotationsmatrix darstellt und

$$\Delta\mathbf{p}_{i-1\to i}^{*} = \begin{bmatrix} l \\ \frac{1}{2}c_{\text{h},i-1}l^2 + \frac{1}{6}\mathring{c}_{\text{h},i-1}l^3 \\ \frac{1}{2}c_{\text{v},i-1}l^2 + \frac{1}{6}\mathring{c}_{\text{v},i-1}l^3 \end{bmatrix}\,, \tag{8.11}$$

$$\Delta\mathbf{q}_{i-1\to i}^{*} = \text{quat}\left(\begin{bmatrix} \Psi = c_{\text{h},i-1}l + \frac{1}{2}\mathring{c}_{\text{h},i-1}l^2 \\ \Theta = c_{\text{v},i-1}l + \frac{1}{2}\mathring{c}_{\text{v},i-1}l^2 \\ \Phi = 0 \end{bmatrix}\right)\,. \tag{8.12}$$

Die Funktion $\text{quat}(\cdot)$ notiert die Umrechnung von Euler-Winkeln in eine Quaternion. Durch die Minimierung der Abweichung der geschätzten Spurbegrenzungspunkte zu ihrer Prädiktion anhand der Klothoide folgt die Schätzung dem Klothoidenmodell.

Der Residuenvektor $\mathbf{r}_{\text{smooth}}$ minimiert Sprünge in der Krümmung und den Fahrspurbreiten $w_i^{(k)}$ von einem Bezugsspurbegrenzungspunkt zum longitudinal nächsten:

$$\mathbf{r}_{\text{smooth},[1]}(\mathbf{x}_i) = (c_{\text{h},i} - c_{\text{h},i+1}) \cdot \mathbf{w}_{\left[\text{smooth}_{c_\text{h}}\right]}\,, \tag{8.13}$$

$$\mathbf{r}_{\text{smooth},[2]}(\mathbf{x}_i) = (c_{\text{v},i} - c_{\text{v},i+1}) \cdot \mathbf{w}_{[\text{smooth}_{c_\text{v}}]}\,, \tag{8.14}$$

$$\mathbf{r}_{\text{smooth},[3]}(\mathbf{x}_i) = \sum_k \left(w_i^{(k)} - w_{i+1}^{(k)} \right) \cdot \mathbf{w}_{[\text{smooth}_w]} \,, \tag{8.15}$$

wobei

$$w_i^{(k)} = s_i^{(k)} - s_i^{(k-1)} \,. \tag{8.16}$$

Dadurch wird ein glatter Verlauf des geschätzten Wegs gewährleistet.

Durch die Kosten $\mathbf{c}_{\text{limit}}$ werden zuletzt die möglichen Krümmungen und Spurbreiten limitiert. Um diese Limits bei starker anderweitiger Indikation auch überschreiten zu können, werden dem Optimierer keine harten Grenzen gesetzt, sondern diese Limits ebenfalls durch Residuen umgesetzt:

$$\mathbf{c}_{\text{limit}}(\mathbf{x}_i) = \sum_k \left(\mathbf{r}^{(k)}_{\text{limit}_{c_\text{h}}}(\mathbf{x}_i) \right)^2 + \sum_k \left(\mathbf{r}^{(k)}_{\text{limit}_{c_\text{v}}}(\mathbf{x}_i) \right)^2 + \sum_k \left(\mathbf{r}^{(k)}_{\text{limit}_w}(\mathbf{x}_i) \right)^2 \,. \tag{8.17}$$

Dafür wird für jeden Spurbegrenzungspunkt $\mathbf{x}_i^{(k)}$ für jeden der Parameter c_h, c_v und w die Kostenfunktion

$$r_{\text{limit}_p}(p) = \exp\left[\Big(|p - \overline{p}_{\text{limit}}| - \Delta p_{\text{limit}} \Big) \cdot \mathbf{w}_{[\text{limit}_p]} \right] \tag{8.18}$$

ausgewertet. Diese lässt die Kosten für einen Parameterwert p außerhalb eines vorgegebenen Bereichs $[\overline{p}_{\text{limit}} - \Delta p_{\text{limit}}; \overline{p}_{\text{limit}} + \Delta p_{\text{limit}}]$ sprunghaft ansteigen. Der erwünschte Bereich für die Spurbreite wird hier auf 2,5 m bis 5 m gesetzt. Die geschätzten Krümmungen werden auf maximal $-0{,}05\,\text{m}^{-1}$ bis $0{,}05\,\text{m}^{-1}$ limitiert. Dies entspricht einem maximalen Kurvenradius von 20 m.

Das durch die Residuen aufgestellte Optimierungsproblem wird schließlich wie bereits bei der Kamera-zu-LiDAR-Kalibrierung (Kapitel 3) und der Terrain-Schätzung (Kapitel 7) mittels des in der Bibliothek *Ceres Solver* [Agarwal et al., 2020] implementierten Levenberg-Marquardt-Algorithmus [Levenberg, 1944, Marquardt, 1963] gelöst. Mit Ausnahme von $\mathbf{w}_{[\text{smooth}_w]} = 100$ werden alle in diesem Abschnitt eingeführten Gewichtungsfaktoren empirisch auf 1000 gesetzt.

8.3.3 Initialisierung und Verlängerung von Straßen und Wegen

Zur Initialisierung der Fahrspurrandpunkte zu Beginn des Trackings hat es sich als ausreichend erwiesen, einen geraden Weg vor dem Fahrzeug anzunehmen. Mit den ersten Messungen wird der Zustand korrigiert. Um Falschmessungen zu vermeiden, wird dabei zunächst nur ein sehr kurzer Weg von 15 m initialisiert. Erst wenn die Schätzung der Spurbegrenzungspunkte mindestens einmal mit Messungen korrigiert wurde und der Bereich des geschätzten Wegs im Hindernis-Grid frei von Hindernissen ist, wird der geschätzte Weg nach vorne erweitert. Zur Verlängerung der ortsfesten, getrackten Wege werden neue Punkte an jeden Fahrspurrand angehängt, indem der Zustandsvektor um einen zusätzlichen Bezugsspurrandpunkt erweitert wird. Dazu wird ausgehend von $\mathbf{x}_i^{(0)}$ anhand der geschätzten Position und Orientierung sowie der geschätzten Krümmungen und Fahrspurbreiten entsprechend des Klothoidenmodells ein neuer Bezugsspurrandpunkt $\mathbf{x}_{i+1}^{(0)}$ prädiziert.

Wenn sich das Ego-Fahrzeug nach vorne bewegt, wird die Wegschätzung um einen neuen Bezugsspurbegrenzungspunkt verlängert, sobald eine minimale Vorausschauweite von 25 m auf Straßen bzw. 20 m auf Feldwegen unterschritten wird. Darüber wird der Weg bis zu einer maximalen Vorausschauweite von 50 m auf Straßen bzw. 40 m auf Feldwegen weiter verlängert, falls der vorderste Punkt einer der Fahrspurbegrenzungen bereits einmal durch eine Messung korrigiert wurde. Dadurch wird die Vorausschau dynamisch an die aktuelle Sicht und die daraus resultierende Beobachtbarkeit angepasst: Während in engen Kurven oder bei schlechten Lichtverhältnissen mit einer verkürzten Vorausschau gewährleistet wird, dass sich der Tracker nicht „verrennt“, erhöht sich die Vorausschau beispielsweise auf einer Geraden. Das erlaubt es, schneller zu fahren oder Überholmanöver zu planen. Entsprechend sollte die zu fahrende Geschwindigkeit abhängig von der Vorausschau gewählt werden. Ist nicht im Vorhinein festgelegt, ob eine Straße oder ein Feldweg getrackt werden soll, so werden die verkürzten Vorausschauweiten des Feldwegs verwendet. Um zu vermeiden, dass der Zustandsvektor unendlich wächst, werden Bezugsspurrandpunkte hinter dem eigenen Fahrzeug konstant gesetzt und sind nicht mehr Teil des zu optimierenden Zustandsvektors.

8.4 Messungen aus Kamerabildern

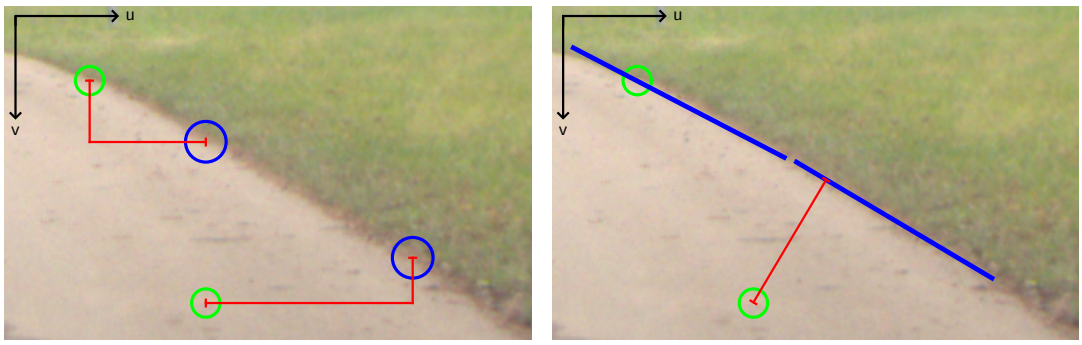
Für eine große Vorausschau wird die Position der Spurbegrenzungspunkte primär im Kamerabild gemessen. Dabei wird nach dem 4D-Ansatz von Dickmanns [2007] jeder Spurbegrenzungspunkt $\mathbf{x}_i^{(k)}$ einzeln gemessen, wobei das Vorwissen seiner letzten Zustandsschätzung berücksichtigt wird. Da ein Spurbegrenzungspunkt mit mehreren Messverfahren zu unterschiedlichen Zeitpunkten in verschiedenen Kamerabildern gemessen worden sein kann, können für jeden Spurbegrenzungspunkt mehrere Bildmessungen existieren. Für jede Bildmessung mit dem Index j von Punkt $\mathbf{x}_i^{(k)}$ ergibt sich ein Residuenvektor $\mathbf{r}_{\mathsf{img},j}^{(k)}(\mathbf{x}_i)$, der den Unterschied zwischen der erwarteten Bildposition (prädizierte Messung) und der tatsächlichen Messung darstellt. Unter Verwendung einer Verlustfunktion $\rho(\cdot)$ summieren sich die Kosten aus den Bildmessungen als:

$$\mathbf{c}_{\mathsf{img}}(\mathbf{x}_i) = \sum_k \sum_j \rho\Big(\mathbf{r}_{\mathsf{img},j}^{(k)}(\mathbf{x}_i)\Big) \ . \tag{8.19}$$

Dazu wird aus dem aktuellen Zustand $\mathbf{x}_i$ die 3D-Position ${}^{\mathsf{odom}}\mathbf{p}_i^{(k)}$ des Spurbegrenzungspunkts $\mathbf{x}_i^{(k)}$ in Odometriekoordinaten abgeleitet und anhand der Eigenbewegungsschätzung und der extrinsischen und intrinsischen Kalibrierung der Kamera cam_j in das Kamerabild projiziert. Daraus ergibt sich die prädizierte Messung

$${}^{\mathsf{img}_j}\mathbf{y}_i^{*(k)} = {}^{\mathsf{img}_j}\,\mathrm{proj}\Big({}^{\mathsf{cam}_j}\mathbf{H}_{\mathsf{ego}} \cdot {}^{\mathsf{ego}}\mathbf{H}_{\mathsf{odom}} \cdot {}^{\mathsf{odom}}\mathbf{p}_i^{(k)}\Big) = \begin{pmatrix} u^* \\ v^* \end{pmatrix} \tag{8.20}$$

für die Bildposition von Fahrspurrandpunkt $\mathbf{x}_i^{(k)}$ in Kamerabild img_j, welche mit dem Ergebnis der tatsächlichen Messung verglichen werden kann. Dafür sind abhängig von der Art der Messung zwei Kostenfunktionen möglich:



(a) Punktmessung (b) Linienmessung

Abbildung 8.4:
Visualisierung der möglichen Residuen für Bildmessungen. In grün werden die prädizierten Punkte eines Fahrspurrands dargestellt, in blau die gemessenen Punkte bzw. Linien. Die roten Linien visualisieren die entsprechend zu minimierenden Residuen. Bei den Punktmessungen entstehen für den oberen Punkt Kosten größer null, obwohl er perfekt auf dem Wegrand liegt.

Punktmessung Wird durch das Messverfahren für jeden Spurbegrenzungspunkt ein konkreter Punkt im Bild gemessen, so könnten im Residuum

$$\mathbf{r}^{(k)}_{\text{img},j}(\mathbf{x}_i) = {}^{\text{img}_j}\mathbf{y}_i^{*(k)} - {}^{\text{img}_j}\mathbf{y}_i^{(k)} = \begin{pmatrix} u^* - u \\ v^* - v \end{pmatrix} \tag{8.21}$$

direkt die Abweichungen der u- und v-Koordinaten minimiert werden. In Abbildung 8.4a wird dies schematisch dargestellt.

Da es jedoch meist schwierig ist, im Kamerabild einen bestimmten Punkt eines kontinuierlichen Fahrspurrands zu identifizieren, können die Messverfahren die konkrete Position eines Fahrspurrandpunkts entlang des Fahrspurrands nicht messen. Willkürlich einen Punkt auf der gemessenen Kante auszuwählen kann jedoch zu irrtümlich hohen Residuen führen, wie das Beispiel in Abbildung 8.4 zeigt.

Linienmessung Daher soll stattdessen für jeden Spurbegrenzungspunkt eine Linie im Kamerabild gemessen werden. In der Kostenfunktion

$$\mathbf{r}^{(k)}_{\text{img},j}(\mathbf{x}_i) = \mathrm{d}\Big({}^{\text{img}_j}\mathbf{y}_i^{*(k)}, \mathrm{h}\Big({}^{\text{img}_j}\mathbf{y}^{(k)}_{i,[1]}, {}^{\text{img}_j}\mathbf{y}^{(k)}_{i,[2]}\Big)\Big) \tag{8.22}$$

wird dann der senkrechte Abstand $\mathrm{d}(\cdot)$ der prädizierten Punktmessung ${}^{\text{img}_j}\mathbf{y}_i^{*(k)}$ zu einer anhand von zwei Punkten gemessenen Linie $\mathrm{h}\Big({}^{\text{img}_j}\mathbf{y}^{(k)}_{i,[1]}, {}^{\text{img}_j}\mathbf{y}^{(k)}_{i,[2]}\Big)$ betrachtet.

Die resultierenden Residuen aller Messungen aller Spurbegrenzungspunkte werden in der Gesamt-Kostenfunktion aufsummiert. Dabei wird für eine erhöhte Robustheit gegen Ausreißer die Huber-Verlustfunktion $\rho_{\mathrm{H}}(\cdot)$ [Huber, 1964] eingesetzt (s. Abschnitt 2.5). Um die Rechenlast nicht immer weiter zu erhöhen, werden für jeden Fahrspurrandpunkt maximal die zehn neuesten Bildmessungen genutzt. Die Messungen dürfen dabei maximal 3 s alt sein.

8.4.1 CRONOS-Kantendetektion auf Straßen

Für die robuste Messung von Spurbegrenzungspunkten im Kamerabild wird eine lokale, gerichtete Kantendetektion nach dem CRONOS-Verfahren von Kuhnert [1988] bzw. Mysliwetz [1990] durchgeführt. CRONOS, detailliert beschrieben in [Dickmanns, 2007, Abschnitt 5.2], ist eine sehr effiziente Methode zur Detektion von Kanten bekannter Orientierung in lokalen Messfenstern. Dabei wird eine Prewitt-Faltungsmaske entsprechend des erwarteten Winkels der gesuchten Kante gedreht. Diese wird anschließend in einer Richtung über das Messfenster bewegt (vgl. Abbildung 8.5) und die Faltungsantwort ausgewertet. Damit müssen zum einen nur kleine Bildbereiche ausgewertet werden, zum anderen werden gezielt nur Kanten einer bestimmten Richtung extrahiert. Das Messverfahren liefert dadurch kaum falsch positive Messungen durch unkorrelierte Kanten anderer Richtungen (s. Abbildung 8.6c).

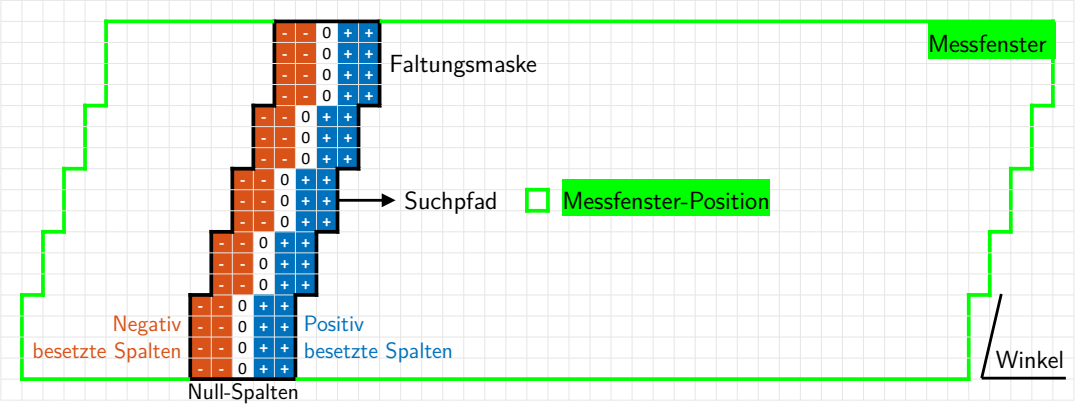
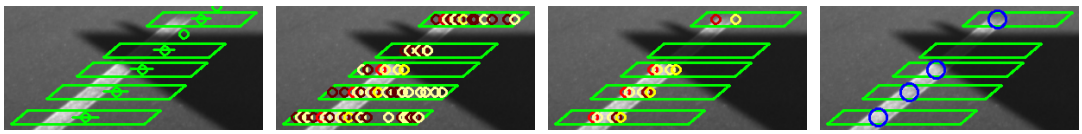


Abbildung 8.5:
Faltungsmaske der CRONOS-Kantendetektion in einem Messfenster. Die Abbildung ist inspiriert durch [Dickmanns, 2007, Figure 5.6] und [von Holt, 2004, Abbildung 7.2].

Zur Messung der Spurbegrenzungspunkte mittels CRONOS wird zunächst das RGB-Kamerabild in ein Graustufenbild umgewandelt. Die weiteren Schritte können an den Beispielen in den Abbildungen 8.6 und 8.7 verfolgt werden: Für jeden Spurbegrenzungspunkt wird ein eigenes Messfenster bestimmt, indem seine geschätzte Position in das Bild projiziert wird und zentriert um die prädizierte Bildposition ein CRONOS-Messfenster aufgesetzt wird. Da aufgrund der Perspektive weit von der Kamera entfernte Punkte im Bild sehr nah beieinander liegen, werden Messfenster, die mit dem Messfenster eines vorherigen Punkts überlappen, ignoriert, um redundante Messungen desselben Bildausschnitts zu vermeiden. Über die verbleibenden Messfenster (siehe Abbildung 8.6a) wird die entsprechend der prädizierten Orientierung des Spurbegrenzungspunkts rotierte CRONOS-Faltungsmaske geschoben. Die Auswertung der Faltungsantwort liefert die in Abbildung 8.6b dargestellten Kantendetektionen zurück. Daraus werden, wie es in Abbildung 8.6c zu sehen ist, mit einem festen Schwellwert Kanten mit hinreichend großer Kantenstärke ausgewählt. Aus diesen Kanten werden im letzten Schritt Messungen gewonnen. Dabei unterscheidet sich das Vorgehen für innere Fahrspurbegrenzungen markierter Straßen und äußere Fahrbahnränder von Straßen und Wegen.



(a) (b) (c) (d)

Abbildung 8.6:
Beispiel für die Anwendung von CRONOS für die Detektion einer Straßenmarkierung. (a) Zunächst werden die geschätzten 3D-Punkte ins Kamerabild projiziert (grüne Kreise) und Messfenster (grüne Parallelogramme) aufgespannt. Die grünen Linien in den projizierten Punkten zeigen die erwartete Breite der Spurmarkierung. (b) Die Faltung mit der CRONOS-Maske liefert sehr viele Detektionen. Rote Kreise zeigen Kanten von dunkel nach hell, gelbe Kreise Kanten von hell nach dunkel. Je stärker eine detektierte Kante ist, desto intensiver wird sie dargestellt. (c) Aus diesen Kanten werden mithilfe eines festen Schwellwerts die stärksten Kanten ausgesucht. Dank der rotierten Faltungsmaske liegen diese nur auf der gesuchten Spurmarkierung, nicht auf dem ebenso starken Schatten. (d) Anhand der besten Kombination einer Dunkel-Hell-Kante mit einer Hell-Dunkel-Kante wird die weiße Spurmarkierung detektiert und eine Messung extrahiert (blaue Kreise).

Innere Spurbegrenzungen Bei inneren Spurbegrenzungen auf mehrspurigen Straßen werden die Straßenmarkierungen gemessen. Dazu werden nach einer Idee aus [Kallwies, 2023] gezielt Paare aus Dunkel-Hell- und Hell-Dunkel-Kanten im richtigen Abstand gesucht. Da die Breite von Schmalstrichen in Deutschland auf 0,12 m standardisiert ist [FGSV, 1993], kann die erwartete Linienbreite im Kamerabild berechnet werden. Wenn der Abstand des besten Dunkel-Hell/Hell-Dunkel-Paars maximal 30 % von der berechneten Linienbreite abweicht, wird deren Mittelpunkt als Messung extrahiert. Das Ergebnis ist in Abbildung 8.6d visualisiert.

Äußere Fahrbahnränder Bei den äußeren Fahrbahnrändern ist die Suche nach weißen Markierungen nicht zielführend. Zum einen ist die äußere Markierung (vor allem von Nachbarspuren) in der Entfernung oft schlecht sichtbar (vgl. Abbildung 8.8), und Kies oder Schnee neben der Fahrbahn können zu falschen Detektionen führen. Zum anderen sollen auch lediglich durch einen Bordstein abgetrennte oder auch gänzlich unmarkierte Straßen und Wege getrackt werden können. Dafür hat es sich als ausreichend robust erwiesen, die innerste in einem Messfenster detektierte Kante (einer gewissen Stärke) als Messung des Straßenrands zu wählen. Dem liegt die Annahme zu Grunde, dass die Straße eine glatte Textur hat und auf der Straße keine starken Kanten gemessen werden. Das Ergebnis ist in Abbildung 8.7 visualisiert.

In beiden Fällen ergibt sich eine Punktmessung ${}^{\text{img}_j}\mathbf{y}_i^{(k)}$ der Bildspalte u und Bildzeile v eines Fahrspurrandpunkts $\mathbf{x}_i^{(k)}$ in einem Bild img_j.

8.4.2 Orientierung der Messfenster und Linienmessung

Zur Vereinfachung wurden in den Abbildungen 8.6 und 8.7 nur horizontale Messfenster aufgesetzt. Wie in Abbildung 8.8a zu sehen ist, können damit jedoch beispielsweise aufgrund einer Kurve beinahe horizontal verlaufende Fahrspurränder nur schlecht

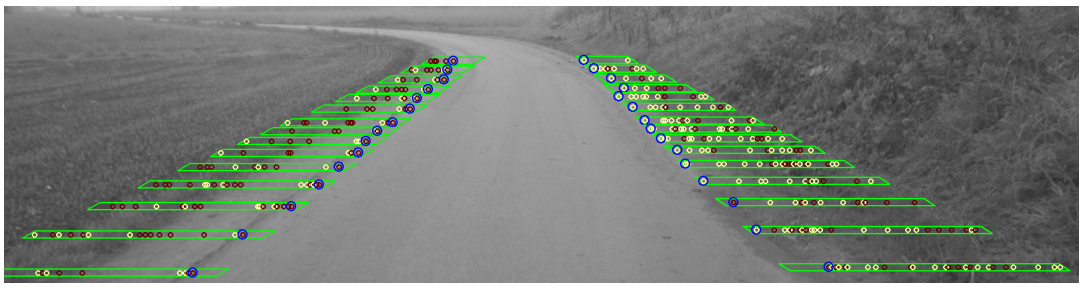
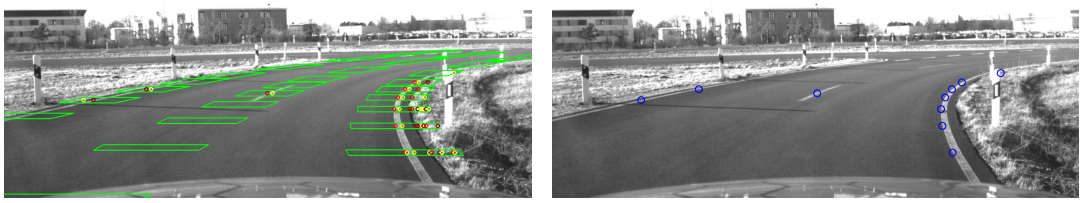


Abbildung 8.7:
Beispiel für die Anwendung von CRONOS für die Detektion eines unmarkierten Fahrbahnrands. Statt nach weißen Straßenmarkierungen zu suchen wird die innerste Kante gemessen. Die blauen Kreise markieren die als Messung gewählten Detektionen. Diese liegen korrekt auf dem Fahrbahnrand und nicht beispielsweise auf der stärkeren Kante zwischen dem an den Teer angrenzenden Kiesstreifen und dem Acker auf der linken Seite.

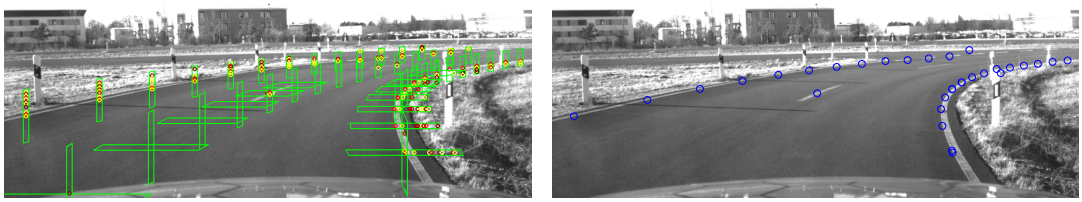
gemessen werden. Daher werden abhängig vom erwarteten Kantenwinkel im Bild horizontale und/oder vertikale Messfenster aufgesetzt. Dabei werden bei einem Kantenwinkel in einem Bereich von $\pm 15°$ um 90° nur horizontale bzw. in einem Bereich von $\pm 15°$ um $0°/180°$ nur vertikale Messfenster aufgesetzt, andernfalls werden Messfenster beider Ausrichtungen aufgesetzt. Ein Beispiel ist in Abbildung 8.8b zu sehen. Horizontale Messfenster haben eine feste Höhe von 17 px. Ihre Breite wird abhängig von der Entfernung so berechnet, dass ein Bereich von 1,5 m abgedeckt wird. Vertikale Messfenster haben eine Breite von 17 px und eine Höhe von 1 m.

Mit (maximal) einem CRONOS-Messfenster jeder Ausrichtung pro Fahrspurrandpunkt können Punkte gemessen werden. Ein einzelner Punkt auf einem Fahrspurrand kann jedoch nur schwer über verschiedene Zeitschritte und Kamerabilder hinweg eindeutig wiedererkannt werden. Daher ist es, wie zuvor beschrieben, naheliegender, Linien zu messen. Zu diesem Zweck wird, wie in Abbildung 8.8c gezeigt, für jeden Punkt ein zweites, gleich rotiertes, aber nach vorne versetztes Messfenster aufgesetzt. Im Falle einer inneren Fahrspurbegrenzung wird in diesem wiederum nach einer weißen Spurmarkierung gesucht. Beim Straßenrand wird je nach der Art der Kante, die im ersten Messfenster als innerste Messung extrahiert wurde, die innerste Hell-Dunkel- oder die innerste Dunkel-Hell-Kante gewählt. Wird auf diese Art und Weise in beiden Messfenstern eine erfolgreiche Messung durchgeführt, so bilden die zwei gemessenen Punkte ${}^{\text{img}_j}\mathbf{y}_{i,[1]}^{(k)}$ und ${}^{\text{img}_j}\mathbf{y}_{i,[2]}^{(k)}$ eine Linienmessung $\mathrm{h}\left({}^{\text{img}_j}\mathbf{y}_{i,[1]}^{(k)}, {}^{\text{img}_j}\mathbf{y}_{i,[2]}^{(k)}\right)$ des Fahrspurrandpunkts $\mathbf{x}_i^{(k)}$ im Bild img_j. Abbildung 8.8c zeigt ein Beispiel.

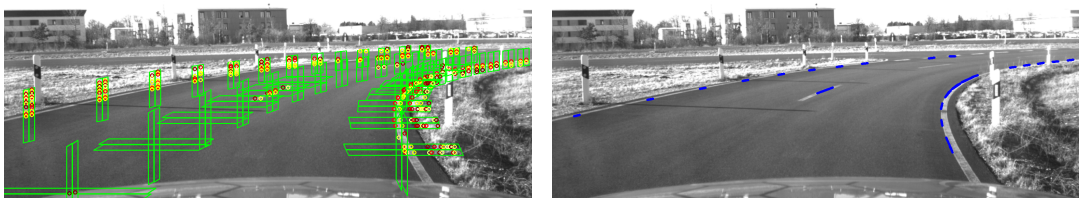
Die Messung einer Linie hat im Vergleich zur Punktmessung noch einen weiteren Vorteil: Durch die Messung mittels zweier Messfenster kann eine Vielzahl an Ausreißern ausgefiltert werden. Zum einen erscheinen Steine, Dreck, Pfützen oder Spiegelungen in den beiden Messfenstern oft nicht im selben Winkel und werden daher nicht von beiden um den gleichen Winkel rotierten Faltungsmasken als Kante erfasst. Zum anderen kann mithilfe des zweiten Messfensters aus den zwei detektierten Kantenpunkten zusätzlich der Winkel der Kante explizit gemessen werden. Dieser kann mit dem aus der Prädiktion der Fahrspurrandpunkte erwarteten Kantenwinkel verglichen werden. Weicht der gemessene Winkel mehr als 10° vom erwarteten Kantenwinkel



(a) CRONOS-Punktmessungen bei Verwendung ausschließlich horizontaler Messfenster



(b) CRONOS-Punktmessungen aus horizontalen und vertikalen Messfenstern



(c) CRONOS-Linienmessungen durch Verwendung eines zweiten Messfensters

Abbildung 8.8:
Veranschaulichung der Vorteile der Durchführung von mehr als einer CRONOS-Messung pro Fahrspurrandpunkt am Beispiel einer markierten Kurve. In (a) werden ausschließlich horizontale Messfenster aufgesetzt, in (b) dagegen sowohl horizontale als auch vertikale. Damit können vor allem im Bild eher horizontal verlaufende Fahrspurränder besser gemessen werden, und die Vorausschau in der Kurve wird deutlich erhöht. Um Linien anstatt einzelner Punkte zu messen, wird in (c) zusätzlich für jedes Messfenster eine zweite Messung in einem (entlang des Wegs) nach vorne versetzten Messfenster durchgeführt.

ab, so wird die Messung als Ausreißer verworfen. Andernfalls kann der gemessene Winkel bei der Korrektur der Spurbegrenzungsschätzung helfen.

8.4.3 CRONOS-Messungen auf Feldwegen

Mit diesen vorgestellten Erweiterungen für Linienmessungen kann das CRONOS-Verfahren nun auch zur Messung der Wegränder von Feldwegen verwendet werden. Im Straßentracking wird eine sehr schmale Faltungsmaske mit nur einer Nullspalte und je zwei positiv und negativ besetzten Spalten (vgl. Abbildung 8.5) verwendet. Die Kanten eines Feldwegs sind jedoch weniger deutlich definiert. Daher wird im Wegetracking eine deutlich breitere Faltungsmaske mit fünf Nullspalten und je zwanzig positiv und negativ besetzten Spalten verwendet. Zusätzlich wird die Auflösung des Bilds auf 40 % reduziert, um kleine Details wegzuglätten. Dadurch wird außerdem die relative Höhe (bzw. bei vertikalen Fenstern die relative Breite) der Messfenster vergrößert, sodass ein größerer Bildbereich betrachtet wird (die Höhe ist historisch bedingt auf 17 px limitiert). Die Breite (bzw. bei vertikalen Fenstern die Höhe) des Messfensters wird ebenfalls auf 2 m bzw. 1,5 m erhöht.

Oftmals weisen Feldwege allerdings im Graubild einen zu geringen Kontrastunterschied zur Umgebung auf. In Abbildung 8.9 ist ein Beispiel hierfür zu sehen. Abbildung 8.9b zeigt dabei die durch die Mittelung der RGB-Farbkanäle berechnete Intensität

$$\mathbf{Y}' = (\mathbf{R}_{\mathsf{RGB}} + \mathbf{G}_{\mathsf{RGB}} + \mathbf{B}_{\mathsf{RGB}})/3\,, \tag{8.23}$$

und Abbildung 8.9c die beispielsweise durch die Bildverarbeitungsbibliotheken *HALCON* [MVTec Software GmbH, 2023] und *OpenCV* [Bradski, 2000] verwendete, verbesserte Abbildung

$$\mathbf{Y}_{\mathsf{YUV}} = 0{,}299 \cdot \mathbf{R}_{\mathsf{RGB}} + 0{,}587 \cdot \mathbf{G}_{\mathsf{RGB}} + 0{,}114 \cdot \mathbf{B}_{\mathsf{RGB}}\,, \tag{8.24}$$

die dem Helligkeitskanal $\mathbf{Y}_{\mathsf{YUV}}$ aus dem YUV-Farbmodell entspricht. In beiden Bildern wird kaum eine erfolgreiche CRONOS-Messung gefunden. Im Farbbild in Abbildung 8.9a ist jedoch ein deutlicher Kontrast zur umliegenden Wiese zu sehen. Das Graubild – auch mit der verbesserten Abbildung – bildet die Informationen aus dem Farbbild also unzureichend ab. Um die Farbinformation besser nutzen zu können, müssen daher andere Farbräume und Farbkanäle untersucht werden. Im Folgenden werden dazu zwei neue Farbmodi $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$ (Abbildung 8.9d) und *Relative Sättigung* (Abbildung 8.9e) vorgestellt, die für viele verschiedene Feldwege den Kontrast verbessern.

In der Literatur wird häufig die Sättigung $\mathbf{S}$ aus dem HSI- oder dem HSV-Farbraum (Farbton, Sättigung, Hellwert, engl. *Hue, Saturation, Value*) betrachtet [Manz et al., 2010, Bayerl und Wuensche, 2014, Fleischmann et al., 2016]. Oftmals wird dabei davon ausgegangen, dass die Straße eine niedrigere Sättigung aufweist als die Umgebung. Dies ist jedoch nicht immer der Fall, wie Abbildung 8.10 zeigt. Weiterhin werden zur Erkennung von Feldwegen die $\mathbf{a}^*$- und $\mathbf{b}^*$- Kanäle des CIELAB-Farbraums [Fleischmann et al., 2016], der von Gevers und Smeulders [1999] vorgestellte $c_1c_2c_3$-Farbraum [Song et al., 2006, Lewis et al., 2012, Fleischmann et al., 2016], die Green-Ratio $\mathbf{G}_{\mathsf{RGB}}/(\mathbf{R}_{\mathsf{RGB}} + \mathbf{G}_{\mathsf{RGB}} + \mathbf{B}_{\mathsf{RGB}})$ (auch *normalized green* genannt) [Tan et al., 2006, Manz et al., 2011, Bayerl und Wuensche, 2014] oder der Farbwert $\mathbf{H}_{\mathsf{HSI}}$ aus dem HSI-Farbraum [Chaturvedi et al., 2004, Franke et al., 2007] vorgeschlagen. In Abbildung 8.11 werden alle diese Farbräume anhand des Beispiel-Feldwegs aus Abbildung 8.9 miteinander verglichen. Jeder davon erhöht den Kontrast des Wegs im Vergleich zum Graubild in Abbildung 8.9c, wobei der Farbwert aus dem HSI-Farbraum und die Sättigung aus den Farbräumen HSI und HSV in diesem Beispiel das beste Resultat zeigen. Bei einem anderen Feldweg, wie er in Abbildung 8.12 gezeigt wird, fehlt jedoch auch diesen Farbmodi der Kontrast.

Um die Sättigung robuster zu machen, empfiehlt es sich, nicht direkt die absolute Sättigung $\mathbf{S}$ zu verwenden, sondern eine relative Sättigung

$$\mathbf{RS} = \mathbf{S}_{\mathsf{HSV}}/\mathbf{V}_{\mathsf{HSV}} \tag{8.25}$$

in Abhängigkeit vom Hellwert $\mathbf{V}_{\mathsf{HSV}}$ aus dem HSV-Farbraum (beziehungsweise der Lichtintensität $\mathbf{I}_{\mathsf{HSI}}$ bei Verwendung des HSI-Farbraums) zu berechnen. Wie die Abbildungen 8.9e, 8.12e und 8.12f zeigen, kann dadurch der Kontrast des Wegs zu



(a) RGB-Farbbild



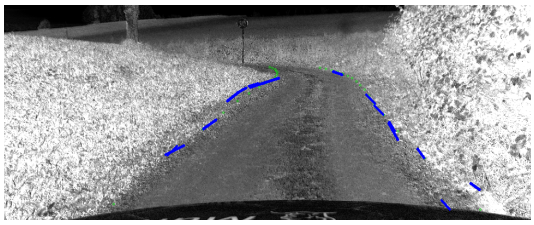
(b) $\mathbf{Y}' = (\mathbf{R}_{\text{RGB}} + \mathbf{G}_{\text{RGB}} + \mathbf{B}_{\text{RGB}})/3$



(c) $\mathbf{Y}_{\text{YUV}}$



(d) $\mathbf{Y}_{\text{YUV}} - \mathbf{I3}_{\text{I1I2I3}}$



(e) Relative Sättigung: $\mathbf{S}/\mathbf{V}$

Abbildung 8.9:
Visualisierung der CRONOS-Messungen eines Feldwegs bei verschiedenen Farbmodi. (a) zeigt das volle RGB-Farbbild. (b) und (c) zeigen auf unterschiedliche Art und Weise berechnete Graubilder. In beiden besteht kaum Kontrast zwischen Feldweg und Wiese. Dadurch kann jeweils nur für einen Spurbegrenzungspunkt eine erfolgreiche Bildmessung durchgeführt werden (blaue Linien). (d) und (e) zeigen die in dieser Arbeit entwickelten Farbmodi, die für viele Feldwege einen deutlich besseren Kontrast bieten. Hier können einige Bildmessungen der Wegränder extrahiert werden.

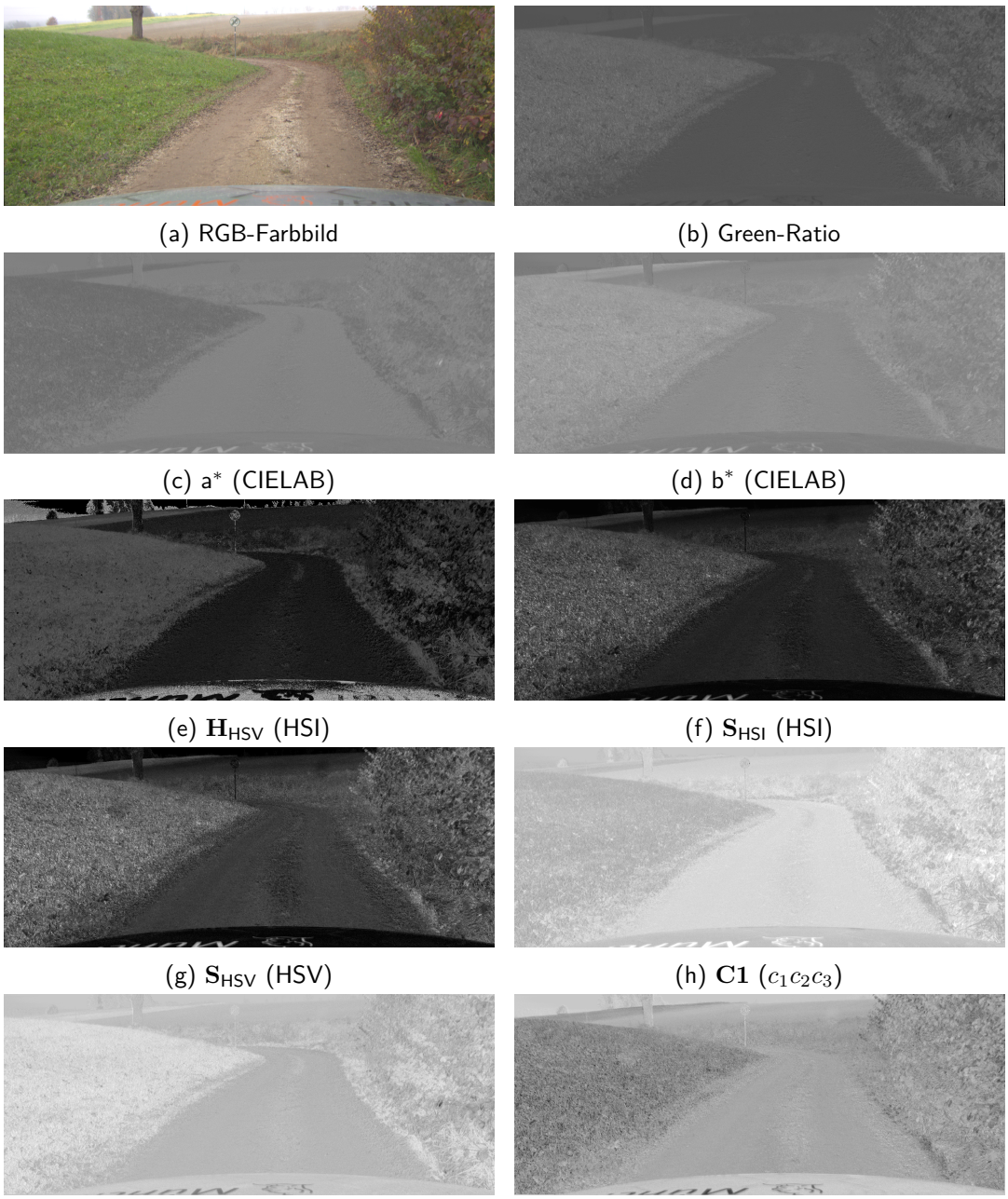


(a) RGB-Farbbild



(b) Sättigung (HSV)

Abbildung 8.10:
Beispiel einer Straße, deren Sättigung aus dem HSV-Farbraum höher ist als die des Straßenrands (helle Pixel zeigen eine hohe Sättigung) – anders als oft angenommen.



(a) RGB-Farbbild (b) Green-Ratio

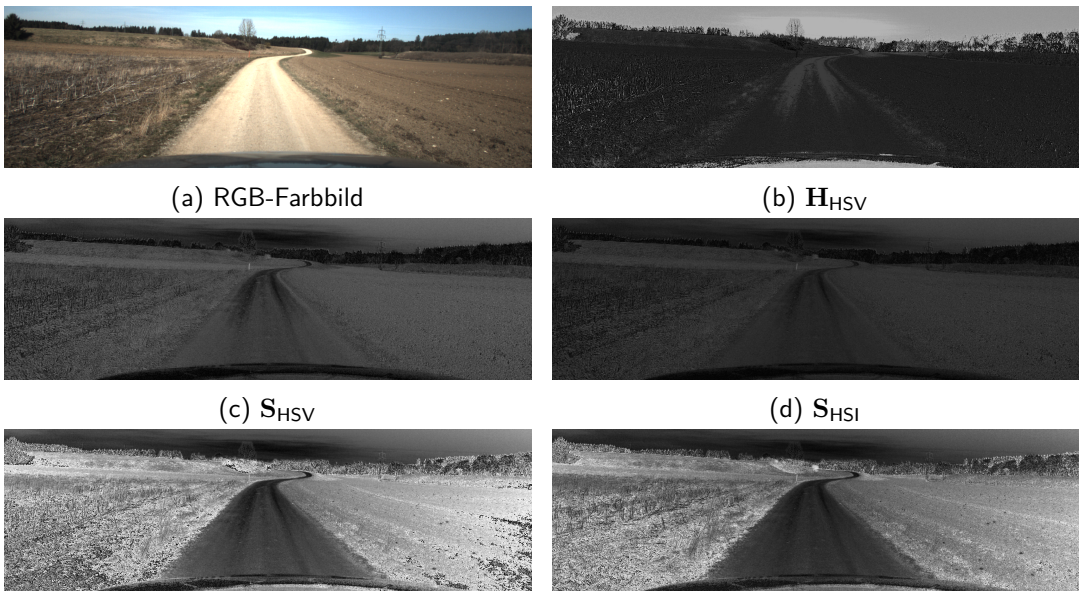
(c) a* (CIELAB) (d) b* (CIELAB)

(e) $\mathbf{H}_{\text{HSV}}$ (HSI) (f) $\mathbf{S}_{\text{HSI}}$ (HSI)

(g) $\mathbf{S}_{\text{HSV}}$ (HSV) (h) $\mathbf{C1}$ ($c_1c_2c_3$)

(i) $\mathbf{C2}$ ($c_1c_2c_3$) (j) $\mathbf{C3}$ ($c_1c_2c_3$)

Abbildung 8.11:
Visualisierung der in der Literatur vorgeschlagenen Farbmodi zur besseren Erkennung von Feldwegen. Die Farbkanäle stammen aus den Farbräumen CIELAB, $c_1c_2c_3$, RGB (Green-Ratio), HSI und HSV.



(a) RGB-Farbbild (b) $\mathbf{H}_{\text{HSV}}$

(c) $\mathbf{S}_{\text{HSV}}$ (d) $\mathbf{S}_{\text{HSI}}$

(e) Relative Sättigung (HSV) (f) Relative Sättigung (HSI)

Abbildung 8.12:
Beispiele für den durch die Verwendung der vorgestellten relativen Sättigung gewonnenen Kontrast verglichen mit der einfachen Sättigung (oder dem Farbwert) eines zweiten Beispiel-Feldwegs.



(a) RGB-Farbbild (b) Sättigung (HSV) (c) Relative Sättigung (HSV)

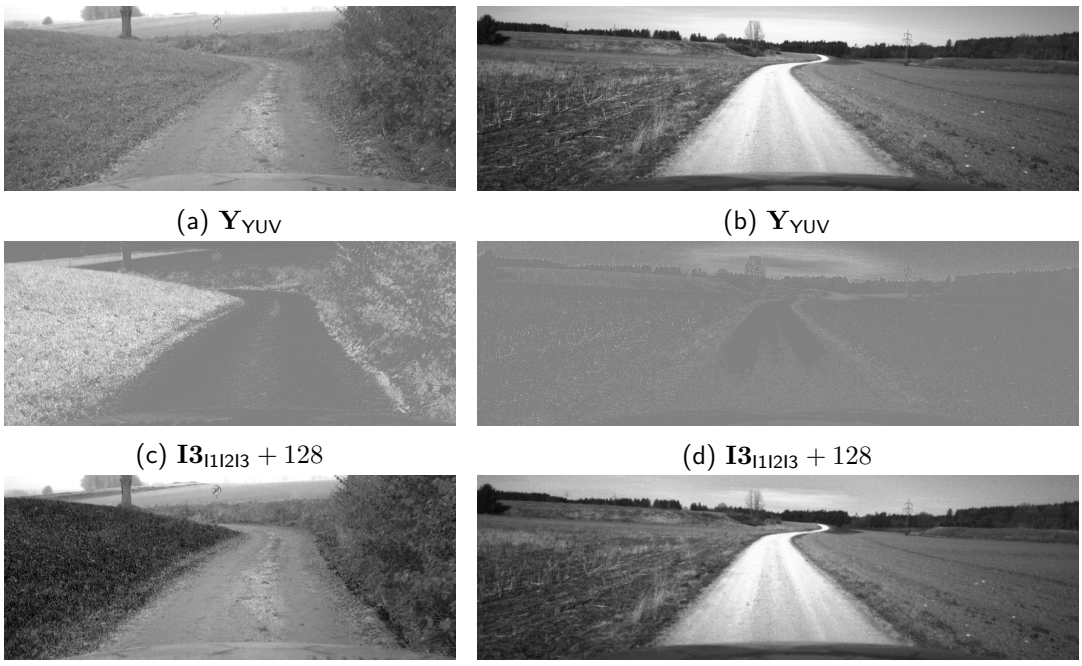
Abbildung 8.13:
Beispiele für den durch die Verwendung der vorgestellten relativen Sättigung gewonnenen Kontrast verglichen mit der einfachen Sättigung anhand eines dritten Beispiel-Feldwegs.

seiner Umgebung signifikant erhöht werden. Dabei unterscheiden sich die relativen Sättigungen aus dem HSI- und dem HSV-Farbraum nur wenig. Da sich die HSV-Farbkanäle leichter mittels *OpenCV* [Bradski, 2000] berechnen lassen, wird daher der HSV-Farbraum verwendet. Abbildung 8.13 zeigt ein drittes Beispiel.

Ein anderer für die Wegeerkennung interessanter Farbkanal ist der $\mathbf{I3}_{\text{I1I2I3}}$-Kanal aus dem von Ohta et al. [1980] vorgestellten I1I2I3-Farbraum. In der Landwirtschaft wird dieser Farbkanal verwendet, um lebende Pflanzen von einem nicht-grünen Hintergrund zu unterscheiden [Woebbecke et al., 1995]. Er berechnet sich als

$$\mathbf{I3}_{\text{I1I2I3}} = 2\mathbf{G}_{\text{RGB}} - \mathbf{R}_{\text{RGB}} - \mathbf{B}_{\text{RGB}} \tag{8.26}$$

und stellt dadurch Grüntöne heraus. Wie man in Abbildung 8.14c sehen kann, führt dies bei grüner Vegetation neben dem Weg zu sehr guten Kontrasten zum gesuchten Weg. Ohne grüne Umgebung ist der $\mathbf{I3}_{\text{I1I2I3}}$-Kanal allerdings sehr unspezifisch (vgl. Abbildung 8.14d) und damit nicht direkt zur Kantenmessung geeignet.



(a) $\mathbf{Y}_{\mathsf{YUV}}$ (b) $\mathbf{Y}_{\mathsf{YUV}}$

(c) $\mathbf{I3}_{\mathsf{I1I2I3}} + 128$ (d) $\mathbf{I3}_{\mathsf{I1I2I3}} + 128$

(e) $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$ (f) $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$

Abbildung 8.14:
Zwei Beispiele für die Kombination des Graubilds $\mathbf{Y}_{\mathsf{YUV}}$ mit dem $\mathbf{I3}_{\mathsf{I1I2I3}}$-Kanal des I1I2I3-Farbraums zu dem vorgestellten Farbmodus $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$. Für die bessere Visualisierung sind die $\mathbf{I3}_{\mathsf{I1I2I3}}$-Bilder aufgehellt dargestellt. Bei einer grünen Umgebung (linke Spalte) wird der Kontrast des Graubilds durch die Subtraktion des $\mathbf{I3}_{\mathsf{I1I2I3}}$-Bilds deutlich verbessert. Gleichzeitig wird der Kontrast in nicht-grünen Umgebungen (rechte Spalte) mit einem unspezifischen $\mathbf{I3}_{\mathsf{I1I2I3}}$-Kanal nicht beeinträchtigt. Die zugehörigen RGB-Farbbilder der Beispiel-Feldwege sind in den Abbildungen 8.11a und 8.12a zu sehen.

Daher wird der $\mathbf{I3}_{\mathsf{I1I2I3}}$-Kanal stattdessen dazu verwendet, den Kontrast des Graubilds zu verbessern. Dazu werden durch die Berechnung der Differenz

$$\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}} = 1{,}299 \cdot \mathbf{R}_{\mathsf{RGB}} - 1{,}413 \cdot \mathbf{G}_{\mathsf{RGB}} + 1{,}114 \cdot \mathbf{B}_{\mathsf{RGB}} \tag{8.27}$$

die grünen Bildbereiche im Graubild zusätzlich verdunkelt. So kann aus dem kontrastarmen Graubild in Abbildung 8.14a durch die Subtraktion des $\mathbf{I3}_{\mathsf{I1I2I3}}$-Kanals in Abbildung 8.14c das kontrastreiche $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$-Bild in Abbildung 8.14e gewonnen werden. Im Falle eines unspezifischen $\mathbf{I3}_{\mathsf{I1I2I3}}$-Kanals in einer nicht-grünen Umgebung, wie in Abbildung 8.14d, bleiben im kombinierten $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$-Bild in Abbildung 8.14f trotzdem die Kanten des Graubilds (Abbildung 8.14b) erhalten.

Mit diesen beiden neuen Farbmodi können nun weit mehr unmarkierte Straßen und Feldwege mithilfe der CRONOS-Bildmessungen erfolgreich getrackt werden. Da die CRONOS-Messung sehr effizient ist und nur etwa 9 ms pro Bild benötigt, können problemlos beide Farbmodi parallel verwendet werden. Ist nicht im Vorhinein festgelegt, ob eine Straße oder ein Feldweg getrackt werden soll, so kann die Messung im $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$-Bild mit den Einstellungen des Straßentrackings und die Messung in der relativen Sättigung mit den Einstellungen des Wegetrackings erfolgen. Alternativ muss bei den Parametern ein Kompromiss gefunden werden.

8.5 Messungen aus LiDAR-Informationen

Die Messungen aus den Kamerabildern werden nun mit zusätzlichen Messungen aus 3D-LiDAR-Informationen fusioniert. Diese beinhalten indirekte Messungen aus dem LiDAR-basierten Umgebungsmodell (Terrain-Schätzung und Hindernis-Grid) sowie direkte Messungen aus einer segmentierten LiDAR-Punktwolke. Die Kosten $\mathbf{c}_{\mathsf{3D}}(\mathbf{x}_i)$ setzen sich damit aus drei im folgenden Abschnitt definierten Residuen zusammen:

$$\mathbf{c}_{\mathsf{3D}}(\mathbf{x}_i) = \sum_k \left(\mathbf{r}^{(k)}_{\mathsf{terrain}}(\mathbf{x}_i)\right)^2 + \left(\mathbf{r}_{\mathsf{obstacles}}(\mathbf{x}_i)\right)^2 + \sum_k \sum_j \rho_{\mathsf{H}}\left(\mathbf{r}^{(k)}_{\mathsf{lidar},j}(\mathbf{x}_i)\right) . \quad (8.28)$$

Das Umgebungsmodell wird unabhängig vom Straßen- und Wegetracking erzeugt und verwendet kein Vorwissen in Form prädizierter Fahrspurrandpunkte. Dadurch kann das Tracking im Falle einer sehr schlechten Schätzung oder bei der Initialisierung durch unabhängige Messungen korrigiert werden. Zudem wird durch die unabhängigen Informationen indirekt der Suchbereich für die lokalen Bildmessungen eingeschränkt. Die Messungen in der segmentierten LiDAR-Punktwolke sorgen ferner für eine erhöhte Robustheit bei visuell schwierigen Feldwegen.

8.5.1 Informationen aus der Terrain-Schätzung

Mithilfe der CRONOS-Kantenmessungen kann die Position der Spurbegrenzungspunkte im zweidimensionalen Kamerabild gemessen werden. Die Zustandsschätzung der Spurbegrenzungspunkte erfolgt jedoch im dreidimensionalen Raum. Ohne Stereoverarbeitung können aus einer Kameramessung im Dreidimensionalen zwar der horizontale und der vertikale Winkel des Spurbegrenzungspunkts zur Kamera bestimmt werden, nicht jedoch seine Entfernung zur Kamera entlang des Sehstrahls. Diese könnte lediglich mit wechselnder Perspektive über die Zeit geschätzt werden. Nach einer Idee aus [Kallwies, 2023] kann jedoch ausgenutzt werden, dass sich die Straße auf dem Boden befindet und dieser fehlende Freiheitsgrad über Informationen aus der Terrain-Schätzung aufgelöst werden: Der Schnittpunkt des Sehstrahls von der Kamera zum gemessenen Spurbegrenzungspunkt mit der Terrain-Schätzung ergibt die gesuchte dreidimensionale Messung des Spurbegrenzungspunkts. Anstatt diesen Schnittpunkt jedoch explizit zu berechnen, wird im Folgenden das gesamte LiDAR-basierte Terrain-Grid in das Straßen- und Wegetracking fusioniert.

Höhenmessung aus der Terrain-Schätzung Über das Höhen-Residuum

$$\mathbf{r}_{\mathsf{terrain},[1]}{}^{(k)}(\mathbf{x}_i) = (z - h_{\mathsf{terrain}}(x,y)) \cdot \mathbf{P}^{-1}_{\mathsf{terrain}}(x,y) \cdot \mathbf{w}_{[\mathsf{terrain}_h]} \quad (8.29)$$

wird zunächst eine Bedingung für die Höhe des Spurbegrenzungspunkts aufgestellt: Die Abweichung der z-Koordinate eines Fahrspurrandpunkts $\mathbf{x}_i^{(k)}$ zu der aus dem Terrain-Grid interpolierten, geschätzten Höhe $h_{\mathsf{terrain}}(x,y)$ des Terrains an der entsprechenden Stelle soll minimal sein. Die Abweichung wird dabei mit der akkumulierten Information (inverse Kovarianz) $\mathbf{P}^{-1}_{\mathsf{terrain}}(x,y)$ aus der Terrain-Schätzung sowie einem Gewichtungsfaktor $\mathbf{w}_{[\mathsf{terrain}_h]} = 0{,}1$ gewichtet.

Normalen aus der Terrain-Schätzung Mithilfe der Normalen aus der Terrain-Schätzung kann zudem die Orientierung des gemessenen Fahrspurrandpunkts im Raum bestimmt werden. Dazu wird für jeden geschätzten Fahrspurrandpunkt $\mathbf{x}_i^{(k)}$ die Normale

$$\mathbf{n}_i^{(k)} = \begin{bmatrix} n_\mathrm{x} \\ n_\mathrm{y} \\ n_\mathrm{z} \end{bmatrix} = \mathbf{R}\left({}^{\mathrm{odom}}\mathbf{q}_i^{(k)}\right) \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{8.30}$$

berechnet, wobei $\mathbf{R}(\mathbf{q})$ die zu $\mathbf{q}$ gehörige Rotationsmatrix darstellt. Im Residuum

$$\mathbf{r}^{(k)}_{\mathrm{terrain},[2,3,4]}(\mathbf{x}_i) = \left(\mathbf{n}_i^{(k)} - \mathbf{n}_\mathrm{terrain}(x,y)\right) \cdot \mathbf{P}^{-1}_\mathrm{terrain}(x,y) \cdot \mathbf{w}_{[\mathrm{terrain}_\mathbf{n}]} \tag{8.31}$$

wird dann die Abweichung dieser Normale von der aus der Terrain-Schätzung interpolierten Normale $\mathbf{n}_\mathrm{terrain}(x,y)$ minimiert, wobei $\mathbf{w}_{[\mathrm{terrain}_\mathbf{n}]}$ auf 0,05 gesetzt wird.

In Kombination mit der Limitierung der vertikalen Steigung in den Modell-Kosten führt das Normalen-Residuum dazu, dass vorzugsweise in ebenen Bereichen nach Wegen gesucht wird. Dies liegt daran, dass die Krümmungen durch das Limitierungs-Residuum $\mathbf{r}^{(k)}_{\mathrm{limit}_{c_\mathrm{v}}}(\mathbf{x}_i)$ nicht nur begrenzt, sondern durch die darin enthaltene Exponentialfunktion auch leicht minimiert werden. Auf diese Art und Weise kann allein mit den Informationen aus der Terrain-Schätzung bereits manchem Wegverlauf grob gefolgt werden, und es werden implizit Gräben vermieden. In Kombination mit den Bildmessungen können damit erfolgreich Straßen und Wege getrackt werden.

8.5.2 Informationen aus dem Hindernis-Grid

Auch das Hindernis-Grid kann das Straßen- und Wegetracking unterstützen: Meist kann davon ausgegangen werden, dass der zu verfolgende Weg frei von statischen Hindernissen ist. Hindernisse neben dem Weg wie Bäume, Sträucher oder Straßenpfosten können daher zur Eingrenzung der möglichen Wegposition verwendet werden. Dies gelingt durch die Einführung von Kosten für Hindernisse auf dem geschätzten Weg. Abbildung 8.15a visualisiert das Prinzip: Bei jedem Bezugsspurbegrenzungs-

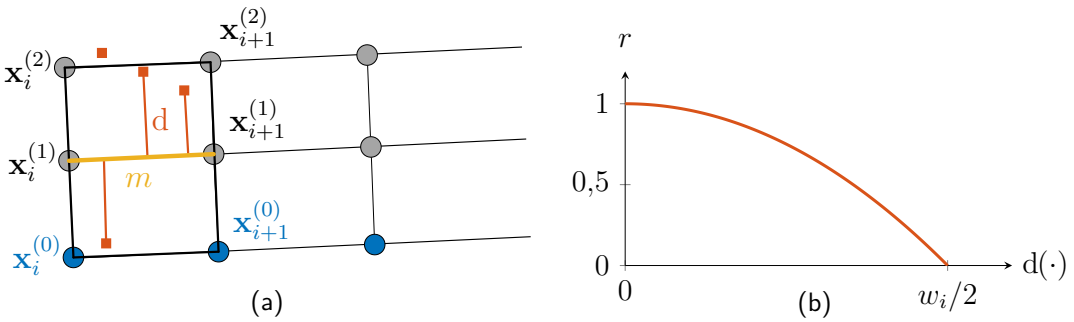
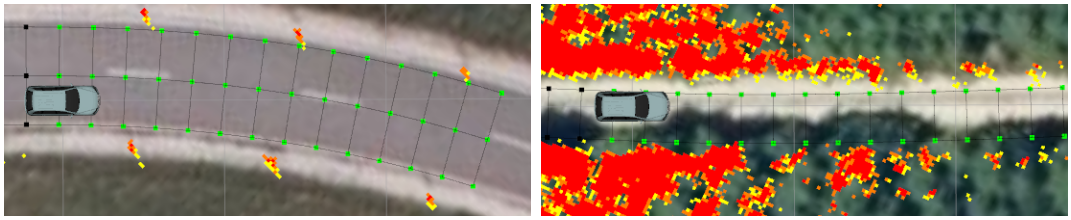


Abbildung 8.15:
Schematische Erklärung, wie die statischen Hindernisse im Straßen- und Wegetracking berücksichtigt werden. Für alle Hindernisse o (rot) im Polygon eines Straßenabschnitts (schwarz) der in (a) gezeigten Straße wird der Abstand $\mathrm{d}(o,m)$ zur Straßen-Mittellinie m (gelb) ausgewertet. Dieser wird mittels der in (b) abgebildeten Kostenfunktion $r = \mathrm{r}(o,\mathbf{x}_i)$, die zum Straßenrand hin abfällt, maximiert. w_i gibt dabei die Straßenbreite bei $\mathbf{x}_i^{(0)}$ an.

punkt $\mathbf{x}_i^{(0)}$ wird der Straßenabschnitt betrachtet, der durch das Polygon zwischen $\mathbf{x}_i^{(k_{\text{min}})}$, $\mathbf{x}_{i+1}^{(k_{\text{min}})}$, $\mathbf{x}_{i+1}^{(k_{\text{max}})}$ und $\mathbf{x}_i^{(k_{\text{max}})}$ aufgespannt wird. Jedes Hindernis o in diesem Straßenabschnitt mit einer Hinderniswahrscheinlichkeit größer als 0,5 soll Kosten verursachen. Um einen Gradienten für den Gradientenabstieg in der Optimierung zu erzeugen, werden diese Kosten abhängig vom Abstand $\mathrm{d}(\cdot)$ des Hindernisses zur Mittellinie m des Wegs definiert. Durch die Minimierung des Residuums

$$\mathbf{r}_{\text{obstacles}}(\mathbf{x}_i) = \sum_o \max\left(0, \underbrace{-\frac{1}{(w_i/2)^2} \cdot \mathrm{d}(o,m)^2 + 1}_{\mathrm{r}(o,\mathbf{x}_i)}\right) \cdot \mathbf{w}_{[\text{obstacles}]} \tag{8.32}$$

wird der Abstand des Hindernisses zur Mitte des Wegs maximiert, wobei $w_i = s_i^{(k_{\text{max}})} + \left|s_i^{(k_{\text{min}})}\right|$ die Breite der Straße angibt ($s_i^{(0)} = 0$) und $\mathbf{w}_{[\text{obstacles}]} = 10$ gesetzt wird. Abbildung 8.15b visualisiert diese mit der Distanz eines Hindernisses zur Mittellinie sinkende Kostenfunktion $\mathrm{r}(o, \mathbf{x}_i)$, die am Rand der Straße schließlich null erreicht. Durch die monoton fallende Kostenfunktion wird ein klarer Gradient garantiert und der Weg durch die Optimierung sozusagen „aus dem Hindernis herausgeschoben“. Anhand dieser indirekten LiDAR-Informationen aus dem Hindernis-Grid und den Terrain-Höhen können die Straße und der Weg in Abbildung 8.16 ohne weitere Messungen allein aufgrund ihrer Hindernisfreiheit getrackt werden.



(a) Markierte Straße (b) Waldweg

Abbildung 8.16:
Zwei Beispiele nur mithilfe des Umgebungsmodells (ohne Bild- oder direkte LiDAR-Messungen) erfolgreich getrackter Straßen und Wege. Zur Visualisierung werden die geschätzten Spurbegrenzungspunkte auf einem Luftbild gezeigt.

8.5.3 Messungen aus einer segmentierten LiDAR-Punktwolke

Durch eine direkte Verarbeitung der LiDAR-Punktwolke können jedoch noch gezieltere Messungen erhalten werden. In der Terrain-Schätzung wurde zur Identifikation möglicher Wegbereiche die geometrische LiDAR-Punktwolken-Segmentierung von Burger und Wuensche [2018] genutzt. Diese identifiziert sog. Bodenflächenpunkte auf ebenen Flächen, die auf einen Weg hindeuten können. Davon kann auch im Straßen- und Wegetracking profitiert werden. In [Burger et al., 2019] werden aus diesen Bodenflächenpunkten direkt die Ränder von Feldwegen extrahiert. Auf Straßen ist die befestige Fläche jedoch meist breiter als die markierten Fahrspuren. Oft schließt sich auch ein ebenes Bankett an. Daher können die Ränder der Bodenflächenpunkte hier nur als äußere Grenze des möglichen Wegbereichs dienen.

In ländlicherem Gebiet sind die Straßen und Wege häufig von ebenen Feldern umgeben, sodass auch neben dem Weg Bodenflächenpunkte segmentiert werden. Abbildung 8.17a zeigt ein Beispiel hierfür. Daher werden nur diejenigen Bodenflächen-Instanzen betrachtet, von denen mindestens ein Punkt innerhalb der aktuellen Schätzung des Wegs liegt. Diese gefilterten Bodenflächenpunkte werden anschließend über drei Zeitschritte akkumuliert, wie in Abbildung 8.17b zu sehen ist. Aus der akkumulierten Punktwolke wird dann für jedes Set an lateral benachbarten Spurbegrenzungspunkten ein (in Abbildung 8.17c blau visualisierter) lateraler Bereich bestimmt, innerhalb dessen sich alle Spurbegrenzungen befinden sollen. Aufgrund dieses Bereichs wird die Messung *LiDAR-Range-Messung* genannt.

Die einzelnen Schritte zu dieser LiDAR-Range-Messung werden in Abbildung 8.18 visualisiert. Aus den akkumulierten Bodenflächenpunkten werden die linken und rechten Randpunkte jedes LiDAR-Rings und jeder segmentierten Bodenflächen-Instanz extrahiert (siehe Abbildung 8.18b). In diese wird mittels Optimierung je eine Klothoide für den linken und rechten Rand gefittet. Jede der Klothoiden ist über ihre Anfangskrümmung c, die Krümmungsänderung $\mathring{c}$, die laterale Ablage d zum Ego-Fahrzeug und den Gierwinkel $\Delta\Psi$ zum Fahrzeugkoordinatensystem definiert:

$$\mathbf{x}' = \begin{bmatrix} c & \mathring{c} & d & \Delta\Psi \end{bmatrix}^\mathsf{T} . \tag{8.33}$$

Folglich sollen für alle Randpunkte ${}^{\text{ego}}\mathbf{b} = [x_\mathbf{b}\ y_\mathbf{b}]^\mathsf{T}$ (in Fahrzeugkoordinaten) die Kosten

$$\mathbf{c}_{\text{cl.}}(\mathbf{b}, \mathbf{x}') = \rho_\text{C}\Big({}^{\text{cloth}}y_\mathbf{b} - \mathrm{f}\Big({}^{\text{cloth}}x_\mathbf{b}\Big)\Big) \cdot \mathbf{w}_{[\text{cl.}]} , \tag{8.34}$$

mit

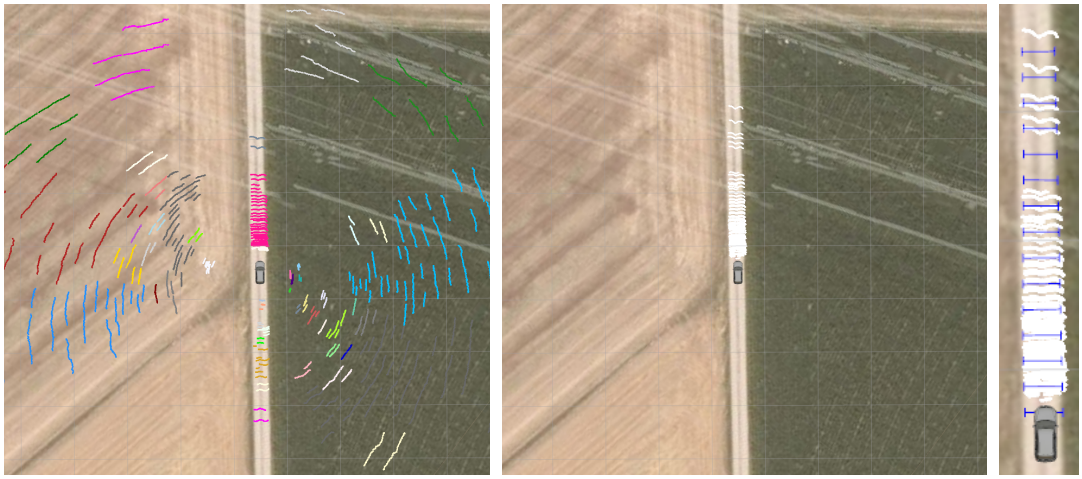
$${}^{\text{cloth}}\mathbf{b} = \begin{bmatrix} \cos(\Delta\Psi) & \sin(\Delta\Psi) & -d\sin(\Delta\Psi) \\ -\sin(\Delta\Psi) & \cos(\Delta\Psi) & -d\cos(\Delta\Psi) \\ 0 & 0 & 1 \end{bmatrix} \cdot {}^{\text{ego}}\mathbf{b} \tag{8.35}$$

und der Klothoidennäherung $\mathrm{f}(l)$ aus Gleichung (8.8) minimiert werden. Um Ausreißer weniger stark zu gewichten, wird die Cauchy-Verlustfunktion $\rho_\text{C}(\cdot)$ angewandt (s. Abschnitt 2.5), wobei $\mathbf{w}_{[\text{cl.}]} = 0{,}05$. Ein weiteres Residuum minimiert den Abstand des Ursprungs der Klothoide zu dem im Wegetracking geschätzten Fahrbahnrand:

$$\mathbf{r}_{\text{disp.}}(\mathbf{x}') = \left(d - {}^{\text{ego}}y_i^{(k_{\text{min/max}})}\right) \cdot \mathbf{w}_{[\text{disp.}]} , \tag{8.36}$$

wobei ${}^{\text{ego}}y_i^{(k_{\text{min/max}})}$ die laterale Ablage der rechtesten (für die rechte Klothoide) bzw. linkesten (für die linke Klothoide) Spurbegrenzung zum Ego-Fahrzeug angibt, und $\mathbf{w}_{[\text{disp.}]} = 0{,}1$. Auf diese Weise werden zum einen Ausreißer ignoriert, zum anderen kann im Falle mehrerer möglicher Straßen (wie bei der Abzweigung in Abbildung 8.18c) diejenige gewählt werden, die besser zur aktuell getrackten Straße passt. Weitere Residuen limitieren entsprechend der im Wegetracking in Gleichung (8.18) vorgestellten Limitierungsfunktion die mögliche Krümmung auf $0{,}025\,\text{m}^{-1}$ sowie die mögliche Krümmungsänderung auf $0{,}003\,\text{m}^{-2}$, um nur wahrscheinliche Ergebnisse zu erhalten. Somit stellt sich insgesamt das Optimierungsproblem

$$\hat{\mathbf{x}}' = \arg\min_{\mathbf{x}'} \sum_\mathbf{b} \mathbf{c}_{\text{cl.}}(\mathbf{b}, \mathbf{x}') + \mathbf{r}^2_{\text{disp.}}(\mathbf{x}') + \mathbf{r}^2_{\text{limit}_c}(\mathbf{x}') + \mathbf{r}^2_{\text{limit}_{\mathring{c}}}(\mathbf{x}') . \tag{8.37}$$



(a) (b) (c)

Abbildung 8.17:
LiDAR-Range-Messung: Beispiel für die Verwendung des Segmentierungsverfahrens von Burger et al. [2019] als direkte, lokale Messung im Wegetracking. (a) Segmentierte Bodenflächen-Instanzen, farblich unterschieden. Neben den gesuchten Punkten auf dem Weg werden auch viele Punkte auf den umliegenden Feldern segmentiert. (b) Anhand der aktuellen Schätzung des Wegs gefilterte sowie akkumulierte Bodenflächenpunkte. (c) Aus den akkumulierten Bodenflächenpunkten gewonnene *LiDAR-Range-Messungen* (in blau visualisiert). Diese bestimmen für jedes Set an lateral benachbarten Punkten einen lateralen Bereich, in dem sich vermutlich der Weg befindet. Das Luftbild dient lediglich der Visualisierung, die Lokalisierung hierfür erfolgt auf Basis von RTK-GNSS.



(a) Segmentierte Punktwolke (b) Linke und rechte Randpunkte (c) Linke und rechte Klothoide (d) LiDAR-Range-Messungen

Abbildung 8.18:
Visualisierung der einzelnen Schritte der LiDAR-Range-Messung an einer Straßen-Abzweigung. (a) Zuerst werden mittels des LiDAR-Punktwolken-Clusterings von [Burger et al., 2019] die Bodenflächen-Instanzen im Bereich der Straßenschätzung herausgefiltert. (b) Dann werden die linken (rot) und rechten (grün) Randpunkte jeder Instanz und jedes LiDAR-Rings extrahiert. Aufgrund einer Verdeckung liegen die rechten Randpunkte der rechten Abzweigung innerhalb des Fahrbahnrands. (c) Die Randpunkte werden verwendet, um mittels Optimierung Klothoiden (rote und grüne Linien) für den linken und rechten Straßenrand zu fitten. Dabei wird die aktuelle Straßenschätzung berücksichtigt, um die zur Schätzung passende Abzweigung zu präferieren. (d) Zuletzt wird für jedes Set an benachbarten Spurbegrenzungspunkten ein lateraler Bereich der Straße (blau) gemessen.

Die Lösung des Optimierungsproblems mit dem *Ceres Solver* [Agarwal et al., 2020] ergibt schließlich für jeden der beiden Fahrbahnränder eine Klothoide.

Aus diesen Klothoiden wird nun für jedes Set an benachbarten Spurbegrenzungspunkten ein möglicher Wegbereich extrahiert. Dafür wird in der Mitte der zum Zeitpunkt der Messung j geschätzten äußeren Spurbegrenzungspunkte $\mathbf{x}_i^{(k_{\mathsf{min}})^j}$ bzw. $\mathbf{x}_i^{(k_{\mathsf{max}})^j}$ ein lokales kartesisches Koordinatensystem $\mathbf{F}_{\mathsf{road},i}^j$ aufgespannt. Innerhalb dieses Koordinatensystems werden mittels der linken und rechten Klothoide obere und untere Schranken für die y-Koordinaten der Spurbegrenzungspunkte bestimmt. Das Ergebnis ist ein lateraler Bereich $[y_{\mathsf{min},i}^j,\ y_{\mathsf{max},i}^j]$, in dem sich die y-Koordinaten der Spurbegrenzungspunkte $\mathbf{x}_i^{(k)}$ im Koordinatensystem $\mathbf{F}_{\mathsf{road},i}^j$ bewegen dürfen. In Abbildung 8.18d wird der gemessene Bereich in blau dargestellt. Abbildung 8.19 zeigt ein schematisches Beispiel. Falls der Klothoidenfit für den linken oder rechten Wegrand fehlschlägt, wird die jeweilige Schranke auf unendlich gesetzt.

Diese Schranken werden in das Straßen- und Wegetracking integriert, indem eine Kostenfunktion aufgestellt wird, die die Kosten für Wegrandpunkte außerhalb dieser Schranken erhöht. Dazu dient die Kostenfunktion $\mathbf{r}_{\mathsf{lidar}}(\mathbf{x}_i)$ in Gleichung (8.28). Zur Auswertung dieser Kostenfunktion werden die Positionen der einzelnen Spurbegrenzungspunkte ${}^{\mathsf{odom}}\mathbf{p}_i^{(k)}$ (entsprechend der Zustände $\mathbf{x}_i$) zunächst in $\mathbf{F}_{\mathsf{road},i}^j$ transformiert:

$$ {}^{\mathsf{road}_i^j}\mathbf{p}_i^{(k)} = {}^{\mathsf{road}_i^j}\mathbf{H}_{\mathsf{odom}} \cdot {}^{\mathsf{odom}}\mathbf{p}_i^{(k)}\,, \tag{8.38} $$

$$ {}^{\mathsf{road}_i^j}\mathbf{H}_{\mathsf{odom}} = \underline{\mathbf{T}}\left(\frac{{}^{\mathsf{odom}}\mathbf{p}_i^{(k_{\mathsf{min}})^j} + {}^{\mathsf{odom}}\mathbf{p}_i^{(k_{\mathsf{max}})^j}}{2}\right) \cdot \underline{\mathbf{R}}\left(\frac{{}^{\mathsf{odom}}\mathbf{q}_i^{(k_{\mathsf{min}})^j} + {}^{\mathsf{odom}}\mathbf{q}_i^{(k_{\mathsf{max}})^j}}{2}\right). \tag{8.39} $$

Damit wird dann für jeden Punkt $\mathbf{x}_i^{(k)}$ und jede Messung j das Residuum

$$ \mathbf{r}_{\mathsf{lidar},j,[1]}^{(k)}(\mathbf{x}_i) = \exp\left[\left({}^{\mathsf{road}_i^j}y_i^{(k)} - y_{\mathsf{max},i}^j\right) \cdot \mathbf{w}_{[\mathsf{lidar}]}\right], \tag{8.40} $$

$$ \mathbf{r}_{\mathsf{lidar},j,[2]}^{(k)}(\mathbf{x}_i) = \exp\left[\left(y_{\mathsf{min},i}^j - {}^{\mathsf{road}_i^j}y_i^{(k)}\right) \cdot \mathbf{w}_{[\mathsf{lidar}]}\right] \tag{8.41} $$

ausgewertet, um die y-Koordinaten ${}^{\mathsf{road}_i^j}y_i^{(k)}$ des Punkts $\mathbf{x}_i^{(k)}$ in $\mathbf{F}_{\mathsf{road},i}^j$ zu beschränken. Dabei wird anstelle eines harten Parameter-Limits die in Gleichung (8.18) beschriebene exponentielle Limitierungs-Funktion verwendet, da die Messungen aufgrund von Ausreißern oder Verdeckungen in der LiDAR-Punktwolke fehlerhaft sein können. In Abbildung 8.18 wird beispielsweise in der Ferne der rechte Rand der Bodenfläche fälschlicherweise zu weit innen gemessen, da keine direkte Sichtverbindung herrscht. Dieser Fall wird durch die kontinuierliche Residuums-Funktion aufgefangen, die bei entsprechend starken anderweitigen Indikationen eine Überschreitung des Limits erlaubt. Zudem wird für eine erhöhte Robustheit gegen Ausreißer die Huber-Verlustfunktion $\rho_{\mathsf{H}}(\cdot)$ eingesetzt (vgl. Gleichung (8.28)), die mit steigenden Kosten (also steigender Abweichung der Straßenschätzung vom LiDAR-Range-Bereich) diese nicht mehr quadratisch, sondern nur noch linear gewichtet. Der konstante Gewichtungsfaktor der LiDAR-Range-Messung wird auf $\mathbf{w}_{[\mathsf{lidar}]} = 1{,}5$ gesetzt.

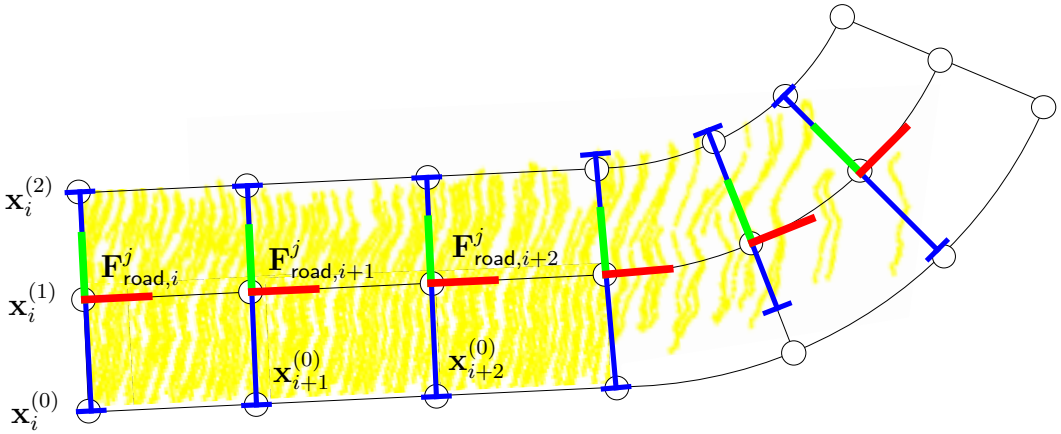
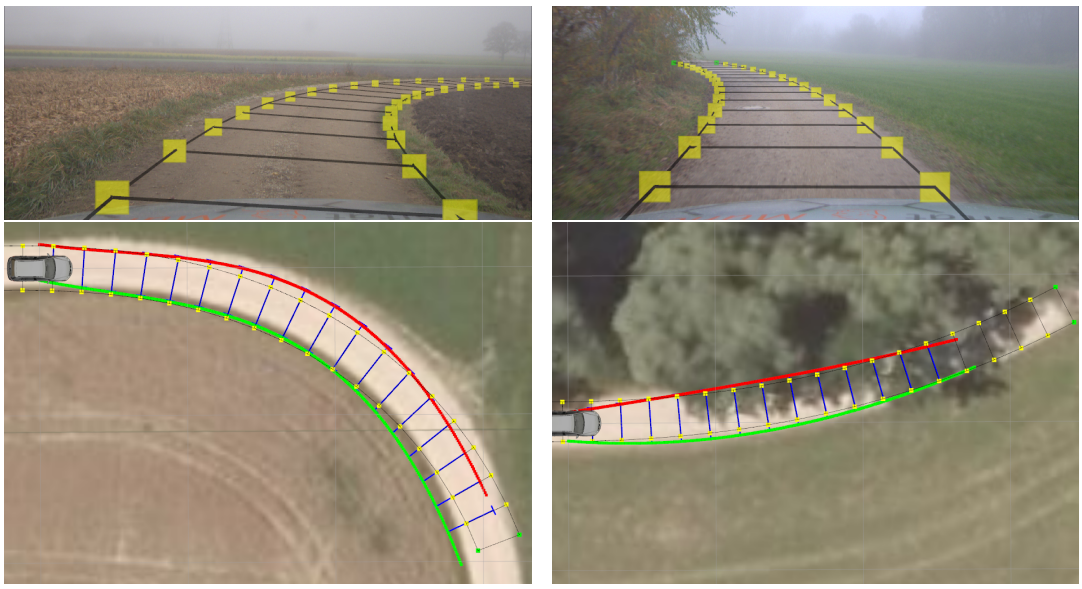


Abbildung 8.19:
Visualisierung einer LiDAR-Range-Messung: Aus der akkumulierten Punktwolke (gelb) werden obere und untere Grenzen (blau) der y-Koordinate im jeweiligen Koordinatensystem $\mathbf{F}^j_{\text{road},i}$ gewonnen.



(a) (b)

Abbildung 8.20:
Zwei Beispiele von Feldwegen, die nur mittels der LiDAR-Range-Messungen und der Terrain-Schätzung (ohne Bildmessungen oder Hindernis-Grid) erfolgreich getrackt werden. Die gelben Punkte zeigen die geschätzten 3D-Spurbegrenzungspunkte in das Kamerabild projiziert bzw. auf einem Luftbild.

Die LiDAR-Range-Messung funktioniert auf Straßen und Feldwegen gleichermaßen und könnte auch als einzige Messung verwendet werden – Abbildung 8.20 zeigt zwei Beispiele von Feldwegen, die nur mit der LiDAR-Range-Messung erfolgreich getrackt werden. In Kombination mit den im vorherigen Abschnitt beschriebenen Bildmessungen erhöht die LiDAR-Range-Messung die Robustheit des Trackings auf schwierigen Wegen oder bei schwierigen Sichtverhältnissen.

8.6 Fusion von Kamera, LiDAR und Modellwissen

Die Fusion aller vorgestellten Messverfahren – der zwei Bildmessungen, der Informationen aus dem Hindernis-Grid und der Terrain-Schätzung und der LiDAR-Range-Messung – sowie des Modellwissens erfolgt implizit durch die Lösung des Optimierungsproblems in Gleichung (8.2). Durch die Minimierung der Summe aller Kostenfunktionen wird der global optimale Zustandsvektor mit den kleinsten Gesamtkosten gesucht. Diese beinhalten zum einen die Summe aller Kosten durch die Abweichung der Schätzung von den verschiedenen Messungen, zum anderen die Summe aller Kosten durch die Abweichung der Schätzung vom Klothoidenmodell und vom Vorwissen über mögliche Wertebereiche der Zustandsparameter. Abbildung 8.21 gibt einen zusammenfassenden Überblick über alle in den Gesamtkosten enthaltenen Residuen. Je nach Gewichtung $\mathbf{w}$ der einzelnen Residuen, die empirisch so bestimmt wurde, dass möglichst viele unterschiedliche Straßen und Wege mit denselben Parametern robust getrackt werden können, werden durch die einzelnen Messungen und Modellannahmen unterschiedlich starke Gradienten erzeugt.

Der kostenoptimale Zustandsvektor stellt einen entsprechend gewichteten Kompromiss dar: Fließt die Bildmessung aufgrund ihrer hohen Präzision mit hoher Gewichtung ein und existieren für einen Wegrand viele eindeutige Bildmessungen, so erzeugt beispielsweise die Schätzung der Straße in Abbildung 8.18 außerhalb des LiDAR-Range-Bereichs geringere Kosten als erzeugt würden, wenn die Schätzung von der stärker gewichteten Bildmessung wegbewegt werden müsste. Die Schätzung folgt hier also der Bildmessung entlang der Straße, obwohl die LiDAR-Range-Messung auf eine schmälere Straße hindeuten würde. Existieren jedoch keine Bildmessungen oder sind die Bildmessungen mehrdeutig wo sich der Wegrand befinden könnte, so kann die LiDAR-Range-Messung über den zusätzlichen (schwächeren) Gradienten einen Hinweis auf die richtige Wegschätzung liefern. Das Hindernis-Grid verursacht auf der anderen Seite hohe Kosten für Wegschätzungen, die statische Hindernisse enthalten, sodass diese Hypothesen nahezu ausgeschlossen werden – außer die anderen Messungen und die Modellannahmen schließen andere Hypothesen durch noch höhere Kosten aus. Befindet sich dagegen kein Hindernis auf dem Weg, so erzeugt das Hindernis-Grid keinerlei Gradienten, und es wird anderen Messungen gefolgt.

Außer von der Gewichtung (und der Maßeinheit) hängt die Präzedenz der Messverfahren und Modellannahmen auch von der Anzahl der Spurbegrenzungspunkte, der Anzahl der Messungen sowie der Passung der Messungen ab. Durch die Huber-Verlustfunktion steigen die Kosten mit steigender Abweichung der Schätzung von den Messungen weniger stark an, sodass sich die Gewichtungsverhältnisse ändern.

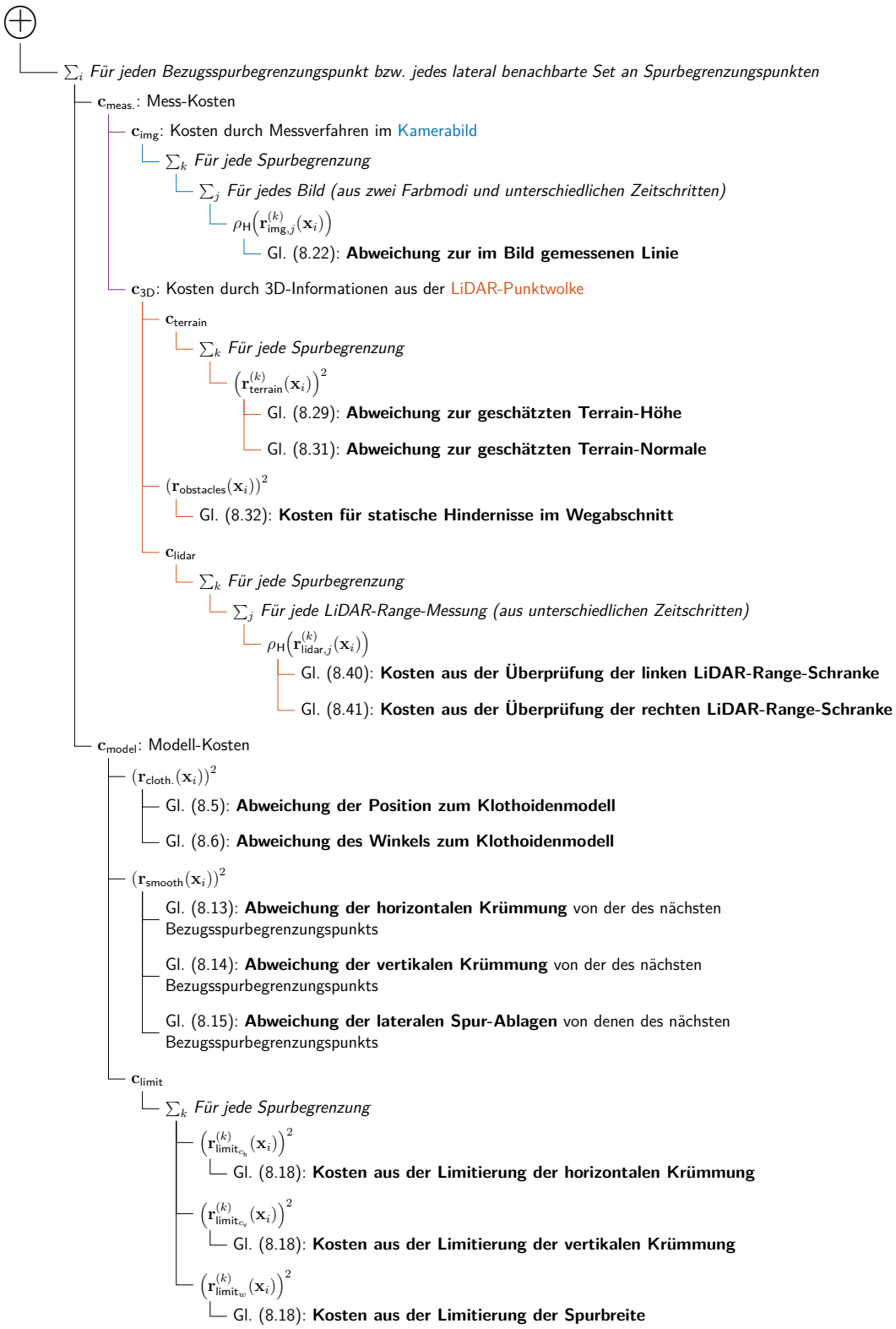


Abbildung 8.21:
Überblick über die Zusammensetzung der im Optimierungsproblem zum Straßen- und Wegetracking aufgestellten Kostenfunktion. Der optimale Zustandsvektor minimiert die Summe der Gesamtkosten aller Residuen.

8.7 Ergebnisse

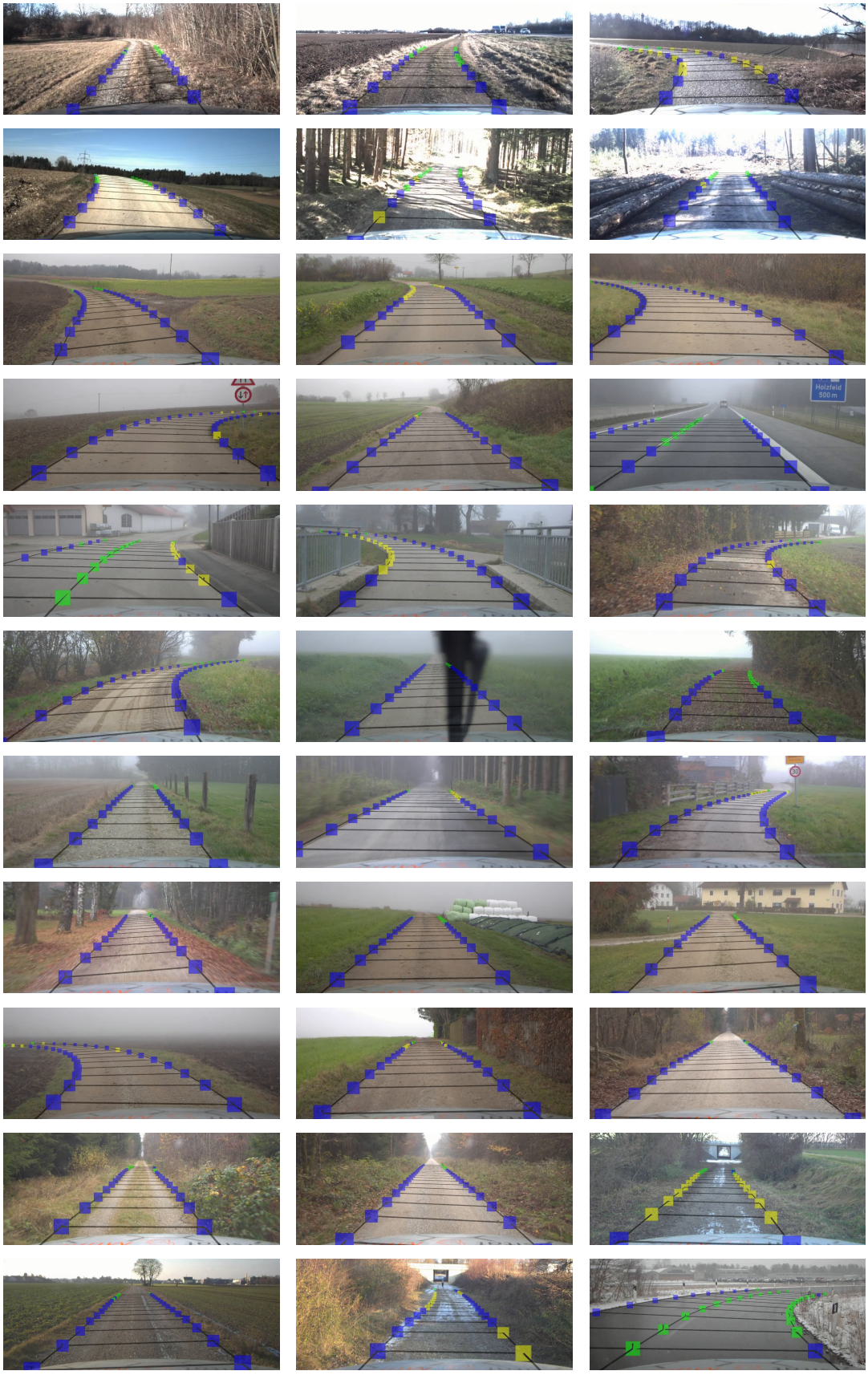
Leider existieren keine öffentlichen Datensätze, die 3D-Informationen von Fahrspurrändern bereitstellen, mit denen die durch die Fusion von Kamera und LiDAR erreichte Robustheit des vorgestellten Straßen- und Wegetrackings quantitativ belegt werden könnte. Auch eine händische Annotation im Kamerabild hätte wenig Aussagekraft bei der Beurteilung der geschätzten 6D-Posen. Deshalb werden an dieser Stelle qualitative Ergebnisse gezeigt.

8.7.1 Qualitative Ergebnisse

Zur Beurteilung der Robustheit des entwickelten Straßen- und Wegetrackings gegenüber verschiedenen Arten und Verläufen von Straßen und Wegen, unterschiedlichen Umgebungen, Untergründen und Wetterbedingungen wurde das Straßentracking mit den institutseigenen Versuchsträgern auf verschiedensten, teils sehr herausfordernden Straßen und Feldwegen in der echten Welt getestet. Abbildung 8.22 zeigt einige Beispiele. Diese beinhalten ein- bis dreispurige markierte und unmarkierte Straßen, Feld- und Waldwege, mit Steigungen, bei Tag, mit Gegenlicht, im Dunkeln, bei Regen, bei Schnee und bei Nebel. In allen Situationen konnten die sehr verschiedenen, meist unbekannten Straßen und Wege ohne Kartenwissen mit denselben, in dieser Arbeit vorgestellten Messverfahren ohne weiteres Tuning robust getrackt werden. Dies ist vor allem der Unabhängigkeit des Verfahrens von Trainingsdaten zu verdanken.

Abbildung 8.22 zeigt nur Momentaufnahmen. Zur Beurteilung der andauernden Robustheit des entwickelten Straßen- und Wegetrackings zeigt Abbildung 8.23 die Ergebnisse einer auf dem Wegetracking basierenden, autonomen Fahrt mit MuCAR-4. Hierfür wurde mittels des Pfadfolgereglers von Bienemann und Wuensche [2023] die Mitte des durch das Wegetracking live geschätzten Wegs verfolgt. Auf einer 20-minütigen, autonomen Fahrt von 5,1 km Länge, über geteerten Weg, über Feldwege und durch den Wald, bei Einbruch der Dunkelheit sowie teilweise mit Schnee auf der Strecke – Abbildung 8.24 zeigt Beispielbilder – erfolgten insgesamt nur fünf Eingriffe durch den Sicherheitsfahrer in Form einer Neuinitialisierung des Wegetrackings. Dabei waren zwei der Eingriffe lediglich durch die Wahl einer falschen Abzweigung an Kreuzungen bedingt. Die längste zwischen zwei Eingriffen gefahrene Strecke beträgt 1,8 km. Damit erweist sich das vorgestellte Wegetracking als auch bei sehr schwierigen Bedingungen auf dem autonomen Fahrzeug einsetzbar.

Ein weiteres qualitatives Beispiel ist in Abbildung 8.25 zu sehen. Um (im Gegensatz zu Abbildung 8.24) die Visualisierung des Trackingergebnisses und die Evaluation der Laufzeiten zu ermöglichen, wurde dieses Beispiel bei manueller Fahrt aufgenommen und das Wegetracking im Nachhinein durchgeführt. Dabei konnte der 1,5 km lange Feldweg ohne Trackingabriss getrackt werden. Weitere qualitative Eindrücke sind in einem unter `https://www.youtube.com/watch?v=criverr4igQ` verfügbaren Video zu sehen. Es zeigt Bewegtbilder der geschätzten Straßen und Wege und visualisiert die Ergebnisse der einzelnen Messverfahren in verschiedenen kurzen Beispielszenen sowie auch in längeren, ungeschnittenen Szenen.



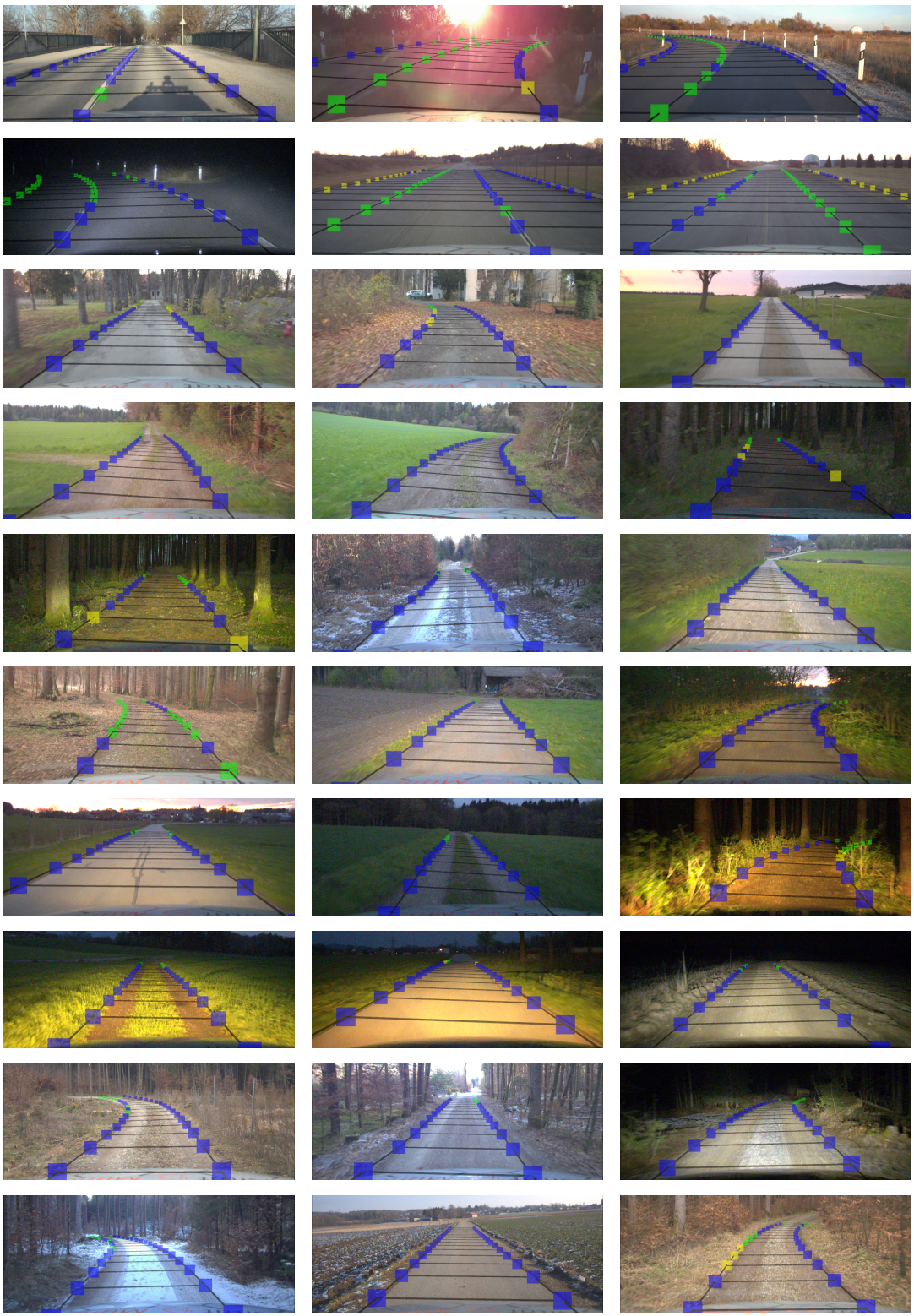


Abbildung 8.22:
Beispiele für verschiedene Straßen und Feldwege, die mit dem vorgestellten Straßen- und Wegetracking erfolgreich getrackt wurden. Die Bilder zeigen die in das Kamerabild projizierten geschätzten Spurbegrenzungspunkte. Punkte, die in den letzten 3 s im Kamerabild gemessen werden konnten, werden in blau dargestellt. Die gelben Punkte konnten nur über die LiDAR-Range-Messungen gemessen werden. Für die grünen Punkte existiert (noch) keine direkte Messung.

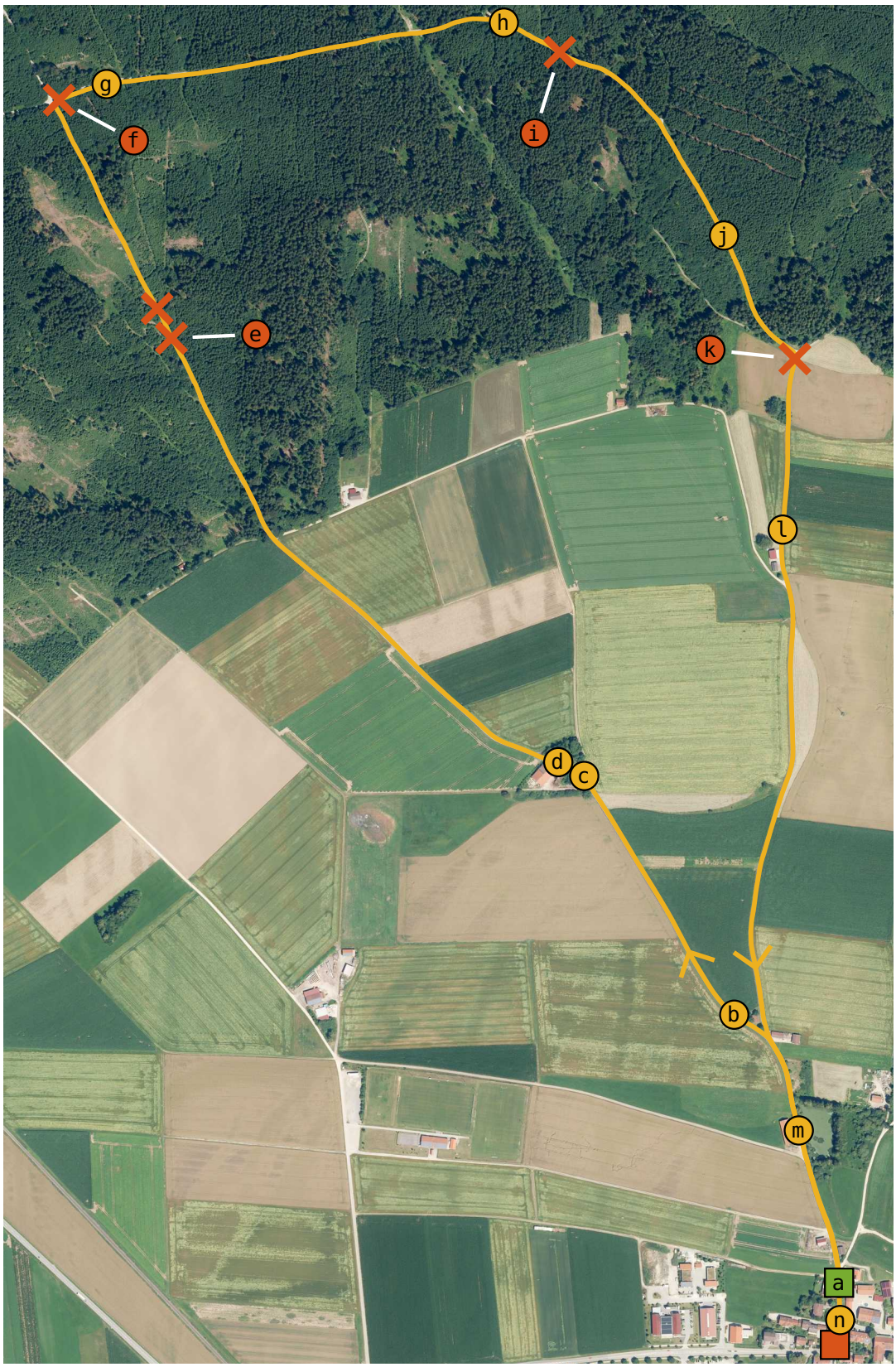


Abbildung 8.23:
Darstellung einer von MuCAR-4 basierend auf dem vorgestellten Wegetracking autonom gefahrenen Strecke (gelb) bei Aying (Landkreis München). Start- und Zielpunkt werden durch das grüne und das rote Quadrat visualisiert. Die roten Kreuze zeigen alle Stellen, an denen ein manueller Eingriff durch den Sicherheitsfahrer vorgenommen wurde. Die Kreise bzw. Buchstaben markieren die Position der in Abbildung 8.24 gezeigten Bildausschnitte.



(a) (b) (c) (d) (e) Eingriff (f) Eingriff an Kreuzung (g) (h) (i) Eingriff (j) (k) Eingriff an Kreuzung (l) (m) (n)

Abbildung 8.24:
Bildausschnitte der in Abbildung 8.23 gezeigten autonomen Fahrt anhand des Wegetrackings.

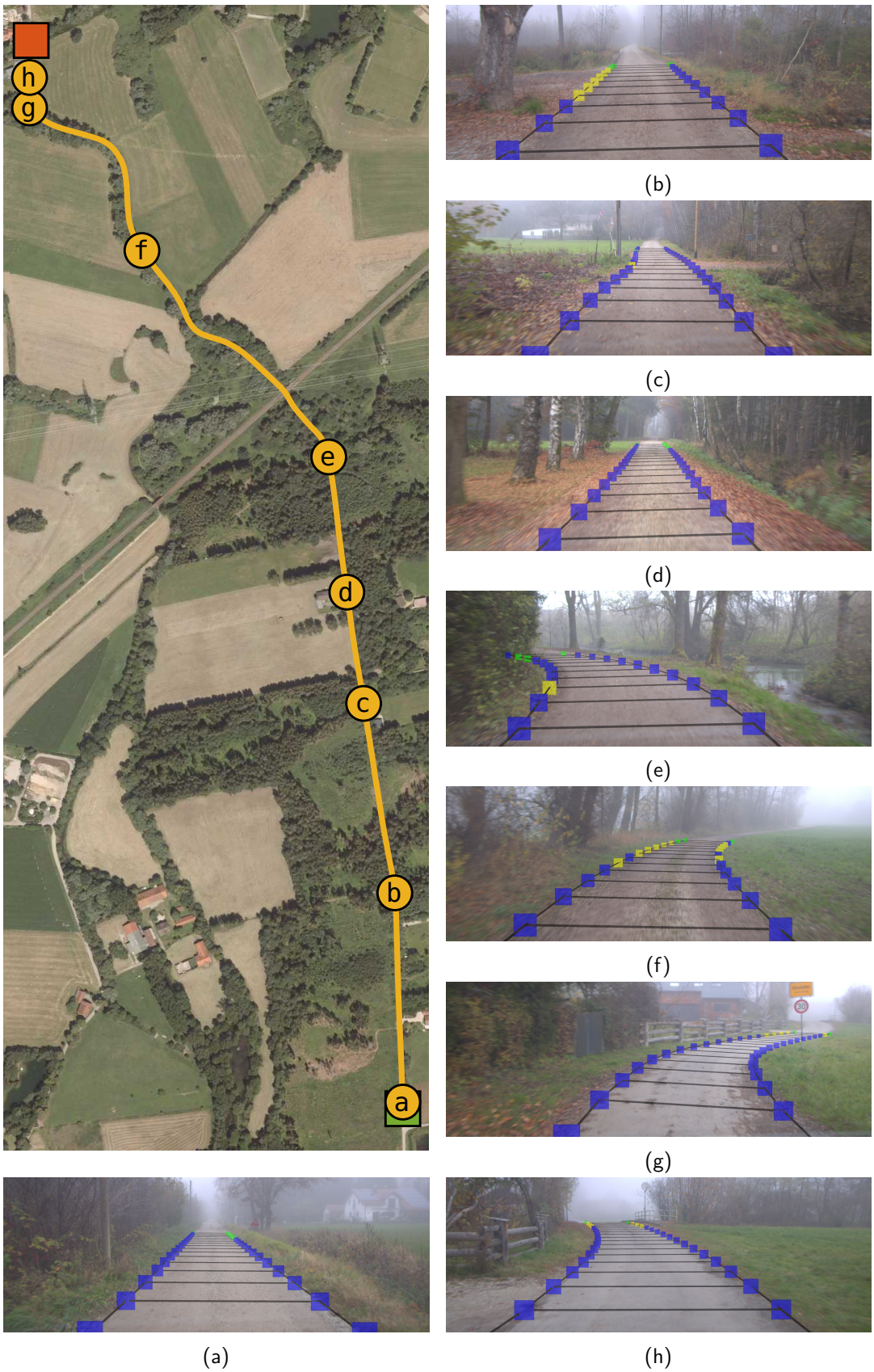


Abbildung 8.25:
Beispiel eines erfolgreich getrackten Wegs bei Ottenhofen (Landkreis Erding). Die gelben Kreise markieren die gezeigten Bildausschnitte. Start- und Zielpunkt (der manuellen Fahrt) werden durch das grüne und das rote Quadrat visualisiert.



Abbildung 8.26:
Beispiel einer markierten Straße, die erfolgreich getrackt wird, obwohl der Versuchsträger etwa 5 m abseits der Straße fährt.

Durch die generische Modellierung der Straße und die robusten Messverfahren kann das Straßen- und Wegetracking nicht nur zur Pfadverfolgung genutzt werden. Es können auch Straßen und Wege getrackt werden, auf denen sich das Fahrzeug nicht selbst befindet, wie das Beispiel in Abbildung 8.26 zeigt.

8.7.2 Laufzeiten und Vorausschauweite

Für den Einsatz des Wegetrackings auf dem Versuchsträger ist zuletzt die Echtzeitfähigkeit des Verfahrens von Bedeutung. Abbildung 8.27a zeigt die Rechenzeiten des Straßen- und Wegetracking-Systems in dem in Abbildung 8.25 gezeigten Szenario. Die CRONOS-Kantenmessung benötigt im Median nur 9 ms pro Kamerabild und Wegschätzung. Dadurch kann die Messung problemlos mit beiden vorgestellten Farbmodi parallel durchgeführt werden. Die LiDAR-Range-Messungen benötigen im Median sogar nur 4 ms. Aufgrund dieser schnellen Messverfahren kann die Messung

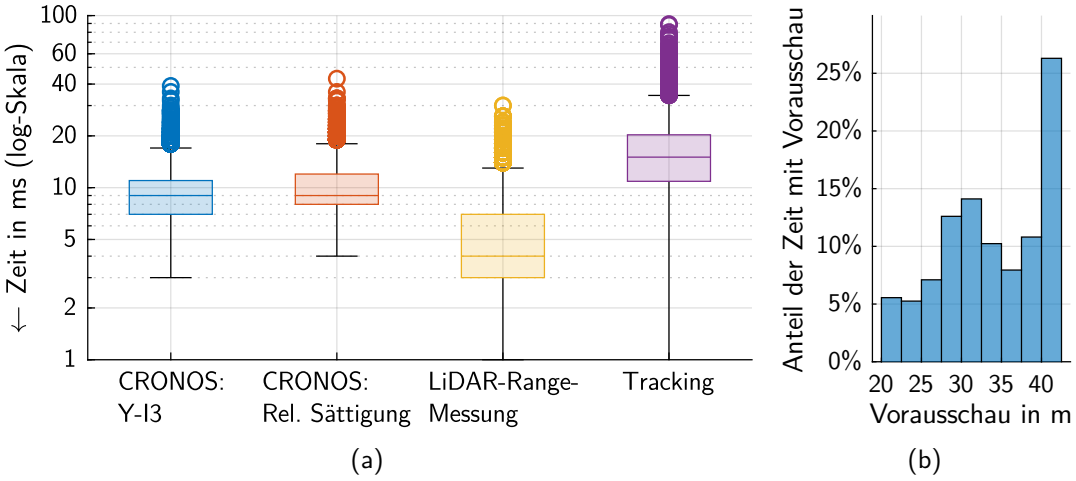


Abbildung 8.27:
(a) Rechenzeiten und (b) Verteilung der Vorausschauweiten des Wegetrackings im Feldweg-Szenario aus Abbildung 8.25.

selbst ohne neue Eingangsdaten mehrmals mit einer korrigierten Wegschätzung als Vorwissen wiederholt und die Schätzung kontinuierlich verbessert werden. Das Schätzverfahren mittels nichtlinearer Optimierung benötigt dabei im Median 15 ms. Die Rechenzeit hat jedoch einzelne Ausreißer bis hin zu 100 ms, nach denen die Optimierung abgebrochen und das letzte Zwischenergebnis verwendet wird. Dank der meist schnellen Konvergenz und der robusten Messverfahren wird, wie Abbildung 8.27b zeigt, zu über 25 % der Zeit die höchstmögliche Vorausschau von 40 m bis 42,5 m vor dem Ego-Fahrzeug erreicht. Lediglich in gut 5 % der Zeit wird nur die minimal gewährleistete Vorausschau von 20 m erzielt.

8.7.3 Grenzen des Verfahrens

Ein Nachteil des Verfahrens ist es, dass keine Unsicherheit der geschätzten Spurbegrenzungspunkte geschätzt wird. Würden nur Bildmessungen und die Terrain-Schätzung berücksichtigt, so könnte über eine Sensitivitätsanalyse in der Optimierung eine statistisch korrekte Unsicherheit ermittelt werden. Aufgrund der vielen zusätzlichen Modellannahmen und Beschränkungen, für die nur empirische Gewichte festgelegt wurden, kann jedoch keine physikalisch begründete Messunsicherheit erhalten werden. Auch die indirekten Messungen aus dem Hindernis-Grid, den Terrain-Normalen oder den LiDAR-Range-Messungen erlauben keine direkte Ableitung einer Unsicherheit.

Das Verfahren stößt zudem an seine Grenzen, wenn das zugrundeliegende Modell der Straße nicht zutrifft und die Straße nicht aus weitgehend parallelen Spurbegrenzungen und direkt benachbarten Fahrstreifen besteht. Dies kommt vor allem bei Straßen im urbanen Umfeld vor. Eine Verkehrsinsel zwischen zwei Spuren, wie in Abbildung 8.28a dargestellt, oder eine größere Freifläche, wie in Abbildung 8.28b gezeigt, könnten beispielsweise nicht modelliert werden.



(a) Verkehrsinsel



(b) Freifläche



(c) Reifenspuren im Gras



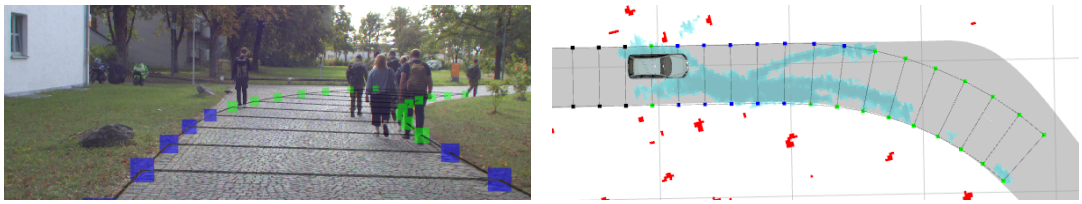
(d) Verdeckung durch andere Verkehrsteilnehmer

Abbildung 8.28:
Beispiele von Straßen und Wegen, für die mit dem entwickelten Straßen- und Wegetracking keine korrekte Schätzung der Spurbegrenzungen erhalten werden kann.

Für ein präzises Tracking sollten außerdem entweder im Kamerabild oder in der LiDAR-Punktwolke zumindest schwache Kanten messbar sein. So könnte beispielsweise ein Weg, der nur durch niedergefahrenes Gras definiert ist, wie das Beispiel in Abbildung 8.28c, nicht robust wahrgenommen werden. Auch eine starke Verdeckung kann zu Problemen führen. Diese tritt hauptsächlich auf, wenn sich sehr viele andere Verkehrsteilnehmer auf dem Weg befinden. Ein Beispiel hierfür ist in Abbildung 8.28d zu sehen: Die vielen Fußgänger auf dem gezeigten Campus-Weg verdecken die entscheidenden Kanten, sodass die Kurve nicht erkannt werden könnte.

8.8 Ausblick: Straßentracking auf Mischverkehrswegen mittels Radar-Sensorik

Um das letztgenannte Problem zu behandeln, wurde in der Fortführung dieser Arbeit ein zusätzliches Messverfahren für das Wegetracking auf Mischverkehrswegen entwickelt [Forkel et al., 2024]. Dieses macht sich die Anwesenheit der anderen Verkehrsteilnehmer sogar zunutze: Dort wo sich andere Verkehrsteilnehmer bewegen, befindet sich mit großer Wahrscheinlichkeit auch eine Straße. Daher werden auf der Grundlage von durch Radar-Sensoren (*Radio Detection And Ranging*) gewonnenen Doppler-Messungen gezielt dynamische Objekte identifiziert. Unter Berücksichtigung der Messunsicherheiten wird dann eine Karte wahrscheinlicher Pfade von Verkehrsteilnehmern akkumuliert und als wahrscheinlicher Straßenbereich in das Wegetracking fusioniert. Damit kann, wie in Abbildung 8.29 zu sehen ist, die verdeckte Kurve aus Abbildung 8.28d erfolgreich getrackt werden.



(a) (b)

Abbildung 8.29:
Erfolgreiches Tracking des in Abbildung 8.28d gezeigten Wegs durch die Fusion von Fußgängerdetektionen in das Wegetracking. Obwohl die Fußgänger den Weg im Kamerabild verdecken, kann das Wegtracking mit dem neuen Messverfahren für Mischverkehrswege die Kurve tracken. (a) In das Kamerabild projizierte, geschätzte Spurbegrenzungspunkte. Für die grünen Punkte konnte innerhalb der letzten 3 s keine Bildmessung extrahiert werden. (b) Draufsicht. Mithilfe der akkumulierten Fußgängerwahrscheinlichkeiten (cyan) kann das Wegetracking die Kurve einleiten. Die geschätzte Straßenbreite nach der Kurve wird korrigiert, sobald wieder Kanten gemessen werden. Die graue Fläche zeigt die Ground-Truth-Straße aus einer hochgenauen Karte. In rot sind statische Hindernisse dargestellt.

Von diesem damit trackbaren Weg auf dem Campus der Universität der Bundeswehr München (UniBwM) existiert eine hochgenaue Karte von 3D Mapping Solutions GmbH [2024]. Damit ist zum ersten Mal eine quantitative Auswertung des Wegetrackings möglich. Für den rechten Wegrand wird dabei über die gesamten 160 m Länge des Wegs ein Mittlerer absoluter Fehler, engl. *Mean Absolute Error* (MAE)

zur Karte von nur 7 cm erzielt. Dies liegt an der Grenze der bei der quantitativen Auswertung erzielbaren Genauigkeit: Sowohl die Generierung der Karte als auch die Auswertung des Wegetrackings in globalen Weltkoordinaten unterliegen Lokalisierungs- und Sensorkalibrierungsfehlern. Da der Weg in der in Abbildung 8.29 gezeigten Kurve nicht den Modellannahmen annähernd paralleler Wegränder entspricht, weicht die Schätzung des linken Wegrands stärker von der Karte ab. Hier beträgt der MAE 18 cm. Da das neue Radar-basierte Messverfahren nur zur Stabilisierung bei Verdeckung dient und die präzise Lokalisierung der Wegränder weiterhin aus den in dieser Arbeit entwickelten Bildmessungen kommt, zeigt diese Auswertung das Potenzial des in diesem Kapitel entwickelten Wegetrackings.

8.9 Diskussion

Mit dem vorgestellten Framework zum Straßen- und Wegetracking und den verschiedenen Messverfahren in Kamerabild und LiDAR-Punktwolke kann eine Vielzahl an markierten und unmarkierten Straßen, Feld- und Waldwegen ohne Kartenwissen bei allen Wetterbedingungen robust getrackt werden. Damit konnte einem Feldweg 1,8 km am Stück autonom gefolgt werden.

Eine Einschränkung, die die maximal mit dem Wegetracking fahrbare Strecke beschränkt, und die bei dieser Fahrt zwei Eingriffe erforderte, ist jedoch das Verhalten an Kreuzungen. Das Wegetracking kann keine Abzweigungen wahrnehmen und folgt daher beispielsweise in Abbildung 8.30a stattdessen weiter dem geradeaus führenden Weg. Dadurch ist es nicht möglich, diesen Hauptweg ohne einen manuellen Eingriff zu verlassen. Nach einem manuellen Abbiegevorgang muss das Tracking auf dem neuen Weg neu aufgesetzt werden. Bei Kreuzungen, die keinen eindeutigen Hauptweg aufweisen, wie dem Beispiel in Abbildung 8.30b, ist nicht vorhersagbar, welchen Weg das Wegetracking tracken wird. Eine gezielte Navigation in eine Richtung ist nicht möglich. In einzelnen Fällen kann das Tracking an dieser Stelle sogar abreißen. Damit ist das Wegetracking alleine nicht für größere Missionen geeignet, die mehr als einen Abschnitt eines Feldwegs umfassen oder über eine Verhaltenssteuerung ein bestimmtes Ziel anfahren sollen. Hierfür wird ein zusätzliches Modul zur Kreuzungserkennung benötigt, das daher im folgenden Kapitel entwickelt wird.



(a) Abzweigung (b) Kreuzung

Abbildung 8.30:
Beispiele für das Verhalten des vorgestellten Wegetrackings bei Kreuzungen. Kreuzungen werden nicht explizit behandelt, sondern es wird dem Hauptweg oder alternativ einem zufälligen Weg entsprechend der Limitierungen gefolgt.

9 Kreuzungserkennung

Für die vollständige Wahrnehmung der statischen Umgebung auf einem Feldweg wurde abschließend eine Erkennung sowie ein Tracking von Wegkreuzungen implementiert. Dazu wurden bestehende Arbeiten am Institut für das in den vorhergehenden Kapiteln vorgestellte neue Umgebungsmodell portiert und mit den in dieser Arbeit gewonnenen Erkenntnissen erweitert. Die Kreuzungserkennung ermöglicht die explizite Wahrnehmung von Feldwegkreuzungen ohne vorheriges Kartenwissen anhand von Kamera- und LiDAR-Messungen und zeigt dabei eine zweite Anwendungsmöglichkeit der im Wegetracking vorgestellten Farbmodi.

9.1 Problemstellung

Im Straßen- und Wegetracking basieren die primären Messungen der Wegränder auf Farbinformationen aus dem Kamerabild. Für eine weitere Vorausschau wurden die Messungen direkt aus dem Kamerabild extrahiert anstatt wie bisher beim Institut für Technik Autonomer Systeme (TAS) aus einem akkumulierten, eingefärbten 2½D-Grid [Manz et al., 2011, Bayerl et al., 2015]. Wie das Beispiel in Abbildung 9.1a zeigt, sind Kreuzungen im Kamerabild jedoch nur sehr schwer erkennbar: Die einzelnen Äste der Kreuzung werden in den verwendeten, eher weitwinkligen Kameras mit Öffnungswinkeln von mehr als 55° (vgl. Abschnitt 2.1) nur durch wenige Pixel abgebildet, die keine direkte Messung der Wegränder im Kamerabild erlauben. Daher muss für die Erkennung von Kreuzungen erneut auf eine Draufsicht zurückgegriffen werden. Dazu kommt das eingefärbte Terrain-Grid aus Abschnitt 7.3.5 zum Einsatz. Dort wurde das Terrain-Grid aus der LiDAR-basierten Terrain-Schätzung durch Projektion der einzelnen Grid-Zellen in das Kamerabild mit RGB-Farben eingefärbt. In dieser Fusion aus räumlichen LiDAR-Informationen und farblichen Bildinformationen sind Kreuzungen, wie in Abbildung 9.1b zu sehen ist, deutlich besser zu erkennen.



(a) Kamerabild (b) Farbiges Terrain-Grid

Abbildung 9.1:
Beispiel derselben Kreuzung (zum selben Zeitpunkt) im Kamerabild und im eingefärbten Terrain-Grid (Draufsicht). Durch die räumliche Information ist die Kreuzung in der Draufsicht besser zu erkennen und kann erfolgreich detektiert werden (blau eingezeichnet).

Für die Detektion von Kreuzungen in einem eingefärbten Grid existieren am Institut TAS bereits bewährte Verfahren. Daher wird in dieser Arbeit keine neue Kreuzungserkennung entwickelt. Stattdessen wurde eine bestehende Kreuzungserkennung für die Verwendung des in dieser Arbeit entwickelten Umgebungsmodells portiert.

Forschungsfrage Die verwendete Kreuzungserkennung von Bayerl und Wuensche [2014] basiert auf der Segmentierung von Flächen, die farblich dem aktuellen Weg ähneln. Dafür kommen verschiedene Farbmodi zum Einsatz. In Abschnitt 8.4.3 wurde bereits geprüft, ob sich diese Farbmodi zur Kantenerkennung im Wegetracking eignen. Dabei wurden jedoch robustere Farbmodi identifiziert. Daher gilt es abschließend zu untersuchen, ob diese im Wegetracking entwickelten Farbmodi auch in der Kreuzungserkennung vorteilhaft sein können.

Gliederung Abschnitt 9.2 beschreibt den Stand der Forschung zur Erkennung von Feldwegkreuzungen. Anschließend wird in Abschnitt 9.3 das Konzept der verwendeten Kreuzungserkennung von Bayerl und Wuensche [2014] vorgestellt. In Abschnitt 9.4 werden dann die im Wegetracking entwickelten Farbmodi in die Kreuzungserkennung integriert und mit den ursprünglich verwendeten Farbmodi verglichen. Abschnitt 9.5 zeigt schließlich Ergebnisse dieser weiterentwickelten Kreuzungserkennung.

9.2 Stand der Forschung

Die Kreuzungserkennung von Bayerl und Wuensche [2014] baut wiederum auf den Vorarbeiten von Manz et al. [2011] (bzw. [Manz, 2013]) auf. Aus einem mit dem Kamerabild eingefärbten Geländemodell werden verschiedene Wegmerkmale extrahiert, mittels derer ein Kreuzungsmittelpunkt und davon abzweigende Kreuzungsäste erkannt werden sollen. Während die Kreuzungsäste bei Manz [2013] als Klothoiden modelliert werden, vereinfachen Bayerl und Wuensche [2014] diese zu Geradenstücken. Ferner ergänzen Bayerl und Wuensche [2014] zwei flächenbasierte Farbmerkmale von Manz [2013] um ein neues, den Grünanteil berücksichtigendes Merkmal (vgl. Abschnitt 9.4), verzichten dafür jedoch auf die von Manz [2013] vorgeschlagenen kantenbasierten Merkmale. Anhand dieser Wegmerkmale im eingefärbten Geländemodell erkennen und tracken Manz et al. [2011] Feldwegkreuzungen mittels eines Partikelfilters. Bayerl und Wuensche [2014] stellen dagegen durch räumliches Abtasten Kreuzungshypothesen auf, überprüfen diese anhand der Wegmerkmale und filtern die Messungen anschließend mittels eines Kalman-Filters.

Ein anderer Ansatz zur Kreuzungserkennung wurde bei TAS von Mueller et al. [2011] verfolgt: Hier werden Kreuzungen anhand von Freiräumen in einem durch eine LiDAR-Punktwolke befüllten Hindernis-Grid erkannt. Da keine zusätzlichen Farb- oder Kanteninformationen beispielsweise aus einem Kamerabild genutzt werden, ist damit jedoch keine Erkennung von Wegen auf offener Fläche möglich. Dies gilt für alle ausschließlich LiDAR-basierten Ansätze zur Kreuzungserkennung, die beispielsweise für Anwendungen in Minen [Morris et al., 2005] oder auch in Innenstädten [Zhao et al., 2022] entwickelt wurden.

Außerhalb des Instituts TAS gibt es (Stand Juni 2024) keine veröffentlichten Arbeiten zur Erkennung von Feldwegkreuzungen ohne Kartenwissen, da das Hauptaugenmerk in der Forschung zum autonomen Fahren auf größeren Straßen liegt. Während frühe Arbeiten zur Erkennung von Straßenkreuzungen, beispielsweise von Waxman et al. [1987], Kushner und Puri [1987] oder Crisman und Thorpe [1993], noch ähnliche Ansätze verfolgten Kreuzungen anhand von Farben und Kanten in Kamerabildern zu erkennen, kann bei Straßenkreuzungen heute jedoch auf Kartendaten zurückgegriffen werden. Bereits bei der DARPA Urban Challenge 2007 implementierten die Teams keine explizite Kreuzungserkennung mehr [Urmson et al., 2008, Bacha et al., 2008, Montemerlo et al., 2008, Kammel et al., 2008]. Daher kann hier nicht mehr auf Innovationen aus dem urbanen Verkehr zurückgegriffen werden.

9.3 Konzept

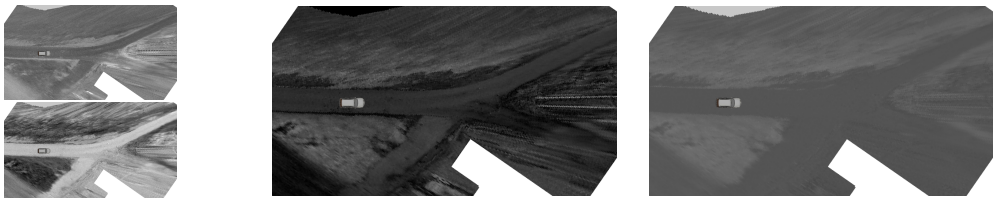
Die von Bayerl und Wuensche [2014] entwickelte Kreuzungserkennung besteht aus zwei Teilen: Zuerst wird aus dem eingefärbten Terrain-Grid für jede Grid-Zelle eine sog. Straßenwahrscheinlichkeit bestimmt. In diesem Straßenwahrscheinlichkeits-Grid werden dann Wege gesucht, die sich in einem Kreuzungsmittelpunkt schneiden, und Kreuzungshypothesen über die Zeit getrackt.

Abbildung 9.2 zeigt die einzelnen Schritte zur Berechnung einer Straßenwahrscheinlichkeit. Unter der Annahme, dass sich das Ego-Fahrzeug bereits auf einem Weg befindet, werden Bereiche gesucht, die farblich ähnlich zu diesem Weg sind, um den weiteren Verlauf des Wegs sowie eventuell kreuzende Wege zu detektieren. Dafür wird für jede Grid-Zelle ein Ähnlichkeitsscore zu einer Region von Interesse, engl. *Region of Interest* (ROI) unter dem Fahrzeug bestimmt. Für den Vergleich kommen drei Farbmodi zum Einsatz. Im ersten Farbmodus wird direkt die Farbe verglichen. Um die Farbe unabhängig von der lokalen Helligkeit vergleichen zu können und damit robust gegenüber Beleuchtungsänderungen beispielsweise durch Schatten zu sein, wird jedoch nicht die RGB-Farbe verglichen, sondern stattdessen – wie von Apostoloff und Zelinsky [2003] zum Tracking unmarkierter Straßen vorgeschlagen – das YUV-Farbmodell verwendet. Im YUV-Farbmodell wird die Helligkeit (Luminanz) $\mathbf{Y}_{\mathrm{YUV}}$ von der Farbinformation (Chrominanz), die aus den Farbanteilen $\mathbf{U}_{\mathrm{YUV}}$ und $\mathbf{V}_{\mathrm{YUV}}$ besteht, getrennt. Damit können für den Farbvergleich die Chrominanz-Kanäle herangezogen werden, während die Luminanz zugunsten der Beleuchtungsinvarianz ignoriert wird. Den zweiten Farbmodus bildet die absolute Sättigung aus dem HSV-Farbraum (Farbton, Sättigung, Hellwert, engl. *Hue, Saturation, Value*). Die meisten Wege weisen eine niedrige Farbsättigung auf, wodurch sie sich oftmals von ihrer Umgebung abheben. Der letzte Farbmodus bestimmt den Grünanteil. Befindet sich der gesuchte Weg neben grünen Wiesen, so ergibt sich hier ein deutlicher Unterschied im Grünanteil. In jedem dieser drei Farbmodi wird der Farbunterschied jedes Pixels zu dem Bereich unter dem Fahrzeug bestimmt. Aus den Farbunterschieds-Grids $\mathbf{D}$ wird dann für jeden Farbmodus ein Feature $\mathbf{F}$ extrahiert, das jeder Grid-Zelle einen Wert zwischen 0 für „kein Weg“ und 1 für „Weg“ zuordnet. Aus der Multiplikation dieser drei Features wird letztlich eine Straßenwahrscheinlichkeit berechnet.

Eingangsdaten: Terrain-Grid mit im RGB-Farbraum akkumulierter Farbinformation.

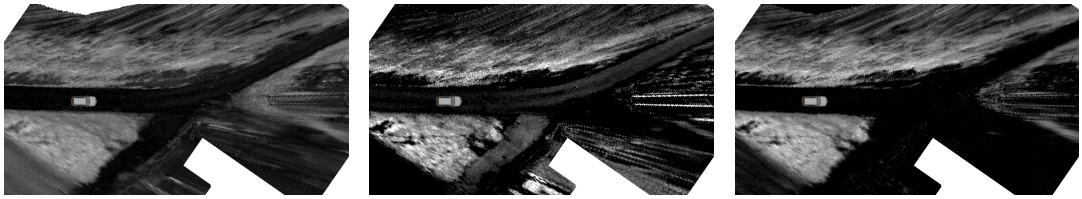


1. Das RGB-Grid wird in drei verschiedene Farbräume und -modi umgerechnet.

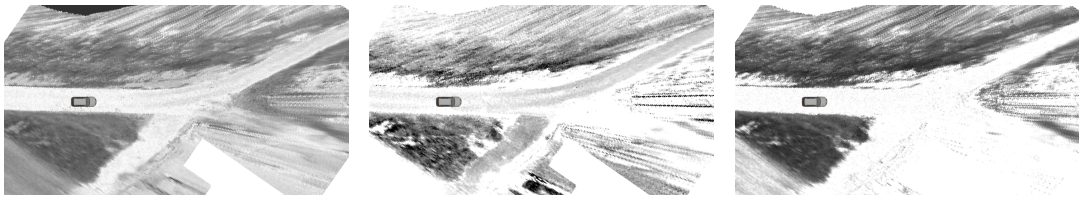


Chrominanz $\mathbf{U}_{\mathrm{YUV}}$ und $\mathbf{V}_{\mathrm{YUV}}$ | Sättigung $\mathbf{S}_{\mathrm{HSV}}$ | Grünanteil

2. In jedem dieser drei Farbmodi wird jedes Pixel mit dem Bereich unter dem Fahrzeug verglichen und je ein Farbunterschieds-Grid $\mathbf{D}$ erzeugt (schwarz: geringer Unterschied).



3. Aus jedem Farbunterschieds-Grid wird ein Feature-Grid $\mathbf{F} \in [0;1]$ berechnet (weiß: 1).



4. Die Multiplikation der drei Features ergibt eine Straßenwahrscheinlichkeit (weiß: hohe Straßenwahrscheinlichkeit).

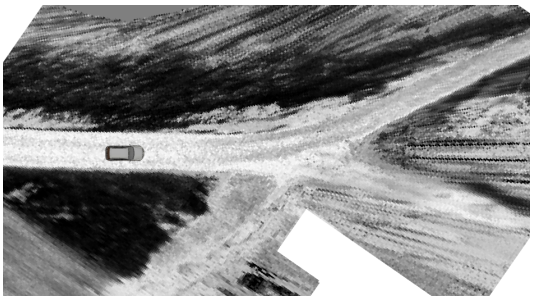
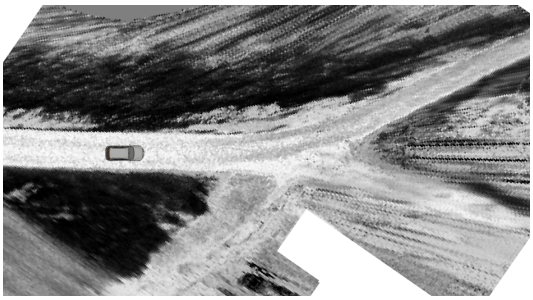
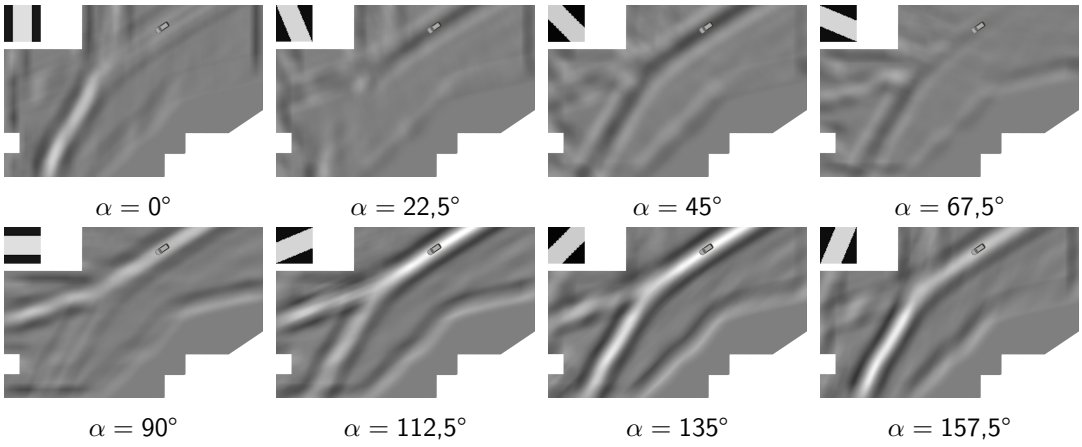


Abbildung 9.2:
Ablauf der Extraktion einer Straßenwahrscheinlichkeit für eine Beispielkreuzung mit drei Kreuzungsästen und einer Hofeinfahrt (erste rechts) nach dem Konzept von Bayerl und Wuensche [2014].

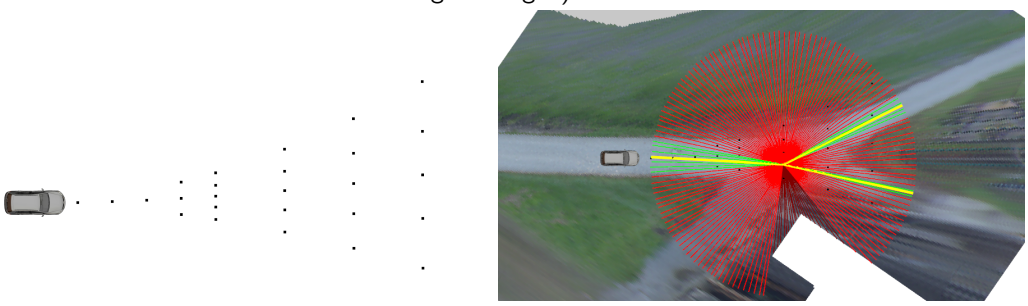
Eingangsdaten: Terrain-Grid mit Straßenwahrscheinlichkeit.



5. Durch die Faltung der Straßenwahrscheinlichkeit mit einer der Wegbreite entsprechenden Maske in acht Winkeln wird eine gerichtete Straßenwahrscheinlichkeit gewonnen. Die Faltung erfolgt in Odometriekoordinaten, weshalb das Grid passend zur Faltungsmaske gedreht visualisiert ist.



6. Diese wird für verschiedene Mittelpunkthypothesen mit verschiedenen Richtungshypothesen ausgewertet (grüne Linien zeigen gültige Richtungshypothesen, gelbe Linien mögliche Wegrichtungen).



Ergebnis: Messungen der Kreuzungsäste (gelb), die über die Zeit gefiltert werden (blau).

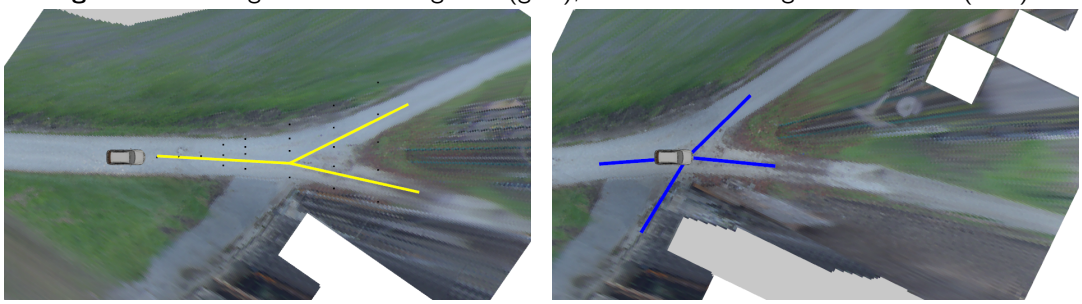
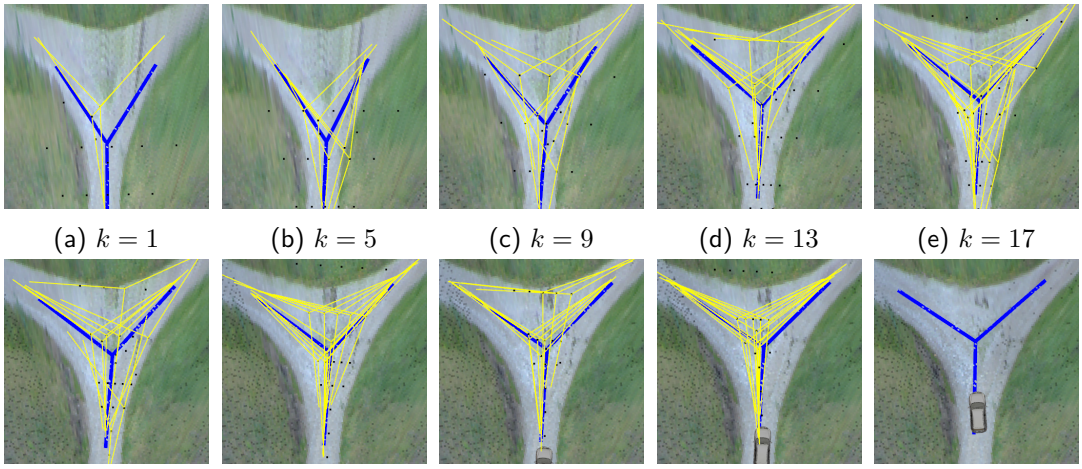


Abbildung 9.3: Ablauf der Kreuzungsdetektion von Bayerl und Wuensche [2014].

Aus dem Straßenwahrscheinlichkeits-Grid werden, wie in Abbildung 9.3 visualisiert, Messungen für ein Kreuzungstracking extrahiert. Dabei wird das Vorwissen über die ungefähre Breite eines Wegs genutzt. Durch eine Faltung mit vorberechneten Faltungsmasken (*Template Matching*) wird gezielt eine entsprechend breite Linie mit einer hohen Straßenwahrscheinlichkeit gesucht, neben der die Straßenwahrscheinlichkeit wiederum niedrig ist. Um Wege verschiedener Richtungen detektieren zu können, werden acht Faltungsmasken mit in unterschiedlichen Winkeln rotierten Balken verwendet. Das Ergebnis der Faltungsoperationen sind acht Grids mit einer gerichteten Straßenwahrscheinlichkeit. Ein Weg entsprechender Breite in einer der Faltungsmaske ähnlichen Richtung verursacht eine stark positive (weiß dargestellte) Antwort in der gerichteten Straßenwahrscheinlichkeit, abgegrenzt von einer stark negativen (schwarz dargestellten) Antwort direkt neben dem Weg. Diese gerichtete Straßenwahrscheinlichkeit dient der Verifikation von Kreuzungshypothesen. Abhängig vom aktuellen Lenkwinkel des Fahrzeugs werden klothoidenförmig mögliche Kreuzungsmittelpunkte in verschiedenen Entfernungen und mit unterschiedlichen Lenkwinkeländerungen gestreut und evaluiert. Für jede dieser Mittelpunkthypothesen werden Richtungshypothesen in Form radialer Strahlen aufgestellt. Für jeden Strahl wird die gerichtete Straßenwahrscheinlichkeit der dem Strahl ähnlichsten Richtung betrachtet. Liegt die mittlere gerichtete Straßenwahrscheinlichkeit entlang des Strahls über einem dynamisch bestimmten Schwellwert, so gilt die Richtungshypothese als gültig. Mehrere benachbarte gültige Richtungshypothesen werden zu einer möglichen Wegrichtung gebündelt. Werden für eine Mittelpunkthypothese mindestens drei mögliche Wegrichtungen gefunden, so bilden diese eine Kreuzungsdetektion. Da die Hypothesen grob gestreut sind, sind die detektierten Kreuzungen sehr ungenau lokalisiert. Daher fließen sie als Messung in ein rekursives Tracking ein (vgl. Abbildung 9.4). Hierfür kommt ein Erweitertes Kalman-Filter (EKF) zum Einsatz.



(a) $k = 1$ (b) $k = 5$ (c) $k = 9$ (d) $k = 13$ (e) $k = 17$

(f) $k = 21$ (g) $k = 25$ (h) $k = 29$ (i) $k = 33$ (j) $k = 40$

Abbildung 9.4:
Visualisierung der Messungen (gelb) und der damit rekursiv verbesserten Schätzung (blau) einer Kreuzung zu verschiedenen Zeitschritten k. In einem Zeitschritt können (aus unterschiedlichen Mittelpunkthypothesen) mehrere Messungen gewonnen werden. Mit steigender Anzahl Messungen wird die Schätzung stetig korrigiert, bis das Filter eingeschwungen ist und die Schätzung perfekt in der Kreuzung liegt.

9.4 Integration der Farbmodi aus dem Wegetracking

Für die Bestimmung der Straßenwahrscheinlichkeit kommen in der Kreuzungserkennung von Bayerl und Wuensche [2014] drei Farbmodi zum Einsatz. Der erste Farbmodus, die Chrominanz, dient dem intuitiven Vergleich der sichtbaren Farbe. So können Wege identifiziert werden, die für den Menschen ähnlich aussehen wie der aktuelle Weg. Die anderen beiden Farbmodi – das Sättigungsfeature und das Grünfeature – sind dagegen gezielte Farbmerkmale, in denen sich ein Weg möglichst stark von seiner Umgebung abhebt. Auch im Wegetracking in Kapitel 8 mussten solche Farbmodi gefunden werden, die den Kontrast zwischen Weg und Umgebung maximieren. Dabei wurden ebenfalls ein Farbmodus basierend auf der Sättigung und ein Farbmodus basierend auf Grüntönen entwickelt. Daher soll nun für das Sättigungs- und das Grünfeature jeweils der ursprünglich von Bayerl und Wuensche [2014] vorgeschlagene Farbmodus mit dem im Wegetracking entwickelten Farbmodus verglichen werden und der robusteste Farbmodus gewählt werden.

9.4.1 Sättigungsfeature

Das Sättigungsfeature von Bayerl und Wuensche [2014] nutzt die (absolute) Sättigung $\mathbf{S}_{\text{HSV}}$ aus dem HSV-Farbraum. Dazu wird aus den zellenweisen maximalen und minimalen Farbwerten im RGB-Farbraum,

$$\mathbf{MAX}_{\text{RGB}} = \max(\mathbf{R}_{\text{RGB}}, \mathbf{G}_{\text{RGB}}, \mathbf{B}_{\text{RGB}}) \,, \tag{9.1}$$

$$\mathbf{MIN}_{\text{RGB}} = \min(\mathbf{R}_{\text{RGB}}, \mathbf{G}_{\text{RGB}}, \mathbf{B}_{\text{RGB}}) \,, \tag{9.2}$$

die Sättigung

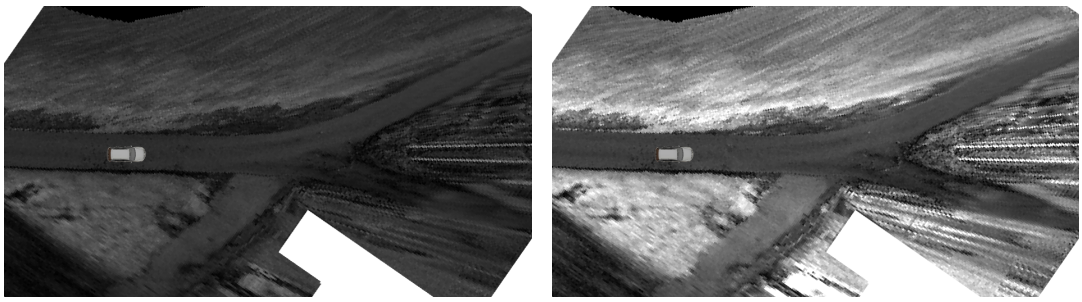
$$\mathbf{S}_{\text{HSV}} = \begin{cases} 0 & \text{wenn } \mathbf{MAX}_{\text{RGB}} = 0 \,, \\ \frac{\mathbf{MAX}_{\text{RGB}} - \mathbf{MIN}_{\text{RGB}}}{\mathbf{MAX}_{\text{RGB}}} & \text{sonst} \end{cases} \tag{9.3}$$

berechnet. In Abbildung 9.5a ist die Sättigung der aus Abbildung 9.2 bekannten Beispielkreuzung dargestellt. Der Sättigungskontrast ist bei dieser jedoch – insbesondere bei der zweiten rechten Abzweigung – nicht optimal.

Im Wegetracking hat es sich als vorteilhaft erwiesen, anstatt der absoluten Sättigung $\mathbf{S}_{\text{HSV}}$ die relative Sättigung $\mathbf{S}_{\text{HSV}}/\mathbf{V}_{\text{HSV}}$, d. h. das Verhältnis der Sättigung $\mathbf{S}_{\text{HSV}}$ zum Hellwert (engl. *Value*)

$$\mathbf{V}_{\text{HSV}} = \mathbf{MAX}_{\text{RGB}} \tag{9.4}$$

des HSV-Farbraums zu betrachten (vgl. Abschnitt 8.4.3). In Abbildung 9.5b ist die relative Sättigung der Beispielkreuzung abgebildet. Im direkten Vergleich mit der absoluten Sättigung zeigt sich, dass die relative Sättigung auch hier den Kontrast des Wegs zur Umgebung deutlich erhöht. Insbesondere hebt sich auch der fragliche Kreuzungsast deutlich von der Umgebung ab. Daher wird von nun an für das Sättigungsfeature die absolute Sättigung durch die relative Sättigung ersetzt.



(a) $\mathbf{S}_{\text{HSV}}$ (b) $\mathbf{S}_{\text{HSV}}/\mathbf{V}_{\text{HSV}}$

Abbildung 9.5:
Vergleich der absoluten Sättigung $\mathbf{S}_{\text{HSV}}$ mit der relativen Sättigung $\mathbf{S}_{\text{HSV}}/\mathbf{V}_{\text{HSV}}$.

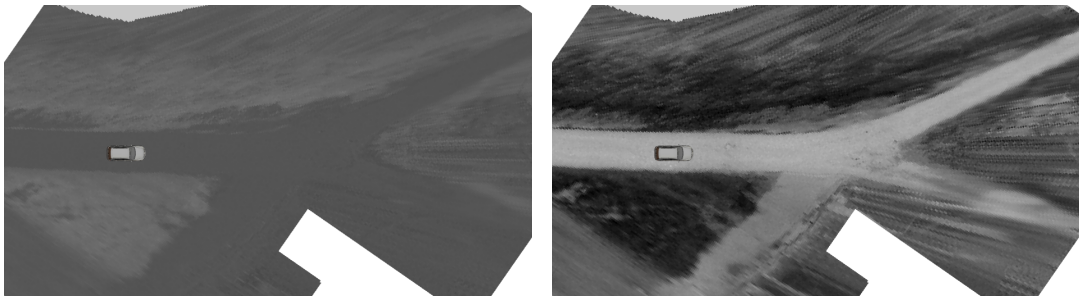
9.4.2 Grünfeature

Als dritter Farbmodus wurde in der Kreuzungserkennung von Bayerl und Wuensche [2014] die sog. Green-Ratio $\mathbf{G}_{\text{RGB}}/\left(\mathbf{R}_{\text{RGB}}+\mathbf{G}_{\text{RGB}}+\mathbf{B}_{\text{RGB}}\right)$ verwendet, die von Tan et al. [2006] unter dem Namen „Normalized Green“ zur Erkennung von Wegen vorgestellt wurde. In der Annahme, dass ein Feldweg oft durch fehlendes Grün aus einer grünen Umgebung heraussticht, setzt die Green-Ratio den Grünkanal aus dem RGB-Farbraum ins Verhältnis zur Summe der Farbintensitäten. Abbildung 9.6a visualisiert die Green-Ratio für den Beispielfeldweg.

Im Wegetracking hat sich zur Kontrasterhöhung zwischen einem typischen Feldweg und einer grünen Umgebung allerdings die Differenz

$$\mathbf{Y}_{\text{YUV}}-\mathbf{I3}_{\text{I1I2I3}}=1{,}299\cdot\mathbf{R}_{\text{RGB}}-1{,}413\cdot\mathbf{G}_{\text{RGB}}+1{,}114\cdot\mathbf{B}_{\text{RGB}} \tag{9.5}$$

aus dem Graubild $\mathbf{Y}_{\text{YUV}}$ und dem $\mathbf{I3}_{\text{I1I2I3}}$-Kanal aus dem I1I2I3-Farbraum von Ohta et al. [1980] als robuster erwiesen als die Green-Ratio (s. Abschnitt 8.4.3). Abbildung 9.6b zeigt $\mathbf{Y}_{\text{YUV}}-\mathbf{I3}_{\text{I1I2I3}}$ für die Beispielkreuzung. Auch hier wird der verbesserte Kontrast im Vergleich zur Green-Ratio deutlich: Insbesondere die beiden rechten Kreuzungsäste heben sich stärker von der nicht-grünen Umgebung zwischen der Hofeinfahrt und der rechten Abzweigung ab. Daher wird für das Grünfeature im Folgenden anstatt der Green-Ratio der Farbmodus $\mathbf{Y}_{\text{YUV}}-\mathbf{I3}_{\text{I1I2I3}}$ verwendet.



(a) $\mathbf{G}_{\text{RGB}}/\left(\mathbf{R}_{\text{RGB}}+\mathbf{G}_{\text{RGB}}+\mathbf{B}_{\text{RGB}}\right)$ (b) $\mathbf{Y}_{\text{YUV}}-\mathbf{I3}_{\text{I1I2I3}}$

Abbildung 9.6:
Vergleich der Green-Ratio mit dem Farbmodus $\mathbf{Y}_{\text{YUV}}-\mathbf{I3}_{\text{I1I2I3}}$ aus dem Wegetracking.

9.4.3 Feature-Grids

Zur Integration in die Kreuzungserkennung wird aus der relativen Sättigung und dem $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$-Farbmodus je ein Feature-Grid berechnet, indem jede Grid-Zelle mit dem Bereich unter dem Fahrzeug verglichen wird. Dazu wird, wie durch Bayerl und Wuensche [2014] vorgeschlagen, der Mittelwert μ_{ROI} des jeweiligen Farbmodus $\mathbf{C}$ auf der Fläche $\mathcal{A}_{\mathsf{ROI}}$ unter dem Fahrzeug bestimmt und durch ein gewichtetes Mittel mit dem Mittelwert aus dem letzten Zeitschritt tiefpassgefiltert:

$$\mu_{\mathsf{ROI},k} = 0{,}8 \cdot \mu_{\mathsf{ROI},k-1} + 0{,}2 \cdot \left(\frac{1}{|\mathcal{A}_{\mathsf{ROI}}|} \sum_{\mathsf{c} \in \mathcal{A}_{\mathsf{ROI}}} \mathbf{C}(\mathsf{c}) \right) . \tag{9.6}$$

Damit für beide Farbmodi dieselbe Feature-Berechnung zum Einsatz kommen kann, wird der $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$-Farbmodus invertiert (d. h. $\mathbf{C}_{\mathsf{Y\text{-}I3}} = 1 - (\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}})$, $\mathbf{C}_{\mathsf{S/V}} = \mathbf{S}_{\mathsf{HSV}}/\mathbf{V}_{\mathsf{HSV}}$), sodass der Weg in beiden Farbmodi eine geringe Intensität hat. Unter der Annahme, dass der Weg meist die niedrigste Sättigung und den niedrigsten Grünanteil aufweist, wird bei der Berechnung des Farbunterschieds-Grids

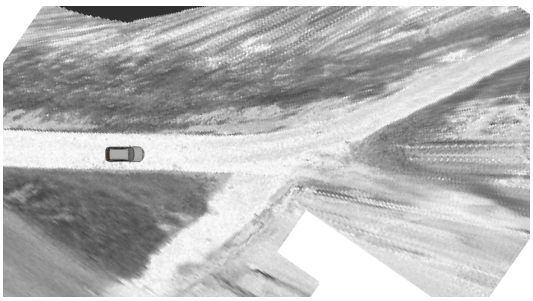
$$\mathbf{D} = \max(0,\, \mathbf{C} - \mu_{\mathsf{ROI}}) \tag{9.7}$$

das Vorzeichen der Farbdifferenz berücksichtigt. Grid-Zellen, die eine noch niedrigere Sättigung und noch weniger grüne Farbe als die ROI unter dem Fahrzeug besitzen, erhalten dementsprechend einen Farbunterschied von 0. Aus dem Farbunterschieds-Grid $\mathbf{D}$ und einem Normalisierungsfaktor n wird schließlich das jeweilige Feature $\mathbf{F} \in [0; 1]$ berechnet:

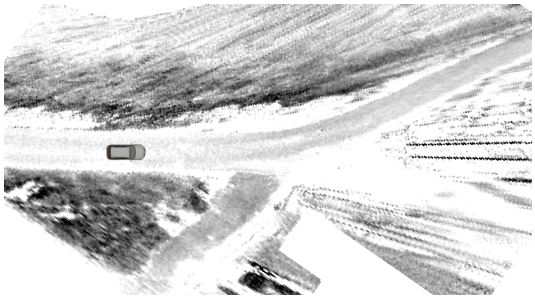
$$\mathbf{F} = 1 - \frac{\mathbf{D}}{n}, \tag{9.8}$$

wobei für das relative Sättigungsfeature $n_{\mathsf{S/V}} = 0{,}5$ und für das neue Grünfeature $n_{\mathsf{Y\text{-}I3}} = 0{,}3$ gewählt werden.

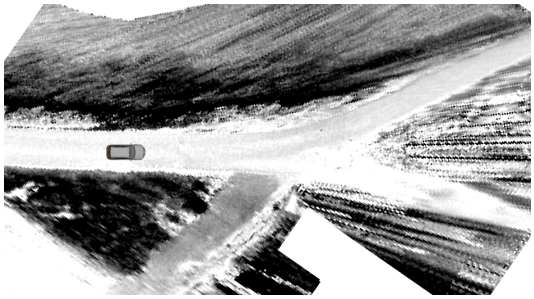
Abbildung 9.7 vergleicht diese Features anhand der Beispielkreuzung mit den bisherigen Features und zeigt die jeweils resultierende Straßenwahrscheinlichkeit. Unter den Farbfeatures von Bayerl und Wuensche [2014] separiert das Farbfeature $\mathbf{F}_{\mathsf{UV}}$ die Kreuzungsäste am deutlichsten mit einem hohen Featurewert auf dem Weg und einem etwas niedrigerem Featurewert neben dem Weg. Beim absoluten Sättigungsfeature $\mathbf{F}_{\mathsf{S}}$ weisen die linke Abzweigung und die Hofeinfahrt dagegen einen niedrigeren Featurewert auf als Bereiche neben dem Weg. Im Grünfeature aus der Green-Ratio $\mathbf{F}_{\mathsf{GR}}$ erhält der Weg zwar wieder einen hohen Featurewert, der gesamte Bereich zwischen den beiden rechten Kreuzungsästen jedoch ebenso. So ist die Erkennung der rechten Kreuzungsäste insgesamt schwierig, und es besteht die Gefahr von falsch-positiven Detektionen. Bei den neuen Features können dagegen alle Kreuzungsäste in allen Farbfeatures eindeutig durch einen hohen Featurewert erkannt und abgegrenzt werden. Zudem erhöht sich der Unterschied zwischen den tatsächlichen Wegabzweigungen und der Hofeinfahrt, sodass mit der Parametrierung des Messverfahrens gewählt werden kann, ob die Hofeinfahrt erkannt werden soll oder nicht. Insgesamt zeigt sich also eine erhöhte Robustheit der Straßenwahrscheinlichkeit durch die Nutzung der im Wegetracking entwickelten Farbmodi.



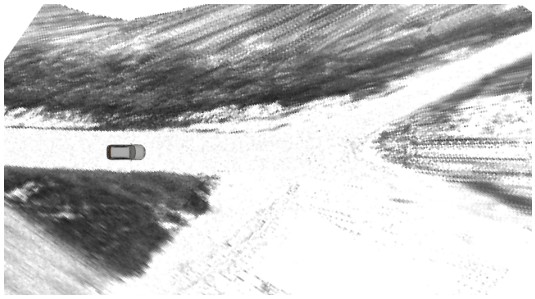
(a) Farbfeature $\mathbf{F}_{\text{UV}}$



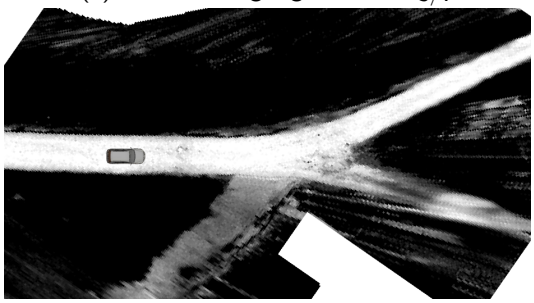
(b) Altes Sättigungsfeature $\mathbf{F}_{\text{S}}$



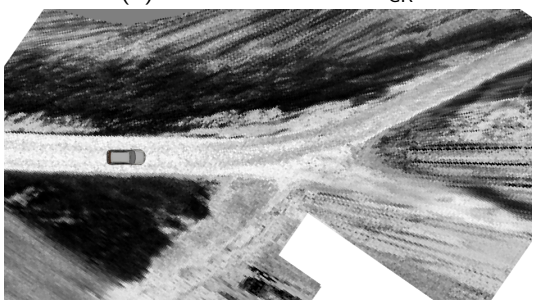
(c) Neues Sättigungsfeature $\mathbf{F}_{\text{S/V}}$



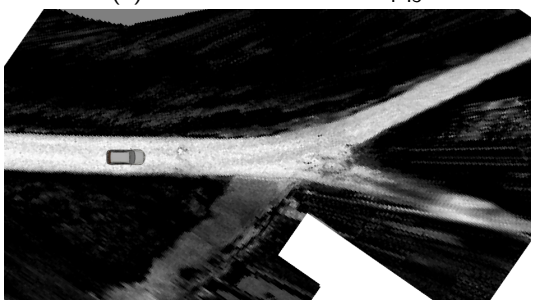
(d) Altes Grünfeature $\mathbf{F}_{\text{GR}}$



(e) Neues Grünfeature $\mathbf{F}_{\text{Y-I3}}$



(f) Straßenwahrscheinlichkeit $\mathbf{F}_{\text{UV}} \cdot \mathbf{F}_{\text{S}} \cdot \mathbf{F}_{\text{GR}}$



(g) Straßenwahrscheinlichkeit $\mathbf{F}_{\text{UV}} \cdot \mathbf{F}_{\text{S/V}} \cdot \mathbf{F}_{\text{Y-I3}}$

Abbildung 9.7:
Vergleich der Features, die aus den von Bayerl und Wuensche [2014] vorgestellten Farbmodi berechnet wurden (links), mit den Features, die auf den im Wegetracking entwickelten Farbmodi basieren (rechts). Ein Featurewert von 1 wird weiß visualisiert, ein Featurewert von 0 schwarz.

9.5 Ergebnisse

Abschließend sollen die Ergebnisse der mit den weiterentwickelten Farbfeatures implementierten Kreuzungserkennung beurteilt werden. Mangels öffentlich verfügbarer Datensätze und vergleichbarer Ansätze in der Literatur können dabei nur qualitative Ergebnisse gezeigt und ein direkter Vergleich zum zugrundeliegenden Verfahren von Bayerl und Wuensche [2014] mit den ursprünglichen Farbmodi gezogen werden.

9.5.1 Qualitative Ergebnisse

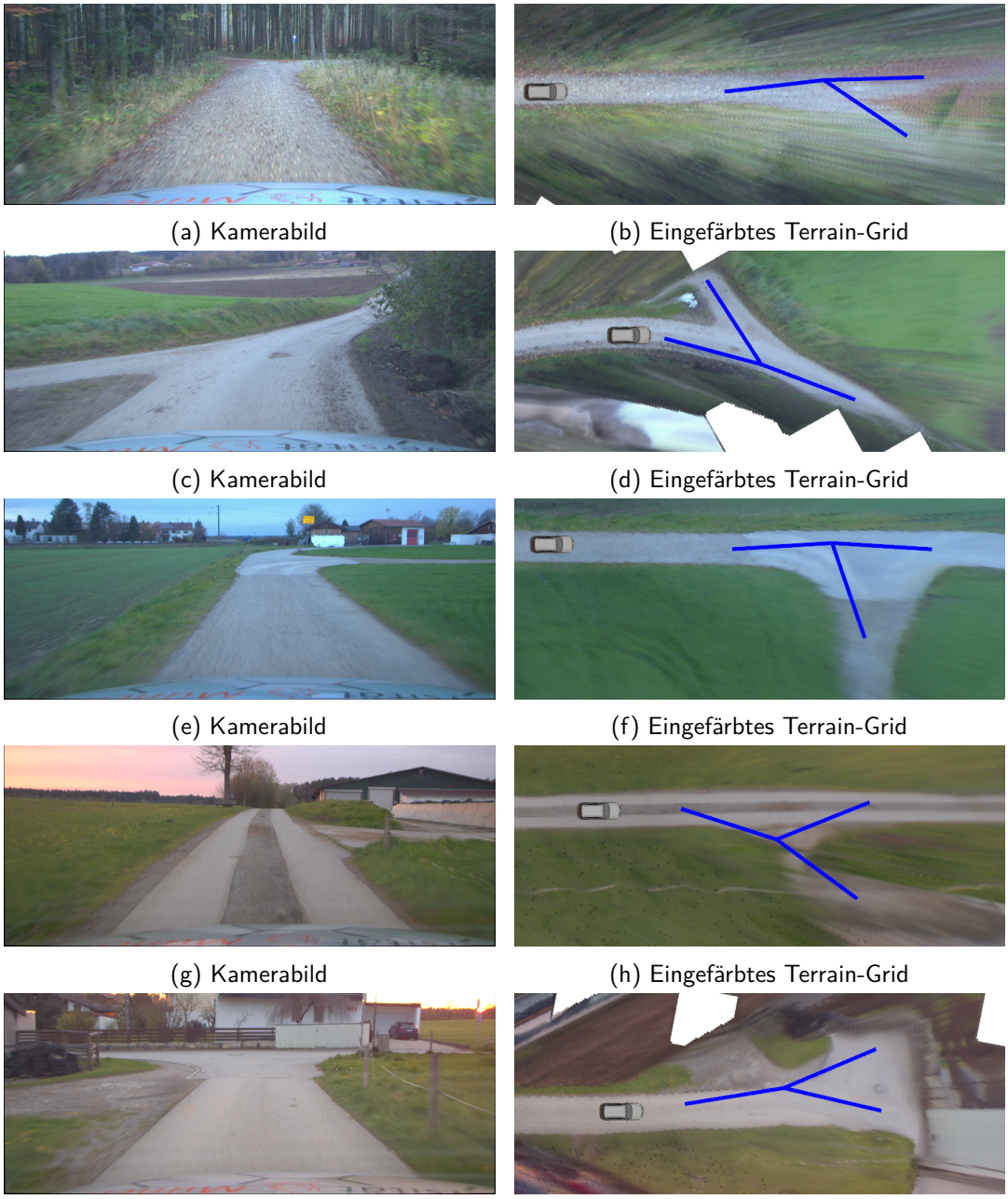
Zur qualitativen Beurteilung des Gesamtverfahrens zeigt Abbildung 9.8 Beispiele verschiedenartiger Feldwege, die im Spätherbst und zu Frühlingsbeginn, also bei nicht optimal sattgrüner Umgebung, ohne Kartenwissen lediglich aus RGB-Kamera- und LiDAR-Informationen erfolgreich erkannt wurden. Um die Vorausschau beurteilen zu können, und da das Tracking der Kreuzungen keine Kontribution dieser Arbeit ist, wird jeweils der (grobe) Zeitpunkt der ersten Kreuzungsdetektion und damit des Aufsetzens des Trackings gezeigt. Zu diesem Zeitpunkt kann eine nachgelagerte Verhaltensplanung bereits entscheiden, welche Richtung eingeschlagen werden soll. Die Schätzung ist allerdings noch nicht perfekt lokalisiert. Wie in Abbildung 9.4 gezeigt wurde, verbessert sich die Kreuzungsschätzung jedoch noch signifikant, bis das Fahrzeug die Kreuzung erreicht.

Mit dem implementierten Verfahren können unter anderem auch herausfordernde Kreuzungen mit Laub (Abbildung 9.8a bzw. 9.8b), Abzweigungen nach hinten (Abbildung 9.8c bzw. 9.8d), Kreuzungen mit wechselndem Straßenbelag (Abbildung 9.8e bzw. 9.8f), Kreuzungen auf Wegen mit geteerten Spurrillen (Abbildung 9.8g bzw. 9.8h) sowie auch kleine Kreuzungen in Dörfern (Abbildung 9.8i bzw. 9.8j) rechtzeitig detektiert und getrackt werden, bevor das Auto eine Kreuzung erreicht. Abbildung 9.9 zeigt weitere Beispiele im Winter. Als Besonderheiten seien hier Schnee (Abbildung 9.9c bzw. 9.9d) und eine Verkehrsinsel (Abbildung 9.9i bzw. 9.9j) genannt. Insgesamt kann eine Vielzahl an Feldwegkreuzungen erkannt werden.

9.5.2 Vergleich mit dem Ausgangsverfahren

Um die implementierte Kreuzungserkennung mit der zugrundeliegenden Kreuzungserkennung von Bayerl und Wuensche [2014] zu vergleichen, wird auf 14 aufgenommenen Feldwegen verglichen, welche Kreuzungen die beiden Verfahren erkannt haben. Dabei werden die erkannten Kreuzungen nur untereinander abgeglichen und keine „wahre“ Gesamtanzahl an Kreuzungen gezählt, um Schwierigkeiten bei der Definition einer Kreuzung sowie gemeinsame Grenzen der Verfahren auszuklammern. Für die Kreuzungserkennung von Bayerl und Wuensche [2014] wird die Original-Implementierung verwendet. Beide Verfahren wurden nicht auf die Aufnahmen hin parametriert.

Insgesamt werden in den Aufnahmen 23 korrekte Feldwegkreuzungen detektiert. Dreizehn davon werden durch beide Implementierungen erkannt. Vier Kreuzungen



(a) Kamerabild (b) Eingefärbtes Terrain-Grid

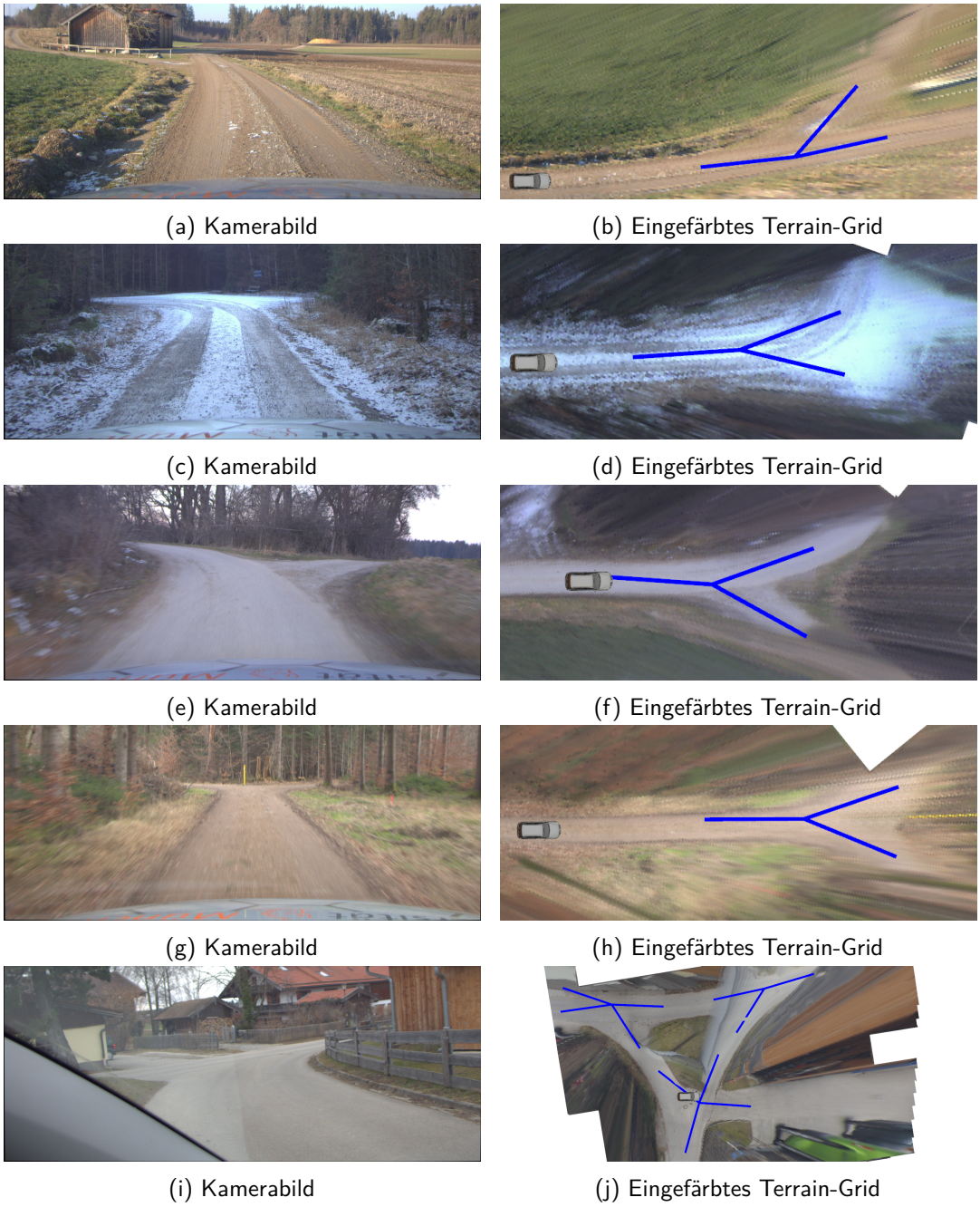
(c) Kamerabild (d) Eingefärbtes Terrain-Grid

(e) Kamerabild (f) Eingefärbtes Terrain-Grid

(g) Kamerabild (h) Eingefärbtes Terrain-Grid

(i) Kamerabild (j) Eingefärbtes Terrain-Grid

Abbildung 9.8:
Beispiele erfolgreich erkannter Kreuzungen in Aying (Landkreis München) und Stephanskirchen (Landkreis Rosenheim). Die Aufnahmen erfolgten in den Monaten November und April am Tag bzw. in der Dämmerung. Gezeigt sind das Kamerabild und die geschätzte Kreuzung auf dem eingefärbten Terrain-Grid zum Zeitpunkt des Aufsetzens des Trackings. Die Schätzungen sind daher noch nicht perfekt lokalisiert. Es kann jedoch nachvollzogen werden, unter welchen Sichtbedingungen die Kreuzung detektiert wurde und ob die Vorausschau ausreichend ist. Durch die Streuung neuer Mittelpunkthypothesen um den erkannten Kreuzungsmittelpunkt wird die Schätzung im nachgelagerten Trackingverfahren mit der Zeit genauer lokalisiert.



(a) Kamerabild (b) Eingefärbtes Terrain-Grid

(c) Kamerabild (d) Eingefärbtes Terrain-Grid

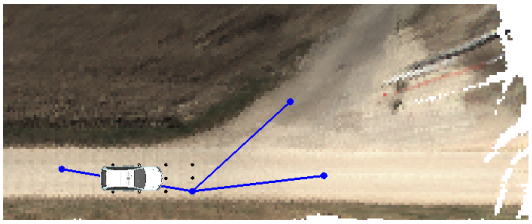
(e) Kamerabild (f) Eingefärbtes Terrain-Grid

(g) Kamerabild (h) Eingefärbtes Terrain-Grid

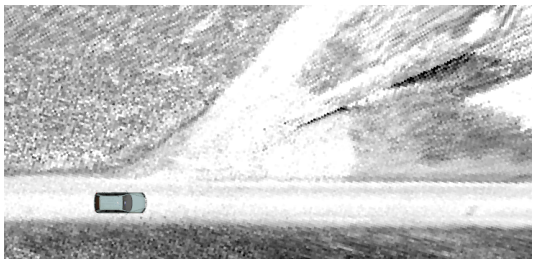
(i) Kamerabild (j) Eingefärbtes Terrain-Grid

Abbildung 9.9:
Verschiedene Kreuzungen in Aying (Landkreis München), die während einer Demonstration im Winter erfolgreich erkannt wurden. Die Aufnahmen erfolgten im Monat Februar am Tag. Gezeigt sind das Kamerabild und die geschätzte Kreuzung auf dem eingefärbten Terrain-Grid zum Zeitpunkt des Aufsetzens des Trackings. Bei der Verkehrsinsel in (j) wurde zunächst die linke Kreuzung erkannt, dann die untere Kreuzung und zuletzt zum gezeigten Zeitpunkt die obere Kreuzung, die im Kamerabild zu sehen ist. Die bewegliche Kameraplattform ist dabei entsprechend des Lenkwinkels stark nach links gedreht.

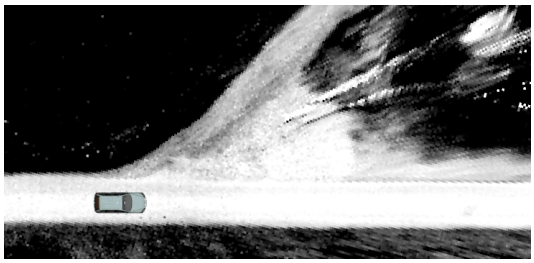
werden nur durch die in dieser Arbeit implementierte Applikation erkannt, sechs weitere dagegen nur durch die Kreuzungserkennung von Bayerl und Wuensche [2014]. Der Großteil dieser abweichenden Detektionen ist durch kleine Diskrepanzen im jeweils verwendeten Umgebungsmodell zu begründen. Einzig die in Abbildung 9.10 abgebildete Kreuzung weist in beiden Verfahren signifikant unterschiedliche Straßenwahrscheinlichkeiten auf. Mit den neuen Farbmodi wird diese Kreuzung nicht erkannt. Ein Vergleich der Farbfeatures zeigt, dass hier das neue Grünfeature aus dem $\mathbf{Y}_{\mathrm{YUV}} - \mathbf{I3}_{\mathrm{I1I2I3}}$-Farbmodus fälschlicherweise eine Abzweigung ausschließt, da diese eine dunklere Farbe aufweist und dadurch der Farbunterschied positiv ausfällt.



(a) Durch Bayerl und Wuensche [2014] erkannte Kreuzung



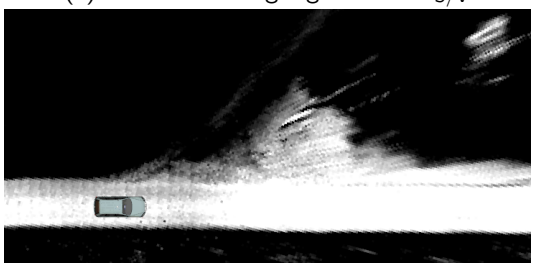
(b) Absolutes Sättigungsfeature $\mathbf{F}_{\mathrm{S}}$



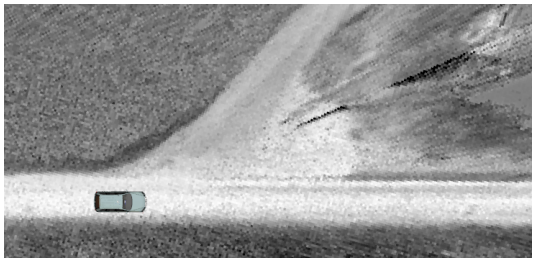
(c) Relatives Sättigungsfeature $\mathbf{F}_{\mathrm{S/V}}$



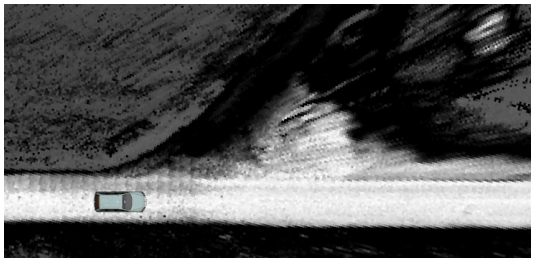
(d) Grünfeature aus der Green-Ratio $\mathbf{F}_{\mathrm{GR}}$



(e) Neues Grünfeature $\mathbf{F}_{\mathrm{Y\text{-}I3}}$



(f) Straßenwahrscheinlichkeit $\mathbf{F}_{\mathrm{UV}} \cdot \mathbf{F}_{\mathrm{S}} \cdot \mathbf{F}_{\mathrm{GR}}$



(g) Straßenwahrscheinlichkeit $\mathbf{F}_{\mathrm{UV}} \cdot \mathbf{F}_{\mathrm{S/V}} \cdot \mathbf{F}_{\mathrm{Y\text{-}I3}}$

Abbildung 9.10:
Eine Kreuzung, die nur durch die Implementierung von Bayerl und Wuensche [2014], nicht jedoch durch das in dieser Arbeit weiterentwickelte Verfahren erkannt wurde. Gezeigt wird der Zeitpunkt des Aufsetzens des Trackings durch ersteres Verfahren sowie die alten (links) und neuen (rechts) Farbfeatures zu diesem Zeitpunkt im Vergleich.

Ein deutlicher Unterschied zwischen den beiden Varianten der Kreuzungserkennung und ihren Farbfeatures liegt in der Anzahl falsch-positiver Detektionen. Während die Implementierung von Bayerl und Wuensche [2014] an zehn unterschiedlichen Stellen fälschlicherweise eine nicht vorhandene Kreuzung erkennt, wird durch die Weiterentwicklung mit den neuen Farbmodi keine einzige falsche Kreuzung mehr detektiert. Dies betrifft vor allem starke Schatten, bei denen das Sättigungsfeature signifikant von der Verwendung der relativen Sättigung profitiert, sowie Übergänge von grünem Gras zu braunen Äckern neben dem Weg, bei denen sowohl die absolute Sättigung als auch die Green-Ratio als Wegmerkmale versagen. Abbildung 9.11 zeigt je zwei Beispiele für diese Situationen. Die anhand der Erkenntnisse aus dem Wegetracking weiterentwickelten Farbfeatures erhöhen also erheblich die Robustheit der Kreuzungserkennung gegenüber falsch-positiven Detektionen.

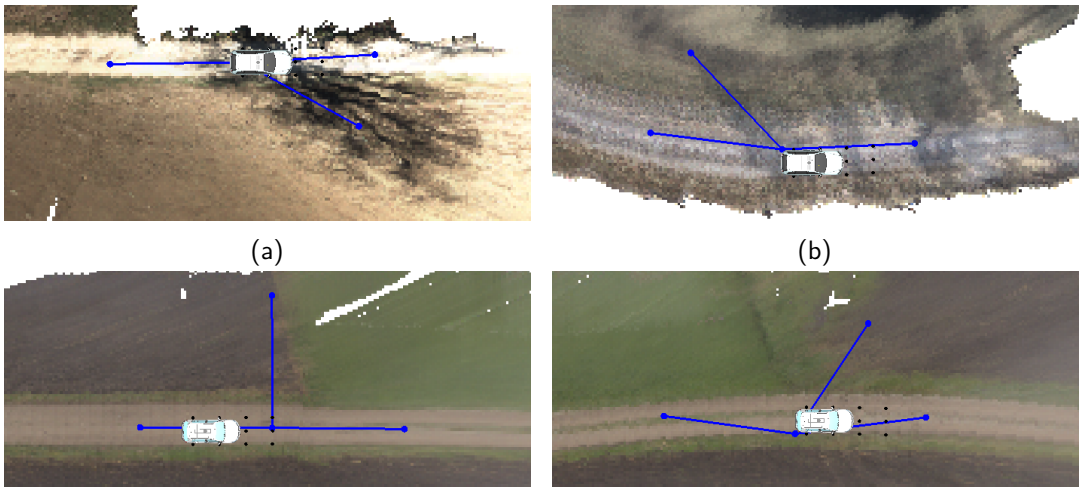


Abbildung 9.11:
Beispiele der falsch-positiven Detektionen durch die ursprüngliche Kreuzungserkennung von Bayerl und Wuensche [2014], die durch die weiterentwickelten Farbmodi eliminiert werden konnten. Gezeigt ist der eingeschwungene Filterzustand nach einigen Zeitschritten.

9.5.3 Grenzen des Verfahrens

Würde man die erkannten Kreuzungen nicht nur zwischen den Implementierungen abgleichen, sondern anhand einer Karte auswerten, so würden sich in beiden Varianten falsch-negative Detektionen zeigen. Diese ergeben sich aus vereinfachenden Modellannahmen, die für die Erkennung der Kreuzungen getroffen wurden.

Bei der Berechnung der Features wird angenommen, dass die Abzweigungen farblich ähnlich zum aktuellen Weg sind. Es werden zwar verschiedene Farbmodi herangezogen, diese werden jedoch alle relativ zu dem Bereich unter dem Fahrzeug ausgewertet. Dadurch kann aber beispielsweise die in Abbildung 9.12 abgebildete Kreuzung mit Dreck auf dem Weg nicht detektiert werden, da die Abzweigung in allen Farbmodi vom aktuellen Weg abweicht. Absolute Farbfeatures, die nicht mit dem aktuellen Weg vergleichen, wären jedoch wiederum sehr anfällig für falsch-positive Detektionen.



(a) Kamerabild



(b) Eingefärbtes Terrain-Grid

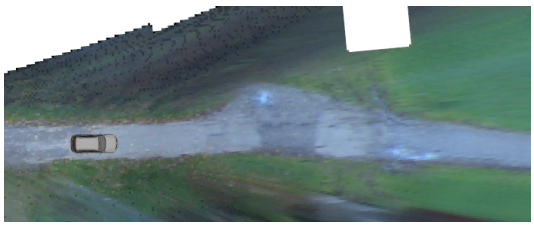
Abbildung 9.12:
Beispiel für eine Kreuzung, die aufgrund der relativen Farbfeatures nicht erkannt werden kann.

Die zweite Modellannahme liegt darin, dass eine Kreuzung einen Schnittpunkt von Wegen unterschiedlicher Richtungen darstellt. Auch das grenzt die detektierbaren Kreuzungen ein. So können Kreuzungen, bei denen die einbiegenden Wege aufgrund von Verdeckungen nicht einsehbar sind, nicht erkannt werden. Abbildung 9.13 zeigt zwei Beispiele. Der menschliche Betrachter würde zwar trotz der Verdeckung aufgrund der dreieckigen Einmündung eine Abzweigung vermuten, die Berücksichtigung dieser Einmündungen würde jedoch sehr viele falsch-positive Detektionen im Bereich von Vorplätzen oder Einfahrten zur Folge haben. Auch Kreuzungen mit großen Freiflächen können unter dieser Modellannahme nicht erkannt werden, da diese keine klaren Kanten in der gerichteten Straßenwahrscheinlichkeit erzeugen.

Schließlich ist es oft schwierig zu definieren, welche Einmündung eine Kreuzung darstellt. So werden meist nur Kreuzungen gleichberechtigter Wege als relevante Kreuzung wahrgenommen. Bei der Einmündung eines Bauernwegs auf eine Landstraße würde ein menschlicher Fahrer dagegen keine Ansage seines Navigationsgeräts wünschen. Zudem sind falsch-positive Detektionen, bei denen das Fahrzeug fälschli-



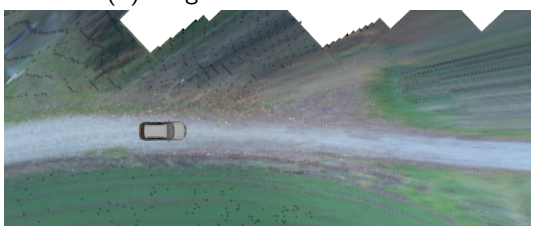
(a) Kamerabild



(b) Eingefärbtes Terrain-Grid



(c) Kamerabild



(d) Eingefärbtes Terrain-Grid

Abbildung 9.13:
Zwei Beispiele für Kreuzungen, die aufgrund der Modellierung einer Kreuzung als Schnittpunkt zweier Wege nicht erkannt werden können.

cherweise den Weg verlassen könnte, schwerwiegender als falsch-negative Detektionen, bei denen bei nächster Gelegenheit abgebogen oder gewendet werden kann. Ohne falsch-positive Ergebnisse zu erhalten, können die falsch-negativen Detektionen jedoch nur mithilfe von Kartenwissen wirklich signifikant reduziert werden.

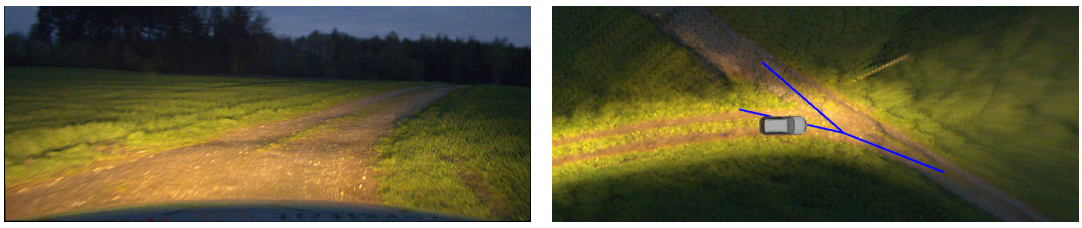
9.5.4 Laufzeit

Die implementierte Kreuzungserkennung ohne Kartenwissen arbeitet in Echtzeit. Sie betrachtet einen 60 m großen Ausschnitt aus dem eingefärbten Terrain-Grid mit 20 cm Auflösung. In diesem Grid-Ausschnitt dauert es im Mittel 10 ms, die einzelnen Farbfeatures und die Straßenwahrscheinlichkeit zu berechnen. Anschließend werden 43 ms für die Faltung der Straßenwahrscheinlichkeit mit den rotierten Wegmasken benötigt. Das Überprüfen der Kreuzungshypothesen und das Tracking beanspruchen zuletzt weitere 20 ms. Insgesamt hat die Kreuzungserkennung damit eine mittlere Laufzeit von 73 ms.

9.6 Diskussion

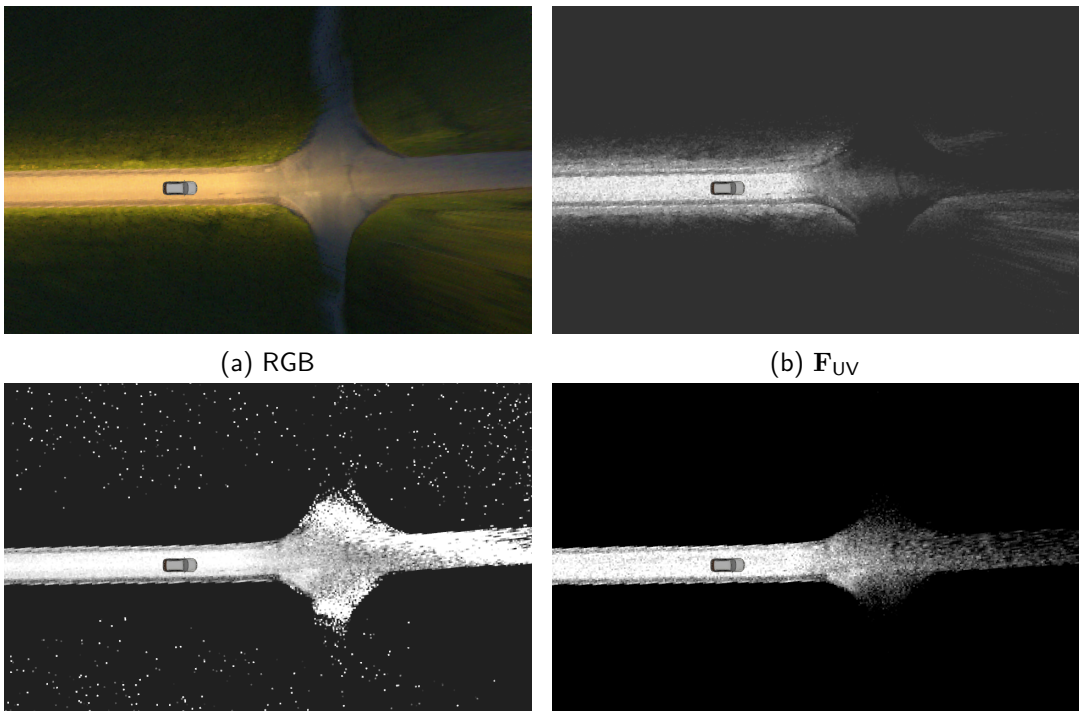
Zusammenfassend zeigt Abschnitt 9.5, dass mit der Fusion von Kamera und LiDAR erfolgreich und in Echtzeit Feldwegkreuzungen erkannt werden können. Zudem bewähren sich die im Wegetracking entwickelten Farbmodi auch in der Kreuzungserkennung: Sie können die Anzahl falsch-positiver Kreuzungsdetektionen im Vergleich zu Vorarbeiten am Institut signifikant reduzieren. Alle gezeigten Auswertungen erfolgten jedoch bei Tag. Da die Farbfeatures in der Kreuzungserkennung ausschließlich für Tageslicht-Bedingungen entwickelt und parametriert sind, stellt sich zuletzt die Frage, wie gut sie bei Dunkelheit funktionieren.

Abbildung 9.14 zeigt ein Beispiel einer Kreuzung bei Dunkelheit, die mit dem weiterentwickelten Verfahren erfolgreich detektiert wurde. Zum abgebildeten Zeitpunkt, zu dem die Kreuzung erstmals gemessen wurde, befindet sich das Fahrzeug jedoch bereits mitten in der Kreuzung. Die Kreuzung konnte erst erkannt werden, sobald die Kreuzungsäste ausreichend beleuchtet wurden. Bei automatisierter Fahrt wäre dabei ein Rangieren nötig gewesen.



(a) Kamerabild (b) Eingefärbtes Terrain-Grid

Abbildung 9.14:
Beispiel einer bei Dunkelheit erkannten Kreuzung. Gezeigt sind das Kamerabild und das eingefärbte Terrain-Grid zum Zeitpunkt des Aufsetzens des Trackings.



(a) RGB (b) $\mathbf{F}_{\text{UV}}$

(c) $\mathbf{F}_{\text{S/V}}$ (d) $\mathbf{F}_{\text{Y-I3}}$

Abbildung 9.15:
Visualisierung des Farbvergleichsfeatures $\mathbf{F}_{\text{UV}}$, des relativen Sättigungsfeatures $\mathbf{F}_{\text{S/V}}$ und des Grünfeatures $\mathbf{F}_{\text{Y-I3}}$ für eine gut einsehbare Viererkreuzung in der Dunkelheit.

Abbildung 9.15 visualisiert die Probleme des Verfahrens bei Dunkelheit anhand einer zweiten, bei Tageslicht einfach zu erkennenden Kreuzung: Im mit dem Kamerabild eingefärbten Terrain-Grid (Abbildung 9.15a) ist eine deutliche, gelbe Spur bis knapp vor das Fahrzeug zu sehen. Hierbei handelt es sich um das Abblend- und das Fernlicht des Ego-Fahrzeugs. Diese erhellen Teile der Umgebung, die im Kamerabild gelblich wahrgenommen werden. Dadurch wird die akkumulierte Farbe der Grid-Zellen verfälscht. Als Resultat schlägt der Farbvergleich mit den Grid-Zellen unter dem Fahrzeug fehl: Das Farbvergleichsfeature basierend auf dem YUV-Farbmodell schließt den kompletten weiteren Weg als möglichen Feldweg aus (Abbildung 9.15b). Das Sättigungs- (Abbildung 9.15c) und das Grünfeature (Abbildung 9.15d) weisen zwar den weiteren Weg geradeaus als Weg aus, da dieser noch durch das Fernlicht beleuchtet wird, verwerfen jedoch auch die dunklen Abzweigungen nach links und rechts.

Folglich eignen sich Farbfeatures aus RGB-Bildern zwar sehr gut zur Erkennung von Kreuzungen bei Tageslicht, jedoch nur begrenzt zur Kreuzungserkennung bei Nacht. Hierfür müssten neben (RGB-)Kamera und LiDAR weitere Sensoren, wie beispielsweise die in [Bayerl et al., 2015] verwendete Infrarot-Wärmebildkamera, fusioniert werden.

Teil IV

Fazit

10 Zusammenfassung und Ausblick

In dieser Arbeit wird eine Repräsentation der statischen Umgebung für autonome Fahrzeuge auf Feldwegen erarbeitet. Um auch komplexe Feldwege robust wahrnehmen zu können, werden die Bilder einer Kamera mit den Messungen eines LiDAR-Sensors fusioniert. Dafür werden im Anschluss an die extrinsische Kalibrierung der beiden Sensoren zunächst Konzepte zur Rohdatenfusion von Kamera und LiDAR untersucht. Letztendlich erfolgt jedoch eine Fusion auf Merkmalsebene in vier einzelnen Modulen zur Repräsentation der statischen Umgebung (vgl. Abbildung 10.1): einem Hindernis-Grid, einer Terrain-Schätzung, einem Straßen- und Wegetracking sowie einer Kreuzungserkennung. Das Hindernis-Grid nutzt primär die LiDAR-Punktwolke, um mittels eines Bayes-Filters Belegtheitswahrscheinlichkeiten für diskrete Grid-Zellen zu schätzen. Das Kamerabild ermöglicht es, zusätzlich Befahrbarkeitswahrscheinlichkeiten zu schätzen. Beide werden zu einer Hinderniswahrscheinlichkeit fusioniert. In der Terrain-Schätzung wird anhand der LiDAR-Punktwolke mittels Optimierung ein geglättetes und interpoliertes Elevationsmodell der Bodenoberfläche geschätzt. Das Kamerabild hilft dabei dynamisch zu entscheiden, welche Bereiche lokal höher aufgelöst werden sollen. Das Straßen- und Wegetracking nutzt gleichermaßen Merkmale aus dem Kamerabild und der LiDAR-Punktwolke, um in einem Optimierungsverfahren die 6D-Posen von Spurbegrenzungspunkten zu schätzen. Kantenmessungen im Kamerabild sorgen dabei für eine hohe Genauigkeit und eine weite Vorausschau. Die Bestimmung der Ausdehnung der Freifläche in der LiDAR-Punktwolke sowie Informationen aus dem Hindernis-Grid und der Terrain-Schätzung erhöhen dagegen die Robustheit bei schwierigen Sichtverhältnissen. In der Kreuzungserkennung werden schließlich die Farbinformationen aus dem Kamerabild in das Terrain-Grid integriert, um darin anhand verschiedener Farbmerkmale Feldwegkreuzungen zu erkennen. Die Ergebnisse wurden in den jeweiligen Kapiteln der Arbeit kritisch diskutiert.

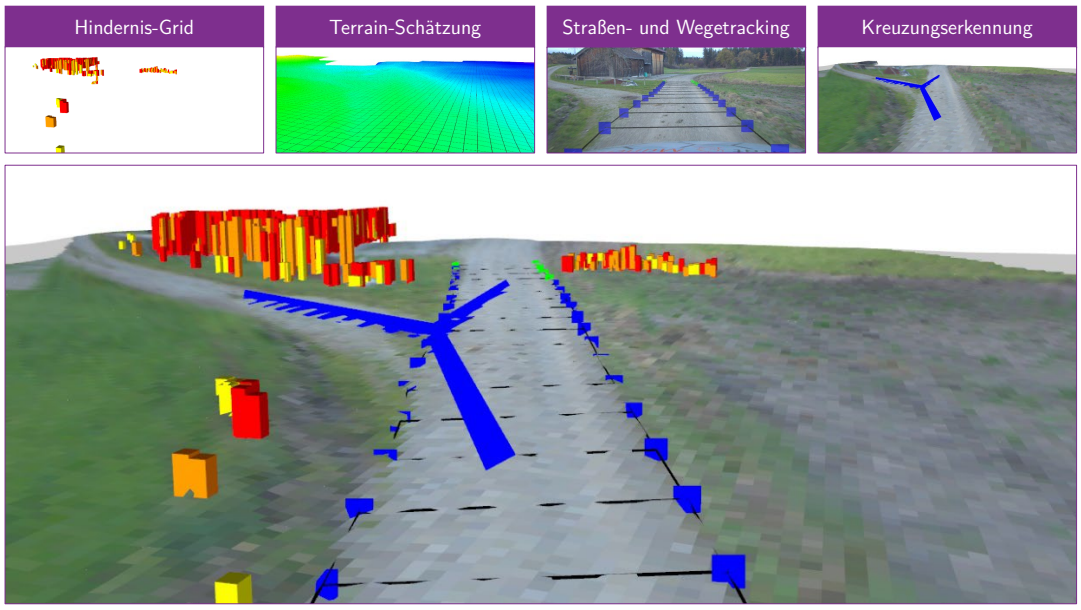


Abbildung 10.1: Entwickelte Repräsentation der statischen Umgebung.

Kontribution Während bei der Entwicklung der Umfeldrepräsentation für Feldwege auf manche Konzepte aus dem autonomen Fahren auf Straßen zurückgegriffen werden konnte, ergaben sich viele neue Herausforderungen durch die Anwendung auf Feldwegen. Zusammengefasst resultieren aus dieser Arbeit folgende wesentliche Forschungsbeiträge zum autonomen Fahren insbesondere auf Feldwegen:

1. Ein Kalibrierungsverfahren zur extrinsischen Kalibrierung von Kameras und LiDAR-Sensoren. Dafür wurde ein generisches Kalibrierungstool entwickelt, das die Kalibrierung beliebig vieler Sensoren zu einem beliebigen Referenzsensor erlaubt und leicht um weitere Sensortypen (z. B. Radar) erweitert werden kann.

2. Ein neuartiges Verfahren zur Fusion einer LiDAR-Punktwolke in die Stereoverarbeitung. Durch eine tiefe Fusion auf Kostenebene ist das Verfahren echtzeitfähig und übertrifft dabei die Genauigkeit aller echtzeitfähigen Verfahren aus der Literatur, die auf dem verwendeten Datensatz evaluiert wurden.

3. Ein Hindernis-Grid für Feldwege mit hoher Vegetation (z. B. auf Grünstreifen in der Mitte des Wegs). Durch die Bestimmung einer Befahrbarkeitswahrscheinlichkeit wird zwischen befahrbarer Vegetation und unpassierbaren Hindernissen unterschieden und eine tatsächliche Hinderniswahrscheinlichkeit geschätzt.

4. Eine Terrain-Schätzung für Feldwege mit kleinen negativen Hindernissen wie Schlaglöchern oder Gräben. Durch die dynamische Anpassung der Auflösung einzelner Kacheln einer Grid-basierten Terrain-Schätzung bleibt diese trotz einer (lokal) erhöhten Auflösung echtzeitfähig.

5. Robuste Messverfahren zum Straßen- und Wegetracking und ihre Integration in ein Optimierungsverfahren. Die entwickelten Messverfahren im Kamerabild, der LiDAR-Punktwolke und dem Umgebungsmodell eignen sich dabei gleichermaßen für Feldwege und (markierte) Straßen. Für die Messung im Kamerabild wurden spezielle Bild-Farbmodi entwickelt, die den Kontrast eines Wegs zu seiner Umgebung erhöhen. Diese konnten sich auch bei der Erkennung von Kreuzungen in einem mit dem Kamerabild eingefärbten Terrain-Grid bewähren. Insgesamt ermöglichen die entwickelten Messverfahren die autonome Fahrt auf komplexen Feldwegen auch bei herausfordernden Wetterbedingungen.

Weiterführende Arbeiten In weiterführenden Arbeiten könnte die entwickelte Umgebungsrepräsentation zur Durchführung von Missionen verwendet werden, um autonom ein bestimmtes Ziel anzufahren. Dazu wird eine Verhaltensschicht benötigt, die an Kreuzungen einen bestimmten zu verfolgenden Kreuzungsast auswählt, der wiederum in das Straßen- und Wegetracking rückgeführt werden müsste, um das Tracking auf diesen Kreuzungsast zu lenken. Da für die Entscheidung über die einzuschlagende Richtung Kartenwissen benötigt wird, könnte dieses dann auch im Straßen- und Wegetracking sowie in der Kreuzungserkennung zu Hilfe genommen werden. Dabei könnten auch bei schlechter Lokalisierung die grobe Position von Kreuzungen und die Richtung von Straßen und Kreuzungen aus einer Karte als Vorwissen verwendet und lediglich die genauen Fahrspurränder nachgeschätzt werden.

Symbolverzeichnis

Abkürzungen

CNN	Neuronales Faltungsnetz, engl. *Convolutional Neural Network*
EKF	Erweitertes Kalman-Filter
GNSS	Globales Navigationssatellitensystem, engl. *Global Navigation Satellite System*
HLS	Farbton, Helligkeit, Sättigung, engl. *Hue, Lightness, Saturation*
HOG	*Histogram of Oriented Gradients*
HSI	Farbton, Sättigung, Lichtintensität, engl. *Hue, Saturation, Intensity*
HSV	Farbton, Sättigung, Hellwert, engl. *Hue, Saturation, Value*
HTM	Homogene Transformationsmatrix
INS	Inertiales Navigationssystem, engl. *Inertial Navigation System*
LBP	Lokale binäre Muster, engl. *Local Binary Pattern*
LiDAR	*Light Detection And Ranging*
MAE	Mittlerer absoluter Fehler, engl. *Mean Absolute Error*
MAP	Maximum-a-posteriori
MarVEye	Multifokales aktives / reaktives Fahrzeug-Auge, engl. *Multifocal active / reactive Vehicle Eye*
MuCAR-3	*Munich Cognitive Autonomous Robot Car 3rd Generation*
MuCAR-4	*Munich Cognitive Autonomous Robot Car 4th Generation*
Radar	*Radio Detection And Ranging*
RANSAC	*Random Sample Consensus*
RGB	Rot, Grün, Blau
RMSE	Wurzel des mittleren quadratischen Fehlers, engl. *Root Mean Square Error*
ROI	Region von Interesse, engl. *Region of Interest*
ROS	*Robot Operating System*
RTK	Echtzeitkinematik, engl. *Real Time Kinematic*
SGM	*Semi-Global Matching*
SIFT	*Scale-Invariant Feature Transform*
SVM	*Support Vector Machine*
TAS	Institut für Technik Autonomer Systeme
UniBwM	Universität der Bundeswehr München

Konventionen

Basisnotation

$(\cdot)_{[i]}$	ite Komponente
$\dot{(\cdot)}$	Geschwindigkeit
t	Tupel

op	Operator
$(\cdot)^\mathsf{T}$	Transponiert
$\mathcal{M}$	Menge
$\boldsymbol{M}$	Dreidimensionale Matrix
M	Eigenname aus Literatur
$\mathbf{M}$	Matrix
s	Skalar
$\mathbf{v}$	Vektor

Allgemein

p	Bild-Pixel $\mathsf{p} = (u, v)$
p	Wahrscheinlichkeit
$\mathbf{p}$	3D-Punkt ${}^{\mathsf{coord}}\mathbf{p} = \begin{bmatrix} x & y & z \end{bmatrix}^\mathsf{T}$ im Koordinatensystem *coord*
$\mathbf{q}$	3D-Orientierung als Quaternion $\mathbf{q} = \begin{bmatrix} q_\mathsf{x} & q_\mathsf{y} & q_\mathsf{z} & q_\mathsf{w} \end{bmatrix}^\mathsf{T}$

Kalibrierung

c	Hauptpunkt $\mathsf{c} = (c_\mathsf{u}, c_\mathsf{v})$ in px
f	Brennweite $\mathsf{f} = (f_\mathsf{u}, f_\mathsf{v})$ in px
s	Pixelabstand (engl. *Pixel Pitch*) $\mathsf{s} = (s_\mathsf{x}, s_\mathsf{y})$
$\mathrm{proj}(\cdot)$	Projektion eines Punkts ins Kamerabild
b	Basisbreite
f	Brennweite in mm

Fahrzeug

h_gc	Bodenfreiheit (engl. *Ground Clearance*)

Klothoiden

$\mathring{c}$	Lineare Krümmungsänderung
c	Krümmung
l	Lauflänge

Farbkanäle

$\mathbf{I3}_\mathsf{I1I2I3}$	I3-Kanal aus dem I1I2I3-Farbraum
$\mathbf{U}_\mathsf{YUV}$	Chrominanz (Farbanteil) aus dem YUV-Farbmodell
$\mathbf{V}_\mathsf{YUV}$	Chrominanz (Farbanteil) aus dem YUV-Farbmodell
$\mathbf{Y}_\mathsf{YUV}$	Grauwert, entspricht Luminanz (Leuchtdichte) aus dem YUV-Farbmodell
$\mathbf{H}_\mathsf{HSI}$	Farbton (engl. *Hue*) aus dem HSI-Farbraum
$\mathbf{I}_\mathsf{HSI}$	Lichtintensität (engl. *Intensity*) aus dem HSI-Farbraum
$\mathbf{S}_\mathsf{HSI}$	Sättigung aus dem HSI-Farbraum
$\mathbf{H}_\mathsf{HSV}$	Farbton (engl. *Hue*) aus dem HSV-Farbraum
$\mathbf{S}_\mathsf{HSV}$	Sättigung aus dem HSV-Farbraum
$\mathbf{V}_\mathsf{HSV}$	Hellwert (engl. *Value*) aus dem HSV-Farbraum
$\mathbf{B}_\mathsf{RGB}$	Blauwert aus dem RGB-Farbraum
$\mathbf{G}_\mathsf{RGB}$	Grünwert aus dem RGB-Farbraum
$\mathbf{R}_\mathsf{RGB}$	Rotwert aus dem RGB-Farbraum

Koordinaten, Transformationen

${}^\mathsf{to}\mathbf{H}_\mathsf{from}$	Homogene Transformationsmatrix von *from* nach *to*
$\mathrm{euler}(\cdot)$	Umrechnung einer Quaternion in Euler-Winkel
$\mathrm{quat}(\cdot)$	Berechnung einer Quaternion aus Euler-Winkeln

$\underline{\mathbf{R}}$	Homogene Rotationsmatrix
$\underline{\mathbf{T}}$	Homogene Translationsmatrix
$\mathbf{R}$	Rotationsmatrix
Φ	Rollwinkel (engl. *Roll*)
Ψ	Gierwinkel (engl. *Yaw*)
Θ	Nickwinkel (engl. *Pitch*)
$\mathbf{t}$	Translationsvektor

Kreuzungserkennung

$\mathbf{F}_{\mathsf{GR}}$	Grünfeature aus der Green-Ratio
$\mathbf{F}_{\mathsf{S/V}}$	Relatives Sättigungsfeature
$\mathbf{F}_{\mathsf{S}}$	Absolutes Sättigungsfeature
$\mathbf{F}_{\mathsf{UV}}$	Farbvergleichsfeature (aus $\mathbf{U}_{\mathsf{YUV}}, \mathbf{V}_{\mathsf{YUV}}$)
$\mathbf{F}_{\mathsf{Y\text{-}I3}}$	Grünfeature aus $\mathbf{Y}_{\mathsf{YUV}} - \mathbf{I3}_{\mathsf{I1I2I3}}$
$\mathbf{D}$	Farbunterschied einer Grid-Zelle zur ROI unter dem Fahrzeug

LiDAR-SGM

$\boldsymbol{C}_{\mathsf{L}}$	LiDAR-Matching-Kosten
$\boldsymbol{C}_{\mathsf{S}}$	Stereo-Matching-Kosten
$\boldsymbol{C}$	(Fusionierte) Matching-Kosten
$\mathbf{D}$	Disparitätsbild
α	LiDAR-Einflussfaktor
d	Disparität

Mathematische Definitionen

$\exists$	Existenzquantor: $\exists x$ – *es existiert ein* x
$\forall$	Allquantor: $\forall x$ – *für alle* x
$\nexists$	Negierter Existenzquantor: $\nexists x$ – *es existiert kein* x
$\lvert\cdot\rvert$	Betrag
$\mathrm{d}(\cdot)$	Euklidische Distanz
$\mathbf{I}$	Einheitsmatrix

Hindernis-Grid

$\tilde{c}_k^{(\mathsf{c})}$	Klassifizierung der Zelle c in Zeitschritt k
$p(\mathrm{obstacle}^{(\mathsf{c})})$	Hinderniswahrscheinlichkeit
$p(\mathrm{occ}^{(\mathsf{c})})$	Belegtheitswahrscheinlichkeit
$p(\mathrm{trav}^{(\mathsf{c})})$	Befahrbarkeitswahrscheinlichkeit
c	Zell-Index $\mathsf{c} = (c_{\mathsf{i}}, c_{\mathsf{j}})$
logit	Natürlicher Logarithmus der Chance
odds	Chance (engl. *Odds*)
$\tilde{h}_{\mathsf{max},k}^{(\mathsf{c})}$	In Zeitschritt k maximal gemessene Höhe in Zelle c
$h_{\mathsf{max}}^{(\mathsf{c})}$	Geschätzte maximale Höhe des Hindernisses in Zelle c
$\tilde{h}_{\mathsf{min},k}^{(\mathsf{c})}$	In Zeitschritt k minimal gemessene Höhe in Zelle c
$h_{\mathsf{min}}^{(\mathsf{c})}$	Geschätzte minimale Höhe des Hindernisses in Zelle c

Optimierung

$\rho_{\mathsf{C}}(\cdot)$	Cauchy-Verlustfunktion
$\rho_{\mathsf{H}}(\cdot)$	Huber-Verlustfunktion
$\rho(\cdot)$	Verlustfunktion
$\mathbf{r}$	Residuum

$\mathbf{w}$	Gewichtungsfaktoren, indiziert als $\mathbf{w}_{[\text{name}]}$

Bayes'sche Filter

$\mathbf{K}$	Kalman-Verstärkungsmatrix
$\mathbf{R}$	Kovarianzmatrix des Messrauschens
$\mathbf{Q}$	Kovarianzmatrix des Prozessrauschens
$\mathbf{P}$	Kovarianzmatrix des Schätzfehlers
$\hat{(\cdot)}$	Geschätzte Werte
$(\cdot)^*$	Prädizierte Werte
$\tilde{(\cdot)}$	Gemessene Werte

Straßen- und Wegetracking

$s_i^{(k)}$	Laterale Verschiebung der Spurbegrenzung k zur Referenz-Spurbegrenzung am longitudinalen Index i
$w_i^{(k)}$	Breite der Fahrspur zwischen den Spurbegrenzungen k und $k-1$ am longitudinalen Index i
w_i	Breite der gesamten Fahrbahn am longitudinalen Index i
$\mathbf{x}_i^{(0)}$	Referenz-Spurbegrenzungspunkt mit longitudinalem Index i
$\mathbf{x}_i^{(k)}$	Spurbegrenzungspunkt mit longitudinalem Index i auf Spurbegrenzung k
$\mathbf{c}$	Kosten bestehend aus mehreren Residuen $\mathbf{r}$
$\mathbf{n}$	Normale $\mathbf{n} = \begin{bmatrix} n_\mathsf{x} & n_\mathsf{y} & n_\mathsf{z} \end{bmatrix}^\mathsf{T}$ eines Punkts

Zustandsraumdarstellung

$\mathbf{D}$	System Eingangsrauschvektor
$\mathbf{B}$	Eingangsmatrix, zeitdiskret
$\mathbf{u}$	Systemeingang
$\mathbf{w}$	Messrauschvektor
$\mathbf{y}$	Messvektor
$\mathbf{s}$	Störvektor
$\mathbf{x}$	Zustandsvektor
$\mathbf{v}$	Systemrauschvektor, zeitdiskret
$\mathbf{\Phi}$	Dynamikmatrix, zeitdiskret
$\mathbf{C}$	Messmatrix

Terrain-Schätzung

$\mathbf{R}_k^{(\mathsf{c_m})^{-1}}$	Information der gemessenen Terrain-Höhe zum Zeitschritt k
$\mathbf{P}_k^{(\mathsf{c_m})^{-1}}$	Information der akkumulierten Terrain-Höhe zum Zeitschritt k
$\mathsf{c_m}$	Zell-Index $\mathsf{c_m} = (c_{\mathsf{m,i}}, c_{\mathsf{m,j}})$ im Mess-Grid
c	Zell-Index $\mathsf{c} = (c_\mathsf{i}, c_\mathsf{j})$ im Glättungs-Grid
$\tilde{h}_k^{(\mathsf{c_m})}$	Gemessene Terrain-Höhe zum Zeitschritt k
$h_k^{(\mathsf{c_m})}$	Akkumulierte Terrain-Höhe zum Zeitschritt k
$m_{\mathsf{x/y}}^{(\mathsf{c})}$	Geschätzte Terrain-Steigung in x-/y-Richtung
$h^{(\mathsf{c})}$	Geschätzte Terrain-Höhe
σ	Erwartete Standardabweichung des Messrauschens

Zeit

k	Diskreter Zeitschritt
t	Kontinuierlicher Zeitpunkt

Eigene Veröffentlichungen

Kallwies, J., Engler, T., Forkel, B., und Wuensche, H.-J. (2020a). Triple-SGM: Stereo Processing using Semi-Global Matching with Cost Fusion. In *Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV)*, Seiten 192–200.

Kallwies, J., Forkel, B., und Wuensche, H.-J. (2020b). Determining and Improving the Localization Accuracy of AprilTag Detection. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 8288–8294.

Forkel, B., Kallwies, J., und Wuensche, H.-J. (2021a). Combined Road Tracking for Paved Roads and Dirt Roads: Framework and Image Measurements. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 1326–1331.

Forkel, B., Kallwies, J., und Wuensche, H.-J. (2021b). Probabilistic Terrain Estimation for Autonomous Off-Road Driving. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 13864–13870.

Forkel, B. und Wuensche, H.-J. (2022a). Combined Road Tracking for Paved Roads and Dirt Roads: LiDAR Measurements and Image Color Modes. In *Proceedings of International Conference on Information Fusion (FUSION)*.

Forkel, B. und Wuensche, H.-J. (2022b). Dynamic Resolution Terrain Estimation for Autonomous (Dirt) Road Driving Fusing LiDAR and Vision. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 1181–1187.

Forkel, B. und Wuensche, H.-J. (2023). LiDAR-SGM: Semi-Global Matching on LiDAR Point Clouds and their Cost-based Fusion into Stereo Matching. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 2841–2847.

Berthold, P., Forkel, B., Michaelis, M., und Wuensche, H.-J. (2023). Joint Multiple Extended Object Tracking by Optimizing the Global Coherence. In *Combined IEEE Symposium Sensor Data Fusion and International Conference on Multisensor Fusion and Integration (SDF-MFI)*.

Forkel, B., Berthold, P., und Maehlisch, M. (2024). Taking Advantage of Road Users Occluding the Road: Supporting Camera-based Road Tracking in Shared Spaces using Radar Doppler Measurements. In *Proceedings of International Conference on Information Fusion (FUSION)*.

Literaturverzeichnis

3D Mapping Solutions GmbH (2024). Ultra HD Maps. URL: https://www.3d-mapping.de/en/our-service/ultra-hd-maps/.

Agarwal, S., Mierle, K., und The Ceres Solver Team (2020). Ceres Solver. URL: https://github.com/ceres-solver/ceres-solver.

Apostoloff, N. und Zelinsky, A. (2003). Robust Vision based Lane Tracking using Multiple Cues and Particle Filtering. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 558–563.

Aquilio, A. S. (2006). *A Framework for Dynamic Terrain with Application in Off-Road Ground Vehicle Simulations*. Dissertation, Georgia State University, Department of Computer Science.

Bacha, A., Bauman, C., Faruque, R., Fleming, M., Terwelp, C., Reinholtz, C., Hong, D., Wicks, A., Alberi, T., Anderson, D., Cacciola, S., Currier, P., Dalton, A., Farmer, J., Hurdus, J., Kimmel, S., King, P., Taylor, A., Covern, D. V., und Webster, M. (2008). Odin: Team VictorTango's Entry in the DARPA Urban Challenge. *Journal of Field Robotics*, 25(8):467–492.

Badino, H., Franke, U., und Pfeiffer, D. (2009). The Stixel World - A Compact Medium Level Representation of the 3D-World. In *Proceedings of Symposium of the German Association for Pattern Recognition*, Seiten 51–60.

Badino, H., Huber, D., und Kanade, T. (2011). Integrating LIDAR into Stereo for Fast and Improved Disparity Computation. In *Proceedings of International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, Seiten 405–412.

Banz, C., Pirsch, P., und Blume, H. (2012). Evaluation of Penalty Functions for Semi-Global Matching Cost Aggregation. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXIX-B3.

Bar-Shalom, Y., Li, X., und Kirubarajan, T. (2001). *Estimation with Applications To Tracking and Navigation: Theory, Algorithms and Software*. John Wiley & Sons.

Bares, J., Hebert, M., Kanade, T., Krotkov, E., Mitchell, T., Simmons, R., und Whittaker, W. (1989). Ambler: An Autonomous Rover for Planetary Exploration. *Computer*, 22(6):18–26.

Bayerl, S. F. X., Luettel, T., und Wuensche, H.-J. (2015). Following Dirt Roads at Night-Time: Sensors and Features for Lane Recognition and Tracking. In *Proceedings of 7th Workshop On Planning, Perception and Navigation for Intelligent Vehicles (PPNIV), IEEE/RSJ International Conference on Intelligent Robots and Systems*, Seiten 117–122.

Bayerl, S. F. X. und Wuensche, H.-J. (2014). Detection and Tracking of Rural Crossroads Combining Vision and LiDAR Measurements. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, Seiten 1274–1279.

Behringer, R., Travis, W., Daily, R., Bevly, D., Kubinger, W., Herzner, W., und Fehlberg, V. (2005). RASCAL - An Autonomous Ground Vehicle for Desert Driving in the DARPA Grand Challenge 2005. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, Seiten 644–649.

Belter, D. und Skrzypczyński, P. (2011). Rough Terrain Mapping and Classification for Foothold Selection in a Walking Robot. *Journal of Field Robotics*, 28(4):497–528.

Belter, D., Łabcki, P., und Skrzypczyński, P. (2012). Estimating Terrain Elevation Maps from Sparse and Uncertain Multi-Sensor Data. In *Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Seiten 715–722.

Beltrán, J., Guindel, C., und García, F. (2022). Automatic Extrinsic Calibration Method for LiDAR and Camera Sensor Setups. *IEEE Transactions on Intelligent Transportation Systems*, Seiten 17677–17689.

Berthold, P., Michaelis, M., Luettel, T., Meissner, D., und Wuensche, H.-J. (2020). Deriving Spatial Occupancy Evidence from Radar Detection Data. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 831–836.

Bienemann, A. und Wuensche, H.-J. (2023). Model Predictive Control for Autonomous Vehicle Following. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*.

Bouzouraa, M. E. (2011). *Belegungskartenbasierte Umfeldwahrnehmung in Kombination mit objektbasierten Ansätzen für Fahrerassistenzsysteme*. Dissertation, Technische Universität München, Fakultät für Elektrotechnik und Informationstechnik.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Broggi, A., Cardarelli, E., Cattani, S., und Sabbatelli, M. (2013). Terrain Mapping for Off-road Autonomous Ground Vehicles Using Rational B-Spline Surfaces and Stereo Vision. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 648–653.

Burgard, W. und Hebert, M. (2008). World Modeling. In Siciliano und Khatib [2008], Seiten 585–610.

Burger, P., Naujoks, B., und Wuensche, H.-J. (2019). Unstructured Road SLAM using Map Predictive Road Tracking. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, 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 *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 2150–2156.

Caraffi, C., Cattani, S., und Grisleri, P. (2007). Off-Road Path and Obstacle Detection Using Decision Networks and Stereo Vision. *IEEE Transactions on Intelligent Transportation Systems*, 8(4):607–618.

Chaturvedi, P., Malcolm, A., und Ibanez Guzman, J. (2004). Real-Time Road Following In Natural Terrain. In *Proceedings of IEEE Conference on Cybernetics and Intelligent Systems*, Volume 2, Seiten 815–820.

Cheng, X., Zhong, Y., Dai, Y., Ji, P., und Li, H. (2019). Noise-Aware Unsupervised Deep Lidar-Stereo Fusion. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seiten 6332–6341.

Choe, J., Joo, K., Imtiaz, T., und Kweon, I. S. (2021). Volumetric Propagation Network: Stereo-LiDAR Fusion for Long-Range Depth Estimation. *IEEE Robotics and Automation Letters*, 6(3):4672–4679.

Courtois, H. und Aouf, N. (2017). Fusion of Stereo and Lidar Data for Dense Depth Map Computation. In *Proceedings of Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*, Seiten 186–191.

Crane III, C. D., Armstrong II, D. G., Touchton, R., Galluzzo, T., Solanki, S., Lee, J., Kent, D., Ahmed, M., Montane, R., Ridgeway, S., Velat, S., Garcia, G., Griffis, M., Gray, S., Washburn, J., und Routson, G. (2006). Team CIMAR's NaviGator: An Unmanned Ground Vehicle for the 2005 DARPA Grand Challenge. *Journal of Field Robotics*, 23(8):599–623.

Crisman, J. und Thorpe, C. (1993). SCARF: A Color Vision System that Tracks Roads and Intersections. *IEEE Transactions on Robotics and Automation*, 9(1):49–58.

Darms, M. (2015). Fusion umfelderfassender Sensoren. In Winner et al. [2015], Seiten 439–451.

Dickmanns, E. D. (2007). *Dynamic Vision for Perception and Control of Motion*. Springer.

Dickmanns, E. D. und Zapp, A. (1987). A Curvature-based Scheme for Improving Road Vehicle Guidance by Computer Vision. In *Mobile Robots I*, Volume 0727, Seiten 161–168. International Society for Optics and Photonics, SPIE.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., und Koltun, V. (2017). CARLA: An Open Urban Driving Simulator. In *Proceedings of the Conference on Robot Learning (CoRL)*.

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 *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Seiten 1532–1538.

Dryanovski, I., Morris, W., und Xiao, J. (2010). Multi-Volume Occupancy Grids: an Efficient Probabilistic 3D Mapping Model for Micro Aerial Vehicles. *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Seiten 1553–1559.

Dumont, M. (2015). *Real-Time View Interpolation for Eye Gaze Corrected Video Conferencing*. Dissertation, Hasselt University, transnationale Universiteit Limburg.

Durrant-Whyte, H. und Henderson, T. C. (2008). Multisensor Data Fusion. In Siciliano und Khatib [2008], Seiten 585–610.

Elfes, A. (1989). Using Occupancy Grids for Mobile Robot Perception and Navigation. *Computer*, 22(6):46–57.

Fankhauser, P. und Hutter, M. (2016). A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation. In Koubaa, A., Herausgeber, *Robot Operating System (ROS) – The Complete Reference (Volume 1)*, Volume 625 of *Studies in Computational Intelligence*, Kapitel 5, Seiten 99–120. Springer.

Fernandez, D. und Price, A. (2005). Visual Detection and Tracking of Poorly Structured Dirt Roads. In *Proceedings of International Conference on Advanced Robotics (ICAR)*, Seiten 553–560.

FGSV (1993). *Richtlinien für die Markierung von Straßen Teil 1: Abmessungen und geometrische Anordnung von Markierungszeichen (RMS-1)*. Forschungsgesellschaft für Straßen- und Verkehrswesen.

Fischler, M. A. und Bolles, R. C. (1981). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM (CACM)*, 24(6):381–395.

Fleischmann, P., Kneip, J., und Berns, K. (2016). An Adaptive Detection Approach for Autonomous Forest Path Following using Stereo Vision. In *Proceedings of IEEE International Conference on Control, Automation, Robotics and Vision (ICARCV)*.

Franke, U., Loose, H., und Knoppel, C. (2007). Lane Recognition on Country Roads. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 99–104.

Gallup, D., Frahm, J.-M., Mordohai, P., und Pollefeys, M. (2008). Variable Baseline/Resolution Stereo. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Geiger, A., Lenz, P., Stiller, C., und Urtasun, R. (2013). Vision meets Robotics: The KITTI Dataset. *The International Journal of Robotics Research*, 32(11):1231–1237.

Geiger, A., Roser, M., und Urtasun, R. (2010). Efficient Large-Scale Stereo Matching. In *Proceedings of Asian Conference on Computer Vision (ACCV)*, Seiten 25–38.

Gevers, T. und Smeulders, A. W. (1999). Color-based object recognition. *Pattern Recognition*, 32(3):453–464.

Gotzig, H. und Geduld, G. (2015). LIDAR-Sensorik. In Winner et al. [2015], Seiten 317–334.

Grewe, R., Hohm, A., Hegemann, S., Lueke, S., und Winner, H. (2012). Towards a Generic and Efficient Environment Model for ADAS. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 316–321.

Guindel, C., Beltrán, J., Martín, D., und García, F. (2017). Automatic Extrinsic Calibration for Lidar-Stereo Vehicle Sensor Setups. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*.

Hadsell, R., Bagnell, J. A., Huber, D., und Hebert, M. (2010). Space-carving Kernels for Accurate Rough Terrain Estimation. *The International Journal of Robotics Research*, 29(8):981–996.

Han, J., Kim, D., Lee, M., und Sunwoo, M. (2012). Enhanced Road Boundary and Obstacle Detection Using a Downward-Looking LIDAR Sensor. *IEEE Transactions on Vehicular Technology*, 61(3):971–985.

He, Y. (2000). *Real-Time Visualization of Dynamic Terrain for Ground Vehicle Simulation*. Dissertation, Graduate College of The University of Iowa.

Himmelsbach, M., Schneider, S., und Wuensche, H.-J. (2011). A Comparison of Error Metrics for Extrinsic Calibration and Fusion of Camera and Multi-Layer LIDAR. In *Proceedings of IASTED International Conference on Signal Processing, Pattern Recognition, and Applications (SPPRA)*.

Himmelsbach, M., von Hundelshausen, F., Luettel, T., Manz, M., Mueller, A., Schneider, S., und Wuensche, H.-J. (2009). Team MuCAR-3 at C-ELROB 2009. In *Proceedings of 1st Workshop on Field Robotics, Civilian European Land Robot Trial 2009*.

Himmelsbach, M., von Hundelshausen, F., und Wuensche, H.-J. (2010). Fast Segmentation of 3D Point Clouds for Ground Vehicles. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 560–565.

Hirschmüller, H. (2005). Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seiten 807–814.

Hirschmüller, H., Buder, M., und Ernst, I. (2012). Memory Efficient Semi-Global Matching. *International Society for Photogrammetry and Remote Sensing*, I-3:371–376.

Hirschmüller, H. und Scharstein, D. (2007). Evaluation of Cost Functions for Stereo Matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hornung, A., Wurm, K., Bennewitz, M., Stachniss, C., und Burgard, W. (2013). OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Autonomous Robots*, 34:189–206.

Huber, P. J. (1964). Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73—101.

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.

Jaspers, H., Himmelsbach, M., und Wuensche, H.-J. (2017). Multi-Modal Local Terrain Maps from Vision and LiDAR. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 1119–1125.

Jaspers, H. und Wuensche, H.-J. (2014). Fast and Robust B-spline Terrain Estimation for Off-Road Navigation with Stereo Vision. In *Proceedings of IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, Seiten 140–145.

Kallwies, J. (2023). Multi-Camera-based Environmental Perception for Autonomous Vehicles. Unveröffentlichter Entwurf der Dissertation vom 24.07.2023, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.

Kammel, S., Ziegler, J., Pitzer, B., Werling, M., Gindele, T., Jagzent, D., Schröder, J., Thuy, M., Goebl, M., Hundelshausen, F. v., Pink, O., Frese, C., und Stiller, C. (2008). Team AnnieWAY's Autonomous System for the 2007 DARPA Urban Challenge. *Journal of Field Robotics*, 25(9):615–639.

Korah, T., Medasani, S., und Owechko, Y. (2011). Strip Histogram Grid for Efficient LIDAR Segmentation from Urban Environments. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Seiten 74–81.

Krogius, M., Haggenmiller, A., und Olson, E. (2019). Flexible Layouts for Fiducial Tags. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Seiten 1898–1903.

Kubertschak, T., Maehlisch, M., und Wuensche, H.-J. (2014). Towards a unified architecture for mapping static environments. In *Proceedings of International Conference on Information Fusion (FUSION)*.

Kuhnert, K.-D. (1988). *Zur Echtzeit-Bildfolgenanalyse mit Vorwissen*. Dissertation, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechik, Institut für Meßtechnik.

Kushner, T. R. und Puri, S. (1987). Progress In Road Intersection Detection For Autonomous Vehicle Navigation. In *Mobile Robots II*, Volume 852, Seiten 19–24. International Society for Optics and Photonics, SPIE.

Kweon, I. S. und Kanade, T. (1992). High-Resolution Terrain Map from Multiple Sensor Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):278–292.

Kálmán, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Transactions of the American Society of Mechanical Engineers (ASME) – Journal of Basic Engineering*, 82:35–45.

Lang, T., Plagemann, C., und Burgard, W. (2007). Adaptive Non-Stationary Kernel Regression for Terrain Modeling. In *Proceedings of International Conference on Robotics: Science and Systems (RSS)*.

Levenberg, K. (1944). A Method for the Solution of Certain Problems in Least Squares. *Quarterly of Applied Mathematics*, 2(2):164–168.

Lewis, D. B., Keller, J. M., Popescu, M., und Stone, K. (2012). Dirt Road Segmentation Using Color and Texture Features in Color Imagery. In *Proceedings of IEEE Symposium on Computational Intelligence for Security and Defence Applications (CISDA)*.

Maddern, W. und Newman, P. (2016). Real-time probabilistic fusion of sparse 3D LIDAR and dense stereo. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Seiten 2181–2188.

Mai, N.-A.-M., Duthon, P., Khoudour, L., Crouzil, A., und Velastin, S. A. (2021). Sparse LiDAR and Stereo Fusion (SLS-Fusion) for Depth Estimation and 3D Object Detection. In *Proceedings of International Conference of Pattern Recognition Systems (ICPRS)*, Seiten 150–156.

Manz, M. (2013). *Modellbasierte visuelle Wahrnehmung zur autonomen Fahrzeugführung*. Dissertation, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.

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 *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 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 of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 2410–2415.

Marquardt, D. W. (1963). An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441.

Matthaei, R., Bagschik, G., Rieken, J., und Maurer, M. (2014). Stationary Urban Environment Modeling using Multi-Layer-Grids. In *Proceedings of International Conference on Information Fusion (FUSION)*.

Meng, H., Zhong, C., Gu, J., und Chen, G. (2021). A GPU-accelerated Deep Stereo-LiDAR Fusion for Real-time High-precision Dense Depth Sensing. In *Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Seiten 523–528.

Menze, M. und Geiger, A. (2015). Object Scene Flow for Autonomous Vehicles. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seiten 3061–3070.

Metzger, K. A., Mortimer, P., und Wuensche, H.-J. (2021). A Fine-Grained Dataset and its Efficient Semantic Segmentation for Unstructured Driving Scenarios. In *International Conference on Pattern Recognition (ICPR)*, Seiten 7892–7899.

Montemerlo, M., Becker, J., Bhat, 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., 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. (2004). A Multi-Resolution Pyramid for Outdoor Robot Terrain Perception. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence*, Seiten 464–469.

Moravec, H. und Elfes, A. (1985). High Resolution Maps from Wide Angle Sonar. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 116–121.

Moravec, H. P. (1996). Robot Spatial Perception by Stereoscopic Vision and 3D Evidence Grids. Technical Report CMU-RI-TR-96-34, Carnegie Mellon University.

Morris, A., Silver, D., Ferguson, D., und Thayer, S. (2005). Towards Topological Exploration of Abandoned Mines. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 2117–2123.

Mortimer, P., Hagmanns, R., Granero, M., Luettel, T., Petereit, J., und Wuensche, H.-J. (2024). The GOOSE Dataset for Perception in Unstructured Environments. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 14838–14844.

Mueller, A., Himmelsbach, M., Luettel, T., von Hundelshausen, F., und Wuensche, H.-J. (2011). GIS-Based Topological Robot Localization through LIDAR Crossroad Detection. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, Seiten 2001–2008.

MVTec Software GmbH (2023). *HALCON/HDevelop Operator Reference*. Version 23.05.

Mysliwetz, B. (1990). *Parallelrechner-basierte Bildfolgen-Interpretation zur autonomen Fahrzeugführung*. Dissertation, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechik.

Nuss, D., Stuebler, M., und Dietmayer, K. (2014). Consistent Environmental Modeling by use of Occupancy Grid Maps, Digital Road Maps, and Multi-Object Tracking. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 1371–1377.

O'Callaghan, S. T. und Ramos, F. T. (2012). Gaussian process occupancy maps. *The International Journal of Robotics Research*, 31(1):42–62.

Ohta, Y.-I., Kanade, T., und Sakai, T. (1980). Color Information for Region Segmentation. *Computer Graphics and Image Processing*, 13(3):222–241.

Olson, E. (2011). AprilTag: A robust and flexible visual fiducial system. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 3400–3407.

Ort, T., Paull, L., und Rus, D. (2018). Autonomous Vehicle Navigation in Rural Environments Without Detailed Prior Maps. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 2040–2047.

Park, J., Yoo, H., und Wang, Y. (2022). Drivable Dirt Road Region Identification Using Image and Point Cloud Semantic Segmentation Fusion. *IEEE Transactions on Intelligent Transportation Systems*, 23(8):13203–13216.

Park, K., Kim, S., und Sohn, K. (2018). High-Precision Depth Estimation with the 3D LiDAR and Stereo Fusion. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 2156–2163.

Payeur, P., Hebert, P., Laurendeau, D., und Gosselin, C. (1997). Probabilistic Octree Modeling of a 3-D Dynamic Environment. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 1289–1296.

Pfeiffer, D. und Franke, U. (2011). Towards a Global Optimal Multi-Layer Stixel Representation of Dense 3D Data. In *Proceedings of British Machine Vision Conference (BMVC)*.

Plagemann, C., Mischke, S., Prentice, S., Kersting, K., Roy, N., und Burgard, W. (2009). A Bayesian Regression Approach to Terrain Mapping and an Application to Legged Robot Locomotion. *Journal of Field Robotics*, 26(10):789–811.

Ramos, F. und Ott, L. (2016). Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent. *The International Journal of Robotics Research*, 35(14):1717–1730.

Rasmussen, C. (2002). Combining Laser Range, Color, and Texture Cues for Autonomous Road Following. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Volume 4, Seiten 4320–4325.

Rasmussen, C. (2006). A Hybrid Vision + Ladar Rural Road Follower. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 156–61.

Rasmussen, C. und Korah, T. (2005). On-Vehicle and Aerial Texture Analysis for Vision-Based Desert Road Following. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

Reich, A. und Wuensche, H.-J. (2021). Monocular 3D Multi-Object Tracking with an EKF Approach for Long-Term Stable Tracks. In *Proceedings of IEEE International Conference on Information Fusion (FUSION)*.

Rieken, J. (2020). *Ein Beitrag zur 3D-Umfeldwahrnehmung für automatisierte Straßenfahrzeuge im urbanen Raum*. Dissertation, Technische Universität Carolo-Wilhelmina zu Braunschweig, Fakultät für Elektrotechnik, Informationstechnik, Physik.

Rieken, J., Matthaei, R., und Maurer, M. (2015). Toward Perception-Driven Urban Environment Modeling for Automated Road Vehicles. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, Seiten 731–738.

Roth-Tabak, Y. und Jain, R. (1989). Building an Environment Model Using Depth Information. *Computer*, 22(6):85–90.

Rudschies, W. (2023). Autonomes Fahren Level 3: Freihändig durch den Stau. Allgemeiner Deutscher Automobil-Club e.V. (ADAC), 5. September 2023. URL: https://www.adac.de/rund-ums-fahrzeug/ausstattung-technik-zubehoer/autonomes-fahren/technik-vernetzung/autonomes-fahren-staupilot-s-klasse. Abgerufen: 3. April 2024.

Rusu, R. B. und Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*.

Ryde, J. und Hu, H. (2010). 3D Mapping with Multi-Resolution Occupied Voxel Lists. *Autonomous Robots*, 28:169–185.

Sack, D. und Burgard, W. (2004). A comparison of methods for line extraction from range data. *IFAC Proceedings Volumes*, 37(8):728–733. IFAC/EURON Symposium on Intelligent Autonomous Vehicles.

Schmid, M. R., Maehlisch, M., Dickmann, J., und Wuensche, H.-J. (2010). Dynamic Level of Detail 3D Occupancy Grids for Automotive Use. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 269–274.

Schneider, S. (2023). *Fusion von Kamera und LiDAR zur modellbasierten autonomen Navigation*. Dissertation, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.

Schneider, S., Himmelsbach, M., Luettel, T., und Wuensche, H.-J. (2010). Fusing Vision and LIDAR - Synchronization, Correction and Occlusion Reasoning. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 388–393.

Schreier, M., Willert, V., und Adamy, J. (2013). From Grid Maps to Parametric Free Space Maps — A Highly Compact, Generic Environment Representation for ADAS. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*, Seiten 938–944.

Seki, A. und Pollefeys, M. (2017). SGM-Nets: Semi-Global Matching with Neural Networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seiten 6640–6649.

Siciliano, B. und Khatib, O., Herausgeber (2008). *Springer Handbook of Robotics*. Springer.

Simon, D. (2006). *Optimal State Estimation*. John Wiley & Sons.

Song, D., Lee, H. N., Yi, J., und Levandowski, A. (2006). Vision-based Motion Planning for an Autonomous Motorcycle on Ill-Structured Road. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Seiten 3279–3286.

Steger, C., Ulrich, M., und Wiedemann, C. (2018). *Machine Vision Algorithms and Applications*. Wiley-VCH.

Tan, C., Hong, T., Chang, T., und Shneier, M. (2006). Color Model-Based Real-Time Learning for Road Following. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, Seiten 939–944.

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 of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 6090–6095.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., und Mahoney, P. (2006). Stanley: The Robot That Won The DARPA Grand Challenge. *Journal of Field Robotics*, 23(9):661–692.

Triebel, R., Pfaff, P., und Burgard, W. (2006). Multi-Level Surface Maps for Outdoor Terrain Mapping and Loop Closing. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Seiten 2276–2282.

Unterholzner, A. (2016). *Sensor Orientation Selection and Adaptive Control of an Actuated Sensor Platform for Autonomous Vehicles*. Dissertation, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M., Dolan, J., Duggins, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T., Kolski, S., Kelly, A., Likhachev, M., McNaughton, M., Miller, N., und Ferguson, D. (2008). Autonomous Driving in Urban Environments: Boss and the Urban Challenge. *Journal of Field Robotics*, 25(8):425–466.

Urmson, C., Anhalt, J., Bartz, D., Clark, M., Galatali, T., Gutierrez, A., Harbaugh, S., Johnston, J., Kato, H., Koon, P., Messner, W., Miller, N., Mosher, A., Peterson, K., Ragusa, C., Ray, D., Smith, B., Snider, J., Spiker, S., Struble, J., Ziglar, J., und Whittaker, W. (2007). A Robust Approach to High-Speed Navigation for Unrehearsed Desert Terrain. In Buehler, M., Iagnemma, K., und Singh, S., Herausgeber, *The 2005 DARPA Grand Challenge: The Great Robot Race*, Volume 36 of *Springer Tracts in Advanced Robotics*, Seiten 45–102. Springer.

Vasudevan, S., Ramos, F., Nettleton, E., Durrant-Whyte, H., und Blair, A. (2009). Gaussian Process Modeling of Large Scale Terrain. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 1047–1053.

Velodyne LiDAR, Inc. (2011). *HDL-64E S2 – User's Manual and Programming Guide*. 63HDL64E S2 Rev. D.

Velodyne LiDAR, Inc. (2018). *VLS-128 User Manual*. 63-9483 Rev. A (Draft).

Viola, P. und Wells, W. (1995). Alignment by Maximization of Mutual Information. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, Seiten 16–23.

von Holt, V. (2004). *Integrale Multisensorielle Fahrumgebungserfassung nach dem 4D-Ansatz*. Dissertation, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechik, Institut für Systemdynamik und Flugmechanik.

von Hundelshausen, F., Himmelsbach, M., Hecker, F., Mueller, A., und Wuensche, H.-J. (2008). Driving with Tentacles: Integral Structures of Sensing and Motion. *Journal of Field Robotics*, 25(9):640–673.

Waldron, K. und Schmiedeler, J. (2008). Kinematics. In Siciliano und Khatib [2008], Seiten 585–610.

Wang, T.-H., Hu, H.-N., Lin, C. H., Tsai, Y.-H., Chiu, W.-C., und Sun, M. (2019). 3D LiDAR and Stereo Fusion using Stereo Matching Network with Conditional Cost Volume Normalization. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Seiten 5895–5902.

Wang, Y. und Park, J. (2021). Transitional Asymmetric Non-local Neural Networks for Real-World Dirt Road Segmentation. In *International Conference on Pattern Recognition (ICPR)*, Seiten 6949–6956.

Waxman, A., LeMoigne, J., Davis, L., Srinivasan, B., Kushner, T., Liang, E., und Siddalingaiah, T. (1987). A Visual Navigation System for Autonomous Land Vehicles. *IEEE Transactions on Robotics and Automation*, 3(2):124–141.

Weiherer, T., Bouzouraa, S., und Hofmann, U. (2013). An interval based representation of occupancy information for driver assistance systems. In *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC)*, Seiten 21–27.

Winner, H., Hakuli, S., Lotz, F., und Singer, C., Herausgeber (2015). *Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*. ATZ/MTZ-Fachbuch. Springer, 3. Edition.

Woebbecke, D., Meyer, G., von Bargen, K., und Mortensen, D. (1995). Color Indices for Weed Identification Under Various Soil, Residue, and Lighting Conditions. *Transactions of the American Society of Agricultural Engineers*, 38(1):259–269.

Wurm, K., Hornung, A., Bennewitz, M., Stachniss, C., und Burgard, W. (2010). OctoMap: A Probabilistic, Flexible, and Compact 3D Map Representation for Robotic Systems. In *Proceedings of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*.

Yang, Y., Wang, Z., Li, X., und Fu, M. (2016). Real-time Terrain Estimation Based on Multi-scale Radial Basis Function for Unmanned Ground Vehicle. In *Proceedings of IEEE International Conference on Information and Automation (ICIA)*, Seiten 659–664.

Ye, C. und Borenstein, J. (2003). A new terrain mapping method for mobile robots obstacle negotiation. In Gerhart, G. R., Shoemaker, C. M., und Gage, D. W., Herausgeber, *Unmanned Ground Vehicle Technology V*, Volume 5083, Seiten 52–62. International Society for Optics and Photonics, SPIE.

Yin, S. und Zhu, L. (2022). Country Road Boundaries Detection for Autonomous Driving Based on LiDAR Point Cloud. In *Proceedings of IEEE International Conference on Electronics Technology (ICET)*, Seiten 1117–1120.

Zabih, R. und Woodfill, J. (1994). Non-parametric Local Transforms for Computing Visual Correspondence. In *Proceedings of European Conference on Computer Vision (ECCV)*, Seiten 151–158.

Zhang, J., Ramanagopal, M. S., Vasudevan, R., und Johnson-Roberson, M. (2020). LiStereo: Generate Dense Depth Maps from LIDAR and Stereo Imagery. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, Seiten 7829–7836.

Zhao, Z., Fu, H., Ren, R., und Sun, Z. (2022). Real-Time Intersection Detection Based on Satellite Image and 3D LIDAR Point Cloud. In *Proceedings of International Conference on Autonomous Unmanned Systems (ICAUS)*, Seiten 2868–2878.