UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

FAKULTÄT FÜR
ELEKTROTECHNIK,
INFORMATIK UND
MATHEMATIK

Kreuzmodale Supervision für automotive Radar- und Kamerasensoren

Von der Fakultät für Elektrotechnik, Informatik und Mathematik
der Universität Paderborn

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von

M.Sc. Christopher Grimm

Erster Gutachter: Prof. Dr.-Ing. Reinhold Häb-Umbach
Zweiter Gutachter: Prof. Dr.-Ing. Markus Gardill

Tag der mündlichen Prüfung: 12.09.2024

Paderborn 2024

Diss. EIM-E/384

Danksagung

Mit der vorliegenden Dissertation geht ein wichtiger Abschnitt meines Lebens zu Ende, der ohne die Unterstützung und das Vertrauen vieler Menschen nicht möglich gewesen wäre. An dieser Stelle möchte ich all jenen danken, die mich während der letzten Jahre begleitet und gefördert haben.

Zunächst möchte ich meinem Doktorvater, Herrn Prof. Dr.-Ing. Reinhold Häb-Umbach von der Universität Paderborn, meinen aufrichtigen Dank aussprechen. Für die Übernahme des Referates dieser Arbeit, seine fachliche Expertise, seine unermüdliche Geduld und die stets offenen und wertvollen Diskussionen bin ich ihm sehr verbunden. Im komplizierten Themengebiet der Nachrichtentechnik hat er ein wundervoll inspirierendes akademisches Fachgebiet erbaut. Insbesondere die Art und Weise der durchgeführten Vortragsreihen werden mir als wertvoller Teil einer akademischen Ausbildung und exzellente Vorbereitung für den wissenschaftlichen Austausch in Erinnerung bleiben.

Herrn Prof. Dr.-Ing. Markus Gardill von der Universität Cottbus danke ich für die Übernahme des Korreferates und die Hinweise zur Verbesserung dieser Arbeit.

Meinen Vorgesetzten und (ehemaligen) Kollegen der Firma HELLA GmbH & Co. KGaA, Herrn Dr.-Ing. Ernst Warsitz, Dr.-Ing. Ridha Farhoud, Dr.-Ing. Tai Fei und Dr. rer. nat. Andreas von Rhein danke ich zutiefst für die Ermöglichung und Unterstützung dieser Arbeit. Sie haben durch konstruktive Anmerkungen und inspirierende Gespräche die Qualität dieser Arbeit entscheidend geprägt. Eure Hilfe ist unbezahlbar und essentieller Nährstoff meines fachlichen und persönlichen Heranreifens.

Darüber hinaus möchte ich mich bei meiner Familie und meinen Freunden bedanken. Euer Zuspruch, eure Geduld und euer Verständnis haben mir stets den Rücken gestärkt, insbesondere in den schwierigen Phasen des Lebens. Meinen Eltern danke ich von Herzen für ihre bedingungslose Unterstützung und dafür, dass sie mir die Möglichkeit gegeben haben, meine Träume zu verwirklichen.

Schließlich gebührt mein größter Dank meiner Frau Christiane, die mich in den letzten Jahren mit Liebe, Geduld und Zuspruch begleitet hat. Du warst immer eine Quelle der Motivation und des Trostes – ohne dich wäre dieses Kapitel meines Lebens nur halb so schön gewesen. Unseren gemeinsamen Kindern Marlene und Karl danke ich für all die wundervollen Momente und freue mich nach Abschluss dieser Arbeit sehr, nun mehr Zeit mit euch verbringen zu können.

Diese Arbeit ist nicht nur das Ergebnis meiner eigenen Anstrengungen, sondern auch das Produkt der Unterstützung all derjenigen, die mir auf diesem Weg zur Seite gestanden haben. Dafür bin ich euch zutiefst dankbar.

Abstract

Automatic signal processing of sensor signals is an integral part of automotive radar sensors. Only the signal processing extracts the relevant information from the sampled sensor signals and enables the driver-assistance system to issue warnings or make automated changes to the control-loop.

Due to parasitic effects, signal processing is suboptimal and new algorithms are constantly being developed to improve the quality of processing. One possible development path is the class of machine learning algorithms. The associated algorithms recognize patterns based on previously presented data sets. An important subclass here are the supervised learning procedures, which require labeled datasets. The datasets consist of input data and corresponding target values. During a training procedure, the algorithms are then automatically optimized and attempt to estimate the target values based on the input data. This work deals with the automatic association of image content from camera and radar sensors. With the help of this association, any data from the camera can be used as a target value for training machine learning procedures on radar data and vice versa. The benefit of this association is demonstrated using the examples of trained direction-of-arrival estimators, target detectors, semantic segmentation of radar spectra as well as radar-power prediction from camera-images.

Zusammenfassung

Die automatische Signalverarbeitung von Sensorsignalen ist ein integraler Bestandteil von automotiven Radarsensoren. Erst die Signalverarbeitung extrahiert die relevanten Informationen aus den abgetasteten Sensorsignalen und ermöglicht dem Fahrerassistenzsystem, gezielt Warnhinweise auszugeben oder automatisiert Veränderungen von Stellgrößen durchzuführen.

Die Güte der zur Signalverarbeitung eingesetzten Algorithmen wird mitunter durch Faktoren wie Laufzeitbedingungen, Robustheit gegenüber Rauschen bzw. parasitären Signalanteilen sowie Robustheit der Modellannahmen der genutzten Algorithmen definiert. Um diese Güte zu verbessern, werden fortlaufend neue Algorithmen entwickelt. Ein möglicher Entwicklungspfad ergibt sich dabei durch die Klasse der Algorithmen zum maschinellen Lernen. Die damit verbundenen Algorithmen erkennen Muster anhand zuvor präsentierter Datensätze. Eine wichtige Unterklasse sind dabei die überwachten Lernverfahren, bei welchen die Datensätze aus den Eingangsdaten der Algorithmen und gelabelten Zielwerten zu den Eingangsdaten bestehen. Während einer Trainingsprozedur werden die Algorithmen dann automatisch optimiert und versuchen die Zielwerte anhand der Eingangsdaten zu schätzen. Diese Arbeit befasst sich mit der automatischen Assoziation von Bildinhalten aus Kamera und Radarsensoren. Mit Hilfe dieser Assoziation lassen sich etwaige Daten der Kamera als Zielwert für das Training von maschinellen Lernverfahren auf Radardaten und umgekehrt verwenden. Der Benefit dieser Assoziation wird demonstriert an den Beispielen von trainierten Winkelschätzern, Zieldetektoren, semantischer Segmentierung von Radar-Spektren sowie einer Radar-Leistungsschätzung aus Kamerabildern.

Inhaltsverzeichnis

1. Einleitung

Automotive-Radarsensoren sind ein integraler Bestandteil für heutige Fahrerassistenzsysteme. Die Sensoren emittieren elektromagnetische (EM)-Wellen in die Umgebung und können aus der Reflexion der Wellen Rückschlüsse über die Umgebung ableiten und dem Fahrerassistenzsystem auf unterschiedlichen Signalabstraktionsebenen präsentieren. Es werden vier Signalabstraktionsebenen unterschieden:

1. **ADU-Ebene**: Die am Radarsensor ankommende Welle wird nach analoger Vorverarbeitung durch einen Analog-Digital-Umsetzer (ADU) abgetastet und somit für eine maschinelle Verarbeitung in digitale Werte gewandelt.

2. **Frequenzspekten-Ebene**: Als erste digitale Verarbeitung werden die ADU-Daten durch Fourier-Transformation in den Frequenzraum transformiert und Eigenschaften der Signale für weitere Verarbeitungsschritte damit zugänglicher gemacht.

3. **Detektions-Ebene**: In den Frequenzspektren werden signifikante Signalanteile detektiert und physikalisch sinnvolle Signaleigenschaften wie Entfernung, Geschwindigkeit und Einfallswinkel geschätzt.

4. **Objekt-Ebene**: Die Detektionen werden zu Objekten gebündelt (engl.: „cluster“) und repräsentieren zusammenhängende Körper wie Fahrzeuge oder Fußgänger.

5. **Warnalgorithmus-Ebene**: Die Objekte werden verfolgt (engl: „tracked“) und deren Fahrtrajektorie geschätzt. Bei gefährlicher Überschneidung mit der Trajektorie des Ego-Fahrzeuges werden entsprechende Warnmeldungen an Fahrerassistenzsystem und Fahrer übermittelt.

Der Fokus dieser Arbeit liegt auf der Verarbeitung von Signalen auf der Detektions-Ebene. Dabei soll insbesondere die Verarbeitung mit modernen Signalverarbeitungsverfahren untersucht werden.

In den letzten zehn Jahren haben sich auf neuronalen Netzwerken (NN) basierende Algorithmen im Vergleich zu klassischen Algorithmen des maschinellen Lernens in verschiedenen Anwendungen als überlegen erwiesen [HDH+19, CMMS11, ZTM+17]. Einen großen Beitrag dafür hat die fortlaufende Forschung und Entwicklung zu Methoden um das maschinelle Lernen geliefert. Nicht zuletzt liegt dies aber auch an der steigenden Anzahl annotierter Trainingsdaten. Mit diesen Trainingsdaten können die Netzwerkparameter des NNs statistisch optimiert werden. Während des Trainings werden dem NN dazu Eingangsdaten präsentiert, welche das NN in Prädiktionen umwandelt. Diese Prädiktionen werden gegen vorgegebene Zielwerte verglichen, die Abweichung ermittelt und dann entsprechend die Netzwerkparameter optimiert [Bis95]. Häufig wird dabei das Überwachungsprinzip bzw. überwachte Lernen angewendet, bei welchem die Zielwerte manuell oder automatisch festgelegt werden. Man spricht dabei von der „Annotation“, dem „Labeling“ oder auch der „Etikettierung“.

1.1. Auflistung automotiver Datensätze und verwandter Arbeiten

Typischerweise stellen Kamera und Lidar die Umgebung in einer für den Menschen vertrauten Art und Weise dar. Entsprechend einfach sind die Daten durch den Menschen zu verstehen und zu annotieren, so dass eine frühe Entwicklung und Veröffentlichung von automotiven Datensätzen mit dem Fokus auf Kamera- und Lidardaten zu beobachten war und den Einstieg in die Entwicklung und den Vergleich von Signalverarbeitungen vereinfachte.

2012 wurde durch [GLSU12] der bekannte *KITTI*-Datensatz veröffentlicht. Wurden zuvor Algorithmen meist unter Laborbedingungen validiert, konnte dies nun mit dem Datensatz, bestehend aus Sensordaten von Global-Positioning-System (GPS), Lidar und Kamera und den dazugehörigen Annotationen für Objekterkennung in typischen Fahrszenarien aus Karlsruhe durchgeführt werden. Einhergehend wurden öffentliche Benchmarks für Stereosehen, optischen-Fluss, visuelle Odometrie und 3D-Objekterkennung gestartet. Seitdem wurden weitere Benchmarks zur Tiefenvervollständigung, semantischen Segmentierung und Szenenfluss-Schätzung aufgenommen [GLU12, MG15a]. Obschon dieser Datensatz ein bedeutender Meilenstein für die Validierung von kamera- und lidarbasierter Signalverarbeitung ist, hat er aufgrund fehlender Radardaten keine direkte Relevanz für radarbasierte Signalverarbeitung.

In 2019 wurde deshalb durch [CBL+19] der *nuScenes*-Multi-Sensor-Datensatz veröffentlicht. Dieser beinhaltet neben Daten aus Kamera und Lidar auch Punktwolken aus Radarsensoren. Der Datensatz beinhaltet Daten aus etwa 242 km Fahrtstrecken in Boston, USA und Singapur. Die Annotationen in Form von 3D-Bounding-Boxen (BBox) wurden primär von menschlichen Labelern bereitgestellt. Als BBox werden rechteckige Hüllen bezeichnet, welche in Kamerabildern oder Punktwolken eingezeichnet werden und die einzelnen Objekte umschließen. Da für die Radarsensoren keine Frequenzdaten vorliegen, aus denen mittels Signalverarbeitung die Detektionen gebildet werden, hat dieser Datensatz keine wesentliche Bedeutung für die vorliegende Arbeit und die Entwicklung entsprechender Signalverarbeitung. Als neuerer Datensatz mit Kamera- und Lidardaten sei der *Waymo*-Datensatz [SKD+20, ECC+, CGQ+23] zu nennen. Da keine Radardaten enthalten sind, hat auch dieser Datensatz keine Relevanz für die vorliegende Arbeit.

Ähnliche Restriktionen ergeben sind für die Datensätze aus [COR+16] und [NOBK17].

Die Sensordaten aus automotiven Radarsensoren, insbesondere niedriger Signalebene, sind für den Menschen nicht/kaum zu interpretieren und konnten daher nicht, wie z.B. Kamerabilder, manuell annotiert werden. In den letzten Jahren haben sich jedoch auch hier Verfahren und Datensätze entwickelt, welche das Training und die Validierung radarbasierter Signalverarbeitungen ermöglichen. Eine unvollständige Zusammenfassung publizierter Verfahren zur automatischen Annotation von Radardaten und dazugehörigen Datensätzen ist nachfolgend gegeben.

2018 2018 wurden durch [HHBD18] Punktwolken automatisch annotiert. Dabei wurde keine Referenzsensorik verwendet und die Punktwolken durch Vorwärts- und Rückwärtsaggregation/Registrierung der Punktwolken verdichtet. Nach der Aggregation wurde einmalig ein NN zur Objektdetektion in den Punktwolken trainiert, welches anschließend zur Annotation des gesamten Datensatzes verwendet wurde. Da der Fokus

der vorliegenden Arbeit die Generierung von Punktwolken selbst ist, ist dieser Ansatz nicht ohne Weiteres kompatibel.

2019 2019 wurde durch [Lim19] neben der radarbasierten Signalverarbeitung eine parallele Signalverarbeitung von Kameradaten mittels NN eingeführt. Die Signale wurden fusioniert und so eine verbesserte Detektion von Fahrzeugen erreicht. Fahrzeuge wurden semi-automatisch anhand von Lidar-Punktwolken annotiert.

In [MFA+19] wurde ein Fahrzeug mit Lidar, Kamera und Radar ausgestattet. In den gefahrenen Autobahnszenarien wurden anhand der Lidar-Punktwolken semi-automatisch Fahrzeuge detektiert und verfolgt. Diese wurden als Annotation für das Training einer Radarsignalverarbeitung mittels NN genutzt. Die Verknüpfung der Annotationen und der Radardaten erfolgt durch Transformation der Radardaten, gegeben in Polarkoordinaten, in das kartesische Koordinatensystem der Lidar-Punktwolken.

Durch [MK19] wurde ein Datensatz bestehend aus Kamerabildern, Lidar- und Radar-Punktwolken veröffentlicht. Darin sind 3D-BBox für Fahrzeuge und Fußgänger enthalten. Außerdem wird beschrieben, wie durch „Active Learning“ der manuelle Annotationsaufwand verringert werden konnte.

2020 Im Jahr 2020 wurde durch [NKK+20] eine Freiraumerkennung mittels Convolutional-Neural-Network (CNN) auf Radardaten trainiert. Die Annotationen wurden dabei automatisch mittels „Structure from Motion“ aus Kameradaten erzeugt.

Durch [PDKG20] wurde eine Objekterkennung für Fußgänger, Fahrzeuge und Fahrradfahrer mittels NN auf range-azimuth-Doppler (RAD)-Spektren vorgestellt. Als Annotationen wurden in den Bildern der verwendeten Stereo-Kamera 3D-BBox für die Instanzen detektiert.

[DWZL20] haben Fahrzeuge durch eine NN basierte Radarsignalverarbeitung detektiert. Die Annotationen für das Training wurden automatisch durch eine Signalverarbeitung von Lidar-Punktwolken generiert.

[MWRH20] haben den *Zendar*-Datensatz veröffentlicht. Dabei wurde eine Detektion bewegter Fahrzeuge mittels CNN Signalverarbeitung von RD-maps erreicht. Zur Annotation der Daten wurden mittels klassischer Verarbeitung Objekte im RAD-Spektrum detektiert und in ein Synthetic-Aperture-Radar (SAR)-Bild projiziert. Durch die Projektion wurden bewegte von stationären Objekten diskriminiert.

Der *CARRADA*-Datensatz wurde von [ONR+21] veröffentlicht. In dem Datensatz wurden Objekte der Klassen Fußgänger, Fahrzeug oder Fahrrad in den Kamerabildern detektiert und verfolgt. Gleichzeitig werden im Radar RAD-Spektrum Ziele mittels eines klasisschen Constant-False-Alarm-Rate (CFAR)-Detektors erkannt und anschließend zu Instanzen geclustert. Durch Assoziation der Instanzen aus Radar und Kameradaten werden die Annotationen aus den Kamerabildern in das Radar Spektrum projiziert. Die Annotationen für den Radar sind entsprechend stark durch die Detektion mittels CFAR bestimmt. In der vorliegenden Arbeit ist die Annotation der Radardaten, basierend ausschließlich auf den Daten der Referenzsensorik, gewünscht, um so nicht durch eine vorhandene Radarsignalverarbeitung beeinflusst zu werden.

[BGM+20] haben den *Oxford Radar RobotCar*-Datensatz vorgestellt. Dieser beinhaltet neben Kamerabildern und Lidar-Punktwolken auch range-Azimuth (RA)-Spektren eines Radarsensors. In [KDMGN20] wurde dieser Datensatz verwendet, um aus Kamera- und

Lidardaten semantische Annotationen für Radar-Punktwolken zu schaffen, gegen welche ein CNN zur Radar-Signalverarbeitung trainiert wurde.

Durch [KPC+20] wurde der *MulRan*-Datensatz veröffentlicht. Dieser beinhaltet neben Lidar-Punktwolken auch RA-Spektren eines verwendeten Radarsensors. Ziel dieses Datensatzes ist die Positionserkennung mit Radar und Lidar. Es werden keine Annotationen für weitere Radaranwendungen geliefert.

[BRZB20] veröffentlichten den *Pointillism*-Datensatz. Darin sind Punktwolken mehrerer Radar- und Lidarsensoren sowie Kamerabilder enthalten. Der Fokus dieses Datensatzes liegt in der Fusion von Punktwolken mehrerer Radarsensoren zur Objektdetektion.

[GMJ+20] veröffentlichten ein Verfahren und einen Datensatz zur Schätzung hochauflösender Bilder aus Radar-Heatmaps. Ein Generative-Adversarial-Network (GAN) prozessiert dabei die Heatmaps aus Radarsensoren und imitiert dadurch hochaufgelöste Tiefenbilder aus einem Stereo-Kamera-System.

2021 Im *RADDet*-Datensatz [ZNL21] aus dem Jahr 2021 wurden Personen, Fahrräder und Fahrzeuge in Radarpunktwolken und Kamerabildern detektiert, miteinander fusioniert und verfolgt. Die fusionierten Objekte wurden in das RAD-Spektrum projiziert und dienten so als Annotation für das Training einer NN basierten Objekterkennung auf Radardaten.

[WJL+21, WWH+21] stellten den *CRUW*-Datensatz vor. Aufgabe ist das Training einer Objekterkennung von Fußgängern, Fahrradfahrern und Fahrzeugen mittels NN basierter Signalverarbeitung der Radar-Frequenzdaten. Annotationen werden aus Kamerabildern und Radar-Punktwolken generiert.

Durch [SDPM+21] wurde ein weiterer Datensatz mit Kamerabildern, Lidar-Punktwolken und RA-Spektren veröffentlicht. Im Gegensatz zu anderen Datensätzen lag der Fokus dabei auf der Bereitstellung von Daten aus unterschiedlichen Witterungsbedingungen. So beinhaltet der Datensatz Aufnahmen aus sonnigen, verregneten, dunklen, vernebelten und verschneiten Tagen. Im Datensatz wurden Fahrzeuge und Fußgänger als 2D-BBox annotiert. [SHS+21] hat den *RadarScenes*-Datensatz bereitgestellt. Annotationen wurden für Fahrzeuge, Fußgänger und Tiere generiert. Eine Beschreibung des manuellen Annotationsprozesses ist u.a. in [Rad21] dokumentiert. Der Datensatz enthält unkalibrierte Kamerabilder und Radar-Punktwolken.

[KSRD21] veröffentlichte den *Radar Ghost*-Datensatz und ein NN basiertes Verfahren zur automatischen Detektion von sogenannten Geisterreflexionen in Radar-Punktwolken.

Durch [DD21] wurde ein Verfahren vorgestellt, um aus RD-maps des Radars Kamerabilder zu rekonstruieren. Dabei kamen Variational-Autoencoder (VAE) zum Lernen des Merkmalraums zum Einsatz.

[MKT21] detektierten Fahrzeuge in RAD-Spektren mittels Graph-NN.

Durch [WJP21] wurde ein GAN als generatives Modell zur Erzeugung von Radar RA-Spektren vorgestellt. Dabei wurden dem GAN Elevationskarten aus Lidar und die RA-Spektren aus dem Radar präsentiert und das GAN so trainiert, dass Daten aus der jeweils anderen Sensordomäne prädiziert werden konnten. Die gezeigten Beispiele sehen plausibel aus und durch Domänen-Adaption konnten weiterhin synthetische Radardaten aus simulierten Lidar-Elevationskarten erzeugt werden.

2022 Im Jahr 2022 wurde von [ROMP22] der *RADIal*-Datensatz vorgestellt. Es stehen Kamerabilder, Lidar- und Radarpunktwolken sowie Radarfrequenzspektren zur

Verfügung. Der Fokus dieses Datensatzes liegt auf der Objekterkennung und Freiraumerkennung mittels Radarpunktwolken und Radarfrequenzspektren. Entsprechende Annotationen werden bereitgestellt.

Zwischen 2021 und 2022 wurde durch [GXRL21, GLX+22] der *UWCR*-Datensatz beschrieben und veröffentlicht. Dieser beinhaltet Kamerabilder und ADU-Daten eines verwendeten Radarsensors. Annotationen wurden für Fahrradfahrer, Fußgänger und Fahrzeuge in Form von 3D-BBox bereitgestellt.

Durch [PPB+22] wurde der *View of Delft*-Datensatz veröffentlicht. Dieser beinhaltet Bilder aus einem Stereo-Kamera System, Lidar- und Radar-Punktwolken aus urbanen Szenarien. Annotiert wurden 3D-BBox für u.a. Fahrzeuge, Fußgänger, Fahrrad- und Motorradfahrer. [ZMZ+22] veröffentlichten den *TJ4DRadSet*-Datensatz. Neben Kamerabildern sind Lidar- und Radar-Punktwolken vorhanden. Annotiert wurden 3D-BBox von bewegten Objekten. Gefahren wurde in Suzhou, China.

[PKW22] haben den *K-Radar*-Datensatz veröffentlicht. Er beinhaltet Kamerabilder, Lidar-Punktwolken und range-azimuth-elevation-Doppler (RAED)-Spektren. Annotiert wurden 3D-BBox bewegter Objekte. Hervorzuheben ist die auffällig umfassende Dokumentation des Kalibrier- und Annotationsprozesses [Ka22].

Durch [BRB22] wurde ein weiteres Verfahren über die Fusion von Kamerabildern und Radarpunktwolken zur Detektion von Fahrzeugen vorgestellt.

[KWL22] veröffentlichten ein Verfahren zur Erkennung von Verdeckungen durch Reflektoren in Radardaten. Zum Einsatz kam ein NN welches RA-Spektren prozessierte und versuchte, Verdeckungen zu detektieren. Trainiert wurde das NN gegen Lidar-Punktwolken. Durch die unterschiedlichen Wahrnehmungseigenschaften von Lidar und Radar kam es zu Falschdetektionen nach dem Training. Aus diesem Grund wurden Vorverarbeitungsschritte unternommen, um diese unterschiedlichen Wahrnehmungseigenschaften zu berücksichtigen und schlussendlich die Falschdetektionen zu reduzieren.

2023 Im Jahr 2023 wurde durch [MBB+23] der *aiMotive*-Datensatz veröffentlicht. Dieser beinhaltet synchronisierte und kalibrierte Daten aus Lidar (Punktwolke), Kamera (Bild) und Radar (Punktwolke). Es wurden 3D-BBox für Fahrzeuge, Fußgänger, Motor- und Fahrräder, Anhänger und Züge bereitgestellt.

Zwischen 2020 und 2023 wurde durch [DSH+20,DSVH+21,Dim23] der *IMEC tracking*-Datensatz vorgestellt. Dieser beinhaltet Sensordaten aus Radar (Punktwolken und RAD-Spektren), Kamera (Bilder) und Lidar (Punktwolke). Der Datensatz ist über [Gen23] erreichbar. Annotiert wurden Fußgänger und Fahrzeuge in Form von BBox.

Durch [CYH+23] wurde der *MSC-RAD4R*-Datensatz veröffentlicht. Er beinhaltet neben Bildern aus einem Stereo-Kamerasystem auch Punktwolken aus Lidar- und Radarsensoren. Annotiert wurden 3D-BBox bewegter Objekte.

[EHR23] stellten einen Datensatz zur Untersuchung von Radar-Cross-Section (RCS) Abhängigkeiten in typischen Fahrszenarien vor. Dabei wurden Fahrzeuge vor einem verfolgenden Radarsensor bewegt, die Punktwolken des Radarsensors akquiriert und so die Abhängigkeit von RCS gegenüber z.B. Aspektwinkeln bei unterschiedlichen Fahrzeugen gemessen.

In [KKPD23] wurde ein Verfahren vorgestellt, welches automatisch Störziele in Radar-Punktwolken und echte (bewegt und stationär) Ziele klassifiziert. Als Ursache für diese Störziele wurden insbesondere Mehrwegeausbreitungen in der Propagation der ausgesen-

deten EM-Wellen des Radars genannt. Diese können nachgelagerte Signalverarbeitungen stören und sind deshalb zu identifizieren.

Durch [BWY+22, BYW+23] wurde der *Boreas*-Datensatz vorgestellt. Neben Kamerabildern und Lidar-Punktwolken sind RD-Spektren eines Radarsensors vorhanden. Annotationen wurden in Form von 3D-BBox für Fußgänger, Fahrzeuge und Fahrradfahrer zur Verfügung gestellt. [JDFMV23] stellten eine NN basierte Freiraum- und Fahrzeugerkennung anhand von RAD-Spektren vor. Annotationen dafür wurden aus Kamerabildern gewonnen.

Durch [HBS+23] wurden semi-automatisch automotive Radardaten anhand von Luftaufnahmen annotiert. Die Kamerabilder wurden durch Unmanned-Aerial-Vehicle (UAV) bzw. Drohnen akquiriert und anschließend durch panoptische Segmentierung verarbeitet. Durch die genaue Kenntnis der Pose von Drohne und Radarsensor konnte die Segmentierung so in das Koordinatensystem des Radars überführt werden.

Für den interessierten Leser sind darüber hinaus in [ZLZ+22, SM23, ZHO23] umfangreiche Übersichten zum Stand der Wissenschaft dokumentiert.

Leider wurden viele der genannten Datensätze erst veröffentlicht, als die praktische Arbeit zu diesem Werk bereits abgeschlossen war. Somit wurde ein eigener, nicht öffentlicher Datensatz für das Training und den Test von Algorithmen entwickelt. Die dabei gemachten Beobachtungen und Methoden sind in diesem Werk niedergeschrieben.

1.2. Wissenschaftlicher Beitrag dieser Arbeit

Bei den im vorigen Abschnitt genannten Arbeiten wurden meist Radardaten ausschließlich in Form von Punktwolken bereitgestellt. Diese werden, wie im nachfolgenden Kapitel näher beschrieben wird, durch die Detektion signifikanter Reflektoren im empfangenen Reflexionssignal extrahiert und somit ist bereits ein Großteil der Signalverarbeitung verrichtet. Da die Detektionen häufig mittels CFAR Algorithmen gebildet werden, wird häufig von Prä- oder Post-CFAR Signalverarbeitungsbene gesprochen. In der zu Beginn dieses Kapitels genannten Einteilung der Signalverarbeitungsebenen entspricht dies Level 1-2 und 3-5. Für die Entwicklung von NN basierter Signalverarbeitung von Prä-CFAR-Algorithmen sind entsprechend auch nur Datensätze mit Eingangsdaten vor dieser Detektionsschicht nützlich.

Ein weiteres Problem der meisten Datensätze ist, dass die Annotationen nur Zielwerte für Objekte der Klasse Fußgänger und Fahrzeuge bereitstellen. Gerade in urbanen Fahrszenarien kann ein wesentlicher Anteil der Reflexionen jedoch von stationären Objekten wie Fahrbahnen und Randbebauungen ausgehen. Da automotive Radarsensoren auch diese stationären Ziele erkennen müssen, sind entsprechende Annotationen notwendig.

Mit dieser Arbeit wird gezeigt, wie sich Annotationen für automotive Prä-CFAR Radardaten unabhängig von ihrer Klasse automatisch erzeugen lassen. Dazu wird ein Verfahren vorgestellt, mit welchem sich Sensordaten von Radar- und Kamerasensoren automatisch miteinander assoziieren lassen. Für jedes Kamerapixel werden die entsprechenden Bins/Gates/Pixel im Prä-CFAR Frequenzspektrum des Radars ermittelt und der Szeneninhalt somit zueinander in Verbindung gebracht. Durch diese Assoziation lassen sich hochwertige Annotationen aus Kameradaten für das Training von überwachten Lernverfahren auf Radardaten anwenden. Die Assoziation zwischen Kamera und

Radardaten ist nicht trivial, da beide Sensoren auf unterschiedlichen Messprinzipien arbeiten und insbesondere auch einen anderen Messraum besitzen. So löst der Radar beispielsweise die Geschwindigkeit von Reflexionen auf, was der Kamera vorenthalten bleibt. In dieser Arbeit wird der Messraum der Kamera durch automatische Schätzung der Geschwindigkeit mit einem neuen Verfahren zur Szenenfluss-Schätzung erweitert. Dabei wird für den Inhalt jedes Kamerapixels die Geschwindigkeit im Raum geschätzt und somit die Doppler-Messfähigkeit des Radars nachgebildet.

Die Hauptmotivation dieser Arbeit liegt auf der Ermöglichung von überwachtem Training einer Signalverarbeitung von Prä-CFAR Radardaten durch NN. Der vorgeschlagene Ansatz ist in Abbildung 1.1 dargestellt. Er kann als generischer Ansatz für die „kreuzmodale“ Überwachung von NN basierter Signalverarbeitung von Frequency-Modulated-Continuous-Wave (FMCW) Radarsignalen betrachtet werden. Für die Projektion von Radardaten auf Kamerabilder wird eine Warping Schicht vorgestellt, welche differenzierbar ist und so in den Trainingsprozess mittels Fehlerrückführung (engl.: „error backpropagation“) eingebaut werden kann.

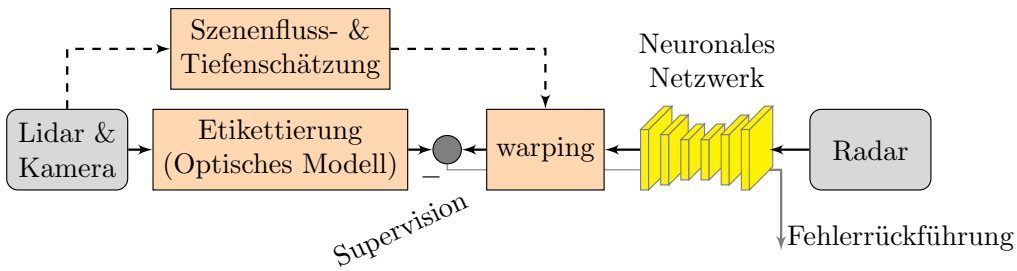


Abbildung 1.1.: **Übersicht über die Trainingspipeline zum überwachten Lernen eines Radar NN:** Das NN auf der rechten Seite wird durch Radardaten gespeist und führe eine Inferenz aus. Diese Inferenz wird durch die Warping Schicht auf das Kamerabild projiziert und mit den Zielwerten aus dem optischen Modell verglichen. Etwaige Abweichungen werden durch Fehlerrückführung durch die Schichten propagiert und entsprechend die NN Parameter optimiert. Die Warpingschicht wird durch Szenenfluss und Tiefenschätzung im Kamerabild unterstützt. Nach [EB6].

Diese Trainings-Pipeline wird für die folgenden radarbasierten Anwendungen demonstriert

- Direction-of-Arrival (DoA) Schätzungen (Abschnitt 7.2),
- Zieldetektionen (Abschnitt 7.3) und
- semantischen Segmentierungen (Abschnitt 7.4).

In dieser Pipeline ergibt sich eine sogenannte „Teacher/Student“ Rollenverteilung [HVD15, HLL+23]. Das optische Modell liefert die Zielwerte und nimmt damit die Rolle des „Lehrers“ ein. Das NN im Radar-Zweig wird gegen diese Zielwerte trainiert und ist somit in der Rolle des „Studenten“. Es wird auch gezeigt, dass sich diese Rollenverteilung umkehren lässt, womit Zielwerte von Radar und Eingangsdaten von Kamera geliefert

werden. Demonstriert wird dies mit der Schätzung der Radar-Empfangsleistung aus Kamerabildern (Abschnitt 7.5).

1.3. Gliederung dieser Arbeit

Der Rest dieser Arbeit ist wie folgt gegliedert. Kapitel 2 gibt eine kurze Einführung in die notwendigen Grundlagen zur klassischen Signalverarbeitung für automotive Radarsysteme sowie eine Einführung in die Optimierung von NNen. In Kapitel 3 wird das gewählte Referenzsensorsystem vorgestellt und die notwendigen Kalibrierungen diskutiert. Kapitel 4 befasst sich mit der Vorverarbeitung der Signale aus dem Referenzsensorsystem. In Kapitel 5 wird ein bestehendes Szenenflussverfahren erweitert und damit eine verbesserte Geschwindigkeitsschätzung der Umgebung erreicht. In Kapitel 6 wird die Projektion von Radarfrequenzdaten in das Kamerabild vorgestellt und dabei unter anderem die Prozessierung aus den vorigen Kapiteln genutzt. Damit werden in Kapitel 7 einige Anwendungen der NN Signalverarbeitung trainiert und der Wert der in dieser Arbeit vorgestellten Methoden bemessen. In Kapitel 8 wird die Arbeit zusammengefasst und ein Ausblick gegeben. Abkürzungsverzeichnis, Übersicht der Symbole und Notationen, Abbildungs-, Tabellen- und Literaturverzeichnis sind dahinter zu finden.

2. Grundlagen zur Signalverarbeitung

In diesem Kapitel werden zunächst die zum Verständnis dieser Arbeit notwendigen Grundlagen zur klassischen Signalverarbeitung von Radarsignalen für automotive Anwendungen beschrieben. Anschließend werden die notwendigen Grundlagen für die Optimierung von NNen beschrieben.

2.1. Radarsignalverarbeitung für automotive Anwendungen

Radartechnik wurde erstmals vor etwa 115 Jahren durch Christian Hülsmeyer entwickelt [Hü04a, Hü04b]. Das entwickelte, „Telemobiloskop“ genannte Messsystem wurde für die Fernortung von Schiffen bei schlechter Sicht eingesetzt. Eine Abbildung aus den Patentschriften ist in Abbildung 2.1 zu sehen.

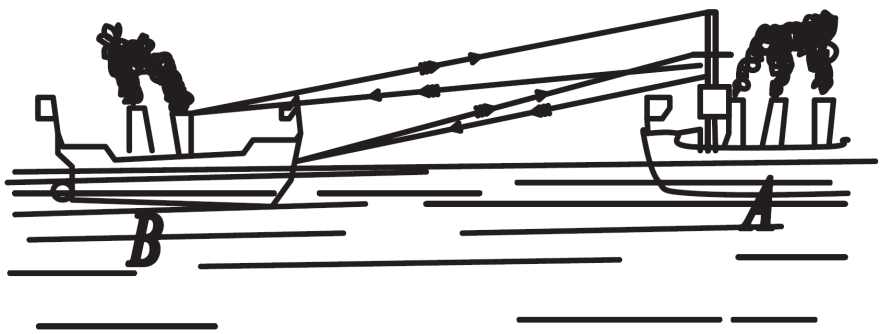


Abbildung 2.1.: **Patentzeichung:** zur Schifffernortung aus [Hü04a].

Schon damals erkannte Hülsemeyer die besondere Bedeutung der theoretischen Abhandlung Maxwells von 1873 über die elektromagnetischen Wellen, welche 1886 durch Hertz experimentell nachgewiesen werden konnten [Hun91]. Es dauerte noch einige Jahre, bis sich die Technik Hülsemeyers in der Luft- und Seeraumüberwachung etablierte. Einem wesentlich breiteren Nutzerpublikum erreichte die Radartechnik durch den Einzug in die Automibiltechnik um 1979 [Mei98].

Um aus den Sensorsignalen Warnhinweise für Fahrer zu präsentieren oder automatische Regeleingriffe durch das Fahrerassistenzsystem durchzuführen, ist eine automatische Auswertung der Radarsignale notwendig. Einige der etablierten Verfahren zur Signalverarbeitung werden in diesem Kapitel vorgestellt. Ziel dieses Kapitels ist es, einen Einblick in elementare Aspekte der Radarsignalverarbeitung zu geben, die zum Verständnis dieser Arbeit notwendig sind. Begonnen wird dabei bei einem Modell zur Beschreibung der Interaktion der vom Radar emittierten Signale an Reflektoren. Darauf folgend wird eine kurze Definition der relativen Größen erfolgen, ehe in einem weiteren Abschnitt auf klassische Signalverarbeitung für automotive Radarsysteme eingegangen wird. Mit Hilfe

dieser Signalverarbeitung lassen sich die von den Reflektoren zurückgeworfenen Signale verarbeiten und wichtige Zustandsgrößen schätzen.

Kein Aspekt dieser Arbeit werden z.B. Informationen zur Antennentechnik sein.

2.1.1. Reflexionspfad

Ein typisches automotives Radargerät sendet EM-Wellen in einem Frequenzbereich um 24 GHz oder 77 GHz aus. Diese Wellen sind größtenteils durch Phasenlage, Amplitude, Polarisation und Wellenlänge spezifiziert. Durch Interaktion, z.B. durch Reflexion, mit der Umgebung, werden die Welleneigenschaften verändert. Eine Beschreibung für diese Welleninteraktion liefern beispielsweise die Fresnelschen-Formeln [Roy08], welche die Änderung der EM-Welle an Grenzflächen beschreibt. Weisen diese Grenzflächen eine Änderung des Brechungsindex auf, so wird die Welle teilweise wieder in das ursprüngliche Übertragungsmedium reflektiert. Ein anderer Teil der ursprünglichen Welle propagiert in das neue Medium. Die sich daraus ergebenden Wellen werden dann als Reflexion und Transmission beschrieben. In Abhängigkeit von dem Einfallswinkel auf der Grenzfläche, der Polarisation der Welle, der Wellenlänge und dem Brechungsindex der Medien werden die Amplitude und die Polarisation der EM-Welle skaliert. Ein Beispiel der Fresnelschen Formel sei gegeben als die relative Signalamplitude der Reflexion und Transmission bei senkrechter $\left((\dots)_s\right)$ und paralleler $\left((\dots)_p\right)$ Polarisation[1] in der Ebene aufgespannt von Einfallsvektor und Ausfallsvektor

$$\left(\frac{E_{0t}}{E_{0e}}\right)_s = \frac{2N_1 \cos\alpha_e}{N_1 \cos\alpha_e + \frac{\mu_{r1}}{\mu_{r2}} N_2 \cos\alpha_a} \tag{2.1}$$

$$\left(\frac{E_{0r}}{E_{0e}}\right)_s = \frac{N_1 \cos\alpha_e - \frac{\mu_{r1}}{\mu_{r2}} N_2 \cos\alpha_a}{N_1 \cos\alpha_e + \frac{\mu_{r1}}{\mu_{r2}} N_2 \cos\alpha_a} \tag{2.2}$$

$$\left(\frac{E_{0t}}{E_{0e}}\right)_p = \frac{2N_1 \cos\alpha_e}{N_1 \cos\alpha_a + \frac{\mu_{r1}}{\mu_{r2}} N_2 \cos\alpha_e} \tag{2.3}$$

$$\left(\frac{E_{0r}}{E_{0e}}\right)_p = \frac{-N_1 \cos\alpha_a + \frac{\mu_{r1}}{\mu_{r2}} N_2 \cos\alpha_e}{N_1 \cos\alpha_a + \frac{\mu_{r1}}{\mu_{r2}} N_2 \cos\alpha_e}. \tag{2.4}$$

Hier entsprechen E_{0t}, E_{0r} und E_{0e} der Wellenamplitude für Transmission, Reflexion und der einfallenden Welle. Weiterhin beschreiben N_1 und N_2 die Brechungsindices der Medien, α_e und α_a sind Winkel zwischen Ein-/Ausfallsvektor und Oberflächennormalen und μ_{r1}, μ_{r2} magnetische Permeabilität der Medien. Zur Veranschaulichung der Welleninteraktion an Grenzflächen ist in Abbildung 2.2 eine graphische Darstellung gegeben. Hierin weisen obere und untere Halbebene unterschiedliche Brechungsindices auf. Die Strahlen sind rot, die senkrechten und parallelen Polarisationen blau und grün dargestellt. Der von oben links kommende Strahl wird an der Grenzfläche in beide Materialien reflektiert. Ein- und Ausfallswinkel für E_{0e} und E_{0r} sind entsprechend des Snelliusschen Brechungsgesetzes identisch eingezeichnet [BDE+09].

Weitere Informationen zur Thematik können in grundlegender Literatur zur Elektrodynamik, Photonik oder physikalischen Optik, beispielsweise [BDE+09], gefunden werden.

[1] Man spricht hierbei von linearer Polarisation. Beliebige Polarisationsrichtungen können aus diesen Polarisationsrichtungen kombiniert werden.

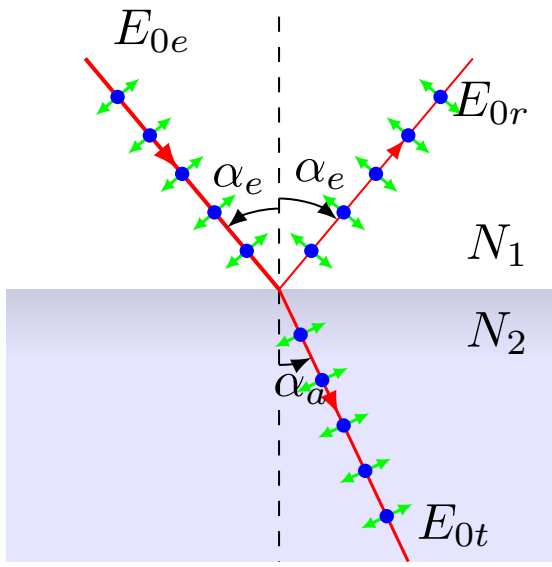


Abbildung 2.2.: **Reflexion an Grenzflächen:** Ein von oben links kommender Strahl wird an der Grenzfläche (Trennfläche zwischen oberer und unterer Halbebene) in das Material und die Umgebung reflektiert. Eintreffende und reflektierende Strahlen sind rot dargestellt. Senkrechte und parallele Polarisation zur Bildebene sind blau und grün dargestellt.

Eine anschauliche Übersicht weiterer Reflexionsmodelle ist z.B. in [TFR21] zu finden. Eine vorgefertigte Literaturübersicht zu allgemeinen Themen der Radarsignalverarbeitung ist in [ZHO23] zu finden.

Wird die reflektierte Welle wieder in Richtung des Radarsensors geworfen, so kann der Sensor die Welle abtasten (Amplitude, Polarisation) und durch geschickte Signalverarbeitung die Reflexionsparameter wie Reflexionsamplitude, Entfernung, Doppler und Einfallswinkel schätzen. Schwierigkeiten bei der Signalverarbeitung ergeben sich insbesondere durch Überlagerungen der Wellen aus der gesamten Szene und durch parasitäre Störsignale (z.B. thermisches Rauschen, elektromagnetische Störeinstrahlung), welche bei der Signalabtastung mit aufgezeichnet werden.

2.1.2. Definition des Radarkoordinatensystems

Die Reflexionsparameter sind relativ zur Position und Bewegung des Radarsensors zu betrachten, welcher hier an einer beliebigen Position der Fahrzeugkarosserie montiert ist. Es sei ein kartesisches Koordinatensystem (KOOS) P mit den Einheitsvektoren $\{\mathbf{e}_{P[x]}, \mathbf{e}_{P[y]}, \mathbf{e}_{P[z]}\}$ fest im Ursprung des Radarsensors definiert, vgl. Abbildung 2.3. Mit Hilfe dieses KOOSs lässt sich die Lage von Reflektoren aus Sicht des Radarsensors beschreiben. Dazu sei beispielhaft ein Reflektor T definiert, welcher sich an der Position $\mathbf{p} = \left[p_{[x]}, p_{[y]}, p_{[z]}\right]^T$ im Koordinatensystem P befindet.

In der Radarsignalverarbeitung ist eine Darstellung in Kugelkoordinaten üblich. Dazu sei im Ursprung von P ein weiteres Koordinatensystem B definiert, dessen erste

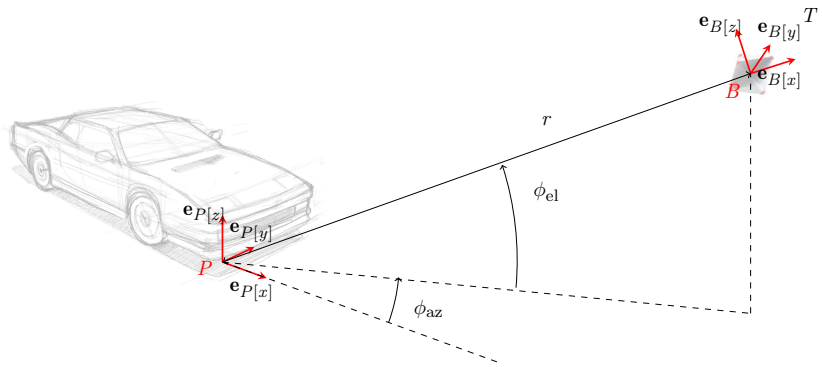


Abbildung 2.3.: **Transformation der Koordinatensysteme:** Reflektor im Radar Koordinatensystem, adaptiert von [Gva, Bos].

Koordinate $\mathbf{b}_{[x]}$ in Richtung des Reflektors zeigt.[2] Die Position des Reflektors im kartesischen Koordinatensystem lässt sich angeben durch

$$\mathbf{p} = \begin{bmatrix} p_{[x]} \\ p_{[y]} \\ p_{[z]} \end{bmatrix} = \begin{bmatrix} \cos(\phi_{\text{az}})\cos(\phi_{\text{el}}) & -\sin(\phi_{\text{az}}) & -\cos(\phi_{\text{az}})\sin(\phi_{\text{el}}) \\ \sin(\phi_{\text{az}})\cos(\phi_{\text{el}}) & \cos(\phi_{\text{az}}) & -\sin(\phi_{\text{az}})\sin(\phi_{\text{el}}) \\ \sin(\phi_{\text{el}}) & 0 & \cos(\phi_{\text{el}}) \end{bmatrix} \begin{bmatrix} b_{[x]} \\ b_{[y]} \\ b_{[z]} \end{bmatrix}. \tag{2.5}$$

Die Koordinatentransformation von B nach P („passive Drehung“) ergibt sich aus sequentieller Euler-Rotation um B_y und P_z um die Winkel ϕ_{el} und ϕ_{az}. Diese Winkel werden im weiteren Verlauf dieser Arbeit als Elevationswinkel und Azimutwinkel bezeichnet. Die kartesischen Koordinaten $\begin{bmatrix} p_{[x]} & p_{[y]} & p_{[z]} \end{bmatrix}^T$ für den Reflektor werden berechnet (Gleichung 2.5) durch das Einsetzen der Winkel ϕ_{az} und ϕ_{el} sowie des Translationsvektors $\mathbf{b} = \begin{bmatrix} b_{[x]} & b_{[y]} & b_{[z]} \end{bmatrix}^T = \begin{bmatrix} r & 0 & 0 \end{bmatrix}^T$.

Neben der Position sei noch die Geschwindigkeit in Kugelkoordinaten definiert. Es sei eine rein translatorische Bewegung in $\mathbf{p}(t)$ angenommen. Die polare Geschwindigkeit ergibt sich dann durch Umstellung von Gleichung 2.5 nach $\mathbf{b}(t)$ aus der zeitlichen Ableitung der Position in Kugelkoordinaten entsprechend

$$\frac{\mathrm{d}\mathbf{b}(t)}{\mathrm{d}t} = \begin{bmatrix} \cos(\phi_{\text{az}})\cos(\phi_{\text{el}}) & \sin(\phi_{\text{az}})\cos(\phi_{\text{el}}) & \sin(\phi_{\text{el}}) \\ -\sin(\phi_{\text{az}}) & \cos(\phi_{\text{az}}) & 0 \\ -\cos(\phi_{\text{az}})\sin(\phi_{\text{el}}) & -\sin(\phi_{\text{az}})\sin(\phi_{\text{el}}) & \cos(\phi_{\text{el}}) \end{bmatrix} \frac{\mathrm{d}\mathbf{p}(t)}{\mathrm{d}t}. \tag{2.6}$$

Hier entsprechen $\frac{\mathrm{d}b_{[x]}}{\mathrm{d}t}$ der Radialgeschwindigkeit und $\frac{\mathrm{d}b_{[y]}}{\mathrm{d}t}$ und $\frac{\mathrm{d}b_{[z]}}{\mathrm{d}t}$ den zwei Tangentialgeschwindigkeitskomponenten. Die Komponenten $\frac{\mathrm{d}p_{[x]}}{\mathrm{d}t}$, $\frac{\mathrm{d}p_{[y]}}{\mathrm{d}t}$ und $\frac{\mathrm{d}p_{[z]}}{\mathrm{d}t}$ entsprechen der longitudinalen, lateralen und vertikalen Geschwindigkeit am Sensorursprung.

[2] Hinweis: In Abbildung 2.3 wurde das Koordinatensystem B aus Gründen der Übersichtlichkeit in die Position des Reflektors geschoben, befindet sich jedoch im Ursprung von P.

2.1.3. Signalmodulation und Vorverarbeitung

In den beiden vorangegangenen Abschnitten wurde eine Definition von Reflexion an Grenzflächen zweier Übertragungsmedien und eine Definition von relativen Koordinaten gegeben. In diesem Abschnitt werden die Signalmodulation und Signalaufbereitung beschrieben, um später eine Schätzung der Reflexionsparameter über Signalverarbeitung zu ermöglichen. Wir konzentrieren uns dabei ausschließlich auf eine Modulation nach dem sogenannten „FMCW Chirp Sequence“ Prinzip, welches nachfolgend erläutert wird. FMCW beschreibt dabei, dass die vom Radarsensor ausgestrahlten EM-Welle in ihrer Frequenz nicht konstant sind. Der Begriff Chirp-Sequence (CS) fasst dabei zusammen, dass eine Folge von Wellen transmittiert wird.

2.1.3.1. Sendesignal

Das in dieser Arbeit verwendete Radargerät verwendet ein Sendesignal der Klasse FMCW-CS. Hierbei wird für einen kurzen Zeitbereich eine Sequenz von frequenzmodulierten EM-Wellen, den sogenannten Chirps, ausgesendet.

Nach [Sch13] kann das frequenzmodulierte Sendesignal eines Chirps dargestellt werden, als

$$s_{\mathrm{TX}}(t) = A_{\mathrm{TX}} \cos \Theta_{\mathrm{TX}}(t) = A_{\mathrm{TX}} \cos \left(2\pi \left(f_0 t + \frac{f_{\mathrm{sweep}}}{2\, T_{\mathrm{chirp}}} t^2 \right) \right), \tag{2.7}$$

wobei A_{TX}, f_0, f_{sweep} und T_{chirp} die Amplitude des Sendesignals, die Grundfrequenz, die Bandbreite und die Chirpperiode sind. t ist die Zeitvariable.

2.1.3.2. Empfangssignal

Das Sendesignal propagiert mit Lichtgeschwindigkeit durch die Szene und wird an Grenzflächen verschiedener Übertragungsmedien reflektiert, siehe Unterabschnitt 2.1.1. Strahlen die reflektierten Wellen wieder in Richtung des Sensors, kann ihre EM-Wirkung dort durch einen ADU zeitdiskret abgetastet werden. Die Abtastung erfolgt über die Dauer eines Chirps. Das Empfangssignal sei als zeitlich verschobene und gedämpfte Kopie des Sendesignals modelliert

$$s_{\mathrm{RX}}(t) = A_{\mathrm{RX}} \cos \left(\Theta_{\mathrm{TX}}(t - \tau(t)) \right), \tag{2.8}$$

mit Amplitude A_{RX}.

Die Signallatenz τ ergibt sich für Ziele konstanter Geschwindigkeit zu

$$\tau(t) = 2 \frac{r_0 + v_r t}{c}, \tag{2.9}$$

wobei r_0, v_r und c der initiale Zielabstand, die radiale Relativgeschwindigkeit und die Lichtgeschwindigkeit sind.

Das Signal s_{RX} wird anschließend mit dem Sendesignal gemischt, tiefpassgefiltert und hier unter der Annahme $A_{\mathrm{TX}} = A_{\mathrm{RX}} = 1$ durch das Runtermischen auf w_{TX} modelliert als

$$s_{\mathrm{d}}(t) = \cos \left(\Theta_{\mathrm{TX}}(t) - \Theta_{\mathrm{RX}}(t) \right) = \cos \left(\Theta_{\mathrm{d}}(t) \right). \tag{2.10}$$

Weiterführend gilt

$$\Theta_{\mathrm{d}}(t) = 2\pi \left(f_0 \tau(t) + \frac{f_{\mathrm{sweep}}}{T_{\mathrm{chirp}}} t\tau(t) + \frac{f_{\mathrm{sweep}}}{2T_{\mathrm{chirp}}} t\tau(t)^2 \right), \tag{2.11}$$

was durch Einsetzen von Gleichung 2.9, der Wellenlänge $\lambda = \frac{c}{f}$ und Vernachlässigung des Terms mit verschwindend geringer Größe durch $\tau(t)^2$ umformuliert wird in

$$\Theta_{\mathrm{d}}(t) = 2\pi \left(2\frac{r_0}{\lambda} + 2\frac{v_r}{\lambda} t + 2\frac{f_{\mathrm{sweep}}}{T_{\mathrm{chirp}}} \frac{r_0}{c} t \right). \tag{2.12}$$

Folgende zusätzliche Notationen seien definiert

$$\phi_0 = 2\frac{r_0}{\lambda}, \tag{2.13}$$

$$f_D = -2\frac{v_r}{\lambda} \tag{2.14}$$

und

$$f_R = 2\frac{f_{\mathrm{sweep}}}{T_{\mathrm{chirp}}} \frac{r_0}{c}, \tag{2.15}$$

wobei ϕ_0, f_D und f_R eine von der Entfernung abhängige konstante Phase, die Dopplerfrequenz und ein von der Entfernung abhängige Frequenz sind. Die Zeit wird diskretisiert zu $t = (kT_{\mathrm{sample}} + lT_{\mathrm{chirp}})$. Die Argumente k und l sind Laufindices zur Beschreibung der Zeit entlang eines Chirps sowie der Zeit über mehrere Chirps, in der Regel „fast-time“ und „slow-time“ genannt. Das zeitkontinuierliche Reflexionssignal eines Chirps $u_d(t)$ wird somit mit einer Abtastperiode von $T_{\mathrm{sample}} = \frac{T_{\mathrm{chirp}}}{K}$ digital abgetastet zu

$$s(k,l) = \cos\left(2\pi \left(\phi_0 + f_R(kT_{\mathrm{sample}} + lT_{\mathrm{chirp}}) - f_D(kT_{\mathrm{sample}} + lT_{\mathrm{chirp}}) \right) \right), \tag{2.16}$$

2.1.3.3. Entfernungsmessung

Mit steigendem $\frac{f_{\mathrm{sweep}}}{T_{\mathrm{chirp}}}$ wird der Einfluss des Doppler-Anteils innerhalb eines Chirps ($f_D k T_{\mathrm{sample}}$) vernachlässigbar gering, so dass sich die Entfernung direkt abschätzen lässt. Dazu wird das ins Basisband gemischte Zeitsignal jedes Chirps l durch eine Diskrete-Fourier-Transformation (DFT) (im Folgendem „Range-FT“ genannt) in den Frequenzraum umgewandelt

$$S_{\mathrm{r}}(u,l) = \sum_{k=0}^{K-1} s(k,l) e^{-2\pi j \frac{uk}{K}}. \tag{2.17}$$

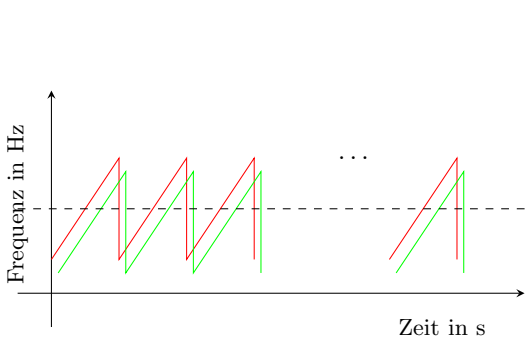
Das Zeitsignal der Länge K wird im Frequenzraum über die Frequenzvariable u beschrieben. Die Zielentfernung lässt sich im Frequenzspektrum über die Beziehung

$$R = u_{\max} \cdot \Delta R = u_{\max} \cdot \frac{c}{2f_{\mathrm{sweep}}}. \tag{2.18}$$

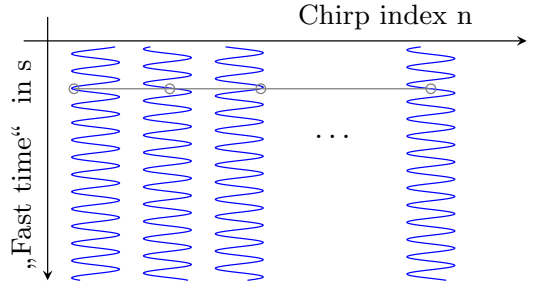
abschätzen, wobei $u_{\max}$ das Frequenzbin des lokalen Maximums im Spektrum ist.

Im vorliegenden Fall ist das abgetastete Zeitsignal reellwertig, wodurch das Frequenzsignal hermitesch gespiegelt wird. Signalfrequenzen oberhalb von $\frac{K}{2}$ werden durch einen analogen Tiefpass gedämpft.

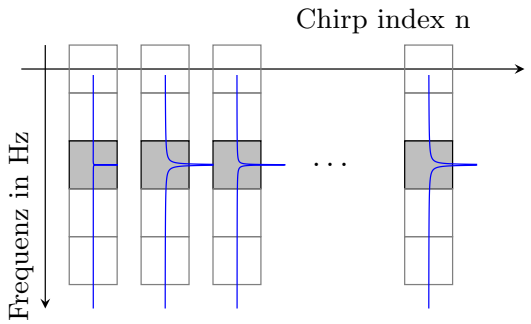
Eine anschauliche Übersicht der bisherigen Verarbeitungsschritte von der Signallatenz zwischen Sende- und Empfangssignal über die Darstellung des abgetasten Signals im Basisband und die Darstellung des Amplitudenspektrums nach der Fouriertransformation aus Gleichung 2.17 ist in Abbildung 2.4 dargestellt.



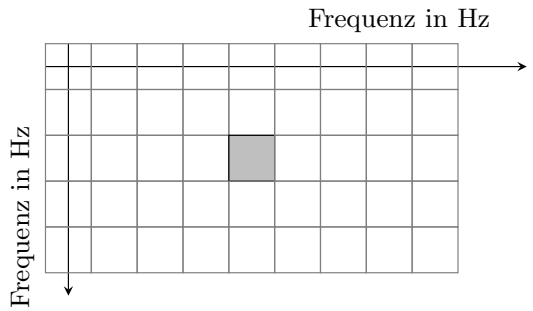
(a) Rot: Sendesignal. Grün: Empfangssignal



(b) Abgestatstes Empfangssignal im Basisband mit Hervorhebung der Phasenlagen der jeweiligen Chirps.



(c) Amplitudenspektrum nach Range DFT



(d) Amplitudenspektrum nach Range und Speed DFT

Abbildung 2.4.: **Signalmodulation und Vorverarbeitung:** Dargestellt sind die vier elementaren Verarbeitungsschritte bei CS Modulation.

2.1.3.4. Geschwindigkeitsmessung

Für die Entfernungsmessung wurde in Unterabschnitt 2.1.3.3 angenommen, dass der Doppler-Anteil innerhalb eines Chirps aufgrund kurzer Beobachtungszeit vernachlässigbar gering ist. Die Beobachtungszeit eines Reflektors über mehrere aneinander gereihte Chirps ist jedoch deutlich länger[3], so dass der Einfluss von Doppler bzw. die Geschwindigkeit über mehrere Chirps hinweg nicht vernachlässigt werden kann. Die Geschwindigkeit führt dazu, dass sich der radiale Abstand eines Reflektors zum Sensor marginal entlang der Zeit verändert. Hierdurch ergibt sich eine Verschiebung der Phase, siehe Abb. 2.4b,

[3] man spricht von der sogenannten „slow time“

welche durch eine zweite DFT entlang des Chirp Index l gemessen werden kann

$$S_{\mathrm{rd}}(u,v) = \sum_{l=0}^{L-1} S_{\mathrm{r}}(u,l)\, e^{-2\pi j \frac{vl}{L}}. \tag{2.19}$$

Diese DFT sei weiterführend „Speed-DFT“ genannt. Diese DFT hat eine Frequenzauflösung von

$$\Delta f_D = \frac{1}{L \cdot T_{\mathrm{chirp}}} \tag{2.20}$$

welche mit einer Phasenverschiebung um $2\frac{vT_{\mathrm{chirp}}}{\lambda}$, vgl. Gleichung 2.9, zu der Relativgeschwindigkeit

$$v_r = v_{\max} \cdot \frac{\lambda}{2} \Delta f_D = v_{\max} \cdot \Delta v \tag{2.21}$$

wird, wobei $_{\max}$ die zum Reflektor korrespondiernde Binposition im Leistungsspektrum ist.

Zum einfacheren Verständnis der Signale wurden z.B. Kopplungsterme zwischen r und v_r nicht weiter berücksichtigt. Der interessierte Leser sei an dieser Stelle an [Sch13] verwiesen.

Das Leistungsspektrum nach Range- und Speed-DFT wird häufig als RD-map bezeichnet. Werden mehrere Sende- und Empfangsantennen verwendet, so ergibt sich die RD-map als inkohärent addiertes Leistungsspektren der Kanäle ($S_{\mathrm{rd,i}}$) zu

$$\mathbf{RD}(u,v) = 20 \log_{10} \sum_i \left| S_{\mathrm{rd,i}}(u,v) \right|. \tag{2.22}$$

Im weiteren Verlauf dieser Arbeit wird die gesamte RD-map als Matrix verarbeitet. Als Darstellung wird die Notation $\mathbf{RD}_{[u,v]}$ verwendet, bei welcher die Indices der Matrix in den tiefgestellten eckigen Klammern dargestellt sind.

2.1.3.5. Doppler-Mehrdeutigkeit

Bedingt durch die Signalabtastung verhält sich, gemäß Nyquist-Shannon Abtasttheorem [Sha49], die maximale Frequenz reziprok zur Abtastperiode. Dies führt dazu, dass die eindeutige Doppler-Geschwindigkeit aus dem komplexwertigen Signal limitiert ist auf

$$f_{D,\max/\min} = \pm \frac{1}{2T_{\mathrm{chirp}}}, \tag{2.23}$$

beziehungsweise

$$v_{\mathrm{r},\max/\min} = \pm \frac{\lambda}{4T_{\mathrm{chirp}}}. \tag{2.24}$$

Überschreitet die Geschwindigkeit eines Objektes diese Eindeutigkeitsgrenze, so wird seine Abbildung in der RD-map zyklisch projiziert. Ein Beispiel dieses Verhaltens ist in Abbildung 2.5 zu sehen. Die linken beiden Bilder stellen Kamerabilder des Szenarios dar, aufgenommen rechts seitlich aus dem Fahrzeug, sowie nach hinten gerichtet. Rechts daneben ist das dazugehörige RD-map dargestellt für den Radarsensor, welcher hinten rechts im Fahrzeug montiert ist. In der RD-map sind zwei gebogene Streifen erhöhter

Helligkeit zu erkennen. Es kann angenommen werden, dass der obere Streifen den Reflexionen des Bordsteins entspricht und der untere Streifen höchstwahrscheinlich der Reflexion der Fassade des Industriegebäudes oder des Zaunes entspricht. Bedingt durch den Einfallswinkel weisen die ausgedehnten Objekte Bordstein, Zaun und Hausfassade-ausgeprägte kinematische Signaturen auf. Diese starten ungefähr bei $-5\,\mathrm{m\,s^{-1}}$ Doppler bis hin zur eindeutigen Doppler-Geschwindigkeit. Danach werden die Ausläufer der Objekte am linken Rand des RD-maps gespiegelt.

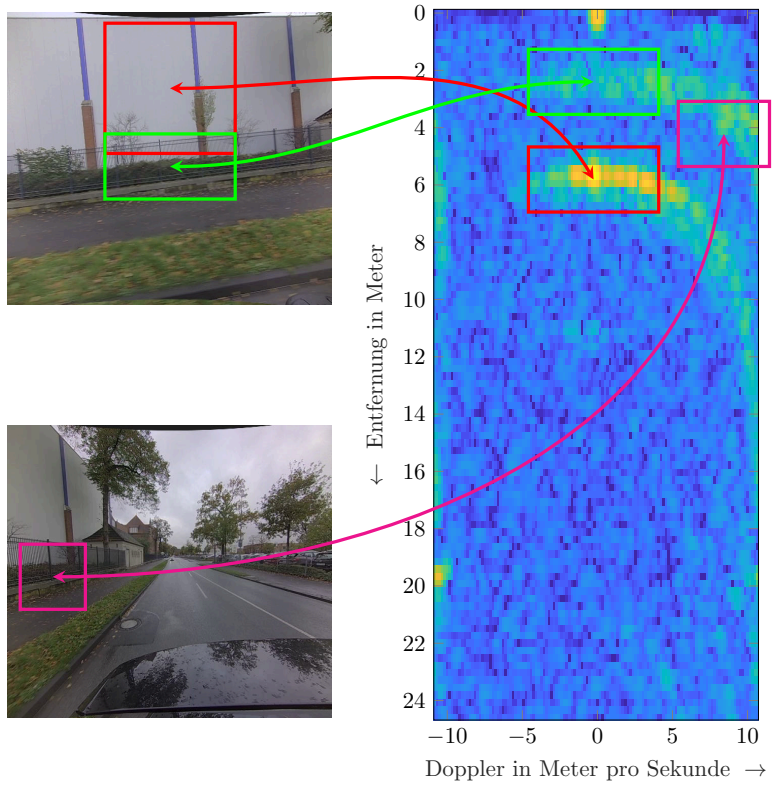


Abbildung 2.5.: **Abbildung einer typischen Fahrszene mit stationären Objekten:** Links: Kameraabbildungen rechts aus dem Fahrzeug heraus, sowie nach hinten gerichtet. Rechts: RD-map der Szene. Bildhelligkeit verhält sich proportional zur Leistung. Szeneninhalt im Kamerabild und RD-map wurden durch farbige Rechtecke verbunden.

2.1.4. Zieldetektion

In Unterunterabschnitt 2.1.3.2 wurde die Amplitude des Empfangssignals zunächst äquivalent zur Sendeamplitude angenommen. Diese Vereinfachung diente ausschließlich der anschaulicheren Beschreibung der CS-Vorverarbeitung. Wie aus Unterabschnitt 2.1.1 bekannt ist, findet jedoch eine Amplitudendämpfung der EM-Welle an Grenzflächen zwischen zwei Übertagungsmedien statt. Leider ergeben sich noch weitere Dämpfungen der Signalamplitude, welche eine Bestimmung der Reflexionsparameter erschweren. Eine makroskopische Beschreibung dieser Dämpfung gibt die bekannte „Radargrundglei-

chung“ an:

$$P_r = \frac{P_t G_t G_r \lambda^2 \sigma}{(4\pi)^3 r^4}. \tag{2.25}$$

Hier beschreiben P_r und P_t die Empfangs- bzw. Sendeleistung. G_t und G_r sind die Verstärkungsfaktoren („Antennengewinn“) für Sende- und Empfangsantennen. σ ist die Pseudogröße RCS und gibt die Rückstrahlleistung eines Objektes im Vergleich zu einem perfekt leitenden Kugelreflektor mit einer Oberfläche von $1\,\mathrm{m}^2$ an. r ist der einfache Abstand zwischen Radargerät und Reflektor. Für tiefere Einblicke zur Radargrundgleichung sei auf weiterführende Literatur wie [Sko01] verwiesen.

Die Amplitude eines Reflektors in der RD-map ergibt sich aus der Leistung nach Gleichung 2.25. In Unterabschnitt 2.1.1 wurde bereits erwähnt, dass sich neben den Signalen der Reflexion auch parasitäre Rauschanteile in das abgetastete Empfangssignal mischen. Das breitbandige Rauschen verringert den Signal-Rausch-Abstand bzw. maskiert die Signale vollständig und erschwert deren Detektion. Werden diese Signale nicht detektiert, so spricht man von einem falsch-negativ (engl.: „false-negative“) Fehler. Werden Rauschsignale hingegen fälschlicherweise als Ziel detektiert, so spricht man vom falsch-positiv (engl.: „false-positive“) Fehler. Letztere Fehler führen zu sogenannten Falschalarmen. Im Kontext von Fahrerassistenzsystemen könnten diese Falschalarme einen Regeleingriff hervorrufen, ohne dass dieser Eingriff gewollt und notwendig gewesen wäre. Bei falsch-negativen Fällen könnte ein notwendiger Regeleingriff ausbleiben. In ihrer einfachsten Form wird bei der Zieldetektion der Leistungswert des Empfangssignals ausgewertet und gegenüber einem definierten Schwellwert verglichen. Überschreitet die Leistung den Grenzwert, so wird von einem Ziel ausgegangen. In der Bildverarbeitung werden darüber hinaus noch andere Detektionsalgorithmen (Pixel-, Kanten-, Regionen-, Textur- und modellorientierte Verfahren) [Jäh05] verwendet, wobei in der Radarsignalverarbeitung hauptsächlich pixelorientierte Verfahren zum Einsatz kommen. Für den interessierten Leser sei ein Beispiel solch eines Detektors nachfolgend gegeben.

Nach [Sch13] seien die Hypothesen

$$H_0 : x = n_g \tag{2.26}$$
$$H_1 : x = n_g + s \tag{2.27}$$

definiert, bei denen n_g als komplexwertiges Gaußsches-Rauschen und s als komplexwertiges Signal eines Pixels in der RD-map ist. Ein „square law detector“ versucht dann, einen adaptiven Schwellwert T zu finden, welcher durch

$$y \leq T \rightarrow H_0 \tag{2.28}$$
$$y > T \rightarrow H_1 \tag{2.29}$$

eine Entscheidung anhand der gemessenen Leistung $y = |x|^2$ bzw. hier $y = \mathbf{RD}_{[u,v]}$ trifft. Hier wird ein zum Cell Averaging Constant False Alarm Rate (CA-CFAR) analoger Detektor [Sch13] verwendet. Dieser Detektor schätzt den Rauschpegel und bestimmt daraus adaptiv den Schwellwert. Die Rauschpegelschätzung entspricht der mittleren Amplitude der Rauschleistung in der Nachbarschaft eines zu untersuchenden Pixels bzw.

Zelle in der RD-map:

$$Z = \frac{\sum_{u_t=-N}^{N} \sum_{v_t=-N}^{N} \mathbf{RD}_{[u_r+u_t, v_r+v_t]}}{(2N+1)^2}. \tag{2.30}$$

Hierbei sei N definiert als die Größe der einseitigen Pixelnachbarschaft, welche zur Schätzung des Rauschlevels verwendet wird. Die Pixelnachbarschaft besteht somit aus $(2N+1)^2$ Pixeln. u_r und v_r seien die Pixelpositionen in Entfernung und Geschwindigkeitsrichtung. Parametriert sei der Schwellwert T als Summe

$$T = Z + 3\sigma_{\mathrm{N}}, \tag{2.31}$$

wobei Z die mittlere Rauschleistung und σ_{N} die Standardabweichung des Rauschens ist. Die Standardabweichung des Rauschens kann beispielsweise über die gesamte RD-map geschätzt werden. Die Leistung eines Pixels in der RD-map muss also um mindestens $3\sigma_{\mathrm{N}}$ größer als die mittlere Leistung der Nachbarschaft sein, um als Ziel nach Gleichung 2.29 erkannt zu werden. Wird das Rauschen als Gauß-Verteilung angenommen, so wird durch diesen Schwellwert ein Großteil aller Realisierungen des Rauschens berücksichtigt. Für weitere Implementierungsmöglichkeiten zum CFAR-Verfahren sei auf weiterführende Literatur verwiesen, z.B. [Sch13].

Als illustratives Beispiel ist in Abbildung 2.6 das Ergebnis von Rauschlevelschätzung und Zieldetektion dargestellt. Die Größe der Pixelnachbarschaft in der RD-map entsprach $N = 5$.

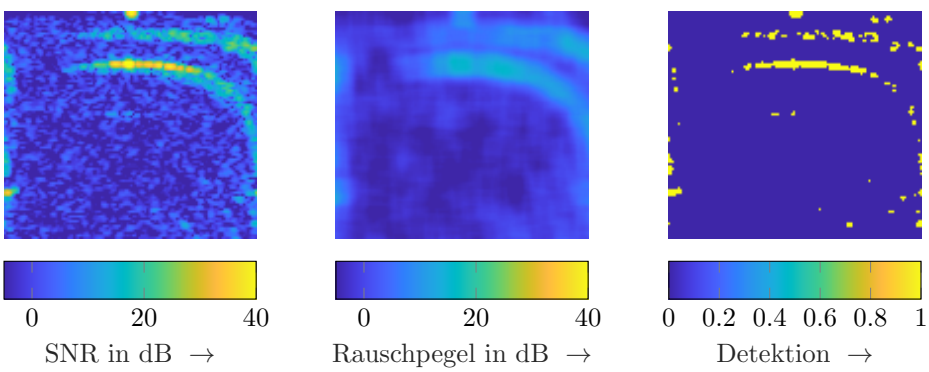


Abbildung 2.6.: **Zieldetektion mittels CFAR:** Links: Typisches RD-map aus Fahrszene. Mitte: Ergebnis der Rauschlevelschätzung mit $N = 5$. Rechts: Ergebnis der CFAR-Detektion. Heller Bildbereiche entsprechen Bewertung nach H_1 Hypothese.

Es ist zu erkennen, dass der CFAR-Algorithmus die Bereiche mit erhöhtem Signal-Rausch-Verhältnis (engl.: Signal-to-noise-ratio (SNR)) als Ziele detektiert hat.

Wurde ein Ziel in der RD-map detektiert, so sind seine Entfernung und seine radiale Geschwindigkeit gegenüber dem Radarsensor anhand seiner Pixelposition in der RD-map und Gleichung 2.18 sowie 2.21 zu schätzen.

2.1.5. Einfallswinkelschätzung

Neben der Zieldetektion nach Unterabschnitt 2.1.4 und der damit verbundenen Identifikation von Entfernung und Doppler von signifikanten Reflexionen in der RD-map ist die Identifikation des Einfallswinkels der Reflexionen eine typische Anwendung bei automotiven Radargeräten. Eine ausführliche Übersicht möglicher Verfahren zur Einfallswinkelschätzung ist beispielsweise in [BM13, Che09] zu finden. In dieser Arbeit wird vom fest verbauten Radargeräten ohne mechanische Schwenkung der Antennen ausgegangen. Eine Vermessung des Einfallswinkels wird ausschließlich über die Verwendung mehrerer Empfangsantennen realisiert, wobei der Signalunterschied der Reflexion zwischen den Antennen prozessiert wird. Eine graphische Übersicht dieses Signal- bzw. Phasenunterschiedes ist in Abbildung 2.7 dargestellt.

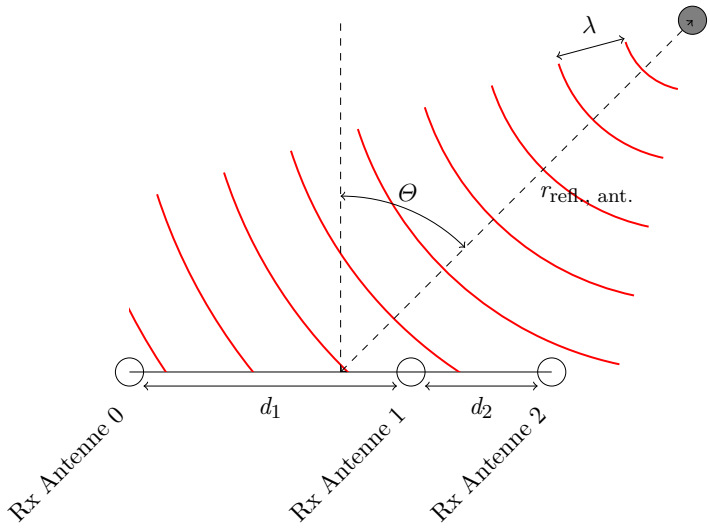


Abbildung 2.7.: **Phasenlage der reflektierten Welle an Empfangsantennen:** Lage EM-Wellen strahlen in lineares Antennenarray. In rot gezeichnet sind beispielhafte Wellenfronten. Durch die unterschiedliche Position der Antennen ergeben sich unterschiedliche Phasenlagen der Wellen.

Dargestellt ist ein Reflektor, welcher zuvor aktiv durch das Sendesignal bestrahlt wurde. Die rot eingezeichneten Kreisbögen sollen die EM-Wellenfronten bzw. Wellenberge der Reflexion darstellen. Zwischen den Wellenfronten ergibt sich ein Abstand gemäß der Wellenlänge λ. Außerdem sind die Positionen dreier Empfangsantennen (Rx Antenna 0, Rx Antenna 1, Rx Antenna 2) mit einem Abstand d_1 bzw. d_2 skizziert. In der Abbildung ist zu erkennen, dass die Wellenfronten die Positionen der Empfangsantennen zu unterschiedlichen Zeitpunkten erreichen bzw. dass die Wellen an den Empfangsantennen unterschiedliche Phasenausprägungen aufweisen.

Die Krümmung κ der Wellenfronten verhält sich reziprok gegenüber dem Abstand r zum Kreismittelpunkt ($\kappa = \frac{1}{r}$), siehe z.B. [Wik22]. Bei hinreichend großem Abstand lässt sich die Wellenfront am ausgedehnten Antennenarray[4] als Gerade annähern. Man spricht dann von einer „ebenen Welle“. Gemäß des sogenannten „Fraunhofer-Abstandes“ ist eine ebene Welle als Annäherung einer Welle erfüllt, wenn der punktförmige Reflektor

[4] Als Antennenarray wird die Anordnung der Empfangsantennen bezeichnet.

im Fernfeld r_{fern} des Antennearrays liegt [Kra88]

$$r_{\text{refl., ant.}} > r_{\text{fern}} = \frac{2L^2}{\lambda} = \frac{2(d_1 + d_2)^2}{\lambda}, \tag{2.32}$$

wobei L die Ausdehnung („Apertur“) der Antenne beschreibt.

Unter der Annahme dieser Bedingung werden nun zwei klassische DoA-Schätzer vorgestellt, welche im weiteren Verlauf dieser Arbeit als Referenzverfahren gegen eigene Winkelschätzer verwendet werden.

2.1.5.1. Monopulsverfahren

Für diese ebene Wellenfornt lässt sich der Wellenvektor in die kartesischen Komponenten

$$\mathbf{k} = k_x \mathbf{e}_x + k_y \mathbf{e}_y \tag{2.33}$$
$$k_x = \sin(\Theta)\|\mathbf{k}\| \tag{2.34}$$
$$k_y = \cos(\Theta)\|\mathbf{k}\| \tag{2.35}$$
$$\|\mathbf{k}\| = \frac{2\pi}{\lambda} \tag{2.36}$$

aufteilen, wobei Θ nach Abbildung 2.7 der Aspektwinkel gegenüber einer zum Antennenarray aufgespannten Normalen ist. Das Koordinatensystem wird hier so gewählt, dass die Antennen auf der x-Achse liegen. Die Phasendifferenz an den Antennen ergibt sich als skalare Projektion des Wellenvektors auf den Richtungsvektor zwischen den Antennen. Somit ergibt sich die Phasendifferenz $\Delta\Phi$ als

$$\Delta\Phi = k_x d = \sin(\Theta)\frac{2\pi}{\lambda} d, \tag{2.37}$$

Durch Auflösen nach Θ ergibt sich

$$\Theta = \arcsin\left(\frac{\lambda\Delta\Phi}{2\pi d}\right), \tag{2.38}$$

welches bei der Kenntnis über die rechten Parameter zur Schätzung des Einfallswinkels führt.

Bei der Bestimmung des Einfallswinkels in einem Antennenarray werden die gemessenen Phasenwinkel und Antennenabstände aller Antennen berücksichtigt. Zunächst sei ein Vektor mit den gemessenen Phasendifferenzen zwischen geometrisch benachbarten Antennen definiert

$$\mathbf{a} = \begin{bmatrix} \angle\left(S_{\text{rd},1}(u,v), S_{\text{rd},2}(u,v)\right) \\ \angle\left(S_{\text{rd},1}(u,v), S_{\text{rd},3}(u,v)\right) \\ \vdots \\ \angle\left(S_{\text{rd},1}(u,v), S_{\text{rd,N}}(u,v)\right) \end{bmatrix} = \begin{bmatrix} a_{1,2} \\ a_{1,3} \\ \vdots \\ a_{1,N} \end{bmatrix}, \tag{2.39}$$

wobei die Phasenwinkel folgend berechnet werden:

$$\angle\Big(S_{\mathrm{rd,i}}(u,v), S_{\mathrm{rd,i\text{-}1}}(u,v)\Big) = \arg\left(S_{\mathrm{rd,i}}(u,v)S^{*}{}_{\mathrm{rd,i\text{-}1}}(u,v)\right). \tag{2.40}$$

Außerdem wird ein Vektor der geometrischen Antennenabstände definiert zu

$$\mathbf{b} = \frac{2\pi}{\lambda}\begin{bmatrix} x_2 - x_1 \\ x_3 - x_1 \\ \vdots \\ x_N - x_1 \end{bmatrix} = \begin{bmatrix} b_{1,2} \\ b_{1,3} \\ \vdots \\ b_{1,N} \end{bmatrix}, \tag{2.41}$$

wobei x_i mit $i = 1, \cdots, N$ die horizontalen Positionen der Antennen sind.

In Gl. 2.37 werden die Phasenwinkel unbeschränkt modelliert. Gleichung 2.40 bildet jedoch die Phasenwinkel nur im Intervall $[-\pi, +\pi)$ ab, so für die korrekte Bestimmung des Einfallswinkels die gemessenen Phasenwinkel eindeutig gemacht werden müssen. Dazu wird zunächst der Einfallswinkel, basierend auf einem eindeutigen Antennenpaar[5] (hier: Antennenpaar $(1, 2)$) gelöst (Bedingung: $d < \lambda/2$),

$$i_{k,l} = \Big[a_{1,2}/\pi * b_{k,l}\Big]. \tag{2.42}$$

Als eindeutiges Antennenpaar wird ein Antennenpaar bezeichnet, dessen Abstand an den Grenzen des Field-of-View (FoV), z.B. für $\Theta = 90°$ gerade eine noch eindeutige Phasendifferenz von $\pm\pi$ ergibt

$$d_{\mathrm{unamb.}} = \frac{\pi\lambda}{2\pi} = \frac{\lambda}{2}. \tag{2.43}$$

Basierend darauf wird die erwartete Phasenabbildung für jedes uneindeutige Antennenpaar berechnet und synthetisch zur gemessenen Phasendifferenz addiert, so dass sich die Phasendifferenzen wieder linear proportional zum Antennenabstand verhalten

$$\mathbf{a}_{\mathrm{linear}} = \begin{bmatrix} a_{1,2} \\ a_{1,3} \\ \vdots \\ a_{1,N} \end{bmatrix} - 2\pi \begin{bmatrix} i_{1,2} \\ i_{1,2} \\ \vdots \\ i_{1,N} \end{bmatrix}. \tag{2.44}$$

Das lineare Gleichungssystem ergibt sich somit analog zu Gleichung 2.38 zu

$$\mathbf{a} = \mathbf{b}\sin(\Phi) \tag{2.45}$$

und kann mit Hilfe der Moore-Penrose Pseudo-Inversen [Pen55] gelöst werden zu

$$\sin(\Phi_{\mathrm{PM}}) = (\mathbf{b}^T\mathbf{b})^{-1}\mathbf{b}^T\mathbf{a}. \tag{2.46}$$

Eine Schätzung des Einfallswinkels für Antennenarrays ergibt sich mittels Phase-

[5] Nach Gleichung 2.37 ist die Phasendifferenz proportional zum Antennenabstand.

comparison-Monopulse (PM) somit zu

$$\Phi_{\mathrm{PM}} = \arcsin\left(\mathbf{b}^T\mathbf{b}\right)^{-1}\mathbf{b}^T\mathbf{a}. \tag{2.47}$$

2.1.5.2. Beamformer

Wurden beim Monopulsverfahren ausschließlich die gemessenen Phasendifferenzen zwischen den Antennen zur Winkelschätzung verwendet, so integriert der Bartlett Beamformer [KV96] zusätzlich die gemessenen Signalamplituden und erreicht ein besseres Schätzergebnis. Außerdem wird durch die Anwendung des Beamformers ein Winkelspektrum erzeugt. Das zuvor erzeugte zweidimensionale Leistungsspektrum bzw. RD-map wird somit um die Dimension zum Sinus des Winkels erweitert. Starke Reflexionen erzeugen ein lokales Maximum über alle drei Dimensionen. Die Entfernung, der Doppler und der Winkel ergeben sich automatisch aus der Auflösung entlang der drei Dimensionen. Ein Vorteil des Beamforming (BF) gegenüber den Punktschätzern, z.B. gegenüber PM, ist, dass Reflexionen auch im Winkel aufgelöst werden und in Entfernung und Doppler überlagerte Reflexionen somit über den Winkel voneinander getrennt werden können, sofern es die Winkelauflösung und spektralen Signaleigenschaften zulassen.

Für die Implementierung des BF-Verfahrens sei zunächst den Antennenabstand zum Antennenursprung als diskrete Variable definiert

$$d[n] = d_s\frac{\lambda}{2}n, \tag{2.48}$$

wobei $d_s = \frac{2d}{\lambda}$ ein Skalar ist, welcher den Abstand der Antennen als Vielfaches der halben Wellenlänge skaliert. $n \in [0, 1, N-1]$ sei der Laufindex zur Indexierung über die N Empfangsantennen.

Das Empfangssignal x für alle N Antennen sei modelliert nach Gleichung 2.37 zu

$$x[n] = e^{j\frac{2\pi}{\lambda}\sin(\Theta)d[n]} = e^{j\frac{2\pi}{\lambda}d_s\sin(\Theta)\frac{\lambda}{2}n}. \tag{2.49}$$

Mittels DFT ergibt sich das Winkelspektrum X zu

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N} = \sum_{n=0}^{N-1} e^{j\frac{2\pi}{\lambda}d_s\sin(\Theta)\frac{\lambda}{2}n}e^{-j2\pi kn/N}. \tag{2.50}$$

Der gesamte Exponent kann abgelesen werden zu

$$j\frac{2\pi}{\lambda}d_s\sin(\Theta)\frac{\lambda}{2}n - j2\pi kn/N. \tag{2.51}$$

Eine kohärente Signalintegration ergibt sich, wenn der Exponent gerade Null wird, wodurch der Exponent aufgelöst werden kann zu

$$\sin(\Theta) = \frac{2k}{d_sN}. \tag{2.52}$$

Der DFT Index k kann damit zur Beschreibung des Einfallswinkels verwendet werden. Die Winkelauflösung des BF ergibt sich als Abstand zweier benachbarter DFT-Zellen,

bzw. Setzen von $k = 1$ zu

$$\Delta\Theta = \arcsin\left(\frac{2}{d_s N}\right). \tag{2.53}$$

Einen Punktschätzer des Einfallswinkels Φ_{BF} für ein Pixel in der RD-map erhält man an der Stelle der maximalen Leistung im Winkelspektrum zu

$$k_{\max} = \underset{k}{\operatorname{argmax}} X[k]X^*[k] \tag{2.54}$$

$$\Phi_{\mathrm{BF}}(k_{\max}) = \arcsin\left(\frac{2k_{\max}}{d_s N}\right). \tag{2.55}$$

Für die Implementierung mittels DFT ist weiterhin zu beachten, dass bei nicht äquidistanten Antennenabständen[6] die Empfangssignale gemäß des minimalen Antennenabstandes mit Nullen aufgefüllt werden müssen. Dieses, als „zero-padding“ benannte, Verfahren ermöglicht darüber hinaus eine Interpolation des Winkelspektrums bzw. eine feinere Diskretisierung und somit eine erhöhte Schätzgenauigkeit von Φ_{BF}.

2.2. Inferenz und Training bei neuronalen Netzwerken

Als Deep-Neural-Network (DNN) bezeichnet man Funktionsapproximatoren, die Eingangsdaten in mehreren Schichten nichtlinear verarbeiten. Durch diese schichtweise, nichtlineare Verarbeitung können diese Netzwerke jede beliebige Übertragungsfunktion abbilden. Der interessierte Leser findet unter dem Begriff „universal approximation theorem“ viele interessante Artikel. Diese theoretische Eigenschaft macht DNNs universell einsetzbar für Aufgaben, bei denen ein deterministischer oder statistischer Zusammenhang zwischen Ein- und Ausgabedaten existiert. Neben dem Aufbau des Netzwerkes ist das Training der Netzwerkparameter kritisch für den erfolgreichen Einsatz des DNNs. In diesem Abschnitt wird beides kurz beleuchtet und somit die Grundlage für das Verständnis der Signalverarbeitung mittels NN in dieser Arbeit geliefert.

2.2.1. Vorwärtsdurchlauf

Für die Inferenz werden die Eingangsdaten schichtweise verarbeitet. Es sei zunächst eine einfache Netzwerkschicht[7] definiert zu

$$\mathbf{y} = f(\mathbf{W}\mathbf{x} + \mathbf{b}). \tag{2.56}$$

Hierin sind $\mathbf{x}$ und $\mathbf{y}$ ein Eingangsvektor bzw. ein Ausgangsvektor. $\mathbf{W}$ und $\mathbf{b}$ sind trainierbare Gewichtsmatrix und Bias. $f(\cdots)$ ist eine nichtlineare Aktivierungsfunktion, wie z.B. Sigmoid, Tangens hyperbolicus oder ReLU. Der Eingangsvektor $\mathbf{x}$ wird somit durch $\mathbf{W}$ und $\mathbf{b}$ linear und das Ergebnis durch $f(\cdots)$ nichtlinear transformiert.

Diese Transformation wird in einem NN mehrfach hintereinander ausgeführt. Dazu

[6] man spricht dann von nicht Uniform-Linear-Array (ULA) Antennen-Array

[7] Demonstriert wird das Netzwerk anhand von sogenannten „fully connected layer“.

sei das beispielhafte Netzwerk mit drei Schichten definiert

$$\mathbf{y}_1 = f_1(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) \tag{2.57}$$
$$\mathbf{y}_2 = f_2(\mathbf{W}_2\mathbf{y}_1 + \mathbf{b}_2) \tag{2.58}$$
$$\mathbf{o} = f_3(\mathbf{W}_3\mathbf{y}_2 + \mathbf{b}_3). \tag{2.59}$$

Jede Netzwerkschicht hat eigenständig trainierbare Projektionsparameter $\mathbf{W}_i$ und $\mathbf{b}_i$ und eine eigens wählbare Nichtlinearität $f(\cdots)_i$. Die Prädiktion des Netzwerkes wird durch $\mathbf{o}$ geliefert.

2.2.2. Rückwärtsdurchlauf

Damit diese Prädiktion sinnvolle Werte erreicht, müssen Projektionsparameter $\mathbf{W}_i$ und $\mathbf{b}_i$ während einer Trainingsprozedur optimiert werden. Dazu wird zunächst die Abweichung vom Zielwert $\mathbf{o}_{\mathrm{gt}}$ ermittelt. Das Subskript *gt* wurde hier in Anlehnung an den englischen Begriff „ground-truth“ gewählt. Je nach Anwendung kann diese Abweichung z.B. als Differenz und konvexer Skalierungsfunktion $\tau(\cdot)$ als

$$l(\mathbf{o}, \mathbf{o}_{\mathrm{gt}}) = \tau\left(\mathbf{e}\right) = \tau\left(\mathbf{o} - \mathbf{o}_{\mathrm{gt}}\right), \tag{2.60}$$

dargestellt werden.

Häufig werden die Parameter mit einem „gradient descent“ Verfahren mit Schrittweite γ trainiert, so dass die Netzwerkparameter iterativ von Schritt (n) nach $(n+1)$ aktualisiert werden zu

$$\mathbf{W}_i^{(n+1)} = \mathbf{W}_i^{(n)} - \gamma \frac{\partial l}{\partial \mathbf{W}_i^{(n)}} \tag{2.61}$$
$$\mathbf{b}_i^{(n+1)} = \mathbf{b}_i^{(n)} - \gamma \frac{\partial l}{\partial \mathbf{b}_i^{(n)}}. \tag{2.62}$$

Der letzte Term beschreibt den Gradienten der Fehlerfunktion, bezogen auf die Netzwerkparameter $\mathbf{W}$ und $\mathbf{b}$, und kann durch Anwendung der Kettenregel berechnet werden. Für die Parameter der Schichten ergeben sich damit

$$\frac{\partial l}{\partial \mathbf{W}_i^{(n)}} = \frac{\partial l}{\partial \tau}\frac{\partial \tau}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\frac{\partial \mathbf{o}}{\partial \mathbf{W}_i^{(n)}} \tag{2.63}$$
$$\frac{\partial l}{\partial \mathbf{b}_i^{(n)}} = \frac{\partial l}{\partial \tau}\frac{\partial \tau}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial \mathbf{o}}\frac{\partial \mathbf{o}}{\partial \mathbf{b}_i^{(n)}}. \tag{2.64}$$

Die Projektionsparameter der anderen Schichten können analog aktualisiert werden. Wesentlich für das Verständnis dieser Arbeit ist die Erkenntnis, dass zur Anwendung der Parameteraktualisierung die Gradienten von der gemessenen Abweichung l, bis zu den Projektionsparametern bestimmt werden können und so die Fehler in die Netzwerkschichten zurück propagiert werden können.

Für eine umfangreichere Beschreibung vieler Themen zur Nutzung von NNen seien dem Leser die Werke [Sch14, GBC16] empfohlen.

3. Aufbau und Kalibrierung des Sensorsystems

Für die im späteren Verlauf dieser Arbeit vorgestellten Algorithmen und Versuche wurde ein Fahrzeug aufgebaut, welches mit einer Vielzahl von Sensoren ausgestattet ist. Ziel ist eine detaillierte, berührungslose Abtastung der Umgebung für die automatische Ermittlung von Zielwerten für den ebenfalls verbauten Radarsensor vom Typ HELLA 5Gen 77 GHz. Mit den Zielwerten sollen später Machine-Learning (ML)-Algorithmen zur Signalverarbeitung trainiert werden. Als Referenzsensoren kommen zwei Lidarscanner vom Typ Velodyne VLP-32C [LiD20] zum Einsatz. Diese aktiven Sensoren strahlen EM-Pulse für bestimmte Elevations- und Azimutwinkel aus und schätzen aus der Reflexion die Tiefe. Durch die Menge der Tiefenmessungen wird dann eine spärliche Punktwolke der Umgebung konstruiert. Ebenso werden zwei Kameras vom Typ FirstSensor DC3C-1-E4P-105 [Fir] verwendet. Kameras sind passive Sensoren und tasten die Wellenlängen im sichtbaren Frequenzbereich für diskrete Elevation- und Azimutwinkel ab. Durch Anordnung benachbarter Winkel werden so Farbbilder konstruiert, welche eine optische Abbildung der Umgebung darstellen. Für Informationen bezüglich der Bewegung und Position des Ego-Fahrzeuges wird ein Differential-GPS-with-Inertial-Navigation-System (DGPS-INS)-System vom Typ GeneSys ADMA-G-Pro+ [Gen20] verwendet. Es sei festzuhalten, dass die später aufgebauten Algorithmen mit diesen Sensoren getestet wurden und somit eine Definition an dieser Stelle notwendig ist. Aus Sicht des Autors ist ein Austausch mit artverwandten Sensoren aber technisch möglich. Eine Darstellung des Versuchsträgers mit den eingebauten Sensoren ist in Abbildung 3.1 gegeben.

3.1. Definition der Koordinatensysteme

Jeder Sensor tastet die Umgebung relativ zu seiner eigenen Position und Ausrichtung ab. Die Sensordaten liegen dann in dem für jeden Sensor spezifischen KOOS vor, siehe Unterabschnitt 2.1.2. Zur Beschreibung der Sensordaten definieren wir für jeden Sensor ein kartesisches KOOS im Sensorursprung entsprechend der Sensorausrichtung. Eine qualitative Übersicht der KOOSen ist in Abbildung 3.1 gegeben. Um die Sensordaten später miteinander fusionieren zu können, müssen die Daten zwischen den KOOSen ineinander überführt werden können. Wir definieren dazu die generische affine Transformation

$$\mathbf{x}_I = {}^{I\leftarrow J}\mathbf{R}\mathbf{x}_J + {}^{I\leftarrow J}\mathbf{t}, \tag{3.1}$$

welche die Transformation der geometrischen Sensordaten in Vektorform $\mathbf{x}_J \in \mathbb{R}^{3\times 1}$ von einem Ursprungskoordinatensystem J in ein Zielkoordinatensystem I überführt. Hierbei entsprechen ${}^{I\leftarrow J}\mathbf{R} \in \mathbb{R}^{3\times 3}$ einer Rotationsmatrix und ${}^{I\leftarrow J}\mathbf{t} \in \mathbb{R}^{3\times 1}$ einem Translationsvektor. Die Darstellung der Sensordaten in Form von $\mathbf{x}_J$ werden wir nachfolgend

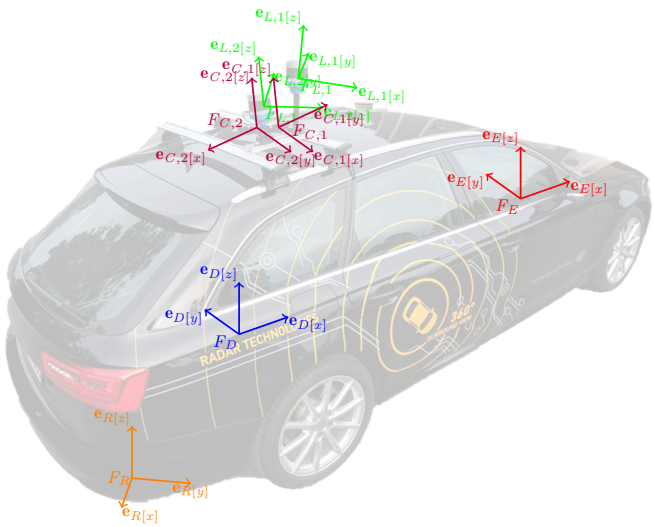


Abbildung 3.1.: **Qualitative Darstellung und Ausrichtung der Koordinatensysteme:** Rot: Fahrzeug-KOOS; Grün: Lidar-KOOS; Blau: DGPS-KOOS; Lila: Kamera-KOOS; Orange: Radar-KOOS. Nach [EB6].

als „intrinsische geometrische Kalibrierung“ bezeichnen. Die Identifikation der Rotationsmatrix und des Translationsvektors wird nachfolgend als „extrinsische geometrische Kalibrierung“ bezeichnet. Die Beschreibung der geometrischen Kalibrierungen der Sensoren ist Bestandteil dieses Kapitels.

Als Synonym des jeweiligen KOOSs definieren wir uns folgende Symbole:

- E: Bezugs KOOS auf dem Ego-Fahrzeug
- D: KOOS des DGPS-INS
- C_1, C_2: KOOSe der zwei Kamerasensoren
- L_1, L_2: KOOSe der zwei Lidarsensoren
- R: KOOS des Radarsensors.

Da die unterschiedlichen Sensoren eigene Zeitbasen haben und die Szene nicht zu gleichen Zeitpunkten abtasten, müssen die zeitlichen Unterschiede in der Abtastung kompensiert werden, um eine synchrone Fusion der Daten zu erreichen. Im Laufe dieses Kapitels wird ebenfalls vorgestellt, wie diese zeitliche Kompensation, auch „temporale Synchronisation“ genannt, durchgeführt wurde.

3.2. Radar

Der hier verwendete Radarsensor ist ein Entwicklungssensor mit dem Antennendesign eines aktuellen Seriensensors, jedoch mit physikalischer Schnittstelle zur Ausgabe von ADU-Daten. Das FoV bzw. die „Strahlbreite“ des Sensors beträgt in Azimut etwa 140° und Elevation ca. 20°.

3.2.1. Intrinsische geometrische Kalibrierung

Wie im Abschnitt 2.1 erwähnt, kann der Radarsensor durch geschickte Signalverarbeitung u.a. die Entfernung, die Geschwindigkeit und den Einfallswinkel der Reflektoren bzw. Reflexionen schätzen. In Gleichung 3.1 ist die Position des Reflektors in kartesischen Koordinaten gefordert. Mit Hilfe von Gleichung 2.5 lässt sich die Position von Polarkoordinaten (Azimut, Elevation, Entfernung) in kartesische Koordinaten überführen

$$\mathbf{x}_R = \begin{bmatrix} \mathbf{e}_{\mathrm{R[x]}} & \mathbf{e}_{\mathrm{R[y]}} & \mathbf{e}_{\mathrm{R[z]}} \end{bmatrix} \cdot \begin{bmatrix} x_{\mathrm{R[x]}} \\ x_{\mathrm{R[y]}} \\ x_{\mathrm{R[z]}} \end{bmatrix} = \begin{bmatrix} \mathbf{e}_{\mathrm{R[x]}} & \mathbf{e}_{\mathrm{R[y]}} & \mathbf{e}_{\mathrm{R[z]}} \end{bmatrix} \cdot \begin{bmatrix} \cos(\phi_{\mathrm{az}})\cos(\phi_{\mathrm{el}}) \\ \sin(\phi_{\mathrm{az}})\cos(\phi_{\mathrm{el}}) \\ \sin(\phi_{\mathrm{el}}) \end{bmatrix} r, \quad (3.2)$$

wobei $\mathbf{e}_{\mathrm{R[x]}}$, $\mathbf{e}_{\mathrm{R[y]}}$ und $\mathbf{e}_{\mathrm{R[z]}}$ die Einheitsvektoren zum KOOS für $\mathbf{x}_R$ sind. Die Schätzung der Einfallswinkel ϕ_{az} und ϕ_{el} können z.B. nach den in Unterabschnitt 2.1.5 vorgestellten Verfahren durchgeführt werden. Bedingt durch Fertigungstoleranzen und Signalbeugung am Stoßfänger ergeben sich Phasenverzerrungen zwischen den Antennen, welche die Modellbeschreibung nach Gleichung 2.37 stören. Eine Kompensation dieser Verzerrung wurde, wie in [Kue17] beschrieben, durch eine Kreuzkalibrierung mit durch Roboter platzierten Referenzzielen durchgeführt. Da diese Kreuzkalibrierung für das Verständnis der Aspekte dieser Arbeit nicht notwendig ist, sei der interessierte Leser auf referenzierte Literatur verwiesen. Ein alternatives Verfahren zur Kreuzkalibrierung zwischen Radar und Lidar ist in [PSK20] zu finden. Ein Verfahren für die Kreuzkalibrierung von Radar und Kamera ist in [CYH+23] zu finden.

3.2.2. Extrinsische geometrische Kalibrierung

Als Zielkoordinatensystem bei der extrinsischen geometrischen Kalibrierung wurde hier das Ego-KOOS gewählt, welches die Pose des Radarsensors relativ zu einem KOOS E im Ego-Fahrzeug beschreibt. Das fortlaufend als „Ego-KOOS“ bezeichnete KOOS, dessen Ausrichtung analog zur ISO-8855 [ISO11, MW04] verläuft, wurde im Zentrum der Fahrzeugvorderachse platziert, siehe Abbildung 3.1. Es wird angenommen, dass sämtliche verwendeten Sensoren fest am Fahrzeug angebracht sind und dass zu keiner Zeit eine Deformation des Fahrzeuges oder der Sensorhalterung bspw. durch dynamische Fahrmanöver entsteht.

Die Identifikation der Rotationsmatrix ${}^{E\leftarrow R}\mathbf{R}$ und des Translationsvektors ${}^{E\leftarrow R}\mathbf{t}$ erfolgte anhand der Computer-Aided-Design (CAD)-Konstruktion. In der CAD-Konstruktion war die Einbauhalterung des Radarsensors bereits konstruiert und somit die Einbaupose festgelegt. Kleinere Abweichungen der Einbauorientierung wurden durch die intrinsische Kalibrierung kompensiert. Die Transformation eines Punktes, gegeben durch den Radarsensor, kann somit folgend in das Ego-KOOS transformiert werden

$$\mathbf{x}_E = {}^{E\leftarrow R}\mathbf{R}\mathbf{x}_R + {}^{E\leftarrow R}\mathbf{t}. \quad (3.3)$$

3.3. Lidar

Das Sensorsetup umfasst zwei Lidarsensoren. Die Kalibrierung beider Sensoren erfolgt analog zueinander. Falls notwendig, werden, wie in Abbildung 3.1 gezeigt, zur Unterscheidung der Lidarsensoren nachgestellte Indizes verwendet, beispielsweise ${}^{E\leftarrow L_1}\mathbf{R}$ für den ersten Lidarsensor.

3.3.1. Intrinsische geometrische Kalibrierung

Analog zum Radarsensor liefert der Lidarsensor Reflexionsparameter in Polarkoordinaten, so dass die Darstellung der Lidar-Daten in kartesischen Koordinaten $\mathbf{x}_L$ analog zur Gleichung 3.2 vorgenommen wird.

3.3.2. Extrinsische geometrische Kalibrierung

Wie beim Radarsensor wurde das Ego-KOOS als Zielsystem der Koordinatentransformation gewählt. Die Identifikation der Transformationsparameter ${}^{E\leftarrow L}\mathbf{R}$ und ${}^{E\leftarrow L}\mathbf{t}$ erfolgt in drei wesentlichen Schritten.

Im ersten Arbeitsschritt wird das Fahrzeug mit den montierten Lidarsensoren auf einer Stellfläche positioniert. Diese Stellfläche sollte möglichst eben sein und in unmittelbarer Nähe zum Fahrzeug (ca. 30 m) keine vertikal ausgedehnten Objekte beinhalten. Nun wird mittels RANSAC-Verfahren [FB81] eine Ebene in der Punktwolke[1] detektiert. Dabei wird iterativ (100 Durchgänge) eine Untermenge (3 Punkte) der Punktwolke gezogen und eine Ebene durch die Untermenge der Punkte gepasst. Danach wird die Anzahl aller Punkte ermittelt, die einen geringen orthogonalen Abstand d zur Ebene aufweisen ($d \leq 0.2\,\mathrm{m}$). Schlussendlich wird die Ebene mit der höchsten Anzahl an Punkten mit geringem Abstand gewählt. Durch die Wahl des Szenarios ist davon auszugehen, dass diese Ebene der Bodenfläche entspricht. Ein Beispiel dieser Detektion ist in Abbildung 3.2 links zu sehen, in der die Bodenpunkte schwarz markiert wurden. Die rot markierten Punkte stellen vertikal ausgedehnte Objekte auf der Stellfläche wie Hauswände, parkende Fahrzeuge und Bäume dar.

Mittels Hauptkomponentenanalyse (engl: Prinicipal-Component-Analysis (PCA)) [F.R01] wird nun der Normalenvektor der detektierten Ebene (in positiver Z-Richtung) berechnet. Dazu fassen wir die kartesischen Koordinaten aller Punkte zur detektierten Ebene in Matrixform zusammen:

$$\mathbf{D}_{\mathrm{surf}} = \begin{bmatrix} \hat{x}_{L,x,1} & \hat{x}_{L,y,1} & \hat{x}_{L,z,1} \\ \vdots & \vdots & \vdots \\ \hat{x}_{L,x,N} & \hat{x}_{L,y,N} & \hat{x}_{L,z,N} \end{bmatrix}. \tag{3.4}$$

Anschließend wird die Kovarianzmatrix $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times 3}$ aus der Punktwolke berechnet und in Eigenwertdarstellung umgewandelt

$$\boldsymbol{\Sigma} = \frac{1}{N}\mathbf{D}_{\mathrm{surf}}^T\mathbf{D}_{\mathrm{surf}} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T. \tag{3.5}$$

$\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$ ist die Eigenwertmatrix und $\boldsymbol{V}$ die Eigenvektormatrix. Die

[1] Menge aller Punkte von x_L

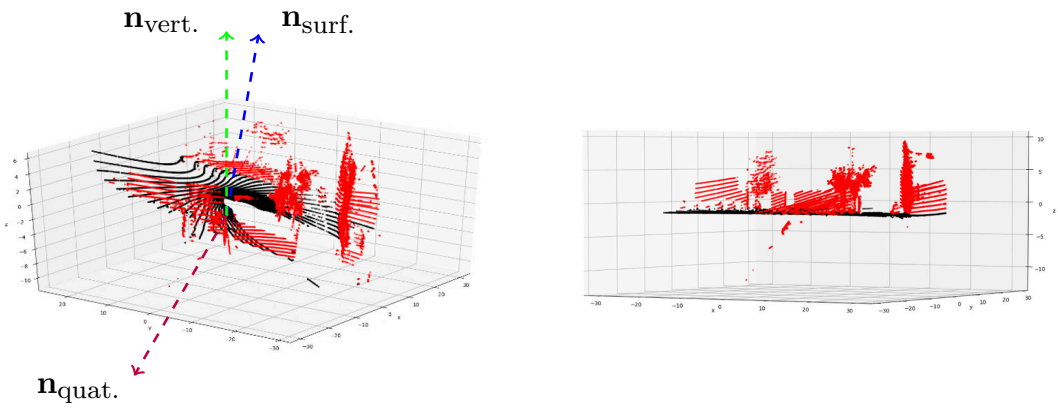


Abbildung 3.2.: **Horizontierung der Bodenreflexionen aus Lidarsensor:** Links: Die schwarzen Punkte markieren die als Boden detektierten Punkte, die roten Punkte entsprechend Ausreißer bei der RANSAC-Schätzung. Ebenfalls eingetragen sind qualitative Darstellungen der identifizierten Oberflächennormalen $\mathbf{n}_{\text{surf.}}$, der Zielausrichtung der Oberflächennormalen $\mathbf{n}_{\text{vert.}}$ und der Drehachse $\mathbf{n}_{\text{quat.}}$. Rechts: Darstellung der Bodenreflexionen (schwarz) und Ausreißer (rot) nach der Rotation.

Eigenvektormatrix enthält drei orthogonal zueinander stehende Vektoren, welche die Hauptkomponenten der Punktwolke darstellen. Durch die vorausgegangene Detektion von Aureißern orthogonal zur Ebene, kann angenommen werden, dass die Punkte in der Ebene deutlich stärker streuen als orthogonal zur Ebene, so dass die Hauptkomponente mit kleinstem Eigenwert eine Schätzung der Normalen zur Ebene ist

$$\mathbf{n}_{\text{surf.}} = \boldsymbol{V}[:, \text{argmin}(\lambda_i)]. \tag{3.6}$$

Wie in Unterabschnitt 3.2.2 beschrieben, sei das Ego-KOOS definiert als kartesisches KOOS in welchem die z-Achse orthogonal zur Fahrbahnoberfläche verläuft. Daraus ergibt sich der Zielvektor der Z-Achse der Ebenen zu $\mathbf{n}_{\text{vert.}} = \begin{bmatrix}0, 0, 1\end{bmatrix}^T$. Es wird eine Rotationsmatrix $R_{\text{surf.}}$ gesucht, welche $\mathbf{n}_{\text{surf.}}$ in den Zielvektor $\mathbf{n}_{\text{vert.}}$ dreht. Aus dem Zielvektor $\mathbf{n}_{\text{vert.}}$ und dem Normalenvektor $\mathbf{n}_{\text{surf.}}$ wird zunächst das Einheitquaternion $\mathbf{q}$ gebildet

$$\mathbf{q} = \begin{bmatrix}\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{q}_4\end{bmatrix}^T = \begin{bmatrix}\cos\left(\frac{\gamma_1}{2}\right), \frac{\sin\left(\gamma_1/2\right)\mathbf{n}_{\text{quat.}}}{|\mathbf{n}_{\text{quat.}}|}\end{bmatrix} \tag{3.7}$$

mit

$$\gamma_1 = \arccos\left(\frac{\mathbf{n}_{\text{surf.}} \cdot \mathbf{n}_{\text{vert.}}}{\|\mathbf{n}_{\text{surf.}}\|\|\mathbf{n}_{\text{vert.}}\|}\right) \tag{3.8}$$

und

$$\mathbf{n}_{\text{quat.}} = \mathbf{n}_{\text{surf.}} \times \mathbf{n}_{\text{vert.}}. \tag{3.9}$$

Anschaulich betrachtet entspricht $\mathbf{n}_{\text{quat.}}$ der Rotationsachse, um welche mit dem Winkel γ_1 gedreht werden muss, damit die Vektoren $\mathbf{n}_{\text{vert.}}$ und $\mathbf{n}_{\text{surf.}}$ kongruent zueinander sind. Diese Rotation kann nach [KUI99] auch in Form einer Rotationsmatrix

$\mathbf{R}_{\text{surf.}} \in \mathbb{R}^{3\times3}$ dargestellt werden, zu

$$\mathbf{R}_{\text{surf.}} = \begin{bmatrix} 1-2(\mathbf{q}_3^2+\mathbf{q}_4^2) & 2(\mathbf{q}_2\mathbf{q}_3-\mathbf{q}_4\mathbf{q}_1) & 2(\mathbf{q}_2\mathbf{q}_4+\mathbf{q}_3\mathbf{q}_1) \\ 2(\mathbf{q}_2\mathbf{q}_3+\mathbf{q}_4\mathbf{q}_1) & 1-2(\mathbf{q}_2^2+\mathbf{q}_4^2) & 2(\mathbf{q}_3\mathbf{q}_4-\mathbf{q}_2\mathbf{q}_1) \\ 2(\mathbf{q}_2\mathbf{q}_4-\mathbf{q}_3\mathbf{q}_1) & 2(\mathbf{q}_3\mathbf{q}_4+\mathbf{q}_2\mathbf{q}_1) & 1-2(\mathbf{q}_2^2+\mathbf{q}_3^2) \end{bmatrix}. \tag{3.10}$$

Durch Drehung der Punktwolke ($\mathbf{R}_{\text{surf.}}\mathbf{D}_{\text{surf}}^T$) ergibt sich eine Punktwolke, in welcher die detektierte Fahrbahn orthogonal zur z-Achse verläuft, siehe Abbildung 3.2 rechts. Da durch die PCA nicht die Ober- oder Unterseite der Fahrbahn erkannt wird, muss $\mathbf{n}_{\text{surf.}}$ gegebenenfalls manuell umgekehrt werden.

Die vorausgegangene Rotation zielt darauf ab, die Punktwolke des Lidars parallel zur Ebene auszurichten. Die Punktwolke kann jedoch noch um die z-Achse des Fahrzeuges verdreht sein. Um diese Verdrehung zu identifizieren, werden sämtliche Punkte oberhalb der Fahreben entsprechend ihrer Position in der Ebene in ein Bild eingetragen, wobei der Grauwert durch die Reflexionsintensität des Punktes bestimmt wird. Nun wird das Fahrzeug geradeaus mit etwa $20\,\mathrm{km\,h^{-1}}$ bewegt. Das Bild wird für jeden Frame neu gerendert. Der Bildinhalt bewegt sich nun ebenfalls geradlinig entsprechend der verbliebenen Verdrehung. Um diese geradlinige Bewegung zu identifizieren, werden mittels Shi-Tomasi [ST94] Detektor wichtige Punkte (engl: „keystones“) des Grauwertbildes detektiert und mit Lucas-Kanade-Methode [LK81] zeitlich verfolgt, siehe Abbildung 3.3.

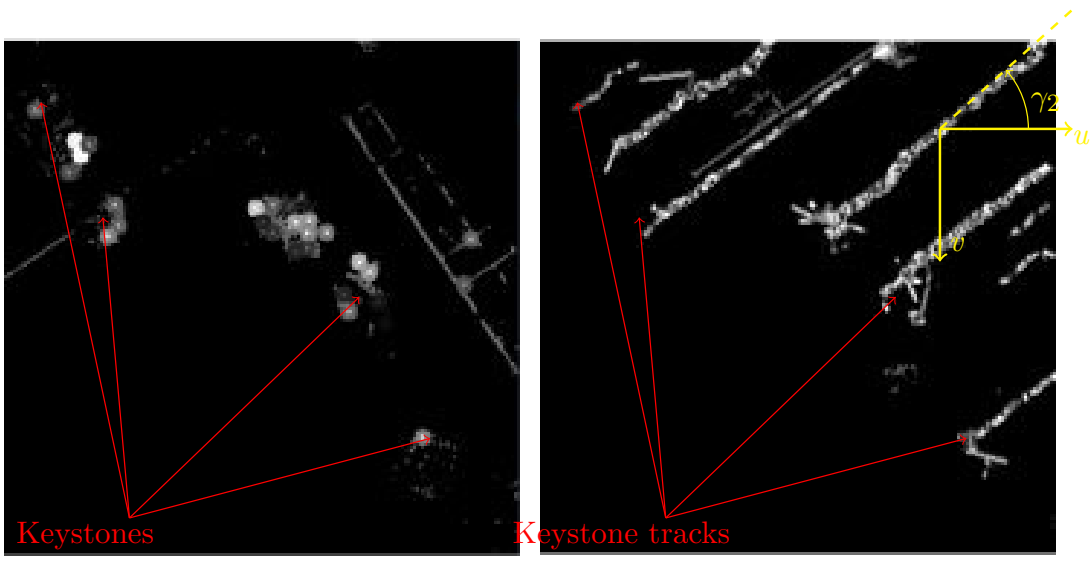


Abbildung 3.3.: **Verfolgung wichtiger keystones in der Lidar-Punktwolke:** Links: Darstellung der nicht in der Fahrbahnebene liegenden Punktwolke sowie der detektierten keystones im Grauwertbild. Rechts: Tracks der keystones nach Geradeausfahrt des Fahrzeuges. Die Tracks verlaufen im Winkel γ_2 zur Bildhorizontalen.

Aus der Gesamtheit aller verfolgten Punkte (engl: „Tracks“) wird die mittlere Orientierung γ_2 gegenüber der vertikalen Bildachse geschätzt. Dabei wurde festgelegt, dass die vertikale Bildachse der Längsachse des Fahrzeuges entsprechen soll und die Punkte somit

bei sich vorwärts bewegendem Fahrzeug vom oberen Bildbereich nach unten wandern sollen. Für die Schätzung von γ_2 kommt wieder ein RANSAC-Verfahren zum Einsatz, um Ausreißer in den Tracks zu detektieren und die Schätzung von γ_2 zu stabilisieren. Die sich daraus ergebene Rotationsmatrix stellt eine Drehung um die z-Achse mit dem Drehwinkel γ_2 dar

$$\mathbf{R}_{\text{keystones}} \begin{bmatrix} \cos(90° - \gamma_2) & \sin(90° - \gamma_2) & 0 \\ -\sin(90° - \gamma_2) & \cos(90° - \gamma_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.11}$$

Die gesamte Rotation des Lidars gegenüber dem Ego-Fahrzeug wird somit zu

$$^{E\leftarrow L}\mathbf{R} = \mathbf{R}_{\text{keystones}}\mathbf{R}_{\text{surf.}} \tag{3.12}$$

identifiziert.

Neben der Rotation wird die Translation $^{E\leftarrow L}\mathbf{t}$ von Lidar und Ego-Koordinaten befüllt

$$^{E\leftarrow L}\mathbf{t} = \begin{bmatrix} x_{\text{Lidar}} & y_{\text{Lidar}} & z_{\text{Lidar}} \end{bmatrix}^T. \tag{3.13}$$

Die Komponente z_{Lidar} lässt sich aus der mittleren vertikalen Höhe aller Punkte der Ebene berechnen, über

$$\begin{bmatrix} x_{1,\text{derot.}} \dots x_{N,\text{derot.}} \\ y_{1,\text{derot.}} \dots y_{N,\text{derot.}} \\ z_{1,\text{derot.}} \dots z_{N,\text{derot.}} \end{bmatrix} = {}^{E\leftarrow L}\mathbf{R}\mathbf{D}_{\text{surf}}^T, \tag{3.14}$$

zu

$$z_{\text{Lidar}} = \frac{1}{N}\sum_{i}^{N} z_{i,\text{derot.}}. \tag{3.15}$$

Die Komponenten x_{Lidar} und y_{Lidar} werden mittels Maßband manuell ausgemessen. x_{Lidar} ist dabei der Abstand der Fahrzeugvorderachse und y_{Lidar} der Abstand von der Fahrzeuglängsachse zum Lidarsensor.

Die Algorithmen können auf einem im Fahrzeug verbautem Personal-Computer (PC) in Echtzeit prozessiert werden und die gesamte Kalibrierung der Lidarsensoren somit innerhalb von etwa 5 min durchgeführt werden.

Die Transformation von Positionen des ersten Lidar KOOS konnte somit wie folgt in das Ego-KOOS vorgenommen werden

$$\mathbf{x}_E = {}^{E\leftarrow L_1}\mathbf{R}\mathbf{x}_{L_1} + {}^{E\leftarrow L_1}\mathbf{t}. \tag{3.16}$$

Für den zweiten Lidarsensor ergibt sich analog eine Transformation zu

$$\mathbf{x}_E = {}^{E\leftarrow L_2}\mathbf{R}\mathbf{x}_{L_2} + {}^{E\leftarrow L_2}\mathbf{t}. \tag{3.17}$$

3.4. Kamera

Die verwendeten Kameras haben einen horizontalen Öffnungswinkel von etwa 105° und einen vertikalen Öffnungswinkel von etwa 80°. Da das FoV einer einzelnen Kamera

nicht ausreicht, um das FoV des verwendeten Radarsensors abzudecken (vgl.: 140°), wurden zwei Kameras an ähnlicher Position, aber mit unterschiedlicher Ausrichtung am Fahrzeug montiert. Durch Fusion beider Kamerabilder oder separate Auswertung der Kamerabilder lässt sich später die Signalverarbeitung auf Radardaten über dessen gesamtes FoV auswerten.

3.4.1. Intrinsische geometrische Kalibrierung

Für die Darstellung von Bildpunkten in der Kamera hin zu kartesischen Koordinaten für eine Transformation gemäß Gleichung 3.1 werden zwei Arbeitsschritte durchgeführt. Dies ist erstens die Rektifizierung des Kamerabildes und zweitens die Beschreibung der kartesischen Koordinaten über ein Lochkameramodell.

3.4.1.1. Rektifizierung des Kamerabildes

Bei der verwendeten Kamera wurde eine positive radiale optische Verzeichnung festgestellt. Dabei handelt es sich um geometrische Abbildungsfehler, welche zu lokalen Verzerrungen des Bildinhaltes führen. Dieser Effekt ist in Abbildung 3.4 links zu sehen. Die gerade verlaufenden Längsbalken und Querrohre an der Decke haben eine ausgeprägte Krümmung im Kamerabild.

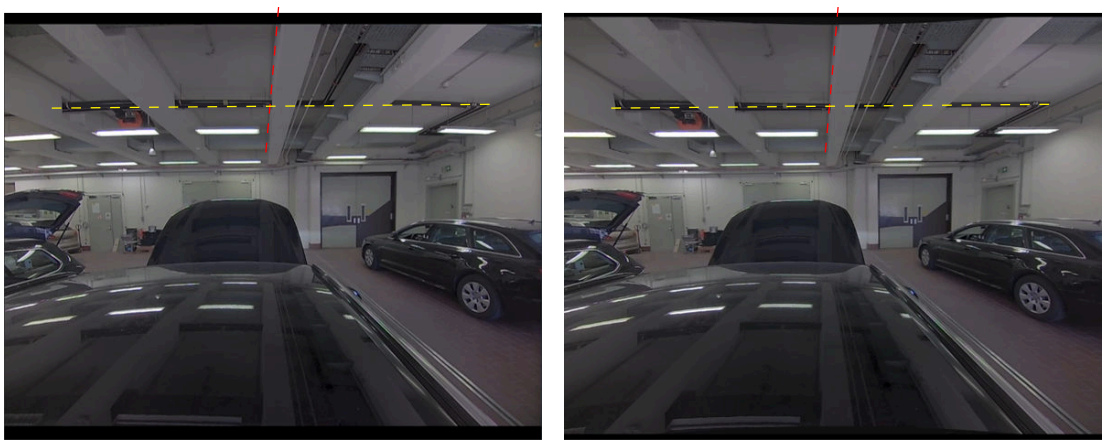


Abbildung 3.4.: **Rektifizierung des Kamerabildes:** Links: unbearbeitetes Bild $\mathbf{RGB}_{\text{dist.}}$. Rechts: Bild nach Rektifizierung $\mathbf{RGB}_{\text{rect.}}$. Die Verzerrung durch die Kameralinse führt zu einer Verzerrung gerader Objekte im Kamerabild, bspw. den Längsträgern an der Decke. Nach der Rektifizierung liegt der Längsträger auch im Bild in einer Flucht, mit der eingezeichneten gelben Linie.

Ursächlich für die optische Verzeichnung ist häufig die Linsengeometrie. In der Regel steigt die optische Verzerrung bei Linsen mit größerem Öffnungswinkel. Die hier verwendete Linse ermöglicht einen verhältnismäßig großen Öffnungswinkel von etwa 105° und eine Entzerrung bzw. Rektifizierung des Bildes ist notwendig, um im weiteren Verlauf eine bestmögliche Projektion von Lidarpunkten zu ermöglichen. Zur Kompensation dieser optischen Verzeichnung bietet die verwendete Kamera eine interne digitale Entzerrung an, deren Ergebnis in Abbildung 3.4 rechts dargestellt ist. Dabei werden die

Bildkoordinaten im Kamerabild nach dem parametrischen Modell aus [Bro66, Zhe99] manipuliert. Die Bildkoordinaten in der verzerrten Optik lassen sich dabei modellieren, als

$$x = \frac{u - c_u}{f_u} \tag{3.18}$$

$$y = \frac{v - c_v}{f_v} \tag{3.19}$$

$$u_{\text{dist.}} = f_u x(1 + \kappa_1 r^2 + \kappa_2 r^4) + c_u \tag{3.20}$$

$$v_{\text{dist.}} = f_v y(1 + \kappa_1 r^2 + \kappa_2 r^4) + c_v \tag{3.21}$$

$$r = \sqrt{x^2 + y^2}, \tag{3.22}$$

wobei u und v die nicht verzerrten Pixel Koordinaten sind. Die Verzerrungsfaktoren $\kappa_1 = -0.11$ und $\kappa_2 = 0.085$ wurden bei einer Identifikation der intrinsischen Kameraparameter geschätzt. c_u, c_v, f_u und f_v werden im nächsten Abschnitt benannt und identifiziert.

Das rektifizierte Kamerabild wird durch den in Algorithmus 1 dargestellten Pseudocode berechnet.

Algorithmus 1: Bild-Rektifizierung

Daten: Verzerrtes Kamerabild $\mathbf{RGB}_{\text{dist.}}$ und intrinsische Kameraparameter $\{f_x, f_x, c_x, c_y, \kappa_1, \kappa_2\}$
Ergebnis: Rektifiziertes Kamerabild $\mathbf{RGB}_{\text{rect.}}$
für $u \leftarrow 0$ **bis** $640 - 1$ **tue**
  für $v \leftarrow 0$ **bis** $480 - 1$ **tue**
    $x \leftarrow (u - c_u)/f_u$
    $y \leftarrow (v - c_v)/f_v$
    $r \leftarrow \sqrt{x^2 + y^2}$
    $K \leftarrow 1 + \kappa_1 r^2 + \kappa_2 r^4$
    $u_{\text{dist.}} \leftarrow \text{round}(f_x K x + c_x)$
    $v_{\text{dist.}} \leftarrow \text{round}(f_y K y + c_y)$
    $\mathbf{RGB}_{\text{rect.}[u,v]} \leftarrow \mathbf{RGB}_{\text{dist.}[u_{\text{dist.}}, v_{\text{dist.}}]}$
return $\mathbf{RGB}_{\text{rect.}}$

Der interessierte Leser sei an dieser Stelle auf weiterführende Literatur zur optischen Verzeichnung verwiesen, bspw. [MMT10, Sze10].

3.4.1.2. Beschreibung der Bildkoordinaten über Lochkameramodel

Nach der erfolgten Rektifizierung des Kamerabildes lässt sich die optische Abbildung durch die Kamera mit guter Näherung durch ein einfaches Lochkameramodell (engl.: camera pinhole model) [FP12] beschreiben[2]. Eine Darstellung des Modells ist in Abbildung 3.5 gegeben.

[2] Äquivalent kann auf die Rektifizierung des Kamerabildes verzichtet werden, wenn im Gegenzug ein entsprechend komplexeres Kameramodell verwendet wird.

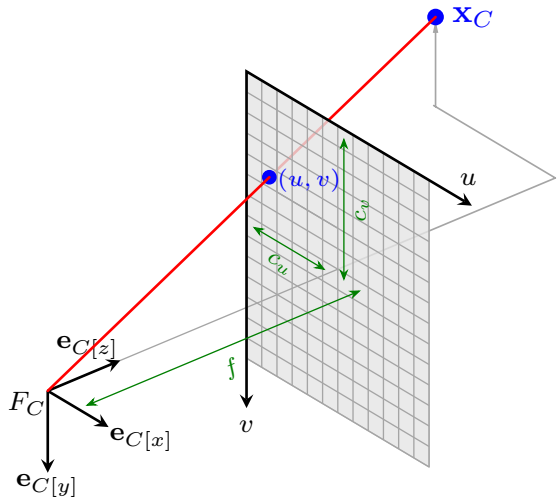


Abbildung 3.5.: **Lochkameramodel:** Projektion eines Punktes aus Kamerakoordinaten in Bildkoordinaten.

Die graue Fläche stellt das Kamerabild dar. Das Koordinatenkreuz F_C die Position und Orientierung der Kamera. Ein Punkt P gegeben in Kamerakoordinaten, durchstößt das Kamerabild an den Bildkoordinaten u und v und wird an dieser Position im Kamerabild dargestellt. Durch Anwenden des Strahlensatzes ergeben sich die folgenden geometrischen Beziehungen

$$\frac{u - c_u}{f_u} = \frac{\mathbf{x}_{C[x]}}{\mathbf{x}_{C[z]}} \tag{3.23a}$$

$$\frac{v - c_v}{f_v} = \frac{\mathbf{x}_{C[y]}}{\mathbf{x}_{C[z]}}. \tag{3.24a}$$

Die intrinsischen Kameraparameter f_u, f_v, c_u, c_v beschreiben die Brennweiten bzw. den Bildversatz und werden in der Einheit Pixel angegeben. Ein Punkt $\mathbf{x}_C \in \mathbb{R}^{(3\times1)}$ wird durch seine Koordinaten, die tiefgestellte eckige Klammer kennzeichnet die jeweilige Komponente, z.B. $x_{C[y]}$, dargestellt.

Durch Umstellen der Gleichungen nach Kamerakoordinaten ergeben sich die kartesischen Koordinaten im Kamera-KOOS aus den Bildkoordinaten und der Tiefe $\mathbf{x}_{C[z]}$ zu

$$\mathbf{x}_C = \begin{bmatrix} x_{C[x]} \\ x_{C[y]} \\ x_{C[z]} \end{bmatrix} = \begin{bmatrix} \frac{u-c_u}{f_u} x_{C[z]} \\ \frac{v-c_v}{f_v} x_{C[z]} \\ x_{C[z]} \end{bmatrix}. \tag{3.25}$$

Zur Identifikation der intrinsischen Kameraparameter wurde die MATLAB-Toolbox [Mat17] verwendet und Kamerabilder einer Szene mit Kalibriermustern aufgezeichnet, siehe Abbildung 3.6. Als Kalibriermuster wurden Schachbrettmuster mit einer Seitenlänge von hier 45 mm auf Papier ausgedruckt und zur Stabilisierung auf Pappe verklebt.

In der MATLAB-Toolbox werden die Eckpunkte sämtlicher Felder des Schachbrettmusters detektiert. Durch Variation der Position der Kalibriermuster im Kamerabild

Abbildung 3.6.: **Kalibriermuster für Identifikation der intrinsischen Kamera Parameter:** Darstellung der verwendeten Kalibriermuster für die intrinsische Kalibrierung der Kameras.

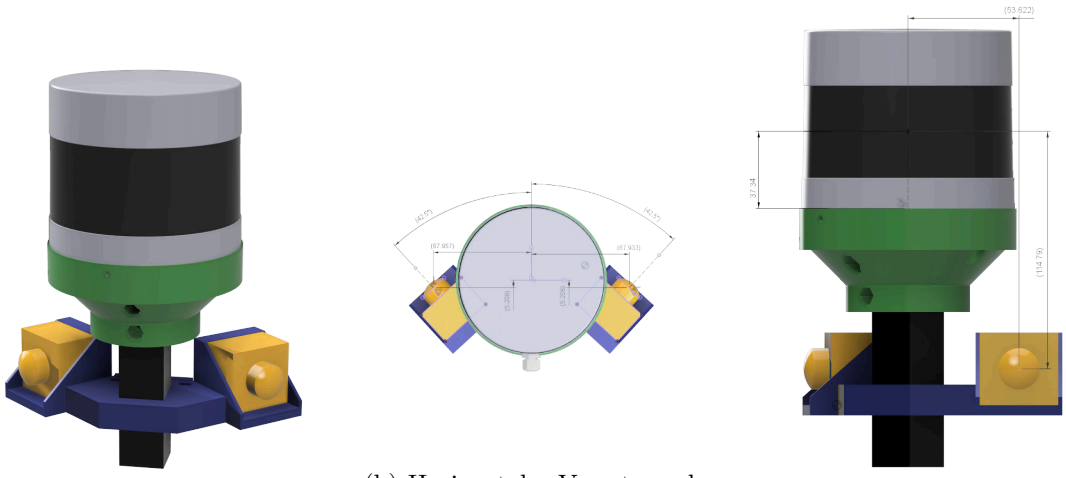
werden die intrinsischen Parameter (Brennweite und optische Achse) des Lochkameramodells automatisch identifiziert. Bei den verwendeten Kameras wurden die Brennweiten $f_u = f_v \approx 510\text{px}$ identifiziert. Die optische Achse $c_u = \frac{640\text{px}}{2} = 320\text{px}$ und $c_v = \frac{480\text{px}}{2} = 240\text{px}$ wurde auf den Bildmittelpunkt zentriert.

3.4.2. Extrinsische geometrische Kalibrierung

Um die Beobachtungsposition von Kamera und Lidar möglichst gleich zu halten und somit Abbildungsunterschiede der Umgebung gering zu halten, wurden die Kameras direkt in der Nähe der Lidarsensoren verbaut, siehe Abbildung 3.7a. Ein weiterer Vorteil ist, dass die extrinsischen geometrischen Kalibrierparameter in Form von Versatz und Verdrehung der Kameras gegenüber den Lidarsensoren aus der CAD-Konstruktion entnommen werden können, siehe Abbildung 3.7b und 3.7c.

Die Fertigung der Halter für Lidar und Kameras erfolgte mittels FDM-3D-Druckverfahren in PETG. Die Halter wurden an einem Vierkantprofil aus Stahl verschraubt. Am Fuß des Vierkantprofils wurden Winkelplatten montiert, welche wiederum über Aluprofile am Dachgepäckträger des Fahrzeugs montiert wurden. Eine Nahaufnahme der Halterung ist in Abbildung 3.8 zu sehen.

Über die Winkelplatten konnte die Verkippung der Lidarsensoren gegenüber dem Fahrzeug eingestellt werden. Die Lidarsensoren wurden manuell so eingestellt, dass sich eine subjektiv homogene und dichte Verteilung der Lidarpings in den Kamerabildern ergab. Die gesamte Konstruktion wurde als ausreichend stabil empfunden und eine Deformation während der Aufnahme des Datensatzes wurde nicht erwartet.



(a) Vorderansicht der Konstruktion

(b) Horizontaler Versatz und Verdrehung der Kameras ggü. Lidar

(c) Vertikaler Versatz der Kameras ggü. Lidar

Abbildung 3.7.: **CAD-Modell der Kamera- und Lidar-Halterung:** Lidar und Kameras (gelb dargestellt) werden über Halter (grün bzw. blau dargestellt) an einem Vierkantprofil (schwarz dargestellt) verschraubt.

Wie zuvor beschrieben, ergibt sich aus der CAD-Konstruktion die extrinsische Kalibrierung der Kamera gegenüber dem Lidarsensor, analog zu Gleichung 3.1 zu

$$\mathbf{x}_L = {}^{L\leftarrow C}\mathbf{R}\mathbf{x}_C + {}^{L\leftarrow C}\mathbf{t}. \tag{3.26}$$

Hierbei ist ${}^{L\leftarrow C}\mathbf{R}$ die Rotationsmatrix vom Kamera in das Lidar KOOS und ${}^{L\leftarrow C}\mathbf{t}$ die entsprechende Translation. Durch Umstellen der Gleichung nach Kamerakoordinaten und mit Hilfe von Gleichung 3.25 können Punktmessungen des Lidars in das Kamerabild transformiert werden. Durch Fertigungs- und Verbautoleranzen sind die Parameter der CAD-Konstruktion nur Näherungen für die extrinsische Kalibrierung. Bei der Projektion der Lidar-Punkte in das Kamerabild konnten marginale Abweichungen zwischen Objektkonturen und Lidar-Punkten festgestellt werden. Zur Verbesserung der Kalibrierung wurde eine manuelle Kreuzkalibrierung zwischen Lidar und Kamera vorgenommen.

Diese Kreuzkalibrierung der extrinsischen Verdrehparameter erfolgte durch Vergleich der projizierten Lidarpunkte im Kamerabild gegenüber dem Bildinhalt. Die Rotationsmatrix wurde folgend modifiziert

$${}^{L\leftarrow C}\mathbf{R} = \mathbf{R}_{\text{manual}}\mathbf{R}_{CAD} \tag{3.27}$$

$${}^{L\leftarrow C}\mathbf{t} = \begin{bmatrix} \Delta X_{\text{CAD, C}} \\ \Delta Y_{\text{CAD, C}} \\ \Delta Z_{\text{CAD, C}} \end{bmatrix} \tag{3.28}$$

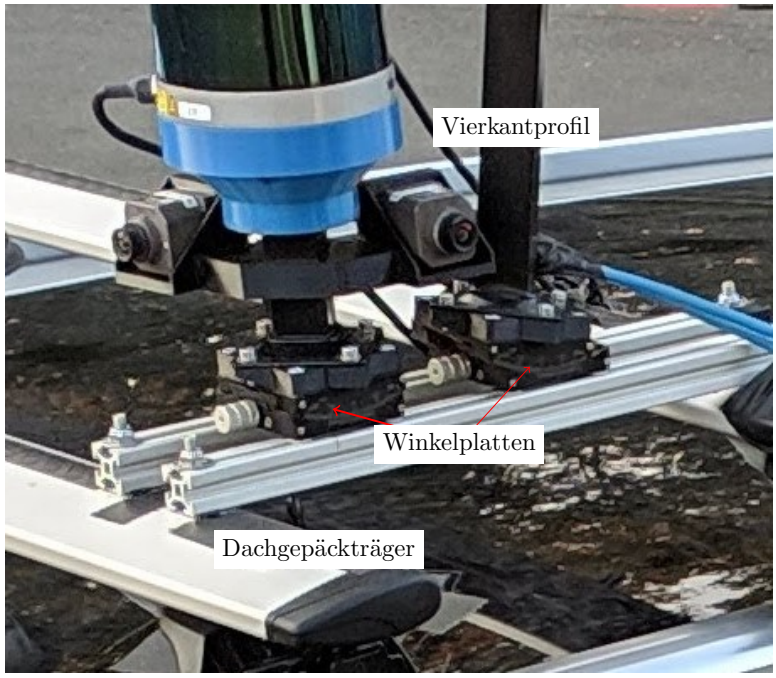


Abbildung 3.8.: **Nahaufnahme der Kamera und Lidar-Halterung:** Die Kameras sind fest am Ständer des hinteren Lidarsensors montiert. Die Lidarsensoren sind verstellbar über Winkelplatten am Fahrzeug montiert.

wobei

$$\mathbf{R}_{CAD} = \begin{bmatrix} \cos\left(\phi_{\text{CAD, C}}\right) & -\sin\left(\phi_{\text{CAD, C}}\right) & 0 \\ \sin\left(\phi_{\text{CAD, C}}\right) & \cos\left(\phi_{\text{CAD, C}}\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.29}$$

die ursprüngliche, aus der CAD-Konstruktion entnommene Transformationsmatrix darstellt. Diese Parameter der Matrix wurden wie folgt aus der CAD Konstruktion abgelesen: Euler-Rotation mit Winkel $\phi_{\text{CAD, C}}$ um die z-Achse und Translation um $\Delta X_{\text{CAD, C}}$, $\Delta Y_{\text{CAD, C}}$ und $\Delta Z_{\text{CAD, C}}$. Für die linke Kamera in Abbildung 3.7b ergaben sich somit beispielhafte Transformationsparameter von $\phi_{\text{CAD, C}} = 42.5°$ und $\Delta X_{\text{CAD, C}} = 0.52\,\text{cm}$, $\Delta Y_{\text{CAD, C}} = -6.8\,\text{cm}$ und $\Delta Z_{\text{CAD, C}} = 11.48\,\text{cm}$.

$\mathbf{R}_{\text{manual}} \in \mathbb{R}^{3\times3}$ ist eine Rotationsmatrix. Anhand statischer Strukturen wie Zäunen oder Laternenmasten wurde die Verdrehung $\mathbf{R}_{\text{manual}}$ manuell so weit modifiziert, bis eine gute Kongruenz von Bildinhalt und Lidarpunkten wahrgenommen wurde. Durch die groben Kalibrierparameter aus der CAD-Konstruktion konnte die feine Kalibrierung im Bildkoordinatensystem erfolgen, was die manuelle Identifikation erheblich vereinfacht hat, da (a) die zusätzlichen Verdrehparameter betragsmäßig $< 2°$ lagen und die 3D-Drehung somit nahezu entkoppelt[3] wird und (b) durch die Identifikation im Bildkoordinatensystem die Drehungen um die Bildachsen leicht zu interpretieren sind. Eine manuelle Nachkalibrierung des Kameraversatzes ist nicht erfolgt, da hier eine geringe

[3]Die Reihenfolge bei sequentieller Drehung um x,y und z ist bei geringen Winkeln nahezu egal. Dies vereinfacht die manuelle Identifikation erheblich.

Abweichung erwartet wurde und somit der Einfluss auf die Projektion vernachlässigbar ist.

3.5. DGPS-INS

3.5.1. Intrinsische geometrische Kalibrierung

Das DGPS-INS ist mit einer Doppelbandantenne mit dynamischen Korrekturdaten über Mobilfunk ausgestattet, womit sich eine präzise Position in Weltkoordinaten messen lässt. Zusätzlich ist das verwendete System mit einer Kreiselplattform ausgestattet, welche die Dreh- und Längsbeschleunigung des Fahrzeuges ermitteln kann. In der Sensorik werden diese Messwerte miteinander fusioniert und erlauben das Schätzen weiterer Zustandsgrößen wie z.B. der Geschwindigkeit. Für die Fusion wurde dem System eine per Hand gemessene Einbauposition von Antenne und Kreiselplattform mitgeteilt. Bei Fahrtbeginn wurde ein nach Herstellerangaben vorgegebenes Fahrtszenario durchfahren, in welchem sich das System selbstständig kalibriert.

3.5.2. Extrinsische geometrische Kalibrierung

Die Kreiselplatform wurde möglichst dicht an der Fahrzeughinterachse montiert und die DGPS-Antenne auf dem Fahrzeugdach, so dass eine Störung von Lidar und Kamera möglichst ausgeschlossen werden konnte. Die Ausgabe der DGPS-INS Daten konnte auf einen beliebigen Fahrzeugbezugspunkt parametriert werden. Als Bezugspunkt wurde die Fahrzeughinterachse gewählt, weil diese sich als besonders günstig für die Berechnung von Fahrzeugdynamikparametern erweist [MW04]. Die Transformation von Positionen aus DGPS-INS in das Ego-KOOS wird folgend ermittelt

$$\mathbf{x}_{\mathrm{E}} = {}^{E\leftarrow D}\mathbf{R}\mathbf{x}_D + {}^{E\leftarrow D}\mathbf{t}, \tag{3.30}$$

wobei ${}^{E\leftarrow D}\mathbf{R} \in \mathbb{R}^{3\times 3}$ eine Einheitsmatrix ist und ${}^{E\leftarrow D}\mathbf{t}_{\mathrm{D}} = \begin{bmatrix} -2.871\,\mathrm{m} & 0\,\mathrm{m} & 0\,\mathrm{m} \end{bmatrix}^T$.

3.6. Weitere Koordinatentransformationen

Die Transformationen für Lidarsensoren, DGPS-INS und Radarsensoren sind bereits im fahrzeugfesten Ego-KOOS gegeben. Die Transformation der Kameradaten dagegen nur bezogen auf den tragenden Lidarsensor. Sollen die Positionen aus der Kamera $\mathbf{x}_c$ ebenfalls im Ego-KOOS dargestellt werden, so wird eine sequentielle Transformation zunächst nach Gleichung 3.26 und danach nach Gleichung 3.16 durchgeführt. Die resultierende Transformation entspricht

$$\begin{aligned} \mathbf{x}_E &= {}^{E\leftarrow L_1}\mathbf{R}^{L_1\leftarrow C_1}\mathbf{R}\mathbf{x}_{C_1} + {}^{E\leftarrow L_1}\mathbf{R}^{L_1\leftarrow C_1}\mathbf{t} + {}^{E\leftarrow L_1}\mathbf{t} \\ &= {}^{E\leftarrow C_1}\mathbf{R}\mathbf{x}_{C_1} + {}^{E\leftarrow C_1}\mathbf{t}. \end{aligned} \tag{3.31}$$

Soll dagegen die Darstellung von Punkten gegeben im Ego-KOOS im Kamera KOOS gegeben werden, so ist Gleichung 3.31 entsprechend nach $\mathbf{x}_{C_1}$ umzustellen

$$\mathbf{x}_{C_1} = \left({}^{E\leftarrow C_1}\mathbf{R}\right)^{-1}\left(\mathbf{x}_E - {}^{E\leftarrow C_1}\mathbf{t}\right). \tag{3.32}$$

Sollen Daten aus dem Ego-Koordinatensystem in das Radarkoordinatensystem transformiert werden, so ist dazu Gleichung 3.3 nach $\mathbf{x}_R$ umzustellen und Gleichung 3.31 einzusetzen

$$\begin{aligned}\mathbf{x}_R &= \left({}^{E\leftarrow R}\mathbf{R}\right)^{-1}\left(\mathbf{x}_E - {}^{E\leftarrow R}\mathbf{t}\right)\\ &= \left({}^{E\leftarrow R}\mathbf{R}\right)^{-1}\left({}^{E\leftarrow C}\mathbf{R}\mathbf{x}_C + {}^{E\leftarrow C}\mathbf{t} - {}^{E\leftarrow R}\mathbf{t}\right)\\ &= {}^{R\leftarrow C}\mathbf{R}\mathbf{x}_C + {}^{R\leftarrow C}\mathbf{t}.\end{aligned} \tag{3.33}$$

Hierbei sind

$${}^{R\leftarrow C}\mathbf{R} = \left({}^{E\leftarrow R}\mathbf{R}\right)^{-1}{}^{E\leftarrow C}\mathbf{R} \tag{3.34}$$

und

$${}^{R\leftarrow C}\mathbf{t} = \left({}^{E\leftarrow R}\mathbf{R}\right)^{-1}\left({}^{E\leftarrow C}\mathbf{t} - {}^{E\leftarrow R}\mathbf{t}\right). \tag{3.35}$$

Analog können die Transformationen der Daten aus anderen Sensoren erreicht werden.

3.7. Temporale Kalibrierung

Die Abtastung der Umgebung sämtlicher Sensoren erfolgt zu diskreten Zeitpunkten. Die Abtastzeitpunkte sind für die unterschiedlichen Sensoren nicht einheitlich. Die Aufgabe der temporalen Kalibrierung ist, die Abtastzeitpunkte der Sensoren zu vereinheitlichen, so dass die Sensoren eine Abtastung der Szene zu identischen Zeitpunkten vornehmen. So wird sichergestellt, dass der Szeneninhalt zwischen den Aufnahmen der Sensoren keine zeitliche Variation aufweist und der Szeneninhalt kongruent zueinander ist. Die verwendeten Sensoren erlauben jedoch keine externe Auslösung der Aufnahme, so dass keine einheitliche Auslösung während der Datenakquise erreicht werden kann. Zusätzlich ergibt sich aufgrund unterschiedlicher Abtastraten eine zeitveränderliche Latenz zwischen den Aufnahmezeitpunkten der Sensoren. Neben der Auslösung der Aufnahme unterscheidet sich weiter die Abtastrate der unterschiedlichen Sensoren.

Nachfolgend wird vorgestellt, wie Unterschiede im Abtastzeitpunkt synthetisch nach Datenakquise korrigiert bzw. kompensiert werden können.

3.7.1. Assoziation der Abtastung

Um eine zeitliche Latenz der Aufnahme zwischen den Sensoren auszugleichen, wird zunächst eine Modellbeschreibung der Aufnahmen erstellt. Der Radarsensor tastet mit einer Frequenz von 20 Hz das angegebene FoV ab. Das gesamte FoV des Radars wird nahezu simultan abgetastet, so dass sich die Abtastung als „global-shutter“[4] beschreiben lässt. Die Kameras haben eine eingestellte Abtastfrequenz von 30 Hz. Auch hier erfolgt nahezu eine simultane Abtastung des FoV. Der Lidar ist ein mechanischer Scanner, welcher mit einer Frequenz von 10 Hz um seine Achse rotiert. Abgetastet wird dabei immer ein kleiner Winkelbereich, zu welchem der Scanner gerade ausgerichtet ist. Das FoV des Lidars wird rollend abgetastet. Man spricht dann vom „rolling-shutter“.

In Abbildung 3.9 ist ein beispielhaftes Abtastdiagramm der Sensoren dargestellt. In der oberen Zeile sind die Abtastzeitpunkte des Radarsensors dargestellt. Die Abtastung

[4] Global shutter bedeutet, dass das gesamte FoV gleichzeitig abgetastet wird.

des gesamten FoVs erfolgt über einen Zeitraum von etwa 8 ms. In der zweiten Zeile sind Abtastungen einer Kamera dargestellt. Die einzelne Abtastung erfolgt über einen Zeitraum von etwa 16 ms. Das gesamte FoV wird simultan abgetastet. In der dritten Zeile sind die Abtastungen des Lidars dargestellt. Die einzelne Abtastung erfolgt über einen Zeitraum von etwa 100 ms. Das FoV wird rollend abgetastet.

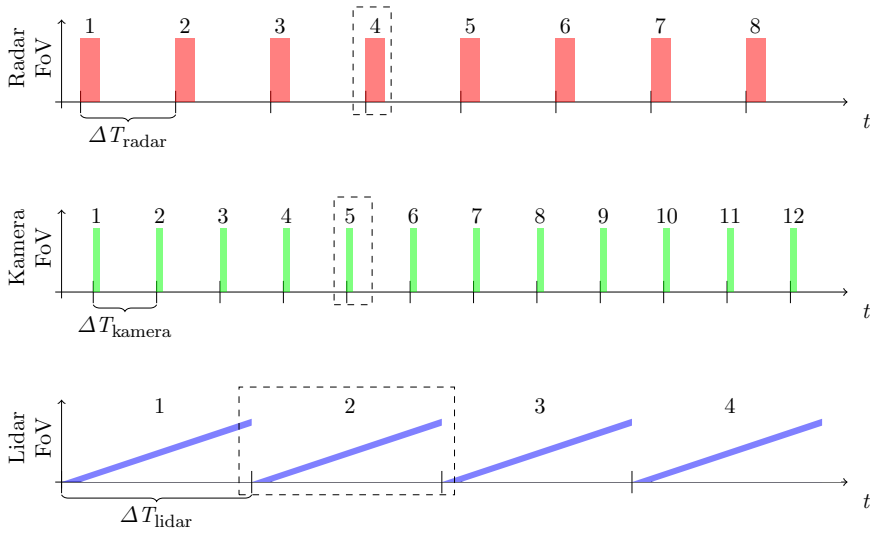


Abbildung 3.9.: **Abtastdiagram der Sensoren:** Die Sensoren tasten zu unterschiedlichen Zeitpunkten und mit unterschiedlichen Frequenzen ab. Radar und Kamera tasten das gesamte FoV gleichzeitig ab, wohingegen Lidar rollend abtastet.

Um einen zeitlichen Versatz der Abtastungen so gering wie möglich zu halten, werden nach der Akquise die Abtastungen der Sensoren gesucht, welche die geringste zeitliche Abweichung voneinander aufweisen. Als Device-under-Test wird der Radarsensor als Taktgeber festgelegt. Zu jedem einzelnen Zyklus des Radarsensors werden also die nächstgelegenen Abtastungen der anderen Sensoren gesucht. Da die Assoziation der Abtastungen nach der Datenakquise erfolgt, kann die zeitliche Assoziation auch nicht kausal sein, d.h. die Abtastungen der anderen Sensoren können auch zu einem späteren Zeitpunkt der Radarauslösung erfolgt sein. Ein Beispiel der Assoziation aus Radar, Kamera und Lidar ist in Abbildung 3.9 als gestrichelte Rechteckboxen markiert. Die assoziierten Frames der sechs Sensoren (Radar, zwei Kameras, zwei Lidar, DGPS-INS) werden hier als 6-Tupel $k(k_{\text{radar}}) = \{k_{\text{radar}}, k_{\text{kamera},1}, k_{\text{kamera},2}, k_{\text{lidar},1}, k_{\text{lidar},2}, k_{\text{dgps}}\}$ zusammengefasst. Zusammen mit der Wahl des 6-Tupels ergeben sich die Startzeitpunkte $t_k = \{t(k_{\text{radar}}), t(k_{\text{kamera},1}), t(k_{\text{kamera},2}), t(k_{\text{lidar1}}), t(k_{\text{lidar},2}), t(k_{\text{dgps}})\}$ der Sensorabtastungen. Bei der Datenakquise kommunizieren alle Sensoren ihre Messdaten mit dem Messrechner als User-Datagram-Protocol (UDP)-Botschaften. Die UDP-Nachrichten enthalten den Startzeitpunkt der Aufnahme, nach dem das 6-Tuple t_k später erstellt werden kann.

Nach der Assoziation wurde die automatische Assoziation anhand der Projektion der Detektionen aus Radar und Lidar in die Kamerabilder bewertet. Diese wurde als gut empfunden. Durch die rollende Abtastung des Lidarsensors ergab sich jedoch insbesondere bei dynamischen Fahrszenarien eine schlechte Kongruenz der Projektion

von Lidar-Messungen im Kamerabild. Diese wird im nächsten Abschnitt dargestellt und synthetisch korrigiert.

3.7.2. Korrektur der Abtastungen aus Lidarsensor

Definiert sei

$$\Delta t_{\text{ping}}(t) = \Delta T_{\text{lidar}} \frac{t - t(k_{\text{lidar}})}{\Delta T_{\text{lidar}}} - \Big(t(k_{\text{kamera}}) - t(k_{\text{lidar}})\Big) = t - t(k_{\text{kamera}}), \tag{3.36}$$

als die zeitliche Differenz eines jeden Lidar Pings[5] bezogen auf den Startzeitpunkt des assoziierten Kameraframes. Der Term $\Delta T_{\text{lidar}} \frac{t - t(k_{\text{lidar}})}{\Delta T_{\text{lidar}}}$ beschreibt die zeitliche Differenz durch die rollende Abtastung. Der Term $\Big(t(k_{\text{kamera}}) - t(k_{\text{lidar}})\Big)$ beschreibt die zeitliche Differenz der Startpunkte von Lidar und Kameraaufnahme.

Ein geometrischer Versatz der Abtastung wird gemäß dieser zeitlichen Differenz kompensiert. Dabei wird angenommen, dass sich die Pings ausschließlich entsprechend der Bewegung des Ego-Fahrzeuges in Weltkoordinaten bewegen, also stationär gegenüber Grund sind. Zur Kompensation wird das in [MDB+17] vorgestellte Verfahren verwendet. Das, auf *Runge-Kutta* basierende Verfahren, integriert die translatorische und rotatorische Bewegung des Fahrzeuges. Das Verfahren ist in Algorithmus 2 zusammengefasst.

Algorithmus 2: Ego-Bewegungs-Korrektur von Lidar-Punkten

Daten: Unkorrigierte Lidar-Punktwolke x_{Ego}, Zeit der Lidarpings bezogen auf Drehstartpunkt t_{ping}, Fahrzeuglängsgeschwindigkeit v_x und Gierrate $\dot{\psi}$
Ergebnis: Korrigierte Lidar-Punktwolke $x_{\text{Ego, corrected}}$
für $p \leftarrow 0$ **bis** $\#\textit{Lidar Pings}$ **tue**

$$\Delta x \leftarrow v_x \Delta t_{\text{ping}}\left(t\left(p\right)\right)$$
$$\Delta \phi_\alpha \leftarrow \dot{\psi} \Delta t_{\text{ping}}\left(t\left(p\right)\right)$$
$$\Delta x_\alpha \leftarrow \Delta x \cos(\Delta \phi_\alpha)$$
$$\Delta y_\alpha \leftarrow \Delta x \sin(\Delta \phi_\alpha)$$
$$x_{\text{Ego, corrected}}(p) \leftarrow \begin{bmatrix} \cos(\Delta\phi_\alpha) & -\sin(\Delta\phi_\alpha) & 0 \\ \sin(\Delta\phi_\alpha) & \cos(\Delta\phi_\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} x_{\text{Ego}}(p) + \begin{bmatrix} \Delta x_\alpha \\ \Delta y_\alpha \\ 0 \end{bmatrix}$$

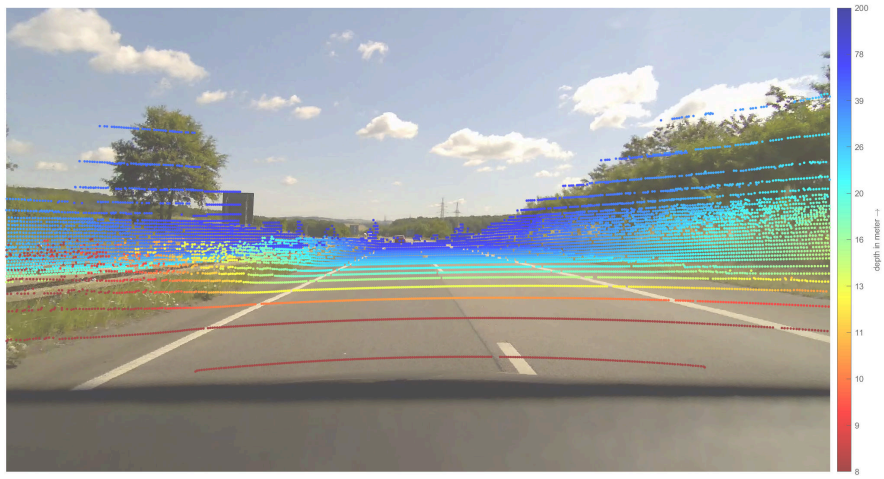
return $x_{\text{Ego, corrected}}$

Dabei entsprechen v_x und $\dot{\psi}$ der Längsgeschwindigkeit und der Gierrate des Fahrzeuges. Da die Messung der Bewegung durch das DGPS-INS im Ego-KOOS vorgenommen wird, werden die Lidar-Pings zunächst in das Ego-KOOS transformiert (Gleichung 3.16) und dort die Kompensation durchgeführt. Durch Bewegung und zeitlichen Versatz ergibt sich eine Translation der Punkte im Ego-KOOS entsprechend Δx_α und Δy_α, sowie eine Rotation um die Hochachse entsprechend um $\Delta \phi_\alpha$.

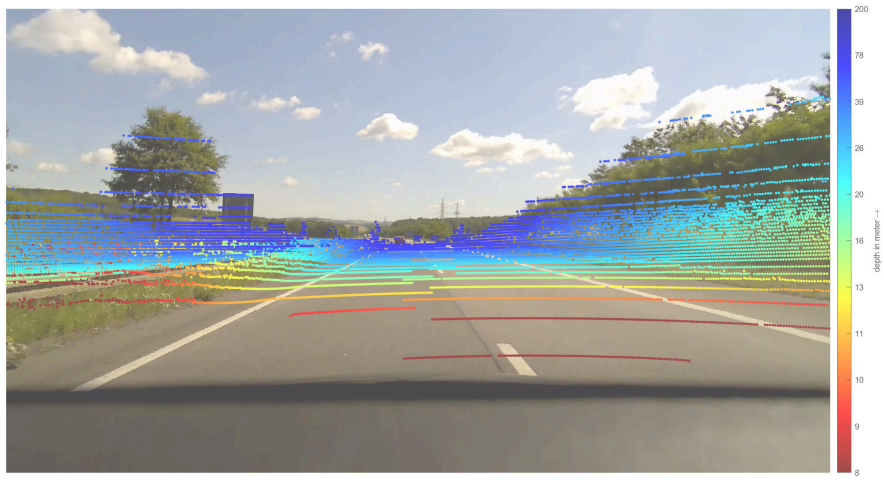
Eine Gegenüberstellung korrigierter und unkorrigierter Lidar-Punktwolken ist in Abbildung 3.10 dargestellt. In dem Beispiel fährt das Ego-Fahrzeug mit üblicher Geschwindigkeit auf einer Autobahn und die Kameras sind nach vorne gerichtet. Zu

[5] Ping als Synonym einer einzelnen räumlichen Abtastung durch den Lidarsensor.

erkennen ist, dass die unkorrigierten Punkte in Abbildung 3.10a den Baum im linken Bildbereich nicht gut treffen. Im Gegensatz dazu schließen die selben Punkte nach der Kompensation in Abbildung 3.10b deutlich besser mit den Konturen des Baums ab.



(a) Ohne Ego-Bewegungskorrektur



(b) Mit Ego-Bewegungskorrektur

Abbildung 3.10.: **Vergleich von projizierten Lidar-Pings auf Kamerabild, vor und nach Ego-Bewegungs Korrektur:** Die Lidar-Pings sind farblich entsprechend der gemessenen Entfernung codiert. Im oberen Bild ist eine schlechte Kongruenz, z.B. am linksseitigen Baum, zu erkennen, welche durch die Ego-Bewegungs Korrektur im unteren Bild behoben wurde.

Angemerkt sei, dass sich diese Kompensation immer auf die einzelnen Kamerabilder bezieht. Da zwei Kameras verwendet werden, wurden die Pings aus den Lidarsensoren jeweils für beide Kamerabilder korrigiert. Zur Vereinfachung der Beschreibung seien im weiteren Verlauf dieser Arbeit immer die korrigierten Lidar-Pings gemeint, wenn Lidar-Daten verwendet werden.

4. Aufbereitung der Sensordaten

Auf den Daten der Referenzsensorik wird eine vorverarbeitende Signalverarbeitung angewandt, welche für die weiteren Verarbeitungsschritte dieser Arbeit benötigt wird. Bei den Verarbeitungsschritten handelt es sich um aus der Literatur bekannte Verfahren zur Tiefenvervollständigung, Normalenschätzung, Schätzung des optischen Flusses und der semantischen Instanz-Segmentierung, welche zur Abgrenzung selbst entwickelter Signalverarbeitungen gesondert in diesem Kapitel vorgestellt werden.

4.1. Tiefenvervollständigung

Trotz der Verwendung von zwei Lidarsensoren werden nicht alle Kamerapixel mit Tiefeninformationen befüllt. Die Ausrichtung der Lidarsensoren wurde so vorgenommen, dass die vertikale Mitte der Kamerabilder dicht mit projizierten Tiefenmessungen aus den Lidarsensoren bedeckt ist. Ein Beispiel dazu ist in Abbildung 4.1 dargestellt.

Abbildung 4.1.: **Abdeckung der Szene durch Lidarmessungen:** Die Abtastungen der Lidarsensoren sind rot (Sensor 1) und grün (Sensor 2) hervorgehoben. Durch Verkippung der Lidarsensoren zueinander wird der Bildinhalt größtenteils dicht vermessen. Es verbleiben jedoch Bildregionen mit geringer Tiefeninformation.

Zu erkennen ist, dass insbesondere im unteren Bereich des Bildes kaum Pixel mit

Tiefeninformationen belegt sind. Analog fehlt auch im oberen Bildbereich die Tiefenmessung, welche aufgrund der zu großen Entfernung zum Himmel nicht durch die Lidarsensoren vermessen wurde. In diesem Abschnitt wird die spärliche Tiefenmessung im Kamerabild durch Tiefenschätzungen ergänzt, so dass für jedes Pixel im Kamerabild ein Tiefenwert geschätzt wird. Da wir im späteren Verlauf dieser Arbeit für das gesamte Kamerabild eine automatische Annotation der Radardaten ableiten werden, ist diese dichte Tiefeninformation notwendig.

In vielen automotiven Datensätzen, wie beispielsweise dem *KITTI*-Datensatz [USS+17], ist die Tiefenvervollständigung (engl.: „depth completion“) eine eigenständige Disziplin. Das Ziel ist dabei, die spärliche Tiefenmessung des Lidars um geschätzte Tiefeninformationen zu erweitern, so dass für das gesamte Kamerabild Tiefeninformation vorliegt. Einige Verfahren zur Tiefenvervollständigung sind in [GLSU12], [DT06] zu finden. In der vorliegenden Arbeit wird das Verfahren aus [HN09] verwendet, welches eine Erweiterung des Markov-Zufallsfelds aus [DT06] darstellt. Das Markovfeld nutzt die Bildgradienten im Kamerabild, um die spärliche Tiefeninformation in Pixelregionen ohne Tiefenmessung zu propagieren. Es wird davon ausgegangen, dass der Tiefenverlauf stetig in Bildbereichen homogener Farbe ist und nur an Farbkanten im Kamerabild Tiefensprünge entstehen. Das Markovfeld propagiert die spärlichen Tiefeninformationen so, dass homogene Bildbereiche auch homogene Tiefeninformationen erhalten und Tiefensprünge an Bildkanten stattfinden. Eine anschauliche Darstellung des Markovfeldes ist in Abbildung 4.2 zu finden.

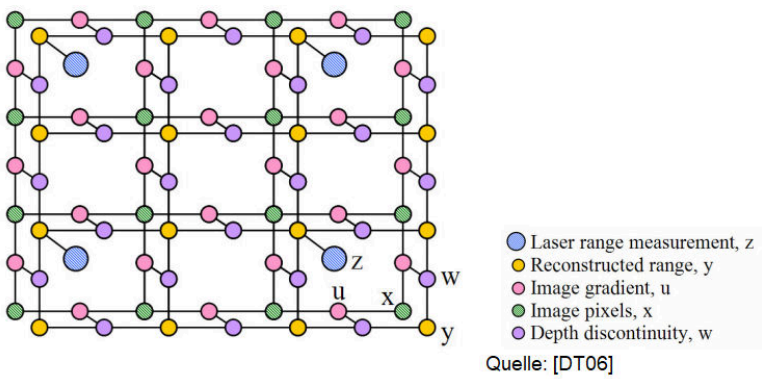


Abbildung 4.2.: **Markov-Zufallsfeld zur Tiefenvervollständigung:** Das Markov Zufallsfeld zur Tiefenvervollständigung aus [DT06]. Die blauen Punkte entsprechen den Tiefenmessungen durch Lidar. Die gelben Punkte entsprechen den geschätzten Tiefen. Die rosafarbenen Punkte entsprechen den Bildgradienten. Die grünen Punkte entsprechen den Bildpixeln. Die lilafarbenen Punkte entsprechen dem Tiefengradienten.

In der Darstellung werden die Pixel des Kamerabildes durch das reguläre Gitter der grünen Punkten dargestellt. Zwischen den Pixeln lassen sich Farbgradienten berechnen, welche durch die rosafarbenen Punkte dargestellt sind. Die blauen Punkte stellen die spärliche Tiefenmessung durch Lidar dar. Ziel ist die Schätzung des dichten Tiefengitters, dargestellt durch gelbe Punkte. Das Gitter dieser gelben Punkte wird durch die Tiefenmessung von Lidar und durch den Farbgradienten des Kamerabildes bzw. daraus abgeleitete Hilfsgitter (lilafarbene Punkte) bestimmt.

Es wurde in [HN09] beobachtet, dass das Markovfeld parallel zum Kamerabild verlaufende Oberflächen bevorzugt, da Tiefengradienten primär an Kanten im Kamerabild zugelassen werden. Verlaufen Objektoberflächen jedoch nicht parallel zum Kamerabild, so haben benachbarte Pixel meist ähnliche Farbintensitäten und trotzdem einen Tiefengradienten. [HN09] erweitern das Markovfeld durch eine Stetigkeitsbedingung zweiter Ordnung auf dem Tiefenbild. Diese zusätzliche Bedingung ermöglicht eine bessere Propagation der Tiefeninformation in Bildbereiche mit linear oder parabolisch verlaufenden Oberflächen. Beispiele für die Tiefenvervollständigung auf unseren Daten mit dem Verfahren aus [HN09] sind in Abbildung 4.3 zu finden.

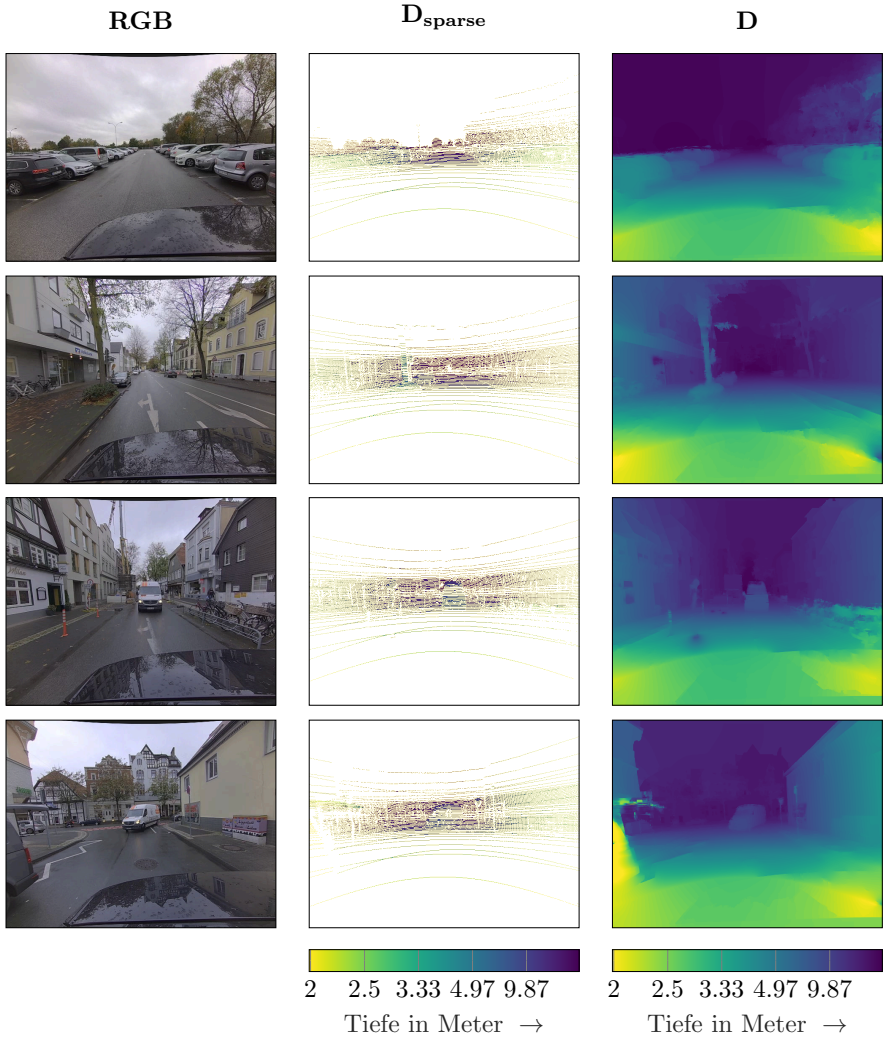


Abbildung 4.3.: **Beispiele der Tiefenvervollständigung:** Links: Kamerabilder; Mitte: Spärliche Tiefeninformation aus Lidarsensoren; Rechts: Verdichtete Tiefeninformation aus Kamerabild.

Die linke Spalte der Abbildungen zeigt beispielhafte Kamerabilder, die mittlere Spalte die Projektion der spärlichen Lidarpunktwolke im Kamerabild, und die rechte Spalte die damit geschätzten dichten Tiefenmasken. Als Referenz werden die Symbole **RGB**, $\mathbf{D}_{\text{sparse}}$ und **D** verwendet. Wie oben beschrieben, ist in Abbildung 4.3 zu sehen,

dass die Farbverläufe der Tiefenmasken **D** an geraden Flächen stetig verlaufen. Die Tiefenvervollständigung wurde als plausibel empfunden und keine weitere Untersuchung zur Genauigkeit der verwendeten Tiefenvervollständigung durchgeführt.

4.2. Schätzung der Oberflächennormalen

Im späteren Verlauf der vorliegenden Arbeit wird geschätzt, von welchen Pixeln im Kamerabild eine signifikante Reflexion für den Radar zu erwarten ist. Die Daten der Referenzsensorik liefern keinen direkten Messwert bezüglich des Reflexionsvermögens von EM-Wellen für 77 GHz Radar. Das Reflexionsvermögen soll daher automatisch aus den Daten der Referenzsensorik geschätzt werden.

Einen hohen Einfluss auf das Reflexionsvermögen bzw. die relative Reflexionsamplitude hat nach den Fresnelschen-Gleichungen aus Unterabschnitt 2.1.1, der Aspektwinkel an Grenzflächen. Für die Berechnung des Aspektwinkels ist der Normalenvektor der Oberflächen zu schätzen, was nachfolgend beschrieben wird.

Zur Schätzung der Oberflächennormalen werden wir der Implementierung aus der freien Mathematik Software Octave [Oct22] folgen. Die Pixel im Tiefenbild der Kamera werden dazu zunächst in kartesische Koordinaten im Radarkoordinatensystem transformiert, siehe Gleichung 3.3, 3.25 und 3.31. Anschließend werden die diagonalen Richtungsvektoren benachbarter Pixel im Kamerabild geschätzt. Über das Kreuzprodukt der Richtungsvektoren wird dann die Oberflächennormale $\mathbf{n}_c$ geschätzt.

In Abbildung 4.4 sind einige Beispiele von Kamerabildern **RGB**, dazugehörigen Tiefenmasken **D** und geschätzten Oberflächennormalen **N** dargestellt. Die Ausrichtung der Oberflächennormalen wurde farblich codiert. Die Beschreibung der Farbcodierung ist der Bildunterschrift zu entnehmen.

Zu erkennen ist, dass Bereiche der Straße und Gehwege fast ausschließlich grün dargestellt sind; die zugehörigen Oberflächennormalen somit hauptsächlich vertikal ausgerichtet geschätzt wurden. In der zweiten, dritten und vierten Zeile sind die Oberflächennormalen der Hauswände hauptsächlich rot dargestellt, was einer horizontal im Kamerabild ausgerichteten Normale entspricht. In der ersten, dritten und vierten Zeile sind Teilbereiche der Fahrzeuge blau dargestellt, welches einer orthogonal zur Bildebene entsprechenden Ausrichtung der Normalen entspricht. Die beschriebenen Bereiche stellen plausible Schätzergebnisse dar, jedoch gibt es auch Bildregionen, in denen die Normalen unplausibel sind. Beispielhaft dafür sind, dass die unteren linken Bildbereiche blau gefärbt sind, obwohl dort ebenfalls die Straße im Kamerabild abgebildet ist und entsprechend eine vertikale Normale bzw. grüne Einfärbung zu erwarten wäre. Ursächlich dafür ist, dass die Oberflächennormalen aus dem vervollständigten Tiefenbild geschätzt werden. Etwaige Fehler in dieser Tiefenschätzung resultieren entsprechend in fehlerhaften Schätzungen der Oberflächennormalen. Es ist wahrscheinlich, dass die Abtastung durch Lidar in manchen Bildregionen zu spärlich war, so dass dort keine vernünftige Tiefenvervollständigung erfolgen konnte. Da der wissenschaftliche Mehrwert dieser Arbeit nicht in der Tiefenverwollständigung und Oberflächennormalenschätzung liegen soll, werden die Ergebnisse für den weiteren Verlauf der Arbeit so akzeptiert und keine mögliche Verbesserung der Oberflächenschätzung durchgeführt.

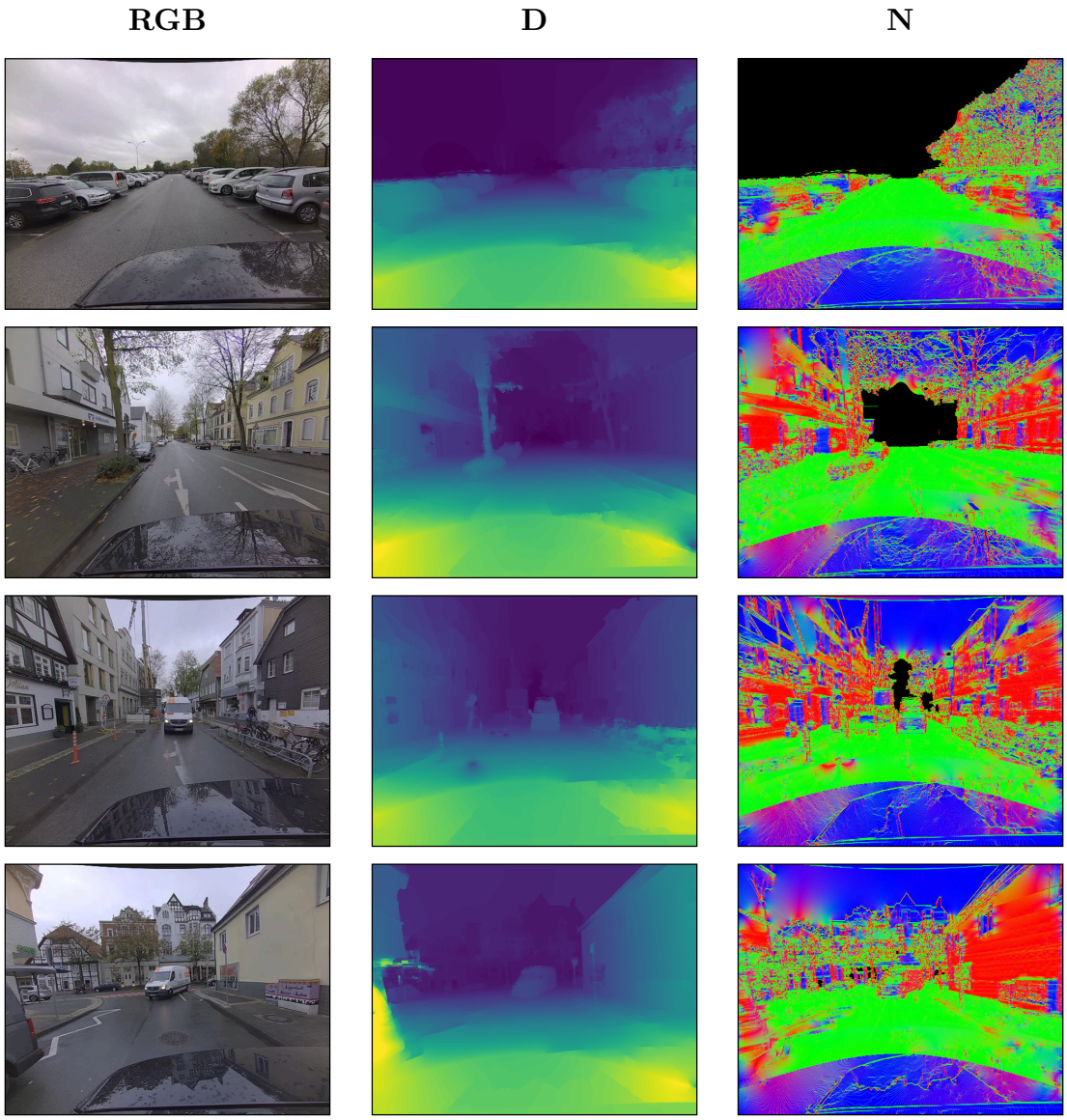


Abbildung 4.4.: **Beispiele der Oberflächennormalenschätzung:** Links: Kamerabilder; Mitte: dichte Tiefenmaske; Rechts: geschätzte Oberflächennormalenausrichtung. Die Koordinaten der Oberflächennormalen wurden in die RGB-Kanäle eingetragen. Rote Pixel entsprechen Oberflächennormalen mit hauptsächlich horizontaler Ausrichtung. Grün entsprechend vertikaler Ausrichtung. Blau entsprechend Ausrichtung orthogonal zur Bildebene.

4.3. Semantische Instanz-Segmentierung der Kamerabilder

Für weitere Verarbeitungsschritte ist ein tieferes Verständnis der Szene notwendig. Ein wichtiger Informationsträger stellen dabei die Bilder der Kameras dar. Zwar fehlt den Kamerabildern im Vergleich zum verwendeten Lidarsensor die Tiefeninformation. Dafür wird zum einen die Farbinformation der Umgebung mit abgebildet und zum anderen eine deutlich größere Winkelauflösung erreicht. Man vergleiche dazu die projizierten Lidar-Pings im Kamerabild mit der Anzahl der Pixel im Kamerabild, siehe z.B. Abbildung 4.1. Diese Merkmale führen dazu, dass die Kamerabilder im Vergleich zu spärlichen Punktwolken, z.B. aus Lidarsensoren, für den menschlichen Betrachter eingängiger und verständlicher sind. Auch im Feld der „maschinellen Bildverarbeitung“ werden Bilder als Informationsträger verwendet und maschinell z.B. semantische Informationen aus den Bildern extrahiert. Eine typische Aufgabe aus der maschinellen Bildverarbeitung stellt dabei die semantische Instanzsegmentierung dar. Das Ziel dabei ist, alle Instanzen gesuchter Objektklassen im Kamerabild zu detektieren. Gleichzeitig sind zu einer Instanz gehörende Pixel zu markieren. Beispiele einer solchen Segmentierung sind in Abbildung 4.5 dargestellt.

Abbildung 4.5.: **Semantische Instanz-Segmentierung:** Beispielhafte Ergebnisse der automatischen semantischen Instanz-Segmentierung.

Die automatische semantische Instanz-Segmentierung der Kamerabilder erfolgte in dieser Arbeit mit Hilfe des Verfahrens „mask_rcnn_inception_v2_coco“ aus dem

Tensorflow-Model-Zoo [HRS+17, Ten20]. Das Verfahren basiert auf einem CNN, welches speziell für die Aufgabe der semantischen Instanz-Segmentierung auf dem *COCO*-Datensatz trainiert wurde. Das Netzwerk wurde gewählt, da es zufriedenstellende Segmentierungen erreicht und dabei noch akzeptable Laufzeit (inklusive Export der Daten) von etwa 300 ms pro Kameraframe auf dem verwendeten PC erreicht.

Für die Inferenz der Segmentierung wird ausschließlich das Kamerabild zum Frame verwendet und die Prädiktion in Form einer Maske $\mathbf{M}$ erzeugt. Die Maske $\mathbf{M}$ umfasst die einzelnen Pixel zu jeder detektierten Objektinstanz. Da das verwendete Verfahren nicht ausschließlich für automotive Anwendungen trainiert wurde, kann es eine Vielzahl von hier nicht relevanten Objektklassen wie z.B. Flugzeugen, Schiffen oder Tieren wie Giraffen oder Walen erkennen. Die im Rahmen dieser Arbeit als wesentlich beurteilten Objektklassen sind: Fußgänger, PKW, LKW, Fahrrad und Motorrad. Detektionen anderer Objektklassen durch das CNN werden im weiteren Verlauf dieser Arbeit ignoriert.

Es seien zusätzlich noch die Masken $\mathbf{M}_{\text{Veh.}}$ und $\mathbf{M}_{\text{Ped.}}$ definiert, welche die Instanzmasken aller PKW, LKW und Zweiräder bzw. aller Fußgänger abbilden.

4.4. Verfeinerung der Instanzmasken durch Clusterbildung

Die zuvor vorgestellte semantische Instanz-Segmentierung erlaubt eine Detektion von Objekten in der Kameraebene. Da die geschätzten Instanzmasken nicht fehlerfrei sind, kommt es zum Beispiel vor, dass die Instanzmaske gelegentlich Pixel detektiert, welche nicht mehr zur abgebildeten Objektinstanz gehören. Häufig kann beobachtet werden, dass z.B. bei hintereinander parkenden Fahrzeugen ein Teil der Pixel des hinteren Fahrzeuges dem vorderen Fahrzeug zugeordnet wird und umgekehrt. Obschon die Objekte im Kamerabild benachbart liegen, so sind sie im dreidimensionalen Raum zwingend getrennt. Diese Eigenschaft wird genutzt, um mögliche Fehler in der Instanz-Segmentierung automatisch zu erkennen. Dazu werden Pixel des Kamerabildes nach der semantischen Instanz-Segmentierung weiter geclustert.

Mit Hilfe der zuvor geschätzten Tiefenmaske aus Abschnitt 4.1 werden die Pixel, die zu einer Instanzmaske gehören, zunächst in Kamerakoordinaten transformiert, siehe Gleichung 3.23a, Gleichung 3.24a und Gleichung 3.31. Die sich ergebene, zu einer Instanzmaske gehörende Punktwolke wird über das Density-Based-Spatial-Clustering-of-Applications-with-Noise (DBSCAN) Verfahren [EKSX96] weiter geclustert. Dabei werden Punkte, welche eine euklidische Abweichung von mehr 0.3 m zum nächstgelegenen Cluster haben, als Ausreißer klassifiziert und aus dem Datensatz ausgeschlossen. Es resultiert eine Menge aller verbliebenen als valide klassifizierten Pixel $\mathcal{P}_{\text{DBSCAN}}$. Für die Wahl der maximalen eingestellten Abweichung wurde angenommen, dass die Winkelauflösung der Referenzsensorik 0.1° beträgt und die maximale Entfernung 30 m. Ein möglicher tangentialer Fehler ergibt sich somit zu $\sin(0.1°)30\,\text{m} = 0.05\,\text{cm}$ bzw. überschlägig als Summe in beide Tangentialrichtungen zu 0.1 cm. Die Parameterwahl sollte diesen Wert nicht unterschreiten, so dass sich 0.3 m als valider Wert ergab.

4.5. Optischer Fluss

Im Folgenden Kapitel 5 wird aus den Daten der Sensoren eine Schätzung der 3D-Bewegung sämtlicher Kamerapixel durchgeführt. Dafür wird eine Schätzung der Be-

wegung in der Bildebene benötigt, bei welcher für jedes Pixel im Kamerabild die Verschiebung in horizontaler und vertikaler Bildachse ermittelt wird. Diese Verschiebung in der Bildebene wird in der Literatur auch als „optischer Fluss“ [HS81] bezeichnet und in unzähligen Veröffentlichungen behandelt.

Zum Zeitpunkt der Bearbeitung dieses Werkes erreichte das von Yin [YDY19] veröffentlichte Verfahren einen der vorderen Plätze im *KITTI* „Optical Flow Evaluation 2015“ benchmark. Das Verfahren hob sich von anderen Verfahren ab, da es neben sehr guten Schätzgenauigkeiten des optischen Flusses auch noch Schätzkonfidenzen auf Pixelebene bereitstellt. Diese Konfidenzen geben einen Anhaltspunkt über Fehler im optischen Fluss, und die entsprechenden Pixel können automatisch aus der Berücksichtigung für die automatische Annotation der Radardaten ausgeschlossen werden.

Es wird an dieser Stelle auf eine ausführliche technische Beschreibung des Verfahrens verzichtet und der interessierte Leser stattdessen auf die entsprechende Veröffentlichung [YDY19] verwiesen. Zum anschaulichen Verständnis seien stattdessen Ergebnisse der optischen Flussschätzung in Abbildung 4.6 dargestellt.

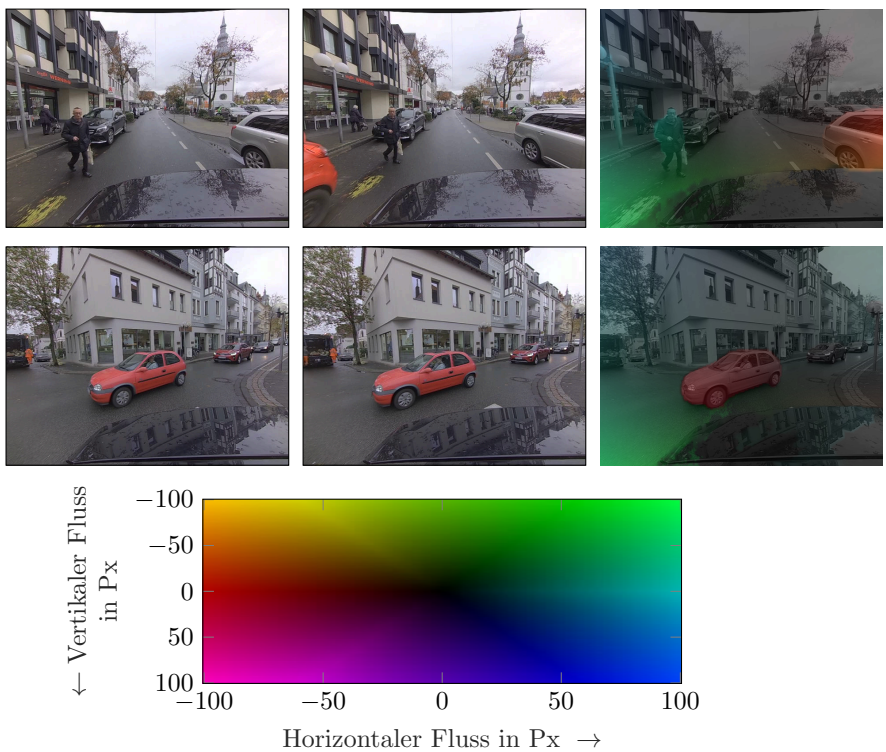


Abbildung 4.6.: **Beispiel des optischen Flusses:** Links und Mitte: Zwei aufeinander folgende Kamerabilder. Rechts; Resultierender optischer Fluss zwischen den Kamerabildern. Unten: Farblegende zum optischen Fluss mit Angabe in Pixeln.

In den ersten beiden Spalten der Abbildung sind jeweils zwei aufeinander folgende Kamerabilder dargestellt, zwischen denen der optische Fluss geschätzt wurde. Der optische Fluss ist jeweils auf der rechten Seite dargestellt. Die farbliche Kodierung entspricht der in der Literatur üblichen Darstellung, wobei der Farbton (engl.: „hue“) gemäß der Bewegungsrichtung des optischen Flusses und der Hellwert (engl.: „value“)

entsprechend dem Betrag des optischen Flusses skaliert wird. Die Farblegende ist im unteren Bereich der Abbildung dargestellt. In den Beispielen ist zu erkennen, dass in den jeweils unteren linken Ecken der Bilder ein diagonal nach rechts oben gerichteter optischer Fluss geschätzt wurde. Dies ist plausibel, da sich die Kamera geradlinig bewegt und die abgebildeten Objekte sich in Richtung des Bildzentrums bewegen. Im unteren Beispiel ist ein roter Kleinwagen zu sehen, welcher sich in Richtung des linken Bildbereiches bewegt und folgerichtig der optische Fluss geschätzt wurde. Im weiteren Verlauf dieser Arbeit werden wir **F** als Bezeichnung für die Maske des optischen Flusses verwenden.

5. Schätzung der Radialgeschwindigkeit aus Referenzsensordaten

In dieser Arbeit werden Verfahren zur kreuzmodalen Supervision von NN basierten Verfahren zur Signalverarbeitung von Radarfrequenzspektren und Kamerabildern vorgestellt. Die Frequenzspektren geben, wie in Unterabschnitt 2.1.3.3, 2.1.3.4 und 2.1.5 gezeigt, die Dimensionen Entfernung, Radialgeschwindigkeit und Einfallswinkel der abgetasteten EM-Wellen wieder. Bei der automatischen Annotation müssen die Labels also entsprechend Entfernung, Doppler und Winkel in Radarfrequenzspektren platziert werden. Entfernung und Einfallswinkel der Umgebung lassen sich mittels der in Kapitel 4 vorgestellten Tiefenvervollständigung schätzen. Zu beachten ist, dass die Tiefenvervollständigung im Kamerakoordinatensystem vorliegt und für die Annotation mittels der Koordinatentransformationen aus Kapitel 3 in das Radarkoordinatensystem transformiert werden muss. Zur vollständigen Platzierung der Labels in dem Radarfrequenzspektrum werden noch Doppler- bzw. Radialgeschwindigkeit der Kamerapixel benötigt. Ein entsprechendes Verfahren zur Geschwindigkeitsschätzung der Kamerapixel wird in diesem Kapitel vorgestellt. Es handelt sich um eine Erweiterung eines aus der Literatur bekannten Verfahrens zur 3D-Szenenflussschätzung.

Das Kapitel wird begonnen mit einer generischen Beschreibung des Szenenflusses (Abschnitt 5.1), ehe in Abschnitt 5.2 eine Kurzbeschreibung der Neuerungen des hier entwickelten Verfahrens zur Szenenflussschätzung vorgestellt wird. Anschließend werden die Eingangsdaten aufgelistet (Abschnitt 5.3) und elementare Mengendefinitionen vorgestellt (Abschnitt 5.4), ehe das zugrunde liegende Modell der Bewegung eingeführt wird (Abschnitt 5.5). Die Bewegungsparameter des Modells werden aus den Eingangsdaten geschätzt, ehe am Ende des Kapitels das Verfahren evaluiert wird. Dabei wird untersucht, welche Szenenflussgenauigkeit von dem vorgestellten Schätzer zu erwarten ist (Abschnitt 5.8).

5.1. Erläuterungen zum 3D-Szenenfluss

Als Szenenfluss bezeichnet man in der Wissenschaft zur digitalen Bildverarbeitung das (semi-)dichte 3D-Vektorfeld zur Beschreibung der Bewegungen von Pixelinhalten relativ zur Kamera. Die Bewegung setzt sich dabei aus zwei Komponenten in der Bildebene sowie einer Komponente orthogonal zur Bildebene, der Bildnormalen, zusammen. Eine Entfernungsmessung in Bildnormalen ist bei typischen Mono-Kameras nicht direkt möglich, somit ist die Berechnung eines 3D-Bewegungsfeldes eine herausfordernde Aufgabe. In vielen Publikationen wird eine Entfernungsmessung in Bildnormalen durch Verwendung von RGB-D oder Stereo-Kamerasystemen erreicht und entsprechend die Schätzung einer Bewegung, insbesondere die Komponente in Richtung der Bildnormalen, verbessert. Eine Übersicht möglicher Szenenflussverfahren ermöglicht die *KITTI Scene Flow Evaluation 2015*, in welcher eine Reihe von Verfahren aufgelistet und bewertet

sind [MHG18, MHG15].

Da immer wieder neue Verfahren mit besseren Schätzergebnissen in der *KITTI* Hierarchie veröffentlicht werden, ist dies Nachweis dafür, dass die Schätzung des 3D-Bewegungsfeldes weiterhin herausfordernd bleibt. Wir definieren das Bewegungsfeld $\boldsymbol{\xi}$ durch Bildung von Differenzenquotienten

$$\boldsymbol{\xi}_{[\mathbf{p}]}^{(k)} = \mathbf{x}_{C[\mathbf{p}]}^{(k)} - \mathbf{x}_{C[q(\mathbf{p})]}^{(k-1)} \tag{5.1}$$

wobei $\mathbf{p}$ ein Pixel im aktuellen Kamerabild und $q(\mathbf{p})$ ein korrespondierendes Pixel im zeitlich benachbarten Kamerabild ist. k und $k-1$ beschreiben zeitlich benachbarte Kameraframes. Die Assoziationsfunktion $q(\mathbf{p})$ ergibt sich nicht automatisch und muss durch das Verfahren zur Szenenflussschätzung selber ermittelt werden. Für eine gute Szenenflussschätzung muss zum einen die Position $\mathbf{x}_C$ der Punkte möglichst fehlerfrei gemessen werden und anschließend die Assoziationsfunktion $q(\mathbf{p})$ eine fehlerfreie Assoziation der Punkte zwischen den Frames durchführen.

5.2. Erweiterung von DRISF zu DRISFwR

Während der Entwicklung dieser Arbeit erreichte das Verfahren Deep-Rigid-Instance-Scene-Flow (DRISF) zeitweise die höchsten Metriken im *KITTI Scene Flow Benchmark*. Eine Gemeinsamkeit mit anderen Verfahren ist, dass z.B. die Szene vorab in unterschiedliche Aktoren unterteilt wird. Diese Aktoren werden als Starrkörper angenommen, dessen Pixel sich mit identischer Bewegung in der Szene fortbewegen. Bei korrekter Segmentierung in Aktoren führt diese Annahme dazu, dass viele Datenpunkte (mehrere hundert Pixel im Kamerabild) für die Schätzung weniger Bewegungsparameter verwendet werden können und eine weniger vom Rauschen beeinflusste Schätzung erreicht werden kann. Typischerweise wird die Bewegung jedes Aktors durch jeweils eine 3D-Translation und 3D-Rotation beschrieben.

Zur Schätzung des Szenenflusses nutzt DRISF optische Informationsquellen, sogenannte „visual Cues“. Hierbei handelt es sich um präprozessierte Informationen aus den Sensordaten. Konkret werden bei DRISF der optische Fluss, die Tiefenmasken der Sequenz und die Instanzmaske verwendet. Für diese Cues definiert DRISF Transformationsgleichungen und Gütefunktionen, welche eine Schätzung des Szenenflusses mit Hilfe eines Gauß-Newton Optimierers ermöglichen.

Ein praktischer Vorteil bietet DRISF gegenüber anderen Verfahren durch die modulare Trennung der Cues und des Optimieres, welches eine einfache Pflege der Cues ermöglicht. Ergibt sich mit der Zeit Zugang zu Algorithmen mit genauer Instanzsegmentierung, optischen Fluss und Tiefenschätzung, so können diese leicht in DRISF ausgetauscht und somit die Schätzqualität des Szenenflusses verbessert werden, ohne die Transformationsgleichungen anpassen zu müssen. Wie in diesem Kapitel gezeigt wird, lässt sich DRISF aber auch leicht mit anderen Cues erweitern. Konkret wird dies durch Erweiterung von DRISF zu Deep-Rigid-Instance-Scene-Flow-with-Radar (DRISFwR) erreicht, welches als zusätzlichen Cue die RD-map des Radars verwendet und eine entsprechende Erweiterung der Transformationsgleichung zur Berücksichtigung des Cues bereitstellt.

Eine grafische Übersicht der Adaption von DRISF zu Deep-Rigid-Instance-Scene-Flow-with-Radar (DRISFwR) ist in Abbildung 5.1 zu finden.

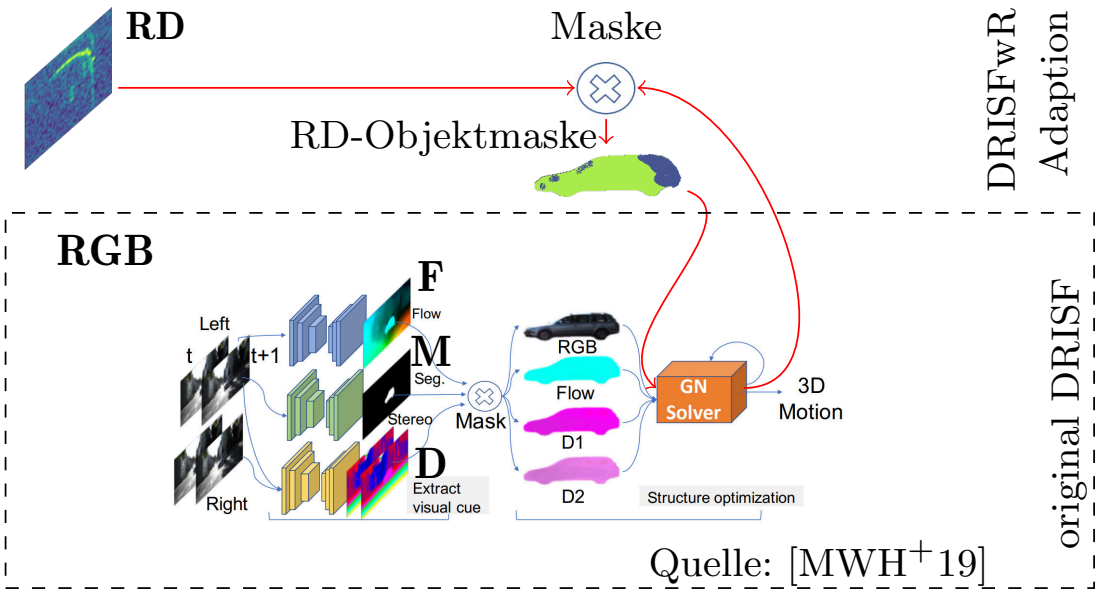


Abbildung 5.1.: **Übersicht von DRISFwR (adaptiert aus [MWH+19]):** Die originale Verarbeitung durch DRISF ist im unteren gestrichelten Rechteck dargestellt. Es werden zunächst aus Stereokamera-Bildpaaren der optische Fluss, Instanzsegmentierung und Tiefenmaske geschätzt. Anschließend wird für die detektierten Instanzen der Szenenfluss mittels Gauß-Newton-Schätzer bestimmt. Darüber ist die Erweiterung zu DRISFwR dargestellt. Nach jeder Iteration wird eine zusätzliche Objektmaske aus der RD-map extrahiert und für die nächste Iteration des Optimierers verwendet. Dieser versucht, Objektmasken entsprechend den Vorgaben durch Variation des Szenenflusses zu extrahieren.

Ausschlaggebend für die Erweiterung zu DRISFwR sind zwei Motivationen. Zum einen besteht der Wunsch, eine gegenüber DRISF verbesserte Szenenflussgenauigkeit zu erreichen. Zum anderen soll durch die Hinzunahme der RD-maps eine automatische Ausrichtung des geschätzten Szenenflusses am Inhalt der RD-maps erreicht werden. Für die zugrunde liegende Aufgabe, die automatische Annotation von Radarfrequenzspektren, ist dies entscheidend, da Fehler in der Platzierung der Annotationen minimiert werden.

Im weiteren Verlauf dieses Abschnittes werden wir die Adaptionen von DRISFwR im Detail vorstellen. Dabei werden die fundamentalen Grundlagen zu DRISF vermittelt. Diese Grundlagen zu DRISF können keinesfalls den gesamten wissenschaftlichen Umfang der Publikationen von DRISF nachbilden. Dem interessierten Leser wird an dieser Stelle empfohlen, die originale Veröffentlichung [MWH+19] zu DRISF zu studieren, bevor mit der weiteren Durchsicht dieser Arbeit fortgefahren wird.

5.3. Eingangsdaten

Als Eingangsdatenmenge für DRISFwR werden vier verschiedene Cues bereitgestellt: Instanzmaske, Tiefenmaske, optischer Fluss und RD-map. Die ersten drei Cues, die

optischen Cues, finden auch bei DRISF Verwendung. Wir führen den letzten Cue als erste Adaption von DRISFwR ein.

5.3.1. Instanzsegmentierung

Bei DRISF wird wie zuvor erwähnt das Kamerabild **RGB** in unterschiedliche Aktoren mit gleichem Bewegungsmuster aufgeteilt. Diese Aufteilung erfolgt automatisch anhand einer Instanzsegmentierung, welche bereits in Abschnitt 4.4 vorgestellt wurde. Abkürzend für die resultierende Instanzmaske werden wir nachfolgend die Abkürzung **M** verwenden.

Bei der Instanzmaske werden ganze Fußgänger oder Fahrzeuge als einzelne Aktoren zusammengefasst. Zwar hätten sich sowohl Fußgänger als auch Fahrzeuge mechanisch besser als Mehrkörpermodelle beschreiben lassen. DRISF verzichtet auf diese Modellierung und nimmt etwaige Ungenauigkeiten zu Gunsten verringerter Komplexität in Kauf.

5.3.2. Tiefenmaske

Für die räumliche Abtastung des Bildinhaltes wird die aus Abschnitt 4.1 bekannte Tiefenvervollständigung verwendet. Dem Verfahren zur Szenenflussschätzung wird dabei die Tiefenmaske zweier zeitlich benachbarter Kameraframes zur Verfügung gestellt, welche wir nachfolgend in der Form $\mathbf{D}^0$ und $\mathbf{D}^1$ abkürzen werden. In der originalen DRISF-Veröffentlichung wurden die Tiefenmasken mittels Stereo-Kamerabildern geschätzt. Da hier jedoch die Tiefenmessung durch Lidarsensoren möglich ist, bietet sich diese Änderung an und demonstriert noch einmal die zuvor beschriebene Modularität des Verfahrens.

5.3.3. Optischer Fluss

Der optische Fluss beschreibt die Verschiebung der Pixel in der Kameraebene zwischen zwei zeitlich benachbarten Kamerabildern. In Abschnitt 4.5 wurde die Schätzung des optischen Flusses mit dem HD^3 Verfahren nach [YDY19] vorgestellt. Wir werden die resultierende Maske des optischen Flusses nachfolgend durch das Symbol **F** abkürzen.

5.3.4. RD-map

Als neue Quelle führt DRISFwR die RD-map **RD** aus dem Radarsensor ein. Eine Einführung zum RD-map wurde bereits in Unterabschnitt 2.1.3.4 gegeben.

Neben der RD-map stellt ein typischer Radarsensor auch noch identifizierte Detektionen bereit. Alternativ oder ergänzend zum RD-map können auch diese Detektionen als Cue für DRISFwR verwendet werden. Der Autor dieser Arbeit hat sich jedoch explizit dafür entschieden, diese Detektionen nicht bei der Inferenz des Szenenflusses bei DRISFwR zu verwenden, da sie (a) als Untermenge in der RD-map bereits enthalten sind und (b) die Szenenflussinferenz möglichst unabhängig von klassischer Radarsignalverarbeitung sein sollte, um mögliche Fehler nicht in das Training eines NN zu transportieren.

5.4. Mengendefinitionen

Bevor mit der Einführung in die Algorithmen von DRISFwR eingestiegen wird, werden nun einige Mengen definiert.

Für eine verkürzte Darstellung von Gleichungen definieren wir $\mathcal{I} = \left\{\mathbf{RGB}^0, \mathbf{RGB}^1, \mathbf{D}^0, \mathbf{D}^1, \mathbf{M}^0, \mathbf{F}, \mathbf{RD}\right\}$. Wird eine oder mehrere der Masken benötigt, so wird stellvertretend $\mathcal{I}$ als Argument verwendet.

Bei der Verarbeitung mittels DRISF oder DRISFwR wird zwischen statischen Objekten und potenziell bewegten Objekten unterschieden. Wir definieren dafür die Menge aller Pixel, welche bei der Instanzsegmentierung als Fußgänger oder Fahrzeuge erkannt wurden

$$\mathcal{P}_{\text{fg}} \coloneqq \left\{\mathbf{p} \middle| \mathbf{M}(\mathbf{p}) \in \{\text{pedestrian, car, truck, bicycle, motorbike}\}\right\}. \tag{5.2}$$

Durch die Hinzunahme von RD-maps in DRISFwR muss berücksichtigt werden, dass Kamerapixel außerhalb des Radar FoV liegen können und somit für den Radar unsichtbar sind. Bei der Optimierung sollen diese Pixel ausgeschlossen werden. Dazu seien entsprechend vom Radar FoV eingeschlossene Pixel definiert als

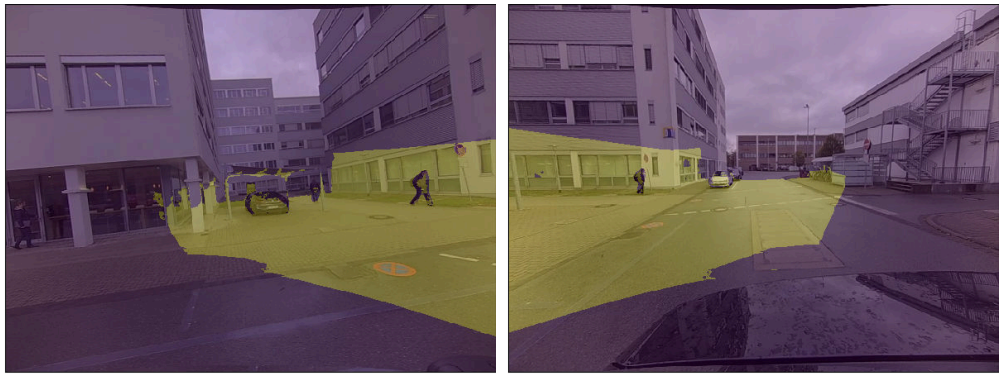
$$\mathcal{P}_{\text{radar}} \coloneqq \left\{\mathbf{p}(\mathbf{x}_R) \middle| |\phi_{\text{az.}}(\mathbf{x}_R)| \leq \frac{135^\circ}{2} \wedge \phi_{\text{el.}}(\mathbf{x}_R)| \leq \frac{22^\circ}{2}\right\}. \tag{5.3}$$

Hierbei sind $\phi_{\text{az.}}(\mathbf{x}_R) = \arctan\left(\mathbf{x}_{R[y]}/\mathbf{x}_{R[x]}\right)$ und $\phi_{\text{el.}}(\mathbf{x}_R) = \arctan\left(\mathbf{x}_{R[z]}/\mathbf{x}_{R[x]}\right)$ der Azimut- und Elevationswinkel der Pixel.

In Abschnitt 4.4 wurden Ausreißer in den Instanzmasken detektiert. Auch diese Ausreißer sollen bei der Optimierung unberücksichtigt bleiben. Dazu definieren wir

$$\mathcal{P}_{\text{i}} \coloneqq \left\{\mathbf{p} \mid \mathbf{p} \in \mathcal{P}_{\text{DBSCAN}} \wedge \mathbf{p} \in \mathcal{P}_{\text{radar}}\right\}. \tag{5.4}$$

Diese Menge wird entsprechend als valide Pixelmenge bezeichnet. Beispiele dieser Pixelmenge sind in Abbildung 5.2 dargestellt.



(a) Kamera 1 (b) Kamera 2

Abbildung 5.2.: **Maske zur Selektion valider Pixel:** Beispiele valider Pixel (gelb) für alle Objekte nach $\mathcal{P}_{\text{i}}$, dargestellt für beide Kameras.

5.5. Definition der Bewegung

Mittels des Überlagerungsprinzips setzten wir die Bewegung des gesamten Bildinhaltes aus mehreren Teilbewegungen zusammen. Zunächst wird die Bewegung der Kamera gegenüber Grund definiert. Nachfolgend werden wir diese Bewegung auch als Hintergrundbewegung bezeichnen und aus den Bewegungsdaten des Fahrzeuges, direkt gemessen über das DGPS-INS, berechnen. Die Hintergrundbewegung ergibt für jedes Pixel im Kamerabild einen Geschwindigkeitsvektor. Nach erfolgter Schätzung der Hintergrundbewegung wird eine additive Komponente der Bewegung für Objekte relativ über Grund geschätzt. Diese Komponente wird nachfolgend als Vordergrundbewegung bezeichnet und für alle Pixel der detektierten Aktoren geschätzt. Formal werden wir den Szenenfluss für jedes Pixel als 3D-Vektor $\boldsymbol{\xi} = [\boldsymbol{\xi}_{[x]}, \boldsymbol{\xi}_{[y]}, \boldsymbol{\xi}_{[z]}]^T$ in kartesischen Koordinaten zusammenfassen. Die Hintergrundbewegung bzw. die relative Bewegung der Kamera über Grund wird definiert als $\boldsymbol{\xi}_{bg}$. Die Vordergrundbewegung bzw. die relative Bewegung des Pixels über Grund wird definiert zu $\boldsymbol{\xi}_{fg}$. Nach dem Überlagerungsprinzip ergibt sich die Komposition beider Bewegungen zu

$$\boldsymbol{\xi} = \boldsymbol{\xi}_{bg} + \boldsymbol{\xi}_{fg}. \tag{5.5}$$

5.6. Bestimmung der Hintergrundbewegung

Für ein Fahrzeug ohne (signifikanten) Schlupf an den Reifen lässt sich die Bewegung gut nach dem Ackermann Modell [MW04] modellieren. Es wird dabei angenommen, dass ausschließlich die Vorderräder schwenkbar sind und die Lenkaufgabe übernehmen. Sämtliche Räder rollen ausschließlich um ihre Drehachse. Verlängert man die Drehachsen, so treffen sie in einem Schnittpunkt zusammen, welcher den Drehpunkt des Fahrzeuges bei Kurvenfahrt darstellt. Dadurch ergibt sich eine longitudinale Bewegung am Hinterachsmittelpunkt mit Rotation um das Zentrum der Hinterachse. Die Bewegung eines beliebigen Punktes auf dem Körper des Ego-Fahrzeuges lässt sich extrapolieren.

Wir beschreiben dazu die Bewegung durch Aufteilung der Geschwindigkeit in translatorische $\boldsymbol{v}_{\text{trans.}}$ und rotatorische $\boldsymbol{v}_{\text{rot.}}$ Komponenten

$$\boldsymbol{v} = \boldsymbol{v}_{\text{trans.}} + \boldsymbol{v}_{\text{rot.}}. \tag{5.6}$$

Die translatorische Geschwindigkeit ist der Anteil, welcher durch die Modellierung geradliniger Bewegungen induziert wird. Die rotatorische Geschwindigkeit ist analog dem Anteil, welcher durch Drehung um im Raum liegenden Achsen induziert wird.

Wir wollen nun die Geschwindigkeit eines beliebigen Punktes auf dem Ego-Fahrzeug betrachten. In Abbildung 5.3 sind dazu beispielhaft die Geschwindigkeitsvektoren zweier Punkte auf dem Fahrzeug eingezeichnet. Der Punkt H_1 vorne links sowie der Punkt H_2 hinten rechts im Fahrzeug.

Das Fahrzeug bewege sich auf einer Kreisbahn. Damit ergibt sich die Bahngeschwindigkeit der Punkte nach den Grundlagen der Dynamik, siehe z.B. [Pal14], zu

$$\boldsymbol{v}_{\text{rot.}} = \boldsymbol{\omega} \times \boldsymbol{r}_{\text{rot.}}, \tag{5.7}$$

wobei $\boldsymbol{\omega}$ die Giergeschwindigkeiten um die Achsen und $\boldsymbol{r}_{\text{rot.}}$ der Verbindungsvektor zwischen Drehachse und Bahnpunkt sind, z.B. $H_1 - R$.

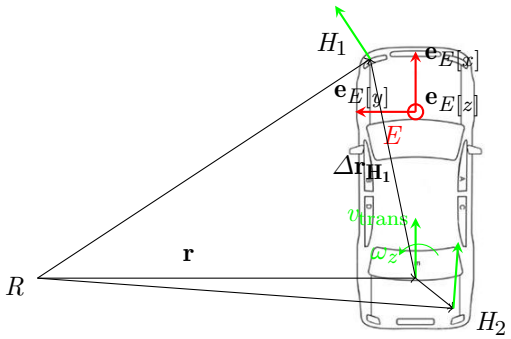


Abbildung 5.3.: **Geschwindigkeit nach Ackermann Prinzip**: Das Fahrzeug rotiert um einen Punkt R mit den Giergeschwindigkeiten $\boldsymbol{\omega}$. Für jede Position auf dem ausgedehnten Fahrzeugkörper ergibt sich aufgrund der unterschiedlichen Verbindungsvektoren zu R eine unterschiedliche Geschwindigkeitskomponente. Fahrzeugkontur nach [ARFssN16].

Das DGPS-INS liefert den Geschwindigkeitsvektor der Fahrzeughinterachse und die Gierraten. Die Geschwindigkeit der Fahrzeughinterachse ergibt sich nach Gleichung 5.7 zu

$$\mathbf{v}_{\text{rot.,HA}} = \boldsymbol{\omega} \times \boldsymbol{r}_{\text{HA}}. \tag{5.8}$$

Ist die relative Position der Punkte H_1 und H_2 gegenüber der Fahrzeughinterachse bekannt, z.B. $\Delta\boldsymbol{r}_{\boldsymbol{H_1}} = H_1 - \boldsymbol{r}_{\text{HA}}$, so ergibt sich der Geschwindigkeitsvektor für Punkt H_1 zu

$$\begin{aligned} \mathbf{v}_{rot.,H_1} &= \boldsymbol{\omega} \times (\boldsymbol{r}_{\text{HA}} + \Delta\boldsymbol{r}_{\boldsymbol{H_1}}) \\ &= \boldsymbol{\omega} \times \boldsymbol{r}_{\text{HA}} + \boldsymbol{\omega} \times \Delta\boldsymbol{r}_{\boldsymbol{H_1}} \\ &= \mathbf{v}_{\text{HA}} + \Delta\mathbf{v}_{H_1}. \end{aligned} \tag{5.9}$$

Die Geschwindigkeit des Punktes setzt sich somit aus der Längsgeschwindigkeit der Fahrzeughinterachse $\mathbf{v}_{\text{HA}}$ und der additiven Geschwindigkeit durch Rotation $\Delta\mathbf{v}_{H_1}$ zusammen.

Wir wollen nun die Position eines beliebigen Punktes p im Ego-Koordinatensystem und Frame k für den nächsten Frame $k+1$ prädizieren. Mit der oben definierten Geschwindigkeit definieren wir die Translation des Punktes im Raum zu

$$\mathbf{t}_{E,bg} = \begin{bmatrix} v_x \Delta T \\ v_y \Delta T \\ v_z \Delta T \end{bmatrix}, \tag{5.10}$$

wobei v_x, v_y, v_z die mit dem DGPS-INS gemessenen Geschwindigkeiten nach Gleichung 5.9 sind. ΔT ist das Abtastintervall zwischen zwei aufeinanderfolgenden Frames. Neben der Translation ergibt sich durch die Drehung des Ego-Fahrzeuges auch eine Drehung des Koordinatensystems. Wir definieren die Drehung als eine Euler-Rotation

um die Fahrzeughochachse zu

$$\mathbf{R}_{E,bg} = \begin{bmatrix} \cos(\omega_z \Delta T) & \sin(\omega_z \Delta T) & 0 \\ -\sin(\omega_z \Delta T) & \cos(\omega_z \Delta T) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{5.11}$$

wobei ω_z die vom DGPS-INS gemessene Gierrate ist. Daraus ermitteln wir die neue geschätzte Position $\tilde{\mathbf{x}}_E^{(k+1)}$ des Punkte $\mathbf{x}_E^{(k)}$ mit

$$\tilde{\mathbf{x}}_{E[\mathbf{p}]}^{(k+1)} = \mathbf{R}_{E,bg}\left(\mathbf{x}_{E[\mathbf{p}]}^{(k)} + \Delta \boldsymbol{r}_{\boldsymbol{H}}\right) - \Delta \boldsymbol{r}_{\boldsymbol{H}} - \mathbf{t}_{E,bg}. \tag{5.12}$$

Zur Kennzeichnung der Position im ersten Frame wird die Symbolik $\{\cdots\}^{(k)}$ verwendet. Für den zweiten Frame entsprechend $\{\cdots\}^{(k+1)}$. $\{\tilde{\cdots}\}$ soll symbolisieren, dass es sich um eine aus dem Szenenfluss geschätzte Position handelt. Bei der Berechnung wurde berücksichtigt, dass nach Ackermann-Modell eine Rotation des Fahrzeuges um die Fahrzeughinterachse erfolgt. In dem dargestellten Modell erfolgte erst die Rotation, bevor die Translation addiert wurde. Es gibt unterschiedliche numerische Verfahren für diese Integration. Anstelle diese aufwendig zu untersuchen, verfeinern wir die kinematischen Zustandsgrößen mittels visueller Odometrie mit dem in [Ali22] vorgestellten Verfahren. Damit folgen wir der Empfehlung aus [MG15a], in welcher eine ungenügende Genauigkeit der Pixelprädiktion mittels Zustandsgrößen aus DGPS-INS beobachtet wurde.

Im weiteren Verlauf besteht besonderes Interesse an der relativen Geschwindigkeit der Umgebung aus Sicht der Kamera. Nach obigem Modell muss dafür die Position der Kamera auf dem Ego-Fahrzeug berücksichtigt werden und damit die Geschwindigkeit der Kamera[1] über Grund ermittelt werden. Durch Anwendung der Koordinatentransformation aus Gleichung 3.31 und Anwendung der Bewegungstransformation aus Gleichung 5.12 erhalten wir

$$\tilde{\mathbf{x}}_{C[\mathbf{p}]}^{(k+1)} = \mathbf{R}_{\mathrm{C,bg}} \mathbf{x}_{C[\mathbf{p}]}^{(k)} + \mathbf{t}_{\mathrm{C,bg}} \tag{5.13a}$$

$$\mathbf{R}_{\mathrm{C,bg}}(k) = \left({}^{E\leftarrow C}\mathbf{R}\right)^{-1} \mathbf{R}_{E,bg}\, {}^{E\leftarrow C}\mathbf{R} \tag{5.13b}$$

$$\begin{aligned} \mathbf{t}_{\mathrm{C,bg}} = &\left({}^{E\leftarrow C}\mathbf{R}\right)^{-1} \mathbf{R}_{E,bg} \left({}^{E\leftarrow C}\mathbf{t} + \Delta \boldsymbol{r}_{\boldsymbol{H}}\right) \\ &- \left({}^{E\leftarrow C}\mathbf{R}\right)^{-1} \left(\Delta \boldsymbol{r}_{\boldsymbol{H}} + \mathbf{t}_{E,bg} + {}^{E\leftarrow C}\mathbf{t}\right). \end{aligned} \tag{5.13c}$$

Dies ergibt schließlich den Hintergrundszenenfluss des Pixels $\mathbf{p}$:

$$\boldsymbol{\xi}_{bg[\mathbf{p}]} = \tilde{\mathbf{x}}_{C[\mathbf{p}]}^{(k+1)} - \mathbf{x}_{C[\mathbf{p}]}^{(k)}, \tag{5.14}$$

Nach Gleichung 5.5 ergibt sich damit der erste Teil des Szenenflusses ($\boldsymbol{\xi}_{bg}$), welcher durch die Bewegung der Kamera über Grund entsteht.

[1] Da die Transformation für beide verwendeten Kameras analog gültig ist, wird auf den Kameraindex in den Gleichungen verzichtet.

5.7. Bestimmung der Vordergrundbewegung

Wesentlich aufwendiger als die Schätzung der Hintergrundbewegung ist die Schätzung der Bewegung von Aktoren über Grund $\boldsymbol{\xi}_{fg}$, da die Bewegung aus den Daten selbst geschätzt werden muss und nicht als direkte Messdaten vorliegt. Die technischen Details dafür werden im Folgenden vorgestellt.

5.7.1. Bewegungsparameter

In Gleichung 5.12 wurde die Positionsänderung durch die Rotation und Translation des Ego-Fahrzeuges beschrieben. Zur Beschreibung der Bewegung wurde die Rotation eines Punktes um die Fahrzeughinterachse berechnet. Damit wurde die Bewegung der Kamera abgeleitet.

Bei größer werdender Distanz zwischen Hinterachse und beobachtetem Punkt steigt der Einfluss der Rotation. Das beschriebene Bewegungsmodell modelliert dabei das spezifische Bewegungsverhalten eines typischen Fahrzeuges im Straßenverkehr und kann deshalb nicht nur für die Modellierung des Ego-Fahrzeuges verwendet werden, sondern auch für andere typische Fahrzeuge im Straßenverkehr, wie Personenkraftwagen (PKW) und Lastkraftwagen (LKW). Im Unterschied zur Ego-Bewegung müssen bei der Bewegung von Aktoren allerdings alle Reflexionspunkte der Aktoren berücksichtigt werden. Durch die geometrische Ausdehnung ergibt sich somit pro Aktor eine ganze Signatur der Bewegung, welche wir kurz überschlagen werden.

Es sei ein typischer PKW mit einer Länge von 5 m angenommen, dessen Hinterachse 1 m vor dem Heck des Fahrzeuges liegt und somit die maximale geometrische Differenz zwischen Fahrzeughinterachse und eines Punktes an der Fahrzeugfront, unbeachtet der Fahrzeugbreite, bei 4 m liegt. Weiterhin sei angenommen, dass das Fahrzeug mit maximal $20\,^\circ\,\mathrm{s}^{-1}$ um die Hinterachse giert[2]. Daraus ergibt sich eine, durch Rotation induzierte, Geschwindigkeit (siehe Gleichung 5.7) von $4\,\mathrm{m}\frac{20^\circ}{\mathrm{s}}\frac{\pi\mathrm{rad}}{180^\circ} \approx \frac{1.4\,\mathrm{m}}{\mathrm{s}}$. Dieser Wert übersteigt die Dopplerauflösung vieler automotiver Radarsensoren und entsprechende Rotation und Objektausdehnung könnte zu einer kinematischen Ausdehnung in der RD-map führen. Ohne empirische Untersuchungen wird hier jedoch angenommen, dass bei typischen Szenarien nur ein kleiner Teil der Fahrzeugkontur gleichzeitig beobachtet wird, so dass die gesamte geometrische und somit kinematische Signatur selten zu beobachten ist. Zugunsten einer sinkenden Komplexität werden wir somit auf die Schätzung der rotatorischen Bewegungsparameter der Verkehrsteilnehmer verzichten und stattdessen ausschließlich die translatorischen Bewegungsparameter ermitteln. Zur Veranschaulichung der beobachteten Geschwindigkeiten aus Sicht des Ego-Fahrzeuges ist in Abbildung 5.4 ein Ego-Fahrzeug und ein gierendes Objekt eingezeichnet. Geschwindigkeitsvektoren sind für einen Punkt an der Fahrzeugfront und dem Fahrzeugheck eingezeichnet.

Weitere typische Verkehrsteilnehmer sind Fußgänger. Aus Sicht des Radars wird bei der Kinematik von Fußgängern häufig zwischen Bewegung von Torso und Extremitäten unterschieden, deren Bewegungsprofil als Mikro-Doppler betitelt wird [vDG08]. Eine Vernachlässigung von Rotationsgeschwindigkeiten beim Fußgänger motivieren wir hier durch zwei Punkte. Erstens wird die Mehrkörperdynamik von Fußgängern ebenfalls nicht in DRISF berücksichtigt, wodurch kein Nachteil von DRISF gegenüber DRISFwR

[2] Dieser Wert wurde bei Durchsicht von [MW04] als ungefährer Maximalwert identifiziert

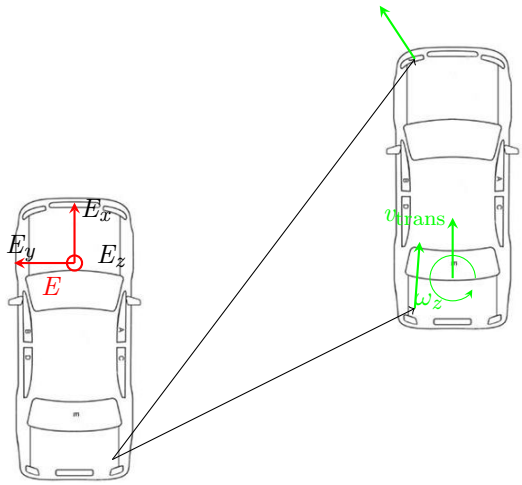


Abbildung 5.4.: **Beobachtete Geschwindigkeiten aus Sicht des Ego-Fahrzeuges:** Zwei beispielhafte Punkte auf dem beobachteten Fahrzeug bewegen sich aus Sicht des Ego-Fahrzeuges mit unterschiedlicher Geschwindigkeit. Zu beachten ist die unterschiedliche Ausrichtung der Geschwindigkeitsvektoren an Fahrzeugfront und -heck. Fahrzeugkontur nach [ARFssN16].

entsteht. Und zweitens: Für die später betrachteten Anwendungen ist eine Annotation der makroskopischen Bewegung bzw. der Torsobewegung in der RD-map ausreichend. Ein auf RD-map operierendes CNN sieht weiterhin die volle kinematische und geometrische Signatur eines reflektierenden Objektes im RD-map, wird jedoch gegen die gemittelten Zielwerte des gesamten Körpers trainiert.

Im Gegensatz zu DRISF wird nun also die Vordergrundbewegung in 3 translatorische Bewegungsparameter kodiert, anstatt in 6 (3 rotational + 3 translational). Diese Vereinfachung wird zur Verringerung der Komplexität und zur möglichen Verbesserung der Robustheit vorgenommen und wie oben beschrieben begründet.

5.7.2. Formulierung der Optimierungsfunktionen

Für eine automatische Schätzung der Bewegungsparameter wird zunächst eine Zielfunktion definiert, welche durch Anpassung der Bewegungsparameter optimiert bzw. minimiert wird. Diese Zielfunktion beschreibt, wie gut die Punkte aufeinanderfolgender Frames durch Anwendung der geschätzten Bewegung ineinander überführt werden können. Die Zielfunktion bringt damit die abgetastete Abbildung der Umgebung und die zu schätzenden Bewegungsparameter in einen für einen Optimierer verständlichen, mathematischen Zusammenhang.

In Anlehnung an DRISF, definieren wir die Zielfunktion bzw. die Optimierungsaufgabe

über mehrere gewichtete Zielfunktionen zu:

$$\min_{\boldsymbol{\xi}} \Big\{ \underbrace{\lambda_{\text{photo}} E_{\text{photo}}(\boldsymbol{\xi};\mathcal{I}) + \lambda_{\text{rigid}} E_{\text{rigid}}(\boldsymbol{\xi};\mathcal{I}) + \lambda_{\text{flow}} E_{\text{flow}}(\boldsymbol{\xi};\mathcal{I})}_{\text{DRISF}} \underbrace{+\lambda_{\text{radar},s_d} E_{\text{radar}}(\boldsymbol{\xi};\mathcal{I})}_{\text{Erweiterung durch DRISFwR}} \Big\}. \tag{5.15}$$

Die obere Reihe beschreibt den photometrischen Fehler (orig.: „photometric error“) E_{photo}, die räumliche Kongruenz (orig.: „rigid fitting“) E_{rigid} und die Konsistenz des Flusses (orig.: „flow consistency“) E_{flow}. Diese Terme sind aus [MWH+19] bekannt. In der unteren Reihe wird die Konsistenz zur Radarmessung E_{radar} durch DRISFwR eingeführt. Die λ Terme sind positive Skalare zur Wichtung der einzelnen Zielfunktionen. Die einzelnen Zielfunktionen sind wie folgt mathematisch beschrieben.

5.7.2.1. Photometrischer Fehler

Bewegt sich ein Objekt im Raum, so verändert sich seine Position zwischen aufeinanderfolgenden Kamerabildern[3].

Die Transformation von Welt- in Bildkoordinaten ist bereits aus den Gleichung 3.23a und 3.24a bekannt. Durch Modifizierung der Gleichungen kann die Bewegung im Raum in die Bewegung im Kamerabild überführt werden. Dazu definieren wir analog zu Gleichung 5.14 zunächst die räumliche Verschiebung eines Punktes:

$$\tilde{\mathbf{x}}_C^{(k+1)} = \mathbf{x}_C^{(k)} + \boldsymbol{\xi}. \tag{5.16}$$

Mittels Kameralochbildmodell prädizieren wir die Pixelposition im neuen Frame $\tilde{\mathbf{p}}^{(k+1)}$ aus der Pixelposition des alten Frames $\mathbf{p}^{(k)}$ und $\boldsymbol{\xi}$ zu:

$$\tilde{\mathbf{p}}^{(k+1)} = \begin{bmatrix} \tilde{u}^{(k+1)} \\ \tilde{v}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \frac{\mathbf{x}_{C[x]}^{(k)} + \boldsymbol{\xi}_{[x]}}{\mathbf{x}_{C[z]}^{(k)} + \boldsymbol{\xi}_{[z]}} f_u + c_u \\ \frac{\mathbf{x}_{C[y]}^{(k)} + \boldsymbol{\xi}_{[y]}}{\mathbf{x}_{C[z]}^{(k)} + \boldsymbol{\xi}_{[z]}} f_v + c_v \end{bmatrix}. \tag{5.17}$$

Es sei angenommen, dass sich die Farbwerte des Punktes zwischen den Frames nicht ändern und der Bildinhalt ausschließlich durch die Bewegung verändert wird. Wird nun das zweite Bild mit dem korrekten Szenenfluss $\boldsymbol{\xi}$ verzerrt, d.h. die Pixel nach Gleichung 5.17 angeordnet, so sollten die Farbwerte sämtlicher Pixel des Ursprungsbildes mit denen des verzerrten Bildes übereinstimmen. Eine unerwünschte farbliche Abweichung wird durch den photometrischen Fehler $e_{\text{photo}} = \mathbf{G}^{(k+1)}_{\left[\tilde{\mathbf{p}}^{(k+1)}\right]} - \mathbf{G}^{(k)}_{\left[\mathbf{p}^{(k)}\right]}$ in einem Skalar zusammengefasst:

$$E_{\text{photo},i}(\boldsymbol{\xi};\mathcal{I}) = \sum_{\mathbf{p}_0 \in \mathcal{P}_i} \rho\Big(e_{\text{photo}}\Big) = \sum_{\mathbf{p}_0 \in \mathcal{P}_i} \rho\Big(\mathbf{G}^{(k+1)}_{\left[\tilde{\mathbf{p}}^{(k+1)}\right]} - \mathbf{G}^{(k)}_{\left[\mathbf{p}^{(k)}\right]}\Big). \tag{5.18}$$

[3] Unter Beachtung der Epipolargeometrie

Hierbei soll $\mathbf{G}^{(k+1)}_{\left[\tilde{\mathbf{p}}^{(k+1)}\right]}$ den Grauwert aus $\mathbf{G}^{(k+1)}$ an der Pixelposition $\tilde{\mathbf{p}}^{(k+1)}$ darstellen.

Das Grauwertbild $\mathbf{G} \in \mathbb{R}^{M \times N}$ wird aus dem Red-Green-Blue (RGB) Kamerabild $\mathbf{RGB} \in \mathbb{R}^{3 \times M \times N}$ transformiert:

$$\mathbf{G}_{[u,v]} = \Big[0.2989, 0.587, 0, 114\Big] \, \mathbf{RGB}_{[u,v]}, \tag{5.19}$$

dabei werden die einzelnen RGB Farbkanäle des Farbwertbildes gewichtet zum Grauwertbild summiert [Uni11]. M und N sind die Pixeldimension des Kamerabildes und u und v die Pixelposition eines ausgewählten Kamerapixels. Alternativ wäre eine homogene Gewichtung der Farbkanäle denkbar. Es wird jedoch erwartet, dass die Skalierung nach Gleichung 5.19 zu homogeneren Bildgradienten führt und somit bessere Optimierungseigenschaften aufweist. Eine empirische Untersuchung wird diesbezüglich nicht vorgenommen.

Maschinell effizient lässt sich die Differenz aus Gleichung 5.18 berechnen, indem das zweite Bild verzerrt wird. Dazu wird ein Gitter der Pixel $\mathbf{p}^{(k)}$ aufgebaut und dieses mit den Grauwerten des zweiten Bildes bei den korrespondierenden Pixeln $\tilde{\mathbf{p}}^{(k+1)}$ befüllt.

In Gleichung 5.18 wird die Farbwertdifferenz über eine robuste Fehlerfunktion ρ transformiert. Mit dieser Fehlerfunktion wird (a) die Differenz in ein konvexes Optimierungsproblem überführt und (b) der Einfluss von Ausreißern auf die gesamte Zielfunktion gewichtet. Im Laufe dieser Arbeit und analog zur originalen DRISF-Veröffentlichung findet hier die generalisierte Charbonnier-Funktion $\rho(x) = (x^2 + \epsilon^2)^\alpha$ Anwendung.

Die Zielfunktion ergibt sich aus der Akkumulation der skalaren Abweichungen aller als valide klassifizierten Pixel $\mathcal{P}_i$.

5.7.2.2. Räumliche Kongruenz

Wurden beim ersten Energieterm die Farbwertabweichungen der Pixel unter der Berücksichtigung der Bewegung gemessen, so wird bei der räumlichen Kongruenz die geometrische Abweichung der Punkte unter Berücksichtigung der Bewegung in kartesischen Koordinaten vermessen.

Wie in Gleichung 5.16 und 5.17 beschrieben, wird für jedes Pixel $\mathbf{p}^{(k)}$ die räumliche Position prädiziert und das korrespondierende Pixel $\tilde{\mathbf{p}}^{(k+1)}$ im neuen Frame geschätzt. Über den aus Abschnitt 4.5 bekannten optischen Fluss berechnen wir die assoziierte Pixelposition $\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + \mathbf{F}\left(\mathbf{p}^{(k)}\right)$ und extrahieren die gemessenen Koordinaten als $\left\{\mathbf{x}^{(k+1)}_{C[x]}, \mathbf{x}^{(k+1)}_{C[y]}, \mathbf{x}^{(k+1)}_{C[z]}\right\}$ aus den Sensordaten des zweiten Frames:

$$\begin{bmatrix} x^{(k+1)}_{C[x]}\left(\mathbf{p}^{(k+1)}\right) \\ x^{(k+1)}_{C[y]}\left(\mathbf{p}^{(k+1)}\right) \\ x^{(k+1)}_{C[z]}\left(\mathbf{p}^{(k+1)}\right) \end{bmatrix} = \begin{bmatrix} \frac{u^{(k+1)} - c_u}{f_u} x^{(k+1)}_{C[z]} \\ \frac{v^{(k+1)} - c_v}{f_v} x^{(k+1)}_{C[z]} \\ x^{(k+1)}_{C[z]} \end{bmatrix}, \tag{5.20}$$

wobei $u^{(k+1)}$, $v^{(k+1)}$ und $x^{(k+1)}_{C[z]}$ die Pixelkoordinaten und die Tiefenmessung an der Stelle $\mathbf{p}^{(k+1)}$ sind.

Die prädizierten Koordinaten $\left\{\tilde{x}^{(k+1)}_{C[x]}, \tilde{x}^{(k+1)}_{C[y]}, \tilde{x}^{(k+1)}_{C[z]}\right\}$ sollten idealerweise mit den

räumlichen Koordinaten im zweiten Frame $\left\{x_{C[x]}^{(k+1)}, x_{C[y]}^{(k+1)}, x_{C[z]}^{(k+1)}\right\}$ übereinstimmen. Als Maß für die Abweichung seien definiert

$$\mathbf{e}_{\text{rigid}}(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I}) = \begin{bmatrix} e_{\text{rigid}[x]}\left(\mathbf{p}^{(k)}, \mathbf{p}^{(k+1)}\right) \\ e_{\text{rigid}[y]}\left(\mathbf{p}^{(k)}, \mathbf{p}^{(k+1)}\right) \\ e_{\text{rigid}[z]}\left(\mathbf{p}^{(k)}, \mathbf{p}^{(k+1)}\right) \end{bmatrix} = \begin{bmatrix} x_{C[x]}^{(k+1)} - \tilde{x}_{C[x]}^{(k+1)} \\ x_{C[y]}^{(k+1)} - \tilde{x}_{C[y]}^{(k+1)} \\ x_{C[z]}^{(k+1)} - \tilde{x}_{C[z]}^{(k+1)} \end{bmatrix} = \begin{bmatrix} x_{C[x]}^{(k+1)} - x_{C[x]}^{(k)} - \xi_{[x]} \\ x_{C[y]}^{(k+1)} - x_{C[y]}^{(k)} - \xi_{[y]} \\ x_{C[z]}^{(k+1)} - x_{C[z]}^{(k)} - \xi_{[z]} \end{bmatrix}.$$

Der Energieterm für die räumliche Abweichung sei damit:

$$E_{\text{rigid},i}(\boldsymbol{\xi}; \mathcal{I}) = \sum_{p_0 \in \mathcal{P}_i} \left(\rho\left(e_{\text{rigid}[x]}\right) + \rho\left(e_{\text{rigid}[y]}\right) + \rho\left(e_{\text{rigid}[z]}\right)\right). \tag{5.21}$$

Wie beim photometrischen Fehler werden die Abweichungen durch die Anwendung der ρ Funktion konvex abgebildet.

5.7.2.3. Konsistenz des Flusses

Zuvor haben wir für jedes Pixel $\mathbf{p}^{(k)}$ im ersten Frame die durch die Bewegung veränderte neue Pixelposition $\tilde{\mathbf{p}}^{(k+1)}$ im neuen Frame geschätzt. Die Differenz der beiden Pixelpositionen $\left(\tilde{\mathbf{p}}^{(k+1)} - \mathbf{p}^{(k)}\right)$ entspricht dem in der Bildebene wirksamen Anteil des Szenenflusses und somit dem durch Szenenfluss verursachten optischen Fluss in der Bildebene. Diesen können wir mit dem aus Abschnitt 4.5 geschätzten optischen Fluss vergleichen. Die Abweichung $\mathbf{e}_{\text{flow}}$ sei definiert zu

$$\mathbf{e}_{\text{flow}}(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I}) = \begin{bmatrix} e_{\text{flow}[u]} \\ e_{\text{flow}[v]} \end{bmatrix} = \left(\tilde{\mathbf{p}}^{(k+1)} - \mathbf{p}^{(k)}\right) - \mathbf{F}_{[p]} = \begin{bmatrix} \left(\tilde{u}^{(k+1)} - u^{(k)}\right) - F_{u[\mathbf{p}]} \\ \left(\tilde{v}^{(k+1)} - v^{(k)}\right) - F_{v[\mathbf{p}]} \end{bmatrix}, \tag{5.22}$$

wobei $\mathbf{F}_u$ und $\mathbf{F}_v$ die zwei Komponenten des optischen Flusses entlang der Bildkoordinaten sind.

Der skalare Energieterm ergibt sich daraus zu:

$$E_{\text{flow},i}(\boldsymbol{\xi}; \mathcal{I}) = \sum_{p_1 \in \mathcal{P}_i} \rho(e_{\text{flow}[u]}) + \rho(e_{\text{flow}[v]}). \tag{5.23}$$

Der optische Fluss bildet ausschließlich die tangentiale Bewegung zur Bildebene ab, womit hier ausschließlich die tangentiale Bewegung optimiert wird[4].

5.7.2.4. Konsistenz der Radarmessung

Bereits in [MWH+19] wurde beschrieben, dass die Gütefunktionen E_{rigid} und E_{flow} in DRISF stark voneinander abhängig und einzig E_{photo} von der Qualität des optischen Flusses sind bzw. ist. Fehler im optischen Fluss haben somit starken Einfluss auf die

[4] Aussage gilt nur bei punktförmigen Zielen, da bei Veränderung der Distanz eine entsprechende Kontraktion im Bild wahrzunehmen ist.

Qualität des geschätzten Szenenflusses. Darüber hinaus sei ergänzt, dass der optische Fluss selbst häufig mittels zu E_{photo} vergleichbaren Metriken geschätzt wird. Etwaige Abweichungen in den Kameraabbildungen haben somit einen Einfluss auf alle drei Gütefunktionen. Die Abhängigkeit von der Qualität des optischen Flusses werden wir durch die Einführung einer vierten, von den Daten des Radars abhängigen, Gütefunktionen reduzieren.

Bevor wir in die mathematischen Details dieser neu eingeführten Gütefunktionen einsteigen, wollen wir das grobe Vorgehen anhand von Abbildung 5.5 darstellen und motivieren. In der oberen blauen Box ist ein Kamerabild und die RD-map aus einem typischen Fahrszenario dargestellt. Das Kamerabild stammt aus der nach hinten gerichteten Kamera und zeigt die Szene, in welcher sich das Fahrzeug mit einer typischen Geschwindigkeit entlang einer Straße bewegt. Im linken Bildteil sind Randbebauungen (Bordstein, Zaun, Büsche) zu sehen, in der Bildmitte zwei folgende Fahrzeuge, zur besseren Sichtbarkeit durch rote Boxen markiert und in einem separaten Ausschnitt vergrößert dargestellt. Diese beiden Fahrzeuge werden auch in der RD-map abgebildet und die entsprechenden Signaturen ebenfalls durch rote Boxen im RD-map gekennzeichnet. Verbindungslinien kennzeichnen die Assoziation der Fahrzeuge zwischen Kamerabild und RD-map. Es ist zu erkennen, dass die Signaturen der Fahrzeuge eine erhöhte Leistung aufweisen, gekennzeichnet durch hellere Pixel im RD-map und sich signifikant vom blau dargestellten Hintergrund unterscheiden. Diese Beobachtung ist typisch und bedingt durch die Reflektivität der Objekte, vgl. Gleichung 2.25. Wie später gezeigt wird, lässt sich der radiale Anteil der räumlichen Geschwindigkeit aus der Szenenflussschätzung berechnen, so dass Objekte aus Kamera und Lidar über ihre Entfernung und Radialgeschwindigkeit in das RD-map projiziert werden können. Diese Projektion ist in Abbildung 5.5 in der gelben Box unten für eines der Fahrzeuge bereits dargestellt. Dabei wurde eine initiale Geschwindigkeit angenommen, welche im linken Teil ($m = 0$) dazu führt, dass die rot dargestellte Signatur des Objektes nicht auf den hellen Pixeln im RD-map liegt, sondern auf Pixeln niedriger Leistung (blaue Pixel) daneben. Entsprechend unserer oben beschriebenen Beobachtung, dass Objekte durch helle Signaturen auffallen, ist diese Projektion also fehlerhaft, und intuitiv würden wir die Signatur nun linksseitig in die helleren Pixel verschieben. Diese Verschiebung ist durch die weiteren Signaturprojektionen ($m = 1$ und $m = 100$) dargestellt[5]. In der grauen Box wurden die Leistungswerte aus der RD-map in das Kamerabild projiziert. Diese Projektion werden wir in Kapitel 6 vorstellen. Analog zur Ausrichtung der Signatur in der RD-map kann beobachtet werden, dass die Helligkeit der Pixel an den Positionen der Objekte im Kamerabild bei $m = 100$ deutlich höher ist als bei $m = 0$ und der physikalischen Erwartung einer erhöhten Reflektivität von Objekten nachgekommen wird.

[5]Diese Verschiebungen wurden automatisch durch die integrative Optimierung des in dieser Arbeit vorgestellten DRISFwR Verfahrens erreicht.

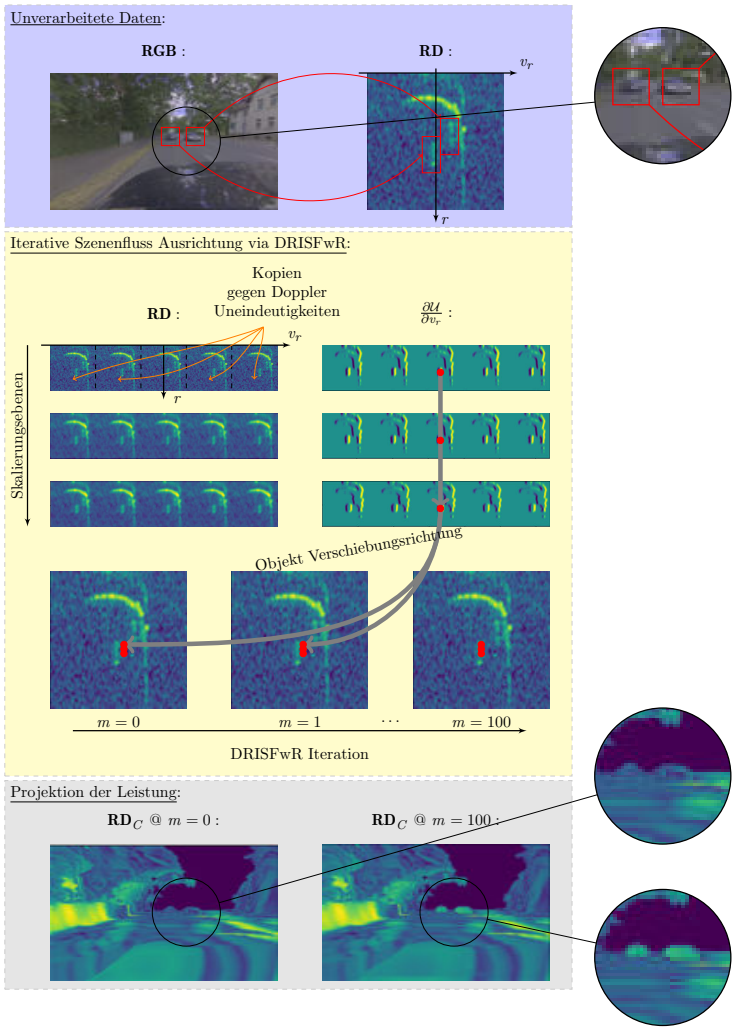


Abbildung 5.5.: **Automatische Ausrichtung vom Szenenfluss in der RD-map:** Blaue Box: Kamerabild und RD-map einer typischen Fahrszene. Rot gekennzeichnet sind zwei dem Ego-Fahrzeug folgende Fahrzeuge. Gelbe Box: Eine ungefähre Projektion eines der Fahrzeuge wurde unten links ($m = 0$) im RD-map eingetragen. An der Stelle der Projektion weist die Projektion kein lokales Leistungsmaximum auf und wird durch DRISFwR linksseitig im RD-map verschoben ($m = 1$, $m = 100$). Zur Steigerung der Robustheit gegenüber lokalen Maxima wurden Skalierungsebenen des RD-map mit unterschiedlicher Filterung verwendet. Graue Box: Projektion der Leistungen aus RD-map in Kamerabild. Es ist zu erkennen, dass nach $m = 100$ DRISFwR Iterationen eine erhöhte Leistung der Fahrzeugpixel zu beobachten ist. Nach [EB6].

Die in Abbildung 5.5 gezeigte Verschiebung der Signatur in der RD-map wird nun automatisiert. Dazu definieren wir zunächst die Projektion eines Pixels aus dem Kamerabild in die RD-map. Dem Pixel im Kamerabild sind aus der Tiefenschätzung (Abschnitt 4.1) und der hier durchzuführenden Szenenflussschätzung die kartesische Position $\mathbf{x}_C$ und die kartesische Geschwindigkeit $\boldsymbol{\xi}$ im Kamerakoordinatensystem be-

kannt. Mit den Koordinatentransformationen aus Gleichung 3.33 transformieren wir die Position aus Kamerakoordinaten in Radarkoordinaten.

Die Position der Projektion in der RD-map wird nach Abschnitt 2.1 durch die radiale Entfernung und die radiale Geschwindigkeit sowie die gewählte Modulation definiert. Die radiale Position ergibt sich aus $\mathbf{x}_R$ mit der euklidischen Norm zu

$$r(\mathbf{x}_R) = \|\mathbf{x}_R\|. \tag{5.24}$$

Es ist weiterhin bekannt, dass die Radialgeschwindigkeit eine Skalarprojektion der 3D-Geschwindigkeit auf den Radialvektor ist:

$$\dot{r}(\boldsymbol{\xi}_{\text{radar}}) = \frac{\mathbf{x}_R^T}{\|\mathbf{x}_R\|}\boldsymbol{\xi}_{\text{radar}}, \tag{5.25}$$

wobei hier $\boldsymbol{\xi}_{\text{radar}}$ die kartesische Geschwindigkeit im Radarkoordinatensystem ist. Diese entspricht der zeitliche Ableitung von $\mathbf{x}_R$ und kann daher mit der Transformation aus Gleichung 3.33 und dem Szenenfluss $\boldsymbol{\xi}$ berechnet werden zu

$$\begin{aligned}\boldsymbol{\xi}_{\text{radar}} &= \frac{\partial \mathbf{x}_R}{\partial t} = {}^{R\leftarrow C}\mathbf{R}\frac{\partial \mathbf{x}_C}{\partial t} \\ &= {}^{R\leftarrow C}\mathbf{R}\boldsymbol{\xi}.\end{aligned} \tag{5.26}$$

Für jedes Pixel im Kamerabild kann mit Hilfe von Gleichung 2.18 und 2.21 die Pixelposition in der RD-map berechnet werden zu

$$\mathbf{p}_{R[\mathbf{p}]} = \begin{bmatrix} u_{R[\mathbf{p}]} \\ v_{R[\mathbf{p}]} \end{bmatrix} = \begin{bmatrix} \dot{r}\left(\left(\boldsymbol{\xi}_{\text{radar}[\mathbf{p}]}\right)/\Delta v\right) \\ r\left(\mathbf{x}_{R[\mathbf{p}]}\right)/\Delta r \end{bmatrix}. \tag{5.27}$$

Der Leistungswert eines Pixels in der RD-map ergibt sich damit zu $RD_{[\mathbf{p}_R]}$.

Nachdem die Projektion in die RD-map damit mathematisch beschrieben ist, wollen wir uns nun der Gütefunktion zuwenden. Bei der Beschreibung von Abbildung 5.5 wurde beobachtet, dass die beiden verfolgenden Fahrzeuge eine erhöhte Leistungssignatur in der RD-map aufweisen und vom Hintergrundrauschen klar zu trennen sind. Die initiale Projektion der Objekte lag allerdings auf dem Hintergrundrauschen. Intuitiv haben wir aus der Beobachtung heraus definiert, dass die initiale Projektion damit fehlerhaft ist. Diesen Fehler wollen wir jetzt messen. Dazu definieren wir die Abweichung der Leistungsprojektion gegenüber einem festzulegenden Zielwert U_{target} als

$$e_{\text{radar}}(\mathbf{p}_R; \mathcal{I}) = U_{\text{target}} - RD_{[\mathbf{p}_R]}. \tag{5.28}$$

Wir legen fest, dass $e_{\text{radar}}(\mathbf{p}; \mathcal{I})$ minimiert werden soll, wenn die Projektion auf einem (lokalen) Leistungsmaximum liegt. U_{target} muss somit dem (lokalen) Leistungsmaximum in der RD-map entsprechen. Dieses Maximum mag stark von Verkehrsteilnehmern und Szenario abhängen. So haben Fußgänger statistisch ein geringeres RCS als LKW und weiter entfernte Ziele eine geringere Signalleistung als näher gelegene, vgl. Gl.2.25. Wünschenswert wäre nun, den Wert von U_{target} entsprechend variieren zu können. Es wäre denkbar, aus den Daten der Instanzsegmentierung und Tiefenmaske den statistisch wahrscheinlichen Zielwert der Leistung für unterschiedliche Verkehrsteilnehmer zu

schätzen. Dies würde den Umfang dieser Arbeit jedoch merklich übertreffen. Der Einfachheit halber werden wir als Zielwert U_{target} den Maximalwert des RD-maps übernehmen und annehmen, dass die initiale Projektion der Objekte in die RD-map gut ist und zumindest in der Pixelnachbarschaft der tatsächlichen Signatur liegt, so dass der Optimierer das lokale Maximum findet. In Unterabschnitt 5.7.7 werden wir spezielle Modifikationen der Kosten vorstellen, u.a. um diese Konvergenz zu steuern. Diese speziellen Modifikationen sind textuell absichtlich getrennt, da sie zum allgemeinen Verständnis von DRISFwR zunächst nicht benötigt werden.

Um die Residuen konvex zu machen, wenden wir, wie zuvor auch, die generalisierte Charbonnier-Funktion an und definieren den Energieterm

$$E_{\text{radar}}(\boldsymbol{\xi};\mathcal{I}) = \sum_{\mathbf{p}\in P_i} \rho\left(e_{\text{radar}}(\mathbf{p};\mathcal{I})\right) \tag{5.29}$$

5.7.3. Lösung mittels Gauß-Newton Optimierer

Nachdem wir im Abschnitt 5.5 die gesuchten Bewegungsparameter und in Unterabschnitt 5.7.2 die Zielfunktionen definiert haben, können wir uns nun der Inferenz der Parameter zuwenden. Analog zu DRISF [MWH$^+$19] werden wir auch hier einen Gauß-Newton (GN)-Optimierer zur Minimierung der nichtlinearen Zielfunktion (Gleichung 5.15) verwenden. Einige der Herleitungen für die speziellen DRISF-Energieterme sind in der erweiterten Veröffentlichung von DRISF [MWH$^+$19] zu finden und werden hier zum einfachen Verständnis für den Leser wiedergegeben. Zur Honorierung der wissenschaftlichen Vorarbeit durch DRISF sei dem Leser die Durchsicht der oben genannten Veröffentlichung nahegelegt.

Wie in [MWH$^+$19] gezeigt wurde, lässt sich die Minimierung der Energiefunktionen mittels GN-Schätzer durchführen. Jeder der Energieterme hat eine Form analog zu:

$$E(\boldsymbol{\xi};\mathcal{I}) = \sum_{\mathbf{p}\in P_i} \rho\left(e(\boldsymbol{\xi},\mathbf{p};\mathcal{I})\right), \tag{5.30}$$

in welche $\mathbf{p}$ die Pixel zu einer Instanz sind und $e(\dots)$ eine skalare Abweichung für das Pixel darstellt. Gesucht wird die Transformation $\hat{\boldsymbol{\xi}}$, welche die Energiefunktion minimiert

$$\hat{\boldsymbol{\xi}} = \arg\min_{\boldsymbol{\xi}} E(\boldsymbol{\xi};\mathcal{I}) = \arg\min_{\boldsymbol{\xi}} \sum_{\mathbf{p}\in P_i} \rho\left(e(\boldsymbol{\xi},\mathbf{p};\mathcal{I})\right). \tag{5.31}$$

Der Verlauf der Residuen $e = \{e_{\text{photo}}, e_{\text{rigid}}, \mathrm{e}_{\text{flow}}, e_{\text{radar}}\}$ ist nicht linear über den Szenenfluss, der somit nicht analytisch gefunden werden kann. Durch Linearisierung der Residuenfunktion kann ein iteratives Suchen nach lokalen Minima umgesetzt werden. Die linearisierte Residuenfunktion ergibt sich zu

$$e'(\Delta\boldsymbol{\xi},\mathbf{p};\mathcal{I}) \coloneqq e(\boldsymbol{\xi}+\Delta\boldsymbol{\xi},\mathbf{p};\mathcal{I}) \approx e(\boldsymbol{\xi},\mathbf{p};\mathcal{I}) + \frac{\partial e(\boldsymbol{\xi},\mathbf{p};\mathcal{I})}{\partial\boldsymbol{\xi}} \cdot \Delta\boldsymbol{\xi}. \tag{5.32}$$

Durch Einsetzten der Linearisierung in Gleichung 5.31 und der Definition der „Jacobi-Matrix“ $\mathbf{J}_p \in \mathbb{R}^{1\times 3}$ der Residuen

$$\mathbf{J}_p = \frac{\partial e(\boldsymbol{\xi},\mathbf{p};\mathcal{I})}{\partial\boldsymbol{\xi}} \tag{5.33}$$

ergibt sich das Optimierungsproblem

$$\begin{aligned}
\Delta\boldsymbol{\xi} &= \arg\min_{\boldsymbol{\xi}} \sum_{\mathbf{p}\in P_i} \rho\left(e'(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I})\right) \\
&= \arg\min_{\boldsymbol{\xi}} \sum_{\mathbf{p}\in P_i} \tau\left(e'(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I})^T e'(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I})\right) \\
&= \arg\min_{\boldsymbol{\xi}} \sum_{\mathbf{p}\in P_i} \tau\left((e(\boldsymbol{\xi}, \mathbf{p}; \mathcal{I}) + \mathbf{J}_p \Delta\boldsymbol{\xi})^T (e(\boldsymbol{\xi}, \mathbf{p}; \mathcal{I}) + \mathbf{J}_p \Delta\boldsymbol{\xi})\right). \qquad (5.34)
\end{aligned}$$

Hierbei wird $\rho(x) = \tau(x^T x)$ verwendet. Wir definieren zusätzlich

$$L_p = e'(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I})^T e'(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I}). \qquad (5.35)$$

Das Minimum ergibt sich an der Stelle des verschwindenden Gradienten zu:

$$\begin{aligned}
0 &\stackrel{!}{=} \frac{\partial E}{\partial \Delta\boldsymbol{\xi}} \\
&= \sum_{\mathbf{p}\in P_i} \frac{\partial \tau(L_p)}{\partial L_p} \frac{\partial L_p}{\partial e'_p} \frac{\partial e'_p}{\partial \Delta\boldsymbol{\xi}} \\
&= \sum_{\mathbf{p}\in P_i} \frac{\partial \tau(L_p)}{\partial L_p} 2(e'_p)^T \mathbf{J}_p \\
&= \sum_{\mathbf{p}\in P_i} \frac{\partial \tau(L_p)}{\partial L_p} 2(e(\boldsymbol{\xi}, \mathbf{p}; \mathcal{I}) + \mathbf{J}_p \Delta\boldsymbol{\xi})^T \mathbf{J}_p. \qquad (5.36)
\end{aligned}$$

Die rechte Seite entspricht der Ableitung der linearisierten Energiesumme aus Gleichung 5.34. Die Aufteilung der Gradienten erfolgt durch Anwendung der Kettenregel.

Umstellen der letzten Zeile ergibt

$$\sum_{\mathbf{p}\in P_i} \frac{\partial \tau(L_p)}{\partial L_p} \mathbf{J}_p^T \mathbf{J}_p \Delta\boldsymbol{\xi} = -\sum_{\mathbf{p}\in P_i} \frac{\partial \tau(L_p)}{\partial L_p} \mathbf{J}_p^T e(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I}) \qquad (5.37)$$

was mit

$$W_p = \frac{\partial \tau(L_p)}{\partial L_p} \qquad (5.38)$$

nach $\Delta\boldsymbol{\xi}$ aufgelöst werden kann:

$$\Delta\boldsymbol{\xi} = -\left(\sum_{\mathbf{p}\in P_i} \mathbf{J}_p^T W_p \mathbf{J}_p\right)^{-1} \left(\sum_{\mathbf{p}\in P_i} \mathbf{J}_p^T W_p e(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I})\right). \qquad (5.39)$$

Nach jedem Iterationsdurchlauf wird der Szenenfluss aktualisiert zu

$$\boldsymbol{\xi}^{(m)} = \boldsymbol{\xi}^{(m-1)} + \Delta\boldsymbol{\xi}, \qquad (5.40)$$

wobei $\boldsymbol{\xi}^{(m)}$ der Szenenfluss im m-ten Iterationsschritt ist.

5.7.4. Herleitung der Residuen

Zur Inferenz der Parameter nach Gleichung 5.37 werden unter anderem die Residuen $e(\dots)$ benötigt. Der besseren Übersicht halber werden die Residuen für die einzelnen Energieterme hier wiederholt.

5.7.4.1. Photometrischer Fehler

Die Residuen für den photometrischen Fehler entsprechen dem Helligkeitsunterschied der Kamerapixel

$$e_{\text{photo}} = \mathbf{G}^{(k+1)}_{\left[\tilde{\mathbf{p}}^{(k+1)}\right]} - \mathbf{G}^{(k)}_{\left[\mathbf{p}^{(k)}\right]}. \tag{5.41}$$

Da das Grauwertbild $\mathbf{G}$ nur einen Kanal hat, ist die Dimension der Residuen per Pixel 1×1.

5.7.4.2. Räumliche Kongruenz

Die Residuen der räumlichen Kongruenz entsprechen dem Abstand, welcher sich aus der Differenz der kartesischen Positionen eines Pixels im Kamerabild und seiner um den Szenenfluss korrigierten Korrespondenz im nächsten Kameraframe ergibt. In Gleichung 5.21 wurde $E_{\text{rigid},i}$ als homogene Summe der Abweichungen in den drei Raumrichtungen zusammengefasst. Mathematisch äquivalent lässt sich dies als drei einzelne Energieterme auffassen und die Residuen für den Optimierer als Vektor darstellen:

$$\mathbf{e}_{\text{rigid}}(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I}) = \begin{bmatrix} x^{(k+1)}_{C[x]} - x^{(k)}_{C[x]} - \xi_{[x]} \\ x^{(k+1)}_{C[y]} - x^{(k)}_{C[y]} - \xi_{[y]} \\ x^{(k+1)}_{C[z]} - x^{(k)}_{C[z]} - \xi_{[z]} \end{bmatrix} \tag{5.42}$$

Die Dimension der Residuen ist 3×1, da der Positionsvektor drei Elemente hat.

5.7.4.3. Konsistenz des Flusses

Die Residuen zur Konsistenz des Flusses ergeben sich aus dem auf die Bildebene projizierten Szenenfluss und dem vermessenen optischen Fluss:

$$\mathbf{e}_{\text{flow}}(\Delta\boldsymbol{\xi}, \mathbf{p}; \mathcal{I}) = \begin{bmatrix} \left(\tilde{u}^{(k+1)} - u^{(k)}\right) - F_{u[\mathbf{p}]} \\ \left(\tilde{v}^{(k+1)} - v^{(k)}\right) - F_{v[\mathbf{p}]} \end{bmatrix}. \tag{5.43}$$

Die Dimension der Residuen ist 2×1, da der optische Fluss zwei Elemente hat.

5.7.4.4. Konsistenz der Radarmessung

Die Residuen zur Radarmessung ergibt sich aus dem Leistungswert eines Pixels in der RD-map $RD_{[\mathbf{p}_R]}$ und einem parametrierten Zielwert U_{target} zu:

$$e_{\text{radar}}(\mathbf{p}_R; \mathcal{I}) = U_{\text{target}} - RD_{[\mathbf{p}_R]}. \tag{5.44}$$

Die Dimension der Residuen ist 1×1, da das RD-map $\mathbf{RD}$ nur einen Kanal hat.

5.7.5. Herleitung der Jacobi-Matrizen

Für die Optimierung mittels GN-Löser sind nach Gleichung 5.34 die partiellen Ableitungen der Residuen entlang aller Komponenten des Szenenflusses in Form von Jacobi-Matritzen notwendig. Diese werden nun für alle Energieterme in diesem Abschnitt vorgestellt.

5.7.5.1. Photometrischer Fehler

Für den photometrischen Fehler ergibt sich nach [MWH+19]

$$
\begin{aligned}
\mathbf{J}_{\text{photo.}} &= \frac{\partial e_{\text{photo.}}(\boldsymbol{\xi}, \mathbf{p}; \mathcal{I})}{\partial \boldsymbol{\xi}} \\
&= \frac{\partial \left(\mathbf{G}^{(k+1)}_{\left[\tilde{\mathbf{p}}^{(k+1)}\right]} - \mathbf{G}^{(k)}_{\left[\mathbf{p}^{(k)}\right]} \right)}{\partial \boldsymbol{\xi}} \\
&= \frac{\partial \mathbf{G}^{(k+1)}}{\partial \tilde{\mathbf{p}}^{(k+1)}} \frac{\partial \tilde{\mathbf{p}}^{(k+1)}}{\partial x} \frac{\partial x}{\partial \boldsymbol{\xi}} - \frac{\partial \mathbf{G}^{(k)}}{\partial \boldsymbol{\xi}} \\
&\approx \left[\frac{\partial}{\partial u} \mathbf{G}^{(k+1)}_{\left[\tilde{\mathbf{p}}^{(k+1)}\right]}, \frac{\partial}{\partial v} \mathbf{G}^{(k+1)}_{\left[\mathbf{p}^{(k+1)}\right]} \right] \begin{bmatrix} \frac{f_u}{x^{(k)}_{C[z]}}, 0, -\frac{x^{(k)}_{C[x]} f_u}{\left(x^{(k)}_{C[z]}\right)^2} \\ 0, \frac{f_v}{x^{(k)}_{C[z]}}, -\frac{x^{(k)}_{C[y]} f_v}{\left(x^{(k)}_{C[z]}\right)^2} \end{bmatrix} \begin{bmatrix} 1, 0, 0 \\ 0, 1, 0 \\ 0, 0, 1 \end{bmatrix},
\end{aligned} \tag{5.45}
$$

wobei $\mathbf{G}^{(k)}_{\left[\mathbf{p}^{(k)}\right]}$ der Farb-/Grauwert im ersten Frame ist und nicht vom geschätzten Szenenfluss abhängt, wodurch $\frac{\partial \mathbf{G}^{(k)}}{\partial \boldsymbol{\xi}} = 0$ ist. Der Gradient von $\mathbf{G}^{(k+1)}_{\left[\tilde{\mathbf{p}}^{(k+1)}\right]}$ dagegen wurde durch Anwendung der Kettenregel in die weiteren Gradienten aufgeteilt. Es wird zunächst der Farb- bzw. Helligkeitsgradient des Bildes in Abhängigkeit der Pixeländerung berechnet und danach der Gradient von Pixelposition in Abhängigkeit zur kartesischen Position (siehe Gleichung 5.17). Beim zweiten Term wurden die Abhängigkeiten der partiellen Ableitungen untereinander vernachlässigt, womit alle Ableitungen effizient berechnet werden können. Gegenüber DRISF wurde angepasst, dass nun ausschließlich drei Translationen zu schätzen sind. Die Dimension der Matrix ist 1×3, da das Grauwertbild $\mathbf{G}$ einen Kanal und der gesuchte Bewegungsvektor $\boldsymbol{\xi}$ drei Elemente hat.

5.7.5.2. Räumliche Kongruenz

Für die räumliche Kongruenz ergibt sich

$$\begin{aligned}\mathbf{J}_{\text{rigid}} &= \frac{\partial \mathbf{e}_{\text{rigid.}}(\boldsymbol{\xi}, \mathbf{p}; \mathcal{I})}{\partial \boldsymbol{\xi}} \\ &= \frac{\partial \begin{bmatrix} x_{C[x]}^{(k+1)} - x_{C[x]}^{(k)} - \xi_{[x]} \\ x_{C[y]}^{(k+1)} - x_{C[y]}^{(k)} - \xi_{[y]} \\ x_{C[z]}^{(k+1)} - x_{C[z]}^{(k)} - \xi_{[z]} \end{bmatrix}}{\partial \boldsymbol{\xi}} \\ &= -\begin{bmatrix} 1,0,0 \\ 0,1,0 \\ 0,0,1 \end{bmatrix}. \end{aligned} \tag{5.46}$$

Die Terme $\mathbf{x}^{(k+1)}$ und $\mathbf{x}^{(k)}$ sind invariant gegenüber dem Szenenfluss $\boldsymbol{\xi}$. Es verbleibt die partielle Ableitung des Szenenflusses gegenüber sich selbst, was unter Schätzung von ausschließlich translatorischen Komponenten zur (negativen) Einheitsmatrix führt. Die Dimension der Matrix ist 3×3, da die Position drei Elemente hat, ebenso wie der gesuchte Bewegungsvektor $\boldsymbol{\xi}$.

5.7.5.3. Konsistenz des Flusses

Für die Flusskonsistenz ergibt sich die Jacobi-Matrix zu

$$\begin{aligned}\mathbf{J}_{\text{flow}} &= \frac{\partial \mathbf{e}_{\text{flow}}(\boldsymbol{\xi}, \mathbf{p}; \mathcal{I})}{\partial \boldsymbol{\xi}} \\ &= \frac{\partial \begin{bmatrix} \left(\tilde{u}^{(k+1)} - u^{(k)}\right) - \mathbf{F}_{u[\mathbf{p}]}) \\ \left(\tilde{v}^{(k+1)} - v^{(k)}\right) - \mathbf{F}_{v[\mathbf{p}]}) \end{bmatrix}}{\partial \boldsymbol{\xi}} \\ &\approx \begin{bmatrix} \frac{f_u}{\mathbf{x}_{C[z]}^{(k)}}, 0, -\frac{\mathbf{x}_{C[x]}^{(k)} f_u}{\left(\mathbf{x}_{C[z]}^{(k)}\right)^2} \\ 0, \frac{f_v}{\mathbf{x}_{C[z]}^{(k)}}, -\frac{\mathbf{x}_{C[y]}^{(k)} f_v}{\left(\mathbf{x}_{C[z]}^{(k)}\right)^2} \end{bmatrix} \begin{bmatrix} 1,0,0 \\ 0,1,0 \\ 0,0,1 \end{bmatrix}. \end{aligned} \tag{5.47}$$

Dies entspricht gerade den letzten beiden Termen der Gradientenkette des photometrischen Fehlers (siehe Gleichung 5.45) und muss bei einer effizienten Implementierung nicht separat neu berechnet werden. Die Dimension der Matrix ist 2×3, da der optische Fluss $\mathbf{F}$ zwei Kanäle hat, der gesuchte Bewegungsvektor $\boldsymbol{\xi}$ drei Elemente.

5.7.5.4. Konsistenz der Radarmessung

Die Jacobi-Matrix zur Konsistenz der Radarmessung ergibt sich zu

$$\begin{aligned}
\mathbf{J}_{\text{radar}} &= \frac{\partial e_{\text{radar}}(\boldsymbol{\xi}, \mathbf{p}; \mathcal{I})}{\partial \boldsymbol{\xi}} \\
&= \frac{\partial \left(U_{\text{target}} - RD_{[\mathbf{p}_R]} \right)}{\partial \boldsymbol{\xi}} \\
&= -\frac{\partial RD_{[\mathbf{p}_R]}}{\partial \mathbf{p}_{R[\mathbf{p}]}} \frac{\partial \mathbf{p}_{R[\mathbf{p}]}}{\partial \boldsymbol{\xi}_{\text{radar}}(\boldsymbol{\xi})} \frac{\partial \boldsymbol{\xi}_{\text{radar}}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} \\
&= -\nabla_{v_r} RD_{[\mathbf{p}_R]} \frac{\mathbf{x}_R^T}{\|\mathbf{x}_R\|} {}^{R\leftarrow C}\mathbf{R}.
\end{aligned} \tag{5.48}$$

Die Dimension der Matrix ist 1×3, da das RD-map **RD** einen Kanal hat, der gesuchte Bewegungsvektor $\boldsymbol{\xi}$ drei Elemente.

Analog zu den anderen Energiefunktionen wurde auch hier die Kettenregel verwendet und so der Gesamtgradient als Produkt einzelner interpretierbarer Terme aufgeteilt. Der erste Term entspricht der Abhängigkeit des RD-maps vom Radarpixel $\mathbf{p}_R$. Da durch den Szenenfluss nur der Radialgeschwindigkeitsanteil von $\mathbf{p}_R$ modifiziert werden kann, entspricht dies somit dem Bildgradienten entlang der Radialgeschwindigkeit. Der Bildgradient wird durch Faltung mit dem Faltungskern $\nabla_{v_r} = [-1, 0, 1]$ in Dopplerrichtung des RD-maps berechnet. Dabei werden das Pixel und seine horizontalen Nachbarn mit dem Faltungskern kombiniert und es ergibt sich für jedes Pixel ein skalarer Wert. Zur Veranschaulichung ist der Bildgradient in Abbildung 5.5 (gelbe Box, rechts) dargestellt.

Der zweite Term entspricht der linearen Abhängigkeit der Radialgeschwindigkeit vom Szenenfluss im Radar-Koordinatensystem. Die Transformation von Szenenfluss im Radar-Koordinatensystem in Radialgeschwindigkeit ist aus Gleichung 5.25 bekannt. Der Gradient ergibt sich durch Ableitung dieser Gleichung nach $\boldsymbol{\xi}_{\text{radar}}$ zu

$$\frac{\partial \mathbf{p}_{R[\mathbf{p}]}}{\partial \boldsymbol{\xi}_{\text{radar}}(\boldsymbol{\xi})} = \frac{\partial \dot{r}(\boldsymbol{\xi}_{\text{radar}})}{\partial \boldsymbol{\xi}_{\text{radar}}} = \frac{\mathbf{x}_R^T}{\|\mathbf{x}_R\|}. \tag{5.49}$$

Die Variation des Szenenflusses hat keine Auswirkung auf den Abstand r in $\mathbf{p}_R$, siehe Gleichung 5.27. In der RD-map wird durch den Szenenfluss schließlich auch nur eine Verschiebung entlang der Doppler-Richtung vollzogen. Folglich ist dieser Gradient Null und hat keinen Einfluss auf die Optimierung. Zu Gunsten einer besseren Übersicht wurde dieser Gradient nicht in Gleichung 5.49 dargestellt. Die Dimension dieses Teilgradienten ist 1×3.

Der letzte Teilgradient aus Gleichung 5.48 ergibt sich aus der Ableitung von Gleichung 5.26 nach $\boldsymbol{\xi}$ zu

$$\frac{\partial \boldsymbol{\xi}_{\text{radar}}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} = \frac{\partial {}^{R\leftarrow C}\mathbf{R}\boldsymbol{\xi}}{\partial \boldsymbol{\xi}} = {}^{R\leftarrow C}\mathbf{R} \tag{5.50}$$

und hat die Dimension 3×3.

5.7.6. Herleitung der Wichtungsmatrizen

Dem Beispiel von DRISF folgend, wird für alle Energieterme die generalisierte Charbonnier-Funktion [SRB10] verwendet

$$\rho(x) = \left(x^2 + \epsilon^2\right)^{\alpha}. \tag{5.51}$$

Hierdurch wird die Energiefunktion für den Optimierer konvex und die Residuen gewichtet. Analog zu DRISF wurde auch hier $\epsilon = 10^{-6}$ und $\alpha = 0.45$ gewählt.

Nach Gleichung 5.38 ist die Wichtungsmatrix definiert als das Produkt aus Wichtungsfaktor λ und der Ableitung der gewählten robusten Kostenfunktion nach den quadratischen Residuen, welche sich zu

$$\frac{\partial \tau(L_p)}{\partial L_p} = \frac{\partial (L_p + \epsilon^2)^{\alpha}}{\partial L_p} = \alpha (L_p + \epsilon^2)^{\alpha - 1} \tag{5.52}$$

ergibt. Am Beispiel der Zielfunktion für Radar ergibt sich somit eine Wichtungsmatrix

$$\begin{aligned} W_{\text{radar}} &= \lambda_{\text{radar}} \frac{\partial \rho^2}{\partial r^2_{\text{radar}}} \\ &= 0.45 \lambda_{\text{radar}} \left(e^2_{\text{radar}} + 10^{-6}\right)^{-0.55}. \end{aligned} \tag{5.53}$$

Die Wichtungsmatrizen der anderen Energieterme ergeben sich analog.

Für die effiziente numerische Optimierung aller Energieterme können die Residuen der Energieterme in einem Vektor und die Jacobi-Matrizen zwecks numerischer Verarbeitung in Matrizenform konkateniert werden. Die Gesamtwichtungsmatrix ist dann als Diagonalmatrix zu bilden. Entsprechend ihrer Auflistung in dieser Arbeit haben die Energieterme 1, 2, 3 und 1 Element/e. Daraus ergibt sich die Dimension der Gesamtrichtungsmatrix zu 7×7.

5.7.7. Spezielle Modifikationen von DRISFwR

In den Abschnitten zuvor wurden die Erweiterungen von DRISF zu DRISFwR durch die Aufnahme des Energieterms „Konsistenz der Radarmessung“ beschrieben. Um die Inferenz weiter zu steuern, wurden weitere Modifikationen vorgenommen. Diese werden in diesem Abschnitt erläutert.

5.7.7.1. Berücksichtigung von Doppler-Mehrdeutigkeit

In Gleichung 5.54 wurde festgelegt, dass die RD-map **RD** zur Messung von Residuen verwendet wird. Dieses entsteht durch zweifache Fouriertransformation auf den abgetasteten Zeitsignalen, vgl. Unterunterabschnitt 2.1.3.4. Durch Verwendung von Fouriertransformation ergibt sich ein Eindeutigkeitsintervall der Geschwindigkeit, und Objekte mit höherer Geschwindigkeit wickeln sich auf der anderen Seite des Geschwindigkeitsspektrums wieder in das RD-map hinein, vgl. Unterunterabschnitt 2.1.3.5. Dieses Phänomen ist allgemein als „Alias-Effekt“ bekannt. Damit die Projektion von Kamerabildern in Radardaten entsprechend Abstand und Geschwindigkeit bei der Szenenflussschätzung plausibel ist, wurde in Abbildung 5.5 das RD-map mehrfach fortgesetzt, so dass beim Überschreiten des eindeutigen Geschwindigkeitsintervalls der oben beschriebene Effekt

berücksichtigt wird. Ein äquivalentes Ergebnis, aber effizientere Implementierung ergibt sich, wenn die geschätzten Pixelpositionen in der RD-map nach Gleichung 5.27 entsprechend der Bildgröße des RD-maps durch Modulo-Operation projiziert werden. Zur grafischen Hervorhebung für den Leser wurde im Rahmen dieser Arbeit jedoch erster Vorschlag implementiert.

5.7.7.2. Virtuelle Reduktion der Auflösung des RD-maps

Wir haben im Laufe dieses Kapitels gesehen, dass ein iterativer Optimierer zur Schätzung des Szenenflusses verwendet wird, welcher das in die RD-map projizierte Objekt in Richtung der Radialgeschwindigkeit verschiebt und dabei versucht, den Zielwert U_{target} aus Gleichung 5.54 der Leistung zu erreichen. Das Bestreben des Optimierers, das projizierte Objekt weiter zu verschieben, hängt insbesondere auch vom Bildgradienten in der RD-map bzw. der horizontalen Helligkeitsänderung ab. Liegt das projizierte Objekt in einem Bereich mit hohem Bildgradienten, hat der Optimierer das Bestreben, den geschätzten Szenenfluss anzupassen und das Objekt im RD-map zu verschieben. Bei verschwindenden Gradienten hat der Optimierer entsprechend kein Bestreben zur Verschiebung. Die Gradienten des RD-maps beim Übergang von Hintergrundrauschen zu Objekten sind häufig sehr steil, erstrecken sich jedoch nur über eine kleine Pixelnachbarschaft um das Objekt, entsprechend des Auflösungsvermögens des Radars. Liegt ein projiziertes Objekt fälschlicherweise außerhalb dieses Bereichs, so stagniert die Optimierung aufgrund des flachen Bildgradienten. Um diese Stagnation zu unterbinden, wird ein Skalierungsraum[6] auf der RD-map gebaut. Das RD-map wird durch mehrfache Anwendung von Gauß-Filter geglättet und die Auflösung des RD-map virtuell durch Anwendung von Max-Pooling Filter reduziert. Durch diese Glättung und Reduktion der Auflösung wird der Bildgradient in einen größeren Bereich um das lokale Bildmaximum verschoben. Zur Veranschaulichung wurden in Abbildung 5.6 ein originales RD-map und zwei Skalierungsebenen eingezeichnet.

In der Abbildung sind die gefilterten RD-maps sowie die Bildgradienten dargestellt. Für den optischen Vergleich wurden die Ergebnisse der Skalierungsebenen durch bilineare Interpolation auf die Bildgröße des originalen RD-maps hochskaliert. Insbesondere in den unteren Skalierungseben ist zu erkennen, dass der Bildgradient bei den lokalen Maxima weiter nach außen verschoben wird als beim ursprünglichen RD-map.

Die Berücksichtigung der Skalierungsebenen wurde in Gleichung 5.15 durch den Parameter s_d angedeutet. Mit $s_d = 1$ bezeichnen wir das originale RD-map und mit $s_d > 1$ die Skalierungsebenen. Diese Modifikation wird durch Anpassung der Gleichung 5.54 und 5.55 vorgenommen. Hierbei wurde U_{s_d} zur Kennung des RD-maps auf der Skalierungsebene s_d verwendet. Im Rahmen dieser Arbeit wurden drei Skalierungsebenen verwendet. Wie in Abbildung 5.6 gezeigt, wurde der Fangbereich damit ungefähr verdoppelt. Eine Optimierung oder tiefere Analyse der Skalierungsebenen wird im Rahmen dieser Arbeit nicht durchgeführt.

$$e_{\text{radar}}(\mathbf{p}_R; \mathcal{I}, s_d) = U_{\text{target}} - RD_{s_d[\mathbf{p}_R]}. \tag{5.54}$$

[6] Skalierungsräume finden häufig in der Schätzung des optischen Flusses Verwendung, siehe z.B. [TBKP12].

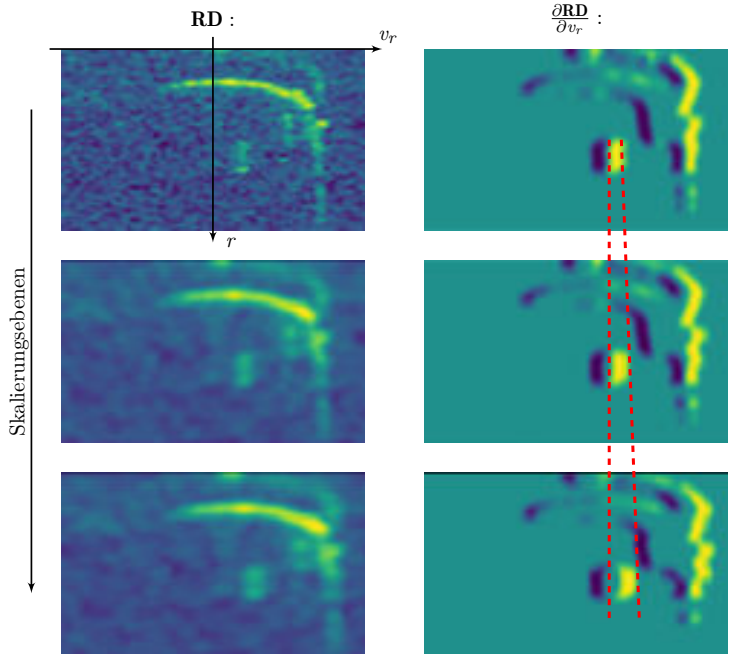


Abbildung 5.6.: **Skalierungsebenen auf der RD-map zur Vergrößerung des Fangbereichs:** Das originale RD-map (oben links), wird durch zweifaches Anwenden von Gauß-Filter, Max-Pooling und bilinearer-Interpolation verschmiert. Durch die Vierschmierung flacht der Bildgradient ab, dehnt sich jedoch über einen weiteren Bildbereich aus. Die zu den Skalierungen gehörenden Bildgradienten sind rechts dargestellt. Die Farbskalierung ist blau für negative Gradienten, gelb für positive Gradienten und grün für neutrale Gradienten. Man beachte hierbei insbesondere die erwähnte Ausdehnung der Bildgradienten über die Skalierungsebenen. Zur Verdeutlichung wurden rote gestrichelte Hilfslinien für einen ausgewählten Bereich der Gradienten eingezeichnet. Deutlich zu erkennen ist, wie sich die Hilfslinien voneinander distanzieren.

$$E_{\text{radar}}(\boldsymbol{\xi}; \mathcal{I}) = \sum_{s_d=1}^{3} \sum_{\mathbf{p}_R \in P_i} \rho\left(e_{\text{radar}}(\mathbf{p}_R; \mathcal{I}, s_d)\right) \tag{5.55}$$

5.8. Evaluation

In der Evaluation von DRISF konnte in der originalen Veröffentlichung [MWH+19] der *KITTI*-Datensatz [MG15b] verwendet werden, welcher unter anderem über Daten aus Stereokameras, Lidar und DGPS-INS verfügt. Durch die Erweiterung von DRISF zu DRISFwR werden RD-maps aus einem Radarsensor für die Inferenz des Szenenflusses benötigt. Diese sind nicht im *KITTI*-Datensatz enthalten, so dass im Rahmen dieser Arbeit ein eigener Datensatz zur Evaluation der Algorithmen akquiriert und erstellt wurde. Nachfolgend wird dieser Datensatz vorgestellt. Anschließend werden Metriken zur Evaluation der Szenenflussschätzgenauigkeit vorgestellt, mit denen DRISFwR bewertet

wird.

Bei DRISF und DRISFwR wird die GN-Optimierung gestoppt, sobald (a) die Aktualisierung konvergiert $\|\Delta\boldsymbol{\xi}\|_2 < 0.1\,\text{mm}$ oder (b) 100 Iterationsschritte durchgeführt wurden.

5.8.1. Datensatz für Evaluation des Szenenflusses

Für die Akquise von realen Sensordaten wurde das in Kapitel 3 vorgestellte Fahrzeug samt Sensorik verwendet. Auf diesem Datensatz werden die Schätzgenauigkeiten der Algorithmen zur Szenenflussschätzung quantifiziert und miteinander verglichen. Für diese Quantifizierung ist eine exakte Referenz des tatsächlichen Szenenflusses notwendig. Ist diese Referenz exakt, entspricht also dem tatsächlichen Szenenfluss, so spricht man von absoluter Referenz (engl.: „ground truth“). Eine Abweichung der Schätzung zur absoluten Referenz entspricht demnach der Abweichung zum tatsächlichen Wert. Ist die Referenz allerdings nicht exakt, so spricht man auch von der relativen Referenz. Eine Abweichung von dieser Referenz entspricht also nicht der Abweichung zum tatsächlichen Szenenfluss. Will man eine Referenz mittels Referenzsensorik abbilden, so wird bedingt durch systematische und stochastische Messungenauigkeiten der Referenzsensorik immer eine relative Referenz geschaffen. Als Faustformel für Referenz wird angenommen, dass die Genauigkeit der Referenzmessung mindestens um eine Dekade besser sein sollte als die Genauigkeit der zu bewertenden Messung. Die im Rahmen dieser Arbeit verwendete Referenz für Szenenfluss leiten wir aus dem Vorbild des *KITTI*-Datensatzes ab.

Im *KITTI*-Datensatz wurde für die Evaluation von optischem Fluss die Lidarpunktwolke aus 11 zeitlich benachbarten Frames mittels Iterative-Closest-Points (ICP) akkumuliert bzw. registriert und die Referenz für den optischen Fluss über die Projektionen der Pings in den nächsten Frame [GLU12] erstellt. Zusätzlich wurden Ausreißer und mehrdeutige Objekte, wie Fenster und Zäune, aus den Referenzdaten durch entsprechendes Labeling entfernt. In [MG15a] wurde beschrieben, dass dieses Vorgehen nur für statische Szenen Gültigkeit besitzt („provide scene-flow ground truth only for rigid scenes without independently moving objects “) und die Referenzgenerierung für Szenenfluss mit unabhängig bewegten Objekten vorgestellt und in *KITTI* integriert. Dabei wurde zunächst der Szenenfluss für statische Objekte berechnet, indem die Lidar-Punktwolken aus mehreren Frames akkumuliert wurden. Für die Akkumulation wurde die Egobewegung aus einer Kombination von ICP und GPS-Daten geschätzt. Punkte von bewegten Objekten wurden danach manuell entfernt. Für den Szenenfluss von unabhängig bewegten Objekten wurden die Pings der bewegten Objekte manuell gekennzeichnet und CAD-Modelle von typischen Fahrzeugen in diese Punktwolken eingepasst. Der Szenenfluss für diese Objekte wurde dann aus der Verschiebung und Verdrehung der CAD-Modelle zwischen den Frames geschätzt. Der gesamte Szenenfluss ergab sich somit als Summe von Szenenfluss von statischen Objekten und Szenenfluss von unabhängig bewegten Objekten. Dieses Vorgehen ist in seiner Qualität aus anderen Datensätzen nicht bekannt, hat jedoch den Nachteil, dass ein sehr zeitintensiver manueller Labelingaufwand zu investieren ist. Maßgeblich dafür ist der zu investierende Aufwand für das manuelle Labeling und die Implementierung der Einpassung von CAD-Modellen. Selbst in *KITTI* konnten mit der Methodik nur etwa 400 Beispiele bereitgestellt werden, obschon diese aus möglichst statistisch unabhängigen Szenarien entstammten. Weiterhin wurden bei der CAD-Einpassung nur starre Körper berücksichtigt und Mehrkörperdynamiken von

Objekten somit nicht berücksichtigt. Trotz dieses hohen Aufwandes wurden in anderen Veröffentlichungen Imperfektionen der Szenenflussreferenz gefunden und beschrieben, siehe bspw. [MWH+19]. Obige Probleme beziehen sich maßgeblich auf die Referenz für unabhängig bewegte Objekte. Um diesen Problemen aus dem Weg zu gehen, werden wir eine Bewertung des Szenenflusses ausschließlich anhand statischer Szenen vornehmen und den Referenzszenenfluss für statische Objekte nach obigem Vorbild generieren. Der Referenzszenenfluss entspricht dem geschätzten Szenenfluss für die Bewegung des Hintergrundes nach Abschnitt 5.6. Da DRISF und DRISFwR zur Szenenflussschätzung keine offensichtliche Differenzierung bei der Szenenflusschätzung zwischen statischen und dynamischen Objekten machen, ist aus Sicht des Autors die Evaluierung anhand statischer Szenen ein guter Indikator für mögliche Verbesserung der Schätzgenauigkeit. Für die Evaluierung wurde ein Parkplatzszenario ausgewählt, in welchem keine bewegten Objekte vorhanden waren. Die gefahrene Trajektorie des Datensatzes ist in Abbildung 5.7 dargestellt. Ein beispielhaftes Kamerabild mit Lidar-Projektionen ist in Abbildung 4.1 dargestellt.

Abbildung 5.7.: **Gefahrene Trajektorie des Datensatzes:** Bei der Akquise wurde etwa 1 h lang durch Lippstadt gefahren. Die Aufzeichnung erfolgte kontinuierlich. Dabei wurden typisch Szenarien aus Stadtverkehr, Autobahn, Landstraße und Parkplatz aufgezeichnet. Vergrößert dargestellt, ist die Fahrtrajektorie aus dem Parkplatzszenario für die Evaluierung der Szenenflussschätzung. Nach [EB6].

Der Datensatz für die Evaluierung hat eine Dauer von etwa 3 min und umfasst etwa 1700 Bilder pro Kamera. Dabei wurden etwa 13000 Instanzen von einzelnen parkenden Fahrzeugen detektiert. Einige Beispiele sind in Abbildung 5.8 dargestellt.

Abbildung 5.8.: **Szenenbeispiele des Datensatzes zur Szenenfluss Evaluierung:** Dargestellt sind einige repräsentative Beispiele aus dem Datensatz zur Szenenfluss Evaluierung. In der oberen Reihe sind die Beispiele der ersten Kamera dargestellt, in der unteren Reihe, der zweiten Kamera. In den Farbbildern sind die Instanzmasken farblich überlagert hervorgehoben.

Die Fahrtrajektorie bestand aus stark variierender Fahrzeugdynamik mit einer Fahrzeuggeschwindigkeit bis etwa $60\,\mathrm{km\,h^{-1}}$. In den Abbildungen wird noch einmal verdeutlicht, dass zwar ausschließlich stationäre Objekte dargestellt sind, diese aber mitunter stark verdeckt sind und den Schwierigkeitsgrad für u.a. die Szenenflussschätzung somit erhöhen.

Für den Datensatz wurden die Instanzen automatisch mit dem in Abschnitt 4.3 vorgestellten Ansatz detektiert und mit dem in Abschnitt 4.4 vorgestellten Clusterverfahren verfeinert. Diese wurden als gut empfunden, obschon kleinere Fehler in den Masken wahrgenommen werden können, siehe beispielsweise einzelne fehlende Objekte in Abbildung 5.8. Unter Berücksichtigung des Aufwandes wurde ein manuelles Nachlabeln der Instanzmasken als nicht notwendig empfunden. Da hier ohnehin statische Szenen bewertet werden und somit benachbarte Pixel einen ähnlichen Szenenfluss aufweisen, ist der Einfluss von Imperfektionen in den Instanzmasken zu vernachlässigen und kein wesentlicher Einfluss auf die Evaluierungsmetriken zu erwarten.

5.8.2. Quantifizierung der Schätzfehler

Im vorigen Abschnitt wurde das Vorgehen zur Bestimmung des „ground-truth“ Szenenflusss vorgestellt. In diesem Abschnitt werden wir einen objektiven Vergleich der Szenenflussschätzer anhand von quantifizierten Abweichungen gegenüber „ground-truth“ Szenenfluss durchführen. Da in dieser Arbeit die Motivation auf einer automatischen Annotation von RD-Daten des Radars liegt, werden wir besonderen Fokus auf Abweichungen in der Radialgeschwindigkeit gegenüber der Referenz legen.

5.8.2.1. Metrik nach *KITTI*

Für die Bewertung von Szenenfluss werden im *KITTI*-Datensatz Fehlerraten für Disparität und optischen Fluss berechnet. Die Disparität ergibt sich im *KITTI*-Datensatz

zu [Mon22]

$$\text{disparity} = \frac{\text{focallength} \cdot \text{baseline}}{\text{depth}}, \tag{5.56}$$

wobei „focallength“ die Brennweite der verwendeten Kamera in Pixeln ist, „baseline“ der laterale Abstand der Stereokameras in Metern und „depth“ die Tiefenmessung in Metern ist. Der optische Fluss ist die Pixelverschiebung in Bildkoordinaten und wird in Pixeln angegeben, vgl. Abschnitt 4.5. Eine kurze Beschreibung, wie im *KITTI*-Datensatz das „ground-truth“ bestimmt wurde, ist im vorigen Abschnitt gegeben. Wenn die Abweichung der Disparität oder optischen Fluss 3px oder 5% übersteigt, wurde dies als Fehler in der Szenenflussschätzung gewertet. Eine Dokumentation über die Motivation hinter den Schwellwerten beim *KITTI*-Datensatz konnte nicht gefunden werden.

5.8.2.2. Angepasste Metrik für den Radar

Eine Aufgabe dieser Arbeit ist das automatische Labeling von RD-Daten. Wir werden im späteren Verlauf dieser Arbeit das Labeling in Kamerabildern durchführen und diese mit den Pixeln aus RD-Daten des Radars assoziieren. Für eine korrekte Assoziation ist eine hinreichend gute Qualität von radialem Abstand und radialer Geschwindigkeit erforderlich. Die Abweichung zum „ground-truth“ sollte die Auflösung des Radars in den RD-Dimensionen nicht überschreiten. Für das verwendete Radar liegt diese Dopplerauflösung bei $0.25\,\mathrm{m\,s^{-1}}$. Im Rahmen dieser Auswertung macht es also durchaus Sinn, sich auf diese Festlegung zu konzentrieren und als Abweichungen in der radialen Geschwindigkeitsschätzung nach SI [BIP06] Einheiten in $\mathrm{m\,s^{-1}}$ zu ermitteln.

Mit den Parametern aus der Kamerakalibrierung und Gleichung 5.20, 5.25 und 5.26 ermitteln wir dazu die Radialgeschwindigkeit in Radarsensorkoordinaten zu

$$v_{\mathrm{r,\ radar}}(\mathbf{p}, \boldsymbol{\xi}) = \frac{\left[\frac{u-c_u}{f_u}, \frac{v-c_v}{f_v}, 1\right] \cdot {}^{R\leftarrow C}\mathbf{R}\boldsymbol{\xi}}{\sqrt{(\frac{u-c_u}{f_u})^2 + (\frac{v-c_v}{f_v})^2 + 1}}. \tag{5.57}$$

Im Zähler wird das Skalarprodukt zweier 1×3 Vektoren gebildet. Folglich handelt es sich um eine skalare Größe.

Wie zuvor beschrieben, werden zur Evaluierung ausschließlich Szenen mit (geo-)stationärem[7] Inhalt herangezogen, so dass „ground-truth“ bzw. der beste Schätzwert des Szenenflusses $\boldsymbol{\xi}_{gt} = \boldsymbol{\xi}_{bg}$ dem Hintergrundszenenfluss aus Gleichung 5.14 entspricht. Damit berechnet sich die Abweichung der Radialgeschwindigkeitskomponente für das Kamerapixel $\mathbf{p}$ zu

$$e_{\mathrm{SF,radial}}(\mathbf{p}) = |v_{\mathrm{r,\ radar}}(\boldsymbol{\xi}_{gt[\mathbf{p}]}) - v_{\mathrm{r,\ radar}}(\boldsymbol{\xi}_{[\mathbf{p}]})|. \tag{5.58}$$

Gemäß oben genannter Bedingung definieren wir zusätzlich einen Fehler, wenn die

[7] Der Begriff „geostationär“ soll ausdrücken, dass die Objekte stationär gegenüber Grund sind. Das Ego-Fahrzeug kann sich jedoch bewegen.

Abweichung der Radialgeschwindigkeit die Auflösung des Radars überschreitet

$$e_{\mathrm{SF,radial,bin}}(\mathbf{p}) = \begin{cases} 1 & e_{\mathrm{SF,radial}}(\mathbf{p}) \geq 0.25\,\mathrm{m\,s^{-1}} \\ 0 & \text{sonst} \end{cases}. \tag{5.59}$$

Da bei DRISF und DRISFwR $\boldsymbol{\xi}_{bg}$ algorithmisch bereits für die Bestimmung des Szenenflusses aller Hintergrundpixel verwendet wurde und für Hintergrundpixel die Abweichung entsprechend verschwindet, wird bei der Auswertung ausschließlich die Abweichung der Vordergrundpixel $\mathcal{P}_{\mathrm{fg}}$ ausgewertet.

Über die Menge der Vordergrundpixel berechnet sich eine mittlere Abweichung pro Kamerabild zu

$$\mathrm{MAE}_{\mathrm{SF,radial}} = \frac{1}{N} \sum_{\mathbf{p} \in \mathcal{P}_{\mathrm{fg}}} e_{\mathrm{SF,radial}}(\mathbf{p}), \tag{5.60}$$

wobei N die Kardinalität der Menge $\mathcal{P}_{\mathrm{fg}}$ ist. Analog berechnet sich die Fehlerrate zu

$$\mathrm{MAE}_{\mathrm{SF,radial,bin}} = \frac{1}{N} \sum_{\mathbf{p} \in \mathcal{P}_{\mathrm{fg}}} e_{\mathrm{SF,radial,bin}}(\mathbf{p}). \tag{5.61}$$

Die mittlere absolute Abweichung mittelt über sämtliche Vordergrundpixel eines Kamerabildes im Testdatensatz. Diese Art der Mittlung ist motiviert durch die Metriken im *KITTI*-Datensatz, bei denen ebenfalls über alle Pixel gemittelt wird. Problematisch dabei ist, dass die Pixelmenge entsprechend der Entfernung skaliert, vgl. Abbildung 3.5, und damit implizit Objekte mit geringer Entfernung zur Linse stärker gewichtet werden. Analog reduziert sich die Pixelanzahl von Objekten häufig durch Teilverdeckung, wie in Abbildung 5.8 dargestellt.

Neben der pixelbasierten Metrik wollen wir eine instanzbasierte Metrik verwenden, um damit unabhängig von der zuvor genannten Skalierung der Pixelmenge über Entfernung zu sein. Dafür definieren wir zusätzlich die mittlere Abweichung über alle Pixel einer Instanz zu

$$e_{\mathrm{SF,radial,instance}}(i) = \sum_{\mathbf{p} \in \mathcal{P}_i} e_{\mathrm{SF,radial}}(\mathbf{p}), \tag{5.62}$$

und die mittlere Abweichung aller Instanzen

$$\mathrm{MAE}_{\mathrm{SF,radial,instance}} = \frac{1}{N_{\mathrm{i}}} \sum_{i=1}^{N_{\mathrm{i}}} \frac{1}{N_{\mathrm{p,i}}} \sum_{\mathbf{p} \in \mathcal{P}_i = i} e_{\mathrm{SF,radial,instance}}(\mathbf{p}), \tag{5.63}$$

wobei N_{i} die Anzahl der Instanzen im Kamerabild und $N_{\mathrm{p,i}}$ der Anzahl der Pixel einer Instanz im Kamerabild ist. Analog wird die mittlere Fehlerrate definiert zu

$$\mathrm{MAE}_{\mathrm{SF,radial,instance,bin}} = \frac{1}{N_{\mathrm{i}}} \sum_{i=1}^{N_{\mathrm{i}}} \frac{1}{N_{\mathrm{p,i}}} \sum_{\mathbf{p} \in \mathcal{P}_i = i} e_{\mathrm{SF,radial,bin}}(\mathbf{p}). \tag{5.64}$$

Eine Auswertung der Tangentialgeschwindigkeit wird hier nicht vorgenommen, da sie für das Labeling von Radardaten und damit für den weiteren Verlauf dieser Arbeit keine Bedeutung haben.

5.8.2.3. Per Bild Fehler

Die Abweichung der radialen Komponente des Szenenflusses gemäß Gleichung 5.60 gegenüber der Referenz ist für drei verschiedene Algorithmen in Abbildung 5.9 a) dargestellt. Rot markiert sind die Abweichungen von HD^3 [YDY19]. Das Verfahren ist kein eigentliches Szenenflussverfahren, sondern ein Verfahren für die Schätzung des optischen Flusses, welches die Assoziation der Lidarpings zwischen den Frames vornimmt. Bei der Schätzung des Szenenflusses kommt keine Segmentierung nach Instanzen zum Einsatz, sodass potenziell zusammenhängende Bewegungsmuster nicht geclustert werden. Grün markiert sind die Abweichungen zur Referenz, erreicht durch das klassische DRISF-Verfahren. In Blau dargestellt sind die Abweichungen durch das in dieser Arbeit vorgestellte DRISFwR Verfahren. Entlang der horizontalen Koordinatenachse sind die Frames des Testdatensatzes aufgelistet. Entlang der vertikalen Koordinatenachse sind die erreichten Abweichungen dargestellt. Zu erkennen ist, dass die Abweichungen mit dem Verfahren nach HD^3 in der Regel größer ausfallen als beim Verfahren DRISF und DRISFwR. Des Weiteren sind die vielen grünen Datenpunkte zu erkennen. Das bedeutet, dass bei gleichem Frame DRISFwR eine geringere Abweichung erreicht hat als DRISF. Um diese Abweichung noch klarer darzustellen, sind in Abbildung 5.9 b) die Verbesserungen der radialen Komponente des Szenenflusses durch DRISFwR gegenüber den anderen Verfahren dargestellt. Positive Werte bedeuten eine Verbesserung durch den Einsatz von DRISFwR. Negative Werte: Eine Verschlechterung durch den Einsatz von DRISFwR.

In fast keinem Frame sind negative Werte (Abbildung 5.9, unten) zu beobachten, so dass kein signifikanter Nachteil durch die Verwendung von DRISFwR festgestellt werden kann. Im Gegenteil, es sind fast ausschließlich positive Werte zu beobachten, welche eine bessere Schätzgenauigkeit durch DRISFwR belegen.

Da sich die eingetragenen Datenpunkte verdecken können und somit im Wesentlichen nur der eingenommene Wertebereich erkennbar ist, sind in alternativer Darstellung die Abweichungen als Histogramm in Abbildung 5.10 dargestellt. Auf der horizontalen Koordinatenachse ist das MAE eingetragen. Auf der vertikalen Koordinatenachse die Häufigkeit.

Zu erkennen ist, dass die MAE-Verteilung bei HD^3 bis etwa $0.8\,\mathrm{m\,s^{-1}}$ reicht. Bei DRISF und DRISFwR beschränkt sich die Verteilung im wesentlichen auf einen Bereich bis etwa $0.1\,\mathrm{m\,s^{-1}}$, wobei bei DRISFwR die Häufigkeit des ersten Balkens ($[-0.01, 0.01)\mathrm{m\,s^{-1}}$) gegenüber DRISF fast verdoppelt wurde (542 vs. 941).

In Abbildung 5.11 ist der nach Gleichung 5.61 ermittelte prozentuale Anteil an Pixeln pro Bild eingezeichnet, welche mit einer Abweichung von mehr als $0.25\,\mathrm{m\,s^{-1}}$ vermessen wurden.

Auch hier ist zu beobachten, dass HD^3 fast in allen Frames eine schlechtere Fehlerrate erreicht als DRISF und DRISFwR. Die Metrik bei DRISFwR ist ein wenig besser als bei DRISF. Werden die Datenpunkte aus allen Frames zusammengefasst, so ergeben sich die in Tabelle 5.1 dargestellten Werte.

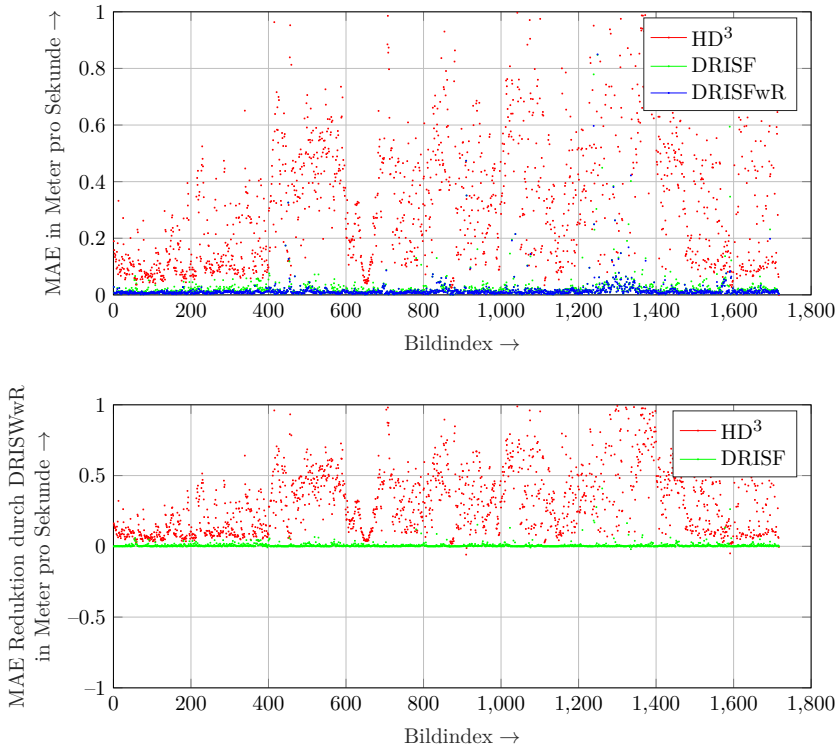


Abbildung 5.9.: **MAE, gemittelt über alle Bildpixel:** Oben dargestellt ist das erreichte Mean-Absolute-Error (MAE) nach Gleichung 5.60. Niedrigere Werte entsprechen einer besseren Schätzung. Unten dargestellt ist die MAE-Differenz der Verfahren gegenüber der MAE aus dem hier vorgestellten DRISFwR. Positive Werte entsprechen einem Benefit durch DRISFwR. Negative Werte bedeuten eine Verschlechterung durch die Verwendung von DRISFwR.

Tabelle 5.1.: **Gemittelten Metriken über alle Frames**. Dargestellt sind die gemittelten MAE und Fehlerraten sowie der Median der mittleren Abweichung für die Verfahren zur Szenenflussschätzung.

Methods	$MAE_{SF,radial}$ (cm/s)	Median $AE_{sf,radial}$ (cm/s)	$MAE_{SF,radial,bin}$ (%)
HD^3	39,8	31,7	50.0016
DRISF	12,3	1,3	11.3126
DRISFwR	11,6	0,9	10.5223

Zu beobachten ist, dass DRISFwR in nahezu allen Szenarien eine Reduktion der Abweichung der radialen Komponente von der Referenz erreichen konnte. Die absolute Abweichung in der radialen Komponente kann aus den Moden der Histogramme aus Abbildung 5.10 abgeschätzt werden. Diese entsprechen in etwa der Median-Abweichung aus Tabelle 5.1.

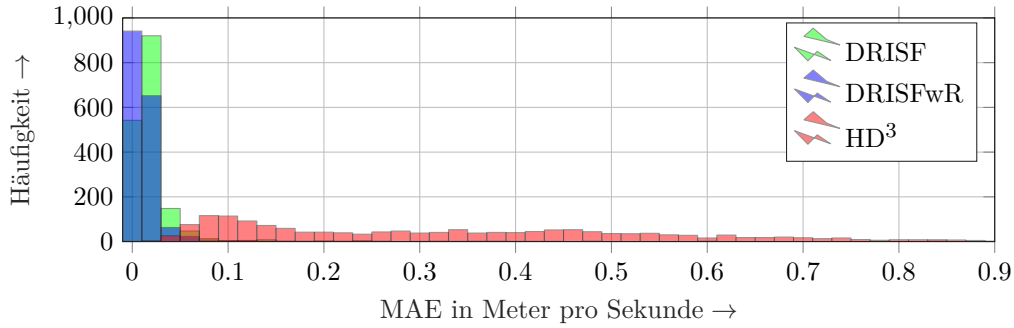


Abbildung 5.10.: **MAE, gemittelt über alle Bildpixel:** Dargestellt ist das erreichte MAE nach Gleichung 5.60 in Form eines Histogramms.

5.8.2.4. Per Instanz Fehler

Die bisherigen Ergebnisse lassen vermuten, dass durch die Hinzunahme des radarbasierten Energieterms, siehe Gleichung 5.15, eine leichte Verbesserung in der Szenenflussschätzung zu erreichen ist. Wie in den Gleichung 5.62, Gleichung 5.63 und Gleichung 5.64 beschrieben, wird die Auswertung nun ebenfalls per Instanz vorgenommen. Analog zu Abbildung 5.9 sind die Abweichungen aller Objekte aus dem Testdatensatz in Abbildung 5.12 dargestellt.

Lagen die Abweichungen in Abbildung 5.9 insbesondere für die Verfahren DRISF und DRISFwR im Wesentlichen unterhalb von $0.1\,\mathrm{m\,s^{-1}}$, so ist in Abbildung 5.12 eine deutlich höhere Streuung der Punkte zu beobachten. Wie zuvor beschrieben, werden in Unterunterabschnitt 5.8.2.3 der Linse nahe gelegene Objekte mit entsprechend größerer Pixelmenge statistisch stärker gewichtet und gleichzeitig genauer, wodurch die geringeren Werte in der Metrik zu erklären sind. Eine weitere Validierung zur Stützung dieser Hypothese wird im Rahmen dieser Arbeit nicht durchgeführt, da ihr für den weiteren Verlauf dieser Arbeit keine wesentliche Relevanz zugeordnet wird.

Weiterhin kann beobachtet werden, dass auch hier DRISFwR gegenüber DRISF im Allgemeinen eine kleinere Abweichung per Objekt erreicht. Dies zeigt sich insbesondere auch in Abbildung 5.12, unten), wo noch einmal die Differenz der Abweichung gegenüber DRISFwR dargestellt sind. Nur bei wenigen Objekten konnte DRISF ein geringeres MAE erreichen als DRISFwR.

Wie zuvor auch wurden auch hier die Abweichungen in Form eines Histogramms gesammelt und in Abbildung 5.13 dargestellt.

In der Histogramm-Darstellung ist zu beobachten, dass die Abweichung zur Referenz bei allen Verfahren im Gros kleiner als die Doppler-Auflösung des Radars ($0.25\,\mathrm{m\,s^{-1}}$) ist.

Der Vollständigkeit halber sind in Abbildung 5.14 die Fehlerraten der Verfahren per Objekt dargestellt. Dabei wurde ermittelt, wie hoch der relative Anteil an Pixeln per Instanz ist, die eine größere Abweichung als die Dopplerauflösung des Radars aufweisen.

Da die Streuung der Datenpunkte mitunter sehr hoch ist und sich viele Datenpunkte optisch überlagern, wird diese Darstellung eher als ungeeignet für eine Bewertung empfunden. Deutlich eingängiger sind die gemittelten Metriken in Tabelle 5.2 zusammengefasst.

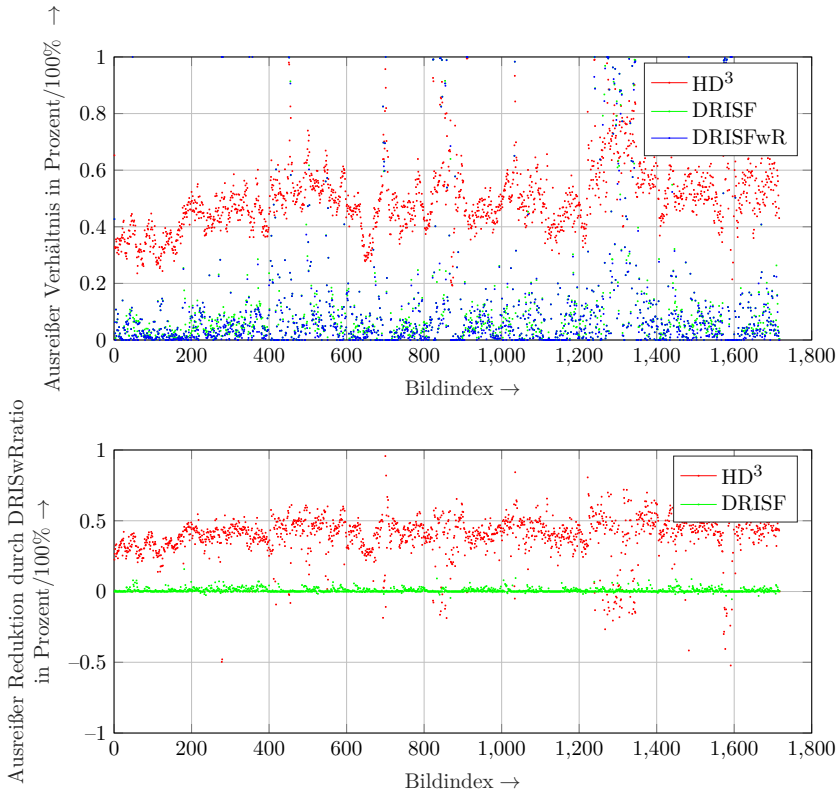


Abbildung 5.11.: **Mittlere Fehlerrate pro Bild:** Oben dargestellt ist die erreichte Fehlerrate nach Gleichung 5.61. Niedrigere Werte entsprechen einer besseren Schätzung. Unten dargestellt ist die Fehlerrate der Verfahren gegenüber der Fehlerrate aus dem hier vorgestellten DRISFwR. Positive Werte entsprechen einem Benefit durch DRISFwR. Negative Werte bedeuten eine Verschlechterung durch die Verwendung von DRISFwR.

Tabelle 5.2.: **Gemittelten Metriken über alle Objekte**. Dargestellt sind die gemittelten MAE und Fehlerraten sowie der Median der mittleren Abweichung für die Verfahren zur Szenenflussschätzung.

Methods	$MAE_{SF,radial}$ (cm/s)	Median $AE_{sf,radial}$ (cm/s)	$MAE_{SF,radial,bin}$ (%)
HD^3	50,2	11,0	63.4834
DRISF	21,8	1,7	34.5126
DRISFwR	16,2	1,4	29.6659

In der Tabelle wurden die Abweichungen und Fehlerraten aller Objekte gemittelt. Zusätzlich ist der Medianwert der Abweichungen dargestellt, welcher in etwa den Moden aus Abbildung 5.13 entspricht. In allen Metriken hat DRISFwR besser abgeschnitten als die anderen Verfahren. Als wissenschaftliche Aussage lässt sich so festhalten, dass durch die Hinzunahme der radarbasierten Gütefunktion eine leichte Verbesserung in der Szenenflussschätzung erreicht wurde. Diese Aussage gilt im Rahmen der hier untersuchten Testbedingungen. Ohne einen Nachweis nach den Methoden der Statistik vorzunehmen,

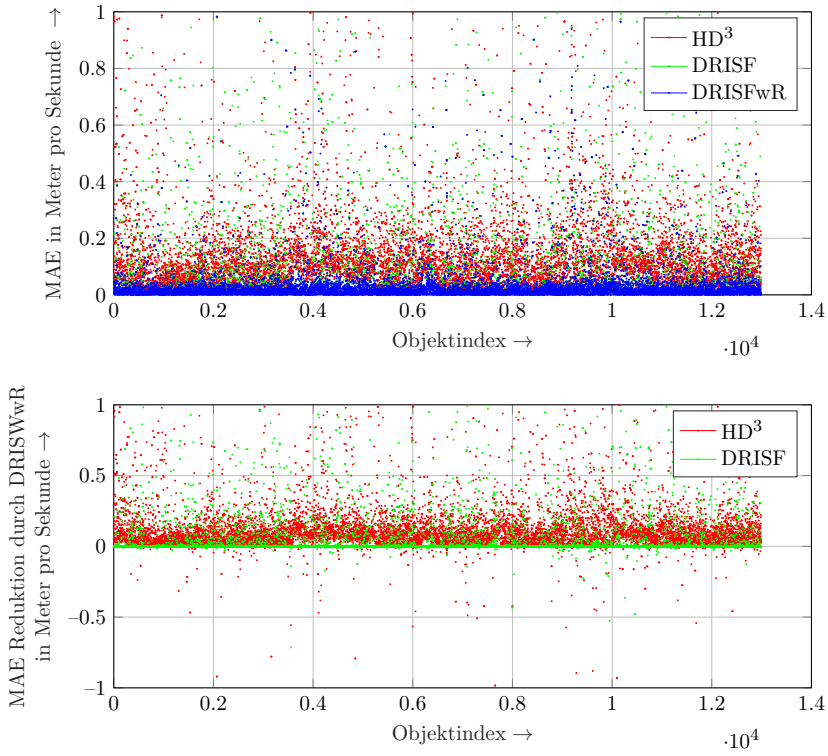


Abbildung 5.12.: **MAE des Szenenflusses aller Objekte:** Oben dargestellt ist das erreichte MAE per Instanz nach Gleichung 5.62. Niedrigere Werte entsprechen einer besseren Schätzung. Unten dargestellt ist die MAE Differenz der Verfahren gegenüber der MAE aus dem hier vorgestellten DRISFwR. Positive Werte entsprechen einem Benefit durch DRISFwR. Negative Werte bedeuten eine Verschlechterung durch die Verwendung von DRISFwR.

wird bei den verwendeten ca. 13000 Instanzen im Testdatensatz von einer statistischen Signifikanz ausgegangen.

5.8.3. Qualitativer Vergleich

Um einen Eindruck von den Fehlern zu erhalten, wurden die Abweichungen nach Gleichung 5.60 zur DRISFwR-Schätzung berechnet und aufsteigend sortiert. Das 10%-, 50%-, 70%- und 90%-Perzentil der Bilder ist in Abbildung 5.15 dargestellt. Zu beachten ist, dass die Abweichung nur innerhalb des FoV des Radarsensors berechnet wurde, siehe Abbildung 5.2, so dass z.B. der offensichtlich fehlerhaft geschätzte Szenenfluss für die Instanz oben rechts in Abbildung 5.15 a) nicht in das Bild zur Abweichung transportiert wird.

Die Bilder aus Abbildung 5.15 bestätigen insgesamt eine gute Qualität der Szenenflussschätzung. Trotz der erheblichen Verdeckung der Instanzen kann kaum eine größere Szenenflussabweichung für die dargestellten Szenarien beobachtet werden. Da hier nach den oben genannten Perzentilen allesamt unterhalb oder gleich dem 90-Perzentil selektierte Szenarien dargestellt sind, bestätigt sich die Verteilung aus Abbildung 5.10.

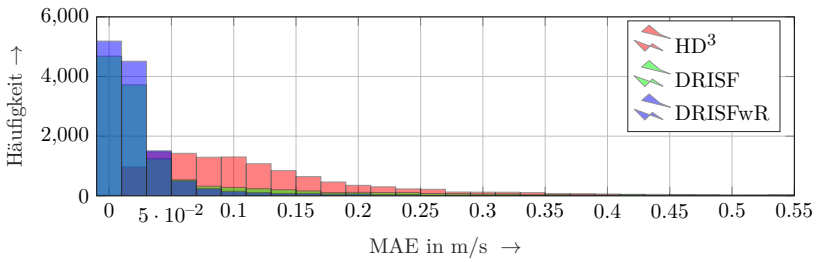


Abbildung 5.13.: **MAE, gemittelt über alle Bildpixel:** Dargestellt ist das erreichte MAE nach Gleichung 5.64 in Form eines Histogramms.

Der Großteil der Daten liegt bei einer geringeren Abweichung bzw. nur ein sehr geringer Prozentteil der Bilder weist eine größere gemittelte Abweichung auf. Zur weiteren Veranschaulichung sind in Abbildung 5.16 die 95–, 97–, 99– und 100– Perzentile dargestellt. Beim 95– und 99–Perzentil ist insgesamt eine kleinere Abweichung $< 0.2\,\mathrm{m\,s^{-1}}$ für fast alle Instanzen erkennbar. Beim 97–Perzentil ist eine größere Abweichung für eine verdeckte Instanz in der zweiten Parkreihe zu erkennen. Beim 100–Perzentil ist ein vollständig falscher Szenenfluss zu beobachten, welcher durch einmalig korrupte Sensordaten entstand.

5.8.4. Vergleich der Laufzeit

Neben der eigentlichen Schätzgenauigkeit ist auch die Laufzeit der Szenenflussschätzung für die Bearbeitung großer Datensätze relevant. Die gemittelten Laufzeiten über den gesamten Datensatz zu der in Abbildung 5.7 dargestellten Fahrtrajektorie sind in Tabelle 5.3 eingetragen. Die Laufzeiten wurden auf einem PC gemessen. Der PC ist mit einem Intel Core I7 Prozessor und einer NVIDIA GEFORCE GTX 1080 Graphics-Processing-Unit (GPU) ausgestattet gewesen. Alle Algorithmen wurden als GPU-Code implementiert.

Tabelle 5.3.: **Vergleich der Inferenzdauer für Szenenflussschätzung**

Methods	runtime
HD^3	0.12s
DRISF	0.6s
DRISFwR	0.8s

Zu erkennen ist, dass DRISFwR in etwa 0.8 s pro Kameraframe benötigt hat. Durch die gestiegene Komplexität gegenüber DRISF hat sich die Laufzeit um etwa 0.2 s verschlechtert. Die Inferenz des etwa einstündigen Datensatzes mit etwa 36000 Frames (72000 Bilder) dauert mit DRISFwR in etwa 16 h.

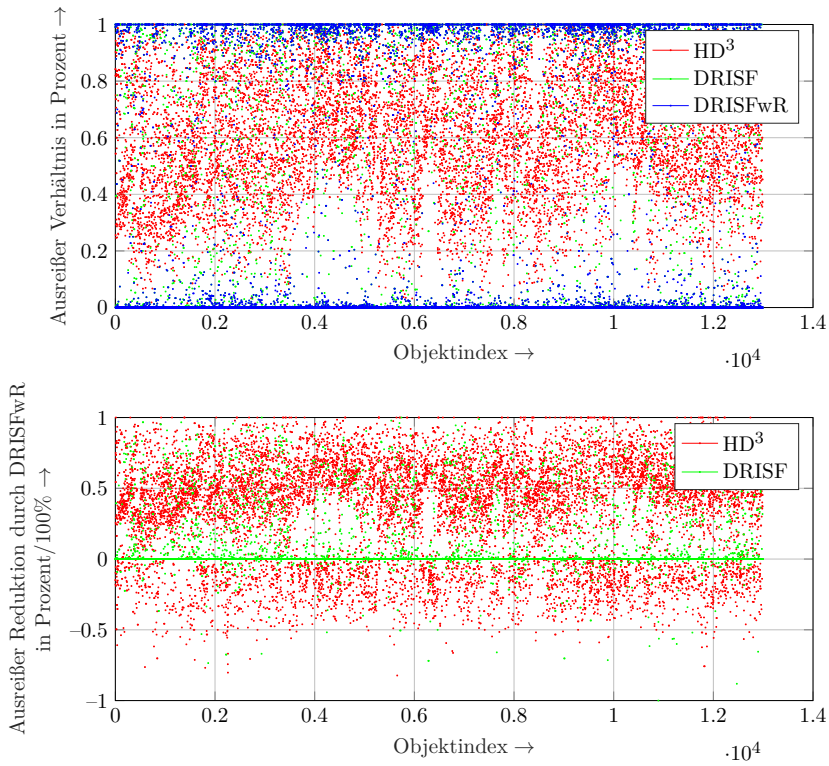


Abbildung 5.14.: **Mittlere Fehlerrate pro Bild:** Oben dargestellt ist das erreichte Fehlerrate nach Gleichung 5.64. Niedrigere Werte entsprechen einer besseren Schätzung. Unten dargestellt ist das Fehlerrate der Verfahren gegenüber der Fehlerrate aus dem hier vorgestellten DRISFwR. Positive Werte entsprechen einem Benefit durch DRISFwR. Negative Werte bedeuten eine Verschlechterung durch die Verwendung von DRISFwR.

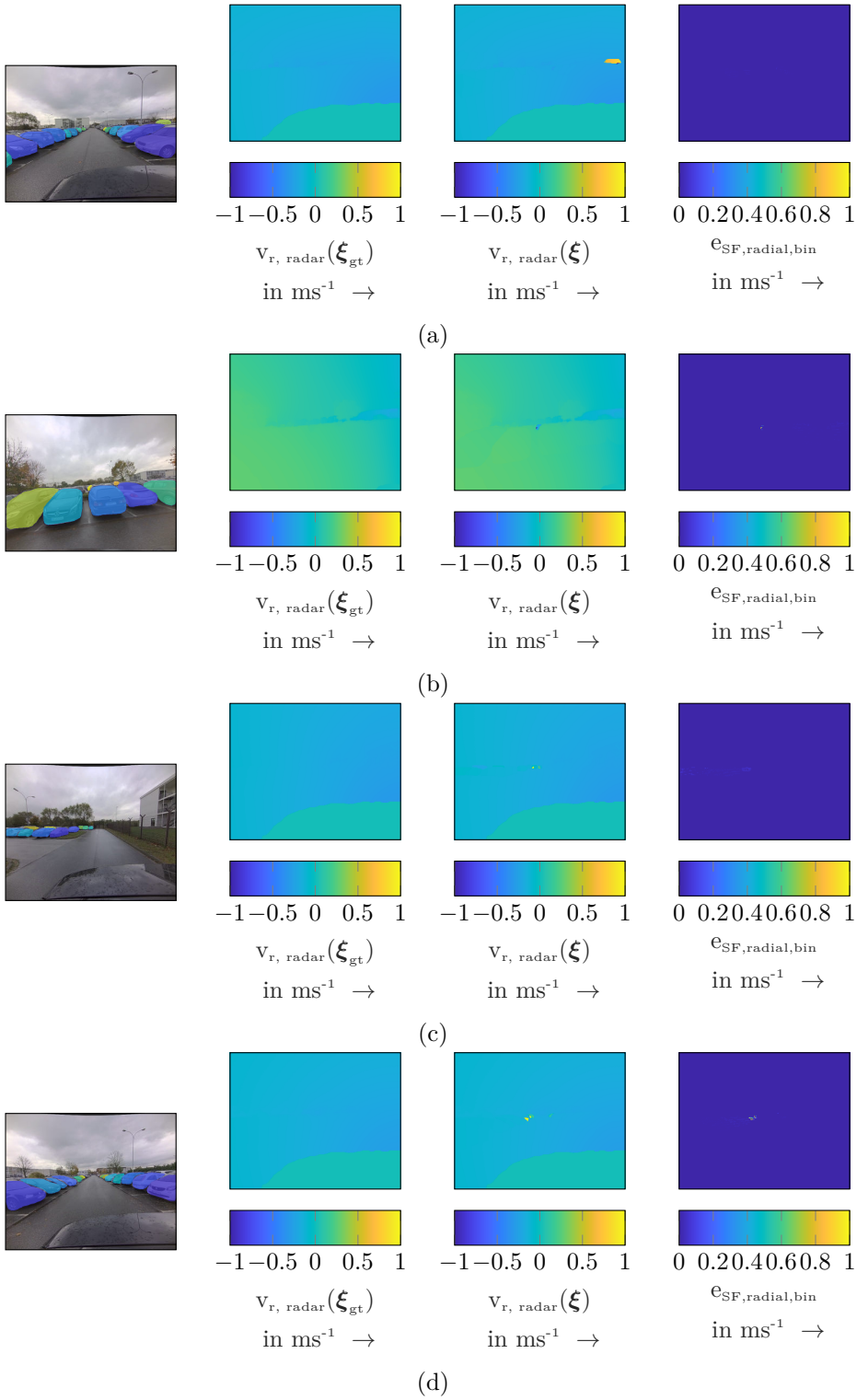


Abbildung 5.15.: **Vergleich von Szenenflussreferenz und -schätzung:** Von links nach rechts: RGB Kamerabild mit hervorgehobenen Instanzen, Referenzszenenfluss, Szenenflusschätzung und daraus ermittelter Abweichung nach Gleichung 5.60. Von oben nach unten: 10%-, 50%-, 70%- und 90%-Perzentil sortiert nach Szenenflussabweichung.

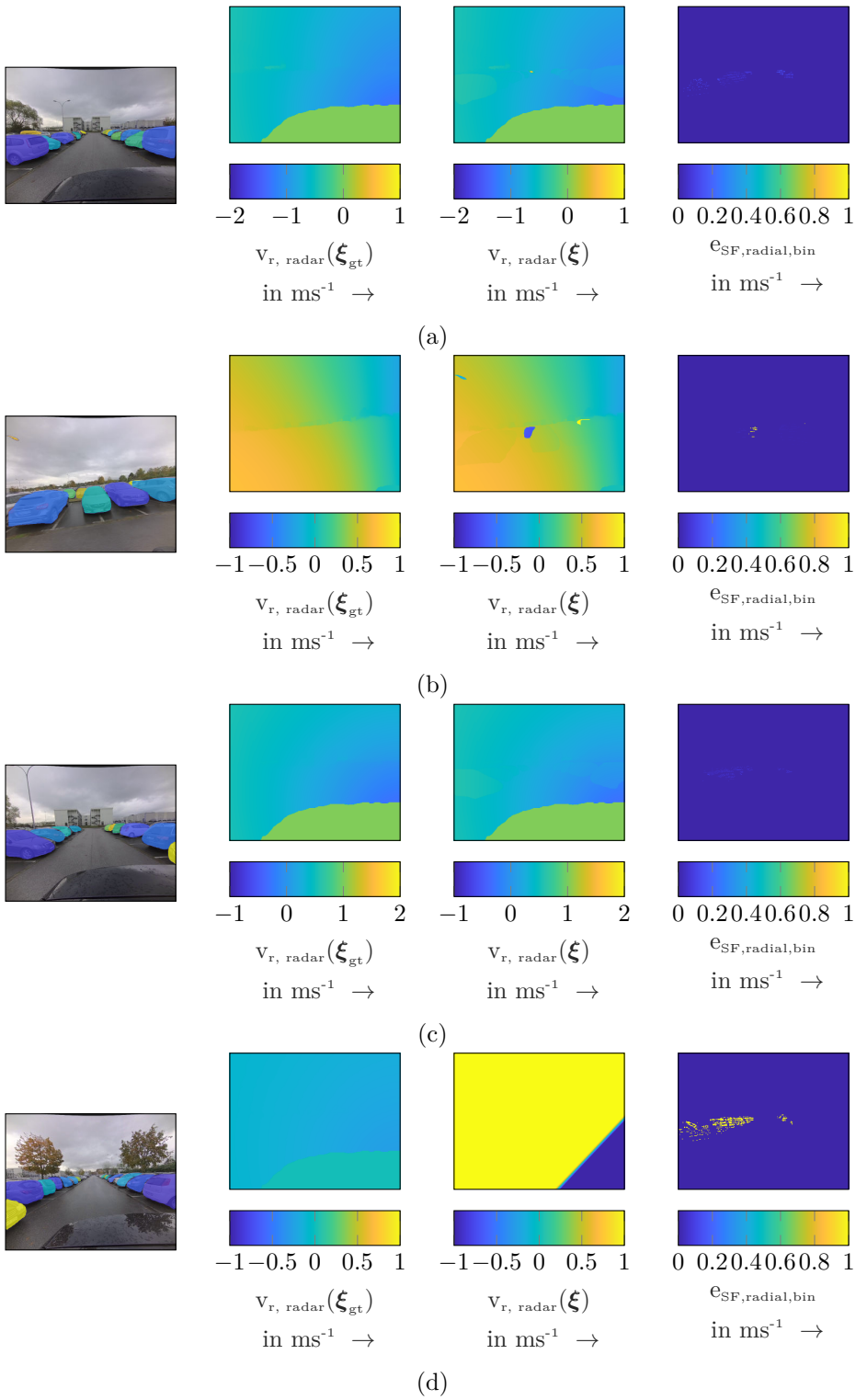


Abbildung 5.16.: **Vergleich von Szenenflussreferenz und -schätzung:** Von links nach rechts: RGB Kamerabild mit hervorgehobenene Instanzen, Referenzszenenfluss, Szenenflusschätzung und daraus ermittelter Abweichung nach Gleichung 5.60.Von oben nach unten: 95%-, 97%-, 99%- und 100%-Perzentil sortiert nach Szenenflussabweichung.

5.9. Zusammenfassung

In diesem Kapitel wurde ein Verfahren zur Schätzung des Szenenflusses um Informationen aus Radarmessungen erweitert. Wie in Abbildung 5.9 und 5.12 gezeigt wurde, konnte dadurch in den meisten Beispielen eine Verringerung der Abweichung in der radialen Komponente des Szenenflusses gegenüber einer Referenz erreicht werden. Nur bei wenigen Instanzen hatte DRISFwR eine größere Abweichung von der Referenz als DRISF, siehe Abbildung 5.12.

Analog zur Vorgehensweise im *KITTI*-Datensatz wurden die Abweichungen per Bild ermittelt. Da diese Statistik Instanzen entsprechend ihrer Pixelanzahl im Kamerabild gewichtet, wurde eine alternative Statistik generiert, bei welcher die Abweichung per Instanz gemittelt wird und die von der Pixelmenge unabhängig ist. Hier konnte eine deutliche Verbesserung gegenüber den anderen Szenenflussverfahren identifiziert werden.

Wie in anderer Literatur beschrieben, bspw. [MG15a], wurde auch hier ein statistischer Vorteil für Verfahren mit Bewegungsmodell gegenüber einem nahezu modellfreien Schätzer beobachtet. Eine Erklärung kann sein, dass Objekte als Starrkörper eine einheitliche Bewegung aufweisen. Für alle Pixel eines Objektes wird somit eine einheitliche Bewegung erwartet, und die Anzahl der zu schätzenden Parameter kann somit auf die Bewegungsparameter des Objektes reduziert werden.

Obwohl DRISFwR eine erhebliche Reduktion der Abweichung in der Radialgeschwindigkeit in der Szenenflussschätzung auf etwa $0.16\,\mathrm{m\,s^{-1}}$ im Mittel und $0.01\,\mathrm{m\,s^{-1}}$ im Median, siehe Tabelle 5.2, erreichen konnte, ergaben sich bei noch etwa 30% der Instanzen eine Abweichung, die größer als die Dopplerauflösung des hier verwendeten Radars ist. Für bestmögliche Labelingergebnisse ist im weiteren Verlauf der Arbeit mit einer manuellen Nachannotation der Daten zu rechnen.

Die Laufzeit von DRISFwR beträgt im Schnitt etwa 0.8 s und übersteigt die von DRISF mit 0.6 s nur leicht. Bei der verwendeten Framerate von 10 Frames-Per-Second (FPS) und dem zweiten Kamerasystem lässt sich eine einstündige Aufnahme in 16 h prozessieren, was im Rahmen dieses Projektes als akzeptabel empfunden wurde.

6. Projektion von RD-Gitter Daten in Kamerabilder

In Abschnitt 2.1 wurde das zweidimensionale RD-Spektrum sowie die Erweiterung mittels Beamforming zum RAD-Spektrum beschrieben. Ziel dieses Kapitels ist es, die Daten der Radarspektren vollständig in Kamerabilder zu projizieren und so eine eingängige visuelle Interpretation der Spektren zu schaffen. Ein Beispiel dieses Vorhabens ist in Abbildung 6.1 dargestellt.

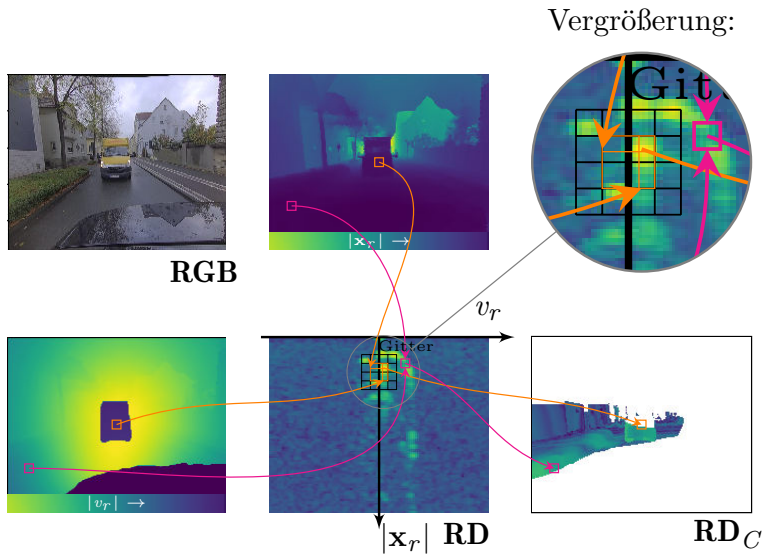


Abbildung 6.1.: **Warping des RD-maps in ein Kamerabild:** Als Beispiel der Warping-Operation wird die Intensität des RD-maps in das Kamerabild **RGB** projiziert. Dazu werden die radiale Entfernung, die radiale Geschwindigkeit und die korrespondierende Intensität in der RD-map bestimmt. Dieser Intensitätswert wird anschließend in das Gitter des Kamerbildes $\mathbf{RD}_C$ eingetragen. Dieser Prozess ist für zwei beispielhafte Kamerapixel (pinke und orangene Boxen) hervorgehoben. Zu beachten ist, dass in $\mathbf{RD}_C$ ausschließlich die Pixel im FoV des Radars gewarpt wurden, dargestellt durch die nicht-weißen Pixel in $\mathbf{RD}_C$. Nach [EB6].

In der Abbildung ist oben links das Kamerabild **RGB** und mittig unten das RD-map **RD** einer Szene dargestellt. Mittels geschätzter Entfernung (mittig oben) und Szenenfluss (unten links) im Kamerabild werden für jedes Pixel im Kamerabild die Positionen im RD-Gitter bestimmt und entsprechende Leistungsintensität in das Kamerabild projiziert $\mathbf{RD}_C$ (unten rechts).

Neben der Visualisierung im Kamerabild ist eine wesentliche weitere Motivation

das überwachte Training von ML-Verfahren auf Radarspektren durch Nutzung bestehender oder einfach zu generierender Annotationen aus Kameradaten. Die Nutzung von Annotationen aus anderen Sensoren („Lehrer“) zum Training eines Verfahrens zur Signalverarbeitung („Schüler“) wird in der Literatur häufig unter der Bdezeichnung „Cross-Modal Supervision“ geführt. Unter diesem Titel gibt es Beispiele aus der Spracherkennung [NSR+20], Radar-Objekterkennung [WJG+20] und WiFi-Radio Signalverarbeitung [ZLA+19].

Wie gleich gezeigt wird, bedeutet das Warping praktisch eine Umstrukturierung der ursprünglichen Pixel- bzw. Voxelanordnung im RD bzw. RAD-Gitter, so dass die Pixel an entsprechender Stelle im Kamerabild positioniert werden. Das Warping ist so aufgebaut, dass es sich nicht auf die Leistungsspektren des Radars beschränkt, sondern alle Daten, welche in Form eines RD bzw. RAD-Gitters vorliegen, in das Kamerabild gewarpt werden können. Dies können z.B. Ausgaben klassischer Signalverarbeitungen wie Winkelmessungen mittels PM-Schätzer oder Detektionkonfidenzen aus CFAR für die Pixel/Voxel im RD sein, insbesondere aber auch Signalverarbeitungen aus ML-Verfahren wie Winkelschätzungen via NN-Verarbeitung. Ein Beispiel dafür ist in Abbildung 6.2 dargestellt.

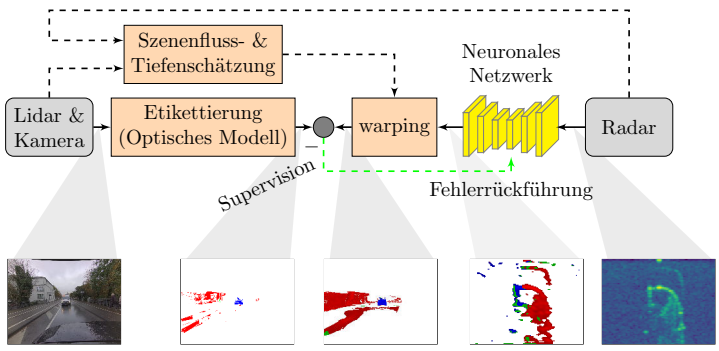


Abbildung 6.2.: **Übersicht des vorgestellten Systems zum überwachtem Training von Radar Signalverarbeitungen:** Lidar und Kamera stellen Sensordaten an das optische Modell (engl.: „vision model“) bereit, welches automatisch Annotationen im Kamerabild generiert. Gleichzeitig werden die vom Radar bereitgestellten Spektren durch ein NN prozessiert und durch die Warping-Schicht in die Domäne des Kamerabildes gewarpt. Die Abweichung zwischen generierter Annotation und gewarpter NN Prädiktion wird berechnet und über die Warping-Schicht zurück in das NN propagiert (grüner Pfeil). Die Warping-Schicht benötigt dichten Szenenfluss und Tiefenschätzung im Kamerabild, welche durch Lidar, Kamera und Radardaten bestimmt werden. Die unteren Bilder zeigen Beispiele einer semantischen Segmentierung an. Von links nach rechts: Kamerabild, semantische Referenzmaske (nur relevante Pixel angezeigt), gewarpte Prädiktion der semantischen Maske, prädizierte semantische Maske im RD-Gitter, RD-map. Nach [EB6].

Neben der Bereitstellung annotierter Trainingsdaten (Abbildung 6.2, optisches Modell), muss ein Training von ML-Verfahren algorithmisch gewährleistet sein. Für das Training von NN wird typischerweise das sogenannte „Error-Backpropagation“ ver-

wendet, bei welchem die Abweichungen zwischen Prädiktion des Netzwerkes und der Annotation ermittelt und sequenziell rückwärtsgerichtet durch alle Schichten des Netzwerkes propagiert werden. Praktisch wird immer ein Gradientenverfahren angewendet, welches den Fehlergradienten in Bezug auf die Parameter der Schichten berechnet. Danach werden die Schichtenparameter mit einer definierten Schrittweite entgegen dem Fehlergradienten aktualisiert. Dieser Vorgang wird bis zu einem Abbruchkriterium wiederholt. Nach erfolgreichem Training sollte die Abweichung zwischen Prädiktion und Label klein sein und damit einhergehend auch der Fehlergradient in jeder Schicht. Um diesen Trainingsprozess zu ermöglichen, muss die Warpingschicht erlauben, den Fehlergradienten zu bestimmen.

In diesem Kapitel wird zunächst eine allgemeine Einführung über das Warping in der Bildverarbeitung gegeben. Dann wird die oben beschriebene Notwendigkeit der Differenzierbarkeit am Beispiel eines generischen neuronalen Netzwerkes hergeleitet. Danach werden unterschiedliche Interpolationsverfahren zur Realisierung des Warpings vorgestellt und deren Vorwärts- und Rückwärtsdurchlauf im Trainingsprozess dokumentiert und somit die Tauglichkeit für das Training theoretisch nachgewiesen. Schlussendlich werden die Ergebnisse unterschiedlicher Warpingschichten dargestellt und visuell bewertet.

6.1. Definition der Warprichtungen

Wie beispielsweise in [RA15] beschrieben, wird der Begriff „Warping“ in der Bildverarbeitung als Synonym für die Anwendung von (geometrischen) Transformationen auf Bildinhalt verwendet. Als einfache Transformationen werden beispielsweise globale Rotation oder Deformation genannt. Aus der Art der Transformation kann ein Verzerrungsfeld berechnet werden, welches festlegt, wie die einzelnen Pixel aus einem Ursprungsbild in ein Zielbild verschoben werden. Dabei werden grundsätzlich zwei Arten von Bildwarping, das Vorwärts-Warping und das Rückwärts-Warping, verwendet. Beim Vorwärtswarp wird jedes Pixel aus dem Ursprungsbild entsprechend der Transformation verschoben, wodurch ein neues Bildgitter entsteht. Dieses neue Bildgitter passt in aller Regel nicht zum gewünschten Bildgitter des Zielbildes, so dass einzelne Pixel im Zielbild nicht oder mehrfach von Pixeln aus dem Ursprungsbild belegt sein können. Dieser Effekt ergibt sich aufgrund der Tatsache, dass die Transformationsfunktion im Allgemeinen nicht bijektiv ist. Beim Rückwärtswarp wird daher ausgehend vom Bildgitter des Zielbildes entsprechend der inversen Transformation ein Pixelgitter über das Ursprungsbild gebildet. Auch hier passt das neu geformte Gitter nicht automatisch zum Gitter des Ursprungsbildes, jedoch kann damit sichergestellt werden, dass alle Pixel im Zielbild prozessiert werden. Liegen das transformierte Gitter und das Gitter vom Ursprungs- oder Zielbild nicht übereinander, so können die Pixel durch Interpolation auf die Zielgitter angenähert werden. Anschauliche Beispiele hierzu sind in [RA15] zu finden.

Im Rahmen dieser Arbeit werden sowohl Vorwärts- als auch Rückwärtswarp verwendet. Der Vorwärtswarp wird verwendet, wenn Pixel des Kamerabildes in das Gitter des RD-maps transformiert werden sollen. Der Rückwärtswarp dagegen, wenn Pixel aus dem Gitter des RD-maps in das Kamerabild transformiert werden sollen.

In Abbildung 6.3 sind die beiden Warp-Richtungen dargestellt. Es ist zu erkennen, dass Quell- und Zielbild eine unterschiedliche Größe haben können. Wie später gezeigt

wird, wird die Transformationsfunktion in beiden Fällen zwischen den Gittern aus Radar und Kamera vollständig aus den Daten der Referenzsensorik (Entfernung, Szenenfluss und Winkel) der Kamerapixel bestimmt.

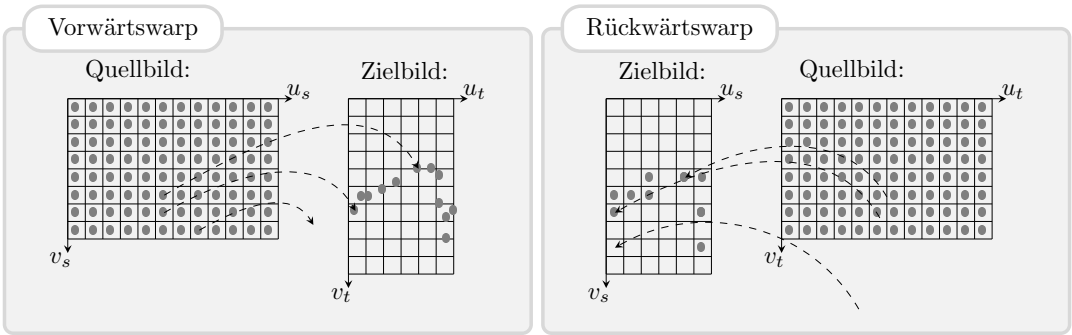


Abbildung 6.3.: **Vergleich der Warp Richtungen:** Beim Vorwärtswarp wird jedes Pixel im Quellbild („Source image“) geometrisch in das Zielbild („Target image“) transformiert und der Pixelinhalt entsprechend einer Interpolatrionsmethode in das Zielpixel eingetragen bzw. akkumuliert. Bei Rückwärtswarp wird ausgehend von jedem Zielbildpixel das entsprechende Pixel im Quellbild berechnet und dessen Wert ins Zielpixel eingetragen. Zeigt die assoziative Verbindungslinie auf eine Position außerhalb des Bildbereiches, wird das Pixel im Zielbild nicht befüllt. Nach [EB7].

Der wesentliche praktische Unterschied zwischen Vorwärts- und Rückwärtswarp ist, dass beim Vorwärtswarp mehrere Pixel des Quellbildes auf das gleiche Pixel im Zielbild zeigen können. Das Zielbild kann durch Akkumulation der Quellpixel aufgebaut werden. Beim Rückwärtswarp zeigt ein Zielpixel immer nur auf ein Pixel im Quellbild und holt sich daher den Grauwert. In beiden Fällen kann Interpolation angewendet werden, wenn nach der Transformation der Pixel die Gitterstruktur nicht getroffen wird.

Es ergibt sich das Problem, dass mathematisch festgelegt werden muss, wie bei der Sammlung mehrerer Quellpixel in ein Zielpixel umzugehen ist.

6.2. Generische Netzwerkstruktur und Trainingsprozess

Soll die Warping-Operation im Trainingsprozess eines NNes verwendet werden, so muss der Fehlergradient über sämtliche Operationen, wie in Abschnitt 2.2 beschrieben, berechnet werden können.

Wie in Abbildung 6.2 gezeigt, findet ein Vergleich von Label im Kamerabild und der gewarpten Netzwerkprädiktion statt. Wir definieren das Label als f_{GT} und die gewarpte Prädiktion f'. Für den Vergleich definieren wir eine Kostenfunktion l z.B. in der Form[1]

$$l(u,v) \quad = \quad \tau(e) = \tau\left(f'\left(u,v,r,\dot{r},\theta\right) - f_{\mathrm{GT}}\left(u,v\right)\right), \tag{6.1}$$

wobei τ eine konvexe Skalierungsfunktion ist und e die Differenz aus Prädiktion und

[1] Andere Formen sind möglich und üblich, z.B. bei Anwendung der Kreuzentropie.

Label ist. Die Prädiktion an der Pixelposition $[u, v]$ hängt zum einen von den Warping-Parametern r und $\dot{r}$ statt, welche die Pixelposition im RD-Gitter definieren, und zum anderen von den Parametern θ des NNes. In unserem Fall sind wir nicht an einem Training der Warping-Parameter interessiert, sondern ausschließlich am Training der Netzwerkparameter. Beim „error-backpropagation“ werden die Warping-Parameter als nicht variabel angesehen.

Verwendet man nun ein klassisches „gradient descent“ Verfahren (Schrittweite γ) zum Training der Netzwerkparameter, so würde eine Aktualisierung der Netzwerkparameter im Schritt n folgende Form annehmen:

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} - \gamma \frac{\partial l}{\partial \boldsymbol{\theta}}. \tag{6.2}$$

Der letzte Term beschreibt den Gradienten der Fehlerfunktion, bezogen auf die Netzwerkparameter, und kann durch Anwendung der Kettenregel modelliert werden zu

$$\frac{\partial l}{\partial \boldsymbol{\theta}} = \frac{\partial l}{\partial \tau}\frac{\partial \tau}{\partial e}\frac{\partial e}{\partial f'}\frac{\partial f'}{\partial f}\frac{\partial f}{\partial \boldsymbol{\theta}}. \tag{6.3}$$

Hierbei ist $\frac{\partial l}{\partial \tau}$ die Ableitung von der Abweichung, bezogen auf die Skalierungsfunktion, und im gezeigten Beispiel $\frac{\partial l}{\partial \tau} = 1$. $\frac{\partial \tau}{\partial e}$ ist die Ableitung von der Skalierungsfunktion, bezogen auf die Differenz e. Wird als Kostenfunktion beispielsweise der mittlere quadratische Fehler verwendet, so ist

$$\tau(e) = \frac{1}{2}e^2 \tag{6.4}$$

und somit

$$\frac{\partial \tau}{\partial e} = e. \tag{6.5}$$

Die Ableitung hängt also ausschließlich von der Wahl der Kostenfunktion ab. $\frac{\partial e}{\partial f'} = 1$ ist die Ableitung der Differenz e zur Prädiktion im Kamerabild f'. $\frac{\partial f'}{\partial f}$ ist die Ableitung der Prädiktion im Kamerabild, bezogen auf die Prädiktion im RD-Gitter, und hängt von der Wahl der Warping-Schicht ab und wird später definiert. Zuletzt ist $\frac{\partial f}{\partial \boldsymbol{\theta}}$ der Gradient der Prädiktion im RD-Gitter bezogen auf die Netzwerkparameter.

In dieser Arbeit wird das Warping von Radardaten aus dem RD-Gitter und dem RAD-Gitter untersucht. Beim Ersten werden ausschließlich Entfernungs- und Geschwindigkeitsinformation von der Referenzsensorik genutzt, um die Position eines Pixels aus dem Kamerabild im RD-Gitter zu bestimmen. Beim Zweiten wird zusätzlich noch der Einfallswinkel verwendet, um die Pixel- bzw. Voxelposition im RAD-Gitter zu bestimmen.

Im Folgenden wird definiert, wie Vorwärts- und Rückwärtswarp durchzuführen sind. Dabei wird insbesondere auch Pixelinterpolation beschrieben. Pixelinterpolation ist notwendig, wenn, so wie in Abbildung 6.1 gezeigt, das Warping nicht exakt auf ein Pixel im Quellbild zeigt und die Pixelindizes somit keine Ganzzahlen, sondern Fließkommazahlen sind.

6.3. Vorwärtswarp

Beim Vorwärtswarp wird hier ausschließlich die nächste Nachbarinterpolation verwendet. Bei der Nächster-Nachbar-Interpolation (NNI) (engl.: „nearest neighbour interpolation“) handelt es sich um eine der einfachsten Formen von Interpolation. Zeigt das Warping nicht exakt auf ein Pixel, so werden die Pixelindizes auf die nächste Ganzzahl gerundet und als Interpolationswert somit der Wert des geometrisch nächsten Nachbarn ausgegeben. Für den interessierten Leser sei als weiterführende Literatur [Han13] empfohlen.

6.3.1. Vorwärtsdurchlauf

Beim Vorwärtswarp ergibt sich ein Vorwärtsdurchlauf der Schicht, wie in Algorithmus 3 beschrieben.

Algorithmus 3: Vorwärtswarp - Vorwärtsdurchlauf

Daten: Kamerabild $\mathbf{C}_\mathrm{s} \in \mathbb{C}^{N_s \times M_s}$, Geschwindigkeitsbild $\mathbf{V}_r \in \mathbb{R}^{N_s \times M_s}$, Entfernungsbild $\mathbf{R} \in \mathbb{R}^{N_s \times M_s}$ und Auflösung $\{\Delta r, \Delta r\}$
Ergebnis: Zielbild $\mathbf{U}_\mathrm{t} \in \mathbb{C}^{N_t \times M_t}$

$\mathbf{U}_\mathrm{t} \leftarrow 0$
für $u_s \leftarrow 0$ **bis** $N_s - 1$ **tue**
  für $v_s \leftarrow 0$ **bis** $M_s - 1$ **tue**
    $u_t \leftarrow \mathrm{round}\left(\frac{V_{r,[u_s,v_s]}}{\Delta v_r}\right)$
    $v_t \leftarrow \mathrm{round}\left(\frac{R_{[u_s,v_s]}}{\Delta r}\right)$
    wenn $(u_t >= 0)$ & $(v_t >= 0)$ & $(u_t < N_t)$ & $(v_t < M_t)$ **dann**
      $U_{\mathrm{t}[u_t,v_t]} += C_{\mathrm{s}[u_s,v_s]}$
return $\mathbf{U}_\mathrm{t}$

Die Grauwerte im Zielbild ergeben sich durch Akkumulation der assoziierten Pixel aus dem Quellbild.

6.3.2. Rückwärtsdurchlauf

Die partielle Ableitung ergibt sich beim Vorwärtswarp an Pixeln, welche zwischen Quell- und Zielbild assoziiert wurden, zu eins. Ansonsten wird der Gradient genullt, siehe Algorithmus 4.

Algorithmus 4: Vorwärtswarp - Rückwärtsdurchlauf

Daten: Kamerabild $\mathbf{C}_\mathrm{s} \in \mathbb{C}^{N_s \times M_s}$, Geschwindigkeitsbild $\mathbf{V}_r \in \mathbb{R}^{N_s \times M_s}$, Entfernungsbild $\mathbf{R} \in \mathbb{R}^{N_s \times M_s}$ und Auflösung $\{\Delta v_r, \Delta r\}$

Ergebnis: Gradienten $\frac{\delta f'}{\delta f} \in \mathbb{C}^{V_t \times U_t}$

$\frac{\delta f'}{\delta f} \leftarrow 0$

für $u_s \leftarrow 0$ **bis** $N_s - 1$ **tue**

  für $v_s \leftarrow 0$ **bis** $M_s - 1$ **tue**

    $u_t \leftarrow \mathrm{round}\left(\frac{V_{r,[u_s,v_s]}}{\Delta v_r}\right)$

    $v_t \leftarrow \mathrm{round}\left(\frac{R(u,v)}{\Delta r}\right)$

    wenn $(u_t >= 0)$ & $(v_t >= 0)$ & $(u_t < N_t)$ & $(v_t < M_t)$ **dann**

      $\frac{\delta f'}{\delta f}(u_s, v_s) = 1$

    sonst

      $\frac{\delta f'}{\delta f}(u_s, v_s) = 0$

return $\frac{\delta f'}{\delta f}$

6.4. Rückwärtswarp

Beim Rückwärtswarp haben sich im Rahmen dieser Arbeit drei mögliche Interpolationsverfahren angeboten, welche im Folgenden kurz vorgestellt werden.

6.4.1. Nächster-Nachbar-Interpolation (NNI)

Auch beim Rückwärtswarp kann die NNI verwendet werden.

6.4.1.1. Vorwärtsdurchlauf

Die NNI beim Rückwärtswarp kann folgendermaßen in Gleichungsform beschrieben werden:

$$f'(u,v) = \begin{cases} f(\lfloor u \rceil, \lfloor v \rceil) & u \in U, v \in V \\ 0 & \text{sonst} \end{cases} \tag{6.6}$$

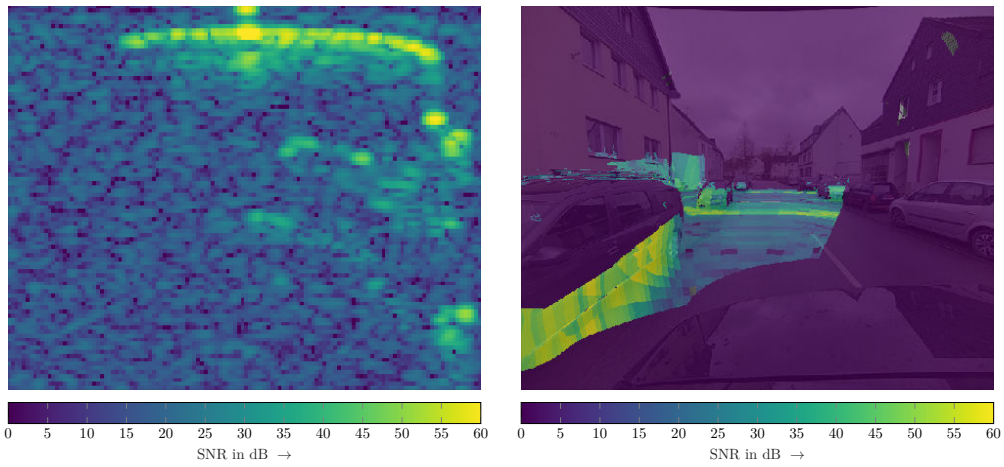
Hierbei ist f der Grauwert an der Pixelposition u, v. $\lfloor \cdots \rceil$ entspricht der Gaußklammer als Rundungsoperation. Durch die Fallunterscheidung erzeugen Pixel außerhalb des Quellbildes einen Interpolationswert von 0.

Um die Interpolationswerte für das Warping zu bestimmen, werden für jedes Pixel im Kamerabild nun zunächst die radiale Entfernung und Geschwindigkeit im Radar-Koordinatensystem bestimmt. Eine Beschreibung hierzu wurde bereits in Gleichung 5.27 gegeben. Durch das Einsetzen der RD-Gitter-Pixelposition $\mathbf{p}_r(\mathbf{p}) = \left[u_r(\mathbf{p}), v_r(\mathbf{p})\right]^T$ in Gleichung 6.6 ergibt sich schließlich der Helligkeitswert für das Kamerapixel $\mathbf{p}$ zu

$$U_{C[\mathbf{p}]} = \begin{cases} U_{[\lfloor u_r(\mathbf{p}) \rceil, \lfloor v_r(\mathbf{p}) \rceil]} & 0 \le u_r < U_r, 0 \le v_r < V_r \\ 0 & \text{sonst} \end{cases} \tag{6.7}$$

wobei U_r und V_r die Dimensionen des RD-maps sind. Wir verwenden $\mathbf{RD}_C$ nun als Synonym für das in das Kamerabild gewarpte RD-map.

Ein stellvertretendes Beispiel für das Warping mittels NNI ist in Abbildung 6.4 dargestellt.



(a) RD-map (b) Gewarptes RD-map auf Kamerabild

Abbildung 6.4.: **Warping des RD-map auf Kamerabild mittels NNI:** Die RD-map auf der linken Seite, wurde durch Anwendung der Interpolation in das Kamerabild gewarpt. Die Positionen in der RD-map ergeben sich aus der zuvor berechneten relativen Entfernung und Geschwindigkeit der Kamerapixel, transformiert in das Radarkoordinatensystem. Pixel im Kamerabild außerhalb des Radar FoV wurden genullt.

Zu erkennen ist in Abbildung 6.4 b), dass den parkenden Fahrzeugen hohe Leistungswerte aus der RD-map zugewiesen wurden. Deutlich geringe Leistung wurde der Fahrbahnoberfläche zugewiesen. Beides wird als plausibel empfunden (vgl. Unterabschnitt 2.1.1). Deutlich zu erkennen ist außerdem eine Blockbildung der Intensitätswerte in Abbildung 6.4 b), z.B. mittig links am parkenden Fahrzeug. Als Blockbildung wird hierbei verstanden, dass benachbarte Pixel exakt die gleiche Intensität aufweisen. Dieser Effekt wird in der Literatur auch als „Mosaik Phänomen“ oder „Mosaik Effekt“ (engl.: „mosaic phenomen“) bezeichnet [Han13] und häufig im Zusammenhang mit der NNI genannt.

6.4.1.2. Rückwärtsdurchlauf

Analog wie bei max-pooling-Schichten [SMB10] aus NN, gilt bei der NNI das „winner takes it all“ Prinzip. Anders als beim max-pooling entscheidet dabei jedoch nicht der Wert der Neuronenaktivierung, sondern die Pixeldistanz in der Nachbarschaft, welches Pixel bzw. dessen Inhalt weiter im Vorwärtsdurchlauf berücksichtigt wird. Im Rückwärtsdurchlauf wird der Fehlergradient dementsprechend auch ausschließlich für das siegreiche Pixel propagiert. Es ergibt sich folgender Gradient über die NNI Schicht:

$$\frac{\partial f'}{\partial f} = \begin{cases} 1 & u = \lfloor u \rceil, v = \lfloor v \rceil \\ 0 & \text{sonst} \end{cases} . \tag{6.8}$$

6.4.2. Bilineare Interpolation

Bei der NNI wurden Pixel aus dem Kamerabild immer auf das Gitter des RD-maps gerundet, wodurch sich Blockbildungen im interpolierten Bild ergaben. Als eine mögliche Alternative benennt die Literatur die bilineare-Interpolation (BI) [Han13]. Hiermit lassen sich Zwischenwerte im zweidimensionalen Gitter interpolieren. Dabei wird der Grauwertgradient zwischen Pixelnachbarn ermittelt und basierend darauf Zwischenwerte approximiert.

6.4.2.1. Vorwärtsdurchlauf

Nach [Han13] lässt sich die Interpolation in folgende Gleichung überführen

$$f'(u,v) = \begin{bmatrix} 1-u' & u' \end{bmatrix} \underbrace{\begin{bmatrix} f(\lfloor u \rfloor, \lfloor v \rfloor) & f(\lceil u \rceil, \lfloor v \rfloor) \\ f(\lfloor u \rfloor, \lceil v \rceil) & f(\lceil u \rceil, \lceil v \rceil) \end{bmatrix}}_{\text{Grauwertnachbarschaft}} \begin{bmatrix} 1-v' \\ v' \end{bmatrix} \tag{6.9}$$

$$\begin{aligned} = \; & f(\lfloor u \rfloor, \lfloor v \rfloor)(1-u')(1-v') + f(\lceil u \rceil, \lfloor v \rfloor)(1-u')v' \\ & + f(\lfloor u \rfloor, \lceil v \rceil)u'(1-v') + f(\lceil u \rceil, \lceil v \rceil)u'v', \end{aligned} \tag{6.10}$$

wobei $\lfloor \cdots \rfloor$ und $\lceil \cdots \rceil$ die Gaußklammern als Operator für Ab- und Aufrunden sind. Der gesuchte Zielwert befindet sich an der Stelle u, v. Die Parameter $u' = u - \lfloor u \rfloor$ und $v' = v - \lfloor v \rfloor$ sind die Dezimalwerte der Pixelposition. Das in das Kamerabild gewarpte RD-map ergibt sich mittels BI zu

$$U_{C[\mathbf{p}]} = \begin{bmatrix} 1-u' & u' \end{bmatrix} \begin{bmatrix} U_{[\lfloor u \rfloor, \lfloor v \rfloor]} & U_{[\lceil u \rceil, \lfloor v \rfloor]} \\ U_{[\lfloor u \rfloor, \lceil v \rceil]} & U_{[\lceil u \rceil, \lceil v \rceil]} \end{bmatrix} \begin{bmatrix} 1-v' \\ v' \end{bmatrix}. \tag{6.11}$$

Ein stellvertretendes Beispiel für das Warping mittels BI ist in Abbildung 6.5 dargestellt.

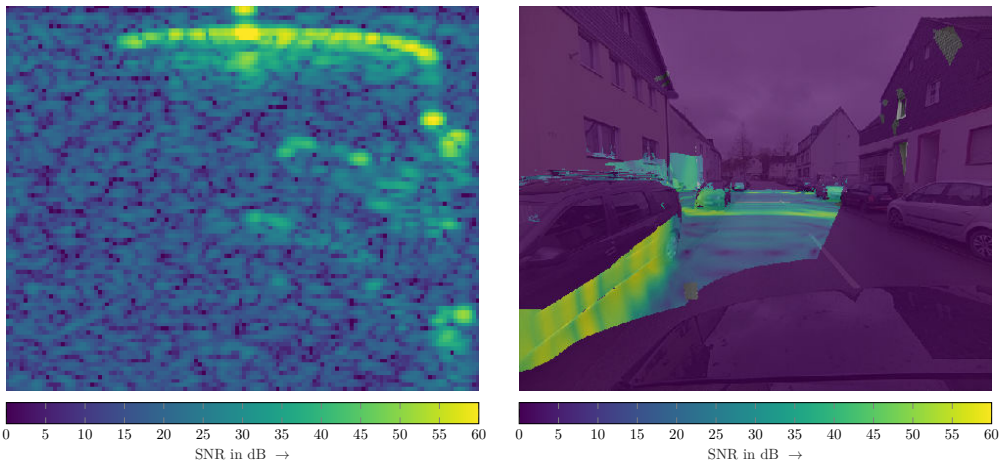
Im Vergleich zu Abbildung 6.4 ist kein „Mosaik-Effekt“ mehr zu erkennen und der Helligkeitsverlauf im gewarpten RD-map ist kontinuierlich. Wie zuvor auch wurden Pixel außerhalb des FoV des Radars genullt.

6.4.2.2. Rückwärtsdurchlauf

Der Gradient $\frac{\partial f'}{\partial f}$ kann über Gleichung 6.10 abgeleitet werden zu

$$\frac{\partial f'}{\partial f} = \begin{bmatrix} \frac{\delta f'}{\delta f(\lfloor u \rfloor, \lfloor v \rfloor)} \\ \frac{\delta f'}{\delta f(\lceil u \rceil, \lfloor v \rfloor)} \\ \frac{\delta f'}{\delta f(\lfloor u \rfloor, \lceil v \rceil)} \\ \frac{\delta f'}{\delta f(\lceil u \rceil, \lceil v \rceil)} \end{bmatrix} = \begin{bmatrix} (1-u')(1-v') \\ (1-u')v' \\ u'(1-v') \\ u'v' \end{bmatrix}. \tag{6.12}$$

Anders als bei der NNI ergibt sich bei der BI ein Gradient für alle beteiligten Pixel. Hieraus ergibt sich möglicherweise ein praktischer Vorteil gegenüber NNI, denn wenn die Warpingparameter (Szenenfluss und Entfernung) marginal fehlerhaft sind, kann das tatsächliche Pixel im RD-Gitter durch BI dennoch getroffen und somit trainiert werden.



(a) RD-map (b) Gewarptes RD-map auf Kamerabild

Abbildung 6.5.: **Warping des RD-map auf Kamerabild mittels BI:** Die RD-map auf der linken Seite, wurde durch Anwendung der Interpolation in das Kamerabild gewarpt. Die Positionen in der RD-map ergeben sich aus der zuvor berechneten relativen Entfernung und Geschwindigkeit der Kamerapixel, transformiert in das Radarkoordinatensystem. Pixel im Kamerabild außerhalb des Radar FoV wurden genullt.

6.4.3. Trilineare Interpolation

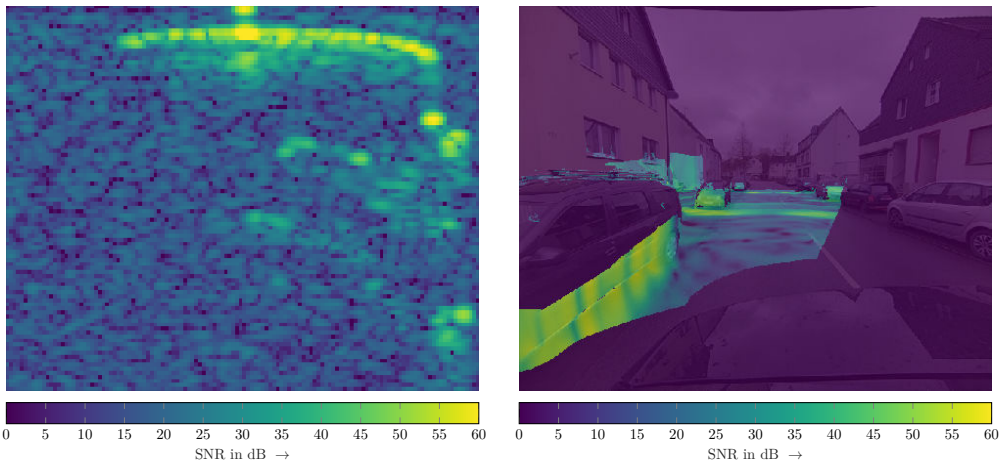
In Unterunterabschnitt 2.1.5.2 wurde gezeigt, dass bei Verwendung mehrerer Antennen neben dem RD-Gitter noch eine weitere Auflösungsdimension ermittelt werden kann. Es ergibt sich der RAD-Raum. Um auch hieraus Werte in das Kamerabild warpen zu können, lässt sich die BI trivial zur trilinearen-Interpolation (TI) um eine Dimension erweitern. Vorwärts- und Rückwärtsdurchlauf der TI Schicht können analog zur BI hergeleitet werden. Da hierbei jedoch 8 Pixel berücksichtigt werden müssen, verzichten wir aus Gründen der Übersicht auf die Darstellung von Vorwärts- und Rückwärtsdurchlauf der Schicht.

Ein stellvertretendes Beispiel für das Warping mittels TI ist in Abbildung 6.6 dargestellt.

Wie auch bei der BI, Abbildung 6.5 sind keine Blockbildungen zu beobachten. Weiterhin fällt auf, dass in Regionen mit niedrigerem SNR, z.B. der Straße, geringere Helligkeiten erreicht wurden als bei der BI. Dieser Effekt ergibt sich durch den SNR Gewinn des angewendeten Beamformers.

Ebenfalls fällt auf, dass die Leistung im Bereich des hinteren Fahrzeuges linksseitig in Abbildung 6.6 fokussierter ist, als an gleicher Position in Abbildung 6.5. Die Leistung konzentriert sich dabei mehr auf die eigentliche Position des Fahrzeuges und etwas weniger auf umliegende Pixel (Straße) mit vergleichbarer Entfernung und Geschwindigkeit. Da nun der Einfallswinkel im Radarspektrum berücksichtigt wird, ist diese Beobachtung plausibel.

Obwohl der Unterschied zwischen bi- und trilinearer Interpolation hier subjektiv kaum bemerkbar ist, erhofft sich der Autor, dass bei zukünftigen Radargenerationen mit



(a) RD-map (b) Gewarptes RAD-map auf Kamerabild

Abbildung 6.6.: **Warping des RD-map auf Kamerabild mittels TI:** Die RD-map auf der linken Seite, wurde durch Anwendung der Interpolation in das Kamerabild gewarpt. Die Positionen in der RD-map ergeben sich aus der zuvor berechneten relativen Entfernung und Geschwindigkeit der Kamerapixel, transformiert in das Radarkoordinatensystem. Pixel im Kamerabild außerhalb des Radar FoV wurden genullt.

verbessertem Azimut-Nebenkeulendämpfung (engl.: „sidelobe level“)[2] ein merklicher Unterschied bemerkbar sein wird.

6.5. Subjektive Bewertung

Nachdem die drei Interpolationsmethoden vorgestellt und die Differenzierbarkeit der Warping-Schichten nachgewiesen wurde, wollen wir nun eine Bewertung vornehmen. In [Han13] fand eine Bewertung der Verfahren anhand von Bildrekonstruktion künstlich verkleinerter Bilder statt. Es wurde also ein Bild verkleinert und anschließend durch die Interpolationsverfahren wieder auf die Originalgröße skaliert, anschließend die Grauwerte des originalen Bildes mit skalierten Bildern verglichen und ein SNR gebildet. Dieses Vorgehen war nur möglich, weil der Zielwert durch das originale Bild gegeben ist. In unserem Fall ist dies leider nicht möglich, weil der Zielwert im Kamerabild nicht bekannt ist. Statt eines quantitativen Vergleichs wollen wir in diesem Abschnitt die offensichtlichen Unterschiede und Probleme diskutieren.

Im Vergleich zur BI ergaben sich bei der NNI deutliche Artefakte durch Blockbildung. Diese Artefakte entstehen ausschließlich durch die Interpolationsmethode und sind auch in anderen Anwendungen bekannt [Han13]. Da die Artefakte bei der BI und TI nicht zu beobachten sind, sind diese beiden Verfahren aus Sicht des Autors zu präferieren. Ein weiterer bereits genannter theoretischer Vorteil von BI und TI ist, dass sich der Helligkeitswert eines Pixels im Kamerabild aus 4 bzw. 8 benachbarten Pixeln im RD bzw. RAD-Gitter zusammensetzt. Bei marginalen Fehlern in der Bestimmung der Warpingparameter (Szenenflusss und Entfernung) ergibt sich eine erhöhte Chance,

[2] Charakteristika zur Antennenspezifikation

das tatsächliche Pixel aus dem RD bzw. RAD-Gitter zu treffen. Dieser Effekt mag einen praktischen Vorteil beim Training von NN ergeben, wird aber nicht weiter untersucht.

Unter Bijektivität bzw. Eineindeutigkeit versteht man in der Mengenlehre eine Abbildung, welche eindeutig zwischen Ursprungsmenge und abgebildeter Menge assoziiert. In unserem Fall ist das RD-Gitter die Ursprungsmenge und das ins Kamerabild gewarpte RD-Gitter die abgebildete Menge. Meist kann die Kamera einzelne Ziele aufgrund hoher Auflösung in einzelne Pixel trennen, im Radar fallen diese Bereiche jedoch noch in gemeinsame Zellen. Eine Bijektion ist dann nicht mehr gegeben. Mit den verwendeten Interpolationsverfahren hat dies zur Folge, dass die Leistungssumme in $\mathbf{RD}_C$ nun nicht mehr der ursprünglichen Leistungssumme aus **RD** entspricht. Elektrische Beiträge einzelner Pixel im Kamerabild werden somit sicher nicht korrekt abgebildet. Das fällt beispielsweise auf, wenn man die Leistungsprojektion der parkenden Fahrzeuge in Abbildung 6.7 betrachtet. Die Regionen, in welchen eine hohe Leistung in $\mathbf{RD}_C$ vorhanden ist, aber aufgrund schwacher Reflektoren nicht zu erwarten ist, wurden hier rot markiert.

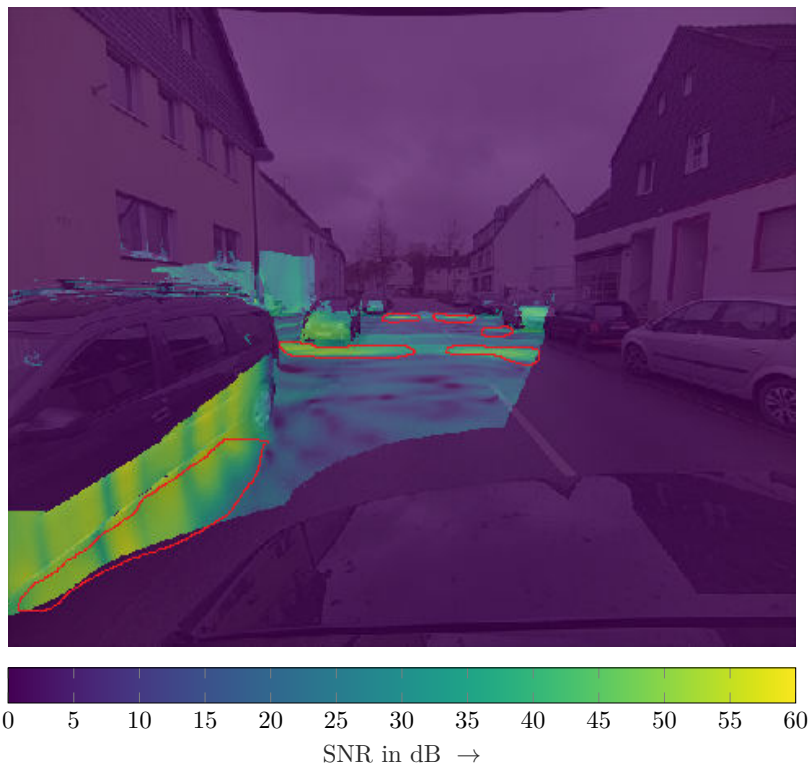


Abbildung 6.7.: **Kennzeichnung unplausibler Leistungswerte nach Warping des RD-map auf Kamerabild mittels TI:** Regionen, in denen eine unerwartet hohe Leistung vorhanden ist, wurden rot markiert.

Nach Sichtung der Daten ist aufgefallen, dass die Kamerapixel in den rot markierten Zonen häufig auf gleiche Pixel in der RD-map verweisen und somit vom Radar nicht in einzelne Pixel aufgelöst wurden. Fällt in diese nicht aufgelösten Pixel dann noch ein starker Reflektor, wie beispielsweise ein Fahrzeug, so wird durch die nicht vorhandene Bijektion der Projektion die Leistung des starken Reflektors in angrenzende Regionen

im Kamerabild projiziert. Auch bei Punktzielen, welche eine kleinere geometrische und kinematische Ausdehnung haben, als es der Radarsensor auflösen kann, gibt es Leistungsverteilungen im RD-map, bedingt durch z.B. Signalimperfektionen, Leakage bzw. Fensterungseffekte. Für die in Abbildung 6.7 rot markierten Bereiche genügt es also bereits, in einer direkten Pixelnachbarschaft im RD-map zu einem starken Reflektor zu liegen. Im Vergleich von BI und TI ergab sich bei TI eine verbesserte Fokussierung der Leistung auf das zweite parkende Fahrzeug und etwas weniger Leistungsprojektion auf die umliegende Straße. Kamerapixel, die bei der BI noch auf ein gleiches RD-Pixel verwiesen haben, verweisen nun auf unterschiedliche Pixel im RAD-Gitter.

6.6. Zusammenfassung

Mit dem in diesem Kapitel vorgestellten Warping wurden Werte aus RD oder RAD-Gitter in das Kamerabild projiziert. Zum einen lassen sich die Werte damit in einer anschaulichen Anordnung im Kamerabild visualisieren. Zum anderen lassen sich dadurch aber auch Annotationen im Kamerabild für das Training von ML-basierten Verfahren verwenden. Ein Nachweis für dieses Training wird in Kapitel 7 erbracht.

Beim Warping wurde festgestellt, dass durch die fehlende Bijektivität des Warpings teilweise unplausibel hohe Leistungen in Kamerapixeln beobachtet wurden. Dies ist technisch damit zu begründen, dass die Leistung aus RD und RAD-Gitter mehrfach auf unterschiedliche Kamerapixel zugeordnet wurde. Bei einem verbesserten Warping müsste sich die beobachtete Leistung eines RD oder RAD-Pixels auf die Kamerapixel verteilen. Im Idealfall würde dabei eine Verteilung entsprechend der elektrischen Beiträge der Kamerapixel erfolgen. Leider geben die Messwerte aus der Referenzsensorik keinen Rückschluss auf diese elektrischen Beiträge, so dass vorerst mit der vorgestellten Warping-Operation ausgekommen werden muss. Im weiteren Verlauf dieser Arbeit werden weitere Untersuchungen durchgeführt, um die elektrischen Beiträge weiter zu identifizieren (Unterabschnitt 7.3.2) und gar auf Kameradaten zu schätzen (Abschnitt 7.5). Als Synonym für die Rückwärts- bzw. Vorwärtswarp wird fortlaufend folgende Notation verwendet:

$$x_{\text{cam}} = \eta_{\text{BW}}\Big(x_{\text{RD-grid}}; \dot{r}, r\Big) \tag{6.13}$$

$$x_{\text{RD-grid}} = \eta_{\text{FW}}\Big(x_{\text{cam}}; \dot{r}, r\Big), \tag{6.14}$$

wobei $x_{\text{RD-grid}}$ eine Variable im RD-Gitter ist und x_{cam} die entsprechend ins Kamerabild gewarpte Variable. η ist Warpingfunktion und $\dot{r}$ und r sind relative radiale Geschwindigkeit und Abstand.

7. Evaluation

In den vorangegangenen Kapiteln wurden aus den Daten der Referenzsensorik durch Signalverarbeitung eine Tiefenvervollständigung und Szenenefluss geschätzt. Anschließend wurden damit die Pixel aus RD oder RAD-Gitter in das Kamerabild gewarpt, vor dem Hintergrund, diese mit Annotationen im Kamerabild vergleichen zu können. In diesem Kapitel werden wir diese Annotationen nutzen und nachweisen, dass damit eine NN-basierte Signalverarbeitung von Daten im RD-Gitter trainiert werden kann. Begonnen wird dabei mit der Vorstellung des eingefahrenen Datensatzes (Abschnitt 7.1). Anschließend werden NN zur Schätzung von Einfallswinkel (Abschnitt 7.2), Zieldetektion (Abschnitt 7.3) und semantischen Segmentierung (Abschnitt 7.4) untersucht. Als zusätzliche Untersuchung werden wir die in Abbildung 6.2 dargestellte Trainingshierarchie (Lehrer vs. Student) umkehren und die Daten aus Radar als Annotation verwenden, um ein NN operierend auf Kameradaten zu trainieren (Abschnitt 7.5).

7.1. Datensatz

Für die Akquise von realen Sensordaten wurde das in Kapitel 3 vorgestellte Fahrzeug samt Sensorik verwendet. Bei einer durchgehenden Fahrt durch Lippstadt, siehe Abbildung 5.7, wurden die Sensorsignale aufgezeichnet. Die Fahrt beinhaltete einen geringen Anteil von Szenen im Hella-Werk, viele Szenen auf öffentlicher Stadtstraße, Szenen auf öffentlicher Stadtautobahn sowie Szenen auf einem Parkplatzgelände. Insgesamt wurden Daten aus etwas mehr als 1 h Aufnahmelänge akquiriert. Dabei wurden etwa 36000 Frames a 10 FPS aufgezeichnet und dabei ungefähr $7 \cdot 10^9$ Datenpunkte auf Pixelebene der Kamera eingefahren.

7.1.1. Datensplit

Für das Training und die Validierung der nachfolgenden Algorithmen wurde der akquirierte Datensatz in 10 s lange Sequenzen a 100 Frames geteilt. Diese Sequenzen wurden zufällig in Trainings-, Evaluations- und Testdatenmengen geordnet. Der Datensplit betrug in etwa $70\%, 15\%, 15\%$. Durch die zeitliche Trennung der Sequenzen soll erreicht werden, dass statistisch unabhängige Beispiele in Trainings und Testdaten vorhanden sind und eine generalisierbare Aussagekraft der Auswertung entsteht.

7.1.2. Manuelle Annotation der Daten

Obschon die in den vorangegangenen Abschnitten vorgestellte Methodik zur automatischen Annotation der Daten dem Stand der Wissenschaft entsprechen mag, so sind Fehler bei der Annotation keineswegs auszuschließen. So wurde z.B. in Tabelle 5.2 festgehalten, dass Fehler im Szenenfluss bei etwa 30% der Objekte größer als die Auflösung des Radars waren. Dieser Wert wurde in Parkplatzszenen mit starker Überdeckung der

Objekte ermittelt. Bei einfacheren Szenarien kann der Fehlerwert also niedriger ausfallen. Gleichzeitig ergeben sich aber noch weitere Fehlermöglichkeiten, durch z.B. Fehler in der Instanzsegmentierung, welche durch diesen empirisch ermittelten Fehlerwert nicht berücksichtigt wurden und die Qualität der Labels des Datensatzes degradieren können. Diese Labelfehler haben einen negativen Einfluss auf die quantitative Größe der Bewertung der Verfahren. Nach manueller Durchsicht der Daten wurden im Wesentlichen drei verschiedene Fehlerbilder beobachtet, welche nachfolgend als Anomalien bezeichnet werden. Diese waren:

1. wenn ein stationäres Objekt fälschlicherweise als bewegtes Objekt erkannt wurde
2. Eine erhebliche Menge der Leistung aus der RD-map nicht in das Kamerabild projiziert wurde $\mathbf{RD}_C$
3. Fehlerhafte Segmentierung der Punktwolke, was zu einer unplausiblen Ausdehnung der geometrischen Signatur im RD-Gitter führt **RD**.

Beispiele zu den Fehlerfällen sind in Abbildung 7.1 dargestellt.

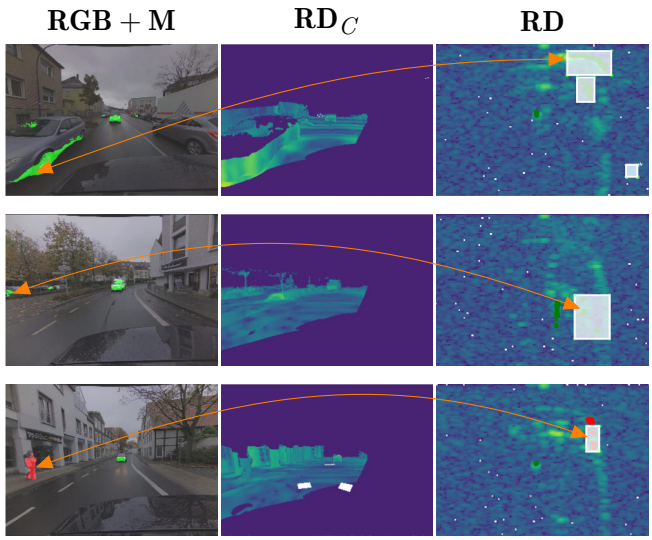


Abbildung 7.1.: **Beispiele von Anomalien im Datensatzlabeling:** Von links nach rechts: Kamerabild mit eingezeichneter Instanzsegmentierung (Rot: Fußgänger; Grün: Fahrzeug) im FoV der Radars, auf das Kamerabild gewarpte RD-map und RD-map mit projizierter Instanzmaske, sowie manuell eingezeichneter Anomaliemasken (weiße Boxen). Die orangen Pfeile deuten korrespondierende Regionen zwischen Kamerabild und RD-map an. In den ersten beiden Zeilen wurden parkende Objekte, aufgrund fehlerhafter Szenenflussschätzung, fälschlicherweise als bewegt detektiert (Fehlertyp 1). In der unteren Zeile ist eine fehlerhafte Instanzsegmentierung eingezeichnet. Dabei wurden Pixel des Hintergrundes als Fußgänger klassifiziert, was in einer unplausiblen geometrischen Ausdehnung in der RD-map resultiert (Fehlertyp 3). Nach [EB6].

In der ersten Spalte der Abbildung sind Kamerabilder mit den farblich eingezeichneten Masken für bewegte Fußgänger (rot) und bewegte Fahrzeuge (grün) dargestellt. In der

zweiten Spalte ist die ins Kamerabild projizierte Leistung aus der RD-map dargestellt. In der letzten Spalte ist die RD-map sowie die Signatur der bewegten Objekte dargestellt. Ebenfalls dargestellt sind die manuell erstellten Anomaliemasken, welche Regionen mit Labelfehlern kennzeichnen.

In der obersten Reihe ist zu erkennen, dass das Fahrzeug am linken Bildrand als bewegt identifiziert wurde, obwohl es sich hierbei um ein parkendes Fahrzeug handelte. Es wurde also offensichtlich eine falsche Geschwindigkeit des Fahrzeuges geschätzt. Bei genauer Betrachtung des RD-maps ist zu erkennen, dass die Signatur des entsprechenden Fahrzeuges nur wenige Pixel in Doppler-Richtung von der stationären Bebauung abweicht. Die Geschwindigkeitsabweichung ist also nur marginal. Trotzdem wurde in diesem Beispiel die Region als fehlerhaft markiert und aus dem Testdatensatz ausgeschlossen. Eine ähnliche Beobachtung wurde bei dem Objekt am linken Bildrand aus der zweiten Bildreihe gemacht. In der letzten Bildreihe ist zu erkennen, dass die Signatur des Fußgängers (rot) in der RD-map eine unplausibel große geometrische Ausdehnung aufweist. Umgerechnet entspräche dies einer mehrere Meter langen radialen Ausdehnung. Zur Maskierung der Anomalie wurde die Box über die als fehlerhaft identifizierte Region in der RD-map gezeichnet.

Glücklicherweise sind diese Anomalien durch einen menschlichen Beobachter leicht zu identifizieren und manuell zu markieren. Diese manuelle Annotation der Anomalien wurde für den gesamten Testdatensatz durchgeführt. Dem Labeler[1] wurden die Daten analog zur Abbildung 7.1 im Labelingframework [Tzu15] präsentiert. Die Frames wurden einzeln in der Reihenfolge der Aufzeichnung dargestellt. Der Labeler konnte mit Tastaturbefehlen schnell zwischen benachbarten Frames dirigieren, um ein besseres Verständnis der Szenendynamik zu erlangen. So war es z.B. einfacher nachzuvollziehen, welche Objekte geparkt waren und wie sich die Labels über die Frames entwickelten. Wurde eine Anomalie erkannt, so wurde manuell eine Box in Kamerabild oder RD-map eingetragen. Die Größe der Box wurde so gewählt, dass die fehlerfreien Daten des Frames erhalten blieben, jedoch keine fehlerhaften Labels verblieben. Wurden diese Masken eingezeichnet, so wurden die entsprechenden Regionen im Testdatensatz exkludiert.

In Summe wurde bei den etwa 10800 Frames des Testdatensatzes in etwa 20.3% der Frames mindestens eine Anomalie erkannt. Die durchschnittliche Zeit zum Labeln eines Frames betrug etwa 5 s. Durch die Verwendung der in dieser Arbeit vorgestellten Verfahren wurde ausschließlich das manuelle Labeln von Anomalien durchgeführt. Die Labelingaufgabe beschränkte sich somit auf eine binäre Klassifikation, bei welcher der menschliche Labeler entscheiden musste, ob das vorgeschlagene Label plausibel oder fehlerhaft ist. Ohne diese Methodik hätte beispielsweise für jedes Pixel in der RD-map ein Label, beispielsweise für Winkelschätzung, manuell vorgegeben werden müssen. Es ist anzunehmen, dass durch das vorgestellte Verfahren eine erhebliche Reduktion des manuellen Aufwandes erreicht werden konnte.

Es sei wiederholt, dass Anomaliemasken ausschließlich für den Testdatensatz gezeichnet wurden. Das im Nachfolgenden beschriebene Training der NN wurde auf den Trainingsdaten durchgeführt.

In Abbildung 6.7 wurden weitere Fehler, verursacht durch die Warpingschicht, dargestellt. Diese wurden bei der manuellen Anotation nicht berücksichtigt und werden stattdessen bei der Auswertung Beachtung finden.

[1] Der Begriff Labeler wurde hier als Synonym für den Autor dieser Arbeit verwendet.

7.2. DoA Schätzung

In erster Anwendung wird das Training eines NNes zur Azimutwinkelschätzung von Reflexionen in den RD-Gittern durchgeführt. Es handelt sich dabei um eine Regressionsaufgabe, bei der eine kontinuierliche Zielvariable zu schätzen ist. In diesem Abschnitt werden wir die Eingangsdaten (Unterabschnitt 7.2.1), die Zielwerte (Unterabschnitt 7.2.2) und Netzwerkarchitekturen (Unterabschnitt 7.2.3) diskutieren, danach die Fehlermetrik definieren (Unterabschnitt 7.2.4 - 7.3.7), die Initialisierung (Unterabschnitt 7.2.8) und den Optimierer des NN (Unterabschnitt 7.2.9) aufzeigen. Abschließend werden der Trainingsprozess (Unterabschnitt 7.2.10) und die Ergebnisse unter Testdaten ermittelt und diskutiert (Unterabschnitt 7.2.11).

7.2.1. Eingangsdaten

Das NN zur Winkelschätzung operiert auf den Eingangsdaten $\boldsymbol{\Gamma}$, welche aus den Frequenzspektren des Radars bestehen. Dazu wurden drei Receiver (Rx)-Kanäle in Frequenzraumdarstellung $\{S_{\text{rd},1}, S_{\text{rd},2}, S_{\text{rd},3}\}$ des Radarsensors mit nicht äquidistantem Antennenabstand wie folgt aufbereitet, bevor sie in das NN eingespeist wurden.

$$\boldsymbol{\Gamma} = \begin{bmatrix} RD \\ \angle\left(S_{\text{rd},2}, S_{\text{rd},1}\right) \\ \angle\left(S_{\text{rd},3}, S_{\text{rd},2}\right) \end{bmatrix} \tag{7.1}$$

Der erste Eingangskanal ist die RD-map, wohingegen als zweiter und dritter Eingangskanal die Differenzen der Phasenspektren $\angle\left(S_{\text{rd,i}}, S_{\text{rd,i-1}}\right)$ zwischen den benachbarten Antennenpaaren ist.

7.2.2. Zielwerte

Für das Training des NNes zur Winkelschätzung müssen neben den Eingangsdaten ebenfalls die Zielwerte bereitgestellt werden. Das NN wird dann so optimiert, dass es versucht, diese Zielwerte aus den Eingangsdaten zu prädizieren.

Bei der Winkelschätzung ist der Einfallswinkel der Reflexionen zu schätzen, siehe bspw. Abbildung 2.7. Um den Einfallswinkel als Zielwert automatisch aus den Daten der Referenzsensorik zu ermitteln, interpretieren wir nun jedes Pixel im Kamerabild als Reflexion. Mit Hilfe von Gleichung 5.20 lässt sich aus den Pixelpositionen und Tiefeninformationen im Kamerakoordinatensystem eine 3D-Punktwolke bilden. Diese Punktwolke wird nun in das Radarkoordinatensystem transformiert, um die Perspektive des Radars nachzustellen, siehe Gleichung 3.33. Der Einfallswinkel in Azimutrichtung wird nun aus der Position in Radarkoordinaten bestimmt zu

$$\phi_{\text{label}}(\mathbf{p}) = \arctan\left(\frac{\mathbf{x}_{R[y][\mathbf{p}]}}{\mathbf{x}_{R[x][\mathbf{p}]}}\right). \tag{7.2}$$

Ebenso kann der Einfallswinkel in Elevationsrichtung ermittelt werden. Als Nachweis für die Funktionsfähigkeit des überwachten Trainings der in dieser Arbeit vorgestellten Pipeline genügt uns die erste Winkelrichtung, so dass $\phi_{\text{label}}(\mathbf{p})$ die Zielwerte der

Winkelschätzaufgabe ausmachen.

7.2.3. Netzwerkarchitektur

Als Netzwerk wurde ein CNN aufgebaut, welches die drei Eingangskanäle entgegennimmt und dann sukzessive in den Netzwerkschichten verarbeitet. Die Dimension der Kanäle wurde über die Netzwerkschichten konstant gehalten, jedoch die Anzahl der Kanäle in den Zwischenschichten variiert. Als Schrittweite (engl.: „stride“) und Füllung (engl.: „padding“) wurde der Einheitsschritt bzw. Gleichfüllung (engl.: „same“) verwendet. In den Zwischenschichten wurden als Aktivierungen LeakyReLUs verwendet. In der letzten Netzwerkschicht wurde eine Tangens-hyperbolicus-Aktivierungsfunktion mit Skalierung auf den Zielwertebereich $[-90, 90]^\circ$ angewendet. Eine grafische Übersicht der NN-Architektur ist in Abbildung 7.2 dargestellt.

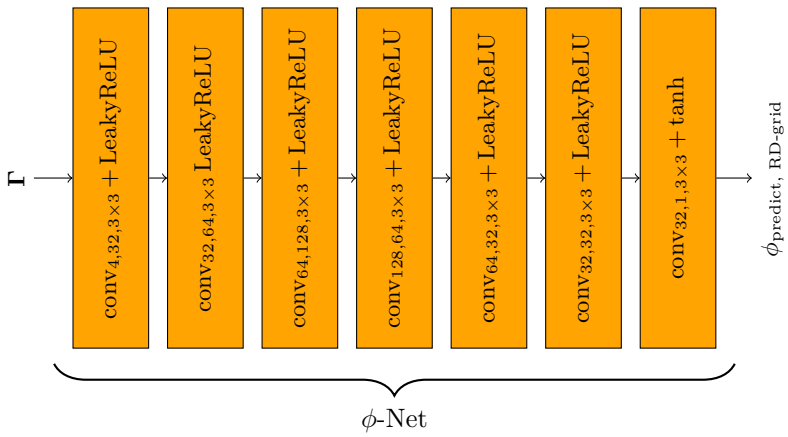


Abbildung 7.2.: **CNN Struktur für Winkelschätzung:** Die Übersicht der Schichten. In jeder Schicht ($\mathrm{conv}_{N_\mathrm{in},N_\mathrm{out},K\times K} + F$) werden Faltung der N_in Eingangskanäle auf N_out Zielkanäle durchgeführt. Dabei werden $K \times K$ Kernel verwendet. Abschließend die mit F spezifizierte Aktivierungsfunktion angewendet.

Wir definieren die Übertragungsfunktion des NN zur Winkelprädiktion als

$$\phi_{\text{predict, RD-grid}} = \phi\text{-Net}(\boldsymbol{\Gamma}). \tag{7.3}$$

Das oben definierte Netzwerk zur Winkelprädiktion hat ausgehend von einem Pixel aus der RD-map mit einer Größe von etwa $0.25\,\mathrm{m} \times 0.25\,\mathrm{m\,s^{-1}}$ ein rezeptives Feld von etwa $3.75\,\mathrm{m} \times 3.75\,\mathrm{m\,s^{-1}}$. Bei der Faltung über die Eingangsdaten wird die NN Prädiktion jedes Pixels also durch die in dem rezeptiven Feld definierte Nachbarschaft an Pixeln beeinflusst. Um zu testen, ob diese Nachbarschaft einen positiven Einfluss auf die Winkelprädiktion hat, wurde zum Vergleich das oben definierte Netzwerk ebenfalls mit 1×1 anstelle von 3×3 Faltungskernen gebaut. Das rezeptive Feld des zweiten NNes beschränkt sich dabei auf ein Pixel der Eingangsdaten, womit die Prädiktion unabhängig von den Werten der Pixelnachbarschaft erfolgt, analog zu den klassischen Verfahren aus Unterabschnitt 2.1.5. Um die Parameteranzahl beider NN-Architekturen identisch zu halten, wird die Kanalzahl des 1×1 NN in den Zwischenschichten verdreifacht

(Schichtmodifizierer: $t = 3$). Die in Abbildung 7.2 dargestellte Architektur entspricht also dem 3×3 NN. Das 1×1 NN ist analog aufgebaut, jedoch mit anderer Kanalzahl. Auf eine analoge Darstellung wird an dieser Stelle verzichtet. Eine Übersicht der NN Architekturen ist in Tabelle 7.1 dargestellt.

Tabelle 7.1.: **NN Architekturen für Winkelschätzung**: Übersicht der Parameter der Netzwerke.

Name	Größe der Faltungskerne	Schichtmodifizierer (t)	Parameterzahl
NN (1x1)	1×1	3	$\approx 195k$
NN (3x3)	3×3	1	$\approx 195k$

7.2.4. Assoziation von Prädiktion und Label durch Warping

Da Prädiktion im RD-Gitter durchgeführt wurde, die Zielwerte jedoch im Format des Kamerabildes vorliegen, werden die Prädiktionen für den Vergleich in das Kamerabild projiziert

$$\phi_{\text{predict, cam}}(\mathbf{p}) = \eta_{\text{BW}}\Big(\phi_{\text{predict, RD-grid}}; v_r(\mathbf{p}), |\mathbf{x}_r(\mathbf{p})|\Big). \tag{7.4}$$

Zur Projektion der Prädiktionen in das Kamerabild wurde die 2D-bilineare Interpolation verwendet. Die 3D-trilineare Interpolation wurde nicht verwendet, weil eine Winkelschätzung basierend auf den Phasendifferenzen der Antennenpaare gewollt war.

7.2.5. Messung der Abweichung

Durch die Projektion der Prädiktion liegen die Schätzwerte nun im Kamerabild vor und können nun pixelweise direkt mit den Zielwerten verglichen werden. Als Metrik für die Abweichung zum Zielwert wird die Charbonnier-Abweichung (siehe bspw. [Bar19]) bestimmt

$$l_{\text{DOA}}(\mathbf{p}) = \sqrt{(\phi_{\text{label}}(\mathbf{p}) - \phi_{\text{predict, cam}}(\mathbf{p}))^2 + 10^{-6}}. \tag{7.5}$$

Die Charbonnier-Abweichung ist eine differenzierbare Approximation der Betragsabweichung und bestraft Abweichung quasi linear. Sie wurde gewählt, da kein statistisches Modell des Winkelfehlers vorhanden war, welches eine andere Wahl begründet hätte.

7.2.6. Selektion der Pixelmenge

Bedingt durch unterschiedliche FoVs von Kamera und Radar dürfen für die Optimierung der Kosten aus Gleichung 7.5 nicht alle Pixel aus Kamera und RD-map verwendet werden. Es gilt vorab zu entscheiden, welche Reflexionen in beiden Sensoren sichtbar sind. Dafür wurde eine Selektion anhand von geometrischen Bedingungen aus den Daten der Referenzsensorik durchgeführt. Daneben wird eine zusätzliche Selektion anhand des SNRs aus der RD-map getestet. Motivation dafür ist, dass viele Pixel in der RD-map Rauschpixel sind und das Training des NN stören können.

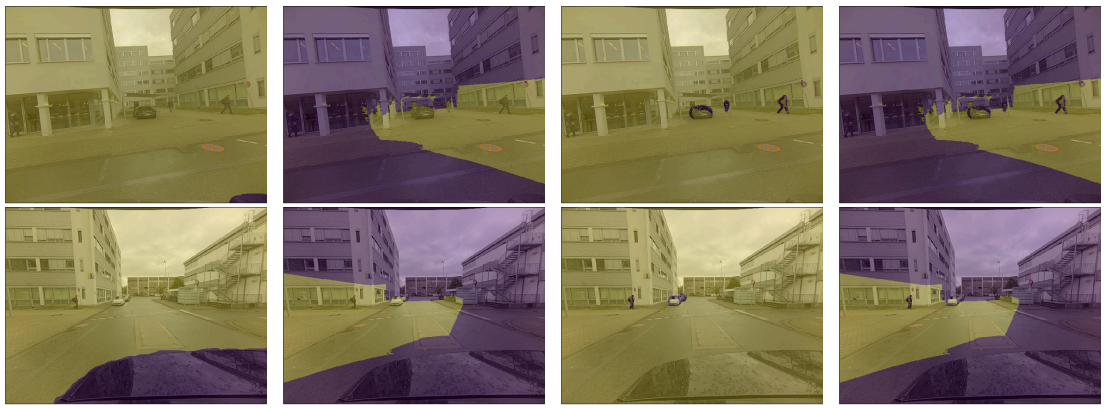
7.2.6.1. Selektion anhand von Daten aus Referenzsensorik

Für die Optimierung der Kosten aus Gleichung 7.5 werden die Abweichungen für die Pixel berücksichtigt, welche folgende Bedingungen erfüllen:

1. **Keine Ego-Reflexion**: Aufgrund der unterschiedlichen Einbauposition von Kamera und Radar bildet die Kamera das Ego-Fahrzeug wesentlich deutlicher ab, als es der Radarsensor tun würde. Es werden daher alle zum Ego-Fahrzeug gehörenden Pixel aus der Optimierungsschleife entfernt. Praktisch wurde dazu manuell eine stationäre Maske $\mathbf{P}_{\text{Ego}}$ gezeichnet, welche diese Pixelmenge beinhaltet. Ein Beispiel dieser Maske ist in Abbildung 7.3a zu sehen, wobei die dunkel gefärbten Pixel den maskierten Pixeln entsprechen.

2. **Sichtbar im Radar FoV**: Das FoV von Kamera und Radar unterscheiden sich deutlich, so dass ausschließlich Pixel in der Optimierung berücksichtigt werden, welche potenziell im FoV des Radars liegen würden. Dazu wurden zunächst die Azimut (vgl. Gleichung 7.2) und Elevationswinkel sowie die Entfernungen sämtlicher Pixel im Radarkorrdinatensystem berechnet. Anschließend wurden die Pixel maskiert, welche außerhalb der Radarkeule liegen. Die Antennenkeule wurde hier mit einem Azimuth- und Elevationsöffnungswinkel von $\pm 70^\circ$ und $\pm 10^\circ$, die maximale Reichweite mit 25 m parametriert. Diese Parameter ergeben sich aus der Konfiguration des hier verwendeten Radarsensors. Ein Beispiel dieser Maske $\mathbf{P}_{\text{FoV}}$ ist in Abbildung 7.3b zu sehen, wobei die dunkel gefärbten Pixel den maskierten Pixeln entsprechen.

3. **Keine Ausreißer bei DBSCAN-Methode**: Bei den Instanzenmasken im Kamerabild kommt es häufig vor, dass die Pixel der Tiefenschätzung Werte aus dahinter liegenden Objekten beinhalten, bspw. die Instanzenmaske eines Fußgängers beinhaltet fälschlicherweise ein paar Pixel des Gehwegs. Werden diese Pixel in die RD-map projiziert, führt das zu einer Überschätzung der geometrischen Ausdehnung. Die fehlerhaft segmentierten Pixel weisen möglicherweise auch eine falsche Doppler-Schätzung auf, so dass diese Pixel an die falsche Doppler-Position im RD-Gitter projiziert werden. Die Pixel der Instanzenmaske werden zusätzlich mit der Tiefenschätzung durch den DBSCAN-Algorithmus segmentiert (engl.: „geclustert“). Die Cluster mit der nicht maximalen Größe werden als Ausreißer behandelt und aus der Instanzenmaske entfernt. Ein Beispiel dieser Maske $\mathbf{P}_{\text{DBSCAN}}$ ist in Abbildung 7.3c zu sehen, wobei die dunkel gefärbten Pixel den maskierten Pixeln entsprechen.



(a) Ego Maske (b) FoV Maske (c) DBSCAN Maske (d) Gesamtmaske

Abbildung 7.3.: **Masken zur Selektion der Pixel:** Beispiele der Masken (gelb dargestellt) zur Selektion der Pixelmenge der Optimierungsläufe. Spaltenweise: Die Pixelmaske zur Maskierung des Ego-Fahrzeuges, Pixel außerhalb des Radar FoVs, DBSCAN Ausreißer und die kombinierte Maske. Zeilenweise: Beispiele für beide Kameras.

Durch die Reduktion der Pixelmenge werden Pixel aus den Gesamtkosten entfernt, welche den Trainingsprozess degradieren würden, da sie fehlerhafte Zielwerte aufweisen. Ein Beispiel der sich ergebenen Gesamtmaske ist in Abbildung 7.3d zu sehen, wobei die dunkel gefärbten Pixel den maskierten Pixeln entsprechen.

Wir führen die drei Masken zu einer Schnittmenge zusammen

$$\mathcal{P}_{\text{all}} = \mathcal{P}_{\text{Ego}} \cap \mathcal{P}_{\text{FoV}} \cap \mathcal{P}_{\text{DBSCAN}}, \tag{7.6}$$

als Menge aller Pixel, die bei der Optimierung berücksichtigt werden.

7.2.6.2. Pixelselektion anhand von SNR

Wir unterteilen die Pixelmenge in der RD-map in die zwei Klassen: Hintergrundrauschen und Signalreflexion. Die erste Klasse soll alle Pixel beinhalten, bei denen der Rauschanteil überwiegt. Bei der zweiten Klasse hingegen sollen Pixel beschrieben werden, bei denen eine Reflexion aus der Umgebung wahrscheinlich ist bzw. bei denen der Signalanteil überwiegt. Für deren Identifikation wird zunächst die Häufigkeitsverteilung des SNRs aller Pixel im Trainingsdatensatz dargestellt, siehe rot gezeichnete Linie in Abbildung 7.4.

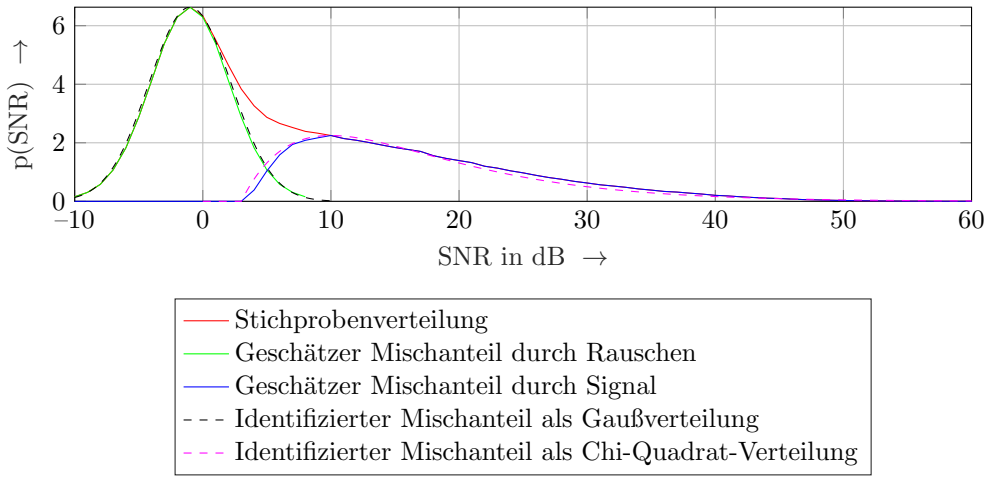


Abbildung 7.4.: **SNR Häufigkeitsverteilung:** Die Probenverteilung des Trainingsdatensatzes ist rot gezeichnet dargestellt. Die identifizierten Mischanteile des Rauschens und der Signalreflexion sind als grün und blau gezeichnete Polygonzüge dargestellt. Zusätzlich wurden die Mischanteile als Gauß Wahrscheinlichkeitsdichtefunktion (WDF) (schwarz gestrichelt) und Chi-Quadrat (pink gestrichelt) approximiert. Nach [EB6].

Analog zur oben beschriebenen Klassifizierung wird angenommen, dass es sich bei der dargestellten Häufigkeitsverteilung um eine Mischverteilung aus Gauß- und Chi-Quadrat-Verteilung handelt. Motiviert wurde die Annahme damit, dass sich Rauscheffekte gemäß dem zentralen Grenzwertsatz über Gaußverteilungen [Dur19] der Form

$$P(x, \mu, \sigma) = \frac{1}{\sqrt{(2\pi\sigma^2)}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\} \tag{7.7}$$

äußern. Radarreflexionen mit dominierender Reflexionsquelle lassen sich z.B. über ein Swerling-Typ-3-Modell beschreiben [Mes06], welches eine Chi-Quadrat-Verteilung (vier Freiheitsgrade) der Form aufweist

$$P(x, \sigma_{\text{avg.}}, \mu) = \frac{4(x-\mu)}{(\sigma_{\text{avg.}}-\mu)^2} \exp\left\{\frac{-2(x-\mu)}{(\sigma_{\text{avg.}}-\mu)}\right\}. \tag{7.8}$$

Zur Identifikation der Mischanteile wurde zunächst die Probenverteilung des Rauschens identifiziert. Dazu wurde der Verlauf der gesamten Häufigkeitsverteilung linksseitig des globalen Maximums nach rechts gespiegelt. Dabei wurde angenommen, dass das globale Maximum dem Mittelwert der Gaußverteilung entspricht. Der resultierende Verlauf ist in Abbildung 7.4 als grün gezeichneter Polygonzug dargestellt. Die Probenverteilung der Reflexionssignale wurde identifiziert, indem die zuvor identifizierte Probenverteilung des Rauschens von der gesamten Probenverteilung subtrahiert wurde. Die Probenverteilung der Signalreflexionen ist in Abbildung 7.4 als blau gezeichneter Polygonzug dargestellt.

Zu erkennen ist, dass sich die grüne und die schwarz gestrichelte bzw. die blaue und die pink gestrichelte Linien nahezu kongruent sind. Es ist also eine hohe Übereinstimmung der eingepassten Verteilungsfunktionen mit den Mischanteilen und kein offensichtlicher

Einwand gegen die oben gemachte Klassifikation zu beobachten.

Damit Rauschpixel bzw. Pixel aus dem linken Mischanteil den Trainingsprozess der NNen nicht stören, werden diese aus der Trainingsmenge entfernt. Die Wahrscheinlichkeit, entsprechende Pixel zu ziehen, beträgt bei 10dB_{SNR} bereits weniger als 0.1%. Oberhalb von 10dB_{SNR} werden maßgeblich Pixel aus dem rechten Mischanteil beobachtet. Damit definieren wir die weitere Pixelmenge

$$\mathcal{P}_{\text{train}} \coloneqq \{\mathbf{p} \mid \mathbf{p} \in \mathcal{P}_{\text{radar}} \wedge \mathbf{RD}_C(\mathbf{p}) > 10\text{dB}_{\text{SNR}}\}. \tag{7.9}$$

Einhergehend wird untersucht, ob sich diese Selektion vorteilhaft auf die Qualität der Winkelschätzung auswirkt. Dazu werden NNen nach den in Tabelle 7.2 gezeigten Parametrierungen trainiert. Es werden also zwei Netwerkarchitekturen (1×1 und 3×3) jeweils mit oder ohne diese Pixelselektion getestet.

Tabelle 7.2.: **NN Architekturen und Konfigurationen mit Pixelselektion nach Leistung**.

Name	Faltungskern (t)	SNR Selektion
NN_0	1×1	-
NN_1	1×1	$\mathbf{RD}_C > 10\text{dB}$
NN_4	3×3	-
NN_5	3×3	$\mathbf{RD}_C > 10\text{dB}$

7.2.7. Gesamtkosten

Die Gesamtkosten ergeben sich als Mittelwert der Abweichungen über die Pixel zu

$$l_{\text{DOA, all}} = \frac{1}{|\mathcal{P}_{\text{all}} \cap \mathcal{P}_{\text{train}}|} \sum_{\mathbf{p} \in \mathcal{P}_{\text{all}} \cap \mathcal{P}_{\text{train}}} l_{\text{DOA}}(\mathbf{p}). \tag{7.10}$$

Wie durch Tabelle 7.2 beschrieben, findet die Selektion der Pixelmenge anhand von SNR nicht für alle Konfigurationen statt, so dass die Menge $\mathcal{P}_{\text{train}}$ in den Kosten teilweise entfällt.

7.2.8. Initialisierung der Parameter

Die Initialisierung der Netzwerkparameter erfolgte für Schichten mit Sigmoid oder Tangens-hyperbolicus-Aktivierungsfunktionen gemäß der Xavier-Methode [GB10]. Für Schichten mit ReLU-Aktivierungsfunktion gemäß der Kaiming-Methode [HZRS15].

7.2.9. Optimierer

Zur Optimierung der Netzwerkparameter wurde ADAM [KB15] mit einer initialen Schrittweite von 10^{-4} und „early stopping“ verwendet.

7.2.10. Trainingsprozess

In jedem Trainingsschritt wird aus dem Trainingsdatensatz ein beliebiges Beispiel gezogen. Die Ziehung von Frame und Kameraindex erfolgt aus Gleichverteilung. Dieses Beispiel wird dem NN zugeführt, Inferenz ausgeführt, die Abweichung vom Zielwert ermittelt und die Netzwerkparameter aktualisiert. Ein Training wurde jedoch nur durchgeführt, wenn das Ego-Fahrzeug in Bewegung ($v_x = [0, 30]\mathrm{m\,s^{-1}}$) war und stationäre Ziele somit in Dopplerdimension besser voneinander getrennt sind. Der Verlauf der Kosten im Trainingsprozess ist in Abbildung 7.5 dargestellt.

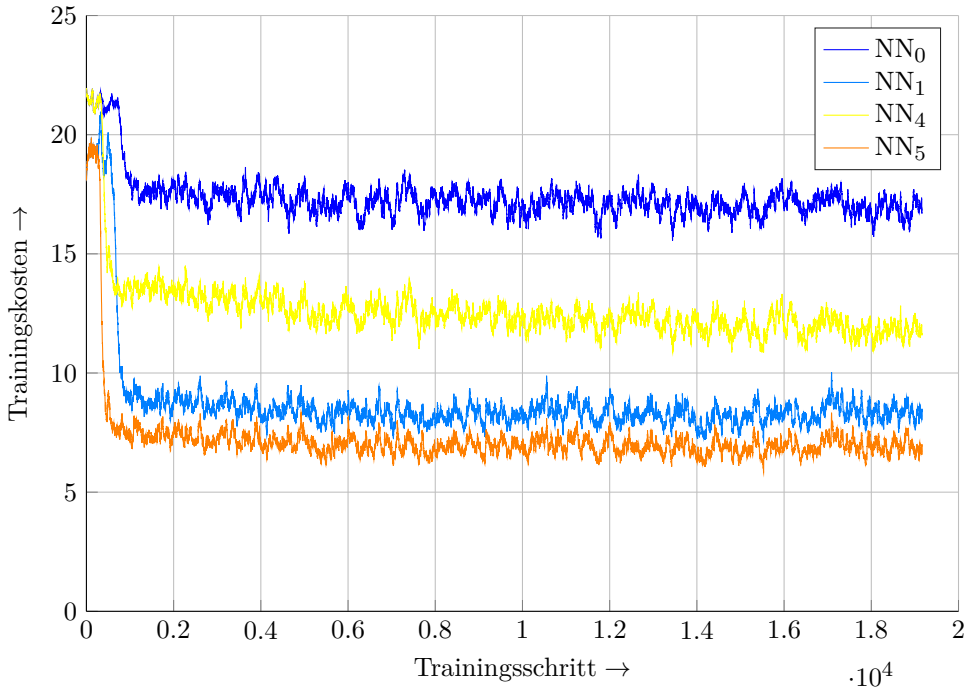


Abbildung 7.5.: **Verlauf des Trainingsprozesses der Winkelnetzwerke:** Die Kosten während des Trainings konvergieren bereits nach einigen Trainingsbeispielen.

Die Winkelnetzwerke wurden für eine Epoche trainiert. Es ist zu beobachten, dass sämtliche NNe nach etwa 1000 Trainingsschritten den größten Teil der Optimierung erfahren haben und augenscheinliche Konvergenz eingetreten ist. Dies entspricht etwa einem Zwanzigstel der gesamten Trainingsmenge. Ursächlich für die schnelle Konvergenz kann die Wahl der Schrittweite, aber auch die Effizienz der Annotation sein. Durch eine noch kleinere Schrittweite des Optimierens könnte die Konvergenz verzögert und gegebenenfalls eine bessere Inferenz ermöglicht werden. Als positives Merkmal der vorgestellten automatischen Annotation wird deren Effizienz angesehen. Pro Frame werden etwa $3 \cdot 10^5$ Kamera-Pixel-Trainingsbeispiele bereitgestellt. Die 1000 Trainingsbeispiele entsprechen bei kontinuierlicher Aufnahme mit 10 FPS einer Länge von weniger als 2 min Aufnahme und bilden dementsprechend etwa $3 \cdot 10^8$ annotierten Eingangs-Ausgangsverknüpfungen des NN dar.

Nach den in Tabelle 7.2 aufgeführten Parametern wurden unterschiedliche Kosten-

funktionen für die NNen festgelegt, wodurch sich die unterschiedlichen Konvergenzwerte in Abbildung 7.5 begründen lassen. Ein Vergleich der NN Schätzgenauigkeit kann basierend auf diesen Trainingskosten nicht vorgenommen werden. Dieser Vergleich wird im folgenden Abschnitt bei identischer Metrik durchgeführt.

7.2.11. Ergebnis

7.2.11.1. Referenzverfahren

Als Referenzverfahren für die Bewertung der Winkelschätzgenauigkeit aus dem NN wird der PM Winkelschätzer (siehe Unterunterabschnitt 2.1.5.1) sowie der Bartlett BF (siehe Unterunterabschnitt 2.1.5.2) aus der klassischen Radar-Signalverarbeitung verwendet.

7.2.11.2. Beschreibung der Metriken

In Abs. 2.1.5 wurden aus dem Modell einer einzelnen reflektierten EM-Welle und der daraus resultierenden Phasendifferenz $\Delta\Phi$ verschiedene Winkelschätzverfahren abgeleitet. Dies ist stark vereinfacht und in der Praxis strahlen meist EM-Wellen aus unterschiedlichen Winkelbereichen in den Radarsensor ein. Eine Trennung der Wellen/Reflektoren entsprechend des Einfallswinkels ist nicht immer möglich. Sowohl das hier vorgestellte NN zur Winkelschätzung als auch der PM-Winkelschätzer liefern einen einzigen Winkelschätzwert pro Zelle im RD-Gitter (sog. „Punktschätzer“). Wird dieser Schätzwert in das Kamerabild projiziert, so wird der einzelne Schätzwert, wie in Abschnitt 6.5 beschrieben, mehreren Kamerapixeln in gleicher Weise zugeordnet. Daraus ergibt sich, dass die projizierte Winkelschätzung gegen ein Ensemble an Zielwerten der zugehörigen Kamerapixel verglichen werden muss. Die Streuung der Zielwerte kann gering ausfallen, wenn es sich um Nachbarpixel im Kamerabild handelt. Sind die Pixel im Kamerabild jedoch weiter gestreut, so können die Zielwerte über einige Winkelgrad streuen. Als Beispiel beider Fälle sei auf Abbildung 6.7 verwiesen. Der erste Fall wurde beim links parkenden Fahrzeug durch die rot markierte Zone hervorgehoben. Der zweite Fall wird durch die weiteren, rot markierten, zur XZ Ebene (im Ego-KOOS) spiegelsymmetrischen Zonen markiert.

Für den Fall, dass einer RD-Gitter-Zelle mehrere Kamerapixel zugeordnet werden, kann basierend auf den Sensordaten nicht festgestellt werden, welches der Kamerapixel bzw. welche Kombination zum Radarsignal geführt hat. Zur Veranschaulichung dieses Dilemmas ist in Abbildung 7.6 ein fiktives Beispiel dargestellt. Im linken Drittel der Abbildung sind zehn komplexe Vektoren (schwarz dargestellt) zu erkennen, welche die sich komplex überlagernden Wellen unterschiedlicher Reflexionspunkte darstellen sollen. Die Winkel der Vektoren gegenüber den Achskoordinaten sollen die einzelnen Phasendifferenzen $\Delta\Phi$ der Wellenanteile beschreiben. Die Länge der Vektoren die Amplitude der Wellenanteile. Durch die komplexe Summation der Vektoren ergibt sich ein resultierender Vektor (rot dargestellt) mit entsprechend resultierenden Phasendifferenz (grün eingezeichnet), welcher die Messung in der RD-Gitter-Zelle darstellt. Die Länge der schwarzen Vektoren bestimmt somit die Wichtung der einzelnen Vektorwinkel auf den Gesamtphasenwinkel. Aus den Daten der Referenzsensorik kann nun zwar der Einfallswinkel der Reflexionspunkte, nicht jedoch deren Amplitude gemessen werden. Im ersten Bild wurde naiv angenommen, dass alle Reflexionspunkte die gleiche Wellenamplitude aufweisen, und der resultierende Summenvektor eingezeichnet.

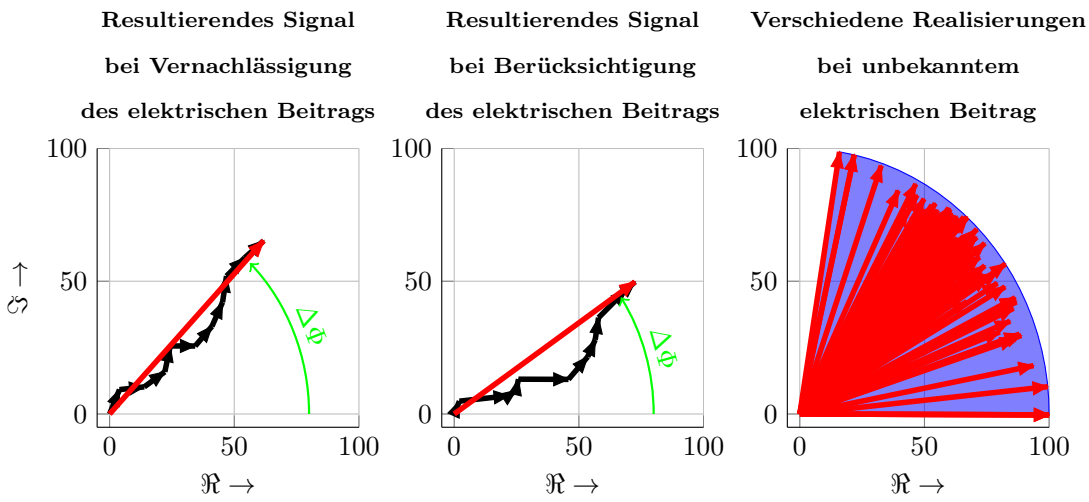


Abbildung 7.6.: **Winkelsynthese bei mehreren Reflektoren:** Links: Zehn komplexe Einzelvektoren (schwarz) gleicher Länge und unterschiedlichem Winkel gegenüber den Koordinatenachsen wurden zu einem resultierenden Vektor (rot) addiert. Die vom Radar gemessene Phasendifferenz $\Delta\Phi$ ist in grün eingezeichnet. Mitte: Die Länge der Einzelvektoren wurde einmalig aus einer Gleichverteilung gezogen bestimmt. Die Winkel der Vektoren entsprechen denen der vorigen Vektoren. Der resultierende Vektor variiert im Vergleich zum ersten Beispiel und somit auch die Phasendifferenz. Rechts: Die Länge der Einzelvektoren wurde mehrmals gezogen und jeweils die resultierenden Vektoren eingetragen. Das Ensemble der resultierenden Vektoren erstreckt sich über einen weiten Winkelbereich, dargestellt über den blauen Kreissektor. Nicht eingezeichnet ist die entsprechende Streuung der Phasendifferenzen.

Um die unbekannten elektromagnetischen Reflexionseigenschaften beispielhaft zu simulieren, wurde die Länge der Vektoren aus einer Verteilungsfunktion (Gleichverteilung) gezogen (mittleres Drittel der Abbildung). Die Wichtung der einzelnen Vektoren fällt damit unterschiedlich aus, und der resultierende Vektor hat einen leicht unterschiedlichen Zielwinkel gegenüber der naiven Variante. Da die Länge der Vektoren unbekannt ist – die elektromagnetischen Reflexionseigenschaften sind aus den Daten der Referenzsensorik ja nicht bekannt –, kann der resultierende Vektor natürlich eine Vielzahl an möglichen Richtungen annehmen. Im letzten Drittel der Abbildung sind deshalb eine Vielzahl an möglichen resultierenden Vektoren dargestellt. Der kleinste Winkel des resultierenden Vektors ergibt sich aus dem kleinsten Winkel aller Einzelvektoren, nämlich dann, wenn die Vektorlänge aller anderen Einzelvektoren gerade Null ist. Der größte Winkel umgekehrt aus dem größten Winkel eines Einzelvektors.

Für die Evaluation bedeutet das, dass, wenn ein Winkelschätzer einen Einfallswinkel schätzt, der im Intervall zwischen kleinstem und größtem Winkelwert der assoziierten Kamerapixel liegt, der Schätzwert ohne weitere Kenntnis der elektromagnetischen Reflexionseigenschaften damit durchaus valide ist und in einer Evaluationsmetrik nicht als möglicher Fehler angezeigt werden darf. Zum Vergleich: Bei dem Training des NN wurde aufgrund der zuvor genannten technischen Limitierung keine Gewichtung der

Einfallsrichtungen in Gl. 7.10 vorgenommen und somit die naive Variante nachgebaut.

Um die Evaluationsmetrik mathematisch auszudrücken, definieren wir vorab die Menge aller Kamerapixel, welche auf das gleiche Pixel $\mathbf{p}_{\mathrm{RD}}$ im RD-Gitter zeigen, als

$$\mathcal{P}_{\mathrm{Cam}}(\mathbf{p}_{\mathrm{RD}}) \coloneqq \left\{ \mathbf{p} \middle| \mathbf{p} \in \mathcal{P}_{\mathrm{radar}} \wedge \mathbf{p}_r(\mathbf{p}) = \mathbf{p}_{\mathrm{RD}} \right\}. \tag{7.11}$$

Zur Erinnerung: Hier ist $\mathcal{P}_{\mathrm{Cam}}$ die Menge aller Kamerapixel im Radar FoV, siehe Gleichung 5.3. $\mathbf{p}_r(\mathbf{p}) = \mathbf{p}_{\mathrm{RD}}$ schneidet diese Menge, indem nur Kamerapixel selektiert werden, welche auf das Radarpixel $\mathbf{p}_{\mathrm{RD}}$ zeigen.

Nun definieren wir die Grenzen des Winkelintervalls zu

$$\phi_{\mathrm{label,min}}(\mathbf{p}_{\mathrm{RD}}) = \min_{\forall p \in \mathcal{P}_{\mathrm{Cam}}(\mathbf{p}_{\mathrm{RD}})} \phi_{\mathrm{label}}(\mathbf{p}) \tag{7.12}$$

und

$$\phi_{\mathrm{label,max}}(\mathbf{p}_{\mathrm{RD}}) = \max_{\forall \mathbf{p} \in \mathcal{P}_{\mathrm{Cam}}(\mathbf{p}_{\mathrm{RD}})} \phi_{\mathrm{label}}(\mathbf{p}). \tag{7.13}$$

Liegt die Prädiktion nun außerhalb des möglichen Winkelintervalls, so wird die Abweichung zu den Intervallgrenzen betragsmäßig festgehalten:

$$q_{\mathrm{DoA}}(\mathbf{p}_{\mathrm{RD}}) = \begin{cases} |\phi_{\mathrm{predict}}(\mathbf{p}_{\mathrm{RD}}) - \phi_{\mathrm{label,min}}(\mathbf{p}_{\mathrm{RD}})| & \phi_{\mathrm{predict}}(\mathbf{p}_{\mathrm{RD}}) < \phi_{\mathrm{label,min}}(\mathbf{p}_{\mathrm{RD}}) \\ |\phi_{\mathrm{predict}}(\mathbf{p}_{\mathrm{RD}}) - \phi_{\mathrm{label,max}}(\mathbf{p}_{\mathrm{RD}})| & \phi_{\mathrm{predict}}(\mathbf{p}_{\mathrm{RD}}) > \phi_{\mathrm{label,max}}(\mathbf{p}_{\mathrm{RD}}) \\ 0 & \text{sonst} \end{cases} \tag{7.14}$$

und für alle Pixel $\mathbf{p}_{\mathrm{RD}} \in \mathcal{P}_{\mathrm{RD}}$ des RD-maps gemittelt zu

$$Q_{\mathrm{DoA}} = \frac{\sum_{\mathbf{p}_{\mathrm{RD}} \in \mathcal{P}_{\mathrm{RD}}} q_{\mathrm{DoA}}(\mathbf{p}_{\mathrm{RD}})}{|\mathbf{P}_{\mathrm{RD}}|}. \tag{7.15}$$

Die in Gleichung 7.15 dargestellte Metrik kann also als MAE mit optimaler[2] Assoziation von Kamera- und Radarpixel interpretiert werden.

Es sei zusätzlich bemerkt, dass hier nur ein einfacher Winkelschätzer trainiert wurde. Für eine mögliche Anwendung mit Winkeltrennung über ein zu schätzendes Azimutspektrum kann die Metrik analog angewendet werden. Allerdings bleibt auch dabei ein unterschiedliches Auflösungsvermögen von Referenzsensorsytsem und Radar bestehen, so dass weiterhin keine bijektive Abbildung entsteht. Jedoch verbessert sich durch Hinzunahme einer weiteren Dimension die Eindeutigkeit, so dass die Menge $\mathcal{P}_{\mathrm{Cam}}(\mathbf{p}_{\mathrm{RD}})$ kleiner ausfallen würde.

7.2.11.3. Quantitative Auswertung

Zur Veranschaulichung der Schätzqualität wurden die Metrik nach Gleichung 7.15 für alle Winkelschätzer auf den Daten des Testdatensatzes ermittelt. Da die Schätzqualität in der Regel stark vom SNR abhängig ist, wurden die Metriken über das SNR aufgetragen. Die erreichten Werte der Winkelschätzer sind in Abbildung 7.7 dargestellt.

[2] Optimal in der Hinsicht der minimalen Winkelabweichung.

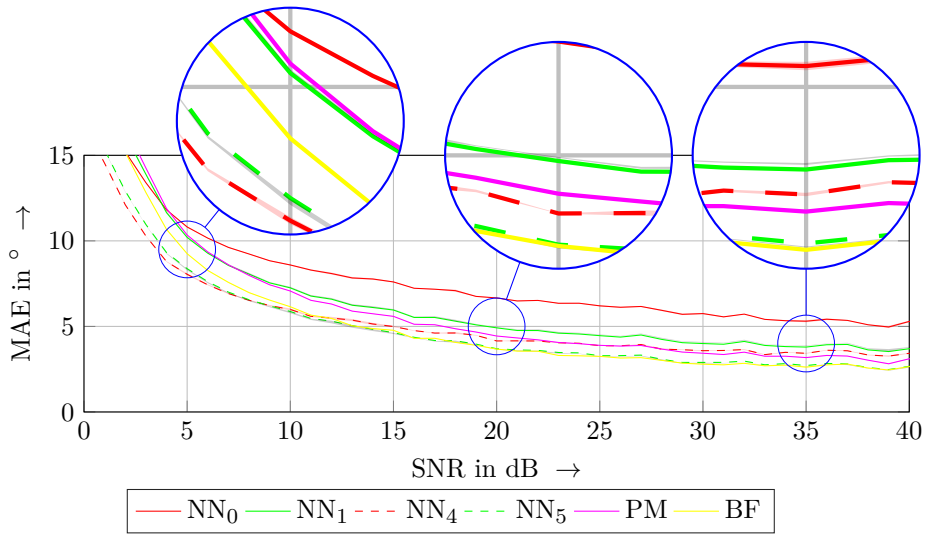


Abbildung 7.7.: **Qualitätsmaße der Winkelschätzer über SNR:** Die mittlere absolute Abweichung der Winkelschätzer über SNR. Vergrößert dargestellt sind jeweils ein Bereich niedrigen SNRs, mittleren SNRs und hohen SNRs.

In Abbildung 7.7 ist zu erkennen, dass der MAE bei allen Schätzern mit steigendem SNR fällt. Dieses Verhalten ist plausibel, da bei steigendem SNR der Rauschanteil gegenüber dem Signalanteil weniger stark ausgeprägt ist und entsprechend die Schätzung weniger beeinflusst.

Da es bei der Menge an Winkelschätzern visuell schwierig ist, den Unterschied zwischen den NNen nach Tabelle 7.2 ausfindig zu machen, wurden die Schätzer in Tabelle 7.3 an drei SNR-Punkten entsprechend der erreichten MAE hierarchisch eingeordnet. Diese gewählten drei SNR Punkte liegen äquidistant zueinander und repräsentieren niedriges, mittleres und hohes SNR.

Tabelle 7.3.: **Hierarchische Einordnung der Schätzer entsprechend erreichter MAE.**

	SNR 5 dB	SNR 20 dB	SNR 35 dB
MAE ↑	NN_0	NN_0	NN_0
	PM	NN_1	NN_1
	NN_1	PM	NN_4
	BF	NN_4	PM
	NN_5	NN_5	NN_5
	NN_4	BF	BF

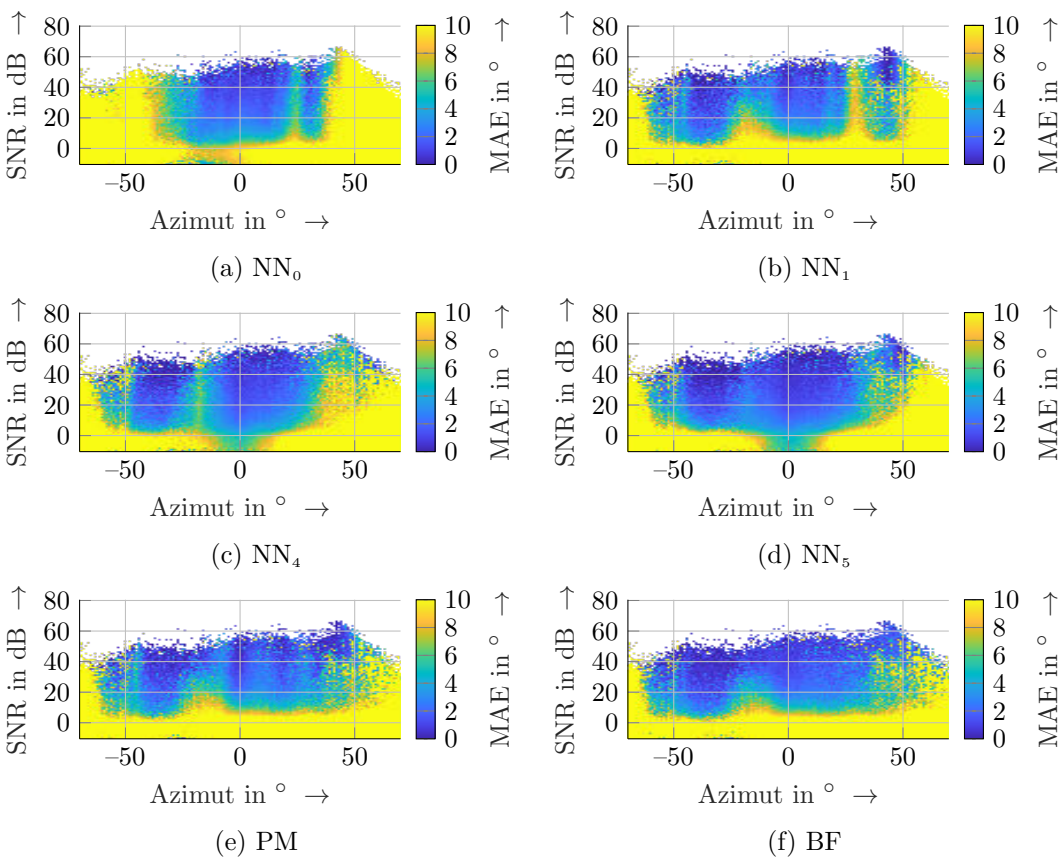
Aus der Tabelle lässt sich nun identifizieren, welche Parameter der NN-Konfiguration einen Einfluss auf die Schätzgenauigkeit gehabt haben. Beginnend beim ersten Parameter, der SNR-Selektion nach Unterunterabschnitt 7.2.6.2, welche bei den ungeraden NNen aktiv war, muss also das MAE der NN-Paare NN_0 vs. NN_1 und NN_4 vs. NN_5 verglichen werden. Bei SNR 20 dB und SNR 35 dB haben alle NNen mit aktivierter SNR-basierter

Pixelselektion ein verbessertes MAE erreicht als ihre Partner. Bei SNR 5 dB ist dieses Bild genau umgedreht, was daran zu begründen ist, dass die Pixelselektion keine Beispiele unterhalb von 10 dB SNR für das Training bereitgestellt hat. Da aber auch hier die NNen mit aktivierter Pixelselektion ähnliche MAE Werte erreicht haben, deutet dies auf eine gute Generalisierung der Schätzer hin.

Nun werden die Unterschiede zwischen NNen mit 1×1 und 3×3 Filterkernen verglichen. Gemäß Tabelle 7.2 sind dazu die NN-Paare NN_0 vs. NN_4 und NN_1 vs. NN_5 zu vergleichen. Aus Tabelle 7.3 ist ersichtlich, dass in allen SNR-Bereichen die NNen mit 3×3 Filterkernen eine geringere MAE erreichen als die analogen 1×1 NNen. Daraus wird gedeutet, dass das Einbeziehen der Pixelnachbarschaft im RD-Gitter einen positiven Einfluss auf die Winkelschätzung hat.

Im Vergleich zu den klassischen Winkelschätzern erreichen die besten NNen, NN_5 und NN_4, in Abbildung 7.7 bis etwa 10 dB SNR ein geringeres MAE, danach ein ähnliches MAE. Es ist festzuhalten, dass es sich beim BF um ein Verfahren mit Winkeltrennfähigkeit handelt, wohingegen die NNen nur auf Punktschätzung des Winkels trainiert wurden. Beim klassischen PM handelt es sich um einen vergleichbaren Punktschätzer, welche allerdings in allen SNR-Regionen dem BF und den beiden NNen unterlegen ist.

Um eventuelle Winkelinhomogenitäten der Winkelschätzer aufzudecken, wurden in Abbildung 7.8 die Metriken aller Schätzer gegenüber SNR und Einfallswinkel dargestellt.



(a) NN_0 (b) NN_1

(c) NN_4 (d) NN_5

(e) PM (f) BF

Abbildung 7.8.: **2D MAE Histogramme:** Darstellung der Abhängigkeit von MAE gegenüber von SNR und Einfallswinkel. Nach [EB6].

Gut zu erkennen sind nun die Winkelbereiche, bei denen die Parameter der NN einen merklichen Unterschied verursacht haben. So kann bei allen NNen mit aktivierter Pixelselektion eine erhebliche Reduktion des MAEs an den äußeren Winkelbereichen beobachtet werden. Man vergleiche dazu Abb. 7.8a vs. 7.8b und 7.8c vs. 7.8d.

Ebenfalls kann beobachtet werden, dass durch die Verwendung von 3×3 anstelle von 1×1 Filterkernen der CNNs eine weitere Verringerung der Winkelinhomogenität erreicht werden konnte. Man vergleiche dazu Abb. 7.8a vs. 7.8c und 7.8b vs. 7.8d.

Im Vergleich zu den klassischen Verfahren (PM und BF) erreichen die besten NNen (NN_5 und NN_4) im Allgemeinen ein geringeres MAE im Winkelintervall $[-40°, 40°]$. In den Randbereichen des FoV erreichen die klassischen Winkelschätzer verringerte MAE. Da es sich bei den NNen um überwachte Lernverfahren handelt, wäre es denkbar, dass die NNen gelernt haben, diese Randbereiche aufgrund geringerer Auftretenswahrscheinlichkeiten zu vernachlässigen. Weitere Anpassungen der Kostenfunktionen, um diese Winkelinhomogenität der NN-basierten Schätzer auszugleichen, sind für die Zukunft vorstellbar, werden im Zuge dieser Arbeit aber nicht weiter verfolgt.

7.2.11.4. Qualitative Auswertung

Um einen subjektiven Eindruck der Winkelschätzung zu bekommen, sind in Abbildung 7.9 die Winkelschätzung von NN_5 und PM dargestellt.

In der linken Spalte sind die Zielwerte des Winkels farblich kodiert über dem Kamerabild dargestellt. Die Farbkodierung des Winkels ist in der Farblegende am unteren Rand der Abbildung zu entnehmen. In zweiter und dritter Spalte sind die Prädiktionen von NN und PM basierter Schätzung ins Kamerabild projiziert. Die Farbhelligkeit wurde entsprechend der Signalleistung in der RD-map skaliert, so dass Rauschbereiche unterdrückt und Signalbereiche hervorgehoben werden. In vierter und fünfter Spalte sind die Prädiktionen analog farblich codiert im RD-map eingetragen. In den letzten beiden Spalten sind die Winkelschätzungen RD-map gegeben. Dabei wurde eine andere Farbkodierung gewählt und die Helligkeit nicht entsprechend des SNRs skaliert.

Zu sehen ist, dass die Winkelschätzer die Zielwerte aus der Referenzsensorik meist gut repräsentieren. Auffällig ist, dass bei der NN basierten Schätzung die Winkel von benachbarten Zellen in der RD-map (rechten Spalten in Abbildung 7.9) erheblich glattere Verläufe aufweisen als bei der PM basierten Schätzung. Daraus wird geschlossen, dass das NN statistische Zusammenhänge zwischen den Winkeln benachbarten Zellen im RD-map gelernt hat.

7.2.11.5. Laufzeitanalyse

Bisher wurden die Winkelschätzer nur anhand ihrer Schätzgenauigkeit verglichen. Wir werden den Vergleich nun mit einer Analyse der Laufzeitkomplexitäten[3] abschließen. Dazu wird die Anzahl der Fließkommaoperationen ermittelt.

Für den PM-Schätzer ergibt sich die Anzahl Fließkommaoperationen aus den einzelnen Termen in Gleichung 2.47. Mit der Anzahl der Antennenpaaren M ergibt sich so eine Operationszahl der Einzelterme wie in Tab 7.4 gelisteten.

[3] Hierbei wird in der Informatik die Anzahl der Rechenschritte zur Lösung eines Problems verstanden [Wik23]

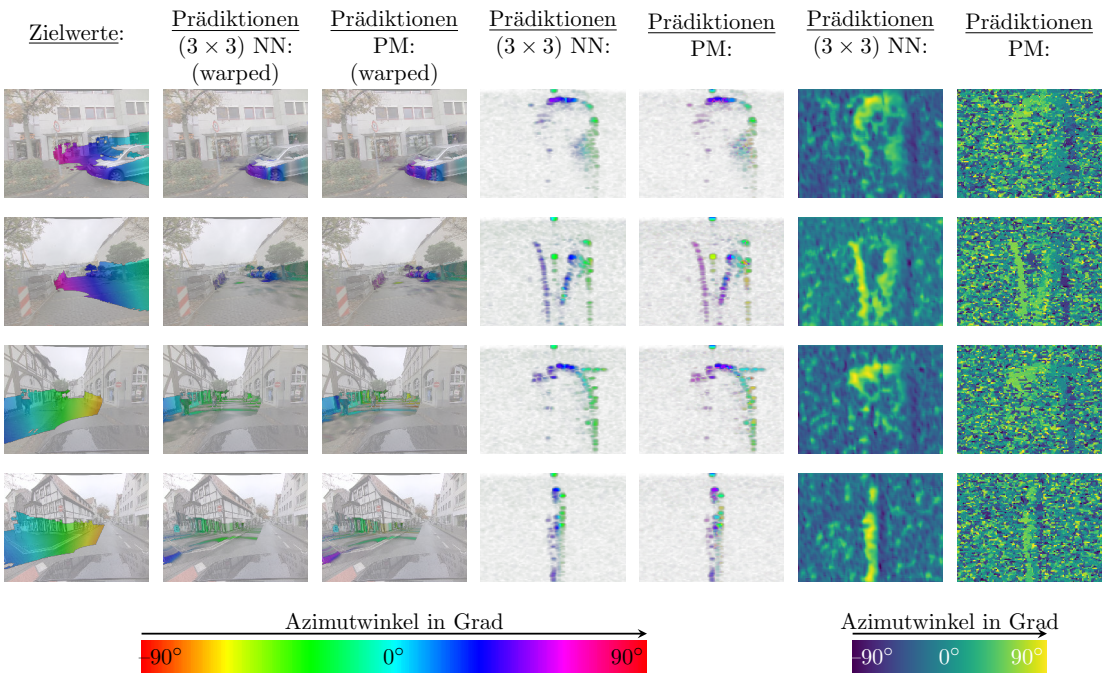


Abbildung 7.9.: **Qualitative Ergebnisse der DoA-Schätzung auf Testdaten**. Von links nach rechts: Referenz-Azimutwinkel, prädizierter Azimutwinkel (Helligkeit entspricht der vom Radar empfangenen Leistung) ind Kamerabild und RD-map. Prädizierter Winkel in Graustufen (Farbe entspricht dem Azimutwinkel). In den Kamerabildern wurden nur Pixel im Radar-FoV visualisiert. Während des Netzwerktrainings werden die Pixelwerte der gewarpten Prädiktion mit denen der Referenz verglichen und die Prädiktion entsprechend trainiert. Nach [EB6].

Tabelle 7.4.: **Anzahl der Operation für PM Verfahren:**

Mathematische Operation	Anzahl der Operationen
A	$3M$
B	$4M$
$B^+ = B^T B$	$M + (M-1)$
$A^+ = B^T A$	$M + (M-1)$
$()^{-1}$	1
$B^+ A^+$	1
arcsin()	1

Mit der Anzahl der Pixel in der RD-map N ergibt sich die Summe der Operationen zu $N * ((3M) + (4M) + (M + (M-1)) + (M + (M-1)) + 1 + 1 + 1)$ bzw. hier zu $128 * 96 * ((3 * 2) + (4 * 2) + (2 + (2-1)) + (2 + (2-1)) + 1 + 1 + 1) = 282624$.

Für den Beamformer ergibt sich die Anzahl der Operationen aus der Anzahl der Pixel in der RD-map und der Anzahl der Operationen der Fast-Fourier-Transformation (FFT)-Engine. Wir verwenden hier beispielhaft die „split-radix“ Methode [JF07], dessen Operationszahl mit $4L\log_2(L) - 6L + 8$ skaliert. Bei einer gewählten FFT-Länge von

$L = 64$ ergibt sich die Anzahl der Operationen zu $N * 1160 = 128 * 96 * 1160 = 14254080$.

Die Fließkommaoperationen in den NNen wurden mit Hilfe einer automatischen Operationszählung [eLa21] ermittelt und zur besseren Übersicht in Tabelle 7.5 dargestellt.

Tabelle 7.5.: **Anzahl der Operation für DoA Verfahren:**

Algorithmus	Anzahl der Operationen	geschätzte Laufzeit bei 300 MHz
PM	282624	0.94 ms
BF	14254080	47.5 ms
NN	4785438700	16 s

Bei Verwendung einer CPU mit einer Taktfrequenz von 300 MHz wurden grob geschätzte Laufzeiten ermittelt (Anzahl der Operationen / Taktfrequenz) und in Tabelle 7.5 eingetragen. Diese Laufzeitabschätzung vernachlässigt von der Implementierung und Architektur abhängige Details, wie z.B. zusätzlichen Operationsaufwand durch Speicheraufwand oder Parallelisierung. Unter Annahme, dass die geforderte Framerate des Radarsensors 20 FPS ist, würde sich das hier dargestellte NN aufgrund der deutlich zu langen Inferenzdauer disqualifizieren. Die verbesserte Genauigkeit der Winkelschätzung durch BF oder NN gegenüber PM wurde durch eine erheblich gesteigerte Rechenkomplexität erkauft. Für die Inferenz auf einem Radarsensor mit typischer Prozessorausstattung sollte die Architektur des NN deutlich vereinfacht werden. Ebenfalls würde sich eine NN-Implementierung mit Ganzzahl-Datentypen (Integer) im Vergleich zur Fließkommazahl-Datentypen (Float) anbieten. Dadurch würde sicherlich die Inferenz beschleunigt, aber insbesondere auch der Speicherbedarf reduziert.

Da der Datenbedarf für das Training eines NN im Allgemeinen mit der Größe des NN steigt, wurde absichtlich eine NN-Architektur gewählt, die dementsprechend schwieriger zu trainieren ist und die hohe Laufzeit des NN somit absichtlich in Kauf genommen. Da die hier verwendeten NNen während des Trainings schnell konvergiert sind, gibt es aus Sicht des Autors wenig Zweifel am Erfolg eines Trainingsprozesses kleinerer NNen mit ausreichend schneller Inferenz für typische Radarsensoren. Ein Laufzeitvorteil durch die Verwendung von NNen gegenüber klassischen Winkelschätzern bzw. Winkeltrennverfahren wurde bereits in [GFFW18, FWG19b, FWG19a] beschrieben und ist deshalb kein weiterer Bestandteil dieser Arbeit.

7.2.11.6. Weitere Analyse

Zusätzlich wurde in A.1 eine Untersuchung zur Reduktion des Einflusses von Labelnoise auf das NN-Training durchgeführt. Die darin vorgeschlagene Methode führte jedoch nicht zu gewünschtem Ergebnis und ist deshalb nicht in dem Hauptteil dieser Arbeit dokumentiert.

7.3. Zieldetektion

In diesem Abschnitt wird ein mögliches Vorgehen zum überwachten Training eines Zieldetektors mittels NN vorgestellt. Bei der Winkelschätzaufgabe kann über die 3D-Position der Pixel der Zielwert des Einfallswinkels und somit das Label bestimmt werden. Da Referenzsensorik und Radarsensor auf unterschiedlichen EM-Wellenlängen

operieren, unterscheidet sich deren Sensitivität gegenüber der Sensorumgebung. Für die Aufgabe der Zieldetektion kann deshalb kein nativer Zielwert aus den Daten der Referenzsensorik verwendet werden. In diesem Abschnitt werden stattdessen die Zielwerte für die Zieldetektionsaufgabe automatisch aus den Daten der Referenzsensorik geschätzt und anschließend ein NN zur Prädiktion dieser Zielwerte auf den Radardaten trainiert.

7.3.1. Eingangsdaten

Analog zur klassischen CFAR-Methode soll auch das NN exklusiv auf dem Leistungsspektrum bzw. RD-map arbeiten. Als einziger Eingangskanal des NNs wird dementsprechend die RD-map bereitgestellt. Wir definieren die Eingangsdaten des NNs als

$$\boldsymbol{\Gamma}_{\mathrm{TD}} = \mathbf{RD}. \tag{7.16}$$

7.3.2. Zielwerte

Für die Winkelschätzung wurde ausschließlich der Einfallswinkel der Reflexionen betrachtet, welcher sich aus der geometrischen Position eines Punktes in der Umgebung ergibt. Mit Kenntnis dieser Position konnte der Zielwert für den Winkelschätzer mit der nativen Messfähigkeit[4] der Referenzsensorik nachgestellt werden. Bei den Zielwerten für die Zieldetektion ist dies nicht so einfach möglich. Ursächlich dafür ist, dass sich EM-Reflexionseigenschaften bzw. die Streuung mit der Wellenlänge ganz wesentlich ändern können. Man vergleiche dazu die relativen Reflexionsamplituden nach Fresnel in Gleichung 2.4, in denen die relativen Reflexionsamplituden im Wesentlichen vom Aspektwinkel und den Brechungsindices der Materialien abhängen. Die Brechungsindices sind eine Funktion der Wellenlänge und können somit die Reflexionsantwort der Sensoren beeinflussen. Für die Aufgabe der Zieldetektion bedeutet das, dass der Radar (ca. 3.54 mm Wellenlänge) nicht automatisch alle Punkte sehen kann, welche z.B. der Lidar sieht (ca. 905 nm Wellenlänge). Folglich wurden mögliche Ziele aus den Daten der Referenzsensorik heuristisch geschätzt.

7.3.2.1. Einflussparameter der Reflexionsamplitude

In Unterabschnitt 2.1.1 wurde bereits beschrieben, wie sich relative Reflexionsamplituden von EM-Wellen nach physikalischer Optik an Grenzflächen verhalten. Für die Zieldetektion mittels NN muss nun festgelegt werden, (a) wo diese Grenzflächen in der Szene positioniert sind und (b) welche Eigenschaften die Grenzflächen aufweisen müssen, um für den Radar überhaupt wahrnehmbar zu sein.

In der Literatur wird häufig zwischen Spiegelreflektoren und diffusen Reflektoren unterschieden, siehe z.B. [Jud88]. Bei Spiegelreflektoren ergeben sich ausschließlich Transmission in das Material und Reflexion nach Fresnel, wobei der Ausfallswinkel der Welle dem Einfallswinkel gespiegelt an der Oberflächennormalen entspricht. Weisen die Grenzflächen geometrische Imperfektionen (raue Oberfläche) auf, wird das eintreffende Signalbündel in viele Richtungen gestreut. Man spricht von der diffusen Reflexion. Das Rayleigh-Kriterium [Woo06] beschreibt den Zusammenhang von Transmission und Reflexion an rauen Oberflächen in Abhängigkeit von der Wellenlänge. Es wird

[4] Der Lidar misst bereits die Position.

beschrieben, dass Material bei kürzerer Wellenlänge eher als diffus wahrgenommen wird. Verlängert sich die Wellenlänge, so verringert sich die Signalstreuung und es kommt eher zur spiegelnden Reflexion. Eine diffuse Reflexion sorgt dafür, dass ein Objekt nahezu unabhängig vom Betrachtungswinkel wahrgenommen werden kann. Demnach sorgt die deutlich kürzere Wellenlänge der Referenzsensorik (Lidar 905 nm) dafür, dass Grenzflächen eher diffus wahrgenommen werden und dementsprechend nahezu unabhängig vom Einfallswinkel wahrgenommen werden können, als es für den Radar mit der größeren Wellenlänge (ca. 3.54 mm) der Fall ist. Nur unter Berücksichtigung der Wellenlänge sollte die Referenzsensorik also in der Lage sein, alle Grenzflächen im Szenario wahrzunehmen, welche auch beim Radarsensor zu einer möglichen Signalreflexion zum Sensor führen würden.

Motiviert durch Gleichung 2.4 wird der Aspektwinkel als eine bedeutsame Größe der Reflexionsamplitude interpretiert. Zwar wird die Reflexionsamplitude auch durch verbleibende Terme in Gleichung 2.4 beeinflusst, welche die Materialeigenschaften der Medien beschreiben. Im Folgenden werden wir jedoch eine automatische Annotation der Zielwerte für die Zieldetektion ausschließlich anhand des Aspektwinkels untersuchen. Begründet wird diese Simplifikation damit, dass der Aspektwinkel sich relativ einfach aus den Daten der Referenzsensorik ermitteln lässt. Eine Schätzung der (frequenzabhängigen) Materialeigenschaften aus den Daten der Referenzsensorik ist vermutlich deutlich aufwendiger und wird daher für mögliche weiterführende Untersuchungen zurückgestellt. Es sei noch einmal erwähnt, dass der Aspektwinkel in Gleichung 2.4 als Multiplikator eingebettet ist und damit für alle Materialeigenschaften einen erheblichen Einfluss auf die Reflexionsamplitude hat. Diese Eigenschaft wird z.B. auch in [Mes06, YLZL06] durch Simulation und Messung bestätigt.

7.3.2.2. Schätzung des Aspektwinkels

Zur Bestimmung des Aspektwinkels an den Grenzflächen wurden in Abschnitt 4.2 bereits die Oberflächennormalen $\mathbf{n}_C = \mathbf{N}$ bestimmt. Da für den Radar der Aspektwinkel, bezogen auf das Radarkoordinatensystem, relevant ist, werden die Oberflächennormalen, gegeben im Kamerakoordinatensystem, zunächst in das Radarkoordinatensystem transformiert. Der Ursprung des Normalenvektors entspricht gerade Koordinaten im Kamerakoordinatensystem $\mathbf{x}_C$ und ist nach Gleichung 3.22 für jedes Kamerapixel zu bestimmen. Die Normalenvektoren stellen Richtungsvektoren dar. Wir definieren deren Start- und Endpunkt als $\mathbf{x}_{C,n,\text{start}} = \mathbf{x}_C$ und $\mathbf{x}_{C,n,\text{end}} = \mathbf{x}_C + \mathbf{n}_C$. Wie in Gleichung 3.33 beschrieben, werden diese beiden Punkte in das Radarkoordinatensystem transformiert und bilden so den Start- und Endpunkt ($\mathbf{x}_{R,n,start}$ bzw. $\mathbf{x}_{R,n,\text{end}}$) des Richtungsvektors in Radarkoordinaten ab,

$$\mathbf{x}_{R,n,\text{start}} = {}^{R\leftarrow C}\mathbf{R}\mathbf{x}_{C,n,\text{start}} + {}^{R\leftarrow C}\mathbf{t} \tag{7.17}$$

und

$$\mathbf{x}_{R,n,\text{end}} = {}^{R\leftarrow C}\mathbf{R}\mathbf{x}_{C,n,\text{end}} + {}^{R\leftarrow C}\mathbf{t}. \tag{7.18}$$

Der Ortsvektor der Oberflächennormalen ergibt sich nun wiederum als Differenz $\mathbf{x}_R = \mathbf{x}_{R,n,\text{end}} - \mathbf{x}_{R,n,\text{start}} = {}^{R\leftarrow C}\mathbf{R}\mathbf{n}_c$. Damit kann der Aspektwinkel aus der Oberflä-

chennormalen x_r und dem Einfallsvektor $\mathbf{x}_{R,n,\text{start}}$ berechnet werden, zu

$$\angle\left(\mathbf{x}_{R,n,\text{start}}, \mathbf{n}_R\right) = \cos^{-1}\left(\frac{\mathbf{x}_{R,n,\text{start}} \cdot \mathbf{n}_R}{\|\mathbf{x}_{R,n,\text{start}}\|\|\mathbf{n}_R(\mathbf{p})\|}\right). \tag{7.19}$$

7.3.2.3. Statistische Untersuchung zum Aspektwinkel

Da in Gleichung 2.4 die Winkelterme ausschließlich über die Cosinus-Funktion skaliert werden, werden wir nachfolgend den Term $\cos\left(\angle\left(\mathbf{x}_{r,n,\text{start}}, \mathbf{n}_r\right)\right)$ mit dem Synonym „Aspektwinkelwert“ verwenden. Dieser Wert und die in das Kamerabild projizierte Leistung/SNR aus Radar sind zur Veranschaulichung in Abbildung 7.10 für ein Kameraframe dargestellt.

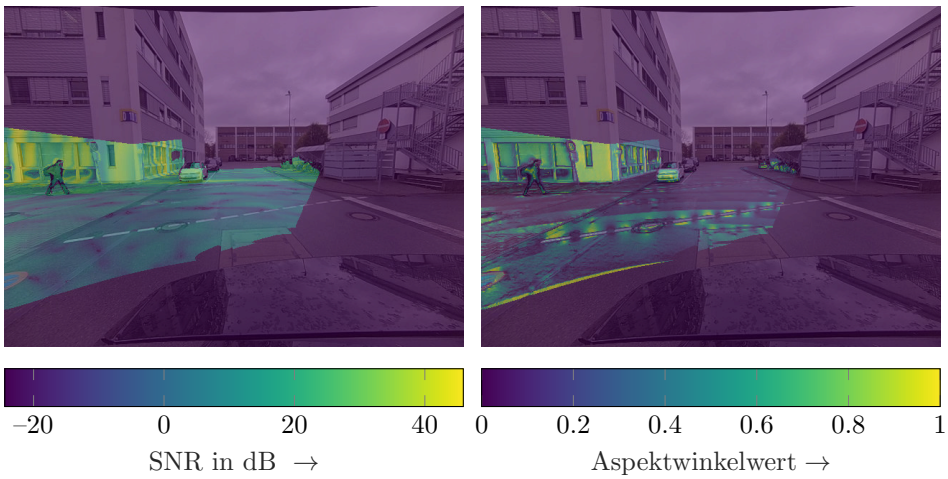


Abbildung 7.10.: **SNR vs. Aspektwinkel:** Links: Projiziertes SNR im Kamerabild. Rechts: Aspektwinkelwert im Kamerbild.

Zu erkennen ist in Abbildung 7.10, dass im Bereich der Gebäudewand, im linken Bildbereich, ein hoher Aspektwinkelwert geschätzt wurde. An gleicher Stelle ist das projizierte SNR aus der RD-map ebenfalls hoch. Ebenso ähnlich verhalten sich die Abbildungen von SNR und Aspektwinkelwert im Bereich des parkenden Fahrzeuges, des Fußgängers und größtenteils auch im Bereich der Straße. Die Straße verlief nahezu glatt und wies keine erkennbaren Oberflächenfehler auf. In weiten Teilen der Straße ist der geschätzte Aspektwinkelwert plausibel, da er als niedrig und homogen geschätzt wurde. Im Bereich des Ego-Fahrzeuges ist der geschätzte Aspektwinkelwert der Fahrbahn inhomogen und von hoher Varianz. Das ist unplausibel und lässt sich technisch durch die im Bereich des Ego-Fahrzeuges spärlichere Tiefenabtastung begründen, welche zu einem höheren Fehler der dichten Tiefenmaske führt, vgl. Abbildung 4.1. Eine Verbesserung der dichten Tiefenmaske und damit einhergehend der Aspektwinkelmaske wäre für die Zukunft wünschenswert, ist aber kein Bestandteil dieser Arbeit. Basierend auf diesem Bild lässt sich ein statistischer Zusammenhang zwischen Aspektwinkelmaske und SNR-Maske durchaus erahnen. Um diese subjektiv wahrgenommene Abhängigkeit anhand einer größeren Datenbasis zu festigen, wurde die Korrelation als Maß der (linearen) Abhängigkeit zwischen den Aspektwinkelwerten und den projizierten SNR-

Werten über den Pearson's Korrelationskoeffizient [FPP07] berechnet. Dabei ergab sich ein Korrelationskoeffizient von 0.5, was als moderate lineare Abhängigkeit gedeutet werden kann. Die Verteilung der Datenpunkte ist in Abbildung 7.11 in Form eines 2D-Histogramms dargestellt. Der Farbwert entspricht der Kombinationshäufigkeit und ist der dargestellten Farblegende zu entnehmen. Da die meisten Kombinationen im Bereich des Hintergundrauschens, gekennzeichnet durch niedriges SNR, entstanden sind, wurde die Helligkeit logarithmisch skaliert, so dass andere Bereiche visuell noch wahrnehmbar bleiben.

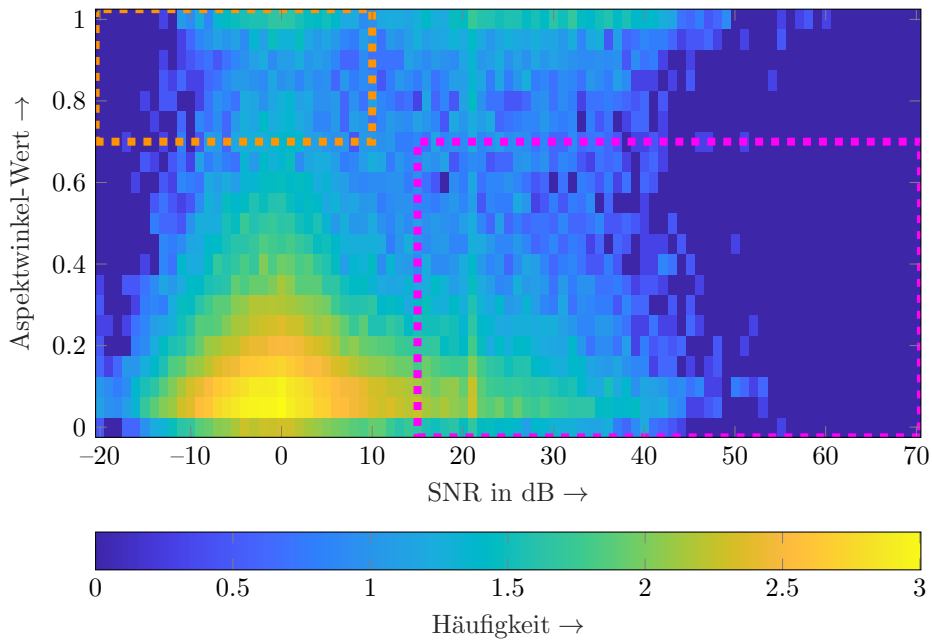


Abbildung 7.11.: **Häufigkeitsverteilung von SNR vs. Aspektwinkelwert:** Die Häufigkeiten sind logarithmisch skaliert.

Wie oben bereits erwähnt, ist eine große Häufung von Kombinationen im Bereich um SNR 0 dB zu erkennen, welche sich vermutlich aus dem Hintergundrauschen in der RD-map ergibt. Dann ist im oberen rechten Bereich eine Ansammlung von Pixeln mit hohem SNR und Aspektwinkel zu erkennen. Diese ersten beiden Ansammlungen stimmen dem hier zugrunde gelegten Modell für die automatische Bestimmung von Zielwerten über Aspektwinkelwerte zu. Dann ergeben sich noch Ansammlungen von Bereichen, bei denen

- das SNR hoch ist und der Aspektwinkelwert gering (hervorgehoben durch pinkfarbiges Rechteck)
- oder das SNR gering und der Aspektwinkelwert hoch ist (hervorgehoben durch orangefarbiges Rechteck).

Die entsprechenden Bereiche wurden in Abbildung 7.11 durch farbige Rechtecke gekennzeichnet. Für die Ursachenforschung dieser Anomalien wurden die RD-Zellen zu den Kombinationen entsprechend farblich kodiert im Kamerabild gekennzeichnet. Ein repräsentatives Beispiel ist in Abbildung 7.12 zu sehen, weitere Beispiele in Anh. A.2.

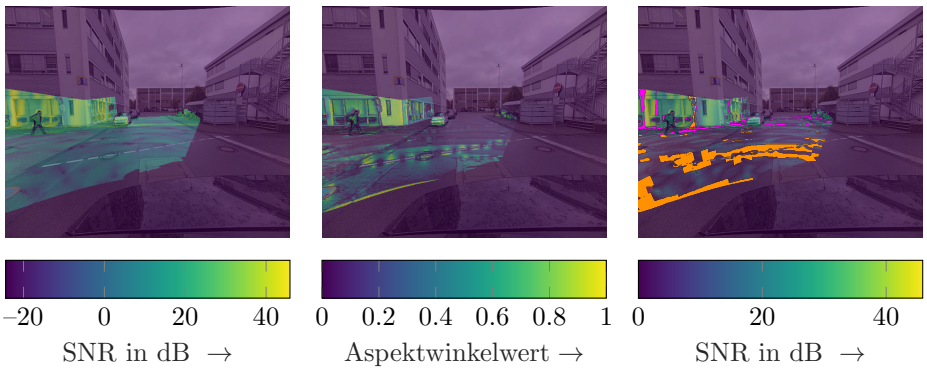


Abbildung 7.12.: **Gegenüberstellung von Aspektwinkel und SNR:** Links: Projiziertes SNR; Mitte: Aspektwinkelwert; Rechts: Projiziertes SNR und farblich markiert (pink, orange) Abweichungen von der Modellbeschreibung.

In Abbildung 7.12 ist links sowie rechts das projizierte SNR im Kamerabild und mittig der Aspektwinkelwert im Kamerabild gezeichnet. Zusätzlich sind im rechten Bild die oben genannten Anomalien markiert. Zu beobachten ist, dass die orange markierten Bereiche hauptsächlich dort zu finden sind, in denen der Aspektwinkel unplausibel ist. Dies kann leicht durch die hier verwendete Methode zur Tiefenvervollständigung und Oberflächennormalenschätzung hervorgerufen werden. Bessere Methoden zur Oberflächennormalenschätzung würden hier wahrscheinlich Abhilfe schaffen, sind jedoch nicht Bestandteil dieser Untersuchung und werden daher nicht getestet. Die pink markierten Bereiche sind in der Regel benachbart zu Bereichen mit hohem SNR und hohem Aspektwinkelwert. In der RD-map sind diese Pixel Nachbarzellen zum lokalen Maximum, welchem bedingt durch z.B. begrenzte Abtastung und Fensterung ebenfalls ein hoher Leistungswert zugeordnet wurde, der deutlich über dem Rauschlevel liegt. Es kann vermutet werden, dass diese Nachbarzellen zu den pink markierten Bereichen in Abbildung 7.12 führen. Es wird festgehalten, dass die Anomalien in der Statistik häufig durch fehlerhaft geschätzte Aspektwinkelwerte und ausgedehnte Leistungssignaturen im RD-map hervorgerufen werden.

Für die automatische Ermittlung der Zielwerte werden Kamerapixel mit einem Aspektwinkelwert ≥ 0.7 als positives Ziel und Pixel mit einem Wert < 0.7 als negatives Beispiel gewählt. Dieser Schwellwert ergab sich nach Sicht der stochastischen Verteilung aus Abbildung 7.11 als subjektiv plausibel. Die Zielmaske ergibt sich somit zu

$$\boldsymbol{\Omega}_{\mathrm{GT}}(\mathbf{p}) = \begin{cases} 1 & |\cos\left(\angle\left(\mathbf{x}_{r,n,\mathrm{start}}(\mathbf{p}), \mathbf{n}_r(\mathbf{p})\right)\right)| \geq 0.7 \\ 0 & \text{sonst} \end{cases} \tag{7.20}$$

Dieses Modell zur automatischen Identifikation von signifikanten Reflexionen/Grenzflächen, basierend auf Daten der Referenzsensorik, berücksichtigt nur den Aspektwinkelwert. Ein statistischer Zusammenhang zwischen Aspektwinkelwert und Reflexionsleistung wurde theoretisch durch die Fresnelschen Gleichungen motiviert und durch reale Messungen hier wahrgenommen. Weitere Merkmale, wie z.B. geschätzte Materialeigenschaft, wurden in dieser Arbeit zur Zielwertschätzung nicht berücksichtigt, können aus Sicht

des Autors aber relevante und interessante weitere wissenschaftliche Fragestellungen ergeben.

7.3.3. Netzwerkarchitektur

Für die Zieldetektion wurde die Netzwerkarchitektur aus der Winkelschätzung, siehe Unterabschnitt 7.2.3, leicht adaptiert. Änderungen sind, dass nur ein Eingangskanal vorhanden ist, siehe Unterabschnitt 7.3.1, und in der letzten Schicht die Sigmoid-Aktivierungsfunktion verwendet wurde. Die Änderung der Aktivierungsfunktion wurde vorgenommen, damit die Prädiktion der Zielwerte im Intervall $[0, 1]$ geschätzt wird. Eine visuelle Übersicht der Netzwerkarchitektur ist in Abbildung 7.13 dargestellt.

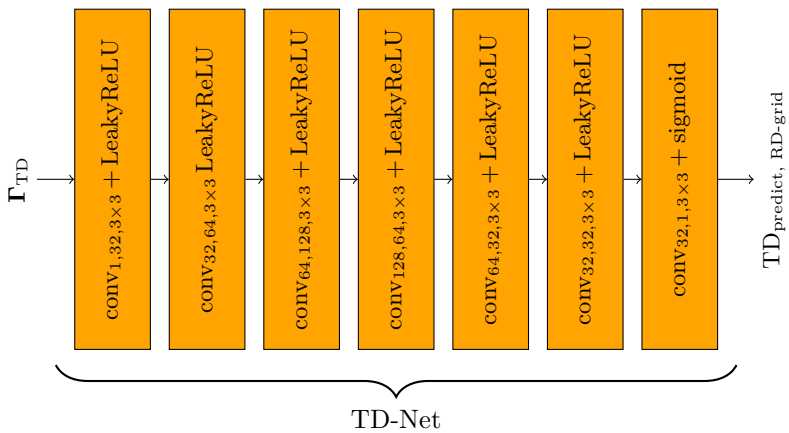


Abbildung 7.13.: **CNN Struktur für Zieldetektion:** Die Übersicht der Schichten.

Wir definieren die Übertragungsfunktion des NN zur Zieldetektion als

$$\mathrm{TD}_{\mathrm{predict,\ RD\text{-}grid}} = \text{TD-Net}(\boldsymbol{\Gamma}_{\mathrm{TD}}). \tag{7.21}$$

7.3.4. Assoziation von Prädiktion und Label durch Warping

Da auch bei der Zieldetektion die Zielwerte im Kamerabild definiert wurden, wird analog zur Winkelschätzung aus Unterabschnitt 7.2.4 das Warping der Prädiktionen vom RD-Gitter in das Kamerabild durchgeführt

$$\mathrm{TD}_{\mathrm{predict,\ cam}}(\mathbf{p}) = \eta_{\mathrm{BW}}\Big(\mathrm{TD}_{\mathrm{predict,\ RD\text{-}grid}}; v_r(\mathbf{p}), |\mathbf{x}_r(\mathbf{p})|\Big). \tag{7.22}$$

Da die Zieldetektion auf dem zweidimensionalen RD-map durchgeführt werden sollte, wurde die 2D-Interpolation verwendet.

7.3.5. Messung der Abweichung

Die Zieldetektion wird als binärer Entscheider abgebildet. Es wird entschieden, ob es sich bei einem Pixel in der RD-map entweder um ein Ziel oder Rauschen handelt. Es handelt sich also um eine sogenannte „Multiclass-Classification“. Nach Wahl der

Sigmoid-Aktivierungsfunktion der letzten Netzwerkschicht wird hier der gängigen Praxis gefolgt und die Kreuzentropie als Maß für die Abweichung der Prädiktion verwendet.

$$l_{\mathrm{TD}}(\mathbf{p}) = -\boldsymbol{\Omega}_{\mathrm{GT}}(\mathbf{p}) \log\left(\mathrm{TD}_{\mathrm{predict,\ cam}}(\mathbf{p})\right) - (1 - \boldsymbol{\Omega}_{\mathrm{GT}}(\mathbf{p})) \log\left(1 - \mathrm{TD}_{\mathrm{predict,\ cam}}(\mathbf{p})\right) \tag{7.23}$$

7.3.6. Allgemeine Selektion der Pixelmenge

Die Auswahl der Pixelmenge für die Optimierung wurde entsprechend Unterunterabschnitt 7.2.6.1 übernommen.

7.3.7. Gesamtkosten

Die Gesamtkosten ergeben sich als Mittelwert der Abweichungen über alle Skalierungsebenen und Pixel.

$$l_{\mathrm{TD,\ all}} = \frac{1}{|\mathcal{P}_{\mathrm{all}}|} \sum_{\mathbf{p} \in \mathcal{P}_{\mathrm{all}}} l_{\mathrm{TD}}(\mathbf{p}). \tag{7.24}$$

7.3.8. Initialisierung der Parameter

Die Initialisierung der Netzwerkparameter erfolgte analog der Beschreibung aus Unterabschnitt 7.2.8.

7.3.9. Optimierer

Die Optimierung der Netzwerkparameter erfolgte analog der Beschreibung aus Unterabschnitt 7.2.9.

Da der Trainingsprozess bei der Winkelprädiktion bereits nach wenigen Trainingsschritten konvergierte, wurde bei der Zieldetektion die initiale Schrittweite für die Optimierungen von 10^{-4} auf 10^{-5} reduziert.

7.3.10. Trainingsprozess

Der Verlauf der Kosten während des Trainings ist in Abbildung 7.14 dargestellt.

Es ist zu beobachten, dass das NN nach etwa 10000 Trainingsschritten größtenteils konvergiert ist und keine signifikante Änderung im Verlauf der Trainingskosten zu beobachten ist.

7.3.11. Ergebnis

Nach dem Training des NN wird nun die Schätzqualität ermittelt und gegenüber einem Referenzverfahren bewertet.

7.3.11.1. Referenzverfahren

Als Referenzverfahren für die Bewertung der Zieldetektion aus dem NN wird ein CFAR-Zieldetektor (siehe Unterabschnitt 2.1.4) aus der klassischen Radarsignalverarbeitung verwendet.

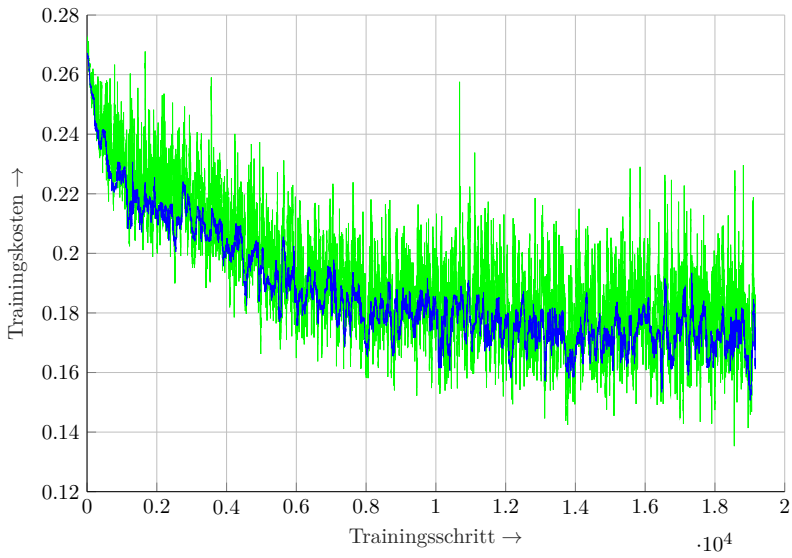


Abbildung 7.14.: **Verlauf des Trainingsprozesses Zieldetektor Netzwerk:** Die Kosten während des Trainings konvergieren bereits nach einigen Trainingsbeispielen. Grün: Trainingskosten je Beispiel. Blau: Geglätteter Verlauf der Trainingskosten

7.3.11.2. Beschreibung der Metriken

Für die automatische Berechnung der Prädiktionsgenauigkeit der Zieldetektion definieren wir eine passende Metrik. Dafür müssen sowohl Zielwerte als auch Prädiktoren als binäre Klassifikationen dargestellt sein. Durch Verwendung der Sigmoid-Aktivierungsfunktion erfolgt die Prädiktion jedoch im Intervall $[0, 1]$. Zur Binärisierung definieren wir eine Netwerkausgabe ≥ 0.5 als Prädiktion eines Ziels und entsprechend < 0.5 als Prädiktion von Rauschen. Es ergeben sich vier mögliche Fälle beim Vergleich von Zielwert $\boldsymbol{\Omega}_{\mathrm{GT}}(\mathbf{p})$ und Prädiktion $\mathrm{TD}_{\text{predict, cam}}(\mathbf{p})$:

- Richtig-positiv: $\mathcal{TP} = \left\{ \mathbf{p} \in \mathcal{P}_{\text{all}} \middle| \boldsymbol{\Omega}_{\mathrm{GT}}(\mathbf{p}) = 1, \mathrm{TD}_{\text{predict, cam}}(\mathbf{p}) \geq 0.5 \right\}$
- Falsch-negativ: $\mathcal{FN} = \left\{ \mathbf{p} \in \mathcal{P}_{\text{all}} \middle| \boldsymbol{\Omega}_{\mathrm{GT}}(\mathbf{p}) = 1, \mathrm{TD}_{\text{predict, cam}}(\mathbf{p}) < 0.5 \right\}$
- Falsch-positiv: $\mathcal{FP} = \left\{ \mathbf{p} \in \mathcal{P}_{\text{all}} \middle| \boldsymbol{\Omega}_{\mathrm{GT}}(\mathbf{p}) = 0, \mathrm{TD}_{\text{predict, cam}}(\mathbf{p}) \geq 0.5 \right\}$
- Richtig-negativ: $\mathcal{TN} = \left\{ \mathbf{p} \in \mathcal{P}_{\text{all}} \middle| \boldsymbol{\Omega}_{\mathrm{GT}}(\mathbf{p}) = 0, \mathrm{TD}_{\text{predict, cam}}(\mathbf{p}) < 0.5 \right\}$.

Bei der Fallunterscheidung werden ausschließlich Kamerapixel aus $\mathcal{P}_{\text{all}}$, siehe Unterunterabschnitt 7.2.6.1, berücksichtigt.

Zusätzlich definieren wir die Menge aller richtigen Klassifikationen zu $\mathcal{CC} = \mathcal{TP} \cup \mathcal{TN}$. Mit Hilfe dieser Mengendefinition berechnen wir die Klassifikationsgenauigkeit als die

Summe aller richtigen Entscheidungen, normiert über die Summe aller Entscheidungen zu

$$\text{Accuracy} = \frac{|\mathcal{CC}|}{|\mathcal{P}_{\text{all}}|}. \tag{7.25}$$

7.3.11.3. Quantitative Auswertung

Bei der Winkelschätzung ergeben sich die Zielwerte durch die native Auflösung der Referenzsensorik, insbesondere jener Winkelauflösung der Kamera. Für die Anwendung der Zieldetektion wurden die Zielwerte über die Modellbeschreibung des Aspektwinkels geschätzt. Dabei hat sich in Unterabschnitt 7.3.2 gezeigt, dass diese automatisch ermittelten Zielwerte, maßgeblich bedingt durch Fehler in der Apsketwinkelschätzung, teilweise unplausibel erscheinen. Beispielsweise ergaben sich hohe Aspektwinkelwerte auf geraden Flächen, siehe Abbildung 7.12. Es ist zu erwarten, dass diese Unplausibilität der Zielwerte sich als Rauschen in den Datenpunkten für Gleichung 7.25 bemerkbar machen und einen Vergleich der beiden Verfahren zur Zieldetektion erschweren wird. Dieses „Label noise“[5] wurde bereits in Unterabschnitt 7.3.2 deutlich und könnte in Zukunft durch den Einsatz einer verbesserten Aspektwinkelschätzung in der Signalverarbeitung der Referenzsensorik verbessert werden. Wir wollen uns daher primär der Fragestellung widmen, ob überhaupt ein zielführendes Training einer Zieldetektion mittels der Zielwertdefinition über Aspektwinkel erreicht werden kann.

Die in Gleichung 7.25 definierte Metrik wurde für sämtliche Frames im Testdatensatz berechnet. Eine Übersicht der Datenpunkte ist in Abbildung 7.15 dargestellt.

Für jedes Kamerabild, bestehend aus einer großen Menge an Pixeln, wurde die mittlere Abweichung von Prädiktion und Zielwert nach Gleichung 7.25 ermittelt. Durch die Mittlung über die Pixelmenge ergibt sich pro Kamerabild eine Genauigkeit zwischen 0% und 100%, welche als Datenpunkt in die linke Seite von Abbildung 7.15 eingetragen wurde. Für den CFAR Detektor wurden diese Punkte rot dargestellt, für die NN basierte Prädiktion grün. Die Bildpunkte wurden auf der rechten Bildseite in Form eines Histogramms gesammelt. In dem Histogramm ist zu sehen, dass die NN basierte Prädiktion (grün) tendenziell eine höhere Klassifikationsgenauigkeit erreicht als der CFAR Detektor. Dies ist nicht verwunderlich, da die NN-basierte Prädiktion explizit auf den vorhandenen Daten (Trainingsdatensatz) optimiert wurde. Gleichzeitig fällt auf, dass die Streuung der Genauigkeit im Histogramm sehr stark ist, besonders bei der CFAR-Prädiktion über den gesamten Wertebereich erfolgt. Es ist denkbar, dass ein Teil der Streuung durch das in Unterunterabschnitt 7.3.2.3 beschriebene „label-noise“ verursacht wurde.

Aufgrund der starken Streuung fällt es schwer, einen klaren Favoriten bei der Zieldetektion hervorzuheben. Die wesentliche wissenschaftliche Fragestellung für diesen Versuch war aber ohnehin nicht, ob ein NN eine bessere Zieldetektion erreichen kann als ein klassischer Zieldetektor, sondern die fundamentalen Fragestellungen, (a) ob mit dem vorgestellten Warping auch eine Zieldetektion trainiert werden kann und (b), ob sich der Aspektwinkel zum Trainieren eines NN für die Anwendung der Zieldetektion eignet. Da bei der Zieldetektion mittels CFAR-Detektor ausschließlich das SNR für

[5] Im Kontext des maschinellen Lernens wird bei fehlerhaften und verrauschten Zielwerten auch von „Label Noise“ gesprochen

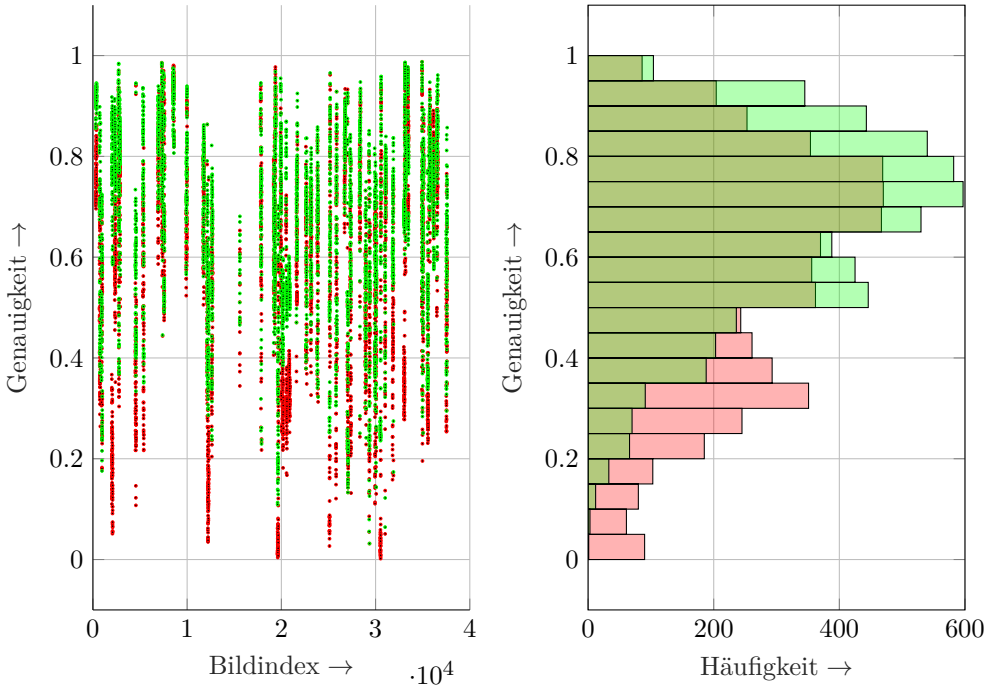


Abbildung 7.15.: **Genauigkeit:** Die Abweichung der Prädiktion gegenüber den automatisch generierten Labeln für alle Bilder im Testdatensatz. Rot: Abweichungen für CFAR; Grün: Abweichungen für NN Prädiktion.

die Bestimmung von Zielen verwendet wird und bereits in Unterabschnitt 7.3.2 eine mindestens moderate lineare Abhängigkeit zwischen SNR und Aspektwinkel identifiziert wurde, überrascht es nicht, dass CFAR und NN-Detektoren bei der oben genannten Metrik vergleichbare Ergebnisse erzielen konnten.

7.3.11.4. Qualitative Auswertung

Zur Veranschaulichung der Zieldetektion sind in Abbildung 7.16 die annotierten Zielwerte über Aspektwinkel und die Prädiktionen aus NN und CFAR sowohl im Kamerabild als auch RD-map dargestellt.

In Bildregionen mit Fahrzeugen oder Häuserwänden scheinen hohe Aktivierungen vorzuliegen. In Bildregionen mit flachem Einstrahlwinkel, wie der Fahrbahn, scheint bei beiden Verfahren tendenziell eher niedrige Aktivierung sichtbar zu sein. Die Beispiele beider Detektoren scheinen plausibel zu sein. Im oben und unten dargestellten Parkszenario sind die Aktivierungen aus NN etwas fokussierter auf den Fahrzeugoberflächen, wohingegen die Aktivierungen mit CFAR auch bei den dazwischenliegenden leeren Flächen hoch sind. Dieser Unterschied könnte mit einer statistischen Optimierung des CFAR-Detektors gegen die Zielwerte vermutlich verringert werden und somit die Schätzungen beider Detektoren weiter angeglichen werden.

Um Unterschiede zwischen CFAR und NN-basierter Zieldetektion zu finden, wurden in Anh. A.2.1 weitere Untersuchungen angestellt.

Analog zu den Ergebnissen aus der quantitativen Auswertung ergeben sich auch bei der qualitativen Bewertung der Prädiktionen nur marginale Unterschiede. Aus Sicht des

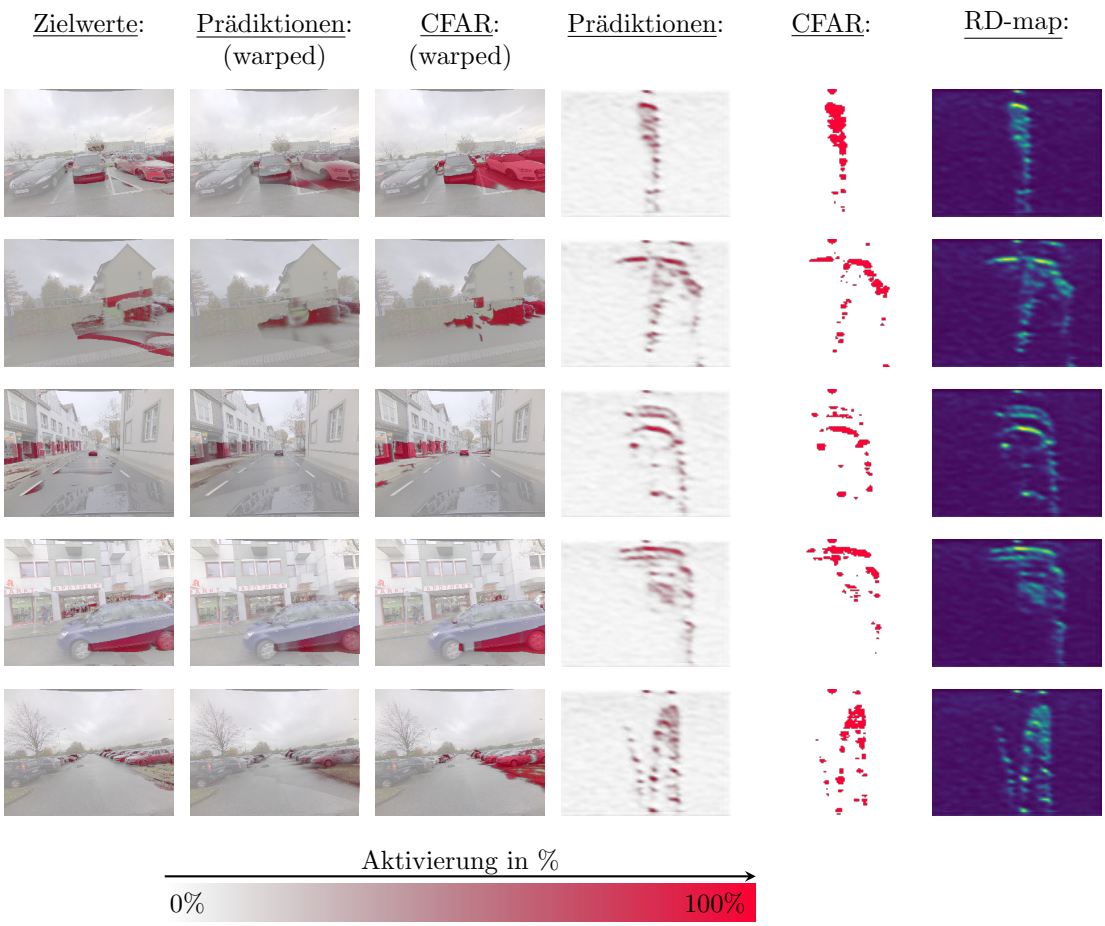


Abbildung 7.16.: **Beispiele der Zieldetektion aus dem Testdatensatz**. Von links nach rechts: RGB-Bild mit farbkodierten Zielen, RGB-Bild mit farbkodierten Prädiktionen aus NN, RGB-Bild mit farbkodierten Prädiktionen aus CFAR, RD-map mit farbkodierten Prädiktionen aus NN, RD-map mit farbkodierten Prädiktionen aus CFAR, RD-map.

Autors ein weiteres Indiz dafür, dass die automatische Annotation der Zielwerte für die Zieldetektion mittels Aspektwinkelwert eine valide Methode ist.

7.3.11.5. Laufzeitanalyse

Zur Bewertung der Laufzeit beider Detektionsverfahren wurde die Anzahl der Operationen geschätzt und in Tabelle 7.6 eingetragen. Die Operationszahl für den CFAR-Algorithmus skaliert mit der Operationszahl zur Berechnung des adaptiven Referenzwertes im 11×11 Fenster ($11 \cdot 11 = 121$ Additionsoperationen) und der Pixelzahl in der RD-map ($128 \times 96 = 12288$). Als Produkt ergab sich eine gesamte Operationszahl von 1486848.

Die Anzahl der Operationen für das NN wurde, wie in Unterunterabschnitt 7.2.11.5 beschrieben, ermittelt.

Die resultierenden Laufzeiten der beiden Verfahren unterscheiden sich um mehrere Zehnerpotenzen. Alle in Unterunterabschnitt 7.2.11.5 gemachten Bewertungen gelten

Tabelle 7.6.: **Anzahl der Operation für TD Verfahren:**

Algorithmus	Anzahl der Operationen	geschätzte Laufzeit bei 300 MHz
CA-CFAR	1486848	5 ms
NN	4785438700	16 s

auch für diesen Abschnitt.

7.3.11.6. Antwort zur wissenschaftlichen Hypothese

Bei den Untersuchungen zur Zieldetektion wurde die Antwort auf folgende Fragestellungen gesucht:

- Kann eine Zieldetektion mittels vorgestelltem Warping trainiert werden?
- Ist der Aspektwinkel ein valides Maß für die automatische Bestimmung von Zielwerten für die Zieldetektion?

Da die trainierte Zieldetektion vergleichbare Ergebnisse zum klassischen CFAR-Detektor erreicht, können beide Fragestellungen bejaht werden. Aus Sicht des Autors wird aber empfohlen, die Qualität der Aspektwinkelschätzung weiter zu verbessern.

In Abschnitt 7.5 wird außerdem noch ein datenbasierter Schätzer vorgestellt, welcher die Leistungsbeiträge der individuellen Kamerapixel schätzt. Es ist möglich, dass dieser in Zukunft auch anstelle der (modellbasierten) Aspektwinkelschätzung für die Zielwertbestimmung bei Detektionsaufgaben verwendet werden kann. Hierdurch ergäbe sich der Vorteil, dass nicht nur geometrische Eigenschaften der Objekte berücksichtigt werden, sondern evtl. auch andere physikalische Eigenschaften, wie z.B. Materialeigenschaften.

7.4. Semantische Segmentierung

Als natürliche Erweiterung zur Zieldetektion wird in diesem Abschnitt die semantische Segmentierung der RD-maps durchgeführt. Bei der semantischen Segmentierung handelt es sich um die Klassifikation der einzelnen Zellen/Pixel in der RD-map. Die Klassifikation wird hier in die Zielklassen „Fußgänger“, „Fahrzeuge“und „stationäre Ziele“ vorgenommen. Zum einen kommen diese Objektklassen im Straßenverkehr häufig vor. Zum anderen weisen diese Objekte im Messraum aber auch spezifische Signaturen auf, so dass eine Trennung über die vorhandenen Radardaten technisch möglich ist. Diese spezifischen Signaturen werden nachfolgend kurz beschrieben.

Bei den stationären Zielen handelt es sich um geostationäre Reflexionen, also Reflexionen, deren Dopplersignatur nur durch die Bewegung des Radarsensors bzw. des Ego-Fahrzeuges über Grund bestimmt ist.

Als Fußgänger bezeichnen wir Menschen in Bewegung. Durch den speziellen Bewegungsapparat und das Bewegungsmuster ergeben sich aus Sicht des Radars spezifische Dopplersignaturen. Der bewegte menschliche Körper kann als Mehrkörpermodell angesehen werden. Durch die oszillierende Bewegung der Extremitäten ergeben sich ausgedehnte Dopplersignaturen in der RD-Map, welche maßgeblich in Abhängigkeit von der Schrittweite, Schrittfrequenz und Schrittphase variieren und als „Mikro-Doppler“ bekannt sind,

siehe z.B. [vDG08]. Der Torso des Menschen liefert die in Doppler nahezu konstante Reflexion, den „Makro-Doppler“.

Als letzte Klasse betrachten wir alle Reflexionen von typischen Fahrzeugen wie PKW, LKW und Motorrad, welche nicht geostationär sind. Sich also durch Bewegung über Grund auszeichnen.

Wie bei der Zieldetektion auch, liefert die Referenzsensorik keine native Annotation für diese Aufgabe. Basierend auf den Daten der Referenzsensorik wird diese Annotation nachfolgend automatisch mittels einer Instanzsegmentierung auf den Kamerabildern geschätzt.

7.4.1. Eingangsdaten

Die semantische Segmentierung soll unter anderem zwischen geostationären Reflexionen und bewegten Reflexionen diskriminieren können. Diese Diskrimination wird in der Literatur häufig als Moving-Target-Indication (MTI) bezeichnet und kann mittels Kenntnis der Eigenbewegung und des Einfallswinkels der Reflexionen über klassische Methoden mit guter Genauigkeit vorgenommen werden, siehe beispielsweise [EB1, EB2, EB3, EB5]. Darüber hinaus wurde z.B. in [EB4] eine Erweiterung der klassischen MTI mit einem Recurrent-Neural-Network (RNN) vorgenommen, bei welchem einzelne Detektionen des Radars geclustert und über ein RNN klassifiziert wurden.

Das hier verwendete NN wird mit zwei Eingangskanälen ausgestattet und versorgt. In dem ersten Eingangskanal wird die RD-map **RD** eingespeist. In dem zweiten Eingangskanal wird die Prädiktion einer klassischen MTI (**MTI**) eingespeist

$$\boldsymbol{\Gamma}_{\mathrm{M}} = \begin{bmatrix} \mathbf{RD} \\ \mathbf{MTI} \end{bmatrix}. \tag{7.26}$$

Für die Berechnung von **MTI** wird eine Adaption der MTI aus [EB3] verwendet. Bei der Implementierung aus [EB3] handelt es sich um einen Hypothesentest mit harter Schwellwertentscheidung, welche eine binäre Entscheidung erzwingt. Grundlage dafür ist die Definition der relativen Bewegung nach Gleichung 2.6.

Wie zuvor erwähnt, kann ein Radarsensor nur die relative radiale Geschwindigkeit vermessen und auflösen, welche sich für geostationären Reflektoren aus der ersten Zeile von Gleichung 2.6 ergibt zu

$$v_{\mathrm{r,stat.}} = -\cos(\phi_{\mathrm{az}})\cos(\phi_{\mathrm{el}})\frac{\delta \mathbf{p}_{[x]}}{\delta t} - \sin(\phi_{\mathrm{az}})\cos(\phi_{\mathrm{el}})\frac{\delta \mathbf{p}_{[y]}}{\delta t} - \sin(\phi_{\mathrm{el}})\frac{\delta \mathbf{p}_{[z]}}{\delta t} = -\frac{\mathrm{d}\mathbf{b}_{[x]}}{\mathrm{d}t}. \tag{7.27}$$

Zu beachten ist, dass in Gleichung 2.6 die Bewegung eines Punktes im Radarkoordinatensystem beschrieben wurde. Da nun die Geschwindigkeit des Sensors verwendet wird, muss entsprechend das Vorzeichen der Bewegungen umgekehrt werden, wodurch der Vorzeichenwechsel gegenüber Gleichung 2.6 zu erklären ist. Damit kann durch Einsetzen der vom Radar gemessenen Einfallswinkel und der über DGPS-INS gemessenen Sensorgeschwindigkeit ein Schätzwert für die radiale Geschwindigkeit einer Reflexion unter der Annahme von Geostationarität berechnet werden. Ist die Differenz μ_E aus dieser geschätzten Geschwindigkeit und der vom Radar gemessenen Radialgeschwindigkeit hinreichend groß

$$\mu_E = v_r - v_{\mathrm{r,stat.}}, \tag{7.28}$$

so kann mit hoher Wahrscheinlichkeit davon ausgegangen werden, dass es sich bei einer untersuchten Reflexion nicht um ein stationäres Ziel handelt. Die Berücksichtigung von Ungenauigkeiten in Winkel- und Geschwindigkeitsschätzung wurde in [EB3] beschrieben, kann in ähnlicher Form aber auch vergleichbaren Veröffentlichungen entnommen werden, z.B. [Kel17]. Diese Ungenauigkeit fassen wir durch σ_E zusammen. Anstelle der binären Entscheidung wird eine Transformation über eine Gauß-Aktivierungsfunktion vorgenommen

$$\Pr_{c_{\text{moving}}} = \exp\left\{-\frac{\mu_E^2}{\left(\sigma_E \cdot Q^{-1}\left(\alpha/2\right)\right)^2}\right\}. \tag{7.29}$$

Wobei $\Pr_{c_{\text{moving}}}$ eine Pseudowahrscheinlichkeit der Klassenzugehörigkeit angibt. Eine niedrige Aktivierung bedeutet, dass es sich beim untersuchten Signal wahrscheinlich nicht um einen geostationären Reflektor handelt. Bei hoher Aktivierung kann es sich weiterhin um einen geostationären Reflektor handeln.

Im Gegensatz zur harten Entscheidung (engl.: „hardstep“) ist die Gauß-Aktivierungsfunktion differenzierbar und etwaige Fehler in der MTI könnten vollständig zurückpropagiert werden. Des Weiteren ergeben sich aus der MTI mittels Gauß-Aktivierungsfunktion kontinuierliche Werte, anstelle der binären Werte beim „Hardstep“. Die Motivation dahinter ist also: a) Fehler in der MTI könnten beim Training ermittelt, durch die MTI-Schicht propagiert werden und schließlich die Winkelschätzung optimiert werden. Und b) durch die kontinuierliche Variable kann das NN die Signaldynamik eigenständig erlernen. Untersuchungen zu diesen Hypothesen werden in dieser Arbeit jedoch nicht durchgeführt.

Ein Beispiel der MTI ist in Abbildung 7.17 dargestellt. In der ersten Spalte ist das Szenario aus Sicht der nach hinten aus dem Egofahrzeug gerichteten Kamera dargestellt. Das Ego-Fahrzeug fährt auf einer Straße und wird von einem Paketdienstfahrzeug verfolgt. Das korrespondierende RD-map ist rechts daneben dargestellt. In der dritten Spalte ist das Ergebnis der MTI in der RD-map ($\Pr_{c_{\text{moving}}}$) dargestellt. Die gelben Regionen zeigen Bereiche hoher Werte von $\Pr_{c_{\text{moving}}}$ und entsprechend bewegte Reflexionen an. Dunkle Regionen zeigen hingegen niedrige Werte von $\Pr_{c_{\text{moving}}}$ und mögliche geostationäre Reflexionen an. Durch die Anordnung im RD-Gitter kann die MTI-Prädiktion über Gleichung 6.14 in das Kamerabild projiziert werden. Das Ergebnis der Projektion ist rechts dargestellt. Es ist zu sehen, dass die Pixel an der Position des Paketdienstfahrzeuges mit hoher Konfidenz als bewegte Ziele erkannt wurden. Die verbleibenden Pixel im Kamerabild sind geostationär und weisen entsprechend eine geringe Konfidenz für bewegte Reflexion auf.

Wir definieren die Menge der Pseudowahscheinlichkeiten für sämtliche Pixel im RD-Gitter als **MTI**.

7.4.2. Zielwerte

Für die semantische Segmentierung wird eine Einteilung der Kamerapixel in die gesuchten Klassen vorgenommen. Für die Klassifikation der Pixel in geostationäre Pixel wird die aus dem Szenenfluss geschätzte kartesische Geschwindigkeit mit der aus dem DGPS-INS gelesenen Ego-Geschwindigkeit im Kamera-Koordinatensystem kompensiert. Diese

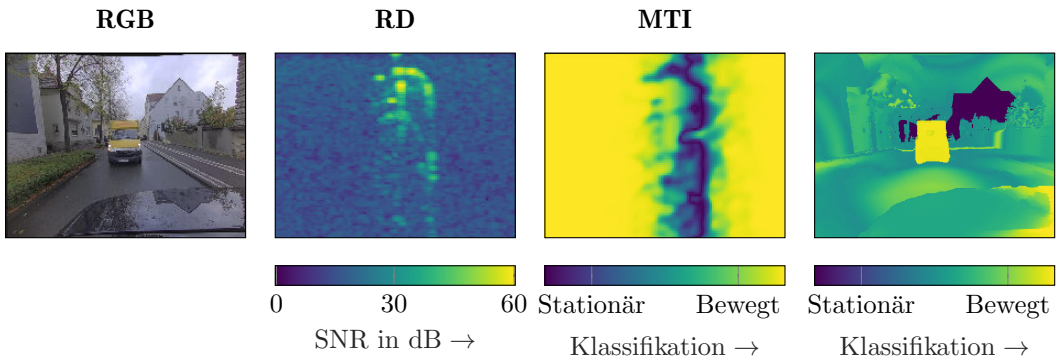


Abbildung 7.17.: **Beispiel für eine Bewegtzielerkennung**. Dargestellt ist ein Szenario, in welchem sich das Ego-Fahrzeug bewegt, während der Paketdienstwagen folgt. Links dargestellt ist das Kamerabild. Rechts daneben die RD-map. Wiederum rechts daneben die Bewegtzielerkennung in der RD-map. Rechts die Bewegtzielerkennung projiziert in das Kamerabild.

entspricht gerade der Vordergrundbewegung $\boldsymbol{\xi}_{fg}$ aus Gleichung 5.5. Anschließend wird diese Relativgeschwindigkeit in das Radarkoordinatensystem zu $\boldsymbol{\xi}_{fg,\text{radar}}$ transformiert. Die Transformation wird analog zu Gleichung 5.26 vorgenommen

$$\boldsymbol{\xi}_{fg,\text{radar}}(\mathbf{p}) = {}^{R\leftarrow C}\mathbf{R}\boldsymbol{\xi}_{fg}(\mathbf{p}). \tag{7.30}$$

Da der Radarsensor nur die Radialgeschwindigkeit messen kann, wird die Geschwindigkeit $\boldsymbol{\xi}_{fg,radar}$ außerdem als relative Radialgeschwindigkeit aus Sicht des Radarsensors dargestellt. Diese Berechnung erfolgt analog zu Gleichung 5.25 als

$$v_{\text{r, fg.}}(\mathbf{p}) = \frac{\mathbf{x}_R(\mathbf{p})^T}{|\mathbf{x}_R(\mathbf{p})|}\boldsymbol{\xi}_{fg,\text{radar}}(\mathbf{p}). \tag{7.31}$$

Der Betrag von $v_{\text{r, fg.}}$ definiert nun die Abweichung der Radialgeschwindigkeit von stationären Reflexionen. Da die Geschwindigkeitsschätzung imperfekt ist, werden alle Pixel als geostationär klassifiziert, deren Betrag von $v_{\text{r, fg.}}$ innerhalb eines Toleranzbereiches $\Delta_{\text{v,stationär}} = 0.2\,\text{m}\,\text{s}^{-1}$ liegt. Die Wahl des Schwellwertes erfolgte unter Berücksichtigung der Geschwindigkeitsauflösung des verwendeten Radarsensors sowie des ermittelten mittleren Schätzfehlers der Geschwindigkeit nach Tabelle 5.2. Die Maske für stationäre Ziele $\boldsymbol{\Psi}_{\text{SS, stat.}}$ ergibt sich somit zu

$$\boldsymbol{\Psi}_{\text{SS, stat.}}(\mathbf{p}) = \begin{cases} 1 & \left|v_{\text{r, fg.}}(\mathbf{p})\right| < \Delta_{\text{v,stationär}} \\ 0 & \text{sonst} \end{cases} \tag{7.32}$$

Aus den Instanzmaske aus Abschnitt 4.3 berechnen sich die Zielmasken für Fußgänger und Fahrzeuge als Schnittmenge der bewegten Pixel zu

$$\boldsymbol{\Psi}_{\text{SS, ped.}} = \mathbf{M}_{\text{Ped.}} \cap \boldsymbol{\Psi}_{\text{SS, stat.}} \tag{7.33}$$

und

$$\boldsymbol{\Psi}_{\text{SS, veh.}} = \mathbf{M}_{\text{Veh.}} \cap \boldsymbol{\Psi}_{\text{SS, stat.}}. \tag{7.34}$$

Beispiele der Zielmasken werden später (Abbildung 7.21) gezeigt.

7.4.3. Netzwerkarchitektur

Für die semantische Segmentierung wurde die Netzwerkarchitektur aus der Zieldetektion, siehe Unterabschnitt 7.3.3, adaptiert. Änderungen sind, dass die Eingangskanäle über separierte Pfade vorprozessiert werden und in der letzten Schicht drei Ausgangskanäle, je einer pro Klasse, vorhanden sind. Die Aktivierung der Ausgangskanäle wird über eine Softmax vorgenommen. Daraus ergibt sich eine „multi-class“ Prädiktion, bei welcher die Summe der Netzwerkausgabe eins ergibt und eine Klassifikation in eine einzelne Klasse gewollt ist. Alternativ ließe sich auch über die Entkopplung der letzten Schicht über z.B. die Sigmoid-Aktivierungsfunktion eine „multi-label“ Prädiktion vornehmen, bei welcher mehrere Klassen parallel hohe Aktivierungen aufweisen dürften.

Da die MTI nach Ansicht des Autors bereits gute Ergebnisse liefert, bestand kein Bedarf, davon wesentlich abzuweichen. Es wurde analog zu „Residual neural network“ [HZRS16] eine „skip connection“ der Eingangsdaten durchgeführt. Im Pfad der „skip connection“ wurde eine inverse Sigmoid-Funktion ($\text{sig}^{-1}(x) = \log\left(\frac{x}{1-x}\right)$) verwendet, um aus den **MTI** Netzwerkaktivierungen zu simulieren.

Eine visuelle Übersicht der Netzwerkarchitektur ist in Abbildung 7.18 dargestellt.

Wir definieren die Übertragungsfunktion des NN zur semantischen Segmentierung als

$$\text{M}_{\text{predict, RD-grid}} = \text{M-Net}(\boldsymbol{\Psi}_{\text{M}}). \tag{7.35}$$

7.4.4. Assoziation von Prädiktion und Label durch Warping

Wie bei der Winkel- und Zieldetektion findet für die Assoziation von Prädiktion und Zielwert ein Warping der Prädiktion in das Kamerabild, analog zu Unterabschnitt 7.2.4, statt. Für ein Pixel $\mathbf{p}$ ergibt sich die Projektion zu

$$\text{M}_{\text{predict, cam}}(\mathbf{p}) = \eta_{\text{BW}}\Big(\text{M}_{\text{predict, RD-grid}}; v_r(\mathbf{p}), |\mathbf{x}_r(\mathbf{p})|\Big). \tag{7.36}$$

7.4.5. Messung der Abweichung

7.4.5.1. Skalierungsraum

Analog zur Winkelschätzung, siehe Abschnitt A.1, wurde auch bei der semantischen Segmentierung ein Skalierungsraum der Kosten erstellt, um so den Einfluss von fehlerhafter Geschwindigkeits- und Tiefenschätzung in den Referenzdaten zu verringern. Die Skalierungskosten ergeben sich zu:

$$\text{M}_{\text{prediction, scaled}}(u, v, s) = \frac{\sum_{u_s=-s}^{s}\sum_{v_s=-s}^{s}\text{M}_{\text{prediction}}(u+u_s, v+v_s)(\mathbf{RD}(u+u_s, v+v_s) - \mathbf{RD}_{\min})}{(2s+1)^2\sum_{u_s=-s}^{s}\sum_{v_s=-s}^{s}(\mathbf{RD}(u+u_s, v+v_s) - \mathbf{RD}_{\min})}. \tag{7.37}$$

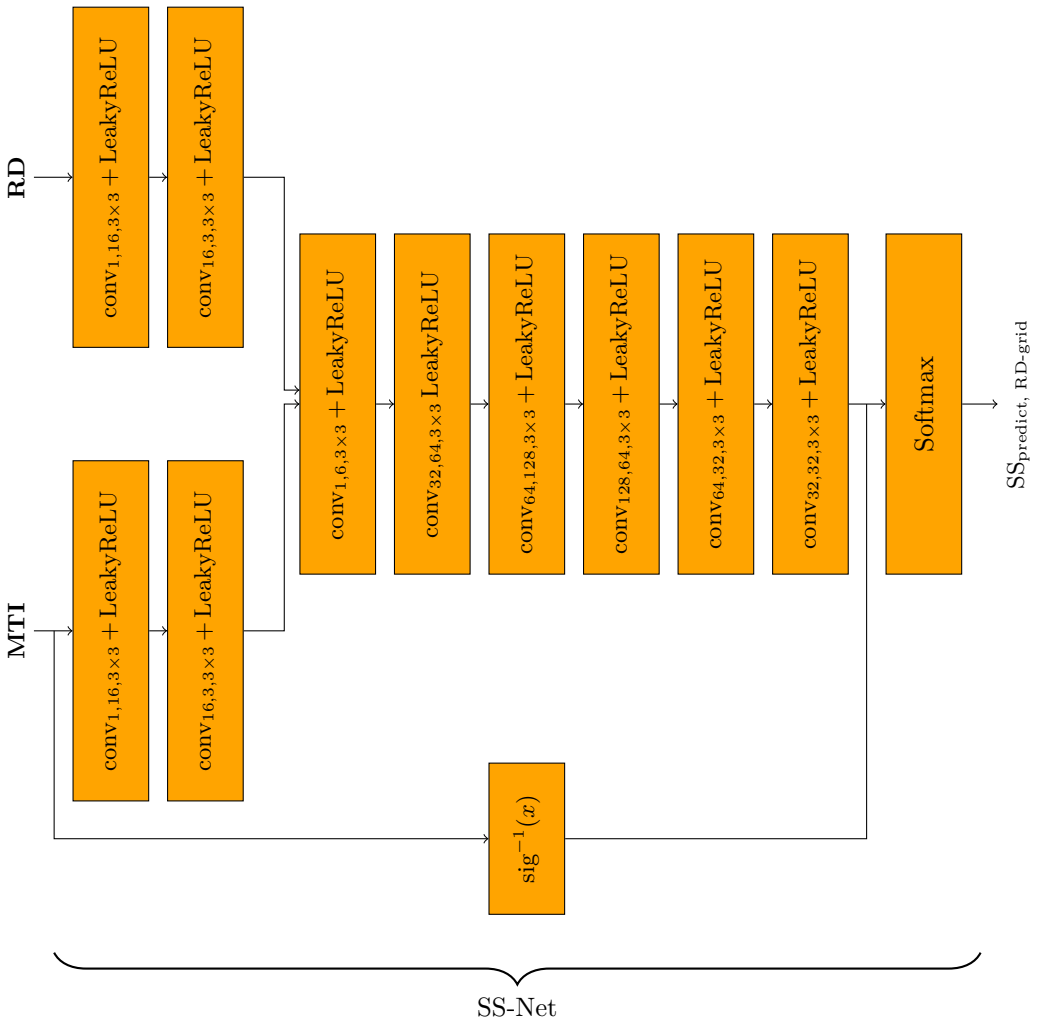


Abbildung 7.18.: **CNN Struktur für semantische Segmentierung:** Die Übersicht der Schichten.

Die Prädiktion im Kamerabild ergibt sich somit zu

$$\mathrm{M}_{\mathrm{predict,\ cam,\ scaled}}(\mathbf{p}) = \eta_{\mathrm{BW}}\Big(\mathrm{M}_{\mathrm{predict,\ scaled}}; v_r(\mathbf{p}), |\mathbf{x}_r(\mathbf{p})|\Big). \tag{7.38}$$

7.4.5.2. Abweichung

Wie in Unterabschnitt 7.4.3 beschrieben, wird bei der semantischen Segmentierung eine „multi-class“ Prädiktion durchgeführt. Ein praktisches Problem beim Training des NN ergab sich durch stark unausgewogene Klassenhäufigkeiten (engl.: „class-imbalance“) des Datensatzes. Ein Großteil der Pixel ist der Klasse geostationärerer Reflexionen zugeordnet, nur ein Bruchteil den Klassen Fußgänger und Fahrzeug. Dies repräsentiert durchaus die statische Verteilung realer Fahrszenarien, kann beim Training eines NN jedoch dazu führen, dass eine einzelne Klasse favorisiert wird. Es könnte also passieren, dass das trainierte NN ausschließlich geostationäre Pixel detektiert. Um diesen Effekt

zu vermeiden und die Klassenimbalance zu berücksichtigen, wird hier mittels „Focal-Loss“ [LGG+20] Kostenfunktion optimiert. Dabei handelt es sich um eine Erweiterung der Kreuzentropie, bei der die Kosten mit der Konfidenz der Prädiktion gewichtet werden. Hierdurch wird der Fokus von einfacher Entscheidung hin zu schwierigeren Entscheidungen verlagert.

Zum übersichtlichen Vergleich von Kreuzentropie und Focal-Loss sind die Definitionen aus [LGG+20] nachfolgend dargestellt

$$\mathrm{CE}(p_t) = -\alpha_t \log\left(p_t\right) \tag{7.39}$$

und

$$\mathrm{FL}(p_t) = -(1-p_t)^{\gamma} \alpha_t \log\left(p_t\right), \tag{7.40}$$

wobei p_t die Netzwerkprädiktion ist, α_t der Zielwerte und $\gamma \geq 0$ der Fokussierungsparameter. Bei der Wahl $\gamma = 0$ wird Focal-Loss zur Kreuzentropie. Bei steigendem γ werden die einfachen bzw. eindeutigen Klassifikationen mit sinkender Größe gewichtet.

Unter Einführung von α_{M}, einer von der Klasse abhängigen Skalierung für die Positivbeispiele, ergibt sich die Kostenfunktion für die semantische Segmentierung zu

$$\begin{aligned} l_{\mathrm{M}}(\mathbf{p}, s) = &-\alpha_{\mathrm{M}} \left(1 - \mathrm{M}_{\mathrm{predict,scaled}}(\mathbf{p}, s)\right)^{\gamma} \Psi_{\mathrm{M}}(\mathbf{p}) \log\left(\mathrm{M}_{\mathrm{predict,scaled}}(\mathbf{p}, s)\right) \\ &- \left(\mathrm{M}_{\mathrm{predict,scaled}}(\mathbf{p}, s)\right)^{\gamma} \left(1 - \Psi_{\mathrm{M}}(\mathbf{p})\right) \log\left(1 - \mathrm{M}_{\mathrm{predict,scaled}}(\mathbf{p}, s)\right). \end{aligned} \tag{7.41}$$

Die Wahl der Parameter der Kostenfunktion (α_{M}, γ, s) für die Optimierung hat einen Einfluss auf die trainierten NN-Parameter. Da keine Anhaltspunkte für diese Werte vorhanden waren, wurden mehrere NNen mit unterschiedlicher Parameterwahl trainiert. Eine Übersicht der Parameterwahl ist in Tabelle 7.7 zu finden.

Tabelle 7.7.: **Parameterwahl für das Training des NN für die semantischen Segmentierung**.

Name	γ	α_{M}(Fußgänger)	Skalierungsebenen s
NN_0	0	10	0
NN_1	0	10	2
NN_2	0	100	0
NN_3	0	100	2
NN_4	2	10	0
NN_5	2	10	2
NN_6	2	100	0
NN_7	2	100	2

Für die Anwendung der Winkelschätzung wurde durch die Anwendung der Skalierungsebenen eher ein Nachteil in der Winkelschätzqualität festgestellt, siehe Unterunterabschnitt 7.2.11.3. Es wird vermutet, dass Fehler in der Szenenflussschätzung bei der semantischen Segmentierung größere Auswirkungen haben als bei der Winkelschätzung, weil hier speziell gegen bewegte Objekte trainiert wird. Bei der Winkelschätzung wurde gegen alle Pixel trainiert, unter denen geostationäre Pixel dominant sind, welche einen deutlich geringeren Schätzfehler des Szenenflusses aufweisen und daher weniger häufig

Label-Noise aufweisen. Es soll daher noch einmal getestet werden, ob die Anwendung der Skalierungsebenen für die semantische Segmentierung vorteilhaft sein kann.

Durch die Verwendung der „Focal-Loss“ Kostenfunktion und des Fokussierungsparameters wird die Klassenimbalance bekämpft, was der Prädiktion dynamischer Objekte zu Gute kommen sollte. Für Objekte der Klasse Fußgänger soll zusätzlich eine manuelle Skalierung über den Parameter α_{M}(Fußgänger) getestet werden, da hier eine deutlich geringere Repräsentation im Datensatz erwartet wurde.

Um die Klassenimbalance über die angepasste Kostenfunktion hinaus zu berücksichtigen, fand das Training ausschließlich in Frames statt, bei denen mindestens ein Objekt der Klasse Fußgänger oder Fahrzeug erkannt wurde. Der Trainingsdatensatz reduzierte sich dadurch auf etwa 2000 Beispiele.

7.4.6. Selektion der Pixelmenge

Zur Selektion der Pixelmenge wird die Zielmaske $\boldsymbol{\Omega}_{\mathrm{GT}}(\mathbf{p})$ aus Unterunterabschnitt 7.3.2.3 verwendet. Das NN soll somit ausschließlich gegen Pixel trainiert werden, welche anhand der Daten der Referenzsensorik als mögliche Ziele identifiziert wurden. Die Maske dieser Ziele sei definiert als

$$\mathcal{SR} = \left\{ \mathbf{p} \in \mathcal{P}_{\mathrm{all}} \middle| \Omega_{\mathrm{GT}}(\mathbf{p}) > 0.5 \right\}. \tag{7.42}$$

7.4.7. Gesamtkosten

Die Gesamtkosten ergeben sich als Mittelwert der Abweichungen über alle Skalierungsebenen nach (Gleichung 7.41) und Pixeln zu

$$L_{\mathrm{M}} = \frac{\sum_{s=0}^{3} \sum_{p \in \mathcal{SR}} l_{\mathrm{M}}(u, v, s)}{\sum_{s=0}^{3} |\mathcal{SR}|}. \tag{7.43}$$

7.4.8. Initialisierung der Parameter

Die Initialisierung der Netzwerkparameter erfolgte analog zur Beschreibung aus Unterabschnitt 7.2.8.

7.4.9. Optimierer

Die Optimierung der Netzwerkparameter erfolgte analog zur Beschreibung aus Unterabschnitt 7.3.9.

7.4.10. Trainingsprozess

Der Verlauf der Kosten über den gesamten Trainingsprozess ist für alle Netzwerkkonfigurationen aus Tabelle 7.7 in Abbildung 7.19 dargestellt.

Zu erkennen ist, dass eine Reduktion der Kosten über die Trainingsbeispiele bei allen NNen vorhanden ist, ein Training im Grundsatz also erfolgreich war. Außerdem ist zu erkennen, dass sich die Trainingskosten für die unterschiedlichen Parameter stark unterscheiden, was durch die unterschiedliche Wahl der Skalierungsparameter α_{M} und γ zu erklären ist. Anhand der Trainingskosten kann nicht prognostiziert werden, welche

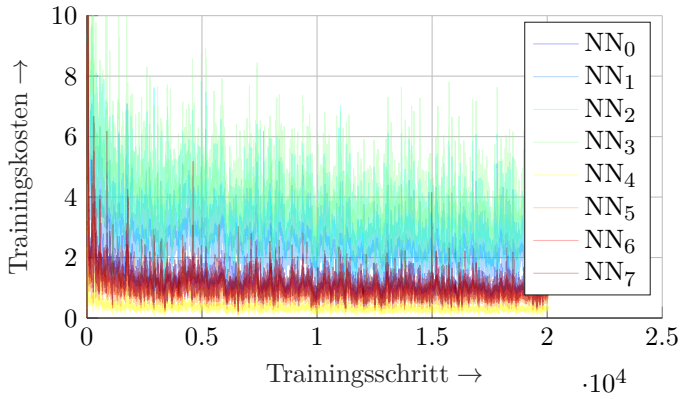


Abbildung 7.19.: **Verlauf des Trainingsprozesses der semantischen Segmentierung:** Die Kosten während des Trainings konvergieren bereits nach einigen Trainingsbeispielen.

NN Wahl die beste Genauigkeit bei der semantischen Segmentierung erreicht. Diese Analyse wird im nachfolgenden Abschnitt durchgeführt.

7.4.11. Ergebnis

7.4.11.1. Beschreibung der Metriken

Für die Bewertung der per-Pixel-Klassifikationsgenauigkeit bei der semantischen Segmentierung, wird die Mean-Intersection-over-Union (MIoU)-Metrik verwendet, welche u.a. in [GOO+17] dokumentiert ist und folgende Form hat

$$MIoU = \frac{1}{K} \sum_{i=1}^{K} \frac{p_{ii}}{\sum_{j=1}^{K} p_{ij} + \sum_{j=1}^{K} p_{ji} - p_{ii}}. \tag{7.44}$$

Darin ist K die Anzahl der unterschiedlichen Klassen, im vorliegenden Fall $K = |\{\text{Stationär, Fußgänger, Fahrzeug }\}| = 3$. Die Werte p_{ii} und p_{ij} zählen die Anzahl der Pixel, bei denen die Klasse des Labels und der Prädiktion identisch sind bzw. die Anzahl der Pixel, bei denen sich die Klassen unterscheiden. Der Index ij bedeutet, dass das Label der Klasse i entspricht, die Prädiktion der Klasse j.

Die MIoU nimmt Werte im Intervall $[0, 1]$ an, wobei höhere Werte einer besseren Klassifikation entsprechen.

Die MIoU mittelt über alle Klassen und fasst somit alle Datenpunkte in einer Metrik zusammen. Nachteilig dabei ist, dass keine klassenspezifische Aussage durchgeführt werden kann. Neben der MIoU wird deshalb auch noch die Konfusionsmatrix [Faw06] über die Klassen ermittelt.

7.4.11.2. Quantitative Auswertung

Die Metriken aller Frames des Testdatensatzes und NN-Parametrierungen wurden berechnet und sind in Abbildung 7.20 in Form von Histogrammen dargestellt. Jeder Eintrag im Histogramm entspricht einem Frame. Die MIoU wurde dabei also für alle Pixel im Kamerabild berechnet und die Verteilung der MIoU in allen Kameraframes

dargestellt. Da die Frames im Datensatz unterschiedliche Szenen abgebildet haben und die Klassifikationsgenauigkeit entsprechend beeinflusst wird, ergibt sich daraus eine Verteilung der MIoU.

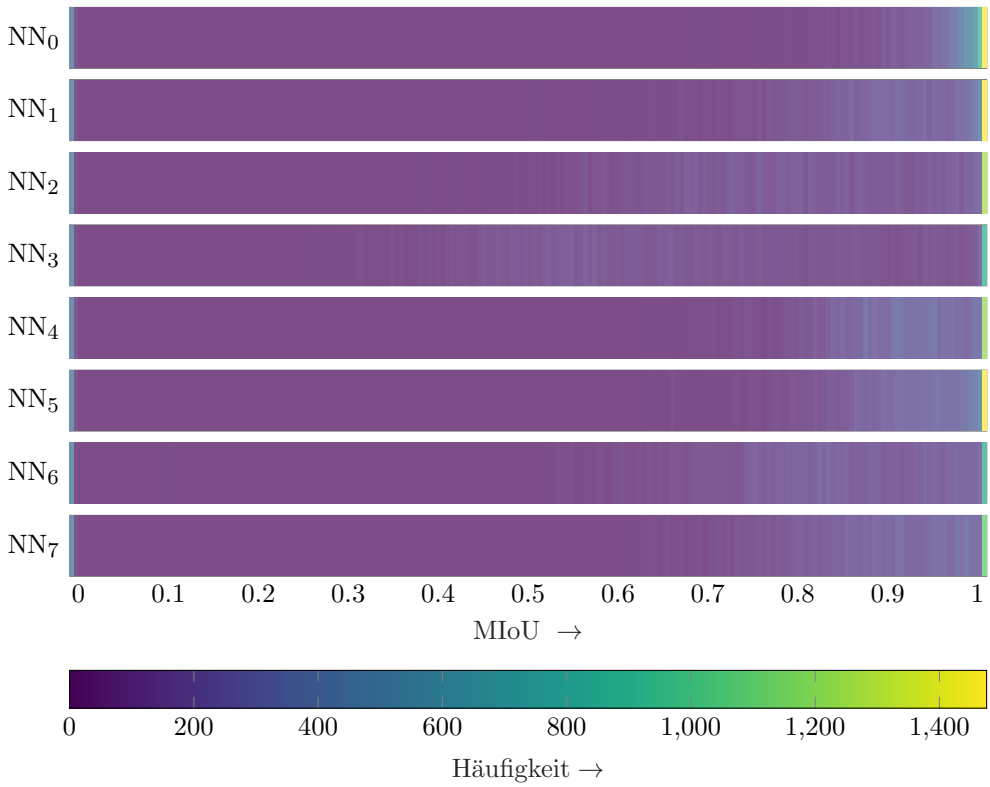


Abbildung 7.20.: **Klassifikationsgenauigkeit der semantischen Segmentierung:** Die MIoU wurde für alle Frames des Testdatensatzen und Netzwerkparametrieriserungen berechnet und dargestellt. Zu sehen ist mitunter eine hohe Fluktuation der Metrik über alle Frames.

Für alle NN-Parametrierungen sind hellere Farbwerte bzw. große Häufigkeiten bei größeren MIoU zu beobachten. Leicht zu beobachten sind Unterschiede der Farbverteilung zwischen den NN-Parametrierungen.

Die Konfusionmatrizen aller NN-Parametrierungen sind in Anh. A.3 dargestellt. Die Diagonalelemente der Konfusionsmatrizen sind in Tabelle 7.8 dargestellt und beschreiben die relative Häufigkeit korrekter Klassifikationen in Abhängigkeit der Klasse.

In Tabelle 7.8 ist zu sehen, dass die Klassifikationsrate für die Klasse „Stationär“ für alle NN Parametrierungen oberhalb von 90% liegt. Die Aktivierung dieser Klasse wird wesentlich durch die „skip-connection“ , zu sehen in Abbildung 7.18, definiert. Unterschiede in der Klassifikationsgenauigkeit für stationäre Ziele zwischen den Parametrierungen ergeben sich somit ausschließlich über die Aktivierungen der Klassen Fußgänger und Fahrzeug, welche die Netzwerkausgabe durch die Softmax-Aktivierungsfunktion skalieren. Die Klassifikationsrate für Fahrzeuge und Fußgänger fällt merklich schlechter aus. Durch Sichtung der Konfusionsmatrizen aus Anh. A.3 kann festgestellt werden, dass bei allen Parametrierungen Pixel vom Typ Fußgänger häufig (39.6 – 60.12%) als stationäre Pixel prädiziert wurden. Diese Art der Falschklassifikation ist durchaus wahrscheinlich

Tabelle 7.8.: **Übersicht der Klassifikationsraten und MIoU für die Parametersets**: Angaben in %

Name	Stationär	Fußgänger	Fahrzeug	MIoU
NN_0	97.66	32.30	74.97	99.89
NN_1	96.17	47.28	57.21	99.94
NN_2	94.19	59.57	59.86	99.00
NN_3	95.67	48.99	36.02	99.46
NN_4	96.16	49.28	47.31	99.82
NN_5	95.53	53.17	65.93	99.66
NN_6	91.36	53.00	66.42	86.43
NN_7	91.14	42.60	42.24	96.47

und plausibel, weil Fußgänger in der Regel eine langsame Geschwindigkeit aufweisen und somit über Doppler kaum von stationären Zielen getrennt werden können. Möglich ist, dass die separate Prozessierung durch **MTI** bzw. die Parametrierung hier nicht ausreicht, um eine bessere Klassifikationsrate für Fußgänger zu erreichen. Auch Fahrzeuge wurden teilweise falsch als stationär klassifiziert, eine größere Menge ($5.53 - 45.05\%$) der Fahrzeuge wurde jedoch falsch als Fußgänger klassifiziert. Für die drei Klassen konnten Klassifikationsraten, deutlich besser als Zufallsziehungen ($33.\overline{3}\%$)erreicht werden. Sinnvolle Änderungen, um die Klassifikationsrate zu verbessern, könnten beispielsweise sein, (a) die **MTI** Klassifikation zu optimieren oder wegzulassen und (b) die Klassifikation über ganze Sequenzen von RD-maps durchzuführen, so dass der Klassifikator zeitlich variierende Merkmale extrahieren kann[6]. Da das Augenmerk dieser Arbeit jedoch nicht auf der Optimierung der semantischen Segmentierung liegt, wird diese Untersuchung für mögliche Folgearbeiten aufgespart.

Anhand der hier ermittelten Klassifikationsraten kann keine unmittelbare Tendenz für die Optimierung mit Skalierungsebenen oder Fokussierungsparametern beobachtet werden. Für den qualitativen Vergleich wird das Parameterset NN_5 gewählt, da es vergleichsweise hohe Klassifikationsgenauigkeiten für alle Klassen aufweist.

7.4.11.3. Qualitative Auswertung

Eine repräsentative Übersicht der Prädiktionen für die Anwendung der semantischen Segmentierung ist in Abbildung 7.21 dargestellt. Wie zuvor beschrieben, wird bei der Prädiktion die Netzwerkausgabe von NN_5 verwendet. In der linken Spalte ist die semantische Maske $\boldsymbol{\Psi}_{\mathrm{M}}$ eingetragen, welche die Zielwerte für das NN-Training darstellt.

In der zweiten Spalte wurden die semantischen Masken $\mathrm{M}_{\mathrm{predict,\ cam}}$ dargestellt. Zur besseren Übersicht wurden Pixel, welche durch das NN zur Zieldetektion als Hintergrund detektiert wurden ($\mathrm{TD}_{\mathrm{predict,\ cam}}(\mathbf{p}) < 0.5$) ausgeblendet. In der letzten Spalte sind die Prädiktionen in der RD-map gezeigt. In allen Spalten wurden die Pixel entsprechend der Klassifikation, siehe Farblegende, eingefärbt.

In Abbildung 7.21 sind unterschiedliche Szenarien mit bewegten Objekten der Klasse Fußgänger und Fahrzeug dargestellt. In der oberen Zeile ist auf der linken Bildseite

[6]Der Benefit einer Zuführung zeitlicher Merkmale für die Klassifikation von Radarzielen wurde u.a. in [Heu13] und [Sau16] nachgewiesen

ein Fußgänger und in der Bildmitte ein Fahrzeug zu sehen. Diese wurden automatisch entsprechend in der Zielmaske eingefärbt. Das NN konnte ausschließlich das Fahrzeug korrekt klassifizieren. In der zweiten Zeile im linken Bildbereich sind zwei Fußgänger zu sehen. Im rechten Bildbereich zwei weitere, welche sich jedoch außerhalb des Radars FoV befinden. Die linken Fußgänger wurden folgerichtig automatisch gelabelt und durch das NN ebenfalls korrekt klassifiziert. In der dritten Zeile ist ein bewegtes Fahrzeug zu erkennen, welches durch das NN erfolgreich klassifiziert wurde. In der letzten Zeile ist ein Fußgänger zu erkennen, welcher ebenfalls erfolgreich klassifiziert wurde. In allen Szenarien wurden Pixel zu stationären Reflexionen zuverlässig eingefärbt. Einige Pixel wurden durch das NN fälschlicherweise als Fußgänger klassifiziert, obwohl die Pixel stationäre Objekte abbilden.

In Abbildung 7.21 wurden Pixel entsprechend der größten Netzwerkausgabe eingefärbt. Zur weiteren Analyse der Inferenz sind in Abbildung 7.22 die vollständigen Netzwerkausgaben der semantischen Segmentierung für die Klassen dargestellt. In der oberen Zeile sind Kamerabilder und RD-map dargestellt. In der zweiten Zeile sind die Netzwerkausgaben für die Klassen projiziert in das Kamerabild (Kamera 2) dargestellt. Es fällt auf, dass die Prädiktion von stationären Zielen, dem Fußgänger und einem fehlenden Fahrzeug plausibel ist. Besonders interessant ist die unterste Zeile. Dort sind die Netzwerkausgaben im RD-Gitter dargestellt. Die Aktivierung für stationäre Ziele wird primär durch die MTI außerhalb des NN festgelegt, siehe Unterabschnitt 7.4.3. Zu beobachten ist, dass Pixelregionen in der näheren Umgebung von stationären Klassifikationen als Fußgänger klassifiziert werden. Weiter entfernte Pixel werden dagegen ausschließlich als Fahrzeug klassifiziert. Das NN scheint entsprechend gelernt zu haben, dass Fußgänger wahrscheinlich niedrigere Geschwindigkeiten aufweisen als Fahrzeuge. Nach Ansicht des Autors ist so eine Annahme durchaus plausibel.

Weitere Beispiele der semantischen Segmentierung inklusive Szenenbeschreibung und Beobachtungen sind in Anh. A.4 zu finden.

7.4.11.4. Antwort zur wissenschaftlichen Hypothese

Nach Bewertung der Beobachtungen aus quantitativer und qualitativer Analyse kann gesagt werden, dass das Training eines NN zur semantischen Segmentierung von RD-maps gegen die automatisch ermittelten Zielwerte aus der Referenzsensorik in Kombination mit der zugeführten MTI deutlich bessere Klassifikationsraten als ein Zufallsklassifikator hervorgebracht hat. Die beobachteten Beispiele der Inferenz waren im Wesentlichen plausibel und attestierten ein erfolgreiches Training. Weitere Verbesserungsmöglichkeiten (bessere MTI, Klassifikation über zeitliche Sequenz) der Netzwerkarchitektur wurden vorgeschlagen, sind aber kein weiterer Bestandteil dieser Arbeit.

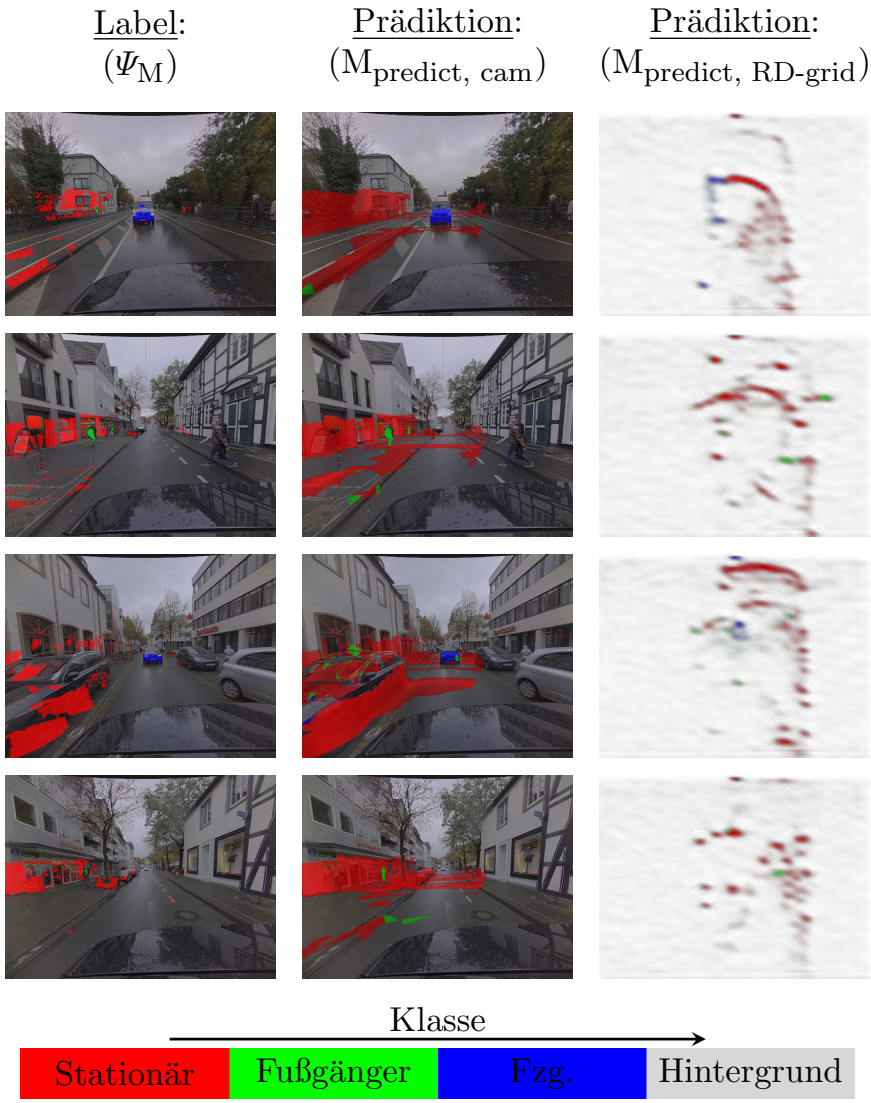


Abbildung 7.21.: **Beispiele der semantischen Segmentierung durch Radar**. Von links-nach-rechts: RGB Bild mit Zielwerten der semantischen Segmentierung, RGB Bild mit Prädiktionen der semantischen Segmentierung und RD-map mit Prädiktionen der semantischen Segmentierung. Alle Pixel wurden entsprechend der größten Klassenzugehörigkeit eingefärbt. Die Farblegende ist am unteren Bildrand zu erkennen.

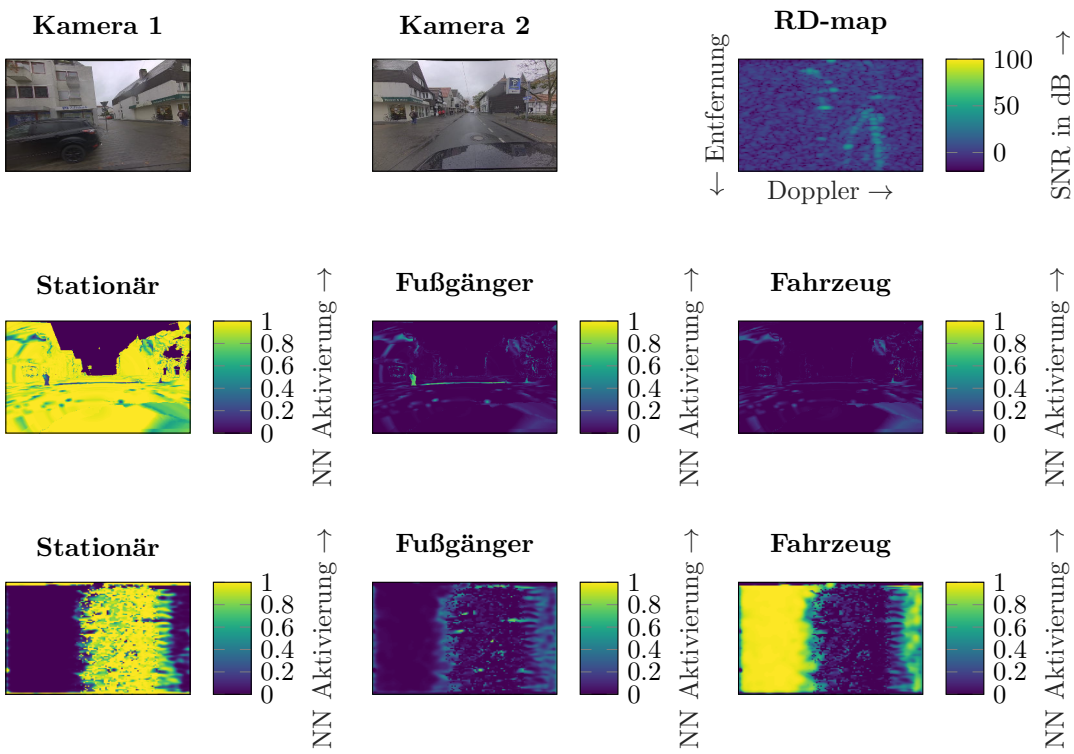
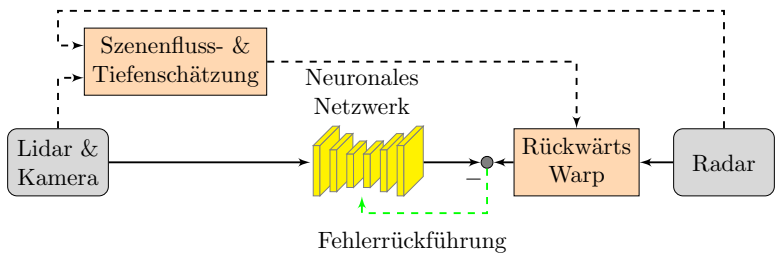


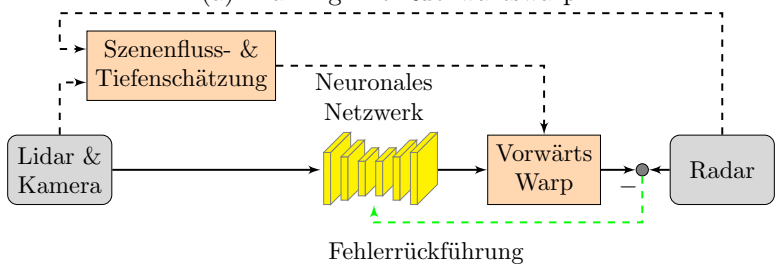
Abbildung 7.22.: **Beispiel der semantischen Segmentierung:** Oben: Die zwei Kamerabilder und die RD-map. Mitte: Netzwerkausgaben für die Klassen „Stationär“, „Fußgänger“ und „Fahrzeug“ gewarpt in das Kamerabild. Unten: Netzwerkausgaben im RD-Gitter.

7.5. Empfangsleistungsschätzung über das Kamerabild

In Abbildung 6.2 wurde die Pipeline zum überwachten Training einer Signalverarbeitung auf Radardaten vorgestellt. Diese wurde gegen Zielwerte kommend von einer Referenzsensorik trainiert. In den vorangegangenen Abschnitten wurde das Training an den Beispielen der Winkelschätzung, Zieldetektion und semantischen Segmentierung demonstriert. Dass es technisch möglich ist, auch eine Signalverarbeitung der Kameradaten gegen die Radardaten zu trainieren, wird in diesem Abschnitt gezeigt. Die allgemeine Struktur der Pipeline zum überwachten Training muss dafür, wie in Abbildung 7.23 gezeigt, nur leicht verändert werden. Das NN[7] prozessiert nun die Kamera- und Lidardaten. Die Radardaten werden über die Warpingschicht in das Pixelgitter der Kamera projiziert. Danach wird die Abweichung der NN-Ausgabe und der projizierten Radardaten ermittelt und das NN durch Fehlerrückführung („error backpropagation“) optimiert.



(a) Training mit Rückwärtswarp



(b) Training mit Vorwärtswarp

Abbildung 7.23.: **Übersicht der vorgestellten Systeme zum überwachtem Training von Signalverarbeitungen der Kamera- und Lidardaten:** Lidar und Kamera stellen Sensordaten für das NN bereit. Im NN wird für jedes Pixel eine Leistungsschätzung durchgeführt. Nach [EB7]. (a) Durch die Rückwärtswarp-Schicht, werden die Zielwerte in das Gitter des Kamerabildes gewarpt und dort mit den Prädiktionen verglichen. Abweichung werden unmittelbar in das NN (b) Durch die Vorwärtswarp Schicht, werden die Leistungswerte auf das Gitter des RD-maps gewarpt und dort mit der RD-map des Radars verglichen. Abweichungen werden gemessen und durch die Warping-Schicht zurück in das NN propagiert.

[7] Andere trainierbare Algorithmen sind hier nicht ausgeschlossen.

Motiviert wird dieses Vorgehen insbesondere auch aus dem Wunsch, die EM-Leistungsbeiträge der einzelnen Kamerapixel zu schätzen und für das Training und die Auswertung der bereits vorgestellten Verfahren zur Verfügung zu stellen. Eine Kurzübersicht des hier vorgestellten Verfahrens wurde in [EB7] veröffentlicht.

Ein wesentlicher Unterschied der beiden in Abbildung 7.23 gezeigten Strukturen ist, dass bei der oberen Struktur mögliche Fehler im Warping nicht durch das NN gelernt und kompensiert werden können. In Abschnitt 6.5 wurde als systematischer Fehler durch das Rückwärtswarp die mehrfache Zuordnung der Leistung eines RD-map-Pixels zu verschiedenen Kamerapixeln identifiziert. In diesem Abschnitt werden beide Strukturen trainiert und deren Prädiktionen gegenübergestellt. Dabei wird dieser systematische Fehler noch einmal offensichtlich gemacht.

7.5.1. Eingangsdaten

Als Eingangswerte zur Leistungsschätzung erhält das NN das Kamerabild $\mathbf{RGB}^0$, die Tiefenmaske $\mathbf{D}^0$, die semantische Instanzmaske $\mathbf{M}^0$ und Maske der geschätzten Aspektwinkel $\mathbf{N}^0$

$$\boldsymbol{\Gamma}_{\mathrm{Po}} = \begin{bmatrix} \mathbf{RGB}^0 \\ \mathbf{D}^0 \\ \mathbf{M}^0 \\ \mathbf{N}^0 \end{bmatrix}. \tag{7.45}$$

Kamerabild und Instanzmaske wurden ausgewählt, damit das NN selbst Informationen aus den optischen Daten extrahieren kann, die Tiefenmaske, weil die Empfangsleistung des Radars gemäß der Radargrundgleichung, siehe Gleichung 2.25, reziprok zur vierten Ordnung davon abhängt. Aspektwinkel wurden bereitgestellt, weil sie nach Gleichung 2.4 unter anderem die relativen Amplituden der Reflexion beeinflussen.

7.5.2. Zielwerte

Das NN soll versuchen, die Leistungswerte aus der RD-map **RD** des Radars zu schätzen. Die Zielwerte werden also direkt durch den Radarsensor erstellt und bereitgestellt.

7.5.3. Netzwerkarchitektur

Für die Leistungsschätzung wurde die Netzwerkarchitektur aus der Winkelschätzung, siehe Unterabschnitt 7.2.3, leicht adaptiert. Änderungen sind, dass nun vier Eingangskanäle bereitgestellt werden, siehe Unterabschnitt 7.5.1 und in der letzten Schicht keine Aktivierungsfunktion verwendet wird, da hier eine Regressionsaufgabe durchgeführt wird. Eine visuelle Übersicht der Netzwerkarchitektur ist in Abbildung 7.24 dargestellt.

Wir definieren die Übertragungsfunktion des NN zur Leistungsschätzung als

$$\mathrm{Po}_{\mathrm{predict,\ cam}} = \text{Po-Net}(\boldsymbol{\Gamma}_{\mathrm{P}}). \tag{7.46}$$

7.5.4. Assoziation von Prädiktion und Label durch Warping

In diesem Kapitel werden wir Untersuchungen von Vorwärts- und Rückwärtswarp aus Kapitel 6 machen. In beiden Fällen liegt die Netzwerkausgabe $\mathrm{Po}_{\mathrm{predict,\ cam}}$ im Gitter des Kamerabildes vor.

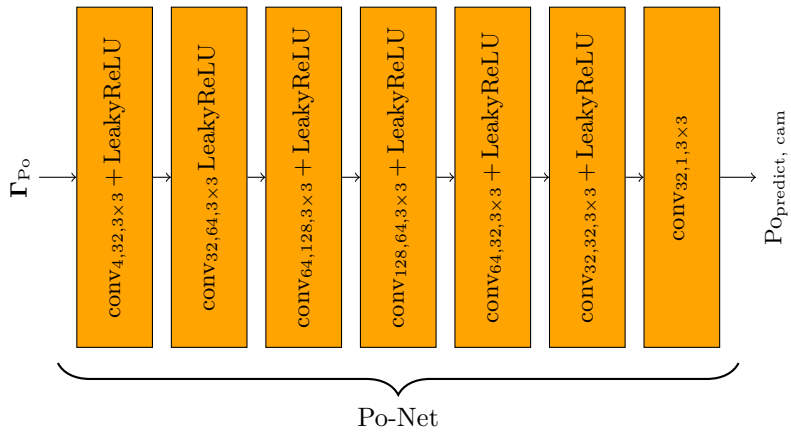


Abbildung 7.24.: **CNN Struktur für Leistungsschätzung:** Die Übersicht der Schichten. Nach [EB7].

7.5.4.1. Vorwärtswarp

Anders als bei den Anwendungen der Winkel-, Ziel- und Semantikschätzung findet die Inferenz hier nicht im Pixelgitter des RD-maps statt, sondern bereits im Gitter des Kamerabildes. Die Pixelwerte werden nun durch Vorwärtswarp in das Gitter des RD-maps gebracht. Die Transformation der Inferenz sei wie folgt durchgeführt:

$$\mathrm{Po}_{\mathrm{predict,\ RD}}(\mathbf{p}_{\mathrm{RD}}) = \eta_{\mathrm{FW}}\left(\mathrm{Po}_{\mathrm{predict,\ cam}}; v_r(\mathbf{p}), |\mathbf{x}_r(\mathbf{p})|\right), \tag{7.47}$$

wobei $\mathrm{Po}_{\mathrm{predict,\ RD}}$ die in das RD-Gitter gewarpten Schätzwerte sind.

7.5.4.2. Rückwärtswarp

Beim Rückwärtswarp bleiben die Werte der Inferenz im Pixelgitter der Kamera. Stattdessen werden die Zielwerte der Leistung vom RD-map in das Kamerabild gewarpt. Die Zielwerte wurden somit bereits in Kapitel 6 als $\mathbf{RD}_C$ definiert und entsprechende Beispiele gezeigt.

7.5.5. Messung der Abweichung

Zur Messung der Abweichung der Leistungsschätzung gegenüber Radar wird angenommen, dass sich die Reflexionen der Objekte nach „Swerling Typ 1“ [Mes06] überlagern. Dabei wird angenommen, dass Objekte durch eine Vielzahl kleinerer Reflektoren mit unterschiedlichem Abstand modelliert werden können. Durch die Überlagerung der Wellen der einzelnen Reflektoren ergibt sich eine Verteilungsdichte des RCS nach Chi-Quadrat mit DoF= 2 ($m = 1$) und dem mittleren RCS $\overline{\sigma}$:

$$p(\sigma|\overline{\sigma}) = \frac{1}{\overline{\sigma}} e^{-\frac{\sigma}{\overline{\sigma}}}. \tag{7.48}$$

Die Empfangsleistung des Radars und somit der Grauwert des RD-maps skaliert nach Gleichung 2.25 mit dem RCS. Wir bestimmen eine passende Kostenfunktion für die

Optimierung des Leistungsschätzers als Maximum-Likelihood-Schätzer für σ. Die Log-Likelihood Funktion zu Gleichung 7.48 ergibt sich zu

$$\log\left(L\left(\overline{\sigma}\right)\right) = \log\left(\prod_{i}^{N} \frac{1}{\overline{\sigma}} e^{-\frac{\sigma_i}{\overline{\sigma}}}\right) = N\log\left(\frac{1}{\overline{\sigma}}\right) - \frac{1}{\overline{\sigma}}\sum_{i}^{N}\sigma_i, \tag{7.49}$$

wobei hier die Verteilung einer Beobachtungsfolge eingesetzt wurde. Die Ableitung ergibt sich zu

$$\frac{\partial \log\left(L(\overline{\sigma})\right)}{\partial \overline{\sigma}} = -\frac{N}{\overline{\sigma}} + \frac{1}{\overline{\sigma}^2}\sum_{i}^{N}\sigma_i. \tag{7.50}$$

Der zugehörige Maximum-Likelihood-Schätzer kann durch Nullsetzen der Log-Likelihood Funktion hergeleitet werden zu

$$\frac{\partial \log\left(L(\overline{\sigma})\right)}{\partial \overline{\sigma}} = 0 = N\overline{\sigma} - \sum_{i}^{N}\sigma_i = \sum_{i}^{N}(\overline{\sigma} - \sigma_i) \Rightarrow \overline{\sigma}^{(ML)} = \frac{\sum \sigma_i}{N}. \tag{7.51}$$

In Gleichung 7.51 ist zu erkennen, dass zur Optimierung die Differenz aus gezogenem und beobachtetem RCS, σ_i und $\overline{\sigma}$, berechnet werden kann.

7.5.5.1. Vorwärtswarp

Für den Vorwärtswarp ergibt sich daraus eine mögliche Kostenfunktion nach

$$l_{Po,FW}(\mathbf{p}_{\mathrm{RD}}) = |\mathrm{Po}_{\mathrm{predict,\ RD}}(\mathbf{p}_{\mathrm{RD}}) - \mathbf{RD}_{[\mathbf{p}_{\mathrm{RD}}]}|. \tag{7.52}$$

Damit der „gradient descent “ Optimierer konvergiert, wurden hier Betragsklammern eingeführt.

7.5.5.2. Rückwärtswarp

Für den Rückwärtswarp ergibt sich daraus eine mögliche Kostenfunktion nach

$$l_{Po,BW}(\mathbf{p}) = |\mathrm{Po}_{\mathrm{predict,\ cam}}(\mathbf{p}) - \mathbf{RD}_{C,[\mathbf{p}]}|. \tag{7.53}$$

Wie beim Vorwärtswarp, wurden auch hier Betragsklammern verwendet.

7.5.6. Selektion der Pixelmenge

Wie für die anderen Trainingsanwendungen, wird beim Training eine Untermenge der Kamerapixel verwendet, um beispielsweise das FoV zwischen Kamera und Radar abzugleichen und detektierte Anomalien auszugrenzen.

7.5.6.1. Vorwärtswarp

Die Auswahl der Pixelmenge für die Optimierung wurde entsprechend Unterunterabschnitt 7.2.6.1 übernommen. Da die Kosten im Pixelgitter des RD-maps akkumuliert

werden, werden die validen Pixel via Vorwärtswarp auf das RD-Gitter projiziert:

$$\mathcal{P}_{\text{all, RD}}(\mathbf{p}_{\text{RD}}) = \eta_{\text{FW}}\Big(\mathcal{P}_{\text{all}}; v_r(\mathbf{p}), |\mathbf{x}_r(\mathbf{p})|\Big). \tag{7.54}$$

7.5.6.2. Rückwärtswarp

Die Auswahl der Pixelmenge für die Optimierung wurde entsprechend Unterunterabschnitt 7.2.6.1 übernommen.

7.5.7. Gesamtkosten

7.5.7.1. Vorwärtswarp

Durch Akkumulation der Abweichungen nach Gleichung 7.52 über die Pixelmenge $\mathcal{P}_{\text{all, RD}}$ ergeben sich die Gesamtkosten beim Vorwärtswarp zu

$$L_{Po,FW} = \frac{1}{|\mathcal{P}_{\text{all, RD}}|} \sum_{\mathbf{p}\in\mathcal{P}_{\text{all, RD}}} l_{Po,FW}(\mathbf{p}). \tag{7.55}$$

Beim Vorwärtswarp werden die Kosten über die validen Pixel im RD-Gitter ermittelt und akkumuliert.

7.5.7.2. Rückwärtswarp

Durch Akkumulation der Abweichungen nach Gleichung 7.53 über die Pixelmenge $\mathcal{P}_{\text{all}}$ ergeben sich die Gesamtkosten beim Vorwärtswarp zu

$$L_{Po,BW} = \frac{1}{|\mathcal{P}_{\text{all}}|} \sum_{\mathbf{p}\in\mathcal{P}_{\text{all}}} l_{Po,BW}(\mathbf{p}). \tag{7.56}$$

Beim Rückwärtswarp werden die Kosten über die validen Pixel im Kamerabild ermittelt und akkumuliert.

7.5.8. Initialisierung der Parameter

Die Initialisierung der Netzwerkparameter erfolgt analog zur Beschreibung aus Unterabschnitt 7.2.8.

7.5.9. Optimierer

Die Optimierung der Netzwerkparameter erfolgte analog zur Beschreibung aus Unterabschnitt 7.2.9.

7.5.10. Trainingsprozess

Der Verlauf der Kosten während des Trainings für Vorwärts- und Rückwärtswarp ist in Abbildung 7.25 dargestellt.

Zu erkennen ist, dass bei beiden Schätzern der Trainingsverlauf abflacht. Für den Schätzer mit Vorwärtswarp passiert die Reduktion der Trainingskosten zunächst deutlich

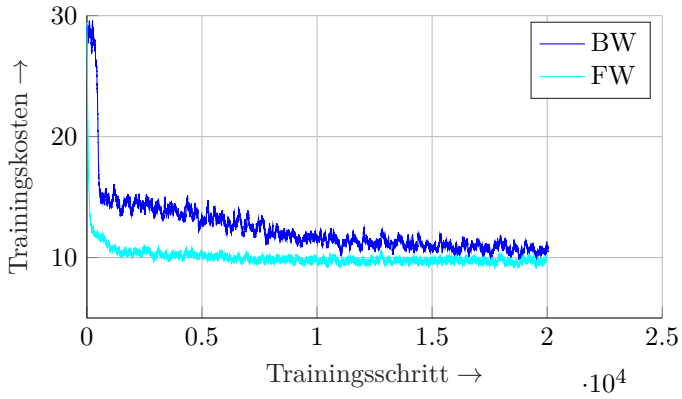


Abbildung 7.25.: **Verlauf des Trainingsprozesses der Leistungsschätzung im Kamerabild:** Für Rückwärtswarp („BW“) und Vorwärtswarp („FW“). Die Kosten während des Trainings konvergieren bereits nach einigen Trainingsbeispielen. Nach [EB7].

schneller als für den Schätzer mit Rückwärtswarp. Insgesamt scheinen die Schätzfehler beim Vorwärtswarp weniger Fluktuationen aufzuweisen als die vom Rückwärtswarp.

Wie zuvor beschrieben, wird erwartet, dass durch Rückwärtswarp die Leistung im Kamerabild überschätzt wird. Es werden je nach Szenario also systematische Fehler induziert, welche das NN-Lernen erschweren. Dies ist beim Vorwärtswarp nicht der Fall. Damit sind nach Ansicht des Autors die schnellere Konvergenz und die geringeren Fluktuationen beim Vorwärtswarp im Schätzfehler also plausibel.

7.5.11. Ergebnis

Nach dem Training der NNen werden nun die Schätzqualitäten im Testdatensatz ermittelt und bewertet.

7.5.11.1. Beschreibung der Metriken

Als Referenz für die Leistung stellt der Radar die RD-map bereit. Dieses fasst nach Abschnitt 2.1 die Empfangsleistung der Reflexionen aus der Szene zusammen. Dabei ist zu beachten, dass das FoV des Radars begrenzt ist und keine Reflexionen von außerhalb erwartet werden. Bedingt durch die unterschiedlichen FoVs von Kamera und Lidar muss folglich die Auswertung der Leistungsschätzung auf das FoV des Radars beschränkt werden. Bereits in Gleichung 7.55 wurden die Abweichungen der Leistung im RD-Gitter gemessen und über valide Pixel im FoV des Radars gemittelt. Für die quantitative Auswertung werden also analog die Schätzfehler für Vorwärts- und Rückwärtswarp nach Gleichung 7.55 im Testdatensatz ermittelt und können somit verglichen werden. Zu beachten ist, dass der Schätzer mit Vorwärtswarp bereits nach dieser Metrik optimiert wurde und folglich bessere Ergebnisse zu erwarten sind. Um einen Eindruck davon zu bekommen, ob die Schätzergebnisse biasfrei sind, werden ebenfalls die Residuen, wie Gleichung 7.55 jedoch ohne Betrag, ermittelt.

7.5.11.2. Quantitative Auswertung

In Abbildung 7.26 sind die erreichten Metriken von Vorwärts- und Rückwärtswarp für die Leistungsschätzung auf Kameradaten dargestellt. Die Metriken wurden für alle Bildindices des Testdatensatzes dargestellt, um einen Eindruck über die Fluktuation aller Testbeispiele zu ermöglichen.

In der oberen Zeile sind die mittleren Abweichungen dargestellt, in der unteren Zeile die mittleren absoluten Abweichungen. Zu beobachten ist, dass beim Rückwärtswarp die mittleren Abweichungen durchweg positiv sind, wohingegen beim Vorwärtswarp die Residuen um 0dB schwanken. Die Leistung wurde beim Rückwärtswarp also, wie erwartet, immer überschätzt.

Der Rückwärtswarp führt also definitiv nicht zu einem biasfreien Schätzer. Beim Vorwärtswarp ist kein offensichtlicher Bias zu beobachten.

Entsprechend fällt bei Beobachtung der Betragsabweichung auf, dass der Vorwärtswarp geringe Fluktuationen in der Testmetrik aufweist, ein Indikator dafür, dass das NN hier besser die Leistungsschätzung generalisieren kann.

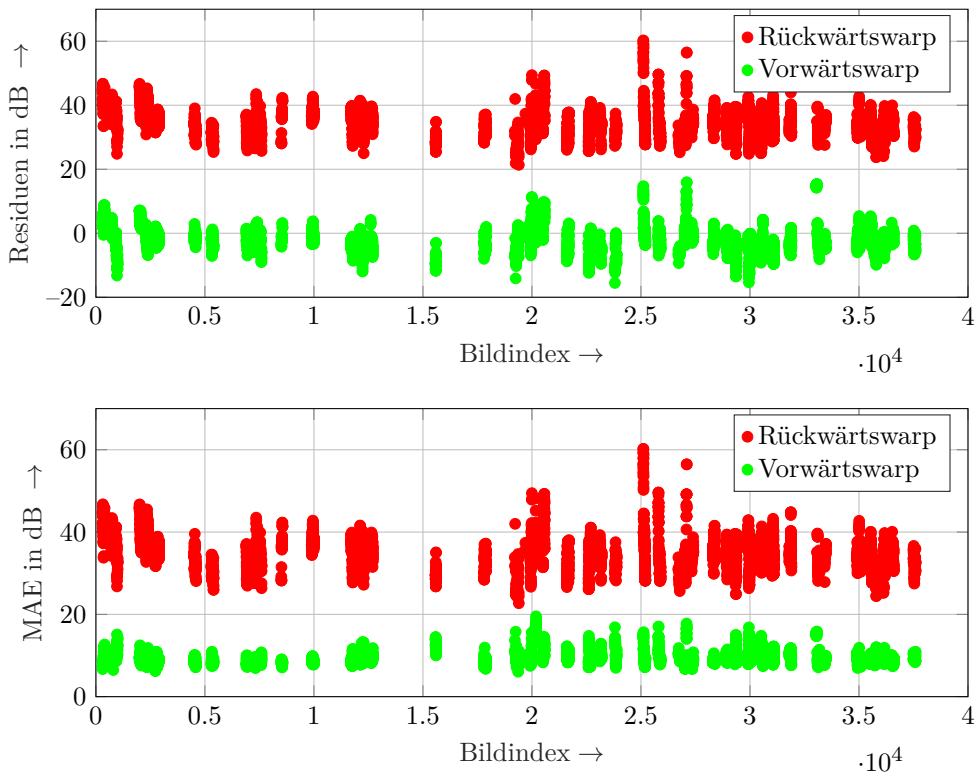


Abbildung 7.26.: **Metriken der Leistungsschätzung:** Oben: Residuen für Vorwärts- und Rückwärtswarp. Unten: Betragsabweichungen. Nach [EB7].

Die in Abbildung 7.26 dargestellten Metriken wurden außerdem gemittelt und in Tabelle 7.9 dargestellt.

In den ersten beiden Zeilen sind die Mittelwerte von Residuen und der Betrag der Residuen dargestellt. Wie oben beschrieben, hat der Rückwärtswarp einen empirischen Bias von knapp 35dB. Beim Vorwärtswarp beträgt der empirische Bias etwa –2.4dB. In den unteren beiden Zeilen sind die Varianz der Residuen und die Varianz der

Tabelle 7.9.: Metiken der Leistungsschätzung auf Testdatensatz

Metrik	Rückwärtswarp	Vorwärtswarp
$\overline{\text{Residues}}$	34.62	-2.36
$\overline{\lvert\text{Residues}\rvert}$	34.98	9.59
Var(Residues)	4.45	3.96
Var(|Residues|)	4.34	1.71

Betragsresiduen dargestellt. Wie oben beobachtet, weist Vorwärtswarp eine deutlich verringerte Varianz der Schätzgenauigkeit auf.

7.5.11.3. Qualitative Auswertung

Um einen Eindruck von der Leistungsprädiktion zu bekommen, werden wir uns nun ein paar ausgewählte Beispiele aus dem Testdatensatz ansehen. Bei der Selektion wurden Beispiele ausgewählt, in welchen nach Ansicht des Autors interessante Objekte bzw. Reflektoren abgebildet sind. Ein vollständiges Beispiel mit Kamerabild, RD-map, Masken zu den validen Pixeln, NN-Prädiktion der Leistung in Kamera und RD-Gitter sowie Abweichungen der Prädiktion gegenüber RD-map ist in Abbildung 7.27 dargestellt.

In der ersten Zeile sind Kamerabilder und RD-maps der Szene dargestellt. Darunter die validen Pixel $\mathcal{P}_{\text{all}}$ und $\mathcal{P}_{\text{all, RD}}$. Die Prädiktion der Leistung von Rückwärts- und Vorwärtswarp ist in Zeile drei zu sehen.

In der vierten, die nicht validen Pixel nach $\mathcal{P}_{\text{all}}$ maskiert.

In Zeile fünf sind die Prädiktionen nach dem Warp in das RD-Gitter dargestellt. In der letzten Zeile sind die Abweichungen der gewarpten Prädiktion gegenüber RD-map zu erkennen.

Bei der Szene handelt es sich um ein Parkplatzszenario mit ein paar parkenden Autos am linken Rand des Kamerabildes. Am rechten Rand sind komplexe Gebäudestrukturen zu erkennen. Die Prädiktionen der Leistung (Zeile drei) zwischen Rückwärts- und Vorwärtswarp unterscheiden sich ganz offensichtlich. Die Grauwerte beim Rückwärtswarp scheinen homogen und gefiltert zu sein. Bei Vorwärtswarp sind die Grauwerte heterogener verteilt und es sind schärfere Übergänge zwischen den Pixeln zu erkennen. Beim Vorwärtswarp erscheinen die vertikalen Pfosten des Zauns am rechten Bildrand deutlich intensiver als beim Rückwärtswarp.

Bei Betrachtung der Residuen aus der letzten Zeile fällt noch die Überschätzung der Leistung durch den Rückwärtswarp besonders auf. Beim Vorwärtswarp sind die Residuen mal positiv, mal negativ.

Weitere Beispiele der Inferenz sind in Abbildung 7.28 dargestellt. Um Platz einzusparen, wurden hier nur die Kamerabilder, RD-maps sowie die Prädiktionen dargestellt. Die Beispiele wurden manuell ausgewählt. Dabei wurden folgende Beobachtungen gemacht.

Beim Beispiel aus der ersten Zeile sind im linken Bildbereich Kettenpfosten zu erkennen. Insbesondere beim Verfahren nach Vorwärtswarp wurde dort eine erhöhte Reflexionsleistung prädiziert. Nach Ansicht des Autors ist das plausibel, da die Kettenpfosten aus Metall bestehen.

Im Beispiel aus der zweiten Zeile ist im linken Bildbereich die Spiegelung eines Fahrzeuges im Fenster eines Geschäfts zu sehen. Die Prädiktoren schätzen hier erhöhte

Empfangsleistung, was unplausibel ist.

Im Beispiel aus der dritten Zeile wurden bei beiden Prädiktoren die Stellen der Bordsteinkante prädiziert. Die Signatur der Bordsteinkante ist auch in der RD-map zu erkennen und daher plausibel.

Im Beispiel aus der vierten Zeile ist eine Leitplanke zu erkennen. In der RD-map ergibt sich insbesondere bei den Leitplankenpfosten erhöhte Leistung. Die Leitplanke an sich scheint aufgrund spiegelnder Reflexion nur geringe Leistung in der RD-map hervorzubringen. Insbesondere beim Prädiktor nach Vorwärtswarp scheint dieses Verhalten (Pfosten hohe Leistung, Leitplanke geringe Leistung) korrekt abgebildet zu werden.

Im Beispiel aus der fünften Zeile ist im linken Bildbereich ein Baum zu sehen. In der RD-map ist die Signatur des Baumes klar zu erkennen. Auch die Prädiktoren weisen dem Baum erhöhte Reflexionsleistung zu.

Im Beispiel aus der sechsten Zeile sind zwei Fahrzeuge dargestellt. In der RD-map sind entsprechend zwei Punkte mit erhöhter Leistung zu erkennen. Die Prädiktoren weisen den Fahrzeugen ebenfalls erhöhte Leistung zu.

Im Beispiel aus der siebten Zeile ist im linken Bildbereich ein Fahrzeug eines Glaserunternehmens abgebildet. Insbesondere das Verfahren nach Vorwärtswarp erkennt die feinen Strukturen der seitlichen Ladefläche und weist diesen, je nach Aspektwinkel, eine erhöhte Leistung zu. Unplausibel erscheint, dass im Bereich der Fahrerkabine deutlich höhere Leistung prädiziert wurde als im Bereich der Ladefläche.

Im Beispiel aus der achten Zeile ist ein Metallpfosten abgebildet. In der RD-map ist dieser durch eine punktförmige Signatur abgebildet. Beide Prädiktoren prädizieren an dieser Stelle eine erhöhte Leistung gegenüber der Umgebung. Auch dieses Szenario wurde plausibel prädiziert.

7.5.11.4. Antwort zur wissenschaftlichen Hypothese

In diesem Abschnitt wurde ein Schätzer der Empfangsleistung eines Radars basierend auf Kameradaten vorgestellt. Nach Wertung des Autors erscheinen die Prädiktionen der Leistung durchaus plausibel. Es wurde gezeigt, dass durch Verwendung von Rückwärtswarp eine systematische Überschätzung der Leistung entsteht, und diese wurde technisch begründet. Durch Verwendung von Vorwärtswarp wurde diese systematische Überschätzung eliminiert bzw. zumindest deutlich reduziert.

Bei der quantitativen Auswertung aus Unterunterabschnitt 7.5.11.2 wurde die Szene global ausgewertet. Die Metriken bilden also ein Mittel aus allen möglichen Objektklassen ab. Für zukünftige Auswertungen ist eine dezidierte Auswertung, bezogen auf einzelne Objektklassen, durchaus interessant, um mögliche systematische Fehler der Prädiktion aufzudecken.

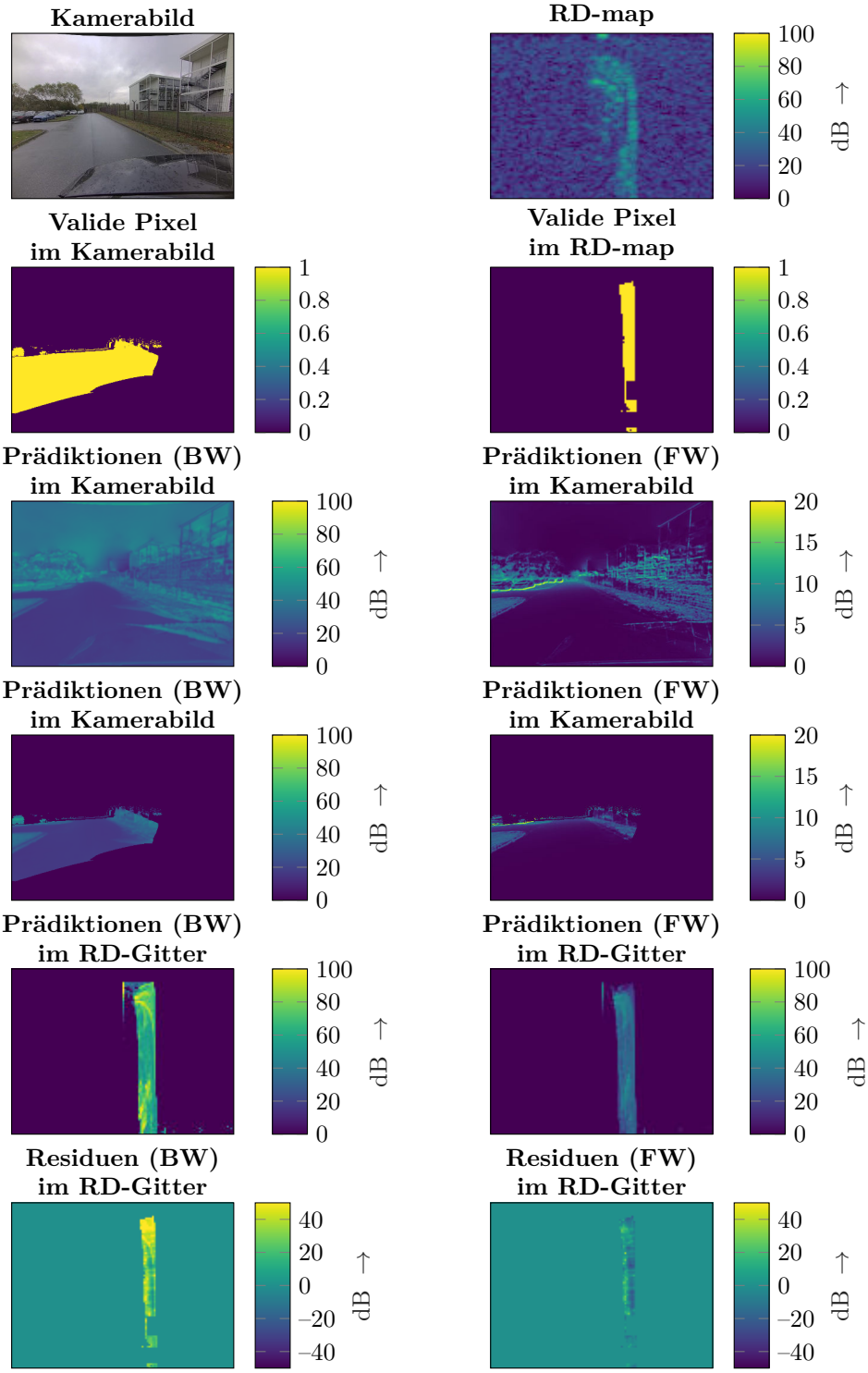


Abbildung 7.27.: **Beispiel der Leistungsschätzung:** 1. Zeile: Kamerabild und RD-map. 2. Zeile: Valide Pixel in Kamerabild und RD-map. 3. Zeile: Prädikation durch NN Trainiert via Rückwärts- bzw. Vorwärtswarp. 4. Zeile: Wie 3. Zeile, aber nur valide Pixel. 5. Zeile: Prädiktionen projiziert mittels Vorwärtswarp in das RD-Gitter. 6. Zeile: Abweichungen von Prädiktionen im RD-Gitter gegenüber RD-map. Nach [EB7].

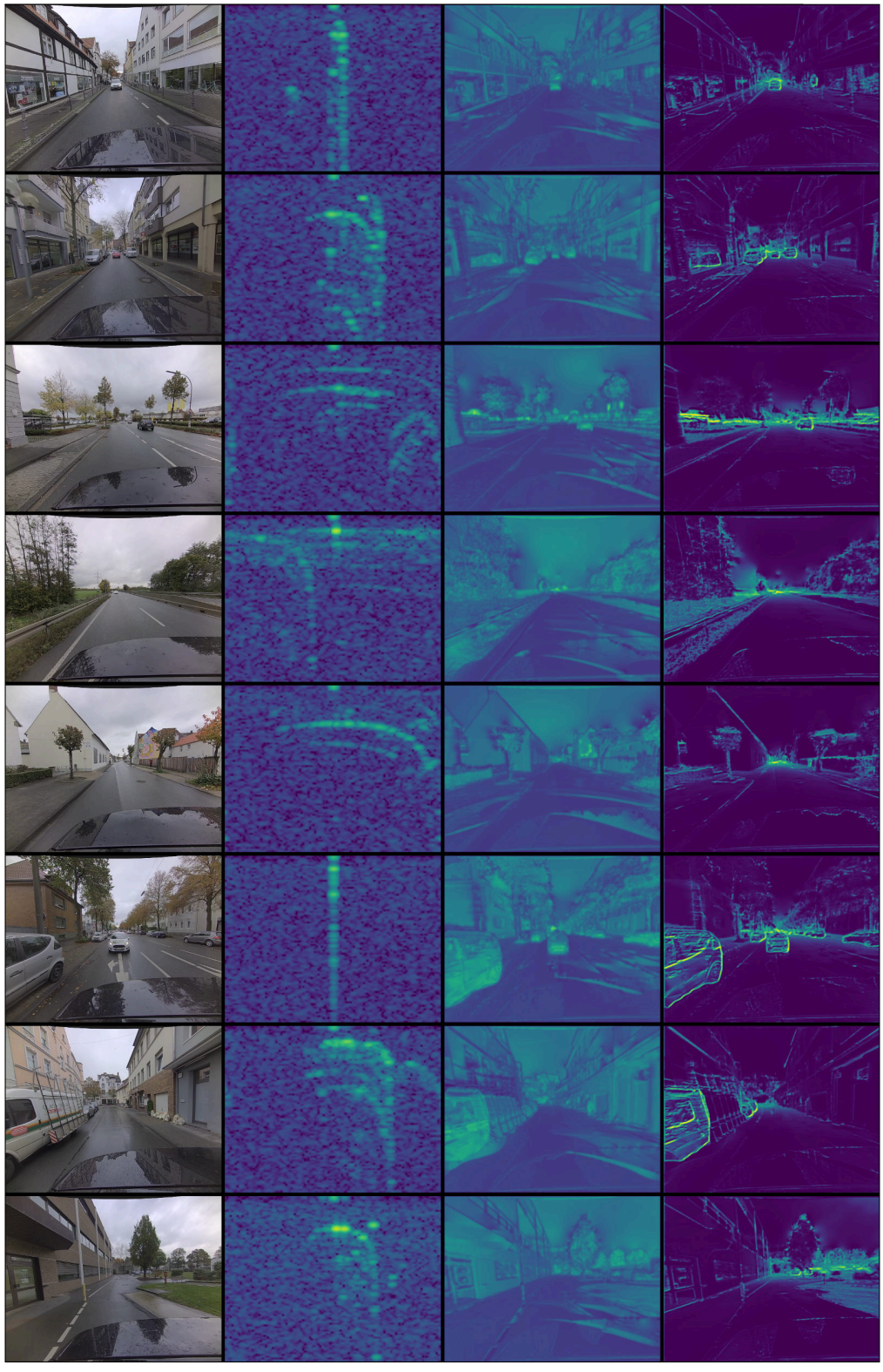


Abbildung 7.28.: **Weitere Beispiele der Leistungsschätzung:** Von links nach rechts: Kamerabild, RD-map, Prädiktion via Rückwärtswarp, Prädiktion via Vorwärtswarp. Verwendet wurde eine „viridis“ Farbcodierung wobei hellere Farbwerte eine höhere Empfangsleistung darstellen. Nach [EB7].

7.6. Zusammenfassung

In diesem Kapitel wurde das in dieser Arbeit vorgestellte Verfahren zum überwachten Training einer NN basierten Radarsignalverarbeitung mittels Warping auf bzw. von Kamerabildern validiert. Die Vielfältigkeit des Verfahrens wurde für die unterschiedlichen Anwendungen der Winkelschätzung, Zieldetektion und Klassifikation von Radardaten, sowie der Radarleistungsschätzung aus Kameradaten demonstriert. In allen Anwendungen konnten plausible Ergebnisse erzielt werden. In den Anwendungen der Winkelschätzung und Zieldetektion wurden die Leistungsfähigkeit der resultierenden Schätzer gegenüber klassischen Schätzern anhand von umfangreichen Echtweltaufnahmen quantifiziert.

8. Zusammenfassung und Ausblick

8.1. Zusammenfassung

Der Wunsch nach immer leistungsfähigeren Advanced-Driver-Assistance-Systems (ADAS) treibt die Entwicklung beteiligter Systeme wie automatische Perzeption, Bewegungsplanung und Aktorik immer weiter an. Für die Perzeption bieten sich eine Vielzahl verschiedener Sensoren an, die ein möglichst genaues Abbild der Fahrzeugumgebung schaffen und ein semantisch automatisiertes Verständnis der Szene ermöglichen. Einer der etablierten Sensortypen sind automotive Radarsysteme. Diese lassen sich aufgrund ihrer Bau- und Wirkweise versteckt hinter beispielsweise Stoßfänger oder Schweller integrieren, ohne dabei den Ausbreitungspfad der ausgesendeten und an der Umgebung reflektierten EM-Wellen zu blockieren. Der Sensor besteht neben der Hardware, welche beispielsweise Radom, Gehäuse, Antennendesign und Schaltkreise einschließt, auch aus Software. In dieser werden die Reflexionen der Umgebung ausgewertet. Gängige Algorithmen für diese Verarbeitung wurden in Abschnitt 2.1 vorgestellt.

Um dem Wunsch nach leistungsfähiger Perzeption nachzukommen, ist eine Weiterentwicklung der Algorithmen denkbar und wahrscheinlich sogar notwendig. Im Rahmen dieser Arbeit wurde untersucht, inwieweit sich NNe zur Umsetzung dieser Algorithmen eignen und welchen statistischen Leistungsvorteil diese erreichen können. Das Training von NNen nach dem Prinzip der Fehlerrückführung (engl.: „error-backpropagation“) erfordert dabei annotierte Trainingsdaten. Zur Annotation der Trainingsdaten in größerer Menge wurde zunächst ein Versuchsfahrzeug mit Referenzsensorik, bestehend aus Kamera, Lidar und DGPS-INS, aufgebaut (Kapitel 3). Die Daten der Referenzsensorik wurden fusioniert und verarbeitet, so dass der wesentliche Messraum des Radarsensors, bestehend aus den Zuständen Entfernung, Radialgeschwindigkeit und Einfallswinkel, abgedeckt werden konnte. Einen wesentlichen Beitrag stellt dabei die Szenenflussschätzung aus Kapitel 5 dar, bei welcher durch Fusion aller Sensoren eine verbesserte und am Radar ausgerichtete Geschwindigkeitsschätzung der Umgebung realisiert werden konnte.

In Kapitel 6 wurde gezeigt, wie Radardaten aus einem RAD-Gitter mittels differenzierbarer Warpingschicht in Kamerabilder projiziert werden können. Analog wurde eine Warpingschicht demonstriert, bei welcher Daten aus dem Pixelgitter der Kamera in das RAD-Gitter des Radars projiziert werden können. Durch die Projektion zwischen den Pixelgittern von Kamera und Radar und der damit verbundenen Assoziation der Pixel lassen sich die automatisiert generierten Annotationen der Kamera nutzen, um ein überwachtes Training einer NN-basierten Signalverarbeitung zu ermöglichen. In Kapitel 7 wurde dies ausgiebig an Beispielen wie einer Winkelschätzung, einer Zieldetektion und semantischen Segmentierung von RD-Spektren demonstriert. Es wurde gezeigt, dass die NN-basierte Signalverarbeitung mindestens ebenbürtige Schätzgenauigkeiten gegenüber klassischen Signalverarbeitungsverfahren unter realistischen Umgebungsszenarien erreichen kann.

Die Assoziation von Radar- und Kamerapixeln erlaubt darüber hinaus das Trainieren

einer NN-Signalverarbeitung der Kameradaten mit Hilfe der Annotationen aus dem RAD-Gitter. Dies wurde am Beispiel einer Leistungsschätzung demonstriert, bei welcher ein NN die Radarempfangsleistung aus den Kameradaten prädiziert. Die Ergebnisse sind vielversprechend und in Summe biasfrei, wenn auch nicht immer plausibel.

Der CUDA und PyTorch basierte Code für den Vorwärtswarp wurde unter `https://github.com/ChrGri/Forward-Warp` veröffentlicht.

8.2. Ausblick

Insbesondere bei der Prädiktion der Radarempfangsleistung aus Kamerabildern ergibt sich nach Empfinden des Autors großes Entwicklungspotential. Die Leistungsschätzung könnte tiefere Einblicke in die Wahrnehmung durch den Radar ermöglichen und bei der Auswertung zum Beispiel bei der Simulation von anderen Radarkonfigurationen nützlich sein. Ebenfalls nützlich könnte die Leistungsschätzung für die Validierung von Radarperzeptionsalgorithmen sein, da sie einen Einblick in das liefert, was für den Radar potenziell sichtbar oder unsichtbar ist.

Beim vorgestellten Leistungsschätzer wurde eine kohärente Summation der reellwertigen Kamerapixel durchgeführt. Es ist denkbar, hier noch die Phasenlagen der Wellen zu schätzen. Außerdem könnte eine Raytracing-Schicht entwickelt werden, welche Mehrwegereflexionen nachbildet. Durch beides könnte das zu schätzende Verhalten durch teilweise „white box engineering“ stückweise vorgegeben und das Schätzproblem des NN so vereinfacht werden.

A. Anhang

A.1. Untersuchungen zum Umgang mit Labelnoise durch Optimierung im Skalierungsraum

Beim Warping der Netzwerkprädiktionen vom RD-Gitter in das Kamerabild kann es durch fehlerhafte Schätzung der Warpingparameter (Entfernung, Geschwindigkeit und Winkel) zu einer fehlerhaften Positionierung im Kamerabild kommen. Durch die falsche Positionierung im Kamerabild werden die Netzwerkprädiktionen im Trainingsprozess mit falschen Zielwerten verglichen, im Allgemeinen auch „Label-noise“ genannt, wodurch eine Degradation des Trainingsprozesses möglich ist. Es wird daher zusätzlich untersucht, ob es vorteilhaft ist, die Auflösung des Radars virtuell zu reduzieren, um somit die Anforderung an die Warpingparameter zu reduzieren und den Einfluss fehlerhafter Warpingparameter auf den Trainingsprozess zu reduzieren.

Um das Auflösungsvermögen des Radars virtuell zu reduzieren, wird auf die Netzwerkprädiktion im RD-Gitter ein 2D-Mittelwertfilter angewendet. Zwar bleibt damit die Dimension des RD-Gitters konstant, jedoch werden die Netzwerkprädiktionen in umliegende Pixelnachbarschaften übertragen und bilden mit der Pixelnachbarschaft eine Linearkombination, welche den Wert der gefilterten Prädiktion bestimmt. Nun wird nicht die ursprüngliche Prädiktion in das Kamerabild gewarpt, siehe Unterabschnitt 7.2.4, sondern die mittelwertgefilterte Prädiktion. Führen fehlerhafte Warpingparameter beim Training nun dazu, dass nicht das korrekte Pixel aus dem RD-Gitter in das Kamerabild gewarpt wird, sondern eins aus seiner Pixelnachbarschaft, so steckt in dieser Pixelnachbarschaft auch die Prädiktion der korrekt gewarpten Pixels. Wird im Trainingsprozess die Pixelabweichung für ein Kamerapixel berechnet, so hat das Parameterupdate für alle RD-Gitter-Pixel aus der Pixelnachbarschaft Folge.

Durch die Mittelwertfilterung der Prädiktion werden leider auch Prädiktionen von RD-Gitter-Pixel trainiert, welche möglicherweise gar nicht zum Kamerapixel passen. Um diese ungewollten Trainingspfade zu blockieren bzw. abzuschwächen, wird kein klassisches Mittelwertfilter verwendet, sondern eine gewichtete Mittelung anhand der Leistung aus der RD-map. Die Wichtung berücksichtigt die Leistung im RD-Gitter, so dass Pixel mit höherer Intensität größeren Einfluss haben:

$$\phi_{\text{prediction, scaled}}(u, v, s) = \frac{\sum_{u_s=-s}^{s} \sum_{v_s=-s}^{s} \phi_{\text{prediction}}(u+u_s, v+v_s)(\mathbf{RD}_{[u+u_s, v+v_s]} - \mathbf{RD}_{\min})}{(2s+1)^2 \sum_{u_s=-s}^{s} \sum_{v_s=-s}^{s} (\mathbf{RD}_{[u+u_s, v+v_s]} - \mathbf{RD}_{\min})}. \quad \text{(A.1)}$$

Hier entspricht $\mathbf{RD}_{[(u,v]} - \mathbf{RD}_{\min}$ der Leistungsdifferenz des Pixels und der global minimalen Leistung, so dass ausschließlich positive Gewichtungen auftreten. Der Parameter s bestimmt die Fenstergröße der Faltung, welche zu $2s+1$ festgelegt wurde.

Die Prädiktion $\phi_{\text{prediction, scaled}}(u, v, s)$ soll sich somit automatisch an die Position der Leistungssignatur in der RD-map angleichen und mögliche Fehler im Bereich der Größe des Faltungskerns in der Positionierung ausgleichen. Das Ergebnis dieser Skalierung ist in Abbildung A.1 an simulierten Daten für $s = 0$ und $s = 2$ dargestellt.

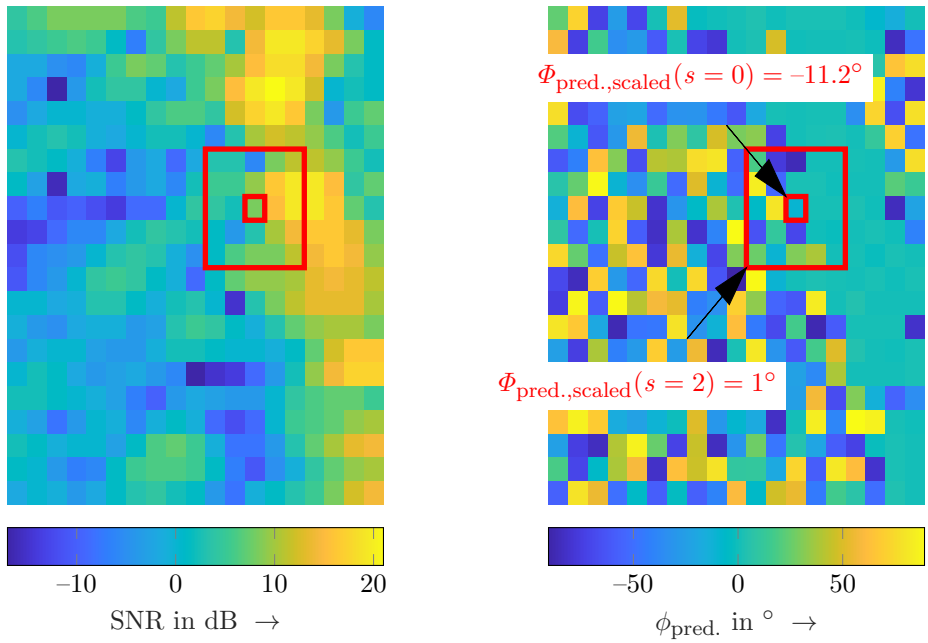


Abbildung A.1.: **Beispiel für die Prädiktion im Skalierungsraum:** Links: Ausschnitt einer simulierten RD-maps. Rechts: Winkelschätzungen für jedes Pixel. In roten Kästen dargestellt, sind die nach Gleichung A.1 gemittelten Winkelschätzungen für $s = 0$ und $s = 2$. Der Zielwert des Winkels für die Signalanteile beträgt 0°. Für das Hintergrundrauschen wurde der Winkel aus einer Gleichverteilung gezogen. Zu erkennen ist, dass bei $s = 2$ der prädizierte Winkel nur um 1°, statt zuvor 11.2° vom Zielwert 0° abweicht.

Eine fehlerhafte Geschwindigkeitsschätzung der Referenzsensorik wurde simuliert, in der nicht ein lokales Maximum selektiert wurde, sondern ein horizontal dazu versetztes Pixel, hervorgehoben durch das kleine rot dargestellte Kästchen. Als Zielwinkel für alle Ziele wurde 0° eingestellt und der Winkel aller Pixel mit einem zusätzlichen Winkel aus einer Gleichverteilung beaufschlagt, um überlagertes Rauschverhalten nachzustellen. Nun wurde der gemittelte Winkel für $s = 0$ (kleine rote Kästchen) und $s = 2$ (größere rote Kästchen) gemäß Gleichung A.1 berechnet und entsprechend in die Abbildung eingetragen. Zu erkennen ist, dass der gemittelte Winkel für $s = 0$ etwa –11.2° entspricht, was gegenüber dem Zielwert von 0° einer ebenso großen Abweichung entspricht. Der gemittelte Winkel bei $s = 2$ entspricht dagegen 1°, was einer deutlich geringeren Abweichung entspricht. Durch die geringe Abweichung zum Zielwert wird eine geringere Anpassung der Netzwerkparameter während des Trainings erzwungen. Dieser Effekt ist erwünscht, da die Abweichung zum Zielwert hier primär durch fehlerhafte Assoziation von Zielwerten und Prädiktionen zwischen Kamerabild und RD-Gitter entstanden ist.

Da die optimale Wahl von s nicht bekannt ist, werden mehrere gemittelte Netz-

werkprädiktionen aus dem Wertebereich $S = [0, 1, 2]$ parallel ermittelt, für jede die Abweichung gegenüber den Zielwerten ermittelt und wiederum gewichtet gemittelt.

Bei Anwendung des Skalierungsparameters s sind die gewarpte Prädiktion, die Pixelkosten und die Gesamtkosten folgendermaßen zu adaptieren:

$$\phi_{\text{predict, cam}}(\mathbf{p}, s) = \eta_{\text{BW}}\Big(\phi_{\text{prediction, scaled}}(s); v_r(\mathbf{p}), |\mathbf{x}_r(\mathbf{p})|\Big). \tag{A.2}$$

$$l_{\text{DOA}}(\mathbf{p}, s) = \sqrt{(\phi_{\text{label}}(\mathbf{p}) - \phi_{\text{predict, cam}}(\mathbf{p}, s)^2 + 10^{-6}}. \tag{A.3}$$

$$l_{\text{DOA, all}} = \sum_{s=0}^{S} \frac{1}{|\mathcal{P}_{\text{all}} \cap \mathcal{P}_{\text{train}}|} \sum_{\mathbf{p} \in \mathcal{P}_{\text{all}} \cap \mathcal{P}_{\text{train}}} l_{\text{DOA}}(\mathbf{p}, s). \tag{A.4}$$

Es wird nun untersucht, wie sich das Training im Skalierungsraum nach dem oben genannten Vorgehen auf die Qualität der Inferenz für Winkelschätzung auswirkt. Dazu werden die in Tabelle A.1 spezifizierten NN Parametrierungen trainiert. Es werden also zwei Netwerkarchitekturen ($1\times$ und 3×3) jeweils mit oder ohne diese Pixelselektion getestet.

Tabelle A.1.: **NN Architekturen und Konfigurationen mit Skalierungsräumen der Prädiktionen**.

Name	Faltungskern	Schichtmodifizierer (t)	Skalierungsebenen
NN_2	1×1	3	0
NN_3	1×1	3	2
NN_6	3×3	1	0
NN_7	3×3	1	2

A.1.1. Trainingsprozess

Analog zu Unterabschnitt 7.2.10 wurde da Training für die NN nach Tabelle A.1 durchgeführt. Der Verlauf der Trainingskosten ist in Abbildung A.2 gezeigt.

Auch hier ist eine Konvergenz der Schätzer nach wenigen Beispielen zu beobachten.

A.1.1.1. Quantitative Auswertung

Analog zu Unterunterabschnitt 7.2.11.3 ist eine quantitative Auswertung vorgenommen worden. Das erreichte MAE der Schätzer ist in Abbildung A.3 über das SNR aufgetragen.

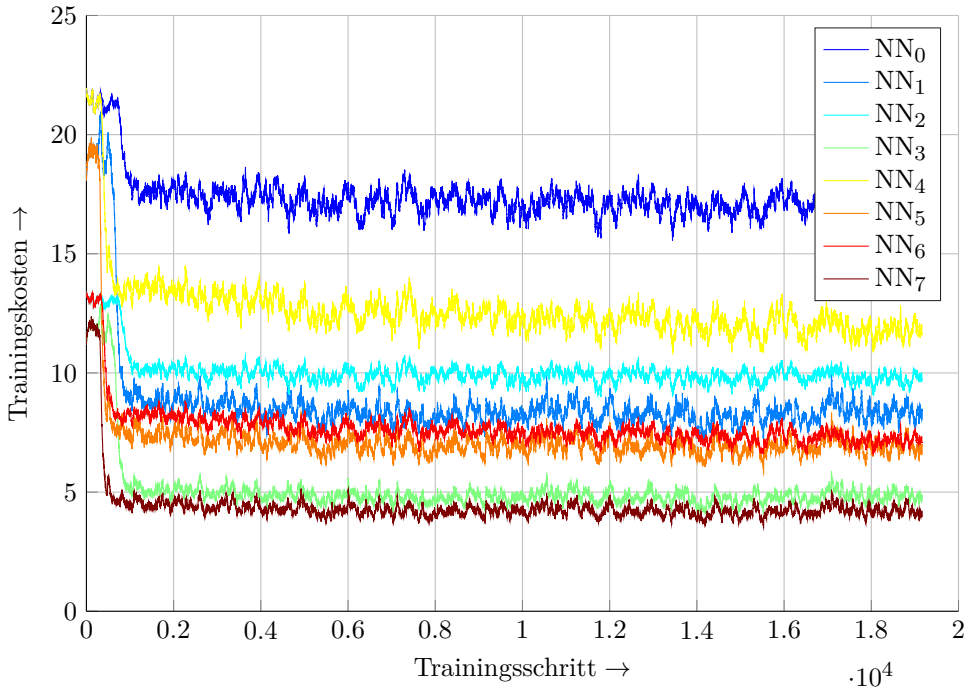


Abbildung A.2.: **Verlauf des Trainingsprozesses der Winkelnetzwerke:** Die Kosten während des Trainings konvergieren bereits nach einigen Trainingsbeispielen.

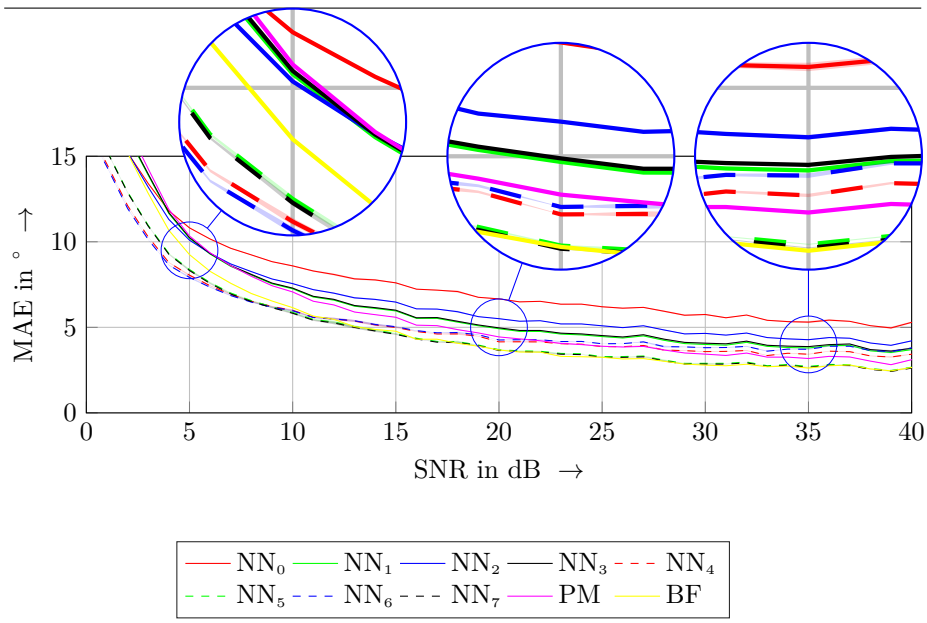


Abbildung A.3.: **Qualitätsmaße der Winkelschätzer über SNR:** Die mittlere absolute Abweichung der Winkelschätzer über SNR. Vergrößert dargestellt sind jeweils ein Bereich niedrigen SNRs, mittleren SNRs und hohen SNRs.

Wie zuvor auch, ist die bessere Schätzgenauigkeit bei steigendem SNR zu beobachten. Zur besseren Übersicht wurden die Schätzer an drei Stützstellen entsprechend erreichtem MAE hierarchisch in Tabelle A.2 aufgelistet.

Tabelle A.2.: **Hierarchische Einordnung der Schätzer entsprechend erreichter MAE**.

	SNR 5 dB	SNR 20 dB	SNR 35 dB
MAE →	NN_0	NN_0	NN_0
	PM	NN_2	NN_2
	NN_3	NN_3	NN_3
	NN_1	NN_1	NN_1
	NN_2	PM	NN_6
	BF	NN_6	NN_4
	NN_5	NN_4	PM
	NN_7	NN_5	NN_5
	NN_4	BF	NN_7
	NN_6	NN_7	BF

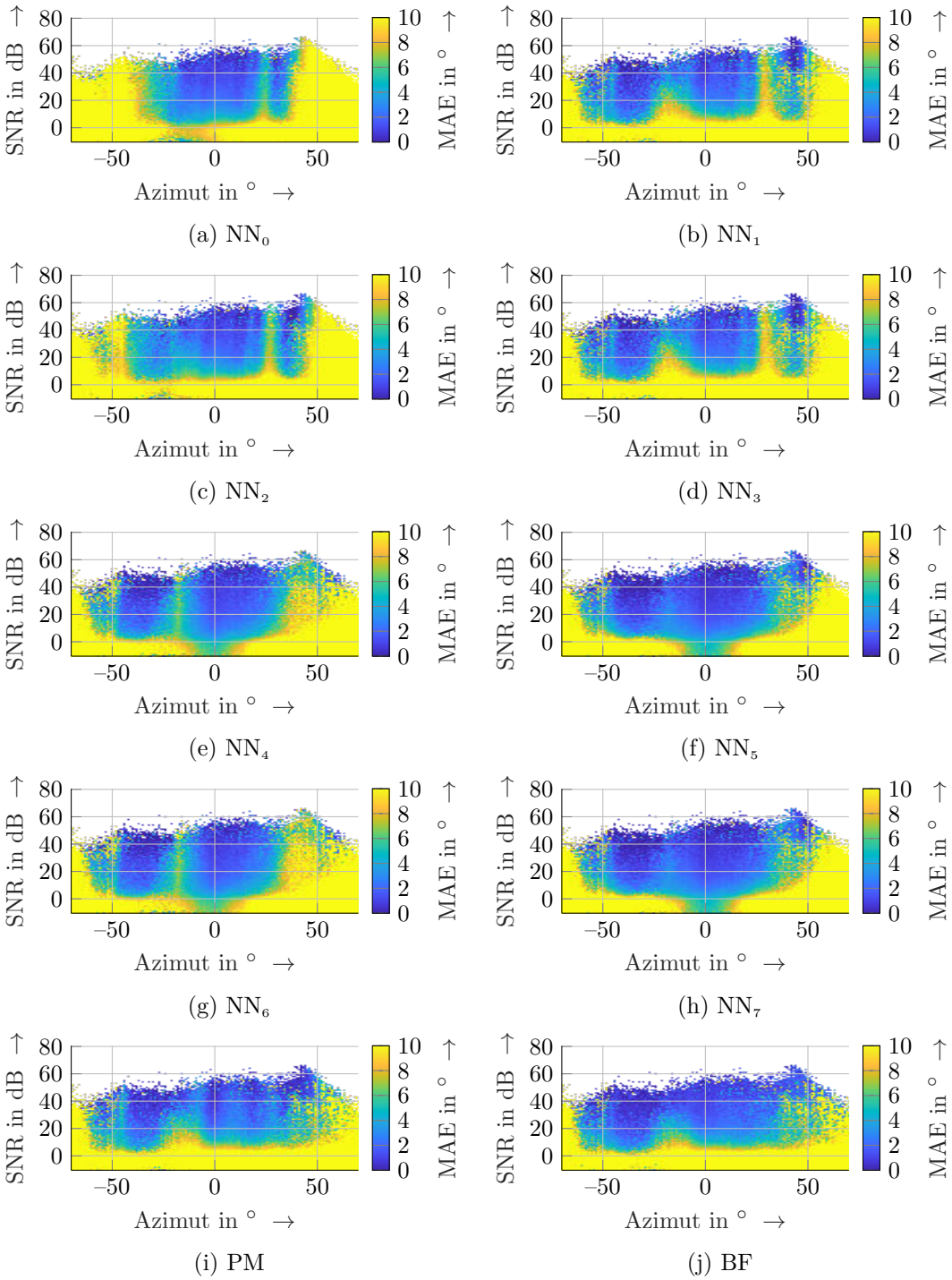
Zur Untersuchung des Einflusses mit der Optimierung der NN im Skalierungsraum sind gemäß Tabelle A.2 die NN Paare NN_0 vs. NN_2, NN_1 vs. NN_3, NN_4 vs. NN_6 und NN_5 vs. NN_7 zu vergleichen. In Tabelle 7.3 zeichnet sich kein klarer Unterschied zwischen NNen mit oder ohne Skalierungsraumoptimierung ab. Bei den über den größten Teil des SNR-Bereichs am besten verhaltenden NNen, NN_5 und NN_7, handelt es sich um ein oberes Paar mit und ohne Skalierungsraumoptimierung. Das Paar erreicht durchweg ein nahezu identisches MAE. Es wird geschlussfolgert, dass die hier implementierte Art der Skalierungsraumoptimierung keinen signifikanten Einfluss auf die Qualität der Winkelschätzer beim verwendeten Datensatz hat.

Um auch hier eventuelle Winkelinhomogenitäten aufzudecken, wurde in Abbildung A.4 das erreichte MAE über Einfallswinkel und SNR aufgetragen.

Für die NN mit Optimierung im Skalierungsraum (NN_2, NN_3, NN_6 und NN_7) sehen die Abbildungen unwesentlich anders aus gegenüber den Abbildungen der regulären NN (NN_0, NN_1, NN_4 und NN_5).

A.1.1.2. Zusammenfassung

Bei der quantitativen Analyse konnten keine signifikanten Unterschiede durch die Anwendung der Optimierung mit Skalierungsebenen beobachtet werden. Es wird daher geschlussfolgert, dass die Optimierung mit Skalierungsebenen hier keinen signifikanten Einfluss auf die Qualität der Winkelschätzgenauigkeit hat.



(a) NN_0 (b) NN_1

(c) NN_2 (d) NN_3

(e) NN_4 (f) NN_5

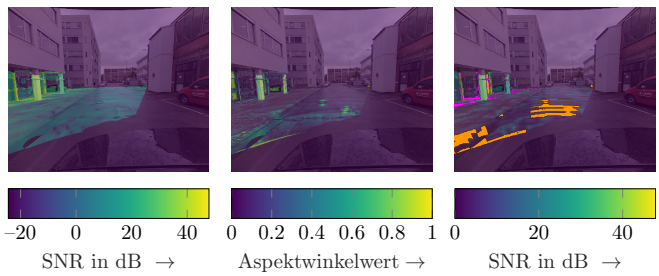
(g) NN_6 (h) NN_7

(i) PM (j) BF

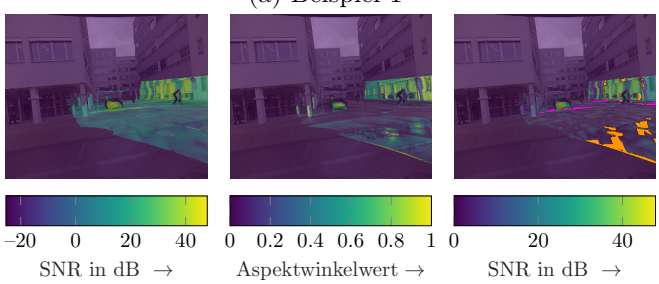
Abbildung A.4.: **2D MAE Histogramme:** Darstellung der Abhängigkeit von MAE gegenüber von SNR und Einfallswinkel.

A.2. Gegenüberstellung von Aspektwinkel und SNR für Zieldetektion

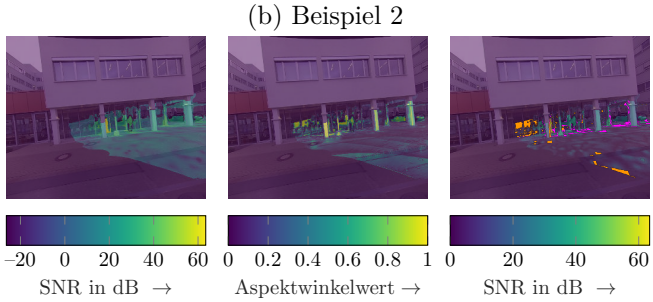
In Unterunterabschnitt 7.3.2.3 wurde eine statistische Untersuchung von Aspektwinkel und SNR durchgeführt. Es wurden automatisch Bildregionen markiert, welche dem eingeführten Modell widersprechen. In Abbildung A.5 sind weitere Beispielframes dargestellt.



(a) Beispiel 1



(b) Beispiel 2



(c) Beispiel 3

Abbildung A.5.: **Gegenüberstellung von Aspektwinkel und SNR:** Links: Projiziertes SNR; Mitte: Aspektwinkel-Wert; Rechts: Projiziertes SNR und farblich markiert (pink, orange) Abweichungen von der Modellbeschreibung.

A.2.1. Gegenüberstellung NN und CFAR Prädiktion für Zieldetektion

Um die Unterschiede zwischen den Detektoren weiter zu analysieren, wurden die Binärmasken[1] der Zieldetektoren miteinander verglichen, um so etwaige Unterschiede untereinander ausfindig zu machen. Für diesen Vergleich wurden die Binärmasken aus

[1] Als Binärmaske wird die Klassifikation der Pixel der RD-map bezeichnet

allen Frames mittels des aus der Bildverarbeitung bekannten Verfahrens Structural-Similarity-Index (SSIM) nach [WBSS04] in Bezug gebracht. Beim SSIM werden die Leuchtdichte, der Kontrast und die Struktur der Bilder miteinander verglichen und in Form eines Skalars im Intervall $[0, 1]$ angegeben. Höhere Werte werden bei sich stärker ähnelnden Bildern erzeugt. Eine tiefere Beschreibung der Metrik würde den Rahmen dieser Arbeit sprengen. Der interessierte Leser sei deshalb auf oben genannte Literatur verwiesen.

Sicherlich sind auch andere Metriken zum Vergleich der Bildinhalte verwendbar. Aufgrund der Popularität wurde sich für SSIM entschieden. Die SSIM-Metrik ist für sämtliche Bildindizes aus dem Testdatensatz in Abbildung A.6 dargestellt.

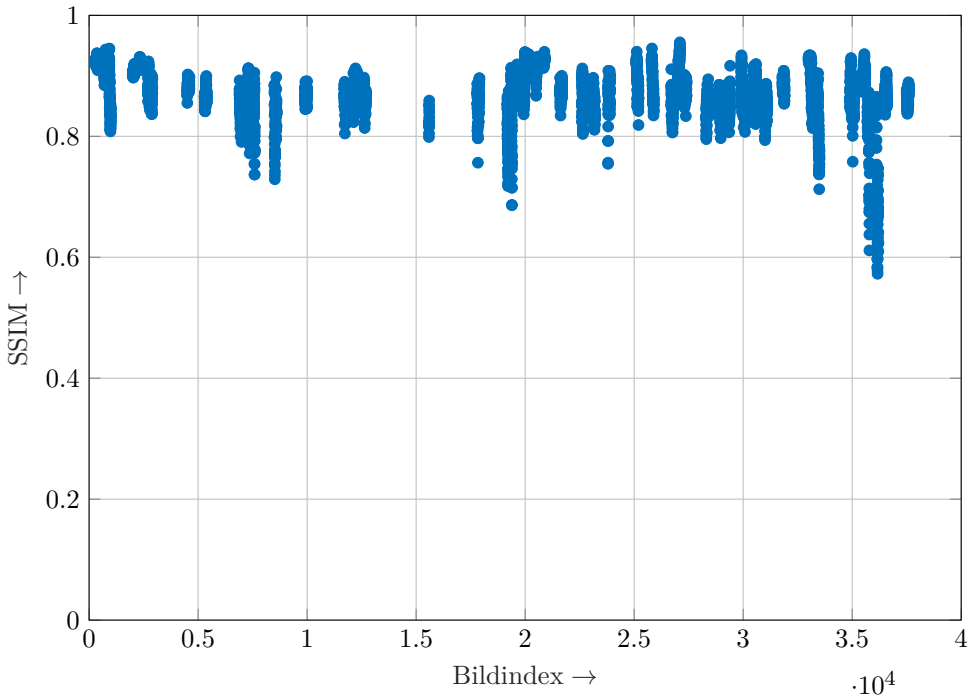
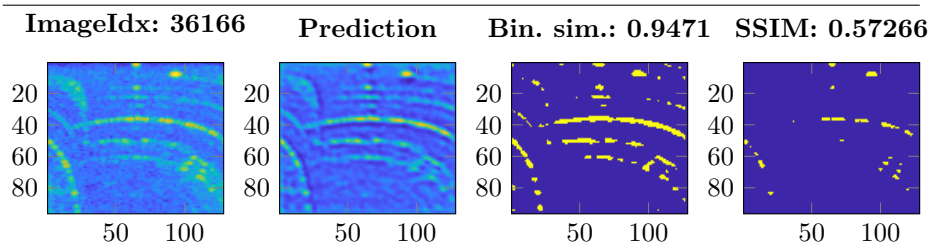


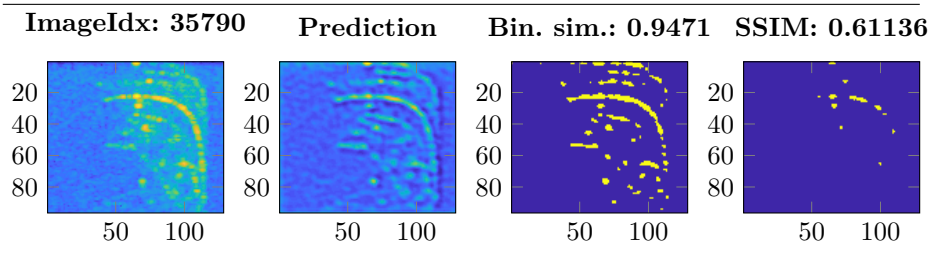
Abbildung A.6.: **Structual Similarity der Detektionen aus CFAR und NN:** Zum automatischen Identifikation der sich am stärksten und am schwächsten ähnelnden Detektionen, wurde der SSIM für alle Frames berechnet.

In Abbildung A.6 ist zu erkennen, dass der SSIM zwischen NN-basierter Detektion und CFAR-Detektion hier immer Werte oberhalb von 0.5 annimmt und größtenteils oberhalb von 0.8 liegt. Die SSIM-Werte ermöglichen nun das Sortieren der Binärmasken entsprechend dieser automatisch ermittelten Ähnlichkeit und somit das Auffinden von Beispielen, bei denen CFAR und NN besonders ähnliche oder unterschiedliche Detektionen erreicht haben. Anhand des SSIM wurden automatisch die fünf Frames mit der geringsten Übereinstimmung ausgewählt und in Abbildung A.7 dargestellt. Die fünf Frames mit der größten Übereinstimmung wurden analog in Abbildung A.8 dargestellt. Bei den Frames mit der größten Übereinstimmung ist, wie zu erwarten, nur ein geringer optischer Unterschied zwischen NN und CFAR-Prädiktion auszumachen. Aus Sicht des Autors ist es hier nicht möglich, eines der beiden Verfahren zu präferieren. In den Beispielen mit der geringsten Übereinstimmung dagegen scheint die Prädiktion

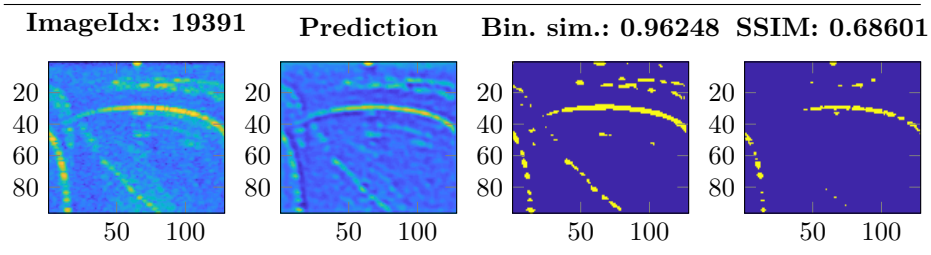
des NN eine bessere Segmentierung der RD-maps durchgeführt zu haben. Der Detektor schlägt in Bereichen erhöhter Leistung häufiger an als der CFAR-Detektor, erzeugt aber keine offensichtlichen Falschdetektionen. Nach Meinung des Autors könnte dieser Unterschied durch eine statistische Optimierung des CFAR-Detektors angepasst bzw. minimiert werden. Aus den Beispielen kann jedoch geschlossen werden, dass der NN-basierte Detektor plausible Entscheidungen trifft und das Training mittels Aspektwinkel erfolgreich durchgeführt werden konnte.



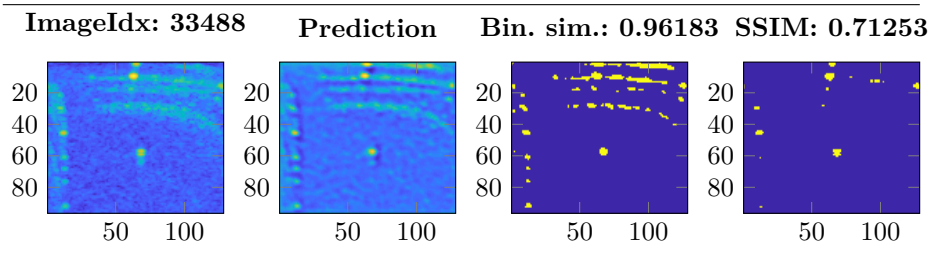
(a) Beispiel 1



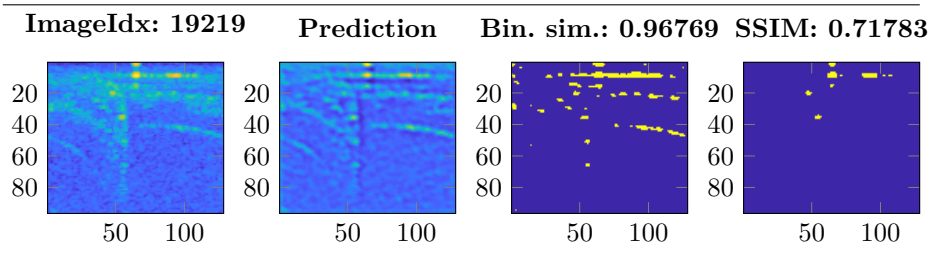
(b) Beispiel 2



(c) Beispiel 3

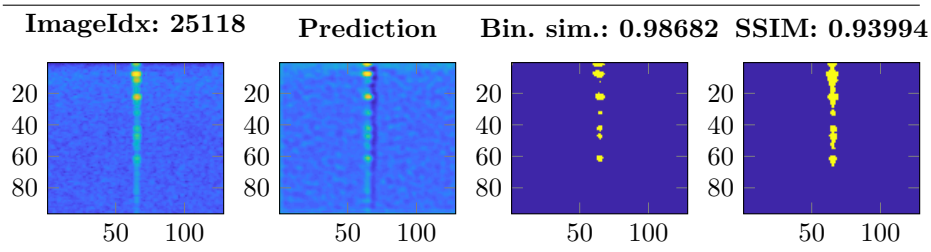


(d) Beispiel 4

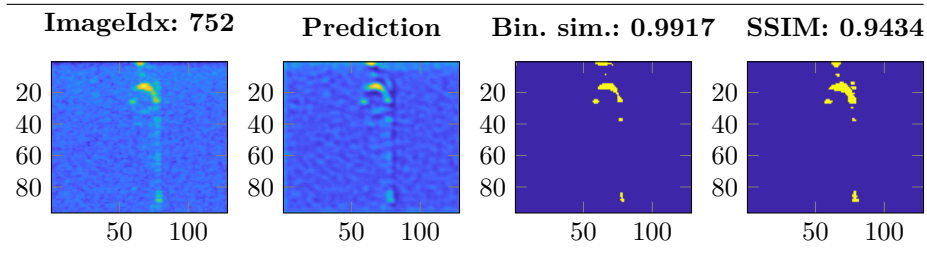


(e) Beispiel 5

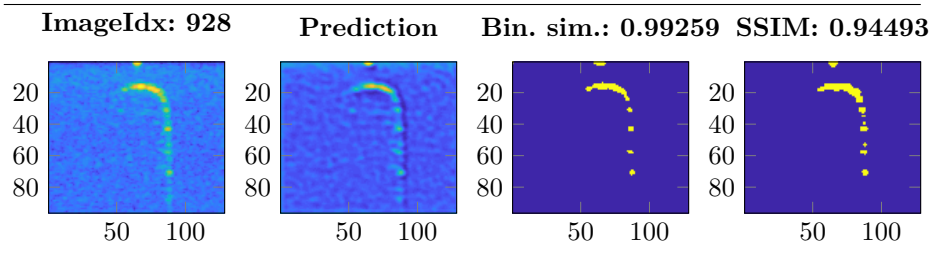
Abbildung A.7.: **Optischer Vergleich der Detektoren (niedrige Übereinstimmung):** Detektionen mit größter SSIM Abweichung. Von links nach rechts: RD-map, NN Prädiktion $\mathrm{TD}_{\text{predict, RD-grid}}$, $\mathrm{TD}_{\text{predict, RD-grid}} > 0.5$, CFAR Prädiktion



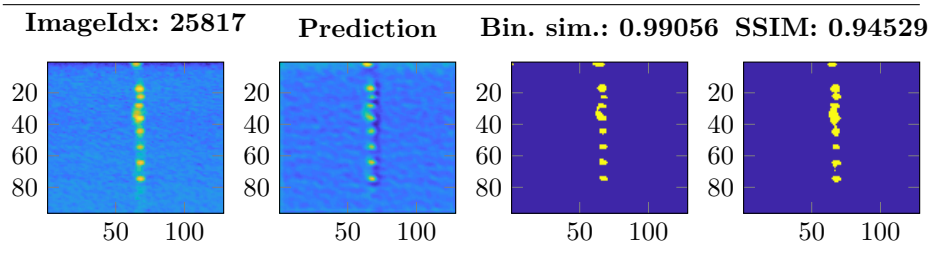
(a) Beispiel 1



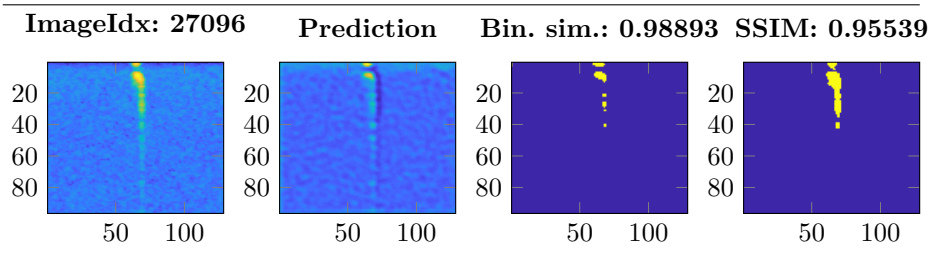
(b) Beispiel 2



(c) Beispiel 3



(d) Beispiel 4



(e) Beispiel 5

Abbildung A.8.: **Optischer Vergleich der Detektoren (hohe Übereinstimmung):** Detektionen mit größter SSIM Ähnlichkeit. Von links nach rechts: RD-map, NN Prädiktion $\mathrm{TD}_{\text{predict, RD-grid}}$, $\mathrm{TD}_{\text{predict, RD-grid}} > 0.5$, CFAR Prädiktion

A.3. Konfusionsmatritzen

Bei der Bewertung der semantischen Segmentierung wurden die Klassifikationsraten der Schätzer herangezogen. Die Klassifikationsrate ist die Wahrscheinlichkeit, mit welcher die Klasse der Objekte korrekt geschätzt wurde. Da neben der korrekten Klassifikation auch die Analyse der Falschklassifikationen interessant sein mag, sind in Tabelle A.3 - A.10 die Konfusionsmatrizen der Schätzer dargestellt.

In den Konfusionsmatrizen sind zeilenweise die tatsächlichen Klassen der Objekte gesammelt, spaltenweise die Prädiktionen zu diesen Klassen. In der ersten Zeile sind somit Objekte vom Typ „Stationär“ gesammelt und die Wahrscheinlichkeiten, mit welchen diese entsprechend als „Stationär“, „Fußgänger“ oder „Fahrzeug“ klassifiziert wurden.

Zu erkennen ist, dass je nach Parametersatz die Klassifikationsraten der Schätzer für die jeweiligen Klassen beeinflusst werden. Zum Beispiel wurde für Paremeterset 4 nur eine Klassifikationsrate für Fahrzeuge von etwa 36% erreicht. Im Vergleich dazu wurde bei Parameterset 1 eine Klassifikationsrate von etwa 75% erreicht. Die Wahl der Parameter hat also einen erheblichen Einfluss auf Klassifikation und kann somit als Werkzeug für das Tuning der selbigen verstanden werden. Die Wahl der Tuningparameter hängt vom Fokus der Echtweltanwendung ab und wird deshalb im Rahmen dieser Arbeit nicht weiter spezifiziert.

Tabelle A.3.: **Konfusionsmatrix für semantische Segmentierung**: Angaben in %, Parameterset 1

		Prädiktion		
		Stationär	Fussgänger	Fahrzeug
Wahre Klasse	Stationär	97.66	2.11	0.23
	Fußgänger	60.12	32.2	7.69
	Fahrzeug	19.5	5.53	74.97

Tabelle A.4.: **Konfusionsmatrix für semantische Segmentierung**: Angaben in %, Parameterset 2

		Prädiktion		
		Stationär	Fussgänger	Fahrzeug
Wahre Klasse	Stationär	96.17	3.8	0.02
	Fußgänger	50.38	47.28	2.33
	Fahrzeug	19.09	23.7	57.21

Tabelle A.5.: **Konfusionsmatrix für semantische Segmentierung**: Angaben in %, Parameterset 3

		Prädiktion		
		Stationär	Fußgänger	Fahrzeug
Wahre Klasse	Stationär	94.19	5.77	0.04
	Fußgänger	39.6	59.57	0.82
	Fahrzeug	16.98	23.15	59.86

Tabelle A.6.: **Konfusionsmatrix für semantische Segmentierung**: Angaben in %, Parameterset 4

		Prädiktion		
		Stationär	Fußgänger	Fahrzeug
Wahre Klasse	Stationär	95.67	4.3	0.03
	Fußgänger	50.52	48.99	0.48
	Fahrzeug	18.93	45.05	36.02

Tabelle A.7.: **Konfusionsmatrix für semantische Segmentierung**: Angaben in %, Parameterset 5

		Prädiktion		
		Stationär	Fußgänger	Fahrzeug
Wahre Klasse	Stationär	96.16	3.82	0.02
	Fußgänger	50.64	49.28	0.08
	Fahrzeug	19.25	33.44	47.31

Tabelle A.8.: **Konfusionsmatrix für semantische Segmentierung**: Angaben in %, Parameterset 6

		Prädiktion		
		Stationär	Fußgänger	Fahrzeug
Wahre Klasse	Stationär	95.53	4.39	0.08
	Fußgänger	43.92	53.17	2.91
	Fahrzeug	16.5	17.57	65.93

Tabelle A.9.: **Konfusionsmatrix für semantische Segmentierung**: Angaben in %, Parameterset 7

		Prädiktion		
		Stationär	Fußgänger	Fahrzeug
Wahre Klasse	Stationär	91.36	8.6	0.05
	Fußgänger	40.43	53	6.57
	Fahrzeug	17.18	16.39	66.42

Tabelle A.10.: **Konfusionsmatrix für semantische Segmentierung**: Angaben in %, Parameterset 8

		Prädiktion		
		Stationär	Fußgänger	Fahrzeug
Wahre Klasse	Stationär	91.14	8.82	0.04
	Fußgänger	43.51	56.37	0.12
	Fahrzeug	15.16	42.6	42.24

A.4. Beispiele der semantischen Segmentierung

Nachfolgend sind einige zusätzliche Beispiele der semantischen Segmentierung dargestellt. In Abbildung A.9 ist ein Szenario mit bewegtem Ego-Fahrzeug zu sehen. Das Ego-Fahrzeug wird von einem Kleintransporter verfolgt. Dieser wird korrekt durch das NN als bewegtes Fahrzeug klassifiziert. Der gehende Fußgänger im linken Bildbereich von Kamera 2 wurde fälschlicherweise als stationär klassifiziert.

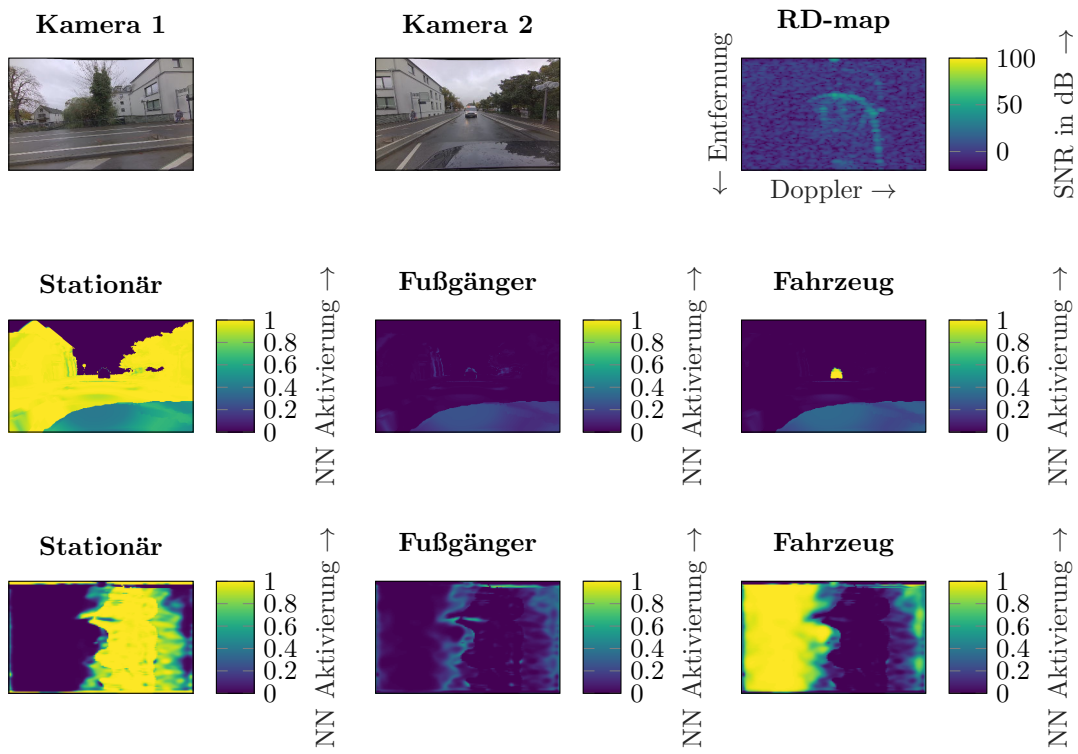


Abbildung A.9.: **Beispiel der semantischen Segmentierung:** Oben: Die zwei Kamerabilder und die RD-map. Mitte: Netzwerkausgaben für die Klassen „Stationär“, „Fußgänger“ und „Fahrzeug“ gewarpt in das Kamerabild. Unten: Netzwerkausgaben im RD-Gitter.

In Abbildung A.10 ist ein Szenario mit bewegtem Ego-Fahrzeug zu sehen. Im linken Bildbereich von Kamera 2 ist ein Fußgänger zu sehen, welcher neben einem parkenden Fahrrad steht. Dieser wurde korrekt als stationär klassifiziert.

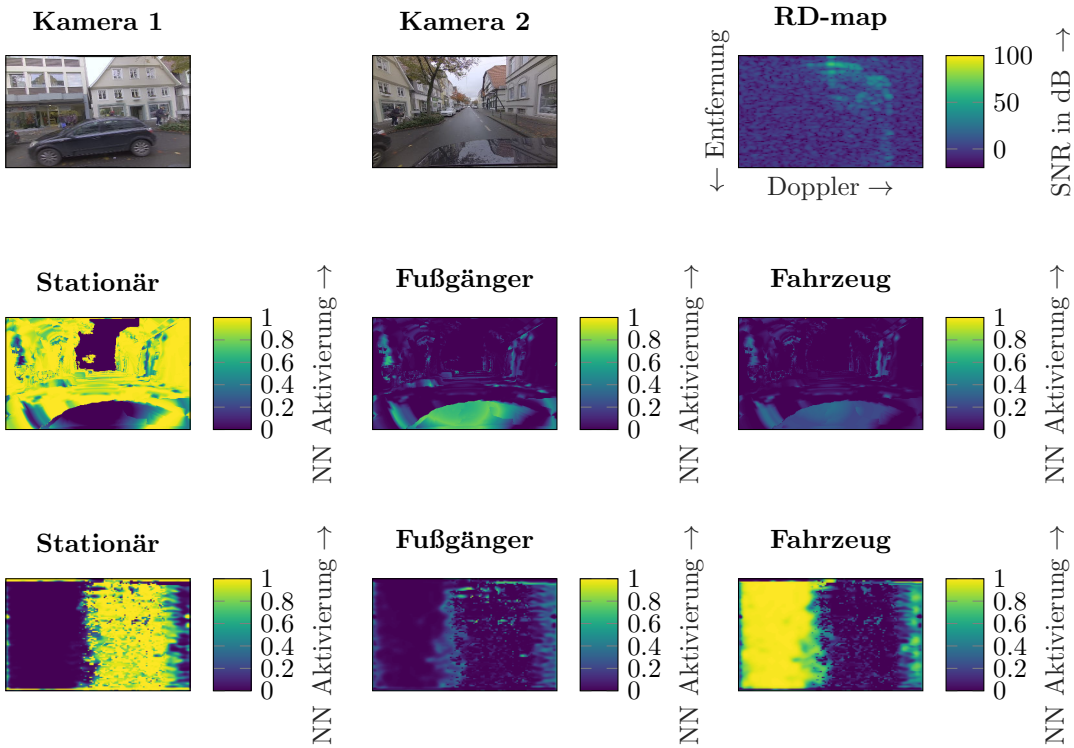


Abbildung A.10.: **Beispiel der semantischen Segmentierung:** Oben: Die zwei Kamerabilder und die RD-map. Mitte: Netzwerkausgaben für die Klassen „Stationät“, „Fußgänger“ und „Fahrzeug“ gewarpt in das Kamerabild. Unten: Netzwerkausgaben im RD-Gitter.

In Abbildung A.11 ist ein Szenario mit stehendem Ego-Fahrzeug zu sehen. Im Bild von Kamera 2 sind ein weiteres stationäres Fahrzeug sowie ein sich bewegendes Fahrrad samt Fahrer zu sehen. Fahrzeug und Fahrrad wurden korrekt als stationär und Vehikel klassifiziert.

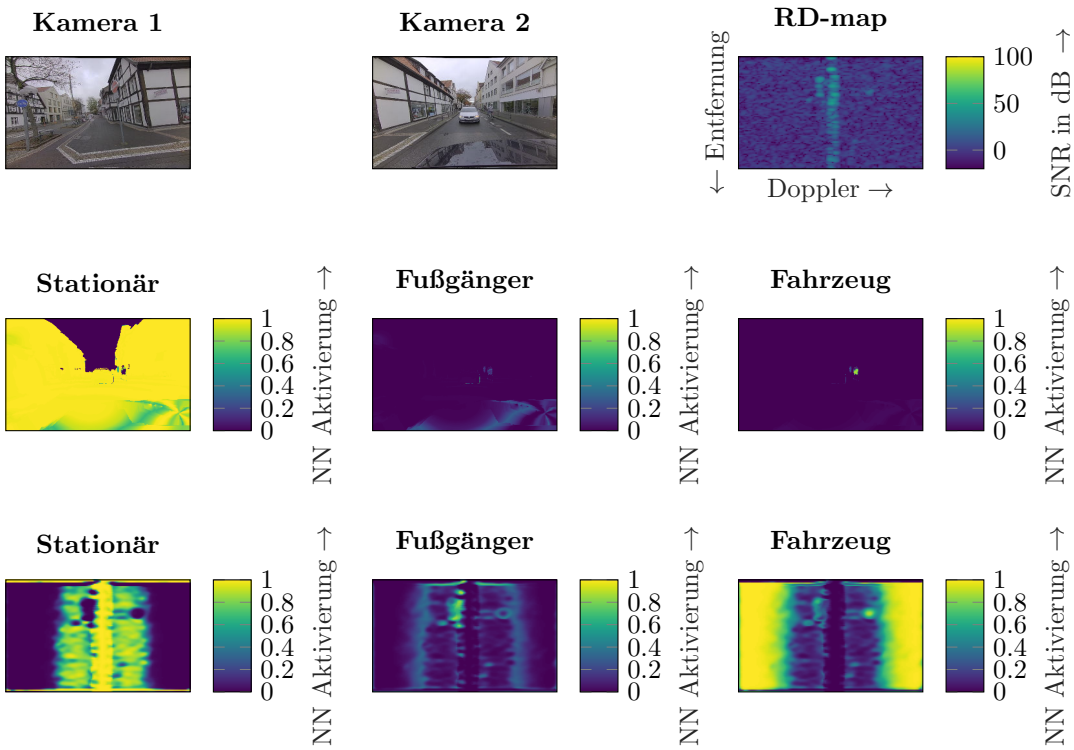


Abbildung A.11.: **Beispiel der semantischen Segmentierung:** Oben: Die zwei Kamerabilder und die RD-map. Mitte: Netzwerkausgaben für die Klassen „Stationät“, „Fußgänger“ und „Fahrzeug“ gewarpt in das Kamerabild. Unten: Netzwerkausgaben im RD-Gitter.

In Abbildung A.12 ist ein Szenario mit fahrendem Ego-Fahrzeug zu sehen. Im Bild von Kamera 2 sind parkende Fahrzeuge zu sehen. Diese wurden korrekt als stationär erkannt.

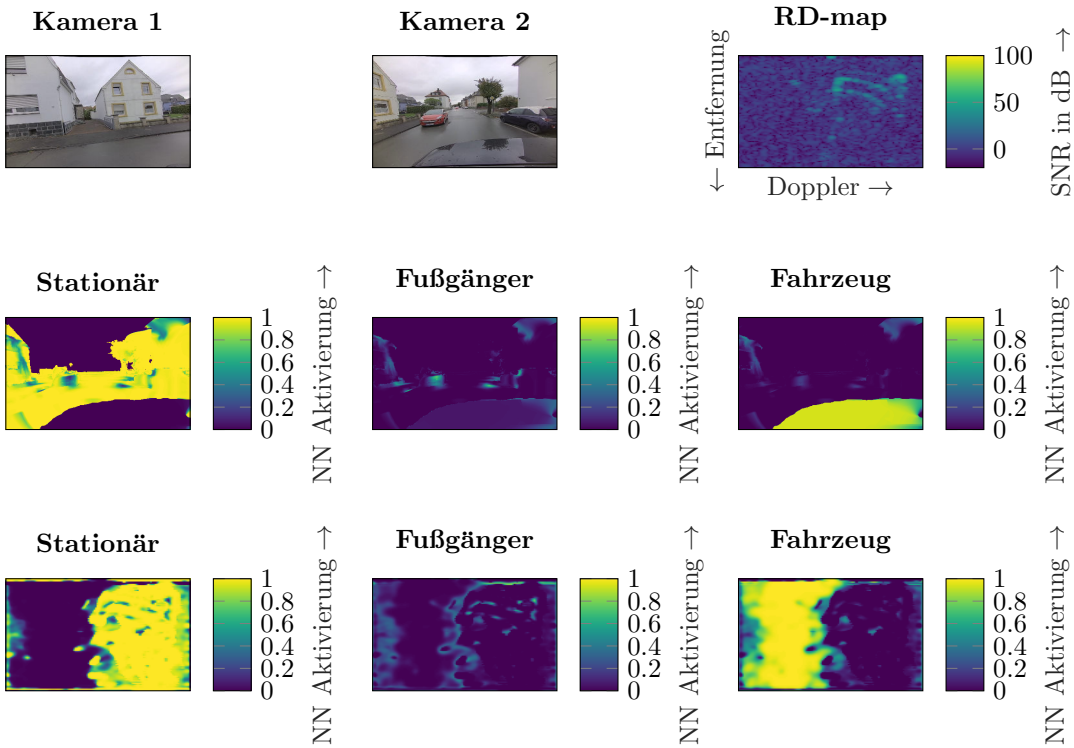


Abbildung A.12.: **Beispiel der semantischen Segmentierung:** Oben: Die zwei Kamerabilder und die RD-map. Mitte: Netzwerkausgaben für die Klassen „Stationät“, „Fußgänger“ und „Fahrzeug“ gewarpt in das Kamerabild. Unten: Netzwerkausgaben im RD-Gitter.

In Abbildung A.13 ist ein Szenario mit fahrendem Ego-Fahrzeug zu sehen. Die Umgebung ist stationär und wurde folgerichtig als stationär klassifiziert.

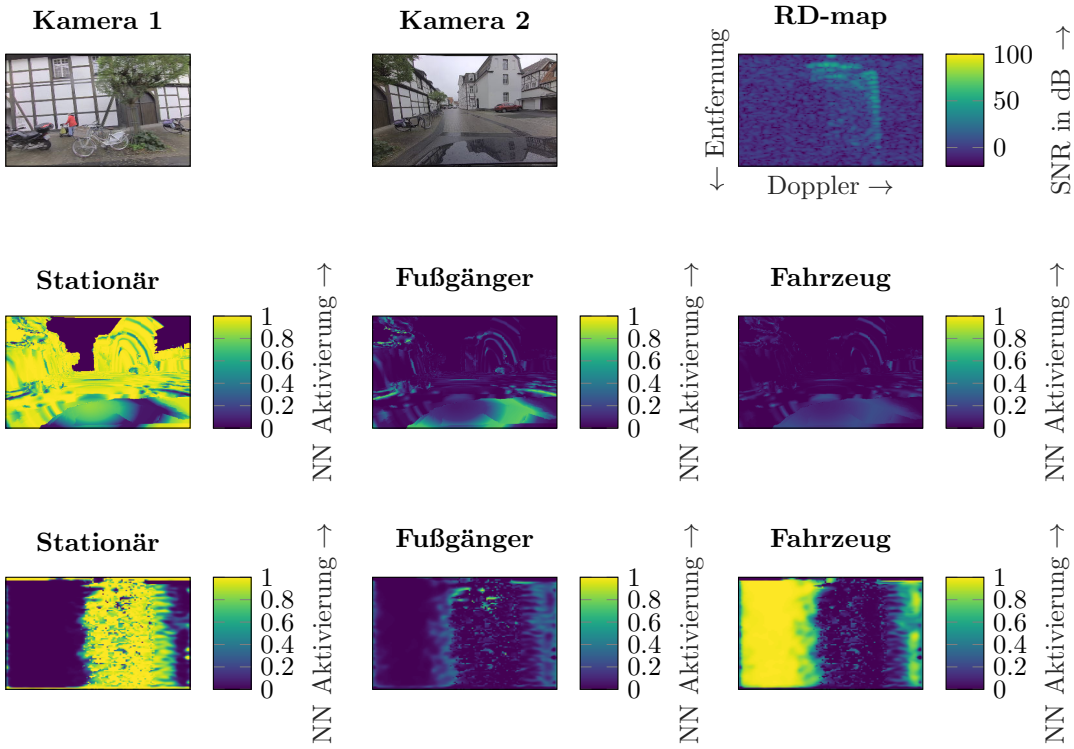


Abbildung A.13.: **Beispiel der semantischen Segmentierung:** Oben: Die zwei Kamerabilder und die RD-map. Mitte: Netzwerkausgaben für die Klassen „Stationät“, „Fußgänger“ und „Fahrzeug“ gewarpt in das Kamerabild. Unten: Netzwerkausgaben im RD-Gitter.

Akronyme

ADAS Advanced-Driver-Assistance-Systems.

ADU Analog-Digital-Umsetzer.

BBox Bounding-Box.

BF Beamforming.

BI bilineare-Interpolation.

CA-CFAR Cell Averaging Constant False Alarm Rate.

CAD Computer-Aided-Design.

CFAR Constant-False-Alarm-Rate.

CNN Convolutional-Neural-Network.

CS Chirp-Sequence.

DBSCAN Density-Based-Spatial-Clustering-of-Applications-with-Noise.

DFT Diskrete-Fourier-Transformation.

DGPS Differential-GPS.

DGPS-INS Differential-GPS-with-Inertial-Navigation-System.

DNN Deep-Neural-Network.

DoA Direction-of-Arrival.

DRISF Deep-Rigid-Instance-Scene-Flow.

DRISFwR Deep-Rigid-Instance-Scene-Flow-with-Radar.

EM elektromagnetische.

FFT Fast-Fourier-Transformation.

FMCW Frequency-Modulated-Continuous-Wave.

FoV Field-of-View.

FPS Frames-Per-Second.

FT Fourier-Transformation.

GAN Generative-Adversarial-Network.

GN Gauß-Newton.

GPS Global-Positioning-System.

GPU Graphics-Processing-Unit.

ICP Iterative-Closest-Points.

KOOS Koordinatensystem.

LKW Lastkraftwagen.

MAE Mean-Absolute-Error.

MIoU Mean-Intersection-over-Union.

ML Machine-Learning.

MTI Moving-Target-Indication.

NN neuronale Netzwerke.

NNI Nächster-Nachbar-Interpolation.

PC Personal-Computer.

PCA Prinicipal-Component-Analysis.

PKW Personenkraftwagen.

PM Phase-comparison-Monopulse.

RA range-Azimuth.

RAD range-azimuth-Doppler.

RAED range-azimuth-elevation-Doppler.

RCS Radar-Cross-Section.

RD range-Doppler.

RGB Red-Green-Blue.

RNN Recurrent-Neural-Network.

Rx Receiver.

SAR Synthetic-Aperture-Radar.

SNR Signal-to-noise-ratio.

SSIM Structural-Similarity-Index.

STFT Short-Time-Fourier-Transformation.

TI trilineare-Interpolation.

UAV Unmanned-Aerial-Vehicle.

UDP User-Datagram-Protocol.

ULA Uniform-Linear-Array.

VAE Variational-Autoencoder.

WDF Wahrscheinlichkeitsdichtefunktion.

Notationen und Symbole

Notationen

Zur einheitlichen Darstellung in dieser Arbeit werden folgende Notationen verwendet:

- Zur Unterscheidung von Skalaren, Vektoren und Matrizen werden Kleinbuchstaben, Kleinbuchstaben in Fett und Großbuchstaben in Fett verwendet, z.B. x, $\mathbf{x}$ und , $\mathbf{X}$.
- Zur Darstellung von Mengen, wird die kalligrafische Darstellung verwendet, z.B. $\mathcal{X}$.
- Zusatzinformationen wie Koordinatensystem, Namen, Indices werden als Kleinbuchstaben an die Variable angehängt, z.B. $\mathbf{x}_i$. Zur Indexierung von Matrixelementen, z.B. Zeilen- und Spaltenindex, werden diese tiefgestellt in eckige Klammern notiert, z.B. $X_{p[ij]}$.
- Zeitliche Kennzeichnung von Symbolen werden durch hochgestellte, in runden Klammern notierte Ziffern notiert, z.B. $x^{(i)}$.
- Als Dezimaltrennzeichen bei Zahlenangaben wird ein Punkt verwendet, z.B. so dass €1.23 dem Wert „ein Euro dreiundzwanzig Cent“ entspricht.
- Komplexe Konjugation wird durch hochgestelltes Sternchen X^* gekennzeichnet.

Symbole

Kapitel 2

ϕ_{az}	Azimutwinkel der Reflexion
ϕ_{el}	Elevationswinkel der Reflexion
$\hat{\mathbf{b}}$	Position des Reflektors in Reflektorkoordinaten
$\hat{\mathbf{r}}$	Position des Reflektors in Radarkoordinaten
r	Relativer Abstand des Reflektors
λ	Wellenlänge
$\mathbf{S}_{\text{rd}}$	komplexwertiges Range Doppler FFT Spektrum
$\mathbf{RD}$	RD-map

Kapitel 3

$\mathbf{x}_J$	3D Koordinaten im Koordinatensystem J in Vektorschreibweise
${}^{I\leftarrow J}\mathbf{R}$	Rotationsmatrix zur Drehung vom Ursprungskoordinatensystem J in das Zielkoordinatensystem I
${}^{I\leftarrow J}\mathbf{t}$	Translationsvektor zur Verschiebung der Koordinaten vom Ursprungskoordinatensystem J in das Zielkoordinatensystem I

Kapitel 4

$\mathbf{RGB}$	Kamerabild mit RGB Farbkanälen
$\mathbf{D}_{\text{sparse}}$	Spärliche Tiefenmaske
$\mathbf{D}$	Dichte Tiefenmaske
$\mathbf{N}$	Maske der Oberflächennormalen
$\mathbf{M}$	Maske der semantische Instanz-Segmentierung
$\mathbf{M}_{\text{Veh.}}$	Instanz-Maske für Fahrzeuge
$\mathbf{M}_{\text{Ped.}}$	Instanz-Maske für Fußgänger
$\mathbf{F}$	Maske des optischen Flusses
$\mathcal{P}_{\text{DBSCAN}}$	Menge der durch DBSCAN geclusterten Pixel

Kapitel 5

$\mathcal{P}_{\text{fg}}$	Menge aller Pixel der Klassen Fußgänger oder Fahrzeug
$\mathcal{P}_{\text{radar}}$	Menge aller Pixel im Radar FoV
$\mathcal{P}_{\text{i}}$	Schnittmenge aller Pixel aus Radar FoV und verfeinerter Instanz-Segmentierung via DBSCAN
$\boldsymbol{\xi}$	Szenenfluss eines Pixels ($\mathbb{R}^{3\times1}$)
$\boldsymbol{\xi}_{bg}$	Szenenfluss induziert durch Bewegung der Kamera über Grund ($\mathbb{R}^{3\times1}$)
$\boldsymbol{\xi}_{fg}$	Szenenfluss induziert durch Bewegung von Aktoren über Grund ($\mathbb{R}^{3\times1}$)
$\mathbf{p} = [u, v]^t$	Pixel im Kamerabild ($\mathbb{N}^{2\times1}$)
$\mathbf{p}_r = [u_r, v_r]^t$	Pixel in der RD-map ($\mathbb{N}^{2\times1}$)
$\dot{r}(\boldsymbol{\xi}_{\text{radar}})$	Radiale Geschwindigkeit eines Punktes im Radarkoordinatensystem
$\boldsymbol{\xi}_{\text{radar6}}$	Kartesische Geschwindigkeit eines Punktes im Radarkoordinatensystem
$\mathbf{R}_{C,bg}$	Rotationsmatrix zur Aufprägung der Drehung bedingt durch Ego-Bewegung, für Punkte gegeben im Kamera-Koordinatensystem
$\mathbf{t}_{C,bg}$	Translationsvektor zur Aufprägung der Translation bedingt durch Ego-Bewegung, für Punkte gegeben im Kamera-Koordinatensystem
$\boldsymbol{\xi}_{\text{radar}}$	Szenenfluss eines Pixels ($\mathbb{R}^{3\times1}$ im Radarkoordinatensystem)

Kapitel 6

$\mathbf{RD}_C$ ..... Auf das Pixelgitter des Kamerabildes projizierte RD-map

$\eta_{\mathrm{BW}}\left(x_{\text{RD-grid}}; \dot{r}, r\right)$ .. Rückwärtswarp Operation

$\eta_{\mathrm{FW}}\left(x_{\mathrm{cam}}; \dot{r}, r\right)$ ..... Vorwärtswarp Operation

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[Ali22] Alishobeiri. Monocular video odometry using opencv. GitHub repository, 2022.

[ARFssN16] Azli Abd Razak, Nor Fohimi, sheikh ahmad zaki shaikh salim, and Noor Hafiz Noordin. Temperature influence on total volatile compounds (tvocs) inside the car cabin of visible light transmittance. *Journal of Engineering and Applied Sciences*, 11:8629–8634, 07 2016.

[Bar19] Jonathan T. Barron. A general and adaptive robust loss function. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4326–4334, 2019.

[BDE+09] M. Bass, C. DeCusatis, J.M. Enoch, V. Lakshminarayanan, G. Li, C. MacDonald, V.N. Mahajan, and E. Van Stryland. *Handbook of Optics, Third Edition Volume I: Geometrical and Physical Optics, Polarized Light, Components and Instruments(set)*. Handbook of Optics. McGraw Hill LLC, 2009.

[BGM+20] Dan Barnes, Matthew Gadd, Paul Murcutt, Paul Newman, and Ingmar Posner. The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6433–6438, 2020.

[BIP06] BIPM. *The International System of Units (SI brochure (EN)): 8th edition, 2006*, volume 1. April 2006.

[Bis95] C.M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, USA, 1995.

[BM13] Nauman Anwar Baig and Mohammad Bilal Malik. Comparison of direction of arrival (doa) estimation techniques for closely spaced targets. *International Journal of Future Computer and Communication*, pages 654–659, 2013.

[Bos] Bosun Bobs trem radar reflector. https://www.bosunbobs.com/en/Trem-Radar-Reflector/m-749.aspx.

[BRB22] Kshitiz Bansal, Keshav Rungta, and Dinesh Bharadia. Radsegnet: A reliable approach to radar camera fusion, 2022.

[Bro66] D. C. Brown. Decentering Distortion of Lenses. *Photometric Engineering*, 32(3):444–462, 1966.

[BRZB20] Kshitiz Bansal, Keshav Rungta, Siyuan Zhu, and Dinesh Bharadia. Pointillism: Accurate 3d bounding box estimation with multi-radars. SenSys '20, page 340–353, New York, NY, USA, 2020. Association for Computing Machinery.

[BWY+22] Keenan Burnett, Yuchen Wu, David J. Yoon, Angela P. Schoellig, and Timothy D. Barfoot. Are we ready for radar to replace lidar in all-weather mapping and localization? *IEEE Robotics and Automation Letters*, 7(4):10328–10335, 2022.

[BYW+23] Keenan Burnett, David J Yoon, Yuchen Wu, Andrew Z Li, Haowei Zhang, Shichen Lu, Jingxing Qian, Wei-Kang Tseng, Andrew Lambert, Keith YK Leung, Angela P Schoellig, and Timothy D Barfoot. Boreas: A multi-season autonomous driving dataset. *The International Journal of Robotics Research*, 42(1-2):33–42, 2023.

[CBL+19] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. [cs.LG], March 2019.

[CGQ+23] Kan Chen, Runzhou Ge, Hang Qiu, Rami Ai-Rfou, Charles R. Qi, Xuanyu Zhou, Zoey Yang, Scott Ettinger, Pei Sun, Zhaoqi Leng, Mustafa Mustafa, Ivan Bogun, Weiyue Wang, Mingxing Tan, and Dragomir Anguelov. Womd-lidar: Raw sensor dataset benchmark for motion forecasting. *arXiv preprint arXiv:2304.03834*, April 2023.

[Che09] Scott Chun-Yang Chen. *Signal Processing Algorithms for MIMO Radar*. PhD thesis, California Institute of Technology, 2009.

[CMMS11] D. Ciresan, U. Meier, J. Masci, and J. Schmidhuber. Multi-column deep neural network for traffic sign classification. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2011.

[COR+16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[CYH+23] Minseong Choi, Seunghoon Yang, Seungho Han, Yeongseok Lee, Minyoung Lee, Keun Ha Choi, and Kyung-Soo Kim. Msc-rad4r: Ros-based automotive dataset with 4d radar. *IEEE Robotics and Automation Letters*, 8(11):7194–7201, 2023.

[DD21] Carsten Ditzel and Klaus Dietmayer. Genradar: Self-supervised probabilistic camera synthesis based on radar frequencies. *IEEE Access*, 9:148994–149042, 2021.

[Dim23] Dimitrievski, Martin. *Cooperative sensor fusion for autonomous driving*. PhD thesis, Ghent University, 2023.

[DSH+20] Martin Dimitrievski, Ivana Shopovska, David Van Hamme, Peter Veelaert, and Wilfried Philips. Weakly supervised deep learning method for vulnerable road user detection in fmcw radar. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8, 2020.

[DSVH+21] Martin Dimitrievski, Ivana Shopovska, David Van Hamme, Peter Veelaert, and Wilfried Philips. Automatic labeling of vulnerable road users in multi-sensor data. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2623–2630, 2021.

[DT06] J. Diebel and S. Thrun. An application of Markov random fields to range sensing. In *proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2006.

[Dur19] Rick Durrett. *Probability: Theory and Examples*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 5 edition, 2019.

[DWZL20] Xu Dong, Pengluo Wang, Pengyue Zhang, and Langechuan Liu. Probabilistic oriented object detection in automotive radar. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 458–467, 2020.

[ECC+] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R. Qi, Yin Zhou, Zoey Yang, Aur'elien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan, Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset.

[EHR23] Lukas Elster, Martin F. Holder, and Manuel Rapp. A dataset for radar scattering characteristics of vehicles under real-world driving conditions: Major findings for sensor simulation. *IEEE Sensors Journal*, 23(5):4873–4882, 2023.

[EKSX96] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the International Conference on Knowledge Discovery and Data (KDD)*, 1996.

[eLa21] eLab. Github repository: pytorch-toolbox. `https://github.com/e-lab/pytorch-toolbox/tree/master/profiler`, 2021.

[Faw06] Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. ROC Analysis in Pattern Recognition.

[FB81] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[Fir] FirstSensor. Blue next network digital camera e4p. `https://www.first-sensor.com/cms/upload/datasheets/DS_Blue_Next_DC3C-1-E4P.pdf`. datasheet.

[FP12] David A. Forsyth and Jean Ponce. *Computer Vision - A Modern Approach, Second Edition.* Pitman, 2012.

[FPP07] David Freedman, Robert Pisani, and Roger Purves. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007.

[F.R01] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[FWG19a] J. Fuchs, R. Weigel, and M. Gardill. Model Order Estimation using a Multi-Layer Perceptron for Direction-of-Arrival Estimation in Automotive Radar Sensors. In *Proceedings of the 2019 IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNet)*, 2019.

[FWG19b] J. Fuchs, R. Weigel, and M. Gardill. Single-snapshot direction-of-arrival estimation of multiple targets using a multi-layer perceptron. In *Proceedings of the 2019 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, 2019.

[GB10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[Gen20] GeneSys Elektronik GmbH. Adma. `https://www.genesys-offenburg.de/index.php?eID=tx_securedownloads&p=110&u=0&g=0&t=1595915060&hash=ccb5d292bcfc9f7f7492e7b888d0a8d6ca71736f&file=fileadmin/user_upload/ProdDescr_ADMA_rel_05.2019.pdf`, 2020. datasheet.

[Gen23] IPI University Gent. Autonomous vehicles and sensor fusion. `https://ipi.ugent.be/research/auto_vehicles/`, 2023. Besucht: 02.11.2023.

[GFFW18] M. Gardill, J. Fuchs, C. Frank, and R. Weigel. A multi-layer perceptron applied to number of target indication for direction-of-arrival estimation in automotive radar sensors. In *Proceedings of the IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2018.

[GLSU12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. *Vision meets Robotics: The KITTI Dataset*. International Journal of Robotics Research (IJRR), 2012.

[GLU12] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[GLX+22] Xiangyu Gao, Youchen Luo, Guanbin Xing, Sumit Roy, and Hui Liu. Githup-repo: Raw adc data of 77ghz mmwave radar for automotive object detection. `https://dx.doi.org/10.21227/xm40-jx59`, 2022.

[GMJ+20] Junfeng Guan, Sohrab Madani, Suraj Jog, Saurabh Gupta, and Haitham Hassanieh. Through fog high-resolution imaging using millimeter wave radar. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11461–11470, 2020.

[GOO+17] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and José García Rodríguez. A review on deep learning techniques applied to semantic segmentation. *CoRR*, abs/1704.06857, 2017.

[Gva] GVAAT's Workshop how to draw vehicles in perspective, a step-by-step guide. https://www.gvaat.com/blog/how-to-draw-vehicles-in-perspective-step-by-step/.

[GXRL21] Xiangyu Gao, Guanbin Xing, Sumit Roy, and Hui Liu. Ramp-cnn: A novel neural network for enhanced automotive radar object recognition. *IEEE Sensors Journal*, 21(4):5119–5132, 2021.

[Han13] Dianyuan Han. Comparison of commonly used image interpolation methods. *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering*, 03 2013.

[HBS+23] Marcel Hoffmann, Sandro Braun, Oliver Sura, Michael Stelzig, Christian Schüßler, Knut Graichen, and Martin Vossiek. Concept for an automatic annotation of automotive radar data using ai-segmented aerial camera images, 2023.

[HDH+19] Jahn Heymann, Lukas Drude, Reinhold Haeb-Umbach, Keisuke Kinoshita, and Tomohiro Nakatani. Joint optimization of neural network-based WPE dereverberation and acoustic model for robust online ASR. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 6655–6659. IEEE, 2019.

[Heu13] S. Heuel. *Fußgängererkennung im Straßenverkehr mit 24GHz Radarsensoren*. PhD thesis, Hamburg University of Technology, 2013.

[HHBD18] Stefan Hoermann, Philipp Henzler, Martin Bach, and Klaus Dietmayer. Object detection on dynamic occupancy grid maps using deep learning and automatic label generation. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 826–833, 2018.

[HLL+23] Chengming Hu, Xuan Li, Dan Liu, Haolun Wu, Xi Chen, Ju Wang, and Xue Liu. Teacher-student architecture for knowledge distillation: A survey, 2023.

[HN09] A. Harrison and P. Newman. Image and sparse laser fusion for dense scene reconstruction. In *Proceedings of the Field and Service Robotics*, 2009.

[HRS+17] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3296–3297, 2017.

[HS81] Berthold K.P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1):185–203, 1981.

[Hun91] Bruce Hunt. *The Maxwellians*. Cornell„ New York, 1991.

[HVD15] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, 2015.

[HZRS15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[Hü04a] Christian Hülsmeyer. Verfahren, um entfernte metallische gegenstände mittels elektrischer wellen einem beobachter zu melden, DE 165546-A, 1904.

[Hü04b] Christian Hülsmeyer. Verfahren, um entfernte metallische gegenstände mittels elektrischer wellen einem beobachter zu melden, DE 169154-A, 1904.

[ISO11] Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary. Standard, International Organization for Standardization, Geneva, CH, 2011.

[Jäh05] B. Jähne. *Digitale Bildverarbeitung*. Springer, Berlin, 2005.

[JDFMV23] Yi Jin, Anastasios Deligiannis, Juan-Carlos Fuentes-Michel, and Martin Vossiek. Cross-modal supervision-based multitask learning with automotive radar raw data. *IEEE Transactions on Intelligent Vehicles*, 8(4):3012–3025, 2023.

[JF07] Steven G. Johnson and Matteo Frigo. A modified split-radix fft with fewer arithmetic operations. *IEEE Transactions on Signal Processing*, 55(1):111–119, 2007.

[Jud88] S. Juds. *Photoelectric Sensors and Controls: Selection and Application, First Edition*. Mechanical Engineering. Taylor & Francis, 1988.

[Ka22] Kaist-avelab. Github repository: K-radar. `https://github.com/kaist-avelab/K-Radar`, 2022. GitHub repository.

[KB15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[KDMGN20] Prannay Kaul, Daniele De Martini, Matthew Gadd, and Paul Newman. RSS-Net: Weakly-Supervised Multi-Class Semantic Segmentation with FMCW Radar. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, 2020.

[Kel17] D. Kellner. *Verfahren zur Bestimmung von Objekt- und Eigenbewegung auf Basis der Dopplerinformation hochauflösender Radarsensoren*. PhD thesis, Technische Hochschule Ulm, 2017.

[KKPD23] Johannes Kopp, Dominik Kellner, Aldi Piroli, and Klaus Dietmayer. Tackling Clutter in Radar Data – Label Generation and Detection Using PointNet++. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1493–1499. IEEE, may 2023.

[KPC+20] Giseop Kim, Yeong Sang Park, Younghun Cho, Jinyong Jeong, and Ayoung Kim. Mulran: Multimodal range dataset for urban place recognition. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6246–6253, 2020.

[Kra88] J.D. Kraus. *Antennas*. Electrical engineering series. McGraw-Hill, 1988.

[KSRD21] Florian Kraus, Nicolas Scheiner, Werner Ritter, and Klaus Dietmayer. The radar ghost dataset – an evaluation of ghost objects in automotive radar data. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8570–8577, 2021.

[Kue17] U. Kuehnau. Sensor development for autonomous driving. Technical report, Automotive Radar Sensors for Semi-Automatic and Autonomous Driving and Parking Systems, IWPC Wolfsburg, 2017.

[KUI99] JACK B. KUIPERS. *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace and Virtual Reality*. Princeton University Press, 1999.

[KV96] H. Krim and M. Viberg. Two decades of array signal processing research. *IEEE Signal Processing Magazine*, 13, 1996.

[KWL22] Pou-Chun Kung, Chieh-Chih Wang, and Wen-Chieh Lin. Radar occupancy prediction with lidar supervision while preserving long-range sensing and penetrating capabilities. *IEEE Robotics and Automation Letters*, 7(2):2637–2643, 2022.

[LGG+20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327, 2020.

[LiD20] Velodyne LiDAR. Ultra puck datasheet. `http://www.mapix.com/wp-content/uploads/2018/07/63-9378_Rev-D_ULTRA-Puck_VLP-32C_Datasheet_Web.pdf`, 2020. Besucht: 27.07.2020.

[Lim19] Teck-Yian Lim. Radar and camera early fusion for vehicle detection in advanced driver assistance systems. In *The Thirty-third Annual Conference on Neural Information Processing Systems, NeurIPS 2019, Vancouver, Canada, Dec. 8-14, 2019*, 2019.

[LK81] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, page 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[Mat17] The Mathworks, Inc., Natick, Massachusetts. *MATLAB version 9.3.0.713579 (R2017b)*, 2017.

[MBB+23] Tamás Matuszka, Iván Barton, Ádám Butykai, Péter Hajas, Dávid Kiss, Domonkos Kovács, Sándor Kunsági-Máté, Péter Lengyel, Gábor Németh, Levente Pető, Dezső Ribli, Dávid Szeghy, Szabolcs Vajna, and Bálint Varga. aimotive dataset: A multimodal dataset for robust autonomous driving with long-range perception. 2023.

[MDB+17] Pierre Merriaux, Yohan Dupuis, Rémi Boutteau, Pascal Vasseur, and Xavier Savatier. Lidar point clouds correction acquired from a moving car based on can-bus data. *CoRR*, abs/1706.05886, 2017.

[Mei98] H. H. Meinel. Automotive millimeterwave radar history and present status. In *1998 28th European Microwave Conference*, volume 1, pages 619–629, 1998.

[Mes06] L. Mesow. *Multisensorielle Datensimulation im Fahrzeugumfeld für die Bewertung von Sensorfusionsalgorithmen*. PhD thesis, Technical University Chemnitz, 2006.

[MFA+19] B. Major, D. Fontijne, A. Ansari, R. T. Sukhavasi, R. Gowaikar, M. Hamilton, S. Lee, S. Grzechnik, and S. Subramanian. Vehicle detection with automotive radar using deep learning on range-azimuth-doppler tensors. In *IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019.

[MG15a] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[MG15b] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[MHG15] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015.

[MHG18] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018.

[MK19] Michael Meyer and Georg Kuschk. Automotive radar dataset for deep learning based 3d object detection. In *2019 16th European Radar Conference (EuRAD)*, pages 129–132, 2019.

[MKT21] Michael Meyer, Georg Kuschk, and Sven Tomforde. Graph convolutional networks for 3d object detection on radar data. In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 3053–3062, 2021.

[MMT10] Pascal Monasse, Jean-Michel Morel, and Zhongwei Tang. Three-step image rectification. In *Proceedings of the British Machine Vision Conference*, pages 89.1–89.10. BMVA Press, 2010. doi:10.5244/C.24.89.

[Mon22] Monodepth. How to get depth form disparity map. GitHub repository, 2022.

[MW04] M. Mitschke and H. Wallentowitz. *Dynamik der Kraftfahrzeuge*. Springer, 2004.

[MWH+19] Wei-Chiu Ma, Shenlong Wang, Rui Hu, Yuwen Xiong, and Raquel Urtasun. Deep rigid instance scene flow. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[MWRH20] Mohammadreza Mostajabi, Ching Ming Wang, Darsh Ranjan, and Gilbert Hsyu. High resolution radar dataset for semi-supervised learning of dynamic objects. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 450–457, 2020.

[NKK+20] Farzan Erlik Nowruzi, Dhanvin Kolhatkar, Prince Kapoor, Fahed Al Hassanat, Elnaz Jahani Heravi, Robert Laganiere, Julien Rebut, and Waqas Malik. Deep open space segmentation using automotive radar. In *2020 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, pages 1–4, 2020.

[NOBK17] G. Neuhold, T. Ollmann, S. Bulò, and P. Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[NSR+20] Arsha Nagrani, Chen Sun, David Ross, Rahul Sukthankar, Cordelia Schmid, and Andrew Zisserman. Speech2action: Cross-modal supervision for action recognition, 2020.

[Oct22] Octave. Surfnorm. `https://github.com/gnu-octave/octave/blob/default/scripts/plot/draw/surfnorm.m`, 2022. GitHub repository.

[ONR+21] Arthur Ouaknine, Alasdair Newson, Julien Rebut, Florence Tupin, and Patrick Pérez. Carrada dataset: Camera and automotive radar with range-angle- doppler annotations. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5068–5075, 2021.

[Pal14] W.J. Palm. *System Dynamics*. McGraw-Hill series in mechanical engineering. McGraw-Hill Higher Education, 2014.

[PDKG20] Andras Palffy, Jiaao Dong, Julian F. P. Kooij, and Dariu M. Gavrila. Cnn based road user detection using the 3d radar cube. *IEEE Robotics and Automation Letters*, 5(2):1263–1270, 2020.

[Pen55] R. Penrose. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413, 1955.

[PKW22] Dong-Hee Paek, Seung-Hyun Kong, and Kevin Tirta Wijaya. K-radar: 4d radar object detection for autonomous driving in various weather conditions. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[PPB+22] Andras Palffy, Ewoud Pool, Srimannarayana Baratam, Julian Kooij, and Dariu Gavrila. Multi-class road user detection with 3+1d radar in the view-of-delft dataset. *IEEE Robotics and Automation Letters*, pages 1–1, 2022.

[PSK20] Yeong Sang Park, Young-Sik Shin, and Ayoung Kim. Pharao: Direct radar odometry using phase correlation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2617–2623, 2020.

[RA15] D. Rueckert and P. Aljabar. *Non-rigid registration using free-form deformations*, pages 277–294. Springer US, Boston, MA, 2015.

[Rad21] RadarScnenes. Radarscenes/labeling. `https://radar-scenes.com/dataset/labeling/`, 2021. Besucht: 02.11.2023.

[ROMP22] Julien Rebut, Arthur Ouaknine, Waqas Malik, and Patrick Pérez. Raw high-definition radar for multi-task learning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17000–17009, 2022.

[Roy08] C. Roychoudhuri. *Fundamentals of photonics*. Bellingham, 2008.

[Sau16] C. Sauloff. Entwicklung eines verfahrens zur objektklassifikation in bezug auf automotive radar-anwendungen. *M.S. thesis, Faculty ofComputer Vision and Remote Sensing, University Berlin*, 2016, June 2016.

[Sch13] C. Schröder. *System Design of an Array Antenna Radar with a Rapid Chirp Waveform*. Berichte aus der Elektrotechnik. Shaker Verlag, 2013.

[Sch14] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *CoRR*, abs/1404.7828, 2014.

[SDPM+21] Marcel Sheeny, Emanuele De Pellegrin, Saptarshi Mukherjee, Alireza Ahrabian, Sen Wang, and Andrew Wallace. Radiate: A radar dataset for automotive perception in bad weather. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7, 2021.

[Sha49] C.E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, jan 1949.

[SHS+21] Ole Schumann, Markus Hahn, Nicolas Scheiner, Fabio Weishaupt, Julius Tilly, Juergen Dickmann, and Christian Wohler. Radarscenes: A real-world radar point cloud data set for automotive applications. pages 1–8, 11 2021.

[SKD+20] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[Sko01] Merrill I. Skolnik. *Introduction to radar systems*. McGraw-Hill international editions. Electrical engineering series. McGraw-Hill, Boston, [Mass.] ;, 3rd ed. edition, 2001.

[SM23] Arvind Srivastav and Soumyajit Mandal. Radars for autonomous driving: A review of deep learning methods and challenges. *IEEE Access*, 11:97147–97168, 2023.

[SMB10] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *Proceedings of the 20th International Conference on Artificial Neural Networks: Part III*, ICANN'10, page 92–101, Berlin, Heidelberg, 2010. Springer-Verlag.

[SRB10] Deqing Sun, Stefan Roth, and Michael J. Black. Secrets of optical flow estimation and their principles. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2432–2439, 2010.

[ST94] Jianbo Shi and Carlo Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[Sze10] R. Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. Springer London, 2010.

[TBKP12] Michael Tao, Jiamin Bai, Pushmeet Kohli, and Sylvain Paris. Simpleflow: A non-iterative, sublinear optical flow algorithm. *Computer Graphics Forum*, 31, 05 2012.

[Ten20] Tensorflow. Detection model zoo. `https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md`, 2020. Besucht: 01.04.2020.

[TFR21] Jörn Thieling, Susanne Frese, and Jürgen Roßmann. Scalable and physical radar sensor simulation for interacting digital twins. *IEEE Sensors Journal*, 21(3):3184–3192, 2021.

[Tzu15] Tzutalin. Labelimg. `https://github.com/tzutalin/labelImg`, 2015. GitHub repository.

[Uni11] International Telecommunicattion Union. Bt.601 : Studio encoding parameters of digital television for standard 4:3 and wide screen 16:9 aspect ratios. Recommendation BT.601, 2011.

[USS+17] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *International Conference on 3D Vision (3DV)*, 2017.

[vDG08] P. van Dorp and F. Groen. *Feature-based human motion parameter estimation with radar*. IEEE Transactions on Radar, Sonar & Navigation, 2008.

[WBSS04] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *Trans. Img. Proc.*, 13(4):600–612, April 2004.

[Wik22] Wikipedia. Krümmung — Wikipedia, the free encyclopedia. `http://de.wikipedia.org/w/index.php?title=Kr%C3%BCmmung&oldid=208068636`, 2022. [Online; accessed 01-March-2022].

[Wik23] Wikipedia. Zeitkomplexität — Wikipedia, the free encyclopedia. `https://de.wikipedia.org/wiki/Zeitkomplexit%C3%A4t`, 2023. [Online; accessed 11-October-2023].

[WJG+20] Yizhou Wang, Zhongyu Jiang, Xiangyu Gao, Jenq-Neng Hwang, Guanbin Xing, and Hui Liu. Rodnet: Radar object detection using cross-modal supervision, 2020.

[WJL+21] Yizhou Wang, Zhongyu Jiang, Yudong Li, Jenq-Neng Hwang, Guanbin Xing, and Hui Liu. Rodnet: A real-time radar object detection network cross-supervised by camera-radar fused object 3d localization. *IEEE Journal of Selected Topics in Signal Processing*, 15(4):954–967, 2021.

[WJP21] Rob Weston, Oiwi Parker Jones, and Ingmar Posner. There and back again: Learning to simulate radar data for real-world applications. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12809–12816, 2021.

[Woo06] I. Woodhouse. *Introduction to Microwave Remote Sensing*. CRC, Boca Raton, USA, 2006.

[WWH+21] Yizhou Wang, Gaoang Wang, Hung-Min Hsu, Hui Liu, and Jenq-Neng Hwang. Rethinking of radar's role: A camera-radar dataset and systematic annotator via coordinate alignment, 2021.

[YDY19] Z. Yin, T. Darrell, and F. Yu. Hierarchical discrete distribution decomposition for match density estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[YLZL06] Y. Yang, J. Lei, W. Zhang, and C. Lu. Target classification and pattern recognition using micro-doppler radar signatures. In *International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, 2006.

[Zhe99] Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 666–673 vol.1, 1999.

[ZHO23] ZHOUYI1023. Awesome radar perception. GitHub repository, 2023.

[ZLA+19] M. Zhao, T. Li, M. Alsheikh, Y. Tian, H. Zhao, A. Torralba, and D. Katabi. Through-wall human pose estimation using radio signals. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[ZLZ+22] Yi Zhou, Lulu Liu, Haocheng Zhao, Miguel López-Benítez, Limin Yu, and Yutao Yue. Towards deep radar perception for autonomous driving: Datasets, methods, and challenges. *Sensors*, 22(11), 2022.

[ZMZ+22] Lianqing Zheng, Zhixiong Ma, Xichan Zhu, Bin Tan, Sen Li, Kai Long, Weiqi Sun, Sihan Chen, Lu Zhang, Mengyue Wan, Libo Huang, and Jie Bai. Tj4dradset: A 4d radar dataset for autonomous driving. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 493–498, 2022.

[ZNL21] Ao Zhang, Farzan Erlik Nowruzi, and Robert Laganiere. Raddet: Range-azimuth-doppler based radar object detection for dynamic road users. In *2021 18th Conference on Robots and Vision (CRV)*, pages 95–102, 2021.

[ZTM+17] X. Zhu, D. Tuia, L. Mou, G. Xia, L. Zhang, F. Xu, and F. Fraundorfer. Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine*, 5, 2017.

Eigene Publikationen

[EB1] C. Grimm. Self-learning methods for automotive radar sensors. Technical report, Automotive Radar Sensors for Semi-Automatic and Autonomous Driving and Parking Systems, IWPC Wolfsburg, 2017.

[EB2] C. Grimm, T. Breddermann, R. Farhoud, T. Fei, E. Warsitz, and R. Haeb-Umbach. Detection of moving targets in automotive radar with distorted ego-velocity information. In Radar and R. S. S. (MRRS), editors, *Proceedings of the IEEE MICROWAVES*, 2017.

[EB3] C. Grimm, T. Breddermann, R. Farhoud, T. Fei, E. Warsitz, and R. Haeb-Umbach. Hypothesis test for the detection of moving targets in automotive radar. In *2017 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS)*, pages 1–6, 2017.

[EB4] C. Grimm, T. Breddermann, R. Farhoud, T. Fei, E. Warsitz, and R. Haeb-Umbach. Discrimination of stationary from movingtargets with recurrent neural networks in automotiveradar. In *Proceedings of the IEEE International Conference on Microwaves for Intelligent Mobility (ICMIM)*, 2018.

[EB5] C. Grimm, R. Farhoud, T. Fei, E. Warsitz, and R. Haeb-Umbach. Detection of moving targets in automotive radar with distorted ego-velocity information. In *2017 IEEE Microwaves, Radar and Remote Sensing Symposium (MRRS)*, pages 111–116, 2017.

[EB6] C. Grimm, T. Fei, E. Warsitz, R. Farhoud, T. Breddermann, and R. Haeb-Umbach. Warping of radar data into camera image for cross-modal supervision in automotive applications. *IEEE Transactions on Vehicular Technology*, 71(9):9435–9449, 2022.

[EB7] C. Grimm, T. Fei, E. Warsitz, R. Farhoud, T. Breddermann, and R. Haeb-Umbach. On the rcs estimation from camera images. In *2023 24th International Radar Symposium (IRS)*, pages 1–12, 2023.