Semantische Umgebungserfassung auf Basis von Radar-Merkmalskarten

Von der Fakultät
Elektrotechnik und Informationstechnik
der Technischen Universität Dortmund
genehmigte

Dissertation

zur Erlangung des akademischen Grades Doktor der
Ingenieurwissenschaften eingereicht von

Jakob Lombacher

Dortmund, 2021

Semantische Umgebungserfassung auf Basis von Radar-Merkmalskarten

Jakob Lombacher

Dissertation in der Fakultät für Elektrotechnik und Informationstechnik, Arbeitsgebiet Bildsignalverarbeitung der Technischen Universität Dortmund.

Tag der mündlichen Prüfung:	06.10.2021
Hauptreferent:	Prof. Dr. Christian Wöhler
Korreferent:	Prof. Dr. Friedhelm Schwenker

Abstract

Autonomous driving and modern driving assistance system require a detailed detection and interpretation of the environment. In addition to a more precise modelling of the shape and state of an object, a better semantical understanding of the driving situation is needed. The high demands on functional safety can often only be satisfied by using redundant systems.

Today, semantic object classification systems are heavily based on optical sensors. Even though the number of radar sensors in a vehicle are constantly increasing and the resolution, accuracy and sensitivity of each sensor improves significantly from one generation to the next, the radar sensor still contributes little to semantic classification.

This study investigates the potential of the radar sensor for semantic environment perception of vehicles. While there are several studies on the classification of dynamic objects there is hardly any research on classification of static objects. This research project focuses on the classification of the static world.

Starting from Radar detections, this thesis covers all processing steps which are necessary to developing a radar classification system. It shows methodologies to aggregate Radar data into feature maps, methods to classify radar data and techniques to improve the training of a classification system. In order to develop and evaluate the classification system a dataset containing several hours of real-world data was created. To label this dataset a method of manual annotation was developed.

Using this dataset, the Radar specific properties of different categories of objects were examined and a meaningful taxonomy was developed. This taxonomy was the basis for the training and evaluation of the classification algorithms. Both cluster-based algorithms and methods of semantic segmentation were used for classification. Cluster-based algorithms first extract object proposals, which are classified in a second step. By contrast, semantic segmentation algorithms operate directly on the radar feature map.

Using these algorithms, the study shows that neural networks which were pre-trained on image data yield significantly better results than networks which are solely trained on radar data. For further improvement of the classification results, a system is presented which optimizes the augmentation of the training data using Genetic Algorithms.

Finally, classification systems for all classes (car, building, pole, curbstone and other) are presented. Based on these systems the study shows that the classification of the static world achieved promising results. Furthermore, it could be shown that it is possible to estimate the orientation and dimensions of partially observed vehicles. This could, for example, add value to autonomous parking systems. Thus, Radar could make valuable contributions to the perception of the vehicle environment.

Kurzfassung

Moderne Fahrerassistenzsysteme und autonomes Fahren benötigen eine detaillierte Umgebungserfassung. Neben einer präzisen Beschreibung von Form und Zustand der umgebenen Objekte, ist auch ein immer besseres semantisches Verständnis der Situation erforderlich. Die Anforderungen der funktionalen Sicherheit führen dazu, dass diese oft nur durch Redundanz erreicht werden können.

Für Systeme zur semantischen Klassifizierung von Objekten werden heutzutage hauptsächlich optische Sensoren verwendet. Obwohl die Anzahl der im Fahrzeug verbauten Radare immer weiter steigt und die Messeigenschaften jedes einzelnen Sensors sich in Auflösung, Genauigkeit und Sensitivität beständig verbessern, leistet der Radarsensor heutzutage nur einen geringen Beitrag zur Semantik.

Im Rahmen dieser Arbeit wird das Potential des Radarsensors zur semantischen Umgebungserfassung im Fahrzeugumfeld untersucht. Während es schon einige Arbeiten gibt, welche die Klassifizierung von dynamischen Objekten behandeln, wird in dieser Arbeit der Fokus auf die statische Umgebung gelegt.

Ausgehend von Radar-Detektionen stellt diese Arbeit die gesamte Verarbeitungs- und Entwicklungskette eines Klassifikationssystems dar. Sie zeigt Methoden auf, die Radardaten über die Zeit in Radar-Merkmalskarten aggregieren. Um Algorithmen zu entwickeln und auszuwerten, wurde ein Datensatz erstellt der mehrere Stunden von realen Situationen im Straßenverkehr beinhaltet. Dieser wurde mit Hilfe einer eigens entwickelten Methode der manuellen Annotation gelabelt.

Anhand dieses Datensatzes werden die Eigenschaften verschiedener Objektklassen im Radar beleuchtet und eine sinnvolle Klasseneinteilung entwickelt. Diese Klasseneinteilung stellt die Grundlage für das Training und die Auswertung der Klassifikationsalgorithmen dar. Zur Bestimmung der Semantik werden sowohl Cluster-basierte Klassifikationsalgorithmen eingesetzt, wie auch Methoden der Semantischen Segmentierung. Bei den Cluster-basierenden Algorithmen extrahiert ein Clustering Algorithmus Objektvorschläge, die anschließend klassifiziert werden. Im Gegensatz

dazu, klassifiziert ein Algorithmus der Semantischen Segmentierung direkt auf Basis der Merkmalskarten.

Mit Hilfe dieser Algorithmen konnte beispielsweise gezeigt werden, dass ein mit Bildern vortrainiertes Neuronales Netz eine deutlich bessere Performance erzielt, als ein rein auf Radardaten trainiertes Netz. Zur weiteren Verbesserung der Klassifikationsergebnisse wird ein System vorgestellt, welches durch Genetische Algorithmen die Augmentierungsparameter der Trainingsdaten während des Trainings optimiert.

Zum Abschluss werden Systeme zur Klassifikation aller Klassen (Fahrzeug, Gebäude, Pfosten, Bordstein und Andere) vorgestellt. Anhand dieser konnte gezeigt werden, dass die Klassifikation der statischen Welt vielversprechende Ergebnisse erzielt und Radar damit wertvolle Beiträge zur Modellierung der Fahrzeugumgebung leisten kann. Zudem konnte gezeigt werden, dass eine Schätzung der Orientierung und Dimension von teilweise observierten Fahrzeugen möglich ist, welches zum Beispiel bei Automatisierung von Parkmanövern einen großen Mehrwert bietet.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Autonomes Fahren ist neben dem elektrischen Antrieb, eines der größten Forschungsgebiete im Bereich des Automobils. Dabei setzt sich ein System zum autonomen Fahren im Wesentlichen aus folgenden Komponenten zusammen: Die Umgebungswahrnehmung, welche ein Umgebungsmodell des Fahrzeuges mithilfe der Sensoren, Karten, etc. erstellt. Einem Modul zur Entscheidungsfindung, welches anhand des Umgebungsmodells die kurz- und langfristige Planung der Fahraufgabe erstellt und die Fahrzeugregelung, welche den geplanten Fahrpfad umsetzt [Mar+17].

Für jede dieser Komponenten ist neben der Funktionalität, besonders die Zuverlässigkeit von großer Bedeutung. So gehen beispielsweise Martin et al. [Mar+17] davon aus, dass die Wahrscheinlichkeit des Systemversagens zumindest unter $10^{-5}\,\mathrm{h}^{-1}$ sein sollte. Anders als bei Fahrerassistenzsystemen, bei dem der Mensch als Rückfallebene zur Verfügung steht, muss ein autonomes Fahrsystem nicht nur Fail-Safe, sondern Fail-Operational sein, um das Fahrzeug bei dem Ausfall eines Teilsystems in einen sicheren Zustand überführen zu können, z. B. das Halten an einem sicheren Ort. Dies kann in den meisten Fällen nur mithilfe von redundanten Systemen erreicht werden [Mau+15].

Ganz besonderes bei der Umgebungserfassung und der Entscheidungsfindung ist dies aufgrund der Komplexität des Straßenverkehrs, mit seinen vielen verschiedenen Verkehrsteilnehmern, eine besondere Herausforderung.

Bei heutigen Fahrerassistenzsystemen, wird die Umgebung durch Radar, Ultraschall und Kamera wahrgenommen. Zusätzlich zu diesen weitverbreiteten Sensoren kamen

in der letzten Zeit neue Sensoren hinzu. Hierzu gehört u. a. der Laserscanner und die Time-of-flight Kamera [WHW09].

Traditionell teilen sich die Sensoren die Aufgaben. Der Ultraschall ist hauptsächlich für die Entfernungsmessung im Nahbereich zuständig, dies wird beispielsweise für Parksysteme eingesetzt. Der Radarsensor hingegen misst Position und Geschwindigkeit im mittleren und weiten Entfernungsbereich. Zudem wird er aufgrund seiner Robustheit gegenüber Wetterbedingungen gerne für sicherheitskritische Systeme wie z. B. Notbremsassistenten und Abstandsregelung verwendet.

Für alle Aufgaben die mit einer semantischen Erfassung der Umgebung zu tun haben, wie z. B. die Schildererkennung oder Fahrspurerkennung wird traditionell die Kamera verwendet. Zum einen, da die Kamera unserer menschlichen Wahrnehmung am ehesten entspricht, zum anderen, weil in diesem Bereich schon viel Forschung betrieben wurde.

Mit steigendem Automatisierungsgrad der Fahrfunktion bekommt die Semantik für die Entscheidungsfindung eine immer größere Bedeutung. Zudem kann dadurch die Zuverlässigkeit einzelner Systeme erhöht werden.

So kann beispielsweise die Lokalisierung des Fahrzeugs auf einer Karte durch semantische Informationen verbessert werden, da dadurch Landmarken besser einer Karte zugeordnet werden können. Zudem kann die Zuverlässigkeit von Landmarken besser bewertet werden, indem bewegliche Objekte, wie beispielsweise parkende Autos, nicht als solche verwendet werden.

Durch klassenspezifische Bewegungsmodelle lässt sich das Tracking von bewegten Objekten verbessern. Dazu ist es notwendig eine semantische Klassenzuordnung zu treffen. Zudem können aufgrund einer semantischen Beschreibung der Umgebung verschiedene Parameter, wie die Entstehungswahrscheinlichkeit von Objekten verbessert werden. Parkende Autos können beispielsweise losfahren oder Menschen aus ihnen aussteigen.

Durch eine Erkennung z. B. des befahrbaren Bereichs, kann das Verhalten anderer Fahrzeuge besser vorausgesagt werden.

Durch ein gutes Umgebungsmodell können auch gefährliche Situationen besser erkannt werden, indem das Verhalten von Verkehrsteilnehmern besser vorausgesagt wird. So muss z. B. bei einem Fußgänger, der sich dem Bordstein nähert berücksichtigt werden, dass er auf die Straße treten könnte oder bei einer Reihe parkender Autos, dass Menschen aus dieser Verdeckung auf die Straße treten.

Auch um menschliches Verhalten nachzuahmen hat die Semantik eine große Bedeutung. Wenn man beispielsweise einen Parkplatz nutzen will, bei dem die einzelnen

Parklücken nicht fest vorgegeben sind, muss sich das Fahrzeug an den schon geparkten Fahrzeugen orientieren.

Obwohl schon mehrere Generationen von Radarsensoren im Automobil verbaut wurden und diese mittlerweile in fast jedem neuen Fahrzeug zu finden sind, wird er heutzutage immer noch hauptsächlich als Distanz und Geschwindigkeitsmesser gesehen. Die semantische Wahrnehmung der Umgebung wird immer noch von der Kamera dominiert.

Trotz der geringen Verbreitung von Laserscannern, gibt es mehr Arbeiten bezüglich der Semantik von Objekten, als beim Radar. Dies lässt sich darauf zurückführen, dass der Laserscanner von den Messeigenschaften der Kamera näher ist. Des Weiteren sind Laserscanner mit einer Punktwolken-Schnittstelle auf dem Markt leichter verfügbar, als Radarsensoren mit einer ähnlichen Schnittstelle.

Das Potenzial der mittlerweile hochauflösenden Radarsensoren, wird dabei für die Semantik nicht vollständig genutzt. Für zukünftige Systeme gilt es dies zu ändern. Damit kann man z. B. für die Absicherung einen redundanten semantischen Sensor zur Verfügung zu stellen. Hierbei wird nicht nur ein weiterer Sensor, sondern auch zusätzliches Messprinzip zur Verfügung gestellt, welches die Ergebnisse anderen Sensoren verifizieren kann. Des Weiteren kann der Radarsensor auch Objekte in weiterer Entfernung und Objekte, die in der Kamera teilweise verdeckt sind, wahrnehmen.

Im Moment gibt es nur sehr wenige Arbeiten zur semantischen Umgebungswahrnehmung mit Radarsensoren. Die wenigen bestehenden, beziehen sich hauptsächlich auf das Erkennen von dynamischen Objekten, welches sich zu großen Teilen auf das Doppler-Profil stützt.

Ziel der Arbeit ist es, das Potenzial des Radars zur semantischen Klassifikation von statischen Objekten zu untersuchen. Hierzu wurden mehrere Teilaspekte entwickelt und untersucht:

- Geeignete Zwischenrepräsentationsebenen zur Akkumulation von Radardaten, um die Objekteigenschaften über die Zeit in Rasterkarten abzubilden.
- Methoden zum Labeln von Radar-Rasterkarten.
- Erstellung eines Datensatzes und statistische Untersuchungen bezüglich der Eigenschaften von Objekten in Radardaten.
- Cluster basierende Klassifikation.
- Zellen basierende Klassifikation mithilfe von Methoden der semantischen Segmentierung.

Die Arbeit ist in mehrere Kapitel gegliedert, die jeweils einen Themenbereich abschließend behandeln. Jedes dieser Kapitel strukturiert sich nach Stand der Technik, Vorstellung der Methode mit einer abschließenden Evaluation oder einem qualitativen Einblick in die Ergebnisse.

Im Kapitel 2 wird der Radarsensor mit seinen Eigenschaften zur Klassifikation vorgestellt. Des Weiteren wird darauf eingegangen, wie diese Informationen in Rasterkarten akkumuliert werden können.

Im Kapitel 3 wird auf das manuelle Labeling von Radardaten und dessen Schwierigkeiten eingegangen und die Qualität der Label evaluiert.

Anschließend werden im Kapitel 4 die daraus erzeugten Datensätze beleuchtet und quantitativ dargestellt.

In den darauffolgenden zwei Kapiteln wird auf die Klassifikation auf Basis von Radar-Belegungskarten eingegangen.

Zuerst werden in Kapitel 5, mithilfe von extrahierten Clustern, einige Aspekte untersucht. Dabei wird die Klasseneinteilung betrachtet, verschiedene Radar-Merkmalskarten untersucht und gezeigt, wie Augmentierungstechniken das Klassifikationsergebnis verbessern können. Des Weiteren wird auf klassische und tiefe Klassifikationsverfahren eingegangen und abschließend ein Ensemble aus beiden untersucht.

In Kapitel 6 werden Methoden zur semantischen Segmentierung, also das Zellenweise klassifizieren von Radarbelegungskarten, untersucht. Hierbei wird im Besonderen auf die Kostenfunktion eingegangen. Des Weiteren wird untersucht, ob mithilfe von Neuronalen Netzwerken die Ausdehnung unvollständig beobachteter Fahrzeuge geschätzt werden kann.

Zum Abschluss wird in Kapitel 7 ein Resümee gezogen und auf weitere Forschungsfelder hingewiesen.

Kapitel 2

Radargrundlagen und Radar-Merkmalskarten

Um semantische Informationen aus der Fahrzeugumgebung zu bestimmen, müssen diese zuerst mit einer geeigneten Sensorik erfasst werden. Hierbei ist im Automobilbereich neben Ultraschall, Kamera und Lidar, der Radar einer der weitverbreitetsten Sensoren.

Der Radar Sensor hebt sich von den anderen Sensoren besonders durch seine hohe Reichweite, der Möglichkeit die Geschwindigkeit bewegter Objekte direkt zu messen und der Robustheit gegenüber Wetter- und Lichtbedingungen hervor.

Neben diesen positiven Eigenschaften hat die Umgebungserfassung mit Radarsensoren einige Herausforderungen. Dazu zählt beispielsweise Clutter, dies sind Messungen die durch unerwünschte Ziele verursacht werden, beispielsweise Reflektionen von der Straßenoberfläche. Zudem sind der Auflösung und Genauigkeit durch die Bandbreite des Signals und der Größe der Antenne Grenzen gesetzt. Obwohl hochauflösende Verfahren existieren, welche für eine Verbesserung sorgen könnten, sind diese aufgrund ihrer Komplexität und der limitierten Rechenleistung nur eingeschränkt einsetzbar. Zudem ist es nicht leicht vorherzusagen, an welchen Punkten von einem Objekt Radarmessungen entstehen. Von außen betrachtet wirkt dies oft wie ein Zufallsprozess.

Über eine zeitliche Filterung der Daten kann das Clutterverhalten unterdrückt und sowohl die Genauigkeit, als auch die Datendichte verbessert werden. Ein veränder-

ter Betrachtungswinkel durch die Bewegung des Eigenfahrzeugs ermöglicht mehrere Perspektiven. Dadurch wandern die Reflexionszentren auf einem Objekt, welches zu einer besseren Abschätzung der Kontur führt. Eine gängige Möglichkeit Messdaten zu akkumulieren sind hierbei Belegungskarten. Für viele Probleme in der Robotik kann die Welt als ebene Fläche betrachtet werden [BHB16], daher kann hierfür meist eine 2-D Repräsentation eingesetzt werden.

Dieses Kapitel beleuchtet die grundlegende Verarbeitung von Radardaten bis hin zu der Belegungskarten. Zunächst wird ein Einblick in die Radargrundlagen und der prinzipiellen Funktionsweise der Signalverarbeitungsalgorithmen gegeben. Wobei der Fokus auf den für die Klassifikation relevanten Eigenschaften gelegt wird. Danach wird eine Übersicht über die Belegungskarten gegeben. Anschließend werden die verwendeten Methoden zur Erzeugung von Radar-Merkmalskarten motiviert und beschrieben. Zum Abschluss wird auf bestimmte Eigenschaften der vom Radar beobachteten Objekte eingegangen und diese mithilfe von realen Messdaten dargestellt.

2.1 Grundlagen der Radartechnik

In diesem Kapitel werden die Eigenschaften des Radarsensors, inklusive der grundlegenden Signalverarbeitungsalgorithmen beleuchtet. Für tief greifende Informationen wird auf einschlägige Literatur verwiesen [Sko08; RSH10; BT08].

Radar (*engl. Radio Detection and Ranging*) ist eine Technologie bei dem die Entfernung, Winkel und Geschwindigkeit eines Objektes mithilfe von elektromagnetischen Wellen bestimmt wird. Es basiert auf dem Prinzip der Ausbreitung von elektromagnetischen Wellen und deren Reflexion, an den zu betrachteten Objekten. 1904 patentierte Christian Hülsmeyer das „Verfahren, um entfernte metallische Gegenstände mittels elektrischer Wellen einem Beobachter zu melden“ [Hül04]. Die darauffolgende Entwicklung konzentrierte sich hauptsächlich auf den militärischen Bereich. Hierbei wurde zuerst die Detektion von Flugzeugen und Schiffen in weiten Entfernungen betrachtet. Von da an, begann die Verbreitung in andere Bereiche. Ab 1941 wird Radar beispielsweise zur Fernerkundung in der Meteorologie eingesetzt [Gre11]. Später wurde Radar in der Astronomie zur Kartierung von Planeten genutzt.

Im Straßenverkehr wurde der Sensor zuerst nur zur Verkehrsüberwachung genutzt, z. B. zur Überwachung von Geschwindigkeitsbeschränkungen [The49]. Hierbei machte man sich die Eigenschaft zu Nutze, die Geschwindigkeit direkt mittels des Dopplereffekts zu messen. Erst 1998 war der Radarsensor zum ersten Mal in einem Fahrzeug zu finden [WHW09]. Dieses wurde für die Adaptive Regelung der Geschwindigkeit eingesetzt. Von dieser Zeit an, sind sowohl die Marktdurchdringung, wie auch die

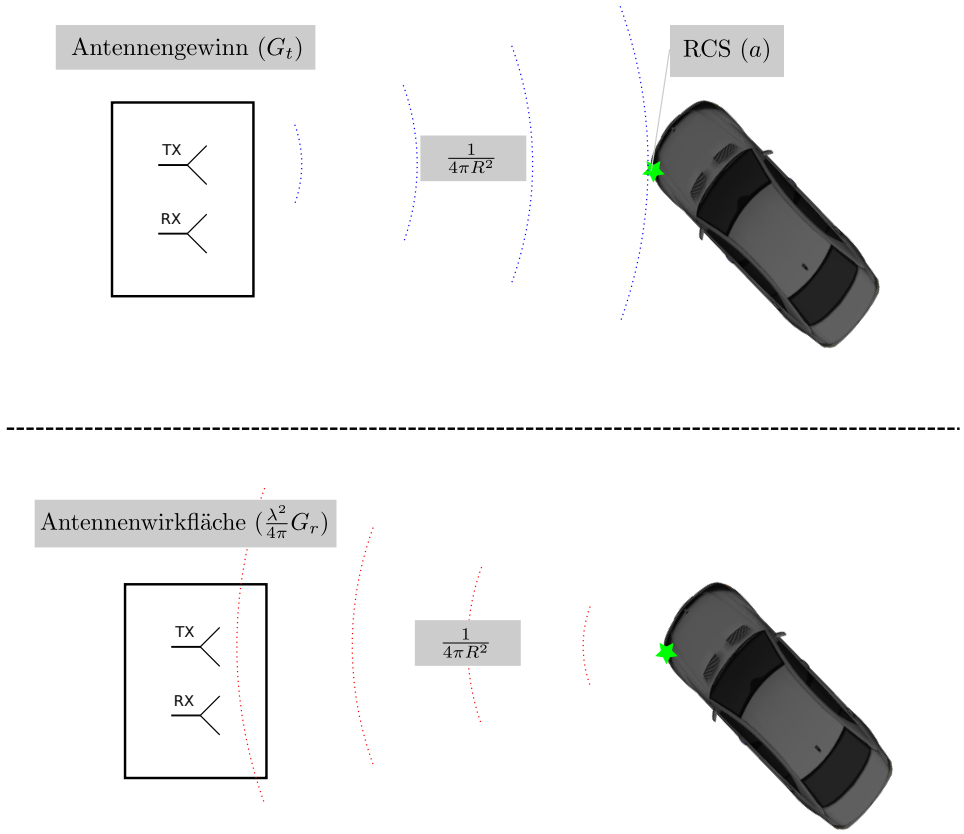


Abbildung 2.1: Radar Übertragungsweg. Für die anschauliche Betrachtung wird davon ausgegangen, dass das Fahrzeug nur ein Reflexionszentrum (Grüner Stern) besitzt.

Anzahl der Radarsensoren pro Fahrzeug gestiegen. Immer neue Einsatzgebiete erhöhen die Anforderungen an den Radarsensor und dessen Signalverarbeitung.

In Abbildung 2.1 ist die grundlegende Funktionsweise eines Radarsensors zu sehen. Im oberen Bereich ist der Sendeweg zum Objekt dargestellt, im unteren Bereich wird der Rückweg, vom Objekt zum Sensor dargestellt. Anhand dieser soll die Radargleichung erklärt werden, diese ist in Gleichung 2.1 dargestellt [Sko08]. Die atmosphärische Dämpfung wird aufgrund der geringen Distanz vernachlässigt [Li+01].

$$P_r = P_t G_t \frac{G_r \lambda^2}{4\pi} \frac{1}{4\pi R_t^2} \frac{1}{4\pi R_r^2} \check{\sigma} \tag{2.1}$$

Im Radarsensor wird zunächst ein elektromagnetisches Signal mit einer bestimmten Energie P_t erzeugt. Dieses wird mit einer Antenne abgestrahlt. Dabei gibt der Antennengewinn G_t die Richtwirkung und den Wirkungsgrad der Antenne an und bestimmt somit die in Hauptrichtung abgestrahlte Leistung [App79].

Die elektromagnetische Welle breitet sich wie in Abb. 2.1 zu sehen, mit nahezu Lichtgeschwindigkeit als Wellenfront aus. Ein Teil der ausgestrahlten Energie trifft somit nach der Entfernung R_t auf ein Objekt der Umgebung, hierbei nimmt die Leistungsdichte quadratisch zum Abstand ab. Das angestrahlte Objekt reflektiert einen Teil der empfangenen Energie in Richtung des Sensors, diese Reflexionsgrad wird durch die Radar-Cross-Section (RCS) $\check{\sigma}$ beschrieben. Andere Teile der Energie werden entweder absorbiert oder in eine andere Richtung weg reflektiert. Das Richtung Sensor reflektierte Signal, wird wiederum als Wellenfront zurückgestrahlt und wird zu Teilen nach der Entfernung R_r von der Empfangsantenne empfangen. Abhängig von der Antennenwirkfläche $A_w = \frac{\lambda^2}{4\pi} G_r$ wird das resultierende Restsignal im Radarsensor verarbeitet.

Aus der Radargleichung ergibt sich die Stärke des rückgestreuten Signals. Ob ein Objekt mit dem Radarsensor detektiert werden kann, hängt unter anderem von dem Signal-Rausch-Verhältnis ab, welches das Verhältnis von der Leistung des empfangenen Signals zur Rauschleistung im System darstellt [Roh83].

In einem realen Fahrzeugumfeld sind in der Regel mehr als ein Objekt im Sichtbereich des Radarsensors. Daher besteht das gemessene Signal in einer Überlagerung sämtlicher rückgestreuter Signale [Swe60]. Mithilfe von Signalverarbeitungsalgorithmen werden die Reflexionszentren extrahiert und deren Entfernung, Winkel und Geschwindigkeit bestimmt.

Radare können aufgrund der ausgesendeten Signalform in Impulsradare und Dauerstrichradare unterteilt werden. Beim Impulsradar wird ein kurzes moduliertes Signal erzeugt und ausgestrahlt [Lud98]. Danach wird der Radarsensor auf den Empfangsmodus umgestellt, um das Signal zu empfangen. Über die Laufzeit des Signals kann dann die Entfernung bestimmt werden.

Beim Dauerstrichradar wird konstant ein Signal ausgesendet und simultan empfangen [Det89]. Durch die kontinuierliche Abstrahlung des Dauerstrichradars, wird eine deutlich geringere Spitzenenergie benötigt als beim Impulsradar, was sich in preislich günstigeren Sensoren bzw. höheren Reichweiten widerspiegelt.

Im Automobilbereich hat sich eine Unterform der Dauerstrichradars etabliert. Das Frequenzmodulierte Dauerstrichradar (FMCW-Radare *engl. frequency modulated continous wave radars*) [Bro05], welches die Frequenz des Modulierten Signals über die Zeit verändert, siehe Abb. 2.2. Durch diese wird das Bestimmen der Entfernung ermöglicht.

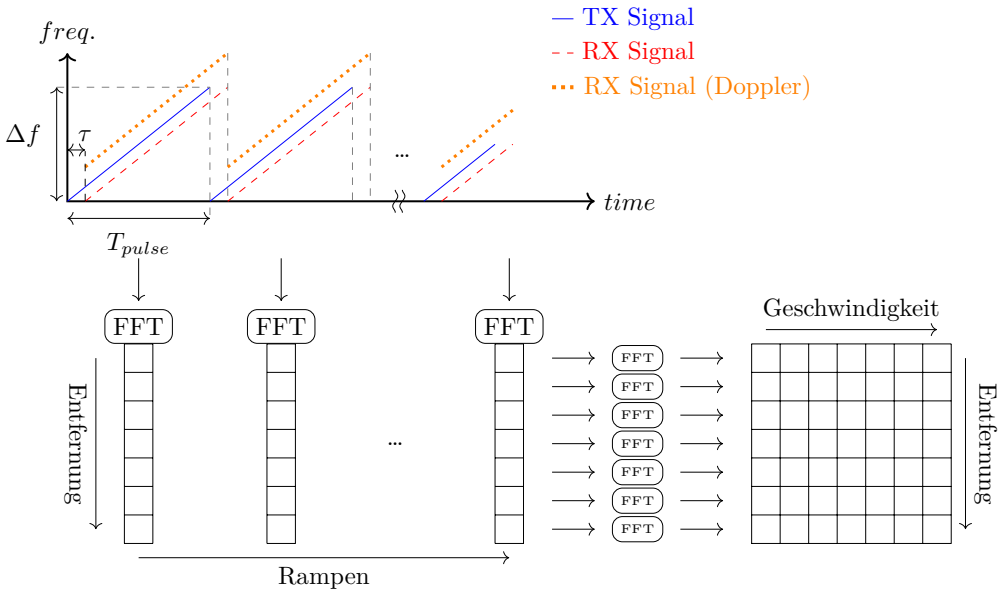


Abbildung 2.2: Frequenz Rampe eines Chrip-Sequence-Radars. Im oberen Teil des Bildes, ist die ausgesendete Frequenzrampe über die Zeit abgebildet. Zusätzlich wird die zugehörige empfangene Frequenzrampe RX eines Ziels in einer bestimmten Entfernung dargestellt. Falls dieses Ziel eine relative Geschwindigkeit zum Sensor besitzt, wird die Rampe durch die Doppler-Geschwindigkeit vertikal verschoben. Dieses ist durch das Signal RX (Doppler) dargestellt. Unten links wird die Bestimmung der Entfernung und nach rechts die Bestimmung der Geschwindigkeit visualisiert.

2.1.1 Entfernung- und Geschwindigkeitsmessung

Durch die Laufzeit des Radarsignals kann die Entfernung von Objekten bestimmt werden. Beim Pulse-Doppler Radar wird ein Radarimpuls zum Zeitpunkt t_{sende} ausgesendet. Dieses Signal wird vom Streuzentrum des Objekts reflektiert und die Restenergie zum Zeitpunkt t_{empfang} vom Sensor empfangen. Die Entfernung r des beobachteten Objektes kann mithilfe der Ausbreitungsgeschwindigkeit der Radarwelle c bestimmt werden. Der Zeitunterschied ergibt sich hierbei aus der Wegstrecke die das Radarsignal zurücklegen muss. Diese setzt sich aus dem Hinweg vom Radar zum Objekt und dem Rückweg vom Objekt zum Radar zusammen. Wenn das Signal auf direktem Weg übertragen wird und die Positionsänderung des Objekts während der Signallaufzeit vernachlässigt werden kann, ist die Entfernung des Hin- und Rückweges gleich. Die Entfernung lässt sich daher wie folgt bestimmen:

$$r = \frac{c}{2}(t_{\text{empfang}} - t_{\text{sende}}) \tag{2.2}$$

Die Geschwindigkeitsmessung erfolgt über die Frequenzverschiebung. Das Radar Signal wird mit der Frequenz f_D ausgesendet und mit der Frequenz f_C empfangen. Daraus kann die Relativgeschwindigkeit des Reflexionszentrums zum Sensor bestimmt werden. Hierbei wird jedoch nur die Geschwindigkeit in radialer Richtung zum Sensor gemessen.

$$v_{\text{rel}} = -\frac{f_D * c}{2f_C} \tag{2.3}$$

Die Funktionsweise eines FMCW-Radars ist ähnlich. Hierbei wird die Entfernung wie in Abb. 2.2 dargestellt, über die horizontale Verschiebung der Rampen und die radiale Geschwindigkeit über die vertikale Verschiebung der Rampen bestimmt.

Dieses kann durch eine zweistufige schnelle Fourier-Transformation (*engl. Fast Fourier Transformation*) (FFT) ausgewertet werden. Durch die kurze Rampendauer des *Chirp* ist die Doppler-Verschiebung durch die Relativgeschwindigkeit des Objektes klein und kann dadurch zunächst vernachlässigt werden. Daher wird die Entfernung innerhalb einzelner Rampen ausgewertet. In einem zweiten Schritt, kann danach die Geschwindigkeit über die Frequenzverschiebung der aufeinanderfolgenden Rampen über die Zeit bestimmt werden.

Der maximal mögliche Messbereich ist hierbei durch die Dauer der einzelnen Rampe begrenzt. Die Auflösung hingegen verbessert sich, je steiler die Flanke des Frequenzanstiegs ist. Da die Bandbreite eines Radars beschränkt ist, gilt es zwischen Flankensteilheit und der Länge des Chirps einen geeigneten Kompromiss zu finden. Weitere Methoden um die Auflösung zu verfeinern sind hochauflösende Verfahren und Verfahren um die Mehrdeutigkeit in der Entfernung aufzulösen [KLK02; SY07].

2.1.2 Winkelmessung

Zur Bestimmung der Position des Rückstreuzentrums im Raum, wird neben der Entfernung der Winkel benötigt. Hier lassen sich die Verfahren in zwei Kategorien einteilen: Dem eigentlichen Schwenken der Antennenkeule und die digitale Strahlformung (DBF *engl. Digital Beam Forming*)[Bar80].

Das Schwenken der Antennenkeule erfolgt entweder mechanisch oder elektronisch. Ersteres kann durch Schwenken der Antenne oder einer Strahlablenkeinheit realisiert werden. Zweiteres erfolgt mittels einer elektronisch steuerbaren Gruppenantenne (*engl. Electronically Steerable Array*) bei der mehrere Einzelantennen mit unterschiedlichen Phasen angesteuert werden.

Beim DBF wird dies mit mehreren Empfangsantennen gelöst, die separat digitalisiert werden. Durch diese können nachfolgende Signalverarbeitungsalgorithmen die Antennenkeule im Nachhinein virtuell schwenken.

In dieser Arbeit beschränkt sich die Winkelbestimmung auf den Azimut Winkel. Dieser beschreibt den Winkel zwischen der x-Achse des Sensors und des Objektes in der horizontalen. Es zeigt sich jedoch, dass verschiedene Anwendungsfelder, darunter auch die Klassifikation von einer Höhenschätzung, d. h. der Winkel in Elevation, profitieren könnte. Bei der eingesetzten Sensorik haben die vertikale Ausdehnung und Position nur indirekt, durch die Breite der Antennenkeule in Elevation, Einfluss.

2.1.3 Antennenöffnungswinkel in Elevation

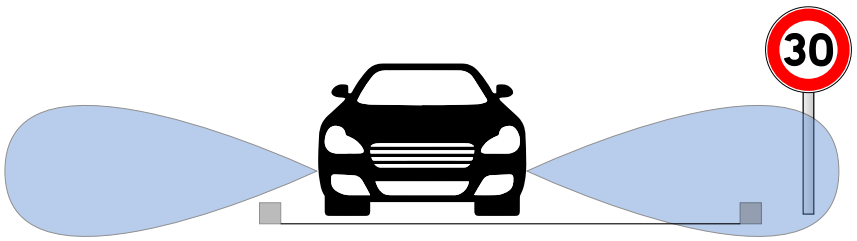


Abbildung 2.3: Antennenöffnungswinkel in Elevation.

Nur wenige am Markt verfügbare Systeme bestimmen einen Elevationswinkel der Reflexionszentren. Auch wenn dies für viele Applikationen wünschenswert wäre, z. B. Hindernisvermeidung, Parken, etc., steht diese Entwicklung bei Automotive Radaren derzeit noch am Anfang. Neben der klassischen Winkelbestimmung, wie sie auch für den Azimut Winkel eingesetzt werden, werden auch alternative Verfahren untersucht, wie z. B. die Winkelbestimmung über Mehrfachreflexionen [Lar+17].

Auch wenn keine direkte Messung der Höhe stattfindet, hat durch die anisotropischen Eigenschaften der Sende- und Empfangsantenne der Elevationswinkel Einfluss, ob ein Objekt detektiert werden kann oder nicht, bzw. welche Teile eines Objektes zurückstrahlen oder nicht. Die Keulenbreite in Elevation beträgt typischerweise zwischen 3° und 10° [WHW09].

Wie in Abbildung 2.3 zu sehen, werden dadurch Objekte nur in bestimmten Entfernungsbereichen detektiert. So sind beispielsweise Randsteine im Nahbereich des Autos nicht mehr im Sichtbereich des Sensors. Ein anderes Beispiel sind Verkehrsschilder, bei denen für einen weiten Entfernungsbereich, nur der Pfosten beobachtet werden kann.

2.1.4 Radarquerschnitt

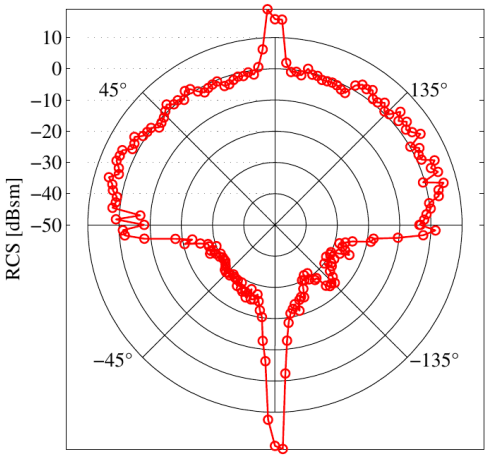


Abbildung 2.4: RCS-Werte eines Verkehrsschildes abhängig vom Betrachtungswinkel, aus [Wer+14].

Um Radarziele vom Hintergrund zu unterscheiden, spielt der Radarquerschnitt (RCS *engl. Radar Cross Section*) eine entscheidende Rolle.

Der Radarquerschnitt bestimmt neben der Entfernung, wie viel Energie des ausgesendeten Signals zum Sensor zurück reflektiert wird. Dieser Radarquerschnitt ist abhängig von Form, Material und Einfallswinkel der Strahlung. Der Radarquerschnitt wird als eine Projektion des Objekts auf eine Fläche einer metallischen Kugel angegeben, welche die gleiche Energiemenge reflektiert [Kno08].

Der Radarquerschnitt gibt dabei die virtuelle Querschnittsfläche an, die ein Objekt haben müsste, wenn es die Welle gleichmäßig in alle Richtungen zurückstreuen würde. Somit entspricht der Radarquerschnitt einer übertragungswegkompensierten Amplitude des empfangenen Signals.

Im Falle eines automobilen Radarsensors wird der RCS-Wert für jedes Reflexionszentrum, welches unter Umständen auch einem Teilobjekt entspricht, geschätzt. Die Rückstreuung wird üblicherweise in $dBsm$ angegeben.

Dieser Wert lässt sich in der Realität oft nur durch Messungen bestimmen. Werber [Wer+14] bestimmt dieses beispielsweise für ein Verkehrszeichen in Abhängigkeit des Betrachtungswinkels in Azimut. Die Ergebnisse sind in Abbildung 2.4 zu sehen, bei der die frontale Ansicht ($-90°$), nur in einem kleinen Winkelbereich, einen hohen RCS aufweist. In allen anderen Bereichen wird das Signal weg reflektiert, während sich die Rückseite des Schildes, zusammen mit dem Pfosten, ähnlich wie ein Tripelspiegel verhält und in einem weiten Winkelbereich ein hohen RCS aufweist.

Tabelle 2.1: RCS-Werte von verschiedenen Objekten im Bereich von 76 GHz–77 GHz.

Objekt	mittlere Radarquerschnitt
Fußgänger	-10 dBsm
Fahrrad	2 dBsm
PKW	20 dBsm
LKW	40 dBsm
Jumbo-Jet	100 dBsm

Yamada et al. [YTN05] untersuchte den Radarquerschnitt von Menschen, dabei wurde auch der Einfluss verschiedener Kleidungsstücke untersucht. Schneider et al. [SBS07] bestimmten durch Messungen den Radarquerschnitt von PKWs in unterschiedlichen Entfernungen und Winkeln. Ebenfalls für PKW entwickelte Schuler[SBW08] ein Simulationsmodell mittels Raytracing und verschiedener virtuellen Scattering-Center. Kärnfeld[Kär+09] zeigte für Fahrzeuge und LKWs ein entfernungsabhängiges Modell der Rückstreuung.

In Tabelle 2.1 sind beispielhaft für einige Objekte der RCS-Wert angegeben.

2.2 Belegungskarten

Belegungskarten sind eine weitverbreitete Methode in der Robotik, um die statische Umgebung zu modellieren und Messdaten über die Zeit zu akkumulieren. Sie wurden erstmals 1986 von Elfes et al. vorgestellt [Elf86]. Der englische Begriff *Occupancy Grid* wurde jedoch erst 1989 geprägt [Elf89].

Belegungskarten ${}^{occ}M$ zerlegen die Welt mithilfe eines Rasters in Zellen ${}^{occ}m_{i,j}$. Es wird angenommen, dass jede Zelle entweder *belegt* oder *frei* sein kann. Diese dis-

kretisierte Welt wird nun probabilistisch behandelt, in dem die a posteriori Wahrscheinlichkeit dieser Belegungszustände geschätzt werden.

Diese Wahrscheinlichkeit wird auf der Grundlage der Sensormessungen $Z = z_0 \dots z_T$ und der Pose des Roboters $X = x_0 \dots x_T$ bestimmt. Bei der Initialisierung wird die Belegungswahrscheinlichkeit einer Zelle üblicherweise auf den Wert 0.5 (*unbekannt*) gesetzt. Danach wird die a posteriori Wahrscheinlichkeit meist mittels dem inversen Sensormodell [Thr02] bestimmt.

$$p({}^{occ}M|z_{0:T}, x_{0:T}) \tag{2.4}$$

Die Problematik dabei ist, dass mit Anzahl der Zellen, die Anzahl der möglichen Karten exponentiell steigt. So kann eine Karte, die einen 100 m x 100 m Ausschnitt der Welt mit einer Rasterung von 0,1 m x 0,1 m unterteilt, aus $\frac{100^2}{0.1^2} = 10e6$ Zellen. Daraus ergeben sich $2^{10e6} \approx 10^{3e6}$ mögliche Belegungszustände.

Mit heutigen Computern ist es nicht möglich, für eine solch hohe Anzahl an Zuständen, die Wahrscheinlichkeitsverteilung in Echtzeit zu bestimmen. Um dieses Optimierungsproblem zu vereinfachen, wird daher meist die Annahme getroffen, dass die Belegungszustände der Zellen unabhängig voneinander sind. Nun kann für jede Zelle separat, die A-Posteriori Wahrscheinlichkeit berechnet werden. Die Komplexität sinkt dabei von einer exponentiellen $\mathcal{O}(2^n)$ zu einer linearen Komplexität $\mathcal{O}(n)$.

$$p({}^{occ}M|z_{0:T}, x_{0:T}) = \prod_i \prod_j p({}^{occ}m_{i,j}|z_{0:T}, x_{0:T}) \tag{2.5}$$

Eine Möglichkeit, um trotzdem die Abhängigkeiten der Zellen zueinander zu modellieren, zeigte Thrun in [Thr03]. Er führte dazu ein *Forward Sensormodell* ein. Das hieraus entstehende Optimierungsproblem wird in dieser Arbeit mittels *expectation maximization* [DLR77] gelöst.

Belegungskarten werden meist genutzt, um die Welt 2-D zu repräsentieren. In einigen Anwendungsfällen werden beispielsweise Höheninformationen benötigt und das Grid auf 2.5-D Grids [MT04; BFP09] oder 3-D Grids [Mor96; TBF00] erweitert.

Die ersten Arbeiten mit Belegungskarten wurden mithilfe des Ultraschallsensors durchgeführt und für Indoor Navigation genutzt [Elf86; Elf89]. Mittlerweile ist der Laserscanner für Belegungskarten der dominierende Sensor [WSD07; Thr98]. Durch die geringe Strahlaufweitung lässt sich hierfür auch leichter ein Freiraummodell erstellen.

Weitere Arbeiten verwenden auch Stereokameras [Sab+04; JM97] und/oder optischer Fluss [Bra+06] zur Erstellung von Belegungskarten.

Konolige [Kon97] entwickelte ein Sensormodell für Ultraschallsensoren, bei dem Spiegeleffekte genauer betrachtet werden. Andert zeigte ein Sensormodell für eine Stereokamera [And09]. Fossel et. al. [Foe01] entwickelten ein Sensormodell für den Radarsensor, mittels der Interpretation des Abstand-Amplitude-Diagramms.

Degerman et al. [DPA16] stellten ein Sensormodell des Radars vor, dass die Detektionswahrscheinlichkeit über den Swerling 1 Fall [Swe60] berechnet. Für die Freiraummodellierung wurde ein heuristisches Modell gewählt.

In [Wer+15b] wurden zwei unterschiedliche Radarkarten zur Selbstlokalisation verglichen. Hierbei handelte es sich zum einen um eine klassische Belegungskarte. Zum anderen wurde eine Karte anhand der RCS-Werte aufgebaut, die somit den RCS-Wert der Zellen über die Zeit mitteln, wobei die unterschiedlichen Messungen über den Abstand gewichtet werden.

Prophet et al. stellte ein erweitertes Freiraummodel für den Radarsensor vor [Pro+18].

Alternativ zur bayesschen Wahrscheinlichkeitstheorie stellt Pagac [PND98] einen auf der Evidenztheorie basierenden Ansatz von Belegungskarten vor. Ribo [RP01] vergleicht diese mit dem Bayesschen Ansatz von Belegungskarten anhand von Ultraschallsensoren.

Schütz et al. [Sch+14b; Sch+14a] setzte für dynamische Objekte lokale Belegungskarten ein. Diese bewegen sich mit dem verfolgten Objekt mit und ermöglichen es daher, zusätzlich zum Bewegungsstatus, die Ausdehnung zu schätzen. Andere Ansätze erweitern Belegungskarten um dynamische Objekte [Ric+09; DON11; Nus+18]. Bei diesen wird für jede Zelle auch die Geschwindigkeit mit geschätzt. Dies ermöglicht eine Migration des Belegungszustandes über die Zeit.

Mittlerweile werden Belegungskarten für zahlreiche Anwendungen eingesetzt. Angefangen bei der Erkennung von Freiraum und dessen Begrenzungen, durch die eine Pfadplanung ermöglicht wird [Elf89], über die Vermeidung von Hindernissen [BK91]. Teilweise werden auch verschiedener Sensorkonzept wie Lidar und Radar [Aht+15] fusioniert.

Selbstlokalisierung [Wer+15a; Rap+16b] zeigten den erfolgreichen Einsatz von Radar basierenden Karten.

Elizar et al. [EP03] entwickelten eine Methode für Simultaneous Localization and Mapping (SLAM) die auf Belegungskarten arbeitet, ohne vorher Landmarken zu extrahieren. Schön et al. [Sch+17a] zeigten eine SLAM Methode welche echzeitfähig

auf Radardaten funktioniert. Lupfer et al. passten diese Methode an, indem sie zusätzlich die gemessene Doppler-Geschwindigkeit in das SLAM Problem integrierten [Lup+17].

Verschiedene Arbeiten zeigen auch die Möglichkeit im Straßenverkehr, den Fahrbahnverlauf zu schätzen [DKL10; KSD10; Sar+11; Gie+17].

2.3 Koordinatensysteme und Messdaten

In diesem Abschnitt werden die in der Arbeit verwendeten Symbole und Begrifflichkeiten definiert. Hierzu werden zunächst verschiedene Koordinatensysteme eingeführt und danach auf die Messgrößen eingegangen.

2.3.1 Koordinatensysteme

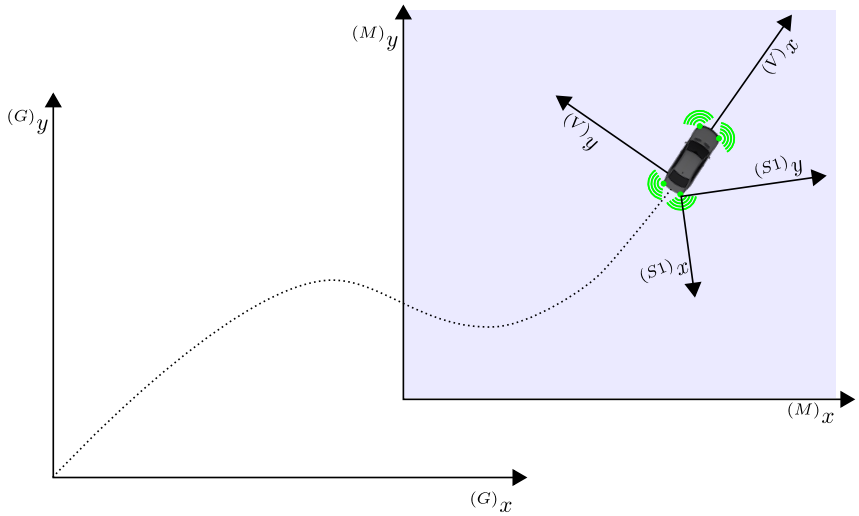


Abbildung 2.5: Definition der in der Arbeit verwendeten Koordinatensysteme. Das Ortsfeste Koordinatensystem ($^{(G)}$), das Karten-Koordinatensysteme ($^{(M)}$), das Fahrzeugfeste Koordinatensystem ($^{(V)}$) und beispielhaft eine Sensorkoordinatensystem ($^{(S1)}$). Die gefahrene Trajektorie ist exemplarisch durch die gestrichelte Line eingezeichnet.

Die in der Arbeit verwendeten Koordinatensysteme sind in Abbildung 2.5 dargestellt. Es wird zwischen dem Ortsfesten- ($^{(G)}$), dem Karten- ($^{(M)}$), dem Fahrzeug- ($^{(V)}$) und den Sensorkoordinatensystemen ($^{(S1)}$,$^{(S2)}$,...,$^{(Sn)}$) unterschieden.

Das Ortsfeste ist ein in der Welt verankertes Koordinatensystem, welches, falls das Fahrzeug über eine GPS gestütztes Inertialsystem verfügt, dem UTM-Koordinatensystem (Universal Transverse Mercator Koordinatensystem) [HBD89] entspricht, andernfalls wird der Nullpunkt und die Ausrichtung am Anfang einer Messung frei gewählt und während einer Messsequenz festgehalten.

Das Fahrzeug feste Koordinatensystem hat seinen Ursprung im Drehzentrum des Fahrzeuges, welcher sich nahe der Hinterachse befindet. Die Ausrichtung des Koordinatensystems entspricht DIN70000 [DIN94], die x-Achse zeigt nach vorne, bei gerader Fahrt in Fahrtrichtung. Die y-Achse ist nach links ausgerichtet. Die Verschiebung und Drehung zwischen dem Ortsfesten- und dem Fahrzeugkoordinaten ist zeitabhängig und wird durch die Translation $({}^{(G)}x_V(t), {}^{(G)}y_V(t))$ und die Rotation um die z-Achse ${}^{(G)}\gamma_V(t)$ definiert.

Die Belegungskarten werden mit dem Fahrzeug mitgeführt, sodass sich der Sichtbereich des Eigenfahrzeuges stets in der Karte befindet. Um Quantisierungsfehler zu vermeiden, wird das Koordinatensystem nicht mit dem Fahrzeugkoordinatensystem gedreht und eine Verschiebung erfolgt nur um ein ganzzahliges vielfaches der Zellgröße. Somit sind nur die Parameter ${}^{(G)}x_M(t)$, ${}^{(G)}y_M(t)$ zeitabhängig, während die Orientierung ${}^{(G)}\gamma_M(t)$ konstant ist.

Die n Sensorkoordinatensysteme sind um ihre Einbauposition $({}^{(V)}x_{Sn}, {}^{(V)}y_{Sn})$ verschoben und um ${}^{(V)}\gamma_{Sn}$ um die z-Achse des Fahrzeugkoordinatensystems gedreht. Die Messungen des Sensors erfolgen im Polarkoordinatensystem. Es wird keine Onlinekalibrierung der Sensoren durchgeführt, somit ist die Verschiebung gegenüber dem Fahrzeugkoordinatensystem zeitinvariant.

Die Elevation wird von den Sensoren nicht gemessen, damit entfällt die Betrachtung der Höheninformation. Beim Einbau wurden die Sensoren so ausgerichtet, dass der Winkel um die x- und y-Achse etwa 0° beträgt. Daher können diese Winkel vernachlässigt werden.

2.3.2 Eigenbewegung

Die Eigenbewegung des Fahrzeuges wird über interne Gierratensensoren und die Raddrehzahlen bzw. Radticks bestimmt. Durch die Annahme einer ebenen Welt werden Bewegungen in vertikaler Richtung vernachlässigt. Die Position des Fahrzeuges lässt sich durch die Integration von Vorwärtsbewegung ${}^{(V)}v_{x,ego}$ und der Giergeschwindigkeit ${}^{(V)}\omega_{z,ego}$ bestimmen.

Die Vorwärtsbewegung wird durch den Umfang des Rades und der Drehgeschwindigkeit bestimmt. Hierbei wird über alle vier Räder gemittelt.

Die Giergeschwindigkeit wird mithilfe des Gierratensensors des ESP Systems bestimmt. Diese Bestimmung der Grierrate erweist sich bis auf einen Temperaturabhängiges Bias als ziemlich genau. Daher wird der Gierratensensor jeweils bei Stillstand des Fahrzeugs kalibriert, da in dieser Zeit keine Giergeschwindigkeit auftreten kann.

Somit lässt sich der Bewegungszustand des Fahrzeuges durch die Geschwindigkeiten

$$v_{ego} = \begin{pmatrix} {}^{(G)}v_{x,ego} \\ {}^{(G)}v_{y,ego} \\ 0 \end{pmatrix} = \begin{pmatrix} \cos\Omega(t)\,{}^{(V)}v_{x,ego} \\ \sin\Omega(t)\,{}^{(V)}v_{x,ego} \\ 0 \end{pmatrix} \tag{2.6}$$

und die Rotation lediglich mit der Giergeschwindigkeit

$$\omega_{ego} = \begin{pmatrix} 0 \\ 0 \\ \omega_{z,ego} \end{pmatrix} \tag{2.7}$$

beschreiben.

Hiermit kann die Pose des Fahrzeuges zum Zeitpunkt $t+1$ wie folgt bestimmt werden.

$$\begin{aligned} {}^{(G)}\vec{x}_{ego,t+1} &= \begin{pmatrix} {}^{(G)}x_{t+1} \\ {}^{(G)}y_{t+1} \\ {}^{(G)}\Theta_{t+1} \end{pmatrix} \\ &= \begin{pmatrix} {}^{(G)}x_t \\ {}^{(G)}y_t \\ {}^{(G)}\Theta_t \end{pmatrix} + \begin{pmatrix} \frac{{}^{(V)}v_{ego}}{\omega_{z,ego}}(-\sin{}^{(G)}\Theta + sin({}^{(G)}\Theta + {}^{(V)}\omega\Delta t)) \\ \frac{{}^{(V)}v_{ego}}{\omega_{z,ego}}(\cos{}^{(G)}\Theta - cos({}^{(G)}\Theta + {}^{(V)}\omega\Delta t)) \\ \omega_{z,ego}\Delta t \end{pmatrix} \end{aligned} \tag{2.8}$$

Zusätzlich kann für jeden dieser Größen eine Varianz angenommen werden.

2.3.3 Radardaten

Durch die im Radarsensor stattfindende Signalverarbeitung werden aus dem rückgestreuten Signal Reflexionszentren extrahiert und deren Parameter bestimmt. Diese Signalverarbeitungsalgorithmen stammen vom Sensorhersteller und können nicht weiter beeinflusst werden.

Pro Sensorzyklus ermittelt der Radarsensor eine Liste von K Radarzielen.

$$Z_t = (z_{t,0}, z_{t,1}, \ldots, z_{t,K}) \tag{2.9}$$

Ein Radarziel wird durch einen Messvektor ausgedrückt, der den Abstand, den Azimutwinkel, die Doppler-Geschwindigkeit und den Radar-Rückstreuquerschnitt in den Sensorkoordinaten beschreibt.

$$^{(Sx)}z_{t,k} = \begin{pmatrix} r_{t,k} \\ \varphi_{t,k} \\ \dot{r}_{t,k} \\ \check{\sigma}_{t,k} \end{pmatrix} \tag{2.10}$$

Für jeden dieser Messgrößen, mit Ausnahme des RCS-Wertes, wird vom Sensorhersteller eine Varianz angegeben.

$$\mathrm{Var}(^{(Sx)}z_{t,k}) = \begin{pmatrix} \mathrm{Var}(r) \\ \mathrm{Var}(\varphi) \\ \mathrm{Var}(\dot{r}) \end{pmatrix} \tag{2.11}$$

2.3.3.1 Kompensation der Eigenbewegung

Um die Belegungskarten nicht zu "verunreinigen", sollen Messungen von sich bewegenden Objekten nicht in die Karte eingetragen werden. Hierbei wird von der Möglichkeit des Radarsensors, die Doppler-Geschwindigkeit direkt zu messen, profitiert. Dies ist instantan für alle Objekte möglich, wenn diese eine radiale Geschwindigkeitskomponente relativ zum Sensor größer als die Messgenauigkeit besitzen.

Da sich das Ego-Fahrzeug bewegt, wird auch für statische Objekte eine Geschwindigkeit gemessen. Daher muss die Doppler-Geschwindigkeit über Grund bestimmt

werden. Dies geschieht, indem man die gemessene Geschwindigkeit bezüglich der bekannten Eigenbewegung kompensiert. Hierzu muss zunächst die Geschwindigkeit des Sensors bestimmt werden und diese dann auf den gemessenen, radialen Geschwindigkeitsvektor projiziert werden.

Zunächst wird also die Geschwindigkeit des Fahrzeuges an der Position des Sensors bestimmt. Hierbei kann das Fahrzeug als starrer Körper angesehen werden.

Die Bewegung des Sensors lässt sich dabei mithilfe der EULERschen Beziehung bestimmen [RS08]. Hierbei gibt r_{sen} den Abstand vom Sensor zum Fahrzeugkoordinatensystem an.

$$v_{sen} = v_{ego} + \omega_{ego} \times r_{sen} \tag{2.12}$$

Diesen Geschwindigkeitsvektor kann man dann, mithilfe des Einbauwinkels des Sensors ${}^{(V)}\gamma_{Sn}$ und des Azimutwinkel der Messung φ auf die gemessene Doppler-Geschwindigkeit projizieren.

Mit dem Winkel der Messung relativ zum Fahrzeug

$${}^{(V)}\varphi = {}^{(V)}\,\gamma_{Sn} + {}^{(Sn)}\,\varphi \tag{2.13}$$

lässt sich die Rotationsmatrix für eine solche Projektion bestimmen.

$$R = \begin{pmatrix} \cos({}^{(V)}\varphi) & -\sin({}^{(V)}\varphi) \\ \sin({}^{(V)}\varphi) & \cos({}^{(V)}\varphi) \end{pmatrix} \tag{2.14}$$

Damit lässt sich die Geschwindigkeit des Sensors bestimmen:

$$v_{sen} = R(v_{ego} + \omega_{ego} \times r_{sen}) = Rv_{ego} + R(\omega_{ego} \times r_{sen}) \tag{2.15}$$

Für ein Fahrzeug im Normalbetrieb (kein Driften) kann weiter davon ausgegangen werden, dass die Quergeschwindigkeit $v_y^{ego} = 0$ ist. Zusätzlich werden Vertikale, Nick- und Wankbewegungen vernachlässigt. Daraus ergibt sich der Bewegungszustand:

$$\begin{aligned}\begin{pmatrix} v_{r,sensor} \\ v_{t,sensor} \end{pmatrix} = &\begin{pmatrix} \cos({}^{(V)}\varphi) & -\sin({}^{(V)}\varphi) \\ \sin({}^{(V)}\varphi) & \cos({}^{(V)}\varphi) \end{pmatrix} \begin{pmatrix} v_x^{ego} \\ \cancelto{0}{v_y^{ego}} \end{pmatrix} \\ &+ \begin{pmatrix} \cos({}^{(V)}\varphi) & -\sin({}^{(V)}\varphi) \\ \sin({}^{(V)}\varphi) & \cos({}^{(V)}\varphi) \end{pmatrix} \begin{pmatrix} \omega x_{sen} \\ -\omega y_{sen} \end{pmatrix}\end{aligned} \tag{2.16}$$

Löst man das ganze nach $v_{r,sensor}$ auf, erhält man:

$$\begin{aligned}v_{r,sensor} = &\cos({}^{(V)}\varphi) v_x^{ego} + \cos({}^{(V)}\varphi)\omega_{z,ego} x_{sen} \\ &+ \sin({}^{(V)}\varphi)\omega_{z,ego} y_{sen}\end{aligned} \tag{2.17}$$

Damit kann nun die gemessenen Dopplergeschwindigkeit um die Geschwindigkeit des Sensors kompensiert werden:

$$\dot{r}^{(G)} = \dot{r} - v_{r,sensor} \tag{2.18}$$

Nun müssen noch die Varianzen der Messgrößen $\mathrm{Var}(v_x^{ego}), \mathrm{Var}(v_y^{ego}), \mathrm{Var}(\omega^{ego})$ transformiert werden. Dies geschieht mittels Linearkombination der Varianzen [Fah+07] und der Rotation mittel Hilfe der Jacobi Matrix. Hierbei wird davon ausgegangen, dass sämtliche Größen unkorreliert sind. Die Varianzen aller anderen Größen sind im Vergleich dazu vernachlässigbar.

$$\begin{aligned}\mathrm{Var}(v_r) = &\mathrm{Var}(\dot{r}) + \cos({}^{(V)}\varphi)^2 \mathrm{Var}(v_x^{ego}) \\ &+ \cos({}^{(V)}\varphi)^2 x_{sen}^2 \mathrm{Var}(\omega) + \sin({}^{(V)}\gamma_{Sn} + {}^{(Sn)}\varphi)^2 y_{sen}^2 \mathrm{Var}(\omega)\end{aligned} \tag{2.19}$$

Anhand dieser Parameter kann nun eine Aussage über den Bewegungszustand einer Radarmessung getroffen werden.

2.4 Radar-Merkmalskarten

Für die Klassifikation von Objekten mithilfe des Radarsensors müssen deren Radar-typische Eigenschaften erfasst und mit der Zeit akkumuliert werden.

Wie schon in Abschnitt 2.2 beschrieben, sind Rasterkarten insbesondere für die statische Welt geeignet.

Die Belegungskarte eignet sich gut, ein geeignetes Sensormodell vorausgesetzt, um die Belegungen in einer statischen Welt abzubilden. Die hierbei erlangte Information besteht darin, mit welcher Wahrscheinlichkeit sich in einer Zelle Materie befindet, mit Ausnahme dessen, was als Hintergrund angesehen wird, wie z. B. die Fahrbahn.

Belegungskarten stellen dem Klassifikator Informationen zur Form von Objekten zur Verfügung. Zur Klassifikation sind jedoch noch weitere Eigenschaften des Objektes außer der Form hilfreich. Dabei gilt es die Sensor-spezifischen Eigenschaften zu nutzen, um damit das Klassifikationsergebnis zu verbessern.

Wie in Kapitel 2.1 beschrieben, misst der Radarsensor zusätzlich zur Position, die Doppler-Geschwindigkeit und den RCS-Wert, bzw. die Amplitude des rückgestreuten Signals. Das Doppler-Spektrum hat eine große Bedeutung für dynamische Objekte. Bei statischen Objekten hat dieses Merkmal eine untergeordnete Bedeutung.

Der RCS-Wert $\check{\sigma}$ hingegen, enthält Informationen über die Struktur und das Material des Reflexionszentrums und somit auch des zugehörigen Objektes. Dieser ist zudem vom Betrachtungswinkel in Azimut φ und Elevation ϑ (und demzufolge der Entfernung) abhängig.

$$\check{\sigma}(\text{Azimutwinkel}, \text{Elevationswinkel}, \text{Material}, \text{Struktur}) \tag{2.20}$$

Ausgehend von einer flachen Welt und einem kleinen Öffnungswinkel der Antennenkeule in Elevation, ist der Einfluss der Elevation gering. Daher wird dieser Effekt nicht gesondert betrachtet. In Abschnitt 2.5 wird dies anhand einiger Beispiele beleuchtet.

Im Folgenden soll die Belegungskarte vorgestellt werden. Danach werden Konzepte von Rasterkarten entwickelt, die in der Lage sind, Radar-spezifische Eigenschaften über die Zeit zu erfassen.

Abbildung 2.6: Szenario, welches für die Diskussion der verschiedenen Merkmalskarten verwendet wird.

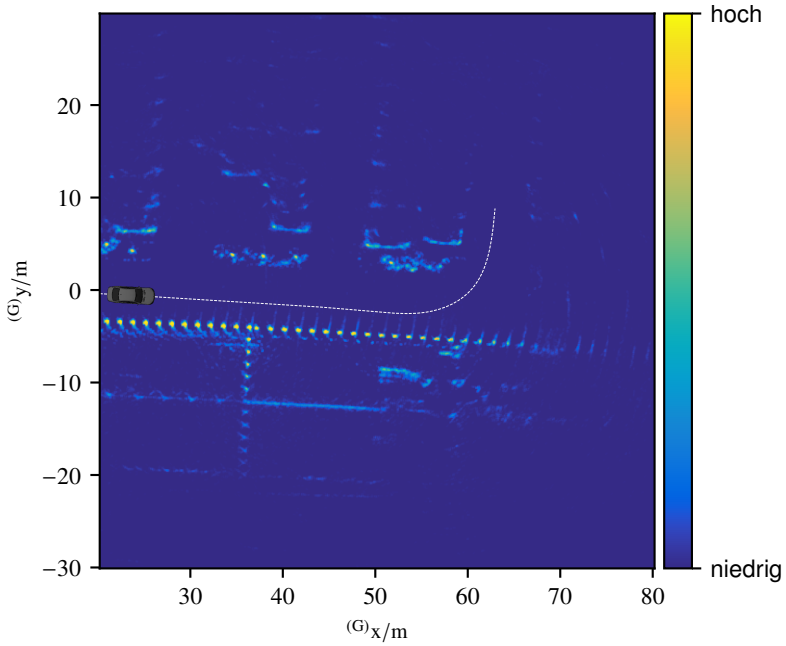


Abbildung 2.7: Beispiel einer Belegungskarte.

2.4.1 Belegungskarte

Die Belegungskarte ${}^{\boldsymbol{occ}}\boldsymbol{M}$ ist eine Karte, welche den Belegungszustand der Welt, also in welchem Bereich der Welt sich Materie befindet, darstellt. Dabei wird die Welt in diskrete Zellen $m_{i,j}$ eingeteilt, sodass der gesamte Bereich der Karte durch diese repräsentiert wird.

$$\operatorname{dom} \boldsymbol{M} = \cup_i \cup_j m_{i,j} \tag{2.21}$$

In der Regel handelt es sich um quadratische Zellen. Wobei der erste Index i die Zelle in x-Richtung und der zweite Index j die Zellen in y-Richtung indiziert. Dabei überschneiden sich die einzelnen Zellen nicht.

$$m_{i,j} \cap m_{k,l} = \varnothing \tag{2.22}$$

Jeder Zelle wird eine Zustandsvariable ${}^{occ}m_{i,j}$ zugeordnet. Diese kann einen der zwei Zustände einnehmen, frei (${}^{occ}m_{i,j} = 0$) oder belegt (${}^{occ}m_{i,j} = 1$). Anhand des inversen Sensormodells kann hierfür die a-posteriori Wahrscheinlichkeit für diese Kartenkonstellation berechnet werden.

$$p({}^{\boldsymbol{occ}}\boldsymbol{M}|z_{0:t}, x_{0:t}) \tag{2.23}$$

Wie bereits in Kapitel 2.2 beschrieben, nimmt die Komplexität dieses Optimierungsproblems exponentiell mit der Anzahl der Zellen der Karte zu. Um diesem Problem zu begegnen, werden für das inverse Sensormodell die Zellen als unabhängig voneinander betrachtet. Damit kann das Optimierungsproblem für jede Zelle getrennt gelöst werden. Die Komplexität steigt damit nur linear mit der Anzahl der Zellen an.

$$p({}^{occ}m_{i,j}|z_{0:t}, x_{0:t}) \tag{2.24}$$

In diesem Fall lässt sich die Wahrscheinlichkeit der ganzen Karte mit dem Produkt der einzelnen Zellen abschätzen.

$$p({}^{\boldsymbol{occ}}\boldsymbol{M}|z_{0:t}, x_{0:t}) = \prod_i \prod_j p({}^{occ}m_{i,j}|z_{0:t}, x_{0:t}) \tag{2.25}$$

Für das Optimierungsproblem wird das binäre Bayes-Filter eingesetzt [Thr02]. Die Grundannahme hierfür ist, dass sich der Belegungszustand der Zelle über den Beobachtungszeitraum nicht ändert. Der Zellenzustand, also *frei* oder *belegt*, bleibt.

Um numerische Instabilitäten zu vermeiden, wird meist die *log-odds* Repräsentation verwendet:

$$l_{i,j,t} = \log \frac{p(^{occ}m_{i,j} = 1 | z_{0:t}, x_{0:t})}{1 - p(^{occ}m_{i,j} = 1 | z_{0:t}, x_{0:t})} \tag{2.26}$$

Diese kann wieder zur Wahrscheinlichkeitsdarstellung überführt werden.

$$p(^{occ}m_{i,j} = 1 | z_{0:t}, x_{0:t}) = 1 - \frac{1}{1 - \exp(l_{i,j,t}})} \tag{2.27}$$

Der verwendete Algorithmus zur Erstellung der Belegungskarte wird in Algorithmus 1 beschrieben. Dabei drückt $N\sigma(z_{k,t})$ den Bereich der N-fachen Standardabweichung aus.

Algorithmus 1 : *Belegungskarten* Update

Data : $\{l_{i,j,t-1}\}$, x_t, $\boldsymbol{z_t}$
Result : $\{l_{i,j,t}\}$

```
foreach cells m_{i,j} ∈ M do
    l_{i,j,t} = l_{i,j,t-1};
    foreach z_{k,t} ∈ z_t do
        if m_{i,j} ∩ Nσ(z_{k,t}) ≠ ∅ then
            l_{i,j,t} = l_{i,j,t} + Inverses Sensormodell(m_{i,j}, x_t, z_{k,t}) − l_0;
        end
    end
end
```

Da diese Karte lediglich für die Klassifikation von Objekten eingesetzt wird und somit lediglich die Form von Objekten in der Belegungskarte relevant ist, kann auf eine Freiraumbestimmung verzichtet werden. Durch die Wahl eines kleinen Kartenausschnittes kann ebenfalls auf eine Degradation der Belegungskarte außerhalb des Sichtbereichs des Sensors verzichtet werden, da der Bereich des Kartenausschnitts nach kurzer Zeit verlassen und bei einer erneuten Befahrung neu initialisiert wird.

Das Sensormodell vereinfacht sich dadurch erheblich und reduziert auch die benötigte Rechenzeit. So können die einzelnen Radar-Targets $z_{t,n}$ getrennt voneinander integriert werden.

Das Sensormodell ergibt sich hierbei aus der Wahrscheinlichkeitsdichte des einzelnen Radar-Targets, mit seiner Distanz r, dem Winkel φ und den zugehörigen Varianzen. Diese wird mithilfe der Jacobi Matrix J vom Polarkoordinatensystem des Sensors in das kartesische Koordinatensystem der Karte überführt.

$$J = \begin{pmatrix} \cos({}^{(V)}\varphi) & -\sin({}^{(V)}\varphi) \\ \sin({}^{(V)}\varphi) & \cos({}^{(V)}\varphi) \end{pmatrix} \tag{2.28}$$

Es ergibt sich dann ein Sensormodell, welches in Abb. 2.8 zu sehen ist. Wobei links das polare Sensormodell und rechts das linearisierte Sensormodell zu sehen ist.

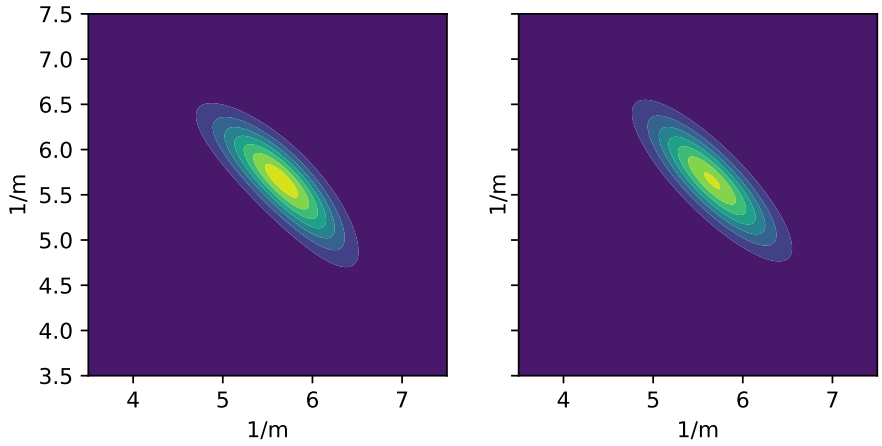


Abbildung 2.8: Sensormodell für eine beispielhafte Messung 8 Meter vom Sensor entfernt. Hierbei wurden die Wahrscheinlichkeitsdichten mittels Sampling in das kartesische Diagramm überführt. Links: Kovarianz des polaren Sensormodells. Rechts: Kovarianz nach Linearisierung.

Hieraus ergeben sich Karten die nur den Belegungszustand, nicht aber den Freiraum erfassen und somit die Zellen den Wahrscheinlichkeitsbereich von 0.5 bis 1 annehmen können.

In Abbildung 2.7 wird eine solche Karte dargestellt, als Referenz ist in Abb. 2.6 das Kamerabild der Szene zu sehen. In ihr lassen sich einige Objekte erkennen. Auf der rechten Seite des Autos ist eindeutig die Pfostenreihe zu erkennen. Von dieser geht eine weitere Pfostenreihe im rechten Winkel zu den Häusern. Diese scheinen sich nach der Gebäudewand fortzusetzen. Dabei handelt es sich im Spiegelobjekte, diese werden in Abschnitt 2.5 diskutiert.

Auf der linken Seite zeichnen sich schemenhaft bepflanzte Inseln ab, die mit einem Bordstein umrandet sind. Auf diesen befinden sich sowohl Bäume, wie auch Straßenlaternen. Diese sind in der Belegungskarte nicht zu unterscheiden. Weiter links zeichnen sich die Formen von Fahrzeugen ab, die je nach Entfernung und Verdeckung mehr oder weniger deutlich sind. Neben den erwähnten Objekten, sind noch andere zu sehen, z. B. Verkaufstische auf der rechten Seite (auf der Position [52, −9]), hier ist es jedoch für den Menschen sehr schwer ein charakteristisches Muster zu erkennen.

2.4.2 RCS-Mittelwert-Karte

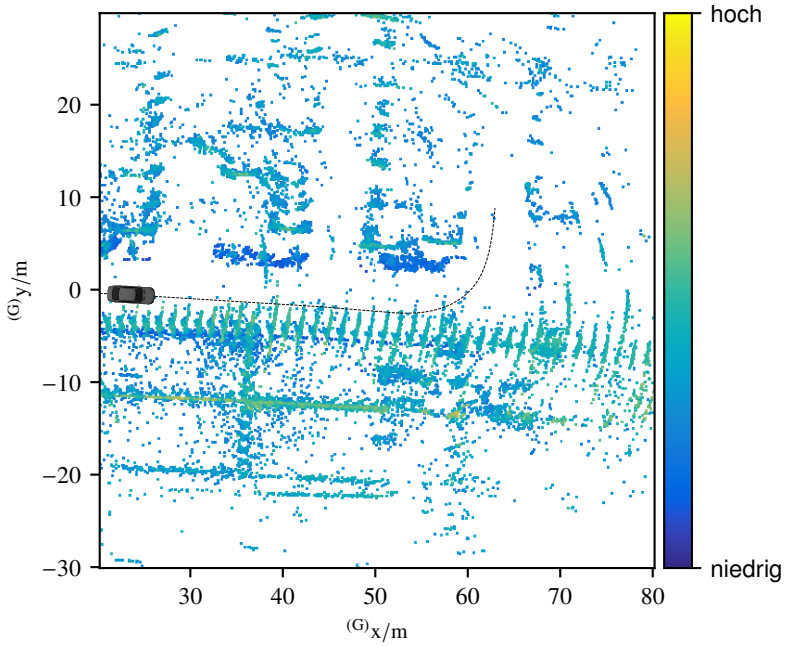


Abbildung 2.9: Beispiel einer RCS-Mittelwert-Karte.

Bei den Belegungskarten wird der RCS-Wert als charakteristisches Merkmal des Radars vernachlässigt. Um diese Eigenschaft zu nutzen, werden im folgenden Merkmalskarten vorgeschlagen, die den RCS-Wert integrieren.

Um eine Größe über die Zeit zu schätzen, kann in vielen Fällen der Mittelwert bzw. ein gewichteter Mittelwert geschätzt werden. Dieses Vorgehen wird bei der RCS-Mittelwert-Karte gewählt. Dabei wird davon ausgegangen, dass jede Zelle unabhängig von ihrem Betrachtungswinkel ähnliche Reflexionseigenschaften besitzt. Die

Grundannahme, dass benachbarte Zellen unabhängig voneinander sind, gilt auch hier.

Nun wird ein Schätzer benötigt, der den RCS-Wert der Zellen ${}^{\overline{RCS}}s_{i,j}$, in Abhängigkeit von den Messungen bestimmt, dabei soll die Varianz mit berücksichtigt werden. Aufgrund der Messung, soll nun der Erwartungswert des RCS-Wertes geschätzt werden. Dies erfolgt mithilfe des gewichteten Mittelwerts. Das Gewicht w_τ entspricht der Wahrscheinlichkeit, dass eine Messung $z_{t,k}$ in die Zelle $m_{i,j}$ fällt. Daraus ergibt sich:

$$ {}^{\overline{RCS}}s_{i,j} = \frac{\sum_{\tau=0}^{t} w_\tau a_\tau}{\sum_{\tau=0}^{t} w_\tau} \tag{2.29} $$

Algorithmus 2 zeigt die Berechnungsvorschrift der Karte.

Algorithmus 2 : *RCS-Mittelwert-Karte* Update

Data : $\{\text{rcssum}_{i,j,t-1}\}$,$\{\text{rcsweight}_{i,j,t-1}\}$, x_t, $\boldsymbol{z_t}$
Result : $\{{}^{\overline{rcs}}m_{i,j,t}\}$

```
foreach cells m_{i,j} ∈ M do
    foreach z_{t,k} ∈ z_t do
        if m_{i,j} ∩ 3σ(z_{k,t}) ≠ ∅ then
            w_{t,k} = Inverses Sensormodell(z_{k,t}) ;
            rcssum_{i,j} = rcssum_{i,j} + w_{t,k} σ̆_{t,k};
            rcsweight_{i,j} = rcsweight_{i,j} + w_{t,k} ;
        end
    end
    if rcsweight_{i,j} > 0 then
        ^{rcs}s_{i,j,t} = rcssum_{i,j} / rcsweight_{i,j} ;
    else
        ^{rcs}s_{i,j,t} = −∞ ;
    end
end
```

In Abb. 2.9 ist diese RCS-Mittelwert-Karte zu sehen. Gegenüber der Belegungskarte unterscheiden sich Objekte mit sehr niedrigem RCS-Wert, wie z. B. Verkehrsinseln und Bäume, erheblich von Objekten mit hohen RCS-Werten, wie z. B. Fahrzeuge, Metallpfosten und Gebäude. Die von der Winkelunsicherheit verursachte Streuung sorgt für eine unscharfe Abbildung der Objekte. Bei den Pfosten, sorgt es beispielsweise dafür, dass die Umgebung einen hohen RCS-Wert bekommt, da im Bereich der Streuung oft nur wenige andere Messungen existieren.

2.4.3 RCS-Minimum und RCS-Maximum-Karte

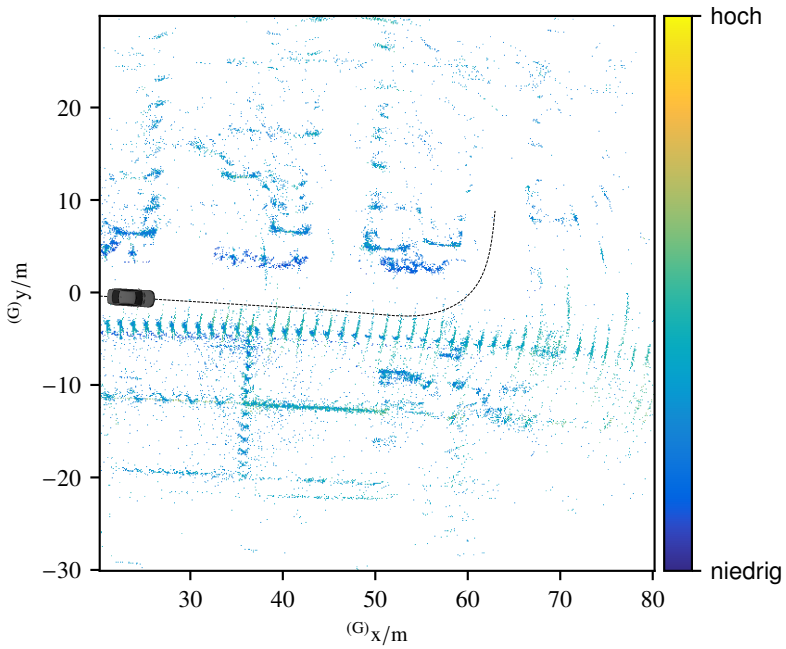


Abbildung 2.10: Beispiel einer RCS-Minimum-Karte.

Durch eine Mittelwertbildung, gehen viele charakteristische Muster von Objekten verloren. Beispielsweise bei einem Metallpfosten auf einer Wiese, führen Reflexionspunkte auf den Pflanzen dazu, dass ein gemittelter RCS-Wert deutlich geringer ist, als bei einem freistehenden Pfosten. Daher werden charakteristische Eigenschaften einzelner Objekte verschleiert. Ein ähnlicher Effekt tritt bei einem stark richtungsabhängigen Reflexionszentrum, wie zum Beispiel einem Bordstein, auf. In einem bestimmten Winkelbereich ist ein großer Rückstreuquerschnitt zu beobachten, während in allen anderen Bereichen ein geringer Rückstreuquerschnitt zu beobachten ist.

Daher besteht eine weitere Möglichkeit in der Bestimmung der Extremwerte, also dem Minimum und dem Maximum des RCS-Wertes.

Die Karten werden hierbei mit dem jeweiligen Gegenteil initialisiert. Also bei der RCS-Minimum-Karte, mit dem höchsten zu erwarteten Wert und umgekehrt.

Eine große Schwäche dieser Art von Karte ist die fehlende Robustheit gegen Ausreißer. Pro Zelle genügt eine einzige Messung mit einem Extremwert des RCS, um den Wert der Zelle zu definieren. Um dies in den Griff zu bekommen, wäre ein lokales

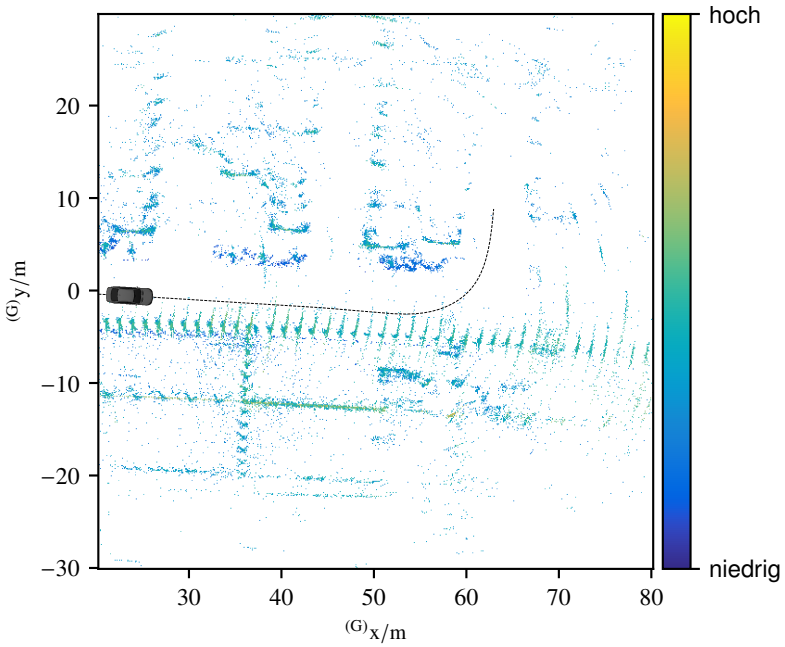


Abbildung 2.11: Beispiel einer RCS-Maximum-Karte

räumliches Filtern benachbarter Zellen mittels Mittelwertbildung denkbar. Da die meisten, in dieser Arbeit eingesetzten Klassifikationsmethoden aus Faltungsnetzen bestehen, und diese in der Lage sind solche räumlichen Filter repräsentieren, wurde auf eine derartige Maßnahme verzichtet.

In Algorithmus 3 und 4 sind die Berechnungsvorschriften für die RCS-Minimum und RCS-Maximum-Karte zu finden.

Auch wenn die Karten auf den ersten Blick sehr ähnlich erscheinen, sind beim genauen Hinsehen deutliche Unterschiede zu erkennen. So sind die gespiegelten Pfosten im Bereich $[36, -15]$ in der RCS-Minimum-Karte kaum von den anderen Pfosten zu unterscheiden. In der RCS-Maximum-Karte ist dieser Unterschied klar zu erkennen. Auch der Unterschied zwischen der Straßenlaterne und den Bäumen ist deutlicher zu sehen. In Zellen in die nur eine Messung fällt, haben die beiden Karten denselben Wert. So ist davon auszugehen, dass der Unterschied stärker wird, je mehr Messungen vorhanden sind. Insbesondere Cluttermessungen mit niedrigem RCS-Wert zerstören die Aussagekraft der RCS-Minimum-Karte. Diese Anfälligkeit besteht im Besonderen auch, da bei diesen Karten die Varianz der Messung unberücksichtigt bleibt.

Algorithmus 3 : *RCS-Minimum-Karte* Update

Data : $\{\widetilde{^{RCS}m}_{i,j,t-1}\}$, x_t, $\boldsymbol{z_t}$
Result : $\{l_{i,j,t}\}$
foreach *cells* $m_{i,j} \in \boldsymbol{M}$ **do**
  foreach $z_{k,t} \in \boldsymbol{z_t}$ **do**
    if $m_{i,j} \cap xyPos(z_{k,t}) \neq \varnothing$ **then**
      $\widetilde{^{RCS}m}_{i,j,t} = \min(z_{RCS}, \widetilde{^{RCS}m}_{i,j,t-1})$;
    else
      $\widetilde{^{RCS}m}_{i,j,t} = \widetilde{^{RCS}m}_{i,j,t-1}$
    end
  end
end

Algorithmus 4 : *RCS-Maximum-Karte* Update

Data : $\{\widehat{^{rcs}m}_{i,j,t-1}\}$, x_t, $\boldsymbol{z_t}$
Result : $\{l_{i,j,t}\}$
foreach *cells* $m_{i,j} \in \boldsymbol{M}$ **do**
  foreach $z_{k,t} \in \boldsymbol{z_t}$ **do**
    if $m_{i,j} \cap xyPos(z_{k,t}) \neq \varnothing$ **then**
      $\widehat{^{RCS}s}_{i,j,t} = \max(\check{\sigma}, \widehat{^{RCS}s}_{i,j,t-1})$;
    else
      $\widehat{^{RCS}s}_{i,j,t} = \widehat{^{RCS}s}_{i,j,t-1}$
    end
  end
end

2.4.4 RCS-Histogramm-Karte

Eine weitere Methode der Problematik von Ausreißern zu begegnen, ist ein Histogramm. Dabei wird der RCS Wert in Wertebereichen zerlegt, um dann die Anzahl der Messungen pro Wertebereich zu erfassen. Erstellt man ein Histogramm pro Zelle können hier leicht Ausreißer extrahiert werden.

Zieht man nun die Varianzen jeder einzelnen Messung in Betracht, kann dies leicht in ein 3D-Mapping Problem überführt werden. Die ersten beiden Dimensionen (i, j) repräsentieren die räumliche Ausdehnung, während die dritte Dimension k den Rückstreuquerschnitt repräsentiert.

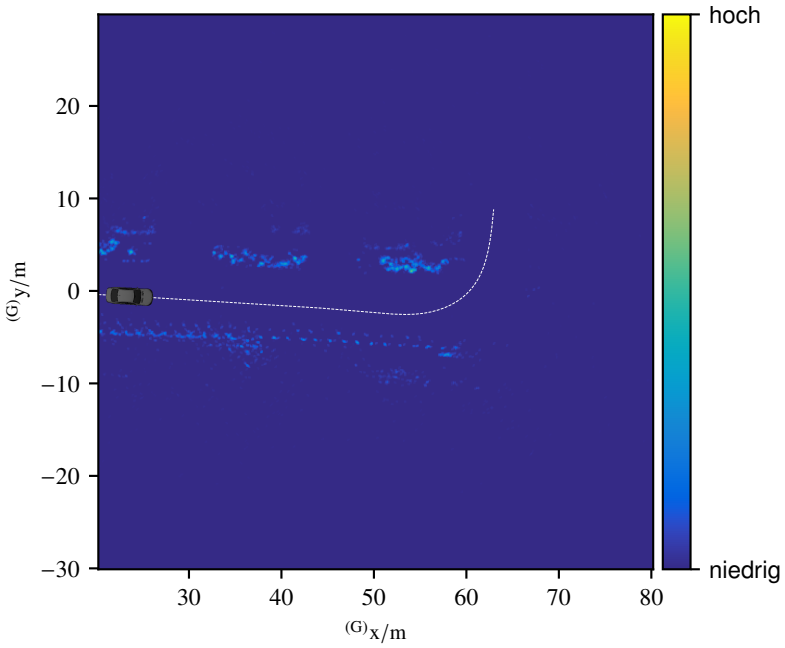


Abbildung 2.12: Beispiel 1. Schicht RCS-Histogramm-Karte.

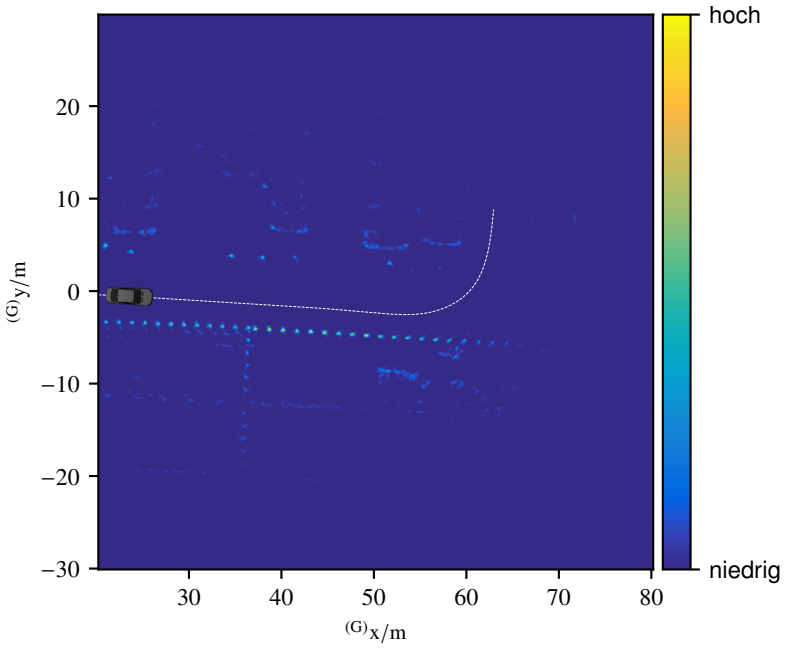


Abbildung 2.13: Beispiel 3. Schicht RCS-Histogramm-Karte.

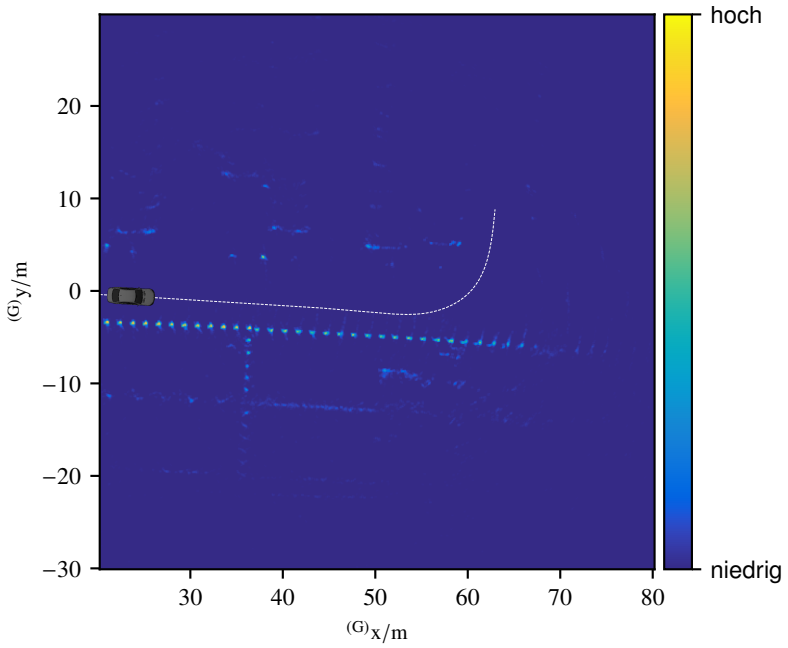


Abbildung 2.14: Beispiel 4. Schicht RCS-Histogramm-Karte.

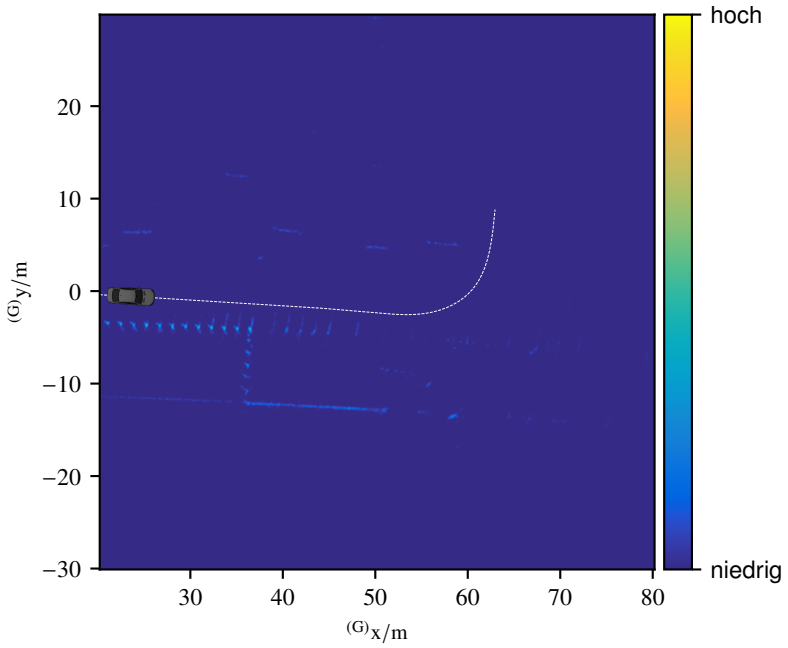


Abbildung 2.15: Beispiel 6. Schicht RCS-Histogramm-Karte.

Algorithmus 5 : *RCS-Histogramm* Update

Data : $\{l_{i,j,k,t-1}\}$, x_t, $\boldsymbol{z_t}$
Result : $\{l_{i,j,k,t}\}$

```
foreach cells m_{i,j} ∈ M do
    foreach rcsbin_k ∈ dom A do
        ^{RCS}l_{i,j,k,t} = ^{rcs} l_{i,j,k,t-1};
        foreach z_{n,t} ∈ z_t do
            if m_{i,j} ∩ 3σ(z_{k,t}) ≠ ∅ and rcsbin_k ∩ RCS(z_{k,t}) ≠ ∅ then
                ^{rcs}l_{i,j,k,t} = ^{rcs} l_{i,j,k,t-1} + Inverses Sensormodell(m_{i,j}, x_t, z_{n,t}) − ^{rcs} l_0;
            end
        end
    end
end
```

Die Diskretisierung des Wertebereichs bestimmt hierbei die Granularität der Auflösung. Je feiner diese ist, desto differenzierter kann die Verteilung der RCS-Werte bestimmt werden. Auf der anderen Seite steigt damit sowohl die Komplexität in Rechenzeit und in Speicherbedarf.

Ebenso wie für die anderen Messgrößen der Belegungskarte kann angenommen werden, dass der RCS-Wert normalverteilt ist, und somit einen Mittelwert und eine Varianz besitzt.

Die Updatevorschrift wird in Algorithmus 5 beschrieben. Mithilfe der Integration über den Wertebereich des Radarquerschnitts, lässt sich die normale Belegungskarte wiederherstellen. Bei der Repräsentation in Logits, kann dies durch ein Einfaches summieren über die RCS-Kanäle erreicht werden. Auf eine separate Berechnung der Belegungskarte kann daher verzichtet werden.

Abbildungen 2.12 bis 2.15 zeigen 4 Schichten einer RCS-Histogramm-Karte, die in 6 Wertebereiche zerlegt worden ist. Es ist zu beobachten, dass sich in den verschiedenen Schichten jeweils andere Objekte deutlicher abbilden. In der ersten Schicht (Abb. 2.12) sind die bepflanzten Inseln deutlich sichtbar, ebenso der Clutter im Bereich der Pfosten. Zusätzlich sind schwache Konturen von einzelnen Fahrzeugen abgebildet. In Schicht 3 (Abb. 2.13) sind die Konturen der Fahrzeuge schon besser zu erkennen. Zusätzlich sind hier die Pfosten und Bäume deutlich abgebildet. Schicht 4 (Abb. 2.14) ist ähnlich der Schicht 3, lediglich Objekte mit einem niedrigeren RCS-Wert, wie Bäume, treten in den Hintergrund. Schicht 6 (Abb. 2.15) enthält nur noch Objekte mit einem sehr hohen RCS-Wert.

Da die Messungen unter Berücksichtigung ihrer Varianz in der Karte eingetragen werden, sind dadurch auch klar die Formen der einzelnen Objekte ersichtlich.

2.5 Radar-Eigenschaften von Objekten

Das Erscheinungsbild eines Objektes im Radarbild hat unterschiedliche Ausprägungen. Dies wird von einer Reihe von Faktoren bestimmt, die sich im Wesentlichen in drei Kategorien einteilen lassen.

- Reflexionseigenschaften des Objektes
- Eigenschaften des Sensors und der Signalverarbeitung
- Eigenschaften der Umgebung

Die Reflexionseigenschaften des Objektes werden im Wesentlichen durch die elektromagnetischen Eigenschaften seines Materials und die Form der Streuzentren bestimmt. So unterscheidet beispielsweise Knott [Kno08] sieben verschiedene Reflexionsmechanismen. Diese Eigenschaften sind meist anisotrop und lassen sich nur für einfache Objekte analytisch bestimmen. Eine Simulation benötigt viel Rechenzeit und genaue Modelle des zu untersuchenden Objekts. Teilweise können diese Eigenschaften im Labor gemessen werden, wie z. B. in Werber et al. [Wer+14] für ein Verkehrsschild durchgeführt wurden.

Prinzipiell lässt sich sagen, dass Kanten und Flächen die durch ihre Anordnung einen Winkelreflektor darstellen, sehr gut reflektieren, während beispielsweise große Metallflächen nur dann gut reflektieren, wenn sie 90° zum Sensor stehen. Ansonsten wird das elektromagnetische Signal wegreflektiert. Bei den Materialien reflektiert Metall sehr gut, während trockenes Holz einen eher schlechten Reflektor darstellt.

Die Eigenschaften des Sensors und der Signalverarbeitung sind für die Sensitivität, die Genauigkeit in Winkel, Entfernung und Doppler verantwortlich. Daneben können verschiedene Antennenkonfigurationen und Signalverarbeitungen Ambiguitäten im Winkel verursachen. Diese können zu Geisterzielen führen.

Der Einfluss der Umgebung ist ein weiterer großer Faktor im automobilen Bereich, da sich durch die dicht belegte Umgebung zahlreiche Störeinflüsse bemerkbar machen. Eine stark reflektierende Umgebung und andere elektromagnetische Sender, sorgen für ein geringes Signal-Rausch-Verhältnis, welches verhindert, dass schwache Ziele detektiert werden.

Bei der Überlagerung von Signalen aus mehreren Streuzentren entsteht bei einer relativen Bewegung zum Objekte Fluktuationsverluste. Diese hat Swerling [Swe60] in seiner Arbeit in vier unterschiedliche Fälle eingeteilt.

Mehrfachreflexionen erzeugen zusätzliche Ziele. So kann z. B. eine Leitplanke dafür sorgen, dass das eigene Fahrzeug als zusätzliches Objekt hinter der Leitplanke wahrgenommen wird.

Durch Mehrfachreflexionen über die Straße, ist es möglich Objekte oder Teile von Objekten wahrzunehmen, welche nicht auf der direkten Sichtachse liegen. Zum Beispiel ein Fahrzeug, dass durch das vorrausfahrende Fahrzeug verdeckt wird. Hierbei treten jedoch Verzerrungen in der Position auf.

Clutter ist ein weiteres Phänomen, das stark durch die Umgebung bedingt wird, so sorgt ein rauer Straßenbelag, z. B. eine Schotterfläche, für Clutterziele, welche für die Umgebungserfassung nicht erwünscht sind.

Im Folgenden soll anhand einiger Beispiele, diese Eigenschaften von Objekten in Radar erläutert und somit dem Leser ein qualitativer Eindruck der in der Arbeit verwendeten Daten, gegeben werden. Zudem soll auf einige Besonderheiten hingewiesen werden. Eine quantitative Betrachtung findet mithilfe des gelabelten Datensatzes in Kapitel 4.5 statt.

2.5.1 Beispiele

Zuerst werden die Messungen und die Belegungskarten von zwei normalen Objekten betrachtet, einem Fahrzeug und einem Zaun. Später wird anhand von Beispielen auf Clutter und Mehrfachreflexionen eingegangen.

In Abb. 2.16 ist ein Fahrzeug nach einer Vorbeifahrt zu sehen. Im unteren Bereich werden die Radarziele vieler Messzyklen als Punktwolke dargestellt, rechts oben befindet sich die daraus errechnete Belegungskarte.

In den Messungen lässt sich erkennen, dass vom ganzen Fahrzeug, Reflexionen extrahiert werden können. Auch von den Bereichen die eigentlich innerhalb des Fahrzeugs liegen. Dies sind meist Reflexionen vom Unterboden, die durch Mehrfachreflexion zwischen dem Fahrzeug und dem Fahrbahnbelag entstehen. Des Weiteren zeigt sich, dass im Bereich der Radkästen und der Stoßstange die meisten Reflexionen entstehen, während auf der Längsseite des Fahrzeugs weniger Ziele zu sehen sind. Durch ihre glatte Oberfläche wird in diesem Bereich die ausgesendete Energie des Sensors, durch eine annäherungsweise totale Reflexion, zu großen Teilen vom Sensor weg reflektiert. Daher entstehen dort wenige Radarziele.

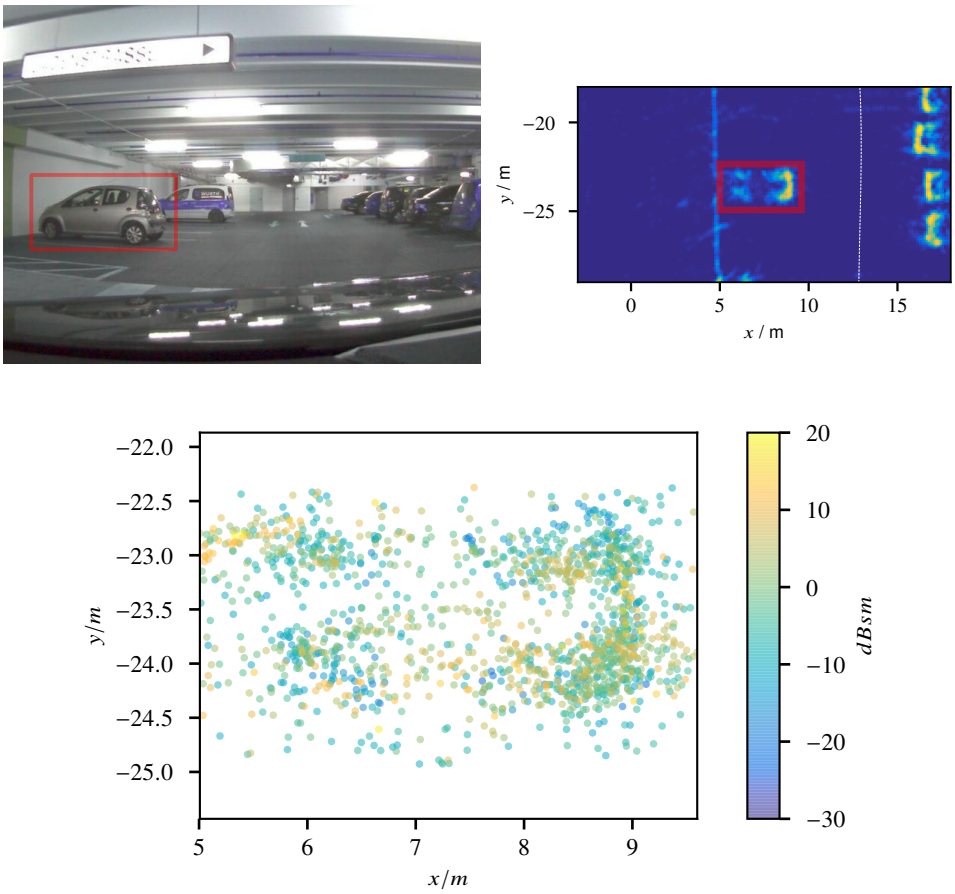


Abbildung 2.16: Beispiel Fahrzeug: Links oben befindet sich ein Foto der Szene. Dieses zeigt rot hervorgehoben das Fahrzeug, welches in den beiden anderen Grafiken betrachtet wird. Die Grafik rechts oben, zeigt den entsprechenden Ausschnitt der Belegungskarte. Die Messungen des Fahrzeugs, der in der Belegungskarte rot umrandete Bereich, sind in der unteren Abbildung als Punktwolke dargestellt. Dabei ist der RCS-Wert farblich codiert.

Da sich die Front des Fahrzeugs auf der dem Sensor abgewandten Seite befindet, kann diese nur indirekt durch Unterbodenreflexionen oder andere Mehrfachreflexionen beobachtet werden. Im Allgemeinen ist der RCS-Wert eines Fahrzeuges relativ hoch, die einzelnen Ziele verteilen sich jedoch über ein sehr breites Spektrum.

In der Belegungskarte bildet sich das Heck des Fahrzeugs gut ab, auch der Bereich der Vorderräder ist gut zu sehen. Die Front und die Seitenflächen des Fahrzeuges sind nur ansatzweise zu erkennen. Trotzdem lässt sich die Außenform des Fahrzeuges

gut abschätzen. In den Messungen ist aber zu sehen, dass sich einige Messungen auch außerhalb des Fahrzeuges befinden. Dies ist entweder durch die Varianz des Sensors, durch Mehrfachreflexionen oder durch Cluttermessungen zu erklären. Den verursachenden Effekt mit Sicherheit zu bestimmen, ist in der realen Welt nur selten möglich.

Da sich in vielen Bereichen niedrige und hohe RCS-Werte überlagern, gehen bei der Integration der Daten in eine RCS-Mittelwert-Karte Information verloren. Daher sind Karten, welche die Extremwerte (RCS-Minimum/RCS-Maximum Karte) oder das Histogramm (RCS-Histogramm-Karte) betrachten, aussagekräftiger.

In Abbildung 2.17 ist ein Teil eines Metallzauns abgebildet. Hierbei lässt sich die Position der einzelnen Pfosten sehr gut in der Belegungskarte erkennen. Diese bilden je nach Art des Zauns, entweder kleine Winkelreflektoren oder sind durch Form und Material oft die stärkeren Radarziele als die Flächen des Zauns. Durch ihren starken Reflexionsgrad sind die Pfosten schon aus weiter Entfernung zu sehen, daher führt der Winkelfehler des Sensors zu einer starken Streuung im Bereich der Pfosten, diese ist in der Punktwolke zu sehen. Wenn dieser Winkelfehler auf das Kartesische System übertragen wird, ist die Varianz in großen Entfernungen deutlich größer. Dadurch prägt sich dieser Effekt auch nicht so stark in der Belegungskarte aus, da diese Messungen durch die höhere Varianz weniger Gewicht haben.

Der RCS-Wert der Zaunelemente sind im Durchschnitt geringer, als die der Pfosten. Ebenso werden weniger Punktziele extrahiert, was zu einer unterschiedlichen Intensität in der Belegungskarte führt. In der Umgebung sind einige Clutterziele zu sehen, die unter anderem durch das Gras verursacht werden. Auch hier werden Messungen mit unterschiedlichem RCS-Wert überlagert, die Verteilung in den einzelnen Bereichen ist jedoch etwas geringer als beim Fahrzeug, sodass anzunehmen ist, dass bei einer RCS-Mittelwert-Karte weniger Information verloren gehen.

Die gezeigten Eigenschaften können genutzt werden, um die beobachteten Objekte zu klassifizieren. In verschiedenen Szenarien treten jedoch auch störende Effekte auf, die im Folgenden erläutert werden.

Ein rauer Untergrund wie z. B. ein geschotterter Untergrund sorgt für erhöhte Anzahl an Clutterzielen. Dies ist in Abb. 2.18 zu sehen. Ähnliches tritt auch bei grasbewachsenem Untergrund auf. In der Regel sind dabei die RCS-Werte der Messungen niedrig. Da eine normale Belegungskarte die Messungen RCS unabhängig einträgt, ist das Clutter in der Belegungskarte deutlich zu sehen. Ein pauschales Filtern würde dazu führen, dass Objektteile mit niedrigem RCS-Wert ebenfalls gefiltert würden. Die RCS-Histogramm-Karte ermöglicht es, diese Messungen als separaten Kanal zu erfassen, und erleichtert damit dem Klassifikator die Belegung in den einzelnen RCS-Wertebereichen gezielt zu verarbeiten.

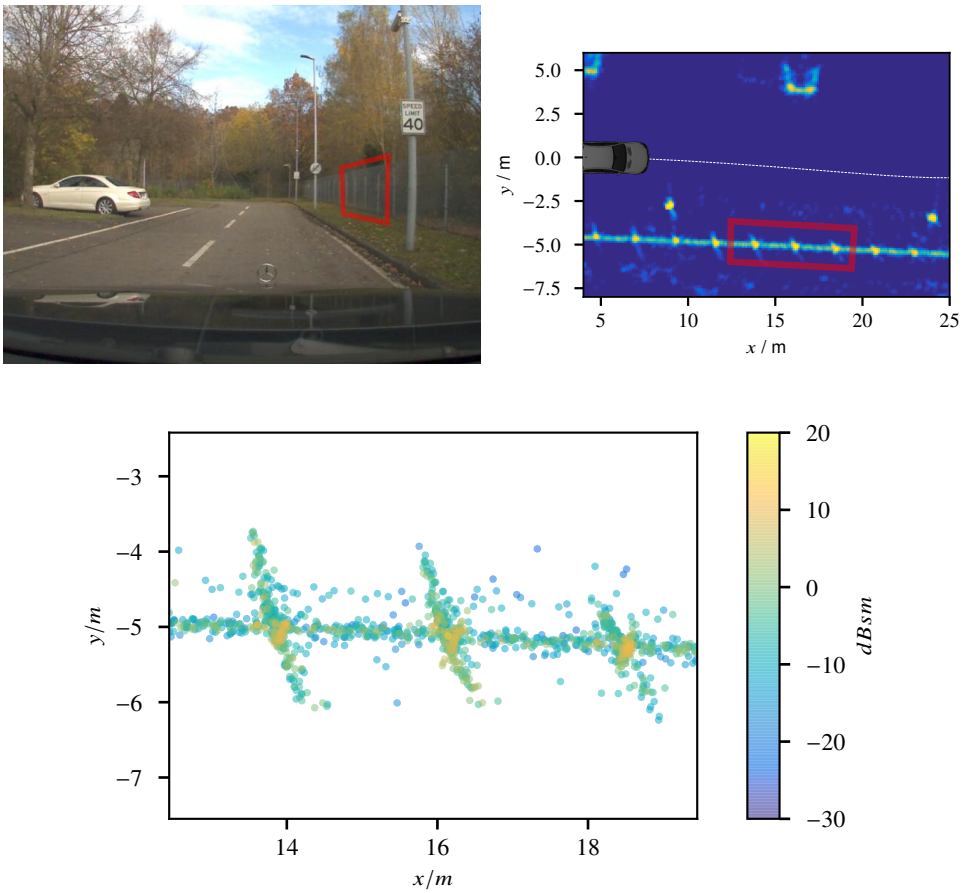


Abbildung 2.17: Beispiel Zaun: Links oben befindet sich ein Foto der Szene, in den anderen beiden Abbildungen wird der Zaun, der mit einer roten Box markiert ist, dargestellt. Oben rechts befindet sich die Abbildung der Belegungskarte. Rot hinterlegt ist hierbei der Bereich, der in der unteren Abbildung dargestellt wird. Die untere Abbildung zeigt die Radarpunktwolke dabei ist der RCS-Wert farblich codiert.

Es existieren jedoch auch Cluttermessungen mit hohem RCS-Wert. Diese sind Beispielsweise durch Dehnfugen an Brücken oder auch wie in Abbildung 2.19 dargestellt in Parkhäusern. In diesem Bild sieht man, dass die Deckenträger des Parkhauses gute Radarziele darstellen. Diese haben aufgrund ihrer Form und ihres Materials gute Reflexionseigenschaften im Radar. Da die verwendeten Radare keinen Elevationswinkel messen, ist der Ursprung nur mit Methoden wie z. B. dem *Doppler Beam Sharpening* zu bestimmen [Lar+17]. Aufgrund des limitierten Öffnungswinkels der Antennenkeule in Elevation sind diese Ziele nur in einem bestimmten Entfernungs-

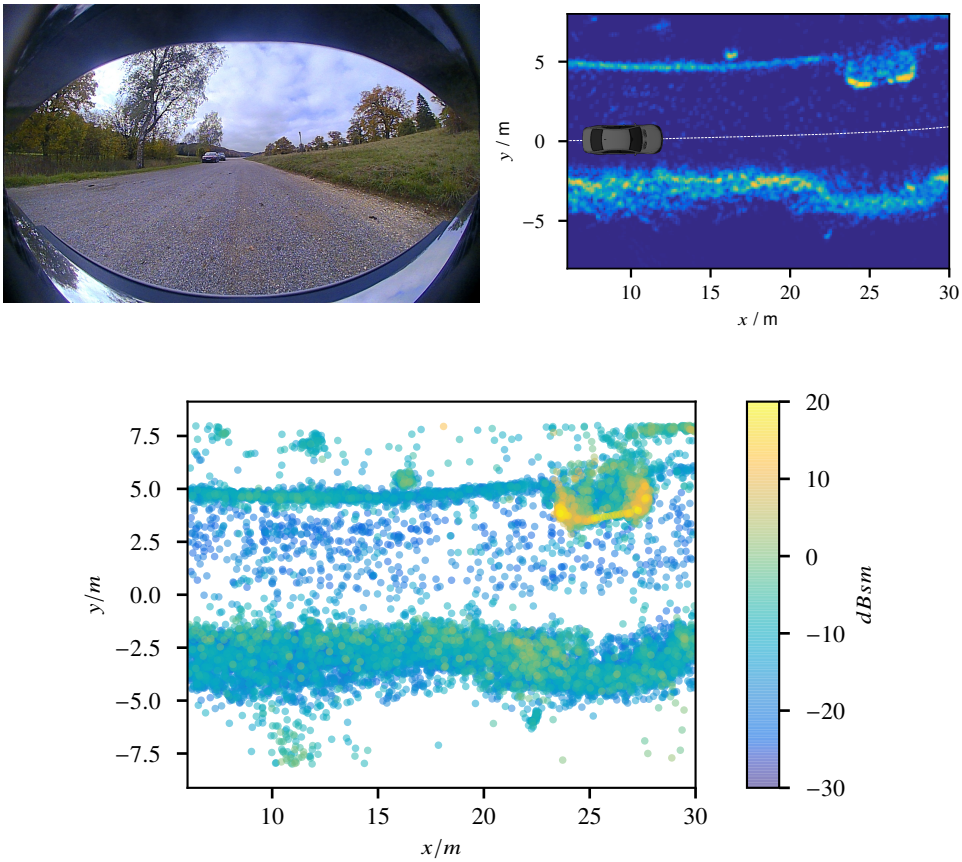


Abbildung 2.18: Beispiel Clutter: Links oben befindet sich ein Foto der Szene, in der ein Fahrzeug auf Kiesuntergrund abgebildet wird. Oben rechts befindet sich die Abbildung der Belegungskarte. Die untere Abbildung zeigt die Radarpunktwolke dabei ist der RCS-Wert farblich codiert.

bereich zu beobachteten.

Ein weiterer Effekt ist die Mehrfachreflexion des Signals zwischen stark reflektierenden Objekten, wie z. B. Metallpfosten oder Metallflächen wie Leitplanken. Hierbei tauchen Objekte als Geistobjekte hinter dem Spiegel auf. Da sich die Position eines gespiegelten Objektes mit der Position des eigenen Fahrzeugs verschiebt, führt dies zu einem Wandern dieser Geisterziele. Durch die zeitliche Integration der Daten in der Belegungskarte sind diese oft nur als Rauschen wahrnehmbar. RCS-Karten, außer der RCS-Histogramm-Karte, sind deutlich empfindlicher, da hier eine einzelne Messung für die Belegung einer Zelle genügt.

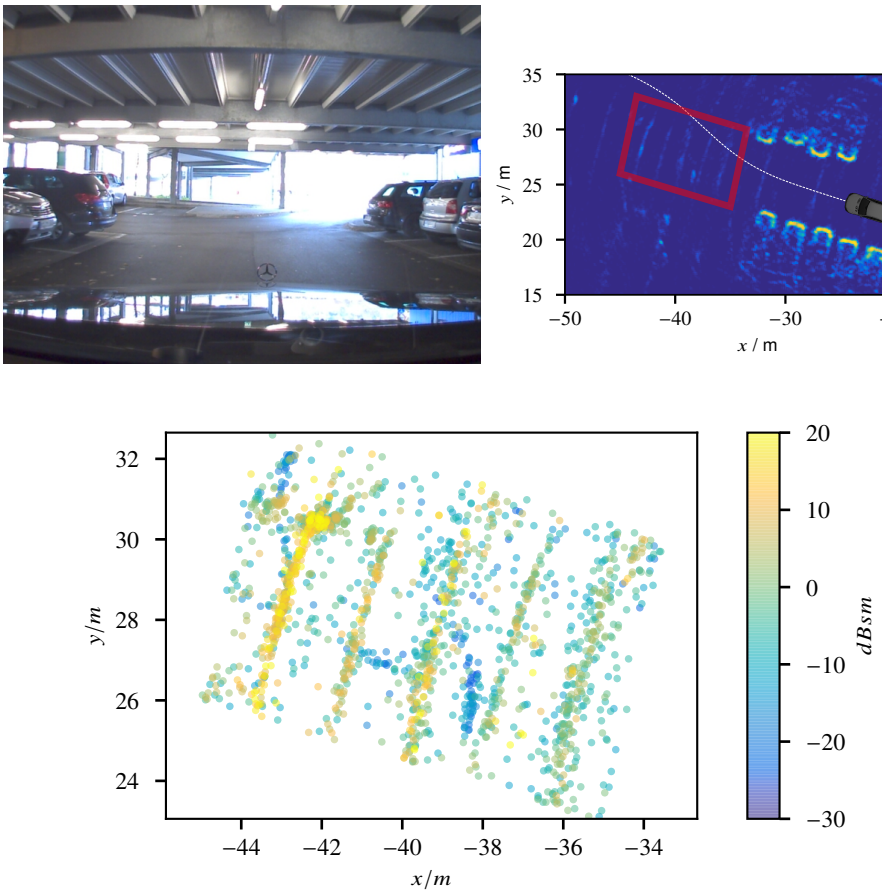


Abbildung 2.19: Beispiel Clutter: Links oben befindet sich ein Foto eines Parkhaus Szenarios. Oben rechts befindet sich die Abbildung der Belegungskarte. Der rot umrandete Bereich in der Belegungskarte ist in der unteren Abbildung als Punktwolke dargestellt. Der RCS-Wert ist dabei farblich codiert.

Lediglich in sehr symmetrischen Szenarien wie z. B. der in Abb. 2.20 dargestellten Brücke, sieht man diese Artefakte. In dieser Abbildung sind diese in mehreren Reihen, hinter dem Geländer der Brücke zu sehen. Hierbei dient sowohl das Tragwerk der Brücke, wie auch das Geländer als Spiegel. Da bei statischen Objekten meist nur die dem Fahrzeug nahe liegenden Objekte relevant sind, und Spiegelobjekte immer eine größere Entfernung als der Spiegel haben, kann dieser Effekt in vielen Fällen vernachlässigt werden.

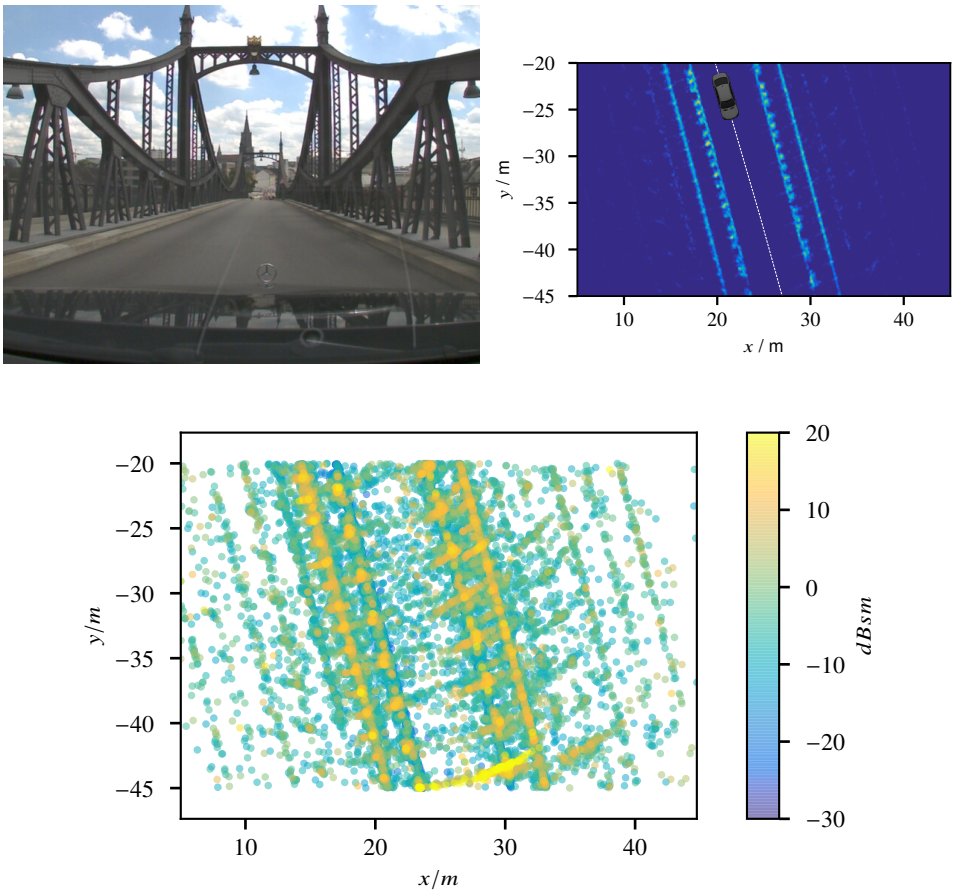


Abbildung 2.20: Beispiel Mehrfachreflexion: Links oben befindet sich das Bild eines Brückenszenarios. Oben rechts befindet sich die Abbildung der Belegungskarte. Die untere Abbildung zeigt die Radarpunktwolke, dabei ist der RCS-Wert farblich codiert.

2.6 Zusammenfassung

In diesem Kapitel wurde auf die grundlegenden Eigenschaften des Radarsensors eingegangen und vorgestellt wie die Merkmale Distanz, Azimut Winkel und Radar-Rückstreuquerschnitt zur Klassifikation genutzt werden können.

Es wurde gezeigt, wie mithilfe von Belegungskarten die Radardaten über die Zeit akkumuliert werden können. Des Weiteren wurden neue Konzepte zu Radar-Merkmalskarten entwickelt, die insbesondere den Rückstreuquerschnitt berücksich-

tigen.

Zum Abschluss wurden dem Leser anhand verschiedener qualitativer Beispiele die Eigenschaften eines automobilen Radars nähergebracht.

Kapitel 3

Labeling von Radardaten

Beim überwachten maschinellen Lernen wird das Modell nicht explizit bestimmt, sondern mithilfe eines Optimierungsalgorithmus aus dem Trainingsdatensatz inferiert. Daher ist ein geeigneter Datensatz ein wichtiges Fundament vieler maschineller Lernalgorithmen. Dieser enthält neben den Eingangsdaten (z. B. Bilder, Temperaturen, etc.), auch die Zielgröße, die als Ergebnis erwartet wird (z. B. Klassenkategorien, Wetterverläufe, etc.). Mithilfe dieses Datensatzes werden die Lernalgorithmen optimiert und ihre Güte evaluiert.

Für viele Problemstellungen stehen mittlerweile öffentlich verfügbare Standarddatensätze zur Verfügung. Dies trifft insbesondere im Bereich der Bildverarbeitung zu. Für andere Bereiche z. B. Wetterdaten oder Börsenkurse können diese Daten aus der Historie gewonnen werden.

Für den Bereich des Radars im automobilen Umfeld stand zum Zeitpunkt der Arbeit kein öffentlicher Datensatz zur Verfügung, daher musste für diese Arbeit ein entsprechender Datensatz aufgenommen und manuell annotiert werden.

In diesem Kapitel wird daher zunächst ein kurzer Überblick über verschiedene Datensätze und deren Annotation gegeben. Danach wird das Labeling des Radardatensatzes beschrieben. Dieses wurde mithilfe von Werkstudenten durchgeführt. Anschließend wird dieser Datensatz evaluiert und eine kurze Zusammenfassung gegeben.

3.1 Datensätze und Labeling

Um einen Datensatz für das überwachte Lernen zu erzeugen, muss ein Lehrer das wahre Label vorgeben. Diesen Prozess bezeichnet man als Labeling. Daher wird der Lehrer im Folgenden auch als Labeler bezeichnet.

Damit gute Klassifikationsergebnisse erzielt werden können, muss ein Datensatz zum einen richtig gelabelt sein, zum anderen muss er auch die Problemstellung möglichst umfassend abbilden. Hierbei ist nicht nur die Anzahl der Trainingsbeispiele entscheidend, sondern auch, ob diese die gesamte Domäne der Aufgabenstellung repräsentieren. Wenn Fahrzeuge von anderen Objekten unterschieden werden sollen, so bedarf es nicht nur Fahrzeuge eines Typs oder einer Marke im Datensatz, sondern es muss eine große Vielfalt abgedeckt werden. Ebenso müssen die Gegenbeispiele das Einsatzszenario des Klassifikators repräsentieren.

Je nach Anwendungsgebiet ist es mehr oder weniger aufwendig diese Daten zu erheben. So können die Daten zur Prognose des Einkaufsverhaltens verschiedener Kunden in einem Onlineshop algorithmisch aus der Historie gewonnen werden, während beim Erkennen von Objekten in aufgenommenen Bildern diese erst manuell annotiert werden müssen.

In der Regel werden diese Trainingsdaten von menschlichen Experten erstellt. Diese müssen, in dem entsprechenden Gebiet, die nötigen Kenntnisse besitzen. Bei der Annotation von Alltagsobjekten wie Bäumen, Menschen, etc. in einem Bild ist (fast) jeder Mensch dazu in der Lage. Bei fachspezifischen Bildern, wie z. B. das Labeling von Tumoren in einem Röntgenbild, kann dies nur durch entsprechendes medizinisches Fachpersonal oder speziell geschultes Personal erfolgen.

Weiter kann das Labeln in Bildern entweder pro Bild als Ganzes erfolgen, indem beispielsweise nur das Hauptobjekt gelabelt wird (z. B. [Jia+09]) oder es werden Gebiete mithilfe von Boundingboxen, entweder mit Rechtecken, Polygonzügen oder Kreisen, gelabelt (z. B. [Gei+13]). Eine weitere Form des Labeling ist die pixelweise Annotation(z. B. [Cor+16]), welche eine der aufwendigsten Formen des Labeling darstellt. Hier wird jedem Pixel eines Bildes einem Objekt und damit einem Label zugeordnet. Dies erfolgt entweder über filigrane Polygone oder über eine pixelweise Maske. Die Toleranz des Übergangs zwischen zwei Objekten beträgt hierbei wenige Pixel.

Je detaillierter gelabelt wird, desto kostenintensiver wird das Labeling. Während ein Labeler für die Zuordnung eines Bildes zu einer Klasse nur wenige Sekunden benötigt, muss beim pixelweisen Labeling mit mehreren Stunden pro Bild gerechnet werden. Um diese Kosten zu senken werden teilweise Pre-Labeling-Techniken eingesetzt, die einen Label-Vorschlag erzeugen, z. B. mittels eines Klassifikators oder

Tabelle 3.1: Datensätze

Datensatz	Sensoren	Beschreibung
MNIST [LC98]	Kamera	70.000 Graustufenbilder von zentrierten ZiffernLabel: pro Bild
ImageNet [Jia+09]	Kamera	ca. 14 Mio. Bilder, die Synsets (Synonymen von Substantiven) zugeordnet sindLabel: hauptsächlich pro Bild aber auch Bounding Boxen
KITTI [MG15; FKG13; Gei+13; GLU12]	Stereo-Kamera Laserscanner GPS-Position	Aufnahmen im Straßenverkehr durch Versuchsfahrzeug mit eingebauten Sensoren (Kamera, Laser), mit verschiedenen Labeln und Ground Truth Methoden.Z.B. Pixel und Box Label.
Cityscapes [Cor+16]		5000 - 25000 Bilder aus 50 Städten mit 30 verschiedenen KlassenLabel: pro Pixel
MSTAR [Ros+98]		X-Band SAR Bilder von hauptsächlich militärischen Fahrzeugen
Wide Angle SAR [Dun+12]	SAR-Radar	SAR Bilder mit großer Apparatur von verschiedenen Fahrzeugen
NEXRAD [NOA91]	Doppler Wetter Radar Netzwerk	Daten von US-Wetterradaren, hierbei wird beispielsweise die Reflektivität und die mittlere Radialgeschwindigkeit erfasst. Label: Wetterhistorie
nuScenes [Cae+20]	KameraLidar Radar Ego-Position	ca. 100.000 annotierte Objekte Label: 3-D Bounding Box
HiRes2019 [MK19]	Kamera Lidar Radar	546 annotierte Radar-Frames

mittels eines Clustering-Algorithmus, welcher die Objektinstanzen extrahiert. Die Labeler haben danach "nur" noch die Aufgabe diese Label zu korrigieren. Diese Methoden erhöhen die Labelgeschwindigkeit, erhöhen aber auch die Gefahr ungewollter Fehler der Labeler. Für SAR-Bildgebende Verfahren schlug z. B. [Pav+99] ein solches System vor.

Für viele Bereiche des maschinellen Lernens haben sich mittlerweile Standarddatensätze etabliert, die je nach Lizenz für die Wissenschaft aber auch für die Industrie öffentlich zur Verfügung gestellt werden. Einer der ersten Datensätze war der *MNIST* Datensatz [LC98]. Ein weiterer sehr populärer Datensatz in der Bildverarbeitung ist der *ImageNet* Datensatz [Jia+09]. Im automobilen Umfeld gehört der *Cityscapes* [Cor+16] und der *KITTI* Datensatz [MG15; FKG13; Gei+13; GLU12] zu den bekannten Datensätzen. Andere Arbeiten versuchen mittlerweile auch schon, die Veränderung der Objekte über die Jahreszeiten zu erfassen [Mad+17].

Im Bereich des Radars gibt es jedoch nur sehr wenige Datensätze. Diese sind in der Regel dem militärischen Bereich zuzuordnen, z. B. *MSTAR* [Ros+98] und *Wide Angle SAR* [Dun+12] oder dem Bereich der Meteorologie *NEXRAD* [NOA91].

Zur Zeit der Arbeit war dem Autor kein größerer Datensatz mit Radar Daten im Automobilen Umfeld bekannt, der öffentlich zur Verfügung stand. Im September 2018 veröffentlichte *nuTonomy-Aptive* einen Datensatz der in Boston und Singapur aufgenommen wurde. Dieser enthält neben einem Lidar und 6 Kameras ebenfalls 5 automobile Radar Sensoren. Des Weiteren veröffentlichte 2019 den Datensatz *HiRes2019* Meyer und Kuschk [MK19] einen Datensatz mit einem High Resolution Radarsensor, einem Lidar und einer Kamera.

In Tabelle 3.1 wird eine Übersicht über die Datensätze und den wichtigsten Eigenschaften gegeben.

3.2 Radar Labeling

In diesem Abschnitt wird das Labeling von Radardaten zur semantischen Klassifikation beschrieben.

Um Daten labeln zu können, muss ein Labeler in der Lage sein, aus den vorhandenen Daten das Label zu ermitteln. Hierzu ist es notwendig, dass der Labeler die Problemstellung kennt. Dazu gehört unter anderem, dass er die verschiedenen Klassen sicher abgrenzen kann. Er weiß, wie die verschiedenen Klassen zu annotieren sind, welche Bereiche der Daten annotiert werden und wie mit Sonderfällen z. B. Verdeckungen umgegangen wird. Dies ist wichtig, da jeder Fehler des Labelers von dem

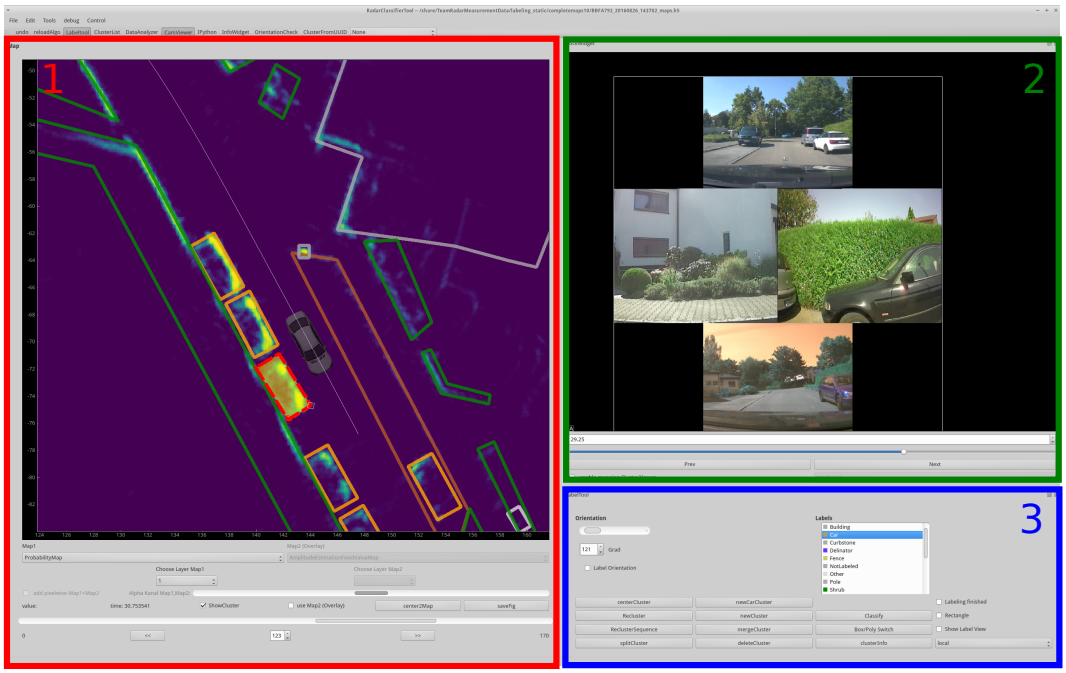


Abbildung 3.1: Ansicht der Software für das Labeling der Radar-Belegungskarte. Bereich 1 zeigt die Radar-Belegungskarte und die Navigationselemente. Der Bereich 2 zeigt das unterstützende Kamera Bild. Bereich 3 enthält das Selektionsmenü für das Label.

Lernalgorithmus mitgelernt wird. Das Labeling von Radardaten unterscheidet sich hierbei maßgeblich von dem Labeling von Bildern.

Für diese Arbeit sollen den Radardaten die semantischen Objektklassen zugeordnet werden, also beispielsweise *Fahrzeug*, *Bordstein*, *Baum*, etc. Bei einem normalen Foto, ist (fast) jeder Mensch ein Experte und kann direkt aus den Daten das zugehörige Label identifizieren. Für Radardaten ist eine direkte Zuordnung maximal durch einen Experten möglich. Auch wenn ihnen dies in allen Fällen möglich wäre, sind Radar-Experten für solche Aufgaben nicht in genügend großer Anzahl verfügbar bzw. zu teuer. Es wird daher eine zusätzliche Informationsquelle benötigt, um die Objektklasse und deren Ausdehnung zu identifizieren.

Eine Möglichkeit einen Datensatz ohne Labeling zu erzeugen bestünde darin, die Daten künstlich, entweder durch Simulation oder durch gestellte Szenarien zu erzeugen. Für die Simulation fehlen jedoch geeignete Modelle, um die Radarantwort komplexer Objekte und deren Umgebung zu bestimmen. Des Weiteren sind die meisten dieser Verfahren rechentechnisch sehr aufwendig. Auch das Stellen von Szenarien

ist sehr aufwendig und spiegelt oft nicht die Komplexität der realen Welt wieder, die beispielsweise durch die Vielfalt der Objekte, Mehrfachreflexionen, etc. entstehen.

Daten auf der Punktziel-Ebene, also anhand der extrahierten Reflexionszentren zu Labeln ist sehr aufwendig. Die Messpunkte können nur sehr schwer einem Objekt zugeordnet werden, insbesondere in hoher Entfernung. Eine Ursache ist, dass sich die Form eines Objekts nicht aus einer einzelnen Radarmessung bestimmen lässt.

Da das Labeln auf der Punktziel-Ebene zu aufwendig und zu fehleranfällig ist, wird auf den Belegungskarten gelabelt. Hierzu werden die Bereiche der Objekte mit Polygonen umrandet und mit einem Label versehen. Es ist jedoch zu beachten, dass Belegungskarten schon eine vorverarbeitete Darstellung der Radardaten sind. Diese Vorverarbeitung unterliegt bestimmten Parametern. Falls direkt auf diesen Daten gelabelt wird, müssen bei der Veränderung eines Parameters, alle Daten neu gelabelt werden.

Aus diesem Grund werden Label-Polygone in einem globalen Koordinatensystem erzeugt. Dies hat den weiteren Vorteil, dass die Label-Polygone auf verschiedene Zeitpunkte der Belegungskarten projiziert werden können. Dies ist natürlich nur für die in der Arbeit betrachteten statischen Objekte möglich. Dynamische Objekte müssen auf eine andere Weise gelabelt werden.

Des Weiteren können dadurch die Radar-Merkmalskarten mit unterschiedlichen Parametern erneut berechnete werden und anschließend die Labelpolygone dann auf die neuen Karten angewendet werden. Daher müssen diese Karten nicht erneut gelabelt werden.

Für das Zuordnen der Klassen erhalten die Labeler Kamerabilder als weitere Unterstützung. Hierzu wurden, zusätzlich zu den Radarsensoren, vier Kameras die ebenfalls einen Blick in alle viel Himmelsrichtungen ermöglichen, im Fahrzeug installiert. Erste Versuche mit zwei Weitwinkelkameras nach vorne und hinten, erwiesen sich als zu fehleranfällig, da seitlich Objekte oft nicht der richtigen Position zugeordnet werden konnten. Außerdem waren manche Objekte dadurch komplett für den Labeler verborgen.

Das Labeltool, wie in Abbildung 3.1 zu sehen, enthält im Wesentlichen drei Elemente. Als erstes die Darstellung der Belegungskarte im Bereich eins. Hier werden die Objekte je nach Form entweder mit einem Polygon oder mit einem Rechteck umrandet. Zusätzlich wird als Orientierungshilfe die Trajektorie des Fahrzeugs dargestellt. Mithilfe des Schieberegelers kann durch den zeitlichen Verlauf der Aufnahme navigiert werden.

Im Bereich zwei, werden die vier Kamerabilder (vorne, links, rechts, hinten) dargestellt. In der Belegungskarte wird hierzu zusätzlich die Position des Fahrzeuges

und damit auch die Position der Kamera dargestellt. Durch den Schieberegeler unter den Kamerabildern, kann unabhängig von der Karte zeitlich durch den Strom der Kamerabilder navigiert werden. Die jeweilige Position des Fahrzeugs in der Karte wird dazu aktualisiert. Das hintere Kamerabild ist horizontal gespiegelt, um eine Orientierung wie im Auto, mit Rückspiegel zu ermöglichen.

Im Bereich 3 wird die Auswahl des Labels getroffen und das Labeling der Objektausrichtung vorgenommen. Ebenso kann hier zwischen Polygon- und Rechteckmodell umgeschaltet werden.

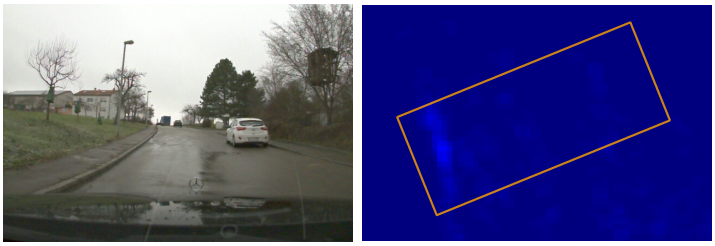
Während der Fahrt bewegt sich die Karte mit dem Fahrzeug mit. Durch die Trajektorie des Fahrzeugs, kann jedoch die Position jeder Karte im ortsfesten Koordinatensystem bestimmt werden. Die Label-Polygone werden in diesem ortsfesten Koordinatensystem gespeichert. Durch die Prämisse, dass nur die statische Welt gelabelt wird, behält das Label innerhalb der Sequenz seine Gültigkeit. Ein Label wird automatisch auf seine zeitlich benachbarten Karten übertragen. Dies verringert den Labelaufwand beträchtlich, da ein Objekt nur einmal innerhalb einer Sequenz gelabelt werden muss.

Beim Labeln beginnt der Labeler daher mit der im zeitlichen Verlauf letzten Belegungskarte. Diese enthält die meiste Information, daher ist die Form der Objekte am besten ausgebildet, und lässt sich auf die vorherigen Belegungskarten übertragen. Von dort aus wird dann rückwärts im Zeitstrahl gearbeitet. Beispielhaft lässt sich dies in Abbildung 3.2 sehen. Das Label kann seine Gültigkeit behalten, auch wenn die Belegungskarte zu einem früheren Zeitpunkt deutlich weniger belegte Zellen besitzt.

Für die Assoziation zwischen Belegungskarte und Kamera ist ein räumliches Erfassen der Szene notwendig, dies bedarf am Anfang einer Orientierungsphase. Daher werden immer zusammenhängende Messsequenzen von derselben Person gelabelt.

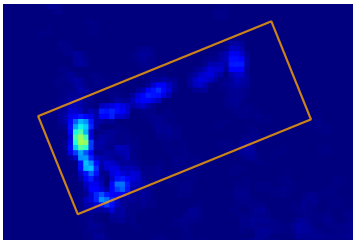
Fahrzeuge werden als Rechteck gelabelt. Die Boundingbox wird dabei so abgeschätzt, dass sie die volle Ausdehnung des Fahrzeuges repräsentiert, d. h. auch wenn beispielsweise die Belegungskarte im abgewandten Bereich des Sensors nicht voll aufgebaut ist, wird trotzdem die Länge und Breite geschätzt. Die geschätzten Dimensionen hängen daher stark vom Labeler ab. In der Praxis hat sich jedoch gezeigt, dass diese Abschätzung hinreichend genau ist.

Bei allen anderen Objekten wird nur der in der Belegungskarte sichtbare Bereich gelabelt, sollte ein Objekt jedoch in mehrere Objekte zerfallen, wird die Boundingbox um alle Teilobjekte gezeichnet. Objekte, die zu groß für eine Karte sind, z. B. ein Bordstein, werden mithilfe von mehreren möglichst großen Polygonen gelabelt.

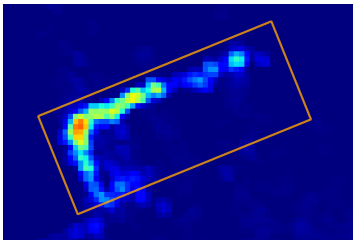


(a) Bild der Szene zum Zeitpunkt t_0

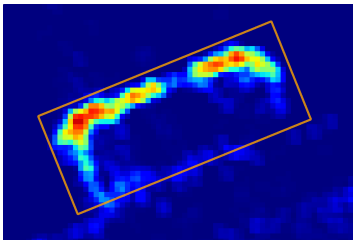
(b) Belegungskarte zum Zeitpunkt t_0



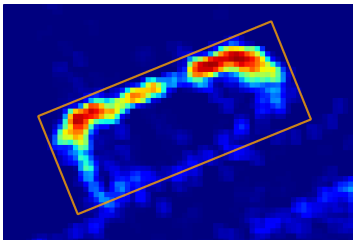
(c) Belegungskarte zum Zeitpunkt t_1



(d) Belegungskarte zum Zeitpunkt t_2



(e) Belegungskarte zum Zeitpunkt t_3



(f) Belegungskarte zum Zeitpunkt t_4

Abbildung 3.2: Die Abbildung zeigt ein Label zu verschiedenen Zeitschritten der Belegungskarte. Die Ausschnitte b-f enthalten das Label des weißen Fahrzeugs das im Bild a zu sehen ist. Das orange Label wurde händisch zum Zeitpunkt t_4 erstellt und danach auf die anderen Zeitschritte mithilfe eines globalen Koordinatensystems übertragen.

Da nur die statische Welt von Interesse ist, kann das Label leicht auf die Messungen zurück projiziert werden. Eine erneute Datenverarbeitung mit anderen Parametern kann so das Label ableiten. Alle Karten basierenden Verarbeitungsmethoden können die Label-Polygone übernehmen, indem sie die Polygone in das dementsprechende Koordinatensystem projizieren.

Objekte die sich bewegen, werden mittels ihrer Dopplergeschwindigkeit gefiltert und nicht in die Belegungskarte eingetragen. Artefakte von dynamischen Objekten, die trotz der Filter in der Belegungskarte entstehen, werden als solche markiert.

Durch das Labeln in der 2-D Welt, kommt es dazu, dass sich Objekte überlappen können. So befindet sich beispielsweise ein Pfosten eines Verkehrsschildes auf einem Bordstein. In diesem Fall soll der Labeler erst das große Objekt, also den Bordstein, als gesamtes Labeln und dann den Pfosten markieren, indem ein zusätzliches Label-Polygon über den Bereich gezeichnet wird.

Eine Auswertung der Labelergebnisse erfolgt in Abschnitt 3.4.

3.3 Taxonomie

Um Objekte zu klassifizieren ist eine geeignete Taxonomie vonnöten, welche eine gute Einteilung der Daten in Objektkategorien ermöglicht. Dies wird im Wesentlichen durch zwei Aspekte bestimmt. Zum einen aus den Möglichkeiten, bei welchen Objektkategorien eine Trennung möglich ist, zum anderen, welche Kategorisierung aus Sicht der Anwendung benötigt wird.

Da diese Arbeit nicht auf eine spezifische Anwendung zugeschnitten ist, sondern das Potenzial der semantischen Klassifikation untersuchen soll, kann die Frage zur Anwendung nur spekulativ mit Blick auf mögliche Anwendungen beantwortet werden.

Bei der Lokalisation sind bewegliche Objekte störend, da diese bei einer erneuten Befahrung möglicherweise nicht mehr vorhanden sind, weitere bewegliche Objekte auftauchen und somit Landmarken fehlen oder zusätzliche Landmarken vorhanden sind, die falsch assoziiert werden können. Das weit schlimmere Szenario ist jedoch, wenn sich die Landmarken nur geringfügig verschieben, wie z. B. eine Parkreihe von Autos, die bei einer weiteren Befahrung um einen halben Meter versetzt steht. Dieses starke Bias kann dann zu Fehlern in der Lokalisierung führen. Diesem Problem kann zum einen mithilfe von Mehrfachbefahrungen begegnet werden [Rap+16a], welches die Zuverlässigkeit der Landmarken an der Häufigkeit des wiederholten Auffindens bewertet. Dies setzt jedoch eine Mehrfachbefahrung voraus, die ohne die Bewertung von Landmarken auskommen muss. Hier kann die Semantik der Objekte helfen,

Tabelle 3.2: Beschreibung der verwendeten Label.

Label	Beschreibung
Fahrzeug	Parkende Fahrzeuge, bei diesem Label wurde zudem versucht den Raum abzuschätzen den das Fahrzeug belegt.
Wand	Gemauerte oder aus Beton gegossene Begrenzer, welche jedoch nicht zu einem Gebäude gehören.
Zaun	Zäune entweder aus Holz oder Metall.
Gebäude	Alle gebäudeähnlichen Strukturen, wie Häuser, Garagen und Industriehallen.
Bordstein	Bordsteine aller Höhen, falls diese in den Radar-Grids erkennbar sind.
Baum	Alle Arten von Bäumen.
Vegetation	Alle Pflanzen die keine Bäume sind, jedoch im Radar zu sehen sind, wie z. B. Büsche, hohes Gras, etc.
Pfosten	Alle Arten von Pfosten, wie z. B. für Verkehrsschilder.
Andere	Alle Objekte die keiner anderen Klasse zugeordnete werden können und in den Radar-Grids zu sehen sind.
bewegt	Alle Objekte, die in der Belegungskarte erscheinen, aber sich während der Vorbeifahrt bewegen z. B. fahrende Autos, laufende Fußgänger etc.
nicht gelabelt	Regionen die aus irgendwelchen Gründen nicht gelabelt werden konnten z. B. ein fehlendes Kamerabild.

um bewegliche Objekte zu erkennen und diese als Landmarken auszuschließen. Im normalen Straßenverkehr sind die meisten mobilen Objekte parkende Fahrzeuge.

Durch die Objektkategorie kann auch auf die Qualität der Landmarken geschlossen werden. Ein Pfosten stellt ein sehr gutes Reflexionszentrum aus jeder Richtung dar, während bei einer Mauer das Reflexionszentrum an der Mauer entlang wandern kann.

Weiterhin sind Objekte interessant, die sich aufgrund des Einflusses der Jahreszeiten

verändern, wie z. B. Pflanzen, da diese je nach Jahreszeiten unterschiedliche Ausprägungen in Radar-Belegungskarten haben können. Dies kann bei der Lokalisierung berücksichtigt werden.

Für die Zuordnung von Radarobjekten zu Messdaten aus anderen Sensoren oder für die Zuordnung von Radarobjekten zu einer Karte, können Objektkategorien ebenfalls hilfreich sein. Dies kann neben der Position ein weiteres Assoziationsmerkmal sein. Hierfür ist es jedoch notwendig, dass die zu assoziierenden Sensoren oder Karten ebenfalls semantische Informationen einer ähnlichen Taxonomie enthalten.

Beim Verfolgen von Objekten hat die Klasse Fahrzeuge eine wichtige Bedeutung, da sie Entstehungspunkte von sich bewegenden Objekten sein können. Das würde auch für andere Verkehrsteilnehmer wie Fahrradfahrer und Fußgänger gelten. Jedoch sind diese nach Einschätzung des Autors mit den in der Arbeit eingesetzten Sensoren nur in bewegter Form klassifizierbar.

Für das autonome Fahren, stellt ein semantisches Modell der Umgebung eine besondere Bedeutung dar, da dieses viele Fahraufgaben beeinflusst. So ist z. B. wichtig zu wissen, an welcher Grenze sich der Bürgersteig von der Fahrbahn abgrenzt. Ebenso welche Objekte den Sichtbereich einschränken oder im heimischen Bereich, dass auch ein grasbewachsener Weg als fahrbarer Bereich erkannt wird.

Ein weiteres Ziel besteht darin, die Klassen so einzuteilen, dass die Aufteilung den Labelern leicht zu vermitteln ist und diese die Kategorien auch im Kamerabild unterscheiden können. So wurde beispielsweise auf die Einteilung verschiedener Arten von Gebäuden verzichtet, auch wenn sich aus Sicht des Radars ein Industriegebäude mit einer Metallkonstruktion, von einem gemauerten Gebäude deutlich unterscheidet.

Natürlicherweise entwickelte sich die Klasseneinteilung, durch das Sammeln von Erfahrungen, über die Jahre weiter. Durch erneutes Labeln und vereinigen von Klassen konnte der Datensatz zu in Tabelle 3.2 definierten Klassen zusammengefasst werden.

3.4 Auswertung und Erfahrungen im Radar-Labeling

Die Labeler erhalten zu Beginn eine Labelanleitung, die anhand von Beispielen die zu labelnden Klassen erklärt. Zusätzlich enthält die Anleitung Hinweise, welche Besonderheiten beim Labeln zu beachten sind, z. B. wie groß die Polygone gezogen werden dürfen. Damit werden auch nicht eindeutige Fälle behandelt, z. B. welche Objekte Zäune sind oder welche Objekte schon zu einer Wand gehören.

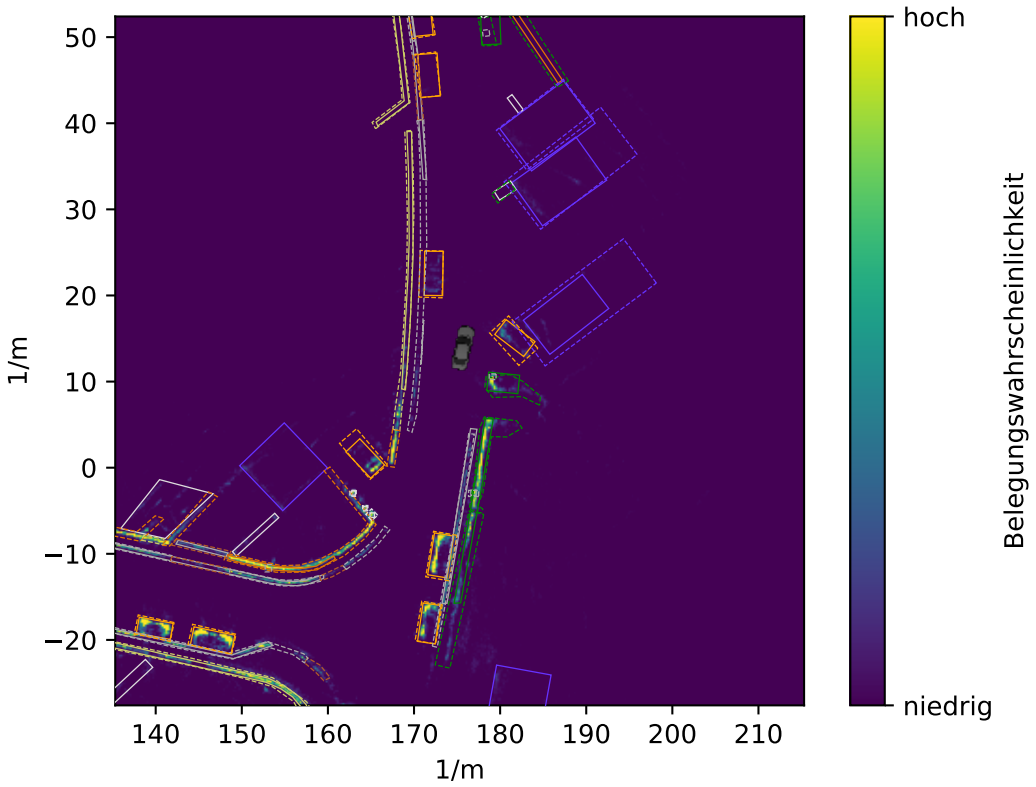


Abbildung 3.3: Von zwei Labelern gelabelte Karte. Die durchgezogenen Linien zeigen die Label des ersten Labelers, während die gestrichelten Linien die Label des zweiten Labelers zeigen.

Am Anfang wird ein neuer Labeler von einem erfahrenen Labeler eingewiesen. Dieser gibt auch Rückmeldung zu den ersten Labelergebnissen. Trotz allem gibt es immer wieder Fehlzuordnungen durch Flüchtigkeitsfehler oder Missverständnisse. Auch die Markierung der Objekte durch die Label-Polygone, wird von Labeler zu Labeler etwas unterschiedlich durchgeführt.

Diese Inkonsistenzen können durch eine Qualitätsprüfung vermindert werden. Diese Qualitätsprüfung ist jedoch ein zeitaufwendiger Prozess und wurde daher nur stichprobenartig durchgeführt.

Im Folgenden soll die Qualität des Labelings ohne nachgelagerte Qualitätsprüfung analysiert werden. Hierzu wurden Sequenzen von verschiedenen Personen unabhängig voneinander gelabelt, dabei haben alle Beteiligten mit einer leeren Karte gestartet. Insgesamt wurden 80 Sequenzen zwei- bis vierfach gelabelt.

In Abbildung 3.3 sind die Labelergebnisse zweier Labeler beispielhaft für eine Szene abgebildet. Dabei werden die Label von Labeler *A* durch durchgezogene Linien und die Label von Labeler *B* durch gestrichelte Linien dargestellt. Die Farbe codiert die Labelklasse. Hier sind verschiedene Arten von Differenzen zu erkennen:

1. Fehlende/zusätzliche Label, wenn Bereiche der Karte bei dem einen Labeler gelabelt werden, aber nicht bei dem anderen Labeler.
2. Das Zuordnen von Objekten zu unterschiedlichen Klassen.
3. Eine unterschiedliche Schätzung der Dimension eines Objektes.
4. Unterschiedliche Segmentierung von länglichen Objekten, wie Bordsteine, etc.

Differenz 1 entsteht hauptsächlich dadurch, dass nicht eindeutig definiert wurde, welche Bereiche zu labeln sind. Die Labeler wurden aufgefordert ein Objekt zu labeln, sobald dieses sich in der Belegungskarte abbildet. Da allerdings nicht feststand, ab welcher Belegungsstufe der Belegungskarte eine Klassifikation möglich ist, wurde der Bereich nicht hart eingeschränkt. Dadurch entstand natürlicherweise eine unterschiedliche Interpretation. Da Bereiche, die nicht gelabelt wurden, auch bei der Klassifikation als unbekannt angesehen werden, wirken sich Ungenauigkeiten in diesem Bereich nur gering auf den Klassifikator aus.

Differenz 2 entsteht entweder durch fehlerhaftes Labeln oder durch eine unterschiedliche Interpretation der Labelklassen. Ganz besonders die Sammelklasse *Andere* ist hierfür anfällig. Fehler in diesem Bereich beeinflussen das Klassifikationsergebnis massiv, da der Klassifikator mit fehlerhaften Beispielen trainiert wird und so bei der Auswertung falsche Schlüsse gezogen werden können.

Die Labeler sind angehalten, bei Fahrzeugen die Dimension zu schätzen, auch wenn diese nicht vollständig abgebildet ist. Bei den restlichen Objekten spielt die Dimension des Label-Polygon keine große Rolle, solange sich das gewünschte Objekte innerhalb befindet und das ganze Objekt umschlossen ist. Die Differenzen im Punkt 3 gehen daher hauptsächlich auf diese Schätzung zurück.

Die 4. Differenz der unterschiedlichen Segmentierung betrifft nur lang gezogene Objekte, wie Bordsteine, etc. Wenn es softwarebedingt zu umständlich ist, sie an einem Stück zu labeln, werden diese beliebig in mehrere Cluster unterteilt. Die unterschiedliche Segmentierung ist für die Klassifikation unbedeutend, da für diese Objekte keine Bestimmung der Instanz durchgeführt wird und für die Bestimmung der Klasse nur die gelabelte Fläche zählt.

Im Folgenden soll nun die Konsistenz der Labelergebnisse anhand der Fehler 1-3 statistisch beurteilt werden. Dies soll auf den einzelnen Objekten erfolgen, d. h. es wird eine Zuordnung der gelabelten Cluster zwischen zwei Labelern gebildet, diese

werden bezüglich ihrer Label verglichen. Des Weiteren wird der Vergleich anhand der gelabelten Fläche getroffen. Ebenso wird für Fahrzeuge anhand von Beispielen die Abschätzung der Dimension bewertet. Abschließend wird noch auf die Labelzeit eingegangen.

3.4.1 Label Konfusions-Matrix

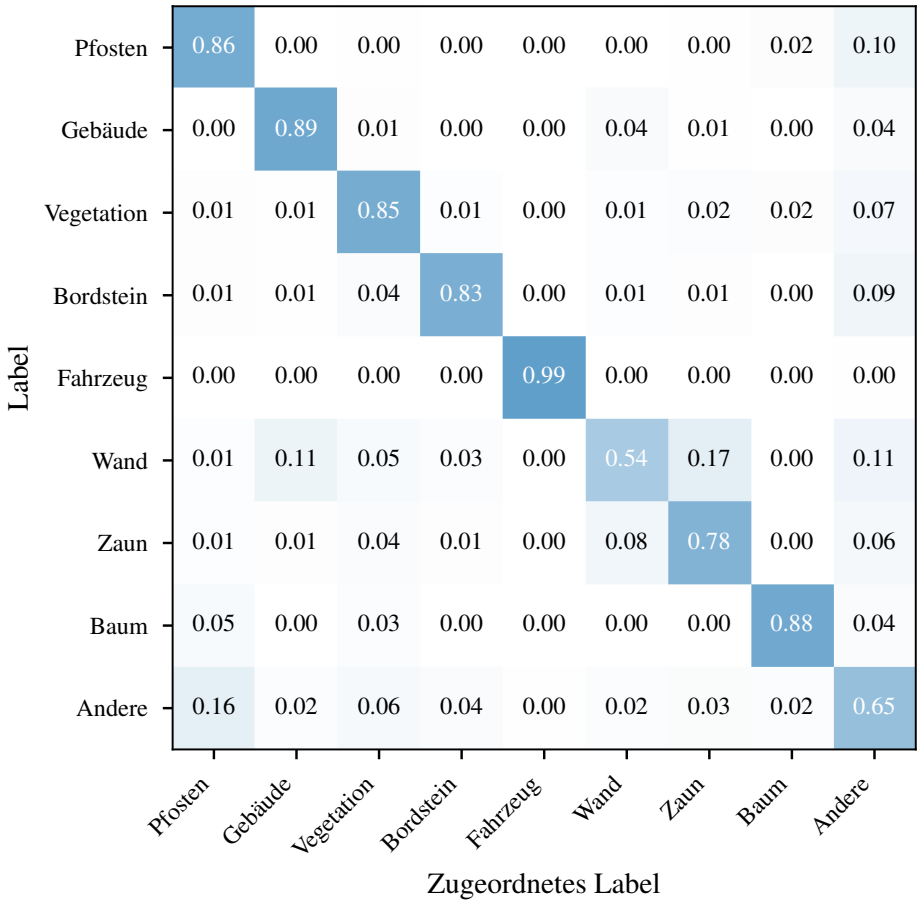


Abbildung 3.4: Konfusionsmatrix der Labeler. Erstellt mit einer minimalen IoU von 0.1. Diese ist zeilenweise normalisiert.

Um eine Evaluation auf Objekten bzw. Label-Polygonen durchführen zu können, ist zuerst eine Zuordnung dieser notwendig. Hierzu wird die Überlappungsfläche der Label genutzt. Dazu werden zuerst die Label eines Labeldurchgangs auf Überlappung

getestet. Gemäß der Labelanleitung wird die Labelfläche des kleineren Polygons vom größeren entfernt, sodass beispielsweise Pfosten von einer Vegetationsfläche ausgeschnitten werden. Somit wird der Umfang der Labelfläche auf den gültigen Bereich reduziert.

Diese korrigierten Label werden genutzt, um die verschiedenen Labeldurchgänge miteinander zu vergleichen. Es werden immer zwei Labeldurchgänge miteinander verglichen und anschließend die Ergebnisse akkumuliert. Dazu wird über die verschiedenen Durchgänge permutiert, sodass jeder Durchgang mit jedem verglichen wird. Im Folgenden werden die zwei verglichenen Labeldurchgänge als A und B bezeichnet.

Die Zuordnung der einzelnen Label erfolgt nach dem Prinzip der besten Übereinstimmung. Hierzu wird jedes Label des Durchgangs A mit jedem Label des Durchgangs B verglichen, in dem die relative Überlappungsfläche (*engl. Intersection over Union*) berechnet wird. Das Label des Durchgangs B, das die größte Überlappungsfläche aufweist, wird dem Label in A zugewiesen. Damit dies erfolgt, wird jedoch eine minimale Überlappungsfläche von 10 % gefordert. Falls keine Zuordnung erfolgen kann, wird das Label in B als fehlend markiert.

Aus dieser Zuordnung kann eine Darstellung ähnlich der Konfusionsmatrix gebildet werden. Dabei stellt die Achse *Label* die Labelklasse von A dar und die Achse *Zugeordnetes Label*, die Labelklasse von B. Durch die Permutation nimmt jeder der verglichenen Labeldurchgänge, jeweils die Rollen A und B ein. Da die relative Überlappung kommutativ ist, könnte man annehmen, dass die Matrix symmetrisch ist. Da jedoch mehreren Labeln in A dasselbe Label in B zugeordnet werden kann, ist dies nicht der Fall.

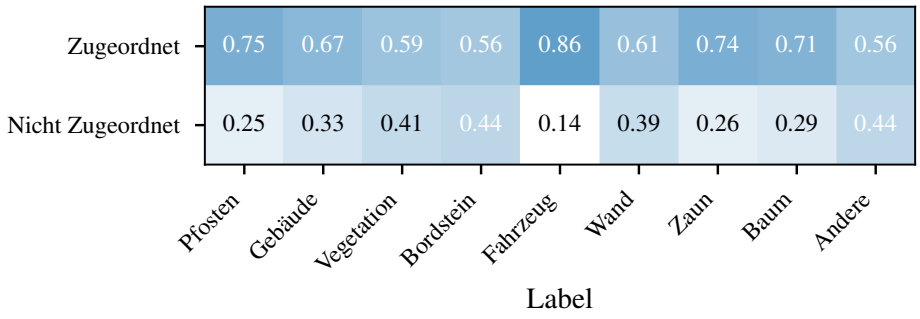


Abbildung 3.5: Anteil der Label die einem anderen zugeordnet werden können.

In Abbildung 3.4 ist eine solche Konfusionsmatrix der Label dargestellt. Der Anteil der Label bei denen keine Zuordnung stattfinden konnte, ist in Abbildung 3.5 zu

sehen. Das optimale Ergebnis wäre die Einheitsmatrix. Dies würde bedeuten, dass die Labeler die Objekte in einem ähnlichen Umfang markiert haben und der gleichen Klasse zugeordnet haben. Auch wenn dies keine Garantie für ein "richtiges" Label ist, ist die unabhängige Übereinstimmung der Personen ein starkes Indiz dafür. Dieses Ergebnis kann jedoch nur annähernd für die Klasse *Fahrzeug* erreicht werden.

Schwierig ist zum Beispiel die Unterscheidung der Klassen *Wand* und *Zaun*. Dies ist unter anderem dadurch verursacht, dass in einigen Fällen Zäune auf Wänden montiert sind. Durch die fehlende Höhenauflösung muss sich der Labeler zwischen diesen Labeln entscheiden. Auch fällt die Unterscheidung zwischen einer alleinstehenden Wand und einem Gebäude nicht immer ganz leicht.

Weiterhin ist die Klasse *Andere* aufgrund ihrer Eigenschaft als Sammelklasse sehr fehleranfällig. Hier fällt es beispielsweise schwer die Klasse *Pfosten* von Klassen die zu *Andere* gehören (z. B. Säulen, etc.) zu trennen. Insgesamt ist die Klasse *Andere* diejenige, welche die meisten Verwechslungen verursacht.

Abbildung 3.5 zeigt, dass ebenfalls ein erheblicher Prozentsatz der Label nicht zugeordnet werden kann. Dieses Problem betrifft hauptsächlich Objekte, die schwächer in der Belegungskarte abgebildet sind, weil sie teilweise von anderen Objekten verdeckt werden oder in zu großer Entfernung stehen. Es gibt auch Fälle, bei denen die Sequenz nicht zu Ende gelabelt wurde und daher in ganzen Bereichen Label fehlen. Diese zeigen, dass der Datensatz nicht an allen Stellen vollständig gelabelt ist. Dem Klassifikator fehlen dadurch einige Beispiele, er wird jedoch nicht negativ beeinflusst.

Die durchschnittliche Übereinstimmung der Label beträgt 82 %. Es ist daher zu erwarten, dass die Klassifikationsergebnisse diesen Wert nicht überschreiten. Um diesen Wert zu verbessern, können verschiedene Maßnahmen ergriffen werden. Zum einen könnten die Voraussetzungen des Labelns verbessert werden. Zudem könnten bessere Kameras eingesetzt werden, da durch das Weitwinkelobjektiv in hohen Entfernungen Objekte nur schwer erkennbar sind. Bei höherer Geschwindigkeit entsteht Unschärfe durch die Bewegung, besonders bei den seitlichen Kameras. Auch bei schwierigen Lichtverhältnissen, wie z. B. Gegenlicht oder Dunkelheit lässt die Qualität der Bilder nach.

Ein fester Einbau und eine Kalibrierung der Kameras würde es ermöglichen den Bereich, in dem sich ein Label befindet, hervorzuheben und damit die Gefahr einer örtlichen Fehlzuordnung zu vermindern.

Zusätzliche Referenzsensorik wie z. B. ein Laserscanner würde es dem Labeler erleichtern die Grenzen von Objekten zu bestimmen. Nicht zuletzt wäre ein automatisiertes

Überprüfen der Labels durch eine Projektion von Radarpunkten in ein Klassifiziertes Bild möglich. Mit Radarsensoren die eine Höhenmessfähigkeit besitzen, könnte dies auch ohne Referenzsensoren möglich sein.

Des Weiteren könnten die Labeler mithilfe eines entsprechenden Trainings besser qualifiziert werden.

Nicht zuletzt sollte eine Qualitätssicherung aufgebaut werden. Diese kann sowohl automatisch erfolgen, z. B. in dem überprüft wird, ob Pfosten ein gewisses Maß nicht überschreiten oder manuell, indem die Labelergebnisse durch einen Erfahrenen Labeler überprüft werden. Beides ist mit großem Aufwand verbunden, würde aber für deutliche Verbesserung sorgen.

3.4.2 Dimensionsabschätzung von Fahrzeugen

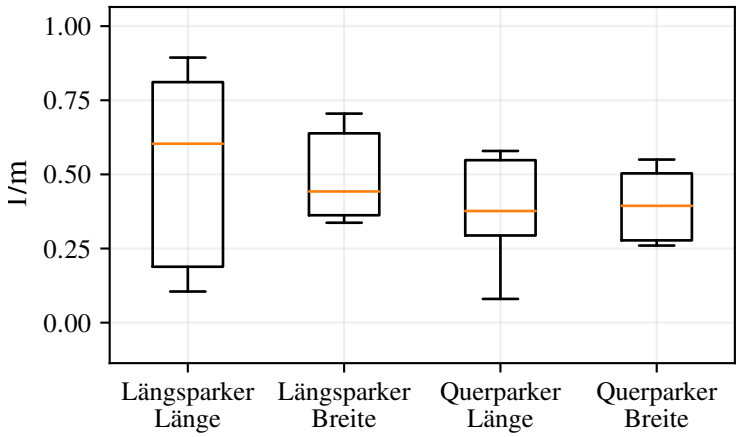


Abbildung 3.6: Box-Plot des absoluten Fehlers von Länge und Breite, jeweils für längs- und querparkende Fahrzeuge. Die Antennen geben das 5 % und 95 %-Quantil an. Bei 10 Stichproben entfällt der oberste und unterste Ausreißer.

Für das Label *Fahrzeug* sind zudem die Ausdehnung in Länge und Breite interessant. Diese sollte nach Labelanleitung auch dann abgeschätzt werden, wenn das Fahrzeug nicht komplett sichtbar ist. Da diese Abschätzungen ein starkes Bias haben können, ist es nicht möglich dies wie im vorherigen Kapitel durch einen Vergleich zweier Labeler zu tun. Es gibt auch keine Aufnahmen bei denen ein Referenzsensor, zur Bestimmung der Fahrzeuglänge, zur Verfügung steht.

Tabelle 3.3: Fahrzeuge

Fahrzeug	Länge	Breite	Geschätzte Länge	Geschätzte Breite	Ausrichtung
Smart 07-14	2.690	1.560	3.271	2.265	längs
VW Golf	4.205	1.759	4.831	2.252	längs
B-Klasse 2005	4.270	1.778	5.164	2.115	längs
Golf	4.199	1.779	5.005	2.247	längs
VW Golf IV	4.149	1.735	4.962	2.452	längs
BMW 3er Kombi	4.633	1.811	4.551	2.228	längs
Skoda Fabia Combi	4.220	1.650	4.500	2.000	längs
VW Golf IV	4.149	1.735	4.254	2.134	längs
VW Golf IV	4.149	1.735	4.307	2.038	längs
Smart	2.695	1.663	3.705	2.350	längs
VW Golf Variant	4.562	1.799	4.500	2.000	quer
VW Golf	4.205	1.759	4.660	2.309	quer
Mazda 3	4.460	1.755	4.778	2.189	quer
Hyundai i20	3.940	1.710	4.519	1.970	quer
Seat Leon	4.315	1.768	4.601	2.400	quer
Skoda Oktavia	4.569	1.769	4.971	2.282	quer
VW Golf IV	4.149	1.735	4.500	2.000	quer
Ford Fiesta 2002	3.918	1.685	4.497	2.039	quer
Ford Fiesta 2002	3.918	1.685	4.500	2.000	quer
Audi A4	4.761	1.826	4.841	2.301	quer

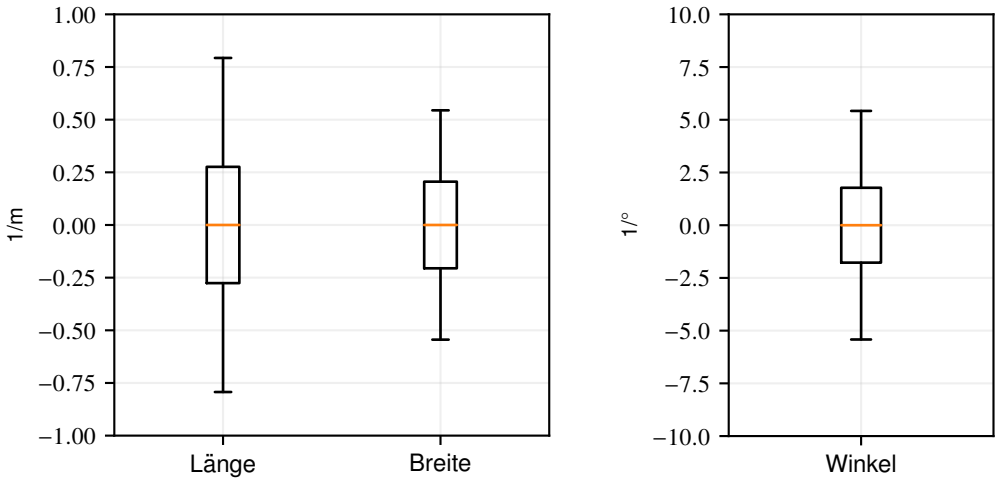


Abbildung 3.7: Box-Plot des absoluten Fehlers von Abweichung in Länge, Breite und Winkel eines Fahrzeugs. Dieser Fehler wurde anhand der mehrfachen gelabelten Sequenzen aus Abschnitt 3.4.1 bestimmt. Die Antennen geben das 5 % und 95 %-Quantil an.

Daher wurden zufällig 20 Fahrzeuge aus dem Datensatz ausgewählt, bei denen der Fahrzeugtyp bestimmt werden konnte. Die Dimension des Fahrzeuges wurde nun anhand des Datenblattes des Fahrzeugherstellers bestimmt. Für die Breite wird das Maß ohne Spiegel verwendet.

Bei den Fahrzeugen wird zwischen Längs- und Querparken unterschieden. Die Richtung wird anhand der Ausrichtung zu der Trajektorie des Sensorfahrzeugs angegeben. Diese Differenzierung wird getroffen, da beim Längsparken das Fahrzeug in seiner gesamten Länge und beim Querparken in der gesamten Breite gesehen wird. Die jeweils andere Dimension kann nur abgeschätzt werden. Die Ergebnisse dieser Auswertung sind in der Tabelle dargestellt.

Der Box-Plot in Abbildung 3.6 zeigt den absoluten Fehler bei der Schätzung von Länge und Breite, jeweils für Längs- und Querparkende Fahrzeuge. Zu erkennen ist, dass die Dimension der Fahrzeuge in der Regel überschätzt wird. Dies ist unter anderem der Varianz der Messungen in der Belegungskarte geschuldet. Hierdurch erhalten auch benachbarte, freie Zellen eine erhöhte Belegungswahrscheinlichkeit. Damit sich möglichst viele Messungen innerhalb des Label-Polygons befinden, erhielten die Labeler die Anweisung, dass das Label-Polygon alle belegten Zellen eines Objektes umschließen soll. Dies führt zu einer Überschätzung der Ausdehnung des Fahrzeugs. Lediglich bei dicht aneinander geparkten Fahrzeugen, wird die Dimension teilweise sogar unterschätzt. Hier muss eine Trennlinie zwischen zwei Fahrzeugen

gefunden werden, während die Belegungswahrscheinlichkeit der Zellen zwischen den Fahrzeugen nur etwas abnimmt.

Abbildung 3.7 zeigt die Abweichung der Dimension und des Winkels über die verschiedenen Labeldurchgänge. Für diese Auswertung werden wie im Abschnitt 3.4.1 beschrieben, die Label mittels Überlappung einander zugeordnet. Danach wird deren Länge und Breite miteinander verglichen. Da nur Schätzungen miteinander verglichen werden, kann kein absoluter Fehler bestimmt werden. Der Median im Box-Plot ist daher null. Für den Vergleich der Länge und Breite wird angenommen, dass die kürzere Seite der Box die Breite angibt.

Die Abweichung bei der Länge und Breite vom Median bewegt sich dabei in einer ähnlichen Größenordnung, wie bei den Verifikationsdaten in Abbildung 3.6. In der Ausrichtung der Box, ist eine Abweichung von wenigen Grad zu beobachten. Ein Bias-Fehler ist nur in wenigen Situationen zu beobachten. So etwas tritt unter anderem auf, wenn in einer Reihe parkender Fahrzeuge, eines etwas schräg parkt. Hier tendieren die Labeler dazu, alle Fahrzeuge mit der gleichen Ausrichtung zu Labeln. Eine genaue Bestimmung des Fehlers könnte nur mit erheblichem Aufwand erfolgen, indem die Ausrichtung der Fahrzeuge mit Vermessungstechnik nachgemessen wird.

3.4.3 Labelzeit

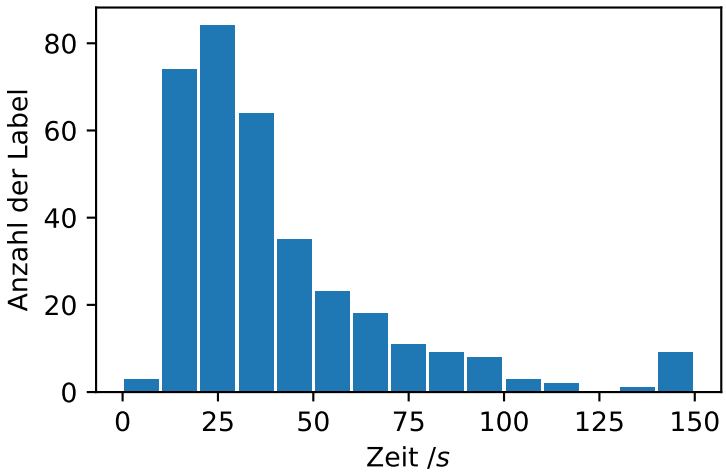


Abbildung 3.8: Histogramm der Zeit pro Polygon Label für 5 Sequenzen mit insgesamt 344 gelabelten Objekten, mit einer Länge von 140 Sekunden.

Ein weiterer wichtiger Aspekt sind die Kosten und damit die Labelzeit die benötigt wird, Objekte in Radar-Belegungskarten zu labeln. Hierfür würde sich eine statistische Erfassung über einen langen Zeitraum anbieten, indem die Zeit pro Label erfasst wird. Um dies im größeren Umfang zu tun, muss sichergestellt werden, dass nicht gegen das Datenschutzgesetz bezüglich Mitarbeiterüberwachung verstoßen wird.

Um ohne größeren Aufwand eine Abschätzung treffen zu können, wurde mit Absprache mit den jeweiligen Labelern, die Zeit exemplarisch an 5 ausgewählten Sequenzen durchgeführt.

Abbildung 3.8 zeigt ein Histogramm der Labelzeit pro Label. Zu beachten ist, dass dies nur eine grobe Abschätzung sein kann, da die Labelzeit stark von der Sequenz und den zu labelnden Objekten abhängt. Zudem hat das Labeln unter dem Bewusstsein der Zeiterfassung stattgefunden. Es ist also davon auszugehen, dass die Labeler sich beeilt haben und somit die normale durchschnittliche Labelgeschwindigkeit langsamer ist. Trotzdem kann die sich daraus ergebende durchschnittliche Labelzeit von 39 Sekunden pro Objekt als Anhaltspunkt dienen.

3.5 Zusammenfassung

In diesem Kapitel wurde ein kurzer Überblick über bestehende Datensätze gegeben. Danach wurde eine effiziente Labelingtechnik für Radardaten vorgestellt, welche auf Belegungskarten basiert. Diese nutzt aus, dass statische Objekte im Weltkoordinatensystem immer dieselbe Position haben, und daher mehrere Zeitschritte der Belegungskarten auf einmal gelabelt werden können.

Durch diese Technik können auch entlegene Teile der Belegungskarte gelabelt werden, die sich erst zu einem späteren Zeitpunkt in der Nähe des Fahrzeuges befinden. Es wurde die Güte der Label eingeschätzt, welche eine Übereinstimmung von etwas über 80 % haben. Es ist davon auszugehen, dass ein auf diesen Daten trainierter Klassifikator diese Güte nicht überschreiten wird.

Ebenso wurde die Schätzung von Fahrzeugdimensionen anhand einiger Beispiele erörtert. Mithilfe von weniger Beispiele wurde ein Anhaltspunkt für die Labelzeit gegeben. Zudem wurden Verbesserungen für das Labeling und der Qualitätskontrolle vorgeschlagen.

Kapitel 4

Daten

In diesem Kapitel werden die in der Arbeit verwendeten Datensätze erläutert. Hierbei wird zunächst der Aufbau des Versuchsträgers mit seinen Sensoren vorgestellt. Anschließend werden die Rahmenbedingungen der Datenaufzeichnung erläutert.

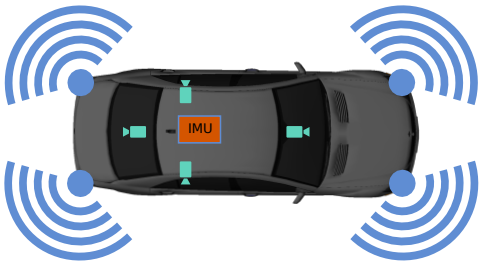
Ausgehend von den aufgezeichneten Daten und der Datenprozessierung, auf die in Kapitel 2.4 eingegangen wurde, wird in diesem Kapitel die Erstellung der Datensätze erklärt. Hierfür werden zwei Methoden verwendet. Zum einen, der Instanz basierende Datensatz, welcher für die klassischen Klassifikationsmethoden verwendet werden kann. Hierbei wird für jedes Datum ein einzelnes Label vergeben. Zum anderen, der auf einzelne Zellen basierende Datensatz für Methoden, welche auf Pixelebene klassifizieren.

Des Weiteren wird erklärt, wie die Datensätze durch Augmentierung künstlich vergrößert werden können. Um einen besseren Einblick in das Klassifizierungsproblem zu erhalten, wird abschließend der Datensatz beispielhaft und statistisch dargestellt.

4.1 Versuchsträger

Im Rahmen dieser Arbeit wurden drei Versuchsträger eingesetzt, wovon einer in Abbildung 4.1 exemplarisch dargestellt wird. Alle Versuchsträger haben einen ähnlichen Aufbau und sind mit den gleichen Radarsensoren ausgestattet. Der bedeu-

(a) Foto eines Versuchsträgers.



(b) Schematische Darstellung der Sensorpositionen.

Abbildung 4.1: Versuchsträger: In blau sind die Position und Ausrichtung der Radarsensoren dargestellt. Diese sind hinter dem Stoßfänger verbaut. In Cyan sind die Positionen der Kameras markiert. In Orange ist das optional inertiale Messsystem (IMU) dargestellt.

tendste Unterschied besteht in der Einbauhöhe der Radarsensoren und der Methode zur Bestimmung der Eigenposition.

Als Versuchsträger wurden zwei Mercedes Benz E-Klassen und ein Mercedes Benz ML eingesetzt. Die Radarsensoren wurden jeweils in den Ecken des Fahrzeugs hinter dem vorderen und hinteren Stoßfänger verbaut. Der Einbauwinkel wurde so gewählt, dass sie eine 360° Rundumsicht ermöglicht.

Bei einem Einbauwinkel von ±30° und einer lateralen Position von ±0,8 m der vorderen Sensoren, ergibt sich dabei in der Mitte des Fahrzeuges ein toter Bereich von maximal 1,2 m vor dem Stoßfänger. Wesentlich größer ist der nicht sichtbare Bereich an der Seite des Fahrzeugs. Bei einem Einbauwinkel von ±150° der hinteren Sensoren demselben lateralen Abstand und der Länge des Fahrzeugs von 4,6 m, erstreckt sich dieser bis maximal 2,3 m. Diese Entfernungen stellen jedoch die maximale Ausdehnung des dreiecksförmigen, nicht sichtbaren Bereiches dar. Bei einer Vorbeifahrt können diese Bereiche meist zu einem früheren oder späteren Zeitpunkt beobachtet werden.

Als Radar werden *FMCW*-Nahbereichssensoren eingesetzt, die eine Reichweite von bis zu 40 Metern besitzen, bei einer Bandbreite von 500 MHz. Durch den weiten Öffnungswinkel des Sichtbereichs kann jederzeit, mit Ausnahme der toten Winkel im Nahbereich, die ganze Fahrzeugumgebung beobachtet werden. Die durch den Hersteller angegebenen Parameter sind in Tabelle 4.1 zu finden.

Der Sensor ist über einen CAN-Bus mit dem Fahrzeug verbunden und hat einen

Tabelle 4.1: Parameter des Radarsensors.

Parameter	Wert
Radar-Type	FMCW
Frequenzbereich	76 GHz
Bandbreite	500 MHz
Azimut FOV	±75°
Elevation FOV	±5°
Range	40 m
Genauigkeit Range	±0,1 m
Genauigkeit Azimut	±1°
Genauigkeit Doppler Geschw.	±0,05 m/s
Winkeltrennfähigkeit	< 10°
Entfernungstrennfähigkeit	< 0,6 m
Dopplertrennfähigkeit	< 0,25 m/s

Messzyklus von ungefähr 20 Hz. Der Sensor stellt die Daten auf der Ebene der Reflexionszentren, also etwa nach der CFAR-Filterung, welche die Rauschsignale unterdrückt zur Verfügung [Sko08]. Eine Assoziation und zeitliche Filterung der Messpunkte zu Objekten findet nicht statt.

Durch die limitierte Übertragungsrate des CAN-Busses entsteht die Einschränkung, dass die Anzahl der gemessenen Reflexionszentren in einem Sensorzyklus durch eine Obergrenze beschränkt ist. Dies führt dazu, dass in Umgebungen mit vielen Objekten, schwächere Reflexionszentren nicht übertragen werden.

Die Eigenbewegung wird über die fahrzeugeigene Seriensensorik mittels Koppelnavigation (engl. *dead reckoning*) bestimmt. Als Eingangsgrößen werden die Radsensoren, der inertiale Gierratensensor und der Lenkwinkel verwendet. Diese Methode wurde für den größten Teil des Datensatzes eingesetzt. Sie hat jedoch den Nachteil, dass Störgrößen z. B. der Schlupf der Räder, ein nicht genau eingestellter Raddurchmesser oder ein zu einfaches Bewegungsmodell des Fahrzeugs zu Fehlern führen, die sich über die Zeit akkumulieren. Da die Gültigkeit der Belegungskarte räumlich beschränkt ist und diese Karte oft nach kurzer Zeit verlassen wird, ist der dadurch

entstehende Fehler meist nicht von großer Bedeutung.

Für weitere Messkampagnen stand eine D-GPS gestützte, hoch genaue Inertialsensorik zur Verfügung, welche die Giergeschwindigkeit mittels Faserkreiseln bestimmt und somit auch in Bereichen mit schlechtem GPS Empfang über lange Zeit eine genaue Eigenposition zur Verfügung stellt.

Alle Daten werden mithilfe eines Messcomputers, der über CAN mit der Sensorik verbunden ist, aufgezeichnet. Um spätere Nachverarbeitung zu ermöglichen, werden die Sensordaten aufgezeichnet. Diese entspricht bei den Radarsensoren den Reflexionszentren, die als Punktziele gespeichert werden. Zur Bestimmung der Eigenbewegung werden die Rad-Ticks, der Lenkwinkel und die Beschleunigungswerte des Gierraten-Sensors aufgezeichnet. Bei der D-GPS gestützten inertialen Messeinheit, werden die ermittelten Positionen, Geschwindigkeiten und Beschleunigungswerte gespeichert. Die Kameradaten werden in der Frequenz von etwa 10 Hz als *JPEG* komprimierte Bilder gespeichert.

4.2 Datensatz

Der gesamte Datensatz besteht aus Aufnahmen über eine Strecke von 200 km bei einer Aufnahmezeit von ca. 9 Stunden. Die Daten wurden im Zeitraum von über 3 Jahren zu verschiedenen Jahreszeiten aufgezeichnet. Die Aufnahmen bestehen zum Großteil aus urbanen Szenarien wie z. B. Seitenstraßen- und Parkplatzszenarien im Geschwindigkeitsbereich von weniger als 50 km/h. Die Routen wurden so gewählt, dass eine möglichst hohe Varianz in den Szenarien vorhanden ist.

Der aufgezeichnete Datensatz wurde, soweit die Objekte sich in der Belegungskarte abbildeten, vollständig gelabelt, sodass die Komplexität der Umgebung ohne weitere Filterung in vollem Umfang in den gelabelten Daten abgebildet wird. In welchen Bereichen Objekte in der Belegungskarte abgebildet werden, ist abhängig von der gefahrenen Geschwindigkeit, der Verdeckung von anderen Objekten, und dem Hintergrundrauschen der Umgebung. In der Regel bilden sich Objekte im Bereich von 15 m - 20 m um die vom Fahrzeug gefahrene Trajektorie ab. Alle Daten wurden im Großraum Ulm und im Großraum Stuttgart aufgezeichnet.

Aus den eingefahrenen Daten wurden Belegungskarten im Bereich von ca. $40m$ um das Fahrzeug aufgezeichnet. Die Belegungskarten bewegen sich dabei mit dem Fahrzeug mit. Im Abstand von 0,5 - 1 Sekunde werden diese Karten gespeichert.

Der Datensatz wurde anschließend, wie im Kapitel 3 beschrieben, gelabelt. Die Beschreibung der verwendeten Label sind aus Tabelle 3.2 zu entnehmen.

Des Weiteren gibt es noch das Label für *bewegte Objekte* und *nicht gelabelte* Bereiche. *Bewegte Objekte* sind alle Objekte, die sich zumindest zeitweise bewegen, während sie sich im Sensorbereich befinden. Das Label *nicht gelabelt* wird verwendet, falls das Objekt nicht erkannt werden kann (z. B. verdeckt in allen Kameras) oder wenn durch einen Fehler in den Sensoren massive Störungen in der Belegungskarte auftraten. Beides muss gesondert behandelt werden und wurde daher vom Datensatz ausgeschlossen.

4.3 Cluster-Datensatz

Zur Klassifikation von Instanzen müssen erst mögliche Vorschläge extrahiert werden. Übliche Verfahren in der Bildverarbeitung sind beispielsweise *K-means* [Mac67], histogrammbasierte Methoden, Kantendetektion oder Schwellwertverfahren. Weitere Methoden wählen die Instanzen basierend auf *Sliding Windows* oder verwenden Neuronale Netzwerke, um die Segmentierung zu bestimmen.

Belegungskarte enthalten, anders als Bilder, keine harten Kantenübergänge, so dass Kanten basierende Methoden versagen würden. Bei Belegungskarten sind jedoch die meisten Objekte von wenig belegten Zellen umgeben. Daher kann hier ein einfaches, schwellwertbasiertes Verfahren eingesetzt werden. Andere Radar basierende Karten wie z. B. die RCS-Maximum-Karte (siehe Kapitel 2) sind für ein solches Verfahren nicht geeignet, da die Zellen in einer Umgebung sehr sprunghafte Werte aufweisen können. Neben dem separieren von einzelnen Objekten muss zudem dafür gesorgt werden, dass das extrahierte Objekt nicht die maximale Eingangsgröße des Klassifikators übersteigt.

Um aus den gelabelten Daten, einen möglichst großen Datensatz zu erzeugen, werden von den Belegungskarten in regelmäßigen Zeitabschnitten Momentaufnahmen gemacht. Diese führen dazu, dass Belegungskarten mit unterschiedlichen Integrationszeiträumen für das Erzeugen des Datensatzes verwendet werden. Dies erhöht die Variantenvielfalt und spiegelt die Realität einer sich über die Zeit aufbauenden Belegungskarte wider. Dieser Effekt der zeitlichen Veränderung eines gelabelten Objektes ist in Abbildung 3.2 des Kapitel 3 dargestellt.

Im Folgenden sollen die Methoden zur Extraktion von Objektvorschlägen und das Zuweisen von Labeln beschrieben werden. Danach wird darauf eingegangen, wie der Datensatz künstlich vergrößert werden kann, die Daten in die einzelnen Datensätze (Training, Validation und Test) aufgeteilt werden und wie mit den ungleich verteilten Klassen umgegangen wird.

4.3.1 Segmentierung

Für die Segmentierung wird die Belegungskarte im Intervall T abgetastet, in Abbildung 4.2a ist eine solche Karte zu sehen. Zudem sind in diesem Bild noch die Label-Polygone eingezeichnet. Diese Auszüge von einer Belegungskarte werden in eine binäre Karte überführt. Dies wird mithilfe von Schwellwerten realisiert. Hieraus werden dann mittels eines *Connected-Component Algorithmus* [PLC00] mögliche Objekte extrahiert.

Die Herausforderung besteht darin, Objekte aus verschiedenen Bereichen der Belegungskarten zu clustern. Objekte die über eine lange Zeit beobachtet wurden und daher stärker in die Sättigung gehen, sowie ferne Objekte, auf die nur wenige Messungen fallen. Es ist schwer, diesen Effekt mithilfe eines geeigneten Sensormodells zu kompensieren und lässt sich daher nur annäherungsweise für einfache Situationen, ohne Verdeckungen, etc., modellieren. Zudem besteht das Problem, dass bestimmte Regionen mehr Clutter und andere weniger enthalten. So erzeugen unebene Flächen, wie Gras oder Schotterstraßen mehr Clutterziele, als eine geteerte Straße.

Um dem zu begegnen, wird der Schwellwert dynamisch für jede Zelle gewählt. Das Verfahren, um diesen Schwellwert zu bestimmen, wurde aus der Radar Signalverarbeitung entlehnt, welcher dort als CA-CFAR (engl. *cell averaging constant false alarm rate*) bekannt ist [Roh11]. Dieser wurde auf 2-D erweitert. Hierzu wird der Mittelwert $\mu_{i,j}$ der Belegungswahrscheinlichkeit in der Nachbarschaft, um die Zelle $m_{i,j}$ bestimmt. Hierzu werden die Zellen mit einem Abstand von n in x und y Richtung herangezogen. Um den Einfluss des zu extrahierenden Spitzenwertes zu limitieren, wird ein kleiner Bereich g von dieser Summierung ausgenommen. Dies entspricht den *guard cells* des CFAR-Algorithmus. Der Mittelwert wird dann mit dem Faktor S skaliert.

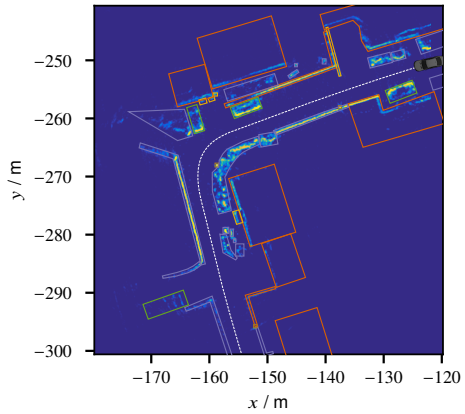
Zusätzlich wird noch ein kleiner konstanter Anteil τ_{const} addiert. Dieser sorgt dafür, dass in freien Regionen nicht zu viele Clutter Objekte entstehen.

Der Schwellwert berechnet sich durch

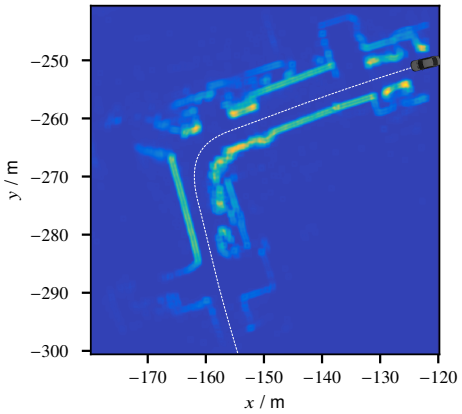
$$\tau_{i,j,thresh} = \tau_{const} + S\mu_{i,j}, \tag{4.1}$$

mit

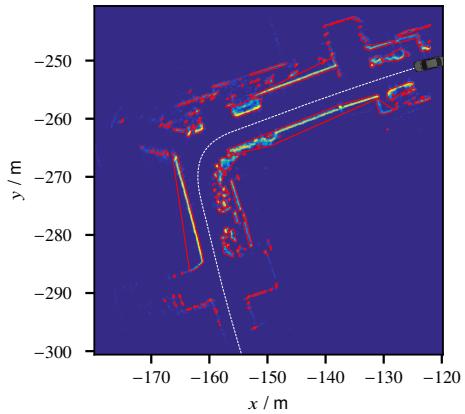
$$\mu_{i,j} = \frac{\sum_{k=-n}^{n}\sum_{l=-n}^{n} {}^{occ}s_{i+k,j+l} - \sum_{k=-g}^{g}\sum_{l=-g}^{g} {}^{occ}s_{i+k,j+l}}{(2n+1)^2 - (2g+1)^2}. \tag{4.2}$$



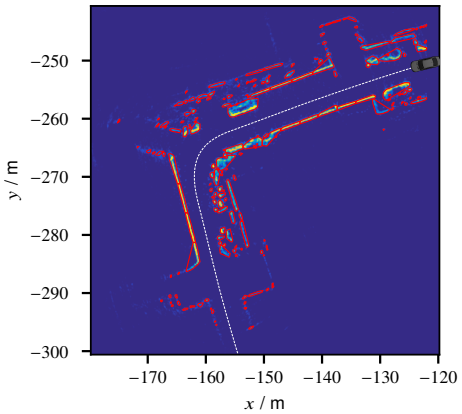
(a) Belegungskarte mit Label-Polygonen.



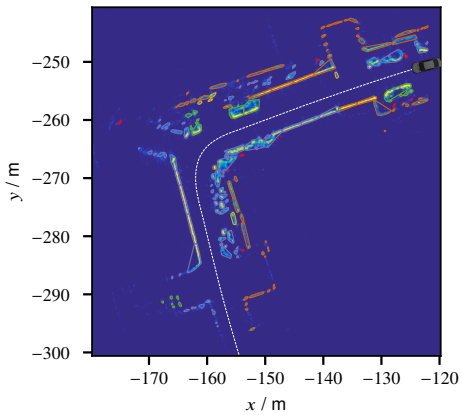
(b) Schwellwertkarte.



(c) Extrahierte Cluster.



(d) Große Cluster in Teilcluster zerlegt.



(e) Cluster mit zugewiesenen Labeln.

Abbildung 4.2: Verarbeitungsschritte beim Erzeugen des Cluster-Datensatzes.

Dadurch entsteht eine Schwellwertkarte, die beispielhaft in Abbildung 4.2b zu sehen ist. Mit dieser wird die Belegungskarte in eine binäre Karte überführt.

Mit der *Connected Component* Analyse können aus der binären Karte Cluster erzeugt werden. Um kleinere Objekte, die durch *Clutter* entstanden sind, zu unterdrücken, werden in einem nächsten Schritt Objekte deren Fläche kleiner ist als a_{min} verworfen. Die Cluster werden wie in Abbildung 4.2c gezeigt, als konvexe Hüllen extrahiert.

Die meisten der in der Arbeit verwendeten Klassifikationsverfahren haben eine fest zu wählende Eingangsgröße. In manchen Fällen, wie beispielsweise bei Häusern oder Bordsteinen, wird diese Größe überschritten. Deshalb ist es in diesen Fällen notwendig, das Objekt in mehrere Teilobjekte zu zerlegen, diese Teilobjekte zu klassifizieren und sie anschließend wieder zusammenzufügen.

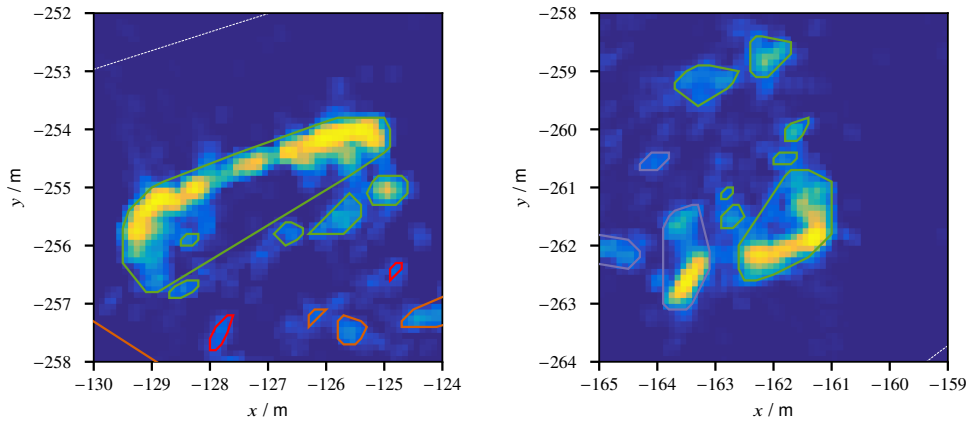
Zu diesem Zweck werden Objekte, deren Ausdehnung in x oder y die maximale Größe s_{max} übersteigt, durch wiederholtes Halbieren zerlegt. Algorithmus 6 zeigt hierbei die Funktion, die zur Aufteilung verwendet wird. Diese ist rekursive definiert, und wiederholt damit den Teilungsprozess bis die gewünschte Clustergröße erreicht worden ist.

Algorithmus 6 : Anwendungen von Label auf Cluster

```
1  Function Split(cluster, maxsize)
2      width <- berechne breite des Cluster ;
3      height <- berechne höhe des Cluster ;
4      if width > s_max or height > s_max then
5          if width > height then
6              cluster1, cluster 2 := halbiereClusterHorizontal(cluster) ;
7              return concat(Split(cluster1), Split(cluster2));
8          else
9              cluster1, cluster 2 := halbiereClusterVertikal(cluster) ;
10             return concat(Split(cluster1), Split(cluster2));
11         end
12     else
13         return cluster;
14     end
```

Eine weitere Schwierigkeit bei der Objektextraktion besteht darin, dass Objekte oftmals in mehrere Cluster zerfallen. Dies ist zum Beispiel bei *Autos* der Fall, bei denen nicht nur die Teile der zugewandten Seite, sondern in vielen Fällen auch die

Räder der abgewandten Seite zu sehen sind. Dies ist sowohl für längs, wie auch quer geparkte Autos der Fall. Zwei Beispiele dafür sind in Abbildung 4.3 zu sehen.



(a) Quer geparkt.

(b) Längs geparkt.

Abbildung 4.3: In grün sind die alle extrahierten Cluster zu sehen die jeweils dem quer bzw. längs geparkten Fahrzeug zuzuordnen sind.

Dieses Phänomen wird jedoch bei den Ausschnitts basierenden Lernen ignoriert. Jedes Cluster, das zu einem Objekt gehört, wird als eigenständige Cluster-Instanz betrachtet. So kann beispielsweise aus einem Auto mehr als eine Cluster-Instanz entstehen, diese werden dann separat voneinander klassifiziert.

Diese können nach der Klassifikation wieder zueinander assoziiert werden. Dazu kann unter anderem auch das Klassifikationsergebnis genutzt werden.

Eine weitere Problematik sind Cluster am Rand der Belegungskarten, da hier ein Objekt künstlich abgeschnitten wird. Um dies zu verhindern, werden Objekte die vollständig im Randbereich der Karte liegen entfernt und gehen nicht in den Datensatz ein. Dies betrifft alle Objekte die näher als die maximale Clustergröße s_{max} am Rand liegen.

4.3.2 Zuweisung der Label

Für das Bilden der Cluster wird die Labelinformation nicht hinzugezogen. Das Clustering wird damit nicht durch das Label beeinflusst und kann daher auch auf nicht gelabelten Daten eingesetzt werden.

Erst in einem zweiten Schritt wird das Label mit dem Cluster verknüpft. Hierbei wird die belegte Fläche eines Clusters hinzugezogen, welche durch die belegten Zellen in der binären Karte, und nicht durch die Fläche der konvexen Hülle definiert wird.

Die belegte Fläche muss sich mindestens zu einem Anteil von τ_{IoC} innerhalb des Label-Polygons befinden.

$$\frac{Cluster \cap Label}{Cluster} > \tau_{\mathrm{IoC}} \tag{4.3}$$

Falls dies auf mehrere Label zutreffen sollte, wird in einem zweiten Schritt das Label ausgewählt, welches das Label zum größeren Teil ausfüllt. Welches also die relative Überschneidung im Verhältnis zur Labelfläche IoL maximiert.

$$IoL = \frac{Cluster \cap Label}{Label} \tag{4.4}$$

Durch dieses Vorgehen erreicht man, dass bei überlappenden Labels in der Regel das kleinere Label dominiert.

4.3.3 Augmentation

Um mehr Daten für das Training des Klassifikators zur Verfügung zu haben, können diese künstlich erzeugt werden. Da ein Erzeugen der Daten von Grund auf oft nicht oder nur sehr schwer möglich ist, werden vorhandene Trainingsdaten so verändert, dass hierbei eine größere Vielfalt der Daten entsteht.

Hierzu werden verschiedene Augmentationstechniken eingesetzt, die im Folgenden vorgestellt und motiviert werden sollen. Da sich dadurch die Anzahl der Trainingsbeispiele vervielfacht, werden die Daten *online* während des Trainings augmentiert. Dies erleichtert auch das Verändern der Augmentierungsparameter in unterschiedlichen Trainingsdurchgängen.

Die Augmentierungsparameter geben dabei Wahrscheinlichkeitsdichten vor, aus denen die einzelnen Parameter der Transformationsalgorithmen gezogen werden. Bei der Rotation, Translation und Skalierung werden diese Parameter einmal für alle Kanäle gezogen, sodass die Ausschnitte aller Kanäle gleichbehandelt werden. Für alle anderen Augmentationen werden die Parameter für jeden Kanal unabhängig voneinander gezogen.

Im Folgenden werden nun die dementsprechenden Techniken vorgestellt und jeweils an Beispielen anschaulich gemacht. Für diese Beispiele wurden Ausschnitte von verschiedenen Klassen ausgesucht. Um einen besseren Eindruck zu bekommen, wurden die Augmentierungsparameter fest gewählt, auch wenn sie in der Praxis aus einer Zufallsverteilung gezogen werden.

4.3.3.1 Translation

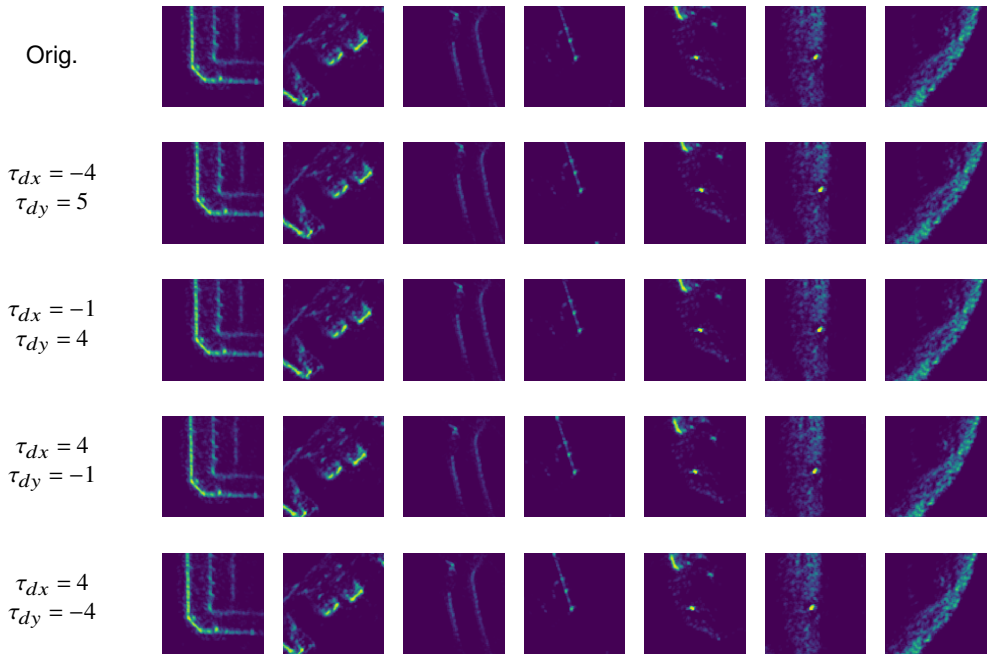


Abbildung 4.4: Beispiele der Translation-Augmentierung. In Zeile 1 sind die originalen Ausschnitte zu sehen. Zeile 2-5 zeigt die augmentierten Ausschnitte mit den jeweiligen Parametern.

Durch geringfügige Varianzen in der Belegungskarte kann der Mittelpunkt des Clusters abhängig von der Objektinstanz, dem Betrachtungswinkel, aber auch der Beobachtungslänge variieren. Der Objektausschnitt wird zufällig um τ_{dx}, τ_{dy} verschoben. Hierbei werden τ_{dx} und τ_{dy} gleichverteilt aus dem Intervall $[-\boldsymbol{\tau}_{trans}, \boldsymbol{\tau}_{trans}]$ gezogen.

Abbildung 4.4 zeigt Beispiele der augmentierten Ausschnitte mit den jeweiligen Parametern.

4.3.3.2 Rotation

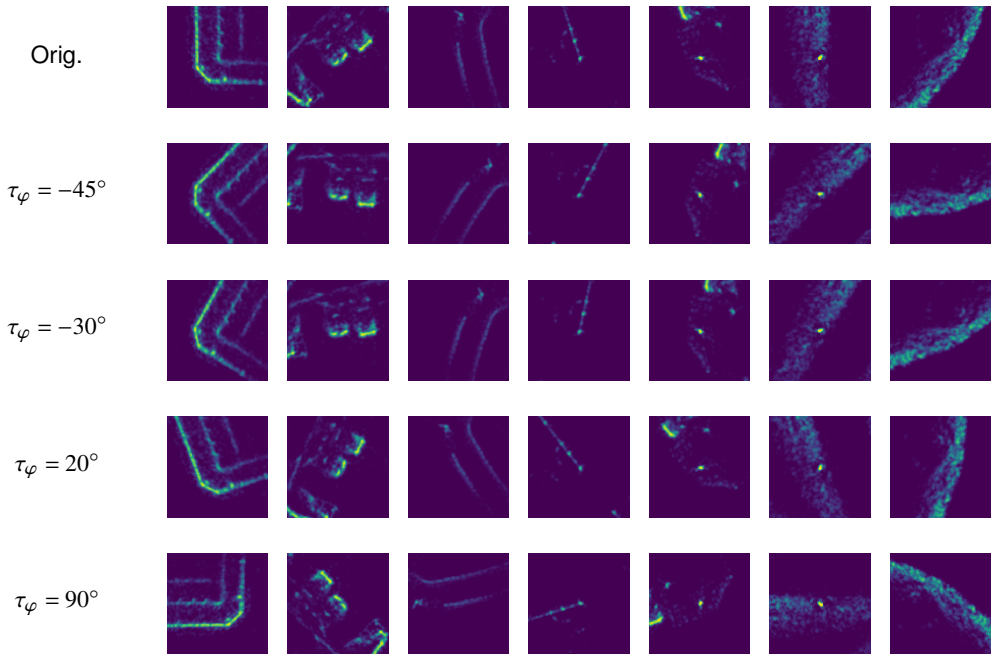


Abbildung 4.5: Beispiele der Rotations-Augmentierung. In Zeile 1 sind die originalen Ausschnitte zu sehen. Zeile 2-5 zeigt die augmentierten Ausschnitte mit den jeweiligen Parametern.

Belegungskarten haben keine natürliche Orientierung, wie es beispielsweise bei Bildern gegeben ist (Vorausgesetzt die Kameraorientierung in der Welt ist bekannt). So gibt es in Bildern beispielsweise bei Häusern, Autos, etc. eine natürliche Orientierung von oben und unten. Dies ist bei Belegungskarten nicht der Fall, da das Sensorfahrzeug dort eine beliebige Orientierung haben kann. Damit kann auch jedes Objekt, das sich in der Belegungskarte befindet, beliebig gedreht sein.

Unter Umständen sind typische Orientierungen für Objekte untereinander vorhanden, z. B. parkt ein Fahrzeug in der Regel längs oder quer zum Bordstein. Dies ist jedoch nicht immer der Fall und lässt keine Rückschlüsse bei der Betrachtung eines einzelnen Objektes zu.

Aus diesem Grund wird der Objektausschnitt zufällig um τ_φ im Bereich von $\pm\tau_{\text{rot}}$ gedreht. Da das neu aufbauen der Merkmalskarten in einem gedrehten Zustand zu aufwendig ist, werden Bildverarbeitungsalgorithmen mit bilinearer Interpolation zum Drehen der schon bestehenden Kartenausschnitte eingesetzt.

Beispiele dieser Augmentierungstechnik sind in Abbildung 4.5 zu finden.

4.3.3.3 Skalierung

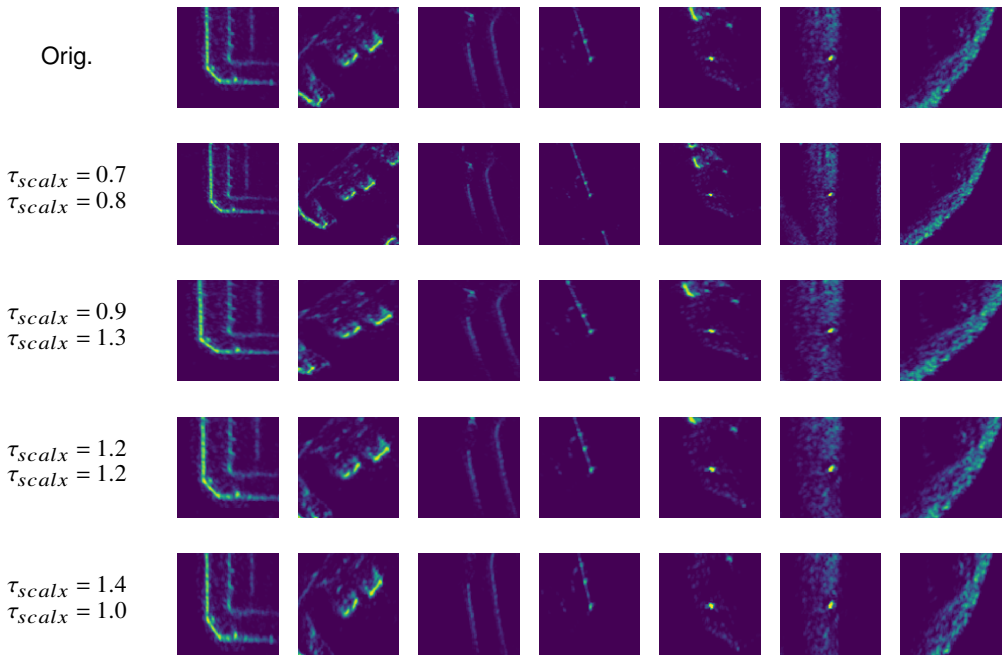


Abbildung 4.6: Beispiele der Skalierungs-Augmentierung. In Zeile 1 sind die originalen Ausschnitte zu sehen. Zeile 2-5 zeigt die augmentierten Ausschnitte mit den jeweiligen Parametern.

Reale Objekte existieren in unterschiedlichen Größen, zudem bildet sich die Größe durch die Varianz des Sensors und durch Effekte wie Mehrwegeausbreitung unterschiedlich ab. Um die Robustheit in der Größendifferenz von Objekten zu erhöhen, werden die Ausschnitte skaliert. Der Objektausschnitt wird zufällig um $\tau_{scalx}, \tau_{scaly}$ skaliert. Hierbei werden τ_{scalx} und τ_{scaly} gleichverteilt aus dem Intervall $[-\tau_{scal}, \tau_{scal}]$ gezogen. Wie bei der Rotation wird auch hier eine lineare Interpolation eingesetzt.

Beispiele dieser Augmentierungstechnik sind in Abbildung 4.6 zu finden.

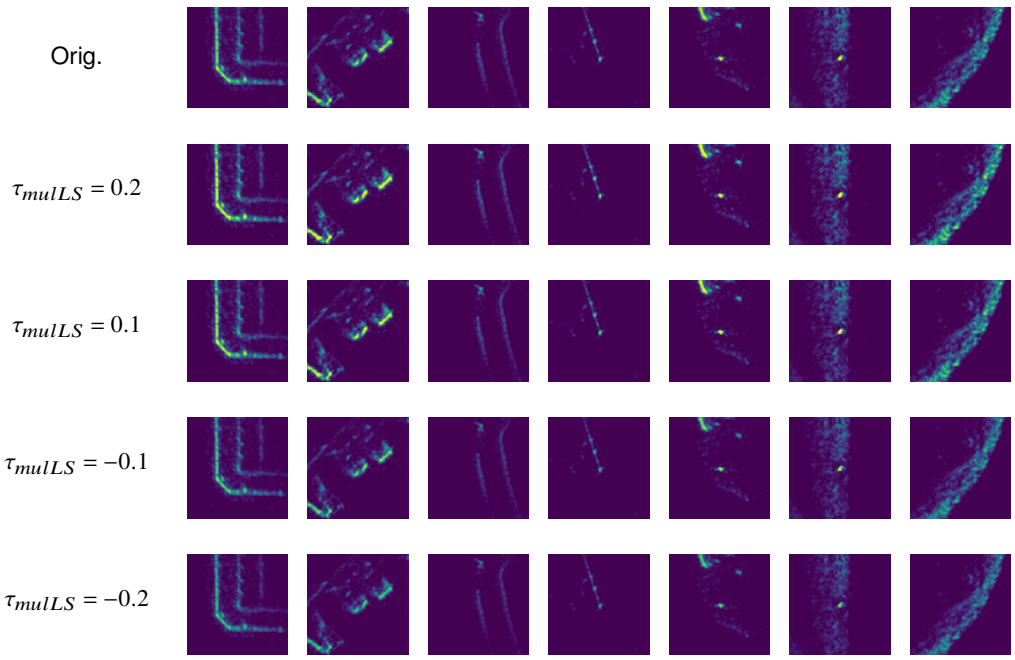


Abbildung 4.7: Beispiele der Multiplikativen-Augmentierung. In Zeile 1 sind die originalen Ausschnitte zu sehen. Zeile 2-5 zeigt die augmentierten Ausschnitte mit den jeweiligen Parametern.

4.3.3.4 Multiplikativer Levelshift

Je nach Sensormodell neigen Belegungskarten dazu, bei längerer Beobachtungszeit desselben Gebietes, mehr in die Sättigung zu gehen als bei kürzerer Beobachtungszeit. Solche Effekte treten beispielsweise bei Befahrungen mit unterschiedlichen Geschwindigkeiten oder durch Verdeckung eines Gebietes auf.

Um die Robustheit der Klassifikator für diesen Effekt zu erhöhen, wird der Wert jeder Zelle mit einem Faktor τ_{mulLS} multipliziert und auf den Minimal- und Maximalwert der Karte gedeckelt. Dieser Faktor wird aus dem Bereich $[\tau_{\text{tmulLS,min}}, \tau_{\text{mulLS,max}}]$ gleichverteilt gezogen.

$$m_{i,j,aug} = \max(\min(m_{i,j} * (1 + \tau_{mulLS}), m_{max}), m_{min}) \tag{4.5}$$

In geringerem Umfang kann dies auch für RCS-Karten gelten, da fokussierende Streuzentren erheblich größeren Schwankungen des RCS-Wertes unterliegen, als diffuse Rückstrahler.

Beispiele dieser Augmentierungstechnik sind in Abbildung 4.7 zu finden.

4.3.3.5 Additiver Levelshift

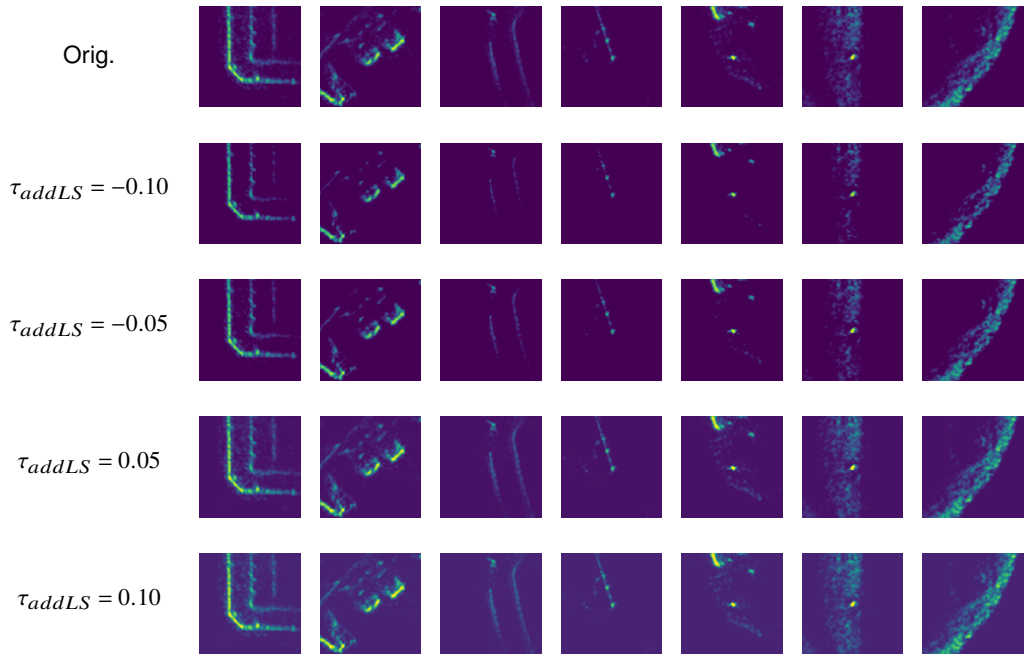


Abbildung 4.8: Beispiele der Additiven-Augmentierung. In Zeile 1 sind die originalen Ausschnitte zu sehen. Zeile 2-5 zeigt die augmentierten Ausschnitte mit den jeweiligen Parametern.

Gleichzeitig gibt es je nach Umgebung unterschiedliches Rauschverhalten. So erzeugen beispielsweise verschiedene Oberflächen, wie Teer oder Schotter unterschiedlichen Mengen an Clutter.

Dieses Verhalten ist additiv, da sich das gesamte Belegungsniveau der Karte hebt oder senkt. Daher wird das Niveau der Karte um τ_{addLS} additiv verschoben. Der Parameter τ_{addLS} wird aus dem Bereich $[\tau_{\text{addLS,min}}, \tau_{\text{addLS,max}}]$ gleichverteilt gezogen.

$$m_{i,j,aug} = \min(\max(m_{i,j} + (1 + \tau_{addLS}), m_{max}), m_{min}) \tag{4.6}$$

Beispiele dieser Augmentierungstechnik sind in Abbildung 4.8 zu finden.

4.3.3.6 Rauschen

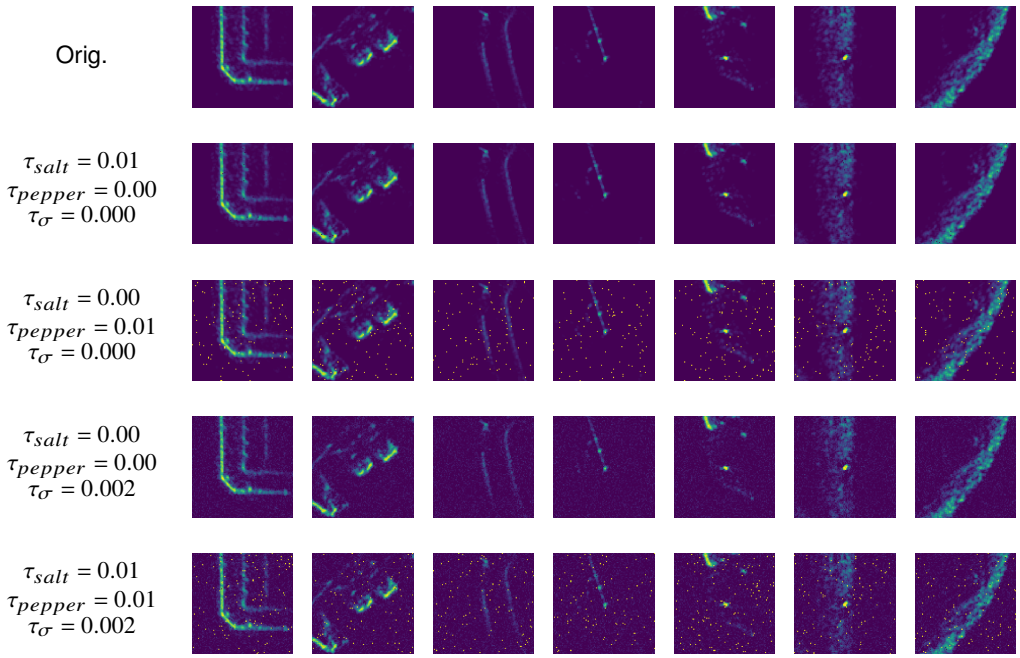


Abbildung 4.9: Beispiele der Rausch-Augmentierung. In Zeile 1 sind die originalen Ausschnitte zu sehen. Zeile 2-5 zeigt die augmentierten Ausschnitte mit den jeweiligen Parametern.

Zusätzlich zu den obigen Methoden erzeugt ein additives *Salt-and-Pepper* Rauschen und ein additives weißes gaußsches Rauschen weitere Vielfalt in den Trainingsdaten. Dies hilft einen robusten Klassifikator zu erhalten und Überanpassung zu vermeiden. Anders als die vorherigen Augmentierungstechniken, erzeugt das Rauschen lokale Unterschiede der Zellen, während die vorherigen Methoden im gleichen Maße auf alle anderen Zellen angewendet wurden.

Die Parameter τ_{salt}, τ_{pepper} und τ_{σ} werden hierbei gleichverteilt aus den Bereichen $\tau_{salt,min}$, $\tau_{salt,max}$, $\tau_{pepper,min}$,$\tau_{pepper,max}$ $\tau_{\sigma,min}$,$\tau_{\sigma,max}$ gezogen.

Beispiele dieser Augmentierungstechnik sind in Abbildung 4.9 zu finden.

4.3.4 Aufteilung der Datensätze

Der Datensatz wird für das Training und die Evaluation in drei Datensätze aufgeteilt, dem Trainingsdatensatz (70 %), dem Validierungdatensatz (15 %) und dem

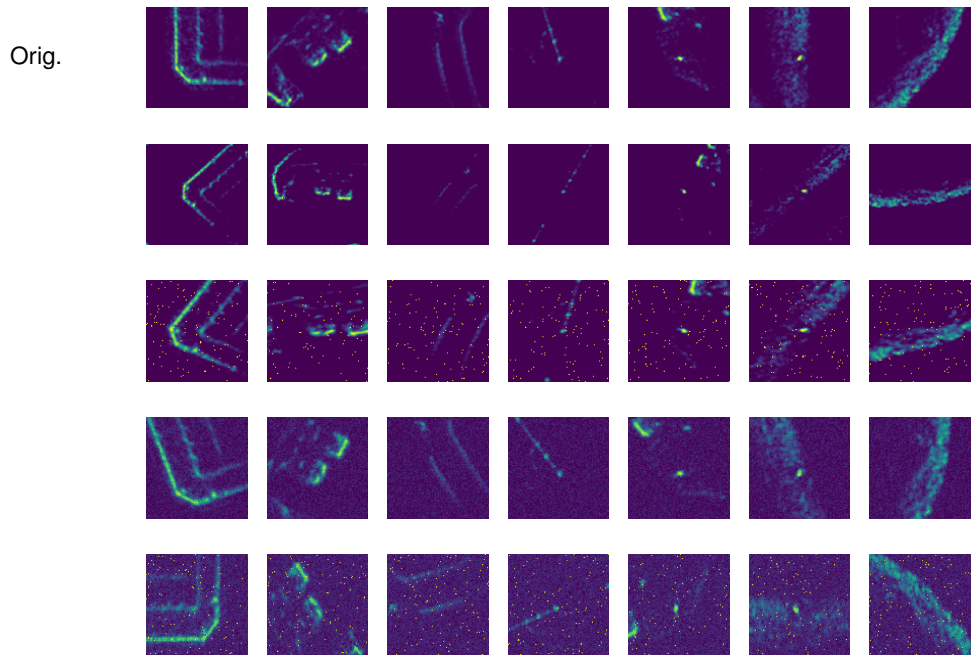


Abbildung 4.10: Beispiele der Kombination aller Augmentierungstechniken. In Zeile 1 sind die originalen Ausschnitte zu sehen. Zeile 2-5 die augmentierten Ausschnitte, die Parameter wurden von den vorherigen Beispielen übernommen.

Testdatensatz (15 %).

Die Beispiele werden zufällig anhand der gelabelten Objekte auf die verschiedenen Datensätze aufgeteilt. Dies ermöglicht zum einen große Varianz in allen Datensätzen. Zum anderen verhindert es, dass korrelierte Daten in der Belegungskarte auf verschiedene Datensätze verteilt werden. Diese Korrelation der Belegungskarte kommt dadurch zustande, dass für eine Karte zum Zeitpunkt t Informationen aus Messungen vom Zeitbereich $[0 : t]$ akkumuliert werden und eine Karte zum Zeitpunkt $t + 1$ von Messung aus dem Zeitbereich $[0 : t + 1]$ verursacht wird.

Aufgrund der relativ kleinen Datenmenge hängt das Klassifikationsergebnis nicht nur von dem Experiment, sondern auch von der Aufteilung der Daten ab. Dies trifft besonders auf die unterrepräsentierten Daten zu. Um die unterschiedlichen Auswertungen der Experimente vergleichbar zu machen, wurde diese Aufteilung einmal vor einer Serie von Experimenten durchgeführt und diese Aufteilung für das ganze Experiment beibehalten. Da die meisten Experimente sehr rechenzeitintensiv sind, wurde auf eine Kreuzvalidierung verzichtet.

Augmentierung wird lediglich beim Trainingsdatensatz durchgeführt, für die Validierung und den Test werden immer originale Daten verwendet. In Abbildung 4.10 sind die Trainingsbeispiele zu sehen, bei denen nun alle Augmentierungstechniken überlagert zum Einsatz kamen.

4.3.5 Balancing

Betrachtet man das Fahrzeugumfeld, ist es schwierig eine a priori Verteilung der Klassen zu bestimmen. Auch wenn es theoretisch möglich wäre, durch eine große Mess- und Labelkampagne die durchschnittliche Verteilung der Objektklassen zu bestimmen, ist sie für das jeweilige Fahrzeugumfeld nur wenig aussagekräftig. Da die Klassenverteilung je nach Szenario, z. B. Innenstadt oder Wohngebiet oder Tageszeit z. B. Tag oder Nacht stark variiert.

Aus diesem Grund wurde sowohl für das Training, wie auch für die Auswertung eine Gleichverteilung aller Klassen angenommen. Der Ausgleich in der Verteilung wird durch ein zufälliges Vervielfältigen der unterrepräsentierten Klassen erreicht. Diese werden im Trainingsdatensatz jedoch unterschiedlich augmentiert.

Eine Interpolation zwischen einzelnen Trainingsbeispielen wie in der Literatur, z. B. [BPM04], vorgeschlagen wird, ist bei Karten nur schwer möglich, da diese erst aufeinander ausgerichtet werden müssten.

4.4 Zellen-Datensatz

Im Gegensatz zum Cluster basierenden Datensatz wie in Abschnitt 4.3 beschrieben, werden bei diesem Datensatz keine möglichen Objekte extrahiert. Stattdessen wird jeder Zelle der Labelkarte ein Label zugewiesen. Dabei hat die Labelkarte die gleichen Dimensionen wie die Belegungskarte. Die Anzahl der Schichten wird jedoch von der Anzahl der Klassen bestimmt, die *one-hot* encodiert in die Karte eingetragen werden.

Die Label werden, unabhängig von der Objektinstanz, in die Karte eingetragen. Dadurch kann auf ein vorheriges Clustering der Daten verzichtet werden. Hiermit können fehlerhafte Zuordnungen der Label durch Inkorrektheiten beim Clustering vermieden werden.

Im Folgenden wird die Labelzuordnung für die zwei in der Arbeit untersuchten Ansätze vorgestellt. Beim Ersten wird eine semantische Segmentierung vorgenommen.

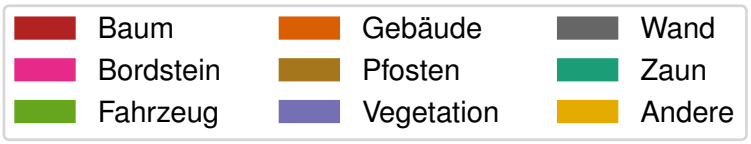
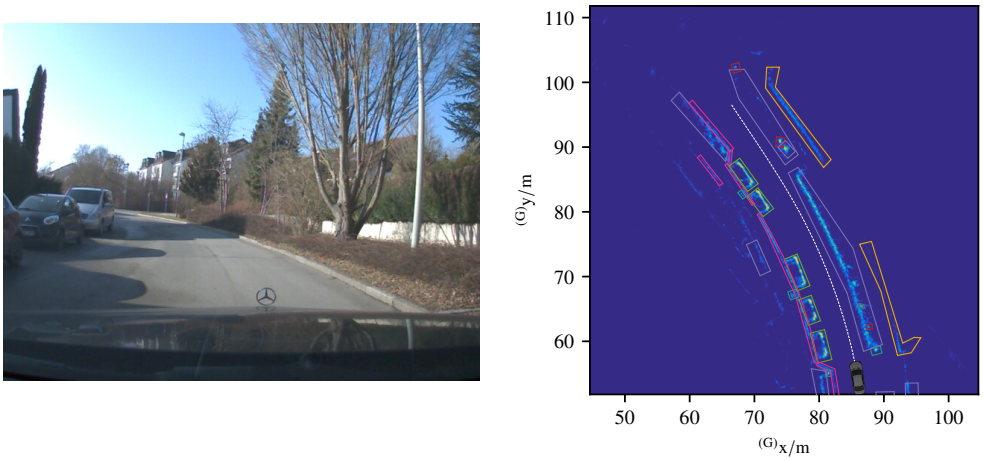


Abbildung 4.11: Legende der Label.



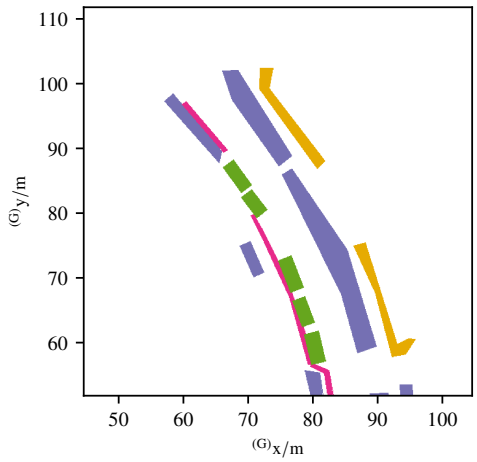
(a) Bild der Front-Kamera. (b) Belegungskarte mit Label-Polygonen.

Abbildung 4.12: Gelabeltes Szenario in einem Wohngebiet.

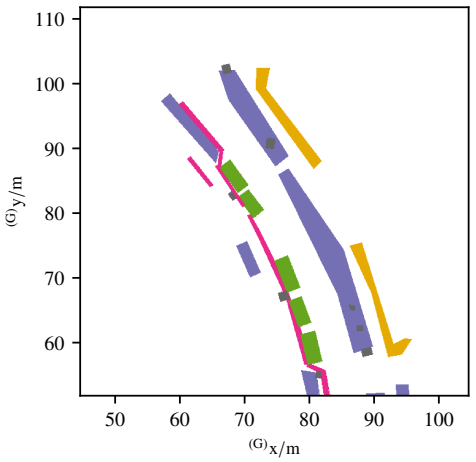
Klassifiziert werden hierbei nur belegte Zellen, d. h. Zellen bei denen die Belegungswahrscheinlichkeit über einem bestimmten Wert liegt. Hierzu muss lediglich für jede Zelle ein Label bestimmt werden.

Beim zweiten Ansatz werden die Label pro Objekt vergeben. Dabei hat der Belegungszustand einer einzelnen Zelle keine Bedeutung. Der Belegungszustand fließt nur in die Bewertung der Gültigkeit eines ganzen Objektes ein. Dieser Datensatz ermöglicht, Instanzen von Objekten zu bilden und deren Abmessungen zu schätzen.

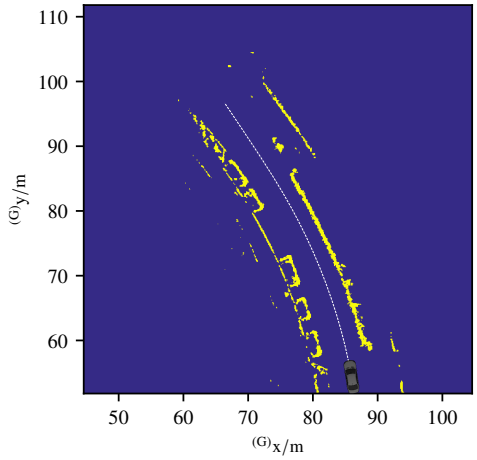
In den folgenden Abschnitten gibt es einige Abbildungen, bei denen die verschiedenen Labelklassen durch unterschiedliche Farben dargestellt werden. Um die Abbildungen kompakt zu halten, wird die Farblegende der Label nur einmal in Abbildung 4.11 dargestellt.



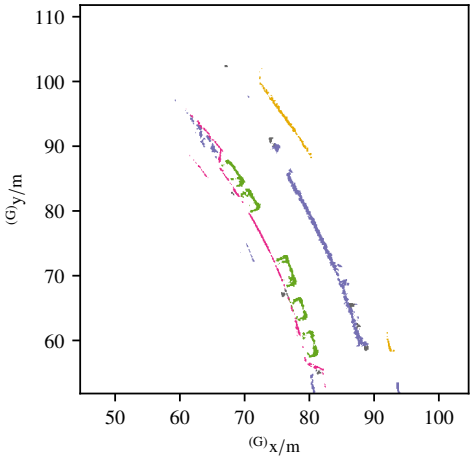
(a) Labelkarte nachdem ein Teil der Labels aus 4.12b eingetragen wurde.



(b) Labelkarte mit allen Labeln aus 4.12b.



(c) Schwellwertkarte welche die gültigen Zellen bestimmt.



(d) Labelkarte 4.13b nach Anwendung der Schwellwertkarte 4.13c.

Abbildung 4.13: Beispiel für die Erstellung von Labelkarten. Die Bedeutung der Farben der Labelkarten (4.13a, 4.13b, 4.13d) ist in der Legende in Abbildung 4.11 zu finden.

4.4.1 Zellenweise Labelzuordnung

Ziel dieser Methode ist, für jede belegte Zelle ein Label zuzuweisen. Alle Zellen, die nicht als belegt gelten, erhalten das Label *Hintergrund*. Dabei ist es irrelevant, ob es sich hierbei um einen wirklichen Freiraum handelt oder ob zu wenig Messpunkte von diesem Bereich vorhanden sind. Dies ist zum Beispiel bei Regionen der Fall, die sich noch weit entfernt vom Fahrzeug befinden oder Regionen, die durch Verdeckungen nicht beobachtet werden konnten. Bei soliden Objekten, wie Häusern, ist beispielsweise nur die Außenkante sichtbar, da die Wände den Innenraum in der Regel komplett abschirmen. Anders verhalten sich Objekte wie Büsche, die teilweise von der elektromagnetischen Strahlung durchdrungen werden.

Um die Label-Polygone in eine Labelkarte zu überführen, wird zunächst die Belegungskarte im zeitlichen Intervall T abgetastet und für jede extrahierte Karte eine Labelkarte derselben Größe wie die Radar-Belegungskarte erstellt. Dieses wird mit dem Label *Hintergrund* initialisiert.

Die Label-Polygone werden nach Fläche absteigend sortiert und nacheinander in die Labelkarte eingetragen. Das Sortieren sorgt dafür, dass Zellen die sich innerhalb von mehreren Label-Polygonen befinden, das dominierende Label zugewiesen wird. Hierbei wird das Dominanzmaß durch die inverse Fläche des Label-Polygons bestimmt. Wie bereits in Abschnitt 4.3.2 beschrieben, dominiert die kleinste Fläche, da flächige Objekte, wie z. B. Vegetation, als gesamtes gelabelt werden. Kleine Objekte wie z. B. Pfosten, werden innerhalb dieses Polygons zusätzlich gelabelt. Daher unterbrechen kleinere Objekte, das großflächigere Label.

Beim Iterieren über die Liste der Label, erhält jede Zelle die eine Überschneidung mit dem Label-Polygon aufweist das dementsprechende Label. Wenn vorher schon ein Label zugewiesen wurde, wird dieses überschrieben.

Für diese Aufgabe können optimierte Algorithmen zum Zeichnen aus der Bildverarbeitung verwendet werden. Hierdurch können im Randbereich Fehler von bis zu einem Pixel entstehen. Da die Varianz durch den Labeler deutlich höher ist, kann dies jedoch vernachlässigt werden.

Zum Schluss wird noch allen Zellen, deren Wert in der Belegungskarte kleiner als der Schwellwert τ_{valid} ist, das Label *Hintergrund* zugewiesen.

Anhand des Beispiels in Abbildung 4.12 soll das Vorgehen noch einmal erläutert werden. Links ist das Bild des Szenarios zu sehen, hierbei handelt es sich um eine gekrümmte Straße in einem Wohngebiet mit Fahrzeugen, Pflanzen und anderen Objekten. Rechts ist die Belegungskarte mit den Label-Polygonen zu sehen. In Abbildung 4.13a wird die Labelkarte gezeigt, nachdem ein Teil der Label, Polygone

mit den größeren Flächen, eingetragen wurden. Abbildung 4.13b enthält nun alle Label. Im Vergleich ist zu sehen wie die Zellen von vorherigen Labeln überschrieben wurden. Somit dominieren Label mit kleiner Fläche, über Label mit größerer Fläche. Der Pfosten der Straßenbeleuchtung dominiert beispielsweise gegenüber der Vegetation.

Bei einigen Klassen wie z. B. *Fahrzeug* und *Bordstein* ist nicht immer klar, welches der Objekte dominiert, da die Flächenverhältnisse nicht klar sind. Um dies zu verbessern, müsste der Labeler eine z-Ebene für die überlappenden Label bestimmen. In der Praxis tritt dies jedoch selten genug auf, dass auf diese zusätzliche Labelarbeit verzichtet werden kann.

Zum Schluss wurde mithilfe der Schwellwertmaske, die in Abbildung 4.13c zu sehen ist, der gültige Bereich der Label definiert, sodass nur belegte Zellen ein Label erhalten. Alle nicht belegten Zellen werden als Hintergrund deklariert. Die finale Labelkarte ist in Abbildung 4.13d zu sehen.

4.4.2 Objektweise Labelzuordnung

Um die Form, Ausrichtung und Dimension von Objekten zu schätzen, müssen die Labeldaten in anderer Form vorliegen. Eine objektweise Betrachtung ist nur für Objekte möglich, die klar voneinander abgrenzbar sind. Zusätzlich muss für die Bestimmung der Form und Ausrichtung ihre Ausdehnung abschätzbar sein.

Ein solcher Datensatz wird in dieser Arbeit nur für das Label *Fahrzeug* erstellt. Andere Klassen wie *Bordstein* oder *Vegetation* sind nur schwer abgrenzbar. Die Ausdehnung der Klasse *Pfosten* ist so klein, dass eine Instanzbildung auch durch das Labeling aus Abschnitt 4.4.1 möglich wäre.

Diese Instanzbildung könnte beispielsweise mit einer eindeutigen Identifikationsnummer oder einem Abstandsmaß der einzelnen Zellen zueinander realisiert werden. Dies müsste aber in einem separaten Schritt gelernt werden.

Da Fahrzeuge in der Belegungskarte in der Regel durch einen Abstand getrennt werden, kann dieser Schritt auch mit einer Labelkarte erreicht werden. Für die objektweise Zuweisung von Labeln sollte jede Instanz aus einer zusammenhängenden Fläche bestehen. Zudem sollten Objekte mit gleichen Labeln durch einen gewissen Abstand getrennt sein. Dies ermöglicht ein Segmentieren von Objektinstanzen.

Um dies trotz evtl. zu großzügig gewählter Label zu gewährleisten, werden hierzu die Label um einen bestimmten Faktor τ_{shrink} verkleinert. Diese Skalierung kann nach einer Prädiktion durch den Klassifikator wieder revidiert werden.

Das weitere Vorgehen ist, wie in Abschnitt 4.4.1, mit Ausnahme der Bestimmung der Gültigkeit des Labels, beschrieben. Ob ein Label valide ist, oder ob es sich um Hintergrund handelt, wird dabei nicht zellenweise bestimmt, sondern einmal pro Label. Falls ein Label-Polygon gültig ist, sind alle Zellen im Label-Polygon valide, ansonsten erhalten alle Zellen des Label-Polygon das Label *Hintergrund*.

Ein Label ist dann gültig, wenn die Fläche der validen Zellen größer als $a_{\text{min, valid}}$ ist und eine Zelle ist valide, wenn der Belegungswert größer als der Schwellwert τ_{valid} ist. So kann unabhängig von der Belegung der Zellen, die Dimensionen des Objekts bestimmt werden.

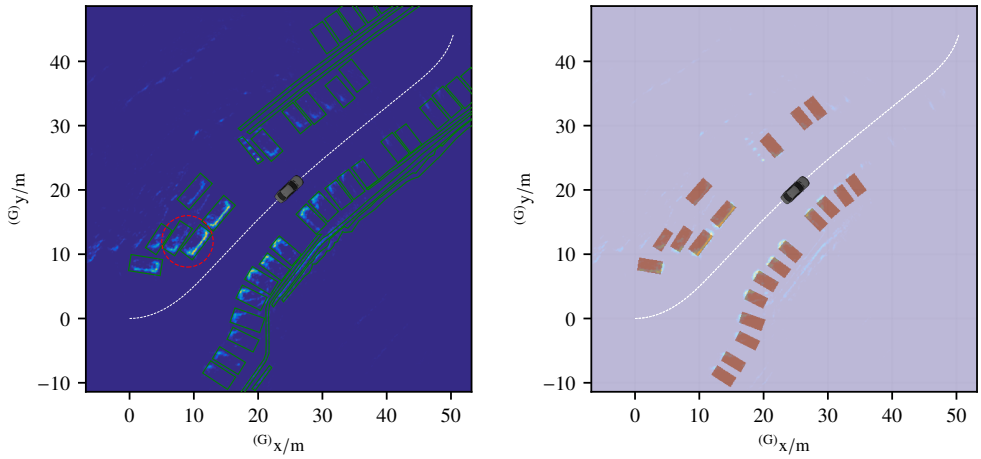
Abbildung 4.14 zeigt das Resultat dieser Vorverarbeitung. Links ist die Belegungskarte mit den vom Labeler gezeichneten Label-Polygonen zu sehen. Die rechte Abbildung zeigt die Labelkarte, die in Grün die einzelnen Autoinstanzen hervorhebt. Alle weißen Zellen erhalten das Label *Hintergrund*. Um dem Leser eine bessere Orientierung in der Karte zu geben, ist die Labelkarte zusätzlich mit der Belegungskarte hinterlegt.

Es ist ebenfalls zu sehen, dass aufgrund des Belegungszustandes, nicht alle Label-Polygone in der Labelkarte eingetragen werden. Diese werden erst zu einem späteren Zeitpunkt der Belegungskarte, bei einem ausgeprägteren Belegungszustand, in die Labelkarte eingetragen. Des Weiteren zeigt die Abbildung, dass die Label-Polygone vor dem Eintragen verkleinert worden sind. Dies ermöglicht eine klare Trennung auch bei sehr eng aneinander liegenden Objekten, wie beispielsweise im Rot hervorgehobenen Bereich in Abbildung 4.14a zu sehen.

4.4.3 Augmentation der Trainingsdaten

Augmentation findet ähnlich zu dem Cluster basierenden Datensatz statt, wie in Abschnitt 4.3.3. Hier sollen lediglich die Unterschiede erwähnt werden.

Da bei Zellen basierenden Klassifikationsmethoden, das Ergebnis in einer ähnlichen Größenordnung vorliegen soll, ergibt eine *Translation* nur im sehr geringen Umfang Sinn. Sie wird daher nicht angewendet. Anders als beim Cluster-Datensatz wird hierfür die ganze Karte rotiert, die Labels werden danach auf der Belegungskarte angewendet. Additiver Levelshift, Multiplikativer Levelshift und Rauschen werden in gleicher Weise wie beim Cluster-Datensatz eingesetzt.



(a) Belegungskarte mit Label-Polygonen von Fahrzeugen.

(b) Zellenweise Label von Fahrzeuginstanzen.

Abbildung 4.14: In grün sind die Fahrzeugzellen hervorgehoben. Zur Anschaulichkeit ist diese Darstellung noch einmal mit einer Belegungskarte hinterlegt. Der rote Kreis in (a) zeigt einen Bereich bei dem die Fahrzeuge nahe beieinander liegen, durch ein Skalieren des Labels lässt sich dieser Bereich in (b) klar trennen.

4.4.4 Aufteilung der Datensätze

Der Datensatz wird auch hier in Trainings- (70 %), Validations- (15 %) und Testdatensatz (15 %) aufgeteilt. Da bei den meisten Trainingsalgorithmen auf einer ganzen Belegungskarte trainiert wird, kann die Unterteilung des Datensatzes nicht anhand der Label geschehen.

Um zu vermeiden, dass korrelierte Daten über die verschiedenen Datensätze verteilt werden, wird anhand der Sequenzen unterteilt. Dies ergibt, dass größere Abschnitte der Daten einem Datensatz zugeordnet werden. Da die einzelnen Objekte nicht gleichmäßig über die Sequenzen verteilt sind, ergeben sich zwischen den Datensätzen geringfügige Unterschiede in der Objektklassenverteilung. Ebenso sind Varianzen z. B. aufgrund der Umgebung oder der gefahrenen Geschwindigkeit nicht so gut über die Datensätze verteilt.

4.4.5 Balancing

Eine Gleichverteilung der Klassen pro Trainingsbatch ist bei Zellen basierenden Ansatz nur schwer umzusetzen. Daher wird diesem Problem mit einer geeigneten Kos-

tenfunktion begegnet. Diese wird bei den jeweiligen Trainingsmethoden beschrieben.

4.5 Überblick und Analyse des Datensatzes

Die Datensätze wurden mithilfe der Algorithmen, wie sind in Abschnitt 4.3 und 4.4 beschrieben sind erstellt. Diese Datensätze enthalten alle in Abschnitt 2.4 beschriebenen Belegungskarten. Diese Zuordnung ist leicht, da zwischen den verschiedenen Karten eine räumliche Zuordnung besteht.

Im Folgenden soll eine Übersicht über den Umfang und die Verteilung der Label gegeben werden. Zudem soll auf Besonderheiten der Datensätze eingegangen werden.

Tabelle 4.2 gibt eine Übersicht über die gelabelten Daten. Mit etwas über 50000 gelabelten Objekten hat der Datensatz eine relevante Größe, mit dessen Hilfe Klassifikatoren trainiert und evaluiert werden können und daher Schlüsse auf die Verallgemeinerung zulassen.

Es ist zu sehen, dass die Verteilung der Klassen nicht gleichmäßig ist. Die Klasse *Fahrzeug* dominiert dabei mit ca. 12000 Objekten, dies geht darauf zurück, dass bei innerstädtischen Aufnahmen auch große Parkplätze befahren worden sind. Bei den Klassen *Wand*, *Gebäude*, *Zaun* und *Bordstein* handelt es sich um lang gezogene Objekte. Daher sind die Label-Polygone in der Regel größer. Dies erklärt die signifikant kleinere Menge der Label-Polygone.

Die Fläche der Label zeigt ein ähnliches charakteristisches Muster. Das Label *Gebäude* besteht zum Großteil aus großflächigen Labeln, während *Bordstein* meist aus schmalen langen Schläuchen besteht, dies spiegelt sich auch in der Labelfläche wider. Bei *Vegetation* ist die Labelfläche gemischt, da diese beispielsweise aus schmalen Hecken, aber auch aus großflächigem Gebüsch besteht. Zudem ist bei der Labelfläche zu beachten, dass die Labeler nur bei der Klasse Fahrzeug instruiert worden sind die Größe abzuschätzen, bei allen anderen Labeln ist lediglich eine Abgrenzung zu anderen Klassen erforderlich.

Die Label lassen sich, über den geometrischen Zusammenhang, von den Polygonen zurück auf die Radar-Messpunkte projizieren. So können mit einem gezeichneten Label-Polygon Messpunkte aus allen Zeitschritten extrahiert werden. Dies setzt natürlich voraus, dass die Objekte sich während der Vorbeifahrt nicht bewegen und dass die Pose des Fahrzeugs in der Karte genau genug zu jedem Messzeitpunkt zur Verfügung steht.

Tabelle 4.2: Übersicht über die gelabelten Daten. Anzahl: Zahl der Label-Polygone, Fläche: Gesamtfläche der Label-Polygone und Messpunkte: Anzahl der Radar-Messpunkte die in ein Label fallen.

Label	Anzahl Label	Fläche m^2	∅ Fläche m^2	Anzahl Messpunkte	∅ Messpunkte
Baum	4612	32480	7.0	909701	197
Bordstein	4187	48852	11.7	1726987	412
Fahrzeug	11531	114149	9.9	7877754	683
Gebäude	3651	584115	160.0	6086255	1667
Pfosten	10590	11751	1.1	1292099	122
Vegetation	7423	553675	74.6	11168539	1504
Wand	1626	19657	12.1	1483453	912
Zaun	2498	39116	15.7	4120078	1649
Andere	8653	101763	11.8	4237692	489
Gesamt	54771	1505558	27.5	38902558	710

Eine solche Rückprojektion ermöglicht eine genauere Analyse auf Basis der Punktwolke. In Tabelle 4.2 ist zu sehen, dass mit dieser Methode etwa 40 Millionen Radar-Messpunkte gelabelt worden sind. Die Verteilung der Anzahl der Messpunkte verhält sich hierbei anders, als die der Fläche. So hat beispielsweise die Klasse *Zaun* und die Klasse *Gebäude* eine ähnliche Anzahl an Messpunkten pro Objekt. Auch wenn die durchschnittliche Labelfläche eines Gebäudes die des Zauns um das 10-fache übersteigt. Dies lässt sich zum einen mit den großflächigeren Labeln bei Gebäuden erklären, zum anderen stehen Zäune öfter nah am Straßenrand als Gebäude und werden dadurch öfter beobachtet. Des Weitern bestehen Zäune oft aus gut reflektierenden Einzelelementen (z. B. Pfosten, Träger, Latten), welche über einen weiten Winkelbereich zu sehen sind, während bei Gebäudeteilen, die aus einer glatten Wand bestehen, das Signal in einem weiten Winkelbereich weg reflektiert wird.

Für die Erstellung der verschiedenen Datensätze wurde immer auf die hier beschriebenen Label zurückgegriffen. Im nächsten Abschnitt werden die Radardaten mithilfe der Label statistisch betrachtet. In den nachfolgenden Abschnitten werden die ein-

zelnen Datensätze vorgestellt.

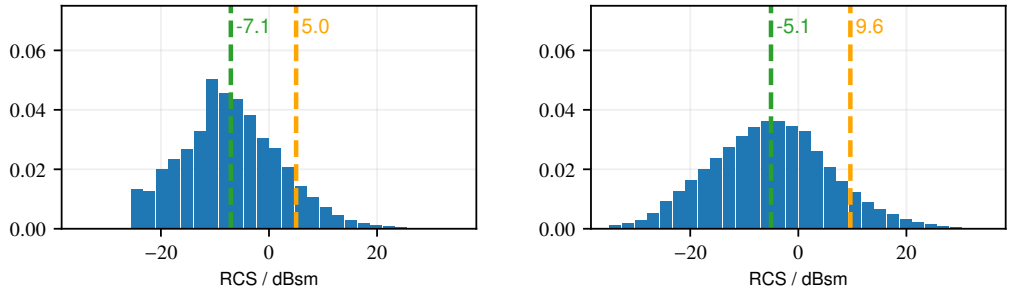
4.5.1 Statistische Betrachtung eines Radarsensors

Zunächst werden zwei verschiedene Sensoren verglichen, der in der Arbeit verwendete Nahbereichsradar (NBR) und ein experimenteller Fernbereichsradar (FBR). Beide Sensoren stellen die Daten auf der Ebene der Reflexionszentren zur Verfügung. Die Datengrundlage für den NBR ist der gesamte vorgestellte Datensatz und weitere nicht gelabelte Sequenzen, mit ungefähr 100 Mio. Messpunkten. Bei dem FBR stehen ungefähr 150 Mio. Messpunkte zur Verfügung, welches einer Aufnahmezeit von ca. 2,5 h entspricht.

In Abbildung 4.15 wird ein Histogramm des RCS-Wertes über alle Messungen dargestellt. In 4.15a ist die Verteilung des NBR zu sehen, in Abbildung 4.15b die des FBR. Die Bestimmung des RCS-Wertes ist komplex, und auch von vielen Umgebungseinflüssen abhängig, trotzdem lassen sich in der Statistik einige Rückschlüsse ziehen.

Beim FBR nimmt die Messverteilung vom Maximum aus, nahezu gleich zu den hohen und den niedrigen RCS-Werten ab. Beim NBR hingegen ist der Abfall zu den kleineren RCS-Werten deutlich stärker, als zu den größeren. Dieser Effekt lässt sich mit den Schnittstellen der Sensoren erklären. Beim FBR werden pro Messzyklus mehrere hundert Messpunkte übertragen, beim NBR wird diese Liste aufgrund der Übertragungsschnittstelle bei deutlich unter 100 abgeschnitten. Da wichtige Ziele statistisch gesehen oft Rückstreuquerschnitte mit hohen RCS-Werten besitzen, und Clutter hingegen eher niedrige, werden vermehrt Messungen mit niedrigen RCS-Werten nicht übertragen.

Trotzdem ist der Mittelwert des Rückstreuquerschnitts größer. Dies lässt sich dadurch erklären, dass neben dem Nahbereich auch Objekte in größerer Entfernung detektiert werden. Hierbei fällt ein größerer Bereich des Objektes in eine Winkelzelle, während Objekte im Nahbereich in mehrere Teilobjekte zerfallen. Damit verteilt sich die rückgestreute Energie auf mehrere Messpunkte. Auch der Bereich des Objektes der in Elevation vom Sensor angestrahlt wird, nimmt mit zunehmender Entfernung zu. Diese Objektteile tragen auch zu einem größeren RCS-Wert bei. Dies lässt sich auch in Abbildung 4.16 sehen, hier werden die RCS-Werte abhängig von der Entfernung aufgetragen. Im Bereich von 0 m bis 10 m nimmt der RCS stark zu. Dies trifft sowohl für den Nah- wie auch für den Fernbereichsradar zu. Der durchschnittliche RCS-Wert steigt anschließend linear geringfügig an. Der Ausreißer im Bereich von 0 m bis 3 m des FBR ist wahrscheinlich durch eine fehlerhafte Kalibrierung verursacht.



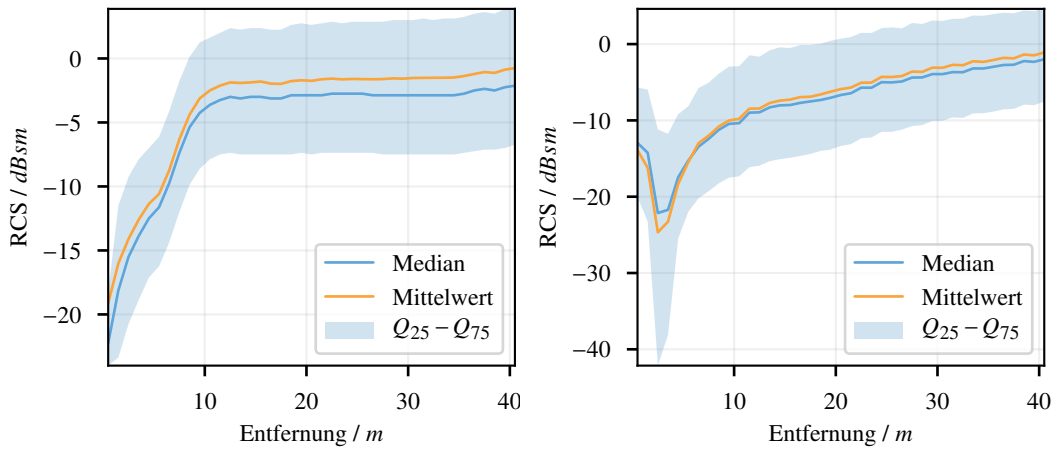
(a) Nahbereichsradar

(b) Fernbereichsradar

Abbildung 4.15: Histogramm der RCS-Werte aller Messungen. Dabei wird zwischen dem verwendeten Nahbereichsradar und einem experimentalen Fernbereichsradar verglichen. In grün wird der Mittelwert des Logarithmischen RCS-Wertes dargestellt. In orange wurde der Mittelwert auf den linearen Werten gebildet.

Es zeigt sich, bei der Betrachtung desselben Entfernungsbereichs, dass der durchschnittliche RCS-Wert des NBR deutlich kleiner ist, als der des FBR. Dies lässt sich durch die höhere Auflösung des FBRs erklären, da Objekte in mehr Reflexionszentren als beim NBR zerfallen, welches zu einem niedrigeren RCS-Wert pro Reflexionszentrum führt.

In Abb. 4.17 ist zu sehen, wie die Anzahl der Messungen zunächst stark ansteigt, bei ca. 5 m ihr Maximum erreicht und danach wieder abfällt. Dies hat mit der realen Verteilung von Objekten zu tun. So befinden sich während der Fahrt nur wenige Objekte im Bereich von 1 m um das Fahrzeug, was den Anstieg der Messungen im Nahbereich erklärt. Des Weiteren werden die Anzahl der Messungen durch die Winkelzellen beeinflusst, die mit zunehmendem Abstand immer größer werden. Daher nimmt die räumliche Auflösung von Reflexionszentren immer weiter ab. Außerdem fallen Objekte mit niedrigem Rückstreuquerschnitt unter das vom Sensor detektierbare Energielevel. Hierbei gilt zu beachten, dass der RCS-Wert entfernungskompensiert ist, das Energielevel aber wie in Gleichung 2.1 auf Seite 7 zu sehen, mit $1/R^4$ abnimmt.



(a) Nahbereichsradar (b) Fernbereichsradar

Abbildung 4.16: Verteilung der RCS-Werte über die Entfernung.

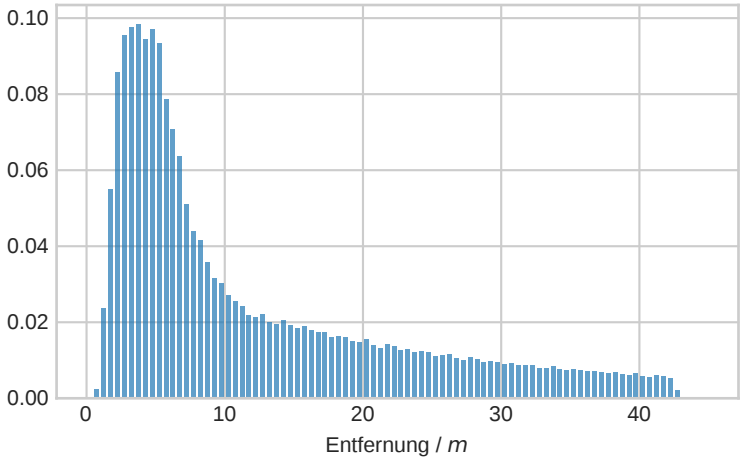
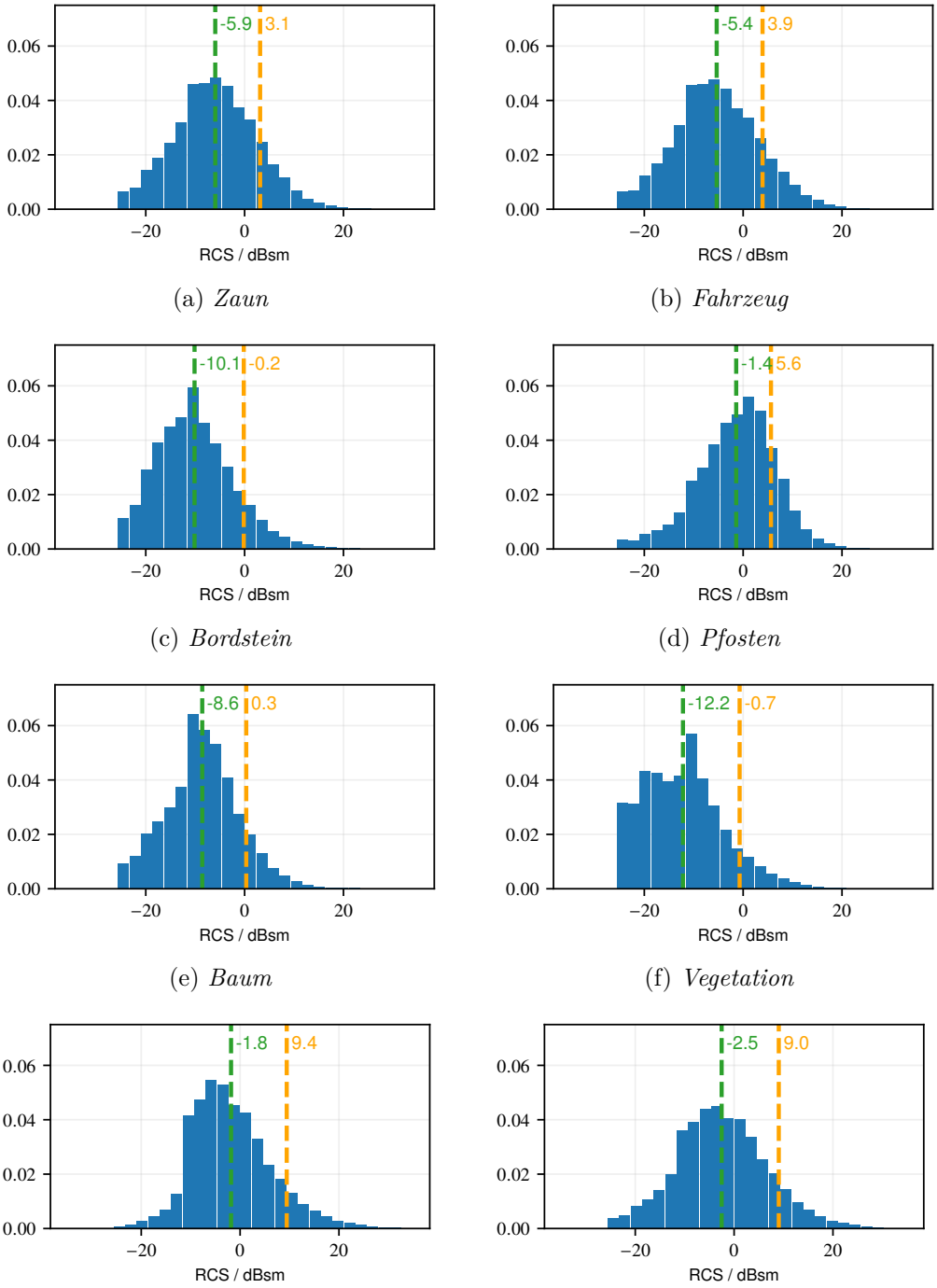


Abbildung 4.17: Verteilung der Anzahl der Messungen über die Entfernung.

4.5.2 Statistische Betrachtung der Klassen

Im Nachfolgenden soll auf die Eigenschaften der einzelnen Objekttypen im Radar eingegangen werden. Dies geschieht mithilfe des gelabelten Datensatzes und wird nur anhand des NBR durchgeführt, da für den FBR keine gelabelten Daten zur Verfügung stehen.



(a) *Zaun* (b) *Fahrzeug*

(c) *Bordstein* (d) *Pfosten*

(e) *Baum* (f) *Vegetation*

(g) *Gebäude* (h) *Andere*

Abbildung 4.18: Histogramm der klassenspezifischen RCS-Werte bezüglich der gelabelten Klassen. In grün ist der Mittelwert auf den logarithmischen Werten, in orange auf den linearen Werten dargestellt.

In Abbildung 4.18 ist die Verteilung des RCS-Wertes über die verschiedenen Klassen dargestellt. Zu beachten ist hierbei, dass die Objekte mit Polygonen gelabelt sind. Daher kann es vorkommen, dass im Randbereich auch Messungen enthalten sind, die nicht zum Objekt gehören. Wenn man zum Beispiel einen Pfosten betrachtet, der eine geringe Grundfläche besitzt, können durch das Labelpolygon auch einzelne Messungen von einem benachbarten Bordstein oder eines pflanzlichen Bewuchses enthalten sein. Umgekehrt sind bei zu klein gewählten Polygonen einige Messungen des Objektes nicht berücksichtigt. Durch die große Anzahl der Label ist jedoch davon auszugehen, dass sich diese Effekte nur im geringeren Umfang auf die Statistik auswirken.

Die Betrachtungsweise der RCS-Statistik soll uns die Bedeutung dieses Merkmals näherbringen. Allerdings kann darüber nur ein Teilaspekt beleuchtet werden, da hierbei nicht auf die örtliche Verteilung des RCS-Wertes innerhalb eines Objektes eingegangen werden kann, beispielsweise dass bei einem Zaun die Pfosten einen höheren RCS-Wert haben als die Flächen.

Der Mittelwert der Verteilungen wird in Grün und in Orange dargestellt. Hierbei handelt es sich zum einen, in Grün dargestellt, um den Mittelwert, der auf den logarithmischen Werten berechnet wurde. Zum anderen, in orange dargestellt, um den wahren Mittelwert der auf den linearen Werten berechnet und ins Logarithmische transformiert worden ist.

Die Klassen lassen sich hierbei schon anhand ihrer Mittelwerte gruppieren. So befinden sich Bordstein, Bäume und Vegetation in einem ähnlichen Bereich, wobei die Klasse *Vegetation* viele Messpunkte in den niedrigen RCS Bereichen besitzen. Bei den anderen Klassen nehmen diese stark ab.

Anhand der RCS-Verteilung ähneln sich Fahrzeuge und Zäune stark. Auch der Pfosten liegt mit dem Mittelwert in einer ähnlichen Größenordnung, jedoch liegt das Maximum der Messwertverteilung bei ca. 0 dBsm. Diese führt dann zu höheren Mittelwert der logarithmischen Werte. Zudem ist ein steiler Abfall zu den großen RCS-Werten vorhanden.

Die Klassen mit den höchsten durchschnittlichen RCS-Werten sind die Klassen *Andere* und *Gebäude*. Hierbei handelt es sich oftmals um größer Strukturen. Die Streuung ist dabei sehr groß.

4.5.3 Cluster-Datensatz

Für die Erstellung dieses Datensatzes wird im Abstand von einer Sekunde aus der Karte ein Auszug erstellt. Von diesen Auszügen werden, wie in Abschnitt 4.3 be-

schrieben, Cluster gebildet und diesen ein Label zugeordnet. Die Parameter für das Vorgehen sind in Tabelle 4.3 zu finden.

Tabelle 4.3: Parameter zum Erstellen des Cluster-Datensatzes

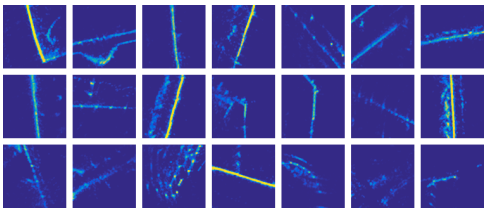
Parameter	Wert
$a_{\min}$	$0{,}3\,\mathrm{m}^2$
$s_{\max}$	$5\,\mathrm{m}$
S	0.7
τ_{const}	0.07
τ_{IoC}	0.9
T	$1\,\mathrm{s}$

In Abbildung 4.19 sind Beispiele der verschiedenen Klassen zu finden, diese wurden zufällig aus dem Datensatz ausgewählt. Aus Gründen der Kompaktheit wird lediglich die Belegungskarte, welche für den Menschen am besten verständlich ist, dargestellt. Andere radarspezifische Merkmalskarten werden im gleichen Ausschnitt extrahiert. Der extrahierte Bereich ist immer so platziert, dass er sich in der Mitte des Ausschnittes befindet. Da Umgebungsinformationen helfen können, das Objekt zu klassifizieren, werden diese in dem Kartenausschnitt **nicht ausgeblendet**.

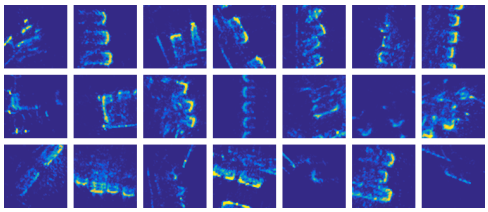
Zusätzlich zu den Karten wird eine Maske mit der extrahierten Fläche des Clusters gespeichert, um bei Bedarf den Hintergrund auszublenden zu können.

Durch das Extrahieren von Clustern in mehreren Zeitschritten, können aus einem Label, mehrere Trainingsbeispiele gewonnen werden. Abbildung 4.20 zeigt dies an zwei Beispielen. Der zeitliche Verlauf ist darin horizontal abgebildet, die Beispiele sind also zu unterschiedlichen Integrationszeiträumen der Karte extrahiert worden. Die vertikale Achse zeigt die Trainingsbeispiele, die in einem Zeitschritt extrahiert wurden, da sich innerhalb des Labels mehrere Cluster gebildet haben. Hierbei ist auf die Verschiebung des Mittelpunkts der Karte zu achten. Da diese Daten korreliert sind, werden sie immer nur in **einem** der Datensätze (Training, Validation oder Test) verwendet. Verzichtet man darauf, würden die Ergebnisse fälschlicherweise zu gut ausfallen.

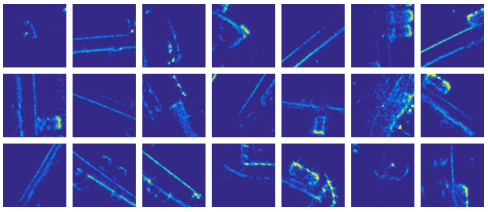
In Tabelle ist die Anzahl der extrahierten gelabelten Cluster zu finden. Es zeigt sich, dass die Anzahl der Cluster, die Anzahl der Label (siehe Tabelle 4.2) um ein Vielfaches überschreitet. Dies ist zum einen der Aufteilung eines Objektes in mehrere



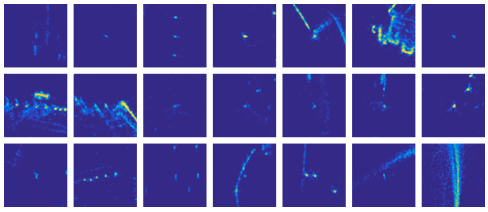
(a) *Zaun*



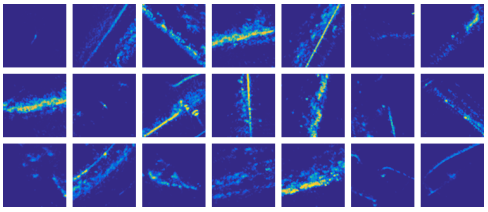
(b) *Fahrzeug*



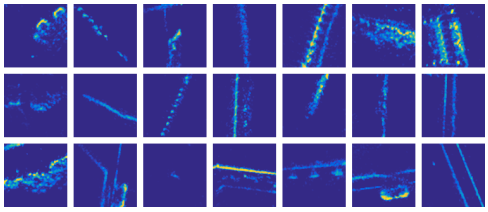
(c) *Bordstein*



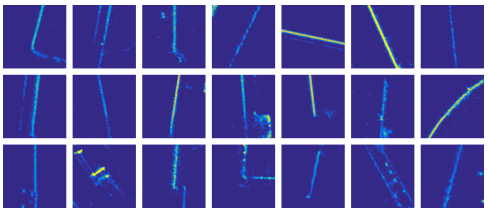
(d) *Pfosten*



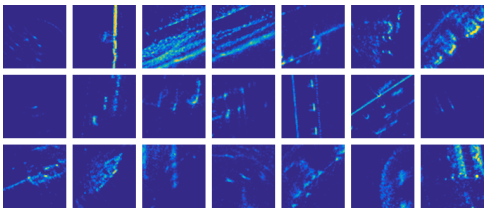
(e) *Baum*



(f) *Vegetation*

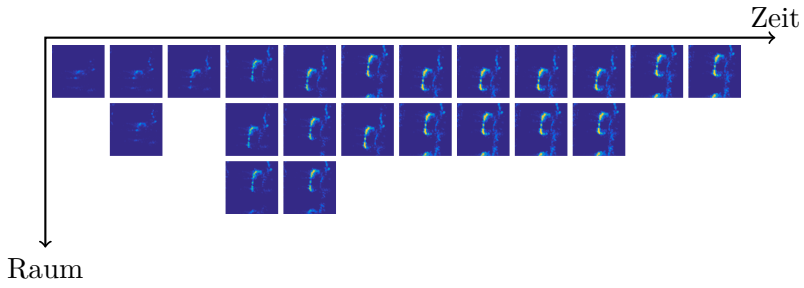


(g) *Gebäude*

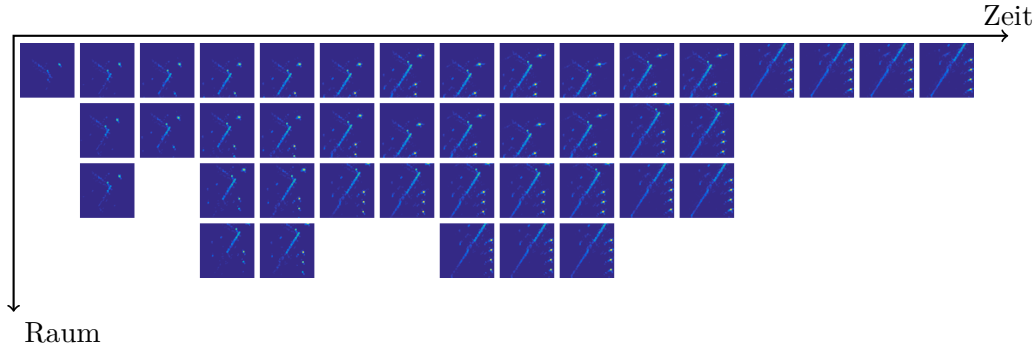


(h) *Andere*

Abbildung 4.19: Beispiele aus dem Cluster-Datensatz.



(a) *Fahrzeug*



(b) *Gebäude*

Abbildung 4.20: Cluster von je einem Label-Polygon über Raum und Zeit. Vertikal sind Cluster aus einem Zeitpunkt der Karte dargestellt. Die Cluster stammen aus unterschiedlichen räumlichen Regionen. Der zeitliche Verlauf ist horizontal dargestellt.

Cluster geschuldet. Es zeigt sich, dass aus einem ausgedehnten Objekt, z. B. der Klasse *Vegetation* deutlich mehr Cluster entstehen, als bei punktförmigen Zielen wie z. B. *Pfosten*. Zum anderen werden über den zeitlichen Verlauf mehrere Cluster vom selben gelabelten Objekt erstellt. Es handelt sich also um dasselbe Gebiet, aber mit unterschiedlicher Ausprägung der Merkmalskarten. Dies könnte noch erhöht werden, indem die Belegungskarten zeitlich häufiger abgetastet werden. Hierdurch würde die Datenmenge erheblich zunehmen, während gleichzeitig die Varianz der Daten und damit ihr Informationsgehalt immer geringer wird.

Die Varianz der Daten wird durch Augmentierung noch weiter erhöht. Um die Datenmenge nicht unnötig zu vergrößern, werden diese Methoden online, zur Trainingszeit durchgeführt. Dasselbe gilt für das Balancieren der Daten, indem einzelne mit unterschiedlichen Augmentierungen wiederholt werden [BPM04].

Tabelle 4.4: Anzahl an gelabelten Clustern

Label	Anzahl der Cluster
Wand	35849
Zaun	95694
Gebäude	82273
Bordstein	59518
Baum	28434
Vegetation	284345
Andere	137975
Pfosten	45921
Fahrzeug	271371
Gesamt	1041380

4.5.4 Zellen Datensatz

Tabelle 4.5: Parameter zur Erzeugung des Zellen Datensatz

Parameter	Wert
Schwellwert (τ_{valid})	0.55
Abtastintervall (T)	1 s
Größe des Kartenausschnitt	40 m× 40 m

Für den Zellen Datensatz wurden, ähnlich wie beim Cluster-Datensatz, Auszüge im Abstand von einer Sekunde von den Karten erstellt. Diese Auszüge werden dann, wie in Abschnitt 4.4 beschrieben, Label-Karten zugeordnet. Anders als beim Cluster-Datensatz ist diese Zuordnung so effizient, dass sie online während des Trainings erfolgen kann. Ein großer Vorteil der online Verarbeitung ist es, neben der Einsparung von Speicherplatz, die Möglichkeit Parameter zu variieren zu können, ohne zeitintensiv einen neuen Datensatz erstellen zu müssen.

Soweit nicht anders beschrieben, werden die Parameter in Tabelle 4.5 für diesen Datensatz verwendet.

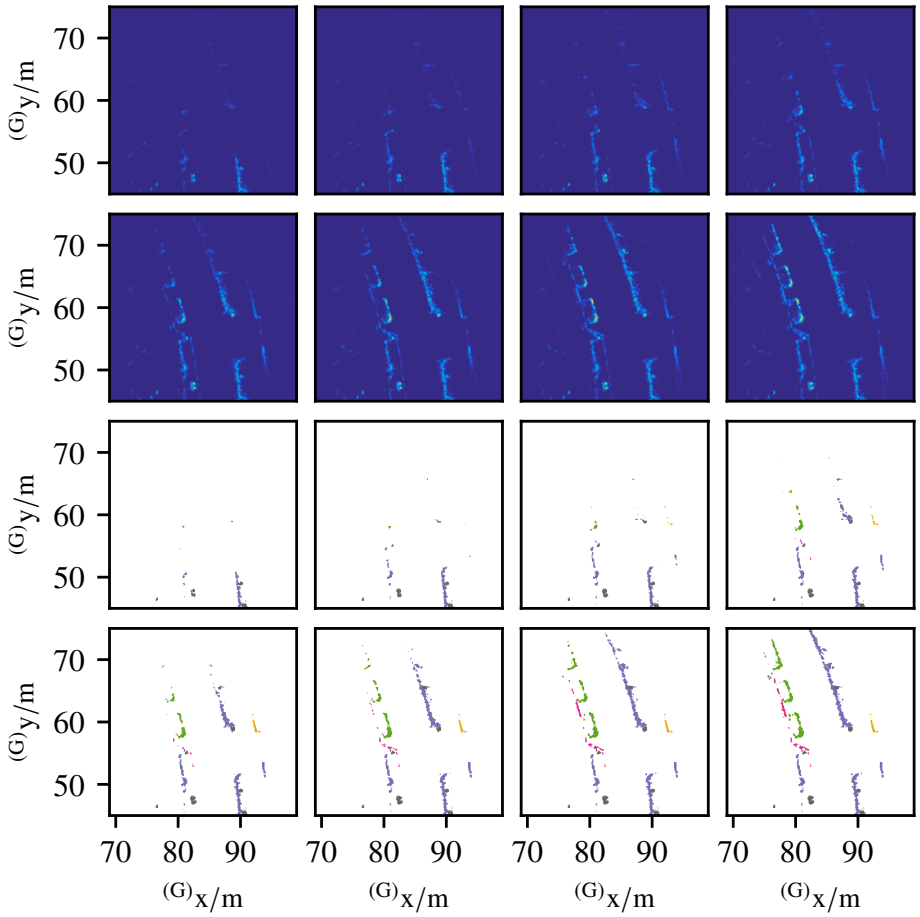


Abbildung 4.21: In der oberen Hälfte der Abbildung ist die zeitliche Entwicklung der Belegungskarte zu sehen. In der unteren Hälfte das jeweils dazu korrespondierende Labelbild.

Auch hier werden durch den zeitlichen Aufbau der Belegungskarte die Label-Polygone mehrfach verwendet. Welche Zellen gelabelt werden und welche das Label Hintergrund erhalten, hängt von der Belegungswahrscheinlichkeit der Belegungskarte ab. Da diese sich über die Zeit verändert, ändert sich auch die Anzahl der gelabelten Zellen mit ihr. Dieser Umstand muss beim Training des Klassifikators beachtet werden.

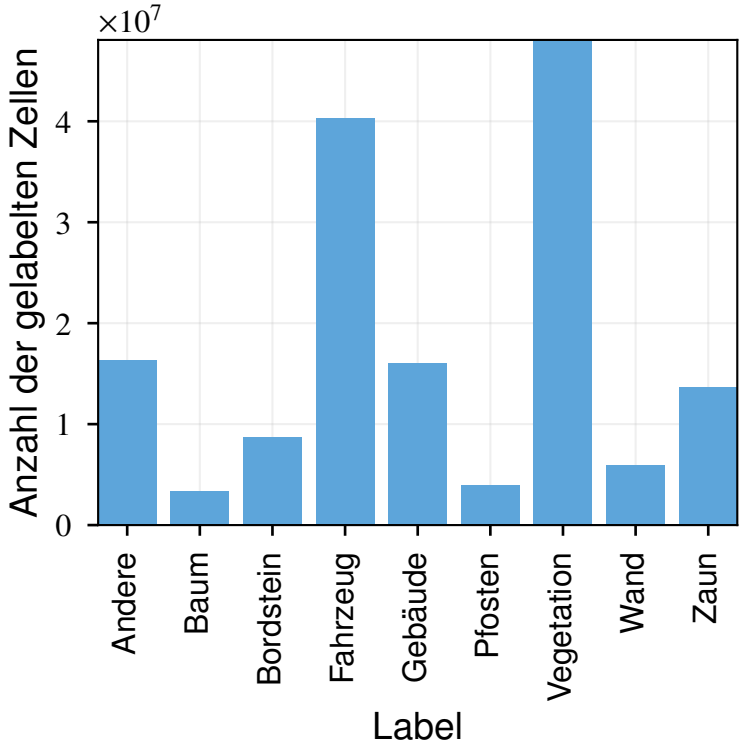


Abbildung 4.22: Anzahl der gelabelten Zellen über den zeitlichen Verlauf. Diese Werte sind ohne Augmentierung mit einem 40 m×40 m Ausschnitt um das Fahrzeug und einer zeitlichen Abtastung von 1 Sekunde erstellt.

Abbildung 4.21 zeigt diesen Effekt, indem es einen Auszug des Datensatzes über die Zeit darstellt. Der Klassifikator erhält in der Regel Auszüge aus Belegungskarten in der Größe von 40 m × 40 m. Um die Beispiele anschaulicher zu machen, wurden hierfür kleinere Ausschnitte 30 m × 30 m gewählt.

Die Strukturen eines Objektes sind über die Zeit immer besser zu sehen, da die Belegungswahrscheinlichkeit der belegten Zellen zunimmt. Je nach Belegungszustand wird das Label der Zellen "gültig", das heißt, das Label ändert sich von *Hintergrund* zu dem Label der jeweiligen Klasse. Dies geschieht jedoch nicht für jede Zelle in der gleichen Geschwindigkeit. So werden gute Radarziele, die öfter Messpunkte verursachen, früher valide, als schwächere Ziele die evtl. auch verdeckt sind. Somit ist z. B. ein Teil eines Fahrzeuges schon in der ersten Karte gültig, während der nahe liegende Bordstein erst einige Sekunden später valide wird. Zudem nimmt die Ausdehnung des validen Bereichs eines Labels mit der Zeit zu.

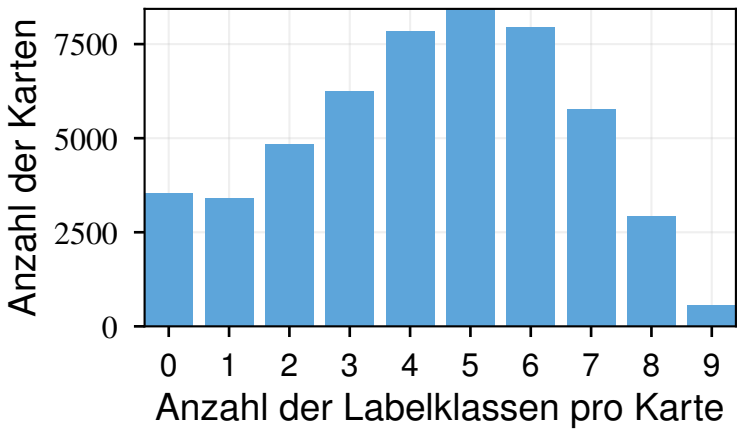


Abbildung 4.23: Histogramm über die Anzahl der Vertretenen Klassen in einem Kartenausschnitt von 40 m×40 m um das Fahrzeug.

Da in einer Belegungskarte mehr als eine Klasse vorkommen kann, ist es nicht möglich, einen Datensatz zu erzeugen, bei dem die Klassen ausgewogen sind. In Abbildung 4.22 ist ein Histogramm der gelabelten Zellen zu sehen. Diese korreliert in einigen Fällen mit der Verteilung der Labelfläche aus Tabelle 4.2, z. B. das Verhältnis *Wand* zu *Zaun*. Andere weichen deutlich davon ab, so hat die Klasse *Gebäude* die größte Labelfläche, ist aber bei der Anzahl der gelabelten Zellen nur auf Platz vier. Hier ist der Effekt des Schwellwerts am größten, da das Gebäude meist als Ganzes gelabelt wird, die Zellen hinter der Mauer in der Regel frei bleiben. Bei der Klasse *Fahrzeug* sorgen Reflexionen vom Unterboden, von den Rädern etc. für ein ausgedehnteres Ziel.

Die Unausgewogenheit der Klassen muss beim Training eines Klassifikators berücksichtigt werden.

Abbildung 4.23 gibt einen Anhaltspunkt zur Klassenvielfalt innerhalb eines Kartenausschnittes. Dieses Histogramm zeigt wie viele unterschiedlich Klassen in einem Kartenausschnitt liegen. Hierbei wird die Anzahl der gültigen Zellen nicht weiter betrachtet. Sobald eine Zelle mit einer Klasse gelabelt ist, gilt diese als vorhanden. Für diese Auswertung wurden der mittlere Ausschnitt 40 m × 40 m um das Fahrzeug analysiert. Hierfür wurden die Karten in ihrer ursprünglichen Ausrichtung, also ohne Drehung verwendet.

Zu sehen ist, dass nur sehr wenig Kartenausschnitte alle 9 Klassen enthalten. Zusätzlich fällt auf, dass es über 3000 Kartenausschnitte gibt, in denen keine Label vergeben wurden. Diese Karten stammen meist vom Anfang einer Sequenz, bei der

sich die Belegungskarte noch aufbauen muss. Zusätzlich gibt es Bereiche, die nicht gelabelt wurden, entweder weil keine Objekte vorhanden waren, Objekte vom Labeler nicht erkannt worden sind oder die Belegungskarte zu spärlich aufgebaut wurde. Mit Ausnahme von null Labeln pro Klasse ähnelt die Verteilung einer Binomial-Verteilung, was darauf schließen lässt, dass es keine starke Korrelation bezüglich des gemeinsamen Vorkommens mehrerer Klassen gibt.

4.5.5 Zellen Datensatz mit objektweiser Labelzuordnung

Tabelle 4.6: Parameter zur Erzeugung des Zellen Datensatzes

Parameter	Wert
τ_{valid}	0,55
T	1 s
$a_{\text{min, valid}}$	0,3 m
τ_{shrink}	0,75

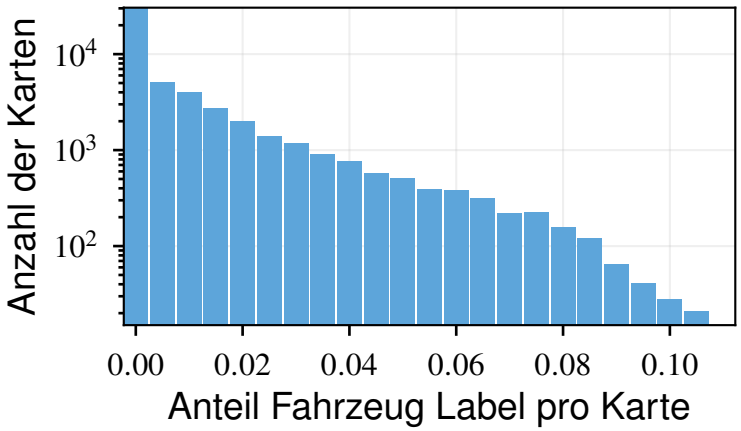


Abbildung 4.24: Histogramm über den Anteil in einem Kartenausschnitt von 40 m×40 m um das Fahrzeug.

Dieser Datensatz berücksichtigt lediglich die Klasse *Fahrzeug*. Die Parameter für diesen Datensatz sind in Tabelle 4.6 zu finden. Sobald das Label eines Fahrzeuges

gültig ist, werden alle Zellen dieser Instanz, unabhängig von ihrem Belegungszustand, gültig.

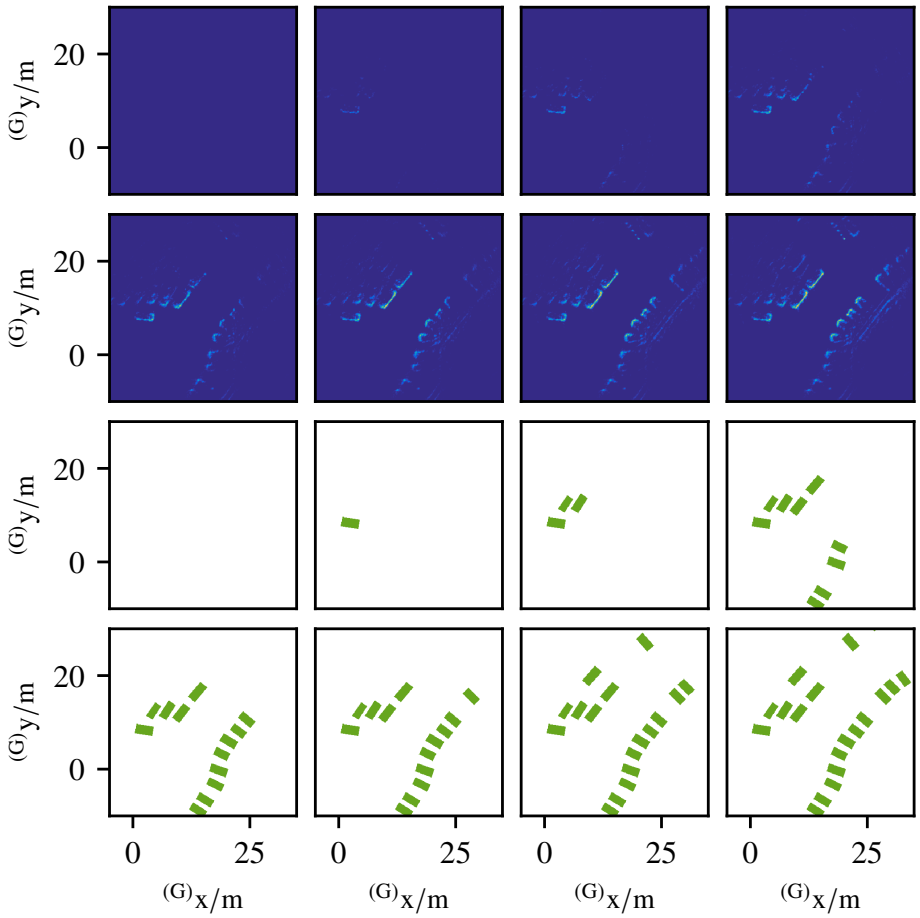


Abbildung 4.25: In der oberen Hälfte der Abbildung ist die zeitliche Entwicklung der Belegungskarte zu sehen. In der unteren Hälfte das jeweils dazu korrespondierende Labelbild.

Abbildung 4.24 zeigt, ähnlich wie im vorherigen Abschnitt, die Labelzuordnung über den zeitlichen Verlauf der Belegungskarte. Auch hier ist es nicht möglich, einen Datensatz mit einer ausgewogenen Anzahl von *Hintergrund* zu *Fahrzeug* Zellen zu erzeugen.

Die Schwierigkeit an diesem Datensatz besteht darin, dass in einigen Karten, das Label noch nicht gültig ist, obwohl als Mensch die charakteristische Form des Fahr-

zeugs schon zu erkennen ist, während in anderen Bereichen, ganz besonders bei Verdeckungen das Label gültig ist, sich aber noch nicht die charakteristische Form herausgebildet hat.

Abbildung 4.25 zeigt mithilfe eines Histogramms die Spärlichkeit der Daten. Es zeigt den relativen Anteil der *Fahrzeug*-Zellen innerhalb einer Karte. Die Anzahl der Karten verhält sich umgekehrt proportional mit der Anzahl *Fahrzeug*-Zellen. Am häufigsten sind die Karten, die nur aus *Hintergrund* bestehen und nur wenige Trainingsbeispiele haben. Sehr selten sind hingegen Karten mit mehr als 10 % Fahrzeug Label. Um dieses Ungleichgewicht zu reduzieren, werden Karten ohne Fahrzeug Label vom Trainingsdatensatz ausgeschlossen.

4.6 Zusammenfassung

In diesem Kapitel wurde ein Überblick über die aufgenommenen Daten gegeben. Es wurden die Versuchsträger und das Sensorsetup mit 4 Radarsensoren vorgestellt. Zudem wurde eine Übersicht über die Aufnahmen gegeben. Diese wurden hauptsächlich im Geschwindigkeitsbereich unter $50\,\mathrm{km\,h^{-1}}$ durchgeführt und stammen großteils aus dem urbanen Bereich.

Danach wurde vorgestellt, wie durch Clustering ein Datensatz von Objektvorschlägen erstellt wurde. Es wurde die Erstellung der Zellen basierenden Datensätze vorgestellt, die zum einen nur einer belegten Zelle ein Label zuweisen, zum anderen die Ausdehnung von geparkten Fahrzeugen erfasst.

Aufgrund der Aufnahmen und der Label, wurde in diesem Kapitel der Charakter von Radarmessungen statistisch ausgewertet und dabei Unterschiede der verschiedenen Objektklassen auf der Basis von Messungen dargestellt.

Des Weiteren wurde ein qualitativer und quantitativer Einblick in die Datensätze gegeben.

Kapitel 5

Klassifikation von Radarclustern

Um Radarcluster zu klassifizieren, kann auf Methoden zurückgegriffen werden, die aus der Bildverarbeitung stammen. Dies ist möglich, da Bilder wie auch Belegungskarten in den entscheidenden Punkten ähnliche Eigenschaften besitzen. Hierzu zählt z. B., dass ein Zusammenhang zwischen benachbarten Regionen besteht und die Informationen ähnlich zu Farb- oder Tiefenbildern in mehrere Kanäle aufgeteilt werden können.

Trotz aller Ähnlichkeiten gibt es auch entscheidende Unterschiede, so stellen 2D Belegungskarten die Welt in der Vogelperspektive dar. Anders als bei Luftbildern wird die Belegungskarte aus Beobachtungen in der Bildebene gebildet. Dies führt dazu, dass sich z. B. Verdeckungen anders in Belegungskarten charakterisieren als bei Bildern. Außerdem bestehen Belegungskarten aus zeitlich integrierten Messdaten, dadurch entstehen, bei verschiedenen Integrationszeiträumen, unterschiedliche Ausprägungen desselben Objekts.

Im Gegensatz zu Bildern sind Belegungskarten spärlich belegt, sodass in der Regel die meiste Fläche unbelegt ist. Die Karten haben außerdem eine feste räumliche Skalierung, daher gibt es keine perspektivische Verzerrung. Damit stellt die Größe eines Objektes ein charakteristisches Merkmal dar.

In diesem Kapitel werden verschiedene Ansätze zur Klassifikation von Radar-Merkmalskarten untersucht. Zuerst erfolgt eine Einführung in verschiedene Ansätze zur Klassifikation, danach wird ein Überblick über deren Einsatz im Bereich Radar und anderen artverwandten Gebieten gegeben.

In den nachfolgenden Abschnitten soll auf mehrere Methoden zur Klassifikation von Radarclustern eingegangen werden. Für alle Methoden wird hierzu der in Abschnitt 4.3 beschriebene Datensatz verwendet.

Zuerst wird auf die Klassifikation mit "klassischen" Klassifikationsverfahren mit vom Experten erstellten Merkmalsextraktoren eingegangen. Danach werden tiefe Lernmethoden, im Speziellen Faltungsnetze (*engl. Convolutional Neural Networks*) (CNNs), betrachtet.

Für die tiefen Lernmethoden werden verschiedene Aspekte untersucht. Zum einen wird ergründet, ob sich auf Bildern vortrainierte Netzwerke für die Klassifikation von Radarclustern eignen. Zum anderen wird auf die Eignung der in Abschnitt 2.4 beschriebenen Radar-Merkmalskarten eingegangen. Des Weiteren wird der Einfluss der Augmentierung der Daten für das Training untersucht.

Zum Abschluss wird noch ein Ensemble aus klassischen und tiefen Lernmethoden betrachtet.

5.1 Stand der Technik

Maschinelles Lernen erhält zurzeit in der Forschung wieder einen großen Aufschwung. Dies begründet sich in dem Erfolg der vergangenen Jahre und dem zunehmenden Bedürfnis aus komplexen Daten Informationen zu gewinnen.

Maschinelles Lernen lässt sich anhand von vielen Merkmalen kategorisieren, z. B. anhand der Lernmethode: Überwachtes Lernen (*engl. supervised learning*), Verstärkendes Lernen (*engl. reinforcement learning*) und Unüberwachtes Lernen (*engl. unsupervised learning*) oder anhand der Diskretisierung: Regression und Klassifikation. Ein weiteres Merkmal zur Kategorisierung, welche sich in den letzten Jahren herausgebildet hat, ist die "Tiefe" der Lernmethoden. So unterscheidet [DY13] beispielsweise zwischen tiefen Lernmethoden (*engl. Deep Learning*) und flachen Lernmethoden (*engl. Shallow Learning*).

Eine gute Charakterisierung dieser Klassifikationsverfahren lässt sich in Abbildung 5.1 nach [GBC16] finden. Hierbei werden die verschiedenen Elemente eines Lernalgorithmus dargestellt und verdeutlicht, welche Teile des Lernalgorithmus die Möglichkeit haben direkt aus den Daten zu lernen und welche Teile durch Experten designt werden.

Ganz links im Bild wird ein regelbasiertes System dargestellt, bei diesem bringt der Experte das ganze Wissen in Form eines Algorithmus und händisch ausgewählten Parametern ein. Der menschliche Experte mag von den Daten lernen, es wird

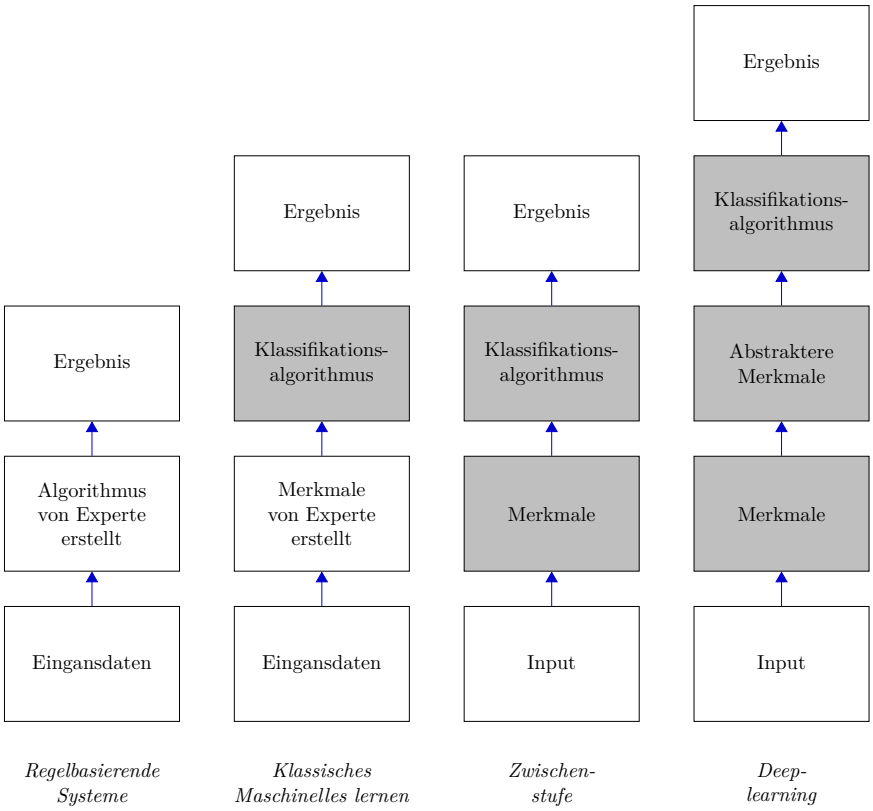


Abbildung 5.1: Einteilung der Klassifikationsverfahren nach [GBC16].

für den Algorithmus jedoch keine Information direkt aus den vorhandenen Daten gewonnen.

Der nächste Ansatz ist die "klassische" maschinelle Lernmethode. Bei diesen werden die Merkmalsextraktoren ebenfalls von Experten entwickelt. Diese projizieren die Eingangsdaten üblicherweise in einen niedrigdimensionalen Raum. Die Trennfunktion bzw. die Regressionsfunktion wird jedoch durch einen Lernalgorithmus aus gelabelten Daten gewonnen. Klassische typische Vertreter für die Lernalgorithmen sind beispielsweise ein *Bayes Klassifikator* [Bis06], ein *Random-Forest* [Bre01] oder eine *Support Vector Machine* [CV95] mit Merkmalsextraktoren wie z. B. HOG [McC86], SIFT [Low04; Low99] oder auch einer einfachen Mittelwertbildung.

Die nächsten Stufen ersetzen die Merkmalsextraktoren ebenfalls durch Lernalgorithmen. Hier wird neben der eigentlichen Klassifikationsentscheidung auch eine Zwischenrepräsentation gelernt. Eingangsdaten werden dabei ohne Vorverarbeitung

in den Algorithmus gegeben. In der Praxis stimmt das natürlich nicht ganz, da beispielsweise bei der Kamera, schon beim Weg vom Bildsensor zum fertigen Bild Algorithmen eingesetzt werden und damit eine Vorverarbeitung stattfindet, dabei wird jedoch weitestgehend auf merkmalsextrahierende Algorithmen verzichtet. Dieses Vorgehen kann beispielsweise mit mehrschichtigen Neuronalen Netzwerken abgebildet werden, bei dem die ersten Schichten die Merkmale extrahieren, während weitere Schichten die Klassifikationsentscheidung treffen. Eine weitere Möglichkeit ist ein *Random-Forest*, der direkt auf der Pixelebene angewandt wird [SJC08].

Die vorherige und die letzte Methode unterscheiden sich nur noch in der Tiefe der Merkmalsextraktoren. Tiefe Lernmethoden sind dadurch charakterisiert, dass mehrere Stufen dieser Merkmalsextraktoren kaskadiert sind und dadurch abstraktere Merkmale bestimmt werden können. Während beispielsweise Kanten in einer frühen Ebene detektiert werden, wird danach durch die Kombination verschiedener Merkmale ein abstrakteres Merkmal berechnet, welches beispielsweise aussagt, ob sich eine Hand in dem Bild befindet. Aus einer Kombination von solchen abstrakten Merkmalen entscheidet der Klassifikator, welche Objektklasse sich im Bild befindet. Ein Vorteil dieser Methode ist, dass die extrahierten Merkmale oftmals so abstrakt sind, dass diese in ihrer trainierten Form für andere Problemstellungen eingesetzt werden können und dadurch lediglich Teile der Merkmalsextraktoren und des Klassifikationsalgorithmus auf das spezifische Problem trainiert werden müssen.

In den letzten Jahren haben tiefe Lernmethoden zu großen Erfolgen im Bildverstehen geführt und lösten z. B. in dem ImageNet Wettbewerb [Rus+15] mit [KSH12] klassische Methoden wie beispielsweise den Gewinner aus dem vorherigen Jahr [PS11] ab.

Der Erfolg von *Deep Learning* Methoden hängt jedoch sehr von der Qualität und Quantität der Trainingsdaten ab. Bei einer zu geringen Anzahl an Trainingsbeispielen tendieren sie aufgrund ihrer hohen Zahl an Parametern zum Auswendiglernen (*engl. Overfitting*). Zudem ist das Training dieser Methoden rechentechnisch sehr kostenintensiv.

Eine Problematik bei tiefen Lernstrukturen ist, den Datenfluss, also das Extrahieren der Merkmale eines Objektes und dessen Klassifikation, nachzuvollziehen. Fehlklassifikationen sind deswegen oftmals nur schwierig zu erklären.

Es wurde beispielsweise in der Verkehrszeichenerkennung gezeigt, dass ein reales Verkehrszeichen durch kleine Aufkleber so manipuliert werden kann, dass ein Stoppschild als Geschwindigkeitsbegrenzung erkannt wird [Evt+18]. Ähnliches wurde auch vorher mit digital manipulierten Bildern gezeigt z. B. in [Sze+13].

Um neuronale Netzwerke besser zu verstehen, haben sich mehrere Arbeiten mit der Visualisierung von Netzeigenschaften beschäftigt [ZF14; NYC16].

Im Folgenden soll nun auf die Klassifikationstechniken eingegangen werden. Da dies ein sehr weites Feld ist, wird sich auf die in der Arbeit verwendeten Techniken und deren Umfeld beschränkt, der Random-Forest mit den dafür benötigten vom Experten designten Merkmalsextraktoren und neuronale Faltungsnetze. Im Weiteren soll ein Überblick der Objektklassifikation im automobilen Umfeld, und im Bereich der Radarverarbeitung geben werden.

5.1.1 Random-Forest

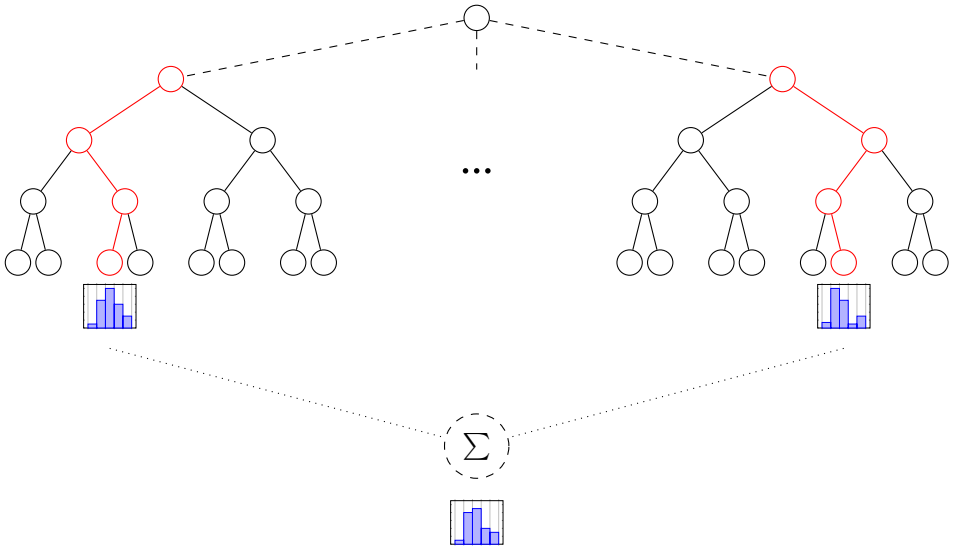


Abbildung 5.2: Schematische Darstellung eines *Random-Forest* Klassifikator.

Der *Random-Forest* Klassifikator wird auf Bildern meist nach der Merkmalsextraktion eingesetzt, und ist damit in der Regel ein Vertreter der klassischen Lernmethoden. Ausnahmen sind beispielsweise bei Shotton et al. zu finden [SJC08]. Gegenüber anderen wie z. B. *Support-Vector-Machines* zeichnet er sich durch ein schnelles und kosteneffizientes Training aus. Durch die überschaubare Anzahl an Hyperparametern gestaltet sich das Trainieren ebenfalls relativ einfach.

1999 prägte Leo Breiman [Bre01] den Begriff *Random-Forest*. Ein *Random-Forest* besteht aus einer Gruppe von Entscheidungsbäumen. Er ist daher auch ein Vertreter der Ensemble Lernmethoden.

Die Bäume werden dabei einzeln, unabhängig voneinander trainiert. Durch verschiedene zufallsbasierte Methoden beim Training der einzelnen Bäume, wie die zufällige

Auswahl der Trainingsbeispiele (*engl. bagging*) und die zufällige Auswahl der Merkmale (*engl. feature bagging*), überwindet der *Random-Forest* das generelle Problem der Überanpassung (*engl. overfitting*) von Entscheidungsbäumen auf den Trainingsdatensatz [HTF09].

Bei der Inferenz werden wie in Abbildung 5.2 dargestellt, die Ergebnisse der einzelnen Bäume zusammengefasst und daraus das Klassifikationsergebnis berechnet.

Nebenbei kann ein *Random-Forest* auch ein Maß für die Bedeutung einzelner Merkmale bestimmen und lässt sich zudem leicht für das Training und die Inferenz parallelisieren [ACS12].

5.1.2 Neuronale Netze

Neuronale Netze sind eine schon lange existierende Klassifikationstechnik, die ersten Konzepte reichen zurück bis in die 1940er Jahre [MP43]. 1979 führte Fukushima unter dem Begriff *Neocognition* das Konzept von neuronalen Faltungsnetzen (*engl. Convolutional Neural Networks (CNN)*) ein [Fuk79; Fuk80]. LeCun et al. verwendete für solche Netze den Backpropagation-Algorithmus, der die Grundlage der meisten heute verwendeten Optimierungsalgorithmen bildet, und prägten den Begriff *convolution* [LeC+89; Le +90].

Nach einem "Hype" in den 1980er und den frühen 1990er Jahren spielten Neuronale Netze eine untergeordnete Rolle. Erst um 2006 mit dem *Deep Belief Network* [HOT06; HS06] erlangten sie auch mithilfe von gesteigerter Rechenleistung erneut Popularität [DY13]. Danach war die Erfolgsgeschichte der Neuronalen Netzwerke nicht mehr zu bremsen. So verdrängten sie, durch die tiefen Netzwerkarchitekturen, in zahlreichen Wettbewerben, viele der klassischen Klassifikationsverfahren [Sch15; Rus+15].

Neuronale Netzwerke werden aus verschiedenen Elementen zusammengesetzt, die meist in Schichten konstruiert sind. Hierzu gehören unter anderem Parameter behaftete Schichten wie Faltungsschichten, Transponierte-Faltungsschichten und Voll-Verbundene Schichten. Diese bestehen hauptsächlich aus Matrizenmultiplikationen mit gelernten Gewichten, gefolgt von einer nichtlinearen Aktivierung. Hierzu werden neben der Sigmoid und der Tangens hyperbolicus Aktivierung heute hauptsächlich das ReLU (*engl. rectified linear unit*) [Hah+00; GBB11] eingesetzt.

Pooling-Schichten wie Max-Pooling oder Average-Pooling sind parameterlos, sie werden zur Dimensionsreduktion eingesetzt.

Mittlerweile gibt es zahlreiche Software-Frameworks, die diese Architekturelemente auf den verschiedenen Hardware-Plattformen umsetzen [Jia+14; Aba+15; Pas+17;

The16] . Diese können dann, entweder durch Programmierung oder durch eine Beschreibungssprache zu einer Netzwerkarchitektur umgesetzt werden.

Auch wenn auf dieser Grundlage mannigfaltige Architekturen möglich sind, haben sich verschiedene Netzwerkarchitekturen durchgesetzt, wie bspw. das *LeNet* [Lec+98], das *AlexNet* [KSH12], das *Googlenet* [Sze+15] oder das *VGG16* [SZ15].

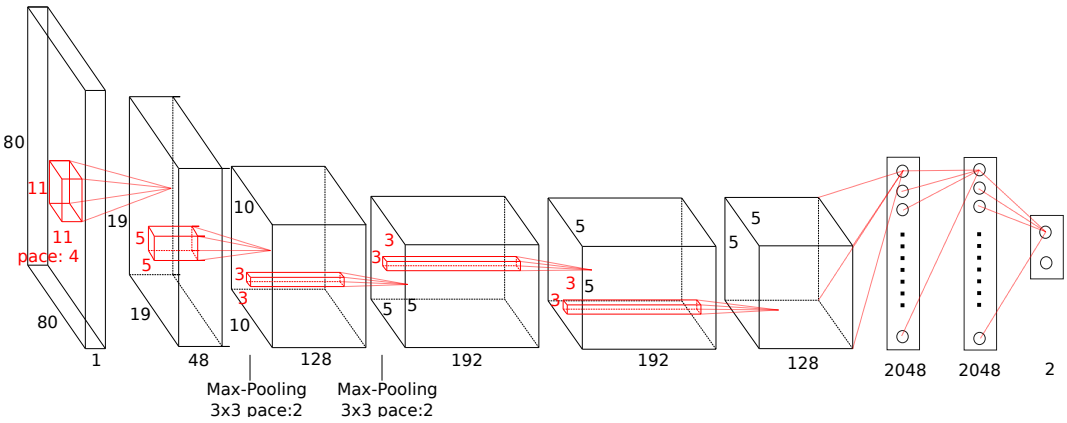


Abbildung 5.3: Variante des *AlexNets* [KSH12].

Bei diesen Architekturen setzt sich der Großteil der Schichten, wie in Abbildung 5.3 am Beispiel des *AlexNets* zu sehen, aus Faltungsschichten zusammen. Da beim Extrahieren der Merkmale oft nur die unmittelbare Umgebung einer Zelle relevant ist, können dadurch unnötige Rechenoperationen vermieden werden, da nur die Zellen in der Größe des Faltungskerns mit einbezogen werden. Ferner wird derselbe Faltungskern auf die komplette Merkmalskarte angewandt, dadurch kann die Anzahl der zu optimierenden Parameter klein gehalten werden. Das Trainieren wird darüber hinaus von positionsinvarianten Merkmalen begünstigt. Am Ende wird meist durch eine voll verbundene Schicht die Klassifikation durchgeführt.

Eine Spezialisierung sind die *Fully Convolutional Neural Networks*. Diese werden meist zur pixelweisen Klassifikation (*engl. Semantic Segmentation*) [GBC09; Far+13; LSD14] eingesetzt. Hierbei wird jedem Pixel eine Klasse zugeordnet. Auf diese Netzwerke wird näher in Kapitel 6 eingegangen.

Im Gegensatz zum Menschen sind die bisher beschriebenen neuronalen Netzwerke gedächtnislos. Um bei der Klassifikation von Zeitreihen die Vergangenheit mit in Betracht zu ziehen, werden als weitere Elemente Neuronen mit Gedächtnis, wie einfache rekurrente Schichten und *Long-Short Term Memory Layer* (LSTM) [HS97] eingesetzt.

Da es sich bei neuronalen Netzen hauptsächlich um Matrizenmultiplikationen handelt, können diese mithilfe vieler Recheneinheiten, gut parallelisiert werden. Hierzu zählen neben modernen CPUs insbesondere Grafikprozessoren (GPUs *engl. graphic processing units*), aber auch speziell für den Bereich des maschinellen Lernens konstruierte TPUs (*engl. tensor processing units*) [Jou+17].

Um einen Klassifikator zu trainieren, wird eine Zielfunktion bzw. eine Kostenfunktion benötigt. Für die Klassifikation wird hierfür meist die Kreuzentropie als Kostenfunktion herangezogen [GBC16]. Diese wird dann im Verlauf des Trainings mithilfe der Trainingsdaten minimiert.

Auch wenn andere Verfahren existieren, die erfolgversprechende Ergebnisse zeigen, wie zum Beispiel genetische Algorithmen [Suc+17], werden die meisten neuronalen Netze nach dem Prinzip des Gradientenabstiegs optimiert. Die zahlreichen Lernalgorithmen versuchen hierbei die Parameter des Gradientenabstiegs zu optimieren, damit ein schnelles Lernen ermöglicht und das Konsolidieren in einem kleinen lokalen Minimum vermieden wird. Die existierenden Algorithmen reichen von anfänglich relativ einfachen Methoden wie *Momentum* [Gér17] und *Nesterove Momentum* [Gér17], welche die Abstiegshistorie berücksichtigen, bis hin zu aufwendigeren Techniken wie, *Adagrad* [DHS11], *Adadelta* [Zei12] und *Adam* [KL14], bei der die 1. und 2. Momente des Gradienten berücksichtigt werden.

In dieser Arbeit wurden hauptsächlich *Tensorflow* [Aba+15] und *Theano* [The16] eingesetzt, welche aus einer mathematischen Formulierung eines Graphen ausführbaren Code erzeugen. Diese können mit dem Highlevel-Framework Keras [Cho+15] angesteuert werden, welche die Basisbausteine der Netzwerkstrukturen und Lernalgorithmen beinhalten.

Des Weiteren wurde das in C++ geschrieben Framework *Caffe* [Jia+14] eingesetzt. Für dieses sind eine Vielzahl der üblichen Netzstruktur mit ihren vortrainierten Parametern, z. B. auf dem ImageNet Datensatz, erhältlich.

Um Überanpassung zu vermeiden, werden Regularisierungstechniken wie z. B. *Dropouts* [Sri+14], *Batch Normalization* [IS15], *Early Stopping* [Gér17] und Augmentation genutzt.

Diese kurze Übersicht über neuronale Netzwerke, kann das Gebiet weder umfassend, noch vollständig beschreiben. Daher wird für eine grundlegende und detailliertere Betrachtung auf einschlägige Literatur verwiesen z. B. [GBC16; Gér17; Bis06; Sch15; Guo+16].

5.1.3 Genetische Algorithmen

Genetische Algorithmen basieren auf Mechanismen der Darwin'schen Evolutionstheorie [Dar59; SD07; Kra17].

Bekannt wurden sie durch die Arbeiten von John Holland in den 1960er bis 1980er Jahre [WM12]. Der Begriff genetische Algorithmen wurde durch die Promotion von Ken De Jong 1975 [DeJ75] geprägt.

Algorithmus 7 : Genetischer Algorithmus.

```
1  Initialisierung der Bevölkerung ;
2  do
3  |  while Bevölkerung der neuen Generation nicht vollständig do
4  |  |  Kreuzen ;
5  |  |  Mutieren ;
6  |  |  Genotype-Phenotype Mapping;
7  |  |  Berechnung der Fitness;
8  |  end
9  |  Selektiere Kandidaten zur Fortpflanzung ;
10 while Abbruchbedingung nicht erfüllt;
```

Die grundlegenden Schritte eines genetischen Algorithmus, wie in der Literatur zu finden [Wei15; Poh00], sind im Pseudocode in Algorithmus 7 zu sehen. In Zeile 1 wird eine neue Bevölkerung erzeugt. Hierbei werden die Gene, d. h. die Parameter, zufällig aus einer sinnvollen Verteilung gezogen.

Danach beginnt ein wiederkehrender Ablauf, welche die Bevölkerung Generationsweise weiterentwickelt. Hierzu werden die Individuen der neuen Generation durch Kreuzen erzeugt. Diese Kreuzung kann beispielsweise durch Kombination also übertragen einzelner Parameter oder Interpolation zwischen den einzelnen Werten erfolgen. Anschließend werden durch zufällige Manipulationen, Mutationen erzeugt.

Mit den so erzeugten Individuen erfolgt die Abbildung vom Genotype zum Phenotype, d. h. die Abbildung von den Parametern zur Ausprägung im Problemraum. In manchen Fällen entspricht der Genotype dem Phenotype, in diesem Fall entfällt dieser Schritt.

Anhand des Phenotypes wird mit einem geeigneten Maß die Fitness berechnet. Diese wird für das Selektieren der Kandidaten genutzt, welche für die Fortpflanzung herangezogen werden. Dieser Prozess wiederholt sich, bis eine vorher gewählte Abbruchbedingung erfüllt ist.

Auch hier wird für eine detaillierter Betrachtung auf einschlägige Literatur verwiesen [Wei15; Poh00; RBK12; WM12; Sim13].

5.1.4 Objektklassifikation im automobilen Umfeld

Die zunehmende Anzahl und Komplexität der Fahrerassistenzfunktionen erfordert ein immer besseres Bild der Umgebung. Diese Abbildung der Umgebung hat nicht nur Anforderungen an die Auflösung und Genauigkeit, sondern erfordert auch deren semantische Interpretation. Damit werden Algorithmen zur Klassifikation von Objekten immer bedeutender.

Ein früher Vertreter der Klassifikation ist die Kamera basierende Verkehrszeichenerkennung, bei der sowohl das Verkehrszeichen an sich, wie auch dessen Bedeutung, z. B. Stoppschild, Geschwindigkeitsbegrenzung 30 km/h, etc., erkannt wird [Rit92; LPH93; Si 94; Pri+93].

Durch Spurverlassenswarner und Spurhaltesysteme wurde es notwendig, die Straßenmarkierungen zuverlässig zu erkennen. Es konnte gezeigt werden, dass dies sowohl mit Kamerasystemen [TN96; Ned+04; Li+16] wie auch mit Laser basierenden Systemen [SDS01; Che+17b] möglich ist.

Neben der Infrastruktur ist auch die Klassifikation von anderen Verkehrsteilnehmern von großer Bedeutung. Hier können sowohl Kamera [TGL09; EKG08] wie auch Laserscanner [LZX16; TLT11] und Radarsensoren (siehe Abschnitt 5.1.5.1) eingesetzt werden. Diese werden bspw. für Folgefahrten oder auch für Notbrems- und Ausweichsysteme genutzt. Um Unfälle zu vermeiden oder deren Schwere zu vermindern, werden weitere Information besonders für schwächere Verkehrsteilnehmer (Radfahrer, Fußgänger) benötigt. Eine Absichtserkennung zeigt sich als hilfreich, um mögliche Verkehrssituationen vorauszusagen [KHG11; Koo+14].

Um autonomes Fahren zu ermöglichen, beschäftigen sich neuere Ansätze mit der Klassifikation der gesamten Fahrzeugumgebung. Dies erfolgt mit der Kamera, unter dem Begriff des *Scene Labeling*, indem ganze Objekte in Superpixeln [FG09] zusammengefasst und klassifiziert werden. Die Klassifikationsergebnisse werden dann entweder in Pixeln, oder aus Effizienzgründen in Stixeln dargestellt [Sch+13a; Enz+12].

Radargestützte Ansätze werden im nächsten Abschnitt vorgestellt.

5.1.5 Objektklassifikation mit Radar

Die Klassifikation im Bereich Radar lässt sich in zwei Bereiche unterteilen, das Klassifizieren von sich bewegenden Objekten und das Klassifizieren von statischen Objekten.

Des Weiteren ist eine Einordnung nach Einsatzgebiet möglich, so kann man bspw. zwischen bodengestützten Radaren, Schiffs Radaren, luft-, satellitengestützen Radaren und automobilen Radaren unterscheiden.

5.1.5.1 Klassifikation von dynamischen Objekten

Bei der Klassifikation von dynamischen Objekten wird vorrangig das Doppler-Spektrum als Merkmal zur Klassifizierung herangezogen. Damit kann zum einen leicht zwischen stehenden und dynamischen Objekten unterschieden werden, zum anderen kann durch ein charakteristisches Bewegungsmuster, aber auch zwischen verschiedenen Klassen dynamischer Objekte unterschieden werden. Beispielsweise bei Fußgängern ergibt sich dieses charakteristische Muster über die unterschiedlichen Geschwindigkeiten in der sich Beine, Torso und Arme bewegen. Zhang et al. [ZLC17] gibt eine Übersicht über die Messung der Charakteristika des Dopplers in verschiedenen Frequenzbändern.

Mittlerweile gibt es schon einige Arbeiten im Bereich der Fußgängererkennung. Heuel et al. [HR10] zeigte mit einem 24-GHz-Radar anhand der Verteilung der Doppler-Geschwindigkeit, dem Signal-Rausch-Verhältnis und dem Profil der Entfernungen, die Trennung zwischen Fahrzeugen und Fußgängern. Die Herkunft dieser Merkmale wurden in [HR12] erläutert. Die Klassifikation konnte verbessert werden, indem Merkmale aus dem Tracking hinzugefügt wurden [HR11].

In [HR13] wurde der Einfluss der Wellenlänge auf diese Merkmale dargelegt. Molchanov et al. [Mol+13] zeigte anhand eines simulierten Radars im 77 GHz Band und einem realen Radar im 24 GHz Band ein Klassifikationsverfahren das robust gegenüber dem Frequenzbereich ist. Narayanan und Zenaldin [NZ15] untersuchten den Einfluss unterschiedlicher Aktivitäten auf das Doppler Spektrum. Hierbei wurde unter anderem das normale Laufen, Stehen, Stehen mit schwingenden Armen, Tragen eines Rucksacks und der Einfluss eines zweiten Fußgängers betrachtet.

Vignaud et al. [Vig+09] zeigten, wie mit hochauflösenden Radaren dieses Spektrum besser analysiert werden kann. Schubert et al. [Sch+15] erweiterten dies, indem er mit hochauflösenden Radaren, sowohl Fußgänger wie auch Fahrradfahrer betrachtet.

Viele dieser Ansätze wurden im Labor oder mit speziellen Messaufbauten untersucht. Schumann et al. [Sch+17b] zeigten mithilfe eines datengetriebenen Ansatzes, dass es möglich ist, verschiedene Klassen auch auf realen Daten im Straßenverkehr zu unterscheiden. Hierzu setzten sie einen Random Forest und LSTM Klassifikatoren ein. Die Klassifikation findet dabei anhand von aus Clustern extrahierten Merkmalen statt, welche anhand der CFAR gefilterten Daten erstellt worden sind. Die untersuchten Klassen waren Fahrzeuge, LKWs, Fußgänger, Fußgängergruppen, Fahrradfahrer und eine Restklasse.

Da insbesondere Radar Clustering Verfahren fehleranfällig sind, zeigten Schumann et al. [Sch+18a] mithilfe eines modifizierten PointNet++ [Qi+17], dass eine semantische Segmentierung ohne vorheriges Clustering möglich ist. Die Entwicklung von Objektklassifikation ist mittlerweile so weit fortgeschritten, dass Radar Dummys für den Euro NCAP Test entwickelt wurden [Sch+13b], um Radar basierende Funktionen der Fußgängererkennung bewerten zu können.

Das Erkennen von dynamischen Objekten spielt auch außerhalb des automobilen Bereichs eine große Rolle. Neumann et al. [NS13] zeigte beispielsweise für Marineradare die Unterscheidung von Meeresclutter, Windturbinen, Helikopter, und Schiffen.

Smith et al. [SWB10] zeigten für ein bodengestütztes Radar (MSTAR) die Klassifikation zwischen Fahrzeugen, Kettenfahrzeugen und Personen. Zusätzlich wurden noch Ziele mit Propellern synthetisch generiert.

Im Bereich der Luftüberwachung zeigten Zyweck und Bogner [ZB96] die Unterscheidung verschiedener Flugzeugtypen.

5.1.5.2 Klassifikation von statischen Objekten

Ein großer Anwendungsbereich für Semantik ist das Kartografieren der Welt. Die Informationen werden meist aus SAR-Radar Bildern extrahiert. Hier spielt zum Beispiel das Erkennen von Straßen [CI07; Jia+05], Gebäuden [SOG05; Zha99; GHS00a] oder Vegetation [Cor+03] eine wichtige Rolle.

Auch wenn im militärischen Bereich nur wenig veröffentlicht wird, finden sich hier einige Arbeiten, die unter dem Thema der automatischen Zielerkennung (*engl. automated target recognition*) zusammengefasst werden können. Viele dieser Arbeiten basieren auf luft- oder satellitengestützten Radaren. Meist wird hierbei mithilfe von synthetischen Aperturen eine höhere Auflösung ermöglicht.

Novak et al. [NON94] unterscheidet mittels eines räumlichen Korrelationsfilters zwischen verschiedenen militärischen Fahrzeugen. Chiang et al. [CMP00] zeigen anhand

von hochauflösenden SAR-Bildern, einen modellbasierenden Ansatz, solche Fahrzeuge zu erkennen. Petterson et al. [Gar+01] untersuchten den Einfluss des Sendeimpulses auf die Detektion und Unterscheidung von militärischen Fahrzeugen. Perrsion und Ferretti [PF07] analysierten die Unterscheidung von verschiedenen Oberflächen wie z. B. Dächer, Gitter, Pfosten mithilfe Satellitengestützter Radare. Yang et al. [YQL05] stellten anhand des *MSTAR* Datensatz einen Klassifikationsalgorithmus vor. Hierbei wurde mithilfe einer SVM ebenfalls zwischen verschiedenen militärischen Fahrzeugen unterschieden. Einen weiteren Überblick über Arbeiten zur Radarzielklassifikation ist in [Kou10] zu finden.

Ein weiteres Anwendungsfeld ist im Bereich der Geo Informations Systeme zu finden, hierbei wird z. B. für das Erstellen von Landkarten die automatische Erkennung verschiedener Kartenelemente benötigt [May99]. Hierzu können optische Systeme wie auch Radar Systeme eingesetzt werden. Dadurch dass diese Elemente ortsfest sind, kann die Karte auch mittels mehrfachen Überfliegens akkumuliert werden. Mithilfe des unterschiedlichen Rückstreuverhalten unterscheiden Ainsworth et al. [ASL08] Menschen gemachte Strukturen von natürlichen Gebieten. Das Erkennen von Gebäuden wurde in [GHS00b] gezeigt. Wie man Straßen aus urbanen Gegenden extrahieren kann, zeigte Hedman [HHS06] mithilfe von SAR Bildern.

Raina et al. [RMU12] stellten eine Möglichkeit vor, wie ein Roboter anhand eines lernenden Verfahrens Hindernisse von Gras bewachsenen Untergrund unterscheiden kann.

Cherniakov et al. [Che+06] zeigte mit einem *Forward-Scattering-Radar* eine Möglichkeit, Bodenziele und verschiedene Fahrzeugklassen zu unterscheiden. Dube et al. [Dub+14] zeigten mithilfe von Radar-Belegungskarten sowohl quer wie auch längs geparkter Fahrzeuge zu erkennen. Urazghildiiev et al. [Ura+07] zeigten mithilfe eines fest installierten Radars, wie man mittels verschiedener Höhenprofile zwischen verschiedenen Fahrzeugen unterscheiden kann. Prophet et al. [Pro+17] zeigte, basierend auf Radarzielen, eine Möglichkeit zur Erkennung von Parklücken. Hierbei wird davon ausgegangen, dass sich die Parklücke rechtwinklig zur gefahrenen Trajektorie befindet.

In vielen Arbeiten im automobilen Bereich wurden oft Einschränkungen getroffen, wie die Beschränkung, dass nur rechtwinklig geparkte Fahrzeuge erkannt werden. Oder das Vorwissen, das man sich auf einem Parkplatz befindet. Eine Arbeit, die eine semantische Klassifikation allgemeiner untersucht, ist dem Autor nicht bekannt.

5.2 Klassische Klassifikationsverfahren

Um Daten mithilfe von klassischen Klassifikationsverfahren, wie z. B. dem *Random-Forest* zu klassifizieren, müssen zunächst aussagekräftige Merkmale extrahiert werden. Direkt die Merkmalskarten als Eingangsdaten zu nutzen, ist aufgrund der Anzahl der Variablen und der geringen Bedeutung einer einzelnen Zelle nicht zielführend.

Diese Merkmale werden mithilfe verschiedener statistischer Methoden extrahiert. Die meisten haben ihre Ursprünge in der Bildverarbeitung. Aus diesen Merkmalen wird ein Merkmalsvektor für jedes Beispiel gebaut und damit ein *Random-Forest* trainiert.

Im Folgenden werden zunächst die Merkmalsextraktoren beschrieben und anschließend auf das Training des *Random-Forest* eingegangen.

5.2.1 Merkmalsextraktoren

Für die Extraktion von Merkmalen werden zwei verschiedene Karten verwendet. Die Belegungskarte und die RCS-Maximum-Karte. Letztere hat sich in Experimenten als aussagekräftiger als die RCS-Mittelwert-Karte erwiesen. Alle Merkmale wurden aus nicht augmentierten Beispielen extrahiert.

Hierzu wird ein Ausschnitt der Belegungskarte im Umfeld des extrahierten Clusters herangezogen. Mithilfe dieser Belegungskarten und 10 Schwellwerten ψ_0 … ψ_{10} werden binäre Schwellwertkarten erstellt. Hierzu wird der Schwellwert auf den gesamten Kartenausschnitt angewandt. Alle Zellen, die größer als der Schwellwert sind, erhalten den Wert “1”, alle anderen “0”. Anschließend werden die zusammenhängenden Flächen bestimmt. Lediglich die zusammenhängende Fläche, welche die größte Überlappung zu dem ursprünglichen Clusterpolygon besitzt, wird behalten, die restlichen Zellen werden zu “0” gesetzt. Auf diesen binären Karten werden verschiedene Merkmale berechnet.

Die Merkmale bestehen aus der **Fläche** der Zellen des Objektes über dem Schwellwert. Zusätzlich wird der **Umfang** des Ausschnittes mithilfe einer konvexen Hülle bestimmt. Des Weiteren wird die **Länge** und **Breite** des minimalen, umschließenden Rechtecks bestimmt.

Ferner wird ein normalisiertes **Histogramm**, der in dem Cluster liegenden Zellen der Belegungskarte gebildet. Hierzu wird der Wertebereich in 10 Intervalle zerlegt. Dasselbe wird noch einmal für die Zellen der RCS-Maximumkarte wiederholt.

Tabelle 5.1: Extrahierte Merkmale

Merkmal	Karten	Parameter
Fläche	Schwellwert Karten (1-10)	10
Umfang	Schwellwert Karten (1-10)	10
Umfang konvexe Hülle	Schwellwert Karten (1-10)	10
Breite	Schwellwert Karten (1-10)	10
Länge	Schwellwert Karten (1-10)	10
Mittelwert	Maskierte Belegungskarten (1-10)	10
Median	Maskierte Belegungskarten (1-10)	10
Maximum	Maskierte Belegungskarten (1-10)	10
Standardabweichnung	Maskierte Belegungskarten (1-10)	10
Minimum	Maskierte RCS-Maximum-Karten (1-10)	10
Mittelwert	Maskierte RCS-Maximum-Karten (1-10)	10
Median	Maskierte RCS-Maximum-Karten (1-10)	10
Maximum	Maskierte RCS-Maximum-Karten (1-10)	10
Standardabweichnung	Maskierte RCS-Maximum-Karten (1-10)	10
Histogramm	Maskierte RCS-Maximum-Karten (1-10)	100
Histogramm	Belegungskarte	10
HOG	Belegungskarte	1352
Annular Statistic Descr.	Belegungskarte	16
Annular Statistic Descr.	RCS-Maximum-Karte	16
Summe		1634

Weiterhin wird das Merkmal **Histogramm of Orientation Gradient (HOG)** [FR95] der Belegungskarte bestimmt. Diese wird auf dem Ursprungscluster ohne Schwellwertbildung bestimmt.

Das letzte Merkmal ist der **Annular Statistic Descriptor** [Rap+15]. Dieser wurde zur Wiedererkennung von Landmarken entwickelt. Sein großer Vorteil liegt in seiner Invarianz zu Rotationen.

Diese extrahierten Merkmale werden zu einem Vektor zusammengefügt. Tabelle 5.1 enthält die Übersicht aller Merkmale und ihrer Größe.

5.2.2 *Random Forest* Klassifikator

Als Klassifikator wurde ein *Random-Forest* Klassifikator implementiert in *scikit-learn* [Ped+11] eingesetzt. Dieser wurde ausgewählt, da er sich gegenüber anderen Verfahren wie z. B. die Support-Vector-Machine einfach und schnell trainieren lässt. Des Weiteren ermöglicht der *Random-Forest*, die Wichtigkeit einzelner Merkmale zu analysieren.

Als Eingangsdaten für die Merkmalsextraktoren wurden die Belegungskarte und die RCS-Maximum-Karte aus dem Cluster-Datensatz verwendet. Auf eine Out-Of-Bag Evaluierung wurde verzichtet, da bei dieser nicht sichergestellt werden kann, dass stark korrelierte Beispiele sowohl zum Training wie auch zur Evaluierung verwendet werden und dadurch Verzerrung in den Ergebnissen auftreten. Daher wird für die Selektion des geeigneten Modells der Validationsdatensatz verwendet.

5.3 CNN-basierende Klassifikationsverfahren

In einem weiteren Schritt werden nun tiefe Lernmethoden anhand des Cluster Datensatzes untersucht. Hierzu werden Faltungsnetzwerke eingesetzt. Anders als bei den merkmalsbasierenden Methoden, erhält der Klassifikator direkt die Merkmalskarten des Clusters als Eingang.

Wenn beispielsweise die Belegungskarte als Eingang verwendet wird, erhält das Netzwerk eine 2-D Matrix als Eingangsdaten. Um mehr als eine Merkmalskarte zu kombinieren, werden diese ähnlich der Farbkanäle bei einem Bild übereinandergestapelt. Die Eingangsmatrix wird dadurch mit den Dimensionen Höhe, Breite und Kanäle zu einer 3-D Tensor.

Für Merkmalskarten wie die RCS-Histogramm-Karte, die schon aus einem 3-D Tensor besteht, wird keine weitere Dimension hinzugefügt, stattdessen werden die einzelnen Schichten der Karte als separate Kanäle hinzugefügt. Dies ist sinnvoll, da es sich hierbei nicht um eine dritte Dimension in Weltkoordinaten handelt, sondern nur um unterschiedliche Aspekte einer Merkmalskarte.

Durch dieses Stapeln der Karten bleibt der räumliche Bezug der Zellen erhalten, d. h. die Zellen, die in Höhe und Breite denselben Index haben, repräsentieren den gleichen Bereich im Weltkoordinatensystem.

Da anhand Minibatches trainiert wird, setzt sich die Eingangsmatrix aus folgenden Dimensionen zusammen.

$$[\text{Batchgröße} \times \text{Höhe} \times \text{Breite} \times \text{Kanäle}]$$

Für die Experimente wird sich an zwei verschiedenen Netzwerkarchitekturen orientiert, dem aus dem Jahre 2012 stammenden *AlexNet* [KSH12] sowie dem *GoogleNet* in der Version aus dem Jahre 2015 [Sze+15; Sze+16].

Im Folgenden werden die verwendeten Architekturen und Algorithmen zum Training beschrieben.

5.3.1 Netzwerk Architektur

Das *AlexNet* ermöglicht, aufgrund seiner geringen Anzahl an Schichten, ein schnelleres Training. Durch die geringere Anzahl an Operatoren sind weniger Rechenschritte erforderlich, ebenso ist die Problematik des verschwindenden oder explodierenden Gradienten durch die wenigen Schichten geringer. Bei einem zufällig initialisierten Netzwerk führt das Training sicherer zu einem Erfolg. Das Netzwerk ist in Abbildung 5.3 auf Seite 115 zu sehen.

Beim *AlexNet* wird auf die Aufteilung des Netzwerkes auf zwei Recheneinheiten verzichtet. Somit besteht das Netzwerk aus fünf Faltungsschichten. Zusätzlich findet nach der ersten, zweiten und vierten Schicht ein *Max-Pooling* statt. Die Faltungsschichten sind so aufgebaut, dass sie die Dimensionen des Eingangsbildes verringern und gleichzeitig die Anzahl der Kanäle vergrößern. Die Größe der Faltungskerne verringert sich in den tieferen Schichten ebenfalls. Nach den Faltungsschichten befinden sich zwei voll verbundene Schichten, gefolgt von der Ausgangsschicht. Außer der Ausgangsschicht verwenden alle Netzwerkschichten eine *ReLU* Aktivierungsfunktion. Um das Ergebnis als Wahrscheinlichkeitsverteilung über die Klassen interpretieren zu können, wird eine *Softmax* Aktivierungsfunktion verwendet.

Das Netzwerk besitzt ca. 40 Mio. Parameter, wobei der Großteil der Parameter in den voll verbundenen Schichten steckt.

Die weitere eingesetzte Netzwerkarchitektur ist das *GoogleNet*, welche ebenfalls weitverbreitet ist und zu vielen Klassifikationsmethoden im Bereich des Bildes hinzugezogen wird. Bei einer zufälligen Initialisierung ist dieses Netzwerk aufgrund seiner vielen Schichten deutlich schwerer zu trainieren.

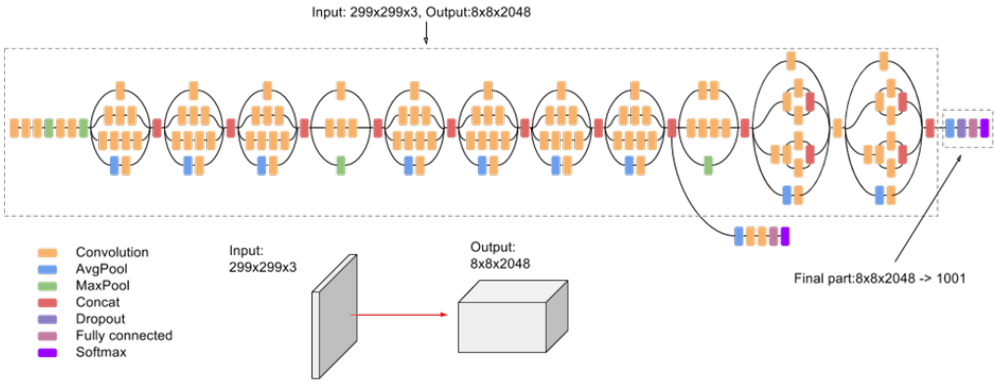


Abbildung 5.4: *GoogleNet Inception v3* Architektur (aus `https://cloud.google.com/tpu/docs/inception-v3-advanced`).

Das Kernelement des *GoogleNet* besteht aus den verschiedenen *Inception* Schichten, die in der *Inception v3* Architektur noch einmal gegenüber dem ursprünglichen Netzwerk verbessert wurde. Hierzu wurde beispielsweise die Anzahl der Parameter der Faltungsschichten reduziert, in dem die Faltungsschichten mit großen Filtern durch 2 Faltungsschichten mit kleineren Filtern ersetzt wurden.

Wie in Abbildung 5.4 zu sehen, haben die Inception Schichten 3 verschiedene Ausprägungsformen und das Netzwerk setzt sich neben Eingangs- und Ausgangsschichten aus 11 solcher Inception Schichten zusammen.

Obwohl das *GoogleNet* deutlich mehr Schichten besitzt als das *AlexNet*, ist die Anzahl der Parameter mit ca. 21 Millionen geringer.

Je nach Klassifikationsaufgaben werden noch die Ein- und Ausgangsschichten angepasst. Dies betrifft beim Eingang die Anzahl der Kanäle und ggfs. die Größe der Ausschnitte. Bei der Ausgangsschicht des Netzwerkes hängt dies von der Anzahl der zu unterscheidenden Klassen ab.

5.3.2 Training

Für alle Experimente wird *Adam* [KL14] als Optimierungsalgorithmus verwendet. Dieser basiert auf dem Verfahren des Gradientenabstiegs und bezieht neben der Lernrate, den Verlauf des 1. und 2. Moment des Gradienten mit ein.

Die Trainings- und Validationsdaten werden für alle Experimente balanciert, sodass von jeder Klasse gleich viele Beispiele vorhanden sind. Dies wird erreicht, indem zufällig ausgewählte Beispiele von unterrepräsentierten Klassen dupliziert werden. Für den Testdatensatz ist dies nicht notwendig, da dieser immer komplett ausgewertet wird, und der Effekt durch einen gewichteten Mittelwert kompensiert werden kann.

Die Daten für einen Minibatch der Trainings- und Validationsdaten werden zufällig, durch ziehen ohne zurücklegen, bestimmt. Erst wenn der Datenpool leer ist, werden alle Daten wieder zurückgelegt.

Wie in Abschnitt 4.3.3 vorgestellt, können verschiedene Augmentierungstechniken eingesetzt werden, um den Trainingsdatensatz zu erweitern. Hierdurch wird das Training von zusätzlichen Hyperparametern beeinflusst. Um diese Abhängigkeit zu minimieren, wurde die Augmentierung für die Basisexperimente auf die Rotation beschränkt. Hierzu wird ein Trainingsbeispiel um einen zufälligen Winkel im Bereich von $\pm$ 180 Grad rotiert. Der Winkel wurde aus einer Gleichverteilung gezogen.

Test- und Validationsdaten werden nicht augmentiert, um eine Verfälschung der Ergebnisse durch evtl. unrealistische Veränderungen zu vermeiden.

5.3.2.1 Augmentierungstechniken

Eine weitere Möglichkeit die Robustheit des Klassifikators zu verbessern und damit seine Performance zu steigern, sind Augmentierungstechniken. Diese gehören zu den Regularisierungstechniken, die auf den Eingangsdaten angewendet werden, um die Merkmalsextraktoren robuster gegen Variationen in den Eingangsdaten zu machen. Die einzelnen Techniken wurden in Abschnitt 4.3.3 vorgestellt.

Da diese von vielen Parametern abhängen und sie damit das Training sowohl negativ wie auch positiv beeinflussen können, wird die Augmentierung separat untersucht.

In Tabelle 5.2 sind noch einmal alle Parameter zusammengefasst. Da zum Training auch nicht augmentierte Beispiele verwendet werden sollen, gibt es einen Bernoulli-Prozess der bestimmt, mit welcher Wahrscheinlichkeit das Beispiel überhaupt augmentiert und welcher Anteil ohne Augmentierung zum Training benutzt werden soll.

Dasselbe wird dann für jeden Augmentierungsparameter wiederholt, sodass nicht alle Augmentierungstechniken für ein Beispiel eingesetzt werden.

Tabelle 5.2: Augmentierungsparameter

Augmentierungstechnik	Auftretens-wahrscheinlichkeit	Verteilung der Parameter
Augmentierung	p_{aug}	
Translation	p_{trans}	$\mathcal{N}(0, \upsilon_{\text{trans}})$
Rotation	p_{rot}	$\mathcal{N}(0, \upsilon_{\text{rot}})$
Skalierung	p_{scale}	$\mathcal{N}(0, \upsilon_{\text{scale}})$
Multiplikativer Niveauverschiebung	p_{multi}	$\mathcal{N}(0, \upsilon_{\text{multi}})$
Additive Niveauverschiebung	p_{add}	$\mathcal{N}(0, \upsilon_{\text{add}})$
Gaußsches Rauschen	p_{gauss}	$\lvert\mathcal{N}(0, \upsilon_{\text{gauss}})\rvert$
Salt Rauschen	p_{salt}	$\lvert\mathcal{N}(0, \upsilon_{\text{salt}})\rvert$
Pepper Rauschen	p_{pepper}	$\lvert\mathcal{N}(0, \upsilon_{\text{pepper}})\rvert$

Des Weiteren wird eine Wahrscheinlichkeitsverteilung benötigt, aus der die Parameter der Augmentierung, z. B. bei der Rotation der Winkel, gezogen werden. Als Wahrscheinlichkeitsverteilung wurde jeweils eine Normalverteilung gewählt. Da anzunehmen ist, dass ein großes Bias bei den Augmentierungsparametern unwahrscheinlich ist und um die Anzahl der Parameter klein zu halten, wurde der Mittelwert fest mit 0 angenommen. So wird die Wahrscheinlichkeitsverteilung lediglich durch die Varianz der Normalverteilung definiert.

Für Augmentierungstechniken bei denen die Definitionsmenge der Parameter nur positive Zahlen umfasst, wie z. B. das Gaußsche Rauschen, wird der Absolutwert des aus der Normalverteilung gezogenen Parameters bestimmt.

Die Schwierigkeit besteht darin, diese Parameter zu optimieren. Es ist nicht davon auszugehen, dass der Einfluss der Augmentierungstechniken unkorreliert ist, daher ist eine separate Optimierung nicht zielführend.

Eine systematische Suche, z. B. eine Rastersuche, würde bei den 17 zu optimierenden Parametern und nur 3 Abtastpunkten pro Parameter, einen Suchraum mit mehr als 100 Mio. Kombinationen ergeben. Darüber hinaus besteht die Möglichkeit, dass

sich bei den Augmentierungstechniken das Optimum der Parameter während des Trainingsverlaufs ändert.

Da das vollständige Trainieren eines Neuronalen Netzwerkes mehrere Stunden bis Tage dauert, erscheint weder eine Rastersuche noch eine randomisierte Suche als sinnvoll.

Um sich trotz der Komplexität dem Optimum der Augmentierungsparameter annähern zu können, wurde in dieser Arbeit ein Algorithmus entwickelt, welcher während des Trainings mittels eines genetischen Algorithmus die Augmentierungsparameter optimiert.

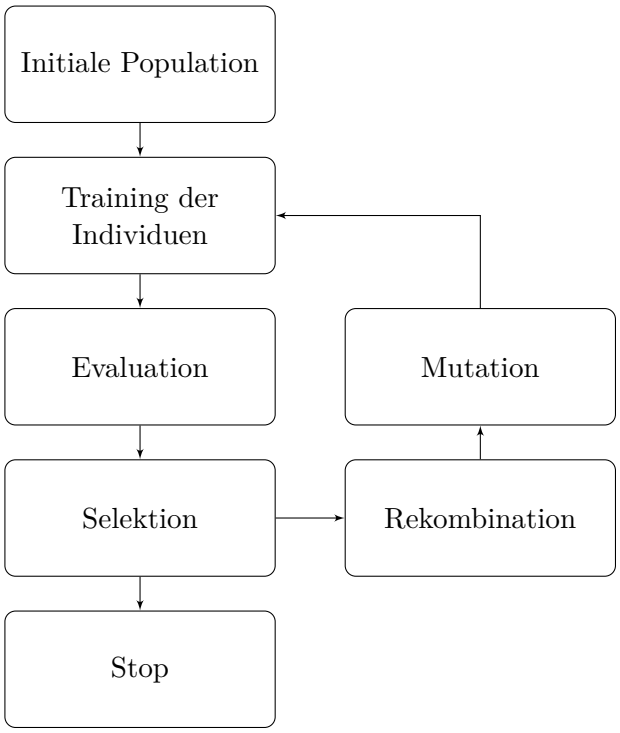


Abbildung 5.5: Genetischer Algorithmus zum Optimieren der Augmentierungsparameter.

Der grundsätzliche Algorithmus ist in Abb. 5.5 dargestellt. Zunächst wird eine Population von η Individuen erzeugt. Jedes Individuum Ind besteht aus dem in Tabelle 5.2 dargestellten Set von Parametern Ψ_{Ind}. Zusätzlich enthält jedes Individuum einen Klassifikator $\hat{\boldsymbol{p}}_{Ind}$ der über die Generationen gen trainiert wird. Dieser besteht aus der Netzwerkarchitektur, den Trainingsgewichten und dem Optimierer mit seinen Parametern und dem internen Status. Die Startparameter für jedes Individuum

werden am Anfang festgelegt. Dies geschieht dadurch, dass diese aus einer sinnvollen Zufallsverteilung gezogen wird.

Im nächsten Schritt wird die Genotype-Phenotype Zuordnung durchgeführt, also wie sich die Parameter (Genotype) auf die Ausprägung (Phenotype) auswirken. Hierzu wird das Netzwerk mit einer bestimmten Anzahl an Trainingsbeispielen trainiert. Dieser Schritt stellt die größten Kosten des Algorithmus dar, da hier die Trainingsdaten augmentiert werden und der Klassifikator trainiert wird.

Im nächsten Schritt wird die Fitness des Klassifikators bestimmt, indem dessen Performance anhand einer bestimmten Anzahl von Validationsdaten bewertet wird. Als Maß für die Fitness wird die Treffergenauigkeit des Klassifikator verwendet.

Wie auch in den vorherigen Experimenten, sind die Klassen sowohl in den Trainings- wie auch den Validationsdaten gleichverteilt. Da die Fitness des Klassifikators nach einem Trainingsschritt stark von den Trainings- und Validierungsbeispielen abhängt, erhalten alle Individuen einer Generation dasselbe Set an Daten.

Nach dem Bestimmen der Fitness jedes Individuums erfolgt die Selektion der Individuen aus denen die nächste Generation hervorgehen soll. Dies geschieht, indem die besten Individuen überleben, dies wird durch ein *Überlebensrate* $p_{Survival}$ bestimmt. Die überlebenden Individuen werden anschließend zum Erzeugen der nächsten Generation verwendet.

Bei der Rekombination wird die neue Generation mittels dreier unterschiedlicher Methoden erzeugt. Erstens werden die besten n_{best} Individuen mit ihren Parametern unverändert übernommen. Zweitens werden, um den Genpool zu erweitern, n_{new} Individuen aus der Anfangsverteilung zufällig erzeugt. Drittens wird die Population mithilfe von Kreuzen wieder auf die ursprüngliche Größe aufgefüllt. Dazu wird die diskrete Rekombination angewendet [Poh00], bei der zufällig zwei Individuen der Elterngeneration selektiert werden. Pro Parameter wird dann zufällig der von dem einen oder dem anderen Elternteil übernommen.

Anders wird bei den Klassifikatoren vorgegangen, hier werden die Klassifikatoren der $n_{best,C}$ Individuen herangezogen. Jedes Individuum der neuen Generation zieht aus diesen zufällig einen Klassifikator als Startpunkt für die nächste Genotype-Phenotype Zuordnung.

Anschließend werden die Parameter der Individuen mutiert. Hierzu werden $n_{mutation}$ Parameter ausgewählt, die zufällig mutiert werden. Als Mutationsverteilung wird eine Gauß-Verteilung angenommen, die durch die Varianz $\sigma_{mutation}$ und dem Mittelwert 0 parametriert wird.

Da aufgrund der begrenzten Rechenzeit die Laufzeit das limitierende Kriterium ist, wird die Abbruchbedingung durch die Anzahl der Generationen festgelegt.

Zusätzlich zu den augmentierten Individuen wurde ein Klassifikator mit denselben Trainingsdaten als Referenz mit trainiert.

Wie durch Experimente herausgefunden wurde, besteht die Schwierigkeit, eine geeignete Anfangsverteilung zu finden. Falls bei diesen Parametern eine schlechte Wahl getroffen wurde, kann es sein, dass der Klassifikator negativ beeinflusst wird und es lange dauert, bis das Training der weiteren Generationen diesen Fehler behebt.

Daher wird zur Bestimmung der Anfangskonfiguration ein weiteres Experiment vorgeschlagen. Dies besteht aus einer leichten Modifikation des oben beschriebenen Algorithmus. Bei diesem wird zum einen der Datensatz stark eingeschränkt. Zum anderen wird der Klassifikator am Anfang jeder Generation neu initialisiert. Die hierdurch gefundene Population, stellt eine gute Anfangskonfiguration für des oben genannte Experiment dar.

5.3.2.2 Vortrainierte Netzwerke

Das Trainieren von tiefen neuronalen Netzwerken ist aufwendig. Es benötigt viel Rechenzeit und große Mengen an Daten, um die Vielzahl an Parametern zu optimieren. Da vor allem gelabelte Daten sehr kostspielig sind, lohnt es sich, Methoden zu finden diese Rechenzeit zu minimieren. Wagner et al. zeigte, dass sich neben anderen Methoden, besonders vortrainierte Netzwerke (*engl. transfer-learning*) lohnen [Wag+13]. Sie demonstrierten dies anhand des MNIST [LC98] und des *Small NORB* [LHB04] Datensatzes. Tajabakhsh zeigt dies anhand von medizinischen Bildern [Taj+16].

Ein vortrainiertes neuronales Netzwerk wird erstellt, indem es auf einem großem Datensatz ähnlicher Struktur trainiert wird. Als Datensatz für Klassifikationen auf Fotos, wird dazu beispielsweise der *ImageNet* Datensatz mit mehreren Millionen Bildern verwendet. Die gelernten Parameter dieses Netzwerkes können nun für die Initialisierung von anderen Klassifikationsaufgaben verwendet werden.

Um ein spezifisches Klassifikationsproblem zu lösen, wird dieses vortrainierte Netzwerk darauf angepasst. Dazu werden meist Schichten am Ende des Netzwerkes entfernt und durch problemspezifische Schichten ersetzt. Für den Fall, dass sich die Struktur der Eingangsdaten verändert hat, müssen auch die ersten Netzwerkschichten angepasst werden. Die neu hinzugefügten Netzwerkschichten werden zufällig initialisiert, während die Gewichte der anderen Schichten erhalten bleiben. Danach wird das Netzwerk mit dem problemspezifischen Datensatz trainiert. Hierbei werden entweder nur die neuen Netzwerkschichten trainiert, oder das gesamte Netzwerk. Dieses Vorgehen ist zum Beispiel in folgenden Arbeiten [RW16; SZ15; DB15; Che+16] zu finden.

Im Bereich von Kamerabildern hat sich gezeigt, dass die vorderen Schichten meist sehr gut allgemeine Merkmalsextraktoren darstellen und damit lediglich die "Klassifizierungsschicht" angepasst werden muss. Daher ist es möglich, mit einem kleinen Datensatz, der das spezifische Problem repräsentiert, gute Erfolge zu erzielen.

Es ist sehr wahrscheinlich, dass dieses Vorgehen auch bei Radar-Belegungskarten funktioniert, z. B. um ein vorher auf Radar-Belegungskarten trainiertes Netz beispielsweise auf eine andere Klasseneinteilung anzupassen.

Viel interessanter ist die Frage, in welchem Umfang Merkmalsextraktoren von CNNs die auf Kameradaten vortrainiert wurden und sich zur Klassifikation von Radardaten eignen. Die Struktur von Radar-Merkmalskarten haben zwar ein ähnliches Format wie Bilder, die Charakteristik der Daten unterscheidet sich aber erheblich. Dem Autor sind keine Arbeiten aus der Literatur bekannt, in denen dies schon einmal untersucht wurde.

Dies soll nun für Radar-Belegungskarten untersucht werden. Das Experiment wurde mithilfe des Trainingsframework *Keras* und einem auf dem ImageNet Datensatz [Jia+09] vortrainierten *GoogleNet* [Sze+16] durchgeführt.

Für das Experiment wurde die erste und letzte Schicht abgeschnitten und durch zufällig initialisierte Schichten ersetzt. Diese wurden dann mit dem im Kapitel 4.3 beschriebenen Datensatz nachtrainiert.

Als Eingangsdaten für des Netzwerk wurde die Belegungskarte und die RCS-Maximum-Karte verwendet. Die Ausgangsklassen werden die Klassen verwendet, die durch das Experiment in Abschnitt 5.5.1 bestimmt wurden.

5.4 Ensemble aus klassischen und tiefen Lernmethoden

Sowohl Abschnitt 5.2, wie auch Abschnitt 5.3 zeigen eine mögliche Form der Klassifikation. Im nächsten Schritt soll untersucht werden, ob mit einer Kombination der Klassifikatoren, einem sogenannten *Ensemble*, eine bessere Performance erzielt werden kann.

Zu beachten ist, dass ein *Random-Forest* schon ein Ensemble-Klassifikator ist, da das Klassifikationsergebnis durch die Kombination einzelner Entscheidungsbäume entsteht. Hier soll jedoch das Ensemble zwischen klassischen und tiefen Lernmethoden untersucht werden. Daher wird das Ergebnis des *Random-Forest* als Ergebnis eines einzelnen Klassifikators betrachtet.

Das Ergebnis von Ensemble-Klassifikationsmethoden hängt nicht nur von ihrer Richtigkeit, sondern auch ihrer Diversität ab [Kun14]. So können schlechtere Klassifikatoren die eine höhere Diversität haben, ein besseres Ergebnis erzeugen, als zwei sich ähnelnde Klassifikatoren.

Ein interessanter Aspekt dabei ist, wie sich die vom Experten designten Merkmalsextraktoren des Random-Forest von den gelernten Merkmalen des CNN unterscheiden. Um dies zu zeigen, werden Klassifikatoren auf demselben Datensatz trainiert. Methoden, welche die Diversität der Klassifikatoren anhand der Daten verstärken, wie z. B. *Bagging*, werden hierfür nicht eingesetzt.

Um Klassifikatoren zu kombinieren sind verschiedene Methoden in der Literatur bekannt. Hier kommen z. B. Gewichtungen [Rok09], Mehrheitsentscheidungen [Kun14], *Perfomance Weighting* [OS96], *Distributed Summation* [CB91] und *Bayesian combination* [Bun92] zum Einsatz.

Bei der *Distributed Summation* werden die Schätzungen der Klassifikatoren, welche als Wahrscheinlichkeitsverteilungen interpretiert werden, aufsummiert. Das Klassifikationsergebnis wird anhand des höchsten Wertes im Vektor bestimmt.

$$Class(\boldsymbol{x}) = \arg\max_{\mathfrak{c}\in\mathfrak{C}} \sum_k \hat{\boldsymbol{p}}_k(\mathfrak{c}|\boldsymbol{x}) \tag{5.1}$$

Bei der *Bayesian Combination* wird für jeden Klassifikator noch zusätzlich die a posteriori Wahrscheinlichkeit, dass der Klassifikator korrekt klassifiziert mit einbezogen. Diese wird anhand der Validationsdaten $\mathfrak{D}_{val}$ bestimmt.

$$Class(\boldsymbol{x}) = \arg\max_{\mathfrak{c}\in\mathfrak{C}} \sum_k P(\mathfrak{c}|\mathfrak{D}_{val})\hat{\boldsymbol{p}}_k(\hat{\boldsymbol{y}}_{pred} = \mathfrak{c}|\boldsymbol{x}) \tag{5.2}$$

Der dritte Ansatz basiert darauf, die Klassifikationsergebnisse mithilfe des naiven Bayes-Klassifikators zu verbinden [Rok09]. Hier werden die Klassifikationsergebnisse mittels Produkt kombiniert.

$$Class(\boldsymbol{x}) = \arg\max_{\mathfrak{c}\in\mathfrak{C}} \hat{P}(\hat{\boldsymbol{y}}_{pred} = \mathfrak{c}) \prod_k \frac{\hat{\boldsymbol{p}}_k(\hat{\boldsymbol{y}}_{pred} = \mathfrak{c})}{\hat{P}(\hat{\boldsymbol{y}}_{pred} = \mathfrak{c})} \tag{5.3}$$

Da nur zwei Klassifikatoren zum Einsatz kommen, entfällt eine Mehrheitsentscheidung. Somit kann nur betrachtet werden, wie sich die Genauigkeit des Klassifikators verbessert, wenn nur einstimmige Klassifikationsergebnisse verwendet werden. Hierbei werden alle anderen Klassifikationsergebnisse zurückgewiesen, dadurch wird natürlich die *Sensitivität* des Klassifikators verringert.

5.5 Auswertung

Nachdem in den vorherigen Abschnitten der prinzipielle Aufbau der Klassifikatoren und deren Training erklärt worden ist, sollen in diesem Kapitel die Ergebnisse einiger Experimente präsentiert werden, um weitere Aufschlüsse über das Klassifizieren von Objekten mithilfe von Radar-Merkmalskarten zu erlangen.

Zum einen soll die Taxonomie verfeinert werden. Hierzu werden Experimente über die Trennbarkeit von Klassen durchgeführt und bewertet. Diese Ergebnisse werden anschließend genutzt, um eine Klasseneinteilung für alle weiteren Experimente zu erhalten.

Mit dieser Klasseneinteilung wird untersucht, welchen Effekt auf Bildern vortrainierte Netzwerke auf das Training von Radar-Belegungskarten haben. Danach werden die verschiedenen Radar-Merkmalskarten als Eingangsdaten für die Klassifikation verglichen. Im Weiteren wird auf den Effekt der Augmentierung eingegangen.

Soweit nicht anders erwähnt, wird für alle neuronalen Netzwerke, *Adam* als Optimierungsalgorithmus mit den in Tabelle 5.3 dargestellten Parametern eingesetzt.

Tabelle 5.3: Parameter des Adam-Optimierers.

Parameter	Wert
Lernrate	0.001
β_1	0.9
β_2	0.999

Um den Einfluss der Klassenverteilung im Datensatz zu sehen, wird bei der Auswertung häufig zwischen Mikro- und Makro-Performanceindikatoren unterschieden. Beim Mikro-Performanceindikator wird der Wert über alle Daten im Test-Set berechnet. Beim Makro-Performanceindikator wird der Wert pro Klasse bestimmt und anschließend über alle Klassen gemittelt, somit wird eine Gleichverteilung der Klassen simuliert.

Im Anschluss werden klassische Klassifikationsverfahren untersucht. Zum Abschluss wird ein Ensemble Klassifikator aus einem klassischen Ansatz und einem tiefen neuronalen Netzwerk Ansatz untersucht.

5.5.1 Taxonomie

In Abschnitt 3.3 wurde Taxonomie schon aus Sicht der Anwendung und aus Sicht des Labeling behandelt. Im Folgenden wird untersucht, welche Objekte in Radar-Belegungskarten mit einem Klassifikator unterschieden werden können. Dieses Experiment wird mithilfe von One-Vs-All Klassifikationsverfahren durchgeführt, bei der jeweils eine Objektklasse von allen anderen Objekten unterschieden wird. Die hier durchgeführten Experimente sind ähnlich zu [Lom+16], jedoch mit deutlich erweiterter Datenbasis. Das Ergebnis wird anhand der ROC Kurven und einer Konfusionsmatrix ausgewertet und damit die in der restlichen Arbeit verwendete Klasseneinteilung festgelegt.

Um den a priori Einfluss, der Häufung der Beispiele für verschiedene Objektklassen, auszuschließen, werden diese ausbalanciert. Dies geschieht, indem Beispiele aus unterrepräsentierten Kategorien vervielfältigt werden. Dies wird sowohl auf Grundlage der verfügbaren Labelkategorien ausgeglichen, sowie in einem zweiten Schritt bezüglich der zwei klassifizierten Klassen des One-Vs-All Klassifikators. Zudem wurden Erkenntnisse aus [Lom+16] beachtet, indem die Klassen *Pfosten (Pole)* und *Verkehrszeichen (Traffic Sign)* zur neu definierten Klasse *Pfosten* zusammengeführt wurden. Und *Büsche (Shurbs)* und *Feld (Field)* zur Klasse *Vegetation* zusammengeführt wurden.

Die Experimente wurden mithilfe eines zufällig initialisierten *AlexNet* durchgeführt.

Durch die getrennte Optimierung, auf den einzelnen Klassen mittels One-vs-All Klassifikatoren, soll eine Abschätzung über die mögliche Trennbarkeit der Klassen mithilfe eines CNNs getroffen werden.

Hierzu wurden pro Objektklasse zehn CNN-Klassifikatoren trainiert. Hierbei wurde das Problem auf ein zwei Klassen Problem reduziert. Die untersuchte Klasse sowie eine Sammelklasse, in der alle anderen Klassen zusammengefasst werden. Die genaue Struktur des CNNs ist in Abbildung 5.3 auf Seite 115 zu finden.

Um ein Bias zu vermeiden, wurde das Netzwerk zufällig initialisiert.

Um die Durchführbarkeit des Experiments zu gewährleisten, wurde das Training auf 200 Epochen beschränkt. Für jede Epoche wurden 600 Batches mit einer *Minibatch*-Größe von 16 verwendet.

Anschließend wird der beste Klassifikator anhand des Validationsdatensatz ausgewählt und mittels Testdatensatz ausgewertet.

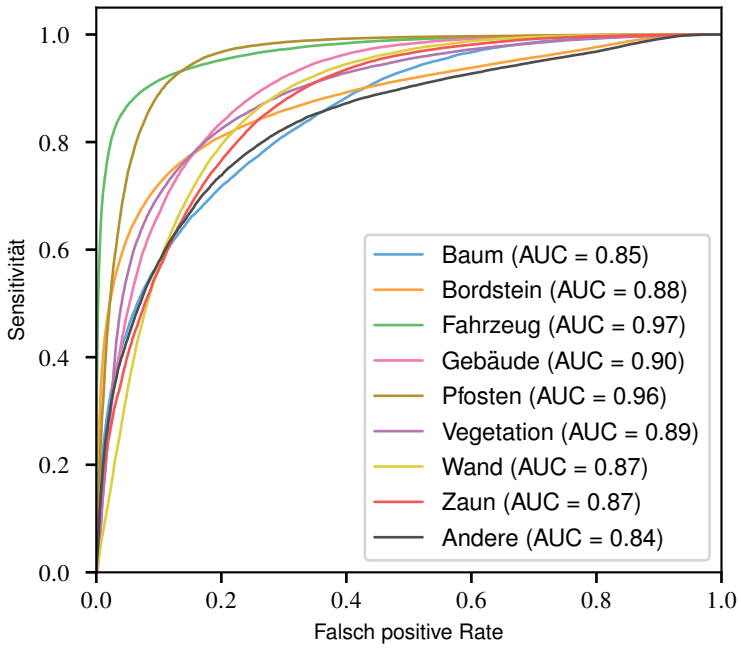


Abbildung 5.6: ROC-Kurve der einzelnen One-vs-All Klassifikatoren.

In Abbildung 5.6 sind die gemittelten ROC (*engl. Receiver Operating Characteristic*) (ROC) Kurven dargestellt. In den Abbildungen 5.7 - 5.9 werden die erweiterten Konfusionsmatrizen angezeigt, diese wurden ebenfalls gemittelt. Hierbei wird, wie bei der normalen Konfusionsmatrix in den Spalten das prädizierte Ergebnis aufgrund der Sammelklassen, angezeigt. In den Zeilen werden jedoch alle gelabelten Klassen aufgeschlüsselt. Damit kann eine präzisere Aussage getroffen werden, welche Label sich im Zweiklassenproblem als schwierig darstellen.

Zu beachten ist, dass die Konfusionsmatrix zeilenweise normiert ist. Daher muss beim Vergleich beachtet werden, dass die prädizierte Klasse ein deutlich stärkeres Gewicht auf die Genauigkeit des Klassifikators hat, als die aufgeschlüsselten Label.

Als die am besten abgrenzbare Klasse zeigt sich *Fahrzeug*. Auf dem ROC Diagramm steigt die Kurve dieser Klasse am steilsten an und hat zudem die größte Fläche unter der Kurve (AUC). In Abb. 5.7 ist zu sehen, dass im geringen Umfang Verwechslungen mit den Klassen *Baum* und *Andere* stattfinden.

Die Klasse *Pfosten* zeigt ebenfalls einen steilen Anstieg auf der ROC Kurve. Bei der Konfusionsmatrix zeigt sich, dass eine hohe Verwechselungsgefahr mit Bäumen

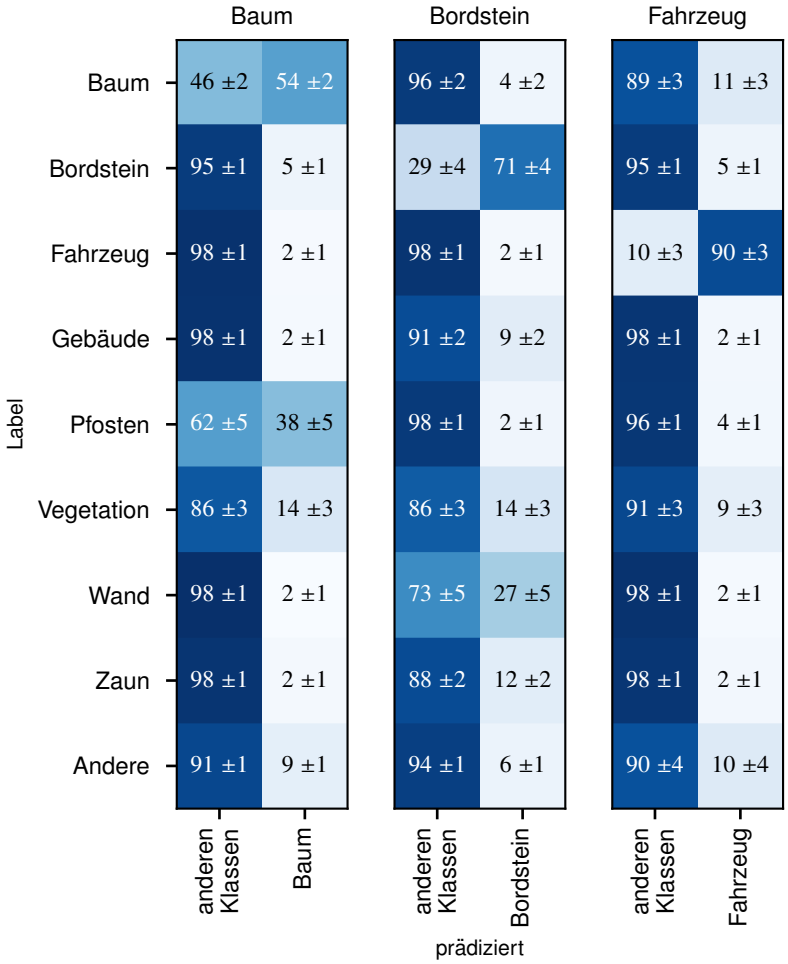


Abbildung 5.7: Erweiterte Konfusionsmatrix: Klassifikationsergebnis versus aller gelabelter Klassen.

besteht, da je nach Baum, dieser entweder nur als Stamm zu sehen ist oder in voller Ausprägung mit seinen Ästen.

Bei den Klassen *Wand*, *Gebäude* und *Zaun*, stellt man in der Konfusionsmatrix eine starke Vertauschung untereinander fest. Daher wurden diese Klassen für die weiteren Untersuchungen zusammengelegt.

Ebenso ist dies bei der Klasse *Baum* und *Vegetation* der Fall, wenn auch nicht so stark ausgeprägt. Besonders in Abb. 5.7 ist dies kaum zu erkennen. Aus diesem Grund und aufgrund der semantischen Ähnlichkeit wurde beschlossen, diese Kategorien zusammen zu legen. Hierbei macht sich auch bemerkbar, dass eine Fichte

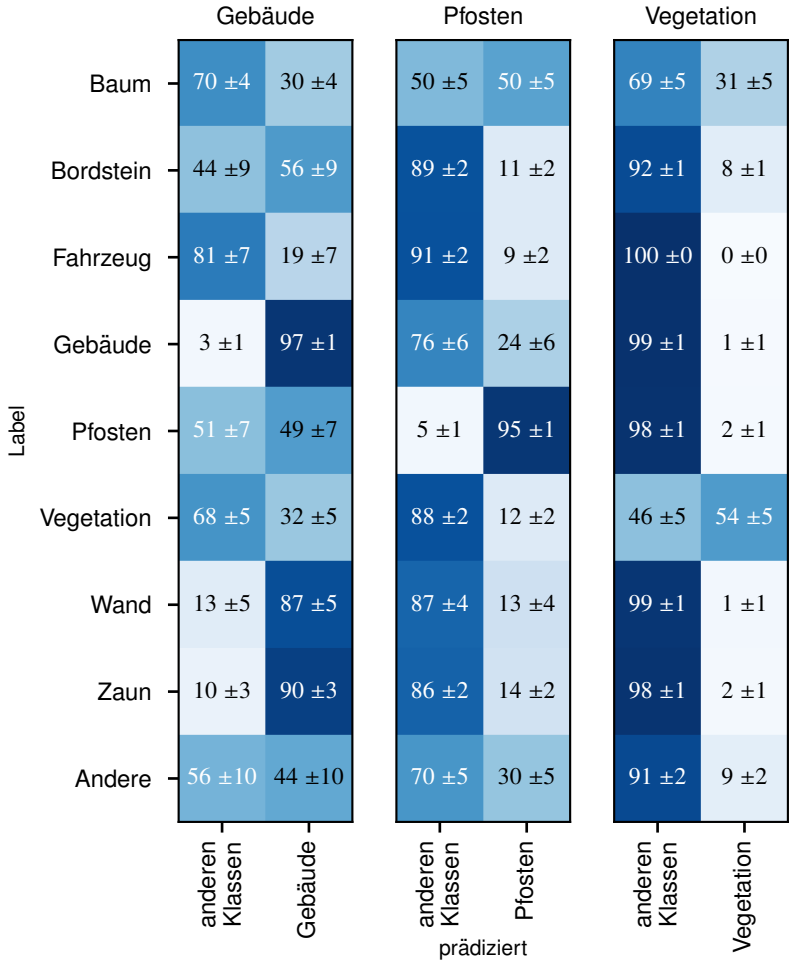


Abbildung 5.8: Erweiterte Konfusionsmatrix: Klassifikationsergebnis versus aller gelabelter Klassen.

mehr Ähnlichkeiten zu einem Busch hat, als zu einem Baum mit einem alleinstehenden Stamm. Die Klasse *Baum* hat zudem noch die Vertauschung mit der Klasse *Pfosten*, durch eine geeignete Darstellung des RCS-Wertes lässt sich dies jedoch in vielen Fällen auflösen.

Die Klasse *Bordstein* zeigt hauptsächlich Verwechslungen mit der Klasse *Wand*. Da diese jedoch in einer eigenen Klassengruppe zusammengefasst wurde und der Bordstein für die Fahraufgabe bedeutend ist, wurde dieser als Klasse erhalten.

Die Klasse *Andere* stellt an sich schon eine Sammelklasse dar. Die Konfusionsmatrix zeigt eine Verwechselungsgefahr mit vielen Klassen, besonders mit den Klassen

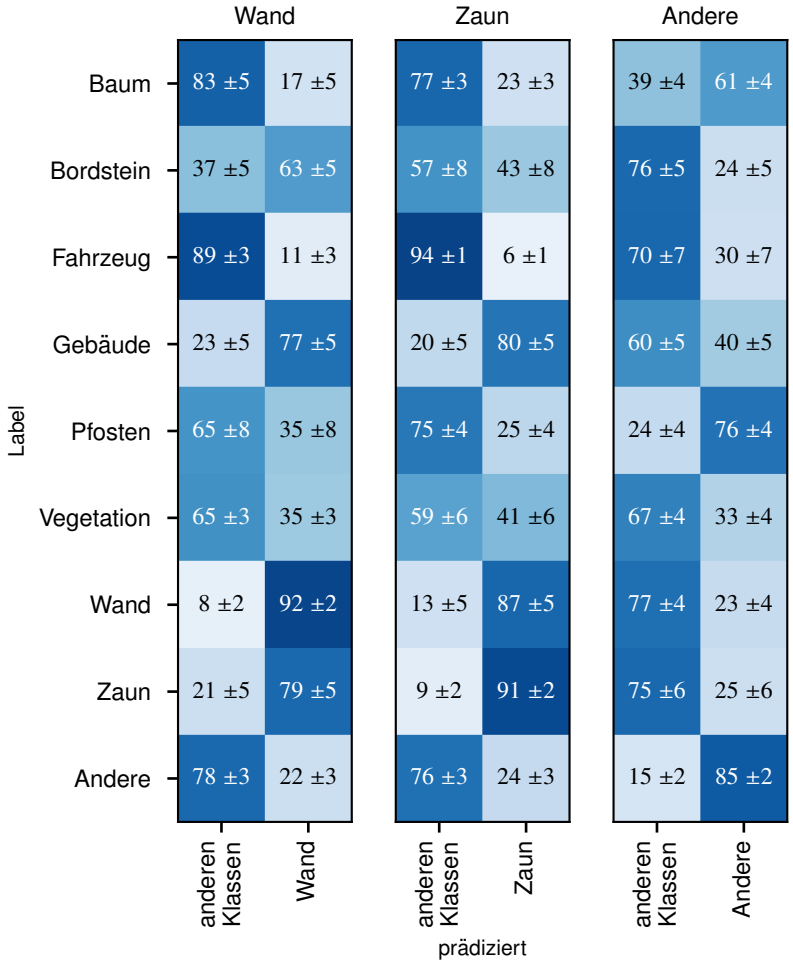


Abbildung 5.9: Erweiterte Konfusionsmatrix: Klassifikationsergebnis gegenüber gelabelten Klassen.

Pfosten und *Baum*. Hier sollten die Strukturen weiter untersucht werden, die eine Verwechslung verursachen. Das Beste wäre es, das Vorkommen dieser Klasse zu minimieren, indem geeignete Kategorien beim Labeln eingeführt werden. Ein Zusammenfassen mit einer anderen Klasse ergibt keinen Sinn, daher muss sie trotz der schlechten Performance als Einzelklasse bestehen bleiben.

Die Zusammenfassung der Klassen ist in Tabelle 5.4 zu finden. Alle weiteren Experimente werden mit dieser Klasseneinteilung durchgeführt.

Tabelle 5.4: Neue Labelklassen

Neue Klasse	Mitgliedsklassen
Fahrzeug	*Fahrzeug*
Gebäude	*Gebäude*, *Zaun*, *Wand*
Bordstein	*Bordstein*
Pfosten	*Pfosten*
Vegetation	*Baum*, *Vegetation*
Andere	*Andere*

5.5.2 Vortrainierte Faltungsnetze

Um den Nutzen, von auf Bilddaten vortrainierten Netzwerke zu analysieren, wurde ein GoogleNet verwendet, dass mithilfe des ImageNet Datensatzes vortrainiert wurde. Diese hat ursprünglich eine Eingangsgröße von 299×299 Pixeln, mit 3 Kanälen (RGB) und einer Größe der Ausgangsdaten von 1000 Klassen.

Für das Experiment wurde die Eingangsgröße auf 140×140 Pixel und 2 Kanäle reduziert. Als Kanäle wurden die Belegungskarten und die Amplitude-Maximum Karte verwendet. Die Eingangsschicht wurde dabei um einen Kanal reduziert und die Gewichte des Kanals verworfen.

Die Anzahl der Ausgangskanäle wurde auf 6, der Anzahl der verwendeten Klassen, reduziert. Diese letzte Schicht wurde in allen Fällen neu initialisiert.

Um den Einfluss eines auf Bilddaten vortrainierten Netzwerkes einschätzen zu können, wurden die Faltungsnetzwerke teilweise im vortrainierten Zustand, teilweise zufällig initialisiert und mit Radardaten nachtrainiert. Es wurden folgende Vorgehensweisen untersucht.

1. Das Training vom zufällig initialisierten Stadium
2. Das Training vom vortrainierten Stadium, bei dem nur die Eingangs- und die Ausgangsschicht (I/O) nachtrainiert wird.
3. Das Training vom vortrainierten Stadium, bei dem das gesamte Netzwerk nachtrainiert wird.

4. Das Training vom vortrainierten Stadium, es wird dabei erst Eingangs- und Ausgangsschicht trainiert, danach wird das gesamte Netzwerk nachtrainiert.

Das Training verlief jeweils, über 2000 Epochen mit 500 Minibatches pro Epoche, bei einer Größe des Minibatches von 16 Beispielen. Für das 4. Experiment wurde ein Klassifikator aus dem 2. Experiment anhand des Validationsdatensatz ausgesucht, und dann auf diesem nachtrainiert. Hierdurch wurde dieser Klassifikator deutlich länger auf den Radardaten trainiert.

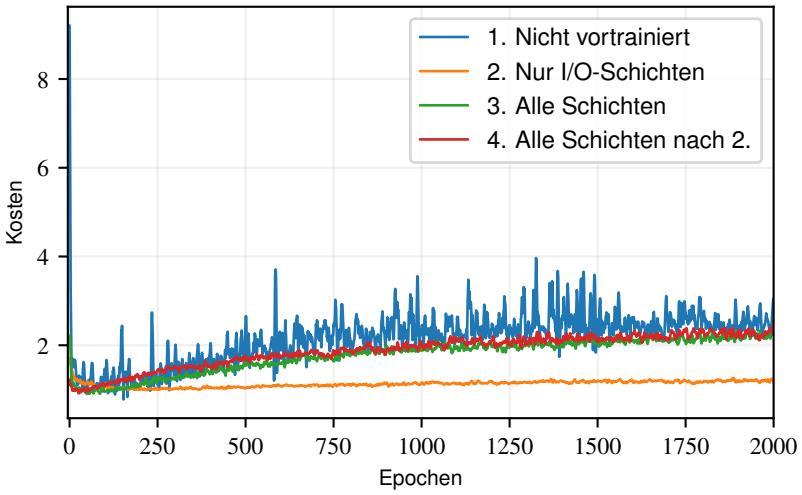


Abbildung 5.10: Verlauf der Kostenfunktion auf dem Validationsdatensatz.

Abbildung 5.10 und Abbildung 5.11 zeigen den Trainingsverlauf mithilfe des Validationsdatensatzes und Abbildung 5.12 und Abbildung 5.13 zeigen den Verlauf mithilfe der Trainingsdaten.

Wenn man auf die Auswertung der Trainingsdaten blickt, zeigt sich, dass jedes Netzwerk in der Lage ist die Klassifikation der gegebenen Beispiele zu erlernen. Hierbei erreichen alle Trainingsarten nahezu die gleichen Resultate auf dem Trainingsdatensatz. Lediglich das Netzwerk, bei dem nur die I/O-Schichten trainiert werden zeigt, dass die Kapazität dieser Schichten nicht ausreicht, die Problemstellung in gleichem Umfang zu erlernen (siehe Abb. 5.13). Hierbei ist zu sehen, dass die Kostenfunktion und die Genauigkeit sich jeweils ungefähr reziprok verhalten. Dies bestätigt, dass die Kostenfunktion das Klassifikationsergebnis gut abbildet.

Beim Training zeigt sich, dass das vorinitialisierte Netzwerk schneller trainiert wird, als das zufällig initialisierte. Auch sieht man, dass Experiment 4 schneller als Experiment 2 trainiert, da hierbei die I/O-Schichten vortrainiert sind. Dies deutet darauf hin, dass die kombinierten Merkmalsextraktoren aus den trainierten I/O-Schichten

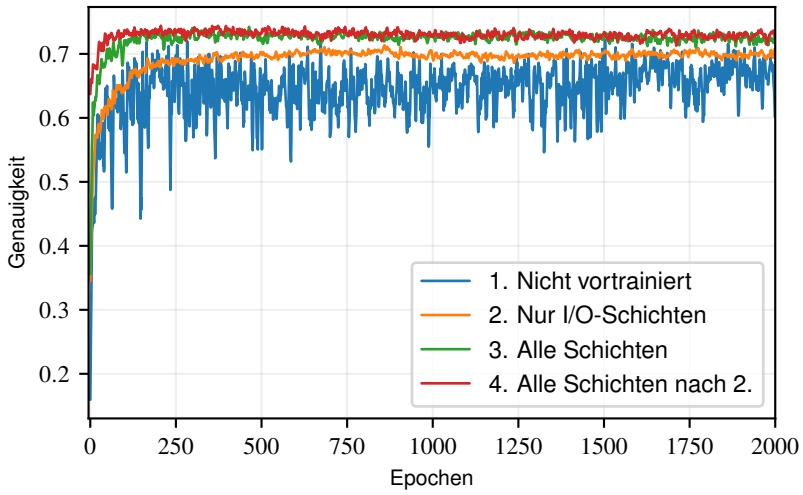


Abbildung 5.11: Verlauf der Genauigkeit auf dem Validationsdatensatz.

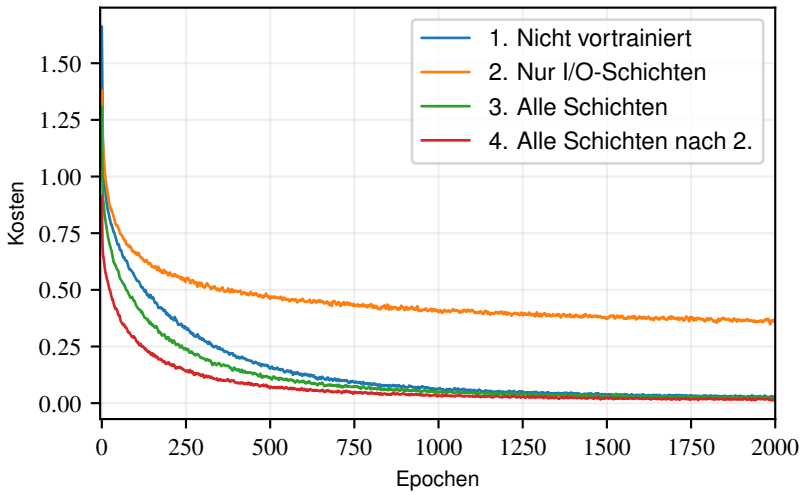


Abbildung 5.12: Verlauf der Kostenfunktion auf dem Trainings-Datensatz.

und den vortrainierten Schichten ein lokales Optimum repräsentiert, dass sich gut für das weitere Training eignet.

Viel interessanter ist die Generalisierungsfähigkeit des Netzwerkes. Hierzu kann die Betrachtung der Ergebnisse des Validationsdatensatzes in Abbildungen 5.10 und 5.11 helfen.

Als Erstes fällt auf, dass sowohl die Kostenfunktion als auch die Genauigkeit im

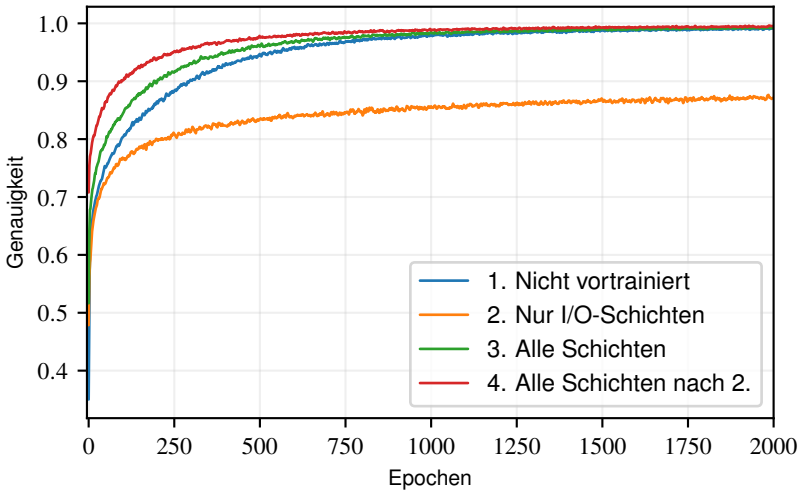


Abbildung 5.13: Verlauf der Genauigkeit auf dem Trainings-Datensatz.

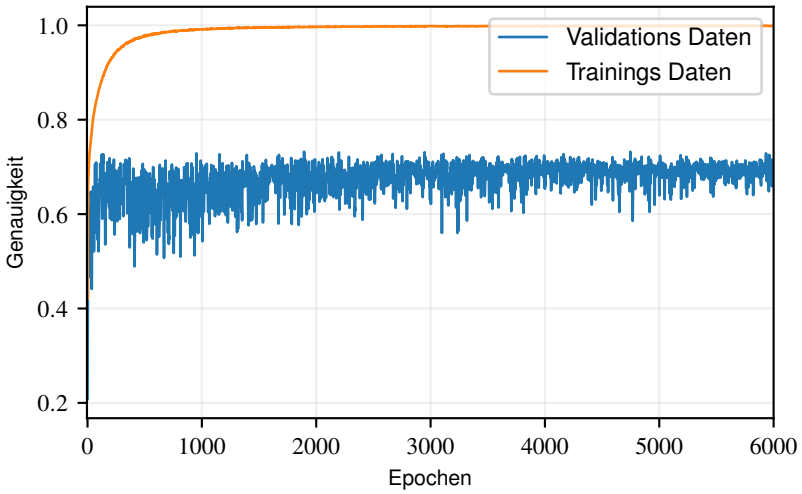
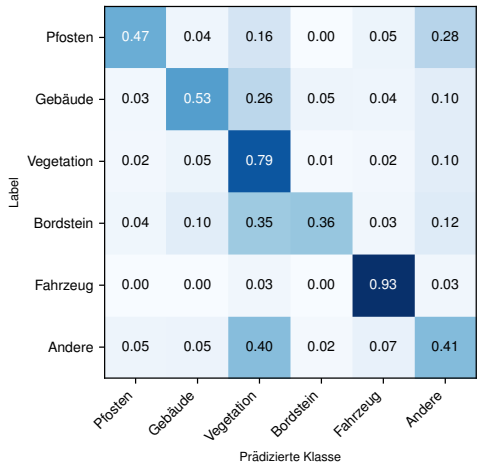
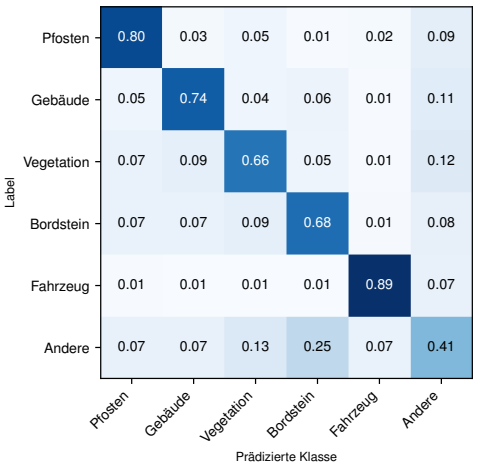


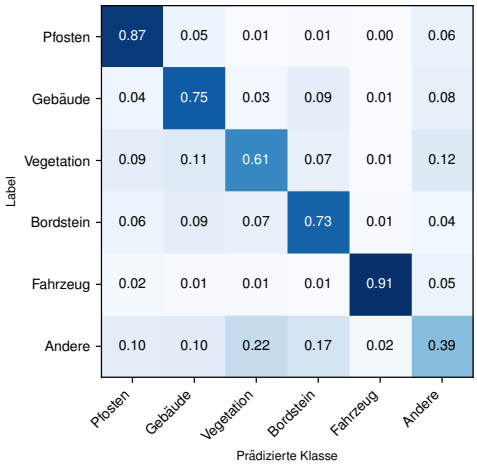
Abbildung 5.14: Verlauf der Genauigkeit auf dem Trainings- und Validationsdatensatz für ein nicht vortrainiertes Netzwerk. Die Anzahl der Epochen wurde dabei auf das Dreifache erhöht.



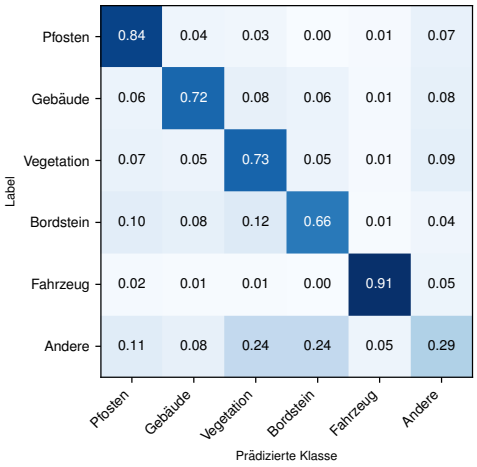
(a) Experiment 1: Nicht vortrainiert.



(b) Experiment 2: Vortrainiert, nur I/O Schichten.



(c) Experiment 2: Vortrainiert, alle Schichten.



(d) Experiment 4: Vortrainiert, alle Schichten nach Experiment 2.

Abbildung 5.15: Konfusionsmatrizen der verschiedenen Experimente auf dem Testdatensatz.

Falle der zufälligen Initialisierung stark schwankt. Dies deutet darauf hin, dass die gelernten Merkmalsextraktoren nicht gut generalisieren und stark von den jeweiligen Beispielen im Validationsdatensatz abhängig sind.

An der Kostenfunktion ist zu sehen, dass die Netzwerke, zur Überanpassung neigen, da trotz einer Verbesserung im Bereich der Trainingsdaten, die Kostenfunktion auf

den Validationsdaten schlechter wird. Dies ist jedoch noch in einem Maße, dass es sich nicht auf das Klassifikationsergebnis auswirkt. Hierbei wird wiederum deutlich, dass Kostenfunktion und Genauigkeit nicht immer korrelieren. Lediglich bei Experiment 2, sind keine Ansätze des Overfittings zu erkennen. Was wiederum der geringen Kapazität des trainierbaren Teils des Netzwerkes zuzurechnen ist.

Neben den starken Schwankungen zeigt das zufällig initialisierte Netzwerk auch die schlechteste Gesamtperformance auf dem Testdatensatz. Es fällt dabei sogar hinter den Klassifikator zurück, bei dem nur die I/O-Schichten trainiert worden sind. Dies lässt wiederum darauf schließen, dass die für Bilder gelernten Merkmalsextraktoren sich für die neue Problemstellung eignen, auch wenn sich das Erscheinungsbild einer Radar-Merkmalskarte erheblich von einem Foto unterscheidet. Beim zufällig initialisierten Training tendiert das Netzwerk dazu, weniger robuste Merkmalsextraktoren auf dem Trainingsdatensatz zu lernen, welche schlechter generalisieren. Ein weiteres Indiz dafür ist die starke Schwankung in den Validationsdaten über die Zeit, da nach jeder Epoche andere Validationsdaten verwendet werden.

Um zu zeigen, dass dieser Effekt nicht nur darauf beruht, dass vortrainierte Netzwerke länger trainiert wurden, wenn auch auf anderen Daten, wurde ein Training durchgeführt, bei dem die Anzahl der Trainingsepochen auf das Dreifache erhöht wurden. Das Ergebnis ist in Abbildung 5.14 dargestellt. Hierbei zeigt sich, dass die durchschnittliche Genauigkeit, in geringem Umfang weiter steigt, aber immer noch eine starke Varianz zwischen den einzelnen Epochen herrscht. Es ist also davon auszugehen, dass die Trainingsdaten des vortrainierten Netzwerkes und nicht nur das längere Training dem Klassifikator zu robusteren Merkmalsextraktoren verhelfen.

Tabelle 5.5: Resultate auf dem Test Datensatz.

	Mikro Genauigkeit	Makro Genauigkeit
1.) Nicht vortrainiert	0,673	0,583
2.) Nur I/O Schichten	0,700	0,698
3.) Alle Schichten	0,767	0,730
4.) Alle Schichten nach 2.	0,770	0,732

In diesem Experiment wurde gezeigt, dass Merkmalsextraktoren, die auf Bilddaten trainiert wurden, sich auch für Radardaten gut eignen. Damit ist dies eine gute Alternative zur zufälligen Initialisierung des Klassifikators. Besonders gut kann man den Effekt nutzen, wenn die angepassten Netzwerkschichten zuerst trainiert und danach alle Netzwerkschichten nachtrainiert werden. Gezeigt wurde dies mithilfe eines

vortrainierten *GoogleNet*. Weitere in diesem Kapitel nicht dargestellte Experimente auf dem *MobileNet* [How+17] bestätigen dieses Ergebnis.

5.5.3 Radar-Merkmalskarten

Anders als bei Bildern handelt es sich bei Radar-Merkmalskarten um schon vorverarbeitete Daten. Eine Klassifizierung von statischen Zielen auf Punktwolken, wie sie z. B. bei [Sch+18b] für dynamische Objekte eingesetzt werden, hat sich bei den eingesetzten Sensoren als nicht erfolgreich erwiesen.

In Abschnitt 2.4 wurden daher verschiedene Merkmalskarten zur Klassifikation von Objekten vorgestellt. In diesem Abschnitt soll nun untersucht werden, wie gut sich die einzelnen Karten zur Klassifikation eignen.

Hierzu wurden für den Cluster-Datensatz verschiedene Merkmalskarten und Merkmalskarten-Kombinationen untersucht. Im Falle mehrerer Karten werden diese anhand der Kanäle gestapelt. Dies ist möglich, da jede Karte dieselbe Auflösung besitzt und die Zellen im Raum aufeinander ausgerichtet sind. Bei der RCS-Histogramm-Karte, die selbst mehrere Kanäle besitzt, wird diese Dimension auch zum Stapeln genutzt.

Im Folgenden wird aufgelistet, welche Kartenkombinationen untersucht werden.

- Belegungskarte
- Belegungskarte & RCS-Histogramm-Karte
- Belegungskarte & RCS-Maximum-Karte
- Belegungskarte & RCS-Minimum-Karte
- Belegungskarte & RCS-Mittelwert-Karte
- RCS-Maximum-Karte

Zur Auswertung wurde das *GoogleNet* eingesetzt. Die Netzwerke wurden dabei zufällig initialisiert. Auf die Verwendung von auf Bildern vortrainierten Netzwerken wurde verzichtet, da die Gefahr besteht, dass die gelernten Merkmalsextraktoren sich für einzelne Karten besser eignen, als für andere. Damit würde das Ergebnis dieses Experimentes verfälscht.

Die Klassifikatoren werden über 500 Epochen mit jeweils 200 Mini-Batches pro Epoche trainiert. Die Größe des Batches besteht aus 32 Beispielen. Nach jeder Epoche wird das Netzwerk auf 6400 zufällig ausgewählten Beispielen des Validationsdatensatzes evaluiert.

Final wurde jede Kombination mittels 2 Trainingsdurchgängen untersucht. Aufgrund der Rechenzeit und der starken Korrelation der beiden Ergebnisse wurde auf eine umfangreichere Auswertung verzichtet.

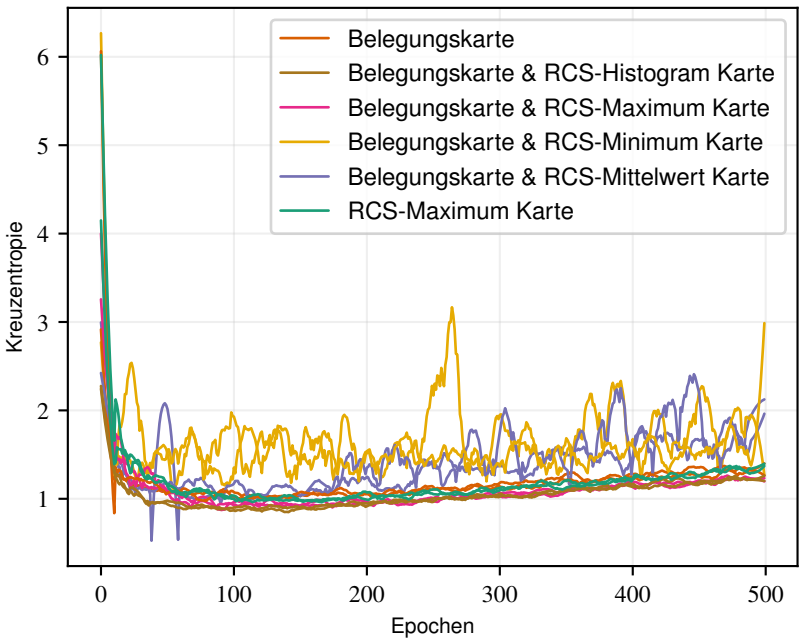


Abbildung 5.16: Verlauf der Kostenfunktion während des Trainings (Validationsdaten).

Die Abbildungen 5.16 bis 5.19 zeigen jeweils den Verlauf der Genauigkeit und der Kostenfunktion während des Trainings auf dem Validations- und auf dem Trainingsdatensatz.

Während des Trainings ist bei allen Kartenkombinationen in der Kostenfunktion des Validationsdatensatzes eine leichte Überanpassung zu beobachten, diese wirkt sich jedoch noch nicht auf die Genauigkeit des Klassifikators aus.

Betrachtet man den Fehler und die Genauigkeit auf dem Validationsdatensatz, so zeigt sich, dass die Belegungskarte mit der RCS-Histogramm-Karte die beste Datendarstellung ist, gefolgt von der Belegungskarte mit der RCS-Maximum Karte.

Die RCS-Maximum-Karte und die Belegungskarte setzen diese Reihenfolge fort. Hierbei zeigt sich, dass die RCS-Maximum-Karte, mehr Aussagekraft besitzt als die Belegungskarte. Dieses Ergebnis ist erstaunlich, da ein Mensch dies in der Regel genau umgekehrt beurteilt, weil in der RCS-Maximum-Karte die Formen mehr verschwommen sind. Wenn man jedoch den Trainingsdatensatz betrachtet, sind die

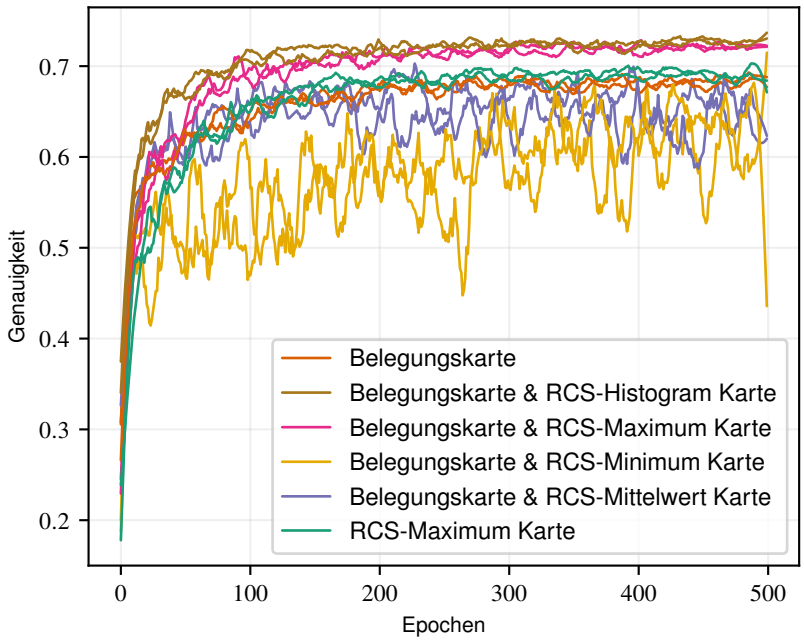


Abbildung 5.17: Verlauf der Genauigkeit während des Trainings (Validationsdaten).

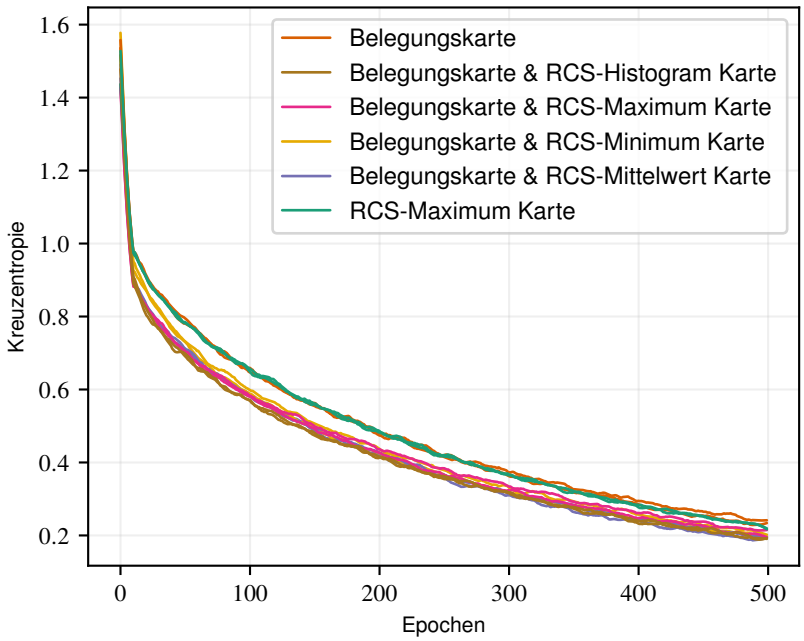


Abbildung 5.18: Verlauf der Kostenfunktion während des Trainings (Trainingsdaten).

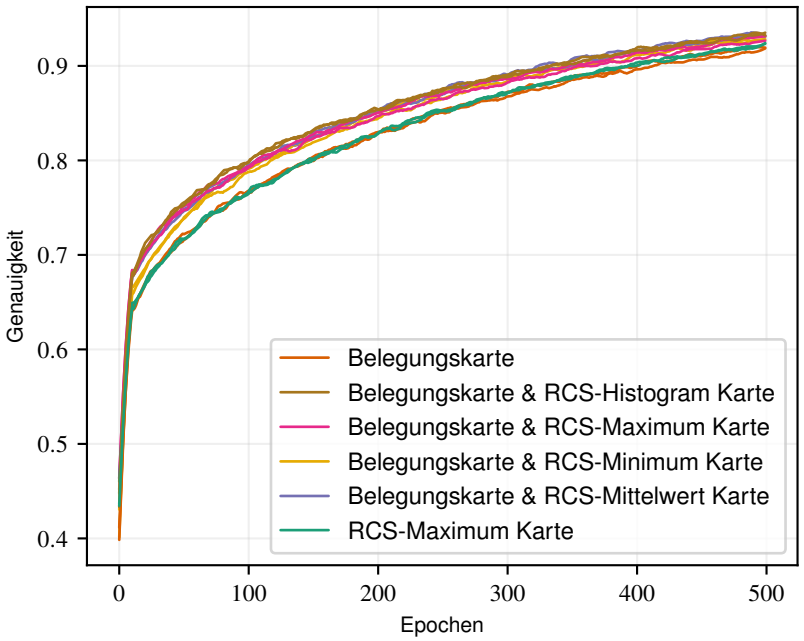
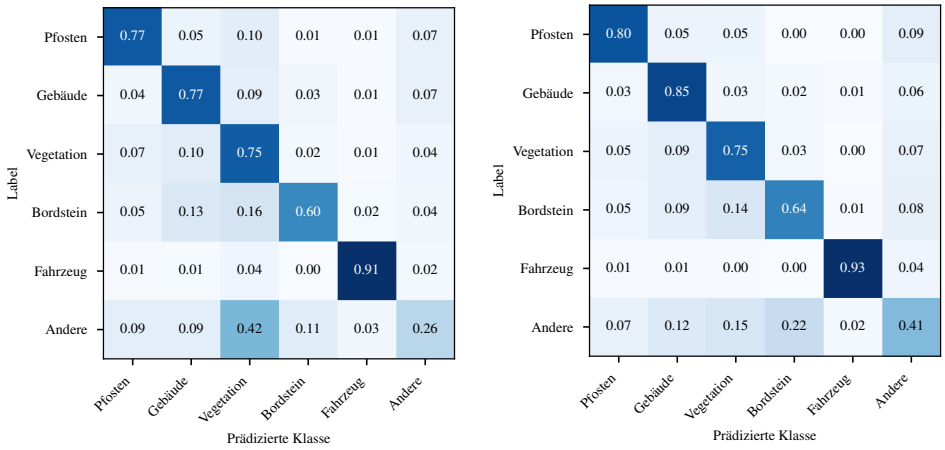


Abbildung 5.19: Verlauf der Genauigkeit während des Trainings (Trainingsdaten).



(a) Belegungskarte

(b) Belegungskarte & RCS-Histogramm-Karte.

Abbildung 5.20: Konfusionsmatrizen der verschiedenen Experimente auf dem Testdatensatz.

Ergebnisse dieser beiden Karten die schlechtesten. Daraus lässt sich ableiten, dass die Merkmale, die aus diesen Karten berechnet werden, schwierig zu lernen sind, aber dadurch ihre Stärken in der Generalisierungsfähigkeit haben.

Am schlechtesten generalisiert die Kombination aus Belegungskarte und RCS-Minimum-Karte, sowie die Kombination aus Belegungskarte und RCS-Mittelwert-Karte. Dies bestätigt ein Blick auf die Trainingsdaten, die dort deutlich besser als beispielsweise die RCS-Maximum-Karte sind. Ein weiteres Indiz für die schlechte Generalisierungsfähigkeit sind die starken Schwankungen des Validationsdatensatzes.

Tabelle 5.6 fasst noch einmal die Ergebnisse auf dem Testdatensatz zusammen.

In Abb. 5.20 sind die Konfusionsmatrizen der Belegungskarte und der RCS-Histogramm-Karte ersichtlich.

Tabelle 5.6: Makro-Genauigkeit auf dem Testdatensatz

	Makro-Genauigkeit
Belegungskarte	0.67
Belegungskarte & RCS-Histogramm-Karte	0.76
Belegungskarte & RCS-Maximum-Karte	0.74
Belegungskarte & RCS-Minimum-Karte	0.48
Belegungskarte & RCS-Mittelwert-Karte	0.40
RCS-Maximum-Karte	0.71

Natürlich könnten diesbezüglich weitere Experimente durchgeführt werden. So ließen sich verschiedene Versionen der RCS-Histogramm-Karte untersuchen, bei dem die Schwellwerte der Klassen angepasst werden. Auch könnten noch weitere Kombinationen untersucht werden.

5.5.4 Vergleich Augmentierungstechniken

Wie in Abschnitt 5.3.2.1 beschrieben, werden zwei Experimente durchgeführt. Das eine wird zum Bestimmen der Anfangspopulation, das andere zum eigentlichen Training eines Klassifikators mittels Augmentierungstechniken eingesetzt. Für beide Experimente wird als Klassifikator ein zufällig initialisiertes *GoogleNet* eingesetzt.

Tabelle 5.7: Parameter des genetischen Algorithmus

Parameter	Experiment Anfangspopulation	Experiment Augmentierung
η	15	15
$p_{Survival}$	0.7	0.7
n_{best}	1	1
n_{new}	1	1
$n_{best,C}$	n.a.	3
$n_{mutation}$	7	7
$\sigma_{mutation}$	1	1

Der Rechner, auf dem das Experiment durchgeführt wurde, besitzt 4 GPUs, daher wurde aus Gründen der Laufzeitoptimierung die Population η so gewählt, dass sie zusammen mit dem Referenz-Klassifikator ein vielfaches von 4 ergibt.

Pro Generation werden 6400 Beispiele aus dem Trainingsdatensatz gezogen. Dieser wird zum Training aller Individuen einer Generation eingesetzt. Dasselbe geschieht mit dem Validations- und Testdatensatz mit je 3200 Beispielen. Diese Teildatensätze werden erzeugt, um die unterschiedliche Entwicklung eines Individuums aufgrund einer anderen Kombination von Beispielen zu verhindern.

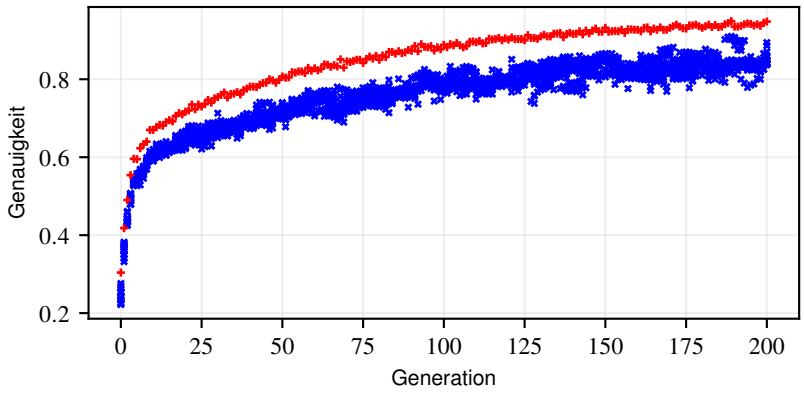
Der Klassifikator jedes Individuums wird mithilfe von Minibatches von 32 Beispielen trainiert. Jeder Klassifikator wird mit 200 dieser Minitbatches trainiert, sodass der Teildatensatz insgesamt nur einmal durchlaufen wird.

Die Evaluation auf den Testdaten während des Trainings, wurde nur für die später zu sehenden Visualisierungen, aber nicht für das Bewerten des Klassifikators verwendet.

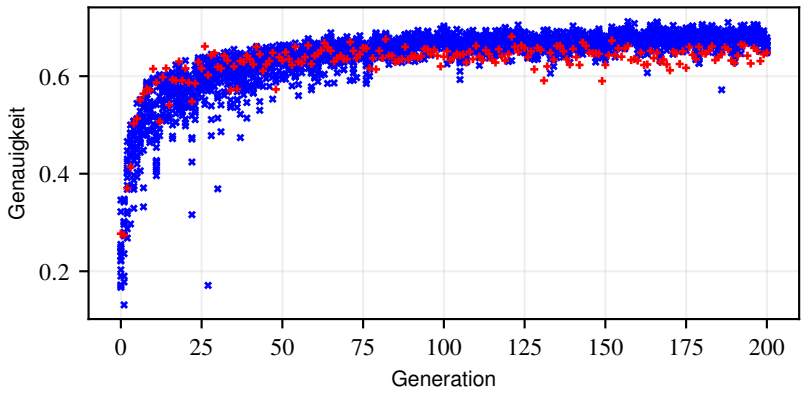
Bei der Selektion werden die besten 70 % der Individuen zur Rekombination verwendet.

Als Eingangsdaten wurde jeweils die Belegungskarte genutzt. Andere Kartenkombination konnten aufgrund der Rechenzeit nicht ausgewertet werden.

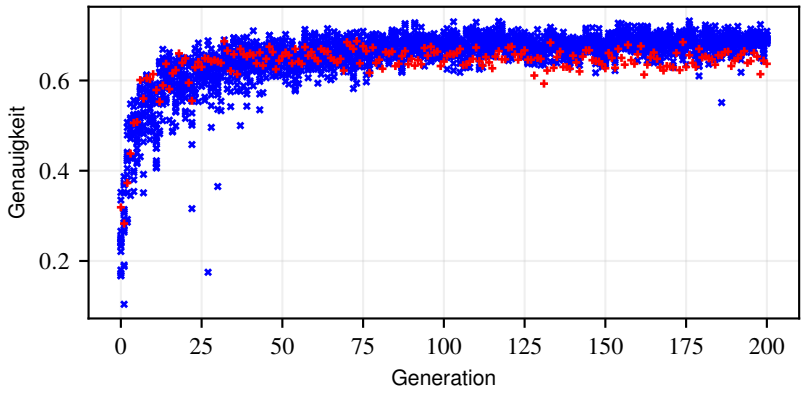
In Abbildung 5.21 ist der Trainingsverlauf des finalen Trainings zu sehen. In Abb. 5.21a ist die Genauigkeit der Daten während des Trainings zu sehen. Dabei ist es



(a) Trainingdaten



(b) Validationsdaten



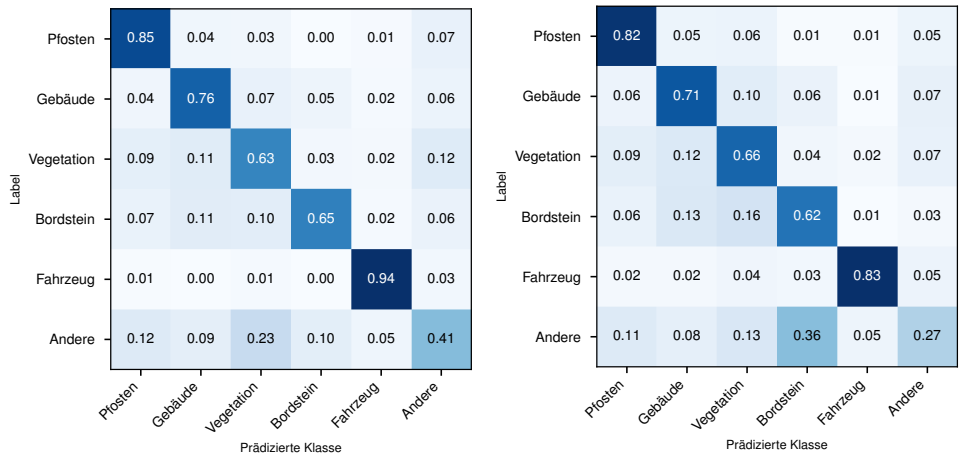
(c) Testdaten

Abbildung 5.21: Zeitlicher Verlauf der Klassifikationsperformance der einzelnen Individuen. In Rot wird der Referenzklassifikator, der ohne Augmentierung trainiert wurde, eingezeichnet.

nicht überraschend, dass der Referenzklassifikator ein besseres Ergebnis erzielt als die augmentierten Individuen.

Die Validationsdaten in Abb. 5.21b zeigen, dass im Verlauf des Trainings die augmentierten Individuen ein besseres Ergebnis erzielen. Während am Anfang des Trainings Augmentierung kaum eine Rolle spielt, macht es sich im weiteren Trainingsverlauf bemerkbar.

Da anhand der Validationsdaten die einzelnen Individuen optimiert werden, muss für die Auswertung der Testdatensatz (Abb. 5.21c) herangezogen werden. Hier bestätigt sich das Verhalten des Validationsdatensatzes. Die Augmentierung gewinnt während des Trainings eine immer größere Bedeutung. Dies ist zum einen dadurch zu erklären, dass am Anfang des Trainings das Neuronale Netzwerk noch genügend aus nicht augmentierten Beispielen lernen kann, zum anderen verbessern sich, durch den genetischen Algorithmus, die Parameter der Augmentierung.



(a) Klassifikator mit Augmentierung.

(b) Referenz Klassifikator ohne Augmentierung.

Abbildung 5.22: Konfusionsmatrizen auf dem Testdatensatz.

Zum Abschluss sind in Abb. 5.22 die Konfusionsmatrizen des Referenz Klassifikators und des besten Individuums das mit Augmentierung trainiert wurde, gegenübergestellt. Diese zeigen eine allgemeine Verbesserung des Klassifikators durch die Augmentierung. Lediglich bei der Klasse *Vegetation* zeigt sich eine leichte Verschlechterung. Ob dies an dem Charakter der Klasse liegt, welche in der Regel keine klaren Strukturen besitzt oder ob es sich um ein zufälliges Phänomen handelt, lässt sich aus den Experimenten nicht mit Sicherheit sagen.

Dieses Experiment zeigt, dass durch Augmentierung das Klassifikationsergebnis verbessert werden kann. Es stellt jedoch nur das Potenzial der Augmentierungstechniken dar. Als weitere Schritt sollte das Experiment über eine längere Zeit mit mehr Individuen durchgeführt und die Parameter des genetischen Algorithmus optimiert werden. Ferner stehen noch zahlreiche andere Augmentierungstechniken zur Verfügung. So können beispielsweise direkt die Messungen des Radar Sensors manipuliert werden. Und damit die Belegungskarten neu aufgebaut werden. Damit könnten beispielsweise Verdeckungen, eine höhere oder geringere Abtastrate und Sensorausfälle simuliert werden.

Um dies in einem realistischen Zeitrahmen durchführen zu können, ist jedoch eine erhebliche Rechenleistung und eine effiziente Implementierung der Algorithmen nötig.

5.5.5 Klassische Klassifikationsverfahren

In diesem Experiment wird das Klassifizieren mit klassischen, vom Menschen entworfenen Merkmalsextraktoren untersucht. Diese werden, wie in Abschnitt 5.2 beschrieben, aus dem Cluster-Datensatz extrahiert. Damit wird anschließend ein Random-Forest Klassifikator trainiert und ausgewertet. Als Klasseneinteilung werden die im 5.5.1 Kapitel beschriebenen 6 Klassen verwendet.

Tabelle 5.8: Hyperparameter Suchbereich.

Parameter	Wertebereich
max. Tiefe des Baums	30, 50, unbeschränkt
max. Anzahl Merkmale	sqrt, log2, 1, 3, 7
min. Anzahl Beispiele	2, 5, 10
Bootstrapping	ja, nein
Aufteilungskriterium	Gini, Entropy
min. Anzahl pro Blatt	1, 3, 5, 10, 15, 30
min. Abnahme der Unreinheit	0, 0.001, 0.003, 0.1, 0.03, 0.1
Gewichten der Klassen	ja, nein, pro Baum
Anzahl der Bäume	10, 30, 100, 150, 300

Zunächst wurden mithilfe des Trainings- und Validationsdatensatzes die Parameter für den *Random-Forest* Klassifikator gesucht. Die untersuchten Werte sind in Tabelle 5.8 zu finden. Da es bei einer Trainingszeit von bis zu 3 Stunden pro *Random-Forest* zu lange dauert, dies mittels eines Suchraster zu optimieren, wurde eine randomisierte Suche verwendet, welche zufällig aus dem Parameterraum auswählt und diese anhand des Makro-F1 Gütemaß bewertet.

Um die Parametersuche weiterhin zu beschleunigen, wurden dafür lediglich die Hälfte der Trainingsdaten verwendet. Insgesamt wurden 2000 zufällig gewählte Parameter untersucht.

Tabelle 5.9: Hyperparameter für finalen Klassifikator.

Parameter	Wertebereich
max. Tiefe des Baums	50
max. Anzahl Merkmale	sqrt
min. Anzahl Beispiele	5
Bootstrapping	nein
Aufteilungskriterium	Gini
min. Anzahl pro Blatt	5
min. Abnahme der Unreinheit	0
Gewichten der Klassen	pro Baum
Anzahl der Bäume	150

Das Ergebnis dieser Parametersuche ist in Tabelle 5.9 zu finden. Da die quantitative Darstellung aller Ergebnisse zu umfangreich ist, sollen diese hier nur qualitativ diskutiert werden. Dabei werden hauptsächlich die besten 100 Parameter betrachtet, die eine Performance von 0.62 bis 0.57 erreichen.

Eines der klarsten Ergebnisse ist die minimale Abnahme der Unreinheit, bei dem ein Knoten noch geteilt wird. Diese liegt, unabhängig von den weiteren Parametern, bei mehr als den ersten 100 Ergebnissen bei 0.

Das Aufteilungskriterium *Gini Impurity* oder *Entropie* hält sich ziemlich die Waage. Das Gewichten der Klasse wird überwiegend als wichtig angesehen. Ob diese Gewichte pro Baum oder für den gesamten Random Forest bestimmt werden, scheint keine Rolle zu spielen. Bei den Parametern der vorderen 10 Plätze spielt dies ebenso

keine Rolle, da hier für den Datensatz in der Regel kein Bootstrapping eingesetzt wird. Die minimale Anzahl der Beispiele pro Blatt und minimale Anzahl der Beispiele die für eine Aufteilung benötigt werden, scheinen keinen großen Einfluss zu haben. Die Anzahl der Bäume für gute Ergebnisse liegt bei etwa 100 bis 150.

Anschließend wurde der beste Parametersatz verwendet, um einen Klassifikator mit dem vollen Datensatz zu trainieren. Dieses Experiment wurde 10-mal wiederholt. Um zu vermeiden, dass durch eine zufällige Aufsplittung, Korrelationen entstehen, wurde auf eine sonst übliche Kreuzvalidierung verzichtet. Sofern die extrahierten Merkmale zur Verfügung stehen, dauert das Training aus einem Intel® Xeon® E5-2687W v4 mit 3 GHz etwa 2.5 Stunden.

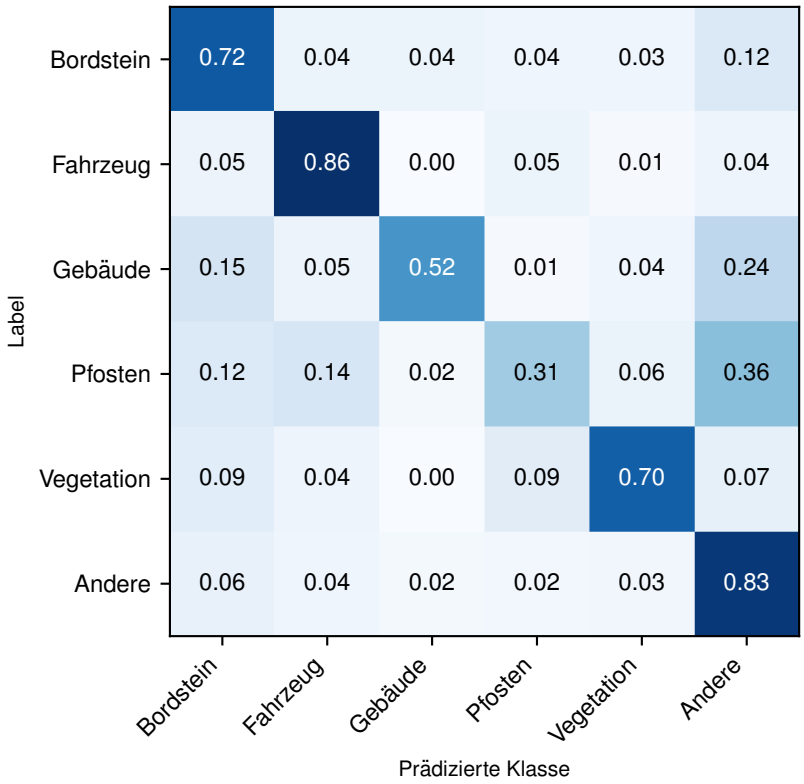


Abbildung 5.23: Konfusionsmatrix des *Random-Forest* Klassifikator auf dem Testdatensatz.

Tabelle 5.10 zeigt die Ergebnisse von drei ausgewählten Klassifikatoren auf dem Test- und Trainingsdatensatz. Hierbei fällt auf, dass trotz der zufälligen Aufspaltungen sehr ähnliche Klassifikationsergebnisse erzielt werden, dies gilt auch für die sieben

Tabelle 5.10: Ergebnisse des *Random-Forest* Klassifikator.

Klassifikator	A	B	C
Mikro-Genauigkeit Test	0.7069	0.6999	0.7030
Makro-Genauigkeit Test	0.6554	0.6489	0.6513
Mikro-Genauigkeit Training	0.9973	0.9974	0.9973
Makro-Genauigkeit Training	0.9984	0.9984	0.9984

hier nicht dargestellten Klassifikatoren. Die fast hundertprozentigen Ergebnisse auf dem Trainingsdatensatz weißen darauf hin, dass die Kapazität des Klassifikators groß genug für die Problemstellung ist.

Die Generalisierungsfähigkeit fällt erwartungsgemäß deutlich schlechter aus. Hierbei erzielt der Klassifikator eine Makro-Genauigkeit von etwa 65 %, wenn alle Klassen ausgeglichen sind. Auffällig ist jedoch, dass die Mikro-Genauigkeit mit 70 % besser ist, obwohl der Klassifikator auf gewichteten Beispielen trainiert worden ist. Es steht zu vermuten, dass die Mehrheitsklassen aufgrund ihres größeren Datensatzes besser gelernt werden können, während für unterrepräsentierte Klassen die nötige Varianz an Beispielen fehlt.

In Abbildung 5.23 ist eine Konfusionsmatrix von dem Klassifikator *A* zu sehen. Da auch hier die Matrizen sehr ähnlich sind, wurde auf weitere Darstellungen verzichtet.Hierbei ist wiederum die Klasse *Fahrzeug* diejenige, bei der die besten Resultate erzielt werden. Auffällig ist das gute Ergebnis für die Klasse *Andere*. Dies geht allerdings auf Kosten der Trefferquote für die Klassen *Pfosten*, *Gebäude* und *Bordstein*.

Der *Random-Forest* Klassifikator ermöglicht es, mithilfe der mittleren *Gini-Impurity* [Gér17], die Wichtigkeit der Merkmale zu beurteilen. In Tabelle 5.11 sind diese Merkmale, nach ihrer Wichtigkeit sortiert, dargestellt. Hierzu wurde die Wichtigkeit von Merkmalen, die aus mehreren Attributen bestehen, zusammengefasst, indem ihre Wichtigkeiten addiert wurden.

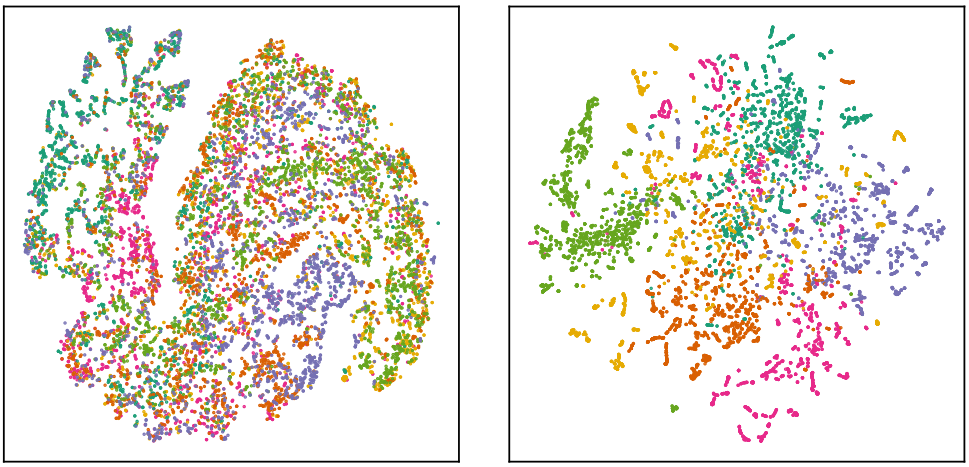
Es ist zu beobachten, dass die Gradienten der Belegungskarten welche durch das *HOG*-Merkmal ausgedrückt werden, eine große Bedeutung haben. Gefolgt von Statistiken über den RCS-Wert und einfachen Formfaktoren wie Umfang und Fläche.

Bei dieser Betrachtung sollte beachtet werden, dass bei korrelierten Merkmalen sich eines davon zufällig hervorheben kann, während das andere an Wichtigkeit verliert.

Tabelle 5.11: Wichtigkeit der Merkmale.

Merkmal	Karten	Wichtigkeit
HOG	Belegungskarte	4.0074
Histogramm	Maskierte RCS-Maximum-Karten	1.7175
Maximum	Maskierte RCS-Maximum-Karten	0.5712
Umfang konvexe Hülle	Schwellwert Karten	0.4431
Median	Maskierte RCS-Maximum-Karten	0.3616
Standardabweichnung	Maskierte RCS-Maximum-Karten	0.3561
Umfang	Schwellwert Karten	0.3494
Fläche	Schwellwert Karten	0.3422
Maximum	Maskierte Belegungskarten	0.3054
Mittelwert	Maskierte RCS-Maximum-Karten	0.2786
Breite	Schwellwert Karten	0.2234
Standardabweichnung	Maskierte Belegungskarten	0.2234
Länge	Schwellwert Karten	0.2206
Mittelwert	Maskierte Belegungskarten	0.2188
Median	Maskierte Belegungskarten	0.2036
Annular Statistic Descriptor	RCS-Maximum-Karte	0.0637
Minimum	Maskierte RCS-Maximum-Karten	0.0554
Annular Statistic Descriptor	Belegungskarte	0.0526
Histogramm	Belegungskarte	0.0025

5.5.6 Ensemble Klassifikation



(a) Klassische Merkmalsextraktoren. (b) Merkmale des CNN.

Abbildung 5.24: T-SNE Grafik der klassischen Merkmalsextraktoren und eines CNN trainiert mit Belegungskarten und RCS-Maximum-Karten. Hierbei wird der Abstand der Merkmalsvektoren der einzelnen Beispiele im 2-D Raum veranschaulicht.

Für dieses Experiment wird der *Random-Forest* Klassifikator aus Abschnitt 5.5.5 eingesetzt. Als tiefe Lernmethode wird ein CNN eingesetzt. Dieses Stammt aus dem Experiment welches in Abschnitt 5.5.3 beschrieben wurde. Dabei wurde ein Netzwerk gewählt, welches auf der Belegungskarte und der RCS-Maximum-Karte trainiert worden ist. Somit basiert der *Random-Forest* und der CNN Klassifikator auf denselben Eingangsdaten. Die Evaluation erfolgt dabei auf dem Testdatensatz.

Bevor ein Ensemble Klassifikator untersucht wird, zeigt Abbildung 5.24 zunächst einmal die Merkmale der einzelnen Klassifikatoren anhand einer t-SNE (T-distributed Stochastic Neighbor Embedding) Grafik [MH08]. Dies ist eine Methode zur Visualisierung von hochdimensionalen Daten. Diese gruppiert die Daten anhand ihrer Abstände in einen niedrigdimensionalen Raum, z. B. in einer 2D-Darstellung. Hierbei wird versucht, die Abstände der einzelnen Punkte ähnlich zu halten, sodass benachbarte Punkte im hochdimensionalen Raum auch im niedrigdimensionalen Raum benachbart sind. Aus dieser Darstellung lässt sich abschätzen wie gut ein Klassifikator diese Daten clustern und damit klassifiziern kann.

Als Eingangsdaten für diese Darstellung wurden für die klassischen Verfahren die extrahierten Merkmale der Merkmalsextraktoren des *Random-Forest* Klassifikators

genutzt. Für die tiefen Verfahren wurde der Ausgang der letzten Faltungsschicht des CNNs, vor den voll verbundenen Schichten, verwendet.

Tabelle 5.12: Ergebnisse der Ensemble Klassifikatoren

Klassifikator	Genauigkeit	Sensitivität
CNN	0.69	0.75
Random Forest	0.66	0.66
Distributed Summation	0.71	0.76
Bayesian Combination	0.71	0.76
Naive Bayes	0.73	0.77
Übereinstimmung	0.82	0.58

Für die Darstellung wurden 12 000 zufällig ausgewählte Beispiele des Testdatensatzes genutzt. Die Klassen sind dabei nach der Legende in Abb. 4.11 farblich kodiert.

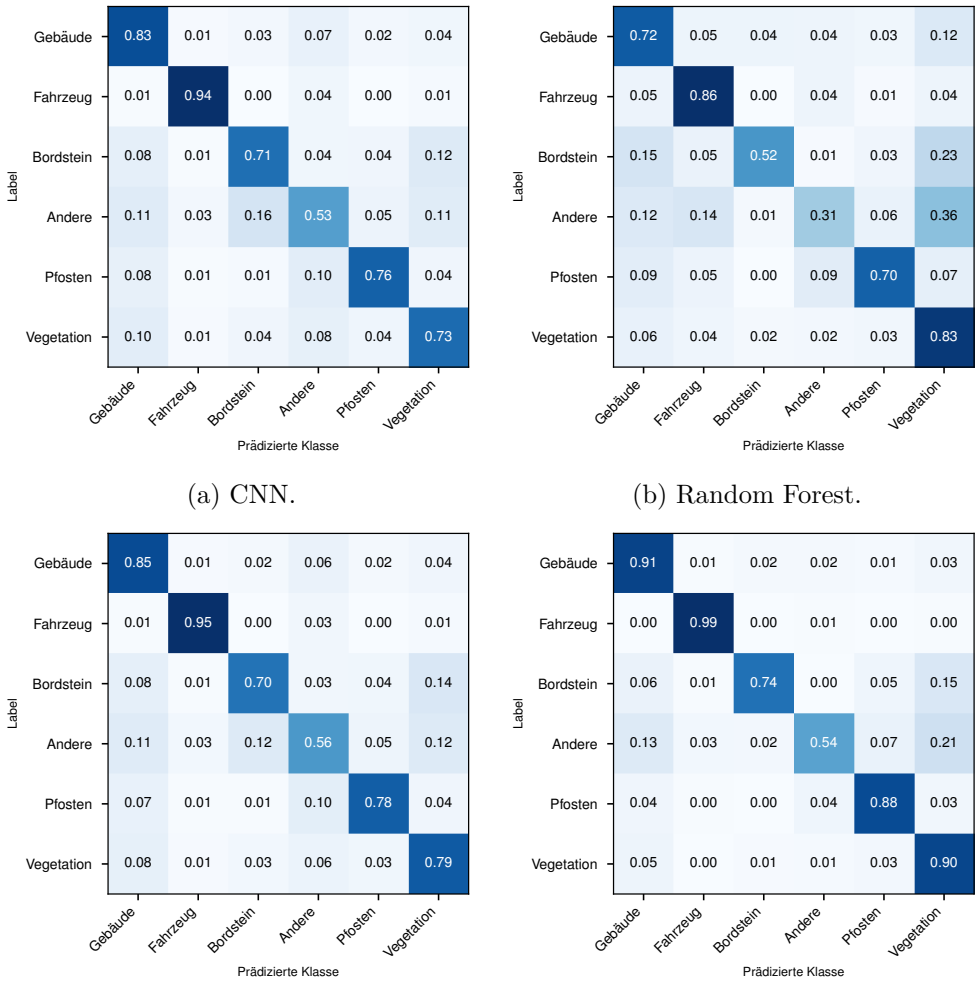
Bei den projizierten Daten zu beobachten, dass auf den extrahierten Merkmalen des CNN, eine Trennung der Klassen besser möglich ist, als auf denen der vom Experten designten Merkmalsextraktoren. Des Weiteren lässt sich bei den Merkmalen des CNN erkennen, dass die Klasse *Andere* keine eindeutige Gruppierung aufweist.

Im nächsten Schritt wurden die Klassifikatoren wie in Abschnitt 5.4 beschrieben miteinander kombiniert. Das Ergebnis des Tests ist in Tabelle 5.12 dargestellt.

Es ist zu sehen, dass durch die Kombination, ein besseres Ergebnis als bei den ursprünglichen Klassifikatoren erreicht werden kann. Der *Naive-Bayes* Ansatz erweist sich am erfolgreichsten, bei der die Klassifikatoren als stochastisch unabhängig voneinander betrachtet werden.

Auf Kosten der Sensitivität kann mittels Übereinstimmung das Ergebnis noch einmal signifikant verbessert werden.

In Abb. 5.25 sind die Konfusionsmatrizen der einzelne Klassifikatoren, des *Naive Bayes* Ensemble und der Klassifikation durch Übereinstimmung dargestellt. Hierbei zeigt sich, dass die Genauigkeit durch die *Naive Bayes* in fast allen Klassen, außer *Vegetation* und *Bordstein* besser ist, als das beste Einzelergebnis. Bei den Klassen *Vegetation* und *Bordstein* liegt das Ergebnis im Bereich zwischen den einzelnen Klassifikatoren. Die signifikante Verschlechterung der Klasse *Vegetation*, könnte durch die hohe Falsch-Positive Rate des *Random-Forest* Klassifikator begründbar sein.



(a) CNN.

(b) Random Forest.

(c) Naive Bayes.

(d) Übereinstimmung.

Abbildung 5.25: Konfusionsmatrizen der ursprünglichen und kombinierten Klassifikatoren.

Mithilfe der Klassifikation durch Übereinstimmung, lässt sich die Genauigkeit der einzelnen Klassen signifikant verbessern. Dabei werden jedoch etwa 60 % der Klasse *Andere*, etwa 30 % der Klassen *Pfosten*, *Bordstein*, *Gebäude*, etwa 20 % der Klasse *Vegetation* und etwa 15 % der Klasse *Fahrzeug* als nicht klassifizierbar zurückgewiesen.

Zusammengefasst lässt sich feststellen, dass die Kombination vom Experten designten Merkmalsextraktoren und gelernten Merkmalsextraktoren zu einem Informati-

onsgewinn führen und das Klassifikationsergebnis verbessern. Als beste Methode ohne Einschränkung hat sich der *Navive Bayes* Ansatz erwiesen. Falls die Genauigkeit auf Kosten der Sensitivität verbessert werden soll, kann man dies mithilfe von Übereinstimmung erreichen.

5.6 Zusammenfassung

In diesem Abschnitt wurden verschiedene Methoden zum Klassifizieren von Radarclustern vorgestellt. Hierbei wurde auf Verfahren mit tiefen neuronalen Netzwerken eingegangen.

Anhand verschiedener Experimente wurden Eigenschaften der Radar spezifischen Klassifikation beleuchtet. Es wurde gezeigt, welche Klassen sich eignen, um sie mithilfe von Radardaten zu unterscheiden.

Es wurde festgestellt, dass die Merkmalsextraktoren von auf Bildern vortrainierten Faltungsnetzwerken, sich auch für die Klassifikation von Radar Clustern eignen. Es kann dadurch eine deutlich bessere Performance erreicht werden, als bei zufällig initialisierten Netzwerken.

Des Weiteren wurden verschiedene Merkmalskarten untersucht und festgestellt, dass sich Belegungskarten, RCS-Histogramm- und RCS-Maximum-Karten gut für die Klassifikation eignen.

In einem weiteren Experiment wurde gezeigt, dass mittels einer geeigneten Augmentierung das Ergebnis noch weitere verbessert werden kann, ohne dass neue Trainingsdaten benötigt werden. Sinnvolle Parameter für die Augmentierungstechniken können mithilfe eines genetischen Algorithmus gefunden werden.

Bei der Gegenüberstellung war zu sehen, dass tiefe Netzwerke klassischen Klassifikationsverfahren überlegen sind. Jedoch wurde auch gezeigt, dass durch die Kombination der beiden Verfahren eine weitere Verbesserung des Klassifikationsergebnisses erzielt werden kann.

Kapitel 6

Semantische Segmentierung von Radar-Belegungskarten

Für die im vorherigen Kapitel beschriebene Klassifikation von Objektinstanzen, müssen zuerst Objektvorschläge aus den Belegungskarten mittels eines Clustering-Algorithmus extrahiert werden. Im zweiten Schritt wird vom Klassifikator eine Klasse zugeordnet. Die Klassifikation hängt daher stark vom Ergebnis des Clustering ab. Hierbei gibt es einige Fälle die Schwierigkeiten verursachen können.

Zu den Problemfällen zählen Objekte, die in der Welt vertikal überlappen. Beispielsweise ein Verkehrsschild, welches genau auf dem Randstein oder mitten in der Vegetation steht. Ein ähnliches Problem tritt bei Objekten auf, die so dicht nebeneinanderstehen, dass sie, aufgrund der Messunsicherheit, in der Belegungskarte miteinander verschwimmen und dadurch nicht mehr klar getrennt werden können. Dies ist z. B. bei sehr dicht parkenden Autos der Fall. Dadurch können Cluster entstehen, die mehrere Objekte umfassen. Sofern diese Objekte unterschiedlichen Klassen angehören, führt dies zwangsweise zu einer Fehlklassifikation.

Ein anderes Problem ist das Zerfallen von Objekten in mehrere Cluster. Dies führt dazu, dass jeder Teil eines Objektes als separater Objektvorschlag klassifiziert werden muss. Auch wenn dadurch einer korrekten Klassifikation nichts im Wege steht, wird ein erhöhter Rechenaufwand erzeugt, da pro Objekt mehrere Cluster klassifiziert und falls nötig, später wieder zusammengefügt werden müssen. Wie bereits in Abb. 4.3 im Abschnitt 4.3.1 beschrieben, tritt dieser Fall häufiger auf. Für ein

System, welches unter Echtzeitanforderungen laufen sollte, kommt hinzu, dass die unterschiedliche Anzahl an Clustern, zu starken Schwankungen in der Rechenzeit führen. Ein Echtzeitsystem muss daher immer die Ressourcen für den schlechtesten Fall vorhalten.

Bei der semantischen Segmentierung wird hingegen auf ein vorheriges Clustering verzichtet, in dem die Klassifikation direkt auf die Karten angewendet und die Klassifikation für jede Zelle bestimmt wird. In einem zweiten Schritt lassen sich, falls nötig, Objektinstanzen mithilfe der Semantik extrahieren. Bei einer naiven Implementierung wäre die pixelweise Klassifikation sehr rechenintensiv, aber durch eine geschickte Mehrfachnutzung von Zwischenergebnissen kann diese sehr effizient realisiert werden.

In diesem Kapitel soll untersucht werden, ob sich Methoden der semantischen Segmentierung auf Radar-Belegungskarten übertragen lassen und welche Schwierigkeiten sich dabei ergeben. Hierzu wird als Erstes ein Überblick über den Stand der Technik gegeben, dieser ist als Ergänzung des Abschnitts 5.1 zu sehen. In Abschnitt 6.2 wird auf die semantische Segmentierung von Radar-Merkmalskarten eingegangen. Abschnitt 6.3 behandelt eine Erweiterung, diese soll ermöglichen, die gesamte Ausdehnung von Objekten zu schätzen. Dies wird anhand der Klasse *Fahrzeug* untersucht.

In den Abschnitten 6.4 und 6.5 werden die Ergebnisse der jeweiligen Verfahren präsentiert.

6.1 Stand der Technik

In diesem Kapitel soll nun auf die Besonderheiten der semantischen Segmentierung eingegangen werden. Eine allgemeine Übersicht über Neuronale Netzwerke und deren Optimierung wurde schon in Abschnitt 5.1.2 gegeben. Eine Übersicht von Klassifikationsverfahren im automobilen Bereich und im Bereich des Radars wurde in den Abschnitten 5.1.4 und 5.1.5 vorgestellt.

Unter semantischer Segmentierung versteht man das Klassifizieren von Bildern auf der Pixelebene. Es geht also nicht mehr darum, ein Bild als Ganzes zu klassifizieren, sondern für jedes Pixel eine Klassenzuordnung zu treffen. Bei ausgedehnten Objekten entstehen dadurch Regionen mit gleicher Klassenzugehörigkeit. Ein Beispiel wie dies bei Bildern aussehen kann, ist in Abbildung 6.1 zu sehen. Dieses aus der Arbeit von Khan et al. [Kha+14] stammende Bild zeigt die Eingangsdaten, die Ergebnisse und das zugehörige Label.

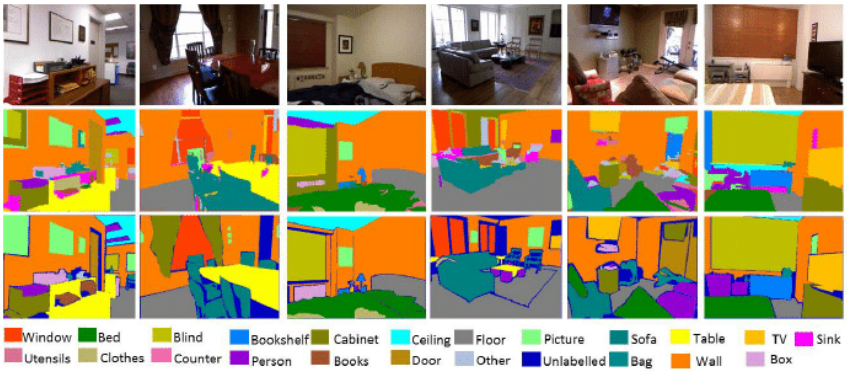


Abbildung 6.1: Beispiel Bild für semantische Segmentierung aus [Kha+14]. In der ersten Zeile befindet sich das Eingangsbild, in der zweiten die Prädiktion des Klassifikators und in der letzten Zeile das gelabelte Bild.

Die semantische Segmentierung ist in der englischsprachigen Literatur unter mehreren Begriffen zu finden, dabei werden die Bezeichnungen *semantic segmentation*, *scene labeling* und *pixelwise classification* annähernd gleichwertig verwendet.

Noch bevor tiefe neuronale Netzwerke populär wurden, zeigten Shotton et al. die semantische Segmentierung mithilfe eines *Random-Forests* [SJC08]. Für die Klassifikation eines Pixels wurde ein Ausschnitt der umliegenden Pixel als Eingangsdaten verwendet. Hierbei wurde auf eine vorherige Merkmalsextraktion verzichtet und direkt die Farbwerte der Pixel verwendet.

2014 stellte Long et al. das Konzept des *Fully Convolutional Network* vor [LSD14]. Basierend auf herkömmlichen Netzwerkarchitekturen wie z. B. das *AlexNet* zeigte er, dass die voll verbundenen Schichten ebenfalls als Faltungsschicht aufgefasst werden können. Hierfür wird die voll verbundene Schicht durch einen Faltungskern ersetzt, welcher so viele Kanäle besitzt, wie Neuronen in der ursprünglichen Schicht vorhanden waren. Zudem wurde vorgestellt, wie effizient mit *Downsampling* und *Upsampling* der Pooling Schichten umgegangen werden kann.

Eine Weiterentwicklung dieser *Fully Convolutional Neural Networks* sind die *Deconvolutional Neuronal Networks* [Zei+10; NHH15]. Bei diesen wird nach den Faltungsschichten eine "umgekehrte" Faltung (*engl. deconvolution oder transposed convolution*) durchgeführt. Noh et al. [NHH15] konstruierten das Netzwerk so, dass nach den Faltungsschichten des Netzwerkes eine gespiegelte Architektur mit "umgekehrten" Faltungsschichten folgt. Als Inverse der Max-Pooling Schichten werden dabei Unpooling Schichten verwendet. Diese sind so konstruiert, dass der Index des maximalen Wertes gespeichert wird. Dieser Index wird in der gespiegelten Unpooling-Schicht

verwendet, um den Eingangswert auf den entsprechenden Index zu projizieren. Alle anderen Zellen erhalten den Wert 0. Durch dieses Vorgehen kann die räumliche Assoziation der Merkmale besser gewährleistet werden.

Ronneberger et al. stellten mit der *U-Net* Architektur einen Ansatz vor, um die räumliche Zuordnung zu verbessern [RFB15]. Für die umgekehrten Faltungen schlugen sie einen zusätzlichen Pfad vor, der an verschiedenen Stellen der Faltungsschichten den Ausgangstensor kopiert und diesen als zusätzlichen Eingang für den Aufwärtspfad zur Verfügung stellt.

Yu et al. präsentierten eine Methode, wie mithilfe von *Dilated Convolution* ein größerer Umgebungskontext verwendet werden kann, ohne dabei den Rechenaufwand zu erhöhen [YK16]. Hierzu tastet ein Filter nicht mehr alle benachbarten Pixel in seinem Einzugsbereich ab, sondern es erfolgt eine Unterabtastung, indem er beispielsweise nur jedes 2. Pixel als Eingang für den Filter verwendet wird. Dadurch vergrößert sich der Kontext des Filters, ohne dass sich der Filter selbst vergrößert. Bei einer Schrittweite von 1 werden die Informationen der ausgelassenen Pixel von dem um eins verschobenen Filter ausgewertet.

Chen et al. erweiterten dieses Prinzip, indem unterschiedlich große Unterabtastungen in seinem Netzwerk parallel verwendet. Dies macht das Netzwerk robust gegenüber Skalierungseffekten, welche in Kamerabildern beispielsweise durch unterschiedliche Entfernungen von Objekten verursacht werden [Che+17a].

Um den Einfluss der Kontextwahrnehmung zu schärfen, stellte Zhao et al. ein Netzwerk vor, dass nach der Extraktion von Merkmalen durch die Faltungsschichten unterschiedlich große Pooling Schichten einsetzt, um Merkmale aus unterschiedlichen Umgebungsbereichen zu extrahieren [Zha+17]. Diese werden anschließend auf dieselbe Größe skaliert. Diese aneinander gestapelten Merkmalskarten werden danach für die Klassifikation verwendet.

6.1.1 Metriken & Kostenfunktion

Die Bewertung eines Algorithmus zur semantischen Segmentierung gestaltet sich bei den meisten Datensätzen schwieriger als bei der Klassifikation von Objektinstanzen. Dies hat den Grund, dass die Klassen in den meisten Datensätzen stark ungleich vertreten sind, und/oder die Anzahl der Pixel sich stark unterscheidet. Ein Ausgleichen durch duplizieren der Beispiele ist nicht so einfach möglich, da die einzelnen Beispiele meist aus mehreren Klassen bestehen.

Bei den Belegungskarten kommt hinzu, dass über 90 % der Zellen/Pixel das Label *Hintergrund* haben. Und somit ein, wie in Abschnitt 4.5.4 zu sehen, starkes Ungleichgewicht hin zu dieser Klasse besteht.

Um Algorithmen der semantischen Segmentierung zu bewerten, wird häufig die *Intersection over Union* (IoU) verwendet [LSD14; Che+17a; NHH15; YK16]. Diese findet man auch unter den Begriffen Jaccard-Koeffizient oder Jaccard-Index [Jac12]. Die IoU bestimmt das Verhältnis der Überlappung zweier Flächen zu ihrer Vereinigung. Hat man also zwei Flächen A und B wird die IoU wie folgt berechnet.

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{6.1}$$

Bei einem binären Klassifikator entspricht das

$$IoU = \frac{\text{Richtig Positiv}}{\text{Falsch Positiv} + \text{Falsch Negativ} + \text{Richtig Positiv}} \tag{6.2}$$

Zur Bewertung eines Klassifikators mit mehreren Klassen wird eine mittlere IoU berechnet. Dies kann auf zwei Arten geschehen, der Mikro- oder der Makro-Performance. Bei der Mikro-Performance werden jeweils alle Intersections und alle Unions summiert und anhand der Summen die Mikro-IoU berechnet. Bei der Makro-Performance wird für jede Klasse separat die IoU ermittelt und anschließend der Mittelwert gebildet. Dadurch wird bei der Makro-IoU erreicht, dass alle Klassen, unabhängig von ihrer Häufigkeit und ihrer durchschnittlichen Fläche, in gleichem Maße berücksichtigt werden.

Ein nicht so häufig verwendeter Maßstab ist der Dice Koeffizient [Pow11], welcher auch unter dem Namen F1-Maß bekannt ist. Dieser unterscheidet sich nur leicht von der IoU, indem die Schnittfläche von A und B doppelt in die Normierung mit eingeht.

$$Dice(A, B) = \frac{2|A \cap B|}{|A| + |B|} \tag{6.3}$$

Auch wenn sich diese Funktionen gut für die Bewertung von Algorithmen zur semantischen Segmentierung eignen, so kann aufgrund der Unstetigkeit der Funktion kein Gradient bestimmt werden. Somit können diese Funktionen nicht als Kostenfunktionen für Lernalgorithmen verwendet werden, die auf dem Gradientenabstieg basieren.

Wie auch bei Instanz basierenden Verfahren wird zur Optimierung häufig die Kreuzentropie verwendet. Dabei kann man sich jedes Pixel als Beispiel eines Minibatches vorstellen, deren Kosten anschließend gemittelt werden.

Die ungleiche Verteilung der Klassen lässt sich dabei durch eine Gewichtung, entweder lokal pro Bild oder über den ganzen Datensatz behandeln. Long et al. schlägt zudem vor, dass mit einer geeigneten Abtastung des Ausgangsbildes eine Verbesserung der Klassenverteilung fürs Training erreicht werden kann [LSD14]. Hierbei fließen nicht mehr alle Pixel in die Kostenfunktion ein, sondern nur noch die, welche durch die Abtastung ausgewählt werden. Hierbei erhöht sich jedoch der Rechenaufwand, da nur noch ein Teil der prädizierten Pixel für den Gradientenabstieg verwendet werden.

Eine weitere Kostenfunktion wurde von Rahman et al. vorgeschlagen [RW16]. Diese basiert auf der IoU, bei der die Berechnung so angepasst wurde, dass die Funktion bezüglich der Netzwerkparameter partiell differenzierbar ist.

Die IoU basiert auf einer binären Entscheidung, ob ein Pixel zu einer Klasse gehört. Der Ausgang des Klassifikators wird jedoch oft als Wahrscheinlichkeitsverteilung interpretiert. Um die IoU zu berechnen, wird dieser mithilfe einer Argmax-Funktion binarisiert. In der Arbeit von Rahman wurde auf diese Binarisierung verzichtet und der Binärvergleich durch eine Multiplikation ersetzt.

Nimmt man den Ausgangstensor des Klassifikators Y_{pred} und den vorgegebene Labeltensor Y_{label} mit den korrespondierenden Zellen $Y_{pred,v}$ und $Y_{label,v}$, wobei $v \in V$ den Index der Zellen darstellt, so kann die modifizierte IoU wie folgt berechnet werden:

$$\mathrm{IoU} = \frac{I(Y_{pred}, Y_{label})}{U(Y_{pred}, Y_{label})} \tag{6.4}$$

mit

$$\mathrm{I}(Y_{pred}, Y_{label}) = \sum_{v \in V} (Y_{pred,v} * Y_{label,v}) \tag{6.5}$$

$$\mathrm{U}(Y_{pred}, Y_{label}) = \sum_{v \in V} (Y_{pred,v} + Y_{label,v} - (Y_{pred,v} * Y_{label,v})) \tag{6.6}$$

Damit wird die Kostenfunktion wie folgt definiert:

$$L_{\mathrm{IoU}} = 1 - \frac{I(Y_{pred}, Y_{label})}{U(Y_{pred}, Y_{label})} \tag{6.7}$$

Der Vorteil dieser Funktion besteht darin, dass sie unabhängig von der Anzahl der Pixel einer Klasse ist. Die Fehler von Klassen mit kleiner Fläche und von Klassen mit großer Fläche, gehen in gleichem Maße in die Kostenfunktion mit ein. Ebenso werden für jede Klasse sowohl Falsch-Positive wie auch Falsch-Negative Klassifikationsergebnisse berücksichtigt.

6.2 Semantische Segmentierung von Radar-Merkmalskarten

Anhand des in 4.5.4 beschriebenen Datensatzes und der in Abschnitt 5.5.3 bestimmten Klassen soll in diesem Kapitel gezeigt werden, wie sich Ansätze der semantischen Segmentierung zum Erzeugen einer semantischen Belegungskarte (*engl. Semantic Radar Grids*) eignen.

Dies wurde in [Lom+17b] für das Beispiel von Fahrzeugen untersucht. In diesem Kapitel wird eine Erweiterung auf alle Klassen, die in Abschnitt 5.5.1 herausgearbeitet wurden, betrachtet. Es wird darüber hinaus eine andere Netzwerkstruktur als in [Lom+17b] verwendet.

Die dazu verwendete Netzwerkarchitektur ist in Abbildung 6.2 zu sehen. Die verwendete Architektur ist inspiriert von [NHH15]. Sie ermöglicht es, als Ergebnis dichte semantische Segmentierungsmasken zu erzeugen. Die Struktur des Netzwerks besteht aus zwei Teilen. Im ersten Teil ist das normale Faltungsnetzwerk zu finden, welches die Eingangsdaten in einen Merkmalsraum mit geringerer räumlichen Auflösung projiziert. Dieses wird durch Faltungsschichten und Max-Pooling Schichten erreicht.

Im zweiten Teil des Netzwerkes werden diese Merkmale stufenweise wieder auf die ursprüngliche Auflösung projiziert. Dies geschieht mithilfe von transponierten Faltungsschichten (*Deconvolution Layer* [Zei+10] oder auch *Transposed convolutional Layer* [DV16]) und Upsampling-Schichten.

Als Aktivierungsfunktion wurde die ReLU-Aktivierungsfunktion verwendet. Der Ausgang des Netzwerkes wurde mithilfe der Softmax Funktion in einen als Wahrscheinlichkeitsverteilung interpretierbaren Wertebereich transformiert.

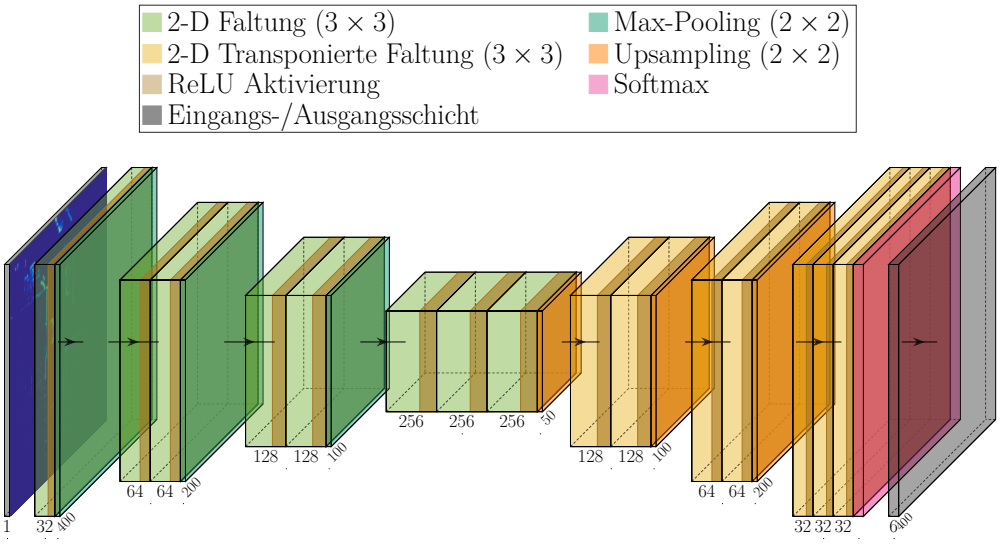


Abbildung 6.2: Hier verwendetes Netzwerk zur semantischen Segmentierung

Um ein robusteres und schnelleres Lernen zu ermöglichen, wurden zusätzlich an verschiedenen Punkten Batch-Normalisierungs-Schichten eingesetzt. Als Lernalgorithmus wurde wiederum Adam eingesetzt.

Die Trainingsdaten bestehen aus Radar-Merkmalskarten in der Größe von 400×400 Zellen um das Fahrzeug. Diese Einschränkung wurde aus zwei Gründen gewählt. Einerseits, um den benötigten Speicherplatz zu limitieren und dadurch ein effizientes Training auf der Grafikkarte mit mehreren Batches zu ermöglichen, andererseits, da die unmittelbare Umgebung des Fahrzeuges für eine Fahraufgabe am relevantesten ist.

Die Daten sind Sequenzweise in HDF5 Dateien gespeichert. Um effizient auf dieser Datenstruktur zu operieren, wurden die einzelnen Trainingsbeispiele wie folgt ausgewählt. Als Erstes wurden zufällig 5 Sequenzen geladen. Aus diesen wurden wiederum zufällig jeweils 10 Szenen ausgewählt, aus der die Karten extrahiert wurden. Diese extrahierten Beispiele wurden dann, in einer zufälligen Reihenfolge dem Klassifikator präsentiert. Dieser Prozess wurde wiederholt, sodass die Auswahl der Beispiele dem Urnenmodell, ziehen mit Zurücklegen, entspricht, mit der Einschränkung, dass jeweils 10 Beispiele aus einer Sequenz stammen. Für den Fall, dass eine leere Karte gezogen wird, wird dieses Beispiel verworfen.

Zum Optimieren des CNNs wird eine geeignete Kostenfunktion benötigt, welche die

ungleiche Verteilung der Klassen berücksichtigt. Diese werden im Folgenden vorgestellt und dann in Abschnitt 6.4 bezüglich ihrer Eignung untersucht.

Bevor die Kostenfunktionen hergeleitet werden, soll noch der Bewertungsmaßstab erläutert werden. Wie beim Cluster bezogenen klassifizieren, soll auch hier das Klassifikationsergebnis einer Klasse, unabhängig von deren Häufigkeit im Datensatzes bewertet werden. Daher wird zur Beurteilung der Makro-IoU vorgeschlagen. Hierdurch gehen die Klassen, unabhängig von ihrem Vorkommen und ihrer Fläche, in das Bewertungsmaß mit ein. So haben auf einem Parkplatz wenige Pfosten mit kleiner Fläche den gleichen Einfluss auf die Bewertung wie viele großflächige Fahrzeuge.

Je nach Anwendungsfall wäre auch ein anderer Bewertungsmaßstab sinnvoll, nach Meinung des Autors eignet sich der gewählte am besten, um allgemeingültige Aussagen zu treffen. Da ansonsten bei dem bestehenden Datensatz Klassen mit geringem vorkommen, basierend auf der Fläche, kaum in die Bewertung einfließen würden (vgl. Tabelle 4.2) und somit nur eine Aussage über die überrepräsentierten Klassen getroffen würde.

Zum Verständnis der verschiedenen Kostenfunktionen ist es hilfreich, sich noch einmal vor Augen zu führen, wie die Label zustande kommen. Beim Labeln werden die einzelnen Objekte von den Labelern umrahmt. Ausgehend von dem Polygon wird das Label auf die Zellen projiziert, es erhalten aber nur die Zellen das Label, deren Belegung größer als τ_{valid} der Belegungskarte ist. Alle anderen Zellen erhalten das Label *Hintergrund*.

Ebenso wie bei der Klassifikation von Clustern kann die Kreuzentropie als Kostenfunktion verwendet werden. Hierbei ist $\hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c})$ die vom Klassifikator vorhergesagte Wahrscheinlichkeit, dass die Zelle v der Karte V zur Klasse $\mathfrak{c}$ gehört. Die Wahrscheinlichkeit des Labels einer Zelle ist entweder mit 0 oder 1 definiert. Es wird durch $p(\boldsymbol{y}_{label,v} = \mathfrak{c})$ angegeben.

$$L_{entro.,v} = -\sum_{\mathfrak{c}\in\mathfrak{C}} p(\boldsymbol{y}_{label,v} = \mathfrak{c}) \log \hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c}) \tag{6.8}$$

Dieses Kostenfunktion wird dann über alle N Zellen der Karte gemittelt.

$$L_{entro.} = \frac{1}{N} \sum_{v\in V} L_{entro.,v} \tag{6.9}$$

Durch die ungleichmäßige Verteilung der Klassen über die Zellen eines Beispiels bietet es sich an, die Klassen reziprok zu ihrem Vorkommen zu gewichten. Dabei

bezeichnet $N_{\mathfrak{c}}$ die Anzahl der Zellen die, laut Label, zur Klasse $\mathfrak{c}$ gehören. Damit ergibt sich die Kostenfunktion zu:

$$L_{entro.,gew.} = -\sum_{v \in V} \sum_{\mathfrak{c} \in \mathfrak{C}} \frac{1}{N_{\mathfrak{c}} + 10^{-9}} p(\boldsymbol{y}_{label,v} = \mathfrak{c}) \log \hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c}) \tag{6.10}$$

Der Term 10^{-9} dient dazu den Sonderfall, dass die Anzahl der Zellen einer Klasse 0 ist abzufangen.

In den beiden vorherigen Kostenfunktionen wird die Klasse *Hintergrund* mitgelernt. Hierbei muss das Netzwerk neben den Klassenzuordnungen den Schwellwert τ_{valid} lernen, ab dem eine Zelle als Hintergrund angesehen wird. Da die Gültigkeit einer Zelle lediglich auf der Belegungskarte und dem Schwellwert beruht, kann diese auch im Nachhinein korrigiert werden. Hierzu wird die Gültigkeitsmaske auf das Klassifikationsergebnis angewendet.

Dadurch kann auf das Lernen der Klasse *Hintergrund* verzichtet werden. Hieraus lassen sich zwei Kostenfunktionen ableiten, welche die Hintergrundzellen mit 0 gewichten und damit ein spärliches Lernen ermöglichen. Die Zellen, die als Hintergrund gelten, können dadurch einer beliebigen Klasse zugeordnet werden, ohne dass dies bestraft wird. Jenes wird erreicht, indem über alle Klassen, außer der Klasse *Hintergrund*, $\overline{\mathfrak{C}}$ summiert wird.

Daraus ergibt sich die Kostenfunktion der Kreuzentropie ohne Hintergrund durch

$$L_{entro.,o.H.} = -\frac{1}{\sum_{\mathfrak{c} \in \overline{\mathfrak{C}}} N_c} \sum_{v \in V} L_{entro.,v} \sum_{\mathfrak{c} \in \overline{\mathfrak{C}}} p(\boldsymbol{y}_{label,v} = \mathfrak{c}) \log \hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c}) \tag{6.11}$$

und die gewichtete Kreuzentropie durch

$$L_{entro.,gew.,o.H.} = -\sum_{v \in V} \sum_{\mathfrak{c} \in \overline{\mathfrak{C}}} \frac{1}{N_{\mathfrak{c}} + 10^{-9}} p(\boldsymbol{y}_{label,v} = \mathfrak{c}) \log \hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c}). \tag{6.12}$$

Als weitere Kostenfunktion kann die L_{IoU} verwendet werden. Diese eignet sich primär für binäre Entscheidungsprobleme. Jedoch ist es möglich, durch eine einfache Adaption diese auch für Probleme mit mehreren Klassen zu verwenden. Hierzu wird die L_{IoU} für jede Klasse einzeln bestimmt und anschließend die Kosten summiert.

$$L_{IoU,MC} = \sum_{\mathfrak{c} \in \mathfrak{C}} \left(1 - \frac{I_{\mathfrak{c}}}{U_{\mathfrak{c}}}\right) \tag{6.13}$$

mit

$$I_{\mathfrak{c}} = \sum_{v \in V} p(\boldsymbol{y}_{label,v} = \mathfrak{c}) \hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c}) \tag{6.14}$$

$$U_{\mathfrak{c}} = \sum_{v \in V} p(\boldsymbol{y}_{label,v} = \mathfrak{c}) + \hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c}) - (p(\boldsymbol{y}_{label,v} = \mathfrak{c}) \hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c})) \tag{6.15}$$

Auch hier kann der Hintergrund ignoriert werden.

$$L_{IoU,MC,o.H.} = \sum_{\mathfrak{c} \in \bar{\mathfrak{C}}} 1 - \frac{I_{\mathfrak{c}}}{\tilde{U}_{\mathfrak{c}}} \tag{6.16}$$

mit

$$\begin{aligned} \tilde{U}_{\mathfrak{c}} = \sum_{v \in V} & p(\boldsymbol{y}_{label,v} = \mathfrak{c}) + \hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c})(1 - p(\boldsymbol{y}_{label,v} = \mathfrak{c}_{\text{Hintergrund}})) \\ & - p(\boldsymbol{y}_{label,v} = \mathfrak{c}) \hat{\boldsymbol{p}}(\boldsymbol{y}_{pred,v} = \mathfrak{c}) \end{aligned} \tag{6.17}$$

Ein weiterer Aspekt, welcher der Optimierung bedarf, ist der Schwellwert τ_{valid}. Dies soll anhand verschiedener Schwellwerte zur Evaluierung und zum Training untersucht werden.

Ebenso wie bei den Cluster basierenden Methoden lassen sich auch verschiedene Radar-Merkmalskarten für die Klassifikation nutzen. Um den Trainingsaufwand zu reduzieren, werden nur ausgewählte Kartenkombinationen untersucht.

6.3 Semantische Segmentierung von vollständigen Fahrzeugen

Durch die Perspektive des Radarsensors werden Objekte meist nur von einer Seite beobachtet. Daher bildet sich nicht die vollständige Form der Objekte in einer Radar-Belegungskarte ab, sondern zum Großteil nur die dem Sensor zugewandte Seite. Bei Häusern können z. B. nur die Wände, aber nicht die Fläche des Hauses erfasst werden. Zum anderen ist ein Objekt oft nur in einem kleinen Winkelbereich beobachtbar, sodass nur ein Teil der Außenkontur des Objektes zu sehen ist. So sieht

Abbildung 6.3: Unstrukturierter Parkplatz.

man beispielsweise bei quer geparkten Fahrzeugen nur den Front- oder den Heckbereich vollständig. Der Rest des Fahrzeuges bleibt teilweise oder ganz verborgen. Auch wenn dieses Phänomen auf einige Objekte zutrifft, soll in diesem Kapitel nur die Klasse *Fahrzeug* betrachtet werden, da diese Klasse für viele Bereiche besonders interessant ist.

Für die Lokalisation ist die vollständige Ausdehnung der Fahrzeuge von Bedeutung, da alle Messpunkte, die sich im Bereich eines Fahrzeuges befinden, ignoriert werden sollen. Die Straßenlaterne hinter dem Fahrzeug stellt jedoch sehr wohl eine gute Landmarke dar.

Ein weiterer Einsatzbereich ist die Unterstützung von automatisierten Parksystemen. Hier ist das Ziel, vorausschauend geeignete Parkplätze zu finden und diese so präzise zu bestimmen, dass eine Trajektorienplanung ermöglicht wird. Hierbei sind bereits parkende Autos ein guter Anhaltspunkt. Besonders wenn, wie in Abb. 6.3 zu sehen ist, keine Parkplatzmarkierungen sichtbar sind, muss ein autonomes Fahrzeug, ähnlich wie der Mensch, den Parkplatz so wählen, wie es durch die bereits parkenden Fahrzeuge vorgeben wird. Hierzu ist es hilfreich, die Ausdehnung und Ausrichtung der schon geparkten Fahrzeuge zu kennen.

Während im vorherigen Kapitel lediglich eine Klassifikation vorgenommen wurde, soll nun untersucht werden, in welchem Umfang es möglich ist, die vollständige Ausdehnung der Objekte zu schätzen. Ebenso soll mit dieser Methode die Möglichkeit einer Instanzsegmentierung der einzelnen Fahrzeuge betrachtet werden. Zusätzlich soll die Orientierung und Größe der Fahrzeuge geschätzt werden.

Hierzu nimmt ein auf einem CNN basierender Algorithmus eine semantische Segmentierung vor. Dabei ist das Ziel alle Zellen auf denen sich das Auto befindet, das

Label *Fahrzeug* zuzuweisen. Aus den extrahierten Objekten wird anschließend die Größe und Ausrichtung der Fahrzeuge bestimmt.

Als Klassifikator wird ein neuronales Netzwerk ähnlich [NHH15] verwendet. Dies ist in Abbildung 6.4 dargestellt und besteht aus 5 *Convolutional*- und 5 *Deconvolutional*-Schichten und aus zwei *Pooling*- und zwei *Unpooling*-Schichten. Für ein schnelleres und robusteres Training wird eine *Batch-Normalization*-Schicht verwendet. Die Erfahrung zeigte, dass weitere *Batch Normalization* Schichten zu keiner weiteren Verbesserung des Klassifikators führen. Als Aktivierungsfunktion wird im Netzwerk die ReLU Funktion verwendet, für den Ausgang eine Sigmoid Aktivierung. Damit wird erreicht, dass der Wertebereich der Ausgabe des Klassifikators zwischen 0 und 1 liegt.

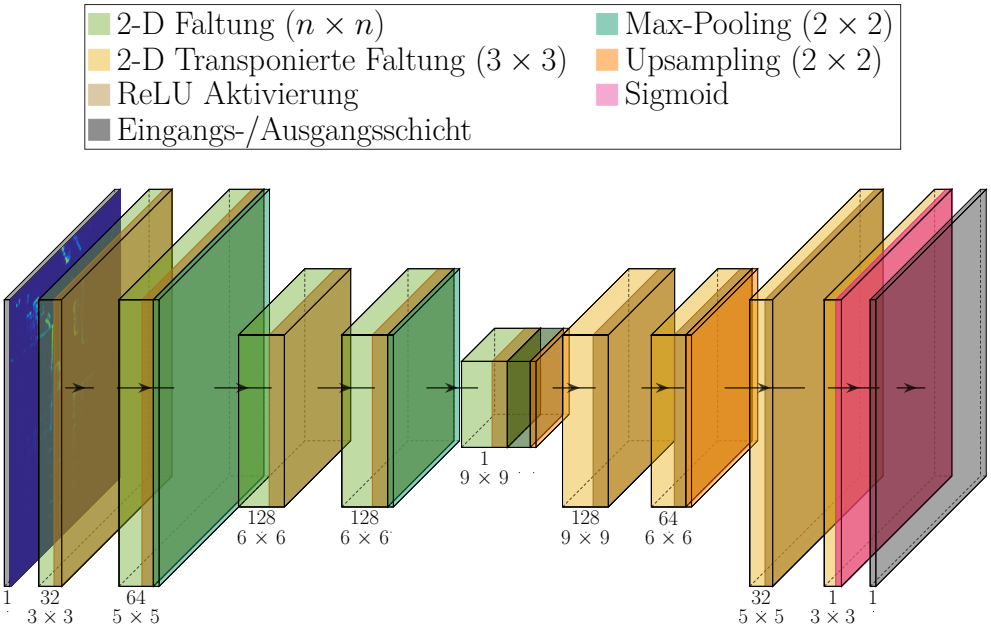


Abbildung 6.4: Hier verwendete Netzwerkstruktur zur Schätzung der Fahrzeugkontur.

Für das Training des CNNs wurde das IoU als Zielfunktion gewählt. Diese hat die Eigenschaft die Überlappung der Flächen unabhängig von deren Größe und deren Häufigkeit zu optimieren. Daher ist die Optimierungsfunktion unabhängig von der Anzahl und der Größe der Fahrzeuge in einer Belegungskarte. Um das Training effizient zu gestalten, wurden Karten die keine Fahrzeuge enthalten nicht zum Training verwendet.

Da die Radarmessungen zeitlich sequenziell in die Belegungskarte eingetragen werden, muss auch hier entschieden werden, ab welchem Belegungszustand ein Label als gültig berücksichtigt wird. Für die Fahrzeuginstanzen wird die Schwelle so gewählt, dass mindestens eine Zelle eines Fahrzeuglabels L_n eine Belegungswahrscheinlichkeit größer als $\tau_{\text{valid}} = 0.6$ aufweisen muss.

$$\text{valid}(L_n, {}^{occ}M_t) = \tau_{\text{valid}} < \max({}^{occ}m_{i,j} \forall i, j (m_{i,j} \cap L_n \neq \varnothing)) \tag{6.18}$$

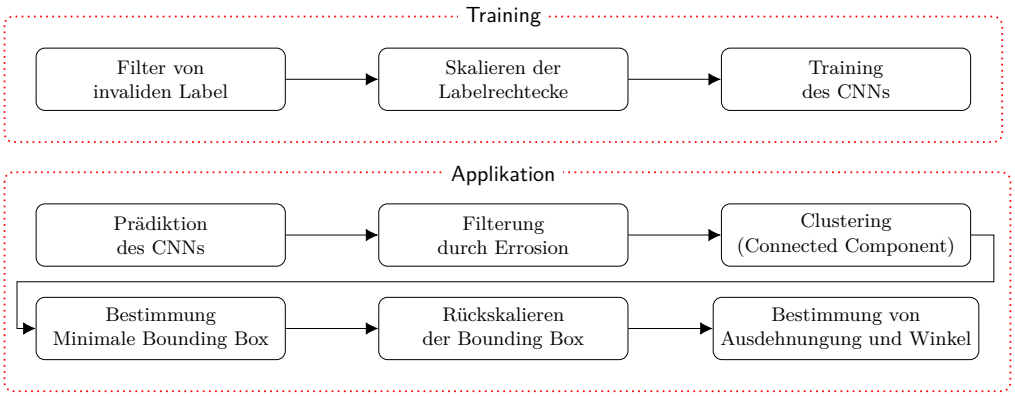


Abbildung 6.5: Ablaufdiagramm zum Segmentieren von Fahrzeug-Instanzen.

Erste Trainingsergebnisse ergaben ein Verschmelzen dicht geparkter Fahrzeuge. Dies kann durch zwei Faktoren erklärt werden. Zum einen ist die Trennfläche zwischen den Fahrzeugen so klein, dass dieser Fehler nur zu einem sehr geringen Anteil in die Kostenfunktion eingeht. Zum andern hatten die Labeler die Aufgabe, ein Rechteck um das gesamte Fahrzeug zu ziehen. Durch die Streuung des Radarsignals und durch Anbauteile die über die Rechteckkontur eines Fahrzeuges herausragen, wie z. B. Spiegel führt dies zu einer geringfügigen Überschätzung der Fahrzeugdimension.

Um diesem Problem zu begegnen, wurde ein Skalierungsfaktor 0.75 eingeführt, der die Seiten des bestehenden Rechtecklabel um diesen Faktor verkleinert. Dadurch sind die Instanzen gut separierbar. Die Größe des Fahrzeugs wird nach der Extraktion mittels dem inversen Faktor $1/0.75$ wiederhergestellt.

Durch das Trainieren mit diesen skalierten Labeln, lassen sich die Fahrzeuge gut voneinander separieren. Einzelne kleine Verbindungsstege zwischen zwei Fahrzeuginstanzen werden durch zwei aufeinander folgende Erosions- und Dilatationsschritte

beseitigt. Danach werden die Instanzen per *Connected-Component*-Analyse separiert und das *minimale gedrehte Bounding Rectangle* [Tou83] bestimmt.

Durch Rückskalierung kann nun die Fahrzeuggröße bestimmt werden. Des Weiteren wird mithilfe der *Bounding Box* die Orientierung des Fahrzeugs bestimmt, dabei wird angenommen, dass die längere Seite des Rechtecks die Längsorientierung des Fahrzeugs darstellt.

Das gesamte Vorgehen wird nochmals in Abb. 6.5 visualisiert.

6.4 Auswertung der Semantischen Segmentierung

In diesem Abschnitt wird die semantische Segmentierung der belegten Zellen anhand verschiedener Radar-Merkmalskarten untersucht. Für die Klassifikation wird die in Abschnitt 5.5.1 hergeleitete Klasseneinteilung verwendet.

Alle Auswertungen werden anhand des in Kapitel 4.5.4 beschriebenen Datensatzes durchgeführt. Im Folgenden werden zunächst mithilfe der Belegungskarten verschiedene Kostenfunktionen untersucht. Anschließend wird die Wahl des Gültigkeitsschwellwerts τ_{valid} untersucht. Zum Schluss wird ein Vergleich zwischen der semantischen Segmentierung mit der Belegungskarte und der RCS-Histogramm-Karte gezogen.

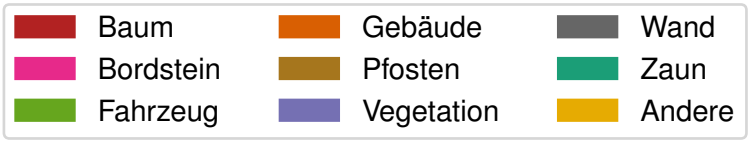


Abbildung 6.6: Farblegende der Klassen der semantischen Belegungskarte.

In Abbildung 6.6 ist die in diesem Kapitel verwendete Farblegende der einzelnen Klassen zu finden, die in allen Abbildungen von semantischen Belegungskarten verwendet wird.

6.4.1 Vergleich der Kostenfunktionen

In diesem Experiment werden die Kostenfunktionen aus Abschnitt 6.2 mit den Gleichungen 6.9, 6.10, 6.11, 6.12, 6.13 und 6.16 verglichen. Die Parameter des Experiments sind in Tabelle 6.1 zu finden.

Für jede Kostenfunktion wurde das Experiment fünfmal durchgeführt. Anschließend wurden mithilfe des Validationsdatensatzes die besten drei Klassifikatoren eines Trainingsdurchgangs bestimmt und diese auf dem Testdatensatz evaluiert. Durch dieses Vorgehen können Ausreißer, die das Ergebnis verfälschen, eliminiert werden.

Tabelle 6.1: Trainingsparameter

Parameter	Wert
Eingangsdaten	Belegungskarte
τ_{valid}	0.6
Minibatch Größe	5
Kartengröße	400×400
Kostenfunktion	*variiert*
Optimierer	Adam
Lernrate	0.001
β_1	0.9
β_2	0.999
Training	500 Epochen a 200 Minibatches

Tabelle 6.2: Vergleich der verschiedenen Kostenfunktionen.

	Kostenfunktion	Makro-IoU	Mikro-IoU
a)	Kreuzentropie (Gl. 6.9)	0.47 ± 0.01	0.61 ± 0.02
b)	Kreuzentropie gew. (Gl. 6.10),	0.48 ± 0.01	0.58 ± 0.02
c)	Kreuzentropie o. Hinterg. (Gl. 6.11)	0.48 ± 0.02	0.62 ± 0.03
d)	Kreuzentropie gew. o. Hinterg. (Gl. 6.12)	0.50 ± 0.00	0.62 ± 0.01
e)	IoU (Gl. 6.13)	0.18 ± 0.15	0.37 ± 0.11
f)	IoU o. Hinterg. (Gl. 6.16)	0.08 ± 0.02	0.26 ± 0.01

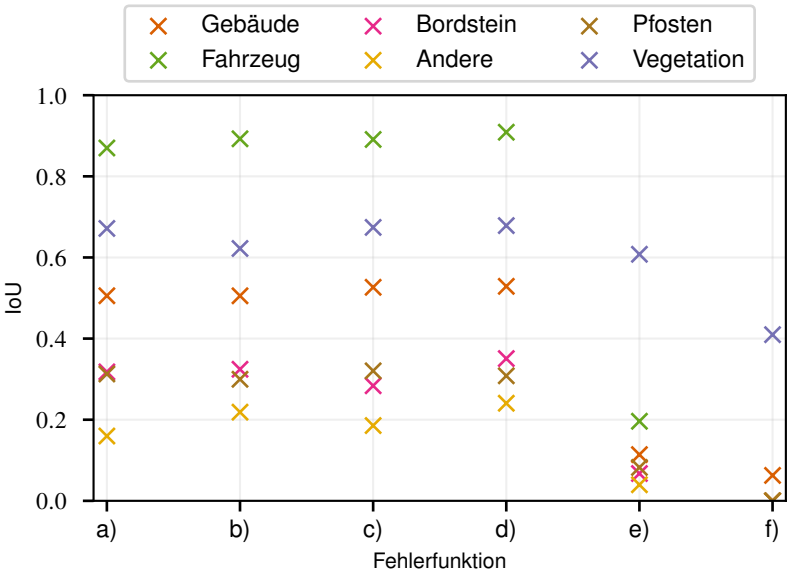
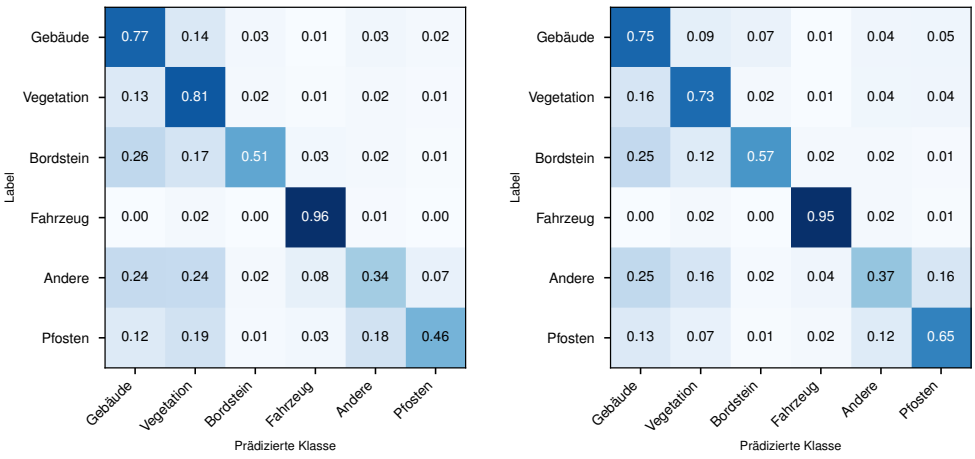


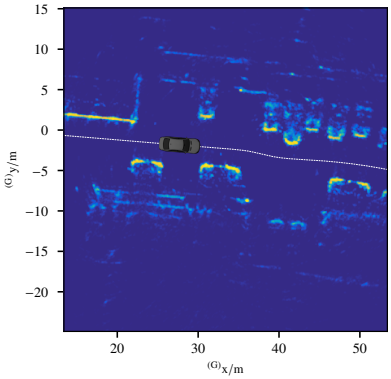
Abbildung 6.7: Mittlere IoU pro Klasse.



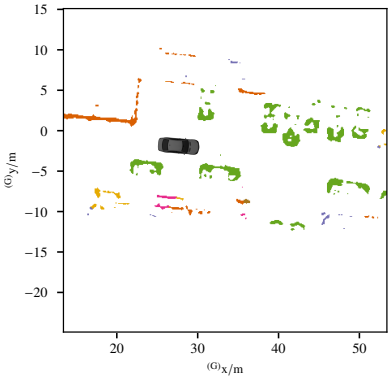
(a) Kreuzentropie o. Hinterg. (b) Kreuzentropie gew. o. Hinterg.

Abbildung 6.8: Vergleich der Konfusionsmatrizen der Methoden c) (nicht gewichtet) und d) (gewichtet).

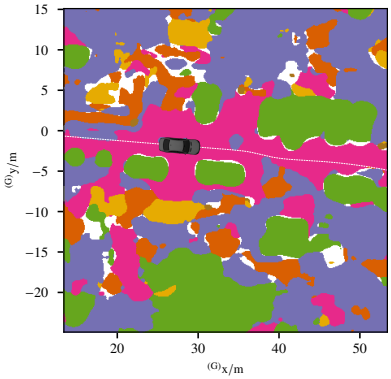
(a) Bild des Szenarios.



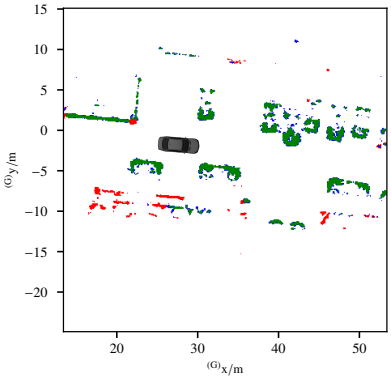
(b) Belegungskarte.



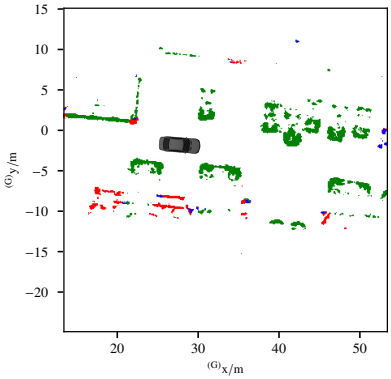
(c) Prädiktion mit Hintergrund.



(d) Prädiktion ohne Hintergrund.

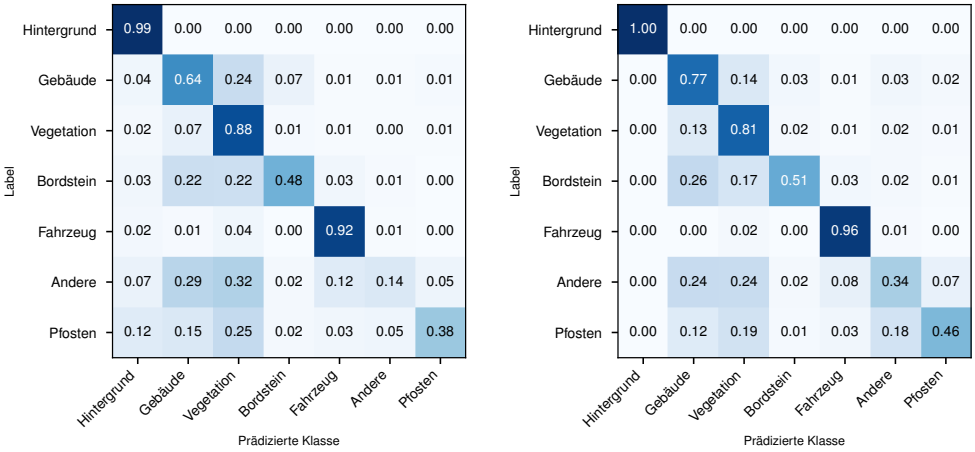


(e) Richtig/Falsch mit Hintergrund.



(f) Richtig/Falsch ohne Hintergrund.

Abbildung 6.9: Gegenüberstellung des Trainings mit Hintergrund und ohne Hintergrund. Die Richtig/Falsch Karte stellt das Klassifikationsergebnis wie folgt dar. Grün: Richtig Klassenzuordnung, Blau: Die richtige Klasse befindet sich in den Top-2-Prädiktionen, Rot: Die richtige Klasse ist nicht in den Top-2-Prädiktionen. Alle Zellen die unter dem Schwellwert τ_{valid} liegen werden in weiß dargestellt und damit nicht gewertet.



(a) Kreuzentropie (b) Kreuzentropie o. Hinterg.

Abbildung 6.10: Vergleich der Konfusionsmatrizen der Methoden a) (mit Hintergrund) und c) (ohne Hintergrund).

Tabelle 6.2 zeigt die Ergebnisse des Experiments. Hierbei wird sowohl die Makro- wie auch die Mikro-IoU angegeben. Da die Anzahl der belegten Zellen zwischen den einzelnen Karten stark variiert, wurden diese zusammen für alle Beispiele des Testdatensatzes berechnet.

In Abbildung 6.7 wird die IoU der unterschiedlichen Klassen separat dargestellt. Hierbei wurde über die Ergebnisse der einzelnen Trainingsdurchgänge gemittelt. Um die Grafik übersichtlich zu halten, wird die Standardabweichung der Ergebnisse nicht dargestellt. Diese ist jedoch außer bei e) und f) kleiner als 0.05.

Klar zu sehen ist, dass die Kostenfunktionen basierend auf der IoU das Schlusslicht einnehmen. Die Trainings mit dieser erwiesen sich am Anfang vielversprechend, jedoch stürzte ihre Performance im Bereich der 20. bis 100. Epochen ab. Danach wurden alle Zellen einer Karte auf eine Klasse abgebildet. Dieses lokale Optimum wurde, während des weiteren Trainingsverlaufs, in der Regel nicht mehr verlassen. Vermutlich führt die Linearität der *Intersection*, die durch eine reine Multiplikation bestimmt wird, zu einer zu geringen Bestrafung von starken Abweichungen. Eine Lösung könnte evtl. darin bestehen, die Softmax Aktivierungsfunktion durch eine geeignetere zu ersetzen.

Da beim Mikro-IoU die Verteilung der Klassen voll mit eingehen, ist zu erwarten, dass die Kostenfunktionen, die keine Gewichtung beinhalten, besser abschneiden, da sowohl beim Training wie auch bei der Auswertung die dominierenden Klassen

bevorzugt werden. Dieser Effekt kann bei der Kostenfunktion a) und b) beobachtet werden. Bei c) und d) schneiden die Klassifikatoren ähnlich ab.

An der Klasse *Pfosten*, der Klasse mit den wenigsten Zellen, sieht man, dass die IoU schlechter ist als bei den nicht gewichteten Beispielen, obwohl, wie in Abbildung 6.8 zu sehen, die Sensitivität der Klasse Pole erheblich ansteigt. Dies liegt daran das die Mehrheitsklassen durch die Gewichtung zwar nur leicht schlechter werden, aber dadurch viele Vertauschungen mit der Klasse Pole entstehen. Da die Konfusionsmatrize zeilenweise normiert wird, fällt die große Anzahl der Vertauschungen nicht auf.

Bei den nicht gewichteten Klassifikatoren werden die Klassen mit großer Fläche, wie z. B. *Vegetation* klar bevorzugt. Dies ist auch im Vergleich der Konfusionsmatrizen in Abb.6.8 zu sehen. Insgesamt lässt sich jedoch die Makro-IoU durch die Gewichtung leicht verbessern.

Das Label *Hintergrund* bedeutet nicht immer eine Abwesenheit von Objekten, es kann auch sein, dass die Bereiche der Belegungskarte nicht lange genug beobachtet wurde. Dies Label wird immer dann vergeben, wenn bei der Belegungskarte der Wert der Zelle unter dem Schwellwert τ_{valid} liegt. Hierbei stellt sich die Frage, wie schwer es einem Klassifikator fällt, diesen Schwellwert zu lernen und diesen durch die hoch- und runterskalierenden Schichten des neuronalen Netzes zu projizieren. Um den dadurch entstandenen Aufwand des neuronalen Netzwerkes zu reduzieren, kann dieser Schritt auch im Nachhinein auf das Klassifikationsergebnis angewendet werden, da die Klasse Hintergrund lediglich vom Schwellwert τ_{valid} abhängt. In Abb. 6.9 werden zwei Klassifikatoren gegenübergestellt. In Abb. 6.10 werden die dementsprechenden Konfusionsmatrizen dargestellt.

In der Konfusionsmatrize 6.10a ist zu sehen, dass das Klassifikationsergebnis der Klasse *Hintergrund* sehr gut ist. Dies ist nicht verwunderlich, da diese die mit Abstand weitverbreiteste Klasse ist. Es ist jedoch zu sehen, dass andere Klassen mit der Klasse *Hintergrund* verwechselt werden. Dies betrifft vor allem Bereiche deren Belegung in der Nähe von τ_{valid} liegt, welches naturgemäß am Rand der Objekte liegt. Daher ist zum Beispiel bei der Klasse *Pfosten* die Vertauschung höher als bei großflächigeren Klassen, wie der Klasse *Fahrzeug*.

Das in Abb. 6.9 gezeigte Beispiel soll dies noch einmal verdeutlichen. Die Vertauschung in den Randbereichen ist auch in diesem gezeigten Beispiel zu beobachten. In Abb. 6.9b ist noch mal die Belegungskarte gezeigt. Abb. 6.9c zeigt das Ergebnis den Klassifikator welcher den Hintergrund mit lernt, während Abb. 6.9d das Ergebnis des Klassifikators zeigt, welcher die Hintergrundklasse ignoriert. Aus Implementierungsgründen wurde dabei die Hintergrundklasse nicht entfernt, sondern lediglich nicht in die Kostenfunktion mit einbezogen. Dies erklärt, warum auch in dieser Abbildung

in einigen Bereichen die Hintergrundklasse zu finden ist. Die fehlende Optimierung auf den Hintergrund führt dazu, dass auch in nicht belegten Bereichen die Zellen zu einer der übrigen Klassen assoziiert werden.

In Abb. 6.9e und 6.9e ist nun die Korrektheit der Klassifikatoren zu sehen. Dieser wurde jedoch lediglich für den Bereich abgebildet, der in der Belegungskarte über τ_{valid} liegt. Dabei wird in Grün das richtige Klassifikationsergebnis angezeigt. Blau und Rot zeigen jeweils eine Fehlklassifikation an. Bei der blauen Farbe befindet sich das richtige Klassifikationsergebnis an Position zwei.

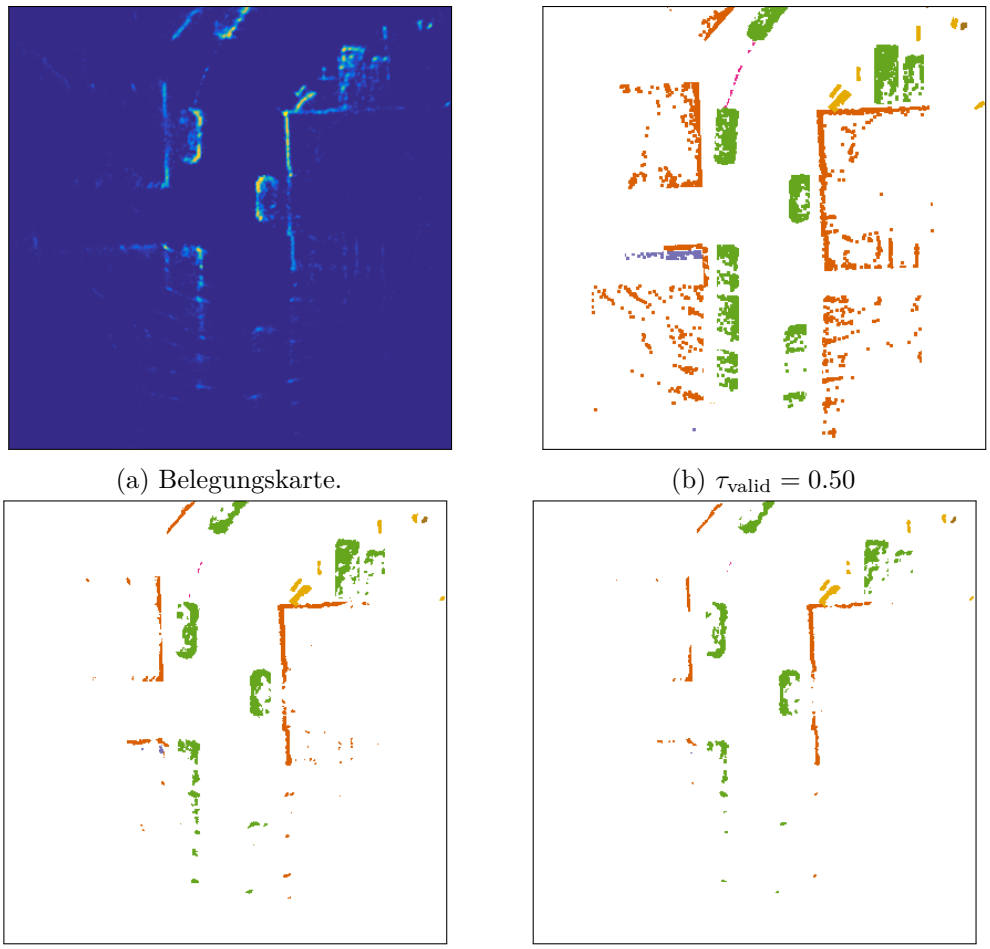
In den nachfolgenden Experimenten werden die Klassen immer ohne Hintergrund trainiert. Da die IoU der Klassen unabhängig von ihrem Vorkommen trainiert werden sollen, wird für das weitere Training die Kostenfunktion mit der besten Makro-IoU gewählt. Wie in Abb. 6.7 zu sehen, verbessert diese die IoU bei allen Klassen, außer der Klasse *Pfosten*.

6.4.2 Einfluss des Gültigkeitsschwellwerts

Bei der Klassifikation von Belegungskarten hängt das Klassifikationsergebnis stark von der Ausprägung der Karte ab. Diese verbessert sich mit zunehmender Beobachtungszeit des Sensors. Im Folgenden soll der Einfluss des Schwellwerts τ_{valid} untersucht werden, welche sowohl auf dem Trainings- wie auch dem Testdatensatz variiert wird. Die im Training verwendeten Parameter sind in Tabelle 6.3 dargestellt. Zur Auswertung werden die Schwellwerte 0.55, 0.60, 0.65 herangezogen. Die Belegungskarte wird dabei sowohl als Eingangsdaten für das Netzwerk, als auch zum Bestimmen der Schwellwertmaske verwendet.

In Abb. 6.11 kann man einen Eindruck gewinnen, welchen Einfluss der Parameter τ_{valid} auf die Labelkarte hat. In Abb. 6.11a ist die Belegungskarte zu sehen, auf die der Schwellwert angewandt wird, in den anderen Bildern sind die Labelkarten zu sehen. Durch den Schwellwert wird auch die Anzahl der gelabelten Zellen beeinflusst. Dies ist in Abb. 6.12 dargestellt, dabei wird der Anteil der Zellen der Gesamtzahl der Zellen einer Karte gegenübergestellt. Wie man sieht, nimmt die Anzahl der Zellen je nach Klasse in einem unterschiedlichen Maß ab. Während bei der Erhöhung des Schwellwertes von 0.50 auf 0.65 bei der Klasse *Pfosten* noch ca. 15 % der Zellen übrig bleiben, sind es bei der Klasse *Bordstein* nur noch ca. 3 %.

Für dieses Experiment wurden wiederum 5 Klassifikatoren pro Schwellwert trainiert und diese anhand der 3 besten ausgewertet. Dieses Vorgehen war notwendig, da es besonders bei den niedrigen Schwellwerten starke Ausreißer nach unten gab. Diese Klassifikatoren wurden anschließend mit unterschiedlichen Schwellwerten anhand des Testdatensatzes ausgewertet.



(a) Belegungskarte.

(b) $\tau_{\text{valid}} = 0.50$

(c) $\tau_{\text{valid}} = 0.55$

(d) $\tau_{\text{valid}} = 0.60$

Abbildung 6.11: Labelkarte mit unterschiedlichen Schwellwerten.

Somit kann das Ergebnis der Klassifikatoren als Matrix dargestellt werden. Diese sind in Tabelle 6.4 als Mittelwert des Makro-IoU und in Tabelle 6.5 als Mittelwert des Mikro-IoU zu finden.

Zunächst soll der Trainingsschwellwert beurteilt werden. Hierzu müssen die Tabellen zunächst zeilenweise betrachtet werden. Unabhängig vom Evaluationsschwellwert ist das Maximum bei dem Trainingsschwellwert $\tau_{\text{valid}} = 0.6$ zu finden. Dies trifft sowohl für die Mikro-, als auch die Makro-IoU zu.

Beim Evaluationsschwellwert handelt sich um einen Designparameter, der die Sensitivität bestimmt, bei welcher der Klassifikator ausgewertet wird. Da die Objek-

Tabelle 6.3: Trainingsparameter

Parameter	Wert
Eingangsdaten	Belegungskarte
τ_{valid}	0.50 / 0.525 / 0.55 / 0.60 / 0.65
Minibatch Größe	5
Kartengröße	400×400
Kostenfunktion	Kreuzentropie gew. (o. Hingerg.) Gl. …
Optimierer	Adam
Lernrate	0.001
β_1	0.9
β_2	0.999
Training	500 Epochen a 200 Minibatches

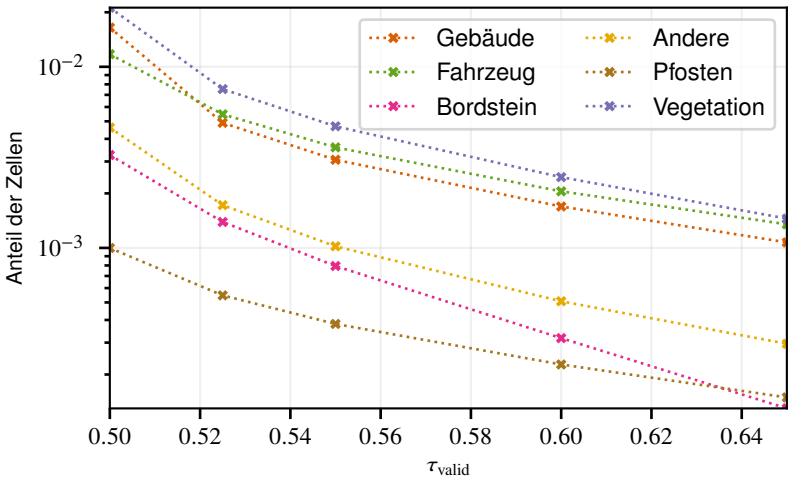


Abbildung 6.12: Anteil der gelabelten Zellen im Trainingsdatensatz abhängig von dem Schwellwert τ_{valid}. Dieses Verhältnis wurde anhand von 10000 zufällig gezogenen Beispielen bestimmt.

Tabelle 6.4: Makro-IoU

		Training τ_{valid}				
		0.500	**0.525**	**0.550**	**0.600**	**0.650**
Eval. τ_{valid}	**0.55**	0.476 ± 0.008	0.483 ± 0.021	0.486 ± 0.007	0.494 ± 0.006	0.464 ± 0.005
	0.60	0.483 ± 0.008	0.492 ± 0.019	0.497 ± 0.009	0.508 ± 0.006	0.481 ± 0.005
	0.65	0.482 ± 0.007	0.491 ± 0.017	0.496 ± 0.010	0.510 ± 0.008	0.482 ± 0.007
	0.70	0.475 ± 0.005	0.484 ± 0.015	0.490 ± 0.013	0.506 ± 0.011	0.475 ± 0.009

Tabelle 6.5: Mikro-IoU

		Training τ_{valid}				
		0.500	**0.525**	**0.550**	**0.600**	**0.650**
Eval. τ_{valid}	**0.55**	0.591 ± 0.023	0.596 ± 0.033	0.614 ± 0.020	0.620 ± 0.005	0.582 ± 0.013
	0.60	0.603 ± 0.025	0.607 ± 0.032	0.630 ± 0.022	0.639 ± 0.007	0.607 ± 0.013
	0.65	0.612 ± 0.024	0.614 ± 0.030	0.640 ± 0.024	0.649 ± 0.008	0.619 ± 0.015
	0.70	0.621 ± 0.022	0.621 ± 0.030	0.649 ± 0.026	0.658 ± 0.009	0.626 ± 0.017

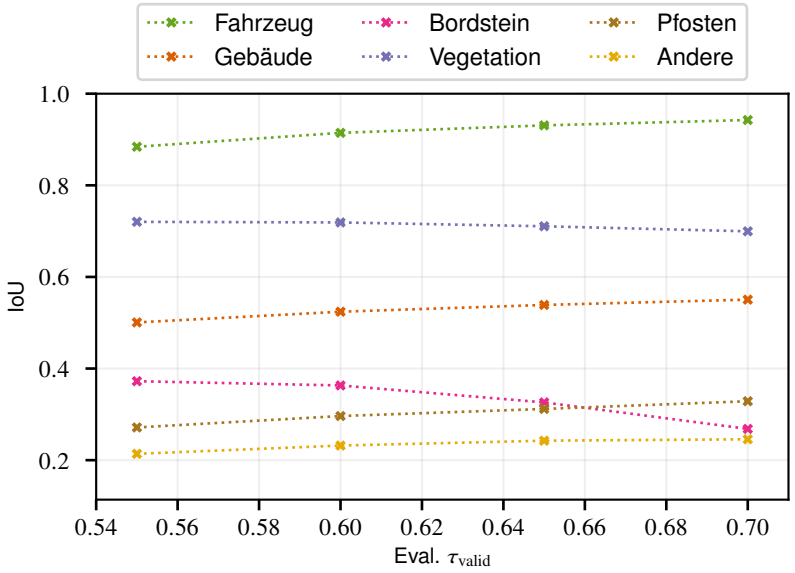


Abbildung 6.13: IoU in Abhängigkeit zu τ_{valid} der Evaluation. Hierbei kam ein Klassifikator zum Einsatz, der mit dem Schwellwert $\tau_{\text{valid}} = 0.6$ trainiert wurde.

te existieren, bevor sie beobachtet werden und durch eine einzelne Messung keine Klassifikation möglich ist, lässt sich dieser nur qualitativ bestimmen. Dazu wird in Abbildung 6.13 die IoU der einzelnen Klassen je nach Schwellwert in der Evaluation gegenübergestellt. Zu beobachten ist dabei, dass mit zunehmendem Schwellwert die IoU die Klassen *Bordstein* und *Vegetation* schlechter werden, während sich die IoU bei allen anderen Klassen verbessert.

Bei der Klasse *Bordstein* handelt es sich eher um eine Klasse, die schlecht reflektierende Oberflächen repräsentiert. Betrachtet man die Konfusionsmatrizen, welche hier aus Platzgründen nicht dargestellt sind, sieht man, dass mit zunehmendem Evaluationsschwellwert die Konfusion mit der Klasse *Gebäude* steigt. Dies lässt sich dadurch erklären, dass im Datensatz ein großer Teil der Bordsteine als *Hintergrund* erklärt werden und nur noch sehr gut reflektierende Bordsteine übrig bleiben (siehe Abb. 6.11). Diese starken Reflexionen charakterisieren sich dadurch, dass das Fahrzeug in langsamer Geschwindigkeit im richtigen Abstand und Winkel am Bordstein vorbeigefahren ist. Hierdurch bekommt der Bordstein eine ähnliche Ausprägung wie ein Gebäude. Da in dieser Auswertung der RCS-Wert nicht berücksichtigt wurde, kann dieser nicht zur Unterscheidung genutzt werden (siehe Abb. 4.18).

Bei der Klasse *Vegetation* ist der Effekt deutlich geringer. Auch hier nimmt im Verhältnis die Vertauschung mit der Klasse *Gebäude* zu. Erfahrungsgemäß lässt sich zumindest ein Teil davon auf Labelfehler zurückführen, bei dem ein Labeler einen in einer Hecke eingewachsenen Zaun, der zur Klasse *Gebäude* gehört, übersehen hat. Ein anderes typisches Problem ist, wenn sich eine Gartenmauer und eine Hecke vertikal überlagern. Durch einen höheren Schwellwert tauchen diese Problemfälle im Verhältnis öfter auf.

Bei den anderen Klassen tritt genau der gegenteilige Effekt ein. Objekte, die sich in weiter Entfernung befinden oder durch Verdeckungen schlecht beobachtbar sind, werden nicht mehr berücksichtigt. Ebenso nimmt der Randbereich ab, wie z. B. bei einem Pfosten der auf einer Wiese steht. Hier wird, je nach Labeler, ein unterschiedlich großer Kreis an Zellen dem Objekt zugeordnet. Da durch die Varianz der Messungen die Belegungskarte verwischt, ist diese Trennung auch als Mensch fehlerbehaftet. Diese könnte beispielsweise durch einen hochauflösenden Laserscanner, welcher als Referenz für das Labeln verwendet wird, verbessert werden.

In dieser Auswertung wurde der Effekt des Schwellwerts τ_{valid}, ab wann eine Zelle einer Klasse zugeordnet wird, gezeigt. Abhängig vom gewünschten Anwendungsfall ist dieser Parameter sinnvoll zu wählen. Es konnte jedoch gezeigt werden, dass durch die Wahl eines geeigneten Schwellwerts für das Training das Ergebnis der Klassifikation verbessert werden kann. Im Folgenden wird dieser Schwellwert sowohl für das Training, als auch für die Auswertung verwendet.

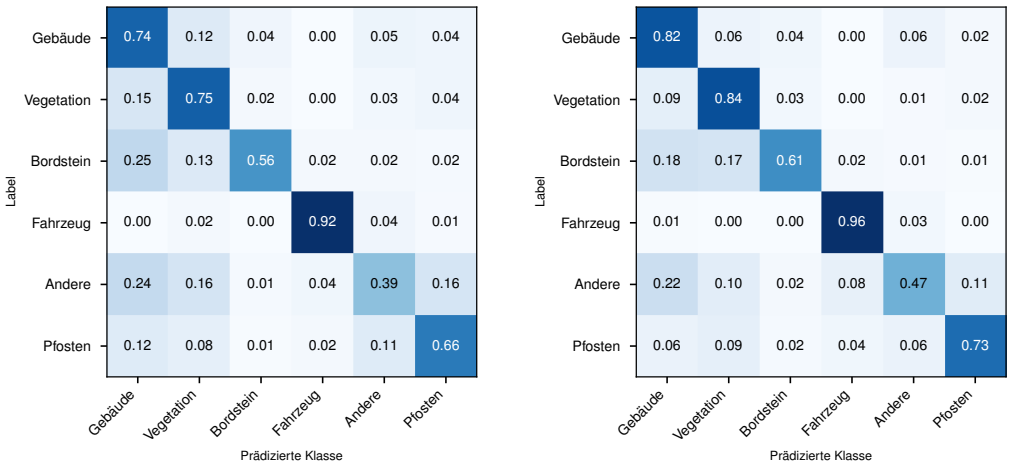
6.4.3 Vergleich Radar-Merkmalskarten

Tabelle 6.6: Trainingsparameter

Parameter	Wert
Eingangsdaten	Belegungskarte/RCS-Histogramm-Karte
τ_{valid}	0.60
Minibatch Größe	5
Kartengröße	400×400
Kostenfunktion	Kreuzentropie gew. (o. Hingerg.) Gl. …
Optimierer	Adam
Lernrate	0.001
β_1	0.9
β_2	0.999
Training	1000 Epochen a 200 Minibatches

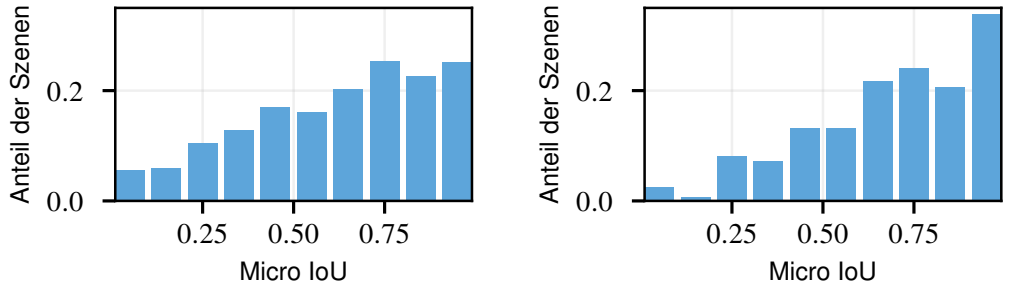
Tabelle 6.7: Mittlere IoU der verschiedenen Trainingsdurchgänge.

	Belegungskarte	RCS-Histogramm-Karte
Makro-IoU	0.51 ± 0.00	0.57 ± 0.01
Mikro-IoU	0.63 ± 0.00	0.71 ± 0.01
IoU Gebäude	0.54 ± 0.01	0.62 ± 0.01
IoU Fahrzeug	0.91 ± 0.00	0.93 ± 0.01
IoU Bordstein	0.38 ± 0.02	0.40 ± 0.01
IoU Andere	0.23 ± 0.01	0.29 ± 0.01
IoU Pfosten	0.33 ± 0.02	0.41 ± 0.02
IoU Vegetation	0.69 ± 0.00	0.79 ± 0.01



(a) Belegungskarte. (b) RCS-Histogramm-Karte.

Abbildung 6.14: Konfusionsmatrix eines Klassifikators auf dem Test-Set auf Basis von Zellen.



(a) Belegungskarte. (b) RCS-Histogramm-Karte.

Abbildung 6.15: Histogramm IoU pro Szene eines Klassifikator.

In diesem Versuch sollen noch einmal zwei verschiedene Merkmalskarten für die Radar-Klassifikation gegenübergestellt werden, die Belegungskarte und die RCS-Histogramm-Karte. Die Belegungskarte, da es sich hierbei um einen weitverbreiteten Ansatz zur Umgebungserfassung handelt. Die RCS-Histogramm-Karte da sich diese in Abschnitt 5.5.3 als eine der erfolgreichsten Ansätze herausgestellt hat.

In Tabelle 6.6 sind die wichtigsten Trainingsparameter dargestellt. Hierbei wurden die Netzwerke doppelt so lange trainiert wie bei den vorherigen Experimenten. Es wurden jeweils 5 Trainingsdurchgänge pro Merkmalskarte durchgeführt. Anhand des Fehlers auf dem Validationsdatensatzes wurde der beste Klassifikator eines jeden

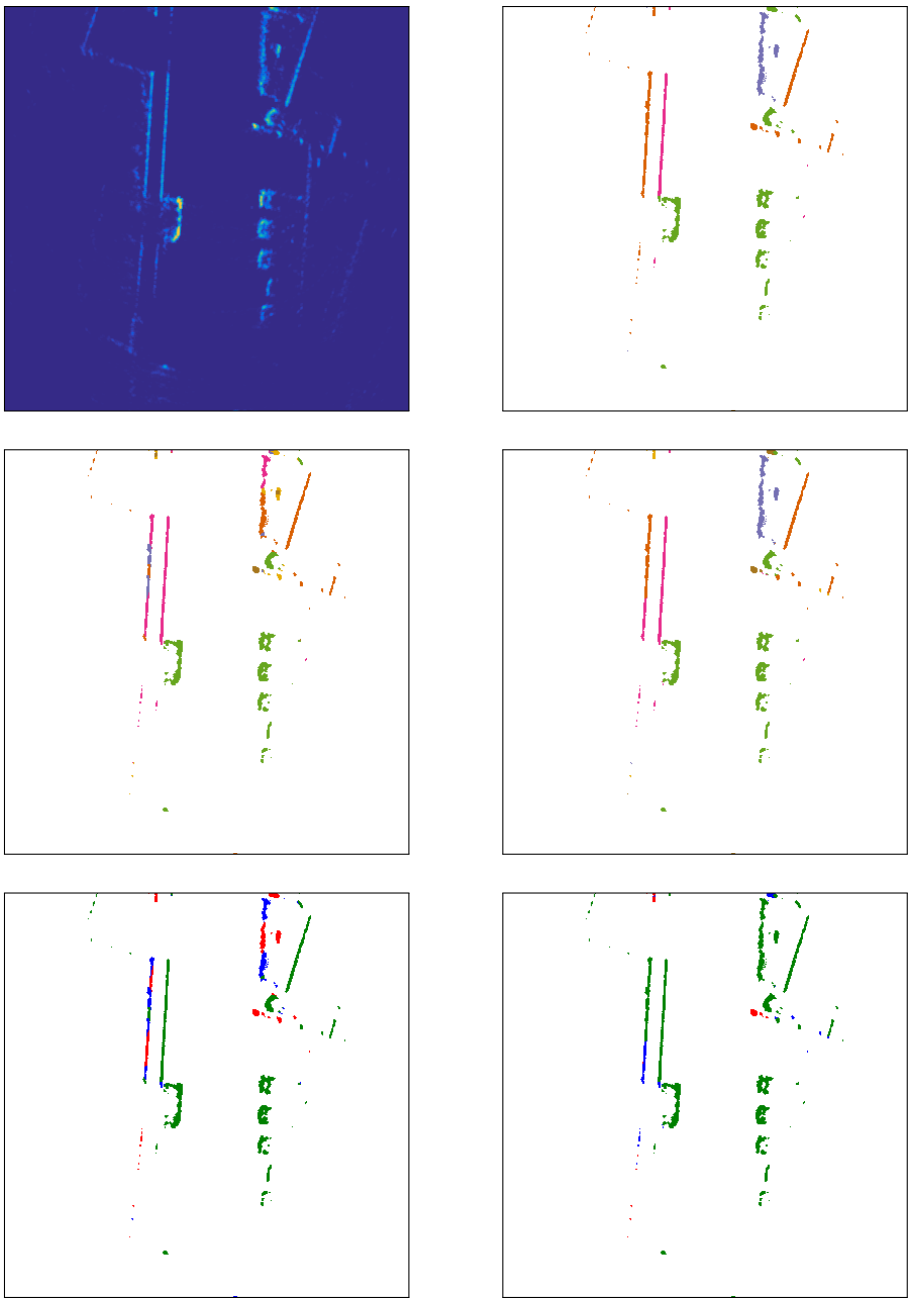


Abbildung 6.16: Vergleich der Klassifikatoren. 1. Zeile: Belegungskarte und das Labelergebnis. 2. Zeile: Klassifikationsergebnis der Belegungskarte und der RCS-Histogramm-Karte. 3. Zeile: Visualisierung des Ergebnisses (Grün: Richtige Klassenzuordnung, Blau: Richtige Klasse auf Platz 2, Rot: Die richtige Klasse ist nicht in den Top 2 Prädiktionen.)

Durchgangs selektiert und die besten 3 auf dem Testdatensatz ausgewertet.

Die Ergebnisse sind in Tab. 6.7 zusammengefasst. Hierbei ist zu erkennen, dass die RCS-Histogramm-Karte eine bessere IoU erreicht. Dies trifft sowohl für die Mikro-IoU als auch die Makro-IoU zu. Die Verbesserung zieht sich durch alle Klassen, sowohl in IoU als auch in Genauigkeit, welche in der Konfusionsmatrix in Abb. 6.14 zu sehen ist. Damit bestätigt sich das Ergebnis der Cluster basierenden Evaluierung.

In Abb. 6.15 ist die Verteilung der IoUs über die verschiedenen Karten im Trainingsdatensatz zu sehen. Da bei den meisten Karten nicht alle Klassen vertreten sind, wurde dafür die Mikro-IoU verwendet. Hierbei ist in allen Bereichen eine Verbesserung zu erkennen. Dies bestätigt sich auch bei der näheren Betrachtung der Beispiele.

In Abb. 6.16 wird anhand eines Beispiels ein Klassifikator, basierend auf der Belegungskarte und ein Klassifikator basierend auf der RCS-Histogramm-Karte verglichen. Besonders bei Strukturen die sich in der Form sehr ähneln, wie z. B. ein Baumstamm und ein Metallpfosten gibt die zusätzliche RCS Information den entscheidenden Ausschlag bei der Klassifikation.

In diesen Experimenten konnte bestätigt werden, dass auch für die semantische Segmentierung durch die RCS-Histogramm-Karte eine Verbesserung des Klassifikationsergebnis zu erzielen ist, welche sich über alle Klassen erstreckt. Daher ist diese Repräsentation vielversprechend. Durch eine Optimierung des Wertebereichs der einzelnen RCS-Klassen ließe sich dieses Ergebnis aller Erwartung nach noch weiter optimieren.

6.5 Auswertung semantischer Segmentierung von geparkten Fahrzeugen

In diesem Experiment wird untersucht, ob es möglich ist geparkte Fahrzeuge zu erkennen und ihre Dimension und Ausrichtung zu schätzen. Hierzu wird der Datensatz aus Abschnitt 4.5.5 verwendet.

Für dieses Experiment wurde ein Netzwerk, wie im Abschnitt 6.3 beschrieben trainiert. Die Parameter des Trainings sind in Tabelle 6.8 zu sehen. Zu beachten ist dabei, dass der Schwellwert τ_{valid} für das gesamte Label gilt. Sobald eine Zelle in diesem Label diesen Schwellwert überschreitet, sind alle Zellen in diesem Label gültig. Gegenüber den vorherigen Experimenten kommt ein weiterer Faktor τ_{shrink} hinzu, der das ursprüngliche Label, bevor es angewendet wird verkleinert. Karten ohne gültige Label wurden für das Training nicht verwendet.

Tabelle 6.8: Trainingsparameter

Parameter	Wert
Eingangsdaten	Belegungskarte/RCS-Histogramm-Karte
τ_{valid}	0.60
τ_{shrink}	0.7
Minibatch Größe	2
Kartengröße	1000×1000
Kostenfunktion	L_{IoU} Gl. 6.7
Optimierer	Adam
Lernrate	$0.001 \ldots 0.01$
β_1	0.9
β_2	0.999
Training	$>$ 5000 Epochen

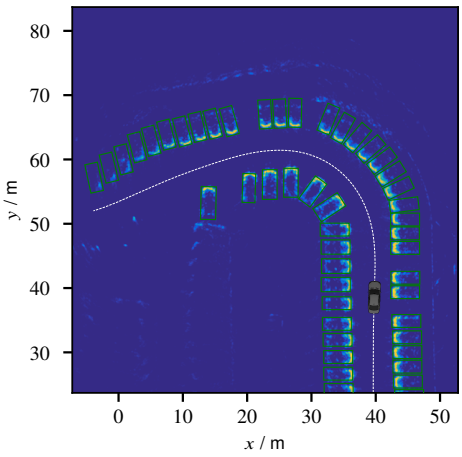
Anders als bei den vorherigen Experimenten wurde dieses Netzwerk über eine längere Zeit, in mehreren Abschnitten trainiert, wobei immer auf die vorherigen trainierten Gewichte zurückgegriffen wurde. Daher ist es nicht mehr möglich, den Trainingsverlauf und die Parameter des Optimierers exakt zu rekonstruieren.

Wie in Abbildung 6.17 dargestellt, werden anhand des Klassifikationsergebnisses Cluster extrahiert und Bounding Boxen gebildet. Auf diesen erfolgt nun die Auswertung.

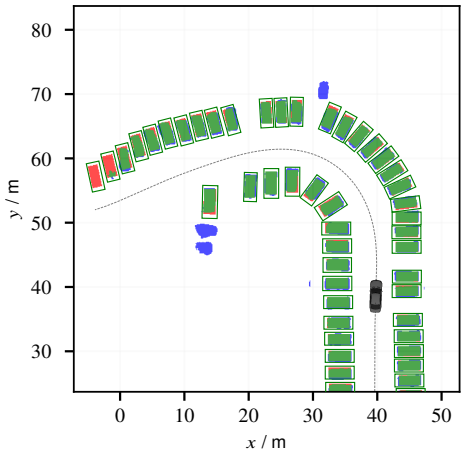
In Tabelle 6.9 sind die Ergebnisse des Klassifikators auf dem Testdatensatz dargestellt. Zudem werden in Abb. 6.18 noch einmal die wichtigsten Größen als Histogramm dargestellt. Dabei ist zu beachten, dass die Auswertung bezüglich der Label stattfindet, da eine genaue Messung der Objektdimensionen nicht vorhanden ist. Hierbei sei auch auf die Abschätzung der Labelgenauigkeit in Abschnitt 3.4 verwiesen.

Die ersten Maße sind in Form der Sensitivität und der Falscherkennungsrate dargestellt. Die Falscherkennungsrate ist dabei definiert wie folgt.

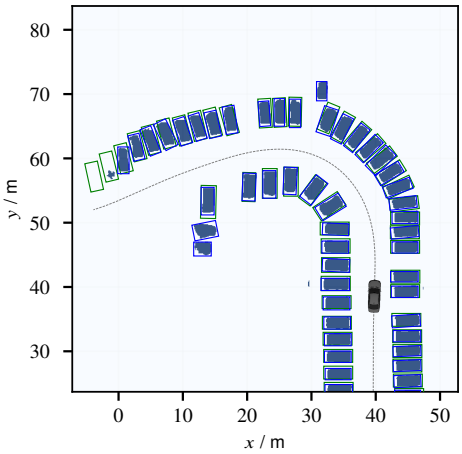
(a) Vordere Kamera des eingezeichneten Fahrzeugs.



(b) Belegungskarte mit Label.



(c) Zellenweise Evaluation.

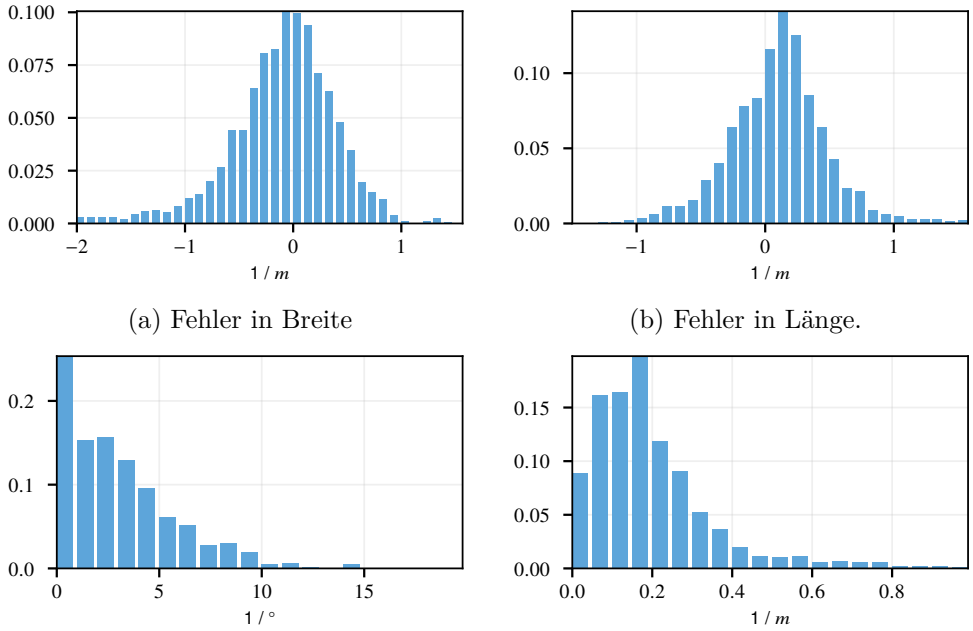


(d) Ausgabe des Klassifikators.

Abbildung 6.17: Beispiel einer Karte zum Segmentieren vollständiger Fahrzeuge. In (a) ist das Bild eines Szenarios zu sehen, indem Fahrzeuge in einem Bogen parken. In (b) zeigt die Belegungskarte mit den eingezeichneten Labeln. In (c) befindet sich die zellenweise Evaluierung des Klassifikators (grün: Richtig Positive, rot: Falsch Negativ, blau: Falsch Positiv). In (d) befindet sich die Ausgabe des Klassifikators mit den geschätzten Polygonen in blau und den wahren in grün.

Tabelle 6.9: Fehlermaße der extrahierten Fahrzeuge.

Fehlermaß	Wert
Sensitivität	0,94
Falscherkennungsrate	0,16
Position: Mittlere Abweichung	0,21 m
Orientierung: Mittler Abweichung	4,19°
Breite: Mittlere Abweichung	0,11 m
Breite: Standardabweichung	0,38 m
Länge: Mittlere Abweichung	−0,07 m
Länge: Standardabweichung	0,5 m



(a) Fehler in Breite

(b) Fehler in Länge.

(c) Fehler in Rotation

(d) Fehler in Position.

Abbildung 6.18: Histogramm der Fehler bei der Schätzung der Position, Größe und Orientierung.

$$\text{Falscherkennungsrate} = \frac{\text{Falsch Positiv}}{\text{Richtig Positiv} + \text{Falsch Positiv}} \tag{6.19}$$

Dabei stellt sich das Problem, dass teilweise Fahrzeuge von dem Klassifikator erkannt wurden, die aufgrund ihrer Abschattung nicht gelabelt worden sind (z. B. an Position (32 m, 70 m) in Abb. 6.17) oder die Aufgrund ihrer geringen Belegung noch nicht gültig sind (z. B. an Position (13 m, 47 m) in Abb. 6.17).

Die mittlere Abweichung ist die Entfernung vom Mittelpunkt des Labels zum Mittelpunkt des geschätzten Fahrzeuges. Um die Dimensionen zu evaluieren, wird jeweils die Länge und Breite des einhüllenden Rechtecks verwendet. Dabei wird die längere Seite der Box immer als Länge und die kürzere Seite als Breite interpretiert. Anhand dieser Ausrichtung wird die Orientierung bestimmt.

Wie zu erkennen ist, tendiert das Netzwerk leicht zum Überschätzen der Breite und zum Unterschätzen der Länge. Da die Belegungskarte an sich nur eine Auflösung von 0,1 m besitzt, ist dieser Fehler vernachlässigbar.

Betrachtet man die Verteilungen in Abbildung 6.18a und 6.18b so sieht man, dass diese annähernd normalverteilt sind. Die Standardabweichung ist dabei relativ hoch. Es muss jedoch beachtet werden, dass die Label selbst eine Varianz in einer ähnlichen Größenordnung aufweisen (siehe Abb. 3.7). Ferner ist die Ausdehnungsschätzung der vom Sensor abgewandten Seite des Objekts schwierig, da nur wenige Daten vorhanden sind.

Die Orientierung ist mit einer Abweichung von 4,19° hingegen relativ genau. Die Schätzung der Ausdehnung wird immer genauer, je länger das Objekt beobachtet wurde und je weiter sich der Beobachtungswinkel aufgespannt hat.

Insgesamt lässt sich ein positives Resultat ziehen, Ausmaße und Orientierung der Fahrzeuge werden meist gut erkannt. Auch wenn ein Fahrzeug nur unvollständig beobachtet wurde, ist der Lernalgorithmus in der Lage die Dimension des Fahrzeugs zu rekonstruieren. Trotz der großen Varianz der Labels ist der Klassifikator in der Lage dies zu lernen. Für eine genauere Bewertung wären Referenzmessungen sinnvoll, welche die wahre Größe und Pose des Fahrzeugs zur Verfügung stellen.

6.6 Zusammenfassung

In diesem Kapitel wurden verschiedene Arten der semantischen Segmentierung von Radar-Merkmalskarten gezeigt. Eine Zellen basierende Multi-Klassen Segmentierung

vorgestellt wurde vorgestellt, ebenso ein System zur Schätzung von Größe und Orientierung von geparkten Fahrzeugen

Bei der Multi-Klassen Segmentierung wurden erfolgreich mehrere Klassen voneinander unterschieden. Es wurde eine geeignete Kostenfunktion vorgestellt und verschiedene Eigenschaften, wie z. B. der Einfluss des Gültigkeitsschwellwerts auf das Klassifikationsergebnis, gezeigt.

Insgesamt konnte eine Makro-IoU von 0.57 erreicht werden. Beachtet man die Umstände des eingeschränkten Sensors (siehe Abschn. 4.1) und die Korrektheit der Labelergebnisse (Abschn. 3.4.1) sind diese Ergebnisse sehr vielversprechend. Durch den Einsatz von Augmentierungstechniken (Abschn. 5.5.4) und vortrainierten Netzen (Abschn. 5.5.2) könnten diese Ergebnisse weiter verbessert werden.

Die Klasse *Fahrzeug* stellt bei der semantischen Segmentierung die mit Abstand am leichtesten zu erkennende Klasse dar. Dies liegt vor allem an ihrer charakteristischen Form und der Eindeutigkeit der Klasse. Dies führt sowohl beim Labeling wie auch beim Klassifizieren zu guten Erkennungsraten.

Beim Segmentieren von ganzen Fahrzeugen wurde gezeigt, dass gute Ergebnisse erzielt werden können und der Klassifikator auch bei in der Belegungskarte spärlich aufgebauten Fahrzeugen in der Lage ist, eine gute Schätzung der Orientierung zu liefern. Besonders für Systeme zur Suche von freien Parklücken könnte sich dieses System gut eignen.

Kapitel 7

Zusammenfassung und Ausblick

In dieser Arbeit wurde erstmalig die semantische Klassifikation von Objekten aufgrund von Radar-Merkmalskarten untersucht. Dafür wurden zunächst die Merkmale eines Radarsensors vorgestellt, welche zur Klassifikation genutzt werden können. Diese wurden in Kapitel 2.5 qualitativ und in Kapitel 4.5 anhand einiger Aspekte quantitativ beleuchtet.

Diese Merkmale bestehen nach der Extraktion der Reflexionszentren mittels eines CFAR aus einer Punktwolke, bei dem jeder Punkt mithilfe des Azimut Winkel und der Distanz im Raum lokalisiert werden kann. Zudem wird die elektromagnetische Reflektivität der Ziele durch den RCS-Wert dargestellt. In Abschnitt 4.5.2 wurde gezeigt, dass sich Klassen über die statistische Verteilung des RCS-Wertes bewerten lassen. Die Möglichkeit, die Doppler-Geschwindigkeit direkt zu messen, erlaubt eine leichte Einteilung in die dynamische und statische Welt. Für die statische Welt haben diese Merkmale aber keine weitere Bedeutung.

In Kapitel 2.4 wurden verschiedene Merkmalskarten vorgestellt, mit denen sich diese Informationen über die Zeit akkumulieren lassen. Durch dieses Filtern der Daten verlieren Messrauschen und Ausreißer an Gewicht. Zudem werden die Formen der Objekte besser dargestellt.

Da für das semantische Klassifizieren von Objekten gelabelte Daten benötigt werden, wurde in Kapitel 3 eine effiziente Methode zum Labeln der Daten entwickelt. Diese ermöglicht es, Objekte innerhalb eines Zeitschritts zu labeln und dieses Label dann auf sämtliche Messungen zu übertragen.

Um die Qualität des Labelings beurteilen zu können, wurde dies quantitativ untersucht, indem verschiedene Labeler gegenübergestellt wurden. Hier hat es sich gezeigt, dass die Qualität zum Teil sehr schwankt. Für weitere Projekte sollte eine erhöhte Aufmerksamkeit auf den Qualitätsprozess des Labeling gelegt werden.

In Kapitel 5 wurden verschiedene Methoden und weitere Aspekte der Klassifikation für Radardaten untersucht. Hier wurde beispielsweise gezeigt, dass das Klassifikationsergebnis verbessert werden kann, indem die Neuronalen Netzwerke auf Fotos vortrainiert und danach mit Radardaten nachtrainiert werden.

Des Weiteren wurde ein System zur Augmentierung von Radardaten vorgestellt, welches mit einem evolutionären Algorithmus die Parameter der Augmentierung lernt.

Es wurde der Vergleich von Faltungsnetzen mit einem klassischen Klassifikationsverfahren mit Merkmalsextraktoren und einem Random Forest gezogen. Obwohl dieses schlechter als das CNN abschneidet, konnte gezeigt werden, dass ein Ensemble aus diesen zwei Methoden eine Verbesserung der Klassifikation bringt. Bei den Merkmalskarten stellte sich heraus, dass die RCS-Histogramm-Karte von den vorgestellten Karten die besten Ergebnisse erzielt.

Abschließend wurde im Kapitel 6 die semantische Segmentierung von Radar-Merkmalskarten vorgestellt. Es wurden verschiedene Kostenfunktionen evaluiert und aufgezeigt, dass durch eine separate Filterung der Klasse *Hintergrund* das Klassifikationsergebnis verbessert werden kann. In einer weiteren Methode wurde dargestellt, dass man anhand der Belegungskarten gut die Größe und Ausrichtung von Fahrzeugen schätzen kann, auch wenn in der Belegungskarte nur Teile von ihnen zu sehen sind.

Die Arbeit konnte zeigen, dass eine semantische Klassifikation von Objekten in Radar möglich ist. Die großteils aus der Bildverarbeitung entlehnten Klassifikationsverfahren wurden dabei auf Radar-Merkmalskarten angewendet. Wie zu erwarten, blieben bei diesem vergleichsweisen neuen Thema viele Fragen noch unbeantwortet.

Durch den in der Arbeit eingesetzten Radarsensor, war die Anzahl an Zielen pro Messzyklus stark eingeschränkt. Dies führte dazu, dass in belegten Regionen, schwache Ziele nicht mehr detektiert wurden. Dies kann je nach Umgebung, einen inkonsistenten Aufbau der Belegungskarten verursachen.

Des Weiteren war der Sensor nicht in der Lage die Elevation der Reflexionszentren zu messen. Dadurch konnten Objekte, die vertikal übereinander lagen, wie z. B. eine Gartenmauer auf der eine Hecke wächst, nicht unterschieden werden. Zudem wird

mit der Elevation ein weiteres wichtiges Merkmal zur Unterscheidung von Objektklassen zur Verfügung gestellt. Mit der neuen Generation von Radarsensoren im Automobil, steht dieses Merkmal zur Verfügung.

Um die Komplexität nicht unnötig zu steigern, wurde in dieser Arbeit ein sehr einfaches Sensormodell für die Radar-Belegungskarten verwendet. Ein geeigneteres Modell könnte die Qualität der Karten verbessern, indem Geschwindigkeit des Eigenfahrzeuges, Verdeckungen und die Sensoreigenschaften berücksichtigt werden.

Eine Möglichkeit wäre, das entsprechende Sensormodel zu lernen. Hierzu gibt es beispielsweise Arbeiten, die das Sensormodel mithilfe eines Laserscanners lernen [Sle+19]. Eine andere Möglichkeit ist es, eine geeignete Merkmalsrepräsentation direkt mithilfe des gelabelten Datensatzes zu lernen. Erste Untersuchungen in diese Richtung, erwiesen sich als erfolgversprechend.

Die Repräsentation von Objekten in Radar-Belegungskarten ist stark abhängig von dem Beobachtungszeitraum und dem Winkel- und Entfernungsbereich, aus dem beobachtet wurde. So kann beispielsweise ein Fahrzeug in großer Entfernung eine ähnliche Repräsentation in der Karte einnehmen, wie ein länger beobachteter Pfosten. Falls dem Klassifikator die Information über die Beobachtung zur Verfügung gestellt würde, kann die Klassifikation verbessert werden. Zudem könnte durch eine solche Information, eine Schätzung der Zuverlässigkeit des Klassifikationsergebnisses erfolgen.

Neben der Größe des Datensatzes sollte auch die Qualität des Labeling weiter erhöht werden und das Labeling noch besser auf die Bedürfnisse von Belegungskarten angepasst werden. Hierzu zählt beispielsweise ein überlagertes Labeln, sodass bei Objekten, wie z. B. ein Busch auf einer Mauer, beide Objekte gelabelt werden können.

Ebenso ist es wichtig, die Taxonomie weiter zu entwickeln. Vor allem sollte die Anzahl an Objekten, die in die Klasse *Andere* fallen, minimiert werden.

Um den Datensatz massiv zu vergrößern, könnten sich automatische Labelingkonzepte, bei denen das Label von der Kamera oder einem Laserscanner transferiert werden, eignen. Hierbei müssen jedoch radarspezifische Besonderheiten, wie z. B. das Tunneln der Welle unter einem Auto, berücksichtigt werden.

Ein weiteres Konzept welches bei Cluster basierenden Klassifikationsverfahren erfolgreich untersucht wurde, ist das raum-zeitliche Klassifizieren von Radargrids. Hierbei kann über den Aufbau der Radar-Merkmalskarten, über die Zeit, weitere Schlüsse auf ihre Objektklasse gezogen werden. Hierzu können rekurrente Netze wie z. B. LSTMs eingesetzt werden.

Eigene Veröffentlichungen

[Lom+15] Jakob Lombacher, Markus Hahn, Jürgen Dickmann und Christian Wöhler. „Detection of arbitrarily rotated parked cars based on radar sensors“. In: *Proc. Int. Radar Symp.* 2015. ISBN: 9783954048533.

[Lom+16] Jakob Lombacher, Markus Hahn, Jürgen Dickmann und Christian Wöhler. „Potential of radar for static object classification using deep learning methods“. In: *2016 IEEE MTT-S Int. Conf. Microwaves Intell. Mobility, ICMIM 2016*. 2016. ISBN: 9781509023677.

[Lom+17a] Jakob Lombacher, Markus Hahn, Jürgen Dickmann und Christian Wöhler. „Object Classification in Radar Using Ensemble Methods“. In: *IEEE MTT-S Int. Conf. Microwaves Intell. Mobil.* 2017. ISBN: 9781509043545.

[Lom+17b] Jakob Lombacher, Kilian Laudt, Markus Hahn, Jurgen Dickmann und Christian Wöhler. „Semantic radar grids“. In: *IEEE Intell. Veh. Symp. Proc.* 2017, S. 1170–1175. ISBN: 9781509048045.

[Lup+17] Stefanie Lupfer, Matthias Rapp, Klaus Dietmayer, Peter Brosseit, Jakob Lombacher, Markus Hahn und Jurgen Dickmann. „Increasing FastSLAM accuracy for radar data by integrating the Doppler information“. In: *2017 IEEE MTT-S Int. Conf. Microwaves Intell. Mobility, ICMIM 2017*. 2017, S. 103–106. ISBN: 9781509043545.

[Rap+16b] Matthias Rapp, Klaus Dietmayer, Markus Hahn, Frank Schuster, Jakob Lombacher und Jürgen Dickmann. „FSCD and BASD: Robust landmark detection and description on radar-based grids“. In: *IEEE MTT-S Int. Conf. Microwaves Intell. Mobil.* 2016. ISBN: 9781509023677.

[Sch+19] Ole Schumann, Jakob Lombacher, Markus Hahn, Christian Wohler und Jürgen Dickmann. „Scene Understanding with Automotive Radar“. In: *IEEE Trans. Intell. Veh.* 8858.c (2019), S. 1. ISSN: 2379-8858 VO -.

[Win+17] Timo Winterling, Jakob Lombacher, Markus Hahn, Jürgen Dickmann und Christian Wöhler. „Optimizing Labelling on Radar Based Grid Maps Using Active Learning“. In: *IEEE MTT-S Int. Conf. Microwaves Intell. Mobil.* 2017.

Betreute Masterarbeiten

[Kil16] Kilian Laudt. „Scene Classification On Radar Based Grid Maps Using-Convolutional Neural Networks“. M.Sc. thesis. Universität Koblenz-Landau, 2016.

[Luc17] Florin Lucaciu. „Design and implementation of modell-based method for the detection and orientation estimation of parked vehicles using radar sensors“. M.Sc. thesis. Universität Stuttgart, 2017.

[Rom17] Marouane Ben Romdhane. „Spatio-temporal Classification of Radar Grid Maps“. M.Sc. thesis. Technische Universität München, 2017.

[Ser15] Alexander Serguchev. „Design and implementation of model-based methods for the detection and orientation estimation of parked vehicles using radar sensors“. M.Sc. thesis. Universität Ulm, 2015.

[Wel18] Tobias Welz. „Extracting Sematic Radar Feature Grids using Recurrent Deep Convolutional Neural Networks“. M.Sc. thesis. Universität Ulm, 2018.

[Win16] Timo Winterling. „Semi Supervised Classification of Radar Based Grid Maps Using Deep Learning Methods“. M.Sc. thesis. Universität Ulm, 2016.

[Zha16] Xiaofeng Zhang. „Dynamic Object Classification Based on High Resolution Doppler Radar and Laser Data“. M.Sc. thesis. Universität Stuttgart, 2016.

Literatur

[Aba+15] Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Man, Rajat Monga, Sherry Moore, Derek Murray, Jon Shlens, Benoit Steiner, Ilya Sutskever, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Oriol Vinyals, Pete Warden, Martin Wicke, Yuan Yu und Xiaoqiang Zheng. *TensorFlow : Large-Scale Machine Learning on Heterogeneous Distributed Systems*. 2015.

[ACS12] Cutler Adele, D. Richard Cutler und John R. Stevens. „Random forests“. In: *Ensemble Mach. Learn. Methods Appl.* Hrsg. von Cha Zhang und Yunqian Ma. Springer US, 2012, S. 157–175. ISBN: 978-1-4419-9325-0.

[Aht+15] Juhana Ahtiainen, Thierry Peynot, Jari Saarinen, Steven Scheding und Arto Visala. „Learned Ultra-Wideband RADAR Sensor Model for Augmented LIDAR-based Traversability Mapping in Vegetated Environments“. In: *18th Int. Conf. Inf. Fusion*. 2015, S. 953–960. ISBN: 9780996452717.

[And09] Franz Andert. „Drawing stereo disparity images into occupancy grids: Measurement model and fast implementation“. In: *2009 IEEE/RSJ Int. Conf. Intell. Robot. Syst.* (2009), S. 5191–5197.

[App79] Jørgen Appel-Hansen. „Accurate Determination of Gain and Radiation Patterns by Radar Cross-Section Measurements“. In: *IEEE Trans. Antennas Propag.* 27.5 (1979), S. 640–646. ISSN: 15582221.

[ASL08] T. L. Ainsworth, D. L. Schuler und J. S. Lee. „Polarimetric SAR characterization of man-made structures in urban areas using normalized circular-pol correlation coefficients“. In: *Remote Sens. Environ.* 112.6 (2008), S. 2876–2885.

[Bar80] P Barton. „Digital Beam Forming for Radar“. In: *Commun. Radar Signal Process. IEE Proc.* Bd. 127. 4. 1980, S. 266–277. ISBN: 0863410219.

[BFP09] Hernán Badino, Uwe Franke und David Pfeiffer. „The Stixel World -A Compact Medium Level Representation of the 3D-World“. In: *Jt. Pattern Recognit. Symp.* 2009, S. 1–10.

[BHB16] Wolfram Burgard, Martial Hebert und Maren Bennewitz. „World Modeling“. In: *Springer Handb. Robot.* 2016, S. 1135–1152. ISBN: 978-3-319-32552-1.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 978-0-387-31073-2.

[BK91] Johann Borenstein und Yoram Koren. „The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots“. In: *IEEE Trans. Robot. Autom.* 7.3 (1991), S. 278–288.

[BPM04] Gustavo Batista, Ronaldo C. Prati und Maria Carolina Monard. „A study of the behavior of several methods for balancing machine learning training data“. In: *ACM SIGKDD Explor. Newsl.* 6.1 (2004), S. 20.

[Bra+06] Christophe Braillon, Cédric Pradalier, Kane Usher, Jim Crowley und Christian Laugier. „Occupancy grids from stereo and optical flow data“. In: *Int. Symp. Exp. Robot.* 2006.

[Bre01] Leo Breiman. „Random forests“. In: *Mach. Learn.* 45.1 (2001), S. 5–32. ISSN: 08856125.

[Bro05] Graham M Brooker. „Understanding Millimetre Wave FMCW Radars“. In: *1st Int. Conf. Sens. Technol.* 2 (2005), S. 152–157.

[BT08] W. G. Bath und G. V. Trunk. „Automatic Detection, Tracking, and Sensor Integration“. In: *Radar Handb.* 2008.

[Bun92] Wray Lindsay Buntine. „A Theory of Learning Classification Rules“. Diss. 1992.

[Cae+20] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan und Oscar Beijbom. „nuScenes: A Multimodal Dataset for Autonomous Driving“. In: March (2020), S. 11618–11628.

[CB91] Peter Clark und Robin Boswell. „Rule induction with CN2: Some recent improvements“. In: *Lect. Notes Comput. Sci. (Lecture Notes Artif. Intell.* 482 (1991). ISSN: 16113349.

[Che+06] M. Cherniakov, R. S.A.R. Abdullah, P. Jančovič, M. Salous und V. Chapursky. „Automatic ground target classification using forward scattering radar“. In: *IEE Proc. Radar, Sonar Navig.* 153.5 (2006), S. 427–437. ISSN: 13502395.

[Che+16] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler und Raquel Urtasun. „Monocular 3D Object Detection for Autonomous Driving“. In: *2016 IEEE Conf. Comput. Vis. Pattern Recognit.* (2016), S. 2147–2156. ISSN: 10636919.

[Che+17a] Liang-Chieh Chen, George Papandreou, Florian Schroff und Hartwig Adam. „Rethinking Atrous Convolution for Semantic Image Segmentation“. In: (2017).

[Che+17b] Ming Cheng, Haocheng Zhang, Cheng Wang und Jonathan Li. „Extraction and Classification of Road Markings Using Mobile Laser Scanning Point Clouds“. In: *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* 10.3 (2017), S. 1182–1196. ISSN: 1939-1404.

[Cho+15] François Chollet u. a. *Keras*. https://github.com/fchollet/keras. 2015.

[CI07] Emmanuel Christophe und Jordi Inglada. „Robust Road Extraction for High Resolution Satellite Images“. In: *IEEE Int. Conf. Image Process.* 1 (2007), S. V – 437–V –440.

[CMP00] Hung Chih Chiang, Randolph L. Moseswith und Lee C. Potter. „Model-based classification of radar images“. In: *IEEE Trans. Inf. Theory* 46.5 (2000), S. 1842–1854. ISSN: 00189448.

[Cor+03] D. G. Corr, A. Walker, U. Benz, I. Lingenfelder und A. Rodrigues. „Classification of urban SAR imagery using object oriented techniques“. In: *Int. Geosci. Remote Sens. Symp.* 1 (2003), S. 188–190.

[Cor+16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth und Bernt Schiele. „The Cityscapes Dataset for Semantic Urban Scene Understanding“. In: *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* (2016), S. 3213–3223. ISSN: 10636919.

[CV95] Corinna Cortes und Vladimir Vapnik. „Support-vector networks“. In: *Mach. Learn.* 20.3 (1995), S. 273–297. ISSN: 0885-6125.

[Dar59] Charles Darwin. *On the Origin of Species by Means of Natural Selection*. 1859. ISBN: 145381468X.

[DB15] Alexey Dosovitskiy und Thomas Brox. „Inverting Convolutional Networks with Convolutional Networks“. In: (2015), S. 1–15.

[DeJ75] K A DeJong. *Analysis of the Behavior of a Class of Genetic Adaptive Systems*. 1975.

[Det89] Jürgen Detlefsen. „Dauerstrichradar“. In: *Radartechnik: Grundlagen, Bauelemente, Verfahren, Anwendungen*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1989, S. 77–90. ISBN: 978-3-642-83600-8.

[DHS11] John Duchi, Elad Hazan und Yoram Singer. „Adaptive Subgradient Methods for Online Learning and Stochastic Optimization“. In: *J. Mach. Learn. Res.* 12 (2011), S. 2121–2159. ISSN: 15324435.

[DIN94] DIN Deutsches Institut für Normung e. V. *DIN 70000 Fahrzeugdynamik und Fahrverhalten*. 1994.

[DKL10] Michael Darms, Matthias Komar und Stefan Lueke. „Map based road boundary estimation“. In: *IEEE Intell. Veh. Symp. Proc.* (2010), S. 609–614. ISSN: 1931-0587.

[DLR77] A. P. Dempster, N. M. Laird und D. B. Rubin. *Maximum Likelihood from Incomplete Data via the EM Algorithm*. Bd. 39. 1. 1977, S. 1–38.

[DON11] Radu Danescu, Florin Oniga und Sergiu Nedevschi. „Modeling and tracking the driving environment with a particle-based occupancy grid“. In: *IEEE Trans. Intell. Transp. Syst.* 12.4 (2011), S. 1331–1342. ISSN: 15249050.

[DPA16] Johan Degerman, Thomas Pernstal und Klas Alenljung. „3D occupancy grid mapping using statistical radar models“. In: *IEEE Intell. Veh. Symp. Proc.* (2016), S. 902–908.

[Dub+14] Renaud Dube, Markus Hahn, Markus Schutz, Jürgen Dickmann und Denis Gingras. „Detection of parked vehicles from a radar based occupancy grid“. In: *Intell. Veh. Symp. Proc.* (2014), S. 1415–1420.

[Dun+12] Kerry E. Dungan, Joshua N. Ash, John W. Nehrbass, Jason T. Parker, LeRoy A. Gorham und Steven M. Scarborough. „Wide angle SAR data for target discrimination research“. In: *Algorithms Synth. Aperture Radar Imag. XIX*. Bd. 8394. 88. 2012, S. 83940M–83940M–13. ISBN: 9780819490728.

[DV16] Vincent Dumoulin und Francesco Visin. „A guide to convolution arithmetic for deep learning“. In: (2016), S. 1–28. ISSN: 16113349.

[DY13] Li Deng und Dong Yu. „Deep Learning: Methods and Applications“. In: *Found. Trends® Signal Process.* 7.3-4 (2013), S. 197–387. ISSN: 09598138.

[EKG08] M Enzweiler, P Kanter und D M Gavrila. „Monocular pedestrian recognition using motion parallax“. In: *IEEE Intell. Veh. Symp.* (2008), S. 792–797.

[Elf86] Alberto Elfes. „A sonar-based mapping and navigation system“. In: *Proceedings. 1986 IEEE Int. Conf. Robot. Autom.* 3.May 1986 (1986), S. 1151–1156.

[Elf89] Alberto Elfes. „Using Occupancy Grids for Mobile Robot Perception and Navigation“. In: *Computer (Long. Beach. Calif).* 22.6 (1989), S. 46–57. ISSN: 00189162.

[Enz+12] Markus Enzweiler, Matthias Hummel, David Pfeiffer und Uwe Franke. „Efficient stixel-based object recognition“. In: *IEEE Intell. Veh. Symp. Proc.* (2012), S. 1066–1071. ISSN: 1931-0587.

[EP03] Austin Eliazar und Ronald Parr. „DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks“. In: *IJCAI Int. Jt. Conf. Artif. Intell.* (2003), S. 1135–1142. ISSN: 10450823.

[Evt+18] Ivan Evtimov, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati und Dawn Song. „Robust Physical-World Attacks on Machine Learning Models“. In: *Conf. Comput. Vis. Pattern Recognit.* 2018.

[Fah+07] Ludwig Fahrmeir, Christian Heumann, Iris Pigeot, Rita Künstler und Gerhard Tutz. *Statistik*. Springer, Berlin, Heidelberg, 2007, S. 612. ISBN: 978-3-540-69713-8.

[Far+13] Clement Farabet, Camille Couprie, Laurent Najman und Yann Lecun. „Learning hierarchical features for scene labeling“. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 35.8 (2013), S. 1915–1929. ISSN: 01628828.

[FG09] Uwe Franke und Stefan Gehrig. „Stereosehen“. In: *Handb. Fahrerassistenzsysteme*. Springer Vieweg, Wiesbaden, 2009. ISBN: 978-3-658-05733-6.

[FKG13] Jannik Fritsch, Tobias Kuehnl und Andreas Geiger. „A new performance measure and evaluation benchmark for road detection algorithms“. In: *Int. Conf. Intell. Transp. Syst.* 28.Itsc (2013), S. 38–61.

[Foe01] Alex Foessel-Bunting. „Radar sensor model for three-dimensional map building“. In: *Proc. SPIE* 4195.412 (2001), S. 127–138. ISSN: 0277786X.

[FR95] Wt Freeman und Michal Roth. „Orientation histograms for hand gesture recognition“. In: *Int. Work. Autom. Face Gesture Recognit.* 12 (1995), S. 296–301.

[Fuk79] Kunihiko Fukushima. „Self-organization of a neural network which gives position-invariant response“. In: *6th Int. Jt. Conf. Artif. Intell.* Bd. 1. Morgan Kaufmann Publishers Inc. 1979, S. 291–293.

[Fuk80] Kunihiko Fukushima. „Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position“. In: *Biol. Cybern.* 202 (1980).

[Gar+01] D. A. Garren, M. K. Osborn, A. C. Odom, J. S. Goldstein, S. U. Pillai und J. R. Guerci. „Enhanced target detection and identification via optimised radar transmission pulse shape“. In: *IEE Proc. Radar, Sonar Navig.* 148.3 (2001), S. 130–138. ISSN: 13502395.

[GBB11] Xavier Glorot, Antoine Bordes und Yoshua Bengio. „Deep sparse rectifier neural networks“. In: *Proc. 14th Int. Conf. Artif. Intell. Stat.* 15 (2011), S. 315–323.

[GBC09] David Grangier, Leon Bottou und Ronan Collobert. „Deep Convolutional Networks for Scene Parsing“. In: *ICML 2009 Deep Learn. Work.* (2009).

[GBC16] Ian Goodfellow, Yoshua Bengio und Aaron Courville. *Deep learning*. 2016.

[Gei+13] A Geiger, P Lenz, C Stiller und R Urtasun. „Vision meets robotics: The KITTI dataset“. In: *Int. J. Rob. Res.* 32.11 (2013), S. 1231–1237. ISSN: 0278-3649.

[Gér17] Aurélien Géron. *Hands-on machine learning with scikit-learn & tensorflow*. O'Reilly Media, 2017, S. 718. ISBN: 9781491962299.

[GHS00a] Paolo Gamba, Bijan Houshmand und Matteo Saccani. „Detection and extraction of buildings from interferometric SAR data“. In: *IEEE Trans. Geosci. Remote Sens.* 38.1 II (2000), S. 611–618. ISSN: 01962892.

[GHS00b] Paolo Gamba, Bijan Houshmand und Matteo Saccani. „Detection and extraction of buildings from interferometric SAR data“. In: *IEEE Trans. Geosci. Remote Sens.* 38.1 II (2000), S. 611–618. ISSN: 01962892.

[Gie+17] T Giese, J Klappstein, J Dickmann und C Wöhler. „Road course estimation using deep learning on radar data“. In: *18th Int. Radar Symp.* (2017), S. 1–7. ISSN: 21555753.

[GLU12] Andreas Geiger, Philip Lenz und Raquel Urtasun. „Are we ready for autonomous driving? the KITTI vision benchmark suite“. In: *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 2012, S. 3354–3361. ISBN: 9781467312264.

[Gre11] Jens Grenzhäuser. „Entwicklung neuartiger Mess- und Auswertungsstrategien ein scannendes Wolkenradar und deren Anwendungsbereiche“. Diss. Karlsruher Institut für Technologie (KIT), 2011. ISBN: 9783866447752.

[Guo+16] Yanming Guo, Yu Liu, Ard Oerlemans, Songyang Lao, Song Wu und Michael S. Lew. „Deep learning for visual understanding: A review“. In: *Neurocomputing* 187 (2016), S. 27–48.

[Hah+00] Richard H.R. Hahnioser, Rahul Sarpeshkar, Misha A. Mahowald, Rodney J. Douglas und H. Sebastian Seung. „Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit“. In: *Nature* 405.6789 (2000), S. 947–951. ISSN: 00280836.

[HBD89] John W. Hager, Jemes F. Behensky und Brad W. Drew. *The Universal Grids: Universal traverse Mercator and Universal Polar Stereographic.* 1989.

[HHS06] Karin Hedman, S Hinz und U Stilla. „A Probabilistic Fusion Strategy Applied to Road Extraction From Multi-Aspect SAR Data“. In: *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* (2006), S. 55–60.

[HOT06] Geoffrey E. Hinton, Simon Osindero und Yee-Whye Teh. „A Fast Learning Algorithm for Deep Belief Nets“. In: *Neural Comput.* 18.7 (2006), S. 1527–54.

[How+17] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto und Hartwig Adam. „MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications“. In: (2017). ISSN: 15071367.

[HR10] Steffen Heuel und Hermann Rohling. „Pedestrian recognition based on 24 GHz radar sensors“. In: *Radar Symp. (IRS), 2010 11th Int.* (2010), S. 1–6. ISSN: 2155-5754.

[HR11] Steffen Heuel und Hermann Rohling. „Two-stage pedestrian classification in automotive radar systems“. In: *2011 12th Int. Radar Symp.* (2011), S. 477–484.

[HR12] Steffen Heuel und Hermann Rohling. „Pedestrian classification in automotive radar systems“. In: *Proc. Int. Radar Symp.* (2012), S. 39–44. ISSN: 21555753.

[HR13] Steffen Heuel und Hermann Rohling. „Pedestrian recognition in automotive radar sensors“. In: *Radar Symp. (IRS), 2013 14th.* 2013.

[HS06] Geoffrey E. Hinton und R. R. Salakhutdinov. „Reducing the Dimensionality of Data with Neural Networks“. In: *Science (80-.).* 313.5786 (2006), S. 504–507. ISSN: 0036-8075.

[HS97] Sepp Hochreiter und Jürgen Schmidhuber. „Long Short-Term Memory“. In: *Neural Comput.* 9.8 (1997), S. 1735–1780.

[HTF09] Trevor Hastie, Robert Tibshirani und Jerome Friedman. *The Elements of Statistical Learning*. Bd. 1. Springer, 2009, S. 305–317. ISBN: 0387848576.

[Hül04] Christian Hülsmeyer. „Verfahren, um entfernte metallische Gegenstände mittels elektrischer Wellen einem Beobachter zu melden“. Patent DE165546. 1904.

[IS15] Sergey Ioffe und Christian Szegedy. „Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift“. In: *Arxiv* (2015), S. 1–11. ISSN: 0717-6163.

[Jac12] Paul Jaccard. „The Distribution of the Flora in the Alpine Zone.“ In: *New Phytol.* 11.2 (1912), S. 37–50. ISSN: 14698137.

[Jia+05] Cheng Li Jia, Ke Feng Ji, Yong Mei Jiang und Gang Yao Kuang. „Road extraction from high-resolution SAR imagery using Hough Transform“. In: *Int. Geosci. Remote Sens. Symp.* 1 (2005), S. 336–339. ISSN: 10012486.

[Jia+09] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li und Li Fei-Fei. „ImageNet: A large-scale hierarchical image database“. In: *CVPR*. 2009, S. 248–255. ISBN: 978-1-4244-3992-8.

[Jia+14] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama und Trevor Darrell. „Caffe: Convolutional architecture for fast feature embedding“. In: *MM 2014 - Proc. 2014 ACM Conf. Multimed.* (2014), S. 675–678.

[JM97] Cullen Jennings und Don Murray. „Stereo Vision based Mapping and Navigation for Mobile Robots“. In: *IEEE Int. Conf. Robot. Autom.* 2 (1997), S. 1694–1699.

[Jou+17] Norman P. Jouppi, Al Borchers, Rick Boyle, Pierre-luc Cantin, Clifford Chao, Chris Clark, Jeremy Coriell, Mike Daley, Matt Dau, Jeffrey Dean, Ben Gelb, Cliff Young, Tara Vazir Ghaemmaghami, Rajendra Gottipati, William Gulland, Robert Hagmann, C. Richard Ho, Doug Hogberg, John Hu, Robert Hundt, Dan Hurt, Julian Ibarz, Nishant Patil, Aaron Jaffey, Alek Jaworski, Alexander Kaplan, Harshit Khaitan, Daniel Killebrew, Andy Koch, Naveen Kumar, Steve Lacy, James Laudon, James Law, David Patterson, Diemthu Le, Chris Leary, Zhuyuan Liu, Kyle Lucke, Alan Lundin, Gordon MacKean, Adriana Maggiore, Maire Mahony, Kieran Miller, Rahul Nagarajan, Gaurav Agrawal, Ravi Narayanaswami, Ray Ni, Kathy Nix, Thomas Norrie, Mark Omernick,

Narayana Penukonda, Andy Phelps, Jonathan Ross, Matt Ross, Amir Salek, Raminder Bajwa, Suresh Bhatia, Emad Boden, NanSamadiani, Chris Severn, Gregory Sizikov, Matthew Snelham, Jed Souter, Dan Steinberg, Andy Swing, Mercedes Tan, Gregory Thorson, Bo Tian, Sarah Bates, Horia Toma, Erick Tuttle, Vijay Vasudevan, Richard Walter, Walter Wang, Eric Wilcox und Doe Hyun Yoon. „In-Datacenter Performance Analysis of a Tensor Processing Unit“. In: *ACM SIGARCH Comput. Archit. News* 45.2 (2017), S. 1–12. ISSN: 01635964.

[Kär+09] Camilla Kärnfelt, Alain Péden, Ali Bazzi, Ghayath El Haj Shhadé, Mohamad Abbas, Thierry Chonavel und Frantz Bodereau. „77 GHz ACC radar simulation platform“. In: *9th Int. Conf. Intell. Transp. Syst. Telecommun.* 4 (2009), S. 209–214.

[Kha+14] Salman Hameed Khan, Mohammed Bennamoun, Ferdous Sohel und Roberto Togneri. „Geometry driven semantic labeling of indoor scenes“. In: *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. 2014. ISBN: 9783319105895.

[KHG11] Christoph G. Keller, Christoph Hermes und Dariu M. Gavrila. „Will the Pedestrian Cross? Probabilistic Path Prediction Based on Learned Motion Features“. In: *Mester*. Bd. 6835 LNCS. 2011, S. 386–395.

[Kil16] Kilian Laudt. „Scene Classification On Radar Based Grid Maps Using-Convolutional Neural Networks“. M.Sc. thesis. Universität Koblenz-Landau, 2016.

[KL14] Diederik Kingma und Jimmy Lei Ba. „Adam: A Method for Stochastic Optimization“. In: *Int. Conf. Learn. Represent.* (2014), S. 1–13. ISSN: 09252312.

[KLK02] Vladimir Katkovnik, Moon Sik Lee und Yong Hoon Kim. „High-resolution signal processing for a switch antenna array FMCW radar with a single channel receiver“. In: *Proc. IEEE Sens. Array Multichannel Signal Process. Work.* Bd. 2002-Janua. 2002, S. 543–547. ISBN: 0780375513.

[Kno08] Eugene F. Knott. „Radar Cross Section“. In: *Radar Handb.* 2008.

[Kon97] Kurt Konolige. „Improved Occupancy Grids for Map Building“. In: *Auton. Robots* 4.4 (1997), S. 351–367.

[Koo+14] Julian Francisco Pieter Kooij, Nicolas Schneider, Fabian Flohr und Dariu M Gavrila. „Context-Based Pedestrian Path Prediction“. In: Hrsg. von David Fleet, Tomas Pajdla, Bernt Schiele und Tinne Tuytelaars. Cham: Springer International Publishing, 2014, S. 618–633. ISBN: 978-3-319-10599-4.

[Kou10] Guy Kouemou. „Radar Target Classification Technologies“. In: *Radar Technol.* December. INTECH Open Access Publisher, 2010. ISBN: 9789533070292.

[Kra17] Oliver Kramer. *Genetic Algorithm Essentials*. Bd. 679. Springer, 2017. ISBN: 978-3-319-52155-8.

[KSD10] Marcus Konrad, Magdalena Szczot und Klaus Dietmayer. „Road course estimation in occupancy grids“. In: *IEEE Intell. Veh. Symp. Proc.* 2010, S. 412–417. ISBN: 9781424478668.

[KSH12] Alex Krizhevsky, Ilya Sutskever und Geoffrey E Hinton. „ImageNet Classification with Deep Convolutional Neural Networks“. In: *Adv. Neural Inf. Process. Syst.* (2012), S. 1–9. ISSN: 10495258.

[Kun14] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms: Second Edition*. Bd. 9781118315. 2014, S. 1–357. ISBN: 9781118914564.

[Lar+17] Amir Laribi, Markus Hahn, J. Dickmann und Christian Waldschmidt. „Vertical Digital Beamforming Versus Multipath Height Finding“. In: (2017), S. 1197–1202.

[LC98] Yann LeCun und Corinna Cortes. *The MNIST database of handwritten digits*. 1998.

[Le +90] Y. Le Cun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard und L. D. Jackel. „Handwritten Digit Recognition with a Back-Propagation Network“. In: *Adv. Neural Inf. Process. Syst.* (1990), S. 396–404. ISSN: 1524-4725.

[LeC+89] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard und L. D. Jackel. *Backpropagation Applied to Handwritten Zip Code Recognition*. 1989.

[Lec+98] Yann Lecun, Léon Bottou, Yoshua Bengio und Patrick Haffner. „Gradient-Based Learning Applied to Document Recognition“. In: *Proc. IEEE*. 1998, S. 2278–2324.

[LHB04] Y. LeCun, Fu Jie Huang und L. Bottou. „Learning methods for generic object recognition with invariance to pose and lighting“. In: *Proc. 2004 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognition, 2004. CVPR 2004.* 2 (2004), S. 97–104. ISSN: 1063-6919.

[Li+01] L. Li, S. M. Sekelsky, S. C. Reising, C. T. Swift, S. L. Durden, G. A. Sadowy, S. J. Dinardo, F. K. Li, A. Huffman, G. Stephens, D. M. Babb und H. W. Rosenberger. „Retrieval of atmospheric attenuation using combined ground-based and airbone 95-GHz cloud radar measurements“. In: *J. Atmos. Ocean. Technol.* 18.8 (2001), S. 1345–1353. ISSN: 07390572.

[Li+16] Yadi Li, Liguo Chen, Haibo Huang, Xiangpeng Li, Wenkui Xu, Liang Zheng und Jiaqi Huang. „Nighttime lane markings recognition based on Canny detection and Hough transform“. In: *2016 IEEE Int. Conf. Real-Time Comput. Robot. RCAR 2016* (2016), S. 411–415.

[Lom+15] Jakob Lombacher, Markus Hahn, Jürgen Dickmann und Christian Wöhler. „Detection of arbitrarily rotated parked cars based on radar sensors“. In: *Proc. Int. Radar Symp.* 2015. ISBN: 9783954048533.

[Lom+16] Jakob Lombacher, Markus Hahn, Jürgen Dickmann und Christian Wöhler. „Potential of radar for static object classification using deep learning methods“. In: *2016 IEEE MTT-S Int. Conf. Microwaves Intell. Mobility, ICMIM 2016*. 2016. ISBN: 9781509023677.

[Lom+17a] Jakob Lombacher, Markus Hahn, Jürgen Dickmann und Christian Wöhler. „Object Classification in Radar Using Ensemble Methods“. In: *IEEE MTT-S Int. Conf. Microwaves Intell. Mobil.* 2017. ISBN: 9781509043545.

[Lom+17b] Jakob Lombacher, Kilian Laudt, Markus Hahn, Jurgen Dickmann und Christian Wöhler. „Semantic radar grids“. In: *IEEE Intell. Veh. Symp. Proc.* 2017, S. 1170–1175. ISBN: 9781509048045.

[Low04] David G Lowe. *US6711293B1 - Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image*. 2004.

[Low99] David G. Lowe. „Object recognition from local scale-invariant features“. In: *Proc. IEEE Int. Conf. Comput. Vis.* 2 (1999), S. 1150–1157.

[LPH93] Ren C. Luo, Harsh Potlapalli und David Hislop. „Traffic sign recognition in outdoor environments using reconfigurable neural networks“. In: *Proc. Int. Jt. Conf. Neural Networks* 2 (1993), S. 1306–1309.

[LSD14] Jonathan Long, Evan Shelhamer und Trevor Darrell. „Fully Convolutional Networks for Semantic Segmentation“. In: *2015 IEEE Conf. Comput. Vis. Pattern Recognit.* (2014), S. 3431–3440. ISSN: 10636919.

[Luc17] Florin Lucaciu. „Design and implementation of modell-based method for the detection and orientation estimation of parked vehicles using radar sensors“. M.Sc. thesis. Universität Stuttgart, 2017.

[Lud98] Albrecht Ludloff. *Praxiswissen Radar und Radarsignalverarbeitung.* 1998.

[Lup+17] Stefanie Lupfer, Matthias Rapp, Klaus Dietmayer, Peter Brosseit, Jakob Lombacher, Markus Hahn und Jurgen Dickmann. „Increasing FastSLAM accuracy for radar data by integrating the Doppler information“. In: *2017 IEEE MTT-S Int. Conf. Microwaves Intell. Mobility, ICMIM 2017.* 2017, S. 103–106. ISBN: 9781509043545.

[LZX16] Bo Li, Tianlei Zhang und Tian Xia. „Vehicle detection from 3D lidar using fully convolutional network“. In: *Robot. Sci. Syst.* 12 (2016). ISSN: 2330765X.

[Mac67] James MacQueen. „Some Methods for classification and Analysis of Multivariate Observations“. In: *5th Berkeley Symp. Math. Stat. Probab. 1967.* Bd. 1. 14. 1967, S. 281–297. ISBN: 1595931619.

[Mad+17] Will Maddern, Geoffrey Pascoe, Chris Linegar und Paul Newman. „1 year, 1000 km: The Oxford RobotCar dataset“. In: *Int. J. Rob. Res.* 36.1 (2017), S. 3–15. ISSN: 0278-3649.

[Mar+17] Helmut Martin, Kurt Tschabuschnig, Olof Bridal und Daniel Watzenig. *Automated Driving.* Hrsg. von Daniel Watzenig und Martin Horn. Cham: Springer International Publishing, 2017, S. 387–416. ISBN: 978-3-319-31893-6.

[Mau+15] Markus Maurer, J Christian Gerdes, Barbara Lenz und Hermann Winner. *Autonomes Fahren. Technische, rechtliche und gesellschaftliche Aspekte.* Springer-Verlag, 2015. ISBN: 9783662458532.

[May99] Helmut Mayer. „Automatic Object Extraction from Aerial Imagery—A Survey Focusing on Buildings“. In: *Comput. Vis. Image Underst.* 74.2 (1999), S. 138–149. ISSN: 10773142.

[McC86] Robert K McConnell. „Method of and apparatus for pattern recognition“. Patent US4567610A. 1986.

[MG15] Moritz Menze und Andreas Geiger. „Object scene flow for autonomous vehicles“. In: *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* Bd. 07-12-June. 2015, S. 3061–3070. ISBN: 9781467369640.

[MH08] Laurens van der Maaten und Geoffrey Hinton. „Visualizing data using t-SNE“. In: *J. Mach. Learn. Res.* 9 (2008), S. 2579–2605.

[MK19] Michael Meyer und Georg Kuschk. „Automotive Radar Dataset for Deep Learning Based 3D Object Detection“. In: (2019), S. 3–6.

[Mol+13] Pavlo Molchanov, Alexey Vinel, Jaakko Astola und Karen Egiazarian. „Radar frequency band invariant pedestrian classification“. In: *Proc. Int. Radar Symp.* 2 (2013), S. 740–745. ISSN: 21555753.

[Mor96] H. Moravec. „Robot spatial perception by stereoscopic vision and 3d evidence grids“. In: *Perception* September (1996).

[MP43] Warren S. McCulloch und Walter Pitts. „A Logical Calculus of the Ideas Immanent in Nervous Activity“. In: *Bull. Math. Biophys.* 5.4 (1943), S. 115–133.

[MT04] Michael Montemerlo und Sebastian Thrun. „A multi-resolution pyramid for outdoor robot terrain perception“. In: *Proc. Natl. Conf. Artif. Intell.* 4 (2004), S. 464–469.

[Ned+04] Sergiu Nedevschi, Rolf Schmidt, Thorsten Graf, Radu Danescu, Dan Frentiu, Tiberiu Marita, Florin Oniga und Ciprian Pocol. „3D lane detection system based on stereovision“. In: *IEEE Conf. Intell. Transp. Syst. Proceedings, ITSC* (2004), S. 161–166.

[NHH15] Hyeonwoo Noh, Seunghoon Hong und Bohyung Han. „Learning deconvolution network for semantic segmentation“. In: *Proc. IEEE Int. Conf. Comput. Vis.* 2015 Inter (2015), S. 1520–1528. ISSN: 15505499.

[NOA91] NOAA National Weather Service (NWS) Radar Operations Center. *NOAA Next Generation Radar (NEXRAD) Level 2 Base Data*. 1991.

[NON94] Leslie M. Novak, Gregory J. Owirka und Christine M. Netishen. „Radar target identification using spatial matched filters“. In: *Pattern Recognit.* 27.4 (1994), S. 607–617. ISSN: 00313203.

[NS13] Christoph Neumann und Hermine Senkowski. „Automatic Target Recognition for short time observation radars“. In: *Proc. Int. Radar Symp.* 2 (2013), S. 593–596. ISSN: 21555753.

[Nus+18] Dominik Nuss, Stephan Reuter, Markus Thom, Ting Yuan, Gunther Krehl, Michael Maile, Axel Gern und Klaus Dietmayer. „A random finite set approach for dynamic occupancy grid maps with real-time application“. In: *Int. J. Rob. Res.* 37.8 (2018), S. 841–866. ISSN: 17413176.

[NYC16] Anh Nguyen, Jason Yosinski und Jeff Clune. „Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks“. In: *Arxiv* (2016), S. 23.

[NZ15] Ram M. Narayanan und Matthew Zenaldin. „Radar micro-Doppler signatures of various human activities“. In: *IET Radar, Sonar Navig.* 9.9 (2015), S. 1205–1215. ISSN: 17518784.

[OS96] David W Opitz und Jude W Shavlik. „Generating Accurate and Diverse Members of a Neural-Network Ensemble“. In: *Adv. Neural Inf. Process. Syst.* 8 (1996), S. 535–541. ISSN: 1049-5258.

[Pas+17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga und Adam Lerer. „Automatic differentiation in PyTorch“. In: NIPS 2017 Autodiff Workshop (2017), S. 1–4.

[Pav+99] I. Pavlidis, D. Perrin, N. P. Papanikolopoulos, W. Au und S. Sawtelle. „A ground truth tool for Synthetic Aperture Radar (SAR) imagery“. In: *Proc. - IEEE Work. Comput. Vis. Beyond Visible Spectr. Methods Appl. CVBVS 1999*. 1999, S. 82–87. ISBN: 0769500501.

[Ped+11] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot und Édouard Duchesnay. „Scikit-learn: Machine learning in Python“. In: *J. Mach. Learn. Res.* 12 (2011), S. 2825–2830. ISSN: 15324435.

[PF07] Daniele Perissin und Alessandro Ferretti. „Urban-Target Recognition by Means of Repeated Spaceborne SAR Images“. In: *IEEE Trans. Geosci. Remote Sens.* 45.12 (2007), S. 4043–4058. ISSN: 0196-2892.

[PLC00] JM Park, CG Looney und HC Chen. „Fast connected component labeling algorithm using a divide and conquer technique.“ In: *15th Int. Conf. Comput. their Appl.* 4 (2000), S. 4–7.

[PND98] Daniel Pagac, Eduardo M. Nebot und Hugh Durrant-Whyte. „An evidential approach to map-building for autonomous vehicles“. In: *IEEE Trans. Robot. Autom.* 14.4 (1998), S. 623–629. ISSN: 1042296X.

[Poh00] Hartmut Pohlheim. *Evolutionäre Algorithmen - Verfahren, Operatoren und Hinweise für die Praxis*. Springer-Verlag, 2000, S. 327. ISBN: 978-3-8351-0219-4.

[Pow11] David M W Powers. „Evaluation: From Precision, Recall and F-Measure To Roc, Informedness, Markedness & Correlation“. In: *J. Mach. Learn. Technol.* 2.1 (2011), S. 37–63.

[Pri+93] L. Priese, V. Rehrmann, R. Schian und R. Lakmann. „Traffic sign recognition based on color image evaluationion“. In: *IEEE Intell. Veh. Symp. Proc.* (1993), S. 95–100.

[Pro+17] Robert Prophet, Marcel Hoffmann, Martin Vossiek, Gang Li und Christian Sturm. „Parking space detection from a radar based target list“. In: *2017 IEEE MTT-S Int. Conf. Microwaves Intell. Mobility, ICMIM 2017* Icmim (2017), S. 91–94.

[Pro+18] Robert Prophet, Henriette Stark, Marcel Hoffmann, Christian Sturm und Martin Vossiek. „Adaptions for Automotive Radar Based Occupancy Gridmaps“. In: *2018 IEEE MTT-S Int. Conf. Microwaves Intell. Mobility, ICMIM 2018* Icmim (2018), S. 1–4.

[PS11] Florent Perronnin und Jorge Sánchez. *Compressed Fisher vectors for LSVR*. 2011.

[Qi+17] Charles R. Qi, Hao Su, Kaichun Mo und Leonidas J. Guibas. „PointNet: Deep learning on point sets for 3D classification and segmentation“. In: *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*. Bd. 2017-Janua. 2017, S. 77–85. ISBN: 9781538604571.

[Rap+15] Matthias Rapp, Tilmann Giese, Markus Hahn, Jurgen Dickmann und Klaus Dietmeyer. „A Feature-Based Approach for Group-Wise Grid Map Registration“. In: *IEEE Conf. Intell. Transp. Syst.* Bd. 2015-Octob. 2015, S. 511–516. ISBN: 9781467365956.

[Rap+16a] Matthias Rapp, Klaus Dietmayer, Markus Hahn, Bharanidhar Duraisamy und Jurgen Dickmann. „Hidden Markov model-based occupancy grid maps of dynamic environments“. In: *19th Int. Conf. Inf. Fusion, Proc.* (2016), S. 1780–1788.

[Rap+16b] Matthias Rapp, Klaus Dietmayer, Markus Hahn, Frank Schuster, Jakob Lombacher und Jürgen Dickmann. „FSCD and BASD: Robust landmark detection and description on radar-based grids“. In: *IEEE MTT-S Int. Conf. Microwaves Intell. Mobil.* 2016. ISBN: 9781509023677.

[RBK12] Grzegorz Rozenberg, Thomas Back und Joost N. Kok. *Handbook of Natural Computing*. Bd. 1-4. 2012, S. 1–2051. ISBN: 9783540929109.

[RFB15] Olaf Ronneberger, Philipp Fischer und Thomas Brox. „U-net: Convolutional networks for biomedical image segmentation“. In: *Med. Image Comput. Comput. Interv. - Lect. Notes Comput. Sci.* Bd. 9351. 2015, S. 234–241. ISBN: 9783319245737.

[Ric+09] Eric Richter, Philipp Lindner, Gerd Wanielik, Kiyokazu Takagi und Akira Isogai. „Advanced occupancy grid techniques for lidar based object detection and tracking“. In: *IEEE Conf. Intell. Transp. Syst. Proc.* (2009), S. 450–454.

[Rit92] W. Ritter. „Traffic sign recognition in color image sequences“. In: *IEEE Intell. Veh. Symp.* (1992), S. 12–17.

[RMU12] Giulio Reina, Annalisa Milella und James Underwood. „Self-learning classification of radar features for scene understanding“. In: *Rob. Auton. Syst.* 60.11 (2012), S. 1377–1388. ISSN: 09218890.

[Roh11] Hermann Rohling. „Ordered statistic CFAR technique - An overview“. In: *Int. Radar Symp.* (2011), S. 631–638.

[Roh83] Hermann Rohling. „Radar CFAR Thresholding in Clutter and Multiple Target Situations“. In: *IEEE Trans. Aerosp. Electron. Syst.* AES-19.4 (1983), S. 608–621. ISSN: 00189251.

[Rok09] Lior Rokach. *Pattern Classification Using Ensemble Methods*. World Scientific Publishing Company, 2009, S. 244. ISBN: 978-981-4271-06-6.

[Rom17] Marouane Ben Romdhane. „Spatio-temporal Classification of Radar Grid Maps“. M.Sc. thesis. Technische Universität München, 2017.

[Ros+98] Timothy Ross, Stephen Worrell, Vincent Velten, John Mossing und Michael Bryant. „Standard SAR ATR evaluation experiments using the MSTAR public release data set“. In: *SPIE Conf. Algorithms Synth. Aperture Radar Imag. V*. 1998.

[RP01] M. Ribo und A. Pinz. „A comparison of three uncertainty calculi for building sonar-based occupancy grids“. In: *Rob. Auton. Syst.* 35.3-4 (2001), S. 201–209. ISSN: 09218890.

[RS08] Hans Albert Richard und Manuela Sander. „Kinematik des starren Körpers.“ In: *Tech. Mech. Dynamik.* 2008.

[RSH10] Mark A. Richards, James A. Scheer und William A. Holm. *Principles of modern radar: Basic principles*. SciTech Publishing, 2010, S. 1–925. ISBN: 9781613531488.

[Rus+15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg und Li Fei-Fei. „ImageNet Large Scale Visual Recognition Challenge“. In: *Int. J. Comput. Vis.* 115.3 (2015), S. 211–252. ISSN: 15731405.

[RW16] Atiqur Rahman und Yang Wang. „Optimizing Intersection-Over-Union in Deep Neural Networks for Image Segmentation“. In: *Bebis G. al. Adv. Vis. Comput. ISVC*. 2016.

[Sab+04] Kohtaro Sabe, Masaki Fukuchi, Jens Steffen Gutmann, Takeshi Ohashi, Kenta Kawamoto und Takayuki Yoshigahara. „Obstacle avoidance and path planning for humanoid robots using stereo vision“. In: *Proc. - IEEE Int. Conf. Robot. Autom.* Bd. 2004. 1. 2004, S. 592–597. ISBN: 0780382323.

[Sar+11] F. Sarholz, J. Mehnert, J. Klappstein, J. Dickmann und B. Radig. „Evaluation of different approaches for road course estimation using imaging radar“. In: *IEEE/RSJ Int. Conf. Intell. Robot. Syst.* 2011, S. 4587–4592. ISBN: 978-1-61284-456-5.

[SBS07] Robert Schneider, Hans-Ludwig Blöcher und Karl M. Strohm. „KOKON – Automotive High Frequency Technology at 77/79 GHz“. In: *Proc. 37th Eur. Microw. Conf.* January 2005. 2007, S. 1526–1529. ISBN: 9782874870019.

[SBW08] Karin Schuler, Denis Becker und Werner Wiesbeck. „Extraction of virtual scattering centers of vehicles by ray-tracing simulations“. In: *IEEE Trans. Antennas Propag.* 56.11 (2008), S. 3543–3551. ISSN: 0018926X.

[Sch+13a] Timo Scharwächter, Markus Enzweiler, Uwe Franke und Stefan Roth. „Efficient Multi-cue Scene Segmentation“. English. In: *Pattern Recognition. GCPR. Lect. Notes Comput. Sci.* Hrsg. von Joachim Weickert, Matthias Hein und Bernt Schiele. Bd. 8142. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, S. 435–445.

[Sch+13b] Eugen Schubert, Martin Kunert, Wolfgang Menzel, Joaquim Fortuny-Guasch und Jean Marc Chareau. „Human RCS measurements and dummy requirements for the assessment of radar based active pedestrian safety systems“. In: *Proc. Int. Radar Symp.* 2 (2013), S. 752–757. ISSN: 21555753.

[Sch+14a] Markus Schutz, Nils Appenrodt, Jurgen Dickmann und Klaus Dietmayer. „Occupancy grid map-based extended object tracking“. In: *IEEE Intell. Veh. Symp. Proc.* Iv (2014), S. 1205–1210.

[Sch+14b] Markus Schütz, Nils Appenrodt, Jürgen Dickmann und Klaus Dietmayer. „Multiple extended objects tracking with object-local occupancy grid maps“. In: *FUSION 2014 - 17th Int. Conf. Inf. Fusion* (2014), S. 1–7.

[Sch+15] Eugen Schubert, Frank Meinl, Martin Kunert und Wolfgang Menzel. „High resolution automotive radar measurements of vulnerable road users - pedestrians & cyclists“. In: *2015 IEEE MTT-S Int. Conf. Microwaves Intell. Mobility, ICMIM 2015*. 2015. ISBN: 9781479972159.

[Sch+17a] Markus Schoen, Markus Horn, Markus Hahn und Juergen Dickmann. „Real-Time Radar SLAM“. In: *11. Work. Fahrerassistenzsysteme und Autom. Fahr.* 2017, S. 1–10.

[Sch+17b] Ole Schumann, Markus Hahn, Jurgen Dickmann und Christian Wöhler. „Comparison of random forest and long short-term memory network performances in classification tasks using radar“. In: *Symp. Sens. Data Fusion.* Bd. 2017-Decem. IEEE, 2017, S. 1–6. ISBN: 9781538631034.

[Sch+18a] Ole Schumann, Markus Hahn, Jurgen Dickmann und Christian Wohler. „Supervised Clustering for Radar Applications: On the Way to Radar Instance Segmentation“. In: *IEEE MTT-S Int. Conf. Microwaves Intell. Mobil.* (2018).

[Sch+18b] Ole Schumann, Markus Hahn, Jürgen Dickmann und Christian Wöhler. „Semantic Segmentation on Radar Point Clouds“. In: *21st Int. Conf. Inf. Fusion.* IEEE, 2018, S. 2179–2186. ISBN: 9780996452762.

[Sch+19] Ole Schumann, Jakob Lombacher, Markus Hahn, Christian Wohler und Jürgen Dickmann. „Scene Understanding with Automotive Radar“. In: *IEEE Trans. Intell. Veh.* 8858.c (2019), S. 1. ISSN: 2379-8858 VO -.

[Sch15] Jürgen Schmidhuber. „Deep Learning in neural networks: An overview“. In: *Neural Networks* 61 (2015), S. 85–117. ISSN: 18792782.

[SD07] S. N. Sivanandam und S. N. Deepa. *Introduction to Genetic Algorithms*. Springer Science & Business Media, 2007, S. 461. ISBN: 9783540731900.

[SDS01] J. Sparbert, K. Dietmayer und D. Streller. „Lane detection and street type classification using laser range images“. In: *IEEE Conf. Intell. Transp. Syst. Proceedings, ITSC* (2001), S. 454–459.

[Ser15] Alexander Serguchev. „Design and implementation of model-based methods for the detection and orientation estimation of parked vehicles using radar sensors“. M.Sc. thesis. Universität Ulm, 2015.

[Si 94] Si Wei Lu. „Recognition of traffic signs using a multilayer neural network“. In: *1994 Proc. Can. Conf. Electr. Comput. Eng.* 1994.

[Sim13] Dan Simon. *Evolutionary Optimization Algorithms*. Wiley, 2013. ISBN: 978-0470937419.

[SJC08] Jamie Shotton, Matthew Johnson und Roberto Cipolla. „Semantic texton forests for image categorization and segmentation“. In: *26th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR*. 2008. ISBN: 9781424422432.

[Sko08] Merrill Ivan Skolnik. *Radar Handbook*. 2008. ISBN: 9780071485470.

[Sle+19] Liat Sless, Gilad Cohen, Bat El Shlomo und Shaul Oron. „Self Supervised Occupancy Grid Learning from Sparse Radar for Autonomous Driving“. In: 2019, S. 9.

[SOG05] Elisabeth Simonetto, Hélène Oriot und René Garello. „Rectangular building extraction from stereoscopic airborne radar images“. In: *IEEE Trans. Geosci. Remote Sens.* 43.10 (2005), S. 2386–2395. ISSN: 01962892.

[Sri+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever und Ruslan Salakhutdinov. „Dropout : A Simple Way to Prevent Neural Networks from Overfitting“. In: *J. Mach. Learn. Res.* 15 (2014), S. 1929–1958. ISSN: 15337928.

[Suc+17] Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O. Stanley und Jeff Clune. „Deep Neuroevolution: Genetic Algorithms Are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning“. In: (2017). ISSN: 07387946.

[SWB10] Graeme E. Smith, Karl Woodbridge und Chris J. Baker. „Radar micro-doppler signature classification using dynamic time warping“. In: *IEEE Trans. Aerosp. Electron. Syst.* 46.3 (2010), S. 1078–1096. ISSN: 00189251.

[Swe60] P. Swerling. „Probability of Detection for Fluctuating Targets“. In: *IRE Trans. Inf. Theory* 6.2 (1960), S. 269–308. ISSN: 21682712.

[SY07] Michael Schoor und Bin Yang. „High-resolution angle estimation for an automotive FMCW radar sensor“. In: *Proc. Int. Radar Symp.* 2007-Janua (2007), S. 1–5. ISSN: 21555753.

[SZ15] Karen Simonyan und Andrew Zisserman. „Very deep convolutional networks for large-scale image recognition“. In: *3rd Int. Conf. Learn. Represent. ICLR 2015*. 2015, S. 1–14.

[Sze+13] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow und Rob Fergus. „Intriguing properties of neural networks“. In: *arXiv Prepr. arXiv …* (2013), S. 1–10.

[Sze+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke und Andrew Rabinovich. „Going deeper with convolutions“. In: *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 07-12-June (2015), S. 1–9. ISSN: 10636919.

[Sze+16] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens und Zbigniew Wojna. „Rethinking the Inception Architecture for Computer Vision“. In: *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 2016-Decem (2016), S. 2818–2826. ISSN: 10636919.

[Taj+16] Nima Tajbakhsh, Jae Y. Shin, Suryakanth R. Gurudu, R. Todd Hurst, Christopher B. Kendall, Michael B. Gotway und Jianming Liang. „Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?“ In: *IEEE Trans. Med. Imaging* 35.5 (2016), S. 1299–1312. ISSN: 0278-0062.

[TBF00] S Thrun, W Burgard und D Fox. „A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping“. In: *Robot. Autom. 2000. Proceedings. ICRA '00. IEEE Int. Conf.* 1.April (2000), 321–328 vol.1.

[TGL09] Quoc Bao Truong, Heo Nam Geon und Byung Ryong Lee. „Vehicle detection and recognition for automated guided vehicle“. In: *ICROS-SICE Int. Jt. Conf. 2009, Proc.* (2009), S. 671–676.

[The16] Theano Development Team. „Theano: A Python framework for fast computation of mathematical expressions“. In: *arXiv e-prints* (2016), S. 19.

[The49] The New York Times. *Speeders in Connecticut to Face Real Radar Test.* 1949.

[Thr02] Sebastian Thrun. *Probabilistic robotics.* Bd. 45. Intelligent robotics and autonomous agents 3. MIT Press, 2002, S. 52–57. ISBN: 9780262201629.

[Thr03] Sebastian Thrun. „Learning occupancy grid maps with forward sensor models“. In: *Auton. Robots* 15.2 (2003), S. 111–127. ISSN: 09295593.

[Thr98] Sebastian Thrun. „Learning metric-topological maps for indoor mobile robot navigation“. In: *Artif. Intell.* 99.1 (1998), S. 21–71. ISSN: 00043702.

[TLT11] Alex Teichman, Jesse Levinson und Sebastian Thrun. „Towards 3D object recognition via classification of arbitrary object tracks“. In: *Proc. - IEEE Int. Conf. Robot. Autom.* (2011), S. 4034–4041. ISSN: 10504729.

[TN96] Arata Takahashi und Yoshiki Ninomiya. „Model-based lane recognition“. In: *IEEE Intell. Veh. Symp. Proc.* (1996), S. 201–206.

[Tou83] Godfried T. Toussaint. „Solving Geometric Problems With the 'Rotating Calipers'.“ In: *IEEE Melecon 83* 1.May (1983), S. 1–8. ISSN: 09507671.

[Ura+07] Ildar Urazghildiiev, Rolf Ragnarsson, Pierre Ridderström, Anders Rydberg, Eric Öjefors, Kjell Wallin, Per Enochsson, Magnus Ericson und Göran Löfqvist. „Vehicle classification based on the radar measurement of height profiles“. In: *IEEE Trans. Intell. Transp. Syst.* 8.2 (2007), S. 245–253. ISSN: 15249050.

[Vig+09] L. Vignaud, A. Ghaleb, J. Le Kernec und J. M. Nicolas. „Radar high resolution range & micro-Doppler analysis of human motions“. In: *2009 Int. Radar Conf. "Surveillance a Safer World", RADAR 2009* July (2009), S. 1–6.

[Wag+13] Raimar Wagner, Markus Thom, Roland Schweiger, Gunther Palm und Albrecht Rothermel. „Learning convolutional neural networks from few samples“. In: *Proc. Int. Jt. Conf. Neural Networks* (2013). ISSN: 2161-4393.

[Wei15] Karsten Weicker. *Evolutionäre Algorithmen*. Teubner, 2015. ISBN: 9783519003625.

[Wel18] Tobias Welz. „Extracting Sematic Radar Feature Grids using Recurrent Deep Convolutional Neural Networks“. M.Sc. thesis. Universität Ulm, 2018.

[Wer+14] Klaudius Werber, Michael Barjenbruch, Jens Klappstein, Jurgen Dickmann und Christian Waldschmidt. „How do traffic signs look like in radar?“ In: *44th Eur. Microw. Conf.* IEEE, 2014, S. 135–138. ISBN: 978-2-8748-7035-4.

[Wer+15a] Klaudius Werber, Michael Barjenbruch, Jens Klappstein, Jurgen Dickmann und Christian Waldschmidt. „RoughCough - A new image registration method for radar based vehicle self-localization“. In: *18th Int. Conf. Inf. Fusion*. 2015, S. 1533–1541. ISBN: 9780982443866.

[Wer+15b] Klaudius Werber, Matthias Rapp, Jens Klappstein, Markus Hahn, Jürgen Dickmann, Klaus Dietmayer und Christian Waldschmidt. „Automotive radar gridmap representations“. In: *2015 IEEE MTT-S Int. Conf. Microwaves Intell. Mobil.* 1. 2015. ISBN: 9781479972159.

[WHW09] Hermann Winner, Stephan Hakuli und Gabriele Wolf. *Handbuch Fahrerassistenzsysteme*. 2009, S. 719. ISBN: 978-3-658-05733-6.

[Win+17] Timo Winterling, Jakob Lombacher, Markus Hahn, Jürgen Dickmann und Christian Wöhler. „Optimizing Labelling on Radar Based Grid Maps Using Active Learning“. In: *IEEE MTT-S Int. Conf. Microwaves Intell. Mobil.* 2017.

[Win16] Timo Winterling. „Semi Supervised Classification of Radar Based Grid Maps Using Deep Learning Methods“. M.Sc. thesis. Universität Ulm, 2016.

[WM12] Darrell Whitley und Andrew M. Sutton. „Genetic Algorithms — A Survey of Models and Methods“. In: *Handb. Nat. Comput.* 2012, S. 637–671. ISBN: 978-3-540-92909-3.

[WSD07] Thorsten Weiss, Bruno Schiele und Klaus Dietmayer. „Robust driving path detection in urban and highway scenarios using a laser scanner and online occupancy grids“. In: *IEEE Intell. Veh. Symp. Proc.* (2007), S. 184–189. ISSN: 1931-0587.

[YK16] Fisher Yu und Vladlen Koltun. „Multi-scale context aggregation by dilated convolutions“. In: *4th Int. Conf. Learn. Represent. ICLR 2016 - Conf. Track Proc.* 2016.

[YQL05] Yinan Yang, Yuxia Qiu und Chao Lu. „Automatic target classification experiments on the MSTAR SAR images“. In: *Proc. - Sixth Int. Conf. Softw. Eng., Artif. Intell. Netw. Parallel/Distributed Comput. First ACIS Int. Work. Self-Assembling Wirel. Netw., SNPD/SAWN 2005* 2005 (2005), S. 2–7.

[YTN05] Naoyuki Yamada, Yuichi Tanaka und Kunitoshi Nishikawa. „Radar cross section for pedestrian in 76GHz band“. In: *35th Eur. Microw. Conf. 2005 - Conf. Proc.* 2.4 (2005), S. 1015–1018.

[ZB96] Anthony Zyweck und Robert E. Bogner. „Radar target classification of commercial aircraft“. In: *IEEE Trans. Aerosp. Electron. Syst.* 32.2 (1996), S. 598–606. ISSN: 0018-9251.

[Zei+10] Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor und Rob Fergus. „Deconvolutional networks“. In: *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* (2010), S. 2528–2535. ISSN: 10636919.

[Zei12] Matthew D. Zeiler. „ADADELTA: An Adaptive Learning Rate Method“. In: *arXiv* (2012), S. 6.

[ZF14] Matthew D. Zeiler und Rob Fergus. „Visualizing and understanding convolutional networks“. In: *Comput. Vis. – ECCV 2014* 8689 LNCS.PART 1 (2014), S. 818–833. ISSN: 16113349.

[Zha+17] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang und Jiaya Jia. „Pyramid scene parsing network“. In: *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017* 2017-Janua (2017), S. 6230–6239. ISSN: 0162-8828.

[Zha16] Xiaofeng Zhang. „Dynamic Object Classification Based on High Resolution Doppler Radar and Laser Data“. M.Sc. thesis. Universität Stuttgart, 2016.

[Zha99] Yun Zhang. „Optimisation of building detection in satellite images by combining multispectral classification and texture filtering“. In: *ISPRS J. Photogramm. Remote Sens.* 54.1 (1999), S. 50–60. ISSN: 09242716.

[ZLC17] Qun Zhang, Ying Luo und Yong-An Chen. *Micro-Doppler Characteristics of Radar Targets*. Butterworth-Heinemann, 2017. ISBN: 978-0-12-809861-5.